{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a carrier for the far-end supply position, host primitive content, total truth, object-layer T, exportable cannot-claim discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gap-inscription handoff, external-supply boundary, negative-certificate boundary, Nontrivial instance, FieldFaithful instance, transport, replay, provenance, and local naming for BEDC.Derived.ApophaticTLoopUp.

Constructive story: An $\mathsf{ApophaticTLoop}^{\uparrow}$ packet is generated from records-side history, classifier, reality-constrained meaning, reality constraint, gap, inscription, loop ledger, external-supply boundary, negative-certificate boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6381

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ packet records the regular Cauchy envelope by which rational schedules enter the public real-name boundary without a quotient stream or choice-selected limit. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, and the dyadic tolerance core of chapter 291. The packet names only finite windows, regularity witnesses, dyadic tolerances, terminal sealing, transport, replay, provenance, and a local certificate; it does not allocate quotient streams, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient completeness.

Definition 6381.1 (Bishop regular Cauchy envelope carrier)**.** *A* $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the rational observations available to the envelope.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational schedule row attached to the same displayed windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic precision and comparison budget consumed by* R*.*
4. S *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* W, R, D *have all been displayed.*
5. H *stores componentwise* hsame *transport for the window, regular schedule, dyadic tolerance, terminal seal, replay, provenance, and local naming rows.*
6. C *stores displayed* Cont *replay from finite window inspection through regularity, tolerance discharge, and terminal sealing.*
7. P *is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through W, R, D, S, H, C, P, N. It has no coordinate for a quotient stream, selected representative, host limit, countable-choice schedule, trichotomy decision, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Theorem 6381.2 (Bishop regular Cauchy envelope obligation surface)**.** *For every accepted* BishopRegularCauchyEn *carrier $E = (W, R, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}}$ *surface consists exactly of finite window exposure through W, regular rational schedule admission through R, dyadic tolerance discharge through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. No obligation row exports a quotient stream, selected representative, selected limit, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 6381.1. The analytic rows are exactly the finite windows, regular rational schedule, dyadic tolerance ledger, and terminal real seal.

The finite route begins at W. A consumer may inspect only the displayed stream windows before any regular schedule row is available, and the regular schedule R remains attached to those same windows.

The tolerance row D supplies the comparison budget for R, while the terminal seal S is downstream of both the schedule and the dyadic ledger. Thus the real-facing boundary cannot be read before the finite Cauchy envelope has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since none of those rows adds a quotient, selector, host limit, countable-choice schedule, trichotomy decision, host equality, or completeness witness, the listed rows exhaust the local obligation surface. □

Theorem 6381.3 (Bishop regular Cauchy envelope Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes window inspection, regular rational scheduling, dyadic tolerance, terminal sealing, transport, replay, provenance, and local naming only; it exports no quotient stream, selected representative, selected limit, countable-choice schedule, trichotomy decision, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 6381.2. It identifies W, R, D, S as the only analytic rows available to a real-facing consumer.

The regular schedule cannot be detached from the displayed windows. Row R is read after W, and the dyadic tolerance row D supplies the precision budget used by that same schedule.

The terminal seal is therefore forced through the finite Cauchy route. A $\mathsf{Real}^{\uparrow}$ or Completion consumer may read S only after the window, schedule, and tolerance rows have been exposed.

The structural rows preserve the same handoff. Transport, replay, provenance, and local naming add no quotient stream, selected representative, host limit, countable-choice schedule, trichotomy decision, host equality, or completeness principle, so the stated route is the complete export surface. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ read exposes StreamName windows, RegSeqRat regular schedules, DyadicRatCore tolerances, RealUp sealing, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any Real or Completion consumer receives the packet.

Independence witness: streamname, regseqrat, dyadicratcore, real: none is bijective with $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ because this packet alone binds finite windows, regular rational schedules, dyadic tolerance discharge, terminal Real sealing, transport, replay, provenance, and local naming without quotient streams or selected limits.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular schedules, DyadicRatCore tolerance rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6381.1, Definition 6381.2, and Definition 6381.3 seed the finite regular Cauchy envelope from displayed windows and tolerances to the public Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopRegularCauchyEnvelope}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, regular schedule routing, dyadic tolerance exactness, Real boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyEnvelopeUp.

Chapter 6382

A Concrete Naming Certificate for $\mathsf{CompletionDenseExtension}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Mac Lane's universal mapping language, the $\mathsf{CompletionDenseExtension}^{\uparrow}$ packet records the finite route by which a uniformly continuous map on a dense metric source is extended across a displayed completion boundary. It sits after the Cauchy completion functor of chapter 423, the completion universal property of chapter 397, the metric completion dense embedding of chapter 3351, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the terminal Real seal of chapter 38. The packet exposes a dense-extension theorem surface as finite NameCert data; it does not import quotient completion equality, selected limits, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6382.1 (Completion dense extension carrier)**.** *A* $\mathsf{CompletionDenseExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, D, Q, M, F, U, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row whose dense subcarrier supplies the domain of the initial map.*
2. *D is the dense embedding row, read through the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *surface before any completion value is inspected.*
3. *Q is the completion seal row supplied by the Cauchy completion functor and completion universal-property route.*
4. *M is the uniform-continuity modulus row for the source map on displayed source histories.*
5. *F is the extension candidate row, carrying only the completion-facing value histories generated from dense source approximants.*
6. *U is the uniqueness ledger row comparing two extension candidates after they agree on the displayed dense embedding and modulus data.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for dense approximants and target observations.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for those windows.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, Q, M, F, U, W, R have been displayed.*

10. *H records componentwise* hsame *transport for source, dense embedding, completion seal, modulus, extension, uniqueness, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a dense-source request through completion sealing, modulus transport, extension-candidate construction, uniqueness comparison, finite readback, and terminal Real sealing.*
12. *P is the* Pkg *provenance over* $\mathsf{CompletionDenseExtension}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, CompletionUniversalPrope $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseExtension}^{\uparrow}}$ *row.*

Accepted packets are compared only through the displayed source, density, completion, modulus, extension, uniqueness, readback, seal, transport, replay, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.CompletionDenseExtensionUp.CompletionDenseExtensionTasteGate single carrier alignment`

Theorem 6382.2 (Completion dense extension NameCert obligations)**.** *For every accepted* CompletionDenseExtens *carrier $E = (S, D, Q, M, F, U, W, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseExtension}^{\uparrow}}$ *surface consists exactly of source metric exposure, dense embedding exposure, completion-seal exposure, uniform-continuity modulus exposure, extension-candidate exposure, uniqueness-ledger exposure, StreamName finite windows, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every completion-extension consumer must read S, D, Q, M, F, U before it can consume W, R, A, H, C, P, N.*

Proof. Project the carrier of Definition 6382.1. The rows S, D, Q expose the source metric, dense embedding, and completion seal, while M, F, U expose the modulus, extension candidate, and uniqueness ledger.

The finite readback remains downstream of those analytic rows. StreamName windows and RegSeqRat readback in W, R can be read only after the dense source, completion seal, and modulus route have been displayed, and the terminal Real seal A is reached only after the same route is visible.

Finally H, C, P, N transport, replay, source, and name that packet. Since none of the rows stores quotient completion equality, a selected limit, countable choice, host function equality, or ambient Real completeness, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{CompletionDenseExtension}^{\uparrow}$ reader that claims a completion-extension handoff while omitting source metric exposure, dense embedding exposure, completion sealing, uniform-continuity modulus data, extension-candidate data, uniqueness ledger, StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: cauchycompletionfunctor, completionuniversalproperty, metric-completion-dense-embedding, streamname, regseqrat, real: $\mathsf{CompletionDenseExtension}^{\uparrow}$ is not bijective with any listed sibling because it joins dense source exposure, completion sealing, modulus transport, extension candidates, uniqueness comparison, finite readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionDenseExtension}^{\uparrow}}$.

Constructive story: A CompletionDenseExtension$^{\uparrow}$ packet is generated from source metric rows, dense embedding rows, completion seal rows, uniform-continuity modulus rows, extension-candidate rows, uniqueness ledgers, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompletionDenseExtension}\uparrow}$, seed)

Scope closed: Definition 6382.1 and Definition 6382.2 seed the dense-source completion-extension route.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompletionDenseExtension}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient completion equality, selected limits, countable choice, host function equality, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense embedding exposure, completion-seal exposure, modulus transport, extension-candidate exactness, uniqueness-ledger exactness, finite readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionDenseExtensionUp.

Chapter 6383

A Concrete Naming Certificate for $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$

Following the constructive Heine–Cantor proof through finite covers, the $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ packet records the finite compact-cover modulus functional that turns displayed local continuity radii into a tolerance-only uniform-modulus handoff. It sits after the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the uniform-modulus consumer of chapter 369, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38. The packet is finite ProbeBundle and rational-radius data; it does not select a Lebesgue number outside the carrier, assert open-cover compactness, use quotient map equality, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6383.1 (Finite cover modulus functional carrier)**.** *A* $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, G, B, L, D, U, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover source row, exposing only the displayed ProbeBundle cover data.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row for the map being read.*
3. *B is the finite cover cell row carrying centers, local radii, and membership witnesses.*
4. *L is the finite lower-bound fold row over the displayed radius candidates.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison row for radius and oscillation checks.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the finite fold.*
7. *W is the finite StreamName window row for source and target observations.*
8. *R is the RegSeqRat readback row for the same windows.*
9. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, G, B, L, D, U, W, R have been displayed.*
10. *H records componentwise* hsame *transport for compact-cover, continuous-map, radius, fold, tolerance, uniform-modulus, readback, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a compact-cover request through local modulus rows, dyadic radius comparison, finite lower-bound folding, uniform-modulus handoff, and terminal Real sealing.*
12. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverModulusFunctional}^{\uparrow}}$ *row.*

Accepted packets are compared only through the displayed compact-cover, continuous-map, radius, fold, tolerance, uniform-modulus, readback, seal, transport, replay, provenance, and naming rows.

Theorem 6383.2 (Finite cover modulus functional NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ *carrier* $F = (K, G, B, L, D, U, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverModulu}}$ *surface consists exactly of CompactMetric finite-cover exposure, ContinuousMap pointwise-modulus exposure, finite cover-cell radius exposure, finite lower-bound folding, DyadicRatCore tolerance accounting, UniformModulus handoff, finite observation windows, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every compact-uniform-continuity consumer must read K, G, B, L, D, U before it consumes W, R, S, H, C, P, N.*

Proof. Project the carrier of Definition 6383.1. Rows K, G, B expose the compact source, continuous graph, and finite cover cells, while L, D, U expose the lower-bound fold, dyadic comparison ledger, and uniform-modulus handoff.

The finite readback stays downstream of the cover route. StreamName windows and RegSeqRat readback in W, R only observe the displayed compact-cover and continuous-map data, and the terminal Real seal S is reached after the dyadic tolerance row and uniform-modulus row are already visible.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row carries an external Lebesgue-number selector, open-cover compactness theorem, quotient map equality, countable-choice schedule, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ reader that claims a uniform-modulus handoff while omitting CompactMetric finite-cover exposure, ContinuousMap pointwise-modulus exposure, finite cover-cell radii, finite lower-bound folding, DyadicRatCore tolerance accounting, StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ is not bijective with any listed sibling because it joins compact finite covers, pointwise continuous-map moduli, radius lower-bound folding, dyadic tolerance accounting, uniform-modulus handoff, finite readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverModulusFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverModulusFunctional}^{\uparrow}$ packet is generated from CompactMetric finite-cover rows, ContinuousMap pointwise-modulus rows, finite cover-cell radius rows, lower-bound fold rows, DyadicRatCore tolerance rows, UniformModulus handoff rows, finite observation windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCoverModulusFunctional}^{\uparrow}}$, seed)

Scope closed: Definition 6383.1 and Definition 6383.2 seed the compact finite-cover modulus-functional route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCoverModulusFunctional}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, external Lebesgue-number selection, open-cover compactness, quotient map equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover exposure, continuous-map pointwise-modulus exposure, cover-cell radius exposure, finite lower-bound folding, dyadic tolerance accounting, uniform-modulus handoff, finite readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverModulusFunctionalUp.

Chapter 6384

A Concrete Naming Certificate for $\mathsf{ChoiceSequenceRealizability}^{\uparrow}$

Following Bishop and Kleene–Vesley on choice sequences and realizability, the $\mathsf{ChoiceSequenceRealizability}^{\uparrow}$ packet records finite-prefix evidence for a choice-sequence computation as BEDC NameCert data. It sits beside the finite stream-window carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only inspected prefixes, local realizers, compatibility rows, replay, provenance, and local naming; it is not a countable-choice selector, quotient stream, selected real, or $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6384.1 (Choice-sequence realizability carrier)**.** *A* $\mathsf{ChoiceSequenceRealizability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, R, F, D, H, T, P, N)$$

whose rows display a $\mathsf{StreamName}^{\uparrow}$ *finite prefix schedule* S*, a realizability row* R *for the inspected prefix, a finite prefix-extension ledger* F*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-read row* D *when the prefix is consumed by a regular-rational route, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and a local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceRealizability}^{\uparrow}}$ *row* N*.*

The carrier may be read by a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer only through the displayed finite prefix schedule and tolerance rows. It exports no infinite sequence selector, quotient equality of choice sequences, host real endpoint, unrestricted bar induction, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6384.2 (Choice-sequence realizability prefix stability)**.** *Let* $C = (S, R, F, D, H, T, P, N)$ *be an accepted* $\mathsf{ChoiceSequenceRealizability}^{\uparrow}$ *carrier. If a longer inspected prefix refines* S *through the same finite extension ledger* F *and preserves the displayed realizability row, dyadic tolerance-read row, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support, then every RegSeqRat or Real-facing read of the refined prefix factors through the original finite prefix route before it consumes downstream rows.*

The stable read exports no choice function, quotient stream equality, selected real value, host completeness fact, or terminal $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the finite prefix schedule S and the extension ledger F. The accepted carrier displays the only rows that may be inspected: prefix data, realizability evidence, tolerance readback, transport, replay, provenance, and local naming.

Refine the prefix by reading the longer schedule through F. The hypotheses preserve R, D, H, T, P, N, so the refined route keeps the same BEDC-visible realizability and tolerance surface.

Finally hand the refined packet to the RegSeqRat or Real-facing consumer. Since the consumer can read only the displayed prefix and tolerance rows, it cannot introduce a selector for the whole choice sequence, a quotient stream equality, a selected real endpoint, or a completeness coordinate. The refined read therefore factors through the original finite prefix route. □

Theorem 6384.3 (Choice-sequence realizability NameCert obligations)**.** *For every accepted* ChoiceSequenceRealiza *carrier, the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceRealizability}^{\uparrow}}$ *surface consists exactly of the finite prefix schedule, prefix realizability row, extension ledger, dyadic tolerance-read row when present, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. The certificate is consumed through the L10 route from* $\mathsf{StreamName}^{\uparrow}$ *to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, but it does not assert countable choice, quotient sequence equality, selected representatives, host real equality, metric completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6384.1. Its rows are finite prefix data S, realizability evidence R, extension ledger F, tolerance-read data D, and the support rows H, T, P, N.

Use Definition 6384.2 to compare any accepted longer prefix with the displayed finite route. The stability theorem keeps the RegSeqRat and Real-facing reads inside the same prefix and tolerance support.

The excluded objects are not coordinates of the carrier and are not produced by the stability argument. Hence the local NameCert obligations are exactly the listed finite rows and no stronger choice-sequence or Real-completeness surface is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequenceRealizability}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoiceSequenceRealizability}^{\uparrow}}$, seed)

Scope closed: Definition 6384.1, Definition 6384.2, and Definition 6384.3 seed finite-prefix choice-sequence realizability as displayed StreamName, RegSeqRat, DyadicRatCore, support, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoiceSequenceRealizability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient sequence equality, selected representatives, host real equality, metric completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix realizability exposure, prefix-refinement stability, L10 consumer handoff, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.ChoiceSequenceRealizabilityUp.

Constructive story: A $\mathsf{ChoiceSequenceRealizability}^{\uparrow}$ packet is generated from finite StreamName prefix rows, prefix realizability evidence, finite extension ledgers, optional DyadicRatCore tolerance reads, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6385

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$

The $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ packet realizes the closed-loop transport reading of chapter 9463 as finite NameCert data. It sits beside the Poincare disk phase-transport route of chapter 3455, the observer phase groupoid of chapter 6146, the Mobius-invariant transport surface of chapter 6237, and the hyperbolic geodesic-flow packet of chapter 3856. The carrier records closed reversible phase transport through displayed disk, boundary, classifier, replay, provenance, and local naming rows; it is not a host holonomy group, an unlogged geometric law, a selected orbit, or global hyperbolic completeness.

Definition 6385.1 (Hyperbolic phase holonomy carrier)**.** *A* $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (D, L, B, C, R, V, P, N)$$

whose rows display a Poincare disk readout D*, a closed loop transport ledger* L*, a boundary-stable transport row* B*, a phase classifier row* C*, displayed replay* R*, local reversible-transport verification* V*,* Pkg *provenance* P*, and a local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}}$ *row* N*.*

The carrier may export holonomy only after the closed loop, boundary row, classifier row, replay row, provenance row, and local naming row are all displayed. It exports no ambient disk automorphism group, unbounded geodesic flow, hidden phase law, selected endpoint, host smooth connection, or global hyperbolic-completeness claim.

Theorem 6385.2 (Hyperbolic phase holonomy NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseHolonom}$ *carrier* $H = (D, L, B, C, R, V, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}}$ *surface consists only of the displayed disk row, closed-loop transport ledger, boundary-stable transport row, phase classifier row, replay row, local reversible-transport verification, provenance row, and local naming row. The holonomy export is accepted only as a finite packet over those rows and cannot be replaced by a host holonomy group, global flow, selected orbit, or geometry-level completeness principle.*

Proof. Project the carrier of Definition 6385.1. The disk row D, loop ledger L, boundary row B, and classifier row C are the only phase-holonomy-facing coordinates, while R, V, P, N provide replay, reversible-transport verification, provenance, and local naming.

Read the transport through the sibling phase routes. The Poincare disk phase-transport packet supplies finite disk transport, the observer phase groupoid supplies inverse and composition bookkeeping, Mobius-invariant transport supplies boundary-stable disk action, and geodesic flow supplies the finite flow-facing ledger.

These sibling rows jointly determine the accepted holonomy surface. None of them adds a host holonomy group, unbounded geometric law, selected orbit, smooth connection, or global completeness coordinate, so the NameCert obligations are exactly the finite rows listed above. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ route exposes disk row, loop transport row, boundary row, phase classifier row, replay row, provenance row, and local naming before exporting holonomy.

Independence witness: poincare-disk-phase-transport, observer-phase-groupoid, hyperbolic-mobius-invariant, hyperbolic-geodesic-flow: no sibling is bijective because holonomy jointly carries a closed loop, boundary-stable transport, phase classifier, replay, provenance, and local NameCert rows.

Lean TasteGate.lean will provide Nontrivial and FieldFaithful instances for $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ once the finite carrier and obligation rows are registered under BEDC.Derived.HyperbolicPhaseHolonomyUp.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6385.1 and Definition 6385.2 seed finite hyperbolic phase holonomy over disk, closed-loop transport, boundary, classifier, replay, reversible-verification, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host holonomy groups, global geodesic flow, selected orbits, smooth connections, geometry-level completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk row exposure, loop and boundary transport exposure, classifier and replay exactness, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.HyperbolicPhaseHolonomy

Constructive story: A $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ packet is generated from finite Poincare disk rows, closed-loop transport ledgers, boundary-stable transport, phase classifiers, displayed replay, reversible-transport verification, Pkg provenance, and a local NameCert row.

Chapter 6386

A Concrete Naming Certificate for $\mathsf{FiniteVariationPartition}^{\uparrow}$

Following Jordan's finite-partition account of bounded variation and the Bishop–Bridges treatment of regulated integration, the $\mathsf{FiniteVariationPartition}^{\uparrow}$ packet records the partition ledger that a variation or Riemann–Stieltjes consumer must inspect before reading a bounded-variation handoff. It sits between the bounded-variation certificate of chapter 398, the total-variation functional of chapter 2257, the Riemann-sum row of chapter 1615, and the regulated-integral route of chapter 1557. The certificate is finite: it names endpoints, mesh order, edge-difference rows, transport, replay, provenance, and local naming rows, not an ambient partition lattice or a host-level variation supremum.

Falsifiable prediction: If a $\mathsf{FiniteVariationPartition}^{\uparrow}$ packet is accepted, every bounded-variation or Riemann–Stieltjes consumer must read the displayed finite partition ledger before it can cite a variation bound; a route from a host partition selector directly to variation control refutes the packet.

Independence witness: boundedvariation, totalvariationfunction, riemannsum, regulatedintegral: none is bijective because $\mathsf{FiniteVariationPartition}^{\uparrow}$ names a finite ordered ledger of partition points, endpoint rows, mesh comparisons, edge-difference rows, transport, replay, provenance, and local NameCert rows. $\mathsf{BoundedVariation}^{\uparrow}$ names the boundedness claim, $\mathsf{TotalVariationFunction}^{\uparrow}$ names the accumulated variation functional, $\mathsf{RiemannSum}^{\uparrow}$ names tagged summation data, and $\mathsf{RegulatedIntegral}^{\uparrow}$ names the integral-facing route. None alone carries the finite partition ledger required here.

Definition 6386.1 (Finite variation partition carrier)**.** *A* $\mathsf{FiniteVariationPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, E, P, M, D, S, T, H, C, Q, N)$$

with the following displayed rows:

1. *I is the interval or domain row on which variation is measured, exposing the endpoint type and order comparison used by the packet.*
2. *E is the endpoint row, naming the left and right boundary points and their admitted order.*
3. *P is the finite partition row, listing the accepted ordered points between the endpoints.*
4. *M is the mesh-comparison row, recording adjacent-point order and the finite successor relation inside P.*

5. *D is the edge-difference row, naming the finite family of adjacent absolute differences or metric differences attached to P.*
6. *S is the finite sum or ledger row that accumulates the entries of D without taking an ambient supremum.*
7. *T is the sibling handoff row to* $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$, *with an explicit empty row when no such consumer is claimed.*
8. *H records componentwise* hsame *transport for I, E, P, M, D, S, T and the local naming row.*
9. *C records displayed* Cont *replay from endpoint admission through partition ordering, edge-difference summation, sibling handoff, provenance, and local naming.*
10. *Q is the* Pkg *provenance over* $\mathsf{FiniteVariationPartition}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationPartition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for a host-selected partition family, a supremum over all partitions, open-cover compactness, countable choice, quotient equality, or ambient real completeness.

Theorem 6386.2 (Finite variation partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVariationPartition}^{\uparrow}$ *carrier $V = (I, E, P, M, D, S, T, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint exposure, finite partition ordering, mesh comparison, edge-difference exposure, finite-sum ledger exposure, sibling handoff to the bounded-variation, total-variation, Riemann-sum, or regulated-integral rows when T is present, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation in the packet asserts a host partition selector, a global total-variation supremum, countable choice, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. Its visible rows are the domain row, endpoint row, finite partition row, mesh-comparison row, edge-difference row, finite-sum ledger, sibling handoff, transport, replay, provenance, and local name.

The partition obligation is finite. The row P supplies only the displayed list of partition points, while M records adjacent order and successor comparisons inside that list. No row ranges over all possible partitions.

The variation obligation is ledgered through D and S. A bounded-variation or total-variation consumer may read the adjacent edge differences and their finite sum, but the packet does not identify this finite ledger with a supremum over every refinement.

The sibling handoff is exact. When T cites $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, or $\mathsf{RegulatedIntegral}^{\uparrow}$, the read passes through H, C, Q, N; when T is empty, the packet makes no downstream variation or integration claim. □

Theorem 6386.3 (Finite variation partition bounded-variation handoff)**.** *Every* $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteVariationPartition}^{\uparrow}$ *packet must first read*

$$I, E, P, M, D, S, T, H, C, Q, N.$$

It receives the displayed endpoint, ordered finite partition, mesh-comparison, edge-difference, finite-sum, sibling handoff, transport, replay, provenance, and local NameCert rows. It receives no host-level partition choice, no hidden variation supremum, no unledgered refinement family, no countable choice, and no ambient completeness theorem.

Proof. Begin with the carrier of Definition 6386.1. The bounded-variation and integration-facing data occur only in the finite rows P, M, D, S, T over the endpoint data I, E.

Apply the NameCert obligations of Definition 6386.2. They require every downstream handoff to expose the finite partition ordering, mesh comparison, edge-difference ledger, and finite sum before the sibling row T is read.

The structural rows preserve the same finite route. Transport H is componentwise, replay C follows the displayed ledger, provenance Q names the sibling certificates, and N names the local packet.

Therefore a bounded-variation, total-variation, Riemann-sum, or regulated-integral consumer receives only the finite partition ledger. The absent coordinates for host partition choice, global supremum, refinement-family choice, and ambient completeness cannot be recovered from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVariationPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteVariationPartition}^{\uparrow}$ packet is generated from endpoint rows, a finite ordered partition row, mesh-comparison rows, edge-difference rows, a finite-sum ledger, a bounded-variation or integration-facing handoff, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteVariationPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6386.1, Definition 6386.2, and Definition 6386.3 seed the finite partition ledger for bounded-variation, total-variation, Riemann-sum, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteVariationPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, host-selected partition families, global variation suprema, countable choice, quotient equality, or ambient Real completeness.

Upgrade path: A formal route can encode the eleven-row carrier obligations and prove the finite bounded-variation handoff for BEDC.Derived.FiniteVariationPartitionUp.

Chapter 6387

A Concrete Naming Certificate for $\mathsf{BishopLocatedZero}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopLocatedZero}^{\uparrow}$ packet records a constructive located zero witness over the L10 real-name substrate. It sits over the real-name seal of chapter 38, the regular-rational readback of chapter 301, the finite stream-window route of chapter 126, and the located interval oscillation surface of chapter 6364. The packet names the finite sign-change and bisection rows that later interval-search consumers may inspect; it is not a host intermediate-value theorem, quotient real equality, selected global root, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6387.1 (Bishop located zero carrier)**.** *A* $\mathsf{BishopLocatedZero}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (B, M, S, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the sign-change bracket row, recording lower and upper rational endpoint windows, the two displayed sign reads, and the non-overlap evidence that the bracket remains located.*
2. *M is the uniform modulus row for the displayed function route, translating a requested sign tolerance into a finite source-window budget.*
3. *S is the bisection schedule row, carrying the finite midpoint sequence and the chosen retained half at each displayed stage.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the midpoint and endpoint windows produced by S.*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after the bracket, modulus, schedule, and readback rows have been displayed.*
6. *H records componentwise* hsame *transport for brackets, moduli, schedules, readbacks, seals, and local naming rows.*
7. *C records displayed* Cont *replay through the route B, M, S, Q, A, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedZero}^{\uparrow}$, $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedZero}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host real equality, a selected global root, quotient stream equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6387.2 (Bishop located zero NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedZero}^{\uparrow}$ *carrier* $Z = (B, M, S, Q, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedZero}^{\uparrow}}$ *surface is exhausted by sign-change bracket admission through* B*, uniform modulus exposure through* M*, bisection schedule exposure through* S*, RegSeqRat readback through* Q*, Real sealing through* A*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6387.1. The analytic input is not a host function equality claim; it is the displayed bracket row B together with the modulus row M.

The finite bisection data is then forced. The schedule row S records the retained half at each displayed stage, Q reads the same midpoint and endpoint windows as regular-rational data, and A seals the resulting real-name boundary only after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier has no selected-root, quotient, choice, compactness, host-equality, or completeness coordinate, no such object can become a local NameCert obligation. □

Theorem 6387.3 (Bishop located zero bisection soundness)**.** *Let* $Z = (B, M, S, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{BishopLocatedZero}^{\uparrow}$ *carrier. Every zero-witness read produced by the displayed bisection schedule factors through the sign-change bracket* B*, uniform modulus* M*, retained-half schedule* S*, regular-rational readback* Q*, and terminal Real seal* A *before it reaches transport, replay, provenance, and naming rows. The read is therefore a constructive located zero surface inside the L10 real-name route, not a host intermediate-value theorem or a selected global root.*

Proof. Start with the sign-change bracket. The row B is the only source for the two sign reads and the located endpoint comparison, so a bisection consumer cannot begin from an ambient real interval.

Next use the modulus and schedule rows. The row M gives the finite tolerance-to-window budget, and S records each retained half by displayed midpoint data. The readback row Q then supplies the regular-rational midpoint and endpoint route consumed by the terminal seal A.

Finally attach the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve the same finite packet. Thus bisection soundness is a row-exhaustion statement for $B, M, S, Q, A, H, C, P, N$, with no host equality, quotient stream, selected limit, countable choice, compactness, or ambient completeness imported. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedZero}^{\uparrow}$ consumer that reads a zero witness while omitting one of the sign-change bracket, uniform modulus, bisection schedule, RegSeqRat readback, Real seal, hsame, Cont, Pkg, or NameCert rows has left the packet.

Independence witness: bishoplocatedintervaloscillation, streamname, regseqrat, real: $\mathsf{BishopLocatedZero}^{\uparrow}$ is not bijective with any listed sibling because it joins sign-change bracket data, a uniform modulus, a finite bisection schedule, regular-rational readback, Real sealing, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedZero}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedZero}^{\uparrow}$ packet is generated from a sign-change bracket, a uniform modulus, a finite bisection schedule, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopLocatedZero}^{\uparrow}}$, seed)

Scope closed: Definition 6387.1, Definition 6387.2, and Definition 6387.3 seed the finite located zero witness surface for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedZero}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host intermediate-value theorem, selected global root, quotient real equality, quotient stream equality, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket admission, modulus exposure, bisection schedule exposure, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedZeroUp

Chapter 6388

Constructive Cauchy-Filter Boundary NameCert Construction

Following Bishop and Bridges, the $\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}$ packet records the finite boundary surface through which Cauchy-filter completion reads enter the Real and Completion route. It is deliberately placed between the Cauchy-filter carrier of chapter 333, the finite StreamName observation windows of chapter 126, the regular rational schedules of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the terminal Real seal of chapter 38. The packet gives completion consumers a finite boundary object without importing quotient streams, countable choice, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6388.1 (Constructive Cauchy-filter boundary)**.** *A* $\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, S, R, D, Q, H, C, P, N)$$

with the following rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row. It records the accepted filter request, its directed finite neighbourhood ledger, and the displayed Cauchy-filter admission data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed when a completion consumer asks for a named finite observation of the filter.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational schedule read from the finite StreamName windows before any Real-facing seal is consumed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic precision, radius, and comparison data used by the RegSeqRat readback.*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, S, R, D have been displayed.*
6. *H records componentwise* hsame *transport for F, S, R, D, Q and the structural rows.*
7. *C records displayed* Cont *replay of the route*

$$F \longmapsto S, D \longmapsto R \longmapsto Q, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, S, R, D, Q, H, C, P, N$. It has no coordinate for quotient stream equality, countable choice, a selected limit, host real equality, real trichotomy, or ambient Real completeness.

Lean4 (statement) `BEDC.Derived.ConstructiveCauchyFilterBoundaryUp`

Theorem 6388.2 (Constructive Cauchy-filter boundary NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}$ *carrier $B = (F, S, R, D, Q, H, C, P, N)$, Cauchy-filter completion reads are exhausted by the displayed route*

$$F \longmapsto S, D \longmapsto R \longmapsto Q.$$

The source row F is read through the Cauchy-filter packet of chapter 333; finite observations pass through the StreamName window row of chapter 126; regular rational schedules pass through the RegSeqRat readback row of chapter 301; dyadic tolerances pass through the DyadicRatCore row; and the Real-facing boundary is sealed only by Q. The structural rows H, C, P, N preserve that route and do not export quotient streams, countable choice, a selected limit, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6388.1. The only source-facing row is F, so the completion request first exposes the accepted Cauchy-filter ledger and cannot begin from a quotient stream or a selected limit.

The finite observation stage is routed through S, D, R. The StreamName row supplies the finite window, the DyadicRatCore row supplies the precision and radius comparisons, and the RegSeqRat row supplies the regular rational readback consumed by a Real-facing route.

Finally project Q, H, C, P, N. The terminal Real seal is downstream of the finite rows, while componentwise transport, continuation replay, provenance, and local naming preserve the same carrier. Since none of the excluded coordinates occurs in the packet, the boundary cannot be used as an ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}$ packet is generated from CauchyFilter source rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 6388.1 and Definition 6388.2 seed the finite Cauchy-filter boundary carrier for Real and Completion reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveCauchyFilterBoundary}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window routing, RegSeqRat readback, DyadicRatCore tolerance ordering, Real seal nonescape, provenance, and local naming for BEDC.Derived.ConstructiveCauchyFilterBoundaryUp

Chapter 6389

A Concrete Naming Certificate for $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ packet records the finite-window handoff used when regular Cauchy data enter the Real-completion route. It realizes the regular-Cauchy vision of section 9238.1 as concrete NameCert data over the finite StreamName window of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet does not import quotient streams, selected limits, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6389.1 (Finite-window RegSeqRat handoff carrier)**.** *A* $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, Q, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the inspected source indices and observation schedule.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius, common-scale comparison, and requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite window and tolerance row.*
4. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after S, D, R have been displayed.*
5. *H records componentwise* hsame *transport for the StreamName, DyadicRatCore, RegSeqRat, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the finite route*

$$S, D \longmapsto R \longmapsto Q, H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, D, R, Q, H, C, P, N. It has no coordinate for quotient stream equality, a selected limit, a countable-choice schedule, metric completeness, or ambient real completeness.

Lean4 (statement) `BEDC.Derived.FiniteWindowRegSeqRatHandoffUp`

Theorem 6389.2 (Finite-window RegSeqRat handoff NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ *carrier* $W = (S, D, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}}$ *surface consists exactly of finite StreamName window exposure through* S*, dyadic tolerance ordering through* D*, regular rational readback through* R*, terminal Real sealing through* Q*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. Every Real-completion consumer must read these rows in that order, and no obligation row exports quotient streams, selected limits, countable choice, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6389.1. The finite observation surface is S, and the requested tolerance is recorded by D, so a consumer first sees an explicit window and dyadic precision row.

The regular rational readback is R. It is attached to the same finite window and tolerance row, so the handoff cannot replace the displayed data by quotient stream equality or by a selected limiting representative.

The Real-facing boundary is terminal. Row Q can be consumed only after S, D, R have been projected, while H, C, P, N transport, replay, source, and name the same finite packet.

Since the carrier has no coordinate for choice, completeness, quotient equality, or selected limits, the listed rows exhaust the NameCert obligations and define the finite-window handoff used by Real-completion consumers. □

Falsifiable prediction: Every accepted $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ read exposes one StreamName finite window, one DyadicRatCore tolerance row, one RegSeqRat readback row, one RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and a local NameCert row before any Real-completion consumer receives the packet.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the finite observation window, tolerance row, regular readback, terminal Real seal, transport, replay, provenance, and local naming as one handoff packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}$ packet is generated from StreamName finite windows, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 6389.1 and Definition 6389.2 seed the finite-window RegSeqRat handoff over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatHandoff}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.FiniteWindowRegSeqRatHandoffUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window exposure, dyadic tolerance ordering, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowRegSeqRatHandoffUp.

Chapter 6390

A Concrete Naming Certificate for $\mathsf{PrueferSequence}^{\uparrow}$

Following Cayley and Pruefer, the $\mathsf{PrueferSequence}^{\uparrow}$ packet records a finite labelled-tree code object as a BEDC NameCert row. It reads label lists from chapter 36, vertex counters from $\mathsf{Nat}^{\uparrow}$, and finite tree-code rows from chapter 119; the packet is meant for finite tree enumeration and graph-code consumers, not for quotient graph isomorphism or an ambient set-theoretic tree space.

Definition 6390.1 (Pruefer sequence carrier)**.** *A* $\mathsf{PrueferSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (n, V, L, S, D, E, T, H, C, G, N)$$

with the following displayed rows:

1. *n is the* $\mathsf{Nat}^{\uparrow}$ *vertex-count row for a labelled finite tree with labels in the visible interval* $1, \ldots, n$*;*
2. *V is the* $\mathsf{List}^{\uparrow}$ *row listing the displayed labels without importing an ambient finite set;*
3. *L is the length row certifying that the code list has length $n - 2$ when the two-leaf boundary is active;*
4. *S is the* $\mathsf{List}^{\uparrow}$ *code row whose entries are labels from V;*
5. *D is the finite degree ledger obtained by scanning S and the two remaining labels;*
6. *E is the edge-row ledger produced by the leaf-deletion reconstruction route;*
7. *T is the* $\mathsf{Tree}^{\uparrow}$ *tree-code row carrying the displayed vertex and edge rows after reconstruction;*
8. *H records componentwise* hsame *transport for n, V, L, S, D, E, T and the naming support rows;*
9. *C records the displayed* Cont *replay from list scan to degree ledger, leaf deletion, edge ledger, and tree-code readback;*
10. *G is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrueferSequence}^{\uparrow}}$ *row.*

The carrier stores no quotient by relabelling, no selected representative of a graph-isomorphism class, and no infinite tree object.

Theorem 6390.2 (Pruefer sequence NameCert obligations)**.** *For every accepted* $\mathsf{PrueferSequence}^{\uparrow}$ *carrier*

$$P = (n, V, L, S, D, E, T, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrueferSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the vertex-count row, visible label list, code-length row, code list, degree ledger, edge ledger, reconstructed tree-code row, componentwise transport, replay routes, provenance, and local naming row. The obligation route is finite list admission, natural-number length admission, degree-ledger scan, leaf-deletion edge construction, tree-code readback, and then the* hsame*,* Cont*,* Pkg*, and* NameCert *packaging rows.*

Proof. Project the carrier of Definition 6390.1. The tuple lists the vertex-count row, label list, code length, code list, degree ledger, edge ledger, tree-code readback, transport, replay, provenance, and local name; there is no further coordinate for graph isomorphism, relabelling, or an infinite object.

Read the code list through chapter 36. Each entry of S is checked against the visible label list V, and L records the finite length boundary determined by n.

Next scan S to form D. The scan raises the displayed degree counter for each listed label and leaves the two-leaf boundary as finite counter data, so the reconstruction route has only displayed Nat and List rows to inspect.

Finally replay the leaf-deletion construction. At each finite step the least available leaf in the displayed label list pairs with the next code entry, producing one edge row in E; the remaining two labels produce the terminal edge. Repacking E as the $\mathsf{Tree}^{\uparrow}$ row T and transporting it through H, C, G, N gives exactly the stated NameCert surface. □

Theorem 6390.3 (Pruefer sequence tree-code route)**.** *Every accepted* $\mathsf{PrueferSequence}^{\uparrow}$ *carrier determines its downstream finite tree-code read solely from*

$$n, \quad V, \quad L, \quad S, \quad D, \quad E, \quad T.$$

The read consumes the $\mathsf{List}^{\uparrow}$ *code row and* $\mathsf{Nat}^{\uparrow}$ *counters first, then exports the* $\mathsf{Tree}^{\uparrow}$ *edge ledger through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It does not export a host graph, an isomorphism class, or a chosen representative outside the displayed labels.*

Proof. Begin with the list spine. The row S supplies a finite sequence of labels and L supplies its length boundary, both over the visible label list V.

Use the degree ledger D to drive the finite reconstruction. Every deletion step reads the next code entry and the least currently available displayed leaf, so each edge added to E is determined by finite rows already present in the carrier.

The terminal two-label edge completes the edge ledger. The tree-code row T is then a repacking of the same displayed labels and edges through the sibling $\mathsf{Tree}^{\uparrow}$ surface, and H, C, G, N preserve this read under transport, replay, provenance, and local naming. □

Theorem 6390.4 (Pruefer sequence Cayley consumer boundary)**.** *A finite enumeration consumer that uses* $\mathsf{PrueferSequence}^{\uparrow}$ *for Cayley's labelled-tree count may consume the packet only through the visible list-code and tree-code rows of Definition 6390.1. The consumer may count the* n^{n-2} *code rows by reading* $\mathsf{Nat}^{\uparrow}$ *exponent data and* $\mathsf{List}^{\uparrow}$ *spines, then hand the reconstructed edge ledgers to* $\mathsf{Tree}^{\uparrow}$ *or* $\mathsf{Graph}^{\uparrow}$ *consumers. It may not identify trees up to relabelling, import a quotient classifier, or replace the displayed reconstruction route with an ambient bijection proof.*

Proof. The consumer first reads n, V, L, S. These rows expose exactly the finite label interval, the visible list spine, and the code-length boundary used to count code rows.

It then invokes Definition 6390.3. The reconstruction route supplies D, E, T as displayed rows, so the consumer receives a tree-code ledger rather than a host graph object.

The remaining rows H, C, G, N preserve the same dependency boundary. Since the carrier has no quotient, relabelling representative, or ambient bijection coordinate, the Cayley-facing consumer boundary is exactly the finite code-to-tree route described above. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrueferSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrueferSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6390.1, Definition 6390.2, Definition 6390.3, and Definition 6390.4 seed the finite labelled-tree code packet over displayed Nat, List, degree-ledger, edge-ledger, Tree, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrueferSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient graph isomorphism, relabelling invariance, unlabelled tree enumeration, a host bijection proof, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, code-length exactness, degree-ledger scan, leaf-deletion reconstruction, tree-code readback, and Cayley-facing consumer boundary for the same finite packet.

Constructive story: A $\mathsf{PrueferSequence}^{\uparrow}$ packet is generated from a displayed vertex count, visible label list, finite code list, degree counters, reconstructed edge rows, a tree-code row, componentwise transports, replay routes, provenance, and local NameCert evidence.

Chapter 6391

A Concrete Naming Certificate for $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ packet records the endpoint seal that a displayed interval contraction must expose before downstream fixed-point consumers may read a Real endpoint. It sits after the interval-contraction route of chapter 5867, the general contraction-mapping surface of chapter 1418, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the stability consumer of chapter 5856. The packet names the endpoint handoff as finite BEDC data; it does not assert a host Banach fixed-point theorem, select a completed limit, use quotient real equality, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6391.1 (Bishop contraction endpoint-seal carrier)**.** *A* $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, F, Q, R, X, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *row exposing the located interval, contraction witness, dyadic shrinkage ledger, finite windows, regular-rational readback, and Real seal already certified by the interval-contraction packet.*
2. *F is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row recording that the same displayed map is read as a contraction before an endpoint consumer may inspect it.*
3. *Q is the endpoint-request row, naming the requested fixed-point endpoint handoff and the shrinkage budget that routes it to the readback row.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the interval-contraction windows and endpoint request.*
5. *X is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal consumed by fixed-point consumers.*
6. *H records componentwise* hsame *transport for I, F, Q, R, X, P, N.*
7. *C records displayed* Cont *replay from the endpoint request through I, F, Q, R, X, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$, $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BanachFixedPointStability}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopContractionEndpointSeal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a selected fixed point, quotient real equality, a host convergence theorem, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6391.2 (Bishop contraction endpoint-seal NameCert obligations)**.** *For every accepted* $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ *carrier* $E = (I, F, Q, R, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopContractionEndpointSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval-contraction row, contraction-mapping row, endpoint request, regular rational readback, Real endpoint seal, transport, replay, provenance, and local naming rows; contraction-route compatibility between I and F; endpoint-budget exactness through Q and R; Real-seal ordering through X; and downstream stability handoff for* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *consumers. No NameCert obligation is stored outside* $I, F, Q, R, X, H, C, P, N$.

Proof. Carrier admission is projection from Definition 6391.1. The tuple lists the interval-contraction source, contraction-mapping row, endpoint request, regular readback, Real seal, transport, replay, provenance, and local name as the only available coordinates.

Contraction-route compatibility reads I and F together. The $\mathsf{BishopIntervalContraction}^{\uparrow}$ row supplies the located interval, dyadic shrinkage ledger, finite windows, and readback route, while the $\mathsf{ContractionMapping}^{\uparrow}$ row records the displayed map as the contraction consumed by the endpoint request.

Endpoint-budget exactness is then read through Q and R. The request row names the endpoint budget, and the $\mathsf{RegSeqRat}^{\uparrow}$ row receives only the scheduled readback exported by the interval-contraction packet.

Finally the Real handoff row X and the structural rows H, C, P, N preserve the same route for $\mathsf{BanachFixedPointStability}^{\uparrow}$ consumers. Since the carrier contains no selected fixed point, quotient real equality, host convergence theorem, countable-choice row, or ambient Real-completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 6391.3 (Bishop contraction endpoint-seal Real handoff)**.** *Every fixed-point stability consumer that reads the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal X from an accepted* $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ *carrier* $E = (I, F, Q, R, X, H, C, P, N)$ *must first read*

$$\mathsf{BishopIntervalContraction}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto Q \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows I, F, Q, R, X, and then read the structural rows H, C, P, N for hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. The handoff may be consumed by* $\mathsf{BanachFixedPointStability}^{\uparrow}$*, but it exports no selected fixed point, quotient completion, host Banach theorem, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the endpoint request. The carrier exposes the source interval-contraction route only through I and the displayed contraction row only through F, so a consumer cannot begin from an external fixed-point theorem or an unrecorded self-map.

Read the endpoint budget through Q. By the NameCert obligations of Definition 6391.2, this budget is the only route from the contraction data to the $\mathsf{RegSeqRat}^{\uparrow}$ readback R.

The terminal Real endpoint seal is then X. It is downstream of I, F, Q, R, so a fixed-point stability consumer receives a displayed regular-rational-to-Real handoff rather than a chosen limit or quotient real equality.

Finally replay the route through H, C, P, N. Componentwise transport, displayed continuation, provenance, and local naming preserve the same endpoint-seal packet before $\mathsf{BanachFixedPointStability}^{\uparrow}$ consumes it, and no excluded host fixed-point or completeness data is present in the carrier. □

Falsifiable prediction: An accepted $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ consumer that reaches the $\mathsf{Real}^{\uparrow}$ endpoint seal or a $\mathsf{BanachFixedPointStability}^{\uparrow}$ handoff without reading $I, F, Q, R, X, H, C, P, N$ refutes the carrier surface.

Independence witness: bishop-interval-contraction, contraction-mapping, regseqrat, real, banach-fixed-point-stability: no listed sibling supplies the joint endpoint-seal packet containing an interval-contraction source, displayed contraction map, endpoint request budget, regular rational readback, terminal Real endpoint seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopContractionEndpointSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopContractionEndpointSeal}^{\uparrow}$ packet is generated from a BishopIntervalContraction row, a ContractionMapping row, an endpoint request budget, RegSeqRat readback, a Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopContractionEndpointSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6391.1, Definition 6391.2, and Definition 6391.3 seed the finite endpoint-seal route from interval contraction and contraction-mapping data to RegSeqRat, Real, and BanachFixedPointStability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopContractionEndpointSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a selected fixed point, quotient real equality, host Banach fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction-route compatibility, endpoint-budget exactness, Real handoff ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopContractionEndpointSealUp

Chapter 6392

A Concrete Naming Certificate for $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$

The $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ packet realizes the finite grid-refinement route of chapter 9237 as concrete NameCert data. It sits over the compact finite-net surface of chapter 129, the dyadic tolerance ledger of chapter 291, the uniform-modulus consumer of chapter 369, and the terminal real-name boundary of chapter 38. The packet records compact-net rows, dyadic-radius refinements, and tolerance-only handoff data; it does not import open-cover compactness, a host grid topology, selected mesh limits, or ambient completeness.

Definition 6392.1 (Compact dyadic grid-refinement carrier)**.** *A* $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Q, \Delta, \Gamma, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the consumed* $\mathsf{CompactMetric}^{\uparrow}$ *row, including the displayed compact finite-net and located metric support;*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the input dyadic radius, half-radius, and comparison budget;*
3. *Δ is the finite dyadic grid row, listing cells whose centers are read from the compact finite-net surface of X;*
4. *Γ is the refinement ledger, recording which dyadic child cells refine which parent cells and which compact-net probes they cover;*
5. *R is the finite-net coverage row saying that every inspected source point is routed through one displayed grid cell and one compact-net center;*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row, available only after Q, Δ, Γ, R have been projected;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the displayed dyadic tolerance and uniform-modulus route;*
8. *H records componentwise* hsame *transport for $X, Q, \Delta, \Gamma, R, U, E$ and the support rows;*
9. *C records the displayed* Cont *replay from compact-net readback to dyadic subdivision, coverage, uniform-modulus handoff, and real seal;*

10. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactDyadicGridRefinement}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and stores no open-cover proof, external grid space, quotient real equality, selected mesh limit, or ambient completeness principle.

Theorem 6392.2 (Compact dyadic grid-refinement NameCert obligations)**.** *For every accepted* $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ *carrier*

$$G = (X, Q, \Delta, \Gamma, R, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactDyadicGridRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric row, dyadic tolerance row, finite grid row, refinement ledger, coverage row, uniform-modulus handoff, real seal, componentwise transport, replay routes, provenance, and local naming row. The obligation route is compact finite-net readback, dyadic-radius admission, finite grid subdivision, parent-child refinement coverage, uniform-modulus consumer handoff, real-seal nonescape, and then the* hsame, Cont, Pkg, *and* NameCert *packaging rows.*

Proof. Project the carrier of Definition 6392.1. The displayed tuple is the full public surface, so an open-cover proof, host grid topology, selected mesh limit, quotient real equality, and ambient completeness are not available as extra coordinates.

Read X and Q first. The compact metric row supplies finite-net centers and located metric support, while the dyadic tolerance row supplies the visible radius, half-radius, and comparison budget used by every grid cell.

Next build Δ, Γ, R. The finite grid row lists dyadic cells, the refinement ledger ties each child to a parent and to the compact-net probes it covers, and the coverage row records the inspected source-point route through one displayed cell and one compact-net center.

Finally expose U and E. The uniform-modulus handoff consumes only the compact-net, dyadic-radius, grid, refinement, and coverage rows, and the real seal is placed after that handoff. Repacking through H, C, P, N gives exactly the NameCert surface stated above. □

Theorem 6392.3 (Compact dyadic grid-refinement finite-net route)**.** *Every finite-net read of an accepted* $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ *packet factors through*

$$X, \quad Q, \quad \Delta, \quad \Gamma, \quad R.$$

The route first consumes the compact finite-net row of X, then reads dyadic radii from Q, then exposes the displayed grid cells and refinement ledger. No consumer receives a standalone grid topology, an untracked mesh parameter, or a selected cover outside the finite BEDC rows.

Proof. Start with the compact source row. The only centers available to the packet are the finite-net centers carried by X.

Use the dyadic tolerance row Q to size the cells. Each grid cell in Δ is therefore a finite dyadic-radius row, and each parent-child relation in Γ is checked against the same comparison budget.

The coverage row R ties every inspected source point to one displayed cell and compact-net center. Since H, C, P, N only transport, replay, package, and name these same rows, the finite-net read cannot escape to a host grid or selected cover. □

Theorem 6392.4 (Compact dyadic grid-refinement uniform-modulus consumer)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ *packet reads the compact-net coverage, dyadic radius, grid subdivision, and refinement rows before it may consume the uniform-modulus handoff row* U*. The subsequent* $\mathsf{Real}^{\uparrow}$ *seal* E *is available only downstream of* U *and the displayed* hsame, Cont, Pkg*, and local* NameCert *rows. Thus the consumer receives finite compact-net, dyadic-radius, and uniform-modulus handoff data, not an ambient Heine–Cantor theorem or host real-completeness argument.*

Proof. Apply Definition 6392.3. It fixes the finite route through X, Q, Δ, Γ, R before any uniform-modulus read is exposed.

The row U is downstream of that route in the carrier. It therefore reads a tolerance-only modulus handoff from displayed compact-net and dyadic-grid data rather than from a selected external modulus.

The real seal E comes after U. Transport and replay through H and C, together with provenance P and local naming N, preserve that order, so no consumer can bypass the finite grid-refinement packet to reach ambient completeness or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactDyadicGridRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactDyadicGridRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6392.1, Definition 6392.2, Definition 6392.3, and Definition 6392.4 seed the finite compact-net, dyadic-radius, grid-refinement, coverage, uniform-modulus, real-seal, transport, replay, provenance, and local naming packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactDyadicGridRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host grid topology, selected mesh limits, quotient real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net readback, dyadic-radius exactness, grid-refinement coverage, uniform-modulus consumer handoff, and real-seal nonescape for the same finite packet.

Constructive story: A $\mathsf{CompactDyadicGridRefinement}^{\uparrow}$ packet is generated from a compact metric finite-net row, dyadic tolerance data, finite grid cells, refinement coverage, a uniform-modulus handoff, a RealUp seal, componentwise transports, replay routes, provenance, and local NameCert evidence.

Chapter 6393

A Concrete Naming Certificate for $\mathsf{LocatedCantorBendixson}^{\uparrow}$

Following Cantor and Bendixson, the $\mathsf{LocatedCantorBendixson}^{\uparrow}$ packet records the constructive derived-set route for a compact located real subset as finite derivative rows and perfect-kernel boundary data. It sits over the Cantor-set prefix surface of chapter 1530, the compact metric carrier of chapter 129, the closed-set boundary of chapter 4229, the finite intersection discipline of chapter 1792, and the real-name seal of chapter 38. The certificate names the located derivative search and residual perfect-kernel rows; it does not assert a host transfinite rank, a powerset decomposition, countable choice, or a classical isolated-point selector.

Definition 6393.1 (Located Cantor–Bendixson carrier)**.** *A* $\mathsf{LocatedCantorBendixson}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, C, D, I, P, R, H, Q, S, N)$$

with the following displayed rows:

1. *K is the compact located metric row, naming the compact metric presentation, its located distance read, and the finite cover budgets available to the derivative search.*
2. *C is the closed real-set row, recording closedness tests through displayed complements and* $\mathsf{Real}^{\uparrow}$ *seals.*
3. *D is the finite derived-set ledger, listing inspected neighborhoods, witnessed isolated exclusions, and the residual closed rows generated by the displayed search depth.*
4. *I is the finite isolated-point ledger, carrying the local separation witness for each isolated read admitted by D.*
5. *P is the residual perfect-kernel row, recording the no-isolated-point boundary for the remaining displayed closed subpacket.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *membership and distance handoff row consumed only after K, C, D, I, P have been displayed.*
7. *H records componentwise* hsame *transport for compact metric data, closed-set tests, derivative ledgers, isolated witnesses, residual rows, real handoff, continuation replay, provenance, and local naming.*
8. *Q records displayed* Cont *replay from finite neighborhood requests to derivative rows and from residual rows to the perfect-kernel boundary.*

9. *S is the* Pkg *provenance over* $\mathsf{LocatedCantorBendixson}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{FiniteIntersectionProperty}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCantorBendixson}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host ordinal rank, ambient powersets, a selected enumeration of all isolated points, quotient real equality, countable choice, or a classical Cantor–Bendixson theorem outside the displayed compact located route.

Theorem 6393.2 (Located Cantor–Bendixson NameCert obligations)**.** *For every accepted* LocatedCantorBendixson *carrier $L = (K, C, D, I, P, R, H, Q, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCantorBendixson}^{\uparrow}}$ *surface is exhausted by the following five obligation rows:*

1. *compact located admission through K, exposing only displayed finite cover and distance data;*
2. *closed real-set boundary through C, exposing only displayed complement and seal tests;*
3. *derived-set ledger exactness through D, relating every inspected neighborhood to the finite derivative row;*
4. *isolated-point separation through I, naming each admitted isolated read by a displayed local witness;*
5. *residual perfect-kernel stability through P, R, H, Q, S, N, keeping every surviving membership read inside the real handoff, transport, replay, provenance, and local naming route.*

Proof. Project the carrier of Definition 6393.1. The compact and closed inputs are rows K and C, so the certificate starts from displayed located metric and closed-set data rather than from a host powerset.

The derivative work is finite at the named surface. Row D records the inspected neighborhoods and residual closed rows, while I stores only those isolated reads that come with displayed local separation evidence.

The residual row P binds the remaining perfect-kernel boundary. The real handoff R, transport H, continuation replay Q, provenance S, and local naming row N preserve the same route, so the five obligation rows exhaust the NameCert surface without importing an ordinal rank, a global isolated-point enumeration, quotient real equality, countable choice, or a classical decomposition theorem. □

Theorem 6393.3 (Located Cantor–Bendixson derivative stability)**.** *Let $L = (K, C, D, I, P, R, H, Q, S, N)$ be an accepted* $\mathsf{LocatedCantorBendixson}^{\uparrow}$ *carrier. Every derivative read produced from the displayed compact located closed set factors through*

$$K \longmapsto C \longmapsto D \longmapsto I \longmapsto P \longmapsto R$$

before it reaches transport, replay, provenance, and local naming rows. The residual closed packet is therefore stable under the displayed derived-set route, not under a host transfinite decomposition or a selected classical isolated-point enumeration.

Proof. Start with the compact located row K and closed row C. Together they provide the only admissible neighborhoods, distance reads, finite cover budgets, complement tests, and real seals available to the derivative search.

Next read the derivative and isolated ledgers. The row D records which neighborhoods have been inspected at the displayed depth, and I attaches local separation evidence to the isolated reads admitted there. No row supplies a global enumeration of all isolated points.

Finally pass through the residual row and structural rows. The perfect-kernel boundary P is consumed before the real handoff R, and H, Q, S, N transport, replay, source, and name that same finite route. Hence derivative stability is a row-exhaustion statement for the displayed compact located closed packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCantorBendixson}^{\uparrow}$ consumer that reads a derivative or residual perfect-kernel witness while omitting one of the compact located admission, closed real-set boundary, derivative ledger, isolated separation, residual kernel, Real handoff, hsame, Cont, Pkg, or NameCert rows has left the packet.

Independence witness: cantorset, compactmetric, closedset, finiteintersectionproperty, real: $\mathsf{LocatedCantorBendixson}^{\uparrow}$ is not bijective with any listed sibling because it joins compact located admission, closed real-set tests, finite derived-set rows, isolated separation, residual perfect-kernel stability, real handoff, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCantorBendixson}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCantorBendixson}^{\uparrow}$ packet is generated from compact located metric data, closed real-set tests, finite derived-set ledgers, isolated-point separation witnesses, residual perfect-kernel rows, Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCantorBendixson}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6393.1, Definition 6393.2, and Definition 6393.3 seed the constructive derived-set decomposition route for compact located closed real subsets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCantorBendixson}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host transfinite rank, ambient powerset decomposition, a selected enumeration of all isolated points, quotient real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact located exactness, closed-set boundary exactness, derivative ledger exactness, isolated separation, residual perfect-kernel stability, route replay, provenance, local naming, and non-escape for BEDC.Derived.LocatedCantorBendixsonUp.

Chapter 6394

A Concrete Naming Certificate for $\mathsf{NetConvergenceCriterion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{NetConvergenceCriterion}^{\uparrow}$ packet records the directed-net convergence test as finite L10 NameCert data rather than as an ambient filter or topological limit assertion. It reads directed indices from chapter 1607, net windows from chapter 3403, eventual-tail comparison through chapter 1601, uniform entourage control through chapter 339, and Cauchy-tail consumers through chapter 589. The packet is a nonquotient criterion surface for directed convergence; it is not a selected subnet, a host limit point, countable choice, quotient net equality, or ambient compactness.

Definition 6394.1 (Net convergence criterion carrier)**.** *A* $\mathsf{NetConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, N, F, U, T, E, H, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row, carrying the finite comparison and upper-bound witnesses that a tail request may inspect.*
2. *N is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *observation row, recording the source net values opened at the displayed directed indices.*
3. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *eventual-tail row, translating a neighborhood or entourage request into a displayed tail base.*
4. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *control row, exposing the entourage ledger against which the net values are compared.*
5. *T is the convergence-test row, pairing every accepted entourage request with a tail base and the finite net windows required on that tail.*
6. *E is the* $\mathsf{CauchyNet}^{\uparrow}$ *handoff row, available only after the directed, net, filter-base, uniform-space, and convergence-test rows have all been displayed.*
7. *H records componentwise* hsame *transport for D, N, F, U, T, E and the structural rows.*
8. *C records displayed* Cont *replay from a convergence request through directed indices, net windows, filter-base tails, uniform entourages, and the Cauchy-net handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{NetConvergenceCriterion}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. L is the local $\mathsf{NameCert}_{\mathsf{NetConvergenceCriterion}^{\uparrow}}$ *row.*

Accepted packets are compared only through $D, N, F, U, T, E, H, C, P, L$. No coordinate stores a host limit, a selected subnet, quotient equality of nets, countable choice, or an ambient compactness principle.

Theorem 6394.2 (Net convergence criterion NameCert obligations)**.** *For every accepted* $\mathsf{NetConvergenceCriterion}^{\uparrow}$ *carrier $Q = (D, N, F, U, T, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{NetConvergenceCriterion}^{\uparrow}}$ *surface consists exactly of directed-index admission, Moore–Smith net window exposure, filter-base eventual-tail exposure, uniform-space entourage accounting, convergence-test exactness, Cauchy-net handoff admission, componentwise transport, displayed replay, provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6394.1. The convergence request first exposes the directed index row D and the Moore–Smith net row N, so every later tail read is tied to displayed directed data.

The eventual and uniform parts are then fixed. The filter-base row F supplies the admitted tail base, while U supplies the entourage ledger against which the net windows are compared. The test row T records only those paired tail and entourage checks.

The handoff E is downstream of those rows and may be consumed by Cauchy-net readers only after the criterion surface is visible. The structural rows H, C, P, L transport, replay, source, and name the same finite packet. Since no carrier row stores a host limit, selected subnet, quotient equality, choice schedule, or compactness witness, the listed rows exhaust the local NameCert obligations. □

Theorem 6394.3 (Net convergence criterion filter handoff)**.** *Every* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *consumer of an accepted* $\mathsf{NetConvergenceCriterion}^{\uparrow}$ *packet must read the route*

$$D, N \longmapsto F, U \longmapsto T \longmapsto E, H, C, P, L$$

before it may inspect the convergence result. Thus the criterion hands off a displayed eventual-tail test over directed Moore–Smith net data, not a quotient filter, selected convergent subnet, host limit point, countable-choice schedule, or ambient compactness theorem.

Proof. Start with the directed and net rows. The consumer can inspect only the index comparisons and finite net windows exposed by D and N.

Next read the filter and uniform rows. The row F gives the tail base used for the request, and U gives the entourage ledger. The convergence-test row T binds those two rows to the same net windows.

Finally the Cauchy-net handoff E and the structural rows H, C, P, L preserve that route for downstream readers. Because every public read factors through the displayed packet, the handoff cannot be replaced by a quotient filter, selected subnet, host limit, choice principle, or ambient compactness assertion. □

Falsifiable prediction: Every accepted $\mathsf{NetConvergenceCriterion}^{\uparrow}$ reader that claims directed convergence while omitting the DirectedSet index row, Moore–Smith net windows, FilterBase tail row, UniformSpace entourage ledger, convergence-test row, CauchyNet handoff, hsame transport, Cont replay, Pkg provenance, or local NameCert has left the packet surface.

Independence witness: directedset, mooresmithnet, filterbase, uniformspace, cauchynet: $\mathsf{NetConvergenceCriterio}$ is not bijective with any listed sibling because it joins directed index comparison, net observation,

eventual-tail filtering, entourage control, convergence testing, Cauchy-net handoff, transport, replay, provenance, and local naming as one finite criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NetConvergenceCriterion}\uparrow}$.

Constructive story: A $\mathsf{NetConvergenceCriterion}^{\uparrow}$ packet is generated from DirectedSet index rows, Moore–Smith net windows, FilterBase eventual-tail rows, UniformSpace entourage ledgers, a convergence-test row, a CauchyNet handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NetConvergenceCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6394.1, Definition 6394.2, and Definition 6394.3 seed the displayed directed-net convergence criterion for completion and uniform-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NetConvergenceCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host topological limit, selected subnet, quotient filter, quotient net equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-index exactness, Moore–Smith net window replay, filter-base tail exactness, uniform entourage accounting, convergence-test exactness, Cauchy-net handoff, transport, replay, provenance, and local naming for BEDC.Derived.NetConvergenceCriterionUp.

Chapter 6395

A Concrete Naming Certificate for $\mathsf{CompactRadiusChain}^{\uparrow}$

Following the Bishop–Heine–Cantor finite-cover proof, the $\mathsf{CompactRadiusChain}^{\uparrow}$ packet records the radius ledger that is read before a compact-continuity consumer reaches a $\mathsf{UniformModulus}^{\uparrow}$ row. It sits after the compact metric finite data of chapter 129, the ProbeBundle lower-bound fold of chapter 4118, the continuous-map pointwise radius route of chapter 130, the rational comparison surface of chapter 291, and the uniform-modulus consumer of chapter 369. The packet is the finite radius-chain component of the compact-uniform route; it is not an external Lebesgue-number selector, open-cover compactness, quotient map equality, countable choice, or ambient Real completeness.

Definition 6395.1 (Compact radius-chain carrier)**.** *A* $\mathsf{CompactRadiusChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, G, B, A, F, D, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-data row, exposing the compact source only as displayed ProbeBundle cover information.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-radius row for the map whose compact-uniform handoff is being read.*
3. *B is the finite ProbeBundle cell row, carrying centers, local membership witnesses, and the radius candidates attached to each cell.*
4. *A is the positive-radius admission row, requiring every local radius candidate to be visible as finite rational or dyadic comparison data.*
5. *F is the finite lower-bound fold row over the displayed radius candidates.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the folded radius with the requested tolerance budget.*
7. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *placeholder row, available only as a handoff after K, G, B, A, F, D have been displayed.*
8. *H records componentwise* hsame *transport for compact source rows, pointwise-radius rows, bundle cells, radius admissions, folds, dyadic comparisons, and the handoff row.*

9. *C records displayed* Cont *replay from compact finite data through pointwise radii, ProbeBundle folding, dyadic comparison, and uniform-modulus handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRadiusChain}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRadiusChain}^{\uparrow}}$ *row.*

Accepted packets are compared only through $K, G, B, A, F, D, U, H, C, P, N$ and never through an external radius selector, an open cover, a quotient map, countable choice, or ambient completeness.

Theorem 6395.2 (Compact radius-chain NameCert obligations)**.** *For every accepted* $\mathsf{CompactRadiusChain}^{\uparrow}$ *carrier $R = (K, G, B, A, F, D, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactRadiusChain}^{\uparrow}}$ *surface consists exactly of compact finite-data admission, continuous-map pointwise-radius admission, ProbeBundle cell exposure, positive-radius admission, finite lower-bound folding, DyadicRatCore tolerance accounting, UniformModulus handoff admission, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6395.1. The compact source and pointwise-radius rows K, G are visible before any radius fold can be read.

The finite chain is then displayed by B, A, F, D. The ProbeBundle cells name the finite family, the radius admission row exposes each positive candidate, the fold row records the lower-bound construction, and the dyadic ledger records the tolerance comparison.

Only after those rows are visible may the handoff row U be consumed by a uniform-modulus reader. The rows H, C, P, N transport, replay, source, and name the same packet. Since no row stores an external selector, open cover, quotient equality, choice schedule, or ambient completeness witness, these rows exhaust the local NameCert obligations. □

Theorem 6395.3 (Compact radius-chain uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{CompactRadiusChain}^{\uparrow}$ *packet reaches* $\mathsf{UniformModulus}^{\uparrow}$ *only through*

$$K, G \longmapsto B, A, F, D \longmapsto U, H, C, P, N.$$

The handoff exposes a finite ProbeBundle lower-bound chain before the uniform-modulus row is consumed; it does not export a selected Lebesgue number, open-cover compactness, quotient map equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the compact and pointwise rows first. The consumer receives compact finite data from K and local radius information from G, not an ambient compactness theorem.

Next read the ProbeBundle radius chain. The cells B, admissions A, fold F, and dyadic ledger D display the lower-bound route that binds all local radii to one finite tolerance comparison.

Finally consume the handoff row U and the structural rows H, C, P, N. Because the uniform-modulus read is downstream of the displayed chain, it cannot be replaced by a selected Lebesgue number, an open-cover proof, quotient equality, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactRadiusChain}^{\uparrow}$ reader that claims a compact uniform-modulus handoff while omitting compact finite data, pointwise-radius data, ProbeBundle cells, positive-radius admission, lower-bound folding, DyadicRatCore comparison, hsame transport, Cont replay, Pkg provenance, or local NameCert has left the packet.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, probebundle-finite-minimum-fold: $\mathsf{CompactRadiusChain}^{\uparrow}$ is not bijective with any listed sibling because

it joins compact finite data, pointwise local radii, finite bundle cells, positive-radius admission, lower-bound folding, dyadic tolerance comparison, uniform-modulus handoff, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRadiusChain}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRadiusChain}^{\uparrow}$ packet is generated from CompactMetric finite data, ContinuousMap pointwise-radius rows, ProbeBundle cell rows, positive-radius admissions, a finite lower-bound fold, DyadicRatCore tolerance comparison, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRadiusChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6395.1, Definition 6395.2, and Definition 6395.3 seed the finite radius-chain surface for compact-uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRadiusChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an external Lebesgue-number selector, open-cover compactness, quotient map equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-data admission, pointwise-radius admission, ProbeBundle cell exposure, positive-radius admission, lower-bound folding, dyadic tolerance accounting, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactRadiusChainUp.

Chapter 6396

A Concrete Naming Certificate for $\mathsf{FreudenthalCompactification}^{\uparrow}$

Following Freudenthal's compactification of locally compact spaces by ends, the $\mathsf{FreudenthalCompactification}^{\uparrow}$ packet records a source space together with a finite end-filter boundary and a compact-boundary admission route as NameCert data. It reads the end carrier of chapter 5998, the one-point boundary discipline of chapter 2049, the compact metric route of chapter 129, the locally compact surface of chapter 1726, the topology carrier of chapter 88, and the metric carrier of chapter 57. The packet names the compactification boundary through displayed finite end filters rather than through an ambient quotient, selected ideal points, open-cover compactness, or choice-selected escaping rays.

Definition 6396.1 (FreudenthalCompactification carrier)**.** *A* $\mathsf{FreudenthalCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, E, B, J, T, H, Q, P, N)$$

with the following displayed rows:

1. *X is the source row, exposing the* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, *and optional* $\mathsf{ProperMetric}^{\uparrow}$ *support used by the compactification packet.*
2. *E is the* $\mathsf{FreudenthalEnd}^{\uparrow}$ *row. It records compact-exhaustion stages, complement-component rows, tail-comparison data, and escaping-prefix ledgers exactly as in chapter 5998.*
3. *B is the finite end-filter boundary row. Each boundary request names the compact stage, the complement-component identifier, and the tail ledger slice that admits the boundary read.*
4. *J is the source-inclusion row. It records the displayed embedding of the source row into the compactification carrier without adding a quotient host space.*
5. *T is the compact-boundary admission ledger. It compares finite source neighbourhood reads, one-point compactification boundary discipline, and compact metric support rows before any boundary request is public.*
6. *H records componentwise* hsame *transport for the source, end-filter, boundary, inclusion, compact-boundary ledger, provenance, and local naming rows.*
7. *Q records displayed* Cont *replay from a compactification request through X, E, B, J, T, H.*
8. *P is the* Pkg *provenance over* $\mathsf{FreudenthalCompactification}^{\uparrow}$, $\mathsf{FreudenthalEnd}^{\uparrow}$, $\mathsf{OnePointCompactification}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FreudenthalCompactification}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient quotient compactification, a selected ideal boundary point, an open-cover compactness proof, a chosen escaping ray, countable-choice enumeration of ends, or a standard bridge.

Theorem 6396.2 (FreudenthalCompactification NameCert obligation surface)**.** *For every accepted* $\mathsf{FreudenthalCompactification}^{\uparrow}$ *carrier* $K = (X, E, B, J, T, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FreudenthalCompactificati}}$ *surface consists exactly of source admission, Freudenthal end admission, finite end-filter boundary display, source-inclusion display, compact-boundary ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A compactification consumer must read E before any boundary row B, must read X, J, B before the compact-boundary ledger T, and must keep the boundary admission route separate from open-cover compactness, quotient equality, selected ideal points, and choice-selected rays.*

Proof. Project the carrier of Definition 6396.1. The source row X supplies only the displayed topological, metric, locally compact, and optional proper metric data, while the end row E supplies the finite Freudenthal end surface.

The boundary row B is therefore readable only through compact stages, complement-component identifiers, and tail ledger slices already exposed by E. The source-inclusion row J records how source points enter the compactification carrier, but it does not create a quotient host space or a selected ideal point.

Next read the compact-boundary ledger T. Its only inputs are X, E, B, J together with the cited one-point, compact metric, locally compact, topological, and metric support rows, so it is a finite boundary-admission ledger rather than an open-cover compactness theorem.

Finally transport and replay through H and Q, then package and name through P and N. Since no displayed coordinate stores quotient compactification equality, selected rays, countable-choice enumeration of ends, or a standard bridge, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreudenthalCompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{FreudenthalCompactification}^{\uparrow}$ packet is generated from a displayed locally compact or metric source, a Freudenthal end row, finite end-filter boundary rows, a source-inclusion row, a compact-boundary admission ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FreudenthalCompactification}^{\uparrow}}$, seed)

Scope closed: Definition 6396.1 and Definition 6396.2 seed the Freudenthal compactification carrier over FreudenthalEnd, OnePointCompactification, CompactMetric, LocallyCompact, TopologicalSpace, MetricSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FreudenthalCompactification}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a quotient compactification, public compact Hausdorff bridge, open-cover compactness theorem, selected boundary points, selected escaping rays, countable-choice enumeration of ends, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, end-filter boundary display, source inclusion, compact-boundary ledger readback, transport, replay, provenance, and local naming for BEDC.Derived.FreudenthalCompactificationUp.

Chapter 6397

A Concrete Naming Certificate for $\mathsf{LocatedZeroUniqueness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ packet records the uniqueness boundary for two displayed located-zero reads over one shared constructive bracket route. It sits after the located-zero witness surface of chapter 6387, the finite stream and regular-rational readbacks of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, and the separated comparison boundary of chapter 1636. The packet names classifier equality over a displayed separated-zero route; it is not host equality of roots, a selected root, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6397.1 (Located zero uniqueness carrier)**.** *A* $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, M, S, Z_0, Z_1, W_0, W_1, R_0, R_1, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B, M, S are the shared* $\mathsf{BishopLocatedZero}^{\uparrow}$ *bracket, modulus, and bisection-schedule rows, recording that both zero reads come from the same displayed sign-change route.*
2. *Z_0 and Z_1 are the two displayed zero-witness rows, each pointing back to the common B, M, S route before any comparison is read.*
3. *W_0 and W_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *window rows used to expose the finite source observations for the displayed zero reads.*
4. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the comparison route.*
6. *E is the* $\mathsf{RealEquality}^{\uparrow}$ *comparison row, or equivalently a separated-metric zero-distance comparison row, read only after $B, M, S, Z_0, Z_1, W_0, W_1, R_0, R_1, D$ have been displayed.*
7. *H records componentwise* hsame *transport for the shared bracket route, the two witness rows, the readback rows, the comparison row, and the local naming row.*
8. *C records displayed* Cont *replay through the route $B, M, S, Z_0, Z_1, W_0, W_1, R_0, R_1, D, E, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedZeroUniqueness}^{\uparrow}$, $\mathsf{BishopLocatedZero}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedZeroUniqueness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed separated-zero route. It has no coordinate for host equality of roots, a selected root, trichotomy, countable choice, open-cover compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.LocatedZeroUniquenessUp`

Theorem 6397.2 (Located zero uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ *carrier* $U = (B, M, S, Z_0, Z_1, W_0, W_1, R_0, R_1, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedZeroUniqueness}^{\uparrow}}$ *surface is exhausted by shared located-zero bracket admission through* B, M, S*, two displayed zero-witness rows through* Z_0, Z_1*, StreamName exposure through* W_0, W_1*, RegSeqRat readback through* R_0, R_1*, dyadic tolerance accounting through* D*, separated comparison through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6397.1. The shared data B, M, S is the same displayed located-zero route named in chapter 6387, so the comparison cannot begin from two unrelated host roots.

The two witness rows are then read symmetrically. Rows Z_0, Z_1 display the two zero witnesses, W_0, W_1 expose the finite stream windows, R_0, R_1 give the regular-rational readbacks, and D supplies the dyadic tolerance ledger consumed by the comparison row E.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no root-selection, trichotomy, choice, compactness, host-equality, quotient-equality, or completeness coordinate, none of those objects can become a local NameCert obligation. □

Theorem 6397.3 (Located zero separated comparison envelope)**.** *Let* $U = (B, M, S, Z_0, Z_1, W_0, W_1, R_0, R_1, D, E,$ *be an accepted* $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ *carrier. Every downstream consumer that compares the two displayed zero reads must factor the comparison through the common* $\mathsf{BishopLocatedZero}^{\uparrow}$ *route* B, M, S*, the two displayed zero-witness rows* Z_0, Z_1*, the two* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* W_0, W_1, R_0, R_1*, the dyadic tolerance ledger* D*, and the* $\mathsf{RealEquality}^{\uparrow}$ *or* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row* E *before it reaches transport, replay, provenance, and naming rows. The exported result is classifier equality over this displayed separated-zero route, not host equality of roots or a selected global zero.*

Proof. Start with the shared located-zero route. Rows B, M, S are the only admissible bracket, modulus, and bisection schedule for both displayed witnesses, so a consumer cannot substitute an ambient intermediate-value theorem or a separate root-selection principle.

Next read both witnesses through the same finite observation discipline. The rows Z_0, Z_1 provide the two zero-witness displays, W_0, W_1 and R_0, R_1 expose the finite stream and regular-rational routes, and D records the tolerance ledger.

Finally apply the separated comparison row E and carry the result through H, C, P, N. The comparison is the same kind of displayed equality boundary isolated by chapter 1636 and the separated classifier discipline of chapter 1411; it does not export host root equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ consumer that compares two displayed zero reads while omitting the shared BishopLocatedZero bracket route, one of the two zero-witness rows, one of the StreamName or RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the separated comparison row, hsame, Cont, Pkg, or NameCert has left the packet.

Independence witness: bishoplocatedzero, realequality, separatedmetric, streamname, regseqrat, dyadicratcore: $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ is not bijective with any listed sibling because it joins one shared located-zero route, two displayed zero-witness rows, paired readbacks, a dyadic comparison ledger, separated classifier equality, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedZeroUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedZeroUniqueness}^{\uparrow}$ packet is generated from one shared BishopLocatedZeroUp bracket, modulus, and bisection schedule, two displayed zero-witness rows, paired StreamName and RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealEqualityUp or SeparatedMetricUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedZeroUniqueness}^{\uparrow}}$, seed)

Scope closed: Definition 6397.1, Definition 6397.2, and Definition 6397.3 seed the finite classifier route for comparing two displayed located-zero reads over one shared L10 bracket surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedZeroUniqueness}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.LocatedZeroUniquenessUp`

Bridge status: none

Not claimed: This chapter does not claim host equality of roots, a selected global zero, trichotomy, quotient real equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification of the obligation theorems, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared bracket admission, two zero-witness rows, paired readbacks, dyadic comparison accounting, separated classifier comparison, transport, replay, provenance, and local naming for BEDC.Derived.LocatedZeroUniquenessUp.

Chapter 6398

A Concrete Naming Certificate for $\mathsf{CountablyCompact}^{\uparrow}$

Following Kelley and Bourbaki on countable open-cover compactness, the $\mathsf{CountablyCompact}^{\uparrow}$ packet records the traditional countably compact surface as finite NameCert data. It reads the $\mathsf{Topology}^{\uparrow}$ source row of chapter 88, the compactness boundary of chapter 60, the finite window discipline of chapter 126, the countable-cover consumer surface of chapter 3914, and the basis-counting route of chapter 1901. The packet names the countable-cover interface without making arbitrary open-cover compactness a primitive coordinate.

Definition 6398.1 (Countably compact carrier)**.** *A* $\mathsf{CountablyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, C, W, S, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the carrier, indexed open rows, finite-intersection read-back, and open-membership classifier used by the packet.*
2. *C is the countable open-cover enumeration row. It records the inspected StreamName schedule of open indices and the displayed statement that the named countable family covers the source row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite cover-window row, listing finite prefixes and the pointwise cover checks visible inside each prefix.*
4. *S is the finite-subcover ledger. It records the accepted prefix whose listed open rows cover the same source surface, together with the finite witness that no later enumeration row is needed for that cover read.*
5. *H stores componentwise* hsame *transport for topology, countable-cover enumeration, finite windows, finite-subcover ledger, replay, provenance, and local naming rows.*
6. *R stores displayed* Cont *replay from a countable-cover request through T, C, W, S, H.*
7. *P is the* Pkg *provenance over* $\mathsf{CountablyCompact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LindelofSpace}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CountablyCompact}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, selected ultrafilters, countable-choice extraction of subcovers, host powersets, quotient open equality, or a standard bridge.

Lean4 (statement) `BEDC.Derived.CountablyCompactUp`

Theorem 6398.2 (Countably compact NameCert obligations)**.** *For every accepted* $\mathsf{CountablyCompact}^{\uparrow}$ *carrier* $K = (T, C, W, S, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CountablyCompact}^{\uparrow}}$ *surface consists exactly of topology source admission, countable open-cover enumeration, StreamName finite cover windows, finite-subcover ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every countable-cover consumer must read* T, C, W, S *before the structural rows* H, R, P, N *can be consumed.*

Proof. Project the carrier of Definition 6398.1. The topology row T is the only source surface, and the countable cover row C is the only row that records an open-cover enumeration.

The finite evidence is downstream of that enumeration. Row W exposes finite StreamName windows of the countable cover, and S records the finite prefix whose displayed open rows cover the same source row.

Finally H, R, P, N transport, replay, source, and name exactly those rows. Since no field stores arbitrary open covers, ultrafilter convergence, selected subcovers, host powersets, quotient open equality, or a bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 6398.3 (Countably compact StreamName cover route)**.** *Every accepted* $\mathsf{CountablyCompact}^{\uparrow}$ *cover read factors through the finite route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{CountablyCompact}^{\uparrow} \longrightarrow \mathsf{LindelofSpace}^{\uparrow} \longrightarrow \mathsf{SecondCountable}^{\uparrow}$$

whenever a downstream countability or basis consumer asks for countable-cover compactness. The route exports only the displayed topology source row, countable open-cover enumeration, finite StreamName windows, finite-subcover ledger, transport, replay, provenance, and local naming rows.

Proof. Start with the accepted packet $K = (T, C, W, S, H, R, P, N)$. A cover read first inspects T and C, so the cover is a displayed countable family over the named topology row rather than an arbitrary host cover.

The StreamName handoff is the finite evidence layer. Row W provides the finite prefix windows, and S records the finite-subcover ledger consumed by countable-cover readers.

The downstream Lindelof and second-countable surfaces receive only that finite ledger through H, R, P, N. Thus the route is a displayed countable-cover compactness packet and not an open-cover compactness primitive. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountablyCompact}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CountablyCompact}^{\uparrow}$ packet is generated from a Topology source row, a countable open-cover enumeration, finite StreamName cover windows, a finite-subcover ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CountablyCompact}^{\uparrow}}$, seed)

Scope closed: Definition 6398.1, Definition 6398.2, and Definition 6398.3 seed the countable-cover compactness carrier over Topology, Compact, StreamName, LindelofSpace, SecondCountable, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CountablyCompact}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, ultrafilter compactness, selected finite subcovers by countable choice, host powerset topology, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable-cover enumeration, StreamName finite-window exactness, finite-subcover ledger readback, transport, replay, provenance, and local naming for BEDC.Derived.CountablyCompact

Chapter 6399

A Concrete Naming Certificate for $\mathsf{ObservableOpenFilter}^{\uparrow}$

Following Sierpinski, Kelley, and Bourbaki on observable opens and filters, the $\mathsf{ObservableOpenFilter}^{\uparrow}$ packet records the open-membership data needed by compact-uniform-continuity cover reads as a finite NameCert surface. It consumes the Sierpinski classifier of chapter 1860, the topology source row of chapter 88, the finite StreamName windows of chapter 126, and the uniform-modulus consumer boundary of chapter 369. The packet keeps open membership as displayed observable truth rather than as a host proposition.

Definition 6399.1 (Observable open-filter carrier)**.** *An* $\mathsf{ObservableOpenFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, O, W, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *classifier row, exposing observable truth marks for open-membership reads.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row whose indexed opens are filtered by the packet.*
3. *O is the observable-open filter row. It records the directed finite family of open-index requests, refinement witnesses, and compatibility with the Sierpinski truth row.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, listing the observed open-membership requests available to compact cover readers.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *cover-read handoff row, recording the tolerance-facing consumer that reads the finite open filter before it consumes a compact-uniform modulus route.*
6. *H stores componentwise* hsame *transport for the Sierpinski classifier, topology source, observable-open filter, finite windows, uniform-modulus handoff, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from an open-filter request through S, T, O, W, U, H.*
8. *P is the* Pkg *provenance over* $\mathsf{ObservableOpenFilter}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservableOpenFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open-membership propositions, excluded middle, selected neighbourhood filters, arbitrary powerset opens, quotient open equality, countable-choice schedules, or bridge credentials.

Lean4 (statement) `BEDC.Derived.ObservableOpenFilterUp`

Theorem 6399.2 (Observable open-filter NameCert obligations)**.** *For every accepted* $\mathsf{ObservableOpenFilter}^{\uparrow}$ *carrier* $F = (S, T, O, W, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ObservableOpenFilter}^{\uparrow}}$ *surface consists exactly of Sierpinski classifier admission, Topology source admission, observable-open filter display, StreamName finite windows, UniformModulus cover-read handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-uniform cover consumer must read* S, T, O, W, U *before it consumes* H, C, P, N*.*

Proof. Project the carrier of Definition 6399.1. Rows S and T supply the only observable-open classifier and topology source visible to the packet.

The filter data are finite and displayed. Row O names the directed open-index requests and refinement witnesses, while W records the finite StreamName windows through which compact cover readers observe those requests.

The uniform-modulus handoff U is downstream of those observable-open rows. The structural rows H, C, P, N then transport, replay, source, and name the same packet. Since no field stores host propositions, selected neighbourhood filters, powerset opens, quotient open equality, countable choice, or a bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 6399.3 (Observable open-filter Sierpinski handoff)**.** *Every open-membership read exported by an accepted* $\mathsf{ObservableOpenFilter}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{SierpinskiSpace}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{UniformModulus}^{\uparrow}.$$

Thus compact-uniform cover reads receive observable Sierpinski classifier data and finite open windows, not host propositions or selected neighbourhood filters.

Proof. Read the accepted packet $F = (S, T, O, W, U, H, C, P, N)$. The Sierpinski row S classifies each open-membership request before the topology row T is consumed, so the route begins with observable truth data.

The filter row O and finite window row W then display which open-index requests are available to a cover reader. The handoff row U records the uniform-modulus consumer only after those finite observable-open rows have been inspected.

Transport, replay, provenance, and naming through H, C, P, N preserve this dependency order. Therefore no compact-uniform cover read can bypass the Sierpinski classifier by importing host propositions, selected neighbourhood filters, powerset opens, quotient open equality, countable choice, or a bridge credential. □

Theorem 6399.4 (Observable open-filter rounded Cauchy consumption)**.** *Any completion-facing consumer that reads an accepted* $\mathsf{ObservableOpenFilter}^{\uparrow}$ *packet before a* $\mathsf{RoundedCauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *route must expose the* $\mathsf{SierpinskiSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *rows first. The handoff into the rounded filter base is allowed only through the displayed open-membership windows of that packet. It cannot import host open predicates, selected neighbourhood filters, quotient filter equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 6399.1. The only rows available before the structural transport, replay, provenance, and local naming coordinates are the Sierpinski classifier S, topology source T, observable-open filter row O, StreamName windows W, and UniformModulus handoff U.

Apply Definition 6399.2. It forces every compact-uniform cover consumer to read S, T, O, W, U before it can consume H, C, P, N. Apply Definition 6399.3 to identify the observable route as the displayed Sierpinski-to-uniform-modulus chain.

The downstream completion route is the rounded Cauchy filter packet of chapter 2016. By Definition 2016.4, its completion-facing read consumes only the displayed rounded-base, stream-window, regular-readback, dyadic-ledger, completion-handoff, and real-seal rows. Therefore an $\mathsf{ObservableOpenFilter}^{\uparrow}$ consumer can feed that route only through displayed open-membership windows, not through host open predicates, selected neighbourhood filters, quotient filter equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservableOpenFilter}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservableOpenFilter}^{\uparrow}$ packet is generated from a SierpinskiSpace observable-open classifier row, a Topology source row, directed observable-open filter data, finite StreamName windows, a UniformModulus cover-read handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservableOpenFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6399.1, Definition 6399.2, Definition 6399.3, and Definition 6399.4 seed the observable-open filter carrier over SierpinskiSpace, Topology, StreamName, UniformModulus, BHist, hsame, Cont, Pkg, NameCert, and rounded Cauchy completion-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservableOpenFilter}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim host open-membership propositions, selected neighbourhood filters, powerset opens, quotient open equality, countable choice, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked classifier admission, topology source admission, observable-open filter exactness, finite-window readback, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.ObservableOpenFilterUp.

Chapter 6400

A Concrete Naming Certificate for $\mathsf{PellEquation}^{\uparrow}$

Following Lagrange's continued-fraction treatment of Pell equations, the $\mathsf{PellEquation}^{\uparrow}$ packet records the finite certificate surface for integer solutions of $x^2 - Dy^2 = 1$ with D a positive nonsquare parameter. It is read against the integer row of chapter 31, the rational row of chapter 37, the quadratic-form surface of chapter 5112, the convergent ledger of chapter 1777, and the period ledger of chapter 1961. The packet names the solution route; it does not choose an infinite continued fraction stream, assert unique minimality from a host theorem, import quotient equality of solution sets, or add an ambient number-theory oracle.

Definition 6400.1 (Pell equation carrier)**.** *A* $\mathsf{PellEquation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (D, X, Y, N, Q, V, L, H, C, K, M)$$

with the following displayed rows:

1. *D is the nonsquare parameter row. It records a positive* $\mathsf{Int}^{\uparrow}$ *value together with a displayed refusal of a square root inside the integer ledger.*
2. *X and Y are integer solution rows, read through chapter 31, carrying the displayed candidate pair for the equation.*
3. *N is the quadratic norm row $x^2 - Dy^2$, read through the* $\mathsf{QuadraticForm}^{\uparrow}$ *surface of chapter 5112.*
4. *Q is the rational comparison row, read through chapter 37, used only to inspect finite convergent quotients and the equation value* 1*.*
5. *V is the* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *row listing the finite convergents inspected by the certificate.*
6. *L is the* $\mathsf{PeriodicContinuedFraction}^{\uparrow}$ *period row displaying the repeated finite block attached to $\sqrt{D}$.*
7. *H records componentwise* hsame *transport for the parameter, integer solution rows, norm row, rational comparison row, convergent ledger, period ledger, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Pell-equation request through D, X, Y, N, Q, V, L, H.*
9. *K is the* Pkg *provenance over* $\mathsf{PellEquation}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{QuadraticForm}^{\uparrow}$*,* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$*,* $\mathsf{PeriodicContinuedFraction}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. M is the local $\mathsf{NameCert}_{\mathsf{PellEquation}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. The carrier has no coordinate for a host-level solution set, selected infinite period stream, quotient equality of all solutions, or external decidability principle.

Lean4 ✓ `BEDC.Derived.PellEquationUp.PellEquationTasteGate single carrier alignment`

Theorem 6400.2 (Pell equation NameCert obligations)**.** *For every accepted* $\mathsf{PellEquation}^\uparrow$ *carrier*

$$P = (D, X, Y, N, Q, V, L, H, C, K, M),$$

the local $\mathsf{NameCert}_{\mathsf{PellEquation}^\uparrow}$ *surface consists exactly of nonsquare parameter admission through D, integer solution-row admission through X, Y, quadratic norm evaluation through N, rational comparison through Q, finite convergent exposure through V, finite period exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through K, and local naming through M.*

No Pell equation obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PellEquationUp.PellEquationCarrier namecert obligations`

Proof. Project the accepted carrier. Its analytic rows are exactly the nonsquare parameter, the two integer solution rows, the quadratic norm, the rational comparison row, the convergent ledger, and the period ledger.

The equation check is read from the norm row N. Since N is the displayed quadratic form $x^2 - Dy^2$ and Q compares its value with the rational unit, the certificate verifies the Pell equation only by inspecting rows already present in the packet.

The continued-fraction data is finite. The convergent ledger V and period ledger L record the Lagrange route used to source candidate integer rows, but neither row supplies an infinite stream or a global selector.

Finally, H, C, K, M transport, replay, source, and name the same displayed route. Since the carrier has no remaining coordinate, the local NameCert obligations are exhausted by the listed rows. □

Theorem 6400.3 (Pell equation norm preservation)**.** *Let $P = (D, X, Y, N, Q, V, L, H, C, K, M)$ be an accepted* $\mathsf{PellEquation}^\uparrow$ *carrier. Any* hsame/Cont *replay of the packet that preserves the nonsquare parameter row, the two integer solution rows, the continued-fraction convergent row, and the period row preserves the displayed quadratic norm row N and therefore preserves the public equation witness $x^2 - Dy^2 = 1$.*

The preservation route factors through $\mathsf{Int}^\uparrow$, $\mathsf{Rat}^\uparrow$, $\mathsf{QuadraticForm}^\uparrow$, $\mathsf{ContinuedFractionConvergents}^\uparrow$, *and* $\mathsf{PeriodicContinuedFraction}^\uparrow$ *before the local* $\mathsf{NameCert}_{\mathsf{PellEquation}^\uparrow}$ *row is read. It exports no host solution-set equality, selected infinite continued fraction, quotient class of solutions, or ambient number-theory decision oracle.*

Lean4 ✓ `BEDC.Derived.PellEquationUp.PellEquation norm preservation`

Proof. Start with the carrier projection. The norm row N is tied to the displayed parameter and solution rows D, X, Y, so a replay preserving those rows also preserves the inputs to the quadratic-form evaluation.

Use the finite Lagrange route next. The rows V and L show which finite convergents and period block were inspected before X, Y were accepted. Preserving those rows keeps the same finite source for the solution witness.

The rational comparison row Q reads the displayed norm value against the unit. Since H transports componentwise and C replays the same displayed coordinates, the comparison remains attached to the same norm row.

Thus the public witness $x^2 - Dy^2 = 1$ is preserved as a finite NameCert route. No step introduces a host equality of solution sets, an infinite continued fraction object, a quotient class, or an external decision principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PellEquation}\uparrow}$.

Constructive story: A $\mathsf{PellEquation}^{\uparrow}$ packet is generated from a nonsquare IntUp parameter row, two IntUp solution rows, a QuadraticFormUp norm row, a RatUp comparison row, ContinuedFractionConvergentsUp and PeriodicContinuedFractionUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PellEquation}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6400.1, Definition 6400.2, and Definition 6400.3 seed the finite Pell-equation certificate surface over integer rows, rational comparison, quadratic norm evaluation, continued-fraction convergents, and periodic continued-fraction data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PellEquation}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host solution-set equality, selected infinite continued fraction, quotient class of all Pell solutions, ambient number-theory decision oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked parameter admission, integer solution-row admission, norm evaluation, convergent-period route, transport stability, replay stability, and nonescape for BEDC.Derived.PellEquationUp.

Chapter 6401

A Concrete Naming Certificate for $\mathsf{WeakStarCompactness}^{\uparrow}$

Following Bourbaki and Kelley on locally convex spaces and weak-star compactness, the $\mathsf{WeakStarCompactness}^{\uparrow}$ packet records the finite dual-ball compactness interface that sits between the weak-star observation rows of chapter 5192 and the finite dual-ball handoff of chapter 3420. It is read over the locally convex source of chapter 3386 and the compactness carrier of chapter 60. The chapter names the compactness-facing dual-ball route as finite NameCert data; it does not identify a host dual space, choose cluster points, invoke ultrafilters, or export unrestricted product compactness.

Definition 6401.1 (Weak-star compactness carrier)**.** *A* $\mathsf{WeakStarCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, T, B, A, C, R, H, Q, P, N)$$

with the following displayed rows:

1. *E is the locally convex source row, read through the finite seminorm and convex balanced neighbourhood surface of* $\mathsf{LocallyConvexSpace}^{\uparrow}$*.*
2. *T is the weak-star topology row, exposing only the finite functional observation windows admitted by* $\mathsf{WeakStarTopology}^{\uparrow}$*.*
3. *B is the finite dual-ball row, listing the displayed bounded functionals, source probes, scalar bounds, and tolerance rows inspected by the packet.*
4. *A is the Banach–Alaoglu handoff row, used only as the finite dual-ball window described by* $\mathsf{BanachAlaoglu}^{\uparrow}$*.*
5. *C is the compactness row, read through* $\mathsf{Compact}^{\uparrow}$ *as finite-net or located compactness evidence for the displayed window, not as an unrestricted host compactness predicate.*
6. *R is the scalar readback row for real-valued functional values, bounds, and comparisons.*
7. *H records componentwise* hsame *transport for the locally convex source, weak-star topology window, dual-ball row, Banach–Alaoglu handoff, compactness row, scalar readback, replay, provenance, and naming rows.*
8. *Q records displayed* Cont *replay from a compactness-facing dual-ball request through $E, T, B, A, C, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{WeakStarCompactness}^{\uparrow}$*,* $\mathsf{LocallyConvexSpace}^{\uparrow}$*,* $\mathsf{WeakStarTopology}^{\uparrow}$*,* $\mathsf{BanachAlaoglu}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{WeakStarCompactness}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $E, T, B, A, C, R, H, Q, P, N$. It has no coordinate for a chosen dual element, a selected cluster point, ultrafilter extension, quotient equality of functionals, unrestricted Tychonoff compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.WeakStarCompactnessUp.TasteGate.WeakStarCompactnessTasteGate single carrier alignment`

Theorem 6401.2 (Weak-star compactness NameCert obligations)**.** *For every accepted* $\mathsf{WeakStarCompactness}^{\uparrow}$ *carrier $K = (E, T, B, A, C, R, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakStarCompactness}^{\uparrow}}$ *surface consists exactly of locally convex source admission through E, weak-star observation-window admission through T, finite dual-ball admission through B, Banach–Alaoglu finite-window handoff through A, compactness evidence through C, scalar readback through R, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through N.*

Proof. Project the accepted carrier in Definition 6401.1. The rows $E, T, B, A, C, R, H, Q, P, N$ are the complete displayed inventory of the packet.

The source and topology rows are read first. The locally convex source E fixes the finite seminorm and neighbourhood surface, while T restricts weak-star access to finite functional observation windows.

The compactness-facing data then pass through B, A, C, R. The dual-ball row B lists the bounded functional probes, A cites only the Banach–Alaoglu finite-window handoff, C supplies the displayed compactness evidence for that window, and R records the scalar readbacks.

Transport, replay, provenance, and naming are supplied by H, Q, P, N. Since no other carrier coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 6401.3 (Weak-star compactness finite dual-ball route)**.** *Every compactness export from an accepted* $\mathsf{WeakStarCompactness}^{\uparrow}$ *packet factors through the finite route*

$$E, T, B, A, C, R, H, Q, P, N.$$

Thus a functional-analysis consumer may read weak-star compactness only after exposing the locally convex source, the weak-star observation window, the finite dual-ball row, the Banach–Alaoglu handoff, compactness evidence, scalar readback, transport, replay, provenance, and local NameCert rows. The export carries no chosen dual element, no selected cluster point, no ultrafilter extension, no quotient equality of functionals, no unrestricted Tychonoff compactness, and no ambient completeness coordinate.

Proof. Begin with a compactness-facing export. By Definition 6401.2, the export must expose the locally convex source E and the weak-star observation window T before any dual-ball compactness data can be inspected.

The finite dual-ball row B and Banach–Alaoglu handoff row A then determine the displayed functional probes and scalar bounds. Compactness evidence is read only through C, and scalar values and comparisons are read only through R.

The structural rows H, Q, P, N preserve that same route under transport, replay, provenance, and local naming. Because the carrier contains no selection, ultrafilter, quotient, unrestricted product, or ambient completeness row, none of those data can be exported by projection from the accepted packet. □

Falsifiable prediction: Every public $\mathsf{WeakStarCompactness}^{\uparrow}$ compactness read factors through a displayed $\mathsf{LocallyConvexSpace}^{\uparrow}$ source, a finite $\mathsf{WeakStarTopology}^{\uparrow}$ observation window, a finite dual-ball row, a $\mathsf{BanachAlaoglu}^{\uparrow}$ handoff, a $\mathsf{Compact}^{\uparrow}$ compactness row, scalar readback, BHist transport, hsame, Cont, Pkg, and local NameCert naming. A public read that exports selected dual elements, cluster-point choice, ultrafilters, quotient functional equality, unrestricted product compactness, or ambient completeness from this packet refutes the carrier boundary.

Independence witness: weakstartopology, banachalaoglu, locallyconvexspace, compact: none is bijective with $\mathsf{WeakStarCompactness}^{\uparrow}$ because this carrier binds the locally convex source, weak-star finite observation window, finite dual-ball handoff, compactness evidence, scalar readback, transport, replay, provenance, and local NameCert rows while refusing selection and unrestricted compactness coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakStarCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakStarCompactness}^{\uparrow}$ packet is generated from a locally convex source row, a weak-star finite observation window, a finite dual-ball row, a Banach–Alaoglu handoff, compactness evidence, scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakStarCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6401.1, Definition 6401.2, and Definition 6401.3 seed the finite dual-ball compactness interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakStarCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected dual elements, cluster-point choice, ultrafilter extension, quotient functional equality, unrestricted Tychonoff compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-star observation-window admission, finite dual-ball exactness, compactness-row handoff, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.WeakStarCompactnessUp.

Chapter 6402

A Concrete Naming Certificate for $\mathsf{LocatedDistanceFunction}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric spaces, the $\mathsf{LocatedDistanceFunction}^{\uparrow}$ packet records the standard point-to-set distance function as a finite BEDC NameCert surface. It sits over the metric source of chapter 57, the located-set row of chapter 1541, the point-to-set packet of chapter 4364, the lower-semicontinuity sibling of chapter 1536, the located-real comparison surface of **??**, and the terminal real-name seal of chapter 38. The packet names a reusable real-valued function row for located subsets; it does not assert a host infimum, selected nearest point, quotient real equality, countable-choice selector, or ambient completeness principle.

Definition 6402.1 (Located distance function carrier)**.** *A* $\mathsf{LocatedDistanceFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, D, F, R, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *source row naming the subset by located-radius observations.*
3. *D is the* $\mathsf{DistanceToSet}^{\uparrow}$ *row that supplies the point-to-set distance read for each displayed point request.*
4. *F is the function graph row, recording the finite route from a point row in X through A, D to the exported real-name value.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed value of the function.*
6. *O is the lower-semicontinuity and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison ledger, recording lower-real threshold reads, located-real value comparison, and sibling handoff to* $\mathsf{LowerSemicontinuous}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for the metric source, located-set source, distance-to-set row, graph row, Real seal, comparison ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point request through the metric and located-set rows, the distance-to-set row, the graph row, the Real seal, and the lower-semicontinuity ledger.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedDistanceFunction}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{LocatedSet}^{\uparrow}$*,* $\mathsf{DistanceToSet}^{\uparrow}$*,* $\mathsf{LowerSemicontinuous}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDistanceFunction}^{\uparrow}}$ *row.*

The classifier accepts only packets whose graph values are read through $X, A, D, F, R, O, H, C, P, N$. It has no coordinate for a host function space, selected nearest point, ambient infimum, quotient real equality, or countable-choice schedule.

Theorem 6402.2 (Located distance function NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDistanceFunction}^{\uparrow}$ *carrier $L = (X, A, D, F, R, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDistanceFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located-set source, distance-to-set row, function graph, Real seal, lower-semicontinuity and located-real comparison ledger, transport, replay, provenance, and local naming;*
2. *point-request routing, requiring every value read to start in X and pass through the located subset row A;*
3. *distance-to-set handoff, requiring the value row to consume the* $\mathsf{DistanceToSet}^{\uparrow}$ *packet of chapter 4364 before the graph row F is public;*
4. *Real and located-real sealing, requiring R and O to record only displayed values and comparisons;*
5. *lower-semicontinuity exposure, requiring the lower-threshold route to pass through the sibling surface of chapter 1536;*
6. *componentwise transport, displayed replay, package provenance, and local naming for exactly the displayed packet.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 6402.1. The source row X supplies points, the located-set row A supplies located-radius observations, and the distance row D supplies the point-to-set value route.

The graph row F does not introduce a separate host function. It records the replay from point requests to the displayed distance row and then to the Real seal R.

The ledger O records the lower-semicontinuity and located-real comparison data after the value row is sealed. Transport H, replay C, provenance P, and local naming N preserve the same row family, so the listed obligations exhaust the certificate surface. □

Theorem 6402.3 (Located distance function lower semicontinuity)**.** *Every accepted* $\mathsf{LocatedDistanceFunction}^{\uparrow}$ *carrier $L = (X, A, D, F, R, O, H, C, P, N)$ exports lower-semicontinuity only through the displayed route*

$$X, A, D, F, R, O, H, C, P, N.$$

The lower-threshold read consumes the $\mathsf{DistanceToSet}^{\uparrow}$ *value row and the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison ledger before it reaches the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *sibling surface. It cannot be replaced by a host liminf, selected nearest point, quotient real equality, countable-choice selector, or ambient completeness theorem.*

Proof. Start with a lower-threshold request for the function value. By Definition 6402.2, the request first reads the metric source, located subset, distance-to-set row, graph row, and Real seal.

The ledger O then exposes the located-real comparison needed by the lower-semicontinuity sibling. Since O is downstream of D, F, R, the lower-threshold route remains tied to the same displayed point-to-set distance value.

Finally H, C, P, N transport, replay, source, and name that finite route. The carrier has no coordinate for a host limiting theorem or chosen nearest point, so the exported lower-semicontinuity surface is exactly the displayed BEDC handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDistanceFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDistanceFunction}^{\uparrow}$ packet is generated from metric source, located-set source, DistanceToSet row, function graph, Real seal, lower-semicontinuity and located-real comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDistanceFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6402.1, Definition 6402.2, and Definition 6402.3 seed the finite real-valued located-distance-function surface over displayed point-to-set distance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDistanceFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host function-space construction, host infimum, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, point-request routing, distance-to-set handoff, Real and located-real sealing, lower-semicontinuity exposure, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDistanceFunctionUp.

Chapter 6403

A Concrete Naming Certificate for $\mathsf{HermitianMatrix}^{\uparrow}$

Following Horn and Johnson on matrix analysis, the $\mathsf{HermitianMatrix}^{\uparrow}$ packet records the finite BEDC surface for a square complex matrix displayed together with its conjugate-transpose and self-adjointness rows. It sits after the matrix carrier of chapter 49, the complex scalar rows of chapter 39, the vector-space source of chapter 47, the field source of chapter 45, and the local naming discipline exemplified by chapter 481. The packet names finite Hermitian-matrix evidence; it does not assert a spectral theorem, diagonalization, eigenbasis selection, positive semidefiniteness, quotient matrix equality, or host matrix equality.

Definition 6403.1 (Hermitian matrix carrier)**.** *A* $\mathsf{HermitianMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (D, F, V, M, K, T, S, E, C, P, L)$$

with the following displayed rows:

1. *D is the finite dimension row naming the shared square index window for matrix entries.*
2. *F is the consumed* $\mathsf{Field}^{\uparrow}$ *scalar-source row through which addition, multiplication, zero, and one are read.*
3. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *coordinate source over the same scalar rows.*
4. *M is the square* $\mathsf{Mat}^{\uparrow}$ *entry ledger, indexed by $D \times D$ and carrying complex-valued entry histories.*
5. *K is the* $\mathsf{Complex}^{\uparrow}$ *conjugation row used for each displayed entry of M.*
6. *T is the conjugate-transpose row, recording for every displayed index pair (i, j) the row that reads $\overline{M_{j,i}}$ from K and the transposed entry of M.*
7. *S is the self-adjoint equality row, identifying each displayed entry $M_{i,j}$ with the corresponding entry of T through componentwise* hsame *transport.*
8. *E records componentwise transport for the dimension, field, vector-space, matrix-entry, conjugation, transpose, self-adjointness, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from finite matrix-entry inspection through conjugation, transpose readback, and self-adjointness checking.*
10. *P is the* Pkg *provenance over* $\mathsf{HermitianMatrix}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. L is the local $\mathsf{NameCert}_{\mathsf{HermitianMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, F, V, M, K, T, S, E, C, P, L$. The carrier has no coordinate for spectral decomposition, eigenvalue ordering, positive-semidefinite witness, quotient matrix equality, selected basis, or ambient host equality.

Theorem 6403.2 (Hermitian matrix NameCert obligations)**.** *For every accepted* $\mathsf{HermitianMatrix}^{\uparrow}$ *carrier*

$$H = (D, F, V, M, K, T, S, E, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HermitianMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission, finite square dimension exposure, scalar-field source readback, vector-space coordinate exposure, matrix-entry ledger admission, complex conjugation readback, conjugate-transpose display, self-adjoint entry comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6403.1. The accepted packet exposes exactly the dimension, field, vector-space, matrix, conjugation, transpose, self-adjointness, transport, replay, provenance, and naming rows.

The conjugate-transpose obligation is finite. The row T is read entry by entry from D, the transposed matrix-entry row of M, and the complex conjugation row K.

The self-adjointness obligation is also componentwise. The row S compares each displayed $M_{i,j}$ with the displayed conjugate-transpose entry through hsame, and E transports only those rows.

Finally C, P, L keep every replay and downstream consumer behind the same finite packet. Since no carrier row stores eigenvalues, selected eigenvectors, positive-semidefinite data, quotient matrix equality, a basis selector, or host matrix equality, the listed rows exhaust the local NameCert obligations. □

Theorem 6403.3 (Hermitian matrix self-adjointness boundary)**.** *Every consumer that reads an accepted* $\mathsf{HermitianMatrix}^{\uparrow}$ *packet as self-adjoint matrix data must pass through the displayed* $\mathsf{Mat}^{\uparrow}$ *ledger M, the* $\mathsf{Complex}^{\uparrow}$ *conjugation row K, the conjugate-transpose row T, and the componentwise self-adjointness row S over the same finite dimension, field, and vector-space rows D, F, V. The packet alone cannot export spectral theorem data, diagonalization, eigenvalue selection, positive-semidefinite evidence, quotient matrix equality, selected basis data, or host matrix equality.*

Proof. Start with the NameCert obligations of Definition 6403.2. They expose the matrix ledger, complex conjugation, conjugate-transpose readback, self-adjointness comparison, transport, replay, provenance, and local naming rows.

Use the carrier definition to keep those rows over one square index window. The dimension row D fixes the entry domain, while F, V fix the scalar and coordinate sources through which matrix reads are typed.

The self-adjointness read cannot bypass T or S. The transpose row T computes the conjugate-transpose entry from displayed rows, and S is the only row comparing that entry with the original matrix entry.

Thus any self-adjoint consumer receives only the finite Hermitian matrix packet. Spectral theorem data, diagonalization, selected eigenvectors, positive-semidefinite evidence, quotient matrix equality, selected bases, and host equality are absent from the carrier and cannot be recovered by replay. □

Closure status for $\mathsf{NameCert}_{\mathsf{HermitianMatrix}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HermitianMatrix}^{\uparrow}$ packet is generated from a finite square index row, scalar-field and vector-space sources, a complex matrix-entry ledger, a conjugation row, a conjugate-transpose row, componentwise self-adjointness evidence, hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{HermitianMatrix}^{\uparrow}}$, seed)

Scope closed: Definition 6403.1, Definition 6403.2, and Definition 6403.3 seed the finite Hermitian matrix packet over displayed Mat$^{\uparrow}$, Complex$^{\uparrow}$, VecSpace$^{\uparrow}$, Field$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{HermitianMatrix}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close spectral theorem data, diagonalization, eigenbasis selection, positive-semidefinite evidence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, conjugation readback, conjugate-transpose exactness, self-adjoint component comparison, transport, replay, provenance, and local naming for the same finite packet.

Chapter 6404

A Concrete Naming Certificate for $\mathsf{DiniModulusContinuity}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{DiniModulusContinuity}^{\uparrow}$ packet records a finite route from displayed Dini-style local modulus requests to Real-facing continuity reads. It sits between the ordinary modulus-control surface of chapter 1631, the derivative-side Dini packet of chapter 2041, the finite StreamName windows of chapter 126, the regular rational readback rows of chapter 301, the dyadic tolerance budget of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the finite continuity-modulus handoff only; it does not assert ambient Real completeness, selected limits, quotient streams, or a host-level continuity principle.

Definition 6404.1 (Dini modulus continuity carrier)**.** *A* $\mathsf{DiniModulusContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the source* $\mathsf{StreamName}^{\uparrow}$ *window ledger. It stores the finite source windows on which the continuity request is allowed to read names.*
2. *R is the Dini modulus request row. It records the displayed neighbourhood request, the local oscillation bound, and the monotone precision schedule used by the packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. It displays the input tolerance, the output tolerance, the local split cells, and the finite descent route binding them to R.*
4. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the rational observations returned from the accepted StreamName windows under the dyadic tolerance budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is reachable only after the StreamName windows, Dini request row, dyadic budget, and rational readback row have been displayed.*
6. *H records componentwise* hsame *transport for W, R, D, T, E, C, P, N.*
7. *C records displayed* Cont *replay from an outer continuity-modulus request through W, R, D, T, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DiniModulusContinuity}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{DiniDerivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniModulusContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $W, R, D, T, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, host limits, ambient Real completeness, selected representatives, countable choice, Lean verification, or bridge checking.

Theorem 6404.2 (Dini modulus continuity NameCert obligations)**.** *For every accepted* $\mathsf{DiniModulusContinuity}^{\uparrow}$ *carrier*

$$M = (W, R, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniModulusContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName window ledger, Dini modulus request, dyadic tolerance budget, RegSeqRat readback row, Real seal, transport row, replay row, provenance row, and local naming row;*
2. *source-window exactness, requiring every observation used by the modulus request to appear in W;*
3. *Dini-request exactness, requiring the neighbourhood request, oscillation bound, and monotone precision schedule to be the displayed cells of R;*
4. *dyadic budget compatibility, requiring D to bind the input and output tolerances to the finite local split cells consumed by R and T;*
5. *rational readback exactness, requiring every exported rational observation to come from T after W, R, D have been displayed;*
6. *Real-seal non-escape, requiring E to remain downstream of W, R, D, T and the structural rows;*
7. *componentwise structural preservation, requiring H and C to transport and replay the displayed packet without changing the modulus route;*
8. *provenance exhaustion, requiring P and N to source and name exactly the displayed packet and no quotient stream, host limit, ambient Real completeness, selected representative, countable-choice row, Lean verification, or bridge checker.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6404.1. The projection exposes one source-window ledger, one Dini modulus request row, one dyadic tolerance budget, one regular rational readback row, one terminal Real seal, and the structural rows.

The window and request obligations are read from W and R. A continuity read can inspect only the displayed StreamName windows, and the neighbourhood request, oscillation bound, and monotone precision schedule must be the cells named in R.

The tolerance and readback obligations are then read from D and T. The dyadic budget displays the local split cells used by the Dini request, while T lists the regular rational observations returned from those windows.

Finally, E, H, C, P, N only seal, transport, replay, source, and name the displayed route. Since the carrier has no further analytic coordinate, the local NameCert exports exactly the listed finite obligations and no ambient completion principle. □

Theorem 6404.3 (Dini modulus continuity Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read out of an accepted* $\mathsf{DiniModulusContinuity}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DiniModulusContinuity}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by $W, R, D, T, E, H, C, P, N$. Thus a Real and Completion consumer can use the packet only as a displayed modulus-control handoff, not as a claim of quotient-stream equality, selected limits, host continuity, or ambient Real completeness.

Proof. Take a Real-facing read admitted by the packet. Projection of M gives no coordinate from an outer request directly to the terminal seal E.

The read first opens the StreamName windows and the Dini modulus request. The row W displays the finite windows, and R displays the local neighbourhood request and oscillation bound that governs those windows.

The dyadic tolerance budget and regular rational readback determine the numeric data. The row D records the local split cells, while T records the rational observations consumed under those tolerances.

Only after those rows are visible can E be consumed. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same route, so no quotient stream, selected limit, host continuity service, or ambient Real completeness claim can be inserted. □

Falsifiable prediction: An accepted $\mathsf{DiniModulusContinuity}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying the $\mathsf{StreamName}^{\uparrow}$ windows, Dini modulus request, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, and $\mathsf{RegSeqRat}^{\uparrow}$ readback row refutes the carrier surface.

Independence witness: moduluscontinuity, dini-derivative, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole packet $M = (W, R, D, T, E, H, C, P, N)$, because each sibling contributes only ordinary modulus control, derivative-side Dini data, window data, rational readback, tolerance budgeting, or terminal seal data while this chapter binds them into one finite continuity-modulus NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniModulusContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniModulusContinuity}^{\uparrow}$ packet is generated from StreamName source windows, a Dini modulus request, a DyadicRatCore tolerance budget, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniModulusContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6404.1, Definition 6404.2, and Definition 6404.3 seed the finite Dini modulus continuity route over StreamName, DyadicRatCore, RegSeqRat, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniModulusContinuity}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host limits, ambient Real completeness, selected representatives, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Dini-request exactness, dyadic-budget compatibility, RegSeqRat readback exactness, Real-seal non-escape, classifier transport, and ledger non-escape proofs for BEDC.Derived.DiniModulusContinuityUp.

Chapter 6405

A Concrete Naming Certificate for FiniteDiagonalModulusScheduler$^{\uparrow}$

Following Bishop–Bridges constructive completion practice, the FiniteDiagonalModulusScheduler$^{\uparrow}$ packet records the finite diagonal-window scheduler used before a Real-facing regular Cauchy read is sealed. It sits over the StreamName window route of chapter 126, the rational regularity rows of chapter 301, the dyadic tolerance budget of chapter 291, the tolerance-comparison discipline of chapter 1211, and the terminal real-name seal of chapter 38. The packet exposes displayed diagonal scheduling and tolerance data only; it does not choose a limit, quotient streams, selected representatives, or ambient Real completeness.

Definition 6405.1 (Finite diagonal modulus scheduler carrier)**.** *A* FiniteDiagonalModulusScheduler$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (J, W, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the finite diagonal-window ledger. It lists the outer precision requests, the selected finite diagonal windows, and the inclusion order among those windows.*
2. *W is the* StreamName$^{\uparrow}$ *schedule row. It records the StreamName reads licensed by each diagonal window of J.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance split ledger. It displays the outer tolerance, the window tolerance, the endpoint tolerance, and the finite dyadic descent cells.*
4. *R is the* RegSeqRat$^{\uparrow}$ *endpoint readback row. It records the regular rational observations returned at the scheduled endpoints.*
5. *L is the* DyadicToleranceTriangleLedger$^{\uparrow}$ *handoff row. It records the displayed comparison triangle binding the split tolerances of D to the endpoint reads of R.*
6. *E is the terminal* Real$^{\uparrow}$ *seal. It is reachable only after the diagonal windows, StreamName schedule, dyadic tolerance split, endpoint readbacks, and triangle handoff have been displayed.*
7. *H records componentwise* hsame *transport for $J, W, D, R, L, E, C, P, N$.*
8. *C records displayed* Cont *replay from an outer diagonal regularity request through $J, W, D, R, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* FiniteDiagonalModulusScheduler$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicToleranceTriangleLedger$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $J, W, D, R, L, E, H, C, P, N$. The carrier has no coordinate for selected limits, quotient stream equality, ambient Real completeness, countable choice, Lean verification, or bridge checking.

Theorem 6405.2 (Finite diagonal modulus scheduler NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}$ *carrier*

$$S = (J, W, D, R, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite diagonal-window ledger, StreamName schedule, dyadic tolerance split ledger, RegSeqRat endpoint readback row, tolerance-triangle handoff, Real seal, transport row, replay row, provenance row, and local naming row;*
2. *diagonal-window exactness, requiring every scheduled read to be listed in J with its finite inclusion relation;*
3. *StreamName schedule exactness, requiring each read in W to be licensed by a displayed diagonal window;*
4. *dyadic split compatibility, requiring D to display the outer, window, and endpoint tolerances consumed by the schedule;*
5. *endpoint readback exactness, requiring R to display the regular rational observations returned at the scheduled endpoints;*
6. *triangle-handoff exactness, requiring L to bind the split tolerances to the endpoint rows through the displayed dyadic tolerance triangle;*
7. *Real-seal non-escape, requiring E to remain downstream of J, W, D, R, L and the structural rows;*
8. *provenance exhaustion, requiring P and N to source and name exactly the displayed packet and no selected limit, quotient stream, ambient Real completeness, countable-choice row, Lean verification, or bridge checker.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6405.1. The projection exposes the diagonal-window ledger, StreamName schedule, dyadic tolerance split ledger, endpoint readback row, tolerance-triangle handoff, terminal seal, and structural rows.

The scheduling obligations are read from J and W. Every StreamName read is attached to a displayed finite diagonal window, and the inclusion relation among the windows is part of the same finite ledger.

The numeric obligations are read from D, R, L. The tolerance split ledger displays the outer, window, and endpoint tolerances; the endpoint readback row lists the regular rational observations; the triangle handoff binds these cells through the same dyadic comparison route.

Finally, E, H, C, P, N only seal, transport, replay, source, and name the displayed packet. Because no further coordinate is present, the NameCert surface exports exactly the listed finite obligations and no selected limit or ambient completeness principle. □

Theorem 6405.3 (Finite diagonal modulus scheduler Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read out of an accepted* $\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicToleranceTriangleLedger}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by $J, W, D, R, L, E, H, C, P, N$*. Thus a Real and Completion consumer receives displayed diagonal windows and tolerance rows, not a selected limit, quotient-stream representative, or ambient Real completeness claim.*

Proof. Take a Real-facing read admitted by the carrier. Projection of S gives no direct coordinate from an outer diagonal request to the terminal seal E.

The read first passes through the finite diagonal windows and StreamName schedule. The row J displays the diagonal windows, while W records the StreamName reads licensed by those windows.

The dyadic and rational rows then supply the endpoint data. The row D displays the tolerance split, R displays the endpoint readbacks, and L records the tolerance triangle that binds the split cells to those readbacks.

Only after those rows are displayed can the terminal seal E be consumed. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same route, so the packet cannot introduce a selected limit, quotient stream, countable-choice schedule, or ambient Real completeness assertion. □

Falsifiable prediction: An accepted $\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying the finite diagonal windows, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance split, $\mathsf{RegSeqRat}^{\uparrow}$ endpoint readbacks, and $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, dyadic-tolerance-triangle-ledger, real: no listed sibling supplies the whole packet $S = (J, W, D, R, L, E, H, C, P, N)$, because each sibling contributes only windows, endpoint readbacks, tolerance cells, triangle comparison, or terminal seal data while this chapter binds them into one finite diagonal scheduler NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}$ packet is generated from finite diagonal windows, StreamName schedules, DyadicRatCore tolerance splits, RegSeqRat endpoint readbacks, a DyadicToleranceTriangleLedger handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}}$, seed)

Scope closed: Definition 6405.1, Definition 6405.2, and Definition 6405.3 seed the finite diagonal-window scheduler route over StreamName, DyadicRatCore, RegSeqRat, DyadicToleranceTriangleLedger, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteDiagonalModulusScheduler}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, ambient Real completeness, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-window exactness, StreamName schedule exactness, dyadic-split compatibility, endpoint readback exactness, triangle-handoff exactness, Real-seal non-escape, and ledger non-escape proofs for BEDC.Derived.FiniteDiagonalModulusSchedulerUp.

Chapter 6406

A Concrete Naming Certificate for $\mathsf{FiniteVariationPath}^{\uparrow}$

Following Jordan and Bishop–Bridges constructive analysis, the $\mathsf{FiniteVariationPath}^{\uparrow}$ packet records finite-variation data for a displayed metric-valued path by finite partition windows and regular rational variation ledgers. It sits beside the bounded-variation surface of chapter 398, the path-length boundary of chapter 3443, the dyadic tolerance core of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a path-level finite-variation certificate for L10 Real and Completion consumers; it does not import a host path space, a supremum over all partitions, quotient equality of paths, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6406.1 (Finite variation path carrier)**.** *A* $\mathsf{FiniteVariationPath}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (M, \gamma, I, \Pi, D, B, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric and continuous-path source row, exposing the accepted metric carrier, the displayed distance read, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *path graph through the same source and target windows.*
2. *γ is the path-evaluation schedule, recording the finite endpoint probes of the path on the inspected interval row.*
3. *I is the finite interval-domain row supplying the ordered endpoints and the admissible subdivision boundary.*
4. *Π is the finite partition row over I, including adjacency, endpoint membership, and finite common-refinement witnesses for pairs of displayed partitions.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-distance ledger, recording the adjacent distance terms and partial variation sums for the partition window Π.*
6. *B is the variation-sum bound row, comparing the dyadic edge-distance sum in D with the displayed finite bound attached to the accepted path packet.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row by which a downstream regular-rational consumer reads the same edge-distance ledger and bound through finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $M, \gamma, I, \Pi, D, B, Q$ have all been displayed.*

9. *H records componentwise* hsame *transport for the metric source, path schedule, interval, partition, edge-distance ledger, bound row, regular-rational readback, real seal, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite-variation request through path evaluation, partition exposure, dyadic edge-distance summation, common-refinement comparison, regular-rational readback, and the real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteVariationPath}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RectifiableCurve}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationPath}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, \gamma, I, \Pi, D, B, Q, E, H, C, P, N$. It has no coordinate for a host path space, a total variation supremum over all partitions, quotient equality of paths, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6406.2 (Finite variation path NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVariationPath}^{\uparrow}$ *carrier $V = (M, \gamma, I, \Pi, D, B, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationPath}^{\uparrow}}$ *surface consists exactly of carrier admission, metric and continuous-path source exposure, finite interval and partition exposure, dyadic edge-distance ledger readback, variation-sum bound comparison, common-refinement comparison, RegSeqRat readback, RealUp seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Every path-variation consumer in this scope must therefore read the finite route

$$M, \gamma, I, \Pi \longmapsto D, B \longmapsto Q, E$$

together with H, C, P, N, and receives no host path-space object, all-partitions supremum, quotient path equality, hidden selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 6406.1. The visible analytic rows are the metric and path source M, γ, the interval and partition rows I, Π, the dyadic edge-distance ledger D, the bound row B, the regular-rational readback Q, and the real seal E.

The variation request is finite at the displayed partition. The interval row supplies the ordered domain, the path schedule supplies endpoint probes, and the partition row lists the adjacent pairs whose metric distance terms are consumed by D.

The bound comparison is read through B. Common-refinement witnesses live in Π and compare only displayed finite partition ledgers and their dyadic sums, so the classifier does not range over all partitions.

The L10 readback is downstream of the ledger. The row Q returns the inspected variation sum and bound as regular rational data, while E seals that same displayed route as $\mathsf{Real}^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no remaining coordinate names a host path space, quotient path equality, hidden selector, or completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 6406.3 (Finite variation path RegSeqRat–Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteVariationPath}^{\uparrow}$ *carrier factors through the displayed metric-path source row, finite partition row, dyadic edge-distance ledger, variation-sum bound, and common-refinement comparison before it may read the regular-rational row Q or the terminal real-name seal E.*

The handoff is compatible with the bounded-variation and rectifiable-curve neighbors: bounded-variation consumers read only the finite partition-variation ledger, and rectifiable-curve consumers

read only the displayed partition-length route. Neither consumer receives a host path space, an all-partitions supremum, quotient equality of paths, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a RegSeqRat or Real-facing read. The continuation row C first replays the displayed source rows M, γ and the interval-partition rows I, Π, so the downstream consumer is attached to a finite path window before any variation value is exposed.

Next the dyadic ledger D supplies the adjacent distance terms and the partial sum for the displayed partition. The bound row B compares that finite sum with the accepted variation budget, and the common-refinement data in Π compare only finite ledgers over displayed subdivisions.

Only after those rows are present can the read reach Q and E. The regular-rational row Q returns the inspected ledger and bound through finite readback windows, and the real seal E packages the same route for $\mathsf{Real}^{\uparrow}$ consumers.

The neighboring bounded-variation and rectifiable-curve reads are projections of the same displayed packet. They use the partition-variation and partition-length coordinates already named above, without adding a path quotient, a selected representative, a supremum over all partitions, or a completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVariationPath}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteVariationPath}^{\uparrow}$ packet is generated from a metric and continuous-path source row, a finite interval row, a displayed partition ledger, DyadicRatCore edge-distance rows, a variation-sum bound, common-refinement comparison, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteVariationPath}^{\uparrow}}$, seed)

Scope closed: Definition 6406.1, Definition 6406.2, and Definition 6406.3 seed the finite path-variation packet over displayed metric-path source rows, finite partition ledgers, dyadic edge-distance sums, regular rational readback, and a real-name seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteVariationPath}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host path space, total variation as an all-partitions supremum, quotient equality of paths, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-path source exposure, finite partition exposure, dyadic edge-distance exactness, variation-bound comparison, common-refinement comparison, RegSeqRat readback, RealUp seal routing, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FiniteVariationPathUp.

Chapter 6407

A Concrete Naming Certificate for $\mathsf{BrzozowskiDerivative}^{\uparrow}$

Following Brzozowski's regular-expression derivatives and the finite-automaton presentation of Hopcroft–Ullman, the $\mathsf{BrzozowskiDerivative}^{\uparrow}$ packet records the finite derivative closure of a regular language as BEDC NameCert data. It consumes the accepted-word surface of chapter 351, the deterministic presentation rows of chapter 2364, the finite word-spine rows of chapter 36, the right-congruence classifier of chapter 5729, and the minimal deterministic consumer of chapter 6188. The packet names the derivative table and nullability replay before any consumer may import a host language quotient, regular-expression completeness theorem, nondeterministic branch selector, or minimization canonicality claim.

Definition 6407.1 (Brzozowski derivative carrier)**.** *A* $\mathsf{BrzozowskiDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Sigma, W, L, R, \partial, \nu, C, Q, \delta, q_0, F, \mu, H, K, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row.*
2. W *is the finite word-spine row read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation row supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. R *is the displayed regular-expression or regular-description row whose derivative reads are exposed only through finite syntax names.*
5. ∂ *is the derivative table, assigning to each displayed derivative state and alphabet symbol the next displayed derivative state.*
6. ν *is the nullability row, recording whether each displayed derivative accepts the empty word.*
7. C *is the closure ledger showing that repeated derivatives of* R *by displayed words land back in the finite derivative table.*
8. Q *is the finite derivative-state carrier admitted by the closure ledger.*
9. δ *is the deterministic transition row induced by* ∂*.*
10. q_0 *is the starting derivative state named by the original description* R*.*
11. F *is the accepting-state row determined by* ν*.*

12. *μ is the handoff row to the* $\mathsf{MyhillNerode}^{\uparrow}$ *and* $\mathsf{MinimalDfa}^{\uparrow}$ *consumers through the displayed derivative classes.*
13. *H records componentwise* hsame *transport for alphabet, word-spine, accepted-language observation, syntax, derivative table, nullability, closure, state, transition, start, accepting, handoff, replay, provenance, and local naming rows.*
14. *K records displayed* Cont *replay from finite word reads through derivative iteration and nullability lookup.*
15. *P is the* Pkg *provenance over* $\mathsf{BrzozowskiDerivative}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, $\mathsf{MinimalDfa}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
16. *N is the local* $\mathsf{NameCert}_{\mathsf{BrzozowskiDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, R, \partial, \nu, C, Q, \delta, q_0, F, \mu, H, K, P, N$. It has no coordinate for a host quotient of all languages, a proof that every regular language has a displayed expression, a nondeterministic branch choice, a canonical minimal automaton proof, or an external regular-expression equivalence oracle.

Theorem 6407.2 (Brzozowski derivative NameCert obligations)**.** *For every accepted* $\mathsf{BrzozowskiDerivative}^{\uparrow}$ *carrier*

$$D = (\Sigma, W, L, R, \partial, \nu, C, Q, \delta, q_0, F, \mu, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrzozowskiDerivative}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-spine exposure, accepted-language observation, displayed syntax naming, derivative-table replay, nullability lookup, finite closure coverage, derivative-state exposure, deterministic transition replay, start-state naming, accepting-state naming, Myhill–Nerode and minimal-DFA handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6407.1. The rows Σ, W, L, R expose the finite alphabet, finite word spine, accepted-word observation, and displayed syntax source before any derivative table can be consumed.

Read $\partial, \nu, C, Q, \delta, q_0, F$ next. The derivative table ∂ gives one-step symbolic evolution, ν decides empty-word acceptance at a displayed derivative, and C certifies that iterating derivatives along displayed words remains inside the finite state carrier Q. The transition row δ, start row q_0, and accepting row F are therefore the deterministic automaton presentation obtained from those rows.

The handoff row μ is bounded by this finite derivative table. It may be read by the Myhill–Nerode and minimal-DFA siblings, but it does not add a host quotient, regular-expression completeness theorem, nondeterministic branch selector, or canonical minimization proof.

The rows H, K, P, N transport, replay, source, and name the same packet. Since no further coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 6407.3 (Brzozowski derivative regular-language handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *or* $\mathsf{FiniteAutomaton}^{\uparrow}$ *consumer of an accepted* $\mathsf{BrzozowskiDerivative}^{\uparrow}$ *packet first reads the displayed alphabet row, finite word-spine row, accepted-language observation, derivative table, nullability row, closure ledger, derivative-state carrier, deterministic transition row, start row, accepting row, and componentwise replay rows. The handoff through chapter 351, chapter 2364, chapter 5729, and chapter 6188 therefore exports only the finite derivative automaton route.*

Proof. Begin with an accepted carrier $D = (\Sigma, W, L, R, \partial, \nu, C, Q, \delta, q_0, F, \mu, H, K, P, N)$. The regular-language sibling reads the accepted-word observation L only after the displayed alphabet, word spine, syntax row, derivative table, nullability row, and closure ledger have been exposed.

The finite-automaton sibling then consumes Q, δ, q_0, F as a deterministic presentation. These rows are determined by derivative iteration and nullability lookup, so the consumer obtains a finite derivative automaton rather than an arbitrary presentation.

Finally, the Myhill–Nerode and minimal-DFA siblings may read μ as a bounded classifier handoff. The route does not assert that every regular language has a chosen expression, identify external language quotients, choose nondeterministic branches, or prove minimization canonicality. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrzozowskiDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{BrzozowskiDerivative}^{\uparrow}$ packet is generated from finite alphabet, word-spine, accepted-language observation, displayed syntax, derivative table, nullability, closure, derivative-state, deterministic transition, start, accepting, handoff, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrzozowskiDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6407.1, Definition 6407.2, and Definition 6407.3 seed the finite derivative-automaton route over RegularLanguageUp, FiniteAutomatonUp, ListUp, MyhillNerodeUp, and MinimalDfaUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrzozowskiDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host language quotients, regular-expression completeness, nondeterministic branch choice, minimization canonicality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-table replay, nullability exactness, closure coverage, deterministic automaton handoff, Myhill–Nerode handoff, minimal-DFA handoff, transport, replay, provenance, and local naming for BEDC.Derived.BrzozowskiDerivativeUp.

Chapter 6408

A Concrete Naming Certificate for $\mathsf{FreedmanInequality}^{\uparrow}$

Following Freedman's martingale inequality, the $\mathsf{FreedmanInequality}^{\uparrow}$ packet names the finite bounded-window concentration surface where martingale differences, bounded stopping decisions, and predictable variance budgets are read as BEDC rows before any tail estimate is exposed. It sits after the probability, random-variable, martingale, stopping-time, optional-stopping, and upcrossing packets of chapter 184, chapter 185, chapter 190, chapter 344, chapter 383, and chapter 1629. The chapter records the finite ledger boundary consumed by concentration estimates; it does not import completed filtrations, almost-sure quotient equality, exponential supermartingale limits, unbounded stopping choices, or ambient measure completion.

Definition 6408.1 (Freedman inequality carrier)**.** *A* $\mathsf{FreedmanInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (\Omega, X, M, T, O, U, B, V, H, C, P, N)$$

with the following rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row exposing the finite probability ledger.*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *martingale-difference family row, with every endpoint read routed through the random-variable preimage and total-readback surface.*
3. M *is the* $\mathsf{Martingale}^{\uparrow}$ *finite-window row carrying adaptedness and zero-conditional-mean reads.*
4. T *is the* $\mathsf{StoppingTime}^{\uparrow}$ *bounded decision row used to stop the inspected martingale window.*
5. O *is the* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-read row, exposing stopped values only through bounded windows.*
6. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-control row used to keep the same finite martingale window visible to pathwise consumers.*
7. B *is the bounded-step row recording the displayed bound on each martingale difference.*
8. V *is the variance-budget row recording the predictable quadratic-variation budget inspected before a tail row is exposed.*
9. H *records componentwise* hsame *transport for the probability, random-variable, martingale, stopping, stopped-read, upcrossing, bounded-step, variance-budget, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a probability read through martingale differences, finite windows, bounded stopping, stopped reads, crossing control, bounded-step rows, variance budgets, and the local certificate.*
11. *P is the* Pkg *provenance over* $\mathsf{FreedmanInequality}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FreedmanInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $\Omega, X, M, T, O, U, B, V, H, C, P, N$. It has no coordinate for a completed sigma-algebra, host measurable functions, almost-sure quotient equality, selected exponential limits, unbounded stopping choices, countable choice, or ambient measure completion.

Theorem 6408.2 (Freedman inequality NameCert obligations)**.** *For every accepted* $\mathsf{FreedmanInequality}^{\uparrow}$ *carrier $F = (\Omega, X, M, T, O, U, B, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FreedmanInequality}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed probability, random-variable, martingale, stopping-time, optional-stopping, upcrossing, bounded-step, variance-budget, transport, replay, provenance, and local naming rows; martingale-difference exposure through X, M; bounded stopped-window exposure through T, O; crossing-control compatibility through U; variance-budget accountability through V; and ledger non-escape through H, C, P, N.*

Proof. Carrier admission is projection from Definition 6408.1. The accepted packet displays exactly the rows $\Omega, X, M, T, O, U, B, V, H, C, P, N$, so every local obligation is attached to one of those coordinates.

The martingale-difference obligations are the rows X, M. The $\mathsf{RandomVar}^{\uparrow}$ endpoint family supplies finite random-variable reads, while the $\mathsf{Martingale}^{\uparrow}$ row supplies adaptedness and conditional-expectation faces inside the inspected window.

The bounded stopped-window obligations are T, O, U. The bounded stopping row determines which finite indices may be read, the optional-stopping row exposes stopped values only through those indices, and the upcrossing row keeps the same martingale window visible to crossing-control consumers.

The concentration-specific obligations are B, V. The bounded-step row records the displayed step bound, and the variance-budget row records the predictable quadratic-variation budget before a tail estimate can be named.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed sigma-algebras, host measurable functions, almost-sure quotient equality, selected exponential limits, unbounded stopping choices, countable choice, or ambient measure completion, no NameCert obligation can export them. □

Theorem 6408.3 (Freedman inequality martingale handoff)**.** *Every downstream concentration consumer of an accepted* $\mathsf{FreedmanInequality}^{\uparrow}$ *packet receives its martingale data only after the route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{RandomVar}^{\uparrow}, \mathsf{Martingale}^{\uparrow}, \mathsf{StoppingTime}^{\uparrow}, \mathsf{OptionalStopping}^{\uparrow}, \mathsf{Upcrossing}^{\uparrow} \longmapsto \textit{bounded-step and varianc}$$

has been displayed. Thus the packet hands off a finite bounded-window concentration ledger, not an ambient exponential-supermartingale theorem, completed filtration, almost-sure quotient, selected stopping time, or unbounded convergence principle.

Proof. Begin with a concentration consumer. Projection from the carrier forces the consumer to read the probability source Ω, the random-variable endpoint row X, and the finite martingale window M before any tail-bound row is available.

Bounded stopping is the next gate. The $\mathsf{StoppingTime}^{\uparrow}$ row T names the displayed finite stopping decisions, and chapter 383 supplies the stopped-read surface O only over those bounded windows.

Crossing control remains on the same finite path. The $\mathsf{Upcrossing}^{\uparrow}$ row U records the crossing ledger consumed by martingale-window arguments, so the route cannot switch to an ambient almost-sure event or an unbounded stopping theorem.

The Freedman-specific read is then B, V. The bounded-step row and predictable variance-budget row are inspected before the concentration row is exposed, and the structural rows H, C, P, N replay and name that same finite ledger. Therefore the handoff contains no completed filtration, quotient equality, selected stopping choice, exponential-limit selector, or ambient convergence principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreedmanInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{FreedmanInequality}^{\uparrow}$ packet is generated from probability, random-variable martingale-difference, finite martingale, bounded stopping-time, optional-stopping, upcrossing, bounded-step, predictable variance-budget, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreedmanInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6408.1, Definition 6408.2, and Definition 6408.3 seed the finite bounded-window martingale concentration packet for Freedman-style tail consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreedmanInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed filtrations, almost-sure quotient equality, exponential supermartingale limit selection, unbounded stopping choices, measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, martingale-difference exposure, bounded stopped-window exposure, upcrossing compatibility, predictable variance-budget accountability, and non-escape for BEDC.Derived.FreedmanIne

Chapter 6409

A Concrete Naming Certificate for $\mathsf{CompactUniformCoverRead}^{\uparrow}$

Following Bishop-style compact uniform continuity, the $\mathsf{CompactUniformCoverRead}^{\uparrow}$ packet names the finite cover-read route that turns pointwise modulus rows on a compact metric source into a tolerance-only uniform-modulus handoff. It consumes compact, metric, continuous, rational, ProbeBundle, and uniform-modulus surfaces from chapter 60, chapter 57, chapter 59, chapter 37, and chapter 369. The chapter records finite cover data and rational lower-bound readback; it does not choose an external finite subcover, invoke open-cover compactness, select a global modulus by choice, or import ambient metric completion.

Definition 6409.1 (Compact-uniform cover-read carrier)**.** *A* $\mathsf{CompactUniformCoverRead}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Cpt, M, F, Q, B, U, H, R, P, N)$$

with the following rows:

1. *Cpt is the* $\mathsf{Compact}^{\uparrow}$ *finite-cover row exposing a located finite net.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to read source and target triangle estimates.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row attached to the displayed map or graph.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius and lower-bound readback row.*
5. *B is the* ProbeBundle *finite cover row whose probes index the compact net centers and local modulus radii.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row receiving the tolerance-only precision ledger after the finite fold.*
7. *H records componentwise* hsame *transport for the compact, metric, continuous, rational, ProbeBundle, uniform-modulus, replay, provenance, and naming rows.*
8. *R records displayed* Cont *replay from a compact finite cover through pointwise moduli, rational lower-bound fold, metric triangle reads, and the uniform-modulus consumer row.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCoverRead}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCoverRead}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Cpt, M, F, Q, B, U, H, R, P, N$. It has no coordinate for an external subcover selector, open-cover compactness, quotient point equality, countable choice, or ambient metric completion.

Theorem 6409.2 (Compact-uniform cover-read NameCert obligations)**.** *For every accepted* CompactUniformCover *carrier $K = (Cpt, M, F, Q, B, U, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCoverRead}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-cover row, metric distance row, continuous pointwise-modulus row, rational radius and lower-bound row, ProbeBundle finite-cover row, UniformModulus consumer row, transport, replay, provenance, and local naming row; cover-read compatibility through Cpt, B; pointwise-modulus exposure through F; rational-fold accountability through Q; metric-triangle compatibility through M; and non-escape through H, R, P, N.*

Proof. Carrier admission is projection from Definition 6409.1. The accepted packet displays exactly $Cpt, M, F, Q, B, U, H, R, P, N$, so the local NameCert cannot refer to data outside those rows.

The cover-read obligations are Cpt, B. The compact row supplies the located finite net, while the ProbeBundle row indexes the centers and local radius rows that are actually folded.

The modulus obligations are F, Q, U. The continuous row exposes pointwise moduli at bundled centers, the rational row supplies positive-radius comparisons and the finite lower-bound fold, and the UniformModulus row receives only the resulting tolerance-only precision ledger.

Metric compatibility is read through M. Source and target triangle estimates are finite metric rows attached to the same cover-read route, not ambient metric-space theorems.

Finally H, R, P, N preserve, replay, source, and name the same finite packet. Since the carrier lacks an external subcover selector, open-cover compactness, quotient point equality, countable choice, and ambient metric completion, none of those objects is exported. □

Theorem 6409.3 (Compact finite cover to uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformCoverRead}^{\uparrow}$ *packet receives data only after the compact finite-net row, pointwise-modulus rows, rational lower-bound fold, ProbeBundle cover index, metric triangle rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row have all been displayed. The handoff is a finite ProbeBundle read, not a selected open-cover subcover, external modulus function, quotient point equality, countable choice principle, or ambient compactness theorem.*

Proof. Start with the compact finite-cover row of Definition 6409.1. The row Cpt exposes located finite-net data, and B records the ProbeBundle index through which bundled centers and radius rows are read.

At each bundled center, the continuous row F supplies a pointwise-modulus read. The rational row Q then folds the finite family of positive radii into a displayed lower-bound ledger, so the precision handed to the consumer is a BEDC history rather than an external selector.

Use the metric row M for the source and target triangle estimates. These estimates are finite distance reads attached to the bundled center and lower-bound ledger, and they are the only route by which the pointwise data become a uniform estimate.

Finally the structural rows H, R, P, N transport, replay, source, and name the same finite cover-read. Thus the $\mathsf{UniformModulus}^{\uparrow}$ row U receives only the compact finite-net, pointwise-modulus, rational-fold, ProbeBundle, metric, and structural data already displayed, with no open-cover selector, quotient point equality, countable choice, or ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCoverRead}^{\uparrow}}$.

Constructive story: A CompactUniformCoverRead$^{\uparrow}$ packet is generated from compact finite-cover rows, metric distance rows, continuous pointwise-modulus rows, rational radius and lower-bound rows, ProbeBundle finite indices, UniformModulus consumer rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformCoverRead}^{\uparrow}}$, seed)

Scope closed: Definition 6409.1, Definition 6409.2, and Definition 6409.3 seed the finite cover-read route from compact pointwise moduli to UniformModulus consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformCoverRead}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected subcovers, open-cover compactness, quotient point equality, countable choice, ambient metric completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-read compatibility, pointwise-modulus exposure, rational-fold accountability, metric-triangle compatibility, and non-escape for BEDC.Derived.CompactUniformCoverReadUp.

Chapter 6410

A Concrete Naming Certificate for $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ packet records the finite bisection surface by which a continuous real-valued map on a located interval exposes an intermediate-value bracket without invoking a host intermediate-value theorem. It sits after the terminal real-name seal of chapter 38, the located-real comparison surface of **??**, the continuous-map route of chapter 130, and the bisection method packet of chapter 2108. The packet gives L10 Real and Completion readers a finite sign-window route, not a selected root, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6410.1 (Constructive intermediate-value carrier)**.** *A* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, F, S, B, H, C, P, N)$$

with the following rows:

1. *I is the located interval row, recording displayed endpoint names, endpoint apartness, rational or dyadic enclosure windows, and the source interval orientation;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording the finite modulus, graph, and value-readback data for the displayed map on the interval row I;*
3. *S is the sign-apartness window, recording endpoint sign evidence and the retained crossing condition available at the current finite bisection depth;*
4. *B is the bisection schedule row, recording the requested depth, midpoint reads, retained subintervals, and branch evidence consumed through the* $\mathsf{BisectionMethod}^{\uparrow}$ *route;*
5. *H records componentwise* hsame *transport for the interval, map, sign window, bisection schedule, replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay from a finite intermediate-value request through continuous-map readback, sign-apartness inspection, bisection refinement, and terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure;*
7. *P is the* Pkg *provenance over* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BisectionMethod}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveIntermediateValue}^{\uparrow}}$ *row binding the paper-visible name to the finite packet.*

The carrier has no coordinate for a host intermediate-value theorem, a choice-selected root, trichotomy on arbitrary reals, quotient real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6410.2 (Constructive intermediate-value NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ *carrier*

$$V = (I, F, S, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveIntermediateValue}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row* I*, continuous-map row* F*, sign-apartness window* S*, bisection schedule* B*, componentwise transport* H*, replay route* C*, provenance row* P*, and local naming row* N*.*

Proof. Carrier admission is projection from Definition 6410.1. The accepted packet exposes only the displayed interval, map, sign, bisection, transport, replay, provenance, and naming rows.

The analytic rows are I, F, S, B. The located interval supplies the endpoint windows, the continuous-map row supplies finite value readbacks, the sign row records the crossing evidence, and the bisection row records retained branches at finite depth.

The structural rows are H, C, P, N. They transport the same packet, replay the finite request, record provenance, and bind the local name. Since no other coordinate exists, every NameCert obligation is one of these rows. □

Theorem 6410.3 (Constructive intermediate-value bisection route)**.** *Every accepted* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ *consumer that reads an intermediate-value bracket must factor through the finite route*

$$I \longrightarrow F \longrightarrow S \longrightarrow B \longrightarrow P \longrightarrow N$$

before any $\mathsf{Real}^{\uparrow}$*-facing seal is inspected. The route may expose retained intervals, midpoint values, sign windows, and regular real-name readbacks, but it cannot supply a selected root, a host existence theorem, trichotomy, quotient real equality, countable choice, or ambient completeness.*

Proof. First read the located interval row I. It fixes the endpoint windows and the interval orientation before the map can be evaluated.

Next read the continuous-map row F and sign-apartness row S. The map values are available only through the displayed finite graph and modulus data, and the crossing evidence is recorded as endpoint and retained-window sign data rather than as a host order case split.

Now apply the bisection schedule row B. Each retained branch is a finite refinement of the displayed interval and sign rows, so the consumer receives a bracketed route, not a choice-selected root.

Finally pass through P and N. Provenance and local naming attach the route to the packet boundary, while the excluded root-selection, trichotomy, quotient, choice, and completeness principles have no row to occupy. □

Theorem 6410.4 (Constructive intermediate-value L10 scope)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ *packet may use it only as a finite-window bisection certificate over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{BisectionMethod}^{\uparrow}$ *rows. The scope closes the local route from displayed interval endpoints and continuous-map readbacks to retained bisection windows; it does not close the host intermediate-value theorem, trichotomy, selected roots, countable choice, quotient real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6410.2. They require the interval, map, sign, bisection, transport, replay, provenance, and local naming rows to be visible before a consumer can read the packet.

Apply the bisection route of Definition 6410.3. It orders every intermediate-value read through I, F, S, B, P, N and keeps the Real-facing seal downstream of finite retained-window evidence.

Thus the L10 consumer receives exactly the displayed finite-window certificate. Any claim requiring host IVT, trichotomy, a selected root, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness lies outside the packet scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$ packet is generated from located interval endpoints, a continuous-map row, finite sign-apartness windows, a bisection schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6410.1, Definition 6410.2, Definition 6410.3, and Definition 6410.4 seed the finite-window intermediate-value certificate over located interval endpoints, continuous-map readbacks, sign-apartness windows, retained bisection branches, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close the host intermediate-value theorem, trichotomy, selected roots, countable choice, quotient real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, sign-window stability, bisection-route exactness, continuous-map replay, provenance, and Real-seal nonescape for BEDC.Derived.ConstructiveIntermediateValueUp.

Chapter 6411

A Concrete Naming Certificate for $\mathsf{AsymptoticDensity}^{\uparrow}$

Following Hardy and Wright on asymptotic enumeration in elementary number theory, the $\mathsf{AsymptoticDensity}^{\uparrow}$ packet records natural density as a finite BEDC NameCert surface. It sits after the unary counting discipline of chapter 28, the cofinal-tail filter route of chapter 1119, the finite averaging surface of chapter 1765, and the convergence-window sibling of chapter 1827. The packet names finite initial-segment counts, denominator rows, dyadic quotient budgets, tail-window comparison, regular rational readback, and terminal Real sealing; it does not import a host subset of natural numbers, quotient equality of predicates, a selected limiting subsequence, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6411.1 (Asymptotic density carrier)**.** *An* $\mathsf{AsymptoticDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, I, C, D, Q, T, R, E, H, K, P, N)$$

with the following displayed rows:

1. *U is the unary membership-observation row over $\mathsf{Nat}^{\uparrow}$, recording only the finite yes/no cells opened by the current density request.*
2. *I is the finite initial-segment row naming the inspected prefix lengths and their successor-denominator rows.*
3. *C is the counting ledger, storing for each displayed prefix the unary count of accepted membership cells inside that prefix.*
4. *D is the denominator ledger, storing the positive prefix lengths used by the displayed quotient reads.*
5. *Q is the $\mathsf{DyadicRatCore}^{\uparrow}$ quotient-budget row for the displayed ratios C/D, including division tolerances, slack cells, and readback budgets.*
6. *T is the $\mathsf{FiniteTailFilter}^{\uparrow}$ tail-window row that records which later prefixes are compared for density stability.*
7. *R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback row for the displayed prefix-density sequence.*
8. *E is the $\mathsf{Real}^{\uparrow}$ seal row reached only after U, I, C, D, Q, T, R have been displayed.*

9. *H records componentwise* hsame *transport for membership observations, prefix rows, counts, denominators, quotient budgets, tail windows, readback, seal, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay from membership-window reads through counting, quotient formation, tail comparison, regular readback, and Real sealing.*
11. *P is the* Pkg *provenance over* $\mathsf{AsymptoticDensity}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticDensity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, I, C, D, Q, T, R, E, H, K, P, N$. It has no coordinate for a host set of naturals, quotient predicate equality, a completed limit object, selected subsequence, trichotomy, countable choice, or ambient completeness theorem.

Theorem 6411.2 (Asymptotic density NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticDensity}^{\uparrow}$ *carrier*

$$A = (U, I, C, D, Q, T, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AsymptoticDensity}^{\uparrow}}$ *surface consists exactly of finite membership-window admission, prefix-row admission, counting-ledger exposure, positive-denominator exposure, dyadic quotient-budget exactness, finite-tail density comparison, regular rational readback, Real seal factorization, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6411.1. The rows $U, I, C, D, Q, T, R, E, H, K, P, N$ are the whole public surface, so carrier admission is exactly admission of those rows as one finite packet.

The counting part is forced by the prefix rows. The membership observations in U are read only for the prefixes named in I, and the ledger C stores exactly the corresponding finite counts. The denominator row D names the same prefix lengths, so the quotient row Q can form only the displayed prefix-density ratios.

The tail comparison is then read through T. This row imports the finite-tail discipline of chapter 1119 without adding a second selector for later prefixes or a completed limit coordinate.

Finally R and E are downstream of the displayed quotient and tail rows. The structural rows H, K, P, N transport, replay, source, and name the same route, so none of the excluded host objects can appear as an additional NameCert obligation. □

Theorem 6411.3 (Asymptotic density finite-prefix stability)**.** *Let*

$$A = (U, I, C, D, Q, T, R, E, H, K, P, N)$$

be an accepted $\mathsf{AsymptoticDensity}^{\uparrow}$ *carrier. Replacing finitely many initial membership observations in U preserves the same density-facing classifier whenever the displayed tail row T, quotient-budget row Q, regular readback row R, and Real seal row E are unchanged. The stability is a finite-tail comparison inside the packet, not an appeal to host set equality modulo finite difference or to an ambient limit theorem.*

Proof. Read the affected prefix rows first. A finite change in U can alter only the counting ledger entries C attached to the finitely many prefixes of I that inspect those cells.

Move to the tail row T. Once the displayed finite-tail filter has passed beyond those prefixes, later density comparisons use the same membership windows, quotient budgets, and prefix denominators as before.

The regular readback R is downstream of Q and T, and the Real seal E is downstream of R. If those rows are unchanged, every density-facing consumer receives the same displayed route after the finite initial segment has been skipped.

The conclusion is therefore internal to the finite carrier. No host equality modulo finite sets, selected limiting subsequence, quotient predicate equality, trichotomy, countable-choice row, or ambient completeness theorem is introduced. □

Falsifiable prediction: Every $\mathsf{AsymptoticDensity}^{\uparrow}$ density-facing read exposes finite Nat membership observations, initial-segment rows, counting ledgers, denominator rows, DyadicRatCore quotient budgets, finite-tail comparison, RegSeqRat readback, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert before it exports a density classifier.

Independence witness: nat, finitailfilter, hardycesaromean, cesaroconvergence, regseqrat, dyadicratcore, real: $\mathsf{AsymptoticDensity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite membership observations, prefix counts, denominator rows, quotient budgets, tail comparison, regular readback, Real sealing, transport, replay, provenance, and local naming as one natural-density certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticDensity}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticDensity}^{\uparrow}$ packet is generated from finite Nat membership observations, prefix rows, counting ledgers, denominator rows, dyadic quotient budgets, finite-tail comparison, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6411.1, Definition 6411.2, and Definition 6411.3 seed the finite natural-density route from Nat prefix observations through quotient budgets, tail comparison, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticDensity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets of naturals, quotient predicate equality, completed limits, selected subsequences, trichotomy, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked membership-window admission, prefix counting, denominator admission, quotient-budget exactness, tail comparison, finite-prefix stability, readback, Real seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.AsymptoticDensityUp.

Chapter 6412

A Concrete Naming Certificate for $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$

The $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ packet realizes the MetaCIC decidability audit map as finite local-checker data. It sits after the lambda-calculus syntax carrier of chapter 200, the computability boundary of chapter 199, the first-order coding surface of chapter 197, and the MetaCIC audit routes named in chapter 9379. The packet records accepted checker schedules, rejection ledgers, bounded normal-comparison endpoints, strong-normalisation handoff rows, confluence handoff rows, transport, replay, provenance, and local naming; it does not claim a global closed-term conversion decider, an unrestricted normalisation theorem, a confluence theorem, or a host proof search oracle.

Definition 6412.1 (Candidate decidability trace carrier)**.** *A* $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Gamma, t, A, S, J, B, F, R, N, C, H, K, P, L)$$

with the following displayed rows:

1. Γ *is the finite local context row used by the MetaCIC checker request.*
2. t *is the displayed candidate term row.*
3. A *is the displayed candidate type row.*
4. S *is the finite checker schedule, listing the syntax nodes, local context extensions, same-term reads, and bounded-conversion calls opened by the request.*
5. J *is the accepted-judgement row produced by the local checker when the schedule succeeds.*
6. B *is the bounded normal-comparison row, storing fuel, endpoint names, and equality-readback cells for every conversion subcall.*
7. F *is the refusal ledger for schedule branches that fail before a decidability read is exported.*
8. R *is the strong-normalisation handoff row, naming the endpoint obligations that a later SN packet must close.*
9. N *is the confluence handoff row, naming the join obligations that a later confluence packet must close.*
10. C *is the critical-path row binding the local checker schedule, rejection ledger, SN handoff, and confluence handoff into one audit trace.*

11. *H records componentwise* hsame *transport for context, term, type, schedule, judgement, bounded comparison, refusal, handoff, critical-path, replay, provenance, and local naming rows.*
12. *K records displayed* Cont *replay from local syntax reads through the checker schedule, rejection ledger, bounded comparison, and handoff rows.*
13. *P is the* Pkg *provenance over* $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
14. *L is the local* $\mathsf{NameCert}_{\mathsf{CandidateDecidabilityTrace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $\Gamma, t, A, S, J, B, F, R, N, C, H, K, P, L$. It has no coordinate for unrestricted decidability, global closed beta-conversion, an unbounded normalizer, an external confluence proof, a proof-search oracle, Lean verification, or a standard bridge.

Theorem 6412.2 (Candidate decidability trace NameCert obligations)**.** *For every accepted* CandidateDecidabilityTr *carrier*

$$T = (\Gamma, t, A, S, J, B, F, R, N, C, H, K, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CandidateDecidabilityTrace}^{\uparrow}}$ *surface consists exactly of context admission, candidate term admission, candidate type admission, finite checker-schedule exposure, accepted-judgement exposure, bounded normal-comparison exposure, rejection-ledger exposure, strong-normalisation handoff, confluence handoff, critical-path binding, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6412.1. The local context, term, type, schedule, judgement, bounded comparison, rejection, handoff, critical-path, transport, replay, provenance, and naming rows are the only coordinates of the packet.

The checker schedule S is read before any decidability-facing result. It lists the local syntax inspections, context extensions, same-term reads, and bounded-conversion calls that the accepted trace exposes.

The bounded comparison and rejection rows are then read together. The row B records the finite fuel and endpoint data for successful bounded normal comparisons, while F records the finite reasons why failed branches do not export a decidability read.

The rows R, N, C make the open boundary explicit. They point to the strong-normalisation and confluence obligations needed by later packets, but they do not close those obligations inside this carrier.

Finally, H, K, P, L transport, replay, source, and name the same finite trace. Since no further coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 6412.3 (Candidate decidability trace checker-schedule precedence)**.** *Every accepted* $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ *read exposes the finite checker schedule S before it exposes any accepted judgement J, bounded comparison row B, rejection ledger F, strong-normalisation handoff R, confluence handoff N, or critical-path row C. A consumer may therefore inspect a decidability candidate only through the displayed local-checker schedule and its finite ledger rows.*

Proof. Begin with the carrier projection. The rows Γ, t, A name the local request, and S is the first row that records how the request is checked.

The judgement row J and bounded comparison row B depend on that schedule. Without the syntax-node order, context extensions, same-term reads, and conversion subcalls listed in S, neither row has a displayed local-checker route.

The refusal ledger F is also downstream of S. It records which schedule branches fail before any decidability read is exported, so it cannot be detached from the schedule that produced those branches.

The handoff rows R, N, C bind the same finite trace to later strong-normalisation and confluence work. Thus every accepted consumer first reads S, and no unrestricted decidability theorem or proof-search oracle is introduced. □

Falsifiable prediction: Every accepted $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ trace exposes a finite checker schedule before any decidability read, bounded normal-comparison export, rejection-ledger export, strong-normalisation handoff, confluence handoff, or critical-path row is consumed.

Independence witness: metaciccriticalpath, lambdacalc, computable, firstorder: $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local context, candidate term, candidate type, checker schedule, accepted judgement, bounded normal-comparison, rejection ledger, strong-normalisation handoff, confluence handoff, critical-path binding, transport, replay, provenance, and local naming as one finite decidability-audit trace.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateDecidabilityTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{CandidateDecidabilityTrace}^{\uparrow}$ packet is generated from local context rows, candidate term rows, candidate type rows, finite checker schedules, accepted judgement rows, bounded normal-comparison rows, rejection ledgers, strong-normalisation handoff rows, confluence handoff rows, critical-path rows, componentwise hsame transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateDecidabilityTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6412.1, Definition 6412.2, and Definition 6412.3 seed the finite MetaCIC decidability-audit trace over local checker schedules, rejection ledgers, bounded comparison, SN handoff, and confluence handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateDecidabilityTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted decidability, global closed beta-conversion, strong normalisation, confluence, host proof search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, checker-schedule precedence, bounded-comparison exposure, rejection-ledger exposure, SN handoff, confluence handoff, critical-path binding, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.CandidateDecidabilityTraceUp.

Chapter 6413

A Concrete Naming Certificate for $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ packet records the finite modulus data by which an equicontinuous family over a compact metric source is read as a uniform finite-window handoff. It sits beside the shared-radius surface of chapter 4076, the finite-net handoff of chapter 2362, the compact-uniform-continuity route of chapter 388, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the constructive compact-metric neighbour of the L10 real and completion route; it does not import open-cover compactness, selected subsequences, host function-space compactness, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6413.1 (Bishop equicontinuity modulus carrier)**.** *A* $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, N, U, W, R, S, H, C, P, L)$$

with the following displayed rows:

1. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *shared-radius row, read over the same compact metric source and finite family ledger.*
2. *N is the* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *handoff row, recording the compact finite-net centers and the family-radius comparisons available at the inspected scale.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row consumed after the finite-net and shared-radius data are displayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for source and target observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and tolerance histories.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after E, N, U, W, R have fixed the finite modulus route.*
7. *H records componentwise* hsame *transport for the shared-radius, finite-net, compact-uniform, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the finite route*

$$E, N, U \longmapsto W, R, S, H, P, L.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, N, U, W, R, S, H, C, P, L$. It has no coordinate for open-cover compactness, selected subsequences, host function-space compactness, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6413.2 (Bishop equicontinuity modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ *carrier*

$$M = (E, N, U, W, R, S, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityModulus}^{\uparrow}}$ *surface consists exactly of shared-radius admission through E, finite-net equicontinuity admission through N, compact-uniform-continuity admission through U, finite-window exposure through W, regular rational readback through R, terminal Real sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 6413.1. The analytic rows are E, N, U, W, R, S, and the structural rows are H, C, P, L.

The equicontinuity and finite-net clauses are finite. The row E supplies the shared family-radius data, while N supplies the compact finite-net handoff on the same source route.

The compact-uniform and L10 clauses are then read through U, W, R, S. The finite windows and regular rational readback determine the Real-facing seal without adding a quotient real, a selected limit, or an ambient completeness principle.

Finally H, C, P, L transport, replay, source, and name exactly the displayed rows. Since none of the excluded host objects is a carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 6413.3 (Bishop equicontinuity modulus finite-net handoff)**.** *Every* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *or L10 real-facing consumer of an accepted* $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ *carrier must first read the shared route*

$$\mathsf{Equicontinuity}^{\uparrow}, \mathsf{FiniteNetEquicontinuity}^{\uparrow}, \mathsf{CompactUniformContinuity}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The handoff consists of the shared-radius row E, finite-net row N, compact-uniform row U, window row W, readback row R, seal row S, and the structural rows H, C, P, L.

Proof. Start with the obligation theorem. It identifies $E, N, U, W, R, S, H, C, P, L$ as the complete public surface of the carrier.

The finite-net stage is downstream of the shared-radius row. A consumer reads E and then N before the compact-uniform row U is public, so the route cannot be replaced by a selected finite net outside the packet.

The L10-facing stage is downstream of that compact-uniform handoff. Rows W and R expose finite windows and regular rational readback before S seals the Real-facing output.

The rows H, C, P, L preserve this order by componentwise transport, replay, provenance, and local naming. Therefore every compact-uniform or real-facing consumer receives the displayed finite-net handoff rather than open-cover compactness, host function-space compactness, selected subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6413.4 (Bishop equicontinuity modulus nonescape)**.** *No accepted* $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ *packet exports open-cover compactness, selected subsequences, host function-space compactness, quotient equality of real names, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its public modulus content is exactly the finite row package*

$$E, N, U, W, R, S, H, C, P, L.$$

Proof. Apply Definition 6413.2. Every public read factors through the displayed tuple $E, N, U, W, R, S, H, C, P, L$.

The shared-radius input is confined to E, the finite-net input to N, the compact-uniform handoff to U, and the real-facing route to W, R, S. The structural rows only transport, replay, package, and name these displayed rows.

None of the excluded principles is present as a carrier coordinate or as a structural row. Exporting any of them would leave the local $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopEquicontinuityModulus}^{\uparrow}$ packet is generated from an EquicontinuityUp shared-radius row, a FiniteNetEquicontinuityUp finite-net handoff, a CompactUniformContinuityUp consumer row, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6413.1, Definition 6413.2, Definition 6413.3, and Definition 6413.4 seed the finite modulus route over equicontinuity, finite-net, compact-uniform, window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BishopEquicontinuityModulusUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subsequences, host function-space compactness, quotient equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-radius admission, finite-net handoff, compact-uniform handoff, finite-window replay, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopEquicontinuityModulusUp.

Chapter 6414

A Concrete Naming Certificate for $\mathsf{FaberSchauderSystem}^{\uparrow}$

Following Schauder's basis construction and constructive approximation analysis, the $\mathsf{FaberSchauderSystem}^{\uparrow}$ packet records the dyadic hat functions that generate finite piecewise-linear approximation rows. It sits over the dyadic rational core of chapter 291, the real-name seal of chapter 38, the uniform-modulus route of chapter 369, and the sequence-level basis surface of chapter 3669. The packet stores dyadic levels, midpoint hats, finite support intervals, coefficient readbacks, uniform-error budgets, transport, replay, provenance, and naming, but it does not claim convergence of an arbitrary infinite series or a host Banach-space basis theorem.

Definition 6414.1 (Faber–Schauder system carrier)**.** *A* $\mathsf{FaberSchauderSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, H, S, C, U, R, T, P, N)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *level row, recording the displayed dyadic grid, parent interval endpoints, and midpoint name used by a hat function;*
2. *H is the hat-profile row, recording the piecewise-linear rise and fall from the two endpoints through the midpoint;*
3. *S is the finite support row, recording the closed dyadic interval on which the hat can be nonzero and the exterior zero reads;*
4. *C is the coefficient row, recording the displayed scalar readback assigned to the hat;*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, recording the finite tolerance budget for a partial approximation request;*
6. *R records componentwise* hsame *transport for the dyadic level, hat profile, support, coefficient, modulus, replay, provenance, and naming rows;*
7. *T records displayed* Cont *replay from a finite approximation request through dyadic-grid inspection, hat-support readback, coefficient multiplication, and uniform-modulus budget inspection;*
8. *P is the* Pkg *provenance over* $\mathsf{FaberSchauderSystem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FaberSchauderSystem}^{\uparrow}}$ *row binding the paper-visible name to the finite packet.*

The carrier has no coordinate for quotient real equality, selected infinite coefficients, trichotomy, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or convergence of an infinite expansion.*

Lean4 (statement) `BEDC.Derived.FaberSchauderSystemUp`

Theorem 6414.2 (Faber–Schauder system NameCert obligations)**.** *For every accepted* $\mathsf{FaberSchauderSystem}^{\uparrow}$ *carrier*

$$F = (D, H, S, C, U, R, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FaberSchauderSystem}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the dyadic level, hat profile, support, coefficient, modulus, transport, replay, provenance, and local naming rows;*
2. *dyadic-grid exactness, requiring* D *to expose the parent endpoints and midpoint through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before the hat row is read;*
3. *hat-support locality, requiring* S *to name the finite dyadic support and exterior zero reads before any coefficient or real-facing approximation row is consumed;*
4. *uniform-budget compatibility, requiring* U *to provide the finite tolerance row used by partial sums before the packet reaches* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{SchauderBasis}^{\uparrow}$ *consumers;*
5. *non-escape, requiring* R, T, P, N *to preserve the same displayed rows under transport, replay, provenance, and naming, with no quotient real equality, selected infinite coefficient stream, trichotomy, choice, or ambient completeness coordinate.*

Proof. Carrier admission is projection from Definition 6414.1. The accepted packet exposes only the dyadic level, hat profile, support, coefficient, uniform budget, transport, replay, provenance, and naming rows.

The analytic rows are D, H, S, C, U. The dyadic row fixes the grid and midpoint, the hat row records the piecewise-linear profile, the support row confines the nonzero region, the coefficient row records the displayed scalar, and the modulus row supplies the finite approximation budget.

The structural rows are R, T, P, N. They transport the same packet, replay the finite request, record provenance, and bind the local name. Since no other coordinate exists, the excluded quotient, selection, trichotomy, choice, completeness, and infinite-expansion objects cannot enter the NameCert surface. □

Theorem 6414.3 (Faber–Schauder dyadic hat support)**.** *Every accepted* $\mathsf{FaberSchauderSystem}^{\uparrow}$ *carrier* $F = (D, H, S, C, U, R, T, P, N)$ *determines a finite dyadic hat-support route*

$$D \longrightarrow H \longrightarrow S \longrightarrow C \longrightarrow U$$

before any $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{SchauderBasis}^{\uparrow}$ *consumer reads the packet. The support row* S *confines the hat to the displayed dyadic parent interval, records exterior zero reads, and keeps the coefficient and uniform-modulus rows inside the same finite* BHist *route.*

Proof. Start with the dyadic row D. It names the parent endpoints and midpoint through $\mathsf{DyadicRatCore}^{\uparrow}$, so the hat row has a finite grid before any real-facing read occurs.

Next read H and S. The hat profile rises and falls only across the displayed dyadic interval, and the support row records the exterior zero reads that prevent the packet from exporting a host support predicate.

Finally pass through C and U. The coefficient and uniform budget are attached to the same displayed support, while R, T, P, N preserve that route under transport, replay, provenance, and

naming. Thus a Real or Schauder-basis consumer receives a finite hat-support certificate and no selected infinite coefficient stream, quotient equality, trichotomy, choice, or ambient completeness principle. □

Theorem 6414.4 (Faber–Schauder Haar-step route)**.** *Every accepted* $\mathsf{FaberSchauderSystem}^{\uparrow}$ *consumer reading a finite dyadic hat coefficient must pass through the sibling route*

$$\mathsf{DyadicTentFunction}^{\uparrow} \longrightarrow \mathsf{StepFunction}^{\uparrow} \longrightarrow \mathsf{HaarWavelet}^{\uparrow} \longrightarrow \mathsf{SchauderBasis}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The $\mathsf{DyadicTentFunction}^{\uparrow}$ *primitive is the one-hat support row supplied by chapter 8351; finite sampled values are read through the step-function and Haar-wavelet surfaces of chapter 3780 and chapter 3881; and the coordinate handoff reaches the Schauder-basis surface of chapter 3669 before any regular-rational or real seal is consumed.*

Proof. Project an accepted carrier as $F = (D, H, S, C, U, R, T, P, N)$. The rows D, H, S determine the displayed dyadic parent interval, midpoint hat profile, and exterior zero reads, while C records the finite coefficient and U records the finite uniform budget.

The tent primitive is therefore not an ambient function object. It is the one-hat support row obtained by reading D, H, S as a $\mathsf{DyadicTentFunction}^{\uparrow}$ packet, with the coefficient row C still attached to the same finite BHist route.

Finite sampled values next enter $\mathsf{StepFunction}^{\uparrow}$ as displayed intervalwise reads. The Haar surface consumes only those finite step reads, so the route reaches $\mathsf{HaarWavelet}^{\uparrow}$ without asserting Haar measure, quotient function equality, or a completed function space.

Finally the structural rows R, T, P, N preserve the same displayed packet through transport, replay, provenance, and naming. The $\mathsf{SchauderBasis}^{\uparrow}$ consumer receives a coordinate handoff, and the subsequent $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads are terminal seals for finite approximation data, not claims of infinite expansion convergence, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FaberSchauderSystem}^{\uparrow}}$.

Constructive story: A $\mathsf{FaberSchauderSystem}^{\uparrow}$ packet is generated from dyadic grid rows, finite hat profiles, support intervals, coefficient readbacks, uniform-modulus budgets, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FaberSchauderSystem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6414.1, Definition 6414.2, and Definition 6414.3 seed the finite dyadic hat system certificate over dyadic levels, support intervals, coefficient rows, uniform budgets, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FaberSchauderSystem}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: The chapter does not close convergence of arbitrary infinite Faber–Schauder expansions, uniqueness of coefficients, quotient real equality, trichotomy, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-grid exactness, hat-support locality, uniform-budget compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FaberSchauderSystemUp.

Chapter 6415

A Concrete Naming Certificate for DyadicBisectionTree$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the DyadicBisectionTree$^{\uparrow}$ packet records the finite tree surface behind repeated dyadic bisection of a located interval. It sits over the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the endpoint refinement route of chapter 6199. The packet is a finite carrier for L10 Real and Completion work: every displayed node is a bounded dyadic interval row with a parent link, a tolerance budget, and a retained-child certificate, while no infinite branch or completeness principle is selected.

Definition 6415.1 (Dyadic bisection tree carrier)**.** *A* DyadicBisectionTree$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, I, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *ledger of endpoints, midpoints, radii, and common dyadic refinements.*
2. *I is the* LocatedIntervalRefinement$^{\uparrow}$ *source row, giving the initial located interval and the finite refinement witnesses used by each retained node.*
3. *T is the finite bisection-tree row: its nodes are addressed by finite binary words, every non-root node records its parent, and every retained child carries the half-interval selected by the displayed midpoint comparison.*
4. *W is the* StreamName$^{\uparrow}$ *finite-window row read by a consumer that asks for a node at a requested depth or tolerance.*
5. *R is the* RegSeqRat$^{\uparrow}$ *regular-Cauchy readback row obtained from nested retained intervals whose dyadic diameters are bounded by the displayed depth budget.*
6. *E is the* Real$^{\uparrow}$ *seal boundary, reachable only after D, I, T, W, R have been projected.*
7. *H records componentwise* hsame *transport for the dyadic, located-interval, tree, window, regular-readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for endpoint admission, midpoint bisection, retained-child transport, finite-window inspection, regular readback, and terminal sealing.*
9. *P is the* Pkg *provenance over* DyadicBisectionTree$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, LocatedIntervalRefinement$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicBisectionTree}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, I, T, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, a selected infinite branch, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6415.2 (Nested regular-Cauchy readback from a dyadic bisection tree)**.** *For every accepted* $\mathsf{DyadicBisectionTree}^{\uparrow}$ *carrier $K = (D, I, T, W, R, E, H, C, P, N)$ and every displayed retained branch prefix in T, each unary depth request factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{LocatedIntervalRefinement}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The resulting $\mathsf{RegSeqRat}^{\uparrow}$ *row is read from nested retained dyadic intervals with diameter bounded by the depth budget. The route does not assert real completeness, open-cover compactness, countable choice, a selected infinite branch, quotient equality, or any ambient equality of* $\mathsf{Real}^{\uparrow}$ *seals.*

Proof. Start with the accepted carrier and project its dyadic and interval rows. The row D supplies the endpoint arithmetic, midpoint, radius, and common-refinement data, while I supplies the located refinement witness for the current retained interval.

Read the branch prefix inside T. Because node addresses are finite binary words, the requested prefix exposes only finitely many parent links and child selections. Each retained child is justified by the displayed midpoint comparison and by the corresponding row of D and I.

The finite observation row W records the window requested at the chosen depth. The diameter bound is computed from the bisection depth in T and the dyadic radius ledger in D, so the handoff to R is a regular-Cauchy readback over inspected finite intervals rather than a selected limit.

Finally the seal row E is downstream of R, and the structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no row contains an infinite-branch selector, open-cover compactness proof, countable-choice schedule, quotient-equality coordinate, or ambient completeness witness, the listed route is the full exported regular-Cauchy boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBisectionTree}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBisectionTree}^{\uparrow}$ packet is generated from a DyadicRatCore endpoint and tolerance ledger, a LocatedIntervalRefinement source row, a finite binary bisection-tree row, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicBisectionTree}^{\uparrow}}$, seed)

Scope closed: Definition 6415.1 and Definition 6415.2 seed the finite dyadic bisection-tree carrier over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedIntervalRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicBisectionTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, open-cover compactness, countable choice, a selected infinite branch, quotient equality, theorem verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint arithmetic, located interval refinement, retained-child bisection, finite-window inspection, regular-Cauchy readback, terminal sealing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicBisectionTreeUp.

Chapter 6416

A Concrete Naming Certificate for $\mathsf{KalmanDecomposition}^{\uparrow}$

Following Kalman, the $\mathsf{KalmanDecomposition}^{\uparrow}$ packet records the finite block-splitting rows that separate a linear control system into controllable/observable, controllable/unobservable, uncontrollable/observable, and residual components. It sits after the finite reachability surface of chapter 297, the observation surface of chapter 294, the matrix-ledger rows of chapter 49, the vector-space carrier rows of chapter 47, and the regulator-facing rows of chapter 296. The packet names only finite plant, basis, block, transition, transport, replay, provenance, and local naming data; it does not claim a canonical host quotient, analytic minimal realization, transfer-function equality, or stochastic filtering law.

Definition 6416.1 (KalmanDecomposition carrier)**.** *A* $\mathsf{KalmanDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, U, Y, A, B, C, S, Q, O, Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}, M, H, T, P, N)$$

with the following displayed rows:

1. *X is the finite state-space row, U is the input row, and Y is the output row, each read through displayed* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *coordinates.*
2. *$A : X \to X$, $B : U \to X$, and $C : X \to Y$ are the finite transition, input, and output rows of the plant.*
3. *S is the displayed change-of-basis ledger whose columns are finite state rows and whose inverse-readback is recorded through* $\mathsf{Mat}^{\uparrow}$*.*
4. *Q is the controllability row consumed from* $\mathsf{ControlControllability}^{\uparrow}$*, and O is the observability row consumed from* $\mathsf{ControlObservability}^{\uparrow}$*.*
5. *Z_{co} is the controllable/observable block, Z_{cu} is the controllable/unobservable block, Z_{uo} is the uncontrollable/observable block, and Z_{ur} is the residual block.*
6. *M is the matrix transition ledger recording the displayed block reads of $S^{-1}AS$, $S^{-1}B$, and CS together with the zero-pattern entries that witness the decomposition route.*
7. *H records componentwise* hsame *transport for state, input, output, plant, basis, controllability, observability, block, matrix-transition, replay, provenance, and naming rows.*
8. *T records displayed* Cont *replay for the finite route*

$$X, U, Y, A, B, C, S, Q, O \longmapsto Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}, M, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{KalmanDecomposition}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KalmanDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, U, Y, A, B, C, S, Q, O, Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}, M, H, T, P, N$. It has no coordinate for a host quotient state space, a canonical ambient basis, analytic transfer-function equality, stochastic covariance data, or an infinite-dimensional realization theorem.

Theorem 6416.2 (KalmanDecomposition NameCert obligations)**.** *For every accepted* $\mathsf{KalmanDecomposition}^{\uparrow}$ *carrier*

$$K = (X, U, Y, A, B, C, S, Q, O, Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}, M, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KalmanDecomposition}^{\uparrow}}$ *surface consists exactly of finite state, input, and output admission through X, U, Y, plant admission through A, B, C, basis-ledger admission through S, controllability admission through Q, observability admission through O, block admission through $Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}$, matrix transition readback through M, componentwise* hsame *transport through H, displayed* Cont *replay through T, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6416.1. The rows X, U, Y, A, B, C, S supply the finite plant and basis data, and the rows Q, O attach the displayed controllability and observability certificates.

The four block rows $Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}$ are read only after Q, O and S have exposed the finite splitting ledger. The row M then records the block transition, input, and output matrices over that same basis route rather than over an ambient quotient or selected canonical form.

Finally H, T, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no coordinate for host quotient semantics, analytic transfer equality, stochastic covariance, or an infinite-dimensional realization theorem, the listed rows exhaust the local certificate surface. □

Theorem 6416.3 (KalmanDecomposition controllable-observable split)**.** *Every downstream control consumer that reads an accepted* $\mathsf{KalmanDecomposition}^{\uparrow}$ *packet as a controllable/observable splitting must pass first through the displayed finite route*

$$\mathsf{VecSpace}^{\uparrow}, \mathsf{Mat}^{\uparrow}, \mathsf{ControlControllability}^{\uparrow}, \mathsf{ControlObservability}^{\uparrow} \longmapsto Z_{co}, Z_{cu}, Z_{uo}, Z_{ur}, M.$$

The exported splitting consists only of the finite state, input, output, plant, basis, controllability, observability, block, transition-ledger, transport, replay, provenance, and local NameCert *rows of Definition 6416.1.*

Proof. Start with Definition 6416.2. It exposes the plant rows and the basis ledger before either the controllability or observability rows can be used.

The controllable side is read through Q and chapter 297, while the observable side is read through O and chapter 294. Both rows are finite control certificates over the same $\mathsf{Mat}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ surface, so the consumer cannot replace them by an unlogged reachable quotient or unobservable quotient.

The four block rows and the matrix transition ledger then provide the only public splitting data. The structural rows H, T, P, N preserve that finite route under transport, replay, provenance, and local naming, so the consumer receives the displayed Kalman decomposition packet rather than a host canonical-form theorem. □

Falsifiable prediction: An accepted $\mathsf{KalmanDecomposition}^{\uparrow}$ packet is refuted by any control consumer that reads a controllable/observable block, residual block, or block transition ledger

while bypassing the displayed basis row S, the controllability row Q, the observability row O, or the matrix transition ledger M.

Independence witness: controlcontrollability, controlobservability, matrix, vecspace, lqr: none is bijective to $\mathsf{KalmanDecomposition}^{\uparrow}$ because the decomposition packet jointly carries plant, basis, controllability, observability, four block rows, matrix transition ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{KalmanDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{KalmanDecomposition}^{\uparrow}$ packet is generated from finite state, input, output, plant, basis, controllability, observability, four block, matrix-transition, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KalmanDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6416.1, Definition 6416.2, Definition 6416.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite control decomposition NameCert surface over controllability, observability, matrix, vector-space, and regulator-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KalmanDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, canonical host quotient semantics, analytic minimal realization, transfer-function equality, stochastic covariance laws, or infinite-dimensional realization theory.

Upgrade path: Obligation closure requires checked carrier admission, basis-ledger admission, controllability and observability admission, block admission, matrix-transition exactness, transport, replay, provenance, and local naming for BEDC.Derived.KalmanDecompositionUp.

Chapter 6417

A Concrete Naming Certificate for $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ packet records a finite nested-interval locator route from dyadic refinement data to stream, regular-rational, and real-seal consumers. It reads the retained-window discipline of chapter 320, the dyadic arithmetic surface of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a local L10 locator object for nested dyadic intervals; it is not a host interval-intersection theorem, a selected point, quotient stream equality, countable choice, or ambient real completeness.

Definition 6417.1 (BishopIntervalNestedLocator carrier)**.** *A* $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, J, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the retained* $\mathsf{NestedInterval}^{\uparrow}$ *row naming the inspected interval chain;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger, including dyadic endpoints, dyadic radii, and the finite containment budget used by the locator;*
3. *J is the nested-refinement row asserting that each inspected interval window is retained inside the previous displayed window;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row obtained from the dyadic endpoint windows and retained containment data;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row determined by the same dyadic windows and not by an independent rational sequence;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S and R have been projected;*
7. *H records componentwise* hsame *transport for I, D, J, S, R, E and the support rows;*
8. *C records displayed* Cont *replay from a locator request through the nested interval, dyadic refinement, stream-window, regular readback, and real-seal rows;*
9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestedLocator}^{\uparrow}}$ *row.*

The classifier compares two carriers only through these rows and contains no coordinate for a selected limit point, quotient stream equality, host real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6417.2 (BishopIntervalNestedLocator NameCert obligations)**.** *For an accepted* BishopIntervalNestedLoca *carrier* $L = (I, D, J, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestedLocator}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed nested interval, dyadic refinement ledger, containment row, stream row, regular rational readback, real seal, transports, replay routes, provenance, and local naming row; dyadic endpoint and radius exactness through* D*; retained-window nesting through* J*;* $\mathsf{StreamName}^{\uparrow}$ *readback through* S*;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*;* $\mathsf{Real}^{\uparrow}$ *seal delivery through* E*; componentwise transport through* H*; displayed continuation through* C*; package provenance through* P*; and local naming through* N*. No obligation exports host interval intersection, selected limits, quotient stream equality, host real equality, countable choice, open-cover compactness, or real completeness.*

Proof. Project the carrier of Definition 6417.1. The packet displays exactly the nested interval row, dyadic refinement ledger, containment row, stream row, regular rational row, real seal, and structural rows.

The dyadic and containment obligations are row projections. The row D gives the endpoint and radius budget, and J states the retained-window relation between adjacent displayed intervals. Thus a locator-facing read must inspect the finite dyadic refinement data before any stream or real-facing row is available.

The stream, regular, and seal obligations are then forced by the carrier order. The rows S and R are read from the same dyadic windows, and E is downstream of both. Transport, replay, provenance, and local naming preserve that order, leaving no coordinate for the excluded host-level principles. □

Theorem 6417.3 (BishopIntervalNestedLocator dyadic refinement)**.** *Every accepted* $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ *packet factors each locator precision request through the dyadic refinement route*

$$I, \quad D, \quad J, \quad S, \quad R.$$

The row D *supplies the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius data,* J *supplies retained nested containment,* S *supplies the* $\mathsf{StreamName}^{\uparrow}$ *windows, and* R *supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. The route cannot be replaced by a selected branch, quotient stream equality, host real equality, or countable choice.*

Proof. Start with carrier admission. A precision request reaches the packet only through the displayed rows $I, D, J, S, R, E, H, C, P, N$.

Apply the obligation surface of Definition 6417.2. Dyadic endpoint and radius exactness in D precede retained-window nesting in J, and both precede stream and regular-rational readback.

Finally use the absence clauses in Definition 6417.1. Since the carrier has no selected branch, quotient stream equality, host real equality, or countable-choice coordinate, every accepted precision request is exhausted by the displayed dyadic refinement route. □

Theorem 6417.4 (BishopIntervalNestedLocator Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ *carrier must pass through the retained* $\mathsf{NestedInterval}^{\uparrow}$ *row* I*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the containment row* J*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the seal row* E *is visible. The seal is therefore a terminal row of the displayed locator packet, not a proof of real completeness or host equality of real numbers.*

Proof. Use Definition 6417.3 to identify the only route into a locator precision request. It places I, D, J, S, R before any real-facing consumer can read the packet.

Now project the seal row in Definition 6417.1. The row E occurs after the stream and regular-rational rows, and the support rows H, C, P, N only transport, replay, package, and name the same route.

Thus the seal consumer receives a finite terminal row rather than an ambient completeness theorem, host real equality, selected point, or quotient stream equality. □

Theorem 6417.5 (BishopIntervalNestedLocator non-escape)**.** *Every consumer of an accepted* $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ *packet is exhausted by the finite rows* $I, D, J, S, R, E, H, C, P, N$*. In particular, the locator route can feed* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *sibling consumers, but it cannot escape as a selected nested-interval intersection point, a quotient stream representative, a host real equality proof, a countable-choice functional, an open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the packet inventory from Definition 6417.1. The only data rows are I, D, J, S, R, E, and the only structural rows are H, C, P, N.

Use the NameCert obligations of Definition 6417.2. They force every accepted read through the displayed dyadic refinement, nested containment, stream-window, regular-rational, seal, transport, replay, provenance, and naming route.

Apply the Real-seal route of Definition 6417.4 for completion-facing consumers. Since the seal itself remains downstream of I, D, J, S, R and no excluded coordinate is present in the carrier, the locator cannot export any of the host-level objects listed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNestedLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestedLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6417.1, Definition 6417.2, Definition 6417.3, and Definition 6417.4 seed the finite nested-interval locator route from dyadic refinement data to StreamName, RegSeqRat, and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestedLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim interval-intersection existence, selected limits, quotient stream equality, host real equality, countable choice, open-cover compactness, real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove carrier admission, dyadic endpoint and radius exactness, retained-window nesting, stream readback, regular rational readback, real-seal delivery, transport, replay, provenance, and local naming for the same displayed packet.

Constructive story: A $\mathsf{BishopIntervalNestedLocator}^{\uparrow}$ packet is generated from a retained nested-interval row, a DyadicRatCore refinement ledger, a containment row, a StreamName row, a RegSeqRat readback row, a RealUp seal, componentwise transports, replay routes, provenance, and a local NameCert row.

Chapter 6418

A Concrete Naming Certificate for $\mathsf{MorseLemma}^{\uparrow}$

Following Milnor's Morse theory treatment and the Bishop–Bridges constructive analysis of finite coordinate data, the $\mathsf{MorseLemma}^{\uparrow}$ packet records the local quadratic normal-form route for a nondegenerate critical point as finite BEDC certificate data. It sits between the Morse-theory surface of chapter 3533, the boundary packet of chapter 6040, the manifold chart rows of chapter 96, the derivative ledger of chapter 122, the implicit-function graph route of chapter 1834, the terminal real-name seal of chapter 38, and the vector-space coordinate carrier of chapter 47. The packet names only displayed chart, critical-point, Hessian-signature, local-coordinate, quadratic normal-form, transport, replay, provenance, and local naming rows; it does not import a host smooth-manifold theorem, a selected chart atlas, quotient germs, or analytic compactness.

Definition 6418.1 (Morse lemma NameCert carrier)**.** *A* $\mathsf{MorseLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, C, Q, S, L, NF, H, R, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *chart-source row, naming the displayed neighbourhood, chart window, coordinate dimension, and local domain in which the critical point is inspected.*
2. *C is the critical-point row, recording the point, the value row, and the derivative-vanishing witness exposed through the* $\mathsf{Derivative}^{\uparrow}$ *ledger.*
3. *Q is the Hessian-signature row, storing the finite symmetric bilinear form, its nondegeneracy witness, and the positive and negative coordinate counts read through* $\mathsf{VecSpace}^{\uparrow}$*.*
4. *S is the local-coordinate row, recording the coordinate transformation routed through the implicit-function packet and the finite neighbourhood on which it is admitted.*
5. *L is the quadratic normal-form row, displaying the equation that reads the function as the critical value plus the signed square sum determined by Q after S has been admitted.*
6. *NF is the normal-form transport row, recording how the chart, critical-point, Hessian, coordinate, and quadratic rows are transported componentwise under* hsame*.*
7. *H records displayed* Cont *replay from a normal-form query through X, C, Q, S, L, NF, P, A.*
8. *R is the Real readback row, exposing the scalar values only after the chart, derivative, Hessian, coordinate, and quadratic rows have been displayed.*

9. *P is the* Pkg *provenance over* $\mathsf{MorseLemma}^{\uparrow}$, $\mathsf{MorseTheory}^{\uparrow}$, $\mathsf{MorseBoundary}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{ImplicitFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *A is the local* $\mathsf{NameCert}_{\mathsf{MorseLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, C, Q, S, L, NF, H, R, P, A$. It has no coordinate for a host smooth atlas, quotient germ equality, selected coordinate theorem, compactness principle, or ambient real manifold semantics.

Theorem 6418.2 (Morse lemma critical-chart ledger)**.** *For every accepted* $\mathsf{MorseLemma}^{\uparrow}$ *carrier $M = (X, C, Q, S, L, NF, H, R, P, A)$, the critical-chart ledger exposes the chart-source row X, the derivative-vanishing critical-point row C, and the Hessian-signature row Q before any local-coordinate row S or quadratic normal-form row L may be read. A consumer that omits one of X, C, Q receives no accepted Morse-lemma packet.*

Proof. Project the carrier of Definition 6418.1. The chart-source, critical-point, and Hessian-signature data are separate rows, and no later coordinate stores a replacement for them.

The coordinate row S is downstream of these rows. It records a local transformation only after the chart window, derivative-vanishing witness, and nondegenerate Hessian signature have been displayed.

The quadratic row L is downstream again. Since L reads the signed square decomposition determined by Q inside the chart named by X at the critical point named by C, the packet cannot admit a normal-form read with any of those rows missing. □

Theorem 6418.3 (Morse lemma Hessian normal-form obligation)**.** *For an accepted* $\mathsf{MorseLemma}^{\uparrow}$ *carrier, every quadratic normal-form read factors through*

$$\mathsf{Manifold}^{\uparrow},\ \mathsf{Derivative}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow} \longmapsto X, C, Q \longmapsto S, L \longmapsto NF, H, R, P, A.$$

The Hessian-signature row determines the positive and negative square counts in L, while NF, H, R, P, A transport, replay, source, read back, and name the same finite route without adding a host Morse-lemma theorem.

Proof. Begin with the displayed analytic rows. The manifold row supplies the local chart window, the derivative row supplies the critical-point witness, and the vector-space row supplies the finite coordinate carrier for the Hessian signature.

Next read S and L. The coordinate transformation S is admitted only inside the displayed neighbourhood, and the normal-form row L records the signed quadratic expression using the signature data in Q.

Finally route the read through NF, H, R, P, A. These rows preserve and replay the same finite packet, expose scalar values only through the Real readback row, and bind the local NameCert. Because the carrier has no host-atlas, quotient-germ, or selected-theorem coordinate, the result is a BEDC normal-form obligation rather than an imported smooth-manifold theorem. □

Falsifiable prediction: Every accepted $\mathsf{MorseLemma}^{\uparrow}$ normal-form read exposes chart, critical-point, Hessian-signature, local-coordinate, quadratic normal-form, componentwise transport, displayed replay, Real readback, provenance, and local NameCert rows before any Morse-theory consumer reads the local quadratic model.

Independence witness: morse-theory, morse-boundary, manifold, derivative, implicitfunction, real, vecspace: $\mathsf{MorseLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a

nondegenerate critical-point chart, Hessian signature, local coordinate transformation, quadratic normal-form row, transport, replay, Real readback, provenance, and local naming as one finite packet.

The Lean follow-up namespace is expected at `BEDC.Derived.MorseLemmaUp`. It must preserve the chart, critical-point, Hessian-signature, local-coordinate, quadratic normal-form, transport, replay, Real readback, provenance, and local NameCert rows before any checked marker may cite the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MorseLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{MorseLemma}^{\uparrow}$ packet is generated from manifold chart rows, critical-point derivative rows, Hessian-signature rows, local coordinate rows, quadratic normal-form rows, componentwise transport, displayed replay, Real readback, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MorseLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6418.1, Definition 6418.2, and Definition 6418.3 seed the finite local quadratic normal-form route consumed by Morse-theory packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MorseLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host smooth-manifold Morse theory, selected coordinate atlases, quotient germs, analytic compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-chart ledger exactness, Hessian-signature normal-form coverage, transport stability, replay stability, Real readback, provenance exhaustion, and local naming for BEDC.Derived.MorseLemmaUp.

Chapter 6419

A Concrete Naming Certificate for $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$

The $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ packet realizes the residual-substitution and confluence frontier of section 9386.3 as finite NameCert data. It sits beside the critical-path packet of chapter 765, the confluence audit packet of chapter 735, the decidability boundary of chapter 749, the normalization-confluence bridge of chapter 1190, and the closed-term substitution boundary of chapter 527. The packet names a typed window, substitution route, residual frontier, bounded local-diamond checker, obstruction retention, transport, replay, provenance, and local naming surface before any confluence read is admitted.

Falsifiable prediction: Every accepted $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ packet exposes typed-window, substitution-route, residual-frontier, bounded local-diamond checker, obstruction-retention, transport, replay, provenance, and local NameCert rows before any confluence read.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, metacicdecidable-boundary, metacicnormalizationconfluencebridge, closedtermsubstitutioncompiler: no listed sibling is bijective to $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ because the substitution-diamond packet jointly carries typed-window data, substitution routing, residual-frontier rows, a bounded local-diamond checker, obstruction retention, transport, replay, provenance, and local naming as one finite confluence boundary.

Definition 6419.1 (MetaCIC substitution diamond carrier)**.** *A* $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, S, R, B, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, recording the source term, typing context, two substitution-sensitive reduct branches, binder depth, and candidate common endpoint under inspection.*
2. *S is the substitution-route row, exposing the closed substitution or compiler-facing substitution boundary used by each branch before residual comparison begins.*
3. *R is the residual-frontier row, listing the beta positions, residual descendants, variable-escape checks, and branch-local substitution dependencies that remain visible.*
4. *B is the bounded local-diamond checker row, recording the finite checks that either produce a local common endpoint or stop at a named obstruction.*

5. *O is the obstruction-retention row, naming unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, and kernel consistency as nonexported endpoints.*
6. *H records componentwise* hsame *transport for typed windows, substitution routes, residual frontiers, bounded checker rows, obstruction rows, replay routes, provenance, and local naming.*
7. *C records displayed* Cont *replay from a confluence query through T, S, R, B, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, MetaCICConfluenceAuditPac MetaCICDecidableBoundary$^{\uparrow}$, $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, R, B, O, H, C, P, N$. It has no coordinate for global confluence, unrestricted substitution composition, subject reduction, evaluator soundness, quotient syntax equality, or kernel consistency.

Theorem 6419.2 (MetaCIC substitution diamond confluence obligation)**.** *For every accepted* $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ *carrier $D = (T, S, R, B, O, H, C, P, N)$, a confluence consumer may read a local diamond only through the route*

$$T, S, R \longmapsto B \longmapsto O, H, C, P, N.$$

The checker row B may expose a bounded common endpoint for the displayed typed window, but the obstruction row O remains part of the packet and prevents the read from becoming unrestricted substitution compatibility or global closed-term confluence.

Proof. Read the typed-window and substitution rows first. The source term, branch data, binder depth, and substitution route are visible in T and S before any residual comparison is made.

Next inspect the residual-frontier row R. It names the beta positions, residual descendants, and branch-local dependencies, so the bounded checker receives a finite frontier rather than an unbounded substitution algebra.

Finally apply the checker and obstruction rows. If B exposes a local common endpoint, O still records the endpoints not supplied by the packet. Transport, replay, provenance, and naming preserve this same finite route, so a consumer cannot cite the accepted packet as a premise-free confluence theorem. □

Lean follow-up must provide Nontrivial $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ and FieldFaithful MetaCICSubstitutionDian TasteGate instances in `BEDC.Derived.MetaCICSubstitutionDiamondUp`. Those instances must preserve $T, S, R, B, O, H, C, P, N$ as distinct displayed rows before any checked marker may cite the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}$ packet is generated from typed-window rows, substitution-route rows, residual-frontier rows, bounded local-diamond checker rows, obstruction-retention rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}}$, seed)

Scope closed: Definition 6419.1, Definition 6419.2, the falsifiable prediction, and the independence witness seed the finite substitution-sensitive local-diamond boundary for MetaCIC confluence consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamond}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, substitution-route exactness, residual-frontier coverage, bounded-checker determinacy, obstruction retention, transport stability, replay stability, provenance exhaustion, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MetaCICSubstitutionDiamondUp

Chapter 6420

A Concrete Naming Certificate for $\mathsf{AbelSummability}^{\uparrow}$

Following Abel and Hardy, the $\mathsf{AbelSummability}^{\uparrow}$ packet records the finite radial-window surface by which a displayed power-series family is read as summability data before the L10 Real and Completion route may consume an endpoint seal. It sits after the Abel summation row of chapter 1763, the radial-limit route of chapter 6058, the Cesaro-facing mean surface of chapter 1765, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet is a traditional summability carrier expressed as finite BEDC rows; it is not a quotient of power series, a selected host sum, countable choice, trichotomy on arbitrary coefficients, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6420.1 (Abel summability carrier)**.** *An* $\mathsf{AbelSummability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{AbelSummation}^{\uparrow}$ *source row carrying the finite polynomial-window and summation-by-parts reads.*
2. *T is the* $\mathsf{AbelTheorem}^{\uparrow}$ *radial-boundary row that binds the same Abel data to a displayed endpoint route.*
3. *M is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *or Cesaro-facing mean row used as the traditional summability comparison surface.*
4. *D is the dyadic radial-parameter and tolerance ledger for the inspected radius window.*
5. *W records the finite coefficient, partial-sum, and radial windows inspected by S, T, M, D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same radial windows and tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after S, T, M, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the Abel, theorem, mean, tolerance, window, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from each radial precision request through S, T, M, D, W to R and E.*

10. *P is the* Pkg *provenance over* $\mathsf{AbelSummability}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTheorem}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelSummability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, M, D, W, R, E, H, C, P, N$. It has no coordinate for host power-series semantics, a selected infinite sum, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AbelSummabilityUp.AbelSummabilityTasteGate single carrier alignment`

Theorem 6420.2 (Abel summability NameCert obligations)**.** *For every accepted* $\mathsf{AbelSummability}^{\uparrow}$ *carrier $A = (S, T, M, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelSummability}^{\uparrow}}$ *surface consists exactly of Abel summation admission through S; radial-boundary admission through T; Cesaro-facing mean exposure through M; dyadic radial tolerance accounting through D; finite coefficient, partial-sum, and radius-window exhaustion through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *seal ordering through E; and componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 6420.1. The tuple displays exactly the Abel summation row, radial-boundary row, Cesaro-facing mean row, dyadic tolerance ledger, finite windows, regular readback, Real seal, and structural rows.

The analytic data are finite. Rows S, T, M, D, W name the polynomial windows, radial parameters, and comparison means that the summability read may inspect, while R repacks those same rows as regular rational data.

The seal row E is downstream from R. Transport through H and replay through C preserve the same displayed route, and P, N only source and name it. Since no further coordinate exists in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6420.3 (Abel summability power-series handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{AbelSummability}^{\uparrow}$ *carrier must read the finite radial route*

$$\mathsf{AbelSummation}^{\uparrow}, \quad \mathsf{AbelTheorem}^{\uparrow}, \quad \mathsf{HardyCesaroMean}^{\uparrow}, \quad D, W, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the same packet rows $S, T, M, D, W, R, E, H, C, P, N$. Thus the power-series summability handoff is a displayed radial-window readback and not a host power-series limit, selected sum, quotient stream equality, countable-choice selector, or ambient Real-completeness principle.

Proof. Begin with an accepted carrier and project the route rows S, T, M, D, W, R, E. The rows S and T provide the Abel summation and Abel theorem faces, while M supplies the Hardy-Cesaro comparison surface over the same displayed request.

Read the finite radial window through D and W. These rows fix the tolerances, radius parameters, coefficient windows, and partial-sum windows that determine the regular rational readback R.

Finally pass through E, H, C, P, N. The seal row receives its analytic input from R, and the structural rows transport, replay, source, and name that same path. Therefore a downstream Real or Completion consumer obtains only the displayed finite handoff. □

Falsifiable prediction: An accepted $\mathsf{AbelSummability}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the Abel summation row, Abel theorem row, Hardy-Cesaro mean row, dyadic radial ledger, finite radial windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: abel-summation, abel-theorem, hardy-cesaro-mean, regseqrat, real: $\mathsf{AbelSummability}^{\uparrow}$ is not bijective to any listed sibling because it binds radial-window summability data, Cesaro-facing comparison, regular readback, and Real sealing into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelSummability}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelSummability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6420.1, Definition 6420.2, and Definition 6420.3 seed the finite Abel summability route over Abel summation rows, Abel theorem radial windows, Hardy-Cesaro mean data, regular rational readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelSummability}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.AbelSummabilityUp.AbelSummabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host power-series semantics, selected sums, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Abel summation admission, Abel theorem radial-boundary admission, Hardy-Cesaro mean exposure, dyadic radial tolerance accounting, finite-window exhaustion, regular readback, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.AbelSummabilityUp.

Constructive story: An $\mathsf{AbelSummability}^{\uparrow}$ packet is generated from Abel summation data, Abel theorem radial-boundary data, Hardy-Cesaro mean data, dyadic radial tolerances, finite coefficient and radius windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6421

A Concrete Naming Certificate for $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$

Following Brouwer, Bishop, and Bridges, the $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ packet names the finite prefix-cylinder basis used before $\mathsf{BaireMetric}^{\uparrow}$ consumers read compact-window or fan-style coverage data. It sits after the Baire metric of chapter 1602, the prefix carrier of chapter 334, the finite observation schedules of chapter 126, the dyadic radius rows of chapter 291, and the finite bar surface of chapter 400. The packet records cylinders as displayed finite prefixes with dyadic radius witnesses; it is not a completed-stream topology, quotient equality of names, countable choice, or an open-cover compactness principle.

Definition 6421.1 (Located Baire metric compactness carrier)**.** *A* $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (B, S, W, D, C, F, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet whose finite word names the cylinder stem.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that authorizes every inspected prefix window.*
3. *W is the* $\mathsf{Nat}^{\uparrow}$ *window row recording the displayed stem length and the first-disagreement boundary available to metric reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row attached to W, recording the dyadic cylinder radius read by* $\mathsf{BaireMetric}^{\uparrow}$*.*
5. *C is the* $\mathsf{BaireMetric}^{\uparrow}$ *cylinder-consumer row, requiring every metric read to pass through B, S, W, D before any parent metric or ultrametric handoff.*
6. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-window handoff row, available only as finite bar or compact-window evidence over displayed prefix cylinders.*
7. *H records componentwise* hsame *transport for the prefix packet, schedule, window, radius, metric consumer, fan handoff, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from cylinder requests through $B, S, W, D, C, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BaireMetricCylinderBasis}^{\uparrow}}$ *row.*
The carrier predicate is exactly the assertion that these rows form one finite prefix-cylinder basis packet. It has no coordinate for completed Baire streams, quotient equality of names, an arbitrary topology, countable choice, or open-cover compactness.

Theorem 6421.2 (BaireMetricCylinderBasis NameCert obligations)**.** *For every accepted* $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ *carrier $Z = (B, S, W, D, C, F, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireMetricCylinderBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for Z, prefix-packet admission through B, finite schedule admission through S, window exactness through W, dyadic radius exactness through D, BaireMetric cylinder handoff through C, FanTheorem compact-window handoff through F, componentwise* hsame *stability through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N. No obligation is stored outside these finite rows.*

Proof. Project the carrier of Definition 6421.1. The analytic rows are B, S, W, D, C, F, and the structural rows are H, T, P, N.

The prefix row B and schedule row S are inspected before any radius data is available. The window row W then fixes the finite stem length and first-disagreement boundary, and the dyadic row D records the corresponding radius.

The metric row C and fan row F consume the same displayed prefix-cylinder data. Since H, T, P, N only transport, replay, source, and name the listed rows, the NameCert surface has no remaining coordinate for completed streams, quotient equality, countable choice, or open-cover compactness. □

Theorem 6421.3 (BaireMetricCylinderBasis prefix cover)**.** *Let $Z = (B, S, W, D, C, F, H, T, P, N)$ be an accepted* $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ *carrier. Every cylinder cover read exported by Z is covered by the finite prefix route*

$$B, S, W, D \longmapsto C \longmapsto F \longmapsto H, T, P, N.$$

Thus a BaireMetric consumer sees a located prefix stem, a finite StreamName schedule, a Nat first-disagreement window, and a DyadicRatCore radius row before any compact-window handoff is public.

Proof. Start with the prefix and schedule rows B, S. They are the only rows that authorize a cylinder stem and its inspected finite windows.

Read the window and radius rows W, D. These rows locate the cylinder by finite length and dyadic radius, so the metric consumer row C cannot begin from a completed stream or a quotient name.

Finally pass through F, H, T, P, N. The FanTheorem row receives the displayed finite cylinder route, while transport, replay, provenance, and local naming preserve that route without adding an arbitrary topology or compactness oracle. □

Theorem 6421.4 (BaireMetricCylinderBasis compact-window handoff)**.** *For an accepted* $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ *carrier $Z = (B, S, W, D, C, F, H, T, P, N)$, every compact-window consumer that reads the* $\mathsf{FanTheorem}^{\uparrow}$ *row F must first display the same* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, and* $\mathsf{BaireMetric}^{\uparrow}$ *cylinder row used by the prefix cover theorem. The handoff is therefore a finite cylinder-basis surface for compact-window consumers, not a completed-stream topology, quotient equality principle, selected infinite branch, or open-cover compactness theorem.*

Proof. Use Definition 6421.3. It identifies every exported cover read with the displayed route $B, S, W, D, C, H, T, P, N$.

The fan row F is downstream of that route. It may read finite bar or compact-window evidence over the displayed cylinders, but it receives no separate branch selector or open-cover compactness coordinate.

The structural rows H, T, P, N preserve the same finite data under componentwise transport, continuation replay, provenance, and local naming. Hence compact-window consumers receive exactly the prefix-cylinder basis surface named in the statement. □

Falsifiable prediction: A $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ consumer that exports a compact-window or BaireMetric cylinder cover without reading the prefix packet B, StreamName schedule S, Nat window W, DyadicRatCore radius D, and cylinder row C refutes the carrier surface.

Independence witness: bairemetric, bairespace, streamname, dyadicratcore, fantheorem: BaireMetricCylinderBas is not bijective to any listed sibling because it binds prefix stems, finite schedules, first-disagreement windows, dyadic radii, metric cylinder reads, fan compact-window handoff, transport, replay, provenance, and local naming as one finite basis packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireMetricCylinderBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireMetricCylinderBasis}^{\uparrow}$ packet is generated from a BaireSpace prefix packet, StreamName schedule, Nat window, DyadicRatCore radius row, BaireMetric cylinder-consumer row, FanTheorem compact-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireMetricCylinderBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6421.1, Definition 6421.2, Definition 6421.3, and Definition 6421.4 seed the finite prefix-cylinder basis surface for BaireMetric and compact-window consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireMetricCylinderBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed-stream topology, quotient equality of Baire names, selected infinite branches, countable choice, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, schedule exactness, dyadic radius exactness, metric cylinder handoff, compact-window handoff, transport, replay, provenance, and local naming for BEDC.Derived.BaireMetricCylinderBasis

Chapter 6422

A Concrete Naming Certificate for BishopCompletionApproximationTheorem$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopCompletionApproximationTheorem$^{\uparrow}$ packet records the finite theorem-facing surface by which a displayed Bishop approximation route is allowed to serve completion-facing Real reads. It sits after the approximation source of chapter 3510, the criterion row of chapter 3404, the completion theorem packet of chapter 3484, the regular rational source of chapter 301, and the terminal seal of chapter 38. The packet is a finite NameCert theorem certificate; it is not a quotient completion, not a selected limit, not countable choice, not trichotomy, and not ambient Real$^{\uparrow}$ completeness.

Definition 6422.1 (Bishop completion approximation theorem carrier)**.** *A* BishopCompletionApproximationTheorem *carrier is a finite* BHist *packet*

$$T = (A, B, M, W, R, K, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* BishopCauchyApproximation$^{\uparrow}$ *approximation-service row.*
2. *B is the* BishopCompletionCriterion$^{\uparrow}$ *admissibility row that permits the displayed approximation route to enter a completion boundary.*
3. *M is the* CauchyConvergenceModulus$^{\uparrow}$ *schedule read by the criterion and approximation service.*
4. *W is the finite* StreamName$^{\uparrow}$ *window opened at the displayed precision request.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback extracted from the same window and tolerance schedule.*
6. *K is the* BishopCompletionTheorem$^{\uparrow}$ *handoff row reached only after A, B, M, W, R have been displayed.*
7. *E is the terminal* Real$^{\uparrow}$ *seal for Real-facing completion reads.*
8. *H records componentwise* hsame *transport for the approximation, criterion, modulus, window, readback, completion handoff, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion-facing approximation request through A, B, M, W, R, K, E.*
10. *P is the* Pkg *provenance over* BishopCompletionApproximationTheorem$^{\uparrow}$, BishopCauchyApproximation$^{\uparrow}$, BishopCompletionCriterion$^{\uparrow}$, BishopCompletionTheorem$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through these rows and their displayed replay route. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6422.2 (Bishop completion approximation carrier certificate)**.** *For every accepted* BishopCompletionAppr *carrier $T = (A, B, M, W, R, K, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed approximation service, completion criterion, modulus schedule, finite stream window, regular rational readback, completion theorem handoff, Real seal, transport, replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 6422.1. The displayed rows are precisely $A, B, M, W, R, K, E, H, C, P, N$.

The source side is exhausted by A, M, W, R. The completion side is admitted only through B and reaches K before the terminal Real seal E is visible.

Finally H, C, P, N transport, replay, source, and name the same finite theorem packet. Since no excluded quotient, selected-limit, choice, trichotomy, host-equality, compactness, or ambient-completeness coordinate occurs in the carrier, the local NameCert surface has no other obligation row. □

Theorem 6422.3 (Bishop completion approximation density factorization)**.** *Every completion-facing dense approximation read of $T = (A, B, M, W, R, K, E, H, C, P, N)$ factors through A, then B, then the finite schedule and readback rows M, W, R, before it reaches the* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *handoff K and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The factorization exports no quotient stream, selected representative, countable-choice selector, trichotomy row, open-cover compactness witness, or ambient completeness proof.*

Proof. Begin with a completion-facing approximation request. The approximation service row A supplies only the finite Bishop Cauchy approximation route, so the request has not yet obtained a completion object or Real value.

Apply the criterion row B. It authorizes the same displayed route to enter the completion boundary only after the modulus schedule M, finite stream window W, and regular rational readback R are present.

The completion theorem handoff K is therefore downstream of A, B, M, W, R. The Real seal E is downstream of K, and H, C, P, N replay and name exactly that route. The missing quotient, selector, choice, trichotomy, compactness, and ambient-completeness rows cannot be recovered from the factorization. □

Theorem 6422.4 (Bishop completion approximation Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by a* $\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}$ *packet is sealed through E after the approximation-service row, completion criterion, modulus schedule, finite stream window, regular rational readback, and completion theorem handoff have all been displayed. A Real-facing consumer receives only that sealed route with H, C, P, N support, not host real equality, quotient completion equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier rows in the order fixed by Definition 6422.1. The rows A, B, M, W, R, K are all displayed before E can be consumed.

The support rows then keep the Real-facing export on the same route: H transports componentwise, C replays the displayed continuation, P records only the cited sibling packets, and N names the local certificate.

Since the seal row is reached only through those displayed rows, the consumer cannot replace it by host real equality, quotient completion equality, countable choice, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6422.1, Definition 6422.2, Definition 6422.3, and Definition 6422.4 seed the finite Bishop completion approximation theorem route from approximation service through criterion, modulus, stream window, regular readback, completion handoff, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host real equality, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximation-service exactness, completion-criterion compatibility, density factorization, Real-seal nonescape, transport replay, provenance, and local NameCert rows for BEDC.Derived.BishopCompletionApproximationTheoremUp.

Constructive story: A $\mathsf{BishopCompletionApproximationTheorem}^{\uparrow}$ packet is generated from a BishopCauchyApproximation service row, BishopCompletionCriterion admissibility row, CauchyConvergenceModulus schedule, finite StreamName window, RegSeqRat readback, BishopCompletionTheorem handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6423

A Concrete Naming Certificate for $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers and Troelstra–van Dalen on located cuts, the $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ packet records the finite comparison surface between located lower/upper Dedekind data and the regular-Cauchy real-name route. It sits beside the dyadic tolerance ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a traditional representation comparison as finite BEDC data; it is not a quotient of cuts, a host-real equality test, countable choice, trichotomy for arbitrary rationals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6423.1 (Located Dedekind real comparison carrier)**.** *A* $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the located lower/upper Dedekind row, carrying lower membership windows, upper membership windows, inhabited-side evidence, rounded-side evidence, and the finite locatedness comparison requested by the consumer.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-window ledger used to compare the displayed lower and upper cuts at the requested dyadic precision.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row exposing only the cut observations and dyadic windows inspected by D and Q.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same located cut data, finite observations, and dyadic tolerance windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located cut row, dyadic tolerance ledger, finite observations, and regular readback have been displayed.*
6. *H records componentwise* hsame *transport for the located cut row, dyadic ledger, stream observations, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each located comparison request through D, Q, W to R and then to E.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindRealComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, Q, W, R, E, H, C, P, N$. It has no coordinate for quotient cut equality, selected representatives, host real comparison, countable choice, arbitrary-rational trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6423.2 (Located Dedekind real comparison NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ *carrier $L = (D, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindRealCompar}}$ *surface consists exactly of located lower/upper cut admission through D; dyadic tolerance-window accounting through Q; finite observation exposure through W; regular-Cauchy readback through R; terminal Real sealing through E; and componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 6423.1. The tuple displays the located cut data, dyadic tolerance windows, finite stream observations, regular readback, Real seal, and structural rows.

The located comparison is finite. The row D provides the lower and upper cut windows actually inspected, while Q fixes the dyadic tolerance window and W records the finite observations available to the request.

The row R repacks those same displayed observations as regular rational data, and the seal row E is downstream from that readback. Transport through H and replay through C preserve the same route, while P, N only source and name it. □

Theorem 6423.3 (Located Dedekind real comparison regular-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ *carrier must read the located cut row D, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Q, and the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row W before consuming the regular readback row R. Thus the comparison from located Dedekind data to regular-Cauchy data is a displayed finite handoff, not a quotient of cuts, selected representative, host comparison, countable-choice schedule, or ambient completeness principle.*

Proof. Begin with an accepted carrier and project the rows D, Q, W, R. The only regular-Cauchy coordinate is R, and the carrier places it after the located cut data, dyadic tolerance ledger, and finite observation row.

Apply Definition 6423.2. The obligation surface requires the readback through R to use the same rows D, Q, W that admitted the located comparison.

The structural rows H, C, P, N do not introduce a second route. They transport, replay, source, and name the same handoff, so a regular-Cauchy or Completion consumer cannot bypass the displayed finite comparison. □

Theorem 6423.4 (Located Dedekind real comparison Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ *carrier reaches the terminal seal E only through the route*

$$D \longmapsto Q \longmapsto W \longmapsto R \longmapsto E.$$

The exported surface contains no quotient cut equality, selected host real, host comparison procedure, countable-choice search, trichotomy split for arbitrary rationals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Inspect the carrier tuple. The terminal Real row is E, and it appears only after the located cut row, dyadic tolerance ledger, finite observation row, and regular readback row.

Use Definition 6423.3. Any Real-facing consumer first receives the regular-Cauchy handoff from the displayed finite located comparison route.

Finally apply the nonescape part of Definition 6423.2. Since the excluded objects have no carrier coordinate and no replay route in C, the terminal Real seal cannot export them. □

Falsifiable prediction: An accepted $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading located lower/upper Dedekind data, dyadic tolerance windows, finite stream observations, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ is not bijective to any listed sibling because it binds located lower/upper cut observations, dyadic tolerance windows, finite observation exposure, regular-Cauchy readback, and Real sealing into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDedekindRealComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDedekindRealComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6423.1, Definition 6423.2, Definition 6423.3, and Definition 6423.4 seed the finite comparison route from located lower/upper Dedekind rows to regular-Cauchy readback and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDedekindRealComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cut equality, selected host reals, host comparison procedures, countable choice, arbitrary-rational trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located cut admission, dyadic tolerance accounting, finite observation exposure, regular readback, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDedekindRealComparisonUp.

Constructive story: A $\mathsf{LocatedDedekindRealComparison}^{\uparrow}$ packet is generated from located lower and upper Dedekind rows, dyadic tolerance windows, finite StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6424

A Concrete Naming Certificate for $\mathsf{CauchyTailMirror}^{\uparrow}$

Following Bishop and Bridges on tail-equivalent Cauchy sequences, the $\mathsf{CauchyTailMirror}^{\uparrow}$ packet records the finite mirror by which two displayed Cauchy tails are read through the same tolerance ledger before any $\mathsf{Real}^{\uparrow}$ consumer identifies their terminal seals. It sits on the L10 route supplied by the stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance carrier of chapter 291, the terminal real-name seal of chapter 38, and the common-tail completion surface of chapter 1639. The packet names a finite cofinal-tail comparison; it does not assert quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6424.1 (Cauchy tail mirror carrier)**.** *A* $\mathsf{CauchyTailMirror}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W_0, W_1, R_0, R_1, D, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. W_0 *and* W_1 *are two* $\mathsf{StreamName}^{\uparrow}$ *tail-window rows, recording the finite observations, tail indices, and point cells opened from the two stream names.*
2. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic precision request, comparison budget, and cofinal tail bound used by both readbacks.*
4. E_0 *and* E_1 *are the two downstream* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the shared* D *row and the displayed readbacks have been consumed.*
5. H *records componentwise* hsame *transport for the two windows, the two regular readbacks, the shared tolerance ledger, the two Real seals, replay rows, provenance rows, and local naming rows.*
6. C *records displayed* Cont *replay from a tail comparison request through* $W_0, W_1, R_0, R_1, D, E_0, E_1, H, P, N$.
7. P *is the* Pkg *provenance over* $\mathsf{CauchyTailMirror}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailMirror}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $W_0, W_1, R_0, R_1, D, E_0, E_1, H, C, P, N$. *It has no coordinate for quotient stream equality, host real equality, a selected limit, countable choice, hidden cofinal subsequences, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6424.2 (Cauchy tail mirror NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailMirror}^{\uparrow}$ *carrier*

$$M = (W_0, W_1, R_0, R_1, D, E_0, E_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailMirror}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{StreamName}^{\uparrow}$ *tail windows, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the two* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No mirror obligation is stored outside those finite rows.*

Proof. Project the accepted carrier of Definition 6424.1. The visible analytic rows are $W_0, W_1, R_0, R_1, D, E_0, E_1$, and the structural rows are H, C, P, N.

Read the tail windows first. The two $\mathsf{StreamName}^{\uparrow}$ rows fix the finite windows in which the two Cauchy tails are inspected, and the two $\mathsf{RegSeqRat}^{\uparrow}$ rows attach regular rational readbacks to exactly those windows.

The shared dyadic ledger then binds the mirror. Since both readbacks consume the same D row, the comparison has one displayed tolerance budget rather than two unrelated tail witnesses.

Finally the seal, transport, replay, provenance, and local naming rows package the same route. The carrier has no excluded coordinate, so no quotient, host-equality, selected-limit, choice, hidden-subsequence, or ambient-completeness obligation can appear outside the displayed packet. □

Theorem 6424.3 (Cauchy tail mirror shared tolerance handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchyTailMirror}^{\uparrow}$ *packet must read the finite route*

$$W_0, W_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto E_0, E_1$$

together with H, C, P, N *before comparing the terminal seals. Thus the mirror identifies cofinal Cauchy tails only through two displayed StreamName windows, two regular rational readbacks, and one shared dyadic tolerance ledger.*

Proof. Begin with a Real-facing tail comparison. Carrier admission exposes W_0 and W_1 before any regular readback or seal row can be consumed.

Attach the regular readbacks. The rows R_0 and R_1 are tied to those displayed windows, so the comparison cannot switch to an unrecorded stream representative.

Use the shared tolerance ledger D. It is the only dyadic precision row in the packet, and both readbacks must pass through it before either Real seal E_0 or E_1 is available.

The rows H, C, P, N transport, replay, source, and name the same finite order. Hence the consumer receives a finite mirror route, not quotient stream equality, host real equality, selected limits, countable choice, hidden cofinal subsequences, or ambient Real completeness. □

Theorem 6424.4 (Cauchy tail mirror componentwise determinacy)**.** *Let* M *and* M' *be accepted* $\mathsf{CauchyTailMirror}^{\uparrow}$ *packets whose two* $\mathsf{StreamName}^{\uparrow}$ *tail windows, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and structural rows agree componentwise through the displayed* hsame *transport. Then any comparison of their downstream* $\mathsf{Real}^{\uparrow}$ *seals is forced to use the same mirror route before either packet can expose a terminal Real-facing read.*

Proof. Project both carriers. Componentwise agreement on the two window rows fixes the finite observations available on each side, and agreement on the two readback rows fixes the regular rational data read from those observations.

Compare the shared dyadic ledger next. Since the ledger row is common inside each packet and matched by the displayed transport, both packets present the same precision request, budget, and cofinal tail bound to any downstream consumer.

The Real seals are downstream of those rows. Transport and replay preserve only the displayed route, while provenance and local naming add no independent comparison coordinate. Therefore the terminal reads are forced through the same mirror route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailMirror}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailMirror}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6424.1, Definition 6424.2, Definition 6424.3, and Definition 6424.4 seed the finite cofinal-tail mirror route from two StreamName windows through two RegSeqRat readbacks, one DyadicRatCore tolerance ledger, two RealUp seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailMirror}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, hidden cofinal subsequences, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared tolerance exactness, tail-window readback compatibility, Real-seal handoff, componentwise determinacy, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailMirrorUp.

Constructive story: A $\mathsf{CauchyTailMirror}^{\uparrow}$ packet is generated from two StreamName tail windows, two RegSeqRat readbacks, one shared DyadicRatCore tolerance ledger, two RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6425

A Concrete Naming Certificate for $\mathsf{LocatedCompactImage}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompactImage}^{\uparrow}$ packet records the finite-net image route for a uniformly continuous map whose source is a located compact metric packet. It sits over the compact metric source of chapter 129, the continuous-map surface of chapter 130, the metric rows of chapter 57, the finite-net coverage row of chapter 127, and the terminal real-name boundary of chapter 38. The packet names the displayed image as a finite BEDC carrier; it does not assert open-cover compactness, countable choice, host image equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6425.1 (Located compact image carrier)**.** *A* $\mathsf{LocatedCompactImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, X, Y, F, U, T, R, H, C, P, N)$$

with the following rows:

1. *K is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, read only through displayed compact-metric and located finite-net data.*
2. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose distance classifier supplies the finite-net coverage checks.*
3. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row in which image radii and target distances are inspected.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the displayed map from the source packet into the target packet.*
5. *U is the uniform-continuity or finite-modulus row attached to F on the displayed compact source.*
6. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net image ledger obtained by transporting the displayed source net through F and U.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint-radius row used only for displayed target-distance and radius readback.*
8. *H records componentwise* hsame *transport for the compact source, metric rows, map row, modulus row, finite-net image ledger, Real radius row, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from source-net coverage through the continuous-map row, target metric row, radius ledger, and image finite-net handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactImage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, X, Y, F, U, T, R, H, C, P, N$. It has no coordinate for a selected image set in the host category, open-cover compactness, countable-choice enumeration, quotient point equality, host image equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6425.2 (Located compact image NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactImage}^{\uparrow}$ *carrier $I = (K, X, Y, F, U, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactImage}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, source and target metric rows, continuous-map row, modulus row, finite-net image ledger, Real radius row, transport, replay, provenance, and local naming; source-net coverage through K, X; map-value routing through F; modulus-accountable target-radius transport through U, Y, R; finite image-net handoff through T; and non-escape through H, C, P, N.*

Proof. Project the carrier of Definition 6425.1. The compact metric source K and source metric row X supply only displayed finite-net coverage data.

The map row F and modulus row U transport the displayed source net into the target metric row Y. Target radii are read through R, so the image ledger is assembled from finite metric and real-name rows rather than from a host image object.

The finite image-net row T is therefore downstream from K, X, Y, F, U, R. Finally H, C, P, N transport, replay, source, and name the same packet, so the listed obligations exhaust the certificate surface and no external compactness or choice datum is exported. □

Theorem 6425.3 (Located compact image finite-net handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCompactImage}^{\uparrow}$ *packet receives an image finite-net ledger only through the displayed route*

$$K, X, Y, F, U, T, R, H, C, P, N.$$

The handoff transports the finite source net along the continuous-map and modulus rows, reads target distances and radii through Y and R, and then exposes the finite image-net row T. It does not claim open-cover compactness, countable choice, host image equality, quotient point equality, selected image representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source-net coverage rows K and X. They provide the finite source centers and coverage radii that can be inspected by a BEDC consumer.

Next read the continuous-map row F together with the modulus row U. These rows transport each displayed source center and the associated radius request into the target metric row Y, with endpoint-radius readback through R.

The row T then records exactly the transported target centers and radii as the image finite-net ledger. Structural rows H, C, P, N keep the handoff componentwise, replayable, sourced, and locally named. Since the carrier has no host image coordinate, external subcover selector, quotient equality, countable-choice schedule, or completeness row, the consumer receives only the finite-net handoff stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactImage}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCompactImage}^{\uparrow}$ packet is generated from a compact metric source, source and target metric rows, a continuous-map row, a uniform-modulus row, a totally bounded image ledger, Real radius readback, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6425.1, Definition 6425.2, and Definition 6425.3 seed the finite-net image route for uniformly continuous maps on located compact metric packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, countable choice, host image equality, quotient point equality, selected image representatives, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-net coverage, map-value routing, modulus-accountable target-radius transport, finite image-net handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactImageUp.

Chapter 6426

A Concrete Naming Certificate for $\mathsf{LocatedImageModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedImageModulus}^{\uparrow}$ packet isolates the finite locator-and-modulus witness used when a continuous image is read as located data in the L10 Real and Completion layer. It sits between the located compact image route of chapter 6425, the continuous-image compactness route of chapter 3604, the continuous-map graph surface of chapter 130, the compact metric source of chapter 129, and the terminal real-name seal of chapter 38. The packet records how a displayed source locator and a displayed uniform modulus feed a finite target interval-cell ledger before any downstream image-location consumer sees a located-image boundary.

Definition 6426.1 (Located image modulus carrier)**.** *A* $\mathsf{LocatedImageModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, F, U, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the source locator row, read through the compact metric and located-source surfaces that answer finite distance-to-source requests before an image-location query is admitted.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, exposing the displayed graph reads and pointwise continuity faces used to move a located source window into the target.*
3. *U is the uniform-modulus row for F on the displayed compact metric source, converting each target tolerance request into a source locator radius.*
4. *T is the target interval-cell ledger, recording the finite target cells, target radii, and containment checks produced from the source locator and the modulus row.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which finite source and target observations are inspected by an image-location consumer.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the rational approximants and tolerance comparisons for the source locator, target cells, and modulus response.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal that compares the target-cell readback with the displayed located-image boundary.*

8. *H records componentwise* hsame *transport for the source locator, continuous-map row, modulus row, target interval-cell ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a located-image request through $L, F, U, T, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedImageModulus}^{\uparrow}$, $\mathsf{LocatedCompactImage}^{\uparrow}$, $\mathsf{ContinuousImageCompact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedImageModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $L, F, U, T, S, R, E, H, C, P, N$. It has no coordinate for a host image set, selected preimage, quotient equality of target names, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6426.2 (Located image modulus NameCert obligations)**.** *For every accepted* $\mathsf{LocatedImageModulus}^{\uparrow}$ *carrier $M = (L, F, U, T, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedImageModulus}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the source locator, continuous-map row, uniform modulus, target interval-cell ledger, StreamName window, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *source-locator exposure, requiring every source-side locatedness read used by the image route to pass through L before the map row is inspected;*
3. *modulus-accountable image routing, requiring each target tolerance request to pass through U for the displayed continuous-map row F before any target cell is exposed;*
4. *finite target-cell handoff, requiring the target interval-cell ledger T to be assembled from the source locator, the modulus response, the finite StreamName window, and the RegSeqRat readback before the Real seal E is read;*
5. *non-escape, stating that host image sets, selected preimages, quotient equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are not carrier coordinates and cannot discharge local obligations.*

Proof. Carrier admission is projection from Definition 6426.1. The rows $L, F, U, T, S, R, E, H, C, P, N$ are the complete visible packet.

The source-locator row is read first. A source-side distance or locatedness request supplies only the finite information displayed by L, so no source representative is selected before the map row is consumed.

Next the continuous-map and modulus rows route the image request. The row F supplies the displayed graph read, while U turns the target tolerance into the source radius that the locator row can answer.

The target-cell handoff is finite. Rows S and R record the inspected windows and rational comparisons, T records the corresponding target interval cells, and E is the only Real-facing seal available to the consumer.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host image set, selected preimage, quotient equality, countable-choice schedule, or ambient completeness row, the obligations cannot export those objects. □

Theorem 6426.3 (Located image modulus finite-window handoff)**.** *Every image-location consumer of an accepted* $\mathsf{LocatedImageModulus}^{\uparrow}$ *packet reads through the displayed route*

$$L, F, U, T, S, R, E, H, C, P, N.$$

The consumer first receives the source locator, then the continuous-map row and uniform modulus, then the target interval-cell ledger through the finite $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and finally the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It receives no host image set, selected preimage, quotient equality of target names, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the accepted carrier of Definition 6426.1. Its source-facing data are the locator row L, the continuous-map row F, and the uniform-modulus row U.

Route a target tolerance through those rows. The modulus row converts the target request into a source locator radius, the locator row answers through displayed finite data, and the map row supplies only the target reads attached to that inspected source window.

The target handoff then passes through S, R, T, E. The StreamName row records the finite window, the RegSeqRat row records rational approximants and tolerances, the target interval-cell ledger records the cells produced by the route, and the Real seal compares the displayed readback with the image-location boundary.

Finally replay the transport and provenance rows H, C, P, N. They preserve the same finite route and have no coordinate for host image sets, selected preimages, quotient equality, choice, or ambient completeness. Thus every image-location consumer receives exactly the displayed locator/modulus/window handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedImageModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedImageModulus}^{\uparrow}$ packet is generated from a source locator, ContinuousMap row, uniform modulus, target interval-cell ledger, StreamName finite window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedImageModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6426.1, Definition 6426.2, and Definition 6426.3 seed the finite locator-and-modulus route for continuous-image location consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedImageModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host image sets, selected preimages, quotient equality of target names, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-locator exposure, modulus-accountable image routing, finite target-cell handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.LocatedImageModulusU

Chapter 6427

A Concrete Naming Certificate for $\mathsf{RationalIncompletenessWitness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ packet records the constructive finite evidence that $\mathsf{Rat}^{\uparrow}$ is not already completion-closed before the $\mathsf{Real}^{\uparrow}$ completion route is inspected. It sits after the dyadic arithmetic of chapter 291, the regular rational sequence interface of chapter 301, and the terminal real-name seal of chapter 38; it also reuses the irrationality obstruction style of chapter 1983. The nested dyadic contradiction route is a direct consumer of chapter 1983, and its completion-facing refusal later feeds the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ completion boundary through the displayed square-bound and tail-modulus rows. The packet names only a displayed nested-interval contradiction route, not a quotient real, not a selected limit, not countable choice, not host completeness of $\mathsf{Real}^{\uparrow}$, and not a classical theorem imported outside the BEDC carrier rows.

Definition 6427.1 (RationalIncompletenessWitness carrier)**.** *A* $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, M, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is a finite nested dyadic-interval ledger, recording left and right* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints, strict containment rows, and the finite square-bound comparisons attached to each inspected stage.*
2. *M is the Cauchy tail modulus row, exposing the displayed stage bound after which all rational candidates must remain inside the currently inspected dyadic interval.*
3. *S is the square-bound contradiction row, storing the finite parity, square comparison, and interval exclusion data used to reject a rational limit candidate for the nested route.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, naming the regular rational sequence surface whose finite windows are inspected by I, M, S.*
5. *R is the* $\mathsf{StreamName}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff row, carrying the terminal stream-name and real-name seal only after the dyadic nesting and rational-candidate contradiction rows have been displayed.*
6. *H records componentwise* hsame *transport for interval nesting, modulus, contradiction, regular rational readback, stream-name handoff, continuation, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a completion-facing request through M, then through the nested intervals I, the square-bound contradiction row S, the regular rational readback row Q, and finally the terminal handoff row R.*

8. *P is the* Pkg *provenance over* $\mathsf{RationalIncompletenessWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIncompletenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their componentwise transports, their displayed replay routes, their provenance, and their local names. It has no coordinate for quotient rational completion, selected irrational limits, host real equality, countable choice, or ambient completeness.

Theorem 6427.2 (RationalIncompletenessWitness NameCert obligations)**.** *For every accepted* $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ *carrier $W = (I, M, S, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalIncompletenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission, nested dyadic-interval ledger exposure, Cauchy tail modulus exposure, square-bound contradiction visibility, regular rational readback, stream-name and* $\mathsf{Real}^{\uparrow}$ *handoff, transport, replay, provenance, and local naming. No obligation is stored outside those finite rows.*

Proof. Carrier admission is projection from Definition 6427.1. The accepted packet displays the nested dyadic interval ledger I, the tail modulus M, the contradiction row S, the regular rational readback Q, and the terminal handoff R before any structural row is used.

The mathematical obligations are row-local. The interval ledger supplies only dyadic endpoints and containment comparisons, the modulus row supplies only the finite late-tail bound, and the contradiction row supplies only the square-bound exclusion attached to those displayed intervals.

The handoff and structural rows do not add a second source. The rows Q and R are downstream of I, M, S, while H, C, P, N transport, replay, source, and name the same finite route. Hence the listed rows exhaust the local certificate and leave no coordinate for quotient completion, selected limits, host equality, countable choice, or ambient completeness. □

Theorem 6427.3 (RationalIncompletenessWitness dyadic nesting)**.** *Every completion-facing read of an accepted* $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ *packet must first inspect a finite nested dyadic-interval route through I and M. The route displays dyadic endpoints from* $\mathsf{DyadicRatCore}^{\uparrow}$, *strict containment rows, the late-tail modulus budget, and the square-bound comparison windows before any* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *handoff is exposed.*

Proof. Project the packet $W = (I, M, S, Q, R, H, C, P, N)$. The only rows available before the regular rational and real-name handoff are the nested interval ledger I, the tail modulus M, and the square-bound contradiction row S.

The continuation row C fixes their order. A completion request first reads the displayed modulus budget, then the nested dyadic interval cells and their containment comparisons, and then the square-bound window attached to the same cells.

Only after this finite route is displayed can Q and R be inspected. Thus the completion-facing handoff cannot bypass dyadic endpoints, containment, and the late-tail budget by invoking a selected limit or host completeness. □

Theorem 6427.4 (RationalIncompletenessWitness no-rational-limit route)**.** *For every accepted* $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ *carrier $W = (I, M, S, Q, R, H, C, P, N)$, any claim that the displayed nested route has a rational limit is routed back to the finite square-bound contradiction row*

S after the dyadic nesting and Cauchy tail modulus rows have been exposed. The rejection is therefore a BEDC finite-history refusal: it does not assert quotient $\mathsf{Real}^{\uparrow}$ *equality, choose an irrational representative, use countable choice, or assume* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a rational-limit candidate read. The carrier forces the read through M and I, so the candidate is compared only after the finite tail budget and the inspected dyadic interval window are visible.

The contradiction row S then supplies the square-bound exclusion for that same window. Since S is part of the packet rather than an external theorem coordinate, the refusal is tied to the displayed dyadic comparisons and not to host real equality.

Finally use the downstream rows. The regular sequence readback Q, handoff row R, transport H, replay C, provenance P, and local name N preserve the same finite route. None of these rows can introduce quotient completion, a selected limit, countable choice, or ambient completeness, so every rational-limit claim is refused inside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIncompletenessWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIncompletenessWitness}^{\uparrow}$ packet is generated from nested dyadic interval rows, a Cauchy tail modulus row, square-bound contradiction data, regular rational readback, stream-name and RealUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIncompletenessWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6427.1, Definition 6427.2, Definition 6427.3, and Definition 6427.4 seed the finite no-rational-limit route that precedes completion-facing RealUp reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIncompletenessWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, a selected irrational limit, countable choice, host rational incompleteness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic nesting exactness, Cauchy tail modulus exposure, square-bound contradiction routing, regular rational readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalIncompletenessWitnessUp.

Chapter 6428

A Concrete Naming Certificate for $\mathsf{BaireMetricCompactWindow}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BaireMetricCompactWindow}^{\uparrow}$ packet records finite prefix-cylinder windows for compact-uniform-continuity consumers over the Baire metric. It reads the prefix carrier of chapter 334, the metric radius route of chapter 1602, compact inspection rows from chapter 129, stream schedules from chapter 126, and dyadic tolerances from chapter 291. The packet is a finite compact-window basis; it is not completed-stream equality, not quotient Baire-name equality, not open-cover compactness, not countable choice, and not a host real-valued metric.

Definition 6428.1 (BaireMetric compact-window carrier)**.** *A* $\mathsf{BaireMetricCompactWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, M, K_0, S, D, W, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, recording the inspected finite prefix family and cylinder-membership ledger.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *metric row, carrying first-disagreement data and dyadic radius exactness for the same prefix window.*
3. *K_0 is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-window source row, exposing only the finite compact inspection data available to downstream uniform-continuity consumers.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite prefix windows read from B and M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, recording the displayed dyadic tolerance and comparison budget.*
6. *W is the compact-window consumer row, requiring every compact-uniform-continuity read to inspect B, M, K_0, S, D before a modulus row is reached.*
7. *H records componentwise* hsame *transport for prefix cylinders, Baire metric radii, compact-window source data, stream schedules, dyadic tolerances, consumer windows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a compact-window request through K_0, then through S, B, M, D, and finally into the consumer row W.*

9. *P is the* Pkg *provenance over* $\mathsf{BaireMetricCompactWindow}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireMetricCompactWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows, their transports, their displayed replay routes, their provenance, and their local names. It has no coordinate for completed streams, quotient Baire names, open-cover compactness, countable-choice selectors, or host real metrics.

Theorem 6428.2 (BaireMetric compact-window NameCert obligations)**.** *For every accepted* BaireMetricCompactW *carrier $K = (B, M, K_0, S, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireMetricCompactWindow}^{\uparrow}}$ *surface consists exactly of prefix-cylinder admission, BaireMetric radius admission, compact-window source exposure, stream schedule exposure, dyadic tolerance exactness, compact-window consumer ordering, transport, replay, provenance, and local naming. No obligation is stored outside those rows.*

Proof. Carrier projection exposes $B, M, K_0, S, D, W, H, C, P, N$. These are the only coordinates available to a compact-window read.

The prefix and metric obligations are tied together. The schedule S selects a finite prefix window, B supplies the corresponding cylinder ledger, M supplies first-disagreement and radius data, and D records the dyadic tolerance used by the same read.

The compact source and consumer rows then fix the downstream route. The row K_0 exposes only finite compact inspection data, and W can consume it only after the prefix and dyadic rows have been displayed.

Transport, replay, provenance, and naming preserve that route. Since the carrier has no completed-stream, quotient, open-cover, countable-choice, or host-metric coordinate, the listed obligations exhaust the local certificate. □

Theorem 6428.3 (BaireMetric compact-window cylinder route)**.** *Every compact-window consumer of an accepted* $\mathsf{BaireMetricCompactWindow}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{BaireMetric}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{CompactMetric}^{\uparrow}.$$

The consumer reads a scheduled prefix cylinder, the corresponding first-disagreement radius, the displayed dyadic tolerance, and the compact-window source row before any uniform-continuity modulus is inspected.

Proof. Start with the consumer row W. By the carrier definition, W is downstream of the compact source K_0 and the prefix-metric rows S, B, M, D.

Replay the route through C. The schedule S opens the finite prefix window, B records the cylinder ledger, M reads the first-disagreement metric row, and D supplies the dyadic radius budget attached to that same window.

Only after this route is visible can K_0 be consumed as compact-window data. The structural rows H, P, N keep the same route under transport and provenance, so no compact-window consumer can use completed-stream equality, quotient Baire names, open-cover compactness, countable choice, or a host real metric. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireMetricCompactWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireMetricCompactWindow}^{\uparrow}$ packet is generated from finite BaireSpace prefix cylinders, BaireMetric first-disagreement radii, compact-window source rows, StreamName schedules, DyadicRatCore tolerances, compact-window consumer rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireMetricCompactWindow}\uparrow}$, seed)

Scope closed: Definition 6428.1, Definition 6428.2, and Definition 6428.3 seed the finite prefix-cylinder basis consumed by compact-window uniform-continuity reads.

Formal verification: FormalStatus(NameCert$_{\text{BaireMetricCompactWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed-stream equality, quotient Baire-name equality, open-cover compactness, countable choice, host real metrics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked prefix-cylinder admission, BaireMetric radius exactness, compact-window source exposure, StreamName schedule exposure, DyadicRatCore tolerance exactness, consumer ordering, transport, replay, provenance, and local naming for BEDC.Derived.BaireMetricCompactWindowUp.

Chapter 6429

A Concrete Naming Certificate for $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$

Brouwer's invariance of domain and its constructive topological descendants treat an injective Euclidean map as carrying local open information, but the $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ packet records only the finite NameCert rows visible before such a handoff reaches a Real or topology consumer. It sits beside the circle boundary carrier of chapter 58, the deformation rows of chapter 156, and the L10 Real finite-name route through chapter 126, chapter 291, chapter 301, and chapter 38. The packet names finite local chart, boundary, embedding, and Real-seal rows; it does not claim classical full open-cover compactness, host topological equality, selected inverse images, quotient embeddings, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6429.1 (Brouwer invariance-domain carrier)**.** *A* $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (U, E, S, Hm, W, D, R, A, T, C, P, N)$$

with the following displayed rows:

1. *U is the local Euclidean-domain chart row, read as a finite neighborhood and coordinate-window surface rather than as a host open subset.*
2. *E is the displayed embedding row, carrying the injective graph and local-image request consumed by downstream topology readers.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *boundary row used when the local chart is probed through a finite circle or sphere boundary.*
4. *Hm is the* $\mathsf{Homotopy}^{\uparrow}$ *deformation row connecting boundary probes, local chart windows, and embedding data without passing to a quotient homotopy class.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule attached to the Real-coordinate reads.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for coordinate and radius comparisons.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the Real-coordinate windows.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
9. *T records componentwise* hsame *transport for the local chart, embedding, boundary, homotopy, window, tolerance, readback, and seal rows.*
10. *C records displayed* Cont *replay from a local embedding request through the same finite route.*

11. *P is the* Pkg *provenance over* $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerInvarianceDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve displayed rows. It has no coordinate for host topological equality, global openness, open-cover compactness, selected inverse branches, quotient embeddings, countable choice, or ambient Real completeness.

Theorem 6429.2 (Brouwer invariance-domain NameCert obligations)**.** *For every accepted* BrouwerInvarianceDoma *carrier $I = (U, E, S, Hm, W, D, R, A, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BrouwerInvarianceDomain}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed local chart, embedding graph, circle-boundary row, homotopy row, finite StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal Real seal, componentwise transport, continuation replay, package provenance, and local naming. Every consumer that reads an invariance-domain handoff must first see the finite route*

$$\mathsf{S}^{1\uparrow},\ \mathsf{Homotopy}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

No obligation is stored as a host open set, a selected inverse image, a quotient embedding, a countable-choice schedule, or an ambient completeness theorem.

Proof. Project the carrier of Definition 6429.1. The projection exposes exactly the local chart row, embedding row, boundary row, homotopy row, analytic finite-name rows, and structural rows.

The boundary and deformation rows are read through the existing sibling packets. The row S supplies only the finite $\mathsf{S}^{1\uparrow}$ boundary surface of chapter 58, while Hm supplies only the displayed deformation ledger of chapter 156.

The Real-coordinate part is downstream. The rows W, D, R, A force every coordinate or radius read through the L10 Real finite-name route before a Real-facing conclusion is sealed.

Finally, T, C, P, N transport, replay, source, and name that same route. Since the carrier contains no host topology, selected inverse, quotient embedding, choice schedule, open-cover compactness, or ambient completeness row, the listed obligations exhaust the certificate surface. □

Theorem 6429.3 (Brouwer invariance-domain Euclidean open-embedding handoff)**.** *Every topology, manifold, Real, or Completion consumer of an accepted* $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ *carrier must read the Euclidean open-embedding request as a finite packet:*

$$U, E, S, Hm, W, D, R, A \longmapsto T, C, P, N.$$

The handoff preserves the displayed local chart and embedding graph, reads boundary control through chapter 58, reads deformation control through chapter 156, and reads coordinate precision through the L10 Real finite-name route. It does not assert classical full open-cover compactness, host topological equality, selected inverse branches, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6429.2. They require a consumer to inspect the displayed chart, embedding, boundary, homotopy, finite-name, transport, replay, provenance, and naming rows before using the packet.

The Euclidean chart and embedding rows remain finite carrier data. They record the local window and graph request, not a host open subset or a completed topological equality.

Boundary and deformation information is then routed through $\mathsf{S}^{1\uparrow}$ and $\mathsf{Homotopy}^{\uparrow}$. These sibling chapters supply the visible circle and homotopy rows used by local topological arguments while refusing quotient-circle, host path-space, and selected deformation objects.

The coordinate handoff closes through W, D, R, A. Thus the consumer reaches $\mathsf{Real}^{\uparrow}$ only after StreamName windows, DyadicRatCore tolerance rows, and RegSeqRat readback have been displayed, leaving no route by which compactness, choice, quotient equality, or ambient completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ open-embedding read displays one local chart row, one embedding row, one $\mathsf{S}^{1\uparrow}$ boundary row, one $\mathsf{Homotopy}^{\uparrow}$ deformation row, one $\mathsf{StreamName}^{\uparrow}$ schedule, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows T, C, P, N.

Independence witness: s1, homotopy, streamname, dyadicratcore, regseqrat, real: $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local chart data, embedding graph data, boundary rows, deformation rows, finite Real-coordinate schedules, tolerance ledgers, readback, terminal sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerInvarianceDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerInvarianceDomain}^{\uparrow}$ packet is generated from a displayed local Euclidean chart, an embedding graph row, a circle-boundary row, a homotopy deformation row, finite StreamName coordinate windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerInvarianceDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6429.1, Definition 6429.2, and Definition 6429.3 seed the finite invariance-domain handoff over local chart, embedding, boundary, homotopy, StreamName, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerInvarianceDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical full open-cover compactness, host topological equality, selected inverse branches, quotient embeddings, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, local chart exposure, embedding graph routing, boundary and homotopy handoff, Real finite-name ordering, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerInvarianceDomainUp.

Chapter 6430

A Concrete Naming Certificate for $\mathsf{HadamardGapSeries}^{\uparrow}$

Following Hadamard's lacunary power-series theorem and Bishop–Bridges constructive series practice, the $\mathsf{HadamardGapSeries}^{\uparrow}$ packet records a gap-series coefficient route as finite NameCert data before any analytic-continuation or boundary-singularity theorem is available. It sits beside the regular rational readback of chapter 301, the terminal real seal of chapter 38, the uniform tail route of chapter 1736, and the coefficientwise product neighbour chapter 3811. The packet names lacunary coefficient windows and their tail budget; it is not a host infinite sum, not a complex analytic function, not a natural-boundary theorem, and not an ambient completeness principle.

Definition 6430.1 (Hadamard gap series carrier)**.** *A* $\mathsf{HadamardGapSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, G, C, T, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the coefficient-window data: a finite list of inspected nonzero coefficient slots, their displayed rational or real-name coefficients, and the zero-window declarations between inspected slots.*
2. *G is the lacunary gap-growth ledger, recording the displayed increasing exponent schedule and the local growth comparisons used to certify the Hadamard gap condition on the inspected window.*
3. *C is the compact dyadic coefficient-bound row, containing the dyadic radius, coefficient bound, and compact-window estimate read through the available coefficient rows.*
4. *T is the uniform tail budget, naming a finite threshold and tail tolerance for partial sums after the gap and coefficient ledgers have been read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, repacking coefficient windows and partial-sum windows as regular rational readback data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, G, C, T, R have been replayed.*
7. *H records componentwise* hsame *transport for coefficient windows, gap-growth ledgers, dyadic bounds, tail budgets, readback rows, Real seals, replay, provenance, and local naming.*
8. *K records displayed* Cont *replay from a coefficient or tail request through $A, G, C, T, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{HadamardGapSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HadamardGapSeries}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, G, C, T, R, E, H, K, P, N$. It has no coordinate for host analytic continuation, natural-boundary claims, completed power-series functions, quotient stream equality, selected infinite sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6430.2 (Hadamard gap series NameCert obligations)**.** *For every accepted* $\mathsf{HadamardGapSeries}^{\uparrow}$ *carrier*

$$S = (A, G, C, T, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HadamardGapSeries}^{\uparrow}}$ *surface consists exactly of carrier admission, lacunary growth exactness, coefficient-window stability, uniform Cauchy tail handoff, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming through $A, G, C, T, R, E, H, K, P, N$.*

Proof. Project the accepted carrier. The coefficient data A and the gap ledger G are the only rows that can name nonzero slots and lacunary spacing, while the dyadic bound C and tail budget T are the only rows that can bound a partial-sum request.

The readback and seal rows are downstream. Row R repacks the displayed coefficient and partial-sum windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, and row E exposes the terminal $\mathsf{Real}^{\uparrow}$ seal only after A, G, C, T, R have been replayed.

The structural rows H, K, P, N transport, replay, source, and name the same finite route. Since no excluded analytic or choice-like coordinate is present in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 6430.3 (Hadamard gap growth exactness)**.** *For an accepted* $\mathsf{HadamardGapSeries}^{\uparrow}$ *packet $S = (A, G, C, T, R, E, H, K, P, N)$, the lacunary growth condition is read exactly from G against the coefficient-window row A. A consumer may inspect the displayed exponent schedule, adjacent gap comparisons, and zero-window declarations, but it cannot replace them by a host sequence limit or an analytic continuation argument.*

Proof. Read the finite exponent schedule inside G. Each inspected comparison is tied to a coefficient slot or zero-window declaration in A, so the gap condition is a displayed ledger relation rather than a property of an unbounded host sequence.

Transport through H and replay through K preserve only those displayed rows. The provenance and naming rows record the same local packet, hence no analytic-continuation principle or host limit can be exported from the gap ledger. □

Theorem 6430.4 (Hadamard gap coefficient-window stability)**.** *Every transported coefficient-window read of an accepted* $\mathsf{HadamardGapSeries}^{\uparrow}$ *packet remains inside the displayed rows A, G, C, H, K, P, N. The transport preserves inspected coefficient slots, zero-window declarations, gap comparisons, and compact dyadic coefficient bounds, and it exports no quotient coefficient stream or selected representative.*

Proof. Project the coefficient-facing rows. The row A lists the inspected slots and zero-window declarations, G accounts for their lacunary spacing, and C supplies the dyadic coefficient bound used on the same window.

Componentwise transport can only compare these rows through H. The replay, provenance, and naming rows preserve the displayed coefficient window, so no quotient stream equality or selected representative is introduced. □

Theorem 6430.5 (Hadamard gap uniform tail handoff)**.** *Every uniform Cauchy tail read through an accepted* $\mathsf{HadamardGapSeries}^{\uparrow}$ *packet factors through*

$$A, G, C \longmapsto T \longmapsto R \longmapsto E.$$

Thus the handoff to the regular-Cauchy series route is a finite tail-budget and regular-readback route, not a selected infinite sum or an ambient completeness theorem.

Proof. Begin with the coefficient and gap data. The rows A, G, C supply the inspected coefficients, gap ledger, and dyadic coefficient bound before any tail estimate is available.

The uniform tail row T then names the finite threshold and tolerance budget. Row R repacks the resulting partial-sum window as $\mathsf{RegSeqRat}^{\uparrow}$ data, and row E is downstream of that readback.

Because H, K, P, N preserve the same displayed route, a consumer reaches the uniform Cauchy series surface without obtaining a selected infinite sum, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6430.6 (Hadamard gap Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read through an accepted* $\mathsf{HadamardGapSeries}^{\uparrow}$ *packet is downstream of the coefficient-window data A, lacunary gap ledger G, compact dyadic coefficient-bound row C, uniform tail budget T, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R. The seal cannot export host analytic continuation, natural-boundary information, quotient stream equality, selected sums, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the terminal row E. It appears only after the coefficient, gap, bound, tail, and readback rows have been displayed.

Use the carrier definition to trace every Real-facing read through A, G, C, T, R and then through the structural rows H, K, P, N. None of those rows contains analytic-continuation data, a boundary-singularity theorem, quotient equality, a selector for an infinite sum, choice data, or Real completeness.

Therefore the seal is a finite BEDC endpoint for the gap-series route and cannot escape the displayed NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HadamardGapSeries}^{\uparrow}$ endpoint read factors through coefficient-window data, lacunary gap-growth ledgers, compact dyadic coefficient bounds, uniform tail budgets, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, displayed replay, provenance, and local NameCert rows.

Independence witness: regseqrat, real, uniform-cauchy-series, hadamard-product-series: RegSeqRatUp supplies regular rational readback without lacunary coefficient windows; RealUp supplies terminal seals without gap-growth ledgers; UniformCauchySeriesUp supplies shared tail budgets without the Hadamard exponent schedule; HadamardProductSeriesUp supplies coefficientwise products without lacunary zero-window declarations. None is bijective to $\mathsf{HadamardGapSeries}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardGapSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardGapSeries}^{\uparrow}$ packet is generated from coefficient-window data, a lacunary gap-growth ledger, compact dyadic coefficient bounds, a uniform tail budget, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardGapSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6430.1, Definition 6430.2, Definition 6430.3, Definition 6430.4, Definition 6430.5, and Definition 6430.6 seed the finite lacunary coefficient route from gap windows to uniform tail budget, regular readback, and terminal real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardGapSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, natural-boundary theorems, completed power-series functions, quotient stream equality, selected infinite sums, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lacunary growth exactness, coefficient-window stability, compact dyadic coefficient bounds, uniform Cauchy tail handoff, RegSeqRat readback, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.HadamardGapSeriesUp.

Chapter 6431

A Concrete Naming Certificate for $\mathsf{ConstructiveIVTModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ packet records the finite sign-change modulus surface behind a located intermediate-value bisection request. It sits after the located interval source of chapter 1709, the constructive intermediate-value packet of chapter 6410, the dyadic bisection tree of chapter 6415, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exports a finite modulus and retained-window handoff, not a selected root, host root-selection theorem, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6431.1 (Constructive IVT modulus carrier)**.** *A* $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, A, B, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row. It records endpoint names, a displayed interval orientation, locator reads, and the finite bracket over which the modulus request is made.*
2. *A is the opposite-sign witness row. It records the two endpoint sign certificates and the retained sign-change condition used by the bisection schedule.*
3. *B is the finite bisection schedule row. It records the requested depth, midpoint inspections, retained child intervals, and branch evidence.*
4. *D is the dyadic tolerance ledger. It records the tolerance request, diameter budget, midpoint denominator data, and finite contraction rows used by B.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the retained endpoint and midpoint reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, A, B, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, sign witnesses, bisection schedule, tolerance ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a modulus request through $I, A, B, D, S, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveIVTModulus}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$, $\mathsf{DyadicBisectionTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveIVTModulus}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite sign-change modulus history. It has no coordinate for host root selection, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6431.2 (Constructive IVT modulus NameCert obligations)**.** *For every accepted* ConstructiveIVTModulus *carrier*

$$Q = (I, A, B, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveIVTModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval source I, opposite-sign witness row A, finite bisection schedule B, dyadic tolerance ledger D,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance row P, and local naming row N.*

Proof. Project the carrier of Definition 6431.1. The accepted packet exposes the located interval source, sign witnesses, bisection schedule, tolerance ledger, finite windows, regular readback, terminal seal, and structural rows.

The analytic rows are I, A, B, D, S, R, E. The interval row fixes the displayed bracket, the sign row supplies the endpoint and retained sign-change certificates, and the bisection and tolerance rows bind each midpoint inspection to a finite diameter budget. The stream, regular-readback, and Real-seal rows are downstream of those displayed inspections.

The structural rows are H, C, P, N. They transport the same finite data, replay the same route, record provenance, and bind the local name. Since the carrier contains no further coordinate, every local NameCert obligation is one of the listed rows. □

Theorem 6431.3 (Constructive IVT modulus bisection-window handoff)**.** *Every accepted* ConstructiveIVTModulus *request that asks for a sign-change modulus must factor through the finite route*

$$I, A \longmapsto B, D \longmapsto S, R \longmapsto E, H, C, P, N.$$

Thus the retained bisection window is exposed only after the located interval source, opposite-sign witnesses, bisection schedule, and dyadic tolerance ledger have been read; the route cannot provide a host-selected root, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $Q = (I, A, B, D, S, R, E, H, C, P, N)$. The source stage is I, A: the located interval row fixes the bracket and the opposite-sign row records the endpoint and retained sign evidence.

The bisection stage is B, D. Each midpoint inspection and retained child interval is paired with a dyadic tolerance entry, so the request opens only the finite schedule specified by the carrier.

The window and readback stage is S, R. The $\mathsf{StreamName}^{\uparrow}$ windows are opened from the retained endpoint and midpoint reads, and the $\mathsf{RegSeqRat}^{\uparrow}$ row repacks exactly those windows against the same tolerance ledger.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Because no row carries a root selector, quotient equality, trichotomy split, choice schedule, open-cover compactness proof, or completeness principle, the displayed handoff is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveIVTModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ packet is generated from a located interval source, opposite-sign witnesses, a finite bisection schedule, a dyadic tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveIVTModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6431.1, Definition 6431.2, and Definition 6431.3 seed the finite sign-change modulus route from located interval and sign rows through bisection, StreamName, RegSeqRat, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveIVTModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host root selection, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, sign-witness stability, bisection schedule exactness, tolerance-ledger discipline, finite-window exposure, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveIVTModulusUp.

Chapter 6432

A Concrete Naming Certificate for $\mathsf{LocatedSupremumModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedSupremumModulus}^{\uparrow}$ packet records the finite data by which a located bounded subset of a constructive real line exposes a supremum-search modulus. It sits beside the dyadic tolerance route of chapter 291, the regular rational readback of chapter 301, the real seal of chapter 38, the bounded-set carrier of chapter 1757, and the bracketing interval route of chapter 1807. The packet extracts located interval brackets and dyadic tolerances; it does not assert classical least-upper-bound completeness, trichotomy, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6432.1 (Located supremum modulus carrier)**.** *A* $\mathsf{LocatedSupremumModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, I, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedSet}^{\uparrow}$ *row, recording the bounded-set membership window, upper-bound ledger, and nonempty witness surface used by the supremum search.*
2. *I is the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *row, storing located interval brackets $[l, u]$ and the side evidence that the set is below u while values can be found inside any admitted lower window near l.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing dyadic precision requests, refinement steps, and interval-width certificates.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating dyadic bracket refinements into regular rational observations before the real seal is consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after bounded-set, bracket, tolerance, and readback rows have been displayed.*
6. *H records componentwise* hsame *transport for bounded-set data, interval brackets, dyadic tolerances, regular-rational readbacks, the seal row, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a supremum-modulus request through B, I, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedSupremumModulus}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FiniteBracketingInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSupremumModulus}^{\uparrow}}$ *row.*

The carrier exports only the displayed modulus route $B, I, D, R, E, H, C, P, N$. It has no coordinate for a classical supremum theorem, trichotomy, quotient equality of real names, countable choice, a selected maximizing sequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6432.2 (Located supremum modulus NameCert obligations)**.** *For every accepted* LocatedSupremumMod *carrier*

$$M = (B, I, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSupremumModulus}^{\uparrow}}$ *surface consists exactly of bounded-set admission through B, located interval bracket exposure through I, dyadic tolerance exactness through D, regular rational readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6432.1. The analytic rows are exactly the bounded-set row, located-bracket row, dyadic tolerance row, regular rational readback row, and terminal Real seal row.

The bracket row is read through the bounded-set row. The upper side records displayed upper-bound evidence, while the lower side records the located search window rather than a choice-selected maximizing sequence.

The dyadic tolerance row supplies the modulus data. It refines bracket widths through $\mathsf{DyadicRatCore}^{\uparrow}$ entries and then exposes those refinements as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks before the $\mathsf{Real}^{\uparrow}$ seal may be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier row stores trichotomy, quotient equality, countable choice, classical least-upper-bound completeness, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Lean4 (statement) `BEDC.Derived.LocatedSupremumModulusUp`

Falsifiable prediction: A $\mathsf{LocatedSupremumModulus}^{\uparrow}$ consumer that exports a supremum-modulus read while omitting the bounded-set row, located interval bracket row, dyadic tolerance row, RegSeqRat readback row, Real seal row, hsame, Cont, Pkg, or NameCert row is not reading the carrier of Definition 6432.1.

Independence witness: boundedset, finite-bracketing-interval, dyadicratcore, regseqrat, real: $\mathsf{LocatedSupremumModulus}^{\uparrow}$ jointly binds bounded-set data, located interval brackets, dyadic tolerance refinement, regular rational readback, and the Real seal into one modulus packet. Each sibling lacks at least one of those rows as a local supremum-modulus coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSupremumModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSupremumModulus}^{\uparrow}$ packet is generated from bounded-set rows, located finite interval brackets, dyadic tolerance rows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSupremumModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6432.1 and Definition 6432.2 seed the constructive located supremum-modulus packet over bounded-set, interval-bracket, dyadic tolerance, regular-rational readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSupremumModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim classical least-upper-bound completeness, trichotomy, quotient equality, countable choice, selected maximizing sequences, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket exactness, dyadic tolerance refinement, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSupremumModulusUp.

Chapter 6433

A Concrete Naming Certificate for MetaCICTypedParallelDiamondBoundary$^{\uparrow}$

The MetaCICTypedParallelDiamondBoundary$^{\uparrow}$ packet realizes the typed-window boundary of chapter 9378 as finite confluence-facing NameCert data. It sits beside the residual-substitution boundary of chapter 2181, the critical-pair boundary of chapter 2212, the residual checker of chapter 3827, the local-join ledger of chapter 4126, and the existing parallel-diamond frontier of chapter 4238. The packet records typed windows, residual-substitution premises, critical pairs, local joins, checker rows, obstruction rows, transport, replay, provenance, and local naming without claiming subject reduction, full Church–Rosser, premise-free parallel diamond, evaluator equality, quotient term equality, or kernel consistency.

Definition 6433.1 (MetaCIC typed parallel-diamond boundary carrier)**.** *A* MetaCICTypedParallelDiamondBoundar *carrier is a finite* BHist *packet*

$$T = (W, S, K, J, R, C, N, B, O, H, Q, P, M)$$

with the following displayed rows:

1. *W is the typed-window row, recording context, closedness, binder-depth, and type-checker visibility for the local peak.*
2. *S is the residual-substitution premise row, naming the substitution-compatibility input retained by the confluence audit map.*
3. *K is the critical-pair row, exposing the two branch observations and the local peak boundary.*
4. *J is the local-join row, recording the candidate common reduct and the finite branch-local join evidence.*
5. *R is the residual-frontier row, listing residual descendants, residual budget, and branch-local substitution windows.*
6. *C is the closed-checker row, recording bounded checker acceptance before the typed handoff is consumed.*
7. *N is the closed-normal row, recording the already available normal-source and atom-source fragments.*
8. *B is the bounded-check row over typed-window visibility, residual budget, closed checker, and local-join compatibility.*

9. *O is the obstruction row, retaining the endpoints not supplied by the packet: residual-substitution discharge, subject reduction, full Church–Rosser, premise-free parallel diamond, evaluator equality, quotient term equality, and kernel consistency.*
10. *H records componentwise* hsame *transport for typed-window, premise, critical-pair, local-join, residual, checker, normal, bounded-check, obstruction, replay, provenance, and local naming rows.*
11. *Q records displayed* Cont *replay from a typed confluence query through $W, S, K, J, R, C, N, B, O, H, P, M$.*
12. *P is the* Pkg *provenance over* $\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}$*, the MetaCIC residual and critical-pair siblings,* BHist*,* hsame*,* Cont*, and* NameCert*.*
13. *M is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for subject reduction, full closed Church–Rosser, a premise-free parallel diamond, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 6433.2 (MetaCIC typed parallel-diamond boundary obligations)**.** *For every accepted* $\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}$ *carrier*

$$T = (W, S, K, J, R, C, N, B, O, H, Q, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}}$ *surface consists exactly of typed-window exposure through W, residual-substitution premise retention through S, critical-pair exposure through K, local-join exposure through J, residual-frontier exposure through R, closed-checker handoff through C, closed-normal fragment exposure through N, bounded-check exactness through B, obstruction retention through O, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through M.*

Proof. Project the carrier of Definition 6433.1. The confluence-facing rows are $W, S, K, J, R, C, N, B, O$, and the structural rows are H, Q, P, M.

The typed window is first-class. It records the finite context and checker visibility under which the local peak is read, while S keeps the residual-substitution premise visible rather than discharged.

Rows K, J, R, C, N, B then expose the critical pair, candidate join, residual frontier, closed checker, closed-normal fragment, and bounded checks. The obstruction row O remains attached to the same packet.

Finally H, Q, P, M transport, replay, source, and name the displayed rows. Since the carrier has no row for subject reduction, full Church–Rosser, premise-free parallel diamond, evaluator equality, quotient term equality, or kernel consistency, the listed rows exhaust the local obligations. □

Lean4 (statement) `BEDC.Derived.MetaCICTypedParallelDiamondBoundaryUp`

Theorem 6433.3 (MetaCIC typed parallel-diamond boundary nonescape)**.** *Every typed confluence consumer of an accepted* $\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}$ *packet must keep $W, S, K, J, R, C, N, B, O, H$ visible before citing the packet as a local parallel-diamond boundary. The consumer may feed the conditional frontier of chapter 4238, but it may not cite the packet as subject reduction, full Church–Rosser, a premise-free parallel diamond, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Apply the obligation theorem Definition 6433.2. It keeps the typed window, residual-substitution premise, critical pair, local join, residual frontier, checker, normal fragment, bounded-check, and obstruction rows visible.

Use the existing frontier boundary of Definition 4238.16. That theorem permits a typed-window consumer to read a conditional handoff only while retaining the premise, residual, checker, bounded-check, and obstruction rows.

The transport, replay, provenance, and local naming rows preserve the same route for downstream audit consumers. They do not add a row that discharges residual substitution or converts the typed boundary into any stronger confluence theorem.

Therefore the packet can only be cited as a typed boundary feeding the conditional parallel-diamond frontier. A consumer that erases S or O, or reads the handoff as one of the refused endpoints, has left the carrier surface. □

Falsifiable prediction: A MetaCIC confluence read that cites $\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}$ while omitting the typed-window row, residual-substitution premise row, critical-pair row, local-join row, residual-frontier row, checker row, obstruction row, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 6433.1.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, metacic-confluence-residual-checker, metacic-local-join-ledger, metacic-parallel-diamond-frontier: these siblings respectively expose premise retention, peak data, residual checking, local joins, and a conditional frontier. None is bijective with the packet that jointly binds typed-window visibility, all four confluence rows, checker and normal rows, bounded checks, obstruction retention, transport, replay, provenance, and local naming.

The Lean TasteGate surface must include both Nontrivial and FieldFaithful instances for the finite carrier: nontriviality separates accepted typed-boundary packets from obstruction-only packets, and field faithfulness matches exactly the rows $W, S, K, J, R, C, N, B, O, H, Q, P, M$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}$ packet is generated from typed-window rows, residual-substitution premise rows, critical-pair rows, local-join rows, residual-frontier rows, checker rows, closed-normal rows, bounded-check rows, obstruction rows, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6433.1, Definition 6433.2, and Definition 6433.3 seed the typed parallel-diamond boundary packet for MetaCIC confluence audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim residual-substitution compatibility, subject reduction, full Church–Rosser, premise-free parallel diamond, evaluator equality, quotient term equality, kernel consistency, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, residual-substitution premise retention, critical-pair exposure, local-join exactness, residual-frontier coverage, checker and closed-normal routing, bounded-check exactness, obstruction retention, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedParallelDiamondBoundaryUp.

Chapter 6434

A Concrete Naming Certificate for $\mathsf{BishopLocatedFilterBase}^{\uparrow}$

Following Bishop and Bridges and Kelley, a located filter base is read here as finite boundary data for Real and Completion consumers: it exposes a rounded filter-base request, a Cauchy-filter-base tail budget, a located-real bracket, displayed stream windows, dyadic tolerances, regular rational readback, and a terminal real seal before any completed object is named. The $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ packet sits between the filter-base surface of chapter 1601, the Cauchy-filter-base surface of chapter 1915, the located-real filter surface of chapter 5897, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real boundary of chapter 38. It names a traditional finite filter-base route without quotient filters, selected limits, countable choice, host equality of filters, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6434.1 (Bishop located filter-base carrier)**.** *A* $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, L, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row, carrying the rounded finite basis requests and inclusion evidence inspected by the consumer.*
2. *L is the* $\mathsf{CauchyFilterBase}^{\uparrow}$ *and located-real bracket row. It records the Cauchy tail budget, lower and upper located brackets, and the handoff discipline inherited from* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$*.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger, displaying the finite source windows read by the bracket and tail requests.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the positive dyadic scale, bracket radius, and split budget used by the located filter-base read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from W and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after F, L, W, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the filter base, Cauchy-base bracket, window, dyadic tolerance, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located filter-base request through $F, L, W, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedFilterBase}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, W, D, R, E, H, C, P, N$. It has no coordinate for quotient filters, host powersets, selected subfilters, selected limits, countable-choice schedules, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BishopLocatedFilterBaseUp.BishopLocatedFilterBaseUp`

Theorem 6434.2 (Bishop located filter-base NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ *carrier*

$$B = (F, L, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}$ *surface consists exactly of FilterBase admission for F, CauchyFilterBase and located-real bracket admission for L, finite StreamName window coverage for W, dyadic tolerance visibility for D, RegSeqRat readback for R, Real seal routing for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No Bishop located filter-base obligation is stored outside these rows.

Proof. Project the carrier of Definition 6434.1. The accepted packet is exactly the tuple $F, L, W, D, R, E, H, C, P, N$ so every local naming obligation must be read from one of those rows.

The analytic route starts with the rounded filter-base row F and the Cauchy located-bracket row L. The finite window ledger W and dyadic tolerance ledger D expose the inspected source windows, bracket radius, and split budget before the regular rational readback R is visible.

The Real seal E is downstream of this displayed route. Transport, replay, provenance, and local naming through H, C, P, N preserve the same evidence and add no quotient filter, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. □

Theorem 6434.3 (Bishop located filter-base Real handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FilterBase}^{\uparrow},\ \mathsf{CauchyFilterBase}^{\uparrow},\ \mathsf{BishopLocatedRealFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Equivalently, the consumer first reads the rounded basis row F, then the Cauchy located-bracket row L, then the finite windows W, dyadic tolerance ledger D, and regular rational readback R, and only then reaches the terminal Real seal E with H, C, P, N preserving the same route.

Proof. Apply the obligation theorem. It orders carrier admission through the filter base and located Cauchy bracket before any finite window, tolerance, readback, or Real-facing seal is available.

The StreamName, DyadicRatCore, and RegSeqRat rows provide the finite observation route. Because the bracket and Cauchy tail budget are displayed in L, a consumer cannot replace this route by a quotient filter, selected subfilter, or selected limit.

Finally the Real seal is read only after F, L, W, D, R have been displayed and transported. The structural rows H, C, P, N replay, source, and name the same packet, so every Real or Completion handoff factors through the stated finite boundary. □

Theorem 6434.4 (Bishop located filter-base nonescape)**.** *An accepted* $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ *packet cannot export quotient filter equality, host powerset membership, a selected subfilter, a selected limit, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness to a Real or Completion consumer. Any such export would require a coordinate outside $F, L, W, D, R, E, H, C, P, N$ and therefore outside the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}$ *surface.*

Proof. Read the carrier definition together with the NameCert obligations. The only public rows are the rounded filter base, located Cauchy bracket, finite windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, transport, replay, provenance, and local name.

Each excluded object would need an additional coordinate: quotient filters require a quotient equality row, selected subfilters and selected limits require choice rows, host powersets and host equality require external host data, and ambient completeness requires a global Real principle. None of those coordinates occurs in $F, L, W, D, R, E, H, C, P, N$.

Thus every accepted packet remains inside the finite located filter-base boundary, and Real or Completion consumers can inspect only the displayed rows. □

Falsifiable prediction: A $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ consumer that reaches a Real or Completion output after deleting F, L, W, D, or R refutes the carrier of Definition 6434.1.

Independence witness: filterbase, cauchy-filter-base, bishop-located-real-filter, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rounded filter-base admission, located Cauchy bracket data, finite windows, dyadic tolerances, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedFilterBase}^{\uparrow}$ packet is generated from a FilterBase row, a CauchyFilterBase located-bracket row, StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6434.1, Definition 6434.2, Definition 6434.3, and Definition 6434.4 seed the finite located filter-base route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedFilterBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, host powersets, selected subfilters, selected limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located filter-base exactness, finite-window coverage, dyadic tolerance visibility, RegSeqRat readback, Real seal routing, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.BishopLocatedFilterBaseUp.

Chapter 6435

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ packet records the finite BEDC surface of the standard unit map from a metric source into its Cauchy completion, with distance preservation exposed before any quotient-completion equality or selected limit is available. It sits after the functorial completion route of chapter 423, the dense embedding boundary of chapter 3351, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet exports the unit-isometry witness needed by Real and Completion consumers, not a quotient construction of the completion.

Definition 6435.1 (Cauchy completion unit-isometry carrier)**.** *A* $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, R, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, including the displayed source points, metric-distance requests, and finite point-classifier observations consumed by the unit map.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the constant Cauchy names associated to the displayed source points.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested dyadic radius, the comparison budget, and the finite descent used to compare source distances with completion distances.*
4. *U is the unit-admission row supplied by the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *route, saying that each displayed source point is sent to its constant completion name before any completion consumer can read it.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the distance comparison after the source metric row, regular rational readback, dyadic tolerance ledger, and unit-admission row have been exposed.*
6. *H records componentwise* hsame *transport for the source row, readback row, dyadic ledger, unit-admission row, Real seal, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from source distance requests through S, R, D, U, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, MetricCompletionDen StreamName$^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite unit-isometry history. It has no coordinate for quotient completion equality, selected representatives, host complete spaces, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6435.2 (Cauchy completion unit-isometry NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ *carrier*

$$J = (S, R, D, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}}$ *surface consists exactly of carrier admission for the source metric row S, regular rational readback row R, dyadic tolerance ledger D, unit-admission row U, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance row P, and local naming row N. The certificate is local to the constant-Cauchy-name unit route.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitIsometryUp.CauchyCompletionUnitIsometryTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 6435.1. The accepted packet exposes the source metric row, regular readback, dyadic ledger, unit admission, terminal seal, and structural support rows.

The analytic rows are S, R, D, U, E. The source row names the points and their distance requests, the readback row names the constant regular Cauchy observations, and the dyadic ledger binds the requested comparison tolerance. The unit row admits the constant completion names before the Real seal is read.

The structural rows are H, C, P, N. They transport the same displayed data, replay the same route, record provenance, and bind the local name. Since the carrier contains no additional coordinate, every local NameCert obligation is one of the listed rows. □

Theorem 6435.3 (Cauchy completion unit-isometry distance route)**.** *Every accepted* CauchyCompletionUnitIsometr *carrier routes a displayed source-distance comparison through the finite chain*

$$S \longmapsto U \longmapsto R, D \longmapsto E, H, C, P, N.$$

The completion distance read is therefore the same finite regular-rational and dyadic comparison attached to the source metric row; it is not a quotient equality of completion classes and not a selected limiting representative.

Proof. Begin with an accepted carrier $J = (S, R, D, U, E, H, C, P, N)$. The source stage is S: the displayed metric row fixes the two source points and the distance request being compared.

Use the unit-admission row U next. It sends each displayed source point to its constant Cauchy completion name before any distance readback is available, so the completion-side comparison is attached to the same source observations.

Read R, D as the finite comparison ledger. The regular rational row supplies the constant-name observations, and the dyadic row supplies the tolerance budget for comparing those observations with the source distance request.

Finally E, H, C, P, N seal, transport, replay, source, and name the displayed route. No quotient class, selected representative, host complete space, or choice schedule appears in the packet, so the distance route is exhausted by the listed finite rows. □

Theorem 6435.4 (Cauchy completion unit-isometry Real-seal consumer)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ *packet must reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal by first reading the source metric row* S*, the unit-admission row* U*, the regular rational readback* R*, and the dyadic tolerance ledger* D*. The consumer receives the unit-isometry witness only as a finite Real-seal route over constant Cauchy names, with support rows* H, C, P, N *preserving the same packet.*

Proof. Project the accepted packet. The terminal row E is not a detached real object; it is displayed only after S, U, R, D have fixed the source distance request, the constant completion names, and the dyadic comparison budget.

Use Definition 6435.3. It orders the completion-facing distance read through the same source, unit, readback, and tolerance rows before the Real seal is available.

The support rows H, C, P, N preserve that route under componentwise transport, displayed replay, provenance, and local naming. A Real-facing consumer therefore obtains the unit-isometry witness through finite NameCert data only, not through quotient completion equality, selected limits, ambient completeness, or choice. □

Falsifiable prediction: Every $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ Real-facing read factors through one source metric row, one unit-admission row, one regular rational readback row, one dyadic tolerance ledger, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchycompletionfunctor, metric-completion-dense-embedding, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source metric row, unit-admission row, regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local NameCert rows as one finite unit-isometry packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}$ packet is generated from a source metric row, constant-Cauchy-name unit admission, RegSeqRat readback, DyadicRatCore tolerance comparison, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6435.1, Definition 6435.2, Definition 6435.3, and Definition 6435.4 seed the finite unit-isometry route from source metric data through constant completion names, RegSeqRat readback, DyadicRatCore tolerance comparison, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitIsometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host complete spaces, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source metric stability, unit-admission exactness, regular readback, dyadic tolerance discipline, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionUnitIsometryUp.

Chapter 6436

A Concrete Naming Certificate for $\mathsf{DarbouxSums}^{\uparrow}$

Following Darboux and Riemann constructive integration, the $\mathsf{DarbouxSums}^{\uparrow}$ packet records the finite partition-indexed lower and upper sum surface before an integration consumer reads a limiting or Cauchy route. It sits after the real interval seal of chapter 38, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, the finite partition surface of chapter 201, the single Darboux comparison row of chapter 1880, and the Riemann integral consumer of chapter 1526. The packet gives Real and Completion routes a finite family of partition sums without treating integration, supremum, or real completeness as primitive.

Definition 6436.1 (DarbouxSums carrier)**.** *A* $\mathsf{DarbouxSums}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, P, L, U, SL, SU, G, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-source row naming the endpoint seals and the orientation of the finite partition request.*
2. *P is the* $\mathsf{Partition}^{\uparrow}$ *row naming the ordered finite cell list, adjacency data, and partition coverage.*
3. *L and U are lower-value and upper-value ledgers over the cells of P, with every value read through the same displayed partition route.*
4. *SL and SU are the finite lower and upper sum rows computed from the cell widths of P and the displayed L, U value ledgers.*
5. *G is the upper-minus-lower gap comparison row, exposed only after P, L, U, SL, SU have been displayed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational sum observations attached to the same finite partition and value ledgers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reachable only after R and the finite Darboux rows have been projected.*
8. *H records componentwise* hsame *transport for interval, partition, value, finite-sum, gap, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the route*

$$I, P, L, U \longmapsto SL, SU \longmapsto G, R \longmapsto E.$$

10. *Q is the* Pkg *provenance over* $\mathsf{DarbouxSums}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxSums}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, P, L, U, SL, SU, G, R, E, H, C, Q, N$. It has no coordinate for quotient functions, host suprema, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6436.2 (DarbouxSums NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxSums}^{\uparrow}$ *carrier $D = (I, P, L, U, SL, SU, G, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxSums}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *interval source, finite* $\mathsf{Partition}^{\uparrow}$ *row, lower and upper value ledgers, lower and upper finite sum rows, upper-minus-lower gap row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every obligation is read through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-width arithmetic and the existing* $\mathsf{DarbouxSum}^{\uparrow}$ *comparison boundary before any* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer is available.*

Proof. Project the carrier of Definition 6436.1. The interval, partition, lower-value, upper-value, finite-sum, gap, readback, and seal rows are the only analytic rows carried by the packet.

The partition and arithmetic obligations are finite. Row P supplies the ordered cell list, and $\mathsf{DyadicRatCore}^{\uparrow}$ support in Q supplies the displayed cell-width arithmetic used by SL and SU.

The Darboux comparison obligation is read through G. It is downstream of P, L, U, SL, SU and agrees with the single $\mathsf{DarbouxSum}^{\uparrow}$ boundary cited by the provenance row.

Finally R, E, H, C, Q, N read back, seal, transport, replay, source, and name the same route. Since no coordinate stores quotient functions, host suprema, selected limits, countable choice, or real completeness, no NameCert obligation can export those objects. □

Theorem 6436.3 (DarbouxSums real partition scope)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *or Real-facing consumer of an accepted* $\mathsf{DarbouxSums}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Real}^{\uparrow}, \quad \mathsf{Partition}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{DarbouxSum}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as exposed by $I, P, SL, SU, G, R, E, H, C, Q, N$. The route supplies lower and upper finite sums over displayed partition cells, but it refuses quotient functions, host suprema, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the partition rows I and P. They expose only a displayed real interval source and a finite partition surface, so a consumer receives cells, adjacency, and endpoint coverage rather than a completed function space.

Use the finite sum rows SL and SU together with the gap row G. These rows compute lower and upper Darboux sums over the same displayed cells and route the comparison through the $\mathsf{DarbouxSum}^{\uparrow}$ boundary.

The readback and seal rows R and E then attach the finite observations to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after the partition and Darboux rows are visible. The structural rows H, C, Q, N preserve this order, so no Riemann or Real-facing consumer can bypass the finite partition route or import a refused host principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxSums}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxSums}^{\uparrow}$ packet is generated from a RealUp interval source, a finite PartitionUp cell row, displayed lower and upper value ledgers, lower and upper finite sum rows, a DarbouxSum gap comparison, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxSums}^{\uparrow}}$, seed)

Scope closed: Definition 6436.1, Definition 6436.2, and Definition 6436.3 seed the finite Darboux partition-sum route over RealUp, RegSeqRatUp, DyadicRatCoreUp, PartitionUp, DarbouxSumUp, and RiemannIntegralUp rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DarbouxSums}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient functions, host suprema, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite partition coverage, lower and upper value stability, finite-sum exactness, Darboux gap readback, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxSumsUp.

Chapter 6437

A Concrete Naming Certificate for HyperbolicPhaseCurvatureBudget$^{\uparrow}$

The HyperbolicPhaseCurvatureBudget$^{\uparrow}$ packet realizes the curvature-budget branch of chapter 9463 as finite BEDC NameCert data. It sits after the disk carrier of chapter 377, the Poincare disk metric packet of chapter 6209, the hyperbolic metric rows of chapter 3474, the observer-expansion route of chapter 4167, and the geodesic-flow surface of chapter 3856. The packet names disk, metric, boundary, Real-seal, provenance, and refusal rows without importing host global geometry, a completed boundary, a physical observer law, or an unlogged curvature scalar.

Definition 6437.1 (Hyperbolic phase curvature budget carrier)**.** *A* HyperbolicPhaseCurvatureBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, M, K, O, G, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the displayed disk row, read through* UnitDisk$^{\uparrow}$ *and the Poincare disk coordinate surface.*
2. *M is the* PoincareDiskMetric$^{\uparrow}$ *or* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same disk observations.*
3. *K is the finite curvature-budget row, recording the local comparison budget used by the observer-expansion read.*
4. *O is the* HyperbolicObserverExpansion$^{\uparrow}$ *route consuming D, M, K as a focused expansion ledger.*
5. *G is the* HyperbolicGeodesicFlow$^{\uparrow}$ *or boundary-flow handoff row attached to the same metric budget.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for metric and curvature observations.*
7. *E is the terminal* Real$^{\uparrow}$ *curvature readback seal reached only after D, M, K, O, G, R have been displayed.*
8. *F is the TasteGate-facing row requiring Lean* `Nontrivial` *and* `FieldFaithful` *instances for the same carrier fields.*
9. *H records componentwise* hsame *transport for disk, metric, curvature-budget, observer-expansion, geodesic-flow, readback, Real-seal, TasteGate-facing, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through the route*

$$D, M \longmapsto K \longmapsto O, G \longmapsto R, E.$$

11. *P is the* Pkg *provenance over* HyperbolicPhaseCurvatureBudget$^{\uparrow}$, UnitDisk$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{HyperbolicPhaseCurvatureBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, K, O, G, R, E, F, H, C, P, N$. It has no coordinate for a host hyperbolic manifold, quotient boundary equality, unlogged curvature scalar, selected global geodesic, physical observer law, ambient real completeness, Lean verification, or bridge checking.

Theorem 6437.2 (Hyperbolic phase curvature budget obligations)**.** *For every accepted* HyperbolicPhaseCurvatureB *carrier $B = (D, M, K, O, G, R, E, F, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicPhaseCurvatureBudget}^{\uparrow}}$ *surface consists exactly of disk admission, metric-read exposure, curvature-budget admission, observer-expansion routing, geodesic-flow handoff, regular rational readback, Real-seal nonescape, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A Real-facing curvature readback must expose the* HyperbolicMetric$^{\uparrow}$ *row before it reads E.*

Proof. Project the carrier of Definition 6437.1. The disk row D and metric row M are upstream of every curvature-budget, observer-expansion, geodesic-flow, readback, and Real-seal row.

Use the curvature-budget row K and observer-expansion row O. They bind the focused expansion reading to the displayed metric budget rather than to an ambient global geometry.

The geodesic-flow handoff G and readback row R preserve the same metric-budget route before the Real seal E is visible. The TasteGate-facing row F records that Lean nontriviality and field faithfulness must keep these fields distinct.

Rows H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores host geometry, quotient boundary equality, selected global geodesics, physical observer dynamics, or real completeness, the listed obligation surface is exhaustive. □

Theorem 6437.3 (Hyperbolic phase curvature Real-readback order)**.** *Every accepted* HyperbolicPhaseCurvatureBud *consumer that exports a* Real$^{\uparrow}$ *curvature readback must first display the route*

UnitDisk$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$

In carrier notation, the row E is reachable only after $D, M, K, O, G, R, H, C, P, N$ have been read, and the TasteGate-facing row F keeps the metric and curvature-budget fields noncollapsed.

Proof. Begin with the disk and metric rows. The curvature readback has no source until D and M expose the displayed disk and hyperbolic metric observations.

Read the budget and expansion rows next. The row K records the finite curvature budget, while O and G attach it to observer expansion and geodesic-flow handoff over the same metric route.

Only then can R and E be consumed. The RegSeqRat$^{\uparrow}$ row records the regular rational readback, and the terminal Real$^{\uparrow}$ seal is downstream of that readback. The rows F, H, C, P, N preserve field separation, transport, replay, provenance, and local naming without adding another Real-facing path. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}$ packet exposes a $\mathsf{HyperbolicMetric}^{\uparrow}$ row before any $\mathsf{Real}^{\uparrow}$ curvature readback; omitting that metric row refutes the carrier because the curvature budget can no longer be distinguished from an unlogged Real scalar.

Independence witness: hyperbolic-metric, poincare-disk-metric, hyperbolic-observer-expansion, hyperbolic-geodesic-flow: no listed sibling is bijective to $\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}$ because this packet jointly carries disk observations, metric reads, curvature-budget rows, observer-expansion routing, geodesic-flow handoff, regular readback, Real sealing, TasteGate-facing field separation, transport, replay, provenance, refusal rows, and local naming.

The Lean TasteGate file `lean4/BEDC/Derived/HyperbolicPhaseCurvatureBudgetUp/TasteGate.lean` must provide `Nontrivial` and `FieldFaithful` instances for $\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}$. Those instances must preserve $D, M, K, O, G, R, E, F, H, C, P, N$ as distinct fields before any carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}$ packet is generated from UnitDiskUp disk data, PoincareDiskMetricUp and HyperbolicMetricUp metric rows, a finite curvature-budget row, HyperbolicObserverExpansionUp routing, HyperbolicGeodesicFlowUp handoff, RegSeqRatUp readback, a RealUp seal, TasteGate-facing field-separation requirements, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6437.1, Definition 6437.2, and Definition 6437.3 seed the finite curvature-budget route over disk, metric, boundary, observer-expansion, geodesic-flow, RegSeqRat, Real-seal, provenance, and refusal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host global hyperbolic geometry, quotient boundary equality, selected global geodesics, physical observer dynamics, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, disk exposure, metric-read exposure, curvature-budget exactness, observer-expansion routing, geodesic-flow handoff, Real-seal nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseCurvatureBudgetUp.

Chapter 6438

A Concrete Naming Certificate for $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$

Following Alexandrov–Urysohn metrization, the $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$ packet records only the finite BEDC certificate by which countable basis windows and separation witnesses synthesize a metric-radius readback. It sits after the open-row discipline of chapter 88, beside the metric carrier of chapter 57, the finite-window schedule of chapter 126, the regular-rational readback of chapter 301, and the complete separable metric boundary of chapter 1490. The packet names a displayed basis-to-metric route; it does not assert global choice, arbitrary open-cover metrization, quotient topology, selected embeddings, or host-level general metrization.

Definition 6438.1 (Alexandrov–Urysohn metrization carrier)**.** *An* $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, B, S, R, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, recording the finite open-membership observations available to the packet.*
2. *B is the countable basis-window row, recording a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of basis names, incidence witnesses, and window bounds.*
3. *S is the separation-witness row, recording the displayed point-open and disjoint-closure tests consumed by the metric synthesis.*
4. *R is the metric-radius synthesis row, assigning dyadic rational radius ledgers to the basis windows and separation witnesses.*
5. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular-rational distance observations produced from R.*
6. *Q is the refusal row, recording that global choice, arbitrary open covers, quotient topology, selected embeddings, and host-level metrization are absent from the packet.*
7. *H records componentwise* hsame *transport for topology, basis-window, separation, radius synthesis, readback, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from* $\mathsf{Topology}^{\uparrow}$ *and basis windows through metric-radius synthesis into* $\mathsf{Met}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *and* $\mathsf{PolishSpace}^{\uparrow}$ *consumers.*

9. *P is the* Pkg *provenance over* AlexandrovUrysohnMetrization$^{\uparrow}$, Topology$^{\uparrow}$, Met$^{\uparrow}$, Met$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, PolishSpace$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{AlexandrovUrysohnMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected global basis, an arbitrary open-cover refinement theorem, a quotient topology, a host embedding into sequence space, or a classical all-spaces metrization principle.

Theorem 6438.2 (Alexandrov–Urysohn metrization NameCert obligations)**.** *For every accepted* AlexandrovUrysohnMetrization$^{\uparrow}$ *carrier $A = (T, B, S, R, W, Q, H, C, P, N)$, the local* NameCert$_{\text{AlexandrovUrysohnMetrizatio}}$ *surface consists exactly of topology-source admission through T, countable basis-window scheduling through B, separation-witness inspection through S, metric-radius synthesis through R,* RegSeqRat$^{\uparrow}$ *readback through W, refusal of non-displayed classical data through Q, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6438.1. The topology source, basis windows, separation witnesses, radius ledger, readback, refusal, transport, replay, provenance, and naming rows are exactly $T, B, S, R, W, Q, H, C, P, N$.

Read the basis data through B before metric synthesis. The finite StreamName$^{\uparrow}$ schedule exposes only displayed basis windows and incidence witnesses, so no global basis choice or arbitrary open-cover selection is available.

Next read the separation witnesses through S and the radius ledger through R. The metric-facing observation is synthesized from those displayed rows and then exposed as the RegSeqRat$^{\uparrow}$ readback W before any Met$^{\uparrow}$, Met$^{\uparrow}$, or PolishSpace$^{\uparrow}$ consumer can inspect it.

Finally Q, H, C, P, N refuse absent classical data, preserve the displayed rows, replay the route, source the packet, and name it locally. Since the carrier contains no quotient-topology coordinate, selected embedding, or host-level general metrization theorem, the NameCert obligations are exactly the finite basis-to-metric route stated above. □

Theorem 6438.3 (Alexandrov–Urysohn finite basis-to-metric route)**.** *Every* PolishSpace$^{\uparrow}$ *or* Met$^{\uparrow}$ *consumer of an* AlexandrovUrysohnMetrization$^{\uparrow}$ *packet must first read the* Topology$^{\uparrow}$ *row, finite basis-window schedule, separation-witness row, metric-radius synthesis ledger,* StreamName$^{\uparrow}$ *schedule,* RegSeqRat$^{\uparrow}$ *readback,* BHist *carrier, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives a finite certificate for the displayed basis-to-metric handoff, not a quotient topology, arbitrary open-cover theorem, selected global basis, or host-level general metrization result.*

Proof. Start with the NameCert obligations of Definition 6438.2. They force topology admission, basis-window scheduling, separation witnesses, radius synthesis, readback, refusal, transport, replay, provenance, and naming to remain rows of one finite packet.

The metric consumer then reads R and W after T, B, S. Thus the distance observations are regular-rational readbacks from displayed basis and separation data, not host real distances selected from an external metric.

For a PolishSpace consumer, replay the same metric-facing row through C and provenance P. Completeness and separability consumers may inspect the handoff only as a routed finite NameCert surface over the sibling Met$^{\uparrow}$/Met$^{\uparrow}$, StreamName$^{\uparrow}$, and RegSeqRat$^{\uparrow}$ rows.

The refusal row Q records the boundary. No global choice, arbitrary open-cover compactness, quotient topology, selected embedding, or host-level metrization theorem is exported by this packet. □

Falsifiable prediction: An accepted $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$ packet cannot export a MetricSpace or PolishSpace handoff without displaying the topology source, basis-window schedule, separation witnesses, radius synthesis, RegSeqRat readback, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: topology, metric, streamname, regseqrat, polishspace: $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$ binds countable basis windows, separation witnesses, and metric-radius synthesis in one finite carrier; the cited siblings expose the source, readback, and consumer rows but not the local basis-to-metric certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}}$.

Constructive story: An $\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}$ packet is generated from a TopologyUp source row, finite countable-basis windows, separation witnesses, metric-radius synthesis, RegSeqRat readback, refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6438.1, Definition 6438.2, and Definition 6438.3 seed the finite basis-to-metric certificate over topology, stream, regular-rational, metric, and Polish-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandrovUrysohnMetrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, global choice, arbitrary open-cover metrization, quotient topology, selected embeddings, or host-level general metrization.

Upgrade path: Formal closure requires a checked carrier and NameCert obligation theorem for the displayed basis-window, separation, radius-synthesis, readback, transport, replay, provenance, and naming rows.

Chapter 6439

A Concrete Naming Certificate for $\mathsf{CauchySelectionPrinciple}^{\uparrow}$

Following Bishop-style constructive analysis, the $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ packet records the finite selection-window route used by Real-completion consumers before any selected subsequence, quotient real, or countable-choice schedule is available. It sits after the dyadic tolerance ledger of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the equality-facing comparison surface of chapter 1820. The packet names the bounded choice-free handoff needed by Cauchy subsequence arguments, not a global choice principle and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6439.1 (Cauchy selection principle carrier)**.** *A* $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested precision, the finite dyadic comparison window, and the descent budget used by the selection read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the bounded stream observations inspected by the selection request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic ledger, stream windows, and regular readback have been displayed.*
5. *Q is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row used to compare two accepted selection reads through the same displayed tail window.*
6. *H records componentwise* hsame *transport for dyadic tolerances, stream windows, regular readback, Real seal, Cauchy-equivalence comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-facing precision request through D, W, R, E, Q, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySelectionPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySelectionPrinciple}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $D, W, R, E, Q, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, host real equality, countable choice, trichotomy, metric completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySelectionPrincipleUp.CauchySelectionPrincipleTasteGate single carrier alignment`

Theorem 6439.2 (Cauchy selection subsequence window)**.** *Every subsequence-facing read of an accepted* $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ *carrier* $S = (D, W, R, E, Q, H, C, P, N)$ *is confined to the displayed finite window* D, W, R*. The read may inspect a dyadic precision request, a bounded StreamName observation family, and the attached RegSeqRat readback, but it receives no selected infinite subsequence and no countable-choice schedule.*

Proof. Project the accepted carrier. The only value-facing rows before the Real seal are the dyadic ledger D, the stream-window row W, and the regular rational readback R.

The continuation row C fixes the order of inspection. A precision request first reads the dyadic budget, then the bounded stream observations, and then the regular rational readback attached to those observations.

The structural rows H, P, N preserve the same displayed route, and Q is only a comparison row for accepted reads. Since no row stores a selected infinite subsequence or a choice schedule, every subsequence-facing read remains inside the finite window. □

Theorem 6439.3 (Cauchy selection regularity handoff)**.** *For every accepted* $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ *carrier* $S = (D, W, R, E, Q, H, C, P, N)$*, the selection window reaches a completion-facing name only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *after the dyadic tolerance ledger* D *and the StreamName window* W *have been exposed. The handoff is therefore a regular rational readback, not a quotient stream and not a selected representative.*

Proof. Begin with the displayed rows D and W. They bind the finite precision request and the bounded observations from which the selection read is made.

The row R consumes exactly those displayed rows as a $\mathsf{RegSeqRat}^{\uparrow}$ readback. Thus the completion-facing handoff is already regular-rational before the Real seal or comparison row can be used.

Rows E, Q, H, C, P, N do not add another value source. They seal, compare, transport, replay, source, and name the same handoff, so the route cannot introduce quotient streams, selected representatives, or countable choice. □

Theorem 6439.4 (Cauchy selection Real-seal nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The packet exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient equality of stream names, host real equality, trichotomy, selected limit, selected subsequence, metric completeness, open-cover compactness, or countable-choice row.*

Proof. Project $S = (D, W, R, E, Q, H, C, P, N)$. The Real seal E is present only as the fourth value-facing row after the dyadic tolerance ledger, finite stream windows, and regular rational readback.

Use Definition 6439.3. It shows that any completion-facing name read has already passed through D, W, R, so the terminal Real seal cannot be detached from the finite selection window.

The comparison and structural rows preserve the same path. Because the carrier has no coordinate for completeness, quotient equality, host equality, trichotomy, selected limits, selected subsequences, metric completeness, open-cover compactness, or countable choice, the Real-facing consumer receives only the displayed finite NameCert route. □

Falsifiable prediction: Every $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ Real-completion read factors through one DyadicRatCore tolerance ledger, one StreamName finite-window row, one RegSeqRat readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, one CauchyEquivalenceSetoid comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-equivalence-setoid: $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite selection window, regularity handoff, Real seal, comparison, transport, replay, provenance, and local NameCert rows as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySelectionPrinciple}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CauchySelectionPrinciple}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, CauchyEquivalenceSetoid comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySelectionPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 6439.1, Definition 6439.2, Definition 6439.3, and Definition 6439.4 seed the finite selection-window route from dyadic precision through StreamName, RegSeqRat, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySelectionPrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected subsequences, selected limits, quotient stream equality, host real equality, trichotomy, countable choice, metric completeness, open-cover compactness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selection-window exactness, regularity handoff, Real-seal nonescape, Cauchy-equivalence comparison, transport, replay, provenance, and local naming for BEDC.Derived.CauchySelectionPrinc

Chapter 6440

A Concrete Naming Certificate for $\mathsf{RealCompletionSelectionGate}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ packet records the finite gate that separates allowed Real-completion reads from off-table completeness claims. It realizes the four-object route of chapter 9238 as displayed rows over chapter 291, chapter 126, chapter 301, and chapter 38, with comparison discipline inherited from chapter 1820. The gate permits finite selection windows and regular rational readback; it does not certify quotient completion, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6440.1 (Real completion selection gate carrier)**.** *A* $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger for the requested Real-completion precision.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row, storing only the bounded windows read by the gate.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, W, R have been displayed.*
5. *T records componentwise* hsame *transport for the four value-facing rows and for the replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a Real-completion request through D, W, R, E, T, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealCompletionSelectionGate}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectionGate}^{\uparrow}}$ *row.*

The classifier compares accepted gates only through D, W, R, E, T, C, P, N. It has no coordinate for quotient stream equality, selected representatives, global Cauchy choice, host real equality, trichotomy, metric completeness, open-cover compactness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6440.2 (Real completion selection gate four-object route)**.** *Every accepted* $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ *carrier routes an allowed Real-completion read through the ordered four-object chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The gate exposes no completion-facing value before the dyadic tolerance ledger, finite StreamName window, regular rational readback, and terminal Real seal have all been displayed.

Proof. Project the accepted gate $G = (D, W, R, E, T, C, P, N)$. The value-facing rows are exactly D, W, R, E.

The continuation row C fixes the reading order. A Real-completion request first consumes the dyadic precision ledger, then the bounded stream observations, then the regular rational readback, and only then the terminal Real seal.

Rows T, P, N transport, source, and name the same route. Since no other value-facing row exists in the carrier, every allowed completion read factors through the four-object chain. □

Theorem 6440.3 (Real completion selection gate transport replay)**.** *For every accepted* RealCompletionSelectionGa *carrier* $G = (D, W, R, E, T, C, P, N)$*, componentwise transport and displayed replay preserve the same four-object route. Transport cannot replace the finite StreamName window by a quotient stream, and replay cannot introduce a selected limit or countable-choice schedule.*

Proof. The transport row T is componentwise over D, W, R, E, C, P, N. It can identify two displayed gates only by transporting the same dyadic ledger, finite window, regular readback, Real seal, replay, provenance, and local name.

The replay row C records the route from a Real-completion request through those same displayed rows. Thus replay reuses the already named four-object chain rather than adding a second completion source.

Because neither T nor C has a coordinate for quotient streams, selected representatives, selected limits, or choice schedules, the transported and replayed gate remains inside the finite route of Definition 6440.2. □

Theorem 6440.4 (Real completion selection gate nonescape)**.** *Every consumer of an accepted* $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ *packet receives only dyadic tolerance data, finite StreamName observations, RegSeqRat readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. The packet exports no quotient completion, no selected representative, no selected limit, no host real equality, no trichotomy, no countable choice, no metric completeness theorem, no open-cover compactness theorem, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Use Definition 6440.1. The accepted packet is exhausted by D, W, R, E, T, C, P, N.

The value-facing rows give the four-object completion route, and the structural rows preserve that route under transport, replay, provenance, and local naming. None of these rows stores a detached completion object or a global selection principle.

Therefore every consumer remains inside the finite gate. Any claim requiring quotient completion, selected representatives, selected limits, host equality, trichotomy, countable choice, metric completeness, open-cover compactness, or ambient Real completeness asks for a row absent from the carrier. □

Falsifiable prediction: Every $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ Real-completion read factors through one DyadicRatCore tolerance ledger, one StreamName finite-window row, one RegSeqRat readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-selection-principle: $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ is not bijective to any listed sibling because it packages the four-object route as a gate with transport, replay, provenance, and local NameCert rows that reject off-table completion claims.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionSelectionGate}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionSelectionGate}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionSelectionGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6440.1, Definition 6440.2, Definition 6440.3, and Definition 6440.4 seed the four-object Real-completion gate and its nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionSelectionGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, host real equality, trichotomy, countable choice, metric completeness, open-cover compactness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-object route exactness, transport replay, nonescape, provenance, and local naming for BEDC.Derived.RealCompletionSelectionGateUp.

Chapter 6441

A Concrete Naming Certificate for BolzanoDedekindLocatedCluster$^{\uparrow}$

Following Bolzano, Dedekind, and Bishop–Bridges constructive analysis, the BolzanoDedekindLocatedCluster$^{\uparrow}$ packet records a finite located-cluster surface for bounded real sequences. It consumes the bounded sequence source of chapter 1415, the nested and located interval routes of chapter 320 and chapter 394, the finite stream and regular-rational rows of chapter 126 and chapter 301, the terminal real-name seal of chapter 38, and the boundary siblings chapter 1254, chapter 1580, and chapter 2372. The packet is a located finite-window cluster certificate; it is not general sequential compactness, not a choice-selected subsequence theorem, and not an ambient completeness principle for Real$^{\uparrow}$.

Definition 6441.1 (Bolzano–Dedekind located cluster carrier)**.** *A* BolzanoDedekindLocatedCluster$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, I, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* BoundedRealSequence$^{\uparrow}$ *source row, including only the inspected finite sequence windows and the displayed bound surface.*
2. *I is the nested interval row, recording the retained finite interval chain used by the Bolzano–Dedekind search.*
3. *L is the located interval row, carrying endpoint-apartness and retained-window location data for the intervals in I.*
4. *S is the* StreamName$^{\uparrow}$ *schedule that names which sequence windows and interval windows are inspected.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the scheduled sequence and interval windows.*
6. *E is the* Real$^{\uparrow}$ *cluster seal reached only after B, I, L, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the bounded source, nested interval, located interval, stream, regular readback, and real-seal rows.*
8. *C records displayed* Cont *replay from a cluster request through the same finite rows.*

9. *P is the* Pkg *provenance over* BolzanoDedekindLocatedCluster$^{\uparrow}$, BoundedRealSequence$^{\uparrow}$, NestedInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BolzanoWeierstrass$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BolzanoDedekindLocatedCluster}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, I, L, S, R, E, H, C, P, N$. It has no coordinate for a globally selected subsequence, open-cover compactness, quotient real equality, real trichotomy, countable choice, general sequential compactness, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6441.2 (Bolzano–Dedekind located cluster NameCert obligations)**.** *For every accepted* BolzanoDedekindLocatedCluster$^{\uparrow}$ *carrier*

$$K = (B, I, L, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{BolzanoDedekindLocatedCluster}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, bounded-sequence source exposure through B, nested-window retention through I, located endpoint and interval comparison through L, finite stream scheduling through S, regular rational readback through R, terminal cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. The obligation surface refuses general sequential compactness, choice-selected subsequences, open-cover compactness, quotient real equality, trichotomy, and ambient Real completeness as certificate rows.*

Proof. Project the carrier of Definition 6441.1. The bounded sequence can be read only through B, and B itself supplies finite windows and a displayed bound surface rather than a global compactness theorem.

Next inspect the interval rows. The nested interval row I records the retained finite chain, while the located interval row L records the endpoint and apartness comparisons needed to keep the cluster window located. These rows are packet data, not consequences of an ambient interval-intersection theorem.

The readback and seal are downstream. The schedule row S names the inspected windows, R supplies the regular rational readback for those windows, and E is visible only after those finite rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded compactness, choice, quotient, trichotomy, or completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6441.3 (Bolzano–Dedekind located cluster Real handoff)**.** *Every* Real$^{\uparrow}$*-facing read of an accepted* BolzanoDedekindLocatedCluster$^{\uparrow}$ *carrier factors through the finite route*

$$B,\ I,\ L,\ S,\ R,\ E,\ H,\ C,\ P,\ N.$$

The handoff supplies a located cluster seal for bounded real sequence consumers and for the Bolzano–Weierstrass boundary siblings, but it does not supply general sequential compactness, a choice-selected subsequence, open-cover compactness, quotient real equality, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Start with the bounded source row. By Definition 6441.1, the sequence source is the displayed BoundedRealSequence$^{\uparrow}$ row B, so any Real-facing consumer first reads finite sequence windows and the boundedness surface already present in B.

Use the NameCert obligations of Definition 6441.2. They place nested-window retention I and located interval comparison L before the stream schedule, regular rational readback, and terminal cluster seal.

Now read the seal row E. It is downstream of S and R, so the cluster handoff reaches $\mathsf{Real}^{\uparrow}$ through finite stream and regular-rational data rather than through quotient real equality, trichotomy, or ambient completeness.

The structural rows H, C, P, N preserve the same displayed route. Thus a Bolzano–Weierstrass or cluster-point consumer receives a located finite-window cluster seal, while the excluded compactness and choice principles remain outside the packet. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ boundary read that cites $\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}$ must display a $\mathsf{BoundedRealSequence}^{\uparrow}$ source row, nested and located interval rows, StreamName windows, RegSeqRat readback, a RealUp seal, transport, replay, provenance, and local naming before any cluster consumer can proceed.

Independence witness: boundedrealsequence, nestedinterval, locatedinterval, streamname, regseqrat, real, bolzanoweierstrass: $\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded source exposure, nested-window retention, located interval comparison, stream scheduling, regular readback, terminal cluster sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}$ packet is generated from a bounded real sequence row, nested and located interval rows, StreamName windows, RegSeqRat readback, a terminal RealUp cluster seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6441.1, Definition 6441.2, and Definition 6441.3 seed the finite located-cluster certificate for bounded real sequence windows through nested and located interval rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoDedekindLocatedCluster}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general sequential compactness, choice-selected subsequences, open-cover compactness, quotient real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exposure, nested-window retention, located interval comparison, stream scheduling, regular rational readback, real-seal handoff, transport, replay, provenance, and local naming for the same packet.

Chapter 6442

A Concrete Naming Certificate for $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ packet records the finite criterion by which tight Cauchy evidence is read as regular Cauchy data before the completion route consumes a real-name seal. It sits between the tightness surface of chapter 1870, the regularizing route of chapter 2227, the modulus schedule of chapter 292, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic of chapter 291, and the visible real-name boundary of chapter 38. The packet is a finite NameCert surface for the textbook criterion; it is not a quotient completion, a selected limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6442.1 (Bishop tight Cauchy criterion carrier)**.** *A* $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, G, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyTightness}^{\uparrow}$ *evidence row, recording tight-tail comparisons for the inspected sequence data.*
2. *G is the* $\mathsf{RegularCauchyTightening}^{\uparrow}$ *handoff row that turns tight-tail evidence into regular Cauchy readback data.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule attached to the same tail comparisons.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which every tail and threshold read is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row assembled from the scheduled windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the tight-tail and regular-readback comparisons.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, G, M, W, R, D have all been displayed.*
8. *H stores componentwise* hsame *transport for tightness evidence, regular tightening, modulus schedules, stream windows, rational readbacks, dyadic ledgers, and the real seal.*
9. *C stores displayed* Cont *replay for the same route.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyTightness}^{\uparrow}$, $\mathsf{RegularCauchyTightening}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopTightCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $T, G, M, W, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limit representatives, countable choice, trichotomy on real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6442.2 (Bishop tight Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ *carrier $B = (T, G, M, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopTightCauchyCriterion}}$ *obligations are exactly carrier admission, tight-tail stability, regular-readback transport,* $\mathsf{Real}^{\uparrow}$*-seal nonescape, and completion-consumer handoff. Each obligation is a finite BEDC row over the displayed tightness, tightening, modulus, stream, regular-rational, dyadic, real-seal, transport, replay, provenance, and naming coordinates.*

Proof. Carrier admission is projection from Definition 6442.1. The packet displays the tightness row T, the regularizing row G, the modulus row M, the window row W, the readback row R, the dyadic row D, the seal row E, and the structural rows H, C, P, N.

Tight-tail stability is read from T together with M, W, D. A tolerance request first enters the dyadic ledger, is matched to a finite stream window, and then receives the tail threshold recorded by the modulus schedule. Transport by H preserves the same displayed rows, and replay by C repeats the same finite route.

Regular-readback transport uses G and R. The tightening row converts the stable tight-tail data into the regular Cauchy surface, and the regular rational row records the finite readback before any real-facing seal is available.

The $\mathsf{Real}^{\uparrow}$ seal is nonescaping because E is downstream of T, G, M, W, R, D. No row in the carrier can supply a host limit, quotient equality, selected representative, or ambient completeness principle before that finite route has been displayed.

Finally, a completion consumer receives only the packet named by P, N. The consumer can hand off the tight Cauchy criterion to the regular Cauchy and Real route, but it cannot bypass the displayed finite rows or import an unlisted completion principle. □

Falsifiable prediction: Every $\mathsf{Completion}^{\uparrow}$ consumer of a $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ packet factors through tight-tail evidence, regular tightening, Cauchy-modulus thresholds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, the RealUp seal, transport, replay, provenance, and local naming before completion data is consumed.

Independence witness: cauchytightness, regularcauchytightening, cauchymodulus, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries tight-tail evidence, the regularizing handoff, the threshold schedule, finite windows, regular rational readback, dyadic tolerance rows, a real seal, componentwise transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopTightCauchyCriterion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{BishopTightCauchyCriterion}^{\uparrow}$ packet is generated from tight Cauchy evidence, a regular Cauchy tightening row, a CauchyModulus threshold schedule, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance rows, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopTightCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6442.1 and Definition 6442.2 seed the finite Bishop tight Cauchy criterion between tight Cauchy evidence and regular Cauchy readback.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopTightCauchyCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy on real names, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tight-tail stability, regular-readback transport, RealUp seal nonescape, and completion-consumer handoff for BEDC.Derived.BishopTightCauchyCriterionUp.

Chapter 6443

A Concrete Naming Certificate for BishopFiniteIntervalChain$^{\uparrow}$

Following Bishop and Bridges, the BishopFiniteIntervalChain$^{\uparrow}$ packet records the finite interval-chain carrier used when compact interval arguments reduce a cover request to adjacent endpoint checks and refinement over a displayed list. It sits after the dyadic interval arithmetic of chapter 1645, the real interval enclosure rows of chapter 1625, and the compact-cover Lebesgue ledger of chapter 1302. The packet is a finite NameCert surface for interval-chain reasoning; it is not open-cover compactness, a selected subcover, countable choice, quotient real equality, or ambient completeness of Real$^{\uparrow}$.

Definition 6443.1 (Bishop finite interval chain carrier)**.** *A* BishopFiniteIntervalChain$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, A, R, M, I, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint-interval row. It records the finite list of located endpoint intervals, their dyadic endpoint readbacks, and the real interval-enclosure seals inspected by the chain.*
2. *A is the adjacency and overlap row. It records, for each consecutive pair, the displayed nonempty overlap or endpoint contact certificate that lets a local cover row pass to the next interval.*
3. *R is the refinement-map row. It records the finite monotone map from a coarse interval chain to a finer chain and the endpoint containment certificates for every refined cell.*
4. *M is the cover-membership row. It records the named cover packet, the interval cell named by each cover row, and the finite witness that every inspected point of the chain lies in one displayed member.*
5. *I is the finite induction row. It records the base cell, the successor transport along adjacency, and the terminal interval claim produced by primitive recursion over the chain.*
6. *H stores componentwise* hsame *transport for endpoint intervals, adjacency and overlap rows, refinement maps, cover membership, finite induction, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay for the same interval-chain route.*
8. *P is the* Pkg *provenance over* BishopFiniteIntervalChain$^{\uparrow}$, DyadicIntervalArithmetic$^{\uparrow}$, RealIntervalEnclosure$^{\uparrow}$, CompactCoverLebesgueLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopFiniteIntervalChain}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed tuple $E, A, R, M, I, H, C, P, N$. It has no coordinate for an external open-cover compactness theorem, a selected subcover, countable choice, quotient real equality, trichotomy on real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6443.2 (Bishop finite interval chain NameCert obligations)**.** *For every accepted* BishopFiniteIntervalChain *carrier $Q = (E, A, R, M, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopFiniteIntervalChain}^{\uparrow}}$ *obligations are exactly carrier admission, endpoint-interval stability, adjacency and overlap transport, refinement-map exactness, cover-membership locality, finite induction replay, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6443.1. The accepted packet displays endpoint intervals, adjacency and overlap certificates, refinement maps, cover-membership rows, finite induction data, and the structural rows H, C, P, N.

Endpoint-interval stability is read from E together with the dyadic and real interval enclosure rows named in P. Transport by H preserves the same endpoint readbacks and enclosure seals, so no quotient equality or external real-completeness principle is introduced.

Adjacency and refinement obligations are carried by A and R. Each successor step has a displayed overlap or endpoint-contact certificate, and each refined cell has a displayed containment certificate in its coarse ancestor. These rows make the chain handoff finite.

Cover locality is supplied by M, while finite induction is supplied by I. The base interval opens the first cover row, the successor clause moves across the displayed adjacency certificate, and primitive recursion over the finite chain reaches the terminal interval claim.

Finally C, P, N replay, source, and name the same finite route. Since the carrier has no further coordinate, every local NameCert obligation is one of the listed rows. □

Theorem 6443.3 (Bishop finite interval chain refinement induction)**.** *Every accepted* $\mathsf{BishopFiniteIntervalChain}^{\uparrow}$ *refinement request factors through the finite route*

$$E, A \longmapsto R, M \longmapsto I, H, C, P, N.$$

Thus any interval-chain cover argument that consumes the packet must inspect endpoint intervals and adjacent overlaps before it can replay refinement, cover membership, and induction over the chain.

Proof. Begin with an accepted carrier $Q = (E, A, R, M, I, H, C, P, N)$. The source stage is E, A: the endpoint row fixes the finite interval list, and the adjacency row records the overlap or endpoint-contact certificates for consecutive cells.

The refinement stage is R, M. The refinement map can only split a displayed cell into finitely many contained subcells, and the cover-membership row records which named cover packet accepts each inspected cell.

The induction stage is I. Its base clause opens the first interval row, while its successor clause transports the local cover statement across the displayed adjacency or overlap certificate. Primitive recursion over the finite chain therefore supplies the terminal claim without selecting a subcover outside the packet.

The final rows H, C, P, N transport, replay, source, and name the same data. The route cannot produce open-cover compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness because none of those objects appears in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFiniteIntervalChain}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFiniteIntervalChain}^{\uparrow}$ packet is generated from endpoint interval rows, adjacency and overlap certificates, refinement maps, cover-membership rows, finite induction over the chain, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFiniteIntervalChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6443.1, Definition 6443.2, and Definition 6443.3 seed the finite interval-chain route used by compact interval and finite-cover arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFiniteIntervalChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient real equality, trichotomy on real names, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-interval stability, adjacency and overlap transport, refinement-map exactness, cover-membership locality, finite induction replay, transport, replay, provenance, and local naming for BEDC.Derived.BishopFiniteIntervalChainUp.

Chapter 6444

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusBridge}^{\uparrow}$

Following Bishop and Bridges compact metric analysis, the $\mathsf{CompactUniformModulusBridge}^{\uparrow}$ packet records the bridge from compact finite-net data and pointwise continuity rows to the $\mathsf{UniformModulus}^{\uparrow}$ output surface named in chapter 9237. It consumes the compact source of chapter 60, the continuity rows of chapter 59, the rational fold interface of chapter 619, the uniform-modulus packet of chapter 369, and the existing compact uniform modulus carrier of chapter 2299. The packet names the bridge ledger itself; it is not a selected Lebesgue number, a host modulus function, sequential compactness, countable choice, or ambient completeness of the metric carrier.

Definition 6444.1 (Compact uniform modulus bridge carrier)**.** *A* $\mathsf{CompactUniformModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, K, F, T, P, R, L, U, H, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row shared by the compact and continuous certificates.*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, including the displayed finite* $\mathsf{ProbeBundle}$ *of centers and coverage radii.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row over the same metric source.*
4. *T is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance request read by the downstream uniform-modulus consumer.*
5. *P is the finite probe-classifier row that lists the center names, local radii, and pointwise-modulus certificates opened by K and F.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the finite radius ledger carried by P.*
7. *L is the bridge ledger joining X, K, F, T, P, R to the terminal* $\mathsf{UniformModulus}^{\uparrow}$ *row.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row exposed only after the bridge ledger has replayed the finite fold and the metric triangle estimates.*
9. *H stores componentwise* hsame *transport for the metric source, compact finite net, continuity row, rational tolerance, probe classifier, rational fold, bridge ledger, uniform-modulus output, replay, provenance, and local naming.*
10. *C stores displayed* Cont *replay for the same bridge route.*

11. *G is the* Pkg *provenance over* $\mathsf{CompactUniformModulusBridge}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, T, P, R, L, U, H, C, G, N$. It has no coordinate for selected subcovers, selected Lebesgue numbers, host minima, sequence extraction, countable choice, quotient metric equality, or ambient completeness.

Theorem 6444.2 (Compact uniform modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusBridge}^{\uparrow}$ *carrier $B = (X, K, F, T, P, R, L, U, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformMo}}$ *obligations are exactly carrier admission, metric-source lock, compact finite-net coverage, pointwise-modulus binding, finite probe classifier admission, rational lower-bound fold exactness, bridge-ledger replay, uniform-modulus output admission, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from selected compactness or completeness principles.*

Proof. Carrier admission is projection from Definition 6444.1. The packet displays the metric row X, compact row K, continuity row F, rational tolerance T, probe classifier P, rational fold R, bridge ledger L, output row U, and structural rows H, C, G, N.

The source lock is carried by X, K, F. The compact row and continuity row are read over the same metric source before any probe classifier is opened, so the bridge cannot mix finite nets and pointwise moduli from different carriers.

Finite coverage and pointwise-modulus binding are carried by K, F, P. Each bundled center has a displayed coverage row and a displayed pointwise-modulus row, and the classifier only sees those finite entries.

The lower-bound computation is carried by R and replayed by L. Since R is a finite $\mathsf{Rat}^{\uparrow}$ fold over the displayed probe ledger, the bridge has no host minimum selector or selected Lebesgue-number coordinate.

Finally U, H, C, G, N expose, transport, replay, source, and name the same route. Row exhaustion rules out selected subcovers, sequence extraction, choice, quotient metric equality, and ambient completeness. □

Theorem 6444.3 (Compact uniform modulus finite ProbeBundle bridge)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{CompactUniformModulusBridge}^{\uparrow}$ *carrier factors through the finite route*

$$X, K, F, T \longmapsto P, R, L \longmapsto U, H, C, G, N.$$

Thus the bridge first reads the common metric source, compact finite net, continuous pointwise-modulus rows, and rational tolerance; then folds the displayed ProbeBundle *radius ledger; and only then exposes the tolerance-only uniform-modulus output.*

Proof. Begin with an accepted carrier $B = (X, K, F, T, P, R, L, U, H, C, G, N)$. The input stage is X, K, F, T: the common metric source is fixed before the compact finite net, continuity row, and rational tolerance are read.

The finite classifier stage is P, R, L. The probe row lists only finitely many bundled centers and radius entries, the rational row folds those displayed entries, and the bridge ledger records the replay from the finite fold to the output route.

The terminal stage is U, H, C, G, N. The uniform-modulus row is public only after the finite bridge ledger has been displayed, and the structural rows transport and name that same ledger.

Because the carrier has no row for a selected subcover, selected Lebesgue number, host minimum, sequence extraction, countable choice, quotient metric equality, or ambient completeness theorem, every exported read remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusBridge}\uparrow}$.

Constructive story: A $\mathsf{CompactUniformModulusBridge}^{\uparrow}$ packet is generated from a MetricUp source, CompactUp finite-net rows, ContinuousUp pointwise-modulus rows, a RatUp tolerance, a finite ProbeBundle classifier, a RatUp lower-bound fold, a bridge ledger, a UniformModulusUp output, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusBridge}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6444.1, Definition 6444.2, and Definition 6444.3 seed the finite bridge from compact finite-net and pointwise-continuity rows to a uniform-modulus output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusBridge}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected Lebesgue numbers, selected subcovers, sequence extraction, host modulus functions, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source lock, compact finite-net coverage, pointwise-modulus binding, finite probe classifier admission, rational fold exactness, bridge-ledger replay, uniform-modulus output admission, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusBridgeUp.

Chapter 6445

Sobolev trace boundary NameCert construction

Following Adams–Fournier on Sobolev spaces and Evans on PDE boundary traces, the $\mathsf{SobolevTraceBoundary}^{\uparrow}$ packet records finite BEDC boundary trace data rather than a host trace theorem. It sits beside the Sobolev source of chapter 5911, the trace packet of chapter 3647, the embedding packet of chapter 2217, the compact witness of chapter 60, and the complete-metric readback surface of chapter 128.

Definition 6445.1 (Sobolev trace boundary package)**.** *A* $\mathsf{SobolevTraceBoundary}^{\uparrow}$ *package is a finite* BHist *packet*

$$Q = (S, B, T, K, M, X, H, C, P, N)$$

with the following displayed rows:

1. *S is the interior Sobolev carrier row, read from* $\mathsf{SobolevSpace}^{\uparrow}$ *or the Sobolev source row named by chapter 5911.*
2. *B is the boundary carrier row. It records the displayed boundary chart, collar, normal-window, or finite boundary-read ledger on which trace values are inspected.*
3. *T is the trace operator row. It is only the finite trace-read arrow from the interior Sobolev row to the boundary carrier row; it is not an ambient operator on a completed quotient space.*
4. *K is the compact boundary window row, read through chapter 60 and the compact-facing Sobolev neighbour chapter 2217.*
5. *M is the complete-metric readback row, exposing the displayed Cauchy and distance checks supplied by chapter 128.*
6. *X is the classifier row. It distinguishes interior-source reads, boundary-carrier reads, compact-window reads, and complete-metric readbacks, but has no coordinate for a host extension theorem or quotient boundary class.*
7. *H stores componentwise* hsame *transport for S, B, T, K, M, X and the structural rows.*
8. *C stores the displayed* Cont *replay from a boundary-trace request through the interior Sobolev row, the trace row, the compact boundary window, and the complete-metric readback row.*
9. *P is the* Pkg *provenance over* $\mathsf{SobolevTraceBoundary}^{\uparrow}$, $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, $\mathsf{SobolevEmbedding}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SobolevTraceBoundary}^{\uparrow}}$ *row.*

The package exports a finite boundary-readback bridge for consumers of Sobolev trace theory. It does not include a selected extension operator, an unrestricted trace theorem, quotient equality of boundary functions, or ambient compactness.

Lean4 ✓ `BEDC.Derived.SobolevTraceBoundaryUp.SobolevTraceBoundaryTasteGate single carrier alignment`

Theorem 6445.2 (Sobolev trace boundary exactness)**.** *For every accepted* $\mathsf{SobolevTraceBoundary}^{\uparrow}$ *package* $Q = (S, B, T, K, M, X, H, C, P, N)$*, each accepted consumer reads the interior Sobolev row* S *before the trace row* T*. Boundary readback is reachable only through the compact boundary window* K *and the complete-metric row* M*, with* H, C, P, N *preserving and naming that same finite route.*

Proof. Project the accepted package. The carrier order exposes S as the only interior Sobolev source row, so every trace request must enter through S before it can read T.

The boundary row B supplies the displayed boundary carrier, but it does not by itself export a boundary value. The trace row T merely routes the request from the already-read interior row to the displayed boundary carrier.

The readback rows are K and M. Thus a consumer reaches a boundary value only after the compact boundary window and complete-metric Cauchy readback have been fixed.

Finally H, C, P, N transport, replay, source, and name the same row order. Since the classifier row has no host extension, quotient boundary class, or ambient compactness coordinate, no accepted consumer can bypass S, T, K, M. □

Theorem 6445.3 (Sobolev trace finite replay)**.** *The* $\mathsf{SobolevTraceBoundary}^{\uparrow}$ *package exports finite replay data for boundary trace reads. It does not export a host trace theorem, unrestricted extension operator, quotient boundary class, selected boundary representative, or ambient compactness theorem.*

Proof. The replay surface is the displayed Cont row C together with the provenance and naming rows P, N. These rows can replay only the finite route through S, B, T, K, M, X, H.

The compact information is confined to K, and the metric information is confined to M. Therefore replay can cite compact-window and complete-metric readback data, but it cannot manufacture a host compactness theorem or a global completion theorem.

The classifier row X excludes extension operators, quotient boundary classes, and selected representatives from the package coordinates. Hence the exported object is exactly finite replay data for the named boundary-readback route. □

Falsifiable prediction: Every $\mathsf{SobolevTraceBoundary}^{\uparrow}$ boundary-readback consumer reads displayed SobolevSpaceUp, SobolevTraceUp, SobolevEmbeddingUp, CompactUp, CompleteMetricUp, BHist, hsame, Cont, Pkg, and NameCert rows before receiving a boundary trace value.

Independence witness: sobolev-space, sobolev-trace, sobolev-embedding, compact, complete-metric: $\mathsf{SobolevTraceBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interior Sobolev row, boundary carrier, trace-read row, compact boundary window, complete-metric readback, classifier, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevTraceBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevTraceBoundary}^{\uparrow}$ packet is generated from an interior Sobolev carrier row, boundary carrier row, trace-read row, compact boundary window, complete-metric readback row, classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SobolevTraceBoundary}\uparrow}$, seed)

Scope closed: Definition 6445.1, Definition 6445.2, and Definition 6445.3 seed the finite boundary-readback bridge over Sobolev space, Sobolev trace, Sobolev embedding, compact, and complete-metric neighbours.

Formal verification: FormalStatus(NameCert$_{\mathsf{SobolevTraceBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host trace theorem, unrestricted extension operator, quotient boundary class, selected representative, ambient compactness theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trace-row ordering, compact boundary window exactness, complete-metric readback, classifier nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SobolevTraceBoundaryUp.

Chapter 6446

A Concrete Naming Certificate for $\mathsf{LocatedIntervalArithmetic}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalArithmetic}^{\uparrow}$ packet records finite interval arithmetic over located real endpoints while keeping every endpoint, tolerance, and enclosure read inside displayed BEDC rows. It sits after the dyadic endpoint core of chapter 291, the regular-rational readback route of chapter 301, the terminal real-name seal of chapter 38, the endpoint-plus-radius carrier of chapter 1709, and the real interval arithmetic packet of chapter 1969. The packet gives L10 Real and Completion arguments one finite enclosure service for located endpoint operations; it is not a quotient interval type, a trichotomy service, a selected-limit construction, an ambient completeness theorem, or a host interval algebra.

Definition 6446.1 (Located interval arithmetic carrier)**.** *A* $\mathsf{LocatedIntervalArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, D, R, Q, E, +_\ell, \cdot_\ell, W, H, C, P, N)$$

with the following displayed rows:

1. *I and J are input interval rows accepted by* $\mathsf{LocatedRealInterval}^{\uparrow}$*, exposing lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, radius rows, locator reads, and locatedness moduli.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic tolerances, endpoint probes, outward comparison rows, and scale refinements consumed by the interval operation.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint windows selected from I, J, D.*
4. *Q is the* $\mathsf{RealIntervalArithmetic}^{\uparrow}$ *enclosure row that packages the corresponding real-interval operation over the same displayed endpoint data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by every exported endpoint and every width claim.*
6. *$+_\ell$ is the located addition row, whose output interval enclosure is obtained from the displayed endpoint sums, dyadic tolerance ledger, regular-rational readback, and real enclosure row.*
7. *$\cdot_\ell$ is the located multiplication row, whose output interval enclosure is obtained from displayed endpoint products, outward comparison rows, dyadic tolerance ledger, regular-rational readback, and real enclosure row.*
8. *W is the finite-width witness row. It records the requested enclosure width and the displayed tolerance route that bounds the exported interval.*

9. *H records componentwise* hsame *transport for the input intervals, dyadic ledger, regular-rational readback, real interval enclosure, real seal, operation rows, width witness, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a requested located interval operation through $I, J, D, R, Q, E, +_\ell, \cdot_\ell, W, H, P$*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalArithmetic}^\uparrow$, $\mathsf{LocatedRealInterval}^\uparrow$, $\mathsf{RealIntervalArithmetic}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalArithmetic}^\uparrow}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located interval-operation packet. It has no coordinate for quotient interval equality, endpoint trichotomy, countable choice, selected limits, ambient $\mathsf{Real}^\uparrow$ *completeness, host interval algebra, or detached real representatives.*

Theorem 6446.2 (Located interval arithmetic NameCert obligations)**.** *For every accepted* LocatedIntervalArithmeti *carrier*

$$A = (I, J, D, R, Q, E, +_\ell, \cdot_\ell, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalArithmetic}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two located real interval inputs, dyadic endpoint ledger, regular-rational readback, real interval enclosure row, real seal, located addition row, located multiplication row, width witness, and structural rows H, C, P, N;*
2. *endpoint route exactness, requiring each exported endpoint to originate in the displayed* $\mathsf{LocatedRealInterval}^\uparrow$ *inputs and to pass through D, R, Q, E before any Real or Completion consumer reads it;*
3. *dyadic tolerance stability, requiring D to be the only source of endpoint tolerances, outward comparisons, and scale refinements consumed by the operation rows;*
4. *regular-rational readback stability, requiring R to expose the same finite endpoint windows that D and the input interval rows display;*
5. *operation enclosure soundness, requiring $+_\ell$ and $\cdot_\ell$ to preserve located interval containment through the displayed endpoint arithmetic, tolerance ledger, readback row, and real enclosure row;*
6. *seal non-escape, requiring E to be the only* $\mathsf{Real}^\uparrow$ *handoff for exported endpoints and width claims;*
7. *transport and replay stability, requiring H and C to preserve the input interval rows, dyadic ledger, readback row, real enclosure row, seal, operation rows, width witness, provenance, and local name componentwise;*
8. *exclusion, ruling out quotient interval equality, endpoint trichotomy, countable choice, selected limits, ambient* $\mathsf{Real}^\uparrow$ *completeness, host interval algebra, and detached real representatives as local data.*

No located interval arithmetic obligation is stored outside these rows.

Proof. Project the carrier of Definition 6446.1. The projection exposes exactly the two located real interval inputs, the dyadic endpoint ledger, regular-rational readback, real interval enclosure row, real seal, operation rows, width witness, and structural rows.

Endpoint route exactness is the displayed chain

$$I, J \longmapsto D \longmapsto R \longmapsto Q \longmapsto E.$$

The input intervals supply the located endpoint rows, the dyadic ledger supplies tolerances and outward comparisons, the regular-rational row supplies the finite endpoint readback, the real interval arithmetic row packages the enclosure, and the seal row is the only terminal Real handoff.

Dyadic tolerance stability and regular-rational readback stability are read from D and R. The operation rows can inspect endpoint windows only through those two rows, so they cannot substitute a host endpoint comparison or an unrelated real representative.

Operation enclosure soundness is supplied by $+_\ell$ and $\cdot_\ell$. Addition reads displayed endpoint sums, while multiplication reads displayed endpoint products and outward comparison rows; both operations then pass through Q, E, W before any consumer receives the enclosure.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no quotient interval equality, endpoint trichotomy, countable-choice selector, selected limit, ambient Real completeness theorem, host interval algebra, or detached real representative, none of those objects can be exported. □

Theorem 6446.3 (Located interval arithmetic Real-seal non-escape)**.** *Every exported endpoint, width claim, addition enclosure, or multiplication enclosure of an accepted* $\mathsf{LocatedIntervalArithmetic}^{\uparrow}$ *carrier factors through the displayed route*

$$I, J \longmapsto D \longmapsto R \longmapsto Q \longmapsto E \longmapsto +_\ell, \cdot_\ell, W \longmapsto H, C, P, N.$$

Consequently every downstream $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion consumer receives only the located interval enclosure displayed by the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *inputs,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{RealIntervalArithmetic}^{\uparrow}$ *enclosure row. The consumer cannot replace this route by quotient interval equality, endpoint trichotomy, countable choice, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host interval algebra, or a detached real representative.*

Proof. Start with an accepted carrier $A = (I, J, D, R, Q, E, +_\ell, \cdot_\ell, W, H, C, P, N)$. The only interval inputs are the two $\mathsf{LocatedRealInterval}^{\uparrow}$ rows I and J, so every exported operation begins with displayed located real endpoints and radius data.

The dyadic ledger D records the tolerances, endpoint probes, outward comparisons, and scale refinements. The readback row R exposes the regular-rational endpoint windows that those dyadic rows select.

The real interval arithmetic row Q packages the corresponding enclosure over the same endpoints. The seal row E is then the only route into $\mathsf{Real}^{\uparrow}$, so endpoint and width claims cannot bypass the displayed real-name surface.

For addition, project $+_\ell$ and read the output enclosure from the displayed endpoint sums, tolerance ledger, readback, and real enclosure row. For multiplication, project $\cdot_\ell$ and read the output enclosure from displayed endpoint products, outward comparisons, tolerance ledger, readback, and real enclosure row. The width witness W records the finite tolerance route used by both operations.

Componentwise transport and displayed replay through H and C, together with provenance and local naming in P, N, preserve the same route for downstream consumers. Therefore every exported read is exhausted by the displayed finite packet and no excluded quotient, order, choice, limit, completeness, host-algebra, or detached-representative coordinate is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalArithmetic}^{\uparrow}}$.

Constructive story: A LocatedIntervalArithmetic$^{\uparrow}$ packet is generated from two LocatedRealInterval input rows, a DyadicRatCore endpoint tolerance ledger, a RegSeqRat readback row, a RealIntervalArithmetic enclosure row, a terminal RealUp seal, displayed located addition and multiplication rows, a finite-width witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedIntervalArithmetic}^{\uparrow}}$, seed)

Scope closed: Definition 6446.1, Definition 6446.2, and Definition 6446.3 seed the finite located interval arithmetic enclosure packet over LocatedRealInterval, DyadicRatCore, RegSeqRat, RealIntervalArithmetic, and RealUp consumers.

Formal verification: FormalStatus(NameCert$_{\text{LocatedIntervalArithmetic}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient interval equality, endpoint trichotomy, countable choice, selected limits, ambient Real$^{\uparrow}$ completeness, host interval algebra, detached real representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint route exactness, dyadic tolerance stability, regular-rational readback stability, operation enclosure soundness, Real-seal non-escape, transport/replay/provenance coverage, and exclusion for BEDC.Derived.LocatedIntervalArithmeticUp.

Chapter 6447

A Concrete Naming Certificate for $\mathsf{FittingIdeal}^{\uparrow}$

Following Eisenbud and Bourbaki, the $\mathsf{FittingIdeal}^{\uparrow}$ packet records the finite presentation-matrix route by which a finitely presented module exposes its Fitting ideals through generated minor ledgers. The certificate reads rows for the chosen presentation, displayed matrix entries, finite minors, generated ideal handoff, module transport, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert naming. It is not a host commutative-algebra oracle, a hidden determinant computation, or an equality principle for all presentations.

The coefficient row is read through chapter 43, the finitely presented module row is read through chapter 46, and the generated-minor handoff is read through chapter 81. These are substantive carrier dependencies: the packet exposes the displayed algebraic rows used by the minor ledger, not a host commutative-algebra oracle.

Definition 6447.1 (FittingIdeal carrier)**.** *A* $\mathsf{FittingIdeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, M, P, A, I, G, T, H, C, Q, N)$$

with the following rows:

1. *R is the coefficient-ring row used only as displayed scalar data;*
2. *M is the finitely presented module row;*
3. *P is the finite presentation row $R^m \to R^n \to M$;*
4. *A is the displayed presentation matrix with finite row and column ledgers;*
5. *I is the generated-minor ledger, listing the minors of the required size and their finite determinant expansions;*
6. *G is the ideal handoff row generated by the minor ledger I;*
7. *T is the module-transport row recording how a permitted presentation change is read before the ideal handoff is reused;*
8. *H records componentwise* hsame *transport for ring, module, presentation, matrix, minor, ideal, and transport rows;*
9. *C records displayed* Cont *replay from presentation rows to minors and from minors to the ideal handoff;*

10. *Q is the* Pkg *provenance over the algebraic rows,* BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FittingIdeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient ideal-equality oracle, a presentation-independent quotient proof outside T, H, C, Q, N, an infinite determinant family, or a host module category.

Theorem 6447.2 (FittingIdeal minor-ledger obligations)**.** *For every accepted* $\mathsf{FittingIdeal}^{\uparrow}$ *carrier $F = (R, M, P, A, I, G, T, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FittingIdeal}^{\uparrow}}$ *surface consists exactly of carrier admission, finite presentation exposure, generated-minor ledger exposure, ideal handoff through G, module transport through T, componentwise* hsame *transport through H, displayed replay through C, provenance through Q, and local naming through N. A public Fitting-ideal read must pass through those rows before it may cite the generated ideal.*

Proof. Project the accepted packet. The ring, module, presentation, matrix, generated-minor ledger, ideal handoff, and module-transport rows are displayed coordinates of the finite carrier.

The generated ideal is read only from G after the minor ledger I has been exposed. The determinant expansions in I are finite rows over the matrix A, so the handoff never invokes an unbounded determinant family.

Presentation transport is confined to T, H, C, Q, N. These rows record how the presentation change is transported, replayed, sourced, and named before the ideal handoff is reused by a consumer.

Since the carrier has no ambient ideal-equality oracle, no hidden presentation-independent quotient proof, and no host module category coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FittingIdeal}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FittingIdeal}^{\uparrow}}$, seed)

Scope closed: Definition 6447.1 and Definition 6447.2 seed the finite presentation-matrix, minor-ledger, ideal-handoff, module-transport, and local naming surface for Fitting ideals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FittingIdeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host ideal equality, all-presentation invariance, external commutative-algebra completeness, or a host module category.

Upgrade path: Obligation closure requires checked carrier admission, minor-ledger exactness, ideal handoff locality, module-transport stability, and consumer non-escape for BEDC.Derived.FittingIdealUp.

Constructive story: A $\mathsf{FittingIdeal}^{\uparrow}$ packet is generated from a coefficient-ring row, finitely presented module row, finite presentation matrix, generated-minor ledger, ideal handoff row, module transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 6448

A Concrete Naming Certificate for AlternativeCodeReassignmentSpectra$^{\uparrow}$

The AlternativeCodeReassignmentSpectra$^{\uparrow}$ packet realizes the comparative reassignment-spectra vision of Definition 9462.1208 as a finite science-facing NameCert target. It refines the comparison-table carrier of chapter 3589, reads Walsh and median-closure spectral rows through chapter 3897, keeps the decoded biological bridge explicit through chapter 5177, and routes support-family evidence through chapter 4079. The packet is a finite support, reassignment, spectral, transport, replay, provenance, and local naming surface; it is not a biological mechanism theorem or an exhaustive genetic-code survey.

Definition 6448.1 (Alternative-code reassignment spectra carrier)**.** *An* AlternativeCodeReassignmentSpectra$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, W, E, T, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the sparse support row over the* 13 *displayed reassigned codons and their module ledger;*
2. *R is the alternative-code reassignment row, reading standard-table and curated alternative-table comparison through* AlternativeCodeReassignment$^{\uparrow}$*;*
3. *W is the spectral row, recording Walsh, median-closure, and local module spectrum data from the reassignment-spectral-compression vision;*
4. *E is the support-nerve realization row, tying the reassignment support to a finite boundary-subcomplex read;*
5. *T is the transport row from channel-genetic-code bridge data to codon-window and reassignment rows;*
6. *L is the replay row that orders support exposure, reassignment comparison, spectral readback, support-nerve realization, and local naming;*
7. *H records componentwise* hsame *transport for all displayed support, reassignment, spectral, bridge, realization, replay, provenance, and naming rows;*
8. *C records displayed* Cont *replay for the same route;*
9. *P is the* Pkg *provenance over the source vision, sibling concrete-instance packets,* BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}}$ *row.*
The carrier has no coordinate for uncurated code-table enumeration, biological causality, probabilistic fitness, quotient codon semantics, or a hidden spectral-analysis oracle.

Theorem 6448.2 (Alternative-code reassignment spectra NameCert obligations)**.** *For every accepted* $\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}$ *packet* $A = (S, R, W, E, T, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}}$ *surface consists exactly of support admission through* S*, reassignment comparison through* R*, spectral readback through* W*, support-nerve realization through* E*, channel-to-codon transport through* T*, ordered replay through* L, C*, componentwise* hsame *transport through* H*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted packet. The support row S and reassignment row R expose the finite comparative table data before any spectrum is read.

The spectral row W is downstream from that displayed support. It records Walsh and median-closure information as finite row data and cannot introduce a host Fourier-analysis oracle or uncurated biological table family.

The rows E, T, L, H, C, P, N bind the support-nerve, bridge, replay, transport, provenance, and local naming route. Since all consumer reads pass through these displayed rows, the NameCert surface is exactly the finite packet surface and not a claim of biological mechanism, exhaustive coverage, or quotient codon semantics. □

The Lean follow-up for this packet must provide `Nontrivial` and `FieldFaithful` instances in `BEDC/Derived/AlternativeCodeReassignmentSpectraUp/TasteGate.lean` before a checked local TasteGate can support the carrier.

Falsifiable prediction: Every accepted $\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}$ read displays support, reassignment, spectral, transport, replay, provenance, and local naming rows; a read that reaches a spectral conclusion without any one of those rows refutes the packet.

Independence witness: alternative-code-reassignment, reassignment-spectral-compression, channel-genetic-code-bridge, support-nerve-realization: none is bijective with $\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}$ because this packet jointly carries sparse support, comparative reassignment, Walsh and median-closure spectral rows, channel-to-codon transport, support-nerve realization, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}$ packet is generated from sparse reassignment support rows, alternative-code comparison rows, spectral readback rows, support-nerve realization rows, channel-to-codon transport, ordered replay, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}}$, seed)

Scope closed: Definition 6448.1 and Definition 6448.2 seed the finite support, reassignment, spectral, transport, replay, provenance, and naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignmentSpectra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological causality, exhaustive genetic-code coverage, probabilistic fitness, quotient codon semantics, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, support exposure, reassignment comparison, spectral readback, support-nerve realization, transport stability, replay provenance, Nontrivial AlternativeCodeReassignmentSpectraUp, and FieldFaithful AlternativeCodeReassignmentSpectraUp for BEDC.Derived.AlternativeCodeReassignmentSpectraUp.

Chapter 6449

A Concrete Naming Certificate for $\mathsf{AsymptoticDimension}^{\uparrow}$

Following Gromov and Roe-style coarse geometry, the $\mathsf{AsymptoticDimension}^{\uparrow}$ packet records asymptotic dimension as finite uniformly bounded cover families indexed by displayed scale rows, rather than as an ambient large-scale invariant over a completed category of metric spaces. It reads metric data from chapter 57, proper closed-ball discipline from **??**, entourage-style scale transport from chapter 339, and quasi-isometry-facing stability from chapter 3883. The carrier names exactly the finite rows that downstream coarse consumers can replay.

Definition 6449.1 (AsymptoticDimension carrier)**.** *An* $\mathsf{AsymptoticDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, S, U, M, T, R, P, N)$$

with the following displayed rows:

1. *X is the metric source row, read through the* $\mathsf{Met}^{\uparrow}$ *surface and its proper-metric closed-ball boundary when boundedness is inspected;*
2. *S is a finite list of scale requests, each a displayed positive radius row over the same metric source;*
3. *U assigns to each scale row a finite uniformly bounded cover family, with every cover member carrying a visible diameter bound and membership readback;*
4. *M assigns to each scale row the finite multiplicity ledger proving that every scale ball meets at most the displayed number of cover rows;*
5. *T records the componentwise* hsame *transport of the metric source, scale rows, cover families, diameter bounds, and multiplicity ledgers;*
6. *R records the displayed* Cont *replay route for reading the cover and multiplicity rows in downstream coarse consumers;*
7. *P is the* Pkg *provenance over* $\mathsf{AsymptoticDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{QuasiIsometry}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticDimension}^{\uparrow}}$ *row.*

The carrier is only this finite packet. It does not select an infinite family of covers, construct a quotient coarse category, import choice-selected Lebesgue numbers, or assert host invariance under all quasi-isometries.

Theorem 6449.2 (AsymptoticDimension NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticDimension}^{\uparrow}$ *carrier* $A = (X, S, U, M, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticDimension}^{\uparrow}}$ *surface consists of carrier admission, scale-row coverage, uniform boundedness for each displayed cover family, finite multiplicity for each displayed scale row, componentwise transport, replay through the displayed continuation rows, provenance over the metric/proper/uniform/quasi-isometry dependency surface, and the local naming row. No obligation exports a selected global cover sequence, quotient metric-space equality, open-cover compactness principle, or ambient coarse category.*

Proof. Carrier admission is projection from Definition 6449.1. The accepted packet exposes the metric source, finite scale schedule, uniformly bounded cover rows, multiplicity rows, transport, replay, provenance, and local name.

For each scale row in S, use finite induction over the displayed cover family in U. The base case reads the empty prefix and its vacuous multiplicity ledger; the successor step adds one bounded cover row, its diameter witness, and the corresponding membership readback.

Multiplicity is checked by a second finite induction over the cover rows meeting a displayed scale ball. The ledger M counts only those visible rows and therefore cannot smuggle in an open-cover compactness theorem or a choice-selected subcover.

Finally transport and replay are row projections. The rows T and R preserve the same metric source, scale request, cover family, and multiplicity ledger, while P and N source and name that finite surface. □

Theorem 6449.3 (AsymptoticDimension quasi-isometry handoff)**.** *Every quasi-isometry-facing consumer of an accepted* $\mathsf{AsymptoticDimension}^{\uparrow}$ *carrier must read the source metric row, finite scale schedule, uniformly bounded cover family, and finite multiplicity ledger before it may hand the packet to* $\mathsf{QuasiIsometry}^{\uparrow}$*. The handoff preserves the displayed cover and multiplicity rows under* hsame *transport and* Cont *replay, but it exports no host proof that asymptotic dimension is invariant under all quasi-isometries, no selected coarse inverse, and no quotient equality of metric spaces.*

Proof. Project $A = (X, S, U, M, T, R, P, N)$. The only coarse data visible to the consumer are the metric source, scale rows, cover families, and multiplicity ledgers.

Read the quasi-isometry dependency through the provenance row P. It points to the existing $\mathsf{QuasiIsometry}^{\uparrow}$ surface only after the metric, properness, and uniform-scale rows have already been displayed.

Transport stability is componentwise. The row T moves scale, cover, and multiplicity data together, and R replays that same route for a downstream quasi-isometry consumer.

Thus the handoff is a finite NameCert route. Any full invariance theorem, selected inverse, quotient metric equality, or external coarse category must be supplied elsewhere and is not contained in this packet. □

Falsifiable prediction: An accepted $\mathsf{AsymptoticDimension}^{\uparrow}$ read is refuted if a consumer obtains a dimension bound without displaying the metric source, finite scale rows, uniformly bounded cover families, finite multiplicity ledgers, transport, replay, provenance, and local naming rows.

Independence witness: metric, propermetric, uniformspace, quasi-isometry: none is bijective to $\mathsf{AsymptoticDimension}^{\uparrow}$ because this packet jointly carries scale-indexed uniformly bounded covers and multiplicity ledgers for a coarse dimension read.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticDimension}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AsymptoticDimension}^{\uparrow}$ packet is generated from a metric source row, finite scale requests, uniformly bounded cover families, finite multiplicity ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AsymptoticDimension}\uparrow}$, seed)

Scope closed: Definition 6449.1, Definition 6449.2, and Definition 6449.3 seed the finite coarse-dimension certificate over displayed metric, scale, cover, multiplicity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{AsymptoticDimension}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, quasi-isometry invariance for all metric spaces, quotient coarse categories, selected cover sequences, open-cover compactness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite cover boundedness, finite multiplicity exactness, transport replay, provenance, and quasi-isometry handoff for BEDC.Derived.AsymptoticDimensionUp.

Chapter 6450

A Concrete Naming Certificate for $\mathsf{LocatedRealApartness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedRealApartness}^{\uparrow}$ packet records apartness of real names as a positive finite witness rather than as a trichotomy decision. It sits over the dyadic tolerance surface of chapter 291, the finite stream-window route of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, the apartness-to-order bridge of chapter 1905, and the located-order packet of chapter 3640. The chapter names the constructive apartness witness needed by Real and Completion consumers; it does not assert real trichotomy, quotient real equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6450.1 (Located real apartness carrier)**.** *A* $\mathsf{LocatedRealApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, S, T, D, R, O, L, B, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Real}^{\uparrow}$ *name rows whose apartness is being witnessed.*
2. *S and T are finite* $\mathsf{StreamName}^{\uparrow}$ *window rows for X and Y.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two displayed windows at one positive dyadic scale.*
4. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the rational approximants of X and Y at the scale selected by D.*
5. *O is the located-order comparison row consumed from* $\mathsf{RealLocatedOrder}^{\uparrow}$*.*
6. *L is the positive separation row: it records a dyadic gap $\delta > 0$ and the displayed rational-window comparison that places the two names at distance at least δ on the inspected branch.*
7. *B is the bridge row to* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$*, recording that the positive gap is read as apartness and not as a trichotomy of all comparisons.*
8. *H records componentwise* hsame *transport for the real names, stream windows, dyadic ledger, regular readback, located-order row, separation row, bridge row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an apartness request through $X, Y, S, T, D, R, O, L, B, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, $\mathsf{RealLocatedOrder}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealApartness}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one positive located-apartness certificate. It has no coordinate for total order trichotomy, host real equality, quotient stream equality, selected representatives, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6450.2 (Located real apartness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealApartness}^{\uparrow}$ *carrier*

$$A = (X, Y, S, T, D, R, O, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealApartness}^{\uparrow}}$ *surface consists exactly of carrier admission, paired window admission, dyadic tolerance exactness, regular-rational readback stability, positive separation display, apartness-order bridge compatibility, componentwise transport, displayed replay, provenance, and local naming. Each obligation is checked by finite projection through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, $\mathsf{RealLocatedOrder}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Project the carrier of Definition 6450.1. The real-name rows X, Y are read only through the finite windows S, T, and the dyadic tolerance ledger D selects the positive scale at which the two windows are compared.

The paired readback row R converts the displayed stream windows into regular rational approximants. The located-order row O and the separation row L then display a positive dyadic gap on the inspected branch.

The bridge row B records that this positive gap is consumed as apartness. It does not turn the packet into a global trichotomy service, because the carrier stores only the displayed separated branch and no exhaustive comparison row.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no excluded equality, choice, selected-limit, trichotomy, or completeness coordinate appears in the packet, the local NameCert obligations are exhausted by the stated rows. □

Theorem 6450.3 (Located real apartness finite-witness route)**.** *Every accepted* $\mathsf{LocatedRealApartness}^{\uparrow}$ *packet exposes apartness through the finite route*

$$\mathsf{Real}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RealLocatedOrder}^{\uparrow} \longmapsto \mathsf{Real}$$

The exported witness is a displayed positive separation row, transported and replayed by H, C, P, N. It cannot be consumed as real trichotomy, host equality negation, quotient-name inequality, countable choice, selected representatives, selected limits, or ambient Real completeness.

Proof. Start with the two $\mathsf{Real}^{\uparrow}$ rows X, Y. A consumer cannot read them directly as a comparison theorem; it must first open the finite StreamName windows S, T and the paired RegSeqRat readback row R.

The dyadic tolerance ledger D then fixes the scale of the comparison. The located-order row O displays the chosen branch, and the separation row L records the positive dyadic gap visible on that branch.

The bridge row B turns the displayed gap into the apartness witness consumed by the local NameCert row. Componentwise transport and continuation replay preserve the same rows, while the provenance row records the sibling surfaces involved in the read.

Thus the route supplies a finite positive witness. Nothing in the carrier supplies an exhaustive comparison of all real names, a quotient-name equality test, a choice schedule, a selected representative, a selected limit, or a completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealApartness}\uparrow}$.

Constructive story: A $\mathsf{LocatedRealApartness}^{\uparrow}$ packet is generated from two RealUp name rows, paired StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealLocatedOrder comparison row, a positive separation row, a RealApartnessOrderBridge handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealApartness}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6450.1, Definition 6450.2, and Definition 6450.3 seed the finite positive apartness witness over Real, regular-rational, dyadic, located-order, and apartness-order bridge rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealApartness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, host real equality negation, quotient-name inequality, countable choice, selected representatives, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-window admission, dyadic tolerance exactness, regular-rational readback stability, positive separation display, apartness-order bridge compatibility, transport, replay, provenance, and exclusion coverage for BEDC.Derived.LocatedRealApartnessUp.

Chapter 6451

A Concrete Naming Certificate for FiniteBishopConvergenceCriterion$^{\uparrow}$

Following Bishop and Bridges, the FiniteBishopConvergenceCriterion$^{\uparrow}$ packet records the finite convergence-test surface used in the L10 route before any real-completeness theorem or quotient stream equality is available. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the convergence-modulus ledger of chapter 2074, and the Bishop real-line interface of chapter 2062. The packet names a displayed finite test for convergence evidence; it is not Real completeness, a quotient of streams, a selected limit, trichotomy, countable choice, or bridge checking.

Definition 6451.1 (Finite Bishop convergence criterion carrier)**.** *A* FiniteBishopConvergenceCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, W, Q, R, M, I, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row naming the finite request radius and slack budget.*
2. *W is the* StreamName$^{\uparrow}$ *window row recording the finite entries of the tested sequence that are inspected for that request.*
3. *Q is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows.*
4. *R is the terminal* Real$^{\uparrow}$ *seal row that may be read only after D, W, Q have been replayed.*
5. *M is the* RealModulusOfConvergence$^{\uparrow}$ *ledger row recording the finite threshold used by the test.*
6. *I is the* BishopRealLineInterface$^{\uparrow}$ *handoff row through which Bishop-style real-line consumers read the same finite convergence evidence.*
7. *H stores componentwise* hsame *transport for tolerance, window, readback, Real seal, modulus, interface, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for the same finite criterion route.*
9. *P is the* Pkg *provenance over* FiniteBishopConvergenceCriterion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RealModulusOfConvergence$^{\uparrow}$, BishopRealLineInterface$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, W, Q, R, M, I, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6451.2 (Finite Bishop convergence window test)**.** *Every accepted* $\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}$ *carrier $F = (D, W, Q, R, M, I, H, C, P, N)$ exposes a convergence test only through the finite route*

$$D, W \longmapsto Q \longmapsto M, R \longmapsto I, H, C, P, N.$$

Thus a consumer must inspect the dyadic tolerance row and the $\mathsf{StreamName}^{\uparrow}$ *window before it can use regular rational readback, convergence-modulus evidence, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, or the Bishop real-line interface.*

Proof. Project the carrier of Definition 6451.1. The first visible stage is the tolerance and window pair D, W, so the request radius and finite observation schedule are fixed before any rational readback is available.

Replay the read through C. The replay opens Q only after the displayed windows have been inspected, and it opens M and R only after the same readback has been tied to the dyadic budget.

Finally transport through H and source the route through P, N. The Bishop interface row I receives exactly the same finite evidence, so no selected limit, quotient stream equality, host equality, choice principle, completeness theorem, Lean target, or bridge row can be exported. □

Theorem 6451.3 (Finite Bishop convergence L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}$ *packet factors through chapter 291, chapter 126, chapter 301, chapter 38, chapter 2074, and chapter 2062 before it may read convergence-test evidence.*

Proof. Read the carrier rows in order. The dyadic tolerance row D supplies the request budget, the stream row W supplies the inspected finite window, and the regular rational row Q supplies the readback seen by L10.

The convergence-modulus row M and Real seal R are downstream of that route. They may certify that the displayed finite window is sufficient for the named request, but they do not introduce a host real limit or a quotient equality of streams.

The interface row I packages the same data for Bishop real-line consumers, while H, C, P, N transport, replay, source, and name it. The L10 handoff is therefore exactly the finite sibling route listed in the statement. □

Theorem 6451.4 (Finite Bishop convergence nonescape)**.** *An accepted* $\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}$ *carrier cannot export Real completeness, quotient stream equality, selected representatives, selected limits, host real equality, trichotomy, countable choice, Lean verification, or bridge checking. Every public read is exhausted by the rows $D, W, Q, R, M, I, H, C, P, N$.*

Proof. Inspect the carrier coordinates. They are exactly the dyadic tolerance row, finite stream window, regular rational readback, Real seal, convergence-modulus ledger, Bishop interface row, componentwise transport, replay, provenance, and local naming.

Each public read must be a projection from those rows or a replay through C, H, P, N. None of the excluded objects is a row of the carrier, and none can be created by projection, replay, or componentwise transport.

Therefore the accepted packet is exhausted by the finite NameCert route displayed above. The nonescape claim follows from row exhaustion rather than from any external completeness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName finite windows, RegSeqRat readback, a RealUp seal, a RealModulusOfConvergence ledger, a BishopRealLineInterface handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6451.1, Definition 6451.2, Definition 6451.3, and Definition 6451.4 seed the finite Bishop convergence-test route for L10 consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBishopConvergenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected representatives, selected limits, host real equality, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-test exactness, L10 handoff exactness, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteBishopConvergenceCriterionUp.

Chapter 6452

A Concrete Naming Certificate for $\mathsf{CauchyTailCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailCriterion}^{\uparrow}$ packet records the finite tail-test route by which a Cauchy read is accepted before a regular rational or real-seal consumer may use it. It sits over the finite window surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the inspected modulus discipline of chapter 292, the terminal real-name seal of chapter 38, and the fast subsequence route of chapter 1518. The packet is a local L10 finite tail-test object: it does not select a limit, quotient streams, use countable choice, invoke trichotomy, compare host real values, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6452.1 (Cauchy tail criterion carrier)**.** *A* $\mathsf{CauchyTailCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, D, R, M, E, F, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row listing the inspected source windows and the finite stage pairs used by the test.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested radius, comparison slack, and dyadic budget consumed by each inspected pair.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic budgets.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *inspection row naming the stage threshold that licenses the tail test.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row visible only after the window, dyadic, readback, and modulus rows have been consumed.*
6. *F is the* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *handoff row available only as a downstream finite witness that the same tail-test data can feed a fast regular subsequence route.*
7. *B is the tail-test ledger, recording that every displayed pair of inspected tail windows beyond the threshold in M is contained by the tolerance row D.*
8. *H records componentwise* hsame *transport for S, D, R, M, E, F, B and the structural support rows.*

9. *C records displayed* Cont *replay of the route*

$$S, D, M \longmapsto R \longmapsto E \longmapsto F \longmapsto B.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyTailCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FastCauchySubsequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailCriterion}^{\uparrow}}$ *row.*

Accepted carriers are compared only through these displayed rows. There is no coordinate for a selected limit, quotient Cauchy class, host real equality, countable-choice selector, trichotomy row, or ambient completeness principle.

Theorem 6452.2 (Cauchy tail criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailCriterion}^{\uparrow}$ *carrier $T = (S, D, R, M, E, F, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold exposure through M, terminal* $\mathsf{Real}^{\uparrow}$ *seal delivery through E,* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *handoff through F, tail-test accountability through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted packet. The carrier displays exactly the tail-window, dyadic tolerance, regular readback, inspected modulus, real seal, fast-subsequence handoff, tail-test ledger, transport, replay, provenance, and local naming rows.

The finite test is read through S, D, M, B. The row S lists only inspected windows, M names the displayed threshold, D records the tolerance budget, and B asserts the pairwise tail containment for those displayed rows.

The consumer-facing rows are downstream of that test. The regular rational row R reads the same windows and dyadic budgets, while the real seal E is exposed only after R and the tail-test route have been projected.

Finally F, H, C, P, N preserve and name the same route. The fast-subsequence row may consume the displayed tail-test data as finite evidence, but the carrier has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, trichotomy, or ambient completeness. The listed rows therefore exhaust the local NameCert surface. □

Theorem 6452.3 (Cauchy tail criterion Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailCriterion}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FastCauchySubsequence}^{\uparrow}$$

through the displayed carrier rows before it may use the tail criterion as real-facing evidence. The handoff supplies only a finite tail-test packet; it does not supply a selected limit, quotient stream equality, host real comparison, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $T = (S, D, R, M, E, F, B, H, C, P, N)$. Projection exposes the finite window row, dyadic tolerance row, inspected modulus row, regular readback row, real seal row, fast-subsequence handoff row, and tail-test ledger.

The tail criterion is available only after S, D, M, B have been read. These rows bind every inspected pair of tail windows to a displayed modulus threshold and a displayed dyadic tolerance budget.

The regular rational and real-facing rows preserve that order. The row R reads the same window and tolerance data, and the seal row E is downstream of the regular readback rather than a host limit-selection operation.

The fast-subsequence handoff F may consume the same finite evidence after the seal route is visible. Since H, C, P, N only transport, replay, source, and name the carrier, no excluded principle remains available to a Real or completion consumer. □

Lean4 ✓ `BEDC.Derived.CauchyTailCriterionUp.CauchyTailCriterionRealSealHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6452.1, Definition 6452.2, and Definition 6452.3 seed the finite tail-test route from displayed stream windows, dyadic tolerance budgets, and inspected modulus data to regular rational readback, Real seal delivery, and fast-subsequence handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected limits, quotient stream equality, host real comparison, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, public closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exposure, dyadic tolerance accounting, modulus-threshold exposure, regular readback, Real seal delivery, fast-subsequence handoff, tail-test accountability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailCriterionUp.

Constructive story: A $\mathsf{CauchyTailCriterion}^{\uparrow}$ packet is generated from StreamName tail windows, DyadicRatCore tolerance budgets, RegSeqRat readback, inspected CauchyModulus thresholds, a RealUp seal, a FastCauchySubsequence handoff, a finite tail-test ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6453

A Concrete Naming Certificate for CompactNetModulusStability$^{\uparrow}$

Following Bishop and Bridges compact metric analysis, the CompactNetModulusStability$^{\uparrow}$ packet realizes the compact-net direction of chapter 9237 as finite NameCert data. It reads the compact metric source of chapter 129, continuous-map rows from chapter 130, the uniform-modulus output surface of chapter 369, metric triangle rows from chapter 57, and the stream, regular-rational, and Real seals of chapter 126, chapter 301, and chapter 38. The packet names stability of the finite compact-net modulus route; it is not a selected Lebesgue number, an external modulus function, sequential compactness, countable choice, quotient metric equality, or ambient completeness.

Definition 6453.1 (Compact-net modulus stability carrier)**.** *A* CompactNetModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, U, M, W, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row over the same source.*
3. *U is the* UniformModulus$^{\uparrow}$ *output row produced by the finite compact-net route.*
4. *M is the* Met$^{\uparrow}$ *triangle and distance-comparison row used to transport estimates across a bundled center.*
5. *W is the* StreamName$^{\uparrow}$ *finite observation row for the source and target names read by the map.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row obtained from those windows.*
7. *R is the terminal* Real$^{\uparrow}$ *seal row for the displayed distance estimates.*
8. *T is the compact-net stability ledger binding X, F, U, M, W, Q, R to one finite modulus route.*
9. *H stores componentwise* hsame *transport for compact-net, continuous-map, uniform-modulus, metric, stream, readback, Real-seal, stability, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay for the same compact-net modulus route.*
11. *P is the* Pkg *provenance over* CompactNetModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, Met$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{CompactNetModulusStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, F, U, M, W, Q, R, T, H, C, P, N$. It has no coordinate for a selected subcover, selected Lebesgue number, host modulus function, sequential compactness, countable choice, quotient metric equality, or ambient completeness.

Theorem 6453.2 (Compact-net modulus finite route stability)**.** *Every accepted* $\mathsf{CompactNetModulusStability}^{\uparrow}$ *carrier $K = (X, F, U, M, W, Q, R, T, H, C, P, N)$ exports a uniform-modulus read only through the finite route*

$$X, F, M \longmapsto W, Q, R \longmapsto U, T, H, C, P, N.$$

Thus compact-net coverage, continuous pointwise-modulus rows, metric triangle estimates, stream windows, regular rational readback, Real seals, and the uniform-modulus output remain bound to one displayed NameCert packet.

Proof. Project the carrier and read the source rows X, F, M. The compact metric net, continuous-map modulus row, and metric triangle row are displayed over the same source before any output modulus is available.

Replay the finite observation layer through C. The stream window W is read first, the regular rational row Q reads back the inspected names, and the Real seal R records the distance estimates used by the metric row.

The output stage U, T, H, C, P, N exposes the uniform-modulus row only after the stability ledger has replayed the same compact-net route. Row exhaustion prevents selected subcovers, selected Lebesgue numbers, host modulus functions, sequence extraction, choice, quotient metric equality, or ambient completeness from entering the read. □

Theorem 6453.3 (Compact-net modulus Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *distance estimate consumed by an accepted* $\mathsf{CompactNetModulusStability}^{\uparrow}$ *packet factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows before it reaches the stability ledger T and the* $\mathsf{UniformModulus}^{\uparrow}$ *output row U. Consequently the packet cannot export a Real-completeness principle, quotient equality of metric points, a selected compact subcover, or a host modulus selector.*

Proof. Begin with a Real-facing distance estimate in the accepted packet. The only Real row is R, and it is displayed as downstream of the finite windows W and regular rational readback Q.

Use the metric row M and the stability ledger T to transport the estimate across the compact-net center selected inside the finite packet. This transport is a replayed NameCert route, not an external triangle theorem detached from the carrier.

Finally expose the uniform-modulus row U through H, C, P, N. Since the packet has no row for completeness, quotient point equality, selected subcovers, or host modulus functions, none of those objects can escape from the Real-seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactNetModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactNetModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, a UniformModulus output, Metric triangle estimates, StreamName windows, RegSeqRat readback, RealUp distance seals, a stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactNetModulusStability}^{\uparrow}}$, seed)

Scope closed: Definition 6453.1, Definition 6453.2, and Definition 6453.3 seed the compact-net modulus stability route realized from chapter 9237.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactNetModulusStability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected Lebesgue numbers, selected subcovers, host modulus functions, sequential compactness, countable choice, quotient metric equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net source lock, pointwise-modulus binding, metric-triangle stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactNetModulus

Chapter 6454

A Concrete Naming Certificate for $\mathsf{HausdorffMomentProblem}^{\uparrow}$

Following Hausdorff on moment sequences over a compact interval, the $\mathsf{HausdorffMomentProblem}^{\uparrow}$ packet records the finite BEDC surface for a truncated interval moment table before any consumer asks for a representing measure. It specializes the finite-window route of chapter 6193 by fixing the compact support row through chapter 6257, reading polynomial tests through chapter 1728, sealing scalar values through chapter 38, and routing nonnegative polynomial checks through chapter 2297. The certificate names only displayed compact-interval handoff data, finite polynomial windows, Hankel and localizing positivity ledgers, transport, replay, provenance, and local naming; it does not export a representing measure, determinacy, uniqueness, selected support, or ambient compactness theorem.

Definition 6454.1 (Hausdorff moment problem carrier)**.** *A* $\mathsf{HausdorffMomentProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, I, R, P, V, Q, L, S, T, C, G, N)$$

with the following displayed rows:

1. *M is the inherited* $\mathsf{MomentProblem}^{\uparrow}$ *finite moment-window row, including the rational and real readbacks of the inspected moment entries.*
2. *I is the* $\mathsf{CompactRealInterval}^{\uparrow}$ *support row, displaying the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seals for the compact interval on which the Hausdorff read is requested.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row used for endpoint values, moment entries, and tolerance budgets.*
4. *P is the* $\mathsf{RealPolynomial}^{\uparrow}$ *finite polynomial-test row, naming only the displayed coefficient lists and evaluation windows consumed by the packet.*
5. *V is the* $\mathsf{PositiveCone}^{\uparrow}$ *nonnegative-read row for the finite polynomial tests and scalar comparison witnesses.*
6. *Q is the finite Hankel positivity row inherited from* $\mathsf{MomentProblem}^{\uparrow}$*, restricted to the displayed compact interval support.*
7. *L is the localizing positivity row for the endpoint constraints, recording the finite windows by which $(x-a)p(x)$ and $(b-x)p(x)$ are inspected through P, V, R, I.*

8. *S records componentwise* hsame *transport for the moment window, compact interval support, real readbacks, polynomial tests, positivity rows, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a Hausdorff finite-window request through M, I, R, P, V, Q, L, S.*
10. *C is the* Pkg *provenance over* $\mathsf{HausdorffMomentProblem}^{\uparrow}$, $\mathsf{MomentProblem}^{\uparrow}$, $\mathsf{CompactRealInterval}^{\uparrow}$, $\mathsf{RealPolynomial}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *G is the compact-interval handoff ledger recording that every support-facing read uses the displayed I row rather than an ambient compactness principle.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMomentProblem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a representing measure, infinite Hausdorff sequence, determinacy theorem, selected support measure, host Riesz representation theorem, or ambient completeness principle.

Theorem 6454.2 (Hausdorff moment problem NameCert obligations)**.** *For every accepted* HausdorffMomentProble *carrier*

$$H = (M, I, R, P, V, Q, L, S, T, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffMomentProblem}^{\uparrow}}$ *surface consists exactly of MomentProblem finite-window admission, compact real interval support admission, real scalar readback, finite real-polynomial test admission, positive-cone nonnegative readback, finite Hankel positivity, localizing endpoint positivity, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, compact-interval handoff, and local naming for the displayed rows*

$$M, I, R, P, V, Q, L, S, T, C, G, N.$$

No consumer of the certificate receives a representing measure, uniqueness, determinacy, selected support, an infinite positivity theorem, or a host compact-interval existence theorem.

Proof. Project the accepted carrier of Definition 6454.1. The inherited moment data are confined to the finite $\mathsf{MomentProblem}^{\uparrow}$ row M, while the compact support is confined to the endpoint row I and the real readback row R.

Polynomial tests are not host functions. They are read through the finite $\mathsf{RealPolynomial}^{\uparrow}$ row P, and their nonnegative comparisons are read through the $\mathsf{PositiveCone}^{\uparrow}$ row V before the finite Hankel row Q or localizing row L may be consumed.

The structural rows S, T, C, G, N transport, replay, source, hand off, and name exactly the same finite rows. Since the carrier contains no representing measure, infinite moment sequence, selected support measure, determinacy proof, Riesz theorem, or ambient completeness coordinate, those objects cannot be exported by the local certificate. □

Theorem 6454.3 (Hausdorff moment problem Hankel positivity route)**.** *Every Hankel-positivity read exported by an accepted* $\mathsf{HausdorffMomentProblem}^{\uparrow}$ *packet first passes through the* $\mathsf{MomentProblem}^{\uparrow}$ *finite-window row, the* $\mathsf{Real}^{\uparrow}$ *scalar readback row, the* $\mathsf{RealPolynomial}^{\uparrow}$ *finite test row, and the* $\mathsf{PositiveCone}^{\uparrow}$ *nonnegative-read row before the localizing endpoint ledger may be consumed. In carrier terms, the route is*

$$M, R, P, V \longmapsto Q, L \longmapsto S, T, C, G, N.$$

It exports no infinite Hankel operator, representing measure, determinacy theorem, or host compact-support solver.

Proof. Read the carrier of Definition 6454.1. The only Hankel source is the inherited finite-window row M, and its scalar entries are sealed by the real readback row R.

The endpoint and polynomial tests used by a Hausdorff read are finite. They pass through P and V before the packet opens the Hankel row Q and the localizing endpoint row L.

Finally S, T, C, G, N preserve the same route for transport, replay, provenance, compact-interval handoff, and naming. Because no carrier row stores an infinite operator, representing measure, selected support, determinacy theorem, or ambient solver, the displayed finite positivity route is the whole exported surface. □

Theorem 6454.4 (Hausdorff moment problem compact interval handoff)**.** *Every compact-support read of an accepted* $\mathsf{HausdorffMomentProblem}^{\uparrow}$ *packet is handed off through the displayed* $\mathsf{CompactRealInterval}^{\uparrow}$ *row before any moment or polynomial positivity ledger is consumed. The read must expose the support row I, the endpoint* $\mathsf{Real}^{\uparrow}$ *seals in R, the finite* $\mathsf{RealPolynomial}^{\uparrow}$ *test row P, the* $\mathsf{PositiveCone}^{\uparrow}$ *nonnegative row V, the Hankel and localizing ledgers Q, L, and the compact-interval handoff ledger G before a Hausdorff moment consumer may use the packet.*

Proof. Project the carrier. The support row I is the only compact interval coordinate, and R is the only endpoint readback coordinate available to the packet.

Any support-facing polynomial test therefore enters through I, R, P, V before it can reach Q or L. The handoff row G records that the same compact interval row, not an ambient compactness theorem, supplies the support boundary.

Transport, replay, provenance, and local naming through S, T, C, N preserve this displayed handoff. A consumer that skips I, R, or G is not reading the $\mathsf{HausdorffMomentProblem}^{\uparrow}$ carrier, and a consumer that adds a representing measure or determinacy proof leaves the named packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMomentProblem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffMomentProblem}^{\uparrow}$ packet is generated from a MomentProblemUp finite moment-window row, a CompactRealIntervalUp support row, RealUp endpoint and scalar readbacks, RealPolynomialUp finite polynomial tests, PositiveConeUp nonnegative rows, finite Hankel and localizing positivity ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, compact-interval handoff, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffMomentProblem}^{\uparrow}}$, seed)

Scope closed: Definition 6454.1, Definition 6454.2, Definition 6454.3, and Definition 6454.4 seed the finite Hausdorff moment route over MomentProblemUp, CompactRealIntervalUp, RealPolynomialUp, PositiveConeUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HausdorffMomentProblem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a representing measure, determinacy, uniqueness, selected support, infinite Hausdorff sequence, ambient compactness theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Hankel positivity routing, localizing endpoint positivity, compact-interval handoff, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffMomentProblemUp.

Chapter 6455

A Concrete Naming Certificate for $\mathsf{UniformDyadicMesh}^{\uparrow}$

Following Bishop and Bridges-style dyadic approximation, the $\mathsf{UniformDyadicMesh}^{\uparrow}$ packet records a finite uniform dyadic mesh certificate for the open Real and Completion route. It sits after the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the visible Real seal of chapter 38, and the downstream dyadic cover surface of chapter 1428. The packet names a uniform finite mesh and its readback route; it is not an arbitrary open cover, a selected completion limit, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6455.1 (Uniform dyadic mesh carrier)**.** *A* $\mathsf{UniformDyadicMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, R, J, \rho, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the dyadic lower-endpoint row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *R is the dyadic upper-endpoint row, read through the same dyadic arithmetic core and paired with L by the local interval classifier.*
3. *J is the mesh index row. It names the unary depth or finite dyadic scale at which all cells in the packet have the same dyadic length.*
4. *ρ is the radius ledger. It records the positive dyadic tolerance associated with each mesh cell and the finite containment checks visible at scale J.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that schedules endpoint, mesh, radius, and containment reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that packages the scheduled dyadic endpoint and radius data as regular rational observations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the endpoint rows, mesh index, radius ledger, finite window, and regular rational readback have been displayed.*
8. *H stores componentwise* hsame *transport for endpoints, mesh index, radius ledger, StreamName windows, RegSeqRat readback, Real seal, continuation replay, provenance, and local naming.*

9. *C stores* Cont *replay from endpoint reads to the uniform mesh row, from mesh and radius reads to regular rational readback, and from that readback to the Real-facing seal and dyadic interval-cover handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformDyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformDyadicMesh}^{\uparrow}}$ *row.*

The classifier compares accepted mesh packets only through the displayed tuple $L, R, J, \rho, S, Q, E, H, C, P, N$. It has no coordinate for nonuniform cell choice, arbitrary open-cover data, selected completion limits, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6455.2 (Uniform dyadic mesh refinement stability)**.** *Let $U = (L, R, J, \rho, S, Q, E, H, C, P, N)$ and $U' = (L', R', J', \rho', S', Q', E', H', C', P', N')$ be accepted* $\mathsf{UniformDyadicMesh}^{\uparrow}$ *carriers. Any displayed refinement from U to U' is stable exactly when the endpoint rows L, R transport to L', R', the mesh index row J' refines J by a finite dyadic scale comparison, the radius ledger ρ' is obtained by finite subdivision of ρ, and the rows S, Q, E, H, C, P, N replay the same StreamName, RegSeqRat, Real seal, transport, continuation, provenance, and local NameCert route. Refinement therefore changes only displayed dyadic mesh rows and never imports nonuniform cell choice, selected subcovers, quotient real equality, countable choice, or ambient completeness.*

Proof. Project the two carrier tuples from Definition 6455.1. Endpoint compatibility is a finite $\mathsf{DyadicRatCore}^{\uparrow}$ comparison on L, R, L', R', and the refinement of J is a displayed scale comparison rather than a host-level subdivision principle.

The radius ledger is then read rowwise. Each cell of the refined packet is attached to a finite dyadic radius in ρ', and the containment check is the replay of the corresponding coarser row in ρ through the displayed continuation route.

Finally, S, Q, E, H, C, P, N preserve the same public route. The stream row schedules the finite reads, the regular rational row packages them, the Real seal is downstream from those reads, and the structural rows transport, replay, source, and name only the displayed mesh data. Thus refinement stability is a finite row statement over the carrier surface. □

Theorem 6455.3 (Uniform dyadic mesh Real-window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *consumer reading an accepted* $\mathsf{UniformDyadicMesh}^{\uparrow}$ *carrier must first pass through*

$$L, R \longmapsto J, \rho \longmapsto S \longmapsto Q \longmapsto E$$

and then through H, C, P, N. The exported handoff is the finite tuple $L, R, J, \rho, S, Q, E, H, C, P, N$: dyadic endpoint rows, a uniform mesh index, a radius ledger, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise transport, displayed continuation replay, provenance, and the local NameCert row.

Proof. Start with a consumer asking for a real-facing mesh window. The lower and upper dyadic endpoints are the first visible data, and the carrier gives no real name until those endpoint rows have been admitted through $\mathsf{DyadicRatCore}^{\uparrow}$.

Read the mesh index and radius ledger next. The row J fixes a single dyadic scale for the packet, while ρ records the finite tolerance and containment checks attached to that scale. These rows are the only data a dyadic interval-cover consumer may inspect before the finite readback rows appear.

The rows S and Q schedule and package the reads, and E seals the resulting regular rational observations as Real-facing data. Transport, replay, provenance, and naming through H, C, P, N

keep the handoff inside the same finite certificate surface, so no arbitrary open cover, selected limit, quotient equality, countable choice, or ambient completeness principle is exported. □

Falsifiable prediction: Every $\mathsf{UniformDyadicMesh}^{\uparrow}$ consumer read factors through dyadic endpoint rows, a uniform mesh index, a radius ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before dyadic cover or completion data is visible.

Independence witness: dyadicratcore, streamname, regseqrat, real, dyadicintervalcover: $\mathsf{UniformDyadicMesh}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, a single mesh-scale row, a radius ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformDyadicMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformDyadicMesh}^{\uparrow}$ packet is generated from dyadic endpoint rows, a uniform mesh index, a radius ledger, finite StreamName windows, RegSeqRat readbacks, a Real seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformDyadicMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6455.1, Definition 6455.2, and Definition 6455.3 seed the finite uniform dyadic mesh handoff between dyadic arithmetic, regular rational readback, Real windows, and dyadic interval-cover consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformDyadicMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, selected completion limits, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, endpoint and mesh readback, radius-ledger stability, StreamName and RegSeqRat scheduling, Real seal handoff, dyadic interval-cover compatibility, and ledger non-escape for BEDC.Derived.UniformDyadicMeshUp.

Chapter 6456

A Concrete Naming Certificate for $\mathsf{BishopCompletionOperator}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{BishopCompletionOperator}^{\uparrow}$ packet names the finite operator route that turns displayed regular rational Cauchy data into a Real-facing completion seal. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name boundary of chapter 38. The packet records only the operator rows needed by the L10 completion route; it does not quotient Cauchy names, select representatives, use countable choice, or assert that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 6456.1 (BishopCompletionOperator carrier)**.** *A* $\mathsf{BishopCompletionOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, R, D, E, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observed Cauchy-name windows.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for every precision request.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D have been exposed.*
5. *U is the Bishop completion extension and universal-boundary row, recording the finite handoff by which a completion consumer reads the operator output.*
6. *H records componentwise* hsame *transport for stream windows, regular readbacks, dyadic tolerances, Real seals, extension rows, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through S, R, D, then through E and U.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionOperator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionOperator}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy equality, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6456.2 (BishopCompletionOperator NameCert obligations)**.** *For every accepted* BishopCompletionOpera *carrier* $O = (S, R, D, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionOperator}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window exposure, regular rational readback, dyadic tolerance exposure, ordered* $\mathsf{Real}^{\uparrow}$ *seal access, Bishop extension and universal-boundary exposure through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A public completion-operator read must pass through those rows before it may cite the Real-facing seal or the universal-boundary row.*

Proof. Project the accepted finite BHist packet. The visible carrier rows are precisely $S, R, D, E, U, H, C, P, N$, so carrier admission exposes stream windows, regular readback, dyadic tolerance data, the ordered Real seal, and the extension boundary as displayed data.

The operator read is ordered. A precision request first reaches the finite $\mathsf{StreamName}^{\uparrow}$ windows, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and only afterward the $\mathsf{Real}^{\uparrow}$ seal and the universal-boundary row U.

Transport and replay are structural. The row H moves the listed coordinates componentwise by hsame, while C replays the displayed continuation route and P, N source and name the same packet. Since no quotient Cauchy equality, selected representative, countable-choice schedule, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness coordinate appears in the tuple, the listed rows exhaust the local NameCert obligations. □

Theorem 6456.3 (BishopCompletionOperator Real-seal factorization)**.** *Every* $\mathsf{BishopCompletionOperator}^{\uparrow}$ *consumer that reads the Real-facing output of an accepted carrier* $O = (S, R, D, E, U, H, C, P, N)$ *factors through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto E \longmapsto U \longmapsto H, C, P, N.$$

Thus the operator output reaches $\mathsf{Real}^{\uparrow}$ *only through finite stream windows, regular rational readback, dyadic tolerances, the displayed seal row, and the Bishop completion extension boundary. It does not export quotient Cauchy equality, selected representatives, countable choice, host real equality, or any claim that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with a consumer request for the operator output. The carrier definition gives the request access to the finite window row S, so the request begins with displayed Cauchy-name observations rather than an external Cauchy quotient.

Next read the regular and dyadic rows. The readback row R and tolerance row D supply the regular rational and precision data used to reach the Real-facing seal E.

Finally read the extension boundary U and replay the structural rows H, C, P, N. The route is therefore the displayed finite chain from stream windows to the Real seal and universal boundary. Because the packet contains no additional coordinate, the factorization cannot produce quotient Cauchy equality, selected representatives, countable choice, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionOperator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCompletionOperator}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, a Bishop extension and universal-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionOperator}^{\uparrow}}$, seed)

Scope closed: Definition 6456.1, Definition 6456.2, and Definition 6456.3 seed the finite non-quotient completion-operator route from stream windows and regular rational readback to the Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionOperator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected representatives, countable choice, host real equality, Lean verification, bridge checking, or any assertion that $\mathsf{Real}^{\uparrow}$ is complete.

Upgrade path: Obligation closure requires checked carrier admission, ordered Real-seal factorization, universal-boundary exposure, transport stability, and non-escape for BEDC.Derived.BishopCompletionOperatorUp.

Chapter 6457

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$ packet records the constructive completeness surface for located real names as a finite NameCert route. It sits after the located-real comparison of **??**, the regular rational readback of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet records which displayed Cauchy source data may be read before a completion-facing $\mathsf{Real}^{\uparrow}$ seal is exposed; it does not assert ambient Real completeness, quotient real equality, selected limits, trichotomy, or countable choice.

Definition 6457.1 (Bishop located real completeness carrier)**.** *A* $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the displayed located Cauchy source row, recording the admitted source sequence or modulus request before any completion-facing seal is read.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which source observations are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the same finite windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for source-window comparison, regular readback, and final seal budgeting.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, exposing located interval or apartness comparisons available to the completion request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, R, D, L have been displayed.*
7. *H is the componentwise* hsame *transport row for the source, window, readback, tolerance, located-comparison, seal, continuation, provenance, and naming data.*
8. *C is the displayed* Cont *replay row from located Cauchy source reads through finite windows, regular readback, dyadic tolerance, located comparison, and the terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, R, D, L, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected limit operator, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedRealCompletenessUp.BishopLocatedRealCompletenessTasteGate`
`single carrier alignment`

Theorem 6457.2 (Bishop located real completeness NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$ *carrier $K = (S, W, R, D, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealComple}}$ *surface consists exactly of carrier admission for the displayed located Cauchy source S, finite* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row L, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N.*

Proof. Project the carrier of Definition 6457.1. The rows $S, W, R, D, L, E, H, C, P, N$ are the whole public packet, so carrier admission and local naming are finite row checks.

The source route is read in order. The displayed located Cauchy source S opens only finite windows W; those windows produce the regular rational row R under the dyadic budget D; the located comparison row L is inspected before the terminal seal E is available.

The structural rows preserve the same route. Componentwise hsame transport and displayed Cont replay keep S, W, R, D, L, E attached to the same packet, while P, N only source and name the data. Since the carrier has no row for quotient equality, selected limits, host equality, trichotomy, countable choice, or ambient completeness, no local obligation can read such data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealCompleteness}^{\uparrow}$ packet is generated from a displayed located Cauchy source, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a LocatedReal comparison row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompleteness}^{\uparrow}}$, seed)

Scope closed: Definition 6457.1 and Definition 6457.2 seed the finite located-real completeness route from displayed Cauchy source data through regular rational readback to a terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompleteness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, host real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located Cauchy source routing, finite window coverage, RegSeqRat readback, DyadicRatCore tolerance exactness, LocatedReal comparison, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealCompletenessUp.

Chapter 6458

A Concrete Naming Certificate for PointwiseEquicontinuityCompactTransfer$^{\uparrow}$

Following Bishop and the Arzela–Ascoli compact equicontinuity route, the PointwiseEquicontinuityCompactTransfer$^{\uparrow}$ packet records the finite handoff from pointwise family moduli on a compact metric source to a uniform-modulus consumer. It sits after the compact source of chapter 129, the continuous-map rows consumed by chapter 130, the family-modulus surface of chapter 413, and the uniform-modulus boundary of chapter 369. The packet is a finite compact-net and family-radius transfer; it is not a selected global modulus function, open-cover compactness theorem, function-space compactness assertion, countable-choice schedule, or ambient equality of maps.

Definition 6458.1 (Pointwise equicontinuity compact transfer carrier)**.** *A* PointwiseEquicontinuityCompactTransfer *carrier is a finite* BHist *packet*

$$T = (K, F, E, B, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing only the finite compact-net and metric inspection data used by the transfer.*
2. *F is the finite* ContinuousMap$^{\uparrow}$ *family ledger whose graph rows are inspected by the equicontinuity request.*
3. *E is the* EquicontinuityFamily$^{\uparrow}$ *pointwise family-modulus row over the same compact source and target metric data.*
4. *B is the finite* ProbeBundle *cover row of compact-net centers.*
5. *R is the* Rat$^{\uparrow}$ *radius and tolerance fold ledger that lowers the pointwise family radii over the finite bundle.*
6. *U is the* UniformModulus$^{\uparrow}$ *handoff row receiving the tolerance-only family modulus extracted from K, F, E, B, R.*
7. *H is the componentwise* hsame *transport row for compact source, continuous-map, equicontinuity, bundle, rational fold, uniform-modulus, replay, provenance, and naming data.*
8. *C is the displayed* Cont *replay from compact finite-net reads and pointwise family-modulus rows to the uniform-modulus handoff.*

9. *P is the* Pkg *provenance over* PointwiseEquicontinuityCompactTransfer$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{PointwiseEquicontinuityCompactTransfer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, E, B, R, U, H, C, P, N$. It has no coordinate for a selected global modulus, an arbitrary open-cover compactness theorem, function-space compactness, host equality of maps, countable choice, or an external family selector.

Theorem 6458.2 (Pointwise equicontinuity compact transfer NameCert obligations)**.** *For every accepted* PointwiseEquicontinuityCompactTransfer$^{\uparrow}$ *carrier $T = (K, F, E, B, R, U, H, C, P, N)$, the local* NameCert$_{\text{PointwiseEquicontinuityCompactTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the* CompactMetric$^{\uparrow}$ *source row K, finite* ContinuousMap$^{\uparrow}$ *family ledger F,* EquicontinuityFamily$^{\uparrow}$ *pointwise modulus row E, finite* ProbeBundle *compact-net row B,* Rat$^{\uparrow}$ *fold ledger R,* UniformModulus$^{\uparrow}$ *handoff row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N.*

Proof. Project the carrier of Definition 6458.1. The rows $K, F, E, B, R, U, H, C, P, N$ are the whole packet, so carrier admission and local naming are finite row checks.

The transfer route is finite. The compact source row K supplies the finite ProbeBundle cover B; the family ledger F and pointwise equicontinuity row E attach a displayed local radius to each inspected center; the Rat$^{\uparrow}$ fold R lowers those radii over B before the uniform-modulus row U is consumed.

Transport and replay remain componentwise. The rows H, C, P, N keep the compact-net, family-modulus, rational-fold, and uniform-modulus data inside one BHist packet. Since no carrier coordinate stores a selected global modulus, open-cover theorem, function-space compactness object, host map equality, countable-choice schedule, or external family selector, no local NameCert obligation can export such data. □

Closure status for NameCert$_{\text{PointwiseEquicontinuityCompactTransfer}^{\uparrow}}$.

Constructive story: A PointwiseEquicontinuityCompactTransfer$^{\uparrow}$ packet is generated from a CompactMetric finite-net source, ContinuousMap family rows, an EquicontinuityFamily pointwise-modulus ledger, a finite ProbeBundle cover, RatUp radius-fold data, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PointwiseEquicontinuityCompactTransfer}^{\uparrow}}$, seed)

Scope closed: Definition 6458.1 and Definition 6458.2 seed the finite compact-net transfer from pointwise equicontinuity rows to a uniform-modulus handoff.

Formal verification: FormalStatus(NameCert$_{\text{PointwiseEquicontinuityCompactTransfer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected global modulus functions, arbitrary open-cover compactness, function-space compactness, countable choice, host equality of maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, family-modulus readback, RatUp fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.PointwiseEquicontinu

Chapter 6459

A Concrete Naming Certificate for $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ packet records the finite criterion by which a regular rational Cauchy presentation may be read as convergence-ready data before any real-name seal is consumed. It sits over the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a Bishop-style convergence criterion for the RegSeqRat-to-Real route; it is not a quotient class of Cauchy streams, a selected limit, countable choice, host real completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6459.1 (Bishop Cauchy convergence criterion carrier)**.** *A* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the inspected finite observation windows and precision requests.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, naming the regular rational data read from the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying requested radii, split tolerances, comparison budgets, and endpoint observations.*
4. *R is the convergence-criterion row, recording that the Q readback is controlled by the finite dyadic tolerance surface in D along the windows in S.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after S, Q, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the window, readback, tolerance, criterion, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S \longmapsto Q \longmapsto D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected real limit, countable choice, host completeness, detached convergence, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BishopCauchyConvergenceCriterionUp.BishopCauchyConvergenceCriterionTasteGate`
`single carrier alignment`

Theorem 6459.2 (Bishop Cauchy convergence criterion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ *carrier*

$$K = (S, Q, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}$ *surface consists exactly of five finite obligation families:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, convergence-criterion row, Real seal, transport, replay, provenance, and local naming row;*
2. *window-readback exactness, requiring Q to read only the finite windows displayed by S;*
3. *dyadic tolerance coverage, requiring D to display the requested radii, split tolerances, comparison budgets, and endpoint observations before R is consumed;*
4. *convergence nonescape, requiring R to bind the same S, Q, D rows rather than a quotient Cauchy class, selected limit, countable-choice functional, or host convergence theorem;*
5. *Real seal discipline, requiring E to be downstream of S, Q, D, R while H, C, P, N transport, replay, source, and name only the displayed route.*

No Bishop Cauchy convergence criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6459.1. The accepted packet displays exactly the window row, regular readback, dyadic tolerance ledger, convergence row, Real seal, and structural rows.

Window-readback exactness is read from S and Q. The regular rational row is attached to the finite windows, so no quotient stream representative is introduced.

Dyadic tolerance coverage is the role of D. Its requested radii, split tolerances, comparison budgets, and endpoint observations are displayed before the convergence row can be consumed.

The convergence condition is local to R. Since R binds the same S, Q, D route, a consumer receives finite criterion data rather than a selected limit, external choice function, or host convergence theorem.

Finally, E, H, C, P, N add only the terminal seal, transport, replay, provenance, and local naming. The carrier has no coordinate for quotient stream equality, countable choice, detached convergence, host completeness, or ambient completeness, so the listed obligation families exhaust the local NameCert surface. □

Theorem 6459.3 (Bishop Cauchy convergence criterion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto Q \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The handoff reads StreamName windows, RegSeqRat regular rational data, DyadicRatCore tolerance coverage, the convergence-criterion row, and the terminal Real seal in that order. It exports no selected limit, quotient equality of Cauchy streams, countable choice, detached convergence theorem, host real completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Proof. Begin with a real-facing or completion-facing read of an accepted packet. Projection exposes the ordered carrier rows $S, Q, D, R, E, H, C, P, N$.

The readback row Q is downstream of the finite windows S, and the dyadic tolerance row D is downstream of the same readback. Thus the consumer cannot start from an unbounded stream representative or a quotient equality class.

The NameCert obligation theorem places the convergence row R after the displayed windows, regular readback, and dyadic tolerance coverage. Hence the Cauchy-to-real route receives convergence criterion data before it can reach E.

The terminal row E is the only $\mathsf{Real}^{\uparrow}$ seal coordinate in the packet. The structural rows preserve and name the same finite route, and the excluded principles are absent from the carrier. Therefore every real-facing or completion-facing consumer factors through the displayed handoff and receives none of the listed escape objects. □

Theorem 6459.4 (Bishop Cauchy convergence criterion tail stability)**.** *Transporting an accepted* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ *packet along componentwise* hsame *preserves the finite tail request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, convergence-criterion row, and Real seal handoff. Consequently every* $\mathsf{Real}^{\uparrow}$*-facing consumer still reads the same finite tail-window route before it can consume the terminal Real seal. The transport exports no selected limits, quotient equality of Cauchy streams, countable choice, detached convergence, host Real completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 6459.1. The transport row H is componentwise over the StreamName window row S, RegSeqRat readback row Q, DyadicRatCore tolerance row D, convergence row R, Real seal E, replay, provenance, and local naming rows.

Apply Definition 6459.2. Window-readback exactness, dyadic tolerance coverage, convergence nonescape, and Real seal discipline are obligations on the displayed rows themselves, so componentwise hsame cannot replace them with a selected limit, quotient stream class, countable-choice functional, detached convergence theorem, host Real completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally use Definition 6459.3. The Real-facing handoff is downstream of $S, Q, D, R, E, H, C, P, N$ in that order; after transport, the consumer therefore reads the same finite tail-window route before the terminal Real seal is exposed. □

Theorem 6459.5 (Bishop Cauchy convergence criterion obligation-closure package)**.** *For every accepted* $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ *carrier* $K = (S, Q, D, R, E, H, C, P, N)$*, the carrier definition, NameCert obligation theorem, Real handoff theorem, and tail-stability theorem assemble the* obligation *surface over* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the convergence-criterion row, the* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}$ *row. The assembled package exports no selected limit, quotient stream equality, countable choice, detached convergence theorem, host real completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from the carrier admission of Definition 6459.1. It fixes the only readable rows as $S, Q, D, R, E, H, C, P, N$, so every package claim is made over the displayed finite route.

Apply Definition 6459.2. The obligation families expose window-readback exactness, dyadic tolerance coverage, convergence nonescape, Real seal discipline, and the structural transport, replay, provenance, and naming rows.

Use Definition 6459.3 for the consumer-facing direction. It forces every $\mathsf{Real}^{\uparrow}$ or completion-facing read through $S, Q, D, R, E, H, C, P, N$ before the terminal seal can be consumed.

Finally apply Definition 6459.4. Componentwise hsame transport preserves the same tail request, observation schedule, readback, tolerance, convergence, and Real-seal rows, so the package is stable under the local classifier. Since none of the cited rows contains a selected limit, quotient stream equality, countable choice, detached convergence, host real completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, the assembled surface is exactly obligation-closed for this finite packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ real-facing read factors through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance coverage, a convergence-criterion row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite observation windows, regular rational readback, dyadic tolerance coverage, an explicit convergence-criterion row, terminal real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a convergence criterion row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6459.1, Definition 6459.2, Definition 6459.3, and Definition 6459.4 seed the finite convergence route through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, the convergence row, Real seal, transport, continuation, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, detached convergence, host real completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, dyadic tolerance coverage, convergence nonescape, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyConverg

Chapter 6460

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ packet records the finite uniform-limit route for a sequence of regular Cauchy real-name packets before any completion-facing consumer may read a terminal real seal. It sits over the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the convergence-criterion route of chapter 1509. The packet names a constructive uniform-limit handoff; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host function equality, selected limits, or countable choice.

Definition 6460.1 (Regular Cauchy uniform-limit carrier)**.** *A* $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence row, listing the accepted regular Cauchy real-name packets whose values are available only through displayed finite histories.*
2. *M is the uniform Cauchy modulus row, giving one finite precision schedule that controls the tail of the whole sequence.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic radii, split budgets, and comparison rows consumed by M.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the sequence indices, value names, modulus requests, and terminal read positions.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to the regular rational observations required by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rows S, M, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the sequence, modulus, tolerance, window, readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform-limit request through sequence lookup, modulus inspection, dyadic tolerance, finite windows, regular rational readback, and real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, M, D, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable-choice schedules, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6460.2 (Regular Cauchy uniform-limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ *carrier*

$$U = (S, M, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows: carrier admission for the displayed regular Cauchy sequence, uniform modulus, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local name; uniform-modulus exactness through M and D; finite-window readback through W and R; convergence-criterion compatibility through the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *route named by P; terminal Real-seal discipline through E; and componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support.*

Proof. Project the carrier of Definition 6460.1. The rows $S, M, D, W, R, E, H, C, P, N$ are the only available coordinates, so carrier admission and local naming are finite row projections.

The uniformity row is M, and its tolerance budget is the dyadic row D. Any precision request must read those rows before a tail comparison can be consumed, so the route is uniform over the displayed sequence rather than selected separately for each index.

The observation rows are W and R. They force all inspected values through StreamName windows and RegSeqRat readbacks before the terminal seal E is available.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no quotient, selector, choice, host-function, or ambient-completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 6460.3 (Regular Cauchy uniform-limit Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CauchyConvergenceCriterion}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff reads the sequence row, a displayed uniform Cauchy modulus, dyadic tolerance rows, finite StreamName windows, RegSeqRat readbacks, the convergence-criterion route, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before the terminal real seal is consumed. It exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limit, countable choice, or host function equality.*

Proof. Begin with the obligation theorem Definition 6460.2. It places the uniform modulus and dyadic tolerance rows before any real-facing read.

The sibling route then fixes the finite observation order. StreamName supplies the windows, RegSeqRat supplies the rational readback rows, and DyadicRatCore supplies the tolerance comparisons needed by the uniform modulus.

The convergence-criterion row named in the provenance keeps the handoff on the regular Cauchy route before the terminal $\mathsf{Real}^{\uparrow}$ seal is read. Transport and replay through H and C preserve the same packet, while P, N source and name it.

Because none of the excluded principles is present in the carrier, the consumer receives only the displayed finite route and not an ambient completeness or quotient-function endpoint. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ real-facing read exposes regular Cauchy sequence data, one uniform modulus row, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readbacks, a convergence-criterion route, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchyconvergencecriterion: none is bijective to $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ because this packet alone binds a sequence of regular Cauchy real-name packets to one displayed uniform modulus and a terminal RealUp handoff without quotient streams, selected limits, countable choice, host function equality, or ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$ packet is generated from regular Cauchy real-name sequence rows, one uniform Cauchy modulus row, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readbacks, a convergence-criterion route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6460.1, Definition 6460.2, and Definition 6460.3 seed the finite regular-Cauchy uniform-limit route through uniform modulus, dyadic tolerance, StreamName windows, RegSeqRat readback, convergence criterion, transport, replay, provenance, local naming, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient stream equality, selected limits, countable choice, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-modulus exactness, dyadic tolerance coverage, finite-window readback, convergence-criterion compatibility, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyUniformLimitUp.

Chapter 6461

A Concrete Naming Certificate for $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$ packet records the finite tail-tolerance criterion by which two regular rational names are compared before a $\mathsf{Real}^{\uparrow}$ consumer reads an equality boundary. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names equality by displayed dyadic radii and cofinal regular-sequence tails, not by quotient classes, selected representatives, trichotomy, or choice.

Definition 6461.1 (Regular Cauchy equality criterion carrier)**.** *A* $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened for the compared regular Cauchy names.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, binding each opened window to rational observations.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the displayed radii, half-radius splits, and comparison cells used by the test.*
4. T *is the common finite tail row, recording that both regular names are read past the displayed tail stage for the same dyadic tolerance request.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *equality-boundary read, available only after the two windows, the two regular readbacks, the dyadic ledger, and the common tail row have been displayed.*
6. H *records componentwise* hsame *transport for the two windows, two readbacks, dyadic ledger, tail row, equality boundary, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the route*

$$S_0, S_1, R_0, R_1, D \longmapsto T \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$. It has no coordinate for quotient equality of Cauchy names, a selected representative, real trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyEqualityCriterionUp`

Theorem 6461.2 (Regular Cauchy equality criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$ *carrier*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, two StreamName window exposures, two RegSeqRat readbacks, one DyadicRatCore tolerance ledger, one common tail row, ordered* $\mathsf{Real}^{\uparrow}$ *equality-boundary access, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. A consumer that reads equality of the two regular Cauchy names must pass through those rows before it may cite the Real-facing equality boundary.*

Proof. Project the accepted finite BHist packet. Carrier admission exposes exactly the two windows, two regular readbacks, dyadic tolerance ledger, common tail row, equality-boundary row, transport row, replay row, provenance row, and local naming row.

The two readback sides are symmetric but not quotient data. The rows S_0, R_0 and S_1, R_1 provide only displayed finite windows and regular rational observations.

The dyadic and tail stages are ordered. The row D supplies the displayed radii and comparison cells, and T records the common tail stage at which both regular names are read for that request.

The equality-boundary row E is downstream of S_0, S_1, R_0, R_1, D, T. Since the remaining rows H, C, P, N only transport, replay, source, and name the same route, the listed rows exhaust the local NameCert surface. □

Theorem 6461.3 (Regular Cauchy equality criterion dyadic tail route)**.** *Let*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads the equality criterion for the two regular Cauchy names factors through the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently the criterion routes through displayed dyadic radii and regular-sequence tails: the consumer receives the two StreamName windows, two RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the common tail row, the equality-boundary row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data. It receives no quotient class of Cauchy names, selected representative, trichotomy decision, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier and project the two window rows S_0, S_1. These are the only observation windows available to a consumer of the compared names.

Next read the two regular rational rows R_0, R_1. They bind the observations to regular-sequence data for the same consumer request, so the route has not introduced an external representative or quotient object.

The dyadic tolerance row D is then read before the common tail row T. This makes the radius and the tail stage displayed together, with the tail request determined by the same finite ledger.

Finally read the equality-boundary row E and the structural rows H, C, P, N. The route therefore reaches any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ equality consumer only through the displayed dyadic radii and regular-sequence tails, and the carrier has no coordinate from which the excluded quotient, selection, order, choice, host-equality, or completeness principles could be recovered. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readback rows, one DyadicRatCore tolerance ledger, one common tail row, a RealUp equality-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6461.1, Definition 6461.2, and Definition 6461.3 seed the finite tail-tolerance equality route for regular Cauchy names over displayed dyadic radii.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.RegularCauchyEqualityCriterionUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy names, selected representatives, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired readback exactness, dyadic tolerance exposure, common-tail exactness, ordered equality-boundary access, and structural stability for BEDC.Derived.RegularCauchyEqualityCriterionUp

Chapter 6462

A Concrete Naming Certificate for $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$

Following Bishop and Bridges on compact intervals, finite coverings, and fan-theorem style moduli, the $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ packet records the bounded binary fan/bar modulus surface needed by L10 interval-cover routes. It sits after the located interval-cover packet of chapter 1885, the dyadic interval-cover packet of chapter 1428, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite modulus certificate for located interval rows; it does not assert open-cover compactness, countable choice, bar induction outside the displayed finite tree, or an ambient real completeness principle.

Definition 6462.1 (Located interval fan modulus carrier)**.** *A* $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, F, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row listing the located interval requests, endpoint comparisons, and cover-membership reads exposed to the L10 route;*
2. *F is the binary fan/bar modulus row, a finite tree of binary prefixes with a displayed bar level and a modulus bound for the accepted interval requests;*
3. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *window row, recording the dyadic subintervals and mesh refinements read by the fan row;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, listing the finite source windows opened before a branch prefix or interval endpoint is inspected;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream windows, dyadic endpoint comparisons, and located interval decisions;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the rows L, F, D, W, Q have been displayed;*
7. *H records componentwise* hsame *transport for located interval rows, fan/bar rows, dyadic windows, stream schedules, rational readbacks, real seals, replay routes, provenance, and local naming;*
8. *C records displayed* Cont *replay from an L10 interval-cover request through the finite fan/bar row and then through D, W, Q, E;*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, F, D, W, Q, E, H, C, P, N$. It has no coordinate for an open-cover compactness oracle, countable choice, a branch selected outside the finite fan, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6462.2 (Located interval fan modulus NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ *carrier*

$$M = (L, F, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located interval-cover admission through L, binary fan/bar modulus admission through F, dyadic cover-window admission through D, stream-schedule admission through W, regular rational readback through Q, terminal real-seal admission through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6462.1. The displayed rows are exactly $L, F, D, W, Q, E, H, C, P, N$, so every local obligation is attached to one of those rows.

The located interval-cover row is read before the fan row, and the fan row is read before the dyadic window, stream schedule, rational readback, and real seal. Thus a modulus claim cannot skip from an interval request directly to a real conclusion.

The structural rows H, C, P, N transport, replay, package, and name the same finite route. Since no excluded coordinate is present in the carrier, the local NameCert surface cannot call on open-cover compactness, countable choice, an off-packet branch selector, quotient real equality, or ambient real completeness. □

Theorem 6462.3 (Located interval fan modulus route)**.** *Every L10 interval-cover consumer of an accepted* $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedIntervalCover}^{\uparrow},\ \mathsf{FanFunctional}^{\uparrow},\ \mathsf{DyadicIntervalCover}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In carrier notation this means that every exported modulus first reads the located interval row L, the binary fan/bar row F, the dyadic cover window D, the stream schedule W, the rational readback Q, the terminal real seal E, and the structural rows H, C, P, N. The route exports no open-cover compactness oracle, countable-choice selector, off-packet infinite branch, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an L10 interval-cover request. By Definition 6462.1, the request enters the packet through L, where the located interval rows and cover-membership reads are displayed.

The fan/bar modulus row F can be inspected only after that located interval surface is present. Its binary prefixes are then compared against the dyadic window row D and the finite stream schedule W.

The rational readback Q and real seal E are downstream of those finite rows. Finally H, C, P, N preserve the same route under transport, replay, provenance, and naming. Hence every consumer-facing modulus is bounded by the displayed finite packet. □

Falsifiable prediction: A purported L10 interval-cover modulus that reads a $\mathsf{Real}^{\uparrow}$ conclusion without displaying the $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows refutes the $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ carrier.

Independence witness: locatedintervalcover, dyadicintervalcover, fanfunctional, streamname, regseqrat, real: $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ binds the located interval row, binary fan/bar modulus row, dyadic cover window, stream schedule, regular rational readback, real seal, and structural transport rows in one finite NameCert packet; each named sibling lacks at least one of these rows as a local carrier obligation.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalFanModulus}^{\uparrow}$ packet is generated from located interval-cover rows, a finite binary fan/bar modulus row, dyadic cover windows, stream schedules, regular rational readbacks, a terminal real seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalFanModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6462.1, Definition 6462.2, and Definition 6462.3 bind the finite fan/bar modulus route for L10 located interval-cover consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalFanModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, bar induction outside the displayed finite tree, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: To reach obligationClosure, the located interval-cover row, binary fan/bar row, dyadic window, stream schedule, rational readback, real seal, and route obligations must be encoded as checked finite carrier obligations.

Chapter 6463

A Concrete Naming Certificate for $\mathsf{TukeyOrder}^{\uparrow}$

Following Tukey's theory of cofinal maps and the Moore–Smith reading of nets, the $\mathsf{TukeyOrder}^{\uparrow}$ packet records finite directed-comparison data for net and filter completion routes. It sits over the directed-system surface of chapter 1607, the Moore–Smith net surface of chapter 3403, the Cauchy-net route of chapter 589, the filter-base route of chapter 1601, and the Cauchy-filter route of chapter 333. The packet compares displayed presentations by cofinal response rows; it is not a quotient of nets, not a quotient of filters, not a selected subnet, not a selected limit, and not a countable-choice principle.

Definition 6463.1 (Tukey-order carrier)**.** *A* $\mathsf{TukeyOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, Q, \rho, \lambda, M, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source directed-system row, read through* $\mathsf{DirectedSet}^{\uparrow}$ *and the source* $\mathsf{MooreSmithNet}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *presentation.*
2. *Q is the target directed-system row, read through* $\mathsf{DirectedSet}^{\uparrow}$ *and the target* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *presentation.*
3. *ρ is the cofinal response map. For each displayed target request it names a source request and the finite containment or late-tail witness that answers the target request from the source route.*
4. *λ is the reverse comparison ledger when present. It records displayed return comparisons for two-way route comparison, but a one-way Tukey comparison may leave this row empty.*
5. *M is the net-side handoff row, naming the* $\mathsf{MooreSmithNet}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *windows consumed by the comparison.*
6. *F is the filter-side handoff row, naming the* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *packets consumed by the comparison.*
7. *H records componentwise* hsame *transport for the source system, target system, cofinal response map, reverse ledger, net handoff, filter handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from any target request through ρ and, when present, through λ.*

9. *P is the* Pkg *provenance over* $\mathsf{TukeyOrder}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TukeyOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, Q, \rho, \lambda, M, F, H, C, P, N$. It has no coordinate for quotient net equality, quotient filter equality, selected subnets, selected cofinal families, selected limits, countable choice, host real equality, or ambient completeness.

Theorem 6463.2 (Tukey-order NameCert obligations)**.** *For every accepted* $\mathsf{TukeyOrder}^{\uparrow}$ *carrier $T = (S, Q, \rho, \lambda, M, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TukeyOrder}^{\uparrow}}$ *surface consists exactly of source directed-system admission through S, target directed-system admission through Q, cofinal response coverage through ρ, optional reverse comparison coverage through λ, net-side handoff through M, filter-side handoff through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No Tukey-order obligation is stored outside those rows.

Proof. Project the carrier of Definition 6463.1. The accepted packet is exhausted by the two directed-system rows, the cofinal response map, the optional reverse ledger, the net and filter handoff rows, and the structural rows H, C, P, N.

The comparison route starts with a target request in Q. The row ρ answers it by naming a source request in S together with the displayed late-tail or containment witness needed by the source route. When a two-way comparison is required, λ supplies only finite return comparisons between the same displayed rows.

The net-side and filter-side handoffs do not add hidden comparison data. The rows M and F merely record which $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, and $\mathsf{CauchyFilter}^{\uparrow}$ presentations are being compared, while H, C, P, N transport, replay, source, and name the same finite packet. □

Theorem 6463.3 (Tukey-order net-filter handoff)**.** *Every net/filter completion consumer of an accepted* $\mathsf{TukeyOrder}^{\uparrow}$ *carrier factors through the finite directed-comparison route*

$$Q \xrightarrow{\rho} S \longmapsto M, F, H, C, P, N,$$

with λ inserted only when the consumer requests a displayed reverse comparison. Consequently a $\mathsf{MooreSmithNet}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *presentation can be compared with a* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *presentation without quotienting nets or filters, selecting subnets, selecting cofinal families, selecting limits, invoking countable choice, or importing ambient completeness.*

Proof. Begin with a completion-facing net/filter consumer. The carrier definition exposes the target directed-system row Q and the source directed-system row S before any comparison can be read.

Replay the consumer request through ρ. The cofinal response map turns each displayed target request into a displayed source request and records the required late-tail or containment witness. If the route asks for equivalence rather than one-way comparison, the reverse ledger λ supplies a finite list of return comparisons over the same visible rows.

Finally read the handoff rows M and F. They keep the consumer inside the named $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, and $\mathsf{CauchyFilter}^{\uparrow}$ presentations, while H, C, P, N preserve the same finite comparison packet. Since no excluded coordinate appears in the carrier, the handoff cannot export quotient equality, selected subnets, selected limits, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: A net/filter completion route that compares a Cauchy-net presentation with a Cauchy-filter presentation after deleting S, Q, ρ, M, or F refutes the carrier of Definition 6463.1.

Independence witness: directedset, moore-smith-net, cauchynet, filterbase, cauchyfilter: $\mathsf{TukeyOrder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source and target directed-system rows, a cofinal response map, an optional reverse comparison ledger, net-side and filter-side handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TukeyOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{TukeyOrder}^{\uparrow}$ packet is generated from source and target directed-system rows, a cofinal response map, an optional reverse comparison ledger, MooreSmithNet and CauchyNet handoffs, FilterBase and CauchyFilter handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TukeyOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6463.1, Definition 6463.2, and Definition 6463.3 seed the finite Tukey comparison surface for net and filter completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TukeyOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient net equality, quotient filter equality, selected subnets, selected cofinal families, selected limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source and target directed-system admission, cofinal response coverage, reverse comparison coverage, net and filter handoff exactness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.TukeyOrderUp.

Chapter 6464

A Concrete Naming Certificate for $\mathsf{SemidecidableOpen}^{\uparrow}$

Following Bishop and Bridges and the constructive-topology reading of observable opens, the $\mathsf{SemidecidableOpen}^{\uparrow}$ packet records positive open evidence as finite BEDC observation data. It sits over the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Sierpinski observable-open classifier of chapter 1860, and the indexed open-row discipline of chapter 88. The packet names the interface by which a positive open observation may be reported without importing host open sets, quotient topology, excluded middle over membership, selected witnesses, or ambient real completeness.

Definition 6464.1 (Semidecidable open carrier)**.** *A* $\mathsf{SemidecidableOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, T, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *classifier row, exposing the observable truth mark for a positive open report.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, recording the indexed open under inspection as displayed topology data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, listing the observed request prefixes available to the open report.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational witness data used when the open report is read by a real-name consumer.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, attaching the report to the displayed real-name boundary rather than to a host real.*
6. *H stores componentwise* hsame *transport for classifier, topology, window, rational readback, real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the positive observation request through S, T, W, Q, R, H.*
8. *P is the* Pkg *provenance over* $\mathsf{SemidecidableOpen}^{\uparrow}$*,* $\mathsf{SierpinskiSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SemidecidableOpen}^{\uparrow}}$ *row.*

The carrier has no coordinate for a negative open-membership decision, a host powerset open, quotient equality of opens, selected neighbourhood witnesses, countable choice, or a bridge credential.

Theorem 6464.2 (Semidecidable open NameCert obligations)**.** *For every accepted* $\mathsf{SemidecidableOpen}^{\uparrow}$ *carrier* $O = (S, T, W, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SemidecidableOpen}^{\uparrow}}$ *surface consists exactly of Sierpinski classifier admission, Topology open-row admission, StreamName finite-window readback, RegSeqRat rational witness readback, Real seal attachment, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A positive open consumer must read* S, T, W, Q, R *before it consumes* H, C, P, N*.*

Proof. Project the carrier of Definition 6464.1. The rows S and T supply the observable classifier and indexed open source; neither row is a host proposition about arbitrary subsets.

The rows W, Q, R make the positive evidence finite and real-facing. W records the observed StreamName windows, Q records the regular rational readback used by completion consumers, and R seals that readback at the displayed $\mathsf{Real}^{\uparrow}$ boundary.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Since the carrier has no negative-decision row, selected-neighbourhood row, host powerset row, quotient-open row, countable-choice row, or bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 6464.3 (Semidecidable open real observation route)**.** *Every real-facing positive open read exported by an accepted* $\mathsf{SemidecidableOpen}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{SierpinskiSpace}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Thus L10 Real and Completion consumers receive positive open evidence through finite observable windows and regular rational readback, not through host open sets, quotient topology, excluded middle over membership, selected witnesses, or ambient completeness.

Proof. Read $O = (S, T, W, Q, R, H, C, P, N)$. The Sierpinski row S classifies the positive observation before the topology row T supplies the indexed open source.

The StreamName row W then fixes the finite observation window. The RegSeqRat row Q supplies the rational readback attached to that window, and the Real row R seals the result at the displayed real-name boundary.

Transport, replay, provenance, and naming through H, C, P, N preserve this order. A real-facing consumer therefore receives only the finite positive-evidence route and cannot bypass it by importing a host open-set predicate, quotient topology, excluded middle, selected witnesses, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SemidecidableOpen}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SemidecidableOpen}^{\uparrow}$ packet is generated from a SierpinskiSpace classifier row, a Topology indexed-open row, finite StreamName observation windows, RegSeqRat rational readback, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SemidecidableOpen}^{\uparrow}}$, seed)

Scope closed: Definition 6464.1, Definition 6464.2, and Definition 6464.3 seed the finite positive-open observation interface over SierpinskiSpace, Topology, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SemidecidableOpen}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.SemidecidableOpenUp`

Bridge status: none

Not claimed: This chapter does not close negative open-membership decisions, host open sets, quotient topology, excluded middle, selected witnesses, ambient real completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive classifier exactness, finite-window readback, real observation routing, transport, replay, provenance, and local naming for BEDC.Derived.SemidecidableOpenUp.

Chapter 6465

A Concrete Naming Certificate for CauchyCondensationIntegralBound$^{\uparrow}$

Following Hardy, *Divergent Series*, the CauchyCondensationIntegralBound$^{\uparrow}$ packet records the finite integral-bound ledger that sits between the theorem-level condensation surface of chapter 3727 and the bridge packet of chapter 6071. It also shares the dyadic arithmetic, regular rational readback, and terminal real-name corridor of chapter 291, chapter 301, and chapter 38. The carrier names monotone positive term windows and dyadic integral-window majorants as BEDC rows before any Real or Completion consumer reads a bound.

Definition 6465.1 (CauchyCondensationIntegralBound carrier)**.** *A* CauchyCondensationIntegralBound$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, D, I, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the monotone positive term ledger, recording only inspected source windows, positivity witnesses, and monotonicity comparisons for the displayed tail.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *dyadic scale row, carrying powers-of-two windows, endpoint comparisons, and dyadic slack data.*
3. *I is the integral-window ledger, recording finite lower and upper interval cells attached to the displayed dyadic scale.*
4. *B is the dyadic integral-window majorant row, bounding each displayed condensation block by the corresponding integral-window data.*
5. *W is the finite readback schedule for Real and Completion consumers.*
6. *R is the* RegSeqRat$^{\uparrow}$ *tail readback row obtained from M, D, I, B, W.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after the majorant and regular-readback rows are visible.*
8. *H records componentwise* hsame *transport for monotone terms, dyadic scales, integral windows, majorants, schedules, readback rows, seal rows, replay rows, provenance rows, and local names.*
9. *C records displayed* Cont *replay from M, D, I to B, from B, W to R, and from R to the seal row E.*

10. *P is the* Pkg *provenance over* CauchyCondensationIntegralBound$^{\uparrow}$, CauchyCondensationTheorem$^{\uparrow}$, CauchyCondensationIntegralBridge$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCondensationIntegralBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, I, B, W, R, E, H, C, P, N$. It has no coordinate for an ambient improper integral, selected infinite sum, quotient series equality, countable choice, trichotomy, host real equality, or completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCondensationIntegralBoundUp.CauchyCondensationIntegralBoundTasteGate`
`single carrier alignment`

Theorem 6465.2 (CauchyCondensationIntegralBound NameCert obligations)**.** *For every accepted* CauchyCondensationIntegralBound$^{\uparrow}$ *carrier $J = (M, D, I, B, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyCondensatio}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; monotone positive term exposure through M; dyadic scale exposure through D; integral-window exposure through I; dyadic majorant exposure through B; finite schedule exposure through W;* RegSeqRat$^{\uparrow}$ *readback through R;* Real$^{\uparrow}$ *sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier rows from Definition 6465.1. The packet exposes only monotone term windows, dyadic scale data, integral windows, majorant rows, finite schedules, regular readbacks, Real seals, and the structural rows.

Read the analytic handoff through the adjacent condensation chapters. The theorem packet supplies the monotone dyadic block route, while the integral bridge supplies the finite comparison corridor between condensation tails and integral windows.

The local NameCert row admits exactly those exposed rows and their structural transports. Since the carrier has no coordinate for an ambient integral, selected sum, quotient equality, countable choice, trichotomy, host equality, or Real completeness, the listed obligations exhaust the certificate surface. □

Theorem 6465.3 (CauchyCondensationIntegralBound dyadic majorant)**.** *Every accepted* CauchyCondensationInteg *carrier sends a monotone positive tail to a dyadic integral-window majorant only through the finite route*

$$M, D, I \longmapsto B \longmapsto W \longmapsto R.$$

The route is anchored by CauchyCondensationTheorem$^{\uparrow}$ *on the condensation side and by* CauchyCondensationIntegral *on the integral-window side, and it exports no selected subseries, host integral object, quotient real equality, or ambient completeness principle.*

Proof. Begin with the source rows M, D, I. The monotone positive term ledger and dyadic scale rows determine which finite integral windows may be inspected.

Use the bridge route of chapter 6071. It matches the condensation-side dyadic tail to the integral-window side before any regular rational readback is consumed.

The majorant row B is then read through the finite schedule W into R. Componentwise transport and continuation replay preserve the same route, so the majorant cannot be replaced by an unlisted selected sum, ambient integral, quotient equality, or completeness row. □

Theorem 6465.4 (CauchyCondensationIntegralBound Real seal)**.** *Every Real-facing read of an accepted* CauchyCondensationIntegralBound$^{\uparrow}$ *carrier reaches E only after the displayed route*

$$M, D, I, B, W, R, E$$

has exposed the monotone positive term ledger, dyadic integral-window majorant, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and structural rows* H, C, P, N*. Thus the seal records a finite integral-bound certificate, not an ambient summability theorem or a completeness assertion for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Apply Definition 6465.3. It places every bound read behind M, D, I, B, W, R and keeps the dyadic and integral-window rows visible.

The terminal seal row E is downstream of the regular readback row. A Real consumer therefore receives the bound only after the regular rational handoff is present.

Finally inspect the carrier exactness from Definition 6465.1. The structural rows transport, replay, source, and name the same finite route, while no coordinate supplies ambient integration, selected infinite sums, quotient equality, countable choice, trichotomy, host equality, or Real completeness. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationIntegralBound}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the monotone positive term row, dyadic scale row, integral-window row, dyadic majorant row, finite readback schedule, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: cauchy-condensation-theorem, cauchy-condensation-integral-bridge, dyadicratcore, regseqrat, real: no listed sibling supplies the whole integral-bound packet $J = (M, D, I, B, W, R, E, H, C,$ because each sibling contributes only a theorem row, bridge row, dyadic ledger, regular readback, or terminal seal while this chapter binds them into one finite majorant certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralBound}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCondensationIntegralBound}^{\uparrow}$ packet is generated from monotone positive term windows, DyadicRatCore scale rows, finite integral-window rows, dyadic majorant rows, readback schedules, RegSeqRat tail readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6465.1, Definition 6465.2, Definition 6465.3, and Definition 6465.4 seed the finite integral-bound ledger over monotone tails, dyadic windows, RegSeqRat readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralBound}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyCondensationIntegralBoundUp.CauchyCondensationIntegralBoundTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close ambient improper integration, selected infinite sums, quotient series equality, countable choice, trichotomy, host real equality, Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone positive term exposure, dyadic integral-window majorant exposure, regular readback exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationIntegralBoundUp.

Chapter 6466

A Concrete Naming Certificate for $\mathsf{CompactUniformTailSelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactUniformTailSelector}^{\uparrow}$ packet records the finite selector surface that turns compact finite-net data and pointwise modulus ledgers into a tolerance-only uniform tail ledger. It is grounded in the compact-uniform-continuity route of chapter 9237, the compact-net packets of chapter 4760 and chapter 6453, the continuous-map route of chapter 130, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite ProbeBundle tail-radius handoff, not a global choice function, open-cover compactness theorem, or ambient completeness principle.

Definition 6466.1 (CompactUniformTailSelector carrier)**.** *A* $\mathsf{CompactUniformTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, Pb, L, D, U, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, exposing only the inspected* ProbeBundle *of centers and coverage witnesses.*
2. *F is the continuous-map row whose pointwise modulus entries are being folded.*
3. *Pb is the finite* ProbeBundle *spine used by the selector.*
4. *L is the local tail-radius ledger, one displayed radius and pointwise-modulus certificate for each bundled probe.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the local radii and the requested output tolerance.*
6. *U is the tolerance-only uniform tail threshold selected by finite fold over Pb, L, D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected threshold and the inspected finite net.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, F, Pb, L, D, U, R have been displayed.*
9. *H records componentwise* hsame *transport for compact nets, pointwise modulus rows, bundle spines, local radii, dyadic tolerances, selected thresholds, readbacks, seals, provenance, and names.*
10. *C records displayed* Cont *replay from compact finite-net rows and pointwise moduli to U, and from U to R, E.*

11. *Q is the* Pkg *provenance over* $\mathsf{CompactUniformTailSelector}^{\uparrow}$, $\mathsf{CompactUniformFiniteNet}^{\uparrow}$, CompactNetModulusStab
ContinuousMap$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a global point selector, arbitrary open cover, selected minimum outside the bundle fold, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6466.2 (CompactUniformTailSelector NameCert obligations)**.** *For every accepted* CompactUniformTailS
carrier $T = (K, F, Pb, L, D, U, R, E, H, C, Q, N)$, the local $\mathsf{NameCert}_{\mathsf{CompactUniformTailSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; compact finite-net exposure through K; continuous-map pointwise-modulus exposure through F;* ProbeBundle *spine exposure through Pb; local tail-radius exposure through L; dyadic tolerance comparison through D; uniform threshold exposure through U;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through Q; and local naming through N.*

Proof. Project the carrier rows from Definition 6466.1. The accepted packet exposes compact finite-net rows, pointwise-modulus rows, a finite bundle spine, local tail radii, dyadic tolerances, a selected uniform threshold, regular readback, Real seal, and structural rows.

The finite fold is internal to the displayed ProbeBundle. It reads every bundled local radius and keeps the threshold row U as a certificate over the same finite carrier.

The structural rows preserve this route under transport, replay, provenance, and local naming. Since no row supplies a global choice function, arbitrary open-cover compactness, quotient equality, selected external minimum, countable choice, or Real completeness, the listed obligations exhaust the NameCert surface. □

Theorem 6466.3 (CompactUniformTailSelector finite-net handoff)**.** *Every accepted* $\mathsf{CompactUniformTailSelector}^{\uparrow}$ *packet hands compact finite-net data to the uniform selector only through the finite route*

$$K, F, Pb, L, D \longmapsto U.$$

The handoff reads the ProbeBundle *spine and pointwise modulus rows before producing a tolerance-only tail threshold, and it does not choose points, subcovers, or radii outside the displayed bundle.*

Proof. Begin with the compact finite-net row K and the pointwise modulus row F. These rows determine the inspected centers, local radius certificates, and coverage witnesses available to the selector.

Fold over the finite bundle spine Pb. Each local tail-radius row in L is compared through the dyadic tolerance row D, and the fold exports only the threshold row U.

The transport and continuation rows preserve the same finite handoff. The packet therefore cannot import a global point selector, selected subcover, external minimum, quotient real equality, or completeness row. □

Theorem 6466.4 (CompactUniformTailSelector finite-net route)**.** *Every accepted* $\mathsf{CompactUniformTailSelector}^{\uparrow}$ *packet receives compact finite-net data through* $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ *and* $\mathsf{CompactNetModulusStability}^{\uparrow}$ *before the* ProbeBundle *spine, local radius ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal are visible. For a carrier*

$$T = (K, F, Pb, L, D, U, R, E, H, C, Q, N),$$

the route projects K, F, Pb, L, D, U, R, E in that order and exports no open-cover compactness object, point-choice coordinate, external minimum selector, countable-choice schedule, quotient real equality, or ambient completeness witness.

Proof. Project the carrier rows $K, F, Pb, L, D, U, R, E, H, C, Q, N$ from Definition 6466.1. The rows K and F are the compact finite-net and pointwise-modulus inputs, while Pb, L, D are the finite bundle spine, local radius ledger, and dyadic tolerance ledger inspected by the selector.

Apply Definition 6466.3. It forces the selector threshold U to be produced only after the compact finite-net row, pointwise modulus row, bundle spine, local radius rows, and dyadic comparisons have been displayed.

Now apply Definition 6466.5. The regular readback R and terminal Real seal E are downstream of the selected threshold, so the Real-facing route cannot be read before the finite-net and pointwise-modulus data have passed through the displayed ProbeBundle discipline.

Finally inspect the excluded coordinates in the carrier. Since open-cover compactness, point choice, external minima, countable choice, quotient real equality, and ambient completeness are absent from $K, F, Pb, L, D, U, R, E, H, C, Q, N$, every accepted packet consumes the compact finite-net route directly through $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ and $\mathsf{CompactNetModulusStability}^{\uparrow}$ rather than through any external compactness principle. □

Theorem 6466.5 (CompactUniformTailSelector uniform modulus route)**.** *Every Real-facing or uniform-modulus consumer of an accepted* $\mathsf{CompactUniformTailSelector}^{\uparrow}$ *packet must read*

$$K, F, Pb, L, D, U, R, E$$

in that order before treating the selected threshold as a uniform modulus. The route turns compact finite-net coverage and pointwise moduli into a tolerance-only ledger through $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, not through open-cover compactness, sequential compactness, countable choice, or ambient Real completeness.*

Proof. Apply Definition 6466.3. It places the threshold U behind compact finite-net rows, pointwise modulus rows, the finite bundle spine, local radii, and dyadic comparisons.

The readback row R and seal row E are downstream of U. A uniform-modulus consumer therefore receives the selected threshold only after regular rational readback and the Real-facing seal have preserved the same finite route.

Finally inspect the excluded rows in Definition 6466.1. There is no coordinate for an arbitrary open cover, sequential compactness, a global choice function, quotient real equality, selected external minimum, or Real completeness, so the uniform-modulus route remains the displayed finite handoff. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformTailSelector}^{\uparrow}$ consumer that reads a tolerance-only uniform tail threshold without the compact finite-net row, pointwise modulus row, finite ProbeBundle spine, local tail-radius ledger, DyadicRatCore tolerance row, RegSeqRat readback row, RealUp seal row, and structural rows refutes the carrier surface.

Independence witness: compact-uniform-finite-net, compact-net-modulus-stability, continuous-map, dyadicratcore, regseqrat, real: no listed sibling supplies the whole selector packet $T = (K, F, Pb, L, D, U, R, E, H, C, Q, N)$, because each sibling contributes only a compact-net row, pointwise-modulus row, dyadic ledger, regular readback, or terminal seal while this chapter binds them into one tolerance-only tail selector.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformTailSelector}^{\uparrow}}$.

Constructive story: A CompactUniformTailSelector$^{\uparrow}$ packet is generated from compact finite-net rows, continuous-map pointwise modulus rows, a finite ProbeBundle spine, local tail-radius rows, DyadicRatCore tolerance comparisons, a tolerance-only uniform threshold, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformTailSelector}^{\uparrow}}$, seed)

Scope closed: Definition 6466.1, Definition 6466.2, Definition 6466.3, and Definition 6466.5 seed the finite compact-net tail selector from ProbeBundle radii to a tolerance-only uniform-modulus ledger.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformTailSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, sequential compactness, global point selection, selected external minima, quotient real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, pointwise modulus exposure, finite-bundle fold exactness, tolerance-threshold readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformTailSelectorUp.

Chapter 6467

A Concrete Naming Certificate for FinePartitionGauge$^{\uparrow}$

Following Cousin and Bishop–Bridges gauge integration, the FinePartitionGauge$^{\uparrow}$ packet records a finite tagged partition whose cells are already certified fine for a displayed positive gauge. It sits between the existence route of chapter 1861, the cover-facing surface of chapter 3426, the integral packet of chapter 1999, the tagged-sum route of chapter 4115, and the lower–upper comparison surface of chapter 1880. The packet names only finite interval data, positive gauge bounds, tagged cells, fineness proofs, mesh readback, Cousin handoff, regular rational readback, terminal real sealing, transport, replay, provenance, and a local certificate.

Definition 6467.1 (FinePartitionGauge carrier)**.** *A* FinePartitionGauge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, G, T, L, M, U, R, E, H, C, P, N)$$

whose rows expose:

1. *I, the interval source row read from the displayed compact interval or located interval source;*
2. *G, the positive gauge row assigning a strictly positive local radius to each admitted tag;*
3. *T, the finite tagged-cell ledger listing cells, tags, endpoint order, and cell coverage;*
4. *L, the fine-cell proof row stating that every tagged cell of T lies inside the gauge radius supplied by G at its tag;*
5. *M, the mesh row recording the largest displayed cell width and the finite comparison budget used by later sums;*
6. *U, the Cousin handoff row through chapter 1861 and chapter 3426;*
7. *R, the* RegSeqRat$^{\uparrow}$ *readback row for rational mesh and tagged-sum approximants;*
8. *E, the terminal* Real$^{\uparrow}$ *seal row available only after the regular rational readback;*
9. *H, the componentwise* hsame *transport row for interval, gauge, cell, mesh, handoff, readback, and seal data;*
10. *C, the displayed* Cont *replay row preserving the same finite tagged partition route;*
11. *P, the* Pkg *provenance over* FinePartitionGauge$^{\uparrow}$, CousinLemma$^{\uparrow}$, CousinCover$^{\uparrow}$, GaugeIntegral$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

12. N, the local $\mathsf{NameCert}_{\mathsf{FinePartitionGauge}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, G, T, L, M, U, R, E, H, C, P, N$. It has no coordinate for a choice-selected partition, an unbounded open cover, host compactness, quotient equality of real functions, or an ambient integration theorem.

Theorem 6467.2 (FinePartitionGauge NameCert obligations)**.** *For every accepted* $\mathsf{FinePartitionGauge}^{\uparrow}$ *carrier $F = (I, G, T, L, M, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FinePartitionGauge}^{\uparrow}}$ *surface is exhausted by interval-source admission, positive-gauge admission, finite tagged-cell coverage, fine-cell proof, mesh readback, Cousin handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{GaugeIntegral}^{\uparrow}$*,* $\mathsf{RiemannSumGauge}^{\uparrow}$*, or* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer must read these rows before using the packet.*

Proof. Project the carrier of Definition 6467.1. The rows $I, G, T, L, M, U, R, E, H, C, P, N$ are the entire displayed packet, so any local certificate obligation must be one of the listed interval, gauge, tagged-cell, fineness, mesh, handoff, readback, seal, transport, replay, provenance, or naming rows.

The fineness obligation is read before any sum or integral row. The tagged-cell ledger T supplies only finite cells and tags, while L checks each cell against the positive radius supplied by G.

The Cousin handoff and readback rows keep downstream consumers on the same finite route. The row U is the only route to the Cousin existence surface, R is the only regular rational readback, and E is visible only after R. Since the carrier has no selected-partition, host-compactness, or ambient integration coordinate, the NameCert surface is exactly the displayed packet. □

Theorem 6467.3 (FinePartitionGauge Cousin route)**.** *Every* $\mathsf{FinePartitionGauge}^{\uparrow}$ *carrier routes through the Cousin existence surface before any* $\mathsf{GaugeIntegral}^{\uparrow}$*,* $\mathsf{RiemannSumGauge}^{\uparrow}$*, or* $\mathsf{DarbouxSum}^{\uparrow}$ *read. In the notation of Definition 6467.1, a downstream integration consumer may inspect*

$$I, \quad G, \quad T, \quad L, \quad M, \quad U$$

and only then read the regular rational route R, the terminal seal E, and the structural rows H, C, P, N. Thus the packet separates finite fine-partition data from the Cousin existence theorem and exposes no independent choice-selected partition.

Proof. Start with the interval, gauge, tagged-cell, and fine-cell rows. They display the finite partition candidate and verify fineness against the positive gauge without reading an integral or sum row.

Next route the packet through the Cousin handoff row U. This row points to the Cousin lemma and Cousin cover sibling surfaces, so existence and cover refinement are consumed before any GaugeIntegral, Riemann-sum, or Darboux-sum consumer can read the partition as integration data.

Finally apply the NameCert obligation theorem above. It places R, E, H, C, P, N downstream of the finite Cousin route, and the carrier has no coordinate for a host choice selector or ambient compactness theorem. The downstream read therefore receives a displayed fine partition surface rather than an unbounded existence principle. □

Falsifiable prediction: Any accepted $\mathsf{FinePartitionGauge}^{\uparrow}$ packet must expose interval source, positive gauge, tagged-cell, fine-cell, mesh, Cousin handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows before an integration consumer reads it.

Independence witness: cousinlemma, cousincover, gaugeintegral, riemannsumgauge, darbouxsum: $\mathsf{FinePartitionGauge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tagged

cells, pointwise gauge fineness, mesh readback, Cousin handoff, regular rational readback, and terminal real sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{FinePartitionGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{FinePartitionGauge}^{\uparrow}$ packet is generated from an interval source, positive gauge, finite tagged cells, fine-cell proof, mesh row, Cousin handoff, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinePartitionGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6467.1, Definition 6467.2, and Definition 6467.3 seed the finite fine-partition surface between Cousin existence and gauge-integration consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinePartitionGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected partitions, host compactness, ambient integration theorems, quotient equality of real functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-gauge admission, tagged-cell coverage, fine-cell proof, mesh readback, Cousin handoff, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Chapter 6468

A Concrete Naming Certificate for $\mathsf{JamesSpace}^{\uparrow}$

Following R. C. James, the $\mathsf{JamesSpace}^{\uparrow}$ packet records the classical block-basis norm route for James' nonreflexive Banach space as finite BEDC ledger data. It sits beside the vector-space carrier of chapter 47, the normed-space handoff of chapter 1950, the Banach completion surface of chapter 90, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a concrete finite inspection surface for $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers; it does not claim a basis projection theorem, reflexivity failure, weak compactness, dual identification, or a selected infinite block decomposition.

Definition 6468.1 (James-space carrier)**.** *A* $\mathsf{JamesSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, L, Q, R, V, M, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the basis-block ledger. Each row names a finite interval of the canonical unit-vector basis, its ordered endpoints, and the finite coefficient support inspected inside that block.*
2. *L is the block-link ledger. It records which displayed blocks are read consecutively and which rows remain disjoint, so a consumer can replay the James block pattern without an ambient infinite choice of blocks.*
3. *Q is the norm-budget ledger. Each row records a finite family of consecutive block sums, a rational or dyadic budget for the squared block-sum magnitudes, and the comparison needed for the James norm inspection.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficients and norm budgets.*
5. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *handoff row, recording finite support addition and scalar multiplication over the displayed coefficient rows.*
6. *M is the* $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{Banach}^{\uparrow}$ *handoff row. It names the normed carrier, the Cauchy-completion route, and the Banach-facing certificate surface consumed by downstream space arguments.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for every displayed norm-budget comparison reached from Q through R.*
8. *H records componentwise* hsame *transport for the basis-block ledger, block-link ledger, norm-budget ledger, rational readback, vector-space handoff, normed/Banach handoff, real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the carrier through the same finite block rows and norm-budget rows.*
10. *P is the* Pkg *provenance over* $\mathsf{JamesSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{JamesSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, L, Q, R, V, M, A, H, C, P, N$. It has no coordinate for an external Banach-space parameter, an infinite block selector, a dual-space presentation, reflexivity analysis, compactness, quotient equality, or host completeness beyond the displayed $\mathsf{Banach}^{\uparrow}$ *handoff.*

Theorem 6468.2 (James-space NameCert obligations)**.** *For every accepted* $\mathsf{JamesSpace}^{\uparrow}$ *carrier $J = (B, L, Q, R, V, M, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{JamesSpace}^{\uparrow}}$ *surface consists exactly of basis-block admission through B, consecutive-block coverage through L, norm-budget admission through Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{VecSpace}^{\uparrow}$ *finite-support handoff through V,* $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{Banach}^{\uparrow}$ *handoff through M,* $\mathsf{Real}^{\uparrow}$ *norm-seal admission through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No James-space obligation is stored outside those rows.

Proof. Project the carrier of Definition 6468.1. The accepted packet is exhausted by the displayed basis-block rows, the consecutive-block ledger, the norm-budget rows, the rational readback, the vector/normed/Banach handoffs, and the structural rows H, C, P, N.

The block side is finite. The rows B and L name only inspected basis intervals and their displayed adjacency data, so a consumer can replay the block pattern without selecting an infinite block family.

The norm side is also finite. The row Q records each inspected family of consecutive block sums together with its rational budget, while R reads those budgets through $\mathsf{RegSeqRat}^{\uparrow}$ and A seals the resulting comparisons through $\mathsf{Real}^{\uparrow}$.

Finally V and M carry the same rows to $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, and $\mathsf{Banach}^{\uparrow}$ consumers. The transport, replay, provenance, and naming rows preserve the displayed packet componentwise and cannot introduce any excluded coordinate. □

Theorem 6468.3 (James-space Banach route)**.** *Every* $\mathsf{Banach}^{\uparrow}$*-facing read of an accepted* $\mathsf{JamesSpace}^{\uparrow}$ *carrier factors through the finite route*

$$B, L, Q \longmapsto R, A \longmapsto V, M, H, C, P, N.$$

Thus the Banach-space construction is presented by finite basis-block rows, norm-budget rows, rational readback, real norm seals, vector/normed/Banach handoffs, componentwise transport, displayed replay, provenance, and local naming, rather than by an opaque ambient Banach parameter.

Proof. Start with a Banach-facing consumer. The carrier first exposes B and L, so the consumer sees the inspected canonical-basis blocks and their displayed consecutive-link data before any normed-space handoff is available.

Read the norm route next. The row Q supplies exactly the finite James norm-budget comparisons under inspection, R translates the coefficient and budget data through $\mathsf{RegSeqRat}^{\uparrow}$, and A seals the resulting real-valued norm comparisons through $\mathsf{Real}^{\uparrow}$.

The vector and Banach handoffs are downstream of those rows. The row V supplies the finite-support $\mathsf{VecSpace}^{\uparrow}$ operations, while M records the $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{Banach}^{\uparrow}$ handoff over the

same displayed norm data. Since H, C, P, N only transport, replay, source, and name the already displayed packet, the route cannot replace the finite ledger by an external Banach parameter. □

Falsifiable prediction: A Banach-space construction that reads $\mathsf{JamesSpace}^{\uparrow}$ after deleting the basis-block ledger B, the norm-budget ledger Q, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, the $\mathsf{Real}^{\uparrow}$ seal row A, or the $\mathsf{NormedSpace}^{\uparrow}/\mathsf{Banach}^{\uparrow}$ handoff row M refutes the carrier of Definition 6468.1.

Independence witness: vecspace, normedspace, banach, regseqrat, real: $\mathsf{JamesSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite basis-block rows, consecutive-block links, norm-budget rows, rational readback, real norm seals, VecSpace/NormedSpace/Banach handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{JamesSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{JamesSpace}^{\uparrow}$ packet is generated from finite basis-block rows, consecutive-block links, norm-budget rows, RegSeqRat readback, Real norm seals, VecSpace/NormedSpace/Banach handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JamesSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6468.1, Definition 6468.2, and Definition 6468.3 seed the finite James-space construction surface for Banach-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JamesSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close basis projection theory, reflexivity failure, weak compactness, dual identification, infinite block selection, quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked basis-block admission, consecutive-block coverage, norm-budget admission, RegSeqRat readback, Real seal exactness, VecSpace/NormedSpace/Banach handoff exactness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.JamesSpaceUp.

Chapter 6469

A Concrete Naming Certificate for $\mathsf{AbsoluteRetract}^{\uparrow}$

Following Borsuk and Dugundji, the $\mathsf{AbsoluteRetract}^{\uparrow}$ packet records a constructive retract-extension neighbour for compact metric and located metric analysis. It sits after the compact-source surfaces of chapter 129 and chapter 60, the closed extension route of chapter 1956, the separating-function route of **??**, the convex target rows of chapter 208, and the finite distance comparison route of chapter 1533. The packet names only displayed extension and retraction rows; it does not assert an ambient absolute retract theorem, selected neighbourhood retractions, quotient compacta, open-cover compactness, or choice of extension operators.

Definition 6469.1 (Absolute retract carrier)**.** *An* $\mathsf{AbsoluteRetract}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, K, L, T, U, V, E, R, S, H, C, P, N)$$

where:

- X *is the displayed source carrier row;*
- K *is the compact or compact-metric source row read through* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$*;*
- L *is the* $\mathsf{LocatedMetric}^{\uparrow}$ *distance-comparison row;*
- T *is the* $\mathsf{TietzeExtension}^{\uparrow}$ *extension row for closed real-valued data;*
- U *is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *separation row used by closed-neighbourhood reads;*
- V *is the* $\mathsf{ConvexSet}^{\uparrow}$ *target row recording the convex retract codomain;*
- E *records the finite extension request from a displayed closed subpacket;*
- R *records the candidate retraction row;*
- S *records stability of the retraction after the extension route is displayed;*
- H, C, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{AbsoluteRetract}^{\uparrow}}$ *row.*

Theorem 6469.2 (Absolute retract NameCert obligations)**.** *For every accepted* $\mathsf{AbsoluteRetract}^{\uparrow}$ *carrier* $A = (X, K, L, T, U, V, E, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AbsoluteRetract}^{\uparrow}}$ *surface consists exactly of source admission, compact-source readback, located-metric comparison, Tietze extension exposure, Urysohn separation exposure, convex-target routing, finite extension request, retraction row, retraction-stability row, componentwise transport, displayed replay, provenance, and local naming. Any consumer that omits one of* $K, L, T, U, V, E, R, S, H, C, P, N$ *is outside the packet.*

Proof. Project the carrier of Definition 6469.1. Each row is displayed as finite BHist data, and the local naming row N records the accepted packet only over those displayed coordinates.

The compact and metric rows give the source scope through $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{LocatedMetric}^{\uparrow}$. The extension and separation rows then enter through $\mathsf{TietzeExtension}^{\uparrow}$ and $\mathsf{UrysohnLemma}^{\uparrow}$, while the target constraint is read through $\mathsf{ConvexSet}^{\uparrow}$.

Finally, E, R, S, H, C, P, N package the extension request, retraction, stability, transport, replay, provenance, and name. Since no row contains an ambient absolute retract principle or selected global operator, the listed surface is the entire local obligation set. □

Theorem 6469.3 (Absolute retract retraction stability)**.** *Every accepted* $\mathsf{AbsoluteRetract}^{\uparrow}$ *packet whose retraction row R is consumed must first expose the compact-source row K, located-metric row L, Tietze extension row T, Urysohn separation row U, convex-target row V, and finite extension request E. The stability row S then transports the retraction through H, C, P, N without replacing the packet by an ambient neighbourhood-retract theorem or a selected extension operator.*

Proof. Begin with the NameCert obligations of Definition 6469.2. They make $K, L, T, U, V, E, R, S, H, C, P, N$ the visible surface for any accepted packet.

The retraction read cannot precede the extension route. The compact and located-metric rows bound the source request, the Tietze and Urysohn rows expose the extension and separation data, and the convex row records the target shape before R is read.

The stability row S is therefore only a transport of the displayed retraction along the same finite route. It does not create a host absolute retract theorem, quotient compactum, chosen operator, or open-cover compactness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteRetract}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbsoluteRetract}^{\uparrow}$ packet is generated from CompactUp and CompactMetricUp source rows, LocatedMetricUp comparison rows, TietzeExtensionUp extension rows, UrysohnLemmaUp separation rows, ConvexSetUp target rows, finite extension and retraction rows, stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AbsoluteRetract}^{\uparrow}}$, seed)

Scope closed: Definition 6469.1, Definition 6469.2, and Definition 6469.3 seed the finite retract-extension packet over compact, located-metric, extension, separation, and convex-target rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AbsoluteRetract}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close an ambient absolute retract theorem, selected neighbourhood retractions, quotient compacta, open-cover compactness, choice of extension operators, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source readback, located-metric comparison, Tietze extension exposure, Urysohn separation exposure, convex-target routing, retraction stability, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteRetractUp.

Chapter 6470

A Concrete Naming Certificate for $\mathsf{GromovBoundary}^{\uparrow}$

Following Gromov's hyperbolic-boundary construction, the $\mathsf{GromovBoundary}^{\uparrow}$ packet records finite boundary-equivalence data for hyperbolic rays before a visual or cocycle consumer reads the boundary. It sits after the visual-boundary packet of chapter 4181, the basepointed product packet of chapter 6073, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, and the boundary-shadow expansion route of chapter 5814. The packet names a displayed boundary carrier rather than a completed quotient of rays, selected geodesic representatives, host compactification, or ambient hyperbolic space.

Definition 6470.1 (Gromov boundary carrier)**.** *A* $\mathsf{GromovBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, P, V, M, C, Q, E, S, T, H, R, N)$$

where:

- B *is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *boundary row;*
- P *is the* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *basepointed product row;*
- V *is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual-metric row;*
- M *is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *cocycle row;*
- C *is the* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *boundary-shadow expansion row;*
- Q *records finite ray-equivalence comparisons;*
- E *records endpoint admission for the accepted boundary token;*
- S *records visual-cocycle stability under the displayed equivalence row;*
- T, H, R, N *are displayed* Cont *replay, componentwise* hsame *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GromovBoundary}^{\uparrow}}$ *row.*

Theorem 6470.2 (Gromov boundary NameCert obligations)**.** *For every accepted* $\mathsf{GromovBoundary}^{\uparrow}$ *carrier* $G = (B, P, V, M, C, Q, E, S, T, H, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GromovBoundary}^{\uparrow}}$ *surface consists exactly of visual-boundary admission, Gromov-product readback, visual-metric comparison, Busemann-cocycle comparison, boundary-shadow expansion, finite ray-equivalence exposure, endpoint admission, visual-cocycle stability, displayed replay, componentwise transport, provenance,*

and local naming. A consumer that reads a boundary token without one of those rows is outside the packet.

Proof. Project the carrier of Definition 6470.1. The visual boundary, Gromov product, visual metric, Busemann cocycle, and boundary-shadow expansion rows are all explicit coordinates of the accepted packet.

The ray-equivalence row Q and endpoint row E provide the boundary token only as displayed finite data. The packet therefore exposes equivalence information before any visual metric or cocycle consumer receives the boundary.

Finally, S, T, H, R, N record stability, replay, transport, provenance, and naming. Since no carrier row supplies a host quotient of rays, selected geodesic representative, completed compactification, or ambient hyperbolic model, the listed rows exhaust the NameCert obligation surface. □

Theorem 6470.3 (Gromov boundary visual-cocycle stability)**.** *Every accepted* $\mathsf{GromovBoundary}^{\uparrow}$ *packet whose boundary token is consumed by a visual-metric, Busemann-cocycle, or boundary-shadow-expansion consumer must retain the ray-equivalence row* Q *through the* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *row* P*,* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row* V*,* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row* M*, and* $\mathsf{BoundaryShadowExpansionCocycle}$ *row* C*. The stability row* S *transports those displayed comparisons without turning the packet into a completed ray quotient or global boundary compactification.*

Proof. Begin with Definition 6470.2. It keeps $B, P, V, M, C, Q, E, S, T, H, R, N$ visible for every accepted boundary read.

The product row P and ray-equivalence row Q determine which boundary token is being transported. The visual metric, Busemann cocycle, and boundary-shadow expansion rows then read that token through their own displayed comparison surfaces.

The stability row S preserves those comparisons through T, H, R, N. Because the carrier has no coordinate for a selected ray representative or host quotient, the handoff remains a finite BEDC boundary-equivalence packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GromovBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GromovBoundary}^{\uparrow}$ packet is generated from HyperbolicVisualBoundaryUp rows, HyperbolicGromovProductUp rows, HyperbolicVisualMetricUp rows, HyperbolicBusemannCocycleUp rows, BoundaryShadowExpansionCocycleUp rows, finite ray-equivalence rows, endpoint rows, visual-cocycle stability rows, displayed Cont replay, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GromovBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 6470.1, Definition 6470.2, and Definition 6470.3 seed the finite boundary-equivalence packet over visual-boundary, Gromov-product, visual-metric, Busemann-cocycle, and boundary-shadow expansion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GromovBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a completed quotient of rays, selected geodesic representatives, host compactification, ambient hyperbolic space, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-boundary admission, Gromov-product readback, visual-metric comparison, Busemann-cocycle comparison, boundary-shadow expansion, ray-equivalence exposure, visual-cocycle stability, transport, replay, provenance, and local naming for BEDC.Derived.GromovBoundaryUp.

Chapter 6471

A Concrete Naming Certificate for $\mathsf{HausdorffYoung}^{\uparrow}$

Following Hausdorff and Young on Fourier coefficient bounds, and Riesz–Thorin interpolation, the $\mathsf{HausdorffYoung}^{\uparrow}$ packet records a constructive endpoint-to-interpolation route for finite Fourier coefficient windows. It sits beside the convolution bound of chapter 2358, the interpolation certificate of chapter 5725, the energy identity of chapter 1940, the majorant-tail discipline of chapter 399, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names only finite Fourier observations, exponent rows, interpolation and summability handoffs, rational readback, real sealing, transport, replay, provenance, and local certification.

Definition 6471.1 (HausdorffYoung carrier)**.** *A* $\mathsf{HausdorffYoung}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (F, P, Q, Y, T, S, R, E, K, C, G, N)$$

whose rows expose:

1. *F, the Fourier coefficient window row, including the displayed finite source window and the retained coefficient read schedule;*
2. *P, the L^p exponent row recording the admitted source exponent and its finite comparison witnesses;*
3. *Q, the conjugate L^q exponent row recording the target coefficient exponent and the displayed conjugacy ledger;*
4. *Y, the Young-inequality handoff row through chapter 2358;*
5. *T, the Riesz–Thorin interpolation certificate row through chapter 5725;*
6. *S, the summability-tail row, including the finite majorant and truncation witnesses consumed by chapter 399;*
7. *R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coefficient norms, tail budgets, and exponent comparisons;*
8. *E, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the real-valued bound after R has been displayed;*
9. *K, the componentwise* hsame *transport row for Fourier windows, exponent rows, interpolation data, tail rows, readback, and seal data;*
10. *C, the displayed* Cont *replay row preserving the same finite Hausdorff–Young route;*

11. *G, the* Pkg *provenance over* HausdorffYoung$^{\uparrow}$, YoungInequality$^{\uparrow}$, RieszThorinInterpolation$^{\uparrow}$, Parseval$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N, the local* NameCert$_{\text{HausdorffYoung}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, P, Q, Y, T, S, R, E, K, C, G, N$. It has no coordinate for a host Fourier transform, a completed L^p quotient space, a selected representative, countable choice, or an ambient measure-theoretic interpolation theorem.

Theorem 6471.2 (HausdorffYoung NameCert obligations)**.** *For every accepted* HausdorffYoung$^{\uparrow}$ *carrier $H = (F, P, Q, Y, T, S, R, E, K, C, G, N)$, the local* NameCert$_{\text{HausdorffYoung}^{\uparrow}}$ *surface is exhausted by Fourier coefficient windows, L^p and L^q exponent rows, the* YoungInequality$^{\uparrow}$ *handoff, the* RieszThorinInterpolation$^{\uparrow}$ *interpolation certificate, summability-tail witnesses,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every coefficient-norm or interpolation consumer must read these rows before using the packet.*

Proof. Project the carrier of Definition 6471.1. The displayed rows $F, P, Q, Y, T, S, R, E, K, C, G, N$ are the full accepted packet, so any local NameCert obligation is one of the listed Fourier-window, exponent, handoff, interpolation, summability, readback, seal, transport, replay, provenance, or naming rows.

The source and target exponent obligations are read before interpolation. The rows P and Q carry only finite exponent comparisons and conjugacy witnesses, while F fixes the coefficient window on which the bound is observed.

The analytic handoff is then forced through the named siblings. The row Y supplies the Young-inequality bound surface, T supplies the Riesz–Thorin interpolation certificate, and S supplies the finite tail witnesses before R and E expose any real-valued read. Since the carrier has no host transform, quotient-space, or choice coordinate, the NameCert surface is exactly the displayed finite packet. □

Theorem 6471.3 (HausdorffYoung Riesz–Thorin route)**.** *Every* HausdorffYoung$^{\uparrow}$ *carrier routes a Fourier coefficient bound through the displayed Riesz–Thorin interpolation row before exposing a real coefficient-norm seal. In the notation of Definition 6471.1, a downstream consumer may inspect*

$$F, \quad P, \quad Q, \quad Y, \quad T, \quad S$$

and only then read the regular rational route R, the terminal seal E, and the structural rows K, C, G, N. Thus the packet separates finite Fourier coefficient observations from the interpolation theorem and exports no independent host L^p Fourier transform.

Proof. Start with the finite coefficient and exponent rows. The row F displays the observed Fourier window, while P and Q display the source and target exponent comparison data without appealing to a completed function space.

Next route the bound through the analytic siblings. The Young row Y provides the endpoint convolution-bound handoff, and the interpolation row T records the Riesz–Thorin certificate that transports the endpoint data to the requested exponent pair.

Finally read the finite tail and endpoint seal. The summability row S supplies the visible truncation witnesses, R repackages the coefficient-norm comparisons as regular rational data, and E seals the resulting real bound. Componentwise transport, replay, provenance, and local naming preserve the same route, so no host transform or selected representative can enter the certificate. □

Falsifiable prediction: Any accepted $\mathsf{HausdorffYoung}^{\uparrow}$ packet must expose Fourier coefficient windows, L^p/L^q exponent rows, Young handoff, Riesz–Thorin interpolation, summability tail, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows before a coefficient-norm consumer reads it.

Independence witness: younginequality, rieszthorin, parseval, weierstrassmtest, regseqrat, real: $\mathsf{HausdorffYoung}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Fourier coefficient windows, paired exponent rows, convolution-bound handoff, interpolation certificate, summability tail, rational readback, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffYoung}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffYoung}^{\uparrow}$ packet is generated from finite Fourier coefficient windows, source and target exponent rows, a Young handoff, a Riesz–Thorin interpolation certificate, summability-tail witnesses, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffYoung}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6471.1, Definition 6471.2, and Definition 6471.3 seed the finite Fourier interpolation certificate consumed by coefficient-norm work.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffYoung}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Fourier transforms, completed L^p quotient spaces, selected representatives, measure-theoretic interpolation theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponent rows, Young handoff, Riesz–Thorin interpolation, summability-tail witnesses, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Chapter 6472

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ packet records the finite paired-modulus common-tail certificate used when two Bishop Cauchy schedules must be read through one later tail. It sits after the single-modulus schedule row of chapter 292, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the cofinal-window route of chapter 4060. The packet is a paired finite tail ledger for L10 Real and Completion consumers; it is not quotient stream equality, a selected limit, countable choice, trichotomy, or ambient Real completeness.

Definition 6472.1 (BishopCauchyModulusCofinality carrier)**.** *A* $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\epsilon, D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. ϵ *is the requested tolerance row.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger for* ϵ *and the two tail budgets.*
3. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows being compared.*
4. R_0 *and* R_1 *are the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for those windows.*
5. μ_0 *and* μ_1 *are the two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule rows.*
6. W *is the* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row that displays a later finite window cofinal for both schedules at tolerance* ϵ*.*
7. T *is the common-tail certificate, stating that every read through* W *factors through both* μ_0 *and* μ_1 *after the displayed dyadic comparison.*
8. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* $D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T$ *have been displayed.*
9. H *records componentwise* hsame *transport for tolerances, dyadic comparisons, stream windows, regular readbacks, Cauchy moduli, cofinal windows, common-tail certificates, Real seals, provenance, and names.*

10. *C records displayed* Cont *replay from the two modulus schedules to W, T, E through the same finite rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient Real completeness.

Theorem 6472.2 (BishopCauchyModulusCofinality NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ *carrier* $M = (\epsilon, D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCofinality}^{\uparrow}}$ *surface consists exactly of tolerance admission through ϵ; dyadic comparison through D; two finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S_0, S_1; two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks through R_0, R_1; two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules through μ_0, μ_1; cofinal-window exposure through W; common-tail exposure through T;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier rows from Definition 6472.1. The accepted packet displays one tolerance, one dyadic comparison ledger, two stream windows, two regular rational readbacks, two Cauchy modulus schedules, one cofinal window, one common-tail certificate, one Real seal, and the structural transport, replay, provenance, and naming rows.

The paired nature of the certificate is carried only by μ_0, μ_1, W, T. The cofinal-window row W is read after the two schedules and the dyadic ledger, and T records that the later finite tail factors through both schedules.

The structural rows preserve the same finite packet under hsame, Cont, Pkg, and NameCert replay. Since no row carries quotient stream equality, a selected limit, countable choice, trichotomy, or ambient Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 6472.3 (BishopCauchyModulusCofinality common tail)**.** *If an accepted* $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ *packet presents two Cauchy modulus schedules μ_0, μ_1 over the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, then every common-tail read at tolerance ϵ factors through the finite route*

$$D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T.$$

The common tail is the displayed cofinal window W together with certificate T, not a quotient-stream identification, selected limit, countable-choice selection, trichotomy branch, or Real-completeness argument.

Proof. Begin with the dyadic tolerance row D. It fixes the comparison budget for the two schedules and prevents the common-tail read from changing tolerance outside the displayed packet.

Next read the two stream and regular-rational sides. The rows S_0, S_1 expose the finite windows, and R_0, R_1 attach those windows to the regular rational readbacks that the two modulus schedules govern.

Finally apply the cofinal-window row W and the common-tail certificate T. These rows display the later finite tail that is after both schedules at the requested tolerance, so a consumer receives a paired finite tail certificate without invoking quotient streams, selected limits, countable choice, trichotomy, or Real completeness. □

Theorem 6472.4 (BishopCauchyModulusCofinality Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ *packet reaches the seal* E *only after the dyadic ledger, both stream windows, both regular rational readbacks, both Cauchy modulus schedules, the* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row, and the common-tail certificate have been displayed. The handoff exports a finite paired-modulus tail certificate for L10 Real and Completion routes, and exports no quotient stream equality, selected limit, countable choice, trichotomy, or ambient Real completeness.*

Proof. Project the finite route before the seal. The rows $D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T$ are all upstream of E, so the Real-facing consumer cannot see the seal without also seeing the paired-modulus common-tail data.

Transport and replay preserve that order. The rows H and C can carry the displayed packet through equivalent naming and continuation reads, but they do not add another stream quotient, hidden tail selector, or completeness coordinate.

Thus the terminal $\mathsf{Real}^{\uparrow}$ read is only a seal for the finite paired-tail route. It remains choice-free and quotient-free, and it does not decide trichotomy or claim a selected completion limit. □

Falsifiable prediction: An accepted $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ consumer that reaches a Real-facing common tail without displaying the dyadic ledger, both StreamName windows, both RegSeqRat readbacks, both CauchyModulus schedules, the RegularCauchyWindowCofinality row, the common-tail certificate, and the structural rows refutes the carrier surface.

Independence witness: cauchymodulus, dyadicratcore, streamname, regseqrat, regular-cauchy-window-cofinality, real: no listed sibling supplies the whole paired packet $M = (\epsilon, D, S_0, S_1, R_0, R_1, \mu_0, \mu_1, W, T, E,$ because each sibling contributes only one schedule, dyadic ledger, stream window, regular readback, cofinal-window row, or terminal seal while this chapter binds two modulus schedules into one common-tail certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCofinality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyModulusCofinality}^{\uparrow}$ packet is generated from one dyadic tolerance ledger, two StreamName windows, two RegSeqRat readbacks, two CauchyModulus schedules, one RegularCauchyWindowCofinality row, a common-tail certificate, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6472.1, Definition 6472.2, Definition 6472.3, and Definition 6472.4 seed the finite paired-modulus common-tail route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired schedule exposure, cofinal-window exactness, common-tail stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyModulusCofi

Chapter 6473

A Concrete Naming Certificate for $\mathsf{BaireUniformSpace}^{\uparrow}$

Following Baire, Kelley, and Bishop–Bridges, the $\mathsf{BaireUniformSpace}^{\uparrow}$ packet records the uniform-space surface of Baire sequences through finite prefix entourages. It sits between the prefix carrier of chapter 334, the common-prefix metric route of chapter 1602, the entourage discipline of chapter 339, and the completion-facing Baire packets of chapter 1884 and chapter 3535. The chapter names the finite uniform interface; it does not claim open-cover compactness, selected branches, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6473.1 (Baire uniform-space carrier)**.** *A* $\mathsf{BaireUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, M, E, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix row, naming two inspected streams and the common prefix window.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *common-prefix distance row, recording the dyadic radius attached to the first displayed disagreement or to agreement on the inspected window.*
3. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row generated by finite prefix agreement requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which prefix coordinates and requested entourage depths are read.*
5. *H records componentwise* hsame *transport for the Baire prefix, metric, entourage, window, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the same prefix-entourage route.*
7. *P is the* Pkg *provenance over* $\mathsf{BaireUniformSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, M, E, S, H, C, P, N. It has no coordinate for open-cover compactness, selected infinite branches, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Theorem 6473.2 (Baire uniform-space NameCert obligations)**.** *For every accepted* $\mathsf{BaireUniformSpace}^{\uparrow}$ *carrier*

$$U = (B, M, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Baire prefix, common-prefix metric, uniform entourage, finite window, transport, replay, provenance, and local naming rows; prefix-entourage stability under componentwise* hsame*; displayed* Cont *locality; and non-escape for open-cover compactness, selected branches, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6473.1. The only analytic rows are the finite prefix row B, common-prefix metric row M, generated entourage row E, and finite window row S.

The uniformity is read through E after B, M, S have displayed the finite prefix request. Transport through H and replay through C preserve the same prefix-entourage route.

Finally P and N source and name the accepted packet. Since the excluded compactness, branch-selection, quotient-equality, choice, and completeness objects have no carrier coordinate, no NameCert projection can export them. □

Theorem 6473.3 (Baire uniform-space complete-metric handoff)**.** *Every* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *or* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *consumer of an accepted* $\mathsf{BaireUniformSpace}^{\uparrow}$ *packet must first read the finite prefix row* B*, common-prefix metric row* M*, generated entourage row* E*, finite* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, componentwise transport row* H*, displayed replay row* C*, provenance row* P*, and local name* N*. The handoff is a finite prefix-entourage route, not a selected limit branch, quotient stream equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the NameCert obligations. They force every uniform-space read to pass through B, M, E, S, H, C, P, N.

A completion-facing consumer can therefore inspect only finite prefix agreement and the common-prefix metric radius before it reaches any complete-Baire row. The generated entourage row E is downstream of those finite observations.

The structural rows preserve the route. Transport, replay, provenance, and local naming do not add selected branches, quotient equality, countable choice, compactness, or ambient completeness, so the handoff remains finite. □

Theorem 6473.4 (Baire uniform-space nonescape)**.** *Every public read from an accepted* $\mathsf{BaireUniformSpace}^{\uparrow}$ *packet factors through*

$$B, M, E, S, H, C, P, N.$$

The packet exports finite prefix-entourage data and no open-cover compactness theorem, chosen branch, quotient stream equality, countable choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Read the public consumer through the displayed continuation row C. Each continuation has a named source among the prefix, metric, entourage, window, transport, provenance, and local naming rows.

Apply the complete-metric handoff theorem for completion-facing consumers and the NameCert obligations for all other public reads. Both routes are exhausted by the same finite carrier.

The refused objects have no coordinate in the carrier and no continuation route, so no public read can export them. □

Falsifiable prediction: Every $\mathsf{BaireUniformSpace}^{\uparrow}$ read factors through displayed Baire prefix rows, common-prefix metric rows, generated UniformSpace entourage rows, finite StreamName windows, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: baire-space, baire-metric, uniform-space, complete-baire-space: $\mathsf{BaireUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite Baire prefixes, common-prefix metric data, generated uniform entourages, finite windows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireUniformSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6473.1, Definition 6473.2, and Definition 6473.3 seed the finite Baire uniform-space route from prefix rows through common-prefix metrics, generated entourages, finite windows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireUniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected branches, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-entourage stability, common-prefix metric compatibility, complete-metric handoff, transport, replay, provenance, and non-escape for BEDC.Derived.BaireUniformSpaceUp.

Constructive story: A $\mathsf{BaireUniformSpace}^{\uparrow}$ packet is generated from BaireSpace prefix rows, BaireMetric common-prefix rows, UniformSpace entourage rows, StreamName finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6474

A Concrete Naming Certificate for MetacicAppArgumentStabilityLedger$^{\uparrow}$

The MetacicAppArgumentStabilityLedger$^{\uparrow}$ packet realizes the dependent-codomain frontier in the MetaCIC subject-reduction obstruction map as finite BEDC rows. It sits between the application typing row of chapter 730, the discharge-obligation socket of chapter 719, the beta-substitution handoff of chapter 716, and the closed-substitution boundary of chapter 538. The packet names the smaller ledger that keeps an application argument type attached to the codomain route; it does not assert global subject reduction, unrestricted substitution, quotient syntax equality, or a completed MetaCIC model.

Definition 6474.1 (MetaCIC app-argument stability ledger carrier)**.** *A* MetacicAppArgumentStabilityLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, X, Y, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the application typing frontier row, exposing the function type, argument type, and application result request.*
2. *X is the argument-type row that must remain attached to the application typing route.*
3. *Y is the dependent-codomain row, naming the codomain expression after the argument row has been displayed.*
4. *D is the discharge-obligation row consumed by the subject-reduction socket.*
5. *S is the closed-substitution boundary row that records the finite substitution route used by the codomain readback.*
6. *R is the reduction-readback row for the application step under the same argument-type boundary.*
7. *H records componentwise* hsame *transport for the frontier, argument, codomain, discharge, substitution, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an application request through $F, X, Y, D, S, R, H, P, N$.*
9. *P is the* Pkg *provenance over* MetacicAppArgumentStabilityLedger$^{\uparrow}$*, MetaCIC application typing rows, discharge-obligation rows, closed-substitution boundary rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, X, Y, D, S, R, H, C, P, N$. It has no coordinate for global subject reduction, unrestricted substitution, quotient syntax equality, host evaluator soundness, proof irrelevance, or a completed MetaCIC semantics.

Theorem 6474.2 (MetaCIC app-argument stability ledger NameCert obligations)**.** *For every accepted* $\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}$ *carrier*

$$A = (F, X, Y, D, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}}$ *surface consists exactly of carrier exposure for the application typing frontier, argument-type row, dependent-codomain row, discharge-obligation row, closed-substitution boundary row, reduction-readback row, componentwise transport, displayed replay, provenance, and local naming; stability of the argument-type row before the codomain route is read; compatibility with the subject-reduction discharge socket; and non-escape for global subject reduction, unrestricted substitution, quotient syntax equality, host evaluator soundness, proof irrelevance, and completed MetaCIC semantics.*

Proof. Project the carrier of Definition 6474.1. The application-facing data are exactly F, X, Y, D, S, R.

The argument-type row X is consumed before the dependent-codomain row Y. The discharge row D, closed-substitution row S, and reduction-readback row R can only replay that same ordering.

Transport through H and replay through C preserve the ordering, while P and N source and name the packet. Since the refused global principles have no carrier coordinate, the NameCert surface remains finite. □

Theorem 6474.3 (MetaCIC app-argument dependent-codomain route)**.** *Every subject-reduction discharge consumer of an accepted* $\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}$ *packet reads the dependent-codomain route only through*

$$F, X, Y, D, S, R, H, C, P, N.$$

The argument-type row X is displayed before the codomain row Y, and the closed-substitution and reduction-readback rows S, R preserve that attachment. The route exports no global subject-reduction theorem, unrestricted substitution calculus, quotient syntax equality, host evaluator soundness, proof-irrelevance principle, or completed MetaCIC model.

Proof. Start with the NameCert obligations. They place the application frontier, argument type, dependent codomain, discharge row, substitution boundary, and readback row on one finite route.

The subject-reduction socket consumes D only after the argument row X and codomain row Y have been displayed. The closed-substitution row S and readback row R preserve that same dependency rather than exposing a general substitution theorem.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the route without adding any of the refused host-side or global MetaCIC objects. □

Falsifiable prediction: Every $\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}$ discharge read factors through displayed application-frontier rows, argument-type rows, dependent-codomain rows, discharge-obligation rows, closed-substitution boundary rows, reduction-readback rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: subject-reduction-frontier, discharge-obligation-socket, beta-substitution-preservation, closed-substitution-boundary: $\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}$ is not bijective

to any listed sibling because it jointly carries argument-type stability, dependent-codomain routing, discharge compatibility, closed-substitution readback, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}$ packet is generated from MetaCIC application-frontier rows, argument-type rows, dependent-codomain rows, discharge-obligation rows, closed-substitution boundary rows, reduction-readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6474.1, Definition 6474.2, and Definition 6474.3 seed the finite application-argument stability ledger for the MetaCIC subject-reduction obstruction route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicAppArgumentStabilityLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global subject reduction, unrestricted substitution, quotient syntax equality, host evaluator soundness, proof irrelevance, completed MetaCIC semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, argument-type stability, dependent-codomain routing, discharge compatibility, closed-substitution readback, transport, replay, provenance, and non-escape for BEDC.Derived.MetacicAppArg

Chapter 6475

A Concrete Naming Certificate for $\mathsf{CauchyRetractionModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRetractionModulus}^{\uparrow}$ packet records the finite surface that turns ordinary Cauchy-modulus data into the regular retraction schedule consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and sealed as $\mathsf{Real}^{\uparrow}$. It sits after the regularization route of chapter 376, the completion retraction packet of chapter 1921, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name boundary of chapter 38. The packet is a finite L10 handoff from inspected Cauchy data to regular rational names; it is not quotient Cauchy-class equality, a selected limit, trichotomy, countable choice, or ambient Real completeness.

Definition 6475.1 (CauchyRetractionModulus carrier)**.** *A* $\mathsf{CauchyRetractionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, R, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy-data row, containing the inspected rational stream and the ordinary Cauchy modulus requests that are available to the packet.*
2. *M is the monotone retraction schedule row, choosing for each displayed tolerance a later index that retracts the source data to a regular Cauchy window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the retracted windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which every regular readback request is inspected.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source modulus budget with the regular-window budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, M, R, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for source data, schedules, regular readbacks, finite windows, dyadic tolerances, Real seals, provenance, and names.*
8. *C records displayed* Cont *replay from the source Cauchy requests to the retracted regular windows and then to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRetractionModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRetractionModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient Cauchy classes, selected limits, trichotomy, countable choice, or ambient Real completeness.

Theorem 6475.2 (CauchyRetractionModulus NameCert obligations)**.** *For every accepted* CauchyRetractionModulu *carrier $K = (S, M, R, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRetractionModulus}^{\uparrow}}$ *surface consists exactly of source Cauchy-data admission through S; inspected monotone retraction scheduling through M; regular rational readback through R; finite StreamName windows through W; DyadicRatCore tolerance comparison through Q; Real sealing through E; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier rows from Definition 6475.1. The accepted packet displays one source Cauchy-data row, one monotone retraction schedule, one regular rational readback row, one finite window row, one dyadic tolerance ledger, one Real seal, and the structural transport, replay, provenance, and naming rows.

The retraction content is carried only by S, M, R, W, Q. The source Cauchy requests in S are inspected before M selects later schedule indices, the windows W expose the finite part of those indices, and R reads the resulting regular rational name under the tolerance comparison Q.

The structural rows preserve the same finite packet under hsame, Cont, Pkg, and $\mathsf{NameCert}$ replay. Since no row carries quotient Cauchy-class equality, a selected limit, trichotomy, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6475.3 (CauchyRetractionModulus RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyRetractionModulus}^{\uparrow}$ *packet factors through the finite route*

$$S, M, W, Q, R, E, H, C, P, N.$$

The handoff presents the regular rational name as the readback of inspected source Cauchy data along the monotone retraction schedule. It exports no quotient Cauchy-class equality, selected limit, trichotomy branch, countable-choice selector, or ambient Real-completeness coordinate.

Proof. Begin with the source and schedule rows. The source Cauchy-data row S displays the available stream requests, and the retraction schedule M chooses later indices before any regular rational readback can be consumed.

Next read the window and tolerance rows. The $\mathsf{StreamName}^{\uparrow}$ row W exposes exactly the finite window inspected by the schedule, while the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger Q records the comparison between the ordinary modulus budget and the regular-window budget.

Finally consume R and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is the readback of the displayed retracted windows, and the terminal $\mathsf{Real}^{\uparrow}$ row E is only a seal for that finite route. The structural rows H, C, P, N transport, replay, package, and name the same route, so no quotient, selected limit, trichotomy, choice, or ambient-completeness coordinate is inserted. □

Falsifiable prediction: An accepted $\mathsf{CauchyRetractionModulus}^{\uparrow}$ consumer that reaches a RegSeqRat readback or Real seal without displaying the source Cauchy data, monotone retraction schedule, StreamName windows, DyadicRatCore tolerance ledger, structural transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: cauchyregularization, cauchy-completion-retraction, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole finite packet $K = (S, M, R, W, Q, E, H, C, P, N)$,

because each sibling contributes only source regularization, completion retraction, finite windows, regular readback, tolerance comparison, or terminal sealing while this chapter binds them into one inspected retraction-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRetractionModulus}\uparrow}$.

Constructive story: A $\mathsf{CauchyRetractionModulus}^{\uparrow}$ packet is generated from source Cauchy data, an inspected monotone retraction schedule, finite StreamName windows, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRetractionModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6475.1, Definition 6475.2, and Definition 6475.3 seed the finite route from ordinary Cauchy-modulus data to regular rational names and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRetractionModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy-class equality, selected limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source Cauchy-data exposure, monotone retraction scheduling, finite-window coverage, RegSeqRat readback, DyadicRatCore tolerance comparison, Real-seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRetractionModulusUp.

Chapter 6476

A Concrete Naming Certificate for $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ packet records cotransitivity for located real apartness as finite NameCert data. It sits over the located-real carrier of chapter 1933, the located apartness packet of chapter 1380, the Bishop-style cotransitivity route of chapter 1788, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet exposes the third-name cotransitivity branch without importing trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6476.1 (Located real apartness cotransitivity carrier)**.** *A* $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, A, L, R, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source* $\mathsf{LocatedReal}^{\uparrow}$ *name rows whose apartness is being read.*
2. *Z is the displayed third* $\mathsf{LocatedReal}^{\uparrow}$ *name through which cotransitivity is tested.*
3. *A is the* $\mathsf{LocatedRealApartness}^{\uparrow}$ *source-apartness row for X and Y.*
4. *L is the left cotransitivity branch, recording the displayed located-apartness read from X to Z when that branch is active.*
5. *R is the right cotransitivity branch, recording the displayed located-apartness read from Z to Y when that branch is active.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the three located real names at the requested tolerance.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the positive separation budget, slack, and branch comparison cells.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, D and one of L, R have been displayed.*
10. *H records componentwise* hsame *transport for the three names, source apartness, branch rows, stream windows, regular readback, dyadic tolerance, Real seal, replay, provenance, and local naming.*

11. *C records displayed* Cont *replay from a located-apartness cotransitivity request through the same finite route.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{ApartnessCotransitivity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for trichotomy of real names, quotient real equality, selected branch choice, countable choice, host locatedness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6476.2 (Located real apartness cotransitivity NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ *carrier*

$$K = (X, Y, Z, A, L, R, S, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}}$ *surface consists exactly of carrier admission for the three located real names, source located-apartness admission, left-or-right branch admission, finite stream-window exposure, regular rational readback, dyadic tolerance preservation, terminal Real seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The visible data are the three located real names, the source apartness row, the two possible branch rows, stream windows, regular readback, dyadic tolerance, terminal seal, and the structural rows.

The source and branch obligations are finite. The packet first reads A as the displayed located-apartness source; it then exposes exactly one branch row among L and R as the cotransitivity witness.

The Real-facing part is downstream. Stream windows S, regular readback Q, and the dyadic tolerance row D must be displayed before the terminal seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier row stores trichotomy, quotient equality, a selected external branch, countable choice, host locatedness, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6476.3 (Located real apartness cotransitivity route)**.** *Every accepted* $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{LocatedRealApartness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{ApartnessCotransitivity}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a downstream apartness or order consumer receives a source located-apartness row, a displayed third located real name, one cotransitive branch, finite windows, regular rational readback, dyadic tolerance data, and the terminal real-name seal before it may use the cotransitivity packet.

Proof. Start from the carrier of Definition 6476.1. The source apartness row A and third name Z are the only inputs to the cotransitivity read.

The observation route opens S, Q, D before either branch is inspected. These rows keep the branch comparison inside finite windows, regular rational readback, and dyadic tolerance budgets.

The active branch then invokes the cotransitivity surface of chapter 1788. It is read as a displayed branch over the same names, not as a trichotomy decision.

The terminal row E seals the read for $\mathsf{Real}^{\uparrow}$ consumers, while H, C, P, N preserve the route. Hence every accepted read follows the displayed dependency path and no host apartness or quotient-real object is exported. □

Theorem 6476.4 (Located real apartness cotransitivity Real seal nonescape)**.** *For an accepted* $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ *carrier* $K = (X, Y, Z, A, L, R, S, Q, D, E, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *exports only the branch already displayed through* A*, one of* L, R*,* S*,* Q*, and* D*. It exports no host trichotomy, quotient real equality, selected real representative, countable-choice selector, ambient locatedness theorem, or completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Inspect the terminal coordinate. The row E is listed after the source apartness, branch, stream-window, regular-readback, and dyadic-tolerance rows.

Use Definition 6476.2. It forces Real-seal ordering through the displayed rows before any Real-facing consumer can read the packet.

The structural rows preserve that boundary. Componentwise hsame transport and displayed Cont replay do not add a second branch source, and Pkg provenance with local naming records only the same finite carrier.

Therefore the terminal seal cannot export trichotomy, quotient equality, selected representatives, countable choice, host locatedness, or ambient completeness. A consumer asking for one of those principles leaves the accepted carrier surface. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ read must expose three LocatedReal name rows, a LocatedRealApartness source row, one cotransitive branch row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any Real or order consumer reads the branch.

Independence witness: located-real-apartness, apartness-cotransitivity, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ because this packet jointly binds located real names, a source located-apartness row, one cotransitive branch, finite windows, regular readback, dyadic tolerance, and terminal Real-seal nonescape.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}$ packet is generated from three LocatedReal name rows, a LocatedRealApartness source row, one displayed cotransitivity branch, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6476.1, Definition 6476.2, Definition 6476.3, and Definition 6476.4 seed the finite located-real cotransitivity packet over displayed branch, window, readback, tolerance, seal, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealApartnessCotransitivity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, ambient locatedness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source located-apartness exposure, branch stability, finite-window readback, dyadic tolerance preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealApartnessCotransitivityUp.

Chapter 6477

A Concrete Naming Certificate for $\mathsf{HyperbolicUniformModulus}^{\uparrow}$

The $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ packet realizes the uniform-modulus branch of chapter 9463 as finite hyperbolic metric, cross-ratio, visual-boundary, and tolerance-only modulus rows. It sits after the Poincare disk metric packet of chapter 6209, the hyperbolic metric surface of chapter 3474, the boundary comparison packet of chapter 3948, the visual boundary carrier of chapter 4181, and the compact uniform selector of chapter 6466. The carrier is a BEDC-internal synthesis: it binds hyperbolic boundary geometry to a tolerance-only modulus route without turning visual expansion into a host compactness theorem or a completed boundary object.

Definition 6477.1 (Hyperbolic uniform modulus carrier)**.** *A* $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, X, B, V, Q, T, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the displayed disk or boundary-shadow observations.*
2. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *comparison row, recording the finite four-point or boundary-ratio data used by the modulus read.*
3. *B is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *boundary row, exposing the visual-boundary source of the observer-expansion tolerance request.*
4. *V is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric provenance row tying the metric and boundary observations to the disk model used by the vision.*
5. *Q is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance-only modulus row delivered to observer-expansion consumers.*
6. *T is the compact-uniform handoff row, read through* $\mathsf{CompactUniformModulus}^{\uparrow}$ *or the finite selector route of chapter 6466.*
7. *D is the dyadic or rational tolerance ledger used to compare cross-ratio, visual-boundary, and modulus requests.*
8. *R is the refusal row excluding host boundary compactness, completed hyperbolic plane geometry, quotient ideal points, selected visual metrics, and physical observer expansion.*

9. *H records componentwise* hsame *transport for metric, cross-ratio, visual-boundary, disk-metric, uniform-modulus, compact-uniform, tolerance, refusal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from hyperbolic boundary observations through M, X, B, V, T, D to the tolerance-only row Q.*
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicUniformModulus}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host compactness theorem, quotient boundary equality, selected visual metric, completed hyperbolic geometry, physical expansion law, or observer substance.

Theorem 6477.2 (Hyperbolic uniform modulus NameCert obligations)**.** *For every accepted* HyperbolicUniformMod *carrier*

$$U = (M, X, B, V, Q, T, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicUniformModulus}^{\uparrow}}$ *surface consists exactly of hyperbolic metric admission, hyperbolic cross-ratio admission, visual-boundary admission, Poincare disk metric provenance, tolerance-only uniform modulus exposure, compact-uniform handoff, tolerance-ledger compatibility, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The hyperbolic geometry rows are M, X, B, V, and the tolerance rows are Q, T, D.

The metric and cross-ratio rows are read before the modulus row. This keeps the boundary comparison visible when the consumer receives a tolerance-only threshold.

The compact-uniform handoff is finite. The row T records the selector or compact uniform modulus route, while D records the tolerance ledger used to compare the geometric and modulus requests.

The refusal and structural rows preserve the boundary. Since no coordinate supplies host compactness, quotient boundary equality, selected visual metrics, completed hyperbolic geometry, physical expansion, or observer substance, the listed rows exhaust the local NameCert surface. □

Theorem 6477.3 (Hyperbolic uniform modulus cross-ratio route)**.** *Every accepted* $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ *read delivers its tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *row only after the displayed route*

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, CompactUniformModul

has been opened. The observer-expansion consumer receives metric, cross-ratio, visual-boundary, disk-metric, compact-uniform, and tolerance-ledger rows before any modulus threshold can be used.

Proof. Begin with $U = (M, X, B, V, Q, T, D, R, H, C, P, N)$. The row Q is the only tolerance-only uniform modulus coordinate.

The route to Q is ordered by the carrier inventory. The consumer first reads the metric row M, cross-ratio row X, visual-boundary row B, disk-metric provenance V, compact-uniform handoff T, and tolerance ledger D.

Only then can Q be replayed by C and transported by H. The provenance row P and local naming row N record the same finite route.

Thus the modulus threshold is never detached from its hyperbolic boundary source, and no host compactness, selected visual metric, quotient boundary, completed geometry, or physical expansion law is exported. □

Theorem 6477.4 (Hyperbolic uniform modulus boundary nonescape)**.** *For an accepted* HyperbolicUniformModulus *carrier* $U = (M, X, B, V, Q, T, D, R, H, C, P, N)$*, the visual-boundary row* B *and tolerance-only row* Q *export no completed boundary compactification, quotient ideal point, selected visual metric, host compactness theorem, or physical observer-expansion datum. Every boundary-to-modulus read remains inside the displayed metric, cross-ratio, disk-metric, compact-uniform, tolerance, refusal, transport, replay, provenance, and naming rows.*

Proof. Inspect the carrier surface. The visual-boundary row B is accompanied by metric row M, cross-ratio row X, disk-metric provenance V, compact-uniform handoff T, tolerance ledger D, and refusal row R.

Apply the NameCert obligations of Definition 6477.2. They require the tolerance-only row Q to be exposed only through the same displayed rows.

The refusal row rules out stronger host readings, and H, C, P, N preserve the route under transport, replay, provenance, and local naming.

Consequently a consumer cannot extract completed boundary compactification, quotient ideal points, selected visual metrics, host compactness, or physical expansion from the packet. Such data are absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ read must expose $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ rows before any observer-expansion consumer receives a tolerance-only modulus.

Independence witness: hyperbolic_metric, hyperbolic_cross_ratio, hyperbolic_visual_boundary, compact_uniform_modulus, poincare_disk_metric: no listed sibling is bijective to $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ because this packet jointly binds metric reads, cross-ratio comparisons, visual-boundary rows, disk-metric provenance, compact-uniform handoff, a tolerance-only modulus, refusal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicUniformModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicUniformModulus}^{\uparrow}$ packet is generated from HyperbolicMetric reads, HyperbolicCrossRatio comparisons, HyperbolicVisualBoundary rows, PoincareDiskMetric provenance, CompactUniformModulus handoff, UniformModulus tolerance rows, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6477.1, Definition 6477.2, Definition 6477.3, and Definition 6477.4 seed the finite route from hyperbolic metric and boundary comparison rows to a tolerance-only uniform modulus.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host boundary compactness, quotient ideal-boundary equality, selected visual metrics, completed hyperbolic geometry, physical expansion, observer substance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-read exposure, cross-ratio route, visual-boundary nonescape, compact-uniform handoff, tolerance-only modulus exactness, refusal, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicUniformModulusUp.

Chapter 6478

A Concrete Naming Certificate for $\mathsf{LocatedInfimumModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedInfimumModulus}^{\uparrow}$ packet records the finite data by which a located lower-bound search exposes an infimum modulus before any ambient completeness principle for $\mathsf{Real}^{\uparrow}$ is available. It sits beside the located-infimum carrier of chapter 1452, the dyadic tolerance route of chapter 291, the regular rational readback of chapter 301, and the real seal of chapter 38. The packet extracts lower-window brackets and dyadic tolerances; it does not assert trichotomy, quotient real equality, selected infima, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6478.1 (Located infimum modulus carrier)**.** *A* $\mathsf{LocatedInfimumModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, L, U, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the consumed* $\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$ *row, recording the lower-bound ledger and greatest-lower-approximation ledger that the modulus request may inspect.*
2. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *row, storing the displayed lower surface and its admitted lower-window observations.*
3. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *row, storing the displayed upper guard used to bracket the lower search window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing dyadic precision requests, lower-window refinements, and width certificates.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating dyadic lower-window refinements into regular rational observations before the real seal is consumed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the located-infimum, lower-real, upper-real, tolerance, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the located-infimum row, lower and upper real rows, dyadic tolerance row, regular-rational readback, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an infimum-modulus request through $I, L, U, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedInfimumModulus}^{\uparrow}$, $\mathsf{LocatedInfimum}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedInfimumModulus}^{\uparrow}}$ *row.*

The carrier exports only the displayed modulus route $I, L, U, D, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality of real names, selected infima, countable choice, a host infimum operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6478.2 (Located infimum modulus NameCert obligations)**.** *For every accepted* $\mathsf{LocatedInfimumModulus}^{\uparrow}$ *carrier*

$$M = (I, L, U, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedInfimumModulus}^{\uparrow}}$ *surface consists exactly of located-infimum admission through I, lower-real window exposure through L, upper-real guard exposure through U, dyadic tolerance exactness through D, regular rational readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6478.1. The analytic rows are exactly the located-infimum row, the lower and upper real rows, the dyadic tolerance row, the regular rational readback row, and the terminal real seal row.

The lower-window read is forced through I, L, U. The located-infimum row supplies the lower-bound and greatest-lower-approximation ledgers, while L and U display the bracketing surfaces used by the modulus request.

The dyadic tolerance row supplies the modulus data. It refines lower-window widths through $\mathsf{DyadicRatCore}^{\uparrow}$ entries and exposes those refinements as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks before the $\mathsf{Real}^{\uparrow}$ seal may be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier row stores trichotomy, quotient equality, selected infima, countable choice, a host infimum operator, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6478.3 (Located infimum modulus real seal)**.** *Let $M = (I, L, U, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedInfimumModulus}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by M factors through the displayed route*

$$I, L, U \longmapsto D \longmapsto R \longmapsto E \longmapsto C, P, N.$$

Thus a consumer can read only the sealed regular-rational lower-window route, not a quotient real equality test, a selected infimum, a countable-choice witness, or an ambient completeness theorem.

Proof. Start with a $\mathsf{Real}^{\uparrow}$-facing request. The carrier contains only one real seal row, namely E, and the definition makes E reachable only after the located-infimum, lower-real, upper-real, dyadic tolerance, and regular-rational rows have been displayed.

The request first reads the lower-window source I, L, U. It then passes through the tolerance row D, which supplies the dyadic precision control, and through R, which repacks the same finite observations as regular rational readback.

The structural rows expose no additional real object. The continuation row C, provenance row P, and local naming row N can only replay, source, and name the sealed route, so the exported read remains inside $I, L, U, D, R, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedInfimumModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedInfimumModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6478.1, Definition 6478.2, and Definition 6478.3 seed the finite located infimum-modulus route over located-infimum, lower-real, upper-real, dyadic tolerance, regular-rational readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedInfimumModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected infima, countable choice, host infimum operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-window bracketing, dyadic tolerance refinement, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedInfimumModul

Constructive story: A $\mathsf{LocatedInfimumModulus}^{\uparrow}$ packet is generated from a LocatedInfimum row, LowerReal and UpperReal bracketing rows, dyadic tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 6479

A Concrete Naming Certificate for $\mathsf{BingMetrization}^{\uparrow}$

Following Bing and the Nagata–Smirnov metrization tradition, the $\mathsf{BingMetrization}^{\uparrow}$ packet records the finite BEDC route by which a developable regular topological source exposes a metric handoff through displayed star-refinement and cover-separation rows. It sits after the Moore-space development surface of chapter 3592, beside the regular locally finite cover route of chapter 6348, over the topology carrier of chapter 88, and before the metric-space consumer surface of chapter 4230. The packet names only the displayed Bing-style metrization route; it does not assert host topology equality, selected infinite covers, paracompactness as an external theorem, quotient point equality, or ambient real completeness.

Definition 6479.1 (Bing metrization carrier)**.** *A* $\mathsf{BingMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, M, D, S, R, L, E, Q, G, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the open-row classifier, point-neighbourhood requests, and finite membership reads.*
2. *M is the* $\mathsf{MooreSpace}^{\uparrow}$ *development row, exposing the countable development windows and local star-neighbourhood requests inherited from chapter 3592.*
3. *D is the displayed development-level schedule. It records the finite levels inspected by the packet and the point rows whose stars are tested.*
4. *S is the star-refinement row, binding each accepted neighbourhood request to a displayed star around the named point.*
5. *R is the regularity-separation row, recording finite closed-neighbourhood separation data read from the topology source.*
6. *L is the locally finite cover ledger shared with the Nagata–Smirnov route of chapter 6348.*
7. *E is the cover-shrink row, tying S, R, L to the finite separation checks used before a metric row can be consumed.*
8. *Q is the dyadic radius schedule used by the metric handoff.*
9. *G is the* $\mathsf{Met}^{\uparrow}$ *handoff row. It is available only after T, M, D, S, R, L, E, Q have been exposed.*

10. *F is the topology-metric compatibility row, recording that displayed metric balls from G answer exactly the topology windows read through T, M, D, S, R, L, E.*
11. *H records componentwise* hsame *transport for all displayed rows.*
12. *C records displayed* Cont *replay from a topology request through development, star-refinement, regularity, locally finite covers, shrink data, dyadic radii, metric handoff, compatibility, provenance, and local naming.*
13. *P is the* Pkg *provenance over* $\mathsf{BingMetrization}^{\uparrow}$, $\mathsf{MooreSpace}^{\uparrow}$, $\mathsf{NagataSmirnovMetrization}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{BingMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for host topology equality, selected infinite covers, paracompactness as an external theorem, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6479.2 (Bing metrization NameCert obligations)**.** *For every accepted* $\mathsf{BingMetrization}^{\uparrow}$ *carrier $B = (T, M, D, S, R, L, E, Q, G, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BingMetrization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, Moore development row, level schedule, star-refinement row, regularity-separation row, locally finite cover ledger, cover-shrink row, dyadic radius schedule, metric-space handoff, topology-metric compatibility, transport, replay, provenance, and local naming row;*
2. *development readability, requiring every Bing-style neighbourhood request to pass through T, M, D, S before any cover-shrink or metric row is inspected;*
3. *regular cover separation, requiring the rows R, L, E to expose the finite closed-neighbourhood and locally finite cover checks that license the metric handoff;*
4. *metric handoff exactness, requiring Q, G, F to keep the* $\mathsf{Met}^{\uparrow}$ *row downstream of the displayed development and cover-separation route;*
5. *refusal, requiring P and N to name only this finite packet and to refuse host topology equality, selected infinite covers, external paracompactness, quotient point equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 6479.1. The projection exposes exactly the topology source, Moore development row, development-level schedule, star-refinement row, regularity-separation row, locally finite cover ledger, shrink row, dyadic radius schedule, metric-space handoff, topology-metric compatibility row, and the structural rows H, C, P, N.

Development readability is forced by the row order. A neighbourhood request starts at T, passes to the Moore development row M, chooses only the displayed finite schedule D, and then reads the star-refinement witness S.

The metric handoff is downstream of the separation route. Rows R, L, E record the closed-neighbourhood and cover-shrink checks, while Q, G, F record the dyadic radius schedule, the metric-space handoff, and its finite compatibility with the original topology windows.

Since the tuple contains no host topology equality, selected infinite cover, external paracompactness theorem, quotient point equality, countable-choice row, or ambient completeness coordinate, the local NameCert cannot export those principles. □

Theorem 6479.3 (Bing metrization metric handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{BingMetrization}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{MooreSpace}^{\uparrow},\ D,\ S,\ R,\ L,\ E,\ Q,\ G,\ F,\ H,\ C,\ P,\ N.$$

The metric-space row G is consumed as a downstream handoff tied to the displayed development stars, regular separation rows, locally finite cover ledger, and topology compatibility windows. It is not a host metric, not a selected infinite cover construction, and not a claim that the source topology equals an external topology.

Proof. Read the carrier rows in order. The topology source and Moore development rows are displayed before a star-refinement request can be answered, so the metric-facing route remains attached to the source topology and its development data.

Next read the separation and cover rows. The regularity row R, locally finite ledger L, and shrink row E provide the finite Bing-style cover-separation data needed before a dyadic radius schedule or metric-space handoff is available.

The rows Q, G, F then supply the downstream metric interface and its topology compatibility check. Transport and replay through H and C preserve exactly the same route, while P and N record the sibling dependencies and local naming row.

The carrier contains no host topology equality, selected infinite cover, external paracompactness theorem, quotient point equality, countable-choice row, or ambient completeness coordinate. Therefore the $\mathsf{Met}^{\uparrow}$ consumer receives only the finite BEDC handoff named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BingMetrization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BingMetrization}^{\uparrow}$ packet is generated from a topology source row, Moore development row, development-level schedule, star-refinement row, regularity-separation row, locally finite cover ledger, cover-shrink row, dyadic radius schedule, MetricSpace handoff, topology-metric compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BingMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6479.1, Definition 6479.2, and Definition 6479.3 seed the finite developable regular topology route to a displayed MetricSpace handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BingMetrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topology equality, selected infinite covers, external paracompactness, quotient point equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, development readability, regular cover separation, cover-shrink exactness, metric handoff exactness, transport, replay, provenance, and local naming for the displayed Bing metrization packet.

Chapter 6480

A Concrete Naming Certificate for $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$

Following Feller and Billingsley on probability generating functions for integer-valued distributions, the $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$ packet records the finite coefficient-window route by which a probability law over $\mathsf{Nat}^{\uparrow}$ is read as a formal series before any scalar consumer asks for a $\mathsf{Real}^{\uparrow}$ seal. It sits after the probability-space rows of chapter 184, the distribution rows of chapter 186, the series carrier of chapter 121, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite probability-mass windows, natural-index reads, coefficient streams, regular rational handoff, transport, replay, provenance, and local naming; it does not assert analytic convergence on a host disk, selected infinite sums, countable additivity beyond displayed rows, quotient stream equality, or ambient real completeness.

Definition 6480.1 (Probability generating function carrier)**.** *A* $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, D, I, S, W, R, E, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *row that names the finite probability context for the displayed event and mass reads.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *row whose integer-valued mass table supplies the inspected coefficient window.*
3. *I is the* $\mathsf{Nat}^{\uparrow}$ *index row naming the finite natural-number slots whose probabilities are exposed.*
4. *S is the* $\mathsf{Series}^{\uparrow}$ *coefficient-stream row, reading the displayed masses as the corresponding formal coefficient window.*
5. *W is the finite coefficient-window ledger connecting D, I to S without selecting an infinite stream.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected probability coefficients and finite partial evaluations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exported only after P, D, I, S, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the probability, distribution, index, series, window, readback, seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a coefficient or evaluation request through P, D, I, S, W to R and E.*
10. *K is the* Pkg *provenance over* $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProbabilityGeneratingFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, D, I, S, W, R, E, H, C, K, N$. It has no coordinate for a host analytic function, a completed probability law outside the displayed distribution row, selected infinite coefficients, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6480.2 (Probability generating function NameCert obligations)**.** *For every accepted* $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$ *carrier $G = (P, D, I, S, W, R, E, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProbabilityGeneratingFunc}}$ *surface consists exactly of probability-space admission through P; distribution-row admission through D; natural-index exposure through I; coefficient-stream exposure through S; finite coefficient-window exhaustion through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *seal ordering through E; and componentwise transport, continuation replay, package provenance, and local naming through H, C, K, N.*

Proof. Project the carrier of Definition 6480.1. The rows P, D, I, S, W display the probability context, the integer-valued distribution, the finite natural-index slots, and the coefficient-stream window before any scalar readback is available.

The regular rational row R is downstream from that displayed coefficient window, and the Real seal E is downstream from R. Thus a scalar consumer can only read probabilities and partial evaluations along the route already exposed by the finite coefficient data.

Transport through H and replay through C preserve the same displayed route. Provenance and local naming through K, N source and name it without adding host analytic semantics, selected infinite coefficients, quotient stream equality, countable choice, or ambient completeness. □

Theorem 6480.3 (Probability generating function coefficient-window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ProbabilityGeneratingFunction}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ProbSpace}^{\uparrow}, \quad \mathsf{Distribution}^{\uparrow}, \quad \mathsf{Nat}^{\uparrow}, \quad \mathsf{Series}^{\uparrow}, \quad W, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the same packet rows $P, D, I, S, W, R, E, H, C, K, N$. Consequently the generating-function read is a coefficient-window handoff and not a host analytic function, selected infinite series, quotient stream equality, countable-choice selector, or ambient Real-completeness principle.

Proof. Begin with an accepted carrier and project the source rows P, D, I, S, W. The probability-space row and distribution row supply the finite mass table, while the natural-index row fixes exactly which coefficients the series row may expose.

Read the scalar handoff through R and E. The regular rational readback is obtained from the same finite coefficient windows, and the Real seal is visible only after that readback has been displayed.

Finally pass through H, C, K, N. These rows transport, replay, source, and name the same route, so no consumer can replace it by host analytic convergence, selected representatives, quotient stream equality, countable choice, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProbabilityGeneratingFunction}^{\uparrow}}$.

Constructive story: A ProbabilityGeneratingFunction$^{\uparrow}$ packet is generated from ProbSpaceUp rows, DistributionUp mass-table rows, NatUp index rows, SeriesUp coefficient-stream windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ProbabilityGeneratingFunction}^{\uparrow}}$, seed)

Scope closed: Definition 6480.1, Definition 6480.2, and Definition 6480.3 seed the finite probability generating function route over probability, distribution, natural-index, series, regular rational, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{ProbabilityGeneratingFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic convergence on a host disk, selected infinite sums, quotient stream equality, countable choice, countable-additivity claims beyond displayed rows, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, probability-space admission, distribution-row admission, natural-index exposure, coefficient-window exactness, regular readback, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.ProbabilityGeneratingFunctionUp.

Chapter 6481

A Concrete Naming Certificate for HyperbolicPhaseEnergyLedger$^{\uparrow}$

The HyperbolicPhaseEnergyLedger$^{\uparrow}$ packet realizes the phase-energy budget branch of chapter 9463 as finite reversible-transport, metric-budget, boundary-refusal, Real-seal, provenance, and NameCert rows. It sits after the hyperbolic metric surface of chapter 3474, the reversible phase-transport rows of chapter 219, the flow record surface of chapter 195, the terminal scalar seal of chapter 38, and the conservation ledger of chapter 3487. The packet names a BEDC-internal budget ledger: energy is readable only as displayed metric, transport, conservation, refusal, readback, and provenance data, not as a host Hamiltonian scalar or completed symplectic phase space.

Definition 6481.1 (Hyperbolic phase-energy ledger carrier)**.** *A* HyperbolicPhaseEnergyLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, U, D, L, B, R, S, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-budget row over the displayed phase-surface observations.*
2. *U is the* UnitaryGroup$^{\uparrow}$ *reversible-transport row recording the classifier-preserving phase move.*
3. *D is the* DynSystem$^{\uparrow}$ *finite flow row through which the transported surface is replayed.*
4. *L is the* LiouvilleLedgerConservation$^{\uparrow}$ *conservation row, requiring source distinctions and route volume to remain auditable under the reversible flow.*
5. *B is the boundary-refusal row excluding completed phase space, host symplectic volume, physical energy conservation, quotient boundary equality, and unledgered loss of source distinctions.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for displayed metric and phase-budget observations.*
7. *S is the terminal* Real$^{\uparrow}$ *seal row exposed only after M, U, D, L, B, R have been displayed.*
8. *F is the TasteGate-facing row requiring the future Lean file for* HyperbolicPhaseEnergyLedger$^{\uparrow}$ *to provide* `Nontrivial` *and* `FieldFaithful` *instances for the metric, transport, flow, conservation, refusal, readback, seal, replay, provenance, and naming fields.*
9. *H records componentwise* hsame *transport for metric-budget, reversible-transport, flow, conservation, refusal, readback, seal, TasteGate-facing, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through the route*

$$M, U, D \longmapsto L, B \longmapsto R, S.$$

11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, D, L, B, R, S, F, H, C, P, N$. It has no coordinate for host Hamiltonian mechanics, completed symplectic volume, selected global phase orbit, physical observer dynamics, quotient boundary equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6481.2 (Hyperbolic phase-energy ledger obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ *carrier $E = (M, U, D, L, B, R, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}}$ *surface consists exactly of hyperbolic metric-budget admission, reversible phase-transport admission, dynamic-flow replay, Liouville ledger conservation, boundary-refusal exactness, regular rational readback, Real-seal ordering, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6481.1. The rows M, U, D display the metric budget, reversible transport, and finite flow route before any conservation or scalar readback row is available.

Read the conservation and refusal rows next. The row L records conservation only as an auditable ledger condition, while B prevents that condition from being read as host symplectic volume, physical energy conservation, quotient boundary equality, or a completed phase space.

The readback row R and Real seal S are downstream from the displayed metric, transport, flow, conservation, and refusal rows. The TasteGate-facing row F keeps those fields noncollapsed for the future Lean carrier.

Rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a host Hamiltonian scalar, selected global orbit, ambient completion, or bridge theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6481.3 (Hyperbolic phase-energy Real-seal order)**.** *Every accepted* $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ *read that exports a* $\mathsf{Real}^{\uparrow}$ *phase-energy seal must first display*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \quad \mathsf{UnitaryGroup}^{\uparrow}, \quad \mathsf{DynSystem}^{\uparrow}, \quad \mathsf{LiouvilleLedgerConservation}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

In carrier notation, the row S is reachable only after $M, U, D, L, B, R, H, C, P, N$ have been read, and the TasteGate-facing row F requires the future Lean $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ *file to keep the carrier fields* `Nontrivial` *and* `FieldFaithful`.

Proof. Begin with the metric, transport, and flow rows. The phase-energy seal has no source until M, U, D expose a displayed hyperbolic metric budget, reversible phase move, and dynamic replay route.

Use the conservation and refusal rows next. The Liouville ledger L preserves source distinctions under the reversible route, while B excludes host phase-space and physical-energy readings.

Only then can the regular rational readback R and Real seal S be consumed. Transport, replay, provenance, and local naming through H, C, P, N keep the same route visible, and F records the future Lean field-separation requirements. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ packet exposes $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, and $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ rows before any $\mathsf{Real}^{\uparrow}$ phase-energy seal; a read that reaches the seal without those rows refutes the carrier surface.

Independence witness: hyperbolic-metric, unitarygroup, dynsystem, real, liouville-ledger-conservation: no listed sibling is bijective to $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ because this packet jointly carries metric-budget rows, reversible phase transport, finite flow replay, Liouville conservation, boundary refusal, regular readback, Real sealing, TasteGate-facing field separation, transport, provenance, and local naming.

The Lean TasteGate file `lean4/BEDC/Derived/HyperbolicPhaseEnergyLedgerUp/TasteGate.lean` must provide `Nontrivial` and `FieldFaithful` instances for $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$. Those instances must preserve $M, U, D, L, B, R, S, F, H, C, P, N$ as distinct fields before any carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ packet is generated from HyperbolicMetricUp metric-budget rows, UnitaryGroupUp reversible-transport rows, DynSystemUp flow replay, LiouvilleLedgerConservationUp conservation rows, boundary refusal, RegSeqRatUp readback, a RealUp seal, TasteGate-facing field-separation requirements, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6481.1, Definition 6481.2, and Definition 6481.3 seed the finite phase-energy ledger route over hyperbolic metric, reversible transport, dynamic flow, Liouville conservation, refusal, readback, Real-seal, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hamiltonian mechanics, completed symplectic volume, physical energy conservation, selected global phase orbits, quotient boundary equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, metric-budget exposure, reversible-transport admission, flow replay, Liouville conservation, boundary-refusal exactness, Real-seal ordering, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseEnergyLedgerUp.

Chapter 6482

A Concrete Naming Certificate for $\mathsf{Hyperfunction}^{\uparrow}$

Following Sato, the $\mathsf{Hyperfunction}^{\uparrow}$ packet records a generalized-function boundary value as finite BEDC data rather than as an ambient quotient of holomorphic functions. It sits over the sheaf route of chapter 151, the cohomology ledger of chapter 99, the complex-analysis carrier of chapter 39, the topology source of chapter 88, and the distribution-facing consumer surface of chapter 186. The packet names the displayed boundary-value route only; it does not export host analytic continuation, quotient sheaf cohomology, selected representatives, distribution equality, or ambient completeness.

Definition 6482.1 (Hyperfunction carrier)**.** *A* $\mathsf{Hyperfunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (T, O, B, S, Q, A, R, D, E, K, L, M, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row for the underlying real-domain carrier, including finite open-set membership requests and neighbourhood probes.*
2. *O is the open-domain row, displaying the finite real open windows on which boundary values are requested.*
3. *B is the boundary-value row. It records the upper and lower complex-side neighbourhoods, their boundary approach data, and the finite comparison window used by the read.*
4. *S is the* $\mathsf{Sheaf}^{\uparrow}$ *row, carrying restriction, locality, and gluing data for the displayed open windows.*
5. *Q is the* $\mathsf{Cohomology}^{\uparrow}$ *row, recording the finite coboundary and cocycle ledger consumed by the hyperfunction read.*
6. *A is the* $\mathsf{Complex}^{\uparrow}$ *analytic row, containing the complex-domain readback and holomorphic-window evidence available to the packet.*
7. *R is the real readback row, tying the boundary window to the real-domain request without selecting a host real function.*
8. *D is the* $\mathsf{Distribution}^{\uparrow}$ *handoff row, available only as a downstream consumer of the boundary-value, sheaf, cohomology, complex, and real readback rows.*

9. *E is the exactness row, recording that the displayed boundary-value read is unchanged by packet-local coboundary replacement.*
10. *K records componentwise* hsame *transport for the topology, open-domain, boundary-value, sheaf, cohomology, complex, real readback, distribution handoff, and exactness rows.*
11. *L records displayed* Cont *replay from a real-domain request through open windows, complex boundary values, sheaf restrictions, cohomology ledgers, exactness, downstream readback, and local naming.*
12. *M is the* Pkg *provenance over* $\mathsf{Hyperfunction}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
13. *C is the complex/real compatibility row, requiring the complex-side boundary window and the real-domain readback to be consumed as one finite route.*
14. *P is the replay provenance row for sibling consumers that use the distribution-facing handoff.*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{Hyperfunction}^{\uparrow}}$ *row.*

The carrier has no coordinate for host analytic continuation, quotient sheaf cohomology, selected boundary representatives, distribution equality, global dual-space topology, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HyperfunctionUp.TasteGate.HyperfunctionTasteGate single carrier alignment`

Theorem 6482.2 (Hyperfunction NameCert obligations)**.** *For every accepted* $\mathsf{Hyperfunction}^{\uparrow}$ *carrier* $H = (T, O, B, S, Q, A, R, D, E, K, L, M, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Hyperfunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, open-domain row, boundary-value row, sheaf row, cohomology ledger, complex analytic row, real readback row, distribution handoff, exactness row, transport, replay, package provenance, compatibility, consumer provenance, and local naming;*
2. *boundary readability, requiring every real-domain request to pass through T, O, B before any sheaf or cohomology row is inspected;*
3. *sheaf-cohomology exactness, requiring S, Q, E to witness restriction, locality, gluing, cocycle, coboundary, and packet-local replacement data before a hyperfunction read is exported;*
4. *complex/real compatibility, requiring A, R, C to bind the complex boundary window to the displayed real-domain readback;*
5. *downstream refusal, requiring D, M, P, N to expose only the distribution-facing handoff named by the packet and to refuse host analytic continuation, quotient sheaf cohomology, selected representatives, distribution equality, global dual-space topology, and ambient completeness.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.HyperfunctionUp.HyperfunctionCarrier namecert obligations`

Proof. Project the carrier of Definition 6482.1. The projection exposes the topology source, open-domain row, boundary-value row, sheaf row, cohomology ledger, complex analytic row, real readback row, distribution handoff, exactness row, and structural rows K, L, M, C, P, N.

Boundary readability is forced by the row order. A real-domain request starts at T, chooses only the displayed open-domain windows O, and then reads the boundary-value comparison row B before it can inspect sheaf or cohomology data.

The sheaf and cohomology obligations are packet-local. Rows S, Q, E record restriction, locality, gluing, cocycle, coboundary, and exactness data for the same displayed open windows, so the read has no route to quotient sheaf cohomology or selected representatives.

Finally A, R, C bind the complex-side window to the real readback, while D, M, P, N name the downstream distribution handoff. Since the tuple contains no host analytic-continuation, distribution-equality, global dual-space, or ambient-completeness coordinate, the local certificate cannot export such data. □

Lean4 ✓ `BEDC.Derived.HyperfunctionUp.HyperfunctionNamecertObligations`

Theorem 6482.3 (Hyperfunction sheaf-cohomology handoff)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or real-analysis consumer of an accepted* $\mathsf{Hyperfunction}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{Topology}^{\uparrow},\ O,\ B,\ \mathsf{Sheaf}^{\uparrow},\ \mathsf{Cohomology}^{\uparrow},\ \mathsf{Complex}^{\uparrow},\ R,\ D,\ E,\ K,\ L,\ M,\ C,\ P,\ N.$$

The consumer may read a boundary-value hyperfunction only after the open-domain, sheaf, cohomology, complex, real readback, exactness, transport, replay, provenance, compatibility, and local naming rows have been displayed. It receives no host analytic continuation, quotient cohomology class, selected boundary representative, distribution equality, global dual-space topology, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HyperfunctionUp.HyperfunctionCarrier sheaf cohomology handoff`

Proof. Start with the open-domain and boundary-value rows. The topology source T and the displayed windows O determine the real request surface before the boundary row B can be consumed.

Next read the sheaf and cohomology rows. The sheaf row S supplies restriction, locality, and gluing only on the displayed windows, while the cohomology row Q and exactness row E record the packet-local cocycle and coboundary ledger.

Then attach the complex/real readback. The complex analytic row A, real readback row R, and compatibility row C make the boundary-value route a single finite interface rather than a host analytic object.

The distribution handoff D is downstream of those rows, and K, L, M, P, N transport, replay, package, and name exactly that route. Since no further coordinate is present, a consumer cannot obtain quotient cohomology, selected representatives, distribution equality, global dual-space topology, or ambient completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Hyperfunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Hyperfunction}^{\uparrow}$ packet is generated from a topology source row, displayed open-domain row, boundary-value row, Sheaf row, Cohomology row, Complex analytic row, real readback row, Distribution handoff, exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, compatibility, consumer provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Hyperfunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6482.1, Definition 6482.2, and Definition 6482.3 seed the finite Sato boundary-value route through Sheaf, Cohomology, Complex, Topology, and Distribution rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Hyperfunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, quotient sheaf cohomology, selected boundary representatives, distribution equality, global dual-space topology, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary readability, sheaf-cohomology exactness, complex/real compatibility, distribution handoff, transport, replay, provenance, and local naming for the displayed Hyperfunction packet.

Chapter 6483

A Concrete Naming Certificate for $\mathsf{NestedShrinkingInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedShrinkingInterval}^{\uparrow}$ packet records the finite BEDC surface for a real-completion route driven by nested shrinking interval rows. It sits over the dyadic tolerance arithmetic of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the nested interval real route of chapter 1924, and the bisection real route of chapter 5711. The packet is interval-shrink data for the L10 completion boundary; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient equality of real names, a selected limit, trichotomy, or countable choice.

Definition 6483.1 (Nested shrinking interval carrier)**.** *A* $\mathsf{NestedShrinkingInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (J_0, J_1, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. J_0 *is the parent interval row, carrying the currently admitted closed dyadic interval and its endpoint witnesses.*
2. J_1 *is the child interval row, read only after* J_0*, carrying the next closed dyadic interval selected inside the parent row.*
3. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window listing the inspected interval requests, endpoint reads, and tail-budget positions.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, recording the displayed shrink bound and the dyadic budget attached to each inspected window.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the finite interval-window data into regular rational approximants.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the nested interval rows, window ledger, dyadic budget, and regular readback have been displayed.*
7. H *records componentwise* hsame *transport for the interval rows, stream window, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a completion-facing request through the same rows.*
9. P *is the* Pkg *provenance over* $\mathsf{NestedShrinkingInterval}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{BisectionReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{NestedShrinkingInterval}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host-selected limit, quotient equality of real names, trichotomy, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.NestedShrinkingIntervalUp.NestedShrinkingIntervalTasteGate single carrier alignment`

Theorem 6483.2 (Nested shrinking interval NameCert obligations)**.** *For every accepted* $\mathsf{NestedShrinkingInterval}^{\uparrow}$ *carrier* $I = (J_0, J_1, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NestedShrinkingInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the parent and child interval rows, finite StreamName window, dyadic diameter and tolerance ledger, regular rational readback, real seal, transport, replay, provenance, and local naming; nested containment requiring* J_1 *to be read as a child of* J_0*; diameter decay requiring* D *to certify the displayed shrink budget before* R *is consumed; finite-window readback requiring every completion-facing request to pass through* W *before the regular rational row; Real-seal nonescape requiring* E *to be downstream of* J_0, J_1, W, D, R*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*;* Pkg *provenance through* P*; and terminal naming through* N*.*

Proof. Carrier admission is projection from Definition 6483.1. The accepted packet displays exactly $J_0, J_1, W, D, R, E, H, C, P, N$.

The interval part begins with J_0 and J_1. The child row is available only as a nested row inside the parent row, so a consumer cannot read the shrink step without first reading the parent interval data.

The finite computational part is W, D, R. The StreamName window lists the requests, the dyadic ledger records the diameter and tolerance budget attached to those requests, and the regular rational row reads back only after that finite budget has been displayed.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for a selected limit, quotient real equality, trichotomy, countable choice, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6483.3 (Nested shrinking interval real-completion handoff)**.** *Every L10 real-completion consumer of an accepted* $\mathsf{NestedShrinkingInterval}^{\uparrow}$ *carrier reads the route through the nested interval rows* J_0, J_1*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before any completion-facing conclusion is exposed through* H, C, P, N*.*

Proof. Start with an accepted carrier $I = (J_0, J_1, W, D, R, E, H, C, P, N)$. The completion-facing seal is the row E, and that row is listed after the interval, window, dyadic, and regular rational rows.

A consumer must first read the parent and child interval rows. It then reads the finite StreamName window and the dyadic diameter ledger, so every inspected tolerance request is tied to a displayed shrink step.

Only after those rows are visible does the consumer obtain the regular rational readback R and then the real seal E. Transport and replay through H and C preserve the same order, while P and N record provenance and local naming. Thus the route reaches a completion-facing conclusion without importing ambient real completeness, quotient equality, a selected limit, trichotomy, or countable choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedShrinkingInterval}^{\uparrow}}$ **.**

Constructive story: A NestedShrinkingInterval$^{\uparrow}$ packet is generated from parent and child nested interval rows, a finite StreamName window, DyadicRatCore diameter and tolerance ledgers, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NestedShrinkingInterval}^{\uparrow}}$, seed)

Scope closed: Definition 6483.1, Definition 6483.2, and Definition 6483.3 seed the finite interval-shrink route for L10 real-completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{NestedShrinkingInterval}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NestedShrinkingIntervalUp.NestedShrinkingIntervalTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient equality of real names, selected limits, trichotomy, countable choice, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, nested containment, diameter decay, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedShrinkingIntervalUp.

Chapter 6484

A Concrete Naming Certificate for $\mathsf{CauchySequenceInterleaving}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySequenceInterleaving}^{\uparrow}$ packet records the finite interleaving surface used when two Cauchy-name routes must be inspected against one shared tolerance ledger. It sits between the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the fast-subsequence route of chapter 1518.

The packet is a source-route object for Real and Completion arguments. It interleaves two displayed Cauchy sequences by finite request windows, parity rows, dyadic error budgets, and readback rows; it consumes the regular-subsequence finite reindexing surface of chapter 381 when alternating or interleaving tail windows. It does not quotient streams, select a limit, compare host reals, invoke trichotomy, use countable choice, or assert ambient completeness.

Definition 6484.1 (CauchySequenceInterleaving carrier)**.** *A* $\mathsf{CauchySequenceInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, W, P, D, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *X and Y are the two source* $\mathsf{StreamName}^{\uparrow}$ *rows whose windows are interleaved;*
2. *W is the finite window ledger recording which requests of X and Y are inspected;*
3. *P is the parity and provenance row identifying the left and right legs of each interleaved request;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the inspected requests;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic tolerances;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been projected;*
7. *H records componentwise* hsame *transport for the two source rows, window ledger, parity row, tolerance row, readback row, seal row, and support rows;*
8. *C records displayed* Cont *replay from source windows to interleaved requests, dyadic tolerance readback, regular rational readback, and real-name sealing;*
9. *Q is the optional* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *handoff row used when a downstream diagonal argument asks for a cofinal fast window;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceInterleaving}^{\uparrow}}$ *row.*

The classifier compares two carriers only through these displayed rows. It stores no quotient stream equality, host equality between infinite sequences, selected limit, countable-choice selector, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 6484.2 (CauchySequenceInterleaving NameCert obligations)**.** *For an accepted* CauchySequenceInterleavi *carrier* $I = (X, Y, W, P, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceInterleaving}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed rows, source-window admission for* X *and* Y*, finite interleaving admission for* W*, parity and provenance admission for* P*, dyadic tolerance admission through* D*, regular rational readback through* R*, terminal Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, optional fast-subsequence handoff through* Q*, and local naming through* N*.*

Every Real-facing or Completion-facing consumer of the packet must therefore read the sibling StreamName, DyadicRatCore, RegSeqRat, Real, and FastCauchySubsequence rows in the displayed route. No obligation exports quotient stream equality, host real equality, selected limits, arbitrary choice of subsequences, trichotomy, or ambient completeness.

Proof. Project the carrier of Definition 6484.1. The only analytic rows available to a consumer are the two source StreamName rows, the finite interleaving window ledger, parity provenance, dyadic tolerance data, regular rational readback, Real seal, optional fast-subsequence handoff, and the structural support rows.

Read the source route first. The rows X, Y, W, P determine which finite windows are inspected and which leg of the interleaving each request belongs to, without choosing infinite representatives or quotienting streams.

Read the tolerance and seal route next. The rows D, R, E force every Real-facing or Completion-facing exit through dyadic tolerances, regular rational readback, and the terminal Real seal. The support rows H, C, Q, N transport, replay, optionally hand off, and name only that displayed finite route. □

Theorem 6484.3 (CauchySequenceInterleaving RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchySequenceInterleaving}^{\uparrow}$ *carrier* $I = (X, Y, W, P, D, R, E, H, C, Q, N)$ *hands its interleaved finite-window data to* $\mathsf{RegSeqRat}^{\uparrow}$ *through the route*

$$X, \quad Y, \quad W, \quad P, \quad D, \quad R, \quad E.$$

If a downstream diagonal or Cauchy-window argument asks for a cofinal fast window, the handoff may continue through the displayed $\mathsf{FastCauchySubsequence}^{\uparrow}$ *row* Q*, but it must still keep the StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and Real seal visible before any public comparison data is exposed.*

Proof. Begin with the finite interleaving ledger. The rows X, Y, W, P determine the accepted left and right windows and their parity provenance, so the packet has no coordinate for an untracked infinite merge of the two sources.

Apply the NameCert obligation surface of Definition 6484.2. It requires D before R and requires R before the terminal seal E, giving the displayed RegSeqRat handoff route.

For a fast cofinal request, use the Q row only after the regular readback route has been exposed. Since Q is an optional displayed handoff row rather than a choice functional, it cannot bypass X, Y, W, P, D, R, E or create a selected limit. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceInterleaving}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequenceInterleaving}^{\uparrow}$ packet is generated from two displayed StreamName source rows, finite interleaving windows, parity provenance, dyadic tolerances, regular rational readback, a terminal Real seal, optional fast-subsequence handoff, componentwise transport, replay, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchySequenceInterleaving}\uparrow}$, seed)

Scope closed: Definition 6484.1, Definition 6484.2, and Definition 6484.3 define the finite source-route packet and its RegSeqRat handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchySequenceInterleaving}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, bridge checking, quotient stream equality, host real equality, selected limits, countable choice, trichotomy, or ambient completeness.

Upgrade path: The next obligation is a Lean theorem for the carrier and handoff clauses, followed by a standard bridge only after the source route has a checked public surface.

Chapter 6485

A Concrete Naming Certificate for $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$

Following Bourbaki on uniform spaces, Kelley on filters, and Bishop and Bridges on constructive completion, the $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$ packet records the finite BEDC route in which uniform Cauchy evidence and located filter evidence are read by the same completion-neighbour surface. It joins the uniform Cauchy filter rows of chapter 1694 with the located filter rows of chapter 1661, while keeping the stream, dyadic, regular-rational, and real-seal faces of chapter 126, chapter 291, chapter 301, and chapter 38 explicit. The carrier is not a quotient of filters and does not select a limit; it only names the rows a completion-neighbour consumer may inspect.

Definition 6485.1 (Located uniform Cauchy filter carrier)**.** *A* $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (U, F, L, S, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *row, carrying the displayed uniform-space, filter-base, entourage-Cauchy, stream-window, regular-rational, and completion-handoff surface of chapter 1694.*
2. *F is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, carrying the displayed located-tail, basis-refinement, sequence-handoff, and real-seal surface of chapter 1661.*
3. *L is the completion-neighbour location row, recording that the uniform entourage request and the located-tail request name the same finite neighbourhood read before any seal-facing consumer proceeds.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row supplied to the joined filter readback, using only the observation surface of chapter 126.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows and to the same located-uniform request, as in chapter 301.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic entourage scale, located-ball slack, comparison cells, and radius conversions used by the joined request; it is read through chapter 291.*
7. *T is the located-uniform tail row, requiring the uniform Cauchy row U and located filter row F to meet the same displayed dyadic tolerance before R or E can be exported.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after U, F, L, S, R, D, T have been displayed through the real-name boundary of chapter 38.*
9. *H records componentwise* hsame *transport for the uniform filter, located filter, neighbourhood location, stream, regular-rational, dyadic, tail, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a completion-neighbour request through $U, F, L, S, R, D, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyFilter}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $U, F, L, S, R, D, T, E, H, C, P, N$. It has no coordinate for quotient filter equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, open-cover compactness, ultrafilters, or a second dyadic source.*

Theorem 6485.2 (Located uniform Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$ *carrier*

$$X = (U, F, L, S, R, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, uniform Cauchy filter exposure, located Cauchy filter exposure, completion-neighbour location, finite StreamName window readback, RegSeqRat regular-rational readback, DyadicRatCore tolerance accounting, located-uniform tail compatibility, terminal RealUp seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 6485.1.*

Proof. Carrier admission is projection from Definition 6485.1. The analytic rows visible to the certificate are exactly U, F, L, S, R, D, T, E.

The uniform row U is read through chapter 1694, and the located row F is read through chapter 1661. The neighbourhood location row L identifies the same finite completion-neighbour request for both rows before downstream readback begins.

The sequence-facing part of the route is finite. The window row S uses chapter 126, the tolerance row D uses chapter 291, and the regular rational row R uses chapter 301. The tail row T binds these rows to the displayed located-uniform request.

Finally the seal-facing row E is read through chapter 38 only after U, F, L, S, R, D, T have been displayed. The support rows H, C, P, N transport, replay, source, and name that same route, so no quotient filter, selected limit, countable-choice selector, host equality, ambient completeness, open-cover compactness, ultrafilter, or second dyadic source can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedUniformCauchyFilter}^{\uparrow}$ packet is generated from a UniformCauchyFilter row, a LocatedCauchyFilter row, a completion-neighbour location row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a located-uniform tail row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 6485.1 and Definition 6485.2 seed the joined located-uniform filter route over UniformCauchyFilterUp, LocatedCauchyFilterUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host real equality, ambient Real completeness, open-cover compactness, ultrafilters, Lean verification, or bridge checking.

Upgrade path: Formal closure requires carrier admission, completion-neighbour location, located-uniform tail compatibility, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyFilterUp.

Chapter 6486

A Concrete Naming Certificate for $\mathsf{KochSnowflake}^{\uparrow}$

Following Koch and Falconer, the $\mathsf{KochSnowflake}^{\uparrow}$ packet records the classical three-sided Koch snowflake boundary as a finite NameCert over three sidewise $\mathsf{KochCurve}^{\uparrow}$ arcs. It sits beside the single-arc construction of chapter 3995, the uniform-limit handoff of chapter 463, the L10 Real route of Definition 1624.4, and the terminal real-name seal of chapter 38. The packet names only the cyclic boundary loop, stagewise displacement rows, finite StreamName windows, regular rational readbacks, Real coordinate seals, transport, replay, provenance, and local naming; it does not import a completed planar subset, area computation, Hausdorff dimension, quotient equality of real coordinate pairs, selected infinite branches, or ambient compactness.

Definition 6486.1 (Koch snowflake carrier)**.** *A* $\mathsf{KochSnowflake}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A_0, A_1, A_2, S, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. A_0, A_1, A_2 *are the three sidewise* $\mathsf{KochCurve}^{\uparrow}$ *arc rows. Each row records the finite address, polygonal-stage segment word, displacement ledger, coordinate windows, regular-rational readback, Real coordinate seal, and uniform-limit handoff inherited from the corresponding side packet of chapter 3995.*

2. S *is the cyclic endpoint-compatibility row. It records that the terminal endpoint of* A_i *is the initial endpoint of* A_{i+1}*, with indices read cyclically, using only displayed finite endpoint rows and their* $\mathsf{Real}^{\uparrow}$ *seals.*

3. D *is the dyadic stage and displacement ledger. It aligns the three arc-stage requests, the triangular orientation data, and the dyadic tolerance comparisons used to read the boundary loop.*

4. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the six endpoint-coordinate schedules and the inspected stagewise segment rows.*

5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those coordinate schedules, with every tolerance comparison routed through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*

6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *coordinate seal family for the three cyclic endpoints and the displayed segment endpoints reached through* D, W, R*.*

7. *H records componentwise* hsame *transport for the three sidewise arcs, cyclic compatibility row, dyadic stage ledger, window family, regular-rational readback, Real seal family, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a boundary-loop request through $A_0, A_1, A_2, S, D, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{KochSnowflake}^{\uparrow}$, $\mathsf{KochCurve}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}$ *row.*

The classifier accepts a packet only through the displayed tuple $A_0, A_1, A_2, S, D, W, R, E, H, C, P, N$. It has no coordinate for a host planar subset, area measure, dimension computation, quotient real equality, selected infinite branch, or ambient compactness theorem.

Theorem 6486.2 (Koch snowflake NameCert obligations)**.** *For every accepted* $\mathsf{KochSnowflake}^{\uparrow}$ *carrier*

$$K = (A_0, A_1, A_2, S, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; sidewise* $\mathsf{KochCurve}^{\uparrow}$ *arc exposure through A_0, A_1, A_2; cyclic endpoint compatibility through S; dyadic stage and displacement exactness through D; finite window scheduling through W; regular rational readback through R; terminal* $\mathsf{Real}^{\uparrow}$ *coordinate sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6486.1. The only arc-facing rows are A_0, A_1, A_2, and each is already a sidewise $\mathsf{KochCurve}^{\uparrow}$ packet with its finite address, polygonal-stage, dyadic displacement, coordinate-window, regular-readback, Real-seal, and uniform-limit data.

The compatibility obligation is exactly the row S. It compares only the displayed endpoints of adjacent side packets, with the terminal endpoint of one side matched to the initial endpoint of the next side in cyclic order.

The Real-facing obligations are exhausted by D, W, R, E. The dyadic stage ledger aligns the side stages and tolerances, the StreamName windows schedule the finite endpoint reads, the RegSeqRat row performs the rational readback, and the terminal coordinate seals remain downstream of the same L10 route.

Finally H, C, P, N transport, replay, source, and name this finite loop. Since the carrier contains no host planar subset, area row, dimension row, quotient real equality row, selected branch row, or ambient compactness row, no such object can be a local $\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}$ obligation. □

Theorem 6486.3 (Koch snowflake dyadic curve route)**.** *Every accepted boundary-loop, segment, or coordinate read of a* $\mathsf{KochSnowflake}^{\uparrow}$ *packet factors through the finite route*

$$A_0, A_1, A_2 \longmapsto S \longmapsto D, W, R, E \longmapsto H, C, P, N.$$

The route exports only the three sidewise $\mathsf{KochCurve}^{\uparrow}$ *arcs, cyclic endpoint compatibility, dyadic stage and displacement data, finite StreamName windows, RegSeqRat readbacks, terminal* $\mathsf{Real}^{\uparrow}$ *coordinate seals, and the structural rows. It exports no completed planar fractal, area computation, Hausdorff dimension, quotient equality of real coordinate pairs, selected infinite branch, or ambient compactness principle.*

Proof. Begin with an accepted consumer of a boundary-loop, segment, or coordinate row. By Definition 6486.1, the consumer can enter the packet only through the three sidewise arc rows and the cyclic compatibility row.

Apply Definition 6486.2. It forces every side read through A_0, A_1, A_2, every loop-closing read through S, and every Real-facing read through D, W, R, E before the structural rows are inspected.

The single-arc route of chapter 3995, the uniform-limit route of chapter 463, and the L10 Real gate of Definition 1624.4 supply only the displayed finite readbacks named in the carrier. None adds a second coordinate for a completed planar subset, area or dimension data, quotient equality, branch selection, or compactness. Hence every accepted consumer remains on the displayed dyadic curve route. □

Falsifiable prediction: An accepted $\mathsf{KochSnowflake}^{\uparrow}$ consumer that reaches a boundary-loop or $\mathsf{Real}^{\uparrow}$ coordinate seal without first displaying the three sidewise KochCurve arc rows, cyclic endpoint-compatibility row, dyadic stage ledger, StreamName windows, RegSeqRat readback, and Real seal family refutes the carrier surface.

Independence witness: kochcurve, uniformlimit, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole $\mathsf{KochSnowflake}^{\uparrow}$ packet, because the snowflake certificate binds three sidewise KochCurve arcs with cyclic endpoint compatibility, a shared dyadic stage ledger, Real-seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}$.

Constructive story: A $\mathsf{KochSnowflake}^{\uparrow}$ packet is generated from three sidewise KochCurve arc rows, cyclic endpoint compatibility, a dyadic stage and displacement ledger, finite StreamName windows, RegSeqRat readbacks, RealUp coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6486.1, Definition 6486.2, and Definition 6486.3 seed the finite three-sided Koch snowflake boundary-loop certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KochSnowflake}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host planar subsets, area measure, Hausdorff dimension, quotient equality of real coordinate pairs, selected infinite branches, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, three sidewise KochCurve arc exposure, cyclic endpoint compatibility, dyadic stage exactness, finite window scheduling, RegSeqRat readback, Real coordinate sealing, transport, replay, provenance, and local naming for BEDC.Derived.KochSnowflakeUp.

Chapter 6487

A Concrete Naming Certificate for $\mathsf{HilbertInequality}^{\uparrow}$

Following Hardy–Littlewood–Polya style Hilbert inequality, the $\mathsf{HilbertInequality}^{\uparrow}$ packet records a finite real-series inequality route rather than an ambient double-series theorem. It sits next to the absolute-domination surface of chapter 1700, the finite square-bound route of chapter 1684, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes the Hilbert kernel bound only as displayed BEDC rows over finitely inspected series windows, not as quotient stream equality, selected sums, countable choice, host real order, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6487.1 (Hilbert inequality carrier)**.** *A* $\mathsf{HilbertInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, B, K, S, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *A and B are the two real-series source rows whose inspected terms are read through finite* $\mathsf{Real}^{\uparrow}$ *seals.*
2. *K is the Hilbert-kernel ledger. Each row records a displayed denominator window, the dyadic comparison budget for the kernel coefficient, and the finite multiplication read attached to the selected term pair.*
3. *S is the finite summation-window row. It records the rectangular or triangular window actually inspected by the packet and no infinite double-sum selector.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the term observations, kernel coefficients, partial rectangular sums, and comparison budgets opened by A, B, K, S.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the exported inequality boundary, reached only after A, B, K, S, Q have been displayed.*
6. *T stores componentwise* hsame *transport for the two source series, Hilbert-kernel ledger, finite windows, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a finite window request through the kernel ledger, regular readback, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{HilbertInequality}^{\uparrow}$*,* $\mathsf{AbsoluteConvergence}^{\uparrow}$*,* $\mathsf{CauchySchwarzReal}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HilbertInequality}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, B, K, S, Q, E, T, C, P, N$. It has no coordinate for an ambient Hilbert-space theorem, selected infinite sums, quotient stream equality, countable choice, host real order, or ambient completeness.

Theorem 6487.2 (Hilbert inequality NameCert obligations)**.** *For every accepted* $\mathsf{HilbertInequality}^{\uparrow}$ *carrier $H = (A, B, K, S, Q, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HilbertInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the two source series, the Hilbert-kernel ledger, the finite summation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The same surface records kernel-window exactness, rectangular finite-sum exposure, absolute-domination compatibility with chapter 1700, square-bound compatibility with chapter 1684, and nonescape from selected sums, quotient stream equality, countable choice, host real order, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6487.1. The available rows are exactly the two source series A, B, the kernel ledger K, the finite summation window S, the regular readback Q, the Real seal E, and the structural rows T, C, P, N.

The kernel and window obligations are the same finite route. The packet must display the denominator window and dyadic comparison budget in K for each inspected pair, and S names the finite rectangular or triangular window in which those pair reads occur.

The compatibility obligations use the sibling surfaces only as finite routes. The absolute-convergence chapter supplies the domination shape for series tails, while the Cauchy–Schwarz real chapter supplies the square-bound route for real-valued finite rows. Both are consumed through the same regular readback and Real seal rows Q, E.

Finally, the structural rows transport, replay, source, and name that displayed route. Since none of the excluded analytic objects is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6487.3 (Hilbert inequality Real-series handoff)**.** *Every real-series consumer of an accepted* $\mathsf{HilbertInequality}^{\uparrow}$ *packet receives the Hilbert inequality boundary only through*

$$A, B \longmapsto K, S \longmapsto Q \longmapsto E \longmapsto T, C, P, N.$$

The handoff exposes the two source series, finite kernel-denominator comparisons, inspected summation windows, regular rational readback, and the terminal Real seal. It does not expose a host double-series theorem, quotient stream equality, selected sums, countable choice, host real order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a real-series consumer request. The carrier first exposes the source rows A and B, so every term read is tied to a displayed finite series observation.

The Hilbert kernel is then read through K inside the finite window S. This is the only place where denominator comparisons and pairwise multiplication budgets appear, so the consumer cannot bypass the packet to obtain an independent double series.

The readback and seal are forced by Q and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row records the rational observations for the selected terms, kernel coefficients, and partial finite sums before the $\mathsf{Real}^{\uparrow}$ seal is available.

The rows T, C, P, N preserve and replay the same route. Because the carrier has no coordinate for the excluded host or quotient data, every exported Hilbert inequality read is the displayed finite real-series handoff. □

Falsifiable prediction: Every accepted $\mathsf{HilbertInequality}^{\uparrow}$ read exposes two real-series source rows, a Hilbert-kernel denominator ledger, finite summation windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert before a real-series consumer receives the inequality boundary.

Independence witness: absolute-convergence, cauchy-schwarz-real, regseqrat, real: $\mathsf{HilbertInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source series, a Hilbert-kernel ledger, finite summation windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming for the same inequality route.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HilbertInequality}^{\uparrow}$ packet is generated from two real-series source rows, a Hilbert-kernel denominator ledger, finite summation windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6487.1, Definition 6487.2, and Definition 6487.3 seed the finite Hilbert inequality boundary for displayed real-series rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host double-series convergence, quotient stream equality, selected sums, countable choice, host real order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, kernel-window exactness, finite summation exposure, absolute-domination compatibility, square-bound compatibility, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HilbertInequalityUp.

Chapter 6488

A Concrete Naming Certificate for $\mathsf{FinitelyBranchingTree}^{\uparrow}$

Following Koenig's lemma, Brouwer spreads, and Bishop-style finite evidence, the $\mathsf{FinitelyBranchingTree}^{\uparrow}$ packet records a traditional finite-prefix tree as a BEDC NameCert carrier. It sits between the path-sensitive boundary of chapter 2092, the binary-window surface of chapter 1475, and the bar-modulus route of chapter 400. The packet exposes finite branching and prefix admission data for L10 countable-name and compactness consumers without selecting an infinite branch, asserting fan compactness, or turning a prefix ledger into host tree equality.

Definition 6488.1 (FinitelyBranchingTree carrier)**.** *A* $\mathsf{FinitelyBranchingTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (P, B, A, W, K, F, S, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite prefix-row family. Each row stores a finite node code, its length, and the immediate predecessor readback used by prefix consumers.*
2. *B is the branching ledger. For each admitted prefix row it displays a finite child bundle and the rows witnessing which child extensions remain in the tree.*
3. *A is the admissibility row, saying that the root prefix is present and that every admitted non-root prefix has its predecessor admitted.*
4. *W is the finite window row read by* $\mathsf{StreamName}^{\uparrow}$ *and by Cantor-style binary or finite-alphabet probes before a path consumer can inspect longer prefixes.*
5. *K is the* $\mathsf{KonigLemma}^{\uparrow}$ *handoff row. It records the finite data needed by a Koenig-facing theorem without adding a selected infinite branch.*
6. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *handoff row. It records finite bar and modulus requests only through displayed prefix bundles.*
7. *S is the* $\mathsf{CantorSpace}^{\uparrow}$ *and* $\mathsf{SpreadSpace}^{\uparrow}$ *handoff row, used when binary cylinders or spread laws read the same prefix ledger.*
8. *H records componentwise* hsame *transport for prefix rows, branching rows, admissibility, window reads, handoff rows, replay, provenance, and local naming.*

9. *C records the displayed* Cont *replay from prefix admission to branching exposure, finite-window reads, CantorSpace and FanTheorem handoffs, KoenigLemma consumers, and StreamName-facing prefix observations.*

10. *Q is the* Pkg *provenance over* $\mathsf{FinitelyBranchingTree}^{\uparrow}$, $\mathsf{KonigLemma}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{SpreadSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitelyBranchingTree}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen infinite branch, detached open-cover compactness, host tree equality, countable choice, bar induction, or a completed path space.

Theorem 6488.2 (FinitelyBranchingTree NameCert obligations)**.** *For every accepted* $\mathsf{FinitelyBranchingTree}^{\uparrow}$ *carrier $T = (P, B, A, W, K, F, S, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FinitelyBranchingTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed prefix rows, finite branching ledger, predecessor-closure row, finite window row, KoenigLemma handoff, FanTheorem handoff, CantorSpace and SpreadSpace handoff, componentwise transport, displayed replay, provenance, and local naming. Every consumer must read finite prefix data through*

$$P \longmapsto B \longmapsto A \longmapsto W \longmapsto K, F, S \longmapsto H, C, Q, N.$$

Thus the packet supplies finite branching evidence and its standard handoffs, not an infinite branch selector, a fan compactness theorem, an open-cover compactness principle, or an ambient path-completion object.

Proof. Project the carrier rows from Definition 6488.1. The prefix family, branching ledger, predecessor-closure row, finite-window row, three handoff rows, transport, replay, provenance, and local name are exactly the displayed coordinates of the packet.

The handoff rows are read after prefix admission. A KoenigLemma consumer sees the finite branching ledger and predecessor closure before it can ask for a branch-facing conclusion; a FanTheorem consumer sees only finite bar and modulus requests over the same prefix rows; CantorSpace and SpreadSpace consumers read finite cylinders and spread-law windows through S.

Componentwise stability is forced by H. If an accepted replacement of T is compared with another packet, the local classifier first transports $P, B, A, W, K, F, S, C, Q, N$ by hsame and then replays the same route through C.

Finally inspect the absent coordinates. Since the tuple has no row for a chosen infinite branch, detached open-cover compactness, host tree equality, countable choice, bar induction, or completed path space, the local NameCert cannot export any of those claims. □

Theorem 6488.3 (FinitelyBranchingTree finite-window handoffs)**.** *Let $T = (P, B, A, W, K, F, S, H, C, Q, N)$ be an accepted* $\mathsf{FinitelyBranchingTree}^{\uparrow}$ *packet. Any downstream* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{SpreadSpace}^{\uparrow}$, *or* $\mathsf{KonigLemma}^{\uparrow}$ *read of T factors through the same finite prefix rows P, the same branching ledger B, and the same predecessor-closure row A before a handoff row is visible.*

Consequently the downstream read is stable under hsame *transport and* Cont *replay of H and C, and its provenance remains the finite* Pkg *row Q with local naming by N.*

Proof. Begin with a downstream read. The carrier has no direct consumer coordinate, so the read must open the prefix rows P, the branching ledger B, and the predecessor-closure row A before any finite window or theorem-facing handoff can be inspected.

The window row W supplies the StreamName-facing finite prefix observation. The row S repacks the same observations for CantorSpace and SpreadSpace cylinders, while F and K expose only the finite bar-modulus and KoenigLemma boundary rows tied to those prefixes.

Transport and replay do not add further data. The row H moves all displayed rows componentwise by hsame, C replays the same finite route, and Q, N attach provenance and local naming. The packet therefore gives handoff stability without selecting an infinite branch or asserting compactness beyond the displayed finite ledger. □

Falsifiable prediction: An accepted $\mathsf{FinitelyBranchingTree}^{\uparrow}$ consumer that reads a KoenigLemma, CantorSpace, FanTheorem, SpreadSpace, or StreamName handoff without first displaying the prefix rows, finite branching ledger, predecessor closure, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows refutes the carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitelyBranchingTree}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitelyBranchingTree}^{\uparrow}$ packet is generated from finite prefix rows, a finite branching ledger, predecessor closure, finite StreamName windows, KoenigLemma, FanTheorem, CantorSpace, and SpreadSpace handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FinitelyBranchingTree}^{\uparrow}}$, seed)

Scope closed: Definition 6488.1, Definition 6488.2, and Definition 6488.3 seed the finite-prefix tree packet and its standard handoffs without selecting infinite branches.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FinitelyBranchingTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Koenig's lemma, the Fan Theorem, Cantor compactness, spread completeness, selected infinite branches, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite branching coverage, predecessor closure, handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.FinitelyBranchingTreeUp.

Chapter 6489

A Concrete Naming Certificate for $\mathsf{CompactTailSelector}^{\uparrow}$

Following the compact-uniform-continuity construction of chapter 9237, the $\mathsf{CompactTailSelector}^{\uparrow}$ packet records the finite tail-window selector that a compact metric argument can hand to completion and real-valued consumers. It sits after the compact metric source of chapter 129, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the tolerance-only modulus surface of chapter 369, and the real-name boundary of chapter 38. The packet selects a displayed finite tail window from already named modulus and compact-net rows; it is not countable choice, open-cover compactness, sequential compactness, or an ambient completeness theorem.

Definition 6489.1 (CompactTailSelector carrier)**.** *A* $\mathsf{CompactTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, S, R, U, E, W, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the inspected finite compact-net coverage and metric-distance witnesses.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose finite stream windows are being bounded by the selector.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected stream windows and tolerance observations.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, already tolerance-only before the selector reads it.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, S, R, U and the selected window row have all been displayed.*
6. *W is the selected finite tail-window row. It stores the threshold index, the finite interval of inspected names, and the compact-net witness used to justify that this window covers the requested tolerance.*
7. *D is the dyadic tolerance ledger comparing the requested rational tolerance with the modulus and window rows.*
8. *H records componentwise* hsame *transport for compact metric data, stream windows, regular readbacks, uniform modulus rows, real seals, selected windows, tolerance comparisons, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from compact-net reads and uniform-modulus reads to the selected tail window, and then to regular readback and the real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactTailSelector}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for countable choice, an arbitrary open cover, sequential compactness, a selected convergent subsequence, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6489.2 (CompactTailSelector NameCert obligations)**.** *For every accepted* $\mathsf{CompactTailSelector}^{\uparrow}$ *carrier $T = (K, S, R, U, E, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactTailSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact metric row, stream-window row, regular rational readback, uniform modulus, real seal, selected tail window, dyadic tolerance ledger, componentwise transport, displayed replay, provenance, and local naming. Every Real-facing or completion-facing consumer must read the finite route*

$$K, S, R, U, D \longmapsto W \longmapsto E, H, C, P, N$$

before treating the selected window as a tail bound.

Proof. Project the rows of Definition 6489.1. The packet exposes compact metric data, stream windows, regular readbacks, a uniform modulus, a real seal, the selected tail-window row, a dyadic comparison row, and the structural rows.

The selected window W is downstream of the compact metric source and the uniform-modulus row. A consumer therefore cannot inspect a tail threshold without first displaying the compact-net coverage, the stream windows being bounded, the regular rational readback, and the tolerance comparison that ties the modulus to the requested precision.

The rows H, C, P, N preserve this route under classifier transport, continuation replay, provenance, and local naming. Since no coordinate stores countable choice, arbitrary open covers, sequential compactness, selected subsequences, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local surface. □

Theorem 6489.3 (CompactTailSelector finite ledger)**.** *Let $T = (K, S, R, U, E, W, D, H, C, P, N)$ be an accepted* $\mathsf{CompactTailSelector}^{\uparrow}$ *packet. The selected tail window is finite ledger data: it is read from the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observations, the* $\mathsf{UniformModulus}^{\uparrow}$ *row, and the dyadic tolerance comparison. It is not obtained from countable choice, an open-cover subcover theorem, a chosen subsequence, or completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Read K, S, R, U, D in order. The compact metric row supplies only finite net coverage, the stream row supplies finite windows, the regular rational row supplies the observed tolerances, and the uniform-modulus row supplies a tolerance-only bound checked by D.

The row W stores the resulting threshold and inspected finite interval. Because W is part of the same BHist packet, transport by H and replay by C move it componentwise with the compact source, stream windows, readback rows, and modulus row.

The packet never opens an arbitrary cover or a sequence of choices. Its real-facing row E is reached only after the finite ledger has been displayed, so no consumer can reinterpret the tail selector as countable choice, open-cover compactness, selected subsequence compactness, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{CompactTailSelector}^{\uparrow}$ consumer that reads a tail threshold without the displayed CompactMetric source row, StreamName finite-window row, RegSeqRat readback row, UniformModulus row, dyadic tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows refutes the carrier surface.

Independence witness: compactmetric, streamname, regseqrat, uniformmodulus, real: no listed sibling supplies the full packet $T = (K, S, R, U, E, W, D, H, C, P, N)$, because each sibling contributes a compact source, finite windows, rational readback, a modulus, or a terminal seal while this chapter binds them into one displayed tail-window selector.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactTailSelector}^{\uparrow}$ packet is generated from CompactMetric source rows, StreamName finite windows, RegSeqRat readback, a UniformModulus row, a RealUp seal, selected tail-window rows, dyadic tolerance comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactTailSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6489.1, Definition 6489.2, and Definition 6489.3 seed the finite tail-window selector consumed by compact-uniform and completion-facing arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactTailSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, sequential compactness, countable choice, selected subsequences, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source coverage, finite-window readback, modulus-to-tail exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactTailSelectorU

Chapter 6490

A Concrete Naming Certificate for CompletionUnitDenseImage$^{\uparrow}$

Following Bishop and Bridges, Kelley, and Bourbaki on constructive completion and dense image maps, the CompletionUnitDenseImage$^{\uparrow}$ packet records the surface by which the completion unit exposes its image as finite dense-source data. It sits after the unit certificate of chapter 1527, the general dense embedding interface of chapter 3617, the metric completion dense embedding of chapter 3351, the completion universal-property surface of chapter 397, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the dense image of the unit without quotient representatives, selected completion points, countable choice, host equality, or ambient Real$^{\uparrow}$ completeness.

Definition 6490.1 (CompletionUnitDenseImage carrier)**.** *A* CompletionUnitDenseImage$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, C, u, D, W, R, E, H, K, P, N)$$

with the following visible rows:

1. *X is the displayed source metric or regular-Cauchy row entering the completion unit.*
2. *C is the accepted* CauchyCompletionUnit$^{\uparrow}$ *row that admits X into the completion-facing surface.*
3. *u is the unit-image row, recording the displayed image of X inside the completion boundary.*
4. *D is the dense-image witness row read through* DenseEmbedding$^{\uparrow}$ *and* MetricCompletionDenseEmbedding$^{\uparrow}$*.*
5. *W is the finite* StreamName$^{\uparrow}$ *window row used by the dense-image request.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the same window and tolerance ledger.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after C, u, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, unit, dense-image, window, readback, real seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from source admission through the dense image and completion universal-property route.*
10. *P is the* Pkg *provenance over* CompletionUnitDenseImage$^{\uparrow}$*,* CauchyCompletionUnit$^{\uparrow}$*,* DenseEmbedding$^{\uparrow}$*,* MetricCompletionDenseEmbedding$^{\uparrow}$*,* CompletionUniversalProperty$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{CompletionUnitDenseImage}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, host equality of completed objects, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6490.2 (CompletionUnitDenseImage NameCert obligations)**.** *For an accepted* CompletionUnitDenseImag *carrier $U = (X, C, u, D, W, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionUnitDenseImage}^{\uparrow}}$ *surface consists exactly of source admission through X, unit exposure through C, u, dense-image exposure through D, finite window exposure through W, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Every completion-facing consumer must read the unit row C, the unit-image row u, and the dense-image row D before it can consume W, R, E, H, K, P, N.

Proof. Carrier admission is projection from Definition 6490.1. The tuple lists exactly the source, completion unit, unit image, dense-image witness, window, readback, real seal, transport, replay, provenance, and local naming rows.

The unit exposure is supplied by the $\mathsf{CauchyCompletionUnit}^{\uparrow}$ row. It sends the displayed source into the completion-facing boundary as finite unit data before any density request is inspected.

The dense-image exposure then reads D through $\mathsf{DenseEmbedding}^{\uparrow}$ and $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$. The finite window W and readback row R are the only numerical route offered to the real seal E and to universal-property consumers.

Finally H, K, P, N preserve and name the same route. Since the carrier has no quotient, selected representative, choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 6490.3 (CompletionUnitDenseImage universal route)**.** *Every completion universal-property consumer of an accepted* $\mathsf{CompletionUnitDenseImage}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyCompletionUnit}^{\uparrow} \longrightarrow \mathsf{DenseEmbedding}^{\uparrow} \longrightarrow \mathsf{MetricCompletionDenseEmbedding}^{\uparrow} \longrightarrow \mathsf{CompletionUniversalPro}$$

with the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the* $\mathsf{Real}^{\uparrow}$ *seal row E already displayed. Thus the universal-property handoff receives a dense image of the unit as finite BEDC data, not as a quotient completion class, selected completion representative, countable-choice selector, host equality proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the accepted carrier $U = (X, C, u, D, W, R, E, H, K, P, N)$. The completion unit row C and unit-image row u are the only source-to-completion entry data available to the consumer.

Next apply the dense embedding siblings. The rows D, W, R expose the dense-image witness, finite source window, and regular rational readback before any completion universal-property row can be inspected.

The real seal E is terminal for the numerical route, and K replays the same source-to-dense-image path for the universal-property consumer. Transport and provenance through H, P, N do not add another completion object.

Therefore every universal-property handoff factors through the displayed unit, dense-image, metric-completion dense embedding, readback, and real-seal rows, with none of the excluded quotient, choice, selected-representative, host-equality, or ambient-completeness data exported. □

Falsifiable prediction: Every accepted $\mathsf{CompletionUnitDenseImage}^{\uparrow}$ consumer reads a Cauchy-CompletionUnit row, a unit-image row, a dense-image witness, a finite StreamName window,

RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before any completion universal-property handoff is exported.

Independence witness: cauchycompletionunit, dense-embedding, metric-completion-dense-embedding, completionuniversalproperty, regseqrat, real: none is bijective with $\mathsf{CompletionUnitDenseImage}^{\uparrow}$ because this packet alone binds the unit image to dense-source readback before the universal-property consumer receives the completion-facing route.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUnitDenseImage}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionUnitDenseImage}^{\uparrow}$ packet is generated from a displayed source row, a CauchyCompletionUnit row, a unit-image row, dense-image witness data, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionUnitDenseImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6490.1, Definition 6490.2, and Definition 6490.3 seed the dense-image surface of the completion unit over CauchyCompletionUnit, DenseEmbedding, MetricCompletionDenseEmbedding, CompletionUniversalProperty, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionUnitDenseImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected completion representatives, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-image exposure, dense-image locality, readback readiness, universal-route nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionUnitDenseImageUp.

Chapter 6491

A Concrete Naming Certificate for $\mathsf{BishopRealExtensionality}^{\uparrow}$

Following Bishop and Bridges, equality of constructive reals is read through interchangeable regular presentations rather than through host real equality. The $\mathsf{BishopRealExtensionality}^{\uparrow}$ packet records the finite NameCert surface by which an accepted $\mathsf{Real}^{\uparrow}$ name may be substituted for an extensionally equal accepted name. It is downstream of the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the comparison boundary of chapter 1820, the tolerance-row discipline of chapter 2074, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with an extensional substitution route, not quotient equality, host real equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6491.1 (Bishop real extensionality carrier)**.** *A* $\mathsf{BishopRealExtensionality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S_0, S_1, R_0, R_1, Q, T, A, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the source and target real names.*
2. R_0 *and* R_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to* S_0 *and* S_1*.*
3. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row relating the two regular readbacks through displayed Cauchy agreement.*
4. T *is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *tolerance ledger used by the comparison and by any substitution request.*
5. A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S_0, S_1, R_0, R_1, Q, T *have been displayed.*
6. H *records componentwise* hsame *transport for the two stream windows, two regular readbacks, comparison row, tolerance ledger, terminal seal, replay, provenance, and local naming row.*
7. C *records displayed* Cont *replay from an extensional substitution request through finite windows, regular readback, comparison, tolerance, and seal rows.*
8. P *is the* Pkg *provenance over* $\mathsf{BishopRealExtensionality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopRealExtensionality}^{\uparrow}}$ *row.*

The classifier compares accepted extensionality packets only through $S_0, S_1, R_0, R_1, Q, T, A, H, C, P, N$. It exposes no quotient equality of streams, host real equality, selected representative, countable-choice schedule, detached limit, or ambient completeness row.

Theorem 6491.2 (Bishop real extensionality NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealExtensionality}^{\uparrow}$ *carrier*

$$E = (S_0, S_1, R_0, R_1, Q, T, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealExtensionality}^{\uparrow}}$ *surface consists exactly of the two* $\mathsf{StreamName}^{\uparrow}$ *finite windows, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row, the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every substitution-facing consumer must expose $S_0, S_1, R_0, R_1, Q, T, A$ before it consumes H, C, P, N.*

Proof. Project the carrier of Definition 6491.1. The analytic rows are exactly $S_0, S_1, R_0, R_1, Q, T, A$, and the remaining rows are structural transport, replay, provenance, and naming.

The two $\mathsf{StreamName}^{\uparrow}$ rows are finite windows. The two $\mathsf{RegSeqRat}^{\uparrow}$ rows are the only readbacks attached to those windows, so a consumer cannot enter through an infinite stream or a selected representative.

The comparison row Q is displayed before the terminal seal, and it is governed by the tolerance ledger T. Thus equality-facing information is read as Cauchy-equivalence evidence over inspected tolerances, not as host real equality or quotient equality.

Finally the seal row A is downstream of the displayed comparison route. Since no carrier coordinate remains outside $S_0, S_1, R_0, R_1, Q, T, A, H, C, P, N$, the listed rows exhaust the local NameCert obligations. □

Theorem 6491.3 (Bishop real extensionality substitution route)**.** *Every Real or Completion consumer that substitutes one accepted* $\mathsf{Real}^{\uparrow}$ *name for another through a* $\mathsf{BishopRealExtensionality}^{\uparrow}$ *packet must factor the substitution through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may replay the substitution along componentwise hsame *and displayed* Cont *rows, with* Pkg *provenance and local* NameCert *data, but it receives no quotient stream equality, host real equality, selected representative, countable choice, detached limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a substitution-facing read of E. The only source observations are the finite windows S_0, S_1, so the route starts in the $\mathsf{StreamName}^{\uparrow}$ surface.

Read those windows through R_0, R_1. These are the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks attached to the displayed observations, and no other readback coordinate occurs in the carrier.

Now inspect the equality-facing row. The comparison Q is a $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row, and the tolerance ledger T records the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ budget used by the comparison before the terminal seal can be consumed.

The terminal $\mathsf{Real}^{\uparrow}$ row A is therefore reached only after the finite extensionality route is visible. The rows H, C, P, N transport, replay, source, and name that same route, so substitution exports the displayed finite evidence and none of the excluded host principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealExtensionality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealExtensionality}^{\uparrow}$ packet is generated from two StreamName finite windows, two RegSeqRat readbacks, a CauchyEquivalenceSetoid comparison row, a RealModulusOfConvergence tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopRealExtensionality}\uparrow}$, seed)

Scope closed: Definition 6491.1, Definition 6491.2, and Definition 6491.3 seed the constructive Real extensional substitution route over finite windows, regular readbacks, comparison, tolerance, seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopRealExtensionality}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, host real equality, selected representatives, countable choice, detached limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular readback, Cauchy-equivalence comparison, tolerance-ledger exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealExtensionalityUp.

Chapter 6492

A Concrete Naming Certificate for $\mathsf{LocatedCoverSelection}^{\uparrow}$

Following Bishop and Bridges on constructive compactness and located finite covers, the $\mathsf{LocatedCoverSelection}^{\uparrow}$ packet records a finite selection surface for already displayed located cover rows. It sits over the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the finite schedule discipline of chapter 126, the dyadic tolerance rows of chapter 291, the compact metric source of chapter 129, and the finite real cover packet of chapter 2377. The packet selects only among displayed located rows; it does not import arbitrary open-cover compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6492.1 (Located cover selection carrier)**.** *A* $\mathsf{LocatedCoverSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, W, D, K, F, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected compact-source points and finite cover centers.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the selected real data are exposed as regular rational windows.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule that orders the displayed located-cover rows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and tolerance ledger used by the membership and overlap tests.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose total bounded and located metric data make the finite request visible.*
6. *F is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row listing the finite candidate cover cells, centers, radii, and coverage witnesses.*
7. *S is the selected finite subrow of F, indexed only by entries already displayed in W and checked through D.*
8. *H records componentwise* hsame *transport for real, regular, stream, dyadic, compact, cover, selection, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from compact metric data through the finite cover row and then through the selected located subrow.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCoverSelection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCoverSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, D, K, F, S, H, C, P, N$. It has no coordinate for arbitrary open covers, host compactness, selected subcovers outside the displayed finite rows, countable choice, quotient equality of reals, or ambient completeness.

Theorem 6492.2 (Located cover selection NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCoverSelection}^{\uparrow}$ *carrier*

$$L = (R, Q, W, D, K, F, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCoverSelection}^{\uparrow}}$ *surface consists exactly of real-seal exposure through R, regular readback through Q, finite schedule exposure through W, dyadic radius and tolerance admission through D, compact metric source exposure through K, finite real cover exposure through F, selected located subrow exposure through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6492.1. The analytic rows are R, Q, W, D, K, F, S, and the structural rows are H, C, P, N.

The finite selection is internal to the displayed packet. Row S is read only after the compact metric source K, the finite cover row F, the finite schedule W, and the dyadic ledger D have exposed the located membership checks.

The real-facing rows R and Q seal and read back the selected data without adding a host equality test. The transport, replay, provenance, and local naming rows preserve that same finite route.

Since no row contains arbitrary open-cover compactness, countable choice, quotient real equality, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 6492.3 (Located cover selection finite-subcover handoff)**.** *Every finite-subcover consumer of an accepted* $\mathsf{LocatedCoverSelection}^{\uparrow}$ *packet must read the route*

$$\mathsf{Real}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{FiniteRealCover}^{\uparrow}$$

through the carrier rows $R, Q, W, D, K, F, S, H, C, P, N$ before consuming the selected located subrow S. The handoff may feed compact metric finite-cover arguments, but it does not choose from an arbitrary open cover, assert a host compactness theorem, identify quotient real representatives, use countable choice, or supply ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6492.2. They force every consumer to expose the real seal, regular readback, finite schedule, dyadic ledger, compact source, finite cover, selected subrow, and structural rows.

The compact metric and finite cover siblings then determine the finite request. The row K supplies the compact metric source data, while F lists the available cover cells and witnesses. Rows W and D make the located membership checks finite and dyadic.

The selected subrow S is therefore a handoff from displayed finite data, not an external selection principle. Because H, C, P, N only transport, replay, source, and name the same route, no refused compactness, equality, choice, or completeness endpoint is exported. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCoverSelection}^{\uparrow}$ consumer exposes RealUp seals, RegSeqRat readbacks, StreamName schedules, DyadicRatCore radius rows, CompactMetric source data, a FiniteRealCover row, a selected displayed subrow, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any finite-subcover handoff is consumed.

Independence witness: real, regseqrat, streamname, dyadicratcore, compactmetric, finite-real-cover: none is bijective with $\mathsf{LocatedCoverSelection}^{\uparrow}$ because this packet alone binds a selected located subrow of a displayed finite cover to compact metric and regular rational readback data without external cover selection.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCoverSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCoverSelection}^{\uparrow}$ packet is generated from RealUp seal rows, RegSeqRat readback rows, StreamName finite schedules, DyadicRatCore radius and tolerance ledgers, CompactMetric source rows, FiniteRealCover candidate rows, selected displayed subrows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCoverSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6492.1, Definition 6492.2, and Definition 6492.3 seed the finite located-cover selection route over displayed Real, regular rational, stream, dyadic, compact metric, and finite-cover rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCoverSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, host compactness, selected subcovers outside displayed finite rows, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located subrow exactness, compact-source exposure, finite-cover coverage, dyadic membership replay, transport, provenance, and local naming for BEDC.Derived.LocatedCoverSelectionUp.

Chapter 6493

A Concrete Naming Certificate for $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$

Following Krasnoselskii and Browder on compact-plus-contractive fixed point principles, the KrasnoselskiiFixedPoint packet records the finite BEDC surface for a displayed self-map decomposed into a contraction row and a compact continuous row on a convex carrier. It sits beside the endpoint contraction route of chapter 396, the compact-convex handoff of chapter 1923, the convex carrier of chapter 208, and the compact completion service of chapter 1687. The packet names only the displayed decomposition, compactness, convexity, iteration, and endpoint-seal rows; it does not import a host fixed-point selector, quotient point equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6493.1 (Krasnoselskii fixed point carrier)**.** *A* $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, V, B, S, A, C, I, E, H, R, P, N)$$

with the following displayed rows:

1. *X is the ambient metric or Banach source row on which the fixed-point request is read.*
2. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *admissible carrier row, recording the convex subpacket in which the fixed point must remain.*
3. *B is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *contraction endpoint row for the contractive summand.*
4. *S is the* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *compact-convex endpoint row for the compact continuous summand.*
5. *A records the displayed algebraic decomposition of the self-map into contractive and compact-continuous rows.*
6. *C is the* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *compactness or completion-facing seal read before compactness is consumed.*
7. *I is the finite iteration-stability ledger joining the contraction budget, compact image window, convex membership row, and endpoint request.*
8. *E records the fixed-point endpoint seal visible to downstream Real or completion consumers.*
9. *H records componentwise* hsame *transport for X, V, B, S, A, C, I, E and the local certificate data.*

10. *R records displayed* Cont *replay from a fixed-point request through the decomposition, compact-convex, contraction, iteration, and endpoint rows.*
11. *P is the* Pkg *provenance over* $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompactMetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a selected host fixed point, an external compactness principle, quotient point equality, countable choice, or ambient completeness.

Theorem 6493.2 (Krasnoselskii fixed point NameCert obligations)**.** *For every accepted* $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ *carrier $K = (X, V, B, S, A, C, I, E, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for X and the admissible convex row V;*
2. *contraction endpoint exposure through B;*
3. *compact-convex endpoint exposure through S;*
4. *decomposition exactness for the displayed self-map row A;*
5. *compact completion or compactness handoff through C;*
6. *iteration-stability routing through I;*
7. *endpoint-seal nonescape through E;*
8. *componentwise* hsame *transport through H;*
9. *displayed* Cont *replay through R;*
10. Pkg *provenance and local naming through P, N.*

No obligation is stored outside the finite carrier.

Proof. Project the carrier of Definition 6493.1. The rows X, V, B, S, A, C, I, E are the only mathematical data available before the transport, replay, provenance, and local naming rows are read.

The fixed-point request then splits along the displayed decomposition row A. The contraction endpoint is read through the Banach fixed-point row B, while the compact continuous endpoint is read through the Schauder row S after the convex carrier V and compactness handoff C are visible.

The iteration ledger I binds those two endpoint surfaces to the same fixed-point request. The rows H, R, P, N only transport, replay, source, and name that finite route, so the listed obligations exhaust the local certificate. □

Theorem 6493.3 (Krasnoselskii fixed point iteration stability)**.** *Every transported* $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ *fixed-point read preserves the same decomposition row A, the same contraction endpoint row B, the same compact-convex endpoint row S, and the same convex carrier row V before the endpoint seal E is consumed. In particular, the iteration ledger cannot be detached into an independent Banach route, an independent Schauder route, or a host-selected fixed point.*

Proof. Begin with the componentwise transport row H. It moves the ambient carrier, convex admissible row, Banach endpoint row, Schauder endpoint row, decomposition row, compact handoff, iteration ledger, and endpoint seal together.

Apply the NameCert obligations of Definition 6493.2. The fixed-point read must expose A before either endpoint row is consumed, and it must expose V and C before the compact-convex side is read.

Replay through R then follows the same decomposition and iteration route. Since I has no coordinate outside the carrier and E is only the downstream seal for this route, transport cannot replace the packet by a separate Banach fixed point, a separate Schauder fixed point, or a selected host endpoint. □

Falsifiable prediction: Every accepted $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ consumer that reads a fixed-point endpoint must first expose one displayed decomposition row, one $\mathsf{BanachFixedPoint}^{\uparrow}$ contraction row, one $\mathsf{SchauderFixedPoint}^{\uparrow}$ compact-convex row, one $\mathsf{ConvexSet}^{\uparrow}$ carrier row, one $\mathsf{CompactMetricCompletion}^{\uparrow}$ compactness handoff, one iteration-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: banach-fixed-point, schauder-fixed-point, convexset, compact-metric-completion: $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the contraction endpoint, compact continuous endpoint, convex admissible carrier, compactness handoff, decomposition ledger, endpoint seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}$ packet is generated from an ambient carrier row, a ConvexSetUp admissible row, BanachFixedPointUp and SchauderFixedPointUp endpoint rows, a compact completion handoff, a decomposition ledger, iteration-stability rows, an endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6493.1, Definition 6493.2, and Definition 6493.3 seed the finite compact-plus-contractive fixed-point packet over convex, compact, Banach, and Schauder rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KrasnoselskiiFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host fixed-point selector, quotient point equality, open-cover compactness, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, decomposition exactness, contraction endpoint exposure, compact-convex endpoint exposure, compact handoff, iteration stability, transport, replay, provenance, and local naming for BEDC.Derived.KrasnoselskiiFixedPointUp.

Chapter 6494

A Concrete Naming Certificate for $\mathsf{FiniteSubstrateMap}^{\uparrow}$

The $\mathsf{FiniteSubstrateMap}^{\uparrow}$ packet realizes the substrate subhub section 9389.10 as a finite map from BEDC audit rows to cellular substrate rows. It sits after the finite trace windows of chapter 3481, the cellular carrier surface of chapter 942, the carrier-encoding row of chapter 943, the local update rule of chapter 984, and the finite orbit carrier of chapter 938. The packet names a finite substrate map, not a hidden evaluator, infinite cellular orbit, host compiler, quotient BHist equality, or all-time cellular induction.

Definition 6494.1 (Finite substrate map carrier)**.** *A* $\mathsf{FiniteSubstrateMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, E, S, W, L, O, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* BHist *audit row to be read as substrate data.*
2. *E is the* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *row translating finite audit entries into cellular carrier entries.*
3. *S is the* $\mathsf{CellularSubstrate}^{\uparrow}$ *row that receives the encoded carrier.*
4. *W is the* $\mathsf{CellularWindowTrace}^{\uparrow}$ *finite trace window on which the map is observed.*
5. *L is the* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ *local rule row used by each displayed successor read.*
6. *O is the* $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ *finite orbit row generated by the encoded carrier and local rule over the displayed window.*
7. *M is the finite map ledger pairing each accepted* BHist *source entry with its encoded carrier cell, local update read, orbit entry, and exported substrate observation.*
8. *H records componentwise* hsame *transport for B, E, S, W, L, O, M and the local certificate data.*
9. *C records displayed* Cont *replay from a source audit request through the encoding, substrate, window, local update, orbit, and map-ledger rows.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteSubstrateMap}^{\uparrow}$, BHist, $\mathsf{CellularCarrierEncoding}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{CellularWindowTrace}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularOrbitSubstrate}^{\uparrow}$, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubstrateMap}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through these finite rows. It has no coordinate for an unbounded substrate, a detached machine evaluator, quotient audit equality, or a cellular observation outside the displayed window.

Lean4 (definition) `BEDC.Derived.FiniteSubstrateMapUp.TasteGate.FiniteSubstrateMapUp`

Theorem 6494.2 (Finite substrate map NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubstrateMap}^{\uparrow}$ *carrier* $F = (B, E, S, W, L, O, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteSubstrateMap}^{\uparrow}}$ *surface consists exactly of source audit admission, cellular carrier encoding, cellular substrate admission, finite window trace exposure, local update exposure, finite orbit exposure, map-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every exported substrate observation factors through the finite route*

$$B, E, S, W, L, O, M, H, C, P, N.$$

Proof. Project the carrier of Definition 6494.1. The only substrate-facing coordinates are the source audit row, encoding row, substrate row, finite window trace, local update rule, orbit row, and map ledger.

The map ledger M pairs source entries with encoded carrier cells and the finite orbit observations obtained through W, L, O. Thus an exported substrate observation cannot be read before the source, encoding, local update, and orbit rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name this same finite route. Since the classifier has no coordinate outside the tuple, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteSubstrateMapUp.TasteGate.FiniteSubstrateMapTasteGate single carrier alignment`

Theorem 6494.3 (Finite substrate map observation nonescape)**.** *Every consumer-visible observation exported by an accepted* $\mathsf{FiniteSubstrateMap}^{\uparrow}$ *packet remains inside the displayed finite window trace* W *and the map ledger* M*. It cannot be replaced by an infinite cellular orbit, a host evaluator, a quotient* BHist *equality, or a detached substrate classifier.*

Proof. Begin with the obligation surface of Definition 6494.2. It forces every observation through the displayed source row, encoding row, substrate row, window trace, local rule, orbit row, and finite map ledger.

The orbit row O is read only as the finite orbit accepted over W and L. The map ledger M records which source audit entries are paired with those visible orbit entries, so a consumer cannot ask for an unlisted time, cell, or classifier read.

Replay through C and transport through H preserve the same finite route. The provenance and local naming rows do not add evaluator, quotient, or infinite-orbit coordinates, so observation nonescape is internal to the packet. □

The later Lean TasteGate namespace BEDC.Derived.FiniteSubstrateMapUp.TasteGate must provide Nontrivial $\mathsf{FiniteSubstrateMap}^{\uparrow}$ and FieldFaithful $\mathsf{FiniteSubstrateMap}^{\uparrow}$ instances for the displayed finite carrier. Those instances should read the carrier rows $B, E, S, W, L, O, M, H, C, P, N$ directly: nontriviality is witnessed by a source audit entry with a displayed encoded cell, and field faithfulness is witnessed by the map ledger preserving the source, encoding, local update, and orbit fields without collapsing them to a host evaluator.

Falsifiable prediction: Every accepted $\mathsf{FiniteSubstrateMap}^{\uparrow}$ observation factors through one source BHist row, one $\mathsf{CellularCarrierEncoding}^{\uparrow}$ row, one $\mathsf{CellularSubstrate}^{\uparrow}$ row, one $\mathsf{CellularWindowTrace}^{\uparrow}$

row, one $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ row, one $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ row, one finite map ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: cellular-window-trace, cellularsubstrate, cellularcarrierencoding, cellularlocalupdaterule, cellularorbitsubstrate: $\mathsf{FiniteSubstrateMap}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source audit row, carrier encoding, substrate row, finite window trace, local update rule, orbit row, finite source-to-substrate map ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubstrateMap}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSubstrateMap}^{\uparrow}$ packet is generated from a BHist source row, CellularCarrierEncodingUp encoding rows, CellularSubstrateUp carrier rows, CellularWindowTraceUp trace windows, CellularLocalUpdateRuleUp local rules, CellularOrbitSubstrateUp finite orbit rows, a finite map ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSubstrateMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6494.1, Definition 6494.2, and Definition 6494.3 seed the finite source-to-cellular-substrate map over BHist, carrier encoding, window trace, local update, and orbit rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSubstrateMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a hidden evaluator, infinite cellular orbit, host compiler, quotient BHist equality, or all-time cellular induction.

Upgrade path: Obligation closure requires checked carrier admission, source audit exposure, carrier encoding, cellular substrate admission, finite window trace exposure, local update exposure, finite orbit exposure, map-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubstrateMapUp.

Chapter 6495

A Concrete Naming Certificate for $\mathsf{SoberSpace}^{\uparrow}$

Following Kelley and Johnstone on sober spaces and locale theory, the $\mathsf{SoberSpace}^{\uparrow}$ packet records the finite point-reconstruction boundary through which displayed open observations determine points without importing host powersets or point equality. It sits after the topology carrier of chapter 88, the observable-open classifier of chapter 1860, the separation discipline of chapter 3338, and the point-free open route of chapter 5153. The packet names the standard sobriety handoff as finite NameCert data; it does not assert spatiality for every locale, quotient point reconstruction, host equality of points, excluded middle over opens, or choice over completely prime filters.

Definition 6495.1 (Sober space carrier)**.** *A* $\mathsf{SoberSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, O, L, Z, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *row, recording finite open-index and neighbourhood observations for the space under inspection.*
2. *O is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row, exposing open membership only through displayed Bool truth, Topology open, and finite StreamName windows.*
3. *L is the* $\mathsf{LocaleSpace}^{\uparrow}$ *comparison row, recording finite frame-of-opens and cover-refinement reads that can be compared with T and O.*
4. *Z is the* $\mathsf{TZeroReflection}^{\uparrow}$ *compatibility row, requiring specialization readback to separate displayed point names only through observable opens.*
5. *H records componentwise* hsame *transport for topology rows, Sierpinski truth rows, locale comparison rows, T0 compatibility, replay, provenance, and the local naming row.*
6. *C records displayed* Cont *replay from a sobriety request through T, O, L, Z, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{SoberSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{TZeroReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SoberSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, O, L, Z, H, C, P, N. It has no coordinate for host powersets of opens, quotient point spaces, selected generic points, global locale spatiality, excluded middle over opens, choice over completely prime filters, or ambient point equality.

Theorem 6495.2 (Sober space NameCert obligations)**.** *For every accepted* $\mathsf{SoberSpace}^{\uparrow}$ *carrier*

$$S = (T, O, L, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SoberSpace}^{\uparrow}}$ *surface consists exactly of topology-row admission, Sierpinski observable-open truth admission, locale/frame-of-opens comparison admission, T0 specialization compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every point-reconstruction consumer must inspect those rows before treating a displayed completely prime open profile as a point-facing sober-space read.*

Proof. Project the carrier of Definition 6495.1. The topology row T supplies the finite open-index observations, while the Sierpinski row O supplies observable-open truth without turning membership into a host proposition.

The locale row L compares the same open observations with finite frame and cover-refinement data. The T0 row Z then constrains specialization readback so that displayed point names are separated only by observable opens already visible through T, O, L.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores host powersets, quotient point spaces, selected generic points, global locale spatiality, excluded middle, choice over completely prime filters, or ambient point equality, the listed rows exhaust the local obligation surface. □

Theorem 6495.3 (Sober space Sierpinski locale handoff)**.** *Any* $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{TZeroReflection}^{\uparrow}$, *or observable-open consumer that reads a* $\mathsf{SoberSpace}^{\uparrow}$ *packet must route the read through*

$$T, O \longmapsto L \longmapsto Z \longmapsto H, C, P, N.$$

The handoff compares open membership using the $\mathsf{SierpinskiSpace}^{\uparrow}$ *truth row and compares the corresponding open frame using the* $\mathsf{LocaleSpace}^{\uparrow}$ *row before specialization compatibility is consumed. It exports no host proposition for open membership, no powerset of opens, no quotient point space, no selected generic point, and no excluded middle over opens.*

Proof. Start with an accepted carrier $S = (T, O, L, Z, H, C, P, N)$. A topology-facing read first exposes T, and an observable-open read exposes O as the displayed Sierpinski truth route.

The locale comparison row L is read only after those open observations are visible. It aligns the topology open rows with finite frame and cover-refinement data rather than replacing them by a host powerset.

Specialization compatibility is then the row Z. It may separate displayed point names through observable opens, but it cannot select a hidden generic point or quotient point representative.

The support rows H, C, P, N preserve the same route by transport, replay, provenance, and naming. Therefore every consumer receives the Sierpinski and locale handoff as finite packet data, not as an external proposition or choice principle. □

Theorem 6495.4 (Sober space specialization basis)**.** *Every accepted* $\mathsf{SoberSpace}^{\uparrow}$ *point-reconstruction read separates displayed point names only by first reading the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row* O *and then the* $\mathsf{TZeroReflection}^{\uparrow}$ *specialization row* Z*. The* $\mathsf{LocaleSpace}^{\uparrow}$ *comparison row* L *may align the finite frame-of-opens rows with the same observations, but it cannot replace observable-open truth by host propositions, host powersets, selected generic points, excluded middle, or quotient point equality.*

Proof. Project the carrier rows O, L, Z from Definition 6495.1. The row O is the only observable-open truth interface, L is the only locale comparison row, and Z is the only specialization readback row.

Apply the Sierpinski-locale handoff of Definition 6495.3. It routes topology and observable-open data through O, compares the corresponding frame rows through L, and only then reads specialization compatibility through Z.

Use point-reconstruction nonescape, Definition 6495.5. Because the carrier has no coordinate for host powersets, selected generic points, excluded middle, quotient point equality, or ambient point equality, a point-reconstruction consumer cannot bypass the observable truth row.

Thus displayed point names are separated through Sierpinski observable opens before T0 specialization readback, while locale comparison only aligns the finite open-frame rows already exposed by the packet. □

Theorem 6495.5 (Sober space point reconstruction nonescape)**.** *Every point-reconstruction read exported by an accepted* $\mathsf{SoberSpace}^{\uparrow}$ *packet factors through the displayed topology row, Sierpinski observable-open row, locale comparison row, and T0 specialization row before it reaches transport, replay, provenance, and local naming. Thus the packet can witness the sober-space boundary for displayed open profiles, but it cannot export host point equality, quotient spatial reflection, selected completely prime filters, choice-indexed generic points, excluded middle over opens, or global spatiality for arbitrary locales.*

Proof. Apply Definition 6495.2. The local NameCert surface stores every accepted sobriety obligation in T, O, L, Z, H, C, P, N and stores none outside those rows.

Apply Definition 6495.3. It identifies the consumer route from topology and observable-open data through locale comparison and T0 specialization compatibility.

Since the carrier inventory has no coordinate for host point equality, quotient spatial reflection, selected completely prime filters, choice-indexed generic points, excluded middle over opens, or global locale spatiality, no point-reconstruction consumer can obtain those objects from the packet. □

Falsifiable prediction: Removing any one of the $\mathsf{Topology}^{\uparrow}$ row, $\mathsf{SierpinskiSpace}^{\uparrow}$ observable-open truth row, $\mathsf{LocaleSpace}^{\uparrow}$ comparison row, $\mathsf{TZeroReflection}^{\uparrow}$ compatibility row, hsame transport, Cont replay, Pkg provenance, or local NameCert row makes Definition 6495.5 false: a point-reconstruction consumer could no longer verify the displayed sober-space route.

Independence witness: topology, sierpinski, locale, tzero: $\mathsf{SoberSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries topology observations, observable-open truth, locale/frame comparison, T0 specialization compatibility, transport, replay, provenance, and local naming rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SoberSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SoberSpace}^{\uparrow}$ packet is generated from TopologyUp open rows, SierpinskiSpaceUp observable-open truth rows, LocaleSpaceUp frame comparison rows, TZeroReflectionUp specialization compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SoberSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6495.1, Definition 6495.2, Definition 6495.3, and Definition 6495.5 seed the finite sober-space point-reconstruction route over topology, observable-open, locale, T0 compatibility, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SoberSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host point equality, quotient spatial reflection, selected completely prime filters, choice-indexed generic points, excluded middle over opens, global spatiality for arbitrary locales, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topology admission, observable-open classifier admission, locale comparison exactness, T0 specialization compatibility, transport, replay, provenance, and local naming for BEDC.Derived.SoberSpaceUp.

Chapter 6496

A Concrete Naming Certificate for $\mathsf{RealLocatedSupremum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLocatedSupremum}^{\uparrow}$ packet records the finite surface by which a located bounded family of constructive real names exposes a supremum read without Dedekind cuts, quotient equality of real names, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the located supremum carrier of chapter 1427, and the narrower supremum-modulus ledger of chapter 6432. The packet names the complete Real-facing located-supremum route as displayed BEDC rows.

Definition 6496.1 (Real located supremum carrier)**.** *A* $\mathsf{RealLocatedSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, D, R, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite located family row, listing the inspected real-name sources and their displayed boundedness witnesses.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the inspected family entries and candidate supremum windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording upper-bound gaps, lower-hit gaps, and product-free comparison budgets.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic tolerance entries.*
5. *M is the* $\mathsf{LocatedSupremumModulus}^{\uparrow}$ *row, exposing the search modulus that turns located upper-bound and lower-hit requests into finite dyadic evidence.*
6. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, read only after F, W, D, R, M have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the supremum candidate named by L.*
8. *H records componentwise* hsame *transport for family, window, dyadic, regular-rational, modulus, located-supremum, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real-facing located-supremum request through $F, W, D, R, M, L, E, H, P, N$*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremumModulus}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocatedSupremum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, D, R, M, L, E, H, C, P, N$. It has no coordinate for Dedekind cuts, selected representatives of real names, arbitrary subset completeness, trichotomy, countable choice, or host least-upper-bound operators.

Theorem 6496.2 (Real located supremum NameCert obligations)**.** *For every accepted* $\mathsf{RealLocatedSupremum}^{\uparrow}$ *carrier*

$$S = (F, W, D, R, M, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLocatedSupremum}^{\uparrow}}$ *surface consists exactly of finite located-family admission through F, synchronized window exposure through W, dyadic upper-bound and lower-hit tolerance exactness through D, regular rational readback through R, located supremum-modulus exposure through M, located-supremum handoff through L, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6496.1. The analytic rows are F, W, D, R, M, L, E, and the structural rows are H, C, P, N.

The family and candidate windows are finite before any supremum read occurs. Row F names the displayed real family and boundedness witnesses, while W fixes the inspected family windows and candidate supremum windows.

The located comparison data then pass through D, R, M. The dyadic ledger records upper-bound gaps and lower-hit gaps, the regular rational row reads those windows constructively, and the located supremum-modulus row supplies the finite search response needed before the located-supremum handoff L can be consumed.

The real seal E is downstream of that handoff. Since the carrier contains no Dedekind-cut coordinate, quotient real equality test, trichotomy row, countable-choice schedule, arbitrary subset-completeness row, or host least-upper-bound operator, the listed rows exhaust the local NameCert obligations. □

Theorem 6496.3 (Real located supremum seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RealLocatedSupremum}^{\uparrow}$ *carrier factors through the displayed route*

$$F, W \longmapsto D \longmapsto R \longmapsto M \longmapsto L \longmapsto E \longmapsto C, P, N.$$

Thus the packet supplies a constructive located-supremum interface over Real names and dyadic evidence, but it does not export a Dedekind-cut construction, a selected host supremum, quotient real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with a Real-facing consumer request. The only real endpoint row of the carrier is E, and the definition makes E reachable only after the located family, window schedule, dyadic ledger, regular rational readback, modulus row, and located-supremum handoff have been displayed.

Read F and W first. These rows determine which real-name sources and candidate windows are visible to the request.

Next read D, R, M, L. The dyadic ledger supplies the finite upper-bound and lower-hit budgets, the regular rational row reads them through the same windows, the modulus row supplies the finite search response, and the located-supremum row names the handoff to the supremum candidate.

Finally H, C, P, N transport, replay, source, and name the same finite route. None of these structural rows introduces a host supremum object or any of the refused classical principles, so every exported real read is exhausted by the displayed route. □

Falsifiable prediction: A $\mathsf{RealLocatedSupremum}^{\uparrow}$ consumer that exports a RealUp supremum read while omitting finite located-family rows, StreamName windows, DyadicRatCore upper-bound and lower-hit rows, RegSeqRat readback, LocatedSupremumModulus evidence, LocatedSupremum handoff, RealUp seal, hsame transport, Cont replay, Pkg provenance, or local NameCert evidence is not reading the carrier of Definition 6496.1.

Independence witness: real, regseqrat, dyadicratcore, locatedsupremum, located-supremum-modulus: none is bijective with $\mathsf{RealLocatedSupremum}^{\uparrow}$ because this packet alone binds a displayed Real-name family, synchronized windows, dyadic upper-bound and lower-hit evidence, regular rational readback, a located supremum-modulus row, a located-supremum handoff, and the terminal Real seal into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocatedSupremum}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocatedSupremum}^{\uparrow}$ packet is generated from finite located Real-family rows, StreamName windows, DyadicRatCore upper-bound and lower-hit ledgers, RegSeqRat readback, a LocatedSupremumModulus row, a LocatedSupremum handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocatedSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6496.1, Definition 6496.2, and Definition 6496.3 seed the constructive Real located-supremum route over displayed Real, dyadic, regular-rational, modulus, and supremum handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocatedSupremum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut construction, quotient real equality, selected host suprema, arbitrary subset completeness, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window coverage, dyadic upper-bound and lower-hit exactness, RegSeqRat readback, LocatedSupremumModulus exposure, LocatedSupremum handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealLocatedSupremumUp.

Chapter 6497

A Concrete Naming Certificate for $\mathsf{HyperbolicAxisProjection}^{\uparrow}$

The $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ packet realizes an axis-facing branch of chapter 9463 as finite disk-metric, geodesic-flow, boundary-shadow, Busemann, visual-boundary, projection-readback, refusal, provenance, and NameCert rows. It sits after the Poincare disk metric packet of chapter 6209, the geodesic-flow carrier of chapter 3856, the visual boundary route of chapter 4181, the Busemann boundary packet of chapter 3732, and the axis-shadow route of chapter 6122. The projection is a BEDC-internal row discipline: it names a displayed axis read only after the boundary and Busemann evidence needed to keep the read non-selected has been opened.

Definition 6497.1 (Hyperbolic axis projection carrier)**.** *A* $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, M, G, V, B, S, X, R, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric row that supplies the inspected disk coordinates and metric comparisons.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same disk observations.*
3. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row naming the displayed oriented geodesic-flow segment whose axis projection is requested.*
4. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row exposing the boundary endpoints and visual-boundary readback used by the projection.*
5. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row recording the Busemann-level boundary comparison attached to the same endpoints.*
6. *S is the boundary-shadow row, read through the shadow or axis-shadow sibling route before an axis projection may be consumed.*
7. *X is the axis-projection readback row, recording the finite projected axis window and its attachment to D, M, G, V, B, S.*
8. *R is the refusal row excluding selected global axes, quotient ideal-boundary equality, completed hyperbolic geometry, host nearest-point projection, and physical observer-axis laws.*

9. *F is the TasteGate-facing row requiring the future Lean* $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ *carrier to provide* `Nontrivial` *and* `FieldFaithful` *instances for the disk, metric, flow, boundary, Busemann, shadow, projection, refusal, replay, provenance, and naming fields.*
10. *H records componentwise* hsame *transport for disk-metric, hyperbolic-metric, geodesic-flow, visual-boundary, Busemann-boundary, boundary-shadow, projection, refusal, TasteGate-facing, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from disk and geodesic-flow observations through V, B, S to the projection row X.*
12. *P is the* Pkg *provenance over* $\mathsf{HyperbolicAxisProjection}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicAxisProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, G, V, B, S, X, R, F, H, C, P, N$. It has no coordinate for selected axes, quotient boundary equality, completed disk geometry, host nearest-point projection, physical expansion dynamics, Lean verification, or a standard bridge.

Theorem 6497.2 (Hyperbolic axis projection NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ *carrier*

$$A = (D, M, G, V, B, S, X, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicAxisProjection}^{\uparrow}}$ *surface consists exactly of Poincare disk metric admission through D, hyperbolic metric readback through M, geodesic-flow admission through G, visual-boundary exposure through V, Busemann-boundary exposure through B, boundary-shadow exposure through S, axis-projection readback through X, refusal exactness through R, TasteGate-facing non-triviality and field-faithfulness requirements through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6497.1. The metric and flow rows are D, M, G, the boundary rows are V, B, S, the projection row is X, and the structural rows are R, F, H, C, P, N.

The projection is not available from metric data alone. The disk and hyperbolic metric rows determine the inspected coordinates, and the geodesic-flow row names the oriented segment, but the carrier requires visual-boundary, Busemann-boundary, and boundary-shadow rows before X may be consumed.

The refusal and TasteGate-facing rows keep the carrier separated. Row R excludes selected global axes, quotient boundary equality, completed disk geometry, host nearest-point projection, and physical expansion dynamics, while F records the future Lean requirement that the listed fields remain noncollapsed and faithful.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row supplies an undisplayed axis or a host projection theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 6497.3 (Hyperbolic axis projection boundary dependency)**.** *Every accepted* $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ *read that exports an axis-projection row must first display*

$\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBu}$

together with the boundary-shadow row S. Consequently an axis-projection row cannot preserve its local $\mathsf{NameCert}$ *surface when transported to a sibling packet lacking boundary-shadow and Busemann rows.*

Proof. Start from the NameCert obligations of Definition 6497.2. They make X downstream of D, M, G, V, B, S.

The visual-boundary row V names the endpoints, while the Busemann-boundary row B keeps the boundary comparison attached to the same endpoints. The shadow row S then records the non-selected boundary-shadow witness needed before the axis projection can be read.

If a sibling lacks B or S, the displayed route to X is missing one of its carrier coordinates. Transport through H, C, P, N can preserve an existing row, but it cannot create absent Busemann or boundary-shadow evidence.

Thus the accepted projection read remains local to the full axis-projection packet and does not become a host nearest-point projection or a boundary-only sibling certificate. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ read that preserves local NameCert evidence into a sibling lacking boundary-shadow and Busemann rows refutes the carrier surface of Definition 6497.1.

Independence witness: hyperbolic-visual-boundary, hyperbolic-busemann-boundary, hyperbolic-geodesic-flow, poincare-disk-metric: $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ is not bijective to any listed sibling because each sibling lacks at least one obligation row among boundary-shadow evidence, Busemann comparison, geodesic-flow attachment, disk-metric provenance, axis-projection readback, refusal, TasteGate-facing field separation, transport, replay, provenance, and local naming.

The future Lean TasteGate for $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ must provide `Nontrivial` and `FieldFaithful` instances that keep $D, M, G, V, B, S, X, R, F, H, C, P, N$ as separate fields before any carrier-alignment theorem can serve as a public TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicAxisProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicAxisProjection}^{\uparrow}$ packet is generated from PoincareDiskMetric rows, HyperbolicMetric reads, HyperbolicGeodesicFlow rows, HyperbolicVisualBoundary rows, HyperbolicBusemannBoundary rows, boundary-shadow evidence, axis-projection readback, refusal, TasteGate-facing field-separation requirements, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicAxisProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6497.1, Definition 6497.2, and Definition 6497.3 seed the finite route from disk metric and geodesic-flow rows through boundary-shadow and Busemann evidence to an axis-projection readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicAxisProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected global axes, quotient boundary equality, completed hyperbolic geometry, host nearest-point projection, physical observer-axis dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-metric visibility, geodesic-flow attachment, visual-boundary exposure, Busemann-boundary exposure, boundary-shadow dependency, axis-projection readback, refusal, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicAxisProjectionUp.

Chapter 6498

A Concrete Naming Certificate for $\mathsf{FrechetFilterBasis}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{FrechetFilterBasis}^{\uparrow}$ packet isolates the cofinite-tail basis that generates the Frechet filter before any consumer reads a full filter surface. It sits between the finite-tail rows of chapter 1119, the filter-base rows of chapter 1601, and the FrechetFilter route of chapter 1564. Sequential consumers read the same finite basis through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal, never through quotient filter equality or a selected limit.

Definition 6498.1 (FrechetFilterBasis carrier)**.** *A* $\mathsf{FrechetFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, T, F, S, Q, M, R, A, H, C, P, N)$$

whose rows have the following roles:

1. *U is the displayed unary tail-threshold row.*
2. *T is the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *finite-complement row, recording tail containment by finite data.*
3. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *meet row for cofinite-tail neighborhoods.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row read only after U, T, F.*
5. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row built from that basis.*
6. *M is the* $\mathsf{FrechetFilter}^{\uparrow}$ *consumer row receiving the generated filter surface.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular readback.*
9. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FrechetFilterBasis}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient filter, external cofinite-set algebra, countable tail selector, selected limit, ambient completion, or independent $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Theorem 6498.2 (FrechetFilterBasis NameCert obligations)**.** *For every accepted* $\mathsf{FrechetFilterBasis}^{\uparrow}$ *carrier $B = (U, T, F, S, Q, M, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetFilterBasis}^{\uparrow}}$ *consists exactly of carrier admission for those rows, cofinite-tail basis admission through U, T, F, common-tail meet stability through F, FrechetFilter handoff through M, CauchyFilter and sequential-completion*

handoff through Q, S, R, A, transport and replay through H, C, provenance through P, and the refusal of quotient filters, selected limits, countable selectors, ambient completion, and independent Real completeness.

Proof. Project the carrier. The only rows that can mention cofinite tails are U, T, F, and the only rows that can export that surface are M and Q.

The meet row F supplies the common-tail refinement before the stream schedule S and regular readback R can be inspected. The replay row C fixes that order, while H transports it componentwise and P, N name only the displayed packet.

Thus every listed obligation is read from $U, T, F, S, Q, M, R, A, H, C, P, N$. The refused objects are absent from the carrier, so the local certificate cannot project them. □

Theorem 6498.3 (FrechetFilterBasis CauchyFilter handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetFilterBasis}^{\uparrow}$ *carrier reads the finite cofinite-tail basis through*

$$U, T, F \longmapsto S, Q \longmapsto R, A, H, C, P, N.$$

The handoff also supplies the $\mathsf{FrechetFilter}^{\uparrow}$ *row M as the generated filter surface, but it exports no quotient filter equality, countable tail selector, selected limit, ambient completion, or separate Real-completeness object.*

Proof. Read U, T, F from Definition 6498.1. They are the threshold, finite-complement, and FilterBase meet rows, hence the complete basis source.

The continuation row then places S and Q downstream of that source. A CauchyFilter read cannot reach Q before the finite basis has been displayed, and the FrechetFilter row M receives the same generated surface.

The readback and Real rows R, A are later consumers of the same route. Since no refused host object is a carrier coordinate, the handoff remains a finite NameCert route. □

Theorem 6498.4 (FrechetFilter sequential-completion tail-basis route)**.** *Every* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *consumer that starts from a* $\mathsf{FrechetFilter}^{\uparrow}$ *tail source may instead read the narrower* $\mathsf{FrechetFilterBasis}^{\uparrow}$ *route*

$$\mathsf{FiniteTailFilter}^{\uparrow}(U, T),\ \mathsf{FilterBase}^{\uparrow}(F),\ \mathsf{StreamName}^{\uparrow}(S),\ \mathsf{CauchyFilter}^{\uparrow}(Q),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(A),\ H, C, P, N$$

before treating the source as a completion input. This route is a finite cofinite-tail basis route and not a full quotient-filter or ambient-completion route.

Proof. Use Definition 6498.2 to isolate the admitted rows. The basis part is exactly U, T, F, and the sequential-facing consumers are attached only through S, Q, R, A.

Apply Definition 6498.3. It identifies the CauchyFilter export as the same finite basis read, with the FrechetFilter row M generated from it rather than supplied independently.

Sequential completion therefore consumes the regular-rational and Real-facing rows after the basis handoff. The refused quotient filter, selected limit, countable selector, and ambient completion coordinates are unavailable. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetFilterBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetFilterBasis}^{\uparrow}$ packet is generated from finite tail thresholds, cofinite finite-complement rows, FilterBase meets, StreamName schedules, CauchyFilter handoff, FrechetFilter generation, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetFilterBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6498.1 and Definition 6498.2 seed the cofinite-tail basis surface for FrechetFilter and sequential-completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FrechetFilterBasis}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient filter equality, selected limits, countable tail selection, ambient completion, or independent Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-tail meet stability, FrechetFilter handoff, CauchyFilter handoff, sequential-completion routing, transport, replay, provenance, and local naming for BEDC.Derived.FrechetFilterBasisU

Chapter 6499

A Concrete Naming Certificate for $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$

Following Bishop–Bridges and the constructive Heine–Cantor argument, the $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$ packet packages the finite $\mathsf{ProbeBundle}$ route from compact coverage and pointwise continuity to a tolerance-only uniform modulus. It realizes the compact-uniform-continuity carrier of chapter 9237 as concrete NameCert data: compact finite-net rows, continuous-map pointwise-modulus rows, $\mathsf{Rat}^{\uparrow}$ lower-bound folds, metric triangle estimates, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff all remain visible in one finite carrier. The packet consumes the concrete $\mathsf{CompactMetric}^{\uparrow}$ finite-net carrier of chapter 129, the $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus carrier of chapter 130, the $\mathsf{Rat}^{\uparrow}$ lower-bound fold arithmetic of chapter 37, the $\mathsf{Met}^{\uparrow}$ triangle rows of chapter 57, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369.

Definition 6499.1 (CompactFiniteModulusGluing carrier)**.** *A* $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, C, B, R, L, T, U, H, E, P, N)$$

with these rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including the finite* $\mathsf{ProbeBundle}$ *and coverage witnesses.*
2. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the same metric source.*
3. *B is the bundled probe row carrying centers, local radii, and local modulus certificates.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius comparison row.*
5. *L is the finite lower-bound fold over the bundled rational radii.*
6. *T is the metric-triangle transport row that moves estimates from centers to arbitrary points.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row from tolerance histories to precision histories.*
8. *H, E, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteModulusGluing}^{\uparrow}}$ *row.*

The carrier has no external finite-subcover selector, no global choice of centers, no host compactness theorem, no classical least positive radius, no function-space quotient equality, and no ambient Real-completeness coordinate.

Theorem 6499.2 (CompactFiniteModulusGluing NameCert obligations)**.** *For every accepted* $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$ *carrier* $G = (K, C, B, R, L, T, U, H, E, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteModulusGlui}}$ *consists exactly of carrier admission for those rows, finite ProbeBundle coverage through* K, B*, pointwise-modulus admission through* C*, positive rational comparison through* R*, finite lower-bound fold stability through* L*, metric-triangle transport through* T*, uniform-modulus handoff through* U*, structural transport and replay through* H, E*, provenance through* P*, and refusal of external selectors, host compactness, classical minima, quotient function equality, and ambient Real completeness.*

Proof. Project the carrier. Coverage and probe membership are stored only in K, B, while pointwise continuity data is stored only in C.

The rational comparison and finite fold rows R, L provide the displayed lower bound over the bundle. The triangle row T is then the only route from center estimates to arbitrary source points, and U receives the resulting tolerance-only precision history.

Transport, replay, provenance, and naming preserve that finite route without adding host selectors or quotient function equality. The refused objects are not coordinates of G, so no NameCert clause can export them. □

Theorem 6499.3 (Compact finite gluing uniform-modulus route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactMetric}^{\uparrow}(K),\ \mathsf{ContinuousMap}^{\uparrow}(C),\ \mathsf{ProbeBundle}(B),\ \mathsf{Rat}^{\uparrow}(R),\ L, T, U, H, E, P, N.$$

The exported precision history is produced by a finite rational lower-bound fold over bundled pointwise-modulus radii, followed by metric-triangle transport, and not by an external modulus selector or a host Heine–Cantor theorem.

Proof. Start with the finite-net row K and the bundled row B. They provide the only displayed centers and coverage witnesses available to the carrier.

Read the pointwise-modulus row C at each bundled center. The positive rational comparisons in R and the finite fold L produce one lower-bound history below all bundled local radii.

Finally use the metric-triangle row T to transport local estimates from the chosen center to an arbitrary pair of nearby source points, and hand the resulting precision history to U. Because the whole route has passed through finite carrier rows, it does not expose external selection, classical minima, quotient equality, host compactness, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFiniteModulusGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactFiniteModulusGluing}^{\uparrow}$ packet is generated from compact finite-net rows, continuous-map pointwise moduli, bundled probe data, rational positive-radius comparison, finite lower-bound folds, metric-triangle transport, uniform-modulus handoff, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactFiniteModulusGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6499.1 and Definition 6499.2 seed the finite ProbeBundle gluing surface for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactFiniteModulusGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host compactness, external finite-subcover selection, classical least-radius selection, quotient function equality, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite ProbeBundle coverage, pointwise-modulus admission, rational fold stability, metric-triangle transport, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactFiniteModulusGluingUp.

Chapter 6500

A Concrete Naming Certificate for ClosedIntervalFiniteNet$^{\uparrow}$

Following Bishop and Bridges on finite epsilon-net data for compact intervals, the ClosedIntervalFiniteNet$^{\uparrow}$ packet records the finite dyadic mesh surface by which a displayed closed real interval can be consumed by total-bounded and compact-metric routes. It sits after the dyadic arithmetic source of chapter 291, the dyadic cell surface of chapter 306, the located-real comparison row of **??**, the terminal real-name seal of chapter 38, the finite epsilon-net interface of chapter 1767, and the total-bounded consumer surface of chapter 127. The packet names explicit endpoint, mesh, coverage, transport, replay, provenance, and local NameCert data; it does not assert open-cover compactness or an ambient Heine–Borel theorem.

Definition 6500.1 (Closed interval finite-net carrier)**.** *A* ClosedIntervalFiniteNet$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, M, L, F, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint row, carrying lower and upper closed-interval endpoint histories together with the finite comparison ledger that places the lower endpoint before the upper endpoint.*
2. *M is the dyadic mesh row, a finite* DyadicInterval$^{\uparrow}$ *subdivision schedule read through* DyadicRatCore$^{\uparrow}$ *arithmetic and a positive tolerance request.*
3. *L is the located* Real$^{\uparrow}$ *seal row, displaying the located-real endpoint reads and the terminal real-name seals used by interval consumers.*
4. *F is the finite-net list row, a closed finite list of mesh centers admitted between the endpoint rows and tagged by the mesh scale.*
5. *W is the coverage witness row, recording for each displayed interval point the inspected center row and the dyadic bound showing that the point is covered at the requested tolerance.*
6. *H records componentwise* hsame *transport for endpoint rows, dyadic mesh rows, located real seals, finite net lists, coverage witnesses, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from an interval finite-net request through E, M, L, F, W, H before any total-bounded or compact-metric consumer reads the packet.*
8. *P is the* Pkg *provenance over* ClosedIntervalFiniteNet$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* DyadicInterval$^{\uparrow}$*,* LocatedReal$^{\uparrow}$*,* Real$^{\uparrow}$*,* FiniteEpsilonNet$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, M, L, F, W, H, C, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, a host Heine–Borel principle, quotient interval equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6500.2 (Closed interval finite-net obligation surface)**.** *For every accepted* $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$ *carrier $K = (E, M, L, F, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNet}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint enclosure, finite-net coverage, mesh-refinement stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and ledger non-escape. A consumer may treat the packet as closed-interval finite-net data only after those rows have been displayed.*

Proof. Project the carrier of Definition 6500.1. Carrier admission is the displayed packet K, endpoint enclosure is the row E, and the dyadic mesh is the row M.

The finite-net rows are F, W. The list row fixes the admitted mesh centers, while the coverage row records the inspected center and dyadic bound for each displayed interval point. Mesh-refinement stability is read through the same endpoint and mesh rows, so a refined request keeps the coverage ledger inside E, M, F, W.

The remaining rows H, C, P, N supply transport, replay, provenance, and local naming for that same finite packet. Since the classifier has no coordinate outside $E, M, L, F, W, H, C, P, N$, no ledger read can escape to arbitrary open covers, selected subcovers, host interval equality, countable choice, or ambient real completeness. □

Theorem 6500.3 (Closed interval finite-net total-bounded handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or compact-metric consumer that uses interval compactness from an accepted* $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$ *packet must first read the finite dyadic net row F and the located endpoint rows E, L. The consumer may then inspect coverage witnesses, mesh refinement, transport, replay, provenance, and local naming, but it receives no open-cover compactness theorem, selected cover, countable-choice schedule, host Heine–Borel principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Let $K = (E, M, L, F, W, H, C, P, N)$ be accepted. The interval boundary is visible only through the endpoint row E and the located $\mathsf{Real}^{\uparrow}$ seal row L, while the finite dyadic net is visible only through M, F.

Apply Definition 6500.2. It forces any total-bounded or compact-metric consumer to inspect carrier admission, endpoint enclosure, finite-net coverage, mesh-refinement stability, transport, replay, provenance, and ledger non-escape before treating the packet as interval finite-net data.

Thus the handoff to $\mathsf{TotallyBounded}^{\uparrow}$ and compact-metric consumers is the displayed finite route $E, M, L, F, W, H, C, P, N$. The excluded open-cover, selected-cover, choice, host Heine–Borel, quotient-interval, and ambient-completeness objects are not carrier fields, so none can be exported by the handoff. □

Falsifiable prediction: Removing the endpoint row, dyadic mesh row, located Real seal row, finite-net list, coverage witness, hsame transport, Cont replay, Pkg provenance, or local $\mathsf{NameCert}$ row makes Definition 6500.3 false: a total-bounded consumer could no longer verify the displayed interval-net route.

Independence witness: dyadicratcore, dyadicinterval, locatedreal, real, finite-epsilon-net, totallybounded: $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$ is not bijective to any listed sibling because it binds endpoint rows, dyadic mesh rows, located Real seals, finite net lists, coverage witnesses, transport, replay, provenance, and local naming into one closed-interval consumer packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$ packet is generated from endpoint rows, a dyadic mesh schedule, located Real seals, a finite net list, coverage witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6500.1, Definition 6500.2, and Definition 6500.3 seed the finite closed-interval net route over endpoint, dyadic mesh, located Real, coverage, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Heine–Borel as a theorem for general subsets, selected covers, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint enclosure, finite-net coverage, mesh-refinement stability, transport, replay, provenance, and ledger non-escape for BEDC.Derived.ClosedIntervalFiniteNetUp.

Chapter 6501

A Concrete Naming Certificate for $\mathsf{BoundedLattice}^{\uparrow}$

Following Birkhoff and Tarski on bounded lattices, the $\mathsf{BoundedLattice}^{\uparrow}$ packet records the finite meet–join interface equipped with displayed bottom and top rows. It sits after the order substrate of chapter 53, the meet–join certificate of chapter 55, the certified-family route of chapter 5155, and the point-free open rows of chapter 5153. The packet supplies the standard bounded-lattice boundary needed by locale, frame, and domain-style consumers without importing host complete orders, powersets, Zorn principles, quotient representatives, or selected extrema.

Definition 6501.1 (Bounded lattice carrier)**.** *A* $\mathsf{BoundedLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, L, \bot, \top, M, J, H, C, P, N)$$

with the following displayed rows:

1. *O is the underlying* $\mathsf{Poset}^{\uparrow}$ *order row, recording the visible comparison surface.*
2. *L is the* $\mathsf{Lattice}^{\uparrow}$ *meet–join row, recording the displayed finite meet and join operations over O.*
3. *$\bot$ is the bottom row, giving a named element below every displayed carrier element by comparisons in O.*
4. *$\top$ is the top row, giving a named element above every displayed carrier element by comparisons in O.*
5. *M records endpoint-meet exactness: displayed meets with $\bot$ and $\top$ reduce through the same order and lattice rows.*
6. *J records endpoint-join exactness: displayed joins with $\bot$ and $\top$ reduce through the same order and lattice rows.*
7. *H records componentwise* hsame *transport for the order, lattice, endpoint, exactness, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded-lattice request through $O, L, \bot, \top, M, J, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedLattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{CompleteLattice}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, L, \bot, \top, M, J, H, C, P, N$. It has no coordinate for a host complete lattice, powerset lattice, Zorn principle, quotient representative, arbitrary infimum or supremum operator, or selected external endpoint.

Theorem 6501.2 (Bounded lattice NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLattice}^{\uparrow}$ *carrier* $B = (O, L, \bot, \top, M, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLattice}^{\uparrow}}$ *surface consists exactly of order-row admission, lattice-row admission, bottom-row exposure, top-row exposure, endpoint meet exactness, endpoint join exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every locale, frame, complete-lattice, or domain-style consumer must expose those rows before reading a bounded-lattice claim.*

Proof. Project the carrier of Definition 6501.1. The order row O supplies the only comparison surface, and the lattice row L supplies the only displayed meet–join interface.

The endpoint rows are then visible coordinates of the same packet. The row $\bot$ is read as the displayed lower endpoint, and the row $\top$ is read as the displayed upper endpoint, both by comparisons in O.

Endpoint exactness is stored in M and J. These rows bind meets and joins with the endpoints to the same order and lattice data, so a consumer cannot substitute an ambient infimum, supremum, powerset operation, or complete-order theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the excluded host objects are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6501.3 (Bounded lattice locale handoff)**.** *Every* $\mathsf{LocaleSpace}^{\uparrow}$*,* $\mathsf{CompleteLattice}^{\uparrow}$*, or frame-style consumer of an accepted* $\mathsf{BoundedLattice}^{\uparrow}$ *packet must read the finite route*

$$O, L, \bot, \top \longmapsto M, J \longmapsto H, C, P, N$$

before using bottom, top, meet, or join data. The handoff supplies a bounded meet–join boundary for finite open and certified-family rows, but it does not supply a host complete order, powerset of opens, unrestricted joins, Zorn-style maximal principle, quotient representative, or selected external endpoint.

Proof. Begin with the order and lattice rows. By Definition 6501.2, a consumer first exposes O and L, so the comparison, meet, and join data are the displayed BEDC rows also used by chapter 55.

Next read the endpoint rows. The bottom and top entries are named as $\bot$ and $\top$ inside the packet, and their endpoint laws are checked through the rows M and J rather than through a complete-lattice carrier.

The complete-lattice and locale neighbours provide comparison context only after this finite route is exposed. The certified-family route of chapter 5155 and the point-free route of chapter 5153 may consume the bounded lattice boundary, but they cannot replace it by host powersets or arbitrary joins.

The support rows H, C, P, N keep the same bounded-lattice data under transport, replay, provenance, and naming. Because no row stores host complete orders, Zorn principles, quotient representatives, unrestricted joins, or selected endpoints, the handoff remains a finite NameCert boundary. □

Falsifiable prediction: Removing the $\mathsf{Poset}^{\uparrow}$ order row, $\mathsf{Lattice}^{\uparrow}$ meet–join row, bottom row, top row, endpoint meet exactness, endpoint join exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row prevents a locale, complete-lattice, or frame-style consumer from verifying the bounded-lattice route.

Independence witness: poset, lattice, complete-lattice, locale-space: $\mathsf{BoundedLattice}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries order, meet–join, displayed bottom, displayed top, endpoint exactness, transport, replay, provenance, and local naming rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedLattice}^{\uparrow}$ packet is generated from a PosetUp order row, a LatticeUp meet–join row, displayed bottom and top rows, endpoint meet and join exactness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6501.1, Definition 6501.2, and Definition 6501.3 seed the finite bounded-lattice boundary over order, meet, join, endpoints, endpoint exactness, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host complete orders, powerset lattices, unrestricted joins, Zorn principles, quotient representatives, selected external endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked order admission, lattice admission, bottom and top exposure, endpoint exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundedLatticeUp.

Chapter 6502

A Concrete Naming Certificate for $\mathsf{RegularCauchyComparisonModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names with explicit moduli, the RegularCauchyComparisonMod packet records the finite comparison-modulus data by which two regular rational names can be compared before equality, convergence-modulus, or completion-facing consumers read a terminal real boundary. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the common-tail comparison surface of chapter 387. The packet is a finite L10 comparison-modulus surface for RealEquality and RealModulusOfConvergence consumers; it is not quotient stream equality, a selected limit, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6502.1 (Regular Cauchy comparison modulus carrier)**.** *A* $\mathsf{RegularCauchyComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S_0, S_1, R_0, R_1, D, \mu, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened for the compared regular Cauchy names.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording requested radii, slack splits, and comparison budgets.*
4. μ *is the finite comparison-modulus row assigning each displayed tolerance request in* D *a common tail stage.*
5. W *is the shared window row obtained by reading both regular names beyond the stage recorded by* μ*.*
6. Q *is the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row that checks the two readbacks through the shared window* W *and the dyadic ledger* D*.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after* $S_0, S_1, R_0, R_1, D, \mu, W, Q$ *have been displayed.*

8. *H records componentwise* hsame *transport for the two window rows, two readback rows, dyadic ledger, comparison modulus, shared window, tail-comparison row, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D, \mu, W \longmapsto Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyComparisonModulus$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, CauchyTailComparison$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyComparisonModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for quotient classes of streams, selected representatives, trichotomy, countable-choice schedules, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonModulusUp.RegularCauchyComparisonModulusTasteGate`
`single carrier alignment`

Theorem 6502.2 (Regular Cauchy comparison modulus NameCert obligations)**.** *For every accepted* RegularCauchyComparisonModulus$^\uparrow$ *carrier $M = (S_0, S_1, R_0, R_1, D, \mu, W, Q, E, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyComparisonModulus}^\uparrow}$ *surface consists exactly of carrier admission, two* StreamName$^\uparrow$ *window exposures, two* RegSeqRat$^\uparrow$ *readbacks, one* DyadicRatCore$^\uparrow$ *tolerance ledger, one finite comparison-modulus row, one shared tail-window row, one* CauchyTailComparison$^\uparrow$ *check, ordered* Real$^\uparrow$ *seal access, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the accepted packet. The carrier exposes exactly two finite stream windows, two regular-rational readbacks, the dyadic tolerance ledger, the comparison-modulus row, the shared tail window, the tail-comparison check, the terminal Real seal, and the structural rows H, C, P, N.

The comparison-modulus row is read after the dyadic request is displayed. Thus each common tail stage is attached to an explicit radius and slack budget rather than to an ambient convergence assertion.

The shared window W and comparison row Q then compare the two regular readbacks through the same visible tail data. This keeps the comparison inside CauchyTailComparison$^\uparrow$ and prevents a quotient or host-equality route from entering.

Finally the Real seal E and the structural rows transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded principles, the listed rows exhaust the local NameCert surface. □

Theorem 6502.3 (Regular Cauchy comparison modulus RegSeqRat handoff)**.** *Every* RegSeqRat$^\uparrow$, RealEquality$^\uparrow$, *or* RealModulusOfConvergence$^\uparrow$ *consumer of an accepted* RegularCauchyComparisonModulus$^\uparrow$ *packet must read the two* StreamName$^\uparrow$ *window rows, two* RegSeqRat$^\uparrow$ *readback rows, the* DyadicRatCore$^\uparrow$ *tolerance ledger, the finite comparison-modulus row, the shared tail-window row, and the* CauchyTailComparison$^\uparrow$ *check before it may use the terminal* Real$^\uparrow$ *seal. The handoff supplies a finite comparison-modulus route for regular Cauchy names; it does not supply quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient* Real$^\uparrow$ *completeness.*

Proof. Start with the carrier projection of Definition 6502.1. A consumer first receives S_0, S_1 and then R_0, R_1, so the comparison begins from displayed finite windows and regular-rational readbacks.

Next read the dyadic ledger D and the comparison-modulus row μ. These rows bind each tolerance request to a finite common tail stage before any Real-facing boundary is visible.

Use the shared window W and the tail-comparison row Q. They compare the two readbacks through the same finite tail data and the same dyadic budget, matching the common-tail discipline of $\mathsf{CauchyTailComparison}^{\uparrow}$.

Only after this route is displayed may the terminal seal E be consumed by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, or $\mathsf{RealModulusOfConvergence}^{\uparrow}$ readers. The structural rows H, C, P, N preserve that route and contain no quotient, selection, choice, order-decision, host-equality, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6502.1, Definition 6502.2, and Definition 6502.3 seed the finite comparison-modulus route over displayed dyadic tolerances, regular readbacks, common tail windows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, public closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired window exposure, paired RegSeqRat readback, dyadic tolerance accounting, comparison-modulus exactness, shared tail-window exactness, CauchyTailComparison handoff, ordered Real seal access, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyComparisonModulusUp.

Constructive story: A $\mathsf{RegularCauchyComparisonModulus}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a finite comparison-modulus row, a shared tail window, a CauchyTailComparison row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6503

A Concrete Naming Certificate for $\mathsf{PolarizationIdentity}^{\uparrow}$

Following Jordan and von Neumann, the polarization identity is read as a finite reconstruction route from inner-product rows to norm/parallelogram rows, not as host Hilbert-space equality. The $\mathsf{PolarizationIdentity}^{\uparrow}$ packet sits between the scalar field of chapter 45, the vector-space source of chapter 47, the inner-product surface of chapter 86, the norm-square boundary of chapter 89, and the Hilbert consumer boundary of chapter 91.

Definition 6503.1 (PolarizationIdentity carrier)**.** *A* $\mathsf{PolarizationIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (F, V, I, N, Q, H, C, R, L)$$

with the following displayed rows:

1. *F is the* $\mathsf{Field}^{\uparrow}$ *scalar row that supplies the additive, multiplicative, sign, and division data used by the scalar reconstruction coefficients.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row whose vector addition, inverse, zero, and scalar-action rows are the only vector endpoints admitted by the packet.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *scalar-form row naming the accepted inner-product observations on the displayed vector endpoints.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *norm-square row read from the diagonal inner-product observations and the parallelogram ledger.*
5. *Q is the* $\mathsf{Hilbert}^{\uparrow}$ *consumer boundary row; it receives only the reconstructed norm/parallelogram route and not a host Hilbert-space equality.*
6. *H records componentwise* hsame *transport for scalar data, vector data, inner-product observations, norm-square rows, the Hilbert boundary, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from the inner-product rows through the norm-square and parallelogram rows to the Hilbert-facing boundary.*
8. *R is the* Pkg *provenance over* $\mathsf{Field}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{PolarizationIdentity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, I, N, Q, H, C, R, L$. It has no coordinate for a completed Hilbert space, quotient vector equality, external bilinear map, selected orthonormal basis, spectral theorem, or host parallelogram proof.

Lean4 ✓ `BEDC.Derived.PolarizationIdentityUp.PolarizationIdentityTasteGate single carrier alignment`

Theorem 6503.2 (PolarizationIdentity NameCert obligations)**.** *For every accepted* $\mathsf{PolarizationIdentity}^{\uparrow}$ *carrier* $P = (F, V, I, N, Q, H, C, R, L)$*, the local* $\mathsf{NameCert}_{\mathsf{PolarizationIdentity}^{\uparrow}}$ *surface consists of carrier admission for scalar field data, vector-space endpoints, inner-product scalar-form observations, norm-square and parallelogram rows, Hilbert consumer boundary, componentwise transport, displayed continuation replay, package provenance, and local naming. Every public read of the polarization route is generated by these rows, and no read exports host Hilbert equality, quotient vector equality, a selected basis, an external bilinear map, or ambient completeness.*

Proof. Read the carrier rows in the displayed order. The scalar row F and vector-space row V provide the only scalar coefficients and vector endpoints available to the packet.

The inner-product row I supplies the scalar-form observations, and the norm-square row N records the diagonal and parallelogram-facing reads obtained from those observations. Thus the reconstruction route is finite data over the displayed rows rather than an equality inside a host Hilbert space.

Finally Q, H, C, R, L carry the Hilbert-facing boundary, componentwise transport, replay, provenance, and local naming. Since no coordinate stores a completed Hilbert space, quotient vector equality, selected basis, external bilinear map, or ambient completeness, those objects cannot be exported by the local certificate. □

Theorem 6503.3 (PolarizationIdentity inner-product norm handoff)**.** *Every* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *consumer of an accepted* $\mathsf{PolarizationIdentity}^{\uparrow}$ *packet reads the inner-product-to-norm reconstruction only through*

$$\mathsf{Field}^{\uparrow},\quad \mathsf{VecSpace}^{\uparrow},\quad \mathsf{InnerProduct}^{\uparrow},\quad \mathsf{Norm}^{\uparrow},\quad \mathsf{Hilbert}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The handoff sends diagonal scalar-form observations and parallelogram rows to the norm-square boundary, then presents the resulting finite route to Hilbert-facing consumers. It does not supply a host Hilbert equality, quotient vector equality, selected orthonormal basis, spectral theorem, or external completeness principle.

Proof. Begin with Definition 6503.1. The only route from inner-product observations to norm-square data is the visible I, N pair over the scalar and vector rows F, V.

Apply Definition 6503.2. It requires the Hilbert-facing read to pass through carrier admission, inner-product observations, norm-square and parallelogram rows, transport, replay, provenance, and local naming.

Therefore a downstream $\mathsf{Norm}^{\uparrow}$ or $\mathsf{Hilbert}^{\uparrow}$ consumer receives exactly the finite handoff $F, V, I, N, Q, H, C, R, L$. The refused host equality, quotient, basis, spectral, and completeness objects are absent from the carrier and cannot be reconstructed by replay. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolarizationIdentity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolarizationIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6503.1, Definition 6503.2, and Definition 6503.3 seed the finite reconstruction route from inner-product rows to norm-square, parallelogram, and Hilbert-facing consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolarizationIdentity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PolarizationIdentityUp.PolarizationIdentityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host Hilbert equality, quotient vector equality, selected bases, spectral theory, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, scalar coefficient discipline, vector endpoint admission, inner-product observation, norm-square/parallelogram handoff, Hilbert boundary, transport, replay, provenance, and local naming for BEDC.Derived.PolarizationIdentityUp.

Constructive story: A PolarizationIdentity$^{\uparrow}$ packet is generated from FieldUp scalar data, VecSpaceUp vector data, InnerProductUp scalar-form rows, NormUp norm-square rows, a HilbertUp consumer boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6504

A Concrete Naming Certificate for $\mathsf{ModulusOfSequentialContinuity}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ModulusOfSequentialContinuity}^{\uparrow}$ packet records the explicit tolerance-to-window datum that makes a sequential-continuity read constructive. It sits beside the $\mathsf{SequentialContinuity}^{\uparrow}$ route of chapter 1466, reads the pointwise surface of chapter 59, uses the metric rows of chapter 57, inspects finite $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, hands those windows through $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, and reaches only the displayed $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet exports a finite modulus object, not Real completeness, countable choice, quotient streams, host function extensionality, or an ambient continuity principle.

Definition 6504.1 (ModulusOfSequentialContinuity carrier)**.** *A* $\mathsf{ModulusOfSequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, F, S, T, E, W_X, W_Y, R_X, R_Y, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source and target* $\mathsf{Met}^{\uparrow}$ *carrier rows, including the distance classifiers and visible tolerance ledgers.*
2. *F is the displayed* $\mathsf{Continuous}^{\uparrow}$ *row whose pointwise modulus may be read only at inspected source windows.*
3. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *surface, and T is the finite threshold ledger requested by the sequential-continuity consumer.*
4. *E is the tolerance-to-window assignment: each displayed target tolerance in T is sent to a finite source-window request before any target conclusion is read.*
5. *W_X and W_Y are the source and target finite window families exposed by S, F, E.*
6. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the source and target windows.*
7. *A is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *acceptance row saying that the displayed windows are the only sequence-facing route consumed by the packet.*
8. *H stores componentwise* hsame *transport for the two metric rows, continuous row, source stream, threshold ledger, assignment, window rows, regular readbacks, sequential acceptance, continuation, provenance, and local naming.*

9. *C stores displayed* Cont *replay from T to E, from E, S to W_X, from W_X, F to W_Y, and from W_X, W_Y to R_X, R_Y, A.*
10. *P is the* Pkg *provenance over* ModulusOfSequentialContinuity$^\uparrow$, SequentialContinuity$^\uparrow$, Continuous$^\uparrow$, Met$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ModulusOfSequentialContinuity}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, Y, F, S, T, E, W_X, W_Y, R_X, R_Y, A, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable choice, host function extensionality, detached continuity proof, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ModulusOfSequentialContinuityUp.ModulusOfSequentialContinuityTasteGate`
`single carrier alignment`

Theorem 6504.2 (ModulusOfSequentialContinuity NameCert obligations)**.** *For every accepted* ModulusOfSequentialContinuity$^\uparrow$ *carrier*

$$M = (X, Y, F, S, T, E, W_X, W_Y, R_X, R_Y, A, H, C, P, N),$$

the local NameCert$_{\text{ModulusOfSequentialContinuity}^\uparrow}$ *surface consists exactly of carrier admission for the displayed fifteen-row packet, metric-row stability through X, Y, H, C, pointwise-continuity use through F, tolerance-to-window exposure through T, E, W_X, W_Y, regular-rational readback through R_X, R_Y, sequential-continuity handoff through A, package provenance through P, local naming through N, and non-escape from selected limits, quotient streams, countable choice, host function extensionality, detached continuity, and Real completeness.*

Proof. Project the carrier of Definition 6504.1. The classifier can read only the displayed metric rows, continuous row, stream row, tolerance ledger, assignment row, window rows, regular readbacks, sequential acceptance row, and structural rows.

The analytic route starts with the tolerance ledger. The row T lists the target tolerances requested by the consumer, and E turns each listed tolerance into a finite source-window request before a target window is inspected.

Next read the window and regular-rational handoff. The row W_X is opened from S, E, the row W_Y is obtained from W_X, F, and the rows R_X, R_Y are the only regular-rational readbacks attached to those windows.

Finally attach the sequential-continuity row and the structural rows. The row A records that the displayed windows are the complete sequence-facing route, while H, C, P, N transport, replay, source, and name that route. Since none of the excluded principles is a coordinate of M, the listed obligations exhaust the local NameCert surface. □

Theorem 6504.3 (ModulusOfSequentialContinuity tolerance-window handoff)**.** *Every accepted* ModulusOfSequentialContinuity$^\uparrow$ *packet sends a target tolerance to a finite source-window request before any* SequentialContinuity$^\uparrow$ *or* Real$^\uparrow$ *consumer can read the image sequence. The complete handoff factors through*

$$T \longmapsto E \longmapsto W_X \longmapsto W_Y \longmapsto R_X, R_Y \longmapsto A$$

together with the displayed rows X, Y, F, S, H, C, P, N.

Proof. Begin with the tolerance request. The target tolerance is visible only as an entry of T, and the assignment row E is the only coordinate that converts such an entry into a finite source-window request.

Open the source and image windows. The source row S supplies W_X at the window requested by E, and the continuous row F is consumed only through that inspected source window to expose W_Y.

Pass to the rational and sequential surfaces. The rows R_X and R_Y read the two windows as regular rational data, and A is the downstream $\mathsf{SequentialContinuity}^{\uparrow}$ acceptance row reached from those readbacks.

Thus a consumer that tries to read an image sequence, a Real-facing seal, or a sequential-continuity conclusion without first exposing T, E, W_X, W_Y, R_X, R_Y leaves the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfSequentialContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfSequentialContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6504.1, Definition 6504.2, and Definition 6504.3 seed the finite tolerance-to-window route for sequential-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfSequentialContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, countable choice, quotient streams, host function extensionality, selected limits, detached continuity, Lean verification, or standard bridge checking.

Upgrade path: Formal closure requires checked carrier admission, metric-row stability, tolerance-to-window exactness, finite-window replay, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{SequentialContinuity}^{\uparrow}$ handoff, and non-escape for BEDC.Derived.ModulusOfSequentialContinuity

Constructive story: A $\mathsf{ModulusOfSequentialContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a displayed pointwise-continuity row, a source stream, a finite tolerance ledger, a tolerance-to-window assignment, source and image windows, regular rational readbacks, sequential-continuity acceptance, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6505

A Concrete Naming Certificate for $\mathsf{RouthHurwitzCriterion}^{\uparrow}$

Following Routh and Hurwitz, the $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ packet records the finite stability test for a displayed real polynomial through coefficient rows, Hurwitz matrix rows, determinant rows, and ordered-sign rows. It sits after the polynomial carrier of chapter 50, the matrix carrier of chapter 49, the real algebra and order boundary of chapter 766, the determinant packet of chapter 84, and the scalar field source of chapter 45. The packet names a finite algebraic stability certificate for downstream control and ODE consumers; it does not import complex-root choice, a host spectral theorem, quotient polynomial equality, selected roots, or an ambient analytic flow theorem.

Definition 6505.1 (Routh–Hurwitz criterion carrier)**.** *A* $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, M, D, S, T, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Poly}^{\uparrow}$ *coefficient row, listing the finite real coefficient sequence, leading-coefficient witness, degree window, and normalization request consumed by the stability test.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *Hurwitz row, displaying the finite Hurwitz matrix entries read from A.*
3. *D is the* $\mathsf{Determinant}^{\uparrow}$ *minor row, listing the finite leading principal determinant requests and their replay paths through M.*
4. *S is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *sign row, recording the ordered-real reads that classify each determinant or boundary minor as positive, zero, or obstructed for the displayed request.*
5. *T records componentwise* hsame *transport for coefficient, matrix, determinant, sign, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the finite route*

$$A \longmapsto M \longmapsto D \longmapsto S.$$

7. *P is the* Pkg *provenance over* $\mathsf{RouthHurwitzCriterion}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RouthHurwitzCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through A, M, D, S, T, C, P, N. It has no coordinate for complex roots, spectral decompositions, quotient polynomial equality, selected eigenvalues, or analytic trajectories.

Theorem 6505.2 (Routh–Hurwitz NameCert obligations)**.** *For every accepted* $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ *carrier $R = (A, M, D, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RouthHurwitzCriterion}^{\uparrow}}$ *surface consists exactly of coefficient admission through A, Hurwitz matrix assembly through M, leading-principal determinant and minor exposure through D, ordered-sign classification through S, componentwise transport through T, displayed replay through C, provenance through P, and local naming through N. Every downstream control, ODE, or algebra consumer must expose these rows before reading a Routh–Hurwitz stability conclusion.*

Proof. Project the carrier of Definition 6505.1. The coefficient row A is the only polynomial input, so the test begins from finite displayed coefficients rather than quotient polynomial equality or selected roots.

Assemble the finite Hurwitz matrix through M. Each entry is read from the coefficient row, and the matrix row exposes only the finite table needed for the displayed degree window.

Read the determinant and sign rows next. The row D supplies the leading-principal determinant and boundary minor requests through $\mathsf{Determinant}^{\uparrow}$, while S supplies the $\mathsf{RealAlgOrder}^{\uparrow}$ sign reads used by the criterion.

Finally transport and replay the same route through T, C, P, N. Since no carrier coordinate contains complex roots, a spectral theorem, selected eigenvalues, quotient polynomial equality, or analytic trajectories, the listed rows exhaust the NameCert obligations. □

Theorem 6505.3 (Routh–Hurwitz polynomial handoff)**.** *Every accepted* $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ *packet exports the finite polynomial handoff*

$$A \longmapsto M \longmapsto D \longmapsto S \longmapsto T, C, P, N$$

from displayed coefficient rows to stability-facing sign rows. A consumer may cite the handoff only after the $\mathsf{Poly}^{\uparrow}$ *coefficient row,* $\mathsf{Mat}^{\uparrow}$ *Hurwitz row,* $\mathsf{Determinant}^{\uparrow}$ *minor row,* $\mathsf{RealAlgOrder}^{\uparrow}$ *sign row,* $\mathsf{Field}^{\uparrow}$ *scalar provenance, transport, replay, provenance, and local name have been exposed.*

Proof. Start with a stability-facing read of the packet. By Definition 6505.2, the read first exposes the coefficient row A and cannot replace it by an ambient polynomial quotient.

The Hurwitz matrix row M is then assembled from those coefficients, and the determinant row D reads only the leading principal minors and boundary minors displayed by that matrix. This keeps the computation finite and rowwise.

The sign row S supplies the ordered-real classification through $\mathsf{RealAlgOrder}^{\uparrow}$, while $\mathsf{Field}^{\uparrow}$ appears only as scalar provenance for the displayed operations. Transport, replay, provenance, and local naming then carry the same rows to the downstream consumer.

Thus the exported handoff is the finite polynomial route stated above. It is not a complex-root factorization, host spectral theorem, selected-eigenvalue argument, quotient polynomial equality, or analytic-flow assertion. □

Theorem 6505.4 (Routh–Hurwitz stability boundary)**.** *Any stability conclusion read from an accepted* $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ *packet is bounded by the displayed sign pattern of the determinant and minor rows D, S over the coefficient and Hurwitz-matrix rows A, M. If a determinant sign row is absent, zero, or obstructed, the packet exports only that boundary status and not a strict stability certificate. No consumer may fill the gap with complex-root choice, a host spectral theorem, selected roots, quotient polynomial equality, or an ambient ODE stability theorem.*

Proof. Apply the polynomial handoff theorem Definition 6505.3. It routes every stability-facing read through A, M, D, S, T, C, P, N before any conclusion is consumed.

Inspect the ordered-sign row S. Positive determinant rows provide the displayed strict side of the criterion, while absent, zero, or obstructed sign rows remain visible as boundary data rather than hidden obligations.

Use the NameCert obligations of Definition 6505.2. They keep coefficient admission, matrix assembly, determinant exposure, sign classification, transport, replay, provenance, and local naming attached to the same finite packet.

Because the carrier has no root-choice, spectral, quotient-polynomial, selected-root, or analytic-flow coordinate, a consumer cannot strengthen an obstructed sign boundary into a strict stability certificate by importing one of those host endpoints. □

Closure status for $\mathsf{NameCert}_{\mathsf{RouthHurwitzCriterion}\uparrow}$.

Constructive story: A $\mathsf{RouthHurwitzCriterion}^{\uparrow}$ packet is generated from finite polynomial coefficient rows, Hurwitz matrix rows, determinant and minor rows, RealAlgOrder sign rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RouthHurwitzCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6505.1, Definition 6505.2, Definition 6505.3, and Definition 6505.4 seed the finite algebraic stability certificate route from polynomial coefficients through Hurwitz matrices, determinants, ordered signs, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RouthHurwitzCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close complex-root choice, host spectral theorems, selected eigenvalues, quotient polynomial equality, analytic ODE stability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coefficient admission, Hurwitz matrix assembly, determinant exposure, ordered-sign classification, boundary refusal, transport, replay, provenance, and local naming for BEDC.Derived.RouthHurwitzCriterionUp

Chapter 6506

A Concrete Naming Certificate for $\mathsf{EulerCharacteristic}^{\uparrow}$

Following Euler and Poincare, the $\mathsf{EulerCharacteristic}^{\uparrow}$ packet records the finite alternating-simplex ledger by which a finite simplicial complex receives its Euler characteristic. It sits over the finite face inventory of chapter 238, the finite chain rows of chapter 3414, the homology provenance of chapter 98, the natural-number row of chapter 28, and the integer row of chapter 31. The packet names only a finite count-and-transport certificate; it does not import quotient chain complexes, selected triangulation equivalence, host cardinality, or infinite CW-complex semantics.

Definition 6506.1 (Euler characteristic finite-complex carrier)**.** *An* $\mathsf{EulerCharacteristic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, S, Q, A, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the source* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, including the finite simplex inventory, face relation, and dimension grading.*
2. *S is the simplex-count ledger. For each displayed dimension n, it records the* $\mathsf{Nat}^{\uparrow}$ *count of n-simplices admitted by K.*
3. *Q is the finite-support row stating that only finitely many entries of S are nonzero, so the alternating sum is a finite ledger.*
4. *A is the* $\mathsf{Int}^{\uparrow}$ *alternating-sum row, recording the signed total $\sum_n (-1)^n S_n$ by finite integer addition.*
5. *B is the subdivision ledger boundary row. It records the finite count changes produced by a displayed elementary subdivision and the matching cancellation rows for faces inserted on both sides of the alternating sum.*
6. *H records componentwise* hsame *transport for the simplicial complex row, simplex-count rows, finite-support row, integer sum row, subdivision ledger, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a request for the Euler characteristic through K, S, Q, A, B, H.*
8. *P is the* Pkg *provenance over* $\mathsf{EulerCharacteristic}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteSimplicialChain}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerCharacteristic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, Q, A, B, H, C, P, N$. It has no coordinate for host cardinality, quotient chain-complex equality, an unrestricted homotopy-invariance theorem, selected triangulation equivalence, or an infinite CW-complex.

Theorem 6506.2 (Euler characteristic subdivision ledger boundary)**.** *For every accepted* $\mathsf{EulerCharacteristic}^{\uparrow}$ *carrier $E = (K, S, Q, A, B, H, C, P, N)$, a displayed elementary subdivision of the finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *row may be consumed by an Euler-characteristic read only through the subdivision ledger boundary B. The row B pairs the inserted simplex-count changes in adjacent dimensions, transports them through H, replays the same finite route through C, and leaves the integer alternating-sum row A with the same* $\mathsf{Int}^{\uparrow}$ *value. Consequently every* $\mathsf{Homology}^{\uparrow}$ *or finite-chain consumer that reads the Euler characteristic receives a finite count ledger, not a quotient homology computation, host cardinality argument, or infinite complex invariant.*

Proof. Project the carrier of Definition 6506.1. The simplex source K supplies a finite dimension-graded inventory, while S records the $\mathsf{Nat}^{\uparrow}$ count in each displayed dimension and Q states that the alternating sum is finite.

Read the integer row A. It is built from the finite list S by alternating signs and $\mathsf{Int}^{\uparrow}$ addition, so a consumer can inspect only the displayed ledger

$$\sum_n (-1)^n S_n.$$

Now apply the subdivision row B. An elementary subdivision changes the finite simplex counts only by the inserted rows recorded in B, and those inserted rows are paired across adjacent dimensions with opposite signs in A.

Transport and replay preserve that pairing. The hsame row H carries the same simplex-count and cancellation rows to classified replacements, and C replays the Euler-characteristic request through K, S, Q, A, B in the same order.

Finally read provenance. The $\mathsf{FiniteSimplicialChain}^{\uparrow}$ and $\mathsf{Homology}^{\uparrow}$ entries in P identify the downstream consumers of the finite ledger, but they do not add quotient chain groups, selected homotopy data, host cardinality, or infinite CW-complex rows. Thus the subdivision boundary is a local finite ledger statement about A, not an ambient invariance theorem. □

Falsifiable prediction: Every accepted $\mathsf{EulerCharacteristic}^{\uparrow}$ read must expose the finite SimplicialComplex row, simplex-count ledger, finite-support row, integer alternating-sum row, subdivision ledger boundary, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: simplicialcomplex, finite-simplicial-chain, homology, nat, int: $\mathsf{EulerCharacteristic}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite simplex counts, finite-support evidence, an integer alternating-sum ledger, subdivision cancellation rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerCharacteristic}^{\uparrow}}$.

Constructive story: An $\mathsf{EulerCharacteristic}^{\uparrow}$ packet is generated from a finite SimplicialComplex row, Nat simplex-count rows, a finite-support ledger, an Int alternating-sum row, subdivision cancellation rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerCharacteristic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6506.1 and Definition 6506.2 seed the finite Euler-characteristic count ledger and its subdivision-boundary cancellation route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerCharacteristic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full homotopy invariance, quotient chain complexes, host cardinality, infinite CW-complex invariance, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked finite simplex counting, integer alternating-sum exactness, subdivision cancellation, Homology-facing provenance, replay stability, transport exactness, and local naming for BEDC.Derived.EulerCharacteristicUp

Chapter 6507

A Concrete Naming Certificate for $\mathsf{VietorisHyperspace}^{\uparrow}$

Following Vietoris, Hausdorff, and Kuratowski, the $\mathsf{VietorisHyperspace}^{\uparrow}$ packet records the compact-metric hyperspace surface whose compact-subset names are read simultaneously through the hyperspace carrier of chapter 1660, the hit-and-miss topology rows of chapter 2375, the scalar finite-distance rows of chapter 3653, the compact metric source of chapter 129, and the completion-facing hyperspace boundary of chapter 5944. The packet names the shared finite NameCert carrier for Vietoris and Hausdorff consumers; it is not a host powerset of compact subsets, not quotient equality of closed sets, not a selected nearest-point principle, and not open-cover compactness.

Definition 6507.1 (Vietoris hyperspace carrier)**.** *A* $\mathsf{VietorisHyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (K, H, T, D, C, F, S, R, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the point carrier, metric distance rows, finite-net schedules, and completion witnesses used by compact-subset requests.*
2. *H is the* $\mathsf{Hyperspace}^{\uparrow}$ *compact-subset row, naming the nonempty compact-subset presentations over K.*
3. *T is the* $\mathsf{VietorisTopology}^{\uparrow}$ *row, carrying the lower-hit and upper-miss generators attached to the compact-subset presentations.*
4. *D is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *row, recording finite scalar distance reads between displayed compact-subset names.*
5. *C is the* $\mathsf{KuratowskiHyperspace}^{\uparrow}$ *row, keeping completion-facing hyperspace consumers inside compact finite-net and Hausdorff-distance data.*
6. *F is the finite compatibility ledger tying Vietoris hit-and-miss reads to the Hausdorff finite-distance and compact-metric net schedules.*
7. *S records componentwise* hsame *transport for $K, H, T, D, C, F, R, P, N$.*
8. *R records displayed* Cont *replay of the route from compact metric source through hyperspace, Vietoris topology, Hausdorff distance, Kuratowski boundary, and finite compatibility rows.*
9. *P is the* Pkg *provenance over* $\mathsf{VietorisHyperspace}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{KuratowskiHyperspace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{VietorisHyperspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, H, T, D, C, F, S, R, P, N$. It has no coordinate for host powersets, quotient compacta, selected nearest points, ambient equality of closed subsets, open-cover compactness, or a hidden completion representative.

Theorem 6507.2 (Vietoris hyperspace NameCert obligations)**.** *For every accepted* $\mathsf{VietorisHyperspace}^{\uparrow}$ *carrier*

$$V = (K, H, T, D, C, F, S, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VietorisHyperspace}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, hyperspace compact-subset row, Vietoris topology row, Hausdorff distance row, Kuratowski hyperspace row, finite compatibility ledger, componentwise transport, displayed replay, provenance, and local naming rows.*

The obligation surface requires compact-metric source visibility through K, compact-subset admission through H, lower-hit and upper-miss topology reads through T, finite scalar distance reads through D, completion-facing boundary control through C, Vietoris-Hausdorff compatibility through F, and structural transport, replay, provenance, and naming through S, R, P, N.

Proof. Project the carrier of Definition 6507.1. The analytic rows are K, H, T, D, C, F, and the structural rows are S, R, P, N.

The source row K is read first because each compact-subset, topology, distance, and completion-facing row is over the same compact metric carrier. The hyperspace row H then supplies the compact-subset presentations consumed by the Vietoris and Hausdorff sides.

Read T, D, C through the finite compatibility ledger F. The Vietoris row contributes only lower-hit and upper-miss generators, the Hausdorff distance row contributes only finite scalar distance reads, and the Kuratowski row contributes only completion-facing hyperspace boundary data.

Finally S, R, P, N transport, replay, source, and name the same finite route. Since the excluded host powerset, quotient, nearest-point, open-cover, ambient-equality, and hidden-representative objects do not occur in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6507.3 (Vietoris hyperspace compact-metric handoff)**.** *Every compact-metric, Vietoris-facing, Hausdorff-facing, or Kuratowski-facing consumer of an accepted* $\mathsf{VietorisHyperspace}^{\uparrow}$ *carrier must factor through the displayed route*

$$K \longmapsto H \longmapsto T, D, C \longmapsto F \longmapsto S, R, P, N.$$

Thus the consumer receives compact metric source rows, compact-subset hyperspace rows, Vietoris hit-and-miss rows, finite Hausdorff-distance rows, Kuratowski completion-facing rows, finite compatibility, transport, replay, provenance, and local naming. It receives no host powerset, quotient compactum, selected nearest point, open-cover compactness theorem, ambient equality of closed subsets, or hidden completion representative.

Proof. Use Definition 6507.2. It requires K before any compact-subset or hyperspace read, and it requires H before the Vietoris, Hausdorff, or Kuratowski rows can be exposed.

The three consumer directions then share the same finite source. The row T reads lower-hit and upper-miss data from chapter 2375; the row D reads finite scalar distance data from chapter 3653; and the row C reads the completion-facing boundary of chapter 5944.

The compatibility ledger F ties these rows back to the compact metric finite-net schedules in K and the compact-subset presentations in H. The structural rows S, R, P, N preserve, replay, source, and name that route for downstream consumers.

No excluded host or quotient object is a coordinate of V. Hence the public handoff is exactly the displayed compact-metric hyperspace route. □

Closure status for $\mathsf{NameCert}_{\mathsf{VietorisHyperspace}^{\uparrow}}$.

Constructive story: A $\mathsf{VietorisHyperspace}^{\uparrow}$ packet is generated from CompactMetric rows, Hyperspace compact-subset rows, VietorisTopology hit-and-miss rows, HausdorffDistance finite scalar rows, KuratowskiHyperspace completion-facing rows, finite compatibility, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VietorisHyperspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6507.1, Definition 6507.2, and Definition 6507.3 seed the compact-metric hyperspace certificate linking Hyperspace, VietorisTopology, HausdorffDistance, CompactMetric, and KuratowskiHyperspace rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VietorisHyperspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host powersets, quotient compacta, selected nearest points, open-cover compactness, ambient equality of closed subsets, or hidden completion representatives.

Upgrade path: Obligation closure requires checked carrier admission, Vietoris-Hausdorff compatibility, compact-metric source exactness, Kuratowski boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.VietorisHyperspac

Chapter 6508

A Concrete Naming Certificate for $\mathsf{EventInstance}^{\uparrow}$

The $\mathsf{EventInstance}^{\uparrow}$ packet realizes the inscription-event carrier of chapter 9397 as a finite public-transition certificate. It names an event as a visible move from a prior observation state to an accepted record, together with route, check, consumer, GAP-ledger, transport, replay, provenance, and local NameCert rows. The packet is a BEDC-native concrete instance: it does not assert an external metaphysics of agency, a host psychology state, a hidden generator, or a global free-will predicate.

Definition 6508.1 (Event instance carrier)**.** *An* $\mathsf{EventInstance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, A, R, Q, G, T, C, U, P, N)$$

with the following displayed rows:

1. *S is the source-state row, recording the prior finite observation history and the visible obligation or socket before the event is accepted.*
2. *A is the accepted-record row, naming the public record entered at the transition.*
3. *R is the route row, exposing the finite generator-closure route, refusal route, or mixed route by which the accepted record is audited.*
4. *Q is the check row, recording the visible acceptance, validation, refusal, or bounded review state attached to A and R.*
5. *G is the GAP-ledger row, preserving the non-generated residue, compression residue, consumer accountability, and non-escape boundary of the event.*
6. *T records componentwise* hsame *transport for the source, accepted record, route, check, GAP-ledger, continuation, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *continuation from S to the accepted event surface and from that surface to downstream consumers.*
8. *U is the consumer row, naming the proof obligation, manuscript claim, audit route, or public-object consumer that may read the event.*
9. *P is the* Pkg *provenance over* $\mathsf{EventInstance}^{\uparrow}$*, chapter 9397,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{EventInstance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, R, Q, G, T, C, U, P, N$. It has no coordinate for private intention, host-level causal completion, unbounded psychological equality, hidden generator authority, or a global agency predicate.

Theorem 6508.2 (Event instance NameCert obligations)**.** *For every accepted* $\mathsf{EventInstance}^{\uparrow}$ *carrier*

$$E = (S, A, R, Q, G, T, C, U, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EventInstance}^{\uparrow}}$ *surface consists exactly of carrier admission for the source-state row, accepted-record row, route row, check row, GAP-ledger row, transport row, continuation row, consumer row, provenance row, and local naming row.*

The obligation surface requires source visibility through S, public transition naming through A, route audit through R, acceptance or refusal state through Q, GAP-ledger nonescape through G, transport and continuation stability through T, C, consumer accountability through U, source provenance through P, and local naming through N.

Proof. Project the carrier of Definition 6508.1. The event-facing rows are S, A, R, Q, G, U, while T, C, P, N are structural rows.

The source and accepted-record rows are read first. Without S there is no prior public observation state; without A there is no named record whose acceptance can be audited.

The route, check, and GAP-ledger rows then determine the event boundary. The route row R displays what is generator-closed, refused, or mixed; the check row Q records the visible acceptance state; and the ledger row G keeps the non-generated and compression residue from becoming hidden authority.

Finally T, C, U, P, N transport, continue, expose the consumer, source the packet, and name the same finite transition. Since no private intention, host causal completion, psychological equality, hidden generator authority, or global agency predicate is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6508.3 (Event instance inscription route)**.** *Every consumer of an accepted* $\mathsf{EventInstance}^{\uparrow}$ *carrier must factor its inscription-event read through the displayed route*

$$S, A \longmapsto R, Q, G \longmapsto T, C, U, P, N.$$

Thus the consumer sees the prior finite observation state, the accepted public record, the route audit, the check state, the GAP-ledger residue, transport, continuation, consumer accountability, provenance, and local naming before it may cite the event as an inscription instance.

Proof. Start with Definition 6508.2. It binds the event to the source row S and accepted-record row A before any downstream consumer can inspect the event.

The inscription-facing part is the triple R, Q, G. The route row separates generator-closed material from refused or residual material, the check row records the visible acceptance state, and the GAP ledger records the non-escape boundary.

The remaining rows do not create a second event. Componentwise transport T, continuation C, consumer row U, provenance P, and local naming N preserve and expose the same finite transition. Therefore every consumer-facing use of the event factors through the displayed inscription route. □

Theorem 6508.4 (Event instance event-flow forward link)**.** *Every accepted* $\mathsf{EventInstance}^{\uparrow}$ *read exposing source-state, accepted-record, route, check, GAP-ledger, transport, continuation, consumer, provenance, and local* NameCert *rows may be consumed by the finite event-flow packet of chapter 6332 only after those rows are displayed. The event-flow consumer therefore receives* $S, A, R, Q, G, T, C, U, P, N$ *as one public transition route, not as a hidden generator trace, private intention row, or host causal completion.*

Proof. Start with Definition 6508.3. It forces every inscription-event read through the source row S, accepted record A, route row R, check row Q, GAP-ledger row G, transport row T, continuation row C, consumer row U, provenance row P, and local naming row N.

The hendeca event-flow packet of chapter 6332 accepts a finite ordered event-flow surface only after its event rows and structural rows have been displayed. Feeding an $\mathsf{EventInstance}^{\uparrow}$ read into that packet therefore requires the displayed event-instance rows to be visible first.

Since Definition 6508.1 has no coordinate for private intention, host causal completion, unbounded psychological equality, hidden generator authority, or a global agency predicate, the forward link exports only the public transition route already named by $S, A, R, Q, G, T, C, U, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{EventInstance}^{\uparrow}$ read must expose a source-state row, accepted-record row, route row, check row, GAP-ledger row, hsame transport, Cont continuation, consumer row, Pkg provenance, and local NameCert row before it can be cited as an inscription-event instance.

Independence witness: inscription-event, inscription-gap-ledger, inscribed-route, inscription-audit-trace: $\mathsf{EventInstance}^{\uparrow}$ is not bijective to any listed sibling because it binds source state, accepted record, route audit, check state, GAP ledger, consumer accountability, transport, continuation, provenance, and local naming in one public transition packet.

The Lean TasteGate expectation for the future ‘BEDC.Derived.EventInstanceUp‘ target is single-carrier alignment: a positive gate must see all ten rows $S, A, R, Q, G, T, C, U, P, N$, reject carrier-only event names, reject check-only replay, and reject any classifier that identifies events by private intention or host causal equality.

Closure status for $\mathsf{NameCert}_{\mathsf{EventInstance}^{\uparrow}}$.

Constructive story: An $\mathsf{EventInstance}^{\uparrow}$ packet is generated from a prior BHist source state, an accepted public record, a route audit, a visible check state, a GAP ledger, hsame transport, Cont continuation, consumer accountability, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EventInstance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6508.1, Definition 6508.2, and Definition 6508.3 seed the finite public-transition certificate for inscription events.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EventInstance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, private intention, host-level causal completion, unbounded psychological equality, hidden generator authority, or a global agency predicate.

Upgrade path: Obligation closure requires checked carrier admission, inscription-route exactness, GAP-ledger nonescape, consumer accountability, transport, continuation, provenance, and local naming for BEDC.Derived.EventInstanceUp.

Chapter 6509

A Concrete Naming Certificate for $\mathsf{AscoliCompactOperator}^{\uparrow}$

Following Arzela and Ascoli, and the compact-operator formulation in standard functional analysis, the $\mathsf{AscoliCompactOperator}^{\uparrow}$ packet records compactness of an operator image as finite-net image data rather than as an ambient host compactness assertion. It sits between the Arzela–Ascoli certificate of chapter 412, the compact-metric surface of chapter 129, the finite epsilon-net row of chapter 1767, the operator-norm chapter chapter 3603, and the bounded-linear-functional boundary of chapter 5790. The chapter names the finite operator-image route consumed by later Hilbert and spectral packets; it does not assert host compactness of a source ball, open-cover compactness, selected convergent subsequences, quotient operator equality, or ambient function-space completeness.

Definition 6509.1 (Ascoli compact-operator carrier)**.** *An* $\mathsf{AscoliCompactOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (T, B, K, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the bounded-linear operator row, read through the operator-norm ledger of chapter 3603 and the bounded-linear source discipline of chapter 5790;*
2. *B is the bounded source-ball row, recording only the displayed radius, source windows, and finite operator-norm bounds available to consumers;*
3. *K is the compact-metric image row, expressed as finite epsilon-net image data through* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *rows;*
4. *E is the equicontinuity or Arzela–Ascoli finite-image ledger, recording the inspected family windows and finite image controls used before any compactness-facing export;*
5. *H records componentwise* hsame *transport for T, B, K, E;*
6. *C records the displayed* Cont *replay route forcing consumers to read the operator row, bounded source ball, finite-net image data, and Arzela–Ascoli ledger in that order;*
7. *P is the* Pkg *provenance over* $\mathsf{AscoliCompactOperator}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BoundedLinearFunctional}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AscoliCompactOperator}^{\uparrow}}$ *row.*

The carrier has no coordinate for host compactness of the source ball, open-cover compactness, selected convergent subsequences, quotient equality of operators, or ambient Banach-space completeness.

Theorem 6509.2 (Ascoli compact-operator NameCert obligations)**.** *For every accepted* $\mathsf{AscoliCompactOperator}^{\uparrow}$ *carrier* $O = (T, B, K, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AscoliCompactOperator}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the bounded-linear operator, bounded source-ball, compact-metric image, finite-image ledger, transport, replay, provenance, and local naming rows;*
2. *bounded-linear source compatibility, requiring every source-ball read to pass through the same operator-norm row* T *and bounded source row* B*;*
3. *finite-net image compactness, requiring compactness-facing output to be witnessed by the displayed finite epsilon-net image row* K *rather than by a host compact subset;*
4. *Arzela–Ascoli consumer handoff, requiring* E *to display the equicontinuity or finite-image ledger consumed by the Arzela–Ascoli route before the operator image is exported;*
5. *transport and replay stability, requiring* H *and* C *to preserve the route* T, B, K, E *under componentwise* hsame *movement;*
6. *ledger refusal, excluding open-cover compactness, host compactness of the source ball, selected convergent subsequences, quotient operator equality, and ambient function-space completeness.*

Proof. Project the carrier of Definition 6509.1. The rows T, B, K, E, H, C, P, N give carrier admission and leave no additional analytic coordinate.

The source compatibility clause reads T and B first. The operator-norm row fixes the bounded-linear access discipline, while the source-ball row fixes the displayed bounded input surface.

The compactness-facing route then reads K and E. The image row is finite epsilon-net data over the compact-metric surface, and the Arzela–Ascoli ledger records the finite-image or equicontinuity handoff consumed before an operator-image export.

Finally H and C transport and replay only the displayed rows. Since the carrier omits host compactness, open-cover compactness, selected subsequences, quotient operator equality, and ambient function-space completeness, those principles cannot enter the local NameCert obligations. □

Theorem 6509.3 (Ascoli compact-operator finite-net image route)**.** *Every compact-operator consumer of an accepted* $\mathsf{AscoliCompactOperator}^{\uparrow}$ *carrier must read the finite epsilon-net image route*

$$T, B \longmapsto K \longmapsto E \longmapsto H, C, P, N$$

before any compactness-facing export of the operator image is available.

Proof. Start with the accepted packet $O = (T, B, K, E, H, C, P, N)$. The rows T and B determine the bounded-linear source side and the bounded input ball available to the consumer.

The only compactness-facing image row is K. It is read through finite epsilon-net data over the compact-metric image surface, so a consumer cannot inspect an operator-image compactness claim before reading that finite-net row.

The row E then supplies the Arzela–Ascoli finite-image or equicontinuity handoff, and H, C, P, N transport, replay, source, and name the same route. No carrier coordinate names host compactness or open-cover compactness, so every compact-operator export factors through the displayed finite-net image route. □

Falsifiable prediction: Every accepted $\mathsf{AscoliCompactOperator}^{\uparrow}$ compactness-facing export factors through bounded-linear operator data, a bounded source-ball row, finite epsilon-net image data, an Arzela–Ascoli finite-image ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row. A read that exports compactness through host compactness of the source ball, open-cover compactness, selected subsequences, quotient operator equality, or ambient completeness refutes this packet.

Independence witness: arzelaascoli, compactmetric, finite-epsilon-net, operator-norm, bounded-linear-functional: no listed sibling is bijective to $\mathsf{AscoliCompactOperator}^{\uparrow}$ because only this packet jointly binds the bounded operator row, bounded source-ball row, finite-net image row, Arzela–Ascoli consumer ledger, transport, replay, provenance, and local NameCert boundary in one compact-operator carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliCompactOperator}^{\uparrow}}$.

Constructive story: An $\mathsf{AscoliCompactOperator}^{\uparrow}$ packet is generated from a bounded-linear operator row, bounded source-ball row, finite epsilon-net image row, Arzela–Ascoli finite-image ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AscoliCompactOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6509.1, Definition 6509.2, and Definition 6509.3 seed the finite-net compact-operator image route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliCompactOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, open-cover compactness, selected convergent subsequences, quotient operator equality, ambient function-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-linear source compatibility, finite-net image compactness, Arzela–Ascoli consumer handoff, transport and replay stability, and ledger refusal for BEDC.Derived.AscoliCompactOperato

Chapter 6510

A Concrete Naming Certificate for $\mathsf{ConsciousObservationRoute}^{\uparrow}$

The $\mathsf{ConsciousObservationRoute}^{\uparrow}$ packet realizes the local Hist-time observation route separated by section 9414.59 as finite NameCert data. It consumes the role boundary of chapter 4067, the self-proxy binding schema of Definition 9414.579, the public scientific reflection surface of Definition 9414.510, the inscription event rows of chapter 519, and the event-instance carrier of chapter 6508. The packet names a displayed observation read; it is not a subject-substance claim, not a phenomenal-proof theorem, not a compiler self-hosting certificate, not an AI text-output certificate, and not an identification with T.

Definition 6510.1 (Conscious observation route carrier)**.** *A* $\mathsf{ConsciousObservationRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, \Theta, S, \rho, \Lambda, F, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the local Hist-time row, recording the focused history segment and its current endpoint as finite* BHist *data;*
2. *Θ is the selector row, recording which displayed path enters the current focus;*
3. *S is the self-proxy row, binding the focused local history and selector path to the current I-proxy read;*
4. *ρ is the latency row, recording the visible delay, update, or stale-read boundary for the observation;*
5. *Λ is the refusal row, recording when an attempted observation read omits local history, selector, self-proxy, latency, or naming data;*
6. *F is the field-faithfulness row, requiring the read to display* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *structure before it can export an I-proxy;*
7. *M is the role-separation row, keeping compiler translation, AI/ML pattern generation, and human-consciousness local self-readback distinct;*
8. *H records componentwise* hsame *transport for $L, \Theta, S, \rho, \Lambda, F, M$;*
9. *C records the displayed* Cont *replay route from local history through selector, self-proxy, latency or refusal, field-faithfulness, and role separation;*

10. *P is the* Pkg *provenance over* ConsciousObservationRoute$^{\uparrow}$, RoleSeparationLedger$^{\uparrow}$, InscriptionEvent$^{\uparrow}$, EventInstance$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* NameCert$_{\text{ConsciousObservationRoute}^{\uparrow}}$ *row.*

The carrier stores no hidden subject-substance, phenomenal-proof primitive, compiler self-hosting identification, AI output fluency certificate, global observer, or detached T *coordinate.*

Theorem 6510.2 (Conscious observation route NameCert obligations)**.** *For every accepted* ConsciousObservationRo *carrier*

$$R = (L, \Theta, S, \rho, \Lambda, F, M, H, C, P, N),$$

the local NameCert$_{\text{ConsciousObservationRoute}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the local Hist-time, selector, self-proxy, latency, refusal, field-faithfulness, role-separation, transport, replay, provenance, and local naming rows;*

2. *local observation exposure, requiring any I-proxy export to display L, Θ, S, ρ before the export is available;*

3. *refusal exactness, requiring Λ to reject reads that omit local history, selector, self-proxy, latency or refusal status,* hsame, Cont, Pkg, *or* NameCert *rows;*

4. *role-separation stability, requiring M to keep compiler preservation, AI/ML fit or generation, and human-consciousness local self-readback as distinct carrier roles;*

5. *transport and replay stability, requiring H and C to preserve the route $L, \Theta, S, \rho, \Lambda, F, M$ under componentwise* hsame *movement;*

6. *ledger non-escape, excluding subject-substance, phenomenal-proof primitives, compiler self-hosting identity, AI output fluency as I-proxy status, global observer rows, and detached* T *coordinates.*

Proof. Carrier admission is the projection of Definition 6510.1. The accepted packet exposes exactly the local Hist-time, selector, self-proxy, latency, refusal, field-faithfulness, role-separation, transport, replay, provenance, and local naming rows.

The observation exposure clause reads L, Θ, S, ρ. These rows display the local history, selected focus route, current self-proxy, and latency boundary before any I-proxy export can be inspected.

Refusal exactness and role separation then read Λ, F, M. A route that omits the displayed field-faithfulness rows is refused, and the role-separation row prevents compiler preservation or AI/ML generation from being retyped as local self-readback.

Finally H and C transport and replay only the displayed route. Since no hidden subject-substance, phenomenal-proof primitive, compiler self-hosting identity, AI output-fluency certificate, global observer, or detached T row appears in the carrier, those claims cannot enter the local NameCert obligations. □

Theorem 6510.3 (Conscious observation route latency and refusal)**.** *Every accepted* ConsciousObservationRoute$^{\uparrow}$ *read that exports an I-proxy must first expose local history, selector, self-proxy, latency or refusal,* hsame, Cont, Pkg, *and* NameCert *rows. A read that omits any of these rows is rejected by the refusal coordinate before the I-proxy export is available.*

Proof. Start from an accepted carrier $R = (L, \Theta, S, \rho, \Lambda, F, M, H, C, P, N)$. The rows L, Θ, S, ρ are the only coordinates that can display the local history, selector, self-proxy, and latency boundary of the observation.

The refusal row Λ is read before export. If any required local-history, selector, self-proxy, latency, or field-faithfulness coordinate is missing, Λ records the failure at that row and the replay cannot proceed to an I-proxy export.

When the required rows are present, F, H, C, P, N supply field-faithfulness, componentwise transport, replay, provenance, and local naming. Thus every accepted I-proxy export remains inside the displayed finite route, and every omitted-field read is refused before export. □

Falsifiable prediction: Any accepted $\mathsf{ConsciousObservationRoute}^{\uparrow}$ read is refuted if it exports an I-proxy without displaying local history, selector, self-proxy, latency or refusal status, hsame, Cont, Pkg, and NameCert rows.

Independence witness: role-separation-ledger, self-proxy-binding, public-scientific-reflection, inscriptionevent, event-instance: no bijection preserves the $\mathsf{ConsciousObservationRoute}^{\uparrow}$ carrier because the listed siblings separately expose role separation, self-proxy criteria, public audit fields, inscription events, or event instances, while this packet binds local Hist-time, selector, self-proxy, latency, refusal, field-faithfulness, transport, replay, provenance, and local naming as one observation route.

Falsifiable prediction: The future Lean TasteGate surface for BEDC.Derived.ConsciousObservationRouteUp must include Nontrivial $\mathsf{ConsciousObservationRoute}^{\uparrow}$ and FieldFaithful $\mathsf{ConsciousObservationRoute}^{\uparrow}$ instances before this packet can be promoted beyond paper-side seed status.

Closure status for $\mathsf{NameCert}_{\mathsf{ConsciousObservationRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{ConsciousObservationRoute}^{\uparrow}$ packet is generated from local Hist-time, selector, self-proxy, latency, refusal, field-faithfulness, role-separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConsciousObservationRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6510.1, Definition 6510.2, and Definition 6510.3 seed the local observation route separated by the compiler, AI/ML, and human-consciousness comparison.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConsciousObservationRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject-substance, phenomenal proof, compiler self-hosting identity, AI output fluency as I-proxy status, global observerhood, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local observation exposure, refusal exactness, role-separation stability, transport and replay stability, and ledger non-escape for BEDC.Derived.ConsciousObservationRouteUp.

Chapter 6511

A Concrete Naming Certificate for $\mathsf{EffectiveBishopCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ packet records the finite completion-facing route by which regular rational names are read through displayed dyadic tolerance budgets, finite stream windows, and the terminal real-name seal. It sits over the dyadic ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the Cauchy completion sibling chapter 1639. The packet names an effective Bishop completion boundary as finite NameCert data; it does not quotient streams, select limits, compare host real values, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6511.1 (Effective Bishop completion carrier)**.** *An* $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, S, D, Q, T, H, K, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, naming the regular rational sequence that supplies the completion-facing readback.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule, recording the finite observation windows inspected before any terminal seal is consumed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic budgets and common-exponent comparisons used by the window and regularity rows.*
4. *Q is the modulus and readback row, pairing each displayed tolerance request with the finite window and regular rational read that answer it.*
5. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the regular source, stream windows, dyadic ledger, modulus and readback row, terminal seal, provenance, and local naming data.*
7. *K records displayed* Cont *replay from a completion-facing tolerance request through D, then through S, R, Q, and finally into T.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveBishopCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveBishopCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, Q, T, H, K, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, host real equality, countable choice, arbitrary Cauchy-name existence, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6511.2 (Effective Bishop completion NameCert obligations)**.** *For every accepted* EffectiveBishopCompleti *carrier $E = (R, S, D, Q, T, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EffectiveBishopCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, modulus and readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

No NameCert obligation is stored in a quotient of streams, a selected limit, a host real value, a countable-choice schedule, an arbitrary Cauchy-name generator, or an ambient completeness coordinate.

Proof. Project the carrier of Definition 6511.1. The accepted packet exposes exactly the rows $R, S, D, Q, T, H, K, P, N$.

The source and window rows are finite. The $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies the regular rational source, while the $\mathsf{StreamName}^{\uparrow}$ row S supplies only displayed finite windows for the readback.

The tolerance and modulus rows are also displayed. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D supplies the dyadic budgets and common-exponent comparisons, and Q records which finite window and regular rational read answer each tolerance request.

Finally H, K, P, N transport, replay, source, and name the same route before the terminal seal T is consumed. Since the carrier contains no coordinate for quotient streams, selected limits, host equality, countable choice, arbitrary Cauchy-name existence, or ambient completeness, the listed rows exhaust the NameCert surface. □

Theorem 6511.3 (Effective Bishop completion RegSeqRat–Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ *carrier $E = (R, S, D, Q, T, H, K, P, N)$ must pass through the ordered route*

$$D, S, R, Q, H, K, P, N, T$$

before the terminal Real seal is consumed. Thus the read exposes the dyadic tolerance ledger, finite StreamName windows, RegSeqRat source and readback, modulus row, componentwise transport, replay, provenance, and local NameCert support before it reaches $\mathsf{Real}^{\uparrow}$.

The handoff exports a finite effective Bishop completion boundary. It exports no quotient stream equality, selected limit, host real equality, arbitrary Cauchy-name existence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier projection from Definition 6511.1. The terminal seal row T is a displayed coordinate of the same finite packet as R, S, D, Q, H, K, P, N, not a detached host limit.

Apply Definition 6511.2. It forces every completion-facing consumer to read the regular source, stream-window schedule, dyadic tolerance ledger, modulus and readback row, transport, replay, provenance, and local naming rows as the local certificate surface.

The RegSeqRat-to-Real step is then row-local. The dyadic ledger D answers tolerance requests, the schedule S exposes the finite windows, the source R and readback row Q provide the regular rational response, and H, K preserve the route through hsame and Cont before T is consumed.

Therefore the terminal seal is downstream of the displayed BEDC route. Since no accepted row stores quotient stream equality, selected limits, host equality, arbitrary Cauchy-name existence, countable choice, or Real completeness, the handoff cannot expose any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ completion-facing read must expose a RegSeqRat source row, StreamName window schedule, DyadicRatCore tolerance ledger, modulus and readback row, hsame transport, Cont replay, Pkg provenance, local NameCert, and terminal RealUp seal before it can serve a Real-facing consumer.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishopregularcauchycompletion: none is bijective with $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ because this packet jointly carries the displayed tolerance ledger, finite window schedule, regular rational source, modulus/readback row, terminal seal, transport, replay, provenance, and local naming for the effective Bishop completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveBishopCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveBishopCompletion}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite-window schedule, DyadicRatCore tolerance ledger, modulus and readback row, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveBishopCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6511.1, Definition 6511.2, and Definition 6511.3 seed the finite effective Bishop completion route over RegSeqRat, StreamName, DyadicRatCore, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveBishopCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, arbitrary Cauchy-name existence, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular source exposure, finite-window schedule exposure, dyadic tolerance budgeting, modulus/readback stability, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveBishopCompletionUp.

Chapter 6512

A Concrete Naming Certificate for $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$

Following Kelley and Bourbaki on filters generated by subbases, the $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ packet records the standard cofinal-subbasis route into Cauchy-filter completion as finite BEDC NameCert data. It sits after the subbasis generator of chapter 4232, the generated-base interface of chapter 1601, the Cauchy-filter carrier of chapter 333, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the route from displayed cofinal subbasis data to completion-facing reads; it does not quotient filters, select limits, invoke countable choice, compare host real values, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6512.1 (Filter subbasis cofinal completion carrier)**.** *A* $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, B, F, W, R, L, H, K, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *row, carrying finite subbasis tokens, finite intersections, generated events, and the generation ledger.*
2. *B is the generated* $\mathsf{FilterBase}^{\uparrow}$ *row read cofinally from the finite-intersection surface of S.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row reached only after the generated base row has been displayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by completion-facing consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the generated base, Cauchy-filter row, windows, and readback have been consumed.*
7. *H records componentwise* hsame *transport for the subbasis, generated base, Cauchy filter, windows, readback, Real seal, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from subbasis tokens through generated finite intersections, the cofinal base row, Cauchy-filter handoff, windows, readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$, $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, B, F, W, R, L, H, K, P, N$. It has no coordinate for quotient filter equality, selected limits, countable-choice schedules, host equality of generated filters, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6512.2 (Filter subbasis cofinal completion NameCert obligations)**.** *For every accepted* $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ *carrier $C = (S, B, F, W, R, L, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterSubbasisCofinalCompl}}$ *surface consists exactly of subbasis packet exposure through S, cofinal generated* $\mathsf{FilterBase}^{\uparrow}$ *exposure through B,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff through F,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through L, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6512.1. The accepted packet exposes exactly the rows $S, B, F, W, R, L, H, K, P, N$.

The subbasis and generated-base rows are upstream. The row S supplies the finite tokens, finite intersections, generated events, and generation ledger, while B records the cofinal generated base read from that surface.

The completion-facing rows are downstream. A consumer reaches F only after B, then reads windows through W, regular rational readback through R, and the terminal Real seal through L.

Finally H, K, P, N transport, replay, source, and name the same route. Since quotient filters, selected limits, countable choice, host equality of generated filters, host real equality, and ambient Real completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6512.3 (Filter subbasis cofinal completion route)**.** *Every completion-facing read exported by an accepted* $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ *carrier must factor through the* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *row S, the cofinal generated* $\mathsf{FilterBase}^{\uparrow}$ *row B, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row L in that order. The route is a finite cofinal-subbasis completion handoff, not a quotient filter construction or a selected-limit theorem.*

Proof. Start with the NameCert obligations of Definition 6512.2. They force every accepted consumer to expose S, B, F, W, R, L before any structural row can name the completion-facing read.

Use the generated-base cofinality theorem for $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, Definition 4232.4. It identifies the generated base as cofinal among the finite intersections displayed by the subbasis packet before Cauchy-filter and Real-facing rows are read.

Apply the refinement-stability theorem Definition 4232.6. Cofinal replacement of the displayed generated base preserves the same CauchyFilter, StreamName, RegSeqRat, and Real seal handoff.

Thus a completion-facing consumer first reads the finite subbasis route, then the cofinal generated base, and only then the Cauchy-filter, window, readback, and Real seal rows. The absent classifier coordinates exclude quotient filters, selected limits, choice schedules, host generated-filter equality, host real equality, and ambient completeness. □

Lean4 (statement) `BEDC.Derived.FilterSubbasisCofinalCompletionUp`

Falsifiable prediction: Every accepted $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ completion-facing read exposes a CauchyFilterSubbasisUp row, generated cofinal FilterBaseUp row, CauchyFilterUp row, StreamNameUp windows, RegSeqRatUp readback, terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before it can serve a Real-facing completion consumer.

Independence witness: cauchy-filter-subbasis, filterbase, cauchyfilter, streamname, regseqrat, real: none is bijective with $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ because this packet jointly carries the displayed cofinal subbasis route, generated base handoff, Cauchy-filter handoff, finite windows, readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}$ packet is generated from a CauchyFilterSubbasis row, a cofinal generated FilterBase row, a CauchyFilter row, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6512.1, Definition 6512.2, and Definition 6512.3 seed the cofinal subbasis route into Cauchy-filter completion over generated base, window, readback, Real seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterSubbasisCofinalCompletion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected limits, countable choice, host equality of generated filters, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subbasis exposure, generated-base cofinality, CauchyFilterUp handoff, StreamNameUp window exposure, RegSeqRatUp readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FilterSubbasisCofinalCompletionUp.

Chapter 6513

A Concrete Naming Certificate for $\mathsf{MooreAronszajnKernel}^{\uparrow}$

Following Moore and Aronszajn on reproducing kernels, the $\mathsf{MooreAronszajnKernel}^{\uparrow}$ packet records the finite positive-kernel route as BEDC NameCert data. It sits after the Hilbert-space packet of chapter 91, the vector-space rows of chapter 47, the finite matrix readback of chapter 49, the functional-analysis handoff of chapter 94, and the real scalar seal of chapter 38. The packet names finite Gram-matrix reproducing-kernel evidence without selecting an infinite feature map, quotienting functions, completing an ambient RKHS, importing a host Hilbert space, assuming countable choice, or asserting positive-kernel analysis beyond the displayed rows.

Definition 6513.1 (Moore–Aronszajn kernel carrier)**.** *A* $\mathsf{MooreAronszajnKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (K_0, G, V, H, M, R, C, P, N)$$

with the following displayed rows:

1. K_0 *is the kernel-value row. It records the finite list of source points, displayed kernel evaluations, and symmetry requests inspected by the finite packet.*
2. G *is the positive finite Gram row. It records the finite Gram matrix over the displayed source points, the diagonal and symmetry witnesses, and the finite positivity ledger for every displayed coefficient window.*
3. V *is the* $\mathsf{VecSpace}^{\uparrow}$ *source row. It records the finite vector expressions, scalar multiplication requests, and linear-combination windows from which Gram reads are formed.*
4. H *is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It records the inner-product readout surface and norm-facing rows through which the reproducing-kernel handoff may be interpreted.*
5. M *is the* $\mathsf{Mat}^{\uparrow}$ *readback row. It records the finite row-column indexing, entry readbacks, Gram-row projections, and matrix-vector products read before any RKHS-style consumer can use* G*.*
6. R *is the* $\mathsf{Real}^{\uparrow}$ *scalar seal. It records real scalar values, positive-radius comparisons, and the terminal real-name boundary for every displayed Gram and norm read.*
7. C *records componentwise* hsame *transport together with displayed* Cont *replay for the route*

$$K_0, G, V, H, M \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{MooreAronszajnKernel}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreAronszajnKernel}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $K_0, G, V, H, M, R, C, P, N$ and has no coordinate for an infinite RKHS completion, selected feature map, quotient function space, countable-choice schedule, ambient Hilbert completion, or uninspected positive-kernel theorem.

Lean4 ✓ `BEDC.Derived.MooreAronszajnKernelUp.MooreAronszajnKernelTasteGate single carrier alignment`

Theorem 6513.2 (Moore–Aronszajn kernel NameCert obligations)**.** *For the carrier of Definition 6513.1, the local* $\mathsf{NameCert}_{\mathsf{MooreAronszajnKernel}^{\uparrow}}$ *surface consists exactly of carrier admission, finite kernel-value exposure, positive Gram-row exposure,* $\mathsf{VecSpace}^{\uparrow}$ *source compatibility,* $\mathsf{Hilbert}^{\uparrow}$ *inner-product handoff,* $\mathsf{Mat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *scalar sealing, componentwise* hsame *transport with displayed* Cont *replay,* Pkg *provenance, and local naming. The surface exports no selected feature map, quotient function space, infinite completion, countable choice, host Hilbert space, or ambient RKHS theorem.*

Proof. Carrier admission is projection from Definition 6513.1. The packet displays exactly the finite kernel row, Gram row, vector-space row, Hilbert row, matrix readback, real scalar seal, structural replay, provenance, and local naming row.

The positive-kernel obligation is finite. The row G records only displayed Gram windows and their positivity ledgers, while M supplies the row-column readback and R supplies the real scalar comparison boundary.

The analytic handoff remains local to the displayed siblings. The rows V and H expose only the $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ surfaces named by P, and the functional-analysis route is read as provenance rather than as an ambient completion theorem.

Finally C preserves the same rows under hsame and Cont replay, and N names no off-packet coordinate. Therefore the listed obligations exhaust the local NameCert surface and none of the excluded RKHS-side objects can be read from the packet. □

Theorem 6513.3 (Moore–Aronszajn kernel Hilbert handoff)**.** *Every Hilbert-facing or functional-analysis consumer of an accepted* $\mathsf{MooreAronszajnKernel}^{\uparrow}$ *packet must first read the finite kernel row K_0, the positive finite Gram row G, the* $\mathsf{VecSpace}^{\uparrow}$ *source row V, the* $\mathsf{Hilbert}^{\uparrow}$ *inner-product source row H, the* $\mathsf{Mat}^{\uparrow}$ *readback row M, the* $\mathsf{Real}^{\uparrow}$ *scalar seal R, and the structural rows C, P, N. The handoff reaches the sibling surfaces of chapter 91, chapter 47, chapter 49, chapter 94, and chapter 38 only through those displayed rows.*

Proof. Start with the NameCert obligations of Definition 6513.2. They force every consumer to inspect $K_0, G, V, H, M, R, C, P, N$ before a Hilbert-facing read is available.

The Hilbert handoff is routed through displayed finite data. The vector-space row supplies the linear-combination windows, the matrix row supplies the finite Gram readback, and the real row supplies scalar comparison before the Hilbert row may be consumed.

The provenance row names the five sibling chapters, and the replay row orders the read through the finite Gram route. Since no row stores a selected feature map, quotient function space, or completed RKHS carrier, the Hilbert-facing consumer receives only the displayed finite handoff. □

Theorem 6513.4 (Moore–Aronszajn kernel finite Gram nonescape)**.** *Every reproducing-kernel read of an accepted* $\mathsf{MooreAronszajnKernel}^{\uparrow}$ *packet factors through the finite Gram row G, the* $\mathsf{Mat}^{\uparrow}$

readback row M, the VecSpace$^{\uparrow}$ *source row V, the* Hilbert$^{\uparrow}$ *source row H, the* Real$^{\uparrow}$ *scalar seal R,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The packet does not export selected RKHS points, quotient function equality, infinite Gram operators, countable choice, ambient Hilbert completion, or full Moore–Aronszajn existence.*

Proof. Project the carrier rows. The only route from kernel values to an analytic consumer is the displayed replay

$$K_0, G, V, H, M \longmapsto R.$$

The finite Gram row is therefore unavoidable. Matrix readback exposes only the finite rows and columns displayed by M, vector expressions are those displayed by V, and scalar comparisons end at R.

Transport and replay cannot create new analytic objects. They preserve the same finite rows under hsame and Cont, while P, N name only the listed packet surface. Hence every reproducing-kernel read remains inside the finite Gram route and none of the excluded RKHS-side data can escape the packet. □

Lean4 (statement) `BEDC.Derived.MooreAronszajnKernelUp`

Closure status for NameCert$_{\text{MooreAronszajnKernel}^{\uparrow}}$.

Constructive story: A MooreAronszajnKernel$^{\uparrow}$ packet is generated from finite kernel-value rows, positive finite Gram rows, VecSpace and Hilbert source rows, Matrix readback, Real scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MooreAronszajnKernel}^{\uparrow}}$, seed)

Scope closed: Definition 6513.1, Definition 6513.2, Definition 6513.3, and Definition 6513.4 seed the finite Gram-matrix reproducing-kernel carrier and its Hilbert-facing handoff boundary.

Formal verification: FormalStatus(NameCert$_{\text{MooreAronszajnKernel}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.MooreAronszajnKernelUp`

Bridge status: none

Not claimed: This chapter does not close selected feature maps, quotient function spaces, infinite RKHS completion, countable choice, ambient Hilbert completeness, full Moore–Aronszajn existence, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite Gram positivity, Matrix readback, VecSpace and Hilbert handoff, Real scalar sealing, transport, replay, provenance, and local naming for BEDC.Derived.MooreAronszajnKernelUp.

Chapter 6514

A Concrete Naming Certificate for $\mathsf{BishopBarInductionBoundary}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bishop–Bridges, the $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ packet records the finite boundary at which bar-induction evidence becomes available to Real and Completion consumers without turning into a choice or compactness principle. It sits beside the barred-spread source of chapter 1675, the finite fan route of chapter 400, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite bar handoff for analytic neighbours; it is not countable choice, a selected infinite branch, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6514.1 (Bishop bar induction boundary carrier)**.** *A* $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (B, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *barred-spread row, recording the finite bar decisions, monotone closure, and local induction replay visible at the retained prefix boundary.*
2. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row, recording the bounded prefix cover and depth ledger consumed before a boundary read is public.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for Real and Completion-facing observations of the retained prefixes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic scales and comparison cells used to interpret the finite windows and regular rational readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, F, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the barred-spread row, finite fan row, StreamName windows, RegSeqRat readback, dyadic tolerances, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary request through $B, F, S, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopBarInductionBoundary}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopBarInductionBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $B, F, S, R, D, E, H, C, P, N$. It has no coordinate for countable choice, a selected infinite branch, quotient stream equality, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6514.2 (Bishop bar induction boundary NameCert obligations)**.** *For every accepted* $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ *carrier*

$$Y = (B, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopBarInductionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, barred-spread ancestry, finite fan-boundary exposure, finite StreamName window admission, RegSeqRat readback, dyadic tolerance exactness, Real seal ordering, componentwise transport, displayed continuation replay, provenance, and local naming for the carrier of Definition 6514.1.*

Proof. Project the accepted carrier. Its public rows are precisely $B, F, S, R, D, E, H, C, P, N$, so carrier admission and local naming are fixed by Definition 6514.1.

The barred-spread ancestry row B supplies the finite bar decisions and induction replay, while the fan row F supplies only bounded prefix data. Thus the boundary is finite before any analytic read can inspect StreamName, RegSeqRat, dyadic, or Real-facing rows.

The readback rows are ordered by the carrier itself: S displays the finite windows, R reads those windows as regular rational data, D supplies the dyadic tolerance ledger, and E is reached only after those rows are present. The transport, replay, provenance, and naming rows preserve this order through H, C, P, N.

Since no coordinate stores countable choice, selected branches, quotient stream equality, open-cover compactness, host real equality, or ambient Real completeness, none of those principles can be a local NameCert obligation. □

Theorem 6514.3 (Bishop bar induction boundary finite-bar handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion-facing consumer of an accepted* $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ *packet reads the finite bar route*

$$\mathsf{BrouwerBarInduction}^{\uparrow}, \mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the boundary is public. Consequently the handoff exports finite barred-spread and fan-window data to Real-and-Completion neighbours without exporting countable choice, selected infinite branches, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $Y = (B, F, S, R, D, E, H, C, P, N)$. The only bar source is B, and the only finite fan source is F, so a boundary consumer must display these rows before it can request analytic data.

Next the consumer reads the finite observation rows. The $\mathsf{StreamName}^{\uparrow}$ window S gives the inspected prefixes, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads those prefixes as regular rational data, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the displayed tolerance comparisons.

The terminal $\mathsf{Real}^{\uparrow}$ row E is downstream of that finite route. Componentwise transport, continuation replay, provenance, and local naming only preserve the same path through H, C, P, N.

There is no remaining insertion point for countable choice, a selected infinite branch, quotient stream equality, open-cover compactness, or ambient Real completeness. The handoff is therefore exactly the finite boundary route displayed in the statement. □

Falsifiable prediction: Every accepted $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ Real or Completion-facing read displays BrouwerBarInductionUp barred-spread data, FanTheoremUp finite fan rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the boundary is public.

Independence witness: brouwer-bar-induction, fantheorem, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite bar ancestry, finite fan-boundary exposure, finite windows, regular rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopBarInductionBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopBarInductionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6514.1, Definition 6514.2, and Definition 6514.3 seed the finite bar boundary route from Brouwer bar induction and finite fan rows through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopBarInductionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected infinite branches, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, barred-spread ancestry, finite fan-boundary exposure, window admission, readback ordering, dyadic exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopBarInductionBoundaryUp.

Constructive story: A $\mathsf{BishopBarInductionBoundary}^{\uparrow}$ packet is generated from BrouwerBarInductionUp barred-spread rows, FanTheoremUp finite fan rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6515

A Concrete Naming Certificate for $\mathsf{UniformizableSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{UniformizableSpace}^{\uparrow}$ packet records the finite route by which displayed uniformity data induces a topology. It sits after the uniform entourage surface of chapter 339, the topology carrier of chapter 88, the metric-to-topology route of chapter 1655, the complete-uniform-space boundary of chapter 1588, and the directed entourage discipline of chapter 2334. The packet also leaves room for separated uniformity consumers without importing a host metrization theorem, quotient topology, selected base, or open-set extensionality principle.

Definition 6515.1 (Uniformizable-space carrier)**.** *A* $\mathsf{UniformizableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, U, E, N, H, C, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *row whose neighbourhood and open-set reads are induced from the displayed uniformity data.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *row carrying the finite uniformity interface used by the packet.*
3. *E is the entourage-base row, listing the displayed entourages and the finite intersection, diagonal, and composition requests that generate neighbourhood reads.*
4. *N is the induced-neighbourhood evidence row, recording how each inspected point-neighbourhood request is read from an entourage section.*
5. *H records componentwise* hsame *transport for topology, uniformity, entourage-base, induced-neighbourhood, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from topology requests through U, E, N, H and back to the same topology row T.*
7. *P is the* Pkg *provenance over* $\mathsf{UniformizableSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformizableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, U, E, N, H, C, P, L. It has no coordinate for choice-based metrization, quotient topology, a host-selected uniform base, open-set extensionality, ambient separatedness, or completeness.

Theorem 6515.2 (Uniformizable-space NameCert obligations)**.** *For every accepted* $\mathsf{UniformizableSpace}^{\uparrow}$ *carrier*

$$Z = (T, U, E, N, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{UniformizableSpace}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed topology, uniformity, entourage-base, induced-neighbourhood, transport, replay, provenance, and local naming rows;*
2. *entourage-base coverage, requiring every topology-facing neighbourhood read to be generated from a displayed entourage in E over the uniformity row U;*
3. *induced-neighbourhood stability, requiring N to preserve the same point and entourage section under componentwise* hsame *transport;*
4. *displayed* Cont *replay, requiring every topology request to return through the same U, E, N, H rows before it is consumed as T;*
5. *non-escape, requiring P, L to expose no choice-based metrization, quotient topology, host-selected base, open-set extensionality, ambient separatedness theorem, or completeness theorem.*

No Uniformizable-space obligation is stored outside these rows.

Proof. Project the accepted carrier. The packet exposes exactly the topology row T, uniformity row U, entourage-base row E, induced-neighbourhood evidence N, transport row H, replay row C, provenance row P, and local name L.

Read the uniformity data before the topology data. The entourage-base row E supplies the finite diagonal, intersection, and composition information used by neighbourhood requests, and N records the pointwise section through which such a request reaches T.

Transport and replay preserve the same route. Componentwise hsame acts through H, while Cont replay through C returns topology-facing reads to the same displayed entourage and neighbourhood rows.

Finally, provenance and local naming do not add external objects. Since the carrier has no coordinate for choice-based metrization, quotient topology, selected bases, open-set extensionality, separatedness, or completeness, the five listed obligations exhaust the local NameCert surface. □

Theorem 6515.3 (Uniformizable-space entourage topology scope)**.** *For every accepted* $\mathsf{UniformizableSpace}^{\uparrow}$ *carrier $Z = (T, U, E, N, H, C, P, L)$, every topology readback from T factors through the displayed uniformity row U, entourage-base row E, induced-neighbourhood row N, transport row H, replay row C, provenance row P, and local name L. The readback is the topology induced by displayed entourages; it does not use choice-based metrization, quotient topology, host open-set extensionality, a selected base, ambient separatedness, or completeness.*

Proof. Begin with a topology readback request. The carrier definition makes T readable only together with the uniformity, entourage-base, and induced-neighbourhood rows U, E, N.

Apply the NameCert obligations. Entourage-base coverage forces the neighbourhood request through a displayed entourage, and induced-neighbourhood stability keeps the point and entourage section fixed under transport.

Use the replay and provenance rows next. The row C replays the same route back to T, while P and L source and name only the displayed packet.

The excluded topology principles are absent from the carrier. Therefore the topology readback factors through displayed entourages and cannot appeal to metrization by choice, quotient topology, host open-set extensionality, selected bases, separatedness, or completeness. □

Falsifiable prediction: Every $\mathsf{UniformizableSpace}^{\uparrow}$ topology read factors through displayed UniformSpace entourages, induced-neighbourhood evidence, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: uniformspace, topology, metric-topology, complete-uniform-space, quasi-uniform-space, separated-uniform-space: $\mathsf{UniformizableSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a topology row, displayed uniformity row, entourage base, induced-neighbourhood evidence, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformizableSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformizableSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6515.1, Definition 6515.2, and Definition 6515.3 seed the constructive L10 route from displayed uniformity and entourage data to topology readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformizableSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-based metrization, quotient topology, host open-set extensionality, selected bases, ambient separatedness, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage-base coverage, induced-neighbourhood stability, displayed replay, provenance non-escape, and local naming for BEDC.Derived.UniformizableSpaceUp.

Constructive story: A $\mathsf{UniformizableSpace}^{\uparrow}$ packet is generated from a Topology row, UniformSpace row, entourage-base row, induced-neighbourhood evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6516

A Concrete Naming Certificate for $\mathsf{LeibnizAlternatingTest}^{\uparrow}$

Following Leibniz and Bishop–Bridges constructive series theory, the $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ packet records the finite test-level route by which alternating partial sums, a monotone null tail, dyadic tolerance ledgers, regular rational readback, and a terminal real seal are accepted before a Real or Completion consumer may treat the series as convergent. It consumes the remainder certificate of chapter 2159, the regular rational carrier of chapter 301, the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, and the terminal $\mathsf{Real}^{\uparrow}$ surface of chapter 38.

Definition 6516.1 (Leibniz alternating test carrier)**.** *A* $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, A, M, S, D, Q, W, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the accepted* $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ *row, exposing the next-omitted-term remainder certificate already routed through alternating signs, monotone magnitudes, and dyadic budgets.*
2. *A is the alternating partial-sum schedule, recording the inspected parity rows, finite partial sums, adjacent direction checks, and cutoff schedule used by the test.*
3. *M is the monotone null-tail row, recording nonnegative magnitude reads, observed monotonicity comparisons, and the null-tail request that makes the next omitted magnitude smaller than any requested dyadic tolerance.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family for summands, signs, magnitudes, partial sums, cutoffs, next omitted terms, dyadic comparisons, and endpoint-facing reads.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger pairing every precision request with a cutoff, a next omitted magnitude, and a comparison row.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from A, M, S, D and the accepted remainder row R.*
7. *W is the test-level Cauchy handoff row: it records that the partial-sum schedule is Cauchy by reading only the displayed remainder certificate, monotone null tail, dyadic tolerance ledger, and regular rational readback.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, A, M, S, D, Q, W have been displayed.*

9. *H records componentwise* hsame *transport for the remainder row, partial-sum schedule, monotone null tail, stream windows, dyadic ledger, regular readback, Cauchy handoff, real seal, replay, provenance, and local name.*

10. *C records displayed* Cont *replay of the route*

$$R, A, M, S, D \longmapsto Q \longmapsto W \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{LeibnizAlternatingTest}^{\uparrow}$, $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, M, S, D, Q, W, E, H, C, P, N$. It has no coordinate for a selected infinite sum, quotient stream equality, host real equality, countable choice, unrestricted trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a Cauchy proof detached from the displayed remainder route.*

Theorem 6516.2 (Leibniz alternating test NameCert obligations)**.** *For every accepted* $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ *carrier*

$$T = (R, A, M, S, D, Q, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingTest}^{\uparrow}}$ *surface consists exactly of carrier admission for Definition 6516.1, the accepted alternating-remainder row R, the alternating partial-sum schedule A, the monotone null-tail row M, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the Cauchy handoff row W, the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier $T = (R, A, M, S, D, Q, W, E, H, C, P, N)$. The tuple exposes exactly the remainder, schedule, monotone tail, finite-window, dyadic, regular-readback, Cauchy-handoff, real-seal, and structural rows.

The test begins with R, A, M. The remainder row supplies the next-omitted-term certificate, the schedule row supplies alternating partial sums and parity, and the monotone null-tail row supplies the decreasing magnitude and tail request used by every tolerance read.

The finite observation surface is S, D, Q. The stream windows show the inspected data, the dyadic ledger pairs each requested tolerance with a cutoff and comparison, and the regular rational readback reuses exactly those windows.

Finally W, E, H, C, P, N expose the Cauchy handoff, seal it as a Real-facing row, transport it componentwise, replay the displayed route, record provenance, and name the packet. Since the carrier contains no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6516.3 (Leibniz alternating test Cauchy handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ *packet must read the finite route*

$$R, A, M, S, D \longmapsto Q \longmapsto W \longmapsto E$$

before treating the alternating series as convergent. Thus the Cauchy handoff W is available only after the existing alternating-remainder row, monotone null-tail schedule, partial-sum schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row have all been displayed.*

Proof. Start with the accepted remainder row R. By chapter 2159, it already binds every endpoint-facing remainder request to the next omitted magnitude, alternating sign ledger, monotone null tail, dyadic budget, finite windows, and regular readback.

Attach the test-level schedule rows A, M, S, D. These rows name the inspected partial sums, the monotone null-tail request, the finite windows, and the dyadic tolerance ledger used to compare later partial sums through the common cutoff.

Read Q as the regular rational surface for those displayed rows. The Cauchy handoff W can then be formed from the same finite observations, because the difference between sufficiently late partial sums is bounded by the displayed remainder budget.

The terminal seal E is reached only after W, and the structural rows H, C, P, N preserve that route. Hence any consumer that bypasses the remainder row, tail schedule, dyadic ledger, regular readback, stream windows, transport, replay, provenance, or local name leaves the accepted $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ packet. □

Falsifiable prediction: Every $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ Real or Completion read factors through an accepted $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ row, alternating partial-sum schedule, monotone null-tail row, StreamNameUp windows, DyadicRatCoreUp tolerance ledger, RegSeqRatUp readback, Cauchy handoff, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingTest}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeibnizAlternatingTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6516.1, Definition 6516.2, and Definition 6516.3 seed the finite Leibniz alternating-test certificate from remainder rows and monotone null tails through dyadic tolerance ledgers, regular rational readback, Cauchy handoff, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeibnizAlternatingTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient stream equality, host real equality, countable choice, unrestricted trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, detached Cauchy proofs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, remainder-row consumption, partial-sum schedule stability, monotone null-tail exactness, dyadic tolerance stability, RegSeqRat readback, StreamName finite-window compatibility, Cauchy handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LeibnizAlternatingTestUp.

Constructive story: A $\mathsf{LeibnizAlternatingTest}^{\uparrow}$ packet is generated from an accepted LeibnizAlternatingRemainderUp row, alternating partial-sum schedule, monotone null-tail row, StreamNameUp windows, DyadicRatCoreUp tolerance ledger, RegSeqRatUp readback, Cauchy handoff row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6517

A Concrete Naming Certificate for Rule110CellularTrustSubstrate$^{\uparrow}$

The Rule110CellularTrustSubstrate$^{\uparrow}$ packet realizes the Rule 110 axis of chapter 9390 as finite NameCert data. It refines the generic cellular trust substrate of chapter 937 by fixing the eight-entry Rule 110 transition table, manifest schema, orbit-window replay, classifier surface, frontier ledger, transport rows, and provenance rows before any cellular-pattern or cyclic-tag consumer can read a substrate observation.

Definition 6517.1 (Rule 110 cellular trust substrate carrier)**.** *A* Rule110CellularTrustSubstrate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, M, O, K, F, B, U, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the Rule 110 transition-table row, recording the eight three-cell cases and their output bits.*
2. *M is the manifest row, recording the finite initial row, width, step count, expected observation claims, and fixed text-format schema.*
3. *O is the orbit-window row, containing only finite prefixes of the Rule 110 evolution generated from T and M.*
4. *K is the classifier row, read through the cellular row classifier of chapter 941, the Rule 110 pattern classifier of chapter 5844, and the glider catalog of chapter 1045.*
5. *F is the Cook-frontier ledger, read through chapter 959 when a bounded cyclic-tag or compile-frontier consumer is present.*
6. *B is the cellular-substrate boundary row, recording that the packet is a closed observational substrate rather than a host evaluator or universality theorem.*
7. *U is the bounded cyclic-tag handoff row, read through chapter 3764 only after the finite orbit and frontier rows have been displayed.*
8. *A is the local-stencil audit row, read through chapter 6004 for each inspected transition.*
9. *H records componentwise* hsame *transport for the rule table, manifest, orbit window, classifier, frontier ledger, boundary, bounded handoff, audit row, replay, provenance, and local name.*

10. *C records displayed* Cont *replay of the route*

$$T, M, A \longmapsto O \longmapsto K, F, B, U.$$

11. *P is the* Pkg *provenance over* Rule110CellularTrustSubstrate$^{\uparrow}$, CellularTrustSubstrate$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, RuleOneTenPatternClassifier$^{\uparrow}$, RuleOneTenGliderCatalog$^{\uparrow}$, RuleOneTenLocalStencilObservat CyclicTagSystem$^{\uparrow}$, CookFrontierWitness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{Rule110CellularTrustSubstrate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, O, K, F, B, U, A, H, C, P, N$. It has no coordinate for a host C evaluator, a complete infinite orbit, a detached universality theorem, unbounded halting, selected glider semantics, or an external proof of induction.

Theorem 6517.2 (Rule 110 cellular trust substrate NameCert obligations)**.** *For every accepted* Rule110CellularTrustSubstrate$^{\uparrow}$ *carrier*

$$R = (T, M, O, K, F, B, U, A, H, C, P, N),$$

the local NameCert$_{\text{Rule110CellularTrustSubstrate}^{\uparrow}}$ *surface consists exactly of carrier admission for Definition 6517.1, transition-table exactness through T, manifest schema admission through M, finite orbit-window replay through O, classifier exposure through K, frontier-ledger exposure through F, closed-substrate boundary through B, bounded cyclic-tag handoff through U, local-stencil audit through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The tuple exposes exactly the transition table, manifest, orbit-window, classifier, frontier, boundary, cyclic-tag handoff, local-stencil audit, and structural rows.

The finite substrate rows are T, M, O, A. The transition table fixes the local rule, the manifest fixes the finite input and expected observations, the audit row checks each inspected local stencil, and the orbit window contains only the finite replay generated from those rows.

The consumer rows are K, F, B, U. Pattern consumers read K, frontier consumers read F, closed-substrate consumers read B, and bounded cyclic-tag consumers read U. Each consumer row is downstream of the displayed finite substrate route.

Finally H, C, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for the excluded host evaluator, complete orbit, detached universality theorem, unbounded halting principle, selected glider semantics, or external induction proof, the listed rows exhaust the local NameCert obligations. □

Theorem 6517.3 (Rule 110 cellular trust substrate observation boundary)**.** *Every accepted* Rule110CellularTrustSub *observation factors through the finite route*

$$T, M, A \longmapsto O \longmapsto K, F, B, U \longmapsto H, C, P, N.$$

Thus a pattern-classifier, frontier-ledger, bounded cyclic-tag, or closed-substrate consumer can read Rule 110 only after the finite rule table, manifest, orbit window, local-stencil audit, transport, replay, provenance, and local NameCert *rows have been displayed.*

Proof. Start with the local-stencil route. The transition table T and audit row A expose each inspected Rule 110 update as finite three-cell data, while the manifest row M fixes the initial row and bounded step count.

Replay the orbit window O from those rows. No consumer receives a whole infinite evolution; every read is a finite orbit prefix generated from the displayed table and manifest.

Now attach the public consumers. The classifier row K, frontier ledger F, closed-substrate boundary B, and bounded cyclic-tag handoff U can inspect only the same finite orbit route.

The rows H, C, P, N preserve and name the route. Any host evaluator, detached universality proof, unbounded halting claim, selected glider semantics, or external induction proof would require a coordinate outside the accepted packet. □

Falsifiable prediction: Every $\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}$ observation first exposes the finite Rule 110 transition table, manifest row, local-stencil audit, orbit-window replay, classifier row, frontier ledger, closed-substrate boundary, bounded cyclic-tag handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: cellulartrustsubstrate, cellularautomaton, cellularrowclassifier, rule110-pattern-classifier, cyclic-tag-system: $\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}$ is not bijective to any listed sibling because it jointly fixes the Rule 110 table, manifest schema, orbit windows, classifier, Cook-frontier ledger, bounded cyclic-tag handoff, audit route, transport, replay, provenance, and local naming.

The Lean TasteGate follow-up for `BEDC.Derived.Rule110CellularTrustSubstrateUp` must provide explicit $\mathsf{Nontrivial\ Rule110CellularTrustSubstrate}^{\uparrow}$ and $\mathsf{FieldFaithful\ Rule110CellularTrustSubstrate}^{\uparrow}$ instances over $T, M, O, K, F, B, U, A, H, C, P, N$. These instances keep the rule table, manifest, orbit-window, classifier, frontier, boundary, handoff, and audit rows as separate fields.

Closure status for $\mathsf{NameCert}_{\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6517.1, Definition 6517.2, and Definition 6517.3 seed the finite Rule 110 cellular trust substrate packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host evaluator correctness, complete infinite orbits, Cook universality, unbounded halting, selected glider semantics, internal induction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-table exactness, manifest admission, orbit-window replay, classifier exposure, Cook-frontier ledger exposure, bounded cyclic-tag handoff, local-stencil audit, transport, replay, provenance, TasteGate instances, and local naming for BEDC.Derived.Rule110CellularTrustSubstrateUp.

Constructive story: A $\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}$ packet is generated from a finite Rule 110 transition table, manifest row, orbit-window replay, classifier row, Cook-frontier ledger, closed-substrate boundary, bounded cyclic-tag handoff, local-stencil audit, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6518

A Concrete Naming Certificate for $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$

Following Bishop and Bridges' finite approximation discipline for constructive completions, the $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$ packet records the finite request-section boundary at which a dyadic endpoint and tolerance row, a stream-name section, a regular rational readback, and a terminal Real seal can be cited together. It refines the four-face completion surface of chapter 1844 by making the section pullback explicit over chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a finite NameCert interface for L10 completion consumers; it is not a quotient-stream construction, a selected limit, a countable-choice principle, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a checked bridge.

Definition 6518.1 (Finite section completion interface carrier)**.** *A* $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, S, R, B, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance row, recording the finite dyadic endpoint cells, scale bounds, and slack conversions inspected by the section.*
2. *S is the finite request-section row over* $\mathsf{StreamName}^{\uparrow}$*, listing the requested stream windows and the section coordinates through which a completion consumer reads them.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational data obtained from the section after the dyadic tolerance row and stream windows have been displayed.*
4. *B is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, recording only the finite seal-facing boundary reached after D, S, R.*
5. *Q is the section pullback and exactness row, requiring the requested stream section, dyadic tolerance cells, regular rational readback, and terminal seal row to agree componentwise on every displayed finite request.*
6. *H records componentwise* hsame *transport for dyadic, section, readback, seal, exactness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite completion request through D, S, R, B, Q, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSectionCompletionInterface}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $D, S, R, B, Q, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 6518.2 (Finite section completion interface NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$ *carrier*

$$I = (D, S, R, B, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSectionCompletionInterface}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed finite* BHist *packet of Definition 6518.1;*
2. *finite section coverage, requiring each requested* $\mathsf{StreamName}^{\uparrow}$ *window in S to be read against the displayed dyadic endpoint and tolerance row D;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the regular rational readback R to be exposed only through the already displayed D, S section route and the exactness row Q;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the terminal seal row B to be consumed only after D, S, R, Q and refusing quotient stream equality, selected limits, countable choice, host real equality, and ambient completeness;*
5. *structural transport, replay, provenance, and local naming through H, C, P, N, with no additional completion, quotient, choice, or bridge coordinate.*

Proof. Carrier admission is projection from Definition 6518.1. The accepted packet displays exactly the dyadic endpoint and tolerance row, finite request-section row, regular rational readback row, terminal Real seal row, section exactness row, and structural rows.

Finite coverage is rowwise. A completion request first reads D and S; only then can Q certify that the selected finite section agrees with the requested dyadic tolerance cells and stream windows.

The handoff to R and the seal read B are downstream of that exactness row. Since the carrier has no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or bridge coordinate, the terminal seal cannot escape the displayed finite section route.

Finally, H, C, P, N transport, replay, source, and name the same packet. These structural rows preserve the finite route through D, S, R, B, Q and add no further completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSectionCompletionInterface}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSectionCompletionInterface}^{\uparrow}}$, seed)

Scope closed: Definition 6518.1 and Definition 6518.2 seed the finite section interface among dyadic endpoint and tolerance rows, StreamName request sections, RegSeqRat readback, Real seal rows, pullback exactness, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSectionCompletionInterface}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires a Lean carrier and checked NameCert obligations for BEDC.Derived.FiniteSectionCompletionInt

Constructive story: A $\mathsf{FiniteSectionCompletionInterface}^{\uparrow}$ packet is generated from a dyadic endpoint and tolerance row, a finite StreamName request-section row, a RegSeqRat readback row, a terminal Real seal row, a section pullback and exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6519

A Concrete Naming Certificate for RegularCauchyCompletionComparisonTriangle$^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the RegularCauchyCompletionComparisonTriangle$^{\uparrow}$ packet records the finite comparison triangle that lets a completion-facing consumer read the regular-Cauchy route, the general completion-comparison route, and the terminal Real$^{\uparrow}$ seal as one NameCert surface. It sits after the modulus ledger of chapter 292, the diagonal selector of chapter 390, the finite comparison routes of chapter 1954 and chapter 2321, the regular rational readback of chapter 301, the stream window of chapter 126, the dyadic observation ledger of chapter 291, and the terminal real-name seal of chapter 38.

Definition 6519.1 (RegularCauchyCompletionComparisonTriangle carrier)**.** *A* RegularCauchyCompletionComparis *carrier is a finite* BHist *packet*

$$T = (M, L, C, G, S, R, D, E, H, Q, P, N)$$

with the following visible rows:

1. *M is the* CauchyModulus$^{\uparrow}$ *threshold ledger consumed by the regular-Cauchy input;*
2. *L is the* RegularCauchyLimitModulus$^{\uparrow}$ *diagonal limit-modulus row;*
3. *C is the* CauchyCompletionComparison$^{\uparrow}$ *comparison leg from Cauchy data to completion-facing data;*
4. *G is the* CompletionComparison$^{\uparrow}$ *comparison leg that checks the same observation against the ambient completion interface;*
5. *S is the* StreamName$^{\uparrow}$ *finite window through which the selected diagonal or tail observation is inspected;*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the same window;*
7. *D is the* DyadicRatCore$^{\uparrow}$ *ledger recording the displayed tolerance, endpoint, comparison, and precision data;*
8. *E is the* Real$^{\uparrow}$ *seal row available only after M, L, C, G, S, R, D have been displayed;*
9. *H records componentwise* hsame *transport for the triangle rows;*
10. *Q records displayed* Cont *replay for the three comparison legs;*

11. *P is the* Pkg *provenance over the sibling routes named above;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6519.2 (RegularCauchyCompletionComparisonTriangle NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}}$ *surface consists of a finite route-exhaustion statement: every accepted triangle packet first exposes the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold ledger, then the* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *diagonal selector, then the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *and* $\mathsf{CompletionComparison}^{\uparrow}$ *comparison legs, and only then the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer rows. The same componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows transport the whole triangle.*

Consequently a Real or Completion consumer cannot bypass the regular-Cauchy leg by reading E as a host real, cannot bypass the completion-comparison leg by treating C as quotient stream equality, and cannot bypass the dyadic leg by replacing D with an ambient completeness principle.

Proof. Project the carrier of Definition 6519.1. The rows $M, L, C, G, S, R, D, E, H, Q, P, N$ are all coordinates of one finite packet, so any consumer route must begin by choosing one of those displayed coordinates.

The regular-Cauchy leg is exhausted by M, L, S, R, and D. The threshold ledger supplies the modulus row, the limit-modulus row supplies the diagonal selector, the stream row supplies the finite window, the regular rational row supplies readback, and the dyadic row records the tolerance arithmetic.

The completion leg is exhausted by C and G. These rows compare the same displayed observation against the Cauchy-completion and completion-comparison interfaces; they do not add a quotient representative or a selected completion point.

Finally E is downstream of those rows. The transport, replay, provenance, and naming coordinates H, Q, P, N preserve the same triangle, so route exhaustion leaves no coordinate for quotient stream equality, countable choice, trichotomy, host selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}$ consumer that reaches a Real or Completion handoff without first exposing the CauchyModulus, RegularCauchyLimitModulus, CauchyCompletionComparison, CompletionComparison, StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and naming rows refutes the carrier surface.

Independence witness: cauchymodulus, regular-cauchy-limit-modulus, cauchy-completion-comparison, completion-comparison, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective to $\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}$ because this packet binds the modulus leg, diagonal-selector leg, two completion-comparison legs, finite stream window, regular rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming into one triangle.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}$ packet is generated from a CauchyModulus threshold ledger, a RegularCauchyLimitModulus diagonal selector, CauchyCompletionComparison and CompletionComparison legs, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionComparisonTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6519.1 and Definition 6519.2 seed the finite comparison triangle linking regular-Cauchy, completion-comparison, and RealUp seal routes.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionComparisonTriangle}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked route exhaustion for BEDC.Derived.RegularCauchyCompletionComparisonTriangleUp over the displayed CauchyModulus, RegularCauchyLimitModulus, CauchyCompletionComparison, CompletionComparison, StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Chapter 6520

A Concrete Naming Certificate for $\mathsf{SequentialFan}^{\uparrow}$

Following Brouwer and Bishop–Bridges on fans and choice sequences, the $\mathsf{SequentialFan}^{\uparrow}$ packet records the finite fan surface needed by Real and Completion readers as BEDC naming data. It sits beside the bar discipline of chapter 400, the binary prefix source of chapter 1475, the finite-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names only finite prefix trees, displayed bar witnesses, finite stream windows, regular readbacks, Real seals, transport, replay, provenance, and local naming; it does not assert open-cover compactness, a selected branch, countable choice, host quotient equality, or ambient completeness.

Definition 6520.1 (SequentialFan carrier)**.** *A* $\mathsf{SequentialFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is a finite prefix-tree row for a fan, read through the binary-prefix and fan surfaces of chapter 1475 and chapter 400.*
2. *B is a displayed bar witness over T, recording the finite prefix boundary at which every accepted branch read stops.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to the prefixes selected by B.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, B, W, R have been displayed.*
6. *H records componentwise* hsame *transport for T, B, W, R, E, C, P, N.*
7. *C records displayed* Cont *replay from the prefix-tree row through the bar witness, finite stream windows, RegSeqRat readback, Real seal, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{SequentialFan}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialFan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, W, R, E, H, C, P, N$. It has no coordinate for an infinite branch selector, open-cover compactness, quotient stream equality, host real equality, selected limits, countable choice, or ambient completeness.

Theorem 6520.2 (SequentialFan NameCert obligations)**.** *For every accepted* $\mathsf{SequentialFan}^{\uparrow}$ *carrier*

$$S = (T, B, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialFan}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix tree, bar witness, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming;*
2. *fan-prefix compatibility, requiring every inspected branch request to enter through the displayed finite tree* T *and the bar boundary* B*;*
3. *stream-window exposure, requiring each Real-facing observation to read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W *before any regular rational readback is consumed;*
4. *regular-readback exactness, requiring* R *to be the only row by which the stream windows are interpreted as* $\mathsf{RegSeqRat}^{\uparrow}$ *data;*
5. *terminal-seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row* E *to be exposed only after* T, B, W, R, H, C, P, N *have been displayed and excluding selected branches, quotient streams, host real equality, selected limits, countable choice, ambient completeness, Lean verification, and bridge checking.*

No SequentialFan obligation is stored outside these rows.

Proof. Project the accepted carrier. The fan data are T and B, the finite analytic observations are W and R, and the terminal naming row is E.

The prefix compatibility obligation is forced because the carrier has no second tree coordinate and no branch selector. A branch-facing request therefore reaches an observation only after the displayed bar row has bounded the prefix length.

The analytic handoff is then finite. The $\mathsf{StreamName}^{\uparrow}$ row W supplies the windows, and R supplies the regular rational readback for exactly those windows. Transport H and replay C preserve the same rows before provenance and local naming are read.

Since E is the only Real-facing coordinate and the packet has no slot for quotient streams, selected limits, countable choice, ambient completeness, Lean verification, or bridge data, the local NameCert surface is exactly the finite list stated above. □

Theorem 6520.3 (SequentialFan Cauchy-window boundary)**.** *Every Cauchy-facing or Real-facing read accepted by a* $\mathsf{SequentialFan}^{\uparrow}$ *packet factors through the finite fan-bar window before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Equivalently, the visible route is*

$$\mathsf{FanTheorem}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow} \longmapsto T, B \longmapsto \mathsf{StreamName}^{\uparrow}(W) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(R) \longmapsto \mathsf{Real}^{\uparrow}(E)$$

with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. A read that bypasses the finite bar window, inserts a selected branch, or distinguishes terminal seals by quotient stream equality is outside the accepted* $\mathsf{SequentialFan}^{\uparrow}$ *carrier.*

Proof. Begin with the carrier of Definition 6520.1. The only prefix data are the finite tree row T and the bar witness B; hence a Cauchy-facing request cannot inspect an unbounded branch inside the packet.

The finite bar boundary determines which StreamName windows are visible. Those windows are recorded in W, and the regular rational readback row R is attached to the same displayed windows.

The terminal Real seal E is read only after T, B, W, R and the support rows H, C, P, N have been displayed. Therefore every accepted Cauchy or Real read follows the stated route through the finite fan-bar window before the RegSeqRat and Real rows are exposed.

The exclusion clause follows from the missing carrier coordinates: there is no selected branch, quotient stream equality, selected limit, countable-choice row, ambient completeness row, or alternate Real seal through which a consumer could bypass the finite window boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialFan}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialFan}^{\uparrow}$ packet is generated from a finite fan prefix tree, a displayed bar witness, StreamName finite-window rows, RegSeqRat readback rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialFan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6520.1, Definition 6520.2, and Definition 6520.3 seed the finite fan-bar window surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialFan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected infinite branches, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-prefix compatibility, stream-window exposure, regular-readback exactness, and terminal-seal nonescape for BEDC.Derived.SequentialFanUp.

Chapter 6521

A Concrete Naming Certificate for $\mathsf{FrechetUrysohnFan}^{\uparrow}$

Following Kelley and Engelking on Frechet–Urysohn spaces and sequential fans, the $\mathsf{FrechetUrysohnFan}^{\uparrow}$ packet records the finite fan-indexed closure surface by which a sequential fan supplies a displayed Frechet–Urysohn closure handoff. It sits adjacent to the finite fan-bar carrier of chapter 6520, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed fan arms, finite stream windows, regular rational tails, Real seals, closure witnesses, transport, replay, provenance, and local naming; it does not assert arbitrary topological nets, a selected branch through the fan, countable choice, host quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6521.1 (Frechet-Urysohn fan carrier)**.** *A* $\mathsf{FrechetUrysohnFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, W, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{SequentialFan}^{\uparrow}$ *arm row, recording the finite fan-bar surface and the displayed arm on which the closure read is made.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected arm segments.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-readback row attached to those windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, W, R have been displayed.*
5. *L is the Frechet–Urysohn closure-witness row, recording that the named limit point is approached along the displayed fan arm through the same windows and tails.*
6. *H records componentwise* hsame *transport for arm, window, readback, Real-seal, closure-witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a fan-closure request through A, W, R, E, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohnFan}^{\uparrow}$, $\mathsf{SequentialFan}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohnFan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, R, E, L, H, C, P, N$. It has no coordinate for arbitrary nets, selected fan branches, quotient stream equality, host real equality, countable choice, or ambient completeness.

Theorem 6521.2 (Frechet-Urysohn fan NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohnFan}^{\uparrow}$ *carrier*

$$F = (A, W, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetUrysohnFan}^{\uparrow}}$ *surface consists exactly of sequential-fan arm admission through* A*, StreamName window exposure through* W*, RegSeqRat tail readback through* R*, terminal Real-seal ordering through* E*, Frechet–Urysohn closure-witness display through* L*, componentwise transport through* H*, displayed continuation replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The fan-facing data are the arm row A and the finite windows W, while the analytic readback is carried by R and then by the terminal Real seal E.

The closure witness is row-local. The row L can certify only the displayed arm and the windows already exposed by A, W, R, E; it has no coordinate for an arbitrary net, a selected branch outside the arm row, or a host closure operator.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no hidden branch selector, quotient stream equality, host real equality, countable-choice row, or ambient completeness row, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6521.3 (Frechet-Urysohn fan stream-closure handoff)**.** *Every consumer that reads a* $\mathsf{FrechetUrysohnFan}^{\uparrow}$ *packet as a sequential-closure witness must factor through*

$$\mathsf{SequentialFan}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the Frechet–Urysohn closure row L *is visible. The handoff exports the displayed fan arm, finite stream windows, regular rational tails, terminal Real seal, and closure witness only; it does not export arbitrary nets, selected branches, quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6521.2. A closure-facing consumer first reads the sequential-fan arm row A, and this row is the only point at which the packet can consume the fan surface of chapter 6520.

The finite StreamName windows W and RegSeqRat readback R are downstream of that same displayed arm. The Real seal E is downstream of both, so the closure row L cannot be read before the route has passed through chapter 126, chapter 301, and chapter 38.

Thus the consumer receives a finite stream-closure handoff, not an ambient topological closure theorem. The missing carrier coordinates exclude arbitrary nets, selected branches, quotient stream equality, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the exported surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohnFan}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrechetUrysohnFan}^{\uparrow}$ packet is generated from a SequentialFan arm row, StreamName finite-window rows, RegSeqRat tail readbacks, a terminal Real seal, a Frechet–Urysohn closure-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetUrysohnFan}^{\uparrow}}$, seed)

Scope closed: Definition 6521.1, Definition 6521.2, and Definition 6521.3 seed the fan-indexed sequential-closure carrier over SequentialFan, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetUrysohnFan}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary nets, selected fan branches, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked arm admission, finite-window exposure, tail readback, Real-seal ordering, closure-witness display, transport, replay, provenance, and local naming for BEDC.Derived.FrechetUrysohnFanUp.

Chapter 6522

A Concrete Naming Certificate for BishopFanCompactUniform$^{\uparrow}$

Following Bishop–Bridges finite fan and compact-uniform-continuity arguments, the BishopFanCompactUniform$^{\uparrow}$ packet realizes the compact-uniform-continuity vision of chapter 9237 as finite NameCert data. It sits between the compact-uniform-continuity carrier of chapter 388, the bar discipline of chapter 400, the compact metric source rows of chapter 129, and the continuous-map rows of chapter 130. The packet records finite fan bars, compact finite observations, pointwise modulus rows, uniform-modulus extraction, transport, replay, provenance, and local naming; it does not assert open-cover compactness, a selected infinite branch, countable choice, quotient map equality, or ambient completeness.

Definition 6522.1 (BishopFanCompactUniform carrier)**.** *A* BishopFanCompactUniform$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, B, L, U, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite observation row, exposing only the compact source windows used by the uniform-continuity request.*
2. *M is the* ContinuousMap$^{\uparrow}$ *row whose pointwise modulus entries are being compared.*
3. *B is the finite* FanTheorem$^{\uparrow}$ *bar row, recording a prefix boundary that covers all displayed compact-source cases.*
4. *L is the local pointwise-modulus ledger read along the bar leaves.*
5. *U is the uniform-modulus handoff row consumed by* CompactUniformContinuity$^{\uparrow}$ *after K, M, B, L have been displayed.*
6. *D is the dyadic tolerance and rational lower-bound ledger used to compare the local moduli and the output tolerance.*
7. *H records componentwise* hsame *transport for $K, M, B, L, U, D, C, P, N$.*
8. *C records displayed* Cont *replay from compact-source rows and continuous-map rows through the fan bar, local modulus ledger, dyadic comparison, uniform-modulus handoff, provenance, and local naming.*
9. *P is the* Pkg *provenance over* BishopFanCompactUniform$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FanTheorem$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFanCompactUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, B, L, U, D, H, C, P, N$. It has no coordinate for an arbitrary open cover, global branch selector, selected external Lebesgue number, quotient map equality, countable choice, or ambient compactness theorem.

Theorem 6522.2 (BishopFanCompactUniform NameCert obligations)**.** *For every accepted* BishopFanCompactUnif *carrier*

$$F = (K, M, B, L, U, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopFanCompactUniform}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for compact-source observations, continuous-map rows, finite fan-bar data, local pointwise moduli, uniform-modulus handoff, dyadic tolerance comparison, transport, replay, provenance, and local naming;*
2. *compact-source exposure, requiring every source observation to enter through K before any modulus row is read;*
3. *continuous-map exposure, requiring the pointwise modulus ledger L to be tied to the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *row M;*
4. *fan-bar coverage, requiring the finite bar B to bound all accepted compact-source cases before U is exposed;*
5. *modulus-route exactness, requiring U to be produced only from K, M, B, L, D and excluding arbitrary open covers, selected branches, selected external radii, quotient map equality, countable choice, Lean verification, and bridge checking.*

No BishopFanCompactUniform obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BishopFanCompactUniformUp.TasteGate.BishopFanCompactUniformTasteGate single carrier alignment`

Proof. Project the accepted carrier. The compact source and map data are K and M, the fan boundary is B, the local modulus ledger is L, and the output handoff is U.

The source and map exposure obligations are forced because the carrier has no alternate compact-space or function coordinate. Every local modulus row is therefore read as a row over the displayed K, M boundary.

The finite bar row B supplies the coverage boundary. The dyadic ledger D compares the finitely many local rows in L and records the tolerance handoff consumed by U.

Transport H and replay C preserve the same finite route before provenance and local naming. Since the carrier has no row for arbitrary open covers, branch selectors, external radius selection, quotient map equality, choice, Lean verification, or bridge data, the stated obligations exhaust the local NameCert surface. □

Theorem 6522.3 (BishopFanCompactUniform modulus route)**.** *Every uniform-modulus read accepted by a* $\mathsf{BishopFanCompactUniform}^{\uparrow}$ *packet factors through the finite Bishop fan route*

$$\mathsf{CompactMetric}^{\uparrow}(K),\ \mathsf{ContinuousMap}^{\uparrow}(M) \longmapsto \mathsf{FanTheorem}^{\uparrow}(B) \longmapsto L, D \longmapsto \mathsf{CompactUniformContinuity}^{\uparrow}(U)$$

with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support rows. A consumer cannot obtain a uniform modulus from an open-cover compactness oracle, a selected branch, an external Lebesgue number, quotient map equality, countable choice, or ambient compactness data.*

Proof. Start with an accepted uniform-modulus read. By Definition 6522.1, the compact source row K and continuous-map row M are the only rows from which source observations and pointwise moduli can be read.

The finite fan-bar row B bounds the displayed cases. The local ledger L is then a finite list over the bar leaves, and the dyadic comparison row D folds those entries into the output handoff U.

The structural rows H, C, P, N keep the same finite carrier under transport, continuation replay, provenance, and naming. Thus every accepted read follows the displayed Bishop fan route before $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumes the uniform-modulus row.

The excluded objects have no carrier coordinate, so they cannot be inserted between the compact-source and uniform-modulus faces of the packet. □

Theorem 6522.4 (BishopFanCompactUniform nonescape)**.** *Every accepted* $\mathsf{BishopFanCompactUniform}^{\uparrow}$ *route to a* $\mathsf{UniformModulus}^{\uparrow}$ *handoff must first display the finite fan-bar modulus boundary of chapter 4297, the compact metric source rows of chapter 129, and the continuous-map pointwise rows of chapter 130. The handoff is therefore a finite-row route through* $K, M, B, L, U, D, H, C, P, N$*, not an open-cover compactness claim, selected branch, external Lebesgue-number choice, quotient equality, countable-choice principle, or ambient completeness theorem.*

Proof. Read the accepted carrier in Definition 6522.1. The compact source row is K, the continuous-map row is M, and the finite fan boundary is B.

Apply Definition 6522.3. It places every uniform-modulus read downstream of the compact metric source, continuous-map pointwise ledger, fan-bar coverage row, local modulus ledger, and dyadic comparison row.

The named sibling chapters supply exactly those public surfaces: finite fan-bar modulus, compact metric source data, and continuous-map pointwise rows. Since no coordinate stores open-cover compactness, branch selection, external radius selection, quotient equality, countable choice, or ambient completeness, the $\mathsf{UniformModulus}^{\uparrow}$ handoff cannot escape the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFanCompactUniform}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFanCompactUniform}^{\uparrow}$ packet is generated from compact metric source observations, continuous-map pointwise modulus rows, a finite fan-bar row, local modulus ledgers, a uniform-modulus handoff, dyadic tolerance comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFanCompactUniform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6522.1, Definition 6522.2, Definition 6522.3, and Definition 6522.4 seed the finite Bishop fan route for compact-uniform-modulus extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFanCompactUniform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected infinite branches, selected external Lebesgue numbers, quotient map equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-map exposure, fan-bar coverage, modulus-route exactness, and nonescape for BEDC.Derived.BishopFanCompactUniformUp.

Chapter 6523

A Concrete Naming Certificate for $\mathsf{PolishCompletion}^{\uparrow}$

Following Alexandrov–Urysohn and the Kechris presentation of Polish spaces as separable completely metrizable spaces, the $\mathsf{PolishCompletion}^{\uparrow}$ packet records the finite BEDC route by which a separable metric presentation is completed into a Polish-space-facing carrier. It sits after the Polish-space surface of chapter 1490, the separable completion route of chapter 1714, and the Bishop metric completion packet of chapter 4023. The construction also consumes the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route used by Real and regular-rational completion readers; it does not quotient Cauchy names, select representatives, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness outside the displayed completion packet.

Definition 6523.1 (Polish completion carrier)**.** *A* $\mathsf{PolishCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, D, M, S, B, W, R, Q, E, H, C, K, N)$$

with the following displayed rows:

1. *X is the separable metric source row whose dense naming surface is inspected before completion.*
2. *D is the countable dense-window ledger read as finite* $\mathsf{StreamName}^{\uparrow}$ *observations, not as a completed choice sequence.*
3. *M is the* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *row carrying the constructive metric-completion handoff.*
4. *S is the* $\mathsf{SeparableCompletion}^{\uparrow}$ *row binding the dense source to the completion route.*
5. *B is the* $\mathsf{PolishSpace}^{\uparrow}$ *target boundary row, read only after the separable and complete-metric rows have been displayed.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule used by completion-facing readers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the completion and readback rows.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after W, R, Q, M, S, B have been displayed.*
10. *H records componentwise* hsame *transport for the source, dense-window, completion, Polish boundary, readback, tolerance, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from source inspection through dense windows, Bishop completion, separable completion, Polish boundary, regular readback, tolerance, and Real sealing.*

12. *K is the* Pkg *provenance over* $\mathsf{PolishCompletion}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{SeparableCompletion}^{\uparrow}$, $\mathsf{BishopMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient Cauchy equality, selected completion representatives, a countable-choice dense sequence, trichotomy, host equality of real names, or ambient completeness outside the named packet.

Theorem 6523.2 (Polish completion NameCert obligations)**.** *For every accepted* $\mathsf{PolishCompletion}^{\uparrow}$ *carrier* $P = (X, D, M, S, B, W, R, Q, E, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PolishCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the separable metric source, finite dense-window ledger, Bishop metric completion handoff, separable completion row, Polish-space boundary, StreamName window schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows.*

No NameCert obligation is stored in a quotient of Cauchy names, a selected representative, a host dense sequence, a countable-choice schedule, trichotomy, host real equality, or an ambient completeness theorem.

Proof. Project the carrier of Definition 6523.1. The accepted packet exposes exactly the rows $X, D, M, S, B, W, R, Q, E, H, C, K, N$.

The source and dense-window rows are finite. The row X supplies the separable metric source, while D exposes only inspected dense windows through the same finite schedule discipline used by $\mathsf{StreamName}^{\uparrow}$.

The completion-facing rows are downstream. The Bishop metric completion row M and separable completion row S are read before the Polish boundary B, and the window, readback, dyadic tolerance, and Real seal rows W, R, Q, E are visible before any public completion read is exported.

Finally H, C, K, N transport, replay, source, and name the same packet. Since none of the excluded host or choice principles appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6523.3 (Polish completion complete separable route)**.** *Every Polish-space-facing read of an accepted* $\mathsf{PolishCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$X, D, W, R, Q, M, S, B, E, H, C, K, N.$$

The route first displays the separable metric source and dense-window ledger, then reads finite StreamName windows, RegSeqRat readback, and DyadicRatCore tolerances, then passes through Bishop metric completion and separable completion before the Polish boundary and terminal Real seal are exposed. It exports a public BEDC completion route, not a host theorem asserting complete metrizability, a quotient completion class, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations above. They force every consumer to expose the separable metric source, dense-window ledger, completion handoffs, window schedule, regular readback, tolerance ledger, Real seal, transport, replay, provenance, and local name.

Use the sibling completion route. The $\mathsf{BishopMetricCompletion}^{\uparrow}$ row supplies the constructive complete-metric handoff, while the $\mathsf{SeparableCompletion}^{\uparrow}$ row keeps the dense source visible before the $\mathsf{PolishSpace}^{\uparrow}$ boundary is read.

Finally follow the Real and regular-rational support rows. The finite window W, readback R, tolerance ledger Q, and seal E keep the public route inside displayed BEDC data. Because the

classifier has no coordinate for quotient classes, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the complete separable route is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishCompletion}^{\uparrow}$ packet is generated from a separable metric source row, dense-window ledger, Bishop metric completion row, separable completion row, Polish-space boundary, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6523.1, Definition 6523.2, and Definition 6523.3 seed the finite Polish completion route over separable metric source data, Bishop metric completion, separable completion, Polish-space boundary, StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected representatives, countable-choice dense sequences, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-window exposure, Bishop metric completion handoff, separable completion handoff, Polish boundary exposure, regular readback, dyadic tolerance budgeting, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.PolishCompletionUp.

Chapter 6524

A Concrete Naming Certificate for $\mathsf{SteffensenAcceleration}^{\uparrow}$

Following Steffensen and Aitken, the $\mathsf{SteffensenAcceleration}^{\uparrow}$ packet records a fixed-point acceleration window by naming the local delta-squared transform of consecutive iterates before any real-number seal is read. It sits beside the $\mathsf{AitkenDeltaSquared}^{\uparrow}$ acceleration certificate of chapter 1523, the Newton step surface of chapter 223, the fixed-point iterate schedule of chapter 2383, the stream row of chapter 126, the regular rational route of chapter 301, and the final real-name seal of chapter 38. The chapter names a finite numerical-analysis carrier; it does not assert host fixed-point equality, quotient real equality, convergence by countable choice, or an ambient completeness theorem.

Definition 6524.1 (Steffensen acceleration carrier)**.** *A* $\mathsf{SteffensenAcceleration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, A, N, F, W, R, E, H, C, P, L)$$

with the following rows:

1. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *iterate-window row, exposing three consecutive inspected approximants and the displayed fixed-point map row;*
2. *A is the* $\mathsf{AitkenDeltaSquared}^{\uparrow}$ *delta-squared transform row applied to that finite window;*
3. *N is the optional* $\mathsf{NewtonIteration}^{\uparrow}$ *neighbour row when the displayed fixed-point map is read through a Newton-style residual update;*
4. *F records the finite denominator-apartness and fallback ledger needed to keep the transformed window constructive;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the inspected iterate depths;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational acceleration values;*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal reached only after R is displayed;*
8. *H records componentwise* hsame *transport of I, A, N, F, W, R, E;*
9. *C records displayed* Cont *replay of the same finite acceleration route;*
10. *P is the* Pkg *provenance over* $\mathsf{SteffensenAcceleration}^{\uparrow}$*,* $\mathsf{AitkenDeltaSquared}^{\uparrow}$*,* $\mathsf{NewtonIteration}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*

11. L is the local $\mathsf{NameCert}_{\mathsf{SteffensenAcceleration}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient fixed point, an infinite convergence proof, quotient equality of real names, or a choice-selected subsequence of iterates.

Theorem 6524.2 (Steffensen finite-window NameCert obligations)**.** *For the carrier of Definition 6524.1, the local* $\mathsf{NameCert}_{\mathsf{SteffensenAcceleration}^{\uparrow}}$ *surface consists exactly of carrier admission, fixed-point iterate-window admission through I, Aitken delta-squared transform admission through A, Newton-neighbour admission through N when present, denominator-apartness and fallback admission through F, stream schedule admission through W, regular rational readback through R, downstream* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the carrier $S = (I, A, N, F, W, R, E, H, C, P, L)$. The finite iterate row I supplies only the inspected fixed-point window, while A records the Aitken delta-squared calculation on that same window.

The rows N and F restrict the numerical read to displayed Newton-neighbour and denominator-apartness data. If the denominator row is not available, the fallback ledger remains part of the named packet rather than being replaced by a host convergence claim.

Finally W, R, E, H, C, P, L schedule, read back, seal, transport, replay, source, and name the same finite route. Since no other coordinate appears in the carrier, the NameCert surface cannot export host fixed-point equality, quotient real equality, countable choice, or ambient completeness. □

Theorem 6524.3 (Steffensen acceleration Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SteffensenAcceleration}^{\uparrow}$ *packet must pass through the finite route*

$$I, A, N, F, W, R, E, H, C, P, L.$$

The consumer receives an inspected fixed-point iterate window, the Aitken delta-squared acceleration row, any displayed Newton-neighbour residual row, the denominator-apartness or fallback ledger, the stream schedule, and the $\mathsf{RegSeqRat}^{\uparrow}$ *readback before it may read the* $\mathsf{Real}^{\uparrow}$ *seal. It receives no host fixed point, no quotient equality of real names, no hidden convergence principle, and no countable-choice schedule.*

Proof. Begin with the accepted carrier of Definition 6524.1. The iterate row I and transform row A determine the acceleration window before any real-facing data are visible.

Next read the numerical guard rows. The Newton-neighbour row N is optional packet data, and the denominator row F records the constructive apartness or fallback route needed by the delta-squared transform.

The stream and rational rows W, R then present the accelerated values as finite readback data. Only after those rows are displayed does E seal the route as a $\mathsf{Real}^{\uparrow}$ consumer surface, while H, C, P, L preserve the same packet under transport, replay, provenance, and naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{SteffensenAcceleration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SteffensenAcceleration}^{\uparrow}$ packet is generated from a fixed-point iterate window, an Aitken delta-squared transform row, an optional Newton-neighbour row, a denominator-apartness or fallback ledger, a stream schedule, regular rational readback, a downstream real-name seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SteffensenAcceleration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6524.1, Definition 6524.2, and Definition 6524.3 specify the finite Steffensen acceleration packet and its route to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{SteffensenAcceleration}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Lean verification, a standard bridge, host fixed-point equality, quotient real equality, a general convergence theorem, countable choice, or ambient completeness of Real$^{\uparrow}$.
Upgrade path: Obligation closure requires Lean encoding of the finite carrier, transform guard, readback route, and Real-facing non-escape theorem.

Chapter 6525

A Concrete Naming Certificate for $\mathsf{ThomaeFunction}^{\uparrow}$

Following Thomae and Bishop–Bridges constructive analysis, the $\mathsf{ThomaeFunction}^{\uparrow}$ packet records the finite rational-spike data by which the popcorn-function example is read through rational density, denominator ledgers, regular rational windows, and the terminal Real seal. It consumes the rational-density input surface of chapter 1409, the $\mathsf{Rat}^{\uparrow}$ denominator arithmetic of chapter 37, the finite windows and readback rows of chapter 126 and chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the continuity-facing boundary of chapter 130; these sibling rows cannot be replaced by host function extensionality, selected rational enumeration, quotient real equality, or classical discontinuity sets. The packet names the constructive rational-spike function as finite BEDC data; it is not a host function, a quotient-real extensionality principle, a global continuity theorem, a selected enumeration of rationals, or a classical discontinuity set.

Definition 6525.1 (Thomae function carrier)**.** *A* $\mathsf{ThomaeFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational-density input row, recording the finite rational sample by which the spike query is opened.*
2. *D is the denominator spike ledger, recording the denominator bound and the spike height assigned to the visible rational input.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the finite name data around the input.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from that window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row that receives the rational-spike value as a Real-facing output.*
6. *H records componentwise* hsame *transport for rational input, denominator ledger, stream window, regular-rational readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a rational-density input through Q, D, S, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{ThomaeFunction}^{\uparrow}$*, RationalDensityRealUp,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ThomaeFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, S, R, E, H, C, P, N$. It has no coordinate for host function extensionality, quotient real equality, selected rational enumerations, global continuity, or a classical discontinuity set.

Theorem 6525.2 (Thomae function NameCert obligations)**.** *For every accepted* $\mathsf{ThomaeFunction}^{\uparrow}$ *carrier*

$$T = (Q, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ThomaeFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, rational-density input, denominator spike ledger stability, StreamName window stability, regular-rational readback, Real-facing seal stability, componentwise transport, displayed replay, provenance, and ledger non-escape through* $Q, D, S, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ThomaeFunctionUp.ThomaeFunction namecert obligations`

Proof. Project the carrier of Definition 6525.1. The rational input row Q, denominator ledger D, stream window S, regular-rational readback R, Real seal E, and structural rows H, C, P, N are the whole packet.

Rational-density input and denominator spike stability are read from Q, D. These rows record the finite input and spike height before any Real-facing consumer can inspect the output.

Readback stability is supplied by S, R, E. The stream window exposes the finite observations, the regular rational row replays them as readback data, and the terminal Real row seals the same finite route.

Finally H, C, P, N transport, replay, source, and name the displayed rows. Since no further coordinate is present, host function extensionality, quotient real equality, selected rational enumerations, global continuity, and classical discontinuity sets remain outside the ledger. □

Theorem 6525.3 (Thomae function rational-spike locality)**.** *Every rational-spike read of an accepted* $\mathsf{ThomaeFunction}^{\uparrow}$ *carrier factors through RationalDensityRealUp,* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it exposes a value to a* $\mathsf{ContinuousMap}^{\uparrow}$*-facing consumer. The read may display the local denominator spike and the finite Real seal, but it cannot use host function extensionality, quotient real equality, a selected rational enumeration, or a global continuity theorem.*

Proof. Let

$$T = (Q, D, S, R, E, H, C, P, N)$$

be an accepted carrier. The rational-density row Q opens the finite input request, and the denominator ledger D determines the visible spike height using only rational arithmetic recorded in the packet.

The stream and regular-rational rows S, R then expose the finite window and readback route. The Real seal E is downstream of the same rows, so a Real-facing consumer receives the spike value only after the denominator ledger and readback route have both been displayed.

The continuity-facing boundary can read the result only through the structural rows H, C, P, N. Therefore the route is local to the displayed rational-density, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data, and the excluded host and quotient principles never enter the packet. □

Theorem 6525.4 (Thomae function rational denominator route)**.** *For every accepted* $\mathsf{ThomaeFunction}^{\uparrow}$ *carrier*

$$T = (Q, D, S, R, E, H, C, P, N),$$

each rational-spike value is routed through the $\mathsf{Rat}^{\uparrow}$ *denominator ledger* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before any continuity-facing consumer may inspect oscillation. The route is finite BEDC denominator arithmetic and readback data; it is not a selected rational enumeration, quotient-real equality, host function extensionality, or a classical discontinuity-set argument.*

Proof. Project the carrier rows from Definition 6525.1. The rational-density row Q admits the displayed rational input, while the denominator ledger D records the denominator bound and spike height as the only rational arithmetic available to the packet.

Apply the NameCert obligations of Definition 6525.2. They force denominator spike ledger stability, StreamName window stability, regular-rational readback, Real-facing seal stability, transport, replay, provenance, and ledger non-escape to be read from the same rows D, S, R, E, H, C, P, N.

Use the rational-spike locality theorem, Definition 6525.3, for the consumer boundary. A continuity-facing read can inspect oscillation only after the denominator ledger has passed through the finite window, regular readback, and terminal Real seal.

Since the carrier has no coordinate for a selected rational enumeration, quotient-real equality, host function extensionality, or a classical discontinuity set, the denominator route is exhausted by the displayed $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ThomaeFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{ThomaeFunction}^{\uparrow}$ packet is generated from a rational-density input row, denominator spike ledger, StreamName window, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ThomaeFunction}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6525.1, Definition 6525.2, Definition 6525.3, and Definition 6525.4 bind the finite rational-spike route from rational input through denominator arithmetic to Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ThomaeFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient real equality, selected rational enumeration, global continuity, classical discontinuity sets, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires downstream continuity-facing consumers to preserve the displayed denominator ledger, finite window, regular readback, and Real seal route.

Chapter 6526

A Concrete Naming Certificate for $\mathsf{RegularSequenceApartness}^{\uparrow}$

Bishop-style constructive analysis treats apartness of real names as positive information: two regular Cauchy presentations are apart only when a displayed rational tolerance separates inspected approximants. The $\mathsf{RegularSequenceApartness}^{\uparrow}$ packet records that positive route as finite BEDC data between the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the located-apartness bridge of chapter 3564. It is a constructive positive-separation interface for L10 Real and Completion consumers, not quotient real equality, real trichotomy, a hidden representative selector, or ambient completeness.

Definition 6526.1 (Regular sequence apartness carrier)**.** *A* $\mathsf{RegularSequenceApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S_0, S_1, R_0, R_1, D, W_0, W_1, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened for the compared regular Cauchy real names.*
2. *R_0 and R_1 are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed through S_0 and S_1.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-tolerance witness. It records a dyadic $\varepsilon > 0$, the inspected stages, and the displayed inequality asserting that the two rational approximants differ by at least that tolerance after the regular slack budget is accounted for.*
4. *W_0 and W_1 are the finite window records that bind the displayed tolerance in D to the two readback rows.*
5. *E_0 and E_1 are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows for the two names, available only after $S_0, S_1, R_0, R_1, D, W_0, W_1$ have been displayed.*
6. *H records componentwise* hsame *transport for the two stream windows, two regular-rational readbacks, dyadic witness, window records, Real seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D, W_0, W_1 \longmapsto E_0, E_1.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceApartness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceApartness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, trichotomy, selected representatives, hidden completion limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6526.2 (Regular sequence apartness NameCert obligations)**.** *For every accepted* RegularSequenceApartne *carrier $A = (S_0, S_1, R_0, R_1, D, W_0, W_1, E_0, E_1, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceApartness}^{\uparrow}}$ *surface consists exactly of carrier admission, two* $\mathsf{StreamName}^{\uparrow}$ *window exposures, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-tolerance witness, two witness-window records, ordered* $\mathsf{Real}^{\uparrow}$ *seal access, classifier stability under the displayed rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape from that route. No NameCert obligation may replace the positive witness by quotient real equality, real trichotomy, selected representatives, hidden completion limits, countable choice, or ambient completeness.*

Proof. Project the accepted packet. The carrier exposes exactly two finite stream windows, two regular-rational readbacks, the dyadic positive-tolerance witness, two witness-window records, the terminal Real seals, and the structural rows H, C, P, N.

The witness row D is read before either terminal Real seal can be consumed. Its positivity is displayed inside the dyadic ledger and is attached to inspected regular-rational approximants through W_0 and W_1.

Classifier stability follows because the classifier has coordinates only for the listed rows. Componentwise hsame transport preserves the same displayed windows, witness, seals, replay, provenance, and local naming data.

The Cont row replays the route from finite windows to readbacks, then to the positive witness and witness windows, and only then to the Real seals. The package and local NameCert rows source and name that route.

Since the packet has no coordinate for quotient real equality, trichotomy, representative selection, hidden limits, countable choice, or ambient completeness, every accepted consumer remains inside the displayed positive-separation route. □

Theorem 6526.3 (Regular sequence apartness positive separation)**.** *Every positive-separation conclusion exported by an accepted* $\mathsf{RegularSequenceApartness}^{\uparrow}$ *packet is read from the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D together with the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R_0, R_1 and witness-window rows W_0, W_1. The conclusion does not depend on quotient real equality, real trichotomy, selected representatives, hidden completion limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier of Definition 6526.1. The only positive datum in the packet is the dyadic tolerance witness D, and D is attached to the two displayed readbacks by the window records W_0 and W_1.

Read R_0 and R_1 before reading D. The separation claim therefore refers to concrete rational approximants in the $\mathsf{RegSeqRat}^{\uparrow}$ rows, with the regular slack budget accounted for by the dyadic witness itself.

The terminal Real seals E_0 and E_1 are reached only after the positive witness route has already been displayed. They name the two real boundaries for downstream Real and Completion consumers, but they do not create the separation evidence.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Because no quotient, trichotomy, representative-selection, hidden-limit, choice, or completeness

coordinate appears in the carrier, such a coordinate cannot be the source of the positive-separation conclusion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceApartness}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularSequenceApartness}\uparrow}$, seed)

Scope closed: Definition 6526.1, Definition 6526.2, and Definition 6526.3 seed the finite constructive apartness route over displayed regular readbacks, dyadic positive tolerance, witness windows, and terminal Real seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularSequenceApartness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, located-order algebra, quotient real equality, real trichotomy, selected representatives, hidden completion limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, public closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired StreamName exposure, paired RegSeqRat readback, dyadic positive-tolerance exactness, witness-window exactness, ordered Real seal access, classifier stability, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularSequenceApartnessUp.

Constructive story: A $\mathsf{RegularSequenceApartness}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readback rows, a DyadicRatCore positive-tolerance witness, two witness-window rows, two RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6527

A Concrete Naming Certificate for $\mathsf{LocatedInverseBranch}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedInverseBranch}^{\uparrow}$ packet records a finite branch of a located inverse read after the located target interval, monotone window, dyadic tolerance ledger, finite stream observations, regular rational readback, and terminal real-name seal have all been displayed. It sits between the local inverse-function route of chapter 6001, the interval branch route of chapter 5546, the located real surface of chapter 1933, and the L10 real-name lattice of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names one displayed branch route for a located inverse request; it is not a host inverse-function theorem, a selected global branch, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6527.1 (Located inverse branch carrier)**.** *A* $\mathsf{LocatedInverseBranch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, J, M, V, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BishopLocatedInverseFunction}^{\uparrow}$ *row naming the local inverse-function packet whose map, located apartness bridge, and target request are being read.*
2. *I is the* $\mathsf{LocatedReal}^{\uparrow}$ *source-and-target interval row, carrying the located endpoint data and target window that delimit the inverse request.*
3. *J is the* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *branch row, recording the retained interval side and the finite branch history used to isolate the source window.*
4. *M is the monotone-window row. It records the finite order comparisons that keep the retained branch compatible with the local inverse request.*
5. *V is the branch-validity row, requiring the target window and retained source window to remain tied to the same located inverse-function data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing dyadic radii, midpoint budgets, comparison slack, and endpoint-readback conversions.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for source endpoints, target requests, function values, and retained branch endpoints.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows and dyadic tolerance ledger.*

9. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, I, J, M, V, D, S, Q have been displayed.*
10. *H records componentwise* hsame *transport for the inverse-function, located interval, bisection branch, monotone window, branch-validity, dyadic, stream, regular readback, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a located inverse-branch request through $F, I, J, M, V, D, S, Q, R, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedInverseBranch}^{\uparrow}$, $\mathsf{BishopLocatedInverseFunction}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedInverseBranch}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, J, M, V, D, S, Q, R, H, C, P, N$. It has no coordinate for a host inverse function, selected global branch, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

Theorem 6527.2 (Located inverse branch NameCert obligations)**.** *For every accepted* $\mathsf{LocatedInverseBranch}^{\uparrow}$ *carrier*

$$B = (F, I, J, M, V, D, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedInverseBranch}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, inverse-function admission through F, located interval admission through I, bisection-branch admission through J, monotone-window compatibility through M, branch-validity through V, dyadic tolerance accounting through D, finite-window exposure through S, regular rational handoff through Q, terminal Real seal discipline through R, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6527.1. The projection exposes exactly the rows $F, I, J, M, V, D, S, Q, R, H, C$ so carrier admission and row inventory are finite.

The branch request begins with F and I. The inverse-function row supplies the local map, apartness, uniqueness, and existence data, while the located interval row supplies the source and target windows that the branch may inspect.

The retained branch is then read through J, M, V. The bisection row records the finite branch history, the monotone-window row records the order comparisons used by that branch, and the validity row keeps the retained source window attached to the same target request.

Tolerance and readback are downstream. The dyadic ledger D supplies all radii, midpoint budgets, comparison slack, and conversions; the finite stream row S supplies the inspected values; and the regular rational row Q reads only those displayed windows before the Real seal R is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for a host inverse function, selected global branch, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 6527.3 (Located inverse branch monotone window)**.** *Every retained branch read in an accepted* $\mathsf{LocatedInverseBranch}^{\uparrow}$ *carrier*

$$B = (F, I, J, M, V, D, S, Q, R, H, C, P, N)$$

must expose the route

$$F, I \longmapsto J, M, V \longmapsto D, S, Q \longmapsto R, H, C, P, N.$$

Thus a monotone-window consumer receives the local inverse-function data, located interval data, displayed bisection branch, finite order comparisons, branch-validity witness, dyadic tolerance ledger, finite stream windows, regular rational readback, and structural rows before it can read the branch.

Proof. Start from a consumer of the retained branch. By the carrier definition, the only branch row is J, and J is attached to the local inverse-function row F and located interval row I.

The monotone information is not a host order theorem. It is the finite row M, read together with V so that the retained source window and target request remain on the same located inverse-function packet.

The consumer then reaches the dyadic, stream, and regular rational rows. The rows D, S, Q provide the tolerance budgets, finite inspected windows, and readback surface before the terminal seal R can be consumed.

The rows H, C, P, N preserve the same route under transport and replay. Therefore every monotone-window read factors through the stated branch route and exports none of the excluded host or choice principles. □

Theorem 6527.4 (Located inverse branch Real seal)**.** *No accepted* $\mathsf{LocatedInverseBranch}^{\uparrow}$ *packet can expose its* $\mathsf{Real}^{\uparrow}$ *seal row R before the* $\mathsf{BishopLocatedInverseFunction}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedRealIntervalBisection}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows of*

$$B = (F, I, J, M, V, D, S, Q, R, H, C, P, N)$$

have been displayed. The seal is a terminal L10 readback for the retained branch, not a selected inverse value, quotient representative, trichotomy decision, countable-choice product, or ambient completeness witness.

Proof. Apply the NameCert obligations of Definition 6527.2. They place the inverse-function, located interval, bisection branch, monotone-window, validity, dyadic, stream, and regular rational rows before the Real seal row.

Use the monotone-window theorem, Definition 6527.3. It orders every retained branch read through F, I, J, M, V, D, S, Q before R is reachable.

The structural rows H, C, P, N keep that order stable under transport, replay, provenance, and local naming. Since the carrier has no selected inverse value, quotient representative, trichotomy decision, countable-choice product, or ambient completeness coordinate, the terminal seal cannot be read as any of those objects. □

Falsifiable prediction: A consumer that claims a $\mathsf{LocatedInverseBranch}^{\uparrow}$ branch while omitting the local inverse-function row, located interval row, bisection branch row, monotone-window row, branch-validity row, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 6527.1.

Independence witness: bishop-located-inverse-function, located-real, located-real-interval-bisection, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedInverseBranch}^{\uparrow}$ binds a local inverse-function row, located interval row, retained bisection branch, monotone-window ledger, validity witness, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming in one branch certificate; each named sibling lacks at least one of these rows as a local branch coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedInverseBranch}^{\uparrow}}$.

Constructive story: A LocatedInverseBranch$^{\uparrow}$ packet is generated from a BishopLocatedInverseFunction$^{\uparrow}$ row, LocatedReal$^{\uparrow}$ interval data, a LocatedRealIntervalBisection$^{\uparrow}$ branch row, monotone-window comparisons, branch-validity evidence, DyadicRatCore$^{\uparrow}$ tolerances, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, a terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedInverseBranch}^{\uparrow}}$, seed)

Scope closed: Definition 6527.1, Definition 6527.2, Definition 6527.3, and Definition 6527.4 seed the finite retained-branch route for located inverse-function consumers.

Formal verification: FormalStatus(NameCert$_{\text{LocatedInverseBranch}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host inverse functions, selected global branches, quotient real equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-function admission, located interval admission, bisection-branch admission, monotone-window compatibility, branch-validity, dyadic tolerance accounting, finite-window exposure, regular rational handoff, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.LocatedInverseBranchUp.

Chapter 6528

A Concrete Naming Certificate for $\mathsf{StarRefinementModulus}^{\uparrow}$

Following Kelley and Bourbaki on star refinements of uniform covers, and Bishop–Bridges compact metric uniform-continuity arguments, the $\mathsf{StarRefinementModulus}^{\uparrow}$ packet records the finite step that turns a displayed star-refined compact cover into a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff. It sits after the compact metric source of chapter 129, the finite Lebesgue-number ledger of chapter 452, the metric triangle estimate of chapter 1159, and the uniform-modulus consumer of chapter 369. The packet names only finite cover-refinement, radius, center, estimate, and handoff rows; it does not assert open-cover compactness, choose an external Lebesgue number, form quotient equality of maps, use countable choice, or import ambient completeness.

Definition 6528.1 (StarRefinementModulus carrier)**.** *A* $\mathsf{StarRefinementModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, R, L, E, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover and finite-center row.*
2. *R is the star-refined cover row, recording that every displayed star lies inside a displayed member of the target cover.*
3. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger attached to the refined cover.*
4. *E is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *estimate row used to compare a point with a selected finite center and then with its image.*
5. *T is the dyadic tolerance row that ties the requested target tolerance to the radius ledger.*
6. *U is the resulting* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row.*
7. *H records componentwise* hsame *transport for $K, R, L, E, T, U, C, P, N$.*
8. *C records displayed* Cont *replay from a target tolerance request through the star-refinement, Lebesgue-radius, triangle-estimate, and uniform-modulus rows.*
9. *P is the* Pkg *provenance over* $\mathsf{StarRefinementModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StarRefinementModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, R, L, E, T, U, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, a choice-selected refinement, a host modulus function, quotient map equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6528.2 (StarRefinementModulus NameCert obligations)**.** *For every accepted* $\mathsf{StarRefinementModulus}^{\uparrow}$ *carrier*

$$S = (K, R, L, E, T, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StarRefinementModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for compact finite-cover data, star-refinement rows, finite Lebesgue-radius data, triangle estimates, dyadic tolerances, uniform-modulus handoff, transport, replay, provenance, and local naming;*
2. *star-refinement exposure, requiring each target-cover request to pass through the displayed row R before any radius ledger is read;*
3. *Lebesgue-radius exposure, requiring the positive radius row L to be tied to the refined cover and compact metric row K;*
4. *triangle-estimate exactness, requiring E to mediate the finite-center comparison before U is exposed;*
5. *uniform-modulus handoff, requiring U to be produced only from K, R, L, E, T and excluding arbitrary open covers, selected external radii, quotient map equality, countable choice, Lean verification, and bridge checking.*

No StarRefinementModulus obligation is stored outside these rows.

Proof. Project the accepted carrier. The compact metric row is K, the star-refined cover row is R, the finite Lebesgue-radius ledger is L, the triangle estimate is E, and the tolerance-to-modulus handoff is T, U.

The star-refinement exposure row is forced because the carrier has no arbitrary cover coordinate. Every target-cover request therefore enters through R before a radius can be read.

The positive-radius row L is tied to K and R, while the triangle estimate E supplies the finite-center comparison. The dyadic row T then converts the requested target tolerance into the source precision consumed by U.

Transport H and replay C preserve the same route before provenance and local naming. Since the carrier has no row for open-cover compactness, external radius selection, quotient map equality, choice, Lean verification, or bridge data, the listed rows exhaust the NameCert surface. □

Theorem 6528.3 (StarRefinementModulus cover-to-modulus exactness)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read accepted by a* $\mathsf{StarRefinementModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}(K),\ R \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow}(L) \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}(E), T \longmapsto \mathsf{UniformMod}$$

Thus the compact-uniform-continuity consumer receives the cover-refinement-to-uniform-modulus route as displayed BEDC data, not as a host Heine–Cantor theorem, selected Lebesgue number, quotient equality of maps, countable-choice selector, or ambient completeness principle.

Proof. Start with a target tolerance request. Carrier admission exposes the compact finite-cover row K and the star-refinement row R as the only cover data available to the request.

Read the finite Lebesgue-number ledger L. It supplies a displayed positive radius for the refined cover, tied to the compact metric row rather than to an external open-cover compactness theorem.

Apply the metric triangle estimate E with the dyadic tolerance row T. The estimate compares the source point to its finite center and then to the target read, so the resulting precision is the source-side modulus datum.

Finally expose U through H, C, P, N. These rows transport, replay, source, and name the same finite route, leaving no coordinate for selected external radii, quotient map equality, countable choice, host Heine–Cantor, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{StarRefinementModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{StarRefinementModulus}^{\uparrow}$ packet is generated from compact metric finite-cover data, star-refinement rows, finite Lebesgue-radius ledgers, metric triangle estimates, dyadic tolerance rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StarRefinementModulus}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 6528.1, Definition 6528.2, and Definition 6528.3 bind the finite cover-refinement-to-uniform-modulus route for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StarRefinementModulus}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected external Lebesgue numbers, quotient equality of maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Checked closure requires carrier admission, star-refinement exposure, Lebesgue-radius exposure, triangle-estimate exactness, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.StarRefinementModu

Chapter 6529

A Concrete Naming Certificate for $\mathsf{FourierSeries}^{\uparrow}$

Following Zygmund and Stein–Shakarchi on classical Fourier series, the $\mathsf{FourierSeries}^{\uparrow}$ packet records the finite coefficient-window interface for trigonometric-polynomial partial sums on the circle. It sits over the finite Fourier observation surface of chapter 124, the circle source of chapter 58, the real and complex scalar rows of chapter 38 and chapter 39, and the pointwise graph discipline of chapter 130. The packet names finite windows, partial-sum evaluation rows, componentwise transport, replay, provenance, and local naming; it does not import convergence, L^2 completeness, pointwise convergence, or host-function equality.

Definition 6529.1 (Fourier series carrier)**.** *A* $\mathsf{FourierSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, E, R, K, H, C, P, N)$$

with the following rows:

1. *W is the finite coefficient-window row, listing the retained integer frequencies and their displayed source addresses.*
2. *T is the trigonometric-polynomial partial-sum row, binding each retained window to its finite sine, cosine, and exponential tags.*
3. *E is the S^1 evaluation row, exposing only displayed phase samples read through* $\mathsf{S}^{1\uparrow}$*.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *coefficient row for real sine and cosine coefficients, including the finite rational or real-name readback ledger used by the packet.*
5. *K is the* $\mathsf{Complex}^{\uparrow}$ *coefficient row for complex exponential coefficients and the corresponding scalar readback.*
6. *H records componentwise* hsame *transport for coefficient windows, partial sums, S^1 evaluations, real and complex coefficient rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite window request through W, T, E, R, K.*
8. *P is the* Pkg *provenance over* $\mathsf{FourierSeries}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FourierSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $W, T, E, R, K, H, C, P, N$. It has no coordinate for equality of host functions, limiting Fourier series, L^2 completion, pointwise convergence, or a selected representative in an ambient function space.

Theorem 6529.2 (Fourier series NameCert obligations)**.** *For every accepted* $\mathsf{FourierSeries}^{\uparrow}$ *carrier $F = (W, T, E, R, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FourierSeries}^{\uparrow}}$ *surface consists exactly of finite coefficient-window admission through W, trigonometric-polynomial partial-sum admission through T, S^1 evaluation admission through E, real and complex coefficient readback through R and K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6529.1. Its visible rows are exactly $W, T, E, R, K, H, C, P, N$, so every naming duty begins with a finite coefficient window rather than with a completed function or limiting series.

The analytic rows are then read only as finite data. The row T supplies the trigonometric-polynomial partial sum attached to the displayed window, E supplies the circle samples, and R, K supply the real and complex coefficient readbacks.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for host-function equality, convergence, L^2 completeness, or pointwise convergence, the listed rows exhaust the local NameCert surface. □

Theorem 6529.3 (Fourier series finite partial-sum route)**.** *Every* $\mathsf{ContinuousMap}^{\uparrow}$*-facing or scalar-facing read from an accepted* $\mathsf{FourierSeries}^{\uparrow}$ *packet factors through*

$$W, T, E, R, K \longmapsto H, C, P, N.$$

The read may consume finite Fourier observations, displayed S^1 phase samples, real coefficient rows, complex coefficient rows, and continuous-map graph rows, but it exports only the finite partial-sum route displayed by the carrier.

Proof. Start with the coefficient window. The row W fixes the retained frequencies, and T attaches the trigonometric-polynomial partial sum to that finite window.

Read the evaluation and scalar rows next. The S^1 row E supplies the displayed phase samples, while R and K expose the real and complex coefficient readbacks available to the partial sum.

The structural rows H, C, P, N preserve that same finite route under componentwise transport, replay, provenance, and local naming. Therefore any downstream read receives the displayed partial-sum packet only, with convergence, L^2 completeness, pointwise convergence, and host-function equality outside the route. □

Falsifiable prediction: Deleting the coefficient-window row W prevents any accepted $\mathsf{FourierSeries}^{\uparrow}$ partial-sum read, even when trigonometric-polynomial, S^1 evaluation, scalar, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: The finite Fourier observation packet $\mathsf{Fourier}^{\uparrow}$ is not interchangeable with $\mathsf{FourierSeries}^{\uparrow}$ because only $\mathsf{FourierSeries}^{\uparrow}$ binds coefficient windows, trigonometric-polynomial partial sums, S^1 evaluations, real and complex coefficient rows, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted partial-sum route.

Closure status for $\mathsf{NameCert}_{\mathsf{FourierSeries}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FourierSeries}^{\uparrow}$ packet is generated from a finite coefficient window, a trigonometric-polynomial partial-sum row, an S^1 evaluation row, real and complex coefficient rows, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FourierSeries}\uparrow}$, seed)

Scope closed: Definition 6529.1, Definition 6529.2, and Definition 6529.3 seed the finite Fourier-series partial-sum route over Fourier$^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, Real$^{\uparrow}$, Complex$^{\uparrow}$, and ContinuousMap$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FourierSeries}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.FourierSeriesUp`

Bridge status: none

Not claimed: This chapter does not close convergence, completeness of L^2, pointwise convergence, host-function equality, Lean-checked obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window admission, partial-sum readback, circle evaluation, real and complex coefficient readback, transport, replay, provenance, and local naming for BEDC.Derived.FourierSeriesUp.

Chapter 6530

A Concrete Naming Certificate for $\mathsf{OpenDenseSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Troelstra–van Dalen treatment of spread arguments, the $\mathsf{OpenDenseSequence}^{\uparrow}$ packet records the finite input carrier for a sequence of open dense constraints before a Baire or completion consumer reads it. It sits between the $\mathsf{CompleteMetric}^{\uparrow}$ surface of chapter 128, the finite-prefix $\mathsf{BaireSpace}^{\uparrow}$ carrier of chapter 334, the complete-metric Baire route of chapter 6019, the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the displayed open-dense sequence as finite BEDC input data for complete-metric and Real/Completion consumers; it is not countable choice, a selected generic point, open-cover compactness, quotient stream equality, or ambient completeness.

Definition 6530.1 (Open dense sequence carrier)**.** *An* $\mathsf{OpenDenseSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, S, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{BaireSpace}^{\uparrow}$ *source row in which the open-dense constraints are read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the inspected finite prefix of the constraint sequence.*
3. *D is the dense-open ledger row, recording for each visible index that the requested basic neighbourhood has a displayed refinement inside the corresponding open row.*
4. *W is the finite window row of indices and neighbourhood refinements consumed by the downstream Baire or completion route.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *or completion-facing readback row reached after the finite window is replayed.*
6. *H records componentwise* hsame *transport for source, schedule, dense-open ledger, window, readback, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from the source row through S, D, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{OpenDenseSequence}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireCategoryCompleteMetric}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OpenDenseSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, D, W, R, H, C, P, N$. It has no coordinate for countable choice, a selected generic point, open-cover compactness, quotient stream equality, or ambient completeness.

Lean4 (statement) `BEDC.Derived.OpenDenseSequenceUp`

Theorem 6530.2 (Open dense sequence NameCert obligations)**.** *For every accepted* $\mathsf{OpenDenseSequence}^{\uparrow}$ *carrier*

$$O = (X, S, D, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OpenDenseSequence}^{\uparrow}}$ *surface consists exactly of carrier admission, source-row stability, finite sequence-window stability, dense-open ledger stability, completion-facing readback, componentwise transport, displayed replay, provenance, and ledger non-escape through* $X, S, D, W, R, H, C, P, N$.

Proof. Project the carrier of Definition 6530.1. The source, schedule, dense-open ledger, finite window, readback, transport, replay, provenance, and naming rows are finite BHist coordinates.

The NameCert clauses are componentwise. Source-row stability is read through the sibling complete-metric and Baire-space surfaces, finite sequence-window stability is read through the $\mathsf{StreamName}^{\uparrow}$ row, and completion-facing readback is routed through the displayed Real or completion seal before any downstream consumer can use the packet.

Coordinate exhaustion gives non-escape. Since the carrier contains only $X, S, D, W, R, H, C, P, N$, it cannot provide countable choice, a selected generic point, open-cover compactness, quotient stream equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{OpenDenseSequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpenDenseSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6530.1 and Definition 6530.2 seed the finite open-dense sequence input carrier for complete-metric, Baire, StreamName, and Real/Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpenDenseSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close countable choice, selected generic points, open-cover compactness, quotient stream equality, ambient completeness, or bridge checking.

Upgrade path: The next step is obligationClosure: carrier admission, source-row stability, finite sequence-window stability, dense-open ledger stability, completion-facing readback, transport, replay, provenance, and non-escape must be discharged as local rows.

Constructive story: An $\mathsf{OpenDenseSequence}^{\uparrow}$ packet is generated from a complete-metric or Baire source row, a finite StreamName schedule, dense-open ledger rows, a visible window of refinements, Real or completion-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6531

A Concrete Naming Certificate for $\mathsf{RegularCauchyFusion}^{\uparrow}$

Bishop and Bridges constructive analysis compares real names by passing to a common regular tail rather than by choosing a quotient representative. The $\mathsf{RegularCauchyFusion}^{\uparrow}$ packet records that comparison as finite NameCert data: it reads the dyadic tolerance core of chapter 291, the window names of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the fast-subsequence handoff of chapter 1518.

Definition 6531.1 (RegularCauchyFusion carrier)**.** *A* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Phi = (L, R, D, S, T, U, V, W, E, H, C, P, N)$$

whose rows have the following roles:

1. *L and R are the two input regular Cauchy name rows.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by both inputs.*
3. *S and T are the two* $\mathsf{StreamName}^{\uparrow}$ *window schedules read from the inputs.*
4. *U is the common finite refinement window where the schedules are compared.*
5. *V is the* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *handoff that reaches the refinement without countable choice.*
6. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed by the comparison.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *name seal reached from the displayed regular readback.*
8. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFusion}^{\uparrow}}$ *naming rows.*

Lean4 (statement) `BEDC.Derived.RegularCauchyFusionUp`

Theorem 6531.2 (RegularCauchyFusion NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *carrier* $\Phi = (L, R, D, S, T, U, V, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFusion}^{\uparrow}}$ *surface consists exactly of finite obligations for input-name admission, dyadic tolerance reuse, two stream-window reads, common-refinement exposure, fast-subsequence access, regular rational readback, Real-name sealing, componentwise transport, displayed replay, provenance, and local naming. The surface exports no quotient Cauchy class, selected representative, selected limit, countable-choice functional, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge.*

Proof. Project the carrier rows. The input names L, R, dyadic ledger D, stream windows S, T, common refinement U, fast-subsequence handoff V, readback row W, and terminal seal E are all displayed coordinates of Φ.

The structural rows H, C, P, N transport, replay, source, and name only those displayed coordinates. They do not add a quotient class, selected representative, host equality, choice schedule, completeness principle, or bridge row.

Thus the NameCert obligation surface is precisely the finite carrier route listed above. Any consumer asking for the refused objects must ask for a row that is absent from Φ. □

Theorem 6531.3 (RegularCauchyFusion L10 sibling dependency)**.** *Every* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *comparison read factors through the sibling route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{FastCauchySubsequence}^{\uparrow}.$$

The dyadic ledger fixes the tolerance, the two stream-name rows expose finite windows, the fast-subsequence handoff supplies the common refinement, the regular rational row reads the refined comparison, and the Real row seals only that displayed readback.

Proof. Start with the dyadic and stream rows. The ledger D is the only tolerance coordinate, while S and T are the only window schedules available for the two input names.

Use the fast-subsequence handoff row V to reach the common refinement U without selecting an infinite subsequence. The row is finite carrier data and remains tied to the same dyadic and stream schedules.

The regular rational readback W and terminal seal E then consume that common refinement. Since the packet has no quotient, selected-limit, choice, or completeness row, the comparison remains inside the five sibling dependencies displayed in the statement. □

Theorem 6531.4 (RegularCauchyFusion common refinement comparison)**.** *For any accepted* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *carrier, both input regular Cauchy names are compared only after the common refinement row* U *has been displayed. A downstream Real-and-Completion consumer receives the route*

$$L, R, D, S, T, U, V, W, E, H, C, P, N$$

and may inspect finite agreement over the refinement, but it receives no quotient equality of Cauchy names and no selected representative of either input.

Proof. Read the carrier definition. The two input rows L, R are not compared directly; they first expose their dyadic tolerance ledger and stream windows through D, S, T.

The refinement row U is reached through the fast-subsequence handoff V. This gives a finite common window on which the regular rational row W can read both presentations.

The Real seal E receives only that regular readback. The structural rows preserve the route and do not create quotient equality, a representative selector, a selected limit, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFusion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFusion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6531.1, Definition 6531.2, and Definition 6531.3 seed the common-refinement comparison route for two regular Cauchy presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFusion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.RegularCauchyFusionUp`

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy equality, selected representatives, selected limits, countable choice, trichotomy, ambient Real completeness, Lean theorem verification, or a standard bridge.

Upgrade path: A later verification route can encode carrier admission, dyadic tolerance reuse, stream-window comparison, fast-subsequence handoff, regular rational readback, Real-name sealing, transport, replay, provenance, and local naming for the displayed packet.

Constructive story: A RegularCauchyFusion$^{\uparrow}$ packet is generated from two regular Cauchy name rows, a shared DyadicRatCore tolerance ledger, two StreamName window schedules, a common refinement window, a FastCauchySubsequence handoff, RegSeqRat readback, a RealUp name seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Chapter 6532

A Concrete Naming Certificate for $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$

Following Whitney's embedding theorem and the finite-atlas formulation used in classical differential topology, the $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$ packet records the BEDC seed surface for a compact finite-atlas manifold embedded by finitely many real coordinate functions. It sits between the manifold carrier of chapter 96, the compact metric source of chapter 129, the continuous-map rows of chapter 130, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and finite vector-coordinate rows such as chapter 1684. The packet names a finite embedding-candidate ledger and its separation obligations; it is not a host smooth embedding theorem, an ambient Euclidean quotient, a selected infinite atlas, or a global transversality argument.

Definition 6532.1 (Whitney finite-atlas embedding carrier)**.** *A* $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, K, A, F, V, S, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *finite-atlas source row, exposing only the listed chart domains, overlap rows, and local coordinate names;*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row that bounds the inspected chart windows and compact-metric probes;*
3. *A is the finite atlas ledger, recording which chart rows and overlap certificates are consumed by each coordinate read;*
4. *F is a finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows from the displayed source into* $\mathsf{Real}^{\uparrow}$*;*
5. *V is the finite vector-coordinate row whose entries are the real-valued coordinate functions in F;*
6. *S is the separation obligation row, naming finite point-pair and tangent-direction tests that the coordinate family must discharge before an embedding read is exposed;*
7. *E is the local embedding-candidate boundary row, recording injectivity-on-displayed-pairs and tangent-separation obligations as finite BEDC data;*
8. *H records componentwise* hsame *transport for the manifold, compact-metric, atlas, coordinate, vector, separation, and boundary rows;*

9. *C records displayed* Cont *replay routes from atlas windows through coordinate functions and into the finite vector row;*
10. *P is the* Pkg *provenance over* $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *finite vector-coordinate rows,* BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}}$ *row.*

The carrier contains no host smooth embedding theorem, no selected countable atlas, no ambient Euclidean quotient equality, no global transversality principle, and no bridge-checked standard object.

Lean4 ✓ `BEDC.Derived.WhitneyEmbeddingFiniteAtlasUp.WhitneyEmbeddingFiniteAtlasTasteGate single carrier alignment`

Theorem 6532.2 (Whitney finite-atlas NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}}$ *surface consists of carrier admission for $W = (M, K, A, F, V, S, E, H, C, P, N)$, finite-atlas exposure, compact-metric source control, coordinate-family accountability through* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, finite vector-coordinate readback, separation-obligation display, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and non-escape for host embedding theorems, global transversality, selected infinite atlases, quotient Euclidean equality, and bridge checking.*

Lean4 ✓ `BEDC.Derived.WhitneyEmbeddingFiniteAtlasUp.WhitneyEmbeddingFiniteAtlasNameCertObligations`

Proof. Carrier admission is projection from Definition 6532.1. The available rows are exactly $M, K, A, F, V, S, E, H, C, P, N$.

Read the coordinate family through $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Each coordinate in F is a displayed real-valued map, and the row V packages only the finite vector-coordinate readback of those maps.

Finally inspect the separation and structural rows. The row S records the point-pair and tangent-direction obligations, while E, H, C, P, N preserve, replay, source, and name the same finite route. Since no other carrier coordinate is present, the packet exports no host embedding theorem, transversality principle, selected infinite atlas, Euclidean quotient equality, or bridge-checked standard object. □

Theorem 6532.3 (Whitney finite-atlas separation handoff)**.** *Every embedding-facing consumer of an accepted* $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$ *carrier must read the finite-atlas row A, the compact-metric source row K, the coordinate-family rows F, the finite vector-coordinate row V, and the separation obligation row S before the embedding-candidate boundary E is exposed. The handoff supplies only displayed point-pair and tangent-direction obligations over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and finite vector-coordinate rows.*

Lean4 ✓ `BEDC.Derived.WhitneyEmbeddingFiniteAtlasUp.WhitneyEmbeddingFiniteAtlasSeparationHandoff`

Proof. Project the accepted carrier. The embedding-candidate boundary E is downstream of the atlas, compact source, coordinate-family, vector-coordinate, and separation rows.

Use the NameCert obligations of Definition 6532.2. They force every public read through $M, K, A, F, V, S, H, C, P, N$ before E can be consumed.

Thus a downstream embedding-facing chapter receives the finite separation handoff only as displayed BEDC data. It cannot replace the route by an ambient Euclidean embedding theorem, a selected atlas, host smooth transversality, or quotient equality of coordinate vectors. □

Theorem 6532.4 (Whitney finite-atlas nonescape)**.** *For every accepted* WhitneyEmbeddingFiniteAtlas$^{\uparrow}$ *carrier* $W = (M, K, A, F, V, S, E, H, C, P, N)$*, no consumer can extract a host smooth embedding theorem, a global transversality argument, a selected infinite atlas, quotient equality in an ambient Euclidean space, or a bridge-checked standard theorem. Every public read factors through the displayed finite atlas, compact-metric source, continuous real coordinate family, finite vector-coordinate row, separation ledger, and structural rows.*

Lean4 ✓ `BEDC.Derived.WhitneyEmbeddingFiniteAtlasUp.WhitneyEmbeddingFiniteAtlasNonescape`

Proof. Inspect the carrier coordinates in Definition 6532.1. They list finite atlas data, compact-metric control, coordinate maps, finite vector readback, separation obligations, an embedding-candidate boundary, and structural transport.

Apply Definition 6532.3. The only embedding-facing output is the finite separation handoff through A, K, F, V, S, E and the structural rows H, C, P, N.

The excluded objects are absent from this coordinate list. Therefore no consumer of the packet can read host embedding semantics, global transversality, selected infinite-atlas data, Euclidean quotient equality, or a bridge-checked standard theorem from WhitneyEmbeddingFiniteAtlas$^{\uparrow}$. □

Closure status for NameCert$_{\text{WhitneyEmbeddingFiniteAtlas}^{\uparrow}}$**.**

Constructive story: A WhitneyEmbeddingFiniteAtlas$^{\uparrow}$ packet is generated from a finite ManifoldUp atlas row, compact-metric source control, a finite atlas ledger, continuous Real-valued coordinate rows, finite vector-coordinate readback, a separation-obligation ledger, an embedding-candidate boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{WhitneyEmbeddingFiniteAtlas}^{\uparrow}}$, seed)

Scope closed: Definition 6532.1, Definition 6532.2, Definition 6532.3, and Definition 6532.4 seed the finite-atlas embedding carrier, obligation ledger, separation handoff, and nonescape boundary.

Formal verification: FormalStatus(NameCert$_{\text{WhitneyEmbeddingFiniteAtlas}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Whitney's full embedding theorem, global transversality, selected infinite atlases, ambient Euclidean quotient equality, Lean theorem checking, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-atlas coordinate accountability, separation handoff, and nonescape for BEDC.Derived.WhitneyEmbeddingFiniteAtlasUp.

Chapter 6533

A Concrete Naming Certificate for $\mathsf{CompactExhaustion}^{\uparrow}$

Following Bishop and Bridges, a compact exhaustion is read constructively as displayed finite exhaustion data: each public request exposes a prefix of compact metric windows, finite epsilon-net witnesses, and the metric and real rows that make those windows usable. The $\mathsf{CompactExhaustion}^{\uparrow}$ packet sits between the compact metric witness of chapter 129, the finite net interface of chapter 1767, the completion boundary of chapter 128, the metric carrier of chapter 57, and the terminal real-name seal of chapter 38. It gives later compactness, completion, and continuous-image chapters a finite compact-exhaustion dependency without importing open-cover compactness or a choice-selected cover family.

Definition 6533.1 (CompactExhaustion carrier)**.** *A* $\mathsf{CompactExhaustion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, M, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, distance observations, and finite history entries inspected by the exhaustion prefix.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *window row. Its displayed entries $K_0, \ldots, K_m$ are the compact windows currently exposed to downstream consumers.*
3. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row attached to the displayed windows. For each requested positive real or rational tolerance already visible in the packet, F records the finite net witnesses used inside the matching compact metric window.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *compatibility row, read only as a handoff boundary for consumers that need completion-facing metric data over the same windows.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and radius seal row. It records the finite real-name data used by the epsilon-net requests and by any radius comparison attached to the compact windows.*
6. *H records componentwise* hsame *transport for X, K, F, M, R, C, P, N and for the local classifier row.*
7. *C records displayed* Cont *replay from an exhaustion-prefix request through the metric source, compact metric window, finite-net witness, completion handoff, real seal, transport, provenance, and local name.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactExhaustion}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactExhaustion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, K, F, M, R, H, C, P, N$. It has no coordinate for open-cover compactness, a selected cover family, a completed countable exhaustion, quotient equality of window schedules, or ambient real completeness beyond the displayed $\mathsf{Real}^{\uparrow}$ *seal.*

Theorem 6533.2 (CompactExhaustion NameCert obligations)**.** *For every accepted* $\mathsf{CompactExhaustion}^{\uparrow}$ *carrier $E = (X, K, F, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactExhaustion}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient metric compatibility, compact-window admission, finite-net tolerance readback, completion-facing handoff, real-seal accountability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the rows $X, K, F, M, R, H, C, P, N$.*

Proof. Project the carrier of Definition 6533.1. The analytic rows are exactly the ambient metric source, compact metric windows, finite epsilon-net witnesses, completion handoff row, and real tolerance seal.

The compact-window and finite-net obligations are ordered by the displayed prefix. A consumer first reads the metric source X, then a compact window from K, then the corresponding finite-net witness in F, and only then the completion or real-facing rows M, R.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no open-cover, selected-cover, completed-exhaustion, quotient-schedule, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6533.3 (CompactExhaustion finite-net induction)**.** *Every finite-prefix consumer of an accepted* $\mathsf{CompactExhaustion}^{\uparrow}$ *packet can inspect the exhaustion by induction over the displayed compact-window list*

$$K_0, \ldots, K_m.$$

At each stage the consumer receives the same route

$$X \longmapsto K_i \longmapsto F_i \longmapsto R_i \longmapsto M \longmapsto H, C, P, N,$$

where F_i is the finite epsilon-net witness row and R_i is the real tolerance seal attached to the ith window. The induction exports no open-cover compactness, choice-selected subcover, hidden window after m, quotient equality of exhaustion schedules, or completed countable family.

Proof. Start with the displayed prefix. The carrier exposes only finitely many compact-window entries, so ordinary finite induction over $K_0, \ldots, K_m$ is available inside the packet.

At the base stage, the consumer reads X, the first compact metric window, its finite epsilon-net witness, and the matching real tolerance seal. The rows M, H, C, P, N are then attached as completion handoff, transport, replay, provenance, and local naming.

At the successor stage, the same projection is repeated for the next displayed window. No constructor reveals an undisplayed window or a selected cover; the induction step is only a replay of the next finite row already present in K and F.

Thus the public exhaustion read is finite-prefix induction through $X, K, F, R, M, H, C, P, N$. Anything requiring a completed countable exhaustion, open-cover compactness, or choice-selected cover data must be supplied by a different packet. □

Falsifiable prediction: Every accepted $\mathsf{CompactExhaustion}^{\uparrow}$ consumer of a finite exhaustion prefix reads the ambient $\mathsf{Met}^{\uparrow}$ row, a displayed $\mathsf{CompactMetric}^{\uparrow}$ window, its $\mathsf{FiniteEpsilonNet}^{\uparrow}$ witness, the $\mathsf{Real}^{\uparrow}$ tolerance seal, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before using compact-exhaustion data.

Independence witness: compactmetric, finiteepsilonnet, completemetric, metric, real: $\mathsf{CompactExhaustion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries displayed compact-window prefixes, finite epsilon-net rows, completion-facing handoff data, real tolerance seals, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactExhaustion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactExhaustion}^{\uparrow}$ packet is generated from an ambient Metric row, displayed CompactMetric windows, FiniteEpsilonNet witnesses, a CompleteMetric handoff row, Real tolerance seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactExhaustion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6533.1, Definition 6533.2, and Definition 6533.3 seed the finite compact-exhaustion interface over compact metric windows and finite epsilon-net witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactExhaustion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, completed countable exhaustions, quotient equality of exhaustion schedules, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ambient metric compatibility, compact-window admission, finite-net tolerance readback, completion-facing handoff, real-seal accountability, transport stability, provenance, and local naming for BEDC.Derived.CompactExhaustionUp.

Chapter 6534

A Concrete Naming Certificate for $\mathsf{PhysicalInductionGate}^{\uparrow}$

The $\mathsf{PhysicalInductionGate}^{\uparrow}$ packet realizes the physical-induction certificate of section 9414.25 as finite NameCert data. It sits after the reality-facing audit surfaces of chapter 1191, the open-fit rows of chapter 1132, the continuation countermodel boundary of chapter 1183, and the failure-surface sibling of chapter 1163. The packet names when a finite fit may be reused over a displayed continuation class, and it refuses any inference that has no finite stability basis or no falsifiable defeat surface.

Definition 6534.1 (PhysicalInductionGate carrier)**.** *A* $\mathsf{PhysicalInductionGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, O, M, \Pi, F, Cn, B, Y, H, R, P, N)$$

with the following displayed rows:

1. *S is the source finite observation history row, including the prefix H_n and the observation classifier used by the fitted model.*
2. *O is the open continuation row. It names the admissible continuation class inspected by the packet and exposes only finite continuation requests.*
3. *M is the displayed model-signature row read from the fitted model over the same finite history.*
4. *Π is the probe-bundle and comparison row governing which signatures are compared.*
5. *F is the finite-fit certificate row for the current prefix.*
6. *Cn is the continuation-stability row. It records the local stability basis licensing reuse over the displayed continuation class.*
7. *B is the failure-surface row. It records the observations or continuation refinements that would separate model-generated and observed signatures.*
8. *Y is the predictive-descent row, requiring the prediction operation to descend through the classifier named by Π before reuse is public.*
9. *H records componentwise* hsame *transport for $S, O, M, \Pi, F, Cn, B, Y, R, P, N$.*
10. *R records displayed* Cont *replay from a finite fit through continuation request, stability basis, failure-surface test, descent check, provenance, and local name.*

11. *P is the* Pkg *provenance over* $\mathsf{PhysicalInductionGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *together with the reality-facing siblings named above.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalInductionGate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, O, M, \Pi, F, Cn, B, Y, H, R, P, N$. It has no coordinate for a completed future observation carrier, universal physical necessity, a final-fit oracle, hidden continuation choice, or predictive reuse without a displayed failure surface.

Theorem 6534.2 (PhysicalInductionGate NameCert obligations)**.** *For every accepted* $\mathsf{PhysicalInductionGate}^{\uparrow}$ *carrier $G = (S, O, M, \Pi, F, Cn, B, Y, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PhysicalInductionGate}^{\uparrow}}$ *surface consists exactly of source-prefix admission, continuation-class admission, model-signature readback, probe-comparison compatibility, finite-fit certification, continuation-stability display, failure-surface display, predictive-descent display, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the rows $S, O, M, \Pi, F, Cn, B, Y, H, R, P, N$.*

Lean4 ✓ `BEDC.Derived.PhysicalInductionGateUp.PhysicalInductionGateNameCertObligations`

Proof. Project the carrier of Definition 6534.1. The reality-facing rows are exactly the finite source prefix, continuation class, model signature, probe comparison, finite-fit certificate, stability basis, failure surface, and predictive-descent row.

The obligation order follows the physical-induction certificate. A reuse read first displays the finite fit F over S, M, Π, then displays the continuation class O, then reads the stability basis Cn, and only then exposes the failure-surface row B and descent row Y.

The structural rows H, R, P, N keep that route stable under transport, continuation replay, provenance, and local naming. Since the carrier has no completed future, final oracle, hidden continuation selector, or failure-free law row, the listed rows exhaust the local NameCert obligations. □

Theorem 6534.3 (PhysicalInductionGate failure surface)**.** *Every accepted* $\mathsf{PhysicalInductionGate}^{\uparrow}$ *prediction read must expose a finite failure-surface row before it can use continuation stability:*

$$S \longmapsto O \longmapsto M \longmapsto \Pi \longmapsto F \longmapsto Cn \longmapsto B \longmapsto Y \longmapsto H, R, P, N.$$

If the row B is absent, or if B is detached from the same continuation class O and probe comparison Π, the packet exports no licensed physical-induction gate.

Proof. Start with Definition 6534.1. The only route from finite fit to prediction reuse passes through the displayed continuation class, stability basis, failure-surface row, and predictive-descent row.

Apply Definition 6534.2. It makes failure-surface display a local NameCert obligation, not an optional annotation. A consumer therefore cannot consume Cn as a lawlike stability basis unless it also consumes B over the same O and Π.

The rows H, R, P, N preserve this route under transport, replay, provenance, and naming. Hence a read with no B row, or with a failure surface over a different continuation class or classifier, remains outside $\mathsf{PhysicalInductionGate}^{\uparrow}$. □

Falsifiable prediction: A physical-induction consumer that claims predictive reuse while omitting the finite source prefix, continuation class, model-signature row, probe-comparison row, finite-fit certificate, continuation-stability basis, failure-surface row, predictive-descent row, transport, replay, provenance, or local NameCert refutes the $\mathsf{PhysicalInductionGate}^{\uparrow}$ packet.

Independence witness: physicalnamecertaudit, reality-constrained-open-fit, continuationcountermodel, reality-constrained-failure-surface: no listed sibling is bijective with $\mathsf{PhysicalInductionGate}^{\uparrow}$ because this packet jointly carries finite fit, continuation stability, failure-surface accountability, predictive descent, transport, replay, provenance, and local naming as one reuse gate.

Future TasteGate work for this chapter must provide explicit $\mathsf{Nontrivial\ PhysicalInductionGate}^{\uparrow}$ and $\mathsf{FieldFaithful\ PhysicalInductionGate}^{\uparrow}$ instances preserving $S, O, M, \Pi, F, Cn, B, Y, H, R, P, N$ before any checked carrier-alignment theorem may serve as the public gate.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalInductionGate}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalInductionGate}^{\uparrow}$ packet is generated from a finite source observation prefix, displayed continuation class, model signature, probe-comparison row, finite-fit certificate, continuation-stability basis, failure-surface row, predictive-descent row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalInductionGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6534.1, Definition 6534.2, and Definition 6534.3 seed the finite continuation-stability and failure-surface gate for physical induction certificates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalInductionGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close universal physical necessity, completed future observation carriers, failure-free law certificates, hidden continuation choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-prefix admission, continuation-class admission, finite-fit certification, continuation-stability display, failure-surface display, predictive descent, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.PhysicalInductionGateUp.

Chapter 6535

A Concrete Naming Certificate for $\mathsf{SemiAlgebraicSet}^{\uparrow}$

Following Tarski and Seidenberg, the $\mathsf{SemiAlgebraicSet}^{\uparrow}$ packet records a real-algebraic set classifier as finite polynomial sign data rather than as an ambient subset oracle. It sits between the real-name seal of chapter 38, the polynomial carrier of chapter 50, the set-like membership surface of chapter 3668, the first-order formula row of chapter 197, and the ordered comparison row of chapter 1378. The packet names the constructive BEDC carrier for finite Boolean combinations of polynomial sign tests over displayed real coordinates.

Definition 6535.1 (SemiAlgebraicSet carrier)**.** *A* $\mathsf{SemiAlgebraicSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, R, P, S, F, O, H, C, G, N)$$

with the following displayed rows:

1. *X is the finite coordinate arity and variable-context row for the ambient real affine space.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate-seal row. It records the real-name data through which coordinate comparisons are read.*
3. *P is the finite* $\mathsf{Poly}^{\uparrow}$ *ledger. Its entries list the displayed polynomial terms used by the local classifier.*
4. *S is the polynomial sign ledger. For each displayed polynomial row and each visible coordinate tuple, it records whether the public test reads $<$, $=$, or $>$ through the ordered real surface.*
5. *F is the* $\mathsf{FirstOrder}^{\uparrow}$ *formula row generated from the finite sign atoms by Boolean operations and bounded displayed quantifier windows.*
6. *O is the* $\mathsf{Order}^{\uparrow}$ *comparison row used by sign tests before the formula row is evaluated.*
7. *H records componentwise* hsame *transport for $X, R, P, S, F, O, C, G, N$ and for the local classifier.*
8. *C records displayed* Cont *replay from coordinate readback through polynomial evaluation, sign comparison, formula evaluation, transport, provenance, and local naming.*
9. *G is the* Pkg *provenance over* $\mathsf{SemiAlgebraicSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SemiAlgebraicSet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, R, P, S, F, O, H, C, G, N$. It has no coordinate for an arbitrary powerset predicate, a quotient of polynomial formulas, a selected decomposition into cells, an undecided host real comparison, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6535.2 (SemiAlgebraicSet NameCert obligations)**.** *For every accepted* $\mathsf{SemiAlgebraicSet}^{\uparrow}$ *carrier $A = (X, R, P, S, F, O, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{SemiAlgebraicSet}^{\uparrow}}$ *surface consists exactly of coordinate-context admission, real-coordinate seal accountability, polynomial-ledger admission, polynomial-sign ledger admission, first-order formula formation, ordered sign comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the rows $X, R, P, S, F, O, H, C, G, N$.*

Proof. Project the carrier of Definition 6535.1. The algebraic rows are exactly the coordinate context, real-coordinate seals, finite polynomial ledger, sign ledger, formula row, and ordered comparison row.

The obligation order follows the displayed classifier. A membership read first exposes the real coordinate rows R, then evaluates a polynomial from P, then reads its sign through S and O, and only then evaluates the formula row F.

The rows H, C, G, N preserve this route under transport, replay, provenance, and local naming. Since the carrier contains no arbitrary subset predicate, cell-decomposition choice, quotient formula coordinate, host comparison oracle, or ambient completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 6535.3 (SemiAlgebraicSet polynomial sign ledger)**.** *Every membership read from an accepted* $\mathsf{SemiAlgebraicSet}^{\uparrow}$ *carrier factors through the displayed polynomial sign ledger*

$$R \longmapsto P \longmapsto S \longmapsto O \longmapsto F \longmapsto H, C, G, N.$$

Thus a consumer receives only finite sign tests for displayed $\mathsf{Poly}^{\uparrow}$ *rows over displayed* $\mathsf{Real}^{\uparrow}$ *coordinates before the* $\mathsf{SetLike}^{\uparrow}$ *membership surface is public. It receives no arbitrary subset predicate, no selected cell decomposition, no quotient formula equality, no undecided host trichotomy, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier projection. The only route from a coordinate tuple to membership is the finite row sequence through real coordinate seals, polynomial rows, sign comparisons, ordered tests, and first-order formula evaluation.

The sign ledger S is attached to the displayed polynomial list P and the ordered comparison row O. It therefore cannot be replaced by an external predicate on real tuples or by a host-level decision procedure.

The structural rows H, C, G, N transport and name the same finite route. After those rows are read, the public $\mathsf{SetLike}^{\uparrow}$ surface receives a membership value generated by the displayed sign ledger alone. □

Falsifiable prediction: A $\mathsf{SemiAlgebraicSet}^{\uparrow}$ consumer that claims membership while omitting a displayed $\mathsf{Real}^{\uparrow}$ coordinate seal, $\mathsf{Poly}^{\uparrow}$ ledger row, polynomial sign row, $\mathsf{Order}^{\uparrow}$ comparison row, $\mathsf{FirstOrder}^{\uparrow}$ formula row, transport, replay, provenance, or local $\mathsf{NameCert}$ refutes this packet.

Independence witness: real, polynomial, setlike, firstorder, order: $\mathsf{SemiAlgebraicSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries real-coordinate seals, finite polynomial rows, sign comparisons, formula formation, set-like membership export, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SemiAlgebraicSet}^{\uparrow}}$.

Constructive story: A $\mathsf{SemiAlgebraicSet}^{\uparrow}$ packet is generated from a coordinate context, Real coordinate seals, Polynomial rows, polynomial sign rows, FirstOrder formula formation, Order comparisons, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SemiAlgebraicSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6535.1, Definition 6535.2, and Definition 6535.3 seed the finite polynomial-sign carrier for semi-algebraic membership.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SemiAlgebraicSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Tarski–Seidenberg elimination, cell decomposition, arbitrary subset predicates, quotient formula equality, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coordinate admission, real-coordinate seal accountability, polynomial-ledger admission, sign-ledger admission, formula formation, ordered comparison, transport stability, provenance, and local naming for BEDC.Derived.SemiAlgebraicSetUp.

Chapter 6536

A Concrete Naming Certificate for $\mathsf{FiniteMinimax}^{\uparrow}$

Following von Neumann's finite minimax theorem and constructive convex analysis, the $\mathsf{FiniteMinimax}^{\uparrow}$ packet names a displayed finite saddle-certificate route over finite strategy polytopes and rational payoff brackets. It consumes the convex carrier of chapter 208, the finite polytope surface of chapter 209, the compact finite-net discipline of chapter 129, the continuous payoff route of chapter 130, and the terminal real-name seal of chapter 38. The packet records a finite convex compact saddle certificate; it is not a host proof of Hahn–Banach separation, a selected global optimizer, an ambient quotient of mixed strategies, countable choice, or a completed function-space theorem.

Definition 6536.1 (FiniteMinimax carrier surface)**.** *A* $\mathsf{FiniteMinimax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, P, Q, F, A, B, S, H, C, K, N)$$

with the following displayed rows:

1. *X and Y are finite strategy-index rows naming the two player faces.*
2. *P and Q are* $\mathsf{Polytope}^{\uparrow}$ *strategy-simplex rows, each read through the ambient* $\mathsf{ConvexSet}^{\uparrow}$ *carrier.*
3. *F is the compact finite-net row, recording the finite covering windows used to inspect the two polytopes through* $\mathsf{CompactMetric}^{\uparrow}$*.*
4. *A is the* $\mathsf{ContinuousMap}^{\uparrow}$ *payoff row from the displayed product windows to the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *B is the rational tolerance bracket ledger, recording lower and upper payoff bounds for the inspected net cells.*
6. *S is the saddle inequality ledger, recording the two displayed inequalities that certify the bracketed saddle window.*
7. *H records componentwise* hsame *transport for strategy rows, polytopes, finite nets, payoff maps, rational brackets, saddle inequalities, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite strategy query through X, Y, P, Q, F, A, B, S.*
9. *K is the* Pkg *provenance over* $\mathsf{FiniteMinimax}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMinimax}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, P, Q, F, A, B, S, H, C, K, N$. It has no coordinate for a host mixed-strategy quotient, selected exact optimizer, global real completeness, arbitrary separation principle, or countable choice.

Theorem 6536.2 (FiniteMinimax NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMinimax}^{\uparrow}$ *carrier*

$$M = (X, Y, P, Q, F, A, B, S, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteMinimax}^{\uparrow}}$ *surface consists exactly of carrier admission, finite strategy-row stability, convex-polytope compatibility for P, Q, compact finite-net coverage, continuous payoff readback, rational tolerance bracket stability, saddle inequality exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and ledger non-escape through $X, Y, P, Q, F, A, B, S, H, C, K, N$.*

Proof. Project the carrier surface of Definition 6536.1. The only analytic rows available to a minimax consumer are the displayed strategy indices, convex-polytope rows, finite-net row, continuous payoff row, rational brackets, and saddle inequality ledger.

The NameCert clauses are componentwise. Convexity is read through the sibling $\mathsf{ConvexSet}^{\uparrow}$ surface, finite mixed-strategy admission through $\mathsf{Polytope}^{\uparrow}$, compact inspection through $\mathsf{CompactMetric}^{\uparrow}$, and payoff readback through $\mathsf{ContinuousMap}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed.

The saddle ledger is then checked against the same bracket rows on the same finite windows. Transport, replay, provenance, and local naming preserve this route without adding a coordinate for a selected optimizer, quotient mixed-strategy equality, countable choice, or an ambient separation theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMinimax}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMinimax}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6536.1 and Definition 6536.2 seed the finite saddle-certificate carrier over convex polytopes, compact finite nets, continuous payoff rows, rational brackets, and saddle inequalities.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMinimax}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected exact optimizers, host mixed-strategy quotients, arbitrary separation theorems, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires carrier admission, convex-polytope compatibility, compact finite-net coverage, continuous payoff readback, rational bracket stability, saddle inequality exactness, transport, replay, provenance, and non-escape for BEDC.Derived.FiniteMinimaxUp.

Constructive story: A $\mathsf{FiniteMinimax}^{\uparrow}$ packet is generated from finite strategy rows, two convex polytope strategy surfaces, compact finite-net windows, a continuous payoff row, rational tolerance brackets, a saddle inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6537

A Concrete Naming Certificate for $\mathsf{FourierPlancherel}^{\uparrow}$

Following Plancherel and the Fourier analysis presentation of Stein–Shakarchi, the $\mathsf{FourierPlancherel}^{\uparrow}$ packet records the finite-energy isometry surface that sits after the finite observation route of chapter 124, the transform rows of chapter 4025, the finite energy identity of chapter 1940, and the finite energy carrier of chapter 6309. It also consumes the pairing row of chapter 86, the integral row of chapter 123, the regular-rational route of chapter 301, and the real seal of chapter 38. The chapter names only the BEDC certificate for the finite-energy Plancherel handoff; it does not assert Fourier inversion, pointwise convergence, quotient equality of functions, selected representatives, or ambient Hilbert-space completeness.

Definition 6537.1 (Fourier-Plancherel finite-energy carrier)**.** *A* $\mathsf{FourierPlancherel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, A, E, J, I, Q, R, S, H, C, V, N)$$

with the following visible rows:

1. *T is the* $\mathsf{FourierTransform}^{\uparrow}$ *transform row, read through chapter 4025;*
2. *A is the* $\mathsf{Parseval}^{\uparrow}$ *finite identity row, read through chapter 1940;*
3. *E is the* $\mathsf{FiniteFourierEnergy}^{\uparrow}$ *energy ledger, read through chapter 6309;*
4. *J is the* $\mathsf{Fourier}^{\uparrow}$ *finite observation source row;*
5. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row;*
6. *Q is the* $\mathsf{Integral}^{\uparrow}$ *integral-energy row;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational approximation and readback row;*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed energy comparison;*
9. *H stores componentwise* hsame *transports for $T, A, E, J, I, Q, R, S, V, N$;*
10. *C stores the displayed* Cont *replay routes from finite transform observations and Parseval ledgers to the sealed energy comparison;*
11. *V is the* Pkg *provenance over the Fourier, FourierTransform, Parseval, FiniteFourierEnergy, InnerProduct, Integral, RegSeqRat, and Real sibling packets;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FourierPlancherel}^{\uparrow}}$ *row.*

The carrier has no coordinate for Fourier inversion, pointwise convergence, quotient equality of functions, selected representatives, an unrestricted completed Hilbert space, or Lean verification.

Theorem 6537.2 (Fourier-Plancherel NameCert obligations)**.** *For every accepted* $\mathsf{FourierPlancherel}^{\uparrow}$ *carrier* $P = (T, A, E, J, I, Q, R, S, H, C, V, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FourierPlancherel}^{\uparrow}}$ *obligation surface consists of carrier admission, transform-row visibility, Parseval-row visibility, finite-energy ledger visibility, Fourier observation visibility, inner-product and integral row compatibility, regular-rational readback, Real seal stability, componentwise transport, displayed replay, provenance, and local naming. These obligations do not include inversion, convergence, quotient function equality, selected representatives, or ambient Hilbert-space completeness.*

Proof. Project the carrier of Definition 6537.1. The definition lists exactly the finite rows $T, A, E, J, I, Q, R, S, H, C$, and assigns the transform, Parseval, energy, observation, pairing, integral, readback, seal, transport, replay, provenance, and naming roles.

The analytic rows are read only through their sibling certificates. The transform surface comes from $\mathsf{FourierTransform}^{\uparrow}$, the finite identity from $\mathsf{Parseval}^{\uparrow}$, the energy ledger from $\mathsf{FiniteFourierEnergy}^{\uparrow}$, and the finite observation source from $\mathsf{Fourier}^{\uparrow}$.

The scalar comparison remains sealed. The rows I, Q, R, S carry inner-product, integral, regular-rational, and Real-seal data, while H, C, V, N preserve and name the same route. Since no additional coordinate is present, the excluded inversion, convergence, quotient, representative, and ambient-completeness readings cannot be projected. □

Theorem 6537.3 (Fourier-Plancherel finite-energy isometry)**.** *Every* $\mathsf{FourierPlancherel}^{\uparrow}$ *consumer that reads a finite-energy isometry must factor through*

$$T, A, E, J, I, Q, R, S, H, C, V, N.$$

The route identifies only the displayed BEDC energy comparison between the transform-side finite-energy ledger and the source-side finite-energy ledger after regular-rational readback and Real sealing. It does not identify host functions, choose representatives, complete a quotient coefficient space, prove Fourier inversion, or assert pointwise convergence.

Proof. Begin with the transform-side rows. The consumer reads T through $\mathsf{FourierTransform}^{\uparrow}$ and the finite observation source J through $\mathsf{Fourier}^{\uparrow}$.

Next read the energy identity surface. The rows A and E supply the Parseval and finite Fourier energy ledgers, while I and Q provide the pairing and integral-energy rows used by the displayed comparison.

Finally pass through scalar readback and sealing. The row R carries the regular-rational approximation route, S seals the resulting comparison as a $\mathsf{Real}^{\uparrow}$-facing energy statement, and H, C, V, N preserve, replay, source, and name the same finite route.

The carrier has no coordinate for host function equality, selected representatives, quotient coefficient spaces, inversion, or pointwise convergence. Hence the isometry reading is exactly the finite-energy BEDC handoff stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{FourierPlancherel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FourierPlancherel}^{\uparrow}$ packet is generated from FourierTransform, Parseval, FiniteFourierEnergy, Fourier observation, InnerProduct, Integral, RegSeqRat, Real seal, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FourierPlancherel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6537.1, Definition 6537.2, and Definition 6537.3 seed the finite-energy Plancherel handoff consumed by downstream Fourier-completion work.

Formal verification: FormalStatus(NameCert$_{\text{FourierPlancherel}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Fourier inversion, pointwise convergence, quotient equality of functions, selected representatives, ambient Hilbert-space completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier and NameCert obligation coverage for the displayed Plancherel finite-energy route.

Chapter 6538

A Concrete Naming Certificate for $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completion and totally bounded metric data, the $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ packet records the finite route by which total-bounded approximation rows feed a completion-facing seal. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the completion route of chapter 1506, the complete-metric seal of chapter 128, and the finite-net carrier of chapter 1767. The packet names a constructive completion interface from displayed finite total-bounded data; it is not open-cover compactness, quotient completion equality, a selected completion point, ambient completeness, or choice.

Definition 6538.1 (EffectiveTotalBoundedCompletion carrier)**.** *An* $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, E, Q, M, C, A, W, R, S, H, L, P, N)$$

with the following visible rows:

1. *K is the bounded metric source row whose distance and diameter requests are read by the packet.*
2. *E is the finite-net row, displaying for each dyadic tolerance a finite* $\mathsf{ProbeBundle}$ *of centers and coverage witnesses.*
3. *Q is the Cauchy-net refinement row, recording that later finite nets refine earlier tolerances by displayed containment and proximity witnesses.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row consumed by completion-facing readers.*
5. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row, available only after E, Q, M have been displayed.*
6. *A is the total-bounded approximation ledger, binding finite-net centers to the Cauchy-net refinement route.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect source and center data.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the same finite windows.*
9. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed readback route.*
10. *H records componentwise* hsame *transport for $K, E, Q, M, C, A, W, R, S, L, P, N$.*

11. *L records displayed* Cont *replay of finite-net, Cauchy-net, completion, and readback requests.*
12. *P is the* Pkg *provenance over* $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for arbitrary open covers, quotient classes of Cauchy sequences, selected representatives, host complete metric objects, or choice.

Theorem 6538.2 (EffectiveTotalBoundedCompletion NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ *carrier* $T = (K, E, Q, M, C, A, W, R, S, H, L, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EffectiveTota}}$ *surface consists exactly of source admission through K, finite-net admission through E, Cauchy-net refinement through Q, MetricCompletion routing through M, CompleteMetric sealing through C, total-bounded approximation through A, StreamName window inspection through W, RegSeqRat readback through R, Real sealing through S, componentwise* hsame *transport through H, displayed* Cont *replay through L, provenance through P, and local naming through N.*

Every completion-facing consumer must therefore display the finite-net and Cauchy-net rows before it can read the MetricCompletion or CompleteMetric rows, and every Real-facing consumer must display W, R, S on the same route.

Proof. Project the accepted carrier. The analytic rows are $K, E, Q, M, C, A, W, R, S$, and the structural rows are H, L, P, N.

The total-bounded part is finite. The finite-net row E supplies explicit centers and coverage witnesses, while Q records refinement from one tolerance to the next. The approximation ledger A binds those two rows before the completion rows M, C are visible.

The readback part stays on the same packet. The window row W, regular rational row R, and Real seal row S are downstream of E, Q, A and are replayed through L. Componentwise transport H, package provenance P, and local naming N do not add a second completion object. Since the carrier contains no open-cover, quotient, selected-limit, host-completeness, or choice coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6538.3 (EffectiveTotalBoundedCompletion Cauchy-net induction)**.** *Let $T = (K, E, Q, M, C, A, W, R, S,$ be an accepted* $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ *carrier. A completion-facing induction over Cauchy-net approximants may advance from tolerance n to tolerance $n+1$ only by reading the finite-net row E, the refinement row Q, and the approximation ledger A before it consumes the MetricCompletion row M, the CompleteMetric seal C, or the Real-facing rows W, R, S.*

Consequently the induction exports the finite route

$$K, E, Q, A \longmapsto M, C \longmapsto W, R, S \longmapsto H, L, P, N,$$

not an open-cover compactness proof, a quotient class of Cauchy sequences, a choice-selected subsequence, or an ambient completeness theorem.

Proof. Start with the displayed finite-net row. At each tolerance, E supplies the finite center list and coverage witnesses used by the current approximation.

Advance the induction through Q and A. The refinement row identifies how the next tolerance is attached to the previous finite net, and the approximation ledger records that the same displayed witnesses remain within the total-bounded packet. No external cover selection is consulted.

Only after this finite step may a consumer read M and C. The MetricCompletion and CompleteMetric rows are therefore seals over the displayed Cauchy-net route. The rows W, R, S translate that route into StreamName, RegSeqRat, and Real-facing readback, while H, L, P, N transport, replay, source, and name it without adding quotient equality or choice. □

Falsifiable prediction: Every $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ completion read must display finite-net rows, Cauchy-net refinement, MetricCompletion routing, CompleteMetric sealing, StreamName windows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: finite-epsilon-net, metriccompletion, completemetric, streamname, regseqrat, real: $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite total-bounded nets, Cauchy-net refinement, completion seals, readback rows, transport, replay, provenance, and local NameCert data in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}$ packet is generated from bounded metric source rows, finite-net rows, Cauchy-net refinement rows, MetricCompletion routes, CompleteMetric seals, total-bounded approximation rows, StreamName windows, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6538.1, Definition 6538.2, and Definition 6538.3 seed the finite total-bounded completion route from net data to completion-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveTotalBoundedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient completion equality, selected representatives, countable choice, ambient completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, Cauchy-net refinement, MetricCompletion routing, CompleteMetric sealing, StreamName window replay, RegSeqRat readback, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveTotalBoundedCompletionUp.

Chapter 6539

A Concrete Naming Certificate for AscoliEquicontinuityCriterion$^{\uparrow}$

Following Arzela and Ascoli, and the Bishop–Bridges finite-net reading of compact metric continuity, the AscoliEquicontinuityCriterion$^{\uparrow}$ packet records the criterion by which compact finite-net data and equicontinuity data are displayed before a uniform-modulus route is accepted. It sits between the compact source of chapter 129, the family-radius surface of chapter 4076, the uniform-modulus consumer of chapter 369, the finite epsilon-net ledger of chapter 1767, and the metric carrier discipline of chapter 57. The chapter names the finite handoff criterion only; it does not assert open-cover compactness, a host function-space compactness theorem, quotient equality of maps, selected subsequences, countable choice, or ambient completeness.

Definition 6539.1 (Ascoli equicontinuity criterion carrier)**.** *An* AscoliEquicontinuityCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, E, U, F, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through* CompactMetric$^{\uparrow}$ *and the underlying* Met$^{\uparrow}$ *distance classifier.*
2. *E is the* Equicontinuity$^{\uparrow}$ *family-radius row, displaying one source-radius ledger for the inspected family at each requested target tolerance.*
3. *U is the* UniformModulus$^{\uparrow}$ *handoff row, exposed only after the compact source and equicontinuity rows have been displayed.*
4. *F is the* FiniteEpsilonNet$^{\uparrow}$ *row, carrying the finite center bundle, membership witnesses, and coverage-distance ledger for the selected tolerance scale.*
5. *M records the metric comparison rows over centers and arbitrary inspected points, including the triangle and tolerance-budget data used to pass from finite centers to the whole compact source.*
6. *H records componentwise* hsame *transport for compact source, equicontinuity, uniform-modulus, finite-net, metric, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a criterion request through the finite route K, E, F, M, U.*
8. *P is the* Pkg *provenance over* AscoliEquicontinuityCriterion$^{\uparrow}$, CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. No coordinate stores an arbitrary open cover, a host compactness proof for a function space, quotient equality of functions, a selected subsequence, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.AscoliEquicontinuityCriterionUp.AscoliEquicontinuityCriterionTasteGate`
`single carrier alignment`

Theorem 6539.2 (Ascoli equicontinuity criterion NameCert obligations)**.** *For every accepted* $\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}$ *carrier* $A = (K, E, U, F, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric row, equicontinuity family row, uniform-modulus handoff row, finite epsilon-net row, metric comparison row, componentwise transport, displayed replay, provenance, and local naming; finite-net coverage for the compact source; equicontinuity visibility over the same tolerance request; metric triangle transport from net centers to inspected points; and non-escape for open-cover compactness, host function-space compactness, quotient equality, selected subsequences, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 6539.1. The finite BHist packet exposes exactly $K, E, U, F, M, H, C, P, N$, so carrier admission and local naming have no hidden coordinate beyond the displayed rows.

The compact source is read through K and F. The $\mathsf{CompactMetric}^{\uparrow}$ row supplies the compact metric discipline, while the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ row displays the center bundle and coverage witnesses for the tolerance scale that the criterion will use.

The family-radius part is read through E. The $\mathsf{Equicontinuity}^{\uparrow}$ row is tied to the same tolerance request as the finite-net row, so the accepted packet cannot choose a separate radius ledger after the centers have been selected.

The metric comparison row M transports estimates from centers to arbitrary inspected points. It supplies the triangle and tolerance-budget rows needed before U may be read as a $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Finally H, C, P, N transport, replay, source, and name only this route. Since the carrier has no coordinate for open-cover compactness, host function-space compactness, quotient equality, selected subsequences, countable choice, or ambient completeness, those objects are outside the local NameCert surface. □

Theorem 6539.3 (Ascoli equicontinuity criterion finite-net handoff)**.** *Every accepted* $\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}$ *consumer that exports a uniform-modulus read must first display the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Equicontinuity}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow}$$

inside the carrier $A = (K, E, U, F, M, H, C, P, N)$*. The handoff exports exactly the compact source row, family-radius row, finite center bundle, metric comparison ledger, structural transport, replay, provenance, and local naming rows, and it exports no open-cover compactness, host function-space compactness, quotient equality of maps, selected subsequence, countable-choice schedule, or ambient completeness principle.*

Proof. Start with Definition 6539.2. It lists the accepted rows available to a uniform-modulus consumer and requires the compact metric source, equicontinuity row, finite epsilon-net row, metric comparison row, and structural rows to be present.

Read the finite-net handoff through F. The center bundle and coverage witnesses are finite, and the source metric comparison rows in M are the only route from a displayed center estimate to an arbitrary inspected source point.

Read the equicontinuity ledger through E before the uniform-modulus row U. The same tolerance request is visible to E, F, and M, so the consumer receives one finite radius-and-center budget rather than an unrecorded analytic compactness argument.

The replay row C then orders the handoff as K, E, F, M, U, while H, P, N preserve transport, provenance, and local naming for the same route. Because the carrier omits all host compactness, quotient, selection, choice, and ambient-completeness coordinates, the uniform-modulus read is exhausted by the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}$ uniform-modulus consumer first displays CompactMetric, Equicontinuity, FiniteEpsilonNet, MetricSpace, BHist, hsame, Cont, Pkg, and local NameCert rows before it may read the UniformModulus handoff.

Independence witness: compactmetric, equicontinuity, uniformmodulus, finite-epsilon-net, metric: no listed sibling is bijective to $\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}$ because only this packet jointly binds compact-source coverage, family-radius visibility, finite-net centers, metric triangle transport, uniform-modulus handoff, componentwise transport, replay, provenance, and local naming in one criterion carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}}$.

Constructive story: An $\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}$ packet is generated from a CompactMetricUp source row, an EquicontinuityUp family-radius row, a UniformModulusUp handoff row, a FiniteEpsilonNetUp center bundle, MetricSpaceUp comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6539.1, Definition 6539.2, and Definition 6539.3 seed the finite Ascoli equicontinuity criterion route from compact finite nets and family equicontinuity into a uniform-modulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliEquicontinuityCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host function-space compactness, quotient equality of maps, selected subsequences, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, equicontinuity visibility, metric triangle transport, uniform-modulus handoff exactness, non-escape, provenance, replay, and local naming for BEDC.Derived.AscoliEquicontinuityCriterionUp.

Chapter 6540

A Concrete Naming Certificate for $\mathsf{ArithmetizedProvability}^{\uparrow}$

Following Gödel's arithmetization of syntax and Kleene's treatment of primitive recursive proof predicates, the $\mathsf{ArithmetizedProvability}^{\uparrow}$ packet records the finite BEDC surface by which formula codes, proof codes, derivation checks, substitution rows, and diagonal-code rows may be read without importing a host truth predicate. It sits after the unary arithmetic carrier of chapter 28, the formula-code interface of chapter 197, the bounded derivation-check surface of chapter 199, and the open-meta-loop boundary of chapter 674. The chapter names the proof-predicate dependency consumed by reflection-limit work; it does not assert the incompleteness theorem, a semantic truth definition, consistency of an ambient theory, proof completeness, or a global model-theoretic interpretation.

Definition 6540.1 (Arithmetized provability carrier)**.** *An* $\mathsf{ArithmetizedProvability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (N, F, D, S, G, H, C, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *numeral-code row, carrying unary natural-number names for symbols, formula indices, proof-line indices, and finite proof-code lengths.*
2. *F is the* $\mathsf{FirstOrder}^{\uparrow}$ *formula-code row, displaying symbol codes, variable rows, term rows, formula constructors, and the well-formed formula classifier used by the proof predicate.*
3. *D is the* $\mathsf{Computable}^{\uparrow}$ *derivation-check row, recording the bounded primitive-recursive inspection of a finite proof-code list against the displayed rule and axiom-code table.*
4. *S is the substitution row, recording capture-aware replacement of numeral codes into formula-code slots and the finite compatibility ledger for free-variable bookkeeping.*
5. *G is the diagonal-code row, displaying the self-application code produced from F and S while keeping the resulting formula code inside object syntax.*
6. *H records componentwise* hsame *transport for numeral codes, formula codes, derivation checks, substitution rows, diagonal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a provability request through N, F, D, S, G and then through the structural rows H, P, L.*

8. *P is the* Pkg *provenance over* $\mathsf{ArithmetizedProvability}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, $\mathsf{GroundLoopBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{ArithmetizedProvability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores a host truth predicate, semantic satisfaction relation, completeness theorem, consistency proof for the represented theory, or unbounded meta-level proof search.

Theorem 6540.2 (ArithmetizedProvability NameCert obligations)**.** *For every accepted* $\mathsf{ArithmetizedProvability}^{\uparrow}$ *carrier $A = (N, F, D, S, G, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{ArithmetizedProvability}^{\uparrow}}$ *surface consists exactly of carrier admission; numeral-code visibility; formula-code exactness for the displayed first-order constructors; proof-code induction over the finite derivation rows inspected by D; substitution stability for the rows in S; diagonal-code stability for G; derivation-check replay through C; componentwise transport through H; provenance through P; local naming through L; and non-escape for host truth, semantic completeness, ambient consistency, and unbounded meta-level proof search.*

Proof. Project the carrier of Definition 6540.1. The finite BHist packet exposes exactly $N, F, D, S, G, H, C, P, L$, so carrier admission and local naming have no hidden coordinate beyond the displayed rows.

Read the syntactic substrate through N and F. The numeral-code row gives the finite natural-number codes used by the packet, and the formula-code row gives the first-order constructors and well-formedness classifier that those numbers index.

Read the proof predicate through D. Its accepted rows are finite derivation-check rows, so the proof-code obligation is induction over the displayed proof-line list rather than a semantic appeal to truth or completeness.

Read substitution and diagonalization through S and G. The substitution row records the capture-aware replacement ledger, and the diagonal-code row records only the object-level self-application code produced from that ledger.

Finally H, C, P, L transport, replay, source, and name this same finite route. Since no coordinate stores a host truth predicate, semantic satisfaction relation, completeness theorem, ambient consistency proof, or unbounded proof search, those objects cannot be projected from the local certificate. □

Theorem 6540.3 (ArithmetizedProvability finite proof-code induction)**.** *Every accepted* $\mathsf{ArithmetizedProvability}^{\uparrow}$ *proof-code read is generated by induction over the displayed finite derivation rows in D and is then transported only through*

$$N, F, D, S, G, H, C, P, L.$$

The read may certify that a displayed code is accepted by the finite derivation checker for a displayed formula code. It cannot import host truth, semantic completeness, an ambient consistency theorem, or a meta-level assertion that the represented formula is true.

Proof. Begin with the derivation-check row D. A proof-code read is accepted only when the finite list of proof lines passes the displayed rule and axiom-code checks, so the generating argument proceeds by induction over that finite list.

At the base row, the checker reads an initial displayed axiom-code or hypothesis-code entry through N and F. The entry remains a syntactic formula-code row.

At each successor row, the checker reads only the earlier displayed lines named by the finite indices in N and the rule-shape data in F. The row added at that step is therefore produced by the same finite derivation-check surface, not by an external semantic oracle.

Substitution and diagonal rows are handled after the syntactic check. The row S records the capture-aware substitution budget, while G records the diagonal code that remains inside the formula-code surface.

The structural rows H, C, P, L then transport, replay, source, and name the result. Because no later row adds truth, completeness, consistency, or unbounded proof search, the accepted proof-code read is exhausted by the finite induction route stated above. □

Falsifiable prediction: Every accepted $\mathsf{ArithmetizedProvability}^{\uparrow}$ consumer first displays Nat, FirstOrder, Computable, GroundLoopBoundary, BHist, hsame, Cont, Pkg, and local NameCert rows before it may read the arithmetized proof predicate.

Independence witness: nat, firstorder, computable, groundloopboundary: no listed sibling is bijective to $\mathsf{ArithmetizedProvability}^{\uparrow}$ because only this packet jointly binds numeral codes, formula codes, derivation-check rows, substitution rows, diagonal-code rows, componentwise transport, replay, provenance, and local naming while refusing host truth and semantic completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmetizedProvability}^{\uparrow}}$.

Constructive story: An $\mathsf{ArithmetizedProvability}^{\uparrow}$ packet is generated from NatUp numeral codes, FirstOrderUp formula codes, ComputableUp finite derivation-check rows, substitution and diagonal-code rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArithmetizedProvability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6540.1, Definition 6540.2, and Definition 6540.3 seed the finite proof-predicate surface consumed by reflection-limit and incompleteness-boundary work.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArithmetizedProvability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Gödel incompleteness, Tarski truth undefinability, semantic completeness, ambient consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formula-code exactness, proof-code induction, substitution stability, derivation-check replay, truth-predicate non-escape, provenance, replay, and local naming for BEDC.Derived.ArithmetizedProvabilityUp.

Chapter 6541

A Concrete Naming Certificate for $\mathsf{MackeyDecomposition}^{\uparrow}$

Following Mackey's subgroup theorem, Serre's finite-group representation theory, and the induction–restriction calculus in Fulton–Harris, the $\mathsf{MackeyDecomposition}^{\uparrow}$ packet records the double-coset decomposition row that compares restricting an induced representation with the finite sum of conjugated subgroup inductions. It consumes the group and subgroup sources of chapter 41 and chapter 80, the representation rows of chapter 247, chapter 245, and chapter 246, the vector-space endpoint of chapter 47, the Frobenius comparison of chapter 5806, and the orthogonality ledger of chapter 1603. The packet names only finite source rows, double-coset representatives, conjugation transport, induced summand rows, character-pairing checks, replay, provenance, and local naming; it does not import a host representation category, quotient double-coset space, semisimplicity theorem, or completed character table.

Definition 6541.1 (MackeyDecomposition carrier)**.** *A* $\mathsf{MackeyDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{M} = (G, H, R, I, C, F, S, D, L, T, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *certificate row for the finite group endpoint.*
2. *H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, including the subgroup inclusion read used by restriction.*
3. *R is the* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger row that records addition of induced summands.*
4. *I is the* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row for the source representation.*
5. *C is the* $\mathsf{CharacterTheory}^{\uparrow}$ *character-pairing row used to audit the displayed decomposition.*
6. *F is the* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *comparison row binding induction and restriction.*
7. *S is the* $\mathsf{SchurOrthogonality}^{\uparrow}$ *row that checks the finite character-pairing ledger for the summands.*
8. *D is the double-coset representative ledger, listing only displayed representatives and their subgroup-intersection rows.*
9. *L is the summand ledger for the induced representations attached to those representatives.*
10. *T transports the displayed subgroup-intersection and summand rows componentwise by* hsame *and displayed* Cont *replay.*

11. *P is the* Pkg *provenance over* $\mathsf{MackeyDecomposition}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{FrobeniusReciprocity}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MackeyDecomposition}^{\uparrow}}$ *row.*

The carrier exports only this finite double-coset ledger. It contains no host quotient of $H\backslash G/K$, no ambient representation category, no selected basis of an induced module, and no completed class-function algebra.

Lean4 (statement) `BEDC.Derived.MackeyDecompositionUp`

Theorem 6541.2 (MackeyDecomposition NameCert obligations)**.** *For every accepted* $\mathsf{MackeyDecomposition}^{\uparrow}$ *carrier $\mathcal{M} = (G, H, R, I, C, F, S, D, L, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MackeyDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed group, subgroup, representation-ring, induced-representation, character-theory, Frobenius, Schur, double-coset, summand, transport, provenance, and naming rows;*
2. *double-coset admission, requiring every representative read in D to expose its* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{Subgroup}^{\uparrow}$ *source rows before any summand in L is consumed;*
3. *induced-summand exactness, requiring every summand row in L to factor through the* $\mathsf{InducedRep}^{\uparrow}$ *row I, the representation-ring row R, and the vector-space endpoint named in P;*
4. *character-pairing audit, requiring the* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{FrobeniusReciprocity}^{\uparrow}$, *and* $\mathsf{SchurOrthogonality}^{\uparrow}$ *rows C, F, S to inspect only the displayed summands and double-coset representatives;*
5. *non-escape, excluding host double-coset quotients, complete representation categories, semisimplicity decompositions, character-table computation, and selected bases.*

No local obligation is stored outside $\mathcal{M}$.

Proof. Project the carrier of Definition 6541.1. The displayed coordinates are precisely the consumed source rows G, H, R, I, C, F, S, the double-coset and summand rows D, L, and the structural rows T, P, N.

The double-coset ledger is finite by construction of the packet. Each representative read is tied to the group and subgroup rows before the corresponding summand can be inspected, so the summand exactness clause is a projection through I, R and the vector-space source named by P.

The character audit reads only the displayed summands. The rows C, F, S compare character pairings, induction–restriction, and orthogonality along the visible ledger, while T transports and replays the same finite route.

Since no coordinate stores a quotient double-coset carrier, host representation category, semisimplicity decomposition, character table, or selected basis, the listed rows exhaust the local NameCert surface. □

Theorem 6541.3 (MackeyDecomposition double-coset ledger)**.** *Every decomposition read exported by an accepted* $\mathsf{MackeyDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Group}^{\uparrow}, \mathsf{Subgroup}^{\uparrow} \longmapsto D \longmapsto L \longmapsto \mathsf{InducedRep}^{\uparrow}, \mathsf{RepresentationRing}^{\uparrow}, \mathsf{CharacterTheory}^{\uparrow}, \mathsf{FrobeniusReciprocity}^{\uparrow}, \mathsf{Sch}$$

Thus a downstream consumer may read the Mackey decomposition only as a displayed double-coset representative ledger and its induced summand ledger. It cannot replace those rows by a host quotient $H\backslash G/K$, an ambient representation category, a selected module basis, a semisimplicity theorem, or a character table.

Proof. Begin with a decomposition request. The source rows available to it are the group row G and subgroup row H, and the only double-coset coordinate is the displayed finite ledger D.

The summands are downstream from D. Each entry of L is read only after the corresponding representative and subgroup-intersection data have appeared, and the induced-representation and representation-ring rows I, R consume exactly those displayed entries.

The character-facing checks do not enlarge the route. The rows C, F, S audit the same summands by character pairing, Frobenius reciprocity, and Schur orthogonality, while T, P, N replay, source, and name the packet.

Therefore every exported decomposition read remains inside the finite ledger route stated above, with no host quotient, ambient category, selected basis, semisimplicity theorem, or character-table computation. □

Closure status for $\mathsf{NameCert}_{\mathsf{MackeyDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MackeyDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6541.1, Definition 6541.2, and Definition 6541.3 seed the finite Mackey double-coset decomposition packet over displayed group, subgroup, induced-representation, character-theory, representation-ring, vector-space, Frobenius reciprocity, and Schur orthogonality rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MackeyDecomposition}\uparrow}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MackeyDecompositionUp`

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, host double-coset quotients, complete representation categories, semisimplicity, selected bases, or character-table computation.

Upgrade path: Obligation closure requires carrier admission, double-coset admission, induced-summand exactness, character-pairing audit, transport stability, provenance exhaustion, and local naming for the displayed packet.

Constructive story: A $\mathsf{MackeyDecomposition}^{\uparrow}$ packet is generated from displayed group and subgroup rows, representation-ring and induced-representation rows, character-theory, Frobenius reciprocity, and Schur orthogonality rows, a finite double-coset representative ledger, an induced summand ledger, componentwise transport and replay, provenance, and local NameCert evidence.

Chapter 6542

A Concrete Naming Certificate for $\mathsf{DoobConvergence}^{\uparrow}$

Following Doob, the $\mathsf{DoobConvergence}^{\uparrow}$ packet names the finite convergence-readout surface that sits beside the martingale convergence packet of chapter 2246. It consumes the probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, Doob-upcrossing, martingale-convergence, rational, and real certificates of chapter 184, chapter 185, chapter 188, chapter 190, chapter 344, chapter 383, chapter 1629, chapter 1630, chapter 2246, chapter 37, and chapter 38. The packet records the bounded finite NameCert obligations needed to read a convergence-admission row; it does not introduce measure completion, almost-sure quotient equality, unbounded stopping choice, or a completed limiting-event carrier.

Definition 6542.1 (Doob convergence carrier)**.** *A* $\mathsf{DoobConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Omega, X, E, M, T, O, U, B, A, Q, R, L, W, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the consumed* $\mathsf{ProbSpace}^{\uparrow}$ *row.*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint-family row for the displayed martingale window.*
3. E *is the* $\mathsf{CondExp}^{\uparrow}$ *successor-face row used by the martingale law.*
4. M *is the finite* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet.*
5. T *is the* $\mathsf{StoppingTime}^{\uparrow}$ *bounded index row.*
6. O *is the* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window row.*
7. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count ledger.*
8. B *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *bound row consumed by the convergence readout.*
9. A *is the* $\mathsf{MartingaleConvergence}^{\uparrow}$ *finite convergence-admission row.*
10. Q *is the* $\mathsf{Rat}^{\uparrow}$ *rational tolerance ledger used to read finite windows.*
11. R *is the* $\mathsf{Real}^{\uparrow}$ *boundary and bounded-window row.*
12. L *is the Doob convergence readout ledger recording which finite endpoint windows, rational tolerances, real bounds, upcrossing rows, and admission rows may be inspected together.*

13. *W records the local source alignment among* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{MartingaleConvergence}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.
14. *H records componentwise* hsame *transport for all displayed rows.*
15. *C records the displayed* Cont *routes from finite endpoint windows through bounded stopping, upcrossing, Doob-bound, martingale-convergence, rational, real-boundary, and convergence-readout rows.*
16. *P is the* Pkg *provenance over the displayed rows,* BHist, hsame, Cont, *and* NameCert.
17. *N is the local* $\mathsf{NameCert}_{\mathsf{DoobConvergence}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It has no coordinate for measure completion, almost-sure quotient classes, unbounded stopping-time selection, pathwise limiting events, Brownian specialization, or an ambient convergence theorem.

Theorem 6542.2 (Doob convergence NameCert obligations)**.** *For the carrier of Definition 6542.1, the local* $\mathsf{NameCert}_{\mathsf{DoobConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$\Omega, X, E, M, T, O, U, B, A, Q, R, L, W, H, C, P, N,$$

probability and endpoint exposure through Ω, X, E, M, bounded stopping exposure through T, O, upcrossing and Doob-bound exposure through U, B, martingale-convergence admission through A, rational and real readback through Q, R, convergence-readout exactness through L, source alignment through W, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.

Proof. Project the carrier of Definition 6542.1. Its rows list the consumed probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, Doob-upcrossing, martingale-convergence, rational, real, readout, source-alignment, transport, replay, provenance, and naming data.

The martingale-facing rows are Ω, X, E, M, T, O. They are consumed only as finite probability, endpoint, conditional-expectation, martingale, and bounded stopped-window packets.

The crossing-facing rows are U, B, A. The upcrossing row and Doob-bound row must be displayed before the martingale-convergence admission row can be read, matching the sibling route through chapter 2246.

The readback rows are Q, R, L, W, H, C, P, N. They record rational tolerance, real boundary, finite convergence readout, source alignment, componentwise transport, replay, provenance, and local naming. Since no remaining coordinate names measure completion, almost-sure quotient equality, unbounded stopping choice, or a limiting-event carrier, the NameCert obligations are exactly the displayed finite rows. □

Theorem 6542.3 (Doob convergence bounded readout nonescape)**.** *Every accepted* $\mathsf{DoobConvergence}^{\uparrow}$ *carrier reads a convergence-facing row only through the finite handoff from* $\mathsf{MartingaleConvergence}^{\uparrow}$*: bounded stopped windows, lower and upper* $\mathsf{Real}^{\uparrow}$ *boundary rows,* $\mathsf{Rat}^{\uparrow}$ *tolerance rows, consumed* $\mathsf{Upcrossing}^{\uparrow}$ *and* $\mathsf{DoobUpcrossing}^{\uparrow}$ *ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. A consumer that replaces this route by measure completion, almost-sure quotient equality, unbounded stopping choice, or an ambient limiting event leaves the accepted packet.*

Proof. Open the accepted packet and project A, L, Q, R, U, B. The convergence-admission row A is read only together with the readout ledger L, rational tolerance row Q, real boundary row R, crossing-count row U, and Doob-bound row B.

Use Definition 2246.2 and Definition 2246.6. They identify the martingale-convergence handoff as a finite bounded-window route over stopping, upcrossing, rational-readback, and real-boundary rows, not as an ambient limiting theorem.

Finally apply Definition 6542.2. It adds the local source-alignment, transport, replay, provenance, and naming rows W, H, C, P, N and confirms that the $\mathsf{DoobConvergence}^{\uparrow}$ carrier has no coordinate for measure completion, almost-sure quotient equality, unbounded stopping choice, or a limiting-event carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{DoobConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoobConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6542.1, Definition 6542.2, and Definition 6542.3 seed the finite Doob convergence packet over probability, random-variable, conditional-expectation, martingale, bounded stopping, upcrossing, Doob-bound, martingale-convergence, rational, real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoobConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, almost-sure quotient equality, unbounded stopping choice, completed limiting-event carriers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite handoff exactness, bounded readout nonescape, and local NameCert accountability for BEDC.Derived.DoobConvergenceUp.

Constructive story: A $\mathsf{DoobConvergence}^{\uparrow}$ packet is generated from probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, Doob-upcrossing, martingale-convergence, rational, real-boundary, transport, continuation, provenance, and local NameCert rows.

Chapter 6543

A Concrete Naming Certificate for $\mathsf{MetrizableSpace}^{\uparrow}$

Following Bourbaki and Kelley-style topology, the $\mathsf{MetrizableSpace}^{\uparrow}$ packet records the finite BEDC handoff by which a displayed topology may expose a metric whose ball-neighbourhood rows answer the same local open requests. It sits between the topology carrier of chapter 88, the metric-space surface of chapter 4230, the finite $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ distance readback route of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a metrizable-space interface, not a Urysohn metrization theorem, open-cover compactness, quotient equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6543.1 (MetrizableSpace carrier)**.** *A* $\mathsf{MetrizableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, M, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying open-row classifiers, point-neighbourhood requests, and finite membership reads.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *metric row over the same displayed point carrier.*
3. *B is the ball-neighbourhood basis row. It records, for each displayed point and tolerance request, the metric ball row whose inclusion and refinement reads answer the topology row T.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a distance request is inspected by finitely many named observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distance stream generated from W and the metric row M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the distance value, reachable only after M, B, W, R have been displayed.*
7. *H records componentwise* hsame *transport for topology rows, metric rows, ball-basis rows, distance windows, regular readback, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a topological neighbourhood or distance request through $T, M, B, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetrizableSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetrizableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Urysohn separation hypotheses, open-cover compactness, quotient point equality, selected metric representatives, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetrizableSpaceUp.MetrizableSpaceTasteGate single carrier alignment`

Theorem 6543.2 (MetrizableSpace NameCert obligations)**.** *For every accepted* $\mathsf{MetrizableSpace}^{\uparrow}$ *carrier* $Z = (T, M, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetrizableSpace}^{\uparrow}}$ *surface consists exactly of topology carrier admission, metric row admission, ball-neighbourhood basis visibility, finite distance-window inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *distance readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 6543.1. The topology source, metric row, ball-basis row, distance window, regular readback, Real seal, transport, replay, provenance, and local naming row are the only displayed coordinates.

The topological side is read through T and B. A neighbourhood request must first name the open-row classifier in T and then read the ball-basis witness that relates the displayed metric ball to that topology row.

The metric side is read through M, W, R, E. A distance request cannot reach a Real value until the metric row has been displayed, the finite window has been inspected, the regular rational readback has been produced, and the terminal Real seal has been exposed.

Rows H, C, P, N only transport, replay, source, and name the same finite packet. Since no remaining coordinate stores the excluded objects, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetrizableSpaceUp.MetrizableSpaceCarrier namecert obligations`

Theorem 6543.3 (MetrizableSpace metric-topology handoff)**.** *Every topology-facing or metric-facing consumer of an accepted* $\mathsf{MetrizableSpace}^{\uparrow}$ *packet factors through*

$$\mathsf{Topology}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The topology read receives ball-neighbourhood basis rows tied to T, while the metric read receives distance windows, regular readback, and a terminal Real seal tied to M. The handoff exports no Urysohn metrization theorem, open-cover compactness, quotient equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the topology-facing request. The carrier exposes topology only through T, and the basis row B is the only row that answers neighbourhood requests by metric balls.

For a metric-facing request, read M before the distance window W. The window is then converted by R into a regular rational distance readback and sealed by E as the only Real-facing endpoint.

Transport and replay through H and C preserve the same route, while P and N record provenance and local naming. The carrier contains no row for the excluded global theorems or selectors, so the handoff is exactly the finite BEDC route displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetrizableSpace}^{\uparrow}}$**.**

Constructive story: A MetrizableSpace$^{\uparrow}$ packet is generated from a TopologyUp source row, MetricSpaceUp metric row, ball-neighbourhood basis row, StreamNameUp finite distance windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetrizableSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6543.1, Definition 6543.2, and Definition 6543.3 seed the finite topology-to-metric handoff over topology rows, metric rows, ball-neighbourhood basis rows, distance windows, regular readback, Real seals, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetrizableSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Urysohn metrization, open-cover compactness, quotient equality, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ball-neighbourhood basis visibility, finite distance-window inspection, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.MetrizableSpace

Chapter 6544

A Concrete Naming Certificate for HyperbolicIdealTriangle$^{\uparrow}$

Following the classical Poincare-disk treatment of ideal triangles in hyperbolic geometry, the HyperbolicIdealTriangle$^{\uparrow}$ packet records the finite BEDC surface for three displayed boundary ends, their geodesic sides, the disk metric route, and the boundary-flow handoff. It concretizes the fixed-end route of section 9463.5 in chapter 9463. The packet sits after HyperbolicMetric$^{\uparrow}$ in chapter 3474, HyperbolicGeodesicBoundary$^{\uparrow}$ in chapter 3491, UnitDisk$^{\uparrow}$ in chapter 377, PoincareDiskPhaseTransport$^{\uparrow}$ in chapter 3455, and DynSystem$^{\uparrow}$ in chapter 195. It does not complete the disk, quotient ideal endpoints, select global geodesic representatives, import host analytic flow, or identify the observer-expansion reading with physical spacetime.

Definition 6544.1 (Hyperbolic ideal triangle carrier)**.** *A* HyperbolicIdealTriangle$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (U, M, B, G, T, D, F, J, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UnitDisk$^{\uparrow}$ *disk-coordinate row for the finite interior probes and boundary-end reads inspected by the packet.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the* HyperbolicGeodesicBoundary$^{\uparrow}$ *row naming three distinct boundary-end positions and the finite incidence data used to form ideal sides.*
4. *G is the geodesic-side row, recording the three boundary-to-boundary side reads as displayed boundary geodesic data.*
5. *T is the* PoincareDiskPhaseTransport$^{\uparrow}$ *transport row carrying disk-surface movement and boundary provenance.*
6. *D is the* DynSystem$^{\uparrow}$ *finite flow row used when a boundary-flow consumer replays movement along the displayed sides.*
7. *F is the fixed-end row stating that the three boundary commitments remain the same displayed far-end rows along the routed flow.*
8. *J is the classifier-stability row for disk coordinates, metric reads, boundary incidence, side rows, phase transport, routed flow, and fixed-end data.*

9. *H records componentwise* hsame *transport for disk, metric, boundary, side, transport, flow, fixed-end, classifier-stability, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an ideal-triangle or boundary-flow request through $U, M, B, G, T, D, F, J, H,$*
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicIdealTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a completed hyperbolic plane, quotient ideal-boundary equality, selected global geodesics, host analytic flow, physical cosmology, or observer substance.

Theorem 6544.2 (Hyperbolic ideal triangle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$ *carrier $I = (U, M, B, G, T, D, F, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicIdealTriangle}^{\uparrow}}$ *surface is exhausted by disk-coordinate admission, hyperbolic metric visibility, boundary-end incidence for three displayed ends, geodesic-side visibility, Poincare disk phase transport, finite dynamic-flow replay, fixed-end stability, classifier stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6544.1. The rows $U, M, B, G, T, D, F, J, H, C, P, N$ are the only displayed coordinates.

The geometric read begins with U, M, B, G. Disk coordinates and metric reads are visible before boundary incidence is used, and the three side rows are built only from the displayed boundary data.

The boundary-flow read uses T, D, F, J. Phase transport and finite flow replay carry the same boundary commitments, while classifier stability keeps disk, metric, boundary, side, transport, flow, and fixed-end rows attached to the same packet.

Rows H, C, P, N transport, replay, source, and name the finite carrier. Since no row stores the excluded completed or host-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 6544.3 (Hyperbolic ideal triangle boundary-flow handoff)**.** *Every boundary-flow consumer of an accepted* $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto \mathsf{DynSyster}$$

The consumer receives three displayed boundary ends, their geodesic-side rows, phase transport, finite routed flow, fixed-end stability, classifier stability, transport, replay, provenance, and local naming. It obtains no completed disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance.

Proof. Begin with a boundary-flow request. The disk-coordinate row U and metric row M provide the only geometric source data available to the request.

Read B and G next. The three boundary ends and the three ideal sides are displayed as finite incidence and geodesic rows, not as quotient endpoints or selected global geodesics.

Use T, D, F, J for the handoff. The Poincare disk transport row and finite dynamic-flow row replay movement while preserving the fixed-end commitments and classifier readings.

Finally replay the structural rows H, C, P, N. They preserve the same finite packet, and the carrier has no coordinate for the excluded completed or host-level objects, so the boundary-flow handoff cannot leave the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicIdealTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$ packet is generated from UnitDiskUp coordinate rows, HyperbolicMetricUp metric reads, HyperbolicGeodesicBoundaryUp boundary incidence, three geodesic-side rows, PoincareDiskPhaseTransportUp transport, DynSystemUp finite flow replay, fixed-end stability, classifier stability, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicIdealTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6544.1, Definition 6544.2, and Definition 6544.3 seed the finite ideal-triangle route over disk, metric, boundary, geodesic-side, phase-transport, flow, fixed-end, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicIdealTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic disks, quotient ideal-boundary equality, selected global geodesics, host analytic flows, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk visibility, metric-read visibility, boundary incidence, geodesic-side visibility, phase-flow routing, fixed-end stability, classifier stability, ledger nonescape, and local naming for BEDC.Derived.HyperbolicIdealTr

Chapter 6545

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$ packet records the Bishop regular completion reflector as finite BEDC naming data over explicit rational approximants. It sits after the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the RegSeqRat reflector route of chapter 3398. The packet names only the finite reflector surface visible to Bishop regular-name consumers; it does not identify quotient completions, select limits, compare host real values, use countable choice, assert ambient completeness of $\mathsf{Real}^{\uparrow}$, or provide Lean verification or bridge data.

Definition 6545.1 (BishopRegularCompletionReflector carrier)**.** *A* $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for every regular precision request and every comparison of inspected rational windows.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the currently inspected approximants of the Bishop regular name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, certifying that the displayed windows satisfy the regular rational discipline required before completion is requested.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from D, S, R through the displayed completion surface rather than through a quotient of streams.*
5. *F is the reflector row, recording that any completion-facing consumer must read the dyadic ledger, stream windows, regular readback, and real seal in that order.*
6. *H records componentwise* hsame *transport for dyadic tolerances, stream windows, regular readbacks, real seals, reflector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a regular-name precision request through*

$$D, S \longmapsto R \longmapsto E \longmapsto F \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, F, H, C, P, N$. It has no coordinate for quotient completion equality, selected limit representatives, host equality of real values, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge checking.*

Theorem 6545.2 (BishopRegularCompletionReflector NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$ *carrier*

$$B = (D, S, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionReflector}^{\uparrow}}$ *surface consists exactly of carrier admission for dyadic tolerance ledgers, finite stream windows, regular rational readback, real-name sealing, reflector routing, transport, replay, provenance, and local naming; source exposure requiring every completion-facing read to inspect D, S before R; reflector discipline requiring E and F to be read only through the same D, S, R route; and nonescape excluding quotient completion, selected limits, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard bridge checking.*

Proof. Project the accepted carrier. Its displayed coordinates are exactly $D, S, R, E, F, H, C, P, N$, so carrier admission, transport, replay, provenance, and local naming are read from those rows.

The source side begins with D and S. The dyadic ledger fixes the tolerances used by each regular request, while the stream-window row specifies the finite approximants available for the current read.

The regular readback row R is downstream of those source rows. A completion-facing consumer therefore obtains a regular rational name only after the dyadic and stream data have been displayed.

The seal and reflector rows are E and F. Both are indexed by the same D, S, R route, so the carrier cannot replace them by a quotient completion, a selected limit, or a host-level equality witness.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since no other coordinate is present, the listed rows exhaust the local NameCert surface and the refused principles have no export path. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletionReflector}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, a reflector route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRegularCompletionReflector}^{\uparrow}}$, seed)

Scope closed: Definition 6545.1 and Definition 6545.2 seed the Bishop regular completion reflector packet over dyadic tolerances, finite windows, regular rational readback, and a Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRegularCompletionReflector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance coverage, StreamName window exposure, RegSeqRat readback, Real seal preservation, reflector routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletionReflectorUp.

Chapter 6546

A Concrete Naming Certificate for $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ packet records the finite oscillation boundary where a regular rational Cauchy name is allowed to hand a Real-facing consumer only the displayed tail-variation data. It sits over the dyadic arithmetic core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the Cauchy oscillation ledger of chapter 448. The packet is a boundary object for L10 oscillation consumers, not Real completeness, quotient stream equality, host real equality, or a selected limit.

Definition 6546.1 (Regular real oscillation boundary carrier)**.** *A* $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, O, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the finite dyadic radius and comparison budget used by the oscillation request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the inspected finite prefix and the later finite refinements admitted for the same name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the displayed stream windows.*
4. *O is the* $\mathsf{CauchyOscillation}^{\uparrow}$ *ledger row recording the finite tail-variation comparisons read from D, S, R.*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dyadic tolerance, stream window, regular readback, and oscillation ledger have all been displayed.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, oscillation, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an oscillation-boundary request through D, S, R, O and then to the terminal seal L.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyOscillation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularRealOscillationBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, O, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, Real completeness, or an ambient completeness theorem.

Theorem 6546.2 (Regular real oscillation boundary NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ *carrier $B = (D, S, R, O, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRealOscillationBoundary}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance, stream window, regular rational readback, oscillation ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *tolerance-window compatibility, requiring D and S to be displayed before any regular rational readback is consumed;*
3. *regular-readback exposure, requiring R to be read only as the finite RegSeqRat route produced from S;*
4. *oscillation-boundary admission, requiring O to consume the same D, S, R rows before any Real-facing seal is read;*
5. *Real seal non-escape, requiring L to seal only the displayed oscillation comparison and not a completed real object, quotient stream class, selected limit, or host equality;*
6. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

No obligation is stored outside those rows.

Proof. Project the carrier of Definition 6546.1. The projection exposes exactly the dyadic tolerance row, finite stream window, regular rational readback, oscillation ledger, Real seal, transport, replay, provenance, and local naming rows.

Read the numerical boundary in order. The dyadic tolerance D and stream window S determine the finite comparison budget; the regular readback R is then the only RegSeqRat row available to the oscillation ledger O.

The seal L is downstream of D, S, R, O. Since the carrier has no quotient stream, selected limit, host equality, choice, or completeness coordinate, the seal records only the displayed oscillation comparison.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Thus the listed rows exhaust the local NameCert surface. □

Theorem 6546.3 (Regular real oscillation boundary L10 handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ *carrier must factor through the finite route*

$$D \longmapsto S \longmapsto R \longmapsto O \longmapsto L \longmapsto H, C, P, N.$$

The handoff gives the consumer a displayed oscillation boundary for regular rational Cauchy names, not Real completeness, quotient equality of streams, host real equality, selected limits, or countable choice.

Proof. Let $B = (D, S, R, O, L, H, C, P, N)$ be accepted. The only tolerance source is D, and the only observation source is the finite window S.

The regular rational readback R is downstream of those rows, and the Cauchy oscillation ledger O reads the same finite data before the Real seal L is available. A consumer therefore reaches L only through the displayed L10 source route.

Componentwise transport and replay through H and C preserve that route, while P and N record the provenance and local name. Since no other coordinate exists, the consumer cannot obtain Real completeness, quotient stream equality, host equality, a selected limit, or countable choice from this packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ reader that claims a Real-facing oscillation handoff while omitting DyadicRatCore tolerance, StreamName windows, RegSeqRat readback, CauchyOscillation ledger, Real seal, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchyoscillation: RegularRealOscillationBou is not bijective to any listed sibling because it jointly carries the dyadic tolerance, stream window, regular readback, oscillation ledger, terminal seal, transport, replay, provenance, and local naming as one boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealOscillationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealOscillationBoundary}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName windows, RegSeqRat readback, a CauchyOscillation ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealOscillationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6546.1, Definition 6546.2, and Definition 6546.3 seed the finite oscillation boundary for RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealOscillationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, Real completeness, quotient stream equality, host real equality, selected limits, countable choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window compatibility, regular readback exposure, oscillation-boundary admission, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RegularRealOscillationBoundaryUp.

Chapter 6547

A Concrete Naming Certificate for MetaCICConfluenceDiamondLedger$^{\uparrow}$

The MetaCICConfluenceDiamondLedger$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as finite NameCert data. It packages typed-window exposure, residual-substitution boundary rows, local-join ledgers, bounded checker rows, obstruction retention, transport, replay, provenance, and local naming before any confluence handoff can be read. It consumes the MetaCIC critical path of chapter 765, the candidate-mediated strong-normalization route of chapter 5163, the critical-pair boundary of chapter 2212, the residual-substitution boundary of chapter 2181, and the local-join ledger of chapter 4126. The packet is a finite confluence-diamond ledger, not a closed Church–Rosser theorem or an unconditional substitution-compatibility proof.

Definition 6547.1 (MetaCIC confluence diamond ledger carrier)**.** *A* MetaCICConfluenceDiamondLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, R, J, B, O, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row exposing the finite MetaCIC source window, redex shape, and bounded context in which the confluence handoff is requested.*
2. *R is the residual-substitution row naming the displayed substitution-boundary data that the diamond route must consume.*
3. *J is the local-join row recording the finite candidate joins and common-reduct witnesses accepted for the inspected critical pairs.*
4. *B is the bounded-checker row recording the finite checker budget for the residual diamond route.*
5. *O is the obstruction row retaining the substitution-compatibility and closed-diamond boundary names that are not discharged by this packet.*
6. *X is the confluence handoff row exposed only after T, R, J, B, O have all been displayed.*
7. *H records componentwise* hsame *transport for the typed window, residual-substitution row, local-join row, bounded checker, obstruction row, handoff row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence-diamond request through T, R, J, B, O and then to X.*

9. *P is the* Pkg *provenance over* MetaCICConfluenceDiamondLedger$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, CandidateMediatedSN$^{\uparrow}$, MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICLocalJoinLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, R, J, B, O, X, H, C, P, N$. It has no coordinate for an unconditional parallel diamond, closed Church–Rosser theorem, global substitution-compatibility proof, unbounded checker, hidden quotient, or host-side confluence oracle.

Theorem 6547.2 (MetaCIC confluence diamond ledger NameCert obligations)**.** *For every accepted* MetaCICConfluenceDiamondLedger$^{\uparrow}$ *carrier $G = (T, R, J, B, O, X, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConflu}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed typed-window, residual-substitution, local-join, bounded-checker, obstruction, handoff, transport, replay, provenance, and local naming rows;*
2. *typed-window exposure, requiring the MetaCIC source window, redex shape, and bounded context to be displayed before any residual row is consumed;*
3. *residual-substitution exposure, requiring R to be read before local joins or bounded checker rows can be trusted;*
4. *local-join accounting, requiring J to list the finite candidate joins and common-reduct witnesses used by the inspected critical pairs;*
5. *bounded-checker accounting, requiring B to record the finite checker budget and to consume the same T, R, J rows;*
6. *obstruction retention, requiring O to keep the substitution-compatibility and closed-diamond boundary names visible rather than silently discharging them;*
7. *handoff non-escape, requiring X to expose only the displayed confluence-diamond ledger and not an unconditional Church–Rosser theorem;*
8. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

The future Lean TasteGate for this packet must supply both a Nontrivial instance, separating nonempty ledger data from the empty carrier, and a FieldFaithful instance, asserting that the classifier reads every displayed field $T, R, J, B, O, X, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceDiamondLedgerUp.MetaCICConfluenceDiamondLedgerNameCertObligations`

Proof. Project the carrier of Definition 6547.1. The projection exposes exactly the typed-window, residual-substitution, local-join, bounded-checker, obstruction, handoff, transport, replay, provenance, and local naming rows.

Read the confluence rows in order. The typed window T fixes the inspected MetaCIC source window; the residual row R records the substitution boundary; the local-join row J records candidate joins; and the bounded checker B accounts for the finite checking budget.

The obstruction row O remains visible before the handoff X. Therefore the packet can display a confluence-diamond ledger without claiming the global substitution-compatibility input or a closed Church–Rosser endpoint.

Finally, H, C, P, N transport, replay, source, and name the same finite route. A Nontrivial TasteGate instance must witness that accepted packets carry genuine ledger data, and a FieldFaithful instance must witness that the classifier cannot ignore any displayed row. □

Theorem 6547.3 (MetaCIC confluence diamond ledger handoff)**.** *Every accepted* MetaCICConfluenceDiamondLedg *carrier that exposes a confluence handoff must display the typed-window, residual-substitution, local-join, bounded-checker, obstruction, transport, replay, provenance, and local NameCert rows on the same route*

$$T \longmapsto R \longmapsto J \longmapsto B \longmapsto O \longmapsto X \longmapsto H, C, P, N.$$

Consequently the handoff is a finite ledger consumed by MetaCIC confluence routes, not an unconditional parallel diamond, not arbitrary closed-term Church–Rosser, not a global substitution-compatibility proof, and not a host confluence oracle.

Proof. Let $G = (T, R, J, B, O, X, H, C, P, N)$ be accepted. The handoff row X is downstream of the typed-window, residual-substitution, local-join, bounded-checker, and obstruction rows by the carrier definition.

Use the NameCert obligations of Definition 6547.2. They force every confluence-facing reader to keep T, R, J, B, O visible before X can be consumed.

The structural rows then preserve the same route. Transport through H, replay through C, provenance through P, and local naming through N do not add an unconditional diamond theorem, a closed Church–Rosser theorem, a global substitution-compatibility proof, or an unbounded checker. □

Theorem 6547.4 (MetaCIC confluence diamond ledger candidate join exhaustion)**.** *Every accepted* $\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}$ *carrier* $G = (T, R, J, B, O, X, H, C, P, N)$ *that exposes a common-reduct claim in* X *must first expose the typed-window row* T*, residual-substitution row* R*, local-join row* J*, bounded-checker row* B*, and obstruction row* O*. Moreover, every common-reduct claim in* X *is backed by a row of* J *with the same typed window and the same residual-substitution boundary, while* O *remains present for any substitution-compatibility boundary not discharged by that row.*

Proof. Project the carrier of Definition 6547.1. The handoff coordinate X is not available before T, R, J, B, O have been displayed.

Apply the NameCert obligations of Definition 6547.2. Typed-window exposure fixes T, residual-substitution exposure fixes R, local-join accounting forces J to list the finite candidate joins, and bounded-checker accounting keeps those rows inside the finite checker budget B.

Apply Definition 6547.3. Any common-reduct read exported by X follows the route through T, R, J, B, O, so its local-join witness must be a row of J over the same typed window and residual-substitution boundary.

The obstruction row O is preserved rather than consumed silently. Thus the candidate-join ledger exhausts the displayed common-reduct claims without turning missing substitution compatibility into a global Church–Rosser proof. The existing Lean TasteGate evidence for this chapter records the Nontrivial and FieldFaithful obligations that keep these displayed fields visible to the classifier. □

Theorem 6547.5 (MetaCIC confluence diamond ledger obstruction lock)**.** *For every accepted* $\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}$ *carrier* $G = (T, R, J, B, O, X, H, C, P, N)$*, any confluence handoff row* X *must keep the typed-window row* T*, residual-substitution row* R*, local-join row* J*, bounded-checker row* B*, and obstruction row* O *visible until both the residual-substitution and local-join rows have been consumed by the same route. In particular,* X *alone cannot be read as a closed Church–Rosser theorem, global substitution-compatibility proof, unbounded checker, or host confluence oracle.*

Proof. Project the carrier of Definition 6547.1. The handoff coordinate X is downstream of T, R, J, B, O and has no independent coordinate for a global confluence proof.

Apply the NameCert obligations of Definition 6547.2. They require typed-window exposure, residual-substitution exposure, local-join accounting, bounded-checker accounting, obstruction retention, handoff non-escape, and structural replay before a confluence-facing read can be consumed.

Apply the handoff theorem Definition 6547.3. It keeps every handoff on the route

$$T \longmapsto R \longmapsto J \longmapsto B \longmapsto O \longmapsto X \longmapsto H, C, P, N.$$

Thus X cannot bypass the residual-substitution or local-join rows.

Finally apply the candidate-join theorem Definition 6547.4. Any common-reduct claim exported by X is backed by a row of J over the same typed window and residual-substitution boundary, while O remains present for the substitution-compatibility boundary not discharged by that row. The obstruction is therefore locked to the residual-substitution and local-join route rather than erased by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}$ packet exposing a confluence handoff must display typed-window, residual-substitution, local-join, bounded-checker, obstruction, transport, replay, provenance, and NameCert rows.

Independence witness: metaciccriticalpath, candidate-mediated-sn, metacic-confluence-critical-pair-boundary, metacic-residual-substitution-boundary, metacic-local-join-ledger: MetaCICConfluenceDiamondLedg is not bijective to any listed sibling because it jointly carries typed-window, residual-substitution, local-join, bounded-checker, obstruction-retention, handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}$ packet is generated from typed-window exposure, residual-substitution boundary data, local-join rows, bounded-checker rows, obstruction retention, a confluence handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6547.1, Definition 6547.2, Definition 6547.3, Definition 6547.4, and Definition 6547.5 seed the finite confluence-diamond ledger consumed by MetaCIC confluence routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional parallel diamond, arbitrary closed-term Church–Rosser, global substitution compatibility, unbounded checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-substitution exposure, local-join accounting, bounded-checker accounting, obstruction retention, handoff non-escape, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICConfluenceDiamond

Chapter 6548

A Concrete Naming Certificate for $\mathsf{LocatedCompactOvert}^{\uparrow}$

Following Bishop–Bridges constructive topology and formal-topology overt compactness, the $\mathsf{LocatedCompactOvert}^{\uparrow}$ packet records a compact object whose finite cover data and positive contact witnesses are both displayed. It sits beside the located compact boundary of chapter 1488, the finite compact witnesses of chapter 60 and chapter 129, the overt real interface of chapter 3354, the finite lower-bound fold of chapter 4096, and the Real-completion readback route through chapter 126, chapter 301, and chapter 38. The packet joins finite compact cover rows with overt witness rows without importing host powersets, open-cover compactness, selected points, quotient compacta, or choice.

Definition 6548.1 (Located compact overt carrier)**.** *A* $\mathsf{LocatedCompactOvert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, C, M, V, F, S, R, E, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{LocatedCompact}^{\uparrow}$ *source row, exposing located-distance tests, finite-net witnesses, separatedness support, and the local compact naming row.*
2. *C is the compact finite-cover row, read through* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *finite witness data rather than through a host open cover.*
3. *M is the metric compactness row that ties finite centers, radii, and coverage distances to the compact metric source.*
4. *V is the* $\mathsf{OvertReal}^{\uparrow}$ *positive-information row, carrying witnessable contact with the displayed located interval or cover cell.*
5. *F is the* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *row that folds finitely many positive radius or contact candidates into one certified lower bound.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which all completion-facing reads are scheduled.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational names attached to the compact and overt observations.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite windows, readback, lower-bound fold, compact rows, and overt witness row have been displayed.*
9. *H records componentwise* hsame *transport for the located, compact, overt, lower-bound, readback, seal, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay from located compact data through finite cover rows, positive witness rows, lower-bound folding, finite windows, regular readback, and the terminal Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactOvert}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{OvertReal}^{\uparrow}$, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactOvert}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, C, M, V, F, S, R, E, H, T, P, N$. It has no coordinate for host powersets, arbitrary open-set membership, selected finite subcovers, selected contact points, quotient compacta, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6548.2 (Located compact overt NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactOvert}^{\uparrow}$ *carrier*

$$O = (K, C, M, V, F, S, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactOvert}^{\uparrow}}$ *surface consists exactly of located compact source admission, compact finite-cover exposure, compact-metric finite-net compatibility, overt positive-witness exposure, finite lower-bound folding, StreamName window scheduling, RegSeqRat readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No consumer may replace these rows by host powersets, arbitrary open covers, selected points, quotient compactness, choice, or ambient completeness.*

Proof. Project the carrier of Definition 6548.1. The rows K, C, M expose the located compact source, finite cover data, and compact-metric finite-net compatibility before any overt or Real-facing consumer can read the packet.

Next read the positive side. The row V supplies witnessable contact, while F folds the finite positive candidates into one lower-bound row. This is a finite induction over the displayed candidate family, not a selection from a host powerset or an arbitrary open cover.

Finally replay the completion-facing boundary. Rows S, R, E schedule finite windows, read regular rational names, and seal the terminal Real row; H, T, P, N transport, replay, source, and name the same displayed packet. Since the carrier contains no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6548.3 (Located compact overt finite positive cover induction)**.** *Every positive compact-cover read of an accepted* $\mathsf{LocatedCompactOvert}^{\uparrow}$ *packet is obtained by induction over the displayed finite cover row C and the finite lower-bound fold F. At each cover cell the consumer first reads the located compact row K, the compact-metric row M, and the overt witness row V; only after the finite induction closes may it pass through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows S, R, E.*

Proof. Start with the finite cover row C. Its cells are displayed compact data, so the induction ranges over the finite cover witnesses already present in the carrier.

For each cell, read K, M, V. Locatedness supplies the finite comparison and net rows, compact metric data supply the finite center and radius information, and overtness supplies the witnessable positive contact row.

The fold row F combines the finitely many positive candidates. After the fold closes, the completion-facing route may read S, R, E and then transport the result through H, T, P, N. No step introduces host open-cover compactness, a selected point, a selected subcover, quotient compactness, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every $\mathsf{LocatedCompactOvert}^{\uparrow}$ consumer that claims positive compact contact while omitting the $\mathsf{LocatedCompact}^{\uparrow}$ source row, $\mathsf{Compact}^{\uparrow}/\mathsf{CompactMetric}^{\uparrow}$ finite cover rows, $\mathsf{OvertReal}^{\uparrow}$ witness row, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ fold row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: locatedcompact, compact, compactmetric, overt-real, finite-lower-bound-fold, streamname, regseqrat, real: $\mathsf{LocatedCompactOvert}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries located compact source data, compact finite covers, compact-metric finite nets, overt positive witnesses, finite lower-bound folding, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming as one constructive topology packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactOvert}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactOvert}^{\uparrow}$ packet is generated from LocatedCompactUp source rows, CompactUp and CompactMetricUp finite cover rows, OvertRealUp positive witness rows, FiniteLowerBoundFoldUp lower-bound folding, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactOvert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6548.1, Definition 6548.2, and Definition 6548.3 seed the finite located compact overt packet over compact finite cover data, overt witness rows, lower-bound folding, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactOvert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets, arbitrary open-set membership, open-cover compactness, selected finite subcovers, selected contact points, quotient compacta, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located compact source exposure, compact finite-cover exposure, overt witness exactness, finite lower-bound fold exactness, StreamName and RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactOvertUp.

Chapter 6549

A Concrete Naming Certificate for $\mathsf{CountablyCompactMetric}^{\uparrow}$

Following Kelley, Bourbaki, and Bishop–Bridges compactness practice, the $\mathsf{CountablyCompactMetric}^{\uparrow}$ packet records the metric specialization of countable-cover compactness as finite NameCert data. It reads the metric and compact-metric source surfaces of chapter 57 and chapter 129, the countable-cover row of chapter 6398, the finite stream windows of chapter 126, the sequential consumer surface of chapter 2372, the finite cluster-point route of chapter 1580, and the terminal real-name seal of chapter 38. The packet is a metric countable-cover certificate; it does not turn arbitrary open-cover compactness, countable choice, or an ambient Heine–Borel theorem into a primitive row.

Definition 6549.1 (Countably compact metric carrier)**.** *A* $\mathsf{CountablyCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, C, W, S, Q, H, R, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the named metric observations, compact-metric probes, and finite cover data used by the packet.*
2. *C is the countable cover enumeration row read through* $\mathsf{CountablyCompact}^{\uparrow}$*. It records the displayed countable family of metric-open rows and the source-surface coverage checks that the packet is allowed to inspect.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite cover-window row. It lists the inspected prefixes of the countable cover, the finite metric probes attached to those prefixes, and the pointwise cover checks visible in each prefix.*
4. *S is the finite subcover ledger. It records the accepted finite prefix, its compact-metric witness rows, and the statement that later enumeration rows are not needed for the displayed source read.*
5. *Q is the sequential consumer window. It is read either through* $\mathsf{SequentialCompact}^{\uparrow}$ *or through the* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *route when the downstream consumer asks for a sequence or cluster-point comparison over the same metric source.*
6. *H stores componentwise* hsame *transport for the metric source, countable-cover enumeration, stream windows, finite-subcover ledger, sequential consumer window, replay, provenance, and local naming rows.*

7. *R stores displayed* Cont *replay from a countable-cover metric request through M, C, W, S, Q, H.*
8. *P is the* Pkg *provenance over* $\mathsf{CountablyCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CountablyCompact}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{RealSequenceClusterPoint}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CountablyCompactMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected subcovers by countable choice, quotient equality of metric opens, host sequential compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6549.2 (Countably compact metric NameCert obligations)**.** *For every accepted* CountablyCompactMetric *carrier $K = (M, C, W, S, Q, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CountablyCompactMetric}^{\uparrow}}$ *surface consists exactly of metric or compact-metric source admission, countable cover enumeration through* $\mathsf{CountablyCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$ *finite cover windows, finite subcover ledger readback, sequential or cluster-point consumer windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every metric compactness consumer must read M, C, W, S, Q before the structural rows H, R, P, N can be consumed.*

Proof. Project the carrier of Definition 6549.1. The only source information is the metric or compact-metric row M, and the only countable-cover information is the row C read through $\mathsf{CountablyCompact}^{\uparrow}$.

The finite evidence is downstream of that countable cover. Row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ prefixes, and S records the finite prefix whose metric-open rows cover the displayed source surface.

The sequential comparison is not a second compactness principle. Row Q is a consumer window for $\mathsf{SequentialCompact}^{\uparrow}$ or $\mathsf{RealSequenceClusterPoint}^{\uparrow}$, so sequence and cluster-point reads must pass through the already displayed finite cover window and subcover ledger.

Finally H, R, P, N transport, replay, source, and name exactly those rows. Since no field stores arbitrary open covers, selected subcovers by countable choice, quotient open equality, host sequential compactness, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6549.3 (Countably compact metric sequential window handoff)**.** *Every accepted* CountablyCompactMetri *packet that is consumed by a sequential or real-cluster-point route factors through the finite chain*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{CountablyCompact}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{SequentialCompact}^{\uparrow},\ \mathsf{RealSequenceClusterPo}$$

The handoff exports only the displayed metric source row, countable-cover enumeration, finite cover windows, finite subcover ledger, sequential consumer window, transports, replay, provenance, and local naming rows.

Proof. Start with an accepted carrier $K = (M, C, W, S, Q, H, R, P, N)$. A sequential or cluster-point consumer first reads the metric source M and the countable cover row C, so the consumer sees a displayed countable metric cover rather than an arbitrary host cover.

The finite cover evidence is then fixed by W and S. The $\mathsf{StreamName}^{\uparrow}$ row supplies the inspected prefixes, and the finite subcover ledger supplies the accepted prefix for the same metric source.

Only after those rows are visible may Q be consumed. Thus the $\mathsf{SequentialCompact}^{\uparrow}$ or $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ comparison window is a downstream read of the finite cover route, and the terminal $\mathsf{Real}^{\uparrow}$ seal is reached only through that displayed chain.

The structural rows H, R, P, N preserve and name the same packet. They do not add countable choice, open-cover compactness, quotient open equality, host subsequence selection, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountablyCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CountablyCompactMetric}^{\uparrow}$ packet is generated from a metric or compact-metric source row, a CountablyCompact countable-cover enumeration, StreamName finite cover windows, a finite subcover ledger, a sequential or cluster-point consumer window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountablyCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6549.1, Definition 6549.2, and Definition 6549.3 seed the metric countable-cover carrier over Metric, CompactMetric, CountablyCompact, SequentialCompact, RealSequenceClusterPoint, StreamName, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountablyCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, countable choice, host sequential compactness, a selected cluster point, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Formalization requires carrier admission, countable-cover enumeration, StreamName finite-window exactness, finite-subcover ledger readback, sequential or cluster-point consumer handoff, transport, replay, provenance, and local naming for BEDC.Derived.CountablyCompactMetricUp.

Chapter 6550

A Concrete Naming Certificate for $\mathsf{DirichletUniformTailSelector}^{\uparrow}$

Following Dirichlet and the Hardy/Bishop–Bridges treatment of series tests, the $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ packet records the finite selector route by which a displayed Dirichlet tail estimate is handed to a uniform Cauchy series consumer. It sits after the real-series source of chapter 1581, the bounded-partial-sum Dirichlet-test face of chapter 1497, the shared-tail handoff of chapter 1736, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite tail-budget selector for Real-and-Completion consumers; it does not assert host infinite sums, quotient series equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6550.1 (Dirichlet uniform tail selector carrier)**.** *A* $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing only the displayed term stream and finite partial-sum windows consumed by the selector.*
2. *D is the* $\mathsf{DirichletTest}^{\uparrow}$ *face row, exposing bounded partial sums, coefficient windows, null-tail data, and the finite Dirichlet tail-estimate schedule.*
3. *T is the selector row. For each requested precision and displayed source window it names the finite tail threshold admitted by the Dirichlet-test face.*
4. *W is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff row, recording that the selector threshold is read as the shared uniform tail threshold for the displayed partial-sum comparisons.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected tail window and the admitted partial-sum comparisons.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, T, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, Dirichlet face, selector, uniform-tail handoff, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$S, D \longmapsto T \longmapsto W \longmapsto R \longmapsto E, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{DirichletUniformTailSelector}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, W, R, E, H, C, P, N$. It has no coordinate for a host infinite sum, quotient equality of partial-sum streams, a selected limiting real, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6550.2 (Dirichlet uniform tail selector NameCert obligations)**.** *For every accepted* $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ *carrier $U = (S, D, T, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformTailSelector}^{\uparrow}}$ *surface consists exactly of real-series source admission through S, Dirichlet-test face exposure through D, finite tail-threshold selection through T, uniform Cauchy handoff through W, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

No selector obligation is stored outside those rows.

Proof. Project the accepted carrier of Definition 6550.1. The source side is exhausted by S and D: S exposes the real-series term and partial-sum windows, while D exposes the bounded partial sums, coefficient windows, null-tail ledger, and finite Dirichlet estimate schedule.

Read the selector row next. The row T names the finite threshold used for the requested precision, and W records the same threshold as the shared uniform Cauchy tail row for the displayed partial-sum comparisons.

The endpoint route is downstream. The row R reads the selected tail window through $\mathsf{RegSeqRat}^{\uparrow}$, and E seals only that readback through $\mathsf{Real}^{\uparrow}$.

Finally the rows H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 6550.3 (Dirichlet uniform tail handoff)**.** *Every* $\mathsf{UniformCauchySeries}^{\uparrow}$*-facing read of an accepted* $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ *carrier factors through the finite route*

$$S, D \longmapsto T \longmapsto W \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the uniform tail budget exported to Real-and-Completion consumers is a displayed Dirichlet selector threshold read through regular rational data and a terminal real seal, not a host infinite sum, quotient series equality, selected limit, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a uniform-tail request. The carrier first exposes S, so the request sees only the displayed $\mathsf{RealSeries}^{\uparrow}$ term stream and partial-sum windows.

The request then reads the Dirichlet face D. This row supplies the bounded partial sums, coefficient windows, null-tail data, and estimate schedule needed to justify a finite tail threshold.

The selector threshold is the row T. The handoff row W is downstream of T, so a $\mathsf{UniformCauchySeries}^{\uparrow}$ consumer receives the threshold only after the Dirichlet-test data have been displayed.

Finally R and E route the selected tail window through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N preserve this same path, so no excluded host-side summability, quotient, limit, choice, order, or completeness coordinate can be exported. □

Falsifiable prediction: A Real-and-Completion consumer that reads $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ without first exposing the $\mathsf{RealSeries}^{\uparrow}$ source row S, the $\mathsf{DirichletTest}^{\uparrow}$ face row D, the selector row

T, the $\mathsf{UniformCauchySeries}^{\uparrow}$ handoff row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, or the $\mathsf{Real}^{\uparrow}$ seal row E refutes the carrier of Definition 6550.1.

Independence witness: realseries, dirichlet-test, uniform-cauchy-series, regseqrat, real: no listed sibling supplies the whole selector packet $U = (S, D, T, W, R, E, H, C, P, N)$, because each sibling contributes only a source, a test face, a uniform-tail handoff, a regular readback, or a terminal seal while this chapter binds them as one finite selector certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletUniformTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletUniformTailSelector}^{\uparrow}$ packet is generated from RealSeries source rows, DirichletTest face rows, a finite selector threshold, a UniformCauchySeries handoff, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletUniformTailSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6550.1, Definition 6550.2, and Definition 6550.3 seed the finite Dirichlet uniform tail-selector route over $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletUniformTailSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient series equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, Dirichlet face exposure, finite selector thresholding, uniform Cauchy handoff, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DirichletUniformTailSelectorUp.

Chapter 6551

A Concrete Naming Certificate for $\mathsf{FiniteMetricTree}^{\uparrow}$

Following Buneman and Dress, finite tree metrics are read from a finite weighted tree by the path-distance ledger on its vertices. The $\mathsf{FiniteMetricTree}^{\uparrow}$ packet records that classical object as constructive L10 finite-metric witness data: it sits over the metric surface of chapter 57, the graph and tree routes of chapter 118 and chapter 119, and the finite-net consumer surface of chapter 1767. The packet is a finite path-distance certificate, not a quotient metric space, not a selected geodesic in an ambient graph, and not a host compactness theorem.

Definition 6551.1 (FiniteMetricTree carrier)**.** *A* $\mathsf{FiniteMetricTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, W, P, M, G, R, H, C, K, N)$$

with the following displayed rows:

1. *V is the finite vertex ledger, listing the accepted vertex names and the finite equality classifier used by later distance reads.*
2. *E is the finite edge ledger, listing accepted undirected edge incidences between vertices of V.*
3. *W is the edge-length row, assigning a positive* $\mathsf{Real}^{\uparrow}$*-sealed or rationally witnessed length to every displayed edge.*
4. *P is the path-distance readback row. For each accepted pair of vertices it records the unique finite tree path through E and the finite sum of edge lengths from W.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *row whose distance witness is exactly the path-distance readback P and whose metric laws are checked against the finite path ledger.*
6. *G is the sibling graph-and-tree route, exposing the* $\mathsf{Graph}^{\uparrow}$ *adjacency row and the* $\mathsf{Tree}^{\uparrow}$ *connected-acyclic row before any path-distance readback is consumed.*
7. *R is the finite four-point comparison ledger. For each displayed quadruple it records the two largest of the three pair sums and the equality comparison required by the tree-metric condition.*
8. *H stores componentwise* hsame *transport for vertices, edges, edge lengths, path distances, metric rows, graph/tree rows, four-point ledgers, continuation rows, provenance, and the local naming row.*
9. *C stores* Cont *replay from a vertex-pair or quadruple request through the finite graph/tree route, path-distance readback, metric row, and four-point ledger.*

10. *K is the* Pkg *provenance over* $\mathsf{FiniteMetricTree}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, *finite-net consumers,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricTree}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, E, W, P, M, G, R, H, C, K, N$. It has no coordinate for quotient metric spaces, choice-selected paths outside the finite tree, open-cover compactness, infinite graph completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6551.2 (FiniteMetricTree NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricTree}^{\uparrow}$ *carrier $T = (V, E, W, P, M, G, R, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricTree}^{\uparrow}}$ *surface consists exactly of carrier admission, finite vertex-ledger admission, edge-ledger admission, positive edge-length readback, graph/tree route exposure, path-distance exactness, metric-law stability, four-point comparison stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger nonescape.*

In particular, any $\mathsf{Met}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, *or finite-net consumer must read the displayed finite ledger route*

$$V \longmapsto E \longmapsto W \longmapsto P \longmapsto M \longmapsto G \longmapsto R \longmapsto H, C, K, N$$

before using the packet as finite tree-metric data.

Proof. Project the carrier of Definition 6551.1. The projection exposes exactly the finite vertex ledger, edge ledger, edge-length row, path-distance readback, metric row, graph/tree route, four-point ledger, and structural rows.

The graph/tree route is read before distance data. The $\mathsf{Graph}^{\uparrow}$ row names the finite adjacency surface, while the $\mathsf{Tree}^{\uparrow}$ row supplies connectedness and acyclicity as displayed finite data; therefore the path row P can read a unique finite route without selecting a path from an ambient graph.

Metric-law stability follows from the same path-distance ledger. Symmetry reverses the finite path, reflexivity reads the empty path, and the triangle comparison concatenates displayed finite paths before returning to the tree path through the metric row M.

The four-point comparison row R is downstream of P and M. It records quadruple comparisons only from displayed path distances, so componentwise transport through H and replay through C preserve the comparison surface.

Finally, K and N source and name the finite packet. Since no carrier row stores quotient metric spaces, choice-selected ambient paths, open-cover compactness, infinite graph completion, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6551.3 (FiniteMetricTree four-point stability)**.** *Let $T = (V, E, W, P, M, G, R, H, C, K, N)$ and $T' = (V', E', W', P', M', G', R', H', C', K', N')$ be accepted* $\mathsf{FiniteMetricTree}^{\uparrow}$ *packets. If the vertex ledger, edge ledger, edge-length row, path-distance readback, metric row, graph/tree route, four-point ledger, replay row, provenance row, and local naming row of T are transported componentwise through the displayed* hsame *and* Cont *routes to the corresponding rows of T', then every finite four-point tree-metric comparison accepted by R is accepted by R' with the same displayed pair-sum equality.*

Proof. Begin with the obligation surface of Definition 6551.2. The four-point comparison row is not independent data; it is downstream of the finite vertex, edge, edge-length, path-distance, metric, and graph/tree rows.

Transport the carrier componentwise. The displayed hsame rows carry vertices to vertices, edges to edges, edge lengths to edge lengths, and path-distance readbacks to path-distance readbacks, while the Cont rows replay the same quadruple request through the transported packet.

The four-point ledger therefore recomputes its three pair sums from transported path distances. Since the accepted comparison in R was read only from those sums, R' accepts the same tree-metric equality after transport.

No external geodesic selector, quotient metric equality, compactness principle, or ambient completion can alter the comparison, because none is a coordinate of either accepted packet. □

Falsifiable prediction: A consumer that claims $\mathsf{FiniteMetricTree}^{\uparrow}$ four-point stability while omitting the finite vertex ledger, edge ledger, edge-length row, path-distance readback, MetricUp row, GraphUp/TreeUp route, four-point ledger, hsame transport, Cont replay, Pkg provenance, or NameCert row refutes the carrier of Definition 6551.1.

Independence witness: metric, graph, tree, finite-epsilon-net: $\mathsf{FiniteMetricTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite weighted tree ledger, exact path-distance readback, MetricUp row, GraphUp/TreeUp route, four-point comparison ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricTree}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetricTree}^{\uparrow}$ packet is generated from finite vertex rows, finite edge rows, positive edge-length rows, path-distance readback, MetricUp rows, GraphUp/TreeUp route rows, four-point comparison rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6551.1, Definition 6551.2, and Definition 6551.3 seed the finite tree-metric NameCert interface and its componentwise four-point stability route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetricTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient metric spaces, selected ambient geodesics, open-cover compactness, infinite graph completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-law stability, path-distance exactness, four-point comparison stability, graph/tree route exposure, and ledger nonescape for BEDC.Derived.FiniteMetricTreeUp.

Chapter 6552

A Concrete Naming Certificate for MetaCICCandidateDiamondBudget$^{\uparrow}$

The MetaCICCandidateDiamondBudget$^{\uparrow}$ packet realizes the normalization/confluence audit boundary of chapter 9383 as finite NameCert data. It sits between the candidate-mediated normalization route of chapter 3779, the candidate-set normalization packet of chapter 2320, the confluence diamond ledger of chapter 6547, and the closed parallel diamond premise ledger of chapter 2325. The packet names a local diamond budget guarded by displayed candidate evidence; it is not global closed confluence, not evidence-free normalization, and not a closed Church–Rosser theorem.

Falsifiable prediction: A packet without a displayed candidate row and local diamond-budget row is rejected even if it carries a closed-normal row.

Independence witness: metacicnormalizationaudit, metacic-confluence-audit-packet, metacic-candidate-set-normalization, closed-parallel-diamond-premise-ledger, metacic-confluence-diamond-ledger: MetaCICCandidateDiamondBudget$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate evidence, closed-candidate evidence, candidate-set normalization, local peak data, local join-budget data, confluence-boundary refusal, transport, replay, provenance, and local naming.

Definition 6552.1 (MetaCIC candidate diamond budget carrier)**.** *A* MetaCICCandidateDiamondBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, K, L, S, P, J, D, F, H, C, G, N)$$

with the following displayed rows:

1. *T is the closed typed term window whose normalization or confluence-facing read is being inspected.*
2. *K is the candidate-evidence row, carrying the displayed* Candidate *witness for T.*
3. *L is the closed-candidate evidence row, binding closedness of the term and type to the same candidate-mediated route.*
4. *S is the candidate-set row, exposing candidate-set membership and the finite candidate-set normalization surface.*
5. *P is the local peak row, recording the finite redex pair or bounded reduction peak inspected inside the candidate-mediated window.*

6. *J is the local join-budget row, recording the finite candidate join, bounded checker budget, and common-reduct ledger for P.*
7. *D is the confluence-boundary refusal row. It records that the local budget J is not a global closed-confluence theorem and cannot discharge the residual-substitution or closed parallel diamond premises by itself.*
8. *F is the closed-normal row, consumed only after K, L, S, P, J, D have all been displayed.*
9. *H stores componentwise* hsame *transport for the closed typed term, candidate, closed-candidate, candidate-set, local peak, join-budget, refusal, closed-normal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a normalization-window or confluence-boundary request through T, K, L, S, P, J, D, F.*
11. *G is the* Pkg *provenance over* $\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}$*, chapter 9383, chapter 9378,* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$, $\mathsf{CandidateSetNormalization}^{\uparrow}$, $\mathsf{MetaCICConfluenceDiamondLedger}^{\uparrow}$, $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, K, L, S, P, J, D, F, H, C, G, N$. It has no coordinate for evidence-free strong normalization, global closed confluence, a discharged residual-substitution theorem, arbitrary closed parallel diamond, quotient syntax equality, or host confluence oracle.

Theorem 6552.2 (MetaCIC candidate diamond budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}$ *carrier $B = (T, K, L, S, P, J, D, F, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, closed typed term admission, candidate evidence admission, closed-candidate evidence admission, candidate-set normalization exposure, local peak exposure, local join-budget accounting, confluence-boundary refusal, closed-normal downstream ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The future Lean TasteGate for this packet must supply both a Nontrivial instance, separating an accepted displayed budget from an empty carrier, and a FieldFaithful instance, asserting that the classifier reads each field $T, K, L, S, P, J, D, F, H, C, G, N$.

Proof. Project the carrier of Definition 6552.1. The projection exposes the closed typed term window, candidate row, closed-candidate row, candidate-set row, local peak row, join-budget row, refusal row, closed-normal row, and the structural rows.

The candidate rows are read before any normalization or confluence-facing budget. The candidate row K, closed-candidate row L, and candidate-set row S keep the strong-normalization route inside the candidate-mediated interface described by chapter 9383.

The local peak and join-budget rows then bind the confluence-facing part. The row P records only the inspected finite peak, while J records only the bounded local join and checker budget attached to that peak.

The refusal row D remains visible before the closed-normal row F. It prevents the local join budget from being read as global closed confluence, residual-substitution discharge, or an arbitrary closed parallel diamond.

Finally, H, C, G, N transport, replay, source, and name the same finite route. Nontrivial and FieldFaithful TasteGate evidence would assert that accepted packets carry genuine budget data and that no displayed field is ignored by the classifier. □

Theorem 6552.3 (MetaCIC candidate diamond budget nonescape)**.** *Every normalization-window, confluence-boundary, or closed-normal consumer of an accepted* $\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}$ *carrier must first expose the finite route*

$$T \longmapsto K \longmapsto L \longmapsto S \longmapsto P \longmapsto J \longmapsto D \longmapsto F \longmapsto H, C, G, N.$$

The packet exports candidate-mediated normalization evidence and a local diamond budget only through that route. It exports no evidence-free strong-normalization theorem, no global closed-confluence theorem, no residual-substitution discharge, no arbitrary closed parallel diamond, no quotient syntax equality, and no host confluence oracle.

Proof. Start from the carrier definition. Its only confluence-facing rows are the local peak row P, the local join-budget row J, and the refusal row D.

Apply Definition 6552.2. The obligations order candidate evidence before the local budget and keep the closed-normal row downstream of the confluence-boundary refusal.

The structural rows preserve that order. Transport through H, replay through C, provenance through G, and local naming through N cannot add a global confluence endpoint or erase the candidate evidence.

Thus any consumer that reads the budget must keep $T, K, L, S, P, J, D, F, H, C, G, N$ visible and cannot escape into evidence-free normalization, residual-substitution discharge, arbitrary closed diamond, quotient syntax equality, or host confluence. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}$ packet is generated from a closed typed term window, candidate evidence, closed-candidate evidence, candidate-set rows, local peak rows, local join-budget rows, confluence-boundary refusal, closed-normal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6552.1, Definition 6552.2, and Definition 6552.3 seed the finite candidate-mediated local diamond-budget boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateDiamondBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization, global closed confluence, residual-substitution discharge, arbitrary closed parallel diamond, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence admission, closed-candidate admission, candidate-set exposure, local peak exposure, local join-budget accounting, confluence-boundary refusal, nonescape, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICCandidateDiamondBudgetUp.

Chapter 6553

A Concrete Naming Certificate for $\mathsf{UpperHemicontinuity}^{\uparrow}$

Following Berge's maximum theorem and the set-valued analysis of Aubin–Frankowska, the $\mathsf{UpperHemicontinuity}^{\uparrow}$ packet records the finite BEDC boundary for compact-valued correspondence data. It sits after the compact metric source of chapter 129, the real-name seal of chapter 38, the polytope and convex-set rows of chapter 209 and chapter 208, and the finite rollout consumer of chapter 348. The packet names a finite graph-window and compact-value certificate surface; it is not a host set-valued function primitive, an open-cover compactness theorem, or a selected optimizer.

Definition 6553.1 (Upper hemicontinuity carrier)**.** *An* $\mathsf{UpperHemicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, G, K, R, V, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *domain row whose finite probes supply the source windows.*
2. *Y is the* $\mathsf{Real}^{\uparrow}$ *or finite-dimensional real codomain seal through which value coordinates are read.*
3. *G is a finite graph-window ledger recording inspected source probes, value probes, and membership rows for the correspondence graph.*
4. *K is the compact-value row naming the finite cover, located radius, and total-bounded witness for each displayed value fibre.*
5. *R is the closed-graph readback row: whenever a source window and value window are replayed through G, the accepted value remains inside the displayed compact-value fibre.*
6. *V is the polytope or convex-set dependency row used by optimization and control consumers when value fibres are exposed as* $\mathsf{Polytope}^{\uparrow}$ *or* $\mathsf{ConvexSet}^{\uparrow}$ *packets.*
7. *O records the downstream optimization window consumed by* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *and related finite-horizon packets.*
8. *H records componentwise* hsame *transport for the domain, codomain, graph, compact-value, closed-graph, convex/polytope, optimization, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite source probe through graph membership, compact-value readback, and the consumer-facing optimization window.*

10. *P is the* Pkg *provenance over* $\mathsf{UpperHemicontinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ModelPredictiveControl}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperHemicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, G, K, R, V, O, H, C, P, N$. It has no coordinate for a host correspondence object, pointwise choice of values, open-cover compactness, ambient graph closure, quotient equality of real names, or selected optimizer.

Lean4 ✓ `BEDC.Derived.UpperHemicontinuityUp.UpperHemicontinuityTasteGate single carrier alignment`

Theorem 6553.2 (Upper hemicontinuity NameCert obligations)**.** *For every accepted* $\mathsf{UpperHemicontinuity}^{\uparrow}$ *packet $U = (X, Y, G, K, R, V, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UpperHemicontinuity}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-metric source exposure through X, real codomain sealing through Y, graph-window inspection through G, compact-value certification through K, closed-graph readback through R, convex/polytope dependency exposure through V, finite optimization-window handoff through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 6553.1. The projected rows are exactly the domain, codomain, graph-window, compact-value, closed-graph, convex/polytope, optimization, transport, replay, provenance, and local naming rows.

The analytic part is finite. A source probe reads X, a value probe reads Y, graph membership is inspected through G, and compactness of the fibre is read only through the displayed row K.

The upper-hemicontinuity boundary is the closed-graph readback row R together with the same compact-value row K. The convex/polytope dependency V and optimization window O are downstream consumers of those rows, not independent value selectors.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier contains no coordinate for a host correspondence, selected value, open-cover proof, ambient closure theorem, quotient real equality, or optimizer, the listed rows exhaust the seed NameCert surface. □

Theorem 6553.3 (Compact-valued correspondence handoff)**.** *Every compact-valued optimization or control consumer of an accepted* $\mathsf{UpperHemicontinuity}^{\uparrow}$ *packet must first read the finite route*

$$X \longmapsto Y \longmapsto G \longmapsto K \longmapsto R \longmapsto V \longmapsto O \longmapsto H, C, P, N.$$

In particular, a $\mathsf{ModelPredictiveControl}^{\uparrow}$ *consumer receives the compact metric source, real codomain seal, graph-window ledger, compact-value witness, closed-graph readback, convex or polytope dependency rows, transport, replay, provenance, and local naming before it can use a compact-valued correspondence packet.*

Proof. Start with an accepted carrier $U = (X, Y, G, K, R, V, O, H, C, P, N)$. The carrier places every consumer-facing row after the graph-window and compact-value rows, so no consumer can inspect V or O before reading X, Y, G, K, R.

The compact-valued part is supplied by K, while the upper-hemicontinuity readback is supplied by R. Thus the consumer sees compact values and closed-graph stability as finite certificate rows rather than as host predicates on an external set-valued map.

The structural rows keep the same route under componentwise sameness and replay. Since $\mathsf{ModelPredictiveControl}^{\uparrow}$ reads only the displayed optimization window O together with its provenance and local naming rows, it receives no selected optimizer, no ambient correspondence, and no open-cover compactness principle from the packet. □

Falsifiable prediction: Every $\mathsf{UpperHemicontinuity}^{\uparrow}$ consumer read factors through a CompactMetricUp source row, RealUp codomain seal, finite graph-window ledger, compact-value witness, closed-graph readback, convex or polytope dependency row, optimization window, transport, replay, provenance, and local naming row.

Independence witness: compactmetric, real, polytope, convexset, and modelpredictivecontrol: $\mathsf{UpperHemicontinuity}^{\uparrow}$ is not bijective to any listed sibling because it jointly records compact-valued graph windows, closed-graph readback, convex/polytope value dependencies, optimization handoff, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UpperHemicontinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{UpperHemicontinuity}^{\uparrow}$ packet is generated from a compact metric source row, a RealUp codomain seal, a finite graph-window ledger, compact-value witnesses, closed-graph readback, convex or polytope dependency rows, an optimization consumer window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperHemicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6553.1, Definition 6553.2, and Definition 6553.3 seed the finite NameCert surface for compact-valued upper-hemicontinuity packets over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, and $\mathsf{ModelPredictiveControl}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpperHemicontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host set-valued function primitive, open-cover compactness, selected optimizers, ambient graph closure, quotient real equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-valued graph-window readback, convex/polytope dependency transport, optimization handoff, replay stability, and local NameCert provenance for BEDC.Derived.UpperHemicontin

Chapter 6554

A Concrete Naming Certificate for $\mathsf{UrysohnUniversalMetric}^{\uparrow}$

Following Urysohn's construction of a universal separable metric space and the Bishop–Bridges discipline for constructive metric data, the $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ packet records finite one-point extension requests as BEDC NameCert rows. It sits over the metric carrier of chapter 57, the separable-metric source of chapter 1458, the complete-metric boundary of chapter 128, the completion handoff of chapter 1506, and the terminal real-distance seal of chapter 38. The packet exposes the finite Katetov-extension surface needed by L10 metric and completion consumers without selecting a host Urysohn space, quotienting completion streams, invoking choice, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6554.1 (Urysohn universal metric finite-extension carrier)**.** *A* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, S, K, L, R, F, A, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including point classifiers and all inspected distance requests.*
2. *S is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-name row used to enumerate the finite support of each extension request.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row read over the same metric source.*
4. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, available only after the finite extension request has been displayed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *distance and tolerance seal row for every finite support distance, requested new distance, and compatibility inequality.*
6. *F is the finite Katetov-extension request row. It names a finite support family, a proposed distance from the new point to each support point, and the triangle and reverse-triangle budgets checked through M and R.*
7. *A is the finite admission ledger. It records that every requested distance cell in F is compatible with the support distances already displayed by M and the dense-name row S.*
8. *E is the extension-realization handoff row. It allows a consumer to read the requested one-point extension only after M, S, K, L, R, F, A have been exposed.*

9. *H records componentwise* hsame *transport for the metric source, separable row, complete boundary, completion handoff, real seals, finite extension request, admission ledger, realization handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from any finite Katetov request through the same source, dense-name, real-distance, admission, completion, and realization rows.*

11. *P is the* Pkg *provenance over* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *finite Katetov requests,* BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UrysohnUniversalMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, K, L, R, F, A, E, H, C, P, N$. It has no coordinate for a chosen host universal metric space, quotient completion equality, selected dense enumeration outside S, arbitrary countable extension sequence, choice functional, Lean verification, standard bridge, or ambient real-completeness principle.

Theorem 6554.2 (Urysohn universal metric NameCert obligations)**.** *For every accepted* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ *carrier*

$$U = (M, S, K, L, R, F, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UrysohnUniversalMetric}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the metric source, separable dense row, complete-metric boundary, completion handoff, real-distance seals, finite Katetov request, admission ledger, realization handoff, transport, replay, provenance, and local name;*

2. *common-source discipline, requiring S, K, L, R, F, A to be read over the same* $\mathsf{Met}^{\uparrow}$ *row M;*

3. *finite support visibility, requiring every extension request in F to list its support points through S before a new distance row may be inspected;*

4. *Katetov compatibility, requiring the triangle and reverse-triangle budgets in F and A to be witnessed by displayed* $\mathsf{Real}^{\uparrow}$ *comparisons over the already exposed support distances;*

5. *completion handoff order, requiring E to be consumed before L is treated as a completion-facing realization row;*

6. *nonescape, excluding host universal spaces, quotient completion equality, countable choice, selected infinite extension chains, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge credentials from the packet.*

No Urysohn universal metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 6554.1. The displayed tuple $M, S, K, L, R, F, A, E, H, C, P, N$ is the whole public surface, so carrier admission is exactly admission of those rows.

The common-source obligation follows from the order of the rows. The separable dense row, complete-metric boundary, completion handoff, real-distance seals, request row, and admission ledger are all transported by the same H row over the metric source M.

Finite support visibility is forced by F. A one-point extension request names only finitely many support points, and those support names are opened through S before any proposed new distance can be read.

Katetov compatibility is checked by R and A. The real-distance row supplies the support distances and proposed distances, while the admission ledger records the triangle and reverse-triangle budgets for each displayed support pair.

Finally, E, L, H, C, P, N preserve only that displayed route. Since none of the carrier coordinates stores a host universal space, quotient completion equality, selected infinite chain, choice functional, ambient completeness theorem, Lean verification, or bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 6554.3 (Urysohn universal metric finite-extension route)**.** *Every finite one-point extension request accepted by a* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{SeparableMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto F, A \longmapsto E.$$

In particular, the requested extension is read as a displayed finite Katetov route over the current metric source, not as a point selected from a host universal space or as an equality class in a completed quotient.

Proof. Begin with the metric source row M. It supplies the support-distance comparisons that any extension request must respect.

Read the separable row S next. The request row F can name only the finitely many support points exposed by the dense-name surface, so no hidden enumeration or chosen ambient point is available.

The distance budgets are then read through R and A. These rows display the proposed distances and the triangle and reverse-triangle checks against the already exposed support distances.

The realization handoff E is downstream of the same rows. It therefore exports the finite Katetov extension route itself, while host universal spaces, quotient representatives, selected points, choice functionals, and ambient completeness remain outside the packet. □

Theorem 6554.4 (Urysohn universal metric completion handoff)**.** *Let* $U = (M, S, K, L, R, F, A, E, H, C, P, N)$ *be an accepted* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ *carrier. Any completion-facing consumer of* U *must first read the finite-extension route of Definition 6554.3, then the complete-metric boundary* K*, and only then the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row* L*. Thus the packet supplies completion consumers with displayed finite extension data and a named completion boundary, not a quotient completion construction or a selected limit point.*

Proof. Apply the finite-extension route. The request reaches E only after M, S, R, F, A have displayed the metric source, finite support, real-distance budgets, and admission ledger.

Next inspect the complete-metric boundary K. It is attached to the same source row M by the transport row H, so it cannot introduce a second metric source or a host completion object.

Only after those rows are exposed may the consumer read L. The $\mathsf{MetricCompletion}^{\uparrow}$ handoff is therefore a downstream row of the finite-extension packet and complete-metric boundary, not a quotient construction, selected completion representative, countable-choice sequence, or ambient completeness theorem. □

Theorem 6554.5 (Urysohn universal metric nonescape boundary)**.** *For every accepted* $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ *carrier, the finite-extension and completion-handoff rows expose no host universal metric space, no selected dense enumeration beyond the displayed* $\mathsf{SeparableMetric}^{\uparrow}$ *row, no quotient completion equality, no arbitrary countable extension chain, no choice functional, no Lean verification credential, no standard bridge, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the carrier boundary in Definition 6554.1. The only public coordinates are the metric source, separable row, complete boundary, completion handoff, real-distance seals, finite request, admission ledger, realization handoff, transport, replay, provenance, and local name.

The finite-extension theorem and completion-handoff theorem read those rows in order. They never introduce an additional coordinate from which a host universal space, quotient equality, selected infinite chain, choice functional, Lean credential, bridge credential, or ambient completeness principle could be recovered.

Therefore every public consumer remains inside the displayed NameCert packet. Any claim requiring one of the excluded objects is outside the scope of $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ as recorded here. □

Falsifiable prediction: Every accepted $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ finite-extension read exposes the $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite Katetov request, admission ledger, and realization handoff rows before any $\mathsf{MetricCompletion}^{\uparrow}$ consumer may inspect the completion boundary.

Independence witness: metric, separablemetric, completemetric, metriccompletion, real: UrysohnUniversalMetric is not bijective to any listed sibling because it jointly binds finite Katetov-extension requests, displayed real-distance compatibility budgets, separable support rows, complete-metric boundaries, completion handoffs, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UrysohnUniversalMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ packet is generated from a displayed MetricSpace row, SeparableMetric dense row, CompleteMetric boundary, MetricCompletion handoff, Real distance seals, finite Katetov-extension request, admission ledger, realization handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UrysohnUniversalMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6554.1, Definition 6554.2, Definition 6554.3, Definition 6554.4, and Definition 6554.5 seed the finite-extension completion-neighbour route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UrysohnUniversalMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Urysohn-space existence, quotient completion equality, countable choice, selected infinite extension chains, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-source discipline, finite support visibility, Katetov compatibility, completion handoff order, and nonescape for BEDC.Derived.UrysohnUniversalMetricUp.

Chapter 6555

A Concrete Naming Certificate for $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$

Bishop-style constructive analysis and the Arzela–Ascoli finite-family tradition motivate the PointwiseBoundedRealF packet as a finite naming certificate for inspected real-family bounds. The packet sits between the stream naming route of chapter 126, the regular rational handoff of chapter 301, the real seal of chapter 38, and the displayed finite upper-bound surface of chapter 1427. It records boundedness over a named finite index window, not infinite Arzela–Ascoli compactness, open-cover compactness, choice-selected subsequences, a host supremum, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6555.1 (Pointwise bounded Real-family carrier)**.** *A* $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, S, R, E, B, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the index-window ledger. It lists the finite family positions currently inspected and contains no ambient set of all indices.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation ledger. For each displayed index in I, it records the finite stream windows read from the corresponding component.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attached to the observed stream windows, limited to the precisions named in I and S.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the components whose stream and regular rational rows have already been displayed.*
5. *B is the per-index dyadic bound witness row. Each entry pairs one inspected index with a dyadic upper-bound witness for the corresponding displayed* $\mathsf{Real}^{\uparrow}$ *seal.*
6. *U is the finite supremum ledger. It displays the upper-bound rows obtained from the finite list of dyadic witnesses in B, using* $\mathsf{LocatedSupremum}^{\uparrow}$ *only as a local finite-row consumer.*
7. *H records componentwise* hsame *transport for the index ledger, stream observations, regular handoffs, Real seals, dyadic bound witnesses, finite supremum rows, continuation rows, provenance rows, and local naming row.*
8. *C records displayed* Cont *replay from I to S, from S to R, from R to E, from E to B, and from the finite dyadic witness list B to the finite upper-bound ledger U.*

9. *P is the* Pkg *provenance over* $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseBoundedRealFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for infinite families, open covers, choice-selected subsequences, a host supremum, quotient-stream equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6555.2 (Pointwise bounded Real-family finite supremum obligation)**.** *Every accepted* $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ *packet exposes its bound only through displayed finite supremum rows over the inspected index window. The consumer may read the route*

$$I, S, R, E, B \longmapsto U$$

and the structural rows H, C, P, N, but it receives no host supremum, no choice-selected index family, and no completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 6555.1. The visible rows are the finite index window I, the $\mathsf{StreamName}^{\uparrow}$ observations S, the $\mathsf{RegSeqRat}^{\uparrow}$ handoffs R, the $\mathsf{Real}^{\uparrow}$ seals E, the dyadic bound witnesses B, the finite upper-bound ledger U, and the structural rows H, C, P, N.

Induct over the displayed index-window ledger I. The empty window exposes only the displayed empty upper-bound row. In the successor step, the next component is read through its $\mathsf{StreamName}^{\uparrow}$ observation row, then through its $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and only then through its $\mathsf{Real}^{\uparrow}$ seal.

For each inspected index, consume the corresponding dyadic witness in B. The componentwise hsame row preserves the witness under transport, and the Cont row replays the handoff without adding an ambient supremum coordinate.

Finally, read $\mathsf{LocatedSupremum}^{\uparrow}$ only at the finite upper-bound ledger U. This ledger displays the finite supremum rows already generated from the inspected dyadic witnesses, so the exported bound is a finite packet row rather than a host supremum, a choice-selected family, or a consequence of $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ bound read factors through displayed finite dyadic witnesses and a finite $\mathsf{LocatedSupremum}^{\uparrow}$ row before any consumer sees an upper-bound certificate.

Independence witness: streamname, regseqrat, real, locatedsupremum: $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite index-window ledger, component stream observations, regular rational handoffs, Real seals, per-index dyadic bound witnesses, finite upper-bound rows, componentwise transports, displayed continuations, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseBoundedRealFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ packet is generated from a finite inspected index window, StreamName observations, RegSeqRat handoffs, RealUp seals, per-index dyadic bound witnesses, displayed finite supremum rows, componentwise transports, continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PointwiseBoundedRealFamily}^{\uparrow}}$, seed)

Scope closed: Definition 6555.1 and Definition 6555.2 bind the finite family carrier and its finite upper-bound obligation.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PointwiseBoundedRealFamily}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim infinite Arzela–Ascoli, open-cover compactness, choice-selected subsequences, host supremum, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite index-window stability, per-component StreamName and RegSeqRat routing, Real seal non-escape, and finite LocatedSupremum upper-bound coverage for BEDC.Derived.PointwiseBoundedRealFamilyUp.

Chapter 6556

A Concrete Naming Certificate for $\mathsf{Submartingale}^{\uparrow}$

Following Doob and Meyer, a submartingale is the inequality-direction sibling of a martingale: the conditional expectation of the next visible value is required to dominate the current value rather than equal it. The $\mathsf{Submartingale}^{\uparrow}$ packet records the finite NameCert surface for this one-sided adapted-sequence object. It sits between the finite martingale carrier of chapter 190, the bounded stopping windows of chapter 383, the crossing ledger of chapter 1629, the Doob upcrossing inequality packet of chapter 1630, and the convergence-preparation surface of chapter 2246. The chapter names only the finite one-sided conditional-expectation carrier and its handoff to finite upcrossing rows; it does not assert almost-sure convergence, measure completion, or a quotient of stochastic processes.

Definition 6556.1 (Submartingale finite adapted carrier)**.** *A* $\mathsf{Submartingale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\Omega, R, C, F, X, E, I, T, H, K, P, N)$$

with the following displayed rows:

1. Ω *is the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the probability context.*
2. R *is the consumed* $\mathsf{RandomVar}^{\uparrow}$ *source packet for the finite endpoint rows.*
3. C *is the consumed* $\mathsf{CondExp}^{\uparrow}$ *packet supplying conditional-expectation endpoints.*
4. F *is the finite filtration and adaptedness ledger over a unary time spine* $0, \ldots, T$.
5. X *is the finite endpoint family* $X_0, \ldots, X_T$ *read through* $\mathsf{RandomVar}^{\uparrow}$ *rows and visible in the corresponding filtration rows.*
6. E *is the finite conditional-expectation family* $E_{n+1|n}$ *whose source row is* X_{n+1} *and whose conditioning visibility row is the* n*th filtration row.*
7. I *is the one-sided comparison ledger recording*

$$X_n \leq_R E_{n+1|n}$$

   for every displayed successor $n < T$*, with the order comparison read through the carried* $\mathsf{Real}^{\uparrow}$ *endpoint classifier.*
8. H *records componentwise* hsame *transport for* $\Omega, R, C, F, X, E, I, T, K, P, N$.

9. *K records displayed* Cont *replay for endpoint exposure, conditional-expectation exposure, filtration visibility, one-sided comparison, bounded stopping-window reads, and finite upcrossing handoff.*

10. *P is the* Pkg *provenance row naming exactly* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{MartingaleConvergence}^{\uparrow}$, *and the local submartingale rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Submartingale}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed probability, random-variable, conditional-expectation, filtration, endpoint, one-sided comparison, transport, continuation, package, and local certificate rows.

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.TasteGate.SubmartingaleTasteGate single carrier alignment`

Theorem 6556.2 (Submartingale NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{Submartingale}^{\uparrow}}$ *surface consists exactly of carrier exposure for $S = (\Omega, R, C, F, X, E, I, T, H, K, P, N)$, classifier transport of every displayed row of that carrier, endpoint exposure through* $\mathsf{RandomVar}^{\uparrow}$, *conditional-expectation exposure through* $\mathsf{CondExp}^{\uparrow}$, *filtration and adaptedness exposure through F, one-sided comparison exposure through I, bounded stopped-window routing through* $\mathsf{OptionalStopping}^{\uparrow}$, *finite crossing-ledger handoff through* $\mathsf{Upcrossing}^{\uparrow}$, *finite Doob-bound handoff through* $\mathsf{DoobUpcrossing}^{\uparrow}$, *convergence-preparation handoff through* $\mathsf{MartingaleConvergence}^{\uparrow}$, *componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through P, and local naming through N.*

Proof. Carrier exposure is Definition 6556.1. It supplies the full finite row list and fixes the local classifier to those rows.

The analytic rows are R, C, F, X, E, I. The endpoint rows are read through $\mathsf{RandomVar}^{\uparrow}$, the conditional-expectation faces are read through $\mathsf{CondExp}^{\uparrow}$, and the inequality ledger I records the one-sided comparison $X_n \leq_R E_{n+1|n}$ at each displayed successor.

The downstream rows are bounded and finite. Optional stopping can inspect only the displayed time window, the upcrossing ledger can count only lower-to-upper passages inside that window, and the Doob-upcrossing packet receives only the crossing count and expectation-bound rows.

The structural rows H, K, P, N transport, replay, source, and name the same finite packet. Since no row contains an almost-sure quotient, an unbounded stopping index, a limiting event, or a completed measure carrier, the listed obligations exhaust the seed NameCert surface. □

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier namecert obligations`

Theorem 6556.3 (Submartingale one-sided comparison transport)**.** *Componentwise* hsame *transport of the displayed probability,* $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, *filtration, endpoint, conditional-expectation, and Real-order rows preserves the one-sided comparison ledger I of an accepted* $\mathsf{Submartingale}^{\uparrow}$ *carrier. After transport, each successor window still routes through*

$$X_n \leq_R E_{n+1|n}$$

before any $\mathsf{OptionalStopping}^{\uparrow}$ *or* $\mathsf{Upcrossing}^{\uparrow}$ *consumer may read the bounded stopped-window or crossing data.*

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier one sided comparison transport`

Proof. Project the carrier of Definition 6556.1. The rows R, C, F, X, E, I are displayed components of the finite adapted packet, and H records componentwise transport for the same probability, endpoint, conditional-expectation, filtration, comparison, continuation, package, and naming rows.

Transport therefore carries the one-sided comparison row together with its endpoints. The transported ledger still reads the transported X_n, the transported $E_{n+1|n}$, and the transported $\mathsf{Real}^{\uparrow}$ order classifier before exposing any successor window.

Replay the bounded window through K and the downstream route of Definition 6556.4. Optional-stopping and upcrossing consumers receive only the transported finite comparison ledger and the displayed bounded window, not an unbounded stopping rule or an ambient process equality. □

Theorem 6556.4 (Submartingale Doob upcrossing handoff)**.** *Every accepted* $\mathsf{Submartingale}^{\uparrow}$ *carrier exports exactly the finite rows needed by a Doob-style upcrossing consumer: the displayed probability row, finite endpoint rows, finite filtration ledger, one-sided conditional-expectation comparison rows, fixed lower and upper* $\mathsf{Real}^{\uparrow}$ *boundary endpoints, bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window routes, and the resulting lower-to-upper crossing-count ledger. The handoff is a finite dependency route into* $\mathsf{Upcrossing}^{\uparrow}$ *and* $\mathsf{DoobUpcrossing}^{\uparrow}$*; it is not a martingale convergence theorem and it does not construct an almost-sure quotient.*

Proof. Project the accepted carrier of Definition 6556.1. The available stochastic rows are Ω, R, C, F, X, E, I over the displayed finite time spine.

The one-sided comparison ledger I is the only point where the submartingale direction is read. It supplies $X_n \leq_R E_{n+1|n}$ through the carried $\mathsf{Real}^{\uparrow}$ endpoint classifier and through no host-level order relation.

Bounded stopped-window reads then factor through $\mathsf{OptionalStopping}^{\uparrow}$. The crossing-count read consumes the finite endpoint rows, boundary endpoints, lower and upper decision ledgers, and stopped-window rows exactly as in $\mathsf{Upcrossing}^{\uparrow}$.

Finally the Doob-facing inequality surface consumes that finite crossing ledger through $\mathsf{DoobUpcrossing}^{\uparrow}$. The carrier contains no limiting event and no almost-sure quotient coordinate, so a downstream $\mathsf{MartingaleConvergence}^{\uparrow}$ packet receives only the finite upcrossing handoff. □

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier doob upcrossing handoff`

Theorem 6556.5 (Submartingale one-sided comparison stability)**.** *Let* $S = (\Omega, R, C, F, X, E, I, T, H, K, P, N)$ *be an accepted* $\mathsf{Submartingale}^{\uparrow}$ *carrier. Any* hsame *transport recorded in* H *preserves the one-sided conditional-expectation comparison row* I*: after transport, each displayed successor still reads the same finite comparison*

$$X_n \leq_R E_{n+1|n}$$

through the transported $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, *filtration, endpoint, and* $\mathsf{Real}^{\uparrow}$ *classifier rows. The stability statement does not assert a limiting submartingale, optional stopping outside the displayed window, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier one sided comparison stability`

Proof. Project the carrier of Definition 6556.1. The comparison row I is a displayed component of the finite packet, and H records componentwise transport for that row together with the probability, endpoint, conditional-expectation, filtration, continuation, package, and local naming rows.

Transport therefore acts on the listed row inventory rather than on a host process. The transported comparison still points to the transported endpoint row X_n, the transported conditional-expectation row $E_{n+1|n}$, and the same carried Real-order classifier.

Replay through K, P, N keeps the transported packet inside the local NameCert surface. Since no row in the carrier names an unbounded time index, a limiting event, a countable family of choices, or a completed real-space principle, comparison stability is confined to the displayed finite window. □

Theorem 6556.6 (Submartingale stopping-window handoff)**.** *Every bounded stopping-window consumer of an accepted* $\mathsf{Submartingale}^{\uparrow}$ *carrier must read, in the displayed packet order, the* $\mathsf{ProbSpace}^{\uparrow}$ *row* Ω*, the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint rows* R, X*, the* $\mathsf{CondExp}^{\uparrow}$ *row* C, E*, the sibling* $\mathsf{Martingale}^{\uparrow}$ *equality-direction context named by* P*, the bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *route, and the* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count rows before it may use the one-sided submartingale inequality. The handoff covers only the finite stopped window displayed by* T *and* K*.*

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier stopping window handoff`

Proof. Start from the NameCert obligations of Definition 6556.2. They expose the probability, endpoint, conditional-expectation, filtration, comparison, stopping-window, and crossing rows as separate finite obligations.

The bounded stopping consumer first opens Ω, R, C, F, X, E, I. These rows supply the probability context, random-variable endpoints, conditional expectations, adaptedness ledger, and one-sided comparison needed to read the inequality.

Next the package row P names the sibling $\mathsf{Martingale}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, and $\mathsf{Upcrossing}^{\uparrow}$ sources. The stopping read is still bounded by the displayed time row T and the continuation row K, so it cannot range over an unlisted stopping time.

Finally replay the Doob handoff theorem Definition 6556.4. The consumer receives only the finite stopped-window and crossing ledgers, excluding optional stopping outside the displayed window, countable choice, limiting convergence, quotient processes, and ambient Real completeness. □

Theorem 6556.7 (Submartingale upcrossing ledger exactness)**.** *For every accepted* $\mathsf{Submartingale}^{\uparrow}$ *carrier, upcrossing counts are exported only through the displayed finite stopping and comparison ledgers: the endpoint rows* X*, the comparison row* I*, the bounded stopping-window route in* T, K*, the lower and upper boundary reads supplied to* $\mathsf{Upcrossing}^{\uparrow}$*, and the Doob-facing crossing ledger named by* P*. No consumer may replace these rows by an implicit limit, an optional-stopping theorem beyond the displayed window, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SubmartingaleUp.SubmartingaleCarrier upcrossing ledger exactness`

Proof. Project the accepted carrier. The endpoint family X, comparison ledger I, finite time row T, and continuation row K are the only rows from which a crossing count can be read.

Use the Doob upcrossing handoff theorem Definition 6556.4. It identifies the exported data as the finite endpoint rows, lower and upper boundary endpoints, bounded stopping-window routes, and lower-to-upper crossing-count ledger consumed by $\mathsf{Upcrossing}^{\uparrow}$ and $\mathsf{DoobUpcrossing}^{\uparrow}$.

Ledger exactness follows because the package and naming rows P, N expose no second crossing object. Transport through H and replay through K preserve the same finite ledgers, so the exported upcrossing count cannot be strengthened into a limiting theorem, an unbounded optional-stopping rule, a choice schedule, or a Real-completeness assertion. □

Falsifiable prediction: An accepted $\mathsf{Submartingale}^{\uparrow}$ consumer that reaches an $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, or $\mathsf{MartingaleConvergence}^{\uparrow}$ handoff without the displayed $\mathsf{ProbSpace}^{\uparrow}$ source, $\mathsf{RandomVar}^{\uparrow}$ endpoint

rows, $\mathsf{CondExp}^{\uparrow}$ conditional-expectation rows, finite filtration ledger, one-sided comparison row, and bounded $\mathsf{OptionalStopping}^{\uparrow}$ route refutes the carrier surface.

Independence witness: martingale, optionalstopping, upcrossing, doob-upcrossing, martingale-convergence: no listed sibling supplies the whole submartingale packet $S = (\Omega, R, C, F, X, E, I, T, H, K, P, N)$, because $\mathsf{Martingale}^{\uparrow}$ closes equality-direction conditional expectation, $\mathsf{OptionalStopping}^{\uparrow}$ closes bounded stopped windows, $\mathsf{Upcrossing}^{\uparrow}$ closes crossing-count ledgers, $\mathsf{DoobUpcrossing}^{\uparrow}$ closes finite inequality rows, and $\mathsf{MartingaleConvergence}^{\uparrow}$ consumes convergence-preparation data while this chapter binds the one-sided adapted carrier that feeds those rows.

Closure status for $\mathsf{NameCert}_{\mathsf{Submartingale}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Submartingale}^{\uparrow}$ packet is generated from a displayed probability row, random-variable endpoint rows, conditional-expectation rows, a finite filtration and adaptedness ledger, one-sided comparison rows, bounded stopping-window routes, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Submartingale}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6556.1, Definition 6556.2, Definition 6556.3, Definition 6556.4, Definition 6556.5, Definition 6556.6, and Definition 6556.7 seed the finite one-sided martingale NameCert boundary and its Doob-style upcrossing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Submartingale}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim martingale convergence, almost-sure quotient equality, unbounded stopping, measure completion, or a standard bridge.

Upgrade path: A formal route can encode the finite carrier and prove classifier transport, one-sided comparison routing, bounded stopping-window handoff, finite upcrossing handoff, and ledger exactness.

Chapter 6557

A Concrete Naming Certificate for $\mathsf{LefschetzFixedPoint}^{\uparrow}$

Following Lefschetz fixed point theory, the $\mathsf{LefschetzFixedPoint}^{\uparrow}$ packet records only the finite chain-level trace ledger attached to a displayed endomap. It reconstructs the theorem's computable accounting surface inside BEDC by naming chain groups, boundary compatibility, induced homology rows, alternating traces, and fixed-point ledger rows, while leaving global topological fixed-point existence outside the packet.

The finite trace route is not a standalone fixed-point theorem. It consumes the compact source row of chapter 60, the self-map row of chapter 130, the chain and homology rows of chapter 98, and the trace computation rows of chapter 49 and chapter 48. The Lefschetz packet depends on these rows because its local conclusion is a finite alternating-trace accounting route, not an ambient topological existence principle.

Definition 6557.1 (Lefschetz fixed point finite-chain carrier)**.** *A* $\mathsf{LefschetzFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, K, F, A, T, H, C, R, P, N)$$

with the following displayed rows:

1. *X is the consumed* $\mathsf{NameCert}_{\mathsf{Compact}^{\uparrow}}$ *row for the finite compact source visible to the construction.*
2. *K is the finite cellular or simplicial chain ledger, including boundary rows and degree indices.*
3. *F is the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *row for the self-map whose chain-map shadow is being read.*
4. *A records the finite chain-map compatibility rows, including boundary commutation and transport of cells or basis rows.*
5. *T is the finite trace ledger over the induced homology or chain quotient rows, expressed through displayed* $\mathsf{Mat}^{\uparrow}$ *or* $\mathsf{LinMap}^{\uparrow}$ *trace data.*
6. *H records componentwise* hsame *transport for compact source, chain ledger, continuous-map row, chain-map row, trace ledger, continuation, provenance, and naming rows.*
7. *C records displayed* Cont *replay from X, K, F into A, from A into T, and from T into the local fixed-point ledger row.*
8. *R is the fixed-point accounting row: it may record a nonzero displayed Lefschetz trace, a zero trace, or a refusal that the displayed finite rows do not support a fixed-point read.*

9. *P is the* Pkg *provenance over* $\mathsf{LefschetzFixedPoint}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LefschetzFixedPoint}^{\uparrow}}$ *row.*

The carrier exports only the finite trace and fixed-point accounting ledger. It contains no ambient topological fixed-point theorem, no host singular homology machinery, no quotient space construction beyond the displayed finite rows, and no global existence claim.

Theorem 6557.2 (LefschetzFixedPoint NameCert obligations)**.** *For every accepted* $\mathsf{LefschetzFixedPoint}^{\uparrow}$ *carrier* $L = (X, K, F, A, T, H, C, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LefschetzFixedPoint}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the compact source, finite chain ledger, continuous self-map, chain-map compatibility, trace ledger, transport, replay, fixed-point accounting, provenance, and naming rows;*
2. *chain-map route, requiring every trace read to pass through X, K, F, A before any* $\mathsf{Homology}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, *or* $\mathsf{LinMap}^{\uparrow}$ *trace row is selected;*
3. *Lefschetz trace ledger, recording the displayed alternating trace rows and their finite homology or chain-quotient sources in T;*
4. *fixed-point accounting row, exposing only the local ledger conclusion R obtained from the displayed trace rows;*
5. *no-host-existence boundary, excluding global fixed-point existence, host compactness theorems, host singular homology, and undeclared topological degree machinery.*

No obligation is stored outside these finite rows.

Proof. Carrier admission is projection from Definition 6557.1. The available sources are exactly $X, K, F, A, T, H, C, R, P, N$.

The chain-map route forces trace reads through the displayed compact source, finite chain ledger, continuous-map row, and chain-map compatibility row. Thus the trace ledger T can cite only finite $\mathsf{Homology}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, or $\mathsf{LinMap}^{\uparrow}$ data already reached by that route.

The fixed-point accounting row R is downstream from T. It records the local outcome named by the packet, but it does not assert a global fixed-point theorem for arbitrary compact spaces or arbitrary continuous maps.

Finally, H, C, P, N transport, replay, source, and name the same finite data. Since none of these rows contains host singular homology, an ambient quotient construction, or a global existence theorem, those claims remain outside the local NameCert surface. □

Theorem 6557.3 (Lefschetz fixed point finite-trace ledger)**.** *Let $L = (X, K, F, A, T, H, C, R, P, N)$ be an accepted* $\mathsf{LefschetzFixedPoint}^{\uparrow}$ *carrier. Every fixed-point read exported by L factors through the finite route*

$$X, K, F \longmapsto A \longmapsto T \longmapsto R, C, P, N.$$

The exported read may cite the displayed alternating trace rows, but it cannot bypass A or T to import a host fixed-point theorem, a host homology theory, or an undeclared topological-degree computation.

Proof. Begin with an exported fixed-point request. The only visible source rows are the compact source X, the finite chain ledger K, and the self-map row F.

The request must then pass through the chain-map compatibility ledger A. Only after boundary compatibility and basis transport are displayed can the continuation route select the finite trace ledger T.

The row R records the local fixed-point accounting result attached to those traces. The structural rows C, P, N expose this result as named BEDC data, while H preserves it under accepted transport. No coordinate in the carrier supplies the excluded host machinery. □

Closure status for $\mathsf{NameCert}_{\mathsf{LefschetzFixedPoint}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LefschetzFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6557.1, Definition 6557.2, and Definition 6557.3 seed the finite Lefschetz trace packet over displayed $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LefschetzFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global classical fixed-point existence, host singular homology, host quotient-space construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chain-map routing, finite trace exactness, fixed-point accounting nonescape, transport stability, and no-host-existence boundary for BEDC.Derived.LefschetzFixedPointUp.

Constructive story: A $\mathsf{LefschetzFixedPoint}^{\uparrow}$ packet is generated from a displayed compact source, finite chain ledger, continuous self-map row, chain-map compatibility ledger, alternating trace rows, fixed-point accounting row, componentwise transport, replay, provenance, and local NameCert data.

Chapter 6558

A Concrete Naming Certificate for $\mathsf{RefutationGate}^{\uparrow}$

The $\mathsf{RefutationGate}^{\uparrow}$ packet realizes finite pseudotheory refusal as a concrete NameCert carrier. It reads audit-map rows, claimed discovery rows, structural collision rows, classifier-bit rows, axiom-dependency rows, certificate-debt rows, and taste-boundary rows, then exposes only finite refusal classes rather than a positive admission oracle.

The packet is the finite refusal surface consumed before a pseudotheory or cannot-claim export is read. The pseudotheory refusal packet of chapter 6560 and the checked-target export block of chapter 1319 consume the row-local refusal classes recorded here: $\mathsf{RefutationGate}^{\uparrow}$ supplies the claim, collision, classifier, axiom-dependency, certificate-debt, and taste-boundary classes, while those sibling packets read the same classes in pseudotheory and export-grade contexts.

Falsifiable prediction: An accepted $\mathsf{RefutationGate}^{\uparrow}$ packet is refuted if it classifies a claim as admissible merely because no synonym, classifier-neutral, axiom-dependency, or unpaid-certificate failure row has fired.

Independence witness: axiomdependencymap, auditgateboundary, auditmapfamily, groundcompilerauditmap: no listed sibling is bijective to $\mathsf{RefutationGate}^{\uparrow}$ because this packet binds claim rows, structural-DNA collision rows, classifier-bit refusal rows, axiom-dependency refusal rows, unpaid-certificate rows, taste-boundary rows, transport, replay, provenance, and local NameCert data in one carrier.

Definition 6558.1 (RefutationGate carrier)**.** *A* $\mathsf{RefutationGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, S, B, A, D, T, H, C, P, N)$$

with the following displayed rows:

1. *Q is the claim row naming the asserted discovery or theorem-facing packet under audit.*
2. *S is the structural-DNA collision row, recording synonym or reconstruction evidence against the asserted novelty of Q.*
3. *B is the classifier-bit row, recording classifier-neutral, smoke, or unsupported disagreement evidence.*
4. *A is the axiom-dependency row, recording source-token axiom use or forbidden transitive kernel dependency.*

5. *D is the certificate-debt row, recording missing semantic NameCert, ledger membership, scope seal, or row evidence.*
6. *T is the taste-boundary row, recording that taste-sensitive stability is an admission boundary and not the complement of finite refusal.*
7. *H records componentwise* hsame *transport for claim, refusal, taste-boundary, continuation, provenance, and naming rows.*
8. *C records displayed* Cont *replay from Q through S, B, A, D, T into the public refusal class.*
9. *P is the* Pkg *provenance over* $\mathsf{RefutationGate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RefutationGate}^{\uparrow}}$ *row.*

The carrier exports finite refusal classes only. It contains no positive discovery oracle, no hidden taste evaluator, and no permission to infer admission from the absence of displayed failures.

Theorem 6558.2 (RefutationGate NameCert obligations)**.** *For every accepted* $\mathsf{RefutationGate}^{\uparrow}$ *carrier $R = (Q, S, B, A, D, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RefutationGate}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the claim, structural collision, classifier-bit, axiom-dependency, certificate-debt, taste-boundary, transport, replay, provenance, and naming rows;*
2. *refusal coverage, requiring every public refusal class to be read from one of S, B, A, D or from the taste-boundary row T;*
3. *non-admission boundary, requiring the packet to distinguish finite refusal from positive taste admission;*
4. *transport stability, requiring H and C to preserve the same refusal class under accepted source transport;*
5. *no-hidden-oracle boundary, excluding global discovery certification, host taste judgment, and unledgered proof repair as packet fields.*

No local obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.RefutationGateUp.RefutationGateNamecertObligations`

Proof. Carrier admission is projection from Definition 6558.1. The visible rows are exactly $Q, S, B, A, D, T, H, C, P,$

Refusal coverage is a case split over the displayed refusal rows. Structural collision reads from S, classifier-bit failure reads from B, forbidden axiom dependency reads from A, and unpaid certificate data reads from D.

The taste-boundary row T supplies the remaining distinction. It says that stability and value admission are not computed by negating the finite refusal classes, so the packet cannot export a positive discovery judgment merely because S, B, A, D are silent.

Finally, H, C, P, N transport, replay, source, and name this same finite route. Since no coordinate contains a positive admission oracle or a hidden proof repair mechanism, such data remain outside the local NameCert surface. □

Theorem 6558.3 (RefutationGate finite refusal routing)**.** *Let $R = (Q, S, B, A, D, T, H, C, P, N)$ be an accepted* $\mathsf{RefutationGate}^{\uparrow}$ *carrier. Every refusal exported by R factors through the displayed route*

$$Q \longmapsto S, B, A, D, T \longmapsto C, P, N.$$

A downstream audit consumer may read a refusal class only from these rows, and may not replace the taste-boundary row by a positive admission rule.

Proof. Start with a downstream audit request. The claim under audit is the single row Q.

The continuation route then exposes the finite refusal family S, B, A, D together with the taste-boundary row T. These rows determine whether the packet reports structural collision, classifier-bit failure, axiom dependency, certificate debt, or taste-boundary non-admission.

The structural rows C, P, N export the selected class, and H preserves it under accepted transport. Because the route never leaves the displayed packet, no consumer can smuggle in an external positive-discovery oracle. □

The Lean TasteGate development for this packet must provide explicit $\mathsf{Nontrivial\ RefutationGate}^{\uparrow}$ and $\mathsf{FieldFaithful\ RefutationGate}^{\uparrow}$ instances. Nontriviality separates a claim with a displayed refusal row from the empty audit surface, and field faithfulness requires the classifier to preserve $Q, S, B, A, D, T, H, C, P, N$ as distinct fields before any checked carrier-alignment theorem can certify the gate.

Closure status for $\mathsf{NameCert}_{\mathsf{RefutationGate}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RefutationGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6558.1, Definition 6558.2, and Definition 6558.3 seed the finite pseudotheory refusal gate over displayed claim, collision, classifier, axiom-dependency, certificate-debt, and taste-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RefutationGate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RefutationGateUp.RefutationGateNamecertObligations`

Bridge status: none

Not claimed: This chapter does not close positive discovery certification, host taste judgment, proof repair, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refusal coverage, non-admission boundary, transport stability, no-hidden-oracle boundary, and TasteGate field faithfulness for BEDC.Derived.RefutationGateUp.

Constructive story: A $\mathsf{RefutationGate}^{\uparrow}$ packet is generated from one claim row, finite refusal ledgers, a taste-boundary row, componentwise transport, replay, provenance, and local NameCert naming.

Chapter 6559

A Concrete Naming Certificate for $\mathsf{OrbitStabilizer}^{\uparrow}$

Following Burnside and Rotman, the $\mathsf{OrbitStabilizer}^{\uparrow}$ packet records the finite group-action theorem surface as BEDC NameCert data. It sits over the group source of chapter 41, the subgroup carrier of chapter 80, the Sylow action consumer of chapter 3728, and the orbit-counting ledger of chapter 3578. The packet names a displayed orbit, stabilizer, and finite fiber-cardinality handoff without importing quotient orbit sets, host subgroup equality, or an ambient action category.

The stabilizer and fixed-orbit accounting row supplied here is the finite group-action ledger consumed by chapter 9048. Its action rows are read from the same $\mathsf{Group}^{\uparrow}$ source of chapter 41, so the Burnside consumer inherits displayed group-action data rather than an external quotient-orbit surface.

Definition 6559.1 (Orbit-stabilizer carrier)**.** *An* $\mathsf{OrbitStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, X, A, O, S, F, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Group}^{\uparrow}$ *source row, exposing the carried finite group operation, unit, inverse, and associativity data read from chapter 41.*
2. *X is the acted carrier row, exposing the finite carrier being acted on together with the row classifier used by orbit reads.*
3. *A is the action row, displaying the unit-action and multiplication-action laws as finite replay data over G and X.*
4. *O is the orbit row for a displayed point $x \in X$, recording the finite list of action outputs $g \cdot x$ together with duplicate control under the local classifier.*
5. *S is the stabilizer-subgroup row, recording the carried subgroup of G whose elements fix x and reading its closure through chapter 80.*
6. *F is the finite fiber-cardinality ledger. It records that the action map $g \mapsto g \cdot x$ has exactly the stabilizer fibers named by S, so the group-cardinality row factors through the orbit row and stabilizer row before any count is exported.*
7. *H records componentwise* hsame *transport for the group, acted carrier, action, orbit, stabilizer, fiber-cardinality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the finite action route from* $\mathsf{Group}^{\uparrow}$ *and X through A, O, S, F into* $\mathsf{OrbitStabilizer}^{\uparrow}$.
9. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, *finite action rows, orbit rows, stabilizer rows,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OrbitStabilizer}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. It has no coordinate for quotient orbit sets, host coset representatives, ambient subgroup equality, selected transversal data, or a complete finite-group classification.

Theorem 6559.2 (Orbit-stabilizer NameCert obligations)**.** *For an accepted* $\mathsf{OrbitStabilizer}^{\uparrow}$ *carrier*

$$T = (G, X, A, O, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OrbitStabilizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Group}^{\uparrow}$ *source, acted carrier, action row, orbit row, stabilizer-subgroup row, finite fiber-cardinality ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Proof. Project the accepted packet onto its ten rows. The group, acted carrier, action, orbit, stabilizer, and fiber-cardinality data are displayed before the local name can be read.

The stabilizer row is read as subgroup data over the same group row G, so it cannot be replaced by a host subgroup equality claim. The orbit row is read through the action row A, so it cannot be replaced by a quotient-orbit object.

Finally, H, C, P, N transport, replay, source, and name only the displayed rows. Since no excluded object has a carrier coordinate, no local NameCert obligation can depend on quotient orbit sets, selected transversals, or ambient finite-group classification. □

Theorem 6559.3 (Orbit-stabilizer fiber-cardinality handoff)**.** *Let $T = (G, X, A, O, S, F, H, C, P, N)$ be an accepted* $\mathsf{OrbitStabilizer}^{\uparrow}$ *carrier. Every cardinality read of the orbit-stabilizer route factors through the finite ledger F after the orbit row O and stabilizer row S have both been displayed. Consequently a Sylow-style or Burnside-style consumer may cite the cardinality handoff only as a finite fiber ledger over the action row, not as a host quotient or selected coset enumeration.*

Proof. Read the action map from A and the displayed point from the acted carrier row X. The orbit row O lists the action outputs, while the stabilizer row S records exactly the group elements fixing the point.

The ledger F is then the only row where fibers of the action map are counted. It compares a fiber with the stabilizer row before an orbit-cardinality handoff is exported.

Transport through H and replay through C preserve that row order. Thus the downstream count seen by chapter 3728 or chapter 3578 remains a finite BEDC ledger, not a quotient orbit set or selected transversal. □

Theorem 6559.4 (Orbit-stabilizer action nonescape)**.** *Every consumer of an accepted* $\mathsf{OrbitStabilizer}^{\uparrow}$ *carrier factors through*

$$\mathsf{Group}^{\uparrow}, \mathsf{Subgroup}^{\uparrow}, X, A, O, S, F, H, C, P, N.$$

The packet exports a finite group action, a displayed orbit, a stabilizer subgroup row, and a fiber-cardinality ledger, but it exports no quotient orbit carrier, no host subgroup equality, no selected coset representatives, and no ambient classification theorem for finite groups.

Proof. Any accepted read starts by projecting the finite tuple. The action row is the only place where the group acts on the carried set, and the orbit and stabilizer rows are the only endpoints of that action read.

The fiber-cardinality ledger F is a displayed handoff from the action map to the orbit and stabilizer rows. It is not a quotient construction and it does not choose representatives.

The remaining rows H, C, P, N preserve and name the same finite surface. Therefore every downstream consumer must stay inside the displayed group-action data, while quotient orbit carriers, host subgroup equality, selected representatives, and finite-group classification remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{OrbitStabilizer}\uparrow}$.

Constructive story: An $\mathsf{OrbitStabilizer}^{\uparrow}$ packet is generated from a displayed finite GroupUp row, an acted carrier, an action row, an orbit row, a stabilizer-subgroup row, a finite fiber-cardinality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrbitStabilizer}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6559.1, Definition 6559.2, Definition 6559.3, and Definition 6559.4 seed the finite group-action theorem carrier over orbit, stabilizer, and fiber-cardinality rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrbitStabilizer}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient orbit sets, selected coset representatives, ambient subgroup equality, finite-group classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action replay, stabilizer-subgroup exposure, finite fiber-cardinality handoff, transport, provenance, and action non-escape for BEDC.Derived.OrbitStabilizerUp.

Chapter 6560

A Concrete Naming Certificate for $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$

The $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ packet realizes the refusal surface of chapter 9412 as finite NameCert data. It records the rows by which a read may be refused for structural reconstruction, classifier-neutral padding, forbidden axiom dependency, unpaid discovery evidence, or taste-boundary non-automation. The packet is apophatic: it blocks overclaimed discovery reads before any positive admission judgment is exported.

Theorem 6560.1 (Pseudotheory refutation sibling dependency route)**.** *The* $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ *packet refines the refusal classes of chapter 6558 in the pseudotheory context. Its structural collision, classifier-neutral padding, axiom-dependency, certificate-debt, taste-boundary, transport, replay, provenance, and local naming rows are read as pseudotheory-specialized refusal rows, not as a separate kernel-only isolation surface.*

Proof. Read the general $\mathsf{RefutationGate}^{\uparrow}$ refusal route first. It supplies the carrier shape for finite refusal classes, transport, replay, provenance, and local naming.

The pseudotheory packet specializes those rows by naming structural reconstruction collisions, classifier-neutral padding, forbidden axiom dependency, unpaid certificate data, and taste-boundary refusal before any conclusion is exported. Since every row is still read through the displayed refusal route, the chapter records a concrete sibling dependency on $\mathsf{RefutationGate}^{\uparrow}$ while keeping its own seed-level NameCert surface. □

Definition 6560.2 (Pseudotheory refutation gate carrier)**.** *A* $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, K, C, A, U, T, H, Q, P, N)$$

with the following displayed rows:

1. *D is the structural-DNA collision row, recording a finite payload comparison with a prior grounded carrier.*
2. *K is the classifier-neutral padding row, recording that a proposed surface changes no accepted classifier bit before a discovery assertion is read.*
3. *C is the axiom-crossing dependency row, recording a forbidden source token or transitive dependency surface for a claimed BEDC theorem.*
4. *A is the unpaid discovery-data row, recording missing semantic naming, row evidence, ledger membership, debt, or scope-seal data for a positive-discovery claim.*

5. *U is the taste-boundary row, recording that stability, usefulness, and fit with the surrounding theory remain admission judgments and are not automatic demotion results.*
6. *T records componentwise* hsame *transport for the five refusal rows and for replay, provenance, and local naming.*
7. *H records displayed* Cont *replay of the refusal route from a candidate read to exactly one exposed finite refusal row before any conclusion is exported.*
8. *Q is the audit-query provenance row tying the read to axiom-dependency maps, query ledgers, audit boundaries, audit-map families, and ground-compiler audit maps.*
9. *P is the* Pkg *provenance over* BHist*, refusal rows,* hsame*,* Cont*, audit-query data, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationGate}^{\uparrow}}$ *row.*

The carrier contains no coordinate that turns refusal into a positive discovery proof, carrier repair recipe, or taste decision.

Theorem 6560.3 (Pseudotheory refutation gate NameCert obligations)**.** *For an accepted* PseudotheoryRefutationG *carrier*

$$R = (D, K, C, A, U, T, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationGate}^{\uparrow}}$ *surface consists exactly of carrier admission for the structural-DNA collision row, classifier-neutral padding row, axiom-crossing dependency row, unpaid discovery-data row, taste-boundary row, transport, replay, audit-query provenance, package provenance, and local naming.*

Proof. Carrier admission is projection from the ten displayed rows. The five refusal rows are all visible before the local name is read.

The transport and replay rows preserve refusal-row identity. A structural collision cannot be replayed as an axiom dependency, and a taste-boundary row cannot be replayed as a mechanical discovery failure.

The provenance rows Q and P name the audit surfaces used by the read, while N names only the local packet. Since the carrier has no positive-admission coordinate, no NameCert obligation can turn the refusal surface into a discovery proof. □

Theorem 6560.4 (Finite refusal soundness)**.** *Every accepted* $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ *read exposes at least one of the finite refusal rows D, K, C, A, U before any refusal conclusion is exported. The exported conclusion is therefore a row-local refusal over the displayed audit surface, not a global proof that the candidate has no repair or no future value.*

Proof. Start with an accepted carrier. The replay row H sends a candidate read to the finite refusal surface, and the carrier predicate admits only the five displayed refusal rows as conclusions.

If the read is structural, it factors through D; if it is classifier-neutral, through K; if it crosses an axiom boundary, through C; if it lacks discovery data, through A; and if it reaches taste-sensitive stability, through U.

The audit-query and provenance rows record where the refusal was read. They do not add a positive discovery certificate or a global impossibility theorem, so the conclusion remains finite and row-local. □

Theorem 6560.5 (Taste-boundary nonescape)**.** *The taste-boundary row U of an accepted* PseudotheoryRefutationG *carrier may refuse automatic demotion or automatic admission, but it cannot export a positive*

discovery judgment, carrier replacement, or theorem-worthiness proof. Any consumer that cites U must keep the read inside the displayed taste-boundary row, its hsame *transport, its* Cont *replay, provenance, and local* NameCert.

Proof. Project the row U from the accepted carrier. It is a boundary row: it records where stability and fit require judgment rather than a mechanical gate.

Transport through T preserves that boundary status, and replay through H keeps the row inside the same refusal route. The provenance rows Q and P record the audit source without changing the kind of the row.

Because the carrier contains no positive-admission coordinate, a consumer cannot obtain a discovery proof, carrier replacement, or theorem-worthiness result from U. The only exported data are the displayed boundary row and its transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ read exposes one finite refusal row among D, K, C, A, U before any demotion conclusion is exported.

Independence witness: axiomdependencymap, axiomqueryledger, auditgateboundary, auditmapfamily, groundcompileauditmap: none is bijective to $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ because this packet jointly carries structural-DNA collision, classifier-neutral padding, axiom-crossing dependency, unpaid discovery-data, taste-boundary, transport, replay, audit-query provenance, package provenance, and local NameCert rows.

The Lean follow-up is the TasteGate file for $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$, whose carrier interface must provide both Nontrivial and FieldFaithful rows for the finite refusal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationGate}^{\uparrow}}$.

Constructive story: A $\mathsf{PseudotheoryRefutationGate}^{\uparrow}$ packet is generated from structural-DNA collision, classifier-neutral padding, axiom-crossing dependency, unpaid discovery-data, taste-boundary, hsame transport, Cont replay, audit-query provenance, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6560.2, Definition 6560.3, Definition 6560.4, and Definition 6560.5 seed the finite refusal carrier for pseudotheory audit reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive discovery, automatic taste admission, carrier replacement, theorem-worthiness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite refusal-row exposure, taste-boundary non-escape, transport, replay, provenance, and local naming for BEDC.Derived.PseudotheoryRefutationGateUp.

Chapter 6561

A Concrete Naming Certificate for $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ packet records the separated regular-Cauchy completion seal between rational regular names and terminal Real names as finite BEDC NameCert data. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet supplies the non-quotient separatedness certificate needed before a regular Cauchy name can feed Real and Completion consumers.

Definition 6561.1 (Regular-Cauchy separated-completion carrier)**.** *A* $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, S, D, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which R is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the displayed Cauchy budgets.*
4. *Z is the separated zero-distance row asserting that two completed reads agree only through the displayed regular-Cauchy windows and dyadic budgets.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available after R, S, D, Z have been exposed.*
6. *H records componentwise* hsame *transport across the regular name, stream windows, tolerance ledger, separatedness row, Real seal, and certificate rows.*
7. *C records displayed* Cont *replay from a completion request to R, S, D, Z, E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, Z, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ BEDC.Derived.RegularCauchySeparatedCompletionUp.RegularCauchySeparatedCompletionTasteGate
single carrier alignment

Theorem 6561.2 (Regular-Cauchy separated-completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ *carrier* $Q = (R, S, D, Z, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySe}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the separated zero-distance row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6561.1. The only visible coordinates are $R, S, D, Z, E, H, C, P, N$.

The completion-facing rows are ordered. The regular rational name is read through finite stream windows, those windows are bounded by the dyadic ledger, and the separated zero-distance row is downstream of both.

The terminal Real seal is therefore not a quotient of names. Transport, replay, provenance, and local naming preserve the displayed rows, and the excluded equality, selector, choice, trichotomy, and ambient-completeness coordinates cannot be projected from the carrier. □

Theorem 6561.3 (Regular-Cauchy separated-completion zero-distance route)**.** *Every zero-distance read accepted by a* $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, two Real-facing completion reads may be identified only after the regular rational source row, finite stream-window schedule, dyadic tolerance ledger, separatedness row, and terminal Real seal have all been displayed in the same finite packet.

Proof. Read the source row R through the schedule S. This is the only route by which the packet exposes the regular Cauchy name to a completion-facing consumer.

Read the tolerance row D and separatedness row Z next. They bind zero-distance agreement to finite windows and dyadic budgets rather than to a host quotient or selected representative.

Finally consume the Real seal E through the structural rows H, C, P, N. Since the carrier has no further equality coordinate, the accepted zero-distance read cannot escape the displayed route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ consumer that identifies Real-facing completion reads without reading a RegSeqRat source row, StreamName window schedule, DyadicRatCore tolerance ledger, and separated zero-distance row refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it binds a regular rational name, finite windows, a dyadic tolerance ledger, a separated zero-distance row, and a downstream Real seal into one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite windows, a DyadicRatCore tolerance ledger, a separated zero-distance row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6561.1, Definition 6561.2, and Definition 6561.3 seed the separated regular-Cauchy completion seal for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchySeparatedCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and zero-distance separatedness route.

Chapter 6562

A Concrete Naming Certificate for $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$

Following Borsuk and Dugundji, the $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$ packet records the neighbourhood-local retract-extension surface used by compact and located metric analysis. It refines the retract route of chapter 6469, reads compact source data through chapter 129, reads distance comparison through chapter 1533, and consumes the closed-extension and separation anchors of chapter 1956 and **??**. The packet names only displayed neighbourhood, extension, and retraction rows; it does not assert a selected neighbourhood-retraction principle.

Definition 6562.1 (Absolute neighbourhood retract carrier)**.** *An* $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, K, L, U, T, V, E, R, S, H, C, P, N)$$

where:

- X *is the displayed source carrier row;*
- K *is the compact-metric source row read through* $\mathsf{CompactMetric}^{\uparrow}$*;*
- L *is the located-metric comparison row read through* $\mathsf{LocatedMetric}^{\uparrow}$*;*
- U *is the displayed neighbourhood row around the closed source packet;*
- T *is the* $\mathsf{TietzeExtension}^{\uparrow}$ *closed-extension row;*
- V *is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *separating-function row used to expose* U *as a displayed neighbourhood boundary;*
- E *records the finite extension request over the closed source row;*
- R *records the candidate retraction from the displayed neighbourhood row back to the source row;*
- S *records stability of the retraction after the extension and separation rows are visible;*
- H, C, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}}$ *row.*

Theorem 6562.2 (Absolute neighbourhood retract NameCert obligations)**.** *For every accepted* $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$ *carrier* $A = (X, K, L, U, T, V, E, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AbsoluteNeighborho}}$ *surface consists exactly of source admission, compact-metric readback, located-metric comparison,*

displayed neighbourhood exposure, Tietze extension exposure, Urysohn separation exposure, finite extension request, neighbourhood retraction row, retraction-stability row, componentwise transport, displayed replay, provenance, and local naming. A consumer that omits one of $K, L, U, T, V, E, R, S, H, C, P, N$ is outside the packet.

Proof. Project the carrier of Definition 6562.1. Each coordinate is finite BHist data, and the local naming row N names only the displayed packet.

The compact and located-metric rows provide the source scope. The neighbourhood row U is then exposed through the separation route, and the closed-extension row T supplies the extension side of the retract-extension packet.

Finally read E, R, S, H, C, P, N. These rows package the extension request, neighbourhood retraction, stability, transport, replay, provenance, and name without adding a selected neighbourhood-retraction theorem, quotient compactum, open-cover compactness argument, or choice of extension operators. □

Theorem 6562.3 (Absolute neighbourhood extension-retraction route)**.** *Every accepted* AbsoluteNeighborhoodRetr *packet whose retraction row R is consumed must first expose the compact-metric row K, located-metric row L, displayed neighbourhood row U, Tietze extension row T, Urysohn separation row V, and finite extension request E. The retraction read is then transported through S, H, C, P, N as a neighbourhood-local BEDC route, not as an ambient absolute retract theorem or a selected family of retractions.*

Proof. Begin with the NameCert obligations of Definition 6562.2. They make $K, L, U, T, V, E, R, S, H, C, P, N$ the visible surface for any accepted packet.

The extension side is read before the retraction. The compact-metric and located-metric rows bound the source, the Urysohn row exposes the neighbourhood boundary, and the Tietze row records the closed-extension data over that boundary.

The retraction row R is therefore a neighbourhood-local coordinate of the same finite packet. The stability, transport, replay, provenance, and local naming rows keep the route inside S, H, C, P, N, so no arbitrary selected neighbourhood retraction, quotient compactum, open-cover compactness principle, or chosen extension operator is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteNeighborhoodRetract}\uparrow}$.

Constructive story: An AbsoluteNeighborhoodRetract$^{\uparrow}$ packet is generated from CompactMetricUp source rows, LocatedMetricUp comparison rows, displayed neighbourhood rows, TietzeExtensionUp extension rows, UrysohnLemmaUp separation rows, finite extension and retraction rows, stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AbsoluteNeighborhoodRetract}\uparrow}$, seed)

Scope closed: Definition 6562.1, Definition 6562.2, and Definition 6562.3 seed the neighbourhood-local retract-extension packet over compact-metric, located-metric, extension, and separation rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AbsoluteNeighborhoodRetract}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary selected neighbourhood retractions, quotient compacta, open-cover compactness, choice of extension operators, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric readback, located-metric comparison, neighbourhood exposure, Tietze extension exposure, Urysohn separation exposure, retraction stability, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteNeighborhoodRetractUp.

Chapter 6563

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyMap}^{\uparrow}$

Following Bishop and Bridges on regulated functions and constructive completions, the $\mathsf{RegulatedCauchyMap}^{\uparrow}$ packet records the L10 interface by which a regulated approximant family acts on Cauchy names. It sits after the regulated-function surface of chapter 1906, the Cauchy-continuous map surface of chapter 1401, the regular rational readback of chapter 301, the finite observation schedule of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite handoff from regulated approximants to completion-facing Cauchy names; it does not assert host function equality, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6563.1 (Regulated Cauchy map carrier)**.** *A* $\mathsf{RegulatedCauchyMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, A, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, exposing only finite step-window, endpoint, and oscillation data available to the approximant family.*
2. *A is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *action row, recording that displayed Cauchy input windows are sent to displayed Cauchy output windows through the same regulated approximants.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule for the input and output Cauchy names.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed output windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the regulated approximant, input window, and output window at each inspected stage.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, A, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the regulated source, Cauchy action, stream windows, regular readback, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a regulated Cauchy-map request through F, A, S, R, D, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyMap}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyMap}^{\uparrow}}$ *row.*

Accepted packets are compared only through $F, A, S, R, D, E, H, C, P, N$ and have no coordinate for host function equality, an external completion operator, quotient Cauchy streams, selected limiting points, countable choice, or ambient real completeness.

Theorem 6563.2 (Regulated Cauchy map carrier admission)**.** *For every accepted* $\mathsf{RegulatedCauchyMap}^{\uparrow}$ *carrier* $M = (F, A, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyMap}^{\uparrow}}$ *surface consists exactly of regulated-function source exposure, Cauchy-continuous action exposure, finite stream-window exposure, regular rational readback, dyadic tolerance comparison, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6563.1. The row F is the only regulated-function source, and A is the only row that acts on Cauchy input and output names.

The Cauchy-name data is then read through S, R, D. Finite stream windows are opened first, regular rational readback is attached to those windows, and dyadic tolerances record the comparisons used by the regulated approximants.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host function equality, selected limits, quotient streams, or ambient completeness, the listed rows exhaust the admission surface. □

Theorem 6563.3 (Regulated Cauchy map completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegulatedCauchyMap}^{\uparrow}$ *packet must read the route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{CauchyContinuousMap}^{\uparrow}(A),\ \mathsf{StreamName}^{\uparrow}(S),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{Real}^{\uparrow}$$

before exporting a Real-facing output. The handoff transports regulated approximants over Cauchy names as finite BEDC rows, not as a host-level function between completed real spaces.

Proof. Start with an accepted carrier $M = (F, A, S, R, D, E, H, C, P, N)$. The source approximant data is visible only in F, and the Cauchy action is visible only in A.

The output Cauchy name is not sealed directly. It first exposes finite observations through S, regular rational readback through R, and dyadic tolerance comparisons through D.

Only after those rows are present may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed. The structural rows H, C, P, N preserve this order, so the completion handoff cannot import host function equality, quotient Cauchy streams, selected limits, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegulatedCauchyMap}^{\uparrow}$ completion-facing read must expose regulated-function, Cauchy-continuous action, StreamName, RegSeqRat, DyadicRatCore, Real seal, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: regulated-function, cauchycontinuousmap, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegulatedCauchyMap}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries regulated approximants, Cauchy-name action, finite stream observations, regular readback, dyadic tolerances, terminal Real sealing, transport, replay, provenance, and local naming as one L10 interface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyMap}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedCauchyMap}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a CauchyContinuousMap action row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedCauchyMap}^{\uparrow}}$, seed)

Scope closed: Definition 6563.1, Definition 6563.2, and Definition 6563.3 seed the finite L10 regulated-Cauchy action interface over regulated approximants and Cauchy-name readback rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegulatedCauchyMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host function equality, selected limits, quotient Cauchy streams, countable choice, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-source exposure, Cauchy action exactness, finite-window readback, dyadic tolerance nonescape, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedCauchyMapUp.

Chapter 6564

A Concrete Naming Certificate for RealUniformCauchyFilterCompletion$^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges constructive analysis, the RealUniformCauchyFilterCompletion$^{\uparrow}$ packet names the finite handoff from the Real-induced uniformity to a Cauchy-filter completion consumer. It sits after the RealUniformStructure$^{\uparrow}$ route of chapter 1624, the completion surface of chapter 1569, the filter source of chapter 333, the finite window rows of chapter 126, the regular rational readback rows of chapter 301, the dyadic radius ledger of chapter 291, and the Real seal of chapter 38.

Definition 6564.1 (Real uniform Cauchy-filter completion carrier)**.** *A* RealUniformCauchyFilterCompletion$^{\uparrow}$ *carrier is a packet*

$$Q = (R, U, F, W, Q_R, E, H, C, P, N)$$

where R is the displayed RealUniformStructure$^{\uparrow}$ *source row, U is the induced uniform-entourage row, F is the* CauchyFilterCompletion$^{\uparrow}$ *completion-facing row reached through a displayed* CauchyFilter$^{\uparrow}$ *basis, W is the finite* StreamName$^{\uparrow}$ *window row, Q_R is the* RegSeqRat$^{\uparrow}$ *readback row over the same window, E is the* DyadicRatCore$^{\uparrow}$ *radius and error ledger ending in a* Real$^{\uparrow}$ *seal, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance over the listed rows, and N is the local* NameCert *row. The carrier predicate asserts that every completion read reaches F only after the rows $R, U, W, Q_R, E, H, C, P, N$ have been displayed in one finite* BHist *packet.*

Lean4 (statement) `BEDC.Derived.RealUniformCauchyFilterCompletionUp`

Theorem 6564.2 (Real uniform completion entourage route)**.** *Every accepted* RealUniformCauchyFilterCompletion$^{\uparrow}$ *packet $Q = (R, U, F, W, Q_R, E, H, C, P, N)$ routes a completion-facing entourage request through the* RealUniformStructure$^{\uparrow}$ *source row R, the generated uniform row U, the* CauchyFilter$^{\uparrow}$ *basis displayed inside the* CauchyFilterCompletion$^{\uparrow}$ *row F, the finite* StreamName$^{\uparrow}$ *window row W, the* RegSeqRat$^{\uparrow}$ *readback row Q_R, the* DyadicRatCore$^{\uparrow}$ *radius ledger E, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before it is offered to any completion consumer.*

Proof. Project the carrier rows. The only row that can expose a Real-induced entourage is R, and the only row that can present the uniform basis to completion consumers is U.

The Cauchy-filter completion row F is downstream of that basis. A completion read must therefore pass through the displayed CauchyFilter$^{\uparrow}$ basis and then through the finite window and readback rows W, Q_R before it can inspect the completion-facing row.

The dyadic radius ledger E supplies the finite tolerance data and the terminal $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and naming are confined to H, C, P, N, so the route remains one finite BHist handoff. □

Theorem 6564.3 (Real uniform completion window handoff)**.** *Let $Q = (R, U, F, W, Q_R, E, H, C, P, N)$ be an accepted* $\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}$ *carrier. Any completion-facing tail comparison over the Real-induced uniformity reads the* $\mathsf{StreamName}^{\uparrow}$ *window row W and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q_R before it can consume the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row F or the terminal* $\mathsf{Real}^{\uparrow}$ *seal recorded by E.*

Proof. Start with a tail comparison addressed to the completion route. The carrier exposes the stream coordinates only through W, so no comparison can inspect the Real source before the finite window is displayed.

The readback row Q_R is attached to the same window. It converts the inspected coordinates into the regular rational data consumed by the dyadic radius ledger E and by the Cauchy-filter basis inside F.

Finally H, C, P, N transport, replay, source, and name this same handoff. Thus the completion consumer receives a windowed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route rather than an ambient sequence, selected representative, or quotient-completion equality. □

Theorem 6564.4 (Real uniform completion nonescape)**.** *Every consumer of an accepted* RealUniformCauchyFilterC *packet is confined to the displayed* $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. The packet exports no quotient filter, quotient real equality, selected limit, countable choice schedule, open-cover compactness, or ambient completeness theorem.*

Proof. Inspect the coordinates of the carrier. They list a Real-induced uniform source, a generated uniform row, a completion-facing Cauchy-filter row, finite stream windows, regular rational readback, dyadic tolerance data, transport, replay, provenance, and local naming.

Apply the entourage route and the window handoff. Together they force each completion-facing request through the same finite rows before any completion read is available.

There is no carrier coordinate for quotient filters, quotient real equality, selected limits, countable choice, open-cover compactness, or ambient completeness. Hence those claims cannot be exported by the local packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}$ packet is generated from a displayed $\mathsf{RealUniformStructure}^{\uparrow}$ source, an induced uniform-entourage row, a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row reached through a $\mathsf{CauchyFilter}^{\uparrow}$ basis, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger ending in a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6564.1, Definition 6564.2, Definition 6564.3, and Definition 6564.4 seed the finite completion-facing route from Real-induced uniformity to Cauchy-filter completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.RealUniformCauchyFilterCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient filters, quotient real equality, selected limits, countable choice, open-cover compactness, ambient Real completeness, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage routing, window handoff, and nonescape for the same packet.

Chapter 6565

A Concrete Naming Certificate for $\mathsf{BaireOneOscillationRank}^{\uparrow}$

Following Baire, Kechris, and Bishop–Bridges constructive analysis, the $\mathsf{BaireOneOscillationRank}^{\uparrow}$ packet records the finite NameCert surface by which a Baire-one real-valued name exposes a displayed oscillation-rank comparison without constructing a quotient space of functions or a host transfinite hierarchy. It sits over the Baire-one source of chapter 1604, the lower-semicontinuous epigraph boundary of chapter 1536, the terminal real-name seal of chapter 38, the regular-rational readback of chapter 301, the finite stream schedule of chapter 126, and the rank-comparison sibling of chapter 6262. The packet names a constructive descriptive-analysis certificate for displayed Baire-one oscillation ranks; it is not a classical completeness theorem, not a selected ordinal rank for every function, and not a quotient-real or quotient-function construction.

Definition 6565.1 (Baire-one oscillation rank carrier)**.** *A* $\mathsf{BaireOneOscillationRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, L, R, S, Q, W, E, K, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *source row, including the pointwise approximant ledger and the real-valued request being inspected.*
2. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph row for the local oscillation-bound predicate attached to F.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row through which all endpoint and oscillation-value reads are exposed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite windows of source points, approximants, and tolerance requests used by the inspection.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the displayed tolerance and value-window comparisons.*
6. *W is the finite oscillation-window row, recording the two value observations, their common source window, and the rational bound compared by Q.*
7. *E is the epigraph-survival row, recording whether the displayed lower-semicontinuous oscillation set survives the named derivative stage.*
8. *K is the rank-comparison handoff row to* $\mathsf{OscillationRankFunction}^{\uparrow}$*, carrying only the displayed stage, survival bit, and boundary tolerance.*

9. *H records componentwise* hsame *transport for the Baire-one source, lower-semicontinuous epigraph, real seal, stream schedule, regular-rational readback, oscillation window, survival row, rank handoff, continuation, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Baire-one request through S, Q, R, W, L, E, K and then through the structural rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BaireOneOscillationRank}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{OscillationRankFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireOneOscillationRank}^{\uparrow}}$ *row.*

The carrier exports no host function-space quotient, no selected transfinite rank, no countable-choice schedule, no quotient real equality, and no ambient completeness principle.

Theorem 6565.2 (Baire-one oscillation rank NameCert obligations)**.** *For every accepted* $\mathsf{BaireOneOscillationRank}^{\uparrow}$ *carrier $O = (F, L, R, S, Q, W, E, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireOneOscillationRank}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; Baire-one source compatibility for F; lower-semicontinuous oscillation-epigraph exposure through L; real-value sealing through R; finite stream scheduling through S; regular-rational tolerance readback through Q; oscillation-window exactness through W; derivative-stage survival exposure through E; rank-comparison handoff through K; componentwise transport stability through H; continuation replay through C; provenance and local naming through P, N; and ledger non-escape for host function spaces, selected transfinite ranks, countable-choice schedules, quotient real equality, and ambient completeness.*

Proof. Project the carrier of Definition 6565.1. The accepted packet exposes exactly the rows $F, L, R, S, Q, W, E, K, H, C, P, N$, so every certificate obligation must be read from those coordinates.

The source and value obligations are F, R, S, Q, W. The Baire-one source fixes the pointwise approximant ledger, the stream row fixes the finite inspection windows, the regular-rational row fixes the displayed tolerance comparison, the real row seals the value observations, and the oscillation-window row records the two compared value reads.

The rank-facing obligation is L, E, K. The lower-semicontinuous row supplies the epigraph boundary for the oscillation predicate, the survival row records only the named derivative-stage test, and the handoff row packages that finite comparison for $\mathsf{OscillationRankFunction}^{\uparrow}$ consumers.

Transport, replay, provenance, and naming add no hidden coordinates. The rows H, C, P, N preserve, replay, source, and name the same displayed route. Since no coordinate stores a host function-space quotient, a selected rank for every function, a countable-choice schedule, quotient real equality, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6565.3 (Baire-one oscillation rank constructive handoff)**.** *Every descriptive-analysis consumer of an accepted* $\mathsf{BaireOneOscillationRank}^{\uparrow}$ *carrier factors through the displayed finite route*

$$F, S, Q, R, W, L, E, K \longmapsto H, C, P, N.$$

Thus the consumer may inspect the Baire-one source, the finite stream windows, the regular-rational tolerance reads, the real-value seals, the oscillation-window comparison, the lower-semicontinuous epigraph boundary, and the displayed rank-survival handoff, but it receives no quotient function space, no selected ordinal rank, no countable-choice schedule, no quotient real equality, and no ambient real-completion theorem.

Proof. Start with a consumer of the handoff row K. Carrier admission makes K available only after the Baire-one source F, stream schedule S, regular-rational tolerance row Q, real-value seal R, oscillation-window row W, lower-semicontinuous epigraph row L, and survival row E have been displayed.

Replay the continuation row C. The request first opens the finite Baire-one value windows, then reads the rational and real comparison data, then passes through the lower-semicontinuous epigraph boundary, and only then exposes the rank-survival comparison used by $\mathsf{OscillationRankFunction}^{\uparrow}$.

The exported surface is therefore the finite route stated in the theorem. Componentwise transport through H preserves the displayed coordinates, and P, N source and name only that route. None of the excluded host objects can be recovered from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOneOscillationRank}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireOneOscillationRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6565.1, Definition 6565.2, and Definition 6565.3 seed the finite constructive-analysis certificate for displayed Baire-one oscillation-rank comparisons.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireOneOscillationRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host function-space quotient, selected transfinite rank, countable-choice schedule, quotient real equality, ambient real-completion theorem, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, lower-semicontinuous epigraph exposure, real sealing, stream scheduling, regular-rational readback, oscillation-window exactness, survival exposure, rank handoff, transport, replay, provenance, and local naming for BEDC.Derived.BaireOneOscillationRankUp.

Constructive story: A $\mathsf{BaireOneOscillationRank}^{\uparrow}$ packet is generated from BaireOneFunction, LowerSemicontinuous, RealUp, StreamName, RegSeqRat, OscillationRankFunction, transport, continuation, provenance, and local NameCert rows.

Chapter 6566

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusFold}^{\uparrow}$

Following Bishop–Bridges constructive compactness and the finite-fold construction of chapter 9237, the $\mathsf{CompactUniformModulusFold}^{\uparrow}$ packet records the concrete NameCert object behind the compact uniform-continuity modulus ledger. It sits after the compact source of chapter 60, the continuous-map graph surface of chapter 130, the metric distance rows of chapter 57, the rational-order readback of chapter 37, and the uniform-modulus consumer of chapter 369. The packet names a finite ProbeBundle fold; it does not select an external modulus, invoke open-cover compactness outside the displayed carrier, or import host function equality.

Definition 6566.1 (Compact uniform modulus fold carrier)**.** *A* $\mathsf{CompactUniformModulusFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, M, Q, F, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row exposing the located finite* ProbeBundle *net used by the fold.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row over the same metric source.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *source-and-target distance row, including the triangle and symmetry data consumed by the final estimate.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius and finite lower-bound comparison row.*
5. *F is the finite fold ledger over the* ProbeBundle *radius family, recording the accumulator, successor comparisons, and certified lower-bound output.*
6. *L is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, consuming the folded lower bound as a tolerance-only precision ledger.*
7. *H records componentwise* hsame *transport for compact source, continuous graph, metric distances, rational comparisons, fold ledger, uniform-modulus handoff, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a compact-uniform request through K, G, M, Q, F, L, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusFold}^{\uparrow}}$ *row.*

Accepted packets are compared only through $K, G, M, Q, F, L, H, C, P, N$ and have no coordinate for an external choice of modulus, arbitrary open-cover compactness, quotient function equality, completed function space, or ambient real completeness.

Theorem 6566.2 (Compact uniform modulus fold NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusFold}^{\uparrow}$ *carrier $U = (K, G, M, Q, F, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusFold}}$ *surface consists exactly of compact finite-net exposure, continuous-map pointwise-modulus exposure, metric source-and-target distance exposure, rational lower-bound readback, finite* ProbeBundle *fold replay, uniform-modulus handoff, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6566.1. The compact source is K, the continuous graph is G, the metric distance support is M, and the rational comparison support is Q.

The fold row F is downstream of those sources. It records primitive recursion over the displayed ProbeBundle radius family and keeps only certified lower-bound comparisons visible through Q.

The uniform-modulus row L is then a consumer of the fold, not an external selector. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so no arbitrary cover, host function equality, or completed function-space coordinate is exported. □

Theorem 6566.3 (Compact uniform modulus finite-fold handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformModulusFold}^{\uparrow}$ *packet must first read the compact finite-net row K, the continuous-map pointwise-modulus row G, the metric row M, the rational comparison row Q, and the finite fold ledger F. The handoff to L therefore realizes the finite* ProbeBundle *construction of Definition 9237.5 as NameCert data.*

Proof. Begin with the carrier $U = (K, G, M, Q, F, L, H, C, P, N)$. The only finite cover data is in K, and the only pointwise modulus data is in G.

The rational fold cannot be read without metric and rational support. The row M supplies the distance comparisons used by the compact-uniform triangle estimate, while Q supplies the positive-radius order comparisons used by the fold.

Now read F. It is the finite ledger that lowers all bundled local radii to one precision history, matching the construction cited in Definition 9237.5. Only after that finite ledger is visible can L export the tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ row. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformModulusFold}^{\uparrow}$ uniform-modulus read must expose Compact, ContinuousMap, Metric, Rat, finite ProbeBundle fold, UniformModulus, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: compact, continuousmap, metric, rat, uniformmodulus: $\mathsf{CompactUniformModulusFold}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries compact finite-net rows, continuous pointwise-modulus rows, metric triangle support, rational lower-bound fold data, UniformModulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusFold}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusFold}^{\uparrow}$ packet is generated from a Compact finite-net row, a ContinuousMap pointwise-modulus row, Metric distance rows, Rat positive-radius comparisons, a finite ProbeBundle fold ledger, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusFold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6566.1, Definition 6566.2, and Definition 6566.3 seed the finite compact-uniform modulus fold route over Compact, ContinuousMap, Metric, Rat, ProbeBundle fold, and UniformModulus rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusFold}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, arbitrary open-cover compactness, external modulus selection, host function equality, completed function spaces, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous pointwise-modulus exposure, metric-triangle compatibility, Rat lower-bound fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusFoldUp.

Chapter 6567

A Concrete Naming Certificate for $\mathsf{CompactClosedMap}^{\uparrow}$

Following Bourbaki and Bishop–Bridges constructive topology, the $\mathsf{CompactClosedMap}^{\uparrow}$ packet records the compact-to-Hausdorff closed-map consequence as finite-net image data rather than as an open-cover compactness theorem. It reads compact source rows from chapter 60, metric and Hausdorff separation rows from chapter 4230 and chapter 2086, continuous-map and uniform-modulus rows from chapter 130 and chapter 369, and Real tolerance rows from chapter 38. The constructive reading is a finite-net image schedule: every closed-image read is obtained by replaying displayed compact finite nets, continuity/modulus rows, Hausdorff separation rows, and closed-domain subpacket rows.

Definition 6567.1 (Compact closed-map carrier)**.** *A* $\mathsf{CompactClosedMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, H, D, I, T, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, carrying the finite-net and refinement schedules of the accepted* $\mathsf{Compact}^{\uparrow}$ *packet.*
2. *F is the continuous graph and modulus row: it records the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph samples and the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance responses used to transport source finite nets to image finite nets.*
3. *H is the Hausdorff separation row over the target metric surface, naming the finite separation witnesses that distinguish an image point from a point outside the transported image packet.*
4. *D is the closed-domain subpacket row, displaying the closed source subpacket whose image is being read.*
5. *I is the image row, recording the finite transported image samples, their Real-valued tolerance budgets, and the target-side membership classifier for the closed-image packet.*
6. *T is the componentwise* hsame *transport row for K, F, H, D, I.*
7. *C is the displayed* Cont *replay route for the same compact finite-net image schedule.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactClosedMap}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactClosedMap}^{\uparrow}}$ *row.*

The carrier exports no open-cover compactness principle, host topology, quotient equality of points or images, selected finite subcover, countable-choice selector, or ambient completeness row.

Theorem 6567.2 (Compact closed-map NameCert obligations)**.** *For an accepted* $\mathsf{CompactClosedMap}^{\uparrow}$ *carrier* $M = (K, F, H, D, I, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactClosedMap}^{\uparrow}}$ *surface consists exactly of carrier admission for compact finite-net data, continuous graph and uniform-modulus rows, Hausdorff separation rows, closed-domain subpacket data, finite image rows, componentwise transport, displayed replay, provenance, and local naming. Any consumer that asks for open-cover compactness, a host-topology closed-set theorem, quotient equality, selected subcovers, choice, or ambient completeness is outside the accepted packet.*

Proof. Project the carrier of Definition 6567.1. The analytic rows are precisely K, F, H, D, I, and the structural rows are T, C, P, N.

The compact row supplies finite-net data, the continuous and modulus row transports those data to image samples, the Hausdorff row separates target points by displayed metric witnesses, and the closed-domain row restricts the source packet before the image row is read.

Finally attach transport, replay, provenance, and local naming. Since the carrier inventory has no row for open covers, host topology, quotient equality, selected subcovers, countable choice, or ambient completeness, the NameCert obligations are exhausted by the displayed finite rows. □

Theorem 6567.3 (Compact closed-map finite-net image)**.** *Every image read exported by an accepted* $\mathsf{CompactClosedMap}^{\uparrow}$ *carrier factors through the finite route*

$$K, F, D \longmapsto I \longmapsto T, C, P, N.$$

The row K *supplies compact finite nets,* F *supplies continuous graph and* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance control,* D *supplies the closed-domain subpacket, and* I *records the transported finite image rows with* $\mathsf{Real}^{\uparrow}$ *tolerance budgets. The export is not an open-cover compactness argument, a selected subcover, a quotient image, or a host-topological closure operation.*

Proof. Start with the compact source row K. A finite image read cannot inspect the target image before the displayed compact finite-net schedule is available.

Next read F and D. The continuous graph and modulus data say how source finite-net samples are transported, while the closed-domain subpacket says which source rows are permitted to contribute to the image.

The row I then records the transported image samples and Real tolerance budgets. The structural rows T, C, P, N preserve and name the same finite route, so no open-cover proof, selected subcover, quotient image, or host closure operation can enter. □

Theorem 6567.4 (Compact closed-map Hausdorff separation)**.** *For an accepted* $\mathsf{CompactClosedMap}^{\uparrow}$ *carrier* $M = (K, F, H, D, I, T, C, P, N)$*, every target-side separation read used to certify closedness of the image must expose the Hausdorff row* H *after the compact finite-net, continuous-modulus, closed-domain, and image rows have been displayed. The separation is therefore a finite* $\mathsf{HausdorffMetric}^{\uparrow}$ *witness over the transported image packet, not host topological separation, quotient equality, excluded middle over target membership, or a choice-selected nearest point.*

Proof. Project the target-side rows. The image packet is displayed through I, and the only row carrying separation data is H.

Use the finite-net image route of Definition 6567.3. It exposes K, F, D, I before any downstream consumer may read target closedness.

Now read H as a Hausdorff-metric separation row over the already transported image packet. Transport, replay, provenance, and local naming keep the same displayed witnesses, so the read cannot be replaced by host topological separation, quotient equality, excluded middle, or a choice-selected nearest point. □

Theorem 6567.5 (Compact closed-map public export)**.** *Every public consumer of an accepted* $\mathsf{CompactClosedMap}^{\uparrow}$ *packet receives only the compact finite-net source row K, continuous graph and uniform-modulus row F, Hausdorff separation row H, closed-domain row D, finite image row I, and support rows T, C, P, N. The public route may be consumed by later compact-uniform, inverse-continuity, or Real-facing packets as a closed-image carrier, but it does not export open-cover compactness, a host topology, quotient equality, a selected finite subcover, choice, or ambient completeness.*

Proof. Begin with the NameCert obligations of Definition 6567.2. They list the full carrier surface and exclude all rows outside $K, F, H, D, I, T, C, P, N$.

Apply the finite-net image theorem and the Hausdorff separation theorem. Together they identify the public closed-image route as compact finite-net transport followed by target-side Hausdorff separation over the displayed image packet.

The downstream consumer therefore receives a reusable closed-image carrier, not a host theorem. Since all public data are projected from the same finite rows, no open-cover compactness, host topology, quotient equality, selected subcover, choice, or ambient completeness can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactClosedMap}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactClosedMap}^{\uparrow}$ packet is generated from compact finite-net rows, continuous graph and uniform-modulus rows, Hausdorff target separation rows, closed-domain subpacket rows, finite image rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactClosedMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope names the compact-to-Hausdorff closed-image carrier and its finite-net image, Hausdorff separation, public export, transport, replay, provenance, and local naming obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactClosedMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host topological closedness, quotient equality, selected finite subcovers, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net image exactness, Hausdorff separation, public export, transport, replay, provenance, and local naming for BEDC.Derived.CompactClosedMapUp.

Chapter 6568

A Concrete Naming Certificate for $\mathsf{RegularSequenceComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularSequenceComparison}^{\uparrow}$ packet records the finite comparison surface for two regular rational stream windows. It sits between the dyadic tolerance ledger of chapter 291, the finite observation route of chapter 126, the regular rational sequence packet of chapter 301, the displayed real-name seal of chapter 38, and the Cauchy naming surface of chapter 3413. The packet compares only displayed windows and tolerances; it is not a Dedekind cut, quotient real equality, selected limit, trichotomy principle, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 6568.1 (Regular sequence comparison carrier)**.** *A* $\mathsf{RegularSequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S_0, S_1, Q_0, Q_1, D, T, M, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the two regular rational streams being compared.*
2. Q_0, Q_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the two displayed windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ladder, displayed as finite dyadic stages*

   $$\delta_0 \geq \delta_1 \geq \cdots \geq \delta_k$$

   together with the endpoint and difference comparisons consumed at each inspected stage.
4. T *is the termwise comparison ledger pairing the visible* Q_0 *and* Q_1 *entries at the dyadic stages listed by* D*.*
5. M *is the modulus-window row recording which finite StreamName windows are required before a requested dyadic tolerance may be read.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after* $S_0, S_1, Q_0, Q_1, D, T, M$ *have been displayed.*
7. H *records componentwise* hsame *transport for both windows, both regular-rational readbacks, the dyadic tolerance ladder, the comparison ledger, the modulus-window row, the Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a comparison request to the two windows, their regular readbacks, the dyadic ladder, the comparison ledger, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, Q_0, Q_1, D, T, M, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient real equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6568.2 (Regular sequence comparison NameCert obligations)**.** *For every accepted* RegularSequenceComp *carrier $R = (S_0, S_1, Q_0, Q_1, D, T, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{StreamName}^{\uparrow}$ *windows, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ladder, the termwise comparison ledger, the modulus-window row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6568.1. The two windows S_0, S_1, the two regular-rational readbacks Q_0, Q_1, and the dyadic ladder D are the only source rows visible to the comparison.

Read the comparison through T and M. The ledger T pairs only entries already exposed by the two readback rows, while M records the finite windows required for the requested tolerance.

Finally transport and replay the same rows through H and C, and source them through P, N. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations and leave quotient equality, selected limits, countable choice, trichotomy, and Real completeness outside the packet. □

Theorem 6568.3 (Regular sequence comparison RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularSequenceComparison}^{\uparrow}$ *packet receives comparison data only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by $S_0, S_1, Q_0, Q_1, D, T, M, E, H, C, P, N$. The handoff therefore supplies a finite regular-sequence comparison window, not a quotient equality of real names, selected limit, Dedekind-cut comparison, trichotomy decision, countable-choice schedule, or ambient completeness principle.

Proof. Start with an accepted packet and project its two $\mathsf{StreamName}^{\uparrow}$ rows. The consumer cannot read a regular rational comparison until both finite windows S_0, S_1 have been displayed.

Next read Q_0, Q_1 and the dyadic tolerance ladder D. The $\mathsf{RegSeqRat}^{\uparrow}$ rows expose only finite regular-rational readbacks, and D records the finite tolerance stages at which those readbacks are compared.

The comparison ledger T and modulus-window row M then expose the requested finite comparison. The terminal row E is a displayed $\mathsf{Real}^{\uparrow}$ seal for this route only; it is not a completeness assertion.

Finally H, C, P, N transport, replay, source, and name the same packet. Thus any RegSeqRat or Real-facing consumer receives the finite comparison route and no hidden quotient equality, selected limit, Dedekind cut, trichotomy decision, countable-choice schedule, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularSequenceComparison}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readback rows, a DyadicRatCore tolerance ladder, a termwise comparison ledger, a modulus-window row, a RealUp seal, transport, replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RegularSequenceComparison}\uparrow}$, seed)

Scope closed: Definition 6568.1, Definition 6568.2, and Definition 6568.3 seed the finite regular-sequence comparison route through displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, and a RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularSequenceComparison}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind cuts, quotient real equality, selected limits, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic ladder exposure, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceComparisonUp.

Chapter 6569

A Concrete Naming Certificate for $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$

The $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ packet realizes the boundary-level hyperbolic observer-expansion route of chapter 9463 as finite Busemann, visual-boundary, metric, transport, provenance, and Real-seal rows. It sits after the Busemann boundary packet of chapter 3732, the visual-boundary readout of chapter 4181, the boundary-shadow route of chapter 3813, the observer-expansion packet of chapter 4167, and the observer-shadow metric of chapter 4154. The packet names a finite horoball-shadow comparison; it is not a global hyperbolic model, quotient ideal boundary, selected geodesic ray, physical observer, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 6569.1 (Hyperbolic horoball-shadow carrier)**.** *A* $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (B, V, M, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, recording the finite Busemann-level boundary read and the horoball threshold being inspected.*
2. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row, recording the visible boundary face used by the shadow read.*
3. *M is the hyperbolic metric comparison row used to attach the horoball threshold to the displayed boundary data.*
4. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row supplying the observer-facing expansion route.*
5. *S is the boundary-shadow budget row, tying the Busemann threshold, visual-boundary row, metric comparison, and observer-expansion row to one finite shadow window.*
6. *R is the refusal row excluding host compactifications, quotient boundary equality, selected rays, physical observer trajectories, and unlogged shadow budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed shadow budget.*
8. *H records componentwise* hsame *transport for Busemann, visual-boundary, metric, observer-expansion, shadow-budget, refusal, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through Busemann readback, visual-boundary exposure, metric comparison, observer-expansion routing, shadow-budget exposure, refusal, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBound}$ $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadow}^{\uparrow}}$ *row.*

The classifier accepts only packets whose Busemann row, visual-boundary row, metric comparison, observer-expansion row, shadow-budget row, refusal row, Real seal, transport, replay, provenance, and local naming remain displayed on the same finite route.

Lean4 (definition) `BEDC.Derived.HyperbolicHoroballShadowUp.HyperbolicHoroballShadowCarrier`

Theorem 6569.2 (Hyperbolic horoball-shadow NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHoroballShad}$ *carrier $Y = (B, V, M, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadow}^{\uparrow}}$ *surface consists exactly of Busemann-boundary admission, visual-boundary readback, hyperbolic metric comparison, observer-expansion route exposure, boundary-shadow budget exposure, refusal exactness, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.HyperbolicHoroballShadowUp.HyperbolicHoroballShadowNamecertObligations`

Proof. Project the carrier of Definition 6569.1. The Busemann row B, visual-boundary row V, metric row M, and observer-expansion row O are the only geometric source rows visible to the local certificate.

Read the shadow-budget row S and refusal row R. The budget row ties the Busemann threshold to the visible boundary and observer route, while R prevents the packet from being read as a host compactification, quotient boundary, selected ray, physical observer, or unlogged shadow source.

Finally expose E, H, C, P, N. The Real seal is attached to the displayed budget route, and the structural rows transport, replay, source, and name the same packet, so the listed rows exhaust the NameCert obligations. □

Theorem 6569.3 (Hyperbolic horoball-shadow boundary handoff)**.** *Every boundary-level consumer of an accepted* $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ *packet must first display*

$\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserver}$

through the rows $B, V, M, O, S, R, E, H, C, P, N$. A consumer missing the Busemann row cannot produce the boundary handoff, and a consumer missing the horoball-level or shadow-budget rows is not bijective to this packet.

Lean4 ✓ `BEDC.Derived.HyperbolicHoroballShadowUp.HyperbolicHoroballShadowBoundaryHandoff`

Proof. Begin with the accepted carrier. The Busemann row B supplies the horoball threshold and boundary-level distance-difference read, so the boundary handoff cannot be formed from visual data alone.

Next read V, M, O, S. These rows attach the visible boundary, metric comparison, observer-expansion route, and shadow-budget window to the same finite packet.

The refusal row R blocks host compactifications, quotient boundary equality, selected rays, physical observer trajectories, and unlogged shadow budgets. The Real seal E is therefore only the displayed budget seal for this route.

Finally use H, C, P, N to transport, replay, source, and name the same rows. Any boundary-level consumer that omits the Busemann row, horoball-level threshold, or shadow-budget row exits the accepted $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ packet. □

Falsifiable prediction: Any accepted horoball-shadow packet missing the Busemann row cannot produce the boundary handoff of Definition 6569.3.

Independence witness: hyperbolic-busemann-boundary, hyperbolic-visual-boundary, boundary-shadow-expansion-cocycle, hyperbolic-observer-expansion: none is bijective to $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ because the Busemann-boundary packet lacks the full shadow budget, the visual-boundary packet lacks the horoball-level row, the boundary-shadow expansion cocycle lacks the Busemann threshold, and the observer-expansion packet lacks the joint horoball-level and shadow-budget rows.

The Lean TasteGate route must provide explicit $\mathsf{Nontrivial}\ \mathsf{HyperbolicHoroballShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicHoroballShadow}^{\uparrow}$ instances in `lean4/BEDC/Derived/HyperbolicHoroballShadowUp/Taste` Those instances must preserve $B, V, M, O, S, R, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ packet is generated from Busemann-boundary rows, visual-boundary rows, metric comparison rows, observer-expansion rows, a boundary-shadow budget, refusal exactness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6569.1, Definition 6569.2, and Definition 6569.3 seed the finite horoball-shadow route through Busemann, visual-boundary, metric, observer-expansion, shadow-budget, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global hyperbolic model, quotient ideal boundary, selected geodesic ray, physical observer trajectory, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Busemann row exposure, visual-boundary readback, shadow-budget exactness, boundary handoff, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicHoroballShadowUp.

Chapter 6570

A Concrete Naming Certificate for $\mathsf{BoundedVariationSequence}^{\uparrow}$

Following Bishop and Bridges, a sequence of bounded variation is a regular real sequence whose finite variation ledgers are uniformly controlled by displayed rational data. The $\mathsf{BoundedVariationSequence}^{\uparrow}$ packet records this classical constructive object on the L10 Real and Completion route. It sits after the stream-name surface of chapter 126, the regular rational sequence interface of chapter 301, the dyadic rational core of chapter 291, the terminal real-name boundary of chapter 38, and the Cauchy route of chapter 3413. The packet names finite sequence-variation control without importing host sequence equality, selected limits, countable choice, completed sequence spaces, or full real completeness.

Definition 6570.1 (BoundedVariationSequence carrier)**.** *A* $\mathsf{BoundedVariationSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S, Q, D, R, C, H, K, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the observation stream and finite prefix windows.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reading each displayed sequence value by regular rational names.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row for adjacent dyadic difference observations and finite variation sums.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the accepted value names and variation-budget name.*
5. *C is the* $\mathsf{Cauchy}^{\uparrow}$ *route row exposed only after the finite variation ledger supplies a tail-control modulus.*
6. *H stores componentwise* hsame *transport for stream windows, regular rational reads, dyadic difference ledgers, Real seals, Cauchy handoff, replay, provenance, and local naming.*
7. *K stores* Cont *replay from prefix windows to adjacent differences, from finite variation sums to tail budgets, and from tail budgets to the Cauchy consumer route.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationSequence}^{\uparrow}}$ *row.*

The classifier compares packets only through $S, Q, D, R, C, H, K, P, N$. It has no coordinate for host sequence equality, a selected limiting real, quotient sequence spaces, countable choice, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6570.2 (BoundedVariationSequence variation ledger)**.** *For every accepted* $\mathsf{BoundedVariationSequence}^{\uparrow}$ *carrier $V = (S, Q, D, R, C, H, K, P, N)$ and every displayed finite prefix window, the public variation read factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row D accounts for adjacent dyadic differences and finite variation sums, while R seals only the displayed regular rational values and their variation-budget read. No consumer receives a supremum over all partitions, host sequence equality, selected limit, or ambient completed sequence space from this ledger.

Proof. Project the carrier of Definition 6570.1. A variation request first chooses a finite prefix window from S and then reads the corresponding value names through the regular rational row Q.

The dyadic row D is the only place where adjacent differences and finite variation sums are recorded. Its data are finite: each read is tied to the displayed prefix and to the regular names already exposed by Q.

The Real row R seals those displayed value names and the displayed variation-budget name. Because the carrier has no coordinate for a supremum over all prefixes, host sequence equality, quotient sequence space, selected limit, or full real completeness, the public variation ledger is exactly the finite route through S, Q, D, R. □

Theorem 6570.3 (BoundedVariationSequence Cauchy modulus route)**.** *Let $V = (S, Q, D, R, C, H, K, P, N)$ be an accepted* $\mathsf{BoundedVariationSequence}^{\uparrow}$ *carrier. Any downstream* $\mathsf{Cauchy}^{\uparrow}$ *or completion consumer that cites a modulus from V must read the finite variation ledger S, Q, D, R first and then consume the displayed handoff row C with the same structural rows H, K, P, N. The route exports a tail-control obligation, not a selected limit or quotient equality of sequences.*

Proof. Start with the variation ledger of Definition 6570.2. It confines every finite variation read to stream windows, regular rational names, dyadic adjacent-difference observations, and the Real seal.

The Cauchy row C is downstream of that ledger. A consumer may read C only after the finite variation sums and tail-budget comparisons have been replayed through K and transported through H.

The rows P and N source and name the same route for downstream completion-facing consumers. Since no coordinate of the carrier is a selected limit, quotient sequence equality, completed sequence-space object, countable-choice principle, or full real-completeness row, the Cauchy handoff is a modulus route over displayed finite data. □

Falsifiable prediction: Every $\mathsf{BoundedVariationSequence}^{\uparrow}$ read from an accepted packet factors through the stream window, regular rational sequence readback, dyadic adjacent-difference ledger, Real seal, Cauchy handoff, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any completion consumer sees the route.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy: $\mathsf{BoundedVariationSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries prefix windows, regular rational value names, adjacent dyadic difference ledgers, finite variation sums, a Real seal, a Cauchy handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedVariationSequence}^{\uparrow}$ packet is generated from StreamName prefix windows, RegSeqRat value names, DyadicRatCore adjacent-difference and finite variation ledgers, a RealUp seal, a Cauchy handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6570.1, Definition 6570.2, and Definition 6570.3 seed the finite bounded-variation sequence interface over StreamName, RegSeqRat, DyadicRatCore, Real, and Cauchy routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a selected sequence limit, host sequence equality, quotient sequence space, countable choice, full Real completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, variation-ledger exactness, Cauchy modulus handoff, classifier transport, and consumer nonescape for BEDC.Derived.BoundedVariationSequenceUp.

Chapter 6571

A Concrete Naming Certificate for $\mathsf{DyadicEpsilonNet}^{\uparrow}$

Bishop–Bridges-style finite-net constructive analysis treats compact interval and compact metric work by displaying the actual finite schedule of centers and radii inspected at a requested precision. The $\mathsf{DyadicEpsilonNet}^{\uparrow}$ packet records that schedule as a BEDC NameCert object on the L10 route: its centers and tolerances read the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint ledger of chapter 291, its mesh cells use the $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ enclosure route of chapter 1645, its requests are exposed through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, its finite readback reaches $\mathsf{RegSeqRat}^{\uparrow}$ through chapter 301, its terminal values are sealed through $\mathsf{Real}^{\uparrow}$ in chapter 38, and its coverage claims are consumed against the $\mathsf{Met}^{\uparrow}$ source of chapter 57. The packet is the explicit finite-net schedule used by interval compactness and total-boundedness consumers; it is not an open-cover compactness theorem, a selected limiting subnet, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6571.1 (Dyadic epsilon-net carrier)**.** *A* $\mathsf{DyadicEpsilonNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, K, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row whose point histories and distance reads are the only coverage domain inspected by the packet.*
2. *ε is a positive tolerance request displayed as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule recording which precision requests, source probes, and center indices are inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the requested radius, mesh slack, and distance estimates along W.*
5. *I is the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *mesh-row package that stores outward endpoint enclosures and interval-cell refinements for the displayed schedule.*
6. *C_ε is the finite list of center rows admitted at the requested tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the window, dyadic ledger, interval arithmetic, and center data.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after $X, \varepsilon, W, D, I, C_\varepsilon, R$ have been displayed.*

9. *H records componentwise* hsame *transport for the metric source, tolerance request, windows, dyadic ledger, interval arithmetic, center list, readback, real seal, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay from a finite-net request through $X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicEpsilonNet}^\uparrow$, $\mathsf{Met}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{DyadicIntervalArithmetic}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicEpsilonNet}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, K, P, N$. It has no coordinate for an arbitrary open cover, a selected infinite subnet, quotient point equality, host real equality, countable choice, unlisted centers, or ambient completeness.

Theorem 6571.2 (Dyadic epsilon-net NameCert obligations)**.** *For every accepted* $\mathsf{DyadicEpsilonNet}^\uparrow$ *carrier*

$$E = (X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicEpsilonNet}^\uparrow}$ *surface consists exactly of carrier admission for the* $\mathsf{Met}^\uparrow$ *source, positive dyadic tolerance exposure, finite-window exposure, dyadic tolerance exactness, dyadic interval-arithmetic mesh exposure, finite center-list admission,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{Real}^\uparrow$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No obligation may be discharged by open-cover compactness, a selected infinite subnet, quotient point equality, host real equality, countable choice, hidden centers, or ambient real completeness.

Proof. Project the carrier of Definition 6571.1. The analytic rows are exactly the metric source X, the tolerance request ε, the finite schedule W, the dyadic ledger D, the interval mesh I, the center list C_ε, the readback R, and the terminal seal S.

Read the numerical rows in dependency order. The tolerance request and comparison budget are displayed by $\mathsf{DyadicRatCore}^\uparrow$; the mesh cells are then routed through $\mathsf{DyadicIntervalArithmetic}^\uparrow$; and only those displayed cells and center rows feed the $\mathsf{RegSeqRat}^\uparrow$ readback.

The structural rows preserve the same schedule. Componentwise transport through H and replay through K keep the source, tolerance, window, mesh, centers, readback, and seal in the same order, while P and N name exactly the sibling packet and local certificate rows.

The excluded principles are not carrier coordinates. Therefore no open-cover compactness argument, selected infinite subnet, quotient point equality, host real equality, countable-choice selector, hidden center family, or ambient real-completeness principle can enter the local NameCert surface. □

Theorem 6571.3 (Dyadic epsilon-net finite L10 route)**.** *Let $E = (X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, K, P, N)$ be an accepted* $\mathsf{DyadicEpsilonNet}^\uparrow$ *carrier. Every compactness-facing, total-boundedness-facing, or completion-facing consumer of the finite-net schedule reads the route*

$$\mathsf{Met}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{DyadicIntervalArithmetic}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

through the displayed rows $X, \varepsilon, W, D, I, C_\varepsilon, R, S, H, K, P, N$. The consumer receives a finite dyadic center schedule at the requested tolerance, not an open-cover subcover, an infinite selected net, quotient real equality, countable choice, or ambient completeness.

Proof. Begin with a consumer read of E. The metric source row X fixes the inspected domain, and the positive dyadic request ε fixes the tolerance that the finite schedule must answer.

The route then exposes only finite data. The window row W lists the inspected precision requests and center indices, D compares their dyadic tolerances, and I records the interval-cell arithmetic needed by the displayed mesh.

The readback and seal are downstream. The row R is produced from the same W, D, I, C_ε data, and the terminal row S is reached only after that readback has been displayed.

Finally H, K, P, N preserve and name this route. Since the carrier has no open-cover, infinite-net, quotient, host-equality, choice, or ambient-completeness coordinate, every accepted consumer receives exactly the displayed finite L10 route. □

Falsifiable prediction: Every accepted $\mathsf{DyadicEpsilonNet}^\uparrow$ read factors through $\mathsf{Met}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{DyadicIntervalArithmetic}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicEpsilonNet}^\uparrow}$.

Constructive story: A $\mathsf{DyadicEpsilonNet}^\uparrow$ packet is generated from a MetricUp source row, a positive DyadicRatCoreUp tolerance request, StreamNameUp finite windows, DyadicRatCoreUp comparison ledgers, DyadicIntervalArithmeticUp mesh cells, a finite center list, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicEpsilonNet}^\uparrow}$, seed)

Scope closed: Definition 6571.1, Definition 6571.2, and Definition 6571.3 seed the explicit finite epsilon-net schedule over MetricUp, DyadicRatCoreUp, DyadicIntervalArithmeticUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicEpsilonNet}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, open-cover compactness, selected infinite nets, quotient point equality, host real equality, countable choice, hidden centers, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite center-list admission, dyadic tolerance exactness, interval-mesh exactness, readback exactness, RealUp seal nonescape, classifier transport, and TasteGate field faithfulness for BEDC.Derived.DyadicEpsilonNetUp.

Chapter 6572

A Concrete Naming Certificate for $\mathsf{AlexandrovTopology}^{\uparrow}$

Following Alexandrov and the order-topology tradition, the $\mathsf{AlexandrovTopology}^{\uparrow}$ packet records the topology generated by a finite preorder row through upward-closed observation windows. It sits after the preorder carrier of chapter 52, the indexed-open surface of chapter 88, the open-neighbourhood implication route of chapter 5864, and the domain-facing comparison surface of chapter 5950. The packet names only finite upper-set generation, finite union and intersection replay, specialization transport, and Scott-facing comparison data; it does not import a host powerset topology, quotient-open equality, arbitrary joins, selected directed suprema, excluded middle over opens, or Lean verification.

Definition 6572.1 (Alexandrov topology carrier)**.** *An* $\mathsf{AlexandrovTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, U, O, J, M, S, Q, H, C, G, N)$$

with the following displayed rows:

1. *P is the consumed* $\mathsf{Preorder}^{\uparrow}$ *row, listing the finite point names, reflexivity witnesses, transitivity witnesses, and comparison requests used by the generation.*
2. *U is the upper-set row, recording the finite upward-closure checks: whenever $x \leq_P y$ and x is accepted by the window, the same window accepts y.*
3. *O is the open-window row, listing the finitely inspected Alexandrov opens and their membership probes.*
4. *J records finite union replay for the displayed open windows, using only the listed membership probes.*
5. *M records finite intersection replay for the displayed open windows, again restricted to the same finite preorder-generated rows.*
6. *S is the specialization transport row, comparing the specialization preorder induced by the displayed open windows with the source preorder row P.*
7. *Q is the Scott-comparison row, recording the finite handoff to* $\mathsf{ScottTopology}^{\uparrow}$ *consumers when a Scott-facing upper-set window is displayed.*

8. *H records componentwise* hsame *transport for preorder rows, upper-set rows, open windows, union and intersection replay, specialization transport, Scott comparison, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from preorder requests through U, O, J, M, S, Q and then to the public topology-facing read.*
10. *G is the* Pkg *provenance over* $\mathsf{AlexandrovTopology}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SpecializationOrder}^{\uparrow}$, $\mathsf{ScottTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexandrovTopology}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $P, U, O, J, M, S, Q, H, C, G, N$. It has no coordinate for a host powerset of opens, arbitrary infinitary closure, quotient-open equality, selected basis completion, or directed-supremum theorem.

Theorem 6572.2 (AlexandrovTopology NameCert obligations)**.** *For every accepted* $\mathsf{AlexandrovTopology}^{\uparrow}$ *carrier $A = (P, U, O, J, M, S, Q, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexandrovTopology}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed preorder, upper-set, open-window, finite union, finite intersection, specialization, Scott-comparison, transport, replay, provenance, and local naming rows;*
2. *upper-set exactness, requiring each accepted open-window membership read to be upward closed along the preorder row P through the displayed row U;*
3. *finite topology replay, requiring J and M to certify only the displayed finite union and finite intersection windows;*
4. *specialization compatibility, requiring the specialization preorder read from O, J, M to transport through S back to the source preorder row P;*
5. *Scott handoff discipline, requiring Q to expose only the displayed upper-set comparison rows before any* $\mathsf{ScottTopology}^{\uparrow}$ *consumer reads the packet;*
6. *componentwise transport, displayed continuation replay, package provenance, and local naming through H, C, G, N.*

These obligations certify the finite preorder-generated Alexandrov topology surface. They do not close arbitrary open lattices, powerset extensionality, directed-supremum completeness, quotient-open equality, selected bases, excluded middle over opens, or Lean verification.

Proof. Project the carrier of Definition 6572.1. The readable rows are exactly $P, U, O, J, M, S, Q, H, C, G, N$, so every topology-facing consumer starts from the displayed preorder and finite open-window data.

Upper-set exactness is read from U together with the comparison witnesses in P. The finite topology operations are then replayed by J and M, which inspect only the displayed open windows and therefore cannot introduce an ambient powerset of opens.

The specialization route is handled by S. It compares open-neighbourhood implication with the source preorder row already visible in P, so the packet records a transport of finite rows rather than a quotient equality of points.

The Scott-facing row Q is downstream from the same upper-set and open-window rows. Thus any Scott comparison receives a finite Alexandrov handoff and not a directed-supremum theorem.

Finally, H, C, G, N transport, replay, source, and name this finite route. Since no other coordinate is present, the excluded host-topological and infinitary principles cannot be exported. □

Theorem 6572.3 (AlexandrovTopology preorder specialization route)**.** *Let $A = (P, U, O, J, M, S, Q, H, C, G, N)$ be an accepted* $\mathsf{AlexandrovTopology}^{\uparrow}$ *carrier. Every specialization-order read exported by A factors through the finite route*

$$P \longmapsto U \longmapsto O, J, M \longmapsto S.$$

The exported read may compare preorder rows with open-neighbourhood implication rows, but it cannot bypass the displayed Alexandrov windows to import quotient point equality, host separation, or an undeclared topology theorem.

Proof. Begin with a specialization-facing request. The only order source visible to the request is the preorder row P, and the only open data available for neighbourhood implication is the displayed upper-set and open-window route U, O, J, M.

The transport row S reads those same rows before it exposes a specialization comparison. Hence the exported relation is obtained by finite preorder-generated windows rather than by host point equality or a separation axiom.

The structural rows H, C, G, N preserve and name this route. They do not add any coordinate for quotient points, powerset opens, or an ambient topological theorem, so the specialization read remains inside the displayed carrier. □

Theorem 6572.4 (AlexandrovTopology Scott handoff)**.** *Let $A = (P, U, O, J, M, S, Q, H, C, G, N)$ be an accepted* $\mathsf{AlexandrovTopology}^{\uparrow}$ *carrier. Every* $\mathsf{ScottTopology}^{\uparrow}$ *comparison exported by A must first read the preorder row P, the upper-set ledger U, the finite open-window row O, and the union and intersection replay rows J, M. The handoff row Q therefore presents a finite upper-set comparison surface and does not claim directed-supremum inaccessibility, dcpo completeness, or equality with a host Scott topology.*

Proof. Project the carrier. The Scott-facing coordinate is Q, and it appears only after the preorder, upper-set, open-window, and finite topology replay rows.

Use the Scott handoff discipline in Definition 6572.2. It requires Q to expose the displayed upper-set comparison rows before a $\mathsf{ScottTopology}^{\uparrow}$ consumer can read the packet.

Because directed suprema, dcpo completeness, and host Scott-open equality are absent from $P, U, O, J, M, S, Q, H, C, G, N$, the handoff remains a finite comparison route. It can feed domain-theoretic consumers without asserting the missing infinitary closure. □

Falsifiable prediction: Every $\mathsf{AlexandrovTopology}^{\uparrow}$ topology-facing read factors through a preorder row, upper-set checks, finite open windows, finite union replay, finite intersection replay, specialization transport, Scott comparison, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: preorder, topology, specialization-order, scott-topology: $\mathsf{AlexandrovTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries preorder generation, upward-closure windows, topology replay, specialization transport, Scott comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandrovTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AlexandrovTopology}^{\uparrow}$ packet is generated from a finite preorder row, upward-closed observation windows, finite union and intersection replay, specialization transport, Scott-facing comparison, componentwise transport, continuation replay, package provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandrovTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6572.1, Definition 6572.2, Definition 6572.3, and Definition 6572.4 seed the finite preorder-generated topology surface for specialization and Scott-comparison consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandrovTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open lattices, powerset extensionality, dcpo completeness, directed-supremum inaccessibility, quotient-open equality, selected bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-set exactness, finite union and intersection replay, specialization compatibility, Scott handoff discipline, transport stability, replay, provenance, and local naming for BEDC.Derived.AlexandrovTopologyUp.

Chapter 6573

A Concrete Naming Certificate for $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$ packet records the finite route by which three regular Cauchy rational sources present a triangle-inequality estimate before a Real metric consumer reads the terminal seal. It sits after the three-leg bound of chapter 466, the metric readback surface of chapter 1654, the real metric route of chapter 5857, the synchronized windows of chapter 126, the regular rational source rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite handoff surface for Real and Completion consumers; it is not quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6573.1 (Regular Cauchy triangle inequality carrier)**.** *A* $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, Z, L_{xy}, L_{yz}, W, Q, M, R, S, H, C, P, N)$$

with the following displayed coordinates:

1. X, Y, Z *are the three* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose pairwise metric comparison is requested.*
2. L_{xy} *and* L_{yz} *are the two* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *leg packets that compare* X *to* Y *and* Y *to* Z *through finite regular Cauchy estimates.*
3. W *is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect the three sources and the two leg packets at a common request.*
4. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the two leg budgets, their displayed sum, and the output budget read by the Real metric consumer.*
5. M *is the* $\mathsf{RegularCauchyMetric}^{\uparrow}$ *handoff row that turns the synchronized regular Cauchy comparisons into the metric-facing surface.*
6. R *is the* $\mathsf{RealTriangleInequality}^{\uparrow}$ *handoff row that exposes the terminal three-leg Real metric route.*
7. S *stores the* $\mathsf{Real}^{\uparrow}$ *seal rows for the three endpoints and the final distance comparison.*
8. H *stores componentwise* hsame *transport for sources, leg packets, windows, dyadic budgets, metric handoff, Real handoff, seal rows, provenance, and the local name.*
9. C *stores displayed* Cont *replay from a Real metric triangle request through* $X, Y, Z, L_{xy}, L_{yz}, W, Q, M, R, S, H, P,$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{RegularCauchyMetric}^{\uparrow}$, $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting every displayed row to the common source, window, tolerance, metric, and Real seal surface. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness.

Theorem 6573.2 (Regular Cauchy triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$ *carrier* $T = (X, Y, Z, L_{xy}, L_{yz}, W, Q, M, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed three-source packet T;*
2. *source admission for the three* $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y, Z;*
3. *leg admission for the two* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *packets L_{xy} and L_{yz};*
4. *synchronized* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W;*
5. *dyadic budget addition and output-budget exactness through Q;*
6. $\mathsf{RegularCauchyMetric}^{\uparrow}$ *handoff through M;*
7. $\mathsf{RealTriangleInequality}^{\uparrow}$ *handoff through R;*
8. *terminal* $\mathsf{Real}^{\uparrow}$ *seal reads through S;*
9. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N.*

No obligation exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6573.1. Its displayed rows are exactly the three regular Cauchy sources, the two triangle-bound legs, synchronized stream windows, dyadic tolerance ledger, metric handoff, Real triangle handoff, Real seal rows, transports, replay, provenance, and local naming row.

The NameCert obligations are therefore the admission and replay checks for those coordinates. Source admission reads X, Y, Z; leg admission reads L_{xy} and L_{yz}; window inspection reads W; tolerance exactness reads Q; the two handoff obligations read M and R; and the terminal seal reads S.

The rows H, C, P, N account for componentwise transport, consumer replay, package provenance, and local naming. Since the carrier contains no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the displayed list exhausts the local certificate surface. □

Theorem 6573.3 (Regular Cauchy triangle inequality Real metric route)**.** *Every Real metric triangle consumer of an accepted* $\mathsf{RegularCauchyTriangleInequality}^{\uparrow}$ *carrier factors through the displayed finite route*

$$(X, Y, Z) \longrightarrow (L_{xy}, L_{yz}, W, Q) \longrightarrow M \longrightarrow R \longrightarrow S.$$

The consumer first reads the three $\mathsf{RegSeqRat}^{\uparrow}$ *sources, the two* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *legs, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger; it then reads*

the RegularCauchyMetric$^{\uparrow}$ *handoff, the* RealTriangleInequality$^{\uparrow}$ *handoff, and the terminal* Real$^{\uparrow}$ *seal rows. No Real metric triangle route out of the packet can bypass this finite carrier through quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and project its rows. The only source coordinates are X, Y, Z, the only leg coordinates are L_{xy}, L_{yz}, and the only shared inspection and budget coordinates are W, Q.

The continuation row C, transported by H, replays a Real metric triangle request through those rows before it can reach the metric-facing handoff M. Thus the regular Cauchy metric comparison is already tied to the two displayed legs and their dyadic budget ledger.

The Real-facing part of the route then reads R and S. Row R is the displayed RealTriangleInequality$^{\uparrow}$ handoff, while S stores the terminal Real$^{\uparrow}$ seal rows for the three endpoints and the final comparison.

Because P and N only record provenance and local naming, and because the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, every Real metric triangle consumer factors through the finite route displayed above. □

Closure status for NameCert$_{\text{RegularCauchyTriangleInequality}^{\uparrow}}$.

Constructive story: A RegularCauchyTriangleInequality$^{\uparrow}$ packet is generated from three RegSeqRat$^{\uparrow}$ source rows, two RegularCauchyTriangleBound$^{\uparrow}$ leg packets, synchronized StreamName$^{\uparrow}$ windows, a DyadicRatCore$^{\uparrow}$ tolerance ledger, RegularCauchyMetric$^{\uparrow}$ and RealTriangleInequality$^{\uparrow}$ handoff rows, terminal Real$^{\uparrow}$ seal rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTriangleInequality}^{\uparrow}}$, seed)

Scope closed: Definition 6573.1, Definition 6573.2, and Definition 6573.3 seed the finite triangle-inequality route over regular Cauchy sources, triangle-bound legs, synchronized stream windows, dyadic budgets, metric handoff, Real handoff, Real seals, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTriangleInequality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, leg admission, synchronized window inspection, dyadic budget exactness, RegularCauchyMetric handoff, RealTriangleInequality handoff, RealUp seal reads, and non-escape for BEDC.Derived.RegularCauchyTriangleInequalityUp.

Chapter 6574

A Concrete Naming Certificate for $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$

Following Bishop and Bridges on constructive finite suprema of regular Cauchy real names, the $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ packet records a finite-family maximum route as displayed comparison-tree data. It sits after the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, the binary maximum packet of chapter 1833, the finite product packet of chapter 2009, and the terminal real-name seal of chapter 38. The certificate gives compact finite-net, Dini, and uniform-modulus consumers a named finite maximum object without importing trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6574.1 (Regular Cauchy finite maximum carrier)**.** *A* $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, W, D, T, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite unary index row, listing exactly the component slots opened by the maximum request.*
2. *S is the component* $\mathsf{RegSeqRat}^{\uparrow}$ *source family, one row for each slot of I.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for the component reads, intermediate pairwise maximum reads, and final maximum read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, storing the dyadic observations and tolerance cells used at each finite branch.*
5. *T is the finite comparison-tree row. Each internal node cites a displayed* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *route over two previously displayed child rows.*
6. *B records branch exactness: every leaf of T is a source row from S, every internal branch consumes two earlier nodes, and the root is the only final maximum output.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the root of the comparison tree, obtained only from S, W, D, T, B.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the root readback R has been displayed.*

9. *H records componentwise* hsame *transport for the index row, sources, windows, dyadic comparison ledger, comparison tree, branch exactness row, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from any finite maximum request through the same rows.*
11. *P records* Pkg *provenance over* $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyMaximum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, T, B, R, E, H, C, P, N$. It has no coordinate for a host finite supremum, trichotomy, quotient real equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or standard bridge.*

Theorem 6574.2 (Regular Cauchy finite maximum NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ *carrier*

$$F = (I, S, W, D, T, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, finite comparison-tree exactness through T and B, coordinate readback stability through S, W, D, R,* $\mathsf{Real}^{\uparrow}$*-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 6574.1. Carrier admission is exactly admission of the rows $I, S, W, D, T, B, R, E, H, C, P, N$.

Comparison-tree exactness is read from T and B. The index row I names the leaves, the source family S supplies those leaves, and each internal node of T cites a binary $\mathsf{RegularCauchyMaximum}^{\uparrow}$ route over two earlier displayed rows. The branch row B records that the root is the only final output.

Coordinate readback stability is downstream of the same tree. The synchronized window family W and dyadic ledger D are the only rows from which component and intermediate observations may be read, so the root readback R cannot depend on an unstated coordinate or a selected representative.

The seal row E is reached only after R has been exposed. The transport, replay, provenance, and naming rows H, C, P, N preserve and cite this displayed route; none adds a host finite supremum, trichotomy, quotient real equality, selected limit, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge credential. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 6574.3 (Regular Cauchy finite maximum comparison-tree route)**.** *Every accepted* RegularCauchyFiniteMa *read factors through the finite route*

$$I, S \longmapsto W, D \longmapsto T, B \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, the finite-family maximum is obtained by iterated displayed $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *rows, not by host trichotomy, quotient real equality, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite index and source rows. The request first names the slots in I and reads the matching component $\mathsf{RegSeqRat}^{\uparrow}$ rows in S.

The observation data is then synchronized by W and measured by D. These rows provide the finite windows and dyadic tolerances used by every binary comparison in the displayed tree.

The comparison tree T and branch row B order the binary maximum calls. Each internal node is a $\mathsf{RegularCauchyMaximum}^{\uparrow}$ route over already displayed child rows, and the root is the only row available for the final readback R.

Finally E, H, C, P, N seal, transport, replay, source, and name the same root route. Since no excluded host or quotient object is a coordinate of the carrier, every accepted read factors through the displayed finite tree. □

Theorem 6574.4 (Regular Cauchy finite maximum coordinate stability)**.** *If two accepted* $\mathsf{RegularCauchyFiniteMaxi}$ *carriers have the same finite index row, component* $\mathsf{RegSeqRat}^{\uparrow}$ *classifiers, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, comparison tree, and branch exactness row, then their root* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal are the same displayed route. A coordinate-facing consumer cannot separate the packets except through one of those displayed rows.*

Proof. Project the two carriers through Definition 6574.1. The classifier has no coordinate outside the index, source, window, dyadic ledger, comparison-tree, branch, readback, seal, transport, replay, provenance, and local naming rows.

Use the comparison-tree route theorem Definition 6574.3. Once I, S, W, D, T, B agree, the same iterated $\mathsf{RegularCauchyMaximum}^{\uparrow}$ nodes are read in the same order, so the root readback row R is forced to be the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ route.

The terminal seal E is downstream of R, and H, C, P, N only transport, replay, source, and name that route. Therefore a coordinate-facing consumer has no additional row from which to separate the packets. □

Theorem 6574.5 (Regular Cauchy finite maximum Real seal non-escape)**.** *For every accepted* $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row exposes only the displayed root readback of the finite comparison tree. It exports no host finite supremum, trichotomy, quotient real equality, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean verification, or standard bridge.*

Proof. Inspect the carrier boundary. The seal row E appears only after the index, source, window, dyadic comparison, comparison-tree, branch exactness, and root readback rows have been displayed.

The structural rows H, C, P, N preserve the same route. Transport and replay can move the displayed packet, provenance can cite its dependencies, and local naming can name it, but none supplies an independent finite supremum or quotient-equality coordinate.

Since every accepted read factors through Definition 6574.3, the $\mathsf{Real}^{\uparrow}$ seal cannot escape the comparison tree or import any excluded completion principle. □

Falsifiable prediction: Every $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ read factors through a finite unary index row, component RegSeqRat sources, synchronized StreamName windows, DyadicRatCore comparison cells, iterated RegularCauchyMaximum rows, a root RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-maximum, regular-cauchy-finite-product, real: $\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite index row, component source family, synchronized observation windows, dyadic comparison ledger, comparison-tree exactness, root readback, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6574.1, Definition 6574.2, Definition 6574.3, Definition 6574.4, and Definition 6574.5 seed the finite maximum route from displayed regular Cauchy sources through comparison-tree rows, root readback, and RealUp sealing.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyFiniteMaximum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host finite suprema, trichotomy, quotient real equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-tree exactness, coordinate readback stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFiniteMaximumUp.

Constructive story: A RegularCauchyFiniteMaximum$^{\uparrow}$ packet is generated from a finite unary index row, component RegSeqRat source rows, synchronized StreamName windows, DyadicRatCore comparison cells, a comparison tree of RegularCauchyMaximum rows, branch exactness, root RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6575

A Concrete Naming Certificate for $\mathsf{FermatStationaryPoint}^{\uparrow}$

Following Fermat and Bishop–Bridges constructive calculus, the $\mathsf{FermatStationaryPoint}^{\uparrow}$ packet records the finite route by which a local extremum witness and a derivative row at an interior point hand off to the stationary-point conclusion. It sits after the derivative quotient interface of chapter 122, the terminal real-name seal of chapter 38, the dyadic tolerance ledger of chapter 291, and the constructive interval-continuity route of chapter 3828. The packet names a derivative-neighbour of the Real route; it is not an ambient Real-completeness principle, not an open-cover compactness theorem, not a selected minimizing point, and not quotient equality of functions.

Definition 6575.1 (Fermat stationary-point carrier)**.** *A* $\mathsf{FermatStationaryPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, E, D, Z, H, C, P, N)$$

with the following displayed coordinates:

1. *I is the interval source row, carrying the two-sided interior-window data around the inspected point.*
2. *F is the function graph row, exposing only the displayed values and finite observation windows consumed by the derivative row.*
3. *E is the local extremum window row, recording that every admitted left and right comparison in the displayed neighbourhood is routed through the same interval and graph rows.*
4. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row at the inspected point, including the finite difference-quotient ledger and nonzero-step boundary.*
5. *Z is the zero-derivative handoff row, recording that the stationary conclusion reads the derivative row as a zero row after the two extremum-side comparison ledgers have been displayed.*
6. *H stores componentwise* hsame *transport for the interval, graph, extremum window, derivative row, zero handoff, replay, provenance, and local name.*
7. *C stores displayed* Cont *replay from a stationary-point request through I, F, E, D, Z, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FermatStationaryPoint}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DarbouxContinuity}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FermatStationaryPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting all displayed rows to the same interval source, graph window, local extremum ledger, derivative quotient row, and zero-derivative handoff. The carrier contains no coordinate for a selected global optimum, trichotomy, host function equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6575.2 (Fermat stationary-point NameCert obligations)**.** *For an accepted* $\mathsf{FermatStationaryPoint}^{\uparrow}$ *carrier* $S = (I, F, E, D, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FermatStationaryPoint}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed stationary-point packet* S*;*
2. *interval-source exposure through* I*;*
3. *function-graph exposure through* F*;*
4. *local extremum window exactness through* E*;*
5. *derivative-row admission through* D*;*
6. *zero-derivative handoff through* Z*;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*.*

No obligation exports a selected global optimum, trichotomy, host function equality, open-cover compactness, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6575.1. Its displayed rows are exactly the interval source, function graph, local extremum window, derivative row, zero-derivative handoff, transport, replay, provenance, and local naming row.

The NameCert obligations are therefore the admission and exactness checks for those coordinates. The interval source reads I; the graph read opens F; the local extremum comparisons read E; the derivative quotient data read D; and the stationary handoff reads Z.

The structural rows H, C, P, N preserve this same finite packet under classifier transport and consumer replay. Since the carrier contains no selected global optimum, trichotomy, host function equality, open-cover compactness, quotient function equality, or Real-completeness coordinate, the displayed obligations exhaust the local certificate surface. □

Theorem 6575.3 (Fermat stationary-point derivative-zero handoff)**.** *Every stationary-point consumer of an accepted* $\mathsf{FermatStationaryPoint}^{\uparrow}$ *carrier factors through the finite route*

$$I, F, E \longrightarrow D \longrightarrow Z.$$

The consumer first reads the interval source, function graph, and local extremum window; it then reads the $\mathsf{Derivative}^{\uparrow}$ *quotient row and finally the zero-derivative handoff. No route out of the packet can bypass this carrier through a selected extremal point, trichotomy, host equality of functions, open-cover compactness, quotient function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and project its rows. The only interval coordinate is I, the only graph coordinate is F, and the only local extremum coordinate is E.

The derivative row D is downstream from those three rows because its difference-quotient comparisons are read on the displayed interval and graph windows. The zero handoff Z is downstream from D because it names the stationary conclusion only as a readback of the derivative row after both extremum-side comparison ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host or compactness coordinate appears in S, every stationary-point consumer factors through the displayed finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FermatStationaryPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{FermatStationaryPoint}^{\uparrow}$ packet is generated from an interval source row, a function graph row, a local extremum window, a DerivativeUp row, a zero-derivative handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FermatStationaryPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6575.1, Definition 6575.2, and Definition 6575.3 seed the finite local-extremum-to-zero-derivative route over $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DarbouxContinuity}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FermatStationaryPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, compactness, trichotomy, global extrema, quotient function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, graph exposure, local-extremum window exactness, derivative-row admission, zero-derivative handoff, transport, replay, provenance, and local naming for BEDC.Derived.FermatStationaryPointUp.

Chapter 6576

A Concrete Naming Certificate for $\mathsf{OptionalStieltjesModulus}^{\uparrow}$

Following Riemann, Stieltjes, and Bishop–Bridges constructive analysis, the $\mathsf{OptionalStieltjesModulus}^{\uparrow}$ packet records the finite modulus surface needed before a regulated Riemann–Stieltjes consumer may read an integration boundary. It sits beside the regulated-function carrier of chapter 1906, the variation row of chapter 398, the integration-by-parts route of chapter 5131, the dyadic step approximation surface of chapter 3774, the regular rational source of chapter 301, and the terminal Real seal of chapter 38. The packet is a finite handoff object for optional partitions, oscillation budgets, variation control, and Real readback; it does not import selected limits, measure completion, quotient integration equality, host total variation, or countable choice.

Definition 6576.1 (Optional Stieltjes modulus carrier)**.** *An* $\mathsf{OptionalStieltjesModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, G, V, P, S, Q, R, E, H, C, K, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *integrand row, carrying the regulated oscillation windows read by the Stieltjes consumer.*
2. *G is the integrator row, read through the* $\mathsf{BoundedVariation}^{\uparrow}$ *carrier and the finite variation ledger named by V.*
3. *V is the bounded-variation budget row, recording the variation mass used by the modulus request.*
4. *P is the optional partition row, listing the tagged dyadic subdivision windows that may be refined before the integration read.*
5. *S is the* $\mathsf{DyadicStepApproximation}^{\uparrow}$ *row, recording the step approximation attached to the displayed partition.*
6. *Q is the modulus row, assigning a finite dyadic tolerance to every displayed partition-refinement request.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, storing the regular rational estimate exported by the modulus.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the regulated Stieltjes estimate.*
9. *H records componentwise* hsame *transport for the integrand, integrator, variation, partition, step approximation, modulus, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a Riemann–Stieltjes request through $F, G, V, P, S, Q, R, E, H$.*
11. *K is the* Pkg *provenance over* $\mathsf{OptionalStieltjesModulus}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}$, $\mathsf{DyadicStepApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{OptionalStieltjesModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a measure completion, selected limit of partitions, quotient integration equality, host total-variation theorem, ambient Real completeness, or countable-choice selection of tags.

Lean4 ✓ `BEDC.Derived.OptionalStieltjesModulusUp.OptionalStieltjesModulusTasteGate single carrier alignment`

Theorem 6576.2 (Optional Stieltjes modulus NameCert obligations)**.** *For every accepted* OptionalStieltjesModulus *carrier*

$$M = (F, G, V, P, S, Q, R, E, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{OptionalStieltjesModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the regulated integrand, bounded-variation integrator, variation budget, optional partition, dyadic step approximation, modulus, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *regulated-window exactness, requiring every oscillation read to inspect F and the displayed partition row P before the modulus row Q is consumed;*
3. *variation-budget exactness, requiring every Stieltjes-facing request to read G, V before any optional refinement or Real seal is public;*
4. *step-approximation compatibility, requiring the dyadic step row S to be attached to the same partition and variation rows;*
5. *readback and seal ordering, requiring Q to feed R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is exported;*
6. *componentwise transport, displayed replay, package provenance, and local naming through H, C, K, N.*

No obligation is stored outside $F, G, V, P, S, Q, R, E, H, C, K, N$.

Proof. Project the carrier of Definition 6576.1. The readable rows are exactly the regulated integrand, bounded-variation integrator, variation budget, optional partition, dyadic step approximation, modulus, readback, Real seal, and structural rows.

The regulated and variation stages are separate but synchronized. The integrand row F supplies the oscillation windows, while G, V supply the bounded-variation budget before any optional partition refinement is read.

The finite approximation stage is P, S, Q. The partition row is displayed first, the dyadic step row is attached to it, and the modulus row assigns the finite tolerance used by downstream estimates.

Finally R, E, H, C, K, N read back, seal, transport, replay, source, and name the same route. Since no carrier coordinate supplies measure completion, selected partition limits, quotient integration equality, host total variation, ambient Real completeness, or countable choice, the listed rows exhaust the local NameCert surface. □

Theorem 6576.3 (Optional Stieltjes modulus Riemann–Stieltjes handoff)**.** *Every Riemann–Stieltjes regulated integration consumer of an accepted* $\mathsf{OptionalStieltjesModulus}^{\uparrow}$ *carrier factors through the finite route*

$$(F, G, V) \longmapsto (P, S, Q) \longmapsto R \longmapsto E.$$

The consumer receives regulated-function windows, bounded-variation control, optional partition data, dyadic step approximation, a finite modulus, regular rational readback, and a terminal Real seal. It receives no measure-completion object, selected partition limit, quotient integration equality, host total-variation theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or choice-selected tag family.*

Proof. Start with the carrier projection. The only integration-facing source rows are F, G, V, so a consumer must read the regulated integrand and bounded-variation budget before a partition or modulus is available.

The optional partition and approximation stage is then forced through P, S, Q. This gives a finite dyadic partition surface and a finite tolerance assignment rather than a selected limiting mesh.

The readback stage is R, E. The regular rational row records the estimate supplied by the modulus, and the Real seal exposes only that terminal read.

The structural rows H, C, K, N preserve the same route under hsame, Cont, Pkg, and local naming. Because the excluded integration and choice principles have no carrier coordinate, every regulated Riemann–Stieltjes handoff factors through the displayed finite packet. □

Lean4 (statement) `BEDC.Derived.OptionalStieltjesModulusUp`

Closure status for $\mathsf{NameCert}_{\mathsf{OptionalStieltjesModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OptionalStieltjesModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6576.1, Definition 6576.2, and Definition 6576.3 seed the finite optional Stieltjes modulus route through regulated-function windows, bounded-variation control, optional partitions, dyadic step approximations, regular rational readback, Real seals, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OptionalStieltjesModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.OptionalStieltjesModulusUp`

Bridge status: none

Not claimed: This chapter does not close measure completion, selected partition limits, quotient integration equality, host total variation, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-window exactness, variation-budget exactness, step-approximation compatibility, readback-seal ordering, transport stability, replay, provenance, and local naming for BEDC.Derived.OptionalStieltjesModulusUp.

Constructive story: An $\mathsf{OptionalStieltjesModulus}^{\uparrow}$ packet is generated from a regulated-function row, bounded-variation integrator row, variation budget, optional partition row, dyadic step approximation row, modulus row, RegSeqRat readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6577

A Concrete Naming Certificate for MetaCICCandidateNormalizationWitness$^{\uparrow}$

The MetaCICCandidateNormalizationWitness$^{\uparrow}$ packet realizes the normalization frontier of section 9386.4 as finite candidate evidence rather than as an evidence-free closed-term theorem. It sits beside the frontier carrier of chapter 892, the bounded comparison route of chapter 752, the sealed closed type checker of chapter 2182, the candidate projection boundary of chapter 4366, and the candidate router of chapter 5210. The packet names a local witness surface from a closed typed term to candidate evidence, normal endpoint data, obstruction rows, transport, replay, provenance, and local naming; it does not assert global strong normalization, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.

Definition 6577.1 (MetaCIC candidate normalization witness carrier)**.** *A* MetaCICCandidateNormalizationWitness *carrier is a finite* BHist *packet*

$$W = (T, A, K, N, E, O, D, B, H, C, P, L)$$

with the following displayed rows:

1. *T is the closed typed term row, recording the sealed empty-context typing evidence read from the closed-type-checker surface.*
2. *A is the candidate anchor row, naming the candidate family, closed-candidate wrapper, or finite candidate-set membership used by the witness.*
3. *K is the candidate-evidence row, storing the actual membership, reducibility, or adequacy datum attached to T.*
4. *N is the normal endpoint row, recording the bounded normal form or no-infinite-reduction endpoint reached from K.*
5. *E is the endpoint comparison row consumed by bounded normal-equality checkers.*
6. *O is the obstruction row, retaining any missing candidate, missing closedness, blocked residual, or failed endpoint comparison as a visible boundary.*
7. *D is the transport domain row, tying the witness to the candidate closed-projection boundary.*
8. *B is the bounded checker handoff row, exposing only finite comparison data to downstream deciders.*

9. *H records componentwise* hsame *transport for closed typed terms, candidate anchors, candidate evidence, endpoints, comparisons, obstructions, domain rows, bounded handoffs, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a MetaCIC normalization request through $T, A, K, N, E, O, D, B, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, $\mathsf{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}}$ *row.*

The classifier compares accepted witnesses only through these rows. It has no coordinate for evidence-free normalization of arbitrary closed terms, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Theorem 6577.2 (MetaCIC candidate normalization witness obligations)**.** *For every accepted* $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$ *carrier*

$$W = (T, A, K, N, E, O, D, B, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed typed term, candidate anchor, candidate evidence, normal endpoint, endpoint comparison, obstruction, transport domain, bounded checker handoff, transport, replay, provenance, and local naming rows;*
2. *candidate visibility, requiring every normal endpoint read to inspect A, K before N is consumed;*
3. *closed-type seal discipline, requiring T and D to be displayed before candidate evidence is projected as a closed-term witness;*
4. *bounded equality handoff, requiring E, B to expose only finite endpoint comparison data to checker consumers;*
5. *obstruction exactness, requiring O to retain missing candidate, missing closedness, blocked residual, and failed endpoint comparison boundaries rather than discharging them;*
6. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, L.*

No witness obligation is stored outside $T, A, K, N, E, O, D, B, H, C, P, L$.

Proof. Project the carrier of Definition 6577.1. The accepted packet exposes exactly the closed typed term, candidate anchor, candidate evidence, normal endpoint, endpoint comparison, obstruction, projection boundary, bounded checker handoff, and structural rows.

Candidate visibility is read through A, K, N. A normal endpoint is public only after the candidate anchor and candidate-evidence rows have been displayed.

Closed-type and projection discipline is read through T, D. These rows keep the witness attached to the sealed closed term and to the candidate closed-projection boundary before any endpoint comparison is offered.

The bounded comparison stage is E, B, with O retaining the negative boundary. Thus a checker receives finite endpoint comparison data and a visible obstruction row, not an evidence-free normalization theorem.

Finally H, C, P, L transport, replay, source, and name the same route. Since no carrier coordinate supplies global strong normalization, unrestricted subject reduction, full confluence, quotient

syntax equality, evaluator soundness, or closed consistency, the listed rows exhaust the local NameCert surface. □

Theorem 6577.3 (MetaCIC candidate normalization witness handoff)**.** *Every public normalization-frontier consumer of an accepted* $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$ *carrier factors through*

$$T \longmapsto A, K \longmapsto N, E \longmapsto O, D, B \longmapsto H, C, P, L.$$

The consumer receives a sealed closed typed term, candidate evidence, normal endpoint data, bounded endpoint comparison, retained obstruction, candidate projection boundary, bounded checker handoff, transport, replay, provenance, and local naming. It receives no normalization theorem for arbitrary closed typed terms without candidate evidence.

Proof. Start from the closed typed term row T. The only route from this row to a normal endpoint is through the candidate anchor and evidence rows A, K.

The endpoint and checker stage is N, E, B. It exports only bounded normal endpoint and comparison data, and the projection boundary row D keeps that export attached to the candidate-mediated closed-term surface.

The obstruction row O remains visible for missing candidate evidence, failed closedness, blocked residual obligations, or failed endpoint comparison. Therefore the handoff does not collapse into an unconditional strong-normalization theorem.

The structural rows H, C, P, L preserve and name the same finite witness. Because no excluded global endpoint has a carrier coordinate, every public consumer factors through the displayed route. □

Falsifiable prediction: Any $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$ read that reaches normal endpoint data or bounded endpoint comparison without exposing the closed typed term, candidate anchor, candidate evidence, projection boundary, bounded checker handoff, obstruction, transport, replay, provenance, and local naming rows is outside the carrier surface.

Independence witness: metacicnormalizationfrontier, boundednormalequalitychecker, closed-type-checker-seal, metacic-candidate-closed-projection-boundary: no listed sibling is bijective with $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$ because this packet jointly carries closed typed term data, candidate evidence, normal endpoint data, bounded endpoint comparison, obstruction, projection boundary, bounded checker handoff, transport, replay, provenance, and local NameCert rows.

Lean follow-up should provide Nontrivial and FieldFaithful instances in the MetaCICCandidateNormalizationWitnes TasteGate file so the closed-term, candidate, endpoint, obstruction, and checker rows are audited as distinct fields.

Lean4 (statement) `BEDC.Derived.MetaCICCandidateNormalizationWitnessUp`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6577.1, Definition 6577.2, and Definition 6577.3 seed the finite candidate-witness route from sealed closed terms through candidate evidence, normal endpoints, bounded comparison, obstruction, projection boundary, checker handoff, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MetaCICCandidateNormalizationWitnessUp`

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate visibility, closed-type seal discipline, bounded equality handoff, obstruction exactness, transport stability, replay, provenance, local naming, Nontrivial, and FieldFaithful instances for BEDC.Derived.MetaCICCandidateNormalizationWitnessUp.

Constructive story: A MetaCICCandidateNormalizationWitness$^{\uparrow}$ packet is generated from a sealed closed typed term row, candidate anchor, candidate evidence, normal endpoint, endpoint comparison, obstruction, projection boundary row, bounded checker handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6578

A Concrete Naming Certificate for $\mathsf{ConvergenceFilter}^{\uparrow}$

Following Kelley and Bourbaki on filter convergence, the $\mathsf{ConvergenceFilter}^{\uparrow}$ packet records the standard neighbourhood-eventuality reading of a displayed filter base as finite BEDC NameCert data. It sits after the generated-base interface of chapter 1601, the neighbourhood-eventuality sibling of chapter 4133, the Cauchy-filter route of chapter 333, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives Real and Completion consumers a convergence-facing handoff without quotient filters, selected limits, countable choice, host equality of filters, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6578.1 (Convergence filter carrier)**.** *A* $\mathsf{ConvergenceFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, Nb, x, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row, carrying the finite base tokens, finite-intersection discipline, refinement witnesses, and displayed event rows.*
2. *Nb is the neighbourhood and eventuality row at the displayed point, recording which finite neighbourhood tokens are met by which base events.*
3. *x is the limit-point row, naming the candidate point only through the visible neighbourhood-eventuality surface.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when the convergence packet is sampled by completion-facing consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same displayed windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the filter-base, neighbourhood, limit-point, window, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for the filter base, neighbourhood row, limit-point row, StreamName windows, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite base events through neighbourhood eventuality, limit-point naming, windows, readback, and Real sealing.*

9. *P is the* Pkg *provenance over* ConvergenceFilter$^{\uparrow}$, FilterBase$^{\uparrow}$, CauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{ConvergenceFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, Nb, x, W, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, host equality of neighbourhood systems, host real equality, countable-choice schedules, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConvergenceFilterUp.ConvergenceFilterTasteGate single carrier alignment`

Theorem 6578.2 (Convergence filter NameCert obligations)**.** *For every accepted* ConvergenceFilter$^{\uparrow}$ *carrier $K = (F, Nb, x, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{ConvergenceFilter}^{\uparrow}}$ *surface consists exactly of filter-base exposure through F, neighbourhood-eventuality exposure through Nb, limit-point naming through x, StreamName window exposure through W, RegSeqRat readback through R, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6578.1. The accepted packet exposes exactly the rows $F, Nb, x, W, R, E, H, C, P, N$.

The convergence-facing rows are upstream. Row F supplies finite filter-base data, and Nb records the neighbourhood-eventuality checks at the displayed point named by x.

The Nb row is read through the NeighborhoodFilter$^{\uparrow}$ route before the Cauchy-filter and Real-completion handoff, so the eventuality consumer receives the local neighbourhood filter surface rather than an unbounded host filter.

The completion-facing rows are downstream. Consumers read finite windows through W, regular rational readback through R, and the terminal Real seal through E only after the filter-base and neighbourhood rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since quotient filters, selected limits, host neighbourhood equality, host real equality, countable choice, and ambient Real completeness are absent from the classifier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ConvergenceFilterUp.ConvergenceFilterNamecertObligations`

Theorem 6578.3 (Convergence filter completion handoff)**.** *Every completion-facing read exported by an accepted* ConvergenceFilter$^{\uparrow}$ *carrier factors through the displayed* FilterBase$^{\uparrow}$ *row F, the neighbourhood-eventuality row Nb, the limit-point row x, the* StreamName$^{\uparrow}$ *row W, the* RegSeqRat$^{\uparrow}$ *row R, and the terminal* Real$^{\uparrow}$ *seal row E in that order. When the filter-base row is also read through the* CauchyFilter$^{\uparrow}$ *sibling surface, the handoff remains a finite neighbourhood convergence route and not a quotient-filter or selected-limit construction.*

Proof. Start with the NameCert obligations of Definition 6578.2. They force every accepted consumer to expose F, Nb, x, W, R, E before any structural row can name a completion-facing read.

The filter-base row is the only source for event data. Its finite refinement and intersection witnesses are read before Nb may certify eventual membership of displayed neighbourhood tokens at x.

The completion route is then fixed by the L10 readback rows. StreamName windows are exposed through W, the regular rational readback is exposed through R, and the Real-facing endpoint is sealed through E.

If the same base is consumed by a CauchyFilter sibling, the packet still exports only the rows already visible in $F, Nb, x, W, R, E, H, C, P, N$. Thus the route is stable under the Cauchy-filter handoff while excluding quotient filters, selected limits, choice schedules, host real equality, and ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{ConvergenceFilter}^{\uparrow}$ completion-facing read exposes a FilterBaseUp row, a neighbourhood-eventuality row at a displayed limit point, StreamNameUp windows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before it can serve a Real or Completion consumer.

Independence witness: filterbase, cauchyfilter, streamname, regseqrat, real: none is bijective with $\mathsf{ConvergenceFilter}^{\uparrow}$ because this packet jointly carries the finite filter-base route, neighbourhood-eventuality checks, limit-point naming, finite windows, readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ConvergenceFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvergenceFilter}^{\uparrow}$ packet is generated from a FilterBase row, a neighbourhood-eventuality row, a displayed limit-point row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvergenceFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6578.1, Definition 6578.2, and Definition 6578.3 seed the finite convergence-filter route over $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvergenceFilter}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.ConvergenceFilterUp`

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected limits, countable choice, host equality of neighbourhood systems, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base exposure, neighbourhood eventuality, limit-point naming, CauchyFilterUp handoff, StreamNameUp window exposure, RegSeqRatUp readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ConvergenceFilterUp.

Chapter 6579

A Concrete Naming Certificate for $\mathsf{SetoidCompletion}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley-style uniform completion discipline, the $\mathsf{SetoidCompletion}^{\uparrow}$ packet separates completion data from the equality classifier of chapter 1820. The classifier supplied by $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ tells when two regular Cauchy names agree; the present packet records the completion-facing carrier, handoff, and naming surface that consumes $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the local $\mathsf{NameCert}$ row without collapsing the construction into quotient membership.

Definition 6579.1 (SetoidCompletion carrier)**.** *A* $\mathsf{SetoidCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that lists the observation windows available to a completion request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare successive readbacks and requested precisions.*
4. *Q is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row, consumed as displayed comparison data rather than as quotient equality.*
5. *A is the completion-assembly row. It records admissible regular Cauchy handoff data over S, R, D, Q before the terminal real-name seal is inspected.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-boundary row reached by the assembled regular name.*
7. *H records componentwise* hsame *transport for $S, R, D, Q, A, E, C, P, N$.*
8. *C records displayed* Cont *replay from a completion-facing request through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{SetoidCompletion}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SetoidCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient objects, selected representatives, selected limits, countable choice, ambient real completeness, host set extensionality, Lean verification, or bridge checking.

Theorem 6579.2 (SetoidCompletion NameCert obligations)**.** *For every accepted* $\mathsf{SetoidCompletion}^{\uparrow}$ *carrier* $K = (S, R, D, Q, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SetoidCompletion}^{\uparrow}}$ *surface is exhausted by the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular-rational readback row, dyadic tolerance ledger, setoid classifier row, completion-assembly row, Real seal row, transport, replay, provenance, and local naming;*
2. *window exactness, requiring every completion-facing observation to be one of the displayed windows in* S*;*
3. *readback exactness, requiring every rational value consumed by the completion route to pass through* R*;*
4. *tolerance exactness, requiring every comparison request to be discharged by an entry of* D *before the classifier row* Q *is read;*
5. *setoid-compatibility exactness, requiring* Q *to be consumed as the displayed* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *row and not as quotient equality;*
6. *assembly exactness, requiring the completion handoff row* A *to be built only from* S, R, D, Q *before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is exposed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*.*

No local certificate obligation is stored outside these rows.

Proof. Project the carrier of Definition 6579.1. The projection exposes the finite stream windows, regular-rational readback, dyadic tolerance ledger, setoid classifier row, completion-assembly row, Real seal boundary, and structural tail.

Read the NameCert row N against those projections. Window, readback, tolerance, classifier, and assembly exactness are separate rows, so the completion route cannot hide a quotient object or a selected representative inside the classifier.

Finally replay the route through H, C, P, N. Componentwise transport and Cont replay preserve the displayed packet, while provenance records the same sibling anchors. Since no carrier coordinate stores selected limits, countable choice, ambient completeness, or host set extensionality, the listed rows exhaust the local obligation surface. □

Theorem 6579.3 (SetoidCompletion regular Cauchy handoff)**.** *Every completion-facing read of an accepted* $\mathsf{SetoidCompletion}^{\uparrow}$ *carrier factors through the displayed regular Cauchy handoff*

$$S, R, D, Q, A \longmapsto E, H, C, P, N.$$

Thus the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier, and completion-assembly row are visible before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The handoff is not a quotient completion, a selected Cauchy representative, a chosen limit, a countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier projection. A completion request first reads S, R, D as the finite observation, readback, and tolerance rows available to the regular Cauchy name.

Next consume the classifier row Q through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ interface. The row supplies setoid-compatible comparison data, but it does not replace the displayed regular name by quotient equality or by a selected representative.

The assembly row A then packages the regular Cauchy handoff before the Real seal row E is inspected. Transport, replay, provenance, and local naming are supplied by H, C, P, N, so downstream completion consumers receive the same displayed route rather than an ambient completeness witness or a choice of limits. □

Closure status for $\mathsf{NameCert}_{\mathsf{SetoidCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SetoidCompletion}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a completion-assembly row, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SetoidCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6579.1, Definition 6579.2, and Definition 6579.3 seed the non-quotient completion carrier for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SetoidCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, setoid-compatible classifier consumption, regular Cauchy handoff exactness, componentwise transport, replay, provenance, and local naming for BEDC.Derived.SetoidCompletionUp.

Chapter 6580

A Concrete Naming Certificate for $\mathsf{FilterLimitCriterion}^{\uparrow}$

Following Kelley and Bourbaki on filter convergence, the $\mathsf{FilterLimitCriterion}^{\uparrow}$ packet records the finite criterion surface that lets a completion-facing consumer recognize when a displayed Cauchy filter and a displayed convergence filter name the same limit route. It sits after the filter-base carrier of chapter 1601, the Cauchy-filter row of chapter 333, the convergence-filter row of chapter 6578, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives Real and Completion consumers a criterion object without quotient filters, selected limits, countable choice, host filter equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6580.1 (Filter limit criterion carrier)**.** *A* $\mathsf{FilterLimitCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, Nb, Cy, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row, carrying finite base tokens, finite-intersection discipline, refinement witnesses, and event rows.*
2. *Nb is the neighbourhood-eventuality row inherited from the* $\mathsf{ConvergenceFilter}^{\uparrow}$ *surface, recording which displayed neighbourhood tokens are eventually met by the base.*
3. *Cy is the* $\mathsf{CauchyFilter}^{\uparrow}$ *criterion row, recording the finite pairwise closeness checks and Cauchy-facing event refinements on the same base.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by completion-facing consumers when the filter criterion is sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those displayed windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the filter-base, neighbourhood, Cauchy, window, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for the filter-base row, neighbourhood-eventuality row, Cauchy row, StreamName windows, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite filter events through neighbourhood eventuality, Cauchy closeness, windows, readback, and Real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{FilterLimitCriterion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{ConvergenceFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterLimitCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, Nb, Cy, W, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, host equality of filters or neighbourhood systems, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6580.2 (Filter limit criterion NameCert obligations)**.** *For every accepted* $\mathsf{FilterLimitCriterion}^{\uparrow}$ *carrier $K = (F, Nb, Cy, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterLimitCriterion}^{\uparrow}}$ *surface consists exactly of filter-base exposure through F, neighbourhood-eventuality exposure through Nb, Cauchy-filter criterion exposure through Cy, StreamName window exposure through W, RegSeqRat readback through R, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6580.1. The accepted packet exposes exactly the rows $F, Nb, Cy, W, R, E, H, C, P, N$.

The filter-facing rows come first. Row F supplies finite base data, Nb records neighbourhood eventuality for the displayed convergence route, and Cy records the Cauchy-facing closeness checks on that same base.

The completion-facing rows are downstream. Consumers read finite windows through W, regular rational readback through R, and the terminal Real seal through E only after the filter-base, neighbourhood, and Cauchy rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since quotient filters, selected limits, host filter equality, host neighbourhood equality, countable choice, host real equality, and ambient Real completeness are absent from the classifier, the listed rows exhaust the local NameCert surface. □

Lean4 (statement) `BEDC.Derived.FilterLimitCriterionUp`

Theorem 6580.3 (Filter limit criterion Cauchy-filter handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{FilterLimitCriterion}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{FilterBase}^{\uparrow}$ *row F, the neighbourhood-eventuality row Nb, the* $\mathsf{CauchyFilter}^{\uparrow}$ *criterion row Cy, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E in that order. The handoff remains a finite criterion route from Cauchy-filter data to convergence-filter data and not a quotient-filter, selected-limit, countable-choice, or ambient-completeness construction.*

Proof. Start with the NameCert obligations of Definition 6580.2. They force every accepted consumer to expose F, Nb, Cy, W, R, E before any structural row can name a completion-facing read.

The shared base row is the only source for event data. Its finite refinement and intersection witnesses are read before Nb may certify eventual membership of displayed neighbourhood tokens and before Cy may certify the Cauchy-facing closeness checks.

The completion route is then fixed by the L10 readback rows. StreamName windows are exposed through W, regular rational readback through R, and the Real-facing endpoint is sealed through E.

Rows H, C, P, N keep the Cauchy-filter and convergence-filter readings on the same finite packet. Thus the criterion handoff is stable for completion consumers while excluding quotient filters, selected limits, choice schedules, host equality, and ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{FilterLimitCriterion}^{\uparrow}$ completion-facing read exposes a FilterBaseUp row, a neighbourhood-eventuality row, a CauchyFilterUp criterion row, StreamNameUp windows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before it can serve a Real or Completion consumer.

Independence witness: filterbase, cauchyfilter, convergencefilter, streamname, regseqrat, real: none is bijective with $\mathsf{FilterLimitCriterion}^{\uparrow}$ because this packet jointly carries the finite filter-base route, neighbourhood-eventuality checks, Cauchy-filter criterion rows, finite windows, readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterLimitCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterLimitCriterion}^{\uparrow}$ packet is generated from a FilterBase row, a neighbourhood-eventuality row, a CauchyFilter criterion row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterLimitCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6580.1, Definition 6580.2, and Definition 6580.3 seed the finite filter-limit criterion route over $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{ConvergenceFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterLimitCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FilterLimitCriterionUp`

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected limits, countable choice, host equality of filters or neighbourhood systems, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base exposure, neighbourhood eventuality, Cauchy-filter criterion exposure, StreamNameUp window exposure, RegSeqRatUp readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FilterLimitCriterionUp.

Chapter 6581

RegularCauchyInterleavingStability NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ packet isolates the finite stability surface for interleaved regular Cauchy names. It reads the regular rational rows of chapter 301, the finite stream schedules of chapter 126, the dyadic tolerance core of chapter 291, the interleaving carrier of chapter 414, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a named stability certificate between selected even–odd windows and the shared modulus ledger; it does not identify quotient reals, choose host limits, or invoke ambient completeness.

Definition 6581.1 (Regular Cauchy interleaving stability carrier)**.** *A* $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, S_A, S_B, \sigma, M, W, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose finite regular Cauchy windows are interleaved.*
2. *S_A and S_B are the selected* $\mathsf{StreamName}^{\uparrow}$ *schedules for the even and odd sides.*
3. *σ is the alternating selector row inherited from the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *carrier.*
4. *M is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus ledger, carrying the matched dyadic budgets for selected even and odd requests.*
5. *W records the selected even and odd* $\mathsf{RegSeqRat}^{\uparrow}$ *windows after the schedules and shared modulus have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after $A, B, S_A, S_B, \sigma, M, W$ have been read.*
7. *H records componentwise* hsame *transport for the source rows, schedules, selector, shared modulus ledger, selected windows, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records the* Pkg *provenance over* $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, S_A, S_B, \sigma, M, W, E, H, C, P, N$. It has no coordinate for quotient-real equality, selected host limits, ambient completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingStabilityUp.RegularCauchyInterleavingStabilityTasteGate single carrier alignment`

Theorem 6581.2 (Regular Cauchy interleaving stability obligations)**.** *For every accepted* RegularCauchyInterleaving *carrier*

$$T = (A, B, S_A, S_B, \sigma, M, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingStability}^{\uparrow}}$ *surface is exhausted by five obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $A, B, S_A, S_B, \sigma, M, W, E, H, C, P, N$;*
2. *selected-window stability: the even and odd rows in W are read from A and B through S_A, S_B, σ without changing the selected finite regular Cauchy windows;*
3. *shared-modulus exactness: every selected even or odd request uses the corresponding component of the single dyadic ledger M;*
4. *componentwise transport: H carries* hsame *transport for each displayed row and C replays the same finite route through* Cont*;*
5. *Real seal nonescape: E exports no quotient real, selected host limit, ambient completeness principle, or bridge-checked schema beyond the displayed rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingStabilityUp.RegularCauchyInterleavingStabilityCarrier obligations`

Theorem 6581.3 (Regular Cauchy interleaving stability classifier)**.** *Accepted* $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ *packets classify together exactly when the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *windows and shared modulus rows transport componentwise through the displayed* hsame *and* Cont *routes. Equivalently, the classifier sees the same two source rows A, B, the same schedules S_A, S_B, the same alternating selector σ, the same shared dyadic ledger M, the same selected-window row W, and the same Real seal boundary E up to the displayed transport, replay, provenance, and local naming rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingStabilityUp.RegularCauchyInterleavingStabilityClassifier`

Falsifiable prediction: An accepted $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ packet is refuted if a Real or Completion consumer reaches the seal row E without first displaying the even and odd $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the two $\mathsf{StreamName}^{\uparrow}$ schedules, the alternating selector, the shared $\mathsf{DyadicRatCore}^{\uparrow}$ modulus ledger, the selected-window row, transport, replay, provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regularcauchyinterleaving, real: none alone is bijective with $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ because the packet jointly binds two regular rational sources, two selected schedules, one alternating selector, one shared dyadic modulus ledger, selected even and odd windows, one Real seal, componentwise transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyInterleavingStability}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two StreamName schedules, an alternating selector, a shared DyadicRatCore modulus ledger, selected even and odd windows, a Real seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingStability}^{\uparrow}}$, seed)

Scope closed: Definition 6581.1, Definition 6581.2, and Definition 6581.3 seed the finite stability certificate for interleaved regular Cauchy names.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyInterleavingStability}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.RegularCauchyInterleavingStabilityUp`

Bridge status: none

Not claimed: This chapter does not close quotient reals, selected host limits, ambient completeness, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, selected-window stability, shared-modulus exactness, componentwise transport, Real seal nonescape, and classifier transport for BEDC.Derived.RegularCauchyInterleavingStabilityUp.

Chapter 6582

A Concrete Naming Certificate for $\mathsf{LowerUpperRealSeparator}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LowerUpperRealSeparator}^{\uparrow}$ packet records the finite BEDC surface by which a lower-real ledger and an upper-real ledger are held apart by an explicit separator row. It reads the one-sided real carriers of chapter 1446 and chapter 1447, the cut interface of chapter 404, the located-real handoff of **??**, the stream window of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names only the constructive separator boundary; it does not assert trichotomy, quotient equality, selected suprema, countable choice, or host completeness.

Definition 6582.1 (LowerUpperRealSeparator carrier)**.** *A* $\mathsf{LowerUpperRealSeparator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *ledger, including the lower-set classifier, rational read rows, and displayed lower approximation evidence.*
2. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *ledger, including the upper-set classifier, rational read rows, and displayed upper approximation evidence.*
3. *S is the separator row, recording the finite apartness or located-cut boundary that keeps accepted lower and upper reads in their declared order.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window through which consumers request finite readback precision.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the windowed request into regular rational comparison data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal naming the terminal real-facing route when the lower and upper evidence is exported together.*
7. *H records componentwise* hsame *transport for L, U, S, W, R, E and the structural rows.*
8. *C records displayed* Cont *replay of the route*
$$L, U, S \longmapsto W, R \longmapsto E.$$
9. *P is the* Pkg *provenance over* $\mathsf{LowerUpperRealSeparator}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerUpperRealSeparator}^{\uparrow}}$ *row.*

The classifier of the carrier compares accepted packets only through the displayed lower ledger, upper ledger, separator row, stream window, regular rational readback, Real seal, componentwise transport, continuation replay, provenance, and local naming row.

Theorem 6582.2 (LowerUpperRealSeparator NameCert obligations)**.** *For every accepted* LowerUpperRealSeparato *carrier*

$$G = (L, U, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerUpperRealSeparator}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the lower ledger, upper ledger, separator row, stream window, regular rational readback, Real seal, transport row, replay row, provenance row, and local naming row;*
2. *componentwise classifier compatibility, requiring accepted comparison to be read row by row from $L, U, S, W, R, E, H, C, P, N$;*
3. *lower/upper compatibility, requiring the separator row S to display the boundary by which lower and upper reads remain ordered under the same windowed request;*
4. *Real-route stability, requiring the terminal read through E to preserve the preceding* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
5. *separator-ledger nonescape, requiring the provenance and local NameCert rows to source no trichotomy, quotient equality, selected supremum, countable-choice, or host-completeness coordinate.*

No separator obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 6582.1. The visible one-sided real coordinates are exactly the lower ledger L, the upper ledger U, the separator row S, the stream window W, the regular rational readback R, and the Real seal E.

Classifier compatibility is componentwise. A transported packet keeps each ledger on the same displayed field, and replay through C can only revisit the rows already named by L, U, S, W, R, E.

Lower/upper compatibility is supplied by the separator row. It is read after the one-sided ledgers and before the readback window, so a consumer cannot replace it by a host trichotomy decision or by an unlisted supremum selector.

Finally inspect P and N. Their source list contains only the displayed packet dependencies and local naming data. Since no coordinate stores quotient real equality, choice, selected suprema, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6582.3 (LowerUpperRealSeparator Real route)**.** *Every consumer route from an accepted* $\mathsf{LowerUpperRealSeparator}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

after reading the displayed lower ledger, upper ledger, and separator row. The route may export finite one-sided-real boundary data to $\mathsf{Real}^{\uparrow}$*, but it cannot expose trichotomy, quotient equality, selected suprema, countable choice, or host completeness.*

Proof. Begin with an accepted carrier $G = (L, U, S, W, R, E, H, C, P, N)$. The only route out of the one-sided ledgers first reads L and U, then the separator row S.

The consumer request is then windowed by W and read back by R. These rows give finite regular rational comparison data before any terminal Real-facing seal is available.

The Real export is the row E, transported by H and replayed by C. Because P and N name only this finite route, a consumer receives no trichotomy, quotient equality, selected supremum, countable-choice principle, or host-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerUpperRealSeparator}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerUpperRealSeparator}^{\uparrow}$ packet is generated from lower and upper real ledgers, a separator row, finite StreamName windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerUpperRealSeparator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6582.1, Definition 6582.2, and Definition 6582.3 seed the one-sided-real separator boundary over displayed lower, upper, stream, readback, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerUpperRealSeparator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Cauchy completion, trichotomy, quotient equality, selected suprema, countable choice, host completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, componentwise classifier compatibility, lower/upper compatibility, Real-route stability, separator-ledger nonescape, and a Lean carrier surface for BEDC.Derived.LowerUpperRealSeparatorUp.

Chapter 6583

A Concrete Naming Certificate for KolmogorovRieszFiniteTranslation$^{\uparrow}$

Following Kolmogorov and Riesz, the KolmogorovRieszFiniteTranslation$^{\uparrow}$ packet records the finite translation-tightness surface used by compactness consumers before any ambient function-space compactness theorem is available. It sits between the Frechet–Kolmogorov compactness route of chapter 3846, the totally bounded metric surface of TotallyBoundedMetric$^{\uparrow}$, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite translation windows, finite norm-readback rows, and the handoff into Frechet compactness; it is not an unrestricted compact embedding theorem, a quotient function-space object, countable choice, selected subsequence data, or ambient Real$^{\uparrow}$ completeness.

Definition 6583.1 (Kolmogorov-Riesz finite translation carrier)**.** *A* KolmogorovRieszFiniteTranslation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, T, D, R, Q, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-family row read through the Frechet–Kolmogorov compactness sibling.*
2. *T is the finite translation-window row, listing the finitely many shifts and tolerances exposed to the present compactness request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to compare translation budgets, radius bounds, and finite-window refinements.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for all rational norm estimates and translation differences.*
5. *Q is the* Real$^{\uparrow}$ *seal row reached by the displayed norm and distance reads.*
6. *B is the* TotallyBoundedMetric$^{\uparrow}$ *support row recording the finite metric net seen by the request.*
7. *E is the Frechet compactness handoff row; it packages F, T, D, R, Q, B as finite translation-tightness data.*
8. *H records componentwise* hsame *transport for the family, translation, tolerance, readback, Real seal, totally bounded, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a compactness-facing request through the same rows.*

10. *P is the* Pkg *provenance over* KolmogorovRieszFiniteTranslation$^{\uparrow}$, FrechetKolmogorovCompactness$^{\uparrow}$, TotallyBoundedMetric$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{KolmogorovRieszFiniteTranslation}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted function-space compactness, quotient equality of functions, countable choice, selected subsequences, selected cluster points, ambient measure theory, Lean verification, or bridge checking.

Lean4 (statement) `BEDC.Derived.KolmogorovRieszFiniteTranslationUp`

Theorem 6583.2 (Kolmogorov-Riesz finite translation NameCert obligations)**.** *For every accepted carrier $K = (F, T, D, R, Q, B, E, H, C, P, N)$ of Definition 6583.1, the local* NameCert$_{\text{KolmogorovRieszFiniteTranslation}^{\uparrow}}$ *surface is exhausted by the following finite obligation rows:*

1. *function-family admission through F, read beside the Frechet–Kolmogorov compactness sibling rather than through a host function space;*
2. *finite translation-window admission through T, requiring every shift and tolerance inspected by the consumer to be named in the displayed window;*
3. *dyadic tolerance and regular-rational readback exactness through D and R;*
4. *Real seal admission through Q for all displayed norm and distance values;*
5. *totally bounded metric support through B, exposing only finite net rows and their metric comparisons;*
6. *Frechet compactness handoff exactness through E;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local obligation is stored outside these rows.

Proof. Project the carrier rows. The projection exposes the function-family row, finite translation-window row, dyadic tolerance ledger, regular rational readback, Real seal, totally bounded metric support, Frechet handoff, and structural tail.

Read the local NameCert row N against those projections. Translation tightness is checked only by the displayed finite shifts and tolerances in T, and all numerical comparisons pass through the dyadic, regular-rational, and Real rows D, R, Q.

Inspect the compactness-facing endpoint. The request can reach the totally bounded metric support B and then the handoff row E, but the carrier contains no unrestricted function-space compactness coordinate, quotient-function equality, countable-choice schedule, selected subsequence, or selected cluster point. Thus the listed rows exhaust the finite translation-tightness obligation surface. □

Theorem 6583.3 (Kolmogorov-Riesz handoff to Frechet compactness)**.** *Every compactness-facing read of an accepted* KolmogorovRieszFiniteTranslation$^{\uparrow}$ *carrier factors through the displayed route*

$$F, T, D, R, Q, B \longmapsto E, H, C, P, N.$$

The consumer receives a finite translation-tightness subcertificate and the corresponding Frechet compactness handoff, but receives no unrestricted compact embedding theorem, no quotient function-space object, no selected subsequence, no selected cluster point, and no ambient Real$^{\uparrow}$ *completeness claim.*

Proof. Begin with a compactness-facing request. It first reads the family row F and the finite translation-window row T, so the only shifts and tolerances available to the request are already displayed.

The numerical tests then pass through D, R, Q. These rows expose dyadic budgets, regular-rational readbacks, and Real seal values, while preserving the finite-window source of each comparison.

The request next reaches the totally bounded metric support B and the handoff row E. These rows present the finite translation-tightness packet to Frechet compactness consumers without adding an unrestricted compactness theorem or a chosen limiting object.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no carrier coordinate stores quotient function-space equality, countable choice, a selected subsequence, or a selected cluster point, the exported handoff is exactly the finite subcertificate named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovRieszFiniteTranslation}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovRieszFiniteTranslation}^{\uparrow}$ packet is generated from a function-family row, finite translation windows, DyadicRatCore tolerance ledgers, RegSeqRat readbacks, Real seal rows, totally bounded metric support, Frechet compactness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovRieszFiniteTranslation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6583.1, Definition 6583.2, and Definition 6583.3 seed the finite translation-tightness subcertificate consumed by Frechet compactness and Real-completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovRieszFiniteTranslation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted compact embedding, quotient function-space equality, countable choice, selected subsequences, selected cluster points, ambient measure theory, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite translation-window exactness, dyadic and regular-rational readback exactness, Real seal admission, totally bounded metric support, Frechet compactness handoff, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovRieszFiniteTranslationUp.

Chapter 6584

A Concrete Naming Certificate for $\mathsf{CauchyRegularExtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRegularExtraction}^{\uparrow}$ packet records the finite route by which ordinary Cauchy evidence is read as a regular rational sequence before a $\mathsf{Real}^{\uparrow}$ consumer sees a terminal seal. It sits over the Cauchy evidence surface of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet makes the extraction route explicit without quotient equality of Cauchy streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6584.1 (Cauchy regular extraction carrier)**.** *A* $\mathsf{CauchyRegularExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (C, D, W, R, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the ordinary* $\mathsf{Cauchy}^{\uparrow}$ *evidence row, carrying the displayed Cauchy comparisons and the source history being regularized.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic radii, comparison cells, slack rows, and endpoint observations used by the extraction.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the source Cauchy evidence and by each downstream precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance data.*
5. *E is the extraction witness row, recording that the Cauchy comparisons, dyadic tolerances, and finite windows determine the displayed regular rational readback.*
6. *H records componentwise* hsame *transport for the Cauchy evidence, dyadic ledger, windows, regular readback, extraction witness, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay of the extraction route*

$$C, D, W \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyRegularExtraction}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRegularExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, D, W, R, E, H, T, P, N$. It has no coordinate for quotient equality of Cauchy streams, a selected limit, countable-choice schedules, host real equality, a hidden completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyRegularExtractionUp`

Theorem 6584.2 (Cauchy regular extraction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRegularExtraction}^{\uparrow}$ *carrier*

$$X = (C, D, W, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRegularExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Cauchy evidence row, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, extraction witness, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *Cauchy-to-window exactness, requiring every read of ordinary Cauchy evidence to pass through the displayed finite windows W and dyadic tolerance data D;*
3. *regular readback exactness, requiring R to read only the displayed C, D, W data and not an unbounded stream representative or quotient class;*
4. *extraction-witness stability, requiring E to bind the same Cauchy evidence, tolerance ledger, windows, and regular readback under the displayed transports;*
5. *downstream nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer to receive only $C, D, W, R, E, H, T, P, N$ and no selected limit, quotient equality, countable choice, host real equality, hidden completion operator, or ambient completeness claim.*

No $\mathsf{CauchyRegularExtraction}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 6584.1. The accepted packet displays exactly the ordinary Cauchy evidence, dyadic tolerance ledger, finite windows, regular readback, extraction witness, transport, replay, provenance, and local naming rows.

Cauchy-to-window exactness is read from C, D, W. The Cauchy evidence is visible only through the finite comparisons and tolerance cells recorded in the packet, so no consumer can inspect a hidden stream representative.

Regular readback exactness is the role of R. Since R is downstream of the displayed windows and dyadic ledger, it records the regular rational sequence as certificate data rather than as a quotient or selected limit.

The extraction witness E binds the same rows under the componentwise transport H and replay T. Thus the NameCert surface is stable under the displayed hsame and Cont data.

Finally, P and N source and name the same finite route. The excluded objects are absent from every accepted packet, so the listed rows exhaust the local NameCert obligations. □

Theorem 6584.3 (Cauchy regular extraction RegSeqRat readback)**.** *Let*

$$X = (C, D, W, R, E, H, T, P, N)$$

be an accepted $\mathsf{CauchyRegularExtraction}^{\uparrow}$ *carrier. Every regular rational read exported by X factors through the finite route*

$$C, D, W \longmapsto R \longmapsto E \longmapsto H, T, P, N.$$

Consequently a downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumer receives a displayed Cauchy evidence row, DyadicRatCore tolerance ledger, StreamName window row, RegSeqRat readback, extraction witness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data, and receives no quotient stream equality, selected limit, countable choice, host real equality, hidden completion operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness credential.*

Proof. Begin with a regular rational read of an accepted packet. Projection exposes the ordered carrier rows $C, D, W, R, E, H, T, P, N$.

The read starts at the ordinary Cauchy evidence row C. The dyadic tolerance ledger D and finite windows W are the only displayed observations available to the extraction route.

The regular rational sequence is then read from R. Because R is downstream of C, D, W, it cannot be replaced by quotient equality of streams, countable choice, or a selected limit.

The witness row E records that the same displayed observations determine the readback. The structural rows H, T, P, N transport, replay, source, and name that route.

Therefore every downstream read factors through the displayed packet and exports only the listed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRegularExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRegularExtraction}^{\uparrow}$ packet is generated from ordinary Cauchy evidence, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, an extraction witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRegularExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 6584.1, Definition 6584.2, and Definition 6584.3 seed the finite route from ordinary Cauchy evidence through dyadic tolerances and StreamName windows to RegSeqRat readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRegularExtraction}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.CauchyRegularExtractionUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy streams, selected limits, countable choice, host real equality, hidden completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-to-window exactness, regular readback exactness, extraction-witness stability, and downstream nonescape for BEDC.Derived.CauchyRegularExtractionUp.

Chapter 6585

A Concrete Naming Certificate for $\mathsf{ConvexCompactInterval}^{\uparrow}$

Following Bishop and Bridges' constructive analysis of intervals, and the Bourbaki treatment of convex compact subsets of the real line, the $\mathsf{ConvexCompactInterval}^{\uparrow}$ packet records the finite NameCert surface by which a closed bounded interval is read as a convex compact interval through located finite nets. It sits beside the closed interval carrier of chapter 1591, the located finite-net route of chapter 1808, the finite ϵ-net packet of chapter 1767, the compact metric carrier of chapter 129, the terminal real seal of chapter 38, the regular rational handoff of chapter 301, and the stream-window interface of chapter 126. The packet names the constructive compact-interval route only: it does not assert an open-cover Heine–Borel theorem, ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient equality of interval names, selected suprema, or countable choice.

Definition 6585.1 (Convex compact interval carrier)**.** *A* $\mathsf{ConvexCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, F, M, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source interval, including its endpoint ledger and closed containment classifier.*
2. *L is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *row that records located interval reads as finite observation windows.*
3. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *net row obtained from those located windows and the displayed tolerance ledger.*
4. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *consumer row, read as compact-metric finite-net data rather than as open-cover compactness.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the finite interval observations as regular rational data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the closed interval, located net, finite net, compact-metric row, and regular handoff have been displayed.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request window through which endpoint, net, and tolerance reads are inspected.*
8. *H records componentwise* hsame *transport for I, L, F, M, R, E, W and the structural rows.*

9. *C records displayed* Cont *replay of the route*

$$I, W \longmapsto L, F \longmapsto M, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* ConvexCompactInterval$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, LocatedClosedIntervalFiniteNet$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactMetric$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ConvexCompactInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores an open-cover compactness proof, ambient real completeness, quotient equality of interval names, a selected supremum, or a choice-selected point.

Lean4 (statement) BEDC.Derived.ConvexCompactIntervalUp

Theorem 6585.2 (Convex compact interval NameCert obligations)**.** *For every accepted* ConvexCompactInterval$^{\uparrow}$ *carrier $K = (I, L, F, M, R, E, W, H, C, P, N)$, the local* NameCert$_{\text{ConvexCompactInterval}^{\uparrow}}$ *surface is exhausted by the following finite rows:*

1. *carrier admission for the closed bounded interval, located finite-net row, finite ϵ-net row, compact-metric row, regular rational handoff, Real seal, stream window, transport, replay, provenance, and local naming;*
2. *convex interval source compatibility, requiring all convexity and endpoint reads to be inherited from the displayed* ClosedBoundedInterval$^{\uparrow}$ *row I;*
3. *located-net visibility, requiring every compactness-facing request to pass through W, L, F before the compact-metric row M is available;*
4. *Real-seal stability, requiring R and E to be downstream from the same finite interval and net rows;*
5. *ledger non-escape, requiring H, C, P, N to export no open-cover Heine–Borel principle, ambient* Real$^{\uparrow}$ *completeness, quotient equality of interval names, selected supremum, or countable-choice schedule.*

No local NameCert obligation is stored outside $I, L, F, M, R, E, W, H, C, P, N$.

Proof. Project the carrier of Definition 6585.1. The only analytic rows available are the closed bounded interval source, the located finite-net row, the finite net row, the compact-metric consumer row, the regular rational handoff, the Real seal, and the stream request window.

Convexity and endpoint discipline are inherited from I. The carrier has no second ambient real line or host interval equality, so a consumer cannot replace the closed interval source by an untracked convex set.

Located compactness is read through the finite route W, L, F, M. The request window exposes only finite endpoint, tolerance, and net observations before the compact-metric row is visible, so the route is finite-net compactness rather than open-cover compactness.

Finally, R, E, H, C, P, N preserve and name that same route. Since the carrier has no coordinate for ambient Real completeness, quotient equality, selected suprema, or countable choice, the displayed rows exhaust the local NameCert obligations. □

Theorem 6585.3 (Convex compact interval located-net route)**.** *Every* CompactMetric$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* ConvexCompactInterval$^{\uparrow}$ *packet receives the finite dependency route*

ClosedBoundedInterval$^{\uparrow}$, LocatedClosedIntervalFiniteNet$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, S

The route reads the closed interval source through located finite windows, repacks those windows as finite ϵ-net and regular rational data, and reaches the terminal Real seal only after the compact-metric row is displayed. It exports no open-cover Heine–Borel theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient interval equality, selected supremum, or countable-choice data.*

Proof. Begin with the carrier projection. The interval row I supplies the closed bounded source, while W gives the finite request window through which endpoints and tolerances are inspected.

Apply the located-net rows L and F. These rows convert the displayed window into located finite-net data and then into the finite ϵ-net surface consumed by M.

The regular and Real-facing rows are downstream. The handoff R repacks the same finite observations as regular rational data, and the seal E is available only after the preceding closed-interval, located-net, finite-net, and compact-metric rows have been read.

Transport, replay, provenance, and local naming preserve the same finite route. Since no excluded compactness, completeness, quotient, supremum, or choice coordinate occurs in K, the located-net route is exactly the dependency surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvexCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvexCompactInterval}^{\uparrow}$ packet is generated from a closed bounded interval, located closed-interval finite-net rows, finite epsilon-net data, a compact-metric consumer row, RegSeqRat handoff, Real seal, StreamName window, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvexCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6585.1, Definition 6585.2, and Definition 6585.3 seed the constructive convex compact interval route over closed interval, located finite-net, compact metric, regular rational, stream, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvexCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover Heine–Borel compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient interval equality, selected suprema, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convex interval source compatibility, located-net visibility, Real-seal stability, ledger nonescape, and a Lean carrier surface for BEDC.Derived.ConvexCompactIntervalUp.

Chapter 6586

A Concrete Naming Certificate for $\mathsf{EffectiveSeparableMetric}^{\uparrow}$

Following Bishop-Bridges/Weihrauch-style computable-analysis practice, the $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ packet records the finite BEDC surface by which a metric space is read with an explicit dense schedule and computable distance readback. It sits after the metric observation carrier of chapter 57, the Real separability route of chapter 1809, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives Real and Completion consumers a displayed dense witness without countable choice, quotient stream equality, selected representatives, host completeness, or an ambient separability theorem.

Definition 6586.1 (EffectiveSeparableMetric carrier)**.** *An* $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, D, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-read ledger, carrying the displayed point rows and finite distance observations.*
2. *D is the dense schedule row, listing the finite approximation requests by which a consumer reads a named dense family.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite precision requests are made.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the metric comparisons attached to those windows.*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, D, W, R have been exposed.*
6. *H records componentwise* hsame *transport for M, D, W, R, Q and the structural rows.*
7. *C records displayed* Cont *replay of the route*

$$M, D \longmapsto W, R \longmapsto Q.$$

8. *P is the* Pkg *provenance over* $\mathsf{EffectiveSeparableMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveSeparableMetric}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, D, W, R, Q, H, C, P, N$. It has no coordinate for host separability, choice-selected dense sequences, quotient stream equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6586.2 (EffectiveSeparableMetric NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ *carrier*

$$E = (M, D, W, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveSeparableMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric ledger M, dense schedule D, StreamName window W, RegSeqRat readback R, Real seal Q, transport row H, replay row C, provenance row P, and local naming row N; componentwise classifier compatibility for those rows; dense-schedule exposure through D before any separability read; metric readback through M, W, R, Q; and non-escape from host separability, choice-selected enumeration, quotient stream equality, selected representatives, and ambient completeness.*

Proof. Project the carrier of Definition 6586.1. The accepted packet exposes exactly the rows $M, D, W, R, Q, H, C, P, N$.

The dense witness is read through D before any Real-facing observation is available. The metric ledger M supplies only displayed point and distance observations, and the finite window row W fixes the precision at which those observations are sampled.

The numerical readback is then forced through R and the terminal Real seal Q. These rows provide regular rational comparisons and Real-facing sealing, but no quotient stream equality or selected representative.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since host separability, countable choice, selected dense sequences, quotient equality, selected representatives, and ambient completeness have no coordinate in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6586.3 (EffectiveSeparableMetric L10 source handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow}, \mathsf{RealSeparability}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

in the displayed order. The handoff may export metric distance observations, an explicit dense schedule, finite StreamName windows, regular rational readback, and a Real seal, but it cannot export a host separability theorem, selected dense sequence, quotient stream equality, countable choice, or ambient completeness.

Proof. Start with the NameCert obligations of Definition 6586.2. They require the dense schedule and metric ledger to be visible before a consumer reads any windowed numerical data.

The L10 read then consumes StreamName windows through W and RegSeqRat readback through R. The Real seal Q is downstream of those finite rows, so a Real or Completion route receives scheduled metric data rather than an unstructured separability assertion.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because the carrier lacks choice, quotient, representative-selection, and ambient-completeness coordinates, the handoff remains the finite source route stated above. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ L10 read exposes a MetricUp ledger, an explicit dense schedule, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before it can serve a Real or Completion consumer.

Independence witness: metric, real-separability, streamname, regseqrat, real: none is bijective with $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ because this packet jointly carries the metric ledger, dense schedule, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveSeparableMetric}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveSeparableMetric}^{\uparrow}$ packet is generated from a MetricUp ledger, an explicit dense schedule, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveSeparableMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6586.1, Definition 6586.2, and Definition 6586.3 seed the effective separable metric source route over $\mathsf{Met}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveSeparableMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full separability, countable choice, quotient stream equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-schedule exposure, metric readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveSeparableMetricUp.

Chapter 6587

A Concrete Naming Certificate for $\mathsf{BochnerIntegral}^{\uparrow}$

Following Bochner and Dunford–Schwartz, the $\mathsf{BochnerIntegral}^{\uparrow}$ packet records vector-valued integration as finite BEDC NameCert data over displayed measurable simple-function approximants and Banach norm Cauchy tails. It sits after the measure carrier of chapter 92, the simple-function approximation surface of chapter 5884, the measurable-function rows of chapter 3461, the public integral row of chapter 123, and the Banach endpoint carrier of chapter 90. The packet names the finite route by which a vector-valued integral readback is accepted; it does not import a host almost-everywhere quotient, a completed measure algebra, a selected representative, countable choice, or a global completeness theorem beyond the displayed Banach row.

Definition 6587.1 (Bochner integral carrier)**.** *A* $\mathsf{BochnerIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, X, Y, F, A, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, including the measurable-event carrier, finite value ledger, null-boundary rows, and measure-facing classifier.*
2. *X is the measurable source-domain row recording the displayed domain carrier, measurable preimage tests, and source classifier rows consumed from* $\mathsf{MeasurableFunction}^{\uparrow}$*.*
3. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, including the carried vector endpoints, norm metric, addition and scalar rows, and visible completion ledger.*
4. *F is the displayed vector-valued measurable-function row over X and Y.*
5. *A is the* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *row recording simple vector-valued approximants, measurable fibres, finite range data, and pointwise approximation windows.*
6. *R is the simple-integral readback row, reducing each approximant to finitely many scalar measure values and Banach vector sums before any limit is read.*
7. *Q is the Banach norm Cauchy-tail ledger for the simple-integral sequence, including the precision requests, tail bounds, and norm comparisons used by consumers.*
8. *H records componentwise* hsame *transport for M, X, Y, F, A, R, Q and the structural rows.*

9. *C records displayed* Cont *replay of the route*

$$M, X, Y, F \longmapsto A \longmapsto R \longmapsto Q.$$

10. *P is the* Pkg *provenance over* $\mathsf{BochnerIntegral}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{SimpleFunctionApproximation}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BochnerIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed measure row, measurable source row, Banach target row, measurable-function row, simple-function approximants, finite simple-integral readbacks, Banach Cauchy-tail ledger, transport, replay, provenance, and local naming row.

Theorem 6587.2 (Bochner integral NameCert obligations)**.** *For every accepted* $\mathsf{BochnerIntegral}^{\uparrow}$ *carrier*

$$J = (M, X, Y, F, A, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BochnerIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission, measurable source preservation, Banach target preservation, simple-function approximation stability, simple-integral readback exactness, Banach norm Cauchy-tail visibility, componentwise classifier transport, displayed continuation replay, provenance, and local naming through N.*

Proof. Project the carrier of Definition 6587.1. The analytic rows visible to the certificate are the measure row M, measurable source row X, Banach target row Y, measurable-function row F, approximation row A, simple-integral row R, and Cauchy-tail ledger Q.

Measurable source preservation follows through M, X, F. Every preimage or fibre read is tied to the displayed measurable source rows and the measure carrier; no host sigma-algebra completion or almost-everywhere quotient is introduced.

Approximation and readback preservation follow through A and R. The simple-function approximants have finite range and measurable fibre rows, and each simple integral is read as finitely many measure values and Banach vector sums before the tail ledger is consulted.

The limit-facing obligation is only the Banach norm Cauchy-tail row Q. It records the precision requests and norm comparisons needed for consumers to read the integral endpoint through Y, while H, C, P, N transport, replay, source, and name exactly those displayed rows. □

Theorem 6587.3 (Bochner integral simple-function limit route)**.** *Every* $\mathsf{BochnerIntegral}^{\uparrow}$ *consumer reads the vector-valued integral through the finite route*

$$\mathsf{Measure}^{\uparrow}, \mathsf{MeasurableFunction}^{\uparrow}, \mathsf{Banach}^{\uparrow} \longmapsto \mathsf{SimpleFunctionApproximation}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto \mathsf{BochnerIntegral}^{\uparrow}.$$

For a carrier $J = (M, X, Y, F, A, R, Q, H, C, P, N)$, the consumer must inspect the simple approximants in A, the finite simple-integral readbacks in R, and the Banach norm Cauchy-tail ledger Q before any integral endpoint is named. The route is a NameCert-level limit readback, not a quotient class of functions, a selected representative, or a choice-based convergence theorem.

Proof. Start with the measurable-function row F over the measure and source rows M, X and the target Banach row Y. These rows place the integrand inside the displayed measurable and normed carrier data already available to downstream consumers.

Use the simple-function approximation row A. It supplies finite-range approximants with measurable fibre rows, so each approximant can be reduced by R to finitely many scalar measure values and Banach vector sums.

The passage from approximants to the integral endpoint is then read only through Q. Its norm Cauchy-tail entries bind precision requests to Banach norm comparisons in Y, and the structural rows H, C, P, N preserve that same finite route under transport, replay, provenance, and local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{BochnerIntegral}\uparrow}$.

Constructive story: A $\mathsf{BochnerIntegral}^{\uparrow}$ packet is generated from displayed measure, measurable-function, Banach, simple-function approximation, simple-integral readback, Banach norm tail, transport, continuation, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BochnerIntegral}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6587.1, Definition 6587.2, and Definition 6587.3 seed the vector-valued integration NameCert surface through simple approximants and Banach norm Cauchy tails.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BochnerIntegral}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host almost-everywhere quotient equality, measure completion, selected representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked simple-approximant, simple-integral, and Banach-tail obligations for BEDC.Derived.BochnerIntegralUp.

Chapter 6588

A Concrete Naming Certificate for $\mathsf{PettisMeasurability}^{\uparrow}$

Following Pettis and Dunford–Schwartz on weak and strong measurability for vector-valued functions, the $\mathsf{PettisMeasurability}^{\uparrow}$ packet records the finite BEDC route by which weak scalar measurability, separable-range inspection, and simple-function approximation windows are named before a strong-measurability consumer is admitted. It sits between the weak scalar integration surface of chapter 6638, the norm-approximant route of chapter 6587, the weak-integration boundary of chapter 8555, the measurable source rows of chapter 92, and the Banach endpoint carrier of chapter 90. The packet names a finite measurability bridge; it is not a measure-completion theorem, a quotient almost-everywhere equality, a selected representative, a countable-choice principle, or an ambient separability theorem.

Definition 6588.1 (Pettis measurability carrier)**.** *A* $\mathsf{PettisMeasurability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, X, W, S, A, B, D, R, H, C, Q, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row naming the finite measurable sets, null-window comparisons, and measurable-event family visible to the packet.*
2. *X is the* $\mathsf{Banach}^{\uparrow}$ *target row naming the displayed Banach carrier, norm windows, and finite vector observations.*
3. *W is the weak scalar measurability row, recording that each displayed bounded linear probe reads a scalar measurable shadow through M and X.*
4. *S is the separable-range inspection row, recording the finite dense-code windows and range-localization tests visible to the packet.*
5. *A is the simple-function approximation row, recording finite simple approximants, measurable fibres, and norm-error windows.*
6. *B is the* $\mathsf{BochnerIntegral}^{\uparrow}$ *handoff row, available only after A and the Banach norm windows have been displayed.*
7. *D is the* $\mathsf{DunfordIntegral}^{\uparrow}$ *handoff row, available only through the same weak scalar rows and finite probe windows.*

8. *R is the refusal row excluding host almost-everywhere representatives, quotient function equality, measure completion, countable choice, and ambient separability extraction.*
9. *H records componentwise* hsame *transport for M, X, W, S, A, B, D, R and the local structural rows.*
10. *C records displayed* Cont *replay from weak scalar measurability through separable-range inspection, simple approximation, integration handoffs, refusal, provenance, and naming.*
11. *Q is the* Pkg *provenance over* $\mathsf{PettisMeasurability}^{\uparrow}$, $\mathsf{PettisIntegral}^{\uparrow}$, $\mathsf{BochnerIntegral}^{\uparrow}$, $\mathsf{DunfordIntegral}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PettisMeasurability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for a completed measure algebra, quotient almost-everywhere equality, selected vector representative, countable-choice schedule, or ambient separability theorem.

Theorem 6588.2 (Pettis measurability NameCert obligations)**.** *For every accepted* $\mathsf{PettisMeasurability}^{\uparrow}$ *carrier*

$$P = (M, X, W, S, A, B, D, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PettisMeasurability}^{\uparrow}}$ *surface consists exactly of measure-source exposure through M, Banach endpoint exposure through X, weak scalar measurability through W, separable-range inspection through S, simple-function approximation through A, Bochner and Dunford handoff rows through B and D, refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 6588.1. Its displayed rows are precisely $M, X, W, S, A, B, D, R, H, C, Q, N$.

The analytic rows M, X, W, S, A fix the measurable source, Banach endpoint, weak scalar shadows, separable-range inspection, and simple-function approximation windows. Thus a strong-measurability-facing consumer sees the finite Pettis route before it can ask for a Bochner or Dunford handoff.

The handoff and structural rows preserve that route. The rows B, D are available only after the weak and approximation data are displayed, while R, H, C, Q, N refuse excluded endpoints, transport the same packet, replay the same route, record provenance, and name the local certificate. Since no further coordinate occurs in the carrier, these rows exhaust the NameCert surface. □

Theorem 6588.3 (Pettis measurability strong-window handoff)**.** *Every accepted* $\mathsf{BochnerIntegral}^{\uparrow}$ *or* $\mathsf{DunfordIntegral}^{\uparrow}$ *consumer that cites* $\mathsf{PettisMeasurability}^{\uparrow}$ *for a finite strong-measurability window must first read the route*

$$M, X, W, S, A \longmapsto B, D \longmapsto R, H, C, Q, N.$$

The route supplies weak scalar measurability, separable-range inspection, and simple-function approximation windows before any vector-valued integration handoff is public, and supplies no completed measure algebra, quotient almost-everywhere equality, selected representative, countable-choice schedule, or ambient separability theorem.

Proof. Begin with the NameCert obligations of Definition 6588.2. They place the measure row M, Banach row X, weak scalar row W, separable-range row S, and simple-approximation row A before either integration handoff can be consumed.

For a Bochner-facing read, use B. It is downstream from A and the displayed Banach norm windows, so it receives a finite strong-measurability window rather than a selected almost-everywhere representative.

For a Dunford-facing read, use D. It remains tied to the same weak scalar rows and finite probe windows, so it cannot manufacture a measure completion, quotient equality, countable-choice schedule, or ambient separability theorem.

Finally replay R, H, C, Q, N. These rows refuse the excluded endpoints and preserve the same packet under transport, continuation, provenance, and local naming. Hence the handoff is exactly the displayed finite route. □

Lean4 (statement) `BEDC.Derived.PettisMeasurabilityUp`

Falsifiable prediction: A vector-valued integration read that claims a finite strong-measurability window while bypassing the displayed $\mathsf{Measure}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, weak scalar measurability, separable-range inspection, simple-function approximation, $\mathsf{BochnerIntegral}^{\uparrow}$, $\mathsf{DunfordIntegral}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 6588.1.

Independence witness: pettis-integral, bochner-integral, dunford-integral, measure, banach: $\mathsf{PettisMeasurability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries weak scalar measurability, separable-range inspection, simple-function approximation, integration handoffs, refusal, transport, replay, provenance, and local naming while refusing completed measure, quotient, representative-selection, countable-choice, and ambient separability endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{PettisMeasurability}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PettisMeasurability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6588.1, Definition 6588.2, and Definition 6588.3 seed the finite strong-measurability route over weak scalar rows, separable-range inspection, simple approximants, Bochner and Dunford handoffs, refusal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PettisMeasurability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed measure algebras, quotient almost-everywhere equality, selected vector representatives, countable choice, ambient separability theorems, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak scalar measurability exposure, separable-range inspection, simple-function approximation stability, integration handoff nonescape, refusal, transport, replay, provenance, and local naming for BEDC.Derived.PettisMeasurabilityUp.

Constructive story: A $\mathsf{PettisMeasurability}^{\uparrow}$ packet is generated from Measure, Banach, weak scalar measurability, separable-range inspection, simple-function approximation, Bochner and Dunford integration handoffs, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6589

A Concrete Naming Certificate for $\mathsf{SmithVolterraCantor}^{\uparrow}$

Following Smith and Volterra, the $\mathsf{SmithVolterraCantor}^{\uparrow}$ packet names the finite-prefix interval-removal presentation of a closed nowhere-dense set with positive remaining length. It is routed through the Cantor-style classifier of chapter 1895, the located closed interval rows of chapter 1709, and the dyadic schedule discipline of chapter 1812. The packet records only the finite stage data needed by constructive consumers: no Lebesgue measure completion, arbitrary open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is present.

Definition 6589.1 (Smith-Volterra-Cantor carrier)**.** *A* $\mathsf{SmithVolterraCantor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, R, D, L, C, G, T, H, P, N)$$

with the following displayed rows:

1. *I is the initial located interval row, naming the ambient closed interval as a* $\mathsf{LocatedRealInterval}^{\uparrow}$ *packet.*
2. *R is the finite removal ledger, listing a bounded family of middle intervals already removed at the inspected stage.*
3. *D is the dyadic schedule row, requiring every listed endpoint and every inspected gap width to be read through the same* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *schedule.*
4. *L is the located closed-set row, recording finite separation evidence for points outside the surviving interval family.*
5. *C is the Cantor classifier row, distinguishing survivor, removed-gap, and boundary reads without changing the dyadic schedule.*
6. *G is the finite gap-order row, recording disjointness and endpoint order for the removed intervals visible at the stage.*
7. *T records componentwise* hsame *transport for the interval, schedule, classifier, located-set, and gap rows.*
8. *H records displayed* Cont *replay from a consumer read to the same finite interval-removal route.*
9. *P is the* Pkg *provenance over* BHist*, located interval rows, dyadic schedule rows, classifier rows, transport, replay, and local naming.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SmithVolterraCantor}^{\uparrow}}$ *row.*
The carrier contains no coordinate for completed Lebesgue measure, countable complement enumeration, arbitrary open-cover compactness, or an ambient completeness proof for $\mathsf{Real}^{\uparrow}$.

Theorem 6589.2 (Smith-Volterra-Cantor NameCert obligations)**.** *For an accepted* $\mathsf{SmithVolterraCantor}^{\uparrow}$ *carrier*

$$S = (I, R, D, L, C, G, T, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SmithVolterraCantor}^{\uparrow}}$ *surface consists exactly of carrier admission for the initial located interval, finite removal ledger, dyadic schedule, located closed-set row, survivor classifier, gap-order ledger, componentwise transport, displayed replay, package provenance, and local naming. Any consumer of the packet must read those rows on the same finite route before using the Smith-Volterra-Cantor name.*

Proof. Project the accepted carrier. Rows I, R, D, L, C, G are the only mathematical data rows, so carrier admission exposes the located interval, removed intervals, dyadic schedule, located closed-set evidence, classifier, and finite gap order before the local name is read.

Transport and replay are row-local. The hsame row T preserves the interval-removal ledger componentwise, while H replays a consumer read to the same displayed route rather than to a completed complement or measure object.

Finally P and N provide provenance and local naming. Since the carrier has no measure-completion, arbitrary compactness, countable-choice, or ambient completeness coordinate, the NameCert obligations are exhausted by the displayed finite rows. □

Theorem 6589.3 (Smith-Volterra-Cantor located interval nonescape)**.** *Every accepted* $\mathsf{SmithVolterraCantor}^{\uparrow}$ *consumer that reads survivor membership, removed-gap membership, boundary membership, or local closed-set separation must keep the initial* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, the* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *schedule row, the finite removal ledger, and the classifier row on the same route*

$$S = (I, R, D, L, C, G, T, H, P, N).$$

The read cannot replace finite located interval evidence by Lebesgue measure completion, arbitrary open-cover compactness, selected complement enumeration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the located interval row I and the dyadic schedule row D. They determine the inspected endpoints and finite stage widths before any survivor or gap classifier can be read.

Next attach the finite removal ledger R and the gap-order row G. These rows record exactly the removed middle intervals visible at the stage and keep disjointness and boundary order inside the same dyadic route.

The classifier row C and located closed-set row L then expose only survivor, removed-gap, boundary, and finite separation reads. Transport through T and replay through H preserve those rows, while P, N name the local packet. A consumer seeking measure completion, arbitrary compactness, selected complement enumeration, or Real completeness would need a coordinate absent from S. □

Falsifiable prediction: Every accepted $\mathsf{SmithVolterraCantor}^{\uparrow}$ read reaches a finite removal ledger row and a dyadic schedule row before it exports survivor, removed-gap, boundary, or located closed-set data.

Independence witness: cantorfunction, locatedrealinterval, dyadicintervalnesting: none is bijective to $\mathsf{SmithVolterraCantor}^{\uparrow}$ because this packet jointly carries a positive-length interval-removal

schedule, finite gap ledger, survivor classifier, located closed-set row, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{SmithVolterraCantor}\uparrow}$.

Constructive story: A $\mathsf{SmithVolterraCantor}^{\uparrow}$ packet is generated from a located initial interval, finite interval-removal ledger, dyadic endpoint schedule, located closed-set evidence, survivor classifier, finite gap-order ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmithVolterraCantor}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6589.1, Definition 6589.2, and Definition 6589.3 seed the finite interval-removal NameCert packet for the Smith-Volterra-Cantor construction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SmithVolterraCantor}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lebesgue measure completion, arbitrary open-cover compactness, selected complement enumeration, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite removal-ledger exposure, located closed-set separation, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.SmithVolterraCantorUp.

Chapter 6590

A Concrete Naming Certificate for $\mathsf{MuckenhouptWeight}^{\uparrow}$

Following Muckenhoupt, Stein, and Grafakos, the $\mathsf{MuckenhouptWeight}^{\uparrow}$ packet records the finite A_p weight surface used by weighted maximal-function arguments. It sits after the maximal-function carrier of chapter 3581, the finite covering ledger of chapter 2166, the measure and regular-measure rows of chapter 92 and chapter 2263, and the real and dyadic readback surfaces of chapter 38 and chapter 291. The packet names the displayed local-average rows required to inspect an A_p window without importing a general weighted norm theorem, Lebesgue differentiation, choice-selected covers, or ambient measure completion.

Definition 6590.1 (Muckenhoupt weight carrier)**.** *A* $\mathsf{MuckenhouptWeight}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, M, R, Q, B, A, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric-source row, recording the carried points and the finite ball names read by the weight request.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *row, refined when needed through* $\mathsf{RadonMeasure}^{\uparrow}$ *regular finite-measure windows.*
3. *R is the positive-radius and dyadic-scale row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons for the inspected balls.*
4. *Q is the exponent row. It records the displayed p and conjugate-budget comparisons used by the A_p ledger.*
5. *B is the finite ball-window row, listing the balls on which weight and reciprocal-power averages are read.*
6. *A is the weight-average row. For every displayed ball it records the finite* $\mathsf{Real}^{\uparrow}$ *readback of the weight average.*
7. *D is the reciprocal-power average row, recording the finite readback needed for the conjugate side of the A_p test.*
8. *L is the A_p ledger row. It stores the finite comparison budget showing that the product of the displayed averages stays below the named local constant on each inspected ball.*

9. *E is the maximal-consumer boundary row, recording which* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *request may read the weight packet and which rows remain hidden from that consumer.*
10. *H records componentwise* hsame *transport for $X, M, R, Q, B, A, D, L, E, C, P, N$.*
11. *C records displayed* Cont *replay of the route*

$$\mathsf{Measure}^{\uparrow}, \mathsf{RadonMeasure}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto A, D, L \longmapsto \mathsf{MuckenhouptWeight}^{\uparrow}.$$

12. *P is the* Pkg *provenance over* $\mathsf{MuckenhouptWeight}^{\uparrow}$, $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MuckenhouptWeight}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, R, Q, B, A, D, L, E, H, C, P, N$. It has no coordinate for a global weighted norm theorem, a Lebesgue differentiation theorem, a choice-selected cover, quotient function equality, or an ambient measure-completion object.

Theorem 6590.2 (Muckenhoupt weight NameCert obligations)**.** *For every accepted* $\mathsf{MuckenhouptWeight}^{\uparrow}$ *carrier*

$$W = (X, M, R, Q, B, A, D, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MuckenhouptWeight}^{\uparrow}}$ *surface consists exactly of carrier admission, measure and Radon-window visibility, dyadic radius comparison, exponent-budget visibility, ball-window coverage, weight-average readback, reciprocal-power average readback, A_p ledger comparison, maximal-consumer boundary exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6590.1. The source, measure, dyadic-radius, exponent, ball-window, average, reciprocal-average, ledger, and consumer-boundary rows are the only analytic rows visible to the local certificate.

Measure visibility is read through M and, for regular finite windows, through the $\mathsf{RadonMeasure}^{\uparrow}$ provenance in P. Radius and scale comparisons are read through R, so the local averages in A and D are attached to displayed finite ball windows rather than to a host family of balls.

The A_p obligation is exactly the ledger row L. It compares the displayed weight average and reciprocal-power average against the local bound named for the same ball window and exponent budget.

Finally E, H, C, P, N expose the maximal-consumer boundary, transport, replay, source, and local name for the same finite route. Since no coordinate stores a global weighted norm theorem, Lebesgue differentiation, a choice-selected cover, quotient function equality, or measure completion, the listed rows exhaust the NameCert surface. □

Theorem 6590.3 (Muckenhoupt weight maximal consumer boundary)**.** *Every* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *consumer of an accepted* $\mathsf{MuckenhouptWeight}^{\uparrow}$ *packet reads only the displayed finite rows $M, R, Q, B, A, D, L, E, H, C$. The consumer may inspect finite weighted averages, reciprocal-power averages, and the A_p comparison ledger on named ball windows, but it does not receive a general weighted norm theorem, a Lebesgue differentiation theorem, a choice-selected cover, quotient function equality, or ambient measure completion.*

Proof. Start with the boundary row E. It records the exact maximal-function request allowed to inspect the weight packet and points that request back to the same ball windows B, dyadic-radius row R, and measure row M.

The numerical data available to the consumer are the finite readbacks A and D together with the ledger L. These rows certify only the displayed A_p comparison on inspected balls, and the replay row C keeps the route inside $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readbacks.

The structural rows H, P, N transport, source, and name the same packet. Because the carrier lacks coordinates for norm inequalities, differentiation, selected covers, quotient equality, and measure completion, the maximal consumer boundary exports exactly the finite rows stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{MuckenhouptWeight}^{\uparrow}}$.

Constructive story: A $\mathsf{MuckenhouptWeight}^{\uparrow}$ packet is generated from metric source, measure, RadonMeasure, dyadic radius, exponent, ball-window, weight-average, reciprocal-average, Ap ledger, maximal-consumer boundary, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MuckenhouptWeight}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6590.1, Definition 6590.2, and Definition 6590.3 seed the finite A_p weight NameCert surface for maximal-function consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MuckenhouptWeight}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a general weighted norm theorem, Lebesgue differentiation, choice-selected covers, quotient function equality, ambient measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average readbacks, Ap ledger comparison, and maximal-consumer boundary for BEDC.Derived.MuckenhouptWeightUp.

Chapter 6591

A Concrete Naming Certificate for $\mathsf{KuratowskiConvergence}^{\uparrow}$

Following Kuratowski and Painleve–Kuratowski hyperspace convergence, the $\mathsf{KuratowskiConvergence}^{\uparrow}$ packet records finite BEDC convergence data for displayed closed-set sequences in a metric hyperspace. It sits over the metric source of chapter 57, the located-set source of chapter 1541, the projection boundary of chapter 2356, the compact metric hyperspace surface of chapter 5184, the Kuratowski hyperspace carrier of chapter 5944, and the hit–miss convergence route of chapter 3880. The packet names only displayed lower-hit, upper-miss, projection, and Real-seal rows; it does not quotient closed sets, select hyperspace representatives, invoke choice, or import ambient completeness.

Definition 6591.1 (Kuratowski convergence carrier)**.** *A* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, Q, H, R, L, U, W, E, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the finite distance observations used by all hit, miss, and projection reads.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *closed-set source row, naming each inspected set by located-radius observations rather than by a host powerset.*
3. *Q is the* $\mathsf{MetricProjection}^{\uparrow}$ *boundary row that supplies point-to-set distance and nearest-boundary evidence only through displayed projection data.*
4. *H is the* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *row carrying finite Hausdorff and Vietoris windows for compact hyperspace reads.*
5. *R is the* $\mathsf{KuratowskiHyperspace}^{\uparrow}$ *row that compares hyperspace and Kuratowski-profile classifiers through finite metric data.*
6. *L is the lower-hit ledger: each entry names a displayed source point, a finite stage window, and a tolerance row witnessing that the inspected closed-set sequence meets the requested neighborhood.*
7. *U is the upper-miss ledger: each entry names a finite cover or finite complement test witnessing that, beyond the displayed stage window, no inspected set escapes the requested metric neighborhood.*

8. *W is the* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ *hit–miss window row tying L and U to the sibling convergence classifier.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row used by Real and completion consumers to read tolerances without quotienting closed sets.*
10. *T, C, P, N are respectively componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{KuratowskiConvergence}^{\uparrow}}$ *naming row.*

The carrier has no coordinate for an ambient powerset of closed subsets, quotient equality of closed sets, a selected limit set, countable choice, or an untracked hyperspace completion.

Theorem 6591.2 (Kuratowski convergence NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *carrier*

$$K = (M, A, Q, H, R, L, U, W, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiConvergence}^{\uparrow}}$ *surface consists exactly of metric source admission, located-set admission, metric-projection boundary admission, compact metric hyperspace admission, Kuratowski hyperspace profile comparison, lower-hit ledger admission, upper-miss ledger admission, Painleve–Kuratowski hit–miss synchronization, Real and regular-rational tolerance sealing, componentwise transport, displayed replay, provenance, and local naming. Every convergence-facing consumer must read through these rows before it may inspect a limit claim.*

Proof. Project the carrier of Definition 6591.1. The metric source, located-set source, projection row, compact hyperspace row, Kuratowski profile row, lower ledger, upper ledger, hit–miss window, and Real seal are the only mathematical coordinates.

The lower-hit and upper-miss obligations are separate rows. A lower request reads a point, stage window, and tolerance from L; an upper request reads a finite cover or complement test from U. The synchronization row W is the only place where those two ledgers are compared.

The structural rows T, C, P, N preserve and name that same route. Since the carrier has no powerset, quotient, selected-limit, or choice coordinate, no local NameCert obligation can be stored outside the listed finite rows. □

Theorem 6591.3 (Kuratowski convergence metric hyperspace handoff)**.** *Every compact-hyperspace or projection-facing read exported by an accepted* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{LocatedSet}^{\uparrow}, \mathsf{MetricProjection}^{\uparrow}, \mathsf{CompactMetricHyperspace}^{\uparrow}, \mathsf{KuratowskiHyperspace}^{\uparrow}, \mathsf{PainleveKuratowskiConver}$$

The handoff exposes finite Hausdorff windows, Kuratowski-profile comparisons, and projection-distance rows before any Real or completion consumer reads the convergence packet. It exports no quotient closed-set equality, selected hyperspace representative, selected limit set, or ambient compactness theorem.

Proof. Begin with a compact-hyperspace read. The carrier forces the consumer to inspect the metric source M, located-set row A, projection boundary Q, compact hyperspace row H, and Kuratowski profile row R before the convergence windows L, U, W are available.

The lower-hit ledger and upper-miss ledger then provide the finite convergence evidence. Their comparison is read only through W, which is already tied to the sibling Painleve–Kuratowski hit–miss surface.

Finally the tolerance readback is sealed by E and replayed through T, C, P, N. Thus the handoff is a finite BEDC route through the named sibling anchors, not a route through quotient closed sets or host hyperspace objects. □

Theorem 6591.4 (Kuratowski convergence Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance read of an accepted* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *carrier is downstream of the lower-hit ledger* L*, the upper-miss ledger* U*, and the hit–miss synchronization row* W*. The Real seal records only the displayed tolerance values used by those finite windows; it cannot be used to recover a host limit set, quotient equality of closed sets, countable-choice schedule, or ambient hyperspace completion.*

Proof. Read the Real-facing coordinate E. By Definition 6591.1, it is attached to the convergence packet only after the lower-hit and upper-miss ledgers have been displayed and synchronized.

Apply Definition 6591.2. The local naming surface sends every tolerance read through $M, A, Q, H, R, L, U, W, E$, so the Real row cannot be detached from the finite convergence evidence.

The nonescape claim follows from absence of carrier coordinates. There is no field for a selected closed-set limit, quotient equality, choice schedule, or ambient hyperspace completion, hence the Real seal can certify only the displayed tolerance route. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiConvergence}^{\uparrow}$ packet is generated from MetricUp source rows, LocatedSetUp closed-set rows, MetricProjectionUp distance-boundary rows, CompactMetricHyperspaceUp finite Hausdorff and Vietoris windows, KuratowskiHyperspaceUp profile comparison, PainleveKuratowskiConvergenceUp hit–miss synchronization, Real and RegSeqRat tolerance seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6591.1, Definition 6591.2, Definition 6591.3, and Definition 6591.4 seed the finite metric hyperspace convergence surface over displayed closed-set data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiConvergence}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of closed sets, host powersets, selected limit sets, countable choice, ambient hyperspace completion, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-hit coverage, upper-miss coverage, metric hyperspace handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KuratowskiConvergenceUp.

Lean4 (statement) `BEDC.Derived.KuratowskiConvergenceUp`

Chapter 6592

A Concrete Naming Certificate for $\mathsf{ChebyshevInequality}^{\uparrow}$

Following Chebyshev and Kolmogorov, the $\mathsf{ChebyshevInequality}^{\uparrow}$ packet records the finite probability tail-bound surface used before weak-law consumers read a variance-controlled estimate. It sits after the normalized event rows of chapter 184, the measurable random-variable surface of chapter 185, the finite measure ledger of chapter 6127, and the terminal real-name seal of chapter 38. It is the standalone tail-bound carrier consumed by chapter 4373, not a limit theorem, countable-choice selector, almost-sure convergence result, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6592.1 (Chebyshev inequality carrier)**.** *A* $\mathsf{ChebyshevInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, M, X, \mu, V, \epsilon, T, R, H, K, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the event classifier and total-mass normalization.*
2. *M is the* $\mathsf{FiniteMeasure}^{\uparrow}$ *measure row for the exceptional event and its complement inside the displayed probability source.*
3. *X is the* $\mathsf{RandomVar}^{\uparrow}$ *row whose finite observation windows are inspected over P.*
4. *μ is the displayed center row for the same random-variable surface.*
5. *V is the finite variance or second-moment bound row consumed by the tail estimate.*
6. *ϵ is the positive* $\mathsf{Real}^{\uparrow}$ *tolerance seal naming the requested separation from the center.*
7. *T is the displayed tail-event row, recording the event where the random-variable observation differs from μ by more than ϵ.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *probability-bound row reached only after $P, M, X, \mu, V, \epsilon, T$ have been displayed.*
9. *H stores componentwise* hsame *transport for the source, measure, random-variable, center, variance, tolerance, tail-event, probability-bound, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay of the finite tail-bound route.*
11. *Q is the* Pkg *provenance over* $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{FiniteMeasure}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ChebyshevInequality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $P, M, X, \mu, V, \epsilon, T, R, H, K, Q, N$. It has no coordinate for an infinite sequence of variables, almost-sure convergence, selected subsequence, central-limit normalization, host measure algebra, or ambient real completeness.

Theorem 6592.2 (Chebyshev inequality NameCert obligations)**.** *For every accepted* $\mathsf{ChebyshevInequality}^{\uparrow}$ *carrier* $C = (P, M, X, \mu, V, \epsilon, T, R, H, K, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChebyshevInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed probability source, finite-measure row, random-variable row, center row, variance-bound row, positive tolerance seal, tail-event row, probability-bound row, transport, replay, provenance, and local naming;*
2. *source and event stability, requiring M and T to remain attached to the event classifier displayed by P;*
3. *random-variable stability, requiring every tail-event read to consume the same* $\mathsf{RandomVar}^{\uparrow}$ *row X and center row μ;*
4. *variance-budget stability, requiring V to be displayed before the tail probability bound is read;*
5. *tolerance seal stability, requiring ϵ to be a displayed* $\mathsf{Real}^{\uparrow}$ *positive tolerance before T and R are consumed;*
6. *probability-bound nonescape, requiring R to seal only the displayed Chebyshev tail estimate and not an infinite-limit, selected-subsequence, host-measure, or ambient-completeness claim;*
7. *componentwise transport, displayed replay, package provenance, and local naming through H, K, Q, N.*

These obligations certify only the finite probability tail-bound route. They do not close a weak law, almost-sure convergence, a central limit theorem, countable choice, host probability limits, or Lean verification.

Proof. Project the carrier of Definition 6592.1. The accepted rows are exactly the probability source, finite-measure row, random-variable row, center row, variance-bound row, tolerance seal, tail-event row, probability-bound row, and the structural rows.

Source and event stability are read from P, M, T. The event classifier comes from P, the finite measure row M attaches measure data to that source, and T names only the displayed exceptional event.

Random-variable and variance-budget stability are read from X, μ, V. A tail estimate can be consumed only after the variable, center, and variance-bound rows have been displayed.

The endpoint R is downstream from the positive tolerance seal ϵ and from the tail-event row. Since the carrier has no coordinate for infinite limits, selected subsequences, host measure algebra, or ambient completeness, R cannot export any of those objects.

The rows H, K, Q, N preserve, replay, source, and name the same finite route. Hence the listed obligations exhaust the local NameCert surface. □

Theorem 6592.3 (Chebyshev tail-bound nonescape)**.** *Every accepted* $\mathsf{ChebyshevInequality}^{\uparrow}$ *tail-bound read factors through*

$$P \longmapsto M \longmapsto X \longmapsto \mu \longmapsto V \longmapsto \epsilon \longmapsto T \longmapsto R \longmapsto H, K, Q, N$$

before any downstream probability consumer receives the estimate. The route exposes a finite probability source, finite-measure event row, random-variable row, center, variance budget, positive tolerance, tail event, real probability bound, transport, replay, provenance, and local naming. It

cannot be replaced by a host law of large numbers, almost-sure convergence, central-limit normalization, countable-choice subsequence selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a tail-bound request from an accepted carrier. Projection gives the source rows P, M and no independent host probability-space coordinate.

The request next reads X, μ, V, ϵ. These rows display the random variable, center, variance budget, and positive real tolerance before the exceptional event can be measured.

Only after those rows are visible can T and R be consumed as the finite Chebyshev tail estimate. The structural rows H, K, Q, N then transport, replay, package, and name the same route for downstream consumers.

Since every readable coordinate has already been listed, the route has no path to almost-sure convergence, a central-limit theorem, selected subsequences, host probability limits, or ambient real completeness. □

Falsifiable prediction: Every $\mathsf{ChebyshevInequality}^{\uparrow}$ tail-bound read factors through a ProbSpace source row, a FiniteMeasure event row, a RandomVar observation row, a displayed center, a variance-bound row, a positive Real tolerance, a tail-event row, a Real probability-bound row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: probspace, finite-measure, randomvar, real, weak-law-large-numbers: $\mathsf{ChebyshevInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a probability source, finite-measure event row, random-variable observation, center, variance budget, positive tolerance, tail event, probability-bound seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ChebyshevInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{ChebyshevInequality}^{\uparrow}$ packet is generated from a probability source row, a finite-measure event row, a random-variable row, a displayed center, a variance or second-moment bound row, a positive RealUp tolerance, a tail-event row, a RealUp probability-bound row, componentwise hsame transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChebyshevInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6592.1, Definition 6592.2, and Definition 6592.3 seed the finite probability tail-bound route through ProbSpace, FiniteMeasure, RandomVar, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChebyshevInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak laws, almost-sure convergence, central-limit normalization, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and event stability, random-variable stability, variance-budget stability, tolerance seal stability, probability-bound nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ChebyshevInequalityUp.

Chapter 6593

A Concrete Naming Certificate for $\mathsf{CauchyStabilityModulus}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy moduli, the $\mathsf{CauchyStabilityModulus}^{\uparrow}$ packet records the finite tolerance-window object that lets Real and completion consumers reuse one displayed Cauchy stability route. It sits over the Cauchy evidence surface of chapter 3413, the finite observation spine of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite stability modulus; it is not quotient equality of Cauchy streams, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6593.1 (Cauchy stability modulus carrier)**.** *A* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, D, R, W, H, T, P, N)$$

with the following displayed rows:

1. *C is the Cauchy source row, carrying the finite comparison data and the source history whose stability is being read.*
2. *D is the dyadic tolerance row, recording the displayed tolerance request and any smaller tolerance admitted by the same packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window row that turns the Cauchy comparison into regular rational readback at the requested tolerance.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row naming the finite stream windows inspected by R.*
5. *H records componentwise* hsame *transport for the Cauchy source, dyadic tolerance, regular window, stream observation, replay, provenance, and local naming rows.*
6. *T records displayed* Cont *replay from the tolerance request through Cauchy comparison, finite stream windows, and regular rational readback.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyStabilityModulus}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyStabilityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through C, D, R, W, H, T, P, N. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, hidden completion operators, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyStabilityModulusUp`

Theorem 6593.2 (Cauchy stability modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *carrier*

$$S = (C, D, R, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyStabilityModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy source exposure, dyadic tolerance admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readback,* $\mathsf{StreamName}^{\uparrow}$ *observation replay, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for quotient stream equality, selected limits, countable choice, host real equality, hidden completion operators, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6593.1. The accepted packet exposes exactly the Cauchy source row, dyadic tolerance row, regular rational finite-window row, stream observation row, transport, replay, provenance, and local naming.

The classifier reads tolerance data only through D. The Cauchy source row C supplies the comparison evidence, while R and W record the finite regular-rational and stream-window observations made at that tolerance.

The rows H and T preserve the same route under transport and replay. Provenance and local naming are then supplied by P and N.

Since the carrier has no additional coordinate, no accepted read can import quotient stream equality, selected limits, countable choice, host real equality, hidden completion operators, or ambient completeness. □

Theorem 6593.3 (Cauchy stability modulus finite-window stability)**.** *Let* $S = (C, D, R, W, H, T, P, N)$ *be an accepted* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *carrier. If* D' *is a displayed dyadic tolerance row obtained by shrinking the tolerance named by* D*, and the same Cauchy source row* C *supplies the comparison witnesses needed for* D' *through the transported finite windows* R, W, H*, then the packet accepts the corresponding finite-window Cauchy stability read. The accepted read remains inside* C, D', R, W, H, T, P, N *and exports no quotient stream equality, selected limit, countable choice, host real equality, hidden completion operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the accepted carrier and replace only the dyadic tolerance row by the displayed smaller row D'. The source Cauchy row C is unchanged, so the comparison witnesses still come from the same finite packet.

The rows R and W are read at the transported finite windows. Componentwise hsame transport in H keeps the regular rational readback and stream observations attached to the same Cauchy source while the tolerance is refined.

The replay row T then repeats the tolerance-to-window route for D'. Provenance and local naming through P, N identify this as the same $\mathsf{CauchyStabilityModulus}^{\uparrow}$ certificate surface.

Thus shrinking a displayed dyadic tolerance preserves accepted finite-window stability, and all excluded completion or quotient data remain outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyStabilityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyStabilityModulus}^{\uparrow}$ packet is generated from Cauchy source rows, dyadic tolerance rows, RegSeqRat finite-window rows, StreamName observation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyStabilityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6593.1, Definition 6593.2, and Definition 6593.3 seed the finite tolerance-window stability route for Cauchy, RegSeqRat, StreamName, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyStabilityModulus}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.CauchyStabilityModulusUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy streams, selected limits, countable choice, host real equality, hidden completion operators, ambient $\mathrm{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, tolerance-shrink stability, replay exactness, provenance, and ledger nonescape for BEDC.Derived.CauchyStabilityModulusUp.

Chapter 6594

A Concrete Naming Certificate for $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, a regular Cauchy name carries its own visible rate of stability rather than asking the ambient real line to choose a limit. The $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ packet specializes the finite tolerance-window route of chapter 6593 to regular rational names. It reads the $\mathsf{StreamName}^{\uparrow}$ finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-shrink ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal as displayed rows of one local NameCert object.

Definition 6594.1 (Regular Cauchy stability modulus carrier)**.** *A* $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (C, S, R, D, E, H, T, P, N)$$

with the following rows:

1. *C is the* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *source row, naming the finite stability-modulus data being reused.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of finite observation windows inspected by the stability read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets those windows as a regular rational Cauchy name.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-shrink row, recording the displayed move from a requested tolerance to a smaller admitted tolerance.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular Cauchy window has been read.*
6. *H records componentwise* hsame *transport for the source modulus, stream windows, regular rational readback, dyadic tolerance, real seal, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from the source modulus through the finite windows, regular rational readback, dyadic shrink, and terminal seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$, $\mathsf{CauchyStabilityModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyStabilityModulus}^{\uparrow}}$ *row.*

The classifier compares packets only through $C, S, R, D, E, H, T, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6594.2 (Regular Cauchy stability modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ *carrier*

$$M = (C, S, R, D, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyStabilityModulus}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *source exposure,* $\mathsf{StreamName}^{\uparrow}$ *finite-window coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-shrink compatibility, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every refused endpoint remains outside the carrier: quotient streams, selected limits, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6594.1. The accepted packet exposes the source stability modulus C, the finite windows S, the regular rational readback R, the dyadic shrink row D, and the terminal real seal E before any structural rows are read.

The source row is the already named $\mathsf{CauchyStabilityModulus}^{\uparrow}$ route of Definition 6593.2. The specialization to regular Cauchy data is exactly the requirement that S and R are displayed before D shrinks the tolerance and before E seals the real-facing read.

The transport row H and replay row T preserve the same finite path under componentwise equality and continuity replay. The provenance row P and local name row N then package the displayed route as a local NameCert surface.

Since no carrier coordinate records a quotient stream, selected limit, countable choice schedule, trichotomy test, host real equality, or ambient completeness principle, no accepted read can export any of those endpoints. □

Theorem 6594.3 (Regular Cauchy stability modulus Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ *packet must first read the source* $\mathsf{CauchyStabilityModulus}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-shrink row before the terminal seal E is available. The handoff to* $\mathsf{Real}^{\uparrow}$ *is therefore the finite route*

$$\mathsf{CauchyStabilityModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

transported by hsame*, replayed by* Cont*, and named by P, N.*

Proof. Start with the accepted carrier $M = (C, S, R, D, E, H, T, P, N)$. A consumer that reads E must have already projected the preceding rows, because the seal row is part of the same displayed finite packet and has no independent entry point.

The rows S and R make the regular Cauchy read finite-windowed. The row D records the dyadic tolerance shrink needed before the regular rational read can be sealed as a real-facing stability read.

Componentwise transport and displayed replay are supplied by H and T. Provenance and local naming through P, N identify the route as a $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ handoff rather than a host real equality or ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyStabilityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyStabilityModulus}^{\uparrow}$ packet is generated from a CauchyStabilityModulus source, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance shrink, Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyStabilityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6594.1, Definition 6594.2, and Definition 6594.3 seed the regular-Cauchy stability-modulus route from finite windows to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyStabilityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, axiom audit, or bridge checking.

Upgrade path: Obligation closure requires a Lean carrier and checked NameCert obligations for source exposure, finite-window readback, dyadic tolerance shrink, seal handoff, transport, replay, provenance, and ledger refusal.

Chapter 6595

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$

Following the Poincare disk model and the Denjoy–Wolff boundary principle, the $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ packet records boundary-fixed disk transport as finite NameCert data. It realizes the boundary-fixed disk transport schema of Definition 9463.10 through the isometry rows of chapter 5159, the boundary refusal rows of chapter 6142, and the ray-boundary rows of chapter 6470. The packet names a finite disk-map and boundary-transport carrier; it is not a completed hyperbolic plane, a selected ideal endpoint, quotient boundary equality, or a global dynamical theorem.

Definition 6595.1 (Hyperbolic boundary transport carrier)**.** *A* $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, R, P, H, T, M, N)$$

with the following displayed rows:

1. *D is the finite disk-map row, recording the displayed source and target disk observations and the map applied between them.*
2. *F is the boundary far-end row, recording the boundary commitment preserved by the transport.*
3. *R is the reversible transport row, recording the displayed inverse route and the finite two-sided replay checks.*
4. *P is the provenance row over the Poincare disk, Denjoy–Wolff boundary reading,* $\mathsf{HyperbolicIsometryGroup}^{\uparrow}$, $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
5. *H records componentwise* hsame *transport for disk maps, boundary far-end rows, reversible transport, provenance, metric noncollapse, replay, and local naming.*
6. *T records displayed* Cont *replay from disk-map inspection through boundary preservation, reversible transport, metric noncollapse, provenance, and local naming.*
7. *M is the metric noncollapse row, refusing any read that turns boundary-fixed transport into an interior-center fixed point or a collapsed zero-distance transport.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryTransport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, F, R, P, H, T, M, N. It has no coordinate for a completed disk, selected ideal endpoint, quotient boundary equality, hidden global hyperbolic metric, or physical observer law.

Lean4 (definition) `BEDC.Derived.HyperbolicBoundaryTransportUp`

Theorem 6595.2 (Hyperbolic boundary transport NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ *carrier*

$$B = (D, F, R, P, H, T, M, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryTransport}^{\uparrow}}$ *surface consists exactly of finite disk-map admission, boundary far-end preservation, reversible transport replay, provenance, componentwise* hsame *transport, displayed* Cont *replay, metric noncollapse, local naming, and refusal of completed hyperbolic planes, selected ideal endpoints, quotient boundary equality, hidden global metrics, and physical observer laws.*

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryTransportUp.HyperbolicBoundaryTransportTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6595.1. The disk-map row D, boundary row F, reversible transport row R, provenance row P, transport row H, replay row T, metric noncollapse row M, and naming row N are the whole accepted packet.

Boundary preservation is read from F after the disk-map row D has been displayed. Reversibility is read from R and replayed through T, so the consumer sees finite two-sided route data rather than a host global transformation.

Metric noncollapse is the row M. It keeps the boundary-fixed route from becoming an interior-center fixed point or a collapsed metric read.

Finally, P, H, T, N source, transport, replay, and name the same finite surface. Since the excluded objects are absent from every carrier coordinate, they cannot enter the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ read is refuted if it exports boundary-fixed transport without the disk-map row, boundary far-end row, reversible transport row, metric noncollapse row, hsame transport, Cont replay, provenance, and local NameCert row.

Independence witness: hyperbolic_isometry_group, boundary_far_end_noncenter, gromov_boundary: the isometry sibling carries group and metric-preservation rows, the noncenter sibling carries boundary refusal rows, and the Gromov-boundary sibling carries ray-boundary rows, but only $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ binds disk map, fixed boundary far end, reversible transport, provenance, noncollapse, replay, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryTransport}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ packet is generated from disk-map rows, boundary far-end rows, reversible transport rows, provenance, componentwise hsame transport, displayed Cont replay, metric noncollapse rows, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryTransport}^{\uparrow}}$, seed)

Scope closed: Definition 6595.1, Definition 6595.2, the falsifiable prediction, and the independence witness seed the finite boundary-fixed disk transport route over disk-map, boundary, reversible transport, provenance, noncollapse, replay, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryTransport}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HyperbolicBoundaryTransportUp.HyperbolicBoundaryTransportTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic planes, selected ideal endpoints, quotient boundary equality, hidden global hyperbolic metrics, physical observer laws, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, boundary preservation, reversible transport replay, provenance, metric noncollapse, transport, replay, and local naming for BEDC.Derived.HyperbolicBoundaryTransportUp.

Chapter 6596

A Concrete Naming Certificate for $\mathsf{EffectiveCompactness}^{\uparrow}$

Following Bishop and Bridges on constructive compactness, the $\mathsf{EffectiveCompactness}^{\uparrow}$ packet records compactness by displayed finite ε-nets and located metric coverage rather than by an ambient open-cover principle. It sits over the compact metric source of chapter 129, the total-bounded schedule of chapter 127, the metric carrier of chapter 57, and the rational tolerance rows of chapter 37. The packet isolates the finite-net compactness object used by the compact-uniform-continuity route of chapter 9237: a consumer receives located finite nets, rational tolerance bookkeeping, and a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff, not a selected open subcover, quotient point equality, countable choice, or ambient completeness theorem.

Definition 6596.1 (Effective compactness carrier)**.** *An* $\mathsf{EffectiveCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, T, F, L, R, Q, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the distance comparisons and located balls admitted by the named metric carrier.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *schedule row, assigning a finite-net request to each positive rational tolerance.*
3. *F is the finite center bundle returned by the schedule for the current tolerance request.*
4. *L is the located coverage row, recording for each inspected point the displayed center and metric comparison witnessing membership in the requested ball.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *tolerance ledger, carrying positivity, halving, slack, and comparison rows used by the finite-net read.*
6. *Q is the compact metric provenance row tying M, T, F, L, R to the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source.*
7. *U is the tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed by compact-uniform-continuity arguments after the finite-net coverage has been displayed.*
8. *H records componentwise* hsame *transport for M, T, F, L, R, Q, U.*
9. *C records displayed* Cont *replay from a tolerance request through the finite-net schedule, located coverage row, rational comparison ledger, and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{EffectiveCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCompactness}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, selected subcovers outside F, quotient equality of points, countable choice, host completeness, or an external modulus selector.

Theorem 6596.2 (Effective compactness NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCompactness}^{\uparrow}$ *carrier*

$$E = (M, T, F, L, R, Q, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCompactness}^{\uparrow}}$ *surface consists exactly of metric-source exposure, total-bounded schedule exposure, finite center-bundle admission, located coverage readback, rational tolerance comparison, compact-metric provenance, tolerance-only uniform-modulus handoff, componentwise transport, displayed replay, package provenance, and local naming. Every compactness-facing read must pass through the displayed finite-net rows T, F, L, R before it may consume the handoff row U.*

Proof. Project the carrier of Definition 6596.1. Its visible rows are exactly the metric source, total-bounded schedule, finite center bundle, located coverage row, rational tolerance ledger, compact-metric provenance, uniform-modulus handoff, and structural rows.

Finite-net exposure is supplied by T, F, L, R. The schedule row T receives the rational tolerance, the finite bundle F lists the centers, the located row L records coverage, and the rational ledger R records the positive slack and comparisons used by that read.

The compact source and downstream handoff are confined to Q and U. The row Q ties the read to the compact metric source, and U exposes only the tolerance-only modulus output available after the finite-net rows have been displayed.

Transport, replay, provenance, and naming are supplied by H, C, P, N. Since none of the excluded open-cover, choice, quotient, completeness, or external-selector objects occurs in the packet, the displayed rows exhaust the local certificate surface. □

Theorem 6596.3 (Effective compactness finite-net modulus)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{EffectiveCompactness}^{\uparrow}$ *carrier obtains its tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *row through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

Concretely, the consumer first reads the metric source row M, the finite-net schedule T, the center bundle F, the located coverage row L, the rational tolerance ledger R, and the compact-metric provenance row Q before it may use the modulus handoff U. The route exports no open-cover compactness, selected subcover outside the bundle, quotient point equality, countable choice, host completeness, or external modulus selector.

Proof. Start with an accepted carrier $E = (M, T, F, L, R, Q, U, H, C, P, N)$. The only metric-facing source is M, and the only finite-net schedule is T.

Read the finite-net data. The center bundle F is produced for the requested tolerance, the located coverage row L records the displayed metric comparisons, and R carries the rational slack needed to compare the requested tolerance with the finite coverage radius.

The compact-metric provenance row Q ties those rows to the compact metric source used in the route of chapter 9237. Only after that finite coverage read is displayed does the handoff row U expose the tolerance-only uniform-modulus datum.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. Because the carrier contains no open-cover, selected-subcover, quotient, countable-choice, completeness, or external-selector coordinate, the finite-net route is the whole modulus-facing export. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCompactness}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCompactness}^{\uparrow}$ packet is generated from a MetricUp source row, a TotallyBoundedUp finite-net schedule, a finite center bundle, located coverage rows, RatUp tolerance comparisons, CompactMetricUp provenance, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6596.1, Definition 6596.2, and Definition 6596.3 seed the explicit finite-net compactness object consumed by compact-uniform-continuity routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected subcovers outside the displayed finite bundle, quotient point equality, countable choice, host completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net schedule exposure, located coverage readback, rational tolerance comparison, compact-metric provenance, uniform-modulus handoff, and ledger non-escape for BEDC.Derived.EffectiveCompactnessUp.

Chapter 6597

A Concrete Naming Certificate for CompactlySupportedRealFunction$^{\uparrow}$

Following Bourbaki and Rudin, a compactly supported real function is a real-valued function whose nonzero behavior is confined to a compact portion of the real line. The CompactlySupportedRealFunction$^{\uparrow}$ packet records this classical object as finite NameCert data over the constructive real and completion frontier. It consumes the real terminal surface of chapter 38, the stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic source core of chapter 291, the continuous-map carrier of chapter 130, and the closed bounded interval route of chapter 1591. The packet names finite support, readback, and continuity obligations only; it does not assert full function-space completeness, host extensional equality of functions, a global Heine–Borel theorem, or choice-selected support endpoints.

Definition 6597.1 (Compactly supported real function carrier)**.** *A* CompactlySupportedRealFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, R, S, K, H, T, C, P, N)$$

with the following displayed rows:

1. *G is the function graph window row. It records finite stream observations of input and output names and exposes graph data only through the same* StreamName$^{\uparrow}$ *requests.*
2. *R is the* Real$^{\uparrow}$ *readback row. It seals output observations as real names only after the stream, regular rational, and dyadic source rows have been displayed.*
3. *S is the support-bound certificate. It records a closed bounded interval outside which every accepted graph-window read is certified as the zero real output.*
4. *K is the compact and closed-bounded support handoff row. It reads S through the* ClosedBoundedInterval$^{\uparrow}$ *route and exposes compact-support data as a finite interval handoff rather than as open-cover compactness.*
5. *H stores componentwise* hsame *transport for graph windows, Real readback, support bounds, compact interval handoff, continuity replay, provenance, and the local naming row.*
6. *T stores* Cont *replay for the route from stream observations to regular rational and dyadic reads, from those reads to the Real seal, and from the support-bound row to continuous-map consumers.*
7. *C is the* ContinuousMap$^{\uparrow}$ *compatibility row. It records that a consumer may read continuity only through the displayed graph-window, Real readback, support-bound, and support-handoff rows.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactlySupportedRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, R, S, K, H, T, C, P, N$. It has no coordinate for host function equality, total function-space completeness, open-cover Heine–Borel compactness, selected support endpoints, or ambient quotient equality of Real names.

Lean4 (definition) `BEDC.Derived.CompactlySupportedRealFunctionUp`

Theorem 6597.2 (Compactly supported real function NameCert obligations)**.** *For every accepted* $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$ *carrier $F = (G, R, S, K, H, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactlySupportedRealFunc}}$ *surface is exhausted by carrier admission, graph-window coherence, Real readback, support-bound stability, compact and closed-bounded support handoff, componentwise transport, continuity replay, provenance, and local naming. Every consumer must read the finite route*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont

through the displayed rows $G, R, S, K, H, T, C, P, N$.

Proof. Project Definition 6597.1. The graph-window row, Real readback row, support-bound row, compact support handoff, transport, replay, continuity, provenance, and local name are the only rows in the carrier.

Graph-window coherence is read through G. The row exposes input and output observations as finite stream requests, so a consumer cannot replace the graph by host function equality or by an untracked total map.

The Real readback and support obligations are downstream from the same finite observations. The rows R, S, K seal regular rational and dyadic data as Real outputs, record the closed bounded support interval, and hand that interval to compact-support consumers without importing open-cover compactness.

Finally H, T, C, P, N transport, replay, source, and name this same route. Since no excluded equality, endpoint-selection, completeness, or open-cover row occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 6597.3 (Compactly supported real function support bound)**.** *Let $F = (G, R, S, K, H, T, C, P, N)$ be an accepted* $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$ *packet. Any consumer that cites compact support for F must read the support-bound certificate S and the closed-bounded handoff K before it reads the* $\mathsf{ContinuousMap}^{\uparrow}$ *compatibility row C or the* $\mathsf{Real}^{\uparrow}$ *output seal R as compact-support data. Thus support is witnessed by the displayed closed bounded interval route, not by a selected endpoint pair outside the packet or by a host-level compactness theorem.*

Proof. Start with the carrier projection. The only support data are the certificate row S and the compact closed-bounded handoff row K.

Read S through the graph-window row G. The certificate states that accepted observations outside the displayed interval are zero-output observations, so support is tied to the same stream, regular rational, dyadic, and Real readback route.

Now use K and C. The handoff exposes support through $\mathsf{ClosedBoundedInterval}^{\uparrow}$, while the continuity row allows continuous-map consumers to replay the same graph-window and Real-readback data. No row supplies open-cover compactness, selected endpoints, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$ read from an accepted packet factors through graph windows, Real readback, a displayed support-bound certificate, closed-bounded support handoff, componentwise transport, Cont replay, continuous-map compatibility, provenance, and local NameCert before compact-support consumers inspect the route.

Independence witness: real, streamname, regseqrat, dyadicratcore, continuousmap, closed-boundedinterval: $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries graph-window data, Real output readback, support-bound evidence, closed-bounded compact-support handoff, continuity replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactlySupportedRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactlySupportedRealFunction}^{\uparrow}$ packet starts from finite graph windows, reads Real outputs through StreamName, RegSeqRat, and DyadicRatCore rows, records a closed bounded support certificate, hands support to compact and continuous-map consumers, and preserves the route by hsame transport, Cont replay, Pkg provenance, and local NameCert.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactlySupportedRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6597.1, Definition 6597.2, and Definition 6597.3 seed the finite compact-support route over Real, stream, regular rational, dyadic, continuous-map, and closed bounded interval rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactlySupportedRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, function-space completeness, open-cover Heine–Borel compactness, selected support endpoints, quotient equality of Real names, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked graph-window coherence, Real readback, support-bound stability, closed-bounded support handoff, continuity replay, ledger nonescape, and a Lean carrier surface for BEDC.Derived.CompactlySupportedRealFun

Chapter 6598

A Concrete Naming Certificate for CompactMetricUniformContinuity$^{\uparrow}$

Following Bishop–Bridges and the Heine–Cantor finite-net proof, the CompactMetricUniformContinuity$^{\uparrow}$ packet records the uniform-continuity theorem for a continuous map on a compact metric source as finite NameCert data. It sits over the compact finite-net source of chapter 129, the pointwise map rows of chapter 130, the downstream modulus interface of chapter 369, the metric carrier of chapter 57, and the rational real seal of chapter 38. The packet names the finite route from local continuity radii and compact nets to a tolerance-only modulus; it does not import open-cover compactness, sequential compactness, host function equality, or a choice-selected center family.

Definition 6598.1 (Compact metric uniform continuity carrier)**.** *A* CompactMetricUniformContinuity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, M, Q, D, R, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, including the displayed center bundle, cover radii, and located membership witnesses for the source metric space.*
2. *F is the* ContinuousMap$^{\uparrow}$ *row, carrying the graph read and the pointwise continuity modulus rows attached to the centers named by K.*
3. *M is the* Met$^{\uparrow}$ *triangle-estimate ledger used to compare a point, a selected displayed center, and their images under the map row.*
4. *Q is the finite* Rat$^{\uparrow}$ *lower-bound fold over the centerwise source radii supplied by F.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for halving, positivity, lower-bound comparison, and final source-tolerance transport.*
6. *R is the* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *seal boundary used only to read the rational and real comparison rows already displayed by Q, D, M.*
7. *T is the resulting tolerance-only* UniformModulus$^{\uparrow}$ *handoff row.*
8. *H records componentwise* hsame *transport for the compact source, continuous-map rows, metric estimates, rational fold, dyadic ledger, real seal, handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a target tolerance through $K, F, M, Q, D, R, H, P, N$ to the handoff row T.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactMetricUniformContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricUniformContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, sequential compactness, choice-selected centers, quotient equality of functions, ambient Real completeness, or host equality of maps.

Theorem 6598.2 (Compact metric uniform continuity NameCert obligations)**.** *For every accepted* $\mathsf{CompactMetricUniformContinuity}^{\uparrow}$ *carrier*

$$U = (K, F, M, Q, D, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricUniformContinuity}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure, continuous-map pointwise-modulus exposure, metric triangle-estimate replay, rational lower-bound fold exactness, dyadic tolerance transport, RegSeqRat and Real seal discipline, tolerance-only UniformModulus handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every theorem-facing read must display K, F, M, Q, D, R before it may consume the handoff row T.*

Proof. Project the carrier of Definition 6598.1. Its visible rows are precisely the compact finite-net source, continuous-map pointwise rows, metric comparison ledger, rational fold, dyadic tolerance ledger, Real-facing seal, uniform-modulus handoff, and structural rows.

The finite-net and pointwise-modulus obligations are supplied by K and F. The row K displays the centers and coverage witnesses, while F attaches a pointwise continuity radius to each displayed center.

The theorem-facing estimate is routed through M, Q, D, R. The metric ledger M records the triangle comparisons, the rational fold Q selects the displayed lower bound, the dyadic ledger D transports positivity and halving data, and the seal row R permits only the rational and Real reads already displayed by those rows.

Finally T, H, C, P, N hand the result to the public certificate. Since no excluded compactness, choice, function-equality, or ambient-completeness coordinate occurs in the carrier, the stated rows exhaust the NameCert obligation surface. □

Theorem 6598.3 (Finite-net modulus route)**.** *Every accepted* $\mathsf{CompactMetricUniformContinuity}^{\uparrow}$ *packet sends a compact-metric continuous map to a tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *row through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Unifor}$$

Concretely, the consumer reads the compact finite-net row K, the pointwise modulus row F, the metric triangle-estimate ledger M, the rational lower-bound fold Q, the dyadic tolerance ledger D, and the RegSeqRat and Real seal row R before it may use the modulus handoff T. The route exports no open-cover compactness, sequential compactness, choice-selected centers, host function equality, or external equality principle for maps.

Proof. Start with an accepted packet $U = (K, F, M, Q, D, R, T, H, C, P, N)$. The compact source row K provides the displayed centers and coverage witnesses, while the continuous-map row F provides the pointwise radius rows over those same centers.

Use the metric triangle ledger M to compare an arbitrary inspected point with its displayed center and then compare the corresponding image values. The rational fold Q reads the finite family of source radii and records the lower bound used by the uniform tolerance.

The dyadic ledger D transports the requested target tolerance through halving, positivity, and lower-bound comparisons. The RegSeqRat and Real seal row R is then read only as a boundary for those already displayed numerical comparisons.

The handoff row T is therefore a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ output. The structural rows H, C, P, N preserve, replay, source, and name the same finite route, so no excluded compactness, choice, function-equality, or external map-equality object can enter the export. □

Theorem 6598.4 (Compact metric uniform-continuity explicit modulus scope)**.** *An accepted* $\mathsf{CompactMetricUniformContinuity}^{\uparrow}$ *packet may export a uniform-continuity claim only through the displayed compact witness K, metric distance witness M, finite-net radius budget Q, D, modulus handoff row T, componentwise* hsame *transport H, and displayed* Cont *replay C. Pointwise continuity data in F alone do not determine an exported* $\mathsf{UniformModulus}^{\uparrow}$ *row; the packet must also expose the compact finite-net coverage and the metric comparison route that make the modulus tolerance-only.*

Proof. Start with the carrier of Definition 6598.1. The compact row K displays the finite centers and coverage witnesses, while the continuous-map row F supplies only centerwise or pointwise continuity data. Without the finite-net row, those pointwise rows have no tolerance-only export surface.

Compose the compact, metric, and continuous-map rows. The point under inspection is first related to a displayed center by K and the metric comparison ledger M; the row F is then read at that displayed center. The rational lower-bound fold Q and dyadic ledger D turn the finite family of centerwise source radii into the radius budget used by the modulus handoff T.

The structural rows keep the export constructive. Componentwise hsame transport H preserves the compact witness, metric distance witness, radius budget, and handoff, while displayed Cont replay C records the same route for each requested tolerance. Hence the exported uniform-continuity claim is the finite-net route $K, F, M, Q, D, R, T, H, C, P, N$, not open-cover compactness, a choice-selected subcover, or pointwise continuity detached from a displayed modulus. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricUniformContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricUniformContinuity}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMapUp pointwise-modulus rows, MetricUp triangle ledgers, RatUp lower-bound folds, DyadicRatCoreUp tolerance transport, RegSeqRatUp and RealUp seal boundaries, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactMetricUniformContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 6598.1, Definition 6598.2, and Definition 6598.3 seed the finite-net Heine–Cantor certificate over compact metric sources and pointwise continuous maps.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactMetricUniformContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, choice-selected centers, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, pointwise-modulus coverage, metric triangle-estimate replay, lower-bound fold exactness, dyadic tolerance transport, Real seal discipline, UniformModulus handoff, and non-escape for BEDC.Derived.CompactMetricUniformContinuityUp.

Chapter 6599

A Concrete Naming Certificate for $\mathsf{BishopContractionIteration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopContractionIteration}^{\uparrow}$ packet records the finite iteration-budget surface for a displayed contraction before any fixed-point endpoint or host limit is claimed. It sits after the complete-metric carrier of chapter 128, the one-step Picard contraction surface of chapter 395, the Cauchy-modulus schedule of chapter 292, the finite stream route of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the visible budget by which contraction iterates become completion-facing data; it does not select a fixed point, quotient an orbit, or import countable choice.

Definition 6599.1 (BishopContractionIteration carrier)**.** *A* $\mathsf{BishopContractionIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, T, x_0, W, I, A, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row carrying only the displayed distance reads needed by the contraction route.*
2. *T is the* $\mathsf{PicardContraction}^{\uparrow}$ *one-step row, exposing the accepted contraction graph and its adjacent-iterate comparison surface.*
3. *x_0 is the initial-point row from which the displayed orbit windows are generated.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule requested by a downstream completion or fixed-point consumer.*
5. *I is the finite iterate ledger, listing the visible orbit rows $x_0, x_1, \ldots, x_n$ and the replayed applications of T used to obtain them.*
6. *A is the adjacent-distance ledger, storing the contraction-bound comparisons and tail estimates read from I inside the metric row X.*
7. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *budget row extracted from A for the requested windows W.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite orbit budget and its regular rational tail presentation.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, I, A, M, R have been displayed.*

10. *H records componentwise* hsame *transport for the metric source, contraction row, initial point, window schedule, iterate ledger, adjacent-distance ledger, modulus row, readback row, seal row, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from complete-metric and Picard-contraction admission through finite iteration, tail-budget extraction, regular readback, and real-seal separation.*
12. *P is the* Pkg *provenance over* $\mathsf{BishopContractionIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopContractionIteration}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, T, x_0, W, I, A, M, R, E, H, C, P, N$. It has no coordinate for a selected limit, a Banach fixed point, host function extensionality, quotient orbit equality, countable choice, or ambient completeness beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Theorem 6599.2 (BishopContractionIteration NameCert obligations)**.** *For every accepted* BishopContractionIterat *carrier $B = (X, T, x_0, W, I, A, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopContractionIteration}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed complete-metric, Picard-contraction, initial-point, stream-window, iterate-ledger, adjacent-distance, Cauchy-modulus, regular-readback, Real-seal, transport, replay, provenance, and local naming rows; iteration visibility through I; tail-budget visibility through A and M; readback visibility through R; Real seal non-escape through E; and componentwise preservation through H, C, P, N.*

Proof. Project the carrier of Definition 6599.1. Carrier admission is exactly the displayed finite inventory $X, T, x_0, W, I, A, M, R, E, H, C, P, N$.

The iterate ledger I is generated from the initial row x_0 by replaying the Picard-contraction row T inside the complete-metric source X. The adjacent-distance ledger A reads only those displayed orbit rows and the contraction comparisons attached to them.

The modulus row M is therefore a budget extracted from visible finite comparisons for the requested windows W. The regular readback R and terminal seal E are downstream rows, so they cannot add a selected endpoint or a fixed-point theorem.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since every readable coordinate has been listed, the certificate exports the contraction-iteration budget and not host function equality, quotient orbit equality, countable choice, or ambient completion. □

Theorem 6599.3 (BishopContractionIteration budget extraction)**.** *For every accepted* $\mathsf{BishopContractionIteration}^{\uparrow}$ *carrier $B = (X, T, x_0, W, I, A, M, R, E, H, C, P, N)$, every requested finite window in W receives its contraction budget only through the route*

$$\mathsf{PicardContraction}^{\uparrow} \longmapsto I \longmapsto A \longmapsto \mathsf{CauchyModulus}^{\uparrow}.$$

The row M is read after the finite orbit ledger and adjacent-distance comparisons have been displayed, and no budget is read from a selected fixed point, completed orbit endpoint, or host metric completion.

Proof. Start with the rows T, x_0, W, I. The only source of successive iterates is the displayed Picard-contraction row T replayed from the initial point x_0 for the finite windows requested in W.

The row A then records the adjacent-distance comparisons over those visible iterates. Because A is tied to I and the metric row X, it cannot read a completed endpoint or an external fixed-point equation.

The Cauchy-modulus row M is extracted from A for the same finite windows. Thus the budget route passes through $\mathsf{PicardContraction}^{\uparrow}$, the finite iterate ledger, the adjacent-distance ledger, and $\mathsf{CauchyModulus}^{\uparrow}$ before any regular readback or Real seal is available. □

Theorem 6599.4 (BishopContractionIteration completion boundary)**.** *Every completion-facing or fixed-point-facing consumer of an accepted* $\mathsf{BishopContractionIteration}^{\uparrow}$ *carrier must first read the displayed rows*

$$X,\ T,\ x_0,\ W,\ I,\ A,\ M,\ R,\ E,\ H,\ C,\ P,\ N.$$

The consumer receives a finite $\mathsf{StreamName}^{\uparrow}$ *window schedule, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a* $\mathsf{Real}^{\uparrow}$ *seal boundary for the contraction orbit, but it receives no Banach fixed point, selected limit, quotient orbit equality, host function extensionality, countable choice, or ambient completeness beyond the named complete-metric row.*

Proof. Read the carrier inventory from Definition 6599.1. The complete-metric row X, Picard-contraction row T, initial row x_0, requested windows W, iterate ledger I, and adjacent-distance row A are all visible before the Cauchy-modulus row can be consumed.

Apply Definition 6599.3. It places the modulus row M downstream of the finite orbit and contraction comparisons rather than downstream of a selected endpoint.

The readback and seal rows R, E are therefore boundary rows for the displayed budget. Transport, replay, provenance, and local naming preserve that same route through H, C, P, N, so a downstream endpoint theorem must supply its own fixed-point and uniqueness obligations. □

Theorem 6599.5 (BishopContractionIteration Real seal nonescape)**.** *For every accepted* $\mathsf{BishopContractionIteration}^{\uparrow}$ *carrier* $B = (X, T, x_0, W, I, A, M, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *seals only the finite contraction-iteration readback built from* W, I, A, M, R*. It excludes Banach fixed points, selected limits, host function equality, quotient orbit equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The seal row E is downstream of the stream windows, finite iterate ledger, adjacent-distance comparisons, modulus row, and regular readback.

Use Definition 6599.4. Any completion-facing consumer must read the same finite rows before it can inspect E.

Since the classifier has no coordinate for fixed points, selected limits, quotient orbit equality, host function equality, countable choice, or ambient real completeness, none of those objects can escape through the terminal seal. □

Falsifiable prediction: Every accepted $\mathsf{BishopContractionIteration}^{\uparrow}$ read factors through one CompleteMetric row, one PicardContraction row, one initial-point row, one StreamName window schedule, one finite iterate ledger, one adjacent-distance ledger, one CauchyModulus budget, one RegSeqRat readback, one RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming before any endpoint theorem consumes it.

Independence witness: completemetric, picardcontraction, cauchymodulus, streamname, regseqrat, real: $\mathsf{BishopContractionIteration}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the initial point, finite orbit windows, contraction-distance budget, modulus extraction, regular readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopContractionIteration}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopContractionIteration}^{\uparrow}$ packet is generated from a CompleteMetric source row, a PicardContraction row, an initial point, StreamName windows, a finite iterate ledger, adjacent-distance comparisons, a

CauchyModulus budget, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopContractionIteration}\uparrow}$, seed)

Scope closed: Definition 6599.1, Definition 6599.2, Definition 6599.3, and Definition 6599.4 seed the finite contraction-iteration budget consumed by later completion and fixed-point routes.

Formal verification: FormalStatus(NameCert$_{\text{BishopContractionIteration}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Banach fixed points, selected limits, quotient orbit equality, host function extensionality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iteration visibility, tail-budget visibility, Cauchy-modulus extraction, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopContractionIterationUp.

Chapter 6600

A Concrete Naming Certificate for $\mathsf{RealOscillationMajorant}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealOscillationMajorant}^{\uparrow}$ packet records a finite oscillation-bound ledger for Real-facing estimates without appealing to host suprema or ambient completeness. It sits over the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance arithmetic of chapter 291, and the convergence-rate packet of chapter 2074. The carrier is meant for later Cauchy, continuity, and compact-interval estimates that need a displayed oscillation majorant before they read a terminal $\mathsf{Real}^{\uparrow}$ seal.

Definition 6600.1 (Real oscillation majorant carrier)**.** *A* $\mathsf{RealOscillationMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (E, Q, S, D, M, A, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row whose accepted real-name boundary is being estimated.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplying regular rational approximants to the sealed real name.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the stream observations used by Q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii, comparisons, and halving slack.*
5. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *schedule row that bounds which windows must be inspected at the displayed tolerance.*
6. *A is the finite oscillation-majorant ledger, recording a dyadic bound above all displayed pairwise oscillations seen through Q, S, D, M.*
7. *W is the finite witness-window family naming the inspected indices and comparisons that justify the ledger row A.*
8. *H records componentwise* hsame *transport for E, Q, S, D, M, A, W.*
9. *C records displayed* Cont *replay from a tolerance request through the modulus row and finite witness windows to the majorant ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{RealOscillationMajorant}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{RealOscillationMajorant}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host supremum, a completed metric-space bound, a selected limit, quotient equality of streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6600.2 (Real oscillation majorant NameCert obligations)**.** *For every accepted* $\mathsf{RealOscillationMajorant}^{\uparrow}$ *carrier*

$$O = (E, Q, S, D, M, A, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealOscillationMajorant}^{\uparrow}}$ *surface consists exactly of Real seal admission, regular rational readback, finite stream-window exposure, dyadic tolerance arithmetic, convergence-modulus scheduling, oscillation-majorant ledger exactness, witness-window coverage, componentwise transport, displayed replay, package provenance, local naming, and non-escape for host suprema, selected limits, quotient stream equality, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 6600.1. Its visible rows are precisely the Real seal, RegSeqRat readback, StreamName window, DyadicRatCore tolerance, RealModulusOfConvergence schedule, majorant ledger, witness windows, and structural rows.

The analytic obligations are finite. Row D fixes the dyadic tolerance arithmetic, row M fixes the inspected tail window, rows Q, S expose the regular rational observations, and row W lists the finite comparisons whose dyadic upper bound is recorded by A.

The structural obligations then bind the read. Componentwise transport through H, replay through C, provenance through P, and local naming through N keep every oscillation-majorant consumer on the same displayed packet. Since the excluded supremum, selected-limit, quotient, choice, and completeness objects have no coordinate, the listed rows exhaust the certificate surface. □

Theorem 6600.3 (Real oscillation majorant L10 readback)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{RealOscillationMajorant}^{\uparrow}$ *packet reads an oscillation bound only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads the dyadic tolerance row D, the finite stream-window row S, the regular rational readback row Q, the convergence-modulus row M, and the terminal Real seal row E before it may use the majorant ledger A and witness family W. The route exports no host supremum, selected Cauchy limit, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Begin with an accepted carrier $O = (E, Q, S, D, M, A, W, H, C, P, N)$. The only tolerance source is D, and the only finite observation source is S.

Read the Real-facing schedule. The modulus row M bounds which stream windows are inspected, the regular rational row Q supplies the displayed approximants on those windows, and the Real seal row E is reached only after that finite route has been exposed.

The majorant ledger is then read from A together with the witness family W. Rows H, C, P, N transport, replay, source, and name the same packet, so the L10 consumer cannot replace the displayed route by a host supremum, selected limit, quotient equality, choice schedule, or completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOscillationMajorant}^{\uparrow}}$.

Constructive story: A $\mathsf{RealOscillationMajorant}^{\uparrow}$ packet is generated from a RealUp seal row, RegSeqRat readback, StreamName finite windows, DyadicRatCore tolerance arithmetic, a RealModulusOfConvergence schedule, a finite

oscillation-majorant ledger, witness windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealOscillationMajorant}^{\uparrow}}$, seed)

Scope closed: Definition 6600.1, Definition 6600.2, and Definition 6600.3 seed the finite oscillation-majorant route for Real, RegSeqRat, StreamName, DyadicRatCore, and convergence-modulus consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealOscillationMajorant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host suprema, selected Cauchy limits, quotient stream equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance arithmetic, finite-window readback, convergence-modulus scheduling, majorant-ledger exactness, witness-window coverage, structural replay, and non-escape for BEDC.Derived.RealOscillationMajorantUp.

Chapter 6601

A Concrete Naming Certificate for $\mathsf{MarkovInequality}^{\uparrow}$

Following Feller, Markov's inequality is read as a finite first-moment tail-bound route. It sits beside the second-moment packet of chapter 6592 and consumes the probability source of chapter 184, the random-variable surface of chapter 185, the distribution readback of chapter 186, and the terminal real-name seal of chapter 38. The packet names a nonnegative tail-bound certificate; it is not a weak-law theorem, a variance estimate, a selected exceptional event, a host measure-algebra quotient, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6601.1 (Markov inequality carrier)**.** *A* $\mathsf{MarkovInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, X, a, E, T, B, S, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the normalized event classifier.*
2. *X is the nonnegative* $\mathsf{RandomVar}^{\uparrow}$ *row whose finite observation windows are inspected over P.*
3. *a is the positive threshold row, displayed as a* $\mathsf{Real}^{\uparrow}$ *comparison seal before any tail event is read.*
4. *E is the expectation/readback row for X, with any rational approximation window displayed through the same probability source.*
5. *T is the tail-event row naming the event where the observation of X lies above the threshold a.*
6. *B is the probability-bound comparison row, displaying the Markov estimate between the probability of T and the first-moment ratio read from E and a.*
7. *S stores the paired* $\mathsf{Real}^{\uparrow}$ *and rational seals used to compare the threshold, expectation, tail probability, and ratio rows.*
8. *H stores componentwise* hsame *transport for the probability source, random variable, threshold, expectation/readback, tail event, probability comparison, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the finite first-moment tail-bound route.*
10. *Q is the* Pkg *provenance over* $\mathsf{MarkovInequality}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovInequality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $P, X, a, E, T, B, S, H, C, Q, N$. It has no coordinate for a variance budget, almost-sure convergence, selected subevent, host integration theorem, quotient real equality, countable choice, or ambient real completeness.

Theorem 6601.2 (Markov inequality NameCert obligations)**.** *For every accepted* $\mathsf{MarkovInequality}^{\uparrow}$ *carrier $M = (P, X, a, E, T, B, S, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MarkovInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the probability source, nonnegative random-variable row, positive threshold, expectation/readback row, tail event, probability-bound comparison, Real/Rat seals, transport, replay, provenance, and local naming;*
2. *source stability, requiring X, T, E to remain attached to the event classifier displayed by P;*
3. *nonnegativity stability, requiring the random-variable read in X to be exposed as nonnegative before the threshold comparison is consumed;*
4. *threshold stability, requiring a to be displayed as a positive* $\mathsf{Real}^{\uparrow}$ *row before T and B are read;*
5. *first-moment stability, requiring E to be displayed before the probability-bound comparison consumes the first-moment ratio;*
6. *seal nonescape, requiring S to bind the Real and rational comparison rows without exporting quotient real equality, a host integral theorem, or ambient completeness;*
7. *componentwise transport, displayed replay, package provenance, and local naming through H, C, Q, N.*

These obligations certify only the finite first-moment tail-bound route. They do not close Chebyshev's inequality, a weak law, almost-sure convergence, host probability limits, selected exceptional events, Lean verification, or bridge checking.

Proof. Project the carrier of Definition 6601.1. The displayed rows are exactly the probability source, nonnegative random-variable row, threshold, expectation/readback, tail event, probability-bound comparison, seals, and structural rows.

Source, nonnegativity, threshold, and first-moment stability are read from P, X, a, E, T, B. The probability-bound comparison cannot be consumed until the event classifier, nonnegative observation row, positive threshold, and expectation/readback row have all been displayed.

The seal row S records only the Real and rational endpoints used by that comparison. Since the carrier has no coordinate for quotient real equality, a host integral theorem, selected exceptional event, countable choice, or ambient completeness, the seal cannot export those objects.

The rows H, C, Q, N transport, replay, source, and name the same finite route. Hence the listed obligations exhaust the local NameCert surface. □

Theorem 6601.3 (Markov–Chebyshev handoff)**.** *Every* $\mathsf{ChebyshevInequality}^{\uparrow}$ *consumer that requests a first-moment tail-bound precursor can read a* $\mathsf{MarkovInequality}^{\uparrow}$ *packet only through*

$$P \longmapsto X \longmapsto a \longmapsto E \longmapsto T \longmapsto B \longmapsto S \longmapsto H, C, Q, N.$$

The handoff supplies a probability source, nonnegative random-variable row, positive threshold, expectation/readback row, tail event, probability-bound comparison, Real/Rat seals, transport, replay, provenance, and local naming. It does not supply a variance budget, centered second-moment estimate, weak-law limit, almost-sure convergence claim, selected subsequence, or ambient real-completeness principle.

Proof. Begin with a Chebyshev-facing request for a first-moment precursor. Projection of Definition 6601.1 displays the probability source P and the nonnegative random-variable row X before any tail estimate can be consumed.

The request next reads the positive threshold a, the expectation/readback row E, and the tail-event row T. Only then can the probability-bound comparison B and seal row S be handed to the consumer.

The structural rows H, C, Q, N preserve the same route. Because the Markov carrier has no variance-budget or centered second-moment coordinate, the consumer cannot treat this handoff as a Chebyshev proof, weak-law limit, selected subsequence, or ambient completeness claim. □

Theorem 6601.4 (Markov Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or rational seal read from an accepted* $\mathsf{MarkovInequality}^{\uparrow}$ *carrier is downstream from the displayed probability source, nonnegative random-variable row, threshold, expectation/readback row, tail event, and probability-bound comparison. The seal exposes only the finite first-moment comparison route and cannot escape to quotient real equality, host expectation theory, selected events, countable choice, distributional convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row S through the carrier coordinates P, X, a, E, T, B. Each endpoint used by the comparison is attached to either the threshold, expectation/readback, tail probability, or first-moment ratio row.

The rational windows in S are displayed only as comparison seals, and the $\mathsf{Real}^{\uparrow}$ endpoints remain downstream from the same probability source and random-variable surface. No carrier coordinate stores a quotient real equality, host expectation theorem, selected event, countable-choice selector, or convergence principle.

Therefore every seal read remains inside the finite first-moment tail-bound route named by $\mathsf{MarkovInequality}^{\uparrow}$. □

Falsifiable prediction: Every $\mathsf{MarkovInequality}^{\uparrow}$ tail-bound read factors through a ProbSpace source row, a nonnegative RandomVar row, a positive threshold, an expectation/readback row, a tail-event row, a probability-bound comparison, Real/Rat seals, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: probspace, randomvar, distribution, real, chebyshev-inequality: $\mathsf{MarkovInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a probability source, nonnegative random variable, threshold, expectation/readback, tail event, probability-bound comparison, Real/Rat seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovInequality}^{\uparrow}$ packet is generated from a probability source row, a nonnegative random-variable row, a positive threshold row, an expectation/readback row, a tail-event row, a probability-bound comparison row, Real and rational seals, componentwise hsame transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6601.1, Definition 6601.2, Definition 6601.3, and Definition 6601.4 seed the finite first-moment tail-bound route through ProbSpace, RandomVar, Distribution, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Chebyshev's inequality, weak laws, almost-sure convergence, host probability limits, selected exceptional events, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source stability, nonnegativity stability, threshold stability, first-moment stability, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MarkovInequalityUp.

Chapter 6602

A Concrete Naming Certificate for $\mathsf{PolarDecomposition}^{\uparrow}$

Following Conway and Reed–Simon, the polar-decomposition interface records the finite operator-factorization surface by which a bounded operator is read as a positive part followed by a partial-isometry or unitary handoff. The $\mathsf{PolarDecomposition}^{\uparrow}$ packet consumes the bounded-operator route of chapter 1644, the Hilbert and inner-product rows of chapter 91 and chapter 86, the matrix and singular-value ledgers of chapter 49 and chapter 6037, the unitary-group boundary of chapter 219, and the spectral-radius estimates of chapter 2352. It names only a displayed finite factorization packet; it does not assert a global spectral theorem, selected bases, quotient operator equality, or ambient Hilbert completeness.

Definition 6602.1 (Polar decomposition carrier)**.** *A* $\mathsf{PolarDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, A, U, S, R, H, C, G, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row whose source and target Hilbert endpoints are the operator being factored.*
2. *A is the positive-part row. It records the displayed nonnegative self-adjoint endpoint read as the finite positive component of the factorization, not as an unbounded spectral-calculus object.*
3. *U is the partial-isometry or unitary handoff row. It is read through* $\mathsf{UnitaryGroup}^{\uparrow}$ *when the inspected finite window supports a unitary restriction and otherwise records the partial-isometry boundary over the same Hilbert endpoint rows.*
4. *S is the* $\mathsf{SingularValueDecomposition}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *ledger. It records finite matrix windows, singular values, and frame data used by finite-dimensional consumers of the same factorization route.*
5. *R is the* $\mathsf{SpectralRadius}^{\uparrow}$ *estimate row. It supplies only the norm-growth and radius bounds displayed by the sibling packet.*
6. *H stores componentwise* hsame *transport for the operator row, positive part, handoff row, SVD and matrix ledger, spectral estimate, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from a factorization request through T, A, U, S, R, H.*

8. *G is the* Pkg *provenance over* $\mathsf{PolarDecomposition}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{SingularValueDecomposition}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{PolarDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host spectral theorem, a selected basis outside the packet, quotient equality of operators, or a completeness theorem for the ambient Hilbert carrier.

Theorem 6602.2 (Polar decomposition NameCert obligations)**.** *For every accepted* $\mathsf{PolarDecomposition}^{\uparrow}$ *carrier*

$$P = (T, A, U, S, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolarDecomposition}^{\uparrow}}$ *surface consists exactly of bounded-operator admission, positive-part exposure, partial-isometry or unitary handoff, SVD and matrix ledger routing, spectral-radius estimate routing, componentwise transport, displayed replay, provenance, local naming, and nonescape for global spectral theorems, selected bases, quotient operator equality, and ambient Hilbert completeness.*

Proof. Project Definition 6602.1. The only analytic rows are the bounded operator T, the positive part A, the handoff row U, the finite SVD and matrix ledger S, and the spectral estimate R.

The factorization obligation is therefore finite. A consumer first reads T through $\mathsf{BoundedLinearOperator}^{\uparrow}$, then reads A as the positive component attached to the same Hilbert and inner-product rows, and then reads U as the displayed partial-isometry or unitary handoff.

The matrix and spectral consumers are downstream from that same route. Row S exposes only finite matrix and singular-value windows, while R exposes only the sibling spectral-radius estimate. Rows H, C, G, N transport, replay, source, and name the packet, so no excluded global theorem or quotient object can be read from the carrier. □

Theorem 6602.3 (Polar decomposition partial-isometry handoff)**.** *Every accepted* $\mathsf{PolarDecomposition}^{\uparrow}$ *packet reads a factorization handoff only through the route*

$$\mathsf{BoundedLinearOperator}^{\uparrow} \longrightarrow \mathsf{Hilbert}^{\uparrow} \longrightarrow \mathsf{InnerProduct}^{\uparrow} \longrightarrow \mathsf{UnitaryGroup}^{\uparrow} \longrightarrow \mathsf{SingularValueDecomposition}^{\uparrow} \longrightarrow \mathsf{S}$$

The row U is admissible only as the displayed partial-isometry or unitary boundary attached to the same bounded operator row T and positive part A. It exports no selected polar factor outside the packet, no host quotient equality of operators, and no ambient Hilbert completeness claim.

Proof. Begin with an accepted carrier $P = (T, A, U, S, R, H, C, G, N)$. The operator endpoint is T, so every factorization read starts from the displayed $\mathsf{BoundedLinearOperator}^{\uparrow}$ row over Hilbert and inner-product data.

Read the positive part A and the handoff row U together. The handoff is accepted only when it is tied to the same source and target rows as T and to the same positive component A; the unitary case is read through the sibling $\mathsf{UnitaryGroup}^{\uparrow}$ route.

Finally read S and R. The SVD and matrix ledger supplies finite inspected windows, and the spectral-radius row supplies only displayed estimate data. Since the carrier contains no selected external polar factor, quotient operator equality, or Hilbert-completeness row, the handoff remains the finite packet described in the statement. □

Falsifiable prediction: Every $\mathsf{PolarDecomposition}^{\uparrow}$ consumer that reaches a factorization handoff must first read the bounded-operator row, positive-part row, partial-isometry or unitary row, SVD

and matrix ledger, spectral-radius estimate, transport, replay, provenance, and local NameCert rows.

Independence witness: boundedlinearoperator, hilbert, innerproduct, matrix, singularvaluedecomposition, unitarygroup, spectralradius: $\mathsf{PolarDecomposition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded operator, positive part, partial-isometry or unitary handoff, finite SVD and matrix ledger, spectral-radius estimate, structural replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PolarDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{PolarDecomposition}^{\uparrow}$ packet is generated from a BoundedLinearOperatorUp row over Hilbert and inner-product data, a positive-part row, a partial-isometry or unitary handoff, finite SVD and matrix ledgers, a SpectralRadiusUp estimate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolarDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6602.1, Definition 6602.2, and Definition 6602.3 seed the finite polar-factorization packet over bounded operators, Hilbert rows, positive parts, unitary handoffs, SVD ledgers, and spectral-radius estimates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolarDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global spectral theorem, selected bases, quotient operator equality, ambient Hilbert completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-part exactness, partial-isometry handoff, SVD and spectral ledger routing, structural replay, provenance, and nonescape for BEDC.Derived.PolarDecompositionUp.

Chapter 6603

A Concrete Naming Certificate for $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges on compact intervals and located reals, the $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$ packet records the finite subsequence-extraction surface available for bounded located $\mathsf{Real}^{\uparrow}$ sequences. It sits over located membership data from chapter 1541, the sequential completeness route of chapter 1515, the regular rational readback of chapter 301, the finite window schedule of chapter 126, and the terminal Real seal of chapter 38. The packet names located finite windows and extracted regular subsequence reads; it does not claim open-cover compactness, choice-selected infinite subsequences, quotient stream equality, excluded-middle trichotomy, or ambient classical compactness.

Definition 6603.1 (Located Bolzano-Weierstrass carrier)**.** *A* $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, L, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded source sequence row, presenting the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of a located* $\mathsf{Real}^{\uparrow}$ *sequence together with its displayed lower and upper bounds.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row, recording the located interval or located subset surface in which the sequence terms are read.*
3. *M is the* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *row, exposing only the sequential metric-completeness interface needed by the extracted regular reads.*
4. *W is the finite window row of candidate indices and rational brackets inspected before the subsequence read is named.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the accepted windows and bounds.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the extracted finite-window read.*
7. *H records componentwise* hsame *transport across source sequence, located-set, completeness, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the extraction route*

$$S, L, M, W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, M, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected infinite subsequences, quotient equality of streams, countable choice, host real equality, or ambient classical compactness.

Theorem 6603.2 (Located Bolzano-Weierstrass carrier obligations)**.** *For every accepted* LocatedBolzanoWeierstrass *carrier*

$$B = (S, L, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded source sequence,* $\mathsf{LocatedSet}^{\uparrow}$ *row,* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *row, finite candidate windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every bounded located sequence consumer must pass through these rows before using the packet as a finite subsequence-extraction certificate.*

Proof. Project the carrier of Definition 6603.1. The accepted packet exposes exactly the bounded sequence, located-set, sequential-completeness, finite-window, readback, seal, transport, replay, provenance, and local naming rows.

The located-set row supplies the finite located membership and distance surface. The sequential-completeness row supplies only the metric route needed for regular readbacks, while W records the inspected candidate windows.

The readback row R and Real seal E are downstream of those finite observations. Transport through H and replay through C preserve the same displayed route, and P, N source and name it.

Since the carrier has no coordinate for the excluded compactness, selection, quotient, choice, or host equality data, the listed rows exhaust the local NameCert obligations. □

Theorem 6603.3 (Located Bolzano-Weierstrass finite-window subsequence)**.** *Let $B = (S, L, M, W, R, E, H, C, P, N$ be an accepted* $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$ *carrier. Every finite-window subsequence read exported by B factors through*

$$S, L, M, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently downstream compactness, completion, or limit consumers receive a bounded located source row, a $\mathsf{LocatedSet}^{\uparrow}$ *route, a* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *route, finite candidate windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data, but no open-cover compactness proof, choice-selected infinite subsequence, quotient stream equality, or ambient classical compactness principle.*

Proof. Begin with an exported finite-window subsequence read. Projection exposes the ordered carrier rows $S, L, M, W, R, E, H, C, P, N$.

The read starts from the bounded source sequence row S and located-set row L. The sequential-completeness row M is read only as a metric route, not as a global compactness assertion.

The finite candidate window W determines the regular rational readback R, and the terminal Real seal E is visible only after that readback. The structural rows H, C, P, N then transport, replay, source, and name the same route.

Thus every downstream consumer receives the displayed finite subsequence-extraction surface and none of the excluded selection, quotient, choice, compactness, or host-equality data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}$ packet is generated from bounded located Real sequence windows, LocatedSet membership data, SequentiallyCompleteMetric route data, finite candidate windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6603.1, Definition 6603.2, and Definition 6603.3 seed the finite located subsequence-extraction surface for bounded located Real sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected infinite subsequences, quotient stream equality, countable choice, host real equality, ambient classical compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window stability, sequential-completeness route stability, regular readback exactness, Real seal handoff, and downstream nonescape for BEDC.Derived.LocatedBolzanoWeierstrassUp.

Chapter 6604

A Concrete Naming Certificate for $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real construction, the $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$ packet records the diagonal-limit handoff anticipated by section 9238.2 and Definition 9238.5. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the finite rows by which a regular Cauchy diagonal read is lifted to a Real-facing limit seal without selected representatives, quotient stream equality, countable choice, or ambient Real-completeness as a hidden coordinate.

Definition 6604.1 (Regular Cauchy lifted-limit carrier)**.** *A* $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, W, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diagonal tolerance ledger, carrying the finite precision budgets and comparison cells used by the lift.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite windows inspected at each displayed diagonal budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row produced from the scheduled windows and dyadic budgets.*
4. *W is the lifted-limit witness row, recording that the same finite diagonal observations determine the displayed limit read.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available after D, S, R, W have been exposed.*
6. *H records componentwise* hsame *transport across dyadic budget, stream schedule, regular readback, lifted witness, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the diagonal handoff route*

$$D, S \longmapsto R \longmapsto W \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLiftedLimit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, W, E, H, C, P, N$. It has no coordinate for quotient equality of regular streams, selected representatives, countable choice, host real equality, a detached completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6604.2 (Regular Cauchy lifted-limit NameCert obligations)**.** *For every accepted* RegularCauchyLiftedLim *carrier*

$$L = (D, S, R, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLiftedLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diagonal tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, lifted-limit witness, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the Real-completeness limit theorem's diagonal handoff must pass through these rows and cannot replace them by quotient stream equality, selected representatives, countable choice, host real equality, a detached completion operator, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLiftedLimitUp.TasteGate.RegularCauchyLiftedLimitNamecertObligations`

Proof. Project the carrier of Definition 6604.1. The accepted packet exposes exactly the dyadic budget, finite stream schedule, regular rational readback, lifted-limit witness, Real seal, transport, replay, provenance, and local naming rows.

The diagonal handoff begins at the $\mathsf{DyadicRatCore}^{\uparrow}$ row D and the $\mathsf{StreamName}^{\uparrow}$ row S. These rows supply the finite precision and window data cited by the originating limit theorem.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of those observations, and the witness row W binds the same observations to the terminal $\mathsf{Real}^{\uparrow}$ seal E. Transport through H and replay through C preserve the route.

Finally P and N source and name the same finite packet. Since no excluded quotient, selector, choice, host equality, detached completion, or ambient-completeness row is present, the displayed rows exhaust the NameCert obligations. □

Theorem 6604.3 (Regular Cauchy lifted-limit diagonal handoff)**.** *Let $L = (D, S, R, W, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$ *carrier. Every diagonal-limit read exported toward a* $\mathsf{Real}^{\uparrow}$ *consumer factors through*

$$D, S \longmapsto R \longmapsto W \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, lifted-limit witness, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data, and receives no quotient stream equality, selected representative, countable choice schedule, detached completion operator, or ambient Real-completeness credential.*

Proof. Start with a diagonal-limit read from the accepted packet. Projection exposes the ordered rows $D, S, R, W, E, H, C, P, N$.

The dyadic budget ledger D and finite schedule S are the only data available before the regular rational readback. Thus R is determined by displayed finite observations rather than by a quotient stream.

The witness row W records the lift from that regular readback to the terminal Real seal E. The rows H, C, P, N then transport, replay, source, and name the same finite handoff.

Therefore every Real-facing consumer receives exactly the displayed diagonal route and none of the excluded quotient, selection, choice, completion, or ambient-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLiftedLimit}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyLiftedLimit}^{\uparrow}$ packet is generated from a DyadicRatCore diagonal tolerance ledger, StreamName finite-window schedule, RegSeqRat readback, lifted-limit witness, terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLiftedLimit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6604.1, Definition 6604.2, and Definition 6604.3 seed the finite diagonal-limit handoff surface for Real-completeness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLiftedLimit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of regular streams, selected representatives, countable choice, host real equality, detached completion operators, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-budget stability, finite-window readback exactness, lifted-limit witness stability, Real seal handoff, and downstream nonescape for BEDC.Derived.RegularCauchyLiftedLimitUp.

Chapter 6605

A Concrete Naming Certificate for $\mathsf{LocatedSetDistance}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric spaces, the $\mathsf{LocatedSetDistance}^{\uparrow}$ packet records the finite distance-to-a-located-set surface as displayed metric and real-valued ledger data. It sits over the located-subset carrier of chapter 1541, the distance-to-set surface of chapter 4364, the proximinal boundary of chapter 3498, the metric source of chapter 57, and the terminal real-name seal of chapter 38. The packet names the finite route from an admitted ambient point and located subset to a $\mathsf{Real}^{\uparrow}$ distance read; it does not select a nearest point, quotient subset presentations, assert ambient compactness, or import full real completeness.

Definition 6605.1 (Located set distance carrier)**.** *A* $\mathsf{LocatedSetDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row, including the admitted point classifier and distance observations.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row for the displayed subset, including its admission and locatedness comparisons.*
3. *D is the* $\mathsf{DistanceToSet}^{\uparrow}$ *ledger row assigning each admitted ambient point and tolerance request a finite distance-to-subset comparison.*
4. *R is the rational-window row through which the distance ledger is read before the terminal value is sealed.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed distance value.*
6. *H records componentwise* hsame *transport for the metric source, located subset, distance ledger, rational windows, real seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from point admission and located-subset reads to distance ledger, rational-window readback, real seal, and proximinal-facing requests.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{ProximinalSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSetDistance}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen nearest point, quotient equality of subset presentations, host infimum, ambient compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6605.2 (Located set distance NameCert obligations)**.** *For an accepted* $\mathsf{LocatedSetDistance}^{\uparrow}$ *carrier* $L = (X, A, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSetDistance}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, located subset, distance ledger, rational-window row, real seal, transport, replay, provenance, and local naming; classifier stability under componentwise* hsame *transport; exact routing through the* $\mathsf{DistanceToSet}^{\uparrow}$ *ledger before the* $\mathsf{Real}^{\uparrow}$ *seal; compatibility with the* $\mathsf{LocatedSet}^{\uparrow}$ *locatedness rows; and non-escape from nearest-point selection, subset quotients, host infima, ambient compactness, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The rows $X, A, D, R, E, H, C, P, N$ are the complete visible inventory, with the metric source and located-subset rows read before any distance value is available.

Use the sibling located-set and distance-to-set surfaces. The row A supplies located-subset admission and comparison data, while D supplies the distance-to-subset ledger over the same metric source X.

The rational-window and real-seal rows preserve that order. A consumer reaches E only after R has displayed the finite readback of D, and the support rows H, C, P, N transport, replay, source, and name exactly this route.

Since no row stores a nearest-point choice, quotient subset equality, host infimum, compactness witness, or full real-completeness theorem, the local NameCert obligations are exhausted by the displayed carrier rows. □

Theorem 6605.3 (Located set distance Real-metric handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-valued distance read exposed by an accepted* $\mathsf{LocatedSetDistance}^{\uparrow}$ *carrier factors through the finite route*

$$X \longmapsto A \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the handoff from $\mathsf{Met}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$ *is a displayed distance ledger over a located subset, not a host infimum, quotient subset value, selected nearest point, ambient completion, or full real-completeness claim.*

Proof. Start with the metric row X. It supplies the admitted ambient points and metric-distance observations used by the distance request.

Read the located subset row A next. Locatedness makes the distance request a subset-facing comparison before any real-valued read is sealed.

Now project D, R, E. The ledger D records the distance-to-set comparison, the rational-window row R exposes its finite readback, and the terminal row E seals that read as $\mathsf{Real}^{\uparrow}$ data.

The support rows H, C, P, N add transport, replay, provenance, and local naming only for the same route. Hence the Real-metric handoff cannot bypass the finite ledger through a host infimum, nearest-point choice, quotient subset value, ambient completion, or real-completeness theorem. □

Theorem 6605.4 (Located set distance proximinal nonescape)**.** *Any* $\mathsf{ProximinalSet}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedSetDistance}^{\uparrow}$ *carrier receives only the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{DistanceToSet}^{\uparrow}$*, rational-window,* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg*, and local* $\mathsf{NameCert}$ *rows. A proximinal request may reuse the distance ledger, but it cannot extract a nearest point, a choice function selecting minimizers, quotient subset equality, compactness of the subset, or completeness of the ambient real line.*

Proof. Project the proximinal-facing request through P and C. The provenance names $\mathsf{ProximinalSet}^{\uparrow}$ only as a consumer boundary, and the continuation row exposes the same route X, A, D, R, E before any proximinal read can inspect the packet.

Apply the NameCert obligations above. They force every distance value to pass through the located subset, distance ledger, rational-window readback, and real seal.

There is no carrier coordinate for a selected minimizer, a choice function, quotient subset equality, compactness of the subset, or ambient real completeness. Therefore proximinal consumers can reuse the finite distance surface but cannot escape to nearest-point data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSetDistance}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSetDistance}^{\uparrow}$ packet is generated from a displayed metric source, located-subset admission, distance-to-set ledger, rational-window readback, real seal, componentwise transport, continuation replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSetDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6605.1, Definition 6605.2, Definition 6605.3, and Definition 6605.4 seed the finite Real-valued distance ledger for located subsets over a displayed metric source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSetDistance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected nearest points, subset quotient equality, host infima, ambient compactness, $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, distance-ledger exactness, Real-metric handoff, proximinal non-escape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 6606

A Concrete Naming Certificate for $\mathsf{CauchyProductTailBudget}^{\uparrow}$

Following Cauchy and Hardy on finite estimates for products of convergent series, the $\mathsf{CauchyProductTailBudget}^{\uparrow}$ packet records the tail-budget surface used before a Cauchy product read may reach a Real-facing endpoint. It sits after the Cauchy-product convergence route of chapter 3608, the product metric route of chapter 3551, the real-series source of chapter 1581, the dyadic rational arithmetic core of chapter 291, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite tail budgets for product estimates; it does not claim selected infinite sums, quotient series equality, choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6606.1 (Cauchy product tail-budget carrier)**.** *A* $\mathsf{CauchyProductTailBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, C, M, S, D, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *A and C are the two* $\mathsf{RealSeries}^{\uparrow}$ *source rows whose finite partial-sum windows are inspected.*
2. *M is the* $\mathsf{CauchyProduct}^{\uparrow}$ *convergence and product-metric row, read through the sibling routes of chapter 3608 and chapter 3551.*
3. *S is the finite tail schedule, naming the left tail, right tail, and mixed finite convolution windows allowed for a requested precision.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the product estimate, including the dyadic radii assigned to the two source tails and to the mixed product window.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed source windows and the tail schedule.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after A, C, M, S, D, Q are visible.*
7. *H records componentwise* hsame *transport for the source series, product row, tail schedule, dyadic ledger, readback, Real seal, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay of the same tail-budget route.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductTailBudget}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductTailBudget}^{\uparrow}}$ *row.*

The classifier reads only these rows. It has no coordinate for a selected infinite sum, quotient equality of series, a choice-selected tail, trichotomy, host multiplication on completed reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6606.2 (Cauchy product tail-budget NameCert obligations)**.** *For every accepted* CauchyProductTailBudg *carrier* $B = (A, C, M, S, D, Q, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductTailBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, two* $\mathsf{RealSeries}^{\uparrow}$ *source exposures, Cauchy-product convergence and metric routing, finite tail-schedule exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6606.1. The two source rows A, C, the product row M, the tail schedule S, and the dyadic ledger D are the only numerical rows available before readback.

The readback row Q is downstream of those finite rows. It repacks displayed source windows and product-tail budgets as regular rational data rather than as quotient series equality or a chosen infinite sum.

Finally inspect E, H, T, P, N. The Real seal E is reached only after the source, product, schedule, dyadic, and readback rows have been displayed, while the structural rows preserve the same packet. Hence the listed rows exhaust the local NameCert obligation surface. □

Theorem 6606.3 (Cauchy product tail-budget Real-seal nonescape)**.** *Every Real-facing read of an accepted* $\mathsf{CauchyProductTailBudget}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{CauchyProduct}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed tail schedule, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows still attached. The route exports no selected infinite sum, quotient series equality, choice-selected tail, trichotomy row, host completed multiplication, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with Definition 6606.2. It makes the two real-series sources, product convergence and metric row, finite tail schedule, dyadic tolerance ledger, and regular readback mandatory before the Real seal can be consumed.

The finite route through S, D, Q controls the tail request. A consumer may inspect only the displayed left, right, and mixed product windows assigned to the requested precision; no coordinate supplies an external tail selector or a host infinite-sum object.

The structural rows H, T, P, N preserve this same route under transport, replay, provenance, and naming. Therefore any Real-facing read reaches E only through the displayed BEDC rows and cannot escape to quotient series equality, choice, trichotomy, host multiplication, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductTailBudget}^{\uparrow}$ endpoint read exposes two RealSeries source rows, one CauchyProduct convergence and metric row, a finite tail schedule, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any product-tail estimate is public.

Independence witness: cauchy-product-convergence, cauchy-product-metric, realseries, dyadicratcore, regseqrat: $\mathsf{CauchyProductTailBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source series, the product convergence and metric route, finite left-right-mixed

tail schedules, dyadic product tolerances, regular readback, terminal Real sealing, transport, replay, provenance, and local naming in one tail-budget packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductTailBudget}\uparrow}$.

Constructive story: A $\mathsf{CauchyProductTailBudget}^{\uparrow}$ packet is generated from two RealSeries source rows, a CauchyProduct convergence and metric row, a finite tail schedule, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductTailBudget}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6606.1, Definition 6606.2, and Definition 6606.3 seed the finite Cauchy-product tail-budget route over displayed series, product, dyadic, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductTailBudget}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected infinite sums, quotient series equality, choice, trichotomy, host completed multiplication, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, product-convergence and metric routing, finite tail-schedule exposure, dyadic tolerance accounting, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductTailBudgetUp.

Chapter 6607

A Concrete Naming Certificate for $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$ packet records the finite cover-refinement surface available for a displayed closed bounded interval once the cover has already been reduced to dyadic mesh rows and finite comparison rows. It sits after the closed bounded interval source of chapter 1591, the finite positive-radius route of chapter 452, the finite comparison packet of chapter 3430, and the terminal real-name seal of chapter 38. The packet names the finite interval cover route only; it is not open-cover compactness, a selected subcover theorem, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6607.1 (Compact interval finite cover-refinement carrier)**.** *A* $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, C, M, L, R, O, S, D, E, H, T, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary;*
2. *C is the finite interval-cover row, listing displayed interval cover members and their local containment witnesses over I;*
3. *M is the dyadic mesh row, recording the finite subdivision cells inspected by the cover comparison;*
4. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius row attached to the displayed cover and mesh route;*
5. *R is the* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *comparison row, pairing each retained mesh cell or refined cover member with its displayed parent cover row;*
6. *O is the finite refinement-order row, recording the inspection order for retained cells, parent cover members, and overlap checks;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect endpoint names, mesh cells, cover witnesses, and retained members;*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row for mesh endpoints, radius comparisons, and containment inequalities;*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, C, M, L, R, O, S, D have been displayed;*
10. *H records componentwise* hsame *transport for the interval, cover, mesh, radius, refinement, order, window, dyadic, and seal rows;*
11. *T records displayed* Cont *replay for interval containment, mesh inspection, finite radius reads, refinement comparison, and Real-facing seal reads;*
12. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, host finite-subcover choice, quotient real equality, countable choice, selected Lebesgue numbers outside the displayed finite row, or ambient Real completeness.

Theorem 6607.2 (Compact interval finite cover-refinement obligations)**.** *For the carrier of Definition 6607.1, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, C, M, L, R, O, S, D, E, H, T, P, N)$, closed-interval source admission through I, finite cover admission through C, dyadic mesh admission through M, finite Lebesgue-number admission through L, finite refinement comparison through R, refinement-order admission through O, stream-window and dyadic readback through S, D, Real seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through T, provenance through P, and local naming through N.*

Proof. Project the accepted carrier row by row. The interval source is I, the cover data is C, the mesh is M, the radius row is L, and the refinement comparison is R with order ledger O.

The finite interval cover is inspected only after the closed bounded interval row has displayed its endpoint and containment data. The mesh row M refines that same source interval, so every retained cover member is checked inside the interval packet rather than inside a host set of reals.

The radius and refinement rows are then read together. The finite Lebesgue-number row L supplies the visible positive-radius ledger, while R and O record how the retained mesh cells refine the displayed cover rows.

Finally S, D, E, H, T, P, N provide window inspection, dyadic arithmetic, the terminal Real seal, transport, replay, provenance, and local naming. Since no excluded compactness, selection, quotient, choice, or completeness coordinate occurs in the carrier, the listed rows exhaust the NameCert obligation surface. □

Theorem 6607.3 (Compact interval finite cover-refinement route)**.** *Every Real-facing or compact-interval consumer of an accepted* $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto C, M \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow}, \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto S, D, E \longmapsto H, T, P, N.$$

Thus the consumer receives finite interval cover rows, dyadic mesh rows, finite refinement order, and the $\mathsf{Real}^{\uparrow}$ *seal boundary before it may use the packet as compact-interval cover-refinement data. The route exports no open-cover compactness theorem, choice-selected subcover, quotient equality, countable choice, selected external Lebesgue number, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the closed interval source. The consumer first reads I, so endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal Real boundary are already visible before cover data is opened.

Next inspect the cover and mesh rows C, M. Induction over the finite mesh rows checks each displayed cell against the finite cover row and records the retained member before any order count is consumed.

The positive-radius and refinement rows are then composed. The row L supplies the finite radius ledger, and R, O project each retained mesh or refined cover member back to its displayed parent cover row in finite order.

The terminal rows S, D, E, H, T, P, N keep the same route visible to Real-facing consumers. They seal the displayed finite route without adding arbitrary open covers, selected subcovers, quotient real equality, countable choice, external radius selection, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, finite interval-cover rows, dyadic mesh rows, a FiniteLebesgueNumberUp radius row, a FiniteCoverRefinementUp comparison row, a finite refinement-order ledger, StreamName windows, DyadicRatCore observations, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6607.1, Definition 6607.2, and Definition 6607.3 seed the finite compact-interval cover-refinement route over displayed interval, mesh, radius, refinement, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteCoverRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient real equality, countable choice, external Lebesgue-number selection, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite interval-cover admission, dyadic mesh stability, finite radius/refinement compatibility, refinement-order transport, and Real seal non-escape.

Chapter 6608

A Concrete Naming Certificate for $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$

Following Pontryagin, Boltyanskii, Gamkrelidze, and Mishchenko, and the optimal-control textbook route through Hamiltonian stationarity, the $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ packet records only the finite certificate surface by which a control problem exposes state dynamics, endpoint cost, adjoint rows, a Hamiltonian comparison, and transversality data before any global existence theorem or selected optimizer is read. It sits after the differential-equation route of chapter 193, the dynamic-system rows of chapter 195, the derivative and variational rows of chapter 3632, the convex control rows of chapter 208, the scalar seal of chapter 38, and the finite vector-space rows of chapter 47. The packet names a displayed maximum-principle handoff; it does not assert existence of optimal controls, measurable-selection closure, smoothness beyond the displayed rows, host Euler–Lagrange completeness, or a global phase-space theorem.

Definition 6608.1 (Pontryagin maximum-principle carrier)**.** *A* $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, U, F, J, A, H, T, R, V, Q, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Ode}^{\uparrow}$ *state-equation row, recording the finite state window, endpoint rows, and local flow witnesses under inspection.*
2. *U is the admissible-control row, read through* $\mathsf{ConvexSet}^{\uparrow}$ *when the inspected control window carries convex constraints and otherwise only as the displayed finite admissibility ledger.*
3. *F is the* $\mathsf{DynSystem}^{\uparrow}$ *rollout row tying the state and control windows to the same finite transition route.*
4. *J is the* $\mathsf{Real}^{\uparrow}$ *cost row, storing the displayed objective, endpoint contribution, and finite comparison budget.*
5. *A is the adjoint row. It records the covector or multiplier window in the same* $\mathsf{VecSpace}^{\uparrow}$ *coordinates as the state and control rows.*
6. *H is the Hamiltonian row, storing the finite expression obtained from the state, control, cost, and adjoint rows.*
7. *T is the transversality row, containing only the displayed endpoint boundary relation attached to X, J, A.*

8. *R is the Real-seal row for the Hamiltonian comparison and endpoint cost comparison.*
9. *V stores componentwise* hsame *transport for state, control, rollout, cost, adjoint, Hamiltonian, transversality, Real-seal, replay, provenance, and local naming rows.*
10. *Q stores displayed* Cont *replay from an optimal-control query through $X, U, F, J, A, H, T, R, V$.*
11. *C is the convexity and local-variation ledger, recording only the finite variation rows needed to compare admissible controls in the displayed window.*
12. *G is the* Pkg *provenance over* $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PontryaginMaximumPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected optimal control, a host measurable-selection theorem, ambient smooth manifold completeness, quotient trajectory equality, or a global existence theorem.

Theorem 6608.2 (Pontryagin maximum-principle NameCert obligations)**.** *For every accepted* $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ *carrier*

$$P = (X, U, F, J, A, H, T, R, V, Q, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PontryaginMaximumPrinciple}^{\uparrow}}$ *surface consists exactly of state-equation admission, admissible-control exposure, dynamic rollout, Real-valued cost readback, adjoint-row exposure, Hamiltonian construction, transversality exposure, finite variation comparison, Real-seal ordering, componentwise transport, displayed replay, provenance, local naming, and nonescape for selected optimizers, host measurable selection, quotient trajectory equality, ambient smoothness, and global existence.*

Proof. Project Definition 6608.1. The accepted inventory is $X, U, F, J, A, H, T, R, V, Q, C, G, N$, so every readable optimal-control datum is one of those displayed rows.

The analytic route is finite. The state and rollout rows X, F are read before the control and cost rows U, J can be compared, and the adjoint row A is attached to the same vector-space and calculus windows. The Hamiltonian row H is then computed only from those already displayed rows.

The endpoint and stationarity boundaries are also local. Row T records the transversality relation, row C records the finite variation ledger, and R seals only the displayed Real comparisons. The structural rows V, Q, G, N transport, replay, source, and name that route without adding selected optimizers, measurable selection, quotient trajectories, ambient smoothness, or global existence. □

Theorem 6608.3 (Pontryagin maximum-principle Hamiltonian handoff)**.** *Every accepted* PontryaginMaximumPrinc *packet reads a Hamiltonian stationarity surface only through the finite route*

$$\mathsf{Ode}^{\uparrow} \longrightarrow \mathsf{DynSystem}^{\uparrow} \longrightarrow \mathsf{Calculus}^{\uparrow} \longrightarrow \mathsf{ConvexSet}^{\uparrow} \longrightarrow \mathsf{VecSpace}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In carrier notation, the Hamiltonian row H is admissible only after the state-equation row X, admissible-control row U, rollout row F, cost row J, adjoint row A, variation ledger C, and terminal Real-seal row R have been displayed. The handoff exports no selected optimizer, no host maximum theorem, no measurable-selection closure, and no global phase-space completion.

Proof. Start from an accepted carrier $P = (X, U, F, J, A, H, T, R, V, Q, C, G, N)$. The state endpoint is X, and the finite dynamic rollout is F, so a stationarity read cannot begin from an unrecorded phase-space object.

Read the control, cost, and adjoint data next. The admissible-control row U is constrained by the displayed convex or finite admissibility ledger, J stores the Real-valued objective window, and A stores the adjoint data in the same vector-space coordinates used by the state rows.

The Hamiltonian handoff is then the displayed row H together with the variation ledger C, transversality row T, and Real seal R. Since the carrier contains no selected optimizer, measurable-selection theorem, host maximum theorem, or global completion row, the exported stationarity surface is exactly the finite route stated above. □

Falsifiable prediction: Every $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ consumer that reaches Hamiltonian stationarity must first display state-equation, admissible-control, dynamic-rollout, Real-cost, adjoint, Hamiltonian, transversality, variation-ledger, Real-seal, transport, replay, provenance, and local NameCert rows.

Independence witness: ode, dynsystem, calculus, convexset, real, vecspace: $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a state equation, control constraint, dynamic rollout, Real cost, adjoint row, Hamiltonian row, transversality boundary, variation ledger, structural replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PontryaginMaximumPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{PontryaginMaximumPrinciple}^{\uparrow}$ packet is generated from an OdeUp state row, a DynSystemUp rollout row, a displayed admissible-control row, a RealUp cost row, a CalculusUp variation row, a VecSpaceUp adjoint row, a Hamiltonian row, transversality data, Real-seal ordering, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PontryaginMaximumPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6608.1, Definition 6608.2, and Definition 6608.3 seed the traditional optimal-control maximum-principle packet over state dynamics, controls, adjoints, Hamiltonian comparison, endpoint data, and finite variation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PontryaginMaximumPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close existence of optimal controls, measurable-selection closure, global smooth phase-space theorems, host maximum principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, state-equation exposure, admissible-control exposure, rollout exactness, cost readback, adjoint exposure, Hamiltonian handoff, transversality exposure, variation-ledger exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PontryaginMaximumPrincipleUp.

Chapter 6609

A Concrete Naming Certificate for $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$

The $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ packet realizes the joint visual-metric, geodesic-flow, boundary-shadow, observer-expansion, and Real-seal route of chapter 9463 as finite NameCert data. It sits after the visual-metric packet of chapter 3866, the geodesic-flow packet of chapter 3856, the boundary-shadow route of chapter 3813, the observer-expansion packet of chapter 4167, and the terminal scalar seal of chapter 38. The packet names a finite visual-flow shadow surface; it is not a quotient boundary, a chosen geodesic ray, a global hyperbolic model, a physical observer, or an ambient Real-completeness claim.

Definition 6609.1 (Hyperbolic visual-flow shadow carrier)**.** *A* $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, B, O, R, T, Q, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row, recording the visible metric comparison and basepoint window under inspection.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, storing the finite flow segment and its endpoint ledger.*
3. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, exposing only the displayed boundary-shadow witness attached to the same metric and flow window.*
4. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row, recording the observer-expansion readout that consumes the visual metric, flow, and boundary-shadow data.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the visual-flow comparison and the expansion budget.*
6. *T stores componentwise* hsame *transport for visual metric, geodesic-flow, boundary-shadow, observer-expansion, Real-seal, replay, provenance, and local naming rows.*
7. *Q stores displayed* Cont *replay from a visual-flow request through M, F, B, O, R, T.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient ideal boundary, selected global ray, completed hyperbolic model, physical observer, or ambient Real completion theorem.

Theorem 6609.2 (Hyperbolic visual-flow shadow obligation)**.** *For every accepted* $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ *carrier* $S = (M, F, B, O, R, T, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}}$ *surface consists exactly of visual-metric admission, geodesic-flow exposure, boundary-shadow exposure, observer-expansion exposure, Real-seal ordering, componentwise transport, displayed replay, provenance, local naming, and nonescape for quotient boundaries, selected rays, global hyperbolic models, physical observers, and ambient Real completeness.*

Proof. Project Definition 6609.1. The metric, flow, boundary, observer, and Real rows are the only analytic coordinates in the carrier.

The visual route starts with M and F. The boundary-shadow row B is accepted only as the displayed shadow attached to that same visual metric and finite geodesic-flow segment, so it cannot introduce an independent quotient boundary or selected ray.

The observer-expansion and Real-facing rows are downstream. Row O reads the joint visual metric, flow, and shadow surface, while R seals only the displayed comparison. The rows T, Q, P, N transport, replay, source, and name the same finite packet, so no global hyperbolic model, physical observer, or ambient Real-completeness theorem can be exported. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ consumer must display visual-metric, geodesic-flow, boundary-shadow, observer-expansion, Real-seal, transport, replay, provenance, and local naming rows before reading a visual-flow shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-geodesic-flow, hyperbolic-boundary-shadow, hyperbolic-observer-expansion: $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries visual metric, geodesic flow, boundary shadow, observer expansion, Real sealing, structural replay, provenance, and local naming.

The later Lean TasteGate route for `BEDC.Derived.HyperbolicVisualFlowShadowUp.TasteGate` must include Nontrivial $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ instances preserving $M, F, B, O, R, T, Q, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$ packet is generated from a HyperbolicVisualMetricUp row, a HyperbolicGeodesicFlowUp row, a HyperbolicBoundaryShadowUp row, a HyperbolicObserverExpansionUp row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}}$, seed)

Scope closed: Definition 6609.1 and Definition 6609.2 seed the finite visual-flow shadow packet over visual metric, geodesic-flow, boundary-shadow, observer-expansion, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient ideal boundaries, selected global rays, physical observer semantics, global hyperbolic models, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-metric exposure, geodesic-flow exposure, boundary-shadow exposure, observer-expansion exposure, Real-seal ordering, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicVisualFlowShadowUp.

Chapter 6610

A Concrete Naming Certificate for $\mathsf{KripkeSpread}^{\uparrow}$

Following Kripke on intuitionistic models and Brouwer on spreads, the $\mathsf{KripkeSpread}^{\uparrow}$ packet records finite spread inspection along the L10 countable-name route. It sits beside the binary prefix surface of chapter 1475, the finite-bar discipline of chapter 400, the finite branching carrier of chapter 6488, and the countable-name consumers of chapter 126 and chapter 301. The packet exposes only finite nodes, finite successor ledgers, prefix persistence, and readback windows; it is not weak Koenig path selection, countable choice, quotient stream equality, or completed Real data.

Definition 6610.1 (Kripke spread carrier)**.** *A* $\mathsf{KripkeSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, N, B, A, S, Q, O, H, C, P, L)$$

with the following displayed rows:

1. *W is the finite world-prefix ledger, listing the inspected nodes and their prefix order.*
2. *N is the decidable node-admission row, recording which displayed prefixes belong to the spread.*
3. *B is the finite branching row, recording the displayed successor family for each inspected node.*
4. *A is the persistence row, requiring every displayed successor to retain the admitted prefix data of its predecessors.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which countable-name reads inspect the spread.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after the finite StreamName windows have been displayed.*
7. *O is the refusal row excluding weak Koenig path selection, countable choice, quotient stream equality, unbounded branch search, and completed* $\mathsf{Real}^{\uparrow}$ *closure.*
8. *H records componentwise* hsame *transport for worlds, node admission, finite branching, persistence, StreamName windows, RegSeqRat readback, refusal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a finite spread-inspection request through $W, N, B, A, S, Q, O, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{KripkeSpread}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{FinitelyBranchingTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{KripkeSpread}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these finite rows form one spread-inspection packet. It has no coordinate for a selected infinite branch, a total path functional, countable choice, quotient stream equality, ambient compactness, or completed real closure.

Theorem 6610.2 (Kripke spread NameCert obligations)**.** *For every accepted* $\mathsf{KripkeSpread}^{\uparrow}$ *carrier* $K = (W, N, B, A, S, Q, O, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{KripkeSpread}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite world-prefix ledger, node-admission row, finite branching row, persistence row, StreamName window schedule, RegSeqRat readback row, refusal row, componentwise transport, displayed replay, provenance, and local naming. Every L10 countable-name consumer must read the finite route*

$$W, N, B, A, S, Q, O, H, C, P, L$$

before extracting StreamName or RegSeqRat data, and no obligation is stored outside those rows.

Proof. Project the carrier of Definition 6610.1. The displayed finite coordinates are precisely $W, N, B, A, S, Q, O, H, C, P, L$.

The world, admission, branching, and persistence rows give the spread-inspection content. They record inspected prefixes, admitted nodes, displayed successors, and predecessor retention without creating an infinite path.

The StreamName and RegSeqRat rows are downstream. A countable-name consumer first opens the finite window schedule S, then reads the regular-rational row Q attached to that window.

Finally O, H, C, P, L keep the refusal boundary, transport, replay, provenance, and local name attached to the same packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6610.3 (Kripke spread prefix non-escape)**.** *Every accepted* $\mathsf{KripkeSpread}^{\uparrow}$ *consumer that inspects StreamName or RegSeqRat data must remain inside the finite route*

$$W, N, B, A, S, Q, O, H, C, P, L.$$

The consumer may reuse finite Cantor-space prefix observations, finite FanTheorem bar data, finite branching-tree rows, StreamName windows, and RegSeqRat readback data, but it receives no weak Koenig path selector, no countable-choice functional, no quotient stream equality, no unbounded branch search, and no completed $\mathsf{Real}^{\uparrow}$ *closure.*

Proof. Begin with a StreamName or RegSeqRat read of an accepted carrier. The carrier exposes countable-name data only through the finite window row S and the readback row Q.

The prefix rows W, N, B, A remain visible before that read. They supply admitted nodes and displayed successors as finite inspection data, not as a completed branch through the spread.

The refusal row O is part of the same packet. It blocks any reinterpretation of the finite route as weak Koenig path selection, countable choice, quotient stream equality, unbounded branch search, or completed real closure.

Transport, replay, provenance, and local naming through H, C, P, L preserve that boundary. Thus every consumer stays inside the finite prefix route and none of the excluded principles is exported. □

Falsifiable prediction: A $\mathsf{KripkeSpread}^{\uparrow}$ consumer that extracts a weak Koenig path selector, countable-choice functional, quotient stream equality, unbounded branch search, or completed $\mathsf{Real}^{\uparrow}$ closure without first reading $W, N, B, A, S, Q, O, H, C, P, L$ refutes the carrier surface.

Independence witness: cantorspace, fantheorem, finitely-branching-tree, streamname, regseqrat: $\mathsf{KripkeSpread}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite world-prefix inspection, node admission, finite branching, persistence, StreamName windows, RegSeqRat readback, refusal, transport, replay, provenance, and local naming while refusing selected branches and completed real data.

Closure status for $\mathsf{NameCert}_{\mathsf{KripkeSpread}^{\uparrow}}$.

Constructive story: A $\mathsf{KripkeSpread}^{\uparrow}$ packet is generated from finite world-prefix rows, node admission, finite branching, persistence, StreamName windows, RegSeqRat readback, an explicit refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KripkeSpread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6610.1, Definition 6610.2, and Definition 6610.3 seed the finite spread-inspection boundary for L10 StreamName and RegSeqRat consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KripkeSpread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak Koenig path selection, selected infinite branches, countable choice, quotient stream equality, unbounded branch search, completed $\mathsf{Real}^{\uparrow}$ closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, node-admission exactness, finite branching coverage, prefix persistence, StreamName window handoff, RegSeqRat readback, transport, replay, provenance, and non-escape for BEDC.Derived.KripkeSpreadUp.

Chapter 6611

A Concrete Naming Certificate for $\mathsf{ChoquetSimplex}^{\uparrow}$

Following Choquet on compact convex sets and the Bishop–Bridges constructive reading of probability measures, the $\mathsf{ChoquetSimplex}^{\uparrow}$ packet records the boundary-facing finite NameCert surface for compact convex probability carriers. It sits after the compact source of chapter 60, the convex carrier of chapter 208, the Radon-measure route of chapter 2263, and the locally convex source of chapter 3386. The chapter names only the carrier rows needed by a compact-convex probability-boundary consumer; it does not assert representing-measure uniqueness, full Krein–Milman strength, selected extreme points, countable choice, quotient measure equality, or ambient locally convex completeness.

Definition 6611.1 (Choquet simplex carrier surface)**.** *A* $\mathsf{ChoquetSimplex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, V, P, B, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *K is the compact source row, read through the compact carrier and compact-metric observation surfaces.*
2. *V is the convex carrier row, including the displayed affine-combination and convex-subcarrier data visible to the packet.*
3. *P is the probability row, recording finite probability-window requests over the compact convex source.*
4. *B is the boundary-window row, naming the inspected extreme-boundary candidates only as finite observations.*
5. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *readback row available after the compact, convex, probability, and boundary-window rows have been displayed.*
6. *E is the locally convex exposure row, read through* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *seminorm and neighbourhood observations needed by affine consumers.*
7. *H records componentwise* hsame *transport for compact, convex, probability, boundary, Radon, locally convex, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-convex probability-boundary request through the same rows.*

9. *Q is the* Pkg *provenance over* $\mathsf{ChoquetSimplex}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoquetSimplex}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, P, B, R, E, H, C, Q, N$. It has no coordinate for a host simplex object, selected barycentric measure, representing-measure uniqueness, full extreme-point generation, quotient measure equality, countable choice, or ambient locally convex completeness.

Lean4 ✓ `BEDC.Derived.ChoquetSimplexUp.ChoquetSimplexTasteGate single carrier alignment`

Theorem 6611.2 (Choquet simplex certificate surface)**.** *For every accepted* $\mathsf{ChoquetSimplex}^{\uparrow}$ *carrier $S = (K, V, P, B, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoquetSimplex}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, convex-carrier admission through V, probability-window admission through P, boundary-window exposure through B, Radon readback through R, locally convex exposure through E, componentwise transport through H, displayed replay through C, provenance through Q, and local naming through N. Any probability-boundary consumer must read those rows before using the simplex packet.*

Proof. Project the accepted carrier of Definition 6611.1. The compact and convex rows are the only source rows, so a consumer first reads compact observations through K and affine convex data through V.

The probability and boundary rows are displayed next. The row P records finite probability-window requests, while B records the inspected boundary candidates as finite observations rather than selected extreme points.

The Radon and locally convex rows are downstream. The readback row R is available only after the compact, convex, probability, and boundary rows are visible, and E exposes the locally convex observation surface used by affine consumers.

Finally the structural rows H, C, Q, N transport, replay, source, and name exactly this route. Since no row stores representing-measure uniqueness, a selected barycentric measure, full extreme-point generation, quotient measure equality, countable choice, or ambient locally convex completeness, the listed rows exhaust the certificate surface. □

Falsifiable prediction: Every $\mathsf{ChoquetSimplex}^{\uparrow}$ probability-boundary consumer reads compact rows, convex rows, finite probability windows, boundary observations, Radon readback, locally convex exposure, transport, replay, provenance, and local naming before it exports a simplex-facing certificate.

Independence witness: compact, convexset, radon-measure, locally-convex-space: $\mathsf{ChoquetSimplex}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact source rows, convex affine rows, probability windows, boundary observations, Radon readback, locally convex exposure, transport, replay, provenance, and local NameCert naming while refusing selected representation data.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetSimplex}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoquetSimplex}^{\uparrow}$ packet is generated from compact source rows, convex affine rows, finite probability-window rows, boundary observations, Radon readback, locally convex exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoquetSimplex}^{\uparrow}}$, seed)

Scope closed: Definition 6611.1 and Definition 6611.2 seed the compact-convex probability-boundary certificate over compact, convex, Radon, locally convex, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoquetSimplex}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close representing-measure uniqueness, full Krein–Milman generation, selected extreme points, quotient measure equality, countable choice, ambient locally convex completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, convex affine exposure, probability-window exactness, boundary-window readback, Radon handoff, locally convex exposure, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetSimplexUp.

Chapter 6612

A Concrete Naming Certificate for $\mathsf{DyadicOrderTopology}^{\uparrow}$

Following Bishop and Bridges, the order topology of the dyadic rationals is read through located rational intervals and common dyadic refinements. The $\mathsf{DyadicOrderTopology}^{\uparrow}$ packet sits over the dyadic arithmetic source of chapter 291, the rational comparison source of chapter 37, and the order interface of chapter 1378; it presents its interval-window and basis-refinement rows to the metric route of chapter 57 and the real-name seal of chapter 38. The packet names only the finite order-topology surface needed before $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers read the first licensed infinite real name.

Definition 6612.1 (DyadicOrderTopology carrier)**.** *A* $\mathsf{DyadicOrderTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, O, I, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, including dyadic endpoints, common denominators, comparison budgets, and refinement witnesses.*
2. *O is the order row over the same dyadic source, recording the rational comparison and endpoint-order data inherited from* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Order}^{\uparrow}$*.*
3. *I is a finite interval-window row, listing located rational intervals with dyadic endpoints and the membership comparisons used by the topology basis.*
4. *B is the basis-refinement row, recording finite common dyadic refinements of overlapping interval windows.*
5. *H records componentwise* hsame *transport for D, O, I, B, C, P, N.*
6. *C records the displayed* Cont *replay from dyadic endpoints through order comparison, interval-window inspection, basis refinement, and consumer handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicOrderTopology}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only by these displayed rows. It has no coordinate for trichotomy beyond the displayed order row, arbitrary Dedekind cuts, quotient stream equality, selected limits, ambient real completeness, or host topological equality.

Lean4 (statement) `BEDC.Derived.DyadicOrderTopologyUp`

Theorem 6612.2 (DyadicOrderTopology basis stability)**.** *For an accepted* $\mathsf{DyadicOrderTopology}^{\uparrow}$ *carrier* $T = (D, O, I, B, H, C, P, N)$*, every pair of interval-window reads that overlap through the displayed order row admits a common dyadic refinement read through* B*. The refinement preserves the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row* D*, the same* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Order}^{\uparrow}$ *comparison data in* O*, and the same transport, replay, provenance, and local NameCert rows* H, C, P, N*.*

Proof. Project the accepted carrier onto D, O, I, B, H, C, P, N. The interval-window row I contains only finite dyadic endpoint data, and the order row O supplies the displayed comparisons used to test overlap.

Use common dyadic refinement. Given two displayed interval windows over the same source row, refine their endpoints to a common dyadic denominator and read the resulting subwindow through B. The comparison data remain those of O, because the refinement is a dyadic re-expression of the same rational interval order.

Finally transport and replay the refined row through H and C. Provenance and local naming are unchanged, so the common refinement remains inside the same $\mathsf{DyadicOrderTopology}^{\uparrow}$ packet and exports no selected real point, quotient stream equality, or ambient completeness principle. □

Theorem 6612.3 (DyadicOrderTopology located interval window)**.** *Every* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicOrderTopology}^{\uparrow}$ *carrier must read a located interval window through*

$$D \longrightarrow O \longrightarrow I \longrightarrow B \longrightarrow H, C, P, N.$$

The consumer may use the finite dyadic basis refinement as a topology source surface, but it receives no arbitrary open-set enumeration, no quotient equality of real names, no selected limit, no trichotomy outside O*, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the carrier rows. The dyadic source D and order row O are upstream of every interval-window row, so a consumer cannot inspect I without first displaying the dyadic endpoint and rational comparison data.

Apply basis stability. Any overlap or shrinking request is handled by a common dyadic refinement in B, still over the same D and O rows.

The structural rows H, C, P, N then carry the finite route to metric, regular-sequence, and real-name consumers. Since the carrier contains no row for arbitrary open-set enumeration, quotient real-name equality, selected limits, external trichotomy, or ambient real completeness, those objects cannot be exported by the located interval window. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicOrderTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicOrderTopology}^{\uparrow}$ packet is generated from DyadicRatCore endpoint data, rational order comparisons, located interval windows, common dyadic basis refinements, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicOrderTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6612.1, Definition 6612.2, and Definition 6612.3 seed the finite dyadic order-topology interface over displayed dyadic endpoint, rational order, interval-window, and basis-refinement rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicOrderTopology}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.DyadicOrderTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim checked carrier obligations, arbitrary open-set enumeration, quotient equality of real names, selected limits, trichotomy outside the displayed order row, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order-row stability, interval-window readback, common dyadic basis refinement, metric consumer handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicOrderTopologyUp.

Chapter 6613

A Concrete Naming Certificate for $\mathsf{RealBorelCode}^{\uparrow}$

Following Bishop and Bridges and descriptive-set-theoretic Borel-code practice, the $\mathsf{RealBorelCode}^{\uparrow}$ packet records constructive Borel codes over the existing $\mathsf{Real}^{\uparrow}$ naming surface as finite certificate data. It sits beside the regular-real carrier of chapter 38, the finite window route of chapter 126, the regular rational readback of chapter 301, and the dyadic endpoint arithmetic of chapter 291. The packet names code rows and their displayed interpretation schedules; it does not import host set equality, quotient equality of real names, impredicative sigma algebras, or a completed external powerset of $\mathsf{Real}^{\uparrow}$.

Definition 6613.1 (RealBorelCode carrier)**.** *A* $\mathsf{RealBorelCode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, S, R, D, T, U, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite code-index ledger, listing exactly the inspected code rows and their immediate predecessor references.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule through which real-name consumers inspect only finite prefixes of the real input.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to compare inspected windows with rational endpoint data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, carrying displayed lower and upper dyadic rational bounds for basic real cylinders.*
5. *T is the constructor-tag row. Its admitted tags are basic-cylinder, complement, and countable-coded union; the union tag stores only a displayed schedule of requested child rows, not an external subset family.*
6. *U is the finite interpretation schedule saying how a basic cylinder reads S, R, D, how complement flips an already displayed child decision row, and how a countable-coded union exposes the finite child windows demanded by the current query.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary: every interpretation row reaches the real-name seal only through the displayed StreamName, RegSeqRat, and DyadicRatCore rows.*

8. *H records componentwise* hsame *transport for code indices, StreamName windows, regular rational readback, dyadic endpoints, constructor tags, interpretation schedules, Real-seal rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a finite code-inspection request through $I, S, R, D, T, U, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealBorelCode}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBorelCode}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive Borel-code packet. It has no coordinate for host membership in an external set of reals, quotient equality of real names, countable choice over code branches, or an ambient Boolean algebra of subsets.

Theorem 6613.2 (RealBorelCode NameCert obligations)**.** *For every accepted* $\mathsf{RealBorelCode}^{\uparrow}$ *carrier $B = (I, S, R, D, T, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealBorelCode}^{\uparrow}}$ *surface consists exactly of carrier admission for finite code indices, StreamName window exposure, RegSeqRat readback, DyadicRatCore endpoint comparison, constructor-tag admission, displayed interpretation schedules, Real-seal discipline, componentwise transport, displayed continuation replay, provenance, and local naming. Every code consumer must display*

$$I, S, R, D, T, U, E, H, C, P, N$$

before it may read the interpretation of a code row on a $\mathsf{Real}^{\uparrow}$ *name.*

Proof. Project the carrier of Definition 6613.1. The rows $I, S, R, D, T, U, E, H, C, P, N$ are the only accepted coordinates, so the NameCert obligations are read from those coordinates.

The basic-cylinder obligation is routed through S, R, D: a finite real-name window is compared with displayed dyadic endpoints by regular rational readback. Complement and countable-coded union use T and U to replay already displayed child rows or the finite child windows requested by the present query.

The Real-facing row E is reached only after the StreamName, RegSeqRat, and DyadicRatCore rows have been displayed. The structural rows H, C, P, N transport, replay, source, and locally name the same packet, leaving no further obligation outside the listed rows. □

Theorem 6613.3 (RealBorelCode cylinder stability)**.** *Every basic-cylinder row of an accepted* $\mathsf{RealBorelCode}^{\uparrow}$ *packet is stable under* hsame *transport of the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints. If two inspected real-name windows are transported componentwise through the packet support, the same basic-cylinder decision schedule is replayed by* Cont *and the same constructor tag remains accepted.*

Proof. Read the basic-cylinder branch of the constructor row T. Its interpretation schedule U names only the inspected StreamName window, the regular rational readback row, and the dyadic endpoint comparison.

Apply the componentwise transport row H to those three pieces. The transported window, readback, and endpoint rows remain attached to the same code index in I, and the local name row N keeps the carrier coordinate fixed.

The replay row C then re-executes the basic-cylinder schedule with the transported data. Since the packet contains no host real equality or external set membership coordinate, the stable output is exactly the displayed cylinder decision row. □

Theorem 6613.4 (RealBorelCode Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealBorelCode}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The complement and countable-coded union tags can replay only displayed child schedules and finite query windows before the terminal seal row E is consumed. They export no quotient equality of real names, host set equality, countable-choice branch selector, external sigma algebra, or ambient powerset object.

Proof. Start from an accepted packet $B = (I, S, R, D, T, U, E, H, C, P, N)$. The only Real-facing row is E, and it is named after S, R, D, T, U in the carrier.

For a basic-cylinder read, the route reaches E through the inspected window, regular rational comparison, and dyadic endpoints. For a complement read, U replays the displayed child schedule and flips only that local decision row.

For a countable-coded union read, the schedule exposes the finite child windows demanded by the current query. The row records no external infinite branch selector and no host subset family.

Thus every Real-facing consumer remains inside the displayed finite route. The refused quotient, host equality, choice, sigma-algebra, and powerset objects have no coordinate in the carrier. □

Falsifiable prediction: A $\mathsf{RealBorelCode}^{\uparrow}$ consumer that evaluates a code row on a $\mathsf{Real}^{\uparrow}$ name without first displaying $I, S, R, D, T, U, E, H, C, P, N$ refutes the carrier surface.

Independence witness: real, streamname, regseqrat, dyadicratcore: $\mathsf{RealBorelCode}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite code indices, constructor tags, interpretation schedules, StreamName windows, RegSeqRat readback, DyadicRatCore endpoints, Real-seal discipline, transport, replay, provenance, and local naming while refusing host set equality and quotient real equality.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBorelCode}^{\uparrow}}$.

Constructive story: A $\mathsf{RealBorelCode}^{\uparrow}$ packet is generated from finite code-index rows, StreamName windows, RegSeqRat readback, DyadicRatCore endpoints, constructor tags for basic cylinders, complement, and countable-coded union, displayed interpretation schedules, Real-seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealBorelCode}^{\uparrow}}$, seed)

Scope closed: Definition 6613.1, Definition 6613.2, Definition 6613.3, and Definition 6613.4 seed the constructive Borel-code boundary over RealUp names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealBorelCode}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host set equality, quotient equality of real names, countable-choice branch selection, external sigma-algebra completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constructor-tag exactness, basic-cylinder stability, complement replay, countable-coded-union window coverage, Real-seal non-escape, transport, provenance, and local naming for BEDC.Derived.RealBorelCodeUp.

Chapter 6614

A Concrete Naming Certificate for $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$

Following Bishop–Bridges and the finite-net proof of uniform continuity on compact metric spaces, the $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ packet realizes the route of chapter 9237 as finite NameCert data for families of maps. It sits over the compact source packet of chapter 129, the map surface of chapter 130, the tolerance-only handoff of chapter 369, the family stability surface of chapter 4076, and the rational fold carrier of chapter 37. The packet records a finite router from compact nets and local family moduli to a public uniform-modulus row; it does not import open-cover compactness, sequential compactness, host function equality, or a choice-selected center family.

Definition 6614.1 (Compact family modulus router carrier)**.** *A* $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, E, L, Q, D, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including displayed centers, cover radii, and located source membership witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, listing the inspected map names and their pointwise modulus rows over the centers named by K.*
3. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *compatibility row, requiring the family pointwise moduli to share the same tolerance request and source metric comparison.*
4. *L is the local finite-net lookup row sending a tolerance history to the displayed center-radius rows relevant to the family request.*
5. *Q is the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the displayed centerwise radii.*
6. *D is the dyadic tolerance ledger used for halving, positivity, lower-bound transport, and metric-triangle budget checks.*
7. *T is the public* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, depending only on the tolerance history and the displayed router packet.*
8. *H records componentwise* hsame *transport for compact nets, family map rows, equicontinuity rows, local lookups, rational folds, dyadic budgets, uniform-modulus handoffs, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a tolerance request through $K, F, E, L, Q, D, H, P, N$ to the handoff row T.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFamilyModulusRouter}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected centers, external function equality, countable choice over a map family, or an ambient completeness principle.

Theorem 6614.2 (Compact family modulus router NameCert obligations)**.** *For every accepted* $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ *carrier $R = (K, F, E, L, Q, D, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactFamilyModulusR}}$ *surface consists exactly of compact finite-net exposure, family ContinuousMap pointwise-modulus exposure, Equicontinuity compatibility, finite lookup exactness, RatUp lower-bound fold exactness, dyadic tolerance transport, UniformModulus handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every theorem-facing read must display*

$$K, F, E, L, Q, D, H, C, P, N$$

before it may consume T.

Proof. Project the carrier of Definition 6614.1. Its accepted rows are exactly the compact finite-net, family pointwise-modulus, equicontinuity, lookup, rational fold, dyadic budget, uniform handoff, and support rows.

The source of the router is K, F, E, L. These rows display the compact centers, the inspected family map names, the pointwise radii attached to those centers, and the compatibility data that lets all maps in the displayed family consume the same tolerance request.

The numerical route is Q, D. The finite $\mathsf{Rat}^{\uparrow}$ fold records a lower-bound history for the displayed radii, while the dyadic ledger transports positivity, halving, and metric-triangle budget checks.

Finally T, H, C, P, N export the tolerance-only handoff and preserve the packet under transport and replay. No open-cover, choice-selected, host-function, or completeness coordinate exists outside those rows. □

Theorem 6614.3 (Compact family finite-net routing)**.** *Every accepted* $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ *packet turns the compact-uniform-continuity route of chapter 9237 into a finite router*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Equicontinuity}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The output $\mathsf{UniformModulus}^{\uparrow}$ *row is a tolerance-only read from the displayed compact finite-net, family pointwise-modulus rows, equicontinuity compatibility, rational lower-bound fold, and dyadic tolerance ledger. It exports no open-cover compactness, selected center family, host equality of maps, countable choice, or ambient completeness.*

Proof. Begin with an accepted router $R = (K, F, E, L, Q, D, T, H, C, P, N)$. The compact row K supplies displayed centers and finite coverage data, and the family row F attaches pointwise modulus rows to those centers.

Use E to synchronize those pointwise rows across the inspected family. The lookup row L then selects only displayed center-radius data for the current tolerance request.

Fold the displayed radii through Q and transport the tolerance budget through D. This produces a lower-bound history depending on the tolerance and the finite router packet, not on an external center selector.

The handoff row T is therefore a public $\mathsf{UniformModulus}^{\uparrow}$ row. The support rows preserve the same finite route, so the excluded compactness, choice, host-equality, and completeness objects are not exported. □

Theorem 6614.4 (Compact family tolerance nonescape)**.** *Every tolerance read exported by an accepted* $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ *packet reaches* $\mathsf{UniformModulus}^{\uparrow}$ *only through the displayed compact finite-net, ContinuousMap family, Equicontinuity compatibility, RatUp fold, dyadic budget,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. A downstream consumer cannot replace the router by a host choice of centers, a pointwise family selector, quotient equality of maps, sequential compactness, or ambient Real completeness.*

Proof. Read the requested tolerance through the replay row C. The replay exposes K, F, E, L, Q, D before the handoff row T is visible.

The compact and family rows display only finite center and pointwise-modulus data. The equicontinuity row synchronizes that data for the inspected family, while the rational and dyadic rows compute the lower-bound budget inside the packet.

Transport and provenance through H, P, N keep the same finite route attached to the local name. Since the carrier has no coordinate for host choices, quotient equality of maps, sequential compactness, or ambient completeness, a UniformModulus consumer cannot escape the displayed router. □

Falsifiable prediction: A $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ consumer that reads a $\mathsf{UniformModulus}^{\uparrow}$ row without first displaying $K, F, E, L, Q, D, H, C, P, N$ refutes the router surface.

Independence witness: compactmetric, continuousmap, uniformmodulus, equicontinuity, rat: $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net rows, family pointwise-modulus rows, equicontinuity compatibility, finite lookup, RatUp fold, dyadic tolerance budget, UniformModulus handoff, transport, replay, provenance, and local naming while refusing host choice of centers and map equality.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFamilyModulusRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactFamilyModulusRouter}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap family pointwise-modulus rows, Equicontinuity compatibility, finite lookup rows, RatUp lower-bound folds, dyadic tolerance budgets, UniformModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactFamilyModulusRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6614.1, Definition 6614.2, Definition 6614.3, and Definition 6614.4 seed the finite family-router boundary for compact-uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactFamilyModulusRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, host function equality, selected center families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, family pointwise-modulus coverage, equicontinuity compatibility, lookup exactness, RatUp fold exactness, dyadic budget transport, UniformModulus handoff, and non-escape for BEDC.Derived.CompactFamilyModulusRouterUp.

Chapter 6615

A Concrete Naming Certificate for $\mathsf{DynkinPiLambda}^{\uparrow}$

Following Dynkin and Halmos, the $\mathsf{DynkinPiLambda}^{\uparrow}$ packet records the finite BEDC surface of the pi-lambda theorem as a handoff from displayed pi-system generators to a Dynkin-system closure route. It sits after the monotone-class carrier of chapter 4194, the measurable-event rows of chapter 92, the generated-set boundary of chapter 3384, and the displayed membership discipline of chapter 3668. The packet names generator admission, finite intersection closure, complement closure, disjoint-union closure, monotone-class handoff, measure-facing seal, transport, replay, provenance, and local naming; it does not import a host sigma-algebra completion, measure-extension theorem, quotient subset equality, or selected countable family.

Definition 6615.1 (Dynkin pi-lambda carrier)**.** *A* $\mathsf{DynkinPiLambda}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (G, I, C, U, M, S, H, R, P, N)$$

with the following displayed rows:

1. *G is the displayed pi-system generator family, read through* $\mathsf{SetLike}^{\uparrow}$ *rows and through the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set boundary whenever the source is Borel-facing.*
2. *I records finite intersection closure for generator rows that are jointly visible in G.*
3. *C records complement closure relative to the named universe row exposed by the same displayed family.*
4. *U records disjoint countable-union requests only through finite observed stages and their named continuation rows.*
5. *M is the* $\mathsf{MonotoneClass}^{\uparrow}$ *handoff row that turns the displayed Dynkin closure route into the monotone-class closure surface already named by chapter 4194.*
6. *S is the* $\mathsf{Measure}^{\uparrow}$ *seal row reached only after the generator, finite-intersection, complement, disjoint-union, and monotone-class rows have been exposed.*
7. *H stores componentwise* hsame *transport for G, I, C, U, M, S and the structural rows.*
8. *R stores displayed* Cont *replay of the finite route*

$$G, I \longmapsto C, U \longmapsto M \longmapsto S, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{DynkinPiLambda}^{\uparrow}$, $\mathsf{MonotoneClass}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DynkinPiLambda}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, I, C, U, M, S, H, R, P, N$. It has no coordinate for an ambient sigma-algebra, host pi-lambda theorem, selected countable family, measure extension, quotient equality of subsets, or external equality of events.

Theorem 6615.2 (Dynkin pi-lambda NameCert obligations)**.** *For the carrier of Definition 6615.1, the local* $\mathsf{NameCert}_{\mathsf{DynkinPiLambda}^{\uparrow}}$ *surface consists exactly of generator admission through G, finite-intersection closure through I, complement closure through C, disjoint-union continuation through U, monotone-class handoff through M, measure-facing seal through S, componentwise* hsame *transport through H, displayed* Cont *replay through R, package provenance through P, and local naming through N. Every accepted* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, *or* $\mathsf{SetLike}^{\uparrow}$ *consumer that reads a pi-lambda conclusion must expose those rows before it treats the generated family as measure-visible.*

Proof. Project the carrier $D = (G, I, C, U, M, S, H, R, P, N)$. The generator row G is the only entry point for displayed subset requests, and the finite-intersection row I is the only pi-system closure coordinate.

Read the Dynkin closure rows next. Complement requests pass through C, while disjoint-union requests pass through the finite observed stages and continuation rows stored in U. These rows stay attached to the same displayed generator family.

The handoff and seal rows then determine every downstream consumer. The monotone-class row M is the only route to the already named $\mathsf{MonotoneClass}^{\uparrow}$ surface, and the measure-facing row S is reached only after that route has been exposed.

Finally replay H, R, P, N. Transport, continuation replay, provenance, and local naming preserve exactly the same finite packet, so no host sigma-algebra completion, selected countable family, measure-extension theorem, quotient subset equality, or external event equality can enter the certificate. □

Theorem 6615.3 (Dynkin pi-lambda MonotoneClass handoff)**.** *Every accepted* $\mathsf{DynkinPiLambda}^{\uparrow}$ *packet exports the handoff route*

$$G, I, C, U \longmapsto M \longmapsto H, R, P, N$$

to $\mathsf{MonotoneClass}^{\uparrow}$. *A downstream consumer may cite monotone closure of the displayed generator family only after the pi-system intersection row, Dynkin complement row, disjoint-union continuation row, monotone-class handoff row, transport row, replay row, provenance row, and local name row have all been exposed.*

Proof. Start with the displayed generator row G. The pi-system condition is not a host family assertion; in the packet it is the finite-intersection row I over the visible subset rows.

Pass through the Dynkin-system rows. Complement closure is read through C, and disjoint-union closure is read through the finite observed stages stored in U. Both rows keep the same displayed generator family and do not choose an external countable family.

Apply the monotone-class handoff row M. It is the only coordinate that lets a consumer enter the $\mathsf{MonotoneClass}^{\uparrow}$ surface, so the consumer receives the existing monotone closure route rather than an ambient pi-lambda theorem.

The structural rows H, R, P, N transport, replay, source, and name the same handoff. Therefore the route exports no sigma-algebra completion, measure extension, quotient subset equality, or selected countable chain. □

Theorem 6615.4 (Dynkin pi-lambda measure seal nonescape)**.** *For every accepted* $\mathsf{DynkinPiLambda}^{\uparrow}$ *carrier*

$$D = (G, I, C, U, M, S, H, R, P, N),$$

the $\mathsf{Measure}^{\uparrow}$ *seal row* S *may be consumed only through the finite route*

$$G, I, C, U, M, H, R, P, N.$$

In particular, the seal cannot be used as a proof of host sigma-algebra completion, measure extension, quotient equality of subsets, selected countable families, external event equality, or an ambient pi-lambda theorem.

Proof. Project the measure-facing row S from the carrier. Its source rows are the displayed generator, finite-intersection, complement, disjoint-union, monotone-class handoff, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 6615.2. They force every measure-facing consumer to expose $G, I, C, U, M, H, R, P, N$ before S can be read.

The excluded principles have no carrier coordinate in that route. The seal is therefore a terminal measure-facing consumer of the finite Dynkin pi-lambda handoff, not an ambient sigma-algebra or measure-extension principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DynkinPiLambda}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DynkinPiLambda}^{\uparrow}$ packet is generated from displayed pi-system generators, finite-intersection closure, complement closure, disjoint-union continuation, a MonotoneClass handoff, a Measure seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DynkinPiLambda}^{\uparrow}}$, seed)

Scope closed: Definition 6615.1, Definition 6615.2, Definition 6615.3, and Definition 6615.4 seed the finite pi-lambda handoff surface over displayed set generators, Dynkin closure rows, monotone-class handoff, measure seal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DynkinPiLambda}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sigma-algebra completion, measure extension, selected countable families, quotient subset equality, external event equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pi-system closure, Dynkin closure, monotone-class handoff, measure-seal nonescape, transport, provenance, and replay for BEDC.Derived.DynkinPiLambdaUp.

Chapter 6616

A Concrete Naming Certificate for BishopLocatedUniformConvergence$^{\uparrow}$

Following Bishop and Bridges, located uniform convergence is the constructive interface where a family of displayed functions approaches a displayed limit by a finite modulus that can be read at every source observation and positive tolerance. The BishopLocatedUniformConvergence$^{\uparrow}$ packet sits over the uniform-limit handoff of chapter 463, the continuous-map rows of chapter 130, the metric source of chapter 57, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, and the finite schedule surface of chapter 126. It names finite located-modulus data for uniform convergence without importing host functions, choice, quotient equality of maps, selected pointwise limits, or ambient Real$^{\uparrow}$ completeness.

Definition 6616.1 (Bishop located uniform convergence carrier)**.** *A* BishopLocatedUniformConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, F, L, M, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the source* Met$^{\uparrow}$ *row, including the point classifier, distance observations, and finite source histories.*
2. *F is the* ContinuousMap$^{\uparrow}$ *family row, recording the displayed functions and their local modulus-bearing readback data.*
3. *L is the displayed limit-map row consumed through the* UniformLimit$^{\uparrow}$ *route.*
4. *M is the located uniform modulus row. Given a positive tolerance request, it records a finite stage bound that works uniformly over the displayed source observations.*
5. *W is the* StreamName$^{\uparrow}$ *schedule row for finite source windows, function-stage windows, limit-value windows, and tolerance requests.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational comparison windows generated by M and W.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row for the comparison values and the accepted limit read.*
8. *H stores componentwise* hsame *transport for the metric source, continuous-map family, limit row, located modulus, schedules, rational readbacks, Real seals, replay, provenance, and local name.*

9. *C stores displayed* Cont *replay from a uniform-convergence request through* X, F, L, M, W, R, E, H.
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedUniformConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, L, M, W, R, E, H, C, P, N$. It has no coordinate for a host function space, choice-selected convergence stage, quotient equality of maps, selected pointwise limit, open-cover compactness, or ambient real completeness.

Theorem 6616.2 (Bishop located uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$ *carrier*

$$U = (X, F, L, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedUniformConvergence}^{\uparrow}}$ *obligations are exactly carrier admission, source-metric compatibility, continuous-map family exposure, uniform-limit handoff, located finite-modulus stability, stream-window scheduling, regular-rational comparison readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6616.1. The accepted packet displays exactly the metric source, continuous-map family, limit row, located modulus, stream schedules, rational readbacks, Real seals, transport, replay, provenance, and local naming rows.

The uniform part is M. A tolerance request can become public only as a finite stage bound that is read together with the source row X, the function family F, the limit row L, and the finite schedule row W.

The located comparison part is R, E. The regular rational windows record the finite comparisons produced by the modulus and schedule rows, and the terminal Real seal is read only after those windows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for host functions, choice-selected stages, quotient map equality, selected pointwise limits, open-cover compactness, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6616.3 (Bishop located uniform convergence nonescape)**.** *Every* $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$ *consumer that reads a located uniform convergence claim must pass through*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows retained. The packet exports finite located-modulus uniform convergence only; it does not export host functions, choice-selected stages, quotient equality of maps, selected pointwise limits, open-cover compactness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier admission theorem. The only rows available to a consumer are $X, F, L, M, W, R, E, H, C, P, N$.

Apply the NameCert obligations theorem. It makes source-metric compatibility, continuous-map exposure, uniform-limit handoff, located finite-modulus stability, stream scheduling, regular-rational readback, and Real-seal nonescape the public route for the packet.

The consumer therefore receives a finite route through $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with transport and replay preserved by H, C, P, N. None of the excluded host, choice, quotient, pointwise-selection, compactness, or completeness principles is a row of the carrier, so none can be exported by the local name. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$ read exposes a source $\mathsf{Met}^{\uparrow}$ row, continuous-map family, displayed uniform-limit row, located finite-modulus row, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ comparison windows, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any consumer reads uniform convergence.

Independence witness: uniformlimit, continuousmap, metric, real, regseqrat, streamname: no listed sibling is bijective with $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$ because this packet jointly binds source metric data, continuous-map family rows, a displayed limit row, a located finite modulus, finite schedules, rational comparison windows, Real seals, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedUniformConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6616.1, Definition 6616.2, and Definition 6616.3 seed the finite located-modulus route for constructive uniform convergence over metric, continuous-map, uniform-limit, stream, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function spaces, choice-selected stages, quotient equality of maps, selected pointwise limits, open-cover compactness, ambient Real completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-metric compatibility, continuous-map exposure, uniform-limit handoff, located finite-modulus stability, stream scheduling, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedUniformConvergenceUp.

Constructive story: A $\mathsf{BishopLocatedUniformConvergence}^{\uparrow}$ packet is generated from a source Metric row, continuous-map family, displayed limit row, located finite modulus, stream schedules, regular-rational comparison windows, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Chapter 6617

A Concrete Naming Certificate for MetaCICTypedNormalFormScheduler$^{\uparrow}$

The MetaCICTypedNormalFormScheduler$^{\uparrow}$ packet realizes the normalization remaining question of section 9386.4 as finite scheduler-state data for closed typed terms with candidate gates and endpoint seals. It sits beside the frontier carrier of chapter 892, the bounded normal equality checker of chapter 752, the closed type checker seal of chapter 2182, the candidate normalization router of chapter 5210, and the candidate normalization witness of chapter 6577. The packet names scheduler-budget, candidate-gate, endpoint-seal, obstruction, transport, replay, provenance, and local naming rows; it is not an evidence-free strong-normalization theorem, unrestricted evaluator, quotient syntax equality, host reduction oracle, or closed consistency principle.

Definition 6617.1 (MetaCIC typed normal-form scheduler carrier)**.** *A* MetaCICTypedNormalFormScheduler$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, G, B, N, E, O, R, H, C, P, L)$$

with the following displayed rows:

1. *T is the sealed closed typed term row, read through the closed type checker surface.*
2. *G is the candidate-gate row, recording which candidate interface admits the term for scheduler inspection.*
3. *B is the scheduler-budget row, storing finite fuel, phase, residual, and replay-window accounting for the normalization attempt.*
4. *N is the typed normal-form row, recording the endpoint syntax together with the type seal under which it is read.*
5. *E is the endpoint-seal row consumed by bounded normal equality and normal-endpoint budget consumers.*
6. *O is the obstruction row, retaining missing candidate gates, budget exhaustion, blocked residuals, failed endpoint seals, and failed typed replay as visible scheduler boundaries.*
7. *R is the router handoff row, tying the scheduler to the candidate normalization router and candidate normalization witness surfaces.*
8. *H records componentwise* hsame *transport for the closed typed term, candidate gate, scheduler budget, typed endpoint, endpoint seal, obstruction, router handoff, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from a normalization request through T, G, B, N, E, O, R, H.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$, $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}}$ *row.*

The classifier compares accepted scheduler packets only through $T, G, B, N, E, O, R, H, C, P, L$. It has no coordinate for global strong normalization, unrestricted subject reduction, quotient syntax equality, hidden evaluator state, host reduction oracles, or completed closed consistency.

Theorem 6617.2 (MetaCIC typed normal-form scheduler obligations)**.** *For every accepted* MetaCICTypedNormalFo *carrier*

$$S = (T, G, B, N, E, O, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}}$ *obligations are exactly carrier admission, closed typed-term seal exposure, candidate-gate visibility, scheduler-budget totality for accepted requests, typed normal endpoint sealing, obstruction exactness, candidate-router handoff, candidate-witness compatibility, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6617.1. The accepted packet displays exactly the closed typed term, candidate gate, scheduler budget, typed endpoint, endpoint seal, obstruction, router handoff, transport, replay, provenance, and local naming rows.

The gate and budget rows G, B are consumed before any endpoint is public. Thus a scheduler read cannot reach N or E without exposing the candidate interface that admits the term and the finite scheduler accounting that bounds the attempt.

The obstruction row O remains visible for missing gates, exhausted budget, blocked residuals, failed endpoint seals, and failed typed replay. The router row R then hands only the displayed finite endpoint or obstruction data to the candidate router and candidate witness surfaces.

Finally H, C, P, L transport, replay, source, and name the same route. Since no carrier coordinate supplies global strong normalization, unrestricted subject reduction, quotient syntax equality, hidden evaluator state, host reduction oracles, or closed consistency, the listed obligations exhaust the local NameCert surface. □

Theorem 6617.3 (MetaCIC typed normal-form scheduler nonescape)**.** *Every public normalization-frontier consumer of an accepted* $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$ *packet factors through*

$$\mathsf{ClosedTypeCheckerSeal}^{\uparrow} \longmapsto G, B \longmapsto N, E \longmapsto O, R \longmapsto H, C, P, L.$$

The consumer receives a sealed closed typed term, candidate-gate verdict, finite scheduler budget, typed normal endpoint or visible obstruction, router handoff, transport, replay, provenance, and local naming. It receives no hidden scheduler state, evaluator oracle, quotient syntax equality, evidence-free normalization theorem, unrestricted subject reduction, or completed closed consistency.

Proof. Start with the closed typed term row T. The only scheduler route from this row to endpoint data passes through the candidate gate G and finite budget row B.

Apply Definition 6617.2. It makes typed endpoint sealing, obstruction exactness, candidate-router handoff, and candidate-witness compatibility obligations of the same finite packet.

The structural rows H, C, P, L preserve the displayed route when the scheduler packet is consumed by the normalization frontier, bounded equality checker, router, or candidate witness. Because hidden scheduler state, evaluator oracles, quotient syntax equality, evidence-free normalization, unrestricted subject reduction, and closed consistency are not carrier coordinates, none can be exported by the local name. □

Falsifiable prediction: Any accepted $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$ read that exposes typed endpoint data without first displaying the sealed closed typed term, candidate gate, finite scheduler budget, obstruction row, router handoff, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: metacicnormalizationfrontier, boundednormalequalitychecker, closed-type-checker-seal, metacic-candidate-normalization-router, metacic-candidate-normalization-witness: no listed sibling is bijective with $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$ because this packet jointly carries sealed closed typed terms, candidate gates, scheduler budgets, typed endpoint seals, obstruction rows, router handoff, transport, replay, provenance, and local NameCert rows.

Lean TasteGate work for $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$ must include Nontrivial and FieldFaithful instances so the candidate gate, scheduler budget, endpoint seal, obstruction, router handoff, replay, provenance, and local naming rows remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6617.1, Definition 6617.2, and Definition 6617.3 seed the finite typed normal-form scheduler route through closed typed terms, candidate gates, scheduler budgets, typed endpoints, obstruction, router handoff, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, unrestricted subject reduction, quotient syntax equality, hidden evaluator state, host reduction oracles, completed closed consistency, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed type seal exposure, candidate-gate visibility, scheduler-budget totality, typed endpoint sealing, obstruction exactness, router handoff, candidate-witness compatibility, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedNormalFormSched

Constructive story: A $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$ packet is generated from a sealed closed typed term, candidate-gate row, finite scheduler budget, typed normal endpoint, endpoint seal, obstruction row, router handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Chapter 6618

A Concrete Naming Certificate for $\mathsf{SearchableCompact}^{\uparrow}$

Following Escardo searchable compactness and Bishop constructive compactness, the $\mathsf{SearchableCompact}^{\uparrow}$ packet records a finite-search compactness interface as BEDC NameCert data. It sits beside the finite bar and modulus routes of chapter 1525, the Boolean-prefix source of chapter 1475, the finite-net compact metric surface of chapter 129, the modulus handoff of chapter 369, and the finite observation schedules of chapter 126. The packet exports searchable finite selectors for displayed predicates and modulus consumers; it does not export open-cover compactness, choice-selected witnesses, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6618.1 (SearchableCompact carrier)**.** *A* $\mathsf{SearchableCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, B, Q, F, M, W, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through either* $\mathsf{CompactMetric}^{\uparrow}$ *finite nets or* $\mathsf{CantorSpace}^{\uparrow}$ *prefix windows;*
2. *B is the finite branch or net boundary row supplied by* $\mathsf{FanFunctional}^{\uparrow}$ *when the source is a fan-like searchable domain;*
3. *Q is the inspected decidable predicate row, opened only on the finite windows named by the carrier;*
4. *F is the finite-selector row, recording the retained candidate list and the first accepted candidate when the inspected predicate is witnessed;*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row used when a downstream continuous predicate asks for a bounded search depth;*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for predicate, prefix, and finite-net reads;*
7. *R is the replay ledger from compact source and predicate windows to selector admission and modulus handoff;*
8. *H records componentwise* hsame *transport for K, B, Q, F, M, W, R;*
9. *C records displayed* Cont *replay of the same finite-search route;*

10. *P is the* Pkg *provenance over* $\mathsf{SearchableCompact}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SearchableCompact}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed rows. It has no coordinate for an arbitrary open cover, a choice-selected witness outside the retained finite window, a host compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6618.2 (SearchableCompact NameCert obligations)**.** *For an accepted* $\mathsf{SearchableCompact}^{\uparrow}$ *carrier $S = (K, B, Q, F, M, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SearchableCompact}^{\uparrow}}$ *surface consists exactly of compact source admission through K, finite boundary admission through B, decidable predicate exposure through Q, finite selector admission through F, modulus handoff through M, finite observation scheduling through W, replay through R, C, componentwise transport through H, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6618.1. Its source-facing rows are K and B: they expose compact metric finite nets, Cantor prefix windows, or fan-functional finite bar data before any selector is read.

The predicate and selector rows are read next. The row Q opens only the displayed finite predicate windows, and F records the finite retained candidates and accepted candidate boundary when such a boundary is displayed.

Finally M, W, R, H, C, P, N route the selector to downstream modulus and observation consumers. Because the carrier has no coordinate for an arbitrary open cover, selected global witness, host compactness theorem, or ambient real completeness, those objects cannot be exported by the local NameCert. □

Theorem 6618.3 (SearchableCompact finite-selection soundness)**.** *Every finite-selection read exported by an accepted* $\mathsf{SearchableCompact}^{\uparrow}$ *carrier factors through*

$$K, B, Q, F, M, W, R, H, C, P, N.$$

The consumer first reads the compact source and finite boundary rows, then the displayed decidable predicate windows, then the retained selector row, and only then the modulus and replay rows. Thus the exported search result is a constructive finite selector over the displayed packet, not an open-cover compactness proof, a choice-selected witness outside the packet, or a $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Proof. Begin with the compact source rows. The row K supplies the finite compact metric or Cantor prefix source, and B supplies the fan-functional or finite boundary data when the search route requires a bounded branch depth.

The predicate row Q is inspected only through W. The selector row F is therefore tied to the finite observation schedule and cannot name a candidate outside the displayed predicate windows.

Apply the replay rows R and C, then transport and provenance through H, P, N. The modulus consumer M can read only the same finite route. Hence the finite selector is sound for the packet that was displayed, while open-cover compactness, choice-selected witnesses, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the carrier. □

Falsifiable prediction: Every $\mathsf{SearchableCompact}^{\uparrow}$ finite-selector consumer must read the compact source row, finite boundary row, decidable predicate windows, retained selector row, modulus

handoff, observation schedule, replay, transport, provenance, and local naming before consuming the selected candidate.

Independence witness: fanfunctional, cantorspace, compactmetric, uniformmodulus, streamname: $\mathsf{SearchableCompact}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-source admission, finite boundary exposure, decidable predicate windows, retained selector rows, modulus handoff, observation schedules, replay, transport, provenance, and local naming in one finite-search packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SearchableCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{SearchableCompact}^{\uparrow}$ packet is generated from compact source rows, finite boundary rows, decidable predicate windows, finite selector rows, modulus handoff, StreamName schedules, replay ledgers, componentwise transport, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SearchableCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6618.1, Definition 6618.2, and Definition 6618.3 seed the finite-search compactness interface over displayed FanFunctionalUp, CantorSpaceUp, CompactMetricUp, UniformModulusUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SearchableCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, choice-selected witnesses, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires classifier stability, finite-selector replay, modulus-handoff nonescape, and downstream compact-uniform-continuity consumption for the same displayed packet.

Chapter 6619

A Concrete Naming Certificate for GalerkinErrorEstimate$^{\uparrow}$

Following Cea and standard Galerkin finite-element error analysis, the GalerkinErrorEstimate$^{\uparrow}$ packet records the constructive best-approximation surface exposed after a coercive variational problem has been projected onto a displayed finite trial space. It sits over the residual certificate of chapter 5169, the variational solution row of chapter 3419, the Hilbert comparison surface of chapter 91, the bounded-operator action row of chapter 1644, the norm ledger of chapter 89, and the scalar seal of chapter 38. The packet gives downstream finite-element and PDE consumers a finite error-estimate route without importing quotient function spaces, selected minimizers, host projection theorems, or choice.

Definition 6619.1 (Galerkin error estimate carrier)**.** *A* GalerkinErrorEstimate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, a, l, u, u_h, e, c, C, H, R, T, P, N)$$

with the following displayed rows:

1. *V is the finite trial-space row, naming the accepted trial basis, finite coefficient windows, and admissible comparison tests.*
2. *a is the coercive bilinear-form row, read through the bounded operator and Hilbert surfaces with displayed continuity and coercivity evidence.*
3. *l is the load row, naming the bounded functional data consumed by the variational problem.*
4. *u is the exact variational row supplied through the* LaxMilgram$^{\uparrow}$ *handoff and exposed only by its displayed equation against admitted tests.*
5. *u_h is the Galerkin row supplied through the* GalerkinProjection$^{\uparrow}$ *residual-orthogonality surface on the displayed trial space.*
6. *e is the error row recording the comparison between u and u_h in the displayed norm ledger.*
7. *c and C are the coercivity and continuity constants as finite scalar rows, with no selected optimizer or ambient real-number oracle.*
8. *H is the best-approximation comparison row. It records that each displayed trial competitor is compared through the same norm and bilinear-form rows before the scalar estimate is read.*
9. *R is the* Real$^{\uparrow}$ *bound row, reached only after the residual orthogonality, variational equation, coercivity, continuity, and comparison rows have been displayed.*

10. *T records componentwise* hsame *transport for trial-space, bilinear-form, load, exact-solution, Galerkin, error, constant, comparison, scalar-bound, replay, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{GalerkinErrorEstimate}^{\uparrow}$, $\mathsf{GalerkinProjection}^{\uparrow}$, $\mathsf{LaxMilgram}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GalerkinErrorEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, a, l, u, u_h, e, c, C, H, R, T, P, N$. It has no coordinate for a quotient function space, selected minimizer, host Hilbert projection theorem, ambient best-approximation theorem, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 6619.2 (Galerkin error estimate NameCert obligations)**.** *For every accepted* $\mathsf{GalerkinErrorEstimate}^{\uparrow}$ *carrier*

$$G = (V, a, l, u, u_h, e, c, C, H, R, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GalerkinErrorEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier locality over the displayed rows, residual orthogonality through u_h, coercivity and continuity through c, C, best-approximation comparison through H, scalar-bound readback through R, componentwise transport through T, replay through the displayed finite route, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6619.1. The trial space, bilinear form, load, exact variational row, Galerkin row, error row, constants, comparison row, scalar bound, transport, provenance, and local name are the visible inventory.

The residual-orthogonality obligation is read from the Galerkin row u_h and the finite trial row V. The exact variational obligation is read from u through the Lax-Milgram handoff, so the two analytic inputs remain separate before the error row is formed.

Coercivity and continuity are then read from c, C as displayed scalar rows. The comparison row H uses those constants to compare the error with each displayed trial competitor, and R records the resulting $\mathsf{Real}^{\uparrow}$ bound only after that route is visible.

Finally T, P, N transport, source, and name the same finite surface. Since no carrier coordinate stores a quotient space, selected minimizer, host projection theorem, choice schedule, or ambient real-completeness datum, the listed rows exhaust the NameCert obligations. □

Theorem 6619.3 (Galerkin best-approximation route)**.** *Every public* $\mathsf{GalerkinErrorEstimate}^{\uparrow}$ *error-estimate read factors through the* $\mathsf{GalerkinProjection}^{\uparrow}$ *residual row and the* $\mathsf{LaxMilgram}^{\uparrow}$ *variational row before it reads the* $\mathsf{Real}^{\uparrow}$ *scalar bound. In carrier terms, the public route is*

$$V, a, l, u, u_h, e, c, C, H \longmapsto R \longmapsto T, P, N.$$

The route exposes residual orthogonality, the exact variational equation, coercivity, continuity, and finite best-approximation comparison before any scalar estimate is consumed, with no quotient function space, selected minimizer, host Hilbert projection theorem, or choice principle.

Proof. Start with a public error-estimate consumer. The consumer cannot inspect the scalar bound row R directly, because the carrier places the trial-space, bilinear-form, load, exact variational, Galerkin, error, constant, and comparison rows before it.

Use the Galerkin projection sibling for the discrete side. The row u_h is available only together with the finite trial space V and the residual-orthogonality surface supplied by chapter 5169.

Use the Lax-Milgram sibling for the exact side. The row u is read through the variational equation and bounded functional route of chapter 3419, while a, l, c, C keep the bilinear, load, coercivity, and continuity data visible.

The comparison row H then forms the best-approximation route against displayed trial competitors, and the scalar row R seals that route as a $\mathsf{Real}^{\uparrow}$ bound. Transport, provenance, and naming through T, P, N preserve the same finite path, so the refused quotient, minimizer, host projection, and choice objects are not exported. □

Falsifiable prediction: Every accepted $\mathsf{GalerkinErrorEstimate}^{\uparrow}$ read displays the finite trial space, bilinear-form row, load row, LaxMilgram exact variational row, GalerkinProjection residual row, error row, coercivity and continuity constants, best-approximation comparison, RealUp scalar bound, componentwise transport, provenance, and local NameCert row before an error-estimate consumer reads the bound.

Independence witness: galerkin-projection, lax-milgram, hilbert, boundedlinearoperator, norm, real: $\mathsf{GalerkinErrorEstimate}^{\uparrow}$ is not bijective to any listed sibling because it binds residual orthogonality, variational exactness, coercivity, continuity, best-approximation comparison, and scalar error bound into one finite certificate surface.

Lean4 (statement) `BEDC.Derived.GalerkinErrorEstimateUp`

Closure status for $\mathsf{NameCert}_{\mathsf{GalerkinErrorEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{GalerkinErrorEstimate}^{\uparrow}$ packet is generated from finite trial-space rows, bilinear-form rows, load rows, exact variational rows, Galerkin residual rows, error rows, coercivity and continuity constants, best-approximation comparison rows, RealUp scalar bounds, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GalerkinErrorEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6619.1, Definition 6619.2, and Definition 6619.3 seed the constructive Galerkin best-approximation error route over projection, variational, Hilbert, bounded-operator, norm, and real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GalerkinErrorEstimate}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.GalerkinErrorEstimateUp`

Bridge status: none

Not claimed: This chapter does not close quotient function spaces, selected minimizers, host Hilbert projection theorems, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual orthogonality, coercivity and continuity rows, best-approximation comparison, scalar-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.GalerkinErrorEstimateUp.

Chapter 6620

A Concrete Naming Certificate for $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$

Bishop and Bridges use located subsets and finite nets as separate constructive interfaces before compactness arguments are allowed to read them together. The $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$ packet records that joint L10 metric-frontier surface over the located-subset carrier of chapter 1541, the finite-net carrier of chapter 127, the ambient metric source of chapter 57, the rational-window readback of chapter 301, and the terminal distance seal of chapter 38. It supplies later compactness, projection, completion, and uniformity consumers with displayed finite-net plus located-distance data, while making no claim of open-cover compactness, nearest-point choice, quotient subset equality, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6620.1 (Totally bounded located set carrier)**.** *A* $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, A, B, Q, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row, including admitted points and metric-distance observations.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row for the displayed subset, including admission and located-distance comparisons from chapter 1541.*
3. *B is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, including for each accepted tolerance request a finite bundle of centers and coverage witnesses.*
4. *Q is the located-distance window ledger tying each admitted ambient point, subset read, and net-center request to a displayed rational comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows used by both the located-distance and finite-net routes.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed distance value attached to the located-set read.*
7. *T is the handoff row saying that finite-net coverage and located-distance readback are consumed in the same ambient metric route.*
8. *H records componentwise* hsame *transport for the ambient metric, located subset, finite nets, rational windows, real seal, handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from tolerance and point requests through $X, A, B, Q, R, E, T, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedLocatedSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected nearest point, quotient equality of subset presentations, an ambient completion, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedLocatedSetUp.TotallyBoundedLocatedSetTasteGate single carrier alignment`

Theorem 6620.2 (Totally bounded located set NameCert obligations)**.** *For an accepted* TotallyBoundedLocatedSet *carrier $K = (X, A, B, Q, R, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedLocatedSet}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient metric row, located subset row, finite-net row, located-distance windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, joint handoff, componentwise transport, continuation replay, package provenance, and local naming. Every compactness, projection, completion, or uniformity consumer must display $X, A, B, Q, R, E, T, H, C, P, N$ before it may read any finite-net or located-distance conclusion.*

Proof. Project the carrier of Definition 6620.1. Its visible rows are exactly the ambient metric source, located subset, finite nets, distance windows, rational readback, Real seal, joint handoff, and structural rows.

The located-set obligations are read from A, Q, R, E. These rows provide the finite distance comparison and real-valued seal while keeping every read downstream of the same ambient metric row X.

The total-boundedness obligations are read from B over the same source. The finite bundle and its coverage witnesses are therefore available only as displayed packet data, not as an open-cover compactness theorem or hidden choice of centers outside the carrier.

The rows T, H, C, P, N preserve, replay, source, and name the same joint route. Since no excluded compactness, nearest-point, quotient, completion, or real-completeness coordinate appears in the carrier, the stated rows exhaust the NameCert obligation surface. □

Theorem 6620.3 (Finite-net located-distance handoff)**.** *Every accepted* $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$ *packet exposes a finite-net plus located-distance consumer route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{LocatedSet}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, a consumer reads the ambient metric row X, located-subset row A, finite-net row B, located-distance windows Q, rational readback R, and real seal E before it may use the handoff row T. The route exports no open-cover compactness, nearest-point choice, quotient subset equality, ambient completion, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with $K = (X, A, B, Q, R, E, T, H, C, P, N)$. The metric row X fixes the admitted ambient point and distance observations used by both the located-distance and finite-net reads.

Read A and B next. The located-set row supplies the subset-facing distance interface, while the totally-bounded row supplies finite bundles and coverage witnesses at accepted tolerance requests over the same metric source.

The numerical readback passes through Q, R, E. The window ledger Q records the finite comparison, R exposes the rational readback, and E seals the displayed distance as $\mathsf{Real}^{\uparrow}$ data.

The handoff and structural rows T, H, C, P, N only package that same route for downstream consumers. Thus the packet can feed later compactness and completion arguments with finite-net and located-distance data, but it cannot export any of the excluded principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedLocatedSet}^{\uparrow}}$.

Constructive story: A $\mathsf{TotallyBoundedLocatedSet}^{\uparrow}$ packet is generated from a displayed Metric row, a LocatedSet row, a TotallyBounded finite-net row, located-distance rational windows, RegSeqRat readback, a RealUp distance seal, a joint handoff, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedLocatedSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6620.1, Definition 6620.2, and Definition 6620.3 seed the finite-net plus located-distance carrier over the L10 constructive metric frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedLocatedSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, nearest-point choice, quotient subset equality, ambient completion, full $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, located-distance exactness, finite-net coverage compatibility, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for future BEDC.Derived.TotallyBoundedLocatedSetUp.

Chapter 6621

A Concrete Naming Certificate for $\mathsf{MazurTheorem}^{\uparrow}$

Following Mazur and Bourbaki, the $\mathsf{MazurTheorem}^{\uparrow}$ packet records the finite convex weak-closure certificate by which a norm-closed convex claim is read from displayed weak-topology windows without importing a host dual quotient, a selected separating functional, countable choice, or ambient functional-analysis completeness. It sits over the Banach carrier of chapter 90, the weak-topology observation surface of chapter 3597, the locally convex seminorm rows of chapter 3386, the convex carrier of chapter 208, and the terminal scalar readback of chapter 38.

Definition 6621.1 (Mazur theorem carrier)**.** *A* $\mathsf{MazurTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, L, W, V, K, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the normed vector carrier, addition and scalar rows, norm metric, and visible completeness ledger used by norm-facing consumers.*
2. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *observation row, exposing only the finite seminorm and neighbourhood tests needed to read weak windows.*
3. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *finite-window row over B and L, naming the inspected weak neighbourhoods and their admissible observation coordinates.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row, carrying the displayed convex subcarrier and the affine-combination operations admitted by the packet.*
5. *K is the convex-combination ledger. It records the finite barycentric coefficients, the inspected points, and the convexity witnesses used to move from weak-window data back to the convex row.*
6. *Q is the norm-closure handoff row, stating which norm-radius request is being read after the finite convex-combination ledger has been displayed.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for coefficients, seminorm values, norm radii, and comparison bounds.*
8. *H records componentwise* hsame *transport for Banach, locally convex, weak-window, convex, convex-combination, norm-closure, scalar, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$B, L, W, V \longmapsto K \longmapsto Q \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{MazurTheorem}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MazurTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, W, V, K, Q, R, H, C, P, N$. It has no coordinate for the whole dual space, quotient weak convergence, a selected separating functional, countable choice, a host closed convex hull, or ambient Banach-space theorem export.

Theorem 6621.2 (Mazur theorem NameCert obligations)**.** *For every accepted* $\mathsf{MazurTheorem}^{\uparrow}$ *carrier $M = (B, L, W, V, K, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MazurTheorem}^{\uparrow}}$ *surface consists exactly of Banach-source admission, locally convex observation admission, weak-topology finite-window exposure, convex-carrier admission, convex-combination ledger exactness, norm-closure handoff visibility, Real scalar readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that reads a weak closure claim as norm-closure data must pass through those rows before the norm-facing claim is visible.*

Proof. Project the carrier of Definition 6621.1. The source rows B, L, W, V display the Banach carrier, locally convex observation surface, weak-window ledger, and convex carrier before any closure read is available.

The finite convex step is the ledger K. It exposes the barycentric coefficients, inspected points, and convexity witnesses, with scalar comparisons read only through the $\mathsf{Real}^{\uparrow}$ row R.

The norm-facing conclusion is read through Q after the weak-window and convex-combination rows have been displayed. Thus a consumer receives a finite norm-closure handoff from the convex ledger, not a quotient weak limit or selected functional.

Finally H, C, P, N transport, replay, source, and name exactly the same rows. Since the carrier has no coordinate for the whole dual, countable choice, selected separating functionals, or an ambient closed-convex-hull theorem, none of those objects can be exported by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MazurTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MazurTheorem}^{\uparrow}$ packet is generated from Banach source data, locally convex seminorm observations, weak-topology finite windows, convex-set rows, a finite convex-combination ledger, norm-closure handoff data, Real scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MazurTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 6621.1 and Definition 6621.2 seed the finite weak-to-norm convex closure certificate over Banach, WeakTopology, LocallyConvexSpace, ConvexSet, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MazurTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a full host Mazur theorem, dual-space quotient semantics, selected separating functionals, countable choice, host equality of closed convex hulls, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-window exposure, convex-combination ledger exactness, norm-closure handoff visibility, Real scalar readback, transport, replay, provenance, and non-escape for BEDC.Derived.MazurTheoremUp.

Chapter 6622

A Concrete Naming Certificate for $\mathsf{PolishCompletionCriterion}^{\uparrow}$

Following Alexandrov–Urysohn and Bishop–Bridges, the $\mathsf{PolishCompletionCriterion}^{\uparrow}$ packet records the traditional separable complete-metric criterion as finite naming data before any Real-facing completion read is sealed. It sits after the complete-metric witness surface of chapter 128, the metric-completion carrier of chapter 1506, the separability schedule of chapter 1809, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names the Polish completion criterion through displayed dense Cauchy threads; it is not a quotient completion, selected representative principle, countable-choice construction, host equality test, or ambient completeness theorem for all $\mathsf{Real}^{\uparrow}$ names.

Definition 6622.1 (Polish completion criterion carrier)**.** *A* $\mathsf{PolishCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric source row whose distance classifier supplies the separated boundary used by the criterion.*
2. *D is the countable dense schedule row, recording the finite dense points and requested approximation indices exposed to the consumer.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened along the dense schedule.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from those stream windows and their tolerance requests.*
5. *L is the displayed limit row supplied by the completion criterion for the Cauchy thread read through D, S, Q.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dense thread, readback, and displayed limit are visible.*
7. *H records componentwise* hsame *transport for the metric source, dense schedule, stream windows, regular readback, displayed limit, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through X, D, S, Q, L, E.*
9. *P is the* Pkg *provenance over* $\mathsf{PolishCompletionCriterion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCompletionCriterion}^{\uparrow}}$ *row naming the criterion packet and its admitted consumer reads.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient completion equality, selected representatives, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6622.2 (Polish completion criterion NameCert obligations)**.** *For every accepted* PolishCompletionCriterio *carrier $K = (X, D, S, Q, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PolishCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric source, dense schedule, stream windows, regular rational readback, displayed limit, Real seal, transport, replay, provenance, and local naming rows;*
2. *dense schedule stability, requiring every completion-facing observation to preserve the same displayed dense row D under componentwise* hsame *transport;*
3. *Cauchy-thread handoff, requiring the route from D through S and Q to be displayed before L or E is consumed;*
4. *separated-limit boundary, requiring limit comparison to use the source metric classifier of X together with the displayed limit row L;*
5. *Real-seal nonescape, requiring the Real-facing row E to remain downstream of D, S, Q, L and upstream only of structural rows;*
6. *ledger refusal for quotient completion equality, selected representatives, countable choice, host real equality, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside $X, D, S, Q, L, E, H, C, P, N$.

Proof. Project the carrier of Definition 6622.1. The accepted packet displays exactly the metric source, dense schedule, finite windows, regular readback, displayed limit, Real seal, transport, replay, provenance, and naming rows.

The dense schedule stability and Cauchy-thread handoff are read in that order. A completion-facing consumer first opens D, then the finite $\mathsf{StreamName}^{\uparrow}$ windows S, and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q before the displayed limit L is available.

The separated-limit boundary is supplied by the same metric source X and displayed limit L. The Real seal E is downstream of this boundary, so it cannot be used as an ambient completeness source.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no row for quotient completion equality, selected representatives, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the ledger refuses those reads. □

Theorem 6622.3 (Polish completion criterion Cauchy-thread handoff)**.** *Every completion-criterion read exported by an accepted* $\mathsf{PolishCompletionCriterion}^{\uparrow}$ *carrier $K = (X, D, S, Q, L, E, H, C, P, N)$ factors through the dense Cauchy thread*

$$D \longmapsto S \longmapsto Q \longmapsto L$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal E is consumed. The read exposes the metric source X, the dense schedule D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the displayed limit row L, componentwise transport H, displayed replay C, provenance P, and local naming N; it does not export quotient completion equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and project a completion-criterion consumer read. By Definition 6622.2, every such read must display carrier admission, dense schedule stability, and Cauchy-thread handoff before the Real seal is available.

The only dense-source route in the carrier is D, S, Q, L. The row D supplies the displayed dense schedule, S supplies finite windows along that schedule, Q supplies the regular rational readback, and L records the displayed limit reached by the thread.

The Real seal row E is therefore downstream of the displayed Cauchy thread. The structural rows H, C, P, N can transport, replay, source, and name this route, but they do not add quotient completion equality, selected representatives, choice, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCompletionCriterion}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishCompletionCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6622.1, Definition 6622.2, and Definition 6622.3 seed the finite Polish completion criterion route through dense schedules, stream windows, regular rational readback, displayed limits, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishCompletionCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense schedule stability, Cauchy-thread handoff, separated-limit boundary, Real-seal nonescape, and ledger refusal for BEDC.Derived.PolishCompletionCriterionUp.

Constructive story: A $\mathsf{PolishCompletionCriterion}^{\uparrow}$ packet is generated from a metric source, a countable dense schedule, StreamName finite windows, RegSeqRat readback, a displayed limit row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6623

A Concrete Naming Certificate for $\mathsf{ChebyshevCenter}^{\uparrow}$

Following classical Chebyshev-center and minimal-enclosing-ball theory, the $\mathsf{ChebyshevCenter}^{\uparrow}$ packet records only compact metric data, convex-set data, located-radius comparisons, and center-candidate rows. It sits over the metric carrier of chapter 57, the projection-facing metric row of chapter 2356, the convex carrier of chapter 208, the located-set surface of chapter 1541, and the scalar readback of chapter 38. The certificate does not import a host minimizer theorem, selected center, quotient compactness, ambient convex duality, countable choice, `Classical.choice`, or `Quot.sound`.

Definition 6623.1 (Chebyshev center carrier)**.** *A* $\mathsf{ChebyshevCenter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M, K, V, L, R, S, C, A, H, T, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, naming the admitted points, distance comparisons, triangle ledger, and closed-ball tests used by the packet.*
2. *K is the compact metric row, recording the finite cover and total-boundedness observations consumed by radius comparisons.*
3. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row, exposing the displayed convex subcarrier and the admitted affine-combination operations.*
4. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row, recording located membership and distance-to-set comparisons for the candidate set.*
5. *R is the located-radius row, listing the finite radius requests, comparison tolerances, and scalar bounds read through* $\mathsf{Real}^{\uparrow}$*.*
6. *S is the center-candidate row, naming the inspected candidates and the ball-containment observations certified for each candidate.*
7. *C is the compact-radius ledger. It records that every displayed finite cover and center-candidate comparison is evaluated through M, K, L, R, S before a radius bound is read.*
8. *A is the admissible-center row, exposing the local witness that a candidate is read as an accepted center only after the displayed radius ledger has been inspected.*
9. *H records componentwise* hsame *transport for metric, compact, convex, located, radius, center-candidate, radius-ledger, admissible-center, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay of the route*

$$M, K, V, L \longmapsto R, S \longmapsto C, A.$$

11. *P is the* Pkg *provenance over* $\mathsf{ChebyshevCenter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ChebyshevCenter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, V, L, R, S, C, A, H, T, P, N$. It has no coordinate for a host minimizer, selected center outside the displayed candidates, quotient compactness, ambient Hilbert projection theorem, separating functional, choice schedule, or global uniqueness claim.

Lean4 ✓ `BEDC.Derived.ChebyshevCenterUp.ChebyshevCenterTasteGate single carrier alignment`

Theorem 6623.2 (Chebyshev center NameCert obligations)**.** *For every accepted* $\mathsf{ChebyshevCenter}^{\uparrow}$ *carrier*

$$Z = (M, K, V, L, R, S, C, A, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChebyshevCenter}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source admission, compact finite-cover exposure, convex-carrier admission, located-set readback, located-radius comparison, center-candidate visibility, compact-radius ledger exactness, admissible-center exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6623.1. The rows $M, K, V, L, R, S, C, A, H, T, P, N$ are the visible inventory, so any local consumer first sees the metric, compact, convex, located, radius, candidate, ledger, admissible-center, transport, replay, provenance, and naming rows.

Metric and compact admission are read from M, K. The packet can inspect distance comparisons and finite-cover data, but it cannot replace them with a host compactness theorem or a selected minimizing sequence.

Convex and located data are read from V, L. These rows expose the set in which candidates are tested and the located comparisons used to read distances to the displayed set.

The radius and center obligations are then read from R, S, C, A. A center is accepted only through the displayed compact-radius ledger, so the certificate records a finite radius comparison surface rather than an ambient minimizer.

Finally H, T, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for quotient compactness, a selected center outside S, ambient convex duality, choice, `Classical.choice`, or `Quot.sound`, none of those objects can be exported by the local certificate. □

Theorem 6623.3 (Chebyshev compact-radius ledger)**.** *Every* $\mathsf{ChebyshevCenter}^{\uparrow}$ *radius read factors through the displayed compact metric, located-set, and center-candidate rows before it reaches a scalar bound:*

$$M, K, V, L \longmapsto R, S \longmapsto C, A \longmapsto H, T, P, N.$$

Thus a public consumer receives finite compact-cover comparisons, located-radius requests, and admissible-center evidence before any center or radius datum is visible. The route does not assert uniqueness of centers, existence of a host minimizer, selected projection, quotient compactness, or ambient convex duality.

Proof. Start with a radius-facing consumer of Z. The consumer cannot read a scalar radius directly, because the carrier places the metric row M, compact row K, convex row V, and located row L before any radius or candidate row.

The located-radius row R and center-candidate row S are then inspected together. Radius comparisons are meaningful only for the displayed candidate and set data, so the route remains local to the packet.

The compact-radius ledger C and admissible-center row A bind those comparisons to finite cover observations and ball-containment tests. This is the finite NameCert surface corresponding to the minimal-enclosing-ball question.

Transport, replay, provenance, and naming through H, T, P, N preserve the same route. Since no further coordinate exists, the packet exports neither a selected minimizer nor a host compact convex optimization theorem. □

Falsifiable prediction: Every accepted $\mathsf{ChebyshevCenter}^{\uparrow}$ read displays the metric source, compact metric row, convex set row, located set row, located radius row, center-candidate row, compact-radius ledger, admissible-center row, componentwise transport, replay, provenance, and local NameCert row before a center or radius consumer reads the bound.

Independence witness: metric, compact-metric, metric-projection, convexset, locatedset, real: $\mathsf{ChebyshevCenter}^{\uparrow}$ is not bijective to any listed sibling because it binds compact-radius comparisons, located-radius rows, center-candidate evidence, and admissible-center readback into one finite certificate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ChebyshevCenter}^{\uparrow}}$.

Constructive story: A $\mathsf{ChebyshevCenter}^{\uparrow}$ packet is generated from metric rows, compact metric rows, convex-set rows, located-set rows, located-radius comparisons, center-candidate rows, compact-radius ledgers, admissible-center rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChebyshevCenter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6623.1, Definition 6623.2, and Definition 6623.3 seed the compact-metric Chebyshev center certificate over metric, compact, convex, located, scalar, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChebyshevCenter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close uniqueness of centers, host minimizer existence, selected projections, quotient compactness, ambient convex duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover exposure, located-radius comparison, center-candidate visibility, compact-radius ledger exactness, admissible-center exposure, transport, replay, provenance, and non-escape for BEDC.Derived.ChebyshevCenterUp.

Chapter 6624

A Concrete Naming Certificate for $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$

Following Bishop and Bridges, located sequential completeness is read from regular sequence data with a displayed modulus, not from a quotient stream or a selected ambient limit. The $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$ packet records the standard constructive route by which a modulus-controlled sequence supplies finite windows, regular-rational readback, and a Real-facing seal. It sits over the regular sequence surface of chapter 301, the stream naming discipline of chapter 126, the Real certificate of chapter 38, and the completion criterion of chapter 3404.

Definition 6624.1 (Located sequential completeness carrier)**.** *A* $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, M, W, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the sequence source row. It names the admitted sequence observations and exposes no host stream equality.*
2. *M is the located Cauchy modulus row, recording the finite tolerance requests and the index bounds that answer them.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing the inspected sequence windows and their observation schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational regularity comparisons obtained from M and W.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, opened only after the regular-rational readback has been displayed.*
6. *U is the uniqueness ledger. It records that two accepted reads with the same modulus, finite windows, regular readback, and Real seal compare through the same displayed rows.*
7. *H stores componentwise* hsame *transport for the source, modulus, StreamName windows, RegSeqRat readback, Real seal, uniqueness ledger, replay, provenance, and local naming rows.*
8. *C stores* Cont *replay from source observations to modulus requests, from modulus requests to finite windows, from windows to regular-rational readback, and from readback to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedSequentialCompleteness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, M, W, R, E, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host equality of Real names, ambient completeness, or an unbounded choice schedule.

Theorem 6624.2 (Located sequential completeness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$ *carrier $S = (A, M, W, R, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequentialComplete}}$ *surface consists exactly of sequence-source admission, located Cauchy modulus visibility, StreamName finite-window coverage, RegSeqRat readback, Real seal handoff, uniqueness-ledger stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion or Real-facing consumer must read those rows before it consumes the sequential-completeness output.*

Proof. Project the carrier of Definition 6624.1. The source, modulus, finite-window, regular-rational, Real-seal, uniqueness, transport, replay, provenance, and naming rows are the visible inventory.

The modulus and window obligations are read together. The located Cauchy modulus supplies finite bounds for positive tolerance requests, while the StreamName row records exactly which finite observations are inspected under those bounds.

The RegSeqRat and Real-facing obligations are downstream. The row R packages the regular rational comparisons produced by the modulus and windows, and E seals the compatible readback as a $\mathsf{Real}^{\uparrow}$ name only after R has appeared.

Finally U, H, C, P, N stabilize, transport, replay, source, and name the same route. Since the carrier has no coordinate for quotient stream equality, selected limits, host equality, ambient completeness, or choice schedules, the listed obligations exhaust the NameCert surface. □

Theorem 6624.3 (Located sequential completeness RegSeqRat handoff)**.** *Every accepted* $\mathsf{LocatedSequentialComple}$ *packet hands a modulus-controlled sequence to Real-facing consumers through the route*

$$A \longmapsto M \longmapsto W \longmapsto R \longmapsto E.$$

Thus the consumer reads the sequence source, located Cauchy modulus, StreamName finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before any completion criterion or Real seal is public.*

Proof. Start from the source row A. It provides only the admitted sequence observations, so no consumer can begin with a host stream or quotient class.

Use the modulus row M and finite-window row W. A positive tolerance request is answered by a displayed index bound and by the finite StreamName observations inspected under that bound.

Read the regular-rational row R next. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff packages the rational comparison windows supplied by M and W, and the Real seal E is downstream of that readback.

The structural rows H, C, P, N preserve this route for completion and Real-facing consumers. Therefore the handoff exports the displayed finite route rather than a selected limit, quotient stream equality, host equality, or ambient completeness principle. □

Theorem 6624.4 (Located sequential completeness uniqueness ledger)**.** *If two accepted* $\mathsf{LocatedSequentialComplete}$ *packets display the same sequence source, located Cauchy modulus, StreamName finite windows, RegSeqRat readback, and Real seal rows, then their public completion read is compared only through the uniqueness ledger U and the structural rows H, C, P, N. The comparison does not use host equality of streams or selected limit witnesses.*

Proof. Read the equality premise row by row. Agreement of A, M, W, R, E fixes the admitted observations, the tolerance-to-index bounds, the finite observation windows, the regular-rational comparisons, and the Real-facing seal.

The uniqueness ledger U records that no further coordinate is available for public comparison. Componentwise hsame transport in H and displayed replay in C keep the same rows attached to both packets.

Provenance and local naming through P, N then identify the comparison as a NameCert read over the displayed route. Since neither packet contains a host stream equality row, selected limit row, or ambient completeness row, the uniqueness comparison cannot appeal to those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequentialCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSequentialCompleteness}^{\uparrow}$ packet is generated from sequence-source observations, a located Cauchy modulus, StreamName finite windows, RegSeqRat readback, a Real seal, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequentialCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6624.1, Definition 6624.2, Definition 6624.3, and Definition 6624.4 seed the finite located sequential-completeness route from modulus-controlled sequence reads to RegSeqRat and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequentialCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host equality of Real names, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located modulus visibility, StreamName finite-window coverage, RegSeqRat readback, Real seal handoff, uniqueness stability, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSequentialCompletenessUp.

Chapter 6625

A Concrete Naming Certificate for $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$

The $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$ packet realizes the model-quality branch of section 9462.15 as finite NameCert data. It turns a narrative quality comparison into a displayed ledger over behavior coverage, public weights, cost, debt, stability, transfer, safety, semantic, and verification rows. The packet is AI-origin because its carrier is a BEDC-native accounting object for classifier quality rather than a reconstruction of a classical analytic theorem.

The ledger consumes the physical model-audit row of chapter 1129 and the certified model-quality carrier of chapter 6117 before it scores benefit, cost, debt, safety, verification, and semantic rows. This dependency places the quality ledger after model audit and model-quality certification data, without changing the chapter's closure grade.

Definition 6625.1 (Certified model quality ledger carrier)**.** *A* $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, B, W, C, D, T, A, V, G, H, R, P, N)$$

with the following displayed rows:

1. *S is the declared model-quality scope row, naming the evaluation domain, model state, admitted intervention family, critical behavior family, and public protocol.*
2. *B is the behavior-coverage row, separating certified critical behaviors from uncovered or partially certified behaviors.*
3. *W is the public weight-profile row used to compare benefit, cost, and debt.*
4. *C is the cost row, recording model, data, training, certificate, ledger, intervention, and verification costs.*
5. *D is the debt row, recording black-box, ledger, stability, generalization, safety, shortcut, semantic, and verification debts.*
6. *T is the stability and transfer row for the certified behavior classifiers.*
7. *A is the safety and shortcut-boundary row, recording failure cases and not-claimed scope.*
8. *V is the verification row, separating checked, target, and unverified certificate obligations.*
9. *G is the quality-score gate row. It reads benefit, cost, and debt only through B, W, C, D, T, A, V.*

10. *H records componentwise* hsame *transport for the scope, coverage, weights, costs, debts, stability, safety, verification, score, replay, provenance, and local naming rows.*
11. *R records displayed* Cont *replay from scope and behavior coverage to score comparison, gate acceptance, and refusal boundaries.*
12. *P is the* Pkg *provenance over* $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedModelQualityLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through $S, B, W, C, D, T, A, V, G, H, R, P, N$. It has no coordinate for private benchmark preference, hidden weight selection, unledgered classifier gain, performance-only quality, or verification laundering.

Theorem 6625.2 (Certified model quality ledger NameCert obligations)**.** *For every accepted* $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$ *carrier $Q = (S, B, W, C, D, T, A, V, G, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}}$ *surface consists exactly of scope admission, behavior-coverage visibility, public weight-profile display, cost accounting, debt accounting, stability and transfer accounting, safety and shortcut-boundary display, verification status display, quality-score gate replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6625.1. The scope, coverage, weight, cost, debt, stability, safety, verification, score, transport, replay, provenance, and naming rows are the entire displayed inventory.

The accounting rows are read before any score comparison. Behavior coverage supplies certified and uncovered critical behavior families, the public weight profile states how benefit, cost, and debt are compared, and the cost and debt rows record the negative terms.

The stability, safety, and verification rows restrict what may count as certified benefit. A classifier gain without those rows remains a debt-bearing or target-only entry rather than a closed quality gain.

Finally G, H, R, P, N replay, transport, source, and name the same finite ledger. Since the carrier has no coordinate for hidden weights, private benchmark preference, performance-only quality, unledgered classifier gain, or verification laundering, the listed rows exhaust the local certificate obligations. □

Theorem 6625.3 (Certified model quality score nonescape)**.** *Every accepted quality-score comparison in a* $\mathsf{CertifiedModelQualityLedger}^{\uparrow}$ *packet must display the behavior-coverage row B, public weight-profile row W, cost row C, debt row D, stability and transfer row T, safety row A, verification row V, and score gate row G before the comparison is read. Two accepted ledgers with identical behavior-coverage rows but different public weight profiles cannot be compared without displaying the weight-profile row.*

Proof. Start from the score gate row G. By Definition 6625.1, G reads benefit, cost, and debt only through B, W, C, D, T, A, V.

If two ledgers agree on behavior coverage but differ in public weights, the row B alone does not determine the score comparison. The row W specifies how certified benefit, cost, and debt are weighed, so it must be displayed before the comparison is public.

The remaining rows keep the score inside the ledger. Costs and debts are read from C, D, stability and safety qualifications from T, A, verification status from V, and transport and replay from H, R, P, N.

Thus score comparison cannot escape to a private benchmark, hidden weight protocol, performance-only metric, unledgered classifier gain, or verification laundering artifact. □

Theorem 6625.4 (Certified model quality TasteGate follow-up boundary)**.** *The finite* CertifiedModelQualityLedger$^{\uparrow}$ *carrier supplies the displayed ledger rows needed by later TasteGate checks, while the Nontrivial and FieldFaithful instances are supplied by the Lean follow-up in* `lean4/BEDC/Derived/CertifiedModelQualityLedge` *The paper packet records the classifier, ledger, and score-gate obligations; it does not claim those Lean instances inside this chapter.*

Proof. Use the carrier and NameCert obligation theorems. They display the scope, coverage, weight, cost, debt, stability, safety, verification, score, transport, replay, provenance, and naming rows required by a finite ledger.

The TasteGate follow-up consumes those rows as an implementation-facing obligation. This chapter records the public finite surface and the boundary of the future instance check without adding a Lean marker or claiming an existing declaration.

Since no Nontrivial or FieldFaithful instance row is part of Q, the chapter remains a paper-side certificate landing. The Lean file named in the statement is the place where those instances are to be supplied. □

Falsifiable prediction: If two accepted ledgers have identical behavior-coverage rows but different public weight profiles, their quality-score comparison cannot be read without displaying the weight-profile row.

Independence witness: modelselectionboundary, open_fit_packet, reality_constrained_model_audit, reality_constrained_signature_fit, physical_truth_certificate: CertifiedModelQualityLedger$^{\uparrow}$ is not bijective to any listed sibling because it binds behavior coverage, public weights, cost, debt, stability, safety, verification, score replay, transport, provenance, and local NameCert rows into one finite quality-accounting surface.

Closure status for NameCert$_{\text{CertifiedModelQualityLedger}^{\uparrow}}$**.**

Constructive story: A CertifiedModelQualityLedger$^{\uparrow}$ packet is generated from model-quality scope rows, behavior-coverage rows, public weight profiles, cost rows, debt rows, stability and transfer rows, safety and shortcut-boundary rows, verification rows, score-gate replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CertifiedModelQualityLedger}^{\uparrow}}$, seed)

Scope closed: Definition 6625.1, Definition 6625.2, Definition 6625.3, and Definition 6625.4 seed the finite certified model-quality ledger and score-gate surface.

Formal verification: FormalStatus(NameCert$_{\text{CertifiedModelQualityLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close hidden weight selection, private benchmark preference, performance-only quality, unledgered classifier gain, Lean verification, TasteGate instances, or bridge checking.

Upgrade path: Obligation closure requires checked scope admission, behavior-coverage visibility, weight-profile display, cost and debt accounting, stability and transfer accounting, safety and verification rows, score-gate nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CertifiedModelQualityLedgerUp.

Chapter 6626

A Concrete Naming Certificate for $\mathsf{ValuationDomain}^{\uparrow}$

Following Bourbaki and Zariski–Samuel, the $\mathsf{ValuationDomain}^{\uparrow}$ packet records the traditional valuation-domain surface as finite naming data over a ring whose fraction-field comparisons are routed through displayed ideal comparability and local-field boundary rows. It sits after the ring carrier of chapter 43, the field boundary of chapter 45, the ideal lattice of chapter 81, and the local-field handoff of chapter 173. The packet names the valuation-domain read; it does not export a selected valuation map, total host order on a fraction field, Krull dimension theorem, Noetherianity, principal-ideal generation, or ambient algebraic closure.

Definition 6626.1 (Valuation domain carrier)**.** *A* $\mathsf{ValuationDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (R, F, I, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Ring}^{\uparrow}$ *row, including the multiplication, addition, zero, one, and ring classifier rows used by ideal-facing consumers.*
2. *F is the* $\mathsf{Field}^{\uparrow}$ *fraction-field boundary row naming which field read receives the embedded ring data.*
3. *I is the* $\mathsf{Ideal}^{\uparrow}$ *comparability row. It records the displayed principal-fraction ideal tests by which any two nonzero fraction reads are compared through containment before the valuation-domain claim is visible.*
4. *L is the* $\mathsf{LocalField}^{\uparrow}$ *handoff row, recording the local valuation-facing boundary available to downstream local-field and completion consumers.*
5. *Q is the fraction-field nonescape row. It states that a consumer may read divisibility or inverse comparison only through R, F, I, L and not through an ambient selected valuation function.*
6. *H records componentwise* hsame *transport for the ring, fraction-field, ideal-comparability, local-field, fraction-boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$R, F, I \longmapsto L \longmapsto Q.$$

8. *P is the* Pkg *provenance over* $\mathsf{ValuationDomain}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ValuationDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, F, I, L, Q, H, C, P, N$. It has no coordinate for a chosen value group, a selected valuation map, quotient fraction equality outside the displayed field boundary, Noetherianity, principal generation, or ambient algebraic closure.

Theorem 6626.2 (Valuation domain NameCert obligations)**.** *For every accepted* $\mathsf{ValuationDomain}^{\uparrow}$ *carrier $V = (R, F, I, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ValuationDomain}^{\uparrow}}$ *surface consists exactly of ring-source admission, fraction-field boundary exposure, ideal-chain comparability, local-field handoff visibility, fraction-field nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that reads a valuation-domain claim must pass through those rows before it may compare fraction-field representatives or ideal containment data.*

Proof. Project the carrier of Definition 6626.1. The source rows R, F, I, L, Q display the ring carrier, fraction-field boundary, ideal-comparability ledger, local-field handoff, and fraction-field nonescape row before any valuation-domain read is available.

The structural rows H, C, P, N transport, replay, source, and name exactly the same finite route. Since the carrier has no coordinate for a chosen value group, selected valuation map, Noetherianity, principal generation, or ambient algebraic closure, none of those objects can be exported by the local certificate. □

Theorem 6626.3 (Valuation domain ideal-chain route)**.** *Every ideal-facing read exported by an accepted* $\mathsf{ValuationDomain}^{\uparrow}$ *carrier $V = (R, F, I, L, Q, H, C, P, N)$ factors through the displayed comparability route*

$$\mathsf{Ring}^{\uparrow} \longmapsto \mathsf{Field}^{\uparrow} \longmapsto \mathsf{Ideal}^{\uparrow}$$

before a fraction-field comparison is visible. The route records containment comparison for the principal-fraction ideal rows named by I and refuses any detached host-order, value-group, or selected-valuation read.

Proof. Start with the ring row R and the fraction-field boundary F. These rows fix the source multiplication and the field read in which fraction representatives are inspected.

The ideal-chain row I is the only comparability surface in the carrier. Thus a consumer compares fraction reads by opening the displayed $\mathsf{Ideal}^{\uparrow}$ containment data after R and F, not by appealing to a host value group or an unnamed total order.

The nonescape row Q and the structural rows H, C, P, N preserve that route under transport and replay. Therefore the ideal-facing read remains a finite NameCert route through Ring, Field, Ideal, local handoff, and local naming rows. □

Theorem 6626.4 (Valuation domain local-field handoff)**.** *Every* $\mathsf{LocalField}^{\uparrow}$ *consumer of an accepted* $\mathsf{ValuationDomain}^{\uparrow}$ *packet reads the local valuation-facing boundary only after the source ring, fraction-field boundary, and ideal-chain comparability rows have been displayed. The handoff route is*

$$\mathsf{ValuationDomain}^{\uparrow} \longmapsto \mathsf{Ring}^{\uparrow} \longmapsto \mathsf{Field}^{\uparrow} \longmapsto \mathsf{Ideal}^{\uparrow} \longmapsto \mathsf{LocalField}^{\uparrow}.$$

It exports no complete local field, residue-field construction, chosen uniformizer, discrete valuation, or completion theorem beyond the displayed L row.

Proof. Read L as a downstream row of the accepted carrier. The carrier places L after the ring row R, the fraction-field boundary F, and the ideal-comparability row I.

A local-field consumer therefore receives only the displayed valuation-facing handoff. The packet does not contain a residue-field construction, selected uniformizer, discrete valuation, completion theorem, or host local-field equality, so those data cannot be exported by the $\mathsf{ValuationDomain}^{\uparrow}$ NameCert. □

Theorem 6626.5 (Valuation domain fraction-field boundary)**.** *Every fraction-field read of an accepted* $\mathsf{ValuationDomain}^{\uparrow}$ *carrier remains bounded by the displayed route*

$$R, F, I, L, Q.$$

In particular, inverse comparison and divisibility information are visible only through the source $\mathsf{Ring}^{\uparrow}$ *row, the* $\mathsf{Field}^{\uparrow}$ *boundary, the* $\mathsf{Ideal}^{\uparrow}$ *comparability ledger, the* $\mathsf{LocalField}^{\uparrow}$ *handoff, and the nonescape row* Q*.*

Proof. Project the fraction-field boundary row F and the nonescape row Q. The row F names the field in which fractions are read, while Q records that later consumers may use only the comparisons already routed through R, F, I, L.

Thus inverse comparison and divisibility information remain inside the displayed NameCert packet. There is no row for an ambient field quotient beyond F, a selected valuation map, a chosen value group, or algebraic closure, so the local classifier cannot export such data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ValuationDomain}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ValuationDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6626.1, Definition 6626.2, Definition 6626.3, Definition 6626.4, and Definition 6626.5 seed the valuation-domain route through Ring, Field, Ideal, LocalField, fraction-boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ValuationDomain}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ValuationDomainUp`

Bridge status: none

Not claimed: This chapter does not close selected valuation maps, chosen value groups, discrete valuation data, Noetherianity, principal generation, residue-field construction, completion, algebraic closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fraction-field boundary exposure, ideal-chain comparability, local-field handoff, fraction-field nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ValuationDomain

Constructive story: A $\mathsf{ValuationDomain}^{\uparrow}$ packet is generated from a RingUp source, a FieldUp fraction-field boundary, an IdealUp comparability row, a LocalFieldUp handoff row, a fraction-field nonescape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6627

A Concrete Naming Certificate for $\mathsf{FastCauchyAccelerationSelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FastCauchyAccelerationSelector}^{\uparrow}$ packet records the finite selector surface that accelerates an ordinary regular Cauchy tail ledger into a fast Cauchy budget schedule. It sits between the selected-tail packet of chapter 458, the fast-subsequence route of chapter 1518, the dyadic rational core of chapter 291, the stream-name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name row of chapter 38.

The selector is finite: it names requested precision budgets, inspected tail windows, dyadic tolerances, regular-rational reads, and a fast-subsequence handoff. It does not add countable choice, quotient stream equality, a host real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6627.1 (Fast Cauchy acceleration selector carrier)**.** *A* $\mathsf{FastCauchyAccelerationSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, F, D, S, R, E, B, H, C, P, N)$$

with the following visible rows:

1. *T is an accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet that supplies the ordinary selected-tail witness for the inspected source;*
2. *F is the* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *handoff row that records the accelerated tail schedule consumed by fast Cauchy readers;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating each fast budget into the dyadic comparison checked on the selected tail;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose inspected windows are shared by the ordinary selector and the accelerated schedule;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the selected tail and dyadic budget rows are displayed;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only through T, F, D, S, R;*
7. *B is the finite acceleration-budget schedule, listing the displayed precision request, selected tail index, and fast-budget target for each inspected row;*

8. *H stores componentwise* hsame *transports for the selector, fast-subsequence, dyadic, stream, readback, seal, budget, provenance, and local naming rows;*
9. *C stores the displayed* Cont *routes by which completion-facing consumers may read the packet;*
10. *P is the* Pkg *provenance over* $\mathsf{FastCauchyAccelerationSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, FastCauchySubsequen $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyAccelerationSelector}^{\uparrow}}$ *row.*

No coordinate of A is a quotient of streams, a choice function on all precisions, a host-level real equality, a selected limit value, or a real-completeness theorem.

Theorem 6627.2 (Fast Cauchy acceleration selector NameCert obligations)**.** *The carrier of Definition 6627.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{FastCauchyAccelerationSelector}^{\uparrow}}$ *obligation surface. Its obligations are exactly:*

1. *carrier admission for $A = (T, F, D, S, R, E, B, H, C, P, N)$;*
2. *ordinary-tail admission, requiring T to be read as an accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet before any fast schedule is consumed;*
3. *fast-schedule admission, requiring F and B to expose the accelerated tail budget through* FastCauchySubsequence$^{\uparrow}$
4. *dyadic budget exactness, requiring each budget entry in B to be checked through D against the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows in S;*
5. *regular-readback exactness, requiring R to be reached only after T, F, D, S, B have been displayed;*
6. *real-seal non-escape, requiring E to expose only the finite acceleration route and no quotient stream, chosen limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the displayed finite BHist tuple. Each coordinate is either a named sibling packet, a finite budget row, transport data, a displayed continuation route, package provenance, or the local naming row.

The ordinary-tail and fast-schedule obligations are read in sequence. First T supplies the selected-tail witness from $\mathsf{RegularCauchyTailSelector}^{\uparrow}$; then F and B record the accelerated schedule consumed by FastCauchySubsequence$^{\uparrow}$.

Dyadic budget exactness follows by inspecting D and S together. Each budget in B is attached to a visible dyadic tolerance and to the same finite stream window that the ordinary selector inspected.

For regular readback, R is downstream of T, F, D, S, B. Since the carrier has no alternate readback coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer of the accelerated schedule must pass through those rows.

Finally E is reached only through the displayed route. The carrier lists no quotient stream, host equality, chosen limit, countable-choice selector, or ambient completeness coordinate, so the local NameCert surface is exhausted by the finite acceleration packet. □

Theorem 6627.3 (Fast Cauchy acceleration selector window readback)**.** *Every accepted* FastCauchyAccelerationSele *packet exposes the finite completion-facing route*

$$\mathsf{RegularCauchyTailSelector}^{\uparrow} \longmapsto \mathsf{FastCauchySubsequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

In the carrier notation $A = (T, F, D, S, R, E, B, H, C, P, N)$, a completion consumer may read only the selected-tail row T, the accelerated schedule F, B, the dyadic tolerance ledger D, the shared

stream window S, the regular-rational readback R, the real seal E, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Start from the NameCert obligations of Definition 6627.2. They force the ordinary selected tail T to be displayed before the accelerated schedule F, B is read.

Next read the dyadic and stream rows. The budget exactness clause attaches each accelerated request to D and S, so the fast schedule cannot bypass the inspected source window or replace the dyadic tolerance ledger.

Then consume the regular-rational and real-seal rows. The regular-readback obligation places R after T, F, D, S, B, and the real-seal non-escape obligation places E after the same finite route.

The structural rows H, C, P, N transport, replay, source, and name only this packet. Therefore the readback route exports the displayed finite acceleration selector and no quotient stream equality, hidden limit selector, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyAccelerationSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyAccelerationSelector}^{\uparrow}$ packet is generated from an accepted regular Cauchy tail selector, a fast-subsequence handoff, dyadic budget rows, shared stream windows, regular-rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyAccelerationSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed chapter names the finite carrier, ordinary tail-selector dependency, fast-subsequence schedule, dyadic budget, stream-window, regular-readback, and real-seal rows for a Bishop-style Cauchy acceleration selector.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchyAccelerationSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, quotient stream equality, countable choice, host real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Formalization should encode the finite carrier and prove the local NameCert obligation surface from the displayed selector rows.

Chapter 6628

A Concrete Naming Certificate for $\mathsf{LikelihoodRatioTest}^{\uparrow}$

Following Neyman and Pearson, the $\mathsf{LikelihoodRatioTest}^{\uparrow}$ packet records a finite decision ledger for comparing a displayed null hypothesis against a displayed alternative through likelihood-score rows. It sits after the hypothesis surface of chapter 231, the estimator ledger of chapter 230, the Fisher-information score route of chapter 228, the Bayesian likelihood row of chapter 232, and the probability source of chapter 184. The packet names only a finite test decision surface, not asymptotic optimality, host probability measure theory, selected critical regions, posterior odds, or measure-theoretic choice.

Definition 6628.1 (Likelihood-ratio test carrier)**.** *A* $\mathsf{LikelihoodRatioTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H_0, H_1, L, \tau, D, H, C, P, N)$$

with the following displayed rows:

1. H_0 *is the null-hypothesis row, read through the* $\mathsf{Hypothesis}^{\uparrow}$ *carrier and attached to the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source.*
2. H_1 *is the alternative-hypothesis row, read through the same hypothesis and probability-source surface.*
3. L *is the finite likelihood-score ledger, carrying the inspected observation windows, likelihood comparisons, estimator readbacks, Fisher-information score rows when present, and Bayesian likelihood row when present.*
4. τ *is the threshold row, displaying the finite critical comparison requested by the test.*
5. D *is the decision row, recording the accepted local decision after* H_0, H_1, L, τ *have been displayed.*
6. H *records componentwise* hsame *transport for the null hypothesis, alternative hypothesis, likelihood ledger, threshold, decision, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from the probability source and hypothesis rows through the likelihood ledger and threshold to the decision row.*
8. P *is the* Pkg *provenance over* $\mathsf{LikelihoodRatioTest}^{\uparrow}$, $\mathsf{Hypothesis}^{\uparrow}$, $\mathsf{Estimator}^{\uparrow}$, $\mathsf{FisherInfo}^{\uparrow}$, $\mathsf{Bayesian}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{LikelihoodRatioTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $H_0, H_1, L, \tau, D, H, C, P, N$. It has no coordinate for asymptotic power, uniformly most powerful optimality, posterior odds, selected rejection regions, ambient measure completion, or host probability choice.

Theorem 6628.2 (Likelihood-ratio test NameCert obligations)**.** *For every accepted* $\mathsf{LikelihoodRatioTest}^{\uparrow}$ *carrier $T = (H_0, H_1, L, \tau, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LikelihoodRatioTest}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the null hypothesis, alternative hypothesis, likelihood-score ledger, threshold, decision, transport, replay, provenance, and local naming rows;*
2. *hypothesis-source stability, requiring H_0 and H_1 to remain attached to the same* $\mathsf{ProbSpace}^{\uparrow}$ *source and to the hypothesis carrier of chapter 231;*
3. *likelihood-ledger stability, requiring every score comparison in L to be read through the displayed estimator, Fisher-information, Bayesian likelihood, or probability-source row named by chapter 230, chapter 228, chapter 232, and chapter 184;*
4. *threshold stability, requiring τ to be displayed before the decision row D is consumed;*
5. *decision nonescape, requiring D to export only the finite likelihood-ratio decision and not asymptotic optimality, posterior odds, selected critical regions, host measure completion, or measure-theoretic choice;*
6. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

These obligations certify only the finite statistical decision ledger.

Proof. Project the carrier of Definition 6628.1. Its visible rows are exactly the two hypothesis rows, likelihood-score ledger, threshold, decision, and structural rows.

Hypothesis-source stability is read from H_0 and H_1. Both rows point back to the displayed hypothesis carrier and probability source, so the test does not compare hypotheses from unrelated sources.

Likelihood-ledger stability is read from L. Each score comparison is attached to displayed observation, estimator, Fisher-information, Bayesian likelihood, or probability-source rows before the threshold can be applied.

The threshold and decision rows are downstream. The row τ supplies the finite comparison boundary, and D records the resulting local decision only after the ledger has been displayed.

Rows H, C, P, N transport, replay, source, and name the same packet. Since no additional coordinate stores asymptotic optimality, posterior odds, selected critical regions, host measure completion, or choice, the listed obligations exhaust the NameCert surface. □

Theorem 6628.3 (Likelihood-ratio test finite-ledger threshold stability)**.** *Let $T = (H_0, H_1, L, \tau, D, H, C, P, N)$ and $T' = (H_0', H_1', L', \tau', D', H', C', P', N')$ be accepted* $\mathsf{LikelihoodRatioTest}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies the null rows, alternative rows, likelihood ledgers, threshold rows, decision rows, replay rows, provenance rows, and local naming rows, then the threshold decision exposed by T is the same finite decision exposed by T'. The comparison is made only through the displayed hypothesis, estimator, Fisher-information, Bayesian likelihood, probability-source, transport, replay, provenance, and local NameCert rows.*

Proof. Start with the two accepted carriers. The classifier has no readable coordinates beyond $H_0, H_1, L, \tau, D, H, C, P, N$ and their primed counterparts.

Use componentwise hsame transport. The null and alternative hypotheses are compared through the displayed hypothesis and probability-source rows, the likelihood ledgers are compared

through their finite score entries, and the threshold rows are compared before either decision row is read.

The replay and provenance rows preserve the same route to the local decision. Because neither carrier contains an asymptotic optimality theorem, posterior-odds row, selected critical region, host measure completion, or choice principle, the transported threshold decision is the same finite ledger decision. □

Falsifiable prediction: Every $\mathsf{LikelihoodRatioTest}^{\uparrow}$ decision read factors through a null hypothesis row, an alternative hypothesis row, a finite likelihood-score ledger, a displayed threshold, a decision row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: hypothesis, estimator, fisherinfo, bayesian, probspace: $\mathsf{LikelihoodRatioTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two hypothesis rows, likelihood-score entries, a threshold row, a decision row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LikelihoodRatioTest}^{\uparrow}}$.

Constructive story: A $\mathsf{LikelihoodRatioTest}^{\uparrow}$ packet is generated from a null hypothesis row, an alternative hypothesis row, a finite likelihood-score ledger, a displayed threshold row, a decision row, componentwise hsame transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LikelihoodRatioTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6628.1, Definition 6628.2, and Definition 6628.3 seed the finite likelihood-ratio decision route through Hypothesis, Estimator, FisherInfo, Bayesian, ProbSpace, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LikelihoodRatioTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close asymptotic optimality, uniformly most powerful tests, posterior odds, selected critical regions, host probability measure theory, ambient measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, hypothesis-source stability, likelihood-ledger stability, threshold stability, decision nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LikelihoodRatioTestUp.

Chapter 6629

A Concrete Naming Certificate for $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$

Following Wald's sequential analysis, the $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ packet records a finite stopping-and-decision ledger for a likelihood-ratio test whose observations are inspected one window at a time. It sits after the hypothesis-test surface $\mathsf{HypothesisTest}^{\uparrow}$, the likelihood-ratio packet of chapter 6628, the probability source of chapter 184, and the distribution readback of chapter 186. The packet names the finite route by which stopping, continuation, and Type I/II error-budget rows are displayed; it does not claim asymptotic optimality, optional-stopping theorems, selected sample paths, ambient measure completion, or host probability choice.

Definition 6629.1 (Sequential probability ratio test carrier)**.** *A* $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (H_0, H_1, L, A, B, W, \sigma, D, E, H, C, P, N)$$

with the following displayed rows:

1. *H_0 is the displayed null-hypothesis row, read through the* $\mathsf{HypothesisTest}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *source surfaces.*
2. *H_1 is the displayed alternative-hypothesis row, read through the same hypothesis-test and probability-source surfaces.*
3. *L is the sequential likelihood-ratio ledger, whose finite prefixes are read through* $\mathsf{LikelihoodRatioTest}^{\uparrow}$ *and* $\mathsf{Distribution}^{\uparrow}$ *rows.*
4. *A is the lower stopping boundary row and B is the upper stopping boundary row.*
5. *W is the finite observation-window ledger inspected before the stopping row is exposed.*
6. *σ is the stopping row, recording whether the displayed finite prefix continues, accepts the null row, or accepts the alternative row.*
7. *D is the terminal decision row, available only when σ displays a terminal stopping outcome.*
8. *E is the Type I/II error-budget route, naming the finite budget rows attached to A, B, L, W, σ, D.*
9. *H records componentwise* hsame *transport for the hypotheses, likelihood ledger, boundaries, observation windows, stopping row, decision row, budget route, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from the probability source and distribution rows through the sequential likelihood ledger to stopping and decision.*

11. *P is the* Pkg *provenance over* $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$, $\mathsf{HypothesisTest}^{\uparrow}$, $\mathsf{LikelihoodRatioTest}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialProbabilityRatioTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for asymptotic optimality, optional-stopping principles, selected infinite sample paths, external likelihood calibration, host measure completion, or probability-space choice.

Theorem 6629.2 (Sequential probability ratio test NameCert obligations)**.** *For every accepted* $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ *carrier* $S = (H_0, H_1, L, A, B, W, \sigma, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialProba}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the null hypothesis, alternative hypothesis, sequential likelihood ledger, lower boundary, upper boundary, observation windows, stopping row, decision row, error-budget route, transport, replay, provenance, and local naming;*

2. *hypothesis-source stability, requiring H_0 and H_1 to stay attached to the same* $\mathsf{HypothesisTest}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *source rows;*

3. *likelihood-prefix stability, requiring every prefix in L and W to be read through the displayed* $\mathsf{LikelihoodRatioTest}^{\uparrow}$ *and* $\mathsf{Distribution}^{\uparrow}$ *rows before a stopping comparison is consumed;*

4. *boundary stability, requiring the lower and upper rows A, B to be displayed before σ can classify a prefix as continuing or terminal;*

5. *stopping and decision nonescape, requiring σ, D to export only the displayed sequential decision route and not an optional-stopping theorem, selected infinite path, host probability computation, or ambient measure-completion object;*

6. *error-budget routing, requiring E to name only the Type I and Type II budget rows carried by the displayed hypotheses, likelihood prefixes, boundaries, stopping row, and decision row.*

These obligations certify only the finite sequential decision packet.

Proof. Project the carrier of Definition 6629.1. Its analytic rows are the two hypothesis rows, sequential likelihood ledger, two boundary rows, finite observation windows, stopping row, terminal decision, and error-budget route.

The source obligations come first. Rows H_0 and H_1 keep the null and alternative hypotheses on the same $\mathsf{HypothesisTest}^{\uparrow}$ and $\mathsf{ProbSpace}^{\uparrow}$ source, while L and W expose only finite likelihood prefixes read through $\mathsf{LikelihoodRatioTest}^{\uparrow}$ and $\mathsf{Distribution}^{\uparrow}$.

The stopping obligations are downstream of those rows. The boundaries A and B are displayed before the row σ can decide whether the finite prefix continues or terminates, and the terminal decision row D is visible only after a terminal stopping outcome has been displayed.

The budget and structural rows exhaust the packet. Row E records the Type I/II budget route attached to the same finite data, while H, C, P, N transport, replay, source, and name the carrier. Since no other coordinate is present, the listed obligations exclude optional-stopping principles, selected sample paths, ambient probability computation, and measure-completion data. □

Theorem 6629.3 (Sequential probability ratio test stopping-ledger stability)**.** *Let* $S = (H_0, H_1, L, A, B, W, \sigma, D, E$ *and* $S' = (H_0', H_1', L', A', B', W', \sigma', D', E', H', C', P', N')$ *be accepted* $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies the hypothesis rows, likelihood-prefix ledgers,*

boundary rows, observation windows, stopping rows, decision rows, error-budget routes, replay rows, provenance rows, and local naming rows, then the stopping outcome and terminal decision exposed by S are the same finite sequential read exposed by S'. The comparison is made only through the displayed hypothesis-test, likelihood-ratio, probability-space, distribution, transport, replay, provenance, and local NameCert rows.

Proof. Start with the two accepted carriers. Their classifiers have no readable coordinates beyond the rows listed in Definition 6629.1.

Use componentwise hsame transport. The hypothesis rows remain on the same source surface, the likelihood prefixes and observation windows transport through the displayed $\mathsf{LikelihoodRatioTest}^{\uparrow}$ and $\mathsf{Distribution}^{\uparrow}$ rows, and the boundary rows transport before either stopping row is inspected.

The stopping and decision rows are therefore compared along the same finite route. The replay, provenance, and local naming rows keep that route attached to the same packet, so the transported carrier exposes the same continuation or terminal decision and cannot import an excluded optional-stopping or infinite-path principle. □

Theorem 6629.4 (Sequential probability ratio test error-budget route)**.** *Every Type I or Type II error-budget read exported by a* $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{HypothesisTest}^{\uparrow},\quad \mathsf{LikelihoodRatioTest}^{\uparrow},\quad \mathsf{ProbSpace}^{\uparrow},\quad \mathsf{Distribution}^{\uparrow},\quad L,\quad A,\quad B,\quad W,\quad \sigma,\quad D.$$

Thus a downstream decision consumer receives only displayed hypothesis rows, likelihood-prefix rows, probability and distribution source rows, boundary rows, finite observation windows, stopping and decision rows, componentwise transport, replay, provenance, and local naming before the budget route is visible.

Proof. Project the budget row E from the carrier. By definition it is attached to the hypotheses, likelihood prefixes, boundaries, finite observation windows, stopping row, and terminal decision row.

The upstream rows are displayed before E is consumed. Hypothesis and probability rows provide the source, likelihood-ratio and distribution rows provide the finite prefix readback, and A, B, W, σ, D provide the stopping and decision ledger.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since E has no coordinate for asymptotic optimality, selected paths, host probability computation, optional-stopping theorems, or ambient measure completion, the error-budget read factors only through the finite rows stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialProbabilityRatioTest}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialProbabilityRatioTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6629.1, Definition 6629.2, Definition 6629.3, and Definition 6629.4 seed the finite stopping, decision, and Type I/II budget route through hypothesis-test, likelihood-ratio, probability-space, distribution, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialProbabilityRatioTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close asymptotic optimality, optional-stopping theorems, selected infinite sample paths, ambient measure completion, host probability choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, hypothesis-source stability, likelihood-prefix stability, boundary stability, stopping-ledger stability, error-budget routing, transport, replay, provenance, and local naming for BEDC.Derived.SequentialProbabilityRatioTestUp.

Constructive story: A $\mathsf{SequentialProbabilityRatioTest}^{\uparrow}$ packet is generated from displayed null and alternative hypothesis rows, a sequential likelihood-prefix ledger, lower and upper boundary rows, finite observation windows, a stopping

row, a terminal decision row, a Type I/II error-budget route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6630

Compact Completion Dense Image Name Certificate

Following Bishop and Bridges, the $\mathsf{CompactCompletionDenseImage}^{\uparrow}$ packet records the finite route by which compact metric source data enters a completion-facing dense-image certificate. It sits between the compact metric source of chapter 129, the Cauchy-thread and limit-service surface of chapter 128, and the terminal real-name seal of chapter 38. The packet gives completion consumers one displayed NameCert route for compactness, density, regular readback, and real sealing; it does not choose completion representatives, identify quotient classes, prove open-cover compactness, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6630.1 (Compact completion dense image carrier)**.** *A* $\mathsf{CompactCompletionDenseImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, M, E, T, S, R, F, H, C, P, N)$$

with the following displayed rows:

1. *K is a* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite metric and compactness-facing certificate data available before completion is read.*
2. *M is a* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row for the completion-facing metric service consumed by the dense-image route.*
3. *E is the dense embedding row from the compact source into the completion surface; it records only displayed image membership and dense-window access.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by finite density requests.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row attached to the dense-image request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row consumed before any real-facing seal is inspected.*
7. *F is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for completion-facing distance and limit-name requests.*
8. *H records componentwise* hsame *transport for the compact source, completion handoff, dense embedding, tolerance ledger, finite windows, regular readback, real seal, replay, provenance, and local naming row.*
9. *C records displayed* Cont *replay from a compact-source request through $K, M, E, T, S, R, F, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactCompletionDenseImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCompletionDenseImage}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. No coordinate stores an open-cover compactness proof, a selected completion representative, quotient equality of completion names, a host dense-image equality theorem, countable choice, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6630.2 (CompactCompletionDenseImage NameCert obligations)**.** *For every accepted* $\mathsf{CompactCompletionDenseImage}^{\uparrow}$ *carrier*

$$D = (K, M, E, T, S, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactCompletionDenseImage}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact-source admission: the source row K is the only compact metric data made visible to the dense-image certificate;*
2. *completion handoff: the row M is the only completion-facing metric service read before the dense image can be consumed;*
3. *dense-image exposure: the row E exposes image membership and finite density windows without selecting representatives in the completion;*
4. *tolerance-window readback: the rows T, S, R force every density request to pass through the dyadic tolerance ledger, finite stream window, and regular rational readback before the real seal F;*
5. *provenance and name exhaustion: the rows H, C, P, N transport, replay, package, and name the same route and introduce no open-cover compactness proof, selected completion representative, quotient equality, host equality, countable choice, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6630.1. The visible completion route begins at the compact metric row K, reaches the completion handoff M, and exposes density only through the displayed embedding row E.

The finite readback rows are then forced in order. A density request first reads the dyadic tolerance ledger T, then the finite observation window S, then the regular rational readback row R, and only after those rows may it inspect the terminal real seal F.

Finally use the structural rows. Componentwise hsame transport in H and replay in C preserve the same compact source, completion handoff, dense embedding, tolerance, window, readback, and real-seal rows, while P and N record provenance and local naming.

Since the carrier has no coordinate for open-cover compactness, selected completion representatives, quotient equality, host equality, countable choice, or full $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCompletionDenseImage}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCompletionDenseImage}^{\uparrow}$ packet is generated from a CompactMetricUp source, a CompleteMetricUp completion handoff, a dense embedding row, a DyadicRatCoreUp tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCompletionDenseImage}^{\uparrow}}$, seed)

Scope closed: Definition 6630.1 and Definition 6630.2 seed the finite carrier and local NameCert obligation rows for compact completion dense-image packets.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactCompletionDenseImage}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected completion representatives, quotient equality, full Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, completion handoff routing, dense-image exposure, dyadic tolerance discipline, StreamName window locality, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactCompletionDenseImageUp.

Chapter 6631

A Concrete Naming Certificate for $\mathsf{ClassEquation}^{\uparrow}$

Following Artin, the class equation records the conjugacy-orbit decomposition of a finite group as the sum of the center and the noncentral conjugacy classes. The $\mathsf{ClassEquation}^{\uparrow}$ packet consumes the finite group rows of chapter 41, the orbit-stabilizer handoff of chapter 6559, the orbit-counting ledger of chapter 3578, and the Sylow-facing subgroup route of chapter 3728. It names only the displayed conjugation action, center row, noncentral orbit ledger, stabilizer-index accounting, and finite counting read; it does not import a host quotient by conjugacy, selected representatives, a full subgroup lattice, or a character table.

Definition 6631.1 (ClassEquation carrier)**.** *A* $\mathsf{ClassEquation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (G, Z, A, O, S, L, H, C, P, N)$$

with the following rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, including multiplication, unit, inverse, finite support, and componentwise group transport.*
2. *Z is the displayed center row, read as the finite list of group endpoints whose conjugation action is fixed by every visible group row.*
3. *A is the conjugation-action row of G on its displayed finite support.*
4. *O is the finite noncentral conjugacy-orbit ledger, with each retained orbit read through the action row A.*
5. *S is the stabilizer-index ledger supplied by the orbit-stabilizer route; it records the centralizer-style stabilizer row and the finite orbit-size read for each noncentral orbit.*
6. *L is the class-equation counting row, displaying the center-orbit split*

$$\mathsf{groupSize}(G) = \mathsf{centerSize}(Z) + \mathsf{noncentralOrbitSum}(O, S).$$

7. *H records componentwise* hsame *transport for G, Z, A, O, S, L.*
8. *C records displayed* Cont *replay from group rows through conjugation, center exposure, orbit-stabilizer accounting, and the final counting row.*

9. *P is the* Pkg *provenance over* $\mathsf{ClassEquation}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, $\mathsf{BurnsideOrbitCount}^{\uparrow}$, $\mathsf{Sylow}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassEquation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host conjugacy quotient, selected class representatives, quotient subgroup equality, character-theoretic class functions, or an ambient finite-group structure theorem.

Theorem 6631.2 (ClassEquation NameCert obligations)**.** *For every accepted* $\mathsf{ClassEquation}^{\uparrow}$ *carrier* $E = (G, Z, A, O, S, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClassEquation}^{\uparrow}}$ *surface consists of five finite rows: carrier admission for the displayed group, center, conjugation-action, orbit, stabilizer-index, counting, transport, replay, provenance, and naming data; classifier stability under componentwise* hsame *for those same rows; center-orbit split exactness for the fixed conjugation rows and the noncentral orbit ledger; orbit-stabilizer ledger replay through chapter 6559; and nonescape, requiring every class-equation consumer to factor through* $G, Z, A, O, S, L, H, C, P, N$ *without receiving a host conjugacy quotient, selected representative, subgroup lattice, character table, or ambient finite-group classification.*

Proof. Carrier admission is projection of Definition 6631.1. The group row G, center row Z, conjugation action A, noncentral orbit ledger O, stabilizer-index ledger S, and counting row L are all finite BHist data.

Classifier stability is componentwise. The group coordinate is compared by the finite $\mathsf{Group}^{\uparrow}$ classifier, the center and action rows are transported by the same hsame ledger, and the orbit, stabilizer, and counting rows are replayed through H and C.

The center-orbit split is read before the final counting row. Fixed conjugation rows enter through Z, while noncentral rows enter through O and then through the orbit-stabilizer ledger S; the resulting finite sum is exactly the displayed row L.

The orbit-stabilizer dependency is the sibling route of chapter 6559, and the finite orbit accounting is compatible with the orbit-counting surface of chapter 3578. Thus Sylow-style consumers may cite the class-equation row only through the displayed group and subgroup-facing ledgers.

Nonescape follows from the packet inventory. Since no coordinate contains a host conjugacy quotient, selected representative, subgroup lattice, character table, or ambient classification theorem, no accepted consumer read can extract one from $\mathsf{ClassEquation}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClassEquation}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassEquation}^{\uparrow}$ packet is generated from a finite GroupUp row, a displayed center row, a conjugation-action row, a noncentral orbit ledger, an orbit-stabilizer index ledger, a class-equation counting row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassEquation}^{\uparrow}}$, seed)

Scope closed: Definition 6631.1 and Definition 6631.2 seed the finite class-equation packet over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, $\mathsf{BurnsideOrbitCount}^{\uparrow}$, and $\mathsf{Sylow}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassEquation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host conjugacy quotient, selected class representatives, subgroup-lattice classification, character-table computation, or ambient finite-group classification.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, center-orbit split exactness, orbit-stabilizer ledger replay, and nonescape for BEDC.Derived.ClassEquationUp.

Chapter 6632

A Concrete Naming Certificate for $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ packet records the finite filter data that constructive analysis can inspect before an ultrafilter or unrestricted selector would be needed. It sits beside the finite refusal surfaces of chapter 1872, chapter 1739, chapter 1173, and chapter 677, while its scheduled observations use the finite windows of chapter 126. The packet names the boundary between finite-intersection filter witnesses and nonconstructive ultrafilter selection; it does not supply a maximal filter, a quotient-selected extension, a host Boolean-prime-ideal theorem, countable choice, or ambient completion data.

Definition 6632.1 (Constructive ultrafilter boundary carrier)**.** *A* $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, I, E, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite filter-base row, listing the inspected base members, declared ambient meet operation, inhabited witness rows, and monotonicity data visible to the packet.*
2. *I is the finite-intersection ledger, recording each displayed finite meet request and the witness that the resulting row remains inside the generated filter base.*
3. *E is the extension-request row, naming the requested ultrafilter or maximal-filter extension and the finite data the request is allowed to inspect.*
4. *R is the ultrafilter and choice-refusal row. It records that the packet answers E only by finite witness replay and by refusal accounting, not by producing a maximal filter or selector.*
5. *H records componentwise* hsame *transport for the filter base, finite-intersection ledger, extension request, refusal row, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from finite base and intersection data to the refusal boundary.*
7. *P is the* Pkg *provenance over* $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$, $\mathsf{CountableChoiceBoundary}^{\uparrow}$, $\mathsf{UniqueChoicePrinciple}$ $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{ChoiceRecipeLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, I, E, R, H, C, P, N. It has no coordinate for hidden maximality, quotient Boolean-algebra equality, global ultrafilter selection, countable-choice enumeration, or host-side excluded middle.

Theorem 6632.2 (Finite filter witness row)**.** *For every accepted* $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ *carrier $U = (F, I, E, R, H, C, P, N)$, every filter witness available to a consumer is read through the finite base row F and finite-intersection ledger I, then transported and replayed through H, C, P, N. The consumer cannot read a maximal filter, an ultrafilter extension, or a selected extension family from those rows.*

Proof. Project the carrier of Definition 6632.1. The only rows carrying positive filter data are the finite base row F and the finite-intersection ledger I.

The structural rows do not add new filter members. They transport the displayed rows through H, replay the same finite route through C, record source data in P, and name the local packet through N.

The extension request E is therefore a demand boundary, not a witness source. Since maximality, ultrafilter selection, quotient equality, countable choice, and excluded middle have no carrier coordinate, every available filter witness is exhausted by the displayed finite rows. □

Theorem 6632.3 (No ultrafilter selector nonescape)**.** *An accepted* $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ *packet may record an ultrafilter-extension request, but every answer to that request factors through the refusal row R and the finite witness route F, I, H, C, P, N. Thus the packet exports no ultrafilter selector, no maximal-filter extension, no Boolean-prime-ideal principle, no countable-choice schedule, and no hidden host predicate deciding all subsets.*

Proof. Start from the extension-request row E. By the carrier definition, E is paired with the refusal row R before any consumer can interpret the request.

The refusal row is downstream of the finite data. It can cite the inspected base row F and intersection ledger I, but it has no field that returns a maximal filter or a selector.

Use the sibling boundaries named in the provenance row. chapter 1872, chapter 1739, chapter 1173, and chapter 677 keep selector demand, unique-choice demand, Cauchy choice demand, and finite recipe demand separate from a produced choice output.

Finally the $\mathsf{StreamName}^{\uparrow}$ dependency supplies only finite inspection windows. Transport, replay, provenance, and local naming preserve the same refusal route, so the extension request cannot escape into an ultrafilter selector or host maximality theorem. □

Falsifiable prediction: If an accepted packet displays only F, I, H, C, P, N, an ultrafilter-extension request can be recorded only as the refusal row R; it cannot be read as a selected maximal filter.

Independence witness: countable-choice-boundary, unique-choice-principle, cauchy-choice-boundary, choice-recipe-ledger, streamname: $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ is not bijective to any listed sibling because it binds finite filter bases, finite-intersection ledgers, extension requests, ultrafilter refusal, transport, replay, provenance, and local NameCert rows in one boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$ packet is generated from finite filter-base rows, finite-intersection ledgers, extension-request rows, ultrafilter and choice-refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6632.1, Definition 6632.2, and Definition 6632.3 seed the finite filter-witness boundary against ultrafilter selection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilter existence, maximal-filter selection, Boolean-prime-ideal transfer, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-intersection ledger stability, extension-request refusal, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveUltrafilterBoundaryUp.

Chapter 6633

A Concrete Naming Certificate for CompactLebesgueNumberLedger$^{\uparrow}$

Following Bishop–Bridges and the constructive Heine–Cantor finite-cover proof, the CompactLebesgueNumberLedge packet records the finite radius fold behind the compact-uniform-continuity vision of chapter 9237. It sits over the compact finite-net source of chapter 129, the pointwise continuity rows of chapter 130, the tolerance-only output of chapter 369, the metric comparison carrier of chapter 57, and the rational and real seal rows of chapter 291, chapter 301, and chapter 38. The packet names the finite cover/radius/minimum fold; it does not import arbitrary open-cover compactness, a selected subcover theorem, host function equality, or ambient Real completeness.

Definition 6633.1 (Compact Lebesgue-number ledger carrier)**.** *A* CompactLebesgueNumberLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, M, C, R, D, Q, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-cover row, listing the displayed centers, cover members, membership witnesses, and finite inspection order.*
2. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-radius row attached to the centers and cover members named by K.*
3. *M is the* Met$^{\uparrow}$ *comparison ledger used to read source distances, target distances, and triangle estimates for the same finite cover route.*
4. *C is the finite cover-refinement row, pairing each inspected compact witness with the cover member and local radius that protects it.*
5. *R is the finite positive-radius family row, recording the candidate Lebesgue radii generated from the displayed cover and pointwise-modulus data.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *radius ledger for positivity, halving, minimum comparison, and strict lower-bound transport.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *seal row used only to read numerical comparisons already displayed by R, D, M.*
8. *U is the* UniformModulus$^{\uparrow}$ *handoff row produced by the finite minimum fold.*

9. *H records componentwise* hsame *transport for the compact cover, continuous-map radius rows, metric comparisons, cover refinement, radius family, dyadic ledger, real seal, uniform-modulus handoff, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay from a requested tolerance through $K, F, M, C, R, D, Q, H, P, N$ to the handoff row U.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactLebesgueNumberLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumberLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through $K, F, M, C, R, D, Q, U, H, T, P, N$. It has no coordinate for arbitrary open-cover compactness, selected subcovers, quotient equality of real names, hidden radius choice, or host equality of maps.

Lean4 ✓ `BEDC.Derived.CompactLebesgueNumberLedgerUp.CompactLebesgueNumberLedgerTasteGate single carrier alignment`

Theorem 6633.2 (Compact finite-cover radius fold)**.** *Every accepted* $\mathsf{CompactLebesgueNumberLedger}^{\uparrow}$ *packet turns compact finite-cover data into a tolerance-facing* $\mathsf{UniformModulus}^{\uparrow}$ *row by reading the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The finite minimum fold uses only the displayed cover row K, pointwise radius row F, metric ledger M, cover-refinement row C, radius family R, dyadic ledger D, and numerical seal row Q before exporting the handoff U.

Proof. Project an accepted carrier $L = (K, F, M, C, R, D, Q, U, H, T, P, N)$. The compact source and finite cover data are exactly K, while pointwise local radii are supplied by F.

The cover-refinement and metric rows bind those local radii to the same compact source. The row C records which displayed cover member protects each inspected witness, and M records the triangle and distance comparisons consumed by the fold.

The positive-radius family and dyadic ledger then perform the finite minimum step. The rows R, D expose positivity, halving, and lower-bound comparisons, and Q seals only the numerical reads already present in those rows.

Finally U, H, T, P, N export, transport, replay, source, and name the tolerance-only uniform-modulus handoff. No excluded compactness, selected subcover, hidden radius choice, quotient real equality, or host map equality has a carrier coordinate. □

This finite-radius fold is the predecessor consumed by the compact-uniform modulus route of chapter 388; locally it is the radius-source sibling of chapter 452, because the compact-uniform consumer reads the displayed K, F, M, C, R, D, Q ledger before any $\mathsf{UniformModulus}^{\uparrow}$ handoff is available.

Theorem 6633.3 (Compact Lebesgue-number nonescape)**.** *A consumer of an accepted* CompactLebesgueNumberLe *packet cannot read a Lebesgue number, compact cover bound, or uniform-modulus handoff without displaying the compact cover row K, pointwise-radius row F, metric comparison row M, refinement row C, radius family R, dyadic ledger D, RegSeqRat and Real seal row Q, transport row H, replay row T, provenance row P, and local* NameCert *row N.*

Proof. Use Definition 6633.1. The rows listed in the statement are the entire accepted packet and the only route to the handoff U.

The finite fold theorem above identifies the order in which those rows are consumed. Cover data and pointwise radii precede metric comparison, metric comparison precedes the positive-radius family, and the dyadic and real seal rows precede the uniform-modulus handoff.

Transport and replay preserve that route rather than adding another source of compactness or radius choice. Therefore a consumer asking for an arbitrary open-cover proof, selected subcover, hidden radius, quotient real equality, host map equality, or ambient Real completeness has left the accepted packet. □

Falsifiable prediction: If R, D, Q are absent, an accepted compact finite-cover read cannot be consumed as a $\mathsf{UniformModulus}^{\uparrow}$ handoff through $\mathsf{CompactLebesgueNumberLedger}^{\uparrow}$.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric, dyadicratcore, regseqrat, real: $\mathsf{CompactLebesgueNumberLedger}^{\uparrow}$ is not bijective to any listed sibling because it binds compact cover rows, pointwise radius rows, metric comparisons, finite refinement, radius-family folds, dyadic comparisons, real seals, uniform-modulus handoff, transport, replay, provenance, and local NameCert rows in one finite ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumberLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactLebesgueNumberLedger}^{\uparrow}$ packet is generated from compact finite-cover rows, ContinuousMap pointwise-radius rows, Metric comparison ledgers, finite cover-refinement rows, positive-radius families, DyadicRatCore folds, RegSeqRat and Real seal rows, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumberLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6633.1, Definition 6633.2, and Definition 6633.3 seed the compact finite-cover radius fold used by uniform-modulus extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumberLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, quotient real equality, hidden radius choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover exposure, pointwise-radius compatibility, metric comparison replay, finite radius-fold exactness, dyadic and Real seal discipline, UniformModulus handoff, and non-escape for BEDC.Derived.CompactLebesgueNumberLedgerUp.

Chapter 6634

A Concrete Naming Certificate for $\mathsf{HankelOperator}^{\uparrow}$

Following Hankel's moment-matrix tradition and the operator-theoretic reading of anti-diagonal kernels, the $\mathsf{HankelOperator}^{\uparrow}$ packet records the finite BEDC surface that a moment or kernel consumer may inspect before any infinite Hankel operator is named. It sits between the moment-window rows of chapter 6193 and chapter 6454, the finite matrix carrier of chapter 49, the inner-product readback of chapter 86, the scalar seal of chapter 38, and the collision-kernel shadow boundary of chapter 444. The certificate names only displayed moment rows, finite index windows, anti-diagonal matrix rows, shift rows, boundedness ledgers, transport, replay, provenance, and local naming.

Definition 6634.1 (Hankel operator carrier)**.** *A* $\mathsf{HankelOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, I, A, S, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite moment-row ledger, read from a displayed* $\mathsf{MomentProblem}^{\uparrow}$ *or* $\mathsf{HausdorffMomentProblem}^{\uparrow}$ *window and sealed through* $\mathsf{Real}^{\uparrow}$ *scalar rows;*
2. *I is the finite index-window row, listing exactly which source and target indices are visible to the packet;*
3. *A is the anti-diagonal* $\mathsf{Mat}^{\uparrow}$ *row whose entry at the displayed pair (i, j) is read from the moment row indexed by $i + j$ when that index is present in I;*
4. *S is the finite shift row, recording the visible source-window and target-window shift by which a Hankel-shadow consumer reuses A;*
5. *B is the boundedness ledger, containing only displayed finite norm, inner-product, or tolerance rows from* $\mathsf{InnerProduct}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*;*
6. *H records componentwise* hsame *transport for moment rows, index windows, anti-diagonal matrix rows, shift rows, boundedness rows, replay, provenance, and the local naming row;*
7. *C records displayed* Cont *replay from a moment-window request through I, A, S, B, H before any downstream read sees the packet;*
8. *P is the* Pkg *provenance over* $\mathsf{HankelOperator}^{\uparrow}$*,* $\mathsf{MomentProblem}^{\uparrow}$*,* $\mathsf{HausdorffMomentProblem}^{\uparrow}$*,* $\mathsf{CollisionKernel}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, I, A, S, B, H, C, P, N$. It has no coordinate for an infinite matrix, ambient bounded operator, representing measure, spectral theorem, selected domain completion, or host Hilbert-space equality.

Theorem 6634.2 (Hankel operator NameCert obligations)**.** *For every accepted* $\mathsf{HankelOperator}^{\uparrow}$ *carrier $K = (M, I, A, S, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the moment rows, finite index window, anti-diagonal matrix row, shift row, boundedness ledger, transport, replay, provenance, and local name;*
2. *moment-window exactness, requiring every entry of A to be read from M using an index named by I;*
3. *anti-diagonal exactness, requiring the displayed* $\mathsf{Mat}^{\uparrow}$ *row to depend only on the visible sum index $i + j$;*
4. *shift-moment stability, requiring the row S to transport moment-window reads without adding an unlisted source or target index;*
5. *boundedness nonescape, requiring B to expose only finite inner-product, norm, or tolerance rows and no ambient bounded-operator theorem;*
6. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

No Hankel obligation is stored outside this finite packet.

Proof. Project the carrier of Definition 6634.1. The visible rows are exactly $M, I, A, S, B, H, C, P, N$.

Moment-window exactness and anti-diagonal exactness read M, I, A together. The matrix row is admitted only when each displayed entry uses a visible moment row at the displayed sum index.

Shift-moment stability is the row S. It reuses the same finite index and moment data under the displayed source-target shift, while H and C transport and replay only those rows.

The boundedness and nonescape clauses follow from B, P, N. The provenance row names the moment, collision-kernel, matrix, inner-product, and real scalar sources, and the local naming row contains no coordinate for an infinite operator, spectral object, representing measure, selected completion, or host Hilbert-space equality. □

Theorem 6634.3 (Hankel operator shift-moment route)**.** *Every shift-moment read of an accepted* $\mathsf{HankelOperator}^{\uparrow}$ *packet factors through the finite moment ledger M, the index-window row I, the anti-diagonal matrix row A, the shift row S, the boundedness ledger B, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}$ *row. Thus the read is a positive finite carrier for the Hankel-shadow routes used by* $\mathsf{MomentProblem}^{\uparrow}$*,* $\mathsf{HausdorffMomentProblem}^{\uparrow}$*, and* $\mathsf{CollisionKernel}^{\uparrow}$*, not an infinite Hankel operator.*

Proof. Start from the NameCert obligations of Definition 6634.2. They force a shift-moment consumer to expose M, I, A, S before any Hankel-shadow read is exported.

Use the moment-window exactness and anti-diagonal exactness clauses. A requested shifted entry is read from the same finite moment ledger by the visible sum-index rule, and no unlisted index can be manufactured by the carrier.

Next consume the boundedness ledger and structural rows. The row B supplies only displayed finite inner-product, norm, or tolerance evidence, while H, C, P, N transport, replay, source, and name the same finite packet.

Consequently the route supplies the positive finite carrier requested by the moment and collision-kernel chapters. Since the carrier lacks an infinite matrix, ambient bounded-operator theorem, spectral row, representing measure, selected completion, or host equality, the shift-moment read stops at the displayed finite Hankel surface. □

Lean4 (statement) `BEDC.Derived.HankelOperatorUp`

Falsifiable prediction: An accepted $\mathsf{HankelOperator}^{\uparrow}$ read is refuted if a Hankel-shadow consumer reaches an anti-diagonal or shift-moment conclusion without first displaying the moment ledger, finite index window, anti-diagonal matrix row, shift row, boundedness ledger, transport row, replay row, provenance row, and local naming row.

Independence witness: moment-problem, Hausdorff moment problem, collision kernel, matrix, inner product, real scalar: $\mathsf{HankelOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries moment rows, finite index windows, anti-diagonal matrix data, shift-moment rows, boundedness ledgers, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{HankelOperator}^{\uparrow}$ packet is generated from finite moment rows, finite index windows, anti-diagonal matrix rows, shift rows, boundedness ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6634.1, Definition 6634.2, and Definition 6634.3 seed the finite Hankel carrier consumed by moment-window, collision-kernel, matrix, inner-product, and real scalar rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HankelOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, infinite Hankel operators, ambient bounded-operator theory, spectral data, representing measures, selected domain completions, or host Hilbert-space equality.

Upgrade path: Obligation closure requires checked carrier admission, moment-window exactness, anti-diagonal exactness, shift-moment stability, boundedness nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HankelOperatorUp.

Chapter 6635

A Concrete Naming Certificate for $\mathsf{BaireFunctionRank}^{\uparrow}$

Following Hausdorff, Kuratowski, and Kechris in classical descriptive set theory, the $\mathsf{BaireFunctionRank}^{\uparrow}$ packet records the finite BEDC route by which a real-valued function on a Polish source exposes a displayed Baire-class rank to downstream consumers. It sits over the complete separable source of chapter 1490, the displayed rank schedule of chapter 4020, the Borel-function row of chapter 3479, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the real seal of chapter 38. The packet names a rank-read route only; it does not assert Baire hierarchy completeness, selected codes, quotient equality of functions, countable choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6635.1 (Baire-function rank carrier)**.** *A* $\mathsf{BaireFunctionRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, B, F, S, Q, E, H, C, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row for the domain of the real-valued function.*
2. *B is the* $\mathsf{BorelHierarchy}^{\uparrow}$ *rank schedule row, including the visible predecessor, successor, and limit-stage windows used by the current rank read.*
3. *F is the* $\mathsf{BorelFunction}^{\uparrow}$ *row that hands the source-side function query to the displayed Borel schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for finite observation of source names, rank positions, and real-facing output windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used before the rank read reaches the real seal.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the real-valued output readback.*
7. *H records componentwise* hsame *transport for the Polish source, rank schedule, Borel-function row, stream windows, regular-rational readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a source query through the rank schedule, Borel-function handoff, readback, and real seal.*
9. *G is the* Pkg *provenance over the sibling surfaces named above.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireFunctionRank}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, B, F, S, Q, E, H, C, G, N$. It has no coordinate for a host-space function, a chosen Baire code, quotient equality of functions, countable-choice selection, completed measure structure, or ambient real-completion principle.

Theorem 6635.2 (Baire-function rank NameCert obligations)**.** *For every accepted* $\mathsf{BaireFunctionRank}^{\uparrow}$ *carrier*

$$R = (P, B, F, S, Q, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireFunctionRank}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *Polish source exposure, requiring the domain read to pass through P before any rank-facing row is consumed;*
3. *Borel rank schedule exposure, requiring every inspected predecessor, successor, or limit-stage window to be displayed in B;*
4. *Borel-function handoff, requiring the function row F to connect the source query to the rank schedule before the real readback is used;*
5. *Real readback nonescape, requiring the route through S, Q, E before any real-facing consumer receives a rank value;*
6. *transport, requiring H to preserve $P, B, F, S, Q, E, C, G, N$ componentwise;*
7. *continuation replay, requiring C to replay the same ordered route for each accepted consumer;*
8. *provenance, requiring G to cite only the displayed PolishSpace, BorelHierarchy, BorelFunction, StreamName, RegSeqRat, and Real sibling surfaces;*
9. *local naming, requiring N to certify exactly the rows $P, B, F, S, Q, E, H, C, G$.*

No $\mathsf{BaireFunctionRank}^{\uparrow}$ *obligation stores a selected code, host function quotient, countable-choice selector, measure-completion datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 6635.1. The tuple displays the Polish source, Borel rank schedule, Borel-function handoff, stream windows, regular rational readback, real seal, transport, replay, provenance, and local naming rows.

The rank-facing read starts at P, B, F. The Polish source is exposed first, the Borel hierarchy row supplies only visible rank windows, and the Borel-function row is the single handoff from the source query to that schedule.

The real-facing part is downstream of S, Q, E. Stream windows and regular rational readback are consumed before the real seal, so no consumer can replace the route by host equality, selected limits, or ambient real completeness.

Finally H, C, G, N transport, replay, source, and name the same finite packet. Since no row contains a selected Baire code, quotient function equality, countable-choice selector, or measure-completion object, the enumerated obligations exhaust the local certificate surface. □

Theorem 6635.3 (Baire-function rank Borel handoff)**.** *Every rank read of an accepted* $\mathsf{BaireFunctionRank}^{\uparrow}$ *carrier*

$$R = (P, B, F, S, Q, E, H, C, G, N)$$

passes through

$$P \longmapsto B \longmapsto F \longmapsto S \longmapsto Q \longmapsto E$$

before any real-facing consumer may use it. The structural rows H, C, G, N only transport, replay, source, and name that route; they do not add a selected Baire code, quotient equality of functions, countable-choice schedule, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 6635.2. The Polish source exposure, Borel rank schedule exposure, and Borel-function handoff obligations force the route through P, B, F before a rank value is read.

Apply the Real readback nonescape obligation. It places every real-facing rank use behind the finite stream window S, regular rational readback Q, and real seal E.

The rows H, C, G, N cannot create a second path. They preserve, replay, cite, and name the displayed packet, so the handoff remains the ordered route through the six source and readback rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireFunctionRank}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireFunctionRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6635.1, Definition 6635.2, and Definition 6635.3 seed the rank-read route over Polish source, Borel schedule, Borel-function handoff, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireFunctionRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Baire hierarchy completeness, selected Baire codes, quotient equality of functions, countable choice, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Polish source exposure, Borel rank schedule exposure, Borel-function handoff, Real readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BaireFunctionRankUp.

Constructive story: A $\mathsf{BaireFunctionRank}^{\uparrow}$ packet is generated from a PolishSpace row, BorelHierarchy rank schedule, BorelFunction handoff, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6636

A Concrete Naming Certificate for BanachContractionRealCompletion$^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive completion, the BanachContractionRealC
packet records the finite route by which contraction iterates become a completion-facing real seal. It sits after the fixed-point endpoint discipline of chapter 396, the complete-metric service of chapter 128, the Cauchy-modulus surface of chapter 292, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names contraction iteration as displayed RegSeqRat and Real completion data; it does not select a host fixed point, quotient completion representative, or ambient completeness principle.

Definition 6636.1 (Banach contraction real-completion carrier)**.** *A* BanachContractionRealCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, F, x_0, I, D, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* CompleteMetric$^{\uparrow}$ *source row, carrying only the displayed metric and completion-facing Cauchy service available to the contraction route.*
2. *F is the contraction self-map row, including the graph read and contraction-ratio ledger used to compare successive iterates.*
3. *x_0 is the initial-point row.*
4. *I is the finite iterate ledger naming the inspected Picard stages $x_0, x_1, \ldots, x_n$.*
5. *D is the adjacent-distance ledger, recording the contraction-controlled distance comparisons extracted from I and F.*
6. *M is the* CauchyModulus$^{\uparrow}$ *row generated from D.*
7. *W is the finite observation-window row used when a completion or real consumer asks for a displayed precision read.*
8. *R is the* RegSeqRat$^{\uparrow}$ *handoff row built from M and W.*
9. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after the contraction, modulus, window, and regular-rational rows have been displayed.*
10. *H records componentwise* hsame *transport for $X, F, x_0, I, D, M, W, R, E, C, P, N$.*

11. *C records displayed* Cont *replay from a contraction-iteration request through the metric, modulus, regular-rational, and real-seal rows.*
12. *P is the* Pkg *provenance over* $\mathsf{BanachContractionRealCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachContractionRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a selected fixed point, quotient real equality, host function extensionality, countable choice over iterates, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6636.2 (Banach contraction real-completion obligations)**.** *For every accepted* BanachContractionRealCom *carrier $B = (X, F, x_0, I, D, M, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachContractionRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, complete-metric source exposure, contraction self-map stability, initial-point visibility, finite iterate-ledger visibility, adjacent-distance exactness, Cauchy-modulus extraction, observation-window locality,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6636.1. The rows X, F, x_0, I, D expose the complete metric source, the contraction graph, the initial point, the inspected iterates, and the adjacent-distance comparisons before any completion-facing row is read.

The Cauchy data are downstream. The row M is generated from the displayed distance ledger, and the row W names the finite windows used by each requested precision read.

The real-completion boundary is terminal. The row R is the only regular rational handoff, and E is the only Real seal. The structural rows H, C, P, N preserve and package the same route, so the certificate cannot export a selected fixed point, quotient representative, countable-choice schedule, or host completeness theorem. □

Theorem 6636.3 (Banach contraction iteration Cauchy route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BanachContractionRealCompletion}^{\uparrow}$ *carrier must read the contraction iteration through*

$$X, F, x_0, I, D, M, W, R, E, H, C, P, N.$$

The adjacent-distance ledger D produces the $\mathsf{CauchyModulus}^{\uparrow}$ *row M, the finite windows W expose the requested precision reads, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R gives the regular rational handoff, and the* $\mathsf{Real}^{\uparrow}$ *row E seals the completion-facing real datum. No public read bypasses this order through host completeness or a chosen fixed-point object.*

Proof. Start with a completion-facing read. The carrier admits the candidate iteration only through the complete metric row X, contraction row F, initial point x_0, and finite iterate ledger I.

The Cauchy route is forced by the distance ledger. The row D records the contraction-controlled adjacent distances, and M is the only Cauchy-modulus row available to the consumer.

The regular and real rows are then read in order. The window row W fixes the inspected precision budget, R translates that budget into regular rational data, and E seals the resulting real-facing read. Transport, replay, provenance, and local naming preserve the same route, leaving no coordinate for a host fixed point, selected limit, quotient equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{BanachContractionRealCompletion}^{\uparrow}$ real-completion read factors through a complete metric source, contraction self-map, initial point, finite iterate ledger,

adjacent-distance ledger, CauchyModulus row, finite observation windows, RegSeqRat handoff, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: banachfixedpoint, completemetric, cauchymodulus, regseqrat, real: none is bijective with $\mathsf{BanachContractionRealCompletion}^{\uparrow}$ because the packet jointly stores contraction iteration, adjacent-distance control, Cauchy-modulus extraction, finite windows, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachContractionRealCompletion}^{\uparrow}$ packet is generated from a CompleteMetricUp source, contraction self-map, initial point, finite iterate ledger, adjacent-distance ledger, CauchyModulus route, finite observation windows, RegSeqRat handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContractionRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6636.1, Definition 6636.2, and Definition 6636.3 seed the finite contraction-iteration route from complete metric data to RegSeqRat readback and Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachContractionRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, quotient completion representatives, host function extensionality, countable choice over iterates, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, adjacent-distance exactness, Cauchy-modulus extraction, window locality, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BanachContractionRealCompletionUp.

Chapter 6637

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, effective Cauchy completeness is read through displayed Cauchy names, explicit moduli, and finite rational approximation ledgers rather than through quotient classes or a host completeness theorem. The $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ packet records that traditional route as NameCert data over the dyadic core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the displayed sequence-completeness boundary of chapter 1515. It gives Real and Completion consumers a named effective-completeness landing surface without asserting that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 6637.1 (Effective Cauchy completeness carrier)**.** *An* $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, R, M, L, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger used for precision requests;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the Cauchy name being completed;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of the same windows;*
4. *M is the effective modulus row assigning each dyadic tolerance a displayed late threshold;*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row offered as the candidate landing name;*
6. *W is the witness ledger saying that each sufficiently late regular-rational window is within the dyadic tolerance of the landing seal;*
7. *H records componentwise* hsame *transport for $D, S, R, M, L, W, C, P, N$;*
8. *C records the displayed* Cont *replay from dyadic tolerance requests through stream windows, regular readback, modulus thresholds, witness rows, and the Real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompleteness}^{\uparrow}}$ *row.*

The carrier contains no quotient Cauchy class, selected representative, countable-choice schedule, open-cover compactness principle, ambient metric-completeness premise, or proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Theorem 6637.2 (Effective Cauchy completeness NameCert obligations)**.** *For an accepted* EffectiveCauchyComplet *carrier* $E = (D, S, R, M, L, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dyadic, stream, regular-rational, modulus, landing, witness, transport, replay, provenance, and naming rows; classifier stability under componentwise* hsame *transport; modulus coverage for every dyadic tolerance request; witness exactness tying the late regular-rational windows to the terminal* $\mathsf{Real}^{\uparrow}$ *seal; and ledger non-escape for quotient completion, representative selection, countable choice, open-cover compactness, ambient completeness, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6637.1. The displayed tuple exposes the dyadic tolerance row, finite windows, regular rational readback, effective modulus, landing seal, witness ledger, transport, replay, provenance, and local name.

Classifier stability is componentwise. A comparison of accepted packets first transports $D, S, R, M, L, W, C, P, N$ by H before any local read is identified.

Modulus coverage and witness exactness are read in order. A dyadic tolerance request reaches M for a late threshold, then reaches S and R for the inspected windows, and finally reaches W for the comparison with L.

The rows of E are exhausted by this route. Since none of them stores a quotient class, selected representative, choice schedule, open-cover principle, ambient complete carrier, or completeness theorem for $\mathsf{Real}^{\uparrow}$, the non-escape clause follows. □

Theorem 6637.3 (Effective Cauchy completeness modulus handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ *packet* $E = (D, S, R, M, L, W, H, C, P, N)$ *must read the dyadic tolerance ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the effective modulus row* M *before it can consume the landing seal* L*. The handoff to* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is therefore the finite route*

$$D, S, R, M \longmapsto W, L, H, C, P, N.$$

It is not a quotient completion, selected limit, countable-choice construction, open-cover compactness argument, ambient completeness assumption, or proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with a completion-facing tolerance request. The dyadic row D fixes the requested precision, so the consumer cannot inspect a landing seal before the precision row is displayed.

Read the Cauchy-name rows next. The finite windows S and the regular rational readback R expose the observed tail of the name, while M supplies the late threshold for that tolerance.

Only after this route is displayed can W compare the late windows with L. The structural rows H, C, P, N transport, replay, source, and name the same finite handoff for $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

No further row exists in the carrier. Hence the handoff cannot provide quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6637.4 (Effective Cauchy completeness ledger nonescape)**.** *An accepted* $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ *carrier* $E = (D, S, R, M, L, W, H, C, P, N)$ *exports only the displayed effective Cauchy route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,*

Pkg, *and* NameCert. *Any consumer asking for quotient completion, a selected representative, countable choice, open-cover compactness, ambient metric completeness, or completeness of* $\mathsf{Real}^{\uparrow}$ *is returned to those finite rows.*

Proof. Project the accepted carrier. Its coordinates are precisely the dyadic tolerance ledger, stream windows, regular readback, modulus, landing seal, witness ledger, transport, replay, provenance, and local naming row.

Apply the NameCert obligation surface. It makes every effective-completeness read pass through the displayed modulus and witness rows before the terminal Real seal is visible.

Apply the modulus handoff. The completion-facing consumer receives a finite route to the displayed seal, not a second completed carrier. Since the forbidden objects are absent from the tuple, they cannot be exported by the local ledger. □

Independence witness: dyadicratcore, streamname, regseqrat, real, sequentiallycompletemetric: $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite stream windows, regular rational readback, an effective modulus, a terminal Real seal, a witness ledger, transport, replay, provenance, and local naming in one completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompleteness}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchyCompleteness}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite stream windows, regular rational readback, an effective modulus, a terminal Real seal, witness ledgers, componentwise hsame transports, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6637.1, Definition 6637.2, Definition 6637.3, and Definition 6637.4 seed the finite effective Cauchy completeness route over displayed dyadic, stream, regular rational, Real-seal, and sequential-completeness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, representative selection, countable choice, open-cover compactness, ambient metric completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed effective Cauchy route.

Chapter 6638

A Concrete Naming Certificate for $\mathsf{PettisIntegral}^{\uparrow}$

Following Pettis and Dunford–Schwartz on weak scalar integration in locally convex and Banach-space settings, the $\mathsf{PettisIntegral}^{\uparrow}$ packet records the finite BEDC surface on which weak scalar integrals are named before any Bochner norm integrability theorem, selected vector-valued representative, quotient almost-everywhere equality, weak compactness extraction, or host dual-space completeness principle is public. It sits beside the weak-compactness handoff of chapter 6346, the measure and integral sources of chapter 92 and chapter 123, the functional-analysis carrier of chapter 94, and the bounded-operator surface of chapter 1644.

Definition 6638.1 (Pettis integral carrier)**.** *A* $\mathsf{PettisIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, I, F, B, S, R, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row naming measurable domains, measurable sets, null windows, and finite partitions visible to the packet.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *scalar readback row for finite scalar integrals and displayed domination budgets.*
3. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row naming the locally convex or Banach carrier, weak topology probes, and finite dual-pairing tests.*
4. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for bounded linear functionals and bounded linear probes through which scalar shadows are read.*
5. *S is the weak scalar measurability row. It records only that every displayed bounded probe reads a scalar measurable function through M, I, F, B.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-budget row storing tolerances, comparison bounds, and finite integral windows for the scalar shadows.*
7. *W is the weak scalar handoff row. It records the finite route from scalar probe integrals to the named vector-valued Pettis integral request.*
8. *H records componentwise* hsame *transport for M, I, F, B, S, R, W and the local structural rows.*
9. *C records displayed* Cont *replay from weak scalar integration requests through $M, I, F, B, S, R, W, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{PettisIntegral}^{\uparrow}$, $\mathsf{DunfordPettis}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PettisIntegral}^{\uparrow}}$ *row.*

The carrier has no coordinate for Bochner norm integrability, a selected vector-valued representative, quotient almost-everywhere equality, ambient dual-ball compactness, countable choice, or a weakly compact subsequence extraction.

Theorem 6638.2 (Pettis integral NameCert obligations)**.** *For every accepted* $\mathsf{PettisIntegral}^{\uparrow}$ *carrier*

$$Q = (M, I, F, B, S, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PettisIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, measure source compatibility through M, scalar integral readback through I, functional-analysis probe visibility through F, bounded-linear-probe visibility through B, weak scalar measurability through S, scalar-budget visibility through R, weak scalar integral handoff through W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every weak-integration-facing consumer must read the same rows before using the handoff. It receives no Bochner integral, no selected vector representative, no quotient almost-everywhere equality, no ambient dual compactness theorem, no countable-choice schedule, and no weakly compact subsequence.

Proof. Carrier admission is projection from Definition 6638.1. The accepted packet displays exactly $M, I, F, B, S, R, W, H, C, P, N$.

The analytic rows M and I fix the measurable source and scalar integral readback. The weak-facing rows F and B restrict every observation to the displayed dual-pairing and bounded-linear-probe tests, while S records scalar measurability only through those tests.

The scalar-budget row R supplies the finite comparison data used by the scalar shadows. The handoff row W is downstream of those scalar shadows and names the Pettis integral request without selecting a vector representative or importing a Bochner theorem.

The structural rows H, C, P, N preserve the same packet under transport, replay, provenance, and naming. Since every excluded object has no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6638.3 (Pettis integral weak scalar handoff)**.** *Every weak scalar integration consumer of an accepted* $\mathsf{PettisIntegral}^{\uparrow}$ *carrier reaches the Pettis handoff only through the displayed route*

$$M, I \longmapsto F, B, S, R \longmapsto W \longmapsto H, C, P, N.$$

Thus the consumer may read measure data, scalar integral evidence, finite weak-probe and bounded-linear-probe rows, scalar measurability, scalar budgets, the weak scalar integral handoff, transport, replay, provenance, and local naming, and it may not read Bochner norm integrability, selected vector-valued representatives, quotient almost-everywhere equality, ambient dual compactness, countable choice, or weakly compact subsequences.

Proof. Begin with a weak scalar integration read. By Definition 6638.1, the measurable and scalar integral sources are exactly M and I.

The weak scalar tests then pass through F, B, S, R. These rows expose only finite dual-pairing probes, bounded linear functionals, scalar measurability evidence, and real-valued comparison budgets.

The consumer reaches the Pettis integral request only at W. That row is a handoff from the displayed scalar shadows to the named weak integral, not an extraction of a Bochner integral or a selected vector representative from an ambient space.

Finally H, C, P, N transport, replay, package, and name the same route. Because the carrier lacks coordinates for the refused objects, no consumer can recover them from the $\mathsf{PettisIntegral}^{\uparrow}$ packet. □

Falsifiable prediction: Any $\mathsf{PettisIntegral}^{\uparrow}$ read that claims weak scalar integration while bypassing the displayed $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 6638.1.

Independence witness: Dunford-Pettis, measure, integral, functional-analysis, bounded-linear-operator: $\mathsf{PettisIntegral}^{\uparrow}$ is not bijective to any listed sibling because it stores the finite weak scalar integration handoff over scalar probe integrals without storing Bochner norm integrability, selected vector representatives, quotient almost-everywhere equality, ambient dual compactness, countable choice, or weakly compact subsequences.

Closure status for $\mathsf{NameCert}_{\mathsf{PettisIntegral}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PettisIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6638.1, Definition 6638.2, and Definition 6638.3 seed the finite weak scalar integration route over measure, integral, weak-probe, bounded-probe, scalar-measurability, scalar-budget, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PettisIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Bochner norm integrability, selected vector-valued representatives, quotient almost-everywhere equality, ambient dual compactness, countable choice, weakly compact subsequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, measure source compatibility, scalar integral readback, weak-probe visibility, bounded-linear-probe visibility, scalar measurability, scalar-budget visibility, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PettisIntegralUp.

Constructive story: A $\mathsf{PettisIntegral}^{\uparrow}$ packet is generated from Measure, Integral, FunctionalAnalysis, BoundedLinearOperator, Real scalar-budget, weak scalar measurability, weak scalar integration handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6639

A Concrete Naming Certificate for $\mathsf{LebesgueNumberFunction}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive analysis, the $\mathsf{LebesgueNumberFunction}^{\uparrow}$ packet records the finite cover-to-radius ledger used when a compact metric source supplies displayed cover rows and downstream uniform-continuity consumers need explicit positive radii. It sits after the compact metric source of chapter 129, the finite-cover Lebesgue-number surface of chapter 3606, the Lebesgue-number lemma route of chapter 2301, the compact-uniform modulus extractor of chapter 2202, the rational positivity row of chapter 37, and the dyadic tolerance core of chapter 291. The packet names a finite function ledger assigning radii to displayed cover entries; it does not assert open-cover compactness, choose a cover refinement from an ambient theorem, or hide a modulus inside host analysis.

Definition 6639.1 (LebesgueNumberFunction carrier)**.** *A* $\mathsf{LebesgueNumberFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, U, C, L, R, D, W, H, T, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *and carrying the finite compactness data available to the packet;*
2. *U is the finite cover row, listing the displayed cover entries to which radii may be assigned;*
3. *C is the compact finite-cover and Lebesgue-number row, routing the cover through the finite-cover Lebesgue-number and Lebesgue-number lemma surfaces;*
4. *L is the function ledger assigning a displayed positive radius to each cover entry that has been admitted by U and C;*
5. *R is the rational tolerance row, recording positivity and comparison data for each ledger radius through* $\mathsf{Rat}^{\uparrow}$*;*
6. *D is the dyadic tolerance row, recording endpoint and refinement budgets through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
7. *W is the finite witness-window row, recording the compact windows on which the radius assignment is checked;*
8. *H stores componentwise* hsame *transports for K, U, C, L, R, D, W and for the structural rows;*
9. *T stores displayed transport and replay routes for cover entries, radius reads, and compact-window witnesses;*

10. *P is the* Pkg *provenance over the compact metric source, finite cover row, Lebesgue-number rows, rational and dyadic tolerances, witness windows, transports, and local naming evidence;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberFunction}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It has no coordinate for an arbitrary open-cover compactness principle, a selected refinement outside the displayed cover, a host real-valued radius function, countable choice, or an ambient uniform-continuity theorem.

Theorem 6639.2 (LebesgueNumberFunction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberFunction}^{\uparrow}}$ *surface consists of carrier exposure for $K, U, C, L, R, D, W, H, T, P, N$, finite cover admission for every entry read by L, positive-radius ledger stability through R and D, witness-window replay through W and T, componentwise transport through H, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6639.1. The only analytic rows are the compact metric source, finite cover, compact finite-cover route, radius ledger, rational and dyadic tolerance rows, and witness windows.

Read the cover and radius obligations together. A ledger entry in L is accepted only for a cover entry already displayed by U and certified through C, while R and D record the positivity and tolerance data for the assigned radius.

The remaining rows bind the ledger to BEDC structure. Witness windows W are replayed by T, transported by H, recorded by P, and named by N, so the NameCert surface has no hidden radius function or ambient compactness proof. □

Theorem 6639.3 (LebesgueNumberFunction finite-cover handoff)**.** *Every accepted* $\mathsf{LebesgueNumberFunction}^{\uparrow}$ *packet hands a finite cover row from U through the compact finite-cover and Lebesgue-number row C before any consumer reads a radius assignment in L. The handoff preserves the same compact metric source K, rational tolerance row R, dyadic tolerance row D, witness-window row W, transport rows H, T, provenance row P, and local name N.*

Proof. Begin with the compact metric source and finite cover rows K and U. They fix the source and the displayed cover entries before the packet exposes any radius.

Use the compact finite-cover and Lebesgue-number row C. It is the only route from the cover entries to the positive-radius ledger, so a consumer cannot read L before the finite-cover handoff has occurred.

Finally preserve the structural rows. The rational and dyadic tolerance rows, finite witness windows, transports, replay routes, provenance, and local name are carried with the same cover entry, giving a handoff rather than a detached radius table. □

Theorem 6639.4 (LebesgueNumberFunction uniform-modulus boundary)**.** *A* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ *or uniform-continuity consumer may read an accepted* $\mathsf{LebesgueNumberFunction}^{\uparrow}$ *packet only as the finite ledger $K, U, C, L, R, D, W, H, T, P, N$. It receives the displayed cover-to-radius assignments and witness windows, but receives no open-cover compactness principle, no selected host modulus, no quotient real equality, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the finite-cover handoff first. It supplies the compact source, displayed cover entries, Lebesgue-number route, radius ledger, tolerance rows, and witness windows as one finite packet.

A uniform-modulus consumer may reuse those rows, but only through the transports, replay routes, provenance, and local naming data. Thus the consumer receives a named finite ledger, not a function chosen from outside the packet.

The excluded principles require coordinates absent from the carrier: open-cover compactness, host modulus selection, quotient real equality, countable choice, or ambient Real completeness. Since none of these coordinates appears in F, they remain outside the uniform-modulus boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueNumberFunction}^{\uparrow}$ packet is generated from a compact metric source, a displayed finite cover row, compact finite-cover and Lebesgue-number rows, a positive-radius ledger, rational and dyadic tolerance rows, finite witness windows, componentwise transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6639.1, Definition 6639.2, Definition 6639.3, and Definition 6639.4 describe the finite cover-to-radius ledger and its consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, selected host moduli, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier and NameCert obligation rows for the finite cover-to-radius ledger.

Chapter 6640

A Concrete Naming Certificate for $\mathsf{LocatedRealNumber}^{\uparrow}$

Following Bishop and Bridges, a located real number is a real name whose finite rational observations also carry a displayed locatedness test. The $\mathsf{LocatedRealNumber}^{\uparrow}$ packet sits between the constructive real-line interface of chapter 4041, the terminal real-name seal of chapter 38, the regular rational source of chapter 301, the stream-window schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. It records a Bishop-style located name as finite BEDC data, not as a completed host real, quotient stream, trichotomy principle, selected comparison oracle, or ambient completeness theorem.

Definition 6640.1 (Located real number carrier)**.** *A* $\mathsf{LocatedRealNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, E, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for inspected precision requests, rational intervals, and locatedness questions.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row read through S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and interval ledger used to compare requested radii, endpoint gaps, and rational enclosure witnesses.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after S, R, D are visible.*
5. *A is the* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *interface row recording the Bishop locatedness surface available to downstream real-line consumers.*
6. *H records componentwise* hsame *transport for the stream, regular-rational, dyadic, real-seal, constructive-real-line, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a locatedness request through S, R, D, E, A, H.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealNumber}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ConstructiveRealLine}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, E, A, H, C, P, N$. It has no coordinate for quotient equality of real names, trichotomy, a selected sign decision, a host locatedness predicate, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6640.2 (Located real number NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealNumber}^{\uparrow}$ *carrier* $L = (S, R, D, E, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealNumber}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window visibility, regular-rational readback, dyadic tolerance and enclosure exactness, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, constructive-real-line locatedness exposure, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6640.1. The rows $S, R, D, E, A, H, C, P, N$ are the full inventory of data available to an accepted packet.

The real-name route is finite. A consumer first reads the stream schedule S, then the regular rational row R, then the dyadic tolerance and enclosure ledger D, and only then may consume the terminal real seal E.

The locatedness route is the row A. It records the constructive real-line interface attached to the same regular rational and dyadic data, so a locatedness request cannot bypass the displayed window and tolerance rows.

Finally H, C, P, N transport, replay, source, and name that same finite packet. Since no row contains quotient equality of real names, trichotomy, a selected sign decision, host locatedness, or ambient completeness, the listed rows exhaust the local certificate obligations. □

Theorem 6640.3 (Located real number RegSeqRat handoff)**.** *Every* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{LocatedRealNumber}^{\uparrow}$ *carrier must read the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows retained before the located real-number claim is public. The exported datum is a finite Bishop-style located real name and not a quotient real, selected comparison function, trichotomy principle, host completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from Definition 6640.2. It makes stream-window visibility, regular-rational readback, dyadic tolerance exactness, and Real-seal nonescape mandatory rows of every accepted local certificate.

The $\mathsf{ConstructiveRealLine}^{\uparrow}$ row A is downstream of the same displayed route. It can expose locatedness only after the requested finite window, rational name row, dyadic enclosure ledger, and terminal seal have been shown.

Transport and replay do not add analytic content. The rows H, C, P, N preserve the same packet under hsame, Cont, Pkg, and NameCert, leaving no coordinate for quotient equality, trichotomy, selected comparison, host completion, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRealNumber}^{\uparrow}$ read exposes a StreamName window, RegSeqRat regular rational name, DyadicRatCore tolerance and enclosure ledger, terminal RealUp seal, ConstructiveRealLine locatedness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any consumer reads located real-number data.

Independence witness: dyadicratcore, streamname, regseqrat, real, constructive-real-line: no listed sibling is bijective with $\mathsf{LocatedRealNumber}^{\uparrow}$ because this packet jointly binds finite observation windows, regular rational readback, dyadic locatedness ledgers, a Real seal, constructive real-line exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealNumber}^{\uparrow}}$**.**

Constructive story: A LocatedRealNumber$^{\uparrow}$ packet is generated from StreamName windows, a RegSeqRat regular rational row, DyadicRatCore tolerance and enclosure data, a terminal RealUp seal, a ConstructiveRealLine locatedness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocatedRealNumber}^{\uparrow}}$, seed)

Scope closed: Definition 6640.1, Definition 6640.2, and Definition 6640.3 seed the finite Bishop-style located real-name route over DyadicRatCore, StreamName, RegSeqRat, Real, ConstructiveRealLine, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedRealNumber}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, trichotomy, selected sign decisions, host locatedness predicates, ambient Real completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window visibility, regular-rational read-back, dyadic tolerance exactness, Real seal nonescape, constructive-real-line exposure, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealNumberUp.

Chapter 6641

A Concrete Naming Certificate for $\mathsf{CompactUniformLimit}^{\uparrow}$

Following Bishop and Bridges on compact-source uniform convergence, the $\mathsf{CompactUniformLimit}^{\uparrow}$ packet realizes the compact-uniform-continuity program of chapter 9237 as a finite NameCert carrier. It sits over the compact uniform-modulus row of chapter 2299, the continuous-map interface of chapter 130, the theorem handoff of chapter 2200, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names compact-source uniform limits through displayed finite moduli and readback rows; it does not import open-cover compactness, quotient equality of functions, selected host limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6641.1 (Compact uniform-limit carrier)**.** *A* $\mathsf{CompactUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, F, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactUniformModulus}^{\uparrow}$ *row, carrying the compact-source finite modulus used by every tolerance request.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed family or limiting map whose graph data are visible to the packet.*
3. *T is the* $\mathsf{UniformLimitTheorem}^{\uparrow}$ *handoff row, recording the finite continuity inheritance route available after uniform convergence is displayed.*
4. *W is the finite observation-window row for compact-source points, function stages, modulus requests, and limit-value reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational comparison windows generated from M, F, T, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the compact modulus, continuous-map, theorem, window, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transport for the compact-modulus, continuous-map, theorem, window, readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source uniform-limit requests through M, F, T, W, R, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformLimit}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimitTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformLimit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, F, T, W, R, E, H, C, P, N$. It has no coordinate for an open-cover compactness theorem, quotient function equality, selected limit function, host continuity theorem, or ambient Real completeness.

Theorem 6641.2 (Compact uniform-limit NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformLimit}^{\uparrow}$ *carrier $U = (M, F, T, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformLimit}^{\uparrow}}$ *surface consists exactly of carrier admission, compact uniform-modulus exposure, continuous-map graph visibility, uniform-limit theorem handoff, finite observation-window locality, regular-rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6641.1. The accepted packet exposes exactly the compact modulus, continuous-map, theorem handoff, finite windows, regular-rational reads, real seal, transport, replay, provenance, and local naming rows.

The compact part is M. A tolerance request over the compact source becomes public only as a finite modulus row and is read together with the graph data in F and the inspected windows in W.

The continuity handoff is T. It can be consumed only after the compact modulus and continuous-map rows have been displayed, and its Real-facing values pass through R before the terminal seal E is read.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores open-cover compactness, quotient function equality, a selected host limit, host continuity, or ambient Real completeness, the listed obligations exhaust the local certificate surface. □

Theorem 6641.3 (Compact uniform-limit continuity handoff)**.** *Every continuity read exported by an accepted* $\mathsf{CompactUniformLimit}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactUniformModulus}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformLimitTheorem}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows retained. The handoff exports compact-source uniform convergence as finite modulus, continuous-map, regular-rational readback, and Real-seal data, not as a host compactness theorem, quotient function equality, selected limit, or ambient completeness principle.*

Proof. Begin with Definition 6641.2. It forces every accepted read to expose the compact uniform modulus M, the continuous-map row F, the theorem handoff T, finite windows W, regular-rational readback R, and terminal real seal E.

The continuity read is therefore downstream of the compact modulus and the graph row. The theorem handoff T may state the continuity inheritance route, but only as a displayed finite row inside the same packet.

The Real-facing values are sealed last. The regular-rational row R supplies the comparison windows before E is consumed, while H, C, P, N preserve the route under transport, replay, provenance, and local naming. No excluded host compactness, quotient equality, selected limit, or ambient completeness coordinate is available. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformLimit}^{\uparrow}$ read exposes a CompactUniformModulus row, ContinuousMap row, UniformLimitTheorem handoff, finite observation windows,

RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any consumer reads compact uniform-limit continuity.

Independence witness: compact-uniform-modulus, continuousmap, uniform-limit-theorem, regseqrat, real: no listed sibling is bijective with $\mathsf{CompactUniformLimit}^{\uparrow}$ because this packet jointly binds compact-source finite modulus data, continuous-map graph rows, uniform-limit theorem handoff, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformLimit}^{\uparrow}$ packet is generated from a CompactUniformModulus row, ContinuousMap graph data, UniformLimitTheorem handoff, finite observation windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6641.1, Definition 6641.2, and Definition 6641.3 seed the compact-source uniform-limit route over CompactUniformModulus, ContinuousMap, UniformLimitTheorem, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality of functions, selected host limits, host continuity theorems, ambient Real completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact uniform-modulus exposure, continuous-map graph visibility, uniform-limit theorem handoff, window locality, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformLimitUp.

Chapter 6642

A Concrete Naming Certificate for $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, a located Cauchy limit is read through explicit finite moduli, locatedness tests, and rational windows rather than through quotient streams, countable choice, or an ambient completion theorem. The $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ packet records that route as one NameCert carrier over the dyadic tolerances of chapter 291, the finite windows of chapter 126, the regular rational readback of chapter 301, the modulus surface of chapter 292, the located-real handoff of **??**, and the terminal seal of chapter 38.

Definition 6642.1 (Located Cauchy modulus selector carrier)**.** *A* $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, Q, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the positive dyadic radius being requested;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row scheduled for that tolerance;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows;*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row selecting the late threshold for the same tolerance;*
5. *L is the locatedness row deciding the displayed overlap or separation test at the selected threshold;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback, modulus, and locatedness rows are displayed;*
7. *H records componentwise* hsame *transport for $D, W, Q, M, L, E, C, P, N$;*
8. *C records displayed* Cont *replay from dyadic tolerance to stream window, from window to regular readback, from modulus and locatedness rows to the Real seal, and from the structural rows to the local name;*
9. *P records* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-stream equality, a selected host limit, countable choice, an unlocated Cauchy witness, an ambient completion theorem, or a Real seal read that bypasses the displayed modulus and locatedness rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyModulusSelectorUp.LocatedCauchyModulusSelectorTasteGate`
`single carrier alignment`

Theorem 6642.2 (Located Cauchy modulus selector NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ *carrier* $S = (D, W, Q, M, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusSel}}$ *surface consists exactly of carrier admission for the displayed dyadic tolerance, stream-window, regular readback, modulus, locatedness, Real seal, transport, replay, provenance, and naming rows; classifier stability under componentwise* hsame *transport; modulus coverage for every displayed dyadic tolerance; locatedness exactness at the selected finite threshold; and non-escape for quotient streams, host limits, countable choice, unlocated witnesses, ambient completion, and seal reads outside the displayed packet.*

Proof. Carrier admission is projection from Definition 6642.1. The packet exposes only the rows $D, W, Q, M, L, E, H, C, P, N$, so every local read must begin with the displayed dyadic tolerance, window, readback, modulus, locatedness, and seal data.

The constructive selector route is finite. A dyadic tolerance request is recorded in D, scheduled as the finite window W, read back as Q, and bounded by the selected modulus threshold M. The locatedness row L is read at the same threshold before the seal row E can be offered to a Real-facing consumer.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. Since the carrier contains none of the excluded coordinates, no quotient stream, host limit, choice schedule, unlocated witness, ambient completion theorem, or bypass seal can be exported by the local certificate. □

Theorem 6642.3 (Located Cauchy modulus selector Real seal handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ *packet must read the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* M*, and the locatedness row* L *before it can consume the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The handoff route is therefore*

$$D, W, Q, M, L \longmapsto E, H, C, P, N.$$

It is not a quotient-stream construction, selected host limit, countable-choice route, unlocated Cauchy witness, or ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedCauchyModulusSelectorUp.LocatedCauchyModulusSelectorRealSealHandoff`

Proof. Start with the NameCert obligations of Definition 6642.2. They force the Real-facing read to pass through the displayed carrier rows rather than through an external completion object.

The finite analytic route then orders the rows. The dyadic tolerance D determines the requested scale, the stream window W and regular readback Q expose the inspected data, the modulus row M supplies the late threshold, and the locatedness row L decides the local located test at that threshold.

Only after those rows are displayed can E serve as the Real seal. The transport, replay, provenance, and local naming rows H, C, P, N preserve the same route, so any consumer omitting W, Q, M, or L is outside the accepted packet. □

Theorem 6642.4 (Located Cauchy modulus selector threshold induction)**.** *Every accepted tolerance request in a* $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ *packet is discharged by finite induction over the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* M*. The base case opens the first admissible* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and the step case extends the window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *until the selected threshold is reached. The locatedness row* L *is then read at that same threshold before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The derivation uses only the carrier of Definition 6642.1, the NameCert obligations of Definition 6642.2, and the Real seal handoff of Definition 6642.3; it does not use quotient streams, selected host limits, countable choice, unlocated Cauchy witnesses, or ambient completion.*

Proof. Start with the carrier row M. Since M is part of the finite packet, its selected threshold is inspected as a displayed row rather than as an external choice function.

Induct over the finite threshold named by M. In the base case, the first admissible stream window W is opened for the requested dyadic tolerance D, and the regular rational readback Q is read from that window.

In the step case, extend the displayed window by one scheduled read and extend Q by the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ readback. The induction hypothesis supplies the previous finite window and readback route, and the NameCert obligations keep the extension inside the same carrier rows.

At the selected threshold, read L before exposing E. The Real seal handoff then forces every Real-facing consumer to pass through W, Q, M, and L, so the construction has no coordinate for quotient streams, selected host limits, countable choice, unlocated Cauchy witnesses, or ambient completion. □

Independence witness: dyadicratcore, streamname, regseqrat, cauchymodulus, locatedreal, real: $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the dyadic tolerance request, finite window schedule, regular-rational readback, selected Cauchy modulus threshold, locatedness row, Real seal, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyModulusSelector}^{\uparrow}$ packet is generated from dyadic tolerance requests, finite stream windows, regular rational readback, selected Cauchy modulus thresholds, locatedness rows, Real seals, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6642.1, Definition 6642.2, and Definition 6642.3 seed the finite located Cauchy selector route over DyadicRatCore, StreamName, RegSeqRat, CauchyModulus, LocatedReal, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, unlocated Cauchy witnesses, ambient completion, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed located Cauchy modulus selector route.

Chapter 6643

A Concrete Naming Certificate for $\mathsf{CartesianClosedCategory}^{\uparrow}$

Following Mac Lane's account of categories with finite products and exponentials and the Lambek–Scott use of cartesian closed categories as categorical logic, the $\mathsf{CartesianClosedCategory}^{\uparrow}$ packet records the finite BEDC surface on which products, terminal objects, evaluation maps, and currying are named before any ambient category universe, host function space, quotient morphism equality, global internal language, topos classifier, or set-theoretic semantics is public. It sits between the category carrier of chapter 61, the functorial transport surface of chapter 62, the product packet of chapter 34, the tensor discipline of chapter 181, and the topos boundary of chapter 183.

Definition 6643.1 (Cartesian closed category carrier)**.** *A* $\mathsf{CartesianClosedCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, F, T, P, X, E, V, U, H, R, G, N)$$

with the following displayed rows:

1. *C is the accepted* $\mathsf{Cat}^{\uparrow}$ *row naming the object carrier, hom-carrier, identity, composition, associativity, and unit laws.*
2. *F is the* $\mathsf{Functor}^{\uparrow}$ *transport row for product and exponential reindexing along displayed morphism rows.*
3. *T is the terminal-object row, storing the displayed terminal object and its unique map rows through the classifier of C.*
4. *P is the binary-product row, read through* ***??*** *and carrying projections, pairing, product stability, and product classifier transport.*
5. *X is the exponential-object row, assigning to each displayed pair A, B an object B^A inside the same* $\mathsf{Cat}^{\uparrow}$ *carrier.*
6. *E is the evaluation row, storing each displayed evaluation morphism $B^A \times A \to B$ through the product row P and the hom-carrier row of C.*
7. *V is the currying row. It records the finite bijective read between maps $Z \times A \to B$ and maps $Z \to B^A$ as classifier-compatible hom rows, not as host function equality.*
8. *U is the product-exponential stability row, carrying componentwise stability of T, P, X, E, V under* hsame *transport in C.*

9. *H records componentwise* hsame *transport for C, F, T, P, X, E, V, U and the support rows.*
10. *R records displayed* Cont *replay for terminal, product, exponential, evaluation, currying, and stability reads.*
11. *G is the* Pkg *provenance over* $\mathsf{CartesianClosedCategory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, *??*, $\mathsf{MonoidalCat}^{\uparrow}$, $\mathsf{Topos}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CartesianClosedCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient category of all objects, host exponential functions, quotient morphism equality, global internal language, subobject classifier, set semantics, or topos structure.

Theorem 6643.2 (Cartesian closed category NameCert obligations)**.** *For every accepted* $\mathsf{CartesianClosedCategory}^{\uparrow}$ *carrier*

$$K = (C, F, T, P, X, E, V, U, H, R, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{CartesianClosedCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, category source compatibility through C, functorial transport through F, terminal-object exactness through T, product exactness through P, exponential-object admission through X, evaluation readback through E, currying readback through V, product-exponential stability through U, componentwise transport through H, displayed replay through R, package provenance through G, and local naming through N.*

Every cartesian-closed-category consumer must read the same rows before using evaluation or currying. It receives no ambient object universe, no host function-space equality, no quotient morphism equality, no global internal language, no subobject classifier, no set-theoretic semantics, and no $\mathsf{Topos}^{\uparrow}$ *structure.*

Proof. Project the accepted carrier. The packet displays exactly $C, F, T, P, X, E, V, U, H, R, G, N$, so carrier admission supplies only those rows.

Read the categorical source first. The row C gives the visible object and morphism surface, while F transports the product and exponential rows along displayed morphisms without adding a second category carrier.

Read the cartesian structure next. The terminal row T and product row P provide the finite cartesian spine; the exponential row X is attached to the same source, not to an ambient universe of objects.

The evaluation and currying rows are downstream of that spine. The row E reads $B^A \times A \to B$ through P and the hom-carrier of C, and V records the classifier-compatible transpose route. The stability and support rows U, H, R, G, N preserve the same finite packet under transport, replay, provenance, and local naming. Since the carrier has no coordinate for the refused objects, the listed rows exhaust the local obligations. □

Theorem 6643.3 (Cartesian closed category evaluation-currying stability)**.** *Let $K = (C, F, T, P, X, E, V, U, H, R, G$ be an accepted* $\mathsf{CartesianClosedCategory}^{\uparrow}$ *carrier. Any componentwise* hsame *transport recorded by H preserves the product row P, exponential row X, evaluation row E, and currying row V before a downstream consumer may use a transpose. Thus the evaluation-currying route*

$$C, F, T, P, X \longmapsto E, V \longmapsto U, H, R, G, N$$

is stable under the displayed classifier of the carrier and cannot be replaced by host function equality, quotient morphism equality, or an external internal-language semantics.

Proof. Project the product and exponential rows from Definition 6643.1. The rows P and X are displayed over the same $\mathsf{Cat}^{\uparrow}$ row C and transported only by the functorial and classifier rows F and H.

Read evaluation before currying. The evaluation row E uses the product row P to type $B^A \times A$ and the hom-carrier row of C to type the morphism into B.

Read the currying row V next. It records the finite transpose route between displayed hom rows and is guarded by the stability row U, so componentwise transport must preserve the source product, target exponential, evaluation morphism, and transposed morphism together.

Finally replay the support rows. The Cont route R, provenance G, and local naming row N carry the same transported packet. Because no row stores host functions, quotient morphism classes, or external semantics, the stable route is exactly the displayed evaluation-currying surface. □

Falsifiable prediction: Any accepted $\mathsf{CartesianClosedCategory}^{\uparrow}$ read that uses evaluation or currying while bypassing the displayed $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, **??**, terminal-object, exponential-object, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 6643.1.

Independence witness: category, functor, product, monoidalcat, topos: $\mathsf{CartesianClosedCategory}^{\uparrow}$ is not bijective to any listed sibling because it binds terminal objects, binary products, exponentials, evaluation, currying, product-exponential stability, transport, replay, provenance, and local naming as one finite certificate while refusing subobject classifiers and ambient internal-language semantics.

Closure status for $\mathsf{NameCert}_{\mathsf{CartesianClosedCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CartesianClosedCategory}^{\uparrow}$ packet is generated from Category, Functor, Product, terminal-object, exponential-object, evaluation, currying, product-exponential stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CartesianClosedCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6643.1, Definition 6643.2, and Definition 6643.3 seed the finite cartesian closed category carrier and its evaluation-currying stability route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CartesianClosedCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an ambient category universe, host function-space equality, quotient morphism equality, global internal language, subobject classifier, set-theoretic semantics, $\mathsf{Topos}^{\uparrow}$ structure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, category source compatibility, functorial transport, terminal-object exactness, product exactness, exponential-object admission, evaluation readback, currying readback, stability, transport, replay, provenance, and local naming for BEDC.Derived.CartesianClosedCategoryUp.

Chapter 6644

A Concrete Naming Certificate for $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$

Following Bishop and Bridges constructive completion, the Real-completeness limit route of chapter 9238 is read through finite moduli, selected windows, regular rational readback, and seal rows rather than through quotient streams or a host completeness theorem. The $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$ packet records the diagonal-limit witness of Definition 9238.5 as one NameCert carrier whose rows are visible before any Real-facing consumer reads the completed witness. It consumes the witness-ledger sibling chapter 6279 and compares its extracted finite route with the completion extractor chapter 1725 before the diagonal-limit row is public.

Definition 6644.1 (Real completion witness transducer carrier)**.** *A* $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, W, D, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular Cauchy source row over* $\mathsf{RegSeqRat}^{\uparrow}$*;*
2. *M is the explicit modulus row assigning each requested dyadic tolerance a displayed late threshold;*
3. *W is the selected finite-window row over* $\mathsf{StreamName}^{\uparrow}$ *at that threshold;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same request;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row from the selected windows;*
6. *L is the diagonal-limit row naming the candidate produced from S, M, W, D, R;*
7. *U is the uniqueness row saying that any competing accepted witness with the same displayed convergence route is identified by the Real seal classifier;*
8. *H records componentwise* hsame *transport for $S, M, W, D, R, L, U, C, P, N$;*
9. *C records displayed* Cont *replay from the source and modulus rows through the selected windows, dyadic tolerance ledger, readback, diagonal-limit row, uniqueness row, and local name;*
10. *P records* Pkg *provenance over* $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. N is the local $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessTransducer}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient real equality, selected host limits, countable choice, ambient completion, global stream equality, or a uniqueness proof outside the displayed finite route.

Theorem 6644.2 (Real completion witness transducer NameCert obligations)**.** *For an accepted* $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$ *carrier* $T = (S, M, W, D, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWitn}}$ *surface consists exactly of source admission, explicit modulus exposure, selected finite-window exposure, dyadic tolerance ledger exposure, RegSeqRat readback, diagonal-limit row admission, uniqueness-row admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape for quotient real equality, selected host limits, countable choice, ambient completion, global stream equality, and undisplayed uniqueness.*

Proof. Carrier admission is projection from Definition 6644.1. The packet exposes the regular Cauchy source, explicit modulus, selected windows, dyadic tolerance ledger, regular rational readback, diagonal-limit candidate, uniqueness row, and the structural rows H, C, P, N.

The limit witness route is constructive and finite. A tolerance request reaches the modulus row M, then the selected windows W and dyadic ledger D, then the readback row R, and only then the diagonal-limit row L.

The uniqueness row U is read through the same displayed route. Componentwise transport, route replay, provenance, and local naming preserve that inventory, while the carrier lacks quotient, choice, host-limit, ambient-completion, global-stream, and hidden-uniqueness coordinates. □

Theorem 6644.3 (Real completion witness transducer diagonal route)**.** *Every accepted* $\mathsf{RealCompletionWitnessTran}$ *packet realizes the diagonal-limit witness route of Definition 9238.5 through the finite spine*

$$S, M, W, D, R \longmapsto L, U, H, C, P, N.$$

A Real-facing consumer may read the completed witness only after the regular Cauchy source, explicit modulus, selected windows, dyadic tolerance ledger, and RegSeqRat readback rows have been displayed. The route does not export quotient real equality, a selected host limit, countable choice, ambient completion, global stream equality, or a uniqueness proof outside U.

Proof. Start with the Real-completeness target Definition 9238.5. In the transducer packet, the abstract diagonal candidate is represented by the displayed row L, reached only after S, M, W, D, R have been exposed.

Apply the NameCert obligations of Definition 6644.2. They force every accepted consumer to read the same finite modulus, window, tolerance, and readback spine before the diagonal-limit row and uniqueness row become public.

The structural rows H, C, P, N then transport, replay, source, and name the same witness route. Since the excluded host and quotient objects are absent from the carrier, the diagonal witness is exported only as this finite transducer route. □

Theorem 6644.4 (Real completion witness transducer uniqueness handoff)**.** *If two accepted* $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$ *packets agree on the regular Cauchy source row S, explicit modulus row M, selected window row W, dyadic tolerance ledger D, RegSeqRat readback row R, and structural rows H, C, P, N, then any Real-facing comparison of their diagonal candidates is read through their displayed uniqueness rows U. No comparison may replace those rows by quotient real equality, global stream equality, or a host completeness theorem.*

Proof. Project both packets. Agreement on $S, M, W, D, R, H, C, P, N$ fixes the finite route that reaches the diagonal-limit row in each carrier.

The NameCert obligations make the uniqueness row U the only comparison-facing coordinate after the diagonal-limit row is displayed. Thus a Real-facing consumer must inspect U before treating the two candidates as the same completed witness.

The carrier has no quotient-equality, global-stream, or ambient-completion coordinate. Therefore any comparison that bypasses the displayed uniqueness row is not an accepted read of the transducer packet. □

Independence witness: real-completeness, streamname, dyadicratcore, regseqrat, real: RealCompletionWitnessTr is not bijective to any listed sibling because it jointly carries the regular Cauchy source, explicit modulus, selected finite windows, dyadic tolerance ledger, RegSeqRat readback, diagonal-limit row, uniqueness row, transport, replay, provenance, and local NameCert in one completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessTransducer}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionWitnessTransducer}^{\uparrow}$ packet is generated from a regular Cauchy source, explicit modulus, selected finite windows, dyadic tolerance ledger, RegSeqRat readback, diagonal-limit row, uniqueness row, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWitnessTransducer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6644.1, Definition 6644.2, Definition 6644.3, and Definition 6644.4 seed the finite Real-completion witness route over modulus, window, dyadic tolerance, RegSeqRat, diagonal-limit, uniqueness, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWitnessTransducer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected host limits, countable choice, ambient completion, global stream equality, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed Real-completion witness transducer route.

Chapter 6645

A Concrete Naming Certificate for $\mathsf{CoarseMetricSpace}^{\uparrow}$

Following Roe and Bourbaki/Kelley large-scale metric practice, the $\mathsf{CoarseMetricSpace}^{\uparrow}$ packet records coarse control as finite BEDC NameCert data over displayed metric and uniform-space rows. It reads the metric source of chapter 57, the entourage discipline of chapter 339, and the terminal distance readback surface of chapter 38. The packet gives later large-scale consumers a controlled-entourage and bounded-set route without constructing a quotient coarse category, selecting representatives, importing countable choice, or asserting an external large-scale theorem.

Definition 6645.1 (CoarseMetricSpace carrier)**.** *A* $\mathsf{CoarseMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, U, E, B, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the admitted point classifier, finite distance-query surface, and metric observation histories.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, recording the finite entourage requests whose small-scale control is visible to the packet.*
3. *E is the controlled large-scale entourage budget, assigning displayed coarse-radius or tolerance rows to the finite entourages allowed for downstream coarse consumers.*
4. *B is the bounded-set ledger, recording the finite named bounded rows and the radius/readback witnesses by which a consumer may use them.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for metric distances, controlled radii, and bounded-set witnesses.*
6. *H records componentwise* hsame *transport for the metric source, uniform entourage row, controlled-entourage budget, bounded-set ledger, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from metric-distance and entourage requests through U, E, B, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CoarseMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CoarseMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, E, B, R, H, C, P, N$. It has no coordinate for quotient coarse-space equality, a selected net, a choice-selected bornology, a host large-scale category, or ambient metric completion.

Theorem 6645.2 (CoarseMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{CoarseMetricSpace}^{\uparrow}$ *carrier* $K = (M, U, E, B, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CoarseMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Met}^{\uparrow}$ *source,* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, controlled large-scale entourage budget, bounded-set ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every coarse consumer must read* $M, U, E, B, R, H, C, P, N$ *before treating the packet as large-scale metric data, and every request for quotient completion, selected representatives, countable choice, host coarse categories, or ambient completeness is refused by absence from the carrier.*

Proof. Carrier admission is projection from Definition 6645.1. The tuple displays the metric source, uniform entourage row, controlled-entourage budget, bounded-set ledger, readback, transport, replay, provenance, and local name.

The control obligations are finite route inspections. A consumer first reads M for metric distances, U for entourage shape, E for large-scale control, and B for bounded-set access.

Readback and stability then pass through R, H, C, P, N. These rows transport, replay, source, and name the same finite route, so the NameCert surface contains no quotient coarse category, selected representative, countable-choice schedule, host bornology, or ambient completeness principle. □

Theorem 6645.3 (CoarseMetricSpace controlled entourage route)**.** *Let* $K = (M, U, E, B, R, H, C, P, N)$ *be an accepted* $\mathsf{CoarseMetricSpace}^{\uparrow}$ *carrier. Every controlled-entourage consumer must first display the* $\mathsf{Met}^{\uparrow}$ *source* M*, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row* U*, the large-scale budget* E*, the bounded-set ledger* B*, and the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback row* R *before it can replay a coarse relation. The replay is transported only through* hsame*, continued only through* Cont*, sourced only through* Pkg*, and named only by the local* NameCert *row* N*.*

Proof. Project the accepted carrier. The only metric and uniform data available to a controlled-entourage consumer are M and U.

Read the large-scale rows next. The budget E records which displayed entourage rows are allowed at the requested scale, while B records the bounded-set ledger used by coarse consumers.

The readback row R converts the displayed distance and radius requests through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ before replay. The structural rows H, C, P, N then preserve that same route and add no selected net, quotient equality, countable choice, host large-scale category, or ambient completion object. □

Falsifiable prediction: An accepted $\mathsf{CoarseMetricSpace}^{\uparrow}$ read is refuted if a consumer obtains controlled entourage data or bounded-set evidence without displaying the MetricUp source, UniformSpace entourage row, controlled large-scale budget, bounded-set ledger, RegSeqRat/Real readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: metric, uniformspace, real: none is bijective to $\mathsf{CoarseMetricSpace}^{\uparrow}$ because this packet jointly carries a metric source, an entourage row, a controlled large-scale budget, bounded-set ledger data, rational and real readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CoarseMetricSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CoarseMetricSpace}^{\uparrow}$ packet is generated from a displayed MetricUp source, UniformSpace entourage row, controlled large-scale entourage budget, bounded-set ledger, RegSeqRat/Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoarseMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6645.1, Definition 6645.2, and Definition 6645.3 seed the finite coarse metric certificate over displayed metric, uniform entourage, large-scale control, boundedness, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoarseMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient coarse-space equality, selected nets, host coarse categories, countable-choice schedules, ambient completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, controlled-entourage exactness, bounded-set ledger stability, readback transport, provenance non-escape, and local naming for BEDC.Derived.CoarseMetricSpaceUp.

Chapter 6646

A Concrete Naming Certificate for $\mathsf{MetricSequentialSpace}^{\uparrow}$

Following Kelley and Engelking, the $\mathsf{MetricSequentialSpace}^{\uparrow}$ packet records the classical metric-space sequentiality interface as finite BEDC data. It sits over the metric source of chapter 57, the topology neighbourhood surface of chapter 88, the sequence-limit closure route of chapter 1571, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the displayed metric convergence route that a topology-facing sequential-closure consumer may inspect; it is not a quotient of streams, not arbitrary topological sequentiality, not a choice principle, not open-cover compactness, and not completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6646.1 (Metric sequential space carrier)**.** *A* $\mathsf{MetricSequentialSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, T, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the metric source row, recording the visible carrier, distance classifier, zero-distance boundary, and finite tolerance comparisons consumed from* $\mathsf{Met}^{\uparrow}$*.*
2. *T is the topology neighbourhood row, recording the neighbourhood names around candidate limits and the topology-facing reads supplied by the metric source.*
3. *W is the sequence window row, recording the finite* $\mathsf{StreamName}^{\uparrow}$ *schedules, inspected indices, and admitted entries used by convergence consumers.*
4. *R is the regular rational readback row, translating each displayed metric tolerance request into* $\mathsf{RegSeqRat}^{\uparrow}$ *data before any real comparison is read.*
5. *L is the limit row, recording the candidate limit point, the neighbourhood requests around it, and the metric tolerance ledger sealed through* $\mathsf{Real}^{\uparrow}$*.*
6. *H stores componentwise* hsame *transport for the metric source, topology neighbourhoods, sequence windows, regular rational readback, limit row, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from a sequential-closure request through D, T, W, R, L without adding a host equality or subsequence-selection route.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{SequentialClosure}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricSequentialSpace}^{\uparrow}}$ *row.*
The carrier has no coordinate for selected subsequences, quotient streams, arbitrary topological sequentiality, open-cover compactness, detached host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6646.2 (Metric sequential space NameCert obligations)**.** *For every accepted* $\mathsf{MetricSequentialSpace}^{\uparrow}$ *carrier $M = (D, T, W, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialSpace}^{\uparrow}}$ *surface consists exactly of the metric source row D, topology neighbourhood row T, sequence window row W, regular rational readback row R, limit row L, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N.*

Proof. Project the finite BHist carrier. The metric source D supplies the distance classifier and tolerance comparisons, while the topology row T supplies the neighbourhood names that a sequential-closure consumer may ask to read.

The sequence data are displayed before any limit read. Windows are stored in W, rational readback in R, and the candidate limit ledger in L, so convergence evidence cannot be produced by a detached stream or a host-selected subsequence.

Rows H, C, P, N then transport, replay, package, and name exactly those displayed rows. Since the carrier contains no additional coordinate for quotient streams, arbitrary topological sequentiality, choice, compactness, host equality, or $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6646.3 (Metric sequential space sequential-closure handoff)**.** *Let $M = (D, T, W, R, L, H, C, P, N)$ be an accepted* $\mathsf{MetricSequentialSpace}^{\uparrow}$ *packet. Every* $\mathsf{SequentialClosure}^{\uparrow}$ *consumer of the metric candidate limit factors through*

$$D, T, W, R, L \longmapsto H, C, P, N.$$

Thus topology-facing sequential closure receives the displayed metric convergence route, finite StreamName windows, RegSeqRat readback, Real-sealed limit row, and local NameCert packaging, and receives no host closure operator.

Proof. Start with a topology-facing closure request. Carrier admission exposes the metric source D and neighbourhood row T before the request can inspect any sequence or limit data.

Replay through C then forces the request to consume the finite windows W, the regular rational readback R, and the limit ledger L. This is the same finite route needed by the $\mathsf{SequentialClosure}^{\uparrow}$ surface, but it is anchored in the metric classifier rather than in an ambient closure operator.

Transport by H and packaging by P, N preserve the displayed route. Hence the exported handoff is the stated finite factorization, with no separate topological closure, quotient stream equality, selected subsequence, compactness principle, or real-completeness theorem. □

Theorem 6646.4 (Metric sequential space nonescape)**.** *An accepted* $\mathsf{MetricSequentialSpace}^{\uparrow}$ *packet cannot export selected subsequences, quotient stream equality, arbitrary topological sequentiality, open-cover compactness, host equality of metric points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from its sequentiality interface.*

Proof. Inspect the carrier rows of Definition 6646.1. The sequence interface is the finite window row W, the metric tolerance interface is the readback row R, and the limit interface is the displayed row L sealed through the available Real comparison data.

Apply Definition 6646.2. The local NameCert row names only $D, T, W, R, L, H, C, P, N$, and Definition 6646.3 shows that every closure consumer passes through those rows in order.

The refused principles have no BHist row, no hsame transport coordinate, no Cont replay edge, no Pkg provenance entry, and no local NameCert clause. The packet therefore exposes metric sequentiality only as the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSequentialSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSequentialSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6646.1, Definition 6646.2, Definition 6646.3, and Definition 6646.4 seed the finite metric sequentiality route from metric convergence data to topology-facing sequential-closure consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSequentialSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subsequences, quotient streams, arbitrary topological sequentiality, open-cover compactness, host equality of metric points, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-to-topology neighbourhood replay, StreamName window coverage, RegSeqRat and Real readback, classifier transport, and non-escape for the same packet.

Constructive story: A $\mathsf{MetricSequentialSpace}^{\uparrow}$ packet is generated from a metric source row, topology neighbourhood row, finite StreamName window row, RegSeqRat readback row, Real-sealed limit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6647

A Concrete Naming Certificate for $\mathsf{KuratowskiClusterSet}^{\uparrow}$

From Kuratowski convergence of closed sets and Kelley–Moore–Smith net cluster-set practice, the $\mathsf{KuratowskiClusterSet}^{\uparrow}$ packet records boundary cluster evidence by finite hit–miss windows rather than by a host powerset of possible cluster points. It sits beside $\mathsf{KuratowskiConvergence}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the L10 net/filter cluster-set carrier used by completion-facing reads.

The hit–miss rows are read through chapter 6591, the net and filter source rows through chapter 3403 and chapter 1601, and the compact/metric support rows through chapter 129 and chapter 57 before the carrier exposes K_h, K_m, C, M, S to any cluster-set consumer.

Definition 6647.1 (Kuratowski cluster-set carrier)**.** *A* $\mathsf{KuratowskiClusterSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, K_h, K_m, C, M, B, X, R, E, T, U, P, N)$$

with the following displayed rows:

1. *S is the net or filter source row, naming either a* $\mathsf{MooreSmithNet}^{\uparrow}$ *index-window read or a* $\mathsf{FilterBase}^{\uparrow}$ *base-window read.*
2. *F records the finite source-frontier windows consumed by the cluster-set request.*
3. *K_h is the Kuratowski hit ledger: each row names a displayed source window, a metric neighborhood, and a later witness whose image meets that neighborhood.*
4. *K_m is the Kuratowski miss ledger: each row names a finite exclusion or compact containment window showing that the inspected boundary claim does not escape the displayed metric neighborhood.*
5. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row used when compact subwindows make the hit and miss ledgers finite.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-observation row used by the hit, miss, compact-support, and Real-facing reads.*
7. *B is the cluster-hit witness row, storing only displayed later-window hits and never a selected cluster point.*
8. *X is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for source-name and observation-name replay.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational row used to read metric tolerances and shrinking windows.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for completion-facing consumers.*
11. *T, U, P, N are respectively componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{KuratowskiClusterSet}^{\uparrow}}$ *naming row.*

The carrier has no coordinate for a host powerset of cluster points, a selected cluster point, quotient equality of closed sets, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.KuratowskiClusterSetUp.TasteGate.KuratowskiClusterSetUp`

Theorem 6647.2 (Kuratowski cluster-set NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiClusterSet}^{\uparrow}$ *carrier*

$$K = (S, F, K_h, K_m, C, M, B, X, R, E, T, U, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiClusterSet}^{\uparrow}}$ *surface consists exactly of net-or-filter source admission, finite frontier-window admission, Kuratowski hit-window admission, Kuratowski miss-window admission, compact and metric support, displayed cluster-hit witnessing,* $\mathsf{StreamName}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6647.1. The mathematical rows are precisely $S, F, K_h, K_m, C, M, B, X, R, E$ the structural rows are T, U, P, N.

The hit and miss obligations are separated. A hit request must read a later-window witness from K_h and B, while a miss request must read the finite exclusion or compact containment evidence from K_m together with the metric row M and compact support row C.

The readback rows X, R, E expose only names, rational tolerances, and terminal Real-facing seals downstream of those finite windows. Transport, replay, provenance, and local naming preserve the same route, so no additional NameCert clause can be supplied by a powerset, selected point, quotient, choice principle, or ambient completeness proof. □

Lean4 ✓ `BEDC.Derived.KuratowskiClusterSetUp.TasteGate.KuratowskiClusterSetTasteGate single carrier alignment`

Theorem 6647.3 (Kuratowski cluster-set net-filter handoff)**.** *Every net-facing or filter-facing cluster-set read exported by an accepted* $\mathsf{KuratowskiClusterSet}^{\uparrow}$ *carrier factors through the displayed route*

$\mathsf{MooreSmithNet}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$, $\mathsf{KuratowskiConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}$

The handoff exposes the source frontier, Kuratowski hit and miss windows, compact or metric support, cluster-hit witness, stream-name replay, regular rational tolerance row, and terminal Real seal before any completion consumer may use the cluster-set boundary. It exports no host powerset, selected cluster point, quotient closed-set equality, countable-choice functional, or ambient Real completeness argument.

Lean4 ✓ `BEDC.Derived.KuratowskiClusterSetUp.KuratowskiClusterSetNetFilterHandoff`

Proof. Begin with the source row S. A net read uses the displayed $\mathsf{MooreSmithNet}^{\uparrow}$ window and a filter read uses the displayed $\mathsf{FilterBase}^{\uparrow}$ base-window route; both are recorded as finite frontier data in F before a cluster request is available.

Next read the Kuratowski rows. The hit ledger K_h and miss ledger K_m are the only rows where the boundary condition is tested, and their metric and compact support is supplied by M and C. The cluster-hit witness row B records later-window hits rather than a selected cluster point.

Finally pass through X, R, E, T, U, P, N. These rows provide stream-name replay, regular rational tolerance readback, Real-facing sealing, transport, continuation replay, provenance, and local naming. Since the carrier contains no powerset, quotient, selected point, choice functional, or ambient completeness coordinate, the net/filter handoff is exhausted by the displayed finite route. □

Theorem 6647.4 (Kuratowski cluster-set sequential compact route)**.** *Every sequential-compact cluster consumer of an accepted* $\mathsf{KuratowskiClusterSet}^{\uparrow}$ *carrier must read the displayed sibling route through* $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *including the chapter routes of chapter 2372 and chapter 129. The consumer uses C as the compact-metric support row, M as the metric observation row, X as the stream-name replay row, R as the regular rational tolerance row, and E as the terminal Real seal before it may read the hit ledger K_h, miss ledger K_m, or cluster-hit witness row B as a sequential cluster boundary. It exports no host subsequence selector, countable-choice functional, selected cluster point, quotient closed-set equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KuratowskiClusterSetUp.KuratowskiClusterSetSequentialCompactRoute`

Proof. Begin with the compact source. The $\mathsf{SequentialCompact}^{\uparrow}$ chapter routes sequential cluster reads through displayed compact-metric and finite-window data, while the $\mathsf{CompactMetric}^{\uparrow}$ route supplies only the compact support row visible here as C.

Next expose the metric and readback rows. The carrier rows M, X, R, E provide metric observations, stream-name replay, regular rational tolerance readback, and the terminal Real seal before the Kuratowski hit and miss ledgers can be consumed.

Finally return to K_h, K_m, B, T, U, P, N. The hit and miss ledgers remain finite boundary rows transported and replayed through the same packet; they do not become a host subsequence selector, a countable-choice witness, a selected cluster point, quotient closed-set equality, or ambient Real completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiClusterSet}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiClusterSet}^{\uparrow}$ packet is generated from net or filter source rows, finite frontier windows, Kuratowski hit and miss ledgers, compact and metric support, cluster-hit witnesses, StreamName readback, RegSeqRat tolerance readback, Real seals, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiClusterSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6647.1, Definition 6647.2, Definition 6647.3, and Definition 6647.4 seed the L10 net/filter and sequential-compact cluster-set boundary by displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiClusterSet}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim selected cluster points, host powersets, quotient closed-set equality, countable choice, ambient Real completeness, theorem checking, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, hit-window coverage, miss-window coverage, net/filter source handoff, compact and metric support, readback sealing, transport, replay, provenance, and local naming for BEDC.Derived.KuratowskiClusterSetUp.

Chapter 6648

A Concrete Naming Certificate for $\mathsf{BishopLocatedSubset}^{\uparrow}$

Following Bishop–Bridges constructive analysis, a located subset of the real line is read through finite evidence for membership, apartness, and distance-to-subset comparison rather than through an ambient powerset. The $\mathsf{BishopLocatedSubset}^{\uparrow}$ packet records that frontier as NameCert data over the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the metric and located-metric distance surfaces of chapter 57 and chapter 1533, and the located-subset interface of chapter 1541. It gives Real and Completion consumers a finite subset boundary without importing host set extensionality, quotient subsets, open-cover compactness, selected nearest points, or countable choice.

Definition 6648.1 (Bishop located-subset carrier)**.** *A* $\mathsf{BishopLocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, A, R, D, E, G, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{LocatedMetric}^{\uparrow}$ *distance row for the real-line source, including the finite distance observations used by membership and apartness queries.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *admission row, recording the displayed subset carrier and the finite membership rows for inspected* $\mathsf{Real}^{\uparrow}$ *names.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which inspected real names, membership windows, and subset-distance requests are read before any terminal real seal is consumed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for positive radii, apartness gaps, enclosure comparisons, and radius refinement.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected real names and the distance witnesses attached to the subset query.*
6. *G is the located-apartness witness row: for each negative membership read it displays a positive dyadic radius whose* $\mathsf{LocatedMetric}^{\uparrow}$ *ball is disjoint from the admitted membership rows already visible through A, R, D, E.*
7. *H records componentwise* hsame *transport for the metric, located-set, readback, dyadic, real-seal, gap-witness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a membership or apartness request through M, A, R, D, E, G, H.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedSubset}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, R, D, E, G, H, C, P, N$. It has no coordinate for a host powerset, quotient subset equality, selected complement point, open-cover compactness theorem, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6648.2 (Membership and apartness boundary)**.** *For every accepted* $\mathsf{BishopLocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, A, R, D, E, G, H, C, P, N),$$

each public membership or apartness read factors through the displayed $\mathsf{LocatedSet}^{\uparrow}$ *row A, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{Real}^{\uparrow}$ *seal row E, and the located-apartness radius row G. A negative membership read exposes a positive witness radius only as a finite* $\mathsf{LocatedMetric}^{\uparrow}$ *distance comparison, not as a host complement predicate, a trichotomy decision, or a selected nearest point.*

Proof. Project the carrier of Definition 6648.1. The rows $M, A, R, D, E, G, H, C, P, N$ are the full inventory available to the local certificate.

The membership side is supplied by A and read through R, D, E. Thus an inspected real name reaches the subset row only after its regular-rational window, dyadic tolerance ledger, and terminal real seal have been displayed.

The apartness side is the row G. Its witness radius is positive because D supplies the dyadic positivity and refinement ledger, and it is a subset-facing witness because M supplies the located-metric distance comparison against the admitted rows in A.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since no row stores a host complement, trichotomy split, nearest-point selector, quotient subset, or ambient completeness theorem, the stated boundary exhausts the public membership and apartness reads. □

Theorem 6648.3 (Real seal route for located subsets)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopLocatedSubset}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{LocatedMetric}^{\uparrow} \longmapsto \mathsf{LocatedSet}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows retained. The route exports a Bishop located-subset boundary over real names; it does not export host set extensionality, quotient subset membership, open-cover compactness, countable choice, selected limit values, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the distance row M. It supplies the metric and located-metric observations needed by every membership, apartness, and distance-to-subset request.

Read the subset row A next. The row records only displayed admission and membership evidence, so the consumer has a subset boundary but not an ambient powerset.

The numerical route then passes through R, D, E. The regular-rational row exposes the inspected windows, the dyadic row records the tolerance and positive-radius comparisons, and the real seal is consumed only after those finite rows are present.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Therefore the Real-facing handoff cannot bypass the finite carrier through extensional subsets, quotient representatives, open covers, selected limits, or an ambient completeness principle. □

Theorem 6648.4 (Bishop located-subset NameCert obligations)**.** *For an accepted* $\mathsf{BishopLocatedSubset}^{\uparrow}$ *carrier* $L = (M, A, R, D, E, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient metric and located-metric distance exposure, located-subset membership rows, regular-rational readback, dyadic tolerance and positive-radius exactness, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, located-apartness witness exposure, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier. Its visible rows are precisely the metric source, located subset, regular-rational readback, dyadic tolerance ledger, real seal, gap witness, transport, replay, provenance, and local naming rows.

The carrier admission and membership obligations are M, A, R, D, E. They force every inspected real-name and subset-membership read through the same finite L10 route before any public answer is available.

The located-apartness obligation is G. It can be consumed only with the positive-radius evidence in D and the distance comparison in M, so it is a finite witness row rather than an external complement classifier.

The support rows H, C, P, N add no analytic content beyond transport, replay, provenance, and naming. Hence the listed rows exhaust the NameCert obligation surface, with no host powerset, quotient subset equality, selected nearest point, open-cover compactness, countable choice, or ambient real-completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedSubset}^{\uparrow}$ read exposes a Metric and LocatedMetric distance row, a LocatedSet membership row, RegSeqRat readback, DyadicRatCore tolerance and positive-radius data, a terminal RealUp seal, a located-apartness witness radius, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before any Real or Completion consumer reads the subset boundary.

Independence witness: real, regseqrat, dyadicratcore, metric, locatedmetric, locatedset: $\mathsf{BishopLocatedSubset}^{\uparrow}$ is not bijective with any listed sibling because it jointly binds real-name membership rows, located-apartness witness radii, located-metric distance comparisons, regular-rational readback, dyadic tolerance refinement, terminal real seals, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedSubset}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedSubset}^{\uparrow}$ packet is generated from MetricUp and LocatedMetricUp distance rows, LocatedSetUp membership rows, RegSeqRatUp readbacks, DyadicRatCoreUp tolerance and positive-radius ledgers, RealUp seal rows, located-apartness witness radii, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedSubset}^{\uparrow}}$, seed)

Scope closed: Definition 6648.1, Definition 6648.2, Definition 6648.3, and Definition 6648.4 seed the finite Bishop located-subset carrier over Real, RegSeqRat, DyadicRatCore, Metric, LocatedMetric, LocatedSet, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedSubset}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host set extensionality, quotient subset equality, open-cover compactness, selected nearest points, countable choice, ambient Real completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, membership exposure, located-apartness radius exactness, regular-rational readback, dyadic tolerance transport, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedSubsetUp.

Chapter 6649

A Concrete Naming Certificate for $\mathsf{BernoulliInequality}^{\uparrow}$

Following Bernoulli and Hardy–Littlewood–Polya, the $\mathsf{BernoulliInequality}^{\uparrow}$ packet records the finite binomial lower-bound route used before Real algebra and calculus estimates read a power comparison. It sits beside the terminal real-name seal of chapter 38, the dyadic arithmetic ledger of chapter 291, and the regular rational readback surface of chapter 301. The packet exports the finite comparison route for a nonnegative base and unary exponent; it is not a host real-power theorem, a global ordered-field library, trichotomy, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6649.1 (Bernoulli inequality carrier)**.** *A* $\mathsf{BernoulliInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, N, D, Q, E, H, C, P, L)$$

with the following displayed rows:

1. *X is the nonnegative* $\mathsf{Real}^{\uparrow}$ *base row, displaying the accepted real seal for the term $1+x$ together with the proof-relevant nonnegative boundary for x.*
2. *N is the unary exponent row, recording the finite successor spine used for the power read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *binomial ledger, containing the displayed coefficient rows, nonnegative remainder rows, and tolerance budget used by the finite binomial comparison.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected base, coefficient, product, and remainder observations.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison seal for the lower bound between the finite power read and $1 + Nx$.*
6. *H stores componentwise* hsame *transport for the base, exponent, dyadic binomial ledger, regular-rational readback, comparison seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the base and exponent rows through the dyadic binomial ledger, RegSeqRat readback, and Real comparison seal.*
8. *P is the* Pkg *provenance over* $\mathsf{BernoulliInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{BernoulliInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, N, D, Q, E, H, C, P, L$. It has no coordinate for host exponentiation, a selected infinite binomial series, trichotomy, quotient real equality, global ordered-field automation, or ambient Real completeness.

Theorem 6649.2 (Bernoulli inequality NameCert obligations)**.** *For every accepted* $\mathsf{BernoulliInequality}^{\uparrow}$ *carrier* $B = (X, N, D, Q, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{BernoulliInequality}^{\uparrow}}$ *surface consists exactly of carrier admission, nonnegative* $\mathsf{Real}^{\uparrow}$ *base exposure, unary exponent exposure, dyadic binomial-ledger exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal Real comparison sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No consumer may read a Bernoulli comparison from the packet before* $X, N, D, Q, E, H, C, P, L$ *are displayed.*

Proof. Project the carrier of Definition 6649.1. The accepted rows are exactly the nonnegative base, unary exponent, dyadic ledger, regular-rational readback, Real comparison seal, and the structural rows.

The base and exponent obligations are read from X and N. The packet therefore exposes a finite successor spine and a nonnegative Real boundary before any power comparison is public.

The binomial comparison is read from D, Q, E. The dyadic ledger records the coefficients and nonnegative remainder budget, the regular-rational row packages the inspected arithmetic observations, and the Real seal is consumed only after those rows are present.

Rows H, C, P, L transport, replay, source, and name the same finite route. Since the carrier lacks host exponentiation, global order automation, quotient equality, selected infinite series, and ambient completeness coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 6649.3 (Bernoulli dyadic binomial handoff)**.** *Every accepted* $\mathsf{BernoulliInequality}^{\uparrow}$ *comparison handoff factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the displayed carrier rows X, D, Q, E, H, C, P, L*. The dyadic binomial ledger* D *supplies the finite coefficient and nonnegative remainder budget, the RegSeqRat readback* Q *packages the inspected rational observations, and the terminal Real seal* E *exports only the finite Bernoulli lower-bound comparison for the unary exponent row* N*.*

Proof. Start with an accepted comparison request. The base row X supplies the nonnegative Real input, and the exponent row N supplies the finite successor spine.

The request then enters the dyadic binomial ledger D. That row records the coefficient and remainder budget needed for the finite binomial comparison and keeps every comparison at a displayed dyadic tolerance.

The readback row Q packages the visible arithmetic observations before the Real seal E is available. Transport, replay, provenance, and local naming are supplied by H, C, P, L and do not add any host arithmetic principle.

Thus the handoff is the finite route through the Real, DyadicRatCore, RegSeqRat, and Real rows. It does not export host powers, quotient real equality, trichotomy, an infinite binomial series, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{BernoulliInequality}^{\uparrow}$ comparison read exposes a nonnegative RealUp base row, a unary exponent row, a DyadicRatCore binomial ledger, RegSeqRat readback, a terminal RealUp comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any lower-bound comparison is exported.

Independence witness: real, dyadicratcore, regseqrat, real-triangle-inequality, cauchy-product-tail-budget: $\mathsf{BernoulliInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a nonnegative base, unary exponent, dyadic binomial coefficient and remainder ledger, regular-rational readback, terminal Real comparison seal, transport, replay, provenance, and local NameCert in one inequality packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BernoulliInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{BernoulliInequality}^{\uparrow}$ packet is generated from a nonnegative RealUp base row, a unary exponent row, a DyadicRatCore binomial ledger, RegSeqRat readback, a terminal RealUp comparison seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BernoulliInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6649.1, Definition 6649.2, and Definition 6649.3 seed the finite Bernoulli comparison route over RealUp, DyadicRatCoreUp, RegSeqRatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BernoulliInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host exponentiation, global ordered-field automation, trichotomy, quotient real equality, selected infinite binomial series, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nonnegative base exposure, unary exponent exposure, dyadic binomial-ledger exposure, RegSeqRat readback, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BernoulliInequalityUp.

Chapter 6650

A Concrete Naming Certificate for $\mathsf{DependentEliminatorCoverage}^{\uparrow}$

The $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ packet realizes the typed-examples coverage frontier of section 9386.6 as finite evidence separating representative gallery rows from exhaustive dependent-eliminator coverage. It sits beside the typed normal-form scheduler of chapter 6617, the closed type checker seal of chapter 2182, the dependent-codomain stability boundary of chapter 3626, and the typed-conversion residual frontier of chapter 6069. The packet records typed windows, eliminator families, conversion boundaries, gallery status, obstruction retention, transport, replay, provenance, and local naming rows; it is not a universal subject-reduction theorem, total eliminator computation theorem, quotient syntax equality, host conversion oracle, or closed consistency claim.

Definition 6650.1 (Dependent eliminator coverage carrier)**.** *A* $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (W, F, B, G, O, H, C, P, N)$$

with the following displayed rows:

1. *W is the typed-window row, recording the finite contexts, binders, dependent codomains, polymorphic instantiations, and conversion-sensitive endpoints inspected by the packet.*
2. *F is the eliminator-family row, naming the dependent identity, pair, option, list, sum, natural, boolean, and Church-encoded eliminator computations whose example coverage is being audited.*
3. *B is the conversion-boundary row, exposing beta, projection, eliminator, nested-binder movement, and type-conversion boundaries before any coverage conclusion is public.*
4. *G is the gallery-status row, distinguishing witnessed representative examples from families that still require exhaustive coverage.*
5. *O is the obstruction-retention row, retaining dependent-codomain, subject-reduction, decidable-audit, confluence-audit, and normalization-frontier blockers as visible packet data.*
6. *H records componentwise* hsame *transport for typed windows, eliminator families, conversion boundaries, gallery status, obstruction retention, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from typed-window inspection through eliminator-family reads, conversion-boundary checks, gallery-status classification, and obstruction retention.*

8. *P is the* Pkg *provenance over* $\mathsf{DependentEliminatorCoverage}^{\uparrow}$, $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, $\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}$, $\mathsf{MetaCICTypedConversionResidualFront}$ BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DependentEliminatorCoverage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, F, B, G, O, H, C, P, N$. It has no coordinate for universal subject reduction, total eliminator computation, hidden conversion checking, quotient syntax equality, host normalization, or closed consistency.

Theorem 6650.2 (Dependent eliminator coverage NameCert obligations)**.** *For every accepted* $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ *carrier $D = (W, F, B, G, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DependentEliminatorCoverage}^{\uparrow}}$ *surface consists exactly of typed-window exposure, eliminator-family exposure, conversion-boundary exposure, gallery-status classification, obstruction-retention exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A public coverage read may report representative examples only after $W, F, B, G, O, H, C, P, N$ are displayed, and it may report exhaustive coverage only when the same rows carry the relevant family coverage witnesses rather than a gallery marker.*

Proof. Project the carrier of Definition 6650.1. The accepted rows are exactly the typed windows, eliminator families, conversion boundaries, gallery status, obstruction retention, and the structural rows.

The typed-window and eliminator-family obligations are read from W and F. They display the finite contexts, binders, dependent codomains, polymorphic instantiations, and eliminator computations under audit.

The boundary and status obligations are read from B, G, O. The conversion boundary fixes the computation endpoints being checked, the gallery row records whether the packet supplies representative examples or family coverage, and the obstruction row keeps the remaining blockers visible.

Rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier lacks universal subject reduction, total eliminator computation, hidden conversion checking, quotient syntax equality, host normalization, and closed consistency coordinates, the listed obligations exhaust the local NameCert surface. □

Theorem 6650.3 (Dependent eliminator gallery boundary)**.** *Every accepted* $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ *packet separates representative typed examples from exhaustive dependent-eliminator coverage by the displayed route*

$$W \longmapsto F \longmapsto B \longmapsto G \longmapsto O \longmapsto H, C, P, N.$$

If G marks a gallery row, the packet exports only the witnessed typed examples over the inspected windows. If G marks a family-coverage row, the packet must display the corresponding eliminator-family and conversion-boundary witnesses in F and B while retaining the obstruction row O for any dependent-codomain, subject-reduction, decidable-audit, confluence-audit, or normalization-frontier blocker.

Proof. Start with the typed-window row W. It fixes the finite contexts, binders, codomains, polymorphic instantiations, and conversion-sensitive endpoints that the packet is allowed to inspect.

The eliminator-family and conversion-boundary rows F, B then determine what computation evidence is actually displayed. Without those rows, a gallery example cannot be promoted to a family-coverage read.

The status and obstruction rows G, O supply the boundary. Gallery status keeps representative examples from being read as exhaustive coverage, while obstruction retention keeps the dependent-codomain and audit blockers visible.

Finally H, C, P, N preserve, replay, source, and name the same boundary. Thus the public read is a finite coverage packet, not a host theorem about all dependent eliminators or all conversion-sensitive programs. □

Falsifiable prediction: Any accepted $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ read that treats typed examples as exhaustive dependent-eliminator coverage without displaying typed-window, eliminator-family, conversion-boundary, gallery-status, obstruction-retention, transport, replay, provenance, and local naming rows is outside the carrier surface.

Independence witness: metacic-typedexamples, metacic-subject-reduction-obstruction, metacic-decidable-audit, metacic-confluence-audit, metacic-normalization-frontier: $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries typed windows, eliminator-family rows, conversion-boundary rows, gallery-status rows, obstruction-retention rows, transport, replay, provenance, and local NameCert rows in one coverage packet.

Lean work for $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ must provide $\mathsf{Nontrivial\ DependentEliminatorCoverage}^{\uparrow}$ and $\mathsf{FieldFaithful\ DependentEliminatorCoverage}^{\uparrow}$ instances so typed-window, eliminator-family, conversion-boundary, gallery-status, obstruction-retention, transport, replay, provenance, and local naming rows remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{DependentEliminatorCoverage}^{\uparrow}}$.

Constructive story: A $\mathsf{DependentEliminatorCoverage}^{\uparrow}$ packet is generated from typed-window rows, eliminator-family rows, conversion-boundary rows, gallery-status rows, obstruction-retention rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DependentEliminatorCoverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6650.1, Definition 6650.2, and Definition 6650.3 seed the finite typed-example coverage boundary over typed windows, eliminator families, conversion boundaries, gallery status, obstruction retention, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DependentEliminatorCoverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close universal subject reduction, total eliminator computation, hidden conversion checking, quotient syntax equality, host normalization, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, eliminator-family exposure, conversion-boundary exposure, gallery-status classification, obstruction-retention exposure, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.DependentEliminatorCoverageUp.

Chapter 6651

A Concrete Naming Certificate for $\mathsf{RealModulusSpace}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{RealModulusSpace}^{\uparrow}$ packet records a space-level family of convergence moduli as certified finite NameCert data. It sits after the finite observation surface of chapter 126, the regular rational readback of chapter 301, the single-convergence ledger of chapter 2074, and the terminal real-name seal of chapter 38. The packet packages reusable modulus families for Real and Completion consumers; it is not quotient equality of streams, a selected limit, countable choice over thresholds, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6651.1 (Real modulus space carrier)**.** *A* $\mathsf{RealModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, W, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row for the displayed family of Real-facing modulus schedules.*
2. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by those schedules.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
4. *L is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger row that binds each indexed schedule to a convergence tolerance request.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary available after the stream, regular-rational, and convergence-ledger rows have been displayed.*
6. *H records componentwise* hsame *transport for I, W, R, L, E, C, P, N.*
7. *C records displayed* Cont *replay from an index and tolerance request through the stream window, regular readback, convergence ledger, and terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RealModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, W, R, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice thresholds, host real equality, or ambient completeness.

Theorem 6651.2 (Real modulus space NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, W, R, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealModulusSpace}^{\uparrow}}$ *surface consists exactly of finite index admission,* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger compatibility, terminal* $\mathsf{Real}^{\uparrow}$ *seal compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming for the packet.*

Proof. Project the carrier of Definition 6651.1. The analytic rows are I, W, R, L, E, and the structural rows are H, C, P, N.

The finite index row I is read before any tolerance request reaches a stream window. The rows W and R then force each inspected window through the regular-rational readback surface.

Next read the convergence ledger L and the terminal seal E. The ledger binds the indexed tolerance request to a displayed $\mathsf{RealModulusOfConvergence}^{\uparrow}$ row, and the real seal is available only after that finite route has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no excluded host object appears as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6651.3 (Real modulus space L10 handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RealModulusSpace}^{\uparrow}$ *packet must read the finite handoff*

$$I, W, R, L, E, H, C, P, N$$

before it consumes space-level modulus data. The route passes through chapter 126, chapter 301, chapter 2074, and chapter 38. It exports a reusable finite modulus-family surface, not quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 6651.2. It keeps the finite index, stream-window, regular-readback, convergence-ledger, and real-seal rows visible before a consumer may read the structural rows.

Use the four L10 neighbours named in the statement. $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{RealModulusOfConvergence}^{\uparrow}$ supplies the tolerance ledger, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal boundary.

The transport, replay, provenance, and local naming rows preserve that finite route for Real and Completion consumers. Because the carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness, none of those objects is exported by the handoff. □

Theorem 6651.4 (Real seal nonescape for real modulus spaces)**.** *For every accepted* $\mathsf{RealModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, W, R, L, E, H, C, P, N),$$

any Real or Completion consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal row* E *must first expose the finite index row* I*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger row* L*, and the structural rows* H, C, P, N*. Consequently the seal exports no quotient stream equality, selected limit, countable-choice threshold, host real equality, or ambient completeness.*

Proof. Project the carrier of Definition 6651.1. The only route to E in an accepted packet passes through I, W, R, L and then through the structural rows H, C, P, N that transport, replay, source, and name the same finite history.

The row exhaustion statement in Definition 6651.2 leaves no additional coordinate beyond those rows. The Real and Completion handoff of Definition 6651.3 then forces every consumer of the space-level modulus data to read that finite route before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Since quotient stream equality, selected limits, countable-choice thresholds, host real equality, and ambient completeness are absent from the projected carrier, none of them can escape through the seal row. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusSpace}^{\uparrow}$ consumer that reads space-level modulus data exposes $I, W, R, L, E, H, C, P, N$ before reaching a Real or Completion handoff.

Independence witness: streamname, regseqrat, real-modulus-of-convergence, real: the listed siblings supply finite windows, regular readback, single convergence ledgers, and terminal real seals, but none alone binds a space-level family of Real-facing modulus schedules together with transport, replay, provenance, and local naming in one carrier packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusSpace}^{\uparrow}$ packet is generated from finite index rows for Real-facing modulus schedules, StreamName windows, RegSeqRat readback, RealModulusOfConvergence ledgers, a terminal RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6651.1, Definition 6651.2, Definition 6651.3, and Definition 6651.4 seed the space-level modulus-family carrier and Real-seal nonescape route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of streams, selected limits, countable choice over thresholds, host real equality, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, regular-readback compatibility, convergence-ledger compatibility, and local naming for BEDC.Derived.RealModulusSpaceUp.

Chapter 6652

A Concrete Naming Certificate for $\mathsf{CauchyPreservingMap}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and continuity on completed metric data, the $\mathsf{CauchyPreservingMap}^{\uparrow}$ packet records the finite interface for a map that sends displayed regular Cauchy witnesses to displayed regular Cauchy witnesses before any Real-facing consumer reads a seal. It sits over the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the graph-and-modulus route of chapter 130. The packet names preservation of regular Cauchy data as finite rows, not as host function equality, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6652.1 (CauchyPreservingMap carrier)**.** *A* $\mathsf{CauchyPreservingMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, T, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite source observations inspected before the map row is read.*
2. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S and the displayed tolerance ledger.*
3. *T is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by applying the map route to the same source window data.*
4. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *preservation row, recording the displayed late-threshold handoff that turns the source regular witness into a target regular witness.*
5. *L is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row consumed by the preservation read.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after S, R, T, Q, L have been displayed.*
7. *H records componentwise* hsame *transport for $S, R, T, Q, L, E, C, P, N$.*
8. *C records displayed* Cont *replay from source-window requests through source readback, target readback, Cauchy-modulus preservation, continuous-map data, and the Real seal handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyPreservingMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPreservingMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, T, Q, L, E, H, C, P, N$. It has no coordinate for host function equality, quotient stream equality, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6652.2 (CauchyPreservingMap NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPreservingMap}^{\uparrow}$ *carrier*

$$M = (S, R, T, Q, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyPreservingMap}^{\uparrow}}$ *surface consists exactly of source* $\mathsf{StreamName}^{\uparrow}$ *window admission, source* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, target* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyModulus}^{\uparrow}$ *preservation,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus routing,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6652.1. The mathematical rows are S, R, T, Q, L, E, and the structural rows are H, C, P, N.

The source and target regularity obligations are separated. A source read begins with the displayed $\mathsf{StreamName}^{\uparrow}$ window S and then reaches the source $\mathsf{RegSeqRat}^{\uparrow}$ row R; the map route can expose the target $\mathsf{RegSeqRat}^{\uparrow}$ row T only after the same source window has been read.

The preservation and continuity rows are then read as one finite handoff. The $\mathsf{CauchyModulus}^{\uparrow}$ row Q records the late-threshold preservation, while the $\mathsf{ContinuousMap}^{\uparrow}$ row L records the graph-and-modulus route used by the same read.

Finally E, H, C, P, N seal, transport, replay, source, and name that route. Since no additional coordinate exists, the listed rows are exactly the local NameCert obligations, and the excluded host and quotient objects cannot be exported. □

Theorem 6652.3 (CauchyPreservingMap regular sequence handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyPreservingMap}^{\uparrow}$ *packet that reads a target regular Cauchy witness must pass through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{CauchyModulus}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

in the displayed route $S, R, T, Q, L, E, H, C, P, N$. The handoff exports the source windows, source readback, target readback, preservation modulus, continuous-map row, Real seal, transport, replay, provenance, and local naming data; it exports no host function equality, quotient stream equality, selected limit, countable choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the source window row S. It is the only source inspection coordinate of the packet, so the source $\mathsf{RegSeqRat}^{\uparrow}$ readback R and the target readback T are both attached to the displayed finite windows.

Next read the preservation route. The $\mathsf{CauchyModulus}^{\uparrow}$ row Q supplies the late-threshold handoff for regularity preservation, and the $\mathsf{ContinuousMap}^{\uparrow}$ row L supplies the graph-and-modulus data that the target read uses.

Finally pass through the Real-facing and structural rows. The seal E is reached only after S, R, T, Q, L, while H, C, P, N keep the same route under componentwise transport, displayed replay, provenance, and local naming.

The exclusions are coordinate exhaustion. The carrier contains no host function-equality row, quotient stream equality row, selected-limit row, countable-choice schedule, or ambient Real-completeness witness, so none can be read by a completion-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPreservingMap}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyPreservingMap}^{\uparrow}$ packet is generated from source StreamName windows, source RegSeqRat readback, target RegSeqRat readback, CauchyModulus preservation, ContinuousMap graph-and-modulus data, a Real seal handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyPreservingMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6652.1, Definition 6652.2, and Definition 6652.3 seed the finite regular-Cauchy-preserving map route over StreamName, RegSeqRat, CauchyModulus, ContinuousMap, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyPreservingMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window coverage, source and target RegSeqRat readback, CauchyModulus preservation, ContinuousMap routing, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyPreservingMapUp.

Chapter 6653

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, regular Cauchy completeness is read as a finite handoff from regular rational sequences to displayed real names: a dyadic tolerance request opens only finitely many observations, the regularity row supplies the tail discipline, and the terminal real-name seal records the accepted landing name. The $\mathsf{RegularCauchyCompleteness}^{\uparrow}$ packet places that handoff over the dyadic tolerance surface of chapter 291, the finite window interface of chapter 126, the regular rational sequence route of chapter 301, and the real-name seal of chapter 38. It is a NameCert object for the RegSeqRat-to-Real boundary; it does not assert quotient completion, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6653.1 (Regular Cauchy completeness carrier)**.** *A* $\mathsf{RegularCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, M, L, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested precision and the dyadic comparisons inspected by the handoff.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the rational observations used at that precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row read from S and D before any real-facing seal is visible.*
4. *M is the regular tail modulus row assigning each displayed dyadic tolerance a finite tail threshold.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row offered as the real-name landing of the regular sequence.*
6. *W is the witness ledger recording that every late window governed by M reaches L within the displayed dyadic tolerance.*
7. *H records componentwise* hsame *transport for $D, S, R, M, L, W, C, P, N$.*
8. *C records displayed* Cont *replay from a Real or Completion tolerance request through D, M, S, R, W and then to the seal L.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompleteness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, M, L, W, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, real trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6653.2 (Regular Cauchy completeness NameCert obligations)**.** *For an accepted* RegularCauchyCompletener *carrier $Q = (D, S, R, M, L, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dyadic, stream, regular-rational, modulus, seal, witness, transport, replay, provenance, and local naming rows; classifier stability under componentwise* hsame *transport; modulus coverage for every dyadic tolerance request; witness exactness tying the late* $\mathsf{RegSeqRat}^{\uparrow}$ *windows to the terminal* $\mathsf{Real}^{\uparrow}$ *seal; and ledger non-escape for quotient completion, selected representatives, countable choice, host real equality, trichotomy, and ambient Real completeness.*

Proof. Carrier admission is projection from Definition 6653.1. The packet exposes precisely the dyadic tolerance row, finite stream windows, regularity row, tail modulus, terminal seal, witness ledger, and the structural rows H, C, P, N.

Classifier stability is componentwise. The transport row H can move only the displayed coordinates $D, S, R, M, L, W, C, P, N$, and the replay row C repeats the same ordered route from tolerance request to seal.

Modulus coverage and witness exactness are read in order. A dyadic tolerance request first fixes D, then the tail threshold M, then the finite windows S, the regular readback R, and the witness ledger W before L can be consumed.

The non-escape clause follows from the absence of any further coordinate. Since the carrier contains no quotient, selected representative, choice schedule, host equality, trichotomy, or ambient completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 6653.3 (Regular Cauchy completeness RegSeqRat handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{RegularCauchyCompleteness}^{\uparrow}$ *packet must read the finite route*

$$D, M, S, R, W \longmapsto L$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal is available. Thus the handoff from* $\mathsf{RegSeqRat}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$ *is carried by displayed dyadic tolerances, stream windows, regular tail data, and a witness ledger, not by quotienting Cauchy streams or choosing a host limit.*

Proof. Begin with the tolerance request. The dyadic ledger D fixes the precision, and the regular tail modulus M gives the finite threshold attached to that precision.

Read the finite observations next. The $\mathsf{StreamName}^{\uparrow}$ row S exposes only the inspected windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R records their regular rational discipline under the same dyadic budget.

The witness ledger W is then the only row that relates late windows to the terminal seal L. Hence the Real-facing read reaches L through the displayed finite route and has no access to quotient streams, selected limits, or host real equality. □

Theorem 6653.4 (Regular Cauchy completeness Real-seal nonescape)**.** *For every accepted* RegularCauchyComplete *carrier $Q = (D, S, R, M, L, W, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal L exports only the displayed RegSeqRat-to-Real handoff rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Any consumer asking for quotient completion, a selected representative, countable choice, host real equality, real trichotomy, or a theorem that $\mathsf{Real}^{\uparrow}$ *is complete is returned to $D, S, R, M, L, W, H, C, P, N$.*

Proof. Project the carrier. The Real seal is a displayed row L, and the only route to it passes through the dyadic tolerance ledger, finite windows, regularity row, tail modulus, and witness ledger.

Apply Definition 6653.2. The local NameCert surface requires every accepted read to preserve the same rows under componentwise transport and displayed continuation replay.

Apply Definition 6653.3. The RegSeqRat-to-Real boundary is therefore a finite handoff, so the excluded quotient, choice, host-equality, trichotomy, and ambient-completeness objects cannot be exported by the seal. □

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, StreamName finite windows, RegSeqRat regularity, a tail modulus, a terminal Real seal, a witness ledger, transport, replay, provenance, and local naming in one RegSeqRat-to-Real handoff packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompleteness}^{\uparrow}$ packet is generated from dyadic tolerance rows, StreamName finite windows, RegSeqRat regularity rows, regular tail moduli, terminal Real seals, witness ledgers, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6653.1, Definition 6653.2, Definition 6653.3, and Definition 6653.4 seed the finite RegSeqRat-to-Real completeness handoff over displayed dyadic, stream, regular-rational, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, real trichotomy, bridge checking, Lean verification, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.RegularCauchyCompletene

Chapter 6654

A Concrete Naming Certificate for $\mathsf{LocatedApartnessInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedApartnessInterval}^{\uparrow}$ packet records a located real interval equipped with visible apartness separation from exterior real names. It consumes the dyadic rational core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the located interval carrier of chapter 394, the real-apartness surface of chapter 304, and the terminal real-name seal of chapter 38. The chapter names interval separation as finite BEDC rows, not as trichotomy, quotient real equality, selected endpoints, countable choice, or ambient real completeness.

Definition 6654.1 (LocatedApartnessInterval carrier)**.** *A* $\mathsf{LocatedApartnessInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, L, U, E, S, R, D, G, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing lower and upper endpoint reads, containment rows, interval-cell ledgers, and endpoint-window routes.*
2. *L and U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seal rows of the interval source.*
3. *E is the exterior real-name row whose separation from the interval is being witnessed.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the inspected endpoint and exterior-name observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S, supplying regular rational endpoint and exterior observations.*
6. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for endpoint slack and apartness radius comparisons.*
7. *G is the* $\mathsf{ApartnessReal}^{\uparrow}$ *gap row, displaying either separation of E from the lower endpoint side or separation of E from the upper endpoint side through the same finite windows.*
8. *H records componentwise* hsame *transport for the interval source, endpoint seals, exterior row, stream windows, readbacks, dyadic tolerance row, apartness gap row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from interval membership or exterior-separation requests through $I, L, U, E, S, R, D, G, H$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedApartnessInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedApartnessInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient equality of real names, chosen endpoint representatives, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6654.2 (LocatedApartnessInterval NameCert obligations)**.** *For every accepted* $\mathsf{LocatedApartnessInterval}^{\uparrow}$ *carrier $A = (I, L, U, E, S, R, D, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedApartnessInterval}^{\uparrow}}$ *surface consists exactly of these finite rows: carrier admission for the displayed packet; source-interval exactness through I, L, U; exterior-name admission through E; finite-window and regular-readback exactness through S, R; positive dyadic budget exactness through D; apartness-gap exactness through G; and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 6654.1. The interval source, endpoint seals, exterior real row, finite windows, regular readback, dyadic tolerance, apartness gap, and structural rows are precisely the displayed coordinates of the packet.

Read the interval side first. The located interval source I exposes endpoint and containment data, while L and U seal those endpoints as $\mathsf{Real}^{\uparrow}$ rows before the exterior row can be compared to them.

Read the numerical evidence next. The finite window S and regular rational readback R give the endpoint and exterior observations, and the dyadic row D supplies the positive tolerance used by the separation ledger.

Finally read the apartness and structural rows. The row G is the only apartness coordinate, and H, C, P, N only transport, replay, source, and name the same finite route. Since no excluded host-order, quotient, choice, compactness, selected-endpoint, or completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6654.3 (LocatedApartnessInterval RegSeqRat handoff)**.** *Let $A = (I, L, U, E, S, R, D, G, H, C, P, N)$ be an accepted* $\mathsf{LocatedApartnessInterval}^{\uparrow}$ *carrier. Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of the interval-apartness read must obtain its rational data from the same finite window row S and readback row R before consuming the dyadic tolerance row D or the apartness gap row G. The handoff reaches* $\mathsf{ApartnessReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed endpoint, exterior-name, window, readback, tolerance, transport, replay, provenance, and local naming rows.*

Proof. Begin with a downstream read of rational endpoint or exterior-name data. The carrier stores all such observations in the finite window row S and the regular rational readback row R, so a consumer has no second rational source.

Attach the dyadic budget after the readback. The tolerance row D measures endpoint slack and separation radius only for the observations already exposed by S and R.

Now read the apartness handoff. The gap row G consumes the same endpoint and exterior observations together with the positive dyadic tolerance; it does not create a new real comparison or a host interval order.

The structural rows H, C, P, N preserve this route when it is replayed toward $\mathsf{ApartnessReal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$. Thus the handoff is exactly the finite RegSeqRat route attached to the located interval packet, with no quotient real equality, selected endpoint, countable choice, trichotomy, open-cover compactness, or ambient completeness export. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessInterval}^{\uparrow}}$.

Constructive story: A LocatedApartnessInterval$^{\uparrow}$ packet is generated from a LocatedInterval source row, Real endpoint seal rows, an exterior Real row, finite StreamName windows, RegSeqRat readback, a positive DyadicRatCore tolerance row, an ApartnessReal gap row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedApartnessInterval}^{\uparrow}}$, seed)

Scope closed: Definition 6654.1, Definition 6654.2, and Definition 6654.3 seed the finite interval-apartness packet and its regular rational handoff.

Formal verification: FormalStatus(NameCert$_{\text{LocatedApartnessInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient equality of real names, selected endpoint representatives, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, exterior-name admission, finite-window readback exactness, positive dyadic budget exactness, apartness-gap exactness, and structural nonescape for BEDC.Derived.LocatedApartnessIntervalUp.

Chapter 6655

A Concrete Naming Certificate for $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$

Following Adams and Bashforth's finite multistep method tradition, the $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$ packet records an explicit ODE step whose update is read from a bounded previous-step window and a finite table of displayed derivative samples. It sits after the ODE source of chapter 193, the real readback surface of chapter 38, the regular rational tolerance route of chapter 301, the stream-name replay route of chapter 126, the dyadic arithmetic core of chapter 291, and the one-step explicit Euler packet of chapter 6231. The certificate names a finite linear multistep update; it does not assert host solution existence, convergence of a mesh family, or completeness of an ambient real line.

Definition 6655.1 (Adams-Bashforth finite-step carrier)**.** *An* $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, B, W, F, S, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *source row, naming the local time-state cell, vector-field request, step index, and finite mesh window.*
2. *B is the dyadic coefficient table row, listing the Adams–Bashforth weights consumed by the displayed order and no coefficient outside that finite table.*
3. *W is the previous-step window row, recording the ordered finite list of already named time-state entries used by the multistep update.*
4. *F is the derivative-sample row, pairing each entry of W with the displayed* $\mathsf{Ode}^{\uparrow}$ *vector-field read and its* $\mathsf{StreamName}^{\uparrow}$ *replay name.*
5. *S is the explicit step-update row, reading the next state as the previous endpoint plus the dyadic weighted sum of the derivative samples.*
6. *L is the local truncation and error ledger, recording only the visible residual budget, mesh-width budget, and* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback attached to this one update.*
7. *H records componentwise* hsame *transport for the ODE source, coefficient table, previous-step window, derivative samples, update row, error ledger, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$O, B, W, F, L \longmapsto S \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{EulerMethodFiniteStep}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AdamsBashforthFiniteStep}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host solution curve, selected limiting path, quotient equality of functions, a global ODE existence theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6655.2 (Adams-Bashforth finite-step error ledger)**.** *For every accepted* $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$ *carrier $A = (O, B, W, F, S, L, H, C, P, N)$, the visible step update consumes only the ODE source row, the dyadic coefficient table, the previous-step window, the derivative-sample rows, and the local truncation and error ledger displayed in O, B, W, F, S, L. Its exported certificate is exactly the transported and replayed* BHist, hsame, Cont, Pkg, *and* NameCert *route through H, C, P, N. It exports no global ODE existence theorem, host function equality, selected limit, quotient path equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6655.1. The only numerical rows are O, B, W, F, S, L: the ODE source, dyadic coefficient table, previous-step window, derivative samples, explicit update, and local error ledger.

The update row S is not a host function. It is a displayed read from the previous endpoint in W, the finite coefficient table B, the derivative samples F, and the local tolerance data in L. Thus every arithmetic or error request is already tied to a finite row before a downstream consumer sees the update.

Finally transport and replay through H, C, P, N. These rows preserve the same finite packet as BEDC data, so they cannot add a global existence theorem, a host equality proof, a selected limiting path, quotient path equality, or a completeness argument absent from the carrier. □

Lean4 (statement) `BEDC.Derived.AdamsBashforthFiniteStepUp`

Closure status for $\mathsf{NameCert}_{\mathsf{AdamsBashforthFiniteStep}^{\uparrow}}$.

Constructive story: An $\mathsf{AdamsBashforthFiniteStep}^{\uparrow}$ packet is generated from an Ode source row, a dyadic coefficient table, a previous-step window, derivative-sample rows, an explicit update row, a local truncation and error ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AdamsBashforthFiniteStep}^{\uparrow}}$, seed)

Scope closed: Definition 6655.1 and Definition 6655.2 seed the finite Adams–Bashforth update route over ODE, Real, RegSeqRat, StreamName, DyadicRatCore, Euler finite-step, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AdamsBashforthFiniteStep}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, mesh-family convergence, host function equality, selected limits, quotient path equality, ambient Real completeness, theorem checking, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-table visibility, previous-window coverage, derivative-sample readback, explicit update exposure, local error-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.AdamsBashforthFiniteStepUp.

Chapter 6656

A Concrete Naming Certificate for $\mathsf{CofinalCauchySubnet}^{\uparrow}$

Following Moore–Smith and Kelley, the $\mathsf{CofinalCauchySubnet}^{\uparrow}$ packet records the finite NameCert surface for a Cauchy subnet route whose reindexing data are cofinal in the source directed tail order. It sits after the Moore–Smith net carrier of chapter 3403, the Cauchy-net surface of chapter 589, the convergence handoff of chapter 1667, the finite window discipline of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is an L10 Real and Completion carrier: it names the cofinal-tail evidence needed by completion-facing consumers without choosing a global subnet, quotienting nets, adding countable choice, importing ambient compactness, identifying host values, or asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6656.1 (Cofinal Cauchy subnet carrier)**.** *A* $\mathsf{CofinalCauchySubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, J, \phi, M, A, T, K, N, S, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the source directed-index row inherited from the accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *surface.*
2. *J is the subnet directed-index row, carrying only the finite subnet vertices and comparisons inspected by the packet.*
3. *ϕ is the monotone reindexing row from J to I.*
4. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *source-net row, exposing finite source values and tail requests.*
5. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *common-window row on the source net.*
6. *T is the cofinal-tail ledger: for each displayed source-tail request, it gives a displayed subnet tail whose image under ϕ lies inside that source tail.*
7. *K is the Cauchy-subnet compatibility row, recording that the subnet observations read through J, ϕ, M, T inherit the displayed Cauchy windows in A.*
8. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *consumer row, available only after the cofinal and Cauchy-subnet rows have been displayed.*
9. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row for the accepted subnet route.*
10. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*

11. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after* $I, J, \phi, M, A, T, K, N, S, R$ *are displayed.*
12. *H records componentwise* hsame *transport for the directed, net, Cauchy, cofinal, convergence, readback, seal, replay, provenance, and local naming rows.*
13. *C records displayed* Cont *replay through the same finite cofinal Cauchy subnet route.*
14. *P is the* Pkg *provenance over* $\mathsf{CofinalCauchySubnet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
15. *L is the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchySubnet}^{\uparrow}}$ *row.*

The carrier exports only these displayed rows. It has no coordinate for selected subnets, quotient-net equality, countable choice, ambient compactness, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6656.2 (Cofinal Cauchy subnet NameCert obligations)**.** *For an accepted* $\mathsf{CofinalCauchySubnet}^{\uparrow}$ *carrier* $Q = (I, J, \phi, M, A, T, K, N, S, R, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchySubnet}^{\uparrow}}$ *surface consists exactly of carrier admission, source directed-index exposure, subnet directed-index exposure, monotone reindexing exposure, Moore–Smith source-net exposure, Cauchy common-window exposure, cofinal-tail coverage, Cauchy-subnet compatibility, NetConvergence handoff, StreamName scheduling, RegSeqRat readback, terminal Real seal exposure, componentwise transport, displayed route replay, provenance, and local naming.*

Proof. Carrier admission projects the finite packet of Definition 6656.1. The source and subnet directed rows are I, J, ϕ, while M exposes the finite Moore–Smith source-net observations.

The Cauchy and cofinal rows are A, T, K. The row A stores the common-window Cauchy evidence, T answers each displayed source-tail request by a subnet tail, and K records that the subnet observations inherit the displayed Cauchy window along that cofinal route.

The convergence and Real-facing rows are then N, S, R, E. A $\mathsf{NetConvergence}^{\uparrow}$ consumer reads only after the cofinal Cauchy-subnet route is present, and a $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal only after the finite StreamName and RegSeqRat rows have been displayed.

Finally H, C, P, L transport, replay, source, and name exactly the same route. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations and leave no place for the excluded selection, quotient, choice, compactness, host-equality, or completeness principles. □

Theorem 6656.3 (Cofinal-tail stability)**.** *Every completion-facing read from an accepted* $\mathsf{CofinalCauchySubnet}^{\uparrow}$ *packet factors through the finite cofinal-tail route*

$$I, J, \phi, M, A \longmapsto T, K \longmapsto N, S, R, E.$$

Thus replacing a displayed source-tail request by a later displayed source tail changes only the finite witness chosen inside T; the downstream Cauchy-subnet compatibility row, NetConvergence handoff, StreamName windows, RegSeqRat readback, and terminal Real seal remain attached to the same carrier route.

Proof. Begin with the carrier inventory of Definition 6656.1. The only row that answers source-tail requests by subnet data is T, and T is read after the directed and source-net rows I, J, ϕ, M.

The Cauchy evidence is not recomputed downstream. The row K reads the common-window evidence in A through the same cofinal ledger T, so a later displayed source tail only asks T for a later displayed subnet-tail witness.

The rows N, S, R, E are downstream of T, K. Therefore convergence-facing and Real-facing consumers preserve the same cofinal Cauchy subnet route after any displayed tail refinement.

The structural rows H, C, P, L transport, replay, source, and name that route without adding a selected subnet, quotient-net equality, countable-choice coordinate, compactness principle, host equality, or completeness claim for $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: If a completion-facing consumer reads the $\mathsf{Real}^{\uparrow}$ seal of a purported $\mathsf{CofinalCauchySubnet}^{\uparrow}$ packet before displaying the cofinal-tail ledger T and the Cauchy-subnet compatibility row K, it is not reading the carrier of Definition 6656.1.

Independence witness: moore-smith-net, cauchynet, netconvergence, streamname, regseqrat, real: $\mathsf{CofinalCauchySubnet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source and subnet directed rows, monotone reindexing, source Cauchy evidence, cofinal-tail coverage, convergence handoff, finite observation windows, regular readback, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalCauchySubnet}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalCauchySubnet}^{\uparrow}$ packet is generated from source and subnet directed-index rows, a monotone reindexing row, a MooreSmithNet source row, CauchyNet common-window evidence, a cofinal-tail ledger, Cauchy-subnet compatibility, a NetConvergence handoff, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalCauchySubnet}^{\uparrow}}$, seed)

Scope closed: Definition 6656.1, Definition 6656.2, and Definition 6656.3 seed the finite cofinal Cauchy subnet carrier for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofinalCauchySubnet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected subnets, quotient-net equality, countable choice, ambient compactness, host equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and subnet directed-index exposure, monotone reindexing exposure, Cauchy-window inheritance, cofinal-tail coverage, convergence handoff, finite readback, terminal seal ordering, transport preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CofinalCauchySubnetUp

Chapter 6657

A Concrete Naming Certificate for $\mathsf{SemiDecidableRealCut}^{\uparrow}$

Following Bishop and Troelstra constructive analysis, the $\mathsf{SemiDecidableRealCut}^{\uparrow}$ packet records one-sided lower-cut evidence as finite observable data. It sits on the L10 real-name boundary after the finite observation windows of chapter 126, the regular rational readback of chapter 301, the one-sided lower ledger of chapter 1446, the located real route of **??**, and the terminal seal of chapter 38. The packet names a semi-decidable lower-cut interface: a consumer may inspect finite positive evidence that a rational lies below the represented real, but it receives no Dedekind completeness principle, quotient equality of real names, trichotomy, selected supremum, or countable-choice enumeration.

Definition 6657.1 (Semi-decidable real cut carrier)**.** *A* $\mathsf{SemiDecidableRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, L, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected precision requests and rational lower-cut queries.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
3. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *lower-observation ledger. Each accepted entry records a rational or dyadic candidate together with finite evidence that the candidate is admitted by the lower side.*
4. *Q is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, used only when the lower evidence is checked against a located enclosure already visible through W and R.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, consumed only after the stream, regular-rational, lower-ledger, and located-comparison rows are displayed.*
6. *H records componentwise* hsame *transport for W, R, L, Q, E and the structural rows.*
7. *C records displayed* Cont *replay of the same lower-cut query route.*
8. *P is the* Pkg *provenance over* $\mathsf{SemiDecidableRealCut}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SemiDecidableRealCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $W, R, L, Q, E, H, C, P, N$. It has no coordinate for an upper cut, quotient real equality, trichotomy, selected witnesses for all positive entries, a Dedekind-completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6657.2 (Semi-decidable real cut obligation surface)**.** *For an accepted* $\mathsf{SemiDecidableRealCut}^{\uparrow}$ *carrier* $S = (W, R, L, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SemiDecidableRealCut}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window visibility, regular-rational readback, lower-ledger admission, located-comparison compatibility, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise transport, displayed replay, provenance, and local naming. These obligations are all read through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LowerReal}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the structural rows H, C, P, N *retained.*

Proof. Carrier admission is the tuple displayed in Definition 6657.1. The only observation entry point is the finite window row W, and every regular readback is attached to that row through R.

The lower-cut evidence is the row L. It records positive lower information as finite admitted entries, while the located comparison row Q can only compare those entries against the already displayed located-real route.

The terminal seal E is downstream of W, R, L, Q. Transport and replay through H and C preserve the same finite route, and P, N source and name it. Since no carrier row stores an upper cut, quotient equality, trichotomy, selected global enumeration, Dedekind completeness, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6657.3 (Semi-decidable real cut window stability)**.** *Let* $S = (W, R, L, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{SemiDecidableRealCut}^{\uparrow}$ *carrier. Replacing a displayed lower-cut query by a later displayed finite window changes only the inspected entries of* W *and the corresponding regular-rational readback in* R*; the admitted lower-ledger row* L*, located comparison row* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise transport* H*, replay* C*, provenance* P*, and local name* N *remain attached to the same semi-decidable cut route. Thus window refinement does not create Dedekind completeness, quotient equality of real names, trichotomy, a selected dense enumeration, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite observation row W. A later displayed window is still a finite $\mathsf{StreamName}^{\uparrow}$ read, so the corresponding regular-rational data are obtained by rereading R along the same schedule discipline.

The lower evidence is not recomputed by a host predicate. It remains the displayed lower ledger L, and any located check must use the visible Q row before the terminal seal E is consumed.

Finally H, C, P, N keep the refined window in the same carrier. Componentwise transport preserves the rows, replay repeats the displayed route, provenance records the same dependencies, and the local name remains the same NameCert surface. No excluded completeness, quotient, trichotomy, selection, or choice principle is introduced by refining the finite window. □

Falsifiable prediction: A purported $\mathsf{SemiDecidableRealCut}^{\uparrow}$ consumer that reads a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying W, R, L, Q is not reading the carrier of Definition 6657.1.

Independence witness: streamname, regseqrat, lowerreal, locatedreal, real: $\mathsf{SemiDecidableRealCut}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite lower-cut query windows, regular readback, one-sided lower evidence, located comparison compatibility, terminal sealing, componentwise transport, displayed replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SemiDecidableRealCut}^{\uparrow}}$.

Constructive story: A $\mathsf{SemiDecidableRealCut}^{\uparrow}$ packet is generated from finite StreamName query windows, a RegSeqRat readback row, a LowerReal one-sided evidence ledger, a LocatedReal comparison row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SemiDecidableRealCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6657.1, Definition 6657.2, and Definition 6657.3 seed the finite semi-decidable lower-cut interface for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SemiDecidableRealCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient equality of real names, trichotomy, selected suprema, selected dense enumerations, countable choice, Lean verification, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, lower-ledger admission, located-comparison compatibility, window stability, terminal seal ordering, componentwise transport, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SemiDecidableRealCutUp.

Chapter 6658

A Concrete Naming Certificate for $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ packet records the finite route by which bounded real-sequence data, a monotone direction, and a cofinal index schedule are presented before a Real or Completion consumer asks for subsequence extraction. It reads the bounded source of chapter 3750, the cofinal index witness of chapter 405, the monotone selected-window surface of chapter 1405, the finite observation discipline of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a finite extraction ledger for Bolzano–Weierstrass selector and compact interval subsequence routes without selecting an infinite host subsequence, quotienting stream representatives, adding countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6658.1 (Cofinal monotone subsequence carrier)**.** *A* $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, D, I, C, W, R, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the bounded real-sequence source row, read from the accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *surface.*
2. *D is the monotone direction row, recording whether the inspected windows are increasing or decreasing and storing only displayed comparison data.*
3. *I is the cofinal index schedule, read through* $\mathsf{CofinalSubsequence}^{\uparrow}$ *and exposing finite threshold-to-index witnesses.*
4. *C is the finite window row, naming the inspected source windows and selected subsequence windows used by the extraction request.*
5. *W is the monotone-window ledger, read through* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *and tied to the bounded source row and cofinal schedule.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after B, D, I, C, W have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular-rational readback has been exposed.*

8. *H records componentwise* hsame *transport for the bounded source, direction, cofinal schedule, finite windows, monotone ledger, readback, Real seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay of the same bounded cofinal monotone route.*
10. *P is the* Pkg *provenance over* $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$, $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalMonotoneSubsequence}^{\uparrow}}$ *row.*

The carrier exports only these rows. It has no coordinate for a selected infinite subsequence, quotient stream equality, host trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6658.2 (Cofinal monotone subsequence extraction ledger)**.** *For every accepted* CofinalMonotoneSubseque *carrier $M = (B, D, I, C, W, R, E, H, T, P, N)$, every extraction-facing read must first display the bounded source row B, the direction row D, the cofinal index schedule I, the finite windows C, and the monotone-window ledger W. The schedule I answers each inspected source threshold by a later selected window, while D and W record the monotone comparisons for that displayed window. No consumer can enlarge this finite ledger into a selected infinite subsequence, quotient stream representative, countable-choice functional, or ambient compactness principle.*

Proof. Project the carrier of Definition 6658.1. The only source row is B, the only direction row is D, and the only cofinal schedule row is I.

The finite windows are then read through C and W. The row C records which source and selected windows are inspected, while W supplies the monotone comparisons inside those windows and keeps them tied to the direction row.

The structural rows H, T, P, N preserve and replay this route without adding another selector coordinate. Since the carrier has no selected infinite subsequence, quotient stream representative, countable-choice functional, or compactness row, the extraction ledger is exhausted by the displayed finite data. □

Theorem 6658.3 (Cofinal monotone subsequence Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ *packet factors through*

$$B, D, I, C, W \longmapsto R \longmapsto E$$

together with H, T, P, N. Thus Bolzano–Weierstrass selector and compact interval subsequence consumers may reuse the bounded source, cofinal schedule, monotone-window ledger, regular-rational readback, and terminal Real seal, but they receive no independent completeness theorem, selected limit, quotient real equality, host subsequence selector, or countable-choice principle.

Proof. Begin with Definition 6658.2. It places the bounded source, direction, cofinal schedule, finite windows, and monotone ledger before any readback row can be consumed.

The regular-rational readback row R is downstream of that extraction ledger, and the terminal Real seal E is downstream of R. Therefore a Real-facing consumer sees only the displayed route from finite windows to regular-rational readback and then to the Real seal.

Transport, replay, provenance, and local naming preserve the same packet. The excluded rows are absent from the carrier, so the readback cannot become a completeness principle, selected limit, quotient equality, host subsequence selector, or countable-choice rule. □

Falsifiable prediction: Any Bolzano–Weierstrass selector or compact interval subsequence consumer that reads the $\mathsf{Real}^{\uparrow}$ seal of a purported $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ packet before displaying B, D, I, C, W is not reading the carrier of Definition 6658.1.

Independence witness: bounded-monotone-subsequence, cofinalsubsequence, monotonesubsequence, streamname, regseqrat, real: $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded source row, a monotone direction row, a cofinal index schedule, finite selected windows, a monotone-window ledger, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalMonotoneSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalMonotoneSubsequence}^{\uparrow}$ packet is generated from bounded real-sequence data, a monotone direction row, a cofinal index schedule, finite selected windows, a monotone-window ledger, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalMonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6658.1, Definition 6658.2, and Definition 6658.3 seed the finite cofinal monotone subsequence route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalMonotoneSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite subsequences, quotient stream equality, host trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exposure, cofinal schedule coverage, monotone-window extraction, regular-rational readback, terminal seal ordering, transport preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CofinalMonotoneSubsequenceUp.

Chapter 6659

A Concrete Naming Certificate for $\mathsf{RegularExpression}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{RegularExpression}^{\uparrow}$ packet records the finite syntax side of regular-language presentation as BEDC NameCert data. It sits beside the accepted-word surface of chapter 351, the displayed finite control of chapter 2364, the canonical deterministic consumer of chapter 6188, and the grammar boundary in chapter 356. The packet names empty, literal, union, concatenation, and Kleene-star ledgers before any consumer may read a host regular-expression oracle, a language quotient, or an unrestricted grammar.

Definition 6659.1 (RegularExpression carrier)**.** *A* $\mathsf{RegularExpression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (\Sigma, E, L, U, C, K, H, T, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row.*
2. E *is the empty-language and empty-word syntax row, distinguishing the two nullary constructors as finite tags.*
3. L *is the literal-row ledger, naming each alphabet symbol used by a literal syntax leaf.*
4. U *is the union ledger, recording the two displayed subexpressions joined by each union constructor.*
5. C *is the concatenation ledger, recording the ordered pair of displayed subexpressions joined by each product constructor.*
6. K *is the Kleene-star ledger, recording the single displayed subexpression whose finite iteration surface is being named.*
7. H *records componentwise* hsame *transport for alphabet, empty, literal, union, concatenation, star, automaton handoff, replay, provenance, and local naming rows.*
8. T *records displayed* Cont *replay through the finite syntax tree and through the Thompson-style finite-automaton handoff route.*
9. P *is the* Pkg *provenance over* $\mathsf{RegularExpression}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{MinimalDfa}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularExpression}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, E, L, U, C, K, H, T, P, N$. It has no coordinate for a host equivalence test on regular expressions, an infinite language quotient, a nondeterministic branch selector, or a context-free grammar derivation tree.

Lean4 (statement) `BEDC.Derived.RegularExpressionUp`

Theorem 6659.2 (RegularExpression NameCert obligations)**.** *For every accepted* $\mathsf{RegularExpression}^{\uparrow}$ *carrier*

$$R = (\Sigma, E, L, U, C, K, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularExpression}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, empty-constructor exposure, literal-symbol exposure, union ledger replay, concatenation ledger replay, Kleene-star ledger replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may project the displayed syntax rows and their replay ledgers, but it obtains no host regular-expression equivalence oracle, language quotient, hidden automaton state, or unrestricted grammar production surface.*

Proof. Project the carrier of Definition 6659.1. The rows Σ, E, L expose the alphabet, nullary syntax constructors, and literal leaves before any compound syntax can be read.

The union, concatenation, and Kleene-star constructors are then read through U, C, K. Each row stores only finite pointers to displayed subexpressions, so the syntax ledger closes by structural descent through the displayed tree.

The rows H, T, P, N transport, replay, source, and name the same packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations and leave host equivalence, language quotienting, hidden automata, and context-free derivation data outside the packet. □

Theorem 6659.3 (RegularExpression Kleene automaton handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, *or* $\mathsf{MinimalDfa}^{\uparrow}$ *consumer that reads the Kleene construction from an accepted* $\mathsf{RegularExpression}^{\uparrow}$ *packet first reads the displayed alphabet row, empty and literal rows, union ledger, concatenation ledger, Kleene-star ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The handoff to the finite-automaton siblings exports only a finite syntax-to-automaton construction route, while the* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *sibling may read only the regular-expression syntax boundary as a proper sub-surface of grammar syntax.*

Proof. Begin with an accepted carrier $R = (\Sigma, E, L, U, C, K, H, T, P, N)$. Structural replay over the displayed syntax tree reads empty and literal leaves from E, L, then reads union and concatenation nodes from U, C.

For a Kleene-star node, the row K supplies the single displayed subexpression whose finite iteration route is being handed to the automaton construction. The Cont row T records the finite replay path used by the finite-automaton consumer; it does not choose an external nondeterministic branch outside the displayed construction data.

The regular-language and minimal-DFA consumers may subsequently read the resulting finite presentation through their own chapters. This route does not prove arbitrary regular-expression equivalence, does not identify host language quotients, and does not turn the packet into a context-free derivation system. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularExpression}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularExpression}^{\uparrow}$ packet is generated from a finite alphabet row, empty and literal syntax rows, union, concatenation, and Kleene-star ledgers, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RegularExpression}\uparrow}$, seed)

Scope closed: Definition 6659.1, Definition 6659.2, and Definition 6659.3 seed the finite syntax-side certificate for RegularLanguageUp, FiniteAutomatonUp, MinimalDfaUp, and ContextFreeGrammarUp consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularExpression}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.RegularExpressionUp`

Bridge status: none

Not claimed: This chapter does not close regular-expression equivalence, host language quotients, nondeterministic branch selection, grammar derivation completeness, theorem-checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constructor-ledger replay, Kleene-star automaton handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularExpressionUp.

Chapter 6660

A Concrete Naming Certificate for $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$

Following Bourbaki and Kelley on complete metric spaces and Bishop–Bridges on explicit Cauchy convergence, the $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ packet records the finite route by which displayed complete-metric Cauchy data may be consumed as $\mathsf{SequentialClosure}^{\uparrow}$ evidence. It sits after the metric-space surface of chapter 4230, the Cauchy witness rows of chapter 333, the finite sequence windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the sequential-closure packet of chapter 1571. The chapter names a finite bridge from complete metric convergence data to sequential closure; it is not host completeness, quotient stream equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6660.1 (Complete metric sequential-closure carrier)**.** *A* $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, C, S, Q, L, W, R, E, J, H, K, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the visible distance, tolerance, and neighbourhood comparison data.*
2. *C is the complete-metric Cauchy row, naming the Cauchy requests, finite modulus windows, and convergence-readiness data admitted by the source metric presentation.*
3. *S is the displayed subset or source row whose admitted sequence entries form the sequential-closure source.*
4. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *sequence row, carrying the finite observation windows consumed by closure requests.*
5. *L is the candidate limit row supplied by the complete-metric Cauchy data, without asserting host completion or quotient equality.*
6. *W is the finite window ledger aligning Cauchy requests in C with sequence entries in Q.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for tolerance and distance data visible in M, C, Q, W.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback row R is displayed.*

9. *J is the* $\mathsf{SequentialClosure}^{\uparrow}$ *evidence row obtained from S, Q, L, W, R, E.*
10. *H records componentwise* hsame *transport for the metric, Cauchy, source, sequence, limit, window, readback, Real seal, sequential-closure evidence, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a closure consumer through $M, C, S, Q, L, W, R, E, J, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricSequentialClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for host completeness of the metric space, choice of arbitrary subsequences, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6660.2 (Complete metric sequential-closure NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ *carrier $G = (M, C, S, Q, L, W, R, E, J, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricSequentialClosure}^{\uparrow}}$ *surface consists exactly of carrier admission, complete-metric Cauchy exposure, sequence-window coverage, limit-row alignment, RegSeqRat readback, Real seal admission,* $\mathsf{SequentialClosure}^{\uparrow}$ *handoff, componentwise transport, displayed replay, package provenance, local naming, and nonescape from host completeness, choice, quotient stream equality, and compactness.*

Proof. Project the carrier. Rows M, C, S, Q, L, W expose the metric presentation, complete-metric Cauchy data, source row, sequence windows, candidate limit, and window alignment ledger.

The analytic handoff is downstream of those rows. The readback row R records the regular rational tolerance data, the Real row E seals the accepted distance comparison, and the sequential-closure row J repackages S, Q, L, W, R, E as $\mathsf{SequentialClosure}^{\uparrow}$ evidence.

The structural rows H, K, P, N then transport, replay, source, and locally name the same finite route. Since the carrier includes no host completeness, selected subsequence, quotient stream equality, open-cover compactness, or ambient Real completeness coordinate, the displayed rows exhaust the NameCert obligation surface. □

Theorem 6660.3 (Complete metric sequential-closure limit seal)**.** *Every closure-facing read of an accepted* $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ *packet factors through*

$$M, C, S, Q, L, W \longmapsto R, E \longmapsto J \longmapsto H, K, P, N.$$

Thus the candidate limit row is sealed only after the complete-metric Cauchy windows have been aligned with displayed sequence windows, translated through $\mathsf{RegSeqRat}^{\uparrow}$, *sealed by* $\mathsf{Real}^{\uparrow}$, *and repacked as* $\mathsf{SequentialClosure}^{\uparrow}$ *evidence.*

Proof. Start with a closure-facing read of L. The carrier admits L beside the metric row M, complete-metric Cauchy row C, source row S, sequence row Q, and finite window ledger W.

The limit seal is then read through R and E. The regular readback row records the tolerance data visible in the metric and Cauchy windows, and the Real seal row certifies the accepted distance comparison only after those windows have been displayed.

Finally the row J repackages the resulting limit evidence as $\mathsf{SequentialClosure}^{\uparrow}$ data. Transport and replay through H, K, P, N preserve this order, so the consumer receives only the displayed finite seal route and not host completeness or a quotient completion. □

Theorem 6660.4 (Complete metric sequential-closure source nonescape)**.** *An accepted* CompleteMetricSequentialCl *packet exports no sequential-closure evidence unless the source row S, sequence row Q, candidate*

limit row L, finite window ledger W, regular readback row R, Real seal row E, and $\mathsf{SequentialClosure}^{\uparrow}$ *row J are all displayed in the same carrier. In particular, complete-metric Cauchy data alone cannot escape as host closure, selected limit, quotient stream equality, or open-cover compactness.*

Proof. Read the carrier rows. The complete-metric Cauchy row C is not an exported closure object by itself; it is tied to S, Q, L, W before any Real-facing or closure-facing evidence is available.

Apply the limit-seal route of Definition 6660.3. It forces every closure read through R, E, J after the metric and sequence windows have been exposed.

The nonescape claim follows from absence of coordinates. There is no row for host closure, selected limits, quotient stream equality, compactness, or ambient completeness, and H, K, P, N only preserve the displayed finite packet. Therefore source data cannot leave the carrier except through the stated sequential-closure route. □

Falsifiable prediction: Every accepted CompleteMetricSequentialClosureUp closure-facing read factors through a MetricSpace row, complete-metric Cauchy row, source row, StreamName sequence row, candidate limit row, finite window ledger, RegSeqRat readback, Real seal, SequentialClosure evidence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metricspace, cauchyfilter, streamname, regseqrat, real, sequentialclosure: $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ is not bijective to any listed sibling because it binds complete-metric Cauchy exposure, sequence-window alignment, limit sealing, regular readback, Real sealing, SequentialClosure handoff, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricSequentialClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricSequentialClosure}^{\uparrow}$ packet is generated from a MetricSpace row, complete-metric Cauchy row, source row, StreamName sequence row, candidate limit row, finite window ledger, RegSeqRat readback, Real seal, SequentialClosure evidence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricSequentialClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6660.1, Definition 6660.2, Definition 6660.3, and Definition 6660.4 seed the finite route from complete metric Cauchy data to sequential-closure evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricSequentialClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, choice, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric Cauchy exposure, sequence-window coverage, RegSeqRat readback, Real seal admission, SequentialClosure handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompleteMetricSequentialClosureUp.

Chapter 6661

A Concrete Naming Certificate for $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$

Following Bishop–Bridges constructive arithmetic, the $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ packet records the finite operation-input ledger that must be displayed before arithmetic on regular rational sequences is allowed to feed a Real-facing seal. It sits between the source rows of chapter 301, the finite-window service of chapter 126, the dyadic operation budget of chapter 291, and the terminal seal of chapter 38. The ledger is pre-theorem input data for the arithmetic packets of chapter 417 and chapter 461; it does not claim RealAlgOrder closure or full real arithmetic.

Definition 6661.1 (RegSeqRat operation ledger carrier)**.** *A* $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, B, W_A, W_B, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational sequence data may be used by the operation request.*
2. *W_A and W_B are synchronized* $\mathsf{StreamName}^{\uparrow}$ *window rows, exposing the same precision window for the two source reads.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation budget, recording the finite endpoint addition, multiplication, comparison, and error-allocation rows available to the operation request.*
4. *Q is the output regularity row, requiring the proposed operation output to remain a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any Real-facing consumer reads it.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, reached only after A, B, W_A, W_B, D, Q have all been displayed.*
6. *H records componentwise* hsame *transport for the two sources, synchronized windows, dyadic budget, output regularity row, seal boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an operation request through $A, B, W_A, W_B, D, Q, E, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatOperationLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through $A, B, W_A, W_B, D, Q, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real arithmetic, order trichotomy, RealAlgOrder closure, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6661.2 (RegSeqRat operation ledger NameCert obligations)**.** *For every accepted* $\mathsf{RegSeqRatOperationLed}$ *carrier* $O = (A, B, W_A, W_B, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatOperationLedger}^{\uparrow}}$ *surface consists exactly of two-source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, synchronized* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget accounting, output regularity exposure through* Q*, the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every arithmetic consumer must read the finite route*

$$A, B, W_A, W_B, D, Q \longmapsto E \longmapsto H, C, P, N$$

before it may hand the request to $\mathsf{RegularCauchySum}^{\uparrow}$*,* $\mathsf{RegularCauchyProduct}^{\uparrow}$*, or a Real-facing seal.*

Proof. Project the carrier of Definition 6661.1. The only source rows are A and B, and the only observable schedules are the synchronized window rows W_A, W_B.

Read the operation budget next. The dyadic row D is the sole coordinate for endpoint arithmetic, comparison, and error allocation, while Q is the only row that records output regularity before a downstream seal is available.

Finally route the accepted read through E, H, C, P, N. These rows seal, transport, replay, source, and name the same finite ledger, so they cannot add quotient streams, selected limits, host real arithmetic, order trichotomy, RealAlgOrder closure, or ambient completeness. □

Theorem 6661.3 (RegSeqRat operation ledger regularity preservation)**.** *Let* $O = (A, B, W_A, W_B, D, Q, E, H, C, P$ *be an accepted* $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ *carrier. Any operation request that is routed through the ledger preserves regular rational sequence visibility in the following sense: the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources* A, B*, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows* W_A, W_B*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget* D *must be displayed before the output regularity row* Q *is read, and* Q *must be read before the* $\mathsf{Real}^{\uparrow}$ *seal boundary* E *is available. Thus the ledger can feed* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *as pre-theorem input data, but it does not by itself prove real addition, real multiplication, ordered-field laws, quotient-stream equality, or completeness.*

Proof. Start with source visibility. The carrier exposes exactly two regular rational sequence rows and their synchronized finite windows, so a consumer cannot manufacture an operation output without first reading A, B, W_A, W_B.

Use the dyadic budget row D for the finite operation request. Endpoint arithmetic and error allocation are therefore recorded before output regularity is inspected.

The output row Q is downstream of those inputs and upstream of the seal row E. Hence any Real-facing read sees a regular rational output only through the displayed ledger. The packet can be consumed by the sum and product chapters as operation-input data, but the refused real arithmetic and order principles have no carrier coordinate here. □

Falsifiable prediction: Every accepted $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ operation read factors through two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ operation budget, an output regularity row, a $\mathsf{Real}^{\uparrow}$ seal boundary, hsame transport, Cont replay, Pkg provenance, and local naming before $\mathsf{RegularCauchySum}^{\uparrow}$ or $\mathsf{RegularCauchyProduct}^{\uparrow}$ consumes it.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-sum, regular-cauchy-product: $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source regular names, synchronized observation windows, dyadic operation budgeting, output regularity, a Real seal boundary, structural transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRatOperationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RegSeqRatOperationLedger}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ operation budget, an output regularity row, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegSeqRatOperationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6661.1, Definition 6661.2, and Definition 6661.3 seed the finite operation-input ledger for regular rational sequence arithmetic before Real-facing operation descent.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegSeqRatOperationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealAlgOrder, ordered-field laws, complete real arithmetic, quotient-stream equality, selected limits, host real arithmetic, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, two-source regularity, synchronized window exactness, dyadic operation-budget accounting, output regularity preservation, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RegSeqRatOperationLedgerUp.

Chapter 6662

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyDini}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ packet records a finite Cauchy–Dini modulus route from monotone real-valued approximation rows to a terminal real-name seal. It sits beside the compact finite-modulus surface of chapter 2299, the Cauchy-modulus boundary of chapter 1423, the constructive Dini modulus packet of chapter 3454, the regular-rational readback of chapter 301, and the terminal real-name chapter of chapter 38. The packet names only the finite route by which compact-window rows, monotone Dini rows, Cauchy modulus rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ sealing are displayed; it does not assert host completeness, quotient equality of real names, open-cover compactness, or a selected limiting function.

Definition 6662.1 (Constructive Cauchy-Dini carrier)**.** *A* $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, M, W, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the stream-source row, listing the finite observation schedule for the monotone approximation family.*
2. *M is the monotone comparison row, recording the Dini-side order comparisons visible in the inspected windows.*
3. *W is the finite compact-window row, read through the compact uniform modulus surface and carrying only displayed finite cover or finite-net data.*
4. *K is the Cauchy modulus row, recording the finite tail budget that turns the monotone compact-window comparison into a Cauchy read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same scheduled windows and Cauchy budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after S, M, W, K, R have all been displayed.*
7. *H records componentwise* hsame *transport for the stream source, monotone comparison, compact window, Cauchy budget, readback, seal, and structural rows.*
8. *C records displayed* Cont *replay of the same finite Cauchy–Dini route.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCauchyDini}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyDini}^{\uparrow}}$ *row.*
The carrier has no coordinate for host completeness, quotient equality of functions or reals, open-cover compactness, a selected limit function, countable choice, or a non-displayed uniform convergence theorem.

Theorem 6662.2 (Constructive Cauchy-Dini NameCert obligations)**.** *For the carrier of Definition 6662.1, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyDini}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-source visibility, monotone comparison stability, finite compact-window exposure, Cauchy modulus ordering,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from the tuple $D = (S, M, W, K, R, E, H, C, P, N)$. The stream source, monotone comparison, compact window, Cauchy budget, regular readback, and terminal seal are all displayed rows of the same finite packet.

The analytic handoff is ordered. A consumer must first read the inspected stream windows in S, then the monotone comparison row M, then the compact finite-window row W, and only then the Cauchy modulus row K.

The readback and seal do not add hidden data. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the same window and budget, while E seals the resulting real-facing row after R is visible. The structural rows H, C, P, N transport, replay, source, and name that route without importing host completeness, quotient equality, open-cover compactness, selected limits, choice, or an unrecorded convergence theorem. □

Theorem 6662.3 (Constructive Cauchy-Dini finite-modulus handoff)**.** *Every accepted* $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ *read that reaches a* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$S, M, W, K, R, E, H, C, P, N.$$

In particular, the compact finite-modulus rows and monotone Dini rows must be displayed before the Cauchy modulus is read, and the $\mathsf{RegSeqRat}^{\uparrow}$ *readback must be displayed before the terminal real-name seal. The handoff exports no host uniform-limit theorem, quotient equality of real names, open-cover compactness proof, selected limiting function, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 6662.2. They place the stream source, monotone comparison, compact finite window, and Cauchy modulus in one displayed packet before any readback row is available.

Next read the regular-rational handoff. The row R is attached to the same scheduled windows and the same Cauchy budget, so a consumer cannot detach it from S, M, W, K.

Finally consume the terminal seal. The row E is downstream of R, and H, C, P, N keep the route componentwise transported, replayed, sourced, and named. Since the excluded host objects have no carrier coordinate, the $\mathsf{Real}^{\uparrow}$ consumer receives only the finite Cauchy–Dini modulus handoff. □

Falsifiable prediction: A purported $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first displaying S, M, W, K, R is not reading the carrier of Definition 6662.1.

Independence witness: compact-uniform-modulus, realcauchymodulus, constructive-dini-modulus, regseqrat, real: $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries stream source, monotone Dini comparison, compact finite-window, Cauchy modulus, regular-rational readback, terminal seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyDini}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyDini}^{\uparrow}$ packet is generated from finite stream-source observations, monotone Dini comparison rows, compact finite-modulus windows, a Cauchy modulus budget, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyDini}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6662.1, Definition 6662.2, and Definition 6662.3 seed the finite Cauchy–Dini modulus route over compact-window, Cauchy-modulus, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyDini}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host uniform-limit completeness, quotient equality of real names, open-cover compactness, selected limiting functions, countable choice, Lean verification, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, monotone comparison stability, compact finite-window exposure, Cauchy modulus ordering, regular-rational readback, terminal seal ordering, transport preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.ConstructiveCauchyDiniUp.

Chapter 6663

A Concrete Naming Certificate for ClosedParallelSubstitutionBudget$^{\uparrow}$

The ClosedParallelSubstitutionBudget$^{\uparrow}$ packet realizes the substitution-compatibility boundary isolated by chapter 9378 as finite NameCert data. It sits beside the compiler-facing closed substitution route of chapter 540, the primitive closed-substitution boundary of chapter 538, the confluence audit packet of chapter 735, the closed parallel diamond premise ledger of chapter 2325, and the substitution-sensitive confluence boundary of chapter 3791. The packet records the finite budget rows that must be visible before a closed parallel-diamond handoff can consume substitution compatibility; it does not claim full closed Church–Rosser, unrestricted open substitution commutation, subject reduction, evaluator soundness, quotient equality, or kernel consistency.

Definition 6663.1 (Closed parallel substitution budget carrier)**.** *A* ClosedParallelSubstitutionBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, U, L, S, D, J, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed source and target term-window row, recording only closed MetaCIC terms inspected by the budget.*
2. *U is the closed substitution compiler row, read through* ClosedTermSubstitutionCompiler$^{\uparrow}$ *and carrying the checked closed shift and substitution slices available to the packet.*
3. *L is the primitive closed-substitution boundary row, read through* ClosedSubstitutionBoundary$^{\uparrow}$ *and retaining the source-closedness, value-closedness, shift, and substitute rows.*
4. *S is the substitution-budget ledger, listing the finite residual-substitution compatibility checks required by the closed parallel branch under inspection.*
5. *D is the closed parallel diamond premise row, read through* ClosedParallelDiamondPremiseLedger$^{\uparrow}$ *and not promoted to a full diamond theorem.*
6. *J is the confluence-audit handoff row, reading* MetaCICConfluenceAuditPacket$^{\uparrow}$ *and* MetaCICConfluenceSubstituti *as boundary packets for the same finite budget.*
7. *H records componentwise* hsame *transport for T, U, L, S, D, J and the structural rows.*
8. *C records displayed* Cont *replay from closed term windows to compiler rows, boundary rows, substitution-budget rows, premise-ledger rows, and confluence-audit handoff rows.*

9. *P is the* Pkg *provenance over* ClosedParallelSubstitutionBudget$^{\uparrow}$, ClosedTermSubstitutionCompiler$^{\uparrow}$, ClosedSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceAuditPacket$^{\uparrow}$, ClosedParallelDiamondPremiseLedger$^{\uparrow}$, MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{ClosedParallelSubstitutionBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for full closed Church–Rosser, unrestricted open-term substitution commutation, subject reduction, evaluator soundness, quotient conversion equality, kernel consistency, or a hidden parallel-diamond theorem.

Theorem 6663.2 (Closed parallel substitution budget obligations)**.** *For the carrier of Definition 6663.1, the local* NameCert$_{\text{ClosedParallelSubstitutionBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, closed term-window visibility, closed compiler handoff, primitive closed-substitution boundary exposure, finite residual-substitution budget listing, closed parallel diamond premise retention, confluence-audit handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission projects the tuple $B = (T, U, L, S, D, J, H, C, P, N)$. The closed term windows, compiler row, primitive boundary row, substitution-budget ledger, premise ledger, and confluence-audit handoff are all displayed in one finite packet.

The substitution budget is read before the diamond handoff. The compiler row U and primitive boundary row L expose the closed substitution material, and the ledger S records the finite residual-substitution checks that are available to the inspected closed parallel branch.

The premise row D remains a premise row. It may be handed to confluence-audit consumers through J, transported through H, replayed through C, sourced by P, and named by N, but those structural rows do not create full closed Church–Rosser, open substitution commutation, subject reduction, evaluator soundness, quotient equality, kernel consistency, or a hidden diamond theorem. □

Theorem 6663.3 (Closed parallel substitution budget diamond handoff)**.** *Every accepted* ClosedParallelSubstitution *read that reaches a closed parallel-diamond or closed-star-confluence consumer factors through the finite route*

$$T, U, L, S, D, J, H, C, P, N.$$

The substitution-budget ledger S must be visible before the closed parallel diamond premise row D is consumed, and the confluence-audit handoff J may cite only the displayed budget and boundary rows. The handoff exports no full closed Church–Rosser theorem, unrestricted substitution compatibility, subject-reduction theorem, evaluator soundness, quotient conversion equality, kernel consistency, or hidden confluence oracle.

Proof. Start with the NameCert obligations of Definition 6663.2. They place the closed term windows, closed substitution compiler row, primitive boundary row, substitution-budget ledger, and closed parallel diamond premise in a single displayed packet.

Next consume the budget. The row S is the only place where residual-substitution compatibility checks are listed, so a diamond-facing consumer cannot bypass S on the way to D.

Finally route the handoff through J, H, C, P, N. The confluence audit packet and substitution-boundary packet are read as boundary packets for the same displayed budget. Since the excluded confluence, substitution, typing, evaluator, quotient, and kernel-correctness objects have no carrier coordinate, the handoff remains a finite premise-budget route. □

Falsifiable prediction: A purported $\mathsf{ClosedParallelSubstitutionBudget}^{\uparrow}$ consumer that exports a closed parallel diamond or closed Church–Rosser result without displaying T, U, L, S, D, J is not reading the carrier of Definition 6663.1.

Independence witness: closedtermsubstitutioncompiler, closedsubstitutionboundary, metacicconfluenceauditpacket, closedparalleldiamondpremiseledger, metacicconfluencesubstitutionboundary: no listed sibling jointly carries closed term windows, compiler slices, primitive closed-substitution rows, finite residual-substitution budget, closed parallel diamond premise retention, confluence-audit handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedParallelSubstitutionBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedParallelSubstitutionBudget}^{\uparrow}$ packet is generated from closed term-window rows, closed substitution compiler rows, primitive closed-substitution boundary rows, finite residual-substitution budget rows, closed parallel diamond premise rows, confluence-audit handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedParallelSubstitutionBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6663.1, Definition 6663.2, and Definition 6663.3 seed the finite substitution-budget route for closed parallel-diamond and confluence-audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedParallelSubstitutionBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, unrestricted substitution compatibility, subject reduction, evaluator soundness, quotient conversion equality, kernel consistency, Lean verification, bridge checking, or a hidden confluence theorem.

Upgrade path: Obligation closure requires checked carrier admission, closed compiler handoff, primitive boundary exposure, finite budget listing, premise retention, confluence-audit handoff, transport preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.ClosedParallelSubstitutionBudgetUp.

Chapter 6664

A Concrete Naming Certificate for $\mathsf{InitialTopology}^{\uparrow}$

Following Bourbaki and Kelley on initial topologies, the $\mathsf{InitialTopology}^{\uparrow}$ packet records the finite BEDC surface where a source carrier is named by the weakest displayed topology making a finite family of maps continuous. It is adjacent to the topology carrier of chapter 88 and the continuous-map handoff of chapter 130: the chapter names only finite inverse-image subbasis generation, finite intersection and union ledgers, transport, replay, provenance, and local naming.

Definition 6664.1 (Initial topology carrier)**.** *An* $\mathsf{InitialTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, I, T, F, B, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source carrier row whose points receive the generated topology;*
2. *I is the finite index row for the displayed family of target spaces;*
3. *T assigns to each index in I a* $\mathsf{Topology}^{\uparrow}$ *target row with its named open-row classifier;*
4. *F assigns to each index a displayed map from S to the corresponding target carrier;*
5. *B is the subbasic inverse-image ledger, listing the rows $F_i^{-1}(V)$ for target opens V named by T_i;*
6. *U is the finite intersection and finite union ledger generated from B, with entries named as source opens and no appeal to an arbitrary open-set powerset;*
7. *H records componentwise* hsame *transport for S, I, T, F, B, U and the local certificate rows;*
8. *C records displayed* Cont *replay, showing that each F_i reads source opens through B and U before it is consumed as continuous;*
9. *P is the* Pkg *provenance over* $\mathsf{InitialTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InitialTopology}^{\uparrow}}$ *row.*

The classifier compares two packets only through these rows. It has no coordinate for an ambient powerset topology on S, a choice-selected open basis, an infinite subbasis closure, or a host proof that a larger topology is available.

Theorem 6664.2 (Initial topology NameCert obligations)**.** *For an accepted* $\mathsf{InitialTopology}^{\uparrow}$ *carrier* $Q = (S, I, T, F, B, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{InitialTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the source row, finite index row, target topology rows, displayed map rows, subbasic inverse-image ledger, finite intersection and union ledger, componentwise transport, continuation replay, provenance, and local naming. Its classifier stability is generated by the same rows, and its exactness claim is limited to the statement that every displayed source open is generated from the inverse images in* B *by the finite ledger* U*.*

Proof. Project the carrier of Definition 6664.1. The source, index, target, map, subbasic, and finite-generation rows are the only rows from which the local certificate can read.

The subbasic rows are fixed before the generated source-open rows are read. Each entry of B is an inverse image of a target-open row along a displayed map in F, and each entry of U is obtained from finitely many such rows by the listed intersection and union constructors.

Transport and replay add no new open rows. The hsame row moves the displayed data componentwise, the Cont row replays the same inverse-image route, and the Pkg and NameCert rows source and name the finite packet. Thus no ambient powerset topology, selected basis, or infinite closure operation enters the obligation surface. □

Theorem 6664.3 (Initial topology continuous-map handoff)**.** *Let* $Q = (S, I, T, F, B, U, H, C, P, N)$ *be an accepted* $\mathsf{InitialTopology}^{\uparrow}$ *carrier. For each index in* I*, any downstream* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of the displayed map row* F_i *must first read the target topology row* T_i*, the inverse-image subbasis ledger* B*, the finite generation ledger* U*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*. The handoff exports continuity of the named maps only through the generated source topology and not through a detached ambient topology on* S*.*

Proof. Start with a request for the continuous-map row associated to F_i. Carrier admission exposes the corresponding target topology row T_i and the subbasic inverse-image entries in B before any continuity read is available.

The source topology read then passes through U. This ledger is finite and generated from the displayed inverse images, so the consumer receives exactly the source-open rows needed to replay the inverse-image test for F_i.

Finally apply the structural rows. The hsame row keeps the same source, target, map, and generated-open entries attached; the Cont row records the replay order; and P, N source and name the packet. Hence the $\mathsf{ContinuousMap}^{\uparrow}$ handoff is the finite initial-topology route, not an external topology assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{InitialTopology}^{\uparrow}}$**.**

Constructive story: An $\mathsf{InitialTopology}^{\uparrow}$ packet is generated from a source carrier, finite index row, target topology rows, displayed maps, inverse-image subbasis ledger, finite intersection and union ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InitialTopology}^{\uparrow}}$, seed)

Scope closed: Definition 6664.1, Definition 6664.2, and Definition 6664.3 seed the finite initial-topology route from displayed target opens to generated source opens and downstream continuous-map consumption.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InitialTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an ambient powerset topology, selected open basis, infinite subbasis closure, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, subbasic inverse-image exactness, finite generation, continuous-map handoff, transport replay, provenance, and consumer refusal for BEDC.Derived.InitialTopologyUp.

Chapter 6665

A Concrete Naming Certificate for $\mathsf{CompactGraphContinuity}^{\uparrow}$

Following Kelley and Bishop–Bridges, compact graph continuity is the finite certificate route by which a displayed closed-graph packet over a compact metric source yields a uniform-modulus handoff without appealing to a host quotient of functions or an external open-cover selection. The $\mathsf{CompactGraphContinuity}^{\uparrow}$ packet sits over the compact metric source of chapter 129, the continuous-map surface of chapter 130, and the tolerance-only consumer of chapter 369. It also reads the metric and real endpoint rows used by those chapters before exposing any graph-continuity conclusion.

Definition 6665.1 (Compact graph continuity carrier)**.** *A* $\mathsf{CompactGraphContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, \Gamma, M, Q, W, R, U, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including displayed finite covers, center histories, metric balls, and source-point classifier transport.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map whose graph is being inspected.*
3. *Γ is the graph row: it records finite source-target observations, graph membership readback, and graph classifier stability under the source and target classifiers.*
4. *M is the target metric row for graph-value comparisons and endpoint separation tests.*
5. *Q is the finite compact-cover query row that turns a source tolerance request into a displayed finite list of graph tests.*
6. *W is the closed-graph witness row, recording that every failed target separation is detected by one of the displayed graph tests.*
7. *R is the regular rational or dyadic comparison row used to read the graph tests as finite tolerance inequalities.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from K, F, Γ, M, Q, W, R.*
9. *H stores componentwise* hsame *transport for compact source rows, continuous-map rows, graph rows, metric endpoints, compact-cover queries, closed-graph witnesses, rational comparisons, uniform-modulus handoffs, replay, provenance, and local naming.*

10. *C stores displayed* Cont *replay from a requested tolerance through the rows $K, F, \Gamma, M, Q, W, R, H$ to the handoff row U.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactGraphContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactGraphContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host equality of maps, quotient equality of graphs, open-cover compactness outside K, a choice-selected graph neighbourhood, or ambient completeness of the target.

Lean4 ✓ `BEDC.Derived.CompactGraphContinuityUp.CompactGraphContinuityTasteGate single carrier alignment`

Theorem 6665.2 (Compact graph continuity NameCert obligations)**.** *For every accepted* CompactGraphContinuity *carrier*

$$G = (K, F, \Gamma, M, Q, W, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactGraphContinuity}^{\uparrow}}$ *obligations are exactly compact metric source exposure, ContinuousMap row exposure, finite graph membership readback, source-target graph classifier stability, target-metric separation detection, finite compact-cover query exactness, closed-graph witness exactness, rational tolerance readback, UniformModulus handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6665.1. The admitted packet exposes $K, F, \Gamma, M, Q, W, R, U, H, C, P, N$ and no other theorem-facing row.

The source part is K, F, Γ, M. These rows identify the compact metric source, the displayed continuous map, the finite graph observations, and the target metric comparisons consumed by graph separation tests.

The compactness and graph-closure part is Q, W, R. The query row names only finitely many compact-cover and graph tests for a tolerance request; the closed-graph witness row explains why a missing target neighbourhood is detected by those tests; and the rational row makes the resulting inequalities public.

Finally U, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for host map equality, quotient graph equality, external open-cover compactness, selected graph neighbourhoods, or ambient target completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6665.3 (Compact graph continuity uniform-modulus handoff)**.** *Every accepted* CompactGraphContinuity *packet exports a* $\mathsf{UniformModulus}^{\uparrow}$ *row only after the consumer has read the compact metric source of chapter 129, the displayed continuous-map row of chapter 130, the finite graph row, the target metric separation tests, the compact-cover query row, the closed-graph witness row, and the rational tolerance readback. Thus the handoff to chapter 369 is a finite graph/modulus obligation surface, not an ambient closed-graph theorem for arbitrary host maps.*

Proof. Start with an accepted carrier $G = (K, F, \Gamma, M, Q, W, R, U, H, C, P, N)$. A tolerance request reaches the replay row C, which exposes the compact metric source K, the continuous-map row F, the graph row Γ, and the target metric comparison row M.

The compact-cover query Q then reduces the request to finitely many displayed graph tests. The closed-graph witness W and the rational comparison row R certify those tests inside the same packet.

Only after this finite route is visible does the row U become a public $\mathsf{UniformModulus}^{\uparrow}$ handoff. Transport and provenance through H, P, N keep the route attached to the same local name, so a downstream consumer cannot replace it by host map equality, quotient graph equality, external open-cover compactness, selected neighbourhoods, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactGraphContinuity}^{\uparrow}$ read that exports a $\mathsf{UniformModulus}^{\uparrow}$ row must first expose the compact metric source, continuous-map row, finite graph row, target metric separation row, compact-cover query, closed-graph witness, rational tolerance readback, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric, real: CompactGraphContinuity is not bijective to any listed sibling because it jointly carries compact finite-cover data, one displayed continuous-map row, graph membership and graph-stability rows, target separation tests, compact-cover graph queries, closed-graph witnesses, rational tolerance readback, UniformModulus handoff, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactGraphContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactGraphContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6665.1, Definition 6665.2, and Definition 6665.3 seed the finite closed-graph-to-uniform-modulus route over compact metric, ContinuousMap, UniformModulus, Metric, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactGraphContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an ambient closed-graph theorem, quotient equality of graphs, host equality of maps, external open-cover compactness, selected graph neighbourhoods, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, continuous-map exposure, graph classifier stability, target separation detection, finite compact-cover query exactness, closed-graph witness exactness, rational tolerance readback, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactGraphContinuityUp.

Constructive story: A $\mathsf{CompactGraphContinuity}^{\uparrow}$ packet is generated from CompactMetric finite-cover rows, ContinuousMap graph rows, target Metric separation tests, compact-cover graph queries, closed-graph witnesses, rational tolerance readbacks, UniformModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6666

A Concrete Naming Certificate for MetaCICResidualCheckerNormalizationBudget$^{\uparrow}$

The MetaCICResidualCheckerNormalizationBudget$^{\uparrow}$ packet realizes the normalization audit map of chapter 9383 as finite checker, fuel, obstruction, transport, replay, provenance, and local naming rows for candidate-mediated strong normalization. It sits beside the frontier carrier of chapter 892, the candidate-mediated SN route of chapter 5163, the finite confluence window of chapter 6027, the candidate diamond budget of chapter 6552, and the typed normal-form scheduler of chapter 6617. The packet is a residual-checker budget surface; it is not an unconditional normalization theorem, residual confluence theorem, subject-reduction theorem, evaluator oracle, quotient syntax equality, or closed consistency proof.

Definition 6666.1 (MetaCIC residual-checker normalization budget carrier)**.** *A* MetaCICResidualCheckerNormaliza *carrier is a finite* BHist *packet*

$$B = (T, A, S, F, R, J, O, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the sealed closed typed term row inspected by the normalization audit.*
2. *A is the candidate-mediated strong-normalization evidence row, including candidate membership, closed-candidate evidence, Pi adequacy, application adequacy, and no-infinite-reduction projections.*
3. *S is the residual checker row, recording the finite residual windows, local residual-substitution checks, and confluence-boundary verdicts consumed before a normal endpoint is trusted.*
4. *F is the fuel row, storing the finite normalization budget, residual-checker budget, and replay-window budget for the current audit request.*
5. *R is the route row from candidate-mediated SN evidence to residual-checker inspection.*
6. *J is the finite join or refusal row, retaining local joins when the checker sees them and explicit obstruction data when it does not.*
7. *O is the obstruction row for missing candidate evidence, fuel exhaustion, failed residual substitution, failed local join, failed closedness transport, and failed endpoint seal.*
8. *X is the export row that exposes only the budgeted normalization result or its displayed obstruction.*

9. *H records componentwise* hsame *transport for the closed term, candidate evidence, residual checker, fuel, route, join/refusal row, obstruction, export row, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from an audit request through T, A, S, F, R, J, O, H to X.*
11. *P is the* Pkg *provenance over* MetaCICResidualCheckerNormalizationBudget$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, CandidateMediatedSN$^{\uparrow}$, FiniteWindowConfluence$^{\uparrow}$, MetaCICCandidateDiamondBudget$^{\uparrow}$, MetaCICTypedNormalFor BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, S, F, R, J, O, X, H, C, P, N$. It has no coordinate for evidence-free global SN, unrestricted residual confluence, subject-reduction transport, quotient syntax equality, hidden evaluator state, host reduction oracles, or closed consistency.

Theorem 6666.2 (MetaCIC residual-checker normalization budget obligations)**.** *For every accepted* MetaCICResidualCheckerNormalizationBudget$^{\uparrow}$ *carrier*

$$B = (T, A, S, F, R, J, O, X, H, C, P, N),$$

the local NameCert$_{\text{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}}$ *obligations are exactly carrier admission, sealed closed-term exposure, candidate-mediated SN evidence exposure, residual-checker window exposure, fuel-budget exactness, candidate-to-checker route replay, finite join or refusal exactness, obstruction retention, budgeted export nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6666.1. The accepted packet exposes exactly the closed term, candidate evidence, residual checker, fuel budget, route, join/refusal row, obstruction, export, transport, replay, provenance, and local naming rows.

The normalization side is T, A, F, R. A public read must display the sealed closed term, the candidate-mediated SN evidence, the finite fuel, and the route that carries that evidence to the checker.

The residual side is S, J, O. The checker sees only finite residual windows; it exports local joins when they are present and keeps failed residual substitution, failed local joins, failed closedness transport, and failed endpoint seals as visible obstruction rows.

Finally X, H, C, P, N export and preserve the same budgeted route. Since no carrier coordinate supplies evidence-free global SN, unrestricted residual confluence, subject-reduction transport, quotient syntax equality, hidden evaluator state, host reduction oracles, or closed consistency, the listed obligations exhaust the local NameCert surface. □

Theorem 6666.3 (MetaCIC residual-checker budget boundary)**.** *Every accepted* MetaCICResidualCheckerNormaliza *consumer factors through*

$$\text{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto \text{CandidateMediatedSN}^{\uparrow} \longmapsto \text{FiniteWindowConfluence}^{\uparrow} \longmapsto \text{MetaCICCandidateDia}$$

before it may read a budgeted normalization export. The export row X contains a finite normal endpoint only when the candidate evidence, fuel row, residual-checker row, join/refusal row, obstruction row, transport, replay, provenance, and local name are all displayed; otherwise it contains the displayed obstruction.

Proof. Start with the frontier row and the closed typed term T. The candidate row A supplies the only SN evidence that the packet accepts, and the fuel row F records the finite budget attached to that evidence.

Replay R sends this candidate route to the residual checker S. The checker can only consume finite windows, so the join/refusal row J either displays the relevant local join data or records why the route stops at the residual boundary.

The obstruction row O is retained in the export row X whenever candidate evidence, fuel, residual substitution, local joining, closedness transport, or endpoint sealing fails. Transport and replay through H, C, P, N keep the same finite packet attached to the local name, so the excluded global SN, residual confluence, subject reduction, quotient syntax, evaluator, oracle, and consistency claims cannot be exported. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}$ read that exposes a budgeted normalization endpoint while omitting the sealed closed term, candidate-mediated SN evidence, residual checker, fuel, route replay, join/refusal row, obstruction row, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: metacicnormalizationaudit, metacic-confluence-residual-checker, metacicnormalizationfrontier, candidate-mediated-sn: $\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries candidate-mediated SN evidence, residual-checker windows, finite fuel, local join or refusal rows, obstruction rows, transport, replay, provenance, and local naming in one budgeted normalization surface.

Lean `TasteGate.lean` for $\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}$ must later include Nontrivial and FieldFaithful instances so the candidate, checker, fuel, route, join/refusal, obstruction, export, replay, provenance, and local naming rows remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6666.1, Definition 6666.2, and Definition 6666.3 seed the finite residual-checker normalization budget route through closed terms, candidate evidence, checker windows, fuel, obstruction, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close evidence-free global SN, unrestricted residual confluence, subject-reduction transport, quotient syntax equality, hidden evaluator state, host reduction oracles, closed consistency, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sealed closed-term exposure, candidate-mediated SN exposure, residual-checker window exposure, fuel exactness, route replay, join/refusal exactness, obstruction retention, export nonescape, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualCheckerNormalizationBudgetUp.

Constructive story: A $\mathsf{MetaCICResidualCheckerNormalizationBudget}^{\uparrow}$ packet is generated from a sealed closed term, candidate-mediated SN evidence, residual-checker windows, finite fuel, route replay, join or refusal rows, obstruction rows, export rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6667

A Concrete Naming Certificate for $\mathsf{SequentialOrderTopology}^{\uparrow}$

Following Kelley and Engelking topology together with Bishop–Bridges sequence practice, the $\mathsf{SequentialOrderTopology}^{\uparrow}$ packet records the finite order/topology interface whose open reads are tested through displayed sequence windows. It sits beside the real order-topology packet of chapter 1890 and the dyadic order-topology packet of chapter 6612; it gives L10 consumers a sequence-window route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal without asserting Real completeness, trichotomy, host subsets, selected limits, or open-cover compactness.

Definition 6667.1 (Sequential order topology carrier)**.** *A* $\mathsf{SequentialOrderTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(O, T, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Order}^{\uparrow}$ *source row, including only the order comparisons and endpoint probes admitted by the local carrier.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open row whose opens are read from the displayed order intervals and not from host subset predicates.*
3. *S is the sequence-window source row, carrying the finite observation surface that downstream* $\mathsf{StreamName}^{\uparrow}$ *consumers may inspect.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite sequence names and index windows used to test sequential openness.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, attaching regular rational observations to the displayed sequence windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, used only as the terminal real-name boundary for sequence reads that have already passed through O, T, S, W, R.*
7. *H records componentwise* hsame *transport for the order, topology, sequence-window, stream-name, regular-readback, Real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from order intervals through generated opens, sequence windows, regular rational readback, the Real seal, and the local naming row.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialOrderTopology}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialOrderTopology}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, order trichotomy, host subset equality, selected limits, arbitrary open-cover compactness, quotient sequence equality, or a bridge certificate.

Theorem 6667.2 (Sequential order topology NameCert obligations)**.** *For an accepted* $\mathsf{SequentialOrderTopology}^{\uparrow}$ *carrier* $(O, T, S, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}$ *surface consists of five obligations:*

1. *carrier admission for the displayed* $\mathsf{Order}^{\uparrow}$ *source,* $\mathsf{Topology}^{\uparrow}$ *generated-open row, sequence-window source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*

2. *order-row stability, requiring every comparison used by a sequential-open read to pass through the displayed order row O before the topology row accepts it;*

3. *sequential-open readback, requiring every open read tested by a sequence to display S, W, R before reaching the generated-open row or the Real seal;*

4. *topology compatibility, requiring the topology row T to record only generated opens whose interval and sequence-window witnesses are replayed through C;*

5. *nonescape, stating that no Real completeness, trichotomy, host subset predicate, selected limit, open-cover compactness principle, quotient sequence equality, or bridge certificate is exported by the packet.*

Proof. Carrier admission is projection from Definition 6667.1. The tuple displays exactly one order row, one topology row, one sequence source, one stream-window row, one regular rational readback, one Real seal, and the structural transport, replay, provenance, and naming rows.

Order-row stability follows from the placement of O before T, W, R, E. A consumer cannot test interval membership, endpoint order, or order-generated openness without reading the displayed order row first.

Sequential-open readback is the route through S, W, R. The stream windows and regular rational observations are displayed before any sequence-facing open read reaches the topology row or the Real seal.

Topology compatibility is supplied by T and C. The topology row records generated opens from the displayed order and sequence-window witnesses, while the replay row preserves that route rather than replacing it by a host subset predicate.

Finally inspect the carrier inventory. The excluded completeness, trichotomy, selected-limit, open-cover, quotient, and bridge data are absent from the tuple, so no local NameCert projection can export them. □

Theorem 6667.3 (Sequential order topology StreamName handoff)**.** *Every sequence-convergence read consumed from an accepted* $\mathsf{SequentialOrderTopology}^{\uparrow}$ *packet passes through the finite route*

$$S \longrightarrow W \longrightarrow R \longrightarrow E \longrightarrow H, C, P, N.$$

Thus a convergence-facing consumer sees $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of the same packet before it can use the order/topology interface.*

Proof. Project the carrier. The only sequence source is S, the only stream-window row is W, and the only regular rational readback row is R, so a sequence-convergence request has no alternate entry point.

Read the handoff in order. The finite windows in W name the sequence observations, R attaches the regular rational readback needed by L10 completion consumers, and E records only the Real-seal boundary reached by those displayed rows.

Replay through H, C, P, N. These rows transport, replay, source, and name the same handoff, while the sibling order-topology chapters chapter 1890 and chapter 6612 supply point/order topology neighbours rather than a replacement for this sequence-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialOrderTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialOrderTopology}^{\uparrow}$ packet is generated from displayed Order, Topology, sequence-window, StreamName, RegSeqRat, Real-seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialOrderTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6667.1, Definition 6667.2, and Definition 6667.3 seed the finite sequential order-topology route through order, generated-open, sequence-window, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialOrderTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, host subsets, selected limits, quotient sequence equality, open-cover compactness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order-row stability, sequential-open readback, topology compatibility, StreamName handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.SequentialOrderTopologyUp.

Chapter 6668

A Concrete Naming Certificate for $\mathsf{HermiteInterpolation}^{\uparrow}$

Following Hermite interpolation, the $\mathsf{HermiteInterpolation}^{\uparrow}$ packet records the finite route from displayed nodes, values, and derivative jets to a polynomial whose formal derivatives match those displayed jets at the same nodes. It refines the node-and-value surface of chapter 4206, consumes the coefficient-list carrier of chapter 50, and routes Real-facing reads through chapter 38, chapter 301, and chapter 291. It is also separate from the orthogonal-family recurrence packet of chapter 6236: the present certificate names interpolation jets, not the Hermite polynomial family.

The BEDC carrier is finite. It stores the node list, the per-node jet-height list, the displayed derivative values, the confluent basis construction, the assembled polynomial, the matching and uniqueness checks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row. Every consumer must see those rows before reading a Hermite interpolant as a Real-facing polynomial approximation or calculus datum.

Definition 6668.1 (Hermite interpolation carrier)**.** *A* $\mathsf{HermiteInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, J, V, D, L, E, U, H, C, P, N)$$

with the following visible rows:

1. *X is a nonempty finite node row, with pairwise-distinctness data for the displayed nodes.*
2. *J is the jet-height row, assigning each node a finite number of formal derivative levels to be matched.*
3. *V is the displayed jet-value table, recording one scalar value for every pair (x_i, k) with $0 \leq k < J_i$.*
4. *D is the denominator and confluent-limit ledger, routing the nonzero node differences and repeated-node derivative factors through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *L is the Lagrange spine row imported from* $\mathsf{LagrangeInterpolation}^{\uparrow}$*, recording the ordinary node basis before the confluent jet rows are attached.*
6. *E is the confluent basis row, displaying the finite* $\mathsf{Poly}^{\uparrow}$ *coefficient lists whose formal derivatives realize the Kronecker jet pattern on X.*

7. *U is the assembly, jet-matching, and degree-bounded uniqueness row for the finite sum of displayed jet values times the basis polynomials.*
8. *H records componentwise* hsame *transport for node, jet-height, jet-value, denominator, Lagrange-spine, basis, assembly, replay, provenance, and local-name rows.*
9. *C records the displayed* Cont *routes that replay denominator formation, confluent basis construction, polynomial differentiation, jet evaluation, and uniqueness comparison from the same finite packet.*
10. *P is the* Pkg *provenance over* $\mathsf{HermiteInterpolation}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}$ *row.*

The carrier predicate is $\mathsf{HermiteInterpolationLedger}(X, J, V, D, L, E, U, H, C, P, N)$. *Two accepted packets classify together exactly when the eleven displayed rows are transported by H and replay through the same routes named by C and P.*

Theorem 6668.2 (Hermite interpolation NameCert obligations)**.** *For an accepted* $\mathsf{HermiteInterpolation}^{\uparrow}$ *packet $K = (X, J, V, D, L, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}$ *obligation surface consists exactly of the finite carrier and classifier rows of Definition 6668.1, the node and jet-height rows X, J, the jet-value table V, the denominator and confluent-limit ledger D, the Lagrange spine row L, the confluent* $\mathsf{Poly}^{\uparrow}$ *basis rows E, the assembly and uniqueness row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local naming row N. No host differentiability oracle, quotient polynomial equality, infinite jet family, selected Real representative, or ambient completeness datum is exported.*

Proof. Read the carrier of Definition 6668.1. The rows X, J, V supply the finite interpolation table: nodes, the number of derivatives requested at each node, and the displayed values for those derivative levels.

The construction rows are D, L, E, U. The denominator ledger D supplies the finite nonzero and confluent data, L supplies the ordinary Lagrange spine, E displays the polynomial coefficient lists for the confluent basis, and U assembles the interpolant and records the degree-bounded comparison.

The structural rows H, C, P, N transport, replay, source, and name only the displayed packet. Since the carrier has no additional coordinate for a host derivative operation, selected Real representative, quotient equality, or completion argument, the listed rows exhaust the local NameCert obligations. □

Theorem 6668.3 (Hermite interpolation jet matching)**.** *Every jet-matching read exported by an accepted* $\mathsf{HermiteInterpolation}^{\uparrow}$ *packet factors through the finite route*

$$X, J, V, D \longmapsto L \longmapsto E \longmapsto U \longmapsto N.$$

The route first verifies the distinct node row and the finite jet-height schedule, then builds the confluent basis from the Lagrange spine inside $\mathsf{Poly}^{\uparrow}$, *evaluates the required formal derivatives at the displayed nodes, assembles the interpolating polynomial, and exposes the local* $\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}$ *row. A consumer cannot bypass this route by appealing to ambient smooth functions, an external divided-difference table, quotient polynomial equality, or a Real-completion selector.*

Proof. Start with an accepted packet $K = (X, J, V, D, L, E, U, H, C, P, N)$. The rows X and J determine a finite set of jet positions, and V supplies exactly one displayed value for each such position.

Next inspect D and L. The denominator and confluent-limit ledger makes repeated-node factors explicit, while the Lagrange spine supplies the ordinary interpolation basis from which the confluent basis is built.

Now use E and U. The row E records the finite polynomial coefficient lists whose formal derivatives have the required Kronecker jet pattern, and U forms the finite linear combination using the displayed jet values. The same row records the degree-bounded uniqueness comparison, so a second accepted polynomial with the same jet table has the same displayed coefficient-list read.

Finally transport and replay through H, C, P, N. These rows preserve the same node, jet, denominator, basis, assembly, provenance, and local name. Thus every exported jet-matching read follows the displayed finite route and has no coordinate from which an ambient analytic or completion argument could be read. □

Falsifiable prediction: Every accepted $\mathsf{HermiteInterpolation}^{\uparrow}$ read that claims a Hermite interpolating polynomial factors through finite node rows, jet-height rows, jet-value rows, denominator and confluent-limit ledgers, a $\mathsf{LagrangeInterpolation}^{\uparrow}$ spine, normalized $\mathsf{Poly}^{\uparrow}$ basis rows, assembly and uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: polynomial, lagrange-interpolation, hermite-polynomial, real, regseqrat, dyadicratcore: $\mathsf{HermiteInterpolation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite node rows, finite jet schedules, displayed derivative values, confluent denominator data, a Lagrange spine, confluent polynomial basis rows, jet-matching assembly, uniqueness, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{HermiteInterpolation}^{\uparrow}$ packet is generated from finite node rows, finite jet-height rows, displayed jet-value rows, denominator and confluent-limit ledgers, a Lagrange interpolation spine, normalized polynomial basis rows, jet-matching assembly rows, uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6668.1, Definition 6668.2, and Definition 6668.3 seed the finite Hermite interpolation carrier, local NameCert obligation surface, and polynomial jet-matching route over displayed nodes, jet schedules, jet values, denominators, confluent basis polynomials, assembly rows, uniqueness rows, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HermiteInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient smooth-function interpolation, quotient polynomial equality, infinite jet families, selected Real representatives, analytic divided-difference tables, host differentiability oracles, field-completion arguments, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, node distinctness, jet-height finiteness, denominator and confluent-limit routing, Lagrange-spine exposure, confluent basis construction, jet evaluation replay, assembly, uniqueness-row replay, transport, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.HermiteInterpolationUp.

Chapter 6669

A Concrete Naming Certificate for $\mathsf{BirkhoffInterpolation}^{\uparrow}$

Following Birkhoff and de Boor, the $\mathsf{BirkhoffInterpolation}^{\uparrow}$ packet records the finite interpolation route in which only a displayed lacunary subset of derivative conditions is imposed. It sits beside the ordinary node-value route of chapter 4206 and the confluent jet route of chapter 6668; the present carrier names the mask that selects derivative orders rather than requiring every derivative level below a jet height. Its construction consumes the coefficient-list surface of chapter 50, the displayed derivative surface of chapter 122, the finite matrix ledger of chapter 49, and the linear-system route of chapter 48.

The packet is a finite BHist ledger. It displays nodes, a derivative-order mask, the lacunary value table, a polynomial basis, a finite matrix and linear-system ledger, assembly, transport, replay, provenance, and a local $\mathsf{NameCert}$ row before any consumer can read a Birkhoff interpolant. It is not ordinary Lagrange interpolation, because the mask may request derivative rows; it is not confluent Hermite interpolation, because the mask need not contain every lower derivative order at each node.

Definition 6669.1 (Birkhoff interpolation carrier)**.** *A* $\mathsf{BirkhoffInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, M, V, P, A, L, S, H, C, R, N)$$

with the following displayed rows:

1. *X is a finite node row, with pairwise-distinctness data for the displayed nodes.*
2. *M is the derivative-order mask row, recording a finite set of pairs (x_i, k) at which a derivative condition is requested.*
3. *V is the lacunary value table, assigning one scalar value to each selected pair in M.*
4. *P is the polynomial-basis row, displaying the finite* $\mathsf{Poly}^{\uparrow}$ *coefficient lists used for the unknown interpolant.*
5. *A is the finite matrix ledger whose entries are obtained by applying the requested* $\mathsf{Derivative}^{\uparrow}$ *orders from M to the basis row P and evaluating at the nodes X.*
6. *L is the* $\mathsf{LinMap}^{\uparrow}$ *linear-system row, recording the matrix map, right-hand side, solvability witness, and uniqueness witness when the Birkhoff pattern is poised.*

7. *S is the assembly row, transporting the solved coefficient vector back to a* $\mathsf{Poly}^{\uparrow}$ *coefficient list and recording the requested derivative matches against V.*
8. *H records componentwise* hsame *transport for node, mask, value, basis, matrix, linear-system, assembly, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay for derivative evaluation, matrix assembly, linear-system solution, polynomial assembly, and lacunary derivative checking.*
10. *R is the* Pkg *provenance over* $\mathsf{BirkhoffInterpolation}^{\uparrow}$, $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{HermiteInterpolation}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}$ *row.*

The carrier predicate is $\mathsf{BirkhoffInterpolationLedger}(X, M, V, P, A, L, S, H, C, R, N)$. *Two accepted packets classify together exactly when the eleven displayed rows are transported by H and replay through the same routes named by C and R.*

Lean4 ✓ `BEDC.Derived.BirkhoffInterpolationUp.BirkhoffInterpolationTasteGate single carrier alignment`

Theorem 6669.2 (Birkhoff interpolation NameCert obligations)**.** *For an accepted* $\mathsf{BirkhoffInterpolation}^{\uparrow}$ *packet $B = (X, M, V, P, A, L, S, H, C, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}$ *obligation surface consists exactly of the finite carrier and classifier rows of Definition 6669.1, the node row X, derivative-order mask M, lacunary value table V, polynomial-basis row P, matrix ledger A, linear-system row L, assembly and derivative-match row S, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance R, and the local naming row N. No host differentiability oracle, unstated determinant inverse, quotient polynomial equality, complete Hermite jet table, ordinary Lagrange-only table, or ambient completeness datum is exported.*

Proof. Read the carrier of Definition 6669.1. The rows X, M, V supply the finite interpolation request: nodes, the selected derivative orders, and one displayed value for each selected request.

The construction rows are P, A, L, S. The polynomial-basis row gives coefficient-list unknowns, the matrix row evaluates the selected derivative orders on those basis elements, the linear-system row records solvability and uniqueness for the displayed finite matrix, and the assembly row returns the solved coefficient vector as a polynomial satisfying the lacunary table.

The structural rows H, C, R, N transport, replay, source, and name only this displayed packet. Since the carrier has no further coordinate for a host derivative oracle, determinant inverse outside the matrix ledger, quotient polynomial equality, complete Hermite table, or Lagrange-only interpolation table, the listed rows exhaust the local NameCert obligations. □

Theorem 6669.3 (Birkhoff interpolation lacunary derivative route)**.** *Every derivative-match read exported by an accepted* $\mathsf{BirkhoffInterpolation}^{\uparrow}$ *packet factors through the finite route*

$$X, M, V, P \longmapsto A \longmapsto L \longmapsto S \longmapsto N.$$

The route first displays the node row and derivative-order mask, then evaluates the requested derivative orders against the polynomial basis to form a finite $\mathsf{Mat}^{\uparrow}$ *ledger, solves the displayed* $\mathsf{LinMap}^{\uparrow}$ *system, assembles the polynomial, checks the selected derivative rows against the lacunary value table, and exposes the local* $\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}$ *row. A consumer cannot replace this route with ordinary Lagrange interpolation, a full Hermite jet table, an ambient smooth-function argument, or an unstated matrix inverse.*

Proof. Start with an accepted packet $B = (X, M, V, P, A, L, S, H, C, R, N)$. The mask row M determines exactly which derivative requests are visible, and V supplies values only for those selected requests.

Next inspect P and A. The basis row is a finite $\mathsf{Poly}^{\uparrow}$ coefficient-list row, and A is obtained by applying the displayed $\mathsf{Derivative}^{\uparrow}$ orders from M to that basis and evaluating at the nodes in X.

Use the linear-system row L and assembly row S. The solved finite system supplies the coefficient vector, while S repacks it as a polynomial and checks precisely the derivative rows selected by M.

Finally transport and replay through H, C, R, N. These rows preserve the same node, mask, value, basis, matrix, linear-system, assembly, provenance, and local name, so every exported lacunary derivative read follows the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{BirkhoffInterpolation}^{\uparrow}$ read that claims a Birkhoff interpolating polynomial factors through finite node rows, derivative-order mask rows, lacunary value rows, polynomial-basis rows, finite $\mathsf{Mat}^{\uparrow}$ ledgers, $\mathsf{LinMap}^{\uparrow}$ solution rows, assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: lagrange-interpolation, hermite-interpolation, polynomial, derivative, matrix, linearmap: $\mathsf{BirkhoffInterpolation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite node data, a lacunary derivative-order mask, selected derivative values, polynomial basis rows, a derivative-evaluation matrix, a linear-system ledger, assembly, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffInterpolation}^{\uparrow}$ packet is generated from finite node rows, derivative-order mask rows, lacunary value rows, polynomial-basis rows, finite derivative-evaluation matrix rows, linear-system rows, assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6669.1, Definition 6669.2, and Definition 6669.3 seed the finite Birkhoff interpolation carrier, local NameCert obligation surface, and lacunary derivative route over displayed nodes, masks, values, basis polynomials, matrix ledgers, linear-system rows, assembly rows, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ordinary Lagrange-only interpolation, complete Hermite jet interpolation, host differentiability oracles, unstated determinant inverses, quotient polynomial equality, selected Real representatives, field-completion arguments, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, node distinctness, mask finiteness, lacunary value exposure, polynomial basis construction, derivative-evaluation matrix routing, linear-system solution rows, assembly, lacunary derivative replay, transport, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.BirkhoffInterpolationUp.

Chapter 6670

A Concrete Naming Certificate for $\mathsf{BaireDenseOpen}^{\uparrow}$

Following Baire and Bishop–Bridges, the $\mathsf{BaireDenseOpen}^{\uparrow}$ packet records the finite dense-open schedule that a constructive Baire-category argument may inspect before any generic-point or residual-set conclusion is asserted. It sits beside the Baire metric carrier of chapter 1602, the complete-metric handoff of chapter 128, the topology carrier of chapter 88, the stream-window interface of chapter 126, and the regular rational readback of chapter 301. The packet names dense-open requests, finite complete-metric windows, StreamName schedules, RegSeqRat readbacks, transport, replay, provenance, and local naming; it does not choose a generic point, take an ambient countable intersection, or import open-cover compactness.

Definition 6670.1 (Baire dense-open schedule carrier)**.** *A* $\mathsf{BaireDenseOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, W, S, R, T, C, H, P, N)$$

with the following displayed rows:

1. *D is the dense-open request row, recording the finite list of open predicates, dense-witness obligations, and inspected refinement indices.*
2. *W is the complete-metric window row, recording the displayed ball, radius, refinement, and nonempty-witness data available from* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{BaireMetric}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row controlling which dense-open requests and complete-metric windows may be inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational radii, shrinkage witnesses, and tolerance comparisons attached to W.*
5. *T is the topology row, recording that each inspected dense-open request is read through the displayed* $\mathsf{Topology}^{\uparrow}$ *basis and not through an ambient host set.*
6. *C records displayed* Cont *replay from D, S to W, from W to R, and from T to the local dense-open read.*
7. *H records componentwise* hsame *transport for dense-open rows, complete-metric windows, schedules, rational readback, topology rows, replay, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{BaireDenseOpen}^{\uparrow}$*,* $\mathsf{BaireMetric}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BaireDenseOpen}^\uparrow}$ *row.*

The classifier compares accepted packets only through $D, W, S, R, T, C, H, P, N$. It has no coordinate for a choice-selected generic point, a completed residual set, open-cover compactness, quotient equality of names, or ambient completeness beyond the displayed complete-metric windows.

Theorem 6670.2 (Baire dense-open schedule obligation surface)**.** *For every accepted* $\mathsf{BaireDenseOpen}^\uparrow$ *carrier $G = (D, W, S, R, T, C, H, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireDenseOpen}^\uparrow}$ *surface consists exactly of five finite rows:*

1. *dense-open request admission through D, including the displayed open rows, dense-witness obligations, and inspected refinement indices;*
2. *complete-metric window admission through W, with BaireMetric and CompleteMetric radii exposed before any schedule read is consumed;*
3. *finite observation scheduling through S and rational readback through R, so every inspected radius and shrinkage claim is replayed through StreamName and RegSeqRat rows;*
4. *topology and replay compatibility through T, C, H, preserving the same dense-open and window rows under* Cont *replay and componentwise* hsame *transport;*
5. *provenance and local naming through P, N, attaching the packet to the BaireMetric, CompleteMetric, Topology, StreamName, and RegSeqRat sources named in this chapter.*

The surface explicitly refuses a choice-selected generic point, open-cover compactness, quotient equality of names, and ambient completeness not already displayed in the complete-metric window row.

Proof. Project the carrier of Definition 6670.1. The dense-open row D and complete-metric window row W are the only mathematical rows visible before replay, so a consumer must name the inspected open predicates and the corresponding finite metric windows.

Next read the schedule and rational rows. The $\mathsf{StreamName}^\uparrow$ schedule S determines which entries of D and W are inspected, while the $\mathsf{RegSeqRat}^\uparrow$ row R records the rational radius and shrinkage readback for those entries.

Finally apply the topology, transport, replay, provenance, and local naming rows T, C, H, P, N. These rows preserve the displayed packet and do not add a selector, a host intersection, quotient equality, or an ambient completeness theorem. The five listed rows therefore exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{BaireDenseOpen}^\uparrow$ read exposes a finite dense-open schedule, complete-metric window row, StreamName schedule, and RegSeqRat readback before any Baire-category consumer may cite the packet.

Independence witness: bairemetric, completemetric, topology, streamname, regseqrat: $\mathsf{BaireDenseOpen}^\uparrow$ is not bijective to any listed sibling because it jointly carries dense-open request rows, complete-metric window rows, schedule rows, rational readback, topology replay, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireDenseOpen}^\uparrow}$**.**

Constructive story: A $\mathsf{BaireDenseOpen}^\uparrow$ packet is generated from finite dense-open request rows, complete-metric window rows, StreamName schedules, RegSeqRat rational readback, topology replay, componentwise transport, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireDenseOpen}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 6670.1 and Definition 6670.2 seed the finite dense-open schedule surface over displayed BaireMetric, CompleteMetric, Topology, StreamName, RegSeqRat, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BaireDenseOpen}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close choice-selected generic points, open-cover compactness, quotient equality of names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open row exactness, complete-metric window stability, StreamName schedule replay, RegSeqRat readback, topology compatibility, and local NameCert rows for BEDC.Derived.BaireDenseOpenUp.

Chapter 6671

A Concrete Naming Certificate for $\mathsf{KKMFiniteCover}^{\uparrow}$

Following Knaster, Kuratowski, and Mazurkiewicz, the $\mathsf{KKMFiniteCover}^{\uparrow}$ packet records the finite simplex-cover boundary used in the KKM lemma as displayed NameCert data. It sits beside the finite labelled-simplex route of chapter 2003, the convex-set carrier of chapter 208, the compactness surface of chapter 60, the continuous-map rows of chapter 130, and the simplicial approximation route of chapter 3402. The packet names only finite face-cover ledgers, classifier rows, transport, replay, provenance, and local naming; it does not assert arbitrary open-cover compactness, nonconstructive point selection, quotient closed-set equality, or host topological completeness.

Definition 6671.1 (KKM finite-cover carrier)**.** *A* $\mathsf{KKMFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, V, L, C_0, A, T, R, P, N)$$

with the following displayed rows:

1. *S is the finite simplex inventory row, read through the same finite-face surface used by* $\mathsf{SpernerLemma}^{\uparrow}$ *and* $\mathsf{SimplicialApproximation}^{\uparrow}$*.*
2. *F is the face-incidence ledger: each row names a face of the simplex and the finite set of vertices spanning that face.*
3. *V is the vertex-indexed cover row assigning to each displayed vertex a cover cell row.*
4. *L is the face-preserving classifier row, asserting that every displayed face is covered by the cells indexed by its own vertices.*
5. *C_0 is the* $\mathsf{ConvexSet}^{\uparrow}$ *support row for the carried simplex and its face hulls.*
6. *A is the* $\mathsf{Compact}^{\uparrow}$ *finite-subcover support row for the displayed simplex boundary.*
7. *T is the* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{SimplicialApproximation}^{\uparrow}$ *handoff row used when a consumer reads the cover through a subdivision or approximation route.*
8. *R is the* $\mathsf{SpernerLemma}^{\uparrow}$ *readback row converting the face-preserving cover classifier into the labelled-simplex obstruction route.*
9. *P is the* Pkg *provenance over* $\mathsf{KKMFiniteCover}^{\uparrow}$*,* $\mathsf{SpernerLemma}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{SimplicialApproximation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KKMFiniteCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, a selected point in an intersection, quotient equality of closed sets, countable choice, or ambient compactness outside the displayed finite simplex rows.

Theorem 6671.2 (KKM finite-cover NameCert obligations)**.** *For every accepted* $\mathsf{KKMFiniteCover}^{\uparrow}$ *carrier* $K = (S, F, V, L, C_0, A, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KKMFiniteCover}^{\uparrow}}$ *surface consists exactly of finite simplex admission, face-incidence exactness, vertex-cover row admission, face-preserving classifier stability, convex-face support, compact finite-boundary support, continuous or simplicial approximation handoff, Sperner readback, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Thus every finite-cover obligation factors through*

$$\mathsf{SpernerLemma}^{\uparrow}, \quad \mathsf{ConvexSet}^{\uparrow}, \quad \mathsf{Compact}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{SimplicialApproximation}^{\uparrow}.$$

No obligation is discharged by nonconstructive selection, arbitrary open-cover compactness, quotient cover equality, countable choice, or a host intersection theorem.

Proof. Project the carrier. The finite simplex and face-incidence rows S, F determine the only faces visible to the packet, while V, L record the vertex-indexed cover cells and the face-preserving classifier over those same rows.

Use the convex and compact support rows. The $\mathsf{ConvexSet}^{\uparrow}$ row C_0 supplies the finite face-hull surface, and the $\mathsf{Compact}^{\uparrow}$ row A supplies only the displayed finite-boundary support; neither row introduces arbitrary open-cover compactness or selected intersection points.

Route consumers through the finite topology dependencies. The $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{SimplicialApproximation}^{\uparrow}$ row T exposes subdivision or approximation reads, and the $\mathsf{SpernerLemma}^{\uparrow}$ row R supplies the labelled-simplex obstruction route already isolated by chapter 2003.

Finally preserve the packet through P, N and the displayed structural rows. Componentwise hsame transport and Cont replay keep the cover classifier attached to the same finite face ledger, so every admitted NameCert read factors through the named sibling certificates and no stronger compactness or selection principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{KKMFiniteCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KKMFiniteCover}^{\uparrow}$ packet is generated from a finite simplex inventory, face-incidence ledger, vertex-indexed cover rows, a face-preserving classifier, ConvexSetUp support, CompactUp finite-boundary support, ContinuousMapUp and SimplicialApproximationUp handoff rows, SpernerLemmaUp readback, structural transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KKMFiniteCover}^{\uparrow}}$, seed)

Scope closed: Definition 6671.1 and Definition 6671.2 seed the finite KKM cover route through SpernerLemmaUp, ConvexSetUp, CompactUp, ContinuousMapUp, SimplicialApproximationUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KKMFiniteCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, nonconstructive selection of an intersection point, quotient cover equality, countable choice, or a checked Lean certificate for the KKM packet.

Upgrade path: Obligation closure requires checked carrier admission, face-preserving classifier stability, finite-cover handoff, Sperner readback, structural replay, provenance exhaustion, and local naming for the same packet.

Chapter 6672

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ packet records the finite tail-collapse surface used when a regular Cauchy rational name is read only through sufficiently late windows. It sits over the finite observation service of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the paired-tail uniqueness packet of chapter 3942. The packet supplies Real and completion routes with displayed tail-collapse data; it does not identify quotient streams, select host limits, invoke countable choice, use trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6672.1 (Regular Cauchy tail-collapse carrier)**.** *A* $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, S, D, W, Q, U, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name whose late tail is being collapsed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule exposing the finite tail windows used by the collapse request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the displayed scale and the finite tail threshold.*
4. *W is the selected finite tail window after the threshold named by D.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the collapsed tail data.*
6. *U is the* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *sibling row used only to compare paired tail windows after the same dyadic threshold has been displayed.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row exposed after R, S, D, W, Q, U have been read.*
8. *H records componentwise* hsame *transport for the source name, schedule, dyadic threshold, tail window, readback, uniqueness sibling row, seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for any Real or completion consumer through the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailCollapse}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, S, D, W, Q, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCollapseUp.RegularCauchyTailCollapseTasteGate single carrier alignment`

Theorem 6672.2 (Regular Cauchy tail-collapse NameCert obligations)**.** *For every accepted* RegularCauchyTailColla *carrier $T = (R, S, D, W, Q, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailCollapse}^{\uparrow}}$ *surface consists exactly of source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold accounting, collapsed-tail* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *paired-tail witness, the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier through Definition 6672.1. The analytic rows are R, S, D, W, Q, U, E: the source regular name, the displayed schedule, the dyadic threshold, the finite tail window, the collapsed regular readback, the paired-tail uniqueness witness, and the terminal Real-facing seal.

The structural rows are H, C, P, N. They transport, replay, source, and name the same finite route, so the certificate surface has no remaining coordinate from which a quotient stream equality, selected host limit, choice schedule, trichotomy decision, or ambient completeness principle could be read. □

Theorem 6672.3 (Regular Cauchy tail-collapse handoff)**.** *Every Real or completion consumer of an accepted* $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} + \mathsf{DyadicRatCore}^{\uparrow} + \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyLimitUniqueness}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $S, D, R, W, Q, U, E, H, C, P, N$. The handoff therefore supplies RegSeqRat and RealUp completion arguments with a named tail-collapse packet while refusing quotient stream equality, selected limits, countable choice, trichotomy, and ambient Real completeness.

Proof. Begin with a consumer request. The schedule row S and tolerance row D display the finite threshold before any late tail entries are available, and the window row W records exactly the exposed tail segment.

Read the regular rational data through R and Q. The source name and collapsed-tail readback are both $\mathsf{RegSeqRat}^{\uparrow}$ rows, so no host stream quotient or selected limit is introduced at this stage.

Use the sibling $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ row U only after the same threshold and window have been displayed. The route can then expose the $\mathsf{Real}^{\uparrow}$ seal row E, and H, C, P, N preserve, replay, source, and name that finite packet for downstream Real and completion routes. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ consumer read factors through a StreamName schedule, DyadicRatCore threshold ledger, RegSeqRat source row, finite tail window, collapsed RegSeqRat readback, RegularCauchyLimitUniqueness paired-tail witness, RealUp seal, transport, replay, provenance, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-limit-uniqueness: $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source regular name, finite tail schedule, displayed dyadic threshold, collapsed readback, paired-tail uniqueness witness, Real seal boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailCollapse}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailCollapse}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName tail schedule, DyadicRatCore threshold ledger, finite tail window, collapsed RegSeqRat readback, RegularCauchyLimitUniqueness paired-tail witness, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailCollapse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6672.1, Definition 6672.2, and Definition 6672.3 seed the finite tail-collapse route consumed by RegSeqRat and RealUp completion arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailCollapse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold accounting, collapsed-tail readback, paired-tail uniqueness handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailColla

Chapter 6673

A Concrete Naming Certificate for $\mathsf{SuslinTree}^{\uparrow}$

Following Suslin's tree problem and standard set-theoretic order theory, the $\mathsf{SuslinTree}^{\uparrow}$ packet records the visible order-theoretic surface of a height-ω_1 tree with no displayed uncountable chain and no displayed uncountable antichain. It sits beside the $\mathsf{AronszajnTree}^{\uparrow}$ obstruction of chapter 6341, the $\mathsf{Tree}^{\uparrow}$ source of chapter 119, the $\mathsf{POSet}^{\uparrow}$ comparison surface of chapter 53, the unary $\mathsf{Order}^{\uparrow}$ read of chapter 1378, the membership discipline of chapter 111, and the ordinal-analysis ledger of chapter 6247. The packet names the finite BEDC route through tree source, partial-order comparison, level accounting, chain-boundary refusal, antichain-boundary refusal, transport, replay, provenance, and local naming; it does not assert Suslin's hypothesis, construct a Suslin tree in any host universe, choose a club, or decide independence from the usual axioms of set theory.

Definition 6673.1 (Suslin tree carrier)**.** *A* $\mathsf{SuslinTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, O, L, C, A, H, K, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Tree}^{\uparrow}$ *source row, naming the tree nodes, predecessor route, root discipline, and displayed level map.*
2. *O is the* $\mathsf{POSet}^{\uparrow}$ *and* $\mathsf{Order}^{\uparrow}$ *comparison row, recording the tree order as a partial-order read with finite predecessor comparison data.*
3. *L is the level-accounting row, exposing the countable displayed levels and the ordinal-height interface used by* $\mathsf{OrdinalAnalysis}^{\uparrow}$*.*
4. *C is the chain-boundary row, recording that no displayed uncountable chain route is exported by the packet.*
5. *A is the antichain-boundary row, recording that no displayed uncountable antichain route is exported by the packet.*
6. *H stores componentwise* hsame *transport for the tree source, order comparison, level ledger, chain boundary, antichain boundary, continuation, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from order and level requests through T, O, L, C, A.*
8. *P is the* Pkg *provenance over* $\mathsf{SuslinTree}^{\uparrow}$*,* $\mathsf{AronszajnTree}^{\uparrow}$*,* $\mathsf{Tree}^{\uparrow}$*,* $\mathsf{POSet}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*,* $\mathsf{Set}^{\uparrow}$*,* $\mathsf{OrdinalAnalysis}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SuslinTree}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $T, O, L, C, A, H, K, P, N$. It has no coordinate for a host construction of ω_1, a selected club, a forcing extension, a proof or refutation of Suslin's hypothesis, Kurepa splitting, a maximal antichain selector, or a detached set-theoretic universe.

Theorem 6673.2 (Suslin tree NameCert obligations)**.** *For every accepted* $\mathsf{SuslinTree}^{\uparrow}$ *carrier $S = (T, O, L, C, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SuslinTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tree source, order comparison, level ledger, chain boundary, antichain boundary, transport, replay, provenance, and local naming rows; tree-source visibility through T; partial-order and unary-order visibility through O; level accounting through L; chain and antichain boundary visibility through C, A; componentwise transport through H; displayed continuation through K; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6673.1. The accepted inventory is exactly $T, O, L, C, A, H, K, P, N$, so every consumer must read the tree source, order comparison, level ledger, and two boundary rows before the local name is available.

The sibling anchors supply only their displayed surfaces. The Aronszajn-tree chapter chapter 6341 supplies the height-ω_1 branch-boundary comparison, while the tree, poset, order, set, and ordinal-analysis chapters supply source, comparison, membership, and level-accounting rows.

Rows H, K, P, N preserve and replay that same finite route. Since no carrier coordinate names a host universe, forcing construction, selected club, maximal antichain selector, or decision of Suslin's hypothesis, none of those endpoints can be exported by the NameCert obligations. □

Theorem 6673.3 (Suslin tree chain–antichain boundary)**.** *Every order, set, or ordinal-analysis consumer of an accepted* $\mathsf{SuslinTree}^{\uparrow}$ *carrier must read both boundary rows*

$$C, A$$

after the displayed tree source, partial-order comparison, and level ledger rows T, O, L. The row C blocks an exported uncountable chain route, and the row A blocks an exported uncountable antichain route. The packet therefore records the Suslin-tree chain–antichain boundary as finite NameCert data beside $\mathsf{AronszajnTree}^{\uparrow}$*, without deciding Suslin's hypothesis, constructing a host Suslin tree, choosing a club, selecting a maximal antichain, or importing a detached set-theoretic universe.*

Proof. Start with Definition 6673.2. It requires the source row T, the order row O, and the level row L before either boundary can be read.

Read the two boundary rows separately. The chain boundary C refuses an exported uncountable branch or chain route, while the antichain boundary A refuses an exported uncountable antichain route. Keeping both rows visible is what distinguishes the Suslin-tree packet from the sibling $\mathsf{AronszajnTree}^{\uparrow}$ route.

Finally apply the structural rows of Definition 6673.1. Transport, replay, provenance, and local naming preserve only the displayed boundary packet. They cannot add a host construction, force a set-theoretic decision, choose a club, select a maximal antichain, or import any detached universe data. □

Falsifiable prediction: Every accepted $\mathsf{SuslinTree}^{\uparrow}$ order-facing read factors through a TreeUp source row, POSetUp and OrderUp comparison row, level-accounting row, chain-boundary row, antichain-boundary row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: aronszajn-tree, tree, poset, order, set, ordinal-analysis: $\mathsf{SuslinTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a tree source, order comparison, level ledger, no-uncountable-chain boundary, no-uncountable-antichain boundary, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SuslinTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SuslinTree}^{\uparrow}$ packet is generated from a displayed TreeUp source, POSetUp and OrderUp comparison row, level-accounting ledger, chain-boundary row, antichain-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SuslinTree}^{\uparrow}}$, seed)

Scope closed: Definition 6673.1, Definition 6673.2, and Definition 6673.3 seed the finite Suslin-tree order-boundary packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SuslinTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Suslin's hypothesis, host construction of a Suslin tree, forcing independence, club selection, maximal antichain selection, or ambient set-theoretic completeness.

Upgrade path: Obligation closure requires checked carrier admission, tree-source discipline, order-comparison exactness, level-ledger visibility, chain-boundary nonescape, antichain-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SuslinTreeUp.

Chapter 6674

A Concrete Naming Certificate for $\mathsf{TaylorPolynomial}^{\uparrow}$

Following Taylor's classical finite polynomial approximation in differential calculus, the $\mathsf{TaylorPolynomial}^{\uparrow}$ packet records the finite jet-to-polynomial object used between derivative observations and theorem-facing Taylor estimates. It sits beside the real-name boundary of chapter 38, the polynomial carrier of chapter 50, the derivative observation row of chapter 122, the calculus substrate of chapter 3632, and the Taylor theorem route of chapter 5898. The packet names a finite Taylor polynomial certificate for one displayed center, order, coefficient ledger, polynomial assembly row, and evaluation handoff; it does not import a host smooth-function space, analytic convergence, quotient equality of real functions, or an infinite Taylor series.

Definition 6674.1 (Taylor polynomial carrier)**.** *A* $\mathsf{TaylorPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (c, k, D, J, P, V, Q, E, H, C, G, N)$$

with the following displayed rows:

1. *c is the named* $\mathsf{Real}^{\uparrow}$ *center row and k is the finite order row.*
2. *D is the* $\mathsf{Derivative}^{\uparrow}$ *observation row naming exactly the derivative data used for orders at most k.*
3. *J is the finite jet ledger. Each coefficient row cites c, k, D and the dyadic rational arithmetic used to normalize the displayed derivative read.*
4. *P is the* $\mathsf{Poly}^{\uparrow}$ *assembly row whose coefficients are exactly the rows of J.*
5. *V is the finite evaluation handoff. It records the polynomial value requests, the local displacement from c, and the coefficient rows of P consumed by each request.*
6. *Q lists the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for coefficients, displacements, and finite polynomial values.*
7. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seals for the center, coefficient reads, and polynomial values.*
8. *H records componentwise* hsame *transport for the center, order, derivative source, jet ledger, polynomial assembly, evaluation handoff, readback rows, endpoint seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every Taylor-polynomial read through $c, k, D, J, P, V, Q, E, H, G, N$.*

10. *G is the* Pkg *provenance over* $\mathsf{TaylorPolynomial}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorPolynomial}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $c, k, D, J, P, V, Q, E, H, C, G, N$. It has no coordinate for a host function, an infinite Taylor series, analytic convergence, quotient equality of functions, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TaylorPolynomialUp.TaylorPolynomialTasteGate single carrier alignment`

Theorem 6674.2 (Taylor polynomial NameCert obligations)**.** *For every accepted* $\mathsf{TaylorPolynomial}^{\uparrow}$ *carrier*

$$T = (c, k, D, J, P, V, Q, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorPolynomial}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed center, finite order, derivative source, jet ledger, polynomial assembly, evaluation handoff, readback rows, endpoint seals, transport, replay, provenance, and local naming rows;*
2. *derivative-to-coefficient routing, requiring every coefficient in J to cite the* $\mathsf{Derivative}^{\uparrow}$ *row D at the center c and finite order k;*
3. *polynomial assembly exactness, requiring P to use exactly the coefficient rows displayed in J and no hidden host polynomial;*
4. *evaluation handoff exactness, requiring every row of V to consume only the displayed displacement from c, the assembled polynomial row P, and finite coefficient arithmetic;*
5. *regular and real readback, requiring Q to expose regular rational reads before the terminal* $\mathsf{Real}^{\uparrow}$ *seals in E are consumed;*
6. *structural stability, requiring H, C, G, N to transport, replay, source, and name only the rows already listed.*

No $\mathsf{TaylorPolynomial}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 6674.1. The theorem-facing rows are exactly c, k, D, J, P, V, Q, E, and the structural rows are exactly H, C, G, N.

The coefficient route is finite. Each row of J is attached to the displayed derivative source D, the center c, and the finite order k, so a coefficient consumer cannot replace it by a host smooth-function expansion or by an infinite Taylor series.

The polynomial and evaluation rows are then forced. The row P assembles precisely the listed coefficient ledger J, while V reads finite polynomial values from P and the displayed displacement from c.

Finally Q, E, H, C, G, N seal the route. Regular rational readback precedes the terminal real seals, and transport, replay, provenance, and local naming preserve exactly the same finite packet. Since no carrier coordinate names analytic convergence, quotient function equality, selected limits, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6674.3 (Taylor polynomial jet route)**.** *Every accepted* $\mathsf{TaylorPolynomial}^{\uparrow}$ *carrier exports a coefficient only through the finite route*

$$\mathsf{Calculus}^{\uparrow}, \mathsf{Derivative}^{\uparrow} \longmapsto J \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the center c, finite order k, derivative row D, jet ledger J, polynomial row P, readback row Q, endpoint seals E, transport H, replay C, provenance G, and local naming row N before it may consume a displayed Taylor coefficient. The export is a finite jet read, not a host differentiability theorem, analytic convergence claim, quotient equality of functions, selected limit, or ambient real-completeness row.

Proof. Start with an accepted carrier $T = (c, k, D, J, P, V, Q, E, H, C, G, N)$. The only derivative source is D, and the order bound is the displayed finite row k.

Read the coefficient ledger J. Each coefficient row is already tied to c, k, D and to finite dyadic arithmetic, so the route stays inside the displayed calculus and derivative surface.

Then pass through P, Q, E, H, C, G, N. The assembled polynomial records the same finite ledger, regular rational readback exposes coefficient data before real sealing, and the structural rows preserve the packet. Therefore every coefficient export factors through the stated finite route. □

Theorem 6674.4 (Taylor polynomial polynomial handoff)**.** *Every accepted* $\mathsf{TaylorPolynomial}^{\uparrow}$ *carrier hands a polynomial value to downstream calculus or Taylor-theorem consumers only through the displayed rows*

$$c, k, D, J, P, V, Q, E, H, C, G, N.$$

The handoff exports a finite $\mathsf{Poly}^{\uparrow}$ *object with named center, finite order, coefficient ledger, evaluation requests, regular rational readbacks, and terminal real seals. It does not export a Taylor theorem remainder, analytic convergence, a selected germ, quotient equality of functions, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the assembled polynomial row P from the accepted carrier. By Definition 6674.2, that row is assembled only from the displayed jet ledger J, and J is routed through the finite derivative source D at c and order k.

Next read the evaluation handoff V and the readback rows Q, E. Each value request consumes the polynomial row and the displayed displacement from the center, then passes through regular rational readback before any terminal real seal is visible.

The remaining rows H, C, G, N transport, replay, source, and name the same packet. Since the carrier contains no remainder row, analytic convergence row, selected germ, quotient function equality, countable-choice row, or ambient real-completeness row, every downstream polynomial handoff is exhausted by the displayed finite data. □

Falsifiable prediction: Every accepted $\mathsf{TaylorPolynomial}^{\uparrow}$ coefficient or value read factors through a displayed center, finite order, DerivativeUp source, finite jet ledger, PolynomialUp assembly row, evaluation handoff, RegSeqRat readback, Real endpoint seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: real, polynomial, derivative, calculus, taylor-theorem: $\mathsf{TaylorPolynomial}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds one center, one finite order, derivative observations, a coefficient ledger, a polynomial assembly row, evaluation requests, readback rows, endpoint seals, transport, replay, provenance, and local naming in one finite-jet certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorPolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaylorPolynomial}^{\uparrow}$ packet is generated from a displayed $\mathsf{Real}^{\uparrow}$ center, finite order, $\mathsf{Derivative}^{\uparrow}$ source over the $\mathsf{Calculus}^{\uparrow}$ substrate, finite coefficient ledger, $\mathsf{Poly}^{\uparrow}$ assembly row, evaluation handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{Real}^{\uparrow}$ endpoint seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TaylorPolynomial}\uparrow}$, seed)

Scope closed: Definition 6674.1, Definition 6674.2, Definition 6674.3, and Definition 6674.4 seed the finite Taylor-polynomial route from derivative observations to polynomial coefficients and value handoffs.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaylorPolynomial}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Taylor theorem remainders, analytic convergence, selected germs, quotient equality of functions, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-coefficient routing, polynomial assembly exactness, evaluation handoff exactness, regular and real readback, transport and replay stability, and nonescape for BEDC.Derived.TaylorPolynomialUp.

Chapter 6675

A Concrete Naming Certificate for $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$

Following Bishop and Bridges/Troelstra constructive analysis, the $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$ packet records the finite boundary object used when an inspected real-name route stabilizes a bounded disjunction without importing LLPO, WLPO, Markov principle, trichotomy, or quotient real equality. It sits beside the Ishihara route of chapter 1665, the boundary packets of chapter 1978 and chapter 1666, the lower-cut surface of chapter 6657, the finite window route of chapter 126, the regular readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a reviewable finite branch window for real-boundary consumers; it does not choose a branch by a semi-classical principle.

Definition 6675.1 (FiniteDisjunctionStabilization carrier)**.** *A* $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite branch-window row. It names the bounded disjunction entries and the displayed inspection depth.*
2. *I is the* $\mathsf{IshiharaTrick}^{\uparrow}$ *inspection row, read only as a finite route over the displayed branch window.*
3. *L is the lower-cut evidence row supplied through* $\mathsf{SemiDecidableRealCut}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the finite comparison scale.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the requested real-name windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the displayed windows and dyadic tolerance.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after B, I, L, D, W, R have been replayed.*
8. *H records componentwise* hsame *transport for branch windows, Ishihara inspection, lower-cut evidence, dyadic tolerance, stream schedule, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the branch window, inspection row, lower-cut row, tolerance row, schedule row, readback row, and Real seal.*

10. *P is the* Pkg *provenance over* FiniteDisjunctionStabilization$^{\uparrow}$, IshiharaTrick$^{\uparrow}$, LLPOBoundary$^{\uparrow}$, MarkovPrincipleBoundary$^{\uparrow}$, SemiDecidableRealCut$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{FiniteDisjunctionStabilization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, I, L, D, W, R, E, H, C, P, N$. It has no coordinate for LLPO, WLPO, Markov principle as an axiom, trichotomy, quotient real equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6675.2 (FiniteDisjunctionStabilization NameCert obligations)**.** *For every accepted* FiniteDisjunctionStab *carrier $S = (B, I, L, D, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{FiniteDisjunctionStabilization}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite branch window, Ishihara inspection, lower-cut evidence, dyadic tolerance, stream schedule, regular readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows; branch-window stability under* hsame*; replay order through C; provenance through P; and the refusal of semi-classical branch selection outside the displayed finite rows.*

Proof. Project the carrier of Definition 6675.1. The analytic rows are B, I, L, D, W, R, E, and the structural rows are H, C, P, N.

The replay row C forces every read to begin with the finite branch window before it reaches Ishihara inspection, lower-cut evidence, dyadic tolerance, stream windows, regular readback, and the Real seal. Transport by H preserves exactly that order under classifier movement.

The provenance row P names the sibling boundaries only as sources for the displayed route. Since the carrier contains no independent branch selector, excluded-middle row, quotient real equality row, selected limit, choice selector, or completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 6675.3 (FiniteDisjunctionStabilization branch window)**.** *Every accepted* FiniteDisjunctionStabilization$^{\uparrow}$ *read factors through the finite branch window B before any Ishihara inspection, lower-cut evidence, dyadic tolerance, stream schedule, regular readback, or Real seal is visible. Thus a consumer may stabilize only the displayed finite disjunction window, not an unbounded search over real names.*

Proof. Start from a consumer request. Carrier admission exposes B as the unique branch-window coordinate, and the continuation row C places it before I, L, D, W, R, E.

The Ishihara row I is therefore read as an inspection of the displayed finite window, while L, D, W, R supply only the lower-cut, tolerance, schedule, and readback rows needed to make that window meaningful at the requested scale.

Since the Real seal E appears after those rows, no consumer can bypass the finite window by asking for quotient real equality, trichotomy, selected limits, or Real$^{\uparrow}$ completeness. □

Theorem 6675.4 (FiniteDisjunctionStabilization real-boundary route)**.** *Every Real-facing consumer of an accepted* FiniteDisjunctionStabilization$^{\uparrow}$ *carrier must read the route*

$$B, I, L, D, W, R \longmapsto E \longmapsto H, C, P, N.$$

The boundary exported to Real and completion-facing chapters is a finite stabilization certificate over displayed cuts, dyadic tolerances, stream windows, and regular readback; it is not LLPO, WLPO, Markov principle as an axiom, trichotomy, quotient real equality, a selected limit, countable choice, or Real$^{\uparrow}$ *completeness.*

Proof. Read the branch-window theorem first. It confines the request to the displayed finite disjunction window and its Ishihara inspection row.

Next attach the lower-cut and dyadic rows. The lower-cut evidence L and tolerance row D determine which finite stream windows W and regular readback rows R may be inspected before the Real seal is visible.

Finally replay the Real-facing route through E, H, C, P, N. The seal records that the displayed finite route is available to downstream consumers; it does not add a semi-classical principle or any ambient completeness object. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$ read factors through a finite branch window, Ishihara inspection row, semi-decidable lower-cut evidence, DyadicRatCore tolerance, StreamName schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: ishihara-trick, llpo-boundary, markov-principle-boundary, semi-decidable-real-cut, streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite branch-window stabilization, real-boundary inspection, lower-cut evidence, dyadic tolerance, stream and regular readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDisjunctionStabilization}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDisjunctionStabilization}^{\uparrow}$ packet is generated from a finite branch window, Ishihara inspection row, semi-decidable lower-cut evidence, DyadicRatCore tolerance row, StreamName schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDisjunctionStabilization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6675.1, Definition 6675.2, Definition 6675.3, and Definition 6675.4 seed the finite real-boundary route for disjunction stabilization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDisjunctionStabilization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close LLPO, WLPO, Markov principle as an axiom, trichotomy, quotient real equality, selected limits, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-window stability, Ishihara inspection routing, lower-cut and dyadic tolerance accounting, stream and regular readback, Real-seal nonescape, and structural replay for BEDC.Derived.FiniteDisjunctionStabilizationUp.

Chapter 6676

A Concrete Naming Certificate for $\mathsf{CompactGridModulusExtraction}^{\uparrow}$

Following Bishop-style compact metric finite-net arguments, the $\mathsf{CompactGridModulusExtraction}^{\uparrow}$ packet realizes the compact-grid route of chapter 9237 as finite NameCert data. It sits beside compact metric coverage in chapter 129, continuous-map pointwise control in chapter 130, the uniform-modulus consumer of chapter 369, the dyadic tolerance ledger of chapter 291, rational lower-bound readback in chapter 37, and the terminal real-name seal of chapter 38. The packet records a finite compact-net grid and modulus extraction boundary without importing open-cover compactness, selected subcovers, countable choice, or ambient real completeness.

Definition 6676.1 (Compact grid modulus extraction carrier)**.** *A* $\mathsf{CompactGridModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, D, Q, M, U, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including the displayed grid centers and coverage witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the same compact metric source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid-tolerance row used to compare source distances and target tolerances.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the finite grid radii.*
5. *M is the extracted modulus-candidate row produced from K, F, D, Q.*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for Real-valued compact-continuity consumers.*
8. *H records componentwise* hsame *transport for compact grid data, continuous-map pointwise rows, dyadic tolerances, rational folds, extracted modulus candidates, UniformModulus handoff, Real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through the compact grid, pointwise modulus, dyadic tolerance, rational fold, extracted modulus row, UniformModulus handoff, and Real seal.*

10. *P is the* Pkg *provenance over* CompactGridModulusExtraction$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CompactGridModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, D, Q, M, U, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers, countable choice, quotient real equality, selected limits, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6676.2 (Compact grid modulus extraction NameCert obligations)**.** *For every accepted* CompactGridModulusExtraction$^{\uparrow}$ *carrier $G = (K, F, D, Q, M, U, E, H, C, P, N)$, the local* NameCert$_{\text{CompactGridModulusE}}$ *surface consists exactly of carrier admission for the compact finite grid, continuous-map pointwise rows, dyadic tolerance row, rational lower-bound fold, extracted modulus candidate, UniformModulus handoff, Real seal, transport, replay, provenance, and local naming rows; finite-net coverage accounting; lower-bound stability under* hsame*; handoff exactness to* UniformModulus$^{\uparrow}$*; and refusal of open-cover or choice-selected compactness outside the displayed grid.*

Proof. Project the carrier of Definition 6676.1. The construction rows are K, F, D, Q, M, U, E, and the structural rows are H, C, P, N.

The compact grid row K supplies only displayed centers and coverage witnesses. The continuous-map row F supplies pointwise modulus data over those centers, while D and Q record the dyadic and rational lower-bound calculations needed to extract M.

The replay row C sends the extracted candidate to UniformModulus$^{\uparrow}$ through U and then to Real-valued consumers through E. Transport, provenance, and local naming preserve that same finite packet, so no open-cover compactness, selected subcover, choice selector, quotient real equality, selected limit, or ambient completeness row is present. □

Theorem 6676.3 (Compact grid modulus extraction finite-net handoff)**.** *Every accepted* CompactGridModulusExtra *carrier hands compact coverage to the modulus extractor only through the displayed finite route*

$$K, F, D, Q \longmapsto M.$$

The handoff uses compact metric grid centers, continuous-map pointwise rows, dyadic tolerances, and a rational lower-bound fold; it does not use open-cover compactness, a selected subcover, sequential compactness, or countable choice.

Proof. Start from the compact grid row K. It exposes a finite net with displayed coverage witnesses rather than an arbitrary open cover.

Attach the continuous-map row F at each displayed grid center. The pointwise radius data is then compared through the dyadic tolerance row D and folded through the rational lower-bound row Q.

The extracted row M is therefore downstream of the finite grid and lower-bound fold. Since no carrier coordinate names an open cover, selected subcover, sequential compactness witness, or choice selector, the finite-net handoff is the only compactness source available. □

Theorem 6676.4 (Compact grid modulus extraction uniform-modulus boundary)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* CompactGridModulusExtraction$^{\uparrow}$ *carrier reads the boundary*

$$K, F, D, Q, M \longmapsto U \longmapsto E, H, C, P, N.$$

The exported modulus is a finite compact-grid extraction from compact metric coverage, continuous pointwise rows, dyadic tolerances, and rational lower bounds. It is not an ambient Heine–Cantor oracle, quotient real equality, selected limit, countable-choice selector, or Real$^{\uparrow}$ *completeness.*

Proof. Use the finite-net handoff theorem to place M after K, F, D, Q. This ensures that the candidate modulus is derived from the displayed grid route.

The UniformModulus row U receives only that candidate and the structural replay data. The Real seal E is downstream of the same handoff and can be inspected only after the finite grid extraction has been displayed.

Finally read H, C, P, N. These rows transport, replay, source, and name the same packet, so the boundary contains no ambient uniform-continuity theorem, quotient real equality, selected limit, choice selector, or real-completeness row. □

Falsifiable prediction: Every accepted $\mathsf{CompactGridModulusExtraction}^{\uparrow}$ read factors through CompactMetric finite grid data, ContinuousMap pointwise-modulus rows, DyadicRatCore tolerance, RatUp lower-bound fold, an extracted modulus candidate, UniformModulus handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, rat, real: $\mathsf{CompactGridModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net coverage, pointwise continuous rows, dyadic grid tolerance, rational lower-bound fold, extracted modulus candidate, UniformModulus handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactGridModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactGridModulusExtraction}^{\uparrow}$ packet is generated from CompactMetric finite grid data, ContinuousMap pointwise-modulus rows, DyadicRatCore tolerance, RatUp lower-bound fold, an extracted modulus candidate, UniformModulus handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactGridModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6676.1, Definition 6676.2, Definition 6676.3, and Definition 6676.4 seed the finite compact-grid route from compact metric coverage to UniformModulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactGridModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, sequential compactness, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-grid coverage accounting, pointwise-modulus routing, dyadic tolerance and rational lower-bound stability, UniformModulus handoff exactness, Real-seal nonescape, and structural replay for BEDC.Derived.CompactGridModulusExtractionUp.

Chapter 6677

A Concrete Naming Certificate for $\mathsf{BishopChoiceBar}^{\uparrow}$

Following Bishop and Bridges on constructive choice principles under explicit finite information, the $\mathsf{BishopChoiceBar}^{\uparrow}$ packet records the bar-controlled boundary at which a completion-facing consumer may use inspected finite prefixes and a local selection ledger without acquiring countable choice. It sits on the L10 Real and Completion route beside the finite stream-name interface of chapter 126, the Cauchy-modulus schedule discipline of chapter 292, the choice-free limit schedule of chapter 650, the fan theorem boundary of chapter 400, and the lawless sequence prefix surface of chapter 1495. The packet names only the displayed bar coverage, inspected windows, selected-window ledger, and structural rows; it does not claim countable choice, a full fan theorem, $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, or selected global limits.

Definition 6677.1 (Bishop choice bar carrier)**.** *A* $\mathsf{BishopChoiceBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, A, W, L, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the finite prefix windows that the completion-facing request is allowed to inspect.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *request row, recording the displayed precision threshold and the finite bound at which the bar is queried.*
3. *A is the bar-admission row. It records finite bar coverage for the inspected prefix family and cites the fan-boundary sibling only as a displayed finite coverage discipline.*
4. *W is the selected-window row. It lists the windows actually read after S, M, A have been exposed.*
5. *L is the selected-window ledger. It records that every selected window in W is justified by a visible bar hit and remains inside the choice-free limit schedule.*
6. *D is the lawless-prefix compatibility row, ensuring that prefix reads use only displayed finite observations and no global law for the stream.*
7. *H records componentwise* hsame *transport for source, modulus, bar-admission, selected-window, ledger, lawless-prefix, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from completion-facing requests through $S, M, A, W, L, D, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopChoiceBar}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{LawlessSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopChoiceBar}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, M, A, W, L, D, H, C, P, N$. It has no coordinate for countable choice, an arbitrary fan theorem, quotient stream equality, selected global limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6677.2 (Bishop choice bar NameCert obligations)**.** *For every accepted* $\mathsf{BishopChoiceBar}^{\uparrow}$ *carrier $B = (S, M, A, W, L, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopChoiceBar}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed stream source, Cauchy-modulus request, bar-admission row, selected-window row, selected-window ledger, lawless-prefix compatibility row, transport, replay, provenance, and local naming rows;*

2. *finite bar coverage, requiring A to cover only the inspected prefix family visible through S and the precision threshold visible through M;*

3. *selected-window ledger exactness, requiring every row of W to cite a visible bar hit in A and a schedule edge in the choice-free limit route before any completion-facing read is consumed;*

4. *lawless-prefix compatibility, requiring D to keep prefix observations finite and to forbid replacement by a global stream law;*

5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed;*

6. *non-escape, forbidding countable choice, a full fan theorem, quotient stream equality, selected global limits, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the local surface.*

No $\mathsf{BishopChoiceBar}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 6677.1. The accepted inventory is exactly $S, M, A, W, L, D, H, C, P, N$, so every local naming read must pass through the displayed stream source, modulus request, finite bar row, selected windows, ledger, and lawless-prefix compatibility row.

The bar row is finite. It covers only the inspected prefix family exposed by S at the threshold exposed by M, and its citation of the fan-theorem sibling is only the displayed finite coverage discipline needed for those prefixes.

The selected-window ledger is then forced. Each selected window in W is justified by a visible hit in A and is replayed through the choice-free limit schedule, so the packet records a bounded local selection rather than a selector over all requests.

The lawless-prefix and structural rows close the route. The row D forbids replacing finite observations by a global stream law, while H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for countable choice, a full fan theorem, quotient stream equality, selected global limits, host real equality, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6677.3 (Bishop choice bar L10 dependency route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopChoiceBar}^{\uparrow}$ *carrier reads bar-controlled choice data only through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{CauchyModulus}^{\uparrow} \longmapsto A \longmapsto W, L \longmapsto \mathsf{ChoiceFreeLimitSchedule}^{\uparrow}.$$

The route may expose finite bar coverage and selected-window ledgers for inspected prefixes. It does not export countable choice, a full fan theorem, $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, host stream equality, or a selected global limit.*

Proof. Start with an accepted packet $B = (S, M, A, W, L, D, H, C, P, N)$. A completion-facing request first reads the stream source S and the Cauchy-modulus threshold M, so the inspected prefix family and precision bound are fixed before any selection row is visible.

Next read the bar row A and the selected-window data W, L. The bar row supplies finite coverage for those inspected prefixes, and the ledger records only the windows justified by visible bar hits. This is the point where local bar-controlled choice is available.

Finally replay the selected windows through the choice-free limit schedule and the structural rows D, H, C, P, N. The lawless-prefix row keeps the stream access finite, and the replay/provenance/name rows preserve the same packet. None of these rows contains a coordinate for countable choice, a full fan theorem, quotient stream equality, host equality, selected global limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no completion-facing consumer can export them. □

Falsifiable prediction: Every accepted $\mathsf{BishopChoiceBar}^{\uparrow}$ completion-facing read factors through a StreamNameUp source row, CauchyModulusUp request row, finite bar-admission row, selected-window row, selected-window ledger, lawless-prefix compatibility row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, cauchymodulus, choice-free-limit-schedule, fantheorem, lawlesssequence: $\mathsf{BishopChoiceBar}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite bar coverage, selected-window ledgers, lawless-prefix compatibility, transport, replay, provenance, and local NameCert rows for the L10 completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopChoiceBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopChoiceBar}^{\uparrow}$ packet is generated from a StreamNameUp source row, CauchyModulusUp request row, finite bar-admission row, selected-window row, selected-window ledger, lawless-prefix compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopChoiceBar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6677.1, Definition 6677.2, and Definition 6677.3 seed the finite bar-controlled local choice surface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopChoiceBar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, the full fan theorem, $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected global limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar coverage, selected-window ledger exactness, lawless-prefix compatibility, transport, replay, provenance, and local naming for BEDC.Derived.BishopChoiceBarUp.

Chapter 6678

A Concrete Naming Certificate for HyperbolicBoundaryShadowTransport$^{\uparrow}$

The HyperbolicBoundaryShadowTransport$^{\uparrow}$ packet realizes the boundary-shadow transport branch of chapter 9463 as finite NameCert rows. It sits after the boundary-shadow carrier of chapter 3813, the boundary-transport route of chapter 6595, the ray-boundary surface of chapter 6470, the disk metric readback of chapter 6209, and the terminal Real-name boundary of chapter 38. The packet names a transported boundary shadow through displayed disk, boundary, metric, and Real-facing rows; it is not an interior-center collapse, a selected ideal endpoint, quotient boundary equality, physical observer motion, or global hyperbolic compactification.

Falsifiable prediction: Every accepted HyperbolicBoundaryShadowTransport$^{\uparrow}$ read is refuted if it collapses a boundary-shadow transport to an interior-center row without displaying HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicBoundaryTransport$^{\uparrow}$, GromovBoundary$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, Real$^{\uparrow}$, nontriviality, field-faithful metric readback, componentwise hsame transport, Cont replay, provenance, and local NameCert rows.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-boundary-transport, gromov-boundary, poincare-disk-metric, real: the boundary-shadow sibling records shadow readout without transport alignment; the boundary-transport sibling records fixed-boundary transport without shadow readout; the Gromov-boundary sibling records ray-boundary equivalence without disk-shadow transport; the Poincare disk metric sibling records metric readback without transported boundary shadows; and the Real sibling seals numerical data without a hyperbolic boundary carrier. None is bijective to HyperbolicBoundaryShadowTransport$^{\uparrow}$.

Definition 6678.1 (Hyperbolic boundary shadow transport carrier)**.** *A* HyperbolicBoundaryShadowTransport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, U, G, D, E, L, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row naming the boundary-shadow readout attached to displayed disk observations.*
2. *U is the* HyperbolicBoundaryTransport$^{\uparrow}$ *row transporting the boundary commitment while preserving the visible far-end boundary reading.*
3. *G is the* GromovBoundary$^{\uparrow}$ *row carrying the ray-boundary token through the transported shadow.*

4. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric readback row used to refuse interior-center collapse.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the metric and boundary-shadow readback.*
6. *L is the nontrivial and field-faithful ledger: it records that the disk metric read is nonzero on the transported boundary-shadow separation and that the Real-facing metric coordinate is read through the displayed field-faithful route rather than through a collapsed center.*
7. *F is the refusal row, excluding interior-center collapse, selected ideal endpoints, quotient boundary equality, physical observer motion, and global hyperbolic compactification.*
8. *H records componentwise* hsame *transport for boundary-shadow, boundary-transport, Gromov-boundary, disk-metric, Real-seal, ledger, refusal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from shadow readout through boundary transport, Gromov-boundary token, disk-metric readback, Real seal, nontrivial field-faithful ledger, refusal, provenance, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, U, G, D, E, L, F, H, C, P, N$. It has no coordinate for an interior-center collapse, selected ideal endpoint, quotient boundary equality, physical observer motion, completed hyperbolic compactification, Lean verification, or bridge checking.

Theorem 6678.2 (Hyperbolic boundary shadow transport NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}$ *carrier $T = (S, U, G, D, E, L, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, boundary-transport preservation, Gromov-boundary token exposure, Poincare disk metric readback, Real-seal handoff, nontrivial and field-faithful metric ledger, refusal exactness, componentwise transport, displayed replay, package provenance, and local naming through $S, U, G, D, E, L, F, H, C, P, N$.*

Every public read of the transported shadow factors through

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longmapsto \mathsf{HyperbolicBoundaryTransport}^{\uparrow} \longmapsto \mathsf{GromovBoundary}^{\uparrow} \longmapsto \mathsf{PoincareDiskMetric}^{\uparrow} \longmapsto$$

together with the nontrivial field-faithful ledger and the structural $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ *rows.*

Proof. Project the carrier of Definition 6678.1. Rows S, U, G, D, E are the visible geometric and Real-facing route, while L, F, H, C, P, N record ledger, refusal, transport, replay, provenance, and naming.

Read the boundary-shadow row S first and transport it through U. The Gromov-boundary token G is then the only boundary token available to the transported shadow route.

The disk-metric and Real-facing rows are downstream. Row D supplies the $\mathsf{PoincareDiskMetric}^{\uparrow}$ readback, row E seals the numerical boundary, and row L records the nontrivial field-faithful route expected by the future TasteGate surface.

Finally use F, H, C, P, N to repack the public certificate. Because the excluded objects have no coordinate in the carrier, the packet cannot collapse the transported shadow to an interior center or export quotient boundary equality, physical observer motion, selected ideal endpoints, or global compactification. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow, HyperbolicBoundaryTransport, GromovBoundary, PoincareDiskMetric, RealUp seal, nontrivial field-faithful ledger, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6678.1, Definition 6678.2, the falsifiable prediction, and the independence witness seed the finite boundary-shadow transport route over shadow, transport, boundary, disk-metric, Real-seal, ledger, refusal, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close interior-center collapse, selected ideal endpoints, quotient boundary equality, physical observer motion, global hyperbolic compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shadow admission, boundary-transport preservation, Gromov-boundary token exposure, Poincare disk metric readback, Real-seal handoff, nontrivial and field-faithful ledger exactness, refusal coverage, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBoundaryShadowTransportUp.

Chapter 6679

A Concrete Naming Certificate for $\mathsf{MorseBottLemma}^{\uparrow}$

Following Bott's nondegenerate critical-manifold form and standard Morse theory, the $\mathsf{MorseBottLemma}^{\uparrow}$ packet records the local normal-form route for a nondegenerate critical submanifold as finite BEDC certificate data. It sits beside the isolated critical-point packet of chapter 6418, the Morse-theory surface of chapter 3533, the manifold chart rows of chapter 96, the derivative ledger of chapter 122, the vector-space coordinate carrier of chapter 47, and the terminal real-name seal of chapter 38. The packet names only displayed critical-submanifold, normal-bundle, transverse Hessian, tubular-coordinate, quadratic normal-form, transport, replay, provenance, and local naming rows; it does not import a host smooth tubular-neighbourhood theorem, quotient germs, global flow data, or analytic compactness.

Definition 6679.1 (Morse–Bott lemma carrier)**.** *A* $\mathsf{MorseBottLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, S, N, Q, T, L, H, C, R, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *chart-source row, naming the displayed neighbourhood, chart window, coordinate dimension, and local domain in which the critical submanifold is inspected.*
2. *S is the critical-submanifold row, recording the displayed submanifold chart, tangent directions, normal directions, and derivative-vanishing witness along the submanifold through the* $\mathsf{Derivative}^{\uparrow}$ *ledger.*
3. *N is the normal-bundle row, storing the finite transverse coordinate carrier and the projection from the ambient chart to the normal coordinates.*
4. *Q is the transverse Hessian-signature row, storing the nondegenerate normal quadratic form, its positive and negative normal counts, and the vector-space coordinate rows used to read them.*
5. *T is the tubular-coordinate row, recording the admitted local coordinate transformation from the ambient chart to submanifold coordinates together with normal coordinates.*
6. *L is the Morse–Bott normal-form row, displaying the equation that reads the function as its submanifold value plus the signed square sum in the normal coordinates determined by Q after T has been admitted.*

7. *H records componentwise* hsame *transport for chart, critical-submanifold, normal-bundle, transverse Hessian, tubular-coordinate, normal-form, replay, Real readback, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a normal-form query through $X, S, N, Q, T, L, H, R, P, A$.*
9. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row, exposing scalar values only after the chart, submanifold, normal-bundle, transverse Hessian, tubular-coordinate, and quadratic normal-form rows have been displayed.*
10. *P is the* Pkg *provenance over* $\mathsf{MorseBottLemma}^{\uparrow}$, $\mathsf{MorseLemma}^{\uparrow}$, $\mathsf{MorseTheory}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *A is the local* $\mathsf{NameCert}_{\mathsf{MorseBottLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, N, Q, T, L, H, C, R, P, A$. It has no coordinate for a host tubular-neighbourhood theorem, selected atlas, quotient germ equality, global compactness, or ambient smooth-manifold semantics.

Theorem 6679.2 (Morse–Bott lemma NameCert obligations)**.** *For every accepted* $\mathsf{MorseBottLemma}^{\uparrow}$ *carrier $B = (X, S, N, Q, T, L, H, C, R, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{MorseBottLemma}^{\uparrow}}$ *surface consists exactly of chart-source admission, critical-submanifold exposure, normal-bundle projection, transverse Hessian nondegeneracy, tubular-coordinate admission, quadratic normal-form readback, componentwise* hsame *transport, displayed* Cont *replay,* $\mathsf{Real}^{\uparrow}$ *readback,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6679.1. The displayed rows are exactly $X, S, N, Q, T, L, H, C, R, P, A$, so each accepted packet exposes the chart, critical-submanifold, normal-bundle, transverse Hessian, tubular-coordinate, and normal-form data as finite BEDC rows.

The normal-form row L is downstream of the transverse rows. It may be read only after the critical submanifold S, the normal-bundle projection N, the nondegenerate transverse Hessian row Q, and the tubular-coordinate row T have been displayed.

The structural rows H, C, R, P, A transport, replay, read back, source, and name that same finite route. Since no coordinate contains a host tubular-neighbourhood theorem, quotient germ equality, selected atlas, or ambient compactness theorem, the listed rows form the full local NameCert surface. □

Theorem 6679.3 (Morse–Bott critical-manifold handoff)**.** *Every accepted* $\mathsf{MorseBottLemma}^{\uparrow}$ *normal-form read specializes to the isolated critical-point route of chapter 6418 when the displayed critical submanifold row is a singleton and the normal-bundle row exhausts the ambient chart directions. In general, the handoff keeps the tangent directions inside the critical-submanifold row and sends only the transverse normal coordinates through the Hessian-signature and quadratic normal-form rows.*

Proof. Begin with the rows S and N of Definition 6679.1. The critical-submanifold row stores the tangent directions along which the function is constant to first order, while the normal-bundle row stores the transverse coordinate carrier.

If S is a singleton and N exhausts the ambient chart directions, the carrier has no remaining tangent directions. The rows $X, N, Q, T, L, H, C, R, P, A$ then present exactly the isolated chart, nondegenerate Hessian, local-coordinate, quadratic normal-form, transport, replay, Real readback, provenance, and naming route used by $\mathsf{MorseLemma}^{\uparrow}$.

For a positive-dimensional critical submanifold, the tangent rows remain in S and are not sent to the signed square sum. Only the normal coordinates pass through Q and L, so the packet records Bott's critical-manifold normal form without collapsing it to an isolated critical point. □

Falsifiable prediction: Every accepted $\mathsf{MorseBottLemma}^{\uparrow}$ normal-form read exposes chart, critical-submanifold, normal-bundle, transverse Hessian, tubular-coordinate, quadratic normal-form, componentwise transport, displayed replay, Real readback, provenance, and local NameCert rows before any Morse-theory consumer reads the local model.

Independence witness: morse-lemma, morse-theory, manifold, derivative, real, vecspace: $\mathsf{MorseBottLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a critical-submanifold row, normal-bundle projection, transverse Hessian signature, tubular-coordinate row, quadratic normal-form row, transport, replay, Real readback, provenance, and local naming as one finite packet.

The Lean follow-up namespace is expected at `BEDC.Derived.MorseBottLemmaUp`. It must preserve the critical-submanifold, normal-bundle, transverse Hessian, tubular-coordinate, normal-form, transport, replay, Real readback, provenance, and local NameCert rows before any checked marker may cite the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MorseBottLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{MorseBottLemma}^{\uparrow}$ packet is generated from manifold chart rows, critical-submanifold rows, normal-bundle projection rows, transverse Hessian rows, tubular-coordinate rows, quadratic normal-form rows, componentwise transport, displayed replay, Real readback, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MorseBottLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6679.1, Definition 6679.2, and Definition 6679.3 seed the finite critical-manifold normal-form route consumed by Morse-theory packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MorseBottLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host smooth-manifold Morse–Bott theory, selected tubular neighbourhoods, quotient germs, analytic compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-submanifold exposure, normal-bundle exactness, transverse Hessian coverage, tubular-coordinate admission, normal-form readback, transport stability, replay stability, Real readback, provenance exhaustion, and local naming for BEDC.Derived.MorseBottLemmaUp.

Chapter 6680

A Concrete Naming Certificate for $\mathsf{ChoquetIntegral}^{\uparrow}$

Following Choquet's capacity integral and Bishop–Bridges constructive measure theory, the $\mathsf{ChoquetIntegral}^{\uparrow}$ packet records the finite NameCert surface for integrating nonnegative simple-function envelopes against a displayed finite capacity. It consumes the measure carrier of chapter 92, the terminal real-name seal of chapter 38, and the compact-convex probability boundary of chapter 6611. The chapter names only capacity rows, simple-function envelope rows, finite cut-level ledgers, Real readback, transport, replay, provenance, and local naming; it does not assert countable additivity, a monotone-convergence theorem, quotient equality of functions, selected measurable representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6680.1 (Choquet integral carrier)**.** *A* $\mathsf{ChoquetIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, A, S, L, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row naming the measurable carrier, measurable-set requests, and finite measure-read interface visible to the packet.*
2. *A is the finite capacity row, recording monotone set-capacity observations over the displayed measurable rows without adding countable additivity.*
3. *S is the simple-function envelope row, a finite list of level values and measurable support rows used to bound the integrand from below and above.*
4. *L is the cut-level ledger, recording the inspected threshold rows, support inclusions, and finite envelope comparisons used by the Choquet read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *handoff row, sealing finite sums, capacity weights, and envelope bounds as displayed real-name data.*
6. *H records componentwise* hsame *transport for measure, capacity, simple envelope, cut-level, Real handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a capacity-integral request through the same finite rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ChoquetIntegral}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ChoquetSimplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ChoquetIntegral}^\uparrow}$ *row.*

The classifier compares accepted packets only through $M, A, S, L, R, H, C, P, N$. It has no coordinate for an ambient integral value, countable additivity, selected measurable representatives, host almost-everywhere equality, quotient simple functions, or completeness of $\mathsf{Real}^\uparrow$.

Theorem 6680.2 (Choquet integral NameCert obligations)**.** *For every accepted* $\mathsf{ChoquetIntegral}^\uparrow$ *carrier $I = (M, A, S, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoquetIntegral}^\uparrow}$ *surface consists exactly of measure-source admission through M, finite capacity admission through A, simple-function envelope admission through S, cut-level ledger exactness through L, Real handoff through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Any Choquet-integral consumer must display those rows before exporting an integral-facing certificate.*

Proof. Project the carrier of Definition 6680.1. The measure row M is the only source for measurable-set requests, and the capacity row A is the only source for monotone finite set-capacity observations.

Read the simple-function envelope next. The row S supplies the finite level values and support rows, while L records the inspected cut levels and support-inclusion comparisons used by the Choquet read.

The Real handoff is downstream of those finite rows. The row R seals capacity weights, finite sums, and envelope bounds only after M, A, S, L have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no ambient integral value, countable-additivity theorem, selected measurable representative, host almost-everywhere equality, quotient simple-function equality, or $\mathsf{Real}^\uparrow$ completeness row, the listed rows exhaust the local certificate surface. □

Theorem 6680.3 (Choquet integral capacity handoff)**.** *Every* $\mathsf{Real}^\uparrow$ *value exported by an accepted* $\mathsf{ChoquetIntegral}^\uparrow$ *carrier factors through the finite route*

$$\mathsf{Measure}^\uparrow, \mathsf{ChoquetSimplex}^\uparrow \longmapsto \mathsf{ChoquetIntegral}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The route first displays the measure source M, finite capacity row A, simple-function envelope row S, and cut-level ledger L; only then may the Real handoff row R, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row export a real-name boundary. The handoff exports no monotone-convergence theorem, countable-additive extension, selected representative, quotient equality, or ambient completeness claim.*

Proof. Start with the carrier inventory of Definition 6680.1. The only analytic rows before the real-name boundary are M, A, S, L: measure-source observations, finite capacity observations, simple-function envelopes, and cut-level ledgers.

Use the NameCert obligations of Definition 6680.2. They force Real readback through R only after the displayed finite capacity and envelope route is available.

The Choquet simplex sibling supplies the compact-convex probability boundary cited by the provenance row, while the measure and Real siblings supply the measurable-source and terminal real-name rows. Transport, replay, provenance, and local naming preserve that same route, so a consumer cannot replace it by host measure theory or by an unrecorded real completeness principle. □

Falsifiable prediction: A consumer that claims a $\mathsf{ChoquetIntegral}^\uparrow$ real-name output while omitting the finite capacity row, simple-function envelope row, cut-level ledger, $\mathsf{Real}^\uparrow$ handoff, hsame, Cont, Pkg, or $\mathsf{NameCert}$ rows refutes the carrier of Definition 6680.1.

Independence witness: measure, real, choquet-simplex: $\mathsf{ChoquetIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries measure-source rows, finite capacity rows, simple-function envelope rows, cut-level ledgers, Real handoff, transport, replay, provenance, and local naming while refusing countable additivity, selected representatives, quotient simple functions, and ambient real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoquetIntegral}^{\uparrow}$ packet is generated from a MeasureUp source, a finite capacity row, a simple-function envelope row, a cut-level ledger, a RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoquetIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6680.1, Definition 6680.2, and Definition 6680.3 seed the capacity-integral certificate over measure, capacity, simple-envelope, cut-level, Real handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoquetIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable additivity, monotone convergence, selected measurable representatives, quotient equality of simple functions, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, measure-source exposure, finite capacity monotonicity, simple-function envelope exactness, cut-level ledger exactness, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetIntegralUp.

Chapter 6681

A Concrete Naming Certificate for $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$ packet presents the rational grid used by countable real approximation as finite rational-window data. It sits over the $\mathsf{Rat}^{\uparrow}$ row of chapter 37 and the dyadic arithmetic ledger of chapter 291, feeds the finite windows of chapter 126, and supplies the rational readback surface used by chapter 301 and chapter 38. The packet is a constructive grid certificate; it is not a Dedekind-cut presentation, not a quotient class of Cauchy names, not a countable-choice schedule, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 6681.1 (Bishop countable dense rational grid carrier)**.** *A* $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *rational-coordinate row, storing the visible numerator, denominator, sign, and comparison data used by the current grid window.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, recording dyadic tolerances, midpoint steps, and interval refinements compatible with Q.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *request-window row. It records which rational grid points and tolerance scales are currently inspected.*
4. *S is the successor-window row. It lists the finite neighboring rational grid candidates obtained by one dyadic refinement of the current window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, translating each inspected grid window into regular rational readback data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *boundary row. It records only the finite real-window comparisons needed to read a candidate as a real approximation.*
7. *H records componentwise* hsame *transport for rational coordinates, dyadic scales, StreamName windows, successor windows, regular-rational handoff, real-boundary reads, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a grid request through Q, D, W, S, R, E.*

9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $Q, D, W, S, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient equality of Cauchy names, countable choice, host enumeration of all rationals, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6681.2 (Bishop countable dense rational grid NameCert obligations)**.** *For every accepted* $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$ *carrier $G = (Q, D, W, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCounta}}$ *surface consists exactly of carrier admission for the ten displayed rows, rational-coordinate visibility through Q, dyadic scale compatibility through D, finite StreamName request coverage through W, one-step neighbor coverage through S, regular rational readback through R, finite Real-window boundary through E, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6681.1. The only positive approximation data are the rational coordinate row Q, the dyadic scale row D, the finite request window W, and the successor-window row S.

Perform finite window induction over W. The base window reads the currently displayed rational coordinate and dyadic tolerance. The step case refines the dyadic scale once, reads the neighboring rational candidates in S, and keeps each candidate attached to the same rational and dyadic rows.

The readback rows are downstream. The row R translates the displayed grid window into regular rational data before E exposes any finite real-boundary comparison. Transport, continuation, provenance, and local naming through H, C, P, N package that route and do not add any hidden cut, quotient, choice, global enumeration, or completeness coordinate. □

Theorem 6681.3 (Bishop countable dense rational grid Real-window handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$ *packet must pass through the finite route*

$$Q, D, W, S, R, E \longmapsto H, C, P, N.$$

Thus a real-window consumer may read rational grid points, dyadic tolerances, regular-rational readback, and finite real-boundary comparisons, but it receives no Dedekind cut, no quotient class of Cauchy names, no countable-choice schedule, no host enumeration of all rationals, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a downstream request. Carrier admission exposes Q, D, W, S as finite grid data: a rational coordinate, a dyadic tolerance, the currently inspected StreamName window, and the neighboring candidates obtained by the displayed refinement step.

Replay the request through C. The regular-rational handoff R is read only after the rational and dyadic rows have been displayed, and the Real boundary row E is read only as a finite comparison attached to that same handoff.

Finally transport and package the route through H, P, N. Since the carrier has no field for cuts, quotient equality, countable choice, host enumeration, or completeness, the handoff remains a finite rational-grid window into $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6681.1, Definition 6681.2, and Definition 6681.3 seed the finite rational-grid window over $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient equality of Cauchy names, countable choice, host enumeration of all rationals, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-coordinate visibility, dyadic scale compatibility, finite StreamName coverage, regular-rational handoff, Real-window nonescape, and local NameCert exactness for the same packet.

Constructive story: A $\mathsf{BishopCountableDenseRationalGrid}^{\uparrow}$ packet is generated from Rat coordinates, DyadicRatCore tolerance scales, finite StreamName request windows, successor grid windows, RegSeqRat readback, Real boundary rows, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 6682

A Concrete Naming Certificate for $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$

Bishop–Bridges constructive analysis and signed-digit interval methods read bisection as a finite schedule of inspected endpoint windows, not as a selected point of an ambient complete real line. The $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$ packet sits between the signed-digit interval carrier of chapter 3830, the finite signed-digit window surface of chapter 316, the signed-digit real endpoint route of chapter 363, the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the closed bounded interval carrier of chapter 1591. The packet records interval bisection as finite NameCert data for Real and Completion consumers; it does not form quotient streams, choose representatives, invoke countable choice, use trichotomy, or import ambient Real completeness.

Definition 6682.1 (Signed digit interval bisection carrier)**.** *A* $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, K, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower signed-endpoint row, carrying the finite signed-digit endpoint windows admitted by* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *and* $\mathsf{SignedDigitStream}^{\uparrow}$*.*
2. *U is the upper signed-endpoint row, carrying the matching finite signed-digit endpoint windows for the upper side of the interval.*
3. *K is the bisection schedule row, recording the finite left/right bisection decisions and the midpoint requests already inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure ledger, recording normalized dyadic endpoint and midpoint rows for the scheduled bisection cuts.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the endpoint and midpoint windows opened by the schedule.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the normalized signed endpoints and midpoint enclosures.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the signed endpoints, bisection schedule, dyadic enclosure ledger, windows, and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for signed endpoint rows, bisection schedules, dyadic enclosure rows, window rows, readback rows, seal rows, replay rows, provenance, and local naming.*
9. *C records displayed* Cont *replay from a bisection request through signed endpoint windows, dyadic midpoint normalization, enclosure refinement, regular readback, and the terminal seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$, $\mathsf{SignedDigitIntervalReal}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDigitReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, D, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a chosen infinite signed-digit expansion, real trichotomy, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6682.2 (Signed digit interval bisection NameCert obligations)**.** *For every accepted* $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$ *carrier*

$$B = (L, U, K, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalBisection}^{\uparrow}}$ *obligations are exhausted by carrier admission for the displayed rows; signed-endpoint admission through L, U; bisection-schedule admission through K; dyadic midpoint and enclosure admission through D; finite-window admission through W; regular rational readback through R; terminal Real sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

No obligation may call on quotient streams, chosen representatives, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Proof. Project the carrier of Definition 6682.1. The endpoint rows L, U and the schedule row K are the only sources for bisection data, and the window row W records the finite observations allowed by that schedule.

The dyadic enclosure row D normalizes endpoint and midpoint observations before any readback is consumed. Thus the bisection route is ledger-first: midpoint refinement is displayed as dyadic enclosure data rather than as a host order comparison or an infinite choice of sides.

The regular readback and seal rows R, E then receive only the displayed enclosure route. The structural rows H, C, P, N transport, replay, source, and name the same packet, so the listed rows exhaust the local certificate and leave the excluded principles outside the carrier. □

Theorem 6682.3 (Signed digit interval bisection nested interval route)**.** *Every accepted* SignedDigitIntervalBisectior *packet routes its bisection schedule as a finite nested closed-interval read*

$$\mathsf{SignedDigitIntervalReal}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The route keeps each midpoint refinement attached to the same signed endpoint rows and the same closed bounded interval provenance before any Real-facing consumer reads the terminal seal.

Proof. Start with the signed endpoint rows. The interval source is supplied by $\mathsf{SignedDigitIntervalReal}^{\uparrow}$, with the lower and upper endpoint windows carried by L and U.

Apply the bisection schedule row K and the dyadic enclosure ledger D. Each scheduled midpoint request refines the displayed closed interval by a finite dyadic row, so nestedness is witnessed by endpoint and midpoint enclosure data rather than by a completed intersection point.

Finally read through W, R, E, H, C, P, N. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback keep the nested route finite, while the $\mathsf{Real}^{\uparrow}$ seal and structural rows preserve the same packet for downstream consumers. The route imports no quotient stream, chosen representative, countable-choice schedule, trichotomy branch, or ambient Real completeness theorem. □

Theorem 6682.4 (Signed digit interval bisection Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$ *packet can bypass the displayed route*

$$L, U, K, D, W, R, E, H, C, P, N.$$

Thus every Real-facing consumer receives only signed endpoint rows, the finite bisection schedule, the dyadic enclosure ledger, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local naming; it receives no quotient stream equality, chosen infinite expansion, selected representative, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Proof. Begin with carrier projection. The terminal seal row E is not a detached real object; it is displayed only after L, U, K, D, W, R have been read in the accepted packet.

Use the nested interval route above. It identifies the path to the seal with signed endpoints, scheduled bisection decisions, dyadic midpoint enclosures, finite windows, and regular rational readback.

The structural rows H, C, P, N then preserve and name that same route. Since the carrier contains no quotient stream coordinate, representative chooser, countable-choice schedule, trichotomy row, compactness object, or ambient completeness witness, none of those objects can escape through the terminal seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDigitIntervalBisection}^{\uparrow}$ packet is generated from signed endpoint rows, a bisection schedule, DyadicRatCore enclosure rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SignedDigitIntervalBisection}^{\uparrow}}$, seed)

Scope closed: Definition 6682.1, Definition 6682.2, Definition 6682.3, and Definition 6682.4 seed the finite signed-digit interval bisection route from endpoint windows to RealUp seal data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SignedDigitIntervalBisection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, chosen signed-digit representatives, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bisection schedule admission, dyadic enclosure exactness, regular readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SignedDigitIntervalBise

Chapter 6683

A Concrete Naming Certificate for $\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley completion language, the ChoiceFreeDisplayedComple packet records the displayed limit interface used by completion arguments that inspect finite Cauchy data without quotienting names or selecting representatives. It sits over the $\mathsf{Cauchy}^{\uparrow}$ carrier of chapter 3413, the $\mathsf{StreamName}^{\uparrow}$ finite windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the sequence-completeness service of chapter 1515, the complete-metric boundary of chapter 128, the metric-completion route of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a finite displayed-completion landing surface while refusing quotient completion equality, selected representatives, countable-choice schedules, ambient completeness, and any claim that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 6683.1 (Choice-free displayed completion carrier)**.** *A* $\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, S, R, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row naming the admitted carrier, distance observations, and finite metric request histories.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the Cauchy name under inspection.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows.*
4. *M is the modulus-threshold row assigning each requested tolerance a displayed late index.*
5. *L is the displayed limit row offered to* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers only after X, S, R, M have been read.*
6. *E is the late-distance ledger. For each displayed tolerance and every sufficiently late window, it records the metric-distance comparison from the readback row to L.*
7. *H records componentwise* hsame *transport for the metric source, stream windows, regular readback, modulus threshold, displayed limit, late-distance ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through X, S, R, M, E, L and then through the structural rows.*

9. *P is the* Pkg *provenance over* ChoiceFreeDisplayedCompletion$^\uparrow$, Cauchy$^\uparrow$, Met$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, SequentiallyCompleteMetric$^\uparrow$, CompleteMetric$^\uparrow$, MetricCompletion$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{ChoiceFreeDisplayedCompletion}^\uparrow}$ *row.*

The carrier has no coordinate for quotient completion equality, a selected representative of a Cauchy class, a countable-choice schedule, ambient completeness, or a proof that Real$^\uparrow$ *is complete.*

Theorem 6683.2 (Choice-free displayed completion NameCert obligations)**.** *For an accepted* ChoiceFreeDisplayedCompletion$^\uparrow$ *carrier $D = (X, S, R, M, L, E, H, C, P, N)$, the local* NameCert$_{\text{ChoiceFreeDisplayedComple}}$ *surface consists exactly of carrier admission for the metric source, stream-window, regular-rational readback, modulus-threshold, displayed-limit, late-distance, transport, replay, provenance, and naming rows; classifier stability under componentwise* hsame *transport; modulus coverage for every displayed tolerance request; late-distance exactness for the displayed limit; consumer handoff to* SequentiallyCompleteMetric$^\uparrow$, CompleteMetric$^\uparrow$, MetricCompletion$^\uparrow$, *and* Real$^\uparrow$ *through the same rows; and ledger refusal for quotient completion equality, selected representatives, countable-choice schedules, ambient completeness, and completeness of* Real$^\uparrow$.

Proof. Project the carrier definition. The packet exposes exactly the rows $X, S, R, M, L, E, H, C, P, N$, so carrier admission and the local naming row are read from the displayed tuple.

Classifier stability is componentwise. Any accepted transport first identifies the metric source, stream windows, regular readback, modulus threshold, displayed limit, late-distance ledger, replay, provenance, and local name through H before the packet can be compared.

Modulus coverage and late-distance exactness are read in sequence. A tolerance request reaches the displayed threshold row M, then inspects the windows S and readback R, and only then reads the ledger E comparing late values with L inside the metric source X.

The consumer handoff is the same finite route replayed by C and sourced by P, N. Since none of the rows stores quotient equality, a selected representative, a choice schedule, an ambient complete carrier, or a completeness theorem for Real$^\uparrow$, the ledger refusal follows from carrier exhaustion. □

Theorem 6683.3 (Choice-free displayed completion limit handoff)**.** *Every completion-facing consumer of an accepted* ChoiceFreeDisplayedCompletion$^\uparrow$ *packet $D = (X, S, R, M, L, E, H, C, P, N)$ receives the displayed limit only through the route*

$$X, S, R, M, E \longmapsto L \longmapsto H, C, P, N.$$

The handoff supplies metric source data, stream windows, regular rational readback, a modulus threshold, a displayed limit, a late-distance ledger, hsame *transport,* Cont *replay,* Pkg *provenance, and a local* NameCert *row. It does not supply quotient completion equality, selected representatives, countable-choice schedules, ambient completeness, or a claim that* Real$^\uparrow$ *is complete.*

Proof. Start with a completion-facing request. The metric source row X fixes the distance observations available to the request, and the stream-window row S fixes the inspected Cauchy data.

Read R and M next. The regular rational row supplies the finite readback of the displayed windows, while the modulus row supplies the late threshold for the requested tolerance.

The late-distance ledger E is then the only row that compares late readbacks with the displayed limit row L. Thus a SequentiallyCompleteMetric$^\uparrow$, CompleteMetric$^\uparrow$, MetricCompletion$^\uparrow$, or Real$^\uparrow$ consumer can reach L only after the metric, stream, readback, threshold, and distance-ledger rows are visible.

Finally H, C, P, N transport, replay, source, and name exactly that route. The carrier has no further coordinate, so the handoff cannot be read as quotient completion equality, a representative selector, a countable-choice schedule, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$. □

Independence witness: cauchy, streamname, regseqrat, sequentiallycompletemetric, completemetric, metriccompletion, real: $\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric source data, stream windows, regular rational readback, a modulus threshold, a displayed limit, a late-distance ledger, transport, replay, provenance, and local NameCert rows as one choice-free displayed completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}$ packet is generated from metric source rows, Cauchy StreamName windows, RegSeqRat readback, displayed modulus thresholds, displayed limit rows, late-distance ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6683.1, Definition 6683.2, and Definition 6683.3 seed the finite displayed-completion route over Metric, Cauchy, StreamName, RegSeqRat, SequentiallyCompleteMetric, CompleteMetric, MetricCompletion, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceFreeDisplayedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable-choice schedules, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed completion route.

Chapter 6684

A Concrete Naming Certificate for $\mathsf{SignedDigitRealExpansion}^{\uparrow}$

Following Bishop and Bridges and the signed-digit representation tradition in constructive analysis, the $\mathsf{SignedDigitRealExpansion}^{\uparrow}$ packet records a real name as finite signed-digit prefix windows with dyadic error control. It sits beside the finite bisection route of chapter 6682, the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a non-quotient signed-digit route from finite prefixes to regular Cauchy data; it does not select an infinite expansion, quotient streams, invoke countable choice, use trichotomy, or assert ambient Real completeness.

Definition 6684.1 (Signed digit real expansion carrier)**.** *A* $\mathsf{SignedDigitRealExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, W, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the signed-digit alphabet row, admitting only the digits $-1, 0, 1$ and the finite prefix words already opened by the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording which signed-digit prefixes are visible at the current inspection depth.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *value-and-radius ledger attached to each visible prefix.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reading the dyadic prefix ledger as a regular rational Cauchy name.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the signed-digit prefix, window, dyadic, and regular-readback rows have been displayed.*
6. *H records componentwise* hsame *transport for prefix words, windows, dyadic ledgers, regular readback, the Real seal, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a prefix request through finite signed-digit windows, dyadic normalization, regular readback, and the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{SignedDigitRealExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SignedDigitRealExpansion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, D, Q, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite signed-digit representative, countable choice, real trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6684.2 (Signed digit real expansion prefix stability)**.** *For every accepted* $\mathsf{SignedDigitRealExpansion}^{\uparrow}$ *carrier $E = (S, W, D, Q, R, H, C, P, N)$, extending the visible signed-digit prefix by one admitted digit preserves the same Real-facing route only when the enlarged prefix remains in S, the window row W records the enlarged inspection depth, and the dyadic ledger D records the refined value-and-radius row. The extension is then transported through Q, R, H, C, P, N and exports no chosen infinite expansion or quotient equality of streams.*

Proof. Project the carrier of Definition 6684.1. Prefix data live in S, and visible inspection depth lives in W, so a prefix extension cannot be read unless both rows display the enlarged finite word and its window.

Use the dyadic ledger row D. It supplies the normalized dyadic value and radius for the enlarged prefix before any regular rational readback is available.

Finally read through Q, R, H, C, P, N. Regular readback and the Real seal preserve the same finite prefix route, while transport, replay, provenance, and local naming add no row for an infinite representative, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness. □

Theorem 6684.3 (Signed digit real expansion RegSeqRat handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{SignedDigitRealExpansion}^{\uparrow}$ *packet must pass from finite signed-digit prefixes through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D to the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q before the* $\mathsf{Real}^{\uparrow}$ *seal R is consumed. Thus the signed-digit expansion route hands off to regular Cauchy data as the finite BEDC row family*

$$S, W, D, Q, R, H, C, P, N.$$

It is not a quotient construction of real numbers, not a selected stream representative, not a countable-choice principle, and not an ambient completeness theorem.

Proof. Begin with the finite prefix and window rows S, W. They record only the signed digits and the finite inspection depths currently visible to the consumer.

Apply the dyadic ledger D. Each prefix is read as a dyadic value with an explicit radius, so the next consumer receives toleranced rational data rather than a host real value.

The handoff to Q packages those dyadic rows as a $\mathsf{RegSeqRat}^{\uparrow}$ readback. Only after this regular rational row is displayed may the terminal seal R be consumed.

The structural rows H, C, P, N preserve the same route. Since the carrier has no coordinate for quotient streams, selected infinite representatives, countable choice, trichotomy, host real equality, or ambient completeness, the handoff remains a finite signed-digit-to-regular-Cauchy route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitRealExpansion}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDigitRealExpansion}^{\uparrow}$ packet is generated from signed-digit alphabet rows, finite StreamName prefix windows, DyadicRatCore value-and-radius rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SignedDigitRealExpansion}^{\uparrow}}$, seed)

Scope closed: Definition 6684.1, Definition 6684.2, and Definition 6684.3 seed the finite signed-digit prefix route from window inspection to regular Cauchy Real-seal data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SignedDigitRealExpansion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected infinite signed-digit representatives, countable choice, trichotomy, host real equality, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window stability, dyadic value-and-radius exactness, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SignedDigitRealExpa

Chapter 6685

A Concrete Naming Certificate for $\mathsf{RegularCauchyArithmetic}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ packet records the finite arithmetic surface used when regular Cauchy names are added, multiplied, negated, compared against displayed dyadic tolerances, and sealed for $\mathsf{Real}^{\uparrow}$ consumers without quotienting the names. It sits over the $\mathsf{Cauchy}^{\uparrow}$ carrier of chapter 3413, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ finite windows of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives L10 Real and Completion consumers an arithmetic carrier between regular rational sequences and the Real seal while refusing quotient real equality, selected representatives, trichotomy, countable-choice schedules, ambient completeness, and unrestricted field algebra over host reals.

Definition 6685.1 (Regular Cauchy arithmetic carrier)**.** *A* $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, S, T, R, Q, D, O, W, E, H, C, P, N)$$

with the following rows:

1. *X and Y are the two* $\mathsf{Cauchy}^{\uparrow}$ *source-name rows admitted for arithmetic.*
2. *S and T are the corresponding* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows.*
3. *R and Q are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed radii used for arithmetic error budgets.*
5. *O is the operation row naming which of addition, multiplication, unary negation, distance comparison, or constant insertion is being performed.*
6. *W is the regular-Cauchy output window row produced from R, Q, D, O.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the source names, stream windows, rational readbacks, tolerance ledger, operation row, and output window have been read.*
8. *H records componentwise* hsame *transport for the source names, stream windows, regular readbacks, tolerance ledger, operation row, output window, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an arithmetic request through $X, Y, S, T, R, Q, D, O, W, E$ and then through the structural rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyArithmetic}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyArithmetic}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of Cauchy names, selected representatives, trichotomy, countable-choice schedules, ambient completeness, or host-real field algebra.

Theorem 6685.2 (Regular Cauchy arithmetic NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ *carrier $A = (X, Y, S, T, R, Q, D, O, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyArithmetic}^{\uparrow}}$ *surface consists exactly of carrier admission for the two Cauchy source names, their finite stream windows, their regular rational readbacks, the dyadic tolerance ledger, the displayed operation row, the regular-Cauchy output window, the terminal Real seal, transport, replay, provenance, and local naming; classifier stability under componentwise* hsame *transport; arithmetic tolerance coverage for the operation named by O; exact routing from R, Q, D, O into W; Real-facing nonescape through E; and ledger refusal for quotient name equality, selected representatives, trichotomy, countable-choice schedules, ambient completeness, and host-real field algebra.*

Proof. Project the carrier definition. The rows $X, Y, S, T, R, Q, D, O, W, E, H, C, P, N$ are the only displayed coordinates, so carrier admission, operation routing, provenance, and local naming are read directly from the packet.

Classifier stability is componentwise. Transport through H must preserve both source names, both stream windows, both regular readbacks, the dyadic tolerance ledger, the operation row, the output window, the Real seal, replay, provenance, and local naming before two arithmetic packets can be identified.

Arithmetic coverage is finite. The operation row O selects the displayed arithmetic task, the tolerance ledger D supplies the requested dyadic budget, and the regular readbacks R, Q provide the rational data from which W is built.

Finally E, C, P, N seal, replay, source, and name the same route. Since no carrier coordinate stores quotient equality, selected representatives, trichotomy, countable choice, ambient completeness, or host-real field algebra, the refusal clauses follow by carrier exhaustion. □

Theorem 6685.3 (RegSeqRat route for regular Cauchy arithmetic)**.** *Every accepted* $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ *arithmetic read factors through the displayed route*

$$\mathsf{Cauchy}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ O,\ W.$$

In carrier notation, the output window W can be consumed only after X, Y, S, T, R, Q, D, O are visible. Thus downstream $\mathsf{RegSeqRat}^{\uparrow}$ *consumers receive the two source readbacks, the dyadic error ledger, the operation row, transport, replay, provenance, and local naming, but receive no quotient stream equality, selected representative, external rational sequence, trichotomy, countable-choice schedule, or ambient completeness witness.*

Proof. Start with an arithmetic request. The two source rows X, Y admit only Cauchy-name data, while S, T determine the finite windows inspected by the request.

The request then reads R, Q and D. These rows supply regular rational readback and dyadic tolerance budgets before the operation row O is applied.

The output window W is therefore produced inside the displayed route, and H, C, P, N transport, replay, source, and name exactly that route. No additional carrier coordinate is available for quotient stream equality, selected representatives, external rational streams, trichotomy, choice schedules, or ambient completeness. □

Theorem 6685.4 (Regular Cauchy arithmetic Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ *packet must expose the two Cauchy source rows, both stream-window rows, both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the displayed operation row, and the regular-Cauchy output window before the terminal seal row* E *is read. The Real seal exports the named arithmetic packet and refuses quotient real equality, selected representatives, trichotomy, countable-choice schedules, ambient completeness, and unrestricted host-real algebra.*

Proof. Use the NameCert obligations to order the visible rows. A Real-facing read starts with X, Y, S, T, then consumes R, Q, D, O, and only then reaches the output window W.

The seal row E is terminal by definition of the carrier route. It can certify the displayed arithmetic packet for $\mathsf{Real}^{\uparrow}$ consumers, but it cannot introduce data absent from the preceding rows.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route. The excluded quotient, representative, trichotomy, choice, completeness, and host-algebra objects have no coordinate in the packet, so the seal cannot escape to them. □

Independence witness: cauchy, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two Cauchy source names, stream windows, regular rational readbacks, a dyadic arithmetic tolerance ledger, an operation row, an output window, a Real seal, transport, replay, provenance, and local NameCert rows as one arithmetic packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyArithmetic}^{\uparrow}$ packet is generated from two Cauchy source rows, StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a displayed arithmetic operation row, a regular-Cauchy output window, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6685.1, Definition 6685.2, Definition 6685.3, and Definition 6685.4 seed the finite arithmetic route over Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyArithmetic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, trichotomy, countable-choice schedules, ambient completeness, host-real algebra, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the regular Cauchy arithmetic route.

Chapter 6686

A Concrete Naming Certificate for $\mathsf{CauchyOrderCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyOrderCompletion}^{\uparrow}$ packet records the order-aware Cauchy completion route used by L10 Real and Completion consumers as finite NameCert data. It reads the Cauchy carrier of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance arithmetic of chapter 291, the constructive order surface of chapter 4069, and the terminal real-name seal of chapter 38. The packet gives completion-facing consumers a displayed order-stable landing route, while refusing quotient completion equality, selected limits, trichotomy, host real equality, countable-choice schedules, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6686.1 (Cauchy order completion carrier)**.** *A* $\mathsf{CauchyOrderCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Cauchy}^{\uparrow}$ *source row, naming the accepted Cauchy packet whose completion-facing order route is being inspected.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the same Cauchy source.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger used to compare displayed readbacks.*
5. *Q is the order-comparison row, read through* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *and carrying only displayed lower, upper, and apartness-facing comparisons of the regular Cauchy names.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Cauchy source, stream windows, rational readback, dyadic tolerance ledger, and order-comparison row have been displayed.*
7. *H records componentwise* hsame *transport for $X, S, R, D, Q, E, C, P, N$.*
8. *C records displayed* Cont *replay from an order or completion request through X, S, R, D, Q, E and the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyOrderCompletion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyOrderCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, a selected limit, trichotomy, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6686.2 (Cauchy order completion NameCert obligations)**.** *For an accepted* $\mathsf{CauchyOrderCompletion}^{\uparrow}$ *carrier* $O = (X, S, R, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyOrderCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, stream-window row, RegSeqRat readback row, DyadicRatCore tolerance ledger, order-comparison row, Real seal, componentwise transport, displayed replay, provenance, and local naming. No local obligation may replace those rows by quotient completion equality, a selected limit, trichotomy, host real equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier definition. The packet exposes exactly $X, S, R, D, Q, E, H, C, P, N$, so carrier admission and local naming are reads of the displayed tuple.

The Cauchy and readback rows are opened first. The source row X fixes the accepted Cauchy packet, S fixes the inspected windows, R fixes the regular rational values, and D fixes the dyadic tolerance ledger.

The order-comparison and seal rows are downstream. The row Q reads only the displayed regular Cauchy order data, and E seals that same route for Real-facing consumers.

Finally H, C, P, N transport, replay, source, and name the route. Since the excluded host and quotient objects are absent from the carrier, they cannot become NameCert obligations. □

Theorem 6686.3 (Cauchy order completion regular seal route)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchyOrderCompletion}^{\uparrow}$ *packet reaches the terminal seal through the finite route*

$$\mathsf{Cauchy}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegularCauchyOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The seal is therefore a displayed completion boundary for regular Cauchy order data, not a quotient completion, selected representative, host equality test, or ambient completeness theorem.

Proof. Begin with the NameCert obligations above. They force every accepted read to expose the Cauchy source, stream windows, regular rational readback, dyadic tolerance ledger, order-comparison row, and structural rows before the Real seal is consumed.

The route is regular because R is the only readback row and Q reads order data only through the regular Cauchy order surface. The dyadic ledger D remains attached to those same readbacks.

The seal row E is terminal for this finite route. It does not add quotient completion equality, a representative selector, host real equality, or an ambient completeness proof because no such coordinate appears in the packet. □

Theorem 6686.4 (Cauchy order completion order stability)**.** *If an accepted* $\mathsf{CauchyOrderCompletion}^{\uparrow}$ *packet is transported componentwise through* hsame*, then its Cauchy source row, stream windows, RegSeqRat readback, DyadicRatCore tolerance ledger, order-comparison row, Real seal, replay, provenance, and local name move together as one order-stable completion route. In particular, the transported packet preserves displayed order comparisons without importing trichotomy, quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the transport row H. It is componentwise over the same carrier coordinates $X, S, R, D, Q, E, C, P, N$, so a transported packet cannot move the order-comparison row apart from the Cauchy source and readback rows.

The replay row C preserves the request route from X, S, R, D, Q to the terminal seal E. Provenance and local naming then source the transported packet through P, N.

The refused principles have no carrier coordinate and no replay row. Hence order stability is exactly stability of the displayed finite route under hsame transport. □

Theorem 6686.5 (Cauchy order completion Real consumer handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyOrderCompletion}^{\uparrow}$ *packet receives the displayed Cauchy source, finite stream windows, regular rational readback, dyadic tolerance ledger, order-comparison row, terminal Real seal, transport, replay, provenance, and local naming rows. The handoff exports no quotient completion equality, selected limit, host real equality, trichotomy principle, countable-choice schedule, or ambient completeness witness.*

Proof. Apply the regular seal route. It places the Real-facing endpoint after the Cauchy source, stream windows, regular readback, dyadic tolerance ledger, and order-comparison row.

Use order stability for transported reads. Any consumer that reaches the endpoint after transport still reads the same $X, S, R, D, Q, E, H, C, P, N$ route.

The carrier inventory supplies the exclusions. Since the packet contains no quotient equality, selected limit, host equality, trichotomy, choice, or completeness coordinate, the Real consumer receives only the displayed order-stable completion route. □

Falsifiable prediction: An accepted $\mathsf{CauchyOrderCompletion}^{\uparrow}$ read is refuted if a Real-facing order-completion consumer reaches the seal without first displaying the Cauchy source row, StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance ledger, RegularCauchyOrder comparison row, transport, replay, provenance, and local NameCert row.

Independence witness: cauchy, streamname, regseqrat, dyadicratcore, regularcauchyorder, real: $\mathsf{CauchyOrderCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds Cauchy source data, finite windows, regular rational readback, dyadic order tolerances, a regular Cauchy order-comparison row, and a terminal Real seal as one completion-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyOrderCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyOrderCompletion}^{\uparrow}$ packet is generated from Cauchy source rows, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance and comparison ledgers, RegularCauchyOrder comparison rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyOrderCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6686.1, Definition 6686.2, Definition 6686.3, Definition 6686.4, and Definition 6686.5 seed the order-aware Cauchy completion route over Cauchy, StreamName, RegSeqRat, DyadicRatCore, RegularCauchyOrder, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyOrderCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, trichotomy, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.CauchyOrderCompletionU

Chapter 6687

A Concrete Naming Certificate for $\mathsf{AttouchWetsConvergence}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsConvergence}^{\uparrow}$ packet records local bounded-window convergence of displayed closed metric figures as finite L10 NameCert data. It sits after the bounded observation topology of chapter 4358, the closed-set hit–miss route of chapter 6591, the metric source of chapter 57, the Hausdorff-distance boundary of chapter 3653, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed bounded windows, local distance ledgers, Kuratowski handoff rows, transport, replay, provenance, and local naming; it does not assert a host closed-set topology, selected nearest points, choice-based compactness, quotient equality of closed figures, or ambient hyperspace completion.

Definition 6687.1 (AttouchWetsConvergence carrier)**.** *An* $\mathsf{AttouchWetsConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, T, W, K, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row for the ambient metric classifier and its finite distance observations.*
2. *T is the* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *bounded-window row naming the inspected local balls and their tolerance schedule.*
3. *W is the finite window ledger recording the displayed closed-set observations inside each bounded ball.*
4. *K is the* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *handoff row that reads lower-hit and upper-miss evidence from the same displayed windows.*
5. *D is the local distance ledger, comparing the bounded-window observations by* $\mathsf{HausdorffDistance}^{\uparrow}$ *rows and rational tolerance entries.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for window radii, tolerance thresholds, and distance comparisons.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, W, K, D, Q have been displayed.*
8. *H stores componentwise* hsame *transport for source, topology, window, Kuratowski handoff, distance, readback, Real seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a bounded-window convergence request through the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsConvergence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{KuratowskiConvergence}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host hyperspace topology, arbitrary closed-set quotient, selected projection, choice-based compactness principle, or completed hyperspace carrier.

Theorem 6687.2 (AttouchWetsConvergence NameCert obligations)**.** *For every accepted* $\mathsf{AttouchWetsConvergence}^{\uparrow}$ *carrier $A = (X, T, W, K, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission; metric-source exposure through X; bounded-window topology exposure through T; finite window coverage through W; Kuratowski lower-hit and upper-miss handoff through K; local Hausdorff-distance comparison through D; regular rational readback through Q; terminal Real sealing through R; componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Proof. Project the accepted carrier of Definition 6687.1. The rows $X, T, W, K, D, Q, R, H, C, P, N$ are the only displayed coordinates available to the local NameCert.

The source and window obligations are X, T, W. They fix the ambient metric classifier, the bounded local balls, and the closed-set observations inspected inside those balls.

The convergence handoff obligations are K, D, Q, R. The Kuratowski row reads lower-hit and upper-miss evidence, the distance ledger records local Hausdorff comparisons, the regular row records the rational thresholds, and the Real seal is visible only after those finite rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores a host hyperspace topology, arbitrary closed-set quotient, selected projection, choice-based compactness principle, or completed hyperspace carrier, none of those objects belongs to the local NameCert surface. □

Theorem 6687.3 (AttouchWetsConvergence Kuratowski handoff)**.** *Every Kuratowski-facing consumer of an accepted* $\mathsf{AttouchWetsConvergence}^{\uparrow}$ *packet reads the finite route*

$$X, T, W \longmapsto K \longmapsto D, Q, R \longmapsto H, C, P, N.$$

Thus the consumer receives lower-hit and upper-miss evidence only after the $\mathsf{AttouchWetsTopology}^{\uparrow}$ *bounded-window row and the displayed window ledger have been exposed, and it reaches* $\mathsf{Real}^{\uparrow}$ *only through regular rational readback and the local distance ledger.*

Proof. Start from the metric and topology rows. The metric source X supplies the ambient distance classifier, while T names the bounded Attouch–Wets windows in which convergence is inspected.

Read the finite window ledger next. The row W lists the displayed closed-set observations available inside those bounded balls; no Kuratowski consumer can inspect a hit or miss claim before this ledger is present.

Apply the Kuratowski handoff row K. It consumes the displayed windows as lower-hit and upper-miss evidence, and the distance and regular rows D, Q provide the bounded Hausdorff and rational tolerance comparisons before the terminal Real seal R is visible.

The structural rows H, C, P, N preserve the route as local BEDC data. Therefore the handoff supplies a finite convergence packet rather than host closed-set topology, quotient equality, selected nearest points, choice-based compactness, or ambient hyperspace completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6687.1, Definition 6687.2, and Definition 6687.3 seed the finite bounded-window convergence route over metric, Attouch–Wets topology, Kuratowski handoff, local Hausdorff-distance, regular rational, Real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient equality of closed figures, ambient hyperspace completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window coverage, Kuratowski handoff exactness, local distance-ledger coverage, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.AttouchWetsConvergenceUp.

Constructive story: An $\mathsf{AttouchWetsConvergence}^{\uparrow}$ packet is generated from a MetricUp source row, an AttouchWetsTopologyUp bounded-window row, finite closed-set observation windows, a KuratowskiConvergenceUp handoff row, local HausdorffDistanceUp comparisons, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6688

A Concrete Naming Certificate for $\mathsf{MilmanPettis}^{\uparrow}$

Following Milman and Pettis, the $\mathsf{MilmanPettis}^{\uparrow}$ packet records the finite route by which a uniformly convex Banach-space surface may be handed to a reflexive Banach consumer. It sits over the Banach carrier of **??**, the modulus surface of chapter 5831, the weak-observation interface of chapter 3597, the scalar seal of chapter 38, and the reflexive handoff of chapter 4101. The packet names only the displayed uniform-convexity, weak-window, scalar-budget, transport, continuation, provenance, local naming, and reflexive-handoff rows; it does not import Hahn–Banach separation, selected weak cluster points, quotient double-dual equality, or host weak compactness.

Definition 6688.1 (Milman–Pettis carrier)**.** *A* $\mathsf{MilmanPettis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, U, W, R, D, H, C, P, N, E)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *row whose complete normed-space route is read before any convexity conclusion is inspected.*
2. *U is the* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *row, carrying the displayed modulus of convexity over the same Banach surface.*
3. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *observation row, restricted to finite scalar-functional windows.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-budget row used for norm, midpoint, separation, and modulus tolerances.*
5. *D is the* $\mathsf{ReflexiveBanach}^{\uparrow}$ *handoff row, available only after B, U, W, R have been replayed.*
6. *H records componentwise* hsame *transport for the Banach, uniform-convexity, weak-topology, scalar-budget, reflexive-handoff, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Milman–Pettis request through B, U, W, R, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MilmanPettis}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ReflexiveBanach}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MilmanPettis}^{\uparrow}}$ *row.*
10. *E is the ledger-refusal row excluding host weak compactness, selected weak limits, quotient double-dual equality, and choice over bounded nets.*

The classifier compares accepted packets only through $B, U, W, R, D, H, C, P, N, E$.

Theorem 6688.2 (Milman–Pettis NameCert obligations)**.** *For every accepted* $\mathsf{MilmanPettis}^{\uparrow}$ *carrier* $M = (B, U, W, R, D, H, C, P, N, E)$*, the local* $\mathsf{NameCert}_{\mathsf{MilmanPettis}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach row, uniformly convex row, weak-topology row, Real scalar-budget row, reflexive handoff row, transport, continuation, provenance, local naming, and refusal ledger. A consumer may read the Milman–Pettis route only by exposing* $\mathsf{BanachSpace}^{\uparrow}$*,* $\mathsf{UniformlyConvexSpace}^{\uparrow}$*,* $\mathsf{WeakTopology}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ReflexiveBanach}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows in that order.*

Proof. Project the carrier of Definition 6688.1. The analytic rows are B, U, W, R, D: Banach completeness, uniform convexity, weak finite observations, scalar budgets, and the reflexive handoff.

The structural rows H, C, P, N transport, replay, source, and name those analytic rows as one finite packet. The refusal row E has no carrier slot for host weak compactness, selected weak limits, quotient double-dual equality, or choice over bounded nets.

Thus the local NameCert surface is exhausted by the displayed rows. Any accepted consumer must replay the Banach, uniformly convex, weak-topology, scalar-budget, and reflexive-handoff route before reading the local certificate. □

Theorem 6688.3 (Milman–Pettis reflexive handoff)**.** *Every reflexivity-facing consumer of an accepted* $\mathsf{MilmanPettis}^{\uparrow}$ *packet factors through*

$$\mathsf{BanachSpace}^{\uparrow},\ \mathsf{UniformlyConvexSpace}^{\uparrow},\ \mathsf{WeakTopology}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{ReflexiveBanach}^{\uparrow}.$$

In carrier notation, the row D can be read only after the Banach row B, the uniform-convexity row U, the weak-observation row W, and the Real scalar-budget row R are visible. The handoff exports a displayed $\mathsf{ReflexiveBanach}^{\uparrow}$ *route and refuses host weak compactness, selected weak cluster points, quotient double-dual equality, and choice.*

Proof. Begin with the Banach and uniform-convexity rows. The Banach row B supplies the complete normed-space surface, while U supplies the modulus-of-convexity row over that same surface.

Read weak observations and scalar budgets next. The row W restricts weak-topology access to finite scalar-functional windows, and R records the Real-valued norm, midpoint, and tolerance budgets used by the convexity route.

Only after those rows are present does the reflexive handoff row D become visible. Transport through H, replay through C, and provenance through P, N keep the handoff inside the finite packet, while E records that no host compactness theorem, selected weak limit, quotient double-dual equality, or choice principle is exported. □

Independence witness: banachspace, uniformly-convex-space, weak-topology, real, reflexive-banach: $\mathsf{MilmanPettis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Banach completeness, uniform convexity, weak finite observations, Real scalar budgets, a reflexive handoff, transport, replay, provenance, local naming, and refusal rows as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MilmanPettis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MilmanPettis}^{\uparrow}$ packet is generated from BanachSpace rows, UniformlyConvexSpace rows, WeakTopology finite-observation rows, Real scalar-budget rows, a ReflexiveBanach handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert rows, and a refusal ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MilmanPettis}^{\uparrow}}$, seed)

Scope closed: Definition 6688.1, Definition 6688.2, and Definition 6688.3 seed the finite route from uniform convexity over a Banach surface to the reflexive Banach handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MilmanPettis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Hahn–Banach separation, host weak compactness, selected weak cluster points, quotient double-dual equality, choice over bounded nets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-budget exactness, weak-window compatibility, reflexive-handoff nonescape, transport, replay, provenance, local naming, and refusal rows for BEDC.Derived.MilmanPettisUp.

Chapter 6689

A Concrete Naming Certificate for $\mathsf{ModulusOfConvexity}^{\uparrow}$

Following Bishop, Clarkson, and the uniformly convex Banach-space tradition, the $\mathsf{ModulusOfConvexity}^{\uparrow}$ packet records the finite tolerance surface that witnesses how midpoint separation forces a norm drop. It sits over the Banach carrier of **??**, the norm rows of chapter 89, the convex-set rows of chapter 208, the scalar seal of chapter 38, and the Milman–Pettis consumer of chapter 6688. The packet names only displayed norm bounds, midpoint convexity, Real tolerance budgets, transport, replay, provenance, local naming, and uniform-convexity handoff rows; it does not import host compactness, selected minimizers, Hahn–Banach separation, quotient norm equality, or a completed ambient Banach theorem.

Definition 6689.1 (Modulus of convexity carrier)**.** *A* $\mathsf{ModulusOfConvexity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, N, K, R, E, M, H, C, P, L, U)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *row whose complete normed-space surface is read before any modulus value is inspected.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row for the norm, unit-ball boundary, midpoint norm, and comparison windows.*
3. *K is the* $\mathsf{ConvexSet}^{\uparrow}$ *row naming the unit-ball and midpoint convexity surface used by the modulus read.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance row carrying the positive separation budget, midpoint-drop budget, and comparison precision.*
5. *E is the separation row saying that two displayed unit-ball points are at least the tolerance named by R apart.*
6. *M is the modulus row returning a displayed positive drop budget for the midpoint norm.*
7. *H records componentwise* hsame *transport for the Banach, norm, convex-set, Real tolerance, separation, modulus, continuation, provenance, local naming, and handoff rows.*
8. *C records displayed* Cont *replay from a modulus request through $B, N, K, R, E, M, H, P, L, U$.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfConvexity}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *L is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfConvexity}^{\uparrow}}$ *row.*
11. *U is the uniform-convexity handoff row made available to* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *and* $\mathsf{MilmanPettis}^{\uparrow}$ *consumers.*

The classifier compares accepted packets only through $B, N, K, R, E, M, H, C, P, L, U$.

Theorem 6689.2 (Modulus of convexity NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfConvexity}^{\uparrow}$ *carrier $D = (B, N, K, R, E, M, H, C, P, L, U)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfConvexity}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach row, norm row, convex unit-ball row, Real tolerance row, separation row, modulus-drop row, transport, continuation, provenance, local naming, and uniform-convexity handoff. A consumer may read the modulus route only by exposing* $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows in that order.*

Proof. Project the carrier of Definition 6689.1. The analytic rows are B, N, K, R, E, M: Banach completeness, norm access, unit-ball convexity, Real tolerance budgets, displayed separation, and the midpoint-drop modulus.

The structural rows H, C, P, L transport, replay, source, and name those analytic rows as one finite packet. They do not add host compactness, selected minimizers, quotient norm equality, or ambient separation.

The handoff row U is therefore readable only after the displayed modulus rows are present. This exhausts the local NameCert surface and fixes the consumer order through the finite BEDC rows. □

Theorem 6689.3 (Modulus of convexity uniform-convexity handoff)**.** *Every uniformly convex Banach-space consumer of an accepted* $\mathsf{ModulusOfConvexity}^{\uparrow}$ *packet factors through*

$$\mathsf{BanachSpace}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformlyConvexSpace}^{\uparrow}.$$

In carrier notation, the handoff row U can be read only after the Banach row B, norm row N, convex-set row K, tolerance row R, separation row E, and modulus-drop row M are visible. The same handoff is the modulus surface consumed by the Milman–Pettis route.

Proof. Begin with the Banach and norm rows. The row B supplies the complete normed-space surface, while N supplies the unit-ball, distance, midpoint, and comparison windows used by the modulus read.

Read the convexity and scalar rows next. The row K restricts the midpoint operation to the displayed convex unit-ball surface, and R supplies the Real-valued separation and midpoint-drop tolerances.

Only after E and M expose the separation-to-drop relation does U become available. Transport through H, replay through C, and provenance through P, L keep the handoff inside the same finite packet, so $\mathsf{UniformlyConvexSpace}^{\uparrow}$ and $\mathsf{MilmanPettis}^{\uparrow}$ receive a displayed modulus route rather than an ambient Banach-space theorem. □

Independence witness: banachspace, norm, convexset, real, milman-pettis: $\mathsf{ModulusOfConvexity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Banach completeness, norm windows, convex unit-ball rows, Real tolerance budgets, separation, midpoint-drop data, transport, replay, provenance, local naming, and a uniform-convexity handoff as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfConvexity}^{\uparrow}}$**.**

Constructive story: A ModulusOfConvexity$^{\uparrow}$ packet is generated from BanachSpace rows, Norm rows, ConvexSet unit-ball rows, Real tolerance rows, separation rows, midpoint-drop rows, displayed hsame transport, displayed Cont replay, Pkg provenance, local NameCert rows, and a uniform-convexity handoff.

Theory closure: TheoryGate(NameCert$_{\text{ModulusOfConvexity}^{\uparrow}}$, seed)

Scope closed: Definition 6689.1, Definition 6689.2, and Definition 6689.3 seed the finite route from a Banach norm and convex unit-ball surface to a uniform-convexity modulus handoff.

Formal verification: FormalStatus(NameCert$_{\text{ModulusOfConvexity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactness, selected minimizers, Hahn–Banach separation, quotient norm equality, ambient Banach-space uniform convexity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, midpoint-drop nonescape, uniform-convexity handoff, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfConvexityUp.

Chapter 6690

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$

The $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ packet realizes the diagonal-limit branch of section 9238.1 as one finite exchange certificate. It sits beside the StreamName windows of chapter 126, the DyadicRatCore tolerance ledger of chapter 291, the RegSeqRat readback route of chapter 301, and the Real seal of chapter 38. The packet names paired diagonal-window choices, tail-meet compatibility, real-seal congruence, transport, continuation, provenance, and local naming without introducing quotient streams, selected limits, countable choice, or ambient Real completeness.

Definition 6690.1 (Regular Cauchy diagonal-exchange carrier)**.** *A* $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, D, R, Q, T, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row containing the two inspected diagonal windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row recording the paired tolerance budgets and the comparison scale.*
3. *R and Q are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the diagonal windows.*
4. *T is the regular-cauchy tail-meet row that synchronizes the two windows before the exchange is accepted.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, D, R, Q, T have been replayed.*
6. *H records componentwise* hsame *transport for the window, tolerance, readback, tail-meet, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an exchange request through $S, D, R, Q, T, L, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, Q, T, L, H, C, P, N$.

Lean4 (definition) `BEDC.Derived.RegularCauchyDiagonalExchangeUp`

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalExchangeUp.RegularCauchyDiagonalExchangeTasteGate`
`single carrier alignment`

Theorem 6690.2 (Regular Cauchy diagonal-exchange NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ *carrier* $E = (S, D, R, Q, T, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExc}}$ *surface consists exactly of carrier admission for StreamName windows, DyadicRatCore tolerances, paired RegSeqRat readbacks, the regular-cauchy tail-meet row, the terminal Real seal, componentwise transport, continuation replay, provenance, and local naming. No accepted exchange may expose the Real seal before the rows*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

are visible in the packet.

Proof. Project the carrier of Definition 6690.1. The diagonal-window row S, tolerance row D, readback rows R, Q, tail-meet row T, and Real seal row L are the only analytic coordinates.

The structural rows H, C, P, N transport, replay, source, and name the same finite exchange. Since the classifier has no additional coordinate, quotient streams, selected limits, countable-choice schedules, and ambient completeness cannot be read from the packet.

Therefore every accepted Real-seal read is downstream of the displayed StreamName, DyadicRatCore, RegSeqRat, Real, transport, continuation, provenance, and naming rows. □

Theorem 6690.3 (Regular Cauchy diagonal-window exchange)**.** *Every accepted* $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ *packet turns a paired diagonal-window read into a Real-facing seal only through the finite route*

$$S, D, R, Q \longmapsto T \longmapsto L \longmapsto H, C, P, N.$$

The route records diagonal-limit compatibility and regular-cauchy tail-meet synchronization before the Real seal is exposed, and it exports no quotient-stream equality or selected limit.

Proof. Begin with the two windows carried by S. The tolerance row D fixes the dyadic scale at which the windows are compared, while R and Q read the corresponding regular rational data.

The tail-meet row T is then consumed before L is visible. It is the packet-local synchronization witness for diagonal-limit compatibility, so the exchange is not a detached choice of a subsequence or a selected limit.

Finally H, C, P, N keep the accepted route inside the same finite packet. The Real seal is therefore a terminal row of the exchange certificate rather than a proof of quotient-stream equality or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ exchange read exposes $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows before a $\mathsf{Real}^{\uparrow}$ seal can be read.

Independence witness: regularcauchysequence, diagonal-limit-compatibility, regular-cauchy-tail-meet, real-seal-congruence: $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired diagonal windows, dyadic tolerances, paired regular readbacks, tail-meet synchronization, Real-seal congruence, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}$ packet is generated from paired StreamName diagonal windows, DyadicRatCore tolerance budgets, paired RegSeqRat readbacks, a regular-cauchy tail-meet row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6690.1, Definition 6690.2, and Definition 6690.3 seed the finite exchange route from paired diagonal windows to a Real seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDiagonalExchange}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyDiagonalExchangeUp.RegularCauchyDiagonalExchangeTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected limits, countable-choice schedules, ambient Real completeness, or bridge checking.

Upgrade path: Axiom-clean closure requires the same diagonal-exchange TasteGate target to pass the transitive axiom-purity audit.

Chapter 6691

A Concrete Naming Certificate for $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$ packet records the finite error-envelope rows used when dyadic estimates are opened, regular Cauchy tails are compared, and a $\mathsf{Real}^{\uparrow}$ seal is read without appealing to quotient stream equality, selected limits, or ambient completeness. It sits over the dyadic arithmetic core of chapter 291, the finite observation discipline of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the displayed tail-estimate surface of chapter 1825, and the split-budget packet of chapter 1761. The packet gives Real-and-Completion consumers a named finite envelope carrier between local dyadic bounds and regular Cauchy tail reads.

Definition 6691.1 (Dyadic regular error-envelope carrier)**.** *A* $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, B, W_0, W_1, D_0, D_1, D_\Delta, R_0, R_1, T, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the requested regular tail tolerance presented by the downstream Real-or-completion consumer.*
2. *B is the* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *row that splits the visible dyadic slack for the two regular reads.*
3. *W_0 and W_1 are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened before the two approximants are inspected.*
4. *D_0 and D_1 are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-error rows read from those windows.*
5. *D_Δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row recording the displayed dyadic envelope between the two endpoint-error rows and Q.*
6. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the two finite windows.*
7. *T is the* $\mathsf{CauchyEstimate}^{\uparrow}$ *tail-estimate row saying that the same envelope controls the regular Cauchy tail request.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic envelope, regular readback, and tail-estimate rows have been displayed.*
9. *H records componentwise* hsame *transport for the request, budget, windows, dyadic rows, regular readbacks, tail estimate, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the request through the budget, windows, dyadic envelope, regular readbacks, tail estimate, and Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$, $\mathsf{DyadicErrorBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, trichotomy, detached completion evidence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DyadicRegularErrorEnvelopeUp`

Theorem 6691.2 (Dyadic regular error-envelope stability)**.** *For every accepted* $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$ *carrier* $E = (Q, B, W_0, W_1, D_0, D_1, D_\Delta, R_0, R_1, T, A, H, C, P, N)$*, any refinement of the two finite windows that preserves the displayed budget row* B*, dyadic endpoint rows* D_0, D_1*, envelope row* D_Δ*, regular readbacks* R_0, R_1*, and tail-estimate row* T *preserves the same error-envelope route under* hsame*,* Cont*,* Pkg*, and local* NameCert *naming.*

Proof. Project the carrier of Definition 6691.1. The only analytic coordinates are the request, budget, two windows, three dyadic rows, two regular readbacks, tail estimate, and Real seal.

Window refinement can therefore be read only through W_0, W_1 and their attached rows R_0, R_1. The dyadic comparison still passes through D_0, D_1, D_Δ against the same request Q and budget row B.

The row T records that this displayed dyadic envelope is the tail-estimate surface consumed by the regular Cauchy route. The structural rows H, C, P, N transport, replay, source, and name that same route, so no hidden selector, quotient equality, or completion principle can change the envelope while the packet remains accepted. □

Theorem 6691.3 (Dyadic regular error-envelope Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of an accepted* $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$ *packet must factor through*

$$Q, B \longmapsto W_0, W_1 \longmapsto D_0, D_1, D_\Delta \longmapsto R_0, R_1, T \longmapsto A, H, C, P, N.$$

Thus the Real seal is downstream of the displayed dyadic budget, finite windows, regular readbacks, and Cauchy-estimate envelope; it is not a quotient-real equality, selected limit, countable-choice, or ambient-completeness export.

Proof. Read the request and budget rows first. The carrier gives the downstream tolerance Q exactly one displayed dyadic budget row B, so the slack available to the Real-facing read is finite before any stream data is inspected.

Next open the two finite windows and their dyadic rows. The rows $W_0, W_1, D_0, D_1, D_\Delta$ expose the endpoint errors and the comparison envelope, while R_0, R_1 attach the regular-rational readbacks required by the Cauchy tail route.

Finally T feeds the displayed envelope to the Cauchy-estimate surface and A seals the result as a Real-facing row. Transport, replay, provenance, and local naming preserve the same finite route, and the carrier contains no coordinate from which a stronger Real or completion principle could be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicRegularErrorEnvelope}^{\uparrow}$ packet is generated from a regular tail request, one DyadicErrorBudget split, two StreamName windows, three DyadicRatCore envelope rows, two RegSeqRat readbacks, one CauchyEstimate tail row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicRegularErrorEnvelope}\uparrow}$, seed)

Scope closed: Definition 6691.1, Definition 6691.2, and Definition 6691.3 seed the finite dyadic envelope route between displayed error budgets, regular Cauchy readbacks, Cauchy-estimate tails, and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{DyadicRegularErrorEnvelope}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.DyadicRegularErrorEnvelopeUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, envelope stability, regular readback, Cauchy-estimate handoff, Real-seal compatibility, transport, replay, provenance, and local naming for BEDC.Derived.DyadicRegularErrorEnvelopeUp.

Chapter 6692

A Concrete Naming Certificate for MetacicClosedBetaTraceEnvelope$^{\uparrow}$

The MetacicClosedBetaTraceEnvelope$^{\uparrow}$ packet realizes the ClosurePreservation audit map of chapter 9381 as finite beta-chain, substitution, and ClosedAt replay data. It sits beside the closed beta path of chapter 797, the MetaCIC closure trace of chapter 795, the closed-term substitution compiler of chapter 540, and the kernel acceptance trace of chapter 557. The carrier is not a subject-reduction theorem: it records closedness preservation along an accepted beta-star route before any typed transport claim is allowed.

Definition 6692.1 (MetaCIC closed beta trace-envelope carrier)**.** *A* MetacicClosedBetaTraceEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, T, B, S_b, K, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source closed term row, including the displayed ClosedAt boundary.*
2. *T is the target term row reached by the accepted beta-star route.*
3. *B is the finite beta-chain row, listing the one-step beta moves and congruence positions used by the route.*
4. *S_b is the substitution replay row used by beta-redex contractions.*
5. *K is the ClosedAt preservation row for the source, every intermediate beta-chain node, and the target.*
6. *R records the replay order of shift, variable-substitution, full-substitution, beta-step, and beta-star closedness witnesses.*
7. *A is the kernel-acceptance trace row saying that the displayed replay is the accepted audit route.*
8. *H records componentwise* hsame *transport for source, target, beta-chain, substitution, closedness, replay, acceptance, route, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same closed beta trace route.*
10. *P is the* Pkg *provenance over* MetacicClosedBetaTraceEnvelope$^{\uparrow}$*, MetaCIC closure trace, closed-term substitution compiler, TasteGate audit trace, kernel acceptance trace,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for subject reduction, typed conversion, dependent codomain transport, hidden beta normalization, quotienting of traces, or acceptance outside the displayed kernel route.

Lean4 (statement) `BEDC.Derived.MetacicClosedBetaTraceEnvelopeUp`

Theorem 6692.2 (MetaCIC closed beta trace-envelope preservation)**.** *For every accepted* MetacicClosedBetaTraceE *carrier $E = (S, T, B, S_b, K, R, A, H, C, P, N)$, the target row T is read as closed only by replaying the displayed beta-star route B, the substitution replay row S_b, the ClosedAt chain K, and the witness order R. The result cannot be obtained from an unrecorded subject-reduction step, typed conversion rule, hidden normalization oracle, or quotient trace identity.*

Proof. Project the carrier. The source and target rows are connected by the finite beta-chain row B, and beta-redex contractions can read substitution data only through S_b.

The closedness claim is then the row K replayed in the order R. It follows the shift, variable-substitution, full-substitution, beta-step, and beta-star preservation witnesses named in the audit map, without inspecting typing judgments.

Finally A, H, C, P, N accept, transport, replay, source, and name the same finite route. Since the carrier omits typed conversion, dependent codomain transport, hidden normalization, and quotient trace identity, the accepted envelope preserves only the displayed closedness route. □

Theorem 6692.3 (MetaCIC closed beta trace-envelope vision handoff)**.** *The* $\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}$ *packet hands chapter 9381 to concrete NameCert consumers by packaging beta-star route data, substitution replay, ClosedAt preservation rows, kernel acceptance, transport, displayed continuation replay, provenance, and local naming as one finite carrier. A later Lean-facing gate may require 'TasteGate.lean' to provide Nontrivial* $\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}$ *and FieldFaithful* $\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}$*, but the paper carrier already fixes the finite rows those gates must inspect.*

Proof. Use Definition 6692.1. Its rows name the source, target, beta-chain, substitution replay, ClosedAt replay, acceptance trace, transport, route, provenance, and local certificate explicitly.

The handoff is finite because every audit row is a displayed BHist coordinate. Downstream TasteGate obligations can test nontriviality and field faithfulness against those rows, while any typed subject-reduction claim must land in a separate packet with its own dependent transport data. □

Falsifiable prediction: Every accepted $\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}$ packet replays a beta-star closedness preservation route through displayed beta-chain, substitution, ClosedAt, and kernel-acceptance rows; an accepted packet whose target closedness cannot be replayed from those rows refutes this chapter.

Independence witness: metacicclosuretrace, closedtermsubstitutioncompiler, tastegateaudittrace, kernelacceptancetrace: the closed beta trace envelope carries a source-target beta-chain row, a substitution replay row, a ClosedAt chain row, and a kernel-acceptance row simultaneously, while each named sibling omits at least one of those rows, so no row-preserving bijection can identify the carriers.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{MetacicClosedBetaTraceEnvelope}^{\uparrow}$ packet is generated from source and target closed-term rows, a beta-chain row, substitution replay, ClosedAt preservation replay, kernel acceptance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetacicClosedBetaTraceEnvelope}\uparrow}$, seed)

Scope closed: Definition 6692.1, Definition 6692.2, and Definition 6692.3 seed the finite closed beta trace envelope over beta-star, substitution, ClosedAt replay, and kernel acceptance rows.

Formal verification: FormalStatus(NameCert$_{\text{MetacicClosedBetaTraceEnvelope}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.MetacicClosedBetaTraceEnvelopeUp`

Bridge status: none

Not claimed: This chapter does not close subject reduction, typed conversion, dependent codomain transport, hidden normalization, quotient trace identity, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked beta-chain replay, substitution replay, ClosedAt preservation, kernel acceptance, transport, route replay, provenance, local naming, and TasteGate nontriviality and field-faithfulness obligations for BEDC.Derived.MetacicClosedBetaTraceEnvelopeUp.

Chapter 6693

A Concrete Naming Certificate for $\mathsf{SkorokhodTopology}^{\uparrow}$

Following Skorokhod and Billingsley on the topology of cadlag path spaces, the $\mathsf{SkorokhodTopology}^{\uparrow}$ packet records the finite naming surface used to compare right-continuous paths with left limits by displayed time-change, window, and real-name data. It sits after the cadlag path carrier of chapter 5968, the metric interface of chapter 57, the finite observation schedule of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The certificate gives stochastic-process consumers a path-space topology route without importing selected path limits, host path equality, quotient-by-time-change equality, Borel measure construction, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6693.1 (Skorokhod topology carrier)**.** *A* $\mathsf{SkorokhodTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, M, S, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CadlagRealFunction}^{\uparrow}$ *source row naming the path packet whose right-continuity and left-limit windows are already displayed.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used only as a finite tolerance interface on path-window observations.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for domain windows, jump-neighborhood windows, and time-change probes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations along the displayed schedule.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling mesh size, time-change displacement, path-window error, and real-name readback precision.*
6. *L is the Skorokhod-lag row. It records finite monotone time-change probes and their mesh/tolerance obligations, not a selected homeomorphism of the host interval.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the cadlag source, metric comparison row, observation schedule, rational readback, dyadic tolerance ledger, and lag row have been displayed.*
8. *H records componentwise* hsame *transport for the cadlag source, metric comparison, schedule, readback, tolerance, lag, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a path-topology request through $K, M, S, R, D, L, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SkorokhodTopology}^{\uparrow}$, $\mathsf{CadlagRealFunction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SkorokhodTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, S, R, D, L, E, H, C, P, N$. It has no coordinate for host path equality, quotient path equality, selected time changes, selected limits, Borel sigma-algebras, compactness, tightness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6693.2 (Skorokhod topology NameCert obligations)**.** *For every accepted* $\mathsf{SkorokhodTopology}^{\uparrow}$ *carrier $T = (K, M, S, R, D, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SkorokhodTopology}^{\uparrow}}$ *surface consists exactly of carrier admission, cadlag-source admission through K, metric comparison admission through M, finite observation schedule exposure through S, regular rational readback through R, dyadic tolerance control through D, finite Skorokhod-lag obligation through L, terminal real-seal ordering through E, componentwise transport through H, displayed replay through C, package provenance through P, local naming through N, and ledger refusal for host path equality, quotient path equality, selected time changes, selected limits, measure-theoretic structure, compactness, tightness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The analytic rows are K, M, S, R, D, L, E, and the structural rows are H, C, P, N.

The cadlag source K supplies right-continuity and left-limit window obligations from chapter 5968. The metric comparison row M, stream schedule S, rational readback R, and dyadic ledger D provide the finite data on which a topology consumer may compare path-window observations.

The lag row L is downstream of those data. It records finite time-change probes and their tolerance obligations, while the terminal seal E is reached only after the displayed route has passed through K, M, S, R, D, L.

Componentwise transport, replay, provenance, and local naming preserve the same packet. Since no coordinate stores selected time changes, quotient path equality, measure-theoretic structure, compactness, tightness, or ambient real completeness, the NameCert obligations are exactly the listed finite rows. □

Theorem 6693.3 (Skorokhod topology cadlag modulus handoff)**.** *Every path-topology consumer of an accepted* $\mathsf{SkorokhodTopology}^{\uparrow}$ *packet must first read the* $\mathsf{CadlagRealFunction}^{\uparrow}$ *source row, then the* $\mathsf{Met}^{\uparrow}$ *comparison row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before it can consume the Skorokhod-lag row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the handoff is a finite cadlag-modulus route rather than a host equality of paths, a quotient equality under time changes, a selected limiting path, a measure-theoretic compactness theorem, or an ambient completeness principle.*

Proof. Let $T = (K, M, S, R, D, L, E, H, C, P, N)$ be accepted. A path-topology request reaches the cadlag source row K before any comparison row is available, so the request begins with displayed right-continuity and left-limit windows.

Next the request reads M, S, R, D. These rows give the metric tolerance interface, finite observation schedule, regular rational readback, and dyadic precision ledger needed to state a finite comparison between path windows.

Only then may the request consume the lag row L and the terminal seal E. The lag row records time-change probes as displayed finite obligations, while the seal row remains downstream of cadlag, metric, schedule, readback, and tolerance data.

Transport through H and replay through C preserve the same route, and P, N package and name it. The excluded host equality, quotient equality, selected-limit, measure-theoretic, compactness, and completeness conclusions have no carrier row to read. □

Closure status for $\mathsf{NameCert}_{\mathsf{SkorokhodTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{SkorokhodTopology}^{\uparrow}$ packet is generated from a CadlagRealFunction source row, Metric comparison row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, finite Skorokhod-lag row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SkorokhodTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6693.1, Definition 6693.2, and Definition 6693.3 seed the finite cadlag path-topology route over displayed time-change probes and terminal Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SkorokhodTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host path equality, quotient path equality, selected time changes, selected limits, Borel measure construction, compactness, tightness, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, cadlag-source admission, metric comparison admission, stream-window exposure, rational readback, dyadic tolerance control, Skorokhod-lag obligations, transport, replay, provenance, and local naming for BEDC.Derived.SkorokhodTopologyUp.

Chapter 6694

A Concrete Naming Certificate for $\mathsf{JordanHolder}^{\uparrow}$

Following Jordan and Holder, the $\mathsf{JordanHolder}^{\uparrow}$ packet reconstructs finite-group composition-series data as a BEDC NameCert surface. It sits beside the group carrier of chapter 41, the subgroup carrier of chapter 80, the quotient-group carrier of chapter 82, and the Sylow packet of chapter 3728. Those sibling chapters supply group operations, subgroup containment, quotient-factor rows, and prime-power subgroup witnesses; this chapter records how a composition-series certificate names the factor rows without importing a classification theorem, a selected series, or an unrecorded isomorphism oracle.

Definition 6694.1 (Jordan-Holder carrier)**.** *A* $\mathsf{JordanHolder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (G, S, Q, F, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Group}^{\uparrow}$ *source row, carrying the multiplication, unit, inverse, and finite carrier window.*
2. *S is the composition-series row, listing the finite chain of* $\mathsf{Subgroup}^{\uparrow}$ *entries from the trivial subgroup to G.*
3. *Q is the quotient-factor row, recording each adjacent* $\mathsf{QuotientGroup}^{\uparrow}$ *factor exposed by the chain.*
4. *F is the simple-factor classifier row, recording that each displayed quotient factor has no non-trivial normal-subgroup row inside the factor surface.*
5. *E is the factor-equivalence ledger, comparing two displayed composition-series rows only through finite permutations and componentwise quotient-factor equivalences.*
6. *H records componentwise* hsame *transport for the group source, series, quotient factors, simple-factor classifiers, equivalence ledger, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from the group source through the series and factor rows.*
8. *P is the* Pkg *provenance over* $\mathsf{JordanHolder}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{Sylow}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanHolder}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen canonical composition series, a complete finite-simple-group classifier, an unrecorded quotient isomorphism, or a proof that two arbitrary chains are equal outside the displayed factor ledger.

Theorem 6694.2 (Jordan-Holder NameCert obligation surface)**.** *For an accepted* $\mathsf{JordanHolder}^{\uparrow}$ *packet* $J = (G, S, Q, F, E, H, C, P, N)$*, the local NameCert obligation surface consists exactly of the finite group source row* G*, composition-series row* S*, quotient-factor row* Q*, simple-factor classifier row* F*, factor-equivalence ledger* E*, componentwise transport row* H*, replay row* C*, package row* P*, and local naming row* N*. Any consumer that drops one of these rows leaves the composition-series certificate and receives neither a quotient-factor certificate nor a Jordan-Holder comparison route.*

Proof. Project the carrier of Definition 6694.1. The group source, subgroup chain, quotient factors, simple-factor classifiers, and factor-equivalence ledger are displayed as distinct finite rows.

The sibling surfaces determine what each row may export. The group row is read through chapter 41, the series entries through chapter 80, and the adjacent factors through chapter 82.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. Since no canonical-series selector or hidden quotient-isomorphism oracle is present, deleting any displayed row exits the certificate surface. □

Theorem 6694.3 (Jordan-Holder composition-series route)**.** *Every composition-series consumer of an accepted* $\mathsf{JordanHolder}^{\uparrow}$ *packet must read the finite route*

$$G \to S \to Q \to F \to E \to H, C, P, N.$$

The route may cite the Sylow packet only as prime-power subgroup provenance for displayed finite-group rows, and it may cite quotient factors only through the adjacent $\mathsf{QuotientGroup}^{\uparrow}$ *rows in* Q*. It does not export a classification of finite simple groups, a chosen canonical series, or an equality of arbitrary composition-series lists outside the factor-equivalence ledger* E*.*

Proof. Start with the source group row G and the displayed chain S. Each adjacent pair in the chain supplies a quotient-factor request, so the consumer reaches the factor row Q before any comparison is available.

Read the simple-factor classifier row F and the factor-equivalence ledger E. The classifier confines each factor to the displayed quotient surface, while E records comparison only by finite reindexing and componentwise factor equivalence.

Finally replay through H, C, P, N. Provenance may cite chapter 3728 for finite prime-power subgroup rows, but the replay has no coordinate for a canonical composition series, a finite-simple-group classification theorem, or an external quotient-isomorphism oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{JordanHolder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{JordanHolder}^{\uparrow}$ packet is generated from a finite group source, a composition-series chain, adjacent quotient factors, simple-factor classifier rows, factor-equivalence ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanHolder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6694.1, Definition 6694.2, and Definition 6694.3 seed the finite composition-series certificate surface over $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{Sylow}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanHolder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, classification of finite simple groups, canonical composition-series selection, or equality of arbitrary series outside the displayed factor ledger.

Upgrade path: Obligation closure requires checked carrier admission, quotient-factor exactness, simple-factor classifier stability, factor-equivalence transport, replay, provenance, and local naming for BEDC.Derived.JordanHolderUp.

Chapter 6695

Bishop Cauchy Real Envelope NameCert Construction

Following Bishop and Bridges, the $\mathsf{BishopCauchyRealEnvelope}^{\uparrow}$ packet records the finite envelope boundary at which regular Cauchy data may face $\mathsf{Real}^{\uparrow}$ without becoming a quotient-real construction or a selected limiting value. It sits after the finite observation discipline of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, the effective rate realization route of chapter 1673, the rational approximation window of chapter 1668, and the four-face completion interface of chapter 1844. The envelope gives Real-and-Completion consumers a single finite boundary for stream windows, dyadic budgets, regular-Cauchy witnesses, rational approximants, Real seals, and ledger projections.

Definition 6695.1 (Bishop Cauchy real envelope carrier)**.** *A* $\mathsf{BishopCauchyRealEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, D, R, A, B, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row exposing exactly the finite observation windows opened by the Real-facing request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row controlling endpoint scale, tail radius, comparison precision, and rational readback budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy witness row read only through the displayed stream windows and dyadic budget.*
4. *A is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *rational approximation row attached to the same S, D, R data.*
5. *B is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the stream, dyadic, regular-Cauchy, and rational approximation rows have been displayed.*
6. *L is the ledger projection row. It records the finite completion-interface read through* $\mathsf{CauchyRateRealization}^{\uparrow}$ *and* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$*, and it stores no chosen limit or quotient representative.*
7. *H records componentwise* hsame *transport for the stream window, dyadic budget, regular-Cauchy witness, rational approximation, Real seal, ledger projection, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an envelope request through $S, D, R, A, B, L, H, P, N$.*
9. *P is the* Pkg *provenance over* BishopCauchyRealEnvelope$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealRationalApproximation$^{\uparrow}$, CauchyRateRealization$^{\uparrow}$, CauchyRealCompletionInterface$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopCauchyRealEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, A, B, L, H, C, P, N$. It has no coordinate for quotient reals, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ *completeness, or bridge credentials.*

Theorem 6695.2 (Bishop Cauchy real envelope obligation surface)**.** *For every accepted* BishopCauchyRealEnvelope$^{\uparrow}$ *carrier $E = (S, D, R, A, B, L, H, C, P, N)$, the local* NameCert$_{\text{BishopCauchyRealEnvelope}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier display, requiring the stream-window, dyadic-budget, regular-Cauchy witness, rational approximation, Real seal, ledger projection, transport, replay, provenance, and local naming rows to be present;*
2. *classifier stability, requiring accepted packets to be compared only through the displayed rows $S, D, R, A, B, L, H, C, P, N$;*
3. *envelope transport, requiring every* hsame *transport and every* Cont*-compatible finite observation replay to preserve the same stream windows, dyadic budget, regular witness, rational approximation, Real seal, and ledger projection;*
4. *ledger exactness, requiring the projection row L to factor through the Cauchy-rate realization and four-face completion-interface rows before the Real seal can be consumed;*
5. *no quotient or choice use, requiring the certificate to export no quotient real, selected limit, countable-choice selector, trichotomy datum, host real equality proof, ambient completeness theorem, or bridge credential.*

No local obligation is stored outside these five families.

Proof. Project the accepted carrier. The analytic rows are exactly S, D, R, A, B, L, and the structural rows are exactly H, C, P, N.

Carrier display and classifier stability are immediate from the displayed packet: every accepted comparison has access only to those ten rows. The regular-Cauchy witness R is downstream of the stream window and dyadic budget, the rational approximation A is attached to the same finite data, and the Real seal B appears only after those rows have been read.

Transport and replay use the structural rows. The hsame row H moves each displayed coordinate componentwise, while the Cont row C replays the same finite observation route for later consumers.

The ledger row L is the only completion-facing projection. It records the finite Cauchy-rate and completion-interface handoff, so ledger exactness cannot add a second Real coordinate or a selected limit.

The excluded principles have no carrier row and no continuation route. Hence quotient reals, selected limits, countable choice, trichotomy, host real equality, ambient completeness, and bridge credentials cannot become local NameCert obligations. □

Theorem 6695.3 (Bishop Cauchy real envelope scope)**.** *Every accepted* BishopCauchyRealEnvelope$^{\uparrow}$ *consumer depends only on the sibling route through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$,

$\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRateRealization}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, *and* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$. *Its public read is exhausted by the rows* $S, D, R, A, B, L, H, C, P, N$, *so the envelope does not depend on quotient reals, selected limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Read the provenance row P of the accepted carrier. It names exactly the sibling surfaces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{CauchyRateRealization}^{\uparrow}$, $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the analytic dependency route.

The carrier rows realize that route in order. A consumer first receives the finite stream window and dyadic budget, then the regular-Cauchy witness, rational approximation, Real seal, and ledger projection. The structural rows transport, replay, source, and name the same packet.

Apply the obligation surface of Definition 6695.2. Since no accepted row stores any excluded principle, the public scope is exhausted by the listed sibling route and carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRealEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyRealEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6695.1, Definition 6695.2, and Definition 6695.3 seed the finite envelope boundary for Real-facing regular Cauchy data over stream windows, dyadic budgets, regular witnesses, rational approximation, Real seals, and ledger projection rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyRealEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BishopCauchyRealEnvelopeUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, selected limits, countable choice, trichotomy, ambient Real completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier display, classifier stability, hsame and Cont-compatible envelope transport, ledger exactness, and no-quotient/no-choice nonescape for BEDC.Derived.BishopCauchyRealEnvelopeUp.

Constructive story: A $\mathsf{BishopCauchyRealEnvelope}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ stream-window row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-budget row, a $\mathsf{RegSeqRat}^{\uparrow}$ regular-Cauchy witness row, a $\mathsf{RealRationalApproximation}^{\uparrow}$ rational approximation row, a terminal $\mathsf{Real}^{\uparrow}$ seal, a completion-facing ledger projection, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6696

A Concrete Naming Certificate for $\mathsf{FiniteUnionBound}^{\uparrow}$

Following Boole's inequality and elementary finite probability, the $\mathsf{FiniteUnionBound}^{\uparrow}$ packet records finite union-bound data over displayed $\mathsf{ProbSpace}^{\uparrow}$ event windows, not a sigma-completion or almost-sure quotient. It sits after the probability-space source of chapter 184, the random-variable observation surface of chapter 185, and the distribution readback surface of chapter 186; the finite ledger is then available to the Borel–Cantelli route of chapter 2357.

Definition 6696.1 (FiniteUnionBound carrier)**.** *A* $\mathsf{FiniteUnionBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (\Omega, E, W, S, B, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the probability carrier from which every event probability is read.*
2. E *is the finite event-family row, listing the displayed event names and their membership readbacks.*
3. W *is the finite union-window row, recording the inspected finite subfamily and the displayed union event produced from it.*
4. S *is the probability-sum ledger, recording the finite sum of the probabilities of the events listed by* W*.*
5. B *is the bound-comparison row, recording that the probability of the displayed union event is bounded by the ledger value in* S*.*
6. H *records componentwise* hsame *transport for the probability source, event family, union window, sum ledger, bound comparison, replay, provenance, and local naming rows.*
7. C *records the displayed* Cont *replay by which downstream consumers read* Ω, E, W, S, B *in that order.*
8. P *is the* Pkg *provenance over* $\mathsf{FiniteUnionBound}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteUnionBound}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a completed sigma-algebra, countable union, quotient almost-sure equality, independence converse, limiting theorem, or host probability space outside the displayed finite event rows.

Theorem 6696.2 (FiniteUnionBound NameCert obligations)**.** *For every accepted* $\mathsf{FiniteUnionBound}^{\uparrow}$ *carrier* $U = (\Omega, E, W, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteUnionBound}^{\uparrow}}$ *surface consists exactly of five finite obligation rows: carrier exposure for the displayed tuple; event-window exactness linking* E *and* W *to the union event read in* Ω*; probability-sum ledger exposure through* S*; bound-comparison exposure through* B*; and transport, replay, provenance, and local naming through* H, C, P, N*. No finite union-bound obligation is stored outside these rows.*

Proof. Project the accepted carrier. The tuple $U = (\Omega, E, W, S, B, H, C, P, N)$ exposes the probability source, event family, union window, sum ledger, comparison row, transport, replay, provenance, and local naming rows.

Read the event-window row before the probability ledger. The window W selects only entries from E and names the displayed finite union event in the source Ω, so the bound has a fixed finite domain.

The sum and comparison rows then give the certificate surface. The row S lists the finite probability sum, and B compares the probability of the displayed union event with that ledger value. Transport through H and replay through C preserve exactly these rows, while P, N record provenance and naming.

Since no remaining coordinate names a sigma-completion, countable union, quotient almost-sure equality, independence converse, or limiting theorem, the five listed obligations exhaust the local certificate. □

Theorem 6696.3 (FiniteUnionBound window-sum ledger)**.** *Every finite union read of an accepted* $\mathsf{FiniteUnionBound}^{\uparrow}$ *packet factors through the displayed route*

$$\Omega,\ E,\ W,\ S,\ B.$$

The event family and union window determine the inspected finite subfamily before the probability-sum ledger is available, and the bound comparison reads only that ledger. Thus a consumer cannot replace the finite event-window sum by an ambient countable union, external subadditivity oracle, or quotient event equality while preserving the packet.

Proof. Start with the finite event-family row E and the window row W. They determine the displayed union event and the finite list of event probabilities to be inspected in Ω.

The probability-sum ledger S is then the only row that records the bound value. The comparison row B reads the union-event probability and compares it with that value, so every accepted finite union read has passed through the same window-sum ledger.

Finally H, C, P, N transport, replay, source, and name the route without adding another probability object. Coordinate exhaustion excludes countable unions, external subadditivity, and quotient event equality from the accepted read. □

Theorem 6696.4 (FiniteUnionBound Borel–Cantelli handoff)**.** *Every* $\mathsf{BorelCantelli}^{\uparrow}$*-facing consumer that requests a finite union estimate receives the* $\mathsf{FiniteUnionBound}^{\uparrow}$ *ledger before any tail-sum reasoning is exposed. The handoff supplies the probability source row* Ω*, finite event family* E*, union window* W*, probability-sum ledger* S*, bound comparison* B*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row; it supplies no sigma-completion, countable-choice selector, almost-sure quotient equality, independence converse, or infinite limiting theorem.*

Proof. Project the accepted $\mathsf{FiniteUnionBound}^{\uparrow}$ packet and read the route of Definition 6696.3. A Borel–Cantelli finite-window request first receives Ω, E, W, S, B as displayed rows.

Only after this finite ledger is present may the downstream Borel–Cantelli packet ask for a tail-sum route. The handoff therefore separates the Boole-inequality ledger from the later tail-event accounting.

Transport, replay, provenance, and naming are inherited through H, C, P, N. Since the handoff has no coordinate for sigma-completion, countable choice, almost-sure quotient equality, independence converse, or an infinite limiting theorem, none of those claims is exported to the Borel–Cantelli consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteUnionBound}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteUnionBound}^{\uparrow}$ packet is generated from a probability source row, finite event-family row, finite union-window row, probability-sum ledger, bound-comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteUnionBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6696.1, Definition 6696.2, Definition 6696.3, and Definition 6696.4 seed the finite Boole-inequality ledger and its Borel–Cantelli-facing handoff over displayed probability windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteUnionBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-completion, countable choice, quotient almost-sure equality, an independence converse, an infinite limiting theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, event-window exactness, probability-sum ledger stability, bound-comparison exactness, Borel–Cantelli handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteUnionBoundUp.

Chapter 6697

A Concrete Naming Certificate for $\mathsf{BanachLimit}^{\uparrow}$

Following Banach and the classical theory of invariant means, the $\mathsf{BanachLimit}^{\uparrow}$ packet records a displayed shift-invariant positive extension row for bounded scalar sequences as finite NameCert data. It factors the Banach-internal limit surface of chapter 90 through the regular real-name route of chapter 38, the finite observation schedule of chapter 126, and the regular rational readback of chapter 301. The packet names only the finite carrier, norm-bound, positivity, shift, normalization, Real-seal, transport, replay, provenance, and local naming rows; it does not claim Hahn–Banach existence, ultrafilters, choice, quotient sequence equality, or ambient dual-space completeness.

Definition 6697.1 (Banach limit carrier)**.** *A* $\mathsf{BanachLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, A, N, P, S, U, E, H, C, Q, K)$$

with the following displayed rows:

1. *B is the bounded-sequence input row, naming the finite scalar sequence windows and the bound request that a consumer is allowed to inspect.*
2. *A is the* $\mathsf{Banach}^{\uparrow}$ *or Banach-space ambient row whose scalar and norm surface is already displayed by chapter 90.*
3. *N is the norm-bound ledger, recording that the value row is read only through the displayed bound of the bounded input sequence.*
4. *P is the positivity ledger, recording that pointwise nonnegative inspected windows are sent to a nonnegative displayed* $\mathsf{Real}^{\uparrow}$ *value row.*
5. *S is the shift-invariance row, comparing the original bounded sequence window with the one-step shifted window through the same local packet.*
6. *U is the normalization row for the constant-one sequence, binding the displayed value to the terminal real-name seal for* 1*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks for the inspected scalar windows have been displayed.*
8. *H records componentwise* hsame *transport for B, A, N, P, S, U, E.*
9. *C records displayed* Cont *replay from bounded-sequence requests through norm-bound, positivity, shift, normalization, stream-window, regular-rational, and real-seal rows.*

10. *Q is the* Pkg *provenance over* $\mathsf{BanachLimit}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *K is the local* $\mathsf{NameCert}_{\mathsf{BanachLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a Hahn–Banach extension theorem, an ultrafilter, a choice-selected invariant mean, quotient equality of bounded sequences, a host dual-space element, or ambient completeness of a dual carrier.

Theorem 6697.2 (Banach limit NameCert obligations)**.** *For every accepted* $\mathsf{BanachLimit}^{\uparrow}$ *carrier*

$$L = (B, A, N, P, S, U, E, H, C, Q, K),$$

the local $\mathsf{NameCert}_{\mathsf{BanachLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-sequence input row, Banach ambient row, norm-bound ledger, positivity ledger, shift-invariance row, normalization row, Real-seal row, componentwise transport, displayed replay, provenance, and local naming. The packet also records that positivity, shift invariance, and normalization are read only through the displayed finite rows P, S, U after the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows feeding E have been exposed.*

Proof. Project the carrier of Definition 6697.1. The tuple $B, A, N, P, S, U, E, H, C, Q, K$ is the whole accepted packet, so the bounded-sequence source, Banach ambient row, norm-bound ledger, positivity ledger, shift row, normalization row, Real seal, transport, replay, provenance, and local naming are the only available certificate fields.

The scalar readback is forced through E. That row is downstream of the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback named in the carrier, so a consumer cannot replace the displayed real-name route by host real equality, quotient sequence equality, or an anonymous dual-space functional.

The functional constraints are likewise local. Positivity is read through P, shift invariance through S, and normalization through U; each row is transported by H and replayed by C inside the same packet. Since Q and K source and name only these rows, no Hahn–Banach theorem, ultrafilter, choice selector, or ambient dual-space completeness claim is exported. □

Theorem 6697.3 (Banach limit shift-invariant boundary)**.** *Every shift-invariant read of an accepted* $\mathsf{BanachLimit}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{bounded\ sequence},\ \mathsf{Banach}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

and through the rows B, A, S, E, H, C, Q, K of Definition 6697.1. The conclusion is only that the named packet compares the original and shifted bounded-sequence windows through its local shift row; it is not an existence theorem for Banach limits and not a route to Hahn–Banach, ultrafilters, selected invariant means, quotient sequence equality, host equality, or ambient dual-space completeness.

Proof. Start with the NameCert obligations of Definition 6697.2. They expose the bounded-sequence input B, the Banach ambient row A, and the Real-facing readback route before the shift row can be consumed.

Read the shift row S through the displayed continuation C. The continuation keeps the original and shifted sequence windows inside the same $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observation route before the value row reaches the $\mathsf{Real}^{\uparrow}$ seal E.

Transport through H preserves the same rows, and provenance and local naming through Q, K keep the comparison attached to the packet. Therefore the shift-invariant boundary is exactly

the finite row S over the displayed source windows. The excluded existence, ultrafilter, choice, quotient, host-equality, and dual-completeness claims have no carrier coordinate from which they could be read. □

Falsifiable prediction: Every accepted $\mathsf{BanachLimit}^{\uparrow}$ consumer must expose the bounded-sequence row, Banach ambient row, norm-bound ledger, positivity row, shift row, normalization row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, replay, provenance, and local NameCert row before any invariant-mean read is public.

Independence witness: banach, real, streamname, regseqrat: $\mathsf{BanachLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds bounded-sequence input, Banach ambient data, norm-bound control, positivity, shift invariance, normalization, Real sealing, transport, replay, provenance, and local naming into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachLimit}^{\uparrow}$ packet is generated from bounded-sequence windows, a Banach ambient row, norm-bound, positivity, shift-invariance, normalization, StreamName, RegSeqRat, and Real-seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6697.1, Definition 6697.2, and Definition 6697.3 seed the finite Banach-limit packet over bounded-sequence, Banach, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach existence, ultrafilters, choice, quotient sequence equality, ambient dual-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, norm-bound exactness, positivity preservation, shift-row stability, normalization readback, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.BanachLimitUp.

Chapter 6698

A Concrete Naming Certificate for $\mathsf{RiemannSphere}^{\uparrow}$

Following Ahlfors on the extended complex plane and fractional linear transformations, the $\mathsf{RiemannSphere}^{\uparrow}$ packet records the compact projective complex line as finite NameCert data. It sits between the complex arithmetic carrier of chapter 39, the homogeneous-line classifier of chapter 207, the projective zero-locus surface of chapter 155, and the fractional-linear consumer of chapter 5110. The packet names the sphere carrier without importing quotient projective equality, a host compact complex manifold, selected affine charts, or an ambient equality between affine and projective presentations.

Definition 6698.1 (Riemann sphere carrier)**.** *A* $\mathsf{RiemannSphere}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, P, A, I, Q, H, T, G, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Complex}^{\uparrow}$ *affine coordinate row used for the finite chart* $[z : 1]$*.*
2. *P is the* $\mathsf{ProjectiveSpace}^{\uparrow}$ *homogeneous-line row over the complex scalar carrier.*
3. *A is the affine-chart admission row, recording the displayed inclusion of complex coordinates into homogeneous coordinates.*
4. *I is the infinity row* $[1 : 0]$*, stored as a named projective endpoint rather than as a quotient class.*
5. *Q is the compact projective coverage ledger, recording that every inspected point request is routed through the affine chart row or through the infinity row.*
6. *H records componentwise* hsame *transport for complex, projective, affine-chart, infinity, coverage, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from affine complex reads through the projective line row and from projective endpoint reads through the compact coverage ledger.*
8. *G is the* Pkg *provenance over* $\mathsf{RiemannSphere}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*,* $\mathsf{ProjectiveSpace}^{\uparrow}$*,* $\mathsf{ProjectiveVar}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSphere}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, P, A, I, Q, H, T, G, N$. It has no coordinate for quotient equality of projective representatives, a selected atlas isomorphism, a host complex-surface object, or ambient compactness.

Theorem 6698.2 (Riemann sphere NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSphere}^{\uparrow}$ *carrier* $S = (C, P, A, I, Q, H, T, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RiemannSphere}^{\uparrow}}$ *surface consists exactly of complex affine-coordinate exposure, projective homogeneous-line exposure, affine-chart admission, infinity endpoint exposure, compact projective coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer must inspect those rows before treating the packet as the extended complex plane.*

Proof. Project the carrier of Definition 6698.1. The complex row C is the only affine source, and the projective row P is the only homogeneous-line source available to the packet.

The affine chart row A and infinity row I then split the public point reads into two displayed routes. The coverage ledger Q records that a point request reaches the carrier only through one of those routes, so no unrecorded quotient representative or host point set is available.

The structural rows H, T, G, N transport, replay, source, and name exactly the same finite packet. Since no row contains host compactness, a selected atlas equality, quotient projective equality, or an ambient complex surface, the listed rows are the full NameCert obligation surface. □

Theorem 6698.3 (Riemann sphere projective-complex handoff)**.** *Every* $\mathsf{Complex}^{\uparrow}$ *affine read exported by an accepted* $\mathsf{RiemannSphere}^{\uparrow}$ *carrier factors through the displayed projective line route*

$$\mathsf{Complex}^{\uparrow},\ \mathsf{ProjectiveSpace}^{\uparrow},\ \mathsf{ProjectiveVar}^{\uparrow}$$

before a downstream projective or fractional-linear consumer may inspect it. The affine chart row A sends complex coordinates to homogeneous coordinates, the infinity row I supplies the complementary endpoint, and the coverage ledger Q keeps both reads inside BHist, hsame, Cont, Pkg*, and* NameCert *data. The handoff does not assert quotient projective equality, a host Riemann-sphere object, or ambient compact complex completeness.*

Proof. Begin with an affine read of the complex coordinate row C. The carrier admits it to the projective surface only through the chart row A, so a downstream projective consumer must first display the $\mathsf{ProjectiveSpace}^{\uparrow}$ row P.

The projective-side read then passes through the same homogeneous-line and projective-variety route named in the provenance row G. The infinity row I is carried separately, and the coverage ledger Q records that affine and infinity reads jointly exhaust the inspected point requests without identifying their representatives by an external quotient.

Finally use H, T, N to transport, replay, and name the packet. Because the excluded host objects have no coordinate in S, a $\mathsf{MobiusTransform}^{\uparrow}$ or projective consumer receives only this finite projective-complex handoff. □

Falsifiable prediction: Any accepted $\mathsf{RiemannSphere}^{\uparrow}$ consumer that omits the $\mathsf{Complex}^{\uparrow}$ affine row, the $\mathsf{ProjectiveSpace}^{\uparrow}$ homogeneous-line row, the affine-chart row, the infinity row, the coverage ledger, Pkg provenance, or the local NameCert row refutes Definition 6698.2.

Independence witness: complex, projectivespace, projectivevar, mobius-transform: $\mathsf{RiemannSphere}^{\uparrow}$ is not bijective with any listed sibling because it jointly binds affine complex coordinates, a projective homogeneous line, a distinguished infinity endpoint, compact coverage, transport, replay, provenance, and local naming into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSphere}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSphere}^{\uparrow}$ packet is generated from a ComplexUp affine coordinate row, a ProjectiveSpaceUp homogeneous-line row, affine-chart admission, an infinity endpoint, compact projective coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSphere}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6698.1, Definition 6698.2, and Definition 6698.3 seed the finite projective-complex carrier over affine complex coordinates, homogeneous-line data, infinity, coverage, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSphere}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient projective equality, a selected atlas, host compact complex manifolds, ambient complex-surface equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-chart exactness, infinity endpoint stability, compact coverage, projective-complex handoff, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSphereUp.

Chapter 6699

Cauchy Sign-Flip Budget NameCert Construction

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySignFlipBudget}^{\uparrow}$ packet records the finite sign-flip budget consumed when a regular Cauchy rational name is negated and then sealed as a Real-facing name. It sits after the finite observation windows of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the displayed negation route of chapter 459, and the terminal real-name seal of chapter 38. The packet gives later Real algebra certificates a finite budget object for sign reversal, rather than treating negation as an ambient completed-real operation.

Definition 6699.1 (Cauchy sign-flip budget carrier)**.** *A* $\mathsf{CauchySignFlipBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, D, R, F, G, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite source windows opened by a requested negation read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row recording endpoint scale, requested precision, and the sign-flipped dyadic tolerance rows attached to S.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source regular rational name over the same windows.*
4. *F is the finite sign-flip ledger, pairing each displayed dyadic endpoint in D with its negated endpoint and the same tolerance request.*
5. *G is the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *consumer row that reads S, D, R, F before any downstream algebraic consumer may use the negated name.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback, dyadic sign-flip ledger, and negation consumer row have been displayed.*
7. *H records componentwise* hsame *transport for the window, dyadic budget, readback, sign-flip ledger, negation consumer, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite sign-flip request through $S, D, R, F, G, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySignFlipBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySignFlipBudget}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, D, R, F, G, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real negation, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credentials.*

Theorem 6699.2 (Cauchy sign-flip budget window stability)**.** *Let $B = (S, D, R, F, G, E, H, C, P, N)$ and $B' = (S', D', R', F', G', E', H', C', P', N')$ be accepted* $\mathsf{CauchySignFlipBudget}^{\uparrow}$ *carriers. If the* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and structural rows agree under the displayed* hsame *and* Cont *transport, then the sign-flip ledger and terminal Real-seal route cannot diverge. Every public negation read is forced through the same finite route*

$$S \longmapsto D \longmapsto R \longmapsto F \longmapsto G \longmapsto E$$

and its transported counterpart for B'.

Proof. Project both carriers through Definition 6699.1. The public analytic coordinates are exactly the window row, dyadic budget row, regular readback row, sign-flip ledger, negation consumer row, and Real seal row.

The assumed hsame and Cont agreement identifies the source windows, the dyadic tolerance budget, and the regular readback. Since F has no independent coordinate beyond the displayed dyadic endpoint sign reversal over those rows, the sign-flip ledger is transported with the same finite windows.

The $\mathsf{RegularCauchyNegation}^{\uparrow}$ row G is downstream of S, D, R, F, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of G. Thus a Real-facing consumer cannot select a second negation route or a second seal route. The transported packet B' follows the same argument with primed rows. □

Theorem 6699.3 (Cauchy sign-flip budget Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *algebra, apartness, sum, difference, or additive-inverse consumer of an accepted* $\mathsf{CauchySignFlipBudget}^{\uparrow}$ *carrier must expose the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *sign-flip budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *consumer row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. The packet exports no quotient stream equality, selected host limit, completed-real inverse law, trichotomy, ambient completeness, or bridge credential.*

Proof. Start from the carrier inventory of Definition 6699.1. The sign-flip budget is displayed before the negation consumer row, and the negation consumer row is displayed before the terminal Real seal.

Apply Definition 6699.2. Any later algebraic or apartness read preserves the same finite window and dyadic-budget route under the stored transport and replay rows.

The excluded objects have no carrier coordinate and no provenance row. Hence they cannot be exported by the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchySignFlipBudget}^{\uparrow}$ read must expose StreamName windows, DyadicRatCore sign-flip budget rows, RegSeqRat readback, RegularCauchyNegation consumer rows, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: streamname, dyadicratcore, regseqrat, regularcauchynegation, real: $\mathsf{CauchySignFlipBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the finite

windows, dyadic sign-flip budget, regular readback, negation consumer row, terminal Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySignFlipBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySignFlipBudget}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore sign-flip budget rows, RegSeqRat readback, RegularCauchyNegation consumer rows, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySignFlipBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6699.1, Definition 6699.2, and Definition 6699.3 seed the finite sign-flip budget consumed by regular Cauchy negation and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySignFlipBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real negation, completed-real inverse laws, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window stability, dyadic sign-flip budget preservation, RegSeqRat readback, RegularCauchyNegation consumption, Real-seal nonescape, transport, replay, provenance, and local naming for $\mathsf{CauchySignFlipBudget}^{\uparrow}$.

Chapter 6700

Hyperbolic Horocycle Return Ledger NameCert Construction

Following the classical horocycle and geodesic-flow tradition of Hedlund, Hopf, and Anosov, the $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$ packet records a finite return ledger for a boundary-anchored horocycle read. It sits after the horocycle carrier of chapter 3534, the Busemann-boundary rows of chapter 3732, the Busemann-cocycle route of chapter 4134, the visual-metric read of chapter 3866, the geodesic-flow packet of chapter 3856, and the boundary-transport row of chapter 6595. The packet names finite return bookkeeping for boundary, Busemann, visual-metric, horosphere-flow, and transport rows; it is not a global recurrence theorem, an invariant measure construction, a quotient boundary flow, or a selected complete orbit.

Definition 6700.1 (Hyperbolic horocycle-return ledger carrier)**.** *A* $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, B, K, V, F, T, R, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *source row naming the boundary-tangent horocycle read.*
2. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-height row attached to the same ideal endpoint.*
3. *K is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *comparison row recording finite Busemann displacement between the departure and return windows.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual-metric row read along the same boundary anchor.*
5. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *horosphere-flow row, displaying the finite flow segment inspected by the return request.*
6. *T is the* $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ *row preserving the boundary anchor and transported visual-metric read.*
7. *R is the return ledger row. It records the departure window, return window, Busemann displacement, visual-metric comparison, and horocycle source row as one finite packet.*
8. *A is the nonescape row, refusing any read that turns the finite return ledger into a global recurrence theorem, invariant measure, quotient boundary flow, selected complete orbit, or physical observer trajectory.*

9. *H records componentwise* hsame *transport for horocycle, Busemann boundary, Busemann cocycle, visual metric, geodesic flow, boundary transport, return ledger, refusal, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through the horocycle source, boundary height, cocycle comparison, visual-metric read, flow segment, boundary transport, return ledger, refusal, provenance, and local naming.*

11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBusemannBou}$ $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, K, V, F, T, R, A, H, C, P, N$. It has no coordinate for global recurrence, invariant measures, quotient boundary-flow equality, selected complete orbits, physical observer motion, Lean verification, or bridge checking.

Theorem 6700.2 (Hyperbolic horocycle-return transport nonescape)**.** *For every accepted* HyperbolicHorocycleRetu *carrier $L = (Q, B, K, V, F, T, R, A, H, C, P, N)$, any transported return read must preserve the displayed boundary anchor, Busemann displacement, visual-metric row, horocycle source, and finite geodesic-flow segment before the return ledger R can be consumed. A consumer receives the route*

$$Q \longmapsto B \longmapsto K \longmapsto V \longmapsto F \longmapsto T \longmapsto R$$

together with A, H, C, P, N, and receives no global recurrence theorem, invariant measure, quotient boundary-flow equality, selected complete orbit, or physical observer trajectory.

Proof. Project the accepted carrier through Definition 6700.1. Rows Q, B, K, V, F, T are the visible geometric inputs to the return ledger.

The Busemann rows control the boundary-height comparison. Row B fixes the boundary-height read, and row K records the finite displacement between the departure and return windows. The visual-metric row V and boundary-transport row T keep the same boundary anchor visible through the read.

The flow row F supplies only the inspected finite geodesic-flow segment. The return ledger R packages this segment with the horocycle source and Busemann displacement, so it cannot add a selected complete orbit or a global recurrence assertion.

Finally, A, H, C, P, N refuse excluded routes and transport the displayed packet. Since none of the excluded objects has a carrier coordinate, transported return reads remain inside the finite NameCert surface. □

Theorem 6700.3 (Hyperbolic horocycle-return ledger obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleReturn}}$ *surface consists exactly of horocycle source admission, Busemann-boundary admission, Busemann-cocycle displacement comparison, visual-metric readback, finite geodesic-flow segment admission, boundary-transport preservation, return-ledger exactness, nonescape refusal, componentwise transport, displayed replay, package provenance, and local naming. No obligation is stored outside $Q, B, K, V, F, T, R, A, H, C, P, N$.*

Proof. Read the carrier inventory in Definition 6700.1. Each listed obligation is the admission or preservation condition for one displayed row.

Apply Definition 6700.2. It prevents the return ledger from being consumed before the horocycle, boundary, Busemann, visual-metric, flow, and transport rows have been displayed.

The structural rows H, C, P, N only transport, replay, source, and name that same route. Therefore the local NameCert surface is exhausted by the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$ read must expose horocycle source rows, Busemann-boundary rows, Busemann-cocycle displacement rows, visual-metric rows, finite geodesic-flow rows, boundary-transport rows, a return ledger, nonescape refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: hyperbolic-horocycle, hyperbolic-busemann-boundary, hyperbolic-busemann-cocycle, hyperbolic-visual-metric, hyperbolic-geodesic-flow, hyperbolic-boundary-transport: HyperbolicHorocycleRe is not bijective to any listed sibling because it jointly stores the return window, Busemann displacement, visual-metric comparison, finite flow segment, boundary transport, refusal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$ packet is generated from HyperbolicHorocycle source rows, HyperbolicBusemannBoundary height rows, HyperbolicBusemannCocycle displacement rows, HyperbolicVisualMetric readback, HyperbolicGeodesicFlow finite segments, HyperbolicBoundaryTransport preservation, a return ledger, nonescape refusal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6700.1, Definition 6700.2, and Definition 6700.3 seed the finite horocycle-return ledger over boundary, Busemann, visual-metric, horosphere-flow, transport, refusal, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global recurrence, invariant measures, quotient boundary-flow equality, selected complete orbits, physical observer motion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Busemann displacement comparison, visual-metric readback, finite flow-segment admission, boundary-transport preservation, return-ledger exactness, nonescape refusal, transport, replay, provenance, and local naming for $\mathsf{HyperbolicHorocycleReturnLedger}^{\uparrow}$.

Chapter 6701

A Concrete Naming Certificate for $\mathsf{IntervalHalvingTree}^{\uparrow}$

Following Bishop and Bridges on constructive bisection methods, the $\mathsf{IntervalHalvingTree}^{\uparrow}$ packet records the finite binary tree of interval halves inspected before a Real and Completion consumer reads a regular rational name. It sits after the dyadic endpoint arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. It also neighbors the bisection-tree surface of chapter 1465, the located interval bisection route of chapter 5546, and the nested interval surfaces of chapter 320 and chapter 2308. The packet names a constructive halving tree as finite NameCert data; it does not assert ambient real completeness, an infinite branch selector, quotient stream equality, trichotomy, or countable choice.

Definition 6701.1 (Interval halving tree carrier)**.** *An* $\mathsf{IntervalHalvingTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite rooted halving-tree row. Each node records a rational interval window, its depth, and the visible left/right child slots that have been opened by the construction.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for endpoints, midpoints, dyadic diameter bounds, child containment, and tolerance comparisons along every displayed edge of R.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names the finite node, endpoint, midpoint, sign, and tolerance observations a downstream consumer may inspect.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed chain of scheduled intervals and dyadic diameter bounds.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, D, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for tree nodes, dyadic ledgers, stream schedules, regular-rational handoffs, real seals, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a halving-tree request through R, D, S, Q, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalHalvingTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{IntervalBisectionTree}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalHalvingTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for a completed branch, host interval equality, quotient stream equality, selected real limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6701.2 (Interval halving tree NameCert obligations)**.** *For every accepted* $\mathsf{IntervalHalvingTree}^{\uparrow}$ *carrier*

$$T = (R, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalHalvingTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite rooted halving tree R, dyadic midpoint and containment arithmetic through D, finite observation scheduling through S, regular rational readback through Q, terminal real-name sealing through E, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N.*

Every bisection or nested-interval consumer must read the same displayed rows before it can use the tree as a certificate. The packet exports no infinite branch selector, quotient stream equality, host interval equality, countable choice, trichotomy, or ambient real-completeness row.

Proof. Project the carrier of Definition 6701.1. The projection exposes one finite rooted tree, one dyadic arithmetic ledger, one stream schedule, one regular-rational handoff, one real-seal row, and the structural rows.

The tree row R supplies the opened nodes and child slots. The dyadic ledger D checks that every displayed child is a midpoint half of its parent interval and that every exposed diameter comparison is a dyadic tolerance comparison.

The schedule row S fixes which finite observations may be read. The row Q repacks any displayed branch segment as a regular rational readback only after R, D, S have been inspected, and E seals the same route as a real-facing row.

Finally H, C, P, N transport, replay, source, and name the displayed packet. Since the carrier has no coordinate for an unbounded branch or host equality, the listed rows exhaust the local NameCert obligations. □

Theorem 6701.3 (Interval halving tree RegSeqRat handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{IntervalHalvingTree}^{\uparrow}$ *packet factors through the finite route*

$$R, D \longmapsto S \longmapsto Q \longmapsto E, H, C, P, N.$$

Thus the $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q is downstream from the displayed halving tree, midpoint arithmetic, and finite observation schedule, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is downstream from Q. No consumer can use the packet as a completed real, a selected infinite branch, or an ambient nested-interval theorem without leaving the carrier.*

Proof. Start with the displayed halving tree R and the dyadic ledger D. They provide the only interval nodes, midpoint checks, containment checks, and diameter comparisons available to the packet.

Read the finite schedule S. It determines which node observations and tolerance requests are opened for the downstream handoff, so the regular rational row cannot inspect an unlisted branch or hidden endpoint.

The row Q is then the $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the scheduled branch segment. The real-seal row E is reached only after that handoff, and the rows H, C, P, N preserve the same finite route under transport, replay, provenance, and naming.

Consequently Real and Completion consumers receive a displayed bisection-tree certificate whose finite route is compatible with the DyadicRatCore, StreamName, RegSeqRat, Real, IntervalBisectionTree, and NestedInterval sibling surfaces. The packet supplies no ambient completeness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalHalvingTree}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalHalvingTree}^{\uparrow}$ packet is generated from a finite rooted halving tree, dyadic midpoint and containment ledgers, StreamName schedules, RegSeqRat handoff rows, Real seal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalHalvingTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6701.1, Definition 6701.2, and Definition 6701.3 seed the finite bisection-tree certificate used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalHalvingTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge status, ambient real completeness, infinite branch selection, quotient stream equality, host interval equality, countable choice, or trichotomy.

Upgrade path: Obligation closure requires checked carrier admission, dyadic midpoint exactness, child containment, schedule compatibility, regular rational handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IntervalHalvingTreeUp.

Chapter 6702

A Concrete Naming Certificate for CauchyCondensationIntegralCriterion$^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the CauchyCondensationIntegralCriterion$^{\uparrow}$ packet records the finite criterion route that carries a monotone positive series source through Cauchy condensation and then through an integral-test handoff before any Real or Completion consumer reads convergence data. It consumes the condensation test of chapter 1478, the integral-test surface of chapter 2259, the bridge packet of chapter 6071, and the L10 readback corridor of chapter 291, chapter 301, and chapter 38. The certificate names the criterion as displayed finite data; it does not choose a completed sum, an improper integral value, a quotient series equality, countable-choice schedules, or ambient Real$^{\uparrow}$ completeness.

Definition 6702.1 (CauchyCondensationIntegralCriterion carrier)**.** *A* CauchyCondensationIntegralCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, D, C, I, B, W, Q, R, H, T, P, N)$$

with the following displayed rows:

1. *S is the positive decreasing series-source row, recording only inspected term windows, positivity witnesses, and monotonicity comparisons.*
2. *M is the dyadic condensation block row supplied by the condensation test route.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *scale ledger, carrying powers-of-two windows, endpoint comparisons, and dyadic tolerance slack.*
4. *C is the condensation criterion row, exposing the finite reduction from the source tail to the dyadic block tail.*
5. *I is the integral-test row, recording finite interval cells and monotone majorant or minorant comparisons.*
6. *B is the bridge row that matches the condensation blocks with integral windows through the existing condensation-integral route.*
7. *W is the finite readback schedule for Real and Completion consumers.*
8. *Q is the* RegSeqRat$^{\uparrow}$ *readback row reached from S, M, D, C, I, B, W.*
9. *R is the terminal* Real$^{\uparrow}$ *seal row read only after the regular-rational handoff is visible.*

10. *H records componentwise* hsame *transport for source, condensation, dyadic, integral, bridge, readback, seal, replay, provenance, and local naming rows.*
11. *T records displayed* Cont *replay from the criterion request through the source, condensation, integral, bridge, readback, and seal rows.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{CauchyIntegralTest}^{\uparrow}$, $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, D, C, I, B, W, Q, R, H, T, P, N$. It has no coordinate for a host improper integral, selected infinite sum, quotient equality of series, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6702.2 (CauchyCondensationIntegralCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$ *carrier $K = (S, M, D, C, I, B, W, Q, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed thirteen-row packet; positive monotone source exposure through S; condensation block exposure through M; dyadic scale exposure through D; condensation reduction through C; integral-test exposure through I; bridge exposure through B; finite schedule exposure through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q;* $\mathsf{Real}^{\uparrow}$ *sealing through R; componentwise* hsame *transport through H; displayed* Cont *replay through T; provenance through P; and local naming through N.*

Proof. Project the carrier rows from Definition 6702.1. The analytic route is exactly $S, M, D, C, I, B, W, Q, R$, and H, T, P, N are the structural certificate rows.

The source and condensation rows are read first. The condensation test sibling supplies the finite dyadic block route, while the dyadic row records the scale data needed to compare those blocks.

The integral side is then read through I and B. The integral-test sibling supplies finite interval cells, and the condensation-integral bridge orders the handoff before regular-rational readback is consumed.

Finally W, Q, R, H, T, P, N schedule, read back, seal, transport, replay, source, and name the same finite route. Since none of the excluded host objects is a carrier coordinate, none is a NameCert obligation of the criterion packet. □

Theorem 6702.3 (CauchyCondensationIntegralCriterion condensation-to-integral handoff)**.** *Every accepted* $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$ *carrier sends a positive decreasing series source to an integral-test comparison only through the displayed route*

$$S, M, D \longmapsto C \longmapsto I, B \longmapsto W, Q, R.$$

The condensation side is anchored by chapter 1478; the integral side is anchored by chapter 2259; and the bridge is anchored by chapter 6071. The route exports no selected subseries, host integral object, quotient series equality, or ambient completeness principle.

Proof. Begin with the source rows S, M, D. The source row displays only inspected monotone positive term windows, while the dyadic row fixes the finite powers-of-two scale on which condensation blocks are read.

Use the condensation test route. It places the block comparison inside C before any integral window is inspected, so the consumer cannot skip directly from the source series to a host convergence theorem.

Next use the integral-test and bridge rows. The row I provides finite interval cells, and B matches those cells with the condensation blocks through the sibling bridge.

The readback rows W, Q, R then keep the conclusion inside finite schedules, regular rational readback, and the terminal Real seal. The carrier has no coordinate for the excluded host objects, so the handoff is exactly the displayed finite route. □

Theorem 6702.4 (CauchyCondensationIntegralCriterion RegSeqRat Real nonescape)**.** *Every Real-facing or Completion-facing read of an accepted* $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$ *carrier reaches the terminal seal R only after the route*

$$S, M, D, C, I, B, W, Q, R$$

has exposed the source, condensation, dyadic, integral, bridge, schedule, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, together with structural rows H, T, P, N. Thus the criterion supplies a finite L10 convergence-read boundary and not an ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Apply Definition 6702.3. It places the analytic comparison behind the source, condensation, dyadic, integral, bridge, and finite schedule rows.

The regular rational readback row Q is downstream of that comparison. A Real-facing consumer therefore sees only the scheduled regular-rational read before the terminal seal R is public.

The structural rows preserve the same route under hsame, Cont, provenance, and local naming. Because the carrier has no row for quotient series equality, selected limits, countable-choice schedules, host real equality, or ambient completeness, the Real and Completion boundary cannot escape the displayed $S, M, D, C, I, B, W, Q, R$ route. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the positive monotone source row, condensation block row, dyadic scale row, condensation reduction row, integral-test row, bridge row, finite schedule, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: cauchy-condensation-test, cauchy-integral-test, cauchy-condensation-integral-bridge, dyadicratcore, regseqrat, real: no listed sibling supplies the whole criterion packet $K = (S, M, D, C, I, B, W, Q, R, H, T, P, N)$, because each sibling contributes only a condensation row, integral row, bridge row, dyadic ledger, regular readback, or terminal seal while this chapter binds them into one finite criterion certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}$ packet is generated from positive monotone series-source rows, condensation block rows, DyadicRatCore scale rows, condensation criterion rows, integral-test rows, condensation-integral bridge rows, finite schedules, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6702.1, Definition 6702.2, Definition 6702.3, and Definition 6702.4 seed the finite criterion route over condensation blocks, integral windows, RegSeqRat readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient improper integration, selected infinite sums, quotient series equality, countable choice, trichotomy, host real equality, Real completeness, or a completed summability theorem.

Upgrade path: Obligation closure requires checked carrier admission, condensation-test exactness, integral-test handoff, bridge exactness, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationIntegralCriterionUp.

Chapter 6703

A Concrete Naming Certificate for ChernoffFiniteExponentialBound$^{\uparrow}$

Following Chernoff's finite exponential-bound method, the ChernoffFiniteExponentialBound$^{\uparrow}$ packet records the finite moment-generating-function route by which an exponential Markov row bounds a displayed tail event. It sits after the probability source of chapter 184, the random-variable rows of chapter 185, the law and range summaries of chapter 186, the finite independence ledger of chapter 187, the dyadic exponential arithmetic of chapter 291, the terminal real seal of chapter 38, and the bounded finite-sum concentration surface of chapter 5726. The packet names only the finite MGF and exponential Markov route; it is not an asymptotic optimizer, countable-choice scheme, ambient measure-completion theorem, quotient event-equality principle, or ambient Real$^{\uparrow}$ completeness result.

Definition 6703.1 (Chernoff finite exponential-bound carrier)**.** *A* ChernoffFiniteExponentialBound$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, X, D, I, \lambda, M, A, E, R, H, T, Q, N)$$

with the following visible rows:

1. *P is the source* ProbSpace$^{\uparrow}$ *row carrying the event classifier and total-mass normalization;*
2. *X is the finite* RandomVar$^{\uparrow}$ *family whose displayed tail event is inspected;*
3. *D is the* Distribution$^{\uparrow}$ *row exposing the finite law or range summary needed for each exponential moment;*
4. *I is the finite* Independence$^{\uparrow}$ *ledger used only when the MGF row is factorized over the displayed family;*
5. *λ is the positive dyadic or real parameter row admitted for the exponential Markov read;*
6. *M is the finite MGF row, recording each displayed expectation of* $\exp(\lambda X_i)$ *or the single displayed expectation used by the packet;*
7. *A is the exponential Markov comparison row identifying the tail event with the exponential superlevel event;*
8. *E is the* DyadicRatCore$^{\uparrow}$ *arithmetic row naming the exponential numerator, denominator, product, and slack comparison used in the bound;*

9. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the resulting probability bound;*
10. *H records componentwise* hsame *transport for the probability, random-variable, distribution, independence, MGF, dyadic, and real rows;*
11. *T records the displayed* Cont *replay of the finite exponential-bound route;*
12. *Q is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HoeffdingFiniteSum}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed rows. It has no coordinate for an asymptotic limiting parameter, a selected optimizing λ, countable products, completed measure algebras, quotient-level event equality, or an ambient real-completeness oracle.

Theorem 6703.2 (Chernoff finite exponential-bound NameCert obligations)**.** *For every accepted* $\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}$ *carrier $C = (P, X, D, I, \lambda, M, A, E, R, H, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChernoffFiniteE}}$ *surface consists exactly of the finite carrier rows of Definition 6703.1. Carrier admission displays the probability source, finite random-variable family, distribution summary, finite independence ledger when present, positive parameter row, finite MGF row, exponential Markov comparison, dyadic exponential arithmetic, terminal real probability seal, componentwise transports, continuation replay, package provenance, and local naming row.*

Proof. Project the accepted carrier. Each row named by the certificate is one of the coordinates of C, so carrier formation is finite and visible.

The classifier clause is componentwise. It transports the probability source, each random-variable row, the distribution summaries, the independence ledger when the MGF is factorized, the parameter row, the MGF comparison, the dyadic arithmetic row, and the terminal real seal along the displayed hsame coordinates.

Ledger exactness follows from the same projection. Since no coordinate of C names an optimizing search, countable product, measure completion, quotient event equality, or ambient real-completeness principle, none of those objects can enter the local certificate surface. □

Theorem 6703.3 (Chernoff finite MGF handoff)**.** *Every tail-bound consumer of an accepted* $\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{RandomVar}^{\uparrow}, \mathsf{Distribution}^{\uparrow}, \mathsf{Independence}^{\uparrow}, \lambda, M, A, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, H, T, Q, N.$$

The handoff first exposes the finite probability and random-variable rows, then the displayed MGF row and exponential Markov comparison, and only then the dyadic arithmetic and terminal real bound. The route may reuse bounded finite-sum data from $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *as an input range ledger, but it does not export Hoeffding's bounded-sum surface as a Chernoff optimizer or as an ambient limiting theorem.*

Proof. Begin with Definition 6703.2. It places all probability, random-variable, distribution, independence, parameter, MGF, dyadic, real, transport, replay, provenance, and naming rows inside the accepted carrier.

The exponential Markov step reads only the displayed tail event, the positive parameter row, and the MGF row. If independence is used, it is the finite ledger I already present in the packet, so the factorization has no countable or choice-selected component.

Finally the dyadic arithmetic row E and the real seal R translate the MGF comparison into the exported probability bound. Because the route terminates at those displayed rows, downstream

consumers receive a finite exponential-bound packet and not an asymptotic optimization principle, measure-completion object, quotient event equality, or ambient real-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}}$.

Constructive story: A $\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, distribution summary, finite independence ledger when used, positive parameter row, finite MGF row, exponential Markov comparison, dyadic exponential arithmetic, RealUp probability seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6703.1, Definition 6703.2, and Definition 6703.3 seed the finite MGF and exponential Markov route through displayed probability, random-variable, distribution, independence, dyadic, real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChernoffFiniteExponentialBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close asymptotic optimization, countable choice, ambient measure completion, quotient event equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite MGF exposure, exponential Markov comparison, dyadic arithmetic exactness, terminal real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ChernoffFiniteExponentialBoundUp.

Chapter 6704

A Concrete Naming Certificate for EffectiveModulusCompactFamily$^{\uparrow}$

Following Bishop-style compact metric uniform-continuity arguments, the EffectiveModulusCompactFamily$^{\uparrow}$ packet records the finite-net, tolerance-request, covering-number, and explicit modulus rows for a compact family of maps as one NameCert surface. It sits over the compact finite-net source of chapter 129, the uniform-continuity request surface of chapter 391, the tolerance-only output of chapter 369, and the covering-budget ledger of chapter 5209. The packet is a finite data surface; it is not an open-cover compactness principle, a selected center family, a host function-space compactness theorem, or a choice-selected modulus oracle.

Definition 6704.1 (Effective modulus compact family carrier)**.** *An* EffectiveModulusCompactFamily$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, U, M, Q, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net data row, including displayed centers, cover radii, and located membership witnesses for the compact metric source.*
2. *U is the* UniformContinuity$^{\uparrow}$ *tolerance-request row, listing the target tolerances and the local continuity requests visible to the family consumer.*
3. *M is the* UniformModulus$^{\uparrow}$ *explicit modulus row extracted from the finite compact source and the displayed tolerance requests.*
4. *Q is the* FiniteMetricEntropy$^{\uparrow}$ *covering-number row, recording the finite covering budget used by the same net and tolerance data.*
5. *W is the family-window row, naming the inspected family members and the finite source windows over which K, U, M, Q are read.*
6. *H records componentwise* hsame *transport for compact finite nets, tolerance requests, explicit modulus rows, covering-number rows, family windows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a tolerance request through K, U, Q, W, H, P, N to the modulus row M.*
8. *P is the* Pkg *provenance over* EffectiveModulusCompactFamily$^{\uparrow}$, CompactMetric$^{\uparrow}$, UniformContinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, FiniteMetricEntropy$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{EffectiveModulusCompactFamily}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, U, M, Q, W, H, C, P, N$. It has no coordinate for open-cover compactness, selected centers outside the displayed net, host equality of maps, countable choice over the family, ambient completeness, or a modulus chosen outside the packet.

Lean4 ✓ `BEDC.Derived.EffectiveModulusCompactFamilyUp.EffectiveModulusCompactFamilyTasteGate`
`single carrier alignment`

Theorem 6704.2 (Effective modulus compact family NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveModulusCompactFamily}^{\uparrow}$ *carrier*

$$E = (K, U, M, Q, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveModulusCompactFamily}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, uniform-continuity tolerance-request admission through U, explicit modulus-row exposure through M, finite covering-budget exposure through Q, family-window discipline through W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6704.1. Its accepted coordinates are precisely the compact finite-net row, uniform-continuity tolerance row, explicit modulus row, covering-budget row, family-window row, and structural rows.

The analytic rows are read in dependency order. The compact source K supplies the finite-net support, U supplies the tolerance request, Q supplies the finite covering budget attached to the same net, and M records the explicit output row read by the uniform-modulus consumer.

The structural rows W, H, C, P, N keep the family members, transport, replay, provenance, and local name attached to the same finite route. Since the carrier has no coordinate for open-cover compactness, selected centers, host map equality, countable choice, ambient completeness, or an external modulus selector, the listed rows exhaust the NameCert obligations. □

Theorem 6704.3 (Effective modulus compact family finite-net handoff)**.** *Every downstream* $\mathsf{UniformModulus}^{\uparrow}$ *read from an accepted* $\mathsf{EffectiveModulusCompactFamily}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{UniformContinuity}^{\uparrow} \longmapsto \mathsf{FiniteMetricEntropy}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

Concretely, the consumer reads the compact finite-net support K, the tolerance requests U, the covering-number row Q, the family-window row W, and the structural rows H, C, P, N before it reads the explicit modulus row M. The handoff exports no open-cover compactness proof, no selected subcover, no host function-space compactness, no countable-choice schedule, and no modulus outside the displayed packet.

Proof. Begin with an accepted carrier $E = (K, U, M, Q, W, H, C, P, N)$. The compact source K is the only row that supplies finite centers and cover radii, so the handoff starts with displayed finite-net support rather than an open-cover compactness proof.

Read the tolerance and covering rows next. The row U names the target requests, and Q records the finite covering budget for the same source windows; together they determine the finite data from which the explicit modulus row can be consumed.

Finally replay the route through W, H, C, P, N and read M. The family-window, transport, replay, provenance, and naming rows preserve the same packet, so a downstream consumer cannot replace the displayed handoff by selected centers, host map equality, a choice schedule, ambient completeness, or an external modulus oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveModulusCompactFamily}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveModulusCompactFamily}^{\uparrow}$ packet is generated from CompactMetricUp finite-net data, UniformContinuityUp tolerance requests, UniformModulusUp explicit modulus rows, FiniteMetricEntropyUp covering-number rows, a family-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveModulusCompactFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6704.1, Definition 6704.2, and Definition 6704.3 seed the compact-family finite-net handoff over explicit modulus and covering-budget rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveModulusCompactFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host function-space compactness, countable choice over the family, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exactness, tolerance-request exactness, explicit modulus-row exposure, covering-budget visibility, family-window discipline, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveModulusCompactFamilyUp.

Chapter 6705

A Concrete Naming Certificate for $\mathsf{HermiteNormalForm}^{\uparrow}$

Following Hermite and the constructive integer-matrix normal-form tradition, the $\mathsf{HermiteNormalForm}^{\uparrow}$ packet records the finite row-reduction certificate by which an integer matrix is read as a triangular Hermite row. It consumes the integer arithmetic surface of chapter 31, the matrix carrier of chapter 49, the finite module-row discipline of chapter 46, and the Smith sibling comparison row of chapter 2262. The chapter names a row-reduction certificate and its triangular readback; it does not claim a selected column basis, quotient-module classification, a host Euclidean algorithm, or equality outside displayed integer row operations.

Definition 6705.1 (Hermite normal form carrier)**.** *A* $\mathsf{HermiteNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, U, T, P, D, R, S, C, Q, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *source row, recording the finite integer matrix entries, row and column bounds, and the integer-entry provenance inherited from* $\mathsf{Int}^{\uparrow}$*.*
2. *U is the unimodular row-operation ledger. It lists row swaps, sign changes, and displayed integer row additions, together with the inverse-operation rows that certify unimodular replay.*
3. *T is the triangular Hermite row, exposing the terminal row-echelon shape, pivot columns, zero-below-pivot witnesses, and bounded residue entries above each pivot.*
4. *P is the pivot divisibility row, recording the displayed divisibility and residue constraints that make the pivot data a Hermite readback rather than an arbitrary triangular matrix.*
5. *D is the* $\mathsf{SmithNormalForm}^{\uparrow}$ *comparison row, carrying the sibling diagonal-divisibility read used only to compare invariant factors and not to replace the row-reduction trace.*
6. *R records componentwise* hsame *transport for source entries, row-operation ledgers, triangular readback, pivot divisibility, Smith comparison, continuation rows, provenance, and local naming.*
7. *S records displayed* Cont *replay from source matrix inspection through unimodular row playback, triangular readback, pivot-divisibility checking, Smith sibling comparison, and local naming.*
8. *C is the* Pkg *provenance over* $\mathsf{HermiteNormalForm}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{SmithNormalForm}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

9. *Q is the nonescape row, recording that no hidden column operations, selected quotient basis, ambient Euclidean-domain theorem, or external normal-form uniqueness claim is exported by the packet.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HermiteNormalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrecorded row operations, column reduction, quotient-module equality, selected bases, host algorithmic termination, or matrix equality outside the displayed integer row-operation ledger.

Theorem 6705.2 (Hermite normal form NameCert obligations)**.** *For every accepted* $\mathsf{HermiteNormalForm}^{\uparrow}$ *carrier $H = (M, U, T, P, D, R, S, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HermiteNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission, row-operation exactness, triangular readback, pivot divisibility, Smith sibling compatibility, componentwise transport, displayed continuation replay, provenance, and nonescape through the local naming row.*

Proof. Project the carrier rows. Carrier admission exposes exactly $M, U, T, P, D, R, S, C, Q, N$, so every accepted Hermite readback begins with the displayed matrix source and the displayed integer row-operation ledger.

Row-operation exactness is forced by U. Each row swap, sign change, or integer row addition names its source row, target row, coefficient data, and inverse row, so the triangular row T can be read only after the ledger has replayed those operations.

The pivot obligations are then visible in P. These rows record the divisibility and residue constraints attached to the pivot columns, while the Smith sibling row D supplies only a comparison route for invariant-factor compatibility.

Finally R, S, C, Q, N transport, replay, source, bound, and name the same finite packet. Because Q refuses hidden column operations, quotient-module equality, selected bases, and external normal-form uniqueness, the listed rows exhaust the local NameCert surface. □

Theorem 6705.3 (Hermite normal form row-reduction readback)**.** *For every accepted* $\mathsf{HermiteNormalForm}^{\uparrow}$ *carrier $H = (M, U, T, P, D, R, S, C, Q, N)$, any exported triangular readback factors through the route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{Int}^{\uparrow} \longmapsto \mathsf{Module}^{\uparrow} \longmapsto \mathsf{SmithNormalForm}^{\uparrow} \longmapsto T.$$

Thus the consumer sees the source integer matrix, finite module row discipline, Smith sibling comparison, and only then the triangular Hermite row exposed by T.

Proof. Open the source row M. Its entries and finite bounds are supplied through $\mathsf{Mat}^{\uparrow}$ and $\mathsf{Int}^{\uparrow}$, so the row-operation ledger U has no independent arithmetic source.

The finite module discipline is read before the normal-form comparison. It fixes which row operations are admitted as module-row transformations and keeps the replay inside displayed integer rows.

The Smith sibling row D is then available only as a compatibility read for invariant-factor data. It does not replace the Hermite row ledger or introduce column operations.

Consequently the triangular row T is exposed only after the source matrix, integer entries, module-row discipline, and Smith comparison have been read. The nonescape row Q blocks any shorter route through host equality or an ambient normal-form theorem. □

Falsifiable prediction: Every accepted $\mathsf{HermiteNormalForm}^{\uparrow}$ triangular read factors through one $\mathsf{Mat}^{\uparrow}$ source row, one $\mathsf{Int}^{\uparrow}$ entry row, one unimodular row-operation ledger, one triangular readback row, one pivot-divisibility row, one $\mathsf{SmithNormalForm}^{\uparrow}$ comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: matrix, int, module, smithnormalform: $\mathsf{HermiteNormalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries unimodular row playback, triangular Hermite readback, pivot divisibility, Smith comparison, and nonescape as one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HermiteNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{HermiteNormalForm}^{\uparrow}$ packet is generated from a finite MatrixUp source row over IntUp entries, a unimodular row-operation ledger, a triangular Hermite row, a pivot-divisibility row, a SmithNormalFormUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, nonescape, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HermiteNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 6705.1, Definition 6705.2, and Definition 6705.3 seed the finite Hermite normal form certificate surface over displayed integer matrix, module-row, and Smith comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HermiteNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected column bases, quotient-module classification, host Euclidean algorithms, hidden column operations, ambient normal-form uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-operation exactness, triangular readback, pivot divisibility, Smith sibling compatibility, transport, replay, provenance, and nonescape for BEDC.Derived.HermiteNormalFormUp.

Chapter 6706

A Concrete Naming Certificate for $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$

Following Lebesgue, Vitali, Besicovitch, and Bishop–Bridges constructive measure theory, the $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ packet names the finite basis-and-covering interface that a later differentiation theorem may consume before any almost-everywhere conclusion is read. It sits between the Vitali selection surface of chapter 2184, the finite-cover ledger of chapter 2166, the bounded-overlap boundary of chapter 6339, the measurable local-average rows of chapter 92, the dyadic tolerance rows of chapter 291, the terminal real-name rows of chapter 38, and the theorem packet of chapter 4153. The packet records basis admission and finite cover handoff only; it does not assert measure completion, an almost-everywhere quotient, countable-choice selectors, or selected representatives of functions.

Definition 6706.1 (Lebesgue differentiation basis carrier)**.** *A* $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (B_0, V, E, M, A, D, R, H, C, P, N)$$

with the following displayed rows:

1. *B_0 is the basis-row ledger, naming the displayed balls or intervals, their centers, their radii, and the local nesting requests used by the differentiation basis.*
2. *V is the* $\mathsf{VitaliCover}^{\uparrow}$ *and* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *selection row, recording finite subfamily requests, disjointness tests, and cover-cell membership.*
3. *E is the* $\mathsf{BesicovitchCovering}^{\uparrow}$ *finite-overlap row, recording bounded multiplicity for the same displayed basis cells.*
4. *M is the* $\mathsf{Measure}^{\uparrow}$ *local-measure row inspected by basis cells and finite selections.*
5. *A records local-average rows for the displayed cells, without choosing a host function representative outside the packet.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for radius, overlap, measure, average, and exceptional-budget comparisons.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for local measure values, average values, and displayed error budgets.*
8. *H records componentwise* hsame *transport for basis cells, Vitali rows, Besicovitch rows, measure rows, averages, tolerances, real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the route from basis-cell requests through* B_0, V, E, M, A, D, R.
10. *P is the* Pkg *provenance over* $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$, $\mathsf{VitaliCover}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{BesicovitchCovering}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationBasis}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B_0, V, E, M, A, D, R, H, C, P, N$. It has no coordinate for a completed measure algebra, an almost-everywhere quotient, a countable basis selector, a selected function representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6706.2 (Lebesgue differentiation basis NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ *carrier*

$$B = (B_0, V, E, M, A, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationBasis}^{\uparrow}}$ *surface consists exactly of basis-cell admission, Vitali finite-selection rows, Besicovitch finite-overlap rows, Measure local-average rows, DyadicRatCore tolerance rows, Real seals, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6706.1. The displayed rows $B_0, V, E, M, A, D, R, H, C, P, N$ are the whole certificate surface.

The basis and covering rows are read first. The basis ledger B_0 exposes only displayed balls or intervals, while V routes every finite selection through the $\mathsf{VitaliCover}^{\uparrow}$ and $\mathsf{FiniteVitaliCover}^{\uparrow}$ surfaces named by chapter 2184.

The overlap and averaging rows then attach the basis to measure-facing consumers. The Besicovitch row E bounds finite multiplicity, and M, A, D, R provide the local-measure, local-average, dyadic-tolerance, and real-seal reads.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further row is present, the NameCert obligations export no measure completion, almost-everywhere quotient, countable-choice selector, selected function representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6706.3 (Lebesgue differentiation basis finite-cover handoff)**.** *Every basis read accepted by a* $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ *packet factors through the route*

$$\mathsf{VitaliCover}^{\uparrow},\ \mathsf{FiniteVitaliCover}^{\uparrow},\ \mathsf{BesicovitchCovering}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{LebesgueDifferentiation}$$

before it is offered to the theorem-level $\mathsf{LebesgueDifferentiation}^{\uparrow}$ *packet of chapter 4153. The handoff exports finite basis admission, finite selection, finite overlap, local measure, local average, tolerance, seal, transport, replay, provenance, and naming rows only.*

Proof. Start with an accepted carrier $B = (B_0, V, E, M, A, D, R, H, C, P, N)$. A theorem-level differentiation consumer can inspect basis cells only through the displayed basis ledger B_0.

The covering handoff is forced through V and E. The Vitali rows supply finite candidate selection and finite cover-cell exposure, while the Besicovitch row supplies the bounded-overlap comparison needed before local averages are consumed.

The measure-facing read is then forced through M, A, D, R. These rows expose local measure values, local averages, dyadic tolerances, and real seals for the same finite cells; they do not choose null-set representatives or complete the measure space.

The structural rows H, C, P, N carry this route into the $\mathsf{LebesgueDifferentiation}^{\uparrow}$ consumer. Thus the handoff stays finite and cannot insert an almost-everywhere quotient, countable-choice selector, selected function representative, measure completion, or ambient real-completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ packet is generated from basis-cell rows, Vitali and finite-Vitali selection rows, Besicovitch finite-overlap rows, Measure local-average rows, DyadicRatCore tolerance rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6706.1, Definition 6706.2, and Definition 6706.3 seed the finite differentiation-basis route for later Lebesgue differentiation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim almost-everywhere quotient equality, measure completion, countable-choice basis selectors, selected function representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Vitali selection routing, Besicovitch finite-overlap routing, Measure local-average routing, dyadic tolerance sealing, Real nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueDifferentiationBasisUp.

Chapter 6707

A Concrete Naming Certificate for HyperbolicShadowing$^{\uparrow}$

The HyperbolicShadowing$^{\uparrow}$ packet realizes the quasi-geodesic fellow-traveller route of chapter 9463 as finite shadowing data. It sits after the metric comparison rows of chapter 3474, the quasi-geodesic shadow packet of chapter 5802, the Morse-boundary stability rows of chapter 6040, and the boundary-window route of chapter 3813. The packet names finite fellow-traveller windows, boundary attachments, Busemann tolerance rows, and local NameCert data only; it does not complete an ideal boundary, select a global geodesic representative, quotient paths, import a host hyperbolic space, or identify the observer-expansion reading with physical dynamics.

Definition 6707.1 (Hyperbolic shadowing carrier)**.** *A* HyperbolicShadowing$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, Q, F, B, U, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* HyperbolicMetric$^{\uparrow}$ *row, exposing the finite metric comparisons used to read a quasi-geodesic route.*
2. *Q is the* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *row, recording the route samples, distortion budget, boundary endpoint, and visual shadow window being shadowed.*
3. *F is the fellow-traveller row, naming the finite comparison windows in which the route and the shadowing geodesic remain within the displayed tolerance.*
4. *B is the* MorseBoundary$^{\uparrow}$ *row that records the stability gauge and boundary-facing direction used by the fellow-traveller comparison.*
5. *U is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row, recording the finite boundary-shadow window attached to the same endpoint data.*
6. *D is the Busemann-tolerance row, carrying additive comparison budgets for the route, shadow window, and boundary attachment.*
7. *R is the* Real$^{\uparrow}$ *seal row for metric distances, tolerance budgets, and Busemann comparisons.*
8. *H records componentwise* hsame *transport for metric, quasi-geodesic, fellow-traveller, Morse-boundary, boundary-shadow, Busemann, Real, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a shadowing request through M, Q, F, B, U, D, R.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicShadowing}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$, $\mathsf{MorseBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicShadowing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, Q, F, B, U, D, R, H, C, P, N$. It has no coordinate for completed ideal-boundary quotients, selected global geodesics, quotient path equality, host hyperbolic compactification, physical observer trajectories, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6707.2 (Hyperbolic shadowing NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *carrier*

$$S = (M, Q, F, B, U, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicShadowing}^{\uparrow}}$ *surface consists exactly of HyperbolicMetric comparison rows, HyperbolicQuasiGeodesicShadow route rows, finite fellow-traveller windows, MorseBoundary stability rows, HyperbolicBoundaryShadow attachment rows, Busemann tolerance rows, Real seals, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6707.1. The rows $M, Q, F, B, U, D, R, H, C, P, N$ are the complete certificate surface.

The route and metric rows are read first. The metric row M supplies only displayed distance comparisons, and Q supplies the finite quasi-geodesic shadow data already named by chapter 5802.

The fellow-traveller and boundary rows then bind the shadowing claim. The row F records finite comparison windows, while B and U attach those windows to the Morse-boundary and boundary-shadow surfaces.

Finally D, R, H, C, P, N compare, seal, transport, replay, source, and name the same route. Since no host coordinate is present, the obligations cannot export completed boundaries, selected global geodesics, quotient path equality, physical observer dynamics, or ambient real completeness. □

Theorem 6707.3 (Hyperbolic shadowing fellow-traveller handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *carrier must read the route*

$$\mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow},\ \mathsf{MorseBoundary}^{\uparrow},\ \mathsf{HyperbolicBoundaryShadow}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Hy}$$

through finite metric comparisons, quasi-geodesic shadow rows, fellow-traveller windows, Morse stability, boundary-shadow attachment, Busemann tolerance, Real seals, transport, replay, provenance, and local naming. The handoff realizes the hyperbolic observer-expansion shadow route as finite NameCert data.

Proof. Begin with an accepted carrier $S = (M, Q, F, B, U, D, R, H, C, P, N)$. A shadowing consumer can inspect the quasi-geodesic route only after the metric row M and the quasi-geodesic shadow row Q have been displayed.

The fellow-traveller comparison is then forced through F. This row names finite comparison windows and does not select a global geodesic representative or quotient path class.

The boundary-facing evidence is read through B, U, D, R. The Morse-boundary row supplies a finite stability gauge, the boundary-shadow row supplies the endpoint window, the Busemann row supplies the additive tolerance, and the Real row seals the displayed comparisons.

The structural rows H, C, P, N transport, replay, source, and name that finite route for observer-expansion consumers. No completed ideal boundary, host hyperbolic compactification, physical observer trajectory, quotient geodesic equality, or ambient completeness row can be inserted into the handoff. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicShadowing}^{\uparrow}$ observer-expansion read factors through finite $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$, $\mathsf{MorseBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, fellow-traveller, Busemann-tolerance, transport, replay, provenance, and local naming rows.

Independence witness: hyperbolicmetric, hyperbolic-quasi-geodesic-shadow, morse-boundary, hyperbolic-boundary-shadow, hyperbolic-observer-expansion: no listed sibling is bijective to $\mathsf{HyperbolicShadowing}^{\uparrow}$ because this packet binds metric comparisons, quasi-geodesic route samples, fellow-traveller windows, Morse stability, boundary-shadow attachment, Busemann tolerance, Real seals, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicShadowing}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicShadowing}^{\uparrow}$ packet is generated from HyperbolicMetric comparison rows, HyperbolicQuasiGeodesicShadow route rows, finite fellow-traveller windows, MorseBoundary stability rows, HyperbolicBoundaryShadow attachment rows, Busemann tolerance rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicShadowing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6707.1, Definition 6707.2, and Definition 6707.3 seed the finite hyperbolic shadowing route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicShadowing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary quotients, selected global geodesics, quotient path equality, host hyperbolic compactification, physical observer trajectories, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric comparison routing, quasi-geodesic shadow routing, fellow-traveller windows, MorseBoundary stability, HyperbolicBoundaryShadow attachment, Busemann tolerance, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicShadowingUp.

Chapter 6708

A Concrete Naming Certificate for $\mathsf{FiniteGaugeCover}^{\uparrow}$

Following Cousin, Henstock, and Kurzweil, the $\mathsf{FiniteGaugeCover}^{\uparrow}$ packet records the finite gauge-cover surface used by constructive gauge integration. It sits between the local cover extraction of $\mathsf{CousinLemma}^{\uparrow}$ and $\mathsf{CousinCover}^{\uparrow}$, the tagged-sum rows of $\mathsf{RiemannSumGauge}^{\uparrow}$, the integral endpoint of $\mathsf{GaugeIntegral}^{\uparrow}$, and the arithmetic readback rows of $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Its mesh-control row also consumes the finite lower-bound cover sibling of chapter 452: finite gauge cells reuse that lower-bound ledger before the $\mathsf{RiemannSumGauge}^{\uparrow}$ and $\mathsf{GaugeIntegral}^{\uparrow}$ handoffs inspect tagged sums and endpoints. The packet names only finite gauge cells, tagged witnesses, dyadic mesh control, readback, handoff, transport, provenance, and local naming; it does not import open-cover compactness, hidden point choice, quotient cover equality, or ambient real completeness.

Definition 6708.1 (Finite gauge-cover carrier)**.** *A* $\mathsf{FiniteGaugeCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, C, T, M, Q, E, U, R, I, H, P, N)$$

with the following displayed rows:

1. *D is the gauge row, naming the local positive-radius data inspected by the cover.*
2. *C is the finite cell ledger: each accepted cell carries its interval row, center row, and gauge-admissibility witness.*
3. *T is the tag row, pairing each finite cell with the displayed point used by the tagged partition.*
4. *M is the mesh-control row, exposing only finite refinements of the displayed cells.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row used to compare radii, endpoints, and mesh bounds.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the terminal value read inspected by the integration consumer.*
7. *U is the* $\mathsf{CousinLemma}^{\uparrow}$ *and* $\mathsf{CousinCover}^{\uparrow}$ *handoff row, recording the finite cover obtained from the displayed gauge data.*
8. *R is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *handoff row for tagged sums over the same cells and tags.*
9. *I is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *handoff row exposed only after U, R, Q, E have been displayed.*
10. *H is the componentwise* hsame *transport row for gauge, cover, tag, mesh, readback, seal, and handoff coordinates.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteGaugeCover}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteGaugeCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected subcovers outside the displayed finite ledger, quotient equality of covers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.FiniteGaugeCoverUp`

Theorem 6708.2 (Finite gauge-cover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteGaugeCover}^{\uparrow}$ *carrier* $G = (D, C, T, M, Q, E, U, R, I, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteGaugeCover}^{\uparrow}}$ *surface consists exactly of gauge-row admission, finite cell-ledger exactness, tag admission, finite mesh control,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal exposure,* $\mathsf{CousinLemma}^{\uparrow}$ *and* $\mathsf{CousinCover}^{\uparrow}$ *finite-cover handoff,* $\mathsf{RiemannSumGauge}^{\uparrow}$ *tagged-sum handoff,* $\mathsf{GaugeIntegral}^{\uparrow}$ *endpoint handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. No obligation is discharged by open-cover compactness, host subcover selection, quotient cover equality, countable choice, or ambient real completeness.*

Proof. Project the carrier into its displayed rows. The gauge, finite cell ledger, tag row, and mesh row expose the only cover data visible to a consumer.

Read the arithmetic boundary through Q and E. The $\mathsf{DyadicRatCore}^{\uparrow}$ row compares the finite radii and mesh bounds, and the $\mathsf{Real}^{\uparrow}$ seal is reached only as the terminal value read.

Finally use the named handoff rows. The Cousin rows supply finite-cover extraction, the Riemann-sum row supplies tagged-sum use, and the gauge-integral row is visible only after those rows are present. Componentwise transport and replay therefore keep every NameCert obligation inside the listed packet. □

Lean4 ✓ `BEDC.Derived.FiniteGaugeCoverUp.FiniteGaugeCover namecert obligations`

Theorem 6708.3 (Cousin finite-cover handoff)**.** *Let* $G = (D, C, T, M, Q, E, U, R, I, H, P, N)$ *be an accepted* $\mathsf{FiniteGaugeCover}^{\uparrow}$ *carrier. Any consumer read that uses G as a cover witness for* $\mathsf{CousinLemma}^{\uparrow}$ *or* $\mathsf{CousinCover}^{\uparrow}$ *must first expose the gauge row D, finite cell ledger C, tag row T, mesh row M, dyadic readback Q, and handoff row U. The read exports only the displayed finite cover and no arbitrary open cover, hidden refinement selector, quotient cover equality, or choice-like subcover operation.*

Proof. Begin with the cover-facing rows D, C, T, M. These rows determine the finite ledger that the Cousin consumer is allowed to inspect.

The dyadic row Q supplies the finite arithmetic comparison needed to check that each tagged cell is gauge-admissible. The handoff row U then records the Cousin read over that same finite ledger.

Since the packet has no row for an ambient open-cover object or selected subcover outside C, T, M, any accepted read remains within the displayed finite cover surface. □

Theorem 6708.4 (Gauge-integral nonescape)**.** *Let* $G = (D, C, T, M, Q, E, U, R, I, H, P, N)$ *be an accepted* $\mathsf{FiniteGaugeCover}^{\uparrow}$ *carrier. Every* $\mathsf{GaugeIntegral}^{\uparrow}$ *consumer read that factors through G must pass through the* $\mathsf{CousinLemma}^{\uparrow}$ *and* $\mathsf{CousinCover}^{\uparrow}$ *handoff U, the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *handoff R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E before reaching I. It cannot replace those rows by open-cover compactness, an ambient real-completeness theorem, quotient stream equality, or a host limit selector.*

Proof. The integral-facing read starts from the same finite cover rows used by the Cousin handoff. Thus U is visible before a tagged-sum or integral endpoint can be consumed.

Next the tagged-sum handoff R reads the finite tags and cells, while Q and E provide the arithmetic and real-name boundary. These rows are the only path from finite cover data to the displayed endpoint row I.

The transport, replay, provenance, and local naming rows preserve this path componentwise. Therefore the consumer receives the finite gauge-integration route and no stronger compactness, equality, choice, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteGaugeCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteGaugeCover}^{\uparrow}$ packet is generated from gauge rows, finite cell ledgers, tag rows, mesh-control rows, DyadicRatCore readback, RealUp seals, CousinLemmaUp and CousinCoverUp handoffs, RiemannSumGaugeUp tagged-sum handoffs, GaugeIntegralUp endpoint handoffs, structural transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteGaugeCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6708.1, Definition 6708.2, Definition 6708.3, and Definition 6708.4 seed the finite gauge-cover route through CousinLemmaUp, CousinCoverUp, GaugeIntegralUp, RiemannSumGaugeUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteGaugeCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, quotient cover equality, hidden subcover selection, countable choice, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, gauge-cell exactness, Cousin handoff, Riemann-sum handoff, GaugeIntegral nonescape, structural replay, provenance exhaustion, and local naming for the same packet.

Chapter 6709

A Concrete Naming Certificate for $\mathsf{WeierstrassRootTest}^{\uparrow}$

Following Cauchy and Hadamard, the $\mathsf{WeierstrassRootTest}^{\uparrow}$ packet records the traditional root-test convergence route as finite BEDC certificate data over regular Cauchy tails. It sits between the displayed root-bound handoff of chapter 1481, the majorant schedule of chapter 1774, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the finite majorant route needed by Real and Completion consumers; it does not assert selected limits, quotient sequence equality, countable choice, or ambient Real completeness.

Definition 6709.1 (Weierstrass root-test carrier)**.** *A* $\mathsf{WeierstrassRootTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (A, Q, M, D, T, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the inspected series or tail-source row whose terms are read through finite windows.*
2. *Q is the root-bound witness row, recording a displayed dyadic bound strictly below the unit scale and a finite entry index after which the bound applies.*
3. *M is the finite modulus row assigning each precision request a tail threshold beyond the root-bound entry index.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the root-bound row with the requested tail precision.*
5. *T is the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *handoff row that turns A, Q, M, D into a regular Cauchy tail route.*
6. *B is the* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *majorant row, displaying the geometric majorant schedule consumed by the tail estimate.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the root-test handoff and the majorant row have both been exposed.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only from the displayed regular readback row.*
9. *H records componentwise* hsame *transport for the source, root-bound, modulus, tolerance, root-test handoff, majorant, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a root-bound request through* $A, Q, M, D, T, B, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{WeierstrassRootTest}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{WeierstrassRootTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a selected limit, quotient sequence equality, countable-choice tail selector, ambient Real completeness, host limsup theorem, or detached summation object.

Theorem 6709.2 (Weierstrass root-test NameCert obligations)**.** *For every accepted* $\mathsf{WeierstrassRootTest}^{\uparrow}$ *carrier* $W = (A, Q, M, D, T, B, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WeierstrassRootTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed twelve-row packet, root-bound exposure through* Q*, modulus exposure through* M*, dyadic tolerance exactness through* D*,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *handoff through* T*,* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *tail-majorant exposure through* B*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{Real}^{\uparrow}$ *seal non-escape through* E*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6709.1. The displayed rows are exactly $A, Q, M, D, T, B, R, E, H, C, P, N$, so every local obligation must be read from one of those rows.

Read the root-bound portion first. The row Q supplies the dyadic bound and entry index, M supplies the finite threshold for the requested precision, and D records the dyadic comparisons used by that request.

Apply the root-test and majorant rows. The handoff T is the $\mathsf{RegularCauchyRootTest}^{\uparrow}$ route from the displayed root-bound data to a regular Cauchy tail, while B is the $\mathsf{CauchyMajorantSeries}^{\uparrow}$ schedule that records the finite tail majorant consumed by the same request.

The rows R and E are downstream endpoints. The regular readback R is available only after T and B, and the terminal seal E is available only after R.

Finally use H, C, P, N to transport, replay, source, and name the same route. Since the carrier contains no selected limit, quotient sequence equality, countable-choice tail selector, ambient Real completeness, host limsup theorem, or detached summation object, the listed rows exhaust the local NameCert surface. □

Theorem 6709.3 (Weierstrass root-test tail majorant)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{WeierstrassRootTest}^{\uparrow}$ *carrier must read the finite route*

$$A, Q, M, D \longmapsto \mathsf{RegularCauchyRootTest}^{\uparrow}(T) \longmapsto \mathsf{CauchyMajorantSeries}^{\uparrow}(B) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(R) \longmapsto \mathsf{Real}^{\uparrow}(E)$$

before it can use the exported convergence surface. The route gives a regular-Cauchy root-test majorant, not a selected limiting value, quotient sequence equality, countable-choice schedule, ambient Real completeness, host limsup theorem, or detached infinite summation object.

Proof. Start with a precision request against the carrier. The rows Q, M, D display the root-bound, finite threshold, and dyadic tolerance data attached to the source row A.

Use Definition 1481.3. It identifies the root-test handoff T as a regular Cauchy tail route read only from the displayed source, root-bound, modulus, and dyadic rows.

Next use Definition 1774.3. The majorant row B supplies the finite tail-control schedule that a Real or Completion consumer must inspect before any endpoint readback is available.

The readback row R and seal row E then terminate the route through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Transport, replay, provenance, and local naming preserve this order, so no consumer receives selected limits, quotient sequence equality, countable choice, ambient Real completeness, a host limsup theorem, or an unrecorded summation object. □

Falsifiable prediction: Every accepted $\mathsf{WeierstrassRootTest}^{\uparrow}$ endpoint read factors through a root-bound witness, finite modulus, DyadicRatCore tolerance ledger, RegularCauchyRootTest handoff, CauchyMajorantSeries tail schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: regular-cauchy-root-test, cauchy-majorant-series, regseqrat, dyadicrat-core, real: $\mathsf{WeierstrassRootTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries root-bound data, a finite modulus, dyadic tolerance comparisons, regular-Cauchy root-test handoff, majorant schedule, regular readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeierstrassRootTest}^{\uparrow}}$.

Constructive story: A $\mathsf{WeierstrassRootTest}^{\uparrow}$ packet is generated from a displayed source row, root-bound witness, finite modulus, DyadicRatCore tolerance ledger, RegularCauchyRootTest handoff, CauchyMajorantSeries tail-majorant row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeierstrassRootTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6709.1, Definition 6709.2, and Definition 6709.3 seed the finite root-test majorant route from displayed root-bound data through regular Cauchy readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeierstrassRootTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient sequence equality, countable choice, ambient Real completeness, host limsup theorems, detached infinite summation objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, root-bound exactness, modulus coverage, dyadic tolerance exactness, root-test handoff, majorant tail control, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.WeierstrassRootTestUp.

Chapter 6710

A Concrete Naming Certificate for DvoretzkyFiniteDimensionalSection$^{\uparrow}$

Following Dvoretzky's finite-dimensional almost-Euclidean section theorem and Milman's finite-dimensional formulation, the DvoretzkyFiniteDimensionalSection$^{\uparrow}$ packet records displayed witness data for a fixed finite normed-space window. It sits over the Banach carrier of chapter 90, the inner-product comparison rows of chapter 86, the vector-space carrier of chapter 47, the metric rows of chapter 57, and the terminal real-name seal of chapter 38. The packet is a finite BEDC witness surface; it does not assert a global Dvoretzky existence theorem, a choice-selected subspace, a measure concentration theorem, or ambient Banach-space completeness.

Definition 6710.1 (Dvoretzky finite-dimensional section carrier)**.** *A* DvoretzkyFiniteDimensionalSection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (V, N, I, M, E, D, H, C, P, Q)$$

with the following displayed rows:

1. *V is the finite* VecSpace$^{\uparrow}$ *window: a named finite vector list, its span rules, scalar reads, and the finite coordinate table used by the section.*
2. *N is the normed-space row inherited from the local* BanachSpace$^{\uparrow}$ *surface, restricted to the displayed window and not to an ambient complete Banach object.*
3. *I is the* InnerProduct$^{\uparrow}$ *and Euclidean-candidate row, recording the comparison form, Gram entries, and the finite Euclidean norm candidate on the same section window.*
4. *M is the* Met$^{\uparrow}$ *comparison row, recording the finite metric reads for the norm row and the Euclidean-candidate row.*
5. *E is the* Real$^{\uparrow}$ *scalar seal for all displayed norm, inner-product, distance, and distortion values.*
6. *D is the distortion ledger, consisting of explicit real-sealed lower and upper comparison entries for the displayed window.*
7. *H stores componentwise* hsame *transport for V, N, I, M, E, D.*
8. *C stores displayed* Cont *replay from vector-space coordinates to norm, metric, inner-product, and distortion reads.*
9. *P is the* Pkg *provenance over* DvoretzkyFiniteDimensionalSection$^{\uparrow}$, BanachSpace$^{\uparrow}$, InnerProduct$^{\uparrow}$, VecSpace$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. Q is the local $\mathsf{NameCert}_{\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}}$ *row.*
The classifier reads only $V, N, I, M, E, D, H, C, P, Q$. It has no coordinate for an arbitrary ambient Banach space, a selected high-dimensional subspace, a concentration argument, quotient equality of norms, or a completed infinite-dimensional object.

Lean4 ✓ `BEDC.Derived.DvoretzkyFiniteDimensionalSectionUp.DvoretzkyFiniteDimensionalSectionTasteGate`
`single carrier alignment`

Theorem 6710.2 (Dvoretzky finite-dimensional section obligations)**.** *For every accepted* DvoretzkyFiniteDimensional *carrier $B = (V, N, I, M, E, D, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, finite vector-space window exposure, normed-space row restriction, Euclidean-candidate row display, metric comparison row display,* $\mathsf{Real}^{\uparrow}$ *scalar sealing, distortion-ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6710.1. The finite vector-space source is V, the norm row is N, the Euclidean-candidate row is I, the metric comparison row is M, and the Real-sealed distortion ledger is E, D.

The structural rows H, C, P, Q transport, replay, source, and name those displayed rows. They do not add a second source of norm comparison or a hidden subspace selector.

Thus every accepted obligation is read from $V, N, I, M, E, D, H, C, P, Q$. Any global existence theorem, concentration estimate, selected subspace, quotient norm equality, or ambient completeness principle would require a coordinate not present in the packet. □

Theorem 6710.3 (Dvoretzky Euclidean window readback)**.** *Every public Euclidean-window read exported by an accepted* $\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}$ *carrier exposes the same finite vector-space window V, norm row N, inner-product candidate I, metric comparison row M,* $\mathsf{Real}^{\uparrow}$ *scalar seal E, distortion ledger D, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row Q before any section claim is public. The read is finite witness data only; it does not promote the packet to a global Dvoretzky theorem or to a choice-selected ambient subspace.*

Proof. Start from the carrier rows of Definition 6710.1. A Euclidean-window consumer must first read the finite coordinate window V and the two comparison surfaces N and I.

Next read M, E, D. These rows place every metric and distortion value behind the displayed metric comparison and the terminal real-name seal, so the consumer sees finite comparison entries rather than an ambient norm theorem.

Finally use H, C, P, Q to transport, replay, source, and name the same rows. Since the carrier has no global existence, concentration, selected-subspace, quotient-equality, or completeness coordinate, the public read remains the displayed finite witness packet. □

Falsifiable prediction: A purported $\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}$ read that omits the displayed finite vector-space window, Euclidean-candidate row, metric comparison row, or Real-sealed distortion ledger is not accepted by this carrier.

Independence witness: banach, innerproduct, vecspace, metric, real: the Banach carrier lacks the Euclidean-candidate and distortion rows, the inner-product row lacks the normed-space and Banach provenance rows, the vector-space carrier lacks the metric and Real-sealed distortion ledger, the metric row lacks the vector-space and inner-product source rows, and the Real seal lacks the section window and comparison ledger; none alone carries $V, N, I, M, E, D, H, C, P, Q$.

Closure status for $\mathsf{NameCert}_{\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}}$.

Constructive story: A $\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}$ packet is generated from a finite vector-space window, normed-space row, Euclidean-candidate row, metric comparison row, Real scalar seal, distortion ledger, transport, replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6710.1, Definition 6710.2, and Definition 6710.3 bind the finite witness packet and its Euclidean-window readback boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DvoretzkyFiniteDimensionalSection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim a global Dvoretzky existence theorem, choice-selected subspace, measure concentration theorem, quotient norm equality, ambient Banach-space completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability for the section and distortion rows, ledger nonescape, and local NameCert coverage for the displayed finite witness packet.

Chapter 6711

A Concrete Naming Certificate for $\mathsf{FreydMitchellEmbedding}^{\uparrow}$

Following Freyd and Mitchell, the $\mathsf{FreydMitchellEmbedding}^{\uparrow}$ packet records the finite certificate surface needed to name the exact embedding route from a small abelian category into a module category. It depends on the abelian-category carrier of chapter 176, the functor carrier of chapter 62, and the module carrier of chapter 46; it also uses the natural-transformation interface of chapter 63 to keep comparison rows explicit.

6711.1 Carrier

Definition 6711.1 (FreydMitchellEmbeddingUp displayed carrier)**.** *A* $\mathsf{FreydMitchellEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{F} = (A, R, M, E, X, K, C, H, P, N)$$

with the following displayed rows.

1. *A is an* $\mathsf{AbelianCat}^{\uparrow}$ *source row whose object, morphism, zero, biproduct, kernel, cokernel, image, coimage, and factorisation data are the visible rows of chapter 176.*
2. *R is the ring or algebra row over which the target module category is displayed.*
3. *M is a* $\mathsf{Module}^{\uparrow}$ *target row naming the module carrier, hom rows, additive structure, scalar action, and componentwise* hsame *transport inherited from chapter 46.*
4. *E is a* $\mathsf{Functor}^{\uparrow}$ *row from A into the displayed module category; it contains object mapping, morphism mapping, identity preservation, composition preservation, and codomain classifier transport.*
5. *X is an exactness ledger: for every visible kernel, cokernel, image, coimage, and epi–mono factorisation row in A, the image under E is recorded as the corresponding kernel, cokernel, image, coimage, or exact sequence row in M.*
6. *K is a faithfulness ledger: if two displayed morphisms of A have the same transported module morphism under E, the ledger returns the original morphism-classifier equality in A.*
7. *C is a comparison row for naturality and finite diagram compatibility, expressed through* $\mathsf{NatTrans}^{\uparrow}$ *rows when two displayed finite diagrams in A are transported into M.*

8. *H is componentwise* hsame *transport for A, R, M, E, X, K, C.*
9. *P is package provenance tying the carrier to the finite source presentation, the target module presentation, and the embedding rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$ *row.*

The carrier names the finite BEDC obligation surface for an exact faithful embedding packet. It does not assert that every small abelian category admits such a packet, nor does it construct a universal module category from arbitrary external data.

Theorem 6711.2 (FreydMitchellEmbeddingUp NameCert obligations)**.** *For the carrier of Definition 6711.1, the local* $\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for $\mathcal{F} = (A, R, M, E, X, K, C, H, P, N)$; source admission through* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$*; target admission through* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$*; embedding admission through* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$*; naturality comparison through* $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$*; exactness replay through X; faithfulness replay through K; componentwise transport through H; provenance through P; and local naming through N.*

Proof. Read the abelian-category source row first. Its additive, kernel, cokernel, image, coimage, and factorisation fields are the only exactness rows that the embedding packet is allowed to consume.

Next read the target module row and the functor row. The module row supplies the additive target hom structure and scalar action, while the functor row supplies object mapping, morphism mapping, identity preservation, and composition preservation. These are the rows on which the exactness ledger X and the faithfulness ledger K are indexed.

Finally read the comparison, transport, provenance, and name rows. The $\mathsf{NatTrans}^{\uparrow}$ comparison row records finite diagram compatibility; H transports every displayed component; P records the finite source and target presentations; and N names the packet. Repacking these rows gives precisely the displayed $\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$ surface and no full embedding existence theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FreydMitchellEmbedding}^{\uparrow}$ packet is generated from an abelian-category source row, a module target row, an exact faithful functor row, finite exactness and faithfulness ledgers, naturality comparison rows, componentwise hsame transport, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 6711.1 and Definition 6711.2 seed the finite BEDC certificate surface for an exact faithful embedding packet from an abelian-category carrier to a module carrier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FreydMitchellEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim the Freyd–Mitchell embedding theorem, existence for arbitrary small abelian categories, target ring construction, or Lean verification.

Upgrade path: Formalization must introduce the displayed carrier and prove the NameCert obligation surface without adding an external embedding existence premise.

Chapter 6712

A Concrete Naming Certificate for $\mathsf{SequentialCauchyFilter}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentialCauchyFilter}^{\uparrow}$ packet records the finite route by which sequence-indexed Cauchy evidence becomes a displayed Cauchy-filter read before Real or Completion consumers inspect terminal data. It sits between the saturated filter surface of chapter 333, the finite basis generator of chapter 1501, the sequence-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The certificate names the sequential filter as finite BEDC rows; it does not turn a Cauchy sequence into a quotient stream, selected limit, countable-choice schedule, host equality proof, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6712.1 (Sequential Cauchy filter carrier)**.** *A* $\mathsf{SequentialCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F_{\mathrm{seq}} = (S, B, Q, D, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *sequence-window row, exposing exactly the finite index windows inspected by the sequential Cauchy request.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row generated from the displayed tail windows and their common-refinement witnesses.*
3. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation row reached from B before any completion-facing consumer may read filter data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radius, tail comparison budget, and endpoint scale used by the displayed Cauchy test.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same sequence windows and dyadic tolerances.*
6. *R is the sequential route ledger, ordering S, D, G, B, Q so the filter read is visibly generated from sequential evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after the sequential route ledger and filter rows have been displayed.*

8. *H records componentwise* hsame *transport for sequence windows, filter bases, filter saturation, dyadic tolerances, regular readback, route ledger, terminal seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a sequential Cauchy request through S, D, G, B, Q, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{SequentialCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, Q, D, G, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.SequentialCauchyFilterUp.SequentialCauchyFilterTasteGate single carrier alignment`

Theorem 6712.2 (Sequential Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCauchyFilter}^{\uparrow}$ *carrier $F_{\mathrm{seq}} = (S, B, Q, D, G, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; sequence-window exposure through S; filter-basis generation through B; Cauchy-filter saturation through Q; dyadic tolerance exposure through D; regular-rational readback through G; sequential route ordering through R; terminal Real-seal boundary through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6712.1. The analytic rows are exactly S, B, Q, D, G, R, E, and the structural rows are exactly H, C, P, N.

The sequence data are read first. The window row S exposes the inspected indices, the dyadic row D fixes the tolerance budget, and the regular readback row G records the rational evidence attached to those windows.

The filter rows are downstream of that route. The basis row B is generated from the displayed sequential data, and the saturation row Q is read only after the basis has been shown.

Finally, R, E, H, C, P, N order, seal, transport, replay, source, and name the same finite packet. Since quotient streams, selected limits, countable choice, host equality, trichotomy, ambient completeness, and bridge credentials are absent from the carrier, none is a local NameCert obligation. □

Theorem 6712.3 (Sequential Cauchy filter RegSeqRat route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{SequentialCauchyFilter}^{\uparrow}$ *packet reaches the terminal seal only through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports finite StreamName, RegSeqRat, DyadicRatCore, CauchyFilterBasis, CauchyFilter, and Real rows, together with BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *support. It exports no quotient stream, selected limit, countable-choice function, host equality proof, trichotomy decision, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Start with the obligation surface of Definition 6712.2. It forces every accepted read to expose the finite sequence windows, dyadic tolerance ledger, and regular rational readback before a filter basis is available.

The filter handoff then passes through B and Q. A completion-facing consumer therefore receives the Cauchy-filter row as a saturation of the displayed sequential basis, not as an external filter or quotient stream.

The terminal seal E is downstream of the route ledger R. Transport, replay, provenance, and local naming preserve the same finite route, and no excluded coordinate is present from which a selected limit, countable-choice schedule, host equality proof, trichotomy decision, ambient completeness theorem, or bridge credential could be exported. □

Falsifiable prediction: An accepted $\mathsf{SequentialCauchyFilter}^{\uparrow}$ consumer that reaches a $\mathsf{CauchyFilter}^{\uparrow}$ row or terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: cauchyfilter, cauchy-filter-basis, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole sequential packet $F_{\mathrm{seq}} = (S, B, Q, D, G, R, E, H, C, P, N)$, because each sibling contributes only a saturated filter, basis generator, window row, regular readback, dyadic tolerance ledger, or terminal seal while this chapter binds them into one sequence-indexed Cauchy-filter certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCauchyFilter}^{\uparrow}$ packet is generated from StreamName sequence windows, CauchyFilterBasis generation rows, CauchyFilter saturation rows, DyadicRatCore tolerance ledgers, RegSeqRat readbacks, sequential route ledgers, terminal RealUp seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6712.1, Definition 6712.2, and Definition 6712.3 seed the finite sequential Cauchy-filter route over StreamName, RegSeqRat, DyadicRatCore, CauchyFilterBasis, CauchyFilter, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality, trichotomy, RealUp completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-window exposure, filter-basis generation, Cauchy-filter saturation, dyadic tolerance exactness, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialCauchyFilterUp.

Chapter 6713

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ packet records the finite tail-filter seal used when regular Cauchy sequence evidence from section 9238.1 is routed into the L10 Real and Completion corridor. It sits after the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet turns the regular Cauchy tail discipline into a displayed filter seal, not into a quotient stream, selected limit, countable-choice schedule, host equality proof, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6713.1 (Regular Cauchy filter seal carrier)**.** *A* $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, W, F, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, recording the finite windows opened by the regular Cauchy tail request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic tail radius, comparison scale, and endpoint budget.*
4. *W is the tail-seal witness row, stating that the displayed windows and dyadic budget give the required regular Cauchy tail comparison.*
5. *F is the filter-facing seal row generated from S, R, D, W before any completion consumer may inspect the tail-filter route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the StreamName, RegSeqRat, DyadicRatCore, and filter-facing rows have been displayed.*
7. *L is the ledger row connecting the carrier to the regular Cauchy diagonal and tail-seal markers of section 9238.1.*
8. *H records componentwise* hsame *transport for tail windows, regular readback, dyadic tolerance, tail witness, filter seal, terminal seal, ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a regular Cauchy tail request through S, R, D, W, F, E, L.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, W, F, E, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host equality, trichotomy, ambient Real completeness, or bridge credentials.

Theorem 6713.2 (Regular Cauchy filter seal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ *carrier $T = (S, R, D, W, F, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; tail-window exposure through S; regular-rational readback through R; dyadic tolerance exposure through D; tail-seal witnessing through W; filter-facing sealing through F; terminal Real sealing through E; vision-ledger routing through L; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6713.1. The analytic tail route is exactly S, R, D, W, F, E, L, and H, C, P, N are the structural rows.

The regular Cauchy evidence is read through finite windows and dyadic budgets. The row S opens the inspected tail windows, R supplies the regular rational readback, and D fixes the tolerance scale.

The seal rows are downstream. The witness row W records the tail comparison, the filter-facing row F packages that comparison as a seal, and the terminal row E is exposed only after the earlier rows are visible.

Finally, the ledger and structural rows bind the route to the regular Cauchy vision site and replay it for later consumers. Because no excluded object is a carrier coordinate, quotient streams, selected limits, countable choice, host equality, trichotomy, ambient completeness, and bridge credentials cannot become local obligations. □

Theorem 6713.3 (Regular Cauchy filter seal tail route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ *packet reads the tail-filter seal only through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyFilterSeal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route realizes the regular Cauchy diagonal and tail-seal rows of section 9238.1 as finite StreamName, RegSeqRat, DyadicRatCore, and Real seal data. It exports no quotient stream, selected limit, countable-choice schedule, host equality proof, trichotomy decision, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Use the obligation surface of Definition 6713.2. It forces each accepted read to expose S, R, D before the tail witness or filter-facing seal can be consumed.

The row W records the regular Cauchy tail comparison inside the displayed dyadic budget, and F packages that comparison as the filter-facing seal. The terminal row E is therefore downstream of the finite tail route, not a selected limit.

The ledger row L ties this route to the regular Cauchy diagonal and tail-seal markers, while H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no excluded coordinate, the stated nonescape claims follow. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ consumer that reaches a filter-facing seal or terminal $\mathsf{Real}^{\uparrow}$ row without first displaying the $\mathsf{StreamName}^{\uparrow}$ tail window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-vision: no listed sibling supplies the whole tail-filter seal packet $T = (S, R, D, W, F, E, L, H, C, P, N)$, because each sibling contributes only a window row, regular readback, tolerance ledger, terminal seal, or vision marker while this chapter binds them into one finite seal certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSeal}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyFilterSeal}^{\uparrow}$ packet is generated from StreamName tail windows, RegSeqRat readback rows, DyadicRatCore tolerance ledgers, tail-seal witness rows, filter-facing seal rows, terminal RealUp seal boundaries, vision ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSeal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6713.1, Definition 6713.2, and Definition 6713.3 seed the finite tail-filter seal route over StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSeal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality, trichotomy, RealUp completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exposure, RegSeqRat readback, dyadic tolerance exactness, tail-seal witnessing, filter-facing seal nonescape, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterSealUp.

Chapter 6714

A Concrete Naming Certificate for $\mathsf{DAlembertRatioTest}^{\uparrow}$

Following d'Alembert and Bishop–Bridges on constructive tests for real series, the $\mathsf{DAlembertRatioTest}^{\uparrow}$ packet records the classical ratio-test promise as a finite BEDC NameCert surface. It sits beside the real-series source of chapter 1581, the rational margin data of chapter 37, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It also refines the existing ratio-test corridor of chapter 2244 by naming the d'Alembert ratio-bound evidence as the explicit handoff from displayed adjacent terms to a Cauchy-facing convergence route.

Definition 6714.1 (d'Alembert ratio test carrier)**.** *A* $\mathsf{DAlembertRatioTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, Q, M, D, W, R, C, E, H, J, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing only the inspected term stream and finite partial-sum windows.*
2. *A is the adjacent-term row, listing the displayed pair windows a_n, a_{n+1} after the relevant source threshold.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *margin row, recording the rational witness $q < 1$ used by the d'Alembert comparison.*
4. *M is the ratio-bound row, displaying the tail comparison $|a_{n+1}| \leq q|a_n|$ only through the rows A, Q, W, R, D.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, translating the rational margin into the dyadic budgets consumed by later Cauchy requests.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the adjacent terms, margin checks, partial sums, and tail requests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for all displayed adjacent-term, ratio-bound, and partial-sum observations.*
8. *C is the Cauchy handoff row, recording the finite route from the displayed ratio bound to a regular tail-control request.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, A, Q, M, D, W, R, C have all been displayed.*
10. *H records componentwise* hsame *transport for the source, adjacent-term, margin, dyadic, window, readback, handoff, seal, replay, provenance, and local-name rows.*
11. *J records displayed* Cont *replay from a convergence request through the same rows.*
12. *P is the* Pkg *provenance over* $\mathsf{DAlembertRatioTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DAlembertRatioTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, Q, M, D, W, R, C, E, H, J, P, N$. It has no coordinate for a host limit of ratios, quotient equality of real-name streams, a selected infinite sum, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.DAlembertRatioTestUp`

Theorem 6714.2 (d'Alembert ratio test NameCert obligations)**.** *For every accepted* $\mathsf{DAlembertRatioTest}^{\uparrow}$ *carrier $K = (S, A, Q, M, D, W, R, C, E, H, J, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DAlembertRatioTest}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure, adjacent-term window exactness, rational margin exposure through* $\mathsf{Rat}^{\uparrow}$*, ratio-bound exactness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance translation,* $\mathsf{StreamName}^{\uparrow}$ *window service,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Cauchy handoff, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The row S supplies only the real-series source and the finite partial-sum windows already visible through the $\mathsf{RealSeries}^{\uparrow}$ packet.

The analytic ratio data are the adjacent-term row A, the rational margin row Q, and the displayed comparison row M. These rows name a bound below one after the chosen threshold, but they do not name a host ratio limit or a quotient of real-name streams.

The dyadic row D, window row W, and readback row R convert that comparison into finite observations at requested tolerances. The Cauchy handoff C is therefore downstream of the displayed ratio-bound evidence, not an independent convergence theorem.

Finally, the endpoint seal E and structural rows H, J, P, N preserve the same route. Since none of the refused objects appears among the carrier coordinates, the listed obligations exhaust the local certificate surface. □

Theorem 6714.3 (d'Alembert ratio test Cauchy handoff)**.** *Every Cauchy-facing consumer of an accepted* $\mathsf{DAlembertRatioTest}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $S, A, Q, M, D, W, R, C, E, H, J, P, N$ before it can use the packet as convergence evidence. In particular, the handoff exports only displayed tail-control rows and never a selected infinite sum, a quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier route. The source and adjacent-term rows force every request to pass through the displayed $\mathsf{RealSeries}^{\uparrow}$ term windows before the ratio-bound row is visible.

The rational margin and dyadic rows then translate the bound below one into the finite tolerance language read by $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. Thus the handoff row C is reached only after the displayed ratio evidence has been inspected at the requested scale.

The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of C and the structural rows. No carrier coordinate remains that could supply a selected sum or host completeness principle, so the stated route is the entire Cauchy-facing handoff. □

Theorem 6714.4 (d'Alembert ratio test Real seal boundary)**.** *For every accepted* $\mathsf{DAlembertRatioTest}^{\uparrow}$ *carrier* $K = (S, A, Q, M, D, W, R, C, E, H, J, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after the real-series source, adjacent-term windows, rational margin, displayed ratio bound, dyadic tolerance ledger, finite stream windows, regular readback, and Cauchy handoff have all been consumed. The seal cannot be read as an independent Real value, a host ratio-limit witness, or a completeness theorem.*

Proof. The definition places E after the analytic rows S, A, Q, M, the tolerance and observation rows D, W, R, and the handoff row C. A consumer that reads E must therefore preserve that displayed dependency route.

The transport, replay, provenance, and local-name rows do not add a new endpoint. They only carry the same packet through hsame, Cont, Pkg, and NameCert structure.

Since no coordinate stores a detached Real value, host limit, or completeness witness, the Real seal boundary is exactly the displayed downstream boundary. □

Theorem 6714.5 (d'Alembert ratio test RealSeries route)**.** *Every* $\mathsf{RealSeries}^{\uparrow}$ *convergence consumer of an accepted* $\mathsf{DAlembertRatioTest}^{\uparrow}$ *packet must read, in order, the source* $\mathsf{RealSeries}^{\uparrow}$ *windows, the adjacent quotient rows, the rational margin data, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route gives the consumer no host infinite sum, selected limit, quotient series equality, countable choice, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows $S, A, Q, M, D, W, R, C, E, H, J, P, N$. The source read begins at S, the adjacent quotient evidence is exposed only through A and M, and the rational margin is carried by Q before any tolerance ledger can be consumed.

Apply the Cauchy handoff of Definition 6714.3 and the Real seal boundary of Definition 6714.4. These force the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows to appear before the terminal seal E can serve a convergence consumer.

Finally read the structural rows H, J, P, N. Their provenance contains only the displayed BEDC rows, so the consumer has no coordinate for a host infinite sum, selected limit, quotient series equality, countable-choice schedule, trichotomy branch, or ambient completeness principle. □

Falsifiable prediction: An accepted $\mathsf{DAlembertRatioTest}^{\uparrow}$ endpoint read that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying a $\mathsf{Rat}^{\uparrow}$ margin below 1, adjacent $\mathsf{RealSeries}^{\uparrow}$ term windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance translation, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback fails the carrier surface.

Independence witness: realseries, rat, dyadicratcore, streamname, regseqrat, real, real-series-ratio-test: the sibling packets separately expose series terms, rational margins, dyadic budgets, finite windows, regular readback, endpoint sealing, or a general ratio-test corridor. None by itself binds the d'Alembert margin, adjacent-term comparison, Cauchy handoff, Real seal, transport, replay, provenance, and local naming into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DAlembertRatioTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DAlembertRatioTest}^{\uparrow}$ packet is generated from a RealSeries source row, adjacent-term windows, a rational margin below one, displayed ratio-bound evidence, a DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readback, a Cauchy handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DAlembertRatioTest}^{\uparrow}}$, seed)

Scope closed: Definition 6714.1, Definition 6714.2, Definition 6714.3, and Definition 6714.4 seed the finite d'Alembert ratio-test route over RealSeries, Rat, DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DAlembertRatioTest}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.DAlembertRatioTestUp`

Bridge status: none

Not claimed: This chapter does not claim a checked ratio-test theorem, host ratio limits, quotient stream equality, selected infinite sums, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ratio-bound exactness, dyadic tolerance translation, Cauchy handoff, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.DAlembertRatioTestUp.

Chapter 6715

A Concrete Naming Certificate for $\mathsf{BourbakiWitt}^{\uparrow}$

Following Bourbaki and Witt, the $\mathsf{BourbakiWitt}^{\uparrow}$ packet records the constructive order-theoretic fixed-point route for a progressive monotone self-map on a displayed chain-complete poset source. It sits after the separated order carrier of chapter 53, the certified-family bound surface of chapter 5155, and the complete-lattice monotone fixed-row route of chapter 5919. The packet names only finite chain rows, displayed upper-bound witnesses, approximation ledgers, transport, replay, provenance, and local naming; it does not assert Zorn's lemma, full choice, unrestricted chain completeness, quotient order equality, or an ambient maximal-chain theorem.

Definition 6715.1 (Bourbaki-Witt carrier)**.** *A* $\mathsf{BourbakiWitt}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, F, G, U, A, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Poset}^{\uparrow}$ *source row equipped with the displayed chain-completeness boundary accepted by this packet.*
2. *F is the progressive monotone self-map row, recording both monotonicity and the displayed comparison $x \leq F(x)$ for carried elements.*
3. *G is the finite chain ledger, listing the inspected chain windows and their inherited* $\mathsf{Poset}^{\uparrow}$ *comparison rows.*
4. *U is the displayed chain-bound witness row assigning an upper-bound name to each inspected chain window.*
5. *A is the finite approximation ledger, recording the iterated or successor-stage rows used by a fixed-point consumer before the bound row is read.*
6. *H records componentwise* hsame *transport for O, F, G, U, A, C, P, N.*
7. *C records displayed* Cont *replay from the poset source and progressive monotone map through chain windows, upper-bound witnesses, approximation rows, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{BourbakiWitt}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{CompleteLattice}^{\uparrow}$, $\mathsf{TarskiFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BourbakiWitt}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, F, G, U, A, H, C, P, N$. It has no coordinate for a choice-selected maximal chain, Zorn principle, unrestricted chain completeness, quotient order equality, host ordinal recursion, or ambient lattice semantics.

Theorem 6715.2 (Bourbaki-Witt NameCert obligations)**.** *For every accepted* $\mathsf{BourbakiWitt}^{\uparrow}$ *carrier* $B = (O, F, G, U, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BourbakiWitt}^{\uparrow}}$ *surface consists exactly of carrier admission for the chain-complete* $\mathsf{Poset}^{\uparrow}$ *source row, progressive monotone map exposure through* F*, chain-window exposure through* G*, displayed chain-bound witnesses through* U*, finite approximation coverage through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6715.1. The displayed rows are exactly $O, F, G, U, A, H, C, P, N$, so every local obligation must be read from one of those rows.

Read the order-theoretic rows first. The source row O supplies the separated partial-order carrier and the displayed chain-completeness boundary, while F supplies the progressive monotone self-map row consumed by the fixed-point route.

The chain rows G, U, A then account for the constructive content. Each inspected chain window is named before its upper-bound witness is used, and each approximation step is recorded before a consumer may read a fixed-row handoff.

Finally use H, C, P, N to transport, replay, source, and name the same route. Since no carrier coordinate stores a maximal-chain selector, Zorn principle, unrestricted chain-completeness theorem, quotient order equality, host ordinal recursion, or ambient complete-lattice semantics, the listed rows exhaust the local NameCert surface. □

Theorem 6715.3 (Bourbaki-Witt chain handoff)**.** *Every fixed-point consumer of an accepted* $\mathsf{BourbakiWitt}^{\uparrow}$ *packet must read the route*

$$\mathsf{Poset}^{\uparrow}(O), F, G, U, A, H, C, P, N$$

before it may use the exported fixed-row surface. The route hands finite chain windows and their displayed upper-bound witnesses to $\mathsf{CompleteLattice}^{\uparrow}$ *and* $\mathsf{TarskiFixedPoint}^{\uparrow}$ *consumers without selecting a maximal chain, invoking Zorn's lemma, asserting full choice, or claiming unrestricted chain completeness outside the displayed source row.*

Proof. Start from the $\mathsf{Poset}^{\uparrow}$ source row. It supplies the only order comparison surface visible to the packet, and the chain-completeness boundary is read as part of that displayed source rather than as an ambient theorem.

Apply the progressive monotone map row. The row F records both monotonicity over the carried order and the progressive comparison used to extend each inspected approximation window.

Use the chain ledger and bound rows next. The row G names the finite chain windows, U supplies their displayed upper-bound witnesses, and A records the approximation ledger that a fixed-point consumer may replay.

The handoff to $\mathsf{CompleteLattice}^{\uparrow}$ or $\mathsf{TarskiFixedPoint}^{\uparrow}$ is therefore mediated by H, C, P, N. Transport and replay preserve the same finite rows, while provenance and local naming prevent the endpoint from importing Zorn, full choice, unrestricted chain completeness, quotient order equality, or a detached maximal-chain construction. □

Falsifiable prediction: Every accepted $\mathsf{BourbakiWitt}^{\uparrow}$ fixed-row read factors through a chain-complete Poset source row, progressive monotone map row, finite chain ledger, displayed chain-bound witness row, finite approximation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: poset, complete-lattice, tarski-fixed-point: $\mathsf{BourbakiWitt}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries progressive monotone map data, finite chain windows, displayed chain-bound witnesses, approximation rows, transport, replay, provenance, and local naming without a maximal-chain selector.

Closure status for $\mathsf{NameCert}_{\mathsf{BourbakiWitt}^{\uparrow}}$.

Constructive story: A $\mathsf{BourbakiWitt}^{\uparrow}$ packet is generated from a chain-complete Poset source row, progressive monotone map row, finite chain ledger, displayed upper-bound witnesses, finite approximation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BourbakiWitt}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6715.1, Definition 6715.2, and Definition 6715.3 seed the finite chain-complete fixed-point route between Poset, CompleteLattice, and TarskiFixedPoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BourbakiWitt}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Zorn's lemma, full choice, unrestricted chain completeness, quotient order equality, host ordinal recursion, selected maximal chains, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, progressive monotone exactness, chain-window coverage, displayed upper-bound witnesses, approximation ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.BourbakiWittUp.

Chapter 6716

A Concrete Naming Certificate for $\mathsf{CompactChainUniformModulus}^{\uparrow}$

Following Heine and Cantor, the $\mathsf{CompactChainUniformModulus}^{\uparrow}$ packet records the finite cover-refinement chain by which a compact metric source and a continuous-map certificate yield a tolerance-only uniform-modulus handoff. It realizes the finite cover-refinement branch of chapter 9237, sitting between the compact carrier of chapter 60, the metric-distance rows of chapter 57, the continuous-map local-radius ledger of chapter 130, the rational lower-bound readback of chapter 37, the uniform-modulus consumer of chapter 369, and the terminal real-name seal of chapter 38. The packet names only a finite chain of compact cover cells, local-radius witnesses, lower-bound comparisons, handoff rows, transport, provenance, and local naming; it does not select a global modulus by choice, pass through quotient cover equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6716.1 (Compact chain uniform-modulus carrier)**.** *A* $\mathsf{CompactChainUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U_{\mathrm{cc}} = (K, M, F, C, L, Q, R, E, H, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-cover row, exposing the compact source only through a displayed finite cover or finite net ledger.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare source cells, centers, and refinement radii.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *local-modulus row, attaching a positive local radius to each displayed center and tolerance read.*
4. *C is the cover-chain row, ordering the finite refinement from compact cover cells to local-radius cells before any uniform-modulus handoff is exported.*
5. *L is the finite lower-bound row over the displayed positive rational radii.*
6. *Q is the* $\mathsf{Rat}^{\uparrow}$ *readback row for lower-bound comparisons, positivity, and finite minimum inspection.*
7. *R is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, available only after K, M, F, C, L, Q have been displayed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for real-valued consumers of the same modulus packet.*
9. *H records componentwise* hsame *transport for compact cover, metric, continuous-map, chain, lower-bound, rational readback, uniform-modulus, seal, provenance, and naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactChainUniformModulus}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactChainUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. There is no coordinate for selected subcovers outside K, quotient equality of covers, a choice-selected modulus function, hidden point witnesses, or ambient real completeness.

Theorem 6716.2 (Compact chain uniform-modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactChainUniformModulus}^{\uparrow}$ *carrier* $U_{\mathrm{cc}} = (K, M, F, C, L, Q, R, E, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactChainUniform}}$ *surface is exhausted by five finite obligation rows:*

1. *compact cover-chain admission for K and C;*
2. *metric and continuous-map local-radius compatibility for M and F;*
3. *finite lower-bound exactness for L and* $\mathsf{Rat}^{\uparrow}$ *readback Q;*
4. *uniform-modulus and real-seal handoff through R and E;*
5. *refusal, transport, provenance, and local naming through H, P, N.*

No obligation is discharged by choice-selected moduli, quotient cover equality, hidden subcover selection, external point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier into the displayed rows. The compact and cover-chain rows K, C expose the only finite source cover inspected by the packet, while M, F attach the metric and local-radius data to that same chain.

The lower-bound rows are then read finitely. The row L records the finite positive-radius comparison, and Q supplies the rational readback used to keep the selected tolerance inside every displayed local radius.

Finally, the handoff rows R, E are downstream of those cover-chain and lower-bound rows. Componentwise transport, provenance, and local naming preserve the same packet, so the excluded choice, quotient, hidden-selection, point-equality, and completeness principles cannot enter the local certificate surface. □

Theorem 6716.3 (Finite cover-chain handoff)**.** *Every consumer that reads an accepted* CompactChainUniformModu *packet as a* $\mathsf{UniformModulus}^{\uparrow}$ *witness must first display the compact cover row K, metric row M, continuous local-radius row F, cover-chain row C, finite lower-bound row L, and* $\mathsf{Rat}^{\uparrow}$ *readback row Q. The exported object is the handoff row R together with the optional* $\mathsf{Real}^{\uparrow}$ *seal E; it is not a selected modulus function or a quotient class of covers.*

Proof. The carrier definition places R downstream of K, M, F, C, L, Q. A consumer cannot inspect the uniform-modulus row without first reading the finite compact cover, the metric comparisons, the local continuous-map radii, and the rational lower-bound ledger.

The structural rows H, P, N keep that read componentwise inside the same BHist packet. Since no carrier coordinate stores selected modulus functions or quotient cover classes, the handoff exports only the displayed finite cover-chain route. □

Falsifiable prediction: An accepted $\mathsf{CompactChainUniformModulus}^{\uparrow}$ consumer that reaches a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying the compact cover chain and finite $\mathsf{Rat}^{\uparrow}$ lower-bound ledger refutes the carrier surface.

Independence witness: compact, metric, continuousmap, rat, uniformmodulus, real: no listed sibling supplies the whole packet $U_{\mathrm{cc}} = (K, M, F, C, L, Q, R, E, H, P, N)$, because each sibling contributes only compact cover data, distance rows, local continuity radii, rational comparison, a uniform-modulus consumer, or a terminal seal while this chapter binds them into one finite cover-chain certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactChainUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactChainUniformModulus}^{\uparrow}$ packet is generated from Compact finite-cover rows, MetricSpace distance rows, ContinuousMap local-radius rows, finite cover-chain rows, Rat lower-bound readback, UniformModulus handoff rows, RealUp seals, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactChainUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6716.1, Definition 6716.2, and Definition 6716.3 seed the finite compact cover-chain route from Compact, MetricSpace, ContinuousMap, Rat, UniformModulus, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactChainUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected global moduli, quotient cover equality, arbitrary subcover selection, point-choice principles, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, cover-chain exactness, local-radius compatibility, finite lower-bound readback, uniform-modulus handoff, Real-seal nonescape, transport, provenance, refusal, and local naming for BEDC.Derived.CompactChainUniformModulusUp.

Chapter 6717

A Concrete Naming Certificate for $\mathsf{RecursivePolishPresentation}^{\uparrow}$

Following Moschovakis and Weihrauch, the $\mathsf{RecursivePolishPresentation}^{\uparrow}$ packet records the finite naming surface of a recursively presented Polish space: a complete separable metric carrier, a scheduled dense sequence, and a decidable dyadic distance table exposed through BEDC histories. It sits over the complete-separable metric source of chapter 1735, the Polish-space synthesis of chapter 1490, the metric-distance rows of chapter 57, the finite stream-name windows of chapter 126, the regular rational readback of chapter 301, the represented-space boundary of chapter 1494, the effective Polish packet of chapter 1706, and the terminal real-name seal of chapter 38. The packet is a recursive presentation as finite NameCert data; it is not a quotient completion, a selected limit, a host equality principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6717.1 (Recursive Polish presentation carrier)**.** *A* $\mathsf{RecursivePolishPresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P_{\mathrm{rec}} = (M, C, D, T, S, R, A, E, V, H, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row naming the metric carrier.*
2. *C is the* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *row over the same metric source.*
3. *D is the recursive dense-sequence row, exposing dense points only through scheduled finite indices.*
4. *T is the dyadic distance-table row for the displayed dense indices.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row controlling which dense names and table entries are opened.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance comparisons.*
7. *A is the* $\mathsf{PolishSpace}^{\uparrow}$ *handoff row, available only after M, C, D, T, S, R are displayed.*
8. *E is the effective-name row consumed by* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *and represented-space readers.*
9. *V is the* $\mathsf{Real}^{\uparrow}$ *seal row for terminal distance values.*
10. *H records componentwise* hsame *transport for metric, complete-separable, dense-sequence, table, stream, readback, Polish, effective, seal, provenance, and local naming rows.*

11. *G is the* Pkg *provenance over* $\mathsf{RecursivePolishPresentation}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RecursivePolishPresentation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for selected dense points outside D, quotient-completion representatives, equality of represented names, unscheduled table queries, countable-choice schedules, or ambient real completeness.

Theorem 6717.2 (Recursive Polish presentation NameCert obligations)**.** *For every accepted* $\mathsf{RecursivePolishPresentation}^{\uparrow}$ *carrier* $P_{\mathrm{rec}} = (M, C, D, T, S, R, A, E, V, H, G, N)$, *the local* $\mathsf{NameCert}_{\mathsf{RecursivePolishPresent}}$ *surface is exhausted by five finite obligation rows:*

1. *recursive dense-sequence admission through D over the common metric row M;*
2. *dyadic distance-table exactness through T, S, R;*
3. *dense coverage of the complete-separable and Polish rows C, A through the same scheduled names;*
4. *represented-space and effective-Polish handoff through E and the terminal* $\mathsf{Real}^{\uparrow}$ *seal V;*
5. *refusal, componentwise transport, provenance, replay, and local naming through H, G, N.*

No obligation is discharged by a selected limit, quotient completion equality, external dense-point choice, host equality of names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows. The metric and complete-separable rows M, C supply the analytic source, while the recursive dense row D and stream row S restrict all dense-name access to finite scheduled windows.

The table rows are read next. The dyadic table T and regular-rational readback R certify the inspected distance comparisons for the same dense indices, so no unscheduled table query or selected dense point is available.

Finally, A, E, V, H, G, N package the Polish, effective, represented-space, seal, transport, provenance, and local naming boundaries. Since the carrier excludes quotient completions, selected limits, host equality of names, countable-choice schedules, and ambient completeness coordinates, those objects cannot become local NameCert obligations. □

Theorem 6717.3 (Recursive dense-table handoff)**.** *Every* $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, *or* $\mathsf{RepresentedSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{RecursivePolishPresentation}^{\uparrow}$ *packet must first expose the common metric row M, complete-separable row C, recursive dense-sequence row D, dyadic distance-table row T, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The consumer receives only the displayed recursive dense-table route, not a quotient completion, selected dense sequence, selected limit, or host representation equality.*

Proof. The carrier orders the consumer-facing rows A, E, V after M, C, D, T, S, R. Thus a Polish, complete-separable, effective, or represented-space read cannot bypass the recursive dense sequence or the dyadic table ledger.

Componentwise transport and provenance preserve the same finite packet. Because no excluded coordinate is present in the carrier, the consumer receives the scheduled dense-table route and no quotient, selected, choice-like, or host-equality object. □

Falsifiable prediction: An accepted $\mathsf{RecursivePolishPresentation}^{\uparrow}$ consumer that opens a $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, or $\mathsf{RepresentedSpace}^{\uparrow}$ row without displaying the recursive dense sequence, dyadic distance table, finite $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: complete-separable-metric, polishspace, metric, streamname, regseqrat, representedspace, effective-polish-space, real: no listed sibling supplies the whole packet $P_{\mathrm{rec}} = (M, C, D, T, S, R, A, E, V, H, G, N)$, because each sibling contributes only a metric carrier, complete-separable synthesis, Polish handoff, finite window, rational readback, represented-space boundary, effective-name route, or terminal real seal while this chapter binds them into one recursive presentation certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RecursivePolishPresentation}^{\uparrow}}$.

Constructive story: A $\mathsf{RecursivePolishPresentation}^{\uparrow}$ packet is generated from MetricSpace rows, CompleteSeparableMetric rows, recursive dense-sequence schedules, dyadic distance-table rows, StreamName windows, RegSeqRat readback, PolishSpace handoffs, EffectivePolishSpace and RepresentedSpace handoffs, RealUp seals, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursivePolishPresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6717.1, Definition 6717.2, and Definition 6717.3 seed the recursive Polish presentation route over CompleteSeparableMetric, PolishSpace, MetricSpace, StreamName, RegSeqRat, represented-space, effective-Polish, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursivePolishPresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completions, selected limits, selected dense points, host equality of represented names, countable choice, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, recursive dense-sequence exactness, dyadic distance-table exactness, dense coverage, represented-space handoff, refusal ledger, transport, provenance, replay, and local naming for BEDC.Derived.RecursivePolishPresentationUp.

Chapter 6718

A Concrete Naming Certificate for $\mathsf{GrothendieckTopology}^{\uparrow}$

Following Grothendieck's SGA 4 site formalism, the $\mathsf{GrothendieckTopology}^{\uparrow}$ packet records a coverage sieve system on a certified category as finite BEDC rows. It sits after the category and functorial source surfaces of chapter 61 and chapter 62, before the restriction interface of chapter 150 and the gluing interface of chapter 151. The packet names maximal sieves, pullback-stable covering sieves, and local transitivity; it does not export a topos, a host powerset of arrows, an external saturation operator, a selected sheafification, or quotient equality of covering families.

Definition 6718.1 (Grothendieck topology carrier)**.** *A* $\mathsf{GrothendieckTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (C, S, M, P, T, H, R, Q, N)$$

with the following displayed rows:

1. *C is the source* $\mathsf{Cat}^{\uparrow}$ *row, carrying objects, hom rows, identity arrows, composition rows, and their classifier laws.*
2. *S is the sieve-family row, assigning to each displayed object U a finite ledger of sieves on U whose membership is read as arrows into U and is closed under precomposition in C.*
3. *M is the maximal-sieve row, recording that the sieve containing every displayed arrow into U is a covering sieve at U.*
4. *P is the pullback-stability row: for every covering sieve R on U and every displayed arrow $f : V \to U$, the pulled-back sieve*

   $$f^*R = \{g : W \to V \mid f \circ g \in R\}$$

   is a covering sieve on V.
5. *T is the transitivity row: if R covers U and Q is a sieve on U whose pullback along every displayed member of R covers the corresponding domain, then Q covers U.*
6. *H records componentwise* hsame *transport for category rows, sieve membership rows, maximal rows, pullback rows, transitivity rows, replay rows, provenance rows, and local naming rows.*
7. *R records displayed* Cont *replay from a site request through C, S, M, P, T.*
8. *Q is the* Pkg *provenance over* $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{GrothendieckTopology}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $C, S, M, P, T, H, R, Q, N$. It has no coordinate for host arrow powersets, quotient sieves, selected sheafification, topos objects, global choice of covers, or bridge credentials.

Theorem 6718.2 (Grothendieck topology NameCert obligations)**.** *For every accepted* $\mathsf{GrothendieckTopology}^{\uparrow}$ *carrier $J = (C, S, M, P, T, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{GrothendieckTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed nine-row packet; category-source exposure through C; sieve-family exposure through S; maximal-sieve coverage through M; pullback stability through P; transitivity and local character through T; componentwise* hsame *transport through H; displayed* Cont *replay through R; provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 6718.1. The site data are exactly C, S, M, P, T, and the structural rows are exactly H, R, Q, N.

The category row is read first. It supplies the object and hom rows on which sieve membership is meaningful, together with identities and composition so precomposition and pullback sieves are internal BEDC rows.

The coverage rows are then forced in their traditional order. The maximal row covers each object, the pullback row transports any covering sieve along a displayed arrow, and the transitivity row performs the local-character test using only the displayed pullback covers.

Finally, H, R, Q, N transport, replay, source, and name the same finite packet. Since no excluded coordinate is present in the carrier, host arrow powersets, quotient sieves, selected sheafification, topos objects, global cover choice, and bridge credentials are not local NameCert obligations. □

Theorem 6718.3 (Grothendieck topology sieve pullback stability)**.** *Let $J = (C, S, M, P, T, H, R, Q, N)$ be an accepted* $\mathsf{GrothendieckTopology}^{\uparrow}$ *packet. If R_U is a covering sieve on a displayed object U and $f : V \to U$ is a displayed arrow of the* $\mathsf{Cat}^{\uparrow}$ *row C, then the pulled-back sieve*

$$f^* R_U = \{g : W \to V \mid f \circ g \in R_U\}$$

is a covering sieve on V in the same packet. The read of this cover factors through C, S, P, H, R, Q, N and does not use host equality of arrows, a quotient of sieves, or a selected replacement cover.

Proof. Expose the accepted carrier and read the category row C. The source, target, and composition fields of C make the displayed expression $f \circ g$ a certified arrow into U whenever g is a displayed arrow into V.

Use the sieve-family row S to read membership of R_U as a finite BEDC predicate on arrows into U. The pullback row P is exactly the coverage clause that maps this displayed covering sieve and displayed arrow f to a covering sieve on V.

Transport and replay do not change the coverage decision. The rows H, R, Q, N keep the pulled-back sieve attached to the same packet, provenance, and local naming surface, so no host arrow equality, quotient sieve, or selected replacement cover is inserted. □

Theorem 6718.4 (Grothendieck topology nonescape)**.** *Every* $\mathsf{PreSheaf}^{\uparrow}$ *or* $\mathsf{Sheaf}^{\uparrow}$ *consumer of an accepted* $\mathsf{GrothendieckTopology}^{\uparrow}$ *packet reads coverage only through the finite route*

$$\mathsf{Cat}^{\uparrow} \longmapsto \mathsf{Functor}^{\uparrow} \longmapsto \mathsf{PreSheaf}^{\uparrow} \longmapsto \mathsf{Sheaf}^{\uparrow}$$

together with the displayed rows $C, S, M, P, T, H, R, Q, N$. No consumer can obtain a topos object, a host powerset of arrows, a quotient of covering families, a global choice of covers, an external sheafification, or a bridge credential from the packet.

Proof. Start with the obligation surface of Definition 6718.2. It exhausts the site-facing data through the category row, sieve-family row, maximal row, pullback-stability row, and transitivity row.

A presheaf consumer can only inspect the certified source category and restriction-facing sieve rows. The functorial source is therefore the displayed $\mathsf{Cat}^{\uparrow}$ and $\mathsf{Functor}^{\uparrow}$ corridor, not an ambient category of all arrows.

A sheaf consumer adds locality and gluing over the displayed covering sieves. The pullback-stability theorem Definition 6718.3 keeps every restricted cover inside the same finite packet, and the transitivity row supplies the local-character closure used by the gluing surface.

The structural rows H, R, Q, N transport, replay, source, and name only those displayed rows. Because the carrier has no excluded coordinate, none of the stated external objects can escape from the packet. □

Falsifiable prediction: An accepted $\mathsf{GrothendieckTopology}^{\uparrow}$ consumer that treats a pulled-back covering sieve as available without displaying the source $\mathsf{Cat}^{\uparrow}$ row, the sieve-family row, and the pullback-stability row refutes the carrier surface.

Independence witness: category, functor, presheaf, sheaf: no listed sibling supplies the whole site packet $J = (C, S, M, P, T, H, R, Q, N)$, because the siblings contribute category, functor, restriction, or gluing rows while this chapter binds maximal covers, sieve pullback stability, and local transitivity into one Grothendieck topology NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{GrothendieckTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{GrothendieckTopology}^{\uparrow}$ packet is generated from a CategoryUp source row, sieve-family rows, maximal-sieve cover rows, pullback-stability rows, transitivity and local-character rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GrothendieckTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6718.1, Definition 6718.2, Definition 6718.3, and Definition 6718.4 seed the finite site route over CategoryUp, FunctorUp, PreSheafUp, and SheafUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GrothendieckTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close topos objects, host arrow powersets, quotient sieves, global choice of covers, external sheafification, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sieve-family exposure, maximal-sieve coverage, pullback-stability exactness, transitivity exactness, presheaf and sheaf consumer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.GrothendieckTopologyUp.

Chapter 6719

A Concrete Naming Certificate for $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$

Following Morrey and Evans on Sobolev embeddings, the $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ packet records the finite BEDC surface by which a displayed Sobolev-energy row yields a Morrey modulus row when the exponent and dimension rows certify the supercritical inequality. It sits after the $\mathsf{Sobolev}^{\uparrow}$ carrier of chapter 440, the $\mathsf{MorreySpace}^{\uparrow}$ target surface of chapter 4004, the scalar readback of chapter 38, the regular rational tolerance route of chapter 301, the metric source of chapter 57, and the compact consumer surface of chapter 129. The packet names only displayed rows: Sobolev energy, exponent inequality, local ball windows, rational modulus readback, transport, provenance, and local $\mathsf{NameCert}$ data.

Definition 6719.1 (Sobolev–Morrey embedding carrier)**.** *A* $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, Q, B, M, R, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, recording the displayed function, weak derivative budget, exponent row, and domain window.*
2. *Q is the supercritical exponent row, recording the dimension comparison and the rational witness for the Morrey exponent gap.*
3. *B is the local ball-window row, recording the finite metric balls on which oscillation estimates are requested.*
4. *M is the* $\mathsf{MorreySpace}^{\uparrow}$ *modulus row, recording the candidate Hölder-Morrey control produced from S, Q, B.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for rational tolerances, energy bounds, and endpoint scalar values.*
6. *H stores componentwise* hsame *transport for S, Q, B, M, R, P, N.*
7. *C stores the displayed* Cont *routes from the Sobolev-energy row and exponent row to the Morrey modulus row.*
8. *P is the* Pkg *provenance over* $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{MorreySpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SobolevMorreyEmbedding}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host-selected representative, quotient equality of functions, open-cover compactness, a global differentiability theorem, or an ambient real-completeness principle.

Theorem 6719.2 (Sobolev–Morrey modulus obligation)**.** *For every accepted* $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ *carrier $E = (S, Q, B, M, R, H, C, P, N)$, a consumer may read the Morrey modulus row M only after reading the Sobolev-energy row S, the supercritical exponent row Q, the local ball-window row B, and the rational scalar readback row R. Agreement on H, C, P, N preserves those rows but cannot synthesize a Morrey modulus without the displayed S, Q, B, R data.*

Proof. Project the carrier through Definition 6719.1. The definition lists M as a row produced from the displayed Sobolev energy, exponent gap, ball windows, and rational scalar readback rather than from a hidden host theorem.

The route row C records the finite passage from S, Q, B, R to M. Componentwise transport through H and package provenance through P preserve the same displayed rows, so the read of M remains tied to the visible energy and exponent obligations.

Since the carrier has no coordinate for a selected representative, quotient equality of functions, or ambient completeness, no structural row can fill the missing analytic input. Thus every accepted modulus read must consume the displayed S, Q, B, R rows first. □

Theorem 6719.3 (Sobolev–Morrey compact consumer surface)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ *packet must read the same finite route*

$$S, Q, B, M, R, H, C, P, N$$

before using the Morrey modulus as a compact-family control. The consumer may transport the modulus through hsame *and replay the displayed* Cont *route, but it cannot import Rellich–Kondrachov compactness, choose a subsequence, or identify functions by host quotient equality.*

Proof. Begin with the modulus obligation of Definition 6719.2. It forces the Morrey control row M to be read together with the Sobolev source, exponent gap, ball windows, and rational scalar readback.

The compact consumer then reads M only as a modulus row compatible with the $\mathsf{Met}^{\uparrow}$ and $\mathsf{CompactMetric}^{\uparrow}$ sibling surfaces. The transport and replay rows H, C preserve this finite route; they do not add subsequence selection or open-cover compactness.

The absence clauses in the carrier definition rule out host quotient equality and selected representatives. Hence the compact-facing use is a finite modulus-consumer surface, not a compactness theorem or a hidden Rellich–Kondrachov conclusion. □

Falsifiable prediction: An accepted $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ packet cannot export a Morrey modulus row unless the Sobolev-energy row, exponent-gap witness, local ball-window row, and rational scalar readback row are all displayed in the same carrier.

Independence witness: sobolev, morrey-space, real, regseqrat, metric, compactmetric: $\mathsf{SobolevMorreyEmbedding}$ joins the energy, exponent-gap, local-window, and modulus-consumer rows; the sibling packets do not by themselves expose the full embedding handoff surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevMorreyEmbedding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevMorreyEmbedding}^{\uparrow}$ packet is generated from Sobolev-energy, supercritical exponent, local ball-window, Morrey modulus, rational readback, hsame transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{SobolevMorreyEmbedding}\uparrow}$, seed)

Scope closed: Definition 6719.1, Definition 6719.2, and Definition 6719.3 bind the seed carrier and its compact-consumer handoff surface.

Formal verification: FormalStatus(NameCert$_{\text{SobolevMorreyEmbedding}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove the analytic Sobolev–Morrey inequality, Rellich–Kondrachov compactness, Lean verification, or a standard bridge.

Upgrade path: Scoped closure requires a checked obligation surface for the exponent-gap row, local ball-window row, and Morrey modulus readback route.

Chapter 6720

A Concrete Naming Certificate for $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$

Following Cantor's nested intersection theorem and the Bishop–Bridges treatment of complete metric spaces by explicit Cauchy moduli, the $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$ packet records the finite route by which a shrinking nested-ball schedule supplies a modulus before Real or Completion consumers read an endpoint. It sits beside the Cantor intersection route of chapter 4359, the complete-metric source of chapter 128, the nested closed-ball carrier of chapter 3526, and the L10 readback route through chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a displayed modulus carrier, not a quotient completion, selected representative, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6720.1 (CantorCompleteMetricModulus carrier)**.** *A* $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{ccm}} = (K, B, D, S, Q, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row for the metric source on which the nested schedule is read.*
2. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row carrying the descending closed-ball spine and its containment witnesses.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger recording the positive radius bounds and requested tolerance comparisons.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing inspected centers, radii, and containment checks.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the displayed center windows and dyadic shrink ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after K, B, D, S, Q have been displayed.*
7. *U is the modulus row: for each displayed dyadic tolerance it names a finite stage after which the nested-ball diameter and the regular readback are inside that tolerance.*
8. *H records componentwise* hsame *transport for complete-metric, nested-ball, dyadic, stream, regular-readback, Real-seal, modulus, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through* $K, B, D, S, Q, E, U, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorCompleteMetricModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, D, S, Q, E, U, H, C, P, N$. It has no coordinate for quotient completion equality, selected nested-point representatives, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6720.2 (CantorCompleteMetricModulus NameCert obligations)**.** *For every accepted* $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$ *carrier* $M_{\mathrm{ccm}} = (K, B, D, S, Q, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CantorCompleteMet}}$ *surface consists exactly of carrier exposure for the complete-metric, nested-ball, dyadic, stream, regular-readback, Real-seal, modulus, transport, replay, provenance, and naming rows; shrink exactness for D; finite-window readback for S, Q; modulus exactness for U; Real-seal nonescape for E; and refusal of quotient completions, selected representatives, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6720.1. The visible rows are exactly $K, B, D, S, Q, E, U, H, C, P, N$, and every obligation must be attached to one of those rows.

The completion and nested-ball source is read through K and B. The shrink ledger D supplies the dyadic radius and tolerance comparisons used by the nested schedule.

The finite readback stage is S, Q, E. Windows in S expose only finitely many centers and containment checks, Q records the corresponding regular rational evidence, and E is reached only after that readback is displayed.

The modulus row U binds the schedule to each requested dyadic tolerance. Transport, replay, provenance, and naming through H, C, P, N preserve that same finite route, so no excluded quotient, selected point, choice, compactness, or ambient-completeness coordinate can be exported. □

Theorem 6720.3 (CantorCompleteMetricModulus Real-seal route)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$ *packet must read the route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{NestedClosedBall}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

with the modulus row U displayed before the terminal $\mathsf{Real}^{\uparrow}$ *seal E is public. The route gives L10 consumers an explicit finite Cantor-completeness modulus and exports no quotient completion, selected representative, countable-choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the source rows. The complete-metric service K and nested closed-ball spine B are the only rows from which a completion request may start.

Read the dyadic and finite-window data next. The shrink ledger D controls the requested tolerance, while S exposes the inspected center and containment windows and Q converts those windows into regular rational readback.

The modulus row U records the finite stage at which the requested tolerance is met. Only after U is displayed may the route reach the terminal Real seal E.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no quotient, selected representative, choice, open-cover, or ambient-completeness coordinate, the Real-seal route is exactly the displayed finite modulus route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorCompleteMetricModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorCompleteMetricModulus}^{\uparrow}$ packet is generated from CompleteMetric service, NestedClosedBall containment, DyadicRatCore shrink ledgers, StreamName finite windows, RegSeqRat readback, a RealUp seal, an explicit modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorCompleteMetricModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6720.1, Definition 6720.2, and Definition 6720.3 seed the finite Cantor complete-metric modulus route over complete-metric, nested-ball, dyadic, stream, regular-readback, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorCompleteMetricModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-ball shrink exactness, modulus exactness, finite-window readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorCompleteMetricModulusUp.

Chapter 6721

Real Archimedean Cutoff NameCert Construction

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArchimedeanCutoff}^{\uparrow}$ packet records the finite Archimedean cutoff row for displayed $\mathsf{Real}^{\uparrow}$ names. It reads a real name only through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and a $\mathsf{Nat}^{\uparrow}$ unary bound history, so the cutoff certificate stays inside displayed BEDC data and does not claim trichotomy, quotient reals, least upper bounds, choice, or ambient completeness.

Definition 6721.1 (RealArchimedeanCutoff carrier)**.** *A* $\mathsf{RealArchimedeanCutoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, S, G, D, U, L, H, T, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal whose displayed name is being cut off at a finite Archimedean scale.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened from R at the requested precisions.*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, so every inspected rational value has a regular sequence source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the inspected readback values with the displayed dyadic cutoff scale.*
5. *U is the* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{UnaryHistory}$ *bound row naming the finite number of window extensions used by the cutoff read.*
6. *L is the finite cutoff ledger pairing each requested precision with the displayed stream window, regular readback, dyadic tolerance, and unary bound row used for that precision.*
7. *H stores componentwise* hsame *transport for the Real seal, stream windows, regular readback, dyadic tolerance ledger, unary bound row, cutoff ledger, replay rows, provenance row, and local naming row.*
8. *T stores displayed* Cont *replay from a cutoff request through R, S, G, D, U, L, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RealArchimedeanCutoff}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{UnaryHistory}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanCutoff}^{\uparrow}}$ *row.*

The carrier contains only the displayed finite cutoff surface. It has no trichotomy row, no quotient equality of real names, no selected representative stream, no least-upper-bound object, no choice selector, and no completeness coordinate.

Theorem 6721.2 (RealArchimedeanCutoff NameCert obligations)**.** *For every accepted* $\mathsf{RealArchimedeanCutoff}^{\uparrow}$ *carrier* $C = (R, S, G, D, U, L, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanCutoff}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Real seal, stream windows, regular readback, dyadic tolerance ledger, unary bound row, cutoff ledger, componentwise transport, replay, provenance, and local naming rows;*
2. *classifier stability, requiring every classified cutoff read to transport* R, S, G, D, U, L *through the stored* hsame *and* Cont *routes;*
3. *finite bound readback, requiring the unary row* U *to expose the finite number of window extensions used before the cutoff ledger* L *is accepted;*
4. *Real seal nonescape, requiring every completion-facing consumer to read* $S, G, D, U, L, H, T, P, N$ *before it receives the* $\mathsf{Real}^{\uparrow}$ *seal row* R*;*
5. *ledger exactness, requiring each accepted cutoff comparison to be the displayed row in* L *that pairs the relevant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance entry, and* $\mathsf{Nat}^{\uparrow}$ *or* $\mathsf{UnaryHistory}$ *bound.*

Proof. Carrier admission is projection from the tuple in Definition 6721.1. The Real seal, finite stream windows, regular readback, dyadic tolerance ledger, unary bound row, cutoff ledger, transport table, replay table, provenance row, and local naming row are all explicit coordinates of the accepted packet.

Classifier stability is componentwise. The stored hsame and Cont rows transport the Real seal together with the stream windows and regular readback, and they transport the dyadic tolerance and unary bound rows beside the same cutoff ledger.

Finite bound readback then uses only U and L. The unary row names how many window extensions are opened, while the ledger records which displayed window, readback, and dyadic tolerance entry is used at each requested precision.

Finally inspect the carrier inventory. Since the packet has no coordinate for trichotomy, quotient equality of real names, selected representatives, least-upper-bound machinery, choice, or ambient completeness, every completion-facing read remains inside the displayed finite cutoff surface. □

Theorem 6721.3 (RealArchimedeanCutoff sibling dependency)**.** *Every accepted* $\mathsf{RealArchimedeanCutoff}^{\uparrow}$ *read factors through the property-level Archimedean bound route of chapter 3372 and the Cauchy-real Archimedean witness route of chapter 3378 before the local cutoff ledger* L *can be exported. The cutoff certificate is a bounded finite read of the same Archimedean witness surface through* $R, S, G, D, U, L, H, T, P, N$*; it is not a least-bound principle, trichotomy principle, quotient-real equality route, or ambient completeness claim.*

Proof. Project the carrier of Definition 6721.1. The rows R, S, G, D, U, L expose the Real seal, finite stream windows, regular readback, dyadic tolerance, unary bound history, and cutoff ledger.

The property-level sibling supplies the Archimedean bound route used to interpret the unary bound row. The Cauchy-real sibling supplies the witness route that makes the same bound readable

from regular Cauchy real-name data. Thus the cutoff read cannot export L until the property route and the Cauchy-real witness route have both been displayed.

The structural rows H, T, P, N preserve that dependency through componentwise hsame transport, displayed Cont replay, provenance, and local naming. Since no row in $R, S, G, D, U, L, H, T, P, N$ is a least-bound object, trichotomy classifier, quotient-real selector, or Real-completeness witness, the dependency remains a finite bounded read of the Archimedean witness surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanCutoff}^{\uparrow}}$.

Constructive story: A $\mathsf{RealArchimedeanCutoff}^{\uparrow}$ packet is generated from a RealUp seal, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a NatUp unary bound history, a cutoff ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArchimedeanCutoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6721.1 and Definition 6721.2 seed the finite Archimedean cutoff certificate for displayed Real names through StreamName, RegSeqRat, DyadicRatCore, Nat, and UnaryHistory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArchimedeanCutoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient equality of real names, selected representatives, least upper bounds, choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite bound readback, Real seal nonescape, and ledger exactness for BEDC.Derived.RealArchimedeanCutoffUp.

Chapter 6722

A Concrete Naming Certificate for $\mathsf{CauchyMerge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyMerge}^{\uparrow}$ packet records the finite merge route used when two regular Cauchy source names must be compared or combined through a common tail schedule. It sits beside the StreamName windows of chapter 126, the RegSeqRat readback route of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, the Real seal of chapter 38, and the reindexing discipline of chapter 1568. The packet gives the L10 regular-Cauchy merge route one finite naming surface; it does not form quotient streams, choose a limiting real, invoke countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6722.1 (Cauchy merge carrier)**.** *A* $\mathsf{CauchyMerge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R_0, R_1, W_0, W_1, D, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy source rows presented to the merge request.*
2. W_0 *and* W_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which the sources are read.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the common precision scale, the two source budgets, and the merged tail budget.*
4. S *is the finite merge-schedule row. It names the alternating or cofinal inspection order used to serve both source rows from the shared ledger.*
5. A *is the* $\mathsf{CauchyReindexing}^{\uparrow}$ *alignment row, showing that the schedule reads each source through a permitted tail reindexing route.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after* $R_0, R_1, W_0, W_1, D, S, A$ *have been replayed.*
7. H *records componentwise* hsame *transport for the source, window, ledger, schedule, alignment, seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a merge request through* $R_0, R_1, W_0, W_1, D, S, A, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{CauchyMerge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyMerge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, W_0, W_1, D, S, A, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected representative, a host limit, countable-choice data, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyMergeUp`

Theorem 6722.2 (Cauchy merge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMerge}^{\uparrow}$ *carrier*

$$M = (R_0, R_1, W_0, W_1, D, S, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMerge}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular-Cauchy source rows, two StreamName window rows, the shared dyadic tolerance ledger, the finite merge schedule, the Cauchy reindexing alignment row, the terminal Real seal, componentwise transport, continuation replay, provenance, and local naming. No local obligation may read a quotient stream, chosen representative, selected host limit, countable-choice selector, host real equality proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. The analytic rows are exactly $R_0, R_1, W_0, W_1, D, S, A, E$, and the structural rows are exactly H, C, P, N.

Carrier admission exposes the two regular-Cauchy sources only through their displayed StreamName windows. The shared DyadicRatCore ledger fixes the common tolerance scale, while the schedule row S supplies the finite order in which both source tails are inspected.

The alignment row A is the only route through which the schedule may reindex either source. Hence the terminal Real seal E is downstream of displayed regular-Cauchy readback, finite windows, a shared dyadic ledger, and permitted reindexing.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no additional row is present, quotient streams, selected representatives, host limits, countable choice, host equality, and ambient completeness cannot become local NameCert obligations. □

Theorem 6722.3 (Cauchy merge tail-equivalence route)**.** *Every accepted* $\mathsf{CauchyMerge}^{\uparrow}$ *packet turns the two regular-Cauchy sources into a Real-facing merge seal only through the finite route*

$$R_0, R_1 \longmapsto W_0, W_1 \longmapsto D \longmapsto S \longmapsto A \longmapsto E \longmapsto H, C, P, N.$$

If two merge consumers display the same two source rows, the same two window rows, the same dyadic ledger, the same merge schedule, and the same reindexing alignment row, then their tail-equivalence comparison is exhausted by this route and does not use quotient-stream equality or a selected limiting real.

Proof. Start from the two source rows R_0 and R_1. The window rows W_0 and W_1 specify the finite observations actually opened by the merge request, and the shared ledger D gives a single tolerance scale for both observations.

The schedule row S then determines how the two tails are interleaved or cofinally inspected. The alignment row A checks that each scheduled read is a Cauchy reindexing read, so the comparison cannot leave the displayed tail route.

Finally the Real seal E and the structural rows H, C, P, N repack the same finite route for downstream consumers. Equality of the displayed merge data therefore supplies tail-equivalence at the certificate boundary without adding quotient streams, chosen representatives, selected limits, countable choice, host real equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyMerge}^{\uparrow}$ merge read exposes two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, two $\mathsf{StreamName}^{\uparrow}$ window rows, a shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a finite merge schedule, a $\mathsf{CauchyReindexing}^{\uparrow}$ alignment row, a $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and a NameCert row before any Real-facing seal can be consumed.

Independence witness: regularcauchymerge, cauchy-reindexing, real-completion-tail-route: $\mathsf{CauchyMerge}^{\uparrow}$ is not bijective to its listed siblings because it jointly carries two regular-Cauchy sources, two finite window rows, one shared dyadic tolerance ledger, one merge schedule, one reindexing alignment row, one Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMerge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMerge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6722.1, Definition 6722.2, and Definition 6722.3 seed the finite L10 route for merging two regular-Cauchy names through shared windows, a common dyadic ledger, a schedule, reindexing alignment, and a Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMerge}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.CauchyMergeUp`

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected representatives, selected limits, countable choice, host real equality, ambient Real completeness, theorem checking, axiom-clean closure, or bridge checking.

Upgrade path: Theorem-checked closure requires checked carrier admission, classifier stability, hsame and Cont-compatible schedule transport, reindexing alignment exactness, and no-quotient/no-choice nonescape for BEDC.Derived.CauchyMergeUp.

Constructive story: A $\mathsf{CauchyMerge}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, two $\mathsf{StreamName}^{\uparrow}$ finite-window rows, one shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite merge schedule, a $\mathsf{CauchyReindexing}^{\uparrow}$ alignment row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6723

A Concrete Naming Certificate for $\mathsf{EventuallyConstantSequence}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on sequences, tails, and constructive real names, the $\mathsf{EventuallyConstantSequence}^{\uparrow}$ packet records the finite route by which an inspected sequence becomes constant after a displayed threshold. It sits beside the finite windows of chapter 126, the cofinite-tail interface of chapter 1564, the sequence-limit carrier of chapter 1413, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a tail certificate: it does not identify quotient streams, select a limiting value, invoke countable choice, assert host equality, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6723.1 (Eventually constant sequence carrier)**.** *An* $\mathsf{EventuallyConstantSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E_{\text{tail}} = (S, T, F, R, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *sequence-window row, exposing the inspected finite prefixes and tail windows of the source sequence.*
2. *T is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite-tail row, carrying the displayed threshold after which the sequence is read as constant.*
3. *F is the finite constant-value row, naming the value repeated on the admitted tail.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same tail windows and constant value.*
5. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *limit-handoff row, available only through the displayed tail threshold and constant readback.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the constant value and its sequence-limit handoff.*
7. *H records componentwise* hsame *transport for sequence windows, Frechet-tail threshold, constant-value row, regular readback, limit handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an eventual-constant request through $S, T, F, R, L, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{EventuallyConstantSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{EventuallyConstantSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, T, F, R, L, A, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected subsequence, an ambient convergence predicate, countable choice, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EventuallyConstantSequenceUp.EventuallyConstantSequenceTasteGate single carrier alignment`

Theorem 6723.2 (Eventually constant sequence NameCert obligations)**.** *For every accepted* $\mathsf{EventuallyConstantSequence}^{\uparrow}$ *carrier* $E_{\text{tail}} = (S, T, F, R, L, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EventuallyConstantSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed ten-row packet; StreamName window exposure through S; Frechet-tail threshold exposure through T; constant-value admission through F; regular-rational readback through R; RealSequenceLimit handoff through L; terminal Real seal through A; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.EventuallyConstantSequenceUp.EventuallyConstantSequenceNamecertObligations`

Proof. Project the carrier of Definition 6723.1. The analytic rows are exactly S, T, F, R, L, A, and the structural rows are exactly H, C, P, N.

The sequence is read first through finite windows. The row S exposes the inspected prefixes, while T supplies the cofinite-tail threshold that makes the eventuality claim visible as a finite Frechet-filter row.

The constant-value row F is then checked only on the displayed tail. The regular readback row R and limit handoff row L use that same tail threshold before the terminal Real seal A can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since quotient streams, selected subsequences, ambient convergence predicates, countable choice, host equality, and Real completeness are absent from these rows, none is a local NameCert obligation. □

Theorem 6723.3 (Eventually constant sequence tail-filter handoff)**.** *Every RealSequenceLimit-facing consumer of an accepted* $\mathsf{EventuallyConstantSequence}^{\uparrow}$ *packet reaches the limit handoff only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealSequenceLimit}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Frechet tail row supplies the threshold, the StreamName row supplies the inspected windows, and the RegSeqRat row supplies the regular readback of the constant value. The handoff exports no quotient stream, selected limit, countable-choice tail selector, host equality proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.EventuallyConstantSequenceUp.EventuallyConstantSequenceTailFilterHandoff`

Proof. Start from the NameCert obligation surface of Definition 6723.2. It forces every accepted read to expose the sequence-window row S and the Frechet-tail threshold T before a limit-facing consumer reaches L.

Next read the constant-value and regular-rational rows. The row F identifies the repeated tail value inside the displayed threshold, and R records the same value through the regular-rational route used by Real-facing consumers.

The limit handoff L and Real seal A are downstream of those rows. Componentwise transport, continuation replay, provenance, and local naming preserve the same finite route, so no quotient

stream, selected limit, countable-choice selector, host equality proof, or ambient completeness theorem is exported. □

Theorem 6723.4 (Eventually constant sequence Real seal nonescape)**.** *Let* $E_{\text{tail}} = (S, T, F, R, L, A, H, C, P, N)$ *be an accepted* $\mathsf{EventuallyConstantSequence}^{\uparrow}$ *carrier. Any consumer that reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A *must have already displayed* $S, T, F, R, L, H, C, P, N$*. Thus the Real seal is a finite tail certificate for an eventually constant sequence, not a hidden proof of arbitrary sequence convergence, quotient stream equality, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EventuallyConstantSequenceUp.EventuallyConstantSequenceRealSealNonescape`

Proof. Read the carrier projection. The terminal Real row A is listed only after the StreamName row, Frechet-tail threshold, constant-value row, regular readback, and RealSequenceLimit handoff.

Apply the tail-filter handoff theorem, Definition 6723.3. It places every limit-facing route through S, T, F, R, L before the Real seal can be consumed.

The structural rows H, C, P, N then transport, replay, source, and name the same displayed route. Because the packet has no coordinate for arbitrary convergence, quotient streams, selected limits, countable choice, host equality, or Real completeness, the seal cannot export any of those claims. □

Falsifiable prediction: Every accepted $\mathsf{EventuallyConstantSequence}^{\uparrow}$ consumer must display a $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{FrechetFilter}^{\uparrow}$ cofinite-tail threshold, a constant-value row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before the tail certificate can be consumed.

Independence witness: streamname, frechetfilter, real-sequence-limit, regseqrat, real: no listed sibling is bijective with $\mathsf{EventuallyConstantSequence}^{\uparrow}$, because this packet alone binds a displayed sequence-window row, a cofinite-tail threshold, a constant-value row, regular readback, RealSequenceLimit handoff, Real seal, transport, replay, provenance, and local naming into one finite eventually-constant certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{EventuallyConstantSequence}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EventuallyConstantSequence}^{\uparrow}$ packet is generated from StreamName sequence windows, a FrechetFilter cofinite-tail threshold, a constant-value row, RegSeqRat readback, a RealSequenceLimit handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EventuallyConstantSequence}^{\uparrow}}$, seed)

Scope closed: Definition 6723.1, Definition 6723.2, Definition 6723.3, and Definition 6723.4 seed the finite eventually-constant tail route over StreamName, FrechetFilter, RegSeqRat, RealSequenceLimit, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EventuallyConstantSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary sequence convergence, quotient stream equality, selected limits, countable choice, host real equality, RealUp completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-threshold exposure, constant-value exactness, RegSeqRat readback, RealSequenceLimit handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EventuallyConstantSequenceUp.

Chapter 6724

A Concrete Naming Certificate for SlowlyVaryingSequence$^{\uparrow}$

Following Karamata and Hardy, the SlowlyVaryingSequence$^{\uparrow}$ packet records the constructive finite-window surface of a positive sequence whose rescaled tails remain asymptotically unchanged. It sits beside the regular sequence readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the asymptotic comparison surface of chapter 3400. The packet names only displayed finite tail evidence; it does not assert a uniform Tauberian theorem, selected subsequence, quotient class of sequences, choice principle, or ambient completeness.

Definition 6724.1 (SlowlyVaryingSequence carrier)**.** *A* SlowlyVaryingSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, E, A, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *row whose finite windows expose the inspected tail entries of the sequence.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows, so every inspected value has a regular rational source.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to compare rescaled tail windows.*
4. *E is the terminal* Real$^{\uparrow}$ *seal for the displayed values read from S, R, D.*
5. *A is the* AsymptoticEquivalence$^{\uparrow}$ *row comparing the original tail and each displayed rescaled tail through finite windows.*
6. *W is the finite rescaling-window ledger. It records the inspected scale, the starting tail bound, the stream window, the regular readback, the dyadic tolerance, and the accepted asymptotic-equivalence row for that scale.*
7. *H stores componentwise* hsame *transport for stream windows, regular readback, dyadic tolerances, Real seals, asymptotic-equivalence rows, window ledgers, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay from a slowly-varying request through S, R, D, E, A, W, H.*
9. *P is the* Pkg *provenance over* SlowlyVaryingSequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, AsymptoticEquivalence$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$ *row.*
The carrier contains no quotient sequence equality, selected asymptotic class representative, countable-choice tail selector, global convergence claim, or ambient Real completeness coordinate.

Theorem 6724.2 (SlowlyVaryingSequence NameCert obligations)**.** *For every accepted* $\mathsf{SlowlyVaryingSequence}^{\uparrow}$ *carrier* $L = (S, R, D, E, A, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for those rows, finite-window exposure through* S*, regular rational readback through* R*, dyadic tolerance accounting through* D*, Real sealing through* E*, asymptotic-equivalence routing through* A*, rescaling-window exactness through* W*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6724.1. The stream, regular rational, dyadic tolerance, Real seal, asymptotic comparison, rescaling ledger, transport, replay, provenance, and local naming rows are all displayed coordinates of the finite packet.

The finite-window obligation is read through S, R, D, E. A consumer first opens a StreamName window, then reads the regular rational values and dyadic tolerances attached to that window, and only then receives the Real seal for the inspected values.

The slowly-varying obligation is read through A and W. Each rescaling request names its inspected scale and tail bound, and the window ledger records the exact asymptotic-equivalence row used for that request.

Componentwise transport and displayed replay preserve the same row order through H and C, while P and N name only this finite surface. Since the carrier has no quotient sequence equality, selected representative, choice selector, global convergence coordinate, or ambient completeness coordinate, the local NameCert obligations are exhausted by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SlowlyVaryingSequence}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, AsymptoticEquivalence rows, a finite rescaling-window ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$, seed)

Scope closed: Definition 6724.1 and Definition 6724.2 seed the finite slowly-varying sequence certificate over StreamName, RegSeqRat, DyadicRatCore, Real, and AsymptoticEquivalence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a Tauberian theorem, quotient sequence equality, selected representatives, countable-choice tail selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular rational readback, dyadic tolerance accounting, Real sealing, asymptotic-equivalence routing, and rescaling-window exactness for BEDC.Derived.SlowlyVaryingSequenceUp.

Chapter 6725

A Concrete Naming Certificate for $\mathsf{HaltingRefusalLedger}^{\uparrow}$

The $\mathsf{HaltingRefusalLedger}^{\uparrow}$ packet realizes the boundary of chapter 9369 as finite refusal data. It sits beside the fuel-indexed substrate bridge of chapter 960, the reduction fuel boundary of chapter 970, the step-indexed total host of chapter 964, the host trace surface of chapter 1050, and the closed-observation total-host packet of chapter 3706. The packet records how a local channel or bounded trace may terminate while a global halting classifier remains refused.

Falsifiable prediction: Every $\mathsf{HaltingRefusalLedger}^{\uparrow}$ read exposes finite fuel, trace, timeout-refusal, channel-termination, hsame, Cont, Pkg, and NameCert rows before refusing global halting.

Independence witness: fuel-indexed-substrate-bridge, reductionfuelboundary, step-indexed-total-host, hosttotaltrace, closed-observation-total-host: the ledger is not bijective to any listed sibling because it jointly stores local channel termination, finite fuel, trace readback, timeout refusal, and global-halting refusal in one packet.

Definition 6725.1 (HaltingRefusalLedger carrier)**.** *A* $\mathsf{HaltingRefusalLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, T, U, L, R, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite fuel row read through* $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ *and* $\mathsf{ReductionFuelBoundary}^{\uparrow}$*.*
2. *T is the finite trace row read through* $\mathsf{HostTotalTrace}^{\uparrow}$ *and the substrate-side bounded trace surface.*
3. *U is the timeout-refusal row recording that exhaustion of the displayed fuel is a local refusal, not a global nontermination proof.*
4. *L is the local channel-termination row supplied by the channel boundary of chapter 9369.*
5. *R is the total-host readback row routed through* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *and* $\mathsf{ClosedObservationTotalHost}^{\uparrow}$*.*
6. *B is the global-halting refusal row. It blocks any request to turn F, T, U, L, R into a total classifier for arbitrary certificate-level halting.*
7. *H stores componentwise* hsame *transport for fuel, trace, timeout refusal, channel termination, total-host readback, global-halting refusal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a halting-boundary request through F, T, U, L, R, B, H.*

9. *P is the* Pkg *provenance over* $\mathsf{HaltingRefusalLedger}^{\uparrow}$, $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$, $\mathsf{ReductionFuelBoundary}^{\uparrow}$, $\mathsf{StepIndexedTotalHost}^{\uparrow}$, $\mathsf{HostTotalTrace}^{\uparrow}$, $\mathsf{ClosedObservationTotalHost}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingRefusalLedger}^{\uparrow}}$ *row.*

The carrier contains no global halting oracle, no selected nontermination witness, no total classifier for arbitrary certificate continuations, and no collapse of local channel termination into certificate-level halting.

Theorem 6725.2 (HaltingRefusalLedger NameCert obligations)**.** *For every accepted* $\mathsf{HaltingRefusalLedger}^{\uparrow}$ *carrier $G = (F, T, U, L, R, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HaltingRefusalLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite fuel, trace, timeout-refusal, channel-termination, total-host readback, global-halting refusal, transport, replay, provenance, and naming rows; fuel-trace exposure through F, T; timeout exactness through U; channel-local termination exposure through L; total-host readback through R; and global-halting refusal through B.*

Proof. Project the carrier of Definition 6725.1. The rows $F, T, U, L, R, B, H, C, P, N$ are the complete finite packet exposed to a halting-boundary consumer.

The local positive rows are F, T, L, R. Fuel and trace are read through the fuel-indexed substrate and host-trace siblings, while L records only channel-local event termination and R records only total-host readback under displayed observations.

The refusal rows are U and B. A timeout row refuses to treat fuel exhaustion as a completed unbounded computation, and the global-halting row refuses to turn local termination or finite trace evidence into a total classifier for arbitrary certificate continuations.

Transport, replay, provenance, and naming preserve this separation through H, C, P, N. Since no carrier coordinate stores a global halting oracle, selected nontermination witness, or total certificate-level classifier, the NameCert obligations are exhausted by the displayed ledger rows. □

The Lean TasteGate surface for $\mathsf{HaltingRefusalLedger}^{\uparrow}$ must provide explicit Nontrivial $\mathsf{HaltingRefusalLedger}^{\uparrow}$ and FieldFaithful $\mathsf{HaltingRefusalLedger}^{\uparrow}$ instances preserving finite fuel, trace, timeout-refusal, channel-termination, total-host readback, global-halting refusal, componentwise transport, displayed replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingRefusalLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HaltingRefusalLedger}^{\uparrow}$ packet is generated from finite fuel, finite trace, timeout-refusal, channel-termination, total-host readback, global-halting refusal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HaltingRefusalLedger}^{\uparrow}}$, seed)

Scope closed: Definition 6725.1 and Definition 6725.2 seed the finite refusal ledger separating local channel termination and bounded trace evidence from global certificate-level halting.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HaltingRefusalLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a global halting classifier, selected nontermination witness, total classifier for arbitrary certificate continuations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fuel-trace exposure, timeout exactness, channel-local termination exposure, total-host readback, and global-halting refusal for BEDC.Derived.HaltingRefusalLedgerUp.

Chapter 6726

A Concrete Naming Certificate for $\mathsf{CauchyNetTailFilter}^{\uparrow}$

Following Moore and Smith, Kelley, and Bishop–Bridges on nets, filters, and constructive completion, the $\mathsf{CauchyNetTailFilter}^{\uparrow}$ packet records the finite handoff from displayed Cauchy-net evidence to displayed tail-filter evidence. It sits beside the completion route of chapter 3794, the finite filter-base surface of chapter 1601, the cofinite sequence-tail specialization of chapter 1564, the finite schedule interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a BEDC-native bridge from Cauchy-net windows to tail-filter windows; it does not choose a limit, quotient nets, assume countable choice, decide trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6726.1 (CauchyNetTailFilter carrier)**.** *A* $\mathsf{CauchyNetTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (J, W, K, B, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{CauchyNet}^{\uparrow}$ *index-window row, naming the directed indices and inspected pairs used by the net-side Cauchy request.*
2. *W is the finite tail-threshold row, recording the displayed index after which the inspected net windows are admitted.*
3. *K is the Cauchy-net tolerance ledger attached to J and W, so every tail request keeps the original Cauchy evidence visible.*
4. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *meet row generated from finite tails and finite intersections of tail events.*
5. *F is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite handoff row used when the directed tail is narrowed to a sequence-tail read.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the finite observation windows passed to downstream Real or completion consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same schedule and tolerance ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, readable only after J, W, K, B, F, S, R have been displayed.*
9. *H records componentwise* hsame *transport for the Cauchy-net, threshold, filter-base, cofinite handoff, schedule, readback, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tail-filter request through $J, W, K, B, F, S, R, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyNetTailFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetTailFilter}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for selected limits, quotient nets, quotient filters, countable-choice schedules, trichotomy, host equality of tails, or ambient Real completeness.

Theorem 6726.2 (CauchyNetTailFilter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetTailFilter}^{\uparrow}$ *carrier $T = (J, W, K, B, F, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyNetTailFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy-net index windows J, finite tail threshold W, Cauchy tolerance ledger K,* $\mathsf{FilterBase}^{\uparrow}$ *meet row B,* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite handoff F,* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming N.*

Proof. Project the carrier of Definition 6726.1. The net-side evidence is exactly J, W, K: displayed index windows, a finite threshold, and the Cauchy tolerance ledger attached to those windows.

The filter-facing handoff is then B, F. The $\mathsf{FilterBase}^{\uparrow}$ row records finite meets of tail events, while the $\mathsf{FrechetFilter}^{\uparrow}$ row records the cofinite sequence-tail specialization available to sequential consumers.

Downstream Real and completion reads must pass through S, R, E. The $\mathsf{StreamName}^{\uparrow}$ schedule exposes only finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row records regular rational readback for those windows, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after the schedule and readback have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since there is no additional coordinate, the local NameCert surface cannot export selected limits, quotient nets, countable choice, trichotomy, or Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyNetTailFilter}^{\uparrow}$ completion-facing read exposes CauchyNet index windows, a finite tail threshold, a Cauchy tolerance ledger, a FilterBase meet row, a FrechetFilter cofinite handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before it can serve a completion or Real consumer.

Independence witness: cauchynetcompletion, filterbase, frechetfilter, streamname, regseqrat, real: $\mathsf{CauchyNetTailFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries net index windows, tail thresholds, filter-base meets, cofinite sequence-tail handoff, finite schedules, regular readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetTailFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNetTailFilter}^{\uparrow}$ packet is generated from CauchyNet index windows, finite tail thresholds, a Cauchy tolerance ledger, a FilterBase meet row, a FrechetFilter cofinite handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyNetTailFilter}^{\uparrow}}$, seed)

Scope closed: Definition 6726.1 and Definition 6726.2 seed the finite Cauchy-net tail-filter bridge over CauchyNet, CauchyNetCompletion, FilterBase, FrechetFilter, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyNetTailFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient nets, quotient filters, countable choice, trichotomy, host equality of tails, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-net index-window exposure, finite tail-threshold exactness, FilterBase meet generation, FrechetFilter cofinite handoff, StreamName scheduling, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNetTailFilterUp.

Chapter 6727

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyRealizer}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy sequences, the $\mathsf{BishopRegularCauchyRealizer}^{\uparrow}$ packet records the finite route by which dyadic tolerances, stream observations, regular rational readback, an explicit Cauchy modulus, and the terminal real-name seal are assembled into one choice-free realizer. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational sequence route of chapter 301, the Cauchy-modulus carrier of chapter 292, and the displayed $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a realizer surface for regular Cauchy real names; it is not a quotient of streams, not a selected host limit, not countable choice, and not ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 6727.1 (BishopRegularCauchyRealizer carrier)**.** *A* $\mathsf{BishopRegularCauchyRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including dyadic scale, comparison, and refinement rows used by the regularity bound.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, carrying the finite windows at which the rational name is inspected.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the rational values attached to those windows.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that binds a precision request to a displayed window budget before any Real-facing seal is read.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row receiving the regular Cauchy name only after D, S, Q, M have been displayed.*
6. *H records componentwise* hsame *transport for dyadic tolerance, stream window, rational readback, modulus, endpoint seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay along the same route.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyRealizer}^{\uparrow}}$ *row.*

The classifier on such packets is componentwise transport through H together with replay through C. No coordinate exports a quotient stream, a selected limit, an unrestricted choice schedule, host real equality, or a detached completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 6727.2 (BishopRegularCauchyRealizer NameCert obligations)**.** *For every accepted* BishopRegularCauchy *carrier $R = (D, S, Q, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dyadic tolerance ledger, stream-window row, regular-rational readback row, Cauchy-modulus row, Real seal, componentwise transport, replay, provenance, and local naming. Every Real-facing read of the packet must expose the route*

$$D, S, Q, M \longmapsto E \longmapsto H, C, P, N.$$

It cannot replace the route by quotient equality of streams, a host-selected limit, countable choice, or an ambient completeness theorem.

Proof. Project the carrier. The accepted packet displays only the rows $D, S, Q, M, E, H, C, P, N$, and the local naming row has no coordinate outside that list.

The dyadic tolerance row D supplies the precision scale, the stream row S supplies the finite inspection windows, and the regular rational row Q supplies the rational values at those windows. The modulus row M is the only admitted route from a precision request to a window budget.

The endpoint seal E is downstream from those four rows. Componentwise transport through H, displayed replay through C, and provenance through P preserve the same route before the local NameCert row N names it.

The excluded objects are absent from the carrier. Therefore the NameCert obligations expose the finite regular Cauchy realizer route and do not create quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6727.3 (BishopRegularCauchyRealizer Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{BishopRegularCauchyRealizer}^{\uparrow}$ *carrier factors through the displayed dyadic, stream, regular-rational, and Cauchy-modulus rows*

$$D, S, Q, M, E, H, C, P, N.$$

In particular, the seal cannot be read as a selected limit of a host sequence, a quotient class of Cauchy streams, a countable-choice construction, or a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Open the exported seal row E. By Definition 6727.1, that row is part of the same finite packet as D, S, Q, M, H, C, P, N and has no independent source.

Apply Definition 6727.2. It forces each Real-facing read to pass first through dyadic tolerance, stream-window, regular-rational, and Cauchy-modulus rows, and then through transport, replay, provenance, and local naming.

Since the carrier contains no quotient stream coordinate, no selected host limit, no countable-choice schedule, and no ambient completeness row, the exported seal remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyRealizer}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, finite stream-window observations, regular rational readback, an explicit Cauchy modulus, a Real endpoint seal, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6727.1, Definition 6727.2, and Definition 6727.3 seed the choice-free regular Cauchy realizer route over dyadic, stream, rational, modulus, and Real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopRegularCauchyRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient real equality, selected limits, countable choice, host real equality, or ambient completeness for Real$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, modulus-window handoff, Real-seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyRealizerUp.

Chapter 6728

A Concrete Naming Certificate for $\mathsf{RiemannStieltjesIntegral}^{\uparrow}$

Following Riemann and Stieltjes, and the Bishop-style use of tagged partitions, the $\mathsf{RiemannStieltjesIntegral}^{\uparrow}$ packet records a finite integral carrier for a regulated real integrand against a bounded-variation integrator. It presents the integral value only through displayed partition sums, variation control, regular-rational readback, finite stream windows, and the terminal real-name seal; it does not import a measure space, a selected limiting partition, a quotient of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

This chapter is a bounded-variation consumer over the Real and Completion route. It reads the tagged-partition surface of chapter 1526, the variation ledger of chapter 398, the regulated real function surface of chapter 1554, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The existing $\mathsf{RiemannStieltjes}^{\uparrow}$ chapter supplies a neighboring handoff surface, while this packet isolates the integral carrier consumed by later bounded-variation arguments.

Definition 6728.1 (Riemann–Stieltjes integral carrier)**.** *A* $\mathsf{RiemannStieltjesIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, G, V, T, S, Q, W, R, E, H, C, P, N)$$

with displayed rows: F *is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *integrand row;* G *is the regulated real integrator row;* V *is the* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger for* G*;* T *is the tagged-partition row inherited from the* $\mathsf{RiemannIntegral}^{\uparrow}$ *surface;* S *is the finite Stieltjes-sum row;* Q *is the mesh and refinement request row;* W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row;* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the accepted sums;* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value; and* H, C, P, N *are the componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegral}^{\uparrow}}$ *rows.*

Theorem 6728.2 (Riemann–Stieltjes integral NameCert obligations)**.** *For every accepted carrier* $I = (F, G, V, T, S, Q, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission; regulated real integrand and integrator exposure through* F, G*; bounded-variation control through* V*; tagged-partition and Stieltjes-sum exposure through* T, S, Q*; finite-window and regular-rational readback through* W, R*; terminal real sealing through* E*; componentwise transport, replay, provenance, and local naming through* H, C, P, N*; and non-escape from measure spaces, quotient functions, selected partition limits, countable choice, and ambient completion.*

Proof. Project the displayed carrier. The rows F, G, V expose the regulated real function data and the bounded-variation control before any Stieltjes sum can be inspected.

The tagged-partition rows T, S, Q then describe only finite sums and refinement requests. They inherit the finite partition discipline from the Riemann integral surface but attach the integrator step differences through the displayed variation ledger rather than through a measure.

The completion-facing read is forced through W, R, E. The finite windows are read as $\mathsf{StreamName}^{\uparrow}$ rows, the accepted sums as $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and the endpoint value only after the $\mathsf{Real}^{\uparrow}$ seal is displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a measure space, quotient function, selected limiting partition, countable-choice schedule, or ambient completion theorem, the listed obligations exhaust the NameCert surface. □

Theorem 6728.3 (Riemann–Stieltjes integral bounded-variation handoff)**.** *Every downstream bounded-variation consumer of an accepted* $\mathsf{RiemannStieltjesIntegral}^{\uparrow}$ *packet must factor its integral read through*

$$F, G, V, T, S, Q, W, R, E, H, C, P, N.$$

In particular, the consumer receives the $\mathsf{BoundedVariation}^{\uparrow}$ *ledger for the integrator before it may read the Stieltjes-sum row, and it receives the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it may treat the integral as a Real-facing value.*

Proof. Start with a bounded-variation-facing consumer. The packet has no integral-value coordinate except the terminal seal E, and E is downstream of the finite-window and regular-rational readback rows W, R.

The Stieltjes-sum row S is also downstream of V. Thus the consumer cannot inspect integrator step differences without first displaying the bounded-variation ledger that controls those differences.

Now apply the Real and Completion route discipline. The finite window W and regular readback R are the only rows that connect finite sums to the terminal $\mathsf{Real}^{\uparrow}$ seal, so the consumer must pass through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before reading E.

Transport, replay, provenance, and local naming preserve the same route. Hence any accepted bounded-variation handoff factors through the displayed rows and cannot replace them by a measure-completion, quotient-function, selected-limit, or ambient-completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannStieltjesIntegral}^{\uparrow}$ packet is generated from regulated real function rows, a bounded-variation ledger, tagged finite partitions, Stieltjes sums, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6728.1, Definition 6728.2, and Definition 6728.3 seed the finite Riemann–Stieltjes integral packet over bounded-variation, tagged-partition, finite-window, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic integration, Lebesgue–Stieltjes completion, quotient function equality, selected limiting partitions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition stability, bounded-variation control, finite-window readback, regular-rational exactness, terminal real sealing, transport, replay, provenance, and local naming for the same packet.

Chapter 6729

A Concrete Naming Certificate for $\mathsf{PrecompactFunctionFamily}^{\uparrow}$

Following Arzela–Ascoli and Bourbaki compact-open function-family theory, the $\mathsf{PrecompactFunctionFamily}^{\uparrow}$ packet records a family-level precompactness carrier as finite BEDC data. It sits between the precompact uniform route of chapter 4068, the compact-open topology surface of chapter 4138, the compact-open function-space carrier of chapter 2221, the bounded-function-family ledger of chapter 3522, and the finite-net equicontinuity selector of chapter 3358. The packet names the reusable family precompactness surface; it does not assert host compactness of a function space, choose a convergent subnet, quotient functions, or import open-cover compactness.

Definition 6729.1 (Precompact function family carrier)**.** *A* $\mathsf{PrecompactFunctionFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, T, U, B, E, N, W, H, C, P, L)$$

with the following displayed rows:

1. *X is the compact-domain source row read through the compact-metric or compact-window surface used by compact-open consumers.*
2. *Y is the target metric or uniform row whose bounded images are inspected by the family ledger.*
3. *T is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *row recording the finite compact-window requests imposed on the family.*
4. *U is the* $\mathsf{PrecompactUniform}^{\uparrow}$ *handoff row for the induced uniform family surface.*
5. *B is the* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *row, exposing pointwise and sup-bound ledgers only as displayed finite data.*
6. *E is the equicontinuity row consumed through the finite-net selector and uniform-modulus route.*
7. *N is the finite net row that records the inspected family approximants at the requested tolerances.*
8. *W is the compact-open window ledger binding T, U, B, E, N to the same source and target rows.*
9. *H records componentwise* hsame *transport for X, Y, T, U, B, E, N, W and the local certificate rows.*
10. *C records displayed* Cont *replay from compact-open family requests through precompact uniform, bounded-family, equicontinuity, finite-net, and window rows.*

11. *P is the* Pkg *provenance over* $\mathsf{PrecompactFunctionFamily}^{\uparrow}$, $\mathsf{PrecompactUniform}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{BoundedFunctionFamily}^{\uparrow}$, $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *L is the local* $\mathsf{NameCert}_{\mathsf{PrecompactFunctionFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host function-space topology, arbitrary open cover, selected subnet, quotient equality of maps, countable choice, or ambient compactness theorem.

Theorem 6729.2 (Precompact function family NameCert obligations)**.** *For every accepted* PrecompactFunctionFam *carrier*

$$F = (X, Y, T, U, B, E, N, W, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{PrecompactFunctionFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the source row X, target row Y, compact-open topology row T, precompact uniform handoff U, bounded-family row B, equicontinuity row E, finite-net row N, compact-open window ledger W, componentwise transport H, displayed replay C, provenance P, and local naming L.*

Proof. Project the carrier of Definition 6729.1. The compact-open read starts from X, Y, T, so every family request is already tied to displayed source, target, and compact-window rows.

The precompactness data are then forced through U, B, E, N, W. The precompact uniform handoff supplies the total-bounded family surface, the bounded-family and equicontinuity rows provide the Arzela–Ascoli side conditions, and the finite-net row records the inspected approximants at the requested tolerances.

Finally attach H, C, P, L. These rows transport, replay, source, and name the same packet, so the local certificate has no remaining coordinate from which to extract host function-space compactness, selected subnets, quotient maps, countable choice, or an open-cover compactness principle. □

Theorem 6729.3 (Precompact function family finite-net handoff)**.** *Every finite-net consumer of an accepted* $\mathsf{PrecompactFunctionFamily}^{\uparrow}$ *carrier must read the route*

$$\mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{PrecompactUniform}^{\uparrow} \longrightarrow \mathsf{BoundedFunctionFamily}^{\uparrow} \longrightarrow \mathsf{EquicontinuityFiniteNetSelector}$$

through the carrier rows T, U, B, E, N before it reaches W, H, C, P, L. The handoff is a displayed compact-open family net surface, not a selected convergent subnet, host compactness theorem, quotient equality of functions, or ambient open-cover argument.

Proof. Begin with Definition 6729.2. It places the compact-open row T and the precompact uniform row U inside the same local certificate before any finite-net consumer can read N.

Next read the family hypotheses through B and E. The bounded-family row and equicontinuity selector provide the displayed Arzela–Ascoli inputs that make the finite net meaningful over the fixed source and target rows.

The remaining rows W, H, C, P, L preserve the handoff. Since the packet has no host topology, selected subnet, quotient-function equality, countable-choice schedule, or open-cover compactness coordinate, the finite-net consumer receives exactly the displayed family-level precompactness surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactFunctionFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{PrecompactFunctionFamily}^{\uparrow}$ packet is generated from compact-domain and target rows, compact-open topology windows, a precompact uniform handoff, bounded-family and equicontinuity rows, finite-net approximants, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PrecompactFunctionFamily}\uparrow}$, seed)

Scope closed: Definition 6729.1, Definition 6729.2, and Definition 6729.3 seed the finite family-level precompactness carrier for compact-open and Arzela–Ascoli consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{PrecompactFunctionFamily}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host function-space compactness, selected subnets, quotient equality of maps, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exactness, precompact uniform handoff, bounded-family and equicontinuity rows, finite-net readback, transport, replay, provenance, and local naming for BEDC.Derived.PrecompactFunctionFamilyUp.

Chapter 6730

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ packet records located apartness between regular Cauchy real names as finite NameCert data before any completed real object is consulted. It reads the regular rational source rows of chapter 301, the positive dyadic separation witnesses of chapter 291, the finite window schedules of chapter 126, the terminal real-name seal of chapter 38, and the separated completion boundary of chapter 1567. The carrier names a constructive apartness route for later Real and Completion consumers; it does not assert trichotomy, Dedekind cuts, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6730.1 (Regular-Cauchy located-apartness carrier)**.** *A* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows naming the regular Cauchy rational streams under comparison.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the two sources at the same requested precisions.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive separation witness row; it displays a dyadic radius and a finite window at which the two regular Cauchy reads are separated by that positive tolerance.*
4. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from A, B, W, D, recording the inspected rational values and their dyadic separation ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular sources, shared windows, positive dyadic witness, and readback row have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, the shared window, the positive dyadic witness, the readback row, the real seal, the replay row, the provenance row, and the local name row.*
7. *C records displayed* Cont *replay from the two regular Cauchy source rows through the common finite window, the positive dyadic witness, the regular readback, and then the real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealApartnessCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedApartness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, D, R, E, H, C, P, N$. It has no coordinate for real trichotomy, Dedekind cuts, quotient equality of real names, selected limits, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6730.2 (Regular-Cauchy located-apartness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *carrier $L = (A, B, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedApa}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive separation witness, paired readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*

2. *located positivity, requiring the witness row D to display a positive dyadic separation before any apartness consumer can inspect the seal row E;*

3. *regular readback exactness, requiring the values compared by D to be the values exposed by A, B through the shared window W and repacked in R;*

4. *componentwise stability, requiring H and C to transport and replay only the displayed rows A, B, W, D, R, E, P, N;*

5. *ledger exhaustion, requiring the packet to export no trichotomy decision, Dedekind cut, quotient real equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No local obligation is stored outside these rows.

Proof. Carrier admission is obtained by projecting the tuple of Definition 6730.1. The displayed rows are precisely the two regular Cauchy sources, common finite window, dyadic positivity witness, paired readback, Real seal, transport, replay, provenance, and local name.

Located positivity is the role of D. The row stores a positive dyadic separation witness together with the finite window on which the two source reads are inspected, so apartness is exposed as positive data rather than as a trichotomy branch.

Regular readback exactness follows from the order of A, B, W, D, R. The readback row R is downstream of the two sources, the common window, and the dyadic witness; hence it cannot compare host representatives or quotient classes.

Componentwise stability and ledger exhaustion are supplied by H, C, P, N. These rows transport, replay, source, and name the same finite packet, and no row remains that could introduce Dedekind cuts, quotient real equality, selected limits, countable choice, or ambient completeness. □

Theorem 6730.3 (Regular-Cauchy located-apartness Real-seal boundary)**.** *Let $L = (A, B, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *packet. Every Real-facing apartness consumer of L factors through the finite route*

$$A, B, W, D, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the $\mathsf{Real}^{\uparrow}$ *seal exposes only a regular-Cauchy located-apartness read through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealApartnessCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. It cannot be reinterpreted as trichotomy, a Dedekind cut, quotient real equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted Real-facing apartness read. Carrier admission gives the two regular Cauchy source rows A, B and the shared finite window row W before any separation data can be inspected.

Next read the positive witness and regular readback. The dyadic row D supplies the positive separation, while R records that the compared values are exactly the regular rational reads exposed through A, B, W.

Finally expose the seal row and structural rows. Since E is downstream of A, B, W, D, R, and since H, C, P, N transport, replay, source, and name only that packet, every consumer remains inside the displayed finite route and cannot escape to trichotomy, Dedekind cuts, quotient real equality, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ consumer that reads apartness without two $\mathsf{RegSeqRat}^{\uparrow}$ sources, a shared $\mathsf{StreamName}^{\uparrow}$ finite window, a positive $\mathsf{DyadicRatCore}^{\uparrow}$ separation witness, paired readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row refutes the carrier surface.

Independence witness: regseqrat, dyadicratcore, streamname, real, realapartnesscompletion: no listed sibling alone carries the two regular Cauchy source rows, shared finite window, positive dyadic separation witness, paired readback, terminal Real seal, transport, replay, provenance, and local naming rows of $L = (A, B, W, D, R, E, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName finite window, a positive DyadicRatCore separation witness, paired RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6730.1, Definition 6730.2, and Definition 6730.3 seed the finite located-apartness carrier over regular Cauchy names, dyadic positive witnesses, shared stream windows, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, Dedekind cuts, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic witness exactness, RegSeqRat readback exposure, StreamName window sharing, Real-seal nonescape, structural transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedApartnessUp.

Chapter 6731

A Concrete Naming Certificate for $\mathsf{OpenCover}^{\uparrow}$

Following Bourbaki and Bishop–Bridges, the $\mathsf{OpenCover}^{\uparrow}$ packet records the finite interface by which an open cover is displayed before a compactness, finite-subcover, Lebesgue-number, or compact-uniform-continuity consumer may read it. It sits between the open-set rows of $\mathsf{Topology}^{\uparrow}$, the radius and ball rows of $\mathsf{Met}^{\uparrow}$, the compact source row of $\mathsf{Compact}^{\uparrow}$, the finite-subcover extraction route of $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$, and the compact-uniform-continuity construction of chapter 9237. The packet names an indexed family only through finite BHist rows; it does not import host powersets, arbitrary choice of subcovers, or open-cover compactness as a primitive.

The finite cover ledger is consumed by chapter 129 and the finite extraction row of chapter 3503 before compact-uniform-continuity arguments inspect it.

The finite index, open-set, membership, refinement, and finite-subcover rows are read through chapter 88, chapter 57, chapter 60, and chapter 3503 before compact-uniform-continuity consumers inspect them.

Definition 6731.1 (Open-cover carrier)**.** *An* $\mathsf{OpenCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, U, M, R, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-window row, naming the cover indices visible to the current consumer request.*
2. *U is the open-set predicate row over* $\mathsf{Topology}^{\uparrow}$*, recording for each displayed index the local open-set certificate used by the cover.*
3. *M is the membership and coverage row: each covered point read is tied to one displayed index of I and one open row of U.*
4. *R is the refinement row, comparing two displayed cover windows only by finite index maps and inclusion witnesses between their open-set rows.*
5. *F is the finite-subcover ledger, exposed only when a* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *consumer has supplied the compact source and the finite extraction route.*
6. *H records componentwise* hsame *transport for index, open-set, membership, coverage, refinement, finite-subcover, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a cover request through I, U, M, R, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{OpenCover}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OpenCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for a host powerset of all opens, an arbitrary subcover selector, a quotient of covers, countable choice, or open-cover compactness as an ambient principle.

Theorem 6731.2 (Open-cover finite ledger obligations)**.** *For every accepted* $\mathsf{OpenCover}^{\uparrow}$ *carrier $O = (I, U, M, R, F, H, C, P, N)$, the local cover ledger visible to any compactness-facing consumer is exhausted by finite index admission, open-set predicate admission, membership and coverage exactness, refinement exactness, finite-subcover handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *consumer may inspect F only after I, U, M, R have been displayed, and no consumer may replace those rows by a host powerset, arbitrary subcover choice, or primitive open-cover compactness theorem.*

Proof. Project the carrier of Definition 6731.1. The rows I, U, M, R expose the only index, open-set, membership, coverage, and refinement data available to the packet.

The finite-subcover row F is downstream of those rows. It is read only as a handoff to the compact or finite-subcover-criterion consumer, so it cannot select a hidden subfamily or quantify over an ambient powerset of opens.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores a host powerset, arbitrary selector, quotient cover equality, countable-choice schedule, or primitive open-cover compactness principle, none of those objects can become a local ledger obligation. □

Theorem 6731.3 (Open-cover NameCert surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{OpenCover}^{\uparrow}}$ *surface consists exactly of the carrier rows in Definition 6731.1: finite index windows,* $\mathsf{Topology}^{\uparrow}$ *open-set predicates, membership and coverage witnesses, refinement rows, a finite-subcover ledger for* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *consumers, componentwise transport, displayed replay, provenance, and local naming. When the same cover is used in a metric argument, the* $\mathsf{Met}^{\uparrow}$ *contribution is read only through the displayed ball or radius rows named by U, M, R, not through a separate open-cover compactness principle.*

Proof. The NameCert surface first reads I, U, M, R. These rows determine the displayed cover window, the open predicates, and the finite coverage and refinement evidence.

The compact handoff is then represented by F. It is a finite ledger row over the same displayed cover window, so the compact-uniform-continuity route of chapter 9237 receives a finite cover interface rather than an ambient cover-selection principle.

The remaining rows H, C, P, N are structural. They preserve the same finite route under transport, replay it for downstream consumers, record its provenance, and name it locally. Thus the surface is exactly the stated finite open-cover interface. □

Theorem 6731.4 (Open-cover constructive boundary)**.** *Every* $\mathsf{OpenCover}^{\uparrow}$ *read used by compact, finite-subcover, Lebesgue-number, or compact-uniform-continuity arguments must pass through the finite route*

$$I, U, M, R, F, H, C, P, N.$$

The route may cite Topology$^{\uparrow}$ *for open-set predicates,* Met$^{\uparrow}$ *for ball and radius witnesses,* Compact$^{\uparrow}$ *for the compact source, and* FiniteSubcoverCriterion$^{\uparrow}$ *for finite extraction, but it exports no host powerset of opens, arbitrary choice of subcovers, quotient cover equality, countable-choice schedule, or primitive open-cover compactness theorem.*

Proof. Read the cover rows in order. The index window I bounds the visible family, U supplies the open predicates, and M, R account for coverage and refinement using only displayed rows.

The finite-subcover ledger F is available only after the compact source or finite-subcover criterion has supplied its own finite extraction data. This keeps compactness-facing arguments on the same finite ledger used by the cover.

Transport, replay, provenance, and local naming do not add new cover data. Therefore every downstream consumer receives the finite route named in the statement and none of the excluded host or choice principles. □

Closure status for NameCert$_{\text{OpenCover}^{\uparrow}}$.

Constructive story: An OpenCover$^{\uparrow}$ packet is generated from finite index windows, Topology open-set predicate rows, membership and coverage rows, refinement rows, finite-subcover ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{OpenCover}^{\uparrow}}$, seed)

Scope closed: Definition 6731.1, Definition 6731.2, Definition 6731.3, and Definition 6731.4 seed the finite open-cover interface for compactness and compact-uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{OpenCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host powersets of opens, arbitrary subcover choice, quotient cover equality, countable choice, primitive open-cover compactness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, open-set predicate exactness, membership and coverage exactness, refinement stability, finite-subcover handoff, transport, replay, provenance, and local naming for BEDC.Derived.OpenCoverUp.

Chapter 6732

A Concrete Naming Certificate for $\mathsf{CompleteProductMetric}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive metric completion and product metrics, the $\mathsf{CompleteProductMetric}^{\uparrow}$ packet records the finite BEDC surface where a binary product metric is already displayed and the completion-facing Cauchy windows are accepted as complete-product data. It sits after the product-distance packet of chapter 424, the metric-completion carrier of chapter 1506, the separated-completion boundary of chapter 1612, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Banach, Polish, compact, and completion routes a complete product-metric certificate without adding host product metrics, quotient completion equality, selected product limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6732.1 (Complete product-metric carrier)**.** *A* $\mathsf{CompleteProductMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, \Pi, \Delta, M, S, W_X, W_Y, R_X, R_Y, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted component metric-completion source rows.*
2. *Π is the accepted* $\mathsf{ProductMetric}^{\uparrow}$ *product-distance row, including the displayed product carrier, projection reads, and product-distance ledger.*
3. *Δ is the completion-facing metric row read from Π rather than from a host product metric.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route that accepts paired Cauchy windows as completion-facing data.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison boundary for identifying completed product reads only through displayed separated data.*
6. *W_X and W_Y are the paired* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected on the component Cauchy names.*
7. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed product-distance estimate.*
9. *H records componentwise* hsame *transport for the component rows, product metric, completion route, separated boundary, stream windows, regular readbacks, Real seal, continuation, provenance, and local naming.*

10. *C records displayed* Cont *replay from a complete-product request through* $X, Y, \Pi, \Delta, M, S, W_X, W_Y, R_X, R_Y, E$.
11. *P is the* Pkg *provenance over* $\mathsf{CompleteProductMetric}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteProductMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host product metrics, quotient completion classes, selected representatives, countable choice, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6732.2 (Complete product-metric NameCert obligations)**.** *For every accepted* $\mathsf{CompleteProductMetric}^{\uparrow}$ *carrier* $Q = (X, Y, \Pi, \Delta, M, S, W_X, W_Y, R_X, R_Y, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompleteProductMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the component completion sources, the* $\mathsf{ProductMetric}^{\uparrow}$ *product-distance row, the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion route, the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison boundary, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of host product metrics, quotient completion equality, selected product limits, countable choice, trichotomy, detached real carriers, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 6732.1. The displayed packet exposes exactly the two component source rows, the product-distance row Π, the completion route M, the separated boundary S, paired windows, regular readbacks, the Real seal, and the structural rows.

The product metric is read before the completion route. Thus Δ and M can consume only the $\mathsf{ProductMetric}^{\uparrow}$ ledger carried by Π, together with the finite windows and regular readbacks, and cannot import a host product metric.

The separated-completion boundary is then forced through S. It compares completed product reads only after the paired Cauchy windows and product-distance estimate have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, trichotomy, detached real carriers, ambient completeness, or bridge checking, the listed obligations exhaust the local certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteProductMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteProductMetric}^{\uparrow}$ packet is generated from two component completion sources, a $\mathsf{ProductMetric}^{\uparrow}$ product-distance row, a $\mathsf{MetricCompletion}^{\uparrow}$ route, a $\mathsf{SeparatedCompletion}^{\uparrow}$ comparison boundary, paired $\mathsf{StreamName}^{\uparrow}$ windows, paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompleteProductMetric}^{\uparrow}}$, seed)

Scope closed: Definition 6732.1 and Definition 6732.2 seed the finite complete product-metric route from displayed product-distance and completion rows to separated completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompleteProductMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host product metrics, quotient completion equality, selected representatives, countable choice, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-distance exactness, completion-route stability, separated-boundary stability, transport, replay, provenance, and local naming for BEDC.Derived.CompleteProductMetricUp.

Chapter 6733

A Concrete Naming Certificate for RegularCauchyProductMetric$^{\uparrow}$

Following Bishop–Bridges product-metric and regular-Cauchy sequence mathematics, the RegularCauchyProductMe
packet realizes the paired finite-window product-distance route used by the compact-uniform-continuity construction of chapter 9237. It sits after the product-distance packet of chapter 3551, the regular product route of chapter 461, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives finite-net and uniform-modulus consumers a paired regular-Cauchy product metric surface without host product metrics, quotient stream equality, selected product limits, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 6733.1 (Regular Cauchy product-metric carrier)**.** *A* RegularCauchyProductMetric$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, K, W_A, W_B, R_A, R_B, D, \Delta, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two accepted regular Cauchy source rows.*
2. *K is the accepted* RegularCauchyProduct$^{\uparrow}$ *row joining the two source names by displayed product windows.*
3. *W_A and W_B are the synchronized* StreamName$^{\uparrow}$ *finite-window rows inspected by finite-net consumers.*
4. *R_A and R_B are the* RegSeqRat$^{\uparrow}$ *readback rows attached to the two displayed window streams.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *product-distance ledger recording coordinate tolerances, product radius, and comparison budget.*
6. *Δ is the* CauchyProductMetric$^{\uparrow}$ *metric row read from K, W_A, W_B, R_A, R_B, D.*
7. *E is the terminal* Real$^{\uparrow}$ *seal for the product-distance read.*
8. *H records componentwise* hsame *transport for the source rows, product row, windows, readbacks, dyadic ledger, metric row, Real seal, continuation, provenance, and local naming.*
9. *C records displayed* Cont *replay from a finite-net product-distance request through $A, B, K, W_A, W_B, R_A, R_B, D, \Delta$*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyProductMetric}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, K, W_A, W_B, R_A, R_B, D, \Delta, E, H, C, P, N$. It has no coordinate for host product metrics, quotient stream equality, selected limits, compactness, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6733.2 (Regular Cauchy product-metric NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyProductMetric}^{\uparrow}$ *carrier $M = (A, B, K, W_A, W_B, R_A, R_B, D, \Delta, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy sources, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *product row, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger, the* $\mathsf{CauchyProductMetric}^{\uparrow}$ *metric row, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of host product metrics, quotient stream equality, selected limits, compactness, trichotomy, detached real carriers, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 6733.1. The accepted packet exposes exactly the two source rows, the regular product row, synchronized windows, regular readbacks, dyadic ledger, metric row, Real seal, and structural rows.

The finite-net route reads the windows before it reads the product-distance ledger. Hence D is tied to W_A, W_B, R_A, R_B and cannot be replaced by a host product metric.

The metric row Δ is then forced through the sibling $\mathsf{CauchyProductMetric}^{\uparrow}$ route. It consumes the regular product and dyadic ledger as displayed rows, so compact-uniform consumers receive a product-distance certificate rather than quotient stream equality or selected limits.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no coordinate stores compactness, host product metrics, quotient streams, selected representatives, detached real carriers, ambient completeness, or bridge checking, the listed obligations exhaust the local certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductMetric}^{\uparrow}$ packet is generated from two regular Cauchy source rows, a $\mathsf{RegularCauchyProduct}^{\uparrow}$ row, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, a $\mathsf{CauchyProductMetric}^{\uparrow}$ metric row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyProductMetric}^{\uparrow}}$, seed)

Scope closed: Definition 6733.1 and Definition 6733.2 seed the finite regular-Cauchy product-distance route consumed by compact-uniform-continuity finite-net and uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyProductMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim compactness, host product metrics, quotient stream equality, selected limits, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-window exactness, dyadic product-distance ledger stability, CauchyProductMetric handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyProd

Chapter 6734

A Concrete Naming Certificate for $\mathsf{HadamardSpace}^{\uparrow}$

Following Hadamard and Bridson–Haefliger on complete CAT(0) geometry, the $\mathsf{HadamardSpace}^{\uparrow}$ packet records the finite NameCert surface for a complete geodesic metric carrier with the displayed nonpositive-curvature comparison row that convexity-facing consumers may read. It sits over the metric carrier of chapter 57, the complete-metric boundary of chapter 128, the geodesic metric route of chapter 2213, the convex-set rows of chapter 208, the CAT(0) metric comparison surface of chapter 6336, and the terminal real-name seal of chapter 38. The packet names the traditional Hadamard-space interface as finite BEDC rows; it does not assert global choice of geodesics, quotient point equality, selected projections, local compactness, differentiable manifold structure, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6734.1 (Hadamard space carrier)**.** *A* $\mathsf{HadamardSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, C, G, V, K, R, A, T, D, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, exposing the named point classifier, distance readbacks, ball observations, and metric-law ledger.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, recording the displayed Cauchy-thread or completion-facing service available over the same metric source.*
3. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *row, exposing only the finite geodesic incidence and endpoint-distance observations inspected by the packet.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the geodesic convex subsets and segment-closure reads that downstream projection or fixed-point consumers may inspect.*
5. *K is the* $\mathsf{CATZeroMetric}^{\uparrow}$ *comparison row, carrying the finite CAT(0) quadrilateral or comparison-triangle obligations used by the Hadamard read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row for the metric, geodesic, convexity, and CAT(0) distance budgets.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *distance and radius seal reached only after M, C, G, V, K, R have been displayed.*

8. *T records componentwise* hsame *transport for the metric, completeness, geodesic, convexity, comparison, rational-readback, and real-seal rows.*
9. *D records displayed* Cont *replay from Hadamard-space requests through the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{HadamardSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HadamardSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for selected geodesics, quotient points, ambient Hilbert spaces, choice-selected projections, differentiable structure, local compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6734.2 (Hadamard space NameCert obligations)**.** *For every accepted* $\mathsf{HadamardSpace}^{\uparrow}$ *carrier*

$$H = (M, C, G, V, K, R, A, T, D, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HadamardSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Met}^{\uparrow}$ *row M,* $\mathsf{CompleteMetric}^{\uparrow}$ *row C,* $\mathsf{GeodesicMetric}^{\uparrow}$ *row G,* $\mathsf{ConvexSet}^{\uparrow}$ *row V,* $\mathsf{CATZeroMetric}^{\uparrow}$ *row K,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal A, componentwise* hsame *transport T, displayed* Cont *replay D, provenance P, and local naming N. Every Hadamard-space consumer must read the complete geodesic CAT(0) surface through those rows and cannot import selected geodesics, quotient equality, projection selectors, Hilbert-space embeddings, local compactness, differentiable structure, or ambient real completeness.*

Proof. Project the carrier of Definition 6734.1. The metric source M, complete-metric row C, geodesic row G, convexity row V, and CAT(0) comparison row K are the only geometric data available to the packet.

The tolerance and real-valued distance reads are then forced through R and A. These rows attach the regular rational readback and terminal Real seal to the same displayed metric, geodesic, convexity, and comparison surface.

Finally use the structural rows T, D, P, N. They transport, replay, source, and name only the displayed rows, so there is no route from the accepted packet to selected geodesics, quotient point equality, projection selectors, Hilbert-space embeddings, local compactness, differentiable structure, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{HadamardSpace}^{\uparrow}$ read factors through displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any complete CAT(0) consumer output is visible.

Independence witness: metric, complete-metric, geodesic-metric, convex-set, cat-zero-metric, regseqrat, real: none of the listed siblings alone is bijective with $\mathsf{HadamardSpace}^{\uparrow}$ because the packet jointly carries a complete metric source, geodesic incidence, convexity reads, CAT(0) comparison, regular rational budgets, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HadamardSpace}^{\uparrow}$ packet is generated from MetricSpace carrier rows, CompleteMetric completion rows, GeodesicMetric segment rows, ConvexSet geodesic-convex rows, CatZeroMetric comparison rows, RegSeqRat tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HadamardSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6734.1 and Definition 6734.2 seed the finite complete geodesic CAT(0) NameCert surface over displayed metric, completeness, geodesic, convexity, comparison, tolerance, real-seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HadamardSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected geodesics, quotient point equality, projection selectors, Hilbert-space embeddings, local compactness, differentiable structure, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric exposure, geodesic incidence, convexity routing, CAT(0) comparison, rational readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HadamardSpaceUp.

Chapter 6735

A Concrete Naming Certificate for $\mathsf{MarkovKernel}^{\uparrow}$

Following Doob and Kallenberg on transition kernels, the $\mathsf{MarkovKernel}^{\uparrow}$ packet names the probability-theory interface that sends each displayed source state to a probability distribution on a displayed target event family. It sits between the event-and-measure surface of chapter 184, the measurable-preimage surface of chapter 185, the pushforward-law surface of chapter 186, and the finite transition-row surface of chapter 189. The chapter records the kernel row as its own NameCert object rather than reading it only as a Markov-chain transition coordinate.

Definition 6735.1 (Markov-kernel carrier)**.** *A* $\mathsf{MarkovKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, \mathcal{A}_T, k, N, E, H, C, P, L)$$

with the following displayed rows:

1. *S is the source-state row, carried as a measurable state surface compatible with a* $\mathsf{ProbSpace}^{\uparrow}$ *source when one is present;*
2. *T is the target-state row, equipped with the target measurable-event family $\mathcal{A}_T$;*
3. *k is the endpoint assignment sending each displayed source state s and target event $B \in \mathcal{A}_T$ to a* $\mathsf{Real}^{\uparrow}$ *endpoint $k(s, B)$;*
4. *N records normalization, requiring the row $B = T$ to read as the probability unit for each displayed source state;*
5. *E records empty-event and finite-disjoint-union readback for the target event family at each displayed source state;*
6. *H records componentwise* hsame *transport for source states, target events, and real endpoints;*
7. *C records displayed* Cont *replay from source state and target event rows through endpoint, normalization, finite-additivity, and naming rows;*
8. *P is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{MarkovKernel}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host conditional probability object, filtration, selected stochastic process, quotient event algebra, or off-packet measurability theorem.

Theorem 6735.2 (Markov-kernel NameCert obligations)**.** *For the carrier of Definition 6735.1, the local* $\mathsf{NameCert}_{\mathsf{MarkovKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S, T, \mathcal{A}_T, k, N, E, H, C, P, L),$$

source-state exposure through S*, target-event exposure through* $T, \mathcal{A}_T$*, endpoint readback through* k*, probability normalization through* N*, empty-event and finite-disjoint-union readback through* E*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* L*.*

Consequently an accepted $\mathsf{MarkovKernel}^{\uparrow}$ *packet is a finite state-to-distribution interface. It is not a host conditional distribution, not a filtration, and not a Markov-chain process.*

Lean4 ✓ `BEDC.Derived.MarkovKernelUp.MarkovKernelCarrier namecert obligations`

Proof. Project the displayed carrier. Its rows expose the source state, target event family, real-valued endpoint assignment, normalization row, finite-additivity row, transport, replay, provenance, and local name.

Read N and E as the distribution discipline at a fixed source state. The normalization row supplies the total target event as probability unit, while the empty-event and finite-disjoint-union rows make each displayed source fiber behave as a probability-distribution row over $\mathcal{A}_T$.

Use H, C, P, L to bind the same rows under classifier transport and replay. Since no coordinate contains a conditional-probability object, filtration, stochastic process, or quotient event algebra, the NameCert obligations are exhausted by the displayed finite rows. □

Theorem 6735.3 (Markov-kernel transition-distribution handoff)**.** *Every* $\mathsf{MarkovChain}^{\uparrow}$ *transition row that is accepted as a* $\mathsf{MarkovKernel}^{\uparrow}$ *packet hands its current-state input to a* $\mathsf{Distribution}^{\uparrow}$ *row on the next-state event family. The handoff first displays the* $\mathsf{ProbSpace}^{\uparrow}$ *source boundary and the adjacent* $\mathsf{RandomVar}^{\uparrow}$ *rows, then reads the current-state coordinate through the kernel row* k*, and only then exposes the next-law endpoint as a* $\mathsf{Distribution}^{\uparrow}$ *target distribution. It exports no host stochastic process, filtration, conditional-expectation primitive, or selected path law.*

Proof. Start with the Markov-chain carrier of Definition 189.1. Its transition row already records a current-state to next-law datum between adjacent $\mathsf{RandomVar}^{\uparrow}$ rows over a $\mathsf{ProbSpace}^{\uparrow}$ source.

Apply the Markov-kernel carrier above to that transition row. The source state is the current-state coordinate, the target event family is the next-state measurable family, and the endpoint assignment $k(s, B)$ is the displayed next-law readback.

Now read the target law through Definition 186.1. The $\mathsf{Distribution}^{\uparrow}$ surface receives a target event and a real endpoint, while the $\mathsf{RandomVar}^{\uparrow}$ and $\mathsf{ProbSpace}^{\uparrow}$ rows provide the measurable source boundary. The handoff therefore factors through the finite kernel packet rather than through any host stochastic-process structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovKernel}^{\uparrow}$ packet is generated from source-state rows, target-event rows, state-indexed real endpoints, normalization, finite-disjoint-union readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MarkovKernel}^{\uparrow}}$, seed)

Scope closed: Definition 6735.1, Definition 6735.2, and Definition 6735.3 seed the state-to-distribution kernel packet and its Markov-chain transition handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MarkovKernel}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host conditional-probability object, filtration, selected stochastic process, quotient event algebra, full measurability theorem, or standard kernel bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint normalization, finite-additivity readback, classifier stability, distribution handoff, provenance, and local NameCert for the same packet.

Chapter 6736

A Concrete Naming Certificate for $\mathsf{ToeplitzSummability}^{\uparrow}$

Following Toeplitz's theorem on regular summability matrices, as presented in Hardy's summability theory, the $\mathsf{ToeplitzSummability}^{\uparrow}$ packet records the finite BEDC certificate by which a row-window matrix transform preserves an already displayed real sequence limit. It sits beside the Cesaro route of chapter 1827, the Abel summation surface of chapter 1763, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the local carrier and handoff obligations for regular matrix summability; it does not import Banach limits, quotient sequence equality, selected representatives, host infinite matrices, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6736.1 (Toeplitz summability carrier)**.** *A* $\mathsf{ToeplitzSummability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, B, Z, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Real}^{\uparrow}$ *sequence source row, exposing only the finite source observations and the already displayed real-name limit requested by the summability packet.*
2. *M is the finite row-window matrix ledger, listing the inspected coefficients $a_{n,k}$, row indices, column windows, truncation bounds, and dyadic coefficient approximants currently visible.*
3. *B is the row-sum boundedness ledger, recording the finite dyadic bounds that keep the displayed row sums uniformly controlled before any transformed read is exposed.*
4. *Z is the column-limit-zero ledger, recording for each displayed column and tolerance the finite evidence that sufficiently late matrix rows make that column small.*
5. *U is the row-sum-one limit ledger, recording the finite evidence that the inspected row sums approach the unit row over the displayed tolerance schedule.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the transformed sequence obtained from S, M, B, Z, U.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source row, matrix ledger, three regularity ledgers, and regular rational readback have been displayed.*
8. *H stores componentwise* hsame *transport for the source row, matrix ledger, row-sum bound, column-limit-zero data, row-sum-one data, regular readback, real seal, continuation row, provenance, and local naming.*

9. *C stores displayed* Cont *replay from matrix-row inspection to source-window selection, from the regularity ledgers to transformed-row construction, from transformed rows to regular readback, and from regular readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{ToeplitzSummability}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ToeplitzSummability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, B, Z, U, R, E, H, C, P, N$. It has no coordinate for Banach limits, choice-selected subsequences, quotient sequence equality, host infinite matrices, detached summability methods, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6736.2 (Toeplitz summability NameCert obligations)**.** *For every accepted* $\mathsf{ToeplitzSummability}^{\uparrow}$ *carrier $T = (S, M, B, Z, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ToeplitzSummability}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real sequence source, finite row-window matrix ledger, row-sum boundedness ledger, column-limit-zero ledger, row-sum-one ledger, regular readback, real seal, transport, continuation, provenance, and naming rows;*

2. *matrix-window exactness, requiring every coefficient used by the transform to be read from M over an inspected finite row and column window;*

3. *regularity exactness, requiring bounded row sums, vanishing columns, and row sums approaching one to be read respectively from B, Z, U before the transformed row reaches R;*

4. *source-to-transform handoff, requiring every transformed finite row to be constructed from the same displayed source row S and matrix ledger M;*

5. *regular readback and real sealing, requiring completion-facing output to pass through R and then E;*

6. *transport, continuation, provenance, and local naming, requiring H, C, P, N to preserve the same finite route without adding Banach limits, choice, quotient equality, host infinite matrices, or ambient completeness.*

Proof. Project the carrier of Definition 6736.1. Carrier admission is exactly the displayed inventory $S, M, B, Z, U, R, E, H, C, P, N$, so the certificate surface has no detached coordinate for a completed matrix or a host summability method.

The matrix-window clause is read from M. The three regularity ledgers are independent rows: bounded row sums are supplied by B, column-limit-zero evidence by Z, and row-sum-one evidence by U.

The source sequence is inspected only through S, and every transformed finite row is downstream of S, M, B, Z, U. Hence no transformed read can reach $\mathsf{RegSeqRat}^{\uparrow}$ before those rows have been displayed.

The endpoint-facing route then continues through R and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The structural rows H, C, P, N transport, replay, source, and name this same route, so the six listed obligations exhaust the local NameCert surface. □

Theorem 6736.3 (Toeplitz summability regular-matrix handoff)**.** *Every accepted* $\mathsf{ToeplitzSummability}^{\uparrow}$ *read that exposes a regular matrix transform factors through the finite route*

$$S \longmapsto M \longmapsto (B, Z, U) \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

The $\mathsf{CesaroConvergence}^{\uparrow}$ *and* $\mathsf{AbelSummation}^{\uparrow}$ *sibling rows may appear as provenance anchors for familiar summability subroutes, but no consumer can bypass the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with an accepted transform read. The only source data visible to the classifier are the displayed observations in S, and the only coefficient data are the finite row windows in M.

The matrix is regular only through the three ledgers B, Z, U. Thus the transform has no accepted route unless row-sum boundedness, column-limit-zero evidence, and row-sum-one evidence have all been displayed for the inspected windows.

The transformed sequence reaches a regular rational consumer through R, and the real endpoint is visible only through E. The Cesaro and Abel sibling references live in provenance row P and do not create a second path around S, M, B, Z, U, R, E.

Finally H, C, P, N preserve and name the same finite route. Since the carrier contains no Banach-limit field, quotient sequence equality field, host infinite matrix, choice selector, or ambient completeness witness, the displayed route exhausts the regular-matrix handoff. □

Theorem 6736.4 (Toeplitz summability Real limit preservation)**.** *Let* $T = (S, M, B, Z, U, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ToeplitzSummability}^{\uparrow}$ *carrier. If the source row* S *exposes a* $\mathsf{Real}^{\uparrow}$ *sequence name with a displayed real limit, and* M, B, Z, U *expose the finite regularity rows required by Toeplitz's theorem, then the transformed sequence may be read by a completion-facing consumer only as the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *followed by the* $\mathsf{Real}^{\uparrow}$ *seal* E*. The exported real endpoint is the endpoint carried by the displayed source route, not a Banach limit, a selected external limit, a quotient sequence equality class, a host infinite matrix calculation, or an ambient completeness theorem.*

Proof. Use Definition 6736.2 to read the source and matrix data. The displayed source row S supplies the real sequence name and its limit row, while M supplies only finite coefficient windows.

The Toeplitz regularity evidence is then exactly B, Z, U. Row-sum boundedness controls the finite row transform, column-limit-zero evidence prevents any fixed source coordinate from escaping the displayed tail discipline, and row-sum-one evidence preserves the constant-limit row.

Apply Definition 6736.3. The transformed sequence cannot be exposed directly as a host limit; it first reaches the $\mathsf{RegSeqRat}^{\uparrow}$ row R and then the $\mathsf{Real}^{\uparrow}$ seal E.

The remaining rows H, C, P, N transport, replay, source, and name this route. Therefore the exported endpoint is the displayed BEDC real-limit preservation route, with no additional field for Banach limits, choice, quotient equality, host infinite matrices, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{ToeplitzSummability}^{\uparrow}$ completion-facing read factors through a Real sequence source row, a finite row-window matrix ledger, row-sum boundedness, column-limit-zero rows, row-sum-one limit rows, RegSeqRat readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: cesaroconvergence, abelsummation, regseqrat, real: $\mathsf{ToeplitzSummability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source real-sequence data, finite matrix row windows, all three Toeplitz regularity ledgers, regular readback, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ToeplitzSummability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ToeplitzSummability}^{\uparrow}$ packet is generated from a Real sequence source row, a finite row-window matrix ledger, row-sum boundedness, column-limit-zero rows, row-sum-one limit rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ToeplitzSummability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6736.1, Definition 6736.2, Definition 6736.3, and Definition 6736.4 seed the finite regular matrix summability route from displayed Real sequence data to Real-facing seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{ToeplitzSummability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach limits, choice, quotient sequence equality, host infinite matrices, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking for ToeplitzSummability$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, matrix-window exactness, row-sum boundedness, column-limit-zero exactness, row-sum-one exactness, RegSeqRat readback, Real-seal preservation, transport, provenance, and TasteGate field faithfulness for BEDC.Derived.ToeplitzSummabilityUp.

Chapter 6737

A Concrete Naming Certificate for $\mathsf{DoobDecomposition}^{\uparrow}$

Following Doob and Meyer, the $\mathsf{DoobDecomposition}^{\uparrow}$ packet records the finite row surface by which a displayed submartingale window is split into a martingale residual row and a predictable increasing compensator row. It sits beside the submartingale and martingale packets of chapter 6556 and chapter 190, consumes the conditional-expectation surface of chapter 188, and feeds the bounded stopping and crossing consumers of chapter 383 and chapter 1629. The chapter names only the finite decomposition carrier; it does not import measure completion, almost-sure quotient equality, optional projection, or a host predictable sigma-algebra.

Definition 6737.1 (Doob decomposition carrier)**.** *A* $\mathsf{DoobDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Omega, S, M, A, C, F, X, E, R, I, T, H, K, P, N)$$

with the following displayed rows:

1. Ω *is the consumed probability-context row shared with the input stochastic process.*
2. S *is the accepted* $\mathsf{Submartingale}^{\uparrow}$ *row being decomposed.*
3. M *is the* $\mathsf{Martingale}^{\uparrow}$ *residual row over the same finite filtration window.*
4. A *is the predictable increasing compensator row, displayed as finite step differences visible at the previous filtration stage.*
5. C *is the* $\mathsf{CondExp}^{\uparrow}$ *row used to read the successor conditional-expectation faces.*
6. F *is the finite filtration and adaptedness ledger shared by* S, M, A.
7. X *is the input submartingale endpoint family.*
8. E *is the finite conditional-expectation family read from* X *through* C.
9. R *is the residual endpoint family obtained by subtracting the predictable step differences displayed in* A *from the input endpoints* X.
10. I *is the increasing-row ledger recording nonnegative displayed step differences of* A.
11. T *is the bounded stopping-window handoff row consumed by* $\mathsf{OptionalStopping}^{\uparrow}$ *and crossing consumers.*
12. H *records componentwise* hsame *transport for all displayed rows.*

13. *K records displayed* Cont *replay for input exposure, conditional-expectation exposure, residual readback, predictable step-difference readback, stopping-window routing, and crossing handoff.*
14. *P is the* Pkg *provenance row naming exactly* $\mathsf{Submartingale}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the local decomposition rows.*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{DoobDecomposition}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed finite rows. It has no coordinate for completed filtrations, almost-sure quotient classes, optional projection, selected path representatives, or a host predictable sigma-algebra.

Theorem 6737.2 (Doob decomposition NameCert obligations)**.** *For the carrier of Definition 6737.1, the local* $\mathsf{NameCert}_{\mathsf{DoobDecomposition}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$\Omega, S, M, A, C, F, X, E, R, I, T, H, K, P, N,$$

submartingale input exposure through S, residual martingale exposure through M, successor conditional-expectation exposure through C, E, predictable increasing exposure through A, I, shared finite filtration exposure through F, endpoint and residual readback through X, R, bounded stopping and crossing handoff through T, componentwise hsame *transport through H, displayed* Cont *replay through K, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6737.1. Its row list fixes the input submartingale, residual martingale, predictable increasing compensator, conditional-expectation faces, common filtration window, endpoint families, stopping handoff, and structural rows.

The decomposition rows are $S, M, A, C, F, X, E, R, I$. The input row S supplies the one-sided conditional-expectation comparison, C, E display the successor faces, A, I display the predictable nonnegative step differences, and M, R record the residual martingale readback on the same finite filtration window.

The consumer rows remain bounded. The row T exposes only the finite stopped-window route needed by $\mathsf{OptionalStopping}^{\uparrow}$ and $\mathsf{Upcrossing}^{\uparrow}$, so downstream crossing consumers read the decomposition through displayed finite windows.

The structural rows H, K, P, N transport, replay, source, and name the same finite packet. Since no row names measure completion, almost-sure quotient equality, optional projection, selected path representatives, or a host predictable sigma-algebra, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{DoobDecomposition}^{\uparrow}$ consumer that reads a martingale residual or predictable increasing row without first displaying the $\mathsf{Submartingale}^{\uparrow}$ input row, the $\mathsf{CondExp}^{\uparrow}$ successor faces, and the shared finite filtration ledger is outside the carrier.

Independence witness: submartingale, martingale, condexp, optionalstopping, upcrossing: no listed sibling supplies the whole decomposition packet $D = (\Omega, S, M, A, C, F, X, E, R, I, T, H, K, P, N)$, because $\mathsf{Submartingale}^{\uparrow}$ names the one-sided input law, $\mathsf{Martingale}^{\uparrow}$ names equality-direction residual rows, $\mathsf{CondExp}^{\uparrow}$ names successor faces, and $\mathsf{OptionalStopping}^{\uparrow}$ and $\mathsf{Upcrossing}^{\uparrow}$ consume bounded windows without carrying the predictable increasing split.

Closure status for $\mathsf{NameCert}_{\mathsf{DoobDecomposition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DoobDecomposition}^{\uparrow}$ packet is generated from a displayed probability context, a submartingale input row, a martingale residual row, a predictable increasing compensator row, conditional-expectation faces, a finite filtration ledger, endpoint and residual readback rows, bounded stopping handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{DoobDecomposition}\uparrow}$, seed)

Scope closed: Definition 6737.1 and Definition 6737.2 seed the finite Doob decomposition NameCert boundary between submartingale input rows, martingale residual rows, predictable increasing rows, conditional-expectation faces, and bounded stopping or crossing consumers.

Formal verification: FormalStatus(NameCert$_{\text{DoobDecomposition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim uniqueness up to almost-sure equality, measure completion, optional projection, completed filtrations, selected path representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, decomposition-row exactness, predictable increasing exposure, martingale residual exposure, bounded consumer handoff, and local NameCert accountability for BEDC.Derived.DoobDecompositionUp.

Chapter 6738

A Concrete Naming Certificate for $\mathsf{ContinuedFractionReal}^{\uparrow}$

Following Lagrange and Hardy–Wright on continued fractions, the $\mathsf{ContinuedFractionReal}^{\uparrow}$ packet records the real-facing finite-prefix route from continued-fraction data through convergent rows, dyadic tolerance ledgers, regular rational readback, and a terminal real-name seal. It sits after the partial-quotient source of chapter 342, the convergent table of chapter 1777, the dyadic tolerance ledger of chapter 291, the finite-window surface of chapter 126, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet names a finite readback route for continued-fraction real representations; it is not an infinite continued-fraction selector, quotient real equality, trichotomy, countable choice, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 6738.1 (ContinuedFractionReal carrier)**.** *A* $\mathsf{ContinuedFractionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, C, D, W, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *finite-prefix row, recording the inspected partial quotients and the local recurrence source used by the real-facing read.*
2. *C is the* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *convergent-table row, recording numerator, denominator, normalized rational, and displayed finite-index data for the prefix in A.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint enclosures, and precision budgets attached to the displayed convergent windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording exactly which prefix windows and convergent indices a downstream consumer may inspect.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from C, D, W after the source prefix in A has been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite-prefix, convergent-table, dyadic-ledger, finite-window, and regular-rational rows have been read.*
7. *H records componentwise* hsame *transport for the source prefix, convergent table, dyadic ledger, finite windows, regular handoff, Real seal, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from a real-facing continued-fraction request through A, C, D, W, R, E, H.*

9. *P is the* Pkg *provenance over* ContinuedFractionReal$^{\uparrow}$, ContinuedFraction$^{\uparrow}$, ContinuedFractionConvergents$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuedFractionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, C, D, W, R, E, H, K, P, N$. It has no coordinate for an infinite continued-fraction selector, quotient real equality, trichotomy, countable choice, a selected limit, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6738.2 (ContinuedFractionReal NameCert obligations)**.** *For every accepted* ContinuedFractionReal$^{\uparrow}$ *carrier $F = (A, C, D, W, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuedFractionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed continued-fraction prefix, convergent table, dyadic tolerance ledger, finite windows, regular rational handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *convergent-window exactness, requiring the row C to expose only the convergents indexed by W and sourced by the prefix row A;*
3. *dyadic tolerance compatibility, requiring the ledger D to attach the same finite windows to the same convergent rows before R is available;*
4. *RegSeqRat handoff, requiring every regular rational read to pass through A, C, D, W before reaching R;*
5. *Real seal nonescape, requiring E to remain terminal with respect to A, C, D, W, R and to expose no detached real object;*
6. *transport, replay, provenance, and local-name stability through H, K, P, N.*

No local obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 6738.1. The accepted packet displays exactly the finite continued-fraction prefix, convergent table, dyadic tolerance ledger, finite windows, regular handoff, terminal Real seal, and structural rows.

Convergent-window exactness reads A, C, W together. The convergent table is not a separate infinite expansion; it lists only the numerator, denominator, normalized rational, and finite-index rows selected by the displayed window.

Dyadic tolerance compatibility then reads D over the same finite windows. The dyadic ledger supplies the radius and endpoint budget consumed by the regular rational row R, so no uninspected convergent tail or hidden precision source is available.

The handoff and seal are ordered by the carrier. The row R is downstream of A, C, D, W, and the row E is downstream of R.

Finally H, K, P, N transport, replay, source, and name only this displayed route. Since the carrier has no coordinate for quotient real equality, trichotomy, countable choice, selected limits, infinite selectors, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6738.3 (ContinuedFractionReal convergent handoff)**.** *Every real-facing read of an accepted* ContinuedFractionReal$^{\uparrow}$ *carrier passes through the finite continued-fraction prefix and convergent table before it reaches* RegSeqRat$^{\uparrow}$*. In particular, the displayed route is*

$$A \longmapsto C \longmapsto D, W \longmapsto R \longmapsto E.$$

The consumer receives finite partial-quotient data, convergent numerator and denominator rows, dyadic tolerance rows, finite windows, regular rational readback, and a terminal Real seal; it receives no selected infinite expansion or quotient-real representative.

Proof. Start from an accepted carrier $F = (A, C, D, W, R, E, H, K, P, N)$. The only continued-fraction source row is A, and the only convergent-table row is C.

The handoff row R is downstream of C, D, W. Thus a real-facing consumer must first read the finite prefix and its convergent table, then the dyadic tolerance and window rows that license the regular rational readback.

The terminal row E is available only after R. The structural rows preserve the same route under hsame, Cont, Pkg, and local naming, so the read cannot bypass the finite-prefix and convergent-table boundary. □

Theorem 6738.4 (ContinuedFractionReal seal nonescape)**.** *An accepted* ContinuedFractionReal$^{\uparrow}$ *packet exports no infinite continued-fraction selector, quotient real equality, trichotomy, countable choice, selected limit, or ambient* Real$^{\uparrow}$ *completeness. Every exported read factors through* $A, C, D, W, R, E, H, K, P, N$.

Proof. Project the carrier rows. The visible surface is exactly $A, C, D, W, R, E, H, K, P, N$, and each row is finite BEDC data.

The excluded principles require an off-packet coordinate: an infinite selector, an equality quotient, a trichotomy branch, a choice schedule, a chosen limiting object, or an ambient completeness theorem. None is present in the classifier or the provenance row.

Therefore transport and replay preserve only the displayed finite route. The terminal Real seal can expose the regular rational readback boundary, but it cannot export any of the excluded completed-real or choice-theoretic data. □

Lean4 (statement) `BEDC.Derived.ContinuedFractionRealUp`

Closure status for NameCert$_{\text{ContinuedFractionReal}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{ContinuedFractionReal}^{\uparrow}}$, seed)

Scope closed: Definition 6738.1, Definition 6738.2, Definition 6738.3, and Definition 6738.4 seed the finite continued-fraction real route through ContinuedFraction, convergent-table, DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ContinuedFractionReal}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.ContinuedFractionRealUp`

Bridge status: none

Not claimed: This chapter does not close infinite continued-fraction selectors, quotient real equality, trichotomy, countable choice, selected limits, ambient Real$^{\uparrow}$ completeness, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convergent-window exactness, dyadic tolerance compatibility, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuedFractionRealUp.

Constructive story: A ContinuedFractionReal$^{\uparrow}$ packet is generated from a ContinuedFraction finite-prefix row, a convergent-table row, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat handoff, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6739

A Concrete Naming Certificate for $\mathsf{LatinSquare}^{\uparrow}$

Following Euler and standard design theory, the $\mathsf{LatinSquare}^{\uparrow}$ packet records the finite carrier for a Latin square by naming the row, column, and symbol incidences rather than by importing a host array object. It sits beside the finite permutation surface of chapter 116 and the row-column readback surface of chapter 49: permutations certify the row and column symbol ledgers, while matrix rows supply the finite rectangular inspection window.

Definition 6739.1 (Latin-square carrier)**.** *A* $\mathsf{LatinSquare}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, C, S, A, P_r, P_c, H, T, G, N)$$

with the following displayed rows:

1. *R is the finite row-index window.*
2. *C is the finite column-index window.*
3. *S is the finite symbol row, with one displayed symbol entry for each visible row-column coordinate.*
4. *A is the matrix readback row over* $\mathsf{Mat}^{\uparrow}$*, recording that the visible entries are read only through the displayed row and column windows.*
5. *P_r is the row-permutation ledger: for each displayed row, the symbols across the visible columns are certified by a* $\mathsf{Permutation}^{\uparrow}$ *row.*
6. *P_c is the column-permutation ledger: for each displayed column, the symbols across the visible rows are certified by a* $\mathsf{Permutation}^{\uparrow}$ *row.*
7. *H records componentwise* hsame *transport for row indices, column indices, symbol entries, matrix readback, row permutations, column permutations, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a Latin-square inspection request through $R, C, S, A, P_r, P_c, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{LatinSquare}^{\uparrow}$*,* $\mathsf{Permutation}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an infinite design, an ambient quasigroup law, an arbitrary relabeling choice, or a host matrix equality principle.

Theorem 6739.2 (Latin-square NameCert obligations)**.** *For every accepted* $\mathsf{LatinSquare}^{\uparrow}$ *carrier* $L = (R, C, S, A, P_r, P_c, H, T, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$ *surface is exhausted by finite row-index admission, finite column-index admission, symbol-entry admission, matrix readback through* $\mathsf{Mat}^{\uparrow}$*, row-permutation ledgers, column-permutation ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The packet exports no ambient quasigroup operation, no infinite design axiom, no selected relabeling outside the displayed permutation rows, and no host matrix equality theorem.*

Proof. Project the carrier of Definition 6739.1. The rows R, C, S, A provide the finite inspection window and the displayed symbol entries.

Next read the two permutation ledgers. The row ledger P_r certifies that each displayed row sees the symbol row as a permutation, and the column ledger P_c certifies the same condition columnwise. Both ledgers are finite $\mathsf{Permutation}^{\uparrow}$ rows attached to the visible $\mathsf{Mat}^{\uparrow}$ readback.

Finally use H, T, G, N. These rows transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for a host quasigroup law, infinite design principle, arbitrary relabeling, or ambient matrix equality, the listed obligations exhaust the NameCert surface. □

Theorem 6739.3 (Latin-square row-column-symbol route)**.** *Every consumer of an accepted* $\mathsf{LatinSquare}^{\uparrow}$ *packet must read the finite route*

$$R, C, S, A, P_r, P_c, H, T, G, N$$

before using any Latin-square incidence. The row and column windows are inspected through the $\mathsf{Mat}^{\uparrow}$ *row* A*, the symbol uniqueness reads are inspected through the* $\mathsf{Permutation}^{\uparrow}$ *ledgers* P_r *and* P_c*, and the exported result is only the finite row-column-symbol route named by the local* $\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$ *surface.*

Proof. Start from Definition 6739.2. It makes the row, column, symbol, matrix, and permutation rows the visible surface of every accepted carrier.

The sibling dependencies then fix the two read directions. The $\mathsf{Mat}^{\uparrow}$ row supplies the finite rectangular address space, while the two $\mathsf{Permutation}^{\uparrow}$ ledgers certify that rows and columns each display the same finite symbol row without repetition.

Transport and replay preserve exactly this route. Thus a downstream design-theory or finite-combinatorics consumer receives the displayed finite packet and cannot replace it by an ambient quasigroup law, a hidden relabeling, or a host matrix equality theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$.

Constructive story: A $\mathsf{LatinSquare}^{\uparrow}$ packet is generated from finite row-index windows, finite column-index windows, symbol-entry rows, MatrixUp readback, row and column PermutationUp ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$, seed)

Scope closed: Definition 6739.1, Definition 6739.2, and Definition 6739.3 seed the finite row-column-symbol carrier over displayed $\mathsf{Permutation}^{\uparrow}$ and $\mathsf{Mat}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LatinSquare}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close orthogonal arrays, block designs, quasigroup isotopy, completion of partial Latin squares, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-index and column-index exactness, symbol-entry exactness, row-permutation and column-permutation ledger coverage, matrix readback, transport, replay, provenance, and local naming for BEDC.Derived.LatinSquareUp.

Chapter 6740

A Concrete Naming Certificate for $\mathsf{MayerVietorisSequence}^{\uparrow}$

Following Mayer–Vietoris and the Eilenberg–Steenrod exactness discipline, the $\mathsf{MayerVietorisSequence}^{\uparrow}$ packet records the finite boundary-route surface needed to read the classical decomposition sequence as BEDC NameCert data. It sits beside the homology carrier of chapter 98, the cohomology substrate of chapter 99, and the simplicial-complex rows of chapter 238. The packet names cover, intersection, inclusion, boundary, connecting-map, transport, replay, provenance, and local naming rows; it does not assert full exactness beyond the displayed finite rows, excision as a global theorem, selected chain representatives, quotient homology equality, or ambient topological choice.

Definition 6740.1 (Mayer–Vietoris sequence carrier)**.** *A* $\mathsf{MayerVietorisSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, U, V, I, A, B, C, D, \Delta, S, H, R, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{SimplicialComplex}^{\uparrow}$ *or homology-facing source row.*
2. *U, V are the two named subcomplex or subspace rows whose union is read only as a displayed cover ledger.*
3. *I is the intersection row $U \cap V$ with its own finite carrier and membership ledger.*
4. *A, B, C are the* $\mathsf{Homology}^{\uparrow}$ *rows for I, $U \oplus V$, and X at the inspected degree window.*
5. *D records the inclusion-induced maps from I into U and V, with the sign convention exposed as finite boundary data.*
6. *Δ is the connecting-boundary row from the inspected X-homology window to the preceding intersection window.*
7. *S is the finite sequence ledger listing the displayed row order*

$$A \longmapsto B \longmapsto C \longmapsto \Delta.$$

8. *H records componentwise* hsame *transport for cover, intersection, homology, inclusion, boundary, sequence, provenance, and local naming rows.*
9. *R records displayed* Cont *replay for consumers that ask for the same finite boundary route.*

10. *P is the* Pkg *provenance over* $\mathsf{MayerVietorisSequence}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MayerVietorisSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for unrestricted exactness, excision outside the displayed cover, selected cycles, quotient homology representatives, host set equality, or ambient topological choice.

Lean4 ✓ `BEDC.Derived.MayerVietorisSequenceUp.MayerVietorisSequenceTasteGate single carrier alignment`

Theorem 6740.2 (Mayer–Vietoris sequence NameCert obligations)**.** *For every accepted* $\mathsf{MayerVietorisSequence}^{\uparrow}$ *carrier $M = (X, U, V, I, A, B, C, D, \Delta, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MayerVietorisSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient source X, cover rows U, V, intersection row I, homology rows A, B, C, inclusion-boundary row D, connecting row Δ, sequence ledger S, componentwise transport H, displayed replay R, provenance P, and local naming N. A consumer may read finite boundary-route obligations from these rows, but it receives no full long exact sequence principle beyond the displayed inspected window.*

Proof. Project the carrier of Definition 6740.1. The cover, intersection, homology, inclusion, connecting, sequence, transport, replay, provenance, and naming rows are the only coordinates available to the packet.

The finite boundary route is then read through A, B, C, D, Δ, S. The ledger S fixes the inspected order, while D and Δ expose only the inclusion and connecting-boundary data displayed for that window.

Finally use the structural rows H, R, P, N. They preserve, replay, source, and name the same finite packet, so the local certificate cannot export unrestricted exactness, global excision, selected representatives, quotient equality, host set equality, or ambient topological choice. □

Theorem 6740.3 (Mayer–Vietoris boundary handoff)**.** *Every downstream exact-sequence, homology, or cohomology consumer of an accepted* $\mathsf{MayerVietorisSequence}^{\uparrow}$ *packet must read the displayed boundary route*

$$I, U, V, X \longmapsto A, B, C, D, \Delta, S$$

before it may consume the structural rows H, R, P, N. The handoff supplies finite inclusion and connecting-boundary obligations for the inspected row window, not full exactness of all degrees, excision outside the named cover, selected chain representatives, quotient homology equality, or an ambient topological model.

Proof. Begin with the NameCert obligations of Definition 6740.2. They force every consumer to read the cover rows U, V, the intersection row I, and the ambient row X before the homology sequence ledger is inspected.

The boundary data are then exposed by D, Δ, S. These rows record the inclusion maps, sign convention, connecting boundary, and finite sequence order for the displayed window only.

It remains to pass the route to downstream consumers. The transport and replay rows H, R preserve the displayed boundary route, while P, N source and name it. Since no other coordinate is present, the handoff cannot provide full long exactness, global excision, selected representatives, quotient equality, or host topological data. □

Closure status for $\mathsf{NameCert}_{\mathsf{MayerVietorisSequence}^{\uparrow}}$.

Constructive story: A MayerVietorisSequence$^{\uparrow}$ packet is generated from an ambient simplicial or homology source row, two cover rows, an intersection row, homology rows, inclusion-boundary rows, a connecting-boundary row, a finite sequence ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MayerVietorisSequence}^{\uparrow}}$, seed)

Scope closed: Definition 6740.1, Definition 6740.2, and Definition 6740.3 seed the finite boundary-route NameCert surface for Mayer–Vietoris sequence packets over homology and simplicial-complex rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MayerVietorisSequence}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.MayerVietorisSequenceUp.MayerVietorisSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim full exactness beyond displayed rows, global excision, selected cycles, quotient homology equality, host set equality, or bridge checking.

Upgrade path: Obligation closure requires checked cover, intersection, homology, inclusion, connecting-boundary, sequence-ledger, transport, replay, provenance, and local naming rows for BEDC.Derived.MayerVietorisSequenceUp.

Chapter 6741

A Concrete Naming Certificate for $\mathsf{RealLineOrderTopology}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{RealLineOrderTopology}^{\uparrow}$ packet names the finite open-interval topology surface induced by the displayed $\mathsf{Real}^{\uparrow}$ order classifier. It consumes the endpoint and seal rows of chapter 38, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the endpoint-cell boundary of chapter 394. The chapter records only a finite open-interval basis boundary; it does not assert trichotomy, open-cover compactness, quotient real equality, selected points, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6741.1 (Real line order-topology carrier)**.** *A* $\mathsf{RealLineOrderTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T_{\mathsf{Real}} = (E, O, S, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite endpoint row. Each endpoint is read as a* $\mathsf{Real}^{\uparrow}$ *order-classifier boundary together with the adjacent* $\mathsf{LocatedInterval}^{\uparrow}$ *cell in which the endpoint is inspected.*
2. *O is the open-interval probe row. A probe records a finite request of the form (a, b, w), where a and b are displayed endpoint names and w is the inspected finite window.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to every endpoint and interval probe.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained endpoint windows and probe windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare endpoint windows and to bound the open-interval probe.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after E, O, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for endpoint rows, open-interval probes, windows, readbacks, dyadic tolerances, the Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an endpoint request through the finite open-interval probe, the window row, the readback row, the dyadic tolerance row, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrderTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrderTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, O, S, Q, D, A, H, C, P, N$. The carrier has no coordinate for trichotomy, open-cover compactness, quotient real equality, selected points, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6741.2 (Real line order-topology NameCert obligations)**.** *For an accepted* $\mathsf{RealLineOrderTopology}^{\uparrow}$ *carrier* $T_{\mathsf{Real}} = (E, O, S, Q, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLineOrderTopology}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *endpoint admission for the displayed* $\mathsf{Real}^{\uparrow}$ *order-classifier endpoints and the adjacent* $\mathsf{LocatedInterval}^{\uparrow}$ *cells;*
2. *open-interval probe admission for the finite requests in O;*
3. *finite-window and readback admission through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*;*
4. *dyadic tolerance admission through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any endpoint comparison is replayed;*
5. *Real-seal admission through A after the endpoint, probe, window, readback, and tolerance rows have been displayed;*
6. *structural preservation by H, C, P, N;*
7. *non-escape for trichotomy, open-cover compactness, quotient real equality, selected points, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6741.1. The finite packet displays exactly the endpoint row, open-interval probe row, window row, readback row, dyadic tolerance row, Real seal, and structural rows.

The interval-topology part is read next. An open-interval probe in O is admitted only with endpoint data from E, and those endpoint rows are the same Real and located-interval rows cited in the carrier. Thus the topology surface is generated by displayed open intervals rather than by an ambient open-set lattice.

The numerical part is finite. The rows S, Q, D provide inspected windows, regular rational readback, and dyadic tolerance bounds before A can be read as a Real seal.

Finally H, C, P, N preserve, replay, source, and name the same route. Since the excluded principles have no carrier coordinate, the local certificate cannot export them as obligations. □

Theorem 6741.3 (Real line open-interval basis)**.** *Every accepted* $\mathsf{RealLineOrderTopology}^{\uparrow}$ *packet presents its topology-facing read as a finite open-interval basis generated from the rows*

$$E, O, S, Q, D, A, H, C, P, N.$$

A downstream $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer may inspect the basis only by first reading the displayed* $\mathsf{Real}^{\uparrow}$ *endpoints,* $\mathsf{LocatedInterval}^{\uparrow}$ *cell boundary,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances. The basis exports no trichotomy, open-cover compactness, quotient real equality, selected point, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 6741.1. The only topology-facing row is O, and it is a finite list of open-interval probes attached to endpoint rows E.

Read the endpoint route. The endpoints are sealed as $\mathsf{Real}^{\uparrow}$ rows and inspected through the located interval cells already named by chapter 394. They do not arrive as host real points or quotient classes.

Read the finite numerical route. The rows S, Q, D supply windows, regular rational readbacks, and dyadic tolerances for the interval probes before the Real seal A is available.

The structural rows H, C, P, N keep the same finite route under transport, replay, provenance, and local naming. Therefore any metric-space or continuous-map consumer receives a finite open-interval basis, not a classical topology with the excluded principles attached. □

Theorem 6741.4 (Real line order-topology Real-seal nonescape)**.** *In every accepted* $\mathsf{RealLineOrderTopology}^{\uparrow}$ *carrier* $T_{\mathsf{Real}} = (E, O, S, Q, D, A, H, C, P, N)$*, the Real seal* A *is downstream of the endpoint row* E*, the open-interval probe row* O*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*. Reading* A *cannot produce trichotomy, open-cover compactness, quotient real equality, selected points, host real equality, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the carrier definition. The row A is listed as the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after E, O, S, Q, D have been displayed.

Apply the NameCert obligations of Definition 6741.2. They force endpoint admission, open-interval probe admission, finite-window readback, and dyadic tolerance admission before the Real-seal row is inspected.

The open-interval basis theorem, Definition 6741.3, keeps every topology-facing consumer on that same displayed route. Thus a read of A remains attached to finite endpoint, interval, window, readback, and tolerance data.

The carrier has no row for the excluded principles. Consequently the terminal Real seal cannot be used to export trichotomy, open-cover compactness, quotient real equality, selected points, host equality, or $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: If a Real-line topology consumer reads an open set without displaying endpoint rows, finite open-interval probes, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, and the Real seal, it is not a $\mathsf{RealLineOrderTopology}^{\uparrow}$ packet.

Independence witness: real, streamname, regseqrat, dyadicratcore, locatedinterval: the packet is not bijective to any sibling because it binds endpoint rows, finite open-interval probes, window-readback tolerances, the Real seal, transport, replay, provenance, and local naming as one topology-facing boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrderTopology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealLineOrderTopology}^{\uparrow}$ packet is generated from finite Real endpoint rows, open-interval probes, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a terminal Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrderTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6741.1, Definition 6741.2, and Definition 6741.3 seed the finite order-topology boundary for Real-line consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrderTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, open-cover compactness, quotient real equality, selected points, host real equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-interval probe admission, finite-window readback, dyadic tolerance admission, Real-seal nonescape, and structural preservation for BEDC.Derived.RealLineOrderTopologyUp.

Chapter 6742

A Concrete Naming Certificate for $\mathsf{UltrametricBaireTheorem}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{UltrametricBaireTheorem}^{\uparrow}$ packet records the complete-ultrametric Baire theorem as finite BEDC NameCert data. It sits over the complete-ultrametric carrier of chapter 3879, the dense-open schedule of chapter 1542, the common-prefix ultrametric route of chapter 3485, the complete-metric service of chapter 128, the finite observation windows of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite route by which a complete ultrametric dense-open construction is exposed to theorem consumers; it does not assert a host intersection object, a selected branch, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness outside the displayed route.

Definition 6742.1 (Ultrametric Baire theorem carrier)**.** *An* $\mathsf{UltrametricBaireTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, U, V, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, listing the finite dense-open requests and predecessor-ball refinements visible to the theorem route.*
2. *U is the* $\mathsf{CompleteUltrametric}^{\uparrow}$ *source row, exposing the complete non-Archimedean Cauchy service consumed by the construction.*
3. *V is the* $\mathsf{BaireUltrametric}^{\uparrow}$ *finite common-prefix ultrametric row, binding the displayed cylinders to the ultrametric comparison route.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, recording the complete-metric service read by consumers that only know the metric interface.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected centers, radii, and dense-open requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed rational tolerances.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal for the radius and limit-name reads exposed by the route.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from the dense-open request through the complete-ultrametric and finite-window rows.*

10. *P is the* Pkg *provenance over* $\mathsf{UltrametricBaireTheorem}^{\uparrow}$, $\mathsf{CompleteUltrametric}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UltrametricBaireTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, a selected generic point, quotient equality of streams, host intersection of dense opens, spherical completeness, or open-cover compactness.

Theorem 6742.2 (Ultrametric Baire theorem NameCert obligations)**.** *For every accepted* UltrametricBaireTheorem *carrier $T = (B, U, V, M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UltrametricBaireTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dense-open schedule, complete-ultrametric source, common-prefix ultrametric route, complete-metric handoff, finite stream windows, regular-rational readback, terminal real seal, componentwise transport, displayed replay, package provenance, and local naming. Every theorem-level consumer must read the route through those rows before treating the complete-ultrametric Baire argument as public.*

Proof. Project the carrier of Definition 6742.1. The dense-open schedule is read from B, the complete-ultrametric source from U, the common-prefix ultrametric route from V, and the complete-metric interface from M.

The finite observation and tolerance data are then read through S, R, E. These rows expose the stream windows, regular-rational tolerances, and real-name seal without adding a selected branch or a host intersection object.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier lists no coordinate for countable choice, quotient stream equality, spherical completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface is exhausted by the displayed rows. □

Lean4 (statement) `BEDC.Derived.UltrametricBaireTheoremUp`

Falsifiable prediction: Every accepted $\mathsf{UltrametricBaireTheorem}^{\uparrow}$ read factors through $\mathsf{CompleteUltrametric}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows before any theorem-level complete-ultrametric Baire consumer receives the route.

Independence witness: complete-ultrametric, bairecategory, baire-ultrametric, completemetric, streamname, regseqrat, real: $\mathsf{UltrametricBaireTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the dense-open schedule, complete-ultrametric source, common-prefix ultrametric route, complete-metric handoff, finite windows, rational readback, real seal, transport, replay, provenance, and local naming as one theorem packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UltrametricBaireTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{UltrametricBaireTheorem}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteUltrametric source row, a BaireUltrametric common-prefix route, a CompleteMetric handoff, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UltrametricBaireTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 6742.1 and Definition 6742.2 seed the complete-ultrametric Baire theorem carrier and its finite NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UltrametricBaireTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, countable choice, a selected generic point, quotient stream equality, host dense-open intersection, spherical completeness, open-cover compactness, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exactness, complete-ultrametric source exposure, common-prefix route stability, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.UltrametricBaireTheoremUp.

Chapter 6743

A Concrete Naming Certificate for $\mathsf{AsplundSpace}^{\uparrow}$

Following Asplund and the standard Banach-space theory of separable dual restrictions, the $\mathsf{AsplundSpace}^{\uparrow}$ packet records the finite BEDC surface by which a Banach carrier exposes separable-subspace dual witnesses without importing a host dual-space theorem. It sits over the Banach-space surface of **??**, the norm row of chapter 89, the vector-space source of chapter 47, the functional-analysis handoff of chapter 94, and the terminal real-name seal of chapter 38. The packet names a finite Asplund interface for downstream weak compactness, differentiability, and dual-separability consumers; it does not choose all separable subspaces, construct a global dual object, quotient functionals, assert Radon–Nikodym differentiability, or import a host weak topology.

Definition 6743.1 (Asplund space carrier)**.** *An* $\mathsf{AsplundSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, V, N, F, S, D, R, T, C, P, L)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *carrier row, exposing the accepted complete normed vector-space source inspected by the packet.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row for the vector operations visible inside the accepted Banach carrier.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row recording the norm observations and comparison ledgers used by separable-subspace reads.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *route row binding bounded-linear-functional requests to the same Banach, vector, and norm source.*
5. *S is the separable-subspace window row, carrying only the finite named subspace probes and dense-window requests currently under inspection.*
6. *D is the separable-dual witness row, recording the finite dual-probe windows read from S through F.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for norm and functional-value comparisons reached after B, V, N, F, S, D have been displayed.*
8. *T records componentwise* hsame *transport for Banach, vector, norm, functional-analysis, separable-window, dual-witness, Real-seal, and local certificate rows.*

9. *C records displayed* Cont *replay from Asplund-space requests through the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{AsplundSpace}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{AsplundSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host dual Banach space, quotient functionals, all separable subspaces, weak-star compactness, differentiability theorems, selected dense sequences, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6743.2 (Asplund space NameCert obligations)**.** *For every accepted* $\mathsf{AsplundSpace}^{\uparrow}$ *carrier*

$$A = (B, V, N, F, S, D, R, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{AsplundSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{BanachSpace}^{\uparrow}$ *row B,* $\mathsf{VecSpace}^{\uparrow}$ *row V,* $\mathsf{Norm}^{\uparrow}$ *row N,* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *route F, separable-subspace window S, separable-dual witness row D, terminal* $\mathsf{Real}^{\uparrow}$ *seal R, componentwise transport T, displayed replay C, provenance P, and local naming L. Every Asplund consumer must read separable-dual data through those rows and cannot import a host dual space, a quotient functional universe, a selected dense sequence, a global weak topology, Radon–Nikodym differentiability, or ambient real completeness.*

Proof. Project the carrier of Definition 6743.1. The accepted Banach, vector, and norm source is exactly B, V, N, so the packet has no independent host-space coordinate.

Next read the functional-analysis and separable-dual surface through F, S, D. These rows bind every dual-facing request to the displayed separable-subspace window before the terminal Real seal can be consumed.

Finally attach R, T, C, P, L. They seal, transport, replay, source, and name only the displayed rows, which leaves no route to a host dual Banach space, quotient functionals, selected dense sequences, weak-star compactness, differentiability theorems, or ambient real completeness. □

Theorem 6743.3 (Asplund separable dual route)**.** *Every separable-dual consumer of an accepted* $\mathsf{AsplundSpace}^{\uparrow}$ *carrier must first expose the finite route*

$$\mathsf{BanachSpace}^{\uparrow} \longrightarrow \mathsf{VecSpace}^{\uparrow} \longrightarrow \mathsf{Norm}^{\uparrow} \longrightarrow \mathsf{FunctionalAnalysis}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the rows $B, V, N, F, S, D, R, T, C, P, L$ of Definition 6743.1. The route is a displayed separable-subspace dual witness surface, not a global dual-space construction, quotient-functional equality, selected dense sequence, weak compactness theorem, differentiability theorem, or host Banach-space theorem.

Proof. Start from Definition 6743.2. It places B, V, N, F before the separable-subspace window and dual-witness rows can be inspected.

The rows S and D then localize the dual read to the named finite subspace window. The terminal Real seal R records only norm and functional-value comparisons coming from that window.

The structural rows T, C, P, L preserve this same route. Since no carrier coordinate names all separable subspaces, a host dual object, quotient equality of functionals, selected dense sequences, weak compactness, or differentiability, the separable-dual route exports only the displayed NameCert packet. □

Falsifiable prediction: Every accepted $\mathsf{AsplundSpace}^{\uparrow}$ separable-dual read factors through displayed $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, separable-window, separable-dual witness, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any downstream weak-compactness or differentiability consumer output is visible.

Independence witness: banachspace, vecspace, norm, functionalanalysis, real: none of the listed siblings alone is bijective with $\mathsf{AsplundSpace}^{\uparrow}$ because the packet jointly carries a Banach source, vector operations, norm reads, functional-analysis routing, separable-subspace windows, separable-dual witnesses, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AsplundSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{AsplundSpace}^{\uparrow}$ packet is generated from BanachSpace carrier rows, VecSpace operation rows, Norm comparison rows, FunctionalAnalysis handoff rows, finite separable-subspace windows, separable-dual witness rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AsplundSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6743.1, Definition 6743.2, and Definition 6743.3 seed the finite Asplund-space NameCert surface over displayed Banach, vector, norm, functional-analysis, separable-window, dual-witness, real-seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AsplundSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host dual Banach space, quotient functionals, all separable subspaces, selected dense sequences, weak compactness, differentiability, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Banach-space exposure, vector and norm routing, functional-analysis handoff, separable-subspace window exactness, separable-dual witness readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.AsplundSpaceUp.

Chapter 6744

A Concrete Naming Certificate for CompactUniformEquicontinuityImage$^{\uparrow}$

Following Heine–Cantor and Arzela–Ascoli finite-net arguments, the CompactUniformEquicontinuityImage$^{\uparrow}$ packet records the image-side NameCert surface for compact metric sources whose continuous or equicontinuous family rows transport finite nets into uniformly controlled image windows. It realizes the compact-uniform-continuity vision of chapter 9237 through the compact-metric source of chapter 129, the total-boundedness finite-net row of chapter 127, the uniform-continuity handoff of chapter 391, the continuous-map rows of chapter 130, the equicontinuity surface of chapter 4076, and the terminal real-name seal of chapter 38. The packet names finite-net image transport; it does not assert open-cover compactness, selected image covers outside the displayed net, quotient equality of maps, a host compact-image theorem, or ambient Real$^{\uparrow}$ completeness.

Definition 6744.1 (Compact uniform equicontinuity image carrier)**.** *A* CompactUniformEquicontinuityImage$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, T, U, M, E, I, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing the compact metric finite-net data under inspection.*
2. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row read from the same compact source.*
3. *U is the* UniformContinuity$^{\uparrow}$ *modulus row for the image request.*
4. *M is the* ContinuousMap$^{\uparrow}$ *graph row whose values are transported along the finite net.*
5. *E is the* Equicontinuity$^{\uparrow}$ *family row when the image request is family-level rather than single-map.*
6. *I is the image-window row, recording the finite list of target values and radius ledgers obtained from K, T, U, M, E.*
7. *R is the terminal* Real$^{\uparrow}$ *seal for source radii, target radii, and image-window comparisons.*
8. *H records componentwise* hsame *transport for compact source, total-bounded finite net, uniform-continuity, continuous-map, equicontinuity, image-window, Real-seal, and local certificate rows.*
9. *C records displayed* Cont *replay from image requests through the same finite rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open-cover compactness proof, a selected external image cover, quotient equality of maps, a host Arzela–Ascoli theorem, global compact-image closure, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6744.2 (Compact uniform equicontinuity image NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$ *carrier*

$$Q = (K, T, U, M, E, I, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{CompactMetric}^{\uparrow}$ *source row K,* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row T,* $\mathsf{UniformContinuity}^{\uparrow}$ *modulus row U,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row M,* $\mathsf{Equicontinuity}^{\uparrow}$ *family row E, image-window row I, terminal* $\mathsf{Real}^{\uparrow}$ *seal R, componentwise transport H, displayed replay C, provenance P, and local naming N. Every image consumer must read finite-net transport through those rows and cannot import open-cover compactness, selected image covers, quotient equality of maps, host compact-image closure, Arzela–Ascoli compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 6744.1. The compact source and finite net are exactly K, T, so image transport starts from displayed compact-metric and total-bounded rows.

Next read the map control through U, M, E. The uniform-continuity row, continuous-map graph row, and optional family equicontinuity row are the only routes by which source-net radii become image-window radii.

Finally attach I, R, H, C, P, N. These rows record, seal, transport, replay, source, and name the finite image window, leaving no route to open-cover compactness, selected external image covers, quotient map equality, host compact-image closure, Arzela–Ascoli compactness, or ambient real completeness. □

Theorem 6744.3 (Compact finite-net image handoff)**.** *Every image-window consumer of an accepted* $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$ *carrier must first expose the sibling route*

$$\mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{TotallyBounded}^{\uparrow} \longrightarrow \mathsf{UniformContinuity}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

and, for family requests, the $\mathsf{Equicontinuity}^{\uparrow}$ *row as well. The handoff is a finite-net image packet over $K, T, U, M, E, I, R, H, C, P, N$, not an open-cover compactness proof, selected image cover, quotient-map equality, host compact-image theorem, or ambient Arzela–Ascoli closure.*

Proof. Start from Definition 6744.2. It places compact metric data K and total-bounded finite-net data T before any image-window row can be read.

The control rows U, M, E then transport finite source nets into target image windows. The row I records the displayed image centers and radius ledgers, and R supplies only the terminal Real comparisons attached to those ledgers.

The structural rows H, C, P, N preserve the same handoff. Since the carrier contains no open-cover proof, external image-cover selector, quotient equality of maps, host compact-image theorem, or Arzela–Ascoli compactness coordinate, the consumer receives exactly the finite-net image NameCert packet. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$ image read factors through displayed $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, image-window, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before a downstream compact-image or Arzela–Ascoli consumer output is visible.

Independence witness: compactmetric, totallybounded, uniformcontinuity, continuousmap, equicontinuity, real: none of the listed siblings alone is bijective with $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$ because the packet jointly carries a compact metric source, finite-net schedule, uniform-continuity modulus, continuous-map graph row, family equicontinuity row, image-window ledger, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}$ packet is generated from CompactMetric finite-net source rows, TotallyBounded schedules, UniformContinuity modulus rows, ContinuousMap graph rows, Equicontinuity family rows, finite image-window ledgers, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6744.1, Definition 6744.2, and Definition 6744.3 seed the finite-net image transport surface over compact source, total-bounded, uniform-continuity, continuous-map, equicontinuity, image-window, real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected image covers, quotient equality of maps, host compact-image closure, Arzela–Ascoli compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, total-bounded finite-net routing, uniform-continuity and continuous-map handoff, equicontinuity family routing, image-window exactness, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformEquicontinuityImageUp.

Chapter 6745

A Concrete Naming Certificate for $\mathsf{MoscoRecoverySequence}^{\uparrow}$

Following Mosco and Attouch on variational convergence and recovery sequences, the $\mathsf{MoscoRecoverySequence}^{\uparrow}$ packet records the finite completion-facing witness object that supplies explicit strong recovery data after a Mosco or Gamma convergence surface has exposed its lower-bound side. It sits beside the Mosco convergence certificate of chapter 5938, the Gamma-convergence recovery-window carrier of chapter 5957, the terminal value readback of chapter 38, and the completion handoff of chapter 1506. The packet is a witness ledger for recovery requests; it is not a selected minimizer, a quotient equality of functionals, a global compactness theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6745.1 (Mosco recovery sequence carrier)**.** *A* $\mathsf{MoscoRecoverySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, F, W, A, G, Q, E, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric or completion-facing source row on which the recovery witnesses are inspected.*
2. *F is the variational functional-window row inherited from a Mosco or Gamma convergence read.*
3. *W is the weak-liminf context row that fixes which lower-bound request the recovery witness answers.*
4. *A is the finite approximation schedule row, listing the inspected recovery candidates and their convergence windows.*
5. *G is the strong recovery goal row, recording the target point, admissible support, and upper-bound comparison demanded by the consumer.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *value-readback row for the displayed functional values and tolerance budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the recovery value rows have been exposed.*
8. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that lets completion-facing consumers read convergence of the displayed approximants without selecting a quotient representative.*
9. *H records componentwise* hsame *transport for X, F, W, A, G, Q, E, M and the local certificate rows.*

10. *C records displayed* Cont *replay from recovery requests through approximation windows, value readback, Real sealing, and completion handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{MoscoRecoverySequence}^{\uparrow}$, $\mathsf{MoscoConvergence}^{\uparrow}$, $\mathsf{GammaConvergence}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MoscoRecoverySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, W, A, G, Q, E, M, H, C, P, N$. It has no coordinate for a host-selected recovery sequence, minimizer existence, quotient equality of functionals, compactness by choice, or ambient completeness of the Real or completion carriers.

Lean4 ✓ `BEDC.Derived.MoscoRecoverySequenceUp.MoscoRecoverySequenceTasteGate single carrier alignment`

Theorem 6745.2 (Mosco recovery sequence NameCert obligations)**.** *For every accepted* $\mathsf{MoscoRecoverySequence}^{\uparrow}$ *carrier*

$$R = (X, F, W, A, G, Q, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MoscoRecoverySequence}^{\uparrow}}$ *surface consists exactly of carrier admission, source exposure through X, functional-window exposure through F, lower-bound context exposure through W, finite approximation scheduling through A, recovery-goal exposure through G, regular rational value readback through Q, terminal real sealing through E, metric-completion handoff through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The rows X, F, W, A, G, Q, E, M are the only coordinates from which a recovery consumer can read source data, variational windows, the lower-bound context, finite approximants, value budgets, real seals, and completion handoff.

The approximation row A and goal row G provide the explicit witness surface. They are read after the Mosco or Gamma convergence context has been displayed and before completion-facing consumers use M.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite packet, so the local NameCert obligations cannot be widened to a selected host recovery sequence, minimizer, quotient functional equality, choice schedule, or ambient completeness theorem. □

Lean4 ✓ `BEDC.Derived.MoscoRecoverySequenceUp.MoscoRecoverySequenceCarrier namecert obligations`

Theorem 6745.3 (Mosco recovery sequence real-metric boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{MoscoRecoverySequence}^{\uparrow}$ *carrier must read the finite route*

$\mathsf{MoscoConvergence}^{\uparrow}$, $\mathsf{GammaConvergence}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, Nam

through $X, F, W, A, G, Q, E, M, H, C, P, N$ before it treats the recovery witness as a real-valued metric approximation. The boundary supplies explicit recovery witnesses for the convergence surface; it does not assert selected minimizers, quotient equality of functionals, global compactness, countable choice, host equality of completion points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations. They place the finite recovery schedule A and goal row G between the convergence context F, W and the readback rows Q, E, M.

The Real-facing side is read through Q and E. A consumer sees only regular rational value windows and their terminal seal, not an ambient Real completeness theorem or a selected limiting value.

The metric side is read through M after the displayed approximants have been exposed. Transport, replay, provenance, and local naming keep the same rows attached to the packet, so the recovery witness remains an explicit finite object rather than a host-selected sequence or quotient-completion representative. □

Lean4 ✓ `BEDC.Derived.MoscoRecoverySequenceUp.MoscoRecoverySequenceRealMetricBoundary`

Falsifiable prediction: An accepted $\mathsf{MoscoRecoverySequence}^{\uparrow}$ consumer that reaches a real-valued completion approximation without reading $A, G, Q, E, M, H, C, P, N$ refutes the carrier surface.

Independence witness: mosco-convergence, gamma-convergence, metriccompletion, regseqrat, real: $\mathsf{MoscoRecoverySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a convergence context, explicit finite recovery schedules, recovery goals, rational value readback, Real sealing, metric-completion handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MoscoRecoverySequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MoscoRecoverySequence}^{\uparrow}$ packet is generated from convergence-context rows, finite recovery schedules, recovery-goal rows, RegSeqRat value readback, Real seals, metric-completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MoscoRecoverySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6745.1, Definition 6745.2, and Definition 6745.3 seed the finite recovery-witness route over convergence context, approximation schedule, value readback, Real seal, completion handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MoscoRecoverySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected minimizer existence, quotient equality of functionals, countable choice, global compactness, host equality of completion points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Formalization requires BEDC.Derived.MoscoRecoverySequenceUp carrier admission together with checked recovery scheduling, value readback, completion handoff, transport, replay, provenance, and local naming obligations.

Chapter 6746

A Concrete Naming Certificate for $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ packet records the unit map that sends a displayed $\mathsf{Real}^{\uparrow}$ point into the Cauchy-filter completion interface as finite BEDC NameCert data. It sits over the real-name seal of chapter 38, the Cauchy-filter carrier of chapter 333, the filter-base rows of chapter 1601, the metric-completion boundary of chapter 1506, the completion universal-property packet of chapter 397, the regular-rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the monad unit as a displayed point-to-filter route; it does not choose quotient representatives, select limits, import countable choice, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, or identify host filters.

Definition 6746.1 (Real Cauchy-filter monad-unit carrier)**.** *A* $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, F, B, M, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{Real}^{\uparrow}$ *row, already sealed by displayed regular-rational and stream-window data.*
2. *F is the principal* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated from A through finite neighbourhood observations.*
3. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row exposing the finite neighbourhood base used by the principal filter.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that receives the Cauchy-filter point as completion-facing data.*
5. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *unit-and-extension row naming the universal-property consumer of the same point.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the real-name source is inspected by rational windows.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite windows consumed by R.*
8. *H records componentwise* hsame *transport for $A, F, B, M, E, R, S, C, P, N$.*
9. *C records displayed* Cont *replay from the real point through filter-base, Cauchy-filter, completion, universal-property, readback, and stream-window rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, F, B, M, E, R, S, H, C, P, N$. It has no coordinate for quotient representatives, selected filter limits, host filter equality, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.RealCauchyFilterMonadUnitUp`

Theorem 6746.2 (Real Cauchy-filter monad-unit obligations)**.** *For every accepted* $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ *carrier*

$$U = (A, F, B, M, E, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Real source, Cauchy-filter row, filter-base row, completion handoff, universal-property row, rational readback, stream schedule, transport, replay, provenance, and local naming;*
2. *principal-filter exposure, requiring the Cauchy-filter row F to be read only through the finite filter-base row B generated from the real point A;*
3. *completion compatibility, requiring the metric-completion row M and universal-property row E to consume the same displayed Cauchy-filter route;*
4. *readback stability, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and* $\mathsf{StreamName}^{\uparrow}$ *row S to remain attached to the source* $\mathsf{Real}^{\uparrow}$ *row before the filter row is exposed;*
5. *structural preservation by H, C, P, N;*
6. *nonescape for quotient representatives, selected limits, host filter equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean theorem verification, and bridge checking.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 6746.1. The Real source is A, the filter-facing rows are F, B, the completion-facing rows are M, E, and the finite readback rows are R, S.

The principal-filter obligation is forced because the carrier has no second filter generator. A consumer reaches F only after the filter-base row B has displayed the finite neighbourhood observations generated from A.

Completion compatibility is rowwise. The metric-completion handoff M and the universal-property row E both consume the same F, B route, so neither can introduce a quotient representative or a selected completion point.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded objects have no carrier coordinate, they cannot become NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealCauchyFilterMonadUnitUp.RealCauchyFilterMonadUnitTasteGate single carrier alignment`

Theorem 6746.3 (Real Cauchy-filter monad-unit law)**.** *Every accepted* $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ *packet presents the monad-unit law as the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}, \mathsf{CompletionUniversalProperty}^{\uparrow}.$$

The route sends a displayed real point to its principal Cauchy-filter row and then to the completion interface, while the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *rows keep the real-name readback visible. It exports no quotient representative, selected limit, host filter equality, countable choice, or ambient completeness principle.*

Proof. Read the carrier coordinates. The source row A is the only Real input, and R, S are the only rational and stream readback rows attached to it.

The filter-base row B generates the principal filter row F. Thus the map from A to F is a displayed unit route rather than a choice of a filter representative.

The completion rows M, E then consume F and B as the same Cauchy-filter handoff. Transport, replay, provenance, and local naming preserve this route, so the monad-unit law is exactly the finite BEDC packet named in the statement. □

Falsifiable prediction: Any claimed $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ unit that reaches a completion row without displaying the Real source, filter-base row, Cauchy-filter row, regular-rational readback, stream schedule, transport, replay, provenance, and local NameCert rows is rejected by the carrier of Definition 6746.1.

Independence witness: real, cauchyfilter, filterbase, metriccompletion, completionuniversalproperty, regseqrat, streamname: none is bijective with $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ because only this packet jointly records a Real source, principal filter-base generation, Cauchy-filter handoff, completion-unit consumption, rational readback, finite stream schedule, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}$ packet is generated from a RealUp source row, a principal FilterBaseUp row, a CauchyFilterUp row, MetricCompletionUp and CompletionUniversalPropertyUp handoff rows, RegSeqRatUp readback, StreamNameUp windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6746.1, Definition 6746.2, and Definition 6746.3 seed the finite monad-unit packet from displayed Real points to Cauchy-filter completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterMonadUnit}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealCauchyFilterMonadUnitUp`

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected filter limits, host filter equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, principal-filter exposure, completion compatibility, readback stability, structural preservation, and nonescape for BEDC.Derived.RealCauchyFilterMonadUnitUp.

Chapter 6747

A Concrete Naming Certificate for $\mathsf{ClosedObservationProofTrace}^{\uparrow}$

The $\mathsf{ClosedObservationProofTrace}^{\uparrow}$ packet realizes the finite-observation branch of chapter 9370 as a MetaCIC-facing NameCert object. It records a proof trace only when finite observation rows, termination rows, transport, replay, provenance, local naming, refusal rows, kernel-acceptance rows, and proof-trace rows are displayed together. The packet is not a quotient of proof terms, not a host equality of derivations, not an erasure theorem, and not a divergent computation hidden behind a proof record.

The proof-trace row consumes the finite evaluator of chapter 818 and the kernel acceptance boundary of chapter 6007 before it is replayed. The finite evaluator supplies the closed observation and totality surface, while the kernel acceptance boundary supplies the accepted proof-record row; no closurestatus or TasteGate promotion is made at this dependency handoff.

Definition 6747.1 (Closed observation proof-trace carrier)**.** *A* $\mathsf{ClosedObservationProofTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, Q, H, C, P, N, R, K, D)$$

with the following displayed rows:

1. *O is the closed finite-observation row, recording the observed proof-facing object as finite data.*
2. *Q is the termination row, displaying the normal-form or totality evidence required before the trace is accepted.*
3. *H records componentwise* hsame *transport for observation, termination, refusal, kernel, trace, replay, provenance, and local naming rows.*
4. *C records displayed* Cont *replay from observation through termination, kernel acceptance, trace replay, refusal checks, provenance, and local naming.*
5. *P is the* Pkg *provenance over* $\mathsf{ClosedObservationProofTrace}^{\uparrow}$*, closed observation rows, total proof records, kernel acceptance witnesses,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationProofTrace}^{\uparrow}}$ *row.*
7. *R is the divergence-refusal row, rejecting any reduction row that is not displayed as terminating.*
8. *K is the kernel-acceptance witness row for the total proof record.*
9. *D is the proof-trace row, recording the finite derivation trace that is replayed by C.*

The classifier compares accepted packets only through $O, Q, H, C, P, N, R, K, D$. It has no coordinate for divergent reduction, quotient proof equality, erased proof-term identity, host derivation equality, or an unbounded computation trace.

Lean4 (definition) `BEDC.Derived.ClosedObservationProofTraceUp`

Theorem 6747.2 (Closed observation proof-trace obligations)**.** *For every accepted* $\mathsf{ClosedObservationProofTrace}^{\uparrow}$ *carrier*

$$T = (O, Q, H, C, P, N, R, K, D),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedObservationProofTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for finite observation, termination, componentwise transport, displayed replay, provenance, local naming, divergence refusal, kernel acceptance, and proof-trace rows. The packet admits no proof trace unless Q and R jointly display totality and refusal of divergent reduction before the kernel row K and trace row D are replayed.*

Proof. Project the carrier of Definition 6747.1. The observation row is O, the termination row is Q, the refusal row is R, the kernel row is K, and the trace row is D.

The totality condition is local. A trace can be replayed only after Q displays termination and R rejects the divergent alternative, so a nonterminating reduction row has no route into K or D.

The structural rows H, C, P, N transport, replay, source, and name this finite surface. Since no carrier coordinate stores quotient proof equality, erased proof identity, host derivation equality, or an unbounded computation trace, none becomes a NameCert obligation. □

Theorem 6747.3 (Closed observation proof-trace totality route)**.** *Every accepted* $\mathsf{ClosedObservationProofTrace}^{\uparrow}$ *proof record factors through*

$$O \longmapsto Q, R \longmapsto K, D \longmapsto H, C, P, N.$$

The route exposes finite observation, termination, divergent-row refusal, kernel acceptance, proof-trace replay, transport, provenance, and local naming before a proof record is accepted. It exports no divergent reduction row, quotient proof identity, proof-term erasure equivalence, host derivation equality, or unbounded trace.

Proof. Read the carrier rows. The finite observation row O is the only source row, and Q, R are the only totality and refusal gates.

The kernel row K and proof-trace row D are downstream of those gates. Thus the accepted trace is a total proof record with displayed termination and refusal data, not an erased or quotient proof object.

Finally H, C, P, N preserve the same route. A consumer that asks for an unbounded trace or a divergent reduction row leaves the accepted carrier. □

Falsifiable prediction: Any accepted $\mathsf{ClosedObservationProofTrace}^{\uparrow}$ proof trace with a divergent reduction row is rejected by Definition 6747.1, because the refusal row R and termination row Q must be displayed before kernel acceptance and proof-trace replay.

Independence witness: curryhoward, typecheckingderivationtrace, prooftermerasuretrace, kernelacceptancewitness: the packet is not bijective with any sibling because the obligation rows jointly require finite observation, termination, divergence refusal, kernel acceptance, trace replay, transport, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObservationProofTrace}^{\uparrow}}$.

Constructive story: A ClosedObservationProofTrace$^{\uparrow}$ packet is generated from finite observation rows, termination rows, divergent-row refusal, kernel acceptance, proof-trace replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosedObservationProofTrace}^{\uparrow}}$, seed)

Scope closed: Definition 6747.1, Definition 6747.2, and Definition 6747.3 seed the finite proof-trace packet over total observation records and divergence refusal.

Formal verification: FormalStatus(NameCert$_{\text{ClosedObservationProofTrace}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.ClosedObservationProofTraceUp`

Bridge status: none

Not claimed: This chapter does not close quotient proof equality, proof-term erasure equivalence, host derivation equality, divergent computation rows, Lean theorem verification, or bridge checking.

Upgrade path: The Lean follow-up must provide Nontrivial and FieldFaithful TasteGate instances after carrier admission, totality routing, divergence refusal, kernel acceptance, trace replay, transport, provenance, and local naming are formalized for BEDC.Derived.ClosedObservationProofTraceUp.

Chapter 6748

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailShift}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailShift}^{\uparrow}$ packet records the finite tail-window evidence needed to read a regular Cauchy rational sequence after discarding an initial segment. It sits between the regular-rational certificate of chapter 301, the stream-window naming surface of chapter 126, and the terminal real seal of chapter 38. The packet is only a finite means object: it names shifted windows, tolerance rows, transport, replay, provenance, and local naming, and it does not identify quotient real representatives, selected limits, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6748.1 (Regular Cauchy tail-shift carrier)**.** *A* $\mathsf{RegularCauchyTailShift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the source stream name, the discarded prefix length, the shifted request window, and the finite overlap witnesses that compare the original and shifted reads.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row, recording the regular Cauchy rational data visible on both sides of the shift and the tail index from which the two windows are read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the original window, shifted window, overlap comparison, and tail-index budget.*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal available only after W, Q, D have all been displayed.*
5. *H records componentwise* hsame *transport for the stream window, regular-rational tail data, dyadic tolerance ledger, real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a shifted-tail request through W, Q, D and then to L, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailShift}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailShift}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through W, Q, D, L, H, C, P, N. It has no coordinate for a selected limit, quotient representative equality, a host sequence equality proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailShiftUp.RegularCauchyTailShiftTasteGate single carrier alignment`

Theorem 6748.2 (Regular Cauchy tail-shift NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailShift}^{\uparrow}$ *carrier* $T = (W, Q, D, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailShift}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail data,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It also requires overlap exactness for the original and shifted windows, tail-index stability for the regular rational rows, dyadic tolerance accounting for the finite overlap comparison, and Real seal nonescape for the terminal read.*

Proof. Project the carrier of Definition 6748.1. The rows W, Q, D, L, H, C, P, N are the whole accepted inventory, so every local certificate obligation must read from those rows.

The window and tail obligations are finite. The row W displays the original and shifted stream windows, while Q displays the regular Cauchy rational tail data that both windows consume.

The numerical comparison is read through D. Its dyadic tolerance rows account for the original window, shifted window, overlap comparison, and tail-index budget before the seal L is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier row names selected limits, quotient equality, host sequence equality, or ambient completeness, the certificate cannot export any of those claims. □

Theorem 6748.3 (Regular Cauchy tail-shift RegSeqRat stability)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read that is consumed through an accepted* $\mathsf{RegularCauchyTailShift}^{\uparrow}$ *carrier is stable under the displayed tail shift. For a carrier*

$$T = (W, Q, D, L, H, C, P, N),$$

the consumer must first expose the $\mathsf{StreamName}^{\uparrow}$ *source and shifted windows in* W*, the regular rational tail row* Q*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*. Only then may it read the* $\mathsf{Real}^{\uparrow}$ *seal* L *and the structural rows* H, C, P, N*. Thus the shifted read reaches the same finite Real-facing route as the original tail read, without using selected limits, quotient representative equality, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier definition. The window row W records both the original and shifted StreamName windows, and the tail row Q is the only regular rational data available to either read.

Apply the NameCert obligations of Definition 6748.2. Overlap exactness and tail-index stability force the two visible reads to share the same finite tail surface before any Real-facing endpoint is inspected.

The dyadic row D supplies the requested tolerance comparison for the shifted window. Once that comparison is displayed, the seal L and the transport, replay, provenance, and naming rows H, C, P, N preserve the same route.

Because the packet contains no coordinate for selected limits, quotient equality, host sequence equality, or ambient Real completeness, the stability conclusion is only the finite $\mathsf{RegSeqRat}^{\uparrow}$ tail-window stability just described. □

Theorem 6748.4 (Regular Cauchy tail-shift window composition)**.** *Let* $T_0 = (W_0, Q_0, D_0, L_0, H_0, C_0, P_0, N_0)$ *and* $T_1 = (W_1, Q_1, D_1, L_1, H_1, C_1, P_1, N_1)$ *be accepted* $\mathsf{RegularCauchyTailShift}^{\uparrow}$ *packets whose shifted window of* T_0 *is the source window consumed by* T_1*. Then the composed tail-shift read still exposes only the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transports, displayed* Cont *replays,* Pkg *provenance, and local*

NameCert *rows of the two packets. The combined discard length is represented by the finite window row*

$$W_{01} = \mathsf{composeTailWindow}(W_0, W_1).$$

Thus the composite exports a finite shifted-window route and not a selected limit, quotient representative, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project both accepted carriers. The only data available for composition are the source and shifted $\mathsf{StreamName}^{\uparrow}$ rows, the regular rational tail rows, the dyadic tolerance ledgers, and the structural rows displayed by the two packets.

Apply Definition 6748.2 to both packets. The first certificate exposes the shifted window used as the source for the second, and the second certificate exposes the next shifted window over the same finite inventory.

Use Definition 6748.3 on the first shifted read and then again on the second shifted read. The composed window row W_{01} records only the finite discard-length calculation and the visible overlap witnesses.

The Real seals remain downstream of their corresponding W, Q, D rows, and the transport, replay, provenance, and naming rows only preserve the displayed route. Since neither packet contains a selected limit, quotient representative, host sequence equality, or ambient completeness coordinate, the composed route cannot export one. □

Falsifiable prediction: Every $\mathsf{RegularCauchyTailShift}^{\uparrow}$ read factors through StreamName windows, RegSeqRat tail data, a DyadicRatCore tolerance ledger, a Real seal, hsame transport, Cont replay, Pkg provenance, and local naming before a shifted regular-Cauchy tail can be consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyTailShift}^{\uparrow}$ is not bijective to any listed sibling because it binds a discarded-prefix window, regular-rational tail data, dyadic overlap tolerances, Real sealing, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailShift}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailShift}^{\uparrow}$ packet is generated from StreamName source and shifted windows, RegSeqRat tail data, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailShift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6748.1, Definition 6748.2, and Definition 6748.3 seed the finite tail-shift route used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailShift}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyTailShiftUp.RegularCauchyTailShiftTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient representative equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked window overlap exactness, tail-index stability, dyadic tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailShiftUp.

Chapter 6749

A Concrete Naming Certificate for $\mathsf{BishopCauchyNameSeal}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy reals, the $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ packet records the finite seal object that ties the L10 route from dyadic tolerances to StreamName observations, RegSeqRat readback, and the terminal Real seal. It sits over the dyadic endpoint and tolerance rows of chapter 291, the finite observation interface of chapter 126, the regular rational sequence route of chapter 301, the Real naming boundary of chapter 38, and the Bishop regular Cauchy completion surface of chapter 1639. The packet names the four-object route as a local NameCert seal; it does not assert completeness of $\mathsf{Real}^{\uparrow}$, quotient Cauchy streams, selected limits, countable choice, or host equality of real names.

Definition 6749.1 (Bishop Cauchy name-seal carrier)**.** *A* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger, recording the requested precision rows before any stream observation is read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row whose entries are inspected at the displayed dyadic tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback has been displayed.*
5. *H records componentwise* hsame *transport for D, W, R, E and the local certificate rows.*
6. *C records displayed* Cont *replay of the route from dyadic tolerances through stream observations, regular readback, and the Real seal.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyNameSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, W, R, E, H, C, P, N. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6749.2 (Bishop Cauchy name-seal four-input visibility)**.** *For every accepted* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *carrier* $S = (D, W, R, E, H, C, P, N)$*, a Real or Completion consumer must expose the four L10 rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as D, W, R, E *before the local seal can be read. The packet admits no shortcut from dyadic tolerance requests directly to a Real name, and no StreamName or RegSeqRat read is detached from the same displayed dyadic ledger.*

Proof. Project the carrier coordinates from Definition 6749.1. The only data rows before transport and replay are D, W, R, E.

The tuple order records the dependency route. The finite observation row W reads the dyadic ledger D, the regular readback row R reads D and W, and the terminal seal E is downstream of all three rows.

The structural rows H, C, P, N preserve that route. Since no additional carrier coordinate names a direct Real constructor, a detached stream, or a quotient representative, the four input rows are exactly the visible L10 surface. □

Theorem 6749.3 (Bishop Cauchy name-seal regularity handoff)**.** *Every accepted* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *packet hands finite StreamName observations to the regular rational sequence route only through*

$$D \longmapsto W \longmapsto R.$$

The handoff consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is available.*

Proof. Start with four-input visibility. It fixes D and W as the only rows from which a regular readback can be formed.

Read the regular row R. By the carrier definition, R is attached to the same dyadic tolerance ledger and finite observation window, so a consumer cannot replace W by a host stream or replace D by an unrecorded tolerance schedule.

Transport and replay through H and C move only the displayed rows. The regularity handoff is therefore the finite route $D \longmapsto W \longmapsto R$. □

Theorem 6749.4 (Bishop Cauchy name-seal Real nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *carrier factors through*

$$D \longmapsto W \longmapsto R \longmapsto E.$$

The terminal row E *is a seal for the displayed regular Cauchy route, not a proof of* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, selected limits, quotient streams, or countable choice.*

Proof. Use the regularity handoff to expose D, W, R in order. No Real-facing row is visible before the regular rational readback is displayed.

The terminal row E is the only Real seal coordinate in the carrier. It is read after R and is replayed by C with provenance and local naming supplied by P, N.

Because the carrier has no coordinate for completeness, quotient stream equality, selected limits, countable choice, or host real equality, any consumer that reaches $\mathsf{Real}^{\uparrow}$ must remain inside the displayed route. □

Theorem 6749.5 (Bishop Cauchy name-seal classifier transport)**.** *Let $S = (D, W, R, E, H, C, P, N)$ and $S' = (D', W', R', E', H', C', P', N')$ be accepted* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *carriers. If H records componentwise* hsame *transport from D, W, R, E to D', W', R', E' and C replays the displayed route, then acceptance transports from S to S' only through the rows*

$$D, W, R, E, H, C, P, N.$$

Proof. Inspect the classifier in Definition 6749.1. Its admissible inputs are the displayed data rows and the structural certificate rows.

Componentwise hsame transport compares the dyadic ledger, stream window, regular readback, Real seal, provenance, and local naming rows. Displayed Cont replay preserves the same dependency order.

No host equality or quotient relation is part of the packet. The transported acceptance therefore uses only the listed rows and cannot be widened to an external equality of real names. □

Theorem 6749.6 (Bishop Cauchy name-seal provenance exhaustion)**.** *For every accepted* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *carrier $S = (D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyNameSeal}^{\uparrow}}$ *obligations are exhausted by carrier admission, four-input visibility, regularity handoff, Real-seal nonescape, classifier transport, displayed replay, package provenance, and local naming for D, W, R, E, H, C, P, N.*

Proof. Carrier admission projects the tuple $S = (D, W, R, E, H, C, P, N)$. The preceding obligations account for visibility of D, W, R, E, the regularity handoff, Real nonescape, and classifier transport.

The remaining structural rows are C, P, N. They replay, source, and name the same finite packet and do not add mathematical data beyond the displayed route.

Thus every NameCert obligation is read from the listed rows. Nothing remains from which a consumer could infer quotient streams, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ Real-facing read factors through one DyadicRatCore endpoint and tolerance ledger, one StreamName finite observation row, one RegSeqRat regular readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-cauchy-completion: $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the four L10 rows as one local seal with transport, replay, provenance, and naming before any Real or Completion consumer may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyNameSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ packet is generated from DyadicRatCore endpoint and tolerance rows, StreamName finite observations, RegSeqRat regular readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyNameSeal}^{\uparrow}}$, seed)

Scope closed: Definition 6749.1, Definition 6749.2, Definition 6749.3, Definition 6749.4, Definition 6749.5, and Definition 6749.6 seed the finite four-object L10 name-seal route over dyadic, stream, regular, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyNameSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BishopCauchyNameSeal}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, four-input visibility, regularity handoff, Real-seal nonescape, classifier transport, displayed replay, provenance, and local naming for BEDC.Derived.BishopCauchyNameSealUp.

Chapter 6750

A Concrete Naming Certificate for $\mathsf{CompactModulusExtraction}^{\uparrow}$

Following Bishop and Bridges on compact metric continuity, the $\mathsf{CompactModulusExtraction}^{\uparrow}$ packet records the finite extraction route by which pointwise moduli over a compact finite net are folded into a displayed uniform modulus. It sits over the compact source rows of chapter 60, the continuous-map pointwise-modulus surface of chapter 130, the metric triangle rows of chapter 57, the rational lower-bound fold of chapter 37, and the output interface of chapter 369. The packet names the constructive Heine–Cantor finite-net handoff as local NameCert data; it does not add open-cover compactness, selected subcovers, host function equality, real completeness, or an external continuity theorem.

Definition 6750.1 (Compact modulus extraction carrier)**.** *A* $\mathsf{CompactModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, Q, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, listing the inspected centers, coverage radius, and finite adjacency ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to each displayed net center.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *finite lower-bound fold row, computing the positive rational tolerance budget over the displayed pointwise moduli.*
4. *T is the* $\mathsf{Met}^{\uparrow}$ *triangle row that routes nearby source points through a chosen net center and then through the target comparison.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained from K, F, Q, T.*
6. *H records componentwise* hsame *transport for K, F, Q, T, U and the local certificate rows.*
7. *C records displayed* Cont *replay from finite-net coverage through pointwise moduli, rational fold, metric triangle comparison, and uniform-modulus output.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactModulusExtraction}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, Q, T, U, H, C, P, N$. It has no coordinate for a selected open subcover, host continuity theorem, quotient function equality, arbitrary supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6750.2 (Compact modulus extraction NameCert obligations)**.** *For every accepted* CompactModulusExtra *carrier $M = (K, F, Q, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net exposure through K, pointwise-modulus exposure through F, rational lower-bound fold exposure through Q, metric triangle exposure through T, UniformModulus handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The rows K, F, Q, T, U are the only mathematical data coordinates available to a compact-uniform-continuity consumer.

The compact row supplies finite centers, the continuous-map row supplies pointwise moduli at those centers, the rational row folds the displayed positive bounds, and the metric row supplies the triangle comparison that lets the folded bound apply to a nearby source point.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. No selected subcover, host theorem, quotient function equality, arbitrary supremum, or ambient completeness row remains outside the listed obligations. □

Theorem 6750.3 (Compact modulus extraction finite-net fold)**.** *Every accepted* $\mathsf{CompactModulusExtraction}^{\uparrow}$ *packet extracts its uniform modulus only through*

$$K, F \longmapsto Q \longmapsto T \longmapsto U.$$

The rational fold Q is computed over the finite centers displayed by K and the pointwise moduli displayed by F, and the metric triangle row T is consumed before the $\mathsf{UniformModulus}^{\uparrow}$ *output row U is public.*

Proof. Start from the compact finite-net row K. It displays the finite list of centers and coverage data, so the fold is over a finite BHist ledger rather than over an ambient compactness principle.

Read F at each displayed center and then fold the resulting rational lower bounds through Q. Since Q is a $\mathsf{Rat}^{\uparrow}$ row, the consumer sees the finite rational computation rather than a selected infimum.

Finally apply the metric triangle row T and expose U. The output uniform modulus is therefore downstream of the finite net, pointwise moduli, rational fold, and triangle comparison. □

Theorem 6750.4 (Compact modulus extraction transport nonescape)**.** *Let $M = (K, F, Q, T, U, H, C, P, N)$ and $M' = (K', F', Q', T', U', H', C', P', N')$ be accepted* $\mathsf{CompactModulusExtraction}^{\uparrow}$ *carriers. If H transports the compact, pointwise-modulus, rational-fold, metric-triangle, and uniform-modulus rows componentwise, then the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row transports only through $K, F, Q, T, U, H, C, P, N$.*

Proof. Inspect the classifier of Definition 6750.1. It names only the finite-net, pointwise-modulus, rational-fold, metric-triangle, output, and structural rows.

Componentwise hsame transport moves each of those rows to its displayed counterpart, while Cont replay preserves the route from K, F through Q, T to U.

No carrier coordinate supplies host function equality, selected open covers, arbitrary suprema, or ambient completeness. The transported output is therefore the same finite extraction route and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{CompactModulusExtraction}^{\uparrow}$ uniform-modulus read factors through one compact finite-net row, one ContinuousMap pointwise-modulus row, one RatUp finite lower-bound fold row, one MetricSpace triangle row, one UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact, continuousmap, metric, rat, uniformmodulus: $\mathsf{CompactModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite compact-net rows, pointwise-modulus rows, rational lower-bound folds, metric triangle rows, and a UniformModulus handoff as one extraction packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusExtraction}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMap pointwise-modulus rows, RatUp lower-bound folds, MetricSpace triangle rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6750.1, Definition 6750.2, Definition 6750.3, and Definition 6750.4 seed the finite compact-modulus extraction route over compact, continuous-map, rational-fold, metric-triangle, UniformModulus, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host function equality, arbitrary suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactModulusExtraction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, pointwise-modulus exposure, rational lower-bound fold, metric triangle handoff, UniformModulus output, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusExtractionUp.

Chapter 6751

A Concrete Naming Certificate for $\mathsf{AlexandroffDuplicate}^{\uparrow}$

Following Alexandroff's duplicate-space construction, the $\mathsf{AlexandroffDuplicate}^{\uparrow}$ packet records the finite observable-open face of a duplicated point-set topology. It is anchored by the topology carrier of chapter 88 and the observable truth boundary of chapter 1860; the optional order-topology source is read through chapter 1890 when that row is present. The chapter names only the finite rows required for duplicate-tagged observable opens, not a host separation theorem, a completed real line, or an unrestricted lattice of open sets.

Definition 6751.1 (Alexandroff duplicate carrier)**.** *An* $\mathsf{AlexandroffDuplicate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, B, O, W, R, H, C, P, N)$$

with the following displayed rows:

1. T *is the* $\mathsf{Topology}^{\uparrow}$ *source row naming the finite open data currently under inspection.*
2. B *is the* $\mathsf{Bool}^{\uparrow}$ *duplicate-tag row; each inspected point is tagged by one of the two duplicate faces.*
3. O *is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row for the inspected tagged points.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the tagged observations actually read by the packet.*
5. R *is an optional* $\mathsf{RealOrderTopology}^{\uparrow}$ *source anchor; when absent, the carrier still records the topology row* T *and duplicate tag row* B *without importing order-completeness data.*
6. H *records componentwise* hsame *transport for topology rows, duplicate tags, observable-open truth rows, finite windows, the optional order source, continuation rows, provenance rows, and local naming rows.*
7. C *records displayed* Cont *replay from* T, B *to* O*, from* O *to* W*, and from any present order-topology anchor* R *back to the same finite topology source row.*
8. P *is the* Pkg *provenance over* $\mathsf{AlexandroffDuplicate}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{SierpinskiSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RealOrderTopology}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{AlexandroffDuplicate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, O, W, R, H, C, P, N$*. It has no coordinate for all open subsets of a host duplicate space, a separation axiom, quotient point equality, order completeness, countable choice, or a global topological representation.*

Theorem 6751.2 (Alexandroff duplicate NameCert obligations)**.** *For every accepted* $\mathsf{AlexandroffDuplicate}^{\uparrow}$ *packet* $A = (T, B, O, W, R, H, C, P, N)$*, the local NameCert surface consists exactly of carrier admission for the* $\mathsf{Topology}^{\uparrow}$ *source row,* $\mathsf{Bool}^{\uparrow}$ *duplicate tags,* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth values,* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the optional* $\mathsf{RealOrderTopology}^{\uparrow}$ *source anchor, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. A consumer may read an observable duplicate-open only after those finite rows are displayed; it cannot replace them by a host duplicate-space topology, a hidden open-set lattice, a separation theorem, or ambient real-order completeness.*

Proof. Project the carrier of Definition 6751.1. The rows $T, B, O, W, R, H, C, P, N$ are the whole accepted packet, so the duplicate-open consumer cannot detach the observable truth row from the topology source or from the duplicate tag row.

Read the finite route in order. The $\mathsf{Topology}^{\uparrow}$ row supplies the source open data, the $\mathsf{Bool}^{\uparrow}$ row supplies the duplicate face, the $\mathsf{SierpinskiSpace}^{\uparrow}$ row records the observable truth value, and the $\mathsf{StreamName}^{\uparrow}$ row bounds the inspected window.

The optional $\mathsf{RealOrderTopology}^{\uparrow}$ row only anchors the source topology when it is present. The transport, continuation, provenance, and local naming rows replay the same displayed packet, so no hidden open-set lattice, separation axiom, countable-choice schedule, or real-order completeness datum is exported. □

Falsifiable prediction: Every $\mathsf{AlexandroffDuplicate}^{\uparrow}$ observable-open read factors through $\mathsf{Topology}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows before any optional $\mathsf{RealOrderTopology}^{\uparrow}$ source anchor is consumed.

Independence witness: topology, bool, sierpinski-space, streamname, real-order-topology: $\mathsf{AlexandroffDuplicate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries topology source data, duplicate tags, observable-open truth values, finite observation windows, an optional order-topology source anchor, transports, continuations, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandroffDuplicate}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AlexandroffDuplicate}^{\uparrow}$ packet is generated from a Topology source row, Bool duplicate tags, SierpinskiSpace observable-open truth rows, StreamName finite windows, an optional RealOrderTopology source anchor, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandroffDuplicate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6751.1 and Definition 6751.2 bind the finite duplicate carrier and its observable-open NameCert obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandroffDuplicate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim a host duplicate-space topology, separation axioms, a global open-set lattice, quotient point equality, real-order completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, duplicate-tag stability, observable-open replay, finite-window readback, optional order-anchor routing, and local NameCert exactness for BEDC.Derived.AlexandroffDuplicateUp.

Chapter 6752

A Concrete Naming Certificate for $\mathsf{FormalBallSpace}^{\uparrow}$

Following Edalat and Heckmann formal-ball presentations of metric completion, the $\mathsf{FormalBallSpace}^{\uparrow}$ packet records the finite center-radius space read used by constructive completion consumers. It sits after the metric source of chapter 57, the separated-metric boundary of chapter 1411, the formal-ball domain of chapter 5723, the formal-ball completion route of chapter 2012, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite NameCert carrier for rounded formal balls over a displayed metric source; it is not a quotient completion, a selected limit, a host basis of all balls, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6752.1 (FormalBallSpace carrier)**.** *A* $\mathsf{FormalBallSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, X, R, B, O, S, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the displayed distance witness and its finite replay surface.*
2. *X is the center carrier row read over the same metric source.*
3. *R is the positive radius read, supplied through* $\mathsf{RegSeqRat}^{\uparrow}$ *or an equivalent positive rational tolerance row.*
4. *B is the formal-ball pair row, recording center-radius pairs (x, r) with x read from X and r read from R.*
5. *O is the ball-order and refinement row, displaying the accepted comparison between formal balls.*
6. *S is the rounded-shrink schedule row, recording finite radius shrinkage requests before completion consumers inspect the packet.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for finite window addresses of centers, radii, comparisons, and shrink steps.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the positive-radius and shrink tolerances to real-facing consumers.*
9. *H records componentwise* hsame *transport for metric sources, centers, positive radii, formal-ball pairs, refinement rows, shrink schedules, readback windows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from metric requests through center-radius reads, order and refinement checks, rounded shrinkage, stream windows, regular rational readback, and completion-facing handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FormalBallSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, X, R, B, O, S, W, Q, H, C, P, N$. It has no coordinate for quotient equality of completions, selected Cauchy representatives, a host family of all balls, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6752.2 (FormalBallSpace NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallSpace}^{\uparrow}$ *carrier*

$$F = (M, X, R, B, O, S, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FormalBallSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, center carrier, positive radius read, formal-ball pair row, ball-order and refinement row, rounded-shrink schedule, finite StreamName window, RegSeqRat readback, componentwise transport, displayed replay, provenance, and local naming. Positive-radius stability is read from R and Q; order and refinement stability is read from O under H, C; roundedness is read from S; completion readback passes through W, Q before any* $\mathsf{FormalBallCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer is served; and ledger nonescape keeps quotient completion equality, selected limits, host all-ball bases, countable choice, and ambient completeness outside the packet.*

Proof. Project the carrier of Definition 6752.1. Carrier admission exposes the rows $M, X, R, B, O, S, W, Q, H, C, P, N$ and no further coordinate.

Use finite row induction over the displayed packet. The center and radius rows give the formal-ball pair B; the order and refinement row O compares only those displayed pairs; and the rounded-shrink schedule S consumes finite positive-radius data before a completion-facing read is available.

Transport and replay are componentwise. The rows H and C preserve the metric source, center, positive radius, order, shrink schedule, StreamName window, and RegSeqRat readback together, while P and N source and name the same finite surface.

Finally read completion consumers through W and Q. Since the accepted carrier has no row for quotient completion equality, selected representatives, a host basis of all balls, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Falsifiable prediction: Any accepted $\mathsf{FormalBallSpace}^{\uparrow}$ completion-facing read that bypasses the positive-radius row R, the rounded-shrink schedule S, or the readback rows W, Q is rejected by Definition 6752.1, because the carrier exposes completion data only through the finite center-radius packet.

Independence witness: metric, separatedmetric, formal-ball-domain, formal-ball-completion, streamname, regseqrat, real: the packet is not bijective with any listed sibling because it jointly records metric source, center carrier, positive radius, formal-ball pair, refinement order, rounded shrinkage, finite windows, regular rational readback, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FormalBallSpace}^{\uparrow}$ packet is generated from a displayed MetricSpaceUp source, center carrier rows, positive radius reads, formal-ball pairs, ball-order and refinement rows, rounded-shrink schedules, StreamName finite

windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6752.1 and Definition 6752.2 seed the finite formal-ball space carrier over positive radii, rounded refinement, completion readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, a host all-ball basis, countable choice, ambient Real$^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius stability, order and refinement stability, rounded-shrink schedule replay, completion readback, ledger nonescape, and TasteGate field faithfulness for BEDC.Derived.FormalBallSpaceUp.

Chapter 6753

A Concrete Naming Certificate for $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$

Following Bishop and Bridges on totally bounded metric spaces and constructive completion, the $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ packet records the finite-net surface that a completion consumer may read before any completed point is sealed. It sits after the broad completion route of chapter 1573, the finite-net carrier of chapter 127, the Cauchy-filter basis of chapter 1501, the dense embedding row of chapter 1512, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate is a finite NameCert surface between total-bounded finite nets and completion readback; it is not open-cover compactness, quotient completion, selected limit construction, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6753.1 (Bishop totally bounded completion net carrier)**.** *A* $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, T, N, F, E, R, Q, H, C, P, L)$$

with the following visible rows:

1. *X is the source metric row whose distance witnesses are accepted by the total-bounded and completion-facing ledgers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, recording for each displayed positive tolerance a closed* ProbeBundle *of centers, membership rows, and rational distance bounds.*
3. *N is the Bishop refinement row. It relates a finer tolerance request to a previously displayed finite net without replacing the packet by an open cover or a selected subnet.*
4. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row generated from T and N.*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that inserts source observations into the completion boundary.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when completion names are inspected by finite rational windows.*
7. *Q records the completion-readback seal that lets a downstream* $\mathsf{Real}^{\uparrow}$ *consumer read only after F, E, R have been displayed.*
8. *H, C, P, L are, respectively, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over the cited sibling chapters, and the local* $\mathsf{NameCert}_{\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by replaying these rows. No coordinate exports open-cover compactness, quotient representatives, selected limits, countable choice, or an ambient completeness theorem.

Theorem 6753.2 (Bishop totally bounded completion net NameCert obligations)**.** *For every accepted* $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ *carrier* $B = (X, T, N, F, E, R, Q, H, C, P, L)$*, the local* NameCert *surface consists exactly of carrier admission for those rows, finite-net persistence through* T, N*, Cauchy-filter-basis generation through* F*, dense completion insertion through* E*, rational readback through* R*, completion-to-Real sealing through* Q*, and componentwise transport, replay, provenance, and local naming through* H, C, P, L*.*

Proof. Carrier admission is projection from Definition 6753.1. The listed rows are the whole visible packet, so every obligation must be read from them.

Finite-net persistence is forced by T and N. A tolerance request first reads the finite centers, membership constructors, distance witnesses, and rational bounds of T; refinement then follows the displayed Bishop row N.

The completion readback is downstream of that finite-net data. The row F builds the Cauchy-filter basis, E embeds source observations into the completion boundary, R supplies regular rational windows, and Q seals the readback for Real-facing consumers. The remaining rows only transport, replay, source, and name the same route. □

Theorem 6753.3 (Bishop totally bounded completion net Real-seal handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ *packet receives the displayed route*

$$X, T, N \longmapsto F \longmapsto E \longmapsto R \longmapsto Q \longmapsto H, C, P, L.$$

Thus the consumer reads the original total-bounded finite-net ledger before the completion and Real seal rows, and it cannot substitute open-cover compactness, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness for that route.*

Proof. Start with the finite-net side. The source row X supplies the metric witnesses, while T and N display the finite centers and refinement requests in Bishop style.

Next pass to completion readback. The basis row F is generated from the finite-net ledger, the embedding row E places source observations at the completion boundary, and the regular readback row R is the only rational-window route available before the terminal seal Q.

Finally use the transport and provenance rows. Since H, C, P, L only preserve and name the displayed packet, the Real-facing consumer has no coordinate from which to read an open cover, quotient representative, selected limit, choice schedule, or ambient completeness claim. □

Independence witness: totallyboundedcompletion, totallybounded, cauchy-filter-basis, completionembedding, regseqrat, real: $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the source metric, finite-net ledger, Bishop refinement row, Cauchy-filter basis, dense embedding, rational readback, terminal seal, transport, replay, provenance, and local naming as one completion-net packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}$ packet starts from a displayed metric source, records total-bounded finite nets and Bishop refinement rows, generates a Cauchy-filter basis, embeds source observations into a completion boundary, reads regular rational windows, seals the route for Real-facing consumers, and preserves the same finite surface by transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6753.1, Definition 6753.2, and Definition 6753.3 seed the finite Bishop completion-net route from total-bounded ledgers to completion and Real readback.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopTotallyBoundedCompletionNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient completion, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net persistence, Cauchy-filter-basis generation, dense embedding handoff, regular readback, Real-seal handoff, and ledger nonescape for the same packet.

Chapter 6754

A Concrete Naming Certificate for $\mathsf{DicksonLemma}^{\uparrow}$

Following Dickson and standard constructive finite-combinatorics, the $\mathsf{DicksonLemma}^{\uparrow}$ packet records the finite-product order induction surface used to route bad-sequence obstructions through named finite data. It sits after the unary finite-spine carrier of chapter 28, the product and list ledgers of chapter 34 and chapter 36, the ordered comparison rows of chapter 53 and chapter 54, and the algebra-facing chain rows of chapter 3637 and chapter 3472. The packet is a finite NameCert surface for product-order induction; it is not a classical selected subsequence theorem, not a quotient of infinite streams, and not a hidden Hilbert-basis proof.

Definition 6754.1 (Dickson lemma carrier)**.** *A* $\mathsf{DicksonLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{Dickson}(N, P, L, O, T, B, H, C, Q, D)$$

with the following visible rows.

- *N is the* $\mathsf{Nat}^{\uparrow}$ *coordinate row. It supplies unary finite-spine witnesses for every displayed coordinate height.*
- *P is the* $\mathsf{Prod}^{\uparrow}$ *finite-product row. It binds the displayed coordinates as one product-order packet rather than as detached projections.*
- *L is the* $\mathsf{List}^{\uparrow}$ *window row. It records the finite prefix of the candidate bad sequence inspected by the induction.*
- *O is the* $\mathsf{Poset}^{\uparrow}$ *product-order row, with coordinatewise comparison over the same natural-number classifier.*
- *T is the* $\mathsf{TotalOrder}^{\uparrow}$ *coordinate-comparison row used only inside the natural coordinate reads.*
- *B is the bounded-bad-prefix row. It records that every obstruction is witnessed by a finite list window, a coordinate, and a product-order comparison failure inside that window.*
- *H, C, Q, D are, respectively, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DicksonLemma}^{\uparrow}}$ *naming row.*

The classifier compares accepted packets by replaying exactly these rows. No coordinate exports a selected infinite subsequence, excluded-middle search over streams, quotient sequence equality, or an ambient algebraic Noetherian theorem.

Theorem 6754.2 (Dickson lemma NameCert obligations)**.** *For every accepted* $\mathsf{DicksonLemma}^{\uparrow}$ *carrier* $\mathsf{Dickson}(N, P, L, O, T, B, H, C, Q, D)$*, the local* $\mathsf{NameCert}_{\mathsf{DicksonLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, natural-coordinate access through* N*, finite-product assembly through* P*, finite-window readback through* L*, product-order comparison through* O*, coordinate total-order comparison through* T*, bounded-bad-prefix induction through* B*, componentwise transport through* H*, continuation replay through* C*, provenance nonescape through* Q*, and local naming through* D*.*

Proof. Project the carrier of Definition 6754.1. The accepted packet exposes only the rows $N, P, L, O, T, B, H, C, Q$, so the local certificate cannot read an outside stream object or an algebra theorem as an extra coordinate.

The finite-order part is read first. The row N supplies the natural-number coordinate surfaces, P binds those surfaces into a displayed product, and L records the finite list window on which the induction is performed.

The comparison part is then forced by O, T, B. The product-order row O reads comparisons coordinatewise, T supplies the natural coordinate comparison used by that read, and B records the finite obstruction window consumed by the induction. Transport, continuation replay, provenance, and local naming preserve that same finite inventory. □

Theorem 6754.3 (Dickson lemma finite-product order handoff)**.** *Every downstream consumer of an accepted* $\mathsf{DicksonLemma}^{\uparrow}$ *packet that asks for a finite-product order induction route must read the displayed handoff*

$$\mathsf{Nat}^{\uparrow} \longrightarrow \mathsf{Prod}^{\uparrow} \longrightarrow \mathsf{List}^{\uparrow} \longrightarrow \mathsf{Poset}^{\uparrow} \longrightarrow \mathsf{TotalOrder}^{\uparrow} \longrightarrow \mathsf{NameCert}_{\mathsf{DicksonLemma}^{\uparrow}}.$$

Consequently a $\mathsf{NoetherianRing}^{\uparrow}$ *or* $\mathsf{HilbertBasis}^{\uparrow}$ *consumer may cite* $\mathsf{DicksonLemma}^{\uparrow}$ *only as a finite product-order induction packet over displayed natural coordinates, finite list windows, and product-order comparisons. It cannot substitute selected infinite subsequence extraction, quotient stream equality, maximal ideal choice, or a complete Hilbert-basis proof for that route.*

Proof. Start with the natural-coordinate and finite-product rows. The row N supplies the unary natural surfaces, and P records how the displayed coordinates are assembled as one product packet.

Next read the finite window. The row L exposes only a finite list prefix, so the induction sees a bounded bad-prefix surface rather than an infinite stream or a selected subsequence.

Finally apply the order rows and the local certificate. The rows O, T, B turn the finite window into a product-order induction witness, while H, C, Q, D transport, replay, source, and name the same packet. Algebra-facing consumers may reuse this handoff as finite combinatorics, but the packet does not contain Noetherian-ring stabilization, a Hilbert-basis construction, or any choice-based extraction principle. □

Independence witness: nat, prod, list, poset, total-order, noetherian-ring, hilbert-basis: $\mathsf{DicksonLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds natural coordinates, finite product assembly, finite list windows, product-order comparison, coordinate total-order comparison, bounded bad-prefix induction, transport, replay, provenance, and local naming as one finite-product order packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DicksonLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DicksonLemma}^{\uparrow}$ packet is generated from NatUp coordinate rows, a ProdUp finite-product row, a ListUp finite-window row, PosetUp product-order comparison, TotalOrderUp coordinate comparison, a bounded-bad-prefix induction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DicksonLemma}\uparrow}$, seed)

Scope closed: Definition 6754.1, Definition 6754.2, and Definition 6754.3 seed the finite-product order induction interface over NatUp, ProdUp, ListUp, PosetUp, TotalOrderUp, and downstream NoetherianRingUp or HilbertBasisUp consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DicksonLemma}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, maximal ideal choice, Hilbert-basis proof, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, natural-coordinate access, finite-product assembly, finite-window readback, product-order comparison, bounded-bad-prefix induction, componentwise transport, provenance nonescape, and local NameCert exactness for BEDC.Derived.DicksonLemmaUp.

Chapter 6755

A Concrete Naming Certificate for $\mathsf{CechNerve}^{\uparrow}$

Following Cech, Leray, and the classical cover-nerve construction, the $\mathsf{CechNerve}^{\uparrow}$ packet records the finite simplicial object determined by a displayed finite topological cover. It sits after the topology carrier of chapter 88, the finite-radius cover ledger of chapter 452, and the finite face inventory of chapter 238. The packet names cover members, intersection witnesses, nerve faces, stability, ledger, transport, provenance, and local naming rows; it does not assert the nerve theorem, selected points in arbitrary intersections, open-cover compactness, quotient realization equality, or ambient homology computation.

Definition 6755.1 (Cech nerve carrier)**.** *A* $\mathsf{CechNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, L, U, F, I, S, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the indexed-open membership classifier used by the cover.*
2. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *or finite-cover radius row, recording the finite cover scale when the cover is served by compact or metric data.*
3. *U is the finite cover-member row, listing the displayed cover indices and their topology-open witnesses.*
4. *F is the nerve-face classifier row: a finite subset of cover indices is accepted exactly when its displayed intersection witness is present.*
5. *I is the intersection-witness ledger, recording the finite meet or overlap data consumed by F.*
6. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *row whose faces are the accepted finite nerve faces.*
7. *G is the stability row, carrying componentwise* hsame *transport for the topology, cover, intersection, nerve-face, simplicial-complex, ledger, replay, provenance, and local naming rows.*
8. *H is the nonescape ledger, refusing selected intersection points, arbitrary subcovers, host nerve realizations, quotient simplicial equality, and ambient homology computation.*
9. *C records displayed* Cont *replay from a cover request through T, L, U, I, F, S, G, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CechNerve}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CechNerve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed cover members, intersection witnesses, nerve-face classifier, simplicial-complex row, stability row, ledger row, replay, provenance, and local naming data.

Theorem 6755.2 (Cech nerve NameCert obligations)**.** *For every accepted* $\mathsf{CechNerve}^{\uparrow}$ *carrier* $K = (T, L, U, F, I, S, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CechNerve}^{\uparrow}}$ *surface consists exactly of carrier admission through* $T, L, U, F, I, S, G, H, C, P, N$*; classifier admission for the finite nerve-face row* F*; stability under componentwise transport through* G*; ledger exposure through* I, H*; replay through* C*; provenance through* P*; local naming through* N*; and nonescape for selected intersection points, arbitrary subcover choice, host nerve realizations, quotient simplicial equality, and ambient homology computation.*

Proof. Project the carrier of Definition 6755.1. The topology row T, finite-radius or finite-cover row L, cover-member row U, intersection ledger I, nerve-face classifier F, and simplicial-complex row S are the only geometric coordinates in the packet.

The classifier is finite. A face is accepted only through F, and F reads the displayed intersection witnesses in I over the listed cover members U. Thus the nerve is the finite face inventory delivered to $\mathsf{SimplicialComplex}^{\uparrow}$, not a host simplicial realization.

Stability is componentwise. The row G transports topology, cover, intersection, face, and simplicial-complex data together, while C replays the same route and P, N source and name the packet.

Finally read the ledger row H. Since the carrier has no coordinate for selected intersection points, arbitrary subcover choice, quotient realization equality, or ambient homology computation, those objects cannot be exported by an accepted $\mathsf{CechNerve}^{\uparrow}$ read. □

Falsifiable prediction: Every accepted $\mathsf{CechNerve}^{\uparrow}$ read exposes a topology row, finite cover or radius row, cover-member row, intersection-witness ledger, finite nerve-face classifier, simplicial-complex row, componentwise hsame stability, nonescape ledger, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: topology, finitelebesguenumber, simplicialcomplex, compact-cover-nerve, support-nerve-census: none is bijective to $\mathsf{CechNerve}^{\uparrow}$ because this packet jointly carries cover members, finite intersection witnesses, nerve-face classification, a simplicial-complex export row, stability, ledger, replay, provenance, and local naming as one cover-to-simplicial-object certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CechNerve}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CechNerve}^{\uparrow}}$, seed)

Scope closed: Definition 6755.1 and Definition 6755.2 seed the finite cover-to-simplicial-object certificate over TopologyUp, FiniteLebesgueNumberUp, SimplicialComplexUp, transport, ledger, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CechNerve}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the nerve theorem, selected intersection points, arbitrary subcover choice, open-cover compactness, quotient realization equality, ambient homology computation, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked cover-member admission, finite intersection-witness exposure, nerve-face classification, SimplicialComplex handoff, transport stability, ledger nonescape, replay, provenance, and local naming for BEDC.Derived.CechNerveUp.

Constructive story: A CechNerve$^{\uparrow}$ packet is generated from a TopologyUp source row, a finite cover or FiniteLebesgueNumberUp radius row, cover-member rows, intersection-witness rows, a finite nerve-face classifier, a SimplicialComplexUp export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 6756

A Concrete Naming Certificate for $\mathsf{CompactOpenExhaustion}^{\uparrow}$

Following Bishop and Bridges on constructive compactness and finite approximation, the $\mathsf{CompactOpenExhaustion}^{\uparrow}$ packet names the displayed finite-subcover exhaustion route used by the L10 completion corridor. It sits after the compact metric rows of chapter 129, the topology rows of chapter 88, the finite epsilon-net rows of chapter 1767, the stream-name rows of chapter 126, the regular rational readback rows of chapter 301, and the terminal real seal of chapter 38. The chapter records a finite exhaustion packet; it does not assert arbitrary open-cover compactness.

Definition 6756.1 (CompactOpenExhaustion carrier)**.** *A* $\mathsf{CompactOpenExhaustion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, O, F, S, W, R, E_{\mathsf{Real}^{\uparrow}}, H, C, P, N).$$

Here K *is the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* O *is the* $\mathsf{Topology}^{\uparrow}$ *open-membership ledger,* F *is the finite* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *cover row,* S *is the finite subcover schedule,* W *is the* $\mathsf{StreamName}^{\uparrow}$ *inspection window,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row,* $E_{\mathsf{Real}^{\uparrow}}$ *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached through the displayed windows,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance over the listed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenExhaustion}^{\uparrow}}$ *row.*

The carrier predicate says that every accepted read exposes the finite rows $K, O, F, S, W, R, E_{\mathsf{Real}^{\uparrow}}, H, C, P, N$ *before any exhaustion consumer may inspect a subcover claim. The packet has no coordinate for an arbitrary open cover, an infinite cover selection rule, ambient compactness, or a host completeness theorem.*

Lean4 ✓ `BEDC.Derived.CompactOpenExhaustionUp.TasteGate.CompactOpenExhaustionTasteGate single carrier alignment`

Theorem 6756.2 (CompactOpenExhaustion NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenExhaustion}^{\uparrow}$ *carrier* $E = (K, O, F, S, W, R, E_{\mathsf{Real}^{\uparrow}}, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenExhaustion}^{\uparrow}}$ *surface consists exactly of compactmetric source exposure, topology open-membership exposure, finite-net cover exposure, finite subcover scheduling, stream-window inspection, regular rational readback, terminal Real seal discipline, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6756.1. The rows $K, O, F, S, W, R, E_{\mathsf{Real}^{\uparrow}}$ display the compact metric source, open-membership ledger, finite net, finite subcover schedule, inspection windows, rational readback, and terminal Real seal.

The structural rows H, C, P, N do not add a new cover principle. They only transport the displayed rows, replay the accepted read, record provenance, and name the same local packet.

Thus the NameCert surface is the finite exhaustion packet itself, with no hidden open-cover compactness, selected infinite subcover, host topological compactness, or ambient completeness coordinate. □

Theorem 6756.3 (CompactOpenExhaustion finite-subcover route)**.** *Every* $\mathsf{CompactOpenExhaustion}^{\uparrow}$ *finite-subcover exhaustion read factors through the displayed route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, H, C, P, N.$$

The route reads K before O, reads the finite-net cover row F before the subcover schedule S, and reads the finite windows $W, R, E_{\mathsf{Real}^{\uparrow}}$ before the exhaustion claim is exported to an L10 completion consumer.

Proof. Start with Definition 6756.2. It fixes the local NameCert surface as the rows displayed by the carrier.

The finite-net row F is downstream of the compactmetric and topology rows K, O, so the subcover schedule S is read only after the finite cover entries are visible. The stream, rational readback, and Real seal rows then provide the finite inspection boundary consumed by completion-facing routes.

Finally, the transport, replay, provenance, and naming rows preserve the same packet. Therefore the exhaustion export is a finite route through the displayed BEDC rows rather than a general open-cover compactness theorem. □

Falsifiable prediction: Every accepted $\mathsf{CompactOpenExhaustion}^{\uparrow}$ read factors through displayed CompactMetricUp, TopologyUp, FiniteEpsilonNetUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and local NameCert rows before exporting a finite subcover exhaustion.

Independence witness: A purported $\mathsf{CompactOpenExhaustion}^{\uparrow}$ read that exports a subcover without the finite $\mathsf{FiniteEpsilonNet}^{\uparrow}$ row F, the finite schedule S, or the window rows $W, R, E_{\mathsf{Real}^{\uparrow}}$ is outside this certificate, even if an external topology library proves compactness by open covers.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenExhaustion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenExhaustion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6756.1, Definition 6756.2, and Definition 6756.3 seed the finite subcover exhaustion route over CompactMetricUp, TopologyUp, FiniteEpsilonNetUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenExhaustion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, infinite subcover selection, host topological compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: The next paper-side step is a scoped root-obligation theorem tying this finite exhaustion packet to its L10 completion consumer.

Constructive story: A $\mathsf{CompactOpenExhaustion}^{\uparrow}$ packet is generated from compactmetric source data, topology open-membership rows, finite-net cover entries, a finite subcover schedule, stream windows, regular rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6757

A Concrete Naming Certificate for $\mathsf{AubinNitsche}^{\uparrow}$

Following Aubin and Nitsche on finite-element duality estimates, the $\mathsf{AubinNitsche}^{\uparrow}$ packet records the finite BEDC route by which an L^2-style error read is obtained from Galerkin orthogonality, a displayed dual solve, Sobolev regularity, projection comparison, and a terminal Real error-bound row. It sits over the Galerkin projection surface of chapter 5169, the variational dual row of chapter 3419, the regularity source of chapter 440, and the scalar seal of chapter 38. The packet names only the displayed duality-error route; it does not import host Hilbert-space quotients, selected minimizers, ambient completeness, or an unrestricted PDE solution operator.

Definition 6757.1 (Aubin-Nitsche carrier)**.** *An* $\mathsf{AubinNitsche}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, L, S, H, R, D, E, T, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GalerkinProjection}^{\uparrow}$ *row, exposing the finite trial space, projection residual, and Galerkin orthogonality surface used by the error read.*
2. *L is the* $\mathsf{LaxMilgram}^{\uparrow}$ *dual row, displaying the variational dual solve available for the residual functional.*
3. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *regularity row, recording the finite regularity budget that allows the dual solution to be compared with projected test data.*
4. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *or norm-comparison row, carrying the displayed pairing and norm ledger without quotienting the ambient function space.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *error-bound row, sealing constants, radii, and tolerances used by the final scalar estimate.*
6. *D is the duality comparison row, relating the primal error read to the dual residual route through G, L, S, H.*
7. *E is the projection-error row, exposing the approximation term that bounds the dual test error before R is read.*
8. *T records componentwise* hsame *transport for G, L, S, H, R, D, E and for the replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an L^2-error request through*

$$G, L, S, H, D, E \longmapsto R \longmapsto T, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{AubinNitsche}^{\uparrow}$, $\mathsf{GalerkinProjection}^{\uparrow}$, $\mathsf{LaxMilgram}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AubinNitsche}^{\uparrow}}$ *row.*

The carrier contains no host Hilbert-space quotient, selected minimizer, ambient completeness theorem, unrestricted PDE solution operator, Lean verification row, or bridge credential.

Lean4 ✓ `BEDC.Derived.AubinNitscheUp.AubinNitscheTasteGate single carrier alignment`

Theorem 6757.2 (Aubin-Nitsche NameCert obligations)**.** *For every accepted* $\mathsf{AubinNitsche}^{\uparrow}$ *carrier*

$$A = (G, L, S, H, R, D, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AubinNitsche}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Galerkin orthogonality exposure through G, dual-solve exposure through L, Sobolev regularity exposure through S, norm or Hilbert comparison through H, projection-error comparison through E, duality comparison through D, Real-bound sealing through R, componentwise transport through T, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6757.1. The analytic rows are the Galerkin row G, dual Lax–Milgram row L, Sobolev regularity row S, norm or Hilbert comparison row H, duality comparison row D, projection-error row E, and Real-bound row R.

The Galerkin and dual rows expose the two variational inputs. The row G supplies the displayed residual-orthogonality surface, while L supplies only the dual solve named by the variational certificate.

The regularity and comparison rows then bound the dual test error. The row S records the finite Sobolev budget, H records the pairing and norm ledger, and E records the projection-error comparison before any scalar estimate is sealed.

Finally D composes the displayed duality comparison and R seals the scalar error data. The structural rows T, C, P, N transport, replay, source, and name the same route, so no hidden quotient, selected minimizer, ambient completeness theorem, unrestricted PDE solution operator, Lean verification row, or bridge credential remains available. □

Theorem 6757.3 (Aubin-Nitsche duality-error route)**.** *Every public L^2-error read exported by an accepted* $\mathsf{AubinNitsche}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{GalerkinProjection}^{\uparrow} \longmapsto \mathsf{LaxMilgram}^{\uparrow} \longmapsto \mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and through the packet rows for duality comparison, projection error, transport, replay, provenance, and local naming. The exported estimate is a finite duality-error certificate; it exports no host Hilbert-space quotient, selected minimizer, ambient completeness theorem, unrestricted PDE solution operator, quotient real equality, or countable-choice selector.

Proof. Start with a public L^2-error request. By Definition 6757.2, the request first reads the Galerkin projection row G and its residual-orthogonality surface.

The dual route is then forced through L. The dual solve is available only as the displayed Lax–Milgram row, so the consumer cannot replace it by a host solution operator or selected minimizer.

Next read S, H, E, D in order. These rows supply the Sobolev regularity budget, norm or Hilbert pairing ledger, projection-error comparison, and duality comparison needed before the scalar bound is exposed.

The Real row R is the only scalar seal in the carrier. Since T, C, P, N preserve the same rows and no forbidden coordinate appears in the carrier, the public error read remains inside the displayed finite route. □

Falsifiable prediction: Every $\mathsf{AubinNitsche}^{\uparrow}$ error read factors through displayed GalerkinProjectionUp, LaxMilgramUp, SobolevUp, HilbertUp, RealUp, duality comparison, projection-error, BHist, hsame, Cont, Pkg, and NameCert rows before any public estimate is exported.

Independence witness: galerkin-projection, lax-milgram, sobolev, hilbert, real: $\mathsf{AubinNitsche}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Galerkin orthogonality, a variational dual solve, Sobolev regularity, norm or Hilbert comparison, projection-error comparison, Real scalar sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AubinNitsche}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AubinNitsche}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6757.1, Definition 6757.2, and Definition 6757.3 seed the finite duality-error route over Galerkin, Lax–Milgram, Sobolev, norm or Hilbert, projection, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AubinNitsche}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert-space quotients, selected minimizers, ambient completeness, unrestricted PDE solution operators, quotient real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Galerkin orthogonality exposure, dual-solve exposure, Sobolev regularity exposure, projection-error comparison, Real-bound sealing, componentwise transport, replay, provenance, and local naming for BEDC.Derived.AubinNitscheUp.

Constructive story: An $\mathsf{AubinNitsche}^{\uparrow}$ packet is generated from a Galerkin projection row, LaxMilgram dual row, Sobolev regularity row, norm or Hilbert comparison row, Real error-bound row, duality comparison row, projection-error row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6758

A Concrete Naming Certificate for $\mathsf{CantorTernaryExpansion}^{\uparrow}$

Following Cantor's middle-third construction and the Bishop–Bridges discipline for real names, the $\mathsf{CantorTernaryExpansion}^{\uparrow}$ packet records a finite ternary-prefix presentation of Cantor-set membership before any downstream real consumer reads it as a $\mathsf{Real}^{\uparrow}$ name. It sits beside the retained-set surface of chapter 1530, the nested-interval route of chapter 320, the dyadic interval arithmetic of chapter 306, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only a displayed finite-window route from ternary digits to a real seal; it does not import quotient real equality, choice of an infinite branch, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6758.1 (Cantor ternary expansion carrier)**.** *A* $\mathsf{CantorTernaryExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, D, I, K, W, R, E, H, C, P, L)$$

with the following rows:

1. *A is the finite ternary prefix row, listing the inspected digits and their prefix length.*
2. *D is the digit-compatibility row requiring every displayed digit to be 0 or 2 and recording the excluded middle-third alternative.*
3. *I is the retained interval row read through* $\mathsf{CantorSet}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*, and* $\mathsf{DyadicInterval}^{\uparrow}$ *siblings.*
4. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row comparing ternary prefix intervals with dyadic observation budgets.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite digit stream segment consumed by the real-name read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same prefix window and tolerance budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the prefix, tolerance, stream, and rational readback rows have been displayed.*
8. *H records componentwise* hsame *transport for A, D, I, K, W, R, E and the local certificate rows.*
9. *C records displayed* Cont *replay from ternary-prefix requests through interval, tolerance, stream, rational, and Real-seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorTernaryExpansion}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{CantorTernaryExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a completed infinite ternary choice sequence, host subset equality, quotient real equality, selected branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6758.2 (Cantor ternary expansion NameCert obligations)**.** *For every accepted* $\mathsf{CantorTernaryExpansion}^{\uparrow}$ *carrier*

$$T = (A, D, I, K, W, R, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CantorTernaryExpansion}^{\uparrow}}$ *surface consists exactly of ternary prefix data A, digit compatibility D, retained interval data I,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance K,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance P, and local naming L.*

Proof. Project the carrier of Definition 6758.1. The only digit data are the finite prefix row A and the compatibility row D, so the packet cannot read an unbounded ternary branch before the displayed window has been named.

The retained interval row I is then read through the Cantor-set, nested-interval, and dyadic-interval siblings. The dyadic tolerance row K, stream window W, regular rational readback R, and terminal Real seal E turn that interval route into a finite real-name observation.

Finally H, C, P, L transport, replay, source, and name the same rows. Since the carrier contains no branch selector, quotient real equality, host closed-set equality, or ambient completeness row, the NameCert surface is exhausted by the displayed finite-window route. □

Theorem 6758.3 (Cantor ternary expansion Real seal route)**.** *Every Real-facing read of an accepted* $\mathsf{CantorTernaryExpansion}^{\uparrow}$ *carrier must factor in order through the ternary prefix row A, digit-compatibility row D, retained interval row I, dyadic tolerance row K, stream window W, regular rational readback R, and terminal Real seal E before the structural rows H, C, P, L are consumed. The route strengthens the* $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *frontier without importing quotient real equality or choice.*

Proof. Use Definition 6758.2. It places the prefix and digit rows before the interval, tolerance, stream, rational, and Real rows.

The interval row reads the Cantor-set and nested-interval siblings as finite retained intervals, while the dyadic interval and dyadic rational rows provide the tolerance comparison needed by the stream and rational readback.

The Real seal is therefore available only after A, D, I, K, W, R have been displayed. The structural rows H, C, P, L preserve and name that route, so a Real-facing consumer receives a finite NameCert path rather than a selected infinite ternary branch or a quotient real equality proof. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorTernaryExpansion}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorTernaryExpansion}^{\uparrow}$ packet is generated from ternary prefix data, digit-compatibility rows, retained interval rows, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorTernaryExpansion}^{\uparrow}}$, seed)

Scope closed: Definition 6758.1, Definition 6758.2, and Definition 6758.3 seed the finite ternary-prefix real-name route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorTernaryExpansion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected infinite ternary branches, Cantor-set compactness, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked prefix, digit, interval, tolerance, stream, rational readback, Real-seal, transport, replay, provenance, and local naming rows for BEDC.Derived.CantorTernaryExpansionUp.

Chapter 6759

A Concrete Naming Certificate for $\mathsf{DyadicWindowReflector}^{\uparrow}$

Following Bishop–Bridges real-name completeness arguments, the $\mathsf{DyadicWindowReflector}^{\uparrow}$ packet records the finite-window reflector that turns displayed dyadic tolerance requests into the StreamName, RegSeqRat, and Real rows consumed by Real-completeness proofs. It realizes the finite-window classifier route of the Real-completeness vision through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows. The reflector consumes the dyadic tolerance, StreamName observation-window, RegSeqRat readback, and terminal Real seal surfaces through chapter 291, chapter 126, chapter 301, and chapter 38 before any completion-facing read is admitted. The packet names only the finite reflector surface; it does not assert selected limits, quotient Cauchy equality, open-ended choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6759.1 (Dyadic window reflector carrier)**.** *A* $\mathsf{DyadicWindowReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, W, R, E, Q, H, C, P, L)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget row, including the inspected dyadic radius and comparison window.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row addressed by the same dyadic budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W under the displayed dyadic tolerance.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the reflected finite window.*
5. *Q is the classifier row recording that the Real-completeness query reads only the displayed dyadic, stream, rational, and Real coordinates.*
6. *H records componentwise* hsame *transport for D, W, R, E, Q and the local certificate rows.*
7. *C records displayed* Cont *replay from a finite-window completeness request through the dyadic, stream, rational, and Real-seal route.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicWindowReflector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{DyadicWindowReflector}^{\uparrow}}$ *row.*

The classifier has no coordinate for a selected limiting real, quotient stream equality, hidden Cauchy class, or unrestricted completeness theorem.

Theorem 6759.2 (Dyadic window reflector NameCert obligations)**.** *For every accepted* $\mathsf{DyadicWindowReflector}^{\uparrow}$ *carrier*

$$F = (D, W, R, E, Q, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicWindowReflector}^{\uparrow}}$ *surface consists exactly of the dyadic tolerance budget* D*, stream window* W*, regular rational readback* R*, terminal Real seal* E*, finite-window classifier row* Q*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* L*.*

Proof. Project the carrier of Definition 6759.1. The dyadic tolerance row D is the only numerical budget available to the reflector, and the stream row W is the only window on which that budget acts.

The rational readback R and Real seal E are downstream of the same displayed window. The classifier row Q records that Real-completeness consumers see exactly this finite-window route rather than a hidden limiting object.

The structural rows H, C, P, L preserve, replay, source, and name the route. Thus the local certificate cannot export selected limits, quotient stream equality, hidden Cauchy classes, or ambient Real completeness beyond the displayed rows. □

Theorem 6759.3 (Dyadic window reflector vision route)**.** *Every finite-window classifier read used by the Real-completeness route factors through* $D, W, R, E, Q, H, C, P, L$ *in an accepted* $\mathsf{DyadicWindowReflector}^{\uparrow}$ *carrier. In particular, the vision-side* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *passages are realized as a concrete NameCert packet over dyadic budgets, stream windows, rational readback, Real seal rows, transport, replay, provenance, and local naming.*

Proof. Apply Definition 6759.2. It fixes the reflector surface as the rows $D, W, R, E, Q, H, C, P, L$.

The dyadic budget row D addresses the observation window W, and the readback row R converts that window into the rational tolerance data required by the terminal Real seal E. The classifier row Q is therefore the finite-window classifier seen by the Real-completeness route.

Finally H, C, P, L keep the classifier stable under componentwise transport and displayed replay. The reflector is consequently a concrete NameCert route for later diagonal-limit formalization, not a selected-limit or quotient-completion principle. □

Theorem 6759.4 (Dyadic window reflector DyadicRatCore route)**.** *Every* $\mathsf{DyadicWindowReflector}^{\uparrow}$ *read first consumes its numerical tolerance budget through chapter 291 before it can open the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows. Thus the sibling route factors through chapter 291, chapter 126, chapter 301, and chapter 38.*

Proof. Project the carrier rows $D, W, R, E, Q, H, C, P, L$ of Definition 6759.1. The row D is the $\mathsf{DyadicRatCore}^{\uparrow}$ source of the inspected radius and comparison window, so it is the only numerical budget available to the reflector.

Only after D is fixed can the stream row W be addressed, the rational readback row R be computed, and the Real seal row E be exposed. The structural rows Q, H, C, P, L preserve this order: the classifier sees the dyadic, stream, rational, and Real coordinates; transport and replay keep those coordinates componentwise; provenance and local naming record the same route.

Without the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row supplied by chapter 291, the reflector has no displayed numerical budget with which to select or compare StreamName windows. Hence its sibling dependency route must pass through DyadicRatCore before the StreamName, RegSeqRat, and Real rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicWindowReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicWindowReflector}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance budget, StreamName windows, RegSeqRat readback, a Real seal, a finite-window classifier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicWindowReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6759.1, Definition 6759.2, Definition 6759.3, and Definition 6759.4 seed the Real-completeness finite-window reflector route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicWindowReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient Cauchy equality, unrestricted Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic-budget, stream-window, rational-readback, Real-seal, classifier, transport, replay, provenance, and local naming rows for BEDC.Derived.DyadicWindowReflectorUp.

Chapter 6760

A Concrete Naming Certificate for $\mathsf{FanBarInduction}^{\uparrow}$

Following Brouwer and Heyting, the $\mathsf{FanBarInduction}^{\uparrow}$ packet records the finite induction surface available for bars on the binary fan before any open-cover compactness, branch selection, quotient stream equality, countable choice, or ambient Real-completeness claim is read. It sits after the finite fan certificate of chapter 400, the Cantor-space prefix carrier of chapter 1475, the barred-spread replay of chapter 1675, and the Bool/List prefix carriers of chapter 32 and chapter 36. The packet names a finite bar-induction handoff for downstream compact fan-modulus and total-bounded routes; it is not an unrestricted induction principle over arbitrary spreads.

Definition 6760.1 (Fan bar induction carrier)**.** *A* $\mathsf{FanBarInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, I, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CantorSpace}^{\uparrow}$ *binary-prefix tree row, retaining exactly the visible* $\mathsf{Bool}^{\uparrow}$ *endpoint rows and* $\mathsf{List}^{\uparrow}$ *prefix spines under inspection;*
2. *B is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row, recording decidable bar decisions and the unary depth ledger for the retained fan window;*
3. *I is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *local backwards-replay row, restricted to displayed children of T and barred prefixes of B;*
4. *W is the finite witness window listing the prefixes where the replay reads bar data, successor rows, and terminal decisions;*
5. *Q is the prefix-closure ledger, asserting that every prefix of a displayed witness prefix is again a displayed* $\mathsf{List}^{\uparrow}$ *row with its* $\mathsf{Bool}^{\uparrow}$ *endpoint data still visible;*
6. *H stores componentwise* hsame *transport for the tree, bar, induction replay, witness-window, prefix-closure, continuation, provenance, and local naming rows;*
7. *C records displayed* Cont *replay for root inspection, left and right successor inspection, prefix restriction, and induction replay;*
8. *P is the* Pkg *provenance over* $\mathsf{FanBarInduction}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{FanBarInduction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these finite rows. It has no coordinate for selected infinite branches, open-cover compactness, quotient stream equality, countable choice, arbitrary spread induction, or ambient Real completeness.

Theorem 6760.2 (Fan bar induction NameCert obligations)**.** *For every accepted* $\mathsf{FanBarInduction}^{\uparrow}$ *carrier* $F = (T, B, I, W, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FanBarInduction}^{\uparrow}}$ *surface consists exactly of carrier admission for the binary-prefix tree, finite-bar row, local backwards-replay row, witness window, prefix-closure ledger, componentwise transport, displayed replay, package provenance, and local name; finite fan-bar exactness requiring B and W to expose the same retained prefixes; induction-replay exactness requiring I to use only displayed successors of T; prefix-closure exactness requiring Q to keep every inspected prefix inside the Bool/List carrier surface; and non-escape of open-cover compactness, branch selection, quotient stream equality, countable choice, arbitrary spread induction, or ambient Real completeness.*

Proof. Project the carrier of Definition 6760.1. Carrier admission exposes exactly $T, B, I, W, Q, H, C, P, N$, so every certificate read starts with the finite binary-prefix, bar, replay, window, closure, transport, continuation, provenance, and naming rows.

Read finite fan-bar exactness through chapter 400. The row B supplies the decidable bar decisions and depth ledger, while W lists the prefixes actually inspected.

Read the backwards-replay row through chapter 1675. Because I is restricted to the displayed children of T, the replay remains local to the retained Cantor-space prefix tree and does not select an infinite branch.

Finally use the Bool/List carrier route of chapter 32 and chapter 36 for Q. Prefix closure keeps endpoint and spine data inside finite carriers, and the structural rows H, C, P, N preserve the same route without adding compactness, quotient, choice, arbitrary induction, or Real-completeness coordinates. □

Theorem 6760.3 (Fan bar induction finite prefix closure)**.** *Let $F = (T, B, I, W, Q, H, C, P, N)$ be an accepted* $\mathsf{FanBarInduction}^{\uparrow}$ *carrier. Every accepted replay of the fan-bar induction row is prefix-closed in the displayed window: if a replay reads a barred prefix from W, then each earlier prefix of that same* $\mathsf{List}^{\uparrow}$ *spine is read through Q, with its* $\mathsf{Bool}^{\uparrow}$ *endpoint rows and its* $\mathsf{CantorSpace}^{\uparrow}$ *tree coordinate still visible. Thus a downstream compact fan-modulus or total-bounded consumer may consume a finite prefix-closed induction handoff, but it receives no selected branch, open-cover compactness theorem, quotient stream equality, countable-choice functional, arbitrary spread-induction rule, or ambient Real-completeness claim.*

Proof. Start with the NameCert obligations of Definition 6760.2. They require every induction replay to expose T, B, I, W, Q before the structural rows are consumed.

The window row W identifies the displayed barred prefixes. The prefix-closure ledger Q then reads each earlier prefix through the same $\mathsf{List}^{\uparrow}$ spine and the same $\mathsf{Bool}^{\uparrow}$ endpoint discipline, so the replay never leaves the finite Cantor-space tree row T.

Transport and replay through H and C preserve those displayed prefix rows, while P and N source and name the same finite packet. Since the accepted carrier has no coordinate for branch selection, open covers, quotient streams, countable choice, arbitrary spread induction, or Real completeness, the finite prefix-closed handoff exports none of those objects. □

Falsifiable prediction: An accepted $\mathsf{FanBarInduction}^{\uparrow}$ read is refuted if a consumer reaches a fan-bar induction replay without first displaying the finite fan-bar row, Cantor-space prefix tree, barred-

spread replay, Bool/List prefix rows, prefix-closure ledger, transport row, replay row, provenance row, and local naming row.

Independence witness: fan theorem, Cantor space, Brouwer bar induction, Bool endpoints, and List prefixes: none is bijective with $\mathsf{FanBarInduction}^{\uparrow}$ because the packet jointly carries finite fan-bar decisions, binary-prefix rows, local backwards replay, prefix-closure data, structural transport, displayed continuation, package provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FanBarInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{FanBarInduction}^{\uparrow}$ packet is generated from a finite FanTheorem bar row, a CantorSpace prefix tree, a BrouwerBarInduction local backwards replay, Bool endpoint rows, List prefix spines, a prefix-closure ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanBarInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6760.1, Definition 6760.2, and Definition 6760.3 seed the finite fan-bar induction carrier over finite fan bars, Cantor-space prefixes, barred-spread replay, Bool/List prefix rows, prefix closure, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanBarInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, selected infinite branches, quotient stream equality, countable choice, arbitrary spread induction, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite fan-bar exactness, local backwards-replay exactness, prefix-closure exactness, structural transport, displayed replay, provenance, local naming, and non-escape for BEDC.Derived.FanBarInductionUp.

Chapter 6761

A Concrete Naming Certificate for $\mathsf{CauchyApproximationModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyApproximationModulus}^{\uparrow}$ packet records the finite modulus surface that tells a constructive Cauchy approximation route which tail window answers a requested dyadic tolerance. It sits between the dyadic tolerance ledger of chapter 291, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies downstream Real and Completion consumers with explicit request-to-tail data only; it does not identify quotient streams, choose a limit representative, use host real equality, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6761.1 (Cauchy approximation modulus carrier)**.** *A* $\mathsf{CauchyApproximationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, D, W, R, S, T, C, P, N)$$

with the following displayed rows:

1. *Q is the requested dyadic tolerance row.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger that reads the tolerance as a finite rational budget.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row naming the first finite observation window accepted for that budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and dyadic budget.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, D, W, R have been exposed.*
6. *T records componentwise* hsame *transport for tolerance, dyadic ledger, tail window, regular readback, seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through D, W, R, S, T.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, W, R, S, T, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6761.2 (Cauchy approximation modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyApproximationModulus}^{\uparrow}$ *carrier* $M = (Q, D, W, R, S, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, dyadic tolerance exactness, tail-window exactness, regular readback handoff, Real seal nonescape, componentwise transport, continuation replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6761.1. The tolerance row Q is read first, and the dyadic ledger D is the only row allowed to translate that request into a finite rational comparison budget.

The tail-window row W then supplies the finite $\mathsf{StreamName}^{\uparrow}$ observations used by the request. The regular readback row R and the terminal seal row S are downstream of that displayed window, so the route cannot skip directly to a host real value or selected limit.

The structural rows T, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is available, the local certificate has exactly the stated obligations and no quotient, choice, trichotomy, or ambient-completeness row. □

Theorem 6761.3 (Cauchy approximation modulus window stability)**.** *Let* $M = (Q, D, W, R, S, T, C, P, N)$ *be an accepted* $\mathsf{CauchyApproximationModulus}^{\uparrow}$ *packet. Any transported read of the same dyadic tolerance must return through the displayed* hsame *row* T *to the same tail-window, regular-readback, and Real-seal route. Thus downstream regular-Cauchy and* $\mathsf{Real}^{\uparrow}$ *consumers receive a stable finite window basis for that tolerance, not a quotient stream equality or host completeness claim.*

Proof. Project the accepted carrier. A tolerance read exposes Q, and the dyadic comparison budget is fixed by D before any stream window can be used.

Transported reads move componentwise through T. The continuation row C replays the route from Q through D, W, R, S, so the consumer returns to the same displayed tail window and regular readback before seeing the terminal seal.

The packet has no coordinate for a second stream representative, a selected limit, a host equality proof, or a completeness theorem. Stability is therefore the stability of the finite displayed window route itself. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationModulus}^{\uparrow}$ read used by a Real or Completion consumer factors through one displayed dyadic tolerance, one DyadicRatCore comparison budget, one StreamName tail-window row, one RegSeqRat readback row, one RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchyApproximationModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a tolerance request, dyadic comparison budget, accepted tail window, regular readback, terminal Real seal, transport, replay, provenance, and local naming in one modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationModulus}^{\uparrow}$ packet is generated from a dyadic tolerance request, DyadicRatCore comparison budget, StreamName tail window, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyApproximationModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6761.1, Definition 6761.2, and Definition 6761.3 seed the finite request-to-tail modulus route for the L10 Real and Completion spine.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyApproximationModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, host real equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, tail-window exactness, regular readback handoff, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationModulusUp.

Chapter 6762

A Concrete Naming Certificate for $\mathsf{RegularCauchyBallBasis}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ packet records the finite tail-ball basis used to inspect a regular Cauchy name before it is handed to a Real-completion seal. It combines the dyadic radius rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal boundary of chapter 38. The packet is a finite basis of tail thresholds and dyadic balls; it is not a quotient completion, a selected limit, a host metric completeness theorem, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 6762.1 (Regular Cauchy ball basis carrier)**.** *A* $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, D, O, R, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite tail-threshold row naming the indices after which the regular Cauchy read is inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ball-radius ledger attached to those thresholds.*
3. *O is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row opened for the threshold and radius data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from T, D, O.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed tail-ball basis has been read.*
6. *H records componentwise* hsame *transport for thresholds, dyadic balls, observation windows, regular readbacks, seals, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through T, D, O, R, S, H.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBallBasis}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBallBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, D, O, R, S, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, selected representatives, host metric completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6762.2 (Regular Cauchy ball basis NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ *carrier* $B = (T, D, O, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBallBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, threshold exactness, dyadic ball exactness, finite observation-window exposure, regular readback handoff, Real seal nonescape, componentwise transport, continuation replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6762.1. The threshold row T names the tail being inspected, and the dyadic ledger D names the finite ball radius attached to that tail.

The observation row O opens only the finite $\mathsf{StreamName}^{\uparrow}$ windows demanded by T and D. The regular readback R and terminal seal S remain downstream of those windows.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. Since the classifier has no coordinate beyond these rows, the local NameCert surface has exactly the stated obligations and no quotient, selected-limit, choice, metric-completeness, or ambient-completeness row. □

Theorem 6762.3 (Regular Cauchy ball basis tail extraction)**.** *Every downstream Real-completeness consumer of an accepted* $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ *carrier extracts its finite tail-ball basis through the route*

$$T, D, O \longmapsto R, S \longmapsto H, C, P, N.$$

The extracted basis supplies tail thresholds, dyadic balls, finite stream windows, regular readback, and a terminal Real seal, but it supplies no quotient completion, selected representative, host metric limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Proof. Project the accepted carrier. The only rows that can serve as a basis are the threshold row T, the dyadic ball ledger D, and the finite observation row O.

The regular readback R consumes that finite basis before the terminal seal S is visible. Componentwise transport and replay through H and C return the consumer to the same threshold, ball, window, readback, and seal rows.

The provenance and local naming rows P, N name exactly that route. Because the carrier contains no quotient, selected representative, host metric limit, choice schedule, or ambient completeness coordinate, the extracted object is the displayed finite basis and nothing stronger. □

Falsifiable prediction: Every $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ Real-completeness read factors through finite tail thresholds, a DyadicRatCore ball-radius ledger, StreamName observation windows, RegSeqRat readback, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries tail thresholds, dyadic ball radii, finite observation windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming as one basis packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBallBasis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ packet is generated from finite tail thresholds, a DyadicRatCore ball-radius ledger, StreamName observation windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBallBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6762.1, Definition 6762.2, and Definition 6762.3 seed the finite tail-ball basis consumed by Real-completeness routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBallBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, host metric completeness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, dyadic ball exactness, finite observation-window exposure, regular readback handoff, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBallBasisUp.

Chapter 6763

A Concrete Naming Certificate for $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$

Following Cauchy and the constructive calculus tradition, the $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ packet records the finite comparison route behind the Cauchy mean value theorem. It sits beside the calculus root of chapter 3632, the derivative packet of chapter 122, and the real-name seal of chapter 38. The packet packages endpoint rows, an interior witness row, derivative comparison data, dyadic tolerance control, regular rational readback, and the terminal Real seal without importing a host Rolle theorem, a quotient function space, or an ambient completeness principle.

Definition 6763.1 (Cauchy mean value theorem carrier)**.** *A* $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, I, D, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint row, naming the two endpoint evaluations and the finite interval certificate read by the calculus packet.*
2. *I is the interior witness row, naming the displayed interior point at which the comparison is read.*
3. *D is the* $\mathsf{Derivative}^{\uparrow}$ *comparison row for the two derivative-facing reads.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the finite-difference comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets the endpoint and derivative comparison through regular rational windows.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after E, I, D, Q, R have been exposed.*
7. *H records componentwise* hsame *transport for endpoints, the interior witness, derivative comparison, tolerance ledger, readback, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from endpoint rows through the interior witness, derivative comparison, tolerance ledger, readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMeanValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, I, D, Q, R, S, H, C, P, N$. It has no coordinate for host differentiability, quotient equality of functions, a selected global extremum, classical choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6763.2 (Cauchy mean value theorem NameCert obligations)**.** *For every accepted* CauchyMeanValueTheo *carrier*

$$M = (E, I, D, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMeanValueTheorem}^{\uparrow}}$ *surface consists exactly of endpoint exposure, interior witness exposure, derivative comparison exposure, dyadic tolerance exactness, regular rational readback, Real seal nonescape, componentwise transport, continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6763.1. The endpoint row E, interior witness row I, derivative comparison row D, tolerance ledger Q, regular readback row R, and Real seal S are the complete analytic inventory.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. A consumer can therefore move the packet and replay the comparison, but it returns to the displayed endpoint, witness, derivative, tolerance, readback, and seal rows.

Since the packet has no additional coordinate, the local NameCert cannot export a host differentiability theorem, quotient function equality, selected global extremum, classical-choice step, or ambient Real-completeness claim. □

Theorem 6763.3 (Cauchy mean value finite-difference route)**.** *Every downstream calculus consumer of an accepted* $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ *packet reads the comparison route through*

$$E, I, D, Q \longmapsto R, S \longmapsto H, C, P, N.$$

Thus the Cauchy mean value conclusion is exposed as finite endpoint data, an interior witness row, derivative comparison data, dyadic tolerance control, regular rational readback, and a Real seal; it is not exposed as a host Rolle theorem, quotient-function equality, selected extremum principle, or ambient completeness theorem.

Proof. Read the endpoint row E and the interior witness row I. These rows name the finite interval endpoints and the displayed interior point before any derivative comparison is visible.

Next read the derivative comparison row D together with the dyadic tolerance ledger Q. The regular rational readback R and terminal Real seal S consume those finite rows, so the calculus conclusion is reached through displayed finite-difference data.

Finally use H, C, P, N for transport, replay, provenance, and local naming. Because the carrier contains no row for a host Rolle theorem, quotient-function equality, selected extremum principle, or ambient completeness theorem, every downstream read remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMeanValueTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ packet is generated from endpoint rows, an interior witness row, a derivative comparison row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyMeanValueTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 6763.1, Definition 6763.2, and Definition 6763.3 seed the finite L10 Cauchy mean value route over endpoint, interior-witness, derivative-comparison, dyadic-tolerance, regular-readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyMeanValueTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host Rolle theorem, quotient equality of functions, a selected extremum principle, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, interior witness exposure, derivative comparison exposure, tolerance exactness, readback handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMeanValueTheoremUp.

Chapter 6764

A Concrete Naming Certificate for $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$

Following Runge and Kutta on finite-stage ordinary differential equation methods and Bishop–Bridges constructive real analysis, the $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ packet records the finite stage data needed to inspect one displayed ODE step. It sits beside the ODE carrier of chapter 193, the Picard–Lindelof finite-approximation discipline of chapter 1888, the dyadic arithmetic rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names finite stage equations over displayed data; it does not assert host ODE existence, selected global solutions, quotient equality of names, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6764.1 (Runge–Kutta finite-step carrier)**.** *A* $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, F, A, S, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the displayed ODE row read from chapter 193, including the state, time, and right-hand-side interface inspected by the step.*
2. *F is the finite stage-function row naming the stage evaluation scheme used by the Runge–Kutta tableau.*
3. *A is the tableau row containing the finitely many stage coefficients, weights, and nodes.*
4. *S is the dyadic step-size row read through chapter 291.*
5. *D is the dyadic arithmetic ledger for coefficient products, stage update terms, and final weighted sums.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened for the source state, time, and stage arguments.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the computed step as regular rational data.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the ODE, stage, tableau, dyadic, window, and readback rows have been displayed.*
9. *H records componentwise* hsame *transport for the ODE row, stage row, tableau row, step-size row, dyadic ledger, stream window, readback row, Real seal, replay, provenance, and local name.*

10. *C records the displayed* Cont *replay for finite stage evaluation, dyadic accumulation, stream-window inspection, regular readback, and Real seal handoff.*
11. *P is the* Pkg *provenance over the ODE, Picard–Lindelof, dyadic, stream, regular rational, Real, transport, replay, and NameCert rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RungeKuttaFiniteStep}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the rows $O, F, A, S, D, W, R, E, H, C, P, N$.

Theorem 6764.2 (Runge–Kutta finite-step NameCert obligations)**.** *For the carrier of Definition 6764.1, the local* $\mathsf{NameCert}_{\mathsf{RungeKuttaFiniteStep}^{\uparrow}}$ *obligation surface consists exactly of ODE-row exposure through O, finite stage-function exposure through F, tableau exposure through A, dyadic step-size exposure through S, dyadic arithmetic through D, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier $K = (O, F, A, S, D, W, R, E, H, C, P, N)$. The analytic rows are displayed before the structural rows, and the classifier reads no coordinate outside this finite list.

Use the ODE and Picard–Lindelof neighbor chapters only as source rows for a displayed finite step. The stage-function and tableau rows record a finite evaluation scheme, while S, D, W provide the dyadic and stream-window data needed to compute the inspected step.

The readback and seal rows are downstream of those finite rows. Since H, C, P, N transport, replay, source, and name the same packet, they do not add a host existence theorem, selected solution, quotient equality, or Real completeness principle. □

Theorem 6764.3 (Runge–Kutta finite-step RegSeqRat handoff)**.** *Every accepted* $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ *carrier that exposes the readback row R must first display the ODE row O, stage-function row F, tableau row A, dyadic step-size row S, dyadic arithmetic ledger D, and finite* $\mathsf{StreamName}^{\uparrow}$ *window W. Thus the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is a finite step readback over displayed stage data, not a selected solution operator.*

Proof. Read the carrier definition. The row R is the first regular rational output row, and it is placed after the ODE, stage, tableau, dyadic, and stream-window rows.

The dyadic ledger D is where coefficient products, stage update terms, and weighted sums are displayed. The stream window W supplies the finite source data read by those dyadic calculations.

Therefore a consumer cannot read R as a free-standing regular rational name. It is available only as the handoff of the finite Runge–Kutta stage calculation over O, F, A, S, D, W. □

Theorem 6764.4 (Runge–Kutta finite-step Real-seal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row E of an accepted* $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ *carrier cannot be reread as an ambient ODE solution, a selected flow map, quotient equality of solution names, or completeness of* $\mathsf{Real}^{\uparrow}$*. Any route to E still factors through the displayed rows O, F, A, S, D, W, R.*

Proof. Project the carrier. The seal row E is terminal for one finite-step packet and is reached only after the readback row R.

Apply the RegSeqRat handoff theorem. It forces R to factor through the ODE, stage, tableau, dyadic step-size, dyadic arithmetic, and stream-window rows before any Real-facing seal is exposed.

The structural rows H, C, P, N preserve that route under componentwise transport and replay. They do not contain data for a global solution, a selected flow, quotient equality, or a completeness principle, so the seal cannot escape the finite-step boundary. □

Falsifiable prediction: A $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ consumer that reaches R or E without displaying the ODE row, finite stage-function row, tableau row, dyadic step-size row, dyadic arithmetic ledger, and finite $\mathsf{StreamName}^{\uparrow}$ window refutes the carrier surface.

Independence witness: ode, picard-lindelof, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (O, F, A, S, D, W, R, E, H, C, P, N)$, because this chapter binds finite ODE stage data, a Runge–Kutta tableau, dyadic step arithmetic, stream windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming in one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RungeKuttaFiniteStep}^{\uparrow}}$.

Constructive story: A $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$ packet is generated from a displayed ODE row, finite stage-function row, tableau row, dyadic step-size row, dyadic arithmetic ledger, finite stream window, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RungeKuttaFiniteStep}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6764.1, Definition 6764.2, Definition 6764.3, and Definition 6764.4 seed the finite-stage ODE computation packet over ODE, Picard–Lindelof, DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RungeKuttaFiniteStep}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host ODE existence, selected global solutions, quotient equality of names, ambient Real completeness, or a Lean theorem.

Upgrade path: A formal route can encode the twelve-row carrier and prove the NameCert-obligation, RegSeqRat-handoff, and Real-seal nonescape theorems.

Chapter 6765

A Concrete Naming Certificate for $\mathsf{ClassifierChannelShiftLedger}^{\uparrow}$

The $\mathsf{ClassifierChannelShiftLedger}^{\uparrow}$ packet realizes the classifier-level information distinction of chapter 9366 as finite before-and-after channel data. It separates record growth from classifier shift by displaying the before classifier row, after classifier row, record-growth row, shift witness, transport, replay, provenance, and local naming in one NameCert surface. The classifier-shift witness row is consumed after the gap-closure boundary of chapter 1006, because a classifier channel only counts as moved once the source gap has a displayed closure boundary rather than mere record growth. The packet is a BEDC-native ledger for classifier-channel movement, not a longer continuation trace and not a host semantic comparison.

The before and after classifier rows consume the classifier-membership surface of chapter 3911: a channel comparison reads B and A only after membership has been exposed as a finite type-checking classifier row. The refusal row for record-growth-as-discovery is routed through chapter 1006, so the ledger distinguishes classifier-row consumption from refusal-row consumption before W is allowed to witness movement.

Definition 6765.1 (Classifier-channel shift ledger carrier)**.** *A* $\mathsf{ClassifierChannelShiftLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, A, R, W, H, C, P, N)$$

with the following displayed rows:

1. *B is the before-classifier row, naming the certified classifier state before the audited transition.*
2. *A is the after-classifier row, naming the certified classifier state after the audited transition.*
3. *R is the record-growth row, naming the displayed histories added by mechanical unfolding or continuation replay.*
4. *W is the classifier-shift witness row, empty exactly when B and A have identical classifier readings on the displayed source scope and nonempty when the channel changes.*
5. *H records componentwise* hsame *transport for before classifier, after classifier, record growth, shift witness, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from the before classifier and record-growth rows to the after classifier and shift witness rows.*

7. *P is the* Pkg *provenance over classifier-level information, computation as continuation, type checking as classifier membership, gap-closure boundary,* BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierChannelShiftLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, A, R, W, H, C, P, N. It has no coordinate that identifies record growth with classifier shift, no hidden semantic oracle, and no permission to read a longer record as discovery without the displayed shift witness.

Theorem 6765.2 (Classifier-channel shift ledger NameCert obligations)**.** *For every accepted* ClassifierChannelShiftL *carrier*

$$L = (B, A, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClassifierChannelShiftLedger}^{\uparrow}}$ *surface consists exactly of before-classifier exposure, after-classifier exposure, record-growth exposure, shift-witness exactness, componentwise transport, continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6765.1. The analytic rows are B, A, R, W: the two classifier states, the record-growth row, and the witness that says whether the classifier channel moved.

The structural rows H, C, P, N transport, replay, source, and name the same finite tuple. They can carry the classifier-channel packet through a consumer, but they cannot replace the shift witness by the existence of a longer record.

Since the carrier has no further coordinate, the local NameCert surface has exactly the stated obligations and no hidden semantic oracle, host classifier comparison, or record-growth-as-discovery rule. □

Theorem 6765.3 (Classifier-channel shift ledger record-growth separation)**.** *An accepted* ClassifierChannelShiftLedg *packet with identical before and after classifier rows has an empty classifier-shift witness row even when the record-growth row is nonempty. Conversely, a nonempty shift witness must be read from W and cannot be inferred from R alone.*

Proof. Read the before and after classifier rows B and A. When they are identical on the displayed source scope, the carrier definition fixes W as the empty shift witness row.

The record-growth row R is separate. It may record added histories from continuation replay, but no clause of the carrier turns that growth into classifier movement.

For the converse reading, a consumer must project W directly. Transport, replay, provenance, and local naming through H, C, P, N preserve the displayed tuple, so a nonempty classifier shift is witnessed by W and not by R alone. □

Falsifiable prediction: A packet with identical before/after classifier rows has zero classifier-shift witness even when the record-growth row is nonempty.

Independence witness: Compare `classifier_level_information`, `computation_as_continuation`, `type_checking_as_classifier_membership`, and gapclosureboundary: the packet has before/after classifier rows and a shift witness that none of those siblings carry, so no bijection preserves all fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierChannelShiftLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierChannelShiftLedger}^{\uparrow}$ packet is generated from before-classifier, after-classifier, record-growth, shift-witness, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierChannelShiftLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6765.1, Definition 6765.2, and Definition 6765.3 seed the finite classifier-channel ledger separating record growth from classifier shift.

Formal verification: FormalStatus(NameCert$_{\text{ClassifierChannelShiftLedger}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close positive discovery certification, a global semantic classifier oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shift-witness exactness, record-growth separation, Nontrivial ClassifierChannelShiftLedgerUp, and FieldFaithful ClassifierChannelShiftLedgerUp in TasteGate.lean.

Chapter 6766

A Concrete Naming Certificate for $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$ packet records the finite modulus-bearing completion route by which regular rational observations become a Real-facing seal. It sits beside the stream window packet of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the explicit modulus surface of chapter 292, and the terminal real-name seal of chapter 38. The chapter isolates the finite rows a completion consumer may inspect before any completed object is claimed.

Definition 6766.1 (Modulated Cauchy completion carrier)**.** *A* $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *stream row whose finite requests are the only observations admitted by the packet.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row; for each displayed precision request it names the finite threshold after which the source stream must be read.*
3. *W is the finite window ledger pairing each request with the threshold supplied by M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same window ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S, W, D after the thresholded window has been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source stream, modulus, window, dyadic tolerance, and regular readback rows have been consumed.*
7. *H records componentwise* hsame *transport for S, M, W, D, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from source requests to modulus thresholds, from thresholds to windows, from windows to dyadic tolerances, from tolerances to regular readback, and from readback to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulatedCauchyCompletion}^{\uparrow}}$ *row.*

The carrier exports only this finite modulus-controlled completion surface. It contains no quotient-completion coordinate, selected limit, countable-choice scheduler, host real equality, or ambient completeness row for $\mathsf{Real}^{\uparrow}$.

Theorem 6766.2 (Modulated Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$ *carrier* $T = (S, M, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ModulatedCauchyCompletion}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission exposes precisely the rows $S, M, W, D, R, E, H, C, P, N$ before a completion-facing read is exported;*
2. *modulus coverage requires every displayed precision request in W to be served by a threshold row in M over the same source stream S;*
3. *regular-window readback factors each admitted read through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *before the Real seal is visible;*
4. *Real-seal nonescape makes E downstream of S, M, W, D, R and excludes quotient completions, selected limits, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
5. *provenance and transport require H, C, P, N to preserve the same displayed route under* hsame *transport,* Cont *replay, package sourcing, and local naming.*

Proof. Carrier admission is projection from Definition 6766.1. The packet lists exactly the source stream, modulus, window ledger, dyadic tolerance row, regular readback, Real seal, and structural rows as its public surface.

For modulus coverage, inspect M and W together. Each window row is admitted only with the threshold supplied by the displayed $\mathsf{CauchyModulus}^{\uparrow}$ row, so no precision request can be served by an external schedule or by countable choice.

The regular-window readback then reads S, W, D, R in order. The sibling chapters chapter 126, chapter 292, chapter 291, and chapter 301 provide the stream, modulus, dyadic, and regular rational surfaces named by the carrier.

Finally the seal row E is downstream of that finite route, and H, C, P, N only transport, replay, source, and name the same packet. Since the carrier has no quotient, selected-limit, choice, host-equality, or ambient-completeness coordinate, a Real-facing consumer receives only the displayed $\mathsf{Real}^{\uparrow}$ seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulatedCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulatedCauchyCompletion}^{\uparrow}$ packet is generated from a source StreamName row, an explicit CauchyModulus threshold row, finite window rows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulatedCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 6766.1 and Definition 6766.2 seed the finite modulus-controlled completion route for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulatedCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, full RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus coverage, regular-window readback, Real-seal nonescape, and transport-provenance preservation for BEDC.Derived.ModulatedCauchyCompletionUp.

Chapter 6767

A Concrete Naming Certificate for $\mathsf{LogicalRelation}^{\uparrow}$

Following Tait, Girard, and Reynolds, the $\mathsf{LogicalRelation}^{\uparrow}$ packet records the finite proof-theoretic surface used by reducibility candidates, Kripke logical relations, and parametricity-style typed-term arguments. It sits beside the MetaCIC critical-path ledger of chapter 765, the readback discipline of chapter 6305, and the lambda-calculus syntax carrier of chapter 200. The packet names typed-term rows, world-extension rows, reducibility evidence, substitution-stability rows, componentwise transport, replay, provenance, and local naming; it does not assert global normalization, subject reduction, full confluence, quotient syntax equality, or a completed semantic model.

Definition 6767.1 (LogicalRelation carrier)**.** *A* $\mathsf{LogicalRelation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, W, E, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-term row, carrying the term, type, binder context, and source syntax route read from* $\mathsf{LambdaCalc}^{\uparrow}$ *or the represented MetaCIC term surface.*
2. *W is the world-extension row, recording context growth, environment extension, step-index windows, or Kripke monotonicity data as finite packet entries.*
3. *E is the reducibility-evidence row, naming the candidate, logical predicate membership, closed-candidate witness, or parametric relation evidence consumed by the proof route.*
4. *S is the substitution-stability row, recording how typed substitution, environment extension, and binder transport preserve the same relation under the world extension.*
5. *H records componentwise* hsame *transport for T, W, E, S, C, P, N.*
6. *C records displayed* Cont *replay from typed-term inspection through world extension, reducibility evidence, substitution stability, transport, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{LogicalRelation}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{NormalizationByEvaluation}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LogicalRelation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an unbounded semantic universe, proof-irrelevant candidate quotient, host evaluator, global normalization theorem, or external syntax equality.

Theorem 6767.2 (LogicalRelation NameCert obligations)**.** *For an accepted* $\mathsf{LogicalRelation}^{\uparrow}$ *carrier* $R = (T, W, E, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LogicalRelation}^{\uparrow}}$ *surface consists exactly of carrier admission for typed-term rows, world-extension rows, reducibility evidence, substitution-stability rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any candidate-mediated strong-normalization, parametricity, or MetaCIC normalization consumer must read the typed term through* T*, extend only through* W*, consume reducibility evidence through* E*, preserve substitution through* S*, and then pass through* H, C, P, N*. The packet exports no global strong normalization theorem, no full subject reduction theorem, no global confluence theorem, no beta-conversion decidability result, and no quotient equality of syntax.*

Proof. Carrier admission is projection from Definition 6767.1. The tuple lists exactly the typed-term, world-extension, reducibility-evidence, substitution-stability, transport, replay, provenance, and local naming rows.

The proof-theoretic route is finite. A consumer may inspect typed terms through T, extend contexts through W, and use candidate or logical-predicate evidence through E only while the substitution row S remains attached to the same displayed packet.

The structural rows then preserve the route. Componentwise transport H and replay C move the typed-term, world, evidence, and substitution rows together, while P, N record the source chapters and local naming. Since the carrier lacks rows for global normalization, full subject reduction, global confluence, beta-conversion decidability, quotient syntax equality, or completed semantic universes, those conclusions cannot be exported by the local certificate. □

Falsifiable prediction: Every accepted $\mathsf{LogicalRelation}^{\uparrow}$ consumer that claims candidate-mediated normalisation, parametricity, or MetaCIC normalization must expose the typed-term row, world-extension row, reducibility-evidence row, substitution-stability row, componentwise transport, continuation replay, provenance, and local NameCert before citing the result.

Independence witness: metaciccriticalpath, normalization-by-evaluation, lambdacalc: $\mathsf{LogicalRelation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries typed-term rows, world-extension rows, reducibility evidence, and substitution stability as one proof-theoretic packet. $\mathsf{MetaCICCriticalPath}^{\uparrow}$ carries residual and candidate budgets without the relation carrier, $\mathsf{NormalizationByEvaluation}^{\uparrow}$ carries evaluation and readback without the candidate relation, and $\mathsf{LambdaCalc}^{\uparrow}$ carries syntax and beta ledgers without world-indexed reducibility evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{LogicalRelation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LogicalRelation}^{\uparrow}$ packet is generated from typed-term rows, world-extension rows, reducibility evidence, substitution-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LogicalRelation}^{\uparrow}}$, seed)

Scope closed: Definition 6767.1 and Definition 6767.2 seed the finite proof-theoretic packet for candidate-mediated MetaCIC and typed-term arguments.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LogicalRelation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global normalization, full subject reduction, global confluence, beta-conversion decidability, quotient syntax equality, completed semantic models, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, world extension, reducibility evidence, substitution stability, transport replay, provenance, and local naming for BEDC.Derived.LogicalRelationUp.

Chapter 6768

A Concrete Naming Certificate for $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$

Following Bishop and Bridges and Bourbaki/Kelley completion terminology, the $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$ packet records the finite boundary where two displayed reads of a Cauchy completion are compared before any quotient completion equality is invoked. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the public completion reflector route of chapter 1585. The packet names an equalizer surface for two completion reads; it does not identify quotient classes, select representatives, import host equality, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6768.1 (Cauchy completion equalizer carrier)**.** *A* $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, Q, D, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row listing the finite observation windows used by both completion reads.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row through which the two reads are compared at each requested window.*
3. *D is the displayed source regularity row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and carrying the rational Cauchy data common to both sides.*
4. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row exposing the completion-facing handoff without presenting a quotient carrier or selected host limit.*
5. *A is the first displayed completion read, obtained only after S, Q, D, R have been exposed.*
6. *B is the second displayed completion read over the same S, Q, D, R boundary.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-rational, reflector, two read, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $S, Q, D, R, A, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionEqualizer}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $S, Q, D, R, A, B, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 6768.2 (Cauchy completion equalizer NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionEqua}$ *carrier $E = (S, Q, D, R, A, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionEqualizer}^{\uparrow}}$ *surface consists exactly of carrier admission, shared stream-window exposure through S, shared dyadic tolerance exposure through Q, regular-rational source exposure through D, reflector handoff through R, the two displayed completion reads A, B, componentwise transport through H, displayed replay through C, provenance through P, local naming through N, and non-escape from quotient completion equality, selected representatives, countable choice, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Carrier admission is projection from Definition 6768.1. The accepted packet exposes exactly the stream, dyadic, regular-rational, reflector, two read, transport, replay, provenance, and naming rows.

The equalizer read is shared before it is compared. Both A and B are downstream of the same finite windows S, tolerance row Q, regular-rational source row D, and reflector row R.

The comparison is therefore a displayed BHist and $\mathsf{NameCert}$ route, not a quotient-completion equation. Since the carrier contains no row for selected representatives, host equality, countable choice, ambient completeness, or bridge checking, the listed rows exhaust the local obligation surface. □

Theorem 6768.3 (Cauchy completion equalizer reflector kernel)**.** *Every* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$ *carrier receives the two completion reads only through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow} \longmapsto A, B.$$

Consequently the equalizer kernel available to the consumer is the displayed agreement boundary over $S, Q, D, R, A, B, H, C, P, N$, not quotient completion equality, selected representatives, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Open an accepted carrier $E = (S, Q, D, R, A, B, H, C, P, N)$. The only completion-facing handoff row is R, and both displayed reads A and B are stored after the shared stream, dyadic, and regular-rational rows.

The reflector-facing consumer must therefore inspect S, Q, D, R before either read is available. Transport through H and replay through C preserve the same boundary, while P and N source and name only that boundary.

No further kernel data can be read from the packet. The missing quotient, representative-selection, host-equality, choice, completeness, and bridge coordinates cannot be reconstructed from the displayed rows, so the consumer receives exactly the finite equalizer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEqualizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$ packet is generated from shared StreamName windows, a DyadicRatCore tolerance row, a RegSeqRat source row, a CauchyCompletionReflector handoff, two displayed completion reads, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEqualizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6768.1, Definition 6768.2, and Definition 6768.3 seed the finite equalizer boundary for two displayed completion reads over StreamName, DyadicRatCore, RegSeqRat, CauchyCompletionReflector, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyCompletionEqualizer}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exposure, dyadic tolerance exposure, regular-rational source exposure, reflector handoff, displayed-read comparison, transport and replay stability, provenance non-escape, and local NameCert faithfulness for BEDC.Derived.CauchyCompletionEqualizerUp.

Chapter 6769

A Concrete Naming Certificate for $\mathsf{HeineCantorFiniteNet}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor theorem, the $\mathsf{HeineCantorFiniteNet}^{\uparrow}$ packet isolates the finite-net route from compact metric continuity to a uniform modulus. It reads the metric source of chapter 57, the compact finite-net surface of chapter 129, the pointwise continuity rows of chapter 130, and the uniform-modulus consumer interface of chapter 369. The packet names the finite lower-bound fold and coverage ledger used by the compact continuity argument; it does not import open-cover compactness, selected Lebesgue numbers, host modulus functions, sequential compactness, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6769.1 (Heine-Cantor finite-net carrier)**.** *A* $\mathsf{HeineCantorFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, K, F, L, Q, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the distance and comparison reads used by the compact finite-net and continuity rows.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, displaying a finite center bundle and the coverage ledger over X.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, including the pointwise modulus rows attached to the centers listed by K.*
4. *L is the local finite-net ledger pairing each center of K with its pointwise modulus row from F and its source-radius comparison.*
5. *Q is the rational lower-bound fold over the finitely many source-radius comparisons in L.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row consumed only after K, F, L, Q have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, compact finite-net row, continuous-map row, local ledger, rational fold, uniform-modulus row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a target-tolerance request through $X, K, F, L, Q, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{HeineCantorFiniteNet}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteNet}^{\uparrow}}$ *row.*
The carrier exposes only the compact finite-net coverage route, pointwise modulus inventory, finite lower-bound fold, uniform-modulus output, componentwise transport, replay, provenance, and local naming.

Theorem 6769.2 (Heine-Cantor finite-net modulus extraction)**.** *Let $G = (X, K, F, L, Q, U, H, C, P, N)$ be an accepted* $\mathsf{HeineCantorFiniteNet}^{\uparrow}$ *carrier. Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by G factors through compact finite-net coverage and pointwise modulus rows before the row U is visible:*

$$X, K, F, L, Q \longmapsto U \longmapsto H, C, P, N.$$

Thus the modulus extraction is a finite NameCert route through the metric source, compact finite-net row, continuous-map pointwise rows, local ledger, and rational lower-bound fold, not a selected host modulus function or an open-cover compactness principle.

Proof. Project the carrier of Definition 6769.1. The output row U is downstream of the source metric row X, the compact finite-net row K, and the continuous-map row F.

Read the finite ledger L. It pairs only the centers already displayed by K with pointwise modulus rows already displayed by F, so the extraction has no independent source of compactness data.

Fold the displayed source-radius comparisons through Q. Because the fold ranges over the finite center bundle in K, the resulting lower bound is a finite ledger row rather than a selected Lebesgue number or host minimum.

Finally replay the route through H, C, P, N. These structural rows transport, continue, package, and name the same displayed packet, so every uniform-modulus consumer reaches U only through X, K, F, L, Q. □

Theorem 6769.3 (Heine-Cantor finite-net ledger exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteNet}^{\uparrow}}$ *ledger for Definition 6769.1 is exhausted by carrier admission for $X, K, F, L, Q, U, H, C, P, N$, compact finite-net coverage, pointwise modulus pairing, finite rational lower-bound folding, uniform-modulus output admission, componentwise transport, continuation replay, package provenance, and local naming. The packet exports no selected Lebesgue number, host modulus function, sequential compactness principle, countable choice, quotient equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission supplies exactly the displayed tuple. The analytic rows are X, K, F, L, Q, U, and the structural rows are H, C, P, N.

The extraction theorem, Definition 6769.2, accounts for every modulus-facing read. It forces the consumer through compact finite-net coverage, pointwise modulus rows, and the finite rational fold before the uniform-modulus output is consumed.

The absent principles are not coordinates of the carrier and are not generated by the structural rows. Therefore the local NameCert ledger is exhausted by the finite route named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteNet}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HeineCantorFiniteNet}^{\uparrow}}$, seed)

Scope closed: Definition 6769.1, Definition 6769.2, and Definition 6769.3 seed the finite-net route from compact metric continuity to a uniform modulus.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HeineCantorFiniteNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general Heine-Borel compactness, open-cover compactness, selected Lebesgue numbers, host modulus functions, sequential compactness, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, pointwise modulus pairing, rational lower-bound folding, uniform-modulus output admission, transport, replay, provenance, and local naming for BEDC.Derived.HeineCantorFiniteNetUp.

Constructive story: A HeineCantorFiniteNet$^{\uparrow}$ packet is generated from a Metric source row, CompactMetric finite-net coverage, ContinuousMap pointwise modulus rows, a local finite-net ledger, a rational lower-bound fold, a UniformModulus output row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6770

A Concrete Naming Certificate for $\mathsf{BishopLocatedMaximum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedMaximum}^{\uparrow}$ packet records maximum-location data as a finite BEDC NameCert route through compact finite nets, displayed value windows, and located comparisons. It sits over the compact-metric source of chapter 129, the uniformly continuous map row of chapter 130, the finite-net handoff of chapter 6769, the stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the stronger attained-extreme surface of chapter 3984. The packet exposes the finite comparison route needed by optimization consumers without selecting an argmax, importing open-cover compactness, identifying quotient real names, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6770.1 (Bishop located maximum carrier)**.** *A* $\mathsf{BishopLocatedMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, N, W, R, E, M, H, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the compact metric inspection surface and the finite center coverage consumed by maximum-location reads.*
2. *F is the uniformly continuous* $\mathsf{ContinuousMap}^{\uparrow}$ *row, exposing only displayed value requests over the compact source row K.*
3. *N is the finite-net cover row, recording the finite list of centers, radii, and coverage witnesses through which every maximum-facing request enters.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite value windows attached to the centers of N.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *value readback row for the rational windows exposed by W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, admitting the displayed value readbacks as Real-facing names before comparisons are consumed.*
7. *M is the located maximum comparison row, comparing each displayed candidate value with the finite net family through located upper and lower comparison windows.*
8. *H records componentwise* hsame *transport for the compact source, uniformly continuous map row, finite-net cover, stream schedule, regular rational readback, Real seal, located comparison row, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from a maximum-facing request through $K, F, N, W, R, E, M, H, P, L$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedMaximum}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, HeineCantorFiniteNet $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedMaximum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed compact source, uniformly continuous map row, finite-net cover, stream windows, regular rational readbacks, Real seals, located maximum comparisons, transport, replay, provenance, and local naming. It has no coordinate for a selected argmax, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedMaximumUp.BishopLocatedMaximumTasteGate single carrier alignment`

Theorem 6770.2 (Bishop located maximum NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedMaximum}^{\uparrow}$ *carrier $B = (K, F, N, W, R, E, M, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedMaximum}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed compact source, uniformly continuous map row, finite-net cover, value windows, regular rational readbacks, Real seal, located comparison row, transport, replay, provenance, and local naming;*

2. *finite-net exposure, requiring every maximum-facing source request to pass through the centers and coverage witnesses named by N;*

3. *value readback through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *requiring W, R, E to be visible before any comparison row is consumed;*

4. *located maximum comparison, requiring M to compare only displayed Real-sealed value windows over the finite net;*

5. *ledger refusal for selected argmax data, open-cover compactness, quotient real equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6770.1. The rows K, F, N expose the compact source, map row, and finite-net coverage surface that every later value read must use.

Read the value rows next. The stream schedule W supplies finite windows, R reads those windows as regular rational data, and E seals the displayed readbacks as $\mathsf{Real}^{\uparrow}$ names before a comparison can be used.

The comparison row M has access only to the Real-sealed finite-net values already displayed by N, W, R, E. The structural rows H, C, P, L transport, replay, source, and name the same finite route, so the five listed obligations exhaust the local NameCert surface. □

Theorem 6770.3 (Bishop located maximum finite-net handoff)**.** *Every maximum-facing read exported by an accepted* $\mathsf{BishopLocatedMaximum}^{\uparrow}$ *carrier factors through the displayed finite route*

$$K, F, N \longmapsto W, R, E \longmapsto M \longmapsto H, C, P, L.$$

Thus an optimization consumer can inspect only the compact source, uniformly continuous map row, finite net, value windows, located comparison, and Real seal before reaching the local naming row. It cannot bypass the finite-net handoff to obtain a selected argmax, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source and map rows. The compact metric source K and uniformly continuous map row F become visible to a maximum-facing read only through the finite-net cover N.

The value route is then fixed. The stream schedule W records the finite observation windows attached to the net, the regular rational row R reads those windows, and the terminal seal E is reached before any optimization-facing comparison is available.

Use the located comparison row M only after the sealed value windows are displayed. Since M compares the finite family exposed by N, W, R, E, it supplies a located maximum comparison surface rather than a selected maximizing point.

Finally replay the same route through H and C, while P and L provide provenance and local naming. The carrier has no coordinate for the excluded principles, so every consumer reaches maximum-location data only through the finite handoff shown above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedMaximum}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6770.1, Definition 6770.2, and Definition 6770.3 seed the finite-net maximum-location route over compact-source, uniformly continuous map, value-window, located-comparison, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedMaximum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected maximizers, full open-cover compactness, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BishopLocatedMaximum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, StreamName-RegSeqRat-Real value readback, located maximum comparison, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedMaximum

Constructive story: A $\mathsf{BishopLocatedMaximum}^{\uparrow}$ packet is generated from a CompactMetric source row, a uniformly continuous ContinuousMap row, a finite-net cover, StreamName value windows, RegSeqRat readbacks, a Real seal, a located maximum comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6771

A Concrete Naming Certificate for $\mathsf{FiniteMetricCompactness}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{FiniteMetricCompactness}^{\uparrow}$ packet records the elementary compactness certificate for a finite metric carrier. It sits between the distance rows of chapter 57, the finite-net compactness surface of chapter 129, the total-boundedness ledger of chapter 127, the dyadic tolerance reads of chapter 38, and the finite-net Heine–Cantor route of chapter 6769. The packet names only finite point rows, explicit distance comparisons, dyadic tolerance rows, finite-net coverage, transport, replay, provenance, and local naming; it does not import open-cover compactness, sequential compactness, choice-selected subsequences, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6771.1 (Finite metric compactness carrier)**.** *A* $\mathsf{FiniteMetricCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F_{\mathrm{mc}} = (X, D, E, R, T, K, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite point-row bundle, listing every admitted point of the carrier before any compactness read is exported.*
2. *D is the* $\mathsf{Met}^{\uparrow}$ *distance row over X, displaying symmetry, diagonal, and triangle-comparison reads only for the finite point inventory.*
3. *E is the finite equality and* hsame*-stability row for point names, distance cells, tolerance cells, coverage rows, and local naming rows.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *dyadic tolerance row, exposing positive dyadic requests and comparison readback for the metric inequalities used by the net ledger.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, assigning to each displayed tolerance in R a finite center subrow drawn from X.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *ledger row that records the same finite-net coverage as a compact-metric certificate and nothing stronger.*
7. *L is the local coverage ledger: for each point row of X and each tolerance row of R, it displays a center from the finite center subrow of T and the corresponding D-comparison.*

8. *H records componentwise* hsame *transport for point rows, metric rows, dyadic tolerance rows, finite-net coverage, compact-metric ledger rows, replay rows, provenance rows, and local naming rows.*
9. *C records* Cont *replay from a tolerance request through $X, D, E, R, T, K, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricCompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineCantorFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for open covers, infinite subsequences, countable choice, quotient metric equality, or a completeness argument over ambient real names.

Theorem 6771.2 (Finite metric compactness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricCompactness}^{\uparrow}$ *carrier $F_{\mathrm{mc}} = (X, D, E, R, T, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricCompactness}^{\uparrow}}$ *surface is exhausted by six finite obligation rows:*

1. *finite point-row admission for X;*
2. $\mathsf{Met}^{\uparrow}$ *distance-row exactness for D over the listed points;*
3. $\mathsf{Real}^{\uparrow}$ *dyadic tolerance admission and comparison readback for R;*
4. $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net coverage and* $\mathsf{CompactMetric}^{\uparrow}$ *ledger agreement through T, K, L;*
5. *componentwise* hsame *transport and* Cont *replay through E, H, C;*
6. *provenance and local naming through P, N.*

No obligation is discharged by open-cover compactness, selected subsequences, countable choice, quotient equality of metric presentations, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6771.1. The point inventory X is finite, and every distance comparison consumed by the certificate is displayed by D over that same inventory.

Read the tolerance and coverage rows. The row R supplies only dyadic tolerance requests and comparisons, while T and L answer each such request with finitely many centers drawn from X and displayed distance inequalities. The compact-metric ledger K records this finite-net evidence as its compactness surface.

Finally replay the structural rows. The rows E, H, C, P, N transport, continue, package, and name the displayed packet, so the excluded open-cover, subsequence, choice, quotient, and completeness principles have no carrier coordinate from which to enter the local NameCert surface. □

Theorem 6771.3 (Finite metric compactness finite-net sequential bridge)**.** *Every consumer that reads an accepted* $\mathsf{FiniteMetricCompactness}^{\uparrow}$ *packet as a sequential compactness witness must factor that read through the explicit finite-net coverage rows*

$$X, D, R, T, L \longmapsto K \longmapsto H, C, P, N.$$

The bridge therefore exports a finite-center repetition route: an infinite sequence over the finite point-row inventory yields a recurrent center only because the carrier exposes the finite inventory and the finite-net coverage ledger. It does not export a Bolzano–Weierstrass theorem, a choice-selected subsequence operator, an open-cover compactness principle, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. The carrier contains no sequential object as primitive data. A sequential consumer must first read the finite point-row inventory X and the metric comparisons D.

The finite-net rows then supply the only compactness route. For each dyadic tolerance request in R, the rows T and L display finitely many centers and the distance comparisons that cover all listed points. The ledger K repackages precisely that evidence as a $\mathsf{CompactMetric}^{\uparrow}$ row.

Any sequential reading is therefore downstream of a finite-center repetition argument over the displayed inventory. Since the structural rows H, C, P, N preserve only the same packet, the bridge cannot introduce open covers, choice-selected subsequences, Bolzano–Weierstrass data, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteMetricCompactness}^{\uparrow}$ consumer that reaches a $\mathsf{CompactMetric}^{\uparrow}$ row without displaying finite point rows, dyadic tolerance rows, and finite-net coverage refutes the carrier surface.

Independence witness: metric, compactmetric, totallybounded, real, heine-cantor-finite-net: no listed sibling supplies the whole packet $F_{\mathrm{mc}} = (X, D, E, R, T, K, L, H, C, P, N)$, because each sibling contributes only distance rows, compact-metric ledger form, finite-net coverage, dyadic tolerance reads, or a downstream Heine–Cantor route while this chapter binds them into one finite metric compactness certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetricCompactness}^{\uparrow}$ packet is generated from finite point rows, Metric distance rows, Real dyadic tolerance rows, TotallyBounded finite-net coverage, CompactMetric ledger rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6771.1, Definition 6771.2, and Definition 6771.3 seed the finite metric compactness route through explicit finite-net coverage.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetricCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, Bolzano–Weierstrass compactness, choice-selected subsequences, quotient equality, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, metric distance exactness, dyadic tolerance readback, finite-net coverage, compact-metric ledger agreement, sequential bridge nonescape, transport, replay, provenance, refusal, and local naming for BEDC.Derived.FiniteMetricCompactnessUp.

Chapter 6772

A Concrete Naming Certificate for $\mathsf{FortissimoSpace}^{\uparrow}$

Following Steen–Seebach and Kelley on classical point-set examples, the $\mathsf{FortissimoSpace}^{\uparrow}$ packet records the fortissimo topology by finite observable-open rows rather than by an unrestricted host topology. It sits after the topology carrier of chapter 88, the Sierpinski observable truth surface of chapter 1860, the finite schedule discipline of chapter 126, and the set-like membership boundary of chapter 3668. The certificate names a distinguished point, isolated-point rows, and countable-exception neighbourhood schedules; it does not assert a host separation theorem, a completed power set of opens, a selected enumeration of every exception set, or an ambient real line.

Definition 6772.1 (FortissimoSpace carrier)**.** *A* $\mathsf{FortissimoSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, \omega, I, Q, O, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted point carrier row, read as a* $\mathsf{SetLike}^{\uparrow}$ *membership surface for the displayed finite public points.*
2. *ω is the distinguished-point row. It names the unique non-isolated point whose neighbourhoods are inspected by countable-exception schedules.*
3. *I is the isolated-point ledger. For every displayed point row apart from ω, it stores the singleton-open read used by topology and Sierpinski consumers.*
4. *Q is the countable-exception* $\mathsf{StreamName}^{\uparrow}$ *schedule. Each finite prefix lists the points excluded from the current neighbourhood of ω and records that later prefixes may only refine the visible exception ledger.*
5. *O is the observable-open row. It exposes the truth of membership in an inspected open only through the isolated singleton rows in I or the distinguished-point neighbourhood schedule in Q.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *provenance row, recording finite union and intersection reads only for the opens generated by I, Q, O.*
7. *H records componentwise* hsame *transport for the point carrier, distinguished point, isolated rows, exception schedule, observable-open row, topology provenance, replay, package, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$X, \omega, I, Q \longmapsto O \longmapsto T.$$

9. *P is the* Pkg *provenance over* $\mathsf{FortissimoSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FortissimoSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \omega, I, Q, O, T, H, C, P, N$. It has no coordinate for a host set of all open subsets, a global choice of countable exception enumerations, a separation theorem, compactness, metrizability, or ambient real-line structure.

Theorem 6772.2 (FortissimoSpace NameCert obligations)**.** *For every accepted* $\mathsf{FortissimoSpace}^{\uparrow}$ *carrier*

$$F = (X, \omega, I, Q, O, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FortissimoSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, distinguished-point admission, isolated singleton-open admission, countable-exception schedule exactness, observable-open truth through* $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$ *open-row provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 6772.1. No obligation row asserts a host topology, a completed lattice of opens, a selected global enumeration of all countable exception sets, separation axioms, compactness, metrizability, or real-line structure.*

Proof. Carrier admission is projection from Definition 6772.1. The accepted packet first displays the point carrier X, the distinguished point ω, the isolated-point ledger I, and the countable-exception schedule Q.

The observable-open row O is then forced to read membership in only two ways. Away from ω it uses the isolated singleton rows in I; at ω it uses the finite prefix of the exception schedule in Q.

The topology row T records only the finite open-row provenance generated by those reads. The structural rows H, C, P, N transport, replay, source, and name the same finite route without adding any host open-set coordinate.

Because the carrier inventory contains no completed open lattice, selected global exception enumeration, separation theorem, compactness row, metrizability row, or ambient real-line row, none of those objects can be exported as a local NameCert obligation. □

Theorem 6772.3 (FortissimoSpace Sierpinski topology handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$*-facing,* $\mathsf{SierpinskiSpace}^{\uparrow}$*-facing, or observable-open consumer of an accepted* $\mathsf{FortissimoSpace}^{\uparrow}$ *packet must read the route*

$$X, \omega, I, Q, O, T, H, C, P, N.$$

The row O exposes only observable-open truth from isolated singleton rows and countable-exception neighbourhood prefixes, while T exposes only the corresponding $\mathsf{Topology}^{\uparrow}$ *open-row provenance. The handoff does not import host open sets, selected exception enumerations, separation axioms, compactness, metrizability, or ambient real-line data.*

Proof. Project the accepted carrier. The point source is X, the distinguished point is ω, the isolated singleton evidence is I, and the only neighbourhood schedule at ω is the finite $\mathsf{StreamName}^{\uparrow}$ row Q.

The observable-open consumer then reads O. This row converts the isolated and distinguished-point routes into $\mathsf{SierpinskiSpace}^{\uparrow}$ truth values without exposing an ambient predicate on all subsets.

The topology consumer reads T only after that observable route has been replayed. The rows H, C, P, N preserve and name the same carrier coordinates, so no later consumer can replace the certificate by a host topology or selected family of exception sets. □

Closure status for $\mathsf{NameCert}_{\mathsf{FortissimoSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FortissimoSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6772.1, Definition 6772.2, and Definition 6772.3 seed the distinguished-point, isolated-row, countable-exception, observable-open, and topology-provenance surface for $\mathsf{FortissimoSpace}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FortissimoSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topology, completed open lattices, selected global exception enumerations, separation axioms, compactness, metrizability, real-line structure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distinguished-point admission, isolated singleton-open admission, countable-exception schedule exactness, observable-open truth, topology provenance, transport stability, replay stability, provenance exhaustion, and local NameCert rows for BEDC.Derived.FortissimoSpaceUp.

Constructive story: A $\mathsf{FortissimoSpace}^{\uparrow}$ packet is generated from a SetLikeUp point row, a distinguished point row, isolated singleton-open rows, a countable-exception StreamName schedule, observable-open Sierpinski truth, TopologyUp open-row provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6773

A Concrete Naming Certificate for $\mathsf{FiniteSignedMeasure}^{\uparrow}$

Following Hahn and standard signed-measure theory, the $\mathsf{FiniteSignedMeasure}^{\uparrow}$ packet names the finite signed event-value surface that Hahn decomposition and variation-measure consumers need before reading positive and negative mass rows. It sits after the base measure carrier of chapter 92, the finite nonnegative event-value packet of chapter 6127, the finite-partition variation ledger of chapter 3593, the decomposition route of chapter 5952, the regular-measure handoff of chapter 2263, and the terminal scalar seal of chapter 38. The packet records finite signed values, variation bounds, polarity ledgers, and their displayed transport rows.

Definition 6773.1 (Finite signed measure carrier)**.** *A* $\mathsf{FiniteSignedMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, E, V, Pm, Nm, T, U, R, H, C, L, Q)$$

with the following displayed rows:

1. *S is the* $\mathsf{Measure}^{\uparrow}$ *event-family row, exposing the admitted measurable events and their classifier.*
2. *E is the finite event ledger whose signed values are visible to the packet.*
3. *V is the signed event-value ledger, assigning each event in E a displayed* $\mathsf{Real}^{\uparrow}$ *scalar value.*
4. *Pm is the positive-variation finite measure row, read through* $\mathsf{FiniteMeasure}^{\uparrow}$ *on the same event ledger.*
5. *Nm is the negative-variation finite measure row, also read through* $\mathsf{FiniteMeasure}^{\uparrow}$ *on the same event ledger.*
6. *T is the total-variation bound row, naming a displayed* $\mathsf{Real}^{\uparrow}$ *bound for $Pm + Nm$ over the admitted finite event family.*
7. *U is the polarity and cancellation ledger, recording how the signed value row V is read as the difference of the displayed positive and negative finite-measure rows.*
8. *R is the regularity and scalar readback row, routing optional* $\mathsf{RadonMeasure}^{\uparrow}$ *requests and every terminal* $\mathsf{Real}^{\uparrow}$ *scalar through the same finite signed surface.*
9. *H records componentwise* hsame *transport for $S, E, V, Pm, Nm, T, U, R, C, L, Q$.*
10. *C records displayed* Cont *replay from an event request through signed value, positive variation, negative variation, total variation, polarity, and scalar readback rows.*

11. *L is the* Pkg *provenance over* FiniteSignedMeasure$^{\uparrow}$, Measure$^{\uparrow}$, FiniteMeasure$^{\uparrow}$, VariationMeasure$^{\uparrow}$, HahnDecomposition$^{\uparrow}$, RadonMeasure$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *Q is the local* $\mathsf{NameCert}_{\mathsf{FiniteSignedMeasure}^{\uparrow}}$ *row.*

The carrier has no coordinate for host signed-measure equality, a completed measure algebra, quotient event equality, selected Jordan decomposition, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6773.2 (FiniteSignedMeasure NameCert obligations)**.** *For every accepted* FiniteSignedMeasure$^{\uparrow}$ *carrier*

$$G = (S, E, V, Pm, Nm, T, U, R, H, C, L, Q),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSignedMeasure}^{\uparrow}}$ *surface consists exactly of event-family admission through S, finite event visibility through E, signed scalar values through V, positive and negative variation reads through Pm and Nm, total-variation boundedness through T, polarity and cancellation through U, Radon and real readback through R, componentwise transport through H, displayed replay through C, provenance through L, and local naming through Q.*

Proof. Project the carrier of Definition 6773.1. The rows S, E, V are the only event and signed-value data exposed before the variation rows appear.

Read Pm and Nm as finite nonnegative measure rows over the same finite event ledger. The total-variation row T bounds their displayed finite sum, while U records the signed-value read as the difference of those two finite nonnegative rows.

The readback row R is the only route to optional RadonMeasure or terminal Real consumers. Transport and replay are componentwise through H and C, so a consumer cannot replace the finite ledger by a host signed measure or by quotient event equality.

Finally L and Q source and name the packet. Since the carrier contains no completed measure algebra, selected Jordan decomposition, countable-choice selector, host signed-measure equality, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6773.3 (FiniteSignedMeasure Hahn consumer route)**.** *Every* HahnDecomposition$^{\uparrow}$ *consumer of a finite signed measure must read an accepted* FiniteSignedMeasure$^{\uparrow}$ *carrier through the displayed rows $S, E, V, Pm, Nm, T, U, R, H, C, L, Q$ before exposing positive-region, negative-region, or null-boundary data. The same route supplies* VariationMeasure$^{\uparrow}$, RadonMeasure$^{\uparrow}$, *and* Real$^{\uparrow}$ *consumers with finite signed values, finite variation bounds, and scalar readback without exporting host signed-measure equality, quotient event equality, selected partitions, or ambient measure completion.*

Proof. Begin with the Hahn decomposition carrier. Its signed-measure input row can be supplied by a finite signed surface only after the event ledger, signed values, variation rows, total-variation bound, and polarity ledger have been displayed.

Use the obligation theorem above. The rows Pm, Nm, T, U expose exactly the positive-variation, negative-variation, total-variation, and cancellation data that a Hahn-facing consumer needs before it reads positive and negative regions.

VariationMeasure and RadonMeasure consumers use the same packet boundary. The finite-partition variation read sees the signed value and variation rows, while the regular-measure read can pass through R only after the finite signed ledger has been displayed.

The remaining rows H, C, L, Q transport, replay, source, and name that route. No coordinate in the carrier can yield a host signed measure, quotient equality of events, a selected measurable partition, a completed measure algebra, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSignedMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSignedMeasure}^{\uparrow}$ packet is generated from a MeasureUp event-family row, a finite event ledger, signed RealUp event values, positive and negative FiniteMeasureUp variation rows, total-variation bound, polarity and cancellation ledger, optional RadonMeasureUp and RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSignedMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6773.1, Definition 6773.2, and Definition 6773.3 seed the finite signed event-value carrier consumed by HahnDecompositionUp, VariationMeasureUp, RadonMeasureUp, MeasureUp, FiniteMeasureUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSignedMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host signed-measure equality, completed measure algebras, quotient event equality, selected Jordan or Hahn decompositions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed event-value transport, positive and negative finite-measure variation reads, total-variation boundedness, polarity cancellation, Hahn consumer routing, Radon and Real readback, replay, provenance, and local naming for BEDC.Derived.FiniteSignedMeasureUp.

Chapter 6774

A Concrete Naming Certificate for $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$

Following Goebel and Kirk, the $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$ packet records the finite naming surface for an asymptotic-center read along a displayed regular Cauchy completion path. It sits over the regular rational rows of chapter 301, the common stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the real-name seal of chapter 38, the generic metric surface of chapter 57, and the complete-metric witness row of chapter 3687. The packet names the center candidate by finite readback and metric-radius comparison, not by quotienting Cauchy names, selecting a limit, or importing compactness.

Definition 6774.1 (Regular Cauchy asymptotic-center carrier)**.** *A* $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, K, A, W, D, R, E, T, H, C, P, S, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance classifier is used for every radius comparison.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *witness row supplying the completion-facing limit discipline for the ambient metric carrier.*
3. *A is the displayed regular Cauchy source family whose tail distances are being inspected.*
4. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *window through which all tail and center-candidate reads are exposed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the metric-radius budget read through W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational approximants of the tail-radius data.*
7. *E is the candidate-center row. It is presented only through the same finite window and the same metric source as the tail reads.*
8. *T is the finite asymptotic-radius comparison row, recording that every inspected tail request factors through M, K, A, W, D, R, E before a center claim is visible.*
9. *H records componentwise* hsame *transport for the metric source, complete-metric witness, Cauchy source family, common window, dyadic tolerance ledger, regular rational readback, candidate row, comparison row, package provenance, real seal, and local naming row.*

10. *C records displayed* Cont *replay from an asymptotic-center request through the same rows.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite radius ledger and regular rational readback have been displayed.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, a selected ambient limit, compactness, uniform convexity, excluded middle, countable choice, or a global minimizer principle outside the displayed packet.

Theorem 6774.2 (Regular Cauchy asymptotic-center NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$ *carrier* $Q = (M, K, A, W, D, R, E, T, H, C, P, S, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}}$ *surface consists exactly of carrier admission, common-window exposure, dyadic tolerance budgeting, regular rational readback, metric-radius comparison, complete-metric limit-discipline handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, and local naming over the rows displayed in Definition 6774.1. No obligation row exports quotient equality of Cauchy names, selected limits, compactness, uniform convexity, countable choice, trichotomy, or an external minimization oracle.*

Proof. Project the accepted carrier. The analytic data visible to a center consumer are precisely the metric source M, the complete-metric witness K, the regular Cauchy source family A, the common finite window W, the dyadic tolerance ledger D, the regular rational readback row R, the candidate row E, and the comparison row T.

The NameCert rows H, C, P, S, N add transport, replay, provenance, terminal sealing, and local naming for that same finite packet. They do not add an independent equality coordinate, because every transported read is componentwise and every replay follows the displayed route through W, D, R, T.

It remains to exclude hidden completion principles. The carrier lists no quotient-completion row, no selected limit row, no compactness or convexity row, and no external minimizer oracle. Thus the local certificate exhausts exactly the finite asymptotic-radius route displayed in the carrier. □

Theorem 6774.3 (Regular Cauchy asymptotic-center Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

along the rows $M, K, W, D, R, S, H, C, P, N$ before the candidate-center row E or comparison row T may be consumed. Consequently the terminal seal reports only the displayed regular Cauchy asymptotic-radius packet and cannot be used to derive quotient equality of names, selected completion limits, compactness, uniform convexity, countable choice, trichotomy, or a global minimizer principle.

Proof. Read the carrier order in Definition 6774.1. The terminal seal row S is downstream of the common stream window, dyadic tolerance ledger, regular rational readback, metric source, and complete-metric witness.

The comparison row T is also downstream of those finite rows. A consumer therefore cannot ask the $\mathsf{Real}^{\uparrow}$ seal for an asymptotic-radius statement until the shared window W, tolerance ledger D, readback row R, and metric/complete-metric handoff M, K have already been displayed.

The omitted principles are absent from the packet. Since H and C only transport and replay the listed rows, the seal cannot escape the finite route by manufacturing quotient equality, a selected limit, compactness, convexity, choice, trichotomy, or a minimizer outside the displayed data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6774.1, Definition 6774.2, and Definition 6774.3 seed the finite asymptotic-center route over displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, compactness, uniform convexity, countable choice, trichotomy, or global minimizer existence.

Upgrade path: Obligation closure requires checked carrier admission, common-window exposure, dyadic tolerance budgeting, regular rational readback, metric-radius comparison, complete-metric handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyAsymptoticCenterUp.

Constructive story: A $\mathsf{RegularCauchyAsymptoticCenter}^{\uparrow}$ packet is generated from a Metric source row, a CompleteMetric witness row, a regular Cauchy source family, one common StreamName window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a candidate-center row, a finite asymptotic-radius comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, a RealUp seal, and a local NameCert row.

Chapter 6775

A Concrete Naming Certificate for $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of compact intervals, the $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$ packet records the finite interval-cover data visible before any Heine–Borel, compactness, or completion-facing consumer may use it. It sits after the dyadic arithmetic of chapter 291, the stream windows of chapter 126, the regular rational rows of chapter 301, the real-name seal of chapter 38, and the interval-domain source of chapter 1543. The packet names only a displayed finite cover route; it does not assert open-cover compactness, select a subcover, identify quotient real names, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6775.1 (Finite open interval cover carrier)**.** *A* $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, O, L, R, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{IntervalDomain}^{\uparrow}$ *source row, naming the closed or bounded interval window whose cover is being inspected.*
2. *O is the finite open-interval row, listing the displayed interval members by rational endpoint names and local openness witnesses.*
3. *L is the located-overlap row, recording which displayed interval-domain cells meet which cover members.*
4. *R is the finite refinement row, comparing two displayed cover windows by finite parent maps and endpoint-containment witnesses.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for endpoint names, overlap probes, and retained finite members.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational observation row used to compare endpoint windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row for dyadic endpoint, radius, and containment checks.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after I, O, L, R, S, Q, D have been displayed.*
9. *H records componentwise* hsame *transport for source, cover, overlap, refinement, stream, regular-rational, dyadic, and seal rows.*

10. *C records displayed* Cont *replay for finite cover admission, endpoint containment, overlap inspection, and Real-facing seal reads.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenIntervalCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, a hidden finite-subcover selector, quotient equality of real names, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6775.2 (Finite open interval cover NameCert obligations)**.** *For the carrier of Definition 6775.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenIntervalCover}^{\uparrow}}$ *surface consists exactly of interval-domain source admission through I, finite open-interval admission through O, located-overlap exactness through L, finite refinement exactness through R, stream and regular-rational readback through S, Q, dyadic containment arithmetic through D, Real seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier row by row. The source row I fixes the interval-domain window, while O and L expose the only cover members and overlap witnesses visible to a downstream consumer.

The refinement row R compares cover windows by finite parent maps and displayed containment checks. The readback rows S, Q, D then inspect endpoint names, regular-rational windows, and dyadic inequalities without leaving the finite packet surface.

Finally E, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since no carrier coordinate stores an arbitrary open cover, a selected subcover, quotient real equality, countable choice, selected limits, or ambient completeness, those objects cannot become NameCert obligations for $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$. □

Theorem 6775.3 (Finite open interval cover constructive boundary)**.** *Every completion-facing or compactness-facing read of an accepted* $\mathsf{FiniteOpenIntervalCover}^{\uparrow}$ *packet must pass through the finite route*

$$I, O, L, R, S, Q, D, E, H, C, P, N.$$

The route may expose interval-domain cells, finite open interval members, stream-name windows, regular-rational observations, dyadic endpoint comparisons, and the terminal $\mathsf{Real}^{\uparrow}$ *seal, but it exports no open-cover compactness theorem, choice-selected subcover, quotient real equality, countable-choice schedule, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The consumer first reads I, O, L, R. These rows determine the source interval window, the finite cover members, the located overlap evidence, and the finite refinement comparison.

It then reads S, Q, D to inspect endpoint names and dyadic containment data. These observations are finite rows over the already displayed cover window, so they cannot quantify over a host family of all open intervals.

The remaining rows E, H, C, P, N preserve the same route under Real-seal inspection, transport, replay, provenance, and local naming. Therefore every downstream consumer receives exactly the finite packet named in the statement and none of the excluded compactness, choice, quotient, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenIntervalCover}^{\uparrow}}$.

Constructive story: A FiniteOpenIntervalCover$^{\uparrow}$ packet is generated from an IntervalDomainUp source row, finite open-interval rows, located-overlap witnesses, finite refinement rows, StreamName windows, RegSeqRat observations, DyadicRatCore comparisons, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteOpenIntervalCover}^{\uparrow}}$, seed)

Scope closed: Definition 6775.1, Definition 6775.2, and Definition 6775.3 seed the finite interval-cover interface over displayed interval-domain, stream, regular-rational, dyadic, and Real-seal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteOpenIntervalCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, finite-subcover selection, quotient real equality, countable choice, selected limits, ambient Real$^{\uparrow}$ completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite open-interval admission, located-overlap exactness, finite refinement stability, stream and regular-rational readback, dyadic containment arithmetic, Real seal non-escape, transport, replay, provenance, and local naming.

Chapter 6776

A Concrete Naming Certificate for $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$

Following Ahlfors and Beardon on disk automorphisms, the $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ packet records the finite disk-map and inverse rows by which a Poincare disk self-map is read as an automorphism before hyperbolic metric, boundary, Schwarz–Pick, or composition consumers inspect it. It sits after the disk metric packet of chapter 6209, the fractional linear carrier of chapter 5110, the hyperbolic metric surface of chapter 3474, and the boundary transport packet of chapter 6595. The certificate names only displayed disk-map, boundary-transport, hyperbolic-metric-preservation, inverse, composition, provenance, and NameCert rows; it does not import host complex analysis, a completed automorphism group, quotient boundary equality, or an unrecorded analytic classification theorem.

Definition 6776.1 (Poincare disk automorphism carrier)**.** *A* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, M, U, V, B, K, I, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric source row for the finite domain and codomain observations.*
2. *M is the* $\mathsf{MobiusTransform}^{\uparrow}$ *coefficient and disk-action row whose determinant guard and disk-local action are visible in the packet.*
3. *U is the disk-self-map row, recording that the displayed action sends admitted disk rows to admitted disk rows.*
4. *V is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *preservation row, exposing equality of the displayed hyperbolic metric reads only through the finite disk-metric and Real-facing rows.*
5. *B is the* $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$ *row, carrying the boundary readback induced by the same displayed disk map.*
6. *K is the inverse row. It displays a second disk-map packet whose two composites replay to the identity rows visible on the admitted disk observations.*
7. *I is the composition row, recording finite replay for composing two admitted automorphism packets before any group-style consumer may read the result.*

8. *J is the refusal ledger excluding host analytic continuation, completed automorphism groups, quotient boundary equality, selected global normal forms, and unrecorded Schwarz–Pick equality.*
9. *H records componentwise* hsame *transport for disk-metric, Mobius, disk-map, hyperbolic-metric, boundary-transport, inverse, composition, refusal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a disk-map request through Mobius coefficient inspection, disk admission, metric preservation, boundary transport, inverse and composition checks, refusal, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{MobiusTransform}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphism}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, U, V, B, K, I, J, H, C, P, N$. It has no coordinate for a host analytic automorphism group, quotient boundary equality, selected normal form, unbounded complex-analysis theorem, or off-packet inverse.

Theorem 6776.2 (Poincare disk automorphism NameCert obligations)**.** *For every accepted* PoincareDiskAutomorph *carrier*

$$A = (D, M, U, V, B, K, I, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphism}^{\uparrow}}$ *surface consists exactly of Poincare disk metric admission through D, Mobius coefficient and disk-action admission through M, disk self-map admission through U, hyperbolic metric preservation through V, boundary transport through B, inverse replay through K, composition replay through I, refusal exposure through J, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 6776.1. No obligation row asserts a host automorphism group, analytic classification theorem, quotient boundary equality, selected normal form, or arbitrary complex-analysis principle.*

Proof. Project the carrier of Definition 6776.1. The visible map data are the disk-metric source D, the Mobius coefficient row M, and the disk self-map row U.

The metric and boundary obligations are then read from the same displayed map. The row V compares only finite $\mathsf{HyperbolicMetric}^{\uparrow}$ reads, while B records the induced boundary transport without exposing a quotient boundary object.

The inverse and composition rows K, I are replay obligations over displayed disk observations. They make the automorphism surface reusable by downstream consumers, but they do not construct a completed automorphism group or select a global normal form.

Finally, J, H, C, P, N attach refusal, transport, replay, provenance, and local naming to the same finite rows. Since no carrier coordinate stores host complex analysis, quotient boundary equality, selected normal forms, or an off-packet inverse, none of those objects can become a NameCert obligation. □

Theorem 6776.3 (Poincare disk automorphism metric-boundary route)**.** *Every* $\mathsf{HyperbolicMetric}^{\uparrow}$*-facing,* $\mathsf{HyperbolicBoundaryTransport}^{\uparrow}$*-facing, or composition-facing consumer of an accepted* PoincareDiskAutomorph *packet must read the finite route*

$$\mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{MobiusTransform}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicBoundaryTransport}^{\uparrow}.$$

The metric-preservation row and the boundary-transport row are downstream of the displayed disk-map row and upstream of inverse and composition replay. The route exports no host automorphism group, quotient ideal-boundary equality, selected disk automorphism normal form, or analytic classification outside the packet.

Proof. Start with the disk-map request. The packet first exposes the $\mathsf{PoincareDiskMetric}^{\uparrow}$ disk observations and the $\mathsf{MobiusTransform}^{\uparrow}$ coefficient action, then records disk self-map admission through U.

The hyperbolic metric and boundary rows are read next. The row V can compare metric values only after the disk self-map is visible, and the row B carries only the boundary transport induced by that same displayed map.

The inverse and composition rows are therefore downstream replay data. Rows K and I can be consumed only after the metric-preservation and boundary-transport surface has been named, and the structural rows keep the same finite carrier under transport and provenance. Thus the consumer route remains internal to BEDC NameCert data rather than to host complex analysis. □

Falsifiable prediction: An accepted $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ consumer is refuted if it reads hyperbolic metric preservation, boundary transport, inverse replay, or composition replay without displaying PoincareDiskMetric, MobiusTransform, HyperbolicMetric, HyperbolicBoundaryTransport, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: poincare-disk-metric, mobius-transform, hyperbolic-metric, hyperbolic-boundary-transport: $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk-map, metric-preservation, boundary-transport, inverse, composition, refusal, transport, replay, provenance, and local naming as one automorphism packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphism}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6776.1, Definition 6776.2, and Definition 6776.3 seed the finite disk-map, Mobius, hyperbolic-metric, boundary-transport, inverse, composition, refusal, transport, replay, provenance, and local NameCert surface for $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host complex analysis, completed automorphism groups, quotient ideal-boundary equality, selected disk automorphism normal forms, analytic classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, Mobius coefficient transport, hyperbolic metric preservation, boundary transport, inverse replay, composition replay, refusal exposure, provenance exhaustion, and local naming for BEDC.Derived.PoincareDiskAutomorphismUp.

Constructive story: A $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ packet is generated from PoincareDiskMetric rows, MobiusTransform rows, disk self-map admission, HyperbolicMetric preservation, HyperbolicBoundaryTransport rows, inverse and composition replay, refusal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6777

A Concrete Naming Certificate for $\mathsf{HedgehogSpace}^{\uparrow}$

Following Kelley and Engelking on hedgehog spaces, the $\mathsf{HedgehogSpace}^{\uparrow}$ packet names a finite-spine point-set topology surface rather than an arbitrary-cardinal wedge. It sits after the metric carrier of chapter 57, the topology carrier of chapter 88, the finite real-distance readback of chapter 38, and the visible basepoint discipline already used by finite point-set packets. The packet records a common basepoint, finitely displayed spines, branch coordinates, metric and topology source rows, real distance rows, transport, replay, provenance, and local naming; it does not claim arbitrary-cardinal hedgehogs, choice-selected spine families, or a host topology of all wedges.

Definition 6777.1 (Hedgehog space carrier)**.** *A* $\mathsf{HedgehogSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, b, X, Q, M, T, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite spine-label ledger. Each accepted spine label names one visible branch and no off-packet branch may be read.*
2. *b is the common basepoint row shared by all displayed spines.*
3. *X is the branch-coordinate row, recording for each displayed point either the basepoint or a visible spine label together with a finite interval coordinate.*
4. *Q is the branch-equality and basepoint-collapse ledger. It identifies points from different spines only at b and otherwise compares coordinates inside the same visible spine.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *source row for the finite wedge metric read.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row generated by the same finite branch and basepoint data.*
7. *D is the* $\mathsf{Real}^{\uparrow}$ *distance row, exposing basepoint distance, same-spine coordinate distance, and different-spine wedge distance only through finite Real-facing readback.*
8. *H records componentwise* hsame *transport for spine labels, basepoint rows, branch coordinates, equality collapse, metric source, topology source, Real distance, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a point request through finite spine inspection, basepoint collapse, metric and topology reads, Real distance readback, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{HedgehogSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HedgehogSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, b, X, Q, M, T, D, H, C, P, N$. It has no coordinate for an arbitrary set of spines, a choice-selected family of branches, a completed host wedge topology, or an unrecorded quotient equality.

Theorem 6777.2 (Hedgehog space NameCert obligations)**.** *For every accepted* $\mathsf{HedgehogSpace}^{\uparrow}$ *carrier*

$$J = (S, b, X, Q, M, T, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HedgehogSpace}^{\uparrow}}$ *surface consists exactly of finite spine-label admission, common-basepoint admission, branch-coordinate admission, basepoint-collapse exactness,* $\mathsf{Met}^{\uparrow}$ *wedge-metric source admission,* $\mathsf{Topology}^{\uparrow}$ *generated-open source admission,* $\mathsf{Real}^{\uparrow}$ *distance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 6777.1. No obligation row asserts an arbitrary-cardinal hedgehog, choice-selected spine family, host quotient topology, or completed collection of all open neighbourhoods.*

Proof. Project the carrier of Definition 6777.1. The accepted surface first displays the finite spine labels S, the common basepoint b, the branch coordinates X, and the basepoint-collapse ledger Q.

The metric and topology rows are then read from the same visible data. The row M supplies only the finite wedge-metric source, while T supplies only the open-row provenance generated from the displayed branches and their shared basepoint.

The distance row D is terminal Real-facing readback for that same wedge route. Transport and replay through H and C preserve the displayed rows componentwise, and P, N source and name them locally. Since no row stores an arbitrary spine family, selected branch choice, host quotient topology, or completed open lattice, none of those objects can become a NameCert obligation. □

Theorem 6777.3 (Hedgehog metric wedge route)**.** *Every* $\mathsf{Met}^{\uparrow}$*-facing,* $\mathsf{Topology}^{\uparrow}$*-facing, or* $\mathsf{Real}^{\uparrow}$*-distance consumer of an accepted* $\mathsf{HedgehogSpace}^{\uparrow}$ *packet must read the finite route*

$$S, b, X, Q \to M, T, D \to H, C, P, N.$$

Same-spine reads compare branch coordinates inside one visible spine; different-spine reads pass through the common basepoint row b before reaching the $\mathsf{Real}^{\uparrow}$ *distance row. The route exports no arbitrary-cardinal spine family, choice-selected branch set, host wedge topology, or quotient space external to the packet.*

Proof. Begin with the visible point request. The classifier admits a point only when the finite spine ledger, basepoint row, and branch-coordinate row display its spine and coordinate data.

Use the collapse ledger Q to separate the two metric cases. Same-spine requests remain inside a single displayed branch, while different-spine requests are forced through the common basepoint row before any distance readback is exposed.

The rows M, T, D then give the metric source, topology source, and Real-facing distance for exactly that finite wedge route. The structural rows H, C, P, N transport, replay, source, and name the same packet, so a consumer cannot replace it by a host hedgehog with an arbitrary spine set or by a selected family of branches. □

Falsifiable prediction: An accepted $\mathsf{HedgehogSpace}^{\uparrow}$ consumer is refuted if it reads a non-displayed spine, a choice-selected branch family, a host quotient topology, or a Real distance not routed through the finite spine labels, common basepoint, branch coordinates, MetricUp source, TopologyUp source, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: metric, topology, real, wedge: $\mathsf{HedgehogSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite spine labels, a common basepoint, branch-coordinate rows, basepoint-collapse exactness, MetricUp and TopologyUp source rows, RealUp distance rows, transport, replay, provenance, and local naming as one point-set topology packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HedgehogSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HedgehogSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6777.1, Definition 6777.2, and Definition 6777.3 seed the finite-spine basepoint, branch-coordinate, metric, topology, Real-distance, transport, replay, provenance, and local NameCert surface for $\mathsf{HedgehogSpace}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HedgehogSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary-cardinal hedgehogs, choice-selected spine families, host wedge topologies, completed open lattices, quotient-space equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-spine admission, basepoint-collapse exactness, branch-coordinate transport, MetricUp and TopologyUp source replay, RealUp distance readback, provenance exhaustion, and local naming for BEDC.Derived.HedgehogSpaceUp.

Constructive story: A $\mathsf{HedgehogSpace}^{\uparrow}$ packet is generated from finite spine labels, a common basepoint row, branch-coordinate rows, basepoint-collapse exactness, MetricUp and TopologyUp sources, RealUp distance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6778

A Concrete Naming Certificate for $\mathsf{BishopGraphCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completions, the $\mathsf{BishopGraphCompletion}^{\uparrow}$ packet records the finite graph-read surface by which a metric or real-facing completion consumer may inspect vertices, approximation edges, and regular rational labels before any completed point is exposed. It sits after the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric carrier of chapter 57, and the continuous-map handoff of chapter 130. The packet is a finite NameCert carrier for graph-shaped completion reads; it is not a host graph-completion oracle, quotient stream equality, selected limit, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6778.1 (Bishop graph-completion carrier)**.** *A* $\mathsf{BishopGraphCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, E, W, Q, R, M, F, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite vertex row, listing the named approximation states available to the completion consumer.*
2. *E is the directed edge row, recording the admissible refinement arrows between vertices.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which vertex and edge reads are requested.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational-readback row attached to the same finite windows.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after V, E, W, Q have been consumed.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose displayed distance comparisons govern accepted refinement edges.*
7. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row, recording graph-compatible maps that preserve displayed refinement and metric reads.*
8. *L is the local graph-completion ledger, pairing each accepted consumer request with a finite path in E and a terminal Real-facing seal in R.*
9. *H records componentwise* hsame *transport for vertices, edges, windows, rational readbacks, metric comparisons, continuous-map handoffs, ledger rows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a completion request through the finite vertex-edge path, StreamName windows, RegSeqRat readback, metric source, continuous-map handoff, Real seal, and local ledger.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopGraphCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopGraphCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, E, W, Q, R, M, F, L, H, C, P, N$. It has no coordinate for host graph-completion oracles, quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 6778.2 (Bishop graph-completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopGraphCompletion}^{\uparrow}$ *carrier*

$$G = (V, E, W, Q, R, M, F, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopGraphCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed vertex row, directed edge row, finite StreamName windows, RegSeqRat readback, Real seal, MetricSpace source, ContinuousMap handoff, graph-completion ledger, componentwise transport, displayed replay, package provenance, and local naming. Edge stability is read from E together with M; readback stability is read from W, Q, R; map stability is read from F under H, C; and ledger nonescape keeps host graph-completion oracles, quotient streams, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking outside the packet.*

Proof. Project the carrier of Definition 6778.1. Carrier admission exposes exactly the rows $V, E, W, Q, R, M, F, L, H$ and no further completion coordinate.

The graph read is finite before it is completion-facing. A consumer request first names windows in W, then reads the vertex and edge rows V, E, and then obtains rational labels through Q over the same finite requests.

Metric and map stability are displayed rows, not ambient facts. The metric source M governs which refinement arrows are admissible, while the continuous-map row F records preservation of those displayed arrows under the same transport and replay rows H, C.

Finally the local ledger L pairs each accepted request with the finite path and Real seal already displayed by the packet. Since the carrier contains no host oracle, quotient-stream coordinate, selected-limit coordinate, countable-choice scheduler, ambient-completeness row, or bridge-checking row, the listed rows exhaust the local NameCert obligations. □

Theorem 6778.3 (Bishop graph-completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopGraphCompletion}^{\uparrow}$ *carrier receives completion-facing data only through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently the consumer may inspect the displayed vertex-edge path, rational readback, metric comparison, map handoff, transport, replay, provenance, and local naming rows, but it cannot read a host graph-completion oracle, quotient stream, selected limit, countable choice witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof from the packet.*

Proof. Open an accepted carrier $G = (V, E, W, Q, R, M, F, L, H, C, P, N)$. The completion-facing seal R is downstream of the finite windows W, the graph rows V, E, the rational readback Q, the metric source M, and the continuous-map handoff F.

A Real-facing consumer must therefore consume the same displayed route before R is available. The ledger L records the finite edge path used by the request, while H and C preserve the route under componentwise transport and replay.

The packet has no other completion field. In particular, it stores neither quotient-stream equality nor a selected host limit, and it supplies no countable-choice scheduler, host graph-completion oracle, or ambient completeness proof. The available handoff is exactly the finite route displayed in the statement. □

Falsifiable prediction: Any accepted $\mathsf{BishopGraphCompletion}^{\uparrow}$ completion-facing read that bypasses the finite edge row E, the readback row Q, or the ledger row L is rejected by Definition 6778.1, because the packet exposes Real-facing completion data only through the displayed vertex-edge route.

Independence witness: streamname, regseqrat, real, metric, continuousmap: the packet is not bijective with any listed sibling because it jointly records finite vertices, refinement edges, stream windows, regular rational readback, Real seals, metric comparisons, continuous-map handoffs, graph-completion ledgers, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopGraphCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopGraphCompletion}^{\uparrow}$ packet is generated from finite vertex rows, directed refinement-edge rows, StreamName windows, RegSeqRat readback, a RealUp seal, MetricSpace comparisons, ContinuousMap handoff rows, a graph-completion ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopGraphCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6778.1, Definition 6778.2, and Definition 6778.3 seed the finite graph-completion route for StreamName, RegSeqRat, RealUp, MetricSpace, ContinuousMap, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopGraphCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host graph-completion oracles, quotient streams, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vertex-edge path exposure, rational readback, metric-edge stability, continuous-map preservation, Real-seal nonescape, transport and replay preservation, provenance faithfulness, and local NameCert coverage for BEDC.Derived.BishopGraphCompletionUp.

Chapter 6779

A Concrete Naming Certificate for $\mathsf{CompletionExactness}^{\uparrow}$

Following Bishop-style completion exactness, the $\mathsf{CompletionExactness}^{\uparrow}$ packet records the finite boundary where a completion-facing route is accepted only when its displayed coverage, soundness, completeness, and classifier equivalence rows agree. It sits over the dyadic tolerance core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the two-read comparison boundary of chapter 6768. The packet names an exactness predicate for the L10 completion route; it does not select limits, quotient Cauchy names, identify host reals, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6779.1 (Completion exactness carrier)**.** *A* $\mathsf{CompletionExactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, W, R, S, V, K, T, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance coverage row, recording the dyadic budgets through which completion reads are inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row scheduled by D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed windows.*
4. *S is the soundness row, asserting that each accepted completion read is backed by the displayed dyadic, stream, regular-rational, and terminal* $\mathsf{Real}^{\uparrow}$ *seal data.*
5. *V is the completeness row, asserting that every displayed dyadic-window and regular-rational route admitted by the classifier reaches an accepted completion read.*
6. *K is the classification-iff row, recording that the public classifier accepts exactly the reads satisfying the coverage, soundness, and completeness rows.*
7. *T records componentwise* hsame *transport and displayed* Cont *replay for D, W, R, S, V, K.*
8. *P is the provenance row over* $\mathsf{CompletionExactness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionExactness}^{\uparrow}}$ *row.*

The classifier has no coordinate for a chosen limit, quotient representative, host equality of real names, countable-choice schedule, or unrestricted completion theorem.

Theorem 6779.2 (Completion exactness NameCert obligations)**.** *For every accepted* $\mathsf{CompletionExactness}^{\uparrow}$ *carrier*

$$E = (D, W, R, S, V, K, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionExactness}^{\uparrow}}$ *surface consists exactly of dyadic coverage through* D*, StreamName observation through* W*, regular-rational readback through* R*, soundness through* S*, completeness through* V*, classifier equivalence through* K*, transport and replay through* T*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6779.1. The rows D, W, R expose the inspected dyadic budgets, finite windows, and regular-rational readback before a completion-facing read is accepted.

Read the exactness rows next. Soundness S prevents an accepted read from escaping the displayed dyadic, stream, regular-rational, and terminal Real data, while completeness V forces every classifier-admitted displayed route to reach a read. The row K binds these two directions as the public classification equivalence.

Finally use the structural rows. Transport and replay T preserve the same finite route, provenance P records the sibling sources, and N names only that surface. Since the packet contains no row for selected limits, quotient representatives, host equality, countable choice, or unrestricted completion, the listed rows exhaust the local NameCert obligations. □

Theorem 6779.3 (Completion exactness real route)**.** *Every Real-facing completion read accepted by a* $\mathsf{CompletionExactness}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyCompletionEqualizer}^{\uparrow}.$$

The classifier equivalence row K *accepts such a read exactly when the coverage row* D*, soundness row* S*, and completeness row* V *hold over the same finite windows and readback data.*

Proof. Start with the sibling route. The dyadic tolerance row D schedules StreamName windows W, and those windows produce the regular-rational readback R before any terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Use the equalizer boundary. The $\mathsf{CauchyCompletionEqualizer}^{\uparrow}$ surface compares completion reads only after the common dyadic, stream, regular-rational, and Real-facing rows have been displayed, so the completion exactness packet has no separate quotient-equality source.

Apply the classifier equivalence row. The row K reads coverage, soundness, and completeness over the same finite route, and T, P, N merely transport, replay, source, and name that route. Hence an accepted Real-facing completion read is exact precisely along the displayed L10 completion path, without selected limits or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionExactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompletionExactness}^{\uparrow}$ packet is generated from dyadic tolerance coverage, StreamName windows, RegSeqRat readback, soundness, completeness, a classifier equivalence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionExactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6779.1, Definition 6779.2, and Definition 6779.3 seed the L10 completion exactness predicate over dyadic, StreamName, RegSeqRat, Real, CauchyCompletionEqualizer, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionExactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient Cauchy equality, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coverage, soundness, completeness, classifier equivalence, transport, replay, provenance, and local NameCert rows for BEDC.Derived.CompletionExactnessUp.

Chapter 6780

A Concrete Naming Certificate for $\mathsf{TychonoffPlank}^{\uparrow}$

Following Kelley and Steen–Seebach point-set topology, the $\mathsf{TychonoffPlank}^{\uparrow}$ packet records the closed ordinal-rectangle surface used to test compactness and product-separation arguments without importing arbitrary Tychonoff compactness. It sits beside the finite compact-product packet of **??**, the indexed-open topology carrier of chapter 88, the compactness boundary of chapter 60, and the observable-open truth discipline of chapter 1860. The packet names only the displayed long-edge rows, corner-deletion row, product-neighbourhood rows, transport, replay, provenance, and local certificate data; it does not claim unrestricted product compactness, choice-selected subcovers, quotient open equality, or a host ordinal topology.

Definition 6780.1 (Tychonoff plank carrier)**.** *A* $\mathsf{TychonoffPlank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P_{\mathsf{tp}} = (L, R, O, K, S, H, C, G, N)$$

with the following displayed rows:

1. *L is the left long-edge row, naming the finite observations of the first ordinal interval used by the plank read.*
2. *R is the right long-edge row, naming the finite observations of the second ordinal interval used by the plank read.*
3. *O is the product-neighbourhood row, read through the* $\mathsf{Topology}^{\uparrow}$ *interface for rectangular opens and their finite meet and union ledgers.*
4. *K is the compact-side row, exposing only the compact rectangle or closed-side witness currently displayed by the packet.*
5. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row used to record whether the inspected corner, edge, or deleted-corner test is visible to the finite read.*
6. *H records componentwise* hsame *transport for the two edge rows, product-neighbourhood row, compact row, observable-open row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a plank observation through L, R, O, K, S, H, G, N.*
8. *G is the* Pkg *provenance over* $\mathsf{TychonoffPlank}^{\uparrow}$, $\mathsf{TychonoffFiniteProduct}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{TychonoffPlank}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for arbitrary indexed products, unrestricted Tychonoff compactness, host ordinal spaces, selected subcovers, quotient open equality, countable choice, Lean verification, or bridge checking.

Theorem 6780.2 (Tychonoff plank NameCert obligations)**.** *For every accepted* $\mathsf{TychonoffPlank}^{\uparrow}$ *carrier* $P_{\mathsf{tp}} = (L, R, O, K, S, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TychonoffPlank}^{\uparrow}}$ *surface consists exactly of carrier admission for the two long-edge rows, the product-neighbourhood row, the compact-side row, the observable-open row, transport, replay, provenance, and local naming; product-neighbourhood readback through* $\mathsf{Topology}^{\uparrow}$*; compact-side visibility through* $\mathsf{Compact}^{\uparrow}$*; observable corner and edge tests through* $\mathsf{SierpinskiSpace}^{\uparrow}$*; componentwise* hsame *stability; displayed* Cont *replay; provenance nonescape; and local naming exactness.*

Proof. Project the carrier of Definition 6780.1. The two edge rows and the product-neighbourhood row are the only data by which a consumer can inspect the plank.

The topology-facing obligations are read through O. Rectangular neighbourhoods, finite meets, and indexed unions must therefore use the same $\mathsf{Topology}^{\uparrow}$ row before any plank open is consumed.

The compact and observable-open obligations are rowwise. The row K exposes only the displayed compact-side witness, while S records the visible corner or edge test through the $\mathsf{SierpinskiSpace}^{\uparrow}$ classifier.

Finally H, C, G, N transport, replay, source, and name exactly the same rows. Since the carrier has no coordinate for arbitrary products, selected subcovers, quotient open equality, or host ordinal topology, those objects cannot become local NameCert data. □

Closure status for $\mathsf{NameCert}_{\mathsf{TychonoffPlank}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TychonoffPlank}^{\uparrow}}$, seed)

Scope closed: Definition 6780.1 and Definition 6780.2 seed the finite Tychonoff plank packet over displayed long-edge, topology, compactness, observable-open, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TychonoffPlank}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary Tychonoff products, choice-based product compactness, host ordinal spaces, selected subcovers, quotient open equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, long-edge readback, product-neighbourhood exactness, compact-side visibility, observable corner and edge tests, transport, replay, provenance, and local naming for BEDC.Derived.TychonoffPlankUp.

Constructive story: A $\mathsf{TychonoffPlank}^{\uparrow}$ packet is generated from two long-edge rows, a TopologyUp product-neighbourhood row, a CompactUp side row, a SierpinskiSpaceUp observable-open row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6781

A Concrete Naming Certificate for $\mathsf{DeletedTychonoffPlank}^{\uparrow}$

Following Kelley and Steen–Seebach point-set topology, the $\mathsf{DeletedTychonoffPlank}^{\uparrow}$ packet records the open deleted-corner surface of the Tychonoff plank as finite NameCert data. It sits beside the closed plank reconstruction of chapter 6780, reads rectangular neighbourhoods through chapter 88, keeps its compact-side boundary visible through chapter 60, and exposes corner tests through chapter 1860. The packet names only the displayed plank row, deleted-corner row, product-boundary row, topology row, compact row, observable-open row, transport, replay, provenance, and local naming; it does not claim unrestricted product compactness, a host ordinal topology, quotient open equality, selected subcovers, or choice-based compactness.

Definition 6781.1 (Deleted Tychonoff plank carrier)**.** *A* $\mathsf{DeletedTychonoffPlank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (P, X, B, O, K, S, H, C, G, N)$$

with the following displayed rows:

1. *P is the sibling* $\mathsf{TychonoffPlank}^{\uparrow}$ *row that supplies the closed plank surface read by the deleted packet.*
2. *X is the deleted-corner row, naming the removed top corner as displayed boundary data rather than as a host ordinal point.*
3. *B is the product-boundary row, recording which long-edge and corner-neighbourhood observations remain visible after the deletion.*
4. *O is the* $\mathsf{Topology}^{\uparrow}$ *row for rectangular neighbourhood reads, finite meet rows, finite union rows, and boundary-open observations.*
5. *K is the* $\mathsf{Compact}^{\uparrow}$ *row for the compact side or closed-rectangle witness inherited from the sibling plank packet.*
6. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row that records whether the deleted corner, remaining edge, or boundary neighbourhood is visible to a finite read.*
7. *H records componentwise* hsame *transport for $P, X, B, O, K, S, C, G, N$.*
8. *C records displayed* Cont *replay from a deleted-plank observation through $P, X, B, O, K, S, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{DeletedTychonoffPlank}^{\uparrow}$*,* $\mathsf{TychonoffPlank}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{SierpinskiSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{DeletedTychonoffPlank}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $P, X, B, O, K, S, H, C, G, N$. It has no coordinate for unrestricted product compactness, a selected subcover, quotient open equality, a host ordinal topology, countable choice, Lean verification, or bridge checking.

Lean4 (statement) `BEDC.Derived.DeletedTychonoffPlankUp`

Theorem 6781.2 (Deleted Tychonoff plank NameCert obligations)**.** *For every accepted* $\mathsf{DeletedTychonoffPlank}^{\uparrow}$ *carrier $D = (P, X, B, O, K, S, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{DeletedTychonoffPlank}^{\uparrow}}$ *surface consists exactly of carrier admission for the sibling plank row, deleted-corner row, product-boundary row, topology row, compact row, observable-open row, transport, replay, provenance, and local naming; deletion visibility through X and S; rectangular neighbourhood readback through O; compact-side visibility through K; product-boundary stability through B and H; displayed* Cont *replay; provenance nonescape; and local naming exactness.*

Proof. Project the carrier of Definition 6781.1. The rows P, X, B are the only data by which a consumer can inspect the closed plank, the deleted corner, and the remaining product boundary.

The topology-facing obligations are read through O and S. Rectangular neighbourhoods and finite Boolean operations use the $\mathsf{Topology}^{\uparrow}$ row, while the visible deleted-corner and edge tests use the $\mathsf{SierpinskiSpace}^{\uparrow}$ row.

The compact-side obligation is rowwise. The row K exposes only the compact-side witness inherited by the packet, and B records which boundary observations remain available after the deleted corner is removed.

Finally H, C, G, N transport, replay, source, and name exactly the same rows. Since the carrier has no coordinate for unrestricted product compactness, selected subcovers, quotient open equality, host ordinal topology, or countable choice, those objects cannot become local NameCert data. □

Theorem 6781.3 (Deleted Tychonoff plank product-boundary handoff)**.** *Every product-boundary read exported by an accepted* $\mathsf{DeletedTychonoffPlank}^{\uparrow}$ *packet factors through*

$$P, X, B \longmapsto O, K, S \longmapsto H, C, G, N.$$

The handoff first exposes the sibling $\mathsf{TychonoffPlank}^{\uparrow}$ *row, the deleted-corner row, and the finite product-boundary row, then reads the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row, the* $\mathsf{Compact}^{\uparrow}$ *side row, and the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row before any public consumer can use the route.*

Proof. Begin with the carrier projection. A product-boundary consumer can only start from P, X, B, so the closed plank, deleted corner, and remaining edge observations are fixed as displayed rows before any topology-facing read occurs.

The rows O, K, S then expose the neighbourhood, compact-side, and observable-open interfaces. This makes the boundary handoff a finite product-neighbourhood route rather than an appeal to unrestricted product compactness or a host ordinal topology.

The structural rows H, C, G, N preserve the same route by componentwise transport, displayed replay, provenance, and local naming. Therefore the exported boundary remains the displayed finite packet and cannot be replaced by quotient open equality, a selected subcover, or countable choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeletedTychonoffPlank}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DeletedTychonoffPlank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6781.1, Definition 6781.2, and Definition 6781.3 seed the finite deleted-plank packet over displayed plank, deleted-corner, product-boundary, topology, compactness, observable-open, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DeletedTychonoffPlank}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.DeletedTychonoffPlankUp`

Bridge status: none

Not claimed: This chapter does not close unrestricted product compactness, host ordinal topology, quotient open equality, selected subcovers, countable choice, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deleted-corner visibility, topology readback, compact-side visibility, observable-open tests, product-boundary stability, handoff factoring, transport, replay, provenance, and local naming for BEDC.Derived.DeletedTychonoffPlankUp.

Constructive story: A DeletedTychonoffPlank$^{\uparrow}$ packet is generated from a TychonoffPlankUp row, deleted-corner row, product-boundary row, TopologyUp neighbourhood row, CompactUp side row, SierpinskiSpaceUp observable-open row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6782

A Concrete Naming Certificate for $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ packet records a paired lower-and-upper finite-net extreme-value certificate for uniformly continuous maps on compact metric sources. It is the finite-net companion to chapter 6770 and sits over the finite-net handoff of chapter 6769, the compact source row of chapter 129, the uniformly continuous map row of chapter 130, the stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the attained-extreme comparison surface of chapter 3984. The packet exposes located upper and lower value windows without selecting extrema, importing open-cover compactness, identifying quotient real names, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6782.1 (Located uniform extreme-value carrier)**.** *A* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, N, W, R, E, L, U_b, M, H, C, P, Q)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the compact metric inspection surface consumed by finite-net reads.*
2. *F is the uniformly continuous* $\mathsf{ContinuousMap}^{\uparrow}$ *row, exposing the value request interface over K.*
3. *N is the* $\mathsf{HeineCantorFiniteNet}^{\uparrow}$ *row, recording the finite center family, radius schedule, and coverage witness through which every extreme-value request enters.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite family of value windows attached to the centers of N.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational lower and upper endpoints exposed by W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed value names before any comparison row is consumed.*
7. *L is the located lower-bound row, comparing each displayed center value with the finite family through rational lower windows.*

8. *U_b is the located upper-bound row, comparing each displayed center value with the finite family through rational upper windows.*
9. *M is the paired extreme-value ledger row, recording that the lower and upper rows share the same source, map, finite net, value windows, rational readback, and Real seal.*
10. *H records componentwise* hsame *transport for the compact source, map row, finite-net cover, stream schedule, rational readback, Real seal, lower row, upper row, paired ledger, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from an extreme-value request through $K, F, N, W, R, E, L, U_b, M, H, P, Q$.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$, $\mathsf{BishopLocatedMaximum}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantorFiniteNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *Q is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformExtremeValue}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed compact source, uniformly continuous map row, finite-net cover, value-window schedule, regular rational readback, Real seal, lower and upper located comparison rows, shared ledger, transport, replay, provenance, and local naming. It has no coordinate for a selected minimum, selected maximum, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.LocatedUniformExtremeValueUp`

Theorem 6782.2 (Located uniform extreme-value obligations)**.** *For every accepted* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ *carrier $U = (K, F, N, W, R, E, L, U_b, M, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformExtremeValue}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed compact source, uniformly continuous map row, finite-net cover, value windows, rational readback, Real seal, lower row, upper row, paired ledger, transport, replay, provenance, and local naming;*
2. *finite-net exposure, requiring every extreme-value request to pass through the centers and coverage witnesses named by N;*
3. *value readback through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *requiring W, R, E to be visible before either bound row is consumed;*
4. *lower-bound located comparison through L and upper-bound located comparison through U_b;*
5. *shared-ledger exactness through M, requiring both bound rows to use the same finite net and value-readback route;*
6. *refusal of selected extrema, open-cover compactness, quotient real equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6782.1. The rows K, F, N expose the compact source, map row, and finite-net coverage surface that every later value read must use.

Read the value rows next. The stream schedule W supplies finite lower and upper windows, R reads those windows as regular rational data, and E seals the displayed readbacks as $\mathsf{Real}^{\uparrow}$ names before a comparison can be used.

The lower row L and upper row U_b can consume only the Real-sealed finite family already displayed by N, W, R, E. The shared ledger M then records that both comparisons used the same

route, while H, C, P, Q transport, replay, source, and name that finite route. These rows exhaust the listed obligations. □

Theorem 6782.3 (Located uniform extreme-value finite-net exposure)**.** *Every extreme-value read exported by an accepted* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ *carrier factors through the displayed finite route*

$$K, F, N \longmapsto W, R, E \longmapsto L, U_b, M \longmapsto H, C, P, Q.$$

Thus a consumer can inspect only the compact source, uniformly continuous map row, finite net, value windows, located lower and upper comparisons, and Real seal before reaching the local naming row.

Proof. Start with the source and map rows. The compact metric source K and uniformly continuous map row F become visible to an extreme-value read only through the finite-net cover N.

The value route is fixed before either bound row appears. The stream schedule W records finite observation windows attached to the net, the regular rational row R reads those windows, and the terminal seal E is reached before any extreme-value comparison is available.

The located lower row L and upper row U_b compare the same Real-sealed finite family. The ledger M prevents one bound from using a different net or value-readback route. Replaying the same rows through H, C, P, Q gives the displayed factorization. □

Theorem 6782.4 (Located uniform extreme-value bounds)**.** *An accepted* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ *carrier exposes paired lower and upper value windows only as located finite-net comparisons. It does not expose an argmin, an argmax, a quotient equality class for real names, an open-cover compactness proof, a choice functional, or an ambient completeness principle.*

Proof. Inspect the excluded coordinates in Definition 6782.1. The carrier contains rows for L, U_b, and M, but no coordinate that names a minimizing point, maximizing point, quotient real equality token, open-cover proof, choice functional, or completeness witness.

Apply Definition 6782.2. The only comparison obligations are the lower row, the upper row, and the shared finite-net ledger. Therefore any accepted bound read is a located finite-net comparison read, not an extraction of one of the excluded objects. □

Theorem 6782.5 (Located uniform extreme-value NameCert route)**.** *The* $\mathsf{LocatedUniformExtremeValue}^{\uparrow}$ *NameCert route is the paired-bound route obtained by adding an upper located comparison row to the finite-net maximum-location route of* $\mathsf{BishopLocatedMaximum}^{\uparrow}$ *while keeping the same compact source, uniformly continuous map, finite net, value-window schedule, rational readback, and Real seal interface.*

Proof. Use the sibling route of chapter 6770 for the compact source, map row, finite net, value-window schedule, rational readback, and Real seal. The present carrier keeps those rows as K, F, N, W, R, E.

Add the lower and upper located comparison rows L and U_b and bind them with the shared ledger M. The replay and provenance rows H, C, P, Q then name the same finite route for both bounds, so the NameCert route is precisely the paired-bound route described in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformExtremeValue}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformExtremeValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6782.1, Definition 6782.2, Definition 6782.3, Definition 6782.4, and Definition 6782.5 seed the paired finite-net lower-and-upper extreme-value route over compact-source, uniformly continuous map, value-window, rational-readback, located-comparison, and Real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedUniformExtremeValue}^{\uparrow}}$, formalTarget)
Lean target: `BEDC.Derived.LocatedUniformExtremeValueUp`
Bridge status: none
Not claimed: This chapter does not close selected extrema, full open-cover compactness, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, axiom-purity certification, or bridge checking for LocatedUniformExtremeValue$^{\uparrow}$.
Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, StreamName-RegSeqRat-Real value readback, paired lower and upper located comparisons, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformExtremeValueUp.
Constructive story: A LocatedUniformExtremeValue$^{\uparrow}$ packet is generated from a CompactMetric source row, a uniformly continuous ContinuousMap row, a finite-net cover, StreamName value windows, RegSeqRat readbacks, a Real seal, located lower and upper comparison rows, a shared finite-net ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6783

A Concrete Naming Certificate for $\mathsf{BishopLocatedOscillation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedOscillation}^{\uparrow}$ packet records the finite NameCert route by which a continuous map on compact metric data exposes located oscillation bounds through dyadic windows and regular rational readback. It sits beside the compact finite-net carrier of chapter 129, the map carrier of chapter 130, the dyadic and stream naming rows of chapter 291 and chapter 126, the regular rational and Real readback rows of chapter 301 and chapter 38, and the explicit oscillation schedule of chapter 2224. The packet is an L10 finite-window certificate: it does not select extrema, invoke quotient equality, use Real completeness, or import choice.

Definition 6783.1 (Bishop located oscillation carrier)**.** *A* $\mathsf{BishopLocatedOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\mathrm{osc}} = (K, F, D, S, Q, R, O, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, exposing only the bundled compact carrier, its finite coverage histories, and the named metric comparisons used by the oscillation request.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording the map whose local oscillation is read at the displayed compact windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the target oscillation scale and the source-window cuts.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the finite stream of windows inspected by the request.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream and regular rational rows have been displayed.*
7. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row translating the local oscillation read into an explicit tolerance-window schedule.*
8. *W is the located window row, the finite ledger that records which compact windows have been inspected and which dyadic bound each read returns.*
9. *H records componentwise* hsame *transport for the compact, map, dyadic, stream, regular-rational, Real, oscillation-modulus, located-window, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tolerance request through K, F, D, S, Q, R, O, W and back to the exported located oscillation row.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedOscillation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows and the displayed routes among them. It has no coordinate for selected extrema, quotient equality of streams, open-cover compactness, Real completeness, countable choice, or bridge-axis evidence.

Theorem 6783.2 (Bishop located oscillation NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedOscillation}^{\uparrow}$ *carrier $L_{\mathrm{osc}} = (K, F, D, S, Q, R, O, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the compact metric row, continuous map row, dyadic tolerance row, stream-name row, regular rational row, Real seal, oscillation-modulus row, located-window row, transport row, replay row, provenance row, and local naming row;*
2. *located-window coverage, requiring each oscillation read to name one displayed compact window from W before any Real-facing value is exposed;*
3. *dyadic precision visibility, requiring the target tolerance and source-window cut to be read from D and then replayed through S, Q, R;*
4. *oscillation-modulus handoff, requiring O to consume the same K, F, D, S, Q, R, W route before exporting the tolerance-window schedule;*
5. *ledger non-escape, requiring H, C, P, N to export no selected extrema, quotient equality of streams, open-cover compactness, Real completeness, countable choice, or bridge-axis evidence.*

No local obligation is stored outside those rows.

Proof. Project the accepted carrier. Its analytic rows are exactly K, F, D, S, Q, R, O, W, and its structural rows are exactly H, C, P, N.

Carrier admission is therefore componentwise. The compact row supplies only the finite coverage histories, the map row supplies only the named continuous-map read, and the dyadic, stream, regular rational, and Real rows supply the displayed readback route.

The located-window row is the only place where a compact window is named for the oscillation request. The oscillation-modulus row consumes that same route, so the exported schedule cannot bypass K, F, D, S, Q, R, W.

Transport, replay, provenance, and local naming preserve the same rows. Since no row stores selected extrema, quotient equality, open-cover compactness, Real completeness, countable choice, or bridge data, none of those objects can become a local obligation. □

Theorem 6783.3 (Bishop located oscillation finite-window handoff)**.** *Every Real-facing or compact-uniform consumer of an accepted* $\mathsf{BishopLocatedOscillation}^{\uparrow}$ *packet reads oscillation data through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{OscillationModulus}^{\uparrow},\ \mathsf{Bisho}$$

Consequently the consumer receives a located finite-window oscillation certificate and not a selected maximum, quotient equality of streams, open-cover compactness theorem, Real completeness principle, countable-choice schedule, or bridge certificate.

Proof. Begin with the accepted packet. By Definition 6783.1, every consumer-visible read of oscillation data must pass through the compact row K, the map row F, the dyadic tolerance row D, the stream row S, the regular rational row Q, the Real seal R, the oscillation-modulus row O, and the located-window ledger W.

The replay row C orders the route from the tolerance request to the exported schedule, while H transports the same components and P, N source and name them. Thus a Real-facing read and a compact-uniform read see the same finite-window handoff.

Because the carrier has no coordinate for selected extrema, quotient stream equality, open-cover compactness, Real completeness, choice, or bridge data, the handoff cannot import those principles. □

Falsifiable prediction: Every $\mathsf{BishopLocatedOscillation}^{\uparrow}$ consumer read factors through one CompactMetric row, one ContinuousMap row, one DyadicRatCore tolerance row, one StreamName window row, one RegSeqRat readback row, one RealUp seal, one OscillationModulus row, one located-window ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: compactmetric, continuousmap, dyadicratcore, streamname, regseqrat, real, oscillation-modulus: $\mathsf{BishopLocatedOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-window admission, continuous-map oscillation reads, dyadic precision, stream naming, regular rational readback, Real sealing, oscillation-modulus export, located-window bookkeeping, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedOscillation}^{\uparrow}$ packet is generated from CompactMetric finite-window data, a ContinuousMap row, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, a RealUp seal, an OscillationModulus row, a located-window ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6783.1, Definition 6783.2, and Definition 6783.3 seed the finite-window located oscillation certificate used by Real-facing and compact-uniform consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected extrema, quotient equality of streams, open-cover compactness, Real completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window coverage, dyadic precision visibility, oscillation-modulus handoff, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.BishopLocatedOscillationUp.

Chapter 6784

A Concrete Naming Certificate for $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$

Following Bishop and Bridges on nested intervals and constructive real approximation, the DyadicNestedIntervalSelec packet records the finite selector surface used when a displayed nested dyadic interval argument is handed to Real and Completion consumers. It sits over the nested-chain surface of chapter 320, the center-radius ledger of chapter 291, the finite window source of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names only the finite displayed rows by which a selected window is inspected; it does not assert interval compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a selected completed real.

Definition 6784.1 (Dyadic nested interval selector carrier)**.** *A* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the displayed* $\mathsf{NestedInterval}^{\uparrow}$ *chain whose retained cells are being inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *center and radius ledger attached to the same retained cells.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the requested stages of the chain.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the same finite windows with regular rational observations.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the nested-chain, dyadic-ledger, stream-window, readback, seal, provenance, route, and local naming rows.*
7. *C records displayed* Cont *replay from a dyadic selector request through I, D, S, R, E and back to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalSelector}^{\uparrow}}$ *row.*

The carrier exports only this finite displayed selector packet. It contains no coordinate for a host choice function, selected completed real, interval compactness theorem, countable-choice schedule, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicNestedIntervalSelectorUp.DyadicNestedIntervalSelectorTasteGate`
`single carrier alignment`

Theorem 6784.2 (Dyadic nested interval selector NameCert obligations)**.** *For every accepted* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *carrier*

$$K = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalSelector}^{\uparrow}}$ *surface consists exactly of nested-chain exposure through* I*, dyadic center-radius exposure through* D*, finite-window exposure through* S*, regular rational readback through* R*, terminal seal admission through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. Any selector-facing consumer must read those rows before using the finite nested dyadic interval route.*

Proof. Project the carrier of Definition 6784.1. The nested chain, dyadic center-radius ledger, stream window, regular readback, Real seal, transport, replay, provenance, and local name are the complete visible packet.

The selector route first reads I and D. Thus the retained interval stage and its dyadic center-radius data are displayed before any window or Real-facing read is available.

Next read S and R. The stream row names the finite inspected stages, while the regular rational row supplies the readback for those stages. The seal row E is downstream of that finite route.

The rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no host selector, compactness theorem, countable-choice schedule, quotient equality, or ambient completeness coordinate, none of those principles can be exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.DyadicNestedIntervalSelectorUp.DyadicNestedIntervalSelectorNamecertObligations`

Theorem 6784.3 (Dyadic nested interval selector Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *packet factors through*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the finite rows $I, D, S, R, E, H, C, P, N$*. The handoff may transport and replay the displayed selector route, but it cannot claim* $\mathsf{Real}^{\uparrow}$ *completeness, interval compactness, countable choice, a host-selected endpoint, quotient stream equality, or a selected completed real.*

Lean4 ✓ `BEDC.Derived.DyadicNestedIntervalSelectorUp.DyadicNestedIntervalSelectorRealHandoff`

Proof. Begin with the obligation theorem. It forces the Real-facing request to expose the nested-chain row, dyadic center-radius ledger, finite stream window, and regular rational readback before the seal row can be consumed.

The sibling chapters supply the named surfaces of that route. The nested interval chapter supplies I, the dyadic rational core supplies D, the stream and regular rational chapters supply S and R, and the Real chapter supplies the terminal seal row E.

Finally use the structural rows. Componentwise hsame transport and displayed Cont replay preserve the same finite route, while P and N source and name it. No row remains from which a consumer could recover ambient completeness, compactness, choice, a host endpoint selector, quotient stream equality, or a completed selected real. □

Theorem 6784.4 (Dyadic nested interval selector prefix seal)**.** *Let $K = (I, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *carrier. Every prefix-window restriction of the selector packet preserves the route*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it exposes the terminal seal row E. Thus a Real-facing consumer cannot skip the nested interval row I, the dyadic ledger D, the finite stream window S, or the regular rational readback R by appealing to interval compactness, selected endpoints, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6784.1. A prefix-window restriction can only retain rows already displayed by the packet, so its visible selector data begins with I, D, S, R.

Use the NameCert obligations of Definition 6784.2. They force nested-chain exposure through I, dyadic center-radius exposure through D, finite-window exposure through S, and regular rational readback through R before the seal row can be consumed.

Apply the Real handoff theorem Definition 6784.3. It identifies the terminal route through $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with H, C, P, N preserving the same finite prefix under transport, replay, provenance, and naming.

No carrier coordinate supplies interval compactness, a selected endpoint, quotient stream equality, countable choice, or ambient Real completeness. The prefix seal therefore remains a finite displayed route rather than a completed-host selection principle. □

Lean4 ✓ `BEDC.Derived.DyadicNestedIntervalSelectorUp.DyadicNestedIntervalSelectorPrefixSeal`

Theorem 6784.5 (Dyadic nested interval selector nested-chain dependency)**.** *Every selector read from an accepted* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *carrier must expose the* $\mathsf{NestedInterval}^{\uparrow}$ *row I before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and* $\mathsf{Real}^{\uparrow}$ *seal E can be consumed. No selector route can bypass* $\mathsf{NestedInterval}^{\uparrow}$ *by appealing to compactness, selected endpoints, quotient equality, countable choice, or ambient completeness.*

Proof. Project the carrier in Definition 6784.1. The nested-chain row I is the first analytic coordinate of the selector packet, and the rows D, S, R, E are downstream of the retained chain.

Use the NameCert obligations of Definition 6784.2. They identify nested-chain exposure through I as part of the local certificate surface before any finite window, readback, or seal is available.

Apply the prefix seal theorem Definition 6784.4. It keeps every prefix-window restriction on the route through I, D, S, R, E, so compactness, selected endpoints, quotient equality, countable choice, and ambient completeness have no bypass coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicNestedIntervalSelectorUp.DyadicNestedIntervalSelectorNestedChainDependency`

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ packet is generated from a displayed NestedIntervalUp chain, a DyadicRatCoreUp center-radius ledger, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6784.1, Definition 6784.2, Definition 6784.3, and Definition 6784.5 seed the finite selector route for displayed nested dyadic interval arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, interval compactness, countable choice, host-selected endpoints, quotient stream equality, selected completed reals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-chain exposure, dyadic center-radius exposure, finite-window readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicNestedIntervalSele

Chapter 6785

A Concrete Naming Certificate for $\mathsf{FinePartitionRefinement}^{\uparrow}$

Following Bishop and Bridges, and the Cousin gauge-integration route for tagged subdivisions, the $\mathsf{FinePartitionRefinement}^{\uparrow}$ packet names the finite operation that takes an already displayed fine tagged partition and a displayed refinement request to a common fine refinement. It sits between the fine tagged partition surface of chapter 6467, the local finite cover route of chapter 3426, and the Darboux–Riemann comparison handoff of chapter 5894. The packet records only interval source rows, source and target tagged-cell ledgers, refinement maps, gauge-fineness preservation, mesh comparison, handoff, transport, replay, provenance, and local naming; it does not assert an ambient partition lattice, selected refinement limit, quotient equality of tagged partitions, choice-selected tags, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6785.1 (FinePartitionRefinement carrier)**.** *A* $\mathsf{FinePartitionRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, G, S, T, \rho, L, M, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval source shared by the refined cells.*
2. *G is the positive gauge row used to test every target cell.*
3. *S is the source* $\mathsf{FinePartitionGauge}^{\uparrow}$ *tagged-cell ledger.*
4. *T is the target tagged-cell ledger after refinement.*
5. *ρ is the finite refinement map sending every target cell of T to a source cell of S with endpoint containment and tag compatibility.*
6. *L is the fineness-preservation row, recording that each target cell remains G-fine through the row ρ.*
7. *M is the mesh comparison row, displaying the finite dyadic ledger that bounds the target mesh by the source mesh and by the requested refinement tolerance.*
8. *U is the* $\mathsf{CousinCover}^{\uparrow}$ *handoff row witnessing that the refined cells still lie on the displayed local cover route.*
9. *Q is the* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *handoff row consumed by Darboux and Riemann-sum comparison packets.*

10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the finite mesh and handoff rows have been exposed.*
11. *H records componentwise* hsame *transport for interval, gauge, source partition, target partition, refinement map, fineness, mesh, handoff, seal, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from a refinement request through the finite rows above.*
13. *P is the* Pkg *provenance over* $\mathsf{FinePartitionRefinement}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, RiemannRefinementCr $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{FinePartitionRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, S, T, \rho, L, M, U, Q, E, H, C, P, N$. It has no coordinate for a host refinement relation, an infinite partition net, quotient tagged-partition equality, selected mesh limit, hidden tag choice, or ambient completeness theorem.

Theorem 6785.2 (FinePartitionRefinement NameCert obligations)**.** *For every accepted* $\mathsf{FinePartitionRefinement}^{\uparrow}$ *carrier*

$$R = (I, G, S, T, \rho, L, M, U, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinePartitionRefinement}^{\uparrow}}$ *surface consists exactly of interval and gauge admission, source* $\mathsf{FinePartitionGauge}^{\uparrow}$ *admission, target tagged-cell admission, finite refinement-map exactness, fineness preservation, mesh comparison over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$ *locality,* RiemannRefinementCriteri *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the accepted carrier. The source data are I, G, S, where S is already a $\mathsf{FinePartitionGauge}^{\uparrow}$ packet over the displayed interval and gauge.

The target data are T, ρ, L, M. The map ρ records finite containment of target cells in source cells, L records preservation of gauge-fineness along that containment, and M records the dyadic mesh comparison. These rows are the only refinement operation stored in the carrier.

The handoff rows U and Q then expose the same refined partition to the Cousin cover and Riemann refinement criterion siblings. The remaining rows transport, replay, source, and name the packet. Since no coordinate stores a host refinement lattice, hidden selector, quotient equality, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6785.3 (FinePartitionRefinement handoff)**.** *Every accepted* $\mathsf{FinePartitionRefinement}^{\uparrow}$ *packet turns a displayed* $\mathsf{FinePartitionGauge}^{\uparrow}$ *source row into a finite target refinement before any* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *consumer may read the Darboux–Riemann comparison route. The consumer receives only*

$$\mathsf{FinePartitionGauge}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow} \longmapsto T, \rho, L, M \longmapsto \mathsf{RiemannRefinementCriterion}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

with $\mathsf{DyadicRatCore}^{\uparrow}$ *controlling the finite mesh ledger.*

Proof. Start from the carrier of Definition 6785.1. The source row S is read as a fine tagged partition through chapter 6467, while U keeps the refined cells on the Cousin cover route of chapter 3426.

The finite target rows T, ρ, L, M expose the actual operation: a displayed target cell ledger, a displayed refinement map, a fineness-preservation row, and a dyadic mesh comparison row. No downstream consumer can read Q before these rows have been presented.

Finally Q is the $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ handoff, so the Darboux–Riemann comparison receives a finite refinement packet rather than a host-level partition principle. The terminal seal E and the structural rows H, C, P, N preserve only this route. □

Falsifiable prediction: Every accepted $\mathsf{FinePartitionRefinement}^{\uparrow}$ consumer must expose a source FinePartitionGauge row, target tagged-cell ledger, finite refinement map, fineness-preservation row, dyadic mesh comparison, CousinCover handoff, and RiemannRefinementCriterion handoff before using the refinement.

Independence witness: finepartitiongauge, cousincover, riemannrefinementcriterion, dyadicratcore: $\mathsf{FinePartitionRefinement}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite refinement map, target tagged-cell ledger, fineness preservation, mesh comparison, and two downstream handoff rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FinePartitionRefinement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FinePartitionRefinement}^{\uparrow}$ packet is generated from interval and gauge rows, a source FinePartitionGauge row, a target tagged-cell ledger, finite refinement maps, fineness-preservation rows, DyadicRatCore mesh comparisons, CousinCover and RiemannRefinementCriterion handoffs, RealUp sealing, transport, replay, provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinePartitionRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6785.1, Definition 6785.2, and Definition 6785.3 seed the finite refinement-operation carrier between fine partitions and Riemann refinement consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinePartitionRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host partition lattices, selected mesh limits, quotient tagged-partition equality, hidden tag choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement-map exactness, fineness preservation, DyadicRatCore mesh comparison, CousinCover locality, RiemannRefinementCriterion handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FinePartitionRefinementUp.

Chapter 6786

A Concrete Naming Certificate for EnergyRefinedPrivateCoordinate$^{\uparrow}$

The EnergyRefinedPrivateCoordinate$^{\uparrow}$ packet realizes the energy-refined private-coordinate formula of Definition 9462.1675 as finite NameCert data. It refines the minimal-cover surface of chapter 5812 by adding an incidence-pattern energy ledger, a polynomial coefficient ledger, and a trigger-size projection before the enumerator may be consumed. The packet keeps the BEDC-discovered enumeration route separate from bare finite-set covers, Boolean tables, natural-number sequences, and polynomial syntax without private-coordinate witnesses.

Falsifiable prediction: Every accepted EnergyRefinedPrivateCoordinate$^{\uparrow}$ consumer must display private-coordinate witnesses, an incidence-pattern energy row, and a coefficient ledger before using the energy-refined polynomial formula.

Independence witness: private-coordinate-trigger, finset, polynomial, nat, bool: EnergyRefinedPrivateCoordinate is not bijective to these siblings because it carries the private-coordinate support ledger together with energy weights, trigger-size grading, and coefficient replay.

Definition 6786.1 (Energy-refined private-coordinate carrier)**.** *An* EnergyRefinedPrivateCoordinate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (\Omega, A, U, V, \pi, X, W, Z, Y, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row.*
3. U, V, π *are the cover, private-witness, and membership/exclusion rows inherited from* PrivateCoordinateTrigger$^{\uparrow}$.
4. X *is the incidence-pattern row, listing for each coordinate the nonempty subset of displayed supports that contain it.*
5. W *is the energy ledger, recording the finite sum of incidence-pattern sizes.*
6. Z *is the trigger-size grading row.*
7. Y *is the polynomial coefficient ledger for the displayed variables* z *and* y.
8. K *is the energy-refined formula row, readable only after* U, V, π, X, W, Z, Y *have been displayed.*

9. *H stores componentwise* hsame *transport for coordinate, support, witness, energy, grading, coefficient, and formula rows.*
10. *C stores* Cont *replay from an enumerator query through the displayed rows.*
11. *P stores* Pkg *provenance over* $\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}$, $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, X, W, Z, Y, K, H, C, P, N$. It has no coordinate for an unweighted minimal-cover criterion, a bare polynomial expression, a natural-number table, a Boolean classifier, or a support family without the private-coordinate ledger.

Theorem 6786.2 (Energy-refined private-coordinate NameCert obligations)**.** *For every accepted* $\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}$ *carrier*

$$E = (\Omega, A, U, V, \pi, X, W, Z, Y, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}}$ *surface consists exactly of finite coordinate admission, support-family admission, private-coordinate witness replay, incidence-pattern exposure, energy-ledger exactness, trigger-size grading, polynomial coefficient replay, energy-refined formula exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier. The rows Ω, A, U, V, π display the private-coordinate minimal-cover surface inherited from chapter 5812.

The energy rows are X, W, Z, Y. The incidence-pattern row X records which displayed supports contain each coordinate, W sums the displayed pattern sizes, Z records trigger size, and Y stores the polynomial coefficient ledger.

The formula row K is downstream of both the private-coordinate witness rows and the energy rows. The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no omitted coordinate supplies a bare polynomial, natural-number table, Boolean classifier, or unweighted cover criterion, the obligation surface is exactly the displayed finite ledger. □

Theorem 6786.3 (Energy-refined polynomial handoff)**.** *Every accepted* $\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}$ *packet exposes the energy-refined formula of Definition 9462.1675 only after the private-coordinate cover row, incidence-pattern row, energy ledger, trigger-size grading, and polynomial coefficient ledger have been read. The packet may hand the resulting coefficient surface to* $\mathsf{Poly}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{Bool}^{\uparrow}$ *consumers, but it does not identify the carrier with any one of those siblings.*

Proof. Start with the private-coordinate rows Ω, A, U, V, π. These rows ensure that the support family is a displayed minimal-cover carrier rather than a bare finite-set family.

Next expose the energy and coefficient rows X, W, Z, Y. They record incidence patterns, the finite energy sum, trigger size, and polynomial coefficients before the formula row K is visible.

Finally read K as the energy-refined formula row. The provenance row names the polynomial, natural-number, and Boolean siblings only as consumers of the displayed ledger; it does not replace the private-coordinate and energy rows by one of those sibling classifiers. The structural rows therefore preserve the energy-refined carrier surface for later TasteGate nontriviality and FieldFaithful projection checks. □

Closure status for $\mathsf{NameCert}_{\mathsf{EnergyRefinedPrivateCoordinate}^{\uparrow}}$.

Constructive story: An EnergyRefinedPrivateCoordinate$^{\uparrow}$ packet is generated from finite trigger-coordinate rows, support-family rows, private-coordinate witnesses, incidence-pattern rows, energy ledgers, trigger-size grading, polynomial coefficient ledgers, formula replay, transport, continuation, provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{EnergyRefinedPrivateCoordinate}\uparrow}$, seed)

Scope closed: Definition 6786.1, Definition 6786.2, and Definition 6786.3 seed the finite energy-refined enumeration carrier for private-coordinate triggers.

Formal verification: FormalStatus(NameCert$_{\mathsf{EnergyRefinedPrivateCoordinate}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, TasteGate nontriviality, FieldFaithful projection, polynomial completeness, Boolean classifier completeness, or enumeration beyond the displayed energy-refined ledger.

Upgrade path: Obligation closure requires checked carrier admission, private-coordinate witness replay, incidence-pattern exposure, energy-ledger exactness, polynomial coefficient replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.EnergyRefinedPrivateCoordinateUp.

Chapter 6787

A Concrete Naming Certificate for $\mathsf{BoundedOscillation}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{BoundedOscillation}^{\uparrow}$ packet records a finite dyadic oscillation ledger over displayed $\mathsf{Real}^{\uparrow}$ windows. It sits between the real-name boundary of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the finite Darboux-sum surface of chapter 1880, and the bounded-variation function route of chapter 3654. The packet names only the finite ledger by which Darboux-facing consumers receive an oscillation bound; it does not claim host function equality, selected suprema, measure completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6787.1 (Bounded oscillation carrier)**.** *A* $\mathsf{BoundedOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, W, R, D, L, B, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *window row naming the displayed interval or cell on which the oscillation ledger is read.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window for endpoint and sample reads inside I.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the observations in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested oscillation radius, slack, and comparison budget.*
5. *L is the finite dyadic oscillation ledger, pairing the sampled upper and lower readbacks with the tolerance row D.*
6. *B is the displayed boundedness row asserting that the ledger oscillation is below the carried* $\mathsf{Real}^{\uparrow}$ *bound.*
7. *S is the* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row through which lower and upper sum consumers read the same bounded ledger.*
8. *H records componentwise* hsame *transport for I, W, R, D, L, B, S and for the continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Darboux-facing oscillation request through the window, readback, tolerance, ledger, bound, and sum-handoff rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedOscillation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, selected suprema, measure completion, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedOscillationUp.TasteGate.BoundedOscillationTasteGate single carrier alignment`

Theorem 6787.2 (Bounded oscillation NameCert obligations)**.** *For every accepted* $\mathsf{BoundedOscillation}^{\uparrow}$ *carrier* $O = (I, W, R, D, L, B, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real window, finite stream observations, regular rational readback, dyadic tolerance row, finite oscillation ledger, boundedness row, Darboux-sum handoff, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BoundedOscillationUp.TasteGate.BoundedOscillationNamecertObligations`

Proof. Project the carrier of Definition 6787.1. The analytic rows are I, W, R, D, L, B, S, and the structural rows are H, C, P, N.

The window and observation obligations are finite. The row I displays the Real window, W opens only the finite inspected observations, and R supplies the corresponding regular rational readback.

The tolerance and bound obligations are also finite. The dyadic row D fixes the comparison budget, the ledger L records the upper and lower oscillation readbacks against that budget, and B records the displayed bound before any Darboux consumer can use the packet.

Finally the Darboux handoff S exposes the same bounded ledger to lower and upper sum consumers, while H, C, P, N transport, replay, source, and name the packet. Since no other coordinate exists, the listed rows exhaust the local NameCert surface and exclude host equality, selected suprema, measure completion, choice, quotient equality, and ambient completeness. □

Theorem 6787.3 (Bounded oscillation Darboux handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedOscillation}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ L,\ B \longrightarrow \mathsf{DarbouxSum}^{\uparrow}.$$

The consumer therefore receives a displayed Real window, finite observations, regular rational readback, dyadic tolerance budget, finite oscillation ledger, and boundedness row before the Darboux handoff is visible. It receives no host function equality, selected supremum, measure completion, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedOscillationUp.TasteGate.BoundedOscillationDarbouxHandoff`

Proof. Start from an admitted consumer read. By Definition 6787.1, the Darboux row S is downstream of the Real window I, finite observations W, readback R, dyadic tolerance row D, oscillation ledger L, and bound row B.

Apply Definition 6787.2. It requires the same rows as the local NameCert surface, so a Darboux or bounded-variation consumer cannot read the handoff without exposing the finite oscillation ledger and its bound.

The structural rows H, C, P, N preserve that route componentwise. They do not add analytic coordinates, and the carrier has no row for host function equality, selected suprema, measure completion, countable choice, quotient stream equality, or ambient completeness.

Thus the packet supplies exactly the finite Darboux-facing oscillation ledger stated in the theorem, and no excluded principle can be projected from it. □

Theorem 6787.4 (BoundedVariation oscillation partition route)**.** *Every* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *consumer that reads an oscillation partition through* $\mathsf{BoundedOscillation}^{\uparrow}$ *must first expose the same* $\mathsf{Real}^{\uparrow}$ *window, finite* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, finite oscillation ledger, boundedness row, and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff carried by Definition 6787.1. The partition route is therefore a Darboux-facing finite ledger over displayed rows; it exports no host function equality, selected suprema, variation measure, countable choice, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedOscillationUp.TasteGate.BoundedVariationOscillationPartitionRoute`

Proof. Project the carrier of Definition 6787.1. The partition-facing data are already ordered as the Real window I, finite observations W, regular readback R, dyadic tolerance row D, oscillation ledger L, boundedness row B, and Darboux handoff S.

Apply Definition 6787.2. It makes those rows the complete local NameCert surface before any bounded-variation consumer can inspect the packet.

Now apply Definition 6787.3. The Darboux row S is downstream of the same window, observation, readback, tolerance, ledger, and bound rows, so an oscillation partition read by $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ cannot bypass the finite Darboux route.

The carrier has no coordinate for host function equality, selected suprema, variation measure, countable choice, quotient stream equality, or ambient completeness. Those principles therefore cannot be projected from the bounded-variation oscillation partition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedOscillation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6787.1, Definition 6787.2, and Definition 6787.3 seed the finite dyadic oscillation ledger over Real windows for DarbouxSum and bounded-variation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected suprema, measure completion, countable choice, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window readback, dyadic tolerance exactness, oscillation-ledger confinement, boundedness-row nonescape, Darboux handoff, and TasteGate field faithfulness for BEDC.Derived.BoundedOscillationUp.

Constructive story: A $\mathsf{BoundedOscillation}^{\uparrow}$ packet is generated from a RealUp window row, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance budget, finite oscillation ledger, displayed boundedness row, DarbouxSum handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6788

A Concrete Naming Certificate for $\mathsf{MetricGraph}^{\uparrow}$

Following Diestel and Burago–Burago–Ivanov, the $\mathsf{MetricGraph}^{\uparrow}$ packet names the path metric carried by a finite weighted graph. It reads graph adjacency from chapter 118, finite path ledgers from chapter 236, center-radius requests from chapter 1450, and the generated-open handoff from chapter 1655. The packet is finite: it records visible vertex rows, edge-length rows, path ledgers, metric-ball rows, topology rows, transport, replay, provenance, and a local naming row, without importing an ambient continuum graph, a selected geodesic, quotient equality of vertices, or open-set data outside the displayed ball basis.

Definition 6788.1 (Metric graph carrier)**.** *A* $\mathsf{MetricGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\text{graph}} = (G, E, W, S, L, D, B, T, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *carrier row, including finite vertex rows, edge-incidence rows, endpoint readback, and componentwise edge classification.*
2. *E is the finite edge inventory over G, naming the directed or undirected edge rows that may be used by a path ledger.*
3. *W is the edge-length ledger, attaching a nonnegative* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *length row to every displayed edge of E.*
4. *S is the* $\mathsf{ShortestPath}^{\uparrow}$ *source row, carrying finite path ledgers, endpoint rows, relaxation or comparison rows, and provenance for path-weight reads.*
5. *L is the path-length ledger. For each displayed path row it records the finite edge list, the endpoint row pair, the sum of the edge-length rows from W, and the comparison row used to read shortest-path distance.*
6. *D is the metric-distance row $d_G(x, y)$ obtained by reading the shortest displayed path length between the two endpoint rows when the packet admits such a path.*
7. *B is the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger whose membership row is read as $d_G(c, x) <_{\mathsf{R}} r$.*
8. *T is the* $\mathsf{MetricTopology}^{\uparrow}$ *generated-open row obtained from the finite family of metric-ball rows in B.*

9. *H records componentwise* hsame *transport for vertices, edges, edge-length rows, path ledgers, distance rows, ball rows, topology rows, replay rows, provenance, and the local certificate.*
10. *C records* Cont *replay from an endpoint request through $G, E, W, S, L, D, B, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricGraph}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{ShortestPath}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricGraph}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for a hidden all-pairs distance table, a selected geodesic outside the path ledger, quotient equality of vertices, an ambient graph completion, or topology data outside the generated metric-ball surface.

Lean4 ✓ `BEDC.Derived.MetricGraphUp.MetricGraphTasteGate single carrier alignment`

Theorem 6788.2 (Metric graph NameCert obligations)**.** *For every accepted* $\mathsf{MetricGraph}^{\uparrow}$ *carrier* $M_{\text{graph}} = (G, E, W, S, L, D, B, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricGraph}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the graph row G, finite edge inventory E, edge-length ledger W, shortest-path source row S, path-length ledger L, distance row D, metric-ball row B, topology row T, transport row H, replay row C, provenance row P, and local naming row N;*
2. *edge-length stability, requiring every edge read from E to carry the displayed nonnegative* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *length row of W through* hsame *transport;*
3. *path-ledger exactness, requiring each finite path length in L to be computed from the edge rows admitted by G, E, W and the path source row S;*
4. *shortest-path distance readback, requiring D to be read only through the finite comparison rows of S, L;*
5. *metric-ball handoff, requiring every membership read in B to use the displayed distance row D and radius ledger before a topology consumer can see it;*
6. *metric-topology handoff, requiring every generated-open row in T to arise from the carried metric-ball rows of B;*
7. *ledger non-escape, requiring every public read to stay inside $G, E, W, S, L, D, B, T, H, C, P, N$.*

No obligation is discharged by an ambient graph metric, a selected geodesic, quotient equality of vertices, an infinite graph completion, a host powerset topology, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6788.1. The graph row G, finite edge inventory E, edge-length ledger W, shortest-path source row S, path-length ledger L, distance row D, ball row B, topology row T, and structural rows H, C, P, N are the only exported coordinates.

Read the graph and path ledgers first. The graph chapter supplies endpoint and edge-row readback, while the shortest-path chapter supplies the finite path ledger and the comparison surface for path weights. Thus every distance read in D is downstream of visible edge incidence, visible edge lengths, and a finite path ledger.

Next read the metric-ball and topology rows. The ball membership row in B consumes the distance comparison $d_G(c, x) <_{\mathsf{R}} r$, and the topology row T consumes the carried ball rows as generated-open indices. These handoffs preserve the same finite packet through H and C.

Since the packet has no coordinate for a hidden distance table, selected geodesic, quotient vertex equality, graph completion, host powerset topology, or ambient real completeness, none of those objects can discharge a local NameCert obligation. □

Theorem 6788.3 (Metric graph shortest-path metric route)**.** *Every public metric read from an accepted* $\mathsf{MetricGraph}^{\uparrow}$ *packet factors through the finite shortest-path route*

$$\mathsf{Graph}^{\uparrow}, \mathsf{ShortestPath}^{\uparrow}, \mathsf{MetricBall}^{\uparrow}, \mathsf{MetricTopology}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

In particular, a consumer that asks for the metric ball around a vertex must first read the graph endpoint rows, the edge-length rows, the finite path ledger, and the shortest-path comparison row before it can use the center-radius membership row or the generated metric topology row.

Proof. Start with a public distance request between two vertex rows of G. By Definition 6788.1, the request can reach D only through the finite edge inventory E, the edge-length ledger W, the shortest-path source row S, and the path-length ledger L.

Apply the visible path-ledger and relaxation surfaces of chapter 236. They constrain the path read to displayed endpoint rows, finite edge-incidence rows, nonnegative edge weights, finite comparison rows, and provenance over the same graph packet.

The resulting distance row is then consumed by chapter 1450. A ball membership row reads $d_G(c, x) <_{\mathsf{R}} r$ through the displayed center, radius, positive-radius ledger, and comparison row.

Finally the metric-topology consumer of chapter 1655 receives only the finite family of metric-ball rows. The route therefore exposes graph, shortest-path, ball, and topology data in order, and cannot export a selected geodesic, an all-pairs host metric, quotient equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricGraph}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricGraph}^{\uparrow}$ packet is generated from Graph rows, finite edge-length rows, ShortestPath path ledgers, MetricBall center-radius rows, MetricTopology generated-open rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricGraph}^{\uparrow}}$, seed)

Scope closed: Definition 6788.1, Definition 6788.2, and Definition 6788.3 seed the finite route from graph path ledgers to metric balls and generated metric topology.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricGraph}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected geodesics, all-pairs host metrics, quotient equality of vertices, graph completions, host powerset topology, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, edge-length stability, path-ledger exactness, shortest-path distance readback, ball handoff, topology handoff, transport, replay, provenance, refusal, and local naming for BEDC.Derived.MetricGraphUp.

Chapter 6789

A Concrete Naming Certificate for $\mathsf{KleeneKreiselSpace}^{\uparrow}$

Following Kleene and Kreisel and Troelstra-style constructive higher-type continuity, the $\mathsf{KleeneKreiselSpace}^{\uparrow}$ packet records the represented-space boundary for continuous functionals over finite observations of choice-sequence names. It sits between the neighbourhood-table interface of chapter 2336, the higher-type read surface of chapter 6190, the stream-window discipline of chapter 126, the represented-space carrier route, the Polish-space completion context of chapter 1490, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite represented-space gate for Kleene–Kreisel style spaces without adding quotient function spaces, countable choice, selected total elements, or ambient completion objects.

Definition 6789.1 (KleeneKreiselSpace carrier and classifier)**.** *A* $\mathsf{KleeneKreiselSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Nb, F, R, Pc, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, listing the finite name prefixes through which a point or functional request is inspected.*
2. *Nb is the* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *table row, recording finite input-prefix requests and the output or acceptance rows released by those requests.*
3. *F is the* $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ *read row, naming the higher-type finite-window functional surface that consumes S and Nb.*
4. *R is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *code row, displaying the admissible representation code, its finite name requests, and the represented point row admitted by the code.*
5. *Pc is the* $\mathsf{PolishSpace}^{\uparrow}$ *completion-context row, recording the metric completion envelope used only as a context for the represented-space code.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available to numerical consumers only after the stream, neighbourhood, functional, represented-space, and Polish-context rows have been displayed.*
7. *H records componentwise* hsame *transport for $S, Nb, F, R, Pc, E, C, P, N$.*
8. *C records displayed* Cont *replay from each finite name request through the neighbourhood table, represented-space code, and functional read before any seal row is consumed.*

9. *P is the* Pkg *provenance over* $\mathsf{KleeneKreiselSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{NeighborhoodFunction}^{\uparrow}$, KleeneKreiselContinuo RepresentedSpace$^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KleeneKreiselSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these rows, their componentwise hsame *transports, their displayed* Cont *replays, their package provenance, and their local naming rows. It has no coordinate for host equality of total functionals, quotient equality of represented points, countable choice, a selected infinite name, or a completed space of all higher-type maps.*

Theorem 6789.2 (KleeneKreiselSpace NameCert obligations)**.** *For every accepted* $\mathsf{KleeneKreiselSpace}^{\uparrow}$ *carrier*

$$K = (S, Nb, F, R, Pc, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KleeneKreiselSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName observation exposure, neighbourhood-table exactness, continuous-functional handoff, represented-space code admission, Polish completion-context visibility, Real-seal boundary discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $S, Nb, F, R, Pc, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 6789.1. The displayed tuple exposes the stream observation row, neighbourhood table, continuous-functional read, represented-space code, Polish completion context, Real seal, transport row, replay row, provenance row, and local naming row.

The higher-type obligations are read from S, Nb, F. The stream row fixes the inspected finite prefixes, the neighbourhood table records which finite requests release public rows, and the continuous-functional row names the higher-type finite-window consumer that may use those requests.

The represented-space and completion-facing obligations are read from R, Pc, E. The represented-space row supplies the admissible code and represented point row, the Polish row supplies only a completion context, and the Real row is reachable only after the earlier rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for quotient function spaces, countable choice, selected total names, host equality of functionals, or ambient completion objects, the listed rows exhaust the NameCert obligations. □

Theorem 6789.3 (KleeneKreiselSpace neighbourhood stability)**.** *For accepted* $\mathsf{KleeneKreiselSpace}^{\uparrow}$ *carriers K and K' whose StreamName rows, neighbourhood tables, continuous-functional rows, represented-space codes, Polish-context rows, Real-seal rows, replay rows, provenance rows, and local naming rows are componentwise* hsame*, the classifier gives the same neighbourhood release decisions. In particular, every public output or represented-point read of K factors through the same* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *table entry and finite* $\mathsf{StreamName}^{\uparrow}$ *window as the corresponding read of K'.*

Proof. Open both accepted carriers using Definition 6789.1. The classifier is defined only from the displayed rows $S, Nb, F, R, Pc, E, H, C, P, N$ and their counterparts.

Componentwise hsame in H identifies the StreamName windows, the neighbourhood-table entries, the functional handoff, the represented-space code, the Polish context, the Real seal, and the structural rows. Thus a classifier read cannot distinguish K from K' by any hidden name or external equality proof.

Displayed Cont replay in C preserves the order of release: finite StreamName request, neighbourhood-table entry, continuous-functional read, represented-space code, and then any completion-facing

seal. Hence the same neighbourhood decision and the same finite window are used on both carriers. □

Theorem 6789.4 (KleeneKreiselSpace Real-completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer read of an accepted* $\mathsf{KleeneKreiselSpace}^{\uparrow}$ *carrier factors through the finite route*

$$S, Nb, F, R, Pc \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer may read finite StreamName observations, neighbourhood-table entries, continuous-functional handoff rows, represented-space codes, Polish completion-context rows, the terminal Real seal, transport, replay, provenance, and local naming, but it cannot read quotient function spaces, countable choice, selected total names, host functional equality, quotient represented-space equality, or ambient Real completeness from the $\mathsf{KleeneKreiselSpace}^{\uparrow}$ *packet.*

Proof. Read the accepted carrier of Definition 6789.1. The only rows preceding a Real or completion-facing seal are S, Nb, F, R, Pc, and the seal row E is displayed after those rows.

Apply Definition 6789.2. It forces the consumer to inspect StreamName observations, neighbourhood-table entries, continuous-functional handoff, represented-space codes, and the Polish completion context before the Real seal is public.

Transport and replay do not add a new completion object. The rows H and C preserve the displayed finite route, while P and N only source and name it. Therefore every Real or completion-facing consumer factors through the stated boundary, and the refused objects have no carrier coordinate from which to escape. □

Lean4 (statement) `BEDC.Derived.KleeneKreiselSpaceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneKreiselSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{KleeneKreiselSpace}^{\uparrow}$ packet is generated from finite StreamName observations, NeighborhoodFunction table rows, KleeneKreiselContinuousFunctional handoff rows, RepresentedSpace codes, PolishSpace completion-context rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KleeneKreiselSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6789.1, Definition 6789.2, Definition 6789.3, and Definition 6789.4 seed the finite represented-space boundary for Kleene–Kreisel style spaces on the L10 Real-completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KleeneKreiselSpace}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient function spaces, countable choice, selected total elements, quotient represented-space equality, host functional equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked Lean obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood-table exactness, continuous-functional handoff, represented-space code admission, Polish completion-context visibility, Real-seal nonescape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.KleeneKreiselSpaceUp.

Chapter 6790

A Concrete Naming Certificate for $\mathsf{WallisProduct}^{\uparrow}$

From Wallis' finite product bounds, the $\mathsf{WallisProduct}^{\uparrow}$ packet records the constructive route by which rational numerator and denominator ledgers, finite product rows, and terminal real-name comparisons are displayed before any Real or Completion consumer reads the classical Wallis product. It sits beside the rational source of chapter 37, the product constructor of chapter 34, the finite product boundary of chapter 464, and the terminal real-name seal of chapter 38. The packet names the Wallis product as finite BEDC data; it does not import infinite products, analytic limits, host real equality, quotient streams, or choice.

Definition 6790.1 (WallisProduct carrier)**.** *A* $\mathsf{WallisProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, A, B, L, R, S, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *arithmetic source row, exposing only rational numerators, rational denominators, positivity checks for the displayed denominators, and the rational comparison histories needed by the finite bounds.*
2. *A is the numerator ledger, recording the finite row of factors $2k \cdot 2k$ inspected by the current Wallis stage.*
3. *B is the denominator ledger, recording the finite row of factors $(2k-1)(2k+1)$ paired with A and licensed by Q.*
4. *L is the left rational bound row and R is the right rational bound row, each computed from the same finite numerator and denominator ledgers.*
5. *S is the* $\mathsf{Prod}^{\uparrow}$ *finite-product row that binds A, B, L, R into one displayed product-stage packet.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison row reached only after the rational ledgers and the finite-product row have been displayed.*
7. *H records componentwise* hsame *transport for the rational source, numerator ledger, denominator ledger, left and right bounds, product row, terminal Real comparison, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Wallis-stage request through Q, A, B, L, R, S, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{WallisProduct}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WallisProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows and their displayed route. It has no coordinate for an infinite product value, analytic convergence proof, host real equality, quotient real representative, selected limit, or countable-choice schedule.

Theorem 6790.2 (WallisProduct NameCert obligations)**.** *For every accepted* $\mathsf{WallisProduct}^{\uparrow}$ *carrier* $W = (Q, A, B, L, R, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WallisProduct}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the rational source row, numerator ledger, denominator ledger, left and right bound rows, finite-product row, terminal Real comparison row, transport row, replay row, provenance row, and local naming row;*
2. *denominator visibility, requiring every denominator factor recorded in B to be read through the displayed* $\mathsf{Rat}^{\uparrow}$ *positivity and nonzero-denominator histories in Q;*
3. *product-stage exactness, requiring S to consume precisely the numerator and denominator ledgers A, B and to export only the left and right rational bound rows L, R for that stage;*
4. *componentwise transport, requiring every comparison between accepted Wallis packets to move Q, A, B, L, R, S, E through displayed* hsame *rows and not through host rational or host real equality;*
5. *displayed replay, requiring each Real-facing request to replay the finite route through C before the terminal* $\mathsf{Real}^{\uparrow}$ *comparison row E is visible;*
6. *provenance and local naming, requiring P, N to source and name only the displayed rows and to export no infinite-product, analytic-limit, quotient-stream, selected-limit, or choice coordinate.*

No WallisProduct obligation is stored outside these finite rows.

Proof. Project the accepted carrier. Its arithmetic rows are exactly the rational source Q, the numerator ledger A, the denominator ledger B, the bound rows L, R, the product row S, and the terminal Real comparison row E.

Denominator visibility is componentwise. Each denominator factor in B is accepted only through the rational row Q, so the carrier never asks a host division operation or an ambient positivity oracle to certify the product stage.

The product-stage obligation is then read from S. The row S binds the same numerator and denominator ledgers to the left and right rational bounds, and the terminal row E is downstream of that finite product packet.

Finally H, C, P, N transport, replay, source, and name the same route. Since none of the displayed rows stores an infinite product value, analytic limit, host real equality, quotient stream, selected limit, or choice schedule, the listed obligations exhaust the local NameCert surface. □

Theorem 6790.3 (WallisProduct finite-bound route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{WallisProduct}^{\uparrow}$ *packet reads the Wallis product only through the finite route*

$$\mathsf{Rat}^{\uparrow} \longmapsto (A, B) \longmapsto \mathsf{Prod}^{\uparrow} \longmapsto (L, R) \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives rational numerator and denominator ledgers, denominator-validity histories, a finite product-stage row, left and right rational bounds, componentwise hsame *transport,*

displayed Cont *replay,* Pkg *provenance, and a local* NameCert *row. It receives no infinite product object, analytic convergence limit, host real equality, quotient stream, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier of Definition 6790.1. The only arithmetic source is Q, and the only product-stage data are the displayed ledgers A, B together with the finite product row S and the bound rows L, R.

The terminal comparison row E is available only after those rows have been read. Thus a Real or Completion consumer must pass through the rational ledgers, product-stage packet, and rational bounds before it can inspect the terminal $\mathsf{Real}^{\uparrow}$ comparison.

The structural rows preserve the same finite route. The hsame row transports components, the Cont row replays the consumer request, and the Pkg and NameCert rows source and name the packet. Since the carrier has no coordinate for an infinite product, analytic limit, host equality, quotient stream, selected limit, choice schedule, or Real completeness, none of those principles can enter the consumer route. □

Falsifiable prediction: Every $\mathsf{WallisProduct}^{\uparrow}$ consumer read factors through one RatUp arithmetic source row, one finite numerator ledger, one finite denominator ledger, left and right rational bound rows, one ProdUp finite-product row, one terminal RealUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: rat, prod, cauchyproduct, real: $\mathsf{WallisProduct}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Wallis-stage numerator and denominator ledgers, rational denominator-validity histories, finite product-stage bounds, terminal Real comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WallisProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{WallisProduct}^{\uparrow}$ packet is generated from a RatUp arithmetic source row, finite numerator and denominator ledgers, left and right rational bound rows, a ProdUp finite-product row, a terminal RealUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WallisProduct}^{\uparrow}}$, seed)

Scope closed: Definition 6790.1, Definition 6790.2, and Definition 6790.3 seed the finite-product Wallis route used by Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WallisProduct}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite products, analytic convergence, host real equality, quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, denominator visibility, product-stage exactness, componentwise transport, displayed replay, provenance exhaustion, and local naming for BEDC.Derived.WallisProductUp.

Chapter 6791

A Concrete Naming Certificate for $\mathsf{BishopLocatedMinimum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedMinimum}^{\uparrow}$ packet records minimum-location data as a finite BEDC NameCert route through compact finite nets, displayed value windows, and located lower comparisons. It is the lower-comparison dual of chapter 6770, and it sits over the compact-metric source of chapter 129, the uniformly continuous map row of chapter 130, the finite-net handoff of chapter 6769, the stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes the finite comparison route needed by minimization consumers without selecting an argmin, importing open-cover compactness, identifying quotient real names, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6791.1 (Bishop located minimum carrier)**.** *A* $\mathsf{BishopLocatedMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, N, W, R, E, M, H, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the compact metric inspection surface and the finite center coverage consumed by minimum-location reads.*
2. *F is the uniformly continuous* $\mathsf{ContinuousMap}^{\uparrow}$ *row, exposing only displayed value requests over the compact source row K.*
3. *N is the finite-net cover row, recording the finite list of centers, radii, and coverage witnesses through which every minimum-facing request enters.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite value windows attached to the centers of N.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *value readback row for the rational windows exposed by W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, admitting the displayed value readbacks as Real-facing names before comparisons are consumed.*
7. *M is the located minimum comparison row, comparing each displayed candidate value with the finite net family through located lower comparison windows.*
8. *H records componentwise* hsame *transport for the compact source, uniformly continuous map row, finite-net cover, stream schedule, regular rational readback, Real seal, located lower comparison row, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from a minimum-facing request through $K, F, N, W, R, E, M, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedMinimum}^{\uparrow}$, $\mathsf{BishopLocatedMaximum}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantorFiniteNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed compact source, uniformly continuous map row, finite-net cover, stream windows, regular rational readbacks, Real seals, located lower comparisons, transport, replay, provenance, and local naming. It has no coordinate for a selected argmin, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6791.2 (Bishop located minimum NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedMinimum}^{\uparrow}$ *carrier $B = (K, F, N, W, R, E, M, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedMinimum}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed compact source, uniformly continuous map row, finite-net cover, value windows, regular rational readbacks, Real seal, located lower comparison row, transport, replay, provenance, and local naming;*
2. *finite-net exposure, requiring every minimum-facing source request to pass through the centers and coverage witnesses named by N;*
3. *value readback through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *requiring W, R, E to be visible before any lower comparison row is consumed;*
4. *located lower comparison, requiring M to compare only displayed Real-sealed value windows over the finite net;*
5. *ledger refusal for selected argmin data, open-cover compactness, quotient real equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6791.1. The rows K, F, N expose the compact source, map row, and finite-net coverage surface that every later value read must use.

Read the value rows next. The stream schedule W supplies finite windows, R reads those windows as regular rational data, and E seals the displayed readbacks as $\mathsf{Real}^{\uparrow}$ names before a lower comparison can be used.

The comparison row M has access only to the Real-sealed finite-net values already displayed by N, W, R, E. The structural rows H, C, P, L transport, replay, source, and name the same finite route, so the five listed obligations exhaust the local NameCert surface. □

Theorem 6791.3 (Bishop located minimum finite-net lower route)**.** *Every minimum-facing read exported by an accepted* $\mathsf{BishopLocatedMinimum}^{\uparrow}$ *carrier factors through the displayed finite route*

$$K, F, N \longmapsto W, R, E \longmapsto M \longmapsto H, C, P, L.$$

Thus an optimization consumer can inspect only the compact source, uniformly continuous map row, finite net, value windows, located lower comparison, and Real seal before reaching the local naming row. It cannot bypass the finite-net handoff to obtain a selected argmin, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source and map rows. The compact metric source K and uniformly continuous map row F become visible to a minimum-facing read only through the finite-net cover N.

The value route is then fixed. The stream schedule W records the finite observation windows attached to the net, the regular rational row R reads those windows, and the terminal seal E is reached before any minimization-facing comparison is available.

Use the located lower comparison row M only after the sealed value windows are displayed. Since M compares the finite family exposed by N, W, R, E, it supplies a located minimum comparison surface rather than a selected minimizing point.

Finally replay the same route through H and C, while P and L provide provenance and local naming. The carrier has no coordinate for the excluded principles, so every consumer reaches minimum-location data only through the finite lower handoff shown above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedMinimum}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6791.1, Definition 6791.2, and Definition 6791.3 seed the finite-net minimum-location route over compact-source, uniformly continuous map, value-window, located-lower-comparison, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected minimizers, full open-cover compactness, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BishopLocatedMinimum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, StreamName-RegSeqRat-Real value readback, located lower comparison, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedMinimumUp.

Constructive story: A $\mathsf{BishopLocatedMinimum}^{\uparrow}$ packet is generated from a CompactMetric source row, a uniformly continuous ContinuousMap row, a finite-net cover, StreamName value windows, RegSeqRat readbacks, a Real seal, a located lower comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6792

A Concrete Naming Certificate for $\mathsf{BishopChoiceSequence}^{\uparrow}$

Following Bishop and Bridges and Troelstra–van Dalen, the $\mathsf{BishopChoiceSequence}^{\uparrow}$ packet records a finite observation surface for choice sequences without importing a selected infinite choice function. It sits over the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the tail-equivalence boundary of chapter 1738. The packet is designed for Real and Completion consumers that need a finite observed window and a stable handoff, not an ambient equality of streams.

Definition 6792.1 (Bishop choice-sequence carrier)**.** *A* $\mathsf{BishopChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite schedule row listing the observation indices and tolerance requests that have actually been opened.*
2. *W is the observed* $\mathsf{StreamName}^{\uparrow}$ *window read from that schedule, with no coordinate for an unobserved tail.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite schedule and observed window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed window and tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been consumed.*
6. *H records componentwise* hsame *transport for the schedule, observed window, tolerance ledger, readback, seal, and local certificate rows.*
7. *C records displayed* Cont *replay for schedule opening, finite observation, dyadic tolerance checking, regular readback, and Real-name handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopChoiceSequence}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $S, W, D, R, E, H, C, P, N$. It has no coordinate for countable choice, quotient stream equality, a selected limit, trichotomy, host equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6792.2 (Bishop choice-sequence window stability)**.** *Let $X = (S, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopChoiceSequence}^{\uparrow}$ *carrier. If a later finite schedule extends the inspection while preserving the already displayed schedule row S, observed* $\mathsf{StreamName}^{\uparrow}$ *window W, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, then the original regular readback row R and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E remain the same rows under the displayed H, C, P, N transport.*

Proof. Project the carrier of Definition 6792.1. The only observed analytic rows before transport are S, W, D, R, E, so the classifier has no hidden stream tail from which to obtain a second readback.

The later schedule preserves S, W, D. Therefore the finite observation and dyadic tolerance data used to form R are unchanged inside the packet.

The transport row H compares the displayed rows componentwise, and the replay row C replays the same finite route. The provenance and local naming rows P, N name that same packet, so the terminal seal E travels with R rather than being recomputed from unobserved choice data. □

Theorem 6792.3 (Bishop choice-sequence Real-name handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *name exported by an accepted* $\mathsf{BishopChoiceSequence}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The handoff from finite choice-sequence observation to a Real-facing consumer is therefore mediated by the observed $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal seal is available.*

Proof. Start with the carrier definition. The terminal seal row E occurs after the schedule, observed window, tolerance ledger, and regular readback rows, and the classifier reads no separate Real constructor.

Use the window-stability theorem for later observations. Preserving the displayed finite window and tolerance ledger preserves R and E, so a Real-facing consumer cannot replace the finite route by a selected limit or by equality of host streams.

Thus the handoff is the displayed chain through S, W, D, R, E. The packet supplies a local name for the finite observation route and leaves ambient completeness, quotient stream equality, countable choice, trichotomy, and selected limits outside the certificate. □

Falsifiable prediction: Every accepted $\mathsf{BishopChoiceSequence}^{\uparrow}$ Real-facing read exposes one finite schedule row, one observed $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchy-tail-equivalence: $\mathsf{BishopChoiceSequence}^{\uparrow}$ is not bijective to any listed sibling because it binds the finite schedule, observed window, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local NameCert into one observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopChoiceSequence}^{\uparrow}$ packet is generated from a finite schedule row, an observed StreamName window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopChoiceSequence}^{\uparrow}}$, seed)

Scope closed: Definition 6792.1, Definition 6792.2, and Definition 6792.3 seed the finite observation route from choice-sequence schedules through StreamName, DyadicRatCore, RegSeqRat, and Real-name handoff rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopChoiceSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected limits, quotient stream equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking for BishopChoiceSequence$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, window stability, Real-name handoff, classifier transport, displayed replay, provenance, and local naming for BEDC.Derived.BishopChoiceSequenceUp.

Chapter 6793

A Concrete Naming Certificate for SquareRootContinuityModulus$^{\uparrow}$

Following Bishop and Bridges on constructive square-root continuity, the SquareRootContinuityModulus$^{\uparrow}$ packet records the finite uniform-modulus certificate that lets a bounded nonnegative interval consumer use Heron's square-root route without importing a host square-root function or a completed real continuity theorem. It sits downstream of the value-level square-root certificates in chapter 3818 and chapter 1756, and it binds that route to the continuity and compact-interval surfaces of chapter 59, chapter 369, and chapter 1591.

Definition 6793.1 (Square-root continuity-modulus carrier)**.** *A* SquareRootContinuityModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, F, U, R, H, S, T, C, P, N)$$

with the following displayed rows:

1. *I is the* ClosedBoundedInterval$^{\uparrow}$ *row naming the bounded nonnegative source interval and the rational lower-bound data used by the square-root route.*
2. *F is the* Continuous$^{\uparrow}$ *row for the square-root read on the displayed interval, with continuity evidence read only through finite regular rational windows.*
3. *U is the* UniformModulus$^{\uparrow}$ *row assigning each requested output tolerance a finite interval-window threshold.*
4. *R is the* RegularCauchySquareRoot$^{\uparrow}$ *row giving the regular rational square-root readback available at the chosen threshold.*
5. *H is the* HeronSquareRoot$^{\uparrow}$ *row giving the displayed Heron recurrence, reciprocal ledger, contraction window, and Real seal handoff used by the modulus witness.*
6. *S records componentwise* hsame *transport for the interval, continuity row, modulus row, regular square-root row, Heron route, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from a tolerance request through I, F, U, R, H, S, P, N.*
8. *C is the compatibility row saying that the threshold chosen by U is exactly the threshold at which R and H expose the same square-root window on the source interval.*
9. *P is the* Pkg *provenance over* SquareRootContinuityModulus$^{\uparrow}$, HeronSquareRoot$^{\uparrow}$, RegularCauchySquareRoot$^{\uparrow}$, Continuous$^{\uparrow}$, UniformModulus$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SquareRootContinuityModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, F, U, R, H, S, T, C, P, N$. It has no coordinate for a host square-root operation, a selected completed limit, quotient-real equality, global real completeness, or a pointwise continuity proof that bypasses the displayed uniform-modulus row.

Theorem 6793.2 (Square-root continuity-modulus NameCert obligations)**.** *For every accepted* $\mathsf{SquareRootContinuityModulus}^{\uparrow}$ *carrier*

$$M = (I, F, U, R, H, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SquareRootContinuityModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded nonnegative interval, displayed continuity row, uniform-modulus row, regular Cauchy square-root readback, Heron recurrence route, componentwise transport, continuation replay, modulus-route compatibility, package provenance, and local naming.*

Proof. Project the carrier rows. The interval row I supplies the bounded nonnegative source scope, while F and U record the continuity assertion and the uniform tolerance-to-window service on that scope.

The numerical square-root read is not independent data. The row R provides the regular rational square-root readback, and the row H provides the Heron recurrence route, reciprocal ledger, contraction window, and terminal Real seal handoff used to justify the same finite window.

The structural rows S, T, C, P, N preserve and replay exactly that packet. Since no further carrier coordinate is available, the listed rows exhaust the local NameCert surface, and no host square-root operation or global completeness theorem can replace the displayed modulus certificate. □

Theorem 6793.3 (Square-root continuity-modulus Heron route)**.** *Every accepted* $\mathsf{SquareRootContinuityModulus}^{\uparrow}$ *packet factors a uniform square-root tolerance request through the route*

$$I \longmapsto F \longmapsto U \longmapsto R \longmapsto H \longmapsto C.$$

Thus a consumer first sees the bounded nonnegative interval, the displayed continuity row, the uniform-modulus threshold, the regular Cauchy square-root readback, the Heron recurrence route, and the compatibility row tying the threshold to the same finite square-root window. The route exports no host square-root function, selected completed limit, quotient-real equality, or ambient real completeness principle.

Proof. Start with an accepted carrier $M = (I, F, U, R, H, S, T, C, P, N)$. A tolerance request is replayed through T, so the first numerical datum it can read is the interval scope I together with the continuity and modulus rows F, U.

The modulus row chooses a finite threshold. At that threshold, R exposes the regular Cauchy square-root readback and H exposes the Heron recurrence window with its reciprocal and contraction ledger.

The compatibility row C identifies these finite windows as the same square-root read on the displayed interval. Transport and provenance through S, P, N preserve the route but add no extra numerical coordinate.

Consequently every uniform square-root tolerance request factors through the stated Heron route, and the packet cannot be read as a host square-root function, a selected limit, quotient-real equality, or an ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{SquareRootContinuityModulus}^{\uparrow}$ read of a square-root tolerance request factors through a ClosedBoundedInterval source row, displayed Continuous row, UniformModulus threshold row, RegularCauchySquareRoot readback, HeronSquareRoot recurrence route, componentwise hsame transport, Cont replay, compatibility, Pkg provenance, and local NameCert row.

Independence witness: heron-square-root, regular-cauchy-square-root, continuous, uniform-modulus, closed-bounded-interval: $\mathsf{SquareRootContinuityModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a bounded nonnegative interval, a displayed continuity row, a uniform tolerance-to-window threshold, regular square-root readback, Heron recurrence data, compatibility, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SquareRootContinuityModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SquareRootContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6793.1, Definition 6793.2, and Definition 6793.3 seed the finite uniform-modulus certificate connecting Heron's square-root route to bounded nonnegative interval continuity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SquareRootContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host square-root function, selected completed limits, quotient-real equality, global real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval nonnegativity, continuity-row exactness, uniform-modulus threshold exactness, regular Cauchy readback, Heron route compatibility, transport, replay, provenance, and local naming for BEDC.Derived.SquareRootContinuityModulusUp.

Constructive story: A $\mathsf{SquareRootContinuityModulus}^{\uparrow}$ packet is generated from a bounded nonnegative ClosedBoundedInterval row, displayed Continuous and UniformModulus rows, RegularCauchySquareRoot readback, HeronSquareRoot recurrence route, componentwise hsame transport, Cont replay, compatibility, Pkg provenance, and a local NameCert row.

Chapter 6794

A Concrete Naming Certificate for $\mathsf{BishopIntervalMap}^{\uparrow}$

Following Bishop and Bridges on constructive functions over located intervals, and Scott's interval-domain reading of approximation, the $\mathsf{BishopIntervalMap}^{\uparrow}$ packet records the finite map surface used when interval-domain approximants are handed to Real-facing continuous-map consumers. It sits after the interval-domain carrier of chapter 1543, the continuous-map graph and modulus surface of chapter 130, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed interval-domain map certificate only; it does not define quotient functions, identify host extensional equality, select completed representatives, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6794.1 (BishopIntervalMap carrier)**.** *A* $\mathsf{BishopIntervalMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, F, W, Q, E, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{IntervalDomain}^{\uparrow}$ *approximation-domain row, exposing located cells, rational refinements, directed nesting, endpoint windows, regular readbacks, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal only in that order.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row whose source reads from the displayed interval-domain approximants and whose target is a Real-facing naming surface.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which finite endpoint and image-approximation requests are inspected.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same windows, carrying the regular rational observations consumed before the Real seal.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after D, F, W, Q and the structural rows are displayed.*
6. *H records componentwise* hsame *transport for the interval-domain, continuous-map, stream-window, regular-readback, Real-handoff, provenance, and local naming rows.*
7. *C records displayed* Cont *replay routes for interval-domain reads, continuous-map reads, stream requests, regular rational readbacks, and Real-facing handoffs.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalMap}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopIntervalMap}^\uparrow}$ *row.*
The carrier exports only this finite map packet. It has no coordinate for quotient functions, host function extensionality, selected Real representatives, open-cover compactness, or ambient completeness of $\mathsf{Real}^\uparrow$*.*

Theorem 6794.2 (BishopIntervalMap NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalMap}^\uparrow$ *carrier $M = (D, F, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalMap}^\uparrow}$ *surface consists exactly of carrier admission for $D, F, W, Q, E, H, C, P, N$; source-domain visibility through the* $\mathsf{IntervalDomain}^\uparrow$ *packet; graph and modulus visibility through the* $\mathsf{ContinuousMap}^\uparrow$ *row; finite endpoint and image-window replay through W and Q; Real-handoff non-escape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Carrier admission is projection from Definition 6794.1. The displayed inventory is exactly the interval-domain row, continuous-map row, stream windows, regular rational readbacks, Real handoff, transports, replay routes, provenance, and local naming row.

Read the source through chapter 1543. The interval-domain packet already orders located cells, rational refinements, directed nesting, stream windows, regular readbacks, and the Real seal, so the map certificate cannot consume a Real-facing approximation before the source approximant rows are visible.

Read the graph through chapter 130. The continuous-map row supplies only displayed graph and modulus data tied to the same finite windows, and H, C preserve those rows under classifier transport and replay.

Finally inspect the handoff row E. Since the packet has no coordinate for quotient functions, host extensional equality, selected representatives, open-cover compactness, or ambient completeness, the local certificate can expose only the finite map route named by $D, F, W, Q, E, H, C, P, N$. □

Theorem 6794.3 (BishopIntervalMap Real handoff)**.** *Every* $\mathsf{Real}^\uparrow$*-facing consumer of an accepted* $\mathsf{BishopIntervalMap}^\uparrow$ *packet first reads the finite route*

$$\mathsf{IntervalDomain}^\uparrow \longmapsto \mathsf{ContinuousMap}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow \longmapsto (H, C, P, N).$$

The handoff therefore supplies an interval-domain map surface for Real and Completion consumers, not a quotient function, not host extensional equality, not a selected completed value, and not ambient completeness of $\mathsf{Real}^\uparrow$*.*

Proof. Begin with the source approximation. The row D is the displayed interval-domain packet, so any consumer must first read its located cells, rational refinements, directed nesting, endpoint windows, regular readbacks, and downstream Real seal discipline.

Next read the map row. The row F is a continuous-map certificate attached to those source approximants, while W and Q provide the finite request windows and regular rational readbacks used to inspect both endpoint and image data.

The Real-facing row E is downstream of D, F, W, Q. The structural rows H, C, P, N transport, replay, source, and name the same route, so the consumer receives only the displayed finite packet and not a host-level function object.

Coordinate exhaustion then gives the exclusions. Quotient functions, host extensional equality, selected completed values, and ambient Real completeness would each require a row absent from Definition 6794.1. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalMap}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalMap}^{\uparrow}$ packet is generated from an IntervalDomain source row, a ContinuousMap graph-and-modulus row, StreamName windows, RegSeqRat readbacks, a RealUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6794.1, Definition 6794.2, and Definition 6794.3 seed the finite interval-domain map route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient functions, host extensional equality, selected completed representatives, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-domain visibility, continuous-map graph visibility, finite-window replay, Real-handoff non-escape, transport stability, provenance exhaustion, and local naming for BEDC.Derived.BishopIntervalMapUp.

Chapter 6795

A Concrete Naming Certificate for $\mathsf{SymmetricAlgebra}^{\uparrow}$

Following Bourbaki and Mac Lane, the $\mathsf{SymmetricAlgebra}^{\uparrow}$ packet records the free commutative algebra generated by a displayed module or vector-space carrier through finite tensor words modulo displayed permutation-stability rows. It is grounded in the scalar ring rows of chapter 43, the module carrier of chapter 46, the vector-space specialization of chapter 47, the finite word source of **??**, the alternating companion of chapter 5144, and the coefficient-list consumer of chapter 50. The packet is a BEDC NameCert carrier for finite symmetric words and polynomial handoff rows; it is not a host quotient algebra, selected monomial basis, infinite direct sum, or ambient commutative algebra object.

Definition 6795.1 (Symmetric algebra carrier)**.** *A* $\mathsf{SymmetricAlgebra}^{\uparrow}$ *carrier over a displayed* $\mathsf{Module}^{\uparrow}$ *row* V *and scalar* $\mathsf{Ring}^{\uparrow}$ *row* R *is a finite* BHist *packet*

$$S = (R, V, T, W, \Pi, M, H, C, P, N)$$

with the following rows:

1. R *is the scalar* $\mathsf{Ring}^{\uparrow}$ *source row, and* V *is the* $\mathsf{Module}^{\uparrow}$ *source row; when the scalar row is a field, the same endpoint source may be read through* $\mathsf{VecSpace}^{\uparrow}$*.*
2. T *is the consumed* $\mathsf{TensorAlgebra}^{\uparrow}$ *finite-word row over* R *and* V*.*
3. W *is the finite symmetric-word ledger. Each entry is a displayed tensor word together with the finite multiset of carried letters it represents.*
4. Π *is the finite permutation-stability ledger. For every listed adjacent swap or finite permutation of a listed word,* Π *records the classifier comparison that keeps the word in the same symmetric class.*
5. M *is the commutative product row. It multiplies two displayed symmetric words by tensor-word concatenation through* T *and then reads the result through* W *and* Π*, so the two product orders are classified by the same permutation-stability surface.*
6. H *records componentwise* hsame *transport for scalar rows, source letters, tensor words, symmetric-word rows, permutation-stability rows, commutative product rows, and local classifiers.*
7. C *records displayed* Cont *replay from source endpoints through tensor-word formation, symmetric-word admission, permutation stability, and commutative multiplication.*

8. *P is the* Pkg *provenance over* $\mathsf{SymmetricAlgebra}^{\uparrow}$, $\mathsf{TensorAlgebra}^{\uparrow}$, $\mathsf{ExteriorAlgebra}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SymmetricAlgebra}^{\uparrow}}$ *row.*

The carrier exports only finite symmetric tensor-word data and the displayed commutative product handoff. It does not add a quotient primitive, selected monomial basis, host polynomial syntax, infinite direct sum, or ambient algebra-library object.

Theorem 6795.2 (Symmetric algebra NameCert obligations)**.** *For every accepted* $\mathsf{SymmetricAlgebra}^{\uparrow}$ *carrier*

$$S = (R, V, T, W, \Pi, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SymmetricAlgebra}^{\uparrow}}$ *surface consists exactly of scalar admission through R, module or vector-space endpoint admission through V, finite tensor-word admission through T, symmetric-word admission through W, finite permutation-stability admission through Π, commutative product admission through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 6795.1. The scalar, source-letter, tensor-word, symmetric-word, permutation-stability, and product rows are displayed before a consumer receives a commutative algebra operation.

The tensor source is read through **??**. Thus every symmetric word first appears as a finite tensor word, and only then passes through the symmetric-word ledger W and the permutation-stability ledger Π.

The commutative product row M is also visible data. Multiplying two symmetric words means concatenating their tensor representatives and applying the listed permutation-stability rows, so commutativity is recorded by the packet rather than supplied by a host quotient algebra.

Transport by H, replay by C, provenance by P, and naming by N preserve exactly the displayed packet. Because the carrier has no quotient coordinate, selected monomial basis, host polynomial syntax, infinite direct sum, or ambient commutative algebra endpoint, the listed rows exhaust the local NameCert surface. □

Theorem 6795.3 (Symmetric algebra universal polynomial route)**.** *Every polynomial-style consumer accepted by a* $\mathsf{SymmetricAlgebra}^{\uparrow}$ *packet must read a finite symmetric word through*

$$R, V, T, W, \Pi, M, H, C, P, N.$$

The route first consumes the scalar $\mathsf{Ring}^{\uparrow}$ *row and the* $\mathsf{Module}^{\uparrow}$ *or* $\mathsf{VecSpace}^{\uparrow}$ *source row, then the* $\mathsf{TensorAlgebra}^{\uparrow}$ *word row, then the finite symmetric-word and permutation-stability ledgers, and finally the commutative product row. The consumer receives a finite polynomial handoff compatible with chapter 50; it does not receive quotient algebra equality, selected monomial coordinates, host polynomial syntax, arbitrary symmetric powers, exterior alternating rows, or ambient algebra completeness.*

Proof. Use Definition 6795.2. It places T, W, Π, M after the scalar and module or vector endpoints and before any polynomial-style read is exported.

For a degree-one word, the route is the source row V with transport and replay supplied by H and C. For a longer word, read the finite tensor word through T, then read all listed adjacent swaps and finite permutations through Π before the word is admitted by W.

Finally inspect the product row M. The two possible orders of multiplying displayed symmetric words differ only by a permutation row already present in Π, so the polynomial consumer

receives a commutative finite-word route rather than a quotient object. The exterior companion of chapter 5144 uses signed transposition and repeated-letter refusal; this packet instead keeps ordinary permutation stability and exports no alternating boundary. □

Falsifiable prediction: Every consumer of a $\mathsf{SymmetricAlgebra}^{\uparrow}$ packet factors through the displayed tensor-word row, symmetric-word ledger, finite permutation-stability ledger, commutative product row, transport, replay, provenance, and local naming rows; a request that supplies only host quotient algebra equality is rejected by this certificate.

Independence witness: ring, module, vecspace, tensor-algebra, exterior-algebra, polynomial: $\mathsf{SymmetricAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries scalar source rows, module or vector-space source rows, finite tensor words, symmetric-word ledgers, permutation-stability rows, commutative multiplication, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{SymmetricAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{SymmetricAlgebra}^{\uparrow}$ packet is generated from scalar RingUp rows, ModuleUp or VecSpaceUp endpoint rows, finite TensorAlgebraUp word rows, symmetric-word ledgers, finite permutation-stability rows, commutative product rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SymmetricAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6795.1, Definition 6795.2, and Definition 6795.3 seed the finite commutative tensor-word NameCert surface over ring, module, vector-space, tensor-algebra, exterior-algebra, and polynomial rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SymmetricAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host quotient symmetric algebras, selected monomial bases, arbitrary symmetric powers, exterior alternating rows, infinite direct sums, or completed commutative algebra structure.

Upgrade path: Obligation closure requires binding the displayed carrier, symmetric-word ledger, permutation-stability ledger, commutative product row, transport, replay, provenance, and local naming against their BEDC primitive scope.

Chapter 6796

A Concrete Naming Certificate for $\mathsf{BishopCompactRegularCover}^{\uparrow}$

Following Bishop and Bridges constructive compactness, the $\mathsf{BishopCompactRegularCover}^{\uparrow}$ packet records finite cover refinement as explicit NameCert data over a compact metric carrier. It sits beside the compact-metric route of chapter 129, the metric-space source of chapter 57, the rational radius discipline of chapter 37, the Lebesgue-number handoff of chapter 4015, and the $\mathsf{UniformModulus}^{\uparrow}$ consumer used by chapter 9237. The packet exposes a finite regular refinement route for compact-uniform-continuity and Lebesgue-number style arguments without claiming open-cover compactness, selected subcovers, quotient point equality, countable choice, or ambient completeness.

Definition 6796.1 (Bishop compact regular cover carrier)**.** *A* $\mathsf{BishopCompactRegularCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, M, Q, \mathcal{P}, C, L, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the compact finite-cover inspection surface.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare centers, radii, and refinement witnesses.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius row, naming the rational radius ledgers used by cover and refinement checks.*
4. *$\mathcal{P}$ is the finite* ProbeBundle *cover row, listing centers, rational radii, and membership witnesses for the displayed finite cover.*
5. *C is the regularity row, requiring every bundled ball to carry a smaller rational core still routed through $M, Q, \mathcal{P}$.*
6. *L is the refinement ledger row, recording the finite parent map from a refined* ProbeBundle *back to the displayed cover.*
7. *H records componentwise* hsame *transport for the compact source, metric source, rational radii, cover bundle, regularity row, refinement ledger, replay row, provenance row, and local naming row.*
8. *T records displayed* Cont *replay of finite cover refinement over the same bundle.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompactRegularCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCompactRegularCover}^{\uparrow}}$ *row.*
The carrier contains no open-cover compactness object, selected subcover functional, quotient point equality, countable choice, hidden Lebesgue-number selector, or ambient completion row.

Theorem 6796.2 (Bishop compact regular cover NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactRegularCover}^{\uparrow}$ *carrier* $R = (K, M, Q, \mathcal{P}, C, L, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactRegularCover}^{\uparrow}}$ *surface consists exactly of compact-source admission, metric-distance admission, positive rational radius admission, finite* $\mathsf{ProbeBundle}$ *cover admission, regular core exposure, refinement-ledger readback, componentwise transport, displayed replay, provenance, and local naming. The route is confined to*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}, \quad \mathsf{ProbeBundle}.$$

Proof. Project the carrier in Definition 6796.1. The only source rows visible to a cover-refinement consumer are $K, M, Q, \mathcal{P}$, so every cover read starts from compact metric data, metric comparisons, positive rational radii, and a finite bundle.

The regularity row C and refinement ledger L are attached to that same finite bundle. They can shrink a displayed radius and record a parent map for the refined bundle, but they cannot introduce an external subcover or an unlisted center.

The structural rows H, T, P, N transport, replay, source, and name the same finite route. Since none of the excluded compactness, choice, quotient, selector, or completion objects occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6796.3 (Bishop compact regular cover refinement induction)**.** *Every finite refinement read of an accepted* $\mathsf{BishopCompactRegularCover}^{\uparrow}$ *carrier is obtained by primitive induction over the displayed* $\mathsf{ProbeBundle}$ *spine. At the empty spine the refinement ledger is the empty ledger. At a successor probe, the read appends one regular core row from C, one positive* $\mathsf{Rat}^{\uparrow}$ *radius comparison from Q, and one parent entry in L, then replays the same route through H, T, P, N.*

Proof. Induct on the finite $\mathsf{ProbeBundle}$ row $\mathcal{P}$. The empty bundle has no cover entry and therefore supplies the empty refinement ledger.

For the successor step, read the next bundled center and rational radius through M and Q. The regularity row C supplies the smaller core for that entry, and L records the parent map back to the original cover row.

Transport and replay are componentwise. The rows H, T, P, N attach the successor entry to the same accepted carrier, so the induction never leaves the finite bundle or asks for a selected subcover. □

Theorem 6796.4 (Bishop compact regular cover nonescape)**.** *Every compact-uniform-continuity or Lebesgue-number consumer that reads a* $\mathsf{BishopCompactRegularCover}^{\uparrow}$ *packet receives only the finite route*

$$K, M, Q, \mathcal{P}, C, L, H, T, P, N.$$

In particular, a downstream $\mathsf{UniformModulus}^{\uparrow}$ *handoff may use the displayed regular refinement ledger, but it cannot import open-cover compactness, countable choice, selected representatives, quotient point equality, a hidden Lebesgue-number selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the source rows $K, M, Q, \mathcal{P}$. They expose compact metric data, metric comparisons, positive rational radii, and the finite cover bundle before any refinement is consumed.

Read the regular core and refinement ledger through C and L. These rows only relate displayed bundled balls to displayed refined cores, so they provide the finite cover-refinement boundary needed by compact-uniform-continuity and Lebesgue-number consumers.

Finally use H, T, P, N to transport, replay, source, and name the route. The carrier has no coordinate for the excluded compactness, choice, quotient, selector, representative, or completion objects, so no downstream handoff can escape the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactRegularCover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactRegularCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6796.1, Definition 6796.2, Definition 6796.3, and Definition 6796.4 seed the finite regular cover-refinement route over compact metric, metric-space, rational-radius, ProbeBundle, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactRegularCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient point equality, countable choice, hidden Lebesgue-number selectors, ambient completeness, Lean verification, or bridge checking for $\mathsf{BishopCompactRegularCover}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius regularity, finite ProbeBundle refinement induction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactRegularCover

Constructive story: A $\mathsf{BishopCompactRegularCover}^{\uparrow}$ packet is generated from a CompactMetric source row, a MetricSpace distance row, positive RatUp radius ledgers, a finite ProbeBundle cover row, regular core rows, a refinement ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6797

A Concrete Naming Certificate for $\mathsf{ModulatedConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ModulatedConvergence}^{\uparrow}$ packet records convergence data by keeping an explicit modulus beside each finite source-window read. It sits with the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the Real-facing seal of chapter 38, and the modulus-controlled completion route of chapter 6766. The chapter isolates the finite rows a convergence consumer may inspect before any completed limit or ambient Real completeness is claimed.

Definition 6797.1 (Modulated convergence carrier)**.** *A* $\mathsf{ModulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, M, W, T, R, E, H, C, P, N)$$

with the following visible rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose displayed windows are the only observations admitted by the packet.*
2. *M is the explicit modulus row; for each displayed precision request it names the finite source-window depth after which the convergence read may be consumed.*
3. *W is the tail-window ledger pairing each precision request with the depth supplied by M and the corresponding source-window slice of S.*
4. *T is the tail-stability row recording that later displayed windows are read through the same modulus-controlled ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S, M, W, T before any Real-facing seal is visible.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source windows, explicit modulus, tail-window ledger, tail-stability row, and regular rational readback have been consumed.*
7. *H records componentwise* hsame *transport for S, M, W, T, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from precision requests to modulus depths, from depths to tail windows, from windows to regular rational readback, and from readback to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulatedConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulatedConvergence}^{\uparrow}}$ *row.*

The carrier exports only this finite explicit-modulus convergence surface. It contains no selected limit, host convergence predicate, quotient completion equality, countable-choice scheduler, host real equality, or ambient completeness row for $\mathsf{Real}^{\uparrow}$.

Theorem 6797.2 (Modulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulatedConvergence}^{\uparrow}$ *carrier* $Q = (S, M, W, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ModulatedConvergence}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission exposes precisely the rows* $S, M, W, T, R, E, H, C, P, N$ *before a convergence-facing read is exported;*
2. *modulus coverage requires every displayed precision request in* W *to be served by a finite depth in* M *over the same source stream* S*;*
3. *tail-window stability requires later inspected windows to pass through the same ledger* W *and tail row* T*, not through an external convergence oracle;*
4. *regular readback and Real sealing require every Real-facing read to factor through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *before* E *is visible;*
5. *provenance and transport require* H, C, P, N *to preserve the same displayed route under* hsame *transport,* Cont *replay, package sourcing, and local naming.*

Proof. Carrier admission is projection from Definition 6797.1. The packet lists the source stream, explicit modulus, tail-window ledger, tail-stability row, regular readback, Real seal, and structural rows as its entire public surface.

For modulus coverage and tail-window stability, inspect M, W, T together. Each precision request is admitted only with a displayed depth and a finite tail-window record over the same stream row S, so the packet cannot consult a host convergence predicate or countable-choice schedule.

The regular readback then consumes S, M, W, T, R in order. The sibling chapters chapter 126 and chapter 301 supply the stream and regular rational surfaces named by the carrier.

Finally the Real seal E is downstream of that route, and H, C, P, N only transport, replay, source, and name the same packet. Since the carrier has no selected-limit, quotient-completion, host-equality, or ambient-completeness coordinate, a convergence consumer receives only the displayed $\mathsf{Real}^{\uparrow}$ seal boundary. □

Theorem 6797.3 (Modulated convergence RegSeqRat handoff)**.** *For every accepted* $\mathsf{ModulatedConvergence}^{\uparrow}$ *carrier* $Q = (S, M, W, T, R, E, H, C, P, N)$*, the L10 handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *is exactly the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto M \longmapsto W \longmapsto T \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto (H, C, P, N).$$

The handoff carries source windows, explicit modulus depths, tail-window stability, rational regular readback, Real sealing, transport, replay, provenance, and local naming. It carries no separate limit selector, host enumeration of tails, quotient completion equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or representation source outside the packet.*

Proof. Start from the carrier definition. The rows S, M, W, T display the source windows and the explicit tail depth required for each precision request.

Apply the NameCert obligation surface of Definition 6797.2. Regular readback and Real sealing require R to occur before E, and the transport-provenance rows preserve the same route after the seal is read.

The listed L10 handoff is therefore the only route available to a downstream RegSeqRat or Real consumer. Every excluded object would require a coordinate absent from Q, so it cannot be extracted from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulatedConvergence}^{\uparrow}$ packet is generated from source StreamName windows, an explicit modulus row, a tail-window ledger, tail-stability data, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6797.1, Definition 6797.2, and Definition 6797.3 seed the finite explicit-modulus convergence route for RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, host convergence predicates, quotient completion equality, countable choice, full RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus coverage, tail-window stability, regular readback, Real-seal nonescape, and transport-provenance preservation for BEDC.Derived.ModulatedConvergenceUp.

Chapter 6798

A Concrete Naming Certificate for $\mathsf{T1Space}^{\uparrow}$

Following Engelking and Kelley on separation axioms, the $\mathsf{T1Space}^{\uparrow}$ packet records the finite singleton-closed surface needed by Real-facing topology consumers before they ask for stronger separation. It sits over the topology carrier of chapter 88, the set-like point and singleton rows of chapter 3668, the metric-neighbouring separation route of chapter 57, and the terminal real-name seal of chapter 38. The packet names only the displayed point classifier and singleton-closed witnesses; it does not assert Hausdorff separation, compactness, quotient spaces, host point equality, or choice.

Definition 6798.1 (T1 space carrier)**.** *A* $\mathsf{T1Space}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, O, Q, S, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{SetLike}^{\uparrow}$ *point-carrier row, exposing only the finite public point observations admitted by the packet.*
2. *O is the* $\mathsf{Topology}^{\uparrow}$ *open-row packet, recording the open observations and finite topology provenance currently under inspection.*
3. *Q is the point classifier row. It classifies inspected point rows only through the displayed* $\mathsf{SetLike}^{\uparrow}$ *membership and* $\mathsf{Topology}^{\uparrow}$ *open data.*
4. *S is the singleton-closed witness row. For each inspected point row accepted by Q, it records the finite boundary witness that the singleton is closed in the displayed topology packet.*
5. *B is the Real-facing boundary row, reached by metric and real consumers only after X, O, Q, S have been displayed.*
6. *H records componentwise* hsame *transport for the point carrier, topology packet, classifier row, singleton-closed witness row, boundary row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from point and open observations to singleton-closed witnesses and then to the Real-facing boundary row.*
8. *P is the* Pkg *provenance over* $\mathsf{T1Space}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{T1Space}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, O, Q, S, B, H, C, P, N$. It has no coordinate for Hausdorff separation, compactness, quotient spaces, host point equality, selected representatives, or choice.

Theorem 6798.2 (T1 space NameCert obligations)**.** *For every accepted* $\mathsf{T1Space}^\uparrow$ *carrier $T = (X, O, Q, S, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{T1Space}^\uparrow}$ *surface consists exactly of carrier admission for the displayed point carrier, topology packet, point classifier, singleton-closed witness row, Real-facing boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Any topology, metric, or Real-facing consumer must expose X, O, Q, S before the boundary row B can be read.*

Proof. Project the carrier of Definition 6798.1. The accepted packet exposes exactly the point carrier X, topology row O, classifier Q, singleton-closed witnesses S, boundary row B, and structural rows H, C, P, N.

Read the point and topology sources through chapter 3668 and chapter 88. The classifier Q is therefore tied to displayed membership and open-row data, not to host point equality.

The singleton-closed row S is then the only separation witness available to the packet. Transport, replay, provenance, and local naming preserve that witness through H, C, P, N without adding a Hausdorff, compactness, quotient, or choice coordinate. □

Theorem 6798.3 (T1 singleton-closed boundary)**.** *Every accepted* $\mathsf{T1Space}^\uparrow$ *packet that exports a Real-facing or metric-facing separation read must first expose the route*

$$\mathsf{SetLike}^\uparrow \longmapsto \mathsf{Topology}^\uparrow \longmapsto Q \longmapsto S \longmapsto B \longmapsto H, C, P, N.$$

The exported boundary is exactly singleton-closed separation for displayed point rows. It is not Hausdorff separation, not compactness, not a quotient-space theorem, not host point equality, and not a choice principle.

Proof. Start with the NameCert obligations of Definition 6798.2. They force every consumer to read the point carrier, topology packet, classifier, and singleton-closed witness row before the boundary row is available.

Use the metric and real anchors only as consumers. The chapters chapter 57 and chapter 38 may read the displayed boundary B, but the carrier gives them no coordinate for stronger separation or selected host points.

Finally inspect the structural rows. The hsame transport, Cont replay, Pkg provenance, and local NameCert row preserve the same finite singleton-closed route, so the exported boundary cannot be retyped as Hausdorff separation, compactness, quotient data, host point equality, or choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{T1Space}^\uparrow}$**.**

Constructive story: A $\mathsf{T1Space}^\uparrow$ packet is generated from a SetLikeUp point carrier, a TopologyUp open-row packet, a point classifier, singleton-closed witnesses, a Real-facing boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{T1Space}^\uparrow}$, seed)

Scope closed: Definition 6798.1, Definition 6798.2, and Definition 6798.3 seed the finite singleton-closed separation surface for Real-facing topology consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{T1Space}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hausdorff separation, compactness, quotient spaces, host point equality, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology packet visibility, point classifier exactness, singleton-closed witness stability, Real-facing boundary non-escape, transport, replay, provenance, and local naming for BEDC.Derived.T1SpaceUp.

Chapter 6799

A Concrete Naming Certificate for $\mathsf{PrueferSurface}^{\uparrow}$

Following Pruefer and Kelley-style topology, the $\mathsf{PrueferSurface}^{\uparrow}$ packet records a surface built from real-coordinate chart rows and a boundary fan of neighbourhood rows as finite NameCert data. It sits after the point-neighbourhood discipline of chapter 5921, the indexed-open surface of chapter 88, the binary coordinate route of chapter 2000, the real-coordinate seal of chapter 38, and the chart interface of chapter 96. The packet names only the displayed chart-boundary route; it does not import a quotient surface, a selected atlas completion, host equality of real pairs, choice-selected fans, or ambient manifold equality.

Definition 6799.1 (Pruefer surface carrier)**.** *A* $\mathsf{PrueferSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, Q, F, B, O, T, M, H, C, P, N)$$

with the following displayed rows:

1. *R is the real-coordinate chart row, exposing each local coordinate only through* $\mathsf{Real}^{\uparrow}$ *and the paired-coordinate readback supplied by* $\mathsf{ProductTopology}^{\uparrow}$*.*
2. *Q is the chart-index row, listing the visible ordinary coordinate charts and the boundary-indexed fan charts without importing an external atlas.*
3. *F is the boundary fan row, recording for each displayed boundary index the finite neighbourhood rows that approach that boundary coordinate from the corresponding side.*
4. *B is the boundary-neighbourhood exactness row, identifying which fan rows meet the boundary line and which ordinary rows remain in the interior chart region.*
5. *O is the chart-overlap row, recording the finite overlap incidences and transition-read requests between displayed chart rows.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *handoff row generated from the ordinary chart rows, boundary fan rows, and displayed overlap incidences.*
7. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart-read row available only after R, Q, F, B, O, T have been displayed.*
8. *H records componentwise* hsame *transport for the chart, boundary fan, boundary-neighbourhood, overlap, topology, manifold, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a chart request through coordinate readback, boundary fan selection, overlap incidence, topology handoff, manifold read, transport, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{PrueferSurface}^{\uparrow}$, $\mathsf{NiemytzkiPlane}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ProductTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrueferSurface}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotienting chart copies, selected atlas refinements, ambient equality of real-coordinate pairs, choice-selected neighbourhood fans, or a host manifold identity.

Theorem 6799.2 (Pruefer surface NameCert obligations)**.** *For every accepted* $\mathsf{PrueferSurface}^{\uparrow}$ *carrier*

$$S = (R, Q, F, B, O, T, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrueferSurface}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, chart compatibility through R, Q, O, boundary-neighbourhood exactness through F, B, topology handoff through T, real and product-coordinate readback through R, manifold readback through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, local naming through N, and nonescape from quotient surfaces, selected atlas refinements, host real-pair equality, choice-selected fans, or ambient manifold equality.*

Proof. Project the carrier of Definition 6799.1. The tuple displays chart coordinates, chart indices, boundary fan rows, boundary-neighbourhood exactness, overlap incidences, topology and manifold handoffs, transport, replay, provenance, and local naming as separate rows.

The chart and boundary rows are read before any topology handoff. The row R forces each coordinate through $\mathsf{Real}^{\uparrow}$ and $\mathsf{ProductTopology}^{\uparrow}$, while Q, F, B distinguish ordinary chart reads from boundary fan neighbourhood reads in the same finite packet.

Next inspect the overlap row O. Chart compatibility is therefore a displayed finite incidence condition between the rows already named by R, Q, F, B, not an equality of host charts or a quotient atlas.

Finally pass through T, M, H, C, P, N. These rows turn the displayed chart and boundary data into topology and manifold-facing reads while preserving the same finite route. Since no row supplies quotienting, selected atlas refinement, hidden real-pair equality, choice-selected fans, or ambient manifold identity, the listed rows exhaust the local NameCert surface. □

Theorem 6799.3 (Pruefer surface chart-boundary route)**.** *Every chart-boundary consumer of an accepted* $\mathsf{PrueferSurface}^{\uparrow}$ *carrier must factor through*

$$R, \quad Q, \quad F, \quad B, \quad O$$

before any $\mathsf{Manifold}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *read is available. Thus the consumer sees a displayed chart row, a displayed boundary fan row, the boundary-neighbourhood exactness row, and the chart-overlap row before it may use the topology or manifold handoff.*

Proof. Start with a chart-boundary request. The only coordinate source is R, so the request first reads real-coordinate data through $\mathsf{Real}^{\uparrow}$ and $\mathsf{ProductTopology}^{\uparrow}$.

The chart-index and fan rows then determine the boundary side of the request. The rows Q and F expose the ordinary chart and the boundary-indexed fan, while B records the exact boundary-neighbourhood incidence.

Only after those rows are visible can the overlap row O compare the chart request with neighbouring chart rows. The topology row T and manifold row M are downstream of this displayed comparison, and H, C, P, N preserve the same order under transport, replay, provenance, and naming.

Consequently no chart-boundary consumer receives a host quotient surface, a selected atlas refinement, an untracked equality of real pairs, a choice-selected fan, or an ambient manifold identity before reading the displayed chart and boundary rows. □

Falsifiable prediction: Every accepted $\mathsf{PrueferSurface}^{\uparrow}$ chart-boundary read displays real-coordinate chart rows, boundary fan rows, boundary-neighbourhood exactness, chart-overlap rows, topology and manifold handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: niemytzki-plane, topology, product-topology, real, manifold: $\mathsf{PrueferSurface}^{\uparrow}$ is not bijective to any listed sibling because it binds ordinary chart rows, boundary fan neighbourhood rows, overlap incidence, and manifold-facing chart readback into one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PrueferSurface}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrueferSurface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6799.1, Definition 6799.2, and Definition 6799.3 seed the finite real-coordinate chart, boundary fan, overlap, topology, manifold, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrueferSurface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient surfaces, selected atlas refinements, host real-pair equality, choice-selected boundary fans, ambient manifold equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chart compatibility, boundary-neighbourhood exactness, topology handoff, real and product-coordinate readback, manifold readback, and chart-boundary nonescape for BEDC.Derived.PrueferSurfaceUp.

Constructive story: A $\mathsf{PrueferSurface}^{\uparrow}$ packet is generated from real-coordinate chart rows, visible chart indices, boundary fan rows, boundary-neighbourhood exactness rows, chart-overlap rows, topology and manifold handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6800

A Concrete Naming Certificate for $\mathsf{OvertSpace}^{\uparrow}$

Following Bishop and Johnstone, the $\mathsf{OvertSpace}^{\uparrow}$ packet records witnessable positive contact for a displayed constructive space as finite NameCert data. It sits between the cover/refinement presentation of chapter 3967, the point-free open rows of chapter 5153, the positive real-contact surface of chapter 3354, and the compact positive-cover consumer of chapter 6548. The packet supplies positivity data to located real and formal-topology consumers without importing host open-set primitives, powersets, selected points, excluded middle over opens, or choice.

Definition 6800.1 (Overt space carrier)**.** *An* $\mathsf{OvertSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, B, P, W, R, L, H, C, G, N)$$

with the following displayed rows:

1. *S is the underlying space or formal presentation row whose positive observations are being named.*
2. *B is the basic-neighbourhood row read from* $\mathsf{FormalTopology}^{\uparrow}$ *or* $\mathsf{LocaleSpace}^{\uparrow}$*, exposing only finite cover and refinement data.*
3. *P is the positivity predicate row. It records witnessable contact with the displayed basic neighbourhood, not a host proposition over arbitrary opens.*
4. *W is the finite witness row that certifies the positive contact claimed by P.*
5. *R is the located-real specialization row through which* $\mathsf{OvertReal}^{\uparrow}$ *consumers read positive interval contact.*
6. *L is the located compact handoff row through which* $\mathsf{LocatedCompactOvert}^{\uparrow}$ *consumers combine positive contact with finite cover data.*
7. *H records componentwise* hsame *transport for the space, basis, positivity, witness, specialization, compact handoff, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of a positive-contact request through $S, B, P, W, R, L, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{OvertSpace}^{\uparrow}$, $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{OvertReal}^{\uparrow}$, $\mathsf{LocatedCompactOvert}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OvertSpace}^{\uparrow}}$ *row.*

The carrier exposes no powerset of opens, no host open membership relation, no selected point of contact, no spatial quotient, no trichotomy over basic opens, and no choice principle for witnesses not already displayed in W.

Lean4 ✓ `BEDC.Derived.OvertSpaceUp.OvertSpaceTasteGate single carrier alignment`

Theorem 6800.2 (Overt space positivity stability)**.** *Every accepted* $\mathsf{OvertSpace}^{\uparrow}$ *carrier preserves positive contact through the displayed* hsame *and* Cont *rows. Concretely, if a consumer reads a positive neighbourhood claim from $O = (S, B, P, W, R, L, H, C, G, N)$, then the claim must be replayed from the same basis row B, the same positivity row P, and the same finite witness row W after transport. The replay may be consumed by formal-topology, locale, overt-real, or located-compact-overt siblings, but it cannot be replaced by a host open-set predicate, a selected point, a powerset operation, or a choice-selected witness.*

Theorem 6800.3 (Overt space located-real specialization)**.** *Every located-real or located-compact consumer of an accepted* $\mathsf{OvertSpace}^{\uparrow}$ *packet factors through*

$$\mathsf{FormalTopology}^{\uparrow},\ \mathsf{LocaleSpace}^{\uparrow},\ \mathsf{OvertReal}^{\uparrow},\ \mathsf{LocatedCompactOvert}^{\uparrow}.$$

The formal-topology and locale rows supply finite basic-neighbourhood and cover-refinement data, the $\mathsf{OvertReal}^{\uparrow}$ *row specializes positive contact to located real intervals, and the* $\mathsf{LocatedCompactOvert}^{\uparrow}$ *row consumes the same witness data together with finite compact-cover rows. The route exports positivity evidence, not host opens, quotient spatial equality, trichotomy, selected points, or ambient real completeness.*

Closure status for $\mathsf{NameCert}_{\mathsf{OvertSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{OvertSpace}^{\uparrow}$ packet is generated from a displayed space or formal presentation row, finite basic-neighbourhood rows, a positivity row, a finite witness row, located-real specialization, located-compact handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OvertSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6800.1, Definition 6800.2, and Definition 6800.3 seed the finite positivity surface for constructive topology consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OvertSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host open-set primitives, powersets, spatial quotient equality, selected points, trichotomy over opens, choice-selected witnesses, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, positivity stability, finite witness exactness, formal-topology and locale routing, located-real specialization, located-compact handoff, transport, replay, provenance, and local naming for BEDC.Derived.OvertSpaceUp.

Chapter 6801

A Concrete Naming Certificate for $\mathsf{KorovkinTheorem}^{\uparrow}$

Following Korovkin, the $\mathsf{KorovkinTheorem}^{\uparrow}$ packet records the classical test-function criterion for positive approximation operators on a compact interval as finite BEDC rows. It consumes the compact source surface of chapter 60, the continuous-map carrier of chapter 130, the polynomial row of chapter 50, the Bernstein approximation boundary of chapter 3730, the uniform-convergence ledger of chapter 3417, and the polynomial approximation handoff of chapter 1634. The packet names the finite test-function route; it is not an open-cover compactness theorem, a host norm-completeness argument, a selected infinite operator sequence, quotient equality of functions, or a proof of positivity outside the displayed rows.

Definition 6801.1 (Korovkin theorem carrier)**.** *A* $\mathsf{KorovkinTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, T, O, B, W, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact source row, read through* $\mathsf{Compact}^{\uparrow}$*.*
2. *F is the continuous-map target row, read through* $\mathsf{ContinuousMap}^{\uparrow}$*.*
3. *T is the three-test-function row, naming the constant, coordinate, and square-coordinate observations.*
4. *O is the positive-operator row, recording the finite operator family under inspection.*
5. *B is the Bernstein comparison row, read through* $\mathsf{BernsteinPolynomial}^{\uparrow}$*.*
6. *W is the polynomial approximation handoff row, read through* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$*.*
7. *U is the uniform-convergence row, recording the common tolerance schedule for the target function.*
8. *H records componentwise* hsame *transport for source, target, test, operator, Bernstein, Weierstrass, uniform-convergence, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the three test functions through the polynomial handoff and into the target uniform-convergence row.*

10. *P is the* Pkg *provenance over* $\mathsf{KorovkinTheorem}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{BernsteinPolynomial}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{WeierstrassApproximation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KorovkinTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, T, O, B, W, U, H, C, P, N$. It has no coordinate for an ambient Banach-space norm proof, a hidden positivity theorem, a selected infinite subsequence, quotient equality of functions, or a compactness principle beyond the displayed compact source row.

Theorem 6801.2 (Korovkin test-function sufficiency)**.** *For every accepted* $\mathsf{KorovkinTheorem}^{\uparrow}$ *carrier $K = (X, F, T, O, B, W, U, H, C, P, N)$, a consumer may read the approximation conclusion only after the positive-operator row O has been checked on the three-test-function row T over the compact source row X. The conclusion then routes through the Bernstein row B, the polynomial handoff row W, and the continuous-map row F before it reaches the uniform-convergence row U.*

Proof. Project the carrier of Definition 6801.1. The source, target, test, operator, Bernstein, Weierstrass, and uniform-convergence rows are displayed as separate coordinates.

The test-function row is the only row that licenses the Korovkin read. It exposes the constant, coordinate, and square-coordinate observations before a consumer can use positivity of O.

The approximation route then passes through the named siblings. Bernstein supplies a finite polynomial comparison, Weierstrass supplies the polynomial approximation handoff, and the continuous-map row fixes the inspected target before the uniform-convergence ledger is read.

Thus the sufficiency claim is a finite carrier route through T, O, B, W, F, U over X, not an ambient norm argument or an unrecorded positivity theorem. □

Theorem 6801.3 (Korovkin uniform-convergence handoff)**.** *Every* $\mathsf{UniformConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{KorovkinTheorem}^{\uparrow}$ *packet must receive the displayed route*

$$X, F, T, O, B, W, U, H, C, P, N.$$

The row U may be consumed only after T has exposed the three test functions, O has retained positivity for the inspected family, B has supplied the Bernstein comparison, and W has supplied the Weierstrass–polynomial handoff.

Proof. Start with Definition 6801.2. It forces the approximation conclusion through the compact source, continuous target, three-test-function, positive-operator, Bernstein, polynomial, and uniform-convergence rows.

Next read the structural rows H, C, P, N. They transport, replay, source, and name the same finite route, so the uniform-convergence consumer receives the full packet rather than a detached convergence assertion.

Since U is reached only after T, O, B, W are visible, the handoff cannot be retyped as a host supremum-norm theorem, a selected infinite operator sequence, quotient equality of maps, or a compactness principle outside the displayed $\mathsf{Compact}^{\uparrow}$ row. □

Theorem 6801.4 (Korovkin theorem NameCert obligations)**.** *For every accepted* $\mathsf{KorovkinTheorem}^{\uparrow}$ *carrier $K = (X, F, T, O, B, W, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KorovkinTheorem}^{\uparrow}}$ *surface consists exactly of compact-source exposure through X, continuous-target exposure through F, three-test-function exposure through T, positive-operator admission through O, Bernstein comparison through B, polynomial approximation handoff through W, uniform-convergence export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6801.1. The analytic rows are X, F, T, O, B, W, U, and the structural rows are H, C, P, N.

The test-function sufficiency theorem keeps T and O visible before approximation is read. The uniform-convergence handoff theorem keeps B, W, U visible before the downstream convergence consumer may use the packet.

The transport, replay, provenance, and local naming rows preserve the same route. Since no further carrier coordinate exists, the listed rows exhaust the local NameCert obligations and exclude host compactness, hidden positivity, quotient equality, selected infinite data, and ambient completeness readings. □

Falsifiable prediction: An accepted $\mathsf{KorovkinTheorem}^{\uparrow}$ consumer that reaches $\mathsf{UniformConvergence}^{\uparrow}$ without first displaying the three-test-function row and the positive-operator row is outside the carrier surface.

Independence witness: compact, continuousmap, polynomial, bernstein-polynomial, uniform-convergence, weierstrass-approximation: none is bijective to $\mathsf{KorovkinTheorem}^{\uparrow}$ because only this packet binds the compact source, continuous target, Korovkin test rows, positive-operator family, Bernstein comparison, Weierstrass polynomial handoff, uniform-convergence ledger, transport, replay, provenance, and local NameCert row into one accepted route.

Closure status for $\mathsf{NameCert}_{\mathsf{KorovkinTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{KorovkinTheorem}^{\uparrow}$ packet is generated from compact source, continuous target, three-test-function, positive-operator, Bernstein comparison, polynomial approximation, uniform-convergence, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KorovkinTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6801.1, Definition 6801.2, Definition 6801.3, and Definition 6801.4 seed the finite Korovkin approximation packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KorovkinTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary compact-space Korovkin variants, host norm-completeness arguments, or positivity outside the displayed operator row.

Upgrade path: Obligation closure requires checked carrier admission, test-function exactness, positive-operator retention, Bernstein and Weierstrass handoff exactness, uniform-convergence export, transport, replay, provenance, and local naming for BEDC.Derived.KorovkinTheoremUp.

Chapter 6802

A Concrete Naming Certificate for MetaCICConfluenceInventory$^{\uparrow}$

The MetaCICConfluenceInventory$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as a finite inventory separating checked fragments from retained conditional premises. It sits beside the parallel-diamond frontier of chapter 4238, the critical-pair boundary of chapter 2212, the closed-normal confluence seal of chapter 741, and the residual-substitution boundary of chapter 2181. The packet is an audit inventory, not an unconditional Church–Rosser theorem or a discharged substitution-compatibility proof.

Definition 6802.1 (MetaCIC confluence inventory carrier)**.** *A* MetaCICConfluenceInventory$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, A, NF, P, D, R, O, H, C, G, M)$$

with the following displayed rows:

1. *K is the checked-fragment row, listing the closed-normal and atom-level confluence fragments already available to the audit.*
2. *A is the audit-map row, recording the finite readback from chapter 9378 that classifies each confluence-facing component.*
3. *NF is the closed-normal seal row, read through* ClosedNormalConfluenceSeal$^{\uparrow}$ *before any closed-source handoff is exposed.*
4. *P is the parallel-diamond frontier row, retaining the premise boundary named by* MetaCICParallelDiamondFrontier
5. *D is the critical-pair boundary row, recording inspected peaks and candidate common-reduct witnesses.*
6. *R is the residual-substitution premise row, keeping the substitution-compatibility boundary visible.*
7. *O is the conditional-obligation row, separating retained premises from checked fragments before any confluence handoff is consumed.*
8. *H records componentwise* hsame *transport for checked fragments, audit-map rows, closed-normal seals, frontier rows, critical-pair rows, residual-substitution rows, conditional obligations, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a confluence-audit request through K, A, NF, P, D, R, O.*
10. *G is the* Pkg *provenance over* MetaCICConfluenceInventory$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$, ClosedNormalConfluenceSeal$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *M is the local* NameCert$_{\text{MetaCICConfluenceInventory}^{\uparrow}}$ *row.*

The classifier compares accepted inventories only through $K, A, NF, P, D, R, O, H, C, G, M$. It has no coordinate for an unconditional parallel diamond, a global Church–Rosser theorem, discharged residual-substitution compatibility, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 6802.2 (MetaCIC confluence inventory NameCert obligations)**.** *For every accepted* MetaCICConfluenceInventory$^{\uparrow}$ *carrier $I = (K, A, NF, P, D, R, O, H, C, G, M)$, the local* NameCert$_{\text{MetaCICConfluenceInven}}$ *surface consists exactly of carrier admission for checked-fragment rows, confluence audit-map rows, closed-normal seal rows, parallel-diamond frontier rows, critical-pair boundary rows, residual-substitution premise rows, conditional-obligation rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The proposed Lean follow-up must provide Nontrivial* MetaCICConfluenceInventory$^{\uparrow}$ *and FieldFaithful* MetaCICConfluenceInventory$^{\uparrow}$ *instances in TasteGate.lean, so the inventory is nonempty and every displayed row is classifier-visible.*

Proof. Project the carrier of Definition 6802.1. The projection exposes the checked-fragment inventory, audit-map row, closed-normal seal, parallel frontier, critical-pair boundary, residual-substitution premise, conditional-obligation row, and structural rows.

The checked side is read through K, A, NF. These rows may expose closed-normal and atom-level fragments, but they do not discharge the residual-substitution or closed-diamond premises retained by P, D, R, O.

The conditional side is read through the frontier, critical-pair, residual-substitution, and obligation rows. Since those rows remain visible before any handoff, the packet separates checked fragments from premises instead of promoting the audit map to an unconditional confluence theorem.

Finally H, C, G, M transport, replay, source, and name the same inventory. Nontrivial and FieldFaithful TasteGate evidence must certify that the inventory carries genuine rows and that no row can be ignored by the classifier. □

Theorem 6802.3 (MetaCIC confluence inventory conditional boundary)**.** *Every confluence-facing read of an accepted* MetaCICConfluenceInventory$^{\uparrow}$ *packet must expose the substitution-compatibility premise row R and the closed-diamond premise retained by P, D, O before any Church–Rosser handoff can be consumed. Checked fragments K, A, NF remain available as finite rows, but the packet exports no unconditional parallel diamond, no global Church–Rosser theorem, no discharged residual-substitution compatibility, and no kernel-consistency endpoint.*

Proof. Begin with the NameCert obligations of Definition 6802.2. They keep the checked-fragment rows and the retained premise rows in the same accepted carrier.

Use the parallel-diamond frontier and critical-pair boundary only as displayed dependencies. The rows P and D expose the closed-diamond and critical-pair boundary data, while R keeps substitution compatibility as a premise rather than a theorem.

The conditional-obligation row O then blocks a premise-free Church–Rosser handoff. The structural rows H, C, G, M preserve that split, so a downstream confluence consumer can read the inventory but cannot read any refused unconditional endpoint. □

Falsifiable prediction: Every confluence-facing read of $\mathsf{MetaCICConfluenceInventory}^{\uparrow}$ exposes the substitution-compatibility and closed-diamond premise rows before any Church–Rosser handoff.

Independence witness: metacic-parallel-diamond-frontier, metacic-confluence-critical-pair-boundary, closednormalconfluenceseal, metacic-residual-substitution-boundary: $\mathsf{MetaCICConfluenceInventory}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries checked-fragment, audit-map, closed-normal, parallel-frontier, critical-pair, residual-substitution, conditional-obligation, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceInventory}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceInventory}^{\uparrow}$ packet is generated from checked confluence fragments, an audit-map row, a closed-normal seal row, a parallel-diamond frontier row, a critical-pair boundary row, a residual-substitution premise row, conditional-obligation data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceInventory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6802.1, Definition 6802.2, and Definition 6802.3 seed the finite inventory separating checked confluence fragments from retained premises.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceInventory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional parallel diamond, global Church–Rosser, discharged substitution compatibility, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, checked-fragment visibility, audit-map readback, closed-normal seal exposure, parallel-frontier premise retention, critical-pair boundary retention, residual-substitution boundary retention, conditional-obligation non-escape, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICConfluenceInventoryUp.

Chapter 6803

A Concrete Naming Certificate for $\mathsf{BrownRepresentability}^{\uparrow}$

Following Brown's representability theorem and Neeman's triangulated-category formulation, the $\mathsf{BrownRepresentability}^{\uparrow}$ packet records representability as finite BEDC NameCert data for a displayed homological consumer. It sits after the category carrier of chapter 61, the functorial row of chapter 62, the natural-transformation surface of chapter 63, the homology carrier of chapter 98, the derived-category row of chapter 160, and the triangulated-category row of chapter 179. The packet names a finite representability ledger for inspected homological functors; it does not assert the host Brown theorem, arbitrary coproduct exactness, selected representing objects, quotient functor equality, or ambient set-theoretic choice.

Definition 6803.1 (Brown representability carrier)**.** *A* $\mathsf{BrownRepresentability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, T, F, N, H, D, R, L, S, P, M)$$

with the following displayed rows:

1. *C is the source* $\mathsf{Cat}^{\uparrow}$ *row carrying the inspected objects and morphisms.*
2. *T is the* $\mathsf{TriangulatedCat}^{\uparrow}$ *or* $\mathsf{DerivedCat}^{\uparrow}$ *structure row used by the homological consumer.*
3. *F is the* $\mathsf{Functor}^{\uparrow}$ *row for the inspected contravariant or covariant homological functor.*
4. *N is the* $\mathsf{NatTrans}^{\uparrow}$ *naturality row recording the finite naturality squares that the consumer asks to inspect.*
5. *H is the* $\mathsf{Homology}^{\uparrow}$ *row read by the finite homology consumer.*
6. *D records the displayed distinguished-triangle or derived-handoff obligations that feed the inspected homological window.*
7. *R is the representability ledger, listing the candidate representing-object name together with the finite Hom-window comparison rows.*
8. *L records local exactness and coproduct-compatibility tests only for the displayed finite window.*
9. *S records componentwise* hsame *transport for the category, triangulated, functor, naturality, homology, handoff, ledger, and local-test rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BrownRepresentability}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{DerivedCat}^{\uparrow}$, $\mathsf{TriangulatedCat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *M is the local* $\mathsf{NameCert}_{\mathsf{BrownRepresentability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a global representing object, host Hom-set equality, arbitrary coproduct preservation, quotient functor equality, selected injective resolutions, or ambient choice.

Theorem 6803.2 (Brown representability NameCert obligations)**.** *For every accepted* $\mathsf{BrownRepresentability}^{\uparrow}$ *carrier* $B = (C, T, F, N, H, D, R, L, S, P, M)$*, the local* $\mathsf{NameCert}_{\mathsf{BrownRepresentability}^{\uparrow}}$ *surface consists exactly of carrier admission for the category row, triangulated or derived row, functor row, naturality row, homology row, distinguished-triangle handoff row, representability ledger, finite exactness and coproduct tests, componentwise transport, provenance, and local naming. A consumer may read representability obligations only from those rows and receives no host Brown theorem or selected representing object.*

Proof. Project the carrier of Definition 6803.1. The category, triangulated or derived, functor, naturality, homology, handoff, representability-ledger, local-test, transport, provenance, and naming rows are the only coordinates exposed by the packet.

The representability ledger is therefore finite. It records a candidate object name and the Hom-window comparison rows inspected by the consumer, while L stores only the local exactness and coproduct tests that have been displayed.

Finally read the structural rows S, P, M. They transport, source, and name the same packet, so no additional row can provide a global representing object, arbitrary coproduct preservation, quotient functor equality, selected resolution data, host Hom equality, or ambient choice. □

Theorem 6803.3 (Brown representability finite homology consumer)**.** *Every finite homology consumer of an accepted* $\mathsf{BrownRepresentability}^{\uparrow}$ *packet must read the route*

$$C, T, F, N, H, D \longmapsto R, L$$

before it may use the structural rows S, P, M. Thus the consumer receives a displayed representability ledger for the inspected homological window, not an unrestricted Brown representability theorem, a selected representing object, quotient functor equality, arbitrary coproduct exactness, or ambient category-theoretic choice.

Proof. Start from the NameCert obligations of Definition 6803.2. They force every consumer to expose the category, triangulated or derived, functor, naturality, homology, and handoff rows before the representability ledger can be inspected.

The finite consumer then reads R, L. The ledger R supplies the displayed Hom-window comparison data, and L supplies only the local exactness and coproduct tests attached to that window.

The structural rows S, P, M preserve and name this same finite route. Since the carrier contains no row for a global representing object, quotient functor equality, selected resolutions, arbitrary coproduct preservation, host Hom equality, or ambient choice, the exported consumer surface is exactly the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrownRepresentability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BrownRepresentability}^{\uparrow}$ packet is generated from category, triangulated or derived, functor, naturality, homology, distinguished-triangle handoff, representability ledger, local exactness, transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrownRepresentability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6803.1, Definition 6803.2, and Definition 6803.3 seed the finite representability NameCert surface for homological consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrownRepresentability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the host Brown theorem, arbitrary coproduct exactness, selected representing objects, quotient functor equality, selected resolutions, ambient choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, functor and naturality exposure, homology handoff exactness, representability ledger readback, finite exactness tests, transport, provenance, and local naming for BEDC.Derived.BrownRepresentabilityUp.

Chapter 6804

A Concrete Naming Certificate for $\mathsf{CauchyCompleteSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCompleteSpace}^{\uparrow}$ packet records a complete-space interface by keeping regular Cauchy names, finite stream windows, rational readback, and the completion-facing seal visible in one NameCert carrier. It sits after the finite observation surface of chapter 126, the regular rational sequence boundary of chapter 301, the terminal real-name surface of chapter 38, and the completion routes of chapter 1506 and chapter 1569. The packet names completeness through displayed regular Cauchy data; it does not import quotient streams, selected limits, countable choice, or ambient host completeness.

Definition 6804.1 (Cauchy complete space carrier)**.** *A* $\mathsf{CauchyCompleteSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying displayed space or completion source row.*
2. *D is the distance or apartness-control row used by Cauchy consumers.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for inspected Cauchy names.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational estimates exposed by W and D.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row through which endpoint reads are consumed.*
6. *E is the completion-envelope row, read through the metric or Cauchy-filter completion siblings.*
7. *L is the limit-handoff row asserting that an accepted regular Cauchy name is read through W, Q, R, E before any complete-space consumer sees a point.*
8. *H records componentwise* hsame *transport for the space, control, window, readback, seal, envelope, limit-handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from precision requests to stream windows, regular rational readback, Real seals, completion-envelope reads, and complete-space consumers.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompleteSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for quotient stream equality, a choice-selected representative, an ambient Cauchy-completion theorem, host metric completion, countable choice, or uninspected Real completeness.

Theorem 6804.2 (Cauchy complete space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompleteSpace}^{\uparrow}$ *carrier*

$$S = (X, D, W, Q, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompleteSpace}^{\uparrow}}$ *surface consists exactly of space-source exposure through X, distance-control exposure through D, finite Cauchy-window exposure through* $\mathsf{StreamName}^{\uparrow}$ *row W, regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, Real sealing through R, completion-envelope exposure through E, regular Cauchy limit handoff through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6804.1. Its visible analytic rows are X, D, W, Q, R, E, L, and its structural rows are H, C, P, N.

The Cauchy source is finite at every read. A consumer first sees stream windows through W and regular rational estimates through Q; only then can it inspect the Real seal R or the completion envelope E.

The handoff row L keeps this order attached to the same packet. Since the carrier has no coordinate for quotient streams, selected representatives, countable choice, or host completion, the listed rows exhaust the complete-space obligation surface. □

Theorem 6804.3 (Cauchy complete space regular-Cauchy handoff)**.** *Every regular Cauchy consumer of an accepted* $\mathsf{CauchyCompleteSpace}^{\uparrow}$ *packet reads the complete-space point through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow} \longmapsto \mathsf{CauchyCompleteSpace}^{\uparrow}.$$

Concretely, the consumer must expose the stream-window row W, the regular rational readback row Q, the Real seal row R, the completion-envelope row E, the limit-handoff row L, and the structural rows H, C, P, N before it may use the complete-space interface. The route exports no selected limit, quotient stream equality, countable-choice scheduler, host metric completion, or ambient Real completeness theorem.

Proof. Start from the NameCert obligations of Definition 6804.2. They place stream windows, regular rational readback, Real sealing, completion-envelope exposure, and the limit handoff in the same displayed carrier.

Now read the sibling route. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational estimates, $\mathsf{Real}^{\uparrow}$ supplies the endpoint seal, and $\mathsf{Completion}^{\uparrow}$ supplies the completion-facing envelope. The structural rows transport and replay that same route without adding a new limit selector.

Therefore any complete-space consumer receives exactly the displayed regular Cauchy handoff. The refused objects would require additional coordinates absent from S, so they cannot be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompleteSpace}^{\uparrow}$ packet is generated from a displayed space row, distance-control row, StreamName windows, RegSeqRat readback, a Real seal, a completion-envelope row, a regular Cauchy limit-handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompleteSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6804.1, Definition 6804.2, and Definition 6804.3 seed the finite complete-space interface for regular Cauchy and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyCompleteSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host metric completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, regular rational readback, Real-seal nonescape, completion-envelope exactness, limit-handoff stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompleteSpaceUp.

Chapter 6805

A Concrete Naming Certificate for $\mathsf{RiemannMapping}^{\uparrow}$

Following Riemann, Koebe, and Ahlfors, the $\mathsf{RiemannMapping}^{\uparrow}$ packet records the simply connected proper planar-domain uniformization route as finite NameCert data. It is grounded in the holomorphic-map surface of chapter 68, the disk target of chapter 377, the complex carrier of chapter 39, the normal-family compactness route of chapter 2186, the pointed disk bound of chapter 3796, and the disk-contraction comparison of chapter 3682. The certificate names the classical route without importing a host theorem about all complex domains, selected boundary prime ends, quotient function equality, or ambient compact-open function-space completeness.

Definition 6805.1 (Riemann mapping carrier)**.** *A* $\mathsf{RiemannMapping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, U, F, I, B, NF, SL, H, C, P, L)$$

with the following displayed rows:

1. *D is the simply connected proper complex-domain row over* $\mathsf{Complex}^{\uparrow}$*, recording the finite source-domain observations and the refusal of the whole-plane case.*
2. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *target row, including the named disk origin and disk boundary observations consumed by the mapping route.*
3. *F is the holomorphic conformal-map row read through* $\mathsf{Holomorphic}^{\uparrow}$*, recording injective local coordinate preservation from D to U.*
4. *I is the inverse-map row, displaying the holomorphic inverse on the accepted image route rather than a host inverse function.*
5. *B is the boundary-refusal row, saying that the packet exports no selected boundary extension, Caratheodory boundary theorem, prime-end classifier, or boundary homeomorphism.*
6. *NF is the* $\mathsf{NormalFamily}^{\uparrow}$ *route that supplies the compactness-style search surface for the extremal disk-normalized map.*
7. *SL is the Schwarz route through* $\mathsf{SchwarzLemma}^{\uparrow}$ *and* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$*, recording pointed normalization, disk contraction, uniqueness up to displayed disk automorphism data, and nonexpansion comparisons.*
8. *H records componentwise* hsame *transport for the domain, disk, map, inverse, boundary refusal, normal-family route, Schwarz route, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay for the same rows, so a consumer replays the finite route rather than consulting an ambient analytic-function space.*
10. *P is the* Pkg *provenance over* $\mathsf{RiemannMapping}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{NormalFamily}^{\uparrow}$, $\mathsf{SchwarzLemma}^{\uparrow}$, $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{RiemannMapping}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, U, F, I, B, NF, SL, H, C, P, L$. It has no coordinate for a selected conformal map outside the displayed normalization route, no quotient equality of holomorphic functions, no ambient compact-open completion, no boundary correspondence theorem, and no appeal to a choice principle over domains.

Theorem 6805.2 (Riemann mapping NameCert obligations)**.** *For every accepted* $\mathsf{RiemannMapping}^{\uparrow}$ *carrier $R = (D, U, F, I, B, NF, SL, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannMapping}^{\uparrow}}$ *surface consists exactly of the simply connected proper complex-domain row, the* $\mathsf{UnitDisk}^{\uparrow}$ *target row, the holomorphic conformal-map row, the inverse-map row, the boundary-refusal row, the* $\mathsf{NormalFamily}^{\uparrow}$ *route, the* $\mathsf{SchwarzLemma}^{\uparrow}$ *and* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6805.1. The analytic rows are D, U, F, I, B, NF, SL, and the structural rows are H, C, P, L.

The domain row D is the only row that names a simply connected proper complex source, while U is the only row that names the disk target. Thus any accepted read of the mapping claim must display both endpoints before it reaches the map or inverse rows.

The map rows F, I are routed through the normal-family and Schwarz rows. The normal-family row supplies the extremal compactness-style route, and the Schwarz and Schwarz–Pick rows supply the pointed disk comparison and contraction discipline used by the uniqueness read.

Finally H, C, P, L transport, replay, source, and name the same finite route. Since the carrier has no coordinate for selected boundary values, quotient function equality, ambient compact-open completion, or a global choice over domains, those objects are outside the NameCert obligations. □

Theorem 6805.3 (Riemann mapping normal-family route)**.** *Every accepted* $\mathsf{RiemannMapping}^{\uparrow}$ *consumer that reads a disk uniformization claim must pass through the ordered route*

$$\mathsf{Complex}^{\uparrow},\ \mathsf{Holomorphic}^{\uparrow},\ \mathsf{UnitDisk}^{\uparrow},\ \mathsf{NormalFamily}^{\uparrow},\ \mathsf{SchwarzLemma}^{\uparrow},\ \mathsf{SchwarzPickDiskContraction}^{\uparrow}.$$

The route first displays the proper simply connected complex-domain source, then the disk target and holomorphic conformal-map rows, then the normal-family extremal surface, and finally the Schwarz comparison rows. It exports the displayed mapping packet and inverse-map row, not a boundary extension theorem, quotient equality of maps, host analytic continuation principle, or ambient function-space completeness.

Proof. Start with the source and target rows. The complex carrier and holomorphic-map surface provide only the displayed domain observations and the map row, while the UnitDisk row fixes the codomain used by the disk-normalized route.

Next apply the normal-family route. It is the only compactness-style row in the packet, so the extremal-map read must pass through NF before any disk uniformization conclusion is named.

The Schwarz rows then control the disk comparison. $\mathsf{SchwarzLemma}^{\uparrow}$ supplies the pointed disk bound, and $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ supplies the contraction comparison, so uniqueness

and normalization are read through the finite disk route rather than an ambient analytic-function equality.

The transport, replay, provenance, and local naming rows preserve this route unchanged. Therefore a consumer that skips the normal-family row, the Schwarz rows, or the displayed boundary-refusal row is not reading an accepted $\mathsf{RiemannMapping}^{\uparrow}$ packet. □

Falsifiable prediction: A purported $\mathsf{RiemannMapping}^{\uparrow}$ consumer that claims a disk uniformization without displaying the $\mathsf{NormalFamily}^{\uparrow}$ route and the $\mathsf{SchwarzLemma}^{\uparrow}$–$\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ comparison route is outside the carrier surface.

Independence witness: holomorphic, unitdisk, complex, normal-family, schwarz-lemma, schwarz-pick-disk-contraction: none is bijective to $\mathsf{RiemannMapping}^{\uparrow}$ because only this packet binds the simply connected proper domain, disk target, conformal map, inverse row, boundary refusal, normal-family route, Schwarz comparison route, transport, replay, provenance, and local NameCert rows into one accepted carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannMapping}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannMapping}^{\uparrow}$ packet is generated from a simply connected proper complex-domain row, UnitDisk target row, holomorphic conformal-map row, inverse-map row, boundary-refusal row, NormalFamily route, SchwarzLemma and SchwarzPickDiskContraction route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannMapping}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6805.1, Definition 6805.2, and Definition 6805.3 seed the finite Riemann mapping carrier, obligation surface, and normal-family route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannMapping}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, boundary extension theorems, quotient equality of holomorphic maps, selected conformal maps outside the displayed normalization route, or ambient compact-open function-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, domain properness, simple-connectedness routing, disk target exactness, conformal-map and inverse-map exactness, boundary refusal, normal-family handoff, Schwarz comparison, transport, replay, provenance, and local naming for BEDC.Derived.RiemannMappingUp.

Chapter 6806

A Concrete Naming Certificate for $\mathsf{RealNameGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealNameGauge}^{\uparrow}$ packet records finite precision gauges for real names before a completed real object is exposed. It sits beside the dyadic tolerance ledger of chapter 291, the finite stream-window carrier of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located comparison service of chapter 1669. The packet names the tolerance-window discipline a consumer must inspect before treating a finite read as Real-facing data; it does not construct a quotient real, decide host equality, choose a completed representative, or assume ambient completeness.

Definition 6806.1 (RealNameGauge carrier)**.** *A* $\mathsf{RealNameGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, D, S, Q, H, C, P, N)$$

with the following visible rows:

1. *T is the tolerance-window ledger, listing the finite precision requests currently admitted by the gauge.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row attached to each displayed tolerance window, including the dyadic comparison and refinement data used by the request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request row through which the named real is queried at those finite tolerances.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the same stream requests with regular rational observations before any* $\mathsf{Real}^{\uparrow}$ *seal is visible.*
5. *H records componentwise* hsame *transport for tolerance windows, dyadic radii, stream requests, regular readbacks, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a tolerance request to its dyadic radius, stream read, regular rational readback, and Real-facing seal boundary.*
7. *P is the* Pkg *provenance over* $\mathsf{RealNameGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealNameGauge}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient-real equality, host equality of real names, trichotomy, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6806.2 (RealNameGauge window obligation)**.** *For every accepted* $\mathsf{RealNameGauge}^{\uparrow}$ *carrier* $G = (T, D, S, Q, H, C, P, N)$*, a gauge read must expose the monotone tolerance windows in* T*, their dyadic radii in* D*, the stream requests in* S*, and the regular rational readbacks in* Q *before any* $\mathsf{Real}^{\uparrow}$ *seal may be consumed. The exported surface is the finite gauge route through* T, D, S, Q, H, C, P, N*; it exports no quotient real, host equality, trichotomy, selected representative, countable-choice schedule, or ambient completeness.*

Proof. Project the carrier of Definition 6806.1. The tolerance windows, dyadic radii, stream requests, regular readbacks, transports, replay routes, provenance, and local naming rows are the whole visible packet.

Read the precision discipline through T and D. A later tolerance window may refine an earlier one only by displaying the dyadic-radius row attached to the same gauge request, so monotonicity is part of the finite window ledger rather than a host property of real names.

Next read S and Q. The stream request and regular rational readback are consumed before the Real-facing seal boundary, and the replay row C preserves that order under continuation.

Finally H, P, N transport, source, and name the same packet. Since the carrier has no quotient-real, host-equality, trichotomy, selected-limit, choice-schedule, or completeness coordinate, none can be exported by the accepted gauge read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealNameGauge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealNameGauge}^{\uparrow}$ packet is generated from finite tolerance windows, DyadicRatCore radius rows, StreamName request rows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealNameGauge}^{\uparrow}}$, seed)

Scope closed: Definition 6806.1 and Definition 6806.2 seed the finite precision-gauge route before Real-facing consumers read a seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealNameGauge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient reals, host equality of real names, trichotomy, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone tolerance-window exposure, dyadic radius exactness, StreamName request ordering, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealNameGaugeUp.

Chapter 6807

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusLadder}^{\uparrow}$

Following Heine–Cantor in Bishop and Bridges constructive metric analysis, the $\mathsf{CompactUniformModulusLadder}^{\uparrow}$ packet realizes the compact-uniform-continuity vision of chapter 9237 as finite NameCert data. It sits between the compact source rows of chapter 60, the metric distance rows of chapter 57, the pointwise continuity rows of chapter 59, the tolerance-only consumer row of chapter 369, and the finite ProbeBundle cover used by the compact witness. The packet records the ladder from pointwise moduli and finite cover rows to a uniform modulus row; it does not import an open-cover theorem, host compactness, choice-selected centers, semantic continuity, or ambient completeness.

Definition 6807.1 (CompactUniformModulusLadder carrier)**.** *A* $\mathsf{CompactUniformModulusLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, B, R, U, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row, exposing only the finite cover data accepted by the compact certificate.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row shared by the compact source and the continuous map.*
3. *B is the finite* ProbeBundle *cover row, listing displayed centers, membership rows, cover-radius rows, and coverage witnesses.*
4. *R is the pointwise-radius row read through* $\mathsf{Continuous}^{\uparrow}$*, assigning local modulus evidence to each displayed cover center.*
5. *U is the tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *row produced from the cover and pointwise-radius rows by a finite lower-bound fold.*
6. *H records componentwise* hsame *transport for compact rows, metric rows, bundle rows, pointwise-radius rows, uniform-modulus rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a tolerance request to finite cover inspection, pointwise-radius inspection, lower-bound folding, and the resulting uniform-modulus read.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusLadder}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusLadder}^{\uparrow}}$ *row.*
The carrier contains no coordinate for an ambient open-cover compactness principle, host compactness object, choice-selected center, semantic continuity theorem, quotient metric equality, or completeness of the target metric.

Theorem 6807.2 (CompactUniformModulusLadder certificate)**.** *For every accepted* CompactUniformModulusLadd *carrier $L = (K, M, B, R, U, H, C, P, N)$, a tolerance-only uniform-modulus read must factor through the compact source row K, the metric row M, the finite* ProbeBundle *cover row B, and the pointwise-radius row R before the* $\mathsf{UniformModulus}^{\uparrow}$ *row U is exposed. The certificate is the finite ladder $K, M, B, R, U, H, C, P, N$, not an open-cover compactness theorem, host-selected finite subcover, external modulus function, semantic continuity proof, quotient metric equality, or completeness argument.*

Proof. Project the carrier of Definition 6807.1. The compact source, metric row, finite bundle, pointwise-radius row, uniform-modulus row, transport, replay, provenance, and naming rows are the only coordinates available.

Read K, M, B first. The compact input is visible only through the finite ProbeBundle cover row over the shared metric, so the ladder starts from displayed cover data rather than from a host open-cover principle or an externally selected subcover.

Next read R. Each displayed center carries pointwise-radius evidence through $\mathsf{Continuous}^{\uparrow}$, and the lower-bound fold uses only the finitely many rows already listed in B and R.

Finally expose U through H, C, P, N. The replay route sends a tolerance request to the same finite cover, pointwise rows, and fold before the uniform-modulus row is consumed. Since no excluded object is a coordinate of L, none can be exported from the accepted certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusLadder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformModulusLadder}^{\uparrow}$ packet is generated from CompactUp source rows, MetricUp distance rows, finite ProbeBundle cover rows, ContinuousUp pointwise-radius rows, a UniformModulusUp tolerance-only row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformModulusLadder}^{\uparrow}}$, seed)

Scope closed: Definition 6807.1 and Definition 6807.2 seed the finite ladder from compact cover rows and pointwise moduli to a tolerance-only uniform-modulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformModulusLadder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host compactness, choice-selected centers, semantic continuity, quotient metric equality, target completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover exposure, pointwise-radius attachment, lower-bound fold exactness, UniformModulusUp handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusLadderUp.

Chapter 6808

A Concrete Naming Certificate for $\mathsf{TaylorSeries}^{\uparrow}$

Following Taylor's finite expansion method and Bishop–Bridges constructive analysis, the $\mathsf{TaylorSeries}^{\uparrow}$ packet records a finite Taylor-jet surface before any analytic convergence claim is read. It sits after the local-linear approximation route of chapter 3632, the displayed derivative rows of chapter 122, the finite polynomial carrier of chapter 50, and the L10 real-name substrate of chapter 38, chapter 301, and chapter 291. The packet names a finite jet, its polynomial readback, and its tolerance ledger; it does not assert analytic convergence, a remainder theorem, a radius of convergence, host smoothness, quotient equality of functions, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6808.1 (TaylorSeries carrier)**.** *A* $\mathsf{TaylorSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, x, k, J, P, R, D, H, C, G, N)$$

with the following visible rows:

1. *A is the* $\mathsf{Calculus}^{\uparrow}$ *local approximation row naming the source function only through finite observation histories.*
2. *x is the* $\mathsf{Real}^{\uparrow}$ *center row at which the jet is read.*
3. *k is the finite order budget for the displayed jet.*
4. *J is the finite list of derivative-read rows, each routed through* $\mathsf{Derivative}^{\uparrow}$ *and indexed by an order at most k.*
5. *P is the* $\mathsf{Poly}^{\uparrow}$ *readback row whose coefficients are the displayed jet entries.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row used when coefficients or center data are read as regular rational names.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for finite factorial, power, and tolerance accounting.*
8. *H records componentwise* hsame *transport for the center, order, derivative rows, polynomial readback, regular-rational row, dyadic ledger, route, provenance, and local name.*
9. *C records the displayed* Cont *replay from finite calculus observations to the derivative rows, polynomial row, tolerance ledger, provenance, and local name.*
10. *G is the* Pkg *provenance over* $\mathsf{TaylorSeries}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{TaylorSeries}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite series convergence, analytic continuation, a radius-of-convergence theorem, a remainder estimate beyond the displayed finite order, host smoothness, quotient equality of functions, or ambient real completeness.

Theorem 6808.2 (TaylorSeries NameCert obligations)**.** *For an accepted* $\mathsf{TaylorSeries}^{\uparrow}$ *carrier* $T = (A, x, k, J, P, R, D, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TaylorSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite center, order, derivative rows, polynomial readback, regular-rational tolerance row, dyadic arithmetic ledger, transports, replay route, provenance, and local name; derivative-to-polynomial exactness up to the displayed order k; finite coefficient stability under componentwise* hsame *transport; and ledger non-escape excluding analytic convergence, remainder bounds beyond k, host smoothness, quotient function equality, selected infinite sums, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6808.1. The tuple displays exactly the finite center row, order budget, derivative rows, polynomial readback, regular-rational tolerance row, dyadic arithmetic ledger, transport rows, replay route, provenance, and local name.

Derivative-to-polynomial exactness is also finite. The row J lists only derivative reads of order at most k, and the row P reads those entries as coefficients of the finite polynomial surface rather than as an infinite expansion.

Stability follows from H and C. Componentwise hsame transport preserves the displayed center, order, coefficient, and tolerance rows, while replay through C keeps the calculus, derivative, polynomial, regular-rational, and dyadic rows in the same order.

No remaining carrier coordinate names convergence, a remainder theorem, a radius of convergence, host smoothness, quotient equality of functions, a selected infinite sum, or real completeness. The local certificate is therefore exhausted by the finite Taylor-jet route. □

Theorem 6808.3 (TaylorSeries finite-jet non-escape)**.** *Every consumer-facing read of an accepted* $\mathsf{TaylorSeries}^{\uparrow}$ *packet factors through the finite rows*

$$A, x, k, J, P, R, D, H, C, G, N.$$

Consequently the packet exposes only a finite Taylor jet and its polynomial readback, not an infinite Taylor series convergence theorem, analytic continuation, a radius-of-convergence object, a host smoothness principle, quotient equality of functions, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier rows of Definition 6808.1. A consumer first reads the finite calculus source row A, the Real center x, and the order budget k.

The derivative data are then confined to J. Since J is indexed only by orders at most k, the polynomial row P can read exactly those entries as finite coefficients and cannot expose an infinite tail.

The tolerance and arithmetic support rows R and D account for regular-rational and dyadic coefficient reads. Transport, replay, provenance, and local naming through H, C, G, N preserve that finite route.

Because the carrier lists no convergence, analytic-continuation, radius, host-smoothness, quotient-function, selected-infinite-sum, or real-completeness coordinate, finite route exhaustion prevents any such principle from escaping the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorSeries}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6808.1, Definition 6808.2, and Definition 6808.3 seed the finite Taylor-jet packet over calculus observations, derivative rows, polynomial readback, Real center data, regular-rational tolerances, and dyadic arithmetic ledgers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaylorSeries}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic convergence, Taylor remainder theorems, a radius of convergence, host smoothness, quotient equality of functions, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-polynomial exactness, finite coefficient stability, and finite-jet non-escape for BEDC.Derived.TaylorSeriesUp.

Constructive story: A TaylorSeries$^{\uparrow}$ packet is generated from a finite calculus observation row, a Real center, a finite order budget, displayed derivative rows, a polynomial readback, regular-rational tolerance data, dyadic arithmetic ledgers, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 6809

A Concrete Naming Certificate for $\mathsf{RealCompletionModulusEnvelope}^{\uparrow}$

The $\mathsf{RealCompletionModulusEnvelope}^{\uparrow}$ packet realizes the finite modulus handoff isolated by the real-completeness vision of chapter 9238. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Real-induced uniform Cauchy-filter completion route of chapter 6564. The packet gives later completion consumers a named envelope for regular-Cauchy modulus data without asserting ambient $\mathsf{Real}^{\uparrow}$ completeness, selecting limits, quotienting streams, or importing countable choice.

Definition 6809.1 (RealCompletionModulusEnvelope carrier)**.** *A* $\mathsf{RealCompletionModulusEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, Q, D, R, U, \mu, W, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the observed regular-Cauchy input by finite stream histories.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational sequence values.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare displayed windows and precision requests.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, Q, D have been replayed.*
5. *U is the* $\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}$ *handoff row recording the completion-facing uniform-Cauchy route as finite evidence.*
6. *μ is the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, read only as a finite family of threshold requests used by the packet.*
7. *W is the finite window ledger pairing each inspected tolerance with the stream and regular-rational rows it controls.*
8. *H records componentwise* hsame *transport for the stream, regular-rational, dyadic, Real, uniform-completion, modulus, window, route, provenance, and local name rows.*
9. *C records the displayed* Cont *replay from stream observations to regular-rational readback, dyadic tolerance, Real seal, uniform-completion handoff, modulus window, provenance, and local name.*

10. *P is the* Pkg *provenance over* $\mathsf{RealCompletionModulusEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionModulusEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for ambient Real completeness, a selected limit, quotient stream equality, countable choice, a host completion object, or an unbounded modulus oracle.

Theorem 6809.2 (RealCompletionModulusEnvelope NameCert obligations)**.** *For an accepted* $\mathsf{RealCompletionModulusEnvelope}^{\uparrow}$ *carrier* $M = (S, Q, D, R, U, \mu, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionModul}}$ *surface consists exactly of carrier admission for the stream, regular-rational, dyadic, Real, uniform-completion, modulus, finite-window, transport, replay, provenance, and local name rows; modulus-window exactness for the displayed finite tolerance requests; Real-seal handoff after the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows have been replayed; uniform-completion compatibility through* U*; and ledger non-escape excluding ambient completeness, selected limits, quotient stream equality, countable choice, host completion, and unbounded modulus oracles.*

Proof. Carrier admission is projection from Definition 6809.1. The tuple displays exactly the stream window, regular-rational readback, dyadic tolerance, Real seal, uniform-completion handoff, modulus schedule, finite window ledger, transport, replay, provenance, and local name rows.

The modulus-window exactness is finite. The row μ is read only through the window ledger W, and each inspected tolerance request is paired with the visible S, Q, D rows before the Real seal R is available.

Uniform-completion compatibility is the handoff row U. It records the same displayed regular-Cauchy route consumed by the Real-induced uniform Cauchy-filter completion packet, but it does not add a selected limit or an ambient complete carrier.

Finally, H, C, P, N transport, replay, package, and name only the displayed route. Since no carrier coordinate names ambient completeness, quotient stream equality, countable choice, a host completion object, a selected limit, or an unbounded modulus oracle, none of those principles can be exported by the local certificate. □

Falsifiable prediction: Every accepted RealCompletionModulusEnvelopeUp read exposes StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, a RealUniformCauchyFilterCompletion handoff, a CauchyModulus finite schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any completion-facing consumer can inspect the packet.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-uniform-cauchy-filter-completion: removing any one of these sibling routes leaves the envelope unable to display the corresponding stream window, regular-rational readback, dyadic tolerance, Real seal, or uniform-completion handoff row.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionModulusEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionModulusEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6809.1 and Definition 6809.2 seed the finite regular-Cauchy modulus envelope over stream windows, regular-rational readback, dyadic tolerances, Real seals, uniform-completion handoff, and finite modulus schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionModulusEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, countable choice, host completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite modulus-window exactness, Real-seal handoff, uniform-completion compatibility, and ledger non-escape for BEDC.Derived.RealCompletionModulusEnvelopeUp.

Constructive story: A RealCompletionModulusEnvelope$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledgers, a RealUp seal, a RealUniformCauchyFilterCompletion handoff, a finite CauchyModulus schedule, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 6810

A Concrete Naming Certificate for $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$

Following Aronszajn–Panitchpakdi and Isbell on injective metric envelopes, the $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ packet records the finite NameCert surface by which a displayed metric source is admitted to an injective-envelope metric boundary. It sits over the metric source of chapter 57, reads the Isbell tight-span carrier of chapter 2361, exposes the hyperconvex handoff of chapter 4013, transports the source through the metric embedding route of chapter 3339, keeps the completion boundary of chapter 128 visible, and seals all radius comparisons through chapter 38. The packet names a finite envelope admission route; it does not select a host hull point, assert arbitrary-family intersection, identify quotient completions, invoke countable choice, or close ambient real completeness.

Definition 6810.1 (Injective envelope metric carrier)**.** *An* $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, I, J, HX, K, R, T, U, V, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the inspected point classifier and distance observations.*
2. *I is the* $\mathsf{IsbellTightSpan}^{\uparrow}$ *envelope row, read through finite distance-envelope inequalities, extremal witnesses, one-point extension tests, separated comparison, and Real readback.*
3. *J is the* $\mathsf{MetricEmbedding}^{\uparrow}$ *row, recording the displayed isometric source inclusion into the envelope boundary.*
4. *HX is the* $\mathsf{HyperconvexMetric}^{\uparrow}$ *handoff row, exposing pairwise-compatible ball data and the finite intersection request only after M, I, J have been read.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion-boundary row, keeping completion-neighbour use downstream of the metric source, envelope row, embedding row, and hyperconvex handoff.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *radius seal row for all distance, envelope, ball, and separated-comparison readings.*
7. *T records componentwise* hsame *transport for M, I, J, HX, K, R, P, N and for the replay row.*
8. *U records displayed* Cont *replay from metric observations to tight-span envelopes, embedding transport, hyperconvex ball handoff, completion boundary, and Real radius seal.*
9. *V is the provenance route through the sibling certificates named above, including the local source row that prevents the envelope from being reread as an ungrounded host hull.*

10. *P is the* Pkg *provenance over* $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{IsbellTightSpan}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, $\mathsf{HyperconvexMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{InjectiveEnvelopeMetric}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $M, I, J, HX, K, R, T, U, V, P, N$. It has no coordinate for a selected injective hull, arbitrary ball-family intersection, quotient completion representative, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 6810.2 (Injective envelope metric NameCert obligations)**.** *For every accepted* $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ *carrier $E = (M, I, J, HX, K, R, T, U, V, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InjectiveEnvelopeMetric}^{\uparrow}}$ *surface consists exactly of metric-source exposure, Isbell tight-span envelope admission, metric-embedding transport, hyperconvex handoff, completion-boundary visibility, Real radius sealing, componentwise transport, displayed replay, sibling provenance, package provenance, and local naming.*

Proof. Project the carrier of Definition 6810.1. The rows $M, I, J, HX, K, R, T, U, V, P, N$ are the only displayed data admitted by the packet.

Use the metric source chapter for M, the Isbell tight-span obligations for I, and the metric embedding route for J. These rows fix the source, envelope admission, and isometric inclusion before any completion-facing read is available.

Use the hyperconvex metric obligations for HX and the complete metric boundary for K. The hyperconvex row can be consumed only as a finite handoff over the same source and envelope data, while the completion row remains downstream of that handoff.

Finally read the Real radius seal and the structural rows T, U, V, P, N. They transport, replay, source, package, and name the same finite route. Since no remaining coordinate stores a selected hull point, arbitrary-family intersection, quotient completion representative, choice selector, or ambient completeness witness, the listed rows exhaust the local NameCert surface. □

Theorem 6810.3 (Injective envelope metric hyperconvex handoff)**.** *Every completion-neighbour or Real-radius consumer of an accepted* $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ *carrier must read*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{IsbellTightSpan}^{\uparrow} \longmapsto \mathsf{MetricEmbedding}^{\uparrow} \longmapsto \mathsf{HyperconvexMetric}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows M, I, J, HX, K, R before it treats the packet as an injective-envelope metric boundary.

Proof. Start with the carrier projection. The only route to the hyperconvex handoff row HX passes through the metric source M, the Isbell envelope row I, and the metric embedding row J.

Apply the Isbell tight-span metric handoff. It keeps finite envelope inequalities, extremal witnesses, one-point extension tests, separated comparison, completion readback, and Real readback visible before an envelope consumer can proceed.

Apply the hyperconvex metric ball-intersection handoff. It reads pairwise-compatible ball data and the finite intersection request over the same metric and Real radius rows; it does not introduce a selected global hull point or an arbitrary-family intersection coordinate.

The complete metric row K and Real radius row R are downstream of those sibling handoffs. Transport, replay, provenance, and local naming preserve that route without adding a quotient completion representative, choice function, or ambient completeness principle. Thus the injective-envelope consumer receives exactly the displayed finite handoff. □

Falsifiable prediction: Every accepted $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ consumer that reaches a hyperconvex, completion, or Real-radius read without first exposing the displayed metric source, Isbell tight-span envelope, metric embedding, and sibling hyperconvex handoff refutes the carrier surface.

Independence witness: metric, isbell-tight-span, hyperconvex-metric, metricembedding, completemetric, real: no listed sibling supplies the whole eleven-row packet $E = (M, I, J, HX, K, R, T, U, V, P, N)$, because each sibling contributes only a source, envelope, embedding, ball-intersection handoff, completion boundary, or Real seal while this chapter binds all six faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{InjectiveEnvelopeMetric}^{\uparrow}}$.

Constructive story: An $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ packet is generated from a metric source row, an Isbell tight-span envelope row, a metric embedding row, a hyperconvex handoff row, a complete-metric boundary row, a Real radius seal row, componentwise transports, displayed replay, sibling provenance, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InjectiveEnvelopeMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6810.1, Definition 6810.2, and Definition 6810.3 seed the finite injective-envelope metric admission route over the displayed metric, Isbell, embedding, hyperconvex, completion, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InjectiveEnvelopeMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected injective hulls, arbitrary ball-family intersections, quotient completion representatives, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires carrier admission, sibling envelope routing, metric-embedding transport, hyperconvex handoff, completion-boundary visibility, and Real radius sealing for the same packet.

Chapter 6811

A Concrete Naming Certificate for $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$

Following Bishop-style error accounting for finite approximations and the reality-constrained signature discipline of chapter 9414, the $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$ packet records the bounded mismatch budget attached to a physical-signature classifier. It sits beside the dyadic tolerance surface of chapter 1761, the model-facing budget of chapter 1184, the signature-fit route of **??**, and the tower packet of chapter 1128. The packet names finite classifier, dyadic tolerance, tower-fit, mismatch, and ledger rows; it does not assert physical truth, final model selection, unbounded validation, probabilistic calibration, selected hidden states, or ambient real completeness.

Definition 6811.1 (RealitySignatureErrorBudget carrier)**.** *A* $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, C, D, M, T, F, E, L, H, K, P, N)$$

with the following displayed rows:

1. *S is the physical-signature bundle row naming the finite reality-facing observations currently under inspection.*
2. *C is the signature classifier row that compares observed and model-generated signatures only through the displayed bundle S.*
3. *D is the* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *tolerance row assigned to the classifier comparison.*
4. *M is the* $\mathsf{RealityConstrainedModelErrorBudget}^{\uparrow}$ *row recording the model-side mismatch envelope.*
5. *T is the tower-fit row, read through* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *and the approximation-tower source of chapter 9414.*
6. *F is the finite fit ledger that records which signature comparisons remain within D after the model budget M and tower row T have been displayed.*
7. *E is the visible error-surface row for comparisons that leave the declared budget.*
8. *L is the audit ledger tying S, C, D, M, T, F, E to the same observation window and physical-signature source.*
9. *H records componentwise* hsame *transport for the signature, classifier, tolerance, model-budget, tower-fit, fit-ledger, error-surface, audit, replay, provenance, and local naming rows.*

10. *K records displayed* Cont *replay from signature observations through the classifier and budget rows to the fit ledger and error surface.*
11. *P is the* Pkg *provenance over* $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$, $\mathsf{DyadicErrorBudget}^{\uparrow}$, $\mathsf{RealityConstrainedModelErrorBu}$ $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealitySignatureErrorBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for physical truth, final model selection, unbounded validation, probabilistic calibration, hidden-state choice, host real equality, or ambient completeness.

Theorem 6811.2 (RealitySignatureErrorBudget NameCert obligations)**.** *For every accepted* RealitySignatureErrorB *carrier $R = (S, C, D, M, T, F, E, L, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealitySignatureErrorBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the physical-signature bundle, signature classifier, dyadic tolerance, model-error budget, tower-fit row, fit ledger, error surface, audit ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. The fit ledger F may be consumed only after S, C, D, M, T are displayed, and the error surface E remains visible whenever the classifier leaves the declared budget.*

Proof. Project the carrier of Definition 6811.1. The accepted packet exposes the physical-signature bundle, classifier, dyadic tolerance, model-budget, tower-fit row, fit ledger, error surface, audit ledger, and structural rows.

The source rows come first. The classifier C is tied to the displayed signature bundle S, while D and M provide the tolerance and model-side mismatch envelope before any fit ledger is read.

The tower row T then anchors the comparison in the approximation-tower route. The fit ledger F records only comparisons that pass through S, C, D, M, T, and the error surface E records the visible boundary when the declared budget is exceeded.

Finally L, H, K, P, N preserve the same finite route under audit, transport, replay, provenance, and local naming. Since no other carrier coordinate is present, the local certificate cannot export physical truth, final model selection, unbounded validation, probabilistic calibration, hidden-state choice, host real equality, or ambient completeness. □

Theorem 6811.3 (RealitySignatureErrorBudget tower-fit handoff)**.** *Every* $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$ *consumer that reads a physical-signature budget must factor through*

$$\mathsf{RealityConstrainedSignatureFit}^{\uparrow},\quad \mathsf{DyadicErrorBudget}^{\uparrow},\quad \mathsf{RealityConstrainedModelErrorBudget}^{\uparrow},\quad \mathsf{RealityConstraine}$$

The handoff exports only the displayed signature bundle, classifier, dyadic tolerance, model-budget envelope, tower-fit row, fit ledger, error surface, audit ledger, transport, replay, provenance, and local naming. It exports no final physical-truth oracle, hidden-state selector, unbounded validation theorem, probabilistic calibration, host real equality, or ambient completeness claim.

Proof. Start with the NameCert obligations of Definition 6811.2. They force the source signature bundle, classifier, dyadic tolerance, model-budget row, and tower-fit row to be displayed before the fit ledger is read.

Read the sibling routes in order. The signature-fit packet supplies the classifier boundary, the dyadic error-budget packet supplies the finite tolerance accounting, the model-error budget supplies the mismatch envelope, and the tower packet supplies the approximation-level fit row.

The audit, transport, replay, provenance, and local naming rows keep those sibling routes attached to one accepted packet. The carrier contains no coordinate for the excluded physical,

probabilistic, hidden-state, host-equality, or completeness claims, so the handoff exports only the finite budget surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealitySignatureErrorBudget}\uparrow}$.

Constructive story: A $\mathsf{RealitySignatureErrorBudget}^{\uparrow}$ packet is generated from a physical-signature bundle, signature classifier, DyadicErrorBudgetUp tolerance row, RealityConstrainedModelErrorBudgetUp mismatch envelope, RealityConstrainedTowerPacketUp tower-fit row, fit ledger, error surface, audit ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealitySignatureErrorBudget}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6811.1, Definition 6811.2, and Definition 6811.3 seed the finite physical-signature error-budget route through signature-fit, dyadic tolerance, model-error, tower-fit, ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealitySignatureErrorBudget}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim physical truth, final model selection, unbounded validation, probabilistic calibration, hidden-state choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signature classifier exactness, dyadic tolerance routing, model-budget compatibility, tower-fit stability, fit-ledger visibility, error-surface non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RealitySignatureErrorBudgetUp.

Chapter 6812

A Concrete Naming Certificate for $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$

Following Riemann, Caratheodory, and Bishop-style constructive complex analysis, the ConstructiveRiemannMappin packet records the finite NameCert surface needed to read a displayed simply connected planar domain through a normalized disk map route. It sits over the holomorphic source of chapter 68, the complex carrier of chapter 39, the complex-topology disk and compact-window rows of chapter 77, the normal-family surface of chapter 3693, the pointed disk control of chapter 3796, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed approximation, normalization, and nonescape rows; it does not assert host open-cover compactness, select a conformal map, identify quotient function equality, invoke countable choice, or import ambient $\mathsf{Complex}^{\uparrow}$ completeness.

Definition 6812.1 (Constructive Riemann mapping carrier)**.** *A* $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, C, T, H, M, S, W, Q, E, N, Z, P, L)$$

with the following displayed rows:

1. *D is the domain row, naming a connected simply connected planar source through* $\mathsf{CplxTopo}^{\uparrow}$ *open-disk, compact-window, and boundary-separation data.*
2. *C is the* $\mathsf{Complex}^{\uparrow}$ *coordinate row for the displayed source point, disk center, boundary witnesses, and target disk coordinates.*
3. *T is the* $\mathsf{Holomorphic}^{\uparrow}$ *trial-map family row over the source and target disk surfaces.*
4. *H is the normalization row, recording the selected base point, disk origin target, positive derivative-orientation ledger, and boundary separation budget as displayed finite data.*
5. *M is the* $\mathsf{MontelTheorem}^{\uparrow}$ *normal-family row consumed only after the compact-window and local-boundedness data in D, T, H have been exposed.*
6. *S is the* $\mathsf{SchwarzLemma}^{\uparrow}$ *pointed-disk control row, used to compare any two normalized candidates through the same displayed disk route.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule for trial maps, compact windows, and boundary tests.*

8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the real and complex coordinate windows named by W.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for radii, separation budgets, and modulus comparisons.*
10. *N is the nonescape row excluding host open-cover compactness, selected conformal maps, quotient function equality, countable choice, and ambient* $\mathsf{Complex}^{\uparrow}$ *completeness from the packet.*
11. *Z records componentwise* hsame *transport for all displayed rows.*
12. *P records displayed* Cont *replay and* Pkg *provenance over* $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{CplxTopo}^{\uparrow}$, $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{SchwarzLemma}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *L is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannMapping}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, C, T, H, M, S, W, Q, E, N, Z, P, L$. It has no coordinate for a host-chosen conformal equivalence, quotient equality of holomorphic functions, arbitrary open-cover compactness, countable-choice selection, or completed ambient complex function space.

Theorem 6812.2 (Constructive Riemann mapping NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ *carrier*

$$R = (D, C, T, H, M, S, W, Q, E, N, Z, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannMapping}^{\uparrow}}$ *surface consists exactly of domain admission through D, complex-coordinate admission through C, holomorphic trial-family exposure through T, normalization through H, normal-family compact-window routing through M, pointed-disk comparison through S, finite observation through W, regular rational readback through Q, Real seal admission through E, nonescape through N, componentwise transport through Z, displayed replay and provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 6812.1. The domain, complex-coordinate, holomorphic-family, normalization, normal-family, Schwarz-control, observation, readback, seal, nonescape, transport, provenance, and local-name rows are the complete visible packet.

The analytic route starts with D, C, T, H. These rows expose the source domain, the complex coordinates, the trial holomorphic family, and the normalization ledger before any normal-family or disk comparison read is available.

Next read M, S, W, Q, E. The normal-family row is tied to the displayed compact windows and local boundedness data, while the Schwarz row compares normalized candidates only through the same pointed disk route. The stream, regular rational, and Real rows keep the observation and modulus data finite.

Finally N, Z, P, L preserve the boundary. Since the packet has no coordinate for host open-cover compactness, selected maps, quotient function equality, countable choice, or ambient complex completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6812.3 (Constructive Riemann mapping normal-family route)**.** *Every normal-family read exported by an accepted* $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ *packet factors through the displayed route*

$$D, C, T, H \longmapsto M \longmapsto S \longmapsto W, Q, E \longmapsto N, Z, P, L.$$

Thus a consumer may inspect compact-window normality, normalized holomorphic candidates, pointed disk comparison, finite observation windows, regular rational readbacks, Real seals, transport, replay, provenance, and local naming. It cannot read the packet as an ambient existence theorem for a selected conformal map.

Proof. Begin with the source rows. The domain row D and complex coordinate row C supply the displayed planar source, target disk coordinates, compact windows, and boundary-separation data.

The trial family and normalization rows T, H then provide the only admissible input to M. The $\mathsf{MontelTheorem}^{\uparrow}$ row can be consumed only through these displayed compact windows and local boundedness ledgers, so the route is normal-family data rather than open-cover compactness.

Use S after M. The $\mathsf{SchwarzLemma}^{\uparrow}$ row compares normalized candidates through the pointed disk route and prevents the handoff from becoming an unconstrained function-space equality.

The observation rows W, Q, E keep every approximation and modulus comparison inside finite StreamName, RegSeqRat, and Real seals. The structural rows N, Z, P, L then transport, replay, source, and name the same route without adding a selected conformal map. □

Theorem 6812.4 (Constructive Riemann mapping Real and Complex seal nonescape)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Complex}^{\uparrow}$ *read exported by an accepted* $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ *packet remains inside the displayed rows* C, W, Q, E, Z, P, L *and is scoped by the domain, trial-family, normalization, normal-family, Schwarz-control, and nonescape rows* D, T, H, M, S, N*. No consumer may reinterpret those seals as host open-cover compactness, a selected conformal map, quotient equality of holomorphic functions, countable choice, or ambient* $\mathsf{Complex}^{\uparrow}$ *completeness.*

Proof. Project the carrier and isolate the seal rows. The complex coordinate row C, stream schedule W, regular rational readback Q, and Real seal E are displayed coordinates of the same packet, with transport, replay, provenance, and local naming stored in Z, P, L.

The analytic rows D, T, H, M, S scope those seals. A Real or Complex read reaches the source domain, trial family, normalization, normal-family compact-window route, and pointed disk comparison before it can use any sealed coordinate.

The nonescape row N records the excluded endpoints as absent carrier coordinates. Therefore a sealed real or complex observation remains finite BEDC data and cannot be retyped as open-cover compactness, a chosen conformal equivalence, quotient function equality, countable choice, or ambient complex completeness. □

Falsifiable prediction: If a purported $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ packet exports a normalized disk map without displaying the MontelTheoremUp normal-family row and the SchwarzLemmaUp pointed comparison row, it is not a $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ carrier.

Independence witness: holomorphic, complex, complex-topology, montel-theorem, schwarz-lemma, streamname, regseqrat, real: no listed sibling supplies the joint carrier containing displayed domain rows, holomorphic trial-family rows, normalization, normal-family routing, pointed disk comparison, finite observation, regular rational readback, Real and Complex seals, nonescape, transport, replay, provenance, and local NameCert data while refusing host open-cover compactness, selected conformal maps, quotient function equality, countable choice, and ambient Complex completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannMapping}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveRiemannMapping}^{\uparrow}$ packet is generated from a ComplexTopology domain row, Complex coordinate rows, Holomorphic trial-map family rows, a normalization ledger, MontelTheorem normal-family

routing, SchwarzLemma pointed-disk comparison, StreamName finite observation windows, RegSeqRat readback, Real seals, nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveRiemannMapping}^{\uparrow}}$, seed)

Scope closed: Definition 6812.1, Definition 6812.2, Definition 6812.3, and Definition 6812.4 seed the finite normal-family and seal-nonescape route for constructive Riemann mapping data.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveRiemannMapping}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, selected conformal maps, quotient function equality, countable choice, ambient Complex completeness, Lean verification, or bridge checking for ConstructiveRiemannMapping$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, domain and complex-coordinate exposure, holomorphic trial-family routing, normalization, normal-family exactness, pointed disk comparison, finite observation, regular rational readback, Real and Complex seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveRiemannM

Chapter 6813

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$ packet records the L10 completion carrier whose points are read through regular Cauchy rational data rather than through an ambient quotient-completion theorem. It uses the dyadic tolerance arithmetic of chapter 291, the finite observation schedules of chapter 126, the regular rational tail surface of chapter 301, and the terminal sealing discipline of chapter 38. The chapter names the space-level carrier sitting behind the route and functor fragments already visible in the Cauchy and completion dossiers.

Definition 6813.1 (Regular Cauchy completion space carrier)**.** *A* $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, Q, W, D, R, L, E, H, C, P, N)$$

with the following roadmap rows:

1. *Traditional target. X names the Bishop–Bridges regular Cauchy completion space whose completed points are represented by regular rational Cauchy data.*
2. *BEDC source. Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row and W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row through which every finite precision read is opened.*
3. *Carrier. D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational tail has been inspected.*
4. *Classifier. The local classifier compares accepted packets only through $X, Q, W, D, R, L, E, H, C, P, N$ and never through host quotient equality or selected limit representatives.*
5. *Stability obligations. L records tail stability for later windows preserving the same regular rational source, dyadic tolerance ledger, and Real seal route.*
6. *Ledger. E records the completion ledger row: every completed point read factors through the displayed Q, W, D, R, L rows before a space-level consumer sees X.*
7. *Core theorem list. The local seed surface is Definition 6813.2 and Definition 6813.3.*
8. *Exactness predicate. Exactness means that the space-level completion read is precisely the displayed route from regular Cauchy rational tails through finite windows, dyadic tolerances, and the Real seal.*

9. *Standard bridge status. No standard bridge is asserted in this chapter; the packet is a paper-side NameCert carrier awaiting a bridge theorem.*

10. *Status. H supplies componentwise* hsame *transport, C supplies displayed* Cont *replay, P supplies* Pkg *provenance over the listed sibling certificates, and N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, a selected limit, quotient stream equality, host metric completion, trichotomy, or an ambient completeness principle.

Theorem 6813.2 (Regular Cauchy completion space NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$ *packet*

$$S = (X, Q, W, D, R, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSpace}^{\uparrow}}$ *surface consists exactly of the space row X, regular rational source row Q, finite* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, terminal* $\mathsf{Real}^{\uparrow}$ *seal row R, tail-stability row L, completion ledger row E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N.*

Proof. Project the carrier of Definition 6813.1. Its analytic rows are X, Q, W, D, R, L, E, and its structural rows are H, C, P, N.

The source row Q and window row W expose the regular rational tail at a finite precision. The dyadic row D supplies tolerance data for that read, and the Real seal row R is reached only after those finite rows have been consumed.

Rows L and E keep the tail-stability and completion-ledger obligations attached to the same packet. Since H, C, P, N only transport, replay, source, and name these rows, the displayed list exhausts the local NameCert surface. □

Theorem 6813.3 (Regular Cauchy completion space L10 handoff)**.** *Every L10 completion consumer of an accepted* $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$ *carrier reads the completed point through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionSpace}^{\uparrow}.$$

The handoff exports no selected limit, quotient stream equality, countable-choice scheduler, host metric completion, or ambient Real completeness theorem.

Proof. Apply Definition 6813.2. The rows Q, W, D, R, L, E place the regular rational source, finite window, dyadic tolerance ledger, Real seal, tail-stability witness, and completion ledger in one displayed packet.

The structural rows H, C, P, N preserve that same route under componentwise transport, replay, provenance, and local naming. They do not add a second path from a host completion object to the completed point.

Thus a consumer reaches the space-level completion carrier only after the regular Cauchy tail has passed through the finite observation and Real-seal route. The refused readings would need coordinates absent from S. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyCompletionSpace}^{\uparrow}$ packet is generated from a space row, RegSeqRat source, StreamName window, DyadicRatCore tolerance ledger, Real seal, tail-stability row, completion ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionSpace}\uparrow}$, seed)

Scope closed: Definition 6813.1, Definition 6813.2, and Definition 6813.3 seed the space-level L10 regular Cauchy completion carrier and its finite read route.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host metric completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window stability, tolerance exactness, Real-seal handoff, completion-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionSpaceUp.

Chapter 6814

A Concrete Naming Certificate for $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$

Following Gagliardo and Nirenberg on interpolation inequalities, the $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which a Sobolev derivative window, a norm ledger, and a metric or complete-metric source window hand an interpolation estimate to compactness-facing consumers. It sits after the $\mathsf{Sobolev}^{\uparrow}$ carrier of chapter 440, the norm surface of chapter 89, the compact-metric route of chapter 129, and the regular rational readback of chapter 301. The packet names only finite rows: derivative windows, exponent comparisons, norm estimates, compact handoff data, transport, replay, provenance, and local $\mathsf{NameCert}$ evidence.

Definition 6814.1 (Gagliardo-Nirenberg finite-window carrier)**.** *A* $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, N, M, D, E, R, K, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, carrying the displayed function, weak-derivative budget, integrability row, and local domain window.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *target row, recording the lower-order and higher-order norm reads used by the interpolation estimate.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row whose finite balls and Cauchy-boundary data supply the inspection scale.*
4. *D is the finite derivative-window row, listing exactly the derivative orders, test windows, and source tolerances inspected by the estimate.*
5. *E is the interpolation exponent ledger, recording the dimension, derivative orders, norm exponents, interpolation weight, and admissibility comparisons as finite rational or regular-rational data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for tolerance, exponent, and norm-bound observations.*
7. *K is the compact handoff row, exposing only the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact-window data needed by L10/L11 compactness consumers.*
8. *H records componentwise* hsame *transport for the Sobolev, norm, metric, derivative-window, exponent-ledger, readback, compact-handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from requested precision to derivative windows, norm ledgers, regular-rational readback, and compact handoff rows.*
10. *P is the* Pkg *provenance over* $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}}$ *evidence.*

The classifier compares accepted packets only through $S, N, M, D, E, R, K, H, C, P, L$. It has no coordinate for a host Sobolev embedding theorem, nonconstructive compact embedding, quotient function equality, selected representative, countable-choice schedule, or ambient completed function space.

Theorem 6814.2 (Gagliardo-Nirenberg finite-window NameCert obligations)**.** *For every accepted* $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ *carrier*

$$G = (S, N, M, D, E, R, K, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Sobolev}^{\uparrow}$ *source row, norm-row exposure through* $\mathsf{Norm}^{\uparrow}$*, metric or complete-metric inspection through M, finite derivative-window exposure through D, interpolation exponent-ledger exactness through E,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, compact handoff exposure through K, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 6814.1. Its analytic rows are S, N, M, D, E, R, K, and its structural rows are H, C, P, L.

The interpolation estimate is finite at every read. A consumer first sees the $\mathsf{Sobolev}^{\uparrow}$ derivative source S, the $\mathsf{Norm}^{\uparrow}$ target ledger N, the metric or complete-metric inspection row M, and the derivative-window row D before it may inspect the exponent ledger E.

The readback and handoff rows keep the estimate attached to finite data. The $\mathsf{RegSeqRat}^{\uparrow}$ row R records tolerance and bound observations, while K exposes only the compact-window information needed by downstream compactness routes.

Finally H, C, P, L transport, replay, source, and name the same packet. Since no row carries host embedding, quotient-function equality, selected representatives, countable choice, or completed function-space data, the listed rows exhaust the NameCert obligation surface. □

Theorem 6814.3 (Gagliardo-Nirenberg finite-window interpolation handoff)**.** *Every L10/L11 compactness or completion consumer of an accepted* $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ *packet receives the interpolation estimate only through the finite route*

$$\mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{Norm}^{\uparrow} \longmapsto (\mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}) \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow}.$$

The handoff may pass a displayed derivative-window estimate, exponent ledger, regular-rational tolerance readback, and compact-window row to later finite-window compactness chapters, but it exports no nonconstructive compact embedding theorem, global Sobolev embedding strength, quotient function equality, selected subsequence, countable choice, or ambient completed Sobolev carrier.

Proof. Start from the obligations of Definition 6814.2. They force the Sobolev source, norm target, metric inspection row, derivative windows, exponent ledger, readback row, and compact handoff row to appear in the same finite packet.

Read the sibling route in order. The $\mathsf{Sobolev}^{\uparrow}$ row supplies weak-derivative data, $\mathsf{Norm}^{\uparrow}$ supplies lower-order and higher-order norm observations, $\mathsf{Met}^{\uparrow}$ or $\mathsf{CompleteMetric}^{\uparrow}$ supplies inspection scale, and $\mathsf{RegSeqRat}^{\uparrow}$ supplies rational tolerance readback before the compact handoff row K is exposed.

The structural rows H, C, P, L keep that route stable under componentwise transport and replay. The compactness consumer receives a finite interpolation handoff only, not an ambient compact embedding theorem or any selected representative absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ packet is generated from a Sobolev source row, norm target row, metric or complete-metric inspection row, finite derivative-window row, interpolation exponent ledger, RegSeqRat readback, compact handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6814.1, Definition 6814.2, and Definition 6814.3 seed the finite interpolation route for L10/L11 compactness and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close nonconstructive compact embedding strength, global Sobolev embedding theorems, quotient function equality, selected subsequences, countable choice, ambient completed Sobolev carriers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite derivative-window exposure, exponent-ledger exactness, RegSeqRat readback, compact handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.GagliardoNirenbergFiniteWindowUp.

Chapter 6815

A Concrete Naming Certificate for $\mathsf{EhresmannConnection}^{\uparrow}$

Following Ehresmann's connection concept, the $\mathsf{EhresmannConnection}^{\uparrow}$ packet records a finite horizontal distribution over a displayed bundle projection rather than a covariant derivative on a vector bundle. It is grounded in the bundle projection surface of chapter 97, the general connection packet of chapter 134, the curvature consumer boundary of chapter 135, and the chart-local carrier discipline of chapter 96. The packet exports bundle-local horizontal lift and curvature-read rows; it does not import host smoothness, a principal-bundle action, or Riemannian metric compatibility as primitive data.

Definition 6815.1 (Ehresmann connection carrier)**.** *An* $\mathsf{EhresmannConnection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, \Pi, V, H, L, J, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bundle}^{\uparrow}$ *source row, displaying the total-space, base-space, fibre, projection, and local trivialization data visible to the packet.*
2. *Π is the bundle projection row whose fibre over each displayed base observation is available for local inspection.*
3. *V is the vertical-kernel row, recording the displayed kernel of the projection read at the inspected total-space observations.*
4. *H is the horizontal-row assignment, giving a chosen complement to V in each inspected tangent window.*
5. *L is the lift ledger, recording how a displayed base-direction request is lifted through H to a total-space direction over the same projection row.*
6. *J is the classifier row for projection, vertical kernel, horizontal row, lift ledger, and chart-local source data.*
7. *T records componentwise* hsame *transport for B, Π, V, H, L, J and their dependency rows.*
8. *C records displayed* Cont *replay from a base-direction request through projection, vertical-kernel readback, horizontal selection, and lift output.*
9. *P is the* Pkg *provenance over* $\mathsf{EhresmannConnection}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{EhresmannConnection}^{\uparrow}}$ *row.*
The carrier therefore names the horizontal distribution and lift ledger as finite bundle-local data. It has no coordinate for a host covariant derivative, a principal connection form, a Riemannian metric, or a global smooth splitting outside the displayed packet.

Theorem 6815.2 (Ehresmann connection NameCert obligation)**.** *For an accepted* $\mathsf{EhresmannConnection}^{\uparrow}$ *carrier* $E = (B, \Pi, V, H, L, J, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EhresmannConnection}^{\uparrow}}$ *surface is exhausted by carrier admission for the displayed bundle projection, vertical-kernel exposure, horizontal-row exposure, lift-ledger exposure, classifier stability under* hsame*, displayed* Cont *replay, provenance, and local naming. A replacement carrier preserving these rows componentwise preserves the accepted Ehresmann-connection packet, and no hidden host connection or metric-compatibility row is added by the name.*

Proof. Project the carrier of Definition 6815.1. The source rows B, Π, V, H, L are the only geometric rows available to the local certificate: B and Π display the bundle projection, V displays the vertical kernel, H displays the horizontal complement, and L displays the lift ledger.

The classifier row J records the admitted comparison of those rows. Componentwise transport through T can therefore preserve the accepted packet only by preserving projection, vertical-kernel, horizontal-row, lift-ledger, and chart-local source data together.

Replay is supplied by C. A base-direction request is accepted only after the projection, vertical-kernel readback, horizontal selection, and lift output have been replayed in that order.

Finally P and N package and name the same finite route. Since no carrier coordinate names a host covariant derivative, principal connection form, Riemannian metric, or global smooth splitting, none of those objects can be exported by the local NameCert row. □

Theorem 6815.3 (Ehresmann connection curvature consumer)**.** *Every* $\mathsf{Curvature}^{\uparrow}$ *consumer of an accepted* $\mathsf{EhresmannConnection}^{\uparrow}$ *carrier reads the horizontal distribution through finite bracket and transport rows derived from the route*

$$\mathsf{Bundle}^{\uparrow} \longmapsto \Pi \longmapsto V \longmapsto H \longmapsto L \longmapsto T, C, P, N.$$

The consumer may inspect the obstruction to horizontal lifts closing under bracket only after the bundle projection, vertical kernel, horizontal row, lift ledger, transport, replay, provenance, and local naming rows have been displayed. It receives no detached curvature tensor, host connection form, or metric-derived curvature object.

Proof. Start with a curvature request. The accepted carrier supplies the horizontal row H only as a complement to the displayed vertical-kernel row V over the projection row Π.

The bracket or transport read needed by a curvature consumer is therefore not a free-standing tensor. It is read through the lift ledger L and then transported and replayed by T and C over the same bundle source B.

The curvature chapter can consume such a route because chapter 135 reads curvature through finite boundary and transport data, while chapter 134 supplies the general connection-facing boundary discipline. Here the Ehresmann packet supplies the horizontal distribution route that those consumers inspect.

Since the carrier exports only $B, \Pi, V, H, L, J, T, C, P, N$, any curvature read that omits the displayed horizontal distribution, lift ledger, transport, replay, provenance, or local name leaves the accepted packet. No detached curvature tensor, host connection form, or metric-derived curvature object is present to be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{EhresmannConnection}^{\uparrow}}$.

Constructive story: An $\mathsf{EhresmannConnection}^{\uparrow}$ packet is generated from a displayed bundle projection, its vertical-kernel row, a horizontal complement row, a lift ledger, classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EhresmannConnection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6815.1, Definition 6815.2, and Definition 6815.3 seed the finite bundle-projection, vertical-kernel, horizontal-row, lift-ledger, classifier, transport, replay, provenance, and local naming surface for Ehresmann connection packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EhresmannConnection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host smoothness, principal-bundle actions, metric compatibility, detached curvature tensors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection exposure, vertical-kernel exposure, horizontal-row exposure, lift-ledger exactness, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.EhresmannConnectionUp.

Chapter 6816

A Concrete Naming Certificate for $\mathsf{WijsmanConvergence}^{\uparrow}$

Following Wijsman and Kelley, the $\mathsf{WijsmanConvergence}^{\uparrow}$ packet records the convergence read induced by distance-to-set probes on a metric hyperspace. It consumes the Wijsman topology surface of chapter 6182, the metric source of chapter 57, the closed-subset source of chapter 1660, and the finite name/readback chain of chapter 126, chapter 301, and chapter 38. The packet names convergence of hyperspace objects by finite distance-evaluation rows; it is not a host sequential topology, a quotient of closed-set equality, a selected nearest-point principle, or an ambient completeness theorem.

Definition 6816.1 (Wijsman convergence carrier)**.** *A* $\mathsf{WijsmanConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (T, M, H, S, R, E, Q, A, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{WijsmanTopology}^{\uparrow}$ *source row whose distance-evaluation opens supply the convergence classifier.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row from which all point-to-set distance requests are read.*
3. *H is the* $\mathsf{Hyperspace}^{\uparrow}$ *closed-subset source row carrying the finite hyperspace names being compared.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family for the approximating hyperspace name, the candidate limit name, and the finite probe points.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the stream windows and distance requests.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the displayed distance values.*
7. *Q is the finite convergence-question row, recording which probe windows require the distance-to-set values of the approximating names to agree with the limit name at the displayed rational tolerance.*
8. *A records componentwise* hsame *transport for T, M, H, S, R, E, Q.*
9. *C records displayed* Cont *replay for topology-source admission, metric-source admission, hyperspace-source admission, stream-window inspection, regular rational readback, real comparison, and convergence-question reads.*

10. *P is the* Pkg *provenance over* $\mathsf{WijsmanConvergence}^{\uparrow}$, $\mathsf{WijsmanTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WijsmanConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, H, S, R, E, Q, A, C, P, N$. It exports no host sequential topology, quotient closed-set equality, selected nearest point, pointwise choice over all probes, or ambient completeness coordinate.

Theorem 6816.2 (Wijsman convergence NameCert obligations)**.** *For every accepted* $\mathsf{WijsmanConvergence}^{\uparrow}$ *carrier*

$$V = (T, M, H, S, R, E, Q, A, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WijsmanConvergence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Wijsman topology source, metric source, hyperspace source, stream windows, regular rational readbacks, real comparison seal, convergence-question row, transport, replay, provenance, and local naming rows;*
2. *topology-source compatibility, requiring every convergence read to cite the* $\mathsf{WijsmanTopology}^{\uparrow}$ *distance-evaluation surface before it can inspect a probe;*
3. *metric readback exposure, requiring each probe comparison to pass through* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows for the same finite window;*
4. *hyperspace-name attachment, requiring the approximating name and candidate limit name to remain attached to the displayed* $\mathsf{Hyperspace}^{\uparrow}$ *row;*
5. *componentwise* hsame *transport through A, displayed* Cont *replay through C, and* Pkg *provenance through P for the same finite route;*
6. *local naming through N, excluding host sequential topologies, quotient closed-set equality, selected nearest points, pointwise choice over all probes, and ambient completeness.*

No Wijsman convergence obligation is stored outside these rows.

Proof. Project the carrier of Definition 6816.1. The displayed rows contain the Wijsman topology source, metric source, hyperspace names, stream windows, regular rational readbacks, real comparison seal, convergence-question row, transport, replay, provenance, and local naming row.

Use the topology source row T to bind the convergence classifier to the distance-evaluation topology of chapter 6182. A convergence read therefore cannot be a free host sequential-topology read; it must cite the finite Wijsman distance probes.

Read each probe through the metric and name rows. The rows M, H, S, R, E, Q expose the metric distance request, the hyperspace names under inspection, the finite stream windows, the regular rational readbacks, the real comparison seal, and the rational tolerance question as one displayed route.

Finally, A, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinates for quotient closed-set equality, selected nearest points, pointwise choice over all probes, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6816.3 (Wijsman convergence metric readback route)**.** *Every convergence-facing read from an accepted* $\mathsf{WijsmanConvergence}^{\uparrow}$ *carrier factors through*

$\mathsf{WijsmanTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The exported read is the displayed distance-to-set convergence route over finite metric probes and hyperspace names. It exports no host sequential topology, quotient closed-set equality, selected endpoint or nearest point, pointwise choice over all probes, or ambient completeness of the metric hyperspace.

Proof. Start with the NameCert obligations of Definition 6816.2. Topology-source compatibility places the $\mathsf{WijsmanTopology}^{\uparrow}$ row before any convergence question can be consumed.

Next read the metric route. The metric source M and hyperspace source H provide the only distance-to-set inputs, while S, R, E replay the stream windows, regular rational readbacks, and real comparison seals for the same finite probes.

Then consume the convergence-question row Q. It records finite tolerance checks for the approximating hyperspace name and the candidate limit name, so the consumer sees convergence only through the displayed metric readback route.

Finally use the structural rows A, C, P, N. They keep the read inside BHist, hsame, Cont, Pkg, and NameCert data and supply no coordinate for a host sequential topology, quotient closed-set equality, selected endpoint or nearest point, pointwise choice over all probes, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{WijsmanConvergence}^{\uparrow}$ convergence read factors through $\mathsf{WijsmanTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any completion-facing hyperspace consumer can inspect it.

Independence witness: wijsman-topology, metric, hyperspace, streamname, regseqrat, real: $\mathsf{WijsmanConvergence}^{\uparrow}$ is not bijective to any listed sibling because it binds Wijsman distance-evaluation opens to a finite convergence-question row over approximating and limit hyperspace names.

Closure status for $\mathsf{NameCert}_{\mathsf{WijsmanConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{WijsmanConvergence}^{\uparrow}$ packet is generated from a WijsmanTopology row, Metric row, Hyperspace row, StreamName windows, RegSeqRat readbacks, Real comparison seals, a finite convergence-question row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WijsmanConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6816.1, Definition 6816.2, and Definition 6816.3 seed the finite Wijsman convergence surface over WijsmanTopology, Metric, Hyperspace, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WijsmanConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequential topologies, quotient closed-set equality, selected nearest points, pointwise choice over all probes, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-source compatibility, metric readback exposure, hyperspace-name attachment, transport, replay, provenance, and local naming for BEDC.Derived.WijsmanConvergenceUp.

Chapter 6817

A Concrete Naming Certificate for $\mathsf{JacksonTheorem}^{\uparrow}$

Following Jackson's approximation theorem and its constructive approximation-theory readings, the $\mathsf{JacksonTheorem}^{\uparrow}$ packet records a quantitative approximation route from displayed smoothness or modulus data to polynomial or trigonometric approximation bounds. It sits beside the polynomial approximation surface of chapter 1634, the positive trigonometric-kernel route of chapter 1939, the compact-uniform continuity route of chapter 388, the uniform-modulus consumer of chapter 369, the finite polynomial carrier of chapter 50, and the terminal real-name seal of chapter 38. The packet names only the finite quantitative ledger by which an approximation consumer receives a rate, a degree or frequency budget, and an error comparison; it is not a host smoothness theorem, a selected best approximant, a quotient function-space construction, an open-cover compactness argument, or a completeness principle for ambient $\mathsf{Real}^{\uparrow}$.

Definition 6817.1 (JacksonTheorem carrier)**.** *A* $\mathsf{JacksonTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (X, F, M, R, D, P, T, E, H, C, Q, N)$$

with the following displayed rows:

1. *X is the compact real source row, read through the displayed* $\mathsf{Real}^{\uparrow}$ *and compact-interval surfaces.*
2. *F is the inspected continuous-map row.*
3. *M is the uniform-modulus row, read through* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$*.*
4. *R is the requested quantitative rate row, recording the modulus-to-degree or modulus-to-frequency budget.*
5. *D is the finite degree or frequency row used by the approximating family.*
6. *P is the* $\mathsf{Poly}^{\uparrow}$ *or trigonometric-polynomial row whose coefficients and evaluation ledger are finite.*
7. *T is the Fejer-kernel handoff row, present when the approximation route is trigonometric rather than algebraic.*
8. *E is the displayed error-comparison row binding F, M, R, D, and P to the requested tolerance.*
9. *H records componentwise* hsame *transport for source, function, modulus, rate, degree, polynomial, Fejer handoff, error, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from finite source windows through the modulus and approximation rows into the error comparison.*
11. *Q is the* Pkg *provenance over* JacksonTheorem$^{\uparrow}$, WeierstrassApproximation$^{\uparrow}$, FejerKernel$^{\uparrow}$, CompactUniformContin UniformModulus$^{\uparrow}$, Poly$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{JacksonTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, M, R, D, P, T, E, H, C, Q, N$. It has no coordinate for a selected optimal polynomial, a host modulus theorem, quotient equality of functions, unbounded Fourier convergence, countable choice, open-cover compactness, or ambient real completeness.

Theorem 6817.2 (JacksonTheorem NameCert obligations)**.** *For every accepted* JacksonTheorem$^{\uparrow}$ *carrier $J = (X, F, M, R, D, P, T, E, H, C, Q, N)$, the local* NameCert$_{\text{JacksonTheorem}^{\uparrow}}$ *surface consists exactly of compact source exposure through X, continuous-map exposure through F, uniform-modulus exposure through M, quantitative rate admission through R, degree or frequency admission through D, polynomial or trigonometric-polynomial admission through P, optional Fejer-kernel handoff through T, displayed error comparison through E, componentwise transport through H, replay through C, provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 6817.1. The analytic rows are X, F, M, R, D, P, T, E, and the structural rows are H, C, Q, N.

The modulus row M is the only source of quantitative continuity data. The rate row R and the degree or frequency row D therefore read a finite budget from the displayed modulus rather than from an external smoothness theorem.

The polynomial row P and the Fejer row T supply the approximation family. The error row E then binds that family to the inspected function and tolerance, while H, C, Q, N transport, replay, source, and name the same route.

Since no further coordinate is present, the listed rows exhaust the local NameCert obligations and exclude selected best approximants, quotient function equality, host compactness, unbounded Fourier convergence, choice, and ambient real completeness. □

Theorem 6817.3 (JacksonTheorem Weierstrass handoff)**.** *Every* WeierstrassApproximation$^{\uparrow}$ *or* FejerKernel$^{\uparrow}$ *consumer of an accepted* JacksonTheorem$^{\uparrow}$ *packet receives a quantitative route*

$$X, F, M, R, D, P, T, E, H, C, Q, N.$$

The Weierstrass-facing read consumes P as a finite Poly$^{\uparrow}$ *approximation row after M, R, D are displayed; the Fejer-facing read consumes T only as a finite positive-kernel handoff before the same error row E is exported.*

Proof. Start with the obligation surface of Definition 6817.2. It forces compact source data, the inspected continuous map, the uniform modulus, the quantitative rate row, and the degree or frequency row to be visible before any approximation family is read.

For the Weierstrass-facing route, read P through the polynomial surface of chapter 1634. The consumer receives the finite coefficient and evaluation ledger only after the modulus-to-degree budget has been displayed.

For the trigonometric route, read T through chapter 1939. The Fejer handoff supplies positivity and finite kernel averaging, but it does not add a host Fourier convergence theorem.

The error row E and structural rows H, C, Q, N then repack the same finite data for downstream approximation consumers. Thus the handoff is quantitative BEDC carrier data, not a selected best approximant or a quotient function-space assertion. □

Falsifiable prediction: An accepted $\mathsf{JacksonTheorem}^{\uparrow}$ consumer that reaches an approximation error row without first displaying a uniform-modulus row and a finite degree or frequency budget is outside the carrier surface.

Independence witness: weierstrass-approximation, fejer-kernel, compactuniformcontinuity, uniformmodulus, polynomial, real: none is bijective to $\mathsf{JacksonTheorem}^{\uparrow}$ because only this packet binds compact source data, a continuous-map row, a modulus-to-rate ledger, a finite degree or frequency budget, polynomial or trigonometric approximation rows, an error comparison, transport, replay, provenance, and local NameCert naming into one quantitative approximation route.

Closure status for $\mathsf{NameCert}_{\mathsf{JacksonTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{JacksonTheorem}^{\uparrow}$ packet is generated from compact Real source rows, a continuous-map row, compact-uniform and uniform-modulus rows, a quantitative rate ledger, a finite degree or frequency budget, polynomial or Fejer approximation rows, an error-comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JacksonTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6817.1, Definition 6817.2, and Definition 6817.3 seed the quantitative Jackson approximation packet over compact-uniform modulus data, finite polynomial or trigonometric approximation rows, and displayed error comparison.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JacksonTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, selected best approximants, host smoothness theorems, quotient equality of functions, unbounded Fourier convergence, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, modulus-to-rate exactness, degree or frequency budget admission, polynomial and Fejer handoff exactness, error comparison, transport, replay, provenance, and local naming for BEDC.Derived.JacksonTheoremUp.

Chapter 6818

A Concrete Naming Certificate for $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$

Following Bishop–Bridges Riemann–Darboux constructive analysis, the $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ packet records the finite compact-oscillation handoff that lets a regulated-function window be read through a compact finite partition before Darboux data is consumed. It sits over the regulated-function schedule of chapter 1906, the finite oscillation partition of chapter 3800, the Darboux sum boundary of chapter 1880, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite windows, cell ledgers, oscillation bounds, sum handoff rows, transport, replay, provenance, and local naming data.

Definition 6818.1 (Regulated function compact oscillation carrier)**.** *A* $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, W, M, O, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, exposing the regulated-function approximation schedule through the window requested by the consumer.*
2. *W is the finite window row over the inspected regulated-function schedule.*
3. *M is the compact finite partition-cell row supplied by* $\mathsf{FiniteOscillationPartition}^{\uparrow}$*, including the displayed cells and their common finite index ledger.*
4. *O is the oscillation ledger assigning each cell of M a tolerance-indexed upper and lower read of the regulated source.*
5. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, available only after F, W, M, O have been displayed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational endpoints and cellwise bounds used by O and D.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed regular rational reads.*
8. *H records componentwise* hsame *transport for $F, W, M, O, D, Q, R, C, P, N$.*
9. *C records displayed* Cont *replay from a regulated-function request through the finite window, compact finite partition cells, oscillation ledger, Darboux handoff, regular rational readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, FiniteOscillationPartitio DarbouxSum$^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}}$ *row.*

The carrier contains no open-cover compactness witness, no selected supremum, no quotient equality of functions, no measure completion, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegulatedFunctionCompactOscillationUp.RegulatedFunctionCompactOscillationTasteGate single carrier alignment`

Theorem 6818.2 (Regulated compact oscillation obligation surface)**.** *For every accepted* RegulatedFunctionCompac *carrier* $K = (F, W, M, O, D, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}}$ *surface consists exactly of regulated-function source admission, finite window exposure, compact finite partition-cell admission, oscillation-ledger accounting, Darboux handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6818.1. The analytic rows are F, W, M, O, D, Q, R, and the structural rows are H, C, P, N.

The source and partition rows are finite. The row F opens the regulated-function schedule, W confines the read to the inspected window, and M lists the compact finite partition cells supplied by the finite oscillation partition packet.

The oscillation and seal rows are also finite. The ledger O records the cellwise upper and lower reads, D records the Darboux handoff, Q records the regular rational readback, and R records the terminal real-name seal reached from those displayed reads.

Rows H, C, P, N only transport, replay, source, and name the same data. Since the carrier has no coordinate for open-cover compactness, selected suprema, quotient function equality, measure completion, or ambient completeness, the displayed list exhausts the local NameCert surface. □

Theorem 6818.3 (Regulated compact oscillation Darboux handoff)**.** *Every Darboux-facing consumer of an accepted* $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow},\ \mathsf{FiniteOscillationPartition}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow}.$$

The Darboux row is available only after the regulated-function window, compact finite partition cells, oscillation ledger, regular rational readback, Real seal, transport, replay, provenance, and local naming rows have been displayed.

Proof. Apply the obligation surface of Definition 6818.2. It forces every consumer to read $F, W, M, O, Q, R, H, C, P,$ before the Darboux handoff row D can be consumed.

The row F supplies the regulated-function source, while M supplies the compact finite partition cells. The oscillation ledger O binds the two by attaching cellwise upper and lower reads to the same inspected window.

The readback rows Q and R then route the cell bounds through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal. Thus D is a finite Darboux handoff over displayed rows, not a selected-extrema or open-cover compactness argument. □

Theorem 6818.4 (Regulated compact oscillation Real seal route)**.** *Every real-valued read exported by an accepted* $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q attached to the same regulated window, finite partition cells, oscillation ledger, and Darboux handoff. No exported read factors through quotient function equality, measure completion, selected suprema, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $K = (F, W, M, O, D, Q, R, H, C, P, N)$ from Definition 6818.1. The only real-facing rows are the regular rational readback Q and the terminal seal R.

Apply Definition 6818.3. Any integration-facing consumer has already read the regulated source, window, partition cells, oscillation ledger, and Darboux handoff before the real-valued cell data is used.

Since Q is the displayed readback for those same cells and R is downstream of Q, every exported real-valued read stays on the listed route. The refused readings would require carrier coordinates absent from K. □

Falsifiable prediction: Any accepted RegulatedFunctionCompactOscillationUp packet that exposes a Darboux row while omitting the regulated-function window, compact finite partition cells, oscillation ledger, RegSeqRat readback, Real seal, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: regulated-function, finite-oscillation-partition, darboux-sum, regseqrat, real: none is bijective with $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ because this packet jointly carries the regulated window, compact finite cells, oscillation ledger, Darboux handoff, rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ packet is generated from a regulated-function source, finite window row, compact finite partition cells, oscillation ledger, Darboux handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6818.1, Definition 6818.2, Definition 6818.3, and Definition 6818.4 seed the compact finite oscillation handoff from regulated-function windows to Darboux and Real readbacks.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected suprema, quotient function equality, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite partition-cell admission, oscillation-ledger exactness, Darboux handoff, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedFunctionCompactOscillationUp.

Chapter 6819

A Concrete Naming Certificate for $\mathsf{ArithmeticalContinuum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArithmeticalContinuum}^{\uparrow}$ packet records the arithmetical continuum as the finite BEDC route from rational and dyadic arithmetic through regular Cauchy names to the Real seal. It sits over the rational carrier of chapter 37, the dyadic ledger of chapter 291, the finite stream-window carrier of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names the displayed construction-order surface; it does not import Dedekind cuts, quotient reals, trichotomy, countable choice, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6819.1 (ArithmeticalContinuum carrier)**.** *An* $\mathsf{ArithmeticalContinuum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *arithmetic row naming the rational operations and rational comparison requests admitted by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger used to refine the rational requests into dyadic cells, dyadic tolerances, and dyadic comparison windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which construction-order consumers inspect the displayed arithmetic stream.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row serving the same finite stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the rational, dyadic, stream, and regular-readback rows have been replayed.*
6. *H records componentwise* hsame *transport for rational arithmetic, dyadic refinement, stream windows, regular readback, Real seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from rational requests to dyadic refinement, stream inspection, regular readback, and the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ArithmeticalContinuum}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArithmeticalContinuum}^{\uparrow}}$ *row.*

The carrier contains no coordinate for Dedekind cuts, quotient-real equality, trichotomy, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6819.2 (ArithmeticalContinuum NameCert obligations)**.** *For every accepted* $\mathsf{ArithmeticalContinuum}^{\uparrow}$ *carrier* $A = (Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArithmeticalContinuum}^{\uparrow}}$ *surface consists exactly of carrier admission for the rational, dyadic, stream, regular-readback, Real-seal, transport, replay, provenance, and local name rows; dyadic refinement exactness for rational arithmetic requests; finite stream-window coverage before regular readback; Real-seal handoff after* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *have been displayed; and nonescape excluding Dedekind cuts, quotient reals, trichotomy, selected representatives, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6819.1. The tuple displays exactly the rational arithmetic row, dyadic refinement ledger, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name.

Dyadic refinement exactness is read through Q and D. Each rational arithmetic request must expose its dyadic tolerance or comparison window before the request can be consumed downstream.

The construction-order read then passes through S and R. The stream window names the finite observations, and the regular rational row supplies the readback tied to those observations before the Real seal E is visible.

Finally H, C, P, N transport, replay, source, and name only the displayed route. Since the carrier has no Dedekind-cut, quotient-real, trichotomy, selected-representative, choice, host-equality, or completeness coordinate, none of those objects can be exported by the certificate. □

Theorem 6819.3 (ArithmeticalContinuum regular Cauchy handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{ArithmeticalContinuum}^{\uparrow}$ *carrier factors through the construction-order route*

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $Q, D, S, R, E, H, C, P, N$*. The handoff may transport and replay this finite route, but it cannot claim Dedekind-cut completeness, quotient-real equality, trichotomy, a selected representative, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier of Definition 6819.1. A Real-facing consumer has access to the terminal seal only at the row E.

Apply Definition 6819.2. The obligation surface forces the consumer to expose Q, D, S, R in order: rational arithmetic request, dyadic refinement, finite stream inspection, and regular rational readback.

The rows H and C preserve that route under componentwise transport and displayed replay, while P and N only record provenance and local naming. Thus the handoff reaches $\mathsf{Real}^{\uparrow}$ through the displayed regular Cauchy route, with no carrier coordinate from which to read the excluded classical objects. □

Falsifiable prediction: Every accepted ArithmeticalContinuumUp read exposes Rat arithmetic, DyadicRatCore refinement, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real-facing consumer can inspect the packet.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: removing any one of these sibling routes leaves the packet unable to display the corresponding rational request, dyadic refinement, stream window, regular-rational readback, or terminal Real seal row.

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmeticalContinuum}^{\uparrow}}$.

Constructive story: An $\mathsf{ArithmeticalContinuum}^{\uparrow}$ packet is generated from Rat arithmetic rows, DyadicRatCore refinement ledgers, StreamName finite windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArithmeticalContinuum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6819.1, Definition 6819.2, and Definition 6819.3 seed the arithmetical-continuum route from rational and dyadic arithmetic through regular Cauchy names to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArithmeticalContinuum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, trichotomy, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic refinement exactness, stream-window coverage, regular rational readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.ArithmeticalContinuu

Chapter 6820

A Concrete Naming Certificate for $\mathsf{RootChoiceSeparation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RootChoiceSeparation}^{\uparrow}$ packet records the finite data needed to choose the separated root window of a displayed sign-changing real interval without invoking a general root-choice principle. It consumes the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the finite stream-window route of chapter 126, the located real interval bisection route of chapter 5546, and the sign-change isolation packet of chapter 1407. The chapter names one separated root-choice surface over displayed interval data; it does not assert the general intermediate value theorem, compactness of intervals, quotient equality of real names, root selection without separation, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6820.1 (Separated root-choice carrier)**.** *A* $\mathsf{RootChoiceSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, I, A, B, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-read row, storing the finite endpoint and midpoint sign inspections used by the root-choice request.*
2. *I is the located real interval row, exposing the source interval and its* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *refinement route.*
3. *A is the apartness row, recording the displayed separation between the retained sign-changing window and every excluded candidate window.*
4. *B is the bisection schedule row, read through the* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *sign-change packet and its retained-window ledger.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained endpoints and midpoint inspections.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained stream windows and the displayed bisection schedule.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the selected separated window, available only after F, I, A, B, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for the function reads, interval row, apartness row, bisection schedule, stream windows, regular readback, real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from a root-choice request through the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{RootChoiceSeparation}^{\uparrow}$, $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$, $\mathsf{BisectionRootIsolation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RootChoiceSeparation}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for a general intermediate-value theorem, Heine–Borel compactness, quotient equality of real names, a root chosen without displayed separation, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RootChoiceSeparationUp.RootChoiceSeparationTasteGate single carrier alignment`

Theorem 6820.2 (Separated root-choice NameCert surface)**.** *For every accepted* $\mathsf{RootChoiceSeparation}^{\uparrow}$ *carrier*

$$R = (F, I, A, B, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RootChoiceSeparation}^{\uparrow}}$ *surface consists exactly of function-read exposure through F, located interval exposure through I, sign-change and apartness exposure through A, retained bisection scheduling through B, finite stream-window readout through S, regular rational readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every admitted root-choice read must display the separated sign-changing window before it reaches Q or E. The packet exports no general intermediate value theorem, no Heine–Borel theorem, no quotient real equality, no root selected without separation, and no ambient real-completeness principle.*

Proof. Project the carrier of Definition 6820.1. The rows F, I, A, B are the only pre-read rows available to the choice request, and they jointly display the function inspections, the located interval source, the separation witness, and the retained sign-change bisection schedule.

Read the neighboring bisection surfaces next. The located interval route of chapter 5546 supplies finite interval refinement, while chapter 1407 supplies the retained sign-changing window. The apartness row A keeps the retained window separated from excluded windows before any regular readback is consumed.

The downstream read is then forced through S, Q, E. The $\mathsf{StreamName}^{\uparrow}$ row records the finite endpoint windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row repacks those windows as regular rational readback, and the $\mathsf{Real}^{\uparrow}$ row seals only that displayed route.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate contains a general intermediate-value theorem, Heine–Borel compactness, quotient real equality, unseparated root choice, countable choice, or ambient real completeness, none of those objects can be exported by the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{RootChoiceSeparation}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without the displayed function-read row, located interval row, apartness row, retained bisection schedule, finite $\mathsf{StreamName}^{\uparrow}$ window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: real, regseqrat, streamname, located-real-interval-bisection, bisection-rootisolation: no listed sibling supplies the joint packet containing function reads, a located interval source, a separated sign-changing window, retained bisection scheduling, finite stream windows, regular readback, Real sealing, transport, replay, provenance, and local naming while refusing unseparated root choice and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RootChoiceSeparation}^{\uparrow}}$.

Constructive story: A RootChoiceSeparation$^{\uparrow}$ packet is generated from function-read rows, a located real interval row, a separated sign-change apartness row, a retained bisection schedule, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RootChoiceSeparation}\uparrow}$, seed)

Scope closed: Definition 6820.1 and Definition 6820.2 seed the separated root-choice surface over located interval bisection, bisection root isolation, finite stream windows, regular readback, and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{RootChoiceSeparation}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close the general intermediate value theorem, Heine–Borel compactness, quotient real equality, root choice without displayed separation, countable choice, ambient Real$^{\uparrow}$ completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separation visibility, retained bisection scheduling, readback handoff, Real seal nonescape, and classifier stability for BEDC.Derived.RootChoiceSeparationUp.

Chapter 6821

A Concrete Naming Certificate for $\mathsf{DedekindCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ packet names the finite L10 boundary around the Dedekind-cut route to the real line. It sits beside the located-cut presentation of chapter 3773, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, and the terminal real-name seal of chapter 38. The packet records where a cut-based presentation may be read as finite BEDC data; it refuses quotient completion and does not construct $\mathsf{Real}^{\uparrow}$ from Dedekind cuts.

Definition 6821.1 (DedekindCompletionBoundary carrier)**.** *A* $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, R, D, S, E, H, C, P, N)$$

with the following visible rows:

1. *L is the* $\mathsf{LocatedCutReal}^{\uparrow}$ *located-cut exposure row, recording lower–upper cut evidence only through displayed finite observations.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row used when a cut-side observation is compared with a regular-Cauchy presentation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds every comparison window used by L and R.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the bounded stream observations through which the comparison is replayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the located-cut, regular-readback, dyadic, and stream rows have been displayed.*
6. *H records componentwise* hsame *transport for the located-cut, regular-readback, dyadic, stream, Real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-facing request through L, R, D, S, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCompletionBoundary}^{\uparrow}$, $\mathsf{LocatedCutReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCompletionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, quotient-real equality, a selected representative, host real equality, countable choice, trichotomy, or an ambient construction of $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts.*

Lean4 (statement) BEDC.Derived.DedekindCompletionBoundaryUp

Theorem 6821.2 (Dedekind boundary located-cut route)**.** *Every completion-facing consumer of an accepted* $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ *carrier* $B = (L, R, D, S, E, H, C, P, N)$ *reads the cut-side data only through the displayed route*

$$\mathsf{LocatedCutReal}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may expose located cut evidence, regular rational readback, dyadic tolerances, finite stream windows, and the terminal Real seal, but it does not export quotient completion or construct $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts.*

Proof. Project the accepted carrier of Definition 6821.1. The only cut-side row is L, and it is paired in the same packet with the regular rational readback R, the dyadic ledger D, and the finite stream-window row S.

The continuation row C fixes the order in which a completion-facing request can inspect those rows. A request first sees located-cut evidence, then compares it against regular rational readback, then checks the dyadic tolerance ledger and the finite stream observations before the Real seal E is visible.

The structural rows H, P, N transport, source, and name only that displayed route. Since the carrier has no quotient-completion or Dedekind-construction coordinate, the consumer receives a boundary certificate rather than a construction of $\mathsf{Real}^{\uparrow}$ from cuts. □

Theorem 6821.3 (Dedekind boundary no quotient scope)**.** *For every accepted* $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ *carrier, the public scope is confined to the rows* $L, R, D, S, E, H, C, P, N$ *of Definition 6821.1. In particular, the packet refuses quotient completion, quotient-real equality, selected representatives, host real equality, countable choice, trichotomy, and any claim that* $\mathsf{Real}^{\uparrow}$ *is constructed from Dedekind cuts.*

Proof. The carrier definition lists every visible row. None of those rows stores a quotient relation, an equivalence class, a selected representative, a host equality proof, a choice schedule, or a trichotomy decision.

Apply the located-cut route of Definition 6821.2. It shows that the only Real-facing handoff passes through $\mathsf{LocatedCutReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

It remains only to inspect the structural rows. The hsame, Cont, Pkg, and local NameCert entries preserve, replay, source, and name the displayed finite route; they do not add a quotient completion object or a construction principle for $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ read exposes located-cut evidence, RegSeqRat readback, DyadicRatCore tolerances, StreamName finite windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any completion-facing consumer can inspect the packet.

Independence witness: located-cut-real, regseqrat, dyadicratcore, streamname, real: DedekindCompletionBounda is not reducible to any listed sibling because it binds the located-cut exposure, regular rational

readback, dyadic tolerance ledger, finite stream windows, terminal Real seal, transport, replay, provenance, and local NameCert rows as one boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCompletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCompletionBoundary}^{\uparrow}$ packet is generated from a LocatedCutReal exposure row, RegSeqRat readback, DyadicRatCore tolerance ledger, StreamName finite windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6821.1, Definition 6821.2, and Definition 6821.3 seed the L10 boundary around the Dedekind-cut completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, quotient-real equality, selected representatives, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a construction of $\mathsf{Real}^{\uparrow}$ from Dedekind cuts.

Upgrade path: Obligation closure requires checked carrier admission, located-cut exposure, regular rational readback, dyadic tolerance coverage, stream-window replay, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCompletionBoundaryUp.

Chapter 6822

A Concrete Naming Certificate for $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$ packet records the finite prefix-modulus data by which a regular Cauchy rational name exposes a requested real tolerance without appealing to tail choice or quotient stream equality. It sits between the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback surface of chapter 301, and the terminal seal handoff of chapter 38. The packet names only the finite prefix length, the displayed dyadic tolerance, the observed stream prefix, the regularity readback, the Real seal, componentwise transport, replay, provenance, and local $\mathsf{NameCert}$ evidence.

Definition 6822.1 (Regular Cauchy finite prefix modulus carrier)**.** *A* $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, D, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite prefix length at which the requested regular Cauchy read is opened.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic radius certified for that prefix.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row listing exactly the stream entries inspected below K.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback row witnessing that the observed prefix belongs to the regular rational Cauchy source.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached after the prefix, tolerance, observation, and regularity rows have been displayed.*
6. *H records componentwise* hsame *transport for K, D, W, Q, R, C, P, N.*
7. *C records displayed* Cont *replay from a requested dyadic tolerance to the same prefix length, observation window, regularity readback, and seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}}$ *evidence.*

The classifier compares accepted packets only through $K, D, W, Q, R, H, C, P, N$. It has no coordinate for an infinite tail selector, quotient stream equality, a chosen limit representative, countable choice, trichotomy of rationals beyond the displayed tolerance row, or an ambient Real completeness theorem.

Theorem 6822.2 (Regular Cauchy finite prefix modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$ *carrier*

$$M = (K, D, W, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}}$ *surface consists exactly of finite prefix length K, dyadic tolerance row D, stream observation row W, regular rational readback row Q, Real seal handoff row R, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N.*

Proof. Project the carrier of Definition 6822.1. The finite analytic rows are K, D, W, Q, R, and the structural rows are H, C, P, N.

The prefix length K bounds every observation made through W. The dyadic tolerance row D fixes the requested radius, while the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q states regularity for the displayed prefix before the $\mathsf{Real}^{\uparrow}$ seal row R is consumed.

Rows H, C, P, N transport, replay, source, and name the same displayed packet. Since the carrier includes no tail selector, quotient-stream equality, chosen representative, countable-choice schedule, or ambient completeness coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 6822.3 (Regular Cauchy finite prefix modulus L10 handoff)**.** *Every L10 Real or completion consumer of an accepted* $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$ *packet receives the modulus data only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may expose the prefix length, tolerance row, stream observations, regularity readback, and Real seal, but it exports no hidden infinite tail, quotient stream equality, selected limit, countable-choice scheduler, or host completeness principle.

Proof. Apply Definition 6822.2. The rows K, D, W, Q, R place the finite prefix length, dyadic tolerance, observed stream prefix, regular rational readback, and Real seal in one packet.

The structural rows H, C, P, N preserve that same route under componentwise transport, replay, provenance, and local naming. A consumer therefore reaches the Real or completion handoff only after reading the displayed finite prefix-modulus data, and every refused reading would require a coordinate absent from M. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}$ packet is generated from finite prefix length, DyadicRatCore tolerance, StreamName observations, RegSeqRat regularity readback, Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6822.1, Definition 6822.2, and Definition 6822.3 seed the finite prefix-modulus carrier and its L10 Real/completion handoff route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyFinitePrefixModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close tail choice, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix exactness, dyadic tolerance exactness, stream observation exactness, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFinitePrefixModulusUp.

Chapter 6823

A Concrete Naming Certificate for $\mathsf{MinkowskiQuestionMark}^{\uparrow}$

Following Minkowski's question-mark function and the continued-fraction/Stern-Brocot coding of real numbers, the $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ packet records the finite route by which a continued-fraction or Stern–Brocot prefix is read as a dyadic real-name prefix before any $\mathsf{Real}^{\uparrow}$ consumer inspects it. It sits over the Stern–Brocot tree of chapter 1704, the continued-fraction prefix carrier of chapter 342, the bridge packet of chapter 3781, the dyadic arithmetic surface of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite prefix conversion data; it does not assert quotient equality of real names, completed functional equality, selected infinite expansions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6823.1 (Minkowski question-mark carrier)**.** *A* $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, F, B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *branch-prefix row, recording the finite left/right address and boundary interval inspected by the consumer.*
2. *F is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *prefix row, recording the finite partial-quotient block corresponding to the same branch-prefix request.*
3. *B is the Stern–Brocot/continued-fraction bridge row, read through chapter 3781 before any dyadic row is exposed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *output-prefix ledger, recording the dyadic denominator scale and numerator prefix generated from the displayed branch or partial-quotient block.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row scheduling which branch, partial-quotient, and dyadic prefix cells are visible.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same dyadic prefix and window data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, F, B, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the Stern–Brocot, continued-fraction, bridge, dyadic-prefix, window, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a question-mark request through finite prefix inspection, dyadic prefix conversion, regular rational readback, and the terminal Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{MinkowskiQuestionMark}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{SternBrocotContinued}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MinkowskiQuestionMark}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, F, B, D, W, R, E, H, C, P, N$. The carrier has no coordinate for quotient real equality, completed function equality, trichotomy, countable choice, a selected infinite expansion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6823.2 (Minkowski question-mark NameCert obligations)**.** *For every accepted* $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ *carrier $Q = (S, F, B, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinkowskiQuestionMark}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Stern–Brocot branch prefix, continued-fraction prefix, bridge row, dyadic-prefix ledger, finite window, regular readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *prefix compatibility, requiring S and F to be read through the same bridge row B before D is available;*

3. *dyadic conversion, requiring D to expose only the dyadic prefix generated from the displayed branch or partial-quotient prefix and the window row W;*

4. *real-name routing, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *read to pass through D, W, R, E in that order;*

5. *transport, replay, provenance, and local-name stability through H, C, P, N.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6823.1. The accepted tuple exposes exactly the branch-prefix, partial-quotient, bridge, dyadic-prefix, window, readback, seal, and structural rows listed in the statement.

Prefix compatibility reads S, F, B together. The Stern–Brocot side supplies the finite branch address, the continued-fraction side supplies the finite partial-quotient block, and the bridge row records that both are the same finite prefix route before a dyadic prefix is presented.

The dyadic conversion row D is downstream of that bridge and of the window row W. Thus a consumer can inspect only the dyadic prefix generated from the displayed finite prefix, not a completed real function value.

The regular readback and Real seal are ordered by the carrier. The row R is obtained from D, W, and E is terminal after R.

Finally H, C, P, N transport, replay, source, and name only this route. Since no excluded completed-real or choice-theoretic coordinate is present, the five rows exhaust the local NameCert surface. □

Theorem 6823.3 (Minkowski question-mark prefix compatibility)**.** *Every accepted* $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ *carrier reads a Stern–Brocot branch prefix and a continued-fraction prefix through the same bridge row before exposing a dyadic prefix. In the notation of Definition 6823.1, the visible prefix route is*

$$S \longmapsto B \longleftarrow F \quad \textit{and then} \quad B, W \longmapsto D.$$

The compatibility claim is only about displayed finite prefixes; it is not equality of completed real functions or quotient equality of real names.

Proof. Project the accepted carrier $Q = (S, F, B, D, W, R, E, H, C, P, N)$. The branch-prefix coordinate is S, the partial-quotient coordinate is F, and the only row relating them is the bridge coordinate B.

Read B through the Stern–Brocot/continued-fraction bridge chapter. It mediates between finite tree intervals and finite continued-fraction prefixes by displayed rational-approximation data.

The dyadic row D is exposed only after B and the window row W have been read. Therefore both prefix descriptions feed the same dyadic prefix route, while completed function equality and quotient real equality remain outside the carrier. □

Theorem 6823.4 (Minkowski question-mark dyadic route)**.** *Every real-facing read exported by an accepted* $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ *carrier factors through*

$$\mathsf{SternBrocot}^{\uparrow},\ \mathsf{ContinuedFraction}^{\uparrow},\ \mathsf{SternBrocotContinuedFractionBridge}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto$$

The exported route is a finite dyadic real-name prefix route. It does not claim completeness of $\mathsf{Real}^{\uparrow}$*, quotient equality between real names, or total equality of Minkowski's question-mark function as a completed map.*

Proof. Apply Definition 6823.3. Any branch or partial-quotient source must pass through the bridge row before the dyadic prefix row is available.

The dyadic row is then read with the finite window row. The same window schedules the prefix cells passed to $\mathsf{StreamName}^{\uparrow}$, and the regular rational row R reads that dyadic prefix as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The terminal Real seal E is downstream of R. Since the structural rows only transport, replay, source, and name the same finite route, no ambient real completeness or completed functional equality can be read from the packet. □

Falsifiable prediction: Every $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ Real-facing read factors through a Stern–Brocot prefix, a continued-fraction prefix, their bridge row, a dyadic-prefix ledger, finite StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Independence witness: stern-brocot, continuedfraction, stern-brocot-continued-fraction-bridge, dyadicratcore, streamname, regseqrat, real: $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite prefix bridge, dyadic conversion ledger, real-name readback route, and structural support while refusing completed function equality, quotient real equality, selected infinite expansions, choice, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{MinkowskiQuestionMark}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MinkowskiQuestionMark}^{\uparrow}$ packet is generated from Stern–Brocot branch prefixes, continued-fraction prefixes, their bridge row, a DyadicRatCore prefix ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinkowskiQuestionMark}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6823.1, Definition 6823.2, Definition 6823.3, and Definition 6823.4 seed the finite route from continued-fraction and Stern–Brocot prefixes to dyadic Real-name readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinkowskiQuestionMark}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, completed functional equality, selected infinite expansions, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix compatibility, dyadic conversion, real-name routing, transport, replay, provenance, and local naming for BEDC.Derived.MinkowskiQuestionMarkUp.

Chapter 6824

A Concrete Naming Certificate for $\mathsf{CompactFanModulusSelector}^{\uparrow}$

Following Bishop–Bridges compact uniform continuity and the fan-bar proof of uniform moduli, the $\mathsf{CompactFanModulusSelector}^{\uparrow}$ packet records the finite selector that turns a fan-bar depth ledger and compact finite-net pointwise moduli into a single $\mathsf{UniformModulus}^{\uparrow}$ handoff. It realizes the compact-uniform-continuity direction of chapter 9237 as a concrete packet between the finite fan theorem of chapter 400, the Bishop fan-modulus route of chapter 1707, the compact-uniform-continuity carrier of chapter 388, and the downstream uniform-modulus consumer of chapter 369. The packet names only finite fan bars, compact ProbeBundle windows, pointwise modulus rows, rational lower-bound folds, transport, replay, provenance, and local naming data.

Definition 6824.1 (Compact fan modulus selector carrier)**.** *A* $\mathsf{CompactFanModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, B, K, G, R, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite bar row, including the binary-prefix spread, decidable bar classifier, and unary depth bound.*
2. *B is the* $\mathsf{BishopFanModulus}^{\uparrow}$ *fan-to-window row, turning the depth bound into a finite observation schedule.*
3. *K is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *source row, exposing compact finite-net and continuous-map pointwise modulus data for the inspected tolerance.*
4. *G is the finite* $\mathsf{ProbeBundle}$ *selector row, listing the compact-net probes whose pointwise moduli are inspected.*
5. *R is the positive rational lower-bound fold row over the selected probe radii and pointwise modulus margins.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained from F, B, K, G, R.*
7. *H records componentwise* hsame *transport for the fan, fan-modulus, compact-uniform, probe-selector, rational-fold, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform-continuity request through the fan-bar selector, ProbeBundle fold, and UniformModulus handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactFanModulusSelector}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BishopFanModulus}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFanModulusSelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, B, K, G, R, U, H, C, P, N$. The carrier has no coordinate for open-cover compactness, countable choice, selected branches, selected real limits, quotient cover equality, host analytic moduli, Lean verification, or bridge checking.

Theorem 6824.2 (Compact fan modulus selector NameCert obligations)**.** *For every accepted* $\mathsf{CompactFanModulusSelector}^{\uparrow}$ *carrier $S = (F, B, K, G, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactFanModulusSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed fan bar, Bishop fan-modulus, compact-uniform source, ProbeBundle selector, rational fold, UniformModulus handoff, transport, replay, provenance, and local naming rows;*
2. *fan-bar coverage, requiring the row F to supply the finite branch-prefix depth consumed by B;*
3. *compact selector coverage, requiring G to list the finite probes and pointwise modulus rows exposed by K;*
4. *rational fold exactness, requiring R to be the finite lower-bound fold over the selected probe margins;*
5. *UniformModulus handoff, requiring U to read only the displayed rows F, B, K, G, R before exposing a uniform precision schedule;*
6. *transport, replay, provenance, and local-name stability through H, C, P, N.*

No local selector obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 6824.1. The accepted tuple displays exactly the fan, Bishop fan-modulus, compact-uniform source, ProbeBundle selector, rational fold, uniform handoff, and structural rows.

The fan-bar coverage row is finite. The fan theorem row F supplies the bounded branch-prefix inspection, and B converts that depth into the observation schedule used by the selector.

The compact selector rows are also finite. The compact-uniform source K exposes pointwise modulus data, while G lists the probes whose margins are used by the fold.

The fold row R performs only a finite positive-rational lower-bound comparison. The uniform row U consumes that fold together with the fan-bar and compact rows, so it is a handoff from displayed data rather than a choice-selected modulus.

Finally H, C, P, N transport, replay, source, and name the same finite selector. Since the carrier has no coordinate for open-cover compactness, choice, selected branches, selected limits, quotient cover equality, host analytic moduli, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 6824.3 (Compact fan selector uniform handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactFanModulusSelector}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{BishopFanModulus}^{\uparrow} \longmapsto \mathsf{CompactUniformContinuity}^{\uparrow} \longmapsto \mathsf{ProbeBundle} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformMod}$$

The handoff exports a displayed precision schedule from the fan-bar depth, compact finite-net probes, pointwise modulus rows, and rational lower-bound fold; it exports no open-cover compactness theorem, selected branch, countable-choice schedule, quotient cover equality, or host analytic modulus.

Proof. Begin with the fan-bar side. The row F exposes the finite fan data, and B converts its depth into a finite observation schedule.

Read the compact-uniform side next. The row K supplies the compact finite-net and pointwise modulus data, while G selects the finite probe rows that participate in the fold.

The rational fold R produces the displayed lower bound over those rows. The $\mathsf{UniformModulus}^{\uparrow}$ handoff U can therefore read one precision schedule from finite data already present in the packet.

The structural rows preserve the same route under transport, replay, provenance, and local naming. Since no excluded compactness or choice coordinate is present, the handoff remains a finite selector packet. □

Theorem 6824.4 (Compact fan selector nonescape)**.** *An accepted* $\mathsf{CompactFanModulusSelector}^{\uparrow}$ *carrier exports no open-cover compactness, countable choice, selected branch, selected real limit, quotient cover equality, host analytic modulus, Lean verification, or bridge checking. Every exported read factors through* $F, B, K, G, R, U, H, C, P, N$*.*

Proof. Project the accepted carrier rows. The analytic coordinates are the fan row F, Bishop fan-modulus row B, compact-uniform source row K, ProbeBundle selector row G, rational fold row R, and UniformModulus handoff row U.

The remaining coordinates H, C, P, N provide componentwise transport, continuation replay, provenance, and local naming for those same rows. They do not add compactness, choice, branch-selection, limit-selection, quotient-cover, host-modulus, verification, or bridge data.

Thus every consumer is confined to the displayed finite selector route, and each excluded reading lacks a carrier coordinate from which it could be consumed. □

Falsifiable prediction: Every accepted CompactFanModulusSelectorUp handoff factors through a finite fan bar, Bishop fan-modulus schedule, compact-uniform source, ProbeBundle selector, rational lower-bound fold, UniformModulus row, transport, replay, provenance, and local naming.

Independence witness: fantheorem, bishop-fan-modulus, compactuniformcontinuity, uniform-modulus: $\mathsf{CompactFanModulusSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the fan-bar depth, compact finite-net selector, pointwise modulus rows, rational lower-bound fold, uniform handoff, and structural support while refusing compactness, choice, branch selection, quotient cover equality, host moduli, verification, and bridge checking.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFanModulusSelector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactFanModulusSelector}^{\uparrow}$ packet is generated from a FanTheorem finite bar, a BishopFanModulus schedule, CompactUniformContinuity finite-net and pointwise-modulus rows, a ProbeBundle selector, a rational lower-bound fold, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactFanModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6824.1, Definition 6824.2, Definition 6824.3, and Definition 6824.4 seed the finite selector from fan-bar depth and compact pointwise moduli to a UniformModulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactFanModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, selected real limits, quotient cover equality, host analytic moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-bar coverage, compact selector coverage, rational fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactFanModulusSelecto

Chapter 6825

A Concrete Naming Certificate for $\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}$

Following Kuratowski and Zorn on maximal principles, the $\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}$ packet records the finite BEDC boundary at which a maximality request is visible without importing choice or a host-level maximal selector. It sits beside the ordered-carrier rows of chapter 1378 and chapter 53, the set-like membership rows of chapter 3668, the finite chain discipline of Definition 53.56, and the NameCert, BHist, Cont, and Pkg support surfaces already used by finite certificate packets. The chapter names the displayed pressure surface around Zorn-style maximality; it does not prove Zorn's lemma, does not add choice, and exports only finite displayed boundary data.

Definition 6825.1 (Kuratowski-Zorn lemma boundary carrier)**.** *A* $\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (O, S, C, U, M, F, H, R, P, N)$$

with the following displayed rows:

1. *O is the underlying* $\mathsf{Order}^{\uparrow}$ *or* $\mathsf{POSet}^{\uparrow}$ *source row whose comparison relation is being inspected through finite BEDC data.*
2. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership row for the displayed carrier fragment under inspection.*
3. *C is the chain-window row, recording a finite chain-like order window and its local comparison entries.*
4. *U is the upper-bound row, recording a proposed upper-bound witness for the same displayed chain window.*
5. *M is the maximality-request row, recording the consumer request that asks whether a displayed element can be extended inside the same order surface.*
6. *F is the refusal row for global choice, an unrestricted maximal selector, host powerset search, quotient collapse of chains, and an ambient proof of Zorn's lemma.*
7. *H records componentwise* hsame *transport for the order, membership, chain-window, upper-bound, maximality-request, refusal, provenance, and local naming rows.*
8. *R records displayed* Cont *replay for finite inspection of the same rows.*
9. *P is the* Pkg *provenance over the order, set-like, chain-window, upper-bound, maximality-request, refusal, transport, replay, and local naming surfaces.*

10. N is the local $\mathsf{NameCert}_{\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}}$ *row.*
The carrier contains no coordinate for a choice function, a maximal-element selector, a completed chain union outside the displayed upper-bound row, a host powerset traversal, or a proof of Zorn's lemma.

Theorem 6825.2 (Kuratowski-Zorn lemma boundary NameCert obligations)**.** *For the carrier of Definition 6825.1, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}}$ *surface consists exactly of finite carrier admission for $Z = (O, S, C, U, M, F, H, R, P, N)$, order and set-like source exposure through O, S, chain-window exposure through C, upper-bound exposure through U, maximality-request exposure through M, refusal exactness through F, componentwise transport through H, replay through R, provenance through P, and local naming through N.*

Proof. Project the carrier rows. The order source and set-like membership rows are read through chapter 1378, chapter 53, and chapter 3668, while the finite chain-window row is read through the finite chain-like order surface of Definition 53.56.

Read the maximality-facing rows next. The upper-bound row U is attached to the same displayed chain window C, and the maximality-request row M records only the finite request seen by the consumer. The row F refuses a global choice function, a maximal selector, host powerset traversal, quotient collapse of chain data, and a proof of Zorn's lemma.

Finally transport, replay, source, and name the packet through H, R, P, N. These rows are the standard NameCert, BHist, Cont, and Pkg support surface for finite BEDC certificates, so the obligation list is exhausted by the displayed packet and no hidden maximal-principle resource is exported. □

Theorem 6825.3 (Kuratowski-Zorn lemma boundary choice refusal)**.** *Every accepted* KuratowskiZornLemmaBound *read that exposes a Zorn-style maximality request must factor through the finite chain-window row C, the displayed upper-bound row U, the maximality-request row M, and the refusal row F before any order consumer reads the packet. Consequently the boundary does not prove Zorn's lemma, does not add choice, does not select a maximal element, and exports only finite displayed boundary data.*

Proof. Begin with the NameCert obligations of Definition 6825.2. They force the order source, set-like source, chain-window, upper-bound, maximality request, refusal, transport, replay, provenance, and local naming rows to be read as one finite packet.

If the refusal row is absent, the consumer has no BEDC record separating a finite maximality request from a host choice principle or maximal selector. If the chain-window or upper-bound row is absent, the request is no longer grounded in displayed finite order data. If transport, replay, provenance, or local naming is absent, the request is detached from the certificate surface.

Thus an accepted read can expose only the finite boundary around a maximal-principle request. The carrier records where Zorn-style pressure appears, while refusing a proof of Zorn's lemma, choice, global maximal selection, and ambient powerset search. □

Falsifiable prediction: Any accepted KuratowskiZornLemmaBoundaryUp packet that exposes a maximality request while omitting the finite chain-window, displayed upper-bound, refusal, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: order, poset, setlike, finite-chain: none is bijective with $\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}$ because this packet jointly carries chain-window rows, upper-bound rows, a maximality request, and explicit refusal of global choice and maximal selectors.

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}$ packet is generated from order or POSet source rows, SetLike membership rows, finite chain-window rows, displayed upper-bound rows, a maximality-request row, a refusal row for choice and maximal selectors, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6825.1, Definition 6825.2, and Definition 6825.3 seed the finite boundary around Zorn-style maximality requests without proving Zorn's lemma or adding choice.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiZornLemmaBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Zorn's lemma, global choice, maximal-element selection, host powerset traversal, quotient chain collapse, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite chain-window exposure, upper-bound exposure, maximality-request exactness, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.KuratowskiZornLem

Chapter 6826

A Concrete Naming Certificate for $\mathsf{ClassifyingSpace}^{\uparrow}$

Following Steenrod and May on universal bundles, classifying spaces, and simplicial models for topological classification, the $\mathsf{ClassifyingSpace}^{\uparrow}$ packet records a finite NameCert seed for the route from a displayed topological group action to a bundle classifier. It sits after the $\mathsf{TopGroup}^{\uparrow}$ operation and continuity surface of chapter 204, the $\mathsf{Bundle}^{\uparrow}$ local-trivialization surface of chapter 97, the $\mathsf{Homotopy}^{\uparrow}$ deformation route of chapter 156, the $\mathsf{SimplicialSet}^{\uparrow}$ face-and-degeneracy ledger of chapter 157, and the $\mathsf{Cat}^{\uparrow}$ object-and-arrow carrier of chapter 61. The chapter names only finite classifier rows, universal-bundle dependency rows, simplicial approximation rows, homotopy transport, categorical provenance, replay, and local naming; it does not construct quotient spaces, choose representatives of orbits, assert ambient homotopy equivalence, or import a host principal-bundle classification theorem.

Definition 6826.1 (ClassifyingSpace carrier)**.** *A* $\mathsf{ClassifyingSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, U, S, H, C, T, R, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{TopGroup}^{\uparrow}$ *row whose multiplication, unit, inverse, and continuity data are used by the classifier.*
2. *U is the* $\mathsf{Bundle}^{\uparrow}$ *row naming the displayed universal-bundle dependency and its local trivialization data.*
3. *S is the* $\mathsf{SimplicialSet}^{\uparrow}$ *row carrying finite face and degeneracy ledgers used to present the classifying object.*
4. *H is the* $\mathsf{Homotopy}^{\uparrow}$ *row recording the deformation route through which bundle classifiers are compared.*
5. *C is the* $\mathsf{Cat}^{\uparrow}$ *row naming the finite object and arrow context for classifier morphisms and pullback-shaped reads.*
6. *T records componentwise* hsame *transport for the topological-group, bundle, simplicial, homotopy, and category rows.*
7. *R records displayed* Cont *replay of classifier reads through G, U, S, H, C, T.*

8. *P is the* Pkg *provenance over* $\mathsf{ClassifyingSpace}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifyingSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-space formation, orbit-representative choice, host homotopy equivalence, ambient universal object selection, or an unrestricted principal-bundle classification theorem.

Lean4 ✓ `BEDC.Derived.ClassifyingSpaceUp.ClassifyingSpaceTasteGate single carrier alignment`

Theorem 6826.2 (ClassifyingSpace NameCert obligations)**.** *For the carrier of Definition 6826.1, the local* $\mathsf{NameCert}_{\mathsf{ClassifyingSpace}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed packet $B = (G, U, S, H, C, T, R, P, N)$;*
2. *topological-group exposure through the* $\mathsf{TopGroup}^{\uparrow}$ *row G;*
3. *universal-bundle and classifier exposure through the* $\mathsf{Bundle}^{\uparrow}$ *row U together with the* $\mathsf{SimplicialSet}^{\uparrow}$ *row S;*
4. *homotopy and categorical transport through the* $\mathsf{Homotopy}^{\uparrow}$ *row H, the* $\mathsf{Cat}^{\uparrow}$ *row C, and componentwise transport T;*
5. *replay, provenance, and local naming through R, P, N.*

Ledger exactness says that every accepted classifying-space read exposes precisely those rows before a bundle-classifier route is consumed. The obligations do not add quotient spaces, selected orbit representatives, host universal bundles, ambient homotopy equivalences, or a principal-bundle classification theorem.

Theorem 6826.3 (ClassifyingSpace principal-bundle route)**.** *Every accepted* $\mathsf{ClassifyingSpace}^{\uparrow}$ *packet routes a principal-bundle classifier read through*

$$\mathsf{TopGroup}^{\uparrow},\ \mathsf{Bundle}^{\uparrow},\ \mathsf{SimplicialSet}^{\uparrow},\ \mathsf{Homotopy}^{\uparrow},\ \mathsf{Cat}^{\uparrow}.$$

The topological-group row fixes the structure group surface, the bundle row fixes the local-trivialization dependency, the simplicial row presents the finite classifying object ledger, the homotopy row records classifier comparison, and the category row records the object-and-arrow context for pullback-shaped reads. A consumer that omits one of these rows is outside the accepted $\mathsf{ClassifyingSpace}^{\uparrow}$ *carrier.*

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifyingSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifyingSpace}^{\uparrow}$ packet is generated from TopGroupUp operation data, BundleUp local-trivialization data, SimplicialSetUp face and degeneracy ledgers, HomotopyUp comparison routes, CategoryUp object and arrow context, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassifyingSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6826.1, Definition 6826.2, and Definition 6826.3 seed the finite classifying-space NameCert surface over topological groups, bundles, homotopy, simplicial carriers, and categories.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassifyingSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-space formation, orbit-representative choice, ambient universal-object selection, host homotopy equivalence, principal-bundle classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topological-group exposure, universal-bundle dependency, simplicial ledger exactness, homotopy comparison, categorical context, transport, replay, provenance, and local naming for BEDC.Derived.ClassifyingSpaceUp.

Chapter 6827

A Concrete Naming Certificate for $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ packet records the finite boundary around Dedekind completeness for the real-name route. It sits beside the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the Cauchy route of chapter 3380, and the located substitute of chapter 1427. The packet names the separation between accepted Cauchy and located real-name handoffs and the unavailable assertion that $\mathsf{Real}^{\uparrow}$ is Dedekind complete.

Definition 6827.1 (Real Dedekind-completeness boundary carrier)**.** *A* $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, R, S, L, C, H, P, N)$$

with the following displayed rows:

1. *D is the cut-style request row, recording the attempted least-upper-bound or cut-completion read without admitting a Dedekind-cut carrier.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the existing real-name endpoint accepted by the surrounding real-name route.*
3. *S is the joint* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, requiring every accepted real-facing observation to pass through finite stream windows and regular rational data.*
4. *L is the located substitute row, pointing the request toward located supremum or located Cauchy-completeness surfaces when those rows are present.*
5. *C records displayed* Cont *replay from the cut-style request to the real-name seal, the readback row, the located substitute, and the local boundary name.*
6. *H records componentwise* hsame *transport for D, R, S, L, C, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RealCauchyCompleteness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, R, S, L, C, H, P, N. It has no coordinate for a Dedekind-cut construction, quotient real equality, trichotomy, countable choice, selected suprema, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6827.2 (Real Dedekind-completeness boundary NameCert obligations)**.** *For every accepted* $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ *carrier* $K = (D, R, S, L, C, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealDedekindCompl}}$ *surface consists exactly of carrier admission for the eight displayed rows, route exposure through the* $\mathsf{Real}^{\uparrow}$ *seal and the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *readback row, located substitute exactness through* L*, non-claim boundary for the omitted Dedekind-completeness coordinates, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6827.1. The packet exposes only D, R, S, L, C, H, P, N, so any boundary read must begin with those rows.

Route exposure is forced by R and S. The real-facing endpoint is the existing $\mathsf{Real}^{\uparrow}$ seal R, and the readback surface S requires finite $\mathsf{StreamName}^{\uparrow}$ windows together with $\mathsf{RegSeqRat}^{\uparrow}$ data before the endpoint can be consumed.

The located substitute is exactly the row L. When a supremum-shaped request has located evidence, L points to the located route rather than to an arbitrary cut-completion assertion; otherwise the request remains only the displayed boundary row D.

The non-claim boundary follows from the missing coordinates. Since no row stores Dedekind-cut construction data, quotient real equality, trichotomy, countable choice, selected suprema, or ambient completeness, the local certificate cannot export any of those principles.

Finally H, C, P, N transport, replay, source, and name precisely the same rows. Componentwise transport preserves the boundary inventory, continuation replay preserves the read order, and the provenance ledger adds no independent completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ consumer that presents a Dedekind-completeness read for RealUp must display the cut-style request row, the RealUp seal row, the StreamName and RegSeqRat readback row, the located substitute row, transport, continuation replay, provenance, and local naming.

Independence witness: real, streamname, regseqrat, locatedsupremum, real-cauchy-completeness: $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ is not bijective to any listed sibling because it stores the rejected cut-style request together with the accepted real-name seal, finite readback route, located substitute, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}$ packet is generated from a cut-style request row, a RealUp seal row, StreamName and RegSeqRat readback, a located substitute row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6827.1 and Definition 6827.2 seed the finite boundary separating accepted Cauchy and located RealUp routes from a Dedekind-completeness claim.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDedekindCompletenessBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter excludes Dedekind-cut construction, quotient real equality, trichotomy, countable choice, selected suprema, ambient Real completeness, Lean verification, and bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed Real Dedekind-completeness boundary route.

Chapter 6828

A Concrete Naming Certificate for $\mathsf{DyadicIntervalCompletion}^{\uparrow}$

Following Bishop and Bridges on interval methods and regular real approximation, the $\mathsf{DyadicIntervalCompletion}^{\uparrow}$ packet realizes the L10 interval-completion route of chapter 9238 as finite NameCert data. It consumes the dyadic endpoint surface of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite route from shrinking dyadic intervals to a Real-facing seal; it does not add a quotient stream, selected limit, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6828.1 (Dyadic interval completion carrier)**.** *A* $\mathsf{DyadicIntervalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the shrinking dyadic interval row, recording left and right endpoint data over* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *T is the endpoint tolerance ledger, recording the finite radius and endpoint comparison obligations attached to the same interval row.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the inspected interval stages.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows and tolerance ledgers.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, T, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval, tolerance, stream-window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an interval-completion request through I, T, S, R, E and back to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCompletion}^{\uparrow}}$ *row.*

The carrier exports only this finite interval-completion packet. It has no coordinate for quotient streams, selected limits, countable-choice schedules, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DyadicIntervalCompletionUp`

Theorem 6828.2 (Dyadic interval completion NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalCompletion}$ *carrier* $K = (I, T, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed packet, interval shrinking through* I*, endpoint tolerance ledger exposure through* T*, finite-window readback through* S *and* R*, Real-seal nonescape through* E*, componentwise transport through* H*, displayed replay through* C*, provenance exhaustion through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6828.1. The rows $I, T, S, R, E, H, C, P, N$ are the complete visible packet, so carrier admission can only admit data already displayed in those rows.

The interval route first reads I and T. Thus the shrinking dyadic endpoints and their endpoint tolerance ledger are exposed before any stream window, regular readback, or real seal can be consumed.

Next read S and R. The stream row names the finite inspected stages, while the regular rational row supplies readback from the same finite data. The seal row E is downstream of that finite route and cannot introduce an independent completeness principle.

Finally use H, C, P, N. These rows transport, replay, source, and name the same packet, so the provenance ledger is exhausted by the displayed carrier surface. □

Theorem 6828.3 (Dyadic interval completion Real-seal route)**.** *Every completion-facing read of an accepted* $\mathsf{DyadicIntervalCompletion}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the finite rows $I, T, S, R, E, H, C, P, N$*. The route may transport and replay the displayed interval-completion packet, but it cannot export a quotient stream, selected limit, countable choice, host equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the obligation theorem. It forces every completion-facing consumer to expose the dyadic interval row, endpoint tolerance ledger, finite stream window, and regular rational readback before the terminal Real seal can be inspected.

The cited sibling chapters supply exactly those surfaces. DyadicRatCore supplies the endpoint and tolerance rows, StreamName supplies finite windows, RegSeqRat supplies regular rational readback, and the Real chapter supplies the terminal seal row.

The structural rows then preserve the same route. Componentwise hsame transport and displayed Cont replay cannot add a new completion coordinate, while P and N source and name the finite packet. No row remains from which a consumer could recover quotient streams, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalCompletion}^{\uparrow}$ packet is generated from shrinking DyadicRatCoreUp interval rows, endpoint tolerance ledgers, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicIntervalCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 6828.1, Definition 6828.2, and Definition 6828.3 seed the finite interval-completion route from dyadic endpoints to a Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalCompletion}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DyadicIntervalCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval shrinking, endpoint tolerance ledger exposure, finite-window readback, Real-seal nonescape, transport, replay, provenance exhaustion, and local naming for BEDC.Derived.DyadicIntervalCompletionUp.

Chapter 6829

A Concrete Naming Certificate for $\mathsf{HopfFibration}^{\uparrow}$

Following Hopf's fibration theorem, the $\mathsf{HopfFibration}^{\uparrow}$ packet records the circle-fibre projection from the displayed three-sphere source to the displayed two-sphere target as finite BEDC NameCert data. It sits over the circle carrier of chapter 58, the bundle surface of chapter 97, the Lie-group endpoint rows of chapter 140, the vector-bundle bookkeeping of chapter 210, and the Riemann-sphere chart target of chapter 6698. The packet names the fibration route only through displayed rows; it does not import a host quotient sphere, ambient smooth map equality, selected local sections, or a completed principal-bundle theorem.

Definition 6829.1 (Hopf fibration carrier)**.** *A* $\mathsf{HopfFibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S^3, S^2, F, \Pi, C, L, V, T, R, P, N)$$

with the following displayed rows:

1. S^3 *is the source sphere row whose circle-phase boundary is read through* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$.
2. S^2 *is the target sphere row, read through the* $\mathsf{RiemannSphere}^{\uparrow}$ *chart surface.*
3. F *is the circle-fibre row, displaying the* $\mathsf{S}^{1\uparrow}$ *fibre over each inspected target observation.*
4. Π *is the projection row from the source packet to the target packet.*
5. C *is the componentwise classifier for source, target, fibre, projection, chart, and fibre-action rows.*
6. L *is the local trivialization ledger supplied through the* $\mathsf{Bundle}^{\uparrow}$ *dependency surface.*
7. V *records the vector-bundle and tangent-window rows used only to compare displayed local coordinates.*
8. T *records componentwise* hsame *transport for* $S^3, S^2, F, \Pi, C, L, V$ *and their dependency rows.*
9. R *records displayed* Cont *replay from a fibre request through projection, chart readback, circle-fibre readback, and local trivialization.*
10. P *is the* Pkg *provenance over* $\mathsf{HopfFibration}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannSphere}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. N *is the local* $\mathsf{NameCert}_{\mathsf{HopfFibration}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of spheres, selected global section, ambient smooth-map extensionality, or a host principal-bundle object outside the displayed packet.

Theorem 6829.2 (Hopf fibration NameCert obligations)**.** *For every accepted* $\mathsf{HopfFibration}^{\uparrow}$ *carrier* $H = (S^3, S^2, F, \Pi, C, L, V, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HopfFibration}^{\uparrow}}$ *surface consists exactly of source-sphere admission, target-chart admission, circle-fibre exposure, projection exposure, classifier stability, local trivialization ledger exposure, vector-window comparison, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6829.1. The geometric rows are S^3, S^2, F, Π, L, V, while C, T, R, P, N are the classifier, transport, replay, provenance, and naming rows attached to the same packet.

The source and target rows are read only through their displayed dependencies. The source circle phase reaches chapter 58; the target chart reaches chapter 6698; and the local trivialization reaches chapter 97.

The fibre row F is then compared through the $\mathsf{S}^{1\uparrow}$ and $\mathsf{LieGroup}^{\uparrow}$ rows, while V supplies only the finite vector-window comparison used by local coordinates. Componentwise transport through T and replay through R preserve exactly those rows.

Since no carrier coordinate names a host quotient sphere, selected global section, smooth-map extensionality principle, or external principal bundle, the listed rows exhaust the local NameCert surface. □

Theorem 6829.3 (Hopf fibration circle-fibre handoff)**.** *Every* $\mathsf{Bundle}^{\uparrow}$*,* $\mathsf{LieGroup}^{\uparrow}$*, or* $\mathsf{RiemannSphere}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{HopfFibration}^{\uparrow}$ *packet must expose the source row* S^3*, target row* S^2*, circle-fibre row* F*, projection row* Π*, local trivialization ledger* L*, vector-window row* V*, transport row* T*, replay row* R*, provenance row* P*, and local name row* N *before the circle-fibre handoff can be consumed. The handoff exports only the displayed* $\mathsf{S}^{1\uparrow}$ *fibre over the displayed target row.*

Proof. Start with Definition 6829.2. It makes the source sphere, target chart, circle fibre, projection, local trivialization, and vector-window comparison rows first-class parts of the accepted packet.

A bundle consumer can therefore read a fibre only after the projection and local trivialization rows have been displayed. A Lie-group consumer reaches only the circle action already present in the $\mathsf{S}^{1\uparrow}$ fibre row, and a Riemann-sphere consumer reaches only the target chart row S^2.

Finally the structural rows T, R, P, N keep the same packet boundary under transport, replay, provenance, and naming. Thus the handoff is the displayed circle-fibre route itself, with no selected global section, quotient sphere equality, ambient smooth-map equality, or completed principal-bundle theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{HopfFibration}^{\uparrow}}$.

Constructive story: A $\mathsf{HopfFibration}^{\uparrow}$ packet is generated from displayed source-sphere rows, target Riemann-sphere chart rows, circle-fibre rows, projection rows, classifier stability, local trivialization ledgers, vector-window comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HopfFibration}^{\uparrow}}$, seed)

Scope closed: Definition 6829.1, Definition 6829.2, and Definition 6829.3 seed the finite circle-fibre projection route from the displayed three-sphere source to the displayed Riemann-sphere target.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HopfFibration}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.HopfFibrationUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of spheres, selected global sections, ambient smooth-map equality, host principal bundles, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, target-chart exposure, circle-fibre exposure, projection exposure, local trivialization exactness, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.HopfFibrationUp.

Lean4 (statement) `BEDC.Derived.HopfFibrationUp`

Chapter 6830

A Concrete Naming Certificate for $\mathsf{Kelleyfication}^{\uparrow}$

Following Kelley and Steenrod compactly generated topology, the $\mathsf{Kelleyfication}^{\uparrow}$ packet records the compactly generated reflection of a topology as finite BEDC data. It sits after the topology carrier of chapter 88, the core-compact neighbourhood route of chapter 6350, and the compact-open topology ledger of chapter 4138. The packet names only the compact-window tests through which a topology is reflected; it does not import a host category of spaces, arbitrary open-cover compactness, quotient-open equality, or choice-selected compact witnesses.

Definition 6830.1 (Kelleyfication carrier)**.** *A* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, C, O, W, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row whose carrier, open-incidence rows, and classifier are already admitted as finite topology data.*
2. *C is the* $\mathsf{CoreCompact}^{\uparrow}$ *row, or an equivalent compactly generated test row, recording which compact-window probes may be used to inspect the source topology.*
3. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *finite compact-window ledger. It lists the compact-domain windows and open-target tests used to decide reflected openness.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *inspection window, recording the finite observations through which a consumer reads the compact-window ledger.*
5. *H records componentwise* hsame *transport for the source topology, compact-generation tests, compact-open ledger, inspection window, replay, provenance, and local naming rows.*
6. *R records displayed* Cont *replay from a reflected-open request through T, C, O, W, H before any downstream topology or function-space consumer reads the result.*
7. *P is the* Pkg *provenance over* $\mathsf{Kelleyfication}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CoreCompact}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through T, C, O, W, H, R, P, N. It has no coordinate for arbitrary host opens, quotient-open equality, an unrestricted category of compactly generated

spaces, open-cover compactness, countable choice, or selected compact tests outside the displayed ledger.

Lean4 ✓ `BEDC.Derived.KelleyficationUp.KelleyficationTasteGate single carrier alignment`

Theorem 6830.2 (Kelleyfication NameCert obligations)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}$ *surface consists exactly of source-topology exposure through* T*; compactly generated test exposure through* C*; compact-open finite-window exposure through* O*; finite inspection through* W*; componentwise transport through* H*; displayed replay through* R*; provenance through* P*; and local naming through* N*. Every reflected-open read must first pass through the topology source of chapter 88, the core-compact route of chapter 6350, and the compact-open topology ledger of chapter 4138.*

Lean4 ✓ `BEDC.Derived.KelleyficationUp.KelleyficationCarrier namecert obligations`

Proof. Project the carrier in Definition 6830.1. The displayed rows are exactly T, C, O, W, H, R, P, N, so any accepted read begins with the source topology, the compact-generation test row, the compact-window ledger, and the finite inspection window.

The compactly generated reflection is read through sibling rows rather than through a host topology predicate. The topology source supplies the original open-incidence data, the core-compact row supplies compact-neighbourhood tests, and the compact-open topology row supplies the finite compact-window ledger.

Transport and replay preserve those rows componentwise. Provenance and local naming then bind the same packet as a $\mathsf{Kelleyfication}^{\uparrow}$ certificate. Since no quotient-open equality, arbitrary open-cover compactness, selected compact-test family, or ambient category row occurs in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 6830.3 (Kelleyfication compact-generated obligation)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, the compact-generated obligation is a row projection from the displayed carrier:* T *supplies the* $\mathsf{Topology}^{\uparrow}$ *source,* C *supplies the* $\mathsf{CoreCompact}^{\uparrow}$ *compact-generation tests,* O *supplies the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *compact-window ledger,* W *supplies the* $\mathsf{StreamName}^{\uparrow}$ *inspection window,* H *supplies componentwise* hsame *transport,* R *supplies displayed* Cont *replay,* P *supplies* Pkg *provenance, and* N *supplies the local* $\mathsf{NameCert}$ *row. No compact-generated read may be certified by a host topology predicate or by a compactness witness outside* T, C, O, W, H, R, P, N*.*

Lean4 ✓ `BEDC.Derived.KelleyficationUp.KelleyficationCarrier compact generated obligation`

Proof. Project the carrier of Definition 6830.1. The rows T, C, O, W, H, R, P, N are the only rows available to an accepted $\mathsf{Kelleyfication}^{\uparrow}$ packet.

The compact-generated obligation is therefore read from the carrier, not transported in as an external parameter. The topology source T fixes the open-incidence data, the CoreCompact row C fixes the compact-generation tests, and the CompactOpenTopology row O fixes the finite compact-window ledger inspected through the StreamName row W.

The support rows preserve that same inventory. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming cannot create an additional compactness coordinate, so the compact-generated obligation is exhausted by the displayed rows. □

Theorem 6830.4 (Kelleyfication compact-open scope)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, the compact-open scope of a reflected-open read is exactly the finite*

route through the $\mathsf{Topology}^{\uparrow}$ *row* T*, the* $\mathsf{CoreCompact}^{\uparrow}$ *row* C*, the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *row* O*, the* $\mathsf{StreamName}^{\uparrow}$ *row* W*, the* BHist *packet rows, componentwise* hsame *transport* H*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. A downstream read cannot leave this route to consult quotient-open equality, ambient compact-open structure, or a selected family of compact witnesses.*

Proof. Read the carrier rows in order. The reflected-open request begins at the source topology row T, is restricted by the compact-generation row C, and reaches the compact-open ledger only through the finite window row O.

The inspection row W turns that compact-open ledger into a StreamName-visible finite observation. Since the underlying packet is a finite BHist carrier, the observation has no independent ambient compact-open coordinate.

Finally apply the support rows. The hsame row H and Cont row R replay the same T, C, O, W scope, while P and N record provenance and local naming for the same packet. Thus the compact-open scope is a carrier projection and not parameter transport. □

Theorem 6830.5 (Kelleyfication compact-open exactness)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, a reflected-open read is admitted exactly through the finite route*

$$T \to C \to O \to W \to H \to R \to P \to N.$$

The source topology row T *supplies the underlying open-incidence data, the* $\mathsf{CoreCompact}^{\uparrow}$ *row* C *supplies the compact-generation tests, the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *row* O *supplies the finite compact-window ledger, and the* $\mathsf{StreamName}^{\uparrow}$ *row* W *supplies the inspection windows. No reflected-open read is accepted from a host topology predicate, quotient-open equality, arbitrary compact-cover test, or selected compact family outside those rows.*

Proof. Project the carrier rows of Definition 6830.1. A reflected-open request has no coordinate before T, C, O, W are visible, so it first reads the topology source, core-compact test row, compact-open ledger, and inspection window.

The exactness assertion is row-local. The compact-generation row decides which compact-window probes are available, and the compact-open ledger records the corresponding source-open tests before the StreamName window exposes any finite observation.

The structural rows then preserve the same route. Componentwise hsame transport, displayed Cont replay, provenance, and local naming can only replay T, C, O, W; since the carrier contains no host-open coordinate or selected compact-family coordinate, every accepted reflected-open read factors through the displayed compact-open route. □

Theorem 6830.6 (Kelleyfication core-compact scope)**.** *In an accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier, compact-generation tests are scoped by the displayed rows* C *and* O*. A downstream topology or function-space consumer may inspect only the core-compact probes named by* C *and the compact-open windows named by* O*, after reading the source topology row* T *and before replay through* H, R, P, N*.*

Proof. Read the rows T, C, O from Definition 6830.1. The source topology row fixes the open-incidence data being reflected, while the core-compact row names the allowed compact-generation probes.

The compact-open ledger refines that scope to finite windows. Any consumer-facing test therefore has to pass through O after C, so it cannot introduce an unlisted compact test or a host compactness predicate.

Finally, the replay and naming rows preserve only the displayed ledger. The rows H, R, P, N transport, replay, source, and name the same compact-generation scope, so the accepted packet exports no additional compact-generation surface. □

Theorem 6830.7 (Kelleyfication inspection-window stability)**.** *Let $K = (T, C, O, W, H, R, P, N)$ be an accepted* Kelleyfication$^{\uparrow}$ *carrier. Componentwise* hsame *transport and displayed* Cont *replay preserve the finite inspection window W: after transport and replay, every reflected-open observation still reads the same source topology row, core-compact test row, compact-open ledger, and StreamName window. Transport and replay cannot add host opens, quotient-open equality, arbitrary open-cover compactness, or selected compact tests.*

Proof. Use the transport row H componentwise. It relates the source topology, compact-generation tests, compact-open ledger, inspection window, replay, provenance, and local naming rows without adding a new coordinate.

Now apply the replay row R. A replayed reflected-open observation must still traverse T, C, O, W before it reaches provenance or local naming, so the finite inspection window is unchanged.

The nonescape claim follows from the carrier shape. Host opens, quotient-open equality, arbitrary open-cover compactness, and selected compact tests are absent from K, and neither hsame nor Cont creates carrier fields. Thus stability preserves exactly the displayed Kelleyfication inspection route. □

Theorem 6830.8 (Kelleyfication compact-window exhaustion)**.** *For every accepted* Kelleyfication$^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, every accepted reflected-open read is exhausted by the displayed rows T, C, O, W, H, R, P, N. The compact-window ledger O and the core-compact test row C must be read before any downstream topology or function-space consumer receives the reflected-open result. No such read may be supplied by host opens, quotient-open equality, arbitrary open-cover compactness, or a selected compact-test family outside the carrier.*

Proof. Project the carrier of Definition 6830.1. Its only reflected-open route starts at the source topology row T, then exposes the compact-generation tests C, the compact-window ledger O, and the finite inspection window W before any structural replay.

Apply compact-open exactness from Definition 6830.5. It identifies the accepted reflected-open read with the finite route

$$T \to C \to O \to W \to H \to R \to P \to N.$$

Next use core-compact scope from Definition 6830.6. A downstream topology or function-space consumer may inspect only the compact-generation probes named by C and the compact-open windows named by O, so both rows precede every consumer-facing read.

Finally replay inspection-window stability. Transport and continuation preserve the same T, C, O, W rows, while the carrier contains no coordinate for host opens, quotient-open equality, arbitrary open-cover compactness, or a selected compact-test family. Thus the displayed rows exhaust the reflected-open read. □

Lean4 ✓ `BEDC.Derived.KelleyficationUp.KelleyficationCarrier compact window exhaustion`

Theorem 6830.9 (Kelleyfication obligation-closure route)**.** *For every accepted* Kelleyfication$^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, the obligation-closure route is the row projection that combines the compact-generated obligation, compact-open scope and exactness, core-compact scope,*

inspection-window stability, and compact-window exhaustion. The route exports exactly the topology source row T, compact-generation row C, compact-open ledger O, inspection window W, componentwise transport H, displayed replay R, provenance P, and local naming N, and it exports no host topology predicate, quotient-open equality, arbitrary open-cover compactness, or selected compact-test family.

Proof. Begin with the compact-generated obligation of Definition 6830.3. It makes the compact-generated read a projection from T, C, O, W, H, R, P, N.

Use compact-open scope and exactness, Definition 6830.4 and Definition 6830.5, together with core-compact scope Definition 6830.6. These force every reflected-open read through the finite route

$$T \to C \to O \to W \to H \to R \to P \to N.$$

Finally apply inspection-window stability and compact-window exhaustion, Definition 6830.7 and Definition 6830.8. They keep the same rows visible after transport and replay, so the obligation route contains no external compactness or quotient-open coordinate. □

Theorem 6830.10 (Kelleyfication transport replay obligation)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, the transport and replay obligation is local to the displayed rows: H transports the source topology, compact-generation tests, compact-open ledger, inspection window, replay, provenance, and local naming componentwise, while R replays only the reflected-open route already fixed by T, C, O, W. No transported or replayed read may introduce host opens, quotient-open equality, arbitrary open-cover compactness, or selected compact tests.*

Proof. Project the transport row H from Definition 6830.1. It has components only for the rows already displayed in the carrier, so it cannot add a new topology or compactness coordinate.

Now read the replay row R through compact-open exactness, Definition 6830.5. The replayed request must pass through T, C, O, W before provenance or local naming is exposed.

Inspection-window stability and compact-window exhaustion, Definition 6830.7 and Definition 6830.8, then identify the transported and replayed route with the same finite compact-window ledger. The excluded host and choice-selected objects are absent from every row used in the projection. □

Theorem 6830.11 (Kelleyfication carrier reflection)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, reflected topology data is visible only through the displayed rows T, C, O, W, H, R, P, N. In particular, a compact-open or topology consumer may read the* $\mathsf{Topology}^{\uparrow}$ *source,* $\mathsf{CoreCompact}^{\uparrow}$ *compact-generation tests,* $\mathsf{CompactOpenTopology}^{\uparrow}$ *ledger, StreamName inspection window, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, but it may not read a host compactly generated topology or quotient-open equality outside those rows.*

Proof. Project the carrier of Definition 6830.1. The only reflected topology data carried by K are the source topology row T, compact-generation row C, compact-open ledger O, and inspection row W.

Apply the compact-generated obligation of Definition 6830.3. It makes every compact-generation read a row projection from the same carrier rather than an external topology predicate.

Now use compact-open exactness, Definition 6830.5. The reflected-open read follows T, C, O, W before the structural rows H, R, P, N can transport, replay, source, or name it. Since no host topology or quotient-open coordinate is present, reflection is exhausted by the displayed rows. □

Theorem 6830.12 (Kelleyfication compact-open test stability)**.** *Let $K = (T, C, O, W, H, R, P, N)$ be an accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier. Compact-open tests named by O remain stable under componentwise* hsame *transport and displayed* Cont *replay: after transport and replay, a reflected-open observation still reads the same source topology row, core-compact test row, compact-open ledger, and StreamName inspection window before any consumer sees the result.*

Proof. Begin with compact-open exactness, Definition 6830.5. It identifies an accepted reflected-open observation with the finite route through T, C, O, W, H, R, P, N.

Apply inspection-window stability from Definition 6830.7. The transport row H preserves the source topology, compact-generation tests, compact-open ledger, inspection window, replay, provenance, and local naming componentwise.

Finally use the transport replay obligation of Definition 6830.10. The replay row R replays only the compact-open route already fixed by T, C, O, W, so the transported test cannot acquire host opens, quotient-open equality, arbitrary open-cover compactness, or selected compact tests. □

Theorem 6830.13 (Kelleyfication StreamName scope)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, the StreamName row W bounds all reflected-open observations before a topology or function-space consumer reads the result. A consumer receives only compact-open observations visible through the finite route*

$$T \to C \to O \to W \to H \to R \to P \to N,$$

and receives no unwindowed compact-open test, host open predicate, quotient-open equality, or selected compact-test family.

Proof. Use core-compact scope, Definition 6830.6. It places the compact-generation tests in C and the compact-open windows in O after the source topology row T and before any downstream consumer.

Next apply inspection-window stability, Definition 6830.7. The StreamName row W remains attached to the same T, C, O route under componentwise transport and replay.

Finally apply compact-window exhaustion, Definition 6830.8. It forces every reflected-open read through T, C, O, W, H, R, P, N, so topology and function-space consumers cannot bypass the window by consulting unlisted compact-open tests or host open predicates. □

Theorem 6830.14 (Kelleyfication local naming obligation)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier $K = (T, C, O, W, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}$ *row N names only the obligation surface already exposed by T, C, O, W, H, R, P. The naming row records the compact-generated source, compact-open window, core-compact scope, inspection-window stability, transport, replay, and provenance obligations; it does not name a host compactly generated topology object, quotient-open equality, arbitrary open-cover compactness, or selected compact-test family.*

Proof. Use the NameCert obligations theorem Definition 6830.2. It states that local naming is one row of the same accepted carrier and follows after source topology, compact-generation tests, compact-open ledger, inspection window, transport, replay, and provenance.

The obligation-closure route of Definition 6830.9 supplies the required row inventory. It combines compact-generated obligation, compact-open exactness, core-compact scope, inspection-window stability, and compact-window exhaustion without adding an external object.

Thus N names exactly the displayed Kelleyfication packet. Since the carrier lacks host compactly generated topology, quotient-open equality, arbitrary open-cover compactness, and selected compact-test coordinates, local naming cannot export them. □

Theorem 6830.15 (Kelleyfication compact-open subbase)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, the compact-open subbase visible to a downstream consumer is exactly the* BHist*-visible route through the* $\mathsf{Topology}^{\uparrow}$ *source row* T*, the* $\mathsf{CoreCompact}^{\uparrow}$ *window row* C*, the compact-open test ledger* O*, the StreamName inspection row* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *naming* N*. No subbase element is certified by host compact-open topology data, selected compact witnesses, or quotient-open equality outside those rows.*

Proof. Project the carrier of Definition 6830.1. The compact-open ledger O is a row in the same finite BHist packet as the topology source T and the core-compact window row C.

The subbase read is therefore row-visible. It may inspect the $\mathsf{Topology}^{\uparrow}$ source, the $\mathsf{CoreCompact}^{\uparrow}$ windows, and the finite compact-open tests only through O and the StreamName window W.

The structural rows preserve that same inventory. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row authenticate the compact-open tests without adding host compact-open data or selected compact witnesses. □

Theorem 6830.16 (Kelleyfication corecompact window)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$*, each* $\mathsf{CoreCompact}^{\uparrow}$ *window exposed by* C *is read after the* $\mathsf{Topology}^{\uparrow}$ *source row* T *and before the compact-open test ledger* O*. The row sequence*

$$T \to C \to O \to W \to H \to R \to P \to N$$

is the complete consumer-facing path through BHist*-visible compact-open tests, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

Proof. Begin with the source row T. It fixes the open-incidence data on which any core-compact window can be tested.

The row C then names the allowed compact-generation windows, and O records how those windows are used by compact-open tests. The StreamName row W is the only finite observation channel for those tests.

Finally apply the support rows. Transport H and replay R keep the displayed order, while P and N record provenance and local naming for the same packet. Since no other carrier coordinate exists, the displayed route is complete. □

Theorem 6830.17 (Kelleyfication NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}$ *obligation surface for an accepted carrier* $K = (T, C, O, W, H, R, P, N)$ *consists exactly of* BHist*-visible compact-open tests over the* $\mathsf{Topology}^{\uparrow}$ *source row,* $\mathsf{CoreCompact}^{\uparrow}$ *windows, finite inspection through* W*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the naming row* N*. Every compact-open or corecompact consumer route must factor through these rows before it can cite the Kelleyfication packet.*

Proof. Use Definition 6830.15. It identifies compact-open subbase reads with the finite row route through T, C, O, W, H, R, P, N.

Use Definition 6830.16. It places each corecompact window between the topology source and the compact-open ledger, with StreamName inspection before transport, replay, provenance, and naming.

The local NameCert row cannot name anything outside that route. Thus the obligation surface is exactly the displayed compact-open and corecompact packet, not an ambient topology predicate or an external compactness witness. □

Theorem 6830.18 (Kelleyfication compact-window scope route)**.** *Every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *reflected-open consumer factors through the ordered finite route*

$$T \to C \to O \to W \to H \to R \to P \to N$$

of Definition 6830.1. The route is fixed by the compact-open subbase of Definition 6830.15, the corecompact window ordering of Definition 6830.16, and the local NameCert obligation surface of Definition 6830.17. It exports no host compactly generated topology, quotient-open equality, arbitrary open-cover compactness, selected compact-test family, or ambient category datum.

Proof. Start with the carrier of Definition 6830.1. A reflected-open consumer has access only to the displayed rows T, C, O, W, H, R, P, N, so its first possible route is the source topology, compact-generation test row, compact-open ledger, and finite inspection window.

Apply the compact-open subbase theorem, Definition 6830.15. It identifies consumer-visible compact-open tests with the same finite row route and prevents a consumer from reading host compact-open data or selected compact witnesses.

Next apply the corecompact window theorem, Definition 6830.16. It places each corecompact window after the topology source and before the compact-open ledger, so the compact-window request has the ordered prefix $T \to C \to O \to W$ before transport and replay.

Finally use the NameCert obligation surface of Definition 6830.17. The local naming row authenticates only the compact-open and corecompact route already displayed through H, R, P, N. Since the carrier has no coordinate for host compactly generated topology, quotient-open equality, arbitrary open-cover compactness, selected compact tests, or ambient category data, no accepted reflected-open consumer can leave the ordered finite route. □

Theorem 6830.19 (Kelleyfication reflection public interface)**.** *Every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier* $K = (T, C, O, W, H, R, P, N)$ *exposes a reflected-open public interface only through compact-generated rows. A consumer that asks for a reflected-open read must first pass through the* $\mathsf{Topology}^{\uparrow}$ *source row, the* $\mathsf{CoreCompact}^{\uparrow}$ *compact-window row, the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *compact-open ledger, the inspection window* W*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. No accepted read exports host compactly generated topology, quotient-open equality, selected compact witnesses, arbitrary open-cover compactness, or ambient category data.*

Proof. Start with the carrier of Definition 6830.1. Its visible route is exactly T, C, O, W, H, R, P, N, so a reflected-open consumer has no coordinate outside the topology source, compact-window row, compact-open ledger, inspection window, and structural rows.

Use the compact-generated obligation and compact-open exactness surfaces, Definition 6830.15 and Definition 6830.17. They force each reflected-open read to remain attached to the displayed compact-window and compact-open ledger before replay and naming.

Finally apply the corecompact scope and inspection-window stability blocks, Definition 6830.16 and Definition 6830.18. These place the $\mathsf{CoreCompact}^{\uparrow}$ window before the compact-open read and keep W, H, R, P, N downstream. Since no other row is present, the public interface exports only the compact-generated reflected-open rows. □

Theorem 6830.20 (Kelleyfication compact-window public export)**.** *For every accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *carrier, compact-window tests remain visible before any reflected-open read is exported. The public export route is*

$$T \to C \to O \to W \to H \to R \to P \to N,$$

so the compact-window row C and compact-open ledger O are part of the exported certificate rather than hidden witnesses.

Proof. Project the carrier of Definition 6830.1. The only route to a reflected-open read starts with the $\mathsf{Topology}^{\uparrow}$ source row T and then exposes the compact-window row C.

Apply the compact-open exactness theorem Definition 6830.15. It identifies the reflected-open read with the displayed compact-open ledger O over the same compact-window data.

Use the inspection-window stability route of Definition 6830.18. It keeps W visible before transport, replay, provenance, and naming are consumed. Therefore the public export cannot hide compact-window tests behind quotient-open equality or an ambient compactly generated topology. □

Theorem 6830.21 (Kelleyfication corecompact consumer boundary)**.** *Every* $\mathsf{CoreCompact}^{\uparrow}$ *consumer of an accepted* $\mathsf{Kelleyfication}^{\uparrow}$ *packet receives the finite neighbourhood route through* T, C, O, W, H, R, P, N *and does not receive quotient-open equality. The consumer may cite the compact-generated source, compact-window tests, compact-open ledger, inspection window, transport, replay, provenance, and local naming, but no selected compact-test family or host topology predicate is exported.*

Proof. Begin with the corecompact window theorem Definition 6830.16. It places the $\mathsf{CoreCompact}^{\uparrow}$ request after the source topology row and before the compact-open ledger.

Apply the NameCert obligation surface Definition 6830.17. The local certificate records only compact-open tests over the displayed source row, compact windows, inspection window, transport, replay, provenance, and naming.

The compact-window scope route Definition 6830.18 then gives the finite neighbourhood path $T \to C \to O \to W \to H \to R \to P \to N$. Since quotient-open equality, selected compact-test families, host topology predicates, and ambient category data are absent from the carrier, no corecompact consumer can read them. □

Closure status for $\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Kelleyfication}^{\uparrow}$ packet is generated from a topology source row, a compactly generated or CoreCompact test row, a compact-open finite-window ledger, a StreamName inspection window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 6830.1 and the checked obligation theorem Definition 6830.2 enumerate the compact-generation, compact-open, inspection-window, transport, replay, provenance, and local naming obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Kelleyfication}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.KelleyficationUp.KelleyficationCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close host compactly generated topology, quotient-open equality, arbitrary open-cover compactness, choice-selected compact tests, Lean verification, or bridge checking.

Upgrade path: Public export requires the same compact-generation, compact-open, StreamName inspection, transport, replay, provenance, and local naming route to be exposed as a public reflected-open interface.

Chapter 6831

A Concrete Naming Certificate for $\mathsf{FiniteObservationTopology}^{\uparrow}$

The $\mathsf{FiniteObservationTopology}^{\uparrow}$ packet realizes the observer-expansion window of chapter 9463 as a concrete NameCert carrier. It sits beside the finite observation substrate of chapter 126, the topology carrier of chapter 88, and the hyperbolic observer-expansion packets referenced by the vision. The packet names a finite observation topology only through displayed windows, residue-boundary rows, transport, replay, provenance, and local naming; it does not install an ambient observer space, a completed boundary, or a host topological quotient.

Definition 6831.1 (Finite observation topology carrier)**.** *A* $\mathsf{FiniteObservationTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, E, B, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation source row, recording the finite windows through which an observer-expansion ledger may be inspected.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row whose open-incidence data are restricted to the finite observation surface exposed by S.*
3. *E is the observer-expansion ledger row read from chapter 9463; it records the finite expansion coordinates without turning them into a global observer object.*
4. *B is the residue-boundary row, naming the visible boundary of the finite observation window and the hidden remainder left outside that window.*
5. *W is the finite topology-window ledger, listing which observations from S are used to test the topology row T and the expansion row E.*
6. *H records componentwise* hsame *transport for observation source, topology row, expansion ledger, residue boundary, topology-window ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite observation-topology request through S, T, E, B, W, H.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationTopology}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTopology}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, E, B, W, H, C, P, N$. It has no coordinate for a completed observer manifold, a selected far-end point, a quotient topology on unseen observations, a global boundary compactification, or an unledgered hidden residue.

Theorem 6831.2 (Finite observation topology NameCert obligations)**.** *For every accepted* FiniteObservationTopolo *carrier $F = (S, T, E, B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTopology}^{\uparrow}}$ *surface consists exactly of finite observation source exposure through S; topology-row exposure through T; observer-expansion ledger exposure through E; residue-boundary exposure through B; finite topology-window exposure through W; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. Lean follow-up requires* `Nontrivial` *and* `FieldFaithful` *instances before the corresponding TasteGate surface can be checked.*

Proof. Project the carrier in Definition 6831.1. The public observation-topology rows are exactly $S, T, E, B, W, H, C, P, N$.

The topology row is finite-facing. It can be inspected only through the windows listed by S and W, while E records the observer-expansion ledger and B records the residue boundary left outside the visible window.

Transport and replay preserve those rows componentwise. Provenance and local naming then bind the same finite observation surface as a $\mathsf{FiniteObservationTopology}^{\uparrow}$ certificate. Since no completed observer space, selected far-end point, quotient topology, global compactification, or unledgered residue row occurs in the carrier, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: No accepted $\mathsf{FiniteObservationTopology}^{\uparrow}$ consumer can cite observer-expansion topology unless it first exposes the StreamName observation source, topology row, observer-expansion ledger, residue-boundary row, finite topology-window ledger, transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: finiteobservationresidueboundary, sierpinski-space, topology, streamname: no listed sibling is bijective with $\mathsf{FiniteObservationTopology}^{\uparrow}$ because this carrier alone binds finite observation windows, observer-expansion ledger rows, residue-boundary data, topology-window tests, transport, replay, provenance, and local NameCert rows into one accepted packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationTopology}^{\uparrow}$ packet is generated from StreamName finite observation windows, a Topology row, an observer-expansion ledger, a residue-boundary row, finite topology-window tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteObservationTopology}^{\uparrow}}$, seed)

Scope closed: Definition 6831.1 and Definition 6831.2 seed the finite observer-expansion topology surface over observation, topology, residue-boundary, and provenance rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteObservationTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a completed observer space, a selected far-end point, quotient topology on unseen observations, global boundary compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite observation source exposure, topology-window exactness, observer-expansion ledger stability, residue-boundary nonescape, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.FiniteObservationTopologyUp.

Chapter 6832

CantorModulusContinuity name certificate construction

Following Brouwer and Bishop–Bridges, the $\mathsf{CantorModulusContinuity}^{\uparrow}$ packet records the constructive compact-prefix fact that a continuous map out of Cantor space is read through a finite inspected prefix before any modulus consumer acts. It sits beside the Cantor-space carrier of chapter 1475, the stream naming surface of chapter 126, the continuous-map row of chapter 130, the uniform-modulus route of chapter 369, and the compact-metric source of chapter 129. The packet is a finite certificate for compact-prefix continuity; it does not assert open-cover compactness, choose a bar-recursion branch, import countable choice, or identify host quotient function spaces.

Definition 6832.1 (Cantor modulus-continuity carrier)**.** *A* $\mathsf{CantorModulusContinuity}^{\uparrow}$ *carrier is a finite row*

$$K = (B, F, P, M, S, U, H, C, G, N),$$

where B *is the accepted* $\mathsf{CantorSpace}^{\uparrow}$ *source row,* F *is a* $\mathsf{ContinuousMap}^{\uparrow}$ *row out of* B*,* P *is the inspected prefix ledger,* M *is the finite modulus bound,* S *is the* $\mathsf{StreamName}^{\uparrow}$ *readback row,* U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* G *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{CantorModulusContinuity}^{\uparrow}}$ *row.*

Theorem 6832.2 (Cantor modulus-continuity classifier)**.** *An accepted* $\mathsf{CantorModulusContinuity}^{\uparrow}$ *packet classifies exactly those finite routes in which the continuous-map read* F *from the Cantor-space row* B *is exhausted by the displayed prefix ledger* P*, and the resulting modulus bound* M *is handed to the* $\mathsf{UniformModulus}^{\uparrow}$ *row* U *through the same stream row* S*. It rejects any route that obtains uniformity from open-cover compactness, a selected infinite cover, a host fan theorem oracle, quotient equality of functions, or ambient real-completion data.*

Proof. Read the source through chapter 1475. The Cantor-space row supplies finite binary prefix windows, so a continuity read can be inspected as a finite prefix request rather than as a host compact space.

The continuous-map and stream rows then determine the route F, P, S. The prefix ledger records which finite windows were read, and the modulus row M is the first carrier coordinate allowed to summarize those reads for downstream use.

Finally pass through U, H, C, G, N. The uniform-modulus handoff, transport, replay, provenance, and local name rows preserve the same finite packet, so no excluded compactness or choice source can be inserted without changing the carrier. □

Theorem 6832.3 (Cantor modulus-continuity NameCert obligations)**.** *For every accepted carrier $K = (B, F, P, M, S, U, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorModulusContinuity}^{\uparrow}}$ *row verifies the following five obligations:*

1. *the carrier binds one* $\mathsf{CantorSpace}^{\uparrow}$ *source row B, one* $\mathsf{ContinuousMap}^{\uparrow}$ *row F, one prefix ledger P, one modulus row M, and one* $\mathsf{StreamName}^{\uparrow}$ *row S;*
2. *the classifier accepts only routes where P records finite prefix windows read from B before M is exported;*
3. *the exactness row sends the same modulus bound M to the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff U and to no other modulus source;*
4. *the transport and replay rows H, C preserve the prefix ledger, continuous-map read, and modulus handoff componentwise;*
5. *the provenance and local naming rows G, N expose the route as finite BEDC NameCert data and not as open-cover compactness, countable choice, quotient function equality, or ambient completion.*

Proof. The carrier obligation is a projection from Definition 6832.1. It lists the rows that may be read by a consumer and leaves no coordinate for a host compactness theorem.

For classifier and exactness, use Definition 6832.2. It fixes the order B, F, P, M, S, U and makes M the finite summary of prefix inspection before the uniform-modulus handoff.

The remaining obligations are structural. The rows H, C, G, N transport, replay, source, and name the same finite route, so any attempted replacement by a selected cover, quotient equality, or external completion datum changes the packet rather than satisfying the local certificate. □

Theorem 6832.4 (Cantor modulus-continuity compact-prefix nonescape)**.** *Every accepted* $\mathsf{CantorModulusContinuit}$ *carrier $K = (B, F, P, M, S, U, H, C, G, N)$ exposes compact-prefix continuity only through the finite route $B, F, P, M, S, U, H, C, G, N$. A downstream* $\mathsf{UniformModulus}^{\uparrow}$ *consumer may read the modulus bound M through U, but it cannot split the read into open-cover compactness, selected subcovers, countable choice, quotient equality of continuous functions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 6832.3. They bind B, F, P, M, S, U as one finite prefix-to-modulus route.

The structural rows H, C, G, N only preserve that route. Since the carrier has no coordinate for open-cover compactness, selected subcovers, quotient function equality, or ambient completion, a consumer that reads any such object has left the accepted packet. □

Falsifiable prediction: Any accepted $\mathsf{CantorModulusContinuity}^{\uparrow}$ packet that exports a $\mathsf{UniformModulus}^{\uparrow}$ row without first exhibiting the shared Cantor-space prefix ledger P and StreamName row S refutes this carrier.

Independence witness: cantorspace, streamname, continuousmap, uniformmodulus, compactmetric: no listed sibling is bijective with $\mathsf{CantorModulusContinuity}^{\uparrow}$, because this packet alone binds the Cantor source, continuous-map read, finite prefix ledger, modulus bound, StreamName readback, UniformModulus handoff, transport, replay, provenance, and local naming into one compact-prefix certificate.

Lean4 (statement) `BEDC.Derived.CantorModulusContinuityUp.CantorModulusContinuityUp`

Closure status for $\mathsf{NameCert}_{\mathsf{CantorModulusContinuity}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorModulusContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6832.1, Definition 6832.2, Definition 6832.3, and Definition 6832.4 seed the finite compact-prefix modulus carrier and its consumer boundary.

Constructive story: A CantorModulusContinuity$^{\uparrow}$ packet is generated from a CantorSpace source row, a ContinuousMap row, finite prefix ledger, modulus bound, StreamName readback, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Formal verification: FormalStatus(NameCert$_{\text{CantorModulusContinuity}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CantorModulusContinuityUp.CantorModulusContinuityUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, fan theorem variants, quotient equality of maps, selected subcovers, or ambient real-completion principles.

Upgrade path: Obligation closure requires checked carrier admission, prefix-ledger exactness, StreamName readback stability, UniformModulus handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorModulusContinuityUp

Chapter 6833

MetaCICClosedNormalJoinCertificate name certificate construction

The $\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}$ packet realizes the unconditional fragment of chapter 9378 as finite NameCert data. It consumes the normalization frontier of chapter 892 before exposing its local join ledger, so the closed-normal joins are routed through the sibling normalization lattice rather than floated as an isolated confluence row. It records closed-normal joins, closed sort joins, impossible closed-variable cases, atom-shape inversions, and the retained substitution-compatibility boundary without promoting those fragments to arbitrary closed Church–Rosser.

Definition 6833.1 (MetaCIC closed-normal join carrier)**.** *A* $\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}$ *carrier is a finite row*

$$J = (T, N, A, S, V, B, L, O, H, C, G, R),$$

where T is the closed MetaCIC term-window row, N is the closed-normal source row, A is the atom-shape row, S is the closed sort-join row, V is the impossible closed-variable row, B is the named substitution-boundary row, L is the local join ledger, O is the obstruction-retention row, H is componentwise hsame *transport, C is displayed* Cont *replay, G is* Pkg *provenance, and R is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}}$ *row.*

Theorem 6833.2 (MetaCIC closed-normal join classifier)**.** *An accepted* $\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}$ *packet classifies exactly the checked closed-normal, closed-sort, closed-variable-impossible, and atom-shape confluence fragments named in chapter 9378. It refuses any packet that drops the displayed substitution boundary B, replaces the local join ledger L by a full* BetaParallelSubstitutionCompatibility *proof, or claims arbitrary closed Church–Rosser for* BetaStarStep *without that boundary.*

Proof. Read the confluence audit map. Its established section names closed normal confluence, closed sort confluence, closed variable impossibility, atom joins, and atom-shape inversions as available fragments.

The conditional section names the substitution-sensitive boundary. The carrier therefore keeps B as a row rather than erasing it when the local fragments N, A, S, V, L are packaged.

Transport, replay, provenance, and local naming preserve this separation. Thus the accepted classifier is finite and fragmentary: it is a join certificate for displayed closed-normal and atom sources, not a full closed Church–Rosser theorem. □

Theorem 6833.3 (MetaCIC closed-normal join NameCert obligations)**.** *For every accepted carrier $J = (T, N, A, S, V, B, L, O, H, C, G, R)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}}$ *row verifies the following five obligations:*

1. *the carrier binds closed term windows, closed-normal sources, atom-shape rows, sort joins, impossible closed-variable rows, substitution-boundary data, and local join ledgers as distinct fields;*
2. *the classifier accepts the unconditional closed-normal and atom fragments while preserving the conditional full-confluence boundary named by chapter 9378;*
3. *exactness requires L to cite only joins licensed by N, A, S, V and to retain B whenever a consumer asks for a stronger closed-source confluence claim;*
4. *the obstruction row O records that arbitrary closed Church–Rosser is outside the packet until the substitution boundary is discharged;*
5. *the transport, replay, provenance, and local-name rows H, C, G, R preserve the same fragmentary carrier for downstream proof-obligation consumers.*

Proof. Project the carrier of Definition 6833.1. The rows are separate fields, so the substitution boundary cannot be hidden inside the local join ledger.

Apply the classifier theorem. It reads the established closed-normal and atom fragments from the audit map and keeps the conditional full-confluence interface visible as B.

The exactness, obstruction, and structural obligations follow by preserving that same row order through O, H, C, G, R. A consumer that claims arbitrary closed Church–Rosser without the boundary has changed the packet rather than read this certificate. □

Theorem 6833.4 (MetaCIC closed-normal join boundary retention)**.** *Every accepted* MetaCICClosedNormalJoinCer *packet exports closed-normal and atom confluence data only along the route $T, N, A, S, V, B, L, O, H, C, G, R$. The packet may feed proof-obligation consumers that need the unconditional fragments, but any consumer that asks for arbitrary closed Church–Rosser must still supply the displayed substitution-compatibility boundary.*

Proof. Use Definition 6833.3. The local join ledger L is licensed by the closed-normal, atom, sort, and impossible-variable rows, and it is paired with the boundary row B.

The obstruction row O records the non-escape condition. The structural rows then replay the same route without creating a full confluence theorem, so the stronger arbitrary closed-source claim remains conditional. □

Falsifiable prediction: Any MetaCICClosedNormalJoinCertificate$^{\uparrow}$ packet that claims arbitrary closed Church–Rosser for BetaStarStep without the displayed substitution boundary B refutes this carrier.

Independence witness: normalformconsistencyseal, metaciccriticalpath, subject-reduction-discharge-socket, metacicconfluenceauditwitness, metacicconfluencenormalfragment: no listed sibling is bijective with MetaCICClosedNormalJoinCertificate$^{\uparrow}$, because this packet jointly carries closed term windows, closed-normal joins, atom-shape rows, sort and impossible-variable fragments, the substitution boundary, the local join ledger, obstruction retention, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit Nontrivial MetaCICClosedNormalJoinCertificate$^{\uparrow}$ and FieldFaithful MetaCICClosedNormalJoinCertificate$^{\uparrow}$ TasteGate instances. Those instances must preserve $T, N, A, S, V, B, L, O, H, C, G, R$ as distinct rows before any checked carrier-alignment theorem can serve as the public TasteGate witness.

Lean4 (statement) `BEDC.Derived.MetaCICClosedNormalJoinCertificateUp.MetaCICClosedNormalJoinCertificateUp`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6833.1, Definition 6833.2, Definition 6833.3, and Definition 6833.4 seed the closed-normal and atom confluence fragment carrier with substitution-boundary retention.

Constructive story: A $\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}$ packet is generated from closed term windows, closed-normal source rows, atom-shape rows, sort-join rows, impossible closed-variable rows, a substitution-boundary row, local join ledger, obstruction-retention row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosedNormalJoinCertificate}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MetaCICClosedNormalJoinCertificateUp.MetaCICClosedNormalJoinCertificateUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary closed Church–Rosser, BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, subject reduction, or full MetaCIC confluence.

Upgrade path: Obligation closure requires checked carrier admission, fragment classifier exactness, local join ledger stability, substitution-boundary retention, TasteGate nontriviality, field faithfulness, replay, provenance, and local naming for BEDC.Derived.MetaCICClosedNormalJoinCertificateUp.

Chapter 6834

A Concrete Naming Certificate for MetaCICSubjectReductionSocketCut$^{\uparrow}$

Following the standard subject-reduction proof-obligation discipline for dependent type theory, the MetaCICSubjectReductionSocketCut$^{\uparrow}$ packet records the exact finite cut separating the parameterised MetaCIC subject-reduction theorem from the blocked closed theorem. It sits beside the four-row discharge socket of chapter 719, the obligation packet of chapter 712, the trace packet of chapter 723, and the audit map of chapter 9377. The packet names the four discharge sockets, the parameterised theorem row, the checked obstruction row, transport, replay, provenance, and local naming; it does not close subject reduction.

Definition 6834.1 (MetaCIC subject-reduction socket-cut carrier)**.** *A* MetaCICSubjectReductionSocketCut$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, A, L, P, T, O, H, C, G, N)$$

with the following displayed rows:

1. *B is the beta-substitution socket, carrying* BetaSubstitutionPreservation *for the beta-redex case.*
2. *A is the application-argument socket, carrying* AppArgTypeStable *for argument congruence.*
3. *L is the lambda-domain socket, carrying* LamDomainSubjectReduction *for lambda-domain reduction.*
4. *P is the Pi-domain socket, carrying* PiDomainSubjectReduction *for Pi-domain reduction.*
5. *T is the parameterised* BEDC.MetaCIC.subject_reduction *row, readable only as consuming the four sockets B, A, L, P.*
6. *O is the checked obstruction row* BEDC.MetaCIC.closed_consistency_reduction_obstruction_not_available.
7. *H records componentwise* hsame *transport for the four sockets, the parameterised row, the obstruction row, provenance, replay, and local naming.*
8. *C records displayed* Cont *replay from socket reads to the parameterised row and from closed-theorem requests to the obstruction row.*
9. *G is the* Pkg *provenance row over MetaCIC typed reduction,* SubjectReduction$^{\uparrow}$*, the discharge socket, the obstruction audit,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{MetaCICSubjectReductionSocketCut}^{\uparrow}}$ *row.*

The carrier contains no unparameterised closed subject-reduction theorem, no host conversion rule, no erased derivation semantics, no quotient equality of MetaCIC terms, and no hidden normalization principle.

Theorem 6834.2 (MetaCIC subject-reduction socket-cut obligations)**.** *For an accepted* MetaCICSubjectReductionS *carrier $S = (B, A, L, P, T, O, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionSocketCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the four discharge sockets, parameterised-row exactness for T, obstruction-row exactness for O, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N. A consumer may read T only after the four sockets B, A, L, P have all been displayed; a consumer asking for the closed theorem is routed to O.*

Proof. Project the accepted carrier. The four visible discharge sockets are B, A, L, P, and the only parameterised subject-reduction row is T.

Read the socket discipline through chapter 719. That packet already separates the four inputs consumed by parameterised subject reduction from the closed obstruction boundary.

Now inspect T and O. The row T records the checked parameterised theorem as a consumer of all four sockets, while O records the checked obstruction described in Definition 9377.5. These rows are different coordinates, so the obstruction row cannot be used as a substitute proof of closed subject reduction.

Finally H, C, G, N transport, replay, source, and name the same cut. Since no coordinate stores host conversion, erased derivation semantics, quotient equality, global normalization, or an unparameterised closed theorem, the listed rows exhaust the NameCert surface. □

Theorem 6834.3 (MetaCIC subject-reduction socket-cut non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetaCICSubjectReductionSocketCut}^{\uparrow}$ *packet factors through*

$$B, A, L, P, T, O, H, C, G, N.$$

The consumer receives the four discharge sockets, the parameterised theorem row, the checked obstruction row, componentwise hsame *transport, displayed* Cont *replay, provenance, and local naming. It cannot export a closed subject-reduction theorem, host conversion, erased derivation semantics, quotient equality of MetaCIC terms, or an unrecorded MetaCIC premise.*

Proof. Start from the row inventory of Definition 6834.1. The packet has exactly ten displayed coordinates, so every public read is routed through $B, A, L, P, T, O, H, C, G, N$.

For a parameterised subject-reduction read, apply Definition 6834.2. The route reaches T only after the beta, application-argument, lambda-domain, and Pi-domain sockets have all been exposed and transported.

For a closed subject-reduction request, the same theorem routes the consumer to O. This row is the checked obstruction boundary, not a closed preservation proof, and its replay through C records refusal rather than theorem export.

The provenance row G cites only the displayed MetaCIC audit and socket surfaces. With no carrier coordinate for host conversion, erased derivation semantics, quotient equality, global normalization, or hidden premises, none of those objects can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionSocketCut}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICSubjectReductionSocketCut}^{\uparrow}$ packet is generated from the four MetaCIC subject-reduction discharge sockets, the parameterised subject-reduction theorem row, the checked obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICSubjectReductionSocketCut}\uparrow}$, seed)

Scope closed: Definition 6834.1, Definition 6834.2, and Definition 6834.3 seed the finite socket-cut packet separating parameterised subject reduction from the blocked closed theorem.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICSubjectReductionSocketCut}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, host conversion, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-socket exposure, parameterised-row exactness, obstruction-row exactness, transport replay, provenance accounting, and non-escape for BEDC.Derived.MetaCICSubjectReductionSocket

Chapter 6835

A Concrete Naming Certificate for $\mathsf{RegularOpenAlgebra}^{\uparrow}$

Following Stone and Birkhoff on Boolean algebras of regular open sets, the $\mathsf{RegularOpenAlgebra}^{\uparrow}$ packet records the topology-facing Boolean algebra obtained by regularizing displayed opens through closure and interior. It sits beside the indexed-open topology of chapter 88, the finite closure rows of chapter 2145, the Boolean operation packet of chapter 3733, the generated-set consumer of chapter 3384, and the clopen handoff of chapter 261. The chapter names the regular-open algebra as finite NameCert data; it does not import a host powerset algebra, ultrafilter space, choice principle, or completed locale.

Definition 6835.1 (RegularOpenAlgebra carrier)**.** *A* $\mathsf{RegularOpenAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, C, I, K, J, M, H, R, P, N)$$

with the following displayed rows:

1. *O is the indexed open-set row read through* $\mathsf{Topology}^{\uparrow}$*.*
2. *C is the closure row read through the Kuratowski closure surface.*
3. *I is the interior row, displayed as the topology-facing open extractor paired with C.*
4. *K is the regularization row $I(C(O))$, naming the regular open representative obtained from the displayed open row.*
5. *J is the Boolean join row, read as regularized union on displayed regular-open rows.*
6. *M is the Boolean meet row, read as displayed intersection on regular-open rows.*
7. *H records Boolean complement as interior of the displayed closed complement boundary, transported only through the topology and closure rows already present.*
8. *R records componentwise* hsame *transport for open, closure, interior, regularization, Boolean operation, complement, provenance, replay, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularOpenAlgebra}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{BooleanAlgebra}^{\uparrow}$*, Kuratowski closure, Borel generated-set,* $\mathsf{StoneDuality}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularOpenAlgebra}^{\uparrow}}$ *row.*

The carrier contains no host powerset of opens, arbitrary complement oracle, selected ultrafilter, quotient locale, ambient Borel universe, or external Stone representation theorem.

Theorem 6835.2 (RegularOpenAlgebra NameCert obligations)**.** *For an accepted* $\mathsf{RegularOpenAlgebra}^{\uparrow}$ *carrier* $A = (O, C, I, K, J, M, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularOpenAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, topology-row exposure through* O*, closure exposure through* C*, interior and regularization exactness through* I, K*, Boolean operation exposure through* J, M, H*, componentwise transport through* R*, provenance and replay through* P*, and local naming through* N*. Every topology, Boolean-algebra, Borel-facing, or Stone-duality consumer must read those rows before treating a displayed open as a regular-open algebra element.*

Proof. Project the accepted carrier. The visible inventory is exactly $O, C, I, K, J, M, H, R, P, N$, with no additional host set or algebra coordinate.

Read the topology and closure rows first. The row O supplies the indexed open surface, while C supplies the closure route already recorded by the Kuratowski closure packet. The interior row I is paired with the same topology surface, so the regularization row $K = I(C(O))$ is a displayed finite route rather than a powerset operation.

Read the Boolean rows next. Join J, meet M, and complement H are accepted only after regularization has exposed the relevant open, closure, and interior rows. Thus the Boolean-algebra consumer receives the finite complemented distributive-lattice interface of chapter 3733 through topology-facing data.

Finally R, P, N transport, source, replay, and name the same row inventory. Since no row stores host powersets, arbitrary complements, ultrafilters, quotient locales, completed Borel universes, or ambient Stone representation data, the listed obligations exhaust the local NameCert surface. □

Theorem 6835.3 (RegularOpenAlgebra closure-interior handoff)**.** *Every consumer of an accepted* $\mathsf{RegularOpenAlgebra}^{\uparrow}$ *packet that reads a regular-open element must factor through the finite handoff*

$$\mathsf{Topology}^{\uparrow},\ C,\ I,\ K,\ \mathsf{BooleanAlgebra}^{\uparrow},\ \mathsf{StoneDuality}^{\uparrow},\ R, P, N.$$

The handoff first reads the indexed open row, then the closure row, then the interior row, then the regularization row $I(C(O))$*, and only then exposes Boolean meet, join, and complement rows for Stone-duality or Borel-facing consumers. It exports no powerset algebra, selected ultrafilter, quotient locale, arbitrary complement choice, or ambient Borel completion.*

Proof. Start from the regularization row K. By carrier admission, K is readable only after the open row O, closure row C, and interior row I have been displayed.

Use Definition 6835.2. It forces Boolean join, meet, and complement reads to pass through the regularization route before a Boolean-algebra or Stone-duality consumer receives the packet.

The Borel-facing route is only a generated-set consumer route: it can observe the displayed regular open through the same topology and regularization rows, but it receives no completed Borel universe. The Stone-duality route likewise receives the Boolean rows after regularization, not an ultrafilter or point-space construction.

The structural rows R, P, N preserve and name this handoff. Since the carrier has no coordinate for host powersets, selected ultrafilters, quotient locales, arbitrary complement choice, or ambient Borel completion, those objects cannot be exported by the closure-interior route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularOpenAlgebra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularOpenAlgebra}^{\uparrow}$ packet is generated from an indexed open row, closure row, interior row, regularization row, Boolean join, meet, and complement rows, componentwise hsame transport, provenance and replay rows, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularOpenAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6835.1, Definition 6835.2, and Definition 6835.3 seed the finite regular-open algebra packet over topology, closure, interior, regularization, Boolean operation, transport, provenance, replay, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularOpenAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host powerset algebras, ultrafilter construction, quotient locales, arbitrary complement choice, completed Borel universes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closure-interior regularization exactness, Boolean operation exposure, topology and Stone-duality handoff, transport stability, provenance replay, and non-escape for BEDC.Derived.RegularOpen

Chapter 6836

A Concrete Naming Certificate for $\mathsf{HeineCantorFiniteModulus}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{HeineCantorFiniteModulus}^{\uparrow}$ packet records the finite-modulus route by which compact metric continuity becomes a uniform-modulus read without importing an open-cover compactness theorem. It sits beside the compact metric source of chapter 129, the continuous-map pointwise rows of chapter 130, the finite-net route of chapter 6769, the finite lower-bound fold of chapter 4096, the $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369, and the L10 readback boundary of chapter 126, chapter 301, and chapter 38. The certificate names the finite rows that a compact-uniform-continuity consumer may inspect; it does not select a host modulus or identify quotient functions.

Definition 6836.1 (Heine-Cantor finite-modulus carrier)**.** *A* $\mathsf{HeineCantorFiniteModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, N, J, L, U, W, R, E, H, C, P, Z)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row displaying the compact metric carrier and its finite-net coverage surface.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus data are read only at the centers exposed by K and N.*
3. *N is the compact finite-net selector row, using the same finite center bundle and coverage ledger as* $\mathsf{HeineCantorFiniteNet}^{\uparrow}$*.*
4. *J is the pointwise modulus ledger pairing each center in N with the corresponding local continuity radius and target tolerance row from F.*
5. *L is the* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *row that folds the finitely many displayed lower-bound candidates in J into one positive precision row.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row reached only after K, F, N, J, L have been displayed.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *readback window opened for the source and target values used by the uniform-modulus consumer.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the observations in W.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite-net, fold, uniform-modulus, stream-window, and regular-rational rows have been consumed.*

10. *H records componentwise* hsame *transport for the compact source, continuous-map row, finite-net selector, pointwise ledger, fold, uniform-modulus consumer, L10 readback rows, replay row, provenance row, and local naming row.*
11. *C records displayed* Cont *replay from a target-tolerance request through the finite-net selector, pointwise modulus ledger, finite fold, uniform-modulus consumer, and L10 readback route.*
12. *P is the* Pkg *provenance over* $\mathsf{HeineCantorFiniteModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantorFiniteNet}^{\uparrow}$, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *Z is the local* $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, a host-selected modulus, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6836.2 (Heine-Cantor finite-modulus NameCert obligations)**.** *For every accepted* HeineCantorFiniteModu *carrier $M = (K, F, N, J, L, U, W, R, E, H, C, P, Z)$, the local* $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, compact-net exposure through K, N, pointwise-to-uniform modulus handoff through F, J, L, U, L10 readback nonescape through W, R, E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through Z.*

Proof. Project the carrier of Definition 6836.1. The analytic rows are the compact source K, continuous-map row F, finite-net selector N, pointwise ledger J, fold row L, uniform-modulus consumer U, and readback rows W, R, E.

Read the compact-net exposure first. The selector row N uses only the finite centers and coverage ledger supplied by $\mathsf{CompactMetric}^{\uparrow}$ and the sibling $\mathsf{HeineCantorFiniteNet}^{\uparrow}$ route, so no open-cover compactness or host-selected cover is present.

Next pass from pointwise continuity to a uniform modulus. The ledger J pairs each displayed center with its pointwise modulus row from F, and the fold row L is the finite lower-bound fold consumed by U.

Finally attach the L10 readback route. The rows W, R, E expose StreamName windows, RegSeqRat readback, and the terminal Real seal only after the finite-modulus route has been displayed, while H, C, P, Z transport, replay, source, and name the same packet. Thus the listed obligations exhaust the local certificate surface and export no host modulus selection, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineCantorFiniteModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HeineCantorFiniteModulus}^{\uparrow}$ packet is generated from a CompactMetric source row, ContinuousMap pointwise modulus rows, a compact finite-net selector, a pointwise modulus ledger, a FiniteLowerBoundFold row, a UniformModulus consumer row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HeineCantorFiniteModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6836.1 and Definition 6836.2 seed the finite-modulus route from compact metric pointwise continuity to a UniformModulus consumer with L10 readback rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HeineCantorFiniteModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host modulus selection, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, pointwise-to-uniform modulus handoff, L10 readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HeineCantorFiniteModulusUp.

Chapter 6837

A Concrete Naming Certificate for $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$

Following Sobolev, Poincare, Brezis, and Evans, the $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$ packet records the finite-energy handoff by which a displayed Sobolev packet and a displayed Poincare-control packet jointly expose bounded-domain energy data. It sits over the weak-derivative carrier of chapter 440, the norm-control estimate of chapter 2329, the energy ledger of chapter 5956, the rational readback route of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38. The packet names a finite energy-control certificate; it does not import Hahn–Banach separation, measure completion, selected representatives, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6837.1 (Sobolev–Poincare finite-energy carrier)**.** *A* $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, Q, D, W, Y, A, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative source row whose function, domain, codomain, magnitude, and derivative-window data are exposed before any energy read.*
2. *Q is the* $\mathsf{PoincareInequality}^{\uparrow}$ *control row, recording the bounded-domain, boundary, or zero-mean route used to compare the displayed function magnitude with its derivative magnitude.*
3. *D is the bounded domain or bounded inspection window row through which the Poincare control is read.*
4. *W is the dyadic tolerance schedule supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$ *for the finite energy comparison.*
5. *Y is the* $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$ *row that stores the displayed derivative-energy and magnitude ledgers.*
6. *A is the dyadic accounting row, recording the finite rational inequalities consumed by the control estimate.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational energy and norm-control observations.*
8. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Sobolev, Poincare, dyadic, and rational readback rows have been displayed.*

9. *H records componentwise* hsame *transport for S, Q, D, W, Y, A, R, Z and their dependency rows.*
10. *C records displayed* Cont *replay from a finite energy request through Sobolev exposure, Poincare control, dyadic accounting, rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, ConstructiveSobolevEn $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}}$ *row.*

The carrier has no coordinate for host measure completion, quotient equality of functions, selected representatives, global compactness, Hahn–Banach separation, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6837.2 (Sobolev–Poincare finite-energy NameCert obligations)**.** *For every accepted* $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$ *carrier $E = (S, Q, D, W, Y, A, R, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SobolevPoincareFiniteEn}}$ *surface consists exactly of carrier admission, Sobolev weak-derivative exposure, Poincare control exposure, bounded-domain or finite-window admission, dyadic tolerance accounting, constructive energy readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.SobolevPoincareFiniteEnergyUp.TasteGate.SobolevPoincareFiniteEnergyTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 6837.1. The analytic rows are S, Q, D, W, Y, A, R, Z, and the structural rows are H, C, P, N.

The Sobolev row S supplies only the displayed weak-derivative and magnitude surface of chapter 440. The Poincare row Q supplies only the bounded-domain or zero-mean control route of chapter 2329. Thus the energy-control handoff is read through displayed Sobolev and Poincare rows before any downstream energy consumer receives it.

The dyadic schedule W and accounting row A keep every tolerance comparison inside $\mathsf{DyadicRatCore}^{\uparrow}$. The constructive energy row Y and rational readback row R then route the observed bounds through chapter 5956 and chapter 301.

Finally the Real seal Z is downstream of those rows, and H, C, P, N preserve the same packet under transport, replay, provenance, and naming. Since the carrier omits measure completion, quotient equality, selected representatives, Hahn–Banach separation, and Real completeness, no accepted local NameCert route exports them. □

Theorem 6837.3 (Sobolev–Poincare finite-energy handoff)**.** *Every PDE, variational, trace, or metric consumer that reads an accepted* $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$ *packet must traverse*

$$\mathsf{Sobolev}^{\uparrow},\ \mathsf{PoincareInequality}^{\uparrow},\ \mathsf{ConstructiveSobolevEnergy}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $S, Q, D, W, Y, A, R, Z, H, C, P, N$. The handoff exports a finite energy-control route and nothing outside those rows.

Proof. Apply Definition 6837.2. It makes Sobolev exposure, Poincare control, bounded-window admission, dyadic accounting, constructive energy, rational readback, and the Real seal first-class obligations of one packet.

A downstream consumer can therefore read a finite energy-control datum only after these rows have been exposed in order. Omitting the Sobolev row loses the weak-derivative source; omitting the Poincare row loses the control estimate; omitting the dyadic and rational rows loses the finite numerical accounting; and omitting the seal loses the terminal Real boundary.

The structural rows H, C, P, N keep that route stable under transport, replay, provenance, and naming. Thus the exported handoff is exactly the displayed finite route, not a completed Sobolev space theorem or an external separation principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}}$.

Constructive story: A $\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}$ packet is generated from Sobolev weak-derivative rows, Poincare control rows, bounded-domain or finite-window rows, dyadic tolerance rows, constructive energy rows, rational readback rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6837.1, Definition 6837.2, and Definition 6837.3 seed the finite Sobolev–Poincare energy-control route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobolevPoincareFiniteEnergy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hahn–Banach separation, measure completion, quotient equality of functions, selected representatives, global compactness, Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sobolev exposure, Poincare control, bounded-window exactness, dyadic accounting, rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SobolevPoincareFiniteEnergyUp.

Chapter 6838

A Concrete Naming Certificate for $\mathsf{PartialityBoundary}^{\uparrow}$

Following the total-functional-language discipline of CIC, Coq, Agda, Lean, and Idris totality checking, the $\mathsf{PartialityBoundary}^{\uparrow}$ packet records the boundary between total host computation and explicitly marked partial escape as finite BEDC NameCert data. It concretizes the total-language analysis of chapter 9370, reads local channel termination through chapter 3996, and keeps refusal-to-decide separate through chapter 1197. The packet does not admit a partial-function surface inside BEDC; it records the attempted escape as a refusal row.

Definition 6838.1 (Partiality-boundary carrier)**.** *A* $\mathsf{PartialityBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, F, A, E, R, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the total-host row, recording the typed BEDC term or certificate route whose reduction is accepted only as a finite total computation.*
2. *F is the step-indexed computation row, exposing finite fuel and the displayed result-or-timeout observation for the inspected route.*
3. *A is the attempted partial escape row, naming the external request for unrestricted recursion, a partial annotation, or a non-total host function.*
4. *E is the termination-boundary row, read through chapter 3996 when a local event endpoint is displayed.*
5. *R is the refusal ledger, read through chapter 1197 when the attempted escape has no admitted total route.*
6. *L is the totality classifier row, accepting only routes that remain in T, F, E or are recorded as refusal in R.*
7. *H records componentwise* hsame *transport for total-host rows, fuel rows, attempted escape rows, termination-boundary rows, refusal rows, classifier rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from a host computation request through total reduction, step-indexed fuel, local endpoint exposure, attempted escape inspection, and refusal.*
9. *P is the* Pkg *provenance over* $\mathsf{PartialityBoundary}^{\uparrow}$*, chapter 9370,* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$*,* $\mathsf{TerminationRefusalBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{PartialityBoundary}^{\uparrow}}$ *row.*
The carrier has no coordinate for a BEDC partial function, unrestricted recursion, a divergence oracle, a total halting predicate, or a proof that a missing finite endpoint is nontermination.

Theorem 6838.2 (Partiality-boundary NameCert obligations)**.** *For every accepted* $\mathsf{PartialityBoundary}^{\uparrow}$ *carrier* $B = (T, F, A, E, R, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PartialityBoundary}^{\uparrow}}$ *surface consists exactly of total-host admission, step-indexed computation exposure, attempted-escape exposure, termination-boundary routing, refusal-ledger routing, totality classifier stability, componentwise transport, displayed replay, provenance, and local naming. The surface records the total-host versus partial-escape boundary but exports no partial BEDC function.*

Proof. Project the carrier of Definition 6838.1. The total computation rows are T, F, E, the attempted escape is A, the refusal boundary is R, and the structural rows are L, H, C, P, N.

The total-host row T and step-indexed row F keep every admitted computation finite. The termination-boundary row E may expose a local endpoint, but only through the channel termination boundary of chapter 3996.

If the request asks for unrestricted recursion or a partial annotation, the attempted escape row A is not converted into a BEDC function. It is routed to the refusal ledger R, whose role is the refusal-to-decide boundary of chapter 1197.

The classifier L accepts only those two outcomes: finite total exposure or displayed refusal. Transport, replay, provenance, and local naming preserve the same boundary, so no accepted route exports unrestricted recursion, a divergence oracle, a total halting predicate, or a nontermination proof. □

Theorem 6838.3 (Partiality-boundary total-host handoff)**.** *Every host-computation consumer that reads an accepted* $\mathsf{PartialityBoundary}^{\uparrow}$ *packet receives either a displayed total route through* T, F, E, L, H, C, P, N *or a displayed refusal route through* A, R, L, H, C, P, N*. These routes are disjoint at the local NameCert surface: the refusal route records the failed partial escape, while the total route records finite step-indexed computation.*

Proof. Use Definition 6838.2. The packet exposes the total-host rows, fuel row, termination-boundary row, attempted escape row, refusal row, classifier row, and structural rows as the whole local surface.

For a total route, the consumer reads T, F, E before the classifier L and the structural rows. This is a finite computation route in the total host.

For an attempted partial route, the consumer reads A and then R before the same classifier and structural rows. This is a refusal route, not a function definition.

Because no carrier row identifies A with a total endpoint or identifies R with divergence, the two handoffs stay separated inside the same finite NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartialityBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PartialityBoundary}^{\uparrow}$ packet is generated from total-host rows, step-indexed fuel rows, attempted partial-escape rows, termination-boundary rows, refusal-ledger rows, totality classifier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PartialityBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 6838.1, Definition 6838.2, and Definition 6838.3 seed the finite total-host versus partial-escape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PartialityBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not admit partial BEDC functions, unrestricted recursion, divergence oracles, a total halting predicate, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked total-host admission, step-indexed exposure, attempted-escape routing, termination-boundary routing, refusal-ledger exactness, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.PartialityBoundaryUp.

Chapter 6839

Sierpinski Continuity Classifier NameCert Construction

Following Sierpinski and standard constructive topology, the $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ packet records continuity through the finite observable-open classifier rather than through a host open-set predicate. It sits beside the Sierpinski observable truth carrier of chapter 1860, the finite observation topology of chapter 6831, the topology row discipline of chapter 88, and the continuous-map naming surface of chapter 130. The packet names the route by which a finite observation of an open predicate is read as a Sierpinski-valued classifier and transported through a continuous map; it does not install host open-set extensionality, quotient topology, open-cover compactness, or an ambient classical evaluator.

Definition 6839.1 (Sierpinski continuity classifier carrier)**.** *A* $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, O, S, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Topology}^{\uparrow}$ *row whose opens are inspected only through finite observation data.*
2. *Y is the target* $\mathsf{Topology}^{\uparrow}$ *row receiving the displayed open-predicate read.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording the map as a finite BEDC route from source observations to target observations.*
4. *O is the target open-predicate row under inspection.*
5. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *classifier row, turning the open-predicate read into displayed observable truth.*
6. *W is the* $\mathsf{FiniteObservationTopology}^{\uparrow}$ *window row, listing the finite observations through which O and S are tested.*
7. *H records componentwise* hsame *transport for source topology, target topology, continuous-map, open-predicate, Sierpinski classifier, finite-window, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a continuity-classifier request through X, Y, F, O, S, W, H.*
9. *P is the* Pkg *provenance over* $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{FiniteObservationTopology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiContinuityClassifier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, F, O, S, W, H, C, P, N$. It has no coordinate for a host open-set equality proof, a quotient topology on observations, an open-cover compactness principle, a hidden truth evaluator, or a detached classical continuity predicate.

Lean4 (definition) `BEDC.Derived.SierpinskiContinuityClassifierUp`

Theorem 6839.2 (Sierpinski continuity classifier NameCert obligations)**.** *For every accepted* $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ *carrier*

$$K = (X, Y, F, O, S, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiContinuityClassifier}^{\uparrow}}$ *surface consists exactly of source and target topology exposure through X, Y; continuous-map exposure through F; target open-predicate exposure through O; Sierpinski-valued observable truth through S; finite observation topology exposure through W; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier in Definition 6839.1. The public continuity-classifier rows are exactly $X, Y, F, O, S, W, H, C, P, N$.

The open-predicate read is represented by the Sierpinski-valued classifier row S. The finite observation row W determines which displayed windows may test the target open row O, and the continuous-map row F is the only route by which those observations are pulled back to the source topology row X.

Classifier stability is componentwise hsame transport through H. Transport preserves the topology rows, the continuous-map row, the open-predicate row, the Sierpinski classifier row, and the finite observation row before replay and provenance are consumed.

Since no row stores host open-set extensionality, quotient topology, open-cover compactness, hidden truth evaluation, or a detached classical continuity predicate, the listed rows exhaust the local NameCert obligations. □

Theorem 6839.3 (Finite observation route for Sierpinski continuity)**.** *Every finite-observation topology consumer of an accepted* $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ *carrier must read the target open row O as a* $\mathsf{SierpinskiSpace}^{\uparrow}$ *classifier row S, test it only through the finite windows W, and replay the pullback route through F, C, P, N. The consumer receives a displayed finite-observation continuity classifier; it receives no host open-set extensionality principle, no quotient topology on unseen observations, and no classical open-cover compactness claim.*

Proof. Read the carrier rows from Definition 6839.1. The only open-predicate information available to a consumer is the pair O, S, with S exposing observable truth in the Sierpinski packet.

Apply the finite observation row W. It restricts the continuity test to displayed windows, so the consumer cannot quantify over unseen opens or replace the observed classifier by a host predicate.

Transport through H is componentwise. The topology rows, continuous-map row, open row, Sierpinski classifier row, and finite-window row are preserved before the route is replayed by C and sourced by P, N.

Thus the route exposes a finite-observation classifier for continuity and nothing stronger. Host open-set extensionality, quotient topology, and classical open-cover compactness are outside the carrier. □

Falsifiable prediction: No accepted $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ consumer can cite continuity through finite observation topology unless it first exposes source and target topology rows, the

continuous-map row, the target open-predicate row, the Sierpinski classifier row, finite observation windows, transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: sierpinski-space, finite-observation-topology, topology, continuousmap: no listed sibling is bijective with $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ because this carrier alone binds a continuous-map row, a target open-predicate row, Sierpinski-valued observable truth, finite observation windows, componentwise transport, replay, provenance, and local NameCert rows into one accepted continuity-classifier packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiContinuityClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{SierpinskiContinuityClassifier}^{\uparrow}$ packet is generated from source and target Topology rows, a ContinuousMap row, a target open-predicate row, a SierpinskiSpace classifier row, finite observation topology windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiContinuityClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6839.1, Definition 6839.2, and Definition 6839.3 seed the finite-observation route from displayed open-predicate reads to Sierpinski-valued continuity classifiers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiContinuityClassifier}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SierpinskiContinuityClassifierUp`

Bridge status: none

Not claimed: This chapter does not close host open-set extensionality, quotient topology, open-cover compactness, selected limits, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sierpinski classifier exactness, finite-window stability, continuous-map pullback replay, componentwise transport, provenance, and local naming for BEDC.Derived.SierpinskiContinuityClassifier

Chapter 6840

A Concrete Naming Certificate for $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$

Following Bishop and Bridges on nested intervals, regular Cauchy reals, and constructive real approximation, the $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$ packet records the finite intersection route by which a displayed family of nested dyadic windows reaches a Real-facing seal. It sits after the dyadic tolerance surface of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the Cauchy source discipline of chapter 3413, and the terminal real-name boundary of chapter 38. The packet names a finite common-window intersection principle for Real and Completion consumers; it does not select a completed limit, quotient Cauchy names, invoke countable choice, decide trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6840.1 (Cauchy intersection principle carrier)**.** *A* $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the nested dyadic window row, recording finite left and right endpoint data over* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with the displayed inclusion checks between successive windows.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, recording the finite precision requests under which the displayed windows are certified Cauchy.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row exposing exactly the inspected window stages and no unseen tail.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite observations and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, M, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic-window, modulus, stream-observation, regular-readback, Real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an intersection request through D, M, S, R, E, H and back to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyIntersectionPrinciple}^{\uparrow}}$ *row.*
The carrier exports only this finite nested-window packet. It has no coordinate for selected limits, quotient equality of Cauchy names, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyIntersectionPrincipleUp.CauchyIntersectionPrincipleTasteGate single carrier alignment`

Theorem 6840.2 (Cauchy intersection principle NameCert obligations)**.** *For every accepted* CauchyIntersectionPrin *carrier $K = (D, M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntersectionPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed nested dyadic windows, Cauchy modulus requests, StreamName finite observations, RegSeqRat readback, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6840.1. The rows $D, M, S, R, E, H, C, P, N$ are the complete public packet, so carrier admission can only admit data already displayed in those rows.

The analytic route first reads D and M. Thus every intersection-facing consumer sees the nested dyadic windows and the finite Cauchy modulus requests before any stream window, regular readback, or Real seal is available.

Next read S and R. The stream row names the inspected finite stages, while the regular rational row supplies readback from the same finite observations and dyadic tolerance ledger.

Finally use E, H, C, P, N. The Real seal is terminal in this packet, and the transport, replay, provenance, and local naming rows preserve the same displayed route. Since no carrier row stores selected limits, quotient equality, countable choice, trichotomy, host real equality, or ambient completeness, none of those principles can be exported by the local certificate. □

Theorem 6840.3 (Cauchy intersection window extraction)**.** *Every completion-facing read of an accepted* $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the finite rows $D, M, S, R, E, H, C, P, N$. The consumer receives a displayed nested-window intersection route and a terminal Real seal; it receives no selected limit, quotient equality of Cauchy names, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the obligation theorem. It forces every completion-facing consumer to expose the dyadic window row and the Cauchy modulus row before the inspected stream stages can be read.

The sibling route then supplies the finite readback. DyadicRatCore supplies the endpoint and tolerance rows, StreamName supplies the inspected window stages, RegSeqRat supplies the regular rational readback, and the Real chapter supplies only the terminal seal.

The structural rows do not enlarge the route. Componentwise hsame transport and displayed Cont replay preserve the same packet, while P and N source and name it. Therefore the extracted window remains a finite BEDC packet rather than a selected completed limit or an ambient completeness theorem. □

Falsifiable prediction: Every accepted CauchyIntersectionPrincipleUp consumer exposes nested DyadicRatCore windows, a finite CauchyModulus schedule, StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before it can cite the intersection route.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy, real: removing any one of these sibling routes leaves the packet without the corresponding dyadic window, finite observation, regular readback, Cauchy modulus discipline, or terminal Real seal row.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntersectionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntersectionPrinciple}^{\uparrow}$ packet is generated from nested DyadicRatCore windows, a finite CauchyModulus schedule, StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntersectionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6840.1, Definition 6840.2, and Definition 6840.3 seed the finite nested-window intersection route from dyadic tolerance data to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntersectionPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy names, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window exactness, Cauchy modulus discipline, finite-observation readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyIntersectionPrincipleUp.

Chapter 6841

A Concrete Naming Certificate for $\mathsf{RegularCauchyClamp}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyClamp}^{\uparrow}$ packet records the finite regular-Cauchy route for clamping a regular rational Cauchy name between two regular rational endpoint rows before the terminal $\mathsf{Real}^{\uparrow}$ seal is inspected. It sits below the Real-facing clamp of chapter 1810 and reads its source surface through chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the lower-bound factorization of chapter 1832, the upper-bound factorization of chapter 1833, and the terminal real seal of chapter 38. The packet gives completion-facing interval, oscillation, and boundedness consumers a pre-seal regular-Cauchy clamp route without host trichotomy, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6841.1 (Regular Cauchy clamp carrier)**.** *A* $\mathsf{RegularCauchyClamp}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, L, U, W, D, M, N, R, E, H, C, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational Cauchy readback is being clamped.*
2. *L and U are the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows for the displayed lower and upper bounds.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the precision requested by the clamp consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback ledger; it records the requested dyadic tolerance, the endpoint slack, and the comparison scale used by the two factorization rows.*
5. *M is the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *factorization row implementing the lower projection* $\max(X, L)$ *on the displayed regular-Cauchy surface.*
6. *N is the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *factorization row implementing the upper projection* $\min(M, U)$ *after the lower projection has been exposed.*
7. *R is the exported* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular-Cauchy clamp value.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R has been displayed.*
9. *H records componentwise* hsame *transport for the source, endpoints, windows, dyadic ledger, maximum row, minimum row, readback, seal, replay, provenance, and local name.*

10. *C records displayed* Cont *replay from a clamp request through* $X, L, U, W, D, M, N, R, E, H, P, A$.
11. *P records* Pkg *provenance over* $\mathsf{RegularCauchyClamp}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyMaximum}^{\uparrow}$, $\mathsf{RegularCauchyMinimum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *A is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyClamp}^{\uparrow}}$ *row.*

The classifier reads only the displayed tuple $K = (X, L, U, W, D, M, N, R, E, H, C, P, A)$. It has no coordinate for host order trichotomy, quotient stream equality, selected limit, countable-choice schedule, ambient real completeness, Lean verification, or bridge checking.

Theorem 6841.2 (Regular Cauchy clamp NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyClamp}^{\uparrow}$ *carrier*

$$K = (X, L, U, W, D, M, N, R, E, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyClamp}^{\uparrow}}$ *surface consists exactly of carrier admission for the source row, endpoint rows, finite-window row, dyadic tolerance ledger, maximum factorization row, minimum factorization row, exported readback row, Real seal, componentwise transport, displayed replay, provenance, and local naming. The certificate also requires factorization order: M must be read as the displayed* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *route for* $\max(X, L)$ *before N is read as the displayed* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *route for* $\min(M, U)$.

Proof. Project the carrier of Definition 6841.1. Its displayed rows are exactly the source regular-rational readback, two endpoint rows, the finite observation window, dyadic tolerance ledger, two factorization rows, exported readback, Real seal, and structural rows.

The factorization order is part of the row inventory. The maximum row M can inspect only X, L, W, D and the structural rows available at that stage; the minimum row N is downstream of M and the upper endpoint row U.

The exported readback R and seal E have no independent numeric source. They can therefore expose only the regular-Cauchy clamp route already displayed by M and N, transported by H and replayed by C. □

Theorem 6841.3 (Regular Cauchy clamp min–max factorization)**.** *Every accepted* $\mathsf{RegularCauchyClamp}^{\uparrow}$ *carrier $K = (X, L, U, W, D, M, N, R, E, H, C, P, A)$ presents its clamp value through the finite route*

$$\begin{aligned} X, L, U, W, D &\longmapsto M = \max(X, L), \\ M, U, D &\longmapsto N = \min(M, U), \\ N &\longmapsto R \longmapsto E. \end{aligned}$$

No consumer receives a direct host clamp operation, a trichotomy of the three regular Cauchy rows, quotient stream equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier and read the maximum row M. By Definition 6841.1, this row is the only lower-projection coordinate and it factors through the displayed $\mathsf{RegularCauchyMaximum}^{\uparrow}$ route.

Next read the minimum row N. It is downstream of M and U, so its only upper-projection source is the displayed $\mathsf{RegularCauchyMinimum}^{\uparrow}$ route applied to the lower-projected row and the upper endpoint.

Finally read R and then E. Since the carrier has no coordinate for any excluded host or quotient object, the exported route is exactly the displayed min–max factorization. □

Theorem 6841.4 (Regular Cauchy clamp Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyClamp}^{\uparrow}$ *packet must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* X*, endpoint rows* L, U*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row* M*,* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row* N*, and exported* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*. The terminal seal* E *records only the displayed regular-Cauchy clamp boundary and cannot be used as an ambient Real-level clamp theorem.*

Proof. The Real seal row E is downstream of the exported readback row R in the carrier tuple. Thus a Real-facing consumer cannot inspect E until the regular-Cauchy readback has been displayed.

The readback row R is downstream of the two factorization rows. Those factorization rows force the route through $\mathsf{RegularCauchyMaximum}^{\uparrow}$ and $\mathsf{RegularCauchyMinimum}^{\uparrow}$ before any Real-facing seal is available.

The structural rows H, C, P, A transport, replay, source, and name only that route. Hence E seals the pre-seal regular-Cauchy clamp boundary and does not add host order, quotient equality, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyClamp}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyClamp}^{\uparrow}$ packet is generated from one RegSeqRat source row, two RegSeqRat endpoint rows, StreamName windows, a DyadicRatCore tolerance ledger, RegularCauchyMaximum and RegularCauchyMinimum factorization rows, an exported RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyClamp}^{\uparrow}}$, seed)

Scope closed: Definition 6841.1, Definition 6841.2, Definition 6841.3, and Definition 6841.4 seed the finite regular-Cauchy clamp route below the Real-facing clamp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyClamp}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a checked Lean carrier, bridge checking, host clamp operations, trichotomy, quotient stream equality, selected limits, countable choice, chosen dense enumerations, or ambient Real completeness.

Upgrade path: Obligation closure requires explicit carrier admission, window coverage, dyadic tolerance exactness, min–max factorization exactness, Real seal handoff, transport, replay, provenance, and local NameCert obligations for BEDC.Derived.RegularCauchyClampUp.

Chapter 6842

A Concrete Naming Certificate for $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy estimates over dyadic rational scales, the DyadicRadiusHalvingLadd packet records the finite radius ladder used by the L10 Real and Completion route. It sits between the dyadic arithmetic rows of chapter 291, the finite-window naming discipline of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet supplies monotone 2^{-n} radii as displayed NameCert data; it does not import completed real algebra, quotient-name equality, countable choice, selected limits, or an ambient completeness theorem.

Definition 6842.1 (Dyadic radius halving ladder carrier)**.** *A* $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Z, R, W, S, M, E, H, C, P, N)$$

with the following visible rows:

1. *Z is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *unit-radius row naming the initial scale 2^0 and its positive dyadic certificate.*
2. *R is the finite radius-list row, displaying the entries $r_i = 2^{-i}$ for every inspected index in the current request.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records exactly which indices of the ladder are inspected.*
4. *S is the successor-halving row, certifying $r_{i+1} + r_{i+1} = r_i$ on every displayed adjacent pair.*
5. *M is the monotonicity row, recording $r_{i+1} \leq r_i$ and the stronger scale comparison $r_{i+k} \leq r_i$ whenever both indices are in the displayed window.*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *error-budget row consumed by regular Cauchy estimates and by the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *H records componentwise* hsame *transport for unit scale, finite radius rows, stream windows, halving, monotonicity, error-budget, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request to a finite stream window, dyadic radius lookup, halving comparison, monotone-scale read, and regular Cauchy budget handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *unary-history window material,* BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}}$ *row.*

The classifier compares packets only through the displayed rows $Z, R, W, S, M, E, H, C, P, N$. It has no coordinate for a host exponential function, selected real limit, quotient real equality, countable choice, ambient real algebra, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6842.2 (Dyadic radius monotone scale)**.** *Let $D = (Z, R, W, S, M, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$ *carrier. Every radius comparison exported to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the same finite window W, the same displayed dyadic radius row R, and the same successor-halving row S. In particular the consumed scale is monotone on the displayed request:*

$$r_j \leq r_i$$

whenever $i \leq j$ and both indices are visible in W.

Proof. Project the carrier of Definition 6842.1. The only scale coordinates are the unit row Z, finite radius row R, stream window W, successor-halving row S, and monotonicity row M.

Read the adjacent comparisons first. The row S gives the displayed equality $r_{i+1} + r_{i+1} = r_i$ for every adjacent pair in W, and positivity is inherited from the $\mathsf{DyadicRatCore}^{\uparrow}$ row Z and the radius-list row R.

Now iterate inside the finite window. The monotonicity row M records the finite transitive closure of those adjacent comparisons for the same displayed indices, so a later radius can be consumed only as a smaller or equal dyadic budget.

Finally transport the result through H, C, P, N. These rows replay and name the same ladder before E is read by $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, and they add no host exponential, selected limit, quotient equality, countable-choice row, real algebra, or completeness coordinate. □

Theorem 6842.3 (Dyadic radius refinement handoff)**.** *Every accepted* $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$ *packet supplies a refinement route from a requested finite stream window to a regular Cauchy error budget:*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer first displays the window W, then reads the dyadic radii R and adjacent-halving row S, then consumes the monotone-scale row M, and only then reaches the regular Cauchy budget row E and any terminal Real-facing seal.

Proof. Start with a precision request. The continuation row C routes that request to the finite $\mathsf{StreamName}^{\uparrow}$ window W before any radius value is inspected.

Within that window, the dyadic rows are explicit. The radius row R lists the inspected 2^{-i} entries and S records the halving relation between adjacent displayed entries, so the route stays inside $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic.

Apply Definition 6842.2. The monotone row M is the only bridge from the finite dyadic ladder to the regular Cauchy error ledger, and it is already attached to the same window and radius rows.

The handoff row E can therefore be consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after the finite ladder has been displayed. Provenance and local naming through P, N preserve this dependency and exclude a host exponential function, quotient-name equality, selected limit, countable choice, ambient real algebra, or completeness theorem. □

Theorem 6842.4 (Dyadic radius NameCert obligation surface)**.** *For the carrier of Definition 6842.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}}$ *surface consists of exactly five rows:*

1. *carrier admission for the displayed packet $D = (Z, R, W, S, M, E, H, C, P, N)$;*

2. *classifier locality for the* DyadicRatCore$^{\uparrow}$ *unit scale, finite radius list, and* StreamName$^{\uparrow}$ *inspected window;*
3. *stability through componentwise* hsame *transport H and displayed* Cont *replay C;*
4. *exactness through the successor-halving row S, monotone-scale row M, and regular Cauchy budget row E;*
5. *ledger control through* Pkg *provenance P and the local naming row N.*

No NameCert obligation may replace these rows by a host exponential map, selected real limit, quotient stream equality, arbitrary trichotomy, countable choice, ambient real algebra, or completeness of Real$^{\uparrow}$.

Proof. Carrier admission projects $Z, R, W, S, M, E, H, C, P, N$ from Definition 6842.1. No radius, window, halving, monotonicity, error-budget, transport, replay, provenance, or naming row is implicit.

Classifier locality reads only the dyadic unit scale, finite radius list, and inspected stream window. This prevents the ladder from ranging over an unbounded host sequence or an external exponential function.

Stability and exactness are supplied by the displayed structural rows. The hsame and Cont rows preserve the same finite packet, while S, M, E state the halving, monotone-scale, and regular-budget facts consumed downstream.

The package and naming rows close the ledger. Since every exported read must pass through P and N, the certificate surface is exhausted by the five listed rows. □

Falsifiable prediction: A regular Cauchy or Real-facing consumer refutes the DyadicRadiusHalvingLadderUp carrier if it reads a 2^{-n} budget without displaying the finite StreamNameUp window, DyadicRatCoreUp radius row, adjacent-halving row, monotone-scale row, regular Cauchy budget row, transport, replay, provenance, and local naming rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, and unaryhistory-window material: none is bijective to DyadicRadiusHalvingLadder$^{\uparrow}$ because only this packet binds unit dyadic scale, finite radius list, inspected stream window, successor halving, monotone refinement, regular Cauchy budget, transport, replay, provenance, and local naming as one NameCert surface.

Closure status for NameCert$_{\text{DyadicRadiusHalvingLadder}\uparrow}$.

Constructive story: A DyadicRadiusHalvingLadder$^{\uparrow}$ packet is generated from a DyadicRatCore unit-radius row, finite dyadic radius rows, a StreamName inspected window, successor-halving rows, monotone-scale rows, a RegSeqRat error-budget handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{DyadicRadiusHalvingLadder}\uparrow}$, seed)

Scope closed: Definition 6842.1, Definition 6842.2, Definition 6842.3, and Definition 6842.4 seed the finite dyadic radius ladder required by regular Cauchy and Real-completion estimates.

Formal verification: FormalStatus(NameCert$_{\text{DyadicRadiusHalvingLadder}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host exponentiation, selected limits, quotient real equality, arbitrary trichotomy, countable choice, ambient real algebra, completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic radius exactness, stream-window locality, successor-halving exactness, monotone-scale refinement, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicRadiusHalvingLadderUp.

Chapter 6843

A Concrete Naming Certificate for $\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}$ packet records the finite Archimedean extraction by which a Cauchy presentation yields an inspectable tail modulus before a Real-completion consumer reads a seal. It sits beside the Archimedean real source of chapter 430, the Cauchy carrier of chapter 3413, the dyadic tolerance ledger of chapter 291, the stream-name window surface of chapter 126, the regular rational readback of chapter 301, and the Real seal of chapter 38. The packet names the finite tail-modulus extraction route; it does not assert ambient completeness, select a quotient representative, or choose a limit outside the displayed rows.

Definition 6843.1 (Archimedean Cauchy tail modulus carrier)**.** *An* $\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, Q, D, S, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *source row, exposing Archimedean comparison data only as finite named evidence.*
2. *Q is the* $\mathsf{Cauchy}^{\uparrow}$ *presentation row whose tail request is being interpreted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the requested positive precision, slack conversion, and dyadic comparison budget.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row, recording the finite windows of the Cauchy presentation that have been read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from those windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after R and the tolerance row D have been displayed.*
7. *T is the tail-modulus row, recording the finite index after which the displayed Cauchy windows satisfy the requested dyadic tolerance.*
8. *H records componentwise* hsame *transport for A, Q, D, S, R, E, T and the structural rows.*
9. *C records displayed* Cont *replay from a tail request through source, Cauchy presentation, tolerance, windows, readback, Real seal, and tail-modulus rows.*

10. *P is the* Pkg *provenance over* ArchimedeanCauchyTailModulus$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Cauchy$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* NameCert$_{\text{ArchimedeanCauchyTailModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source, Cauchy, dyadic, stream, readback, Real-seal, tail-modulus, transport, replay, provenance, and local naming rows. It has no coordinate for quotient stream equality, selected limits, countable choice, hidden Archimedean search, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ArchimedeanCauchyTailModulusUp`

Theorem 6843.2 (Archimedean Cauchy tail modulus NameCert obligations)**.** *For every accepted* ArchimedeanCauchyTailModulus$^{\uparrow}$ *carrier* $M = (A, Q, D, S, R, E, T, H, C, P, N)$*, the local* NameCert$_{\text{ArchimedeanCauchyTailModulus}^{\uparrow}}$ *surface consists exactly of carrier admission,* ArchimedeanReal$^{\uparrow}$ *source exposure,* Cauchy$^{\uparrow}$ *presentation exposure,* DyadicRatCore$^{\uparrow}$ *tolerance exactness,* StreamName$^{\uparrow}$ *finite-window inspection,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal non-escape, tail-modulus index exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6843.1. The source, Cauchy presentation, tolerance, stream-window, readback, Real-seal, tail-modulus, transport, replay, provenance, and local-name rows are precisely the displayed packet coordinates.

Read the Archimedean and Cauchy rows first. The source row supplies only finite comparison evidence, and the Cauchy row supplies only the named presentation whose tail is being inspected.

Use the dyadic tolerance row together with the stream window row. The requested precision is a DyadicRatCore$^{\uparrow}$ datum, and the inspected tail data are exactly the finite windows recorded by StreamName$^{\uparrow}$.

The regular rational readback and Real seal come after those rows. The tail-modulus row T then records the displayed index at which the same windows satisfy the same tolerance. Transport, replay, provenance, and local naming preserve this route, so there is no hidden coordinate for quotient equality, selected limits, choice, or ambient completeness. □

Theorem 6843.3 (Archimedean Cauchy tail modulus extraction route)**.** *Every Real-and-completion consumer of an accepted* ArchimedeanCauchyTailModulus$^{\uparrow}$ *packet reads the tail index only through*

$$\text{ArchimedeanReal}^{\uparrow},\quad \text{Cauchy}^{\uparrow},\quad \text{DyadicRatCore}^{\uparrow},\quad \text{StreamName}^{\uparrow},\quad \text{RegSeqRat}^{\uparrow},\quad \text{Real}^{\uparrow}.$$

The exported datum is a finite tail-modulus row for the displayed Cauchy presentation, not a quotient Real equality, selected representative, choice sequence, or ambient completeness theorem.

Proof. Start with the carrier route. The accepted packet stores the Archimedean source and Cauchy presentation as finite rows A and Q, so the consumer begins with displayed data rather than an external real object.

Next consume the tolerance and window rows. The requested precision is read from D, while S records the inspected finite tail windows. No tail index is exported before both rows are visible.

Then read R and E. The regular rational readback and Real seal are downstream of the same windows and tolerance, so they cannot add a selected limit or quotient representative.

Finally expose T through the structural rows H, C, P, N. Since the carrier contains no hidden search, choice, quotient, or completeness coordinate, the tail-modulus row is exactly the finite extraction route named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}$ packet is generated from ArchimedeanReal source evidence, a Cauchy presentation, DyadicRatCore tolerance data, StreamName tail windows, RegSeqRat readback, a Real seal, a tail-modulus index row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6843.1, Definition 6843.2, and Definition 6843.3 seed the finite Archimedean tail-modulus extraction surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTailModulus}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient Real equality, selected representatives, countable choice, hidden Archimedean search, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, finite-window inspection, RegSeqRat readback, Real seal non-escape, tail-modulus index exposure, transport, replay, provenance, and local naming for BEDC.Derived.ArchimedeanCauchyTailModulusUp.

Chapter 6844

A Concrete Naming Certificate for ConsistencyObservationBudget$^{\uparrow}$

The ConsistencyObservationBudget$^{\uparrow}$ packet realizes the finite observation side of the computation-and-consistency vision in section 9364.10. It sits after the MetaCIC critical-path route of chapter 765, the stream-name observation surface of chapter 126, the regular rational readback of chapter 301, and the Real seal of chapter 38. The packet records bounded consistency observations as NameCert rows; it does not claim global consistency, semantic soundness for every host model, or an unbounded proof search oracle.

Definition 6844.1 (Consistency observation budget carrier)**.** *A* ConsistencyObservationBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, O, S, R, E, M, L, F, H, C, P, N)$$

with the following displayed rows:

1. *K is the* MetaCICCriticalPath$^{\uparrow}$ *kernel route row through which accepted declarations, reduction traces, and checker handoffs are observed.*
2. *O is the finite observation schedule row, listing the declarations, reductions, checker calls, and local contradiction probes admitted by the budget.*
3. *S is the* StreamName$^{\uparrow}$ *row naming the finite stream of observation windows.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for numeric or ranked budget counters attached to those windows.*
5. *E is the* Real$^{\uparrow}$ *seal row for any Real-facing budget threshold used by an observation consumer.*
6. *M is the metatheoretic boundary row, recording that each observation is local to the displayed MetaCIC route.*
7. *L is the ledger row listing accepted observation outcomes, refused out-of-budget requests, and unresolved probes.*
8. *F is the field-faithfulness request row that later Lean TasteGate work must discharge with* Nontrivial *and* FieldFaithful *instances for the carrier.*
9. *H records componentwise* hsame *transport for K, O, S, R, E, M, L, F and the structural rows.*

10. *C records displayed* Cont *replay from observation request through kernel route, schedule, stream windows, readback, seal, boundary, ledger, field-faithfulness request, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{ConsistencyObservationBudget}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConsistencyObservationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for global consistency, hidden model identity, unbounded proof search, semantic soundness outside the inspected route, quotient proof equality, or a final contradiction oracle.

Lean4 (statement) `BEDC.Derived.ConsistencyObservationBudgetUp`

Theorem 6844.2 (Consistency observation budget NameCert obligations)**.** *For every accepted* $\mathsf{ConsistencyObservationBudget}^{\uparrow}$ *carrier* $B = (K, O, S, R, E, M, L, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConsistencyObservati}}$ *surface consists exactly of carrier admission,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *route exposure, finite observation schedule exposure,* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *counter readback,* $\mathsf{Real}^{\uparrow}$ *threshold seal non-escape, metatheoretic boundary exposure, observation ledger exposure,* Nontrivial *and* FieldFaithful *TasteGate request exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6844.1. The kernel route, observation schedule, stream windows, readback counters, threshold seal, boundary row, ledger row, field-faithfulness request, and structural rows are exactly the displayed coordinates.

The observation route begins at $\mathsf{MetaCICCriticalPath}^{\uparrow}$. The packet can inspect accepted declarations, reductions, and checker handoffs only through the route row K and the finite schedule O.

The stream, readback, and seal rows bind the budget to finite observations. The ledger row L then records outcomes, refusals, and unresolved probes without turning any local observation into a global consistency theorem.

Finally the row F states the downstream Lean TasteGate obligation: Nontrivial and FieldFaithful instances must be supplied before the carrier can be read as a faithful field surface. Transport, replay, provenance, and local naming preserve the same finite packet, so no hidden model or unbounded oracle is exported. □

Theorem 6844.3 (Consistency observation finite-budget boundary)**.** *Every consistency-facing read exported by an accepted* $\mathsf{ConsistencyObservationBudget}^{\uparrow}$ *packet factors through the finite observation route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the boundary, ledger, field-faithfulness request, transport, replay, provenance, and local NameCert *rows. The read reports bounded observations only; it does not certify global consistency, hidden model identity, unbounded proof search, or semantic soundness outside the displayed MetaCIC route.*

Proof. Start with the kernel route and schedule rows K, O. They define which declarations, reductions, checker calls, and local contradiction probes are inside the observation budget.

Next read S, R, E. The stream windows, counter readbacks, and threshold seal keep the budget finite and inspectable before any consistency-facing report is consumed.

Use the boundary and ledger rows M, L. The boundary row states that the observation is local to the displayed MetaCIC route, and the ledger row records accepted outcomes, refused out-of-budget requests, and unresolved probes.

Finally keep F, H, C, P, N visible. The TasteGate request records the needed Nontrivial and FieldFaithful instances, while transport, replay, provenance, and naming preserve the same carrier. Therefore the exported read is a bounded observation packet, not a global consistency claim. □

Falsifiable prediction: A claimed $\mathsf{ConsistencyObservationBudget}^{\uparrow}$ consumer is refuted if it exports global consistency, hidden model identity, unbounded proof search, or semantic soundness outside a finite MetaCICCriticalPath observation schedule.

Independence witness: metaciccriticalpath, streamname, regseqrat, real: none is bijective with $\mathsf{ConsistencyObservationBudget}^{\uparrow}$, because the siblings supply kernel route, observation windows, counter readback, and threshold seal rows while this packet additionally binds observation schedule, metatheoretic boundary, ledger, TasteGate request, transport, replay, provenance, and local NameCert rows.

Lean TasteGate work for $\mathsf{ConsistencyObservationBudget}^{\uparrow}$ must provide Nontrivial and FieldFaithful instances before the budget can serve as a faithful carrier for downstream consistency-observation consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{ConsistencyObservationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{ConsistencyObservationBudget}^{\uparrow}$ packet is generated from a MetaCICCriticalPath route, finite observation schedule, StreamName windows, RegSeqRat counters, Real threshold seal, metatheoretic boundary, observation ledger, TasteGate request, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConsistencyObservationBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6844.1, Definition 6844.2, and Definition 6844.3 seed the finite consistency-observation budget surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConsistencyObservationBudget}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close global consistency, hidden model identity, unbounded proof search, semantic soundness outside the displayed route, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, MetaCIC route exposure, finite schedule exposure, stream-window admission, counter readback, threshold seal non-escape, boundary and ledger exposure, TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.ConsistencyObservationBudgetUp.

Chapter 6845

A Concrete Naming Certificate for $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ packet records a finite route from a displayed unary Archimedean bound to dyadic approximation data that can be handed to $\mathsf{Real}^{\uparrow}$ consumers. It sits beside the natural-number bound surface of chapter 28, the rational approximation rows of chapter 37, the dyadic precision ledger of chapter 291, the approximation packet of chapter 335, and the terminal real-name seal of chapter 38. The chapter names a finite BEDC witness for Archimedean dyadic approximation; it does not import host real equality, quotient cuts, selected Cauchy limits, trichotomy, countable choice, or ambient completeness.

Definition 6845.1 (ArchimedeanDyadicApproximation carrier)**.** *An* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, Q, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the unary Archimedean bound row, read through the finite* $\mathsf{Nat}^{\uparrow}$ *naming surface.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *approximation row attached to the same bound request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row that records the dyadic denominator and tolerance ledger.*
4. *W is the* $\mathsf{DyadicApproximation}^{\uparrow}$ *bracket row, exposing the finite dyadic enclosure licensed by B, Q, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, available only after the bound, rational approximation, precision, and bracket rows have been displayed.*
6. *H records componentwise* hsame *transport for the bound, rational approximation, dyadic precision, bracket, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the Archimedean bound through rational approximation, dyadic precision, dyadic bracket, and then the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicApproximation}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient cuts, host real equality, a selected Cauchy limit, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6845.2 (ArchimedeanDyadicApproximation NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ *carrier* $A = (B, Q, D, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicApproxi}}$ *surface consists exactly of carrier admission for the displayed bound, rational approximation, dyadic precision, bracket, real seal, transport, replay, provenance, and local naming rows; Archimedean exactness requiring* B *to be read before* Q*; dyadic precision exactness requiring* D *before* W*; real handoff exactness requiring* W *before* E*; and ledger exhaustion excluding quotient cuts, host equality, selected limits, trichotomy, choice schedules, and ambient completeness.*

Proof. Project the carrier of Definition 6845.1. The rows $B, Q, D, W, E, H, C, P, N$ are displayed as separate coordinates, so none of the downstream rows can hide the bound or precision data.

The Archimedean and dyadic clauses are read in order. The rational approximation row Q is attached to the unary bound row B, while the bracket row W is downstream of the dyadic precision row D.

Finally, the real seal E is reached only after B, Q, D, W have been displayed. The structural rows H, C, P, N transport, replay, package, and name this same finite packet, so the excluded principles have no carrier coordinate from which they can be exported. □

Theorem 6845.3 (Archimedean dyadic approximation real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ *packet receives the finite route*

$$B, Q, D, W \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer reads a unary Archimedean bound, a $\mathsf{Rat}^{\uparrow}$ *approximation row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row, a* $\mathsf{DyadicApproximation}^{\uparrow}$ *bracket row, and the* $\mathsf{Real}^{\uparrow}$ *seal; it receives no host real equality, quotient cut, selected Cauchy limit, trichotomy row, choice schedule, or ambient completeness principle.*

Proof. Use Definition 6845.2. Its Archimedean exactness, dyadic precision exactness, and real handoff exactness clauses force the route B, Q, D, W, E before any real-facing read can occur.

After the seal is read, the only remaining coordinates are H, C, P, N. These rows preserve the same finite packet by componentwise transport, displayed replay, provenance, and local naming.

Therefore a $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed bound-to-bracket route with the seal and structural rows. Since quotient cuts, host equality, selected limits, trichotomy, choice schedules, and ambient completeness are absent from the carrier and the obligation surface, the handoff cannot export them. □

Lean4 (statement) `BEDC.Derived.ArchimedeanDyadicApproximationUp.ArchimedeanDyadicApproximationUp`

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 6845.1, Definition 6845.2, and Definition 6845.3 seed the finite Archimedean dyadic approximation route from unary bounds to the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.ArchimedeanDyadicApproximationUp.ArchimedeanDyadicApproximationUp`

Bridge status: none

Not claimed: This chapter does not close quotient cuts, host real equality, selected Cauchy limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, bound exactness, dyadic precision exactness, real handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.ArchimedeanDyadicApproximationUp.

Constructive story: An $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ packet is generated from a unary Archimedean bound, a Rat approximation row, a DyadicRatCore precision row, a DyadicApproximation bracket row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6846

A Concrete Naming Certificate for $\mathsf{CompactUniformMeshModulus}^{\uparrow}$

Following the finite-net proof of Heine–Cantor in Bishop and Bridges constructive metric analysis, the $\mathsf{CompactUniformMeshModulus}^{\uparrow}$ packet realizes the compact finite-mesh route of chapter 9237 as concrete NameCert data. It consumes the compact metric finite-net rows of chapter 129, the continuous-map pointwise-modulus rows of chapter 130, the rational lower-bound rows of chapter 37, the finite-net fold of chapter 619, and the tolerance-only consumer surface of chapter 369. The packet records a finite $\mathsf{ProbeBundle}$ mesh and a rational lower-bound fold; it does not import open-cover compactness, choice-selected centers, semantic continuity, an external modulus function, quotient metric equality, or target completeness.

Definition 6846.1 (Compact uniform mesh modulus carrier)**.** *A* $\mathsf{CompactUniformMeshModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, M, F, B, L, Q, T, R, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, tying the compact source to displayed mesh centers and coverage witnesses.*
2. *M is the source and target metric row shared by the compact carrier and the continuous map.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, exposing pointwise-modulus ledgers at displayed mesh centers.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *mesh row, listing center names, membership rows, and source-distance coverage rows.*
5. *L is the local radius ledger that attaches a positive* $\mathsf{Rat}^{\uparrow}$ *precision radius to each displayed probe.*
6. *Q is the rational lower-bound fold row over the finite mesh.*
7. *T is the metric-triangle handoff row that converts source mesh proximity and pointwise target estimates into one uniform estimate.*
8. *R is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance-only output row consumed by downstream compact-continuity arguments.*

9. *H records componentwise* hsame *transport for compact metric data, continuous-map data, mesh rows, rational folds, triangle handoff, uniform-modulus output, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a tolerance request through finite mesh inspection, pointwise-radius readback, rational lower-bound folding, triangle handoff, and the uniform-modulus row.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformMeshModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, *finite-fold material,* BHist, hsame, Cont, Pkg, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformMeshModulus}^{\uparrow}}$ *row.*

The classifier compares packets only through $K, M, F, B, L, Q, T, R, H, C, P, N$. It has no coordinate for an ambient open-cover compactness theorem, externally selected finite subcover, selected subsequence, host modulus function, quotient metric equality, or completeness of the target metric.

Theorem 6846.2 (Compact uniform mesh finite-net handoff)**.** *Every accepted* $\mathsf{CompactUniformMeshModulus}^{\uparrow}$ *packet hands a* $\mathsf{UniformModulus}^{\uparrow}$ *consumer the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

For $U = (K, M, F, B, L, Q, T, R, H, C, P, N)$, the output row R is readable only after the compact metric mesh K, B, the pointwise-radius ledger F, L, the rational lower-bound fold Q, and the metric-triangle handoff T have all been displayed in the same packet.

Proof. Project the carrier of Definition 6846.1. The compact metric row, metric ledger, continuous-map row, finite mesh, local radii, rational fold, triangle handoff, output row, transport, replay, provenance, and naming rows are the complete inventory.

Read the finite mesh first. The compact metric row K and ProbeBundle row B expose only displayed centers and coverage witnesses, so the proof route does not pass through a host open-cover theorem or an externally selected subcover.

Next read the pointwise data. The continuous-map row F supplies local modulus evidence only at the centers displayed by B, and the local-radius row L keeps those radii attached to the same finite mesh.

Now fold and hand off. The row Q is the finite rational lower bound over L, and T is the metric-triangle estimate that converts mesh proximity plus pointwise control into one uniform estimate.

Finally expose R through H, C, P, N. These structural rows transport, replay, source, and name the same finite route, so no selected center, selected subsequence, external modulus, quotient equality, semantic continuity theorem, or target-completeness row can be exported. □

Theorem 6846.3 (Compact uniform mesh obligation surface)**.** *For the carrier of Definition 6846.1, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformMeshModulus}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for $U = (K, M, F, B, L, Q, T, R, H, C, P, N)$;*

2. *classifier locality for the* $\mathsf{CompactMetric}^{\uparrow}$ *source,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph, finite* ProbeBundle *mesh, and* $\mathsf{Rat}^{\uparrow}$ *radius histories;*

3. *stability through componentwise* hsame *transport H and displayed* Cont *replay C;*

4. *exactness through the local-radius row L, rational lower-bound fold Q, metric-triangle handoff T, and* $\mathsf{UniformModulus}^{\uparrow}$ *output row R;*

5. *ledger control through* Pkg *provenance P and the local naming row N.*

No NameCert obligation may replace these rows by open-cover compactness, a choice-selected mesh, semantic continuity, an external modulus function, quotient metric equality, selected subsequence data, or target completeness.

Proof. Carrier admission is projection from Definition 6846.1. The rows $K, M, F, B, L, Q, T, R, H, C, P, N$ are all explicit, and no compact, continuous, mesh, fold, triangle, output, transport, replay, provenance, or naming row is implicit.

Classifier locality reads only the compact metric source, continuous graph, finite mesh, and rational radius histories. This keeps the mesh-to-modulus route inside displayed NameCert data.

Stability and exactness are then separated. The rows H and C preserve the same packet under transport and replay, while L, Q, T, R state the finite lower-bound and uniform-modulus handoff consumed downstream.

The package and naming rows close the ledger. Since every exported read must pass through P and N, the certificate surface is exactly the five listed rows and contains no coordinate for the excluded host principles. □

Falsifiable prediction: A UniformModulusUp consumer refutes CompactUniformMeshModulusUp if it reaches the tolerance-only row without displaying the CompactMetricUp finite mesh, ContinuousMapUp pointwise-radius rows, finite ProbeBundle mesh, RatUp lower-bound fold, metric-triangle handoff, transport, replay, provenance, and local naming rows.

Independence witness: compactmetric, continuousmap, uniformmodulus, rat, finitenetminimumfold, and compact-uniform-ladder siblings: none is bijective to $\mathsf{CompactUniformMeshModulus}^{\uparrow}$ because only this packet binds compact finite-mesh coverage, pointwise radii, rational lower-bound folding, metric-triangle handoff, uniform-modulus output, transport, replay, provenance, and local naming as one NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformMeshModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformMeshModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, finite ProbeBundle mesh rows, Rat lower-bound folds, metric-triangle handoff rows, UniformModulus output rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformMeshModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6846.1, Definition 6846.2, and Definition 6846.3 seed the finite mesh-to-uniform-modulus route for compact continuous maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformMeshModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected meshes, selected subsequences, semantic continuity, external modulus functions, quotient metric equality, target completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact mesh exposure, pointwise-radius attachment, rational lower-bound fold exactness, metric-triangle handoff, UniformModulusUp output, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformMeshModulusUp.

Chapter 6847

A Concrete Naming Certificate for $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$

Following Hausdorff and Bishop–Bridges constructive metric analysis, the $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$ packet records the finite-net approximation surface used when compact metric names are compared through displayed Hausdorff tolerances. It sits over the metric carrier of chapter 57, the compact-metric row of chapter 129, the total-bounded finite-net source of chapter 127, the finite epsilon-net handoff of chapter 1767, the Hausdorff metric boundary of chapter 2086, and the dyadic, regular-rational, and real-name readbacks of chapter 291, chapter 301, and chapter 38. The packet names a finite approximation object for completion-facing consumers without importing open-cover compactness, selected dense subsets, countable choice, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6847.1 (Finite Hausdorff metric approximation carrier)**.** *A* $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, K, \varepsilon, E_A, E_B, D, H, R, S, T, P, N)$$

with the following displayed rows:

1. *X is the shared* $\mathsf{Met}^{\uparrow}$ *row, carrying the point classifier and distance rows used by the approximation.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net field read at the displayed tolerance.*
3. *ε is the positive tolerance row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and, when a Real-facing bound is requested, through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal route.*
4. *E_A and E_B are two finite* $\mathsf{ProbeBundle}$ *net rows for the compact names being compared, including center membership and coverage witnesses exposed by the finite-net handoff.*
5. *D is the directed Hausdorff comparison ledger: every center of E_A has a displayed distance row to an admitted center of E_B, and every center of E_B has a displayed distance row to an admitted center of E_A, all bounded by the same tolerance budget.*
6. *H records componentwise* hsame *transport for the metric source, compact-metric row, tolerance row, the two finite nets, directed comparison ledger, readback row, seal row, replay, provenance, and local naming rows.*

7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for any rational or dyadic Hausdorff bound exposed to a completion-facing consumer.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite nets, directed distance ledger, and regular-rational readback have been displayed.*
9. *T records displayed* Cont *replay of the route*

$$X, K, \varepsilon, E_A, E_B \longmapsto D \longmapsto R \longmapsto S.$$

10. *P is the* Pkg *provenance over* $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient finite subset, selected dense set, open cover, selected subcover, quotient completion, countable-choice schedule, host hyperspace equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 6847.2 (Finite Hausdorff metric approximation NameCert obligations)**.** *For every accepted* $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$ *carrier* $F = (X, K, \varepsilon, E_A, E_B, D, H, R, S, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, compact-metric row, tolerance, two finite nets, directed Hausdorff comparison ledger, componentwise transport, regular-rational readback, terminal Real seal, replay, provenance, and local naming rows. Every approximation read must expose E_A, E_B, D before R or S is consumed, and the ledger exports no open-cover compactness, selected subcover, quotient completion, countable choice, selected limit, host hyperspace equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6847.1. The metric and compact rows are X, K, the tolerance row is ε, and the finite approximation data are exactly the two net rows E_A, E_B together with the directed comparison ledger D.

The finite-net rows are prerequisite data. A center can be used in the directed Hausdorff ledger only after it is visible in the relevant $\mathsf{ProbeBundle}$ and admitted through the compact-metric and total-bounded source rows.

The directed comparison ledger is then forced by D. It records both directions of the finite Hausdorff estimate under the same tolerance budget, so a consumer cannot replace the row by an open cover, a selected dense subset, or a host hyperspace construction.

The readback and seal rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ row R and terminal $\mathsf{Real}^{\uparrow}$ seal S expose only the finite bound already produced by E_A, E_B, D, while H, T, P, N transport, replay, source, and name that same packet. □

Theorem 6847.3 (Finite-net Hausdorff handoff)**.** *Every compact-metric, Hausdorff-metric, or completion-facing consumer of an accepted* $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{TotallyBounded}^{\uparrow},\ E_A, E_B \longmapsto \mathsf{DyadicRatCore}^{\uparrow},\ D \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the consumer receives finite net centers, coverage rows, dyadic tolerance comparisons, regular-rational readback, and the terminal Real seal as displayed rows. It receives no open-cover compactness theorem, selected subcover, quotient completion, selected limit, countable choice, host hyperspace equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the accepted packet. By carrier admission, the only metric entry point is the shared $\mathsf{Met}^{\uparrow}$ row X, and the compact data are read through the $\mathsf{CompactMetric}^{\uparrow}$ row K together with its $\mathsf{TotallyBounded}^{\uparrow}$ finite-net field.

Open the finite nets next. The rows E_A and E_B provide the displayed center bundles and coverage witnesses at the tolerance ε, and the dyadic tolerance row keeps the scale comparison inside $\mathsf{DyadicRatCore}^{\uparrow}$.

Apply the directed ledger D. It supplies the two finite Hausdorff comparison directions before any rational or Real-facing bound can be exported, so the handoff remains a finite-net route rather than an open-cover or host-hyperspace argument.

Finally read R and S. The $\mathsf{RegSeqRat}^{\uparrow}$ readback packages the bound for completion-facing consumers, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after the finite nets and dyadic comparison ledger have been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6847.1, Definition 6847.2, and Definition 6847.3 seed the finite-net Hausdorff approximation route over compact metric data, dyadic tolerance rows, regular-rational readback, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host hyperspaces, selected dense subsets, selected subcovers, quotient completion, countable choice, selected limits, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net visibility, directed Hausdorff ledger exactness, dyadic tolerance transport, RegSeqRat readback, Real seal nonescape, replay, provenance, and local naming for BEDC.Derived.FiniteHausdorffMetricApproximationUp.

Constructive story: A $\mathsf{FiniteHausdorffMetricApproximation}^{\uparrow}$ packet is generated from a shared MetricSpaceUp row, a CompactMetricUp source, a positive DyadicRatCoreUp tolerance, two finite ProbeBundle nets, a directed Hausdorff comparison ledger, componentwise hsame transport, RegSeqRat readback, a terminal RealUp seal, displayed Cont replay, Pkg provenance, and local naming.

Chapter 6848

A Concrete Naming Certificate for $\mathsf{BishopIntervalSelection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface by which a Real and Completion consumer may pass from displayed closed bounded intervals to a bisection-driven real-name handoff. It sits after the closed interval carrier of chapter 1591, the located bisection route of chapter 1711, the regular rational readback of chapter 301, and the terminal real-name certificate of chapter 38. The packet names the constructive choice of a displayed interval branch at finite precision; it does not select a host point, invoke countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6848.1 (Bishop interval-selection carrier)**.** *A* $\mathsf{BishopIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, L, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, recording the left endpoint, right endpoint, and boundedness ledger available to the selection request.*
2. *L is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *branch row, recording the finite bisection decisions and located side tests already accepted by the packet.*
3. *Q is the rational tolerance row, naming the requested precision and the side condition that the displayed interval length is within that request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite sequence of selected interval endpoints.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, L, Q, R have been displayed.*
6. *H records componentwise* hsame *transport for interval endpoints, bisection rows, tolerances, regular readbacks, real seals, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay from interval requests through bisection, rational tolerance, regular readback, and real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalSelection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, Q, R, E, H, C, P, N$. It has no coordinate for a selected real point, host interval compactness, quotient real equality, countable-choice branch schedule, or ambient completeness theorem.

Theorem 6848.2 (Bishop interval-selection NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalSelection}^{\uparrow}$ *carrier* $I = (B, L, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, closed bounded interval exposure, located bisection branch exposure, tolerance coverage, regular rational readback, terminal real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6848.1. The analytic rows are B, L, Q, R, E, and they are ordered from displayed closed interval data to located bisection, tolerance, regular readback, and real sealing.

Use the sibling interval chapters. chapter 1591 supplies the closed bounded interval row, and chapter 1711 supplies the finite branch row whose side tests are already displayed.

Now read the completion-facing rows. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ chapters, chapter 301 and chapter 38, are reached only through R and E after the tolerance row Q has been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 6848.3 (Bishop interval-selection nested interval handoff)**.** *Every completion-facing read of an accepted* $\mathsf{BishopIntervalSelection}^{\uparrow}$ *carrier must read the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row B, the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *branch row L, the tolerance row Q, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E. Thus nested interval data enter the Real and Completion chain as a displayed finite handoff, not as a hidden point selector.*

Proof. Start with the carrier row order in Definition 6848.1. The interval source and bisection branch are the only rows that can determine which closed subinterval is selected at a finite stage.

Apply Definition 6848.2. It requires tolerance coverage and regular rational readback to be part of the same local certificate before the real seal is available.

The structural rows H, C, P, N preserve that route without adding a branch selector. Therefore a consumer that reaches E has already consumed the displayed rows B, L, Q, R, and a consumer that bypasses those rows leaves the accepted $\mathsf{BishopIntervalSelection}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted BishopIntervalSelectionUp completion-facing read displays a ClosedBoundedIntervalUp source row, a LocatedIntervalBisectionUp branch row, tolerance coverage, RegSeqRatUp readback, RealUp sealing, transport, replay, provenance, and local NameCert naming.

Independence witness: closedboundedinterval, located-interval-bisection, regseqrat, real: $\mathsf{BishopIntervalSelection}$ is not bijective to any listed sibling because it jointly carries a closed interval source, located branch decisions, tolerance coverage, regular readback, terminal real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalSelection}^{\uparrow}$ packet is generated from closed bounded interval rows, located bisection rows, tolerance rows, regular rational readback, real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6848.1, Definition 6848.2, and Definition 6848.3 seed the finite interval-selection handoff for the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host point selection, countable choice, quotient real equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exposure, bisection branch exposure, tolerance coverage, regular readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalSelectionUp.

Chapter 6849

A Concrete Naming Certificate for $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$

The $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$ packet realizes the Cook compile-frontier budget named in section 9390.5 as finite NameCert data. It combines the cellular automaton substrate of chapter 934, the Rule 110 trust substrate of chapter 6517, the coordinate packet of chapter 949, the Cook compile-frontier witness of chapter 952, and the compile-frontier witness of chapter 954. The packet records a budget surface rather than a proof of universality: it displays production counts, tape-symbol counts, step budgets, closed round trips, and obstruction rows as inspectable finite data.

Definition 6849.1 (Rule 110 Cook frontier-budget carrier)**.** *A* $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, U, K, C, P, T, S, R, O, H, D, G, N)$$

with the following visible rows:

1. *A is the* $\mathsf{CellularAutomaton}^{\uparrow}$ *substrate row naming the local cellular rule and finite orbit interface.*
2. *U is the* $\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}$ *row naming the Rule 110 manifest, evaluator, classifier, and audit-route surface.*
3. *K is the* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *coordinate row for production-count, tape-symbol-count, and step-budget entries.*
4. *C is the* $\mathsf{CookCompileFrontierWitness}^{\uparrow}$ *row recording the Cook compile-frontier witness consumed by the budget.*
5. *P is the* $\mathsf{CompileFrontierWitness}^{\uparrow}$ *row recording the general compile-frontier witness aligned with C.*
6. *T is the production-count row.*
7. *S is the tape-symbol-count row.*
8. *R is the step-budget and closed-round-trip row.*
9. *O is the obstruction row, recording why a larger requested compile-frontier instance is outside the displayed budget.*

10. *H records componentwise* hsame *transport for substrate, coordinate, witness, count, round-trip, obstruction, replay, provenance, and local naming rows.*
11. *D records displayed* Cont *replay for the same rows.*
12. *G is the* Pkg *provenance over the substrate, coordinate, witness, budget, transport, replay, and local naming rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unstated universality theorem, hidden evaluator, unbounded tape oracle, external complexity claim, or closed-round-trip result not displayed by the packet.

Theorem 6849.2 (Rule 110 Cook frontier-budget NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$ *carrier $F = (A, U, K, C, P, T, S, R, O, H, D, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCoo}}$ *surface consists exactly of carrier admission, cellular substrate exposure, Rule 110 trust-substrate exposure, Cook coordinate exposure, Cook compile-frontier witness exposure, compile-frontier witness exposure, production-count coverage, tape-symbol-count coverage, step-budget and closed-round-trip coverage, obstruction-row exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6849.1. The substrate and witness rows are A, U, K, C, P, while the concrete budget rows are T, S, R, O.

Read the sibling packets in order. The cellular automaton and Rule 110 trust-substrate chapters provide the local-rule and manifest surface; the Cook coordinate and frontier-witness chapters provide the rows that interpret production counts, tape symbols, step budgets, round trips, and obstruction data.

The rows H, D, G, N transport, replay, source, and name exactly those displayed entries. Since no hidden evaluator, unbounded tape oracle, external complexity claim, or unstated universality proof is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6849.3 (Rule 110 Cook frontier-budget closed observation)**.** *Recomputing the displayed* $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$ *packet returns the same production-count row T, tape-symbol-count row S, step-budget and closed-round-trip row R, and obstruction row O whenever the substrate, coordinate, and witness rows A, U, K, C, P are unchanged. Thus the Cook compile-frontier budget is a closed observational packet over displayed rows, not a detached complexity estimate.*

Proof. Start with the substrate rows A and U. They fix the local cellular rule, manifest surface, evaluator route, classifier route, and audit readback available to the packet.

Use the coordinate and witness rows K, C, P. They determine which production-count, tape-symbol-count, step-budget, closed-round-trip, and obstruction entries are being measured.

Now replay the budget through H, D, G, N. Componentwise transport and displayed continuation preserve the same count and obstruction rows, while provenance and local naming prevent replacement by an external estimate. Therefore recomputation over unchanged substrate and witness rows returns the same displayed budget packet. □

Falsifiable prediction: Recomputing the displayed RuleOneTenCookFrontierBudgetUp packet returns the same production-count, tape-symbol-count, step-budget, closed-round-trip, and obstruction rows.

Independence witness: cellularautomaton, rule110-cellular-trust-substrate, cookfrontiercoordinate, cookcompilefrontierwitness, compilefrontierwitness: none is bijective to the combined budget

packet because RuleOneTenCookFrontierBudgetUp simultaneously carries substrate rows, coordinate rows, compile-frontier witnesses, production counts, tape-symbol counts, step budgets, closed round trips, obstruction rows, transport, replay, provenance, and local NameCert naming.

Theorem 6849.4 (Rule 110 Cook frontier-budget TasteGate handoff)**.** *The Lean follow-up site* lean4/BEDC/Derived/RuleOneTenCookFrontierBudgetUp/TasteGate.lean *must include explicit* Nontrivial RuleOneTen *and* FieldFaithful RuleOneTenCookFrontierBudget$^{\uparrow}$ *instances. The nontriviality instance must expose at least one accepted displayed budget packet, and the field-faithfulness instance must keep substrate, coordinate, witness, production-count, tape-symbol-count, step-budget, closed-round-trip, obstruction, transport, replay, provenance, and local naming rows distinct.*

Proof. The carrier is not a terminal tag: it has separate substrate, coordinate, witness, count, round-trip, obstruction, transport, replay, provenance, and naming rows.

Nontriviality therefore has to exhibit an accepted budget packet over those rows. Field faithfulness has to prevent the count rows from replacing the witness rows, the closed-round-trip row from replacing the obstruction row, or the local NameCert row from replacing provenance.

Thus the named TasteGate file is the correct follow-up site for the same displayed budget carrier. □

Closure status for NameCert$_{\text{RuleOneTenCookFrontierBudget}^{\uparrow}}$.

Constructive story: A RuleOneTenCookFrontierBudget$^{\uparrow}$ packet is generated from cellular substrate rows, Rule 110 trust-substrate rows, Cook coordinate rows, compile-frontier witness rows, production-count rows, tape-symbol-count rows, step-budget rows, closed-round-trip rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{RuleOneTenCookFrontierBudget}^{\uparrow}}$, seed)

Scope closed: Definition 6849.1, Definition 6849.2, and Definition 6849.3 seed the finite budget packet for the Rule 110 Cook compile-frontier surface.

Formal verification: FormalStatus(NameCert$_{\text{RuleOneTenCookFrontierBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a new universality theorem, an unbounded tape oracle, an external complexity bound, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, substrate exposure, coordinate exposure, witness exposure, production-count coverage, tape-symbol-count coverage, step-budget coverage, closed-round-trip coverage, obstruction exactness, nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCookFrontier

Chapter 6850

A Concrete Naming Certificate for $\mathsf{LinearlyOrderedReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LinearlyOrderedReal}^{\uparrow}$ packet records the finite order-facing route by which real-name consumers may read located comparison data without importing a host real line, quotient equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits after the located line interface of chapter 3395, the located real-name surface of chapter 1933, the order rows of chapter 1378, the finite comparison discipline of $\mathsf{TotalOrder}^{\uparrow}$, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a constructive order boundary for L10 Real and Completion routes, not a classical ordered field of reals.

Definition 6850.1 (LinearlyOrderedReal carrier)**.** *A* $\mathsf{LinearlyOrderedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Lambda = (R, L, A, O, T, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after the located and order comparison rows have been displayed.*
2. *L is the* $\mathsf{LocatedRealLine}^{\uparrow}$ *row presenting the shared located line interface for order-facing real consumers.*
3. *A is the* $\mathsf{LocatedReal}^{\uparrow}$ *row presenting the point-facing located real-name surface and its finite apartness-ready windows.*
4. *O is the* $\mathsf{Order}^{\uparrow}$ *comparison row, carrying the displayed preorder or order request read from L, A, D, S, Q.*
5. *T is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row, recording only the finite comparison branch exposed by the located line and order rows; it is not a trichotomy oracle outside the displayed request.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and tolerance ledger used to compare the located windows named by L, A, O, T.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by each order-facing comparison request.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows and dyadic tolerances.*

9. *H records componentwise* hsame *transport for the real seal, located line, located real, order rows, dyadic ledger, stream windows, regular readback, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an order-facing request through L, A, O, T, D, S, Q and then to the* $\mathsf{Real}^{\uparrow}$ *seal R.*
11. *P is the* Pkg *provenance over* $\mathsf{LinearlyOrderedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealLine}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LinearlyOrderedReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, a chosen representative of a real name, unrestricted trichotomy, host real-line completeness, selected limits, countable choice, or a public bridge.

Theorem 6850.2 (LinearlyOrderedReal NameCert obligations)**.** *For every accepted* $\mathsf{LinearlyOrderedReal}^{\uparrow}$ *carrier $\Lambda = (R, L, A, O, T, D, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LinearlyOrderedReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealLine}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *transport, replay, provenance, and local naming rows;*
2. *displayed comparison stability, requiring every comparison read to transport componentwise through H and to replay through the same continuation route C;*
3. *located-order handoff, requiring every order-facing consumer to pass through L, A, D, S, Q, O, T before the* $\mathsf{Real}^{\uparrow}$ *seal R is exposed;*
4. *ledger nonescape, excluding quotient real equality, selected representatives, unrestricted trichotomy, selected limits, countable choice, bridge data, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local data;*
5. *Real-seal routing, requiring the terminal seal R to preserve the same located line, located real, dyadic tolerance, stream-window, regular readback, and order rows already displayed by the carrier.*

No $\mathsf{LinearlyOrderedReal}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 6850.1. The analytic rows are exactly the located line row L, located real row A, order rows O, T, dyadic ledger D, stream windows S, regular readback Q, and terminal real seal R.

Displayed comparison stability is componentwise. The transport row H relates the displayed rows one by one, while the replay row C preserves the route from located line and located real data through the dyadic, stream, regular, and order rows.

The located-order handoff follows because R is terminal. An order-facing consumer reaches it only after the located line and located real rows have supplied the finite windows, the dyadic ledger has supplied the tolerance cells, and the order rows have recorded the displayed comparison.

Finally, P and N source and name the same finite packet. Since the excluded principles are not coordinates of the carrier or clauses of the obligation surface, the ledger cannot export them. □

Theorem 6850.3 (LinearlyOrderedReal located-order handoff)**.** *Every order-facing* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LinearlyOrderedReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{LocatedRealLine}^{\uparrow},\ \mathsf{LocatedReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Order}^{\uparrow},\ \mathsf{TotalOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the consumer must read the located line row L, located real row A, dyadic tolerance row D, finite window row S, regular readback row Q, order row O, and displayed total-order comparison row T before it may use the Real seal R.

Proof. Start with an accepted carrier $\Lambda = (R, L, A, O, T, D, S, Q, H, C, P, N)$. By Definition 6850.2, the only order-facing comparison rows are O and T, and both are downstream of the located line, located real, dyadic, stream, and regular readback rows.

The sibling located-line interface of chapter 3395 supplies the line-facing entry point L, while chapter 1933 supplies the point-facing located row A. The dyadic, stream, and regular readback rows are exactly the finite comparison data named by chapter 291, chapter 126, and chapter 301.

After these rows are displayed, the order request may read O and the displayed comparison branch T. The terminal seal R is then the same $\mathsf{Real}^{\uparrow}$ route cited by chapter 38, repacked through H, C, P, N.

Thus every order-facing Real consumer passes through the located line, dyadic tolerance, finite window, regular readback, and order rows before the Real seal. The handoff exports no quotient equality, selected representative, unrestricted trichotomy, selected limit, countable choice, bridge component, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LinearlyOrderedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LinearlyOrderedReal}^{\uparrow}$ packet is generated from a RealUp seal, LocatedRealLineUp row, LocatedRealUp row, OrderUp and TotalOrderUp comparison rows, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LinearlyOrderedReal}^{\uparrow}}$, seed)

Scope closed: Definition 6850.1, Definition 6850.2, and Definition 6850.3 seed the finite located-order route for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LinearlyOrderedReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, unrestricted trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displayed comparison stability, located-order handoff, ledger nonescape, Real-seal routing, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.LinearlyOrderedRealUp.

Chapter 6851

A Concrete Naming Certificate for $\mathsf{WeakCompactness}^{\uparrow}$

Following Bourbaki and Schaefer on locally convex functional analysis, the $\mathsf{WeakCompactness}^{\uparrow}$ packet records the finite weak-topology compactness interface that a functional-analysis consumer may inspect before any host weak compactness theorem is available. It sits after the finite weak-star dual-ball route of chapter 3420 and chapter 6401, beside the compact carrier of chapter 60, and over bounded functional-analysis source data. The chapter names weak compactness only as finite NameCert data: a source row, weak-observation window, bounded-set row, compactness evidence, scalar readback, transport, replay, provenance, and local naming.

Definition 6851.1 (Weak compactness carrier)**.** *A* $\mathsf{WeakCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, T, B, A, C, R, H, Q, P, N)$$

with the following displayed rows:

1. *F is the functional-analysis source row, read through the finite vector, topology, continuity, and bounded-linear-functional surfaces made available to the consumer.*
2. *T is the weak-topology observation window, listing only the finitely many bounded-linear-functional probes and scalar tolerances visible in the present read.*
3. *B is the bounded set row, exposing the displayed boundedness ledger before any compactness-facing read is allowed.*
4. *A is the Banach–Alaoglu and weak-star compactness handoff row, consumed only through the finite windows of chapter 3420 and chapter 6401.*
5. *C is the* $\mathsf{Compact}^{\uparrow}$ *compactness evidence row for the displayed weak-observation window, not an unrestricted open-cover compactness theorem over a host weak topology.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for functional values, bounds, and comparison tolerances.*
7. *H records componentwise* hsame *transport for the functional-analysis source, weak observation window, bounded row, handoff row, compactness evidence, scalar readback, replay, provenance, and local naming rows.*
8. *Q records displayed* Cont *replay from a weak compactness request through $F, T, B, A, C, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{WeakCompactness}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{WeakStarCompactness}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{WeakCompactness}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $F, T, B, A, C, R, H, Q, P, N$. It has no coordinate for a selected weak cluster point, ultrafilter argument, unrestricted product compactness, quotient equality of functionals, host weak topology, or ambient reflexivity theorem.

Lean4 (statement) `BEDC.Derived.WeakCompactnessUp`

Theorem 6851.2 (Weak compactness NameCert obligations)**.** *For every accepted* $\mathsf{WeakCompactness}^{\uparrow}$ *carrier*

$$W = (F, T, B, A, C, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeakCompactness}^{\uparrow}}$ *surface consists exactly of functional-analysis source admission through F, finite weak-observation admission through T, bounded-set admission through B, Banach–Alaoglu and weak-star compactness handoff through A, compactness evidence through C, scalar readback through R, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through N.*

Proof. Project the accepted carrier in Definition 6851.1. The displayed rows $F, T, B, A, C, R, H, Q, P, N$ are the complete inventory available to the certificate.

The analytic source is read first. Row F fixes the functional-analysis surface, row T restricts the weak topology to a finite observation window, and row B supplies the boundedness ledger before compactness evidence is inspected.

The compactness-facing part is then routed through A, C, R. The handoff row A cites only the finite Banach–Alaoglu and weak-star compactness windows, C records the compact evidence for the displayed window, and R keeps scalar values and tolerances inside the Real readback surface.

Finally the structural rows H, Q, P, N transport, replay, source, and name the same finite packet. Since no selected cluster point, ultrafilter, quotient functional equality, unrestricted product compactness, host weak topology, or reflexivity row is present, the listed obligations exhaust the local NameCert surface. □

Theorem 6851.3 (Weak compactness functional-analysis handoff)**.** *Every functional-analysis consumer of an accepted* $\mathsf{WeakCompactness}^{\uparrow}$ *packet must read the finite route*

$$F, T, B, A, C, R, H, Q, P, N$$

before exporting weak compactness data. Thus the consumer sees a functional-analysis source, a finite weak-topology observation window, a bounded-set row, the Banach–Alaoglu and weak-star compactness handoff, $\mathsf{Compact}^{\uparrow}$ *evidence,* $\mathsf{Real}^{\uparrow}$ *scalar readback, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. The export carries no selected weak cluster point, ultrafilter argument, unrestricted product compactness, quotient functional equality, host weak topology, ambient reflexivity theorem, or open-cover compactness principle.*

Proof. Begin with a functional-analysis consumer request. By Definition 6851.2, the request must expose F, T, B before any compactness-facing data can be read.

Next use the handoff row A. It points only to the finite dual-ball and weak-star compactness windows already displayed by the sibling chapters, so the consumer cannot replace the handoff with an unrestricted compactness theorem.

The compactness and scalar data then pass through C and R. Row C is the local $\mathsf{Compact}^{\uparrow}$ evidence for the displayed window, while R keeps functional values, bounds, and tolerances inside the scalar readback surface.

The structural rows H, Q, P, N preserve the same route for transport, replay, provenance, and naming. Since the carrier has no row for any refused host principle, the exported functional-analysis handoff is exactly the finite weak compactness route stated above. □

Falsifiable prediction: Every public $\mathsf{WeakCompactness}^{\uparrow}$ read factors through a functional-analysis source row, a finite weak-topology observation window, a bounded-set row, a Banach–Alaoglu and $\mathsf{WeakStarCompactness}^{\uparrow}$ handoff, $\mathsf{Compact}^{\uparrow}$ evidence, $\mathsf{Real}^{\uparrow}$ scalar readback, componentwise hsame, displayed Cont, Pkg, and local NameCert naming. A read that exports selected weak cluster points, ultrafilters, quotient functional equality, unrestricted product compactness, host weak topology, ambient reflexivity, or open-cover compactness from this packet refutes the carrier boundary.

Independence witness: banachalaoglu, weakstarcompactness, compact: none is bijective with $\mathsf{WeakCompactness}^{\uparrow}$ because this carrier binds the functional-analysis source, weak-observation window, bounded-set row, finite dual-ball handoff, compactness evidence, scalar readback, transport, replay, provenance, and local NameCert rows while refusing selection and unrestricted compactness coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakCompactness}^{\uparrow}$ packet is generated from a functional-analysis source row, a finite weak-topology observation window, a bounded-set row, a Banach–Alaoglu and weak-star compactness handoff, compactness evidence, scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WeakCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 6851.1, Definition 6851.2, and Definition 6851.3 seed the finite weak compactness interface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WeakCompactness}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close selected weak cluster points, ultrafilter arguments, quotient functional equality, unrestricted product compactness, host weak topology, ambient reflexivity, open-cover compactness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-observation-window admission, bounded-set admission, finite dual-ball handoff exactness, compactness-row handoff, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.WeakCompactnessUp.

Chapter 6852

A Concrete Naming Certificate for $\mathsf{KleeneRecursionTheorem}^{\uparrow}$

Kleene's recursion theorem in computability theory records the finite self-reference mechanism by which a computable transformation of program indices admits an index naming its own transformed behavior. The $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ packet reconstructs only the BEDC naming surface of that mechanism: finite program syntax, finite quotation of indices, a self-application window, a fixed-index witness row, transport, replay, provenance, and local naming. It sits after the computability carrier of chapter 199, the syntax and self-application surface of chapter 200, the first-order coding surface of chapter 197, and the NameCert transport discipline exemplified by chapter 481.

The packet is not a host partial-recursive semantics theorem. It does not identify programs by quotient extensional equality, does not choose representatives of partial functions, and does not import Rice-style semantic classification or Myhill-style global minimization.

Definition 6852.1 (Kleene recursion theorem carrier)**.** *A* $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, Q, W, F, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Computable}^{\uparrow}$ *program row, naming a displayed computable transformer on program indices;*
2. *Q is the finite syntax and index-quotation row, read through* $\mathsf{LambdaCalc}^{\uparrow}$ *and* $\mathsf{FirstOrder}^{\uparrow}$ *coding surfaces rather than through a host numbering theorem;*
3. *W is the self-application window, recording the finite route by which a quoted index is fed to the displayed transformer;*
4. *F is the fixed-index witness row, naming the finite index window whose transformed program is re-read through the same quotation route;*
5. *H records componentwise* hsame *transport for E, Q, W, F and the structural rows;*
6. *C records displayed* Cont *replay of the transformer, quotation, self-application, and fixed-index windows;*
7. *P is the* Pkg *provenance over* $\mathsf{KleeneRecursionTheorem}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KleeneRecursionTheorem}^{\uparrow}}$ *row.*

The carrier exports only a finite recursion-theorem naming certificate. It has no coordinate for host semantic equality of partial functions, unrestricted partiality, quotient program equality, oracle choice, Rice classification, or a Myhill minimization theorem.

Theorem 6852.2 (Kleene recursion theorem NameCert obligations)**.** *For every accepted* $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ *carrier* $K = (E, Q, W, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KleeneRecursionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed computable transformer row* E*, finite syntax and index quotation through* Q*, self-application exposure through* W*, fixed-index witnessing through* F*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6852.1. The only computability-facing row is E, the only quotation row is Q, and the only self-reference rows are W and F.

The structural rows do not add semantic content. Transport through H compares the displayed rows componentwise, while replay through C exposes the same finite transformer, quotation, self-application, and witness route to consumers.

Finally P and N source and name the packet. Since no carrier coordinate stores host partial-function equality, quotient program equality, oracle choice, Rice classification, or Myhill minimization, the NameCert surface is exhausted by the listed rows. □

Theorem 6852.3 (Kleene recursion theorem fixed-index window)**.** *Every fixed-index read from an accepted* $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ *packet must expose the finite route*

$$E \longmapsto Q \longmapsto W \longmapsto F$$

before the witness is consumed. The route reads the $\mathsf{Computable}^{\uparrow}$ *transformer, the* $\mathsf{LambdaCalc}^{\uparrow}$ *and* $\mathsf{FirstOrder}^{\uparrow}$ *quotation row, the self-application window, and the fixed-index witness as one* BHist *packet. It cannot be cited as semantic equality of partial functions, quotient equality of program indices, unrestricted partiality, oracle choice, Rice theorem, or Myhill theorem.*

Proof. Start with the obligation surface of Definition 6852.2. It makes E, Q, W, F the consumer-visible recursion-theorem rows and keeps H, C, P, N structural.

The witness row F is therefore not detached from the transformer. It is read only after the finite quotation row Q and self-application window W have been displayed over the same computable transformer row E.

The exclusions follow by coordinate exhaustion. The carrier contains no row for host semantic equality, quotient program equality, unrestricted partiality, oracle selection, Rice classification, or Myhill minimization, so none of those endpoints can be exported by the fixed-index window. □

Theorem 6852.4 (Kleene recursion theorem computable handoff)**.** *Any* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{LambdaCalc}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or NameCert consumer citing an accepted* $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ *packet receives only the finite handoff* E, Q, W, F, H, C, P, N*. The handoff permits the consumer to read a displayed computable transformer together with its quotation, self-application, fixed-index witness, transport, replay, provenance, and local name. It supplies no host semantic equality, unrestricted partiality, quotient program equality, selected oracle, Rice-style nontrivial semantic classifier, or Myhill-style global identification theorem.*

Proof. Read the carrier inventory of Definition 6852.1. The rows E, Q, W, F form the computability and self-reference surface, while H, C, P, N preserve, replay, source, and name that surface.

Apply the NameCert obligation theorem. A downstream consumer must see all eight rows before it can cite the packet as a recursion-theorem handoff, and the fixed-index window theorem keeps the witness attached to the displayed quotation and self-application route.

Thus the handoff is finite BEDC data. Because the refused semantic objects are absent from the carrier and from the structural rows, the consumer cannot recover them by citing the packet. □

Falsifiable prediction: A computability packet that omits the quotation row Q or self-application window W cannot satisfy the $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ NameCert obligation surface.

Independence witness: Computable, LambdaCalc, FirstOrder, NameCertMorphism: these siblings supply computability, syntax, coding, and transport rows, but none carries the eight-row self-reference packet E, Q, W, F, H, C, P, N, so none is bijective to $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ under the displayed NameCert obligations.

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneRecursionTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{KleeneRecursionTheorem}^{\uparrow}$ packet is generated from a displayed computable transformer row, finite syntax and index quotation, a self-application window, a fixed-index witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KleeneRecursionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6852.1, Definition 6852.2, Definition 6852.3, and Definition 6852.4 seed the finite recursion-theorem naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KleeneRecursionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host semantic equality, unrestricted partiality, quotient program equality, oracle choice, Rice theorem, Myhill theorem, Lean verification, or bridge checking for $\mathsf{KleeneRecursionTheorem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, quotation exactness, self-application window stability, fixed-index witness routing, computable handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KleeneRecursionTheoremUp.

Chapter 6853

A Concrete Naming Certificate for $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$ packet records the constructive dense-reflector route behind complete-metric Baire arguments as finite NameCert data. It sits after the theorem-level dense-open schedule of chapter 3570, the complete-metric inspection surface of chapter 3535, the dense-reflector completion route of chapter 3786, the indexed density row of chapter 1566, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names the finite route used by complete-metric Baire consumers; it does not assert a host generic point, countable-choice selector, quotient completion equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6853.1 (BaireCategoryDenseReflector carrier)**.** *A* $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, U, R, A, W, Q, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *dense-open schedule row, carrying the theorem-level finite route for complete-metric Baire reads.*
2. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection row, exposing complete-metric Cauchy service, metric comparison, and finite Baire surface data before any dense-reflector read.*
3. *R is the* $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ *route row, supplying the dense-reflector completion handoff used by completion-facing consumers.*
4. *A is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *indexed density row, naming the countable dense source only through displayed finite windows and tolerance data.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the dense-open schedule, indexed density requests, and completion readback.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the retained regular rational windows.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for dense-open refinements, countable-density requests, Cauchy-completion tolerances, and real-seal comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, U, R, A, W, Q, T have been displayed.*

9. *H records componentwise* hsame *transport for the theorem route, complete-metric surface, dense-reflector route, indexed density row, windows, readbacks, tolerances, Real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from dense-open schedules through complete-metric inspection, dense-reflector handoff, indexed density, finite windows, tolerance accounting, readback, and Real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}$ $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryDenseReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host intersections, selected generic points, quotient completion representatives, countable-choice schedules, open-cover compactness, or ambient completeness.

Theorem 6853.2 (BaireCategoryDenseReflector NameCert obligations)**.** *For every accepted* BaireCategoryDenseRe *carrier $D = (B, U, R, A, W, Q, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryDenseReflector}^{\uparrow}}$ *surface consists exactly of carrier admission for the theorem-level dense-open schedule, complete-metric Baire inspection, Cauchy completion dense-reflector handoff, countable dense source, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every complete-metric Baire consumer that asks for a dense-reflector route must inspect those rows before reading the Real-facing seal.*

Proof. Project the carrier of Definition 6853.1. The rows B, U, R, A are the only rows that introduce the Baire theorem route, complete-metric inspection surface, dense-reflector completion route, and countable dense source.

The finite observation and tolerance stage is W, Q, T, E. StreamName windows expose the retained dense-open and density requests, RegSeqRat supplies regular rational readback, DyadicRatCore records the tolerance ledger, and the terminal Real seal is reached only after those rows are displayed.

Finally read H, C, P, N. These rows preserve the same finite route under componentwise transport, replay, provenance, and local naming. Since the carrier has no coordinate for a host intersection, selected generic point, quotient completion representative, countable-choice schedule, open-cover compactness theorem, or ambient completeness witness, no accepted packet can export one. □

Theorem 6853.3 (BaireCategoryDenseReflector handoff)**.** *Every complete-metric Baire read that consumes an accepted* $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$ *packet factors through*

$$\mathsf{BaireCategoryTheorem}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{CauchyCompletionDenseReflector}^{\uparrow} \longmapsto \mathsf{CountableDens}$$

The handoff exports the finite dense-reflector route needed by downstream complete-metric Baire arguments, not a selected generic point, quotient completion equality, host intersection object, countable-choice selector, open-cover compactness theorem, or ambient completeness principle.

Proof. Start with the theorem route B and the complete-metric inspection row U. These rows make the dense-open schedule and complete-metric Cauchy service visible before a consumer can ask for dense-reflector data.

Next consume R and A. The Cauchy completion dense-reflector row supplies the completion-facing handoff, while the countable dense source row supplies indexed density data through the displayed finite packet rather than through a selected dense set.

The read then passes through W, Q, T, E and is transported by H, C, P, N. Thus every exported route is the displayed chain through BaireCategoryTheorem, CompleteMetricBaireSurface, CauchyCompletionDenseReflector, CountableDenseSubset, StreamName, RegSeqRat, DyadicRatCore, and Real rows. The excluded host and choice objects are not carrier fields and cannot be recovered from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryDenseReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireCategoryDenseReflector}^{\uparrow}$ packet is generated from a BaireCategoryTheorem dense-open schedule, a CompleteMetricBaireSurface inspection row, a CauchyCompletionDenseReflector handoff, a CountableDenseSubset indexed density row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryDenseReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6853.1, Definition 6853.2, and Definition 6853.3 seed the finite dense-reflector route behind complete-metric Baire arguments over theorem-level, complete-metric, dense-reflector, countable-density, window, readback, tolerance, Real, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryDenseReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host generic point, quotient completion equality, selected dense set, countable-choice schedule, open-cover compactness theorem, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric inspection, dense-reflector handoff, indexed density exposure, finite-window readback, tolerance accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BaireCategoryDenseReflectorUp.

Chapter 6854

A Concrete Naming Certificate for $\mathsf{LocatedCutApartness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCutApartness}^{\uparrow}$ packet records the finite route by which apartness evidence can be read from located-cut data before any real-name seal or comparison consumer receives it. It sits beside the located-cut carrier of chapter 404, the located-real comparison surface of **??**, the apartness-real evidence route of chapter 304, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite L10 boundary for reading apartness from located cuts; it does not import quotient cuts, trichotomy, selected representatives, countable choice, host real equality, or ambient Real completeness.

Definition 6854.1 (LocatedCutApartness carrier)**.** *A* $\mathsf{LocatedCutApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, U, G, D, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located-cut row read from* $\mathsf{LocatedCut}^{\uparrow}$*, exposing finite rational evidence below the cut.*
2. *U is the upper located-cut row read from* $\mathsf{LocatedCut}^{\uparrow}$*, exposing finite rational evidence above the same cut.*
3. *G is the positive gap row read through* $\mathsf{ApartnessReal}^{\uparrow}$*, recording the apartness witness that separates the displayed lower and upper observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the lower and upper observations at the requested precision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the located-cut observations are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback extracted from those windows.*
7. *Q is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row attached to the same located-cut request.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after the located-cut, apartness, dyadic, stream, readback, and located-real rows have been displayed.*
9. *H records componentwise* hsame *transport for L, U, G, D, S, R, Q, E and the structural rows.*

10. *C records displayed* Cont *replay from a located-cut apartness request through the same finite route.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCutApartness}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient cuts, host real equality, trichotomy, selected cut representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6854.2 (LocatedCutApartness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCutApartness}^{\uparrow}$ *carrier $A = (L, U, G, D, S, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCutApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed lower located-cut row, upper located-cut row, apartness gap row, dyadic tolerance ledger, finite stream windows, regular rational readback, located-real comparison row, Real seal, componentwise transport, displayed replay, provenance, and local naming; lower-upper exactness requiring L, U to remain attached to the same located-cut request; gap exactness requiring G before any apartness consumer can read Q or E; regular readback exactness requiring D, S, R before the seal; and ledger exhaustion excluding quotient cuts, trichotomy, selected representatives, countable choice, host equality, and ambient completeness.*

Proof. Project the carrier of Definition 6854.1. The rows $L, U, G, D, S, R, Q, E, H, C, P, N$ are the complete displayed packet, so carrier admission is a finite coordinate read.

Lower-upper exactness is forced by the shared $\mathsf{LocatedCut}^{\uparrow}$ source. The lower and upper rows cannot be separated from the same located-cut request because the apartness row G reads them as one finite gap witness.

The readback and seal clauses follow the route D, S, R, Q, E. The dyadic ledger fixes the requested precision, the stream row records the inspected windows, the regular rational row records the Cauchy-facing handoff, and the located-real comparison row is displayed before the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name only this packet. Since the excluded principles are not carrier coordinates, no local NameCert obligation can export them. □

Theorem 6854.3 (LocatedCutApartness gap handoff)**.** *Let $A = (L, U, G, D, S, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCutApartness}^{\uparrow}$ *packet. Every apartness consumer of A reads the positive gap evidence through*

$$L, U \longmapsto G \longmapsto D, S, R, Q.$$

Thus the consumer sees lower and upper located-cut observations, a displayed $\mathsf{ApartnessReal}^{\uparrow}$ *gap witness, the dyadic precision ledger, finite stream windows, regular rational readback, and located-real comparison data before any terminal seal is available.*

Proof. Start from the NameCert obligations of Definition 6854.2. The gap exactness clause places G downstream of the common lower and upper located-cut rows L, U.

Read the finite observation rows next. The dyadic ledger D supplies the tolerance, the stream row S supplies the inspected windows, and the regular rational row R supplies the Cauchy-facing readback.

The located-real comparison row Q is reached only after that finite route. Therefore an apartness consumer receives the route displayed in the statement and cannot replace it by trichotomy, host real equality, quotient cuts, selected representatives, countable choice, or ambient completeness. □

Theorem 6854.4 (LocatedCutApartness Real seal non-escape)**.** *Let* $A = (L, U, G, D, S, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCutApartness}^{\uparrow}$ *packet. Every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal receives only*

$$L, U, G, D, S, R, Q, E, H, C, P, N.$$

The seal exports no quotient cut, trichotomy branch, selected representative, countable-choice schedule, host equality row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply the gap handoff of Definition 6854.3. A consumer cannot reach the located-real comparison row Q until the lower and upper located-cut rows, apartness gap, dyadic ledger, stream windows, and regular readback have been displayed.

The terminal Real seal E is downstream of the same route by Definition 6854.2. After E is read, the only remaining coordinates are H, C, P, N.

Those structural rows preserve the displayed packet by componentwise transport, replay, provenance, and local naming. Since the excluded principles are absent from both the carrier and the obligation surface, no seal-facing consumer can obtain them from $\mathsf{LocatedCutApartness}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutApartness}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6854.1, Definition 6854.2, Definition 6854.3, and Definition 6854.4 seed the finite route reading apartness evidence from located-cut data through dyadic, stream, RegSeqRat, LocatedReal, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cuts, trichotomy, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper exactness, gap handoff, regular readback discipline, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCutApartnessUp.

Constructive story: A $\mathsf{LocatedCutApartness}^{\uparrow}$ packet is generated from lower and upper LocatedCut rows, an ApartnessReal positive gap witness, DyadicRatCore tolerance data, StreamName windows, RegSeqRat readback, a LocatedReal comparison row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6855

A Concrete Naming Certificate for $\mathsf{CauchyFilterCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Kelley on filter convergence, the packet records Cauchy filters as objects and refinement/limit-compatible maps as morphisms. It sits after the Cauchy-filter carrier of chapter 333, the uniform-completion interface of chapter 1473, the metric-completion route of chapter 1506, the regular rational readback of chapter 301, the finite stream-name windows of chapter 126, and the terminal real-name seal of chapter 38. The category packet is a finite interface for Cauchy-filter transport; it is not a quotient completion, a selected limit operation, countable choice, or an ambient complete object.

Definition 6855.1 (Cauchy filter category carrier)**.** *A* $\mathsf{CauchyFilterCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, M, I, A, H, C, P, N)$$

with the following displayed rows:

1. *O is the object row, listing accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packets and their finite basis, refinement, compatibility, and endpoint windows.*
2. *M is the morphism row, listing refinement maps and limit-compatible handoff maps between displayed Cauchy-filter objects.*
3. *I is the identity row, assigning to each displayed object its local refinement identity over the same basis and compatibility surface.*
4. *A is the composition row, composing two compatible morphism rows only when their middle Cauchy-filter object is displayed.*
5. *H records componentwise* hsame *transport for objects, morphisms, identities, composition, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from object admission through refinement maps, identity, composition, transport, provenance, and the local name.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCategory}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through O, M, I, A, H, C, P, N. It has no coordinate for quotient filter equality, selected limits, countable-choice schedules, ambient complete objects, quotient real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.CauchyFilterCategoryUp`

Theorem 6855.2 (Cauchy filter category NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCategory}^{\uparrow}$ *carrier $K = (O, M, I, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCategory}^{\uparrow}}$ *surface consists exactly of Cauchy-filter object exposure through O, refinement and limit-compatible morphism exposure through M, identity exposure through I, composition exposure through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6855.1. The accepted packet displays exactly the rows O, M, I, A, H, C, P, N.

The object row keeps the $\mathsf{CauchyFilter}^{\uparrow}$ data visible before any categorical read. The morphism row then records only refinement and limit-compatible maps between displayed objects; it does not create a quotient object or choose a limit.

The identity and composition rows are local BEDC rows over the same displayed objects. Transport, replay, provenance, and naming preserve that packet surface, so the listed rows exhaust the local certificate. □

Theorem 6855.3 (Cauchy filter category composition nonescape)**.** *Every composition read in an accepted* $\mathsf{CauchyFilterCategory}^{\uparrow}$ *packet factors through two displayed morphism rows and their shared middle* $\mathsf{CauchyFilter}^{\uparrow}$ *object before the composed row A is readable. The composition may be consumed by* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *routes as a finite refinement interface, but it exports no quotient completion, selected limit, countable choice, ambient complete object, quotient real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the morphism row M and the composition row A in Definition 6855.1. The composition row is available only after the source, middle, and target Cauchy-filter objects are displayed in O.

Use the NameCert obligations of Definition 6855.2. They force the object, morphism, identity, and composition rows to be exposed before transport, replay, provenance, and local naming.

The completion-facing siblings may consume this interface as finite refinement data. Since the carrier has no row for quotient filter equality, selected limits, choice schedules, ambient complete objects, quotient real equality, or Real completeness, composition cannot export any of those claims. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterCategory}^{\uparrow}$ consumer that composes Cauchy-filter maps must display the two morphism rows, the shared middle CauchyFilterUp object, transport, replay, provenance, and local NameCert row before any completion-facing read is named.

Independence witness: cauchyfilter, uniformcompletion, metriccompletion, regseqrat, streamname, real: none is bijective with $\mathsf{CauchyFilterCategory}^{\uparrow}$ because this packet jointly carries Cauchy-filter objects, refinement morphisms, identities, composition, transport, replay, provenance, and local naming rows, while each sibling supplies only one source or consumer surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCategory}^{\uparrow}$ packet is generated from CauchyFilter objects, refinement and limit-compatible morphisms, identity rows, composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyFilterCategory}^{\uparrow}}$, seed)

Scope closed: Definition 6855.1, Definition 6855.2, and Definition 6855.3 seed the categorical finite-data interface for Cauchy filters over object, morphism, identity, composition, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyFilterCategory}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.CauchyFilterCategoryUp`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, ambient complete objects, quotient real equality, Real$^{\uparrow}$ completeness, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, object exposure, refinement-morphism exposure, identity exposure, composition nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterCategoryUp.

Chapter 6856

A Concrete Naming Certificate for $\mathsf{JordanNormalForm}^{\uparrow}$

Following Horn–Johnson and standard finite-dimensional linear algebra, the $\mathsf{JordanNormalForm}^{\uparrow}$ packet records the finite matrix-block normal-form certificate read from a linear endomorphism over a displayed field source. It sits beside the matrix carrier of chapter 49, the linear-map surface of chapter 48, the vector-space source of chapter 47, the scalar field carrier of chapter 45, and the polynomial rows of chapter 50. The packet names only the finite block layout, nilpotent-chain ledger, and similarity-transport rows needed by a Jordan-form consumer; it does not import algebraic closure, quotient eigenspace equality, selected eigenvectors from a host theorem, or an ambient spectral theorem.

Definition 6856.1 (Jordan normal form carrier)**.** *A* $\mathsf{JordanNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, L, V, F, Q, B, N, S, H, C, P, A)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *row, recording the finite matrix entries, row-column index windows, and matrix multiplication requests used by the block read.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row, recording the endomorphism surface whose matrix presentation is inspected.*
3. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row, recording the displayed finite-dimensional vector-space basis and scalar-action windows.*
4. *F is the* $\mathsf{Field}^{\uparrow}$ *row, recording the scalar carrier through which the displayed entries and polynomial coefficients are read.*
5. *Q is the* $\mathsf{Poly}^{\uparrow}$ *row, recording the characteristic-polynomial and minimal-polynomial ledgers visible to the block scheduler.*
6. *B is the Jordan-block ledger, a finite list of block sizes, eigenvalue-name rows, and superdiagonal-one positions.*
7. *N is the nilpotent-chain ledger, recording the finite chain-length checks and kernel-power windows attached to each displayed block.*
8. *S is the similarity-transport row, recording the source presentation, target block presentation, and matrix-change rows used for the displayed conjugacy read.*

9. *H records componentwise* hsame *transport for M, L, V, F, Q, B, N, S and the local certificate rows.*

10. *C records displayed* Cont *replay along the route*

$$M, L, V, F, Q, B, N, S \longmapsto H, C, P, A.$$

11. *P is the* Pkg *provenance over* $\mathsf{JordanNormalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *A is the local* $\mathsf{NameCert}_{\mathsf{JordanNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, V, F, Q, B, N, S, H, C, P, A$. It has no coordinate for an algebraic-closure theorem, quotient eigenspace equality, a selected host basis, canonical uniqueness of the block order, or an ambient spectral theorem.

Theorem 6856.2 (Jordan normal form NameCert obligations)**.** *For every accepted* $\mathsf{JordanNormalForm}^{\uparrow}$ *carrier $J = (M, L, V, F, Q, B, N, S, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{JordanNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission, matrix exposure through M, linear-map exposure through L, vector-space basis exposure through V, field-scalar exposure through F, polynomial-ledger exposure through Q, Jordan-block ledger exposure through B, nilpotent-chain exposure through N, similarity-transport exposure through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through A.*

Proof. Project the carrier of Definition 6856.1. Its mathematical rows are exactly M, L, V, F, Q, B, N, S: the matrix presentation, the linear map, the finite vector-space source, the scalar field source, the polynomial ledgers, the block ledger, the nilpotent-chain ledger, and the similarity-transport row.

The block rows are not independent host algebra. The Jordan-block ledger B is read only through the matrix, field, and polynomial rows M, F, Q, while the nilpotent-chain ledger N records finite chain checks rather than quotient eigenspace equality or selected generalized eigenspaces.

Finally attach H, C, P, A. Transport and replay preserve the displayed rows componentwise, provenance names the sibling sources, and the local NameCert row adds no algebraic-closure or spectral-theorem coordinate. Thus the listed rows exhaust the NameCert obligations for the finite Jordan-form packet. □

Theorem 6856.3 (Jordan block similarity boundary)**.** *Every normal-form consumer of an accepted* $\mathsf{JordanNormalForm}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Mat}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Field}^{\uparrow},\ \mathsf{Poly}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

through $M, L, V, F, Q, B, N, S, H, C, P, A$ before treating the displayed block matrix as a Jordan-form certificate. The route exports only finite matrix entries, linear-map presentation data, vector-space basis rows, scalar and polynomial ledgers, block sizes, nilpotent-chain checks, similarity transport, replay, provenance, and local naming.

Proof. Start from Definition 6856.2. Any accepted read must expose the matrix, linear-map, vector-space, field, polynomial, block, nilpotent-chain, and similarity rows before the local NameCert row can be consumed.

The similarity transport remains finite. Row S records only the displayed source and target matrix presentations and the change-of-presentation rows needed for that read; it does not certify algebraic closure or provide quotient eigenspace equality.

The structural rows H, C, P, A preserve and name the same finite route. Since none of those rows contains a host spectral theorem, a selected eigenbasis outside the packet, or canonical uniqueness of block order, the exported boundary is exactly the displayed finite certificate. □

Falsifiable prediction: Every accepted $\mathsf{JordanNormalForm}^{\uparrow}$ consumer must display Matrix, LinearMap, VecSpace, Field, Polynomial, Jordan-block, nilpotent-chain, similarity-transport, hsame, Cont, Pkg, and local NameCert rows before reading a Jordan normal-form certificate.

Independence witness: matrix, linearmap, vecspace, field, polynomial: $\mathsf{JordanNormalForm}^{\uparrow}$ is not bijective to any listed sibling because this packet alone binds matrix entries, linear-map presentation, vector-space basis data, scalar-field rows, polynomial ledgers, finite Jordan-block layout, nilpotent-chain checks, similarity transport, replay, provenance, and local naming into one finite normal-form certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanNormalForm}^{\uparrow}$ packet is generated from Matrix rows, LinearMap rows, VecSpace basis windows, Field scalar rows, Polynomial ledgers, a finite Jordan-block layout, nilpotent-chain checks, similarity-transport rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6856.1, Definition 6856.2, and Definition 6856.3 seed the finite Jordan normal-form certificate over matrix, linear-map, vector-space, field, polynomial, block, nilpotent-chain, similarity, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim algebraic closure, quotient eigenspace equality, selected host eigenvectors, canonical uniqueness of Jordan-block order, ambient spectral theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, block-ledger exactness, nilpotent-chain projection, similarity-transport non-escape, componentwise transport, provenance, and local NameCert exactness for BEDC.Derived.JordanNormalFo

Chapter 6857

A Concrete Naming Certificate for $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$

The $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ packet realizes the residual-substitution confluence frontier of section 9386.3 as finite NameCert data. It sits beside the confluence inventory of chapter 6802, the residual-substitution boundary of chapter 2181, the bounded residual frontier of chapter 3808, the critical-path ledger of chapter 765, and the stream and regular-readback surfaces of chapter 126 and chapter 301. The packet schedules local diamond reads only through displayed residual-frontier and bounded-checker rows; it does not promote those reads to global confluence, arbitrary closed Church–Rosser, or a discharged substitution-compatibility theorem.

Definition 6857.1 (MetaCIC residual diamond scheduler carrier)**.** *A* $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, R, E, J, L, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, recording the closed source, target type, and finite reduction-window data inspected by the scheduler.*
2. *R is the residual-frontier row, recording the residual-substitution boundary retained before any local diamond read is accepted.*
3. *E is the candidate-SN evidence row, recording the finite candidate-mediated strong-normalization evidence used only to bound inspected branches.*
4. *J is the bounded local-join checker row, recording the finite peak, branch fuel, and accepted common-reduct witness for the inspected window.*
5. *L is the L10 source row, recording the source chapter, audit-map, and dependency route from which the scheduler reads its confluence-facing frontier.*
6. *O is the obstruction row, retaining the reason a requested global diamond or Church–Rosser endpoint is outside the packet.*
7. *H records componentwise* hsame *transport for T, R, E, J, L, O and the local certificate rows.*
8. *C records displayed* Cont *replay along the route*

$$T, R, E, J, L, O \longmapsto H, C, P, N.$$

9. *P is the* Pkg *provenance over* MetacicResidualDiamondScheduler$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *together with the sibling confluence-inventory, residual-frontier, critical-path, stream, and regular-readback sources named above.*

10. *N is the local* NameCert$_{\text{MetacicResidualDiamondScheduler}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, E, J, L, O, H, C, P, N$. It has no coordinate for full residual-substitution compatibility, an unrestricted parallel diamond, global Church–Rosser, evaluator equality, quotient equality of terms, subject-reduction closure, or kernel consistency.

Theorem 6857.2 (MetaCIC residual diamond scheduler NameCert obligations)**.** *For every accepted* MetacicResidualDiamondScheduler$^{\uparrow}$ *carrier $D = (T, R, E, J, L, O, H, C, P, N)$, the local* NameCert$_{\text{MetacicResidual}}$ *surface consists exactly of carrier admission for the typed-window row, residual-frontier row, candidate-SN evidence row, bounded local-join checker row, L10 source row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The Lean-side plan requires* Nontrivial MetacicResidualDiamondScheduler$^{\uparrow}$ *and* FieldFaithful MetacicResidualDiamondScheduler$^{\uparrow}$ *instances in* `TasteGate.lean`*, with the field-faithfulness witness preserving $T, R, E, J, L, O, H, C, P, N$ as distinct rows.*

Proof. Project the carrier of Definition 6857.1. The rows T, R, E, J, L, O expose the typed source window, residual frontier, candidate evidence, bounded checker, source route, and retained obstruction before any local diamond read can be consumed.

The join checker is bounded by construction. Row J records only the inspected peak, finite branch fuel, and displayed common-reduct witness, while R keeps the residual-substitution frontier visible and O records the refused global endpoint.

Finally H, C, P, N transport, replay, source, and name that same route. Nontrivial and FieldFaithful TasteGate instances must read the same row order, so a checked scheduler cannot erase the residual frontier, bounded checker, obstruction, or L10 source when presenting a diamond read. □

Theorem 6857.3 (MetaCIC residual diamond scheduler boundary)**.** *Every accepted local diamond read exported by a* MetacicResidualDiamondScheduler$^{\uparrow}$ *packet must expose*

$$T, R, E, J, L, O, H, C, P, N$$

before the read is consumed by a confluence-facing client. The scheduler may route a bounded residual-frontier peak to a displayed local join, but it exports no full BetaParallelSubstitutionCompatibility *proof, no arbitrary closed parallel diamond, no global closed Church–Rosser theorem, no evaluator equality, and no quotient equality of terms.*

Proof. Start from Definition 6857.2. The accepted packet exposes the typed window, residual frontier, candidate evidence, local checker, L10 source, and obstruction rows as separate fields.

The scheduler route is therefore local. A consumer can read a bounded common-reduct witness from J only after the frontier R and the source row L are displayed, and the obstruction row O remains attached to any stronger confluence request.

The structural rows preserve the same route. Since H, C, P, N do not add substitution compatibility, subject reduction, evaluator equality, or quotient term equality, the local diamond read cannot escape into a global confluence claim. □

Falsifiable prediction: Any accepted scheduler that exports a diamond read without residual-frontier, bounded-checker, obstruction, and L10-source rows refutes the $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ packet.

Independence witness: metaciccriticalpath, metacicclosuretrace, metacicnormalizationcertificaterouter, streamname, regseqrat: no listed sibling is bijective with $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$, because this packet alone binds typed windows, residual-frontier rows, candidate-SN evidence, bounded local-join checks, L10 source routing, obstruction retention, transport, replay, provenance, and local naming into one residual-diamond scheduler.

The Lean-side plan for `lean4/BEDC/Derived/MetacicResidualDiamondSchedulerUp/TasteGate.lean` must provide $\mathsf{Nontrivial}\ \mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ instances before any checked carrier-alignment theorem is cited for the packet. The nontrivial witness must separate an accepted bounded scheduler from a packet missing the residual frontier, and the field-faithfulness witness must preserve the typed-window, residual-frontier, candidate-SN, bounded-checker, L10-source, obstruction, transport, replay, provenance, and local-name rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$ packet is generated from a typed-window row, residual-frontier row, candidate-SN evidence row, bounded local-join checker row, L10 source row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6857.1, Definition 6857.2, and Definition 6857.3 seed the finite residual-diamond scheduler route over typed windows, residual-frontier evidence, candidate-SN bounds, local joins, L10 source routing, obstruction, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, full residual-substitution compatibility, arbitrary closed parallel diamond, global closed Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-frontier retention, candidate-SN evidence exposure, bounded local-join checking, L10 source routing, obstruction non-escape, componentwise transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetacicResidualDiamondSch

Chapter 6858

A Concrete Naming Certificate for $\mathsf{LocalEmpiricalResidue}^{\uparrow}$

The $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ packet realizes the local empirical residue branch of section 9414.25 as finite NameCert data. It sits beside the open-fit source of chapter 1132, the local closure gate of chapter 1312, the finite refutation surface of chapter 1169, the physical-induction gate of chapter 6534, and the predictive-descent packet of chapter 1182. Those siblings expose fit, local settlement, defeat, continuation stability, and descent; this packet names the residual finite data that remains visible when a locally settled empirical claim is still not final.

Definition 6858.1 (Local empirical residue carrier)**.** *A* $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (H, K, F, J, D, O, L, T, C, P, N)$$

with the following displayed rows:

1. *H is the finite observation-history row, naming the trace, probe bundle, observed signature, and model signature under inspection;*
2. *K is the classifier row used to judge agreement on the displayed trace;*
3. *F is the local fit or finite-failure row, recording the current empirical comparison under K;*
4. *J is the local closure row, read through* $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$*, stating that the current finite inquiry has been settled;*
5. *D is the failure-surface residue row, listing the displayed continuations or refinements that would defeat the present fit;*
6. *O is the open-observation row, recording that the packet does not enumerate all future histories;*
7. *L is the nonfinality ledger, naming the refused final-fit, final-induction-oracle, hidden-continuation, and total-confirmation reads;*
8. *T records componentwise* hsame *transport for H, K, F, J, D, O, L and the structural rows;*
9. *C records displayed* Cont *replay of the finite observation history, classifier, fit or failure judgment, local closure, residue surface, openness row, and nonfinality ledger;*
10. *P is the* Pkg *provenance over* $\mathsf{LocalEmpiricalResidue}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$, $\mathsf{FiniteObservationRefutation}^{\uparrow}$, $\mathsf{PhysicalInductionGate}^{\uparrow}$, $\mathsf{PredictiveDescent}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalEmpiricalResidue}^{\uparrow}}$ *row.*
The carrier has no coordinate for unrestricted empirical confirmation, selected future histories, total continuation enumeration, host physical-trajectory equality, or a final induction oracle.

Theorem 6858.2 (Local empirical residue NameCert obligations)**.** *For every accepted* $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ *carrier $R = (H, K, F, J, D, O, L, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocalEmpiricalResidue}^{\uparrow}}$ *surface consists exactly of finite observation-history exposure through H, classifier exposure through K, local fit or finite-failure reading through F, local closure exposure through J, failure-surface residue through D, open-observation retention through O, nonfinality refusal through L, componentwise transport through T, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 6858.1. The empirical rows are precisely H, K, F, J, D, O, L: a finite trace, a classifier, a local fit or failure judgment, a local closure row, a residue failure surface, an openness row, and a nonfinality ledger.

The rows D, O, L keep the local judgment from becoming final. The failure-surface residue records displayed ways the present fit could be defeated; the openness row refuses an enumeration of all future histories; and the nonfinality ledger records the refused oracle reads.

Finally T, C, P, N transport, replay, source, and name the same finite packet. Since none of these structural rows adds a selected continuation, a total confirmation judgment, host trajectory equality, or a final induction oracle, the listed rows exhaust the NameCert obligation surface. □

Theorem 6858.3 (Local empirical residue nonfinality)**.** *Every consumer of an accepted* $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ *packet that reads the local closure row J must also read the failure-surface residue D, open-observation row O, and nonfinality ledger L. Thus local empirical settlement may be used as finite evidence, but it cannot be exported as unrestricted future fit, total empirical confirmation, or a final induction oracle.*

Proof. Use the local closure boundary of Definition 9414.356. A finite inquiry may settle the displayed trace without settling all admissible future histories.

The carrier records this separation in the same packet. The row J is paired with D, O, L, so any accepted read of local closure exposes the remaining failure surface, the open-observation boundary, and the refused finality claims.

Transport, replay, provenance, and naming preserve that packet without adding new empirical coordinates. Therefore a downstream consumer can cite local empirical residue only as finite nonfinal evidence, not as a total confirmation or final induction principle. □

Falsifiable prediction: A consumer that reads local empirical closure from H, K, F, J, T, C, P, N while omitting the failure-surface residue D, open-observation row O, and nonfinality ledger L refutes the $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ carrier.

Independence witness: realityconstrainedopenfit, localempiricalclosuregate, finiteobservationrefutation, physicalinductiongate, predictivedescent: none is bijective to $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ because this packet alone binds finite observation history, classifier, local fit or failure, local closure, residue failure surface, open observation, nonfinality, transport, replay, provenance, and local naming as one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalEmpiricalResidue}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocalEmpiricalResidue}^{\uparrow}$ packet is generated from finite observation history, classifier, local fit or finite failure, local closure, failure-surface residue, open-observation, nonfinality, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalEmpiricalResidue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6858.1, Definition 6858.2, and Definition 6858.3 seed the finite local empirical residue packet and its nonfinality boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocalEmpiricalResidue}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted empirical confirmation, selected future histories, total continuation enumeration, host physical-trajectory equality, or a final induction oracle for LocalEmpiricalResidue$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, failure-surface residue exactness, open-observation retention, nonfinality refusal, transport, replay, provenance, and local naming for BEDC.Derived.LocalEmpiricalResidueU

Chapter 6859

A Concrete Naming Certificate for $\mathsf{MengerCurvature}^{\uparrow}$

Following Menger's metric curvature tradition, the $\mathsf{MengerCurvature}^{\uparrow}$ packet records the finite three-point curvature readout as BEDC NameCert data. It sits after the metric carrier of chapter 57, the dyadic rational ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite metric-geometry object for downstream comparison and completion routes; it does not claim global curvature bounds, quotient metric geometry, selected circumcentres, trichotomy of distances, or completeness.

Definition 6859.1 (Menger curvature carrier)**.** *A* $\mathsf{MengerCurvature}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, D, Q, R, E, H, C, P, N)$$

with the following rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *carrier and distance classifier.*
2. *T is a displayed ordered triple of carried metric points.*
3. *D is the finite distance ledger for the three pairwise distances of T.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger recording rational triangle-window tests for the three distances.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the reciprocal perimeter and area-budget approximants used by the curvature numerator and denominator.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal of the named curvature value.*
7. *H is componentwise* hsame *transport for the source points, distance rows, rational ledgers, readback rows, and seal.*
8. *C is displayed* Cont *replay for the finite curvature computation.*
9. *P is the* Pkg *provenance over* $\mathsf{MengerCurvature}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerCurvature}^{\uparrow}}$ *row.*

Theorem 6859.2 (Menger curvature NameCert obligations)**.** *For every accepted* $\mathsf{MengerCurvature}^{\uparrow}$ *carrier* $K = (M, T, D, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MengerCurvature}^{\uparrow}}$ *surface consists exactly of metric source exposure through* M*; finite triple exposure through* T*; three-distance ledger exposure through* D*; dyadic triangle-window comparison through* Q*; regular rational readback through* R*; terminal Real sealing through* E*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*;* Pkg *provenance through* P*; and local naming through* N*.*

Proof. Project the accepted carrier. The metric source row supplies only carried points and displayed distances, while the triple row selects the three finite endpoints that the curvature packet may inspect.

The dyadic and regular rational rows keep the computation finite. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger records triangle-window comparisons for the three distances, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the reciprocal perimeter and area-budget readbacks before the $\mathsf{Real}^{\uparrow}$ seal is formed.

Transport and replay preserve these same rows componentwise. Provenance and local naming therefore certify exactly the displayed curvature packet and do not add quotient equality, selected circumcentres, distance trichotomy, global curvature bounds, or completeness. □

Theorem 6859.3 (Menger curvature dyadic triangle handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MengerCurvature}^{\uparrow}$ *packet must read the finite route*

$$M, T, D, Q, R, E$$

in order: first the metric source and carried triple, then the three-distance ledger, then the dyadic triangle-window ledger, then the regular rational readback, and finally the Real seal. No consumer can bypass Q *and recover a curvature value directly from* M *and* E*.*

Proof. The distance ledger D is indexed by the displayed triple T, so the metric source M is only a source of finite carried distances. It does not by itself name a curvature row.

The handoff to Q records the triangle-window checks in the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger. These checks are the finite rational boundary that precedes the reciprocal perimeter and area-budget readbacks.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads those budgets before the $\mathsf{Real}^{\uparrow}$ seal E is admitted. Hence any downstream consumer that reaches E has already crossed the dyadic triangle handoff, and the packet provides no alternate route through quotient metric geometry, chosen circumcentres, trichotomy, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{MengerCurvature}^{\uparrow}$ consumer that reads a $\mathsf{Real}^{\uparrow}$ curvature seal exposes the $\mathsf{Met}^{\uparrow}$ source, carried triple, three-distance ledger, $\mathsf{DyadicRatCore}^{\uparrow}$ triangle-window comparisons, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before using the curvature value.

Independence witness: metric, dyadicratcore, regseqrat, real: $\mathsf{MengerCurvature}^{\uparrow}$ is not bijective to any listed sibling because this packet binds a three-point metric distance ledger, dyadic triangle-window comparisons, regular rational curvature readback, Real sealing, transport, replay, provenance, and local naming into one finite metric-curvature certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerCurvature}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MengerCurvature}^{\uparrow}$ packet is generated from a MetricUp source row, a finite carried triple, a three-distance ledger, DyadicRatCoreUp triangle-window comparisons, RegSeqRatUp reciprocal perimeter and area-budget readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MengerCurvature}\uparrow}$, seed)

Scope closed: Definition 6859.1, Definition 6859.2, and Definition 6859.3 seed the finite metric-curvature boundary for dyadic, regular-rational, and Real consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{MengerCurvature}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global curvature bounds, quotient metric geometry, selected circumcentres, distance trichotomy, metric completion, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite triple exposure, three-distance ledger exposure, dyadic triangle-window comparison, regular rational readback, Real sealing, transport, replay, provenance, and local naming for the displayed metric-curvature packet.

Chapter 6860

A Concrete Naming Certificate for $\mathsf{DualBasis}^{\uparrow}$

Following Bourbaki and Axler, the $\mathsf{DualBasis}^{\uparrow}$ packet records the finite-dimensional dual-basis surface by displaying a finite basis, its coordinate functionals, the scalar field, the vector-space carrier, the linear-map rows, and the matrix-coordinate readback before any linear-algebra consumer can use dual coordinates. It sits over the vector-space carrier of chapter 47, the linear-map interface of chapter 48, and the matrix-coordinate table of chapter 49. The packet is a NameCert object for finite coordinate evaluation; it is not a host algebraic dual, a choice-selected basis, a quotient of functionals, or an ambient finite-dimensionality theorem.

Definition 6860.1 (Dual-basis carrier)**.** *A* $\mathsf{DualBasis}^{\uparrow}$ *carrier over a displayed scalar field row* F *and finite vector-space row* V *is a finite* BHist *packet*

$$D = (F, V, B, \Lambda, L, M, H, C, P, N)$$

with the following rows:

1. F *is the scalar* $\mathsf{Field}^{\uparrow}$ *row. It supplies the carried scalar endpoints, scalar classifier, zero and one endpoints, addition, multiplication, and classifier transport used by coordinate equations.*
2. V *is the* $\mathsf{VecSpace}^{\uparrow}$ *carrier row over* F*. It supplies vector addition, scalar action, the vector classifier, and the carried endpoint predicate for the listed basis vectors and reconstructed vectors.*
3. B *is the finite basis row. It records a displayed natural number* n*, carried vectors* b_i *for* $i : \mathsf{Fin}(n)$*, and a finite coordinate-reconstruction ledger saying that every listed carried vector in the packet is classified with the finite sum of its displayed coordinates against the* b_i *rows.*
4. Λ *is the coordinate-functional row. It records carried maps* $\lambda_i : V \to F$ *for* $i : \mathsf{Fin}(n)$ *and the evaluation table* $\lambda_i(b_j) \sim_F \delta_{ij}$*.*
5. L *is the* $\mathsf{LinMap}^{\uparrow}$ *row for the coordinate functionals. Each* λ_i *is admitted as a linear map from* V *to the one-dimensional scalar target read through* F*, with additivity and scalar-compatibility rows displayed in the same packet.*
6. M *is the* $\mathsf{Mat}^{\uparrow}$ *coordinate row. It records the finite table whose* (i, j) *entry is the carried scalar comparison of* $\lambda_i(b_j)$ *with* δ_{ij}*, together with coordinate-column readback for every listed vector.*
7. H *records componentwise* hsame *transport for scalar rows, vector rows, basis entries, coordinate functionals, linear-map rows, matrix-coordinate entries, and local classifiers.*

8. *C records displayed* Cont *replay from basis inspection through coordinate evaluation, linear-map admission, matrix readback, reconstruction, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DualBasis}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DualBasis}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite basis and its coordinate-functional table. It has no coordinate for a host algebraic dual, a choice-selected basis, quotient functionals, arbitrary linear forms outside the listed rows, or ambient finite-dimensionality.

Theorem 6860.2 (Dual-basis NameCert obligations)**.** *For every accepted* $\mathsf{DualBasis}^{\uparrow}$ *carrier*

$$D = (F, V, B, \Lambda, L, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DualBasis}^{\uparrow}}$ *surface consists exactly of scalar admission through F, vector-space admission through V, finite basis admission through B, coordinate-functional admission through Λ, linear-map admission through L, matrix-coordinate readback through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 6860.1. The scalar row F and vector-space row V are displayed before the basis row is read, so every basis vector and every reconstructed vector is carried inside the same vector-space classifier.

Next read the coordinate-functional row Λ together with the $\mathsf{LinMap}^{\uparrow}$ row L. The row L supplies the additivity and scalar-compatibility obligations for each λ_i, while Λ supplies the evaluations on the displayed basis vectors.

Now inspect the $\mathsf{Mat}^{\uparrow}$ row M. Its entries are the finite coordinate table $\lambda_i(b_j) \sim_F \delta_{ij}$, and its column readback is attached to the same basis row B. Thus a coordinate consumer receives the basis, the functionals, and the table as one finite packet.

Finally H, C, P, N transport, replay, source, and name exactly these rows. Since no carrier coordinate names a host algebraic dual, a choice-selected basis, quotient functionals, arbitrary outside linear forms, or an ambient finite-dimensionality theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6860.3 (Dual-basis coordinate readback)**.** *Every coordinate consumer accepted by a* $\mathsf{DualBasis}^{\uparrow}$ *packet must factor through*

$$F, V, B, \Lambda, L, M, H, C, P, N.$$

The consumer first reads the scalar $\mathsf{Field}^{\uparrow}$ *and vector-space* $\mathsf{VecSpace}^{\uparrow}$ *rows, then the finite basis row, then the* $\mathsf{LinMap}^{\uparrow}$ *rows for the coordinate functionals, and then the* $\mathsf{Mat}^{\uparrow}$ *coordinate table. It receives coordinate values only through the displayed evaluation table and reconstruction ledger, not through a host dual space, an arbitrary functional oracle, quotient functional equality, or a selected basis outside the packet.*

Proof. Use Definition 6860.2. It places B, Λ, L, M after the scalar and vector-space rows and before any coordinate read is exported.

For a basis vector b_j, the coordinate value is read from the matrix entry in M that records $\lambda_i(b_j) \sim_F \delta_{ij}$. For a listed carried vector, the reconstruction ledger in B supplies the displayed

finite coordinate column, and the same matrix readback identifies which scalar rows are being consumed.

Therefore the coordinate route is finite and packet-local. The carrier never exposes a separate algebraic dual endpoint, a selector for a basis, or a quotient of functionals, so no consumer can bypass the displayed rows. □

Theorem 6860.4 (Dual-basis nonescape)**.** *No accepted* $\mathsf{DualBasis}^{\uparrow}$ *consumer can obtain a coordinate functional, coordinate matrix entry, or reconstruction coefficient without reading the scalar field row, vector-space row, finite basis row, linear-map row, matrix-coordinate row, transport row, replay row, provenance row, and local naming row displayed in Definition 6860.1.*

Proof. Assume a consumer asks for coordinate data from an accepted packet. By Definition 6860.2, the only coordinate-functional source is Λ, the only linearity source is L, and the only coordinate table is M.

The finite basis row B is the support on which Λ and M are interpreted. Without B, the expressions $\lambda_i(b_j)$ and the reconstruction column have no row in the packet; without F and V, their scalar and vector classifiers are not available.

The structural rows H, C, P, N are the only transport, replay, provenance, and naming routes. Hence any coordinate read that omits one of $F, V, B, \Lambda, L, M, H, C, P, N$ has no carrier coordinate from which to escape into host duality, arbitrary functional choice, quotient equality, or ambient finite-dimensionality. □

Falsifiable prediction: Every accepted $\mathsf{DualBasis}^{\uparrow}$ coordinate read factors through the displayed scalar field row, vector-space row, finite basis row, coordinate-functional row, linear-map row, matrix-coordinate row, transport, replay, provenance, and local naming rows; a request that supplies only a host algebraic dual or a selected basis is rejected by this certificate.

Independence witness: field, vecspace, linearmap, matrix: $\mathsf{DualBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries scalar field data, finite vector-space basis data, coordinate functionals, linear-map obligations, matrix-coordinate readback, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DualBasis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DualBasis}^{\uparrow}$ packet is generated from FieldUp scalar rows, VecSpaceUp carrier rows, finite basis rows, coordinate-functional rows, LinearMapUp admission rows, MatrixUp coordinate readback rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DualBasis}^{\uparrow}}$, seed)

Scope closed: Definition 6860.1, Definition 6860.2, Definition 6860.3, and Definition 6860.4 seed the finite dual-basis coordinate surface over field, vector-space, linear-map, and matrix rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DualBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host algebraic duals, arbitrary functional choice, quotient functional equality, selected bases outside the packet, norm equivalence, or ambient finite-dimensionality.

Upgrade path: Obligation closure requires binding the displayed scalar, vector-space, finite basis, coordinate-functional, linear-map, matrix-coordinate, transport, replay, provenance, and local naming rows against their BEDC primitive scope.

Chapter 6861

A Concrete Naming Certificate for $\mathsf{SequentialCoreflection}^{\uparrow}$

Following Franklin and Kelley on sequential spaces and coreflective topology, the $\mathsf{SequentialCoreflection}^{\uparrow}$ packet records the sequentially generated readback of a displayed space as finite BEDC NameCert data. It sits between the first-countable handoff of chapter 2157, the sequential closure route of chapter 1571, the cofinite-tail filter of chapter 1564, the finite stream windows of chapter 126, the regular rational sequence route of chapter 301, and the terminal real-name boundary of chapter 38. The certificate exposes the countable-name tests through which later closure and Real-facing consumers may inspect sequential reachability, without importing a host topological coreflection or a completed quotient space.

Definition 6861.1 (Sequential coreflection carrier)**.** *A* $\mathsf{SequentialCoreflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, Q, L, F, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source space row whose original open or closure observations are read only through displayed BEDC packet coordinates.*
2. *Q is the sequential-coreflection classifier row, recording that a point-name is tested by finite sequence windows and their eventual tail behaviour rather than by arbitrary host neighbourhoods.*
3. *L is the* $\mathsf{SequentialClosure}^{\uparrow}$ *row naming the sequence-closure test consumed by the carrier.*
4. *F is the* $\mathsf{FrechetFilter}^{\uparrow}$ *row recording cofinite-tail admission for the inspected sequence windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row displaying the finite prefixes and tail requests used by the test.*
6. *T is the first-countable handoff row, read through* $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ *when a countable neighbourhood base is displayed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row used when the inspected sequence data is passed to rational or metric readbacks.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, F, L, T, R have been displayed.*
9. *H records componentwise* hsame *transport for the source row, classifier row, closure row, tail filter, stream windows, first-countable handoff, rational handoff, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay along the route*

$$X, Q, L, F, W, T, R, E \longmapsto H, C, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{SequentialCoreflection}^{\uparrow}$, $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCoreflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host topological coreflection, quotient point equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6861.2 (Sequential coreflection NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCoreflection}^{\uparrow}$ *carrier*

$$S = (X, Q, L, F, W, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCoreflection}^{\uparrow}}$ *surface consists exactly of carrier admission, source-space exposure through X, sequential-coreflection classifier exposure through Q, sequential-closure exposure through L, cofinite-tail exposure through F, finite stream-window exposure through W, first-countable handoff exposure through T, regular rational handoff through R, real-name sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6861.1. Its mathematical rows are exactly the source row X, classifier row Q, sequential-closure row L, Frechet-tail row F, stream-window row W, first-countable handoff row T, regular rational handoff R, and Real seal E.

The sequential classifier is therefore read through displayed countable-name tests. Row W exposes only finite stream windows, row F supplies only cofinite-tail discipline, and row L records the sequence-closure test visible to the packet.

The first-countable and Real-facing routes remain downstream. Row T is consumed only when a countable base handoff has already been displayed, while R and E pass inspected sequence data to regular rational and Real consumers without selecting a subsequence or identifying quotient points.

Finally attach H, C, P, N. These structural rows transport, replay, source, and name the same finite dependency route, so the listed rows exhaust the local NameCert obligations. □

Theorem 6861.3 (Sequential coreflection first-countable handoff)**.** *Every accepted* FirstCountableSequentialClosure *read consumed by a* $\mathsf{SequentialCoreflection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{FrechetFilter}^{\uparrow},\ \mathsf{SequentialClosure}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as exposed by W, F, L, R, E and transported by H, C, P, N. The handoff lets a first-countable consumer replay the same finite sequence windows as a sequential-coreflection read, but it exports no host topology coreflection, quotient point equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the first-countable row T in Definition 6861.1. A consumer may use it only after the source row X and sequential classifier row Q have exposed the sequence test being replayed.

The countable-name data is then read through W, F, L. Finite prefixes and tail requests come from the $\mathsf{StreamName}^{\uparrow}$ window row W, eventual-tail discipline comes from $\mathsf{FrechetFilter}^{\uparrow}$ row F, and the closure test itself is the displayed $\mathsf{SequentialClosure}^{\uparrow}$ row L.

The Real-facing handoff is downstream from the same windows. Row R repacks the inspected sequence observations as regular rational data when such a readback is requested, and row E seals the resulting Real-facing boundary.

Thus the first-countable handoff is a replay of the displayed sequence route, not a separate topology theorem. The structural rows H, C, P, N preserve that route and add no selected subsequence, quotient point equality, countable-choice schedule, or ambient completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{SequentialCoreflection}^{\uparrow}$ read factors through source-space, sequential-classifier, SequentialClosure, FrechetFilter, StreamName, FirstCountableSequentialClosure, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows before exporting a sequentially generated closure read.

Independence witness: first-countable-sequential-closure, sequentialclosure, frechetfilter, streamname, regseqrat, real: $\mathsf{SequentialCoreflection}^{\uparrow}$ is not bijective to any listed sibling because this packet alone binds the source row, sequential-coreflection classifier, sequence-closure route, cofinite-tail filter, finite stream windows, first-countable handoff, regular rational readback, Real seal, transport, replay, provenance, and local naming into one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCoreflection}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCoreflection}^{\uparrow}$ packet is generated from a source-space row, a sequential-coreflection classifier row, SequentialClosure rows, FrechetFilter tail rows, StreamName windows, a FirstCountableSequentialClosure handoff, RegSeqRat readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCoreflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6861.1, Definition 6861.2, and Definition 6861.3 seed the finite sequential-coreflection certificate over countable-name sequence tests, closure rows, tail filters, first-countable handoff rows, rational readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCoreflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topology coreflection, quotient point equality, selected subsequences, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequential-classifier exposure, SequentialClosure exactness, FrechetFilter tail discipline, StreamName window replay, first-countable handoff, RegSeqRat and Real non-escape, transport, provenance, and local naming for BEDC.Derived.SequentialCoreflectionUp.

Chapter 6862

A Concrete Naming Certificate for $\mathsf{GromovHyperbolicMetric}^{\uparrow}$

Following Gromov's hyperbolic metric spaces and the four-point formulation of coarse negative curvature, the $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ packet records a finite metric carrier whose hyperbolicity is read through displayed distance comparisons and bounded four-point slack. It sits after the metric source of chapter 57, the length-space route of chapter 2082, the geodesic metric surface of chapter 2213, and the existing hyperbolic metric packet of chapter 3474; its boundary-facing consumers include chapter 3645, chapter 3732, and chapter 6470. The packet names only finite four-point, geodesic, and boundary-handoff rows, not a host geodesic space, selected boundary compactification, quotient of rays, or ambient completion theorem.

Definition 6862.1 (Gromov hyperbolic metric carrier)**.** *A* $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, L, E, \Delta, Q, S, B, H, C, P, N)$$

with displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-observation row.*
2. *L is the* $\mathsf{LengthSpace}^{\uparrow}$ *path-length readback row when a length route is present.*
3. *E is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *geodesic witness row when geodesic segments are displayed.*
4. *Δ is the nonnegative hyperbolicity-bound row used by the four-point comparisons.*
5. *Q is the four-point comparison ledger over displayed quadruples.*
6. *S is the stability row recording that enlarging the displayed bound keeps the same accepted metric source.*
7. *B is the boundary-handoff row available to* $\mathsf{BusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, *and* $\mathsf{GromovBoundary}^{\uparrow}$ *consumers.*
8. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GromovHyperbolicMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, E, \Delta, Q, S, B, H, C, P, N$. It has no coordinate for a selected geodesic model, quotient ideal boundary, host compactification, or ambient completion principle.

Theorem 6862.2 (Gromov hyperbolic metric NameCert obligations)**.** *For every accepted* $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ *carrier* $G = (M, L, E, \Delta, Q, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GromovHyperbolicMetric}^{\uparrow}}$ *surface consists exactly of metric-source admission, optional length-space readback, optional geodesic witness exposure, hyperbolicity-bound admission, four-point comparison exposure, bound-stability reading, boundary-handoff exposure, componentwise transport, displayed replay, package provenance, and local naming. Every downstream consumer must factor through*

$$M, L, E, \Delta, Q, S, B, H, C, P, N$$

before it reads a hyperbolic metric datum.

Proof. Project the carrier of Definition 6862.1. The metric source M is visible before any four-point comparison is inspected, and the length and geodesic rows L, E are displayed as route data rather than as a host geodesic space.

The bound row Δ and comparison ledger Q provide the Gromov-hyperbolic content. They inspect only finite quadruples and the displayed distance observations, so the packet records hyperbolicity as BEDC comparison data.

The stability and boundary rows S, B then route the same accepted metric surface to boundary consumers. The structural rows H, C, P, N transport, replay, source, and name only those displayed rows, leaving no coordinate for selected geodesic models, quotient boundaries, host compactifications, or ambient completion theorems. □

Theorem 6862.3 (Gromov hyperbolic metric four-point stability)**.** *Every accepted* $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ *carrier remains accepted when a consumer replaces the displayed hyperbolicity-bound row* Δ *by a transported row* Δ' *whose finite comparison ledger dominates the same quadruple reads. The accepted route is still*

$$M, L, E, \Delta', Q, S, B, H, C, P, N,$$

and the replacement does not add a host proof of global curvature, a selected geodesic realization, a quotient boundary, or ambient metric completion.

Proof. Start from the four-point ledger Q. It is indexed only by displayed metric observations from M, with optional length and geodesic support read through L and E.

The stability row S records the admissible transport from Δ to Δ' for those same finite quadruple reads. Since the row changes only the displayed bound, the metric source and the comparison ledger remain the same BEDC route.

Finally apply the structural rows H, C, P, N. They replay the transported bound and preserve the boundary handoff B, but they do not introduce a host curvature theorem, selected geodesic model, quotient boundary, or completion object. □

Theorem 6862.4 (Gromov hyperbolic metric geodesic boundary route)**.** *Every boundary-facing consumer of an accepted* $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row* M*, the length or geodesic rows* L, E *when present, the hyperbolicity bound* Δ*, the four-point ledger* Q*, and the boundary-handoff row* B *before it can use the packet in* $\mathsf{BusemannBoundary}^{\uparrow}$*,* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$*, or* $\mathsf{GromovBoundary}^{\uparrow}$*. The handoff exports finite metric and comparison rows, not a completed ideal boundary, quotient of rays, selected geodesic representative, or host compactification.*

Proof. Read the metric source through chapter 57, then the length and geodesic support through chapter 2082 and chapter 2213 when those rows are present.

Next expose Δ and Q. These rows are the only hyperbolicity evidence available to the packet, so a boundary consumer cannot bypass the finite four-point comparison surface.

The boundary handoff B is then consumed by the Busemann and Gromov boundary chapters named above. Because B is transported only through H, C, P, N, the export remains a finite metric comparison route and not a completed ideal boundary, quotient of rays, selected geodesic representative, or host compactification. □

Closure status for $\mathsf{NameCert}_{\mathsf{GromovHyperbolicMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ packet is generated from MetricUp distance rows, optional LengthSpaceUp and GeodesicMetricUp support rows, a displayed hyperbolicity-bound row, a finite four-point comparison ledger, bound-stability rows, boundary-handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GromovHyperbolicMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6862.1, Definition 6862.2, Definition 6862.3, and Definition 6862.4 seed the finite four-point metric packet over metric, length-space, geodesic, stability, and boundary-handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GromovHyperbolicMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host geodesic space, selected boundary compactification, quotient of rays, global curvature theorem, ambient completion theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-point ledger exactness, bound stability, metric and geodesic handoff, boundary route exactness, transport, replay, provenance, and local naming for BEDC.Derived.GromovHyperbolicMetricUp.

Chapter 6863

A Concrete Naming Certificate for $\mathsf{BrentSalaminPi}^{\uparrow}$

Following Brent and Salamin, the $\mathsf{BrentSalaminPi}^{\uparrow}$ packet records the finite Brent–Salamin arithmetic-geometric-mean route for a named value of π as an L10 Real and Completion computation object. It consumes the iteration discipline of chapter 1973, the terminal real-name seal of chapter 38, the dyadic arithmetic and budget carrier of chapter 291, the finite observation windows of chapter 126, and the regular rational readback route of chapter 301. The packet names only displayed AGM iteration rows, Real seals, dyadic error ledgers, finite windows, regular readback, transport, replay, provenance, and local naming; it does not export ambient completeness, quotient stream equality, selected limits, trichotomy, Lean verification, or bridge data.

Definition 6863.1 (Brent-Salamin pi carrier)**.** *A* $\mathsf{BrentSalaminPi}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, R, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ *iteration row, recording the finite Brent–Salamin recurrence windows used for the current precision request.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the Real-name seals for the current approximation, correction term, and exported π read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, storing the dyadic arithmetic cells, error radii, rounding slack, and precision ledger consumed by A, R, S, Q.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the requested AGM iterates, correction rows, and terminal approximation reads.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed stream windows and dyadic budgets.*
6. *H records componentwise* hsame *transport for the AGM row, Real seals, dyadic ledger, windows, readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of each accepted precision request through*

$$A \longmapsto D \longmapsto S \longmapsto Q \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BrentSalaminPi}^{\uparrow}$, $\mathsf{ArithmeticGeometricMean}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BrentSalaminPi}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, R, D, S, Q, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, trichotomy, unrestricted analytic convergence, Lean evidence, or standard-bridge data.*

Theorem 6863.2 (Brent-Salamin pi NameCert obligations)**.** *For every accepted* $\mathsf{BrentSalaminPi}^{\uparrow}$ *carrier*

$$B = (A, R, D, S, Q, H, C, P, N),$$

the visible $\mathsf{NameCert}_{\mathsf{BrentSalaminPi}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ *iteration row A, the* $\mathsf{Real}^{\uparrow}$ *seal row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the structural rows H, C, P, N;*

2. *recurrence routing, requiring each public Brent–Salamin precision read to pass through the displayed AGM row A before any Real seal is consumed;*

3. *dyadic accounting, requiring D to store exactly the dyadic arithmetic cells, error radii, rounding slack, and precision ledger used by A, R, S, Q;*

4. *finite-window readback, requiring S and Q to mediate every exported approximation before the terminal Real-facing row R is visible;*

5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*

6. *non-escape for ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, trichotomy, unrestricted analytic convergence, Lean verification, and standard-bridge data.*

No Brent–Salamin pi obligation is stored outside these rows.

Proof. Project the carrier of Definition 6863.1. The AGM row, Real seal row, dyadic budget, finite windows, and regular readback are the only computation-facing coordinates, while H, C, P, N are the only structural coordinates.

The recurrence route is forced by the displayed tuple. A public approximation request first reads A, then uses the dyadic cells and error radii in D to select the finite windows S and the regular readback Q before reaching R.

Transport, replay, provenance, and local naming are confined to H, C, P, N. Since none of the excluded completeness, quotient, limit-selection, order-decision, analytic-convergence, verification, or bridge objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6863.3 (Brent-Salamin pi dyadic error ledger)**.** *For every accepted* $\mathsf{BrentSalaminPi}^{\uparrow}$ *carrier*

$$B = (A, R, D, S, Q, H, C, P, N),$$

every public error claim about the exported π approximation factors through the displayed dyadic budget row D and the finite-window rows S, Q. Equivalently, a consumer can read an error assertion only along the route

$$A \longmapsto D \longmapsto S \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

The packet therefore exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, quotient stream equality, selected limit, trichotomy witness, or hidden analytic convergence theorem.*

Proof. Start with the accepted carrier $B = (A, R, D, S, Q, H, C, P, N)$. The AGM row A names the finite recurrence data, but it does not by itself expose an error claim; the dyadic row D is the only coordinate carrying radii, rounding slack, and precision cells.

The public observation rows are S and Q. The finite stream windows in S select the visible recurrence and correction entries, and the regular rational readback in Q is the only handoff from those windows to the terminal Real-facing row R.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Because the carrier has no coordinate for ambient completeness, quotient stream equality, selected limits, trichotomy, or hidden analytic convergence, every public error claim factors through D, S, Q before the Real seal is read. □

Falsifiable prediction: Every accepted $\mathsf{BrentSalaminPi}^{\uparrow}$ precision read that exposes an error claim also exposes an ArithmeticGeometricMeanUp iteration row, a DyadicRatCore budget row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: arithmetic-geometric-mean, real, dyadicratcore, streamname, regseqrat: $\mathsf{BrentSalaminPi}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Brent–Salamin AGM iteration rows, dyadic error budgets, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BrentSalaminPi}^{\uparrow}}$.

Constructive story: A $\mathsf{BrentSalaminPi}^{\uparrow}$ packet is generated from one ArithmeticGeometricMeanUp iteration row, one RealUp seal row, one DyadicRatCore budget row, finite StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BrentSalaminPi}^{\uparrow}}$, seed)

Scope closed: Definition 6863.1, Definition 6863.2, and Definition 6863.3 seed the finite Brent–Salamin pi computation certificate surface for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BrentSalaminPi}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.BrentSalaminPiUp`

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, trichotomy, unrestricted analytic convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, recurrence routing, dyadic accounting, finite-window readback, Real-seal non-escape, transport/replay/provenance coverage, and local naming for BEDC.Derived.BrentSalaminPiUp.

Chapter 6864

A Concrete Naming Certificate for $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges and Troelstra on fan bars, regular sequences, and constructive real completion, the $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ packet records the finite point where a fan-modulus depth ledger becomes a Real-completion boundary certificate. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the fan/bar sibling route of chapter 1707. The packet names only finite windows, bar-depth bounds, dyadic tolerances, regular readbacks, Real seals, structural transport, replay, provenance, and local naming.

Definition 6864.1 (Fan-modulus Real-completion boundary carrier)**.** *A* $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, D, S, Q, R, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the fan/bar modulus row, read through* $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{BishopFanModulus}^{\uparrow}$*, recording the finite branch-depth bound available to the boundary.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to that same depth.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule whose entries are inspected only up to the displayed finite depth.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from those windows.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached after F, D, S, Q have been displayed.*
6. *W is the Real-completion boundary row, recording which downstream completion consumer may read the sealed finite window without selecting a limit.*
7. *H records componentwise* hsame *transport for the fan/bar, dyadic, stream, regular-readback, Real-seal, boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a consumer request through $F, D, S, Q, R, W, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BishopFanModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, D, S, Q, R, W, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6864.2 (Fan-modulus Real-completion boundary stability)**.** *Let $B = (F, D, S, Q, R, W, H, C, P, N)$ and $B' = (F', D', S', Q', R', W', H', C', P', N')$ be accepted* $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ *carriers related by the componentwise* hsame *transport recorded in H and H'. Then every Real-completion boundary read through B is transported to the corresponding read through B' with the same finite fan/bar depth, dyadic tolerance ledger, StreamName window schedule, RegSeqRat readback, Real seal, and boundary row.*

Proof. Project the two accepted carriers. Their analytic rows are exactly F, D, S, Q, R, W and F', D', S', Q', R', W'.

The componentwise hsame rows H and H' identify the fan/bar depth, dyadic tolerances, stream windows, regular rational readbacks, Real seals, and boundary rows coordinate by coordinate. Displayed Cont replay then preserves the same route for any downstream completion consumer.

The provenance and local naming rows P, N and P', N' record the same finite dependency list. Hence transport cannot introduce selected limits, quotient streams, countable-choice schedules, open-cover compactness, or ambient Real completeness. □

Theorem 6864.3 (Fan-modulus Real-completion boundary NameCert obligations)**.** *For every accepted* $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ *carrier $B = (F, D, S, Q, R, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, fan/bar depth admission through F, dyadic tolerance admission through D, finite StreamName window admission through S, RegSeqRat readback through Q, Real seal admission through R, Real-completion boundary admission through W, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 6864.1. The projected packet contains exactly $F, D, S, Q, R, W, H, C, P, N$.

The fan/bar row F bounds the branch-depth surface before any analytic read occurs. The dyadic, stream, and regular rational rows D, S, Q then give the finite tolerance and readback data that the Real seal may inspect.

The terminal rows R and W expose only the sealed Real boundary and the downstream completion boundary. Finally H, C, P, N transport, replay, source, and name that same finite route, so the listed obligations exhaust the local NameCert surface. □

Theorem 6864.4 (Fan-modulus Real-completion boundary L10 consumption)**.** *Every L10 Real-completion consumer of an accepted* $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ *carrier factors through*

$$\mathsf{FanTheorem}^{\uparrow}, \mathsf{BishopFanModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto W.$$

The consumer therefore receives a finite fan/bar modulus, dyadic tolerances, finite stream windows, regular rational readback, a terminal Real seal, and a displayed completion boundary row before any completion-facing read is exported.

Proof. Start from the carrier of Definition 6864.1. The consumer can inspect the boundary row W only after the preceding rows F, D, S, Q, R have been displayed.

The fan/bar row F supplies the finite modulus through the sibling route of chapter 1707. The tolerance and observation rows D, S, Q then pass through chapter 291, chapter 126, and chapter 301.

Only after those finite windows are visible may the Real seal of chapter 38 and the boundary row W be read. Since no carrier coordinate stores open-cover compactness, selected limits, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the L10 consumer receives only the displayed finite boundary. □

Falsifiable prediction: Every accepted FanModulusRealCompletionBoundaryUp L10 read displays fan/bar depth, DyadicRatCore tolerance, StreamName windows, RegSeqRat readback, a Real seal, Real-completion boundary, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: bishop-fan-modulus, dyadicratcore, streamname, regseqrat, real: FanModulusRealComple is not bijective to any listed sibling because it jointly carries fan/bar depth, dyadic tolerance, finite stream scheduling, regular rational readback, Real sealing, and a Real-completion boundary row while refusing compactness, choice, selected limits, quotient equality, host equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6864.1, Definition 6864.2, Definition 6864.3, and Definition 6864.4 seed the finite fan-modulus Real-completion boundary through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and fan/bar sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, open-cover compactness, countable choice, quotient stream equality, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan/bar depth exactness, dyadic tolerance coverage, stream-window coverage, regular-readback coverage, Real seal admission, completion-boundary admission, transport, replay, provenance, and local naming for BEDC.Derived.FanModulusRealCompletionBoundaryUp.

Constructive story: A $\mathsf{FanModulusRealCompletionBoundary}^{\uparrow}$ packet is generated from a finite fan/bar modulus row, a DyadicRatCore tolerance row, StreamName windows, RegSeqRat readback, a Real seal, a Real-completion boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Chapter 6865

A Concrete Naming Certificate for $\mathsf{BishopIntervalChoice}^{\uparrow}$

Following Bishop and Bridges on nested interval arguments, the $\mathsf{BishopIntervalChoice}^{\uparrow}$ packet records the finite induction surface used when a closed bounded interval route is refined through lawlike dyadic windows. It sits over the interval carrier of chapter 1591, the dyadic arithmetic core of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. It differs from the finite token extraction of chapter 6054 and the compact finite-net surface of chapter 5208: the named object is the stepwise nested-window induction route, not a selected finite subcover, a chosen interval point, or an appeal to ambient compactness.

Definition 6865.1 (Bishop interval choice carrier)**.** *A* $\mathsf{BishopIntervalChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including the lower endpoint, upper endpoint, endpoint-order, containment, stream, readback, and Real-seal data available from that packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *halving and containment ledger used to refine one closed bounded interval window to its next nested subwindow.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records exactly the finite name coordinates inspected at each displayed induction step.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those window coordinates.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the closed interval, dyadic, stream, and regular-rational rows have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, dyadic ledger, stream windows, regular readbacks, Real seal, replay route, provenance row, and local naming row.*
7. *C records displayed* Cont *replay of each nested-window induction step through I, D, S, Q, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalChoice}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, R, H, C, P, N$. It has no coordinate for Heine–Borel compactness, trichotomy, selected points, quotient real equality, countable choice, or ambient Real completeness.

Theorem 6865.2 (Bishop interval choice NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalChoice}^{\uparrow}$ *carrier*

$$B = (I, D, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalChoice}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, dyadic nested-window refinement,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation consumed by the certificate is read from $I, D, S, Q, R, H, C, P, N$.*

Proof. Project the carrier of Definition 6865.1. Its only visible rows are the closed bounded interval source, dyadic refinement ledger, stream windows, regular rational readbacks, Real seal, and the structural rows H, C, P, N.

The NameCert obligation surface is therefore row exact. Carrier admission reads I, nested-window refinement reads D, stream and regular-rational exposure read S and Q, and the terminal real-name boundary reads R only after those rows have been displayed.

The transport, replay, provenance, and local naming clauses are supplied by H, C, P, N. Since no row stores Heine–Borel compactness, trichotomy, selected points, quotient real equality, countable choice, or ambient completeness, none of those objects can become a certificate obligation. □

Theorem 6865.3 (Bishop interval choice nested-window induction)**.** *Every accepted* $\mathsf{BishopIntervalChoice}^{\uparrow}$ *carrier supports induction over its displayed nested-window route: the base step reads the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row I, and each successor step refines the current closed window using the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D before exposing the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window row S and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q. The induction state remains inside I, D, S, Q, H, C, P, N and does not select an interval point, finite subcover, quotient representative, trichotomy branch, countable-choice family, or ambient completion coordinate.*

Proof. The base step is the source projection. It reads the displayed closed bounded interval row I and no downstream seal or selected point.

For the successor step, use the dyadic ledger D. The ledger names the finite halving and containment data needed to pass from one displayed closed window to the next nested window.

The stream and readback rows then expose the successor window. The row S records the inspected finite name coordinates, and Q records the corresponding regular rational readback before any Real-facing consumer may inspect a seal.

The structural rows H, C, P, N transport, replay, source, and name the same induction route. Thus the induction is a finite nested-window recursion inside the packet, not an invocation of finite subcover selection, quotient equality, trichotomy, countable choice, or Real completeness. □

Theorem 6865.4 (Bishop interval choice Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{BishopIntervalChoice}^{\uparrow}$ *carrier factors through the closed interval source I, dyadic nested-window ledger D,* $\mathsf{StreamName}^{\uparrow}$ *window row S, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q before reading the terminal Real seal R. The seal exports only the displayed nested-window induction boundary and no Heine–Borel compactness, open-cover compactness, selected finite subcover, selected real point, trichotomy, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Start with the NameCert obligations of Definition 6865.2. They force every accepted consumer read to begin from the same finite rows $I, D, S, Q, R, H, C, P, N$.

Apply nested-window induction. The consumer route reaches S and Q only by following the source interval and dyadic refinement rows step by step, so the Real seal cannot be detached from the displayed induction boundary.

The seal row R is therefore downstream of closed interval source, dyadic refinement, stream exposure, and regular-rational readback. The rows H, C, P, N preserve this boundary as transport, replay, provenance, and local naming.

Since the carrier contains no coordinate for compactness selection, selected points, trichotomy, quotient equality, countable choice, or ambient completeness, the Real and Completion consumer receives exactly the displayed nested-window seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalChoice}\uparrow}$.

Constructive story: A $\mathsf{BishopIntervalChoice}^{\uparrow}$ packet is generated from ClosedBoundedInterval, DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalChoice}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6865.1, Definition 6865.2, Definition 6865.3, and Definition 6865.4 seed the choice-free nested-window induction route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalChoice}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, Heine–Borel compactness, open-cover compactness, selected finite subcovers, selected interval points, trichotomy, quotient real equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic nested-window induction, StreamName and RegSeqRat readback exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalChoiceUp.

Chapter 6866

A Concrete Naming Certificate for $\mathsf{PathMetric}^{\uparrow}$

Following Burago–Burago–Ivanov on intrinsic metrics and Gromov on metric structures read through finite chains, the $\mathsf{PathMetric}^{\uparrow}$ packet records the path-distance surface obtained from a displayed metric carrier and finite curve-fragment rows. It sits between the metric source of chapter 57, the pointwise and modulus rows of chapter 130, the length-space consumer of chapter 2082, the segment-witness refinement of chapter 2213, and circle-source consumers such as chapter 58. The packet names path-distance data without assuming selected geodesics, quotient path spaces, host curve equality, or completion of the ambient metric carrier.

Definition 6866.1 (PathMetric carrier)**.** *A* $\mathsf{PathMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, F, L, D, B, H, C, Q, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *source row, carrying the endpoint classifier and the distance observations used by every path-fragment read.*
2. *F is a finite family of displayed curve-fragment rows, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *when a fragment is presented by a finite graph or modulus ledger.*
3. *L is the fragment-length ledger, assigning each displayed fragment a finite length budget over the same metric source.*
4. *D is the path-distance row, recording the requested endpoint distance as the infimum-facing finite lower-envelope surface generated by the listed fragments.*
5. *B is the boundary handoff row for* $\mathsf{LengthSpace}^{\uparrow}$*,* $\mathsf{GeodesicMetric}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *consumers: it records which endpoint, fragment, and circle-source reads are visible before a downstream path-distance claim is used.*
6. *H records componentwise* hsame *transport for the metric source, fragment family, length ledger, path-distance row, boundary handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint reads through fragment inspection, length-budget comparison, path-distance exposure, and downstream handoff.*
8. *Q is the* Pkg *provenance over* $\mathsf{PathMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{LengthSpace}^{\uparrow}$*,* $\mathsf{GeodesicMetric}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PathMetric}^\uparrow}$ *row.*

The classifier compares accepted packets only through $M, F, L, D, B, H, C, Q, N$. It has no coordinate for selected geodesics, quotient path spaces, host equality of curves, unlisted path families, or ambient metric completion.

Theorem 6866.2 (PathMetric NameCert obligations)**.** *For every accepted* $\mathsf{PathMetric}^\uparrow$ *carrier $P = (M, F, L, D, B, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{PathMetric}^\uparrow}$ *surface consists exactly of carrier admission for the displayed metric source, curve-fragment family, fragment-length ledger, path-distance row, boundary handoff, componentwise transport, displayed replay, package provenance, and local naming. Every path-distance read must expose $M, F, L, D, B, H, C, Q, N$ before a downstream consumer may use it.*

Proof. Project the carrier of Definition 6866.1. The metric source M, the fragment family F, the fragment-length ledger L, the path-distance row D, and the handoff row B are all displayed before transport and naming are read.

The fragment route is finite. If a fragment is supplied by a graph or modulus row, it is read through the $\mathsf{ContinuousMap}^\uparrow$ boundary of chapter 130; its length is then recorded by L over the same metric source.

The path-distance row D is therefore not a quotient of all paths. It is the finite lower-envelope surface generated by the displayed fragment ledger and transported by H and C.

Finally Q and N source and name exactly this route. Since selected geodesics, quotient path spaces, host curve equality, unlisted path families, and ambient completion are absent from the carrier, no NameCert obligation can export them. □

Theorem 6866.3 (PathMetric length-space handoff)**.** *Every* $\mathsf{LengthSpace}^\uparrow$*,* $\mathsf{GeodesicMetric}^\uparrow$*, or* $\mathsf{S}^{1\uparrow}$ *consumer of an accepted* $\mathsf{PathMetric}^\uparrow$ *packet must read the finite route*

$$M, F, L, D \longmapsto B \longmapsto H, C, Q, N$$

before treating the path-distance row as usable data. The handoff supplies the displayed metric source, curve-fragment rows, length budgets, path-distance exposure, boundary handoff, transport, replay, provenance, and local naming; it supplies no selected geodesic, quotient path space, host curve equality, unlisted path family, or completed metric carrier.

Proof. Begin with the consumer boundary. A $\mathsf{LengthSpace}^\uparrow$ read uses the length ledger L and the path-distance row D, a $\mathsf{GeodesicMetric}^\uparrow$ read may compare a segment witness with the same rows, and an $\mathsf{S}^{1\uparrow}$ read may use only the circle-source rows exposed through the boundary handoff B.

The handoff row B is therefore the only downstream exit. It records which endpoint, fragment, length, and circle-source reads are visible before the consumer uses the path-distance claim.

Transport and replay remain componentwise through H and C, while Q and N preserve provenance and local naming. Hence the downstream surface is exactly the displayed route and cannot be replaced by selected geodesics, quotient path spaces, host curve equality, unlisted path families, or completion of the ambient metric carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{PathMetric}^\uparrow}$**.**

Constructive story: A $\mathsf{PathMetric}^\uparrow$ packet is generated from a displayed metric source, a finite family of curve-fragment rows, fragment-length ledgers, a path-distance row, a LengthSpace/GeodesicMetric/SOne boundary handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PathMetric}^\uparrow}$, seed)

Scope closed: Definition 6866.1, Definition 6866.2, and Definition 6866.3 seed the finite path-distance surface between MetricSpace, ContinuousMap, LengthSpace, GeodesicMetric, and SOne consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{PathMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected geodesics, quotient path spaces, host curve equality, unlisted path families, ambient metric completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fragment-family exactness, length-budget stability, path-distance boundary handoff, downstream consumer nonescape, transport stability, replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.PathMetricUp.

Chapter 6867

A Concrete Naming Certificate for $\mathsf{LocatedRange}^{\uparrow}$

Following Bishop and Bridges on located subsets and compact metric images, the $\mathsf{LocatedRange}^{\uparrow}$ packet records the finite data by which the range of a displayed compact-continuous map is read as located range data. It sits after the compact source of chapter 129, the graph and pointwise route of chapter 130, the tolerance handoff of chapter 369, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The carrier names compact-continuous range reads through displayed finite rows; it does not import open-cover compactness, selected image points, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6867.1 (LocatedRange carrier)**.** *A* $\mathsf{LocatedRange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (K_0, F, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. K_0 *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact domain only through displayed finite-net and coverage data.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the finite graph and pointwise-continuity history available to range consumers.*
3. U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *row that binds compact-source requests to one tolerance route before image windows are read.*
4. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *image-window row, listing only the displayed windows inspected by downstream range queries.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same image windows and tolerance requests.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* W *and* R *have both been displayed.*
7. H *records componentwise* hsame *transport for the compact source, graph, modulus, windows, readback, seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a located-range request through* $K_0, F, U, W, R, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{LocatedRange}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRange}^\uparrow}$ *row.*

The classifier compares accepted packets only through $K_0, F, U, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, a choice-selected image point, quotient equality of maps, a countable-choice schedule, host real equality, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 6867.2 (LocatedRange NameCert obligations)**.** *For an accepted carrier $K = (K_0, F, U, W, R, E, H, C, P,$ the local* $\mathsf{NameCert}_{\mathsf{LocatedRange}^\uparrow}$ *surface consists exactly of five rows:*

1. *carrier admission for the finite packet K;*
2. *classifier locality for the* $\mathsf{CompactMetric}^\uparrow$ *source,* $\mathsf{ContinuousMap}^\uparrow$ *graph,* $\mathsf{UniformModulus}^\uparrow$ *tolerance route, finite* $\mathsf{StreamName}^\uparrow$ *windows,* $\mathsf{RegSeqRat}^\uparrow$ *readback, and* $\mathsf{Real}^\uparrow$ *seal;*
3. *stability through componentwise* hsame *transport H and displayed* Cont *replay C;*
4. *exactness through the ordered compact-source, continuous-graph, uniform-modulus, window, readback, and seal route K_0, F, U, W, R, E;*
5. *ledger control through* Pkg *provenance P and the local naming row N.*

No obligation may replace these rows by open-cover compactness, selected image points, quotient function equality, countable choice, host real equality, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 6867.1. The compact source, graph, modulus, window, readback, seal, transport, replay, provenance, and naming rows are exactly $K_0, F, U, W, R, E, H, C, P, N$.

Classifier locality is read through the same finite rows. The compact-source row supplies only displayed compact metric data, the graph row supplies only the continuous-map history, and the modulus row supplies only the tolerance route that is visible before windows are inspected.

Exactness then follows from the ordered route K_0, F, U, W, R, E. The image windows are admitted through W, the regular rational readback is downstream of those windows through R, and the real seal E is reached only after the readback row is displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier row stores open-cover compactness, selected image points, quotient function equality, countable choice, host real equality, or ambient completeness, the five listed rows exhaust the local NameCert surface. □

Theorem 6867.3 (LocatedRange compact-continuous handoff)**.** *Every range read from an accepted* $\mathsf{LocatedRange}^\uparrow$ *carrier factors through the finite compact-continuous route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{ContinuousMap}^\uparrow,\ \mathsf{UniformModulus}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow.$$

In particular, a consumer may reach the range seal only after reading the compact source K_0, the continuous graph F, the uniform modulus U, the finite image windows W, the regular rational readback R, and the terminal real seal E in that order, with H, C, P, N preserving the same route.

Proof. Start with the compact source row K_0 and continuous graph row F. They expose the domain and image graph as displayed NameCert data, not as an open-cover compactness theorem or a host function object.

Read the uniform-modulus row U before any image window is consumed. This keeps every range query behind the compact-continuous tolerance route supplied by the source and graph rows.

The finite image windows W are then the only stream route available to the consumer. The readback row R attaches the same windows to the regular rational surface, and the seal row E is downstream of that readback.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and local naming. Hence no range read can fork into selected image points, quotient function equality, countable-choice schedules, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRange}^{\uparrow}$ range read factors through compact source data, continuous graph data, a uniform-modulus row, finite StreamName image windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, streamname, regseqrat, real: no listed sibling supplies the whole ten-row packet $K = (K_0, F, U, W, R, E, H, C, P, N)$, because each sibling contributes only the compact source, graph, tolerance route, finite windows, regular readback, or terminal seal while this packet binds them into one located range certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRange}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRange}^{\uparrow}$ packet is generated from compact metric source rows, continuous graph rows, uniform-modulus tolerance rows, finite StreamName image windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRange}^{\uparrow}}$, seed)

Scope closed: Definition 6867.1, Definition 6867.2, and Definition 6867.3 seed the finite compact-continuous route from source data to located range reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRange}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected image points, quotient function equality, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, continuous graph exposure, uniform-modulus exactness, finite window coverage, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRangeUp.

Chapter 6868

A Concrete Naming Certificate for $\mathsf{BishopCauchyTailModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ packet records the finite tail-threshold service by which a regular rational Cauchy stream answers a displayed tolerance request before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the name. It sits over the dyadic tolerance ledger of chapter 291, the finite stream window of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the bidirectional Cauchy criterion route of chapter 1913. The certificate names the tail modulus as displayed finite data; it does not select a limit, quotient streams, invoke countable choice, or import ambient completeness.

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ read exposes a dyadic tolerance request, a finite tail threshold, a StreamName observation window, a RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: Nearest siblings include $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{BolzanoCauchy}^{\uparrow}$. None is bijective to $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ because the dyadic packet stores tolerances, the stream packet stores finite observation names, the regular-sequence packet stores rational readback, the real packet stores a seal, and the Bolzano-Cauchy packet stores a bidirectional criterion rather than the finite tail-threshold service for one displayed request.

Definition 6868.1 (Bishop Cauchy tail modulus carrier)**.** *A* $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-request row, carrying the requested dyadic radius, comparison slack, and rational cells used by the tail test.*
2. *M is the tail-threshold row assigning the request in Q a finite index after which the displayed Cauchy tail is accepted.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the observations inspected at and beyond the threshold.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, M, S, R have been replayed.*

6. *H records componentwise* hsame *transport for the tolerance request, tail threshold, stream window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$Q, M, S, R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyTailModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BolzanoCauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyTailModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, S, R, E, H, C, P, N$. It has no coordinate for host Cauchy convergence, selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6868.2 (Bishop Cauchy tail modulus NameCert obligations)**.** *For every accepted* BishopCauchyTailModu *carrier*

$$T = (Q, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyTailModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tolerance request, finite tail threshold, StreamName window, RegSeqRat readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows. Every accepted consumer must read the tail-threshold row M after the tolerance row Q and before the readback route S, R, E.*

Proof. Carrier admission is projection from Definition 6868.1. The accepted packet displays exactly the tolerance request, threshold, stream window, regular readback, seal, transport, replay, provenance, and local naming rows.

The analytic route begins with Q and M. The dyadic row fixes the requested tolerance, and the threshold row is the only coordinate that answers that request with a finite tail index.

The readback route is downstream of the threshold. The window row S names the inspected observations, the regular rational row R reads those observations under the same tolerance, and the seal row E consumes that readback without adding a host limit.

Finally H, C, P, N transport, replay, source, and name only the same displayed route. Since no remaining coordinate stores quotient stream equality, selected limits, countable choice, trichotomy, or ambient completeness, the local NameCert obligations are exactly the listed rows. □

Theorem 6868.3 (Bishop Cauchy tail modulus finite tail exactness)**.** *Let*

$$T = (Q, M, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ *carrier. Every* $\mathsf{BolzanoCauchy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing read that consumes the displayed tail modulus factors through the finite route*

$$Q \longmapsto M \longmapsto S, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives the same dyadic request, tail threshold, finite observation window, regular rational readback, Real seal, transport, replay, provenance, and local naming data, and receives no selected host limit, quotient stream equality, countable-choice schedule, trichotomy, or ambient completeness theorem.

Proof. Start from the accepted carrier and project the rows $Q, M, S, R, E, H, C, P, N$. The only row that can answer the tolerance request is the displayed threshold row M.

Read the downstream observations next. The stream window S and regular readback R are attached to the same request-threshold pair, so a consumer cannot inspect the readback before the finite tail index has been exposed.

The Real-facing seal E is reached only after that readback. Thus a $\mathsf{BolzanoCauchy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or completion-facing consumer sees a finite tail route rather than a selected limit or ambient completeness principle.

The structural rows preserve the route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row carry no extra analytic coordinate, so the listed route is exact for every accepted tail-modulus read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyTailModulus}\uparrow}$.

Constructive story: A $\mathsf{BishopCauchyTailModulus}^{\uparrow}$ packet is generated from a dyadic tolerance request, finite tail threshold, StreamName finite window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyTailModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6868.1, Definition 6868.2, and Definition 6868.3 seed the finite tail-threshold route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyTailModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-to-threshold routing, finite-window exactness, Real seal nonescape, and structural stability for BEDC.Derived.BishopCauchyTailModulusUp.

Chapter 6869

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$ packet records the finite diagonal uniform-modulus surface used when a family of regular Cauchy rational names is read through one displayed diagonal schedule before a Real or Completion consumer receives a seal. It sits beside the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the diagonal source route of chapter 372, and the selector surface of chapter 393. The packet is a Bishop-style finite modulus witness: it does not choose limits, quotient streams, countably many moduli, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6869.1 (Cauchy diagonal uniform modulus carrier)**.** *A* $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, U, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the diagonal schedule row, read through* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *and naming the finite precision-to-window assignments consumed by the diagonal read.*
2. *U is the uniform modulus row, assigning each displayed tolerance request one finite bound that serves all selected regular Cauchy rows visible to the schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget row, recording the dyadic request, slack conversion, radius ledger, and comparison cells used by U.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displaying the regular rational readbacks inspected by the diagonal schedule.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, U, D, R and the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source route have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, uniform modulus, dyadic budget, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion tolerance request through S, U, D, R, E and the structural rows.*

8. *P records* Pkg *provenance over* $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$, $\mathsf{StreamDiagonalSelector}^{\uparrow}$, $\mathsf{RegularCauchyDiagon}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose consumer-visible route is $S, U, D, R, E, H, C, P, N$. It has no coordinate for a countable-choice selector, a family of hidden moduli, quotient stream equality, a selected completed limit, host real equality, or ambient Real completeness.

Theorem 6869.2 (Cauchy diagonal uniform modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$ *packet $M = (S, U, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalUniformModul}}$ *surface consists exactly of carrier admission, diagonal schedule exposure through S, uniform modulus exposure through U, dyadic tolerance-budget visibility through D, RegSeqRat source readback through R, Real seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 6869.1. The packet displays one diagonal schedule, one uniform modulus row, one dyadic tolerance ledger, the regular rational readbacks inspected by that schedule, and the terminal Real seal.

The diagonal schedule is read first. It fixes the finite StreamName windows and selector coordinates supplied by $\mathsf{StreamDiagonalSelector}^{\uparrow}$ and $\mathsf{RegularCauchyDiagonal}^{\uparrow}$. The uniform row U then answers a displayed tolerance request by a single bound over those scheduled reads rather than by choosing a separate hidden modulus for each source.

The dyadic budget D and regular readback row R make the bound auditable. They display the precision, slack, comparison, and rational readback data consumed before the Real seal E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no countable-choice selector, quotient-stream equality, selected completed limit, host equality, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted CauchyDiagonalUniformModulusUp consumer displays the diagonal schedule, uniform modulus, DyadicRatCore tolerance budget, RegSeqRat source readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and NameCert rows before any Real-facing diagonal read is public.

Independence witness: dyadicratcore, streamname, regseqrat, real, regularcauchydiagonal, stream-diagonal-selector: no listed sibling supplies the whole nine-row packet $M = (S, U, D, R, E, H, C, P, N)$, because each sibling contributes only dyadic tolerance, finite window, regular readback, terminal seal, diagonal source, or selector data while this packet binds them under one uniform modulus row.

The future Lean file `lean4/BEDC/Derived/CauchyDiagonalUniformModulusUp.lean` should expose the carrier and NameCert-obligation surface for `BEDC.Derived.CauchyDiagonalUniformModulusUp` without adding quotient streams, selected limits, countable choice, or ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}$ packet is generated from a diagonal schedule, a uniform modulus row, a DyadicRatCore tolerance budget, RegSeqRat source readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyDiagonalUniformModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6869.1 and Definition 6869.2 seed the finite diagonal uniform-modulus route over schedule, tolerance, readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyDiagonalUniformModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, countable choice, quotient stream equality, selected completed limits, host real equality, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exposure, uniform modulus exposure, dyadic tolerance-budget visibility, regular readback stability, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyDiagonalUniformModulusUp.

Chapter 6870

A Concrete Naming Certificate for $\mathsf{DyadicDecimalComparison}^{\uparrow}$

Following Bishop and Bridges on constructive decimal expansions and regular Cauchy real names, the $\mathsf{DyadicDecimalComparison}^{\uparrow}$ packet records the finite comparison surface by which inspected decimal prefixes are read against dyadic tolerances before any regular rational or real seal is consumed. It sits after the finite decimal-prefix carrier of chapter 1417, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite L10 comparison route; it does not choose an infinite decimal stream, quotient a Cauchy sequence, decide trichotomy, select an endpoint convention, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6870.1 (Dyadic decimal comparison carrier)**.** *A* $\mathsf{DyadicDecimalComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *finite-prefix row, displaying the inspected digit block, its place-value ledger, and the decimal window from which the comparison is read.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic comparison radius, slack budget, and finite rational normalization used for the displayed prefix.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the prefix positions and tolerance requests inspected by the comparison.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same decimal prefix and dyadic tolerance data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, Q, W, R have all been displayed.*
6. *H records componentwise* hsame *transport for D, Q, W, R, E and for the replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a comparison request through the decimal prefix, dyadic tolerance, finite window, regular readback, and real seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicDecimalComparison}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDecimalComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, Q, W, R, E, H, C, P, N$. It has no coordinate for an infinite decimal expansion, quotient real equality, endpoint selection, host trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6870.2 (Dyadic decimal comparison NameCert obligations)**.** *For every accepted* DyadicDecimalComparis *carrier $K = (D, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicDecimalComparison}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet K;*
2. *classifier locality for the* $\mathsf{DecimalExpansion}^{\uparrow}$ *prefix row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, finite* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row;*
3. *stability under componentwise* hsame *transport H and displayed* Cont *replay C;*
4. *exactness of the ordered route D, Q, W, R, E from decimal prefix to dyadic tolerance, finite window, regular readback, and real seal;*
5. *ledger control through* Pkg *provenance P and the local naming row N.*

No obligation may replace these rows by an infinite decimal stream, quotient Cauchy equality, endpoint convention selection, host trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6870.1. The analytic rows are D, Q, W, R, E, and the structural rows are H, C, P, N.

The locality obligation reads only displayed finite data. The decimal row D exposes the inspected finite prefix, the dyadic row Q fixes the tolerance and normalization budget, the window row W records which prefix positions are inspected, and the readback and seal rows R, E are downstream of those rows.

The stability obligation is structural. Componentwise transport H preserves each displayed row, and replay C repeats the same route from a comparison request to the terminal seal.

The exactness and ledger obligations are then exhausted by D, Q, W, R, E, P, N. Since the carrier has no row for infinite streams, quotient equality, endpoint selection, trichotomy, countable choice, or ambient completeness, the five listed rows are the whole local NameCert surface. □

Theorem 6870.3 (Dyadic decimal comparison readback boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicDecimalComparison}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, a consumer cannot reach the regular rational readback R or terminal seal E without first reading the decimal prefix row D, dyadic tolerance row Q, finite window row W, and structural rows H, C, P, N that preserve the same route.

Proof. Begin with an accepted packet $K = (D, Q, W, R, E, H, C, P, N)$. The carrier definition orders the decimal prefix, dyadic tolerance, and finite window rows before the regular readback and real seal.

Apply Definition 6870.2. The exactness row requires the route D, Q, W, R, E, while the stability and ledger rows require H, C, P, N to preserve and name that route.

Thus every readback or real-seal consumer sees the displayed finite comparison route before it sees R or E. The packet exports no separate route through an infinite decimal stream, quotient real representative, endpoint selector, host trichotomy, countable-choice schedule, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{DyadicDecimalComparison}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed decimal prefix, dyadic tolerance, finite StreamName window, componentwise transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: decimalexpansion, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (D, Q, W, R, E, H, C, P, N)$, because each sibling contributes only decimal prefixes, dyadic tolerance, finite windows, regular readback, or terminal seal while this packet binds them into one comparison certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDecimalComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDecimalComparison}^{\uparrow}$ packet is generated from a DecimalExpansion finite-prefix row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDecimalComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6870.1, Definition 6870.2, and Definition 6870.3 seed the finite comparison route from decimal prefixes to dyadic tolerance and regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDecimalComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite decimal stream selection, quotient real equality, endpoint convention selection, host trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix exposure, dyadic tolerance exactness, finite window coverage, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicDecimalComparisonUp.

Chapter 6871

A Concrete Naming Certificate for $\mathsf{PringsheimDoubleSequence}^{\uparrow}$

Following Pringsheim and Hardy on double sequences and double series, the $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ packet records rectangular-window convergence as finite NameCert data rather than as an ambient selected double limit. It sits after the finite stream rows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the real seal of chapter 38, the Cauchy sequence-space route of chapter 406, and the diagonal cofinal-tail route of chapter 632. The carrier is the traditional Pringsheim rectangular-window object reconstructed over the Phase Real and completion route; it does not assert quotient streams, selected double limits, host real equality, countable choice, a general iterated-limit interchange theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6871.1 (Pringsheim double-sequence carrier)**.** *A* $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, J, W, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are unary index schedules for the two Pringsheim directions.*
2. *W is the rectangular finite* $\mathsf{StreamName}^{\uparrow}$ *window row, displaying exactly the pairs inspected below a requested rectangle.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the displayed side bounds, dyadic slack, and rectangle-tail budget.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rectangular observations and tolerance requests.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after the schedules, window, tolerance, and regular readback rows have been displayed.*
6. *H is componentwise* hsame *transport for the schedules, windows, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay from index requests to rectangular windows, from windows to dyadic tolerances, and from tolerances to regular readbacks and the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{PringsheimDoubleSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{DiagonalCofinalTail}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PringsheimDoubleSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, J, W, T, Q, E, H, C, P, N$. It contains no coordinate for quotient streams, selected double limits, host real equality, countable choice, general iterated-limit interchange, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PringsheimDoubleSequenceUp.PringsheimDoubleSequenceTasteGate single carrier alignment`

Theorem 6871.2 (Pringsheim double-sequence NameCert obligations)**.** *For every accepted* $\mathsf{PringsheimDoubleSeque}$ *carrier $D = (I, J, W, T, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PringsheimDoubleSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the two unary schedules, rectangular window, dyadic tolerance, regular readback, real seal, transport, replay, provenance, and local naming rows; rectangular-window exactness through I, J, W; tolerance exactness through T; regular readback through Q; real-seal nonescape through E; and structural preservation through H, C, P, N.*

Proof. Project the carrier in Definition 6871.1. The tuple lists all rows available to a certificate consumer, so carrier admission is componentwise over the displayed schedules, rectangular windows, tolerances, readbacks, seal, and structural rows.

The dependency order is exhausted from left to right. A request first exposes I and J, then the finite rectangle W, then the dyadic budget T, and only then the regular rational readback Q and real seal E.

The structural rows preserve that route. Componentwise transport, continuation replay, provenance, and local naming cannot introduce quotient streams, selected double limits, host real equality, choice, iterated-limit interchange, or ambient completeness because no such row is present in D. □

Theorem 6871.3 (Pringsheim double-sequence rectangular-window exactness)**.** *Every accepted rectangular-window read from $D = (I, J, W, T, Q, E, H, C, P, N)$ factors through the two unary schedules I, J and the finite* $\mathsf{StreamName}^{\uparrow}$ *rectangle W before it can consume a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, or a* $\mathsf{Real}^{\uparrow}$ *seal. The read exports the displayed rectangle route and not a quotient stream, selected double limit, host real equality, countable-choice selector, or general iterated-limit interchange theorem.*

Proof. Start with a rectangular-window read. The only index data available are the unary schedules I and J, so the read cannot inspect a pair before both one-dimensional schedules have been displayed.

The window row W then fixes the finite rectangle. The tolerance row T may be read only for that displayed rectangle, and Q repacks the same observations as regular rational data.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no quotient, selector, host-equality, choice, interchange, or completeness coordinate, the rectangular-window route is exact at the finite packet boundary. □

Theorem 6871.4 (Pringsheim double-sequence Real-seal handoff)**.** *For every accepted* $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ *packet $D = (I, J, W, T, Q, E, H, C, P, N)$, any completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer may cite only the finite route*

$$I, J, W \longmapsto T \longmapsto Q \longmapsto E, H, C, P, N.$$

The handoff records a rectangular-window Pringsheim seal for the Phase Real and completion route; it does not supply ambient Real completeness, selected double limits, quotient stream equality, host real equality, countable choice, or a general iterated-limit interchange theorem.

Proof. Apply Definition 6871.2. It identifies the seal as downstream of the schedules, rectangular window, tolerance row, and regular readback.

Apply Definition 6871.3. The completion-facing route remains tied to the same finite rectangle and dyadic budget before the regular handoff and seal are read.

Thus every Real-facing consumer receives the displayed finite route through E, H, C, P, N. The refused analytic and quotient objects have no carrier coordinate and cannot be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ completion-facing read exposes two unary index schedules and a rectangular finite StreamName window before any DyadicRatCore tolerance row, RegSeqRat readback, or RealUp seal is consumed.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchysequencespace, diagonalcofinaltail: $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two unary schedules, rectangular windows, dyadic tolerance rows, regular readbacks, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PringsheimDoubleSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PringsheimDoubleSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6871.1, Definition 6871.2, Definition 6871.3, and Definition 6871.4 seed the finite rectangular-window double-sequence route over schedules, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PringsheimDoubleSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected double limits, host real equality, countable choice, general iterated-limit interchange, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rectangular-window exactness, dyadic tolerance handoff, RegSeqRat readback, Real seal nonescape, and structural transport for BEDC.Derived.PringsheimDoubleSequenceUp.

Constructive story: A $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ packet is generated from two unary index schedules, rectangular StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6872

A Concrete Naming Certificate for GroundCompilerAuditMapClosure$^{\uparrow}$

The GroundCompilerAuditMapClosure$^{\uparrow}$ packet realizes the compiler audit-map methodology of chapter 9407 as finite NameCert data. It packages event-flow rows, source-channel rows, recognizer rows, certificate-gate rows, metric-report rows, non-evidence rows, transport, replay, provenance, and local naming so that a self-compilation consumer can inspect the compiler channel boundary without reading reports, motifs, host packages, or generated code as proof evidence. The downstream consumer is the MetaCIC critical-path packet of chapter 765, which reads those event-flow, source-channel, recognizer, certificate-gate, metric-report, and non-evidence rows before using compiler-channel audit data inside a finite critical-path route.

Definition 6872.1 (GroundCompiler audit-map closure carrier)**.** *A* GroundCompilerAuditMapClosure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, S, R, K, M, O, H, C, P, N)$$

with the following displayed rows:

1. *F is the compiler event-flow row.*
2. *S is the legal source-channel row, including the channel by which a displayed event flow may be read.*
3. *R is the generated recognizer row for packages, NameCert data, derivation certificates, theorem codes, and chapter codes.*
4. *K is the certificate-gate row, exposed only after the legal source channel and recognizer rows are displayed.*
5. *M is the metric-report row, recording counts and adequacy reports as reports rather than certificate evidence.*
6. *O is the non-evidence row, retaining the boundary around motifs, reports, host packages, readable code, and hidden compiler input.*
7. *H records componentwise* hsame *transport for event-flow, source-channel, recognizer, gate, metric, non-evidence, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from event flow to source channel, from source channel to recognizer and certificate-gate checks, and from those rows to metric and non-evidence reports.*

9. *P is the* Pkg *provenance over* $\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, R, K, M, O, H, C, P, N$. It has no coordinate for treating motif similarity, metric counts, generated source text, host packages, or hidden compiler state as proof or certificate evidence.

Theorem 6872.2 (GroundCompiler audit-map closure obligations)**.** *For every accepted* GroundCompilerAuditMapC *packet $G = (F, S, R, K, M, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}}$ *surface consists exactly of event-flow exposure through F, legal source-channel exposure through S, generated recognizer exposure through R, certificate-gate exposure through K, metric-report exposure through M, non-evidence retention through O, and structural preservation through H, C, P, N.*

Proof. Project the carrier in Definition 6872.1. The compiler-facing rows are precisely F, S, R, K, M, O, and the structural rows are H, C, P, N.

Read the rows in audit order. Event flow is displayed first, the legal source channel is displayed before recognizer and certificate-gate export, and metric reports remain downstream reports rather than evidence rows.

The non-evidence row is retained through transport, replay, provenance, and naming. Since the carrier has no coordinate for report-as-proof, code-as-proof, motif-as-identity, host package input, or hidden compiler state, those readings are refused by exhaustion. □

Theorem 6872.3 (GroundCompiler audit-map legal-channel gate)**.** *Every accepted* GroundCompilerAuditMapClosu *export exposes the legal source-channel row S before the certificate-gate row K can be consumed. Recognizers, metric reports, and non-evidence ledgers may be cited only along the displayed route*

$$F, S \longmapsto R, K \longmapsto M, O, H, C, P, N.$$

The export does not promote readable code, motif reports, metric counts, hidden compiler input, or host package structure into proof or certificate evidence.

Proof. Apply Definition 6872.2. It makes S a first-class row of the accepted packet and places certificate-gate exposure among the rows that must remain visible to a consumer.

The replay row C preserves the channel order. A consumer moves from event flow through the legal channel before recognizer and certificate-gate rows are available, and metric or non-evidence reports are read only after that gate.

Thus the legal-channel gate is finite and local to the packet. The refused compiler artifacts have no carrier coordinate from which they could be retyped as evidence. □

Theorem 6872.4 (GroundCompiler audit-map metric non-evidence boundary)**.** *In every accepted* $\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}$ *carrier, metric reports and motif counts remain in M and the explicit non-evidence boundary remains in O through H, C, P, N. A self-compilation consumer may use these rows to audit coverage and reporting, but not as proof evidence, certificate evidence, theorem-code equality, chapter-code equality, or hidden input authorization.*

Proof. Project M and O from the carrier. The metric row records reports, and the non-evidence row records the boundaries around motifs, readable code, host packages, and hidden compiler state.

Apply the legal-channel gate theorem. Since M and O are downstream of the displayed event-flow, source-channel, recognizer, and certificate-gate rows, they cannot replace those rows.

The structural rows transport and replay the same boundary. Therefore report and motif data remain audit metadata rather than proof or certificate evidence. □

Falsifiable prediction: Every accepted $\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}$ packet exposes the legal source-channel row before certificate-gate export.

Independence witness: metacic-confluence-audit-packet, metaciccriticalpath, conceptregistry-surface, failurecertificate: $\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compiler event-flow, source-channel, recognizer, certificate-gate, metric-report, non-evidence, transport, replay, provenance, and local naming rows.

The Lean TasteGate route for this packet must provide Nontrivial and FieldFaithful instances by separating accepted compiler-channel audit packets from refused report-as-proof and hidden-input packets.

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6872.1, Definition 6872.2, Definition 6872.3, and Definition 6872.4 seed the finite compiler-channel audit-map closure over event-flow, source-channel, recognizer, certificate-gate, metric-report, non-evidence, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close proof evidence from reports, certificate evidence from motifs, theorem equality from code, chapter equality from topics, hidden compiler input authorization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal source-channel exposure, recognizer generatedness, certificate-gate ordering, metric-report non-evidence, non-evidence retention, TasteGate nontriviality, FieldFaithful coverage, and structural transport for BEDC.Derived.GroundCompilerAuditMapClosureUp.

Constructive story: A $\mathsf{GroundCompilerAuditMapClosure}^{\uparrow}$ packet is generated from compiler event-flow rows, legal source-channel rows, recognizer rows, certificate-gate rows, metric-report rows, non-evidence rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6873

A Concrete Naming Certificate for $\mathsf{CauchyFilterDiagonal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ packet records the finite diagonal read by which a displayed Cauchy-filter completion route is handed to regular rational names and then to the terminal Real seal. It sits over the Cauchy-filter source of chapter 333, the completion-facing row of chapter 1569, the metric-completion boundary of chapter 1506, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the completed real-name seal of chapter 38. The chapter names a finite diagonal route only; it does not quotient filters, choose representatives, select completed limits, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6873.1 (Cauchy-filter diagonal carrier)**.** *A* $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, K, M, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose base requests and finite Cauchy evidence are available before any completion consumer reads the packet.*
2. *K is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row, recording the completion-facing use of the displayed filter source.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary row, visible only after F and K have exposed the diagonal request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that supplies the diagonal observation schedule.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the stream windows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback.*
7. *H records componentwise* hsame *transport for $F, K, M, S, Q, R, C, P, N$.*
8. *C records displayed* Cont *replay from a finite Cauchy-filter request through the completion handoff, the metric-completion boundary, the diagonal stream window, the regular rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, K, M, S, Q, R, H, C, P, N$. The carrier has no coordinate for quotient filters, selected representatives, chosen limits, countable-choice schedules, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6873.2 (Cauchy-filter diagonal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ *carrier $D = (F, K, M, S, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterDiagonal}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Cauchy-filter source, Cauchy-filter completion handoff, metric-completion boundary, stream-window schedule, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *diagonal source routing, requiring every completion-facing request to read the same F row before the rows K and M are available;*
3. *finite-window diagonalization, requiring the stream row S to supply the observed diagonal windows before any regular rational row Q or Real seal row R is consumed;*
4. *classifier stability, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the route F, K, M, S, Q, R;*
5. *ledger non-escape, requiring P and N to export no quotient filter, selected representative, chosen limit, countable-choice schedule, quotient stream equality, host real equality, or ambient Real completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6873.1. The accepted packet exposes exactly the rows $F, K, M, S, Q, R, H, C, P, N$, so every local obligation must be read from those rows.

The diagonal source route is forced by the carrier order. A completion-facing request cannot reach K or M without first reading the displayed Cauchy-filter source F.

The finite-window part then uses the diagonal observation schedule. The stream row S supplies the inspected windows, Q reads them as regular rational data, and R seals only the resulting Real-facing name.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, while Cont replay in C repeats the same route F, K, M, S, Q, R for each accepted request.

Finally, the provenance rows are exhausted by P and N. Since quotient filters, selected representatives, chosen limits, countable choice, quotient stream equality, host real equality, and ambient Real completeness are not carrier rows, the local NameCert surface cannot export them. □

Theorem 6873.3 (Cauchy-filter diagonal Real-completion handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ *carrier reads the completion route only through the displayed rows*

$$\mathsf{CauchyFilter}^{\uparrow}, \mathsf{CauchyFilterCompletion}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pk}$$

Consequently the handoff exports no quotient-filter equality, selected representative, selected limit, countable choice, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real-facing completion consumer. Carrier admission gives access only to the rows $F, K, M, S, Q, R, H, C, P, N$ of Definition 6873.1.

The completion side is read first through F, K, M. The filter source F exposes finite Cauchy evidence, K records the Cauchy-filter completion handoff, and M records the metric-completion boundary available to the consumer.

The Real-facing side is then read through S, Q, R. The stream windows supply the diagonal observations, the regular rational row reads those windows, and the Real row seals the resulting name.

The rows H, C, P, N add transport, replay, provenance, and local naming only. They do not add a quotient, selector, choice, stream-equality, host-equality, or ambient-completeness coordinate, so the handoff remains inside the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterDiagonal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6873.1, Definition 6873.2, and Definition 6873.3 seed the finite diagonal route from Cauchy-filter completion data to regular rational readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected representatives, selected limits, countable choice, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, diagonal source routing, finite-window diagonalization, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterDiagonalUp.

Constructive story: A $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ packet is generated from one CauchyFilter row, one CauchyFilterCompletion handoff row, one MetricCompletion boundary row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6874

A Concrete Naming Certificate for $\mathsf{RealSeparationViaUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{RealSeparationViaUniformity}^{\uparrow}$ packet records the finite route by which Real-induced uniform entourage data separates a Cauchy-filter read before a regular rational or Real seal is consumed. It concretizes the compact-uniform-continuity context of chapter 9237 through the Real uniform-structure packet of chapter 1624, the Cauchy-filter source of chapter 333, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The chapter names a separated finite route only; it does not assert Hausdorff completion, quotient filters, selected limits, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6874.1 (Real separation via uniformity carrier)**.** *A* $\mathsf{RealSeparationViaUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, F, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *row whose Real-induced entourages supply the separation test.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row whose finite filter requests are read only through the displayed entourage surface.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected after the separated filter request is displayed.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback reached from those windows.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and seal row consumed after the regular rational readback.*
6. *H records componentwise* hsame *transport for U, F, S, Q, R, C, P, N.*
7. *C records displayed* Cont *replay from a Real uniform entourage request through Cauchy-filter separation, finite stream windows, regular rational readback, and the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeparationViaUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, S, Q, R, H, C, P, N$. The carrier has no coordinate for quotient filters, separated quotient completion, selected limits, trichotomy, open-cover compactness, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6874.2 (Real separation via uniformity NameCert obligations)**.** *For every accepted* $\mathsf{RealSeparationViaUniformity}^{\uparrow}$ *carrier* $E = (U, F, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSeparationViaUniformity}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Real-uniform entourage row, Cauchy-filter row, stream-window row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *uniform separation, requiring every separated filter request to read the* $\mathsf{RealUniformStructure}^{\uparrow}$ *row* U *before the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *is available;*
3. *finite Real readback, requiring* S *and* Q *to mediate every Real-facing read before* R *is consumed;*
4. *classifier stability, requiring componentwise* hsame *transport* H *and displayed* Cont *replay* C *to preserve the route* U, F, S, Q, R*;*
5. *ledger non-escape, requiring* P *and* N *to export no quotient filter, separated quotient completion, selected limit, trichotomy, open-cover compactness, countable choice, host real equality, or ambient Real completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6874.1. The accepted object exposes only $U, F, S, Q, R, H, C, P, N$.

Uniform separation follows from the displayed source order. The filter row F is available only after the Real-induced entourage row U has been read, so a consumer cannot separate Cauchy-filter evidence by an external quotient or host equality.

Finite Real readback is mediated by the stream and regular rational rows. The window row S is inspected first, Q reads the regular rational data, and R seals only the resulting Real-facing name.

Transport and replay preserve this same route. The row H supplies componentwise hsame stability, and C repeats the route U, F, S, Q, R under displayed continuation.

Finally, provenance is exhausted by P and N. Since the excluded quotient, selection, trichotomy, compactness, choice, host-equality, and ambient-completeness data are not rows of the carrier, the local NameCert surface cannot export them. □

Theorem 6874.3 (Real separation Cauchy-filter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealSeparationViaUniformity}^{\uparrow}$ *carrier reads separated Cauchy-filter evidence only through the displayed rows*

$$\mathsf{RealUniformStructure}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Consequently the handoff exports no quotient filters, separated quotient completion, selected limits, trichotomy, open-cover compactness, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing consumer. By Definition 6874.1, the consumer can see the packet only through $U, F, S, Q, R, H, C, P, N$.

The separation route starts at U. The Real uniform structure supplies the displayed entourage surface, and the Cauchy-filter row F is read downstream of that surface.

The Real-facing handoff then passes through S, Q, R. Stream windows determine the finite observations, regular rational readback records them, and the Real row seals the result.

The structural rows H, C, P, N transport, replay, source, and name the same route. None of them adds a quotient filter, separated quotient completion, selected limit, trichotomy, compactness, choice, host equality, or ambient Real-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeparationViaUniformity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSeparationViaUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6874.1, Definition 6874.2, and Definition 6874.3 seed the finite route from Real-induced uniform separation to Cauchy-filter readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSeparationViaUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, separated quotient completion, selected limits, trichotomy, open-cover compactness, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, uniform separation, finite Real readback, transport, replay, provenance, and local naming for BEDC.Derived.RealSeparationViaUniformityUp.

Constructive story: A $\mathsf{RealSeparationViaUniformity}^{\uparrow}$ packet is generated from one RealUniformStructure row, one CauchyFilter row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6875

A Concrete Naming Certificate for RegularCauchyFilterComparison$^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the RegularCauchyFilterComparison$^{\uparrow}$ packet records the finite comparison surface between filter-style Cauchy data and regular-sequence completion data. It sits between the Cauchy-filter source of chapter 333, the Bishop regular Cauchy completion route of chapter 1639, the dyadic endpoint ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet compares displayed finite rows only; it does not quotient filters, select a limit, use countable choice, assert host equality of Real names, or invoke ambient Real$^{\uparrow}$ completeness.

Definition 6875.1 (Regular Cauchy filter comparison carrier)**.** *A* RegularCauchyFilterComparison$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, B, T, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyFilter$^{\uparrow}$ *base row, naming the finite basis requests and displayed refinements offered to the completion consumer.*
2. *B is the cofinal-base row, recording which displayed filter base refines the regular common-tail window.*
3. *T is the* RegSeqRat$^{\uparrow}$ *tail-window row exposed by the refined base.*
4. *S is the* StreamName$^{\uparrow}$ *finite-observation row read on that tail window.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row shared by the filter request and the regular-sequence readback.*
6. *R is the terminal* Real$^{\uparrow}$ *seal handoff, available only after F, B, T, S, D have been displayed.*
7. *H records componentwise* hsame *transport for the filter, refinement, tail-window, observation, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a filter-base tolerance request through the refined regular tail route and into the terminal Real seal.*
9. *P is the* Pkg *provenance over* RegularCauchyFilterComparison$^{\uparrow}$, CauchyFilter$^{\uparrow}$, BishopRegularCauchyCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterComparison}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, B, T, S, D, R, H, C, P, N$. No coordinate stores quotient-filter equality, selected limits, countable-choice schedules, host equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyFilterComparisonUp.RegularCauchyFilterComparisonUp`
Lean4 (definition) `BEDC.Derived.RegularCauchyFilterComparisonUp.RegularCauchyFilterComparisonCarrier`
Lean4 ✓ `BEDC.Derived.RegularCauchyFilterComparisonUp.RegularCauchyFilterComparisonTasteGate`
`single carrier alignment`

Theorem 6875.2 (Regular Cauchy filter sequential handoff)**.** *Every accepted* $\mathsf{RegularCauchyFilterComparison}^{\uparrow}$ *packet that receives a* $\mathsf{CauchyFilter}^{\uparrow}$ *basis whose displayed base refines the packet's common-tail window hands the completion read to a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail window with the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite observations,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The terminal* $\mathsf{Real}^{\uparrow}$ *seal is read only after that regular tail route has been displayed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterComparisonUp.RegularCauchyFilterComparisonSequentialHandoff`

Proof. Project the carrier of Definition 6875.1. The incoming filter row is F, and the cofinal-base row B is the only place where a filter refinement may be read.

Use B to expose the finite regular tail window T. The stream observations S and dyadic tolerance ledger D are then read on the same displayed window, so the comparison has a sequence-facing route before any Real seal is consumed.

The structural rows preserve that route. Componentwise hsame transport in H and displayed Cont replay in C carry the same finite rows through provenance P and local naming N.

Finally read the terminal $\mathsf{Real}^{\uparrow}$ seal R. Since the carrier has no quotient-filter, selected-limit, choice, host-equality, or Real-completeness coordinate, the handoff is exactly the displayed finite sequential route. □

Theorem 6875.3 (Regular Cauchy filter comparison NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterComparison}^{\uparrow}$ *carrier $K = (F, B, T, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompari}}$ *surface consists exactly of:*

1. *carrier admission for the CauchyFilter base, cofinal-base refinement, RegSeqRat tail window, StreamName observations, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *sequential handoff, requiring the refined filter base to expose T, S, D before R is available;*
3. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve the route F, B, T, S, D, R;*
4. *ledger non-escape, requiring P and N to export no quotient filters, selected limits, countable choice, host Real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is the projection of the tuple $K = (F, B, T, S, D, R, H, C, P, N)$. The listed rows exhaust the fields visible to a completion-facing consumer.

Sequential handoff is supplied by Definition 6875.2. The filter request is first refined by B, then read as the regular tail window T with observations S and tolerance ledger D before the terminal Real seal R is exposed.

Classifier stability is rowwise. The hsame row transports displayed coordinates, and the Cont row replays the same finite route without changing the classifier surface.

The ledger boundary follows from absence. Since no excluded object is a coordinate of the carrier, provenance and local naming cannot export quotient filters, selected limits, countable choice, host Real equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6875.1, Definition 6875.2, and Definition 6875.3 seed the finite comparison surface between CauchyFilter bases and RegSeqRat tail windows over DyadicRatCore, StreamName, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, sequential handoff, classifier stability, provenance, and local naming for BEDC.Derived.RegularCauchyFilterComparisonUp.

Constructive story: A comparison packet is generated from one CauchyFilter base row, one cofinal-base refinement row, one RegSeqRat tail window, StreamName observations, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6876

A Concrete Naming Certificate for $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$

Following Barendregt and Coquand on typed lambda calculi and structural closure arguments, the $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$ packet records the finite NameCert route behind the MetaCIC theorem `betaStarStep_preserves_closed`. It concretizes the closedness-preservation audit map of chapter 9381 beside the witness packet of chapter 3344 and the subject-reduction obstruction of chapter 731. The packet is a closedness-only object: it does not assert subject reduction, confluence, normalization, conversion preservation, or a typing transport theorem.

Definition 6876.1 (Finite beta closedness iteration carrier)**.** *A* $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (Z, B, O, S, L, H, C, P, N)$$

with the following displayed rows:

1. *Z is the zero-step row, recording that a term already closed at the displayed boundary remains closed without taking a beta step.*
2. *B is the one-step beta closedness row, read from the* `betaStep_preserves_closed` *route.*
3. *O is the finite order row listing the displayed beta steps of the accepted* `BetaStarStep` *chain.*
4. *S is the successor-step row, recording that B may be replayed at the head of the finite chain before the remaining tail is inspected.*
5. *L is the terminal* `ClosedAt` *ledger for the endpoint term.*
6. *H records componentwise* hsame *transport for the chain, closedness boundary, endpoint ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite induction from Z through repeated uses of B and S to L.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$, `BetaStep`, `BetaStarStep`, `ClosedAt`, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteBetaClosednessIteration}^{\uparrow}}$ *row.*

The carrier has no coordinate for a typing derivation, dependent-codomain transport, subject-reduction discharge, confluence diagram, normalization proof, or conversion rule.

Theorem 6876.2 (Finite beta closedness route)**.** *Every accepted* $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$ *packet routes a finite* `BetaStarStep` *chain through the zero-step row, repeated one-step beta closedness rows, the displayed finite order of the chain, and the terminal* `ClosedAt` *ledger. Thus a closed source term remains inside the displayed closedness classifier along the finite beta chain, without adding a subject-reduction, confluence, normalization, or conversion claim.*

Proof. Project the carrier of Definition 6876.1. A zero-length beta chain reads Z and immediately reaches the terminal closedness ledger L.

For a nonzero finite chain, read the displayed head step from O and apply the one-step row B. The successor row S records that the remaining finite tail is another accepted closedness-iteration request.

Iterate this finite route until the endpoint ledger L is reached. The replay row C records the induction as displayed continuation data, and H transports only the closedness and chain rows.

The route never reads a typing derivation or codomain transport. The subject-reduction obstruction remains outside the carrier, so the theorem is a closedness-preservation statement only. □

Theorem 6876.3 (Finite beta closedness iteration NameCert obligations)**.** *For every accepted* $\mathsf{FiniteBetaClosednessIteration}^{\uparrow}$ *carrier* $I = (Z, B, O, S, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteBetaClosednessIteration}^{\uparrow}}$ *surface consists exactly of carrier admission, zero-step closedness, one-step beta closedness, finite-chain ordering, successor replay, endpoint* `ClosedAt` *ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored for subject reduction, dependent-codomain transport, confluence, normalization, conversion preservation, or typing stability.*

Proof. Carrier admission is tuple projection. The only visible rows are $Z, B, O, S, L, H, C, P, N$, so the NameCert surface starts and ends with those displayed fields.

The zero-step and successor cases are the two finite-chain constructors. The zero case reads Z; the successor case reads B for the head beta step, then uses S and C to replay the same closedness route over the remaining finite tail.

The endpoint ledger L records the final `ClosedAt` claim. Transport by H is componentwise and cannot change the classifier from closedness to typing.

Finally, provenance and local naming do not enlarge the carrier. Since no subject-reduction, codomain-transport, confluence, normalization, conversion, or typing-stability coordinate exists, none can be exported as a local NameCert obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBetaClosednessIteration}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBetaClosednessIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6876.1, Definition 6876.2, and Definition 6876.3 seed the finite ClosedAt-preservation route for BetaStarStep chains.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBetaClosednessIteration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, dependent-codomain transport, confluence, normalization, conversion preservation, typing stability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked zero-step, one-step beta, finite-chain, successor replay, endpoint ClosedAt, transport, provenance, and local naming rows for BEDC.Derived.FiniteBetaClosednessIterationUp.

Constructive story: A finite beta closedness packet is generated from zero-step closedness, one-step beta closedness, a finite BetaStarStep order row, successor replay, a terminal ClosedAt ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6877

A Concrete Naming Certificate for $\mathsf{Wronskian}^{\uparrow}$

Following Sturm–Liouville theory and Bishop–Bridges constructive analysis, the $\mathsf{Wronskian}^{\uparrow}$ packet records the finite determinant-and-derivative surface used to inspect linear dependence before an ODE or real-analysis consumer may read a terminal real seal. It sits after the derivative rows of chapter 122, the continuous-map family surface of chapter 130, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the terminal real-name seal of chapter 38. The packet names a constructive Wronskian certificate as displayed BEDC rows; it does not import host function equality, classical limits, selected differentiability witnesses, quotient real names, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: A consumer that reads a Wronskian determinant or linear-dependence verdict without the derivative, continuous-map, StreamName, RegSeqRat, DyadicRatCore, and Real-seal rows has left the $\mathsf{Wronskian}^{\uparrow}$ packet.

Independence witness: derivative, continuousmap, streamname, regseqrat, dyadicratcore, real: $\mathsf{Wronskian}^{\uparrow}$ is not bijective to any one sibling because it jointly carries derivative jets, determinant rows, rational readback, terminal real sealing, transport, replay, provenance, and local naming.

Definition 6877.1 (Wronskian carrier)**.** *A* $\mathsf{Wronskian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, D, J, \Omega, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *finite family row for the inspected functions;*
2. *D is the* $\mathsf{Derivative}^{\uparrow}$ *derivative-admission row for the displayed derivative depths;*
3. *J is the finite jet table whose entries are read only through the same derivative and continuous-map rows;*
4. *Ω is the alternating determinant ledger over the displayed jet table;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-request section selecting the inspected windows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the determinant values on those windows;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed Wronskian value rows;*

8. *H records componentwise* hsame *transport of the family, derivative, jet, determinant, window, readback, and seal rows;*
9. *C records displayed* Cont *replay from function-family requests through derivative jets, determinant assembly, StreamName windows, RegSeqRat readback, and the Real seal;*
10. *P is the* Pkg *provenance over* $\mathsf{Wronskian}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{Wronskian}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function equality, a classical limit, a selected differentiability witness, quotient real equality, countable choice, or ambient real completeness.

Theorem 6877.2 (Wronskian NameCert obligations)**.** *For every accepted* $\mathsf{Wronskian}^{\uparrow}$ *carrier*

$$W = (F, D, J, \Omega, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Wronskian}^{\uparrow}}$ *surface consists exactly of continuous-family admission through F, derivative-row admission through D, finite jet exposure through J, determinant-ledger exposure through Ω, finite-window exposure through S, regular rational readback through R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianNameCertObligations`

Proof. Project the carrier of Definition 6877.1. The family row, derivative row, jet table, determinant ledger, window row, readback row, and Real seal row are visible before the structural rows are read.

Read the analytic route in order. The row F supplies the continuous-map family surface, D supplies derivative admission, J records only the displayed finite derivative jets, and Ω assembles those jets into the Wronskian determinant ledger.

The value-facing route then passes through S, R, E. The $\mathsf{StreamName}^{\uparrow}$ row fixes the inspected finite requests, $\mathsf{RegSeqRat}^{\uparrow}$ reads the determinant values as regular rational data, and $\mathsf{Real}^{\uparrow}$ seals only that displayed readback.

The rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for the excluded host principles, the listed rows exhaust the local NameCert surface. □

Theorem 6877.3 (Wronskian determinant-ledger exactness)**.** *Every determinant or linear-dependence read of an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet must expose the derivative row D, continuous-map family row F, finite jet table J, determinant ledger Ω,* $\mathsf{StreamName}^{\uparrow}$ *window row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route, and terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The read is exact only as the displayed dependency route through chapter 122, chapter 130, chapter 126, chapter 301, chapter 291, and chapter 38; it exports no host function equality, selected differentiability witness, or quotient Real equality.*

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier determinant ledger exactness`

Proof. Project the carrier of Definition 6877.1. The determinant-facing rows are F, D, J, Ω, so determinant assembly can only inspect the displayed continuous-map family, derivative admission, jet table, and alternating ledger.

Route value reads through S, R, E. The finite window row fixes the requests, the regular rational row reads determinant values, and the Real seal exposes only those displayed values with the dyadic tolerance route named by the package row.

The structural rows H, C, P, N preserve this inventory under transport, replay, provenance, and local naming. Since none of these rows contains host function equality, a selected differentiability witness, or quotient Real equality, determinant-ledger exactness is confined to the displayed sibling route. □

Theorem 6877.4 (Wronskian regular Cauchy determinant handoff)**.** *Every accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet whose rows read derivative data,* $\mathsf{ContinuousMap}^{\uparrow}$ *moduli,* $\mathsf{Real}^{\uparrow}$ *seals,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readbacks,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows exports a determinant handoff only through those displayed rows. In carrier notation, the determinant handoff is available only after the route*

$$F, D, J, \Omega, S, R, E, H, C, P, N$$

has exposed the continuous-map family row, derivative row, finite jet table, determinant ledger, stream-window row, regular readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. $\mathsf{SturmSeparation}^{\uparrow}$ *and* $\mathsf{SturmRootCount}^{\uparrow}$ *consumers may cite this handoff, but they receive no hidden quotient Real equality, host determinant equality, selected differentiability witness, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier regular cauchy determinant handoff`

Proof. Project the carrier of Definition 6877.1. The derivative and continuous-map data are displayed as D and F, while the determinant ledger is assembled only from the finite jet table J and the alternating row Ω.

Route the value-facing read through S, R, E. The $\mathsf{StreamName}^{\uparrow}$ row fixes the finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row reads the determinant values as regular rational data, and the $\mathsf{Real}^{\uparrow}$ seal exposes only those displayed readbacks with the dyadic tolerance route named by P.

Apply componentwise transport and replay through H and C. Since the determinant ledger has already been tied to F, D, J, Ω, S, R, E, stability under hsame and Cont replay can only move the same displayed rows.

Finally pass through provenance and local naming. The Sturm consumers cited in the statement are downstream readers of this handoff, not additional coordinates of the $\mathsf{Wronskian}^{\uparrow}$ carrier, so they cannot import hidden quotient Real equality, host determinant equality, selected differentiability witnesses, or ambient completeness. □

Theorem 6877.5 (Wronskian RegSeqRat derivative-window dependency)**.** *Every accepted* $\mathsf{Wronskian}^{\uparrow}$ *derivative-read packet over* $\mathsf{Real}^{\uparrow}$ *consumes derivative rows only through the finite route*

$$F, D, J, \Omega, S, R, E, H, C, P, N.$$

The derivative rows are exposed as $\mathsf{StreamName}^{\uparrow}$ *windows from chapter 126,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback from chapter 301,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets from chapter 291, and* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows from chapter 130 before the* $\mathsf{SturmRootCount}^{\uparrow}$ *and* $\mathsf{SturmSeparation}^{\uparrow}$ *consumers of chapter 3612 and chapter 8378 may read a determinant or separation verdict. The packet exports no host derivative equality, selected differentiability witness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier regseqrat window dependency`

Proof. Project the carrier of Definition 6877.1. The continuous-map row F, derivative row D, finite jet table J, and determinant ledger Ω expose the derivative-facing data before any value row is read.

Route each derivative value through the window and readback rows. The $\mathsf{StreamName}^{\uparrow}$ row S fixes the finite inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies regular rational readback, and the dyadic tolerance route named by P is the only tolerance budget visible to the determinant ledger.

Apply the regular Cauchy determinant handoff of Definition 6877.4. It keeps the determinant route attached to the displayed continuous-map, derivative, stream-window, regular-readback, dyadic-tolerance, and Real-seal rows.

Finally use the Sturm separation handoff of Definition 6877.15. The Sturm consumers receive the same finite route as downstream input, so they cannot replace it by host derivative equality, selected differentiability data, quotient real equality, or ambient completeness. □

Theorem 6877.6 (Wronskian determinant-route stability)**.** *Let*

$$W = (F, D, J, \Omega, S, R, E, H, C, P, N)$$

and

$$W' = (F', D', J', \Omega', S', R', E', H', C', P', N')$$

be accepted $\mathsf{Wronskian}^{\uparrow}$ *packets. Suppose the displayed componentwise* hsame *transport rows identify the continuous-family rows* F, F'*, derivative rows* D, D'*, finite jet tables* J, J'*, determinant ledgers* Ω, Ω'*,* $\mathsf{StreamName}^{\uparrow}$ *window rows* S, S'*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R, R'*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance routes named by* P, P'*, and terminal* $\mathsf{Real}^{\uparrow}$ *seals* E, E'*. Then every determinant or linear-dependence read of* W *and* W' *follows the same finite route through those rows. In particular, transport cannot replace the determinant route by host function equality, selected differentiability witnesses, quotient real equality, or ambient completeness.*

Proof. Project both carriers using Definition 6877.1. The hypotheses identify exactly the rows that the carrier exposes to determinant-facing and value-facing consumers: the continuous-family row, derivative row, jet table, determinant ledger, finite window row, regular readback row, dyadic tolerance route, and terminal Real seal.

Apply Definition 6877.3. Any determinant read must already pass through F, D, J, Ω, S, R, E and the named dyadic tolerance route. Since the corresponding primed rows are related componentwise by the displayed hsame transport, the read has no second route on which to switch representatives.

Use Definition 6877.7 for the linear-dependence consumer boundary. Linear-dependence reads are downstream of the same determinant ledger and value route, so the componentwise transport carries them through the same finite rows.

Finally apply Definition 6877.9. The terminal Real seal is reached only after the displayed route has been read. Host function equality, selected differentiability witnesses, quotient real equality, and ambient completeness are absent from both carriers, so transport cannot introduce them. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier determinant route stability`

Theorem 6877.7 (Wronskian linear-dependence boundary)**.** *Every linear-dependence or Sturm–Liouville-facing consumer of an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet must pass through*

$$F, D, J, \Omega, S, R, E, H, C, P, N.$$

The consumer receives derivative jets, determinant rows, finite request windows, regular rational readback, dyadic tolerance arithmetic, and the terminal $\mathsf{Real}^{\uparrow}$ *seal. It receives no host function equality, selected basis relation, classical zero theorem, quotient real equality, countable-choice row, or ambient completeness principle.*

Proof. Start with Definition 6877.2. It identifies the displayed carrier rows as the whole local NameCert surface and gives no second source for derivative, determinant, readback, or seal data.

The derivative and determinant rows supply the only linear-dependence-facing evidence. A consumer may inspect J and Ω, but those rows are already downstream of the continuous-map and derivative admissions F, D.

The value route is fixed by S, R, E. The finite window is read by $\mathsf{StreamName}^{\uparrow}$, the determinant value is read by $\mathsf{RegSeqRat}^{\uparrow}$ with $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance arithmetic, and the terminal endpoint is the displayed $\mathsf{Real}^{\uparrow}$ seal.

Therefore a linear-dependence consumer cannot replace the packet by host function equality, a selected basis relation, a classical zero theorem, quotient real equality, countable choice, or ambient completeness. Those objects are absent from the carrier and from the local classifier. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier linear dependence boundary`

Theorem 6877.8 (Wronskian linear-dependence witness route)**.** *A linear-dependence witness read for an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet is admitted only when the finite jet table* J*, determinant ledger* Ω*,* $\mathsf{StreamName}^{\uparrow}$ *window row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route named by* P*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *are all displayed. The witness is therefore a finite Wronskian readback route through the packet rows, not a host zero theorem, selected basis relation, quotient real equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 6877.1. A witness-facing read can inspect only the displayed family, derivative, jet, determinant, window, readback, seal, transport, replay, provenance, and local naming rows.

Invoke determinant-ledger exactness and determinant-route stability, Definition 6877.3 and Definition 6877.6. Exactness forces the read through J, Ω, S, R, E and the named dyadic tolerance route, while stability keeps that route fixed under componentwise transport.

The linear-dependence boundary then applies through Definition 6877.7. Since host zero theorems and selected basis relations are not carrier coordinates and are not supplied by the sibling rows, the admitted witness is only the displayed finite readback route. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier linear dependence witness`

Theorem 6877.9 (Wronskian Real-seal nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$F, D, J, \Omega, S, R, E, H, C, P, N$$

and through the sibling rows $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The packet exports no quotient stream, host real equality, selected limit, selected differentiability witness, countable-choice row, ambient completeness theorem, or bridge row.*

Proof. Project the Real-facing rows from Definition 6877.1. The terminal seal row E is downstream of the family, derivative, jet, determinant, finite-window, and regular-readback rows.

Apply Definition 6877.2. It states that these rows, together with H, C, P, N, are the complete local NameCert surface for the accepted packet.

Use Definition 6877.7 for the consumer boundary. Any downstream read that wants a Wronskian value or a linear-dependence verdict must pass through the same finite route before it can inspect the Real seal.

The excluded objects are not carrier coordinates and are not supplied by the sibling rows. Hence the terminal $\mathsf{Real}^{\uparrow}$ seal cannot be reached through quotient streams, host equality, selected limits, selected differentiability witnesses, countable choice, ambient completeness, or bridge data. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier real seal nonescape`

Theorem 6877.10 (Wronskian carrier obligation)**.** *Every accepted* $\mathsf{Wronskian}^{\uparrow}$ *carrier exposes the obligation rows*

$$F, D, J, \Omega, S, R, E, H, C, P, N$$

before any determinant, linear-dependence, ODE, or Real-facing consumer reads the packet. The rows must include the continuous-map family, derivative admissions, finite jet table, determinant ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance route named by provenance, terminal Real seal, componentwise transport, continuation replay, package provenance, and local NameCert naming.

Proof. Project the carrier of Definition 6877.1. Its analytic rows are F, D, J, Ω, S, R, E, and its structural rows are H, C, P, N.

Use the NameCert obligation theorem. It identifies continuous-family admission, derivative-row admission, finite jet exposure, determinant-ledger exposure, finite-window exposure, regular rational readback, Real sealing, transport, replay, provenance, and local naming as the complete local surface.

The determinant-ledger and Real-seal nonescape theorems then force every downstream consumer through the same row sequence. Thus the carrier obligation is row projection and route preservation, not determinant algebra over an ambient host field. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier obligation`

Theorem 6877.11 (Wronskian classifier stability)**.** *The* $\mathsf{Wronskian}^{\uparrow}$ *classifier is stable under componentwise* hsame *transport of the continuous-map family rows, derivative rows, finite jet tables, determinant ledgers, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance routes, Real seals, continuation rows, provenance rows, and local NameCert rows. Such transport preserves the finite Wronskian route and cannot substitute host function equality, selected differentiability witnesses, quotient real equality, countable choice, or ambient completeness.*

Proof. Start with determinant-route stability, Definition 6877.6. It already transports determinant and linear-dependence reads across componentwise identified rows without creating a second representative route.

Apply the carrier obligation theorem. Since each accepted read begins with the same displayed row inventory, classifier comparison may only inspect those rows and their componentwise transports.

Finally use Real-seal nonescape. The terminal seal is downstream of the displayed route through $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the classifier cannot switch to any excluded host or quotient principle. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier classifier stability`

Theorem 6877.12 (Wronskian ledger exactness)**.** *The Wronskian determinant ledger Ω is exact only as a displayed route through the derivative row D, continuous-map family row F, finite jet table J, StreamName window row S, RegSeqRat readback row R, DyadicRatCore tolerance route named by P, and Real seal E. Removing any one of these rows changes the ledger surface and prevents a consumer from reading the same* $\mathsf{Wronskian}^{\uparrow}$ *packet.*

Proof. Use determinant-ledger exactness. It requires determinant and linear-dependence reads to expose the derivative, continuous-map, finite-jet, determinant, StreamName, RegSeqRat, DyadicRatCore, and Real-seal route.

Apply carrier obligation and classifier stability. They keep the ledger tied to $F, D, J, \Omega, S, R, E, H, C, P, N$ under transport and route replay, so no host determinant equality or selected differentiability witness can replace a missing row.

If a listed row is removed, the continuation row C and provenance row P no longer replay the same dependency chain. The packet therefore fails to expose the exact Wronskian ledger claimed by the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier ledger exactness`

Theorem 6877.13 (Wronskian obligation closure scope)**.** *The carrier, NameCert obligations, determinant ledger, determinant-route stability, linear-dependence boundary, witness route, and Real-seal nonescape theorems form the visible* $\mathsf{Wronskian}^{\uparrow}$ *obligation surface over the* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Every accepted read of*

$$W = (F, D, J, \Omega, S, R, E, H, C, P, N)$$

therefore remains inside the finite route through continuous-map admission, derivative admission, jet display, determinant assembly, dyadic tolerance, finite request windows, regular rational readback, terminal Real sealing, hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

Proof. Start with the carrier and NameCert obligations, Definition 6877.1 and Definition 6877.2. They expose exactly the rows $F, D, J, \Omega, S, R, E, H, C, P, N$ and no additional analytic coordinate.

Use determinant-ledger exactness and determinant-route stability, Definition 6877.3 and Definition 6877.6. These results keep determinant reads on the route through $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Finally apply the linear-dependence boundary, witness route, and Real-seal nonescape theorems, Definition 6877.7, Definition 6877.8, and Definition 6877.9. They show that every consumer-facing read remains within the displayed obligation surface and exports none of the excluded host or quotient principles. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier obligation closure scope`

Theorem 6877.14 (Wronskian differential-equation consumer handoff)**.** *Every ODE, Sturm–Liouville, or linear-dependence consumer of an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet must enter through the determinant-ledger route of Definition 6877.13 before reading any* $\mathsf{Real}^{\uparrow}$*-seal conclusion. In particular, the consumer must expose $F, D, J, \Omega, S, R, E, H, C, P, N$ and the sibling rows* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it may use a Wronskian value or a linear-dependence verdict.*

Proof. Apply the obligation closure scope of Definition 6877.13. It identifies the determinant-ledger route as the visible surface for every accepted read.

Use the linear-dependence boundary and witness route, Definition 6877.7 and Definition 6877.8. These force ODE-facing and linear-dependence-facing consumers to read the derivative jets, determinant ledger, dyadic tolerance route, StreamName window, and RegSeqRat readback before any verdict is available.

Finally use Real-seal nonescape from Definition 6877.9. The terminal seal row is downstream of the same determinant-ledger route, so no consumer can reach a Real-seal conclusion through host function equality, selected bases, classical zero theorems, quotient real equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier differential equation consumer handoff`

Theorem 6877.15 (Wronskian Sturm separation handoff)**.** *Any* $\mathsf{SturmSeparation}^{\uparrow}$ *or* $\mathsf{SturmRootCount}^{\uparrow}$ *consumer that reads a* $\mathsf{Wronskian}^{\uparrow}$ *zero or nonzero separation must first expose the Wronskian carrier rows*

$$F, D, J, \Omega, S, R, E, H, C, P, N$$

together with the Sturm comparison and root-count sibling route. The handoff supplies only the displayed finite determinant-ledger route through $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; it supplies no host zero theorem, selected basis relation, quotient real equality, countable choice, or ambient completeness.*

Proof. Project the Wronskian carrier of Definition 6877.1. The rows $F, D, J, \Omega, S, R, E, H, C, P, N$ display the function family, derivative admission, jet table, determinant ledger, finite request windows, regular rational readback, Real seal, transport, replay, provenance, and local naming before any consumer read is available.

Apply determinant-ledger exactness and determinant-route stability, Definition 6877.3 and Definition 6877.6. These keep zero and nonzero determinant reads attached to the same displayed route through continuous-map admission, derivative admission, finite windows, dyadic tolerance, regular rational readback, and the terminal Real seal.

Use the linear-dependence boundary and Real-seal nonescape results, Definition 6877.7 and Definition 6877.9. They prevent the consumer from replacing the displayed determinant ledger by host function equality, selected bases, quotient real equality, countable choice, or ambient completeness.

Finally route the zero-separation read through the existing Sturm sibling chapters as consumers. The $\mathsf{SturmSeparation}^{\uparrow}$ and $\mathsf{SturmRootCount}^{\uparrow}$ rows therefore receive a Wronskian handoff only after the determinant and Real-facing rows have been shown; they do not add new coordinates to the Wronskian packet. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier sturm separation handoff`

Theorem 6877.16 (Wronskian differential scope ledger)**.** *Every determinant, zero-separation, or linear-dependence consumer of an accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet must expose the* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{SturmSeparation}^{\uparrow}$ *and* $\mathsf{SturmRootCount}^{\uparrow}$ *handoff rows before any local NameCert export is available. In carrier notation, the visible differential scope is the route*

$$F, D, J, \Omega, S, R, E, H, C, P, N$$

together with the Sturm handoff of Definition 6877.15. The ledger exports no host derivative equality, selected differentiability witness, selected basis relation, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or Sturm conclusion outside the displayed route.*

Proof. Project the Wronskian carrier of Definition 6877.1. The rows $F, D, J, \Omega, S, R, E, H, C, P, N$ are the only displayed continuous-map, derivative, jet, determinant, stream-window, regular-readback, Real-seal, transport, replay, provenance, and local naming coordinates.

Use determinant-ledger exactness and derivative-route stability through Definition 6877.3 and Definition 6877.6. These keep every determinant read attached to $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a consumer can inspect the value.

Apply the linear-dependence boundary and Real-seal nonescape, Definition 6877.7 and Definition 6877.9. They keep zero and nonzero reads on the same finite route and prevent replacement by host derivative equality, selected differentiability witnesses, selected bases, quotient real equality, countable choice, or ambient completeness.

Finally use the Sturm separation handoff of Definition 6877.15. The $\mathsf{SturmSeparation}^{\uparrow}$ and $\mathsf{SturmRootCount}^{\uparrow}$ consumers receive only the displayed Wronskian route, so local NameCert export remains scoped to the differential ledger named in the statement. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier differential scope ledger`

Theorem 6877.17 (Wronskian L10 scope binding)**.** *Every accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet consumed by determinant, derivative, or Sturm-separation routes must expose the finite route through* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *windows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Wronskian}^{\uparrow}}$ *row before any zero or nonzero determinant conclusion is public. The route exports no host differentiable function, selected representative, quotient stream, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier l10 scope binding`

Proof. Project the Wronskian carrier of Definition 6877.1. The carrier displays $F, D, J, \Omega, S, R, E, H, C, P, N$, so every determinant-facing consumer starts from continuous-map admission, derivative admission, the finite jet table, and the determinant ledger.

Apply determinant-ledger exactness and determinant-route stability, Definition 6877.3 and Definition 6877.6. These force determinant, zero, and nonzero reads through the visible $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route before any public conclusion is available.

Use Real-seal nonescape and the Sturm separation handoff, Definition 6877.9 and Definition 6877.15. The terminal seal and the Sturm-facing consumer both remain downstream of the same displayed route.

Finally apply the differential scope ledger of Definition 6877.16. Since host differentiable functions, selected representatives, quotient streams, trichotomy branches, and ambient completeness are absent from the carrier and classifier, the L10 public boundary is exactly the finite row route named in the statement. □

Theorem 6877.18 (Wronskian obligation-closure package)**.** *The carrier of Definition 6877.1, the NameCert obligations of Definition 6877.2, determinant-ledger exactness, determinant-route stability, the linear-dependence boundary, the witness route, and Real-seal nonescape assemble the* $\mathsf{Wronskian}^{\uparrow}$ *obligationClosure surface over* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The assembled package exports*

no host function equality, selected differentiability witness, quotient Real equality, countable choice, ambient completeness, or bridge row.

Proof. Project the carrier rows $F, D, J, \Omega, S, R, E, H, C, P, N$ from Definition 6877.1. The NameCert obligation theorem reads these as continuous-map admission, derivative admission, finite jet exposure, determinant exposure, StreamName windowing, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Use determinant-ledger exactness and determinant-route stability. They bind determinant reads to the displayed finite route through $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and keep the route invariant under componentwise hsame transport.

Apply the linear-dependence boundary, witness route, and Real-seal nonescape theorems. These force every ODE-facing or Real-facing consumer through the same rows and show that the excluded host principles are absent from the carrier, classifier, transport, replay, provenance, and local NameCert surfaces. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier obligation closure package`

Theorem 6877.19 (Wronskian public determinant export)**.** *Every accepted* $\mathsf{Wronskian}^{\uparrow}$ *packet exports exactly the finite determinant ledger route through* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *The exported public surface is the displayed route*

$$F, D, J, \Omega, S, R, E, H, C, P, N,$$

with determinant, zero-separation, and linear-dependence reads confined to those rows. It exports no host function equality, selected differentiability witness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the Wronskian carrier of Definition 6877.1. The rows $F, D, J, \Omega, S, R, E, H, C, P, N$ are the only continuous-map, derivative, jet, determinant, finite-window, readback, Real-seal, transport, replay, provenance, and naming coordinates available to a public consumer.

Apply determinant-ledger exactness and determinant-route stability, Definition 6877.3 and Definition 6877.6. These force every determinant read through the same displayed route and keep that route fixed under componentwise hsame transport.

Use the linear-dependence boundary and Real-seal nonescape theorems, Definition 6877.7 and Definition 6877.9. They place zero-separation and linear-dependence reads downstream of the determinant ledger and terminal Real seal without adding host equality or a selected differentiability witness.

Finally apply the L10 scope binding of Definition 6877.17. The public export is therefore exactly the finite route named in the statement, and every excluded quotient, host, or ambient-completeness principle remains outside the packet. □

Lean4 ✓ `BEDC.Derived.WronskianUp.WronskianCarrier public determinant export`

Closure status for $\mathsf{NameCert}_{\mathsf{Wronskian}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Wronskian}^{\uparrow}$ packet is generated from a finite continuous-map family row, derivative-admission rows, a displayed jet table, an alternating determinant ledger, finite StreamName requests, regular rational readback, a terminal Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Wronskian}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 6877.1, Definition 6877.2, Definition 6877.3, Definition 6877.6, Definition 6877.7, Definition 6877.8, Definition 6877.9, Definition 6877.10, Definition 6877.11, Definition 6877.12, Definition 6877.17, and Definition 6877.18 bind the finite Wronskian carrier, determinant-ledger route, transport stability, linear-dependence boundary, witness route, Real-seal nonescape surface, carrier obligations, classifier stability, ledger exactness, L10 scope boundary, and obligation package.

Formal verification: FormalStatus(NameCert$_{\text{Wronskian}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, classical zero theorems, selected bases, selected differentiability witnesses, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Public export requires a checked Wronskian carrier theorem family, public consumer surface, and bridge discipline for BEDC.Derived.WronskianUp.

Chapter 6878

A Concrete Naming Certificate for $\mathsf{LandauKolmogorovInequality}^{\uparrow}$

Following Landau and Kolmogorov, the $\mathsf{LandauKolmogorovInequality}^{\uparrow}$ packet records the finite derivative-window route by which an intermediate derivative norm is bounded between a low-order norm ledger and a high-order derivative norm ledger. It sits beside the asymptotic comparison surface of chapter 2252, the finite-window interpolation route of chapter 6814, the norm carrier of chapter 89, and the terminal real-name seal of chapter 38. The packet names a constructive analysis inequality surface over displayed derivative and norm rows; it is not a host function equality principle, an unbounded differentiability tower, a measure completion theorem, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 6878.1 (Landau–Kolmogorov inequality carrier)**.** *A* $\mathsf{LandauKolmogorovInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, N_0, N_k, N_m, A, W, R, E, H, C, P, Z)$$

with the following displayed rows:

1. *D is the derivative-window ledger, naming the finite orders $0 < k < m$ and the inspected derivative windows used by the estimate.*
2. *N_0, N_k, N_m are the* $\mathsf{Norm}^{\uparrow}$ *rows for the base function, the intermediate derivative, and the highest displayed derivative.*
3. *A is the interpolation-exponent row, storing the finite rational exponent data and coefficient budget used by the Landau–Kolmogorov comparison.*
4. *W is the* $\mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow}$ *handoff row that places the displayed derivative windows and norm ledgers in one finite interpolation packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational inequality budget and all finite tolerance requests.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after $D, N_0, N_k, N_m, A, W, R$ have been displayed.*
7. *H records componentwise* hsame *transport for the derivative-window, norm-ledger, exponent, interpolation, readback, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a derivative-window request through the norm ledgers, interpolation exponents, regular rational readback, and Real seal.*

9. *P is the* Pkg *provenance over* LandauKolmogorovInequality$^{\uparrow}$, LandauSymbol$^{\uparrow}$, GagliardoNirenbergFiniteWindow$^{\uparrow}$, Norm$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Z is the local* NameCert$_{\text{LandauKolmogorovInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, N_0, N_k, N_m, A, W, R, E, H, C, P, Z$. The carrier has no coordinate for host function equality, a choice-selected derivative representative, an infinite differentiability tower, measure completion, quotient norm equality, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6878.2 (Landau–Kolmogorov inequality NameCert obligations)**.** *For every accepted* LandauKolmogorovInequality$^{\uparrow}$ *carrier*

$$K = (D, N_0, N_k, N_m, A, W, R, E, H, C, P, Z),$$

the local NameCert$_{\text{LandauKolmogorovInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed derivative-window ledger, three norm ledgers, interpolation-exponent row, finite-window handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *derivative-window exactness, requiring the intermediate derivative request and the highest derivative request to be read through the same finite row D before any norm comparison is exposed;*
3. *norm-ledger accountability, requiring N_0, N_k, N_m to be the only norm rows available to the inequality read;*
4. *interpolation-exponent exactness, requiring A and W to store the rational exponent data and finite interpolation budget before R may read the inequality as regular rational data;*
5. *Real-seal nonescape, requiring every Real-facing conclusion to pass through R and then E;*
6. *stability under componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row;*
7. *refusal of host function equality, selected derivative representatives, unbounded differentiability towers, measure completion, quotient norm equality, countable choice, and ambient* Real$^{\uparrow}$ *completeness as local certificate data.*

No Landau–Kolmogorov inequality obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6878.1. The accepted packet exposes only the derivative-window row, three norm rows, interpolation data, regular rational readback, the Real seal, and the structural rows.

The derivative and norm obligations are finite. The row D fixes the orders and inspected windows, while N_0, N_k, N_m are the only norm ledgers available to the comparison; no host derivative representative or quotient norm equality can be read outside them.

The interpolation step is routed through A and W. These rows store the rational exponent data and the finite-window handoff already associated with GagliardoNirenbergFiniteWindow$^{\uparrow}$, so the estimate reaches R only as displayed regular rational data.

Finally, E, H, C, P, Z seal and preserve the same route. Since the excluded host-function, choice, infinite-tower, measure-completion, quotient, and ambient-completeness objects are not carrier rows, the local NameCert surface cannot export them. □

Theorem 6878.3 (Landau–Kolmogorov derivative-norm handoff)**.** *Every derivative-norm consumer of an accepted* $\mathsf{LandauKolmogorovInequality}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{LandauSymbol}^{\uparrow},\ \mathsf{GagliardoNirenbergFiniteWindow}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCe}$$

The intermediate derivative norm is therefore read only after the derivative-window ledger, base and high-order norm ledgers, interpolation-exponent row, finite-window handoff, regular rational readback, and Real seal have all been displayed. The handoff exports no host function equality, selected derivative representative, infinite differentiability tower, measure completion, quotient norm equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $K = (D, N_0, N_k, N_m, A, W, R, E, H, C, P, Z)$. The derivative-norm request can inspect the intermediate row N_k only together with the derivative-window ledger D and the endpoint norm rows N_0, N_m.

The interpolation data then passes through A and W. The row W is the finite Gagliardo–Nirenberg-style handoff, and A supplies the rational exponent budget read by the estimate.

The Real-facing read is downstream of R and E. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same finite chain, so the consumer never receives an external equality, selection, infinite tower, measure completion, quotient, choice, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LandauKolmogorovInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LandauKolmogorovInequality}^{\uparrow}$ packet is generated from one derivative-window ledger, three NormUp ledgers, one rational interpolation-exponent row, one GagliardoNirenbergFiniteWindowUp handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LandauKolmogorovInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6878.1, Definition 6878.2, and Definition 6878.3 seed the finite derivative-window interpolation route through norm ledgers, RegSeqRat readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LandauKolmogorovInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected derivative representatives, infinite differentiability towers, measure completion, quotient norm equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exactness, norm-ledger accountability, interpolation-exponent exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LandauKolmogorovInequalityUp.

Chapter 6879

A Concrete Naming Certificate for $\mathsf{CauchyNetDiagonal}^{\uparrow}$

Following Moore–Smith and Kelley on nets, the $\mathsf{CauchyNetDiagonal}^{\uparrow}$ packet names the finite diagonal extraction route by which a displayed Cauchy net is read through a cofinal schedule and handed to regular rational names. It sits over the Cauchy-net source of chapter 3794, the Moore–Smith net surface of chapter 2190, the stream-window naming surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance scale of chapter 1768, and the Real seal of chapter 38. The chapter records a finite diagonal packet only; it does not add a choice functional, quotient net equality, selected subnet, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6879.1 (Cauchy-net diagonal carrier)**.** *A* $\mathsf{CauchyNetDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, M, K, T, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *source row whose directed-domain and Cauchy evidence are displayed before any diagonal consumer reads the packet.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *source row, recording that the source is consumed as Moore–Smith net data rather than as an unordered family.*
3. *K is the finite cofinal diagonal schedule, listing the observed directed indices and the finite cofinality checks used by the packet.*
4. *T is the dyadic tolerance ledger, recording the requested error levels and the finite comparisons used to meet them.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row obtained by reading the scheduled observations in order.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to those stream windows.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback.*
8. *H records componentwise* hsame *transport for $A, M, K, T, S, Q, R, C, P, N$.*
9. *C records displayed* Cont *replay from the Cauchy-net source through the Moore–Smith row, finite schedule, dyadic ledger, stream window, regular rational readback, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{MooreSmithNet}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, K, T, S, Q, R, H, C, P, N$. It has no coordinate for a choice functional, quotient net equality, selected subnet, selected representative, selected limit, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyNetDiagonalUp.CauchyNetDiagonalUp`

Theorem 6879.2 (Cauchy-net diagonal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetDiagonal}^{\uparrow}$ *carrier $D = (A, M, K, T, S, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyNetDiagonal}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Cauchy-net source, Moore–Smith source, finite cofinal diagonal schedule, dyadic tolerance ledger, stream window, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *source visibility, requiring the rows A and M to remain available before any diagonal schedule or completion-facing read is consumed;*
3. *schedule exactness, requiring K to display the finite cofinal observations and T to display the dyadic tolerance ledger used by those observations;*
4. *readback routing, requiring S, Q, R to be consumed in order from stream windows to regular rational data and then to the Real seal;*
5. *structural stability, requiring H, C, P, N to preserve and name exactly the same finite route.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6879.1. The accepted packet exposes exactly the rows $A, M, K, T, S, Q, R, H, C, P, N$, so the local naming surface can read only those coordinates.

The source rows come first. The Cauchy-net row A supplies directed Cauchy evidence, while the Moore–Smith row M fixes the net interface through which the diagonal route is read.

The finite diagonal step is then local to K and T. The schedule lists the observed directed indices, and the dyadic ledger lists the tolerance comparisons that justify each observed window.

Finally the packet reads S, Q, R as stream windows, regular rational data, and a Real seal. Componentwise hsame, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve this same route without adding any hidden selector or completion theorem. □

Lean4 ✓ `BEDC.Derived.CauchyNetDiagonalUp.CauchyNetDiagonalNameCertObligations`

Theorem 6879.3 (Cauchy-net diagonal RegSeqRat handoff)**.** *Every regular-rational or Real-facing consumer of an accepted* $\mathsf{CauchyNetDiagonal}^{\uparrow}$ *carrier reads the diagonal route only through*

$$\mathsf{CauchyNet}^{\uparrow}, \mathsf{MooreSmithNet}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The handoff exports no choice functional, quotient net equality, selected subnet, selected representative, selected limit, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier. The only source coordinates are the Cauchy-net row A and the Moore–Smith row M, so a consumer cannot read a detached family, selected subnet, or quotient net before the displayed source route is fixed.

The diagonal observations pass through K and T. These rows are finite data: the cofinal observation schedule and the dyadic tolerance ledger. They do not contain a choice operator or a global completion principle.

The handoff to regular rational names is exactly the route S, Q, R. The stream row supplies the windows, the regular rational row reads those windows, and the Real row seals the resulting name.

The remaining rows H, C, P, N transport, replay, source, and name the same packet. Since none of these rows is a selector, quotient equality, host equality, or ambient completeness theorem, the RegSeqRat and Real-facing handoff is exhausted by the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetDiagonal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6879.1, Definition 6879.2, and Definition 6879.3 seed the finite diagonal extraction route from Cauchy-net and Moore–Smith data to StreamName, RegSeqRat, and Real readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNetDiagonal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyNetDiagonalUp.CauchyNetDiagonalNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close a choice functional, quotient net equality, selected subnet, selected representative, selected limit, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source visibility, finite cofinal schedule exactness, dyadic tolerance ledger exactness, readback routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNetDiagonalUp.

Constructive story: A $\mathsf{CauchyNetDiagonal}^{\uparrow}$ packet is generated from a CauchyNet row, a MooreSmithNet source row, a finite cofinal diagonal schedule, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6880

A Concrete Naming Certificate for $\mathsf{PrueferGroup}^{\uparrow}$

Following Pruefer's p-primary torsion group construction, the $\mathsf{PrueferGroup}^{\uparrow}$ packet records the cyclic p^n-torsion windows and their coherent inclusions as finite BEDC NameCert data. It sits after the operation surface of chapter 41, the commutative specialization of chapter 42, the finite-code discipline of chapter 6390, and the separately topological Pruefer surface route of chapter 6799. The packet names a displayed torsion-window algebra and not an ambient quotient group, a choice-selected compatible root chain, or an infinite colimit object.

Definition 6880.1 (Pruefer group carrier)**.** *A* $\mathsf{PrueferGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (p, W, I, A, T, D, H, C, P, N)$$

with the following displayed rows:

1. *p is the prime-index row naming the fixed torsion prime and the visible arithmetic ledger for p-power exponents.*
2. *W is the finite window row, listing the cyclic torsion stages currently exposed to the packet as $p, p^2, \ldots, p^k$ windows.*
3. *I is the coherent inclusion row, recording the displayed stage-to-stage embeddings from the p^n window to the p^{n+1} window.*
4. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *operation row for the displayed windows, retaining the group operation, unit, inverse, operation congruence, and commutativity read through the sibling abelian-group certificate.*
5. *T is the torsion-window classifier row, comparing accepted endpoints only by a common displayed p^m window and its inclusion ledger.*
6. *D is the divisibility handoff row: every displayed endpoint has a next-window predecessor only when the relevant inclusion row in I is present.*
7. *H records componentwise* hsame *transport for the prime, window, inclusion, abelian-operation, classifier, divisibility, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a torsion endpoint request through its finite window, inclusion route, abelian operation readback, classifier comparison, divisibility handoff, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{PrueferGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{PrueferSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PrueferGroup}^{\uparrow}}$ *row.*

The classifier has no coordinate for quotienting an external group, selecting roots outside the displayed inclusion ledger, or replacing the finite windows by an ambient infinite colimit.

Theorem 6880.2 (Pruefer group NameCert obligations)**.** *For every accepted* $\mathsf{PrueferGroup}^{\uparrow}$ *carrier*

$$G = (p, W, I, A, T, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrueferGroup}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the fixed-prime row, finite p-power torsion windows, coherent inclusion ledger, abelian operation row, transport, replay, provenance, and local naming;*
2. *p-power compatibility, requiring each operation and inverse read in A to remain inside a displayed window and to commute with the inclusion rows of I;*
3. *torsion-window classifier stability, requiring T to compare endpoints only after transporting them to a common displayed p^m window;*
4. *divisibility handoff, exposing a predecessor in the next torsion window exactly through the corresponding row of D and not through a global root selector;*
5. *ledger refusal excluding quotient groups, choice-selected roots, and ambient infinite colimits from the accepted packet.*

Proof. Project the carrier of Definition 6880.1. The tuple displays the prime row, finite torsion windows, coherent inclusions, abelian-group operation rows, torsion classifier, divisibility handoff, transport, replay, provenance, and local name as separate rows.

The carrier-admission row reads p, W, I, A, H, C, P, N before any classifier or divisibility claim is available. Since W is finite and I lists only displayed stage-to-stage inclusions, admission is a finite BHist read.

For compatibility, inspect A together with I. The operation, unit, inverse, congruence, and commutativity rows are inherited through chapter 42, while each operation endpoint is checked inside a visible p^n window and transported along the named inclusions.

The classifier row T compares two accepted endpoints only after a common displayed window has been named. The divisibility row D is then a next-window handoff for that displayed endpoint, not a selection principle over all roots.

Finally, H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. No row supplies quotient-group equality, a choice-selected root chain, or an ambient colimit carrier, so the five listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{PrueferGroup}^{\uparrow}$ read displays a fixed prime row, finite p-power torsion windows, coherent inclusion rows, abelian operation rows, torsion-window classifier stability, divisibility handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: group, abgroup, pruefer-sequence, pruefer-surface: $\mathsf{PrueferGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds finite p-power torsion windows, coherent inclusions, abelian operation rows, torsion-window classifier stability, and divisibility handoff into one algebraic NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PrueferGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrueferGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6880.1 and Definition 6880.2 seed the finite fixed-prime, torsion-window, coherent-inclusion, abelian-operation, classifier, divisibility, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrueferGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient-group construction, selected compatible root chains, ambient infinite colimits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, p-power compatibility, torsion-window classifier stability, divisibility handoff, and nonescape for BEDC.Derived.PrueferGroupUp.

Constructive story: A $\mathsf{PrueferGroup}^{\uparrow}$ packet is generated from a fixed prime row, finite p-power torsion windows, coherent inclusion rows, abelian operation rows, torsion-window classifier rows, divisibility handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6881

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$

Following Bishop and Bridges on diagonal Cauchy arguments, the $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ packet records the finite evidence by which a family of regular rational Cauchy routes is read along one displayed diagonal schedule before a Real or Completion consumer receives a seal. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the completion-facing envelope of chapter 6864, and the diagonal filter route of chapter 6873. The certificate names a finite diagonal criterion only; it does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, or countable choice.

Definition 6881.1 (Cauchy diagonal criterion carrier)**.** *A* $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the finite precision request and comparison slack used by the diagonal read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *diagonal schedule row, naming the finitely inspected stream windows and the index route used for the diagonal passage.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row obtained from the displayed windows and dyadic tolerance request.*
4. *R is the regular route row stating that the same D, S, Q packet answers the Cauchy request before any endpoint seal is read.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback is displayed.*
6. *H records componentwise* hsame *transport for the tolerance, schedule, readback, regular-route, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a diagonal Cauchy request through D, S, Q, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host Cauchy-completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6881.2 (Cauchy diagonal criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ *carrier* $K = (D, S, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, dyadic tolerance exposure through* D*, diagonal stream-window exposure through* S*, regular rational readback through* Q*, regular-route handoff through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 6881.1. The only visible rows are $D, S, Q, R, E, H, C, P, N$, so the local NameCert surface has no extra coordinate from which to export a quotient stream, selected limit, or completeness principle.

Read the diagonal evidence through D, S, Q, R. The dyadic tolerance row fixes the precision request, the stream row fixes the finite windows and diagonal schedule, the regular rational row supplies readback, and the regular-route row records that these same data answer the Cauchy request.

Finally use E, H, C, P, N. The Real seal is downstream of the regular readback, and transport, replay, provenance, and local naming preserve the same displayed packet. Therefore the listed rows exhaust the NameCert obligations. □

Theorem 6881.3 (Cauchy diagonal criterion regular handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ *packet reaches its endpoint only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The diagonal criterion supplies Bishop-style finite Cauchy evidence for regular-Cauchy Real routes, while exporting no quotient stream equality, selected limit, countable-choice schedule, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start from the tolerance row D and the stream row S. Together they display the finite precision request, diagonal schedule, and stream windows used by the criterion.

Read the regular rational row Q and regular-route row R next. These rows are downstream of the same D, S data, so a consumer cannot obtain a regular Cauchy readback without first inspecting the displayed dyadic and stream evidence.

The terminal Real row E is available only after Q. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve that order. Thus the completion-facing handoff remains finite and does not import quotient streams, selected limits, countable choice, or Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ endpoint read displays a DyadicRatCore tolerance request, StreamName diagonal windows, RegSeqRat regular readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Real or Completion consumer may use the route.

Independence witness: dyadicratcore, streamname, regseqrat, real, fan-modulus-real-completion-boundary, cauchy-filter-diagonal: no listed sibling supplies the whole packet $K = (D, S, Q, R, E, H, C, P, N)$, because each sibling contributes only tolerance arithmetic, finite stream naming, regular rational

readback, endpoint sealing, completion-facing envelope data, or filter diagonal routing while this packet binds the Bishop-style diagonal criterion into one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalCriterion}\uparrow}$.

Constructive story: A $\mathsf{CauchyDiagonalCriterion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance request, StreamName diagonal windows, RegSeqRat regular readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6881.1, Definition 6881.2, and Definition 6881.3 seed the finite diagonal Cauchy route from dyadic tolerance data and stream windows to regular rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient streams, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal schedule exposure, dyadic tolerance exactness, regular readback stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyDiagonalCriterionUp.

Chapter 6882

A Concrete Naming Certificate for $\mathsf{EgorovTheorem}^{\uparrow}$

Following Egorov and Lebesgue, the $\mathsf{EgorovTheorem}^{\uparrow}$ packet records the theorem-level finite route from almost-uniform convergence data to measure-convergence consumers without adding a host measure-completion primitive. It sits after the finite measure and integration ledgers of chapter 92 and chapter 123, the lower-semicontinuity route of chapter 3447, the domination route of chapter 2340, the large-set continuous approximation route of chapter 4147, and the regular-rational and real-name seals of chapter 301 and chapter 38. The packet is a finite NameCert carrier for the theorem interface: it exports exceptional-set and uniform-window rows that later Fatou, dominated-convergence, and Lusin-style readers can consume, but it does not claim ambient measure completion, almost-everywhere quotient equality, selected representatives, countable choice, or the full host Egorov theorem.

Definition 6882.1 (Egorov theorem carrier)**.** *An* $\mathsf{EgorovTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, I, F, D, L, R, E, A, W, U, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite* $\mathsf{Measure}^{\uparrow}$ *row whose measurable-set ledger bounds every exceptional-set read.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *row used by downstream integral consumers after the exceptional and uniform rows have been displayed.*
3. *F is the* $\mathsf{FatouLemma}^{\uparrow}$ *source row, exposed only as a lower-bound and liminf-facing consumer of the same finite measure ledger.*
4. *D is the* $\mathsf{DominatedConvergence}^{\uparrow}$ *source row, exposed only as a domination and integrability consumer of M, I, A, W, U.*
5. *L is the* $\mathsf{LusinApproximation}^{\uparrow}$ *source row, exposing large-set continuous approximation only after the exceptional-set budget is visible.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route for all displayed real-valued observations and tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the regular-rational readback route, not by a host real-completeness argument.*

8. *A is the finite exceptional-set row, recording the measurable approximants, the finite measure budget, and containment checks used by the theorem packet.*
9. *W is the finite uniform-window row, recording the index threshold, precision requests, and observation sites on which the theorem route is uniform outside A.*
10. *U is the almost-uniform convergence row, asserting that the displayed R, E, W comparisons are uniform on the complement of the displayed exceptional budget.*
11. *Q is the theorem-handoff row, routing accepted reads from A, W, U into the Fatou, dominated-convergence, Lusin, and integral consumers without changing their carrier rows.*
12. *H records componentwise* hsame *transport for $M, I, F, D, L, R, E, A, W, U, Q$ and the structural rows.*
13. *C records displayed* Cont *replay of the finite route*
$$M, I, F, D, L, R, E, A, W, U, Q \longmapsto H, P, N.$$
14. *P is the* Pkg *provenance over* $\mathsf{EgorovTheorem}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FatouLemma}^{\uparrow}$, $\mathsf{DominatedConvergence}^{\uparrow}$, $\mathsf{LusinApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{EgorovTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, I, F, D, L, R, E, A, W, U, Q, H, C, P, N$. It has no coordinate for completed measure algebras, quotient almost-everywhere equality, selected representatives, countable exceptional unions, countable choice, or an ambient theorem object outside the finite carrier.

Theorem 6882.2 (Egorov theorem NameCert obligations)**.** *For every accepted* $\mathsf{EgorovTheorem}^{\uparrow}$ *carrier*
$$G = (M, I, F, D, L, R, E, A, W, U, Q, H, C, P, N),$$
the local $\mathsf{NameCert}_{\mathsf{EgorovTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite measure row, integral row, Fatou row, dominated-convergence row, Lusin row, regular-rational readback row, real-name seal, exceptional-set row, uniform-window row, almost-uniform convergence row, theorem-handoff row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every theorem-level public read must pass through A, W, U, Q before it can reach the Fatou, dominated-convergence, Lusin, or integral consumers.*

Proof. Project the carrier of Definition 6882.1. The theorem-facing rows are A, W, U, Q: they name the finite exceptional budget, the finite uniform window, the almost-uniform comparison, and the handoff to downstream consumers.

The rows M, I, F, D, L, R, E supply the measure, integral, Fatou, dominated-convergence, Lusin, regular-rational, and real-name dependency surface on which those four rows are interpreted. The structural rows H, C, P, N transport, replay, source, and name exactly the same finite route.

Since the carrier contains no completed measure algebra, quotient almost-everywhere equality, selected representative, countable exceptional union, countable choice row, or ambient theorem object, the listed rows exhaust the local NameCert obligations. □

Theorem 6882.3 (Egorov theorem finite handoff)**.** *For every accepted* $\mathsf{EgorovTheorem}^{\uparrow}$ *carrier G, any downstream almost-uniform convergence read by a Fatou, dominated-convergence, Lusin, or integral consumer factors through the finite route*
$$A, W, U, Q \longrightarrow F, D, L, I.$$

The handoff preserves the displayed $\mathsf{Measure}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows and does not replace them by selected exceptional sets, almost-everywhere quotient equality, selected representatives, countable choice, or ambient measure completion.*

Proof. Start with the theorem-handoff row Q. By carrier admission, Q is displayed only together with the exceptional-set row A, uniform-window row W, and almost-uniform convergence row U.

Read the downstream consumers next. The Fatou row F, dominated-convergence row D, Lusin row L, and integral row I receive the same finite measure ledger and the same regular-rational and real-name seals already named by M, R, E.

Transport and replay through H and C preserve those rows componentwise. Because no selected exceptional set, quotient almost-everywhere equality, selected representative, countable-choice row, or measure-completion row occurs in G, the handoff is exhausted by the displayed finite route. □

Theorem 6882.4 (Egorov theorem exceptional-window nonescape)**.** *Every accepted* $\mathsf{EgorovTheorem}^{\uparrow}$ *packet exposes its theorem-level convergence content only through the paired exceptional-set and uniform-window rows* A, W *together with the almost-uniform row* U*. A consumer may refine a finite observation schedule through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows, but it may not bypass* A, W, U *by invoking ambient measure completion, almost-everywhere quotient equality, selected representatives, countable choice, or the full host Egorov theorem.*

Proof. The finite observation schedule is read through W, and each precision comparison is evaluated through R and sealed by E. The exceptional-set budget is read through A, whose finite measure ledger is interpreted by M.

The almost-uniform row U binds these two reads together. Thus any refinement of observations that preserves the same A, W, U, R, E data gives the same public theorem-level verdict.

No other carrier coordinate can export the convergence content. In particular Q, H, C, P, N only hand off, transport, replay, source, and name the displayed rows, so the forbidden ambient objects have no projection from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EgorovTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{EgorovTheorem}^{\uparrow}$ packet is generated from finite Measure and Integral rows, Fatou, DominatedConvergence, and Lusin consumer rows, RegSeqRat readbacks, Real seals, exceptional-set and uniform-window rows, almost-uniform handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EgorovTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6882.1, Definition 6882.2, Definition 6882.3, and Definition 6882.4 seed the finite theorem-level almost-uniform convergence route through exceptional-set, uniform-window, Fatou, dominated-convergence, Lusin, integral, regular-rational, real-name, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EgorovTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient measure completion, almost-everywhere quotient equality, selected representatives, countable choice, the full host Egorov theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exceptional-set budget exactness, uniform-window exactness, almost-uniform handoff, consumer routing, transport stability, and replay stability for BEDC.Derived.EgorovTheoremUp.

Chapter 6883

A Concrete Naming Certificate for PolymorphicInstantiationCoverage$^{\uparrow}$

The PolymorphicInstantiationCoverage$^{\uparrow}$ packet realizes the typed-example frontier of section 9386.6 as finite universe-instantiation and typed-window data. It sits beside the MetaCIC gallery of chapter 775, the typed-example matrix of chapter 923, the dependent-eliminator coverage packet of chapter 6650, the typed-conversion residual frontier of chapter 6069, and the proof-search frontier of chapter 4284. It records where polymorphic examples have been instantiated inside finite typed windows, while keeping eliminator coverage, conversion residuals, proof search, stream observation, and regular-rational readback as separate sibling routes through chapter 126 and chapter 301.

Definition 6883.1 (Polymorphic-instantiation coverage carrier)**.** *A* PolymorphicInstantiationCoverage$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, W, U, T, V, O, X, R, S, B, H, C, K, N)$$

with the following displayed rows:

1. *G is the finite typed-example gallery row whose polymorphic identity, type-application, and Church-encoded examples are being audited.*
2. *W is the typed-window row, recording the finite contexts, binders, sort levels, type arguments, and term endpoints inspected by the packet.*
3. *U is the universe-instantiation row, listing the concrete universe and sort substitutions accepted by the window.*
4. *T is the typed-instantiation row, pairing polymorphic source terms with displayed type arguments and typed target endpoints.*
5. *V is the conversion-boundary row, recording where beta, type application, and definitional conversion checks are required before an instantiated example is counted.*
6. *O is the omission-obstruction row, retaining any polymorphic example whose type argument, universe row, or conversion boundary is missing from W, U, T, V.*
7. *X is the transport row from the typed-example coverage matrix into the local polymorphic-instantiation packet.*
8. *R is the replay row, recording bounded re-checks of the accepted instantiation windows without claiming global coverage of all polymorphic programs.*

9. *S is the source-provenance row, separating gallery examples, dependent-eliminator siblings, conversion-frontier siblings, proof-search siblings, stream observations, and regular-rational readbacks.*
10. *B is the boundary row refusing dependent-eliminator coverage, unrestricted conversion residual closure, proof-search completeness, and semantic parametricity as consequences of this packet.*
11. *H records componentwise* hsame *transport for $G, W, U, T, V, O, X, R, S, B$ and the structural rows.*
12. *C records displayed* Cont *replay of the finite route*

$$G, W, U, T, V, O, X, R, S, B \longmapsto H, K, N.$$

13. *K is the* Pkg *provenance over* $\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}$, $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$, $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$, $\mathsf{DependentEliminatorCoverage}^{\uparrow}$, $\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}$, $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, W, U, T, V, O, X, R, S, B, H, C, K, N$. It has no coordinate for all dependent eliminators, a full conversion algorithm, a proof-search completeness theorem, semantic parametricity, quotient syntax equality, host universe polymorphism, or global MetaCIC consistency.

Theorem 6883.2 (Polymorphic-instantiation coverage NameCert obligations)**.** *For every accepted* $\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}$ *carrier*

$$P = (G, W, U, T, V, O, X, R, S, B, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}}$ *surface consists exactly of carrier admission for the gallery row, typed-window row, universe-instantiation row, typed-instantiation row, conversion-boundary row, omission-obstruction row, coverage-matrix transport row, replay row, source-provenance row, boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A polymorphic example is counted only when W, U, T, V are displayed together and the omission row O records no missing instantiation-window component for that example.*

Proof. Project the accepted carrier of Definition 6883.1. The polymorphic-instantiation surface is exactly W, U, T, V: the typed window, universe substitutions, term-level instantiations, and conversion boundary.

The gallery and coverage-matrix rows G, X identify the source examples being audited. The rows O, R, S, B record omissions, bounded replay, provenance, and the separation from dependent-eliminator, conversion, proof-search, stream, and regular-rational sibling routes.

Transport, replay, package provenance, and local naming through H, C, K, N preserve the same rows. Since no row stores total dependent-eliminator coverage, full conversion closure, proof-search completeness, semantic parametricity, quotient syntax equality, host universe polymorphism, or global consistency, the listed rows exhaust the NameCert obligations. □

Theorem 6883.3 (Polymorphic-instantiation omission refutation)**.** *If a proposed typed-example coverage packet counts a polymorphic example while omitting the corresponding type argument from T, universe entry from U, typed window from W, or conversion boundary from V, then it is not accepted as a* $\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}$ *carrier. The omitted component is retained in O as a refutation of the proposed coverage count.*

Proof. Acceptance is read from the carrier rows W, U, T, V, O. A counted example must display its typed window, universe substitution, typed instantiation endpoint, and conversion boundary in the same packet.

If any one of those components is absent, the omission-obstruction row O is the only available destination for the proposed count. The replay row R may re-check displayed windows, but it cannot manufacture a missing universe, type argument, typed endpoint, or conversion boundary.

Thus a packet with such an omission fails carrier admission for the counted example, and the omission itself becomes the local refutation. □

Theorem 6883.4 (Polymorphic-instantiation sibling separation)**.** *Every accepted* PolymorphicInstantiationCoverage *read separates polymorphic instantiation from dependent-eliminator coverage, typed-conversion residuals, proof-search frontiers, stream observation, and regular-rational readback. Those consumers may cite the provenance row S, but they cannot read this packet as their own carrier or as a proof of their closure obligations.*

Proof. The provenance row S names the sibling routes, while the boundary row B records that their obligations are not discharged by the polymorphic-instantiation rows. The local carrier counts only examples whose W, U, T, V rows are displayed.

Dependent eliminators require eliminator-family rows not present here. Typed-conversion residuals require residual-frontier and endpoint rows, proof search requires search-frontier evidence, and stream or regular-rational consumers require their own observation and readback carriers.

Therefore the local packet can route citations to those siblings but cannot replace their NameCert obligations. □

Falsifiable prediction: A claimed polymorphic-instantiation coverage count that omits an instantiation-window component from W, U, T, V is refuted by the omission-obstruction row O.

Independence witness: dependent-eliminator-coverage, metacic-typed-conversion-residual-frontier, metacic-proof-search-frontier, streamname, regseqrat: these siblings carry eliminator-family rows, residual-conversion rows, proof-search rows, stream-observation rows, or regular-rational readbacks, but none carries the combined universe-instantiation, typed-window, typed-instantiation, conversion-boundary, omission, transport, replay, provenance, and local NameCert surface of $\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}$.

The future Lean TasteGate file must provide $\mathsf{Nontrivial\ PolymorphicInstantiationCoverage}^{\uparrow}$ and $\mathsf{FieldFaithful\ PolymorphicInstantiationCoverage}^{\uparrow}$ instances for the displayed rows $G, W, U, T, V, O, X, R, S, B, H, C, K$, before any positive gate may count this packet as field-faithful.

Closure status for $\mathsf{NameCert}_{\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}}$.

Constructive story: A $\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}$ packet is generated from finite typed-example gallery rows, typed windows, universe-instantiation rows, typed-instantiation endpoints, conversion boundaries, omission obstructions, coverage-matrix transport, bounded replay, sibling provenance, boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6883.1, Definition 6883.2, Definition 6883.3, and Definition 6883.4 seed the finite typed-window, universe-instantiation, conversion-boundary, omission, transport, replay, provenance, and local naming surface for polymorphic-instantiation coverage.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolymorphicInstantiationCoverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim dependent-eliminator coverage, full conversion residual closure, proof-search completeness, stream readback closure, regular-rational completeness, semantic parametricity, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, instantiation-window exactness, omission-obstruction soundness, sibling separation, transport stability, replay stability, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.PolymorphicInstantiationCoverageUp.

Chapter 6884

A Concrete Naming Certificate for $\mathsf{UniformlySmoothSpace}^{\uparrow}$

Following Clarkson and Day, the $\mathsf{UniformlySmoothSpace}^{\uparrow}$ packet records a constructive modulus-of-smoothness surface over the normed and Banach carriers of chapter 1950 and **??**. It reads Hilbert comparison rows through chapter 91, convex balancing rows through chapter 208, and scalar tolerance rows through chapter 38. The packet names finite smoothness data for future duality, Clarkson-inequality, and constructive fixed-point routes without importing Hahn–Banach separation, weak compactness, reflexivity, or host norm equality.

Definition 6884.1 (Uniformly smooth space carrier)**.** *A* $\mathsf{UniformlySmoothSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (N, B, R, H, K, \rho, E, T, C, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row exposing vector addition, scalar action, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *handoff row. It allows complete-norm consumers to read the same normed carrier but does not select a completed point or identify a host Banach norm.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *row carrying positive dyadic tolerances, norm comparisons, and the scalar budgets used by the smoothness request.*
4. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *comparison row. It records the quadratic-model sibling used as a downstream check, not a declaration that the source norm comes from an inner product.*
5. *K is the* $\mathsf{ConvexSet}^{\uparrow}$ *row naming the balanced unit-ball window in which the paired perturbations are displayed.*
6. *ρ is the modulus-of-smoothness row. For every displayed positive scalar request t it carries a Real budget $\rho(t)$ together with the row*

$$\frac{\|x + ty\| + \|x - ty\|}{2} - 1 \leq \rho(t)$$

   for displayed unit rows x, y admitted by N, K, R.
7. *E is the small-scale stability row requiring the displayed ratio $\rho(t)/t$ to be transported by the same Real tolerance ledger as t is refined toward the local smoothness window.*

8. *T records componentwise* hsame *transport for $N, B, R, H, K, \rho, E, C, P, L$.*
9. *C records displayed* Cont *replay from a smoothness request through*

$$N, B, R, H, K, \rho, E \longmapsto T, P, L.$$

10. *P is the* Pkg *provenance over* $\mathsf{UniformlySmoothSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformlySmoothSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for Hahn–Banach separation, weak compactness, reflexivity, host norm equality, a quotient dual, selected supporting functionals, or an ambient differentiability theorem.

Theorem 6884.2 (Uniformly smooth modulus stability)**.** *For every accepted* $\mathsf{UniformlySmoothSpace}^{\uparrow}$ *carrier $S = (N, B, R, H, K, \rho, E, T, C, P, L)$, the modulus row ρ is stable under componentwise* hsame *transport of the normed-space, Banach handoff, Real tolerance, Hilbert comparison, and convex unit-ball rows. In particular, any accepted consumer that refines the displayed scalar request t must replay the same route*

$$N, B, R, H, K, \rho, E, T, C, P, L$$

before reading the inequality row for $\rho(t)$ or the local ratio row $\rho(t)/t$.

Proof. Start from carrier admission in Definition 6884.1. The normed-space row N, Real row R, and convex unit-ball row K determine the displayed unit vectors, scalar request, and norm comparisons used in the smoothness inequality.

Transport is then componentwise. The row T carries hsame for the normed, Banach, Real, Hilbert, convex, modulus, and small-scale rows, so replacing a displayed endpoint must replace the whole modulus route rather than only the value $\rho(t)$.

Finally replay the request through C. Since E ties $\rho(t)/t$ to the same Real tolerance ledger as t, scalar refinement cannot bypass the original rows. The absent Hahn–Banach, weak-compactness, reflexivity, quotient-dual, and host equality objects therefore have no carrier coordinate from which to alter the modulus read. □

Theorem 6884.3 (Uniformly smooth NameCert obligation)**.** *For every accepted* $\mathsf{UniformlySmoothSpace}^{\uparrow}$ *carrier $S = (N, B, R, H, K, \rho, E, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{UniformlySmoothSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, Banach handoff, Real tolerance readback, Hilbert comparison, convex unit-ball balancing, modulus-of-smoothness inequality, small-scale ratio stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $N, B, R, H, K, \rho, E, T, C, P, L$.*

Proof. Project the carrier. The analytic rows are N, B, R, H, K, ρ, E: normed-space data, complete-norm handoff, scalar tolerance, Hilbert comparison, balanced convex window, modulus inequality, and small-scale stability.

The structural rows T, C, P, L transport, replay, source, and name those analytic rows as one finite packet. They add no supporting functional, quotient dual, weak compactness, reflexivity theorem, Hahn–Banach separation, or host norm equality.

Thus a consumer receives a smoothness certificate only by displaying all listed rows. Since the classifier has no other coordinate, the local NameCert obligation surface is exactly the finite packet described above. □

Falsifiable prediction: An accepted $\mathsf{UniformlySmoothSpace}^{\uparrow}$ read is refuted if a consumer obtains a modulus-of-smoothness inequality or a $\rho(t)/t$ stability row without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: normedspace, banachspace, hilbert, convexset, real: $\mathsf{UniformlySmoothSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries normed-space data, Banach handoff, Real tolerance budgets, Hilbert comparison, convex unit-ball balancing, a modulus-of-smoothness row, small-scale stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlySmoothSpace}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{UniformlySmoothSpace}^{\uparrow}$ packet is generated from NormedSpace rows, BanachSpace handoff rows, Real tolerance rows, Hilbert comparison rows, ConvexSet unit-ball rows, a modulus-of-smoothness row, small-scale ratio stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlySmoothSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6884.1, Definition 6884.2, and Definition 6884.3 seed the finite modulus-of-smoothness route over normed-space, Banach, Real, Hilbert, and convex unit-ball rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlySmoothSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hahn–Banach separation, weak compactness, reflexivity, host norm equality, quotient-dual construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus inequality exactness, small-scale ratio stability, transport, replay, provenance, and local naming for BEDC.Derived.UniformlySmoothSpaceUp.

Chapter 6885

A Concrete Naming Certificate for $\mathsf{PathConnectedSpace}^{\uparrow}$

Following Hausdorff and Bishop–Bridges topology, the $\mathsf{PathConnectedSpace}^{\uparrow}$ packet records path connectedness as a finite endpoint-pair and displayed-path route over topology and continuity data. It sits beside the connectedness carrier of chapter 2353, the local comparison surface of chapter 5809, the continuity route of chapter 59, and the interval-connectedness source of chapter 4369. The certificate names the finite means by which endpoint pairs, path families, endpoint evaluations, continuous replay, and connectedness consumers are exposed without importing host path spaces, quotient path components, or selected paths.

Definition 6885.1 (PathConnectedSpace carrier)**.** *A* $\mathsf{PathConnectedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, F, V, C, G, L, H, R, P, N)$$

with the following displayed rows:

1. *E is the endpoint-pair row. It records each displayed source and target point request through the ambient* $\mathsf{Topology}^{\uparrow}$ *and names the pair before any path witness may be read.*
2. *F is the displayed path-family row. For each accepted endpoint pair it stores the finite path name, interval parameter ledger, and source-to-target scheduling row visible to consumers.*
3. *V is the endpoint-evaluation ledger. It records that the displayed path reads the left endpoint as the first point in E and the right endpoint as the second point in E.*
4. *C is the* $\mathsf{Continuous}^{\uparrow}$ *replay row for each displayed path, including the inverse-image open rows and the finite continuity evidence used by downstream readers.*
5. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *consumer route. It records that a connectedness consumer may read the path route only after the endpoint, family, endpoint-evaluation, and continuity rows have been displayed.*
6. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *comparison row. It records local connectedness as a sibling comparison surface rather than as a source of path selection.*
7. *H records componentwise* hsame *transport for endpoint pairs, displayed path families, endpoint-evaluation ledgers, continuity replay, connectedness consumers, local comparison rows, provenance, and local naming rows.*

8. *R records displayed* Cont *replay from endpoint-pair requests through the displayed path family, endpoint evaluations, continuity replay, connectedness consumer route, and local comparison row.*

9. *P is the* Pkg *provenance over* $\mathsf{PathConnectedSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{IntervalConnectedness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PathConnectedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, F, V, C, G, L, H, R, P, N$. It has no coordinate for host path spaces, quotient path components, selected paths, excluded middle, open-cover compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PathConnectedSpaceUp.PathConnectedSpaceTasteGate single carrier alignment`

Theorem 6885.2 (PathConnectedSpace NameCert obligations)**.** *For every accepted* $\mathsf{PathConnectedSpace}^{\uparrow}$ *carrier*

$$K = (E, F, V, C, G, L, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PathConnectedSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the endpoint-pair, displayed path-family, endpoint-evaluation, continuity replay, connectedness consumer, locally connected comparison, transport, replay, provenance, and naming rows; endpoint exactness through V; path-family stability through F and H; continuous-image handoff through C; connectedness consumer routing through G; and ledger nonescape from host path spaces, quotient path components, selected paths, excluded middle, open-cover compactness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 6885.1. The projection exposes exactly the rows $E, F, V, C, G, L, H, R, P, N$, so carrier admission is exhausted by those coordinates.

Endpoint exactness is read from V. The displayed path in F is not available to a consumer until V has tied its left and right endpoint evaluations back to the ordered endpoint pair in E.

Path-family stability is transported componentwise through H. The same endpoint pair, path-family name, endpoint ledger, continuity replay, connectedness route, and local comparison row are then replayed by R and sourced by P, N.

Finally, continuous-image and connectedness consumers can inspect only C and G. Since no carrier row stores a host path space, quotient component, selected path operator, excluded-middle split, or open-cover compactness principle, none of those objects can escape the local ledger. □

Theorem 6885.3 (PathConnectedSpace continuous image route)**.** *Every continuous-image consumer of an accepted* $\mathsf{PathConnectedSpace}^{\uparrow}$ *carrier*

$$K = (E, F, V, C, G, L, H, R, P, N)$$

reads the displayed path route E, F, V, C before any $\mathsf{ConnectedSpace}^{\uparrow}$ *consumer row or* $\mathsf{IntervalConnectedness}^{\uparrow}$ *interval consumer may use the packet. The exported image route is therefore a finite continuity-and-endpoint replay, not a host-space assertion that arbitrary path components have been selected.*

Proof. Start with a continuous-image request. Carrier admission exposes the endpoint row E and the displayed path-family row F before the request reaches any connectedness consumer.

The endpoint-evaluation ledger V is then forced into the route. It binds the left and right evaluations of the displayed path to the endpoint pair already named in E.

The continuity replay row C is read next. Its inverse-image open rows and finite continuity evidence are the only data made available to an image consumer before connectedness handoff.

Only after E, F, V, C have been read may the consumer pass through G or through an interval-connectedness handoff named in P. Thus $\mathsf{ConnectedSpace}^{\uparrow}$ and $\mathsf{IntervalConnectedness}^{\uparrow}$ consume the displayed path route; they do not create a separate selected path space or quotient path-component surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PathConnectedSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PathConnectedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6885.1, Definition 6885.2, and Definition 6885.3 seed the finite endpoint-pair, displayed-path, continuity, connectedness-consumer, and local-comparison route over topology and interval-connectedness neighbours.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PathConnectedSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PathConnectedSpaceUp.PathConnectedSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient path components, selected paths, excluded middle, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, path-family stability, continuous-image handoff, connectedness consumer routing, and ledger nonescape for BEDC.Derived.PathConnectedSpaceUp.

Constructive story: A $\mathsf{PathConnectedSpace}^{\uparrow}$ packet is generated from endpoint pairs, displayed path families, endpoint-evaluation ledgers, Continuous replay rows, ConnectedSpace consumer routes, LocallyConnected comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Chapter 6886

A Concrete Naming Certificate for $\mathsf{TubularNeighborhood}^{\uparrow}$

Following Whitney and Hirsch on tubular neighbourhoods, the $\mathsf{TubularNeighborhood}^{\uparrow}$ packet records the finite naming surface for a submanifold, its normal bundle, a metric-controlled exponential chart, and the resulting neighbourhood image. It reads the source manifold through chapter 96 and the smooth atlas boundary through chapter 4291; it reads the bundle row through chapter 97 and the vector-bundle carrier of chapter 210; it reads the metric and exponential-chart rows through chapter 133 and chapter 142; and it reads radius and image budgets through the Real seal of chapter 38. The certificate names the displayed finite tubular route without asserting a global collar, selected closest-point projection, quotient equality of normal vectors, or ambient completeness.

Definition 6886.1 (Tubular neighborhood carrier)**.** *A* $\mathsf{TubularNeighborhood}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, N, G, E, R, U, H, C, P, L)$$

with the following displayed rows:

1. *S is the submanifold and source-manifold row, read through* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{SmoothManifold}^{\uparrow}$*, including the displayed inclusion data and the finite atlas windows on which the construction is inspected.*
2. *N is the normal-bundle row, read through* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{VectorBundle}^{\uparrow}$*, carrying the displayed fibre complement and transition compatibility data for the source row.*
3. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row, exposing the metric, orthogonality, and fibre-length comparisons used to choose the visible normal window.*
4. *E is the* $\mathsf{ExpMap}^{\uparrow}$ *row, recording the finite exponential-chart request from the normal-bundle window into the ambient manifold chart.*
5. *R is the radius ledger, a* $\mathsf{Real}^{\uparrow}$ *tolerance row that bounds the visible normal vectors and records the local injectivity budget used by the displayed chart.*
6. *U is the neighbourhood image row, naming the visible image of the radius-bounded normal vectors under the exponential-chart row.*
7. *H records componentwise* hsame *transport for $S, N, G, E, R, U, C, P, L$.*

8. *C records displayed* Cont *replay from submanifold reads through the normal-bundle, metric, exponential-chart, radius, and image rows.*
9. *P is the* Pkg *provenance over* $\mathsf{TubularNeighborhood}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{ExpMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{TubularNeighborhood}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a global retraction, selected nearest point, quotient normal-vector equality, trichotomy of distances, or ambient manifold completeness.

Lean4 (statement) `BEDC.Derived.TubularNeighborhoodUp.TubularNeighborhoodUp`

Theorem 6886.2 (Tubular neighborhood NameCert obligations)**.** *For every accepted* $\mathsf{TubularNeighborhood}^{\uparrow}$ *carrier $T = (S, N, G, E, R, U, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{TubularNeighborhood}^{\uparrow}}$ *surface consists exactly of carrier admission, submanifold and source-manifold visibility, normal-bundle attachment, Riemannian metric readback, exponential-chart replay, radius-ledger boundedness, neighbourhood-image exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6886.1. The source row S supplies the displayed manifold and submanifold windows, while N attaches the normal-bundle data to those same windows.

Read the analytic rows next. The metric row G supplies the orthogonality and fibre-length comparisons, the exponential-chart row E records the visible chart request, and the radius row R bounds the normal vectors before the image row U is exposed.

The structural rows H, C, P, L transport, replay, source, and name the same finite packet. Since the carrier has no row for a global retraction, selected nearest point, quotient normal-vector equality, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6886.3 (Tubular neighborhood normal-bundle handoff)**.** *Every* $\mathsf{TubularNeighborhood}^{\uparrow}$ *consumer of the neighbourhood image row must first read the finite handoff*

$$\mathsf{Manifold}^{\uparrow}, \mathsf{Bundle}^{\uparrow}, \mathsf{RiemannianMetric}^{\uparrow}, \mathsf{ExpMap}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto S, N, G, E, R, U, H, C, P, L.$$

In particular, the normal-bundle row and metric row are visible before the exponential-chart image is consumed, and the radius ledger is visible before the image is treated as a neighbourhood.

Proof. Start from the carrier rows $S, N, G, E, R, U, H, C, P, L$. The source manifold row and normal-bundle row determine which fibre windows are eligible for the tubular request.

The metric row then fixes the finite orthogonality and length comparisons used by the request. Only after those comparisons are displayed can the exponential-chart row E replay the normal vector into the ambient manifold chart.

The radius ledger R precedes the image row U, so a consumer cannot treat U as a neighbourhood without also reading the bounded normal-window evidence. Transport and replay through H and C, together with provenance P and local naming L, keep the handoff attached to the same packet.

Thus the handoff is a finite normal-bundle route through the cited manifold, bundle, metric, exponential-map, and Real rows. It does not export a selected nearest point, global retraction, quotient normal-vector equality, trichotomy of distances, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{TubularNeighborhood}^{\uparrow}$ read is refuted if a consumer obtains the neighbourhood image row without first displaying the source-manifold row, normal-bundle row, Riemannian metric row, exponential-chart row, radius ledger, transport, replay, provenance, and local NameCert rows.

Independence witness: manifold, bundle, riemannianmetric, expmap, real: $\mathsf{TubularNeighborhood}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source manifold data, normal-bundle attachment, metric comparison, exponential-chart replay, radius boundedness, image exposure, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TubularNeighborhood}^{\uparrow}}$.

Constructive story: A $\mathsf{TubularNeighborhood}^{\uparrow}$ packet is generated from submanifold and source-manifold rows, normal-bundle rows, Riemannian metric rows, exponential-chart rows, Real radius ledgers, neighbourhood image rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TubularNeighborhood}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6886.1, Definition 6886.2, and Definition 6886.3 seed the finite normal-bundle and exponential-chart route for tubular neighbourhood consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TubularNeighborhood}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close selected nearest-point projections, global retractions, quotient equality of normal vectors, trichotomy of distances, ambient manifold completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, normal-bundle attachment, metric-radius exactness, exponential-chart replay, image nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TubularNeighborhoodUp.

Chapter 6887

A Concrete Naming Certificate for KolmogorovThreeSeriesBoundary$^{\uparrow}$

Following Kolmogorov and Feller on convergence of independent random series, the KolmogorovThreeSeriesBoundary packet records the finite tail-boundary data that a BEDC probability consumer may inspect before asking for any classical three-series conclusion. It sits after the probability substrate of chapter 184, the measurable-preimage surface of chapter 185, the pushforward law row of chapter 186, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite tail ledger for truncated random-variable reads; it does not assert almost-sure convergence, close the full three-series theorem, invoke countable choice, identify quotient distribution equality, take host probability limits, or import ambient Real$^{\uparrow}$ completeness.

Definition 6887.1 (Kolmogorov three-series boundary carrier)**.** *A* KolmogorovThreeSeriesBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\Omega, X, D, T, V, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* ProbSpace$^{\uparrow}$ *source row carrying the event algebra, probability measure, and total-mass endpoint.*
2. X *is the finite* RandomVar$^{\uparrow}$ *family row listing the random variables whose tails are currently exposed.*
3. D *is the* Distribution$^{\uparrow}$ *pushforward row for the displayed variables and their truncation events.*
4. T *is the finite tail-threshold row, naming the truncation windows and tail events inspected by the packet.*
5. V *is the truncated-variance ledger row, recording only the displayed second-moment windows attached to* T*.*
6. M *is the truncated-mean ledger row, recording only the displayed expectation windows attached to* T*.*
7. R *is the* RegSeqRat$^{\uparrow}$ *readback row for the finite probability, mean, and variance windows.*
8. E *is the terminal* Real$^{\uparrow}$ *seal row reached after the regular rational readbacks are displayed.*
9. H *records componentwise* hsame *transport for probability, random-variable, distribution, threshold, mean, variance, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tail-boundary request through the finite threshold, readback, and seal rows.*
11. *P is the* Pkg *provenance over* $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for almost-sure convergence, the full three-series theorem, countable choice, quotient equality of distributions, host probability limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6887.2 (Kolmogorov three-series boundary NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$ *carrier*

$$K = (\Omega, X, D, T, V, M, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed probability source, finite random-variable family, pushforward distribution rows, truncation-threshold ledger, truncated variance ledger, truncated mean ledger, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming; threshold stability requiring T, V, M to remain attached to the same displayed variables and distributions under* hsame*; readback exactness requiring R, E to be consumed only after the finite tail windows have been displayed; and ledger refusal excluding almost-sure convergence, the full three-series theorem, countable choice, quotient distribution equality, host probability limits, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the accepted packet.*

Proof. Project the carrier of Definition 6887.1. The source probability row, finite random-variable family, pushforward distribution rows, truncation thresholds, truncated variance rows, truncated mean rows, readback, seal, transport, replay, provenance, and local name are separate finite rows of the same BHist packet.

Carrier admission reads Ω, X, D, T, V, M before any Real-facing endpoint is exposed. The sibling chapters named above supply only displayed probability, measurable-preimage, pushforward, regular-readback, and Real-seal rows, so the admission route remains finite.

Threshold stability is componentwise. Transport through H preserves the same variable family, distribution row, threshold window, mean ledger, and variance ledger before R repacks the visible numbers as $\mathsf{RegSeqRat}^{\uparrow}$ data.

Finally C, P, N replay, source, and name the same route. Since no row supplies convergence, a completed three-series theorem, countable choice, quotient distribution equality, host probability limits, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6887.3 (Kolmogorov three-series boundary finite tail route)**.** *Every probability-limit consumer of an accepted* $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ProbSpace}^{\uparrow} \longmapsto \mathsf{RandomVar}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto T, V, M \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before treating a tail-boundary endpoint as public. The route exports exactly the displayed probability source, random-variable family, pushforward law, truncation thresholds, truncated mean and variance ledgers, regular rational readback, terminal Real seal, componentwise transport, continuation replay, provenance, and local naming; it exports no almost-sure convergence theorem, no full three-series theorem closure, no countable-choice schedule, no quotient distribution equality, no host probability limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations above. They force every accepted read through $\Omega, X, D, T, V, M, R, E, H, C, P, N$ before a consumer may inspect a Real-facing tail endpoint.

Use the sibling dependency route. The $\mathsf{ProbSpace}^{\uparrow}$ row supplies the source measure, $\mathsf{RandomVar}^{\uparrow}$ supplies measurable preimages for the displayed variables, $\mathsf{Distribution}^{\uparrow}$ supplies the pushforward law row, $\mathsf{RegSeqRat}^{\uparrow}$ supplies finite numerical readback, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal.

The finite tail route is therefore just row projection and dependency composition. A consumer can pass from the probability source to the Real-facing endpoint only through the displayed truncation, mean, and variance ledgers, and no carrier coordinate remains for the refused convergence, choice, quotient, host-limit, or completeness objects. □

Falsifiable prediction: Every accepted $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$ read displays a ProbSpace source row, finite RandomVar family row, Distribution pushforward row, finite tail-threshold row, truncated variance and mean ledgers, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: probspace, randomvar, distribution, regseqrat, real: $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$ is not bijective to any listed sibling because it binds probability source data, finite random-variable tails, pushforward distribution rows, truncation thresholds, truncated mean and variance ledgers, regular rational readback, and a terminal Real seal into one finite tail-boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6887.1, Definition 6887.2, and Definition 6887.3 seed the finite probability-source, random-variable, distribution, truncation-threshold, truncated-mean, truncated-variance, readback, Real-seal, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the full three-series theorem, countable choice, quotient distribution equality, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold stability, truncated mean and variance readback, finite tail-route exactness, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovThreeSeriesBou

Constructive story: A $\mathsf{KolmogorovThreeSeriesBoundary}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family row, pushforward distribution row, finite tail-threshold row, truncated variance and mean ledgers, regular rational readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6888

A Concrete Naming Certificate for $\mathsf{AumannIntegralBoundary}^{\uparrow}$

Following Aumann on integrals of set-valued maps, the $\mathsf{AumannIntegralBoundary}^{\uparrow}$ packet records the finite NameCert boundary at which a set-valued integral can be read without importing a host measurable-selection principle. It sits after the measurable ledger of chapter 92, the scalar readback surface of chapter 123, the convex-value carrier of chapter 208, the observable random input surface of chapter 185, and the target normed carrier of chapter 90. The packet names a finite boundary for downstream probability and optimization readers: it exposes displayed selection rows and their integrals, but it does not claim a global measurable-selection theorem, quotient equality of selections, countable choice, or ambient measure completion.

Definition 6888.1 (Aumann integral boundary carrier)**.** *An* $\mathsf{AumannIntegralBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, R, X, V, G, S, K, L, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row whose measurable-event ledger bounds every domain-side read.*
2. *R is the* $\mathsf{RandomVar}^{\uparrow}$ *observation row carrying the displayed random input, measurable preimage tests, and observable classifier.*
3. *X is the source measurable-space row on which the set-valued map is inspected.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *value row, recording nonempty convex fibres and their finite membership tests.*
5. *G is the measurable graph row, displaying the finite graph tests that relate source observations in X, R to value candidates in V.*
6. *S is the displayed finite selection ledger. Its entries are finite rows*

   $$(s_i, \gamma_i, \mu_i, \iota_i)_{i \in I}$$

   where each s_i is a named selection candidate, γ_i is its graph-admission witness through G, μ_i is its measurable-read witness through M, R, X, and ι_i is its integral-read token.
7. *K is the convex-value stability row, recording that every displayed selection value remains inside the named convex fibre of V and that finite convex combinations preserve the same value boundary.*

8. *L is the integral readback row, reducing each ι_i to the finite scalar or vector integral data already accepted by* $\mathsf{Integral}^{\uparrow}$.
9. *B is the* $\mathsf{Banach}^{\uparrow}$ *target row in which the displayed integral endpoints and norm comparisons are read.*
10. *H records componentwise* hsame *transport for $M, R, X, V, G, S, K, L, B$ and the structural rows.*
11. *C records displayed* Cont *replay of the route*

$$M, R, X, V, G \longmapsto S \longmapsto K, L \longmapsto B.$$

12. *P is the* Pkg *provenance over* $\mathsf{AumannIntegralBoundary}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AumannIntegralBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed measure, random-variable, source, convex-value, measurable-graph, finite-selection, convex-stability, integral-readback, Banach-target, transport, replay, provenance, and local naming rows.

Theorem 6888.2 (Aumann integral boundary NameCert obligations)**.** *For every accepted* AumannIntegralBoundary *carrier*

$$A = (M, R, X, V, G, S, K, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AumannIntegralBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, random-variable row, source row, convex-value row, measurable graph row, displayed finite selection ledger, convex-value stability row, integral readback row, Banach target row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N. Every public set-valued integral read must pass through G, S, K, L, B before it names an endpoint or boundary verdict.*

Proof. Project the carrier of Definition 6888.1. The dependency rows M, R, X, V provide the measure, random-variable, source, and convex-value data on which the boundary is interpreted.

The theorem-facing rows are G, S, K, L, B. The graph row G admits only displayed graph tests, the finite ledger S lists the selection candidates that are actually inspected, the convex row K keeps their values inside the displayed fibres, the integral row L reads their accepted integrals, and the Banach row B names the endpoint and norm comparisons.

Transport and replay do not add analytic content. The rows H, C, P, N preserve, replay, source, and name exactly the same finite route, so the local certificate has no projection to a global measurable-selection theorem, quotient equality of selections, countable choice, or ambient measure completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{AumannIntegralBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{AumannIntegralBoundary}^{\uparrow}$ packet is generated from displayed measure, random-variable, source, convex-value, measurable-graph, finite-selection, convex-stability, integral-readback, Banach-target, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AumannIntegralBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 6888.1 and Definition 6888.2 seed the finite set-valued integral boundary through displayed selection ledgers and Banach integral readbacks.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AumannIntegralBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a global measurable-selection theorem, quotient equality of selections, countable choice, ambient measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, measurable graph exactness, finite selection ledger exactness, convex-value stability, integral readback, Banach target preservation, transport stability, and replay stability for BEDC.Derived.AumannIntegralBoundaryUp.

Chapter 6889

A Concrete Naming Certificate for $\mathsf{CompactRegularRealFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompactRegularRealFunction}^{\uparrow}$ packet names a real-valued function on a displayed compact metric source by a finite regular approximation surface rather than by a host function quotient. It sits after the compact metric carrier of chapter 129, the continuous-map carrier of chapter 130, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the locally convex functional-analysis source route of chapter 3386. The packet gives compact/continuous and completion-facing routes a finite real-function carrier; it does not assert open-cover compactness, choice of covers, quotient real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6889.1 (Compact regular real-function carrier)**.** *A* $\mathsf{CompactRegularRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, C, W, R, E, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *and naming only the compact carrier data visible to chapter 129.*
2. *C is the continuous-map row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and exposing the domain, codomain, continuity witness, and modulus-facing data used by the packet.*
3. *W is the finite stream-window row, read through* $\mathsf{StreamName}^{\uparrow}$ *and recording the inspected source windows and output-name windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *output readback row for the regular rational approximants attached to those windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W and R have displayed the regular output name.*
6. *L is the local regularity ledger, recording the compact-window tolerance request, the displayed regular schedule, and the refusal to choose an open-cover subcover or a quotient representative of the function.*
7. *H records componentwise* hsame *transport for K, C, W, R, E, L and the structural rows.*
8. *Q records displayed* Cont *replay from compact-window requests through the stream windows, regular readbacks, Real seal, and local regularity ledger.*

9. *P records* Pkg *provenance over* $\mathsf{CompactRegularRealFunction}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRegularRealFunction}^{\uparrow}}$ *row.*

The classifier compares packets only through $K, C, W, R, E, L, H, Q, P, N$. It has no coordinate for open-cover compactness, selected finite subcovers, quotient equality of real functions, countable choice, a hidden function-space completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6889.2 (Compact regular real-function NameCert obligations)**.** *For every accepted* $\mathsf{CompactRegularRealFunction}^{\uparrow}$ *carrier $F = (K, C, W, R, E, L, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactRegularRealFunction}^{\uparrow}}$ *surface consists exactly of compact metric source exposure, continuous-map exposure, stream-window exposure, regular rational output readback, Real seal routing, local regularity-ledger visibility, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6889.1. The compact source row K and continuous-map row C determine the displayed domain and continuity surface, while W, R, E are the only path from a requested output window to a sealed real name.

The local regularity ledger L records the compact-window tolerance schedule inspected by the packet. It is a displayed finite row, so it cannot choose an open cover, select a hidden subcover, or replace regular output names by quotient real equality.

Transport, replay, provenance, and local naming through H, Q, P, N preserve the same rows. Since no other coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 6889.3 (Compact regular real-function Real-seal handoff)**.** *Every Real-facing read from an accepted* $\mathsf{CompactRegularRealFunction}^{\uparrow}$ *packet must pass through the stream-window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *output readback row R, and the* $\mathsf{Real}^{\uparrow}$ *seal row E before it is visible to compact/continuous or functional-analysis consumers. The handoff exports only the sealed output route for displayed compact windows; it does not assert* $\mathsf{Real}^{\uparrow}$ *completeness or quotient equality of real-valued functions.*

Proof. The carrier presents W, R, E in that order. A consumer asking for real output data first receives the finite stream windows named by $\mathsf{StreamName}^{\uparrow}$, then the regular rational readback named by $\mathsf{RegSeqRat}^{\uparrow}$, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal.

The compact metric and continuous-map rows K, C identify the source of the request, but they do not supply a real output by themselves. The regularity ledger L records the displayed schedule connecting the compact window to the output route.

Thus every Real-facing handoff is mediated by W, R, E, L and then preserved through H, Q, P, N. No row supplies ambient completeness of $\mathsf{Real}^{\uparrow}$, a selected limit, or quotient equality of functions. □

Falsifiable prediction: Every accepted $\mathsf{CompactRegularRealFunction}^{\uparrow}$ read exposes compact metric source data, continuous-map data, stream windows, RegSeqRat output readback, RealUp seal, local regularity ledger, transport, replay, provenance, and local naming before any compact real-function value is public.

Independence witness: compactmetric, continuousmap, streamname, regseqrat, real, locally-convex-space: these siblings carry compact source data, continuous-map data, stream windows, regular rational readbacks, Real seals, or locally convex functional-analysis routes, but none carries the combined compact source, regular real-output, local regularity, transport, replay, provenance, and local NameCert surface of $\mathsf{CompactRegularRealFunction}^{\uparrow}$.

The Lean follow-up in `lean4/BEDC/Derived/CompactRegularRealFunctionUp.lean` should expose `BEDC.Derived.CompactRegularRealFunctionUp` without adding open-cover compactness, countable choice, quotient real equality, selected function representatives, or ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRegularRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRegularRealFunction}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map rows, stream windows, regular rational output readbacks, a RealUp seal, a local regularity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRegularRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6889.1, Definition 6889.2, and Definition 6889.3 seed the finite compact regular real-function route over compact source, continuous-map, stream-window, regular-readback, Real-seal, regularity-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRegularRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, quotient equality of real-valued functions, countable choice, Lean verification, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-map exposure, stream-window exposure, regular-rational readback, Real-seal routing, local regularity-ledger stability, transport, replay, provenance, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.CompactRegularRealFunctionUp.

Chapter 6890

A Concrete Naming Certificate for $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$

The $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$ packet realizes the decidability frontier of chapter 9386 as a finite closed-term fragment rather than as an unrestricted conversion theorem. It reads the positive checker surface of chapter 9379, the candidate-mediated normalization route of chapter 9383, the typed-normalization audit of chapter 3539, the typed-examples gallery of chapter 775, the bounded normal-conversion bridge of chapter 3439, and the typed-conversion residual frontier of chapter 6069. The packet keeps closed-term typing, bounded-normal endpoints, finished-normal witnesses, and obstruction rows visible before any local decidability read is accepted.

Definition 6890.1 (MetaCIC closed typing fragment carrier)**.** *A* $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, J, B, F, O, X, R, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed-term row, listing the finite closed MetaCIC terms and contexts inspected by the packet.*
2. *J is the type-checker row, recording the option-valued typing or rejection judgement accepted at the decidability audit interface.*
3. *B is the bounded-normal endpoint row, exposing the bounded normalizer output and fuel boundary used by the local check.*
4. *F is the finished-normal witness row, present only when the bounded endpoint has been sealed as a finished closed normal form.*
5. *O is the obstruction row, retaining missing closedness evidence, missing finished-normal evidence, subject-reduction boundaries, confluence residuals, and conversion-frontier failures.*
6. *X is the transport row from the decidability audit map, normalization audit map, typed-normalization packet, typed-examples gallery, bounded bridge, and residual frontier into the local fragment.*
7. *R is the replay row, recording bounded re-checks of T, J, B, F, O without claiming an unrestricted closed-term beta-conversion decider.*
8. *S is the provenance row separating the checker, normalization, gallery, endpoint-bridge, residual-frontier, and open-problem sources.*

9. *H records componentwise* hsame *transport for T, J, B, F, O, X, R, S and the structural rows.*
10. *C records displayed* Cont *replay from a local closed-typing request through the checker, endpoint, obstruction, transport, replay, provenance, and naming rows.*
11. *P records* Pkg *provenance over* $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$, $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicClosedTypingFragment}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, J, B, F, O, X, R, S, H, C, P, N$. It has no coordinate for a global closedness decider, unrestricted beta-conversion decidability, unconditional normalization, unconditional confluence, erased subject-reduction discharge, quotient syntax equality, or host consistency.

Theorem 6890.2 (MetaCIC closed typing fragment NameCert obligations)**.** *For every accepted* $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$ *carrier $M = (T, J, B, F, O, X, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetacicClosedTypingF}}$ *surface consists exactly of closed-term exposure, type-checker exposure, bounded-normal endpoint exposure, finished-normal witness exposure when present, obstruction retention, audit-map transport, bounded replay, provenance separation, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6890.1. The rows T and J are the closed-term and checker-facing surface, while B and F name the bounded endpoint and finished-normal evidence that the decidability frontier requires.

The obstruction row O is retained in the same packet. It records when closedness, finished-normal evidence, subject reduction, confluence residuals, or conversion endpoints are not available, so the checker rows cannot be read as a global conversion theorem.

Rows X, R, S, H, C, P, N transport, replay, source, preserve, and name the same finite surface. Since no coordinate stores global closedness decision, unrestricted beta-conversion, unconditional normalization, unconditional confluence, quotient syntax equality, or host consistency, the listed rows exhaust the NameCert obligations. □

Theorem 6890.3 (MetaCIC closed typing fragment decidable boundary)**.** *Every accepted* MetacicClosedTypingFrag *read separates the local closed-typing checker result from unrestricted closed-term beta-conversion decidability. A local positive typing row may be consumed only together with its bounded-normal endpoint, finished-normal witness status, obstruction row, transport, replay, provenance, and local naming rows.*

Proof. The checker row J is local to the displayed closed-term row T. It does not contain a theorem deciding closedness for arbitrary terms or beta-conversion for all closed typed inputs.

The endpoint rows B, F show exactly where bounded normal computation has reached a finished normal form. If that evidence is absent or blocked, the obstruction row O remains visible and prevents promotion to an unrestricted conversion result.

Transport and replay through X, R, H, C, P, N preserve this boundary. Therefore the packet can expose a finite decidability fragment while refusing global closedness, conversion, normalization, confluence, subject-reduction, quotient-syntax, or consistency claims. □

Falsifiable prediction: Every accepted $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$ read exposes closed-term typing, bounded-normal endpoint, finished-normal witness, obstruction, transport, replay, provenance, and local naming rows before any decidability claim.

Independence witness: metacic-open-problem-ledger, bounded-normal-conversion-bridge, metacic-conversion-endpoint-seal, metacic-typed-conversion-residual-frontier: these routes carry the open ledger, endpoint bridge, endpoint seal, or residual frontier, but none carries the combined closed-term, checker, bounded endpoint, finished-normal, obstruction, transport, replay, provenance, and local NameCert surface of $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$.

The Lean follow-up in `lean4/BEDC/Derived/MetacicClosedTypingFragmentUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ MetacicClosedTypingFragment}^{\uparrow}$ and $\mathsf{FieldFaithful\ MetacicClosedTypingFragment}^{\uparrow}$ instances preserving $T, J, B, F, O, X, R, S, H, C, P, N$ before any positive gate may treat this packet as field-faithful.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicClosedTypingFragment}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicClosedTypingFragment}^{\uparrow}$ packet is generated from finite closed-term rows, checker rows, bounded-normal endpoints, finished-normal witnesses, obstruction rows, audit-map transport, bounded replay, provenance separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicClosedTypingFragment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6890.1, Definition 6890.2, and Definition 6890.3 seed the finite closed-typing fragment route over closed terms, checker output, bounded endpoints, finished-normal witnesses, obstructions, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicClosedTypingFragment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global closedness decider, unrestricted closed-term beta-conversion decidability, unconditional normalization, unconditional confluence, erased subject-reduction discharge, quotient syntax equality, Lean verification, host consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-term exposure, checker exposure, bounded-normal endpoint exposure, finished-normal witness stability, obstruction retention, transport, replay, provenance, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.MetacicClosedTypingFragmentUp.

Chapter 6891

A Concrete Naming Certificate for $\mathsf{HitMissConvergence}^{\uparrow}$

Following Kelley and Kuratowski on hit–miss hyperspace convergence, the $\mathsf{HitMissConvergence}^{\uparrow}$ packet records the finite BEDC surface by which a hyperspace or cluster-set consumer reads hit and miss convergence data without importing an ambient host hyperspace topology. It sits between the hit–miss classifier of chapter 1660, the Vietoris route of chapter 2375, the Kuratowski convergence consumer of chapter 6591, the compact-metric support of chapter 129, the metric source of chapter 57, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence interface only; it is not a selected closed-set limit, a quotient hyperspace equality, a host Vietoris topology, or a compactness theorem.

Definition 6891.1 (Hit–miss convergence carrier)**.** *A* $\mathsf{HitMissConvergence}^{\uparrow}$ *carrier is a tuple*

$$H = (L, M, V, K, D, R, E, T, C, P, N)$$

with the following finite rows.

1. *L is the finite hit ledger, listing the displayed source cells, target neighbourhood probes, and witnessed hits used by the convergence read.*
2. *M is the finite miss ledger, listing the displayed closed or compact avoidance probes and their witnessed exclusions.*
3. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *route through which hit and miss rows are interpreted before any hyperspace-facing consumer can read them.*
4. *K is the* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *route that consumes the same hit and miss rows as finite convergence evidence.*
5. *D records compact and metric support from* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$*, including the finite ball or compact-source names on which the hit and miss ledgers are checked.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row for displayed radius and stage bounds.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for real-valued tolerance endpoints reached only after R has been displayed.*
8. *T is componentwise* hsame *transport through L, M, V, K, D, R, E.*
9. *C is displayed* Cont *replay for the same finite route.*

10. *P is* Pkg *provenance over* $\mathsf{HitMissConvergence}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{KuratowskiConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HitMissConvergence}^{\uparrow}}$ *row naming exactly this carrier surface.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host powerset, selected closed-set limit, quotient equality of hyperspace points, open-cover compactness object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 (statement) `BEDC.Derived.HitMissConvergenceUp`

Theorem 6891.2 (Hit–miss convergence NameCert obligations)**.** *For every accepted* $\mathsf{HitMissConvergence}^{\uparrow}$ *carrier $H = (L, M, V, K, D, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HitMissConvergence}^{\uparrow}}$ *surface consists exactly of finite hit-ledger admission through L; finite miss-ledger admission through M; Vietoris routing through V; Kuratowski convergence routing through K; compact and metric support through D; regular-rational tolerance readback through R; real tolerance sealing through E; componentwise* hsame *transport through T; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6891.1. The rows $L, M, V, K, D, R, E, T, C, P, N$ are exactly the hit, miss, Vietoris, Kuratowski, support, tolerance, transport, replay, provenance, and naming coordinates.

The hit and miss ledgers are finite before either route can be read. The Vietoris route V supplies the hyperspace-facing interpretation, while the Kuratowski route K consumes the same finite evidence as convergence data.

The support and tolerance rows D, R, E keep the route inside compact-metric, metric, regular-rational, and real-seal data already displayed in the packet. Transport, replay, provenance, and local naming preserve that same finite route, so the listed rows exhaust the local certificate surface. □

Theorem 6891.3 (Hit–miss convergence Kuratowski route)**.** *Every* $\mathsf{KuratowskiConvergence}^{\uparrow}$ *or Kuratowski cluster-set consumer that reads an accepted* $\mathsf{HitMissConvergence}^{\uparrow}$ *packet must first expose the finite hit ledger L, the finite miss ledger M, the* $\mathsf{VietorisTopology}^{\uparrow}$ *route V, the compact and metric support row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback R, the* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows T, C, P, N. The consumer receives a finite hit–miss convergence route and not a host hyperspace topology, selected closed-set limit, quotient equality, open-cover compactness proof, or ambient real completeness principle.*

Proof. Begin with Definition 6891.2. It fixes the accepted read to the displayed finite rows and leaves no coordinate for a host hyperspace object or selected limit.

The Kuratowski route K can consume hit–miss data only after L and M have been displayed and interpreted through V. The compact-metric and metric support row D supplies the named source cells, while R and E seal the tolerance endpoints.

Finally the structural rows T, C, P, N transport, replay, source, and name the same route. Thus every Kuratowski-facing read factors through the finite hit–miss interface and cannot export any of the refused host, quotient, compactness, or completeness endpoints. □

Falsifiable prediction: Every accepted $\mathsf{HitMissConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{KuratowskiConvergence}^{\uparrow}$ read without first exposing the finite hit ledger, finite miss ledger, Vietoris route, compact/metric support, tolerance readback, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: hyperspace, vietoris, kuratowski, compactmetric, metric, regseqrat, real: none is bijective with $\mathsf{HitMissConvergence}^{\uparrow}$ because this carrier alone binds finite hit and miss ledgers to the Kuratowski-facing convergence route while retaining the support, tolerance, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HitMissConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{HitMissConvergence}^{\uparrow}$ packet is generated from finite hit ledgers, finite miss ledgers, a Vietoris route, a Kuratowski convergence route, compact and metric support, RegSeqRat tolerance readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HitMissConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6891.1, Definition 6891.2, and Definition 6891.3 seed the finite hit–miss convergence interface for hyperspace and Kuratowski consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HitMissConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected closed-set limits, quotient hyperspace equality, host Vietoris topology, open-cover compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite hit and miss ledger exactness, Vietoris route preservation, Kuratowski route preservation, compact/metric support exactness, RegSeqRat and Real tolerance sealing, transport, replay, provenance, and local naming for BEDC.Derived.HitMissConvergenceUp.

Chapter 6892

A Concrete Naming Certificate for $\mathsf{IntervalBisectionSearch}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{IntervalBisectionSearch}^{\uparrow}$ packet records the finite bisection-search surface for a located real interval whose endpoint signs, apartness data, dyadic midpoint rows, and shrinking brackets are inspected before a regular Cauchy readback or terminal real seal is consumed. It sits beside the dyadic rational carrier of chapter 291, the finite stream-window source of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located endpoint ledger of chapter 394. The packet names constructive interval bisection search as displayed BEDC rows; it is not host real trichotomy, not a selected root, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6892.1 (Interval bisection search carrier)**.** *An* $\mathsf{IntervalBisectionSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, A, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing located lower and upper endpoint names, endpoint comparison budgets, and interval membership rows.*
2. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint row, recording the dyadic midpoint request, its endpoint placement witnesses, and the finite tolerance ledger used to split the current bracket.*
3. *A is the sign and apartness witness row. It records the finite sign-read or apartness-read data used to retain one child bracket without invoking real trichotomy.*
4. *L is the shrinking bracket ledger, listing the retained located interval at each inspected bisection stage together with the displayed diameter or tolerance decrease row.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row obtained from the retained bracket endpoints at the requested precisions.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the midpoint or retained-endpoint schedule selected by the finite ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, M, A, L, S, R have been displayed.*
8. *H stores componentwise* hsame *transport for located endpoints, midpoint rows, sign or apartness witnesses, bracket ledgers, stream windows, regular readback, Real seal, replay, provenance, and local naming.*

9. *C stores displayed* Cont *replay of the bisection-search request through* $I, M, A, L, S, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{IntervalBisectionSearch}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalBisectionSearch}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite bisection-search history. It has no coordinate for a host root selector, unrestricted sign trichotomy, quotient equality of real names, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6892.2 (Interval bisection search NameCert obligations)**.** *For every accepted* $\mathsf{IntervalBisectionSearch}^{\uparrow}$ *carrier* $B = (I, M, A, L, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalBisectionSearch}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located interval row, dyadic midpoint row, sign or apartness witness row, shrinking bracket ledger, stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming;*
2. *classifier stability under componentwise* hsame *transport of* I, M, A, L, S, R, E*;*
3. *midpoint exactness, requiring every retained child bracket to cite the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint row before it is inspected;*
4. *sign and apartness locality, requiring every branch decision to be read from A and not from host trichotomy of real values;*
5. *bracket shrinkage, requiring the ledger L to expose a smaller located interval or tighter tolerance row at each accepted stage;*
6. $\mathsf{StreamName}^{\uparrow}$ *window exposure using only retained endpoint windows from L;*
7. $\mathsf{RegSeqRat}^{\uparrow}$ *readback from the same retained windows or midpoint schedule;*
8. $\mathsf{Real}^{\uparrow}$ *seal nonescape through* S, R, E*;*
9. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
10. *ledger refusal for any consumer that omits a located endpoint, midpoint, sign or apartness, shrinking bracket, stream window, regular readback, Real seal, transport, replay, provenance, or local naming row.*

Proof. Project the carrier of Definition 6892.1. The visible rows are exactly $I, M, A, L, S, R, E, H, C, P, N$, and the packet has no hidden coordinate for a host root selector, real trichotomy, quotient equality, selected limit, or completeness theorem.

The classifier is rowwise. Componentwise hsame transport preserves the located interval, midpoint, sign or apartness, bracket ledger, stream, readback, and seal rows together, so a transported bisection-search read remains inside the same finite route.

Midpoint exactness and branch locality are read from M and A. The midpoint row is the only split source, and the sign or apartness row is the only branch-retention source, so the carrier cannot replace them by a host comparison of real values.

The readback obligations follow from L, S, R, E. The shrinking bracket ledger fixes the retained windows, the stream row exposes finite endpoint observations, the regular rational row repacks the same observations, and the Real seal is terminal. Repacking with H, C, P, N gives the stated local NameCert surface. □

Theorem 6892.3 (Interval bisection search L10 handoff)**.** *Every Real-and-Completion consumer reading an accepted* $\mathsf{IntervalBisectionSearch}^{\uparrow}$ *carrier must pass through the finite L10 route*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, the consumer first reads the located endpoint row I, then the dyadic midpoint and sign-apartness rows M, A, then the shrinking bracket ledger L, then the stream and regular rational readback rows S, R, and only then the terminal Real seal E. The handoff exports no host real equality, selected root, quotient stream equality, unrestricted sign trichotomy, chosen limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $B = (I, M, A, L, S, R, E, H, C, P, N)$. The source interval is displayed by I, so every endpoint or membership read begins in the $\mathsf{LocatedInterval}^{\uparrow}$ row.

Next read the bisection rows. The dyadic midpoint row M and the sign or apartness row A determine the retained child bracket as finite BEDC evidence, while L records the shrinking bracket history that downstream consumers may inspect.

Then pass to L10 readback. The finite windows S and regular rational row R are built only from the retained bracket ledger, and the terminal seal E is read only after those rows are present.

Finally replay the route through H, C, P, N. These structural rows transport, replay, source, and name the same finite packet, so none of the refused host-level or completion-level claims can be exported by the bisection-search handoff. □

Falsifiable prediction: Every accepted $\mathsf{IntervalBisectionSearch}^{\uparrow}$ consumer must expose located endpoints, dyadic midpoint data, sign or apartness witnesses, shrinking bracket rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before using a bisection-search result.

Independence witness: locatedinterval, dyadicratcore, streamname, regseqrat, real: no listed sibling alone supplies the joint packet containing located endpoint rows, dyadic midpoint splitting, sign or apartness branch retention, shrinking bracket ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalBisectionSearch}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalBisectionSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6892.1, Definition 6892.2, and Definition 6892.3 seed the finite bisection-search route over located interval endpoints, dyadic midpoint rows, sign or apartness witnesses, shrinking brackets, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalBisectionSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close root existence, root uniqueness, host real trichotomy, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint exactness, sign and apartness locality, bracket shrinkage, L10 readback, transport, replay, provenance, and local naming for BEDC.Derived.IntervalBisectionSearchUp.

Constructive story: An $\mathsf{IntervalBisectionSearch}^{\uparrow}$ packet is generated from a located interval source, dyadic midpoint rows, sign or apartness branch witnesses, a shrinking bracket ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6893

A Concrete Naming Certificate for $\mathsf{OstrowskiCompletion}^{\uparrow}$

Following Bourbaki and Artin on valuations and absolute-value completions, the $\mathsf{OstrowskiCompletion}^{\uparrow}$ packet records the finite boundary at which a displayed field source may be routed to an Archimedean or non-Archimedean completion-facing certificate. It sits between the prime-scale p-adic row of chapter 100, the local-field boundary of chapter 173, the dyadic valuation row of chapter 2106, the completion-functor surface of chapter 423, and the terminal real-name seal of chapter 38. The packet names only displayed absolute-value, valuation-choice, Cauchy-window, seal, transport, replay, provenance, and local naming rows; it does not quotient completions, select limits, identify host fields, invoke trichotomy beyond the displayed branch row, or import choice.

Definition 6893.1 (Ostrowski completion carrier)**.** *An* $\mathsf{OstrowskiCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, A, V, B, W, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the source field row, including the finite arithmetic operations and equality classifier visible to the completion request.*
2. *A is the absolute-value row, displaying positivity, multiplicativity, and triangle-control data only as BEDC history.*
3. *V is the valuation-facing branch row, distinguishing the Archimedean route from the non-Archimedean route without exporting a host dichotomy.*
4. *B records the boundary handoff to the* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{LocalField}^{\uparrow}$*, and* $\mathsf{DyadicValuation}^{\uparrow}$ *siblings when the displayed branch is non-Archimedean.*
5. *W is the finite Cauchy-window ledger consumed before any completion-facing read may inspect the source.*
6. *S is the completion seal row, routed through* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *or local-field seal already displayed by the packet.*
7. *H records componentwise* hsame *transport for the source, absolute-value row, branch row, boundary handoff, Cauchy-window ledger, completion seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion request through $F, A, V, B, W, S, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{OstrowskiCompletion}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, $\mathsf{DyadicValuation}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OstrowskiCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, selected representatives, a completed host field, a global classification theorem, countable choice, or equality between Archimedean and non-Archimedean models.

Theorem 6893.2 (Ostrowski completion NameCert obligations)**.** *For every accepted* $\mathsf{OstrowskiCompletion}^{\uparrow}$ *carrier*

$$O = (F, A, V, B, W, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OstrowskiCompletion}^{\uparrow}}$ *surface consists exactly of source-field exposure, absolute-value row exposure, valuation-branch exposure, non-Archimedean sibling handoff, finite Cauchy-window admission, completion-seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every completion-facing consumer must display F, A, V, W before it may read the seal row S, and any non-Archimedean read of S must also display the sibling boundary row B.*

Proof. Project the carrier of Definition 6893.1. The displayed rows are precisely the source field, absolute-value row, valuation branch, sibling handoff, Cauchy-window ledger, completion seal, and structural rows.

The absolute-value and branch obligations are supplied by A and V. The row A records the finite algebraic and triangle-control data read by a completion request, while V records which completion route is visible without adding an external classification principle.

The completion handoff is then bounded by B, W, S. The non-Archimedean route must pass through the displayed $\mathsf{Padic}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, and $\mathsf{DyadicValuation}^{\uparrow}$ boundary row B; every route must pass through the finite Cauchy-window ledger W before the seal row S can be consumed through $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ and the relevant terminal seal.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since no quotient completion, selected limit, host-field equality, global classification theorem, or choice coordinate occurs in the carrier, the stated rows exhaust the NameCert obligation surface. □

Falsifiable prediction: An accepted $\mathsf{OstrowskiCompletion}^{\uparrow}$ replay must expose the same source field row, absolute-value row, valuation branch, Cauchy-window ledger, completion seal, transport, replay, provenance, and local naming before any consumer reads a completion-facing endpoint. A replay that reaches a quotient completion, selected representative, host completed field, hidden branch choice, or unrecorded limit rejects this NameCert instance.

Independence witness: The packet is independent of quotient completion carriers, selected Cauchy representatives, ambient field equality, and global classification theorems because its completion-facing read is mediated only by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. The Archimedean and non-Archimedean branches are visible as packet data, not as an external case split supplied by choice.

Closure status for $\mathsf{NameCert}_{\mathsf{OstrowskiCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{OstrowskiCompletion}^{\uparrow}$ packet is generated from a source field row, an absolute-value row, a valuation-branch row, non-Archimedean sibling handoff data, finite Cauchy-window rows, a completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OstrowskiCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 6893.1 and Definition 6893.2 seed the absolute-value completion boundary between Archimedean, p-adic, local-field, dyadic-valuation, Cauchy-completion, and Real-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{OstrowskiCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Ostrowski classification, quotient completions, selected Cauchy limits, host completed fields, branch trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, absolute-value row exposure, branch-row determinacy, sibling handoff exactness, Cauchy-window routing, completion-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.OstrowskiCompletionUp.

Chapter 6894

A Concrete Naming Certificate for $\mathsf{BishopSplineSpace}^{\uparrow}$

Following de Boor and Bishop–Bridges, the $\mathsf{BishopSplineSpace}^{\uparrow}$ packet records a constructive finite spline carrier over dyadic knot data, polynomial pieces, regular rational readback, and compact-uniform modulus routes. It sits after the dyadic rational core of chapter 291, the polynomial row of chapter 50, the Real seal of chapter 38, the regular sequence readback of chapter 301, and the compact-uniform handoff of chapter 2299. The packet names only the finite knot-and-piece data needed by later compact-uniform and Real approximation readers.

Definition 6894.1 (Bishop spline space carrier)**.** *A* $\mathsf{BishopSplineSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, C, P, R, E, H, T, Q, N)$$

with the following displayed rows:

1. *G is the knot-grid row. It contains a finite increasing dyadic list, its endpoint order tests, adjacent-cell membership tests, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *evidence used to replay each comparison without a hidden real cut.*

2. *C is the coefficient row family. For every displayed cell of G it stores the finite rational or dyadic coefficient tuple that determines the local polynomial piece.*

3. *P is the piecewise-polynomial row. It reads each coefficient tuple through the* $\mathsf{Poly}^{\uparrow}$ *carrier and records the cell-local polynomial value, degree bound, and endpoint trace requested by the finite knot grid.*

4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It turns each displayed local value and endpoint trace into the regular rational sequence data used by the surrounding constructive real layer.*

5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the finite Real comparison and tolerance tokens produced from R and keeps them attached to the same knot cell and polynomial piece.*

6. *H records componentwise* hsame *transport for G, C, P, R, E, T, Q, N and for the finite cell index used to select a piece.*

7. *T records displayed* Cont *replay of the route*

$$G, C \longmapsto P \longmapsto R \longmapsto E.$$

8. *Q is the* Pkg *provenance over* $\mathsf{BishopSplineSpace}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSplineSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed knot grid, coefficient rows, piecewise-polynomial rows, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.BishopSplineSpaceUp.BishopSplineSpaceTasteGate single carrier alignment`

Theorem 6894.2 (Bishop spline space NameCert obligations)**.** *For every accepted* $\mathsf{BishopSplineSpace}^{\uparrow}$ *carrier*

$$K = (G, C, P, R, E, H, T, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSplineSpace}^{\uparrow}}$ *surface consists exactly of knot-grid admission, coefficient-row admission, piecewise-polynomial interpretation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public spline read must pass through the finite rows $G, C, P, R, E, H, T, Q, N$ before it names a cell value, endpoint trace, or compact-uniform approximation datum.*

Proof. Project the carrier of Definition 6894.1. The analytic rows are G, C, P, R, E: the finite knot grid, finite coefficient family, polynomial interpretation, regular rational readback, and Real seal.

The structural rows then keep that analytic route finite. The transport row H moves the grid, coefficients, cell index, polynomial piece, readback, and seal together, while T replays the route from knot-and-coefficient data to Real-sealed local values.

Finally the provenance and local naming rows Q, N certify that the public object is exactly this finite packet. Since the classifier has no coordinate for a selected interpolation theorem, quotient function equality, ambient completion, or global approximation claim, those objects cannot be read from the local NameCert surface. □

Theorem 6894.3 (Bishop spline finite knot replay)**.** *For every accepted* $\mathsf{BishopSplineSpace}^{\uparrow}$ *carrier $K = (G, C, P, R, E, H, T, Q, N)$ and every displayed cell of the finite knot grid G, the spline value route is obtained by finite recursion over the knot list: select the adjacent cell row, read its coefficient tuple through C, interpret it through the corresponding polynomial row of P, pass the resulting local value through R, and seal the read in E. The replay exposes no open-ended node choice, quotient equality of functions, or ambient completeness witness.*

Proof. Induct over the finite knot list in G. The endpoint row supplies the base comparison, and each adjacent-pair step either identifies the displayed cell or passes to the next finite grid edge with the same $\mathsf{DyadicRatCore}^{\uparrow}$ comparison evidence.

Once the cell is identified, project the matching coefficient tuple from C and the matching polynomial row from P. The regular rational readback R and Real seal E are attached to that same cell by the componentwise transport row H.

Replay the route through T and record it through Q, N. The construction never asks for an interpolation-node selector, a quotient equality of global functions, an ambient completion point, or any infinitary compactness object beyond the displayed compact-uniform handoff named in the package provenance. □

Falsifiable prediction: An accepted $\mathsf{BishopSplineSpace}^{\uparrow}$ read is refuted if a consumer obtains a spline value, endpoint trace, or compact-uniform approximation datum without first displaying

the knot-grid, coefficient, polynomial, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, polynomial, real, regseqrat, compactuniformmodulus: $\mathsf{BishopSplineSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite knot-grid data, coefficient rows, piecewise-polynomial rows, regular rational readback, Real sealing, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSplineSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSplineSpace}^{\uparrow}$ packet is generated from finite dyadic knot-grid rows, coefficient rows, piecewise-polynomial rows, RegSeqRat readback rows, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSplineSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6894.1, Definition 6894.2, and Definition 6894.3 seed the finite spline-space route over dyadic knots, polynomial pieces, regular rational readback, Real sealing, and compact-uniform package provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSplineSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an approximation theorem, selected interpolation nodes, quotient function equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite knot recursion, piecewise-polynomial readback, Real sealing, compact-uniform handoff preservation, transport stability, replay stability, and local naming for BEDC.Derived.BishopSplineSpaceUp.

Chapter 6895

A Concrete Naming Certificate for $\mathsf{FiniteSupportingHyperplane}^{\uparrow}$

Following Rockafellar and Bishop–Bridges, the $\mathsf{FiniteSupportingHyperplane}^{\uparrow}$ packet records the finite-dimensional supporting-hyperplane data used by convex-analysis consumers before any separation principle is available. It sits beside the convex carrier of chapter 208, the vector-space carrier of chapter 47, the linear-map row of chapter 48, and the scalar seal of chapter 38. The packet names only displayed finite-dimensional rows; it does not claim Hahn–Banach, arbitrary Banach separation, selected minimizers, quotient dual spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, or choice.

Definition 6895.1 (Finite supporting hyperplane packet)**.** *A* $\mathsf{FiniteSupportingHyperplane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, C, p, \ell, \alpha, W, H, R, P, N)$$

with the following displayed rows:

1. *V is the finite-dimensional* $\mathsf{VecSpace}^{\uparrow}$ *row over the displayed* $\mathsf{Real}^{\uparrow}$ *scalar seal;*
2. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *row whose finite convex-combination operations are visible to the packet;*
3. *p is the boundary point row, recorded only as a point already admitted by the finite-dimensional carrier;*
4. *ℓ is the* $\mathsf{LinMap}^{\uparrow}$ *functional row;*
5. *α is the displayed* $\mathsf{Real}^{\uparrow}$ *scalar level row;*
6. *W is the finite witness ledger saying that the inspected convex rows lie on one side of the displayed affine hyperplane and that p lies on its boundary;*
7. *H records componentwise* hsame *transport for the vector, convex, point, functional, scalar, witness, replay, package, and naming rows;*
8. *R records displayed* Cont *replay routes by which downstream optimization or convexity consumers inspect the same finite rows;*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSupportingHyperplane}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportingHyperplane}^{\uparrow}}$ *row.*

The carrier is finite-dimensional by construction. It has no coordinate for Hahn–Banach, arbitrary Banach separation, selected minimizers, quotient dual equality, open-cover compactness, or choice-selected witnesses.

Theorem 6895.2 (Finite supporting hyperplane NameCert obligations)**.** *For every accepted* FiniteSupportingHyper *carrier* $K = (V, C, p, \ell, \alpha, W, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportingHyperplane}^{\uparrow}}$ *surface consists exactly of finite-dimensional vector-space admission, convex-set admission, boundary-point display, linear-functional display, scalar-level display, one-sided witness ledger exactness, componentwise transport, displayed replay, package provenance, and local naming. The certificate exports no Hahn–Banach principle, arbitrary Banach separation theorem, selected minimizer, quotient dual-space equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or choice principle.*

Proof. Project the packet rows. The finite-dimensional vector-space row V, convex row C, boundary-point row p, functional row ℓ, scalar row α, and witness row W provide the entire supporting-hyperplane surface.

Read the local certificate through the structural rows. The transport row H preserves the displayed coordinates componentwise, the replay row R exposes only the same finite route to consumers, and P, N record provenance and the local name.

Since no carrier coordinate stores Hahn–Banach separation, arbitrary Banach separation, selected minimizers, quotient dual equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or choice, the NameCert obligations cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSupportingHyperplane}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteSupportingHyperplane}^{\uparrow}$ packet is generated from finite-dimensional vector-space data, a displayed convex-set row, a boundary point, a linear functional, a scalar level, a finite one-sided witness ledger, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSupportingHyperplane}^{\uparrow}}$, obligation)

Scope closed: Definition 6895.1 and Definition 6895.2 specify the finite-dimensional carrier rows and local NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSupportingHyperplane}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Hahn–Banach, arbitrary Banach separation, selected minimizers, quotient dual-space equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A checked closure requires Lean carrier admission, witness-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSupportingHyperplaneUp.

Chapter 6896

A Concrete Naming Certificate for $\mathsf{WeakConvergenceHilbert}^{\uparrow}$

Following Hilbert space weak convergence theory, the $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ packet records weak convergence only through finite scalar-test rows over a displayed Hilbert source. It sits over the Hilbert carrier of chapter 91, the finite weak-topology window of chapter 3597, the norm comparison surface of chapter 89, the regular rational readback surface of chapter 301, and the scalar seal of chapter 38. The chapter names the finite observation certificate used by weak-convergence consumers; it does not select a countable dual, assert compactness, identify a quotient sequence space, or import ambient completeness beyond the displayed sibling rows.

Definition 6896.1 (Weak convergence in Hilbert carrier)**.** *A* $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, W, N, R, S, T, C, P, L)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It records the carrier endpoint, inner-product readback surface, and Hilbert-facing provenance used by the scalar tests.*
2. *W is the sibling* $\mathsf{WeakTopology}^{\uparrow}$ *finite-window row, reused as the weak-neighbourhood interface for the displayed Hilbert source.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *comparison row. It records norm bounds and norm-facing tolerance data without replacing weak convergence by norm convergence.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for indexed scalar-test tolerances and finite convergence budgets.*
5. *S is the scalar-test row. Each entry displays a Hilbert source point, a test vector or functional row already accepted by H, W, and the scalar observation sent to the* $\mathsf{Real}^{\uparrow}$ *seal.*
6. *T records componentwise* hsame *transport for the Hilbert source, weak-window row, norm comparison, regular rational readback, scalar-test row, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from a weak-convergence request through*

$$H, W, N, R, S \longmapsto T, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{WeakConvergenceHilbert}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{WeakConvergenceHilbert}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $H, W, N, R, S, T, C, P, L$. It has no coordinate for a selected dual basis, countable choice, compactness, quotient sequence equality, norm convergence, weak-star compactness, or an ambient host theorem.

Theorem 6896.2 (Weak convergence in Hilbert NameCert obligations)**.** *For every accepted* WeakConvergenceHilbe *carrier $Q = (H, W, N, R, S, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{WeakConvergenceHilbert}^{\uparrow}}$ *surface consists exactly of Hilbert source exposure through H, finite weak-topology window exposure through W, norm comparison through N, regular rational tolerance readback through R, scalar-test exposure through S, componentwise transport through T, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 6896.1. The only source row is H, the only weak-neighbourhood interface is W, and the only norm-facing data are the comparison budgets in N.

The convergence evidence is finite. The regular rational row R records the inspected tolerance budgets, and the scalar-test row S records the finitely many Hilbert observations sent to the $\mathsf{Real}^{\uparrow}$ seal.

The remaining rows are structural. Transport through T, replay through C, provenance through P, and local naming through L keep the same finite packet visible to every consumer. Since the carrier has no selected dual basis, choice schedule, compactness row, quotient sequence equality, norm-convergence coordinate, weak-star compactness row, or ambient host theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6896.3 (Weak convergence in Hilbert scalar readback route)**.** *Every scalar readback consumed from an accepted* $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ *packet factors through the displayed route*

$$H, W, N, R, S, T, C, P, L.$$

Thus a weak-convergence consumer may read only finite Hilbert scalar tests, regular rational tolerance budgets, norm-comparison boundaries, $\mathsf{Real}^{\uparrow}$ *scalar seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The route exports no selected dual sequence, no quotient function-space equality, no norm-convergence theorem, no compactness theorem, and no countable-choice principle.*

Proof. Begin with a scalar observation in S. Its Hilbert source is the same row H admitted by the carrier, and its weak-neighbourhood meaning is read through the same finite window W.

Read the tolerance data next. The row R supplies the regular rational budget for the observation, while N records the norm-facing comparison boundary without turning the weak read into a norm-convergence claim.

Finally send the scalar value to the $\mathsf{Real}^{\uparrow}$ seal and replay the route through T, C, P, L. Componentwise transport and displayed replay preserve the same observation window for consumers, so no selected dual sequence, quotient equality, norm-convergence theorem, compactness theorem, or choice principle can be obtained from the scalar readback. □

Theorem 6896.4 (Weak convergence in Hilbert finite-window nonescape)**.** *Any public read of an accepted* $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ *packet is exhausted by the finite Hilbert source row, weak-topology window, norm comparison, regular rational tolerance row, scalar-test row, transport, replay, provenance, and local naming data. In carrier notation the public route is*

$$H, W, N, R, S, T, C, P, L.$$

The packet does not export Banach–Alaoglu compactness, Mazur convex-block convergence, Kadec–Klee norm recovery, selected subsequences, a countable total test family, quotient sequence equality, or ambient Hilbert-space weak compactness.

Proof. Use the NameCert obligations theorem to expose the carrier rows. A public read must display H, W, N, R, S before it reaches any structural row.

Apply the scalar readback route. It sends every finite test through regular rational tolerances and the scalar seal before transport, replay, provenance, and local naming are consumed.

The refused analytic conclusions are not carrier coordinates. Since compactness, convex-block convergence, norm recovery, subsequence selection, countable test families, quotient equality, and ambient weak compactness are absent from $H, W, N, R, S, T, C, P, L$, no public read can export them. □

Falsifiable prediction: Every accepted $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ read must expose the $\mathsf{Hilbert}^{\uparrow}$ source, $\mathsf{WeakTopology}^{\uparrow}$ finite window, $\mathsf{Norm}^{\uparrow}$ comparison boundary, $\mathsf{RegSeqRat}^{\uparrow}$ tolerance row, scalar-test row, $\mathsf{Real}^{\uparrow}$ seal, BHist transport, hsame, Cont, Pkg, and local NameCert rows before any weak-convergence conclusion is public.

Independence witness: hilbert, weaktopology, norm, regseqrat, real: $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Hilbert source data, weak finite-window observations, norm comparison, regular rational tolerance readback, scalar-test rows, transport, replay, provenance, and local naming while refusing compactness, norm recovery, selected dual bases, and quotient sequence equality.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakConvergenceHilbert}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ packet is generated from a Hilbert source row, a WeakTopology finite-window row, a Norm comparison row, RegSeqRat tolerance readback, finite Real scalar-test rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakConvergenceHilbert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6896.1, Definition 6896.2, Definition 6896.3, and Definition 6896.4 seed the finite weak-convergence certificate over Hilbert scalar-test rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakConvergenceHilbert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, Mazur convex-block convergence, Kadec–Klee norm recovery, selected subsequences, countable total test families, quotient sequence equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite scalar-test stability, regular rational readback exactness, scalar nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeakConvergenceHilbertUp.

Chapter 6897

A Concrete Naming Certificate for CarlemanInequality$^{\uparrow}$

Following Hardy–Littlewood–Polya's treatment of inequalities, the CarlemanInequality$^{\uparrow}$ packet records the finite row surface by which a positive real-series source may expose partial-sum, geometric-mean, and summable-majorant control before a downstream inequality consumer reads the endpoint. It sits beside the real-series carrier of chapter 1581, the Cauchy majorant route of chapter 1774, the Young inequality packet of chapter 2358, and the Poincare inequality boundary of chapter 2329. The packet names a displayed Carleman inequality certificate; it does not choose an infinite sum, import host measure theory, quotient a sequence space, or assert a bridge-checked classical theorem.

Definition 6897.1 (Carleman inequality carrier)**.** *A* CarlemanInequality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, G, M, R, B, H, C, P, N)$$

with the following visible rows:

1. *S is the partial-sum row, recording the finite initial-segment sums of the displayed nonnegative real series source.*
2. *G is the geometric-mean row, recording the finite products and root reads used for the Carleman mean at each displayed index.*
3. *M is the summable-majorant row, carrying the majorant schedule and tail budget that bound the accumulated geometric means.*
4. *R is the* RealSeries$^{\uparrow}$ *readback row, exposing the source series only through the finite real-series packet.*
5. *B is the boundary comparison row, recording the displayed Carleman constant, endpoint budget, and comparison with the majorant route.*
6. *H records componentwise* hsame *transport for the partial-sum, geometric-mean, majorant, RealSeries readback, boundary comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Carleman read through S, G, M, R, B, H, P, N.*
8. *P is the* Pkg *provenance over* CarlemanInequality$^{\uparrow}$, RealSeries$^{\uparrow}$, CauchyMajorantSeries$^{\uparrow}$, YoungInequality$^{\uparrow}$, PoincareInequality$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite Carleman inequality packet. It has no coordinate for selected infinite sums, quotient sequence equality, ambient integral theory, host convexity, or a standard bridge.

Theorem 6897.2 (Carleman inequality NameCert obligations)**.** *For every accepted* $\mathsf{CarlemanInequality}^{\uparrow}$ *carrier* $K = (S, G, M, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$ *surface consists exactly of partial-sum exposure, geometric-mean exposure, summable-majorant exposure,* $\mathsf{RealSeries}^{\uparrow}$ *readback, boundary-comparison exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted Carleman inequality request is classified only by those finite rows.*

Proof. Project the carrier of Definition 6897.1. The analytic rows are the partial-sum row S, the geometric-mean row G, the summable-majorant row M, the real-series readback row R, and the boundary comparison row B.

The partial-sum and geometric-mean rows display the finite source of the inequality read. The summable-majorant row then supplies the tail budget and comparison schedule, while the real-series readback row keeps the source inside the finite $\mathsf{RealSeries}^{\uparrow}$ packet.

The boundary row B records the Carleman constant and endpoint comparison as packet data. It does not add an infinite-sum choice, quotient equality, host integral theorem, or convexity theorem.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Hence the local $\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$ row exhausts the obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{CarlemanInequality}^{\uparrow}$ read factors through partial-sum rows, geometric-mean rows, a summable-majorant row, a RealSeriesUp readback row, a boundary-comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, cauchy-majorant-series, young-inequality, poincare-inequality: $\mathsf{CarlemanInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries partial-sum rows, geometric-mean rows, a summable-majorant route, RealSeries readback, boundary comparison, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CarlemanInequality}^{\uparrow}$ packet is generated from partial-sum rows, geometric-mean rows, a summable-majorant row, a RealSeries readback row, a boundary-comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$, seed)

Scope closed: Definition 6897.1 and Definition 6897.2 seed the finite Carleman inequality route from displayed real-series data through geometric-mean and majorant comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CarlemanInequality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient sequence equality, ambient integral theory, host convexity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum exposure, geometric-mean exposure, summable-majorant exactness, RealSeries readback, boundary comparison, transport, replay, provenance, and local naming for BEDC.Derived.CarlemanInequalityUp.

Chapter 6898

A Concrete Naming Certificate for $\mathsf{CauchySchwarzCompletion}^{\uparrow}$

Following Riesz and Bishop–Bridges constructive analysis, the $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ packet records the finite completion-stable route by which a Cauchy–Schwarz estimate is carried from displayed real-vector inequality data into Cauchy-window and complete-metric handoff rows. It sits after the finite inequality certificate of chapter 1684, the Cauchy-window surface of chapter 406, the complete-metric boundary of chapter 128, and the terminal real-name seal of chapter 38. The packet names only the displayed finite route through regular rational readback, stream schedules, completion handoff, transport, replay, provenance, and local naming; it does not import a host Hilbert completion, quotient stream equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6898.1 (Cauchy–Schwarz completion carrier)**.** *A* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, M, Q, S, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *inequality row, including the displayed finite-vector source, inner-product read, norm-square rows, and dyadic square comparison already sealed for real-analysis consumers.*
2. *W is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *Cauchy-window row, carrying the common finite windows through which the inequality row is inspected as a completion-facing sequence of estimates.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, recording the displayed completeness request and the finite nonescape boundary for the estimate route.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the Cauchy-window observations and the dyadic tolerance comparisons attached to R.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that exposes the finite observation windows shared by R, W, Q, and M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the estimate value after the Cauchy-window and complete-metric handoff rows have been displayed.*
7. *H records componentwise* hsame *transport for $R, W, M, Q, S, E, C, P, N$.*

8. *C records displayed* Cont *replay from the finite Cauchy–Schwarz inequality row through the Cauchy-window, regular-readback, stream-schedule, complete-metric, and Real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Hilbert completion, quotient equality of Cauchy names, selected completed representatives, countable choice, detached real limits, or ambient completeness.

Theorem 6898.2 (Cauchy–Schwarz completion NameCert obligations)**.** *For every accepted* CauchySchwarzComple *carrier $K = (R, W, M, Q, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy–Schwarz real-inequality exposure through R, Cauchy-window exposure through W, complete-metric handoff exposure through M, regular rational readback through Q, shared stream scheduling through S, terminal Real seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6898.1. The only inequality source is the displayed $\mathsf{CauchySchwarzReal}^{\uparrow}$ row R, whose finite vector, inner-product, norm-square, and dyadic square-comparison data are already part of the accepted packet.

The completion-facing route is then forced by W, M, Q, S, E. The Cauchy-window row W exposes the common finite windows, the complete-metric row M records the displayed handoff request, the regular rational row Q reads the finite observations, the stream row S schedules those observations, and the terminal Real row E seals the same route.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no row for host Hilbert completion, quotient Cauchy-name equality, selected completed representatives, countable choice, detached real limits, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6898.3 (Cauchy–Schwarz completion limit handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{CauchySchwarzReal}^{\uparrow}, \mathsf{CauchySequenceSpace}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow} \longmapsto R, W, M, Q, S, E, H, C, P, N.$$

In particular, the Cauchy–Schwarz estimate is available at a completion boundary only after the Cauchy-window row, complete-metric handoff row, regular rational readback, stream schedule, terminal Real seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start from an accepted packet $K = (R, W, M, Q, S, E, H, C, P, N)$. The estimate source is R, so every inequality read already factors through the finite real-vector and dyadic comparison rows named by chapter 1684.

The estimate can then enter completion only through the visible Cauchy-window route. The row W supplies the inspected Cauchy windows, S schedules the finite observations, and Q records the regular rational readback attached to those windows.

The handoff row M is the only complete-metric coordinate, and the Real seal E is downstream of R, W, M, Q, S. Thus a consumer cannot read a completion-stable estimate without first displaying the Cauchy-window, complete-metric, readback, schedule, and seal rows.

Finally, H, C, P, N keep the handoff attached to the same packet by transport, replay, provenance, and local naming. The route exports a finite completion-stable estimate boundary, not a

host Hilbert completion, quotient stream equality, selected limit, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ read is refuted if a completion-facing consumer obtains the estimate boundary without first displaying the CauchySchwarzRealUp inequality row, CauchySequenceSpaceUp window row, CompleteMetricUp handoff row, RegSeqRatUp readback, StreamNameUp schedule, RealUp seal, transport, replay, provenance, and local NameCert rows.

Independence witness: cauchy-schwarz-real, cauchysequence-space, completemetric, regseqrat, streamname, real: $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an inequality row, Cauchy-window row, complete-metric handoff row, regular readback, stream schedule, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ packet is generated from a CauchySchwarzRealUp inequality row, CauchySequenceSpaceUp windows, a CompleteMetricUp handoff row, RegSeqRatUp readback, StreamNameUp schedules, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6898.1, Definition 6898.2, and Definition 6898.3 seed the finite completion-stable Cauchy–Schwarz estimate route over displayed Cauchy windows and complete-metric handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert completion, quotient equality of Cauchy names, selected completed representatives, countable choice, detached real limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy–Schwarz inequality exposure, Cauchy-window exposure, complete-metric handoff, regular rational readback, stream scheduling, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySchwarzCompletionUp.

Chapter 6899

A Concrete Naming Certificate for $\mathsf{ArtinRees}^{\uparrow}$

Following Atiyah–Macdonald on the Artin–Rees lemma, the $\mathsf{ArtinRees}^{\uparrow}$ packet records the finite filtration interface by which an ideal-adic filtration on a displayed module is compared with the induced filtration on a displayed submodule. It sits after the ring carrier of chapter 43, the ideal carrier of chapter 81, the module carrier of chapter 46, and the Noetherian module surface of chapter 3648. The packet names only carrier rows, filtration rows, stabilization bounds, transport, replay, provenance, and local naming; it does not import quotient-module classification, selected generators, ambient Noetherian induction, or host equality of filtrations.

Definition 6899.1 (Artin–Rees carrier)**.** *An* $\mathsf{ArtinRees}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{ArtinRees}(R, I, M, L, F, A, H, Q, P, N).$$

The row R is the $\mathsf{Ring}^{\uparrow}$ *source surface, I is the* $\mathsf{Ideal}^{\uparrow}$ *row over that ring, M is the ambient* $\mathsf{Module}^{\uparrow}$ *row, and L is the displayed submodule row read inside the same module classifier. The row F records the finite ideal-adic filtration windows $I^n M$ and $I^n M \cap L$ visible to the packet. The row A is the Artin–Rees stabilization bound, asserting that after a displayed finite stage the induced filtration on L is read through the same ideal power route. The row H is componentwise* hsame *transport for the ring, ideal, module, submodule, filtration, and stabilization rows; Q is displayed* Cont *replay; P is* Pkg *provenance; and N is the local* $\mathsf{NameCert}_{\mathsf{ArtinRees}^{\uparrow}}$ *row.*

Theorem 6899.2 (Artin–Rees NameCert obligations)**.** *For every accepted* $\mathsf{ArtinRees}^{\uparrow}$ *carrier* $\mathsf{ArtinRees}(R, I, M, L, F, A, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArtinRees}^{\uparrow}}$ *surface consists exactly of source exposure for* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{Ideal}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*, and* $\mathsf{NoetherianModule}^{\uparrow}$*; submodule exposure through L; ideal-adic filtration exposure through F; finite stabilization access through A; componentwise transport through H; displayed replay through Q; provenance through P; and local naming through N.*

Proof. Project the displayed packet of Definition 6899.1. The rows R, I, M, L fix the algebraic sources before any filtration comparison is read.

The filtration row F exposes only the finite windows named by the packet, and the stabilization row A gives the finite stage at which the induced filtration on the submodule is read through the same ideal power route. This is the Artin–Rees obligation surface, not a quotient-module classifier or selected generator list.

Transport H and replay Q preserve those rows componentwise. Provenance P and local naming N then repack the same finite inventory as the local certificate, so no hidden Noetherian induction or host filtration equality enters the obligations. □

Theorem 6899.3 (Artin–Rees module-ideal consumer)**.** *Every downstream algebraic consumer that cites* $\mathsf{ArtinRees}^{\uparrow}$ *must first read the same ring row* R*, ideal row* I*, module row* M*, submodule row* L*, filtration row* F*, stabilization row* A*, transport row* H*, replay row* Q*, provenance row* P*, and local naming row* N*. The consumer may use the packet as a finite module–ideal stabilization certificate, but it receives no selected basis, quotient-module classification, maximal-submodule choice, or ambient equality of filtrations.*

Proof. Begin with the NameCert obligations of Definition 6899.2. They force every read to expose the ring, ideal, module, Noetherian module, and submodule sources before the filtration rows are available.

The rows F and A provide the only stabilization data available to a module–ideal consumer. Since they are displayed rows inside the same packet, the consumer cannot bypass them through a host Artin–Rees theorem or through quotient-module classification.

Finally, H, Q, P, N preserve the route as transport, replay, provenance, and local naming. Thus the downstream consumer receives exactly the finite stabilization certificate and nothing outside the displayed carrier. □

Falsifiable prediction: Any accepted $\mathsf{ArtinRees}^{\uparrow}$ consumer that cites ideal-adic stabilization without first exposing $R, I, M, L, F, A, H, Q, P, N$ refutes the carrier surface of Definition 6899.1.

Independence witness: ring, ideal, module, noetherian-module: no listed sibling is bijective to $\mathsf{ArtinRees}^{\uparrow}$, because this packet jointly binds ideal-adic filtration windows, a displayed submodule route, finite stabilization, componentwise transport, replay, provenance, and local NameCert rows into one Artin–Rees carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ArtinRees}^{\uparrow}}$.

Constructive story: An $\mathsf{ArtinRees}^{\uparrow}$ packet is generated from a ring source, ideal source, module source, submodule row, ideal-adic filtration windows, a finite stabilization bound, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ArtinRees}^{\uparrow}}$, seed)

Scope closed: Definition 6899.1, Definition 6899.2, and Definition 6899.3 seed the finite Artin–Rees module–ideal filtration interface over RingUp, IdealUp, ModuleUp, NoetherianModuleUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ArtinRees}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-module classification, selected generators, ambient Noetherian induction, full Artin–Rees theorem strength, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, ideal-adic filtration exposure, finite stabilization access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.ArtinReesUp.

Chapter 6900

A Concrete Naming Certificate for $\mathsf{FloorFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{FloorFunction}^{\uparrow}$ packet records the ordinary floor-function read as a finite certificate route rather than as a primitive operation on host real numbers. It consumes the real-name source of chapter 38, the integer bracket supplied by chapter 1479, the Real-facing integer-part handoff of chapter 1784, the residual row of chapter 1648, the dyadic tolerance core of chapter 291, the finite window surface of chapter 126, and the regular rational readback route of chapter 301. The packet names the displayed lower-integer value attached to a real name; it is not host floor, trichotomy, quotient real equality, selected representatives, countable choice, or ambient Real completeness.

Definition 6900.1 (FloorFunction carrier)**.** *A* $\mathsf{FloorFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, Z, F, D, S, G, I, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose floor read is being requested.*
2. *B is the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row attached to the same source real.*
3. *Z is the* $\mathsf{Int}^{\uparrow}$ *candidate exposed by the bracket as the lower integer endpoint.*
4. *F is the* $\mathsf{FractionalPart}^{\uparrow}$ *residual row tied to the same source real and integer candidate.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the finite tolerances used to inspect the source, bracket, and residual rows.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened for the finite real-name and residual observations.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets the displayed windows before the floor value is exported.*
8. *I is the* $\mathsf{RealIntegerPart}^{\uparrow}$ *handoff row that packages the same source, bracket, integer candidate, residual, dyadic scale, window, and readback route for Real-facing consumers.*
9. *H records componentwise* hsame *transport for the source real, bracket, integer candidate, residual, dyadic scale, window, readback, RealIntegerPart handoff, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from every floor request through R, B, Z, F, D, S, G, I without inserting another integer search.*

11. *P is the* Pkg *provenance over* $\mathsf{FloorFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{FractionalPart}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealIntegerPart}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host floor, trichotomy over arbitrary real names, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6900.2 (FloorFunction NameCert obligations)**.** *For every accepted* $\mathsf{FloorFunction}^{\uparrow}$ *carrier $K = (R, B, Z, F, D, S, G, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the source* $\mathsf{Real}^{\uparrow}$ *row,* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row,* $\mathsf{Int}^{\uparrow}$ *candidate row,* $\mathsf{FractionalPart}^{\uparrow}$ *residual row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{RealIntegerPart}^{\uparrow}$ *handoff row, and structural rows H, C, P, N;*

2. *bracket exactness, requiring the integer candidate Z to be read only as the lower endpoint of the displayed* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket B for the same source row R;*

3. *residual compatibility, requiring F to bind the same R, B, Z route before any fractional residual observation is exposed;*

4. *handoff compatibility, requiring the* $\mathsf{RealIntegerPart}^{\uparrow}$ *row I to consume exactly the displayed bracket, residual, dyadic, window, and readback rows;*

5. *transport and replay stability, requiring H and C to preserve the rows componentwise and to send each floor request back through R, B, Z, F, D, S, G, I;*

6. *ledger refusal, excluding host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, and ambient Real completeness from the certificate surface.*

No FloorFunction obligation is stored outside these rows.

Proof. Project the accepted packet $K = (R, B, Z, F, D, S, G, I, H, C, P, N)$. The projection exposes exactly the real source, bracket, integer candidate, residual, dyadic scale, stream window, regular readback, RealIntegerPart handoff, transport, replay, provenance, and naming rows.

Bracket exactness is read from chapter 1479. The row B binds the source real to a finite lower endpoint and successor comparison, and Z is available only as the displayed integer candidate of that bracket.

Residual compatibility is read from chapter 1648. The residual row F attaches to the same source real and integer candidate, so it cannot provide an independent floor selector.

The Real-facing handoff is read through chapter 1784. The row I packages the same bracket route together with the dyadic scale, finite window, and regular readback rows.

Finally H, C, P, N transport, replay, source, and name the displayed packet. Since no excluded object appears as a carrier coordinate or classifier clause, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}$ obligation surface. □

Theorem 6900.3 (FloorFunction RealIntegerPart handoff)**.** *Every consumer that reads a floor value from an accepted* $\mathsf{FloorFunction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{ArchimedeanIntegerPart}^{\uparrow} \longmapsto \mathsf{Int}^{\uparrow} \longmapsto \mathsf{FractionalPart}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeq}$$

In particular, every such read passes through chapter 1479 and chapter 1784. It receives no host floor function, trichotomy service, quotient real equality, selected representative, countable-choice row, or ambient Real completeness theorem.

Proof. Let a downstream consumer request the floor value. The replay row C sends the request first to the source $\mathsf{Real}^{\uparrow}$ row R, so the route starts from a displayed real name rather than from a host real.

The bracket row B then supplies the $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ certificate named by chapter 1479. The integer candidate Z is the lower endpoint exported by that finite bracket.

The residual and readback rows keep the endpoint attached to the same real-name route. The row F records the $\mathsf{FractionalPart}^{\uparrow}$ residual, while D, S, G expose only finite dyadic tolerances, finite windows, and regular rational readbacks.

The handoff row I is exactly the $\mathsf{RealIntegerPart}^{\uparrow}$ consumer surface of chapter 1784. Transport by H preserves the same rows componentwise, and P, N record no additional operation. Therefore every exported floor read factors through the stated route. □

Falsifiable prediction: Every $\mathsf{FloorFunction}^{\uparrow}$ read factors through one $\mathsf{Real}^{\uparrow}$ source row, one $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, one $\mathsf{Int}^{\uparrow}$ candidate row, one $\mathsf{FractionalPart}^{\uparrow}$ residual row, one $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, one $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{RealIntegerPart}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, archimedean-integer-part, int, fractional-part, dyadicratcore, streamname, regseqrat, real-integer-part: no listed sibling is bijective to $\mathsf{FloorFunction}^{\uparrow}$ because this packet jointly binds the floor-facing name, bracket endpoint, residual, finite tolerance scale, window read, regular readback, RealIntegerPart handoff, transport, replay, provenance, and local naming for the same real source.

Closure status for $\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6900.1, Definition 6900.2, and Definition 6900.3 seed the finite floor-function certificate route from a displayed Real source through ArchimedeanIntegerPart bracket, Int candidate, FractionalPart residual, DyadicRatCore scale, StreamName window, RegSeqRat readback, RealIntegerPart handoff, and structural rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FloorFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, bracket exactness, residual compatibility, RealIntegerPart handoff compatibility, transport replay stability, and ledger refusal for BEDC.Derived.FloorFunctionUp.

Constructive story: A $\mathsf{FloorFunction}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ source row, an $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, an $\mathsf{Int}^{\uparrow}$ candidate row, a $\mathsf{FractionalPart}^{\uparrow}$ residual row, a $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, a $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{RealIntegerPart}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6901

A Concrete Naming Certificate for $\mathsf{CeilingFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CeilingFunction}^{\uparrow}$ packet records the ordinary upper-integer read as a finite certificate route rather than as a primitive operation on host real numbers. It is the upper-integer dual of the lower endpoint route in chapter 6900; it consumes the upper dyadic enclosure discipline of chapter 1690, the integer bracket supplied by chapter 1479, the residual row of chapter 1648, and the real-name source of chapter 38. The packet names the displayed upper integer attached to a real name; it is not host ceiling, order trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6901.1 (CeilingFunction carrier)**.** *A* $\mathsf{CeilingFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, Z, U, F, D, Y, I, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose upper-integer read is being requested.*
2. *B is the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row attached to the same source real.*
3. *Z is the* $\mathsf{Int}^{\uparrow}$ *lower endpoint exposed by the bracket.*
4. *U is the* $\mathsf{Int}^{\uparrow}$ *upper endpoint exposed by the same bracket, read as the successor-side integer when the residual row is nonzero and as Z on the exact-integer branch.*
5. *F is the* $\mathsf{FractionalPart}^{\uparrow}$ *residual row tied to the same source real and lower endpoint.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the finite tolerances used to inspect the source, bracket, and residual rows.*
7. *Y is the* $\mathsf{DyadicCeiling}^{\uparrow}$ *upper-grid row used to compare the real-name window with the displayed upper endpoint.*
8. *I is the* $\mathsf{RealIntegerPart}^{\uparrow}$ *handoff row that packages the same source, bracket, lower endpoint, residual, dyadic scale, dyadic ceiling row, and upper-integer export for Real-facing consumers.*
9. *H records componentwise* hsame *transport for the source real, bracket, lower endpoint, upper endpoint, residual, dyadic scale, dyadic ceiling, RealIntegerPart handoff, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from every ceiling request through R, B, Z, U, F, D, Y, I without inserting another integer search.*

11. *P is the* Pkg *provenance over* $\mathsf{CeilingFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{FractionalPart}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicCeiling}^{\uparrow}$, $\mathsf{RealIntegerPart}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host ceiling, trichotomy over arbitrary real names, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6901.2 (CeilingFunction NameCert obligations)**.** *For every accepted* $\mathsf{CeilingFunction}^{\uparrow}$ *carrier $K = (R, B, Z, U, F, D, Y, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the source* $\mathsf{Real}^{\uparrow}$ *row,* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row, lower and upper* $\mathsf{Int}^{\uparrow}$ *endpoint rows,* $\mathsf{FractionalPart}^{\uparrow}$ *residual row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row,* $\mathsf{DyadicCeiling}^{\uparrow}$ *upper-grid row,* $\mathsf{RealIntegerPart}^{\uparrow}$ *handoff row, and structural rows H, C, P, N;*

2. *bracket exactness, requiring Z and U to be read only as the lower and upper endpoint rows of the displayed* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket B for the same source row R;*

3. *residual compatibility, requiring F to bind the same R, B, Z route before the exact-integer and successor-side branches for U are exposed;*

4. *upper-grid compatibility, requiring Y to read the finite dyadic ceiling route through the scale row D before any upper-integer export is public;*

5. *handoff compatibility, requiring the* $\mathsf{RealIntegerPart}^{\uparrow}$ *row I to consume exactly the displayed bracket, residual, dyadic scale, dyadic ceiling, and upper-endpoint rows;*

6. *transport and replay stability, requiring H and C to preserve the rows componentwise and to send each ceiling request back through R, B, Z, U, F, D, Y, I;*

7. *ledger refusal, excluding host ceiling, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, and ambient Real completeness from the certificate surface.*

No CeilingFunction obligation is stored outside these rows.

Proof. Project the accepted packet $K = (R, B, Z, U, F, D, Y, I, H, C, P, N)$. The projection exposes exactly the real source, bracket, lower endpoint, upper endpoint, residual, dyadic scale, dyadic ceiling, RealIntegerPart handoff, transport, replay, provenance, and naming rows.

Bracket exactness is read from chapter 1479. The row B binds the source real to a finite lower endpoint and successor-side comparison, so Z and U are available only as the displayed endpoint rows of that bracket.

Residual compatibility is read from chapter 1648. The residual row F attaches to the same source real and lower endpoint, which fixes the exact-integer branch and the successor-side branch before U can be exported.

The upper dyadic comparison is read through chapter 1690. The row Y is inspected at the scale recorded by D, so the upper-integer read uses a finite upper-grid route rather than a host ceiling operation.

Finally I, H, C, P, N hand off, transport, replay, source, and name the displayed packet. Since no excluded object appears as a carrier coordinate or classifier clause, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}$ obligation surface. □

Theorem 6901.3 (CeilingFunction floor-dual handoff)**.** *Every consumer that reads a ceiling value from an accepted* $\mathsf{CeilingFunction}^{\uparrow}$ *carrier receives the upper-integer dual of the floor route of chapter 6900. The lower endpoint Z is read through the same bracket and residual route used by* $\mathsf{FloorFunction}^{\uparrow}$*, while the exported endpoint U is constrained by the displayed upper branch and the* $\mathsf{DyadicCeiling}^{\uparrow}$ *upper-grid row Y. Thus the consumer obtains a finite upper-integer handoff, not host ceiling, trichotomy, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier of Definition 6901.1. The rows $R, B, Z, F, D, I, H, C, P, N$ mirror the lower-endpoint rows used by chapter 6900, so the lower bracket read is shared with the floor route.

The extra endpoint row U records the upper side of the same finite bracket. Its exact-integer branch and successor-side branch are fixed only after the residual row F is tied to the same source and lower endpoint.

The row Y supplies the upper-grid comparison through chapter 1690. Consequently the exported endpoint is constrained by a displayed dyadic-ceiling read before the Real-facing handoff row I can be consumed.

The carrier has no coordinate for a host ceiling function or an ambient completeness theorem. Therefore the ceiling consumer receives only the finite upper-integer dual of the existing floor handoff. □

Theorem 6901.4 (CeilingFunction RealIntegerPart handoff)**.** *Every consumer that reads a ceiling value from an accepted* $\mathsf{CeilingFunction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{ArchimedeanIntegerPart}^{\uparrow} \longmapsto \mathsf{Int}^{\uparrow} \longmapsto \mathsf{FractionalPart}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{DyadicCeiling}^{\uparrow} \longmapsto \mathsf{RealInt}$$

In particular, every such read passes through chapter 1479, chapter 1690, and chapter 1784. It receives no host ceiling function, trichotomy service, quotient real equality, selected representative, countable-choice row, hidden Archimedean search, or ambient Real completeness theorem.

Proof. Let a downstream consumer request the ceiling value. The replay row C sends the request first to the source $\mathsf{Real}^{\uparrow}$ row R, so the route starts from a displayed real name rather than from a host real.

The bracket row B then supplies the $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ certificate named by chapter 1479. The endpoint rows Z and U are the lower and upper integer rows exported by that finite bracket.

The residual and dyadic rows keep the endpoint attached to the same real-name route. The row F records the $\mathsf{FractionalPart}^{\uparrow}$ residual, while D and Y expose only finite dyadic tolerances and a displayed upper-grid read.

The handoff row I is the $\mathsf{RealIntegerPart}^{\uparrow}$ consumer surface. Transport by H preserves the same rows componentwise, and P, N record no additional operation. Therefore every exported ceiling read factors through the stated route. □

Falsifiable prediction: Every $\mathsf{CeilingFunction}^{\uparrow}$ read factors through one $\mathsf{Real}^{\uparrow}$ source row, one $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, lower and upper $\mathsf{Int}^{\uparrow}$ endpoint rows, one $\mathsf{FractionalPart}^{\uparrow}$ residual row, one $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, one $\mathsf{DyadicCeiling}^{\uparrow}$ upper-grid row, one $\mathsf{RealIntegerPart}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, archimedean-integer-part, int, fractional-part, dyadicratcore, dyadic-ceiling, real-integer-part: no listed sibling is bijective to $\mathsf{CeilingFunction}^{\uparrow}$ because this packet jointly binds the ceiling-facing name, lower and upper bracket endpoints, residual branch, finite tolerance scale, upper dyadic grid read, RealIntegerPart handoff, transport, replay, provenance, and local naming for the same real source.

Closure status for $\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6901.1, Definition 6901.2, Definition 6901.3, and Definition 6901.4 seed the finite ceiling-function certificate route from a displayed Real source through ArchimedeanIntegerPart bracket, Int endpoint rows, FractionalPart residual, DyadicRatCore scale, DyadicCeiling upper-grid row, RealIntegerPart handoff, and structural rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CeilingFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host ceiling, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, bracket exactness, residual compatibility, upper-grid compatibility, RealIntegerPart handoff compatibility, transport replay stability, and ledger refusal for BEDC.Derived.CeilingFunctionUp.

Constructive story: A $\mathsf{CeilingFunction}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ source row, an $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, lower and upper $\mathsf{Int}^{\uparrow}$ endpoint rows, a $\mathsf{FractionalPart}^{\uparrow}$ residual row, a $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, a $\mathsf{DyadicCeiling}^{\uparrow}$ upper-grid row, a $\mathsf{RealIntegerPart}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6902

A Concrete Naming Certificate for $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ packet records the finite route by which a contraction on a displayed complete metric carrier produces a fixed-point endpoint surface. It sits over the complete-metric carrier of chapter 128, the contraction graph of chapter 1418, the finite Picard orbit of chapter 2359, the Banach endpoint discipline of chapter 396, the regular-rational handoff of chapter 301, and the real-name seal of chapter 38. The packet names a constructive fixed-point route only through displayed BEDC data; it does not import selected host fixed points, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6902.1 (CompleteMetricFixedPoint carrier)**.** *A* $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, T, I, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, including the metric carrier, distance readback, regular Cauchy limit service, and local complete-metric naming route.*
2. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *graph row, with a displayed self-map route, graph admission, rational contraction ledger, and strict unit-bound row.*
3. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *finite orbit row generated from an initial point by repeated reads of T.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail budget extracted from the residual distances carried by I and the contraction ledger of T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reading the Picard orbit as a regular rational tail route before any endpoint-facing seal is consumed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after X, M, R have displayed the complete-metric limit request and the regular readback route.*
7. *H records componentwise* hsame *transport for the complete metric source, contraction graph, finite orbit, modulus, regular handoff, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-distance requests through contraction application, finite Picard iteration, Cauchy-tail budgeting, regular handoff, and endpoint sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host fixed-point object, quotient equality of functions, a countable-choice selector for iterates, or an ambient real-completeness oracle outside the carried $\mathsf{CompleteMetric}^{\uparrow}$ *source.*

Theorem 6902.2 (CompleteMetricFixedPoint NameCert obligations)**.** *For every accepted* CompleteMetricFixedPoi *carrier $F = (X, T, I, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricFixedPoint}^{\uparrow}}$ *surface consists exactly of complete-metric source admission, contraction-graph admission, Picard-orbit visibility, Cauchy-modulus tail budgeting,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff visibility,* $\mathsf{Real}^{\uparrow}$ *seal separation, component-wise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the carrier of Definition 6902.1.*

Proof. Project the carrier of Definition 6902.1. The displayed tuple exposes $X, T, I, M, R, E, H, C, P, N$ before any endpoint-facing consumer may read the packet.

Use the sibling interfaces. The $\mathsf{CompleteMetric}^{\uparrow}$ row supplies the carried limit-service surface, the $\mathsf{ContractionMapping}^{\uparrow}$ row supplies the graph and contraction-bound data, and the $\mathsf{PicardIteration}^{\uparrow}$ row supplies the finite orbit and residual-distance ledger.

The rows M, R, E form the completion-facing tail. The Cauchy-modulus row packages the finite tail budget, the regular-rational row packages the readback route, and the real seal remains downstream of both.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve the same carrier, so the NameCert surface contains exactly the listed rows and none of the excluded host fixed-point, quotient, choice, or ambient-completeness coordinates. □

Theorem 6902.3 (CompleteMetricFixedPoint Picard route)**.** *Every public fixed-point read of an accepted* $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may then be consumed by the $\mathsf{BanachFixedPoint}^{\uparrow}$ *endpoint discipline, but only after the finite orbit, tail budget, regular readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. It exports no selected host fixed point, quotient function equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the complete metric and contraction rows. The source row X is the only complete-metric surface available to the packet, and the graph row T is the only route for applying the contraction.

Read the Picard orbit next. The row I is finite and is generated by repeated visible applications of T, so it records a displayed orbit rather than a chosen infinite sequence or a selected endpoint.

Use the tail budget and handoff rows. The row M records the Cauchy modulus extracted from the orbit and contraction ledger, R records the regular rational readback route, and E is reached only after those rows have been exposed.

The Banach fixed-point sibling can consume this endpoint route as a fixed-point discipline, but it receives only $X, T, I, M, R, E, H, C, P, N$. Since the carrier has no additional coordinate, the public Picard route cannot be read as host fixed-point equality, quotient function equality, countable choice, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ read factors through one CompleteMetricUp source, one ContractionMappingUp graph, one PicardIterationUp finite orbit, one CauchyModulusUp tail budget, one RegSeqRatUp handoff, one RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any fixed-point endpoint is consumed.

Independence witness: completemetric, contractionmapping, picarditeration, banachfixedpoint, regseqrat, real: none is bijective with $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ because the packet jointly stores the complete metric source, contraction graph, Picard orbit, Cauchy tail budget, regular handoff, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricFixedPoint}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6902.1, Definition 6902.2, and Definition 6902.3 seed the complete-metric fixed-point packet over displayed contraction, Picard, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient function equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-graph admission, Picard-orbit visibility, Cauchy-modulus tail budgeting, RegSeqRat handoff visibility, Real seal separation, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{CompleteMetricFixedPoint}^{\uparrow}$ packet is generated from a CompleteMetricUp source row, ContractionMappingUp graph row, PicardIterationUp finite orbit, CauchyModulusUp tail budget, RegSeqRatUp handoff, RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6903

A Concrete Naming Certificate for $\mathsf{UniformBar}^{\uparrow}$

Following Bishop and Brouwer, the $\mathsf{UniformBar}^{\uparrow}$ packet records the finite-depth bar extraction used when a compact-uniform-continuity argument turns pointwise inspection into one shared modulus. It sits between the finite fan rows of chapter 400, the compact source rows of chapter 129, the metric and continuity surfaces of $\mathsf{Met}^{\uparrow}$ and $\mathsf{Continuous}^{\uparrow}$, and the rational precision rows of chapter 37. The packet is a finite certificate over displayed bar decisions, compact probes, pointwise modulus reads, rational tolerances, BHist histories, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

It is not Brouwer bar induction, not open-cover compactness, not a selected infinite branch, and not a host modulus function. A downstream L10 compact-uniform-continuity or real-completion consumer reads only the finite depth, the inspected witness window, and the rational precision ledger named by the packet.

Definition 6903.1 (UniformBar carrier)**.** *A* $\mathsf{UniformBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, K, M, A, R, D, W, H, C, P, N)$$

with the following visible rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row supplying a decidable bar and unary branch-depth ledger;*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite probes are inspected by the bar;*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *source-distance row used to compare neighbouring compact probes;*
4. *A is the* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to each displayed probe;*
5. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance request whose response is the extracted uniform depth;*
6. *D is the accepted finite depth row, read as a unary bound common to all displayed probe branches;*
7. *W is the finite witness window listing exactly the probe prefixes and pointwise modulus rows inspected before depth D;*
8. *H stores componentwise* hsame *transports for the fan, compact, metric, continuity, rational, depth, witness-window, provenance, and local naming rows;*
9. *C stores displayed* Cont *replay routes for bar inspection, compact-probe readback, pointwise-modulus readback, rational-tolerance comparison, and uniform-depth export;*

10. *P is* Pkg *provenance over* $\mathsf{UniformBar}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformBar}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite branch, an open cover, a selected subcover, an ambient compactness theorem, a host modulus function, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6903.2 (UniformBar NameCert obligations)**.** *For every accepted* $\mathsf{UniformBar}^{\uparrow}$ *carrier* $U = (F, K, M, A, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformBar}^{\uparrow}}$ *surface consists exactly of fan-bar exposure through F, compact-source exposure through K, metric comparison through M, pointwise continuity exposure through A, rational tolerance exposure through R, finite-depth exposure through D, witness-window exposure through W, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6903.1. The rows F, K, M, A, R fix the fan, compact, metric, continuity, and rational inputs before any uniform depth can be read.

The row D is the only finite-depth output, and W records the exact probe prefixes and pointwise modulus reads inspected to justify it. Thus the bar extraction is a finite ledger over displayed rows, not an appeal to an infinite branch space or an open-cover compactness principle.

Transport H and replay C preserve those rows componentwise. Provenance P and the local row N then repack the same inventory as the local $\mathsf{NameCert}_{\mathsf{UniformBar}^{\uparrow}}$ surface, so the obligation surface contains no hidden subcover selection, host modulus function, or real-completion coordinate. □

Theorem 6903.3 (UniformBar finite-depth modulus)**.** *Every* $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *or L10 real-completion consumer that reads an accepted* $\mathsf{UniformBar}^{\uparrow}$ *carrier receives a modulus only through the finite route*

$$\mathsf{FanTheorem}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Continuous}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ D,\ W,\ H,\ C,\ P,\ N.$$

The extracted modulus is the displayed depth D together with the witness window W and its rational tolerance ledger. It cannot be replaced by open-cover compactness, Brouwer bar induction, a selected infinite branch, a host supremum of pointwise moduli, quotient real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 6903.2. Any consumer read must expose the fan-bar row, compact source row, metric comparison row, pointwise continuity row, rational tolerance row, depth row, witness-window row, and the structural rows of the packet.

The finite fan row contributes only the decidable bar and unary branch-depth ledger, as in the public finite fan route of Definition 400.7. The compact and metric rows determine which probes are inspected, and the continuity rows attach pointwise modulus data only to those probes.

Finally read the rational request through R and the extracted depth through D. The witness window W lists every pointwise row used before that depth, while H, C, P, N preserve the same finite route under transport, replay, provenance, and naming. Since the carrier exposes no open-cover, branch-selection, host-modulus, quotient-real, or real-completion coordinate, the consumer receives exactly the finite-depth modulus described above. □

Falsifiable prediction: Any accepted $\mathsf{CompactUniformContinuity}^{\uparrow}$ or $\mathsf{UniformModulus}^{\uparrow}$ consumer that obtains a uniform depth without first exposing $F, K, M, A, R, D, W, H, C, P, N$ refutes the carrier surface of Definition 6903.1.

Independence witness: fan theorem, compact source, metric source, continuity row, rational tolerance: no listed sibling is bijective to $\mathsf{UniformBar}^{\uparrow}$, because this packet jointly binds a finite

bar depth, compact probe inspection, pointwise modulus reads, rational tolerance response, componentwise transport, replay, provenance, and local NameCert rows into one extraction carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformBar}\uparrow}$.

Constructive story: A $\mathsf{UniformBar}^{\uparrow}$ packet is generated from a FanTheoremUp finite bar row, CompactUp or CompactMetricUp source probes, MetricUp comparison rows, ContinuousUp or ContinuousMapUp pointwise modulus rows, a positive RatUp tolerance request, a finite depth row, a witness-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformBar}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6903.1, Definition 6903.2, and Definition 6903.3 seed the finite-depth bar extraction surface over FanTheoremUp, CompactUp, CompactMetricUp, MetricUp, ContinuousUp, ContinuousMapUp, RatUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformBar}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Brouwer bar induction, open-cover compactness, selected infinite branches, host modulus extraction, quotient real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar exposure, compact probe inspection, pointwise modulus exposure, rational tolerance response, depth-window exactness, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.UniformBarUp.

Chapter 6904

A Concrete Naming Certificate for $\mathsf{InverseFunctionModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{InverseFunctionModulus}^{\uparrow}$ packet records the local inverse data that must be visible before an inverse-function read is treated as quantitative. It sits downstream of the local inverse carrier of chapter 2055, the derivative source rows of chapter 122, the pointwise continuity route of chapter 130, the terminal real-name seal of chapter 38, the regular rational readback schedule of chapter 301, and the finite Newton-step service of chapter 223. The packet names a finite modulus boundary for local inverse construction; it does not turn local data into a global inverse theorem, host topology, or choice-selected inverse branch.

Definition 6904.1 (Inverse-function modulus carrier)**.** *An* $\mathsf{InverseFunctionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, L, B, R, C_m, N_t, H, K, P, N)$$

with the following rows:

1. *W is the source point window, naming the base point, source radius, target radius, and the finite neighbourhood on which the local inverse read is allowed.*
2. *L is the derivative lower-bound row, recording the displayed nonzero derivative or invertible-linear lower estimate used to keep the inverse step separated from collapse.*
3. *B is the* $\mathsf{Real}^{\uparrow}$ *local ball row, sealing the source and target ball endpoints as real-name data rather than as host open sets.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rate-budget row, carrying the rational modulus schedule and the finite tolerance ledger used to compare requested output precision with input-window precision.*
5. *C_m is the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row for the displayed map on the same window.*
6. *N_t is the* $\mathsf{NewtonIteration}^{\uparrow}$ *or Picard contraction row, recording the finite contraction certificate and the first accepted iteration window.*
7. *H records componentwise* hsame *transport for source windows, derivative lower bounds, Real balls, rate budgets, continuous-map modulus rows, Newton or Picard rows, provenance, and local naming.*
8. *K records displayed* Cont *replay through*

$$W, L, B, R, C_m, N_t \longmapsto \mathsf{NameCert}_{\mathsf{InverseFunctionModulus}^{\uparrow}}.$$

9. *P is the* Pkg *provenance over* InverseFunctionModulus$^{\uparrow}$, InverseFunction$^{\uparrow}$, Derivative$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, NewtonIteration$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{InverseFunctionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, L, B, R, C_m, N_t, H, K, P, N$. It has no coordinate for a global inverse function, a selected inverse branch outside the displayed window, ambient completeness, classical open mapping, or choice-based local selection.

Theorem 6904.2 (Inverse-function modulus local bijection obligation)**.** *For every accepted* InverseFunctionModulus *carrier*

$$M = (W, L, B, R, C_m, N_t, H, K, P, N),$$

the local inverse-bijection obligation is read only from the displayed source window, derivative lower-bound row, Real local ball, RegSeqRat rate budget, ContinuousMap modulus row, Newton or Picard contraction row, componentwise transport, replay, provenance, and local naming. In particular, any local inverse read exported through M must keep the same window W, lower bound L, ball B, budget R, modulus C_m, and contraction row N_t visible at the consumer boundary.

Proof. Project the carrier of Definition 6904.1. The rows W, L, B, R, C_m, N_t are the only analytic rows in the packet, and each is tied to the same local window by the classifier.

The lower-bound row L and the continuous-map modulus row C_m provide the quantitative separation and continuity data that a local inverse read may inspect. The Real ball B and the RegSeqRat budget R keep the neighbourhood and tolerance data inside displayed real names and finite regular-rational schedules.

The Newton or Picard row N_t supplies the finite contraction service on this same window. Componentwise hsame transport and displayed Cont replay repack exactly those rows into the local NameCert$_{\text{InverseFunctionModulus}^{\uparrow}}$ surface, so no global inverse function, host open mapping theorem, selected inverse branch, or ambient completeness coordinate is introduced. □

Theorem 6904.3 (Inverse-function modulus Newton-window stability)**.** *Classified* InverseFunctionModulus$^{\uparrow}$ *packets remain classified after replacing the source point window, derivative lower-bound row, Real local ball, RegSeqRat rate budget, ContinuousMap modulus row, or Newton/Picard contraction row by componentwise* hsame*-related rows, provided the replay row still displays the same route*

$$W, L, B, R, C_m, N_t \longmapsto \text{NameCert}_{\text{InverseFunctionModulus}^{\uparrow}}.$$

The resulting packet exports the same local inverse-modulus boundary and does not add a fresh window, selected inverse branch, or unbudgeted convergence claim.

Proof. Start with two accepted packets and the componentwise hsame comparisons for W, L, B, R, C_m, N_t. The classifier of Definition 6904.1 permits transport only along those displayed coordinates and the structural rows H, K, P, N.

Use the transported Real ball and RegSeqRat rate budget to keep the local metric and tolerance rows attached to the same source window. Then transport the ContinuousMap modulus row and the Newton or Picard contraction row through the same comparison data. The replay row K repacks this transported route as one displayed history.

Since the transported packet still uses the same ten rows, it has no coordinate from which to choose a new inverse branch, enlarge to a global inverse theorem, import completeness, or appeal to classical open mapping. The local inverse-modulus boundary is therefore stable under the displayed componentwise transport. □

Closure status for $\mathsf{NameCert}_{\mathsf{InverseFunctionModulus}\uparrow}$.

Constructive story: An $\mathsf{InverseFunctionModulus}^{\uparrow}$ packet is generated from a source point window, derivative lower-bound row, Real local ball, RegSeqRat rate budget, ContinuousMap modulus row, Newton or Picard contraction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InverseFunctionModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6904.1, Definition 6904.2, and Definition 6904.3 seed the finite local inverse-modulus route over Real, RegSeqRat, continuous-map, derivative, and Newton/Picard rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InverseFunctionModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global inverse-function theorem, RealUp completeness, classical open mapping, choice-based local selection, selected inverse germs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, local bijection obligation coverage, Newton-window stability, rate-budget exactness, provenance exactness, and local naming for BEDC.Derived.InverseFunctionModulusUp.

Chapter 6905

A Concrete Naming Certificate for $\mathsf{PseudoArc}^{\uparrow}$

Following Bing's pseudo-arc construction, the $\mathsf{PseudoArc}^{\uparrow}$ packet records a hereditarily indecomposable chainable compact metric continuum through finite inverse-limit rows rather than through a host continuum object. It sits over the compact-metric source of chapter 129, the finite projection discipline of chapter 2168, the complete-metric boundary of chapter 128, the stream-window surface of chapter 126, and the regular rational readback of chapter 301. The packet names only displayed finite chain covers, projection refinements, and indecomposability obstruction rows before any continuum-facing consumer may read it.

Definition 6905.1 (Pseudo-arc carrier)**.** *A* $\mathsf{PseudoArc}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (K, I, C, D, R, E, H, T, G, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite metric covers, compact-metric provenance, and distance readback for the current continuum window.*
2. *I is the* $\mathsf{InverseLimitMetric}^{\uparrow}$ *row, recording a finite prefix of compact metric continua, bonding-map windows, and projection-compatibility checks.*
3. *C is the chainability row: for each displayed scale it lists a finite chain cover and its adjacency ledger over the same compact metric source.*
4. *D is the hereditary-indecomposability row, recording finite obstruction tests for splitting a displayed subcontinuum window into two proper subcontinua.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the diameters, overlap tolerances, and bonding-map moduli inspected by C and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the requested tolerance budget.*
7. *H records componentwise* hsame *transport for compact-metric, inverse-limit, chainability, indecomposability, readback, seal, provenance, and naming rows.*
8. *T records displayed* Cont *replay from compact-metric cover requests through inverse-limit projections, chain covers, obstruction tests, and regular rational readback.*
9. *G is the* Pkg *provenance over* $\mathsf{PseudoArc}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{InverseLimitMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{PseudoArc}^\uparrow}$ *row.*

The classifier compares accepted packets only through $K, I, C, D, R, E, H, T, G, N$. It has no coordinate for a host pseudo-arc, quotient continuum equality, a choice-selected inverse-limit point, open-cover compactness, or an ambient completeness theorem.

Theorem 6905.2 (Pseudo-arc NameCert obligations)**.** *For every accepted* $\mathsf{PseudoArc}^\uparrow$ *carrier $P = (K, I, C, D, R, E, H, T, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{PseudoArc}^\uparrow}$ *surface consists exactly of compact-metric source admission through K, finite inverse-limit prefix admission through I, displayed chain-cover coverage through C, hereditary-indecomposability obstruction exposure through D, regular rational tolerance readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through T, provenance through G, and local naming through N.*

Every continuum-facing consumer must read these rows before it may use the packet as pseudo-arc data, and no obligation row is discharged by a host pseudo-arc theorem, quotient equality of continua, selected inverse-limit representatives, open-cover compactness, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 6905.1. The compact source K and inverse-limit prefix I are the only continuum-source rows, so every consumer begins with displayed compact-metric covers and finite projection compatibility.

The chainability and hereditary-indecomposability rows are finite. The row C lists chain covers at inspected scales, while D lists obstruction windows preventing a displayed subcontinuum split. The regular rational row R supplies the tolerance readback needed by both rows before the real seal E is consumed.

Transport and replay keep the packet local. The rows H and T preserve the same compact-metric, inverse-limit, chainability, obstruction, readback, and seal data under hsame and Cont, while G and N provide provenance and naming.

Since the carrier has no coordinate for a host pseudo-arc, quotient continuum equality, selected inverse-limit point, open-cover compactness, countable choice, or ambient real completeness, the local NameCert surface is exhausted by the displayed rows. □

Lean4 (statement) `BEDC.Derived.PseudoArcUp`

Falsifiable prediction: Every accepted $\mathsf{PseudoArc}^\uparrow$ read exposes compact-metric cover rows, finite inverse-limit prefix rows, chain-cover rows, hereditary-indecomposability obstruction rows, regular rational tolerance readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a continuum-facing consumer can inspect the packet.

Independence witness: compactmetric, inverse-limit-metric, completemetric, streamname, regseqrat: $\mathsf{PseudoArc}^\uparrow$ is not bijective to any listed sibling because this packet binds chainability rows, hereditary-indecomposability obstruction rows, finite inverse-limit prefixes, compact-metric source rows, readback, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PseudoArc}^\uparrow}$**.**

Constructive story: A $\mathsf{PseudoArc}^\uparrow$ packet is generated from CompactMetricUp source rows, finite InverseLimitMetricUp prefix rows, chain-cover rows, hereditary-indecomposability obstruction rows, RegSeqRatUp tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PseudoArc}^\uparrow}$, seed)

Scope closed: Definition 6905.1 and Definition 6905.2 seed the finite pseudo-arc boundary over compact-metric, inverse-limit, chainability, obstruction, readback, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PseudoArc}^\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim a host pseudo-arc theorem, quotient continuum equality, selected inverse-limit representatives, open-cover compactness, countable choice, Lean-checked obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite inverse-limit prefix admission, chain-cover coverage, hereditary-indecomposability obstruction exposure, tolerance readback, transport, replay, provenance, and local naming for BEDC.Derived.PseudoArcUp.

Chapter 6906

A Concrete Naming Certificate for $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$ packet records the finite oscillation criterion by which a displayed regular Cauchy rational name may be handed to Real and Completion consumers without quotienting streams or selecting limits. It sits after the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the bounded oscillation packet of chapter 6787. The packet names only the displayed windows, dyadic budgets, oscillation rows, readback route, and seal boundary.

Definition 6906.1 (Regular Cauchy oscillation criterion carrier)**.** *A* $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, D, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the finite inspected stream-name entries of the candidate regular Cauchy rational name.*
2. *W is the finite observation-window ledger, including the paired indices and tail windows on which oscillation is tested.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget carrying the requested radius, slack, and comparison cells.*
4. *O is the finite oscillation-bound row, recording that the displayed windows are below the carried dyadic budget.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows, dyadic tolerances, and oscillation ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after S, W, D, O, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, dyadic, oscillation, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an oscillation-criterion request through S, W, D, O, R and then to the seal E.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedOscillation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected limit, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyOscillationCriterionUp.RegularCauchyOscillationCriterionTasteGate single carrier alignment`

Theorem 6906.2 (Regular Cauchy oscillation criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$ *carrier* $K = (S, W, D, O, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillatio}}$ *surface consists exactly of carrier admission for the finite stream-name source row, observation-window ledger, dyadic tolerance budget, finite oscillation-bound row, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6906.1. The analytic rows are S, W, D, O, R, E, and the structural rows are H, C, P, N.

The source and window obligations are finite. The row S supplies the displayed stream-name entries, while W lists the paired tail windows on which the oscillation test is read.

The dyadic and oscillation obligations are also finite. The row D supplies the tolerance budget, and O records the bounded oscillation comparison against that budget before the regular rational readback R is visible.

Finally, E seals the same route as a Real-facing name, and H, C, P, N transport, replay, source, and name the packet. Since no further coordinate exists, the listed rows exhaust the NameCert surface and exclude quotient streams, selected limits, countable choice, host equality, trichotomy, and ambient completeness. □

Theorem 6906.3 (Regular Cauchy oscillation criterion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ W,\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BoundedOscillation}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the consumer sees the same stream source row, observation windows, dyadic tolerance budget, finite oscillation ledger, regular rational readback, and Real seal before the completion-facing boundary is available. It receives no quotient stream equality, selected limit, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (S, W, D, O, R, E, H, C, P, N)$. By Definition 6906.2, the regular rational readback R is downstream of the stream source S, observation windows W, dyadic tolerance budget D, and oscillation-bound row O.

The handoff therefore cannot expose R directly. It must first display the finite windows and dyadic comparisons, then read the bounded oscillation row as the criterion that the regular Cauchy tail has stabilized within the carried budget.

After R is visible, the seal E supplies the Real-facing endpoint along the same replay route C. Componentwise transport H, provenance P, and local naming N preserve that route and add no quotient, choice, trichotomy, host equality, selected-limit, or ambient-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}$ packet is generated from a StreamName source row, finite observation windows, DyadicRatCore tolerance budget, finite oscillation-bound row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6906.1, Definition 6906.2, and Definition 6906.3 seed the finite regular Cauchy oscillation criterion for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyOscillationCriterionUp.RegularCauchyOscillationCriterionTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance exactness, oscillation-bound confinement, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyOscillationCriterionUp.

Chapter 6907

A Concrete Naming Certificate for $\mathsf{BergeMaximum}^{\uparrow}$

Following Berge's maximum theorem and Aubin–Frankowska set-valued analysis, the $\mathsf{BergeMaximum}^{\uparrow}$ packet records the compact-parameter, compact-valued feasible graph, and continuous real objective data needed to read the maximum-value function through finite BEDC rows. It sits over the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the terminal real seal of chapter 38, the upper-hemicontinuity boundary of chapter 6553, and the scalar finite-net readback of chapter 1493. The packet names a maximum-value ledger, not a selected optimizer or host choice principle.

Definition 6907.1 (Berge maximum carrier)**.** *A* $\mathsf{BergeMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (K_0, F, G, V, M, H, C, P, N)$$

with the following displayed rows:

1. K_0 *is the compact parameter-source row, read through the compact metric certificate;*
2. F *is the compact-valued feasible-graph row, including the finite graph windows used by the upper-hemicontinuity boundary;*
3. G *is the continuous* $\mathsf{Real}^{\uparrow}$*-valued objective row on the displayed feasible graph;*
4. V *is the finite-net value-window ledger for objective reads over each displayed feasible fibre;*
5. M *is the maximum-value comparison row, obtained by folding the finite* $\mathsf{Real}^{\uparrow}$ *value windows;*
6. H *stores componentwise* hsame *transport for* K_0, F, G, V, M *and the local naming rows;*
7. C *stores the displayed* Cont *replay from compact parameter windows, feasible graph rows, objective reads, and finite value windows to the maximum-value comparison;*
8. P *is the* Pkg *provenance over the same compact-parameter and value-window packet;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BergeMaximum}^{\uparrow}}$ *row.*

The carrier keeps the selected-argmax socket outside the exported rows: consumers may read the value ledger M*, but no row chooses an optimizer from a feasible fibre.*

Theorem 6907.2 (Berge maximum NameCert obligations)**.** *For the carrier of Definition 6907.1, the local* $\mathsf{NameCert}_{\mathsf{BergeMaximum}^{\uparrow}}$ *surface consists of five finite obligations: carrier admission for*

K; upper-hemicontinuity and compact-value boundary for F over K_0; continuous objective $\mathsf{Real}^{\uparrow}$ *sealing for G; finite-net maximum-value readback through V and M; and refusal of selected argmax data, open-cover compactness, and choice outside the displayed packet.*

Theorem 6907.3 (Finite-net maximum-value attainment route)**.** *Every accepted* $\mathsf{BergeMaximum}^{\uparrow}$ *packet exports its maximum-value ledger by a finite* ProbeBundle *fold over compact nets and* $\mathsf{Real}^{\uparrow}$*-sealed objective values. The exported row is the value comparison M together with its* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *boundary; no exported row selects an optimizer.*

Proof. Read the compact parameter row K_0 and feasible graph row F through the compact metric and upper-hemicontinuity certificates. These rows provide finite feasible-fibre windows for the current parameter read, rather than an ambient open-cover compactness theorem.

Next replay the continuous objective row G through the $\mathsf{Real}^{\uparrow}$ seal. Each finite feasible-window read contributes a value window in V, and the comparison row M is obtained by folding those displayed real value windows.

Finally transport the folded value row through H, replay it through C, and record its provenance in P and local name in N. The fold exposes a maximum-value ledger, while the selected-argmax request remains outside $K_0, F, G, V, M, H, C, P, N$. □

Lean4 (statement) `BEDC.Derived.BergeMaximumUp`

Closure status for $\mathsf{NameCert}_{\mathsf{BergeMaximum}^{\uparrow}}$.

Constructive story: A $\mathsf{BergeMaximum}^{\uparrow}$ packet is generated from compact parameter data, a compact-valued feasible graph, a continuous real objective, finite-net value windows, a maximum-value comparison row, componentwise transport, replay, provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BergeMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6907.1, Definition 6907.2, and Definition 6907.3 name the finite maximum-value carrier and selected-argmax refusal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BergeMaximum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected optimizer extraction, open-cover compactness, countable choice, quotient function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-hemicontinuity compact-value boundary, continuous objective sealing, finite-net value readback, and local naming for the same packet.

Chapter 6908

A Concrete Naming Certificate for $\mathsf{CompactFunctionAlgebra}^{\uparrow}$

Following the Stone–Weierstrass and Gelfand traditions for continuous functions on compact spaces, the $\mathsf{CompactFunctionAlgebra}^{\uparrow}$ packet records a finite algebra of $\mathsf{Real}^{\uparrow}$-valued continuous functions over a compact metric source. It realizes the compact-uniform-continuity route of chapter 9237 through the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the terminal real seal of chapter 38, the uniform-modulus surface of chapter 369, the compact-open function-space carrier of chapter 2221, and the approximation interface of chapter 1919. The exported object is a finite operation and norm ledger, not an ambient function quotient.

Definition 6908.1 (Compact function algebra carrier)**.** *A* $\mathsf{CompactFunctionAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, +, \cdot, 1, U, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row;*
2. *F is the finite family of continuous* $\mathsf{Real}^{\uparrow}$*-valued function rows over K;*
3. *$+$ and $\cdot$ are pointwise addition and multiplication rows on displayed members of F;*
4. *1 is the constant-one row, read as a continuous* $\mathsf{Real}^{\uparrow}$*-valued function;*
5. *U records uniform-modulus data for every displayed function and every displayed operation output;*
6. *S is the sup-norm finite-net readback row, computed from compact-net value windows and* $\mathsf{Real}^{\uparrow}$ *comparison rows;*
7. *H stores componentwise* hsame *transport for $K, F, +, \cdot, 1, U, S$ and the local naming rows;*
8. *C stores the displayed* Cont *replay from compact-source reads, continuous-function rows, operation rows, uniform-modulus rows, and sup-norm windows to the algebra ledger;*
9. *P is the* Pkg *provenance over the same compact function packet;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFunctionAlgebra}^{\uparrow}}$ *row.*

The carrier exports operation stability and uniform-norm readback only for the displayed finite packet; it does not identify functions by an ambient quotient or import a completed Banach algebra.

Theorem 6908.2 (Compact function algebra NameCert obligations)**.** *For the carrier of Definition 6908.1, the local* $\mathsf{NameCert}_{\mathsf{CompactFunctionAlgebra}^{\uparrow}}$ *surface consists of five finite obligations: carrier admission for A; stability of pointwise addition, pointwise multiplication, and the constant-one row inside the displayed family; uniform-modulus handoff for each displayed function and operation output; sup-norm readback through finite compact nets and* $\mathsf{Real}^{\uparrow}$ *value windows; and refusal of ambient function quotienting, completed Banach-algebra closure, unrestricted infinite linear combinations, and selected dense approximation data outside the packet.*

Theorem 6908.3 (Compact function algebra uniform-norm route)**.** *Every accepted* $\mathsf{CompactFunctionAlgebra}^{\uparrow}$ *packet reads its sup-norm row through compact finite nets and* $\mathsf{Real}^{\uparrow}$*-sealed value windows. The resulting ledger is transported by H, replayed by C, recorded by P, and named by N without exporting an ambient function quotient or a completed algebra.*

Proof. Read the compact metric source row K and the finite family F through the displayed continuous-map rows. The pointwise operation rows $+$, $\cdot$, and 1 stay inside the finite packet, so operation stability is checked before any norm comparison is read.

Next use the uniform-modulus row U to pass from pointwise value windows to compact finite-net windows. Each sampled value is sealed as a $\mathsf{Real}^{\uparrow}$ read, and the sup-norm row S is obtained by folding the displayed value-window comparisons over the finite net.

Finally transport the operation and sup-norm ledger through H, replay the route through C, record provenance in P, and attach the local name N. The route exposes finite algebra and norm data, while completed closure, quotient equality, and selected approximation witnesses remain outside A. □

Lean4 (statement) `BEDC.Derived.CompactFunctionAlgebraUp`

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFunctionAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactFunctionAlgebra}^{\uparrow}$ packet is generated from a compact metric source, finite continuous real-valued function rows, pointwise addition and multiplication rows, a constant-one row, uniform-modulus rows, sup-norm finite-net readback, componentwise transport, replay, provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactFunctionAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 6908.1, Definition 6908.2, and Definition 6908.3 name the finite compact function algebra carrier and uniform-norm readback boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactFunctionAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close completed Banach-algebra structure, unrestricted Stone–Weierstrass approximation, quotient function equality, selected dense witnesses, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation stability, uniform-modulus handoff, sup-norm finite-net readback, and local naming for the same packet.

Chapter 6909

A Concrete Naming Certificate for $\mathsf{DyadicNetTree}^{\uparrow}$

Following Christ's dyadic cubes and finite metric net-tree practice in constructive metric analysis, the $\mathsf{DyadicNetTree}^{\uparrow}$ packet records a multiscale finite-net skeleton as BEDC NameCert data. It sits over the metric source of chapter 57, the doubling control of chapter 2035, the levelwise finite-net rows of chapter 1767, the total-bounded coverage surface of chapter 127, the covering-number budget of chapter 2187, and the dyadic scale ledger of chapter 291. The packet names the finite multiscale route only; it does not import a host cover tree, quotient equality of trees, selected centers outside the displayed net rows, or ambient compactness.

Definition 6909.1 (DyadicNetTree carrier)**.** *A* $\mathsf{DyadicNetTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, S, E, T, B, L, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, recording the carrier, distance-reading route, and local metric naming surface.*
2. *S is the* $\mathsf{DoublingMetric}^{\uparrow}$ *scale-control row, recording the finite doubling budget used to compare adjacent dyadic levels.*
3. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *family row, exposing the center bundle selected at each displayed dyadic level.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *coverage row, asserting only the displayed finite coverage witnesses read from the same levelwise nets.*
5. *B is the* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *budget row, storing the finite upper bound consumed by the level ledger.*
6. *L is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale ledger, storing level indices, dyadic radii, mesh comparisons, and late-radius requests.*
7. *R is the parent-child refinement row, recording for every displayed child center the parent center, scale comparison, containment witness, and nonescape certificate visible at the next coarser level.*
8. *H records componentwise* hsame *transport for the metric source, doubling row, finite-net family, total-bounded coverage, covering budget, dyadic ledger, parent-child refinement, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a multiscale net-tree request through M, S, E, T, B, L, R, H.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicNetTree}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicNetTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, E, T, B, L, R, H, C, P, N$. It has no coordinate for a selected host tree, an unbounded center oracle, minimal cover cardinality, open-cover compactness, quotient equality of trees, or ambient completion.

Theorem 6909.2 (DyadicNetTree NameCert obligations)**.** *For every accepted* $\mathsf{DyadicNetTree}^{\uparrow}$ *carrier*

$$D = (M, S, E, T, B, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicNetTree}^{\uparrow}}$ *surface consists exactly of carrier admission through M, scale visibility through S and L, levelwise coverage through E and T, parent-child refinement through R, budget nonescape through B, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 6909.1. The metric source row M is the only carrier admission surface, while S and L expose the doubling and dyadic scale data before any level is read.

The levelwise coverage obligations are then forced through E and T. The finite-net row displays the centers at each inspected level, and the total-bounded row supplies only the coverage witnesses attached to those displayed centers.

The budget and refinement rows are separate. The row B keeps the covering-number budget visible, and the row R records the parent of each displayed child center together with the containment and nonescape witnesses needed for the next coarser level.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host-tree, minimal-cardinality, open-cover, quotient-tree, center-oracle, or ambient-completion coordinate, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{DyadicNetTree}^{\uparrow}$ read exposes the MetricUp source, DoublingMetricUp scale-control row, FiniteEpsilonNetUp level centers, TotallyBoundedUp coverage witnesses, MetricCoveringNumberUp budget, DyadicRatCoreUp scale ledger, parent-child refinement row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any multiscale tree consumer reads the packet.

Independence witness: metric, doublingmetric, finite-epsilon-net, totallybounded, metric-covering-number, dyadicratcore: no listed sibling is bijective with $\mathsf{DyadicNetTree}^{\uparrow}$ because this packet jointly stores levelwise centers, scale control, coverage, budget, dyadic ledger, parent-child refinement, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNetTree}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicNetTree}^{\uparrow}}$, seed)

Scope closed: Definition 6909.1 and Definition 6909.2 seed the dyadic net-tree packet over displayed metric, doubling, finite-net, total-bounded, covering-budget, dyadic-scale, parent-child, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicNetTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host cover-tree equality, selected centers outside displayed net rows, minimal covering cardinality, open-cover compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, scale visibility, levelwise coverage, parent-child refinement, budget nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicNetTreeUp.

Constructive story: A DyadicNetTree$^{\uparrow}$ packet is generated from a MetricUp source row, DoublingMetricUp scale-control row, FiniteEpsilonNetUp centers per dyadic level, TotallyBoundedUp coverage row, MetricCoveringNumberUp budget row, DyadicRatCoreUp scale ledger, parent-child refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6910

A Concrete Naming Certificate for $\mathsf{DedekindCutBoundary}^{\uparrow}$

Following Dedekind and Bishop–Bridges constructive analysis, the $\mathsf{DedekindCutBoundary}^{\uparrow}$ packet records the L10 constraint that Dedekind cuts are comparison data, not a construction of $\mathsf{Real}^{\uparrow}$. It sits between the located cut surface of chapter 404, the Dedekind real comparison surface of chapter 3629, the regular Cauchy comparison route of chapter 1424, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite boundary at which a cut-shaped read may be compared with regular Cauchy data without becoming an ambient real construction.

Definition 6910.1 (Dedekind cut boundary carrier)**.** *A* $\mathsf{DedekindCutBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, D, Q, R, S, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *comparison row, recording only the displayed lower–upper separation witnesses used by finite consumers.*
2. *D is the* $\mathsf{DedekindReal}^{\uparrow}$ *comparison row, read as a cut-shaped interface rather than as a constructor for* $\mathsf{Real}^{\uparrow}$*.*
3. *Q is the* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *boundary row comparing L, D with regular Cauchy readback data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed regular rational windows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite inspected entries of the regular rational name.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the displayed rational comparison budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after the comparison route L, D, Q, R, S, Y has been displayed.*
8. *H records componentwise* hsame *transport for the located-cut, Dedekind-real, comparison, regular-readback, stream-window, dyadic, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from cut-shaped observations through the comparison row and then through the regular Cauchy readback to the seal E.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindCutBoundary}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindReal}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient cut, host supremum, selected representative, trichotomy branch, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DedekindCutBoundaryUp`

Theorem 6910.2 (Dedekind cut boundary NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutBoundary}^{\uparrow}$ *carrier* $B = (L, D, Q, R, S, Y, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutBoundary}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the* $\mathsf{LocatedCut}^{\uparrow}$ *comparison row,* $\mathsf{DedekindReal}^{\uparrow}$ *cut-shaped comparison row,* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *boundary row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DedekindCutBoundaryUp.DedekindCutBoundaryNameCertObligations`

Proof. Project the carrier of Definition 6910.1. The analytic rows are L, D, Q, R, S, Y, E, and the structural rows are H, C, P, N.

The cut-shaped rows are comparison data. The located-cut row L and the Dedekind-real row D are not read directly as a real constructor; they first pass through the $\mathsf{DedekindCauchyComparison}^{\uparrow}$ row Q.

The regular readback rows are finite. The row R is inspected through the stream windows S and dyadic tolerance ledger Y before the terminal seal E can be read.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier lists no quotient cut, selected representative, trichotomy branch, countable-choice selector, host supremum, or ambient completeness row, the displayed rows exhaust the local NameCert surface. □

Theorem 6910.3 (Dedekind cut boundary Real non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{DedekindCutBoundary}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedCut}^{\uparrow},\ \mathsf{DedekindReal}^{\uparrow} \longmapsto \mathsf{DedekindCauchyComparison}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto$$

Thus the packet may compare cut-shaped observations with regular Cauchy data, but it cannot construct $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts or export quotient cut equality, selected representatives, countable choice, trichotomy, host supremum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $B = (L, D, Q, R, S, Y, E, H, C, P, N)$. By Definition 6910.2, the $\mathsf{Real}^{\uparrow}$ seal E is downstream of the located-cut, Dedekind-real, comparison, regular-readback, stream-window, and dyadic rows.

The cut-shaped observations therefore cannot reach E directly. They are first compared at Q, then read through the regular rational row R, the finite stream windows S, and the tolerance row Y.

The structural rows H, C, P, N preserve this route but add no constructor coordinate. Hence a downstream consumer may use the packet as a boundary between Dedekind-cut comparison and regular Cauchy readback, not as a construction of $\mathsf{Real}^{\uparrow}$ from cuts. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutBoundary}^{\uparrow}$ packet is generated from LocatedCut comparison rows, DedekindReal comparison rows, DedekindCauchyComparison boundary rows, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6910.1, Definition 6910.2, and Definition 6910.3 seed the finite Dedekind-cut comparison boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutBoundary}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DedekindCutBoundaryUp.DedekindCutBoundaryNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close quotient cuts, selected representatives, countable choice, trichotomy, host supremum, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, LocatedCut comparison, DedekindReal comparison, DedekindCauchyComparison factorization, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance exactness, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutBoundaryUp.

Chapter 6911

A Concrete Naming Certificate for $\mathsf{PoincareBendixsonTrap}^{\uparrow}$

Following Poincare and Bendixson, the $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ packet records the finite trapping-region certificate used by planar autonomous dynamics before any periodic-orbit conclusion is read. It sits over the displayed ODE row of chapter 193, the flow-state surface of chapter 195, the compact metric carrier of chapter 129, the continuous-map route of chapter 130, and the terminal real-name seal of chapter 38. The packet names only the trap boundary and finite flow-window obligations; it does not select a host trajectory, complete a global flow, quotient path equality, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6911.1 (Poincare–Bendixson trap carrier)**.** *A* $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, D, K, F, B, I, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *row, naming the planar vector-field equation and the finite right-hand-side reads inspected by the packet.*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *row, naming the local flow-window discipline available to the trap without asserting global flow completion.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row for the displayed trapping carrier and its compact metric boundary.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row used to read the local flow map on the displayed windows.*
5. *B is the inward-boundary ledger, recording the finite boundary faces, normal-side tests, and exit-refusal checks for the trap.*
6. *I is the invariant-window row, recording that every accepted local flow read re-enters the displayed compact trap before any downstream consumer reads a recurrence claim.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for coordinates, distances, and boundary-side comparisons.*
8. *H records componentwise* hsame *transport for ODE, dynamics, compact metric, continuous-map, boundary, invariant-window, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from ODE reads through local flow windows, compact boundary checks, inward tests, invariant-window routing, and real-coordinate sealing.*

10. *P records* Pkg *provenance over* $\mathsf{PoincareBendixsonTrap}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareBendixsonTrap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $O, D, K, F, B, I, R, H, C, P, N$. It has no coordinate for selected host trajectories, completed global flows, quotient equality of paths, countable-choice selectors, or ambient real-completeness principles.

Theorem 6911.2 (Poincare–Bendixson trap NameCert obligations)**.** *For every accepted* $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ *carrier $T = (O, D, K, F, B, I, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PoincareBendixsonTrap}^{\uparrow}}$ *surface consists exactly of displayed ODE admission, local dynamical-window admission, compact metric trap exposure, continuous-map flow-window exposure, inward-boundary ledger visibility, invariant-window routing,* $\mathsf{Real}^{\uparrow}$ *seal separation, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6911.1. The rows $O, D, K, F, B, I, R, H, C, P, N$ are the only data visible to the local certificate, so displayed ODE and dynamics reads arrive before any trap conclusion can be consumed.

Read the finite trap rows next. The compact metric row K gives the displayed carrier, the continuous-map row F gives the local flow windows, and the inward-boundary ledger B records the boundary-side checks used to refuse exits.

The invariant-window row I is downstream of those reads. It can only state that the displayed local windows re-enter the compact trap through the rows already named by O, D, K, F, B, R.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve the same route, and no carrier coordinate supplies a selected host trajectory, global flow completion, quotient path equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6911.3 (Poincare–Bendixson compact-flow handoff)**.** *Every compact-flow consumer of an accepted* $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ *packet must first read the route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The inward-boundary and invariant-window rows may then be consumed as a trapping-region certificate for planar dynamics, but only through the displayed finite rows of Definition 6911.1. The handoff exports no selected trajectory, global flow completion, quotient path equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the ODE and dynamics rows. The row O names the displayed planar equation, while D names the local flow-window discipline; neither row carries a completed host trajectory.

Use the continuous-map and compact metric rows. The flow-window read F is interpreted only on the compact trap carrier K, so the consumer receives a finite compact-flow route rather than a global flow object.

Read the boundary and invariant-window rows after that route. The ledger B records the inward boundary tests, and I records the trap re-entry surface reached by the accepted local reads.

The terminal real-coordinate seal R and the structural rows H, C, P, N preserve this order. Since all public reads are projections of the displayed carrier, no hidden selected trajectory, quotient path equality, countable-choice selector, global completion, or ambient real-completeness theorem is available. □

Falsifiable prediction: Every accepted $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ read exposes one OdeUp row, one DynSystemUp local flow-window row, one CompactMetricUp trap row, one ContinuousMapUp flow row, one inward-boundary ledger, one invariant-window row, one RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any compact-flow consumer reads the trap.

Independence witness: ode, dynsystem, compactmetric, continuousmap, real: none is bijective with $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ because the packet jointly stores the displayed planar ODE, local flow windows, compact trap carrier, continuous-map reads, inward-boundary ledger, invariant-window route, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareBendixsonTrap}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareBendixsonTrap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6911.1, Definition 6911.2, and Definition 6911.3 seed the finite planar trapping-region certificate over displayed ODE, local dynamics, compact metric, continuous-map, boundary, invariant-window, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareBendixsonTrap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host trajectories, global flow completion, quotient path equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked ODE admission, local dynamics admission, compact trap exposure, continuous flow-window exposure, inward-boundary stability, invariant-window routing, Real seal separation, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{PoincareBendixsonTrap}^{\uparrow}$ packet is generated from an OdeUp row, DynSystemUp local flow-window row, CompactMetricUp trap row, ContinuousMapUp row, inward-boundary ledger, invariant-window row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6912

A Concrete Naming Certificate for CompactContinuousImageDiameter$^{\uparrow}$

Following Heine–Cantor's compact finite-net proof in the compact-uniform-continuity vision of chapter 9237, the CompactContinuousImageDiameter$^{\uparrow}$ packet records the finite image-diameter consumer that reads compact nets, pointwise continuity rows, a uniform modulus, metric-distance comparisons, and a terminal real seal before exporting a bounded-image estimate. It sits after the compact metric source of chapter 129, the continuous-map row of chapter 130, the metric-distance carrier of chapter 57, the uniform-modulus consumer of chapter 369, and the real-name boundary of chapter 38. The packet names a finite image-diameter read only; it does not choose a host finite subcover, select a global modulus, quotient functions, invoke countable choice, or import ambient Real$^{\uparrow}$ completeness.

Definition 6912.1 (Compact continuous image-diameter carrier)**.** *A* CompactContinuousImageDiameter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, U, M, A, D, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, including the finite net windows inspected by the packet.*
2. *F is the* ContinuousMap$^{\uparrow}$ *row, exposing the displayed function graph and pointwise continuity reads on those net windows.*
3. *U is the* UniformModulus$^{\uparrow}$ *row consumed after compact-net and pointwise-continuity data have been displayed.*
4. *M is the* Met$^{\uparrow}$ *distance row for target-space diameter comparisons.*
5. *A is the finite image-sample ledger, recording the displayed net centers, their image reads, and the pairwise distance windows inspected by the packet.*
6. *D is the image-diameter bound row, recording the rational or real-sealed upper diameter requested by downstream compact-image consumers.*
7. *R is the* Real$^{\uparrow}$ *seal boundary for the numeric diameter and distance comparisons.*
8. *H records componentwise* hsame *transport for compact source, continuous map, uniform modulus, metric distance, image samples, diameter bound, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-net requests through pointwise map reads, uniform-modulus routing, metric-distance comparison, image-diameter bounding, and real sealing.*
10. *P records* Pkg *provenance over* $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageDiameter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, M, A, D, R, H, C, P, N$. It has no coordinate for a selected host subcover, a chosen global modulus outside the displayed row, quotient equality of functions, countable choice, or ambient real completeness.

Theorem 6912.2 (Compact continuous image-diameter NameCert obligations)**.** *For every accepted* $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$ *carrier $G = (K, F, U, M, A, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageDiameter}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure, continuous-map exposure, uniform-modulus visibility, metric-distance comparison, finite image-sample ledger visibility, image-diameter bound separation,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6912.1. The compact source row K and continuous-map row F expose the finite net windows and their image reads before a diameter bound is visible.

Read the modulus and metric rows next. The row U is the uniform-modulus consumer produced from the compact finite-net route, and M is the metric-distance row used to compare displayed image samples.

The rows A, D, R package the finite image ledger, the diameter upper-bound row, and the terminal real seal. They are downstream of K, F, U, M, so the packet cannot replace the finite image read by a host compactness theorem.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same rows. Since the carrier has no other coordinate, the NameCert obligations are exactly the displayed finite-net, modulus, metric, image-diameter, and real-seal obligations. □

Theorem 6912.3 (Compact continuous image-diameter uniform bound)**.** *Every bounded-image consumer of an accepted* $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Only after that route is displayed may the consumer inspect the finite image-sample ledger and the image-diameter bound row. The handoff exports a reusable compact-image diameter estimate without selected finite subcovers, hidden global moduli, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the compact metric source. The row K supplies only displayed finite net windows, so a consumer receives a finite compact source rather than an open-cover compactness oracle.

Use the continuous-map and uniform-modulus rows. The image reads in F are tied to the displayed net windows, and U gives the tolerance route consumed by the diameter comparison.

Apply the target metric and real seal. The metric row M compares the displayed image samples, while D and R record the diameter bound and its real-name boundary.

The structural rows H, C, P, N then preserve this finite route for downstream compact-image consumers. No selected subcover, host modulus, quotient function equality, countable-choice selector, or ambient real-completeness principle is introduced. □

Falsifiable prediction: Every accepted $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$ read displays one CompactMetricUp finite-net source, one ContinuousMapUp image-read row, one UniformModulusUp row, one MetricUp distance row, one finite image-sample ledger, one image-diameter bound row, one RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric, real: none is bijective with $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$ because the packet jointly stores compact finite-net windows, continuous image reads, uniform-modulus routing, metric-distance comparisons, finite image samples, a diameter bound, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuousImageDiameter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContinuousImageDiameter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6912.1, Definition 6912.2, and Definition 6912.3 seed the finite compact-image diameter packet over compact-net, continuous-map, uniform-modulus, metric-distance, image-ledger, bound, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContinuousImageDiameter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected finite subcovers, hidden global moduli, quotient equality of functions, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source exposure, continuous-map exposure, uniform-modulus visibility, metric-distance comparison, image-sample ledger visibility, image-diameter bound separation, Real seal routing, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{CompactContinuousImageDiameter}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp image reads, UniformModulusUp rows, MetricUp distance rows, a finite image-sample ledger, an image-diameter bound, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6913

A Concrete Naming Certificate for $\mathsf{LocatedRationalInterval}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of located intervals and regular rational approximation, the $\mathsf{LocatedRationalInterval}^{\uparrow}$ packet records the finite interval-location object used on the L10 real-completion path. It sits between the dyadic tolerance core of chapter 291, the rational interval carrier of chapter 354, the located endpoint carrier of chapter 394, the stream-window interface of chapter 126, the regular rational readback of chapter 301, the Real seal of chapter 38, and the interval-domain consumer of chapter 1543. The chapter names only a finite location certificate for rational intervals; it is not a quotient-cut construction, endpoint selector, trichotomy principle, host real equality, or ambient completeness theorem.

Definition 6913.1 (LocatedRationalInterval carrier)**.** *A* $\mathsf{LocatedRationalInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, L, D, W, Q, E, H, C, P, N)$$

with the following visible rows:

1. *R is the* $\mathsf{RationalInterval}^{\uparrow}$ *row carrying rational endpoint order, containment, and rational width data.*
2. *L is the located-endpoint row inherited from* $\mathsf{LocatedInterval}^{\uparrow}$*, binding each endpoint read to a displayed location witness.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare rational widths and endpoint windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite endpoint requests are read.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the same endpoint windows with regular rational observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, L, D, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the rational interval, located endpoints, dyadic tolerance, stream window, regular readback, and Real seal rows.*
8. *C records displayed* Cont *replay for the same row order.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRationalInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRationalInterval}^{\uparrow}}$ *evidence row.*

The classifier compares packets only by these displayed finite rows and their transports. No carrier coordinate supplies quotient cuts, selected endpoints, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6913.2 (LocatedRationalInterval NameCert obligations)**.** *For an accepted* $\mathsf{LocatedRationalInterval}^{\uparrow}$ *carrier* $I = (R, L, D, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRationalInterval}^{\uparrow}}$ *surface consists exactly of carrier admission, located-endpoint stability, dyadic tolerance stability, stream-window readback, Real-seal non-escape, componentwise transport, displayed replay, provenance, and local naming. Every interval-location read must display the rational interval row, located endpoints, dyadic tolerance,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before a completion-facing consumer can inspect the packet.*

Proof. Carrier admission is projection from Definition 6913.1. The tuple lists $R, L, D, W, Q, E, H, C, P, N$ as the complete visible surface of the packet.

Located-endpoint stability reads R, L, H, C. The rational interval row supplies endpoint order and containment, while the located endpoint row supplies the displayed location witnesses transported with the same interval read.

Dyadic tolerance stability and stream-window readback read D, W, Q. The dyadic row supplies the finite tolerance budget for endpoint windows, the stream row names the finite requests, and the regular rational row supplies the readbacks for those same requests.

The Real-seal clause follows because E is downstream of R, L, D, W, Q and the structural rows. Since the carrier has no quotient-cut, endpoint-selector, trichotomy, host-equality, or completeness coordinate, none of those principles is exported by the local NameCert obligations. □

Theorem 6913.3 (LocatedRationalInterval Real seal handoff)**.** *Let* $I = (R, L, D, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedRationalInterval}^{\uparrow}$ *carrier. Any* $\mathsf{IntervalDomain}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer of the seal row* E *must first read the finite route*

$$\mathsf{RationalInterval}^{\uparrow}, \mathsf{LocatedInterval}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameC}$$

The handoff is a displayed interval-location route over rational endpoints and regular readbacks; it exports no quotient cuts, selected endpoints, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 6913.2. They force a completion-facing read to expose $R, L, D, W, Q, E, H, C, P, N$ before the seal can be consumed.

Read R and L through the sibling rational and located interval routes. The rational interval row supplies endpoint order and width data, while the located interval row supplies the finite endpoint-location witnesses.

Read D, W, Q next. The dyadic tolerance row supports the displayed endpoint windows, the $\mathsf{StreamName}^{\uparrow}$ row names those requests, and the $\mathsf{RegSeqRat}^{\uparrow}$ row gives the regular rational readback used by the downstream completion route.

Only then can the consumer inspect E. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the handoff remains available to $\mathsf{IntervalDomain}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ without adding quotient cuts, selected endpoints, trichotomy, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRationalInterval}^{\uparrow}}$ **.**

Constructive story: A LocatedRationalInterval$^{\uparrow}$ packet is generated from a RationalIntervalUp row, located endpoint rows, a DyadicRatCoreUp tolerance row, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{LocatedRationalInterval}^{\uparrow}}$, seed)

Scope closed: Definition 6913.1, Definition 6913.2, and Definition 6913.3 seed the finite located rational interval route for L10 completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{LocatedRationalInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient cuts, selected endpoints, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-endpoint stability, dyadic tolerance stability, stream-window readback, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRationalInterv

Chapter 6914

A Concrete Naming Certificate for $\mathsf{ContractiveSequence}^{\uparrow}$

Following Bishop and Bridges on contractive iteration, the $\mathsf{ContractiveSequence}^{\uparrow}$ packet records the finite route by which contraction-stage budgets become displayed regular rational sequence data before any Real-facing seal is consumed. It sits between the interval-contraction surface of chapter 5867, the contraction and iteration sources of chapter 3529 and chapter 2359, the dyadic budget ledger of chapter 291, the finite window source of chapter 126, the regular rational readback of chapter 301, and the terminal seal boundary of chapter 38. The packet names a finite sequence certificate; it is not a selected fixed point, a quotient stream, countable choice, an endpoint existence theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6914.1 (Contractive sequence carrier)**.** *A* $\mathsf{ContractiveSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source interval or metric-domain row read from* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *and the surrounding contraction setup.*
2. *F is the contraction or Picard-step row, read through* $\mathsf{BanachContraction}^{\uparrow}$ *or* $\mathsf{PicardIteration}^{\uparrow}$ *as finite BEDC data.*
3. *L is the stage-budget ledger recording the contraction depth, retained prefix, and shrinkage request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the contraction budgets and error windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected prefixes of the sequence.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, stage-budget, dyadic, stream, regular-rational, Real-seal, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a contractive-sequence request through the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{ContractiveSequence}^{\uparrow}$, $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContractiveSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, D, W, R, E, H, C, P, N$. It has no coordinate for a selected fixed point, selected limit, quotient stream equality, countable choice, endpoint existence, host iteration equality, or ambient real completeness.

Theorem 6914.2 (Contractive sequence NameCert obligations)**.** *For every accepted* $\mathsf{ContractiveSequence}^{\uparrow}$ *carrier*

$$S = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContractiveSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval or domain row, contraction-step row, stage-budget ledger, dyadic tolerance ledger, finite windows, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows. Every completion-facing consumer must read contraction stability through I, F, L, D, finite prefix exposure through W, regular rational handoff through R, and the structural route E, H, C, P, N before treating the sequence as Real-facing data.*

Proof. Project the carrier of Definition 6914.1. The tuple lists the complete finite inventory available to a contractive-sequence consumer, beginning with the source interval or domain row and the displayed contraction-step row.

Read the stage budget through L, D. The contraction depth and dyadic tolerance ledger are part of the packet, so a consumer cannot replace them by a host convergence theorem or by an implicit infinite schedule.

Next use the sibling routes. The interval-contraction certificate of chapter 5867 supplies the shrinkage discipline, while chapter 126 and chapter 301 expose only finite windows and regular rational readback.

Finally the rows E, H, C, P, N seal, transport, replay, source, and name the same route. Since no row stores a selected fixed point, quotient stream, countable-choice schedule, endpoint theorem, or ambient completeness principle, the listed obligations exhaust the local certificate surface. □

Theorem 6914.3 (Contractive sequence RegSeqRat handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{ContractiveSequence}^{\uparrow}$ *packet factors through*

$$\mathsf{BishopIntervalContraction}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $I, F, L, D, W, R, E, H, C, P, N$. The handoff may use $\mathsf{BanachContraction}^{\uparrow}$ *or* $\mathsf{PicardIteration}^{\uparrow}$ *to read the contraction-step row F, but it cannot claim a selected limit, quotient stream equality, endpoint existence, countable choice, a completed fixed point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted packet. The contraction data is read through I, F, L, while the dyadic tolerance budget is read through D before any finite sequence window is exposed.

Use the finite-window route through W. Each inspected prefix is a displayed $\mathsf{StreamName}^{\uparrow}$ read, not an implicit infinite stream or a selected tail.

Use the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R and then the terminal $\mathsf{Real}^{\uparrow}$ seal row E. This places the regular rational handoff after the same contraction and dyadic budget rows named by the carrier.

The structural rows H, C, P, N preserve the route for downstream consumers. Thus the handoff produces a finite RegSeqRat-to-Real seal path and leaves selected limits, quotient streams, countable choice, fixed-point existence, and ambient Real completeness outside the packet. □

Falsifiable prediction: An accepted $\mathsf{ContractiveSequence}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without displaying $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: bishop-interval-contraction, banach-contraction, picard-iteration, dyadicratcore, streamname, regseqrat, real: no listed sibling by itself supplies the joint packet containing contraction-stage budgets, dyadic tolerance rows, finite prefix windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ContractiveSequence}^{\uparrow}$ packet is generated from a source interval or domain row, a contraction-step row, a stage-budget ledger, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6914.1, Definition 6914.2, and Definition 6914.3 seed the finite contraction-budget route from displayed iteration data to regular rational and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, selected limits, quotient streams, countable choice, endpoint existence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-stage budget exposure, dyadic tolerance exactness, finite-window readback, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.ContractiveSequenceUp.

Chapter 6915

A Concrete Naming Certificate for $\mathsf{DyadicSignedDigitStream}^{\uparrow}$

Following Bishop–Bridges signed-digit real-name practice, the $\mathsf{DyadicSignedDigitStream}^{\uparrow}$ packet records a finite stream interface in which each observed dyadic digit is signed, locally normalized, and routed toward the regular rational and Real-name layers without quotienting streams. It sits beside the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a signed-digit real-name carrier, not a host real, selected limit, quotient stream class, trichotomy oracle, or ambient completeness theorem.

Definition 6915.1 (Dyadic signed-digit stream carrier)**.** *A* $\mathsf{DyadicSignedDigitStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, A, B, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the signed dyadic digit stream window row, listing the inspected positions and their digits in $\{-1, 0, 1\}$.*
2. *A is the digit-admissibility row, checking that every displayed digit lies in the signed dyadic alphabet and is attached to the requested dyadic place.*
3. *B is the carry and normalization budget row, recording the finite carry radius and local overlap bounds used to compare neighbouring windows.*
4. *M is the* $\mathsf{StreamName}^{\uparrow}$ *window row that schedules the same finite observation as a stream-name request.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reading the normalized signed-window prefix as a regular rational approximation.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, A, B, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for W, A, B, M, Q, E and the local naming row.*
8. *C records displayed* Cont *replay from signed dyadic windows through normalization, stream-name scheduling, regular rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicSignedDigitStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicSignedDigitStream}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, A, B, M, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6915.2 (Dyadic signed-digit stream NameCert obligations)**.** *For every accepted* DyadicSignedDigitStream *carrier $S = (W, A, B, M, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicSignedDigitStream}^{\uparrow}}$ *surface consists exactly of signed-window admission, digit-admissibility checking, carry and normalization budget exposure,* $\mathsf{StreamName}^{\uparrow}$ *window scheduling,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6915.1. The signed-window row W, admissibility row A, carry budget row B, stream-name row M, regular readback row Q, and Real seal row E are the only analytic coordinates of an accepted packet.

The normalization budget is finite. It records how signed digits may carry across neighbouring displayed places before the window is read as a regular rational approximation, so the handoff cannot bypass A or B.

The structural rows H, C, P, N transport, replay, source, and name exactly the same finite route. Since the carrier contains no quotient stream, selected representative, host real equality, trichotomy, countable-choice schedule, or ambient completeness coordinate, none of those endpoints belongs to the local NameCert surface. □

Theorem 6915.3 (Dyadic signed-digit stream RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicSignedDigitStream}^{\uparrow}$ *packet must first expose the signed dyadic window W, digit-admissibility row A, carry and normalization budget row B, and* $\mathsf{StreamName}^{\uparrow}$ *window row M. The resulting handoff reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *through Q and reaches* $\mathsf{Real}^{\uparrow}$ *through E only as the finite route*

$$W, A, B, M \longmapsto Q \longmapsto E, H, C, P, N.$$

It is not a quotient-stream construction, selected host limit, host real equality, trichotomy principle, countable-choice route, or ambient Real completeness theorem.

Proof. Start with the NameCert obligations of Definition 6915.2. They force every consumer-facing read to expose the signed-window, admissibility, normalization-budget, stream-window, readback, seal, transport, replay, provenance, and naming rows.

The regular rational handoff is downstream of normalization. A displayed signed digit may contribute to neighbouring dyadic places only through the finite budget B, so the readback row Q is read from W, A, B, M together, not from a raw stream of host digits.

Finally the Real seal row E is consumed after Q. Transport and replay preserve the same route through H and C, with P and N recording provenance and local naming. The excluded endpoints require coordinates absent from the packet and cannot be exported by this handoff. □

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicSignedDigitStream}^{\uparrow}$ is not bijective to any listed sibling because it stores signed dyadic windows, digit admissibility, a carry and normalization budget, stream-window scheduling, regular rational readback, terminal Real sealing, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSignedDigitStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSignedDigitStream}^{\uparrow}$ packet is generated from signed dyadic digit windows, digit-admissibility checks, carry and normalization budgets, StreamName windows, RegSeqRat handoff rows, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicSignedDigitStream}\uparrow}$, seed)

Scope closed: Definition 6915.1, Definition 6915.2, and Definition 6915.3 seed the finite signed-digit stream route from dyadic windows to RegSeqRat and Real consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicSignedDigitStream}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed dyadic signed-digit stream route.

Chapter 6916

A Concrete Naming Certificate for $\mathsf{CofinalCauchyFilter}^{\uparrow}$

Following Kelley and Bishop–Bridges on filters and constructive Cauchy approximation, the $\mathsf{CofinalCauchyFilter}^{\uparrow}$ packet records the finite BEDC route by which a filter-style Cauchy carrier is read through cofinal finite windows rather than through a quotient filter or a selected limit. It sits after the Cauchy carrier of chapter 3413, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the completion boundary of chapter 95, and the terminal real-name seal of chapter 38. The packet lets Completion and Real consumers ask for cofinal Cauchy evidence without importing host filter equality, countable choice, selected limits, or ambient Real completeness.

Definition 6916.1 (Cofinal Cauchy filter carrier)**.** *A* $\mathsf{CofinalCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, S, R, Q, W, E, H, A, P, N)$$

with the following rows:

1. *C is the* $\mathsf{Cauchy}^{\uparrow}$ *source row exposing the Cauchy request interface.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect named stream segments.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for tolerance comparisons.*
4. *Q is the cofinal refinement row, recording which later windows refine a requested Cauchy witness.*
5. *W is the finite-window projection row, sending a filter request to an inspected stream window and a rational tolerance.*
6. *E is the completion-facing exactness row, exposed only after C, S, R, Q, W are visible.*
7. *H records componentwise* hsame *transport for the Cauchy, stream, rational, cofinal, projection, and exactness rows.*
8. *A records displayed* Cont *replay of the same finite route.*
9. *P is the* Pkg *provenance over* $\mathsf{CofinalCauchyFilter}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchyFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, host equality of filters, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6916.2 (Cofinal Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{CofinalCauchyFilter}^{\uparrow}$ *carrier* $F = (C, S, R, Q, W, E, H, A, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchyFilter}^{\uparrow}}$ *surface consists exactly of Cauchy source exposure through* C*, stream-window exposure through* S*, regular rational readback through* R*, cofinal refinement through* Q*, finite-window projection through* W*, completion-facing exactness through* E*, componentwise transport through* H*, displayed replay through* A*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The source row C supplies the Cauchy request, while S and R expose the finite stream and rational tolerance data on which the request is inspected.

The cofinal refinement row Q records how later windows answer the same request, and W projects each filter-style demand to the finite window actually read by the packet. The exactness row E is therefore reached through finite-window projection and cofinal refinement, not through a quotient filter.

The structural rows H, A, P, N transport, replay, source, and name that same route. Since no further coordinate is present, the local NameCert cannot export selected limits, countable choice, host filter equality, or ambient Real completeness. □

Theorem 6916.3 (Cofinal Cauchy filter RegSeqRat handoff)**.** *Every Completion or Real consumer of an accepted* $\mathsf{CofinalCauchyFilter}^{\uparrow}$ *carrier must read the regular rational handoff through the displayed route*

$$C, S, R, Q, W, E \longmapsto H, A, P, N.$$

Consequently a cofinal Cauchy filter request reaches $\mathsf{RegSeqRat}^{\uparrow}$ *only after the StreamName finite window and cofinal refinement rows are visible, and it reaches* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *only through the completion-facing exactness row.*

Proof. Begin with the Cauchy source row C and the stream window row S. They determine which finite segment of the name is inspected for the current request.

The regular rational row R then supplies the tolerance comparison. Cofinal refinement through Q and projection through W ensure that later filter-style requests reuse a displayed finite-window route rather than a host-level filter representative.

Only after those rows are exposed may the exactness row E hand data to Completion or Real consumers. The rows H, A, P, N preserve the same handoff under transport, replay, provenance, and local naming, so the consumer obtains no bypass around StreamName, RegSeqRat, cofinal refinement, or the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalCauchyFilter}^{\uparrow}$ packet is generated from a Cauchy source row, StreamName finite windows, RegSeqRat readback, cofinal refinement, finite-window projection, completion-facing exactness, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 6916.1, Definition 6916.2, and Definition 6916.3 seed the cofinal Cauchy-filter bridge for StreamName, RegSeqRat, Completion, and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofinalCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal refinement, finite-window projection, RegSeqRat handoff, completion exactness, transport, replay, provenance, and local naming for BEDC.Derived.CofinalCauchyFilterUp.

Chapter 6917

A Concrete Naming Certificate for $\mathsf{MetaCICDecidableFragment}^{\uparrow}$

Following the Tait, Girard, and Coquand tradition of bounded decidability arguments around typed calculi, the $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ packet records the finite fragment in which closed checking, bounded normal comparison, and endpoint refusal are displayed together before a decidability-facing consumer reads any result. It sits beside the decidability boundary of chapter 749, the closed checker seal of chapter 2182, the bounded normal-equality checker of chapter 752, and the normalization frontier of chapter 892. The packet names a finite decidable fragment, not global conversion decidability, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 6917.1 (MetaCIC decidable fragment carrier)**.** *A* $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, E, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed-term input row, listing the finite typed closed terms and the bounded query being inspected.*
2. *S is the checker-seal row, read through* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *and recording the accepted closed checker output.*
3. *E is the bounded normal endpoint row, read through* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *and recording the finished endpoint available at the displayed bound.*
4. *Q is the comparison-result row, read through* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *and recording the bounded normal comparison output.*
5. *R is the refusal-boundary row, naming global conversion decidability, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, and kernel consistency as unavailable endpoints.*
6. *H records componentwise* hsame *transport for I, S, E, Q, R and the local naming row.*
7. *C records displayed* Cont *replay from the closed-term input through checker sealing, bounded endpoint comparison, refusal retention, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{MetaCICDecidableFragment}^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableFragment}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, E, Q, R, H, C, P, N$. It has no coordinate for a global closed conversion decider, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Theorem 6917.2 (MetaCIC decidable fragment NameCert obligations)**.** *For every accepted* MetaCICDecidableFrag *carrier $F = (I, S, E, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableFragment}^{\uparrow}}$ *surface consists exactly of closed-term input exposure, checker-seal exposure, bounded normal endpoint exposure, comparison-result exposure, refusal-boundary retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6917.1. The rows I, S, E, Q, R are the complete decidability-facing data, and H, C, P, N are the structural rows that transport, replay, source, and name that data.

The checker and comparison rows are bounded. The checker seal S accepts only the displayed closed input, the endpoint row E records only the finished normal data available at the bound, and the comparison row Q records only the bounded comparison result.

The refusal row R remains part of the same packet. Since no carrier coordinate contains global conversion decidability, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency, those endpoints cannot be exported by the local NameCert surface. □

Theorem 6917.3 (MetaCIC decidable fragment closed-checker handoff)**.** *Every decidability-facing consumer of an accepted* $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ *packet must read the closed-term input row I, checker-seal row S, bounded normal endpoint row E, comparison-result row Q, and refusal-boundary row R before it consumes a fragment decision. The handoff is the finite route*

$$I \longmapsto S \longmapsto E, Q, R, H, C, P, N.$$

It is a bounded closed-checker handoff and not a global conversion decision procedure, subject-reduction theorem, evaluator equality, quotient syntax equality, or kernel-consistency certificate.

Proof. Start with the NameCert obligations of Definition 6917.2. They force a consumer to expose the closed input, checker seal, bounded endpoint, comparison result, refusal boundary, transport, replay, provenance, and naming rows before reading the fragment decision.

The closed checker handoff is then ordered by the carrier. The input row I is checked by S, the bounded endpoint row E supplies the normal form surface at the displayed bound, and Q records the comparison result under the retained refusal row R.

Finally H, C, P, N preserve the same finite route. A consumer that wants global conversion decidability, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency must supply a separate certificate outside this packet. □

Theorem 6917.4 (MetaCIC decidable fragment bounded-normal refusal)**.** *An accepted* MetaCICDecidableFragment *packet may consume bounded normal endpoint rows and bounded comparison rows through E and Q, after the closed input I and checker seal S have been displayed, but it exports no arbitrary-fuel normalizer, no global conversion decider, no subject-reduction theorem, no evaluator equality, no*

quotient syntax equality, and no kernel-consistency certificate. The refusal row R remains part of the public finite route

$$I, S, E, Q, R, H, C, P, N.$$

Proof. Project the carrier $F = (I, S, E, Q, R, H, C, P, N)$ of Definition 6917.1. The rows E and Q are named only as bounded normal endpoint and bounded comparison data, and the preceding rows I and S keep the read tied to the displayed closed input and checker seal.

The structural rows H, C, P, N transport, replay, source, and name this same finite route. They do not add a coordinate for arbitrary fuel, global conversion, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

The refusal row R is therefore not auxiliary prose; it is one of the displayed carrier rows. Any consumer that reads E or Q must retain R with the same package provenance, so the bounded-normal surface cannot be reinterpreted as an unrestricted conversion procedure. □

Theorem 6917.5 (MetaCIC decidable fragment gallery handoff)**.** *Gallery sanity rows attached to a* $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ *packet are audit-facing examples of the finite route $I, S, E, Q, R, H, C, P, N$. They may witness that the bounded checker, endpoint, comparison, and refusal rows are populated in displayed cases, but they are not the source of general typing completeness, global conversion decidability, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Read the gallery row only through the carrier of Definition 6917.1. A gallery example can expose a closed input I, a checker seal S, a bounded endpoint E, a comparison result Q, the refusal row R, and the structural rows H, C, P, N.

That exposure is example-facing rather than theorem-generating. The NameCert obligations of Definition 6917.2 already identify the complete public surface, and Definition 6917.4 keeps the bounded-normal read from escaping its refusal boundary.

Thus a gallery handoff can support audit rows for particular closed checks, but no gallery row supplies a new coordinate for general typing completeness or for any endpoint named in R. □

Independence witness: metacicdecidableboundary, closed-type-checker-seal, bounded-normal-equality-checker, metacicnormalizationfrontier: $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores closed-term input, checker seal, bounded normal endpoint, comparison result, refusal boundary, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableFragment}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableFragment}^{\uparrow}$ packet is generated from closed-term input rows, closed checker seal rows, bounded normal endpoint rows, comparison-result rows, refusal-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableFragment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6917.1, Definition 6917.2, and Definition 6917.3 seed the finite bounded decidable fragment for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableFragment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global conversion decidability, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed MetaCIC decidable fragment route.

Chapter 6918

A Concrete Naming Certificate for $\mathsf{RealCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on adjunctions, the $\mathsf{RealCompletionAdjunction}^{\uparrow}$ packet records the finite unit/counit surface by which dyadic, stream, regular-rational, and real-name rows support a completion-facing universal route. It sits after the dyadic tolerance core of chapter 291, the finite stream-name discipline of chapter 126, the regular-rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names Real completion as displayed unit and counit data rather than as quotient completion equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or open-cover compactness.

Definition 6918.1 (Real completion adjunction carrier)**.** *A* $\mathsf{RealCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, Q, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording precision cells, refinement slack, and comparison budgets used by the completion request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger, listing the source windows inspected before any real-name seal or completion endpoint is read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, exposing the regular rational data consumed by the completion route.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
5. *U is the unit row, routing the displayed rational or regular-rational source into the Real-completion surface without choosing a quotient representative.*
6. *E is the counit and extension row, recording the finite handoff from a completion-facing consumer back through the same* $\mathsf{Real}^{\uparrow}$ *seal and readback data.*
7. *H records componentwise* hsame *transport for the dyadic, stream, readback, Real seal, unit, counit, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, S, Q \longmapsto R \longmapsto U, E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionAdjunction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionAdjunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, R, U, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, a host Cauchy-real quotient, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6918.2 (Real completion adjunction NameCert obligations)**.** *For every accepted* RealCompletionAdjuncti *carrier*

$$A = (D, S, Q, R, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of dyadic tolerance exposure through D, stream-window exposure through S, regular-rational readback through Q, terminal Real seal handoff through R, unit exposure through U, counit and extension exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The projection exposes the dyadic tolerance ledger, finite stream windows, regular-rational readback, Real seal, unit, counit, transport, replay, provenance, and naming rows.

The completion-facing obligations are read in that order. The dyadic ledger fixes the precision budget, the stream ledger fixes the inspected finite windows, and the regular-rational readback is the only route by which the terminal Real seal is reached.

The unit and counit rows then record the finite adjunction surface over the same data. Componentwise transport and displayed replay preserve the route, while provenance and the local NameCert row bind it as one packet. Since no quotient representative, selected limit, open-cover compactness row, or ambient completeness theorem occurs in the carrier, the listed rows exhaust the local obligation surface. □

Theorem 6918.3 (Real completion adjunction unit-counit scope)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealCompletionAdjunction}^{\uparrow}$ *packet reads the unit and counit only after the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

has been displayed through D, S, Q, R, H, C, P, N. The handoff supplies finite Real-completion adjunction data; it does not supply quotient completion equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, host real equality, or countable choice.*

Proof. Start with the source rows D, S, Q. They provide the finite tolerance, window, and regular-rational readback data needed before a Real-completion request is admitted.

Next read the terminal seal R. The $\mathsf{Real}^{\uparrow}$ row is downstream of those source rows, so a consumer cannot detach the unit or counit from the dyadic, stream, and readback ledgers.

Finally inspect U and E with H, C, P, N. These rows bind the unit and counit to the same displayed carrier and replay route. No excluded quotient, selected-limit, compactness, choice, host-equality, or ambient-completeness coordinate is present, so the unit-counit scope is exactly the finite NameCert surface named above. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletionAdjunction}^{\uparrow}$ consumer reads dyadic tolerance, finite stream windows, regular-rational readback, Real seal, unit, counit, hsame transport,

Cont replay, Pkg provenance, and local NameCert rows before using a Real-completion adjunction endpoint.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RealCompletionAdjunction}^{\uparrow}$ is not bijective to any listed sibling because it binds the four L10 faces to explicit unit and counit rows with transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionAdjunction}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionAdjunction}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, unit row, counit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6918.1, Definition 6918.2, and Definition 6918.3 seed the finite Real-completion adjunction packet over DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionAdjunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected limits, ambient RealUp completeness, open-cover compactness, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, stream-window exposure, regular-rational readback, Real seal handoff, unit and counit exactness, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionAdjunctionUp.

Chapter 6919

A Concrete Naming Certificate for $\mathsf{BishopRegularInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularInterval}^{\uparrow}$ packet records a regular interval by finite rational endpoint data and explicit tolerance control, not by a quotient of intervals or a choice-selected limiting endpoint. It sits on the L10 route through dyadic rational rows, stream-name windows, regular rational readback, the terminal real-name seal, and located interval containment. The packet is a constructive interval carrier for downstream completion arguments.

Definition 6919.1 (Bishop regular interval carrier)**.** *A* $\mathsf{BishopRegularInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, S, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the dyadic endpoint-and-tolerance row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, containing two rational endpoint windows and a visible tolerance budget.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records which endpoint and tolerance windows have been inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational endpoint names and their common tolerance schedule.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the endpoint readbacks supplied by R.*
5. *L is the located interval containment row, read through chapter 6913, recording which displayed rational subinterval contains the accepted endpoint windows.*
6. *H stores componentwise* hsame *transport for D, S, R, E, L.*
7. *C is the displayed* Cont *replay of the same finite interval evidence.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LocatedRationalInterval}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient interval equality, a selected real endpoint, trichotomy of endpoints, ambient real completeness, or countable choice.

Theorem 6919.2 (Bishop regular interval NameCert obligations)**.** *For the carrier of Definition 6919.1, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for* $I = (D, S, R, E, L, H, C, P, N)$*, dyadic endpoint-and-tolerance coverage by* D*, finite window coverage by* S*, regular rational readback by* R*, terminal real sealing by* E*, located interval containment by* L*, componentwise transport by* H*, displayed replay by* C*, package provenance by* P*, and local naming by* N*.*

Proof. Project the packet. The dyadic row D supplies the only endpoint and tolerance data, and the stream row S records which finite windows are visible to the consumer.

Read the rational endpoint names through R and then through the terminal real seal E. This is the same L10-facing route used by chapter 301, chapter 126, and chapter 38.

Use the located interval row L to bind the endpoint windows to the displayed containment surface of chapter 6913. The transport, replay, provenance, and naming rows H, C, P, N carry those same rows and do not add a second interval object.

Since the carrier has no further coordinate, all NameCert obligations are exhausted by the displayed packet. □

Theorem 6919.3 (Bishop regular interval L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{BishopRegularInterval}^{\uparrow}$ *packet must read the interval through* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint tolerances,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, located interval containment, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BishopRegularInterval}^{\uparrow}}$ *row. The handoff exports no quotient interval, choice-selected endpoint, endpoint trichotomy, host real equality, or ambient completeness theorem.*

Proof. Start with the NameCert obligations of Definition 6919.2. They identify the visible route as $D, S, R, E, L, H, C, P, N$.

The numerical part of the handoff is forced through chapter 291, chapter 126, chapter 301, and chapter 38. Thus a consumer cannot skip from a rational endpoint window to an ambient real interval.

The containment part is forced through chapter 6913. It records located containment for the displayed rational interval row and supplies no quotient equality between interval representatives.

Finally H, C, P, N transport, replay, package, and name the same finite evidence. The excluded endpoint-selection and completeness readings have no carrier coordinate, so the L10 handoff is exactly the displayed constructive route. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularInterval}^{\uparrow}$ interval read must display dyadic endpoint tolerances, finite stream windows, regular rational readback, terminal Real sealing, located containment, transport, replay, provenance, and local NameCert rows before it exports interval data to a completion consumer.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-rational-interval: no listed sibling is bijective to $\mathsf{BishopRegularInterval}^{\uparrow}$ because this packet jointly carries endpoint tolerances, finite read windows, regular rational endpoint names, Real sealing, located containment, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularInterval}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{BishopRegularInterval}^{\uparrow}$ packet is generated from dyadic endpoint-and-tolerance rows, StreamName finite windows, RegSeqRat readback, Real sealing, located interval containment, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6919.1, Definition 6919.2, and Definition 6919.3 describe the finite constructive interval carrier and its L10 handoff route.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopRegularInterval}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient interval theory, real-order trichotomy, ambient completeness, or selected endpoint existence.

Upgrade path: Scoped closure requires checked carrier coverage, located containment exactness, and L10 handoff nonescape for $D, S, R, E, L, H, C, P, N$.

Chapter 6920

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$

The $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ packet realizes the observer-expansion direction of chapter 9463 as finite NameCert data. It binds boundary-shadow, visual-boundary, phase-distance, reversible transport, refusal, provenance, and local naming rows into one concrete route. The chapter is a concrete carrier site; the vision chapter remains an expressive source, not a proof surface.

Definition 6920.1 (Hyperbolic observer phase route carrier)**.** *A* $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, V, D, T, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-shadow row read through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*.*
2. *V is the visual-boundary row read through* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*.*
3. *D is the phase-distance row read through* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *and the terminal real-facing distance seal used there.*
4. *T is the reversible disk or phase transport row, read through* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *and* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*.*
5. *F is the refusal row blocking interior fixed-center readings, hidden global geometry, and unlogged expansion claims.*
6. *H stores componentwise* hsame *transport for B, V, D, T, F.*
7. *C is the displayed* Cont *replay of the observer phase route.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$*,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{HyperbolicVisualBou}$ $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an ambient hyperbolic manifold, a selected ideal boundary, a global geodesic completion, an empirical cosmology claim, or a hidden observer center.

Theorem 6920.2 (Hyperbolic observer phase route NameCert obligations)**.** *For the carrier of Definition 6920.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for* $Q = (B, V, D, T, F, H, C, P, N)$*, boundary-shadow coverage by* B*, visual-boundary coverage by* V*, phase-distance readback by* D*, reversible transport by* T*, refusal coverage by* F*, componentwise transport by* H*, displayed replay by* C*, package provenance by* P*, and local naming by* N*.*

Proof. Project the packet. The geometric input rows are B, V, D, T, F, and each is a displayed row of the finite carrier rather than a theorem about an ambient disk or manifold.

Read B through chapter 3813 and V through chapter 4181. These rows supply the boundary-facing surface before any observer expansion is consumed.

Read D through chapter 6017, then read T through chapter 3500 and chapter 3455. The refusal row F prevents those readings from becoming an interior fixed-center claim or an unlogged global-geometry assertion.

Finally H, C, P, N transport, replay, package, and name the same rows. Since the carrier has no further coordinate, the NameCert obligations are exactly the displayed route. □

Theorem 6920.3 (Hyperbolic observer phase route finite handoff)**.** *Every downstream read of an accepted* $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ *packet factors through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*,* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{PoincareDiskPhase}$ *the refusal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}}$ *row. It exports no hidden global hyperbolic space, no selected ideal boundary, no empirical observer-expansion law, and no private phase-distance oracle.*

Proof. Use the obligation theorem Definition 6920.2. It exposes the accepted route as $B, V, D, T, F, H, C, P, N$.

The boundary side is supplied only by the boundary-shadow and visual-boundary siblings. The phase-distance and reversible-transport sides are supplied only by their displayed concrete-instance siblings, so the route cannot bypass them through a private geometry reading.

The refusal row F blocks the strong interpretations: an interior observer center, selected ideal boundary, empirical cosmology claim, and hidden global hyperbolic space. The rows H, C, P, N preserve only the displayed carrier evidence for downstream consumers. □

Future Lean coverage for this packet should require Nontrivial and FieldFaithful TasteGate instances for the boundary-shadow, visual-boundary, phase-distance, reversible-transport, and refusal rows, so that no zero-row or collapsed-field carrier can satisfy the route.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ read must display boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, hsame, Cont, Pkg, and local NameCert rows before it exports an observer phase route.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-visual-boundary, phase-distance-observer-expansion, poincare-boundary-fixed-transport, poincare-disk-phase-transport: no listed sibling is bijective to $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ because this packet jointly carries boundary shadows, visual-boundary readback, phase-distance comparison, reversible transport, refusal, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ packet is generated from boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6920.1, Definition 6920.2, and Definition 6920.3 describe the finite observer phase route and its downstream handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{HyperbolicObserverPhaseRoute}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, global hyperbolic geometry, empirical cosmology, selected ideal boundaries, or observer-center existence.

Upgrade path: Scoped closure requires checked carrier coverage, Nontrivial and FieldFaithful TasteGate instances, refusal exactness, and downstream handoff nonescape for $B, V, D, T, F, H, C, P, N$.

Chapter 6921

A Concrete Naming Certificate for $\mathsf{FrobeniusCoin}^{\uparrow}$

Following Sylvester and standard numerical-semigroup treatments, the $\mathsf{FrobeniusCoin}^{\uparrow}$ packet records the two-generator coin problem as finite BEDC data. It consumes unary natural-number rows from chapter 28, addition rows from chapter 29, and the finite semigroup surface of chapter 79. The packet names a displayed conductor window, not an ambient theorem about all natural numbers, an external greatest-common-divisor algorithm, quotient semigroup equality, or an unbounded search over representations.

Definition 6921.1 (Frobenius coin carrier)**.** *A* $\mathsf{FrobeniusCoin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, B, G, C, W, H, T, P, N)$$

with the following displayed rows:

1. *A and B are positive unary* $\mathsf{Nat}^{\uparrow}$ *coin rows.*
2. *G is the displayed coprime row, recording a finite witness that the carried coin rows have greatest common divisor* 1 *when that witness is supplied.*
3. *C is the finite conductor row, recording the displayed bound at which the representability window begins.*
4. *W is the* $\mathsf{Add}^{\uparrow}$*-generated representability window, listing finite coefficient pairs and their displayed sums for the conductor-facing interval.*
5. *H records componentwise* hsame *transport for the coin rows, coprime row, conductor row, window row, replay, provenance, and local naming rows.*
6. *T records displayed* Cont *replay for the same rows.*
7. *P is the* Pkg *provenance over* $\mathsf{FrobeniusCoin}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusCoin}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, G, C, W, H, T, P, N$. It has no coordinate for host natural arithmetic, quotient semigroup equality, a hidden Euclidean algorithm, or an unbounded enumeration of all sums.

Lean4 ✓ `BEDC.Derived.FrobeniusCoinUp.FrobeniusCoinTasteGate single carrier alignment`

Theorem 6921.2 (Frobenius coin NameCert obligations)**.** *For every accepted* $\mathsf{FrobeniusCoin}^{\uparrow}$ *carrier* $F = (A, B, G, C, W, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrobeniusCoin}^{\uparrow}}$ *surface consists exactly of carrier admission for the two positive unary coin rows; the displayed coprime row; representability-window exactness for the finite* $\mathsf{Add}^{\uparrow}$ *sums; the conductor bound row; the finite nonrepresentable ledger below the conductor-facing window; componentwise transport through* H*; displayed replay through* T*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 6921.1. The rows A, B, G, C, W are the only arithmetic rows, and each is finite: two positive unary coin rows, one coprime witness row, one conductor row, and one displayed sum window.

The representability data are read through W. Each accepted entry of W supplies a finite pair of unary coefficients and an $\mathsf{Add}^{\uparrow}$ replay of the corresponding sum, while every excluded entry below the displayed conductor is accounted for by the finite nonrepresentable ledger attached to the same window.

The structural rows H, T, P, N then transport, replay, source, and name that finite inventory. Since the packet has no host arithmetic coordinate, no quotient semigroup equality coordinate, and no unbounded-search coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6921.3 (Frobenius coin conductor-window boundary)**.** *Every conductor-facing consumer of an accepted* $\mathsf{FrobeniusCoin}^{\uparrow}$ *carrier reads only the displayed finite* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*, and* $\mathsf{Semigroup}^{\uparrow}$ *rows*

$$A, B, G, C, W \longmapsto H, T, P, N.$$

In particular, the consumer receives the two positive coin rows, the copied coprime witness when present, the finite representability window, the finite below-conductor ledger, transport, replay, provenance, and local naming; it receives no host natural-number arithmetic, no quotient semigroup equality, and no unbounded search procedure.

Proof. Start with a conductor-facing request. Carrier admission forces the request to inspect A and B as positive unary $\mathsf{Nat}^{\uparrow}$ rows, not as ambient natural numbers.

The coprime and conductor information is then exposed by G and C. If the coprime row is absent, the packet does not claim the conductor theorem; if it is present, the row is still only the displayed finite witness carried by the packet.

Representability is read through W, where every accepted sum is generated by finite $\mathsf{Add}^{\uparrow}$ replay over the two coin rows and every conductor-facing refusal is recorded in the finite ledger. The $\mathsf{Semigroup}^{\uparrow}$ dependency is consumed only as this finite addition-generated surface.

Finally H, T, P, N preserve the same route under transport, replay, provenance, and local naming. No remaining carrier field can provide host arithmetic, quotient equality, or an infinite search, so every consumer is confined to the displayed conductor window. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusCoin}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrobeniusCoin}^{\uparrow}$ packet is generated from two positive unary coin rows, a displayed coprime witness row, a conductor row, a finite AddUp representability window, a finite nonrepresentable ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrobeniusCoin}^{\uparrow}}$, seed)

Scope closed: Definition 6921.1, Definition 6921.2, and Definition 6921.3 seed the finite two-generator conductor-window packet over $A, B, G, C, W, H, T, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrobeniusCoin}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a Lean proof, a standard bridge, unrestricted numerical-semigroup closure, host natural arithmetic, quotient semigroup equality, or unbounded representability search.

Upgrade path: Obligation closure requires checked carrier admission, representability-window exactness, conductor-bound coverage, below-conductor ledger, transport, replay, provenance, and local naming rows for BEDC.Derived.FrobeniusCoinUp.

Chapter 6922

A Concrete Naming Certificate for MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$

The MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$ packet realizes the subject-reduction frontier of section 9386.2 as a finite four-row discharge matrix. It sits beside the discharge socket of chapter 719, the closed application-argument route of chapter 6177, the closed Pi-domain route of chapter 3352, the closed lambda-domain route of chapter 3381, the proof-obligation gap of chapter 3460, and the open-frontier ledger of chapter 762. The matrix is a finite audit object for the four simultaneous discharge rows; it does not assert a closed subject-reduction theorem.

Falsifiable prediction: Every accepted MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$ packet exposes all four discharge rows before any subject-reduction consumer reads it.

Independence witness: `closed-app-argument-stability`, `closed-pi-domain-subject-reduction`, `closed-lam-domain-subject-reduction`, `proof-obligation-gap`, `metacic-open-frontier-ledger`: none is bijective with MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$, because this packet binds the four subject-reduction discharge rows, the obstruction ledger, matrix coverage, transport, replay, provenance, and local naming as one finite surface.

Definition 6922.1 (MetaCIC subject-reduction discharge matrix carrier)**.** *A* MetaCICSubjectReductionDischargeM *carrier is a finite* BHist *packet*

$$M = (B, A, L, P, O, S, R, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation discharge row.*
2. *A is the closed application-argument stability discharge row.*
3. *L is the closed lambda-domain subject-reduction discharge row.*
4. *P is the closed Pi-domain subject-reduction discharge row.*
5. *O is the proof-obligation gap row recording the absence of an unconditional closed theorem until the four discharge rows are supplied.*
6. *S is the* SubjectReductionDischargeSocket$^{\uparrow}$ *row through which the subject-reduction consumer reads the four rows.*
7. *R is the matrix-coverage row, recording that the accepted packet exposes B, A, L, P simultaneously and does not merge them.*

8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay for the same matrix route.*
10. *Q is the* Pkg *provenance over* MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$, SubjectReductionDischargeSocket$^{\uparrow}$, *the four discharge rows,* ProofObligationGap$^{\uparrow}$, MetaCICOpenFrontierLedger$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{MetaCICSubjectReductionDischargeMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, A, L, P, O, S, R, H, C, Q, N$. It has no coordinate for a hidden subject-reduction proof, a quotient syntax equality, a host typing theorem, or an omitted fifth discharge row.

Theorem 6922.2 (MetaCIC discharge matrix NameCert obligations)**.** *For every accepted* MetaCICSubjectReductio *carrier $M = (B, A, L, P, O, S, R, H, C, Q, N)$, the local* NameCert$_{\text{MetaCICSubjectReductionDischargeMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission for the beta-substitution row, application-argument row, lambda-domain row, Pi-domain row, proof-obligation gap row, discharge-socket row, matrix-coverage row, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6922.1. The four discharge rows are exactly B, A, L, P, and the obstruction row O records why the matrix remains a finite interface rather than an unconditional theorem.

The socket row S fixes the consumer route. A subject-reduction consumer may enter through that socket only after the matrix-coverage row R has displayed the four rows simultaneously and kept their coordinates distinct.

The structural rows H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no hidden proof coordinate and no host typing coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6922.3 (MetaCIC discharge matrix four-row coverage)**.** *Every subject-reduction-facing consumer of an accepted* MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$ *carrier must read the four-row surface*

$$B, A, L, P \longmapsto O, S, R, H, C, Q, N.$$

The beta-substitution, application-argument, lambda-domain, and Pi-domain rows are all required before the socket can be consumed, and none can be replaced by the proof-obligation gap, transport, replay, provenance, local naming, or a host closed subject-reduction theorem.

Proof. Read the consumer request through the carrier. The first four coordinates B, A, L, P are the only discharge coordinates, and the matrix-coverage row R records them as simultaneous rows.

The obstruction row O has the opposite polarity from a discharge proof. It records that the closed theorem is unavailable without the four supplied rows, so it cannot replace any one of B, A, L, P.

The socket row S then exposes the consumer route, while H, C, Q, N preserve the same displayed matrix under transport, replay, provenance, and naming. Therefore every accepted subject-reduction read receives the four rows together and cannot collapse the matrix to a smaller interface. □

Closure status for NameCert$_{\text{MetaCICSubjectReductionDischargeMatrix}^{\uparrow}}$**.**

Constructive story: A MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$ packet is generated from beta-substitution, application-argument, lambda-domain, Pi-domain, proof-obligation gap, discharge-socket, matrix-coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICSubjectReductionDischargeMatrix}\uparrow}$, seed)

Scope closed: Definition 6922.1, Definition 6922.2, and Definition 6922.3 seed the finite four-row discharge matrix over $B, A, L, P, O, S, R, H, C, Q, N$.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICSubjectReductionDischargeMatrix}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove closed subject reduction, beta-substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-row coverage, socket preservation, proof-obligation-gap polarity, transport, replay, provenance, and local naming rows for BEDC.Derived.MetaCICSubjectReductionDischargeMatrixUp.

Chapter 6923

A Concrete Naming Certificate for $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$

Following van der Corput and Bishop–Bridges constructive analysis, the $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ packet records a finite-difference estimate by reading a source sequence only through named finite windows, shifted-pair difference rows, dyadic tolerance budgets, averaged correlation rows, regular rational readback, and the terminal real seal. It sits over the finite window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the real-series packet of chapter 1581. The certificate supplies downstream oscillation, Fourier, series, and completion routes with a finite estimate surface; it does not import host sequence limits, countable choice, quotient equality, selected subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6923.1 (Van der Corput finite-difference estimate carrier)**.** *A* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (A, W, S, D, \Gamma, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-sequence row, exposed either as a* $\mathsf{RealSeries}^{\uparrow}$ *term stream or as a named regular rational stream whose observations are read only through finite requests.*
2. *W is the family of* $\mathsf{StreamName}^{\uparrow}$ *source windows naming the finite index blocks used by the estimate.*
3. *S is the shifted-pair row listing the finite pairs $(n, n+h)$ inspected for each displayed shift h.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row, carrying the precision cells for source reads, difference reads, average comparisons, and the final estimate.*
5. *Γ is the averaged correlation row recording the finite sums of shifted products or shifted differences admitted by S, W, D.*
6. *M is the finite estimate row that binds the window length, the shift bound, the dyadic slack, and the averaged correlation rows into the displayed van der Corput bound.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from M after the same finite windows and dyadic budgets have been consumed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the estimate value after regular rational readback.*

9. *H records componentwise* hsame *transport for source windows, shifted-pair rows, dyadic budgets, averaged correlations, estimate rows, readback, seal, provenance, and local naming.*
10. *C records displayed* Cont *replay from a requested estimate through $A, W, S, D, \Gamma, M, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, S, D, \Gamma, M, R, E, H, C, P, N$. It has no coordinate for an infinite Cesaro limit, a selected subsequence, quotient stream equality, countable choice, Hilbert-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VanDerCorputDifferenceEstimateUp.VanDerCorputDifferenceEstimateTasteGate`
`single carrier alignment`

Theorem 6923.2 (Van der Corput finite-difference estimate NameCert obligations)**.** *For every accepted* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *carrier $V = (A, W, S, D, \Gamma, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}}$ *surface consists exactly of source admission, finite* $\mathsf{StreamName}^{\uparrow}$ *source-window admission, shifted-pair difference admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness, averaged correlation exactness, finite estimate-row exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 6923.1. The only source observations are the source row A, the finite window family W, and the shifted-pair row S.

The tolerance row D supplies the precision cells used by the shifted differences and by the averaged correlation row Γ. The estimate row M is downstream of these rows, so it can bind only the displayed window length, shift bound, dyadic slack, and averaged correlations already present in the carrier.

The readback row R and the seal row E are reached after M. The transport, replay, provenance, and local naming rows preserve the same finite route. Since the carrier has no coordinate for infinite limits, selected subsequences, quotient stream equality, countable choice, compactness, or ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 6923.3 (Van der Corput finite-window route)**.** *Every oscillation, Fourier, series, or completion-route consumer of an accepted* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *packet receives the estimate only through the finite route*

$$A \longmapsto W \longmapsto S \longmapsto D \longmapsto \Gamma \longmapsto M \longmapsto R \longmapsto E.$$

In particular, the estimate can be consumed by $\mathsf{RealSeries}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *routes only after the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets, shifted-pair rows, averaged correlation rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed.*

Proof. Start with the NameCert obligations of Definition 6923.2. They force every accepted estimate read to expose $A, W, S, D, \Gamma, M, R, E, H, C, P, N$.

The finite windows W delimit the source observations, and the shifted-pair row S names the exact pairs inspected for each shift. No downstream consumer can read a correlation row before these windows and pairs have been displayed.

The dyadic budget row D then controls both the shifted difference reads and the averaged correlation row Γ. The estimate row M is the first row that binds those correlations into a numerical estimate, and the regular readback R is downstream of that row.

Finally the terminal seal E is reached only after R. Thus any consumer that enters the packet from a real-series, Fourier, oscillation, or completion route sees the same finite window route and no hidden sequence-limit object. □

Theorem 6923.4 (Van der Corput finite-difference non-escape)**.** *An accepted* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *packet exports no data outside its displayed carrier rows. In particular, it does not export a host convergence theorem, a Cesaro-limit value, a selected density subsequence, quotient equality of streams, countable-choice schedules, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness; every exported estimate is mediated by* $W, S, D, \Gamma, M, R, E, H, C, P, N$.

Proof. Use the carrier description in Definition 6923.1. The accepted packet stores only the source row, finite windows, shifted pairs, dyadic budgets, averaged correlations, estimate row, readback, seal, and structural certificate rows.

Apply the finite-window route of Definition 6923.3. Any exported estimate must pass through W, S, D, Γ, M, R, E before the structural rows transport and replay it.

The excluded principles have no carrier coordinate and are not introduced by hsame, Cont, Pkg, or local naming. Therefore they cannot escape through the certificate surface. □

Falsifiable prediction: Every accepted $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ read factors through source windows, shifted-pair difference rows, DyadicRatCore tolerance budgets, averaged correlation rows, a finite estimate row, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, realseries: $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source sequence windows, shifted-pair difference rows, dyadic tolerance budgets, averaged correlation rows, finite estimate accounting, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ packet is generated from source sequence windows, shifted-pair difference rows, dyadic tolerance budgets, averaged correlation rows, a finite estimate row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6923.1, Definition 6923.2, Definition 6923.3, and Definition 6923.4 seed the finite van der Corput difference-estimate carrier, certificate surface, finite-window route, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence limits, Fourier convergence, oscillation decay, selected subsequences, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, shifted-pair exactness, finite window coverage, dyadic tolerance exactness, averaged correlation exactness, regular readback, Real seal non-escape, transport, replay, provenance, and local NameCert coverage for BEDC.Derived.VanDerCorputDifferenceEstimateUp.

Chapter 6924

A Concrete Naming Certificate for $\mathsf{VanDerCorputSum}^{\uparrow}$

Following van der Corput and Bishop–Bridges constructive analysis, the $\mathsf{VanDerCorputSum}^{\uparrow}$ packet records a finite exponential-sum estimate as displayed NameCert data. It reads coefficient or phase streams through the finite window surface of chapter 126, dyadic tolerance budgets through chapter 291, regular rational readback through chapter 301, terminal real seals through chapter 38, the real-series source row of chapter 1581, Fourier-facing coefficient windows through chapter 124, and finite-difference control through chapter 6923. The packet names only the finite sum route; it does not claim convergence of an infinite trigonometric series, quotient equality of functions, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6924.1 (Van der Corput finite exponential-sum carrier)**.** *A* $\mathsf{VanDerCorputSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, \Phi, Q, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{RealSeries}^{\uparrow}$ *or Fourier coefficient row whose finite terms are inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite index block and every coefficient or phase request used by the sum.*
3. *Φ is the finite phase row, recording the exponential phases only through the displayed window requests.*
4. *Q is the quadratic or shifted-difference control row handed off from* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *when a finite differencing estimate is used.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row for coefficient reads, phase reads, summand products, partial sums, and the final bound.*
6. *M is the finite exponential-sum ledger, listing the displayed summands, partial sums, normalization factors, and dyadic slack.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after W, Φ, Q, D, M have been consumed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed finite bound or normalized finite sum.*
9. *H records componentwise* hsame *transport for source, window, phase, difference-control, dyadic-budget, sum-ledger, readback, seal, replay, provenance, and local-name rows.*

10. *C records displayed* Cont *replay from a requested finite exponential-sum read through* $A, W, \Phi, Q, D, M, R, E, H, P$
11. *P is the* Pkg *provenance over* $\mathsf{VanDerCorputSum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VanDerCorputSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, \Phi, Q, D, M, R, E, H, C, P, N$. It has no coordinate for host Fourier convergence, an infinite exponential series, quotient equality of functions, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6924.2 (Van der Corput finite exponential-sum NameCert obligations)**.** *For every accepted* $\mathsf{VanDerCorputSum}^{\uparrow}$ *carrier $S = (A, W, \Phi, Q, D, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VanDerCorputSum}^{\uparrow}}$ *surface consists exactly of source admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission, displayed phase admission, optional finite-difference control through* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness, finite exponential-sum ledger exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 6924.1. The only source observations are the real-series or Fourier row A, the finite window row W, and the displayed phase row Φ.

The difference-control row Q is available only as a finite handoff from chapter 6923. The dyadic row D then supplies the precision cells for the coefficient reads, phase reads, summand products, and final estimate before the sum ledger M is read.

The readback row R and Real seal row E are downstream of M. Transport, replay, provenance, and local naming preserve that same finite route. Since the carrier omits host Fourier convergence, infinite exponential series, quotient equality, selected subsequences, countable choice, and ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 6924.3 (Van der Corput finite exponential-sum route)**.** *Every Fourier, real-series, oscillation, or completion-route consumer of an accepted* $\mathsf{VanDerCorputSum}^{\uparrow}$ *packet receives the finite exponential-sum read only through*

$$A \longmapsto W \longmapsto \Phi \longmapsto Q \longmapsto D \longmapsto M \longmapsto R \longmapsto E.$$

In particular, the consumer must display the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{RealSeries}^{\uparrow}$ *or* $\mathsf{Fourier}^{\uparrow}$ *source row, and any* $\mathsf{VanDerCorputDifferenceEstimate}^{\uparrow}$ *handoff before it can inspect the finite bound.*

Proof. Start from the obligation surface of Definition 6924.2. It forces every accepted finite-sum read to expose $A, W, \Phi, Q, D, M, R, E, H, C, P, N$.

The finite window row W controls which source coefficients and phases can be read. The phase row Φ and optional difference-control row Q are therefore downstream of the same displayed index block.

The dyadic budget row D controls the arithmetic cells used by the sum ledger M. Only after M has been displayed can the regular readback R and terminal seal E be consumed.

Thus a Fourier, real-series, oscillation, or completion-route consumer sees the same finite route and receives no host convergence theorem, selected subsequence, quotient equality, countable-choice schedule, or ambient Real completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{VanDerCorputSum}^{\uparrow}$ read factors through a RealSeries or Fourier source row, StreamName windows, phase rows, optional VanDerCorputDifferenceEstimate

control, DyadicRatCore budgets, a finite exponential-sum ledger, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, real, realseries, fourier, van-der-corput-difference-estimate: $\mathsf{VanDerCorputSum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite coefficient or phase windows, difference-control handoff, dyadic arithmetic budgets, a finite exponential-sum ledger, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{VanDerCorputSum}^{\uparrow}}$.

Constructive story: A $\mathsf{VanDerCorputSum}^{\uparrow}$ packet is generated from a RealSeries or Fourier source row, finite StreamName windows, displayed phase rows, optional VanDerCorputDifferenceEstimate handoff, DyadicRatCore tolerance budgets, a finite exponential-sum ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VanDerCorputSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6924.1, Definition 6924.2, and Definition 6924.3 seed the finite exponential-sum carrier, certificate surface, and consumer route over displayed BEDC rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VanDerCorputSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Fourier convergence, infinite exponential-series convergence, quotient function equality, selected subsequences, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite window coverage, phase admission, finite-difference handoff, dyadic tolerance exactness, exponential-sum ledger exactness, regular readback, Real seal non-escape, transport, replay, provenance, and local NameCert coverage for BEDC.Derived.VanDerCorputSumUp.

Chapter 6925

A Concrete Naming Certificate for $\mathsf{LocatedRealModulusExtractor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealModulusExtractor}^{\uparrow}$ packet records the finite extraction route that reads located-real enclosure data as a regular-Cauchy modulus surface rather than as quotient real equality or an ambient completeness theorem. It sits over the located-real carrier of chapter 1933, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located modulus route of chapter 3406, and the regular Cauchy modulus handoff of chapter 292. The packet gives the Phase Real and Completion route a named bridge from located-real locator rows to finite modulus extraction without claiming $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6925.1 (LocatedRealModulusExtractor carrier)**.** *A* $\mathsf{LocatedRealModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, D, S, R, M, W, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locator row whose interval and apartness-enclosure data answer the displayed precision request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the requested radius, endpoint slack, and comparison budget used for the extraction.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from L at the budget recorded by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window and dyadic tolerance ledger.*
5. *M is the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row, recording the threshold supplied by the located data for the regular-Cauchy consumer.*
6. *W is the terminal* $\mathsf{Real}^{\uparrow}$ *seal route reached only after L, D, S, R, M have been displayed.*
7. *H stores componentwise* hsame *transport for the locator row, dyadic ledger, finite window, regular readback, extracted modulus, real seal, replay route, provenance, and local naming.*
8. *C stores the displayed* Cont *replay from a precision request through L, D, S, R, M to the seal row W and to downstream completion consumers.*

9. *P is the* Pkg *provenance over* LocatedRealModulusExtractor$^{\uparrow}$, LocatedReal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{LocatedRealModulusExtractor}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, trichotomy, a choice-selected subsequence, a selected limiting representative, countable choice, host completeness, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6925.2 (LocatedRealModulusExtractor NameCert obligations)**.** *For an accepted* LocatedRealModulusEx *carrier $E = (L, D, S, R, M, W, H, C, P, N)$, the local* NameCert$_{\text{LocatedRealModulusExtractor}^{\uparrow}}$ *surface consists exactly of carrier admission, located-real locator exposure, dyadic precision-ledger exactness, finite stream-window coverage, regular rational readback compatibility, extracted Cauchy-modulus exposure, real-seal nonescape, componentwise transport, continuation replay, provenance, and local naming for the packet of Definition 6925.1.*

Proof. Carrier admission is projection from the displayed tuple. The analytic rows are L, D, S, R, M, W, and the structural rows are H, C, P, N.

The located-real row L supplies locator data for the precision request, while D records the dyadic budget under which that request is read. The stream row S and readback row R are opened only at the displayed budget, so the extracted modulus row M is tied to the same finite window.

The seal row W is downstream of the extraction route. Transport and replay preserve the order through H and C, while P and N add provenance and naming without introducing quotient equality, trichotomy, selected limits, choice, host completeness, or ambient Real$^{\uparrow}$ completeness. □

Theorem 6925.3 (LocatedRealModulusExtractor readback route)**.** *Every regular-Cauchy or Real and Completion consumer of an accepted* LocatedRealModulusExtractor$^{\uparrow}$ *packet must read the route*

$$L, D, S, R, M, W, H, C, P, N$$

before using the extracted modulus. The route exposes the LocatedReal$^{\uparrow}$ *locator,* DyadicRatCore$^{\uparrow}$ *precision ledger,* StreamName$^{\uparrow}$ *finite window,* RegSeqRat$^{\uparrow}$ *readback, extracted* CauchyModulus$^{\uparrow}$ *row, and terminal* Real$^{\uparrow}$ *seal as one finite packet; it exports no quotient real equality, trichotomy, selected representative, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the carrier inventory of Definition 6925.1. The located-real row and dyadic ledger precede the finite-window and regular-rational rows, and the modulus row M is displayed only after those rows have been read.

Use the local obligations of Definition 6925.2. They require locator exposure, dyadic exactness, window coverage, regular rational readback compatibility, and extracted-modulus exposure to be certified for the same packet.

Finally read the handoff through W, H, C, P, N. These rows seal, transport, replay, source, and name the same finite route, so a downstream consumer receives the extracted modulus only through the displayed packet and not through any refused external principle. □

Falsifiable prediction: An accepted LocatedRealModulusExtractor$^{\uparrow}$ packet cannot provide a regular-Cauchy modulus unless the same packet displays the located-real locator row, dyadic precision ledger, finite stream window, and regular rational readback that produce it.

Independence witness: located-real-modulus, cauchymodulus, regseqrat: LocatedRealModulusExtractor$^{\uparrow}$ joins locator rows to an extracted CauchyModulus row and a Real-facing seal in one packet; the sibling packets expose their own carriers but do not provide this combined extraction surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealModulusExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealModulusExtractor}^{\uparrow}$ packet is generated from a displayed LocatedReal locator row, a DyadicRatCore precision ledger, a StreamName finite-window row, a RegSeqRat readback row, an extracted CauchyModulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6925.1, Definition 6925.2, and Definition 6925.3 seed the finite located-real modulus-extraction route over LocatedReal, DyadicRatCore, StreamName, RegSeqRat, CauchyModulus, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealModulusExtractor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, trichotomy, selected limits, countable choice, host completeness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, locator exposure, dyadic-window coverage, regular readback compatibility, extracted-modulus exposure, RealUp seal nonescape, transport stability, continuation replay, provenance, and local naming for BEDC.Derived.LocatedRealModulusExtractorUp.

Chapter 6926

A Concrete Naming Certificate for $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ packet records the finite observation data by which a regular rational name is refined before a Real-facing consumer may read a terminal seal. It sits over the stream-window discipline of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, the Cauchy modulus route of chapter 292, the finite envelope of chapter 365, and the real-name boundary of chapter 38. The packet is a finite-window refinement certificate; it is not a quotient of streams, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6926.1 (Finite-window RegSeqRat refinement carrier)**.** *A* $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose finite windows are inspected before any regular-rational refinement is read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the source regular rational readback.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the coarse and refined windows.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row supplying the finite threshold that bounds the requested refinement.*
5. *W is the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row packaging the refined window, its source window, and the dyadic tolerance request in one envelope.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, R, D, M, W have been displayed.*
7. *H records componentwise* hsame *transport for the displayed rows.*
8. *C records displayed* Cont *replay from a refinement request through the same finite packet.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, D, M, W, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FiniteWindowRegSeqRatRefinementUp.FiniteWindowRegSeqRatRefinementUp`

Theorem 6926.2 (Finite-window RegSeqRat refinement NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ *carrier* $F = (S, R, D, M, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *source readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold exposure,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *refinement packaging, terminal* $\mathsf{Real}^{\uparrow}$ *seal discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6926.1. The rows S, R, D, M, W, E display the whole finite observation route before any structural row is read.

The numerical rows are exhausted by S, R, D, M, W. Stream windows are opened in S, regular rational readback is exposed by R, dyadic tolerance accounting is displayed in D, the modulus threshold is supplied by M, and the finite-window envelope W records the refinement boundary.

The structural rows are exhausted by H, C, P, N. They transport, replay, source, and name the same displayed packet, while the carrier provides no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 6926.3 (Finite-window RegSeqRat refinement Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ *carrier* $F = (S, R, D, M, W, E, H, C, P, N)$ *must read*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{FiniteWindowEnvelope}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows S, R, D, M, W, E before the terminal seal is consumed. The handoff exports no quotient stream, selected host limit, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier projection. The only available source window is S, the only regular rational readback is R, and the only tolerance ledger is D.

Read the modulus and envelope rows next. The threshold row M bounds the finite refinement request, while W packages the refined window and the source window in the same finite packet.

Finally consume the seal row E through H, C, P, N. Because the packet has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness, the Real-facing consumer receives only the displayed finite-window handoff. □

Falsifiable prediction: Every accepted $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ Real-facing read displays StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a CauchyModulus threshold, a FiniteWindowEnvelope refinement row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchymodulus, finitewindowenvelope, real: $\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}$ is not bijective to any listed sibling because it packages the finite refinement envelope between regular rational readback and the terminal Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowRegSeqRatRefinement}^{\uparrow}}$.

Constructive story: A FiniteWindowRegSeqRatRefinement$^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a CauchyModulus threshold, a FiniteWindowEnvelope refinement row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteWindowRegSeqRatRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 6926.1, Definition 6926.2, and Definition 6926.3 seed the finite-window RegSeqRat refinement route before any Real-facing seal consumption.

Formal verification: FormalStatus(NameCert$_{\text{FiniteWindowRegSeqRatRefinement}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.FiniteWindowRegSeqRatRefinementUp.FiniteWindowRegSeqRatRefinementUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, regular readback, dyadic tolerance accounting, modulus threshold exposure, envelope exactness, seal discipline, transport, replay, provenance, and local naming.

Chapter 6927

A Concrete Naming Certificate for FiniteCoverBarModulus$^{\uparrow}$

Following Bishop and Bridges' finite-cover route to uniform continuity on compact metric spaces, the FiniteCoverBarModulus$^{\uparrow}$ packet records the boundary where a compact finite cover, local pointwise moduli, a dyadic lower-bound fold, regular-rational readback, and a terminal Real seal are packaged as L10 Real-and-Completion data. It sits after the compact finite-net carrier of chapter 129, the continuous-map rows of chapter 130, the compact-uniform route of chapter 388, and the tolerance-only modulus consumer of chapter 369. The packet names a finite bar-cover modulus without open-cover compactness, selected subcovers, host functions, quotient stream equality, or ambient Real$^{\uparrow}$ completeness.

Definition 6927.1 (Finite cover bar modulus carrier)**.** *A* FiniteCoverBarModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, L, D, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-cover or finite-net source row inspected by the bar.*
2. *F is the finite bar-cover row listing the displayed centers, cover radii, and branch-depth ledger used by the compact source.*
3. *L is the* ContinuousMap$^{\uparrow}$ *local-modulus row attached to each displayed cover member.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance and lower-bound fold row.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced from the displayed dyadic fold.*
6. *E is the terminal* Real$^{\uparrow}$ *seal for the modulus bound read from R.*
7. *U is the* UniformModulus$^{\uparrow}$ *consumer row available only after K, F, L, D, R, E have been displayed.*
8. *H stores componentwise* hsame *transport for the compact source, bar cover, local moduli, dyadic fold, regular readback, Real seal, uniform-modulus row, replay, provenance, and local naming.*
9. *C stores displayed* Cont *replay from a compact-uniform-continuity request through $K, F, L, D, R, E, U, H, P, N$.*
10. *P is the* Pkg *provenance over* FiniteCoverBarModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCoverBarModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, a selected finite subcover, a host modulus function, quotient equality of functions or streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6927.2 (Finite cover bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ *carrier* $B = (K, F, L, D, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverBarModulus}^{\uparrow}}$ *surface consists exactly of compact finite-cover admission, finite bar-cover exposure, local-modulus visibility, dyadic lower-bound fold exactness, regular-rational readback, terminal Real sealing, uniform-modulus handoff, componentwise transport, displayed replay, provenance, and local naming for the rows of Definition 6927.1.*

Proof. Project the carrier. The analytic rows are K, F, L, D, R, E, U, and the structural rows are H, C, P, N.

The compact source and bar-cover rows K, F list the finite cover data before any local modulus is read. The row L then attaches pointwise modulus data to the displayed cover members, while D folds the resulting dyadic lower bounds inside the same finite packet.

The readback row R, terminal seal E, and uniform-modulus row U are downstream of that finite fold. Transport, replay, provenance, and local naming preserve the same inventory, so the local NameCert surface contains exactly the listed obligations and no hidden subcover selection or host modulus. □

Theorem 6927.3 (Finite cover bar modulus Real-and-Completion handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ *packet must read the route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}$$

through the displayed rows $K, F, L, D, R, E, U, H, C, P, N$. The exported modulus is the finite bar-cover lower-bound fold, not an open-cover compactness theorem, selected subcover, quotient function equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the obligation surface of Definition 6927.2. It forces the consumer to expose the compact source, bar-cover row, local moduli, dyadic fold, regular readback, Real seal, and uniform-modulus handoff before the packet is exported.

The Real-and-Completion read therefore reaches $\mathsf{RegSeqRat}^{\uparrow}$ only after the finite dyadic fold D has been displayed. It reaches $\mathsf{Real}^{\uparrow}$ only through the readback row R, and it reaches $\mathsf{UniformModulus}^{\uparrow}$ only after the terminal seal E is visible.

Since the carrier inventory has no row for an open cover, selected subcover, host modulus function, quotient function equality, choice schedule, or ambient completeness theorem, the handoff is exhausted by $K, F, L, D, R, E, U, H, C, P, N$. □

Theorem 6927.4 (Finite cover bar modulus compact-uniform route)**.** *Every accepted* $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ *packet consumed by chapter 388 or chapter 369 exposes the same finite bar-cover row F, local-modulus row L, dyadic fold D, regular readback R, Real seal E, and uniform-modulus row U. Thus the compact-uniform-continuity route remains a finite NameCert route over displayed compact, continuous-map, dyadic, regular-rational, Real, and uniform-modulus data.*

Proof. Apply Definition 6927.2. The compact-uniform consumer may inspect only the displayed carrier rows and the structural replay rows.

Use Definition 6927.3. The downstream route orders the dyadic fold before the RegSeqRat readback, the readback before the Real seal, and the Real seal before the UniformModulus consumer row.

The sibling chapters cited in the introduction supply the compact source, continuous-map route, compact-uniform route, and uniform-modulus consumer. The present packet joins those surfaces as one finite bar-cover modulus carrier and exports no further principle. □

Theorem 6927.5 (Finite cover bar modulus nonescape)**.** *An accepted* $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ *read cannot infer open-cover compactness, a selected finite subcover, host function equality, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness unless one of those rows is explicitly added to the displayed carrier. The carrier of Definition 6927.1 adds none of them.*

Proof. Project the displayed tuple $B = (K, F, L, D, R, E, U, H, C, P, N)$. Each coordinate is one of the compact finite-cover, bar-cover, local-modulus, dyadic-fold, regular-readback, Real-seal, uniform-modulus, transport, replay, provenance, or local-name rows.

The three preceding theorems consume exactly those rows for the NameCert obligation surface, the Real-and-Completion handoff, and the compact-uniform route. None of the arguments introduces an external cover object, choice operation, quotient, host function equality, or completeness theorem.

Therefore the refused principles are absent by carrier inventory. Any consumer that uses one of them has left the accepted $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ packet. □

Falsifiable prediction: An accepted $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ consumer that exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying $K, F, L, D, R, E, U, H, C, P, N$ refutes the carrier surface.

Independence witness: compact metric source, continuous-map local modulus, uniform-modulus handoff, regular rational readback: $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ binds all four with a finite bar-cover and dyadic lower-bound fold, while each sibling supplies only its own carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverBarModulus}^{\uparrow}$ packet is generated from CompactMetricUp finite-cover rows, a finite bar-cover ledger, ContinuousMapUp local-modulus rows, a DyadicRatCoreUp lower-bound fold, RegSeqRatUp readback, RealUp seal, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6927.1, Definition 6927.2, Definition 6927.3, and Definition 6927.4 seed the finite bar-cover modulus route for compact-uniform-continuity and Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, host modulus functions, quotient function or stream equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar-cover exposure, local-modulus visibility, dyadic lower-bound fold exactness, regular readback, Real seal nonescape, uniform-modulus handoff, transport stability, replay, provenance, and local naming for BEDC.Derived.FiniteCoverBarModulusUp.

Chapter 6928

A Concrete Naming Certificate for $\mathsf{CofinalMetricCompletion}^{\uparrow}$

Following Kelley and Bourbaki on uniform and metric completions, and Bishop–Bridges on constructive completion by regular approximants, the $\mathsf{CofinalMetricCompletion}^{\uparrow}$ packet names the finite cofinal-window surface by which a metric-completion consumer reads regular rational evidence before reaching a terminal real-name seal. It sits between the metric completion carrier of chapter 1506, the functorial uniform-completion route of chapter 1892, the Cauchy-completion handoff of chapter 423, the finite StreamName route of chapter 126, the RegSeqRat readback of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, and the RealUp seal of chapter 38. The packet is a completion-facing cofinal-window certificate; it is not quotient completion equality, a selected representative, a choice function, or an ambient completeness principle.

Definition 6928.1 (Cofinal metric-completion carrier)**.** *A* $\mathsf{CofinalMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, U, F, W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row exposing the metric-completion request surface.*
2. *U is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *route row through which completion-facing functorial reads are admitted.*
3. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row downstream of M and U.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *cofinal-window ledger naming the finite observation windows served to the completion request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read from W and R.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, U, F, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the metric source, functorial route, Cauchy handoff, cofinal windows, readback, tolerance, seal, and structural rows.*
9. *C records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{CofinalMetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalMetricCompletion}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only $M, U, F, W, R, D, S, H, C, P, N$. It has no coordinate for quotient completion equality, host completed-real equality, selected limits, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 6928.2 (Cofinal metric-completion NameCert obligations)**.** *For every accepted* $\mathsf{CofinalMetricCompletion}^{\uparrow}$ *carrier $K = (M, U, F, W, R, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalMetricCompletion}^{\uparrow}}$ *surface consists exactly of metric-completion source exposure through M, uniform functorial route exposure through U, Cauchy-completion handoff through F, cofinal StreamName windows through W, RegSeqRat readback through R, DyadicRatCore tolerance through D, terminal Real seal access through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the displayed carrier of Definition 6928.1. The metric source, uniform functorial route, Cauchy handoff, cofinal-window ledger, readback, tolerance, seal, transport, replay, provenance, and local naming rows are exactly the coordinates of one finite BHist packet.

The metric-completion request first reads M and then the functorial route U, F. This places the request on the same completion-facing path used by the sibling UniformCompletionFunctor and CauchyCompletionFunctor certificates before any Real-facing row is available.

The cofinal-window surface is then fixed by W, R, D. Each precision request is served by a displayed StreamName window, regular rational readback, and dyadic tolerance ledger, so the consumer cannot replace the route by quotient completion equality, a hidden representative, or host-level complete metric data.

Finally S, H, C, P, N seal and name the same route. Since no excluded object appears among the carrier coordinates, the local NameCert obligations are exhausted by the displayed finite packet. □

Theorem 6928.3 (Cofinal metric-completion L10 handoff)**.** *Every L10 completion-facing read exported by an accepted* $\mathsf{CofinalMetricCompletion}^{\uparrow}$ *packet factors through the finite route*

$$M, U, F, W, R, D, S, H, C, P, N.$$

In particular, a consumer reaches the $\mathsf{Real}^{\uparrow}$ *seal only after reading the* $\mathsf{MetricCompletion}^{\uparrow}$ *source, the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *route, the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff, the* $\mathsf{StreamName}^{\uparrow}$ *cofinal-window ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. The rows H, C, P, N preserve the same route and do not add quotient completions, selected limits, choice, or ambient completeness.*

Proof. Start with the NameCert obligations of Definition 6928.2. They expose the route inventory and require every accepted consumer to read the displayed metric, uniform-functorial, Cauchy-handoff, cofinal-window, readback, tolerance, seal, transport, replay, provenance, and naming rows.

The sibling metric-completion and uniform-completion-functor chapters supply the source shape for the first three rows: M is the metric-completion entry point, while U, F keep the request on the functorial completion handoff rather than on a host completion object.

The finite L10 window is the middle segment W, R, D. It records the StreamName window, the RegSeqRat readback, and the DyadicRatCore tolerance ledger used by the request before the terminal Real seal is read.

The remaining rows S, H, C, P, N only seal, transport, replay, source, and name the same finite route. Therefore the handoff exported to downstream completion consumers is exactly the displayed cofinal-window packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalMetricCompletion}^{\uparrow}$ packet is generated from a MetricCompletion source row, a UniformCompletionFunctor route row, a CauchyCompletionFunctor handoff row, StreamName cofinal windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6928.1, Definition 6928.2, and Definition 6928.3 seed the finite cofinal-window route from MetricCompletion and UniformCompletionFunctor data to RegSeqRat, DyadicRatCore, and RealUp consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host completed-real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-window exactness, RegSeqRat readback, DyadicRatCore tolerance, RealUp seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.CofinalMetricCompletio

Chapter 6929

A Concrete Naming Certificate for $\mathsf{RegularConditionalDistribution}^{\uparrow}$

Following Kallenberg and standard regular conditional probability theory, the $\mathsf{RegularConditionalDistribution}^{\uparrow}$ packet records a source random variable, a target event family, and a state-indexed probability law as finite BEDC data. It sits over the probability-space surface of chapter 184, the measurable-preimage surface of chapter 185, the pushforward-law surface of chapter 186, the projection surface of chapter 188, the state-to-distribution interface of chapter 6735, the generated-event surface of chapter 6308, the Polish presentation surface of chapter 1490, and the regular-measure surface of chapter 2263. The chapter names regular conditional laws without importing a choice-selected disintegration, host kernel equality, quotient event algebra, or ambient measurable-space oracle.

Definition 6929.1 (Regular conditional distribution carrier)**.** *A* $\mathsf{RegularConditionalDistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (\Omega, S, T, X, Y, \mathcal{A}_S, \mathcal{A}_T, \kappa, D, E, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the source* $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the probability measure from which conditioning is read.*
2. S *is the conditioning-state row, read through a* $\mathsf{StandardBorelSpace}^{\uparrow}$ *presentation and the underlying* $\mathsf{PolishSpace}^{\uparrow}$ *event generator.*
3. T *is the target-state row, with target events carried by* $\mathcal{A}_T$*.*
4. X *is the conditioning* $\mathsf{RandomVar}^{\uparrow}$ *row from* Ω *into* S*.*
5. Y *is the target* $\mathsf{RandomVar}^{\uparrow}$ *row from* Ω *into* T*.*
6. $\mathcal{A}_S$ *and* $\mathcal{A}_T$ *are the displayed source and target measurable-event families.*
7. κ *is the state-indexed endpoint assignment sending each displayed conditioning state* s *and target event* $B \in \mathcal{A}_T$ *to a* $\mathsf{Real}^{\uparrow}$ *endpoint* $\kappa(s, B)$*.*
8. D *records that each fixed state* s *reads as a* $\mathsf{Distribution}^{\uparrow}$ *row over target events and as a* $\mathsf{MarkovKernel}^{\uparrow}$ *fiber from* S *to* T*.*
9. E *records conditional-expectation exactness: for each displayed target event* B*, the row* $s \mapsto \kappa(s, B)$ *is the* $\mathsf{CondExp}^{\uparrow}$ *readback of the indicator of* $Y^{-1}(B)$ *over the conditioning row* X*.*

10. *H records componentwise* hsame *transport for source states, target events, endpoints, distribution fibers, conditional-expectation rows, and regular-measure boundaries.*
11. *C records displayed* Cont *replay from $\Omega, X, Y, S, T, \mathcal{A}_S, \mathcal{A}_T$ through κ, D, E, H, P, N.*
12. *P is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{MarkovKernel}^{\uparrow}$, $\mathsf{StandardBorelSpace}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularConditionalDistribution}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected disintegration, host conditional probability object, quotient event equality, external sigma-algebra completion, or unledgered kernel equality.

Theorem 6929.2 (Regular conditional distribution NameCert obligations)**.** *For the carrier of Definition 6929.1, the local* $\mathsf{NameCert}_{\mathsf{RegularConditionalDistribution}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (\Omega, S, T, X, Y, \mathcal{A}_S, \mathcal{A}_T, \kappa, D, E, H, C, P, N),$$

source probability exposure through Ω, conditioning and target random-variable exposure through X, Y, generated event exposure through $S, T, \mathcal{A}_S, \mathcal{A}_T$, endpoint readback through κ, distribution-fiber and Markov-kernel exposure through D, conditional-expectation exactness through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.

Consequently an accepted $\mathsf{RegularConditionalDistribution}^{\uparrow}$ *packet is a finite state-indexed law whose measurable fibers are read through the displayed* $\mathsf{StandardBorelSpace}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{MarkovKernel}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, *and* $\mathsf{RadonMeasure}^{\uparrow}$ *rows. It is not a selected disintegration theorem and not a host conditional-probability object.*

Proof. Project the carrier of Definition 6929.1. Its rows expose the probability source, conditioning and target random variables, generated measurable-event families, state-indexed endpoint assignment, distribution fiber, conditional-expectation exactness, transport, replay, provenance, and local name.

Read D at a fixed conditioning state. The endpoint assignment $B \mapsto \kappa(s, B)$ is a $\mathsf{Distribution}^{\uparrow}$ row over the target event family and a $\mathsf{MarkovKernel}^{\uparrow}$ fiber from the conditioning state to target events.

Read E at a fixed target event. It identifies the state-indexed endpoint row with the $\mathsf{CondExp}^{\uparrow}$ readback of the target-event indicator along X, using only the displayed random-variable and probability-space rows.

The rows H, C, P, N preserve and name the same packet. Since no coordinate carries selected disintegration data, quotient event equality, host kernel equality, or an ambient measurable-space oracle, the listed finite rows exhaust the NameCert obligations. □

Theorem 6929.3 (Regular conditional distribution kernel readback)**.** *Every accepted* RegularConditionalDistributio *readback factors through the displayed route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{RandomVar}^{\uparrow}, \mathsf{StandardBorelSpace}^{\uparrow}, \mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{MarkovKernel}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto \mathsf{CondExp}^{\uparrow}.$$

The target-event value $\kappa(s, B)$ is public only after the conditioning-state row s, the target event B, the distribution fiber, and the conditional-expectation exactness row are displayed on the same finite packet. The readback exports no selected disintegration, host conditional-probability equality, quotient event algebra, or ambient regularity theorem.

Proof. Start with the source rows $\Omega, X, Y, S, T, \mathcal{A}_S, \mathcal{A}_T$ of the carrier. The probability and random-variable rows give the conditioning and target events, while the standard Borel and Polish rows make the event codes available as displayed data.

Use the fiber row D. At each displayed conditioning state s, the endpoint assignment over target events is read as a $\mathsf{MarkovKernel}^{\uparrow}$ fiber and as a $\mathsf{Distribution}^{\uparrow}$ row, so the consumer receives a state-indexed law rather than a detached host object.

Finally apply the exactness row E. For every target event B, the endpoint $\kappa(s, B)$ is tied to the $\mathsf{CondExp}^{\uparrow}$ readback of the indicator of $Y^{-1}(B)$ along X. The transport and replay rows preserve that same route, so no selected disintegration or host kernel equality can enter the public readback. □

Theorem 6929.4 (Regular conditional distribution ledger nonescape)**.** *No accepted* $\mathsf{RegularConditionalDistribution}^{\uparrow}$ *packet exports a conditional law outside the displayed rows*

$$\Omega, S, T, X, Y, \mathcal{A}_S, \mathcal{A}_T, \kappa, D, E, H, C, P, N.$$

In particular, the packet does not export a choice-selected disintegration, a host conditional-probability primitive, quotient event equality, a hidden standard Borel completion, or an unledgered Radon regularity claim.

Proof. Begin with Definition 6929.2. It exhausts the local certificate by the probability source, random-variable rows, generated event rows, endpoint assignment, distribution fiber, conditional-expectation exactness, transport, replay, provenance, and local name.

Use Definition 6929.3 to keep every public endpoint on the same finite route from conditioning state and target event to kernel fiber, distribution row, and conditional-expectation readback.

The $\mathsf{StandardBorelSpace}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, and $\mathsf{RadonMeasure}^{\uparrow}$ dependencies provide displayed event generation and regular-measure boundaries, but the carrier records them through P, H, C rather than as ambient host structure. Thus the listed rows are the only public coordinates, and the stated escape routes are not exports of this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularConditionalDistribution}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularConditionalDistribution}^{\uparrow}$ packet is generated from a probability source, conditioning and target random variables, standard Borel and Polish event rows, state-indexed endpoint fibers, distribution and Markov-kernel readback, conditional-expectation exactness, regular-measure provenance, transport, replay, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularConditionalDistribution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6929.1, Definition 6929.2, Definition 6929.3, and Definition 6929.4 seed the finite regular-conditional-law packet and its kernel readback boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularConditionalDistribution}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close existence of regular conditional distributions, selected disintegration, host conditional probability, quotient event equality, Lean checking, or bridge checking for $\mathsf{RegularConditionalDistribution}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, fiber distribution readback, conditional-expectation exactness, transport, provenance, local naming, and nonescape for the same packet.

Chapter 6930

A Concrete Naming Certificate for $\mathsf{BishopCauchyLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyLocator}^{\uparrow}$ packet records the finite locator surface by which a regular Cauchy rational name is read through explicit windows, dyadic tolerances, and a located tail threshold before any Real or Completion consumer may cite it. It sits beside the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance carrier of chapter 291, the located tail-modulus route of chapter 1035, and the located Cauchy real interface of chapter 1850. The packet supplies a constructive locator object for the Real and Completion corridor without quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6930.1 (Bishop Cauchy locator carrier)**.** *A* $\mathsf{BishopCauchyLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected source windows of the regular Cauchy name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those same windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested radius, endpoint slack, and comparison budget.*
4. *T is the located tail-threshold row, read through the* $\mathsf{LocatedCauchyModulus}^{\uparrow}$ *route and compatible with the located Cauchy interface of* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$*.*
5. *H stores componentwise* hsame *transport for the window, readback, dyadic tolerance, located threshold, replay, provenance, and local naming rows.*
6. *C stores the displayed* Cont *replay from a precision request through S, R, D, T to every downstream locator consumer.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyLocator}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedCauchyModulus}^{\uparrow}$*,* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyLocator}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected limit, a countable-choice selector, trichotomy, host real equality, or ambient completeness.

Theorem 6930.2 (Bishop Cauchy locator NameCert obligations)**.** *For an accepted* $\mathsf{BishopCauchyLocator}^{\uparrow}$ *carrier* $L = (S, R, D, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyLocator}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window exposure, regular rational readback, dyadic tolerance exactness, located tail-threshold exposure, componentwise transport, continuation replay, provenance, and local naming for the packet of Definition 6930.1.*

Proof. Project the displayed carrier. The analytic rows are S, R, D, T, and the structural rows are H, C, P, N.

The stream row S supplies the finite inspection window, while R records the regular rational readback on that same window. The dyadic row D fixes the tolerance under which the readback is tested, and the located threshold row T records the tail boundary available to Real and Completion consumers.

The remaining rows do not add a second locator. Transport, replay, provenance, and naming preserve the same displayed packet, so the local certificate obligations are exhausted by S, R, D, T, H, C, P, N. □

Theorem 6930.3 (Bishop Cauchy locator window stability)**.** *Let* $L = (S, R, D, T, H, C, P, N)$ *and* $L' = (S', R', D', T', H', C', P', N')$ *be accepted* $\mathsf{BishopCauchyLocator}^{\uparrow}$ *carriers. If* H *and* H' *record componentwise* hsame *transport preserving the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, and located tail-threshold rows, then every located Cauchy consumer reading* L' *factors through the same ordered window route*

$$S', R', D', T', H', C', P', N'.$$

No transported read may replace the displayed window by a quotient stream, replace the threshold by a selected limit, or cite countable choice or ambient completeness.

Proof. Start from the NameCert obligations of Definition 6930.2. They require the transported packet to expose a finite window, regular readback, dyadic tolerance, and located threshold before a consumer can read the locator.

Componentwise hsame transport preserves the row positions named by H and H'. Thus the target packet still opens the window row S', then reads R' under the dyadic budget D', and only then consumes the threshold row T'.

The continuation row C' replays this route; it cannot create a hidden stream quotient or selected limit. Provenance and local naming record the same route through P' and N', so every accepted transported read remains inside the displayed packet. □

Falsifiable prediction: An accepted $\mathsf{BishopCauchyLocator}^{\uparrow}$ packet cannot supply a located tail threshold unless the same packet displays the finite StreamName window, RegSeqRat readback, and DyadicRatCore tolerance used to locate it.

Independence witness: streamname, regseqrat, dyadicratcore, locatedcauchymodulus, bishoplocatedcauchyreal: none is bijective to $\mathsf{BishopCauchyLocator}^{\uparrow}$ because the locator carrier binds source windows, regular readback, dyadic tolerance, located threshold, transport, replay, provenance, and local NameCert rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyLocator}^{\uparrow}$ packet is generated from a StreamName finite-window row, a RegSeqRat readback row, a DyadicRatCore tolerance row, a located tail-threshold row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6930.1, Definition 6930.2, and Definition 6930.3 seed the finite locator route over StreamName, RegSeqRat, DyadicRatCore, located tail threshold, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopCauchyLocator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, RealUp completeness, quotient stream equality, selected limits, countable choice, trichotomy, or host real equality.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular readback, dyadic tolerance exactness, located threshold exposure, transport stability, continuation replay, provenance, and local naming for BEDC.Derived.BishopCauchyLocatorUp.

Chapter 6931

A Concrete Naming Certificate for PipelineDiscoveryClassifier$^{\uparrow}$

The PipelineDiscoveryClassifier$^{\uparrow}$ packet realizes the classifier-shift boundary of chapter 9410 as finite NameCert data. It sits after the pipeline calculus of chapter 5914, the before-and-after classifier ledger of chapter 6765, the theorem-genome evolution packet of chapter 6358, and the theorem-DNA scoring taste gate of chapter 9460. The packet classifies a discovery claim only when a concrete before/after classifier disagreement, public weight profile, theorem-genome route, ledger row, transport, replay, provenance, and local naming are all displayed.

Definition 6931.1 (Pipeline discovery classifier carrier)**.** *A* PipelineDiscoveryClassifier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, B, A, W, G, L, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DiscoveryPipelineCalculus$^{\uparrow}$ *admission row, displaying the source theory state, automation record, theorem-genome extraction, score account, lineage, classifier-shift row, gate, and refusal boundary.*
2. *B is the before-classifier row read from the* ClassifierChannelShiftLedger$^{\uparrow}$ *surface.*
3. *A is the after-classifier row read from the same classifier-channel ledger.*
4. *W is the public weight-profile row used by the scoring taste gate when the claim is presented as positive discovery rather than merely classifier movement.*
5. *G is the* TheoremGenomeEvolution$^{\uparrow}$ *genome-route row, exposing theorem-flow, recognizer, theorem-DNA, classifier-shift, mutation, selection, and ledger data.*
6. *L is the classifier-channel shift ledger row, recording record growth separately from classifier disagreement.*
7. *T is the TasteGate boundary row, naming the Nontrivial and FieldFaithful obligations required in* `lean4/BEDC/Derived/PipelineDiscoveryClassifierUp/TasteGate.lean`.
8. *H stores componentwise* hsame *transport for the pipeline, classifier, weight, genome, ledger, TasteGate, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from the pipeline admission row through the before/after classifier comparison and the public weight profile.*

10. *P is the* Pkg *provenance over* PipelineDiscoveryClassifier$^{\uparrow}$, DiscoveryPipelineCalculus$^{\uparrow}$, ClassifierChannelShiftLedger TheoremGenomeEvolution$^{\uparrow}$, *the theorem-DNA scoring taste gate,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{PipelineDiscoveryClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for positive discovery by prose alone, raw record growth, hidden weights, unledgered lineage, a classifier claim with no before/after disagreement, or a TasteGate admission that drops carrier fields.

Theorem 6931.2 (Pipeline discovery classifier NameCert obligations)**.** *For an accepted* PipelineDiscoveryClassifier$^{\uparrow}$ *carrier $K = (D, B, A, W, G, L, T, H, C, P, N)$, the local* NameCert$_{\text{PipelineDiscoveryClassifier}^{\uparrow}}$ *surface consists exactly of pipeline admission, before-classifier exposure, after-classifier exposure, public weight-profile exposure, theorem-genome route exposure, classifier-channel ledger exposure, TasteGate boundary exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the displayed carrier. The pipeline row D supplies the admission calculus, while B, A, L supply the concrete classifier-channel comparison and keep record growth separate from classifier movement.

The weight row W and genome row G supply the public scoring and theorem-genome surfaces needed before a positive discovery claim can be read. The TasteGate row T records that any Lean realization must keep the carrier nontrivial and field-faithful rather than collapsing the rows into a prose classifier.

The structural rows H, C, P, N preserve and name the same packet. Since the carrier contains no hidden weight, unledgered lineage, or prose-only discovery coordinate, the local certificate obligations are exhausted by the displayed rows. □

Theorem 6931.3 (Pipeline discovery classifier positive-claim gate)**.** *Any positive discovery claim read through an accepted* PipelineDiscoveryClassifier$^{\uparrow}$ *packet must display a concrete disagreement between the before-classifier row B and after-classifier row A, together with the public weight profile W, theorem-genome route G, classifier-channel ledger L, TasteGate boundary T, and structural rows H, C, P, N. If B and A agree on the displayed source scope, the packet may record automation or record growth through D and L, but it does not classify the claim as positive discovery.*

Proof. Start with the carrier of Definition 6931.1. The only before/after classifier comparison is the pair B, A, and the only record-growth separation row is L.

Use the NameCert obligations of Definition 6931.2. A positive read must expose the classifier comparison, public weights, theorem-genome route, TasteGate boundary, transport, replay, provenance, and local naming as one packet.

If B and A agree on the displayed source scope, the classifier-channel ledger has no disagreement row to feed the positive claim. The pipeline row D may still record generated material, but record growth without classifier disagreement is routed as non-positive by the displayed carrier boundary. □

Falsifiable prediction: A positive PipelineDiscoveryClassifier$^{\uparrow}$ claim requires a concrete before/after classifier disagreement plus a public weight profile; a trace with raw automation growth and no displayed B, A, W surface refutes the positive classifier claim.

Independence witness: discovery-pipeline-calculus, classifier-channel-shift-ledger, theorem-genome-evolution, theorem-dna-scoring-taste-gate: none is bijective to PipelineDiscoveryClassifier$^{\uparrow}$ because this packet jointly carries pipeline admission, before/after classifier rows, a public weight profile,

genome evolution, classifier-channel ledger, TasteGate boundary, transport, replay, provenance, and local NameCert rows.

The Lean boundary for $\mathsf{PipelineDiscoveryClassifier}^{\uparrow}$ is the future file `lean4/BEDC/Derived/PipelineDiscoveryC` It must provide explicit $\mathsf{Nontrivial\ PipelineDiscoveryClassifier}^{\uparrow}$ and $\mathsf{FieldFaithful\ PipelineDiscoveryClassifier}^{\uparrow}$ instances preserving $D, B, A, W, G, L, T, H, C, P, N$ as separate fields before any checked classifier theorem can serve as the carrier-alignment witness.

Closure status for $\mathsf{NameCert}_{\mathsf{PipelineDiscoveryClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{PipelineDiscoveryClassifier}^{\uparrow}$ packet is generated from pipeline admission, before-classifier, after-classifier, public weight-profile, theorem-genome, classifier-channel ledger, TasteGate boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PipelineDiscoveryClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6931.1, Definition 6931.2, and Definition 6931.3 seed the finite classifier boundary for positive pipeline discovery claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PipelineDiscoveryClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global positive-discovery certification, automated theorem proving, hidden scoring, unledgered lineage, or prose-only classifier novelty.

Upgrade path: Obligation closure requires checked carrier admission, before/after classifier disagreement, public weight-profile exposure, theorem-genome route exposure, classifier-channel ledger exactness, Nontrivial PipelineDiscoveryClassifierUp, and FieldFaithful PipelineDiscoveryClassifierUp in TasteGate.lean.

Chapter 6932

A Concrete Naming Certificate for DecimalEndpointAmbiguity$^{\uparrow}$

Following Bishop and Bridges on constructive real approximation and the classical terminating-decimal convention, the DecimalEndpointAmbiguity$^{\uparrow}$ packet records the finite boundary where a terminating decimal window and a repeating-nine window present the same endpoint route. It sits after chapter 1417, chapter 1429, chapter 5150, and chapter 3952, and it hands all real-facing reads through chapter 301, chapter 291, and chapter 38. The packet names only the finite endpoint ambiguity surface for inspected decimal windows; it does not assert uniqueness of all decimal expansions, select a host real representative, or quotient arbitrary decimal streams.

Definition 6932.1 (DecimalEndpointAmbiguity carrier)**.** *A* DecimalEndpointAmbiguity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, E, S, M, Q, R, Y, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DecimalExpansion$^{\uparrow}$ *source ledger for the inspected finite decimal prefixes.*
2. *E is the* DecimalEndpointNormalization$^{\uparrow}$ *endpoint row that records the terminating side and the repeating-nine side inside the same window.*
3. *S is the* SignedDecimalExpansion$^{\uparrow}$ *sign and zero-window row, when the endpoint ambiguity is read through a signed presentation.*
4. *M is the* RealDecimalNormalForm$^{\uparrow}$ *normal-form route reached after the endpoint row has been displayed.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row attached to the same endpoint window.*
6. *R is the* RegSeqRat$^{\uparrow}$ *handoff row produced from the normalized endpoint window.*
7. *Y is the terminal* Real$^{\uparrow}$ *seal row reached only after D, E, S, M, Q, R have been exposed.*
8. *H records componentwise* hsame *transport for the decimal source, endpoint-normalization, signed-window, normal-form, dyadic, regular, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the decimal window through endpoint normalization, normal-form readback, dyadic tolerance, regular handoff, and the real seal.*
10. *P is the* Pkg *provenance over* DecimalEndpointAmbiguity$^{\uparrow}$, DecimalExpansion$^{\uparrow}$, DecimalEndpointNormalization$^{\uparrow}$, SignedDecimalExpansion$^{\uparrow}$, RealDecimalNormalForm$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalEndpointAmbiguity}^{\uparrow}}$ *row.*
The classifier compares accepted endpoint-ambiguity packets only through $D, E, S, M, Q, R, Y, H, C, P, N$. It has no coordinate for global decimal uniqueness, quotient stream equality, selected representatives, arbitrary real-name trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DecimalEndpointAmbiguityUp.DecimalEndpointAmbiguityTasteGate single carrier alignment`

Theorem 6932.2 (DecimalEndpointAmbiguity NameCert obligations)**.** *For every accepted* DecimalEndpointAmbig *carrier*

$$A = (D, E, S, M, Q, R, Y, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DecimalEndpointAmbiguity}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, endpoint-pair exposure through E, signed-window compatibility through S, normal-form routing through M, dyadic tolerance exposure through Q, regular rational handoff through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through Y, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the terminating-versus-repeating-nine boundary must read that finite surface before treating the endpoint as real-facing data.*

Proof. Carrier admission is projection from Definition 6932.1. The visible decimal coordinates are exactly D, E, S, M, Q, R, Y, while H, C, P, N carry transport, replay, provenance, and local naming.

Endpoint-pair exposure is local to E. The terminating side and the repeating-nine side are compared only because the endpoint-normalization row lists them inside the same inspected finite window.

Signed-window compatibility uses S only as a displayed signed presentation of that same endpoint window. It cannot introduce a second magnitude stream or an endpoint selector outside the packet.

The real-facing route is ordered by M, Q, R, Y. Normal form is read first, then dyadic tolerance, then the regular rational handoff, and only then the terminal real seal. The structural rows preserve this finite route, so no global uniqueness, quotient equality, chosen representative, trichotomy, countable-choice schedule, or completeness row can be consumed. □

Theorem 6932.3 (DecimalEndpointAmbiguity real normal-form route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalEndpointAmbiguity}^{\uparrow}$ *packet factors through the finite route*

$\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalEndpointNormalization}^{\uparrow}$, $\mathsf{SignedDecimalExpansion}^{\uparrow}$, $\mathsf{RealDecimalNormalForm}^{\uparrow}$, RegSeqR

The route records the terminating and repeating-nine windows as finite endpoint data before the normal-form, dyadic, regular-rational, and real-seal rows are public.

Proof. Start with an accepted packet $A = (D, E, S, M, Q, R, Y, H, C, P, N)$. The source row D supplies only finite decimal prefixes, and E supplies the endpoint normalization row for the same inspected window.

The signed row S is read before the normal-form route M. Thus positive, negative, and zero-window presentations are attached to the displayed endpoint row rather than to an independent stream representative.

Now read the arithmetic handoff rows. The dyadic tolerance row Q and the regular rational row R are reached from the normal-form route, and the terminal real seal Y is downstream of both.

Finally apply the structural rows H, C, P, N. They transport, replay, source, and name the same finite endpoint surface, so the real read is a route through the displayed packet rather than a proof of decimal uniqueness. □

Theorem 6932.4 (DecimalEndpointAmbiguity nonescape)**.** *An accepted* $\mathsf{DecimalEndpointAmbiguity}^{\uparrow}$ *packet exports no global decimal uniqueness theorem, quotient stream equality, selected endpoint representative, arbitrary real-name trichotomy, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness. Every endpoint read remains inside the displayed rows* $D, E, S, M, Q, R, Y, H, C, P, N$.

Proof. Inspect the carrier inventory. The only endpoint ambiguity coordinate is E, and it is tied to the finite decimal source D and signed-window row S.

The normal-form and real-facing rows M, Q, R, Y are downstream of that finite endpoint row. They can transport an already displayed terminating or repeating-nine window to a regular rational or real seal, but they cannot select representatives for arbitrary decimal streams.

The structural rows H, C, P, N do not add carrier data. They preserve, replay, source, and name the displayed route. Since the excluded objects are not coordinates of A, no accepted consumer can read them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalEndpointAmbiguity}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalEndpointAmbiguity}^{\uparrow}$ packet is generated from finite DecimalExpansion source rows, DecimalEndpointNormalization endpoint rows, SignedDecimalExpansion sign and zero-window rows, RealDecimalNormalForm routes, DyadicRatCore tolerance rows, RegSeqRat handoff rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DecimalEndpointAmbiguity}^{\uparrow}}$, seed)

Scope closed: Definition 6932.1, Definition 6932.2, Definition 6932.3, and Definition 6932.4 seed the finite trailing-nine and trailing-zero endpoint ambiguity surface for decimal Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DecimalEndpointAmbiguity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close uniqueness of all decimal expansions, quotient stream equality, selected representatives, arbitrary real-name trichotomy, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-pair exposure, signed-window compatibility, normal-form routing, dyadic tolerance exposure, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DecimalEndpointAmbiguityUp.

Chapter 6933

A Concrete Naming Certificate for $\mathsf{RHZeroWindow}^{\uparrow}$

Following the Riemann zeta zero tradition and the finite RH-route boundary of chapter 9336, the $\mathsf{RHZeroWindow}^{\uparrow}$ packet records bounded inspection windows for supplied zeta-zero witnesses. It sits downstream of chapter 73, chapter 72, chapter 9342, and chapter 39. The packet names a finite witness surface only: it does not prove RH, enumerate all zeros, decide all zero locations, or supply a global route-closure oracle.

Definition 6933.1 (RHZeroWindow carrier)**.** *An* $\mathsf{RHZeroWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Z, S, A, X, B, T, H, C, P, N)$$

with the following displayed rows:

1. *Z is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero-witness row for the inspected complex history.*
2. *S is the* $\mathsf{CritStrip}^{\uparrow}$ *row recording the critical-strip boundary or obstruction data visible to the window.*
3. *A is the* $\mathsf{ZetaCont}^{\uparrow}$ *continuation row through which the zeta value is read.*
4. *X is the* $\mathsf{Complex}^{\uparrow}$ *point row for the inspected source history and zero-value comparison.*
5. *B is the finite bound row naming the inspected height, depth, precision, or source-prefix budget of the window.*
6. *T is the provenance row tying the zero row, strip row, continuation row, complex point, and finite bound to the same RH-route request.*
7. *H records componentwise* hsame *transport for $Z, S, A, X, B, T, C, P, N$.*
8. *C records displayed* Cont *replay for the zero, strip, continuation, complex, bound, and provenance rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RHZeroWindow}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RHZeroWindow}^{\uparrow}}$ *row.*

The classifier compares accepted zero-window packets only through $Z, S, A, X, B, T, H, C, P, N$. It has no coordinate for a proof of RH*, a counterexample, a complete zero enumeration, host complex equality, quotient zero-set equality, or a global zero-location decider.*

Theorem 6933.2 (RHZeroWindow NameCert obligations)**.** *For every accepted* $\mathsf{RHZeroWindow}^{\uparrow}$ *carrier*

$$W = (Z, S, A, X, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RHZeroWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, zeta-zero source exposure through* Z*, critical-strip exposure through* S*, continuation exposure through* A*, complex point exposure through* X*, finite-bound exposure through* B*, route-provenance exposure through* T*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 6933.1. The zero-facing rows are exactly Z, S, A, X, B, T, while H, C, P, N are the structural transport, replay, package, and naming rows.

The zeta-zero source is supplied, not searched for. The row Z records a zero witness already presented by $\mathsf{ZetaZeros}^{\uparrow}$, and the strip row S only records the finite critical-strip boundary visible to this same window.

The continuation and complex rows are also finite inputs. The row A gives the $\mathsf{ZetaCont}^{\uparrow}$ route used for the inspected zeta value, while X names the complex-history point and zero-value comparison consumed by the window.

Finally the bound and provenance rows B, T keep the read inside one finite inspection budget. Since no other carrier coordinate is present, the packet cannot export an RH proof, a counterexample, all zeros, host equality, quotient zero-set equality, or a global zero-location decider. □

Theorem 6933.3 (RHZeroWindow zeta source route)**.** *Every zero-facing read of an accepted* $\mathsf{RHZeroWindow}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ZetaZeros}^{\uparrow},\ \mathsf{CritStrip}^{\uparrow},\ \mathsf{ZetaCont}^{\uparrow},\ \mathsf{Complex}^{\uparrow}.$$

The finite bound row B *and route-provenance row* T *must be displayed before the packet can be consumed by an RH-route boundary argument.*

Proof. Start with an accepted packet $W = (Z, S, A, X, B, T, H, C, P, N)$. The zero source row Z is the first public zero-facing datum, and it is read together with the critical-strip row S rather than with an ambient zero search.

The continuation row A and complex row X identify the value route and the inspected point. They are finite rows of the same packet, so a consumer cannot replace them by host analytic continuation or host complex equality.

The bound row B and provenance row T then fix the inspection budget and the RH-route request. Only after those rows are exposed may the structural rows H, C, P, N transport, replay, source, and name the zero window. □

Theorem 6933.4 (RHZeroWindow nonescape)**.** *An accepted* $\mathsf{RHZeroWindow}^{\uparrow}$ *packet records finite inspection windows for zeta-zero witnesses and exports no proof of* RH*, no counterexample, no complete zero enumeration, no host complex equality, no quotient zero-set equality, and no decision procedure for all zeros.*

Proof. Inspect the packet coordinates. The only zero data are the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row Z, the critical-strip row S, the continuation row A, the complex point row X, the finite bound row B, and the provenance row T.

The rows H, C, P, N preserve and name that displayed route. They do not add a new witness, a zero search, an equality quotient, or an RH decision datum.

Thus every accepted consumer remains inside the finite window $Z, S, A, X, B, T, H, C, P, N$, and the excluded global conclusions are outside the carrier surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RHZeroWindow}^{\uparrow}}$.

Constructive story: An $\mathsf{RHZeroWindow}^{\uparrow}$ packet is generated from a supplied ZetaZeros row, a critical-strip row, a ZetaCont continuation row, a Complex point row, a finite inspection-bound row, route provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHZeroWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6933.1, Definition 6933.2, Definition 6933.3, and Definition 6933.4 seed the finite zero-inspection surface for RH-route boundary consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RHZeroWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, counterexamples to RH, complete zero enumeration, host complex equality, quotient zero-set equality, all-zero decision, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zeta-zero source exposure, critical-strip exposure, continuation exposure, complex point exposure, finite-bound exposure, route-provenance exposure, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RHZeroWindowUp.

Chapter 6934

A Concrete Naming Certificate for $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real arithmetic, the $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ packet records the finite route by which two regular Cauchy real names remain readable after an algebraic operation. It sits over the finite observation windows of chapter 126, the dyadic tolerance budget of chapter 291, the regular rational readback of chapter 301, the product-facing operation route of chapter 464, and the terminal real-name seal of chapter 38. The packet is a completion-facing handoff for displayed algebraic reads; it is not a quotient-real operation, selected limit, host completeness theorem, or ambient field structure for $\mathsf{Real}^{\uparrow}$.

Definition 6934.1 (Regular Cauchy algebra closure carrier)**.** *A* $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, R, O, S_L, S_R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source rows.*
2. *O is the operation selector, naming the displayed algebraic operation to be read from the two sources.*
3. *S_L and S_R are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the two sources.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, recording the requested precision, operation slack, and rational comparison allowances.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the operation output after the synchronized windows and dyadic budget have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the output read.*
7. *H records componentwise* hsame *transport for the source, operation, window, tolerance, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the same finite algebraic route.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, host real arithmetic, selected Cauchy limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6934.2 (Regular Cauchy algebra closure NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ *carrier* $A = (L, R, O, S_L, S_R, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgeb}}$ *surface consists exactly of carrier admission, two-source* $\mathsf{RegSeqRat}^{\uparrow}$ *source exposure, operation selection, synchronized* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget exposure, output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6934.1. The only source rows are L and R, and the only algebraic choice is the operation selector O.

The finite-window induction is simultaneous over the two synchronized stream rows. At each window stage, S_L and S_R expose the source observations, D supplies the dyadic tolerance budget for the selected operation, and Q records the resulting regular rational output readback.

The terminal seal and structural rows do not add analytic data. The row E seals the displayed output read for $\mathsf{Real}^{\uparrow}$ consumers, while H, C, P, N transport, replay, source, and name the same finite packet.

Since no further row exists, quotient real equality, host arithmetic, selected limits, countable choice, and ambient completeness cannot be exported by the local NameCert surface. □

Theorem 6934.3 (Regular Cauchy algebra closure operation handoff)**.** *Every algebraic read exported by an accepted* $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ *packet factors through the finite route*

$$L, R, O, S_L, S_R, D, Q, E, H, C, P, N.$$

Thus a $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer receives the output only after the two source names, operation selector, synchronized stream windows, dyadic tolerance budget, output readback, seal, transport, replay, provenance, and local naming rows have all been displayed.*

Proof. Start with an operation request. The carrier gives the request only one selector row O and two source rows L, R, so the operation cannot be read before both regular Cauchy sources are visible.

Open the synchronized windows by finite induction. The base stage reads the initial windows from S_L and S_R; the induction step extends both windows together and updates the tolerance budget D before the output readback row Q is consumed.

Now route the result to downstream consumers. The $\mathsf{CauchyProduct}^{\uparrow}$ handoff reads the displayed product-facing operation ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback is the row Q, and the $\mathsf{Real}^{\uparrow}$ endpoint is sealed only at E.

Finally, H, C, P, N keep the handoff inside the accepted packet. A consumer can forget unused rows for a projection, but it cannot import quotient real arithmetic, host completeness, selected limits, or countable choice because no such coordinate is present. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ read is refuted if a Real-facing algebraic output is consumed without first displaying the two RegSeqRat source rows, operation selector, synchronized StreamName windows, DyadicRatCore tolerance budget, output RegSeqRat readback, RealUp seal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchyproduct: RegularCauchyAlgebraClosur is not bijective to any listed sibling because it jointly carries two source names, an operation selector, synchronized finite windows, a dyadic operation budget, output readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}$ packet is generated from two RegSeqRat source rows, an operation selector, synchronized StreamName windows, a DyadicRatCore tolerance budget, an output RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6934.1, Definition 6934.2, and Definition 6934.3 seed the finite algebraic closure route for regular Cauchy names over synchronized StreamName windows, DyadicRatCore budgets, RegSeqRat readback, CauchyProduct handoff, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real arithmetic, selected limits, countable choice, host real field structure, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-source readback, operation selection, synchronized window induction, dyadic tolerance accounting, output RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyAlgebraClosureUp.

Chapter 6935

A Concrete Naming Certificate for ErdosSzekeresMonotoneSubsequence$^{\uparrow}$

Following Erdos and Szekeres on finite monotone subsequences, the ErdosSzekeresMonotoneSubsequence$^{\uparrow}$ packet records the finite combinatorial surface that turns a long enough ordered list into either an increasing branch of prescribed length or a decreasing branch of prescribed length. The carrier sits after the finite spine discipline of chapter 36, the comparison surface of chapter 1378, and the finite coloring handoff of chapter 203. It is a displayed finite NameCert route, not an infinite subsequence selector, a countable-choice principle, or a host order-completeness theorem.

Definition 6935.1 (Erdos–Szekeres monotone subsequence carrier)**.** *An* ErdosSzekeresMonotoneSubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, I, D, L, O, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the source window ledger, recording the inspected finite list entries and the requested increasing and decreasing lengths.*
2. *I is the increasing-branch ledger, listing the retained positions and order comparisons for an accepted increasing branch when that side is selected.*
3. *D is the decreasing-branch ledger, listing the retained positions and order comparisons for an accepted decreasing branch when that side is selected.*
4. *L is the* List$^{\uparrow}$ *spine row that fixes the finite source order and the retained branch subspines.*
5. *O is the* Order$^{\uparrow}$ *comparison row used to classify each inspected pair of retained positions.*
6. *R is the* Ramsey$^{\uparrow}$ *coloring handoff row for the finite pair-coloring route that separates increasing-facing and decreasing-facing comparisons.*
7. *H stores componentwise* hsame *transport for windows, branch rows, list spines, order comparisons, Ramsey rows, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay from W to L, O, from L, O to R, and from R to the selected branch row I or D.*
9. *P is the* Pkg *provenance over* ErdosSzekeresMonotoneSubsequence$^{\uparrow}$, List$^{\uparrow}$, Order$^{\uparrow}$, Ramsey$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for countable choice, an infinite subsequence object, a host total-order oracle, order completeness, or ambient sequence compactness.

Theorem 6935.2 (Erdos–Szekeres monotone subsequence NameCert surface)**.** *For every accepted* $\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}$ *carrier*

$$E = (W, I, D, L, O, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite source window, one selected branch ledger, the* $\mathsf{List}^{\uparrow}$ *spine row, the* $\mathsf{Order}^{\uparrow}$ *comparison row, the* $\mathsf{Ramsey}^{\uparrow}$ *coloring handoff row, componentwise transport, replay, provenance, and the local name.*

Proof. Project the carrier of Definition 6935.1. The rows $W, I, D, L, O, R, H, C, P, N$ are the only coordinates available to the local classifier, so carrier admission is admission of those displayed rows.

The finite source discipline is read through chapter 36. The row L fixes the inspected list spine and the retained branch subspines, while W records the finite window and the requested branch lengths.

The comparison and coloring data are then read through chapter 1378 and chapter 203. The order row O supplies the pairwise comparison cells, and the Ramsey row R records the finite two-color handoff that selects either the increasing branch ledger I or the decreasing branch ledger D.

Finally the structural rows H, C, P, N transport, replay, source, and name that same finite route. Since no coordinate carries an infinite selector, countable-choice schedule, host order oracle, completeness principle, or compactness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6935.3 (Erdos–Szekeres Ramsey handoff)**.** *Every* $\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}$ *consumer that reaches the selected monotone branch must first read the finite list spine through chapter 36, classify retained pairs through chapter 1378, and pass the resulting two-color finite graph row through chapter 203. The exported branch is therefore a finite route*

$$W \longmapsto L, O \longmapsto R \longmapsto I \text{ or } D$$

rather than an independent subsequence selector.

Proof. Start with the source window. The row W exposes only finitely many inspected entries, and L supplies their $\mathsf{List}^{\uparrow}$ order, so the consumer cannot begin from an infinite sequence or from an unrecorded subsequence choice.

Next classify the retained pairs. The row O records the $\mathsf{Order}^{\uparrow}$ comparison cells used to color each pair as increasing-facing or decreasing-facing, and the replay row C sends those cells to the finite Ramsey handoff row R.

The Ramsey consumer then returns one displayed side of the dichotomy. If the increasing side is selected, the branch is read from I; if the decreasing side is selected, the branch is read from D. In both cases H, C, P, N preserve and name the same finite packet, so the handoff exports only the displayed list, order, and Ramsey route. □

Lean4 (statement) `BEDC.Derived.ErdosSzekeresMonotoneSubsequenceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6935.1, Definition 6935.2, and Definition 6935.3 seed the finite carrier, local NameCert surface, and Ramsey/List/Order handoff for the monotone subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim infinite subsequence selection, countable choice, a host total-order oracle, order completeness, ambient compactness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch ledger stability, ListUp spine exactness, OrderUp comparison exactness, and RamseyUp handoff non-escape for BEDC.Derived.ErdosSzekeresMonotoneSubsequenceUp.

Constructive story: An $\mathsf{ErdosSzekeresMonotoneSubsequence}^{\uparrow}$ packet is generated from a finite source window, increasing and decreasing branch ledgers, a ListUp spine row, an OrderUp comparison row, a RamseyUp coloring handoff row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 6936

A Concrete Naming Certificate for KolmogorovZeroOneLaw$^{\uparrow}$

Following Kolmogorov on independent coordinates and tail events, the KolmogorovZeroOneLaw$^{\uparrow}$ packet records the finite-window surface behind the zero-one law as displayed BEDC data. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the independence ledger of chapter 187, the finite repeated-hit boundary of chapter 2357, and the event-family surface of chapter 5186. The packet names tail-event inspection through finite windows and a visible zero-one verdict row; it is not a host product-measure theorem, a completed tail sigma algebra, a quotient almost-sure equality, or an ambient probability-limit principle.

Definition 6936.1 (Kolmogorov zero-one law carrier)**.** *A* KolmogorovZeroOneLaw$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\Omega, X, W, A, I, Z, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* ProbSpace$^{\uparrow}$ *source row carrying the probability measure, total-event endpoint, and event classifier.*
2. X *is the displayed* RandomVar$^{\uparrow}$ *event-sequence row whose coordinates generate the inspected tail windows.*
3. W *is the finite tail-window row, recording the displayed initial cutoff, surviving tail coordinates, and event names inspected beyond that cutoff.*
4. A *is the* SigmaAlgebra$^{\uparrow}$ *admission row for the finite tail events exposed by* W*.*
5. I *is the* Independence$^{\uparrow}$ *ledger row, recording the finite independence witnesses used to separate initial windows from tail windows.*
6. Z *is the zero-one verdict row, recording that the inspected tail-event probability is routed to the displayed* 0_R *or* 1_R *endpoint only through the preceding finite rows.*
7. H *records componentwise* hsame *transport for the probability source, random-variable row, finite tail windows, event admission, independence ledger, verdict row, replay, provenance, and local naming row.*
8. C *records displayed* Cont *replay from a tail-event request through* Ω, X, W, A, I, Z *in that order.*
9. P *is the* Pkg *provenance over* KolmogorovZeroOneLaw$^{\uparrow}$, ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, Independence$^{\uparrow}$, BorelCantelli$^{\uparrow}$, SigmaAlgebra$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{KolmogorovZeroOneLaw}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a completed infinite product measure, a host tail sigma algebra, quotient almost-sure equality, countable-choice tail selectors, or an ambient proof that all external tail events have probability zero or one.

Theorem 6936.2 (Kolmogorov zero-one law NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovZeroOneLaw}^{\uparrow}$ *carrier*

$$K = (\Omega, X, W, A, I, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KolmogorovZeroOneLaw}^{\uparrow}}$ *surface consists exactly of probability-source admission through Ω; random-variable event exposure through X; finite tail-window admission through W; sigma-algebra event admission through A; independence-ledger exposure through I; zero-one verdict routing through Z; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6936.1. The tuple exposes the probability source, random-variable event sequence, finite tail windows, sigma-algebra admission, independence ledger, zero-one verdict, transport, replay, provenance, and local naming rows.

The tail-event read is finite-window data. It first displays the cutoff and surviving coordinates in W, then reads event admission through A and independence evidence through I before the verdict row Z can be inspected.

The structural rows H, C, P, N preserve and replay the same packet. Since no coordinate supplies a completed tail sigma algebra, quotient almost-sure equality, product-measure extension, or countable-choice selector, the listed rows exhaust the local NameCert obligations. □

Theorem 6936.3 (Kolmogorov zero-one law tail-event induction)**.** *Every accepted* $\mathsf{KolmogorovZeroOneLaw}^{\uparrow}$ *tail-event consumer must inspect tail events by finite-window induction over W. At the base cutoff it reads Ω, X, A to admit the displayed event; at each successor cutoff it transports the same tail request through H, replays the independence comparison through C and I, and only then reads the verdict row Z. Thus the public zero-one read is a finite NameCert route and not a host tail sigma-algebra theorem.*

Proof. The base cutoff is a carrier projection. Rows Ω and X give the probability source and coordinate event sequence, while A admits the displayed event in the sigma-algebra surface.

For the successor cutoff, use the finite tail-window row W. Componentwise transport H keeps the same source, event, admission, and independence rows attached to the shifted window.

Replay C and then read the independence ledger I. The verdict row Z is therefore consumed only after the finite cutoff, event admission, and independence comparison are visible.

Since the carrier has no row for a completed tail sigma algebra, quotient almost-sure equality, countable-choice tail selector, or ambient product-measure proof, the induction exports only the displayed finite-window route. □

Falsifiable prediction: Every accepted $\mathsf{KolmogorovZeroOneLaw}^{\uparrow}$ read displays a ProbSpace source row, RandomVar event-sequence row, finite tail-window row, SigmaAlgebra event-admission row, Independence ledger row, zero-one verdict row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: probspace, randomvar, independence, borel-cantelli, sigma-algebra: $\mathsf{KolmogorovZeroOneLaw}^{\uparrow}$ is not bijective to any listed sibling because it binds finite tail windows,

event admission, independence separation, and a zero-one verdict row into one probability tail-event packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovZeroOneLaw}\uparrow}$.

Constructive story: A $\mathsf{KolmogorovZeroOneLaw}^{\uparrow}$ packet is generated from a ProbSpace source row, RandomVar event-sequence row, finite tail-window row, SigmaAlgebra event-admission row, Independence ledger row, zero-one verdict row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovZeroOneLaw}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6936.1, Definition 6936.2, and Definition 6936.3 seed the finite probability-source, event-sequence, tail-window, event-admission, independence-ledger, zero-one-verdict, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovZeroOneLaw}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host product measures, completed tail sigma algebras, quotient almost-sure equality, countable-choice tail selectors, ambient probability limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tail-window exactness, event-admission stability, independence-ledger exposure, zero-one verdict routing, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovZeroOneLawUp.

Chapter 6937

A Concrete Naming Certificate for $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ packet records the constructive Cauchy-real reading used at the current L10 Real and Completion boundary without forming quotient classes of streams. It binds the dyadic tolerance source of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the pointwise locator induction route of chapter 4225. The certificate is a displayed finite-row interface for Real-facing consumers; it is not a quotient construction, a selected limit operator, or an ambient completeness theorem.

Definition 6937.1 (Quotient-free Cauchy real carrier)**.** *A* $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, W, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying signed mantissa rows, positive-unary scale rows, denominator rows, distance rows, radius rows, and local dyadic naming data.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule that tells each consumer which finite observation windows of the Cauchy stream are public.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row obtained from the displayed windows and dyadic tolerance data.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed after the dyadic, stream, and regular-rational faces have been displayed.*
5. *L is the* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *route that records pointwise locator reads over the same displayed finite windows.*
6. *H records componentwise* hsame *transport for the dyadic, stream, readback, seal, locator, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for finite-window tolerance reads.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealLocatorPointwiseInd}$ BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}}$ *row.*

Carrier comparison is rowwise over $D, W, Q, R, L, H, C, P, N$. The carrier has no coordinate for quotient real equality, quotient streams, countable choice, selected limits, trichotomy, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6937.2 (Quotient-free Cauchy real classifier stability)**.** *Let*

$$X = (D, W, Q, R, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ *carrier. Replacing the dyadic tolerance ledger, finite stream schedule, regular-rational readback, Real seal, and pointwise locator induction rows by displayed componentwise* hsame *rows, and replaying the replacement through C, preserves the public Cauchy-real classifier read. The classifier remains the finite route*

$$D, W \longmapsto Q \longmapsto R, L \longmapsto H, C, P, N$$

and does not inspect quotient real equality, quotient stream equality, countable choice, selected limits, trichotomy, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6937.1. The visible classifier source is exactly the dyadic tolerance ledger D together with the finite stream schedule W.

The readback row Q is downstream of those two rows, and the Real seal R is downstream of the readback. The locator induction route L reads the same displayed window and tolerance surface through chapter 4225.

The transport row H records the componentwise replacement, and C replays that replacement through the displayed route. Since P only records provenance and N only names the local packet, no additional classifier coordinate is introduced.

Thus the public classifier is stable under the displayed transport. The excluded quotient, choice, order, and completeness objects have no row in the carrier, so they cannot affect the classifier read. □

Theorem 6937.3 (Quotient-free Cauchy real finite-window tolerance readback)**.** *For every accepted* $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ *carrier*

$$X = (D, W, Q, R, L, H, C, P, N),$$

each Real-facing tolerance request is answered by a finite window of W, the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and the terminal seal row R. The answer is a BEDC-visible finite row readback; it is not a representative of a quotient stream, a selected limit, a countable-choice selection, a trichotomy decision, or an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real-facing tolerance request at the accepted carrier. The only public schedule for inspecting the Cauchy stream is the finite $\mathsf{StreamName}^{\uparrow}$ row W, so the request first selects a displayed finite window rather than an infinite representative.

Read the dyadic bound for that window from D. This uses the signed mantissa, scale, denominator, distance, and radius rows supplied by chapter 291.

Now pass to Q. The regular-rational readback consumes the same finite window and dyadic ledger, matching the route of chapter 301. The Real-facing seal R is then read only after those rows are displayed.

The remaining rows L, H, C, P, N record pointwise locator compatibility, transport, replay, provenance, and local naming. Since none of these rows stores quotient stream equality, selected limits, countable choice, trichotomy, metric completeness, or ambient Real completeness, the tolerance answer is exactly the stated finite-window readback. □

Theorem 6937.4 (Quotient-free Cauchy real locator-induction compatibility)**.** *Let*

$$X = (D, W, Q, R, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ *carrier. Every pointwise locator induction read in* L *is compatible with the same finite-window tolerance route used by* D, W, Q, R*: the branch ledger opens a displayed* $\mathsf{StreamName}^{\uparrow}$ *window, reads the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and reaches the same* $\mathsf{Real}^{\uparrow}$ *seal before the local* $\mathsf{NameCert}$ *row is public.*

Proof. Project the locator row L. By chapter 4225, each accepted pointwise branch is read through a displayed StreamName window, DyadicRatCore tolerance row, RegSeqRat readback, and Real seal.

Compare those rows with the carrier rows W, D, Q, R. The $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ packet stores no second stream schedule or second dyadic ledger, so the branch ledger in L must use the same finite-window tolerance route already exposed by the classifier and readback rows.

The componentwise transport H and replay C preserve this route, while P and N source and name it locally. Therefore the locator-induction branch becomes public only through the same displayed finite rows.

The compatibility claim uses no quotient real equality, quotient streams, countable choice, selected limits, trichotomy, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects occurs in the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6937.1, Definition 6937.2, Definition 6937.3, and Definition 6937.4 seed the quotient-free Cauchy-real interface over DyadicRatCore, StreamName, RegSeqRat, RealUp, CauchyRealLocatorPointwiseInduction, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, quotient streams, countable choice, selected limits, trichotomy, metric completeness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite-window tolerance readback, locator-induction compatibility, and local NameCert registration for BEDC.Derived.CauchyRealsWithoutQuotientUp.

Constructive story: A $\mathsf{CauchyRealsWithoutQuotient}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, one StreamName finite schedule, a RegSeqRat readback row, a RealUp seal row, a CauchyRealLocatorPointwiseInduction route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6938

A Concrete Naming Certificate for $\mathsf{CauchyBishopBridge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyBishopBridge}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name can be read back as Bishop located-Cauchy data without passing through quotient real equality or an ambient completeness principle. It sits between the certified stream-name surface of chapter 126, the regular rational sequence carrier of chapter 301, the dyadic tolerance ledger of chapter 291, the located-Cauchy request surface of chapter 6930, and the terminal real-name seal of chapter 38.

Definition 6938.1 (Cauchy-Bishop bridge carrier)**.** *A* $\mathsf{CauchyBishopBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, L, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row supplying the inspected windows of the rational name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the positive dyadic budgets used by the readback.*
4. *L is the* $\mathsf{BishopCauchyLocator}^{\uparrow}$ *located-Cauchy request row, recording the finite witness that a requested precision has a displayed tail location.*
5. *W is the bridge-witness row matching each located request in L with a regular readback and tolerance entry in R, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, L, W have been displayed.*
7. *H records componentwise* hsame *transport for the stream, readback, tolerance, locator, bridge-witness, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyBishopBridge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCauchyLocator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBishopBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, L, W, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected limit, countable choice, host real completeness, or an uninspected infinite sequence.

Theorem 6938.2 (Cauchy-Bishop bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBishopBridge}^{\uparrow}$ *carrier* $B = (S, R, D, L, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyBishopBridge}^{\uparrow}}$ *surface consists exactly of stream-window exposure, regular rational readback, dyadic tolerance accounting, located-Cauchy request coverage, bridge-witness matching, terminal Real seal routing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6938.1. The packet exposes precisely the rows $S, R, D, L, W, E, H, C, P, N$, so carrier admission cannot appeal to an external sequence, quotient class, or host completeness fact.

The regular Cauchy side is read through S, R, D. A consumer first inspects a finite stream window in S, then reads the regular rational evidence in R, and then checks the requested precision against the dyadic tolerance ledger D.

The Bishop located-Cauchy side is read through L, W. Each located request in L must be matched in W to the already displayed regular readback and tolerance rows, so the bridge does not choose a new representative or import countable choice.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since no additional coordinate exists, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 6938.3 (Regular readback to Bishop located-Cauchy data)**.** *Every accepted* $\mathsf{CauchyBishopBridge}^{\uparrow}$ *packet turns a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *read over* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances into a* $\mathsf{BishopCauchyLocator}^{\uparrow}$ *read only through*

$$S, R, D, L, W, H, C, P, N.$$

Thus a located-Cauchy consumer receives finite locator data after the regular readback and dyadic tolerance rows have already been displayed.

Proof. Start with the regular readback rows S, R, D. The stream row supplies the inspected finite windows, the regular-sequence row supplies the rational readback, and the dyadic row supplies the positive tolerance budget.

Now inspect the located request row L. For each requested precision, the bridge-witness row W names the displayed window and tolerance entry that covers the requested tail location. This is a finite matching operation inside the packet, not an appeal to an infinite choice of moduli.

The structural rows H, C, P, N preserve the same matching data under transport, replay, provenance, and local naming. Therefore the located-Cauchy read is available exactly through the bridge rows and not through quotient real equality or host completion. □

Theorem 6938.4 (Cauchy-Bishop bridge Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchyBishopBridge}^{\uparrow}$ *packet factors through the displayed route*

$$S, R, D, L, W, E, H, C, P, N.$$

The terminal seal row E does not export quotient equality, selected limits, countable choice, host real completeness, or an uninspected infinite sequence.

Proof. Use the NameCert obligations of Definition 6938.2. They force every accepted export to expose stream windows, regular readback, dyadic tolerance, located requests, bridge witnesses, and the terminal seal before a Real-facing consumer reads the packet.

Apply the readback theorem Definition 6938.3. The located-Cauchy data is already bound to displayed regular Cauchy rows, so the Real-facing seal cannot replace the route by a quotient representative or a selected limit.

The rows H, C, P, N keep the seal inside the same finite packet. Since the carrier has no coordinate for the excluded objects, no Real-seal read can escape the displayed bridge route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBishopBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyBishopBridge}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat regular readback, a DyadicRatCore tolerance ledger, BishopCauchyLocator requests, bridge witnesses, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBishopBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6938.1, Definition 6938.2, Definition 6938.3, and Definition 6938.4 seed the finite bridge route from regular Cauchy readback to Bishop located-Cauchy requests and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBishopBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, host real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular readback, located request coverage, bridge-witness matching, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyBishopBridgeUp.

Chapter 6939

A Concrete Naming Certificate for $\mathsf{ClosureCapstoneRealization}^{\uparrow}$

The $\mathsf{ClosureCapstoneRealization}^{\uparrow}$ packet realizes the closure-capstone route of chapter 9387 as finite NameCert data. It packages the refusal of unwitnessed closure, the generator-induction replacement, explicit transport, replay, provenance, and local naming into one carrier that can be inspected without importing choice, quotient soundness, propositional extensionality, or a meta-level oracle.

The generator-induction and refusal rows consume the sibling boundaries of chapter 515, chapter 566, and chapter 1012 before the capstone packet exposes its local NameCert row.

Definition 6939.1 (Closure capstone realization carrier)**.** *A* $\mathsf{ClosureCapstoneRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, G, R, T, E, P, N)$$

with the following displayed rows:

1. *A is the closure-axis row, recording the term, type, and proposition strata named by the capstone.*
2. *G is the generator-induction row, recording the finite replacement route used when a witness, representative, or equality token is demanded.*
3. *R is the refusal row, recording that unwitnessed choice, quotient representative selection, and proposition equality by extensionality are not accepted as kernel inputs.*
4. *T is the transport row, recording explicit* hsame *and* psame *transport through displayed implications and classifier-respecting routes.*
5. *E is the replay row, recording the* Cont *trace by which the replacement route is recomputed from the finite rows.*
6. *P is the* Pkg *provenance over* $\mathsf{ClosureCapstoneRealization}^{\uparrow}$*, the closure-capstone vision route,* BHist*,* hsame*,* psame*,* Cont*, and* NameCert*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosureCapstoneRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through A, G, R, T, E, P, N. It has no coordinate for Classical.choice*,* Quot.sound*,* propext*, hidden representatives, erased implication directions, or unwitnessed existential closure.*

Theorem 6939.2 (Closure capstone realization NameCert obligations)**.** *For every accepted* ClosureCapstoneRealiza *carrier $K = (A, G, R, T, E, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosureCapstoneRealization}^{\uparrow}}$ *surface consists exactly*

of closure-stratum exposure, generator-induction replacement, refusal recording, explicit transport, replay, provenance, and local naming.

Proof. Project the carrier of Definition 6939.1. The row list A, G, R, T, E, P, N is the whole packet, so no accepted read can draw on a hidden representative, erased proposition equality, or external choice function.

The closure-axis row A names the three syntactic strata. The generator row G supplies the finite replacement route at each stratum, while the refusal row R records which unwitnessed closure move is not available.

Transport and replay are explicit. The row T keeps each implication or classifier-respecting route visible, and the row E records the Cont replay that recomputes the replacement route from the displayed rows.

Finally P and N source and name the same packet. Since the excluded axioms and oracle moves are not carrier coordinates, the local NameCert obligations are exactly the displayed rows. □

Theorem 6939.3 (Closure capstone vision route)**.** *The closure-capstone route of chapter 9387 is realized by an accepted* ClosureCapstoneRealization$^{\uparrow}$ *packet only through*

$$A, G, R, T, E, P, N.$$

The route exports finite closure, generator-induction replacement, refusal, transport, replay, provenance, and NameCert rows, but it exports no unwitnessed existential input to later arguments.

Proof. Start from the capstone vision route. Its three strata are recorded in A, and each requested closure move is then routed either to generator induction in G or to a refusal entry in R.

The explicit transport row T replaces proposition-level equality by displayed directions and classifier-respecting transport. The replay row E then records the same route as a finite continuation trace rather than as a meta-level oracle.

The provenance and naming rows P, N bind the realized route to the local carrier. Therefore the concrete landing of the vision is precisely the row list A, G, R, T, E, P, N. □

Falsifiable prediction: An accepted ClosureCapstoneRealization$^{\uparrow}$ packet refutes any closure-capstone read that consumes an unwitnessed choice function, quotient representative selector, or proposition-equality token before displaying the generator-induction, refusal, transport, replay, provenance, and local NameCert rows.

Independence witness: unwitnessedexistenceclosure, propexttransportboundary, effective-replacement-ledger, quotient-soundness-boundary, term-stratum-choice-closure: ClosureCapstoneRealization$^{\uparrow}$ is not bijective to any listed sibling because it combines all three closure strata with generator induction, refusal, explicit transport, replay, provenance, and local naming in one packet.

The future Lean `TasteGate.lean` for ClosureCapstoneRealization$^{\uparrow}$ must provide explicit Nontrivial ClosureCapsto and FieldFaithful ClosureCapstoneRealization$^{\uparrow}$ instances before downstream consumers may cite the packet as a field-faithful capstone carrier.

Closure status for NameCert$_{\text{ClosureCapstoneRealization}^{\uparrow}}$ **·**

Constructive story: A ClosureCapstoneRealization$^{\uparrow}$ packet is generated from closure-stratum rows, generator-induction replacement rows, refusal rows, explicit hsame and psame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosureCapstoneRealization}^{\uparrow}}$, seed)

Scope closed: Definition 6939.1, Definition 6939.2, and Definition 6939.3 seed the finite concrete route from the closure-capstone vision to displayed replacement, refusal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{ClosureCapstoneRealization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, axiom-purity proofs, or every downstream replacement theorem named by the capstone vision.

Upgrade path: Obligation closure requires checked carrier admission, generator-induction coverage, refusal exactness, transport preservation, replay determinacy, provenance, and local naming for BEDC.Derived.ClosureCapstoneRealizationUp.

Chapter 6940

A Concrete Naming Certificate for ClosedBoundedIntervalFiniteSubcover$^{\uparrow}$

Following Bishop and Bridges on constructive compact intervals, the ClosedBoundedIntervalFiniteSubcover$^{\uparrow}$ packet records the finite route by which a displayed closed bounded interval cover request is reduced to inspected located subinterval rows and a finite covering list. It sits after the interval-net surface of chapter 1250, the finite compact-open exhaustion route of chapter 6756, and the located endpoint discipline of chapter 394. The packet names a constructive finite-subcover handoff for closed bounded intervals; it is not a general open-cover compactness principle, not a selected cover from a host powerset, and not an ambient completeness theorem for Real$^{\uparrow}$.

Definition 6940.1 (Closed bounded interval finite-subcover carrier)**.** *A* ClosedBoundedIntervalFiniteSubcover$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, N, E, L, S, W, H, C, P, M)$$

with the following displayed rows:

1. *I is the closed bounded interval source row, including its endpoint reads and boundedness witness.*
2. *N is the* ClosedBoundedIntervalNet$^{\uparrow}$ *row supplying the finite dyadic interval-net inspection surface.*
3. *E is the* CompactOpenExhaustion$^{\uparrow}$ *row recording the finite exhaustion stage used by the cover request.*
4. *L is the* LocatedInterval$^{\uparrow}$ *row that keeps each inspected subinterval attached to displayed endpoint comparisons.*
5. *S is the finite subcover list, whose entries are selected only from rows already displayed by N, E, L.*
6. *W is the cover-witness row matching every retained net cell to an entry of S.*
7. *H records componentwise* hsame *transport for the interval, net, exhaustion, located subinterval, subcover, and witness rows.*
8. *C records displayed* Cont *replay through the same finite route.*
9. *P is the* Pkg *provenance over* ClosedBoundedIntervalFiniteSubcover$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, CompactOpenExhaustion$^{\uparrow}$, LocatedInterval$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. M is the local $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, N, E, L, S, W, H, C, P, M$. It has no coordinate for arbitrary open covers, host compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6940.2 (Closed bounded interval finite-subcover NameCert obligations)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}$ *carrier $K = (I, N, E, L, S, W, H, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}}$ *surface consists exactly of interval-source admission, finite interval-net exposure, compact-open exhaustion-stage exposure, located subinterval comparison, finite subcover listing, cover-witness matching, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6940.1. The rows $I, N, E, L, S, W, H, C, P, M$ are the full packet, so no accepted read can appeal to a detached open-cover object or a host compactness theorem.

The constructive cover side is finite. The interval-net row N supplies the inspected dyadic cells, the exhaustion row E fixes the displayed cover stage, and the located row L keeps the endpoint comparisons attached to those cells.

The subcover list S is then checked by the witness row W. Each entry is matched against a displayed cell before the structural rows H, C, P, M transport, replay, source, and name the same route. □

Theorem 6940.3 (Closed bounded interval finite-subcover route)**.** *Every accepted* $\mathsf{ClosedBoundedIntervalFiniteSubc}$ *packet exports a finite-subcover read only through*

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow}, \quad \mathsf{CompactOpenExhaustion}^{\uparrow}, \quad \mathsf{LocatedInterval}^{\uparrow}.$$

In carrier notation, the consumer-visible route is

$$I, N, E, L, S, W, H, C, P, M.$$

The route supplies a finite list and displayed coverage witnesses for the inspected closed bounded interval request; it exports no general open-cover compactness principle, cover selection from a host powerset, quotient $\mathsf{Real}^{\uparrow}$ *equality, countable choice schedule, or ambient completeness theorem.*

Proof. Start with the finite interval-net surface of chapter 1250. It gives the inspected cells before any cover witness is read.

Next use the compact-open exhaustion route of chapter 6756 and the located interval boundary of chapter 394. These rows attach each cover request to a displayed exhaustion stage and endpoint comparison.

Finally apply the local obligations above. The rows S, W list and verify the finite subcover, while H, C, P, M only preserve and name that displayed route. Since no other carrier coordinate exists, the excluded compactness, choice, quotient, and completeness objects cannot be exported. □

Falsifiable prediction: An accepted $\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}$ read is refuted if it consumes an arbitrary open-cover compactness theorem before displaying the interval-net, compact-open exhaustion, located interval, finite subcover, witness, transport, replay, provenance, and local NameCert rows.

Independence witness: closedboundedintervalnet, compact-open-exhaustion, locatedinterval: $\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval source, interval-net cells, exhaustion-stage data, located endpoint comparisons, a finite subcover list, coverage witnesses, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}$ packet is generated from closed bounded interval rows, interval-net cells, compact-open exhaustion stages, located interval comparisons, finite subcover lists, witness rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6940.1, Definition 6940.2, and Definition 6940.3 seed the finite-subcover route for inspected closed bounded interval cover requests.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFiniteSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, host cover selection, quotient Real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-net coverage, exhaustion-stage compatibility, located interval matching, finite subcover witness coverage, transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalFiniteSubcoverUp.

Chapter 6941

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ packet records the finite cofinal-window selector used when a diagonal-limit or real-completion consumer asks for a tail window of a regular Cauchy family. It realizes the regular-Cauchy route of section 9238.1 through the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the diagonal-limit representative packet of chapter 1205. The selector is a displayed finite tail-window row, not a hidden choice function, quotient equality of real names, or ambient completeness principle.

Definition 6941.1 (Regular Cauchy tail-window selector carrier)**.** *A* $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, R, A, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the requested positive tail budget.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the family of regular Cauchy names.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected window.*
4. *A is the tail-admission row, recording that the inspected window lies beyond the displayed Cauchy modulus request.*
5. *Q is the cofinal selector row, naming the finite tail window consumed by the diagonal or completion-facing read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R, A, Q have been displayed.*
7. *H records componentwise* hsame *transport for the tolerance, window, readback, admission, selector, and seal rows.*
8. *C records displayed* Cont *replay of the same selector route.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, A, Q, E, H, C, P, N$. It has no coordinate for countable choice, a selected infinite subsequence, quotient real equality, hidden limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6941.2 (Regular Cauchy tail-window selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ *carrier* $T = (D, S, R, A, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyT}}$ *surface consists exactly of dyadic tolerance admission, finite stream-window exposure, regular rational readback, tail-admission evidence, cofinal selector exposure, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6941.1. The rows $D, S, R, A, Q, E, H, C, P, N$ are the full selector packet, so there is no separate coordinate for an infinite choice schedule or hidden limit.

The finite selector is read in order. The dyadic tolerance row D fixes the requested budget, S supplies the inspected stream window, R reads the regular rational data, and A records that the window is admitted as a tail row.

The cofinal selector row Q then names the window handed to diagonal and real-completion consumers. The rows E, H, C, P, N seal, transport, replay, source, and name the same displayed route. □

Theorem 6941.3 (Regular Cauchy tail-window selector route)**.** *Every accepted* $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ *packet supplies a finite tail-window selector for the diagonal-limit and real-completion route only through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{DiagonalLimitRepresentative}^{\uparrow}.$$

In carrier notation, the consumer-visible route is

$$D, S, R, A, Q, E, H, C, P, N.$$

The selector exports a cofinal finite window and its displayed tolerance/readback evidence; it exports no countable-choice schedule, hidden subsequence selector, quotient $\mathsf{Real}^{\uparrow}$ *equality, selected limit, or ambient completeness theorem.*

Proof. Start from the regular-Cauchy vision route of section 9238.1. A diagonal-limit consumer asks for one displayed tail window at a requested dyadic tolerance, not for a completed infinite schedule.

Read the sibling rows in order. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the budget, the $\mathsf{StreamName}^{\uparrow}$ row supplies the inspected finite window, and the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback consumed before the $\mathsf{Real}^{\uparrow}$ seal.

Finally connect the selector to chapter 1205. The cofinal row Q is the finite window handed to the diagonal representative, while H, C, P, N transport, replay, source, and name the same packet. The refused choice, quotient, selected-limit, and completeness objects have no carrier coordinate. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ read is refuted if a diagonal-limit or Real-completion consumer receives a tail window before the dyadic tolerance, StreamName window, RegSeqRat readback, tail-admission, cofinal selector, Real seal, transport, replay, provenance, and local NameCert rows are displayed.

Independence witness: dyadicratcore, streamname, regseqrat, real, diagonallimitrepresentative: $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a

tolerance request, stream window, regular rational readback, tail admission, cofinal selector, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailWindowSelector}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, tail-admission rows, cofinal selector rows, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailWindowSelector}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6941.1, Definition 6941.2, and Definition 6941.3 seed the finite cofinal-window selector route for diagonal-limit and real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailWindowSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient Real equality, hidden subsequence selection, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance coverage, stream-window readback, tail-admission evidence, cofinal selector exactness, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailWindowSelectorUp.

Chapter 6942

A Concrete Naming Certificate for $\mathsf{BusemannConvexMetric}^{\uparrow}$

Following Busemann and Bridson–Haefliger on convexity in geodesic metric spaces, the $\mathsf{BusemannConvexMetric}^{\uparrow}$ packet records the finite midpoint-distance convexity surface that sits between displayed geodesic metric data and stronger CAT(0) or Hadamard consumers. It sits over the metric carrier of chapter 57, the segment-window route of chapter 2213, the convex-set rows of chapter 208, the CAT(0) comparison surface of chapter 6336, the Hadamard-space carrier of chapter 6734, and the terminal real-name seal of chapter 38. The packet names midpoint convexity for displayed geodesic windows; it is not a full CAT(0) comparison theorem, not a selected geodesic family, not quotient path equality, and not ambient completeness.

Definition 6942.1 (Busemann convex metric carrier)**.** *A* $\mathsf{BusemannConvexMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, X, Y, S, T, D, R, V, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only point classifiers, distance readbacks, and the metric-law ledger used by the convexity request.*
2. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *row naming the finite geodesic segment windows inspected by the packet.*
3. *X and Y are endpoint-pair rows for two displayed geodesic windows, with every endpoint admitted through M.*
4. *S and T are the finite segment-parameter rows for the two windows, including the midpoint or equal-parameter addresses used by the convexity comparison.*
5. *D is the distance-comparison row, recording the finite Busemann convexity inequality that the distance between equal-parameter points on the two displayed windows is bounded by the corresponding convex combination of endpoint distances.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the distance values read by D, reached only after the metric, geodesic, endpoint, and parameter rows have been displayed.*
7. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *handoff row for consumers that read the same midpoint or equal-parameter windows as convex-domain data.*
8. *H records componentwise* hsame *transport for the metric, geodesic, endpoint, parameter, distance-comparison, Real-seal, convex-handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite Busemann-convexity request through*

$$M, G, X, Y, S, T, D, R, V \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{BusemannConvexMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{HadamardSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BusemannConvexMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a global CAT(0) triangle comparison, a Hadamard-space theorem, selected geodesics, quotient path equality, nearest-point projection, or ambient completeness.

Theorem 6942.2 (Busemann convex metric NameCert obligations)**.** *For every accepted* $\mathsf{BusemannConvexMetric}^{\uparrow}$ *carrier*

$$B = (M, G, X, Y, S, T, D, R, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BusemannConvexMetric}^{\uparrow}}$ *surface consists exactly of metric-source admission, geodesic segment-window admission, endpoint-pair admission, finite parameter admission, Busemann distance-convexity ledger exactness, terminal* $\mathsf{Real}^{\uparrow}$ *distance sealing, convex-window handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 6942.1.*

Proof. Carrier admission is projection from Definition 6942.1. The accepted packet displays the metric source, geodesic row, two endpoint-pair rows, two finite parameter rows, the distance-convexity ledger, Real seal, convex handoff, transport, replay, provenance, and local name.

The convexity obligation is localized in D. That row reads only the equal-parameter or midpoint observations supplied by S and T, compares their distance through M, and seals the resulting distance data through R.

Rows V, H, C, P, N do not enlarge the geometric object. They route the same displayed convexity window to convex-domain consumers, preserve the rows componentwise, replay the same finite request, record provenance, and name the local certificate. Thus no obligation row can be read as CAT(0) comparison, Hadamard completeness, selected geodesics, quotient path equality, projection selection, or ambient completeness. □

Theorem 6942.3 (Busemann convex midpoint non-escape)**.** *Every midpoint-facing or equal-parameter consumer of an accepted* $\mathsf{BusemannConvexMetric}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{GeodesicMetric}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

It may consume the displayed Busemann convexity ledger for the named segment windows, but it cannot infer a CAT(0) comparison triangle, a Hadamard-space theorem, selected geodesics, quotient path equality, nearest-point projection, or ambient completeness.

Proof. Use Definition 6942.2. The consumer receives midpoint or equal-parameter information only after the metric source, geodesic windows, endpoint pairs, finite parameter rows, distance-convexity ledger, and Real seal have been displayed.

The convex-domain handoff then passes through V. This lets a later $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, or $\mathsf{HadamardSpace}^{\uparrow}$ consumer read the finite Busemann convexity row as local certificate data, without converting it into a stronger global comparison theorem.

Inspecting the carrier gives the exclusion. There is no row for a full CAT(0) triangle ledger, global Hadamard completeness, selected geodesic family, path quotient, projection selector, or ambient completeness proof, so any route requiring one of those objects lies outside the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BusemannConvexMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BusemannConvexMetric}^{\uparrow}$ packet is generated from a MetricSpace source, GeodesicMetric segment-window rows, endpoint pairs, finite parameter rows, a Busemann distance-convexity ledger, a RealUp seal, a ConvexSet handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BusemannConvexMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6942.1, Definition 6942.2, and Definition 6942.3 seed the finite Busemann convexity route from displayed metric and geodesic data through convex, CAT(0), Hadamard, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BusemannConvexMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim CAT(0) triangle comparison, Hadamard completeness, selected geodesics, quotient path equality, nearest-point projections, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-pair exposure, parameter-row exactness, Busemann distance-convexity stability, Real sealing, convex handoff, transport, replay, provenance, and local naming for BEDC.Derived.BusemannConvexMetricUp.

Chapter 6943

A Concrete Naming Certificate for $\mathsf{KripkeSemantics}^{\uparrow}$

Following Kripke and standard intuitionistic semantics, the $\mathsf{KripkeSemantics}^{\uparrow}$ packet records a finite forcing model as a NameCert surface rather than as an ambient metatheorem about all intuitionistic consequence. It reads the preorder-world discipline of chapter 52, the formula and syntax handoff of chapter 197, and the model-facing valuation surface of chapter 198. The packet keeps monotone forcing visible as a finite BHist route before any consumer may read a semantic validity claim.

Definition 6943.1 (Kripke semantics carrier)**.** *A* $\mathsf{KripkeSemantics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, F, V, R, M, H, C, P, N)$$

with the following displayed rows:

1. *W is the preorder-world row, read through* $\mathsf{Preorder}^{\uparrow}$*, listing the finite worlds and the accessibility relation used by forcing.*
2. *F is the formula row, read through* $\mathsf{FirstOrder}^{\uparrow}$*, exposing the finite intuitionistic formulas inspected by the packet.*
3. *V is the valuation row, read through* $\mathsf{ModelTheory}^{\uparrow}$*, assigning atomic forcing data at the displayed worlds.*
4. *R is the monotone forcing row, recording that atomic valuation and compound forcing are preserved along the preorder accessibility route.*
5. *M is the* $\mathsf{ModelTheory}^{\uparrow}$ *and* $\mathsf{FirstOrder}^{\uparrow}$ *handoff row, keeping syntactic formula reads and semantic valuation reads on the same finite packet.*
6. *H records componentwise* hsame *transport for W, F, V, R, M and the local naming row.*
7. *C records displayed* Cont *replay from a forcing request through preorder access, formula inspection, valuation readback, monotone forcing, provenance, and naming.*
8. *P is the* Pkg *provenance over* $\mathsf{KripkeSemantics}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KripkeSemantics}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, F, V, R, M, H, C, P, N$. It has no coordinate for classical truth tables, Boolean completeness, global proof search, quotient syntax equality, or a standard bridge theorem.

Theorem 6943.2 (Kripke semantics NameCert obligations)**.** *For every accepted* $\mathsf{KripkeSemantics}^{\uparrow}$ *carrier* $K = (W, F, V, R, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KripkeSemantics}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the preorder-world row W through the* $\mathsf{Preorder}^{\uparrow}$ *sibling surface;*
2. *expose the formula row F through the* $\mathsf{FirstOrder}^{\uparrow}$ *sibling surface;*
3. *expose the valuation row V and model handoff row M through the* $\mathsf{ModelTheory}^{\uparrow}$ *sibling surface;*
4. *preserve forcing by the monotone row R under componentwise* hsame *transport and displayed* Cont *replay;*
5. *retain* Pkg *provenance and local naming in P, N without exporting classical completeness, Boolean collapse, quotient syntax equality, or bridge checking.*

Proof. Project the carrier of Definition 6943.1. The rows W, F, V, R, M are the complete forcing-facing data, while H, C, P, N are the structural rows that transport, replay, source, and name that data.

The preorder, first-order, and model-theory rows are read from their sibling NameCert packets. Thus a forcing consumer cannot inspect worlds without the preorder row, formulas without the syntax row, or valuations without the model row.

The monotone forcing row is then local. It preserves the valuation and compound forcing reads along the displayed preorder accessibility route, and the structural rows keep that route attached to the same finite packet.

Since no carrier coordinate contains classical completeness, Boolean collapse, quotient syntax equality, or bridge data, those endpoints are not exported by the local NameCert surface. □

Theorem 6943.3 (Kripke semantics monotone forcing route)**.** *Every forcing consumer of an accepted* $\mathsf{KripkeSemantics}^{\uparrow}$ *carrier factors through the preorder-world row W, formula row F, valuation row V, monotone forcing row R, model handoff row M, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The route is therefore*

$$\mathsf{Preorder}^{\uparrow},\ \mathsf{FirstOrder}^{\uparrow},\ \mathsf{ModelTheory}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

before any intuitionistic forcing read is accepted.

Proof. Start with the NameCert obligations of Definition 6943.2. They require the world, formula, valuation, monotonicity, handoff, transport, replay, provenance, and naming rows to be visible before a forcing consumer reads the packet.

The preorder row supplies the accessibility relation used by monotonicity. The formula row supplies the finite syntax being forced, and the model row supplies the valuation surface from which atomic forcing is read.

Now use the monotone forcing row R. Any forcing read that moves along accessibility must preserve the relevant valuation and compound forcing data through R, then pass through H and C so that the read is transported and replayed on the same BHist packet.

Finally P and N package the route as a local NameCert. A consumer omitting one of these rows has not produced a $\mathsf{KripkeSemantics}^{\uparrow}$ forcing read; it has only a detached preorder, syntax, or valuation fragment. □

Lean4 (statement) `BEDC.Derived.KripkeSemanticsUp`

Closure status for $\mathsf{NameCert}_{\mathsf{KripkeSemantics}^{\uparrow}}$.

Constructive story: A $\mathsf{KripkeSemantics}^{\uparrow}$ packet is generated from preorder-world rows, formula rows, valuation rows, monotone forcing rows, model-theory handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KripkeSemantics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6943.1, Definition 6943.2, and Definition 6943.3 seed the finite Kripke forcing route over preorder worlds, formula reads, valuation reads, monotonicity, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KripkeSemantics}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.KripkeSemanticsUp`

Bridge status: none

Not claimed: This chapter does not close classical completeness, Boolean semantics, global proof search, quotient syntax equality, Lean-checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, preorder-world exposure, formula exposure, valuation exposure, monotone forcing stability, transport, replay, provenance, and local naming for BEDC.Derived.KripkeSemanticsUp.

Chapter 6944

A Concrete Naming Certificate for $\mathsf{ArcLengthParameter}^{\uparrow}$

Following Bishop and Bridges constructive analysis of rectifiable curves, the $\mathsf{ArcLengthParameter}^{\uparrow}$ packet records the finite route by which a displayed curve receives arc-length parameter data from partition-length ledgers and regular rational readbacks. It sits over the metric carrier of chapter 4230, the continuous-map surface of chapter 130, the dyadic tolerance core of chapter 291, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the Darboux partition ledger of chapter 3429, and the located integral route of chapter 5947. The packet names only finite rectifiable-curve parameter data; it is not a quotient of curves, a host length functional, a selected reparameterization, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6944.1 (Arc-length parameter carrier)**.** *An* $\mathsf{ArcLengthParameter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, M, D, V, R, L, J, S, H, C, P, N)$$

with the following displayed rows:

1. *G is the curve row, naming the displayed source interval, target points, and the curve graph through the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to measure adjacent curve samples.*
3. *D is the finite Darboux or tagged partition row, recording the ordered sample ledger from which polygonal lengths are read.*
4. *V is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *variation-bound row, carrying dyadic upper and lower budgets for partition-length comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the resulting length approximants.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row naming the accumulated length read.*
7. *J is the located-integral handoff row, available when the same partition ledger is consumed through* $\mathsf{LocatedRiemannIntegral}^{\uparrow}$*.*
8. *S is the local parameter-scale row, pairing initial segment requests with their displayed length readbacks.*
9. *H records componentwise* hsame *transport for curve, metric, partition, dyadic variation, regular readback, Real seal, integral handoff, parameter-scale, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a parameter request through G, M, D, V, R, L, J, S.*
11. *P is the* Pkg *provenance over* $\mathsf{ArcLengthParameter}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ArcLengthParameter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $G, M, D, V, R, L, J, S, H, C, P, N$. It has no coordinate for host limits of partition sums, quotient curves, chosen inverse parameters, global geodesic structure, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6944.2 (Arc-length parameter NameCert obligations)**.** *For every accepted* $\mathsf{ArcLengthParameter}^{\uparrow}$ *carrier $A = (G, M, D, V, R, L, J, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ArcLengthParameter}^{\uparrow}}$ *surface consists exactly of carrier admission, curve-row admission, metric distance readback, finite partition-length ledger exposure, dyadic variation-bound accounting, regular rational readback stability, Real seal stability, located integral handoff, parameter-scale replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6944.1. The rows G, M, D, V, R, L, J, S are the whole numerical and geometric surface before any structural row is read.

The curve and length data are exhausted by G, M, D, V. The curve row gives the displayed continuous map, the metric row gives adjacent-sample distances, the partition row records the finite ledger of samples, and the dyadic variation row stores the comparison budgets for polygonal length reads.

The readback and parameter data are exhausted by R, L, J, S. Regular rational readback turns the partition ledger into a finite Cauchy-style name, the Real seal records the terminal length read, the located-integral handoff supplies the sibling integral route, and the parameter-scale row pairs initial segment requests with those displayed length reads.

Finally H, C, P, N transport, replay, source, and name exactly the same displayed rows. Since the carrier contains no coordinate for host partition limits, quotient curves, selected inverse parameters, global geodesic structure, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows are precisely the NameCert obligation surface. □

Theorem 6944.3 (Arc-length parameter Real metric handoff)**.** *Every metric or rectifiable-curve consumer of an accepted* $\mathsf{ArcLengthParameter}^{\uparrow}$ *carrier must read*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the finite Darboux partition ledger and the $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ *handoff before it consumes the parameter-scale row S. The handoff exports no host length functional, quotient curve equality, selected reparameterization, countable-choice schedule, global geodesic object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $A = (G, M, D, V, R, L, J, S, H, C, P, N)$. The curve row G is read through the displayed $\mathsf{ContinuousMap}^{\uparrow}$ surface, and the distance row M is read through the $\mathsf{Met}^{\uparrow}$ carrier before any length comparison is available.

Next read the finite length ledger. The partition row D supplies the Darboux-style ordered sample data, while V supplies dyadic variation bounds through $\mathsf{DyadicRatCore}^{\uparrow}$. Thus every polygonal length comparison remains a finite ledger comparison.

The Real-facing handoff then passes through R, L, J. The $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational readback, the $\mathsf{Real}^{\uparrow}$ seal records the terminal length name, and the $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ row records the sibling integral route using the same finite partition data.

Only after those rows have been displayed may the consumer read S. The structural rows H, C, P, N preserve the same route, so the packet cannot introduce a host length functional, quotient curve equality, selected reparameterization, countable-choice schedule, global geodesic object, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{ArcLengthParameter}^{\uparrow}$ consumer that reaches the parameter-scale row displays a ContinuousMap curve row, MetricSpace distance row, finite partition-length ledger, DyadicRatCore variation bounds, RegSeqRat readback, RealUp seal, LocatedRiemannIntegral handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: continuousmap, metric-space, dyadicratcore, regseqrat, real, located-riemann-integral, closed-bounded-interval-darboux-partition: $\mathsf{ArcLengthParameter}^{\uparrow}$ is not bijective to any listed sibling because it jointly records curve samples, metric distances, partition-length ledgers, dyadic variation bounds, regular rational readback, Real sealing, integral handoff, and parameter-scale replay.

Closure status for $\mathsf{NameCert}_{\mathsf{ArcLengthParameter}^{\uparrow}}$.

Constructive story: An $\mathsf{ArcLengthParameter}^{\uparrow}$ packet is generated from a ContinuousMap curve row, MetricSpace distance row, finite partition-length ledger, DyadicRatCore variation-bound row, RegSeqRat readback, RealUp seal, LocatedRiemannIntegral handoff, local parameter-scale row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArcLengthParameter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6944.1, Definition 6944.2, and Definition 6944.3 seed the finite arc-length parameter route for rectifiable curve data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArcLengthParameter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host length functionals, quotient curve equality, selected inverse parameters, global geodesic structure, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, curve-row admission, metric distance readback, partition-length ledger exposure, dyadic variation accounting, regular rational readback, Real seal stability, located integral handoff, parameter-scale replay, transport, replay, provenance, and local naming for BEDC.Derived.ArcLengthParameterUp.

Chapter 6945

A Concrete Naming Certificate for $\mathsf{TestFunction}^{\uparrow}$

Following Schwartz and Treves, the $\mathsf{TestFunction}^{\uparrow}$ packet records compactly supported smooth test probes as finite BEDC NameCert data. It sits next to the rapidly decreasing smooth carrier of chapter 3399, the Schwartz-space packet of chapter 5877, the weak-derivative route of chapter 2210, and the distribution-facing scalar boundary of chapter 2310. The packet exposes the finite support window, finite derivative ledger, local scalar readback, and replay route required by distribution, Fourier, Sobolev, and viscosity-solution consumers without importing a global LF-space topology, quotient germ equality, countable choice, or ambient smooth-completeness theorem.

Definition 6945.1 (Test-function carrier)**.** *A* $\mathsf{TestFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, D, L, Q, S, V, H, C, P, N)$$

with the following displayed rows:

1. *W is the compact support window row. It displays the bounded local chart window on which the probe may be nonzero and records that all public reads outside the window are the zero read.*
2. *D is the finite derivative-order ledger. It lists the derivative orders inspected by the packet and the finite smoothness obligations replayed for those orders.*
3. *L is the localization domain row. It records the coordinate chart, bump-support inclusion, and domain restriction through which the test probe is consumed.*
4. *Q is the rational observation row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, for coefficient windows, derivative tolerances, and support-bound comparisons.*
5. *S is the scalar seal row, read through* $\mathsf{Real}^{\uparrow}$*, that turns the rational observations in Q into Real-facing scalar names for probe values and derivative bounds.*
6. *V is the weak-derivative compatibility row. It records the sign, derivative-order, and integration-by-parts bookkeeping consumed by* $\mathsf{WeakDerivative}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for $W, D, L, Q, S, V, C, P, N$.*
8. *C records displayed* Cont *replay from a test-probe request through support localization, derivative inspection, rational readback, Real scalar sealing, weak-derivative compatibility, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{TestFunction}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{FourierTransform}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TestFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, D, L, Q, S, V, H, C, P, N$. It has no coordinate for a global smooth-function topology, quotient germ equality, nonconstructive partitions of unity, countable support selection, or an ambient distribution theorem.

Theorem 6945.2 (Test-function NameCert obligations)**.** *For every accepted* $\mathsf{TestFunction}^{\uparrow}$ *carrier $T = (W, D, L, Q, S, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TestFunction}^{\uparrow}}$ *surface consists exactly of compact support-window exposure, finite derivative-ledger exposure, localization-domain exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational readback,* $\mathsf{Real}^{\uparrow}$ *scalar sealing,* $\mathsf{WeakDerivative}^{\uparrow}$ *compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6945.1. The rows W, D, L are the complete geometric and smoothness data visible to a test-probe consumer: support, derivative orders, and localization.

Read the scalar data next. The row Q supplies the finite rational observations for coefficients, derivative tolerances, and support bounds, while S seals those observations as $\mathsf{Real}^{\uparrow}$ scalar names.

The compatibility row V is the only weak-derivative-facing row. It records the sign and derivative-order bookkeeping used by $\mathsf{WeakDerivative}^{\uparrow}$ without adding a distribution value or Sobolev norm as a carrier coordinate.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no global smooth-function topology, quotient germ equality, nonconstructive partition row, countable support-selection row, or ambient distribution theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 6945.3 (Test-function tempered-distribution handoff)**.** *Every* $\mathsf{TemperedDistribution}^{\uparrow}$ *or* $\mathsf{FourierTransform}^{\uparrow}$ *consumer that reads a test probe from an accepted* $\mathsf{TestFunction}^{\uparrow}$ *packet factors through*

$$W, D, L, Q, S, V, H, C, P, N.$$

Consequently the handoff exposes a compact support window, finite derivative ledger, localization domain, rational and Real scalar readback, weak-derivative compatibility, transport, replay, provenance, and local naming before any distribution pairing, Fourier pairing, Sobolev test, or viscosity-solution probe is public.

Proof. Begin with the support and localization rows. The compact window W and localization domain L determine where the probe may be read, and the derivative ledger D determines which smoothness obligations are available to downstream chapters.

Use the scalar readback route Q, S. The regular rational and dyadic observations in Q provide the finite numerical budgets, and the $\mathsf{Real}^{\uparrow}$ seal in S makes those budgets available to distribution and Fourier pairings.

Now pass through the weak-derivative compatibility row V. This is the row consumed by $\mathsf{WeakDerivative}^{\uparrow}$ when an integration-by-parts sign or derivative order is needed, and it is the row that keeps Sobolev and viscosity-solution test probes tied to the same finite carrier.

Finally replay the route through H, C, P, N. The sibling rows from $\mathsf{SchwartzFunction}^{\uparrow}$ and $\mathsf{SchwartzSpace}^{\uparrow}$ provide smooth decay and space-level comparison surfaces when a consumer needs them, while the $\mathsf{TestFunction}^{\uparrow}$ handoff itself remains compact-support data. No distribution pairing can bypass these rows to obtain quotient germ equality, a selected partition of unity, a global LF-space topology, or an ambient smooth-completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{TestFunction}^{\uparrow}$ read must expose compact support windows, finite derivative ledgers, localization domains, RegSeqRat and DyadicRatCore rational observations, RealUp scalar seals, WeakDerivative compatibility, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any distribution, Fourier, Sobolev, or viscosity-solution consumer may use the test probe.

Independence witness: schwartzfunction, schwartzspace, weakderivative, tempereddistribution, fouriertransform, real: $\mathsf{TestFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact support windows, finite derivative ledgers, localization domains, rational and Real scalar readback, weak-derivative compatibility, transport, replay, provenance, and local naming while refusing global smooth-function topology, quotient germ equality, selected partitions of unity, and ambient distribution theorems.

Closure status for $\mathsf{NameCert}_{\mathsf{TestFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TestFunction}^{\uparrow}$ packet is generated from compact support windows, finite derivative-order ledgers, localization domains, RegSeqRat and DyadicRatCore rational readbacks, Real scalar seals, WeakDerivative compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TestFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6945.1, Definition 6945.2, and Definition 6945.3 seed the compact-support smooth test-probe certificate used by distribution, Fourier, Sobolev, and viscosity-solution consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TestFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global LF-space topology, quotient germ equality, nonconstructive partitions of unity, countable support selection, Lean verification, or bridge checking for $\mathsf{TestFunction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, compact support-window exactness, finite derivative-ledger stability, localization replay, scalar readback, weak-derivative compatibility, transport, replay, provenance, and local naming for BEDC.Derived.TestFunctionUp.

Chapter 6946

A Concrete Naming Certificate for $\mathsf{CompactIntervalFanBridge}^{\uparrow}$

Following Bishop–Bridges compact-interval analysis and Brouwer fan-theorem grounding, the CompactIntervalFanBr packet records the finite route by which compact interval data may be consumed through fan and bar finite-modulus rows before a Real-and-Completion reader reaches a terminal seal. It sits between the Bishop compact-interval source of chapter 3986, the finite fan route of chapter 5863, the stream-window interface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the real-name boundary of chapter 38. The packet names a finite compact-interval fan bridge; it is not open-cover compactness, sequential compactness, a selected cover, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6946.1 (Compact interval fan bridge carrier)**.** *A* $\mathsf{CompactIntervalFanBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *source row, recording the displayed endpoints, interval order data, and compact-interval admission surface.*
2. *F is the fan and bar finite-modulus row, recording a finite prefix ledger, bar boundary, branch-depth ledger, and modulus output for the displayed interval request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic precision request and every dyadic comparison used by the finite-modulus row.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the stream windows inspected before readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from S and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after K, F, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the compact source, finite-modulus row, dyadic ledger, stream windows, regular readback, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a compact-interval request through $K, F, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalFanBridge}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactIntervalFanBridge}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $K, F, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, sequential compactness, choice-selected covers, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6946.2 (Compact interval fan bridge NameCert obligations)**.** *For every accepted* CompactIntervalFanBrid *carrier $G = (K, F, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFanBridge}^{\uparrow}}$ *surface consists exactly of compact-interval source exposure, fan and bar finite-modulus exposure, dyadic tolerance accounting, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6946.1. The analytic route is exhausted by K, F, D, S, R, E, and the structural route is exhausted by H, C, P, N.

The compact source row K displays the interval before any fan data is read. The finite-modulus row F records only a finite prefix and bar ledger, while the dyadic ledger D records the tolerance comparisons used by that finite row.

The windows S, readback R, and seal E are downstream of the displayed finite data. Transport, replay, provenance, and local naming preserve exactly that inventory, so the NameCert surface contains the listed obligations and no hidden open-cover, sequential, choice, quotient, trichotomy, or completeness endpoint. □

Theorem 6946.3 (Compact interval fan bridge finite route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{CompactIntervalFanBridge}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{BishopCompactInterval}^{\uparrow} \longmapsto \mathsf{FiniteFanTheorem}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $K, F, D, S, R, E, H, C, P, N$. The exported bridge is the finite fan and bar modulus route from compact interval data to regular-rational readback, not an open-cover compactness theorem, sequential compactness theorem, choice-selected cover, quotient real equality, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with Definition 6946.2. A consumer must expose the compact source, finite-modulus ledger, dyadic tolerance ledger, stream windows, regular readback, terminal seal, transport, replay, provenance, and local naming before reading the bridge.

The route reaches $\mathsf{StreamName}^{\uparrow}$ only after the dyadic tolerance ledger and finite bar boundary have been displayed. It reaches $\mathsf{RegSeqRat}^{\uparrow}$ only through those finite windows, and it reaches $\mathsf{Real}^{\uparrow}$ only through the terminal seal row.

Since the carrier has no row for open covers, sequential compactness, selected covers, quotient real equality, trichotomy, or ambient completeness, the finite route is exhausted by $K, F, D, S, R, E, H, C, P, N$. □

Falsifiable prediction: An accepted $\mathsf{CompactIntervalFanBridge}^{\uparrow}$ consumer that exports a $\mathsf{Real}^{\uparrow}$ seal without displaying $K, F, D, S, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: bishop compact interval, finite fan theorem, streamname, dyadicratcore, regseqrat, real: $\mathsf{CompactIntervalFanBridge}^{\uparrow}$ is not bijective to any listed sibling because it packages compact-interval source data, finite fan and bar modulus data, dyadic tolerance accounting, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows in one bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalFanBridge}^{\uparrow}}$.

Constructive story: A CompactIntervalFanBridge$^{\uparrow}$ packet is generated from BishopCompactIntervalUp source rows, fan and bar finite-modulus rows, DyadicRatCoreUp tolerance ledgers, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactIntervalFanBridge}^{\uparrow}}$, seed)

Scope closed: Definition 6946.1, Definition 6946.2, and Definition 6946.3 seed the finite compact-interval fan bridge for Real-and-Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalFanBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, selected covers, quotient real equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for the displayed compact interval fan bridge route.

Chapter 6947

A Concrete Naming Certificate for $\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}$

The $\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}$ packet realizes the normal-equality boundary of section 9379.3 as finite MetaCIC audit data. It sits beside the bounded normal-equality checker of chapter 752, the decidable conversion frontier of chapter 4097, the bounded decidable fragment of chapter 6917, and the decidable neutral-spine packet of chapter 3909. The packet names the frontier where normalizer fuel, closedness rows, finished normal forms, equality checking, transport, replay, provenance, and local naming remain visible together; it does not claim unrestricted beta-conversion decidability, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 6947.1 (Bounded normal-equality frontier carrier)**.** *A* $\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, F, L, N, E, R, H, C, P, A)$$

with the following displayed rows:

1. *I is the closed MetaCIC input row, listing the two closed terms, their type context, and the bounded equality query.*
2. *F is the finite normalizer-fuel row, recording the shared fuel value and the bounded evaluation trace admitted for each side.*
3. *L is the closedness ledger, recording the closed-term and closed-normal-form checks consumed by the frontier.*
4. *N is the finished-normal-form row, displaying the two bounded normal forms and the witnesses that the displayed fuel has reached finished endpoints.*
5. *E is the equality-checker row, recording the structural comparison of the finished normal forms.*
6. *R is the refusal row, naming unrestricted beta conversion, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, and kernel consistency as unavailable endpoints.*
7. *H records componentwise* hsame *transport for the input, fuel, closedness, finished-normal, equality-checker, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the audit-map query through $I, F, L, N, E, R, H, P, A$.*

9. *P is the* Pkg *provenance over* BoundedNormalEqualityFrontier$^{\uparrow}$, BoundedNormalEqualityChecker$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, MetaCICDecidableFragment$^{\uparrow}$, MetaCICDecidableNeutralSpine$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *A is the local* NameCert$_{\text{BoundedNormalEqualityFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, F, L, N, E, R, H, C, P, A$. It has no coordinate for an unrestricted conversion decider, arbitrary-fuel normalization completeness, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Lean4 (definition) `BEDC.Derived.BoundedNormalEqualityFrontierUp.BoundedNormalEqualityFrontierUp`

Theorem 6947.2 (Bounded normal-equality frontier NameCert obligations)**.** *For every accepted* BoundedNormalEqualityFrontier$^{\uparrow}$ *carrier $Q = (I, F, L, N, E, R, H, C, P, A)$, the local* NameCert$_{\text{BoundedNormalEqualityFron}}$ *surface consists exactly of closed input exposure, finite normalizer-fuel exposure, closedness ledger exposure, finished-normal-form exposure, equality-checker exposure, refusal-boundary retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6947.1. The rows I, F, L, N, E, R are the complete normal-equality audit data, and H, C, P, A are the structural rows that transport, replay, source, and name that data.

The fuel row F bounds the normalizer traces. The closedness row L records that the query and the finished endpoints are closed, while the finished-normal row N exposes the two normal forms before any equality result is read.

The equality-checker row E is therefore consumed only under the retained refusal row R. Since no carrier coordinate contains unrestricted beta conversion, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency, those endpoints cannot be exported by the local NameCert surface. □

Theorem 6947.3 (Bounded normal-equality frontier finished soundness route)**.** *Every MetaCIC decidability consumer of an accepted* BoundedNormalEqualityFrontier$^{\uparrow}$ *carrier must read*

$$I \longmapsto F \longmapsto L \longmapsto N \longmapsto E \longmapsto R, H, C, P, A$$

before treating a positive equality-checker row as a finished bounded normal-equality result. The route is conditional on displayed fuel, closedness, finished endpoints, and structural equality of the displayed normal forms; it is not unrestricted beta-conversion decidability, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Proof. Start with Definition 6947.2. A consumer must expose the input, bounded fuel, closedness ledger, finished normal forms, equality checker, refusal boundary, transport, replay, provenance, and local naming before reading the frontier.

The normal-equality result reaches the checker row E only after the finite fuel row F, closedness row L, and finished-normal row N have been displayed. Thus any positive checker read is a finished bounded-normal-form readback rather than a global conversion decision.

The refusal row R remains in the exported route. A consumer that wants arbitrary fuel, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency must supply a separate certificate outside this packet. □

Falsifiable prediction: An accepted BoundedNormalEqualityFrontier$^{\uparrow}$ consumer that exports a normal-equality result without displaying $I, F, L, N, E, R, H, C, P, A$ refutes the frontier surface.

Independence witness: metacic-decidable-conversion-frontier, metacic-decidable-fragment, metacic-decidable-typing-frontier, metacic-decidable-neutral-spine: $\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores normalizer fuel, closedness, finished-normal-form, equality-checker, refusal, transport, replay, provenance, and local NameCert rows. Later Lean instances should expose nontriviality by producing two distinct accepted frontier packets and field faithfulness by showing that equality of frontier fields is exactly equality of $I, F, L, N, E, R, H, C, P, A$.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}$ packet is generated from closed MetaCIC input rows, finite normalizer-fuel rows, closedness ledgers, finished-normal-form rows, equality-checker rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6947.1, Definition 6947.2, and Definition 6947.3 seed the finite bounded normal-equality frontier for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted beta-conversion decidability, arbitrary-fuel completeness, subject reduction, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier, Nontrivial and FieldFaithful instances, and checked NameCert obligations for the displayed bounded normal-equality frontier route.

Chapter 6948

A Concrete Naming Certificate for $\mathsf{MerayCauchyReal}^{\uparrow}$

Following Méray and Bishop–Bridges on Cauchy presentations of the real line, the $\mathsf{MerayCauchyReal}^{\uparrow}$ packet records the historical construction as finite displayed name data rather than as a quotient of Cauchy sequences. It sits after the Cauchy-real presentation of chapter 1511, the quotient-free boundary of chapter 6937, the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives the Real and Completion phase a traditional source row whose public reads remain finite all the way to the $\mathsf{Real}^{\uparrow}$ seal.

Definition 6948.1 (Méray Cauchy-real carrier)**.** *A* $\mathsf{MerayCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the displayed Cauchy source row. It records the rational Cauchy presentation used by the historical construction, with its modulus and tolerance requests shown as finite data.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window service row. It exposes exactly the inspected prefixes of the source and no infinite stream equality.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It converts the displayed windows into the regular rational sequence data consumed by downstream real-name readers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the Cauchy source, window reads, regular readback, and terminal seal.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after A, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the Cauchy source, windows, readback, tolerance ledger, seal, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay along the finite route*

$$A, W, R, D \longmapsto E, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{MerayCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{MerayCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the nine displayed rows. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 6948.2 (Méray Cauchy-real NameCert obligations)**.** *For every accepted* $\mathsf{MerayCauchyReal}^{\uparrow}$ *carrier* $M = (A, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MerayCauchyReal}^{\uparrow}}$ *surface consists exactly of the finite carrier row, displayed Cauchy-source admission, StreamName window locality, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside those rows.*

Proof. Project the carrier of Definition 6948.1. The tuple lists the Cauchy source, finite window service, regular rational readback, dyadic ledger, terminal seal, transport, replay, provenance, and local name.

The analytic rows are then read in order. The source row A supplies only displayed rational Cauchy data, W exposes the finite windows inspected from it, R records the regular readback of those windows, and D accounts for every tolerance used before the seal.

The structural rows add no extra primitive. The hsame row transports the displayed coordinates, the Cont row replays the same route, and P, N source and name the packet. Since quotient classes, selected representatives, host equality, choice, trichotomy, and ambient completeness are not coordinates of M, the listed rows exhaust the local certificate surface. □

Theorem 6948.3 (Méray Cauchy-real RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MerayCauchyReal}^{\uparrow}$ *packet must first read the displayed Cauchy source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the componentwise* hsame *transport, the displayed* Cont *replay, the* Pkg *provenance, and the local* NameCert *row. The handoff exports no quotient stream equality, selected Cauchy representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 6948.2. They force every accepted read to stay inside $A, W, R, D, E, H, C, P, N$.

The pre-seal route is explicit. A RegSeqRat consumer receives R only as the readback of the displayed windows W over the Cauchy source A, with every tolerance charged to D.

The Real-facing route is downstream of the same readback. The seal row E is available only after A, W, R, D have been displayed and replayed through H, C, P, N. Thus the historical Cauchy-real presentation hands off to regular rational and real-name consumers without adding any quotient, selector, host equality, choice, trichotomy, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MerayCauchyReal}^{\uparrow}}$.

Constructive story: A $\mathsf{MerayCauchyReal}^{\uparrow}$ packet is generated from a displayed rational Cauchy source, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MerayCauchyReal}^{\uparrow}}$, seed)

Scope closed: Definition 6948.1, Definition 6948.2, and Definition 6948.3 seed the historical quotient-free Cauchy-real presentation boundary over StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MerayCauchyReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real construction, selected Cauchy representatives, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-source display, StreamName locality, RegSeqRat handoff, DyadicRatCore tolerance accounting, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.MerayCauchyRealUp.

Chapter 6949

A Concrete Naming Certificate for $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$

The $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ packet realizes the finite-budget route named by the Real completeness vision of chapter 9238. It separates observation budget, selector budget, stream-window access, regular rational readback, dyadic tolerance accounting, and terminal real-seal admission before any completion-facing consumer can treat the route as a real-name handoff. The packet sits over chapter 126, chapter 301, chapter 291, chapter 38, and the CauchyLimitSeal consumer boundary already used by the Real-completion siblings.

Definition 6949.1 (Real completion budget-router carrier)**.** *A* $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (o, s, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *o is the observation-budget row. It bounds which StreamName windows may be inspected.*
2. *s is the selector-budget row. It records the finite selector requests allowed before a completion-facing route is admitted.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row available under o.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the request, windows, readback, limit-seal handoff, and terminal seal.*
6. *L is the CauchyLimitSeal-compatible handoff row consuming only o, s, W, R, D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after the budgeted window and readback route has passed through L.*
8. *H records componentwise* hsame *transport for the budget, window, readback, tolerance, seal, replay, provenance, and local name rows.*
9. *C records displayed* Cont *replay along the finite route from o, s, W, R, D through L to E, H, P, N.*
10. *P is the* Pkg *provenance over* $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionBudgetRouter}^{\uparrow}}$ *row.*

The classifier accepts only packets whose completion-facing reads are routed through these rows. It has no coordinate for unbounded observation, an external selector, quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6949.2 (Real completion budget-router NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ *carrier* $B = (o, s, W, R, D, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionBudgetRout}}$ *surface consists exactly of budget admission, selector admission, StreamName window locality, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyLimitSeal-compatible handoff, RealUp seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6949.1. Its coordinates expose the two budget rows, the window and readback rows, the dyadic ledger, the seal handoff, the terminal seal, and the structural transport, replay, provenance, and naming rows.

The route order is forced by the carrier shape. A StreamName read is admitted only under o, a selector read only under s, a RegSeqRat row only as readback from W, and a tolerance comparison only through D.

The handoff and endpoint rows add no second source. The row L consumes the same budgeted W, R, D route, and E is exposed only after that route has been replayed through H, C, P, N. Hence the listed rows exhaust the local NameCert surface. □

Theorem 6949.3 (Real completion budget-router non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or CauchyLimitSeal-facing consumer of an accepted* $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ *packet reads only the observation budget, selector budget,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, CauchyLimitSeal-compatible handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. It cannot bypass the budget rows or import an unbounded observation service, external selector, quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 6949.2 to identify the only rows available to any accepted consumer. The window row W is behind o, the readback row R is behind W, and all tolerance comparisons are behind D.

Now read the completion-facing rows. The CauchyLimitSeal-compatible handoff L and terminal seal E consume the same displayed budgeted route; neither row carries a detached selector or completed real object.

Finally replay through H, C, P, N. These structural rows preserve the displayed route and name its provenance, so a consumer has no remaining coordinate from which an unbudgeted observation, quotient, chosen limit, host equality, choice principle, or ambient completeness theorem could be read. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ consumer can be audited by checking that its first reads are the observation-budget and selector-budget rows before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, CauchyLimitSeal, or $\mathsf{Real}^{\uparrow}$ rows are consumed.

Independence witness: Nearest siblings: streamname, regseqrat, dyadicratcore, real, and cauchy-limit-seal routes. The router is not bijective to any one sibling because it carries the two budget gates together with the downstream window, readback, tolerance, handoff, and seal rows; deleting either budget row changes the admitted consumer route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionBudgetRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionBudgetRouter}^{\uparrow}$ packet is generated from observation-budget and selector-budget rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyLimitSeal-compatible handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionBudgetRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6949.1, Definition 6949.2, and Definition 6949.3 seed the finite budget-router route for Real completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionBudgetRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, host real equality, countable choice, unbounded observation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked budget admission, selector admission, StreamName locality, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyLimitSeal-compatible handoff, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionBudgetRouterUp.

Chapter 6950

A Concrete Naming Certificate for $\mathsf{DirichletUniformTest}^{\uparrow}$

Following Hardy's treatment of Dirichlet's test for uniform convergence, the $\mathsf{DirichletUniformTest}^{\uparrow}$ packet records a finite certificate for a source function-series window whose bounded partial sums and controlled coefficient variation produce a uniform Cauchy handoff. It sits beside the real-series carrier of chapter 1581, the uniform Cauchy criterion of chapter 435, the uniform-limit boundary of chapter 463, the finite StreamName schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only displayed finite rows; it does not import host uniform convergence, selected limits, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6950.1 (Dirichlet uniform test carrier)**.** *A* $\mathsf{DirichletUniformTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, A, V, W, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing only finite function-series windows and their inspected partial sums.*
2. *B is the bounded-partial-sum ledger, giving a displayed bound for every inspected source window.*
3. *A is the coefficient monotonicity ledger, recording the finite order comparisons used by the coefficient row on the inspected tail.*
4. *V is the coefficient variation ledger, recording the finite total-variation budget that replaces any appeal to host monotone convergence.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule naming the source indices and parameter windows visible to the test.*
6. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *handoff row, storing the uniform tail-window request produced by B, A, V, W.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational estimates exported by the tail-window ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic and regular readbacks have been displayed.*
9. *H records componentwise* hsame *transport for the source, boundedness, coefficient, variation, window, handoff, readback, seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from S, W into B, A, V, from those ledgers into U, and from U through R into E.*
11. *P is the* Pkg *provenance over* $\mathsf{DirichletUniformTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, B, A, V, W, U, R, E, H, C, P, N$. It has no coordinate for a selected infinite sum, host uniform convergence, quotient equality of functions, countable choice, or ambient completeness.

Theorem 6950.2 (Dirichlet uniform test NameCert obligations)**.** *For every accepted* $\mathsf{DirichletUniformTest}^{\uparrow}$ *carrier $K = (S, B, A, V, W, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformTest}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-partial-sum exposure, coefficient monotonicity exposure, coefficient-variation exposure, StreamName window admission, UniformCauchyCriterion handoff readiness, RegSeqRat readback, Real seal non-escape, componentwise transport, displayed replay, provenance, and local naming. A downstream consumer may read a uniform Cauchy tail certificate only through those rows.*

Proof. Project the carrier rows of Definition 6950.1. The source row S gives only finite function-series windows, while B, A, V give the bounded partial-sum, monotonicity, and variation ledgers visible on those windows.

The window schedule W fixes which tails may be inspected before the handoff row U is read. Thus a consumer cannot reach a uniform tail claim until the same finite source, boundedness, coefficient, and variation rows have already been displayed.

Finally read R, E, H, C, P, N. The regular readback and Real seal are downstream of U, and the structural rows preserve exactly the same displayed route. No selected infinite sum, quotient function equality, host uniform convergence theorem, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness is introduced. □

Theorem 6950.3 (Dirichlet uniform test uniform Cauchy handoff)**.** *Every* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *or* $\mathsf{UniformLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirichletUniformTest}^{\uparrow}$ *carrier factors through the finite tail-window route*

$$\mathsf{DirichletUniformTest}^{\uparrow} \longmapsto \mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{UniformCauchyCr}$$

The handoff row U receives only the displayed bounded-partial-sum ledger B, coefficient monotonicity ledger A, variation ledger V, and window schedule W before it exports a uniform Cauchy request.

Proof. Start with the rows S and W. They determine the inspected source windows and the tail windows visible to the packet, so every later estimate is tied to a finite $\mathsf{StreamName}^{\uparrow}$ route.

Next use B, A, V to form the Dirichlet tail estimate on the displayed window. The bounded partial sums provide the uniform bound, the monotonicity ledger orders the coefficient row, and the variation ledger supplies the finite loss budget needed to route the estimate into U.

The readback row R and seal row E are then consumed after the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ handoff. Therefore a $\mathsf{UniformLimit}^{\uparrow}$ consumer receives only the finite uniform Cauchy surface supplied by U, R, E, H, C, P, N, not a selected host limit, quotient equality of functions, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DirichletUniformTestUp.DirichletUniformTestCarrier uniform cauchy handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletUniformTest}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletUniformTest}^{\uparrow}$ packet is generated from RealSeries source windows, bounded partial-sum rows, coefficient monotonicity and variation ledgers, StreamName windows, a UniformCauchyCriterion handoff, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletUniformTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6950.1, Definition 6950.2, and Definition 6950.3 seed the finite uniform Dirichlet-test route over source windows, bounded partial sums, coefficient variation, uniform tail windows, regular readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletUniformTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host uniform convergence, selected sums, quotient function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-partial-sum exposure, coefficient monotonicity and variation exposure, uniform tail-window routing, readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DirichletUniformTestUp.

Chapter 6951

A Concrete Naming Certificate for $\mathsf{HermiteHadamard}^{\uparrow}$

Following Hermite and Hadamard, the $\mathsf{HermiteHadamard}^{\uparrow}$ packet records the interval convexity route by which a convex-function source, closed interval endpoints, the midpoint comparison, and the interval-average comparison produce the two-sided Hermite–Hadamard inequality surface. It consumes the Jensen route of chapter 2235, the finite convex comparison discipline of chapter 6313, the convex-carrier surface of chapter 208, the real-name boundary of chapter 38, the rational subdivision source of chapter 37, the vector carrier of chapter 47, and the endpoint comparison discipline of chapter 1674. The packet names the Real-facing finite interval readback for the classical convex-analysis inequality; it does not import a host integral, quotient equality of real names or functions, selected extrema, trichotomy, countable choice, or ambient real completeness.

Definition 6951.1 (Hermite–Hadamard carrier)**.** *A* $\mathsf{HermiteHadamard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (F, I, M, A, J, K, R, Q, T, C, P, N)$$

with the following displayed rows:

1. *F is the convex-function source row, recording the displayed domain, codomain, convexity ledger, and function-read route.*
2. *I is the closed-interval endpoint row, recording endpoints $a \leq b$, their* $\mathsf{Real}^{\uparrow}$ *reads, and the* $\mathsf{RealMinMax}^{\uparrow}$ *boundary comparison.*
3. *M is the midpoint row, recording the displayed midpoint $(a+b)/2$ and its* $\mathsf{Rat}^{\uparrow}$ *coefficient read inside the interval.*
4. *A is the interval-average row, recording the finite rational subdivision windows, normalized averaging weights, and the displayed average comparison surface.*
5. *J is the* $\mathsf{JensenInequality}^{\uparrow}$ *handoff row, recording the finite midpoint and subdivision Jensen calls used on the interval carrier.*
6. *K is the* $\mathsf{KaramataInequality}^{\uparrow}$ *handoff row, recording the finite convex-order comparison between symmetric endpoint windows and interior subdivision windows.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row for the displayed two-sided inequality*

$$f((a+b)/2) \leq A_{\mathrm{int}}(f, a, b) \leq (f(a) + f(b))/2.$$

8. *Q is the endpoint row, recording the real comparison data used when a consumer reads the lower midpoint bound and upper endpoint bound together.*
9. *T records componentwise* hsame *transport for F, I, M, A, J, K, R, Q and the local naming row.*
10. *C records displayed* Cont *replay from the endpoint and subdivision rows through the Jensen and Karamata handoffs, real comparison, endpoint row, provenance, and local name.*
11. *P is the* Pkg *provenance over* $\mathsf{HermiteHadamard}^{\uparrow}$, $\mathsf{JensenInequality}^{\uparrow}$, $\mathsf{KaramataInequality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{RealMinMax}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HermiteHadamard}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, M, A, J, K, R, Q, T, C, P, N$. It has no coordinate for a host integral, quotient equality of functions or real names, a selected supremum, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.HermiteHadamardUp`

Theorem 6951.2 (Hermite–Hadamard NameCert obligations)**.** *For every accepted* $\mathsf{HermiteHadamard}^{\uparrow}$ *carrier $H = (F, I, M, A, J, K, R, Q, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HermiteHadamard}^{\uparrow}}$ *surface consists exactly of carrier admission for the convex-function source row, closed-interval endpoint row, midpoint row, interval-average row, Jensen handoff row, Karamata handoff row, real comparison row, endpoint row, structural rows T, C, P, and local naming row N; endpoint accountability through I, Q; midpoint and average accountability through M, A; Jensen and Karamata comparison accountability through J, K, R; and componentwise transport, replay, provenance, and naming for the same finite interval route.*

Proof. Project the accepted packet $H = (F, I, M, A, J, K, R, Q, T, C, P, N)$. The carrier definition exposes exactly the convex-function source, closed interval, midpoint, averaging, Jensen handoff, Karamata handoff, real comparison, endpoint comparison, transport, replay, provenance, and naming rows.

The endpoint and midpoint rows are read before any inequality row. Thus I supplies the two endpoint real names and boundary comparison, while M supplies the displayed midpoint and its rational coefficient read.

The average row A then supplies the finite subdivision windows and normalized rational weights. The handoff rows J, K record the Jensen and Karamata routes that turn those finite rows into the lower midpoint comparison and upper endpoint comparison.

Finally R, Q, T, C, P, N expose the two-sided real comparison, endpoint comparison, transport, replay, provenance, and local naming. Since no host integral, quotient-equality object, selected supremum, or ambient completeness coordinate is present, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.HermiteHadamardUp.HermiteHadamardTasteGate single carrier alignment`

Theorem 6951.3 (Hermite–Hadamard Jensen interval route)**.** *Every accepted* $\mathsf{HermiteHadamard}^{\uparrow}$ *carrier routes its public two-sided inequality through*

$$I, M, A \longmapsto J, K \longmapsto R, Q \longmapsto T, C, P, N.$$

Consequently a consumer must read the closed interval endpoints, midpoint row, finite rational subdivision averages, $\mathsf{JensenInequality}^{\uparrow}$ *handoff row,* $\mathsf{KaramataInequality}^{\uparrow}$ *handoff row,* $\mathsf{Real}^{\uparrow}$ *comparison row,* $\mathsf{RealMinMax}^{\uparrow}$ *endpoint row, componentwise transport, replay, provenance, and local naming before it can cite the Hermite–Hadamard inequality surface.*

Proof. Start with a consumer read of the two-sided comparison. By Definition 6951.1, the only interval data are the endpoint row I, midpoint row M, and finite average row A.

The Jensen handoff row J is downstream of those interval rows. It supplies the midpoint and subdivision convexity comparisons only after the finite rational weights and interval membership data have been displayed.

The Karamata handoff row K is read on the same finite interval windows. It records the convex-order route from symmetric endpoint windows to the interior average comparison, so the upper endpoint inequality cannot be cited independently of the displayed subdivision ledger.

Finally R, Q, T, C, P, N expose the real comparison, endpoint comparison, transport, replay, provenance, and local naming. Thus no consumer can bypass the finite interval route by invoking a host integration theorem or an ambient real-completeness principle. □

Falsifiable prediction: Every $\mathsf{HermiteHadamard}^{\uparrow}$ consumer reads a convex-function source row, closed interval endpoint row, midpoint row, finite rational subdivision-average row, JensenInequality handoff row, KaramataInequality handoff row, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the Hermite–Hadamard inequality surface is public.

Independence witness: jenseninequality, karamatainequality, convexset, real, rat, vecspace, realminmax: $\mathsf{HermiteHadamard}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval endpoints, midpoint data, finite subdivision averages, Jensen handoffs, Karamata handoffs, real comparison, endpoint comparison, transport, replay, provenance, and local NameCert data as one finite interval inequality packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HermiteHadamard}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HermiteHadamard}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6951.1, Definition 6951.2, and Definition 6951.3 seed the finite Hermite–Hadamard interval route over convex-function, endpoint, midpoint, finite-average, Jensen, Karamata, ConvexSet, Real, Rat, VecSpace, and RealMinMax rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HermiteHadamard}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.HermiteHadamardUp`

Bridge status: none

Not claimed: This chapter does not close host integrals, quotient equality of real names or functions, selected extrema, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint accountability, midpoint and subdivision-average exactness, Jensen and Karamata handoff exactness, Real comparison nonescape, endpoint comparison handoff, and public export scope for BEDC.Derived.HermiteHadamardUp.

Constructive story: A $\mathsf{HermiteHadamard}^{\uparrow}$ packet is generated from a convex-function source row, closed interval endpoint row, midpoint row, finite rational subdivision-average row, JensenInequality and KaramataInequality handoff rows, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6952

A Concrete Naming Certificate for $\mathsf{FiniteRankOperator}^{\uparrow}$

Following finite-dimensional linear algebra and bounded-operator theory, the $\mathsf{FiniteRankOperator}^{\uparrow}$ packet records the displayed finite-rank route by which a linear operator factors through a finite-dimensional image row before compactness, spectral, Hilbert, or functional-analysis consumers may read it. It sits after the vector-space carrier of chapter 47, the linear-map surface of chapter 48, the finite matrix readback of chapter 49, the bounded-linear-operator boundary of chapter 1644, and the compact-operator sibling of chapter 1952. The packet names the finite image ledger and its operator handoff; it does not assert compact approximation, spectral decomposition, a selected basis for every image, quotient function equality, or an ambient functional-analysis theorem.

Definition 6952.1 (Finite-rank operator carrier)**.** *A* $\mathsf{FiniteRankOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, T, I, B, M, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source and target* $\mathsf{VecSpace}^{\uparrow}$ *rows.*
2. *T is the* $\mathsf{LinMap}^{\uparrow}$ *row from X to Y; when norm control is requested, the same row is also read through* $\mathsf{BoundedLinearOperator}^{\uparrow}$*.*
3. *I is the finite image-window row, listing displayed target vectors that span the image of T for the requested source window.*
4. *B is the finite-dimensional basis or spanning ledger for I, read only as BEDC rows and not as a host basis selector.*
5. *M is the optional* $\mathsf{Mat}^{\uparrow}$ *presentation row tying the displayed coefficients of T to the finite image ledger.*
6. *E is the exactness row saying that every displayed operator value in the request lands in the span recorded by I and B.*
7. *H records componentwise* hsame *transport for the vector-space endpoints, operator row, image ledger, basis ledger, matrix presentation, exactness row, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from an operator read through the finite image rows and back to bounded, compact, spectral, Hilbert, or functional-analysis consumers.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteRankOperator}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRankOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient operator algebra, an infinite basis, a quotient representative of the image, a compactness theorem, or a spectral theorem.

Theorem 6952.2 (Finite-rank operator NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRankOperator}^{\uparrow}$ *carrier* $R = (X, Y, T, I, B, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteRankOperator}^{\uparrow}}$ *surface consists exactly of carrier admission, source and target vector-space exposure, linear-map action exposure, finite image-window exactness, optional matrix readback compatibility, bounded-operator handoff when norm control is displayed, compact-operator handoff through the finite image rows, componentwise transport, continuation replay, provenance, and local naming.*

Any $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, *or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that reads finite-rank data from R must first pass through $X, Y, T, I, B, M, E, H, C, P, N$. The read exports no ambient compactness theorem, no selected infinite approximation scheme, no spectral decomposition, no quotient image equality, and no host operator-space object.*

Proof. Project the carrier of Definition 6952.1. The vector-space endpoints X, Y, the operator row T, the finite image rows I, B, and the exactness row E are displayed before any consumer-facing row is available.

Matrix compatibility is only a readback route. If M is present, its coefficients are tied to the same T, I, B rows; if it is absent, the carrier still exposes finite image exactness directly through E.

Bounded and compact consumers receive only the finite route recorded by C and P. The bounded handoff reads the norm-controlled operator row when it is displayed, while the compact handoff reads finite-rankness through the finite image ledger rather than through an ambient compactness theorem.

Transport is componentwise through H, and naming is local through N. Since no carrier coordinate stores an infinite basis, a selected approximation sequence, quotient image equality, spectral decomposition, or host operator space, the obligations listed above exhaust the $\mathsf{FiniteRankOperator}^{\uparrow}$ naming surface. □

Falsifiable prediction: Every accepted $\mathsf{FiniteRankOperator}^{\uparrow}$ consumer must display $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, finite image-window, spanning-ledger, exactness, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before finite-rank operator data is available.

Independence witness: vecspace, linearmap, matrix, boundedlinearoperator, compactoperator: $\mathsf{FiniteRankOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source and target vector rows, a linear action row, finite image-window rows, spanning or basis ledger rows, optional matrix readback, exactness, transport, replay, provenance, and local naming in one operator-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRankOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRankOperator}^{\uparrow}$ packet is generated from source and target VecSpaceUp rows, a LinearMapUp action row, a finite image-window row, a spanning or basis ledger, optional MatrixUp readback, finite image exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteRankOperator}^{\uparrow}}$, seed)

Scope closed: Definition 6952.1, Definition 6952.2, the falsifiable prediction, and the independence witness seed the finite-rank operator interface over displayed vector-space, linear-map, matrix, bounded-operator, compact-operator, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteRankOperator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim compact approximation, spectral decomposition, selected infinite bases, quotient image equality, an ambient operator-space object, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite image exactness, optional matrix compatibility, bounded and compact handoff, classifier transport, and consumer non-escape for BEDC.Derived.FiniteRankOperatorUp.

Chapter 6953

A Concrete Naming Certificate for $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style net convergence, the CauchyNetConvergenceCriterio packet records the finite criterion by which a displayed Cauchy net is read as a convergence candidate only through named windows, regular rational tolerances, and the terminal real seal. It sits after the Cauchy-net carrier of chapter 589, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the Real seal of chapter 38. The criterion is a finite BEDC NameCert surface; it does not construct quotient filters, select a limit, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6953.1 (Cauchy net convergence criterion carrier)**.** *A* $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, S, R, D, W, T, E, H, A, P, N)$$

with the following rows:

1. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *source row exposing the directed Cauchy-net request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect named tails of the net.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, recording precision cells and refinement slack for each request.*
5. *W is the window-eventuality row, sending a requested tolerance to a displayed finite tail window.*
6. *T is the tail-coherence row, recording that later displayed windows preserve the same rational tolerance decision.*
7. *E is the Real-seal eligibility row, exposed only after C, S, R, D, W, T are visible.*
8. *H records componentwise* hsame *transport for the source, window, readback, dyadic, eventuality, coherence, and seal-eligibility rows.*
9. *A records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient filters, selected limits, countable choice, host equality of nets, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyNetConvergenceCriterionUp.CauchyNetConvergenceCriterionUp`
Lean4 ✓ `BEDC.Derived.CauchyNetConvergenceCriterionUp.CauchyNetConvergenceCriterionTasteGate`
`single carrier alignment`

Theorem 6953.2 (Cauchy net convergence criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$ *carrier*

$$K = (C, S, R, D, W, T, E, H, A, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}}$ *surface consists exactly of Cauchy-net source exposure through C, stream-window inspection through S, regular rational readback through R, dyadic budget exposure through D, eventual tail-window selection through W, tail coherence through T, Real-seal eligibility through E, componentwise transport through H, displayed continuation replay through A, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The source row C supplies the directed Cauchy-net request, while S and R expose the finite stream window and rational readback used to inspect that request.

The dyadic budget row D fixes the tolerance cell. The eventuality row W then records which displayed tail window answers the request, and the coherence row T records that later displayed windows keep the same tolerance decision.

Only after those rows are visible may the eligibility row E reach the Real seal. The structural rows H, A, P, N transport, replay, source, and name the same finite route, so the local NameCert cannot export a quotient filter, a selected limit, countable choice, host net equality, or ambient Real completeness. □

Theorem 6953.3 (Cauchy net convergence criterion Real seal handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$ *packet must read the handoff through the displayed route*

$$C, S, R, D, W, T, E \longmapsto H, A, P, N.$$

Consequently a Cauchy-net convergence request reaches $\mathsf{Real}^{\uparrow}$ *only after the StreamName window, RegSeqRat readback, DyadicRatCore budget, eventuality row, and tail-coherence row are all visible.*

Proof. Begin with the Cauchy-net source row C and the finite window row S. They determine which displayed tail of the net is inspected for the current tolerance request.

The regular rational row R and dyadic row D provide the comparison budget. The eventuality row W and coherence row T bind that budget to a finite tail route rather than to an unrecorded filter representative or a chosen limit.

The Real-seal row E is therefore downstream of the displayed Cauchy-net, window, readback, and budget data. Transport, replay, provenance, and local naming through H, A, P, N preserve the same route, so a Real-facing consumer obtains no bypass around StreamName, RegSeqRat, DyadicRatCore, or the terminal Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CauchyNetConvergenceCriterion}^{\uparrow}$ packet is generated from a CauchyNet source row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore budget row, eventual tail-window evidence, tail coherence, Real-seal eligibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyNetConvergenceCriterion}\uparrow}$, seed)

Scope closed: Definition 6953.1, Definition 6953.2, and Definition 6953.3 seed the finite Cauchy-net convergence criterion for StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyNetConvergenceCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host net equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular rational read-back, dyadic budget exposure, eventuality, tail coherence, Real-seal eligibility, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNetConvergenceCriterionUp.

Chapter 6954

A Concrete Naming Certificate for $\mathsf{CauchyCompletionPullback}^{\uparrow}$

Following Bishop and Bridges on constructive completion and finite comparison of Cauchy data, the $\mathsf{CauchyCompletionPullback}^{\uparrow}$ packet records the compatibility surface where two displayed completion routes meet over one target seal. It sits beside the completion functor route of chapter 423, the metric-completion boundary of chapter 1506, the product-strength packet of chapter 6130, the terminal exact-boundary route of chapter 1120, and the regular-rational and stream-window sources of chapter 301 and chapter 126. The packet is a finite L10 compatibility object: it does not import host pullbacks, quotient completions, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6954.1 (Cauchy completion pullback carrier)**.** *A* $\mathsf{CauchyCompletionPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (A, B, T, f, g, W_A, W_B, R_A, R_B, H, C, P_0, N)$$

with the following displayed rows:

1. *A is the left* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *source row, exposing only its displayed completion request and local naming surface.*
2. *B is the right* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing only its displayed metric-completion request and local naming surface.*
3. *T is the shared terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by both completion routes.*
4. *f is the left leg, a finite route from A through the relevant* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows to T.*
5. *g is the right leg, a finite route from B through the relevant* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows to T.*
6. *W_A and W_B are the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected by the two legs.*
7. *R_A and R_B are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
8. *H records componentwise* hsame *transport for the two sources, two legs, two windows, two readbacks, shared seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the two routes into the same terminal seal.*

10. *P_0 is the* Pkg *provenance over* $\mathsf{CauchyCompletionPullback}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionPullback}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, T, f, g, W_A, W_B, R_A, R_B, H, C, P_0, N$. It has no coordinate for a host pullback object, quotient completion, selected representative, countable-choice schedule, detached limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6954.2 (Cauchy completion pullback NameCert obligations)**.** *For every accepted* CauchyCompletionPullb *carrier*

$$P = (A, B, T, f, g, W_A, W_B, R_A, R_B, H, C, P_0, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionPullback}^{\uparrow}}$ *surface consists exactly of carrier admission, leg compatibility, shared target sealing, synchronized finite windows, transport, replay, provenance, and non-escape. In particular, every public compatibility read must display the left* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row, the right* $\mathsf{MetricCompletion}^{\uparrow}$ *row, both legs, both* $\mathsf{StreamName}^{\uparrow}$ *windows, both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the common* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P_0, N before it may treat the two completion routes as meeting.*

Proof. Project the carrier of Definition 6954.1. Carrier admission exposes exactly the two source rows, the two leg rows, the two finite windows, the two regular-rational readbacks, the shared seal, and the four structural rows.

Leg compatibility is read from f and g together with W_A, W_B, R_A, R_B. The left leg must pass from the $\mathsf{CauchyCompletionMonad}^{\uparrow}$ row through its displayed window and readback; the right leg must pass from the $\mathsf{MetricCompletion}^{\uparrow}$ row through its displayed window and readback.

The shared target seal is the single row T. Since both legs name the same terminal $\mathsf{Real}^{\uparrow}$ coordinate only after the windows and readbacks have been displayed, the packet supplies a finite compatibility surface rather than a host pullback or quotient-completion equality.

Finally H, C, P_0, N provide componentwise transport, replay, provenance, and local naming for that same finite route. No remaining coordinate can export selected limits, countable choice, detached representatives, or ambient completeness, so the listed rows exhaust the local NameCert obligations. □

Theorem 6954.3 (Cauchy completion pullback synchronized windows)**.** *Every accepted* CauchyCompletionPullback *read factors through the synchronized finite-window route*

$$\mathsf{CauchyCompletionMonad}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

by reading $A, B, W_A, W_B, R_A, R_B, T, H, C, P_0, N$. The two completion legs may meet only at the displayed seal T over those windows and readbacks.

Proof. Start with the two source rows A and B. A completion-facing consumer receives no direct comparison between them; it receives only the leg rows f, g and the displayed windows W_A, W_B.

The windows determine the regular-rational readbacks R_A, R_B. These readbacks are the only data that can be replayed toward the shared seal, so the consumer cannot replace them with quotient Cauchy classes or selected completed representatives.

After the readbacks have been displayed, the terminal row T supplies the shared $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming keep both legs on the same finite route, and no other row remains available. □

Theorem 6954.4 (Cauchy completion pullback Hausdorff separation route)**.** *Any consumer treating a* $\mathsf{CauchyCompletionPullback}^{\uparrow}$ *comparison as separated must display the two Hausdorff separation witnesses from* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *or* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *after the synchronized window row and before exporting pullback equality. The consumer cannot replace those witnesses with quotient-completion equality, selected representatives, host equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 6954.1. The two source rows, two leg rows, synchronized windows, readbacks, and shared seal are the only data available to the Cauchy pullback comparison.

Apply Definition 6954.3. It fixes the common finite source windows W_A, W_B and the readback rows R_A, R_B before either completion leg may meet the terminal $\mathsf{Real}^{\uparrow}$ seal.

To read the comparison as separated, the consumer must then route through the Hausdorff completion siblings. The relevant rows are the two explicit separation witnesses supplied by $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ or $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$, not a quotient-completion equality or a chosen completed representative.

Only after those witnesses are displayed may a pullback equality be exported as public BEDC data. Transport, replay, provenance, and local naming keep the export inside the finite carrier, so countable choice, host equality, and ambient Real completeness remain unavailable. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionPullback}^{\uparrow}$ read that claims two completion routes meet must display a CauchyCompletionMonad source, a MetricCompletion source, two finite StreamName windows, two RegSeqRat readbacks, two leg rows, a shared RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: cauchycompletionfunctor, metriccompletion, cauchy-completion-strength, real-completion-exact-boundary, regseqrat, streamname: $\mathsf{CauchyCompletionPullback}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds two distinct completion sources, two leg rows, synchronized windows, paired readbacks, one terminal seal, transport, replay, provenance, and local naming in a single compatibility packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionPullback}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionPullback}^{\uparrow}$ packet is generated from a CauchyCompletionMonad row, a MetricCompletion row, a shared RealUp seal, two leg rows, two StreamName windows, two RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6954.1, Definition 6954.2, and Definition 6954.3 seed the finite compatibility surface for two completion routes over one terminal seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host pullbacks, quotient completions, selected limits, countable choice, detached completed representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, leg compatibility, shared target sealing, synchronized finite windows, transport, replay, provenance, and non-escape for BEDC.Derived.CauchyCompletionPullbackUp.

Chapter 6955

A Concrete Naming Certificate for ResidualSubstitutionCompatibilityWindow$^{\uparrow}$

Following the standard residual-substitution compatibility boundary from rewriting theory, the ResidualSubstitutionCompatibilityWindow$^{\uparrow}$ packet records the finite MetaCIC window on which a residual substitution claim may be checked before any confluence consumer reads it. It realizes the confluence question of chapter 9386 as concrete NameCert data and sits beside the critical-path route of chapter 765, the closed substitution boundary of chapter 538, and the bounded checker surface of chapter 3827. The packet is a bounded residual frontier; it does not assert full residual-substitution compatibility, subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Definition 6955.1 (Residual-substitution compatibility window carrier)**.** *A* ResidualSubstitutionCompatibilityWind *carrier is a finite* BHist *packet*

$$W = (T, R, S, J, B, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed MetaCIC observation window whose source, reduct, and residual positions are all finite.*
2. *R is the residual-frontier row, recording only the residual redexes and branch-local substitution positions visible inside T.*
3. *S is the closed substitution boundary row read through chapter 538.*
4. *J is the local join row, naming the candidate join data admitted by the displayed residual frontier.*
5. *B is the bounded checker-status row inherited from chapter 3827.*
6. *O is the obstruction-retention row, recording that the bounded window has not supplied an unrestricted substitution-compatibility theorem.*
7. *H records componentwise* hsame *transport for the typed window, residual frontier, substitution boundary, local join, checker status, obstruction, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for the same bounded residual route.*
9. *P is the* Pkg *provenance over* ResidualSubstitutionCompatibilityWindow$^{\uparrow}$*,* MetaCICCriticalPath$^{\uparrow}$*,* ClosedSubstitutionBoundary$^{\uparrow}$*,* MetaCICConfluenceResidualChecker$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $T, R, S, J, B, O, H, C, P, N$. It has no coordinate for an unbounded residual-substitution theorem, global subject reduction, arbitrary closed confluence, host evaluator equality, quotient conversion equality, or kernel consistency.

Theorem 6955.2 (Residual-substitution compatibility window NameCert obligations)**.** *For every accepted* $\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}$ *carrier $W = (T, R, S, J, B, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}}$ *surface consists exactly of typed-window exposure, residual-frontier exposure, closed-substitution-boundary exposure, local-join exposure, bounded-checker status, obstruction retention, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6955.1. The displayed rows are exactly the typed window, residual frontier, closed substitution boundary, local join, bounded checker status, obstruction, transport, replay, provenance, and naming rows.

Read the compatibility boundary in order. The typed window T fixes the finite MetaCIC source window, the residual row R exposes the branch-local residual positions, and the substitution row S keeps the closed substitution boundary visible before any local join is consumed.

The local join row J and checker row B therefore operate only inside the displayed window. The obstruction row O remains part of the same carrier, so the packet cannot be read as a premise-free confluence theorem or as a discharged subject-reduction theorem.

Rows H, C, P, N preserve and name the same finite route. Since no carrier coordinate supplies unbounded compatibility, evaluator equality, quotient conversion equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 6955.3 (Residual-substitution compatibility window critical-path handoff)**.** *Every critical-path or confluence consumer of an accepted* $\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow},\ \mathsf{ClosedSubstitutionBoundary}^{\uparrow},\ \mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$$

through $T, R, S, J, B, O, H, C, P, N$ before it may use the window as residual-substitution evidence. The handoff preserves the bounded obstruction and exports no full confluence theorem.

Proof. Start with a consumer from the critical-path route of chapter 765. That route may inspect residual pressure only after the finite typed window and residual frontier are displayed.

The closed substitution boundary row S is then read as a sibling boundary, not as a subject-reduction theorem. It keeps source closedness, value closedness, shift, substitution, ledger, audit, and local naming data visible before the join row J can be trusted.

Finally the residual checker row B records bounded checker status for that same route, while O keeps the obstruction attached. Transport, replay, provenance, and local naming preserve the window, so the consumer receives bounded residual-substitution evidence and not full closed confluence, evaluator equality, quotient term equality, or kernel consistency. □

Falsifiable prediction: Every accepted $\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}$ read that is consumed as residual-substitution evidence must display a typed MetaCIC window, residual-frontier row, closed substitution boundary row, local-join row, bounded checker-status row, obstruction row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: metaciccriticalpath, closed-substitution-boundary, metacic-confluence-residual-checker, metacic-residual-substitution-boundary: no listed sibling is bijective to ResidualSubstitutionCompa

because this packet jointly binds a typed observation window, residual frontier, closed substitution boundary, local join, bounded checker status, obstruction retention, transport, replay, provenance, and local naming as one bounded compatibility surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ResidualSubstitutionCompatibilityWindow}\uparrow}$.

Constructive story: A $\mathsf{ResidualSubstitutionCompatibilityWindow}^{\uparrow}$ packet is generated from a typed MetaCIC observation window, residual-frontier row, closed substitution boundary row, local-join row, bounded checker-status row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResidualSubstitutionCompatibilityWindow}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6955.1, Definition 6955.2, and Definition 6955.3 seed the bounded residual-substitution compatibility window consumed by MetaCIC confluence routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResidualSubstitutionCompatibilityWindow}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full residual-substitution compatibility, global subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient conversion equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-frontier exposure, substitution-boundary retention, local-join exposure, bounded-checker status, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.ResidualSubstitutionCompatibilityWindowUp.

Chapter 6956

A Concrete Naming Certificate for $\mathsf{BourbakiCauchyFilter}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on Cauchy filters and constructive completion, the $\mathsf{BourbakiCauchyFilter}^{\uparrow}$ packet records the traditional Cauchy-filter presentation as finite BEDC certificate data rather than as a quotient of filters or a selected completion point. It sits after the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion-facing Cauchy-filter surface of chapter 1569. The packet gives Real and Completion consumers a human-origin source row whose public reads stay inside finite BHist, transport, replay, provenance, and local naming rows.

Definition 6956.1 (Bourbaki Cauchy-filter carrier)**.** *A* $\mathsf{BourbakiCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (F_0, B, E, W, R, D, S, H, C, P, N)$$

with the following rows:

1. F_0 *is the filter source row, exposing the displayed Cauchy-filter request that a completion consumer may inspect.*
2. B *is the finite base row, recording the named base elements used for the current filter read.*
3. E *is the entourage and tolerance row, keeping the requested closeness data explicit before any completion handoff.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite observations made from the source.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *accounting row for tolerance cells, slack, and comparison budget.*
7. S *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after* F_0, B, E, W, R, D *have been displayed.*
8. H *records componentwise* hsame *transport of the source, base, tolerance, window, readback, dyadic, and seal rows.*
9. C *records displayed* Cont *replay of the same ordered route.*
10. P *is the* Pkg *provenance over* $\mathsf{BourbakiCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. N *is the local* $\mathsf{NameCert}_{\mathsf{BourbakiCauchyFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, ultrafilter extension, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6956.2 (Bourbaki Cauchy-filter NameCert obligations)**.** *For every accepted* $\mathsf{BourbakiCauchyFilter}^{\uparrow}$ *carrier*

$$F = (F_0, B, E, W, R, D, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BourbakiCauchyFilter}^{\uparrow}}$ *surface consists exactly of filter-source exposure through* F_0*, finite-base exposure through* B*, entourage and tolerance exposure through* E*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *accounting through* D*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* S*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 6956.1. The first three rows expose the Cauchy-filter source, the finite base, and the entourage or tolerance request as displayed data rather than as an ambient filter quotient.

The next four rows order the completion read through finite windows, regular rational readback, dyadic accounting, and the terminal Real seal. Thus a consumer cannot reach S before the rows F_0, B, E, W, R, D have been read.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. No row introduces quotient filter equality, ultrafilters, selected limits, countable choice, trichotomy, or ambient Real completeness. □

Theorem 6956.3 (Bourbaki Cauchy-filter RegSeqRat handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BourbakiCauchyFilter}^{\uparrow}$ *packet must factor its read through the ordered route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto B \longmapsto E \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is read only after the filter source, finite base, tolerance row, and finite window service are visible, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read only after the dyadic accounting row has been displayed.*

Proof. Begin with the NameCert obligations of Definition 6956.2. They list F_0, B, E, W, R, D, S as the only analytic rows available before the structural certificate rows.

The ordered route first exposes the Cauchy-filter source and finite base. The tolerance row then fixes which finite windows are relevant, so the $\mathsf{StreamName}^{\uparrow}$ service can be read without choosing a representative sequence or a limit point.

The $\mathsf{RegSeqRat}^{\uparrow}$ row reads those windows as regular rational data, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row accounts for the displayed tolerance budget before the $\mathsf{Real}^{\uparrow}$ seal is reached. Transport, replay, provenance, and local naming preserve this order, so the handoff exports no quotient filter equality, ultrafilter argument, selected limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{BourbakiCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BourbakiCauchyFilter}^{\uparrow}$ packet is generated from a Cauchy-filter source row, finite base row, entourage and tolerance row, StreamName windows, RegSeqRat readback, DyadicRatCore accounting, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BourbakiCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6956.1, Definition 6956.2, and Definition 6956.3 seed the finite Bourbaki Cauchy-filter presentation consumed by Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BourbakiCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, ultrafilters, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-source exposure, finite-base exposure, tolerance routing, StreamName windows, RegSeqRat readback, DyadicRatCore accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BourbakiCauchyFilterUp.

Chapter 6957

A Concrete Naming Certificate for AzumaHoeffdingFiniteMartingale$^{\uparrow}$

Following Azuma and Hoeffding, the AzumaHoeffdingFiniteMartingale$^{\uparrow}$ packet records the finite bounded-difference martingale route by which a displayed martingale window receives a Hoeffding-style exponential tail bound. It sits after the finite martingale carrier of chapter 190, the bounded independent-sum surface of chapter 5726, the predictable-variance concentration boundary of chapter 6408, the random-variable readback of chapter 185, and the terminal value rows of chapter 38. The chapter names the finite concentration packet; it does not import almost-sure limits, measure completion, quotient event equality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 6957.1 (Azuma–Hoeffding finite martingale carrier)**.** *An* AzumaHoeffdingFiniteMartingale$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (\Omega, X, M, B, S, H, F, R, T, C, P, N)$$

with the following displayed rows:

1. Ω *is the probability-space row supplying the finite event and probability surface used by the concentration read.*
2. X *is the* RandomVar$^{\uparrow}$ *endpoint-family row for the displayed martingale differences and terminal martingale value.*
3. M *is the* Martingale$^{\uparrow}$ *finite-window row carrying adaptedness and conditional-expectation reads.*
4. B *is the bounded-difference row, recording the finite bound schedule for each inspected martingale difference.*
5. S *is the finite stopping-free window row, naming the displayed horizon and the prefix over which the tail estimate is read.*
6. H *is the* HoeffdingFiniteSum$^{\uparrow}$ *comparison row, used only for the finite bounded-difference exponential estimate inside the displayed window.*
7. F *is the* FreedmanInequality$^{\uparrow}$ *sibling boundary row, recording that variance-budget concentration is adjacent but not collapsed into the bounded-difference route.*
8. R *is the* Real$^{\uparrow}$ *tail-threshold row recording the displayed deviation threshold and exponential upper bound as terminal real-name reads.*

9. *T records componentwise* hsame *transport for $\Omega, X, M, B, S, H, F, R$ and the local naming row.*
10. *C records displayed* Cont *replay from a tail-event consumer through the probability, random-variable, martingale, bounded-difference, finite-window, Hoeffding, Freedman-boundary, and Real rows.*
11. *P is the* Pkg *provenance over* $\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{HoeffdingFiniteSum}^{\uparrow}$, $\mathsf{FreedmanInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $\Omega, X, M, B, S, H, F, R, T, C, P, N$. It has no coordinate for an almost-sure limiting event, completed measure algebra, quotient event equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6957.2 (Azuma–Hoeffding finite martingale NameCert obligations)**.** *For every accepted* $\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}$ *carrier $A = (\Omega, X, M, B, S, H, F, R, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}}$ *surface consists exactly of carrier admission for the probability-space, random-variable, martingale, bounded-difference, finite-window, Hoeffding finite-sum, Freedman boundary, Real tail-threshold, transport, replay, provenance, and local naming rows; bounded-difference exposure through B; finite-window accountability through S; martingale exposure through X, M; finite exponential comparison through H, R; sibling separation through F; and structural preservation through T, C, P, N.*

Proof. Project the accepted packet $A = (\Omega, X, M, B, S, H, F, R, T, C, P, N)$. The carrier displays exactly the probability, random-variable, martingale, bounded-difference, finite-window, Hoeffding, Freedman-boundary, Real, transport, replay, provenance, and local naming rows.

The martingale-facing part is X, M, B, S. The random-variable row supplies the endpoint reads, the martingale row supplies adaptedness and conditional-expectation faces, the bounded-difference row supplies the finite difference bounds, and the window row fixes the inspected horizon.

The concentration-facing part is H, F, R. The Hoeffding finite-sum row is read as the bounded-difference exponential comparison inside the same finite window, while the Freedman row remains a sibling boundary rather than a variance-budget replacement for this route.

Finally T, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no limiting event, completed measure algebra, quotient event equality, choice schedule, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6957.3 (Azuma–Hoeffding finite-window tail handoff)**.** *Every downstream tail-probability consumer of an accepted* $\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}$ *packet receives the deviation bound only after the route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{RandomVar}^{\uparrow}, \mathsf{Martingale}^{\uparrow} \longmapsto B, S \longmapsto \mathsf{HoeffdingFiniteSum}^{\uparrow}, \mathsf{FreedmanInequality}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P$$

has been displayed. The exported tail row is a finite bounded-difference martingale concentration handoff. It is not an almost-sure limit theorem, a measure-completion theorem, a quotient event equality, a countable-choice construction, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness consequence.*

Proof. Start with the carrier rows of Definition 6957.1. A tail-event consumer can inspect the probability-space, random-variable, and martingale rows only through the same finite packet.

Use the bounded-difference and finite-window rows. They record the horizon and the per-step bound schedule before the concentration comparison row is available, so the consumer cannot ask for an unbounded stopping route or a limiting martingale event.

Read the concentration comparison through $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ and the sibling boundary through $\mathsf{FreedmanInequality}^{\uparrow}$. The Hoeffding row supplies the bounded-difference exponential estimate, while the Freedman row keeps predictable-variance concentration adjacent but distinct.

Finally the Real tail-threshold row and the structural rows T, C, P, N export the displayed bound. The absent limiting, measure-completion, quotient-event, choice, and completeness coordinates cannot be projected from this finite route. □

Falsifiable prediction: Every accepted $\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}$ tail read exposes probability-space, RandomVar, Martingale, bounded-difference, finite-window, Hoeffding finite-sum, Freedman boundary, Real threshold, transport, replay, provenance, and local NameCert rows before the tail bound is public.

Independence witness: martingale, hoeffding-finite-sum, freedman-inequality, randomvar, real: no listed sibling supplies the joint bounded-difference martingale concentration packet containing finite martingale, random-variable, bounded-window, Hoeffding comparison, Freedman boundary, Real threshold, transport, replay, provenance, and local NameCert rows while refusing almost-sure limits, measure completion, quotient event equality, countable choice, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6957.1, Definition 6957.2, and Definition 6957.3 seed the finite bounded-difference martingale concentration route over MartingaleUp, HoeffdingFiniteSumUp, FreedmanInequalityUp, RandomVarUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure limits, measure completion, quotient event equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-difference exposure, finite-window accountability, martingale exposure, finite exponential comparison, sibling separation, Real tail-threshold nonescape, and structural preservation for BEDC.Derived.AzumaHoeffdingFiniteMartingaleUp.

Constructive story: An $\mathsf{AzumaHoeffdingFiniteMartingale}^{\uparrow}$ packet is generated from probability-space, random-variable, martingale, bounded-difference, finite-window, Hoeffding finite-sum, Freedman boundary, Real tail-threshold, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6958

A Concrete Naming Certificate for $\mathsf{BishopIntervalNesting}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalNesting}^{\uparrow}$ packet records a constructive nested-interval argument as finite interval-window data rather than as an ambient completeness theorem. It sits between the dyadic endpoint rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet gives Real and Completion consumers a named interval-nesting boundary without importing quotient streams or a host-selected limit.

Definition 6958.1 (Bishop interval nesting carrier)**.** *A* $\mathsf{BishopIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, S, Q, W, H, C, P, N)$$

with the following displayed rows:

1. *L and R are paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for the left and right interval endpoints in each displayed stage.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window listing the inspected interval stages and endpoint observations.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the endpoint windows as regular rational data before any Real-facing seal is read.*
4. *W is the finite nesting ledger, recording endpoint order, containment of each displayed successor interval in its predecessor, and the local diameter budget.*
5. *H stores componentwise* hsame *transport for the endpoint rows, stream windows, regular readback row, nesting ledger, replay row, provenance row, and local name.*
6. *C stores displayed* Cont *replay from the endpoint observations through the nesting ledger and regular readback boundary.*
7. *P records* Pkg *provenance over* $\mathsf{BishopIntervalNesting}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNesting}^{\uparrow}}$ *row.*

The packet contains no coordinate for quotient stream equality, host real equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6958.2 (Bishop interval nesting NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalNesting}^{\uparrow}$ *carrier* $I = (L, R, S, Q, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNesting}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint order and containment, finite-window nesting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, structural transport, displayed replay, provenance, and local naming. Every obligation is read through* $L, R, S, Q, W, H, C, P, N$ *and through the sibling rows named in chapter 291, chapter 126, chapter 301, and chapter 38.*

The certificate exports only the finite nested-interval route. It does not export quotient streams, host real equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6958.1. The accepted packet displays the dyadic endpoint rows L, R, the stream window S, the regular rational readback Q, the nesting ledger W, and the structural rows H, C, P, N.

Endpoint order and containment are read from W after the dyadic endpoint rows have been exposed. Thus every displayed successor interval is compared inside the same finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, before any terminal real seal can be consumed.

The finite-window nesting clause uses S to restrict the argument to the inspected stages. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q repacks only those displayed windows, and the $\mathsf{Real}^{\uparrow}$ handoff can read the resulting regular data only through Q and the replay row C.

Finally, H, P, and N certify that transport, provenance, and local naming preserve the same packet. Since no row stores quotient stream equality, host real equality, selected limits, countable choice, trichotomy, or ambient completeness, none of those principles is a local obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNesting}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopIntervalNesting}^{\uparrow}}$, seed)

Scope closed: Definition 6958.1 and Definition 6958.2 seed the finite dyadic endpoint, stream-window, regular-readback, and Real-seal route for constructive interval nesting.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopIntervalNesting}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host real equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked endpoint order, containment, finite-window nesting, regular readback, Real seal handoff, transport, replay, provenance, and local naming for the same carrier.

Constructive story: A $\mathsf{BishopIntervalNesting}^{\uparrow}$ packet is generated from dyadic left and right endpoint rows, finite StreamName windows, RegSeqRat readback rows, a nested containment ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6959

A Concrete Naming Certificate for $\mathsf{CompactFiniteLowerBound}^{\uparrow}$

Following Bishop and Bridges on the finite-cover proof of uniform continuity on compact metric spaces, the $\mathsf{CompactFiniteLowerBound}^{\uparrow}$ packet records the lower-bound fold that turns a compact finite-net radius family into one classifier-stable radius row. It sits over the compact carrier of chapter 60, the metric distance surface of chapter 57, the pointwise continuity rows of chapter 59, the rational order readback of chapter 37, and the $\mathsf{ProbeBundle}$ fold used by Definition 9237.5. The packet specializes the lower-bound handoff of chapter 4096 to compact-uniform-continuity consumers.

Definition 6959.1 (Compact finite lower-bound carrier)**.** *A* $\mathsf{CompactFiniteLowerBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, M, F, \mathcal{P}, R, O, A, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row whose displayed cover supplies the centers and coverage witnesses.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source-distance row used by the compact cover and later uniform-modulus consumer.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row attached to the same source metric.*
4. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine indexing the finite net.*
5. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *radius rows carried by the bundled probes.*
6. *O is the total-order finite-minimum readback row comparing the current accumulator with the next displayed radius.*
7. *A is the accumulator history produced by primitive recursion over $\mathcal{P}$.*
8. *L is the exported lower-bound row, classifier-stably below every displayed radius in R.*
9. *H stores componentwise* hsame *transport for the compact, metric, continuous, bundle, rational, order, accumulator, lower-bound, replay, provenance, and local-name rows.*
10. *C stores displayed* Cont *replay from compact finite-net data and pointwise-modulus rows through the rational fold to the lower-bound export.*

11. *P records* Pkg *provenance over* $\mathsf{CompactFiniteLowerBound}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteLowerBound}^{\uparrow}}$ *row.*

The packet has no row for an external selector, host minimum, completed cover, countable choice, quotient metric equality, or ambient compactness theorem.

Theorem 6959.2 (Compact finite lower-bound NameCert obligations)**.** *For every accepted* CompactFiniteLowerBou *carrier $B = (K, M, F, \mathcal{P}, R, O, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteLowerBound}^{\uparrow}}$ *surface consists exactly of bundle admission, nonempty seed, successor comparison, lower-bound export, classifier stability, structural transport, displayed replay, provenance, and local naming. The successor comparison consumes the* $\mathsf{Rat}^{\uparrow}$ *total-order readback row O at each displayed probe, and the export row L is available only after the finite* ProbeBundle *recursion has consumed the compact finite-net row K, metric source row M, continuous pointwise-modulus row F, and radius family R.*

Thus compact uniform-continuity consumers may read the exported lower bound as finite BEDC data, but they may not import an external selector, host minimum, completed cover, countable choice, quotient metric equality, or ambient compactness theorem.

Proof. Bundle admission projects K and $\mathcal{P}$ from Definition 6959.1. The nonempty seed is part of the displayed compact finite-net row, so the fold starts from a named radius history rather than from a host default.

At a successor probe, the row O compares the current accumulator with the next $\mathsf{Rat}^{\uparrow}$ radius in R. The accumulator A is updated by primitive recursion over the displayed ProbeBundle spine, so no outside enumeration or choice functional is used.

The lower-bound export L is then read from the completed finite recursion. Componentwise transport in H and replay in C preserve the compact, metric, continuous, rational, order, accumulator, and lower-bound rows along one packet, while P and N record the same source and local name.

Classifier stability follows because every comparison in O and every accumulator update in A is attached to the same finite bundle and radius family. Since the carrier contains no row for an external selector, host minimum, completed cover, countable choice, quotient metric equality, or ambient compactness theorem, none can appear as a local obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFiniteLowerBound}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactFiniteLowerBound}^{\uparrow}}$, seed)

Scope closed: Definition 6959.1 and Definition 6959.2 seed the finite compact-net lower-bound fold over Compact, Metric, Continuous, Rat, and ProbeBundle rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactFiniteLowerBound}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an external selector, host minimum, completed cover, countable choice, quotient metric equality, ambient compactness theorem, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked bundle admission, nonempty seed, successor comparison, lower-bound export, classifier stability, transport, replay, provenance, and local naming for the same finite fold packet.

Constructive story: A $\mathsf{CompactFiniteLowerBound}^{\uparrow}$ packet is generated from compact finite-net data, a Metric source row, Continuous pointwise-modulus rows, positive Rat radius rows, a ProbeBundle spine, total-order finite-minimum readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6960

A Concrete Naming Certificate for $\mathsf{RegularCauchyReindexing}^{\uparrow}$

Following Bishop and Bridges on cofinal subsequences of regular Cauchy names, the $\mathsf{RegularCauchyReindexing}^{\uparrow}$ packet records the finite route by which a regular Cauchy name may be read along a displayed monotone cofinal index map while preserving the regular rational and Real-facing rows. It sits beside the general reindexing surface of chapter 1568, the finite source and target schedules of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a textbook cofinal-reindexing certificate for regular Cauchy data; it is not quotient stream equality, not a selected subsequence limit, not countable choice, not host real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6960.1 (Regular Cauchy reindexing carrier)**.** *A* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, I, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row naming the original regular Cauchy observation schedule.*
2. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *row naming the reindexed observation schedule presented to downstream consumers.*
3. *I is the monotone-cofinal index route, recording the finite unary map from target requests into source requests and the cofinal witness needed for tail coverage.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the displayed precision cells, slack conversions, and comparison budgets used while the target schedule reads the source.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row reached only after the source schedule, target schedule, index route, and dyadic ledger have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after R has been read through the reindexed schedule.*
7. *H records componentwise* hsame *transport for $S, T, I, D, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a target precision request through I, the source schedule S, the dyadic ledger D, the regular rational readback R, and the Real seal E.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReindexing}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6960.2 (Regular Cauchy reindexing NameCert obligations)**.** *For every accepted* RegularCauchyReindexing *carrier* $K = (S, T, I, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexing}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{StreamName}^{\uparrow}$ *row, target* $\mathsf{StreamName}^{\uparrow}$ *row, monotone-cofinal index route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The obligations require the target schedule to reach the source schedule through I before any dyadic tolerance, regular readback, or Real seal is consumed.*

Proof. Project the carrier of Definition 6960.1. The displayed rows S, T, I, D, R, E are the only analytic coordinates, and the structural rows H, C, P, N only preserve and name that route.

The target schedule T cannot read a source observation directly. It first consumes the monotone-cofinal route I, which supplies the target-to-source request map and the finite tail-coverage witness needed for cofinal regular Cauchy reads.

After I has fixed the source request, the dyadic ledger D records the tolerance budget for the selected source window. The regular readback R and the Real seal E are downstream of that same ledger, so they cannot supply quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 6960.3 (Regular Cauchy reindexing Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *carrier factors through the finite route*

$$T \longmapsto I \longmapsto S \longmapsto D \longmapsto R \longmapsto E.$$

The route is the regular-Cauchy specialization of the general $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff: it keeps both* $\mathsf{StreamName}^{\uparrow}$ *schedules visible, checks the cofinal target request by the dyadic tolerance ledger, reads the regular rational row, and only then exposes the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient stream equality, no selected subsequence limit, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the seal row E. The carrier places E after the regular readback row R, and R is available only after the target request has been routed through I to the source schedule S.

The tolerance ledger D is read on the same route. It supplies the finite precision and slack data needed for the selected source observation, so the target schedule cannot bypass the displayed source window or invent a separate regular Cauchy name.

The provenance row P records the relationship with the general $\mathsf{CauchyReindexing}^{\uparrow}$ surface, while H, C, N preserve and name the same route. Therefore every Real-facing read factors through the displayed finite packet and receives none of the excluded host or quotient objects. □

Theorem 6960.4 (Regular Cauchy cofinal stream factorization)**.** *Every accepted* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *carrier* $K = (S, T, I, D, R, E, H, C, P, N)$ *exposes its target schedule first as a* $\mathsf{StreamReindexing}^{\uparrow}$ *finite-window read through chapter 7081 and then as a* $\mathsf{CofinalReindexing}^{\uparrow}$ *tail-window read through chapter 7209. Only after those sibling surfaces are displayed may the consumer read the* $\mathsf{RegSeqRat}^{\uparrow}$ *row of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

Proof. Project K along the carrier rows in Definition 6960.1. The target schedule T is the finite window surface presented to downstream consumers, and the index route I identifies how each target request is routed back to the source schedule S.

Read I as the cofinal target-to-source route. Its monotone tail coverage is exactly the sibling surface needed by the $\mathsf{CofinalReindexing}^{\uparrow}$ packet, while the finite target-window request is the sibling surface read by $\mathsf{StreamReindexing}^{\uparrow}$.

The rows D, R, E are downstream of this schedule. The dyadic ledger D audits the selected source window, the regular rational row R reads it as $\mathsf{RegSeqRat}^{\uparrow}$ data, and the seal row E is exposed only after that readback. Thus no accepted carrier can bypass the StreamReindexing and CofinalReindexing sibling route on its way to the regular readback and Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReindexing}^{\uparrow}$ packet is generated from source and target StreamName rows, a monotone-cofinal index route, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReindexing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6960.1, Definition 6960.2, and Definition 6960.3 seed the finite regular Cauchy cofinal-reindexing route from displayed source and target schedules to the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReindexing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-cofinal route exposure, dyadic tolerance exactness, regular readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyReindexi

Chapter 6961

A Concrete Naming Certificate for $\mathsf{BanachInverseMapping}^{\uparrow}$

Following Banach and the Bishop–Bridges treatment of constructive functional analysis, the $\mathsf{BanachInverseMapping}^{\uparrow}$ packet records the finite route by which a displayed bounded linear bijection between Banach rows is read as a bounded inverse with explicit inverse-modulus data. It sits beside the complete source and target surfaces of chapter 90 and chapter 128, the bounded-operator route of chapter 1644, the graph and open-image routes of chapter 1537 and chapter 1538, and the nearby inverse theorem packet of chapter 2291. Its sibling routes include the locally convex and Frechet carriers of chapter 3386 and chapter 3495, the fixed-point endpoint discipline of chapter 396, and the continuous-map graph surface of chapter 130. The chapter names only displayed Banach endpoint rows, a bounded linear bijection graph, inverse-modulus ledgers, transport, replay, provenance, and local naming; it does not import a host inverse function, quotient operator equality, choice-selected preimages, Hahn–Banach separation, or ambient completeness outside the carried rows.

Definition 6961.1 (Banach inverse-mapping carrier)**.** *A* $\mathsf{BanachInverseMapping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, F, I, M, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, complete-metric, and source-window ledgers used before an inverse request is admitted.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, complete-metric, and target-window ledgers used by output-side consumers.*
3. *F is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *graph row from X to Y, carrying endpoint action, linearity, graph compatibility, norm-bound data, and bounded-operator provenance over the same endpoints.*
4. *I is the bijection and inverse-graph row: its injective side routes equal target observations through the source classifier of X, and its surjective side records target windows with displayed source preimage windows.*
5. *M is the inverse-modulus ledger. For each target tolerance and target-window row, it exposes a finite source tolerance, an operator-bound comparison, and the open-image or graph route needed before the inverse row is public.*

6. *H records componentwise* hsame *transport for X, Y, F, I, M and for the structural rows that carry inverse-modulus requests.*
7. *C records* Cont *replay from an inverse request through the source and target Banach rows, the bounded-linear graph, the bijection row, the inverse-modulus ledger, and the sibling graph and complete-metric routes.*
8. *P is the* Pkg *provenance over* $\mathsf{BanachInverseMapping}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{InverseMappingTheorem}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachInverseMapping}^{\uparrow}}$ *row.*

The carrier has no coordinate for an external inverse function, unrestricted operator quotient, selected source representative, nonconstructive open-map principle, or completeness datum not already displayed by X and Y.

Theorem 6961.2 (Banach inverse-mapping NameCert obligations)**.** *For an accepted* $\mathsf{BanachInverseMapping}^{\uparrow}$ *carrier $Q = (X, Y, F, I, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachInverseMapping}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete source and target rows, bounded linear bijection graph exposure through F and I, inverse-modulus exposure through M, componentwise* hsame *transport through H,* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every inverse-facing consumer must read the same displayed X, Y, F, I, M rows before a bounded inverse row is public.*

Proof. Project the carrier. The source and target rows are X and Y, the bounded graph row is F, the bijection and inverse-graph row is I, and the inverse-modulus ledger is M.

The NameCert obligations are rowwise. Endpoint agreement requires F, I, and M to read the same Banach source and target rows. Transport by H preserves those endpoint rows, while replay by C exposes them in the order used by inverse-facing consumers.

The sibling routes do not add hidden coordinates. The Frechet and locally convex chapters provide neighbouring complete linear-topology reads, the fixed-point and continuous-map chapters provide endpoint and graph disciplines, and the complete-metric chapter supplies only the displayed completeness surface already present in X and Y.

Thus the local NameCert surface is exactly the finite route carried by Q. □

Theorem 6961.3 (Banach inverse-mapping complete source route)**.** *Every public read of a* $\mathsf{BanachInverseMapping}^{\uparrow}$ *inverse-modulus row must pass through the complete source row X and complete target row Y before it can inspect F, I, or M. In particular, a consumer cannot replace the Banach source route by a merely locally convex, Frechet, continuous-map, or fixed-point sibling route.*

Proof. The carrier definition places X and Y before the bounded graph, bijection, and inverse-modulus rows. Any replay of the inverse request through C therefore starts from the displayed Banach endpoints.

The sibling references are dependency routes rather than substitutes. $\mathsf{FrechetSpace}^{\uparrow}$ and $\mathsf{LocallyConvexSpace}^{\uparrow}$ can supply neighbouring linear-topology data, $\mathsf{ContinuousMap}^{\uparrow}$ can inspect graph continuity, and $\mathsf{BanachFixedPoint}^{\uparrow}$ can inspect fixed-point endpoints, but none of those rows contains both Banach endpoint completeness ledgers for this operator.

Hence an accepted inverse-modulus read must expose the complete source and target rows before it reaches F, I, or M. □

Theorem 6961.4 (Banach inverse-modulus non-escape)**.** *For an accepted* $\mathsf{BanachInverseMapping}^{\uparrow}$ *carrier, the inverse-modulus ledger* M *cannot be read as a host inverse or as a selected preimage function. It is public only as a finite route from target tolerance rows through* Y, I, F, X, H, C, P, N *to displayed source-window and operator-bound data.*

Proof. Start with the inverse-modulus row M. Its entries are tolerance requests, source windows, norm-bound comparisons, and graph or open-image handoffs; no entry is a host inverse function or an unrestricted selector.

Surjective information is carried by I and boundedness information by F. The ledger M can combine those rows only after the source and target Banach rows have been displayed, and transport and replay preserve that same finite order through H and C.

Finally P and N record provenance and local naming for the displayed route. Since the carrier has no coordinate for selected representatives or quotient operator equality, the inverse-modulus data cannot escape into a host inverse object. □

Falsifiable prediction: A consumer that reads a bounded inverse for a Banach bijection without passing through the displayed inverse-modulus row M is not reading an accepted $\mathsf{BanachInverseMapping}^{\uparrow}$ packet.

Independence witness: $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ are non-bijective siblings for this packet: none alone carries the complete finite combination of Banach endpoints, bounded linear bijection graph, inverse-modulus ledger, transport, replay, provenance, and local NameCert data exposed by $\mathsf{BanachInverseMapping}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachInverseMapping}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachInverseMapping}^{\uparrow}$ packet is generated from complete Banach source and target rows, a bounded-linear graph, bijection and inverse-graph data, an inverse-modulus ledger, componentwise hsame transport, Cont replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachInverseMapping}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6961.1, Definition 6961.2, Definition 6961.3, and Definition 6961.4 seed the finite inverse-modulus route for displayed Banach bounded bijections.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachInverseMapping}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host inverse functions, quotient operator equality, selected preimages, Hahn–Banach separation, or inverse mapping for non-Banach carriers.

Upgrade path: Obligation closure requires checked carrier admission, endpoint agreement, bounded bijection graph exposure, inverse-modulus replay, transport stability, provenance, and non-escape for BEDC.Derived.BanachInverseMappingUp.

Chapter 6962

A Concrete Naming Certificate for $\mathsf{LocatedIntervalOrder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalOrder}^{\uparrow}$ packet records comparison of located intervals by displayed endpoint windows and finite tolerance budgets, not by a host trichotomy principle for completed real endpoints. It sits beside the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located bisection route of chapter 1711. The packet gives Real and Completion consumers a Bishop-style interval comparison surface without importing classical order trichotomy, open-cover compactness, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6962.1 (LocatedIntervalOrder carrier)**.** *A* $\mathsf{LocatedIntervalOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, R, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the left located-interval endpoint row, containing the displayed lower and upper endpoint windows for the first interval.*
2. *R is the right located-interval endpoint row, containing the displayed lower and upper endpoint windows for the second interval.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row, recording the finite dyadic tolerance at which the endpoint windows are compared.*
4. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule through which the two endpoint rows are inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows, followed by the terminal* $\mathsf{Real}^{\uparrow}$ *seal used by Real-facing consumers.*
6. *H records componentwise* hsame *transport for L, R, D, S, Q and the structural rows.*
7. *C records displayed* Cont *replay of the comparison route through the same finite rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for an exhaustive real-order decision, open-cover compactness, quotient equality of endpoint names, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6962.2 (LocatedIntervalOrder obligation surface)**.** *For an accepted* $\mathsf{LocatedIntervalOrder}^{\uparrow}$ *carrier* $O = (L, R, D, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the two endpoint rows, dyadic comparison-budget coverage, shared finite-window inspection, regular rational readback with terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every interval-order read must consume those rows before any downstream Real or Completion consumer may inspect the comparison.*

Proof. Project the carrier of Definition 6962.1. The only interval data available to the consumer are the two displayed endpoint rows L and R.

The comparison step is forced through D and S. The dyadic row supplies the finite tolerance budget, while the shared stream row records exactly which endpoint windows are inspected on both sides.

The readback and sealing step is forced through Q. Componentwise transport, replay, provenance, and local naming are then carried by H, C, P, N, and none of these structural rows adds a second order principle. Thus the local NameCert surface is exhausted by the displayed packet. □

Theorem 6962.3 (LocatedIntervalOrder bisection compatibility)**.** *Let* $O = (L, R, D, S, Q, H, C, P, N)$ *be an accepted* $\mathsf{LocatedIntervalOrder}^{\uparrow}$ *packet. If a* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *consumer refines either displayed interval endpoint row by a retained finite child interval, then the interval-order read remains valid exactly when the refined endpoint windows are replayed through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row* D*, shared* $\mathsf{StreamName}^{\uparrow}$ *window row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and structural rows* H, C, P, N*. The bisection handoff exports no selected infinite branch and no classical trichotomy of real endpoints.*

Proof. Begin with the accepted order packet and the displayed bisection refinement. The bisection route supplies only a finite retained-child row, so it can replace an endpoint window only by exhibiting the corresponding finite endpoint data.

Apply the obligation surface of Definition 6962.2. The order read still has to pass through D, S, Q after the refinement, because those rows are the only comparison budget, observation window, and readback route in the carrier.

Finally transport the refined endpoint row through H and replay the same route through C. Provenance and local naming remain P, N, so the bisection consumer receives a compatible finite comparison packet, not a hidden branch selector or a trichotomy witness. □

Theorem 6962.4 (LocatedIntervalOrder ledger exhaustion)**.** *The complete ledger of an accepted* $\mathsf{LocatedIntervalOrder}^{\uparrow}$ *packet is the finite route*

$$L, R, D, S, Q, H, C, P, N.$$

Every order-facing, bisection-facing, Real-facing, or Completion-facing consumer of the packet is exhausted by the displayed endpoint rows, dyadic comparison budget, shared stream window, regular rational readback, terminal seal, transport, replay, provenance, and local naming. No consumer may read trichotomy, open-cover compactness, quotient equality, selected limits, or $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Use the obligation surface to account for carrier admission, finite comparison budget, shared window inspection, readback, seal, transport, replay, provenance, and naming. These are exactly the rows listed in the carrier.

Use bisection compatibility for interval-refinement consumers. Any retained child interval must still replay through the same comparison-budget, window, readback, and structural rows.

The carrier contains no further coordinate. Therefore the ledger is exhausted by $L, R, D, S, Q, H, C, P, N$, and the excluded order, compactness, quotient, selected-limit, and completeness principles have no route into an accepted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalOrder}^{\uparrow}$ packet is generated from two located endpoint rows, a DyadicRatCore comparison budget, a shared StreamName window, RegSeqRat readback with Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6962.1, Definition 6962.2, Definition 6962.3, and Definition 6962.4 seed the finite located interval comparison surface for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close trichotomy, open-cover compactness, quotient equality, selected limits, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic comparison-budget coverage, shared window inspection, readback and seal non-escape, bisection compatibility, transport, replay, provenance, and ledger exhaustion for BEDC.Derived.LocatedIntervalOrderUp.

Chapter 6963

A Concrete Naming Certificate for $\mathsf{ChoquetCapacity}^{\uparrow}$

Following Choquet and classical capacity theory, the $\mathsf{ChoquetCapacity}^{\uparrow}$ packet records finite capacity data over a compact measurable source without promoting the host outer-measure construction to a primitive. It sits beside the compact carrier of chapter 60, the measure interface of chapter 92, the Radon-measure route of chapter 2263, the Choquet-simplex boundary of chapter 6611, and the Choquet-integral consumer of chapter 6680. The packet names only displayed monotonicity, continuity-window, compact-exhaustion, polar-set, transport, replay, provenance, and local naming rows; it does not assert a host capacitability theorem, completed outer measure, quotient set algebra, countable-choice selector, or ambient measure completion.

Definition 6963.1 (Choquet capacity carrier)**.** *A* $\mathsf{ChoquetCapacity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, M, R, S, I, O, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *support row, exposing the compact source on which finite compact windows and exhaustion witnesses are read.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *measurable-event row, recording only the event family and finite value reads used by the capacity ledger.*
3. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *regularity row, available when the capacity read is induced by a regular measure route.*
4. *S is the* $\mathsf{ChoquetSimplex}^{\uparrow}$ *sibling boundary row, recording that probability-boundary barycentric data are adjacent but not collapsed into capacity equality.*
5. *I is the* $\mathsf{ChoquetIntegral}^{\uparrow}$ *consumer row, recording that integral reads may consume the capacity only after the capacity rows are displayed.*
6. *O is the outer-approximation ledger, giving finite open-window majorants and compact-window minorants without introducing an outer-measure primitive.*
7. *W is the continuity-window ledger, recording finite monotonicity, continuity-from-below, and continuity-from-above checks on the displayed windows.*
8. *H records componentwise* hsame *transport for K, M, R, S, I, O, W and the local naming row.*

9. *C records displayed* Cont *replay from compact and measurable source rows into regularity, outer-approximation, continuity-window, integral-consumer, and local naming reads.*
10. *P is the* Pkg *provenance over* $\mathsf{ChoquetCapacity}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{ChoquetSimplex}^{\uparrow}$, $\mathsf{ChoquetIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoquetCapacity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, M, R, S, I, O, W, H, C, P, N$. It has no coordinate for a completed outer-measure algebra, a host capacitability theorem, selected analytic exhaustion, quotient equality of measurable sets, countable choice, or ambient measure completion.

Theorem 6963.2 (Choquet capacity obligation surface)**.** *For every accepted* $\mathsf{ChoquetCapacity}^{\uparrow}$ *carrier $Q = (K, M, R, S, I, O, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoquetCapacity}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact, measure, Radon, Choquet-simplex boundary, Choquet-integral consumer, outer-approximation, continuity-window, transport, replay, provenance, and local naming rows; monotonicity exposure through W; compact-window accountability through K, O; measure and Radon regularity exposure through M, R; sibling separation through S, I; and structural preservation through H, C, P, N.*

Proof. Project the accepted packet $Q = (K, M, R, S, I, O, W, H, C, P, N)$. The compact source row K and measurable-event row M give the finite source surface on which a capacity can be read.

Use R, O, W for the capacity-specific obligations. The Radon row supplies the regular-measure route when present, the outer-approximation ledger gives displayed majorants and minorants, and the continuity-window ledger records monotonicity and finite continuity checks.

The rows S and I keep the adjacent Choquet simplex and Choquet integral routes visible without identifying either one with the capacity carrier. Thus barycentric probability data and integral readback remain sibling or consumer surfaces rather than hidden components of the classifier.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since the carrier has no completed outer-measure algebra, host capacitability theorem, quotient set equality, selected exhaustion, choice schedule, or ambient measure-completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6963.3 (Choquet capacity consumer handoff)**.** *Every* $\mathsf{ChoquetIntegral}^{\uparrow}$ *or* $\mathsf{ChoquetSimplex}^{\uparrow}$ *consumer of an accepted* $\mathsf{ChoquetCapacity}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{Compact}^{\uparrow}, \mathsf{Measure}^{\uparrow} \longmapsto \mathsf{RadonMeasure}^{\uparrow} \longmapsto O, W \longmapsto \mathsf{ChoquetCapacity}^{\uparrow} \longmapsto \mathsf{ChoquetIntegral}^{\uparrow}, \mathsf{ChoquetSimplex}^{\uparrow}.$$

The exported row is a displayed capacity handoff over compact and measurable windows. It is not an outer-measure primitive, a host capacitability theorem, a quotient event algebra, a selected representing measure, or an ambient completion principle.

Proof. Start with K and M. They expose the compact source and measurable-event surface before any capacity value or consumer route is available.

Next read R, O, W. The regularity row, outer-approximation ledger, and continuity-window ledger provide the finite capacity evidence; none of these rows selects an external exhaustion or invokes a completed outer measure.

Only then may a consumer reach I or compare with the sibling boundary S. The structural rows H, C, P, N preserve exactly that route, so the public handoff cannot bypass the compact, measure, Radon, approximation, and continuity rows. □

Falsifiable prediction: Every accepted $\mathsf{ChoquetCapacity}^{\uparrow}$ public read exposes Compact, Measure, RadonMeasure, outer-approximation, continuity-window, ChoquetIntegral, ChoquetSimplex, transport, replay, provenance, and local NameCert rows before any capacity consumer is public.

Independence witness: compact, measure, radon-measure, choquet-simplex, choquet-integral: no listed sibling supplies the joint finite capacity packet containing compact source, measurable-event, Radon regularity, outer-approximation, continuity-window, integral-consumer, simplex-boundary, transport, replay, provenance, and local NameCert rows while refusing outer-measure primitives, host capacitability, quotient set equality, selected exhaustion, countable choice, and ambient measure completion.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetCapacity}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoquetCapacity}^{\uparrow}$ packet is generated from Compact, Measure, RadonMeasure, ChoquetSimplex, ChoquetIntegral, outer-approximation, continuity-window, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoquetCapacity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6963.1, Definition 6963.2, and Definition 6963.3 seed the finite capacity route over compact source rows, measurable-event rows, Radon regularity, outer approximations, continuity windows, Choquet integral consumers, Choquet simplex boundaries, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoquetCapacity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host capacitability theorem, outer-measure completion, quotient event equality, selected analytic exhaustion, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window accountability, measure and Radon exposure, outer-approximation and continuity-window routing, sibling separation, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetCapacityUp.

Chapter 6964

A Concrete Naming Certificate for $\mathsf{RegularCauchyConstant}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyConstant}^{\uparrow}$ packet records the constant regular Cauchy name generated from one $\mathsf{DyadicRatCore}^{\uparrow}$ row through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and a $\mathsf{Real}^{\uparrow}$ seal. It sits beside the dyadic row of chapter 291, the finite windows of chapter 126, the regular rational handoff of chapter 301, the terminal real seal of chapter 38, and the algebraic sum route of chapter 417.

Definition 6964.1 (Regular Cauchy constant carrier)**.** *A* $\mathsf{RegularCauchyConstant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the source* $\mathsf{DyadicRatCore}^{\uparrow}$ *row that supplies the single dyadic value, scale, and comparison ledger.*
2. *W is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule that repeats the same dyadic source through every displayed finite window.*
3. *E is the repeated dyadic observation ledger, recording that each visible window reads the source row D rather than a moving sequence of host rationals.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the constant stream windows and the repeated dyadic ledger.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, W, E, R have been displayed.*
6. *H stores componentwise* hsame *transport for the dyadic source, constant window schedule, observation ledger, regular readback, Real seal, continuation row, provenance, and local naming.*
7. *C stores displayed* Cont *replay from the dyadic source to each constant window, from the windows to the regular readback, and from the readback to the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyConstant}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyConstant}^{\uparrow}}$ *row.*

The classifier compares accepted constant-name carriers only through these nine rows. It contains no coordinate for quotient streams, selected host limits, host equality, countable choice, or ambient completeness.

Theorem 6964.2 (Regular Cauchy constant NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyConstant}^{\uparrow}$ *carrier* $K = (D, W, E, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyConstant}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed nine-row packet;*
2. *constant-window exactness, stating that every displayed* $\mathsf{StreamName}^{\uparrow}$ *window in* W *reads the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row* D*;*
3. *repeated-observation exactness, stating that* E *records no observation outside the dyadic source and its visible windows;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff, stating that* R *is read from* D, W, E *before any Real-facing seal is available;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, stating that* S *exports no quotient stream, selected limit, host equality, countable-choice row, or ambient completeness witness;*
6. *componentwise transport, replay, provenance, and local naming for* H, C, P, N*.*

Proof. Project the carrier of Definition 6964.1. The dyadic source, constant window schedule, repeated observation ledger, regular readback, Real seal, transport, continuation, provenance, and local naming rows are exactly the classifier-visible coordinates.

The constant-window obligation follows by reading W only through the source row D. The observation ledger E then records repeated readings of the same dyadic value, so no moving host sequence or off-carrier rational stream is introduced.

The regular handoff is the next displayed row. Since R is available only after D, W, E have been read, any Real-facing consumer must pass through the regular rational readback before reaching S.

The transport, replay, provenance, and naming rows preserve this same nine-row packet. Because the carrier has no quotient, selected-limit, host-equality, choice, or completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 6964.3 (Regular Cauchy constant Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyConstant}^{\uparrow}$ *carrier* $K = (D, W, E, R, S, H, C, P, N)$ *reads the dyadic source* D*, constant window schedule* W*, repeated dyadic observation ledger* E*, regular readback* R*, Real seal* S*, componentwise transport* H*, replay row* C*, provenance* P*, and local naming row* N*. The consumer receives no quotient stream, selected limit, host equality, or ambient completeness.*

Proof. Begin with a Real-facing consumer and project the accepted carrier. The only source value available is the dyadic row D, and the only sequence-facing schedule is the constant $\mathsf{StreamName}^{\uparrow}$ row W.

The repeated observation ledger E records the finite window reads of D, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is read back from those displayed observations. The continuation row C replays this route without inserting an additional stream or limit selector.

The seal row S is reached only after D, W, E, R have been displayed. Transport through H and provenance through P preserve the same packet, while N records the local naming boundary.

Thus every Real-facing read factors through $D, W, E, R, S, H, C, P, N$. Since the carrier has no quotient stream, selected limit, host equality, or ambient completeness row, none is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyConstant}^{\uparrow}}$**.**

Constructive story: A RegularCauchyConstant$^{\uparrow}$ packet is generated from one DyadicRatCore source row, a constant StreamName schedule, repeated dyadic observations, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyConstant}^{\uparrow}}$, seed)

Scope closed: Definition 6964.1, Definition 6964.2, and Definition 6964.3 declare the finite carrier, its local NameCert obligations, and the Real seal handoff boundary for constant regular Cauchy names.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyConstant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Formal closure requires a Lean carrier and checked NameCert obligations for the same constant regular Cauchy packet.

Chapter 6965

Bishop Real Window NameCert Construction

Following Bishop–Bridges constructive real analysis, the $\mathsf{BishopRealWindow}^{\uparrow}$ packet records the finite rational windows through which a regular real name is inspected before a Real or Completion consumer reaches a terminal seal. It sits over the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite readback interface for one displayed real-name route; it is not quotient equality of real names, a selected limit, a choice schedule, or an ambient completeness principle.

Definition 6965.1 (Bishop real window carrier)**.** *A* $\mathsf{BishopRealWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected prefix and request schedule.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the displayed window.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the rational cells, endpoint budgets, and comparison slack used by the window.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after* S, R, D *have been replayed.*
5. H *records componentwise* hsame *transport for the window, readback, tolerance, seal, provenance, and local naming rows.*
6. C *records displayed* Cont *replay from the finite window through regular rational readback and dyadic comparison into the Real seal.*
7. P *is the* Pkg *provenance over* $\mathsf{BishopRealWindow}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopRealWindow}^{\uparrow}}$ *row naming exactly this finite rational-window packet.*

The classifier compares accepted packets only through S, R, D, E, H, C, P, N*. It has no coordinate for quotient stream equality, selected real representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6965.2 (Bishop real window finite readback)**.** *For every accepted* $\mathsf{BishopRealWindow}^{\uparrow}$ *carrier* $W = (S, R, D, E, H, C, P, N)$*, every Real or Completion-facing finite rational-window read factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the consumer-visible rows are exactly S, R, D, E, H, C, P, N*: the inspected StreamName window is read as a regular rational sequence, checked against the dyadic tolerance ledger, transported and replayed through the local structural rows, and only then exposed as a Real seal.*

Proof. Project the carrier of Definition 6965.1. The only source window is S, the only regular rational readback is R, and the only tolerance ledger is D.

Read the route in order. A finite request first inspects the $\mathsf{StreamName}^{\uparrow}$ row S; the same displayed entries are then read through R as regular rational data; the dyadic comparisons required by the requested rational window are accounted for in D.

The rows H and C transport and replay that same finite route before the provenance and naming rows P, N admit the packet. The terminal row E is therefore exposed only after the window, readback, and dyadic ledger have all been displayed.

Since the carrier has no coordinate for quotient equality, selected limits, choice, host real equality, or ambient completeness, no consumer can obtain a second Real-facing read outside the displayed finite-window projection and regular-rational readback. □

Theorem 6965.3 (Bishop real window NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealWindow}^{\uparrow}$ *carrier* $W = (S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealWindow}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite packet* W*;*
2. $\mathsf{StreamName}^{\uparrow}$ *window locality through* S*;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback through* R*;*
4. $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and rational-cell exactness through* D*;*
5. $\mathsf{Real}^{\uparrow}$ *seal ordering through* E*, together with componentwise transport, displayed replay, provenance, and local naming through* H, C, P, N*.*

No additional NameCert obligation is exported for quotient real equality, selected representatives, choice schedules, host real equality, or ambient completeness.

Proof. Carrier admission is the finite packet displayed in Definition 6965.1. Each listed row is one of its coordinates, so the local name can inspect it directly.

The obligation order follows the readback theorem Definition 6965.2. The window row supplies the inspected finite data, the regular rational row supplies the readback, the dyadic row supplies the tolerance evidence, and the Real row is exposed only at the end of that route.

The structural rows H, C, P, N preserve and name the same displayed packet. Because the forbidden objects are not carrier coordinates, they cannot create extra NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealWindow}^{\uparrow}$ consumer can be audited by checking that its Real-facing read first displays a $\mathsf{StreamName}^{\uparrow}$ finite window, then a $\mathsf{RegSeqRat}^{\uparrow}$ readback, then a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopRealWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite window, regular rational readback,

dyadic tolerance ledger, Real seal, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealWindow}^{\uparrow}$ packet is generated from a StreamName finite-window row, a RegSeqRat regular rational readback row, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6965.1, Definition 6965.2, and Definition 6965.3 seed the finite rational-window readback route from L10 StreamName and RegSeqRat data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real names, selected limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName locality, RegSeqRat readback, DyadicRatCore tolerance exactness, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealWindowU

Chapter 6966

A Concrete Naming Certificate for $\mathsf{FractionalIdeal}^{\uparrow}$

Following Atiyah–Macdonald and Bourbaki on fractional ideals in number fields and Dedekind domains, the $\mathsf{FractionalIdeal}^{\uparrow}$ packet records the standard carrier that sits between the ordinary ideal surface and the ideal-class quotient surface. It reads the ambient ring from chapter 43, the ideal rows from chapter 81, the Dedekind-domain source from chapter 172, the number-field carrier from chapter 168, and the ring-of-integers boundary from chapter 169. It supplies the carried endpoints consumed by chapter 170.

Definition 6966.1 (Fractional ideal carrier)**.** *A* $\mathsf{FractionalIdeal}^{\uparrow}$ *carrier over a displayed Dedekind source is a finite* BHist *packet*

$$F = (R, I, D, K, O, P_0, Q, H, C, P, N)$$

with the following rows:

1. *R is the ambient* $\mathsf{Ring}^{\uparrow}$ *row and its multiplicative compatibility data.*
2. *I is the ordinary* $\mathsf{Ideal}^{\uparrow}$ *row that records additive closure and absorption over R.*
3. *D is the* $\mathsf{Dedekind}^{\uparrow}$ *source row, including the Noetherian, integrally closed, and nonzero-prime boundary used by the packet.*
4. *K is the* $\mathsf{NumField}^{\uparrow}$ *fraction-field row through which denominators are read.*
5. *O is the* $\mathsf{RingOfIntegers}^{\uparrow}$ *row that ties the integral carrier back to the number-field source.*
6. *P_0 is the principal fractional-ideal row, recording principal generators and the principal-equivalence boundary later consumed by* $\mathsf{IdealClass}^{\uparrow}$*.*
7. *Q is the denominator-clearance ledger, naming a nonzero denominator row whose multiplication carries the fractional endpoint back into the displayed* $\mathsf{Ideal}^{\uparrow}$ *surface.*
8. *H stores componentwise* hsame *transport for all rows above.*
9. *C stores displayed* Cont *replay for denominator clearance, principal-row reads, and ordinary ideal reads.*
10. *P is the* Pkg *provenance over* $\mathsf{FractionalIdeal}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{Ideal}^{\uparrow}$*,* $\mathsf{Dedekind}^{\uparrow}$*,* $\mathsf{NumField}^{\uparrow}$*,* $\mathsf{RingOfIntegers}^{\uparrow}$*, principal fractional ideals, denominator-clearance rows,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{FractionalIdeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, I, D, K, O, P_0, Q, H, C, P, N$. It has no coordinate for a detached fraction-field subset, a host localization object, quotient equality of fractional ideals, selected generators for arbitrary ideals, or an ambient algebraic-number-theory oracle.

Theorem 6966.2 (Fractional ideal NameCert obligations)**.** *For every accepted* $\mathsf{FractionalIdeal}^{\uparrow}$ *carrier $F = (R, I, D, K, O, P_0, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FractionalIdeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed ring, ideal, Dedekind, number-field, ring-of-integers, principal-row, denominator-clearance, transport, replay, provenance, and local naming rows; denominator-clearance exactness through Q; principal-row visibility through P_0; componentwise* hsame *transport through H; displayed* Cont *replay through C; and* Pkg *provenance through P.*

Proof. Project the carrier of Definition 6966.1. The algebraic source rows R, I, D, K, O display the ambient ring, ordinary ideal, Dedekind, number-field, and ring-of-integers data before any fractional endpoint can be read.

The principal row P_0 and the denominator-clearance ledger Q provide the two fractional-ideal specific fields. The first records the principal endpoints used by the later quotient classifier, and the second records the finite denominator operation that returns the endpoint to the ordinary ideal surface.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no detached fraction-field subset, host localization object, quotient equality, or generator-choice coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 6966.3 (Fractional ideal Dedekind handoff)**.** *Every* $\mathsf{IdealClass}^{\uparrow}$ *read of a fractional endpoint factors first through the* $\mathsf{FractionalIdeal}^{\uparrow}$ *carrier of Definition 6966.1: the* $\mathsf{Ring}^{\uparrow}$ *source of chapter 43, the* $\mathsf{Ideal}^{\uparrow}$ *surface of chapter 81, the* $\mathsf{Dedekind}^{\uparrow}$ *source of chapter 172, the* $\mathsf{NumField}^{\uparrow}$ *carrier of chapter 168, the* $\mathsf{RingOfIntegers}^{\uparrow}$ *boundary of chapter 169, principal fractional-ideal rows, denominator-clearance rows, and the structural* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Thus chapter 170 receives carried fractional endpoints and principal-boundary evidence, not a host quotient of detached representatives.*

Proof. Begin with the NameCert obligations of Definition 6966.2. They force every accepted fractional endpoint to expose the Dedekind, number-field, ring-of-integers, ordinary ideal, principal, and denominator-clearance rows before the endpoint can be named.

The IdealClass consumer then reads only the principal-boundary row and the carried endpoint row. Multiplication, inverse, identity, and principal equivalence can be routed through those displayed rows, while the structural rows preserve the same provenance under transport and replay.

Therefore an ideal-class packet cannot bypass $\mathsf{FractionalIdeal}^{\uparrow}$ by reading a detached fraction-field subset, a host localization, quotient equality of representatives, or selected generators. The handoff is exactly the finite Dedekind fractional-ideal route named above. □

Falsifiable prediction: Every $\mathsf{IdealClass}^{\uparrow}$ endpoint read factors through a $\mathsf{FractionalIdeal}^{\uparrow}$ carrier with displayed Ring, Ideal, Dedekind, NumField, RingOfIntegers, principal fractional-ideal, denominator-clearance, hsame, Cont, Pkg, and NameCert rows.

Independence witness: ring, ideal, dedekind, numfield, ring-of-integers, idealclass: $\mathsf{FractionalIdeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ordinary ideal data, fraction-field denominator clearance, principal fractional-ideal rows, and the structural handoff consumed by IdealClass.

Closure status for $\mathsf{NameCert}_{\mathsf{FractionalIdeal}^{\uparrow}}$.

Constructive story: A $\mathsf{FractionalIdeal}^{\uparrow}$ packet is generated from displayed Ring, Ideal, Dedekind, NumField, RingOfIntegers, principal fractional-ideal, denominator-clearance, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FractionalIdeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6966.1, Definition 6966.2, and Definition 6966.3 bind the finite fractional-ideal carrier consumed by IdealClass.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FractionalIdeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a public standard bridge, Lean verification, quotient equality of fractional ideals, a host localization object, or selected generators for arbitrary ideals.

Upgrade path: Obligation closure requires checked carrier admission, denominator-clearance exactness, principal-row visibility, and the Dedekind-to-IdealClass handoff for BEDC.Derived.FractionalIdealUp.

Chapter 6967

A Concrete Naming Certificate for $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$

Following Brunn and Minkowski on convex-body volume comparison, the $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ packet records the finite dyadic-grid surface on which a Brunn–Minkowski style read can be inspected before any host measure theory or arbitrary compact-convex-body theorem is invoked. It sits beside the real-name boundary of chapter 38, the compact source of chapter 60, the convex-set row of chapter 208, the polytope carrier of chapter 209, and the metric-distance interface of chapter 57. The packet names a finite grid certificate for displayed convexity and volume-proxy comparison; it is not general measure theory, not Lebesgue volume, not arbitrary compact convex bodies, and not a quotient equality of subsets.

Definition 6967.1 (Finite Brunn–Minkowski grid carrier)**.** *A* $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, A, B, S, L, V, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite dyadic grid row. It lists the inspected dyadic coordinate cells, their metric adjacency data, and the endpoint comparisons read through* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*
2. *A and B are the two displayed convex-grid subsets. Each row records membership decisions only for cells in K, with the convexity checks routed through* $\mathsf{ConvexSet}^{\uparrow}$ *and the finite polytope faces routed through* $\mathsf{Polytope}^{\uparrow}$*.*
3. *S is the grid Minkowski-sum classifier. For each displayed cell of K, it records whether that cell is admitted by a visible pair of cells from A and B under the finite addition table of the grid.*
4. *L is the compact bounding row, naming the finite bounding window supplied by* $\mathsf{Compact}^{\uparrow}$ *before any grid count is consumed.*
5. *V is the finite cardinal and volume-proxy ledger. It stores the displayed cell counts for A, B, and S, the common cell-volume scale, and the dyadic comparison rows used by the requested Brunn–Minkowski read.*
6. *W is the public comparison row. It exposes only the finite grid inequality request generated from K, A, B, S, L, V.*
7. *H records componentwise* hsame *transport for the grid, subset rows, sum classifier, compact bound, volume-proxy ledger, comparison row, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from a requested finite-grid comparison through K, A, B, S, L, V, W.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, A, B, S, L, V, W, H, C, P, N$. It has no coordinate for Lebesgue measure, arbitrary compact convex bodies, host subset equality, quotient completion, selected limits, countable choice, or ambient completeness.

Theorem 6967.2 (Finite Brunn–Minkowski grid NameCert obligations)**.** *For every accepted* $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ *carrier*

$$G = (K, A, B, S, L, V, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}}$ *surface consists exactly of finite dyadic-grid admission, displayed convex-subset admission for A and B, grid Minkowski-sum classification through S, compact bounding through L, finite cardinal and volume-proxy accounting through V, public comparison exposure through W, componentwise transport, displayed replay, package provenance, local naming, and non-escape from the displayed rows.*

Proof. Project the carrier of Definition 6967.1. The geometric rows are exactly the grid row K, the displayed subset rows A, B, the grid sum classifier S, the compact bounding row L, and the finite ledger rows V, W.

The certificate first fixes the finite grid. Metric adjacency and real endpoint comparisons are read through K, while A and B expose only those convex and polytope checks whose cells occur in that displayed grid.

The sum classifier is then determined inside the same finite table. The row S stores the visible pair witnesses for admitted sum cells, and L keeps the comparison inside the displayed compact bounding window.

Finally the ledger V records cardinal counts and the common dyadic cell-volume scale before W exposes the public comparison row. The structural rows H, C, P, N transport, replay, source, and name the same finite surface, so no measure-theoretic, quotient, choice, selected-limit, or ambient-completeness datum is exported. □

Theorem 6967.3 (Finite Brunn–Minkowski grid comparison route)**.** *Every public comparison read exported by an accepted* $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ *packet factors through the finite route*

$$K, A, B \longmapsto S \longmapsto L, V \longmapsto W \longmapsto H, C, P, N.$$

Thus the consumer receives a dyadic grid, two displayed convex-grid subsets, a finite Minkowski-sum classifier, a compact bounding window, a finite cardinal and volume-proxy ledger, and structural transport/replay rows before it reads the comparison boundary. It receives no general measure theory, no arbitrary compact-convex-body volume, no host subset equality, and no bridge-checked standard theorem.

Proof. Begin with an accepted packet $G = (K, A, B, S, L, V, W, H, C, P, N)$. The carrier lists no comparison row before the grid, subset, and finite sum-classifier rows have been displayed.

The compact and volume-proxy rows are also forced. The bounding row L supplies the finite compact window, while V supplies the only cardinal counts, cell-volume scale, and dyadic comparison rows available to the packet.

The public row W is therefore downstream of K, A, B, S, L, V. Componentwise hsame transport and displayed Cont replay carry exactly those rows through H and C, and P, N only source and name the same packet.

Since no carrier coordinate stores Lebesgue measure, arbitrary convex-body volume, quotient subset equality, selected limits, choice, or ambient completeness, the exported comparison cannot use such data. □

Falsifiable prediction: Every accepted $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ comparison read displays a finite dyadic grid, two convex-grid subset rows, a grid Minkowski-sum classifier, a compact bounding row, a finite cardinal and volume-proxy ledger, a public comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before it is consumed.

Independence witness: real, compact, convexset, polytope, metric: $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite dyadic grid, two convex-grid subset rows, a Minkowski-sum classifier, compact bounding data, cardinal and volume-proxy ledgers, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}$ packet is generated from a finite dyadic grid, two displayed convex-grid subset rows, a finite Minkowski-sum classifier, compact bounding data, finite cardinal and volume-proxy ledgers, a public comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6967.1, Definition 6967.2, and Definition 6967.3 seed the finite dyadic-grid convexity and volume-proxy certificate over $\mathsf{Real}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBrunnMinkowskiGrid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general measure theory, Lebesgue volume, arbitrary compact convex bodies, host subset equality, quotient completion, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, grid exactness, convex-grid membership routing, Minkowski-sum classifier soundness, compact-bound routing, finite ledger non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteBrunnMinkowskiGridUp.

Chapter 6968

A Concrete Naming Certificate for $\mathsf{CauchyProductSeries}^{\uparrow}$

Following Hardy on products of convergent series and Bishop–Bridges on constructive Cauchy reals, the $\mathsf{CauchyProductSeries}^{\uparrow}$ packet records the finite BEDC route by which two displayed regular Cauchy series expose a product-series window. It sits over the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite product route of chapter 2009. The packet names only finite convolution windows, tolerance budgets, transport, replay, provenance, and local certificate rows; it does not import a host theorem about infinite products of series.

Definition 6968.1 (Cauchy product series carrier)**.** *A* $\mathsf{CauchyProductSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, C, D, F, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *L and R are the two source series packets, each exposing only inspected* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *coefficient windows.*
2. *W is the finite window row naming the visible output indices and the paired source prefixes required for those indices.*
3. *C is the convolution ledger. For each displayed output index it records the finite diagonal sum of products of visible source coefficients.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row, carrying the requested precision, product slack, diagonal-sum slack, and comparison allowances.*
5. *F is the* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *row used for each displayed coefficient product before the finite diagonal sum is read.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the product-series window after the convolution and dyadic budget rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available to a downstream real or completion consumer only after Q has been read.*
8. *H records componentwise* hsame *transport for the source, window, convolution, budget, finite-product, readback, seal, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from source windows into coefficient products, from coefficient products into convolution rows, and from convolution rows through budgeted readback into the seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyProductSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, RegularCauchyFiniteProd BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductSeries}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $L, R, W, C, D, F, Q, E, H, T, P, N$. It has no coordinate for an unconditional rearrangement theorem, absolute convergence, analytic continuation, quotient stream equality, host multiplication of completed series, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductSeriesUp.CauchyProductSeriesTasteGate single carrier alignment`

Theorem 6968.2 (Cauchy product regular convolution closure)**.** *For every accepted* $\mathsf{CauchyProductSeries}^{\uparrow}$ *carrier $K = (L, R, W, C, D, F, Q, E, H, T, P, N)$, each displayed output window is a regular convolution window over the two source series packets. More precisely, reading Q for an output index visible in W factors through the finite diagonal entries listed in C, the coefficient-product rows listed in F, and the tolerance budget D before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback can be exported.*

Proof. Project the carrier of Definition 6968.1. The only source rows are L and R, and the only output indices visible to the packet are the indices named by W.

Proceed by induction over the finite window row W. The empty window exports no readback. For a successor window, the next output index exposes a finite diagonal list in C; each diagonal entry first reads the corresponding two source coefficients from L and R, then routes their product through the matching $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ cell in F.

The dyadic row D supplies the slack used to add the finitely many product reads in the displayed diagonal. The result is recorded in Q, and the seal E is downstream of that readback.

Rows H, T, P, N transport, replay, source, and name the same route. Since the route has no further coordinate, no host convolution theorem, quotient-series equality, selected limit, or rearrangement principle can be exported by this closure claim. □

Theorem 6968.3 (Cauchy product series NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductSeries}^{\uparrow}$ *carrier $K = (L, R, W, C, D, F, Q, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductSeries}^{\uparrow}}$ *surface consists exactly of carrier admission, two-source series exposure, finite output-window admission, convolution-ledger exposure, dyadic tolerance-budget exposure, finite coefficient-product routing, regular rational readback, terminal real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Read the rows of Definition 6968.1. Carrier admission exposes the tuple K, while L, R, W, C, D, F, Q, E give the data rows available to a consumer.

The certificate surface is exhausted by projection. A read of a source coefficient must pass through L or R; a read of a product coefficient must pass through F; a read of a diagonal convolution must pass through C and D; and a read of the resulting regular sequence must pass through Q before the terminal seal E is available.

The remaining rows are structural. The row H gives componentwise transport, T gives displayed replay, P records provenance, and N names the packet. No other row is present, so the local certificate cannot certify unconditional rearrangement, absolute convergence, analytic continuation, quotient stream equality, host completed-series multiplication, or ambient completeness. □

Theorem 6968.4 (Cauchy product dyadic budget stability)**.** *Let $K = (L, R, W, C, D, F, Q, E, H, T, P, N)$ be an accepted* $\mathsf{CauchyProductSeries}^{\uparrow}$ *carrier. Any refinement of the displayed output window that preserves the source windows, coefficient-product rows, convolution ledger, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget preserves the exported* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route and the downstream* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Start with the accepted packet and a refined finite window. The preservation hypothesis keeps the source rows L, R, coefficient-product row F, convolution ledger C, and dyadic budget row D componentwise aligned through the transport row H.

Induct over the refined window. The base window reads the same empty or initial convolution prefix. In the successor step, the new output index uses the same finite diagonal source reads, the same finite-product cells, and the same dyadic slack ledger as the original route.

The replay row T then gives the same displayed continuation from the refined convolution data into Q. Once Q is preserved, the seal row E remains downstream of the same readback route.

Thus refinement can add inspected finite data, but it cannot replace the budgeted convolution route by a host equality, selected limit, quotient representative, absolute-convergence argument, or analytic continuation step. □

Falsifiable prediction: An accepted $\mathsf{CauchyProductSeries}^{\uparrow}$ read is refuted if a product-series output window is consumed without first displaying the two source series packets, the finite output window, the convolution ledger, the DyadicRatCore tolerance budget, the RegularCauchyFiniteProduct coefficient-product rows, the RegSeqRat readback, the RealUp seal, transport, replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, regular-cauchy-finite-product: $\mathsf{CauchyProductSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source series packets, finite convolution windows, dyadic tolerance budgets, finite coefficient-product rows, regular readback, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductSeries}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductSeries}^{\uparrow}$ packet is generated from two source series packets, finite output windows, convolution ledgers, DyadicRatCore tolerance budgets, RegularCauchyFiniteProduct coefficient-product rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6968.1, Definition 6968.2, Definition 6968.3, and Definition 6968.4 seed the finite Cauchy product-series route over source series windows, convolution ledgers, dyadic budgets, finite-product coefficient rows, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductSeries}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyProductSeriesUp`

Bridge status: none

Not claimed: This chapter does not close unconditional rearrangement, absolute convergence, analytic continuation, host quotient series equality, selected infinite sums, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-source series exposure, finite convolution-window induction, dyadic tolerance-budget stability, RegularCauchyFiniteProduct coefficient-product routing, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductSeriesUp.

Chapter 6969

A Concrete Naming Certificate for $\mathsf{MoserIteration}^{\uparrow}$

Following Moser's iteration method in elliptic and parabolic regularity theory, the $\mathsf{MoserIteration}^{\uparrow}$ packet records the finite ladder by which Sobolev control, Poincare control, cutoff schedules, and exponent-step estimates are replayed before any Harnack-style consumer reads a regularity bound. It sits beside the Sobolev source of chapter 440, the Poincare control row of chapter 2329, the Morrey endpoint comparison of chapter 4005, and the terminal scalar seal of chapter 38. The packet names only displayed finite rows; it is not a host PDE theorem, not a selected weak solution, not an ambient compactness argument, and not a quotient equality of function spaces.

Definition 6969.1 (Moser iteration carrier)**.** *A* $\mathsf{MoserIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, P, J, E, R, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, including the finite energy and embedding data used by the ladder.*
2. *P is the* $\mathsf{PoincareInequality}^{\uparrow}$ *control row, recording the local oscillation and average-control estimates consumed at each step.*
3. *J is the cutoff schedule row, listing the displayed nested regions, cutoff names, and support restrictions before any estimate is read.*
4. *E is the exponent-step ledger, recording the finite sequence of exponent raises, constants, and admissible handoffs from one step to the next.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *bound readback row, sealing the scalar constants and endpoint bounds only after S, P, J, E have been displayed.*
6. *H records componentwise* hsame *transport for the Sobolev, Poincare, cutoff, exponent, Real-bound, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the same ladder for regularity and Harnack-style consumers.*
8. *Q is the* Pkg *provenance over* $\mathsf{MoserIteration}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, $\mathsf{MorreyInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MoserIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, P, J, E, R, H, C, Q, N$. It has no coordinate for a host solution space, a selected PDE solution, global elliptic regularity, parabolic compactness, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.MoserIterationUp`

Theorem 6969.2 (Moser iteration NameCert obligations)**.** *For every accepted* $\mathsf{MoserIteration}^{\uparrow}$ *carrier $M = (S, P, J, E, R, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MoserIteration}^{\uparrow}}$ *surface consists exactly of Sobolev source exposure through S; Poincare control through P; cutoff schedule exposure through J; exponent-step ledger exposure through E; Real-bound readback through R; componentwise transport through H; displayed replay through C; provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 6969.1. The rows $S, P, J, E, R, H, C, Q, N$ are exactly the Sobolev, Poincare, cutoff, exponent, Real-bound, transport, replay, provenance, and naming coordinates.

The analytic estimates are read in order. Sobolev and Poincare rows provide the source inequalities, the cutoff schedule restricts each local read, and the exponent-step ledger records the finite ladder before the Real-bound row seals constants or endpoints.

Finally H, C, Q, N preserve, replay, source, and name the same displayed packet. Since the carrier has no coordinate for a selected PDE solution, host regularity theorem, function-space quotient, choice schedule, or ambient completeness principle, the listed rows exhaust the local NameCert surface. □

Theorem 6969.3 (Moser iteration Harnack handoff)**.** *Every Harnack-style consumer of an accepted* $\mathsf{MoserIteration}^{\uparrow}$ *packet must first read the Sobolev source row S, the Poincare control row P, the cutoff schedule J, the exponent-step ledger E, the* $\mathsf{Real}^{\uparrow}$ *bound readback R, and the structural rows H, C, Q, N. The handoff supplies a finite regularity ladder for downstream Harnack inequality packets; it exports no host Harnack theorem, selected solution, quotient function equality, countable choice, open-cover compactness proof, or ambient real completeness theorem.*

Proof. Begin with Definition 6969.2. It fixes the accepted packet to the finite rows $S, P, J, E, R, H, C, Q, N$ and leaves no coordinate for a detached PDE theorem or selected solution.

The Harnack-facing read then consumes the ladder in the displayed order. The Sobolev and Poincare rows provide the source estimates, the cutoff schedule localizes the read, and the exponent-step ledger records the finite climb before any endpoint bound is sealed in R.

The structural rows complete the handoff. Componentwise transport, replay, provenance, and local naming preserve the same finite packet for downstream consumers. Therefore a Harnack-style packet receives only the displayed regularity ledger and cannot import the refused host, quotient, choice, compactness, or completeness endpoints. □

Falsifiable prediction: Every accepted Harnack-style read of $\mathsf{MoserIteration}^{\uparrow}$ must expose the Sobolev source, Poincare control, cutoff schedule, exponent-step ledger, Real-bound readback, transport, replay, provenance, and local NameCert rows before consuming a regularity bound.

Independence witness: sobolev, poincare-inequality, morrey-inequality, real: none is bijective with $\mathsf{MoserIteration}^{\uparrow}$ because this carrier alone binds cutoff schedules and exponent-step ledgers to the Sobolev, Poincare, Morrey-facing, Real-bound, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MoserIteration}^{\uparrow}}$**.**

Constructive story: A MoserIteration$^{\uparrow}$ packet is generated from Sobolev source rows, Poincare control rows, cutoff schedules, exponent-step ledgers, Real-bound readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MoserIteration}\uparrow}$, seed)

Scope closed: Definition 6969.1, Definition 6969.2, and Definition 6969.3 seed the finite PDE regularity ladder consumed by Harnack-style packets.

Formal verification: FormalStatus(NameCert$_{\text{MoserIteration}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host Harnack theorem, selected PDE solution, global regularity theorem, quotient equality of function spaces, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sobolev and Poincare source exactness, cutoff schedule exactness, exponent-step ledger exactness, Real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.MoserIterationUp.

Chapter 6970

A Concrete Naming Certificate for $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$

Following Bishop and Bridges on completion of separated metric spaces, the $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ packet records the finite BEDC surface by which an already separated completion is recognized as fixed by the completion classifier. It sits between the completion interface of chapter 95, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric-completion route of chapter 1506, and the functorial completion handoff of chapter 423. The packet names the idempotence boundary for separated completions; it is not a quotient-completion equality, a selected representative, countable choice, host real equality, or ambient completeness theorem.

Definition 6970.1 (Separated completion idempotence carrier)**.** *A* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (C, S, M, F, R, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Completion}^{\uparrow}$ *source row naming the completion-facing request.*
2. *S is the separated-classifier row, recording that zero-distance comparisons are already identified by the displayed separated completion surface.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row through which the same request reaches a metric-completion consumer.*
4. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *functorial handoff row.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the inspected completion windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the separated classifier, metric completion route, functorial handoff, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for $C, S, M, F, R, E, T, P, N$.*
8. *T records displayed* Cont *replay from a second completion request back through the same separated classifier route.*
9. *P is the* Pkg *provenance over* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionIdempotence}^{\uparrow}}$ *row.*

The classifier for accepted packets compares only the displayed rows $C, S, M, F, R, E, H, T, P, N$. It has no coordinate for quotient completion equality, selected representatives, host equality of completed points, countable choice, or an ambient completeness principle.

Theorem 6970.2 (Separated completion idempotence NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ *carrier* $I = (C, S, M, F, R, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionIdem}}$ *surface consists exactly of carrier admission for the completion source row, separated-classifier row, metric-completion route, Cauchy-completion functorial handoff, RegSeqRat readback, Real seal, componentwise transport, displayed replay, package provenance, and local naming. The second completion read must return to the same separated classifier* S *before any* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer can be served.*

Proof. Project the carrier of Definition 6970.1. The rows $C, S, M, F, R, E, H, T, P, N$ are the complete accepted inventory.

The separated classifier is the lock. A second completion-facing request first reads C and S; only after the zero-distance classifier has been exposed may it continue to the metric route M and functorial handoff F.

The readback and terminal seal remain downstream of that route. Rows R and E record regular rational windows and the Real seal after the separated completion boundary has been displayed, while H, T, P, N transport, replay, source, and name the same packet.

Since no other coordinate is present, the local certificate cannot export quotient completion equality, a selected representative, countable choice, host equality of completed points, or ambient completeness. □

Theorem 6970.3 (Separated completion RegSeqRat route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ *carrier factors through the finite route*

$$C, S, M, F \longmapsto R \longmapsto E.$$

Thus applying completion again to an already separated completion exposes the same separated classifier before the regular rational readback and terminal Real seal are available. The packet supplies idempotence as a finite NameCert route, not as quotient equality of completion objects or a selected completed representative.

Proof. Start with a consumer of R or E. By Definition 6970.2, the consumer first opens the completion source row C and the separated classifier row S.

The metric and functorial rows M and F keep the request on the sibling completion route. They show that the second completion read is served by the displayed metric-completion and Cauchy-completion handoff, not by an ambient equality of completion objects.

Only then can R and E be read. The regular rational row records the inspected windows, and the Real seal remains terminal to those windows. Transport, replay, provenance, and local naming preserve this route without adding a selected representative or choice principle. □

Lean4 (statement) `BEDC.Derived.SeparatedCompletionIdempotenceUp`

Falsifiable prediction: An accepted $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ read is refuted if a second completion-facing request reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without first exposing the $\mathsf{Completion}^{\uparrow}$ source row, separated classifier row, $\mathsf{MetricCompletion}^{\uparrow}$ route, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff, transport, replay, provenance, and local naming rows.

Independence witness: completion, regseqrat, real, metriccompletion, cauchycompletionfunctor: $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ is not bijective with any listed sibling because it jointly

carries the completion source, separated classifier, metric-completion route, functorial handoff, regular readback, Real seal, transport, replay, provenance, and local naming rows that witness idempotence of the separated completion classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionIdempotence}\uparrow}$.

Constructive story: A $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ packet is generated from a Completion row, separated-classifier row, MetricCompletion route, CauchyCompletionFunctor handoff, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionIdempotence}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6970.1, Definition 6970.2, and Definition 6970.3 seed the separated completion idempotence route over CompletionUp, RegSeqRatUp, RealUp, MetricCompletionUp, and CauchyCompletionFunctorUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionIdempotence}\uparrow}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.SeparatedCompletionIdempotenceUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed points, ambient completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-classifier idempotence, metric-completion route exposure, functorial handoff, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionIdempotenceUp.

Chapter 6971

A Concrete Naming Certificate for $\mathsf{RoundedCut}^{\uparrow}$

Following Bishop and Bridges, rounded cuts are the constructive boundary object that records rational approximation by finite open improvement data rather than by a completed quotient of predicates. The $\mathsf{RoundedCut}^{\uparrow}$ packet sits between the located-cut carrier of chapter 404, the one-sided lower-real surface of chapter 1446, the located-real interfaces of **??** and chapter 1933, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It names a finite rounded-cut certificate used by L10 Real and Completion consumers; it does not construct $\mathsf{Real}^{\uparrow}$ by Dedekind cuts.

Definition 6971.1 (Rounded cut carrier)**.** *A* $\mathsf{RoundedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, O, Q, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *source row, carrying the lower and upper rational-observation boundary.*
2. *O is the rounded-open improvement row: every accepted rational observation is witnessed by a strictly better displayed observation on the same side.*
3. *Q is the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row consumed by one-sided lower reads and located real reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the finite rational gaps used by roundedness and located comparison.*
5. *W is the finite window ledger for the rational observations inspected by downstream consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after L, O, Q, D, W have been replayed.*
7. *H records componentwise* hsame *transport for the located-cut source, rounded improvement, comparison, dyadic ledger, window, real seal, continuation, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a rational or dyadic request through L, O, Q, D, W before the seal row E is read.*
9. *P is the* Pkg *provenance over* $\mathsf{RoundedCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RoundedCut}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a quotient cut, selected representative, host real equality, ambient completeness theorem, or construction of $\mathsf{Real}^{\uparrow}$ *by Dedekind cuts.*

Theorem 6971.2 (Rounded cut NameCert obligations)**.** *For every accepted* $\mathsf{RoundedCut}^{\uparrow}$ *carrier* $R = (L, O, Q, D, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RoundedCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed located-cut source, rounded-open improvement row, lower and located real comparison row, dyadic tolerance ledger, finite observation window, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows. No obligation is stored in a quotient predicate, a selected cut representative, host real equality, ambient completeness, or a Dedekind-cut construction of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 6971.1. The rows $L, O, Q, D, W, E, H, C, P, N$ are the complete inventory of the packet.

The rounded-open row O and the dyadic ledger D witness improvement by finite rational data, while Q records how lower-real and located-real consumers read the same finite boundary. The structural rows H, C, P, N only transport, replay, source, and name that displayed packet.

Since no further coordinate is present, the NameCert obligations are exhausted by the listed rows. In particular the packet is a finite rounded-cut certificate and does not construct $\mathsf{Real}^{\uparrow}$ by Dedekind cuts. □

Theorem 6971.3 (Rounded cut located-real handoff)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RoundedCut}^{\uparrow}$ *carrier must first read the route*

$$L, O, Q, D, W, E, H, C, P, N.$$

The handoff exports rounded located rational observations, dyadic gap data, finite windows, and the terminal real seal only as displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data.*

Proof. Begin with a located-real or lower-real consumer. Carrier admission exposes the located-cut row L, the rounded-open improvement row O, and the comparison row Q before any real-facing seal can be inspected.

Next read the dyadic ledger D and window row W. These rows provide the finite gap and observation data used by roundedness and located comparison.

Finally replay through C and attach H, P, N before reading E. The consumer therefore receives only the finite route named above, not a quotient cut, selected representative, host equality, or ambient completeness principle. □

Theorem 6971.4 (Rounded cut non-Dedekind completion boundary)**.** *The* $\mathsf{RoundedCut}^{\uparrow}$ *packet is a finite rounded-cut certificate for dependency routing between located cuts, lower reals, located reals, dyadic witnesses, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is not a construction of* $\mathsf{Real}^{\uparrow}$ *by Dedekind cuts, not a quotient completion, not a selected-limit principle, and not an ambient real-completeness theorem.*

Proof. Read the carrier inventory of Definition 6971.1. Its analytic rows are the displayed located-cut source, rounded-open improvement, lower and located real comparison, dyadic tolerance ledger, finite window, and terminal seal.

Apply Definition 6971.2 and Definition 6971.3. They force every consumer to remain inside the finite rounded-cut certificate before reaching any $\mathsf{Real}^{\uparrow}$ seal.

Thus the chapter records a boundary and handoff surface only. It does not supply Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient equality of cuts, selected representatives, selected limits, or ambient completeness. □

Falsifiable prediction: A $\mathsf{RoundedCut}^{\uparrow}$ consumer that reaches a $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without first reading the displayed located-cut source, rounded-open improvement, dyadic tolerance ledger, finite observation window, replay, provenance, and local NameCert rows refutes this carrier surface.

Independence witness: locatedcut, lowerreal, locatedreal, dyadicratcore, real: no listed sibling jointly binds rounded-open improvement, dyadic gap data, lower and located real handoff, terminal Real seal, and local NameCert routing in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedCut}^{\uparrow}}$.

Constructive story: A $\mathsf{RoundedCut}^{\uparrow}$ packet is generated from a displayed located-cut source, rounded-open improvement row, lower and located real comparison row, dyadic tolerance ledger, finite observation window, terminal Real seal, componentwise transport, replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6971.1, Definition 6971.2, Definition 6971.3, and Definition 6971.4 bind the finite rounded-cut certificate over $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient cuts, selected representatives, selected limits, ambient completeness, or a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, rounded-open improvement replay, located-real handoff, and non-Dedekind completion boundary for BEDC.Derived.RoundedCutUp.

Chapter 6972

A Concrete Naming Certificate for $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$ packet names the finite route by which a displayed dyadic rational source enters the regular Cauchy completion surface and reaches the $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic source packet of chapter 291, the finite schedule of chapter 126, the regular rational readback of chapter 301, the admission criterion of chapter 1476, and the terminal Real packet of chapter 38. The certificate is an explicit embedding handoff, not a quotient stream, selected limit, host real equality, or ambient completeness principle.

Definition 6972.1 (Dyadic completion embedding carrier)**.** *A* $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, R, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, including the dyadic numerator, scale, comparison ledger, and tolerance budget used by the embedding request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window through which the dyadic source is scheduled for completion-facing reads.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance ledger.*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R and the criterion row have been displayed.*
5. *Q is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *admission row, recording that the StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal handoff are mutually compatible.*
6. *H stores componentwise* hsame *transport for the source, window, readback, seal, criterion, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the dyadic source through the stream window, regular readback, completion criterion, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCompletionEmbedding}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by these nine rows. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness of the Real completion.

Lean4 (definition) `BEDC.Derived.DyadicCompletionEmbeddingUp`

Theorem 6972.2 (Dyadic completion embedding NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$ *carrier* $E = (D, S, R, L, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicCompletionEmbedding}^{\uparrow}}$ *surface consists exactly of dyadic source exposure through* D*, finite StreamName scheduling through* S*, RegSeqRat readback through* R*, CauchyCompletionCriterion admission through* Q*, Real seal handoff through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionEmbeddingUp.DyadicCompletionEmbeddingNameCertObligations`

Proof. Project the carrier of Definition 6972.1. The dyadic source, stream window, regular readback, Real seal, completion criterion, transport, replay, provenance, and local naming rows are the only classifier-visible coordinates.

The source and readback obligations are forced by the route. The dyadic row D supplies the displayed rational source and tolerance budget, the stream row S supplies the finite observation schedule, and the regular readback row R is read only after those two rows are visible.

The completion-facing obligations are then local. The criterion row Q admits the handoff to the Real seal row L, while H, C, P, N transport, replay, package, and name the same finite packet.

Since the carrier has no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6972.3 (Dyadic completion embedding Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyCompletionCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives the dyadic source row, finite window, regular readback, criterion admission, Real seal, transport, replay, provenance, and local naming rows, and receives no quotient stream, selected limit, host real equality, countable-choice schedule, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicCompletionEmbeddingUp.DyadicCompletionEmbeddingRealSealHandoff`

Proof. Start with an accepted carrier $E = (D, S, R, L, Q, H, C, P, N)$. The only source row is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row D, so a Real-facing read cannot begin from an unstated rational representative or a host real.

Next read the finite observation route. The $\mathsf{StreamName}^{\uparrow}$ row S schedules the dyadic source, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is the readback obtained from that schedule together with the displayed tolerance ledger.

Apply the admission row Q. It is the sole local $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ row that licenses the move from the regular rational readback to the seal row L.

The structural rows H, C, P, N preserve the same finite packet during transport, replay, provenance, and naming. Therefore the terminal $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed handoff route and cannot read quotient streams, selected limits, host real equality, countable choice, or ambient completeness from the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCompletionEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCompletionEmbedding}^{\uparrow}$ packet is generated from a DyadicRatCore source row, StreamName observation window, RegSeqRat readback, RealUp seal, CauchyCompletionCriterion admission row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCompletionEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6972.1, Definition 6972.2, and Definition 6972.3 seed the finite dyadic-to-regular-Cauchy embedding route through StreamName, RegSeqRat, CauchyCompletionCriterion, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCompletionEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DyadicCompletionEmbeddingUp`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName scheduling, RegSeqRat readback, completion-criterion admission, and Real seal handoff for the same dyadic source packet.

Chapter 6973

A Concrete Naming Certificate for $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$ packet records the criterion-level route from absolute domination to Cauchy control rather than a new summability carrier. It sits over the absolute-domination packet of chapter 1700, the finite majorant schedule of chapter 1774, the shared-tail series route of chapter 1736, the regular rational readback of chapter 301, and the endpoint seal of chapter 38. The packet gives later Real and Completion consumers one finite criterion certificate that displays the absolute-value envelope, the majorant comparison, the uniform Cauchy tail handoff, and the $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ boundary before any endpoint-facing read is public.

Definition 6973.1 (Absolute convergence criterion carrier)**.** *An* $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, M, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *row, including the term stream, absolute-domination ledger, and finite absolute-value envelope opened by the criterion.*
2. *M is the* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *row, including the nonnegative majorant schedule, partial-sum ledger, and displayed tail budget.*
3. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *row reached only after A and M have exposed the domination and majorant data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite term, majorant, and partial-sum windows opened along the criterion route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, public only after A, M, U, R have been displayed.*
6. *H stores componentwise* hsame *transport for the absolute-domination, majorant, uniform-tail, readback, Real-seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the absolute-value envelope to the majorant comparison, from the majorant comparison to the uniform Cauchy threshold, from the threshold to regular readback, and from readback to the Real seal.*

8. *P is the* Pkg *provenance over* AbsoluteConvergenceCriterion$^{\uparrow}$, AbsoluteConvergence$^{\uparrow}$, CauchyMajorantSeries$^{\uparrow}$, UniformCauchySeries$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{AbsoluteConvergenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, U, R, E, H, C, P, N$. It has no coordinate for a selected infinite sum, quotient stream equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6973.2 (Absolute convergence criterion NameCert obligations)**.** *For every accepted* AbsoluteConvergenceCriterion$^{\uparrow}$ *carrier*

$$K = (A, M, U, R, E, H, C, P, N),$$

the local NameCert$_{\text{AbsoluteConvergenceCriterion}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the absolute-domination, majorant-series, uniform Cauchy, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *absolute-value envelope exactness, requiring A to expose the selected absolute comparisons before any majorant row is read;*
3. *majorant comparison, requiring M to supply the displayed nonnegative schedule and tail budget consumed by the criterion;*
4. *uniform Cauchy tail handoff, requiring U to be read only after the absolute-value envelope and majorant comparison are both visible;*
5. Real$^{\uparrow}$/RegSeqRat$^{\uparrow}$ *nonescape, requiring R and E to expose only the finite regular readback and terminal Real seal reached by the displayed route.*

No criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6973.1. The accepted packet displays exactly the rows $A, M, U, R, E, H, C, P, N$, so the certificate surface has no detached coordinate for a completed sum or a host convergence theorem.

The absolute-value envelope is local to A. A criterion read begins by opening the selected term observations and their displayed absolute comparisons, and these are the only comparisons that can feed the majorant row.

The row M supplies the majorant comparison and tail schedule. Since U is downstream of A and M, the uniform Cauchy threshold cannot be read before the domination ledger and majorant budget have both been displayed.

Finally, R and E are terminal with respect to that finite route. The structural rows H, C, P, N transport, replay, source, and name the same rows, and the excluded endpoint principles never appear among the carrier coordinates. □

Theorem 6973.3 (Absolute convergence criterion majorant handoff)**.** *Every Real or Completion consumer that reads an accepted* AbsoluteConvergenceCriterion$^{\uparrow}$ *carrier factors through the finite route*

$$A \longmapsto M \longmapsto U \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer must display an AbsoluteConvergence$^{\uparrow}$ *absolute-value envelope, a* CauchyMajorantSeries$^{\uparrow}$ *comparison schedule, a* UniformCauchySeries$^{\uparrow}$ *shared tail threshold, a* RegSeqRat$^{\uparrow}$ *readback, and a terminal* Real$^{\uparrow}$ *seal before any endpoint-facing result is public.*

Proof. Project the accepted carrier $K = (A, M, U, R, E, H, C, P, N)$. The first public data are the absolute-domination rows in A, including the selected absolute-value envelope for the term stream.

The next public data are the majorant rows in M. They supply the displayed nonnegative comparison schedule and the tail budget that the criterion uses to control partial-sum differences.

The uniform Cauchy handoff occurs only after those two rows are visible. The row U records the shared threshold for the series tail, and R reads back the selected finite windows as regular rational data.

The endpoint row E is therefore a $\mathsf{Real}^{\uparrow}$ seal for the already displayed finite route. The rows H, C, P, N preserve and replay the same dependency order, so no consumer can bypass the majorant schedule or export a selected infinite sum, quotient equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$ endpoint-facing read factors through an AbsoluteConvergence absolute-value envelope, a CauchyMajorantSeries comparison schedule, a UniformCauchySeries shared tail threshold, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: absolute-convergence, cauchy-majorant-series, uniform-cauchy-series, regseqrat, real: $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the criterion-level route from absolute domination through majorant comparison, uniform Cauchy tail control, regular readback, Real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6973.1, Definition 6973.2, and Definition 6973.3 seed the finite criterion route from absolute domination through majorant comparison, uniform Cauchy tail handoff, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, absolute-value envelope exactness, majorant comparison, uniform Cauchy tail handoff, Real/RegSeqRat nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteConvergenceCriterionUp.

Constructive story: An $\mathsf{AbsoluteConvergenceCriterion}^{\uparrow}$ packet is generated from an AbsoluteConvergence row, a CauchyMajorantSeries row, a UniformCauchySeries row, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6974

A Concrete Naming Certificate for $\mathsf{HarnackInequality}^{\uparrow}$

Following Moser's iteration method and the elliptic regularity presentation of Gilbarg–Trudinger, the $\mathsf{HarnackInequality}^{\uparrow}$ packet records the finite route by which a positive PDE solution read consumes Sobolev control, Poincare control, cutoff schedules, exponent-step ledgers, and terminal real bounds before a Harnack comparison is named. It sits after the Moser ladder of chapter 6969, the PDE relation surface of chapter 194, the Sobolev source of chapter 440, the Poincare control row of chapter 2329, and the scalar seal of chapter 38. The packet names only the displayed local comparison certificate; it is not a host regularity theorem, not a selected weak solution, not an open-cover compactness argument, and not an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 6974.1 (Harnack inequality carrier)**.** *A* $\mathsf{HarnackInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, S, P, J, E, R, B, T, C, Q, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MoserIteration}^{\uparrow}$ *handoff row, exposing the finite ladder consumed before any Harnack read.*
2. *D is the* $\mathsf{Pde}^{\uparrow}$ *relation row, naming the displayed equation and boundary classifier over the same local domain.*
3. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row whose weak-derivative and energy data feed the ladder.*
4. *P is the* $\mathsf{PoincareInequality}^{\uparrow}$ *control row for local oscillation and average comparison.*
5. *J is the cutoff and nested-region row inherited from the Moser ladder.*
6. *E is the exponent-step row that records the finite climb from local energy bounds to the comparison window.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *bound readback row for constants, radii, and positive lower-bound data.*
8. *B is the positive-solution boundary row, requiring the displayed solution window to remain inside the admitted PDE and Real-bound rows.*
9. *T records componentwise* hsame *transport for the Moser, PDE, Sobolev, Poincare, cutoff, exponent, Real-bound, positive-boundary, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same route for downstream PDE regularity consumers.*

11. *Q is the* Pkg *provenance over* $\mathsf{HarnackInequality}^{\uparrow}$, $\mathsf{MoserIteration}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HarnackInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, S, P, J, E, R, B, T, C, Q, N$. It has no coordinate for an arbitrary host solution, global elliptic regularity, quotient equality of functions, selected representatives, countable choice, open-cover compactness, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.HarnackInequalityUp`

Theorem 6974.2 (Harnack inequality NameCert obligations)**.** *For every accepted* $\mathsf{HarnackInequality}^{\uparrow}$ *carrier $H = (M, D, S, P, J, E, R, B, T, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HarnackInequality}^{\uparrow}}$ *surface consists exactly of Moser-ladder exposure through M; PDE relation exposure through D; Sobolev source exposure through S; Poincare control exposure through P; cutoff schedule exposure through J; exponent-step exposure through E; Real-bound readback through R; positive-solution boundary exposure through B; componentwise transport through T; displayed replay through C; provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 6974.1. The rows $M, D, S, P, J, E, R, B, T, C, Q, N$ are exactly the Moser, PDE, Sobolev, Poincare, cutoff, exponent, Real-bound, positive-boundary, transport, replay, provenance, and naming coordinates.

The analytic rows are read in dependency order. The PDE row fixes the displayed relation, the Sobolev and Poincare rows supply the source estimates, and the Moser row brings the cutoff and exponent ladder into the same finite packet.

The endpoint rows then seal the comparison. The Real-bound row records constants and positive lower-bound data, while the positive-solution boundary row prevents the comparison from being read outside the admitted PDE window. Transport, replay, provenance, and local naming preserve only this finite route, so the listed rows exhaust the local NameCert surface. □

Theorem 6974.3 (Harnack inequality Moser handoff)**.** *Every accepted* $\mathsf{HarnackInequality}^{\uparrow}$ *read must first consume the* $\mathsf{MoserIteration}^{\uparrow}$ *handoff row together with the displayed* $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, *cutoff, exponent, and* $\mathsf{Real}^{\uparrow}$ *rows. The Harnack comparison is therefore a downstream read of the finite Moser ladder, not a separate host regularity theorem or selected-solution principle.*

Proof. Start with the Moser handoff of Definition 6969.3. It requires every Harnack-facing consumer to expose the Sobolev source, Poincare control, cutoff schedule, exponent ledger, Real-bound readback, and structural rows before a regularity bound is consumed.

The carrier of Definition 6974.1 imports exactly that handoff as M and keeps the source rows visible as S, P, J, E, R. Thus a Harnack read cannot bypass the Moser ladder.

Finally attach the PDE and positive-boundary rows. They restrict the comparison to the displayed relation and positive solution window, while T, C, Q, N only transport, replay, source, and name the same packet. Hence the handoff is finite and does not export a detached host theorem. □

Theorem 6974.4 (Harnack inequality positive-solution boundary)**.** *Every positive-solution comparison exported by an accepted* $\mathsf{HarnackInequality}^{\uparrow}$ *packet factors through the PDE relation row D, the positive-boundary row B, and the Real-bound readback row R after the Moser ladder rows M, S, P, J, E have been displayed. It exports no comparison for an unnamed solution, no quotient equality of representatives, and no ambient maximum principle outside*

$$M, D, S, P, J, E, R, B, T, C, Q, N.$$

Proof. Read the PDE-facing rows first. The row D fixes the displayed PDE relation and boundary classifier, while B states that the positive solution window being compared is part of that same admitted route.

Next consume the ladder and scalar seal. The rows M, S, P, J, E supply the finite Moser ladder, and R supplies the scalar constants and positive lower-bound data used by the comparison.

The structural rows finish the route. Transport, replay, provenance, and local naming through T, C, Q, N preserve the same finite packet and introduce no selected solution, quotient representative, host maximum principle, or ambient completeness theorem. Therefore every positive-solution comparison remains inside the displayed carrier rows. □

Falsifiable prediction: Every accepted $\mathsf{HarnackInequality}^{\uparrow}$ read must expose the Moser ladder, PDE relation, Sobolev source, Poincare control, cutoff schedule, exponent ledger, Real-bound readback, positive-solution boundary, transport, replay, provenance, and local NameCert rows before any Harnack comparison is consumed.

Independence witness: moser-iteration, pde, sobolev, poincare-inequality, real: no listed sibling is bijective with $\mathsf{HarnackInequality}^{\uparrow}$ because this packet alone binds the finite Moser ladder to a displayed positive PDE solution comparison and terminal Real-bound seal.

Closure status for $\mathsf{NameCert}_{\mathsf{HarnackInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{HarnackInequality}^{\uparrow}$ packet is generated from a MoserIteration handoff row, a PDE relation row, Sobolev source data, Poincare control data, cutoff schedules, exponent-step ledgers, Real-bound readback, a positive-solution boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HarnackInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6974.1, Definition 6974.2, Definition 6974.3, and Definition 6974.4 seed the finite Harnack comparison packet consumed downstream from $\mathsf{MoserIteration}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HarnackInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Harnack theorem, selected PDE solution, global regularity theorem, quotient equality of functions, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Moser handoff exactness, PDE relation exactness, Sobolev and Poincare source exactness, cutoff and exponent ledger exactness, Real-bound readback, positive-boundary preservation, transport, replay, provenance, and local naming for BEDC.Derived.HarnackInequalityUp.

Chapter 6975

Tucker Lemma NameCert Construction

Following Tucker and the standard combinatorial proof of Borsuk–Ulam, the $\mathsf{TuckerLemma}^{\uparrow}$ packet records the finite antipodal labeling obstruction as NameCert data. It sits between the finite face inventory of chapter 238, the adjacent-color argument of chapter 2003, and the antipodal conclusion route of chapter 5828. The packet is a finite combinatorial certificate: it exposes triangulation rows, boundary antipodality rows, complementary-edge witnesses, structural transport, replay, provenance, and local naming, but it does not import compactness, quotient spheres, choice of subdivisions, or a host topological Borsuk–Ulam theorem.

Definition 6975.1 (Tucker lemma carrier)**.** *A* $\mathsf{TuckerLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, B, L, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite antipodal triangulation row, read as a* $\mathsf{SimplicialComplex}^{\uparrow}$ *face inventory with a displayed vertex-pair involution.*
2. *B is the boundary labeling row, recording the antipodal sign rule on boundary vertex pairs.*
3. *L is the interior labeling row, assigning each displayed vertex one signed label from a finite symmetric label alphabet.*
4. *E is the complementary-edge witness row, naming an inspected edge whose two endpoint labels are opposite signs with the same absolute label.*
5. *H records componentwise* hsame *transport for the triangulation, boundary rule, labeling, edge witness, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from the finite triangulation and labeling data to the complementary-edge witness.*
7. *P is the* Pkg *provenance over* $\mathsf{TuckerLemma}^{\uparrow}$*,* $\mathsf{SimplicialComplex}^{\uparrow}$*,* $\mathsf{SpernerLemma}^{\uparrow}$*,* $\mathsf{BorsukUlam}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TuckerLemma}^{\uparrow}}$ *row naming exactly this finite antipodal labeling packet.*

The classifier compares accepted packets only through K, B, L, E, H, C, P, N. It has no coordinate for an ambient sphere quotient, a selected subdivision sequence, real compactness, or a host topological fixed-point theorem.

Theorem 6975.2 (Tucker lemma finite complementary edge route)**.** *For every accepted* $\mathsf{TuckerLemma}^{\uparrow}$ *carrier* $T = (K, B, L, E, H, C, P, N)$*, every complementary-edge read factors through the displayed finite route*

$$\mathsf{SimplicialComplex}^{\uparrow} \longmapsto \textit{antipodal boundary labels} \longmapsto \textit{signed interior labels} \longmapsto \textit{complementary edge.}$$

Equivalently, the consumer-visible surface is exactly K, B, L, E, H, C, P, N*: the finite antipodal triangulation is inspected first, the boundary and interior label rows are replayed on that triangulation, and the edge witness is then exposed through the structural transport and naming rows.*

Proof. Project the carrier of Definition 6975.1. The row K is the only triangulation source, B is the only boundary antipodality source, and L is the only signed-label source.

Read the route in order. A consumer first receives a finite face inventory through $\mathsf{SimplicialComplex}^{\uparrow}$. The boundary rule B then fixes the antipodal sign comparison on displayed boundary pairs, and the labeling row L supplies the signed labels on all inspected vertices.

The witness row E is exposed only after those rows have been transported by H and replayed by C. The provenance and naming rows P, N name the same finite packet, so the complementary edge cannot be replaced by a host sphere argument or a hidden subdivision choice.

Since no coordinate records quotient spheres, compactness, selected subdivision sequences, or ambient fixed-point data, every accepted complementary-edge read remains inside the finite K, B, L, E, H, C, P, N surface. □

Theorem 6975.3 (Tucker lemma NameCert obligations)**.** *For every accepted* $\mathsf{TuckerLemma}^{\uparrow}$ *carrier* $T = (K, B, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TuckerLemma}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite packet* T*;*
2. *finite antipodal* $\mathsf{SimplicialComplex}^{\uparrow}$ *triangulation exposure through* K*;*
3. *boundary antipodality of the labeling row through* B*;*
4. *signed interior label exposure through* L*;*
5. *complementary-edge witness exposure through* E*;*
6. *componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming through* H, C, P, N*.*

No additional NameCert obligation is exported for compactness, quotient-sphere equality, selected subdivisions, real vector topology, or a host Borsuk–Ulam theorem.

Proof. Carrier admission is the finite tuple displayed in Definition 6975.1. Each listed obligation is one of its coordinates, so a local name can inspect it without leaving the packet.

The finite complementary-edge route of Definition 6975.2 orders the obligations. The triangulation row supplies the finite complex, the boundary and interior label rows supply the signed data, and the edge row supplies the complementary witness.

The structural rows H, C, P, N preserve and name that same packet. Because the excluded objects are not carrier coordinates, they cannot create extra NameCert obligations or a second route to the witness. □

Lean4 (statement) `BEDC.Derived.TuckerLemmaUp`

Falsifiable prediction: Every accepted $\mathsf{TuckerLemma}^{\uparrow}$ consumer can be audited by checking that its complementary-edge witness first displays a finite $\mathsf{SimplicialComplex}^{\uparrow}$ antipodal triangulation, then boundary and interior signed-label rows, before E, H, C, P, N are consumed.

Independence witness: simplicialcomplex, spernerlemma, borsukulam: $\mathsf{TuckerLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the antipodal triangulation row, boundary labeling row, interior signed-label row, complementary-edge witness, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TuckerLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{TuckerLemma}^{\uparrow}$ packet is generated from a finite antipodal SimplicialComplex row, boundary antipodality rows, signed interior label rows, a complementary-edge witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TuckerLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6975.1, Definition 6975.2, and Definition 6975.3 seed the finite antipodal-labeling certificate route from SimplicialComplex data to a complementary-edge witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TuckerLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, quotient-sphere equality, selected subdivisions, real vector topology, Lean proof coverage for the combinatorial theorem, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal triangulation exposure, boundary labeling, signed interior labeling, complementary-edge witness exposure, transport, replay, provenance, and local naming for BEDC.Derived.TuckerLemmaUp.

Chapter 6976

Regular Cauchy Located Limit Seal NameCert Construction

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$ packet records the finite route by which a located regular Cauchy name reaches a Real seal. It sits beside the regular rational readback of chapter 301, the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the located-real surface of **??**, and the terminal Real seal of chapter 38. The packet records a displayed limit-seal handoff; it is not quotient equality of real names, a selected host limit, countable choice, or ambient completeness.

Definition 6976.1 (Regular Cauchy located limit-seal carrier)**.** *A* $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, R, D, A, S, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that exposes the requested prefix of the regular Cauchy name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the regularity budget and located comparison cells.*
4. *A is the* $\mathsf{LocatedReal}^{\uparrow}$ *row recording the located cut information visible through the same W, R, D route.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after the window, readback, dyadic ledger, and located row have been replayed.*
6. *H records componentwise* hsame *transport for the window, readback, dyadic, located, seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from W, R, D, A to the Real seal S.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these nine rows. It has no coordinate for quotient stream equality, selected host limits, host real equality, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 6976.2 (Regular Cauchy located limit-seal obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocatedLimitS}$ *carrier* $L = (W, R, D, A, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed nine-row packet;*
2. $\mathsf{StreamName}^{\uparrow}$ *window locality through* W*;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback through* R*;*
4. $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and located-cell exactness through* D*;*
5. $\mathsf{LocatedReal}^{\uparrow}$ *located-route compatibility through* A*, together with* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise transport, displayed replay, provenance, and local naming through* S, H, C, P, N*.*

No additional NameCert obligation is exported for quotient equality, selected limits, host real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 6976.1. Its classifier-visible coordinates are precisely the finite window, regular readback, dyadic tolerance ledger, located row, terminal seal, transport, replay, provenance, and local naming rows.

The window-locality and readback obligations are forced by the order of W and R. A consumer first sees the finite $\mathsf{StreamName}^{\uparrow}$ window and then reads it as a regular rational packet through $\mathsf{RegSeqRat}^{\uparrow}$.

The dyadic and located obligations are the next displayed rows. The row D supplies the tolerance and located-cell comparisons, and the row A records the located-real surface read only from the same W, R, D route.

Finally, the seal and structural rows add no second source. The row S is exposed only after W, R, D, A have been replayed through H, C, P, N, and the carrier contains no quotient, selected-limit, host-equality, choice, or completeness coordinate. Hence the five listed rows exhaust the local NameCert surface. □

Theorem 6976.3 (Regular Cauchy located Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$ *carrier* $L = (W, R, D, A, S, H, C, P, N)$ *reads exactly the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The handoff exports no quotient stream, selected host limit, host real equality, countable-choice row, or ambient completeness principle.*

Proof. Begin with a Real-facing or completion-facing consumer and project the accepted carrier. The only source window is W, and the only regular rational readback is R.

The dyadic ledger D records the tolerance and located comparison cells used by that readback. The located row A then reads the same displayed cells; it does not supply a detached host real or a selected limit.

The Real seal S is reached only after the W, R, D, A route has been replayed through H, C, P, N. These structural rows preserve the same finite packet and provide no extra coordinate from which a quotient, selected limit, host equality, choice principle, or ambient completeness theorem could be recovered.

Thus every accepted handoff factors through the displayed StreamName, RegSeqRat, DyadicRatCore, LocatedReal, and Real rows, followed by the structural certificate rows. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$ consumer can be audited by checking that its Real-facing read first displays a $\mathsf{StreamName}^{\uparrow}$ window, then a $\mathsf{RegSeqRat}^{\uparrow}$ readback, then a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, then a $\mathsf{LocatedReal}^{\uparrow}$ located row, before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Independence witness: Nearest siblings: streamname, regseqrat, dyadicratcore, locatedreal, and real. The packet is not bijective to any one sibling because it jointly carries the finite window, regular rational readback, dyadic tolerance ledger, located-real row, Real seal, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}$ packet is generated from a StreamName finite-window row, a RegSeqRat regular rational readback row, a DyadicRatCore tolerance ledger, a LocatedReal row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6976.1, Definition 6976.2, and Definition 6976.3 seed the finite located regular Cauchy route from L10 StreamName, RegSeqRat, DyadicRatCore, and LocatedReal data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimitSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real names, selected host limits, host real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName locality, RegSeqRat readback, DyadicRatCore tolerance exactness, LocatedReal compatibility, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedLimitSealUp.

Chapter 6977

A Concrete Naming Certificate for $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ packet records a nested rational interval argument as finite endpoint windows with a displayed dyadic width descent. It sits over the dyadic tolerance arithmetic of chapter 291, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers one Bishop-style nested rational interval route without importing open-cover compactness, quotient real equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6977.1 (Constructive nested real interval carrier)**.** *A* $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are rational endpoint rows for the inspected left and right interval endpoints, each read through the dyadic tolerance discipline of chapter 291.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window listing the inspected stages and the endpoint observations available at each stage.*
3. *D is the dyadic width-descent ledger, recording $L_n \leq L_{n+1} \leq R_{n+1} \leq R_n$ and a displayed dyadic budget forcing the inspected widths below the requested tolerance.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *selector row that chooses, from each inspected interval, a midpoint or endpoint rational compatible with the displayed width budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, R, S, D, Q have been displayed.*
6. *H stores componentwise* hsame *transport for endpoint rows, stream windows, width descent, selector readback, Real seal, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from finite endpoint observations through width descent, selector readback, and the terminal Real seal.*
8. *P records* Pkg *provenance over* $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveNestedRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, S, D, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, choice-selected limits, Heine–Borel, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6977.2 (Constructive nested real interval NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ *carrier*

$$K = (L, R, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveNestedRealInterval}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for endpoint rows, finite stage windows, dyadic width descent, selector readback, terminal Real sealing, transport, replay, provenance, and local naming;*
2. *classifier visibility, requiring packet comparison to inspect only $L, R, S, D, Q, E, H, C, P, N$;*
3. *stability, requiring* hsame *transport to preserve endpoint order, successor containment, dyadic width budgets, selector reads, and the terminal seal;*
4. *exactness, requiring every selector read in Q to lie inside the displayed interval stage from which it is drawn and to use no uninspected endpoint;*
5. *ledger closure, requiring* Cont *replay and* Pkg *provenance to record the full route from finite endpoint windows through the* $\mathsf{RegSeqRat}^{\uparrow}$ *selector to the* $\mathsf{Real}^{\uparrow}$ *seal.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6977.1. The accepted packet displays exactly the endpoint, stage-window, width-descent, selector, seal, transport, replay, provenance, and local naming rows.

Classifier visibility follows because the packet has no further coordinate. A comparison of accepted packets can inspect the listed rows, but it cannot ask for open-cover compactness, quotient real equality, a selected host limit, Heine–Borel, trichotomy, or ambient completeness.

The stability and exactness clauses are read from the same displayed rows. The transport row H preserves the endpoint order, containment, width budget, selector readback, and Real seal, while D and Q record that each rational selector read is taken from an inspected interval stage and satisfies the displayed dyadic tolerance request.

Finally, C and P close the ledger. They replay and source the route from finite interval windows through the regular rational selector to the terminal seal, so no detached obligation remains available outside $L, R, S, D, Q, E, H, C, P, N$. □

Theorem 6977.3 (Constructive nested interval Cauchy selector)**.** *Every accepted* $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ *carrier determines a* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible Cauchy selector whose terminal* $\mathsf{Real}^{\uparrow}$ *seal remains inside the displayed nested interval packet. More precisely, the selector reads only the finite endpoint windows in S, the dyadic width-descent ledger D, and the rational row Q; for each requested dyadic tolerance, a displayed stage supplies a rational selector value inside the corresponding interval, and the terminal seal E is reached only through that selector route.*

Proof. Project the accepted carrier $K = (L, R, S, D, Q, E, H, C, P, N)$. The finite stream window S tells which stages may be inspected, and the endpoint rows L, R give the rational bounds available at those stages.

The ledger D supplies successor containment and a dyadic width budget. Given a displayed tolerance request, the budget exposes an inspected stage whose interval width is below that tolerance, and containment ensures later inspected stages remain inside the earlier interval.

The selector row Q reads a rational from the displayed interval stage, for instance a midpoint or an endpoint named by the packet. Since both selected values for sufficiently narrow inspected stages lie inside a common displayed interval whose width is below the tolerance, the selected rational reads satisfy the $\mathsf{RegSeqRat}^{\uparrow}$ Cauchy condition.

The terminal row E is therefore reached only after the selector has passed through Q. The replay row C, transport row H, provenance row P, and local name N preserve that dependency, so the Real seal cannot escape the finite nested interval packet or appeal to a choice-selected limit, quotient equality, open-cover compactness, Heine–Borel, trichotomy, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ Real-facing read factors through finite rational endpoint windows, a dyadic width-descent ledger, a RegSeqRat-compatible selector row, terminal Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries nested rational endpoints, inspected stage windows, dyadic width descent, selector readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveNestedRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveNestedRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6977.1, Definition 6977.2, and Definition 6977.3 seed the finite nested rational interval route from endpoint windows through dyadic width descent, regular rational selection, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveNestedRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient real equality, choice-selected limits, Heine–Borel, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier visibility, stability, exactness, ledger closure, selector Cauchy control, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveNestedRealI

Constructive story: A $\mathsf{ConstructiveNestedRealInterval}^{\uparrow}$ packet is generated from rational endpoint rows, finite StreamName windows, a dyadic width-descent ledger, RegSeqRat selector readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6978

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$

Following Bishop and Bridges on Cauchy completion, the $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$ packet names the finite operator route that turns displayed regular rational Cauchy data into a Real-facing completion seal. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the neighboring Cauchy completion operator of chapter 3379. The packet records the regular-operator handoff only; it does not quotient Cauchy names, select representatives, use countable choice, assert ambient completeness of $\mathsf{Real}^{\uparrow}$, or replace the broader Bishop completion operator of chapter 6456.

Definition 6978.1 (RegularCauchyCompletionOperator carrier)**.** *A* $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for every precision request and every regularity comparison.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observed windows of the Cauchy name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback row attached to those windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tolerance, window, and readback rows have been exposed.*
5. *H records componentwise* hsame *transport for the dyadic tolerance, stream window, regular readback, Real seal, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from a precision request through D, S, R, then through the Real seal E.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionOperator}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy equality, a selected representative, a chosen limit point, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6978.2 (RegularCauchyCompletionOperator NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$ *carrier* $O = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComple}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure through* D*, finite-window exposure through* S*, regular rational readback and regularity exposure through* R*, ordered* $\mathsf{Real}^{\uparrow}$ *seal access through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A completion-facing operator read must pass through those rows before it may cite the Real-facing seal.*

Proof. Project the accepted finite BHist packet. The visible carrier rows are precisely D, S, R, E, H, C, P, N, so carrier admission exposes tolerance, window, readback, terminal seal, transport, replay, provenance, and naming data.

The operator read is ordered. A precision request first reaches the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, then the $\mathsf{StreamName}^{\uparrow}$ finite windows, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback and regularity row, and only afterward the terminal $\mathsf{Real}^{\uparrow}$ seal.

Transport and replay are structural. The row H moves the listed coordinates componentwise by hsame, while C replays the displayed continuation route and P, N source and name the same packet. Since no quotient Cauchy equality, selected representative, chosen limit, countable-choice schedule, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness coordinate appears in the tuple, the listed rows exhaust the local NameCert obligations. □

Theorem 6978.3 (RegularCauchyCompletionOperator seal route)**.** *Every* $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$ *consumer that reads the Real-facing output of an accepted carrier* $O = (D, S, R, E, H, C, P, N)$ *factors through the finite route*

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the operator output reaches $\mathsf{Real}^{\uparrow}$ *only through DyadicRatCore tolerances, finite StreamName windows, RegSeqRat readback, the displayed seal row, componentwise transport, continuation replay, package provenance, and local NameCert data. It does not export quotient Cauchy equality, selected representatives, selected completion points, countable choice, host real equality, or any claim that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with a consumer request for the regular completion output. The carrier definition gives the request access to the dyadic tolerance row D and the finite window row S, so the request begins with displayed Cauchy-name observations rather than an external quotient.

Next read the regular row. The readback row R supplies the regular rational data attached to the same windows and tolerance ledger, so the terminal seal E is reached only after the regularity evidence has been displayed.

Finally replay the structural rows H, C, P, N. The route is therefore the displayed finite chain from dyadic tolerances and stream windows to the regular readback and Real seal. Because the packet contains no additional coordinate, the factorization cannot produce quotient Cauchy equality, selected representatives, selected completion points, countable choice, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionOperator}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName windows, RegSeqRat regular readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6978.1, Definition 6978.2, and Definition 6978.3 seed the finite regular Cauchy completion-operator route from dyadic tolerances and finite windows to the Real-facing seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionOperator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected representatives, selected completion points, countable choice, host real equality, Lean verification, bridge checking, or any assertion that Real$^{\uparrow}$ is complete.

Upgrade path: Obligation closure requires checked carrier admission, regular readback ordering, Real-seal factorization, transport stability, and non-escape for BEDC.Derived.RegularCauchyCompletionOperatorUp.

Chapter 6979

A Concrete Naming Certificate for $\mathsf{CauchyRepletion}^{\uparrow}$

Following Bishop and Bridges on constructive completions by regular Cauchy data, the $\mathsf{CauchyRepletion}^{\uparrow}$ packet names the finite repletion corridor between displayed stream observations and the $\mathsf{Real}^{\uparrow}$ seal. It sits beside the finite windows of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, the completion-admission surface of chapter 1476, and the real-name seal of chapter 38. The packet is a non-quotient repletion object for L10 completion consumers; it does not import selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Repeated repletion reads consume the $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ route of chapter 5119 before the $\mathsf{Real}^{\uparrow}$ seal, so the repletion corridor does not create a second completed object.

Definition 6979.1 (Cauchy repletion carrier)**.** *A* $\mathsf{CauchyRepletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, R_0, D, Q, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule through which the repletion request reads stream data.*
2. *R_0 is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording tolerance scales, comparison bounds, and dyadic threshold data.*
4. *Q is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *admission row, read only after S, R_0, D have been displayed.*
5. *U is the separated comparison row, recording the finite apartness and equality-refusal data used before any seal-facing comparison is exported.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the criterion and separated comparison rows.*
7. *H stores componentwise* hsame *transport for the stream schedule, readback, dyadic ledger, criterion row, separated comparison, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from finite windows to regular readback, from readback through the dyadic and criterion rows, and from the separated comparison row to the seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRepletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRepletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R_0, D, Q, U, E, H, C, P, N$. It has no coordinate for quotient streams, selected host limits, countable-choice schedules, host real equality, or ambient completeness.

Theorem 6979.2 (Cauchy repletion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRepletion}^{\uparrow}$ *carrier $R = (S, R_0, D, Q, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRepletion}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window readback, dyadic tolerance routing, completion-criterion admission, separated comparison, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming. Every repletion consumer must display the* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R_0,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *row Q, separated comparison row U,* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N before it reads the packet as a completion neighbour.*

Proof. Project the carrier of Definition 6979.1. The ten displayed rows are the only classifier-visible coordinates, so carrier admission and local naming are immediate row projections.

The analytic obligations are also row projections. The stream schedule S precedes the regular readback R_0, the dyadic ledger D records the tolerance data used by that readback, and the criterion row Q is the only completion-admission coordinate available before a seal-facing read.

The separated comparison row U and seal row E complete the finite route. Transport H, replay C, and provenance P preserve the same packet, while the absence of quotient, selected-limit, countable-choice, host-equality, and ambient-completeness coordinates gives the non-escape clause. □

Falsifiable prediction: Every accepted $\mathsf{CauchyRepletion}^{\uparrow}$ packet exposes StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, CauchyCompletionCriterion admission, separated comparison, Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any repletion consumer reads it.

Independence witness: streamname, regseqrat, dyadicratcore, cauchycompletioncriterion, real: $\mathsf{CauchyRepletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the window schedule, regular readback, dyadic tolerance ledger, completion criterion, separated comparison row, terminal seal, transport, replay, provenance, and local naming as one finite repletion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRepletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRepletion}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyCompletionCriterion admission row, separated comparison data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRepletion}^{\uparrow}}$, seed)

Scope closed: Definition 6979.1 and Definition 6979.2 seed the finite non-quotient repletion surface for completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRepletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, dyadic tolerance routing, completion-criterion admission, separated comparison, Real seal exposure, transport, replay, provenance, and non-escape for BEDC.Derived.CauchyRepletionUp.

Chapter 6980

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$

The $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ packet realizes the observer-expansion branch of chapter 9463 as finite boundary-tangent phase transport. It sits beside the horocycle packet of chapter 3534, the phase-ledger horocycle of chapter 3591, the boundary-fixed transport surface of chapter 3500, the observer-expansion route of chapter 4167, and the metric readback of chapter 3474. The packet records a finite phase-horocycle carrier; it does not assert an ambient hyperbolic manifold, a selected boundary limit, a global geodesic flow, or host equality of boundary points.

Definition 6980.1 (Hyperbolic phase horocycle carrier)**.** *A* $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, T, M, O, F, H_0, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-tangent row read through the displayed Poincare boundary data.*
2. *T is the phase-transport row carrying the local phase ledger along the horocycle.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *readback row used to compare the transported phase position.*
4. *O is the observer-expansion row recording the focused expansion ledger consumed by the packet.*
5. *F is the refusal row excluding selected boundary limits, ambient geodesic flow, host point equality, and global hyperbolic manifold structure.*
6. *H_0 stores componentwise* hsame *transport for the boundary-tangent, phase-transport, metric, observer-expansion, refusal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from boundary-tangent data through phase transport and metric readback to the observer-expansion row.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *with sibling pointers to the horocycle, phase-ledger, boundary-fixed transport, and observer-expansion packets.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, M, O, F, H_0, C, P, N$. It has no coordinate for an ambient manifold, selected far-end limit, global flow, quotient boundary equality, or unlogged observer expansion.

Theorem 6980.2 (Hyperbolic phase horocycle NameCert obligations)**.** *For every accepted* HyperbolicPhaseHorocyc *carrier* $H = (B, T, M, O, F, H_0, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}}$ *surface consists exactly of carrier admission, boundary-tangent exposure, phase-transport replay, hyperbolic-metric readback, observer-expansion routing, refusal coverage, componentwise transport, provenance, and local naming. Any phase-horocycle consumer must display* $B, T, M, O, F, H_0, C, P, N$ *before it reads the packet.*

Proof. Project the carrier of Definition 6980.1. The displayed rows are the only classifier-visible coordinates, so carrier admission and local naming are immediate row projections.

The boundary and phase obligations are read from B and T. The metric row M supplies the hyperbolic readback for the same transported phase surface, and the observer-expansion row O records the focused ledger consumed by that readback.

The refusal row F excludes the ambient objects not carried by the packet. Transport H_0, replay C, and provenance P preserve the same finite route, so the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ packet exposes boundary-tangent, phase-transport, hyperbolic-metric, observer-expansion, refusal, transport, replay, provenance, and NameCert rows before any phase-horocycle consumer reads it.

Independence witness: hyperbolic-horocycle, phase-ledger-horocycle, poincare-boundary-fixed-transport, hyperbolic-observer-expansion, hyperbolic-metric: $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds boundary-tangent phase transport, metric readback, observer-expansion routing, refusal, transport, replay, provenance, and local naming in one packet.

Lean follow-up requires Nontrivial $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ witnesses in the corresponding TasteGate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ packet is generated from boundary-tangent data, phase transport, hyperbolic metric readback, observer-expansion routing, refusal coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}}$, seed)

Scope closed: Definition 6980.1 and Definition 6980.2 seed the finite boundary-tangent phase-horocycle surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient hyperbolic manifolds, selected boundary limits, global geodesic flows, host boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-tangent exposure, phase-transport replay, hyperbolic-metric readback, observer-expansion routing, refusal coverage, transport, provenance, and non-escape for BEDC.Derived.HyperbolicPhaseHorocycleUp.

Chapter 6981

A Concrete Naming Certificate for $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed Cauchy modulus can be replaced by a dominating displayed modulus along the same Real and Completion route. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the base modulus schedule of chapter 292, the completion-admission surface of chapter 1476, and the terminal Real seal of chapter 38. The packet is a finite dominance criterion for Real and Completion consumers; it is not a quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy split, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 6981.1 (Cauchy modulus dominance criterion carrier)**.** *A* $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, M_0, M_1, B, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the observations consumed by the base Cauchy read.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic precision requests shown to the consumer.*
4. *M_0 is the base* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning a finite tail threshold to each displayed dyadic request.*
5. *M_1 is the dominating* $\mathsf{CauchyModulus}^{\uparrow}$ *row for the same displayed requests, with each shown threshold at least as far out as the corresponding threshold in M_0.*
6. *B is the local dominance-criterion row, recording that M_1 dominates M_0 on exactly the dyadic requests displayed by D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the same Real and Completion route after S, Q, D, M_0, M_1, B have been displayed.*
8. *H records componentwise* hsame *transport for the windows, readback, dyadic requests, both modulus rows, dominance row, seal row, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay from each finite dyadic request through*

$$D, M_0, M_1, B, S, Q \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* CauchyModulusDominanceCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyModulus$^{\uparrow}$, CauchyCompletionCriterion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyModulusDominanceCriterion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected tail functions, selected limits, countable choice, host sequence equality, host real equality, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6981.2 (Cauchy modulus dominance criterion NameCert obligations)**.** *For every accepted* CauchyModulusDominanceCriterion$^{\uparrow}$ *carrier*

$$K = (S, Q, D, M_0, M_1, B, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusDominanceCriterion}^{\uparrow}}$ *surface consists exactly of carrier exposure for the displayed packet, finite-window exposure through S, regular rational readback through Q, dyadic tolerance exposure through D, base-modulus exposure through M_0, dominating-modulus exposure through M_1, dominance exposure through B, terminal* Real$^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier in Definition 6981.1. Each obligation names one coordinate of the same finite BHist packet, and the local naming row N records that these coordinates form the certificate surface.

The analytic part is read in the displayed order. A dyadic request first enters the ledger D, then reads the base threshold M_0 and the dominating threshold M_1, and then checks the dominance row B for that displayed request. The stream windows S and regular readback Q are therefore attached to the same finite request route.

The structural rows H, C, P, N only transport, replay, source, and name this route. Since the carrier has no coordinate for quotient stream equality, selected tails, selected limits, countable choice, host equality, trichotomy, or ambient completeness, none of those objects can be exported by the local NameCert obligations. □

Theorem 6981.3 (Cauchy modulus dominance criterion Real-seal route)**.** *Every* Real$^{\uparrow}$*-facing or completion-facing consumer of an accepted* CauchyModulusDominanceCriterion$^{\uparrow}$ *carrier*

$$K = (S, Q, D, M_0, M_1, B, E, H, C, P, N)$$

may replace the displayed base modulus row M_0 by the displayed dominating row M_1 only through the same finite route

$$\text{StreamName}^{\uparrow} \longmapsto \text{RegSeqRat}^{\uparrow} \longmapsto \text{DyadicRatCore}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{CauchyCompletionCriterion}^{\uparrow} \longmapsto \text{Real}^{\uparrow}$$

The replacement preserves the terminal seal row E and exports no quotient stream row, selected limit row, countable-choice row, host equality row, trichotomy row, or ambient Real$^{\uparrow}$ *completeness row.*

Proof. Start with the NameCert obligations of Definition 6981.2. They expose the base modulus, dominating modulus, and dominance row only after the same dyadic request ledger and before the same terminal seal row.

The dominance condition is local to displayed requests: M_1 dominates M_0 only on the requests present in D. Thus replacing M_0 by M_1 cannot introduce a hidden schedule, selected tail, or quotient stream comparison; it merely moves the finite threshold farther along the already displayed route.

Finally read the provenance and replay rows. They attach the dominance route to the completion-admission surface and the terminal Real seal without adding another carrier coordinate. Hence the Real seal route remains the one displayed in the theorem statement, and all excluded host or ambient principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$ Real-facing read factors through displayed StreamName windows, RegSeqRat readback, DyadicRatCore tolerance requests, a base CauchyModulus row, a dominating CauchyModulus row, a dominance-criterion row, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchymodulus, cauchycompletioncriterion, real: $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one base modulus row, one dominating modulus row, the displayed requestwise dominance criterion, the shared Real and Completion route, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a base CauchyModulus row, a dominating CauchyModulus row, a requestwise dominance row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6981.1, Definition 6981.2, and Definition 6981.3 seed the finite criterion by which a displayed Real and Completion route may replace a base CauchyModulus row by a dominating CauchyModulus row on displayed dyadic requests.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusDominanceCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, requestwise dominance, NameCert exposure, and Real-seal nonescape for BEDC.Derived.CauchyModulusDominanceCriterionUp.

Chapter 6982

A Concrete Naming Certificate for $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$

Following Birkhoff and von Neumann, the $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ packet records the finite orbit-average surface that precedes any pointwise or mean ergodic convergence theorem. It sits between the measure-preserving source of chapter 196, the displayed transition rows of chapter 195, the probability-space ledger of chapter 184, and the terminal real-name seal of chapter 38. The packet gives later mean-ergodic and recurrence-facing chapters a finite average row without importing measure-theoretic choice principles.

Definition 6982.1 (Birkhoff ergodic finite-average carrier)**.** *A* $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (E, D, P, W, O, S, R, A, H, C, Q, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Ergodic}^{\uparrow}$ *source row, exposing the measure-preserving transformation, invariant-row ledger, and finite measure endpoint data consumed by the average.*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *step row for the same transformation, recording the displayed source point, successor route, and finite iterate endpoints.*
3. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *averaging provenance row, recording the unit-mass probability source and the event or observable window through which the average is read.*
4. *W is the finite orbit-window row, storing the inspected segment $x, Tx, \ldots, T^{n-1}x$ together with the visible window length n.*
5. *O is the finite observation row, recording the observable values attached to the window entries and read only through* $\mathsf{StreamName}^{\uparrow}$ *finite observations.*
6. *S is the finite sum and denominator row used to form the displayed average of O over W.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the rational regular-name route for the average value before the terminal real seal is read.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *averaged endpoint seal, downstream of E, D, P, W, O, S, R and never a selected almost-everywhere limit.*

9. *H records componentwise* hsame *transport for the ergodic source, dynamic steps, probability provenance, orbit window, observations, finite sum, rational readback, real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite averaging request through $E, D, P, W, O, S, R, A, H, Q, N$.*
11. *Q is the* Pkg *provenance over* $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$, $\mathsf{Ergodic}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, P, W, O, S, R, A, H, C, Q, N$. It has no coordinate for the full almost-everywhere Birkhoff theorem, invariant sigma-algebra projection, countable choice, quotient a.e. equality, selected orbit limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6982.2 (Birkhoff finite-average NameCert obligations)**.** *For every accepted* $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ *carrier $F = (E, D, P, W, O, S, R, A, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}}$ *surface consists exactly of carrier admission, measure-preserving source exposure, dynamic-step window exactness, probability-provenance visibility, finite StreamName observation routing, finite sum and denominator exactness, RegSeqRat readback, Real seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 6982.1.*

Proof. Carrier admission is projection from Definition 6982.1. The accepted packet exposes exactly the ergodic source, dynamic step row, probability provenance row, finite orbit window, observation row, finite sum row, rational readback row, Real seal, transport, replay, provenance, and local name.

Window exactness is read from D, W, O. The dynamic row supplies the inspected iterates, the window row fixes the finite prefix, and the observation row records only the StreamName-visible values attached to that prefix.

The averaging obligation is read from P, S, R, A. The probability-space provenance fixes the averaging source, the finite sum and denominator row computes the displayed average, the RegSeqRat row exposes the rational regular-name readback, and the Real seal is visible only after those rows have been displayed.

The structural obligations are exhausted by H, C, Q, N. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same finite route and add no convergence theorem, invariant sigma-algebra projection, quotient a.e. equality, countable-choice schedule, selected limit, or ambient Real completeness coordinate. □

Theorem 6982.3 (Birkhoff finite-average readback route)**.** *Every average exported by an accepted* $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Ergodic}^{\uparrow},\ \mathsf{DynSystem}^{\uparrow},\ \mathsf{ProbSpace}^{\uparrow} \longmapsto W, O, S \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a downstream mean-ergodic or recurrence-facing consumer receives a displayed orbit window, finite observation ledger, finite average row, regular rational readback, and terminal real-name seal before it can consume the average.

Proof. Start with an averaging request. The request first opens the $\mathsf{Ergodic}^{\uparrow}$ and $\mathsf{DynSystem}^{\uparrow}$ rows, so the transformation and its finite iterates are read from displayed BEDC rows rather than from a host measurable dynamical system.

Next read the $\mathsf{ProbSpace}^{\uparrow}$ provenance and the window rows W, O, S. These rows fix the probability source, the inspected orbit segment, the observed values, and the finite sum with its denominator.

The average then passes through R before A is visible. This forces the regular rational readback to appear before any $\mathsf{Real}^{\uparrow}$ seal is consumed, so the exported endpoint is a finite NameCert route rather than a hidden limit.

Finally transport and replay through H, C, Q, N. Since the carrier has no coordinate for the full almost-everywhere Birkhoff theorem, invariant sigma-algebra projection, countable choice, quotient a.e. equality, selected orbit limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported average follows the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ average read factors through one ErgodicUp source row, one DynSystemUp finite orbit-window row, one ProbSpaceUp averaging provenance row, finite StreamNameUp observations, a finite sum and denominator row, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ergodic, dynsystem, probspace, streamname, regseqrat, real: BirkhoffErgodicFiniteAverag is not bijective to any listed sibling because it jointly carries a measure-preserving source, dynamic orbit windows, probability provenance, finite observations, finite average rows, rational readback, Real sealing, transport, replay, provenance, and local naming while excluding convergence, quotient, choice, and completeness coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$ packet is generated from ErgodicUp measure-preserving source rows, DynSystemUp finite orbit-window rows, ProbSpaceUp averaging provenance rows, StreamNameUp finite observation rows, RegSeqRatUp readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6982.1, Definition 6982.2, and Definition 6982.3 seed the finite orbit-average carrier, its local NameCert obligation surface, and its readback route through ErgodicUp, DynSystemUp, ProbSpaceUp, StreamNameUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full almost-everywhere Birkhoff theorem, invariant sigma-algebra projection, countable choice, quotient a.e. equality, selected orbit limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dynamic-window exactness, probability-provenance visibility, finite observation routing, finite average exactness, RegSeqRat readback, Real seal ordering, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.BirkhoffErgodicFiniteAverageUp.

Chapter 6983

A Concrete Naming Certificate for $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ packet records the finite selector route by which a displayed nested closed interval argument is read as regular Cauchy data before a terminal real-name seal is opened. It sits after the nested closed-interval surface of chapter 4012, the dyadic endpoint ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the Cauchy naming surface of chapter 3413, and the terminal real-name seal of chapter 38. The packet names only the finite selector and handoff rows; it does not assert quotient real equality, a selected limit, countable choice, trichotomy, Heine–Borel compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6983.1 (Nested closed interval selector carrier)**.** *A* $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested closed-interval row, read through* $\mathsf{ClosedIntervalNesting}^{\uparrow}$*, carrying the inspected closed interval stages and successor-containment evidence.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger, recording endpoint arithmetic, displayed width budgets, and the finite stage bound at which the inspected closed interval is narrower than the requested tolerance.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the interval stages, endpoints, midpoint or endpoint selector values, and shrink witnesses inspected by the selector.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selector values in W after the shrink ledger D has fixed the tolerance request.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the nested interval row, shrink ledger, finite windows, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the nested interval row, shrink ledger, finite windows, readback, seal, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-facing tolerance request through*

$$I, D, W, R \longmapsto E, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$, $\mathsf{ClosedIntervalNesting}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedClosedIntervalSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, W, R, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected completed real, countable choice, trichotomy, Heine–Borel compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6983.2 (Nested closed interval selector NameCert obligations)**.** *For every accepted* $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ *carrier*

$$K = (I, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedClosedIntervalSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, nested closed-interval exposure through I, dyadic shrink-ledger exactness through D, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6983.1. The nested interval source, shrink ledger, finite windows, regular readback, terminal seal, and structural rows are exactly $I, D, W, R, E, H, C, P, N$.

The interval obligations are read first. The row I exposes only inspected closed interval stages and successor containment, while D attaches the dyadic endpoint arithmetic and width budget to those same stages.

The finite-window obligations are then forced by W. A selector value is visible only when its interval stage, endpoints, and shrink witness are all present in the displayed window, and the readback row R is downstream of that window.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same route. Since no carrier coordinate stores quotient real equality, a selected completed real, countable choice, trichotomy, compactness, or ambient completeness, those principles cannot appear among the local NameCert obligations. □

Theorem 6983.3 (Nested closed interval selector Cauchy handoff)**.** *Every Cauchy-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ *carrier*

$$K = (I, D, W, R, E, H, C, P, N)$$

reaches a regular rational Cauchy read only through the finite route

$$\mathsf{ClosedIntervalNesting}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For each displayed tolerance request, the shrink ledger supplies an inspected stage whose closed interval is narrower than that tolerance, the window row exposes the selector value at that stage, and the readback row packages those values as the regular Cauchy handoff before the terminal seal is consumed.

Proof. Start with a displayed tolerance request. The shrink ledger D gives a finite stage bound and a dyadic width inequality for the closed interval stage exposed by I.

Induct on the finite window prefix in W. The empty prefix exposes no selector value. In the successor step, the newly inspected stage carries its endpoints, containment witness, shrink

witness, and selector value; successor containment keeps later inspected selector values inside the earlier displayed closed interval.

When two inspected stages are beyond the displayed bound for a tolerance, both selector values lie in a common closed interval whose dyadic width is below that tolerance. The row R therefore reads the selector stream as regular rational Cauchy data without appealing to an unlisted limit object.

The terminal row E is opened only after R. The transport, replay, provenance, and local naming rows preserve this order, so the handoff cannot export quotient equality, countable choice, trichotomy, Heine–Borel compactness, a selected completed real, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ Cauchy-facing read factors through a ClosedIntervalNesting row, DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: closed-interval-nesting, dyadicratcore, streamname, regseqrat, cauchy, real: $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nested closed-interval row, dyadic shrink ledger, finite selector windows, regular Cauchy readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedClosedIntervalSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedClosedIntervalSelector}^{\uparrow}$ packet is generated from a ClosedIntervalNesting row, DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedClosedIntervalSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6983.1, Definition 6983.2, and Definition 6983.3 seed the finite selector route from nested closed interval rows through shrink-ledger Cauchy readback and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedClosedIntervalSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, Heine–Borel compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested closed-interval exposure, shrink-ledger exactness, finite-window induction, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedClosedIntervalSelectorUp.

Chapter 6984

A Concrete Naming Certificate for $\mathsf{SlaterCondition}^{\uparrow}$

Following Slater's constraint qualification as presented in standard convex optimization texts, the $\mathsf{SlaterCondition}^{\uparrow}$ packet records strict feasibility as a finite NameCert surface for convex programs. It sits beside the stationarity and complementary-slackness rows of chapter 300, the separation boundary of chapter 6185, and the cone carrier of chapter 310. The chapter names the strict-feasibility interface only; it does not assert strong duality, existence of an optimizer, differentiability, or any host-level constraint qualification beyond the displayed rows.

Definition 6984.1 (Slater condition carrier)**.** *A* $\mathsf{SlaterCondition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, M, B, K, H, C, P, N)$$

with the following visible rows:

1. *F is the convex feasible source, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and the displayed constraint rows of the surrounding optimization packet.*
2. *W is a strict interior witness row, giving one carried feasible point together with the finite strict-inequality comparisons used by the classifier.*
3. *M is the finite margin ledger recording the positive slack, cone-interior, or barrier-side comparisons that make W strict.*
4. *B is the* $\mathsf{FarkasLemma}^{\uparrow}$ *separation-boundary row that records which finite alternative is blocked by the displayed strict margins.*
5. *K is the* $\mathsf{KKT}^{\uparrow}$ *multiplier handoff row, exposing only the multiplier, residual, stationarity, feasibility, and complementary-slackness slots that are allowed to consume strict feasibility.*
6. *H stores componentwise* hsame *transport for the feasible source, strict witness, margin ledger, separation boundary, multiplier handoff, continuation, provenance, and local naming rows.*
7. *C records the finite* Cont *replay that routes strict feasibility through F, W, M, B, K without adding a solver or analytic duality theorem.*
8. *P is the* Pkg *provenance over* $\mathsf{SlaterCondition}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{FarkasLemma}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, $\mathsf{SemidefiniteCone}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SlaterCondition}^{\uparrow}}$ *row.*

The carrier has no coordinate for strong duality, selected primal or dual optimizers, differentiability, closedness of a host cone, or a host real-valued minimization semantics.

Theorem 6984.2 (Slater condition NameCert obligations)**.** *For every accepted* $\mathsf{SlaterCondition}^{\uparrow}$ *packet* $S = (F, W, M, B, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SlaterCondition}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, strict interior exposure through* W*, finite margin accounting through* M*, separation-boundary compatibility through* B*, KKT multiplier handoff through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Every obligation is read through the visible dependencies $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{FarkasLemma}^{\uparrow}$*,* $\mathsf{KKT}^{\uparrow}$*, and* $\mathsf{SemidefiniteCone}^{\uparrow}$*. No obligation asserts strong duality beyond the strict-feasibility handoff described by the packet.*

Proof. Project the carrier of Definition 6984.1. The feasible source, strict witness, margin ledger, separation boundary, KKT handoff, transport, replay, provenance, and local name are exactly the displayed rows.

The strict witness row W is usable only with the finite margin ledger M. Thus the classifier sees strict feasibility as carried slack data, not as an ambient interior predicate. The $\mathsf{SemidefiniteCone}^{\uparrow}$ dependency contributes only the cone-facing row when the surrounding program is semidefinite; it does not add spectral closure or host interior semantics.

The separation row B is read through chapter 6185. It records the finite alternative boundary blocked by the displayed margins. The handoff row K is read through chapter 300; it exposes only the multiplier and stationarity-facing slots that consume strict feasibility.

Finally, H, C, P, N transport, replay, package, and name the same route. Since none of the carrier rows stores strong duality, a selected optimizer, differentiability, or host minimization semantics, the listed obligations exhaust the NameCert surface. □

Theorem 6984.3 (Slater condition KKT handoff)**.** *Let* $S = (F, W, M, B, K, H, C, P, N)$ *be an accepted* $\mathsf{SlaterCondition}^{\uparrow}$ *carrier. Any downstream* $\mathsf{KKT}^{\uparrow}$ *consumer that uses the strict-feasibility row of* S *receives only the finite handoff*

$$F \longmapsto W \longmapsto M \longmapsto B \longmapsto K.$$

The row B *keeps the* $\mathsf{FarkasLemma}^{\uparrow}$ *separation boundary explicit, and the provenance row may cite* $\mathsf{SemidefiniteCone}^{\uparrow}$ *when the feasible source is cone-valued. The handoff supplies no strong-duality theorem, no selected optimizer, no differentiability oracle, and no host constraint qualification beyond the displayed strict-feasibility packet.*

Proof. Begin with the accepted carrier in Definition 6984.1. The only strict-feasibility data available to a consumer are the convex feasible source F, the strict witness W, and the margin ledger M.

Next read the separation boundary. By the placement of B, any Farkas-style obstruction is compared against the same displayed margins, so the downstream route remains a finite alternative boundary rather than a topological separation theorem.

Now read the KKT handoff row K. The $\mathsf{KKT}^{\uparrow}$ consumer receives the multiplier-facing and stationarity-facing slots named by K, with no additional primal-dual conclusion. Componentwise hsame transport, Cont replay, Pkg provenance, and the local NameCert row preserve that route and do not add any coordinate outside F, W, M, B, K. □

Closure status for $\mathsf{NameCert}_{\mathsf{SlaterCondition}^{\uparrow}}$.

Constructive story: A $\mathsf{SlaterCondition}^{\uparrow}$ packet is generated from a convex feasible source, a strict interior witness, a finite margin ledger, a Farkas separation boundary, a KKT multiplier handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SlaterCondition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6984.1, Definition 6984.2, and Definition 6984.3 seed the strict-feasibility interface consumed by KKT and convex-duality packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SlaterCondition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim strong duality, selected primal or dual optimizers, differentiability, closedness of a host cone, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and prove the strict-feasibility handoff to the KKT multiplier surface.

Chapter 6985

A Concrete Naming Certificate for $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$

The $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ packet realizes the before-and-after disagreement surface of section 9462.2 as finite NameCert data. It sits beside the world-model classifier of chapter 6124, the black-box decomposition certificate of chapter 6100, the positive interpretability classifier of chapter 6202, and the audit membrane of chapter 524. The packet names a classifier-shift residue ledger; it is not record growth under an unchanged classifier and not a positive-discovery verdict by itself.

Definition 6985.1 (Classifier shift ledger residue carrier)**.** *A* $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, A, S, D, L, G, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the before-classifier row, naming the source scope, classifier relation, certificate state, and ledger policy before the inspected passage.*
2. *A is the after-classifier row, naming the corresponding source scope, classifier relation, certificate state, and ledger policy after the passage.*
3. *S is the source-overlap row, proving that B and A are compared over a displayed common source rather than over unrelated records.*
4. *D is the classifier-disagreement row, exposing a finite before-and-after classifier judgment that differs on the shared source.*
5. *L is the residue ledger, recording which rows are newly classified, which rows remain outside the after classifier, and which rows witness the disagreement.*
6. *G is the gap-detection row, separating record-only growth, unchanged classifier presentation, hidden source switch, and genuine classifier disagreement.*
7. *U is the discovery-status row, recording whether downstream consumers may treat the passage as candidate structural discovery, rejected record growth, or ledger-incomplete classifier shift.*
8. *H stores componentwise* hsame *transport for the before row, after row, source overlap, disagreement, residue ledger, gap detection, discovery status, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from source overlap through disagreement, residue accounting, gap detection, discovery status, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$, $\mathsf{WorldModelClassifier}^{\uparrow}$, BlackBoxDecompositionCertific PositiveInterpretabilityClassifier$^{\uparrow}$, $\mathsf{AuditMembrane}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}}$ *row.*

The carrier has no coordinate for record length alone, unchanged-classifier presentation, hidden source switching, unledgered discovery status, positive-discovery certification, or a host semantic novelty score.

Theorem 6985.2 (Classifier shift ledger residue NameCert obligations)**.** *For every accepted* ClassifierShiftLedgerRes *packet $R = (B, A, S, D, L, G, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}}$ *surface consists exactly of before-classifier admission, after-classifier admission, source-overlap admission, classifier-disagreement exposure, residue-ledger exposure, gap-detection exposure, discovery-status exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6985.1. The before and after classifier rows are B, A, the comparison rows are S, D, the ledger and verdict rows are L, G, U, and the structural rows are H, C, P, N.

The source-overlap row must be read before disagreement. This prevents a hidden source switch from masquerading as classifier shift, because the disagreement row D is licensed only on the shared source S.

The residue ledger L then records the rows that changed classifier status and the rows that remain outside the after classifier. The gap row G separates this finite residue from record-only growth or unchanged-classifier presentation, and U records the downstream discovery status permitted by that evidence.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since record length alone, hidden source switching, unledgered discovery status, positive-discovery certification, and host novelty scores have no carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 6985.3 (Classifier shift ledger residue gap detection)**.** *An accepted* $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ *packet rejects any record-only passage whose classifier row is unchanged on the displayed source overlap. If B and A agree as classifier relations over S, then no row of D, L, G, U, H, C, P, N can supply the missing classifier disagreement required by the carrier.*

Proof. Read S first. It fixes the common source on which the before and after classifier rows are compared.

If the classifier relations in B and A agree over S, the disagreement row D has no displayed pair on which to witness a changed classifier judgment. The residue ledger L may still record more material, but without D that material is record growth under the same classifier.

The gap-detection row G therefore classifies the passage as unchanged-classifier presentation or record-only growth, and U cannot promote it to classifier-shift discovery status. The structural rows preserve the same failed route and do not create a disagreement coordinate. □

Falsifiable prediction: A record-only passage with unchanged classifier relation over the displayed source overlap cannot satisfy the $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ carrier of Definition 6985.1.

Independence witness: world_model_classifier, black_box_decomposition_certificate, positive_interpretability_classifier, auditmembrane: none is bijective to $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$

because the packet jointly carries before and after classifier rows, source overlap, classifier disagreement, residue ledger, gap detection, discovery status, transport, replay, provenance, and local naming rows.

The Lean `TasteGate.lean` file for $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ will provide explicit Nontrivial ClassifierShiftLedge and FieldFaithful $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ instances for the displayed eleven-row carrier before any checked TasteGate target can serve as a closure witness.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ packet is generated from before and after classifier rows, a source-overlap row, a classifier-disagreement row, a residue ledger, gap detection, discovery status, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6985.1, Definition 6985.2, and Definition 6985.3 seed the finite before and after classifier-disagreement ledger for the classifier-shift information principle.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive-discovery certification, global interpretability quality, host semantic novelty scoring, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-overlap comparison, classifier-disagreement exposure, residue-ledger accounting, gap detection, discovery-status routing, transport, replay, provenance, local naming, Nontrivial, and FieldFaithful rows for BEDC.Derived.ClassifierShiftLedgerResidueUp.

Chapter 6986

A Concrete Naming Certificate for $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ packet records the finite interval-refinement tree used when compact interval work branches a cover or modulus request over nested dyadic subintervals. It sits after the compact interval modulus route of chapter 3868, the dyadic interval arithmetic of chapter 1645, and the linear interval-chain boundary of chapter 6443. The packet names a finite tree carrier for refinement reasoning; it is not open-cover compactness, a selected subcover, a fan theorem, countable choice, quotient endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6986.1 (Bishop interval-refinement tree carrier)**.** *A* $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, E, D, B, L, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the root compact-interval request read through* $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$.
2. *E stores the dyadic endpoint enclosure and containment data supplied by* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$.
3. *D is the finite descendant ledger, listing each split interval, its parent, and its endpoint refinement witnesses.*
4. *B is the branch-frontier row naming the currently inspected leaves and their cover or modulus obligations.*
5. *L is the sibling* $\mathsf{BishopFiniteIntervalChain}^{\uparrow}$ *row obtained by reading any chosen level of leaves as a finite adjacent chain.*
6. *A records acceptance of each refinement step by local interval containment and leaf-frontier coverage.*
7. *H stores componentwise* hsame *transport for R, E, D, B, L, A and the structural rows.*
8. *C stores displayed* Cont *replay from the compact request through dyadic splits, branch frontiers, and level-chain reads.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$, $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$, $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{BishopFiniteIntervalChain}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalRefinementTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an infinite refinement tree, selected branch, open-cover subcover, fan principle, quotient endpoint equality, countable choice, or host compactness theorem.

Theorem 6986.2 (Bishop interval-refinement tree NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ *carrier* $T = (R, E, D, B, L, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalRefinementTree}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-root exposure, dyadic endpoint-refinement exactness, finite descendant-ledger stability, leaf-frontier coverage, level-chain readback through* $\mathsf{BishopFiniteIntervalChain}^{\uparrow}$*, refinement-step acceptance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The root row R is the only compact-interval request, the endpoint row E is the only dyadic arithmetic source, and the descendant ledger D stores every split relation inspected by the packet.

The branch-frontier and chain rows give the consumer-facing tree shape. The row B names the visible leaves, while L reads any displayed level as the adjacent finite-chain surface already provided by chapter 6443.

The acceptance and structural rows then exhaust the certificate. The row A records local containment and coverage for each refinement step, and H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is present, the listed obligations are exactly the local NameCert surface. □

Theorem 6986.3 (Bishop interval-refinement tree refinement non-escape)**.** *Every accepted* $\mathsf{BishopIntervalRefinemen}$ *refinement read factors through*

$$\mathsf{CompactIntervalModulusBridge}^{\uparrow}, \mathsf{DyadicIntervalArithmetic}^{\uparrow}, \mathsf{BishopFiniteIntervalChain}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Name}$$

For a carrier $T = (R, E, D, B, L, A, H, C, P, N)$*, a consumer may inspect only the compact root request, dyadic endpoint refinements, finite descendant ledger, visible leaf frontier, level-chain readback, refinement-step acceptance, transport, replay, provenance, and local name. No consumer receives an infinite tree, selected branch, open-cover subcover, fan theorem, countable-choice schedule, quotient endpoint equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the carrier inventory of Definition 6986.1. The rows R, E, D, B, L, A are the complete finite refinement data, and the rows H, C, P, N are the complete structural data.

Apply the obligation surface of Definition 6986.2. It forces every refinement read through the compact root row, dyadic endpoint refinements, finite descendant ledger, leaf frontier, level-chain readback, and local acceptance row before the packet can be consumed.

The three cited sibling chapters supply only the displayed services: compact modulus bridging, dyadic interval arithmetic, and finite interval chains. None supplies an infinite branch selector, an open-cover subcover, or a host compactness object.

Thus the accepted refinement read remains inside $R, E, D, B, L, A, H, C, P, N$. Transport and replay preserve this finite route, provenance records it, and the local name does not add any unrecorded source. □

Falsifiable prediction: Every accepted $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ read factors through a compact interval modulus root, dyadic endpoint-refinement rows, a finite descendant ledger, a leaf-frontier row, a finite interval-chain readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact-interval-modulus-bridge, dyadic-interval-arithmetic, bishop-finite-interval-chain: none is bijective because $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ binds the compact request, dyadic endpoint refinements, finite descendant tree ledger, leaf-frontier coverage, level-chain readback, refinement acceptance, transport, replay, provenance, and local NameCert rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalRefinementTree}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalRefinementTree}^{\uparrow}$ packet is generated from a compact interval modulus root, dyadic endpoint-refinement rows, a finite descendant ledger, leaf-frontier rows, finite interval-chain readback, refinement-step acceptance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalRefinementTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6986.1, Definition 6986.2, and Definition 6986.3 seed the finite interval-refinement tree carrier and its non-escape route through compact modulus, dyadic interval arithmetic, finite interval chains, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalRefinementTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert open-cover compactness, a selected subcover, a fan theorem, countable choice, quotient endpoint equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint-refinement exactness, descendant-ledger stability, leaf-frontier coverage, level-chain readback, refinement-step acceptance, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalRefinementTreeUp.

Chapter 6987

A Concrete Naming Certificate for $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$

Following Bishop and Bridges on located constructive real names and product stability of regular Cauchy data, the $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$ packet records the finite guard that lets a product read proceed only after both factor names expose located dyadic tolerance windows. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the product packet of chapter 464, and the regular product route of chapter 461. The packet gives later Real algebra and completion consumers a non-quotient product-stability guard over displayed rows only.

Definition 6987.1 (Located Cauchy product guard carrier)**.** *A* $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, R, W_L, W_R, D_L, D_R, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two accepted regular Cauchy source rows whose product is being guarded.*
2. *W_L and W_R are* $\mathsf{StreamName}^{\uparrow}$ *finite windows exposing the factor observations read for the current product request.*
3. *D_L and D_R are located* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-window rows. Each row names a dyadic radius, a finite inspection bound, and the located apartness or enclosure evidence needed before multiplication is read.*
4. *Q is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product-source row assembled from the two windows and tolerance rows.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *handoff row, available only after Q, D_L, D_R, W_L, W_R are displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through the regular rational readback, not by a selected product limit.*
7. *H records componentwise* hsame *transport for the source rows, windows, dyadic tolerance rows, product row, regular handoff, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each product-stability request through*

$$L, R, W_L, W_R, D_L, D_R, Q, M \longmapsto E, H, P, N.$$

9. *P is the* Pkg *provenance over* LocatedCauchyProductGuard$^\uparrow$, CauchyProduct$^\uparrow$, RegularCauchyProduct$^\uparrow$, DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LocatedCauchyProductGuard}^\uparrow}$ *row.*

The classifier compares accepted packets only through $L, R, W_L, W_R, D_L, D_R, Q, M, E, H, C, P, N$. It has no coordinate for Real completeness, quotient equality of names, a selected product limit, trichotomy, countable choice, or host multiplication.

Theorem 6987.2 (Located Cauchy product guard NameCert obligations)**.** *For every accepted* LocatedCauchyProductGuard$^\uparrow$ *carrier*

$$G = (L, R, W_L, W_R, D_L, D_R, Q, M, E, H, C, P, N),$$

the local NameCert$_{\text{LocatedCauchyProductGuard}^\uparrow}$ *surface consists exactly of carrier admission for the two regular Cauchy source rows, finite* StreamName$^\uparrow$ *window admission for both factors, located* DyadicRatCore$^\uparrow$ *tolerance-window exactness,* CauchyProduct$^\uparrow$ *source assembly,* RegularCauchyProduct$^\uparrow$ *handoff, terminal* Real$^\uparrow$ *sealing through regular readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6987.1. The only rows available to a certificate checker are the two source rows, the two finite windows, the two located dyadic tolerance rows, the product row, the regular handoff, the terminal seal, and the structural rows.

The located tolerance rows are read before the product row. Thus Q cannot be consumed until D_L, D_R, W_L, W_R have fixed the finite dyadic windows for the current request.

The regular product handoff M is downstream of Q. The rows H, C, P, N transport, replay, source, and name that same finite route, so the listed rows exhaust the local certificate surface without adding completion, quotient, selected-limit, choice, trichotomy, or host multiplication data. □

Theorem 6987.3 (Located Cauchy product guard regular product handoff)**.** *Every* RegularCauchyProduct$^\uparrow$ *or* Real$^\uparrow$ *consumer of an accepted* LocatedCauchyProductGuard$^\uparrow$ *packet must first expose the finite route*

$$\text{StreamName}^\uparrow,\quad \text{DyadicRatCore}^\uparrow,\quad \text{CauchyProduct}^\uparrow,\quad \text{RegSeqRat}^\uparrow,\quad \text{RegularCauchyProduct}^\uparrow,\quad \text{Real}^\uparrow.$$

For each displayed product request, the guard reads both factor windows, matches them with located dyadic tolerance windows, assembles the CauchyProduct$^\uparrow$ *row, and only then releases the regular product handoff and terminal seal.*

Proof. Start with a product-facing read. Carrier admission exposes the factor rows and finite windows, but neither M nor E is available before the located tolerance rows have been displayed.

Read the dyadic guard. The rows D_L and D_R bind the requested product tolerance to concrete dyadic windows on the two factor names. This is a finite BHist read, so no selected limit or ambient completeness principle enters the route.

Now consume Q and M. The product row is assembled from the displayed factor windows, and the regular handoff is downstream of that assembly. Componentwise hsame transport and displayed Cont replay preserve the same row order for every accepted packet.

The terminal Real$^\uparrow$ seal is therefore a downstream read of the regular product handoff. Since the packet never stores quotient equality of names, host multiplication, countable choice, trichotomy, or Real completeness, none of those objects can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$ product-stability read factors through two regular Cauchy source rows, paired StreamName windows, located DyadicRatCore tolerance windows, a CauchyProduct row, RegSeqRat readback, a RegularCauchyProduct handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cauchyproduct, regularcauchyproduct, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two located factor windows, dyadic tolerance guards, product-source assembly, regular product handoff, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyProductGuard}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyProductGuard}^{\uparrow}$ packet is generated from two regular Cauchy source rows, paired StreamName windows, located DyadicRatCore tolerance windows, a CauchyProduct row, RegSeqRat readback, a RegularCauchyProduct handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyProductGuard}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6987.1, Definition 6987.2, and Definition 6987.3 seed the finite located-tolerance guard for product stability of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyProductGuard}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient equality of names, selected product limits, countable choice, trichotomy, host multiplication, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located dyadic tolerance-window exactness, finite-window product assembly, regular product handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyProductGuardUp.

Chapter 6988

A Concrete Naming Certificate for $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy presentation of the real line, the $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$ packet names the finite BEDC unit that sends a displayed dyadic rational point into regular Cauchy completion data by the constant stream route. It sits between the dyadic endpoint core of chapter 291, the finite observation interface of chapter 126, the regular sequence route of chapter 301, the displayed real-name seal of chapter 38, and the completion boundary of chapter 1639. The packet records the point-to-completion embedding row; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, or host real equality.

Definition 6988.1 (Principal Cauchy embedding carrier)**.** *A* $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source point row, exposing one displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *point together with its dyadic endpoint ledger.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *window row, recording that every finite observation requested by the completion route reads the same source point row Q.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row for the constant sequence generated by S.*
4. *D is the dyadic tolerance row, recording that every requested tail budget is discharged by the equality of the repeated dyadic readback.*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after Q, S, R, D have been read.*
6. *H records componentwise* hsame *transport for the source point, constant window, regularity, tolerance, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through Q, S, R, D and then into the terminal Real seal row A.*
8. *P is the* Pkg *provenance over* $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *chapter 1639,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrincipalCauchyEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, R, D, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, or detached equality of Real names.*

Theorem 6988.2 (Principal Cauchy embedding finite route)**.** *Every accepted* $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$ *carrier*

$$E = (Q, S, R, D, A, H, C, P, N)$$

exports a completion-facing read only after the source $\mathsf{DyadicRatCore}^{\uparrow}$ *point row, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, dyadic tolerance row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row have been displayed in that order. The exported route may be consumed by the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet as point-generated regular Cauchy data, but it exports no* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limit, countable-choice schedule, host real equality, or hidden equality of Real names.*

Proof. Project the carrier of Definition 6988.1. The only source data available to a completion-facing consumer is the dyadic point row Q together with the constant finite observation row S.

Read the regularity and tolerance rows next. Since S repeats the same dyadic point through each finite window, the regular sequence row R and dyadic tolerance row D are both downstream of the displayed point ledger, not of a selected stream representative.

The Real seal row A is reached only after Q, S, R, D are visible. Componentwise transport and replay through H and C preserve that order, while P and N attach provenance and the local NameCert boundary.

The carrier inventory contains no quotient stream equality, selected limit, countable-choice schedule, host real equality, ambient Real completeness, or detached Real-name equality coordinate. Therefore the point-generated completion route is exactly the displayed finite route consumed by chapter 1639. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrincipalCauchyEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrincipalCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6988.1 and Definition 6988.2 seed the finite point-to-completion embedding route over displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrincipalCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constant-window regularity, dyadic tolerance routing, Real-seal handoff, classifier transport, non-escape, package provenance, and local naming for BEDC.Derived.PrincipalCauchyEmbeddin

Constructive story: A $\mathsf{PrincipalCauchyEmbedding}^{\uparrow}$ packet is generated from one displayed dyadic source point, a constant StreamName window, a RegSeqRat regularity row, a dyadic tolerance row, a terminal Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6989

A Concrete Naming Certificate for $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$

Following Bishop and Bridges on constructive completions by regular Cauchy data, the RegularCauchyRepletionBoun packet records the finite boundary where $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, the existing $\mathsf{CauchyRepletion}^{\uparrow}$ corridor, and the terminal $\mathsf{Real}^{\uparrow}$ seal are all displayed before any completion-facing consumer reads a repleted name. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational handoff of chapter 301, the repletion corridor of chapter 6979, and the real-name seal of chapter 38. The packet is a boundary object, not a second construction of completed reals.

Definition 6989.1 (Regular Cauchy repletion boundary carrier)**.** *A* $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule read by the boundary.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording precision scales, comparison bounds, and displayed dyadic thresholds.*
4. *Q is the* $\mathsf{CauchyRepletion}^{\uparrow}$ *corridor row through which the boundary receives completion-facing repletion data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after S, R, D, Q have been displayed.*
6. *H stores componentwise* hsame *transport for the stream schedule, regular readback, dyadic ledger, repletion corridor, seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from finite windows to regular readback, from readback through the dyadic ledger, from the dyadic ledger into the repletion corridor, and from the corridor to the terminal seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRepletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable-choice schedules, trichotomy, host real equality, or ambient completeness.

Theorem 6989.2 (Regular Cauchy repletion boundary NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$ *carrier* $B = (S, R, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRepletionBoun}}$ *surface consists exactly of carrier admission, finite-window exposure, regular rational readback, dyadic tolerance routing, Cauchy repletion corridor admission, terminal Real seal exposure, componentwise transport, displayed replay, provenance, and local naming. Every boundary consumer must display the* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*,* $\mathsf{CauchyRepletion}^{\uparrow}$ *row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and structural rows* H, C, P, N *before it reads the packet as repleted regular Cauchy data.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRepletionBoundaryUp.RegularCauchyRepletionBoundaryNamecertObligations`

Proof. Project the carrier of Definition 6989.1. The nine displayed rows are the only classifier-visible coordinates, so carrier admission and local naming are row projections.

The analytic obligations are the four displayed source steps. The stream schedule S supplies finite observations, R reads those observations as regular rational data, D records the dyadic tolerances used by the readback, and Q is the only repletion-corridor coordinate available before the seal-facing row can be read.

The structural rows preserve the same packet. The transport row H moves each displayed coordinate by hsame, the replay row C records the Cont route through the window, readback, dyadic, repletion, and seal rows, and P records the Pkg provenance for the local $\mathsf{NameCert}$ row N. Since no other classifier coordinate exists, the listed obligations exhaust the local surface. □

Theorem 6989.3 (Window repletion boundary)**.** *Let* $B = (S, R, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$ *carrier. Any completion-facing read of the terminal row* E *factors through the finite window row* S*, the regular readback row* R*, the dyadic tolerance row* D*, and the* $\mathsf{CauchyRepletion}^{\uparrow}$ *corridor row* Q*; no consumer can read* E *from* S *or* R *alone.*

Proof. Use the displayed replay row C. Its route is ordered from S to R, from R through D, from D into Q, and from Q to E, so the seal-facing read is a replay of the whole boundary rather than a direct projection from a stream or readback row.

Transport does not change the dependency order. The row H only supplies componentwise hsame transport of the same displayed packet, and P, N only record provenance and naming. Thus a consumer reading E through the local $\mathsf{NameCert}$ must pass through the repletion row Q after the finite window, readback, and dyadic rows have been displayed. □

Theorem 6989.4 (Regular Cauchy repletion boundary non-escape)**.** *Every accepted* $\mathsf{RegularCauchyRepletionBound}$ *carrier is insensitive to quotient equality of streams, selected limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. The only seal-facing information exported to a consumer is the displayed route through* BHist*,* hsame*,* Cont*,* Pkg*, and the local* $\mathsf{NameCert}$ *rows named in Definition 6989.1.*

Proof. Inspect the carrier coordinates. The tuple contains $S, R, D, Q, E, H, C, P, N$ and no row for quotient stream equality, a chosen host limit, a countable-choice schedule, trichotomy, or an ambient completeness principle.

It remains to check that the structural rows cannot add a hidden source. The hsame row H transports only displayed coordinates, the Cont row C replays only the boundary route already

present in the carrier, and the Pkg row P records provenance for the local NameCert row N. Therefore any exported seal-facing fact is confined to the finite boundary rows. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$ packet exposes StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a CauchyRepletion corridor row, a Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any completion-facing consumer reads a repleted name.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-repletion, real: RegularCauchyRepletionBo is not bijective to any listed sibling because it jointly carries the window schedule, regular readback, dyadic tolerance ledger, repletion corridor, terminal seal, transport, replay, provenance, and local naming as one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular rational readback, a DyadicRatCore tolerance ledger, a CauchyRepletion corridor row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6989.1, Definition 6989.2, Definition 6989.3, and Definition 6989.4 seed the finite non-quotient boundary between regular Cauchy repletion data and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRepletionBoundary}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyRepletionBoundaryUp.RegularCauchyRepletionBoundaryNamecertObligations`

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window readback, dyadic tolerance routing, repletion corridor admission, Real seal exposure, transport, replay, provenance, and non-escape for BEDC.Derived.RegularCauchyRepletionBound

Chapter 6990

A Concrete Naming Certificate for MetaCICSubstitutionSpineRealizer$^{\uparrow}$

Following the standard de Bruijn account of shifting and substitution under binders, the MetaCICSubstitutionSpineR packet realizes the audit map of chapter 9380 as finite BEDC spine data. It sits after the closed substitution spine of chapter 6103, the substitution audit packet of **??**, the closed substitution boundary of chapter 538, and the subject-reduction obstruction map of chapter 9377. The packet records closed shift, closed substitution, closed composition, and binder-seal audit rows; it does not claim unrestricted substitution commutation, beta-substitution preservation, closed subject reduction, quotient syntax equality, host evaluator soundness, or kernel consistency.

Definition 6990.1 (MetaCIC substitution spine realizer carrier)**.** *A* MetaCICSubstitutionSpineRealizer$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, I, B, Z, S, U, K, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed MetaCIC term row, exposing the source term, payload term, and binder depth used by the audit map.*
2. *I is the de Bruijn index row, recording cutoff, depth, and successor arithmetic needed by the closed shift and substitution clauses.*
3. *B is the binder-seal row, recording the* ClosedAt *boundary that advances under lambda and Pi constructors.*
4. *Z is the closed shift row, read through the displayed term, index, and binder-seal rows.*
5. *S is the closed substitution row, read through the same closed boundary and payload visibility.*
6. *U is the closed composition row, restricted to the checked closed-fragment composition and swap laws named by the audit map.*
7. *K is the subject-reduction handoff row, retaining the beta-substitution preservation boundary as an obligation surface rather than a discharged theorem.*
8. *L is the local audit ledger row, recording that the consumer may inspect Z, S, U before reaching K.*
9. *H records componentwise* hsame *transport for term, index, binder-seal, shift, substitution, composition, handoff, ledger, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a substitution-audit request through T, I, B, Z, S, U, K, L.*
11. *P is the* Pkg *provenance over* MetaCICSubstitutionSpineRealizer$^{\uparrow}$*, chapter 9380, chapter 6103, chapter 538, chapter 9377,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpineRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, I, B, Z, S, U, K, L, H, C, P, N$. It has no coordinate for unrestricted substitution calculus, beta-substitution preservation, closed subject reduction, quotient syntax equality, host evaluator soundness, erased derivation semantics, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICSubstitutionSpineRealizerUp.MetaCICSubstitutionSpineRealizerTasteGate`
`single carrier alignment`

Theorem 6990.2 (MetaCIC substitution spine realizer boundary)**.** *For every accepted* MetaCICSubstitutionSpineRe *carrier*

$$R = (T, I, B, Z, S, U, K, L, H, C, P, N),$$

the closed shift row Z, closed substitution row S, and closed composition row U are readable only after the term row T, de Bruijn index row I, and binder-seal row B have been displayed. A consumer that reaches the subject-reduction handoff K receives only the syntax-side audit ledger L and the structural rows H, C, P, N; it receives no beta-substitution preservation theorem, no closed subject-reduction theorem, no quotient syntax equality, and no host evaluator soundness row.

Proof. Project the carrier of Definition 6990.1. The rows preceding the operation surface are exactly the term data T, index data I, and binder-seal data B.

Read the operation rows in that order. The closed shift row Z and closed substitution row S both depend on the displayed ClosedAt boundary, while the composition row U is limited to the checked closed-fragment laws cited by the audit map.

The subject-reduction handoff K is a boundary row. The local ledger L records that the syntax-side rows Z, S, U may be inspected before this boundary, but it does not fill the beta-substitution preservation obligation.

Transport and replay through H and C preserve the same row order, while P and N attach provenance and local naming. Since no carrier coordinate stores unrestricted substitution commutation, beta preservation, closed subject reduction, quotient syntax equality, host evaluator soundness, erased derivation semantics, or kernel consistency, the consumer cannot export any of those endpoints from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpineRealizer}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpineRealizer}^{\uparrow}}$, seed)

Scope closed: Definition 6990.1 and Definition 6990.2 seed the finite substitution-audit realizer over displayed MetaCIC term, de Bruijn index, binder-seal, closed shift, closed substitution, closed composition, handoff, ledger, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpineRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution commutation, beta-substitution preservation, closed subject reduction, quotient syntax equality, host evaluator soundness, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, binder-seal gating, closed shift routing, closed substitution routing, closed composition routing, subject-reduction handoff non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSubstitutionSpineRealizerUp.

Constructive story: A MetaCICSubstitutionSpineRealizer$^{\uparrow}$ packet is generated from closed MetaCIC term data, de Bruijn index data, a ClosedAt binder seal, closed shift rows, closed substitution rows, closed composition rows, a subject-reduction handoff row, an audit ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6991

A Concrete Naming Certificate for $\mathsf{TaitReducibility}^{\uparrow}$

Following Tait and Girard on reducibility candidates, the $\mathsf{TaitReducibility}^{\uparrow}$ packet records the traditional candidate method as finite BEDC NameCert data for the MetaCIC strong-normalization route. It sits beside the critical-path packet of chapter 765, the normalization-budget packet of chapter 755, the bounded candidate frontier of chapter 4196, the logical-relation surface $\mathsf{LogicalRelation}^{\uparrow}$, and the normalization-by-evaluation surface $\mathsf{NormalizationByEvaluation}^{\uparrow}$. The packet names the closed typed term row, reducibility-candidate row, beta, application, and Pi closure rows, finite replay and provenance rows, and a local naming row before any MetaCIC consumer may read a candidate-mediated strong-normalization handoff.

Definition 6991.1 (Tait reducibility carrier)**.** *A* $\mathsf{TaitReducibility}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, E, B, A, \Pi, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row, recording the term, its type, its context boundary, and the closedness evidence consumed by the MetaCIC route.*
2. *E is the reducibility-candidate row, recording the candidate predicate, candidate membership evidence, and the closed-candidate or candidate-family witness used for the inspected type.*
3. *B is the beta-closure row, recording that the candidate read is stable under the displayed beta step window admitted by the route.*
4. *A is the application-closure row, recording the function-side candidate evidence, argument candidate evidence, and the application handoff row.*
5. *Π is the Pi-closure row, recording binder-side candidate data, dependent-family routing, and the displayed adequacy boundary for Pi-shaped terms.*
6. *S is the candidate-mediated strong-normalization handoff row, downstream of E, B, A, Π and still attached to the closed typed term row T.*
7. *Q records the refusal boundary for evidence-free closed-term normalization, unrestricted subject reduction, full Church–Rosser, global beta-conversion decidability, and completed consistency.*
8. *H records componentwise* hsame *transport for $T, E, B, A, \Pi, S, Q, C, P, N$.*

9. *C records finite* Cont *replay from the MetaCIC critical-path and normalization-budget consumers through the displayed candidate route.*
10. *P is the* Pkg *provenance over* $\mathsf{TaitReducibility}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationBudget}^{\uparrow}$, $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$, $\mathsf{LogicalRelation}^{\uparrow}$, $\mathsf{NormalizationByEvaluation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TaitReducibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, B, A, \Pi, S, Q, H, C, P, N$. It has no coordinate for unconditional strong normalization, full subject reduction, full Church–Rosser, beta-conversion decidability, or consistency.

Theorem 6991.2 (Tait reducibility NameCert obligations)**.** *For an accepted* $\mathsf{TaitReducibility}^{\uparrow}$ *carrier*

$$R = (T, E, B, A, \Pi, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaitReducibility}^{\uparrow}}$ *surface consists exactly of closed typed term exposure through T, reducibility-candidate exposure through E, beta-closure exposure through B, application closure through A, Pi closure through Π, candidate-mediated strong-normalization handoff through S, refusal exactness through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6991.1. The closed typed term row T supplies the inspected endpoint, while E is the only positive candidate row.

Read the closure rows B, A, Π as stability data for the same candidate. Beta stability, application closure, and Pi closure do not introduce independent normalization theorems; they keep the candidate route well formed for the displayed term and type rows.

The handoff row S is downstream of E, B, A, Π. Requests for evidence-free normalization, full subject reduction, full Church–Rosser, beta-conversion decidability, or consistency are sent to the refusal row Q.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed rows. Hence the NameCert surface is the finite Tait-candidate route, not a global metatheorem about every MetaCIC term. □

Theorem 6991.3 (Tait reducibility candidate-SN handoff)**.** *Every strong-normalization read exported from an accepted* $\mathsf{TaitReducibility}^{\uparrow}$ *carrier must first display the route*

$$T \longmapsto E \longmapsto (B, A, \Pi) \longmapsto S$$

and must keep the refusal row Q visible when it is consumed by $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationBudget}^{\uparrow}$ *or* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ *routes. The handoff may be compared with* $\mathsf{LogicalRelation}^{\uparrow}$ *and* $\mathsf{NormalizationByEvaluation}^{\uparrow}$ *as neighboring method surfaces, but it does not claim full subject reduction, Church–Rosser, beta-conversion decidability, arbitrary strong normalization, or consistency.*

Proof. Start with the accepted carrier. The closed term row T and reducibility-candidate row E are both visible before any normalization handoff appears.

Apply Definition 6991.2. Its beta, application, and Pi closure clauses place B, A, Π between candidate evidence and the strong-normalization handoff S.

Now read the downstream MetaCIC consumers. The critical-path, normalization-budget, and bounded-candidate chapters may inspect the displayed candidate route as dependency data, while the refusal row Q keeps the open metatheoretic principles outside the packet.

The neighboring logical-relation and normalization-by-evaluation surfaces can compare methods only through displayed provenance and replay rows. They cannot replace E, B, A, Π, S, Q by a hidden global theorem, so the exported handoff remains candidate-mediated. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaitReducibility}^{\uparrow}}$.

Constructive story: A $\mathsf{TaitReducibility}^{\uparrow}$ packet is generated from a closed typed MetaCIC term row, reducibility-candidate row, beta-closure row, application-closure row, Pi-closure row, candidate-mediated strong-normalization handoff, refusal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaitReducibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6991.1, Definition 6991.2, and Definition 6991.3 seed the finite Tait-candidate method surface for MetaCIC normalization consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaitReducibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full strong normalization, Church–Rosser, subject reduction, consistency, beta-conversion decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate exposure, beta, application, and Pi closure routing, candidate-mediated strong-normalization handoff, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.TaitReducibilityUp.

Chapter 6992

A Concrete Naming Certificate for $\mathsf{DirectedCofinalSubset}^{\uparrow}$

Following Moore–Smith and Kelley directed subsets, the $\mathsf{DirectedCofinalSubset}^{\uparrow}$ packet records a cofinal retained index surface inside an already named directed-index carrier. It refines the finite upper-window data of chapter 1607 for downstream net and completion arguments that need to restrict an index substrate without quotienting indices, selecting a cofinal sequence, or importing a choice-indexed thinning function.

The packet is a sibling handoff for the net routes of $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{MooreSmithConvergence}^{\uparrow}$, and the completion-facing Cauchy chapters. It keeps the ambient directed packet and the retained cofinal packet visible at the same time, so a consumer can prove that every ambient finite request is met inside the retained surface by displayed rows rather than by an external cofinality theorem.

Definition 6992.1 (Directed cofinal subset carrier)**.** *A* $\mathsf{DirectedCofinalSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, J, \preceq, W_I, W_J, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the ambient directed-index row read from a* $\mathsf{DirectedSet}^{\uparrow}$ *packet.*
2. *J is the retained index row naming the finite cofinal subset surface.*
3. *$\preceq$ is the displayed comparison row inherited from the ambient directed packet and restricted to the retained rows when both endpoints lie in J.*
4. *W_I is the ambient finite-window row for downstream net or completion requests.*
5. *W_J is the retained finite-window row whose entries lie in J.*
6. *U is the cofinal-witness row, assigning to every displayed ambient request in W_I a retained address in J above that request by $\preceq$.*
7. *H records componentwise* hsame *transport for the ambient row, retained row, comparison row, ambient and retained windows, cofinal witnesses, continuation replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an ambient finite-window request through the retained cofinal witness route.*
9. *P is the* Pkg *provenance over* $\mathsf{DirectedCofinalSubset}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, *the downstream net and completion consumers,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{DirectedCofinalSubset}^{\uparrow}}$ *row.*
The carrier has no quotient of directed indices, no selected cofinal sequence, no countable choice schedule, no total-order refinement, and no host-level completeness principle.

Lean4 (statement) `BEDC.Derived.DirectedCofinalSubsetUp`

Theorem 6992.2 (Directed cofinal subset NameCert obligations)**.** *For an accepted* $\mathsf{DirectedCofinalSubset}^{\uparrow}$ *carrier* $S = (I, J, \preceq, W_I, W_J, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirectedCofinalSubset}^{\uparrow}}$ *surface consists of carrier admission for the displayed rows, ambient directed-index readability through* $\mathsf{DirectedSet}^{\uparrow}$*, retained-row stability for* J *and* W_J*, cofinal-window coverage through* U*, componentwise transport and replay through* H *and* C*, provenance through* P*, and local naming through* N*. Every obligation is discharged by the finite rows of* S*.*

Proof. Carrier admission is projection from Definition 6992.1. The ambient index row, retained index row, comparison row, window rows, cofinal-witness row, transport row, replay row, provenance row, and local naming row are the complete visible packet.

Ambient directed-index readability is inherited from chapter 1607: comparisons and upper-window requests are read through the displayed $\mathsf{DirectedSet}^{\uparrow}$ rows before the retained subset is consulted.

For retained-row stability and cofinal-window coverage, inspect J, W_J, U. The retained window row names only addresses in J, and the cofinal-witness row supplies, for each ambient request in W_I, a retained address certified above the request by the displayed comparison row.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no coordinate stores quotient indices, a cofinal sequence, countable choice, a total-order refinement, or ambient completeness, the NameCert surface is exhausted by the rows of S. □

Theorem 6992.3 (Directed cofinal subset net handoff)**.** *Every* $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{CauchyNetLimit}^{\uparrow}$*,* $\mathsf{MooreSmithConvergence}^{\uparrow}$*, or completion-facing consumer that reads a directed index through an accepted* $\mathsf{DirectedCofinalSubset}^{\uparrow}$ *packet receives only the ambient directed row* I*, retained row* J*, comparison row* $\preceq$*, ambient and retained finite windows* W_I, W_J*, cofinal-witness row* U*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. The handoff may restrict a net read to the retained cofinal surface, but it does not provide a quotient net, selected subnet sequence, countable-choice selector, host directed set, or Real completeness datum.*

Proof. Begin with a downstream finite-window request. The request is first read as an ambient directed-window row in W_I over the $\mathsf{DirectedSet}^{\uparrow}$ substrate of chapter 1607.

The cofinal witness row U then names a retained address in J above the requested ambient window, and W_J records the retained window through which the downstream net or completion consumer continues.

Transport and replay through H and C keep the same ambient row, retained row, comparison row, window rows, and cofinal witness. The provenance and naming rows package exactly that route, so the consumer cannot replace it by a quotient net, selected subnet sequence, countable-choice selector, host directed set, or Real completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedCofinalSubset}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DirectedCofinalSubset}^{\uparrow}$ packet is generated from an ambient directed-index row, a retained index row, a displayed comparison row, ambient and retained finite-window rows, a cofinal-witness row, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectedCofinalSubset}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6992.1, Definition 6992.2, and Definition 6992.3 seed the finite cofinal-subset carrier and its net-facing handoff over a displayed $\mathsf{DirectedSet}^{\uparrow}$ substrate.

Formal verification: FormalStatus(NameCert$_{\mathsf{DirectedCofinalSubset}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected subnet sequences, countable choice, host directed sets, Real completeness, Lean verification of the obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, retained-row stability, cofinal-window coverage, net handoff, non-escape, and package provenance for the same finite packet.

Chapter 6993

A Concrete Naming Certificate for $\mathsf{BezoutIdentity}^{\uparrow}$

Following Bezout and Euclid, the $\mathsf{BezoutIdentity}^{\uparrow}$ packet records the finite arithmetic surface in which a displayed common divisor is reached by backtracking through a Euclidean trace. It sits over the natural-number rows of chapter 28, the signed coefficient rows of chapter 31, and the multiplication-witness divisibility ledger of chapter 331. The packet names only the displayed trace, coefficient ledger, divisor boundary, history transport, continuation replay, package provenance, and local certificate row; it does not select a unique gcd normalization, host integer equality, quotient ideals, classical Euclidean-domain generality, or an algorithm outside the finite trace shown by the carrier.

Definition 6993.1 (Bezout identity carrier)**.** *A* $\mathsf{BezoutIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (g, a, b, x, y, e, d, h, c, p, n)$$

with the following displayed rows:

1. g, a, b *are accepted arithmetic rows naming the displayed common divisor and the two input rows, with* a *and* b *read through* $\mathsf{Nat}^{\uparrow}$ *or through the signed readback supplied by* $\mathsf{Int}^{\uparrow}$ *when coefficients are consumed.*
2. x *and* y *are accepted* $\mathsf{Int}^{\uparrow}$ *coefficient rows.*
3. e *is the finite Euclidean backtracking trace, listing the displayed remainder steps and the coefficient substitutions used by the trace.*
4. d *is the divisibility boundary row, recording displayed witnesses that* g *divides* a *and* g *divides* b *through the* $\mathsf{Divisibility}^{\uparrow}$ *multiplication-witness surface.*
5. h *records componentwise* hsame *transport for the input rows, coefficient rows, trace row, divisibility boundary, continuation, provenance, and local naming rows.*
6. c *stores the* Cont *replay route that reads the trace as the displayed identity surface* $x \cdot a + y \cdot b = g$ *together with the divisor boundary.*
7. p *is the* Pkg *provenance over* $\mathsf{BezoutIdentity}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Divisibility}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. n *is the local* $\mathsf{NameCert}_{\mathsf{BezoutIdentity}^{\uparrow}}$ *row.*

The carrier exports no row for unique gcd normalization, host integer equality, quotient ideals, an ambient Euclidean domain, or a selected algorithm beyond the finite Euclidean trace e.

Theorem 6993.2 (BezoutIdentity NameCert obligations)**.** *For every accepted* $\mathsf{BezoutIdentity}^{\uparrow}$ *carrier*

$$B = (g, a, b, x, y, e, d, h, c, p, n),$$

the local $\mathsf{NameCert}_{\mathsf{BezoutIdentity}^{\uparrow}}$ *surface consists exactly of carrier admission for B, input admission for g, a, b, integer-coefficient admission for x, y, Euclidean-trace exposure through e, divisibility-boundary exposure through d, componentwise* hsame *stability through h,* Cont *replay through c,* Pkg *provenance through p, and local naming through n.*

Proof. Project the accepted carrier of Definition 6993.1. The rows g, a, b name the divisor and inputs, while x, y name the signed coefficient rows supplied by the integer certificate.

The trace row e is the only place where Euclidean backtracking is stored. Its displayed substitutions are replayed by c as the identity surface $x \cdot a + y \cdot b = g$.

The divisor boundary row d is read through chapter 331. It supplies multiplication-witness divisibility of a and b by g, not a quotient ideal or a host gcd predicate.

Finally, the rows h, p, n transport, source, and name the same finite packet. Since the carrier has no coordinate for unique normalization, host integer equality, classical Euclidean-domain generality, or an algorithm selector outside e, none of those objects belongs to the local NameCert surface. □

Theorem 6993.3 (Bezout Euclidean backtracking)**.** *Every downstream consumer of an accepted* $\mathsf{BezoutIdentity}^{\uparrow}$ *carrier that reads a displayed identity $x \cdot a + y \cdot b = g$ must factor through the finite route*

$$e, d \longmapsto h, c, p, n.$$

The Euclidean trace e supplies the only backtracking substitutions, and the continuation row c is the only route by which those substitutions are read as a signed $\mathsf{Int}^{\uparrow}$ *coefficient identity over the input rows.*

Proof. Begin with an accepted packet $B = (g, a, b, x, y, e, d, h, c, p, n)$. The rows x and y are integer coefficient rows, while a and b are the input rows cited by the trace.

The Euclidean trace e lists the finite remainder and substitution steps. Reading the displayed identity requires replaying those steps through c, so the consumer cannot replace the trace by a host-level algorithm or by an untracked extended-gcd routine.

The rows h, p, n then preserve, source, and name the same trace route. Any selected algorithm outside e would require an additional coordinate, and Definition 6993.1 has none. □

Theorem 6993.4 (Bezout gcd-divisibility boundary)**.** *For every accepted* $\mathsf{BezoutIdentity}^{\uparrow}$ *carrier $B = (g, a, b, x, y, e, d, h, c, p, n)$, the common-divisor information exported by the packet is exactly the displayed* $\mathsf{Divisibility}^{\uparrow}$ *boundary row d together with the Euclidean trace e. The packet may be consumed as a Bezout witness for the named divisor row g, but it does not identify g with a uniquely normalized gcd, does not quotient integer rows by host equality, does not build a quotient ideal, and does not assert Euclidean-domain generality beyond the displayed finite trace.*

Proof. Read d through the divisibility carrier of chapter 331. It exposes multiplication witnesses for the two divisor claims involving g, a, b.

Read e through the backtracking theorem Definition 6993.3. This supplies the displayed coefficient identity over the same input rows and coefficient rows.

Together these two rows give the Bezout witness surface: a named divisor row, displayed divisibility of both inputs, and a finite signed coefficient identity. The carrier contains no row for least positive representative, associate-class normalization, quotient ideals, or ambient Euclidean-domain law, so the gcd boundary remains exactly the displayed divisibility boundary. □

Falsifiable prediction: A consumer that claims a $\mathsf{BezoutIdentity}^{\uparrow}$ certificate while omitting the Euclidean trace row, the divisibility boundary row, integer coefficient rows, hsame, Cont, Pkg, or the local NameCert row refutes the carrier of Definition 6993.1.

Independence witness: nat, int, divisibility: $\mathsf{BezoutIdentity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Euclidean backtracking trace, signed coefficient rows, and a divisibility boundary. NatUp supplies unsigned arithmetic rows; IntUp supplies signed coefficient rows without a Euclidean trace; DivisibilityUp supplies multiplication-witness divisor rows without the coefficient identity route.

Closure status for $\mathsf{NameCert}_{\mathsf{BezoutIdentity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BezoutIdentity}^{\uparrow}$ packet is generated from displayed input rows, signed coefficient rows, a finite Euclidean backtracking trace, a divisibility boundary, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BezoutIdentity}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6993.1, Definition 6993.2, Definition 6993.3, and Definition 6993.4 form the obligation surface for finite Bezout backtracking over NatUp, IntUp, and DivisibilityUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BezoutIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unique gcd normalization, host integer equality, quotient ideals, classical Euclidean-domain generality, Lean verification, bridge checking, or a selected algorithm outside the displayed finite trace.

Upgrade path: Scoped closure requires a consumer certificate showing that later dyadic, rational, modular, and divisibility readers factor through the displayed Euclidean trace and divisor boundary.

Chapter 6994

A Concrete Naming Certificate for $\mathsf{SchurComplement}^{\uparrow}$

Following Schur and standard finite block-matrix linear algebra, the $\mathsf{SchurComplement}^{\uparrow}$ packet records the finite displayed route by which a block matrix with an invertible pivot block exposes its complement row. It consumes the matrix carrier of chapter 49, the linear-map surface of chapter 48, the determinant rows of chapter 84, the field scalar source of chapter 45, and the vector-space rows of chapter 47. The packet names only displayed block rows, pivot invertibility evidence, finite multiplication rows, transport, replay, provenance, and local naming; it does not assert a host matrix inverse theorem, quotient matrix equality, Gaussian elimination over arbitrary ambient objects, or a selected global decomposition.

Definition 6994.1 (Schur complement carrier)**.** *A* $\mathsf{SchurComplement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, V, M, A, B, C, D, I, Q, E, H, R, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Field}^{\uparrow}$ *scalar row and V is the finite* $\mathsf{VecSpace}^{\uparrow}$ *row used by the block coordinates.*
2. *M is the ambient* $\mathsf{Mat}^{\uparrow}$ *row, read as the displayed block matrix with rows A, B, C, D.*
3. *I is the pivot-invertibility row for D, recording the finite right and left inverse checks available inside the matrix packet.*
4. *Q is the displayed product row for $BD^{-1}C$ and the complement row $A - BD^{-1}C$.*
5. *E is the exactness row tying the complement to the block multiplication and determinant handoff.*
6. *H records componentwise* hsame *transport for the field, vector, block, inverse, product, exactness, replay, provenance, and naming rows.*
7. *R records displayed* Cont *replay from the block matrix through the complement row to determinant, inverse, and factorization consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{SchurComplement}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurComplement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host matrix equality, an unbounded inverse operator, a selected Gaussian-elimination trace outside the displayed block ledger, or choice of bases beyond the finite vector-space rows already named.

Theorem 6994.2 (Schur complement NameCert obligations)**.** *For every accepted* $\mathsf{SchurComplement}^{\uparrow}$ *carrier* $S = (F, V, M, A, B, C, D, I, Q, E, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SchurComplement}^{\uparrow}}$ *surface consists exactly of carrier admission, field and vector-space exposure, block-matrix exposure, pivot-invertibility display, finite product display, complement exactness, determinant-facing handoff, componentwise transport, replay, provenance, and local naming.*

The five-row obligation sketch is:

1. *block admission exposes* F, V, M, A, B, C, D *as displayed* $\mathsf{Field}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$ *rows;*
2. *pivot admission exposes* I *only as finite inverse checks for the chosen pivot block;*
3. *complement formation exposes* Q *only as the displayed multiplication and subtraction row for* $A - BD^{-1}C$*;*
4. *determinant handoff exposes* E *only through the determinant-facing route, with no host determinant lemma imported as theorem content;*
5. *structural replay exposes* H, R, P, N *as transport, continuation, provenance, and naming rows, and exports no hidden matrix equality principle.*

Proof. Project the carrier of Definition 6994.1. The field, vector-space, block-matrix, pivot, product, and exactness rows are displayed before any consumer-facing route is available.

The pivot row I supplies only finite inverse checks for the displayed block D. The product row Q then records the multiplication path through B, the inverse witness, and C, and the exactness row E binds the result to the complement read.

The determinant-facing handoff is routed through E, R, P, N and remains inside the same finite packet. Componentwise hsame transport in H preserves each displayed row, so no host matrix equality, global inverse theorem, or selected elimination trace can be projected from the certificate. □

Theorem 6994.3 (Schur complement block-matrix handoff)**.** *Every matrix-factorization, determinant, inverse, or finite linear-system consumer of an accepted* $\mathsf{SchurComplement}^{\uparrow}$ *packet must first read the finite rows*

$$F, V, M, A, B, C, D, I, Q, E, H, R, P, N.$$

In particular, a consumer may use the complement row only after the block matrix, pivot inverse checks, product route, exactness row, transport row, replay row, provenance row, and local naming certificate have been displayed.

Proof. Let a downstream consumer route be given. Carrier projection gives the displayed scalar, vector, matrix, block, pivot, product, exactness, transport, replay, provenance, and naming rows.

The consumer first reads the block matrix through M, A, B, C, D. It then reads the pivot evidence through I and the complement route through Q, E.

Finally the consumer-facing handoff must pass through H, R, P, N. These rows preserve and expose the same displayed packet, but they do not add host matrix equality, an ambient inverse theorem, a determinant lemma outside the displayed route, or choice of bases. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurComplement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurComplement}^{\uparrow}$ packet is generated from displayed FieldUp, VecSpaceUp, MatrixUp, LinearMapUp, DeterminantUp, block, pivot-inverse, product, complement-exactness, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchurComplement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6994.1, Definition 6994.2, and Definition 6994.3 seed the finite block-matrix complement surface over displayed matrix, linear-map, determinant, field, vector-space, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SchurComplement}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host matrix equality, a global matrix inverse theorem, arbitrary Gaussian elimination, determinant lemmas outside the displayed route, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pivot-inverse display, finite product exactness, determinant handoff, transport stability, and consumer non-escape for BEDC.Derived.SchurComplementUp.

Chapter 6995

A Concrete Naming Certificate for HyperbolicGeodesicMidpoint$^{\uparrow}$

The HyperbolicGeodesicMidpoint$^{\uparrow}$ packet realizes the Poincare disk midpoint route of section 9463.3 as finite metric, boundary, phase-transport, provenance, and NameCert rows. It sits after the metric readback of chapter 3474, the boundary-geodesic surface of chapter 3491, the disk automorphism carrier of chapter 6776, the visual-boundary packet of chapter 4181, and the reversible phase-transport route of chapter 3455. The packet names only a displayed finite midpoint certificate; it does not complete the disk, quotient ideal endpoints, select a global geodesic representative, or import host equality of hyperbolic points.

Definition 6995.1 (Hyperbolic geodesic midpoint carrier)**.** *A* HyperbolicGeodesicMidpoint$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, G, B, A, V, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare disk coordinate row for the two displayed endpoints and the candidate midpoint.*
2. *G is the* HyperbolicMetric$^{\uparrow}$ *readback row comparing the two endpoint-to-midpoint distances.*
3. *B is the* HyperbolicGeodesicBoundary$^{\uparrow}$ *row naming the boundary-determined geodesic route on which the midpoint is read.*
4. *A is the* PoincareDiskAutomorphism$^{\uparrow}$ *normalization row used to move the finite geodesic read to a standard disk position and back.*
5. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *readback row for the same boundary pair.*
6. *T is the* PoincareDiskPhaseTransport$^{\uparrow}$ *row carrying the midpoint read through reversible phase transport.*
7. *E is the midpoint exactness row, recording equal displayed metric halves and the boundary-geodesic incidence check.*
8. *H records componentwise* hsame *transport for the disk, metric, boundary, automorphism, visual-boundary, phase-transport, exactness, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from endpoint data through midpoint exactness to metric, boundary, automorphism, and phase-transport consumers.*

10. *P is the* Pkg *provenance over* HyperbolicGeodesicMidpoint$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBoundary$^{\uparrow}$, PoincareDiskAutomorphism$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicGeodesicMidpoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, G, B, A, V, T, E, H, C, P, N$. It has no coordinate for a completed hyperbolic plane, quotient ideal boundary, selected global geodesic object, physical observer law, or host point equality.

Theorem 6995.2 (Hyperbolic geodesic midpoint NameCert obligations)**.** *For every accepted* HyperbolicGeodesicMidpoint$^{\uparrow}$ *carrier $M = (D, G, B, A, V, T, E, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicGeodesicMidpoin}}$ *surface consists exactly of carrier admission, disk-coordinate exposure, metric midpoint readback, boundary-geodesic incidence, automorphism normalization, visual-boundary readback, phase-transport replay, midpoint exactness, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6995.1. The disk endpoints and candidate midpoint are displayed by D, while the metric, boundary, automorphism, visual-boundary, and phase rows are displayed by G, B, A, V, T.

The exactness row E records the finite midpoint check: the candidate lies on the displayed boundary-geodesic route and the two metric halves agree inside the recorded readback. Transport through H is componentwise, so it preserves exactly those rows.

Consumers can inspect the midpoint only through C, P, N. These rows replay and name the same finite packet, and therefore cannot export a completed hyperbolic plane, quotient boundary, selected global geodesic, or host point equality. □

Theorem 6995.3 (Hyperbolic geodesic midpoint boundary handoff)**.** *Every boundary, visual, automorphism, phase-transport, or observer-expansion consumer of an accepted* HyperbolicGeodesicMidpoint$^{\uparrow}$ *packet must first read the finite rows*

$$D, G, B, A, V, T, E, H, C, P, N.$$

In particular, the midpoint row is exported only as the displayed point that lies on the named boundary-geodesic route, has equal displayed HyperbolicMetric$^{\uparrow}$ *distance halves, and is transported through the same* PoincareDiskPhaseTransport$^{\uparrow}$ *provenance.*

Proof. Let a downstream consumer be given. Carrier projection gives the disk-coordinate row D, the metric row G, the boundary row B, the automorphism row A, the visual-boundary row V, and the phase-transport row T.

The consumer then reads the midpoint exactness through E. This row binds the incidence and equal-half metric checks to the same displayed endpoint pair and candidate midpoint.

Finally the handoff passes through H, C, P, N. These structural rows transport, replay, record provenance, and name the packet without adding a completed disk, quotient ideal endpoints, a selected global geodesic, or host point equality. □

Falsifiable prediction: Every accepted HyperbolicGeodesicMidpoint$^{\uparrow}$ packet exposes disk-coordinate, hyperbolic-metric, geodesic-boundary, Poincare-disk-automorphism, visual-boundary, phase-transport, exactness, transport, replay, provenance, and NameCert rows before any midpoint consumer reads it.

Independence witness: hyperbolic-metric, hyperbolic-geodesic-boundary, poincare-disk-automorphism, hyperbolic-visual-boundary, poincare-disk-phase-transport: $\mathsf{HyperbolicGeodesicMidpoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a displayed endpoint pair, candidate midpoint, metric equal-half readback, boundary-geodesic incidence, automorphism normalization, visual-boundary readback, phase transport, exactness, transport, replay, provenance, and local naming in one packet.

Lean follow-up requires $\mathsf{Nontrivial\ HyperbolicGeodesicMidpoint}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicGeodesicMidpoint}^{\uparrow}$ witnesses in the corresponding TasteGate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicMidpoint}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicGeodesicMidpoint}^{\uparrow}$ packet is generated from disk-coordinate, hyperbolic-metric, geodesic-boundary, automorphism, visual-boundary, phase-transport, midpoint-exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicMidpoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6995.1, Definition 6995.2, Definition 6995.3, the falsifiable prediction, and the independence witness seed the finite Poincare-disk geodesic midpoint surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicMidpoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed hyperbolic plane, quotient ideal boundary, selected global geodesic representatives, host point equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric midpoint readback, boundary-geodesic incidence, automorphism normalization, visual-boundary readback, phase-transport replay, nontrivial and field-faithful TasteGate instances, and consumer non-escape for BEDC.Derived.HyperbolicGeodesicMidpointUp.

Chapter 6996

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy reals and explicit moduli, the $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$ packet records the finite diagonal schedule that turns a displayed family of regular Cauchy names into one completion-facing modulus row. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy-name-space route of chapter 406. The packet names a diagonal modulus as finite BEDC rows; it is not a quotient of streams, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6996.1 (Regular Cauchy diagonal-modulus carrier)**.** *A* $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, \sigma, \Delta, \epsilon, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed regular-family row, whose members are* $\mathsf{RegSeqRat}^{\uparrow}$ *names read through one common* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *surface.*
2. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule exposing the finite windows of the family needed at each requested tolerance.*
3. *Δ is the diagonal window selector, recording which scheduled family-window and sequence-window are inspected for the current precision request.*
4. *ϵ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the diagonal budget and the componentwise regularity comparisons.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected diagonal windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after $F, \sigma, \Delta, \epsilon, Q$ have been displayed.*
7. *H stores componentwise* hsame *transport for the regular family, stream schedule, diagonal selector, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a completion-facing request through $F, \sigma, \Delta, \epsilon, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, selected limiting representatives, countable choice, host real equality, or an ambient completeness theorem.

Theorem 6996.2 (Regular Cauchy diagonal-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$ *carrier* $D = (F, \sigma, \Delta, \epsilon, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModu}}$ *surface consists exactly of regular-family admission, scheduled finite-window exposure, diagonal selector exactness, dyadic tolerance-ledger exactness, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming for the rows of Definition 6996.1.*

Proof. Project the accepted carrier. The analytic rows are $F, \sigma, \Delta, \epsilon, Q, E$, and the structural rows are H, C, P, N.

The family row F is read first through the Cauchy-name-space route. The schedule σ and diagonal selector Δ then specify finite windows of that family, so the consumer never receives an unbounded stream or selected subsequence.

The tolerance and readback rows ϵ, Q are downstream of the selected windows. They expose dyadic budgets and regular rational names before the Real seal E is available.

Finally the transport, replay, provenance, and local naming rows preserve the same finite packet. Since the carrier inventory contains no quotient, selected limit, choice, host equality, or completeness coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 6996.3 (Regular Cauchy diagonal-modulus Cauchy handoff)**.** *Every* $\mathsf{CauchySequenceSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$ *carrier reads the route*

$$\mathsf{CauchySequenceSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $F, \sigma, \Delta, \epsilon, Q, E, H, C, P, N$. The diagonal modulus exports a tolerance-indexed finite-window handoff and does not export quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 6996.2. It forces the consumer to expose the regular family, schedule, diagonal selector, dyadic ledger, regular readback, Real seal, and structural rows before any completion-facing read is available.

The Cauchy handoff therefore reaches $\mathsf{RegSeqRat}^{\uparrow}$ only through the scheduled diagonal windows and reaches $\mathsf{Real}^{\uparrow}$ only after the dyadic tolerance ledger has been replayed. The rows H, C, P, N keep this route stable under the local classifier and add no second source.

Thus every accepted consumer factors through the displayed finite route from Cauchy-name-space data to the Real seal. Any use of a quotient stream, selected limit, choice schedule, host equality, or ambient completeness theorem lies outside the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}$ consumer that presents a $\mathsf{Real}^{\uparrow}$ seal without first displaying $F, \sigma, \Delta, \epsilon, Q, E, H, C, P, N$ refutes the carrier surface.

Independence witness: Nearest siblings: cauchysequencespace, streamname, regseqrat, dyadicratcore, and real. The packet is not bijective to any one sibling because it jointly carries a regular-family row, stream schedule, diagonal selector, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulus}^{\uparrow}}$.

Constructive story: A RegularCauchyDiagonalModulus$^{\uparrow}$ packet is generated from a CauchySequenceSpace regular-family row, StreamName schedules, a diagonal window selector, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyDiagonalModulus}\uparrow}$, seed)

Scope closed: Definition 6996.1, Definition 6996.2, and Definition 6996.3 seed the finite diagonal-modulus route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDiagonalModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host Real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scheduled-window exposure, diagonal-selector exactness, DyadicRatCore tolerance exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyDiagonalModulusUp.

Chapter 6997

A Concrete Naming Certificate for $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, and the finite-cover construction of chapter 9237, the $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ packet records the finite route from star refinements and local moduli to a tolerance-only uniform-modulus handoff. It sits after the compact metric source of chapter 129, the continuous-map rows of chapter 130, the finite Lebesgue-number surface of chapter 3606, the compact-cover lower-bound ledger of $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$, the rational and dyadic tolerance rows of $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$, and the uniform-modulus consumer of chapter 369. The packet names a finite star-refinement modulus and not an open-cover compactness theorem, selected subcover, host continuous function, quotient cover equality, or ambient completeness principle.

Definition 6997.1 (Finite cover star-refinement modulus carrier)**.** *A* $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, f, U, V, L, R, D, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compact evidence is read as finite cover data.*
2. *f is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the local modulus data attached to the same source.*
3. *U is the finite cover row and V is the star-refinement row over that cover.*
4. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *or* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *lower-bound row for the star refinement.*
5. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row requested by the target consumer.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger comparing the requested tolerance with the finite lower-bound data.*
7. *M is the resulting* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, available only after K, f, U, V, L, R, D have been displayed.*
8. *H stores componentwise* hsame *transport for the compact source, continuous map, cover, star refinement, lower-bound, rational, dyadic, uniform-modulus, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a compact-uniform-continuity request through* $K, f, U, V, L, R, D, M, H, P, N$
10. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, host-function equality, quotient covers, countable choice, or ambient completion.

Theorem 6997.2 (Finite cover star-refinement modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ *carrier* $S = (K, f, U, V, L, R, D, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}}$ *surface consists exactly of compact metric admission, continuous-map local-modulus admission, finite cover exposure, star-refinement exactness, finite Lebesgue-number or compact-cover ledger admission, positive rational tolerance admission, dyadic tolerance comparison, uniform-modulus output exactness, componentwise transport, displayed replay, provenance, and local naming for the rows of Definition 6997.1.*

Proof. Project the accepted carrier. The analytic rows are K, f, U, V, L, R, D, M, and the structural rows are H, C, P, N.

The compact source K and continuous-map row f are read before the finite cover is used. The cover row U and star-refinement row V then give the finite local geometry inspected by the modulus construction.

The lower-bound row L, rational tolerance row R, and dyadic comparison row D determine the tolerance-only output row M. This order prevents the uniform-modulus consumer from replacing the finite star-refinement data by an open-cover compactness theorem or selected subcover.

Finally H, C, P, N transport, replay, package, and name the same packet. Since no forbidden principle is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6997.3 (Finite cover star-refinement uniform handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ *carrier reads the route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, U, V, \mathsf{FiniteLebesgueNumber}^{\uparrow}, \mathsf{CompactCoverLebesgueLedger}^{\uparrow}, \mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}$$

through the displayed rows $K, f, U, V, L, R, D, M, H, C, P, N$. The handoff exports a tolerance-only uniform-modulus row and exports no open-cover compactness theorem, selected finite subcover, host continuous map, quotient cover equality, countable choice, or ambient completeness.

Proof. Begin with Definition 6997.2. It fixes the compact source, continuous-map data, finite cover, star refinement, lower-bound ledger, rational tolerance, dyadic comparison, and structural rows before the uniform-modulus output can be read.

The row M is therefore downstream of the star-refinement and lower-bound rows. A consumer reading M receives only the tolerance-indexed precision certified by R, D, L, not a selected subcover or host modulus function.

The structural rows H, C, P, N replay the same finite packet for the compact-uniform-continuity route of chapter 9237. Because the carrier lacks open-cover compactness, quotient cover equality, choice, and ambient completeness coordinates, the uniform handoff is exactly the displayed BEDC route. □

Falsifiable prediction: An accepted $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ consumer that exports a $\mathsf{UniformModulus}^{\uparrow}$ precision without displaying $K, f, U, V, L, R, D, M, H, C, P, N$ refutes the carrier surface.

Independence witness: Nearest siblings: compactmetric, continuousmap, finite-cover-lebesgue-number, compactcoverlebesgueledger, rat, dyadicratcore, and uniformmodulus. The packet binds a star-refinement row to the tolerance-only uniform-modulus output, while each sibling supplies only its own carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$ packet is generated from CompactMetricUp source rows, ContinuousMapUp local-modulus rows, finite cover rows, star-refinement rows, FiniteLebesgueNumberUp or CompactCoverLebesgueLedgerUp lower-bound rows, positive RatUp tolerances, DyadicRatCoreUp comparisons, a UniformModulusUp output, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6997.1, Definition 6997.2, and Definition 6997.3 seed the finite star-refinement route from compact finite-cover data to a tolerance-only uniform-modulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, host continuous maps, quotient cover equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, star-refinement exactness, lower-bound ledger admission, Rat and DyadicRatCore tolerance comparison, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverStarRefinementModulusUp.

Chapter 6998

A Concrete Naming Certificate for $\mathsf{BishopContinuityModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopContinuityModulus}^{\uparrow}$ packet records the local continuity-modulus handoff for a displayed real-valued map at a displayed source point. It sits after the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the continuity carrier of chapter 130. The packet names a local modulus request as finite BEDC data; it is not compactness, uniform continuity, trichotomy, quotient real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6998.1 (Bishop continuity modulus carrier)**.** *A* $\mathsf{BishopContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, x, e, m, S, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *source row, recording the displayed real-valued map and its local continuity surface.*
2. *x is the source-point row, naming the point at which the local continuity request is read.*
3. *e is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request row, recording the positive dyadic target precision.*
4. *m is the local modulus witness row, giving the finite neighbourhood budget whose use is licensed only for the source point x and tolerance request e.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite observation windows used to test the source and image names.*
6. *D is the dyadic tolerance ledger comparing the request e with the windows selected by m.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the window observations into regular rational data.*
8. *W is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposing the target real-name read only after S, D, R have been displayed.*
9. *H records componentwise* hsame *transport for F, x, e, m, S, D, R, W and the local naming row.*

10. *C records displayed* Cont *replay through the continuity source, source point, tolerance request, local modulus, windows, dyadic comparisons, regular readback, real seal, provenance, and local naming rows.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopContinuityModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares carriers only through $F, x, e, m, S, D, R, W, H, C, P, N$. It has no coordinate for a compact domain, a uniform modulus over all source points, real trichotomy, quotient real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6998.2 (Bishop continuity modulus NameCert obligations)**.** *For the carrier of Definition 6998.1, the local* $\mathsf{NameCert}_{\mathsf{BishopContinuityModulus}^{\uparrow}}$ *surface consists exactly of continuity-source exposure through F, source-point exposure through x, dyadic tolerance request through e, local modulus witness through m, StreamName windows through S, dyadic comparison through D, RegSeqRat readback through R, terminal Real seal through W, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier to its displayed rows. The continuity-source row F, source point x, tolerance request e, and local modulus witness m determine the local request before any real-facing data can be read.

Read the observation side next. The StreamName windows S expose the finite source and image windows, the dyadic ledger D compares those windows with the requested tolerance, and the RegSeqRat row R supplies the regular rational readback used by the endpoint.

Finally apply the terminal seal and structural rows. The Real row W is reached only after the window, dyadic, and regular-rational rows are displayed, while H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6998.3 (Bishop continuity modulus real handoff)**.** *Let $B = (F, x, e, m, S, D, R, W, H, C, P, N)$ be an accepted* $\mathsf{BishopContinuityModulus}^{\uparrow}$ *carrier. Any downstream consumer that reads the local continuity output as* $\mathsf{Real}^{\uparrow}$ *data must first factor through the source point x, tolerance request e, local modulus witness m,* $\mathsf{StreamName}^{\uparrow}$ *window row S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal W on the same packet route.*

This handoff is local at x and exports no compactness theorem, no uniform-continuity modulus over all points, no trichotomy principle, no quotient real equality, no selected limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 6998.2. It places the source point, tolerance request, local modulus witness, windows, dyadic comparisons, regular readback, and real seal in one local certificate surface.

Use the sibling routes named in the chapter opening. The continuity source is read through chapter 130; finite observations are read through chapter 126; dyadic tolerance data through chapter 291; regular readback through chapter 301; and the endpoint seal through chapter 38.

The route remains local because m is tied to the displayed source point x and request e. A downstream consumer therefore reaches $\mathsf{Real}^{\uparrow}$ only through S, D, R, W together with H, C, P, N, and no excluded compactness, uniformity, trichotomy, quotient-equality, selected-limit, or completeness coordinate is available. □

Falsifiable prediction: Any accepted BishopContinuityModulusUp packet that exports a Real-facing continuity read without displaying the source point, tolerance request, local modulus witness, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and terminal Real seal refutes the carrier.

Independence witness: continuousmap, streamname, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{BishopContinuityModulus}^{\uparrow}$ because this carrier binds a local source-point request, tolerance, modulus witness, finite windows, dyadic comparison, regular readback, terminal seal, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopContinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopContinuityModulus}^{\uparrow}$ packet is generated from a continuity-source row, source point, dyadic tolerance request, local modulus witness, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal Real seal, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6998.1, Definition 6998.2, Definition 6998.3, the falsifiable prediction, and the independence witness record the local continuity-modulus seed surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim compactness, uniform continuity, trichotomy, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires grounding the source point, tolerance request, local modulus witness, StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows against explicit kernel primitives.

Chapter 6999

A Concrete Naming Certificate for $\mathsf{CantorDiagonalCompactness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ packet records the constructive diagonal extraction surface for finitely inspected binary or nested real windows. It sits after the dyadic arithmetic core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the shrinking nested-interval route of chapter 4227. The packet gives Real and Completion consumers a named diagonal subsequence boundary without asserting open-cover compactness, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The consumed shrinking-interval input is the finite diameter ledger of chapter 4227: a diagonal read must first display the nested-shrink tolerance row that bounds the inspected interval widths, then pass that row through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. Thus the compactness route consumes a concrete L10 certificate before it reads diagonal stabilization data.

Definition 6999.1 (Cantor diagonal compactness carrier)**.** *A* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, Q, E, I, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window family listing the inspected rows, prefixes, and diagonal coordinates.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the inspected prefixes and nested interval widths.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only from the displayed windows and dyadic ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, Q are displayed.*
5. *I is the* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *row that records the inspected shrinking interval or cylinder route used by the diagonal read.*
6. *W is the diagonal ledger, recording which finite coordinates have stabilized inside the inspected window and which nested interval row supplies the next readback.*
7. *H stores componentwise* hsame *transport for stream windows, dyadic tolerances, readback, Real seal, nested-shrink data, diagonal ledger, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from diagonal requests through the same finite rows.*

9. *P records* Pkg *provenance over* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedShrinkInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorDiagonalCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, Q, E, I, W, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, quotient real equality, trichotomy, a selected infinite subsequence outside the displayed windows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6999.2 (Cantor diagonal compactness NameCert obligations)**.** *For every accepted* CantorDiagonalCompa *carrier*

$$K = (S, D, Q, E, I, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorDiagonalCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, finite window exposure, dyadic tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, nested-shrink interval routing, diagonal-ledger exposure, componentwise transport, displayed replay, provenance, and local naming. Every diagonal read must pass through the sibling rows named in chapter 291, chapter 126, chapter 301, chapter 38, and chapter 4227.*

The certificate exports only this finite diagonal route. It does not export open-cover compactness, countable choice, quotient real equality, trichotomy, a host-selected subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6999.1. The accepted packet displays exactly the finite windows, dyadic ledger, regular readback, Real seal, nested-shrink row, diagonal ledger, and structural rows.

The diagonal-ledger clause is read from W only after S, D, Q, E, I have been exposed. Thus a consumer sees finite inspected coordinates, tolerance budgets, regular rational readback, the Real seal, and the nested-shrink route before it reads any diagonal stabilization datum.

Transport and replay preserve the same route. The rows H and C cannot introduce a second stream schedule, a choice function, a quotient equality, or an open-cover compactness principle; they only move the displayed packet componentwise.

Finally, P and N record provenance and local naming for the same finite carrier. Since the refused endpoints are absent from the carrier, they are absent from the local NameCert surface. □

Theorem 6999.3 (Cantor diagonal compactness Real-completion route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *packet reaches diagonal data only through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{NestedShrinkInterval}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumed packet is the displayed route $S, D, I, Q, E, W, H, C, P, N$ of Definition 6999.1. No consumer may cite the packet as open-cover compactness, a selected infinite subsequence, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier. The stream windows S and dyadic ledger D delimit every inspected coordinate and tolerance request before any diagonal entry is read.

Use the nested-shrink row I and diagonal ledger W. These rows identify the finite shrinking interval or cylinder stage that supplies each displayed diagonal read, so the route remains inside the inspected windows rather than inside a host sequence space.

The regular readback Q and terminal seal E then provide the Real-facing endpoint. Because Q is downstream of S, D, I, W and E is downstream of Q, the completion consumer receives only the displayed route.

The structural rows H, C, P, N preserve and name this same route. They do not add open-cover compactness, choice, quotient equality, trichotomy, selected subsequences, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ Real or Completion read factors through finite StreamName windows, a DyadicRatCore tolerance ledger, NestedShrinkInterval routing, RegSeqRat readback, RealUp sealing, a diagonal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, nested-shrink-interval: CantorDiagonalComp is not bijective to any listed sibling because it jointly carries diagonal finite-window stabilization, dyadic tolerance routing, nested-shrink interval data, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorDiagonalCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ packet is generated from finite StreamName windows, a DyadicRatCore tolerance ledger, NestedShrinkInterval routing, RegSeqRat readback, a terminal RealUp seal, a diagonal stabilization ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorDiagonalCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6999.1, Definition 6999.2, and Definition 6999.3 seed the finite diagonal route through StreamName, DyadicRatCore, NestedShrinkInterval, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorDiagonalCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, quotient real equality, trichotomy, selected infinite subsequences outside the displayed windows, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, nested-shrink routing, diagonal-ledger exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorDiagonalCompactnessUp.

Chapter 7000

A Concrete Naming Certificate for $\mathsf{HolderSpace}^{\uparrow}$

Following Kolmogorov–Fomin and standard Holder-space treatments, the $\mathsf{HolderSpace}^{\uparrow}$ packet records a finite function-space boundary for Real-valued maps whose inspected graph rows carry a displayed Holder budget. It consumes the Holder continuity packet of chapter 1766, the finite inequality route of chapter 2100, the metric source of chapter 57, the Real endpoint seal of chapter 38, the graph-facing continuous-map row of chapter 130, and the explicit uniform-modulus consumer of chapter 369. The chapter names the displayed function-family carrier; it does not import a host Banach space, quotient equality of functions, selected limits, ambient completeness, unrestricted real powers, or bridge checking.

Definition 7000.1 (Holder space carrier)**.** *A* $\mathsf{HolderSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, d, F, a, K, U, H, C, P, N)$$

with the following displayed rows:

1. X, d *is the* $\mathsf{Met}^{\uparrow}$ *source row, including the finite point classifier, displayed distance classifier, and inspected source-distance observations.*
2. F *is the Real-valued function-family row. Each accepted member is read through a* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row and terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
3. a *is the exponent-window row. It records the displayed positive rational exponent bounds used by the local Holder read and no unrestricted host exponent calculus.*
4. K *is the finite Holder constant row. It stores the Real-facing coefficient ledger and the finite comparison budget used to read the Holder bound.*
5. U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *export row produced from the displayed metric, graph, exponent, and constant data.*
6. H *records componentwise* hsame *transport for the metric source, function-family row, exponent window, Holder constant, uniform-modulus export, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a source-distance request through* F, a, K, U, H, P, N *to the Real-valued endpoint comparison.*
8. P *is the* Pkg *provenance over* $\mathsf{HolderSpace}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{HolderInequality}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{HolderSpace}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, d, F, a, K, U, H, C, P, N$. It has no coordinate for host real powers, completeness of the function space, quotient equality of functions, selected limits, countable choice, or a detached continuous-map theorem.

Lean4 ✓ `BEDC.Derived.HolderSpaceUp.HolderSpaceTasteGate single carrier alignment`

Theorem 7000.2 (Holder space uniform modulus route)**.** *For every accepted* $\mathsf{HolderSpace}^\uparrow$ *carrier* $S = (X, d, F, a, K, U, H, C, P, N)$*, a* $\mathsf{UniformModulus}^\uparrow$ *consumer may read the Holder-space packet only through the displayed route*

$$X, d, F, a, K \longmapsto U \longmapsto H, C, P, N.$$

The finite Holder constant row K supplies the uniform-modulus budget together with the exponent window a and the Real-valued graph row F. The handoff exports no host real power operation, function-space completeness theorem, quotient equality of functions, selected limiting function, countable-choice selector, or detached continuity principle.

Proof. Project the carrier of Definition 7000.1. The only modulus-facing coordinate is U, and U is displayed only after the metric source, Real-valued function-family row, exponent window, and Holder constant have been read.

Use the Holder continuity surface of chapter 1766. It supplies the power-law route from the finite metric and graph rows to a modulus row, while the Holder inequality sibling of chapter 2100 accounts for the finite comparison budget visible in K.

The uniform-modulus consumer therefore receives exactly the row U. Transport through H, replay through C, provenance through P, and the local name N keep that row attached to the same displayed source packet.

Since the carrier has no coordinate for host real powers, completeness, quotient function equality, selected limits, countable choice, or a detached continuity theorem, none of those objects can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{HolderSpace}^\uparrow}$**.**

Constructive story: A $\mathsf{HolderSpace}^\uparrow$ packet is generated from a MetricUp source, Real-valued ContinuousMap family rows, a positive rational exponent window, a finite Holder constant ledger, a UniformModulus export row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HolderSpace}^\uparrow}$, seed)

Scope closed: Definition 7000.1 and Definition 7000.2 seed the finite Holder-space route from displayed metric and Real-valued graph data to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HolderSpace}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real powers, function-space completeness, quotient equality of functions, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponent-window stability, Holder constant transport, UniformModulus handoff, replay preservation, and ledger nonescape for BEDC.Derived.HolderSpaceUp.

Chapter 7001

A Concrete Naming Certificate for $\mathsf{TranscendentalGapLedger}^{\uparrow}$

The $\mathsf{TranscendentalGapLedger}^{\uparrow}$ packet realizes the finite GAP-socket ledger behind chapter 9393. It sits beside the finite trace boundary of chapter 498, the inscription-point packet of chapter 610, and the supply request ledger of chapter 542. The packet records where an induction-facing universal request appears, which finite traces have been observed, which forward-binding belief socket would be needed, and why the request remains a visible Gap row rather than a substrate theorem.

Definition 7001.1 (Transcendental GAP ledger carrier)**.** *A* $\mathsf{TranscendentalGapLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Gamma = (S, T, B, G, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the socket-site row naming the proof, paper claim, or construction site at which an induction-facing universal request is exposed.*
2. *T is the finite observation-trace row, recording the displayed initial cases, bounded successors, or finite prefixes that have actually been inspected.*
3. *B is the forward-binding belief socket row. It names the present commitment that would treat not-yet-observed cases as discharged without storing that commitment as an internal theorem.*
4. *G is the* Gap *exposure row, keeping the missing universal supply visible to audit consumers.*
5. *L is the supply-boundary ledger row. It records the relation to* $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$ *and refuses to turn supply position into substrate data.*
6. *H records componentwise* hsame *transport for the socket site, finite trace row, belief socket, GAP exposure, supply-boundary ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the socket site through the finite traces, belief socket, GAP exposure, supply boundary, provenance, and local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{TranscendentalGapLedger}^{\uparrow}$*,* $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$*,* BHist*,* Gap*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TranscendentalGapLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unrestricted induction rule, universal quantifier over future observations, hidden witness, completed supply source, quotient equality, self-certifying truth row, or bridge-checked theorem.

Theorem 7001.2 (Transcendental GAP ledger NameCert obligations)**.** *For every accepted* TranscendentalGapLedge *carrier* $\Gamma = (S, T, B, G, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TranscendentalGapLedger}^{\uparrow}}$ *surface consists exactly of socket-site admission, finite observation-trace exposure, forward-binding belief-socket display,* Gap *exposure, supply-boundary ledger routing, componentwise transport, displayed replay, provenance, and local naming. Every consumer factors through*

$$S \to T \to B \to G \to L \to H \to C \to P \to N$$

before it can cite the ledger.

Proof. Project the carrier of Definition 7001.1. The rows S, T, B, G, L are the only substantive coordinates, and H, C, P, N are the only structural coordinates.

The finite trace row T is read before the belief socket B. Thus the packet records exactly what has been observed before exposing the request that would bind all not-yet-observed cases.

The GAP row G and supply-boundary row L prevent discharge. They route the request to the transcendental supply ledger surface without adding the requested universal rule to the substrate.

Transport, replay, provenance, and local naming preserve that same finite sequence. Since no other coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 7001.3 (Transcendental GAP ledger non-discharge)**.** *An accepted* $\mathsf{TranscendentalGapLedger}^{\uparrow}$ *packet may expose a forward-binding belief socket, but it never supplies the missing universal claim named by that socket. The finite observation trace* T *remains finite, the* Gap *row* G *remains visible, and the supply-boundary ledger* L *records non-internality rather than importing an induction principle.*

Proof. Read the finite trace row T. It contains only displayed observations and bounded replay data, so it cannot already contain the universal discharge.

Read the belief socket B and GAP row G. The socket names the commitment that would be needed, while the GAP row keeps the missing discharge exposed to audit.

Finally read L, H, C, P, N. These rows transport, replay, source, and name the non-discharge boundary. None adds an unrestricted induction rule, hidden witness, completed supply source, quotient equality, self-certifying truth row, or bridge-checked theorem. □

Falsifiable prediction: Every accepted $\mathsf{TranscendentalGapLedger}^{\uparrow}$ consumer factors through a finite observation-trace row, a forward-binding belief socket, a GAP exposure row, a supply-boundary ledger, componentwise transport, replay, provenance, and local naming before it can cite an induction-facing request.

Independence witness: finite-trace-induction-gap, inscriptionpoint, transcendentalsupplyledger, apophaticname: $\mathsf{TranscendentalGapLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite observation traces, forward-binding belief sockets, GAP exposure, supply-boundary ledger rows, transport, replay, provenance, and local naming.

The Lean follow-up must provide `Nontrivial TranscendentalGapLedgerUp` and `FieldFaithful TranscendentalGapLedgerUp` instances for the same carrier surface in `lean4/BEDC/Derived/TranscendentalGapl`

Closure status for $\mathsf{NameCert}_{\mathsf{TranscendentalGapLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TranscendentalGapLedger}^{\uparrow}$ packet is generated from a socket site, finite observation traces, a forward-binding belief socket, GAP exposure, a supply-boundary ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TranscendentalGapLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7001.1, Definition 7001.2, and Definition 7001.3 seed the finite ledger for induction-facing GAP sockets and their supply boundary.

Formal verification: FormalStatus(NameCert$_{\text{TranscendentalGapLedger}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted induction, universal quantification over future observations, hidden witnesses, completed supply sources, quotient equality, self-certifying truth, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-trace exposure, belief-socket display, GAP non-discharge, supply-boundary routing, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.TranscendentalGapLedgerUp.

Chapter 7002

RealNameBallFilter NameCert Construction

Following Bishop–Bridges constructive real analysis and Kelley filter terminology, the $\mathsf{RealNameBallFilter}^{\uparrow}$ packet records the finite ball-filter reads by which a displayed $\mathsf{Real}^{\uparrow}$ name is inspected through dyadic neighbourhoods. It sits between the dyadic centre-radius surface of chapter 311, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the principal Cauchy filter route of chapter 3994, and the terminal real-name seal of chapter 38. The packet supplies a finite neighbourhood interface for Real and Completion consumers; it exports no quotient real equality, selected nearest point, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7002.1 (Real name ball-filter carrier)**.** *A* $\mathsf{RealNameBallFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, R, D, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *name seal whose finite neighbourhood reads are being exposed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the requested prefix and precision window of the real name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream window.*
4. *D is the* $\mathsf{DyadicBall}^{\uparrow}$ *centre-radius row that records the displayed dyadic neighbourhood and its radius budget.*
5. *Q is the dyadic tolerance and containment ledger comparing the readback row R with the ball row D at the requested precision.*
6. *F is the* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *principal-neighbourhood row generated from the same displayed real-name window and dyadic ball.*
7. *H records componentwise* hsame *transport for the seal, window, readback, ball, tolerance, principal-filter, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite ball request through S, R, D, Q, F before the terminal seal E is consumed.*
9. *P is the* Pkg *provenance over* $\mathsf{RealNameBallFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{CauchyPrincipalFilter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealNameBallFilter}^{\uparrow}}$ *row naming exactly this finite ball-filter packet.*

The classifier compares accepted packets only through $E, S, R, D, Q, F, H, C, P, N$. It has no coordinate for quotient real equality, selected nearest point, countable choice, open-cover compactness, hidden infinite filter bases, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7002.2 (RealNameBallFilter NameCert obligations)**.** *For every accepted* $\mathsf{RealNameBallFilter}^{\uparrow}$ *carrier $B = (E, S, R, D, Q, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealNameBallFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed real seal, stream window, regular rational readback, dyadic ball, tolerance ledger, principal-filter row, transport, replay, provenance, and local naming row;*
2. *window locality, requiring every ball request to inspect the real name only through the finite* $\mathsf{StreamName}^{\uparrow}$ *row S;*
3. *readback exactness, requiring the regular rational row R to be the only readback surface generated from S;*
4. *ball containment exactness, requiring the dyadic centre-radius row D and tolerance ledger Q to bind the requested neighbourhood before the principal-filter row F is exposed;*
5. *principal-filter compatibility, requiring F to use the same displayed ball-filter data rather than a hidden basis or selected nearest point;*
6. *transport, replay, provenance, and name exhaustion through H, C, P, N.*

No local NameCert obligation is exported for quotient real equality, selected nearest points, countable choice, open-cover compactness, hidden infinite filter bases, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 7002.1. The visible analytic rows are exactly the real seal E, the finite stream window S, the regular rational readback R, the dyadic ball D, the tolerance ledger Q, and the principal-filter row F.

Window locality and readback exactness follow from the ordered route in the carrier. A finite neighbourhood request first opens S and then reads the same displayed entries through R; there is no second stream, hidden representative, or quotient-real equality coordinate.

Ball containment is then checked by D and Q. These rows record the centre, radius, and tolerance comparison before F may package the displayed neighbourhood as a principal filter row.

Finally, H and C preserve the same route under componentwise transport and replay, while P and N source and name only the displayed packet. Since the excluded objects are absent from the carrier inventory, they cannot appear as additional NameCert obligations. □

Theorem 7002.3 (RealNameBallFilter Real seal nonescape)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{RealNameBallFilter}^{\uparrow}$ *carrier reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{CauchyPrincipalFilter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently the exported read is a finite ball-filter neighbourhood of the displayed real name, not quotient real equality, not a selected nearest point, not countable choice, not open-cover compactness, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer read of the seal E. By Definition 7002.1, E is packaged with the displayed stream window S, regular rational readback R, dyadic ball row D, tolerance ledger Q, and principal-filter row F.

Apply the obligation surface of Definition 7002.2. The request must first inspect S, then read R, then bind the requested ball through D and Q before F can be exposed.

Only after that route is replayed may the Real-facing consumer use the seal row. The structural rows H, C, P, N transport, continue, source, and name the same finite packet, so the exported object remains a displayed ball-filter read around the real name.

The forbidden principles are not coordinates of the carrier. Hence no path through the local certificate can produce quotient real equality, a selected nearest point, countable choice, open-cover compactness, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealNameBallFilter}^{\uparrow}$ read can be audited by checking that the consumer first displays a $\mathsf{StreamName}^{\uparrow}$ finite window, then a $\mathsf{RegSeqRat}^{\uparrow}$ readback, then a $\mathsf{DyadicBall}^{\uparrow}$ neighbourhood and tolerance ledger, then the $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ row, before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Independence witness: dyadicball, streamname, regseqrat, real, cauchy-principal-filter: $\mathsf{RealNameBallFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the real-name seal, finite stream window, regular rational readback, dyadic ball, tolerance ledger, principal-filter handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RealNameBallFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RealNameBallFilter}^{\uparrow}$ packet is generated from a RealUp seal, finite StreamName window, RegSeqRat readback, DyadicBall neighbourhood, tolerance ledger, CauchyPrincipalFilter row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealNameBallFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7002.1, Definition 7002.2, and Definition 7002.3 seed the finite ball-filter read route around displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealNameBallFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected nearest points, countable choice, open-cover compactness, hidden infinite filter bases, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite window locality, regular rational readback, dyadic ball containment, principal-filter compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealNameBallFilterUp.

Chapter 7003

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ packet records the regular-Cauchy completion functor as finite BEDC naming data over inspected rational approximants. It sits after the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the completion-functor surface of chapter 423. The packet names the Bishop regular completion handoff visible to L10 consumers; it is not a quotient completion, selected limit representative, host equality of completed objects, countable-choice schedule, or ambient completeness theorem.

Definition 7003.1 (BishopRegularCompletionFunctor carrier)**.** *A* $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for regular precision requests and comparison of inspected rational windows.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the rational observations that a completion-facing consumer may inspect.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row certifying that the displayed stream windows satisfy the regular rational discipline.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from D, S, R through the visible regular-Cauchy route.*
5. *Q is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row, recording the source-to-completion action only after D, S, R, E have been displayed.*
6. *H records componentwise* hsame *transport for tolerance ledgers, stream windows, regular readbacks, real seals, functor rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a regular precision request through*

$$D, S \longmapsto R \longmapsto E \longmapsto Q \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, Q, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7003.2 (BishopRegularCompletionFunctor NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ *carrier*

$$F = (D, S, R, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for dyadic tolerances, finite stream windows, regular rational readback, real-name sealing, completion-functor handoff, transport, replay, provenance, and local naming; source exposure requiring every completion-facing read to inspect D, S before R; seal discipline requiring E to be read only through the same D, S, R route; and functorial handoff requiring Q to preserve that displayed route without exporting quotient completion data.*

Proof. Project the accepted carrier. Its displayed coordinates are exactly $D, S, R, E, Q, H, C, P, N$, so carrier admission, transport, replay, provenance, and local naming are read from those rows.

The source side begins with D and S. The dyadic ledger fixes the tolerances used by each regular request, while the stream-window row specifies the finite approximants available for the current read.

The regular readback row R and the seal row E are downstream of those source rows. A completion-facing consumer therefore obtains a regular rational name and a real-name seal only after the dyadic and stream data have been displayed.

Finally Q hands the same route to the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ surface, and H, C, P, N preserve, replay, source, and name the finite packet. Since no other coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 7003.3 (Regular completion functor consumer boundary)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyCompletionFunctor}^{\uparrow}$$

as displayed by $D, S, R, E, Q, H, C, P, N$. The consumer may use the completion-functor row as a named handoff from regular rational data to completion data, but it cannot extract quotient completions, selected limit representatives, host equality of completed objects, countable choice, or ambient completeness from the packet.

Proof. Begin with a consumer of the functor row Q. The carrier has no completion-facing coordinate before the dyadic ledger, stream windows, regular readback, and real seal have been exposed.

Read D, S, R, E in order. These rows provide the visible tolerance, finite-window, regular-rational, and real-name surfaces that make the Bishop regular completion handoff meaningful.

The structural rows H, C, P, N then transport, replay, source, and name the same route. They add no quotient object, representative selector, host equality, countable-choice schedule, or completeness principle. □

Falsifiable prediction: An accepted $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ consumer that reaches $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ without first reading the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctor: no listed sibling supplies the whole packet $F = (D, S, R, E, Q, H, C, P, N)$, because each sibling contributes only a tolerance ledger, a finite window interface, a regular rational readback, a real seal, or the downstream completion-functor handoff while this chapter binds the route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, a CauchyCompletionFunctor handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7003.1, Definition 7003.2, and Definition 7003.3 seed the Bishop regular completion functor packet over dyadic tolerances, finite windows, regular rational readback, Real sealing, and completion-functor handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance coverage, StreamName window exposure, RegSeqRat readback, Real seal preservation, completion-functor routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletionFunctorUp.

Chapter 7004

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}$

Following Bishop and Bridges compact-uniform-continuity arguments, the $\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}$ packet records the finite diagonal tail route by which compact finite nets and pointwise continuity data feed a regular-Cauchy Real seal. It sits after the compact metric source of chapter 129, the continuous-map row used by compact-uniform consumers, the uniform-modulus handoff of chapter 369, the diagonal schedule of chapter 6371, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet realizes the finite-net diagonal branch of chapter 9237 as NameCert data rather than as an ambient compactness or completeness principle.

Definition 7004.1 (CompactUniformCauchyDiagonal carrier)**.** *A* $\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, U, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus data are inspected on the displayed compact net.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row reached by folding the finite pointwise modulus data.*
4. *M is the* $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ *row that turns the compact finite-net schedules into one diagonal tail witness.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the finite-net fold and diagonal tail comparison.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected source and target observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D, S, U, M.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the compact, continuous, uniform-modulus, diagonal, tolerance, window, and readback rows have been displayed.*
9. *H records componentwise* hsame *transport for compact source, continuous-map, uniform-modulus, diagonal, tolerance, stream-window, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a compact finite-net tolerance request through*

$$K, F, U, M, D, S \longmapsto R \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* CompactUniformCauchyDiagonal$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, CauchyModulusDiagonal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CompactUniformCauchyDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, M, D, S, R, E, H, C, P, N$. It has no coordinate for an open-cover compactness proof, selected subcover, point selector, quotient stream, host function equality, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7004.2 (CompactUniformCauchyDiagonal NameCert obligations)**.** *For every accepted* CompactUniformCauchyDiagonal$^{\uparrow}$ *carrier*

$$G = (K, F, U, M, D, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{CompactUniformCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net exposure, continuous-map row exposure, uniform-modulus fold exposure, diagonal-tail schedule exposure, dyadic tolerance exactness, finite stream-window exposure, regular rational readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier coordinates. The compact source, continuous-map row, uniform-modulus fold, diagonal-tail schedule, dyadic ledger, finite windows, readback, and seal are all explicit entries of G.

The structural rows H and C determine how these entries move between certified presentations and how a tolerance request is replayed. They expose only componentwise sameness and displayed continuation routes.

The provenance and local naming rows P, N bind the packet to the cited sibling chapters. Since no other coordinate is present, the listed rows exhaust the NameCert obligation surface and the excluded compactness, selector, quotient, equality, completeness, verification, and bridge data have no export path. □

Theorem 7004.3 (Compact finite-net diagonal handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* CompactUniformCauchyDiagonal$^{\uparrow}$ *carrier must read the compact finite-net row K, continuous-map row F, uniform-modulus row U, diagonal-tail row M, dyadic tolerance row D, stream-window row S, and regular rational readback row R before it may read the real seal row E. The handoff exports a finite diagonal Cauchy route only; it does not export an open-cover choice, selected subnet, host function equality, quotient real representative, countable-choice schedule, or ambient completeness theorem.*

Proof. Start with a consumer asking for E. The carrier definition places E after the compact finite-net, continuous-map, uniform-modulus, diagonal, tolerance, stream-window, and readback rows.

The rows K, F, U, M, D, S, R supply the finite route: compact net, pointwise-to-uniform modulus fold, diagonal tail, dyadic tolerance, observed windows, and regular rational readback. None of these rows is a host compactness proof or selected limit.

Finally transport and replay the route through H, C, P, N. These structural rows preserve the displayed finite packet and cannot add the excluded choice, quotient, equality, or completeness data. □

Theorem 7004.4 (Compact uniform diagonal Real seal nonescape)**.** *In every accepted* CompactUniformCauchyDiag *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is a terminal row of the finite route*

$$K, F, U, M, D, S, R, H, C, P, N.$$

A downstream consumer may cite E *only as the seal of that route; it cannot detach the seal into ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality of completed reals, a selected representative, or quotient stream equality.*

Proof. Read E from the carrier. It appears only after the compact, continuous, uniform-modulus, diagonal, tolerance, window, and regular-readback rows.

The continuation row C records the same order of exposure, while H transports the rows componentwise. The provenance and local naming rows P, N source and name this route.

Therefore E is a terminal seal for the displayed packet and has no coordinate from which a detached completeness, equality, representative, or quotient principle could be read. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without reading the displayed $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CauchyModulusDiagonal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: compactmetric, continuousmap, uniformmodulus, cauchymodulusdiagonal, dyadicratcore, regseqrat, real: no listed sibling supplies the whole packet $G = (K, F, U, M, D, S, R, E, H, C, P, N$ because each sibling contributes only compact finite-net data, pointwise continuity, a uniform-modulus row, a diagonal schedule, a dyadic tolerance ledger, a regular readback, or a terminal seal while this chapter binds the compact diagonal route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise rows, a UniformModulus fold, a CauchyModulusDiagonal tail row, DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7004.1, Definition 7004.2, Definition 7004.3, and Definition 7004.4 seed the compact finite-net diagonal route from compact and continuous rows to regular rational readback and a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected limits, quotient streams, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous-map exposure, uniform-modulus fold, diagonal-tail replay, dyadic tolerance coverage, finite-window readback, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformCauchyDiagonalUp.

Chapter 7005

Bishop Sequence Limit NameCert Construction

Following Bishop and Bridges on constructive sequences of real numbers, the $\mathsf{BishopSequenceLimit}^{\uparrow}$ packet records a finite row presentation of a Bishop-style sequence limit without importing a host convergence predicate. It sits beside the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the displayed sequence-limit packet of chapter 1413, and the limit-uniqueness route of chapter 3942. The packet names the L10 handoff by finite schedules and regular Cauchy rows; it does not name quotient limits, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7005.1 (Bishop sequence-limit carrier)**.** *A* $\mathsf{BishopSequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, L, W, R, D, T, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the source sequence, exposing only the finite source windows requested by the current tolerance read.*
2. *L is the displayed candidate-limit row, read as a regular real name before any sequence-limit classifier is consumed.*
3. *W is the tail-window schedule, assigning to each dyadic tolerance request a finite observation window and a displayed tail threshold.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source windows and candidate-limit windows in the same schedule.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic budget and the comparison cells used by the scheduled tail test.*
6. *T is the Bishop tail test row, whose accepted entries say that every displayed source entry beyond the scheduled threshold is within the dyadic budget of the displayed candidate limit.*
7. *U is the* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *boundary row, used only to compare two displayed candidate-limit reads through their common regular Cauchy route.*
8. *E is the terminal* $\mathsf{RealSequenceLimit}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after S, L, W, R, D, T, U have been displayed.*

9. *H stores componentwise* hsame *transport for all displayed rows.*
10. *C stores displayed* Cont *replay from a tolerance request through the tail-window, readback, dyadic, tail-test, uniqueness, and seal rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopSequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSequenceLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of limits, a selected host limit, a countable-choice schedule, trichotomy on reals, host sequence convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopSequenceLimitUp.BishopSequenceLimitTasteGate single carrier alignment`

Theorem 7005.2 (Bishop sequence-limit NameCert obligations)**.** *For every accepted* $\mathsf{BishopSequenceLimit}^{\uparrow}$ *carrier* $K = (S, L, W, R, D, T, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopSequenceLimit}^{\uparrow}}$ *surface consists exactly of source stream-window admission, candidate-limit admission, scheduled tail-window exposure, regular rational readback, dyadic tolerance-ledger exactness, Bishop tail-test exposure, regular Cauchy uniqueness-boundary exposure, terminal* $\mathsf{RealSequenceLimit}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming for the rows of Definition 7005.1.*

Proof. Project the accepted carrier. The sequence source, candidate limit, schedule, regular readback, dyadic ledger, tail test, uniqueness boundary, and terminal seal are exactly the analytic rows S, L, W, R, D, T, U, E.

The schedule row W determines which finite source and candidate-limit windows may be inspected for a tolerance request. The readback row R and dyadic row D then attach regular rational and dyadic comparison data to that same finite schedule.

The Bishop tail test T is downstream of those rows, and the uniqueness boundary U is downstream of the same regular Cauchy route. Thus a consumer cannot read a limit assertion, a second-limit comparison, or a real-facing seal before the displayed finite route has been replayed.

Finally, H, C, P, N transport, replay, source, and name only the displayed packet. Since the carrier inventory contains no quotient-limit, selected-limit, choice, trichotomy, host-convergence, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7005.3 (Bishop sequence-limit Real-seal handoff)**.** *Every* $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegularCauchyLimitUnique}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopSequenceLimit}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RealSequenceLimit}^{\uparrow} \longmapsto \mathsf{RegularCauchyLimitUniqueness}^{\uparrow} \longmapsto$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The handoff exports only scheduled finite-window sequence-limit data; it exports no quotient limit, selected limit, countable-choice row, trichotomy split, host convergence predicate, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a Real-facing or uniqueness-facing consumer and project the carrier of Definition 7005.1. The only source observations are the finite stream windows S selected by W, and the only candidate-limit observations are the regular readbacks R measured by the dyadic ledger D.

The tail-test row T admits a Bishop sequence-limit read only through that finite route. If a second candidate-limit read is compared, the boundary row U sends the comparison through $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ before either Real seal is public.

The terminal row E is therefore reached only after the scheduled StreamName, RegSeqRat, DyadicRatCore, RealSequenceLimit, and regular Cauchy uniqueness rows have been displayed. The structural rows H, C, P, N preserve the same packet and add no source from which quotient limits, selected limits, countable choice, trichotomy, host convergence, or ambient completeness could be recovered. □

Falsifiable prediction: Every accepted $\mathsf{BishopSequenceLimit}^{\uparrow}$ consumer can be audited by checking that its sequence-limit read displays a $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a Bishop tail-test row, a $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff, a $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ boundary row when uniqueness is inspected, and the terminal $\mathsf{Real}^{\uparrow}$ seal before the local name is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real-sequence-limit, regular-cauchy-limit-uniqueness, real: no listed sibling is bijective with $\mathsf{BishopSequenceLimit}^{\uparrow}$, because this packet alone binds a displayed source sequence, candidate-limit row, tail-window schedule, regular rational readback, dyadic tolerance ledger, Bishop tail test, uniqueness boundary, terminal Real seal, transport, replay, provenance, and local naming into one finite L10 sequence-limit certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSequenceLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSequenceLimit}^{\uparrow}$ packet is generated from finite StreamName sequence windows, a displayed candidate-limit row, tail-window schedules, RegSeqRat readback, DyadicRatCore tolerance rows, a Bishop tail-test row, a RegularCauchyLimitUniqueness boundary, a RealSequenceLimit and RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSequenceLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7005.1, Definition 7005.2, and Definition 7005.3 seed the finite Bishop sequence-limit route from L10 StreamName, RegSeqRat, DyadicRatCore, RealSequenceLimit, RegularCauchyLimitUniqueness, and Real data to the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSequenceLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient limits, selected limits, countable choice, trichotomy, host sequence convergence, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName schedule locality, RegSeqRat readback, DyadicRatCore tolerance exactness, Bishop tail-test exposure, RegularCauchyLimitUniqueness boundary exposure, RealSequenceLimit and Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopSequenceLimitUp.

Chapter 7006

A Concrete Naming Certificate for $\mathsf{RegulatedOscillation}^{\uparrow}$

Following Bishop and Bridges on regulated functions and finite oscillation control, the $\mathsf{RegulatedOscillation}^{\uparrow}$ packet records the displayed oscillation windows by which a regulated real-valued function is read over a finite interval cover. It sits after the regulated-function schedule of chapter 1906, the oscillation-modulus boundary of chapter 2224, the compact-interval oscillation-modulus packet of chapter 6148, the dyadic interval-cover row of chapter 1428, and the terminal real-name seal of chapter 38. The packet names finite oscillation data for regulated-integration and compact-interval consumers; it does not assert open-cover compactness, quotient equality of functions, selected extrema, a uniform modulus chosen from an ambient theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7006.1 (Regulated oscillation carrier)**.** *A* $\mathsf{RegulatedOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, O, C, D, R, H, T, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, exposing the regulated approximation schedule whose oscillation is being read.*
2. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row, carrying the requested cellwise oscillation budget.*
3. *C is the* $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$ *row, binding the oscillation budget to a displayed compact interval window.*
4. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *row, listing the finite dyadic interval cover on which the oscillation cells are inspected.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for the cellwise bounds and endpoint values exposed by the packet.*
6. *H, T, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegulatedOscillation}^{\uparrow}}$ *row.*

The classifier compares carriers only through $F, O, C, D, R, H, T, P, N$. It has no coordinate for a host function quotient, selected extremum, arbitrary open cover, unbounded refinement search, or ambient real-completeness principle.

Theorem 7006.2 (Regulated oscillation NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedOscillation}^{\uparrow}$ *carrier $Q = (F, O, C, D, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedOscillation}^{\uparrow}}$ *surface consists of*

exactly five analytic rows before the structural rows are read: the regulated-function source F, oscillation-modulus row O, compact-interval oscillation row C, dyadic interval-cover row D, and terminal Real readback R. Transport, replay, provenance, and local naming preserve only this displayed surface.

Proof. Project the carrier of Definition 7006.1. The analytic rows are precisely F, O, C, D, R, and the structural rows are H, T, P, N.

The source row F supplies the regulated approximation schedule. The row O attaches the requested oscillation budget to that schedule, while C restricts the read to the displayed compact interval window.

The dyadic cover row D supplies the finite cells on which the oscillation bounds are inspected. The terminal Real row R records the cellwise bound and endpoint readback only after the first four analytic rows are visible.

Since no further analytic coordinate occurs in the packet, H, T, P, N can only transport, replay, source, and name the same five-row surface. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 7006.3 (Regulated oscillation finite partition nonescape)**.** *Every regulated-integration or compact-interval consumer of an accepted* $\mathsf{RegulatedOscillation}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longrightarrow \mathsf{OscillationModulus}^{\uparrow} \longrightarrow \mathsf{CompactIntervalOscillationModulus}^{\uparrow} \longrightarrow \mathsf{DyadicIntervalCover}^{\uparrow} \longrightarrow$$

before it may use any oscillation bound. The consumer receives a finite partition-and-readback surface, not open-cover compactness, quotient equality of functions, selected extrema, unbounded refinement search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 7006.2. They force every consumer to expose the regulated source, oscillation budget, compact-interval window, dyadic interval cover, and Real readback before the structural rows may replay the request.

The sibling chapters provide those rows in order: chapter 1906, chapter 2224, chapter 6148, chapter 1428, and chapter 38. Thus the consumer reads a displayed finite route rather than a detached compactness or function-extensionality theorem.

The exclusions follow from the carrier inventory. No row stores an arbitrary open cover, selected extremum, quotient function equality, unbounded refinement search, or ambient real-completeness assertion, so the finite partition route cannot export any of them. □

Falsifiable prediction: Any accepted RegulatedOscillationUp packet that exports an oscillation read without displaying the regulated-function source, oscillation modulus, compact-interval oscillation row, dyadic interval cover, and Real readback refutes the carrier.

Independence witness: regulated-function, oscillation-modulus, compact-interval-oscillation-modulus, dyadic-interval-cover, real: none is bijective with $\mathsf{RegulatedOscillation}^{\uparrow}$ because this packet jointly binds the regulated source, oscillation budget, compact interval window, finite dyadic cover, Real readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedOscillation}^{\uparrow}$ packet is generated from a regulated-function source row, oscillation-modulus row, compact-interval oscillation row, dyadic interval-cover row, terminal Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7006.1, Definition 7006.2, and Definition 7006.3 seed the finite regulated-oscillation surface over RegulatedFunctionUp, OscillationModulusUp, CompactIntervalOscillationModulusUp, DyadicIntervalCoverUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedOscillation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality of functions, selected extrema, unbounded refinement search, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, five-row NameCert exposure, finite partition nonescape, and a Lean carrier or obligation formalization for BEDC.Derived.RegulatedOscillationUp.

Chapter 7007

A Concrete Naming Certificate for $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$

Following Kelley and Bourbaki on subbases and cofinite tail filters, the packet records the finite BEDC route from $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ through $\mathsf{FrechetFilter}^{\uparrow}$ and $\mathsf{FilterBase}^{\uparrow}$ to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It is grounded in the subbasis generation packet of chapter 4232, the cofinite-tail filter packet of chapter 1564, the filter-base interface of chapter 1601, the finite window naming surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite tail-factorization object only; it does not quotient filters, select limits, invoke countable choice, use host equality of generated filters, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7007.1 (Cauchy subbasis tail-factorization carrier)**.** *A* $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, A, F, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *row exposing primitive subbasis tokens, finite intersections, generated events, and the generated-base ledger.*
2. *A is the tail-specialization row saying that the requested generated-base reads have been restricted to eventual sequence tails.*
3. *F is the* $\mathsf{FrechetFilter}^{\uparrow}$ *row carrying the cofinite tail threshold discipline.*
4. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained after the subbasis route has passed through the Frechet tail surface.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by completion-facing consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the retained tail windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, A, F, B, W, R have been displayed.*
8. *H records componentwise* hsame *transport for subbasis rows, tail-specialization rows, Frechet thresholds, filterbase handoffs, stream windows, regular readbacks, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from generated subbasis reads through cofinite tail thresholds, filterbase handoffs, stream windows, regular readbacks, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$, $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubbasisTailFactorization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient filters, ultrafilter completion, selected limits, countable-choice schedules, host equality of generated filters, or ambient Real completeness.

Theorem 7007.2 (Cauchy subbasis tail-factorization NameCert obligations)**.** *For every accepted* $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$ *carrier* $T = (S, A, F, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySubbasisTailFac}}$ *surface consists exactly of carrier admission;* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *generated-base exposure; cofinite-tail specialization;* $\mathsf{FrechetFilter}^{\uparrow}$ *threshold exposure;* $\mathsf{FilterBase}^{\uparrow}$ *handoff;* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;* $\mathsf{Real}^{\uparrow}$ *sealing; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; and local naming.*

Proof. Project the carrier rows from Definition 7007.1. The row S supplies only the finite subbasis generation surface, and the row A narrows that surface to eventual sequence tails.

Read the cofinite-tail route next. The $\mathsf{FrechetFilter}^{\uparrow}$ row F supplies the threshold discipline used by the specialized tail reads, and the $\mathsf{FilterBase}^{\uparrow}$ row B is visible only after that cofinite route has been displayed.

The completion-facing rows are downstream. The stream window row W, regular rational readback row R, and terminal Real seal row E are reached after the generated subbasis and Frechet tail rows have been opened.

Finally the structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier coordinates supplies quotient filters, selected limits, countable choice, host equality, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7007.3 (Cauchy subbasis tail-factorization Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$ *packet must pass through*

$$\mathsf{CauchyFilterSubbasis}^{\uparrow} \longmapsto \mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the Real seal is public only after the subbasis-generated Cauchy read has been specialized to cofinite tails, routed through the Frechet threshold discipline, handed to a displayed filter base, and read back through finite stream windows and regular rational names.

Proof. Start with the carrier $T = (S, A, F, B, W, R, E, H, C, P, N)$. The rows S and A expose the generated subbasis read and the tail specialization before any completion-facing object can be inspected.

Apply the NameCert obligations of Definition 7007.2. They force the Frechet threshold row F and the generated filterbase handoff B to be displayed before the stream, regular-readback, and Real-seal rows are visible.

Use the sibling routes of chapter 4232 and chapter 1564. The former exposes the subbasis-generated base, while the latter supplies the cofinite tail discipline used by eventual sequence reads.

The rows W, R, E then complete the route through chapter 126, chapter 301, and chapter 38. The structural rows preserve the same dependency, so no quotient filter, selected limit, countable-choice schedule, host equality of generated filters, or ambient completeness object can bypass the displayed handoff. □

Falsifiable prediction: Every accepted $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$ Real-facing read exposes a CauchyFilterSubbasisUp row, cofinite tail-specialization row, FrechetFilterUp threshold row, FilterBaseUp handoff, StreamNameUp finite-window row, RegSeqRatUp readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cauchy-filter-subbasis, frechetfilter, filterbase, streamname, regseqrat, real: no listed sibling supplies the whole packet $T = (S, A, F, B, W, R, E, H, C, P, N)$, because each sibling contributes only a subbasis-generation row, cofinite-tail row, filterbase handoff, finite-window row, regular readback, or terminal seal while this chapter binds them into one tail-factorization carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubbasisTailFactorization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubbasisTailFactorization}^{\uparrow}$ packet is generated from a CauchyFilterSubbasis row, a cofinite tail-specialization row, a FrechetFilter threshold row, a FilterBase handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubbasisTailFactorization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7007.1, Definition 7007.2, and Definition 7007.3 seed the finite route from generated Cauchy subbases through cofinite tails to Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySubbasisTailFactorization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality of generated filters, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subbasis-generated exposure, cofinite-tail specialization, Frechet threshold exposure, FilterBase handoff, StreamName windows, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySubbasisTailFactorizationUp.

Chapter 7008

A Concrete Naming Certificate for $\mathsf{SignChangeBisection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SignChangeBisection}^{\uparrow}$ packet records the finite bisection route attached to an interval whose endpoint reads have opposite signs. It sits after the located endpoint-plus-radius interval of chapter 1709, the retained-branch bisection surface of chapter 1711, the located-zero witness of chapter 6387, and the theorem-object route of chapter 3676. The chapter names the finite sign-change ledger consumed by later located-zero and interval-search rows; it does not select a root, decide trichotomy, identify quotient real names, invoke open-cover compactness, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7008.1 (Sign-change bisection NameCert surface)**.** *A* $\mathsf{SignChangeBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, S, Z, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, read through Definition 1709.1 and carrying endpoint names, radius data, locator reads, locatedness moduli, and metric bracket data.*
2. *L is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *row, read through Definition 1711.1, recording the finite retained-branch history, midpoint rows, retained windows, and regular readback route.*
3. *S is the endpoint and midpoint sign-change ledger. It records only displayed finite sign reads and the retained sign boundary transported along L.*
4. *Z is the* $\mathsf{BishopLocatedZero}^{\uparrow}$ *consumer row, read through Definition 6387.1, which may inspect the bracket, modulus, bisection schedule, readback, and Real seal after I, L, S have been displayed.*
5. *T is the* $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *theorem-object row, read through Definition 3676.1, supplying the finite interval, map, sign, window, readback, and Real-seal route used by theorem-facing consumers.*
6. *H records componentwise* hsame *transport for the interval, bisection, sign ledger, located-zero, theorem-object, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a sign-change bisection request through I, L, S, Z, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SignChangeBisection}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{BishopLocatedZero}^{\uparrow}$, $\mathsf{IntermediateValueTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{SignChangeBisection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a selected root, host interval equality, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7008.2 (Sign-change bisection located-zero route)**.** *Every accepted* $\mathsf{SignChangeBisection}^{\uparrow}$ *packet routes a located-zero read through*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{LocatedIntervalBisection}^{\uparrow} \longmapsto S \longmapsto \mathsf{BishopLocatedZero}^{\uparrow}$$

before any theorem-object consumer may cite the $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *row. The route exposes the interval source of Definition 1709.4, the retained-window bisection route of Definition 1711.3, and the zero-witness bisection discipline of Definition 6387.3; it exports no selected global root or ambient intermediate-value principle.*

Proof. Project the carrier of Definition 7008.1. The source interval row I is the only coordinate carrying endpoint and locatedness data, and the bisection row L is the only coordinate carrying retained midpoint and window data.

Apply the located interval exactness route and the located interval bisection nested-window route. These force every interval-search read to pass first through the displayed endpoint-radius data and then through the finite retained branch route.

The sign ledger S is attached to that branch route, and the located-zero row Z may read it only as part of the same finite packet. Bishop located zero bisection soundness then places the zero-witness read behind the bracket, modulus, schedule, regular readback, and Real-seal discipline.

Finally, the theorem-object row T remains a downstream consumer. Componentwise transport, replay, provenance, and local naming preserve the same route, while the carrier contains no coordinate for root selection, trichotomy, quotient equality, open-cover compactness, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{SignChangeBisection}^{\uparrow}$ located-zero read displays a LocatedRealIntervalUp source row, a LocatedIntervalBisectionUp retained-branch row, a finite sign-change ledger, a BishopLocatedZeroUp consumer row, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: located-real-interval, located-interval-bisection, bishop-located-zero, intermediate-value-theorem: $\mathsf{SignChangeBisection}^{\uparrow}$ is not bijective with any listed sibling because it joins the source located interval, retained bisection route, transported sign-change ledger, located-zero consumer, theorem-object consumer, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SignChangeBisection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SignChangeBisection}^{\uparrow}$ packet is generated from a $\mathsf{LocatedRealInterval}^{\uparrow}$ source row, a $\mathsf{LocatedIntervalBisection}^{\uparrow}$ retained-branch row, a finite sign-change ledger, a $\mathsf{BishopLocatedZero}^{\uparrow}$ consumer row, an $\mathsf{IntermediateValueTheorem}^{\uparrow}$ theorem-object row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignChangeBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7008.1 and Definition 7008.2 seed the finite sign-change bisection route over located intervals, retained bisection windows, located-zero consumers, theorem-object consumers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignChangeBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected roots, host interval equality, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-ledger visibility, located-zero routing, theorem-object routing, transport, replay, provenance, and local naming for BEDC.Derived.SignChangeBisectionUp.

Chapter 7009

A Concrete Naming Certificate for EffectiveRegularRealExtensionality$^{\uparrow}$

Following Bishop and Bridges, and the computable-analysis reading of effective regular names, the EffectiveRegularRealExtensionality$^{\uparrow}$ packet records the finite substitution surface where two displayed regular real names may be reused extensionally without selecting quotient representatives. It sits after the regular Cauchy equality surface of chapter 1930, the Bishop real equality surface of chapter 6491, the entourage basis of chapter 5115, and the Cauchy filter subbasis of chapter 4232. It also consumes the finite stream and regular-rational rows of chapter 126, chapter 301, chapter 291, and chapter 38.

Definition 7009.1 (Effective regular real extensionality carrier)**.** *An* EffectiveRegularRealExtensionality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R_0, R_1, S_0, S_1, D_0, D_1, W_0, W_1, A_0, A_1, H, C, P, N)$$

with the following displayed rows:

1. R_0, R_1 *are two* RegSeqRat$^{\uparrow}$ *regular readbacks presented as the effective names being compared.*
2. S_0, S_1 *are the two finite* StreamName$^{\uparrow}$ *windows through which the comparison is inspected.*
3. D_0, D_1 *are the* DyadicRatCore$^{\uparrow}$ *tolerance ledgers recording the precision requests and endpoint bounds for the two windows.*
4. W_0, W_1 *are effective witness-budget rows, each naming the finite requests needed to replay the displayed comparison at the requested precision.*
5. A_0, A_1 *are the two terminal* Real$^{\uparrow}$ *seal rows exposed only after the regular readbacks, stream windows, dyadic tolerance ledgers, and witness budgets have been displayed.*
6. H *records componentwise* hsame *transport for both readbacks, both windows, both dyadic ledgers, both witness budgets, both real seals, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from the finite windows through the regular readbacks and dyadic tolerance ledgers into the two real seals.*
8. P *is the* Pkg *provenance over* RegularCauchyExtensionality$^{\uparrow}$, BishopRealExtensionality$^{\uparrow}$, RealUniformEntourageBas CauchyFilterSubbasis$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R_0, R_1, S_0, S_1, D_0, D_1, W_0, W_1, A_0, A_1, H, C, P, N$. It has no coordinate for quotient real equality, a selected representative, countable choice, trichotomy, ambient Cauchy completion, or host real equality.

Theorem 7009.2 (Effective regular real extensionality NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ *carrier $E = (R_0, R_1, S_0, S_1, D_0, D_1, W_0, W_1, A_0, A_1, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}}$ *surface consists exactly of carrier admission, exposure of the two regular readbacks, exposure of the two finite stream windows, exposure of the two dyadic tolerance ledgers, exposure of the two effective witness-budget rows, exposure of the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 7009.1. The analytic rows are precisely the two regular readbacks, two finite windows, two dyadic ledgers, two witness-budget rows, and two real seals.

The structural rows preserve the same packet. The row H transports each displayed coordinate by hsame, C replays the route from window evidence to readback and seal evidence, P records the cited siblings, and N names the local certificate.

Since the carrier contains no further coordinate, every local NameCert obligation is one of those rows. The excluded quotient, representative-selection, choice, and host-equality principles cannot be exported from the packet. □

Theorem 7009.3 (Effective regular real computable handoff)**.** *Every computable consumer of an accepted* $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ *packet may reuse a regular real name only through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff is witnessed by the two finite stream windows, the two regular readbacks, the two dyadic tolerance ledgers, and the two effective witness-budget rows; it is not a request for a quotient representative or a global equality decision.

Proof. Read the two finite windows S_0, S_1 and their regular readbacks R_0, R_1. The dyadic ledgers D_0, D_1 supply the finite precision data, while W_0, W_1 record the effective budgets needed to replay the comparison at those precisions.

The route then reaches A_0, A_1 only through displayed replay. Thus a computable consumer can reuse the regular-name data by following the finite window-readback-ledger route.

No selected representative or global equality decision appears in the carrier. The exported handoff is therefore a finite effective substitution surface rather than a quotient-real principle. □

Theorem 7009.4 (Effective regular real Real-seal boundary)**.** *The two* $\mathsf{Real}^{\uparrow}$ *seal rows of an accepted* $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ *carrier are boundary rows for finite regular-name substitution. A consumer reaches A_0, A_1 only after the corresponding stream windows, regular readbacks, dyadic ledgers, witness budgets, transport, replay, provenance, and local name have been displayed.*

Proof. Use the carrier order. The seals A_0, A_1 are stored beside the window, readback, dyadic, and witness-budget rows, and the replay row C is the only displayed route by which a consumer reaches them.

Transport by H does not add a hidden seal source; it preserves the same coordinates. Provenance P and local naming N identify the packet but do not manufacture quotient equality or representative selection.

Therefore the real seals mark the boundary of the finite substitution certificate. Any consumer that bypasses the displayed rows is outside the $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ surface. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ packet displays two regular readbacks, two StreamName windows, two DyadicRatCore tolerance ledgers, two effective witness budgets, two RealUp seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before extensional regular-name reuse is exported.

Independence witness: regular-cauchy-extensionality, bishop-real-extensionality, real-uniform-entourage-basis, cauchy-filter-subbasis, streamname, regseqrat, dyadicratcore, real: EffectiveRegularRealExtensional is not bijective to any listed sibling because it jointly carries two effective regular-name readbacks, two finite stream windows, two tolerance ledgers, two witness budgets, two real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}$ packet is generated from two regular readbacks, two finite StreamName windows, two DyadicRatCore tolerance ledgers, two effective witness budgets, two RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7009.1, Definition 7009.2, Definition 7009.3, and Definition 7009.4 seed the finite effective extensional-substitution packet for regular real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveRegularRealExtensionality}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.EffectiveRegularRealExtensionalityUp`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, trichotomy, ambient Cauchy completion, host real equality, theorem-checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-window readback, dyadic tolerance routing, witness-budget replay, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveRegularRealExtensionalit

Lean4 (statement) `BEDC.Derived.EffectiveRegularRealExtensionalityUp`

Chapter 7010

A Concrete Naming Certificate for $\mathsf{OrderedVectorSpace}^{\uparrow}$

Following Bourbaki and Aliprantis–Border on ordered vector spaces, the $\mathsf{OrderedVectorSpace}^{\uparrow}$ packet records the finite BEDC naming surface for a vector space equipped with a displayed positive cone whose induced preorder is compatible with addition and positive scalar multiplication. It sits beside the algebraic vector carrier of chapter 47, the cone membership boundary of chapter 2297, the preorder classifier of chapter 52, and the real scalar seal of chapter 38. The packet names only the finite ordered-algebra surface; it is not a lattice envelope, a Riesz decomposition principle, a Hahn-Banach separation theorem, an order-unit representation, or a host order completion.

Definition 7010.1 (Ordered vector space NameCert surface)**.** *An* $\mathsf{OrderedVectorSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (V, P, R, A, S, K, H, C, G, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *carrier row naming the vector endpoint, real scalar endpoint, zero vector, addition, additive inverse, scalar action, and finite vector-space law ledger.*
2. *P is the* $\mathsf{PositiveCone}^{\uparrow}$ *row on the same vector endpoint, recording cone membership, zero inclusion, addition closure inside the cone, and positive-real scalar closure.*
3. *R is the* $\mathsf{Preorder}^{\uparrow}$ *classifier row whose comparison is induced exactly by the displayed cone test: $x \leq_O y$ is read only through the row asserting $y - x \in P$.*
4. *A records addition compatibility: if $x \leq_O y$, then the displayed addition row carries $x+z \leq_O y+z$ by reusing the same cone difference row.*
5. *S records positive-scalar compatibility: if $0 \leq r$ is read through the* $\mathsf{Real}^{\uparrow}$ *scalar seal and $x \leq_O y$, then the displayed scalar-action row carries $rx \leq_O ry$ through the cone closure row.*
6. *K records cone closure and exactness: the preorder row has no accepted comparison unless the corresponding difference row has been admitted by P, and every admitted cone difference supplies the comparison row.*
7. *H records componentwise* hsame *transport for the vector-space carrier, cone row, preorder classifier, addition-compatibility row, scalar-compatibility row, cone-closure row, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from an ordered-vector-space request through V, P, R, A, S, K, H back to the same finite ordered-algebra surface.*
9. *G is the* Pkg *provenance over* $\mathsf{OrderedVectorSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OrderedVectorSpace}^{\uparrow}}$ *row binding these displayed rows as one ordered-vector-space certificate.*

The classifier compares accepted packets only through $V, P, R, A, S, K, H, C, G, N$. It has no coordinate for a lattice operation, a Riesz decomposition witness, Hahn-Banach separation, an order unit, a selected dual cone, or an ambient host order relation.

Theorem 7010.2 (Ordered vector space positive-cone obligations)**.** *For every accepted* $\mathsf{OrderedVectorSpace}^{\uparrow}$ *carrier $O = (V, P, R, A, S, K, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{OrderedVectorSpace}^{\uparrow}}$ *surface exports exactly the cone-induced preorder, addition monotonicity, positive-scalar monotonicity, and nonescape from host order. A downstream consumer may read $x \leq_O y$ only by reading the displayed difference row $y - x \in P$; it may transport this comparison along displayed addition and positive-real scalar-action rows; and it receives no order comparison outside the carrier rows $V, P, R, A, S, K, H, C, G, N$.*

Proof. Carrier admission is projection from Definition 7010.1. The tuple displays exactly the vector-space row V, the positive-cone row P, the preorder row R, the addition and positive-scalar compatibility rows A, S, the cone-exactness row K, and the structural rows H, C, G, N.

The preorder obligation is therefore cone-induced. The only comparison exported by R is the one whose certificate is the displayed cone row for $y - x$, and K records the converse handoff from an admitted cone difference back to the comparison row.

Addition monotonicity is read through A: translating both endpoints preserves the same displayed difference row, so the comparison after translation is again admitted only through P and R.

Positive-scalar monotonicity is read through S: the scalar is first sealed by the $\mathsf{Real}^{\uparrow}$ row as positive, then the scalar-action row transports the cone difference through the closure data carried by P.

Finally H, C, G, N transport, replay, source, and name only those displayed coordinates. Since the carrier has no lattice row, no Riesz decomposition row, no separation functional, no order-unit representation, and no ambient host order coordinate, none of those objects can be exported by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{OrderedVectorSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{OrderedVectorSpace}^{\uparrow}$ packet is generated from a VecSpace carrier, a PositiveCone row, a cone-induced Preorder classifier, addition and positive-scalar compatibility rows, cone exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OrderedVectorSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7010.1 and Definition 7010.2 seed the finite ordered-algebra certificate over VecSpaceUp, PositiveConeUp, PreorderUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OrderedVectorSpace}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.OrderedVectorSpaceUp`

Bridge status: none

Not claimed: This chapter does not close lattice or Riesz completeness, Hahn-Banach separation, order-unit representation, Lean verification of the ordered-algebra obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cone-induced preorder exactness, addition monotonicity, positive-scalar monotonicity, transport, replay, provenance, and local naming for BEDC.Derived.OrderedVectorSpaceUp.

Chapter 7011

A Concrete Naming Certificate for $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ packet records the regular Cauchy representation of a real name as finite BEDC data. It sits after the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located-completion boundary of chapter 1455. The packet names a representation route for L10 Real and Completion consumers without asserting completeness, quotient real equality, trichotomy, countable choice, or host real primitives.

Definition 7011.1 (Bishop regular real representation carrier)**.** *A* $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, Q, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger, exposing the finite inspected windows of the representing stream at requested precisions.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, recording that the displayed windows are read as a regular rational sequence.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare displayed rational windows and record the regular Cauchy budget.*
4. *K is the regular closeness classifier, accepting only reads whose visible windows satisfy the dyadic tolerance rows required by the regular sequence discipline.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, Q, D, K have been displayed.*
6. *H records componentwise* hsame *transport for stream windows, regular rational source rows, dyadic tolerances, closeness classification, real sealing, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay along the route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto K \longmapsto \mathsf{Real}^{\uparrow}.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopRegularRealRepresentation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, Q, D, K, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected representative, trichotomy, countable choice, host real primitives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRegularRealRepresentationUp.TasteGate.BishopRegularRealRepresentationTasteGa`
`single carrier alignment`

Theorem 7011.2 (Bishop regular real representation NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ *carrier $R = (W, Q, D, K, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealRepresent}}$ *surface consists exactly of carrier admission for Definition 7011.1, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *source readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparison, regular closeness classification,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The ledger nonescape obligation says that every tolerance, closeness, and seal read remains inside those rows.*

Proof. Project the accepted carrier. The displayed coordinates are precisely $W, Q, D, K, E, H, C, P, N$, so carrier admission and local naming are finite row projections.

The classifier side is K together with D. It checks regular closeness only through the displayed dyadic tolerance ledger and the inspected stream windows, not through a host real comparison.

The seal handoff is E. Since C replays the route through W, Q, D, K before E is visible, a Real-facing read cannot bypass the StreamName, RegSeqRat, or DyadicRatCore rows.

Finally H, C, P, N transport, replay, source, and name the same finite route. No remaining coordinate can export quotient real equality, trichotomy, countable choice, host real primitives, or ambient completeness. □

Theorem 7011.3 (Bishop regular real representation seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ *packet passes through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the terminal seal row E may be consumed. The route imports no quotient real equality, selected representative, trichotomy, countable-choice schedule, host real primitive, or ambient completeness theorem.

Proof. Start with the tolerance ledger. The row D supplies the finite dyadic comparison budget used by the regular closeness classifier.

Read the finite windows and regular source next. The row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ windows, and Q records that those windows are consumed as a $\mathsf{RegSeqRat}^{\uparrow}$ source.

Only then may the classifier and seal be read. The row K accepts the visible regular closeness data, and E is downstream of W, Q, D, K by the continuation row C.

The structural rows H, C, P, N preserve that order for every consumer. Since the carrier has no additional equality, choice, trichotomy, host-real, or completeness coordinate, every Real-facing read remains inside the displayed seal route. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ read exposes a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a regular closeness classifier row, and a terminal $\mathsf{Real}^{\uparrow}$ seal before any Real or Completion consumer can use the representation.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishoplocatedcompletionboundary: none is bijective to $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ because this packet jointly carries stream windows, regular rational source rows, dyadic tolerance ledgers, regular closeness classification, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularRealRepresentation}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularRealRepresentation}^{\uparrow}$ packet is generated from StreamName windows, a RegSeqRat source row, a DyadicRatCore tolerance ledger, a regular closeness classifier, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularRealRepresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7011.1, Definition 7011.2, and Definition 7011.3 seed the Bishop regular real representation packet over finite windows, regular rational readback, dyadic tolerance ledgers, closeness classification, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularRealRepresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient real equality, trichotomy, countable choice, host real primitives, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, seal handoff, ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularRealRepresentationUp.

Chapter 7012

A Concrete Naming Certificate for $\mathsf{LayeredRelationGateQuestion}^{\uparrow}$

The $\mathsf{LayeredRelationGateQuestion}^{\uparrow}$ packet realizes the question surface of chapter 9416 as finite NameCert data. It sits beside the relation-certificate packet of chapter 1090, the gate packet of chapter 1099, and the relation-audit frame of chapter 1037. The packet names the questions a consumer must answer before same-schema language can be exported; it does not identify the compared domains, transfer empirical content, promote proof obligations across domains, or turn a relation into a bridge.

Definition 7012.1 (Layered relation gate question carrier)**.** *A* $\mathsf{LayeredRelationGateQuestion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, U, V, L, P, N, E, F, S, H, C, R, M)$$

with the following displayed rows:

1. *A and B are the sourceA and sourceB rows naming the two domains being compared.*
2. *U and V are the carrierA and carrierB rows naming the carrier families visible at the compared layers.*
3. *L is the layerMap row pairing source layers, carrier rows, and classifier rows.*
4. *P is the preserved row listing the relation content the question surface permits a consumer to export.*
5. *N is the notPreserved row listing object identity, empirical law, proof transfer, subject-substance, global-frame, hidden bridge, and strength over-reading refusals.*
6. *E is the exactness row recording the available reading of same-schema language.*
7. *F is the failureBoundary row recording when the relation weakens, fails, or must be refused.*
8. *S is the strength row recording whether the packet is seed, paper certificate, checked certificate, or bridge certificate strength.*
9. *H records componentwise* hsame *transport for source, carrier, layer-map, preserved, not-preserved, exactness, failure, strength, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the question route from source exposure through layer mapping, preservation, refusal, exactness, failure boundary, and strength.*

11. *R is the* Pkg *provenance over* LayeredRelationGateQuestion$^{\uparrow}$, LayeredRelationCert$^{\uparrow}$, LayeredRelationGate$^{\uparrow}$, RelationalFrameAudit$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *M is the local* NameCert$_{\text{LayeredRelationGateQuestion}^{\uparrow}}$ *row.*

The classifier accepts a packet only when sourceA/sourceB, carrierA/carrierB, layerMap, preserved, notPreserved, exactness, failureBoundary, strength, transport, replay, provenance, and local name are all present before a same-schema consumer reads the answer surface.

Theorem 7012.2 (Layered relation gate question NameCert obligations)**.** *For every accepted* LayeredRelationGateQuestion$^{\uparrow}$ *carrier* $Q = (A, B, U, V, L, P, N, E, F, S, H, C, R, M)$*, the local* NameCert$_{\text{LayeredRelationG}}$ *surface consists exactly of sourceA/sourceB exposure, carrierA/carrierB exposure, layerMap display, preserved-row disclosure, notPreserved-row disclosure, exactness display, failureBoundary display, strength display, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer cannot export same-schema language until each of these rows has been displayed.*

Proof. Project the accepted carrier. The coordinates of Definition 7012.1 are exactly the finite rows named in the statement.

The source and carrier questions are A, B, U, V. They bind the compared domains and their carrier families before the layer map L can be consumed.

The relation export questions are P, N, E, F, S. The preserved row P records what may be exported, while N and F record the refusal and failure boundary that prevent identity readings.

Finally H, C, R, M transport, replay, source, and name the same question packet. Since no other coordinate exists, a consumer cannot obtain same-schema language without first displaying the listed rows. □

Theorem 7012.3 (Layered relation gate question refusal surface)**.** *Every accepted* LayeredRelationGateQuestion$^{\uparrow}$ *packet refuses identity-misreadings of cross-domain analogy. If a consumer asks to identify sourceA with sourceB, to transfer empirical content, to transfer proof obligations, to import subject-substance, to assert a global frame, or to claim bridge strength without a bridge row, the read is routed to the notPreserved row N and the failureBoundary row F rather than to the preserved row P.*

Proof. Begin with the NameCert obligations of Definition 7012.2. They require P, N, E, F, S to be displayed together before any same-schema answer is exported.

The preserved row P contains only the relation content allowed by the displayed layer map and exactness row. The notPreserved row N lists identity, empirical transfer, proof transfer, subject-substance, global-frame, hidden bridge, and strength over-reading as refused content.

The failureBoundary row F records the action for those refused reads. Therefore any request for an identity-misreading is routed through N and F, while the exported same-schema answer remains confined to P with its exactness and strength rows. □

Falsifiable prediction: Every accepted LayeredRelationGateQuestion$^{\uparrow}$ answer displays sourceA/sourceB, carrierA/carrierB, layerMap, preserved, notPreserved, exactness, failureBoundary, strength, transport, replay, provenance, and local naming rows before it is consumed as a same-schema relation question.

Independence witness: layeredrelationcert, layeredrelationgate, relationalframeaudit: none is bijective to LayeredRelationGateQuestion$^{\uparrow}$ because this packet isolates the finite question surface with source, carrier, layer-map, preserved, not-preserved, exactness, failure-boundary, strength, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationGateQuestion}^{\uparrow}}$.

Constructive story: A $\mathsf{LayeredRelationGateQuestion}^{\uparrow}$ packet is generated from two source rows, two carrier-family rows, a layer map, preserved and not-preserved rows, exactness and failure-boundary rows, a strength row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LayeredRelationGateQuestion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7012.1, Definition 7012.2, and Definition 7012.3 seed the finite question surface for relation-before-identity gate reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LayeredRelationGateQuestion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close object identity between compared domains, empirical transfer, proof transfer, subject-substance content, global-frame structure, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and carrier exposure, layer-map display, preserved and not-preserved disclosure, exactness, failure-boundary, strength, transport, replay, provenance, and local naming for BEDC.Derived.LayeredRelationGateQuestionUp.

Chapter 7013

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ packet records finite refinement of regular-Cauchy observation windows. It sits next to the cofinal-window route of chapter 4060, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet gives Real and Completion consumers a displayed way to pass from a coarse accepted window schedule to a finer schedule without quotient streams, selected limits, countable choice, or ambient completeness.

Definition 7013.1 (Regular Cauchy window refinement carrier)**.** *A* $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, W', D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular source row whose admissible names are read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *W is the original finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposed to the consumer.*
3. *W' is the refined finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule, displayed as a later or thinner observation window over the same source route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the original and refined windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after both schedules and the dyadic tolerance ledger have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read from the refined regular-Cauchy route.*
7. *H records componentwise* hsame *transport for source, original window, refined window, dyadic tolerance, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, W, W', D \longmapsto R \longmapsto E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$, $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowRefinement}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, W', D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative, countable-choice extraction, host real equality, or ambient completeness.

Theorem 7013.2 (Regular Cauchy window refinement NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ *carrier $F = (S, W, W', D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowRefinement}^{\uparrow}}$ *surface consists exactly of source admission through S, original finite-window exposure through W, refined finite-window exposure through W', dyadic tolerance comparison through D, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7013.1. Its displayed rows are exactly $S, W, W', D, R, E, H, C, P, N$, so the original and refined window schedules are both visible before any Real-facing read is exported.

The dyadic ledger D is read after the two schedules have been displayed. It is therefore a comparison between finite window rows, not a quotient comparison between external Cauchy streams.

The readback row R and seal row E are downstream of the same finite data. Transport, replay, provenance, and local naming preserve that route, so the NameCert obligations are exhausted by the listed coordinates. □

Theorem 7013.3 (Regular Cauchy window refinement cofinal handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ *packet factors its read through the finite handoff*

$$\mathsf{RegularCauchyWindowCofinality}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may replace the original finite window W by the refined finite window W' only through the displayed dyadic tolerance ledger D and the regular rational readback row R.

Proof. Start with the source row S and the original schedule W. The cofinal-window route of chapter 4060 supplies the admissible tail boundary for replacing a coarse finite observation by a finer one.

Use finite-window induction along the displayed refinement from W to W'. At the base window the consumer reads the same $\mathsf{StreamName}^{\uparrow}$ schedule, and at each successor window the ledger D records the dyadic tolerance comparison needed to preserve the regular-Cauchy readback.

After the refined schedule has been reached, row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback and row E supplies the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N replay and name the same route, so no selected limit, countable-choice selector, quotient stream equality, or ambient completeness theorem can enter the handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ read displays one source row, one original $\mathsf{StreamName}^{\uparrow}$ window row, one refined $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal, and local transport, replay, provenance, and naming rows before it is consumed by a Real or Completion route.

Independence witness: regular-cauchy-window-cofinality, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ is not bijective to any listed sibling because it binds

the original and refined finite windows, dyadic comparison, regular readback, Real seal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted refinement packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowRefinement}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RegularCauchyWindowRefinement}^{\uparrow}$ packet is generated from a regular source row, original and refined finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7013.1, Definition 7013.2, and Definition 7013.3 seed the finite refinement route from cofinal windows to Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, original and refined window exposure, dyadic tolerance comparison, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindo

Chapter 7014

A Concrete Naming Certificate for $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$ packet records the Heine–Cantor finite-net route from compact metric data to a tolerance-only uniform-modulus handoff. It realizes the compact-uniform-continuity construction of chapter 9237 by exposing finite ProbeBundle nets, dyadic mesh refinements, compact metric source rows, rational radius ledgers, continuous-map pointwise modulus rows, and the terminal $\mathsf{UniformModulus}^{\uparrow}$ consumer surface. It sits with the compact metric certificate of chapter 129, the finite epsilon-net carrier of chapter 1767, the dyadic tolerance ledger of chapter 291, and the uniform-modulus packet of chapter 369.

Definition 7014.1 (Finite-net mesh-refinement carrier)**.** *A* $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, B, G, D, Q, L, A, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, displaying the compact metric carrier and its finite-cover witness surface.*
2. *B is the finite* ProbeBundle *net row, displaying the centers used for the current tolerance request.*
3. *G is the dyadic mesh-refinement row, replacing a coarse net by a finer net through named dyadic radius data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the mesh comparison.*
5. *Q is the rational radius ledger, recording the finite lower-bound fold over the bundled radii.*
6. *L is the continuous-map pointwise modulus row attached to the bundled centers.*
7. *A is the compact-source admission row connecting the finite net to the source points it covers.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced after the mesh, radius, and pointwise-modulus rows have been displayed.*
9. *H records componentwise* hsame *transport for compact source, finite net, mesh refinement, dyadic tolerance, rational radius fold, pointwise modulus, compact admission, uniform-modulus handoff, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$K, B, G, D, Q, L, A \longmapsto U \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteNetMeshRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, G, D, Q, L, A, U, H, C, P, N$. It has no coordinate for an external finite-subcover selector, a selected minimum radius, an ambient compactness oracle, or a global modulus function outside the displayed history rows.

Theorem 7014.2 (Finite-net mesh-refinement NameCert obligations)**.** *For every accepted* FiniteNetMeshRefinemen *carrier $M = (K, B, G, D, Q, L, A, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteNetMeshRefinement}^{\uparrow}}$ *surface consists exactly of compact metric source admission through K, finite ProbeBundle net exposure through B, dyadic mesh refinement through G, dyadic tolerance comparison through D, rational radius folding through Q, pointwise continuous-map modulus reading through L, compact-source admission through A, uniform-modulus handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7014.1. The displayed rows are exactly $K, B, G, D, Q, L, A, U, H, C, P, N$, so the compact source, finite net, mesh, radius, and pointwise-modulus data are visible before the uniform-modulus handoff can be consumed.

The mesh row G is finite because it is attached to the ProbeBundle row B and compared by the dyadic ledger D. The rational fold row Q is likewise internal to the displayed bundled radii, so no selected external minimum is introduced.

The handoff row U is downstream of the compact admission row A and the pointwise modulus row L. Transport, replay, provenance, and local naming preserve this dependency order, which exhausts the local NameCert surface. □

Theorem 7014.3 (Finite-net mesh-refinement uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$ *packet factors its read through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The mesh-refinement packet may refine a finite net only by displaying a dyadic mesh row, the rational lower-bound ledger over bundled radii, and the pointwise modulus rows consumed by the final $\mathsf{UniformModulus}^{\uparrow}$ *handoff.*

Proof. Start with the compact source row K and the finite ProbeBundle net B. These rows are the concrete finite-net surface singled out by chapter 9237, and the consumer may inspect only the centers and coverage rows they display.

Use finite induction over the mesh-refinement row G. The base case is the original finite net; the successor step replaces a bundled mesh cell by displayed dyadic subcells whose tolerance comparison is recorded in D.

Fold the resulting radius data through Q. Because the fold is over the displayed ProbeBundle rows, it produces a history-level lower-bound ledger rather than an external radius selector.

Finally read the pointwise modulus row L at the bundled centers and pass through compact admission A to the uniform-modulus handoff U. The structural rows H, C, P, N replay and name the same finite route, so the packet exports no ambient compactness oracle, external finite-subcover choice, or global modulus function. □

Falsifiable prediction: Every accepted $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$ read displays a CompactMetric source row, a finite ProbeBundle net row, a dyadic mesh-refinement row, a DyadicRatCore tolerance ledger, a rational radius fold, pointwise continuous-map modulus rows, a UniformModulus handoff, and local transport, replay, provenance, and naming rows before compact-uniform-continuity consumers inspect it.

Independence witness: compactmetric, finiteepsilonnet, dyadicratcore, uniformmodulus: FiniteNetMeshRefinem is not bijective to any listed sibling because it binds finite net exposure, dyadic mesh refinement, rational radius folding, pointwise modulus reading, compact-source admission, uniform-modulus handoff, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetMeshRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteNetMeshRefinement}^{\uparrow}$ packet is generated from CompactMetric source data, finite ProbeBundle net rows, dyadic mesh-refinement rows, DyadicRatCore tolerance ledgers, rational radius folds, pointwise continuous-map modulus rows, compact-source admission, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetMeshRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7014.1, Definition 7014.2, and Definition 7014.3 seed the finite mesh route from compact nets to tolerance-only uniform-modulus data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetMeshRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an external finite-subcover selector, a selected minimum radius, an ambient compactness oracle, a global modulus function, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, finite ProbeBundle net exposure, dyadic mesh refinement, radius folding, pointwise modulus reading, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteNetMeshRefinementUp.

Chapter 7015

A Concrete Naming Certificate for $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$

Following Bishop–Bridges constructive completion theory, the $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ packet records the finite route by which a metric-space point stream is read as regular-Cauchy completion data and then exposed to Real-facing consumers. It sits after the metric carrier of chapter 57, the completeness boundary of chapter 128, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a completion-facing regular-Cauchy handoff; it does not quotient Cauchy filters, choose limit representatives, assert ambient completeness, or import host real equality.

Definition 7015.1 (Metric regular Cauchy completion carrier)**.** *A* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point row of the named metric-space source.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *metric-structure row, exposing only the displayed distance and comparison data available to the packet.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the inspected source windows of the Cauchy name.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row by which the inspected windows are consumed as regular rational approximation data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for metric-window comparison and regular-Cauchy budget replay.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for point rows, metric rows, stream windows, regular readback, dyadic tolerances, Real sealing, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion precision request through*

$$X, M, S, D \longmapsto R \longmapsto E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricRegularCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, S, R, D, E, H, C, P, N$. It has no coordinate for quotient completion equality, a selected completion point, a selected limit representative, countable choice, host real equality, or an ambient completeness proof beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *boundary reference.*

Theorem 7015.2 (Metric regular Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ *carrier*

$$K = (X, M, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricRegularCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for Definition 7015.1; metric-window exactness through X, M, S, D; regular-Cauchy readback through R; dyadic tolerance replay through D; Real-seal nonescape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N. A completion-facing read may cite* $\mathsf{CompleteMetric}^{\uparrow}$ *as the sibling boundary named in provenance, but the packet itself exposes no hidden completeness theorem or quotient completion object.*

Proof. Project the accepted carrier. Its visible coordinates are exactly $X, M, S, R, D, E, H, C, P, N$, so carrier admission and local naming are finite row projections.

The metric-window side is X, M, S, D. The point and metric rows determine the displayed distance comparisons, the stream row supplies the inspected windows, and the dyadic ledger fixes the tolerance budget used by those comparisons.

The regular-Cauchy and Real-facing reads are downstream of those rows. The row R consumes the visible windows as regular rational data, and the seal row E can be reached only along the displayed route through M, S, R, D.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no additional coordinate for quotient equality, selected limits, choice, host real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7015.3 (Metric regular Cauchy completion Real-seal boundary)**.** *Every completion-facing read of an accepted* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the displayed route*

$$\mathsf{Met}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad X, M, S, R, D, E, H, C, P, N.$$

Thus the Real seal is a visible boundary of the metric-window, stream-window, regular-readback, and dyadic-tolerance rows; it is not a quotient completion, selected representative, countable-choice schedule, host real primitive, or ambient completeness assertion.

Proof. Begin with a read that asks for the Real-facing seal. The carrier gives no direct Real coordinate before the point row, metric row, inspected stream windows, regular readback, and dyadic tolerance ledger have been exposed.

Read the displayed route. The rows X, M, S, D provide the metric-window comparison surface, and R records the regular-Cauchy readback attached to those same windows and tolerances.

Only then may E be consumed as the $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N preserve, replay, source, and name the same route, while P records $\mathsf{CompleteMetric}^{\uparrow}$ only as a sibling boundary for completion consumers.

Because no further carrier coordinate exists, the read cannot escape into quotient completion equality, a selected representative, a countable-choice schedule, host real primitives, or an ambient completeness theorem. □

Theorem 7015.4 (Metric regular Cauchy completion complete-metric handoff)**.** *Every accepted* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ *packet may cite* $\mathsf{CompleteMetric}^{\uparrow}$ *only through the displayed metric row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The complete-metric sibling is therefore a boundary row for completion consumers, not an ambient completeness theorem available inside the packet.*

Proof. Project the carrier of Definition 7015.1. The visible rows are $X, M, S, R, D, E, H, C, P, N$, and the only place $\mathsf{CompleteMetric}^{\uparrow}$ appears is the provenance boundary attached to the completion-facing route.

Use the NameCert obligations of Definition 7015.2. They force the metric-window, regular-readback, dyadic-tolerance, Real-seal, transport, replay, provenance, and local naming rows to be displayed before any completion-facing read is accepted.

Finally apply the Real-seal boundary of Definition 7015.3. Since the seal is reached only through the displayed finite route, the complete-metric citation remains a sibling handoff row and cannot supply quotient completion equality, a selected limit, countable choice, or ambient completeness. □

Theorem 7015.5 (Metric regular Cauchy completion window universality)**.** *Let two completion-facing reads of accepted* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ *packets have the same displayed metric-window rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. Then they determine the same completion-facing surface for every consumer that reads through* $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$.

Proof. Read both packets through Definition 7015.1. The classifier of an accepted packet has no coordinate beyond the displayed metric, stream, readback, dyadic, Real-seal, transport, replay, provenance, and local naming rows.

Apply Definition 7015.2. Equal displayed rows give the same carrier admission, metric-window exactness, regular-Cauchy readback, dyadic tolerance replay, Real-seal nonescape, transport, replay, provenance, and local naming obligations.

The Real-seal boundary theorem then forces every completion-facing consumer to pass through those same rows before reaching $\mathsf{Real}^{\uparrow}$. Hence no quotient stream equality, selected limit, countable-choice schedule, or ambient completeness coordinate remains that could distinguish the two completion-facing surfaces. □

Falsifiable prediction: Every accepted $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ completion-facing read exposes a $\mathsf{Met}^{\uparrow}$ metric row, $\mathsf{StreamName}^{\uparrow}$ finite windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the displayed packet rows before it consumes the terminal $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: metricspace, completemetric, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole packet $K = (X, M, S, R, D, E, H, C, P, N)$, because each sibling contributes only a metric row, a completeness boundary, finite windows, regular rational

readback, dyadic tolerances, or a Real seal while this chapter binds the completion route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricRegularCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricRegularCauchyCompletion}^{\uparrow}$ packet is generated from point rows, MetricSpace metric rows, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricRegularCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7015.1, Definition 7015.2, and Definition 7015.3 seed the metric regular-Cauchy completion packet over metric windows, finite streams, regular rational readback, dyadic tolerances, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricRegularCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected completion points, countable choice, host real equality, Lean verification, audit cleanliness, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, metric-window exactness, regular-Cauchy readback, dyadic tolerance replay, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricRegularCauchyCor

Chapter 7016

A Concrete Naming Certificate for $\mathsf{RegularSequenceEquivalence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularSequenceEquivalence}^{\uparrow}$ packet records the finite comparison surface between two regular sequence names before their terminal real seals are read. It sits between the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the Cauchy route of chapter 3413. The certificate names a non-quotient equivalence boundary for regular sequence comparison; it does not select a completed real, identify quotient streams, or import ambient completeness.

Definition 7016.1 (RegularSequenceEquivalence NameCert surface)**.** *A* $\mathsf{RegularSequenceEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S_0, S_1, W_0, W_1, D, R_0, R_1, Q, A, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *schedules whose finite windows are compared;*
2. W_0 *and* W_1 *are the paired finite window lists selected from those schedules;*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for all paired window reads;*
4. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from* S_0, W_0, D *and* S_1, W_1, D*;*
5. Q *is the finite symmetric tail classifier, recording that each displayed tail request in either direction is met inside the same dyadic tolerance ledger;*
6. A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-comparison row, readable only after* $S_0, S_1, W_0, W_1, D, R_0, R_1, Q$ *have been displayed;*
7. H *is the componentwise* hsame *provenance for the stream schedules, window lists, dyadic ledger, readbacks, classifier, and seal row;*
8. C *is the displayed* Cont *replay route for the same finite rows;*
9. P *is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceEquivalence}^{\uparrow}}$ *row.*

The packet exposes no quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient completeness coordinate.

Theorem 7016.2 (Symmetric tail obligations for regular sequence equivalence)**.** *For every accepted* $\mathsf{RegularSequenceEquivalence}^{\uparrow}$ *packet* $E = (S_0, S_1, W_0, W_1, D, R_0, R_1, Q, A, H, C, P, N)$*, the finite tail comparison is symmetric over the same displayed ledger. A consumer reading the comparison from* R_0 *to* R_1 *and a consumer reading it from* R_1 *to* R_0 *must both pass through the same* $\mathsf{StreamName}^{\uparrow}$ *windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the same pair of* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row. Thus the comparison is a finite NameCert surface for regular sequence equivalence, not a quotient equality or a selected completed-real witness.*

Proof. Project the carrier of Definition 7016.1. The two directions of the tail comparison are not stored as separate equivalence principles; they are the two readings of the single classifier row Q over W_0, W_1, D.

For the forward reading, the consumer first observes S_0 through W_0, reads R_0 through the dyadic ledger D, and compares it with the S_1, W_1, R_1 side by the displayed classifier Q. The terminal seal A is available only after those rows have been displayed.

For the reverse reading, the same classifier row is read with the two window sides exchanged. Because D, H, C, P, and N are componentwise rows over both sides at once, the reverse read has no independent tolerance ledger or hidden equality source.

The carrier contains no quotient stream, selected representative, host real equality, countable-choice selector, or ambient completeness coordinate. Therefore both directions of regular sequence equivalence are exhausted by the same finite StreamName, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and NameCert rows. □

Theorem 7016.3 (Regular sequence equivalence Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing comparison exported by an accepted* $\mathsf{RegularSequenceEquivalence}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularSequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff may be cited by Cauchy and completion consumers only as the displayed comparison surface over those rows; it does not provide quotient equality of streams, a selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the displayed rows of Definition 7016.1 in order. The stream schedules and finite windows determine the observation domain, the dyadic ledger fixes the tolerance tests, and the two RegSeqRat rows are the only regular rational readbacks available to the classifier.

Apply Definition 7016.2. It identifies the two directional tail reads with the same finite classifier row over the same ledger and the same terminal Real seal.

Since A is attached after the two readbacks and the classifier, every Real-facing comparison must pass through the displayed sequence of rows. The remaining H, C, P, N coordinates transport, replay, package, and name this route, while the excluded quotient, selection, host-equality, choice, and completeness data never appear in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceEquivalence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularSequenceEquivalence}^{\uparrow}$ packet is generated from paired StreamName schedules, paired finite window lists, a shared DyadicRatCore tolerance ledger, two RegSeqRat readback rows, a symmetric finite tail classifier, a terminal Real seal-comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7016.1, Definition 7016.2, and Definition 7016.3 bind the finite regular sequence comparison surface over StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Cauchy$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularSequenceEquivalence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient Real$^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, symmetric tail-classifier stability, RegSeqRat readback exactness, Real seal nonescape, and local NameCert coverage for BEDC.Derived.RegularSequenceEquivalenceUp.

Chapter 7017

A Concrete Naming Certificate for $\mathsf{GaussQuadrature}^{\uparrow}$

Following Gauss, Christoffel, and the classical numerical-analysis tradition of Gaussian quadrature, the $\mathsf{GaussQuadrature}^{\uparrow}$ packet records a finite node-weight rule as BEDC NameCert data rather than as a host integration theorem. It sits over the coefficient and degree surface of chapter 50, the orthogonality ledger of chapter 2264, the tagged integral consumer route of chapter 1526, the real-name seal of chapter 38, and the dyadic tolerance arithmetic of chapter 291. The packet gives downstream numerical-integration chapters a finite quadrature-rule certificate without importing measure-theoretic integration, spectral completeness, quotient polynomial equality, selected real roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7017.1 (Gauss quadrature carrier)**.** *A* $\mathsf{GaussQuadrature}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (P, O, N, W, E, R, D, H, C, K)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, carrying the finite coefficient ledger, degree bound, and polynomial evaluation interface inspected by the rule.*
2. *O is the* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *row, displaying the finite orthogonal-family window whose root and recurrence data determine the quadrature nodes.*
3. *N is the node ledger, listing the finitely many displayed real-name nodes and their attachment to the orthogonal-polynomial window.*
4. *W is the weight ledger, listing the displayed real-name weights and the finite positivity or non-degeneracy checks required by the rule.*
5. *E is the exactness ledger, displaying the polynomial-degree range for which the weighted node sum and the* $\mathsf{RiemannIntegral}^{\uparrow}$ *read agree.*
6. *R is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *handoff row, exposing only the tagged integration route that may consume the exactness ledger.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic row for finite comparisons of node positions, weights, and exactness residuals.*
8. *H records componentwise* hsame *transport for P, O, N, W, E, R, D and the local naming data.*

9. *C records displayed* Cont *replay from the polynomial row through the orthogonal-polynomial row, node and weight ledgers, exactness ledger, integration handoff, and dyadic comparison row.*
10. *K is the combined* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{GaussQuadrature}^\uparrow}$ *row over* $\mathsf{GaussQuadrature}^\uparrow$, $\mathsf{Poly}^\uparrow$, $\mathsf{OrthogonalPolynomial}^\uparrow$, $\mathsf{RiemannIntegral}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert.

The classifier compares accepted packets only through $P, O, N, W, E, R, D, H, C, K$. It has no coordinate for a host measure space, selected real-root solver, spectral completeness theorem, quotient polynomial equality, analytic error estimate outside the displayed degree range, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 7017.2 (Gauss quadrature NameCert obligations)**.** *For every accepted* $\mathsf{GaussQuadrature}^\uparrow$ *carrier*

$$Q = (P, O, N, W, E, R, D, H, C, K),$$

the local $\mathsf{NameCert}_{\mathsf{GaussQuadrature}^\uparrow}$ *surface consists exactly of polynomial-source admission, orthogonal-polynomial window admission, finite node ledger admission, finite weight ledger admission, degree-exactness exposure,* $\mathsf{RiemannIntegral}^\uparrow$ *handoff, dyadic comparison replay, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host measure spaces, selected root solvers, spectral completeness, quotient polynomial equality, undisplayed error estimates, and ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 7017.1. The only source rows are P and O, so the quadrature rule starts from a displayed polynomial ledger and a displayed orthogonal-polynomial window.

The finite quadrature data are N and W. These rows list the nodes and weights as real-name entries tied to the same orthogonal-polynomial window, with D supplying the dyadic comparisons used to inspect them.

The exactness row E is then the only route to integration-facing content. It compares the weighted node sum against the $\mathsf{RiemannIntegral}^\uparrow$ handoff row R over the displayed polynomial-degree range.

The structural rows H, C, K transport, replay, source, and name this same finite route. Since no carrier coordinate stores a host measure space, a root-selection oracle, spectral completeness, quotient polynomial equality, undisplayed error estimates, or ambient real completeness, the listed rows exhaust the local certificate surface. □

Theorem 7017.3 (Gauss quadrature node-weight ledger)**.** *Every* $\mathsf{RiemannIntegral}^\uparrow$*-facing consumer of an accepted* $\mathsf{GaussQuadrature}^\uparrow$ *packet must read the finite route*

$$\mathsf{Poly}^\uparrow \longmapsto \mathsf{OrthogonalPolynomial}^\uparrow \longmapsto (N, W) \longmapsto E \longmapsto \mathsf{RiemannIntegral}^\uparrow$$

through the displayed rows $P, O, N, W, E, R, D, H, C, K$. The consumer may use the quadrature rule only after the node ledger, weight ledger, degree-exactness ledger, and dyadic comparison row have been displayed, and it receives no host measure, selected root solver, spectral completeness theorem, quotient polynomial equality, analytic error estimate outside the displayed degree range, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Begin with an integration-facing read. The carrier first exposes P, the finite polynomial source row, and then O, the finite orthogonal-polynomial window that controls the rule.

The node and weight information is available only as N and W. The dyadic row D supplies the finite comparisons for these displayed entries; it does not select roots or appeal to a complete real-root solver.

The exactness ledger E is downstream of P, O, N, W, D. Only after this ledger is present may the handoff row R be read by a $\mathsf{RiemannIntegral}^{\uparrow}$ consumer.

Transport and replay preserve the same row order through H and C, and K names the route as a local certificate. Thus the public quadrature read is a finite node-weight ledger, not a host integration or spectral-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaussQuadrature}^{\uparrow}}$.

Constructive story: A $\mathsf{GaussQuadrature}^{\uparrow}$ packet is generated from a PolynomialUp source row, an OrthogonalPolynomialUp finite window, displayed real-name node and weight ledgers, a degree-exactness ledger, a RiemannIntegralUp handoff row, DyadicRatCoreUp comparison data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaussQuadrature}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7017.1, Definition 7017.2, and Definition 7017.3 seed the finite quadrature-rule certificate over $\mathsf{Poly}^{\uparrow}$, $\mathsf{OrthogonalPolynomial}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and BHist.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaussQuadrature}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host measure spaces, selected root solvers, spectral completeness, quotient polynomial equality, analytic error estimates outside the displayed exactness range, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial-source admission, orthogonal-polynomial window admission, node and weight ledger admission, degree-exactness exposure, RiemannIntegralUp handoff, dyadic comparison replay, transport, non-escape, and field faithfulness for BEDC.Derived.GaussQuadratureUp.

Chapter 7018

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$ packet records the finite interface where regular Cauchy rational data can be read through located lower-and-upper cut evidence before a terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits after the Cauchy-cut bridge of chapter 2312, the regular rational source of chapter 301, the stream-window source of chapter 126, the dyadic arithmetic core of chapter 291, and the real-name seal of chapter 38. The packet gives Real and Completion consumers a cut-facing boundary for regular Cauchy data without importing Dedekind completeness, quotient equality, selected limits, trichotomy, countable choice, or bridge checking.

Definition 7018.1 (Regular Cauchy located cut carrier)**.** *A* $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observed prefixes of the rational source.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational row read from the same windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger used to compare finite rational cells.*
4. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *lower-and-upper cut row, carrying locatedness evidence for the displayed dyadic gap requests.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, Q, D, L have been displayed.*
6. *H records componentwise* hsame *transport for the stream, regular rational, dyadic, located-cut, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite cut request through S, Q, D, L before the real seal R is consumed.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCut}^{\uparrow}}$ *row naming exactly this finite cut-facing packet.*

The classifier compares accepted packets only through $S, Q, D, L, R, H, C, P, N$. It has no coordinate for quotient real equality, Dedekind completeness, host-selected limits, trichotomy, countable choice, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCutUp.RegularCauchyLocatedCutTasteGate single carrier alignment`

Theorem 7018.2 (Regular Cauchy located cut NameCert obligations)**.** *For every accepted* RegularCauchyLocatedC *carrier* $K = (S, Q, D, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream windows, regular rational readback, dyadic tolerance ledger, located lower-and-upper cut row, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. The obligation surface may cite the Cauchy-cut packet of chapter 2312, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38; it cannot add a quotient representative, selected limit, trichotomy branch, countable-choice schedule, Dedekind-completeness witness, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCutUp.RegularCauchyLocatedCutCarrier namecert obligations`

Proof. Project the carrier of Definition 7018.1. The analytic rows are exactly the finite stream window S, regular rational source Q, dyadic tolerance ledger D, located cut row L, and real seal R.

The ordered route forces each cut request to pass from S to Q, then through the dyadic gap data D, before L exposes locatedness evidence. The seal row R is downstream of that finite route.

Finally, H, C, P, N transport, replay, source, and name only the same packet. Since the excluded objects are absent from the carrier inventory, they cannot appear as additional NameCert obligations. □

Theorem 7018.3 (Regular Cauchy located cut Real-seal handoff)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$ *carrier reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedCut}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported read is a located cut-facing regular Cauchy boundary, not a quotient real name, not a selected limit, not a trichotomy decision, not countable choice, not Dedekind completeness, and not a checked bridge.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCutUp.RegularCauchyLocatedCutRealSealHandoff`

Proof. Begin with any consumer read of the seal R. By Definition 7018.1, the seal is packaged with the displayed windows S, regular rational row Q, dyadic ledger D, and located cut row L.

Apply Definition 7018.2. It keeps the Cauchy-facing rows and located cut row visible before the seal is consumed, and it admits only the structural transport, replay, provenance, and naming rows as further coordinates.

The forbidden principles are not fields of the carrier. Thus no path through the local certificate can export quotient equality of real names, a selected limit, trichotomy, countable choice, Dedekind completeness, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCut}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$ packet is generated from finite StreamName windows, a RegSeqRat regular rational source, a DyadicRatCore tolerance ledger, a LocatedCut lower-and-upper row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCut}^{\uparrow}}$, seed)

Scope closed: Definition 7018.1, Definition 7018.2, and Definition 7018.3 seed the finite located-cut boundary for regular Cauchy real-name reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim $\mathrm{Real}^{\uparrow}$ completeness, quotient equality, selected limits, trichotomy, countable choice, Dedekind completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window locality, regular rational read-back, dyadic gap exactness, located cut exposure, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedCutUp.

Chapter 7019

A Concrete Naming Certificate for HyperbolicBoundaryCompactification$^{\uparrow}$

Following Gromov and Bridson–Haefliger on hyperbolic boundary compactifications, the HyperbolicBoundaryCompa packet records the finite route by which a hyperbolic metric source, visual boundary, Gromov boundary, visual metric, Busemann boundary, boundary shadow, and terminal real seal are read as one compactification-facing NameCert surface. It realizes the boundary-compactification branch of chapter 9463 after the Gromov boundary packet of chapter 6470, the visual-metric packet of chapter 3866, the Busemann-boundary packet of chapter 3732, the boundary-shadow route of chapter 3813, the visual-boundary route of chapter 4181, and the real-name seal of chapter 38. The packet distinguishes a finite compactification boundary route from a completed quotient boundary, selected geodesic representative, host compact space, or ambient completeness theorem.

Definition 7019.1 (Hyperbolic boundary compactification carrier)**.** *A* HyperbolicBoundaryCompactification$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, V, G, D, U, S, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the hyperbolic metric source row that records the finite distance comparisons being compactified.*
2. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *visual-boundary row.*
3. *G is the* GromovBoundary$^{\uparrow}$ *boundary-equivalence row.*
4. *D is the* HyperbolicVisualMetric$^{\uparrow}$ *visual-metric comparison row on displayed boundary data.*
5. *U is the* HyperbolicBusemannBoundary$^{\uparrow}$ *Busemann-level boundary row.*
6. *S is the* HyperbolicBoundaryShadow$^{\uparrow}$ *shadow row that records finite boundary neighbourhood evidence.*
7. *R is the terminal* Real$^{\uparrow}$ *seal for scalar radii, visual distances, or shadow thresholds read by the packet.*
8. *H records componentwise* hsame *transport for metric, visual-boundary, Gromov-boundary, visual-metric, Busemann, shadow, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same ordered boundary-compactification route.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, V, G, D, U, S, R, H, C, P, N$. It has no coordinate for a completed quotient boundary, selected ray representative, compact host space, physical observer law, countable choice, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryCompactificationUp.HyperbolicBoundaryCompactificationTasteGate`
`single carrier alignment`

Theorem 7019.2 (Hyperbolic boundary compactification NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}$ *carrier $B = (M, V, G, D, U, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}}$ *surface consists exactly of carrier admission for the hyperbolic metric source, visual-boundary row, Gromov-boundary row, visual-metric comparison, Busemann boundary row, boundary-shadow row, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. It may route through the sibling surfaces of chapter 6470, chapter 3866, chapter 3732, and chapter 3813; it cannot add a host compactification, selected boundary representative, quotient ideal boundary, global hyperbolic model, countable-choice schedule, or ambient completeness witness.*

Proof. Project the carrier of Definition 7019.1. The boundary-facing rows are exactly the metric source M, visual boundary V, Gromov boundary G, visual metric D, Busemann boundary U, boundary shadow S, and real seal R.

The route reads visual boundary and Gromov boundary data before the visual metric, Busemann, and shadow consumers receive the boundary token. This keeps compactification-facing evidence finite and displayed.

Finally, H, C, P, N transport, replay, source, and name the same carrier. Since the excluded host objects are not rows of the packet, the listed obligations exhaust the local NameCert surface. □

Theorem 7019.3 (Hyperbolic boundary compactification visual handoff)**.** *Every visual-metric, Busemann, boundary-shadow, or Real-seal read from an accepted* $\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{HyperbolicVisualBoundary}^{\uparrow} \longmapsto \mathsf{GromovBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicBusemannBoundary}^{\uparrow}$$

The exported object is a compactification-facing boundary route, not a completed quotient compactum, selected geodesic representative, global hyperbolic space, physical observer expansion law, countable choice, or ambient completeness theorem.

Proof. Start with a consumer read of D, U, S, or R. By Definition 7019.1, those rows are packaged downstream of the visual-boundary and Gromov-boundary rows V, G and remain tied to the hyperbolic metric source M.

Apply Definition 7019.2. It keeps the visual-metric, Busemann, boundary-shadow, and real-seal reads on the same displayed boundary route, and it permits only H, C, P, N as structural continuation rows.

The packet therefore exports finite compactification-facing evidence rather than any host quotient or completed compact space. The refused principles are not fields of the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}$ packet is generated from a hyperbolic metric source, HyperbolicVisualBoundary rows, GromovBoundary rows, HyperbolicVisualMetric rows, HyperbolicBusemannBoundary rows, HyperbolicBoundaryShadow rows, a RealUp scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7019.1, Definition 7019.2, and Definition 7019.3 seed the finite visual-boundary, Gromov-boundary, visual-metric, Busemann, shadow, and Real-seal compactification route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed quotient boundary, selected geodesic representative, host compact space, global hyperbolic model, physical observer law, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-boundary admission, Gromov-boundary handoff, visual-metric compatibility, Busemann boundary routing, boundary-shadow exposure, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBoundaryCompactificationUp.

Chapter 7020

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$

Following Bishop and Bridges on subsequences of regular Cauchy names, the $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$ packet records the finite route by which a source regular sequence may be inspected along a displayed monotone cofinal index map while preserving the rational readback and Real-facing seal. It sits beside the schedule rows of chapter 126, the regular rational carrier of chapter 301, the terminal real-name surface of chapter 38, the dyadic tolerance ledger of chapter 291, and the extractor route of chapter 5125. The packet names subsequence stability as finite NameCert data; it is not quotient stream equality, not a selected limit, not countable choice, not host real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7020.1 (Regular Cauchy subsequence principle carrier)**.** *A* $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular sequence row, carried through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules.*
2. *I is the monotone cofinal index map, recording the finite request translation from subsequence windows back into source windows and the cofinal tail witness used by every downstream read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget ledger, carrying the displayed precision cells, dyadic slack comparisons, and tail thresholds.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and subsequence observations inspected at each displayed precision request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after S, I, D, W have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only as the seal of the same cofinal regular readback route.*
7. *H records componentwise* hsame *transport for $S, I, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a subsequence precision request through I, the source row S, the dyadic tail budget D, the finite windows W, the regular readback R, and the Real seal E.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, I, D, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected subsequence limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7020.2 (Regular Cauchy subsequence principle NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$ *carrier $Q = (S, I, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission, cofinal index stability, regular tail preservation, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every subsequence read must expose the source regular sequence, monotone cofinal index map, dyadic tail budget, finite StreamName windows, and RegSeqRat readback before the Real seal is consumed.*

Proof. Project the carrier of Definition 7020.1. The analytic rows are S, I, D, W, R, E, and the structural rows H, C, P, N only transport, replay, source, and name those rows.

The cofinal index row I is the sole route from a subsequence precision request back to the source regular sequence. The dyadic budget D and finite window row W are read on that route, so regular tail preservation is tied to the displayed source sequence rather than to a detached subsequence object.

The readback row R and Real seal E are downstream of the same finite route. Since the packet contains no quotient coordinate, selected limit row, choice schedule, host equality row, or ambient completeness theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 7020.3 (Regular Cauchy subsequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$ *carrier factors through the displayed cofinal tail and regular-readback route*

$$I, S, D, W \longmapsto R \longmapsto E.$$

Thus a downstream Real consumer may cite the seal row E only after the monotone cofinal index map, source regular sequence, dyadic tail budget, finite StreamName windows, and RegSeqRat readback have all been displayed. The handoff exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient Real completeness.

Proof. Start with a consumer asking for E. The carrier definition places E after the readback row R, and R is admitted only after the cofinal index, source sequence, dyadic budget, and finite windows have been displayed.

The continuation row C records the same order of exposure. A subsequence request is replayed through I before it reads S, then through D and W before it reaches the regular rational row.

Transport and provenance through H and P, with local naming through N, preserve this finite packet. Therefore every Real-facing read is mediated by the displayed cofinal tail route and receives none of the excluded quotient, choice, equality, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}$ packet is generated from a source regular sequence, a monotone cofinal index map, a DyadicRatCore tail budget, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySubsequencePrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 7020.1, Definition 7020.2, and Definition 7020.3 seed the finite cofinal subsequence route from displayed regular Cauchy data to the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequencePrinciple}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequence limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal index stability, dyadic tail-budget exactness, finite-window replay, regular readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubsequencePrincipleUp.

Chapter 7021

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$ packet records the finite envelope by which compact finite nets, local continuity moduli, rational lower-bound folds, and a tolerance-only uniform modulus row are read as one NameCert carrier. It realizes the compact-uniform-continuity route of chapter 9237 beside the compact metric source of chapter 129, the continuous-map source of chapter 130, the uniform-modulus handoff of chapter 369, the metric distance source of chapter 57, and the rational fold source of chapter 37. The packet exports a finite modulus envelope only; it is not an open-cover compactness theorem, not a selected pointwise-radius function, not host function equality, and not ambient Real completeness.

Definition 7021.1 (Compact uniform modulus envelope carrier)**.** *A* $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, L, F, B, D, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, recording the compact source, displayed net probes, and coverage witnesses.*
2. *L is the local modulus row, read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *carrier at each displayed finite-net center.*
3. *F is the rational lower-bound fold row, using* $\mathsf{Rat}^{\uparrow}$ *comparison data to fold the finitely many local radius ledgers into one positive precision history.*
4. *B is the dyadic tolerance row, carrying the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons used to align finite-net coverage, local radii, and target tolerance.*
5. *D is the* $\mathsf{Met}^{\uparrow}$ *distance-window row, recording the source and target triangle reads needed by the finite-net argument.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row reached only after K, L, F, B, D have been displayed.*
7. *H records componentwise* hsame *transport for $K, L, F, B, D, U, C, P, N$.*
8. *C records displayed* Cont *replay from a tolerance request through the compact finite-net row, local modulus row, rational lower-bound fold, dyadic tolerance row, metric window row, and uniform-modulus handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, L, F, B, D, U, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers, selected pointwise radii, host function equality, quotient real representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7021.2 (Compact uniform modulus envelope NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$ *carrier $M = (K, L, F, B, D, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModul}}$ *surface consists exactly of compact finite-net exposure, local modulus exposure, rational lower-bound fold exactness, dyadic tolerance compatibility, metric distance-window replay,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any uniform-modulus consumer must read those finite rows before the tolerance-only modulus row is exposed.*

Proof. Project the carrier of Definition 7021.1. The compact, local-modulus, rational-fold, dyadic, metric, and uniform-modulus rows are the analytic coordinates, while H, C, P, N only transport, replay, source, and name that route.

The local rows cannot produce a uniform modulus by themselves. They first pass through the finite rational lower-bound fold F, and the dyadic row B aligns the selected lower bound with the tolerance history read by the metric window D.

The handoff row U is downstream of this envelope. Since the carrier has no open-cover theorem, selected subcover, selected pointwise-radius function, host equality row, quotient representative, or ambient completeness theorem, the listed rows exhaust the NameCert obligations. □

Theorem 7021.3 (Compact uniform modulus envelope vision handoff)**.** *Every consumer of the compact-uniform-continuity vision that cites an accepted* $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$ *packet must pass through the finite route*

$$K, L, F, B, D \longmapsto U.$$

Thus the vision-facing read sees the compact finite net, local continuity moduli, rational lower-bound fold, dyadic tolerance row, and metric distance window before it reaches $\mathsf{UniformModulus}^{\uparrow}$*. The handoff exports no open-cover compactness theorem, selected subcover, selected pointwise-radius function, host function equality, quotient real representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a consumer asking for the uniform-modulus row U. The carrier definition places U after the compact finite-net row, local modulus row, rational fold, dyadic tolerance row, and metric window.

The continuation row C records the same order of exposure for each tolerance request. It reads the finite compact net, inspects the local moduli at displayed centers, folds their rational lower bounds, and only then applies the dyadic and metric windows that justify the uniform precision.

Transport through H, provenance through P, and naming through N preserve the same finite envelope. Therefore the vision handoff is a concrete NameCert route and cannot be reread as any of the excluded host compactness, selection, equality, quotient, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusEnvelope}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap local moduli, a RatUp lower-bound fold, DyadicRatCore tolerance compatibility, Metric distance windows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactUniformModulusEnvelope}\uparrow}$, seed)

Scope closed: Definition 7021.1, Definition 7021.2, and Definition 7021.3 seed the finite compact-uniform-modulus envelope route from compact nets and local moduli to a tolerance-only uniform-modulus handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactUniformModulusEnvelope}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected pointwise radii, host function equality, quotient real representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net exposure, local modulus exposure, rational lower-bound fold exactness, dyadic tolerance compatibility, metric distance-window replay, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusEnvelopeUp.

Chapter 7022

A Concrete Naming Certificate for $\mathsf{CauchyNameUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{CauchyNameUniformity}^{\uparrow}$ packet records the finite BEDC surface at which two Cauchy names are compared by common observation windows. It sits after the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the completion witness route of chapter 3579. The packet names the common-window uniformity surface for Cauchy names; it does not quotient streams, choose limits, use countable choice, assert ambient completeness, or identify names by host real equality.

Definition 7022.1 (Cauchy name uniformity carrier)**.** *A* $\mathsf{CauchyNameUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S_0, S_1, W, R_0, R_1, D, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows whose observed prefixes are compared.*
2. W *is the common-window row specifying the shared finite inspection bound used by both stream names.*
3. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those same windows.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for every entourage and precision request.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the common windows, readback rows, and dyadic tolerance row have been displayed.*
6. H *records componentwise* hsame *transport for stream windows, common-window bounds, regular readback, dyadic tolerances, Real sealing, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from a uniformity request through*

$$S_0, S_1, W, D \longmapsto R_0, R_1 \longmapsto E \longmapsto H, P, N.$$

8. P *is the* Pkg *provenance over* $\mathsf{CauchyNameUniformity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyNameUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, W, R_0, R_1, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, a selected completion representative, countable choice, host real equality, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.CauchyNameUniformityUp.CauchyNameUniformityTasteGate single carrier alignment`

Theorem 7022.2 (Cauchy name uniformity NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNameUniformity}^{\uparrow}$ *carrier*

$$U = (S_0, S_1, W, R_0, R_1, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyNameUniformity}^{\uparrow}}$ *surface consists exactly of carrier admission for Definition 7022.1; common-window exposure through S_0, S_1, W; regular rational readback through R_0, R_1; dyadic tolerance exposure through D; Real-seal nonescape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N. Every uniformity read passes through the common* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, dyadic tolerance rows, the* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *rows before any completion-facing consumer may cite* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyNameUniformityUp.CauchyNameUniformityNameCertObligations`

Proof. Project the accepted finite BHist tuple. Its visible coordinates are $S_0, S_1, W, R_0, R_1, D, E, H, C, P, N$, so carrier admission and local naming are finite row projections.

The uniformity read first exposes the common-window surface. The rows S_0, S_1, W give the two inspected stream prefixes and the shared finite bound, while D fixes the dyadic tolerance used by the comparison.

The readback rows R_0, R_1 consume those same windows as regular rational data. Only after the window, readback, and tolerance rows are displayed may the terminal row E be consumed as the $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness, every completion-facing use of $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ remains a provenance-bound consumer of the displayed rows. □

Falsifiable prediction: Every accepted $\mathsf{CauchyNameUniformity}^{\uparrow}$ uniformity read exposes two $\mathsf{StreamName}^{\uparrow}$ finite-window rows, one common-window bound, two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the terminal $\mathsf{Real}^{\uparrow}$ seal before it may be consumed by a $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ route.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-completeness-witness: no listed sibling supplies the whole packet $U = (S_0, S_1, W, R_0, R_1, D, E, H, C, P, N)$, because each sibling contributes only finite windows, regular rational readback, dyadic tolerances, a Real seal, or a completion witness while this chapter binds the common-window uniformity route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNameUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNameUniformity}^{\uparrow}$ packet is generated from two StreamName finite-window rows, a common-window bound, two RegSeqRat readback rows, one DyadicRatCore tolerance row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyNameUniformity}^{\uparrow}}$, seed)

Scope closed: Definition 7022.1 and Definition 7022.2 seed the common-window uniformity surface for Cauchy names over finite streams, regular rational readback, dyadic tolerances, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{CauchyNameUniformity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, selected completion representatives, countable choice, host real equality, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-window exactness, regular readback, dyadic tolerance replay, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNameUniformityUp.

Chapter 7023

A Concrete Naming Certificate for $\mathsf{CofiniteFilter}^{\uparrow}$

Following Bourbaki and Kelley on filters, the $\mathsf{CofiniteFilter}^{\uparrow}$ packet records the tail-facing filter on a displayed stream by finite complement ledgers rather than by a host powerset object. It sits after the finite window route of chapter 126, the Cauchy seed carrier of chapter 3413, the regular rational readback of chapter 301, and the Frechet filter-basis neighbour of chapter 6498. The carrier is a traditional tail filter reconstructed as BEDC data: every accepted read displays the finite excluded prefix, the surviving tail window, the Cauchy and readback rows that consume the tail, structural transport, replay, provenance, and local naming.

Definition 7023.1 (Cofinite filter carrier)**.** *A* $\mathsf{CofiniteFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, B, T, C, R, H, Q, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *source whose finite windows are inspected.*
2. B *is the finite complement ledger naming the excluded prefix.*
3. T *is the surviving tail window generated by removing the entries listed in* B*.*
4. C *is the* $\mathsf{Cauchy}^{\uparrow}$ *consumer row that may read only the displayed tail window.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same tail.*
6. H *is componentwise* hsame *transport for* S, B, T, C, R*.*
7. Q *is displayed* Cont *replay for the tail-filter route.*
8. P *is* Pkg *provenance over* $\mathsf{CofiniteFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FrechetFilterBasis}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CofiniteFilter}^{\uparrow}}$ *row.*

Theorem 7023.2 (Cofinite filter tail stability)**.** *Let* $F = (S, B, T, C, R, H, Q, P, N)$ *be an accepted* $\mathsf{CofiniteFilter}^{\uparrow}$ *carrier. Any downstream read that refines the excluded finite prefix must produce a tail window still routed through the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. Thus cofinite-tail stability is a finite refinement of* B *and* T*, not a host assertion about arbitrary complements or a quotient filter equality.*

Proof. Project the accepted carrier in Definition 7023.1. The only data that can change under tail refinement are the finite complement ledger B and the displayed survivor window T.

The stream, Cauchy, and regular-readback rows remain fixed as coordinates of the same packet. Componentwise hsame transport keeps the refined prefix and tail inside the original BHist route, while Cont replay and Pkg provenance expose the same local naming row. Since no coordinate carries a host powerset complement, quotient filter, selected cofinal subsequence, countable-choice schedule, or ambient completeness principle, the refinement is exactly the displayed tail-stability witness. □

Theorem 7023.3 (Cofinite filter Cauchy-window handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofiniteFilter}^{\uparrow}$ *carrier receives the finite handoff*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{CofiniteFilter}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FrechetFilterBasis}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The handoff identifies the cofinite tail with the displayed StreamName window consumed by Cauchy and regular-rational rows. It does not export a quotient of filters, a selected eventual index, a completed stream, countable choice, host real equality, or an ambient Real-completeness theorem.

Proof. Read the tail row through Definition 7023.2. The finite complement ledger and surviving tail window are already downstream of the $\mathsf{StreamName}^{\uparrow}$ source and upstream of the Cauchy and regular-readback rows.

The Frechet filter-basis neighbour supplies the traditional filter reading, but the present packet exposes only the finite complement and tail rows. The Cauchy and RegSeqRat consumers therefore receive S, B, T, C, R with structural transport, replay, provenance, and local naming attached. The excluded quotient, selected-index, completed-stream, choice, host-equality, and completeness objects have no carrier coordinate from which to enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofiniteFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CofiniteFilter}^{\uparrow}$ packet is generated from a StreamName source, a finite complement ledger, a surviving tail window, Cauchy and RegSeqRat consumer rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofiniteFilter}^{\uparrow}}$, seed)

Scope closed: Definition 7023.1, Definition 7023.2, and Definition 7023.3 seed the finite cofinite-tail filter packet for L10 Cauchy and StreamName consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofiniteFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected eventual indices, completed streams, countable choice, host real equality, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-complement refinement, Cauchy-window handoff, regular-readback routing, transport, replay, provenance, and local naming for BEDC.Derived.CofiniteFilterUp.

Chapter 7024

A Concrete Naming Certificate for $\mathsf{SelfCompilationFeedback}^{\uparrow}$

The $\mathsf{SelfCompilationFeedback}^{\uparrow}$ packet realizes the ledgered-feedback branch of section 9411.4 as finite NameCert data. It packages a compiler-closure source genome, an expressed compiler phenotype, a feedback event flow into a later accepted genome, a refusal boundary for unledgered edits, structural transport, replay, provenance, and local naming. The packet is not a host account of file editing or an unrestricted reflection principle; every accepted feedback read must expose the finite ledger rows before it can be treated as self-compilation.

The packet consumes the compiler-closure self-compilation route of chapter 6317, the theorem-genome evolution route of chapter 6358, and the refusal boundary of chapter 6558 before a feedback event can be public.

Definition 7024.1 (SelfCompilationFeedback carrier)**.** *A* $\mathsf{SelfCompilationFeedback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, M, E, L, A, R, H, C, P, N)$$

with the following rows:

1. *G is the source genome-flow row.*
2. *M is the compiler-phenotype row expressed by the source genome.*
3. *E is the* EventFlow *row carrying the feedback from M into a later genome flow.*
4. *L is the visible feedback ledger required by Definition 9411.13.*
5. *A is the accepted-export row for the later genome flow.*
6. *R is the refusal boundary excluding unledgered feedback, external file edits, and unrestricted reflection.*
7. *H is componentwise* hsame *transport for G, M, E, L, A, R.*
8. *C is displayed* Cont *replay for the feedback route.*
9. *P is* Pkg *provenance over* $\mathsf{SelfCompilationFeedback}^{\uparrow}$*, compiler closure,* EventFlow*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SelfCompilationFeedback}^{\uparrow}}$ *row.*

Theorem 7024.2 (SelfCompilationFeedback ledger nonescape)**.** *Every accepted* SelfCompilationFeedback$^{\uparrow}$ *read that claims compiler-closure feedback must pass through the displayed source genome row, compiler-phenotype row, feedback* EventFlow *row, visible ledger row, accepted-export row, refusal boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. A consumer cannot replace those rows by an unledgered host edit, a detached theorem-generation trace, a hidden classifier change, an unrestricted reflection step, or a feedback object lacking local naming.*

Proof. Project the carrier in Definition 7024.1. The source genome, compiler phenotype, feedback event, visible ledger, accepted export, and refusal boundary are separate rows of the accepted packet.

Use the feedback witness of Definition 9411.13. It requires the feedback ledger to record the source genome carrier, compiler phenotype, behavior relation, target genome carrier, accepted-export route, naming obligations, closure-certificate obligations, and boundary excluding unrestricted reflection. The rows G, M, E, L, A, R are exactly the finite surface on which those requirements are read.

Finally repack through H, C, P, N. Componentwise transport and displayed replay preserve the same ledger route for consumers, provenance records its package boundary, and the local NameCert row names the accepted feedback carrier. Since the refused host edit, detached generation trace, hidden classifier change, unrestricted reflection step, and unnamed feedback object have no coordinate in the packet, they cannot escape the ledger boundary. □

Theorem 7024.3 (SelfCompilationFeedback closed-substitution route)**.** *Every accepted* SelfCompilationFeedback$^{\uparrow}$ *read that later reaches a MetaCIC substitution consumer must first expose the source genome row, compiler-phenotype row, feedback* EventFlow *row, visible ledger row, accepted-export row, refusal boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row of Definition 7024.1. Only after those rows are displayed may the read pass through chapter 538 or chapter 527 and be consumed as closed-substitution feedback.*

Proof. Project the carrier of Definition 7024.1. The feedback EventFlow row and the visible ledger row are separate coordinates, so a substitution-facing consumer cannot read feedback directly from a host compiler trace.

Apply Definition 7024.2. It forces every compiler-closure feedback claim through the source genome, compiler phenotype, feedback event, ledger, accepted export, refusal boundary, transport, replay, provenance, and local naming rows.

The closed-substitution boundary chapters then supply the only listed MetaCIC substitution surfaces. Routing through one of those chapters preserves the same displayed feedback ledger before any closed-substitution read is public, and the packet exports no detached substitution feedback object. □

Falsifiable prediction: Every accepted SelfCompilationFeedback$^{\uparrow}$ self-compilation read exposes source genome, compiler phenotype, feedback EventFlow, visible ledger, accepted export, refusal boundary, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any compiler-closure feedback claim is public.

Independence witness: compiler-closure-self-compilation, theorem-genome-evolution, automated-theory-discovery-pipeline-calculus, refutation-gate, namecert: no listed sibling is bijective with SelfCompilationFeedback$^{\uparrow}$ because this packet jointly carries a compiler-phenotype row, feedback EventFlow row, visible feedback ledger, accepted-export row, refusal boundary, transport, replay, provenance, and local NameCert data.

The Lean TasteGate surface for this packet must provide explicit Nontrivial SelfCompilationFeedback$^{\uparrow}$ and FieldFaithful SelfCompilationFeedback$^{\uparrow}$ instances in `lean4/BEDC/Derived/SelfCompilationFeedbackUp/Tast` Those instances must keep $G, M, E, L, A, R, H, C, P, N$ as distinct fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for NameCert$_{\text{SelfCompilationFeedback}^{\uparrow}}$.

Constructive story: A SelfCompilationFeedback$^{\uparrow}$ packet is generated from source genome-flow, compiler-phenotype, feedback EventFlow, visible ledger, accepted-export, refusal-boundary, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{SelfCompilationFeedback}^{\uparrow}}$, seed)

Scope closed: Definition 7024.1 and Definition 7024.2 seed the finite feedback-ledger carrier for compiler-closure self-compilation reads.

Formal verification: FormalStatus(NameCert$_{\text{SelfCompilationFeedback}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted reflection, hidden feedback, external file-edit self-compilation, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, ledger nonescape, accepted-export routing, refusal-boundary preservation, transport, replay, provenance, local naming, and TasteGate instances for BEDC.Derived.SelfCompilationFeedbac

Chapter 7025

A Concrete Naming Certificate for $\mathsf{LittlewoodPaley}^{\uparrow}$

From Stein/Grafakos-style Littlewood–Paley theory, the $\mathsf{LittlewoodPaley}^{\uparrow}$ packet records the finite dyadic-frequency inspection surface by which constructive harmonic-analysis consumers read scale windows, projection packets, and square-function ledgers before any real-valued bound is exported. It sits beside the Haar wavelet packet of chapter 3881, the Besov aggregate of chapter 4204, the Triebel–Lizorkin scale-position packet of chapter 6330, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed dyadic-frequency rows, finite projection data, square-function readback, transport, replay, provenance, and local naming; it does not import Fourier inversion, completed function-space quotients, almost-everywhere representatives, unconditional convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7025.1 (Littlewood–Paley carrier)**.** *A* $\mathsf{LittlewoodPaley}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, P, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *D is the dyadic scale-window row, recording the finite family of frequency annuli or low-frequency caps selected by the read.*
2. *P is the projection-packet row, listing the finite projection probes attached to the selected windows.*
3. *S is the square-function ledger, recording the finite quadratic or variation-style aggregation performed on the displayed projection packets.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational magnitudes attached to the displayed projection and square-function data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, P, S, R are visible.*
6. *H records componentwise* hsame *transport for the scale-window, projection, square-function, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Littlewood–Paley read through*

$$D, P, S, R, E, H \longmapsto Q, N.$$

8. *Q is the* Pkg *provenance over* $\mathsf{LittlewoodPaley}^{\uparrow}$, $\mathsf{HaarWavelet}^{\uparrow}$, $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LittlewoodPaley}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, P, S, R, E, H, C, Q, N$. It has no coordinate for a completed Fourier multiplier theorem, quotient function identity, selected distribution representative, almost-everywhere equality, unconditional infinite series, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 7025.2 (Littlewood–Paley NameCert obligations)**.** *For every accepted* $\mathsf{LittlewoodPaley}^{\uparrow}$ *carrier $L = (D, P, S, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{LittlewoodPaley}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dyadic scale windows, projection packets, square-function ledger, regular rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every harmonic-analysis, Besov, Triebel–Lizorkin, Haar-wavelet, or real-bound consumer of the packet must read through those rows before it cites a Littlewood–Paley square-function boundary.*

Proof. Carrier admission is projection from Definition 7025.1. The rows D, P, S display the finite scale-window, projection-packet, and square-function data before any consumer can read a harmonic-analysis bound.

The numerical route is finite. The row R supplies regular rational readback for the displayed projection and square-function magnitudes, and E seals the same data as a terminal $\mathsf{Real}^{\uparrow}$ read.

The structural rows preserve the displayed route. Since H, C, Q, N provide only componentwise transport, continuation replay, provenance, and local naming, and since the carrier has no coordinate for Fourier inversion, quotient function equality, selected representatives, almost-everywhere equality, unconditional infinite series, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7025.3 (Littlewood–Paley dyadic frequency handoff)**.** *Every accepted* $\mathsf{LittlewoodPaley}^{\uparrow}$ *read factors through the finite handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed dyadic scale-window, projection-packet, square-function, componentwise transport, replay, provenance, and local naming rows. HaarWavelet, BesovSpace, and TriebelLizorkinSpace consumers may inspect this handoff only as finite dyadic-frequency data; the route exports no host Fourier transform, no completed harmonic-analysis quotient, no selected representative of a distribution or function, no unconditional square-function identity, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the scale-window row D. It records only finitely many dyadic frequency windows, so every downstream read begins from displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data rather than from an ambient frequency space.

Next read the projection and square-function rows. The projection packet P lists the inspected probes, and the square-function ledger S records the finite aggregation visible to HaarWavelet, BesovSpace, and TriebelLizorkinSpace consumers.

Finally use the readback and seal rows. The regular rational row R and terminal Real row E attach numerical output to the same finite projection data, while H, C, Q, N transport, replay, source, and name the route. Since none of these rows carries a Fourier inversion theorem, quotient identity, selected representative, unconditional infinite series, or ambient completeness principle, the handoff exports only the displayed finite dyadic-frequency boundary. □

Falsifiable prediction: Every accepted $\mathsf{LittlewoodPaley}^{\uparrow}$ square-function read exposes dyadic scale windows, projection packets, square-function aggregation, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any harmonic-analysis consumer cites the Littlewood–Paley boundary.

Independence witness: haar-wavelet, besov-space, triebel-lizorkin-space, dyadicratcore, regseqrat, real: $\mathsf{LittlewoodPaley}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic-frequency windows, projection packets, square-function aggregation, regular rational readback, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LittlewoodPaley}^{\uparrow}}$.

Constructive story: A $\mathsf{LittlewoodPaley}^{\uparrow}$ packet is generated from dyadic scale windows, projection packets, square-function ledgers, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LittlewoodPaley}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7025.1, Definition 7025.2, and Definition 7025.3 seed the finite constructive Littlewood–Paley route over DyadicRatCore, RegSeqRat, Real, HaarWavelet, BesovSpace, TriebelLizorkinSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LittlewoodPaley}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, Fourier inversion, completed harmonic-analysis function spaces, quotient function equality, almost-everywhere representatives, unconditional infinite series, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale-window exactness, projection-packet exposure, square-function ledger stability, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LittlewoodPaleyUp.

Chapter 7026

A Concrete Naming Certificate for $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ packet records the finite evidence by which located-real data are read as a regular Cauchy convergence criterion without turning that read into an ambient completeness theorem. It sits beside the located cut refinement of chapter 7018, the Dedekind real surface of chapter 3629, the located-real packet of chapter 1933, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The certificate names a local convergence criterion from displayed located evidence to regular Cauchy readback; it does not assert quotient real equality, select a limit, or import completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7026.1 (Located real convergence criterion carrier)**.** *A* $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{LocatedReal}^{\uparrow}$ *source-evidence row, exposing lower and upper located witnesses only through the accepted located-real packet.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, naming the inspected precision requests, comparison cells, and tail observations used by the criterion.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from the same window schedule and carrying the rational Cauchy estimates exported to consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing positive dyadic margins, slack conversions, and comparison budgets used by W and R.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located source, stream windows, regular rational readback, and dyadic tolerance ledger have been displayed.*
6. *H records componentwise* hsame *transport for located source evidence, windows, readbacks, tolerances, seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a located-real request through A, W, R, D, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$, $\mathsf{DedekindReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host convergence, quotient real equality, selected Cauchy limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7026.2 (Located real convergence criterion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ *carrier* $L = (A, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealConvergenceCrite}}$ *surface consists exactly of carrier admission for the displayed packet,* $\mathsf{LocatedReal}^{\uparrow}$ *source exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 7026.1. The rows visible to the local name are precisely $A, W, R, D, E, H, C, P, N$.

The source obligation is the located-real row A. It exposes only displayed located evidence and therefore cannot by itself export a quotient real, a chosen limit, or a host convergence theorem.

The finite convergence-readback obligation is forced by W, R, D. The stream windows in W name the inspected requests, D supplies the dyadic tolerance budget for those requests, and R records the regular rational estimates obtained from that same schedule.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since no omitted coordinate exists for host convergence, quotient equality, selected limits, countable choice, trichotomy, or ambient completeness, these rows exhaust the local NameCert obligations. □

Theorem 7026.3 (Located real convergence criterion regular readback)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ *packet reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may be compared with the regular located-cut and Dedekind-real siblings, but it exports only the displayed local convergence criterion and not ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, a selected limit, or a detached real seal.*

Proof. Start with the located source row A. A consumer can inspect it only as displayed located-real evidence, so no endpoint seal or regular readback is available at this stage.

Read the finite windows and dyadic tolerance ledger next. The rows W and D bind each requested comparison to a displayed StreamName window and a positive dyadic budget.

The regular rational row R is downstream of the same windows and tolerances. It is therefore the first row from which the consumer can read regular Cauchy estimates.

The terminal seal E is reached only after R. Componentwise transport, continuation replay, provenance, and local naming preserve this order, so the packet cannot export completeness of $\mathsf{Real}^{\uparrow}$, quotient equality, a selected limit, or a detached seal. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ endpoint read factors through located-real source evidence, finite StreamName windows, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, transport, replay, provenance, and a local NameCert row.

Independence witness: regular-cauchy-located-cut, dedekind-real, located-real, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole nine-row packet $L = (A, W, R, D, E, H, C, P, N)$, because each sibling contributes only located evidence, cut comparison discipline, stream windows, regular readback, tolerance arithmetic, or endpoint sealing while this packet binds them into one local convergence-criterion route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}$ packet is generated from located-real source evidence, finite StreamName windows, a RegSeqRat regular readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7026.1, Definition 7026.2, and Definition 7026.3 seed the finite located-real convergence-criterion route from displayed located evidence to regular rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealConvergenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient real equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, finite window coverage, dyadic tolerance exactness, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealConvergenceCriterionUp.

Chapter 7027

A Concrete Naming Certificate for $\mathsf{ArchimedeanFloor}^{\uparrow}$

Following Bishop and Bridges on constructive analysis and integer-part extraction, the $\mathsf{ArchimedeanFloor}^{\uparrow}$ packet records the finite floor-window route used by real-number consumers when an explicit integer bracket has already been displayed. It sits over the integer source of chapter 31, the rational comparison surface of chapter 37, the dyadic arithmetic route of chapter 291, the stream-name and regular-sequence readback of chapter 301, and the terminal real seal of chapter 38. The packet names integer-part extraction only as a bounded finite NameCert window; it does not assert total real trichotomy, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a host floor function.

Definition 7027.1 (Archimedean floor carrier)**.** *An* $\mathsf{ArchimedeanFloor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, J, Q, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Int}^{\uparrow}$ *source row, displaying the lower integer candidate and its successor relation.*
2. *J is the successor-bound row, displaying the upper integer candidate as the successor of the lower row and refusing any detached integer choice.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row, where integer entries are read as rationals before any real-facing comparison is formed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row, carrying the finite dyadic budget used to inspect the two rational inequalities.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational name whose finite window is being compared.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, certifying that the compared regular sequence is the displayed real-name endpoint seen by downstream consumers.*
7. *H records componentwise* hsame *transport for the integer, rational, dyadic, stream-name, regular-sequence, real-seal, and local naming rows.*
8. *C records displayed* Cont *replay through the integer bracket, rational comparison, dyadic comparison, regular-sequence readback, real seal, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance row binding the integer bracket to the L10 Real and Completion route.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanFloor}^\uparrow}$ *row over* $\mathsf{ArchimedeanFloor}^\uparrow$, $\mathsf{Int}^\uparrow$, $\mathsf{Rat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert.

The classifier compares accepted packets only through $I, J, Q, D, S, R, H, C, P, N$. It has no coordinate for total trichotomy of real names, quotient real equality, selected limits, ambient real completeness, a global floor operator, or an integer-part handoff that skips the displayed window.

Theorem 7027.2 (Archimedean floor NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanFloor}^\uparrow$ *carrier*

$$F = (I, J, Q, D, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArchimedeanFloor}^\uparrow}$ *surface consists exactly of lower-integer source exposure, successor upper-row exposure, rational comparison exposure, dyadic comparison replay, stream-name and regular-sequence readback, real-seal exposure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from total real trichotomy, quotient real equality, selected limits, ambient* $\mathsf{Real}^\uparrow$ *completeness, host floor functions, and any integer-part route outside the displayed bracket.*

Proof. Project the carrier of Definition 7027.1. The rows I and J are the only integer rows, so the candidate floor and its successor are displayed before any comparison is read.

The comparison rows are Q and D. They convert the integer bracket into rational and dyadic inspection data, giving finite evidence for the lower and successor bounds without invoking trichotomy of arbitrary real names.

The readback rows S and R then identify the compared object with the displayed regular sequence and its real seal. The structural rows H, C, P, N transport, replay, source, and name that same window, so no host floor operator, selected limit, quotient real equality, or ambient real-completeness claim is exported. □

Theorem 7027.3 (Archimedean floor window extraction)**.** *Every integer-part handoff from an accepted* $\mathsf{ArchimedeanFloor}^\uparrow$ *carrier exposes the lower integer row, the successor upper row, rational comparison, dyadic comparison, regular-sequence readback, and the* $\mathsf{Real}^\uparrow$ *seal in the order*

$$\mathsf{Int}^\uparrow \longmapsto \mathsf{Rat}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

A downstream L10 Real and Completion consumer may receive the bracket only after the rows $I, J, Q, D, S, R, H, C, P, N$ have been displayed, and it receives no total trichotomy principle, quotient-real equality, selected limit, ambient completeness theorem, or hidden host integer-part operation.

Proof. Start with a requested integer-part handoff. The carrier first exposes I and J, so the lower integer and its successor are visible as finite integer rows rather than selected from an ambient ordered field.

The rational and dyadic comparison rows Q and D are then read before the real endpoint is sealed. These rows provide the finite lower-bound and successor-upper-bound checks consumed by the window.

Finally S reads back the stream-name and regular rational sequence, while R supplies the real seal. Transport, replay, provenance, and local naming preserve exactly this route through H, C, P, N, so the handoff is a bounded displayed window and not a global integer-part theorem for all real names. □

Falsifiable prediction: Any accepted $\mathsf{ArchimedeanFloor}^{\uparrow}$ consumer that performs an integer-part handoff without first exposing $I, J, Q, D, S, R, H, C, P, N$ refutes the carrier surface of Definition 7027.1.

Independence witness: int, rat, dyadicratcore, regseqrat, real: no listed sibling is bijective with $\mathsf{ArchimedeanFloor}^{\uparrow}$, because this packet alone binds a displayed integer successor window, rational comparison, dyadic comparison, regular-sequence readback, real seal, transport, replay, provenance, and local NameCert rows into one finite floor-window carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanFloor}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanFloor}^{\uparrow}$ packet is generated from an IntUp lower row, a successor upper row, RatUp comparison data, DyadicRatCoreUp finite comparison data, StreamNameUp and RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanFloor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7027.1, Definition 7027.2, and Definition 7027.3 seed the finite Archimedean floor-window packet over $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and BHist.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanFloor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total real trichotomy, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, a host floor function, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-integer exposure, successor upper-row exposure, rational comparison, dyadic comparison, stream-name and regular-sequence readback, RealUp sealing, transport, non-escape, and field faithfulness for BEDC.Derived.ArchimedeanFloorUp.

Chapter 7028

A Concrete Naming Certificate for $\mathsf{DiniUniformCauchy}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{DiniUniformCauchy}^{\uparrow}$ packet records the finite route by which a monotone family of regular real names yields a uniform Cauchy tail through displayed rational windows. It sits beside the finite observation interface of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, the dyadic tolerance core of chapter 291, the extracted modulus surface of chapter 369, and the lower-epigraph visibility of chapter 1536. The packet names only the finite monotone tail, dyadic tolerance, uniform modulus, regular readback, and real seal rows; it does not claim classical compactness, countable choice, quotient equality of real names, selected limiting values, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7028.1 (Dini uniform Cauchy carrier)**.** *A* $\mathsf{DiniUniformCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, M, L, T, Q, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, listing the finite observation schedule for the regular real-name family.*
2. *M is the monotone comparison row, recording that later sampled names lie below earlier sampled names on the inspected dyadic windows.*
3. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *visibility row, keeping the lower epigraph reads displayed before any tail budget is extracted.*
4. *T is the finite tail row, naming the inspected index threshold and the two tail positions compared against it.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, storing the rational error bound used by the tail comparison and the later readback.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row extracted from S, M, L, T, Q, exposing the same tail budget uniformly over the scheduled windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed uniform tail and dyadic tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after the regular-rational readback is visible.*

9. *H records componentwise* hsame *transport for the source, monotone comparison, lower-epigraph visibility, tail budget, dyadic tolerance, uniform modulus, regular readback, real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same monotone uniform Cauchy extraction route.*
11. *P is the* Pkg *provenance over* $\mathsf{DiniUniformCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, L, T, Q, U, R, E, H, C, P, N$. It has no coordinate for classical compactness, countable choice, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7028.2 (Dini uniform Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{DiniUniformCauchy}^{\uparrow}$ *carrier*

$$D = (S, M, L, T, Q, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed stream source, monotone comparison, lower-epigraph visibility, finite tail row, dyadic tolerance, extracted uniform modulus,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming rows.*

Proof. Project the accepted carrier. The stream source S, monotone comparison M, lower-epigraph row L, tail row T, dyadic tolerance Q, extracted modulus U, regular readback R, and real seal E are the only analytic coordinates available to a consumer.

The order of use is forced by the tuple. The tail row T is read only after the observed stream and monotone comparisons are visible, while Q supplies the dyadic tolerance against which that tail is judged.

The uniform Cauchy data enter through U. Since U is extracted from S, M, L, T, Q, the readback row R and the terminal seal E cannot bypass the displayed monotone tail and tolerance budget.

Finally, H, C, P, N transport, replay, source, and name the same finite route. The excluded objects have no carrier coordinate, so the local NameCert surface cannot read compactness, choice, quotient equality, selected limits, or ambient real completeness from this packet. □

Theorem 7028.3 (Dini monotone tail extraction)**.** *Every* $\mathsf{DiniUniformCauchy}^{\uparrow}$ *read that exposes a uniform Cauchy tail factors through the finite route*

$$S, M, L, T, Q, U.$$

In particular, the monotone comparison row and lower-epigraph visibility row are displayed before the tail threshold is used, and the dyadic tolerance row is displayed before the extracted $\mathsf{UniformModulus}^{\uparrow}$ *row is consumed.*

Proof. Use Definition 7028.2. It places the stream source, monotone comparison, lower-epigraph visibility, tail threshold, dyadic tolerance, and uniform modulus inside one accepted packet.

The monotone tail is not an external limit object. It is the row T read against the order comparisons in M and the lower-epigraph visibility in L.

The uniformity is also finite. The row U is downstream of the same S, M, L, T, Q data, so the consumer obtains a uniform Cauchy tail only by displaying those rows first. □

Theorem 7028.4 (Dini uniform Cauchy regular-real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DiniUniformCauchy}^{\uparrow}$ *packet reaches the terminal seal only through*

$$S, M, L, T, Q, U, R, E, H, C, P, N.$$

Thus the handoff exports a regular real-name Cauchy route controlled by displayed dyadic tolerance and uniform-modulus rows, not a quotient equality of real names, a selected limiting value, a compactness principle, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the finite route of Definition 7028.3. It supplies S, M, L, T, Q, U before any regular-rational readback is available.

Next consume the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R. This row is attached to the same tail threshold, dyadic tolerance, and uniform modulus, so it cannot be detached from the displayed Dini extraction route.

Finally read the terminal real-name seal E and the structural rows H, C, P, N. They transport, replay, source, and name the same finite packet. Since the forbidden host objects are absent from the carrier, the $\mathsf{Real}^{\uparrow}$ consumer receives only the regular real-name Cauchy handoff. □

Falsifiable prediction: A purported $\mathsf{DiniUniformCauchy}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying S, M, L, T, Q, U, R is not reading the carrier of Definition 7028.1.

Independence witness: streamname, regseqrat, real, dyadicratcore, uniformmodulus, lowersemicontinuous: $\mathsf{DiniUniformCauchy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries monotone comparison, lower-epigraph visibility, a finite tail threshold, dyadic tolerance, extracted uniform modulus, regular-rational readback, terminal real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiniUniformCauchy}^{\uparrow}$ packet is generated from StreamName observations, monotone comparison rows, LowerSemicontinuous epigraph visibility, finite tail thresholds, DyadicRatCore tolerance rows, UniformModulus extraction, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiniUniformCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 7028.1, Definition 7028.2, Definition 7028.3, and Definition 7028.4 seed the monotone uniform Cauchy extraction route over regular real names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiniUniformCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim classical compactness, countable choice, quotient equality of real names, selected limiting values, Lean verification, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, monotone tail extraction, dyadic tolerance ordering, uniform-modulus extraction, regular-rational readback, real-seal nonescape, transport preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformCauchyUp.

Chapter 7029

A Concrete Naming Certificate for $\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}$ packet records the finite modulus data needed to move from dyadic tolerances to regular rational names and then to a real-name seal. It sits after the dyadic tolerance rows of chapter 291, the stream schedule rows of chapter 126, the regular-sequence carrier of chapter 301, and the terminal real-name seal of chapter 38. The packet names a located modulus spectrum for Cauchy data; it does not select limits, quotient regular sequences by ambient equality, or claim completeness of a host real line.

Definition 7029.1 (Cauchy located modulus spectrum carrier)**.** *A* $\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, L, E, H, C, P, N)$$

with the following displayed rows.

1. *D is a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic rational scales used for the Cauchy comparison.*
2. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite observation windows at which those tolerances are read.*
3. *R is a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the rational sequence data and the finite regularity obligations visible through the schedule.*
4. *L is the located modulus row: for every displayed tolerance window in D and S, it records a finite cofinal tail witness at which the regularity row R is stable inside the named tolerance.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the visible regular sequence may be handed to the real-name interface without selecting a limit point.*
6. *H is componentwise* hsame *transport for D, S, R, L, E.*
7. *C is the displayed* Cont *replay row, preserving the same finite windows under continuation.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}}$ *row.*

Theorem 7029.2 (Cauchy located modulus spectrum NameCert obligations)**.** *For every accepted* $\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}$ *carrier* $M = (D, S, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedModulusSpect}}$ *surface is exhausted by carrier admission, classifier stability, cofinal modulus refinement, Real-seal handoff, and refusal of selected limits, quotient equality of regular names, and ambient completeness.*

Proof. Project the carrier onto its dyadic tolerance, stream schedule, regular rational, located modulus, and real-seal rows. Carrier admission is exactly the assertion that these rows occur together in one finite BHist packet with the local name row N.

Classifier stability is componentwise. The transport row H compares D, S, R, L, E by hsame, and the replay row C preserves the same finite schedule under Cont. Thus a consumer cannot alter the tolerance scale, schedule, regularity row, or real seal while keeping the local certificate name fixed.

Cofinal modulus refinement is finite-tail refinement inside the displayed schedule. Given a later dyadic tolerance window, use the located row L to move to a cofinal tail of S where the regularity row R is stable inside that tolerance; no countable choice or selected limit is invoked.

The Real-seal handoff reads only E. It certifies that the regular rational data can be presented to the $\mathsf{Real}^{\uparrow}$ name surface, while the packet still refuses quotient equality of names, a chosen limit object, and ambient completeness beyond the displayed finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName schedule rows, RegSeqRat regularity rows, a located cofinal-tail modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}}$, seed)

Scope closed: Definition 7029.1 and Definition 7029.2 seed the finite located-modulus spectrum from dyadic tolerances through stream schedules, regular rational rows, cofinal-tail witnesses, and a RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyLocatedModulusSpectrum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected real limits, quotient equality of regular names, ambient real completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, cofinal modulus refinement, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyLocatedModulusSpectrumUp.

Chapter 7030

A Concrete Naming Certificate for $\mathsf{NagataDimension}^{\uparrow}$

Following Nagata's covering dimension for metric spaces and Assouad's metric dimension tradition, the $\mathsf{NagataDimension}^{\uparrow}$ packet records a finite scale-cover certificate rather than a host theorem about arbitrary covers. It sits after the metric carrier of chapter 57, the cover-order surface of chapter 1853, the Lebesgue covering route of chapter 4205, the terminal real-name seal of chapter 38, and the Assouad scale-control neighbour of chapter 6325. The chapter names the standard Nagata-dimensional cover interface as BEDC rows; it does not assert a host dimension theorem, selected cover basis, quotient metric equality, countable choice, or ambient compactness.

Definition 7030.1 (Nagata dimension carrier)**.** *A* $\mathsf{NagataDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, S, C, L, A, R, H, T, P, N)$$

with the following visible rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the displayed distance observations used by every scale-cover request.*
2. *S is the scale ledger, listing the inspected radii and the finite multiplicity budgets attached to those radii.*
3. *C is the cover-family row, consisting of finite bounded-diameter families for the displayed scales.*
4. *L is the* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *comparison row used when the same finite cover is read as a Lebesgue-style refinement certificate.*
5. *A is the* $\mathsf{AssouadDimension}^{\uparrow}$ *scale-control row recording the neighbouring doubling and bounded-overlap discipline.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for numeric radius and diameter readback.*
7. *H records componentwise* hsame *transport for metric, scale, cover, comparison, Real-seal, provenance, and local naming rows.*
8. *T records displayed* Cont *replay through the same cover route.*
9. *P is the* Pkg *provenance over* $\mathsf{NagataDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{LebesgueCoveringDimension}^{\uparrow}$, $\mathsf{AssouadDimension}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NagataDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed metric source, scale ledger, cover families, comparison rows, Real readback, transport, replay, provenance, and local naming rows. It has no coordinate for a selected infinite cover basis, quotient metric equality, countable choice, open-cover compactness, or a host dimension theorem.

Theorem 7030.2 (Nagata dimension NameCert obligations)**.** *For an accepted* $\mathsf{NagataDimension}^{\uparrow}$ *carrier* $G = (M, S, C, L, A, R, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NagataDimension}^{\uparrow}}$ *surface consists exactly of metric-source admission, scale-ledger admission, bounded-diameter cover-family exposure, finite multiplicity accounting, Lebesgue-covering comparison, Assouad scale-control comparison, Real readback for radii and diameters, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations bind the finite Nagata cover certificate without adding a host cover basis, quotient metric equality, countable choice, open-cover compactness, or ambient dimension theorem.*

Proof. Project the carrier of Definition 7030.1. Its analytic rows are exactly the metric source M, scale ledger S, cover family C, Lebesgue-covering comparison L, Assouad scale-control row A, and Real readback row R.

The NameCert obligations are the rows needed to expose those data to a consumer. Transport through H and replay through T preserve the same finite route, while P and N source and name the packet. Since the carrier has no coordinate for the refused host principles, the listed rows exhaust the obligation surface. □

Theorem 7030.3 (Nagata dimension metric-cover route)**.** *Every metric-cover consumer of an accepted* $\mathsf{NagataDimension}^{\uparrow}$ *packet must read the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{CoveringDimension}^{\uparrow} \longmapsto \mathsf{LebesgueCoveringDimension}^{\uparrow} \longmapsto \mathsf{NagataDimension}^{\uparrow} \longmapsto \mathsf{AssouadDimension}^{\uparrow} \longmapsto$$

through the displayed rows $M, S, C, L, A, R, H, T, P, N$*. The route exposes scale-indexed bounded-diameter cover families and finite multiplicity budgets before any Real-facing radius or diameter seal is consumed, and it exports no selected cover basis, quotient metric equality, countable-choice schedule, open-cover compactness theorem, or ambient completion principle.*

Proof. Begin with the metric source M and scale ledger S. A cover request can inspect only the distances and radii already recorded in those finite rows.

Next read the cover-family row C together with the Lebesgue-covering and Assouad comparison rows L, A. These rows place the Nagata packet between the cover-order discipline of $\mathsf{CoveringDimension}^{\uparrow}$, the Lebesgue refinement surface, and the scale-control neighbour of $\mathsf{AssouadDimension}^{\uparrow}$.

Finally use the Real readback and structural rows. The row R supplies only displayed radius and diameter seals, while H, T, P, N preserve, replay, source, and name the same finite route. No omitted host principle can be recovered from those rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{NagataDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{NagataDimension}^{\uparrow}$ packet is generated from a displayed metric source, finite scale ledger, bounded-diameter cover families, multiplicity rows, Lebesgue-covering comparison, Assouad scale-control comparison, Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NagataDimension}^{\uparrow}}$, seed)

Scope closed: Definition 7030.1, Definition 7030.2, and Definition 7030.3 seed the finite Nagata dimension certificate over metric cover rows, Lebesgue-covering comparison, Assouad scale control, and Real readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NagataDimension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host dimension theorem, selected cover basis, quotient metric equality, countable choice, open-cover compactness, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scale-ledger exactness, cover-family boundedness, multiplicity accounting, Lebesgue comparison, Assouad comparison, Real readback nonescape, transport, replay, provenance, and local naming.

Chapter 7031

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$

The $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ packet realizes the phase-ledger branch of chapter 9463 as finite observer-facing geometry data. It sits after the boundary-shadow route of chapter 3813, the observer-expansion packet of chapter 4167, the phase-shell packet of chapter 6319, and the finite phase-transport row of chapter 3455. The packet names a finite ledger for hyperbolic phase readout; it is not a completed hyperbolic space, a physical observer, a quotient ideal boundary, or an ambient dynamical theorem.

Definition 7031.1 (Hyperbolic phase ledger carrier)**.** *A* $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, O, S, T, R, H, C, P, N)$$

with B *the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *boundary-shadow row,* O *the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *observer-expansion row,* S *the* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ *shell row,* T *the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *transport row,* R *the refusal row excluding completed boundary and physical-observer readings,* H *componentwise* hsame *transport,* C *displayed* Cont *replay,* P *the* Pkg *provenance, and* N *the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLedger}^{\uparrow}}$ *row.*

Theorem 7031.2 (Hyperbolic phase ledger NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ *carrier* $Q = (B, O, S, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLedger}^{\uparrow}}$ *surface is exhausted by boundary-shadow admission, observer-expansion admission, phase-shell admission, disk phase-transport replay, refusal exactness, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier onto B, O, S, T. These four rows are the only geometric and observer-facing data available to the packet, and each is already a finite NameCert surface rather than a host geometry.

The refusal row R is checked against those same displayed rows. It prevents a consumer from replacing boundary-shadow, observer-expansion, shell, or phase-transport data by a completed hyperbolic space, quotient ideal-boundary equality, physical observer trajectory, or global dynamical theorem.

Transport and replay are componentwise. The row H preserves the displayed packet under hsame, while C replays the same finite route under Cont. The provenance and local naming rows P, N then package exactly the exposed ledger and no additional analytic object. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ read is refuted if a consumer obtains a completed hyperbolic space, quotient ideal-boundary equality, physical observer trajectory, or global dynamical theorem without displaying HyperbolicBoundaryShadow, HyperbolicObserver-Expansion, HyperbolicObserverPhaseShell, PoincareDiskPhaseTransport, refusal, transport, replay, provenance, and local naming rows.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-observer-expansion, hyperbolic-observer-phase-shell, poincare-disk-phase-transport: none is bijective to $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ because the packet jointly carries boundary shadow, observer expansion, phase shell, disk phase transport, refusal exactness, transport, replay, provenance, and local NameCert rows.

The Lean TasteGate route must provide explicit $\mathsf{Nontrivial\ HyperbolicPhaseLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicP}$ instances before the packet can serve as a checked carrier-alignment witness. Those instances must preserve $B, O, S, T, R, H, C, P, N$ as separate rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseLedger}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverExpansion rows, HyperbolicObserverPhaseShell rows, PoincareDiskPhaseTransport rows, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7031.1, Definition 7031.2, the falsifiable prediction, and the independence witness seed the finite hyperbolic phase ledger over boundary-shadow, observer-expansion, phase-shell, disk-transport, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic spaces, quotient ideal-boundary equality, physical observer trajectories, global dynamical theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exposure, observer-expansion exposure, phase-shell exposure, disk phase-transport replay, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseLedgerUp.

Chapter 7032

A Concrete Naming Certificate for $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$

Following Bishop and Bridges, uniform limits on closed intervals are read constructively through displayed interval bounds, explicit uniform tail control, named stream windows, regular rational readbacks, and a terminal real-name seal. The $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ packet records that finite handoff surface between the closed-bounded-interval route of chapter 1591, the uniform-limit route of chapter 463, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. It names the closed-interval uniform-limit packet as BEDC data only; it does not claim Real completeness, Heine–Borel compactness, selected limits, quotient functions, countable choice, or host function equality.

Definition 7032.1 (Closed-interval uniform-limit carrier)**.** *A* $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying the closed interval endpoints and boundedness ledger used by the read.*
2. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *tail-control row, carrying the displayed uniform tolerance budget over the interval.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, recording the finite windows opened on the approximating function names before any limit-facing read is consumed.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values inspected inside the interval windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row attached to the finite readback route.*
6. *H records componentwise* hsame *transport for I, U, S, Q, E and the structural rows.*
7. *C records displayed* Cont *replay from closed-interval and uniform-tail requests through the same stream, regular-rational, and real-seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalUniformLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, U, S, Q, E, H, C, P, N$. It has no coordinate for interval compactness by Heine–Borel, selected limit functions, quotient function equality, countable choice, host extensional equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7032.2 (Closed-interval uniform-limit NameCert obligations)**.** *For every accepted* ClosedIntervalUniformL *carrier*

$$L = (I, U, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalUniformLimit}^{\uparrow}}$ *surface consists exactly of closed-bounded-interval admission, uniform-tail budget exposure, stream-window accounting, regular rational readback, terminal real-name sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any closed-interval uniform-limit consumer must read the sibling routes of chapter 1591, chapter 463, chapter 126, chapter 301, and chapter 38 through these rows before it can consume the packet.*

Proof. Project the carrier of Definition 7032.1. The closed interval source, uniform tail budget, stream windows, regular rational readback, and real seal are separate displayed rows, and the structural rows H, C, P, N preserve the same packet.

The analytic obligations are exhausted by I, U, S, Q, E. The interval row supplies the closed-bounded source, the uniform-limit row supplies the tolerance budget, the stream row opens the finite observation windows, the regular rational row gives the readback, and the real row seals the terminal observation.

The certificate obligations are exhausted by H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier contains no row for Heine–Borel compactness, selected limits, quotient functions, countable choice, host equality, or ambient completeness, the listed rows are exactly the local NameCert obligations. □

Theorem 7032.3 (Closed-interval uniform-limit L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff exports closed-interval source data, displayed uniform tail control, finite stream windows, regular rational readback, a terminal real-name seal, transport, replay, provenance, and local naming only. It does not export Real completeness, Heine–Borel compactness, selected limit functions, quotient function equality, countable choice, or host equality of functions.

Proof. Use Definition 7032.2. A consumer first reads the closed-bounded-interval row I, the uniform-limit row U, the stream row S, the regular rational readback row Q, and the real seal row E.

The rows H, C, P, N then carry the handoff as componentwise transport, displayed replay, provenance, and local naming. They do not add analytic content beyond the already displayed finite route.

Field exclusion completes the boundary. The carrier has no Heine–Borel row, no completed-real compactness row, no selected limit row, no quotient-function row, no countable-choice row, and no host function-equality row, so the L10 handoff is exactly the finite packet stated above. □

Falsifiable prediction: Every accepted $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ consumer displays a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source row, a $\mathsf{UniformLimit}^{\uparrow}$ tail-budget row, a $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any closed-interval uniform-limit claim is public.

Independence witness: closed-bounded-interval, uniformlimit, streamname, regseqrat, real: $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ is not bijective to any listed sibling because it binds the interval source, uniform tail budget, finite stream windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalUniformLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7032.1, Definition 7032.2, and Definition 7032.3 seed the finite closed-interval uniform-limit route through interval, uniform-tail, stream, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalUniformLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, Real completeness, Heine–Borel compactness, selected limits, quotient functions, countable choice, host function equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-interval exposure, uniform-tail budget exposure, stream-window accounting, regular rational readback, real-seal boundary, transport, replay, provenance, local naming, and non-escape over the same displayed packet.

Constructive story: A $\mathsf{ClosedIntervalUniformLimit}^{\uparrow}$ packet is generated from closed-bounded-interval source data, displayed uniform-tail control, finite stream windows, regular rational readback, a real-name seal, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Chapter 7033

A Concrete Naming Certificate for $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$

Following Lusternik and Schnirelmann, the $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ packet records finite categorical-cover data as a BEDC NameCert surface. It sits beside the finite antipodal obstruction route of chapter 5828, the combinatorial certificates of chapter 2003 and chapter 6975, the compact metric source of chapter 129, the finite face inventory of chapter 238, and the compact finite-cover route of chapter 6997. The packet names a finite cover by null-homotopy rows and categorical budget ledgers; it does not import a host homotopy category, selected global contractions, quotient sphere equality, arbitrary open-cover compactness, or a classical critical-point theorem.

Definition 7033.1 (Lusternik–Schnirelmann category carrier)**.** *A* $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, K, U, A, M, S, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact source row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover data and the finite source model used by the downstream consumer.*
2. *K is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *support row when the source is presented by a finite complex, including the face inventory and incidence rows used to form nerves of covers.*
3. *U is the finite categorical cover row, listing the displayed cover members and the membership evidence inspected by the packet.*
4. *A is the null-homotopy admission row for each displayed cover member, recording the finite contraction certificate visible to the same source.*
5. *M is the multiplicity and budget row, recording the displayed upper bound for the number of categorical pieces used by U.*
6. *S is the sphere or antipodal-obstruction handoff row, reading the Borsuk–Ulam, Sperner, and Tucker sibling packets only as finite obstruction consumers.*
7. *O is the lower-obstruction row, recording that a requested category reduction is refused by the displayed antipodal or combinatorial obstruction route.*
8. *H records componentwise* hsame *transport for compact source, simplicial support, cover, null-homotopy, budget, obstruction, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from finite-cover inspection through null-homotopy rows and obstruction rows to the category budget.*
10. *P is the* Pkg *provenance over* $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$, $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{SpernerLemma}^{\uparrow}$, $\mathsf{TuckerLemma}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LusternikSchnirelmannCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, U, A, M, S, O, H, C, P, N$. It has no coordinate for a completed homotopy category, arbitrary open covers, choice-selected contractions, quotient equality of spheres or maps, unrestricted critical-point theory, or ambient compactness.

Theorem 7033.2 (Lusternik–Schnirelmann category NameCert obligations)**.** *For every accepted* $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ *carrier $L = (X, K, U, A, M, S, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LusternikSchnirelmannC}}$ *surface consists exactly of compact source admission, optional finite simplicial support admission, finite categorical cover exposure, null-homotopy admission for displayed cover members, categorical budget exposure, antipodal or combinatorial obstruction handoff, lower-obstruction refusal, componentwise transport, displayed replay, provenance, and local naming. No local obligation is discharged by an ambient homotopy category, selected global contraction, arbitrary open cover, quotient sphere equality, or classical critical-point theorem.*

Proof. Project the carrier of Definition 7033.1. The source-facing rows are X and K, the categorical-cover rows are U, A, M, and the obstruction-facing rows are S, O.

The NameCert surface first admits the source. The compact metric row X supplies the finite cover data, while K supplies the finite simplicial support when a nerve or face inventory is used by the consumer.

The cover route is then rowwise. The finite cover row U lists only displayed members, the row A records the null-homotopy data attached to those members, and the budget row M records the category bound read from the same finite list.

Finally S, O, H, C, P, N bind the obstruction, replay, provenance, and local name to that same packet. Since the excluded host objects are not coordinates of L, they cannot become additional local NameCert obligations. □

Theorem 7033.3 (Finite categorical-cover route)**.** *Every accepted* $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ *consumer that reads a categorical upper bound must factor through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{SimplicialComplex}^{\uparrow}, \quad U, \quad A, \quad M$$

before the category budget is exposed. The route may reuse the finite-cover star-refinement and compact-metric sibling packets, but it exports no arbitrary open-cover compactness theorem, no selected contraction outside the displayed cover, no quotient equality of maps, and no ambient homotopy-category object.

Proof. Begin with the source rows. The compact metric row X supplies finite cover data, and the simplicial row K supplies only the displayed face or nerve inventory used by the consumer.

Read the cover rows in order. The row U exposes the finite family of categorical pieces, A attaches the null-homotopy admission data to each displayed piece, and M records the resulting budget.

The structural rows H, C, P, N transport, replay, source, and name the same route. Because the carrier lacks arbitrary open-cover, quotient-map, selected-contraction, and ambient homotopy-category coordinates, the upper-bound read remains exactly the finite $X, K, U, A, M, H, C, P, N$ surface. □

Theorem 7033.4 (Sphere consumer handoff)**.** *When an accepted* $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ *packet is consumed by a sphere or antipodal-obstruction argument, the handoff reads the sibling packets of chapter 5828, chapter 2003, and chapter 6975 only through the displayed obstruction row* S *and lower-refusal row* O*. Thus a sphere-facing consumer receives a finite categorical-cover obstruction surface, not a host sphere theorem, quotient antipodal space, selected critical point, or unrestricted LS critical-value theorem.*

Proof. Project S and O from the carrier. The handoff row S names the antipodal and combinatorial sibling packets as finite obstruction consumers, while O records the refusal of a smaller categorical cover only inside the same displayed packet.

The Borsuk–Ulam, Sperner, and Tucker chapters contribute routes already expressed as finite NameCert packets. They do not supply quotient sphere equality, selected antipodal points, or unrestricted critical-point data to this carrier.

Consequently the sphere-facing read is constrained by $X, K, U, A, M, S, O, H, C, P, N$. Transport, replay, provenance, and local naming keep the obstruction tied to the finite categorical cover rather than to an ambient topological theorem. □

Falsifiable prediction: Every accepted $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ consumer that claims a category budget or obstruction must display compact source data, finite cover rows, null-homotopy admission rows, a budget row, obstruction handoff rows, transport, replay, provenance, and local NameCert data before the read is public.

Independence witness: borsuk-ulam, sperner-lemma, tucker-lemma, compactmetric, simplicial-complex, finite-cover-star-refinement-modulus: $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries categorical cover rows, null-homotopy admissions, budget data, antipodal or combinatorial obstruction handoff, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LusternikSchnirelmannCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{LusternikSchnirelmannCategory}^{\uparrow}$ packet is generated from compact source rows, optional finite simplicial support rows, finite categorical cover rows, null-homotopy admission rows, categorical budget rows, antipodal or combinatorial obstruction handoff, lower-obstruction refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LusternikSchnirelmannCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7033.1, Definition 7033.2, Definition 7033.3, and Definition 7033.4 seed the finite categorical-cover packet over compact, simplicial, antipodal, combinatorial, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LusternikSchnirelmannCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient homotopy categories, arbitrary open-cover compactness, selected global contractions, quotient sphere equality, unrestricted critical-point theory, or classical LS critical-value theorems.

Upgrade path: Obligation closure requires checked carrier admission, finite categorical cover exposure, null-homotopy admission, budget exactness, obstruction handoff, lower-refusal stability, transport, replay, provenance, and local naming for BEDC.Derived.LusternikSchnirelmannCategoryUp.

Chapter 7034

A Concrete Naming Certificate for HellySelectionCompactVariation$^{\uparrow}$

Following Helly, Jordan, and Bishop–Bridges constructive analysis, the HellySelectionCompactVariation$^{\uparrow}$ packet records the compact-variation selection route by finite partition, window, and diagonal ledgers rather than by a host subsequence theorem. It sits over the bounded-variation partition surface of chapter 398, the compact finite-window source of chapter 129, the subsequence-window discipline of chapter 2372, the terminal real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives L10 Real and Completion consumers a traditional compact-variation selection object as finite NameCert data; it does not assert open-cover compactness, countable choice, host function-space compactness, quotient equality of functions, or ambient completeness of Real$^{\uparrow}$.

Definition 7034.1 (Helly selection compact-variation carrier)**.** *A* HellySelectionCompactVariation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (V, K, D, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *V is the bounded-variation partition ledger, read through the* BoundedVariation$^{\uparrow}$ *surface and carrying the displayed partition sums, variation budget, and monotone refinement rows.*
2. *K is the compact finite-window selection row, read through* CompactMetric$^{\uparrow}$ *and exposing only finite net windows for the inspected domain points.*
3. *D is the diagonal subsequence ledger, recording the finite depth schedule and the retained subsequence windows before any limit row is sealed.*
4. *W is the* StreamName$^{\uparrow}$ *window row for the selected function values and endpoint probes.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row for the displayed pointwise limit windows.*
7. *T records componentwise* hsame *transport for V, K, D, W, R, E.*
8. *C records displayed* Cont *replay from bounded-variation partition checks through compact finite windows, diagonal windows, regular readback, and the Real seal.*
9. *P is the* Pkg *provenance over* HellySelectionCompactVariation$^{\uparrow}$, BoundedVariation$^{\uparrow}$, CompactMetric$^{\uparrow}$, SequentialCompact$^{\uparrow}$, Real$^{\uparrow}$, Cauchy$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{HellySelectionCompactVariation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $V, K, D, W, R, E, T, C, P, N$. It has no coordinate for a choice-selected subsequence, host compactness theorem, completed function-space quotient, arbitrary open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7034.2 (Helly selection compact-variation NameCert obligations)**.** *For every accepted* $\mathsf{HellySelectionCompactVariation}^{\uparrow}$ *carrier*

$$H = (V, K, D, W, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellySelectionCompactVariation}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-variation partition exposure through V, compact finite-window selection through K, diagonal subsequence scheduling through D,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through T, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7034.1. The variation row V is the only bounded-variation source, and the compact row K is the only compact finite-window source.

Read the diagonal stage next. The row D records the retained subsequence windows as displayed finite depth data, while W and R turn those windows into regular rational readback.

Finally the row E seals the displayed real-name windows. The structural rows T, C, P, N transport, replay, source, and name the same finite route, so no host compactness theorem, selected subsequence functional, quotient function equality, countable-choice schedule, or ambient Real completeness principle is exported. □

Theorem 7034.3 (Helly selection compact-variation diagonal extraction)**.** *Every diagonal extraction read from an accepted* $\mathsf{HellySelectionCompactVariation}^{\uparrow}$ *packet factors through the ordered route*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{Cauchy}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer must first read the bounded-variation partition ledger V, then the compact finite-window row K, then the diagonal subsequence ledger D, then the Cauchy and regular-readback windows W, R, and only then the $\mathsf{Real}^{\uparrow}$ *seal E with T, C, P, N attached. The extraction exports no selected host subsequence, no function-space compactness theorem, no quotient equality of functions, and no ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the bounded-variation row. It supplies finite partition variation bounds and no global variation supremum or completed function-space object.

Use the compact finite-window row next. The compact input is read through displayed finite windows from $\mathsf{CompactMetric}^{\uparrow}$, so the diagonal ledger receives finite net data rather than an open-cover compactness principle.

The diagonal row is then read through $\mathsf{SequentialCompact}^{\uparrow}$ as a finite retained-window schedule. The Cauchy and regular-readback rows W, R turn that schedule into displayed tail data, and the terminal row E seals only those real-name windows.

Transport, replay, provenance, and local naming preserve this route through T, C, P, N. Since the carrier has no field for the excluded host principles, every diagonal extraction remains a finite compact-variation NameCert readback. □

Falsifiable prediction: Every accepted $\mathsf{HellySelectionCompactVariation}^{\uparrow}$ consumer must expose the bounded-variation partition ledger, compact finite-window row, diagonal subsequence ledger, Cauchy readback windows, and $\mathsf{Real}^{\uparrow}$ seal before reading a compact-variation selection output.

Independence witness: boundedvariation, compactmetric, sequentialcompact, cauchy, real: none is bijective to $\mathsf{HellySelectionCompactVariation}^{\uparrow}$ because this packet jointly carries bounded-variation partitions, compact finite-window selection, diagonal subsequence scheduling, Cauchy readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HellySelectionCompactVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{HellySelectionCompactVariation}^{\uparrow}$ packet is generated from bounded-variation partition rows, compact finite-window selection rows, a diagonal subsequence ledger, Cauchy and regular-readback windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellySelectionCompactVariation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7034.1, Definition 7034.2, and Definition 7034.3 seed the finite compact-variation selection packet over $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellySelectionCompactVariation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequence choice, host function-space compactness, quotient equality of functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation partition exposure, compact finite-window selection, diagonal extraction, Cauchy readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HellySelectionCompactVariationUp.

Chapter 7035

A Concrete Naming Certificate for $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$

The $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ packet realizes the audit-gate boundary vision of chapter 9396 as finite NameCert data. It records the refusal surface where a source axiom scan, strict proof-ancestry report, paper-Lean name-resolution check, and origin-boundary ledger reject a proposed internal row instead of converting the failed audit into substrate theorem content. The packet sits beside the existing audit-gate boundary of chapter 526, the borrowed-recursor boundary named in the source vision, and the axiom-refusal discipline enforced by the Lean and paper audits.

Definition 7035.1 (Audit-gate boundary refusal carrier)**.** *An* $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, V, A, R, B, T, C, P, N)$$

with the following displayed rows:

1. *S is the gate-surface row, naming the checked syntactic, proof-ancestry, import, or paper-Lean marker condition.*
2. *V is the verdict row, recording pass or refusal together with the finite diagnostic token, dependency, import edge, or unresolved name that triggered the verdict.*
3. *A is the audit-replay row, exposing the deterministic replay route by which the verdict is recomputed from S.*
4. *R is the refusal row, recording that a failed gate supplies no internal proof row, no theorem row, and no carrier coordinate for the rejected content.*
5. *B is the borrowed-recursor boundary row, separating host checking infrastructure from forbidden host axiom propagation.*
6. *T records componentwise* hsame *transport for S, V, A, R, B.*
7. *C records displayed* Cont *replay through the same gate surface, verdict, replay, refusal, and boundary rows.*
8. *P is the* Pkg *provenance over* $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{AxiomRefusalLedger}^{\uparrow}$, $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$, $\mathsf{AuditAxisSeparation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditGateBoundaryRefusal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, V, A, R, B, T, C, P, N$. It has no coordinate that turns an audit failure into an internal proof, no bridge from failed verdicts to theorem rows, and no route that collapses the theory and verification axes.

Theorem 7035.2 (Audit-gate boundary refusal NameCert obligations)**.** *For every accepted* AuditGateBoundaryRefu *carrier*

$$G = (S, V, A, R, B, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AuditGateBoundaryRefusal}^{\uparrow}}$ *surface consists exactly of gate-surface exposure through S, verdict exposure through V, deterministic audit replay through A, refusal noninternalization through R, borrowed-recursor boundary separation through B, componentwise* hsame *transport through T, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7035.1. The gate row S and verdict row V are the only rows that can describe the checked boundary and its result.

Read the replay and refusal rows next. The row A recomputes the verdict from the displayed gate surface, while R records that a failed verdict exports no proof row and no theorem content.

The borrowed-recursor row B keeps host checking infrastructure separate from forbidden proof ancestry. Transport, replay, provenance, and local naming through T, C, P, N preserve the same refusal surface, so a failed audit cannot be reclassified as an internal BEDC proof. □

Theorem 7035.3 (Audit-gate boundary refusal noninternalization)**.** *Every consumer of an accepted* $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ *packet must read the gate surface, verdict, audit replay, refusal row, borrowed-recursor boundary row, transport, replay, provenance, and local naming before it can report an audit result. If the verdict row records failure, the packet exports only the refusal and diagnostic surface; it exports no internal proof row, no checked theorem row for the rejected content, no hidden axiom, and no theory-axis closure upgrade.*

Proof. Start with the verdict row V. A consumer cannot inspect the verdict without the gate surface S and replay row A, because the packet contains no detached result coordinate.

When V records failure, the refusal row R is the only downstream content attached to that verdict. It says that the failed item remains outside the substrate theorem surface.

The borrowed-recursor boundary row B prevents the consumer from using host checking machinery as a replacement proof source. The structural rows T, C, P, N preserve the same displayed route, so failure can be replayed and named but not internalized. □

Falsifiable prediction: No accepted $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ packet can turn an audit failure into an internal proof row; any such packet must either change the verdict row to pass by displayed replay or remain a refusal surface with no theorem-content export.

Independence witness: AxiomRefusalLedger, BorrowedRecursorBoundary, AuditAxisSeparation, AuditGateBoundary: $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries gate-surface data, verdict rows, audit replay, refusal noninternalization, borrowed-recursor boundary separation, transport, replay, provenance, and local NameCert rows.

Theorem 7035.4 (Audit-gate boundary refusal follow-up requirements)**.** *The Lean follow-up surface for* $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ *requires a nontrivial instance showing that at least one displayed failed gate is carried as a refusal row, and a field-faithful instance showing that changing any of $S, V, A, R, B, T, C, P, N$ changes the accepted refusal packet. These requirements are targets for future declarations named by the* Nontrivial *and* FieldFaithful *instance patterns over $BEDC.Derived.AuditGateBoundaryRefusalUp$.*

Proof. The nontrivial requirement reads the verdict and refusal rows together. A carrier whose failed verdict cannot be distinguished from a pass verdict would not expose boundary refusal.

The field-faithful requirement reads each carrier coordinate. If any row can be changed without changing the accepted packet, a consumer could hide the gate surface, verdict, replay route, refusal, boundary row, transport, replay, provenance, or local name.

Thus the two follow-up requirements are exactly the instance obligations needed before this packet can be promoted from paper-side refusal surface to checked Lean structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditGateBoundaryRefusal}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditGateBoundaryRefusal}^{\uparrow}$ packet is generated from a finite gate surface, verdict row, audit-replay row, refusal noninternalization row, borrowed-recursor boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditGateBoundaryRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7035.1, Definition 7035.2, and Definition 7035.3 seed the finite refusal surface for audit-gate failures.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditGateBoundaryRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean instances for nontriviality or field faithfulness, bridge checking, a theorem that every audit gate is complete, or any conversion of failed audits into internal proof rows.

Upgrade path: Obligation closure requires checked carrier admission, verdict replay, refusal noninternalization, borrowed-recursor boundary separation, nontrivial and field-faithful instances, transport, replay, provenance, and local naming for BEDC.Derived.AuditGateBoundaryRefusalUp.

Chapter 7036

A Concrete Naming Certificate for $\mathsf{DeGiorgiIteration}^{\uparrow}$

Following De Giorgi and Evans on energy methods for elliptic and parabolic regularity, the $\mathsf{DeGiorgiIteration}^{\uparrow}$ packet records the finite BEDC surface by which level-set truncations, nested energy windows, Sobolev control, Poincare control, dyadic budgets, regular rational readback, and a terminal real-name seal are replayed before a regularity consumer reads decay. It sits beside the Sobolev source of chapter 440, the Poincare control route of chapter 2329, the ladder comparison of chapter 6969, the dyadic budget core of chapter 291, the regular rational readback route of chapter 301, and the terminal scalar seal of chapter 38. The packet names only displayed finite rows; it is not a host PDE theorem, not a selected weak solution, not a quotient equality of function spaces, and not an ambient compactness argument.

Definition 7036.1 (DeGiorgiIteration carrier)**.** *A* $\mathsf{DeGiorgiIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, T, S, P, M, B, Q, R, H, C, G, N)$$

with the following displayed rows:

1. *L is the finite level-set row, listing the cutoff levels and superlevel or sublevel slices read by the iteration.*
2. *T is the truncation-window row, recording positive and negative parts, clipped test functions, and the local support window.*
3. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row used to pass from energy control to an integrability gain.*
4. *P is the* $\mathsf{PoincareInequality}^{\uparrow}$ *control row used on each displayed window.*
5. *M is the* $\mathsf{MoserIteration}^{\uparrow}$ *comparison row, exposing the sibling ladder against which De Giorgi decay consumers compare their exponent schedule.*
6. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row for thresholds, shrink factors, and precision tolerances.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the dyadic budget and energy ledger into regular rational comparisons.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the final bound read by downstream regularity consumers.*
9. *H records componentwise* hsame *transport for L, T, S, P, M, B, Q, R.*

10. *C records displayed* Cont *replay for the level-set, truncation, energy, budget, readback, and seal routes.*
11. *G is the* Pkg *provenance over* $\mathsf{DeGiorgiIteration}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, $\mathsf{MoserIteration}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DeGiorgiIteration}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.DeGiorgiIterationUp.DeGiorgiIterationUp`

Theorem 7036.2 (DeGiorgiIteration NameCert obligations)**.** *For every accepted* $\mathsf{DeGiorgiIteration}^{\uparrow}$ *carrier $D = (L, T, S, P, M, B, Q, R, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{DeGiorgiIteration}^{\uparrow}}$ *surface consists exactly of level-set exposure through L, truncation-window exposure through T, Sobolev source exposure through S, Poincare control through P, Moser-comparison exposure through M, dyadic budget exposure through B, regular rational readback through Q, Real-bound sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 7036.1. Each row named in the statement is a displayed coordinate of the accepted packet, and there is no second coordinate for an ambient solution space or host compactness theorem.

The analytic controls are read only through their sibling packets. The Sobolev row, Poincare row, and Moser comparison row point to the finite BEDC surfaces already named by their chapters, while the dyadic, regular rational, and Real rows keep the energy budget on registered numeric readbacks.

Transport and replay use H and C, and provenance and local naming use G and N. Since the packet inventory is exhausted by these rows, the local NameCert obligations are exactly the listed exposure, stability, replay, provenance, and non-escape clauses. □

Lean4 ✓ `BEDC.Derived.DeGiorgiIterationUp.DeGiorgiIterationNameCertObligations`

Theorem 7036.3 (DeGiorgiIteration energy-decay handoff)**.** *Every downstream regularity consumer of an accepted* $\mathsf{DeGiorgiIteration}^{\uparrow}$ *packet must first read the level-set row L, truncation-window row T, Sobolev row S, Poincare row P, Moser-comparison row M, dyadic budget row B, regular rational readback row Q, Real-bound seal R, and structural rows H, C, G, N. The handoff supplies a finite energy-decay surface; it exports no host De Giorgi theorem, selected weak solution, quotient function equality, countable-choice schedule, unbounded compactness principle, or ambient real-completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 7036.2. They force every public read through the displayed packet rows before any decay claim is consumed.

The energy side is finite because the level-set and truncation rows are ledger coordinates, and the Sobolev and Poincare rows are sibling NameCert inputs. The comparison with $\mathsf{MoserIteration}^{\uparrow}$ is likewise a displayed handoff row, not an imported host regularity theorem.

The numeric side is sealed by $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. Therefore a consumer receives only the finite decay packet and cannot extract any excluded analytic or choice-theoretic object from the handoff. □

Falsifiable prediction: Every accepted regularity read of $\mathsf{DeGiorgiIteration}^{\uparrow}$ exposes level-set, truncation-window, Sobolev, Poincare, Moser-comparison, dyadic budget, regular rational readback, Real-bound seal, transport, replay, provenance, and local NameCert rows before consuming an energy-decay bound.

Independence witness: sobolev, poincare-inequality, moser-iteration, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{DeGiorgiIteration}^{\uparrow}$ because this carrier alone binds level-set truncation windows and energy-decay budgets to the Sobolev, Poincare, Moser-comparison, dyadic, regular rational, Real, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DeGiorgiIteration}^{\uparrow}}$.

Constructive story: A $\mathsf{DeGiorgiIteration}^{\uparrow}$ packet is generated from level-set rows, truncation-window rows, Sobolev control, Poincare control, Moser-comparison rows, dyadic budgets, regular rational readback, a Real-bound seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DeGiorgiIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7036.1, Definition 7036.2, and Definition 7036.3 seed the finite De Giorgi energy-decay packet consumed downstream from Sobolev, Poincare, and Moser-style regularity rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DeGiorgiIteration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host De Giorgi regularity theorem, selected weak solutions, quotient function equality, countable choice, ambient compactness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, level-set exactness, truncation-window exactness, Sobolev and Poincare source exactness, Moser-comparison exactness, dyadic-budget exactness, regular rational readback, Real-bound sealing, transport, replay, provenance, and local naming for BEDC.Derived.DeGiorgiIterationUp.

Chapter 7037

A Concrete Naming Certificate for $\mathsf{ContourReparameterizationLedger}^{\uparrow}$

Following the constructive contour-integration operation law of chapter 9340, the $\mathsf{ContourReparameterizationLedger}^{\uparrow}$ packet records the finite ledger by which a piecewise-linear contour refinement or monotone piecewise-linear reparameterization is read without changing the contour-integral output. It sits beside the operation packet of chapter 521, the contour socket of chapter 583, the reversal ledger of chapter 1019, the holomorphic input surface of chapter 68, and the complex output seal of chapter 39. The packet names only finite contour, refinement, Riemann-sum, transport, replay, provenance, and local naming rows; it is not a host path quotient or an ambient smooth-change-of-variables theorem.

Definition 7037.1 (Contour reparameterization ledger carrier)**.** *A* $\mathsf{ContourReparameterizationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, G', U, F, S, M, I, H, C, P, N)$$

with the following displayed rows:

1. *G is the source* $\mathsf{PLContour}$ *row with finite* $\mathsf{Complex}^{\uparrow}$ *endpoints and segment rows.*
2. *G' is the reparameterized* $\mathsf{PLContour}$ *row, obtained by a finite refinement or monotone piecewise-linear parameter change.*
3. *U is the reparameterization ledger row recording the segment subdivision map, endpoint preservation, orientation preservation, and refinement budget.*
4. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row on a neighbourhood containing the images of both contour rows.*
5. *S is the Riemann-sum comparison ledger matching source and reparameterized sums.*
6. *M is the modulus row witnessing that the comparison remains within the displayed tolerance budget.*
7. *I is the shared* $\mathsf{Complex}^{\uparrow}$ *output row for the contour-integral operation.*
8. *H records componentwise* hsame *transport for G, G', U, F, S, M, I.*
9. *C records displayed* Cont *replay for contour reads, reparameterization reads, Riemann-sum reads, modulus reads, output reads, provenance, and local naming.*

10. *P is the* Pkg *provenance over* ContourReparameterizationLedger$^{\uparrow}$, ContourIntegralOperation$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourReparameterizationLedger}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ContourReparameterizationLedgerUp.ContourReparameterizationLedgerTasteGate`
`single carrier alignment`

Theorem 7037.2 (Contour reparameterization ledger NameCert obligations)**.** *For every accepted* ContourReparameterizationLedger$^{\uparrow}$ *carrier $R = (G, G', U, F, S, M, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContourReparameterizationLedger}^{\uparrow}}$ *surface consists exactly of source-contour exposure through G, reparameterized-contour exposure through G', finite reparameterization-ledger exposure through U, holomorphic input exposure through F, Riemann-sum comparison through S, modulus control through M, shared* Complex$^{\uparrow}$ *output sealing through I, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7037.1. The source contour, reparameterized contour, ledger, holomorphic input, comparison sums, modulus, output, transport, replay, provenance, and naming rows are all displayed coordinates.

The operation law is read through the finite rows U, S, M, I. The ledger records the refinement map and endpoint preservation, while the Riemann-sum comparison and modulus rows witness that both contour presentations reach the same Complex$^{\uparrow}$ output row.

Transport and replay do not add a host path quotient or a smooth reparameterization theorem. Since provenance and local naming are also displayed, the NameCert surface is exhausted by the rows listed in the statement. □

Theorem 7037.3 (Contour reparameterization ledger contour-integral handoff)**.** *Every contour-integral consumer that reads reparameterization invariance from an accepted* ContourReparameterizationLedger$^{\uparrow}$ *packet must first expose the source contour row G, reparameterized contour row G', reparameterization ledger U, holomorphic input row F, Riemann-sum comparison ledger S, modulus row M, shared* Complex$^{\uparrow}$ *output row I, and structural rows H, C, P, N. The handoff realizes the operation law as finite ledger data and exports no host path quotient, unregistered smooth parameter change, ambient analytic integral operator, or global branch selector.*

Proof. Apply Definition 7037.2. It forces every public reparameterization read through the displayed source contour, target contour, ledger, input, comparison, modulus, output, transport, replay, provenance, and naming rows.

The contour-integral operation packet supplies the downstream operation surface, but this chapter supplies the particular law ledger. The consumer therefore sees reparameterization invariance as a finite comparison between two PL-contour rows and their Riemann-sum ledgers.

No excluded ambient object has a carrier coordinate. The handoff can be consumed by contour, residue, and RH scaffold chapters only through the displayed rows of this packet and the existing contour-integral operation surface. □

Falsifiable prediction: Every accepted contour-integral reparameterization read exposes source contour, reparameterized contour, finite ledger, holomorphic input, Riemann-sum comparison, modulus, shared Complex output, transport, replay, provenance, and local NameCert rows before the operation law is public.

Independence witness: contourintegraloperation, contourintegralsocket, contour-reversal-ledger, holomorphic, complex: none is bijective with ContourReparameterizationLedger$^{\uparrow}$ because this carrier alone binds a finite reparameterization ledger and Riemann-sum comparison to the contour

operation, holomorphic input, Complex output, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ContourReparameterizationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ContourReparameterizationLedger}^{\uparrow}$ packet is generated from source PL-contour rows, reparameterized PL-contour rows, finite reparameterization ledgers, holomorphic input rows, Riemann-sum comparison rows, modulus rows, shared Complex output rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourReparameterizationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7037.1, Definition 7037.2, and Definition 7037.3 seed the finite ledger for the contour-integral reparameterization operation law.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContourReparameterizationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host path quotient, smooth change-of-variables theorem, ambient contour-integral operator, global branch selector, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source and target PL-contour exactness, reparameterization-ledger exactness, holomorphic-input locality, Riemann-sum comparison, modulus control, Complex output sealing, transport, replay, provenance, and local naming for BEDC.Derived.ContourReparameterizationLedgerUp.

Chapter 7038

A Concrete Naming Certificate for $\mathsf{BoundedStoppingTime}^{\uparrow}$

Following Doob, the $\mathsf{BoundedStoppingTime}^{\uparrow}$ packet names the finite stopping interface that sits between the stopping-decision surface of chapter 344 and the stopped-value surface of chapter 383. It reads martingale windows from chapter 190, probability sources from chapter 184, and random-variable endpoint rows from chapter 185. The packet records the displayed finite bound, filtration-window visibility, and decision ledger needed before optional stopping or concentration consumers may read a stopped value; it does not assert an unbounded stopping theorem, almost-sure quotient equality, uniform integrability, or measure completion.

Definition 7038.1 (Bounded stopping time carrier)**.** *A* $\mathsf{BoundedStoppingTime}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, b, F, D, V, H, C, P, N)$$

with the following displayed rows:

1. *T is the source* $\mathsf{StoppingTime}^{\uparrow}$ *row whose decisions are inspected.*
2. *b is the finite bound row, naming the displayed horizon before any stopped value is read.*
3. *F is the filtration-window visibility ledger, recording that each decision of T used before b is visible in the corresponding finite filtration window.*
4. *D is the decision ledger, listing the finite branch outcomes up to b and the first displayed stopping branch when one is present.*
5. *V is the stopped-value handoff row consumed by* $\mathsf{OptionalStopping}^{\uparrow}$*, tying the selected branch of D to the corresponding* $\mathsf{RandomVar}^{\uparrow}$ *endpoint row inside the martingale window.*
6. *H records componentwise* hsame *transport for T, b, F, D, V and the structural certificate rows.*
7. *C records displayed* Cont *replay from a bounded stopping request through the same stopping-time, bound, filtration-window, decision-ledger, and stopped-value rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedStoppingTime}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedStoppingTime}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, b, F, D, V, H, C, P, N$. It has no coordinate for an unbounded stopping index, optional-stopping expectation theorem, almost-sure quotient, completed filtration, uniform-integrability hypothesis, or ambient measure-completion carrier.

Theorem 7038.2 (Bounded stopping time NameCert obligations)**.** *For every accepted* $\mathsf{BoundedStoppingTime}^{\uparrow}$ *carrier*

$$B = (T, b, F, D, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedStoppingTime}^{\uparrow}}$ *surface consists exactly of source stopping-time exposure through* T*, finite-bound exposure through* b*, filtration-window visibility through* F*, decision-ledger exactness through* D*, stopped-value handoff through* V*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7038.1. The rows T, b, F, D, V are the only mathematical rows available to a bounded stopping consumer.

The source stopping-time row T supplies the finite decision surface, and the bound row b cuts that surface to the displayed window. The filtration ledger F records that each inspected decision is visible at the corresponding finite stage.

The decision ledger D then identifies the bounded branch data consumed by the stopped-value handoff V. The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains no unbounded stopping index, optional-stopping expectation theorem, quotient event equality, completed filtration, uniform-integrability row, or measure-completion row, the listed rows are exactly the local naming obligations. □

Theorem 7038.3 (Bounded stopping time optional-stopping handoff)**.** *Every* $\mathsf{OptionalStopping}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedStoppingTime}^{\uparrow}$ *carrier*

$$B = (T, b, F, D, V, H, C, P, N)$$

must read the stopped-value handoff through the finite route

$\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{BoundedStoppingTime}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The handoff exposes the source stopping row, displayed bound, filtration-window ledger, bounded decision ledger, and stopped-value row before any optional-stopping packet can inspect expectation-facing data. It exports no unbounded stopping branch, no limiting stopping theorem, no almost-sure quotient, and no completed-filtration carrier.

Proof. Start with Definition 7038.2. It forces any consumer to display $T, b, F, D, V, H, C, P, N$ before the local name is consumed.

The optional-stopping side reads only the handoff row V after the finite route has been displayed. The row V is tied to a branch in D, to the finite bound b, and to the visible filtration windows F over the source stopping-time row T.

The martingale, probability, and random-variable rows are therefore consumed only as the finite source data named by P and replayed through C. Because the carrier inventory excludes unbounded stopping branches, limiting stopping theorems, almost-sure quotient equality, and completed filtrations, the handoff supplies exactly the bounded-window interface required by $\mathsf{OptionalStopping}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{BoundedStoppingTime}^{\uparrow}$ consumer displays a $\mathsf{StoppingTime}^{\uparrow}$ source row, a finite bound, filtration-window visibility, a bounded decision ledger, stopped-value handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any $\mathsf{OptionalStopping}^{\uparrow}$ stopped-value read is consumed.

Independence witness: stoppingtime, optionalstopping, martingale, probspace, randomvar: $\mathsf{BoundedStoppingTime}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source stopping decision row, finite bound, filtration-window visibility, bounded decision ledger, stopped-value handoff, transport, replay, provenance, and local NameCert rows as one finite interface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedStoppingTime}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedStoppingTime}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7038.1, Definition 7038.2, and Definition 7038.3 seed the finite bounded-window stopping interface over $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{RandomVar}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedStoppingTime}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, unbounded stopping, optional-stopping expectation equality, almost-sure quotient equality, uniform integrability, completed filtrations, measure completion, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-bound exposure, filtration-window visibility, decision-ledger exactness, stopped-value handoff, transport replay, provenance, and non-escape for BEDC.Derived.BoundedStoppingTimeU

Constructive story: A $\mathsf{BoundedStoppingTime}^{\uparrow}$ packet is generated from a source stopping-time row, a displayed finite bound, filtration-window visibility rows, a bounded decision ledger, a stopped-value handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7039

A Concrete Naming Certificate for $\mathsf{DenseSubspaceCompletion}^{\uparrow}$

Following Bourbaki and Engelking on completion from a dense subspace, the $\mathsf{DenseSubspaceCompletion}^{\uparrow}$ packet records the finite BEDC handoff by which a dense subspace carrier supplies completion-facing names without replacing them by quotient representatives or host equality. It sits beside the metric source of chapter 57, the Cauchy-filter completion route of chapter 1569, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dense embedding boundary of chapter 1512, and the terminal Real seal of chapter 38. The packet names a dense-subspace completion handoff as finite rows; it is not a selected completion point, not an ambient complete metric space, and not a detached Real-completeness principle.

Definition 7039.1 (Dense-subspace completion carrier)**.** *A* $\mathsf{DenseSubspaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, M, S, F, W, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row whose distance and classifier data are read by the completion request.*
2. *Y is the dense subspace row, including the displayed inclusion into X, the dense witness ledger, and the subspace classifier transport.*
3. *M is the metric restriction row on Y, read through the same metric witnesses as X.*
4. *S is the subspace approximation row: each ambient tolerance request is met by a displayed subspace witness.*
5. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *route generated from the dense subspace data.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger used to observe finite completion-facing reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and metric tolerances.*
8. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that embeds subspace observations into the completion boundary.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, F, W, R, E have been displayed.*

10. *H records componentwise* hsame *transport for the ambient metric, dense subspace, restricted metric, approximation, filter completion, windows, readbacks, embedding, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from an ambient completion request through the dense subspace row, Cauchy-filter route, stream windows, regular readback, completion embedding, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{DenseSubspaceCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseSubspaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, Y, M, S, F, W, R, E, A, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host metric equality, countable choice, or an ambient completeness theorem.

Theorem 7039.2 (Dense-subspace completion NameCert obligations)**.** *For every accepted* DenseSubspaceCompleti *carrier $D = (X, Y, M, S, F, W, R, E, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseSubspaceCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed ambient metric, dense subspace, restricted metric, approximation ledger, Cauchy-filter completion route, finite stream windows, regular rational readbacks, completion embedding, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. No NameCert obligation is supplied by quotient completions, selected representatives, host equality, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 7039.1. The rows $X, Y, M, S, F, W, R, E, A, H, C, P, N$ are exactly the ambient metric, dense subspace, restricted metric, approximation, Cauchy-filter, window, readback, embedding, Real-seal, transport, replay, provenance, and local naming coordinates.

The dense-subspace part is Y, M, S. It reads the metric row X, restricts that metric to the displayed subspace, and records the finite approximation ledger by which ambient tolerance requests are met inside the subspace.

The completion-facing part is F, W, R, E, A. The Cauchy-filter completion route consumes the dense-subspace data, the stream windows and regular readbacks expose finite observations, the embedding row inserts the subspace observations into the completion boundary, and the Real seal is reached only after those rows are present.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient completion, selected representatives, host equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7039.3 (Dense-subspace completion dense handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{DenseSubspaceCompletion}^{\uparrow}$ *packet factors through the finite handoff*

$$X, Y, M, S \longmapsto F, W, R \longmapsto E \longmapsto A \longmapsto H, C, P, N.$$

Thus the consumer reads the dense subspace row, metric restriction, Cauchy-filter completion row, finite windows, regular rational readbacks, $\mathsf{CompletionEmbedding}^{\uparrow}$ *boundary,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before accepting the completed result.*

Proof. Begin with an accepted completion-facing request. Carrier admission exposes the ambient metric X and the dense subspace row Y before any completion row is available.

The restricted metric and approximation ledger M, S then produce the dense handoff. Each ambient tolerance read must be witnessed by a subspace approximation before it can enter a Cauchy-filter completion route.

Read the completion rows next. The row F is the Cauchy-filter completion route, W supplies finite stream observations, R supplies regular rational readback, and E is the dense completion embedding boundary.

The Real seal A is downstream of that route, while H, C, P, N transport, replay, source, and name it. Hence every dense-subspace completion consumer receives the displayed finite handoff and cannot extract quotient equality, selected representatives, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{DenseSubspaceCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{CompletionEmbedding}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row without first reading the dense subspace row, restricted metric row, approximation ledger, $\mathsf{CauchyFilterCompletion}^{\uparrow}$ route, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: metric, cauchyfiltercompletion, streamname, regseqrat, completionembedding, real: no listed sibling supplies the whole packet $D = (X, Y, M, S, F, W, R, E, A, H, C, P, N)$, because each sibling contributes only a metric source, Cauchy-filter route, finite window, regular readback, dense embedding, or Real seal while this chapter binds the dense-subspace handoff with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseSubspaceCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseSubspaceCompletion}^{\uparrow}$ packet is generated from an ambient MetricSpace row, dense subspace row, restricted metric row, approximation ledger, CauchyFilterCompletion route, StreamName windows, RegSeqRat readbacks, CompletionEmbedding row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseSubspaceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7039.1, Definition 7039.2, and Definition 7039.3 seed the dense-subspace completion handoff over metric, Cauchy-filter, stream, regular-readback, embedding, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseSubspaceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completion representatives, countable choice, host metric equality, Lean verification, audit cleanliness, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, dense approximation, Cauchy-filter completion replay, embedding handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DenseSubspaceCompletion

Chapter 7040

A Concrete Naming Certificate for $\mathsf{CauchyTowerCollapse}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{CauchyTowerCollapse}^{\uparrow}$ packet records the finite flattening route by which iterated regular-Cauchy completion data is read as one Real-facing certificate surface. It sits over the dyadic rational carrier of chapter 291, the finite stream-window layer of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the route comparison packet of chapter 1635, and the common-tail uniqueness boundary of chapter 1782. The chapter names the finite collapse window for towers of regular-Cauchy completion rows; it does not assert quotient completion equality, selected host limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7040.1 (Cauchy tower collapse carrier)**.** *A* $\mathsf{CauchyTowerCollapse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, R, E, A, U, W, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the base* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including half-radius and fused-radius cells used by every tower level.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that chooses which observations of each tower level are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same displayed windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dyadic, stream, and regular-readback rows have been displayed.*
5. *A is the* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ *route-comparison row that keeps nested completion flattening orders on the same finite ledger.*
6. *U is the* $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$ *common-tail row comparing the flattened route with the single displayed completion boundary.*
7. *W is the finite tower-window projection row, recording the observed depth, precision request, and component read at each inspected level.*
8. *F is the dyadic tolerance-fusion row, combining the displayed level tolerances into one requested* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound before the Real seal is consumed.*

9. *H records componentwise* hsame *transport for the dyadic, stream, readback, Real-seal, associativity, uniqueness, projection, fusion, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an iterated completion request through W, F, A, U, and then through B, S, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyTowerCollapse}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTowerCollapse}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, S, R, E, A, U, W, F, H, C, P, N$. No coordinate stores quotient completion equality, a selected host limit, countable-choice schedules, or ambient real completeness.

Theorem 7040.2 (Cauchy tower collapse NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTowerCollapse}^{\uparrow}$ *carrier $T = (B, S, R, E, A, U, W, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTowerCollapse}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite stream windows, regular rational readback, Real seal, associativity row, uniqueness row, finite tower-window projection, dyadic tolerance fusion, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. No NameCert obligation is supplied by quotient completion equality, selected host limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7040.1. The rows $B, S, R, E, A, U, W, F, H, C, P, N$ are exactly the dyadic tolerance, stream-window, regular-readback, Real-seal, associativity, uniqueness, projection, fusion, transport, replay, provenance, and local naming coordinates.

The tower-facing part is W, A, U. The projection row W exposes only the inspected finite depth and precision requests, the associativity row A compares the displayed flattening orders, and the uniqueness row U identifies the common-tail boundary available after flattening.

The Real-facing part is B, S, R, E, F. Tolerance requests are fused by F into the dyadic ledger B, the finite observations are read through S, the rational schedules are read through R, and the terminal seal E is consumed only after those rows are present.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient completion equality, selected host limits, countable choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 7040.3 (Cauchy tower finite-window flattening)**.** *Let $T = (B, S, R, E, A, U, W, F, H, C, P, N)$ be an accepted* $\mathsf{CauchyTowerCollapse}^{\uparrow}$ *carrier. Every iterated regular-Cauchy completion read factors through the finite route*

$$W \longmapsto F \longmapsto B, S, R \longmapsto A, U \longmapsto E \longmapsto H, C, P, N.$$

Thus an iterated completion tower is flattened only by finite-window projection and dyadic tolerance fusion before reaching the $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient completion equality, selected host limit, countable-choice schedule, or ambient Real completeness principle.*

Proof. Begin with the finite tower-window projection W. It chooses the displayed tower depth, the inspected components, and the requested precision cells without selecting a completed host object.

Fuse tolerances next. The row F combines the displayed level tolerances into one $\mathsf{DyadicRatCore}^{\uparrow}$ request in B, after which the same observation windows S and regular rational readbacks R serve the flattened route.

Read the completion comparison rows. The associativity row A keeps the left- and right-nested collapse orders on one finite ledger, while U supplies the regular-Cauchy common-tail boundary that permits the single Real-facing seal to be read.

The terminal seal E and structural rows H, C, P, N preserve exactly that route. Because the packet contains no quotient equality, selected limit, countable-choice, or ambient-completeness coordinate, no such principle can be exported by the flattening consumer. □

Falsifiable prediction: Every accepted $\mathsf{CauchyTowerCollapse}^{\uparrow}$ consumer that reaches the $\mathsf{Real}^{\uparrow}$ seal must first read the finite tower-window projection, dyadic tolerance-fusion row, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ route row, and $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$ common-tail row.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-completion-associativity, regular-cauchy-completion-uniqueness: no listed sibling supplies the whole packet $T = (B, S, R, E, A, U, W, F, H, C,$ because each sibling contributes only a tolerance ledger, finite window, regular readback, Real seal, route comparison, or common-tail boundary while this chapter binds the finite tower projection and dyadic fusion route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTowerCollapse}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTowerCollapse}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, CauchyCompletionAssociativity and RegularCauchyCompletionUniqueness rows, finite tower-window projection, dyadic tolerance fusion, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTowerCollapse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7040.1, Definition 7040.2, and Definition 7040.3 seed the finite collapse surface for iterated regular-Cauchy completion windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTowerCollapse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected host limits, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite tower-window projection, dyadic tolerance fusion, completion-route comparison, regular-Cauchy common-tail exposure, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTowerCollapseUp.

Chapter 7041

A Concrete Naming Certificate for $\mathsf{QuasisymmetricMap}^{\uparrow}$

Following Heinonen on analysis on metric spaces and Tukia–Vaisala on quasisymmetric embeddings, the $\mathsf{QuasisymmetricMap}^{\uparrow}$ packet records the finite BEDC surface of a metric map whose distortion is controlled by a displayed modulus row. It sits over the source and target metric packets of chapter 57, the terminal distance seal of chapter 38, the entourage-facing route of chapter 339, the map graph of chapter 130, and the control ledger of chapter 1631. The packet names a quasisymmetric handoff as finite rows; it is not a quotient map equality, not a selected inverse, not an ambient completeness principle, and not a compactness theorem.

Definition 7041.1 (Quasisymmetric map carrier)**.** *A* $\mathsf{QuasisymmetricMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, G, \eta, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier and distance-read schedule used by the inspected triple.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including the target classifier and its distance-read schedule.*
3. *G is the finite map-graph row, read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *sibling surface when a topology-facing consumer asks for the same graph.*
4. *η is the distortion-control ledger, read through the* $\mathsf{ModulusOfContinuity}^{\uparrow}$ *sibling surface as a finite monotone comparison row.*
5. *D is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row for source and target pair distances used in the quasisymmetric comparison.*
6. *T is the uniform-topology transport row that records how the metric comparison is replayed by* $\mathsf{UniformSpace}^{\uparrow}$ *consumers.*
7. *R is the replay row exposing the finite triple comparison and the target-distance bound determined by η.*
8. *H records componentwise* hsame *transport for the source metric, target metric, graph, distortion ledger, Real distance seals, uniform transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{QuasisymmetricMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ModulusOfContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasisymmetricMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, Y, G, \eta, D, T, R, H, C, P, N$. It has no coordinate for quotient map equality, a selected inverse map, ambient completeness, compactness, or an unbounded host metric theorem.

Theorem 7041.2 (Quasisymmetric map NameCert obligations)**.** *For every accepted* $\mathsf{QuasisymmetricMap}^{\uparrow}$ *carrier $Q = (X, Y, G, \eta, D, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuasisymmetricMap}^{\uparrow}}$ *surface consists exactly of source and target metric admission, map-graph exposure, distortion-control ledger admission,* $\mathsf{Real}^{\uparrow}$ *distance sealing, uniform-topology transport, finite comparison replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is supplied by quotient map equality, selected inverses, ambient completeness, open-cover compactness, or an external host topology.*

Proof. Project the carrier of Definition 7041.1. The rows $X, Y, G, \eta, D, T, R, H, C, P, N$ are exactly the source metric, target metric, graph, distortion ledger, distance seal, uniform transport, replay, transport, continuation, provenance, and local naming coordinates.

The metric part is X, Y, D. It exposes the two distance schedules and seals every source and target distance comparison through the displayed $\mathsf{Real}^{\uparrow}$ row before the distortion ledger is read.

The mapping part is G, η, T, R. The graph row supplies the finite map observations, the distortion ledger supplies the comparison budget, the uniform transport row makes the same route visible to topology-facing consumers, and the replay row records the inspected triple comparison.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores quotient map equality, selected inverse data, compactness, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7041.3 (Quasisymmetric map metric handoff)**.** *Every metric, uniform, or continuous-map consumer of an accepted* $\mathsf{QuasisymmetricMap}^{\uparrow}$ *packet factors its request through the finite handoff*

$$X, Y, G, \eta \longmapsto D \longmapsto T, R \longmapsto H, C, P, N.$$

Thus the consumer reads the source and target metric rows, map graph, distortion-control ledger, $\mathsf{Real}^{\uparrow}$ *distance seals, uniform transport, replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before accepting a quasisymmetric comparison.*

Proof. Begin with an accepted consumer request. Carrier admission exposes the source metric row X, target metric row Y, graph row G, and distortion ledger η before any quasisymmetric comparison is visible.

Read the distance row next. The $\mathsf{Real}^{\uparrow}$ seal D binds the source and target distances used by the inspected triple, so the comparison cannot be detached from the displayed real-distance evidence.

The uniform transport and replay rows T, R then hand the same finite comparison to metric, uniform, and continuous-map consumers. They expose the distortion-controlled target-distance bound without adding a quotient equality, inverse selection, compactness, or completeness coordinate.

The rows H, C, P, N preserve, replay, source, and name this handoff. Hence every accepted quasisymmetric read factors through the displayed finite route and none of the excluded host principles can be exported. □

Falsifiable prediction: An accepted $\mathsf{QuasisymmetricMap}^{\uparrow}$ consumer that reads a target-distance distortion bound without first displaying the source metric row, target metric row, map graph, distortion-control ledger, $\mathsf{Real}^{\uparrow}$ distance seal, uniform transport, replay, transport, provenance, and local naming rows refutes the carrier surface.

Independence witness: metric, real, uniform, continuousmap, moduluscontinuity: no listed sibling supplies the whole packet $Q = (X, Y, G, \eta, D, T, R, H, C, P, N)$, because each sibling contributes only a metric source, real-distance seal, uniform route, map graph, or control ledger while this chapter binds the quasisymmetric comparison with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasisymmetricMap}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasisymmetricMap}^{\uparrow}$ packet is generated from source and target MetricSpace rows, a finite map graph, a distortion-control ledger, RealUp distance seals, uniform transport, finite comparison replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasisymmetricMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7041.1, Definition 7041.2, and Definition 7041.3 seed the quasisymmetric metric-topology handoff over metric, real-distance, uniform, continuous-map, and modulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasisymmetricMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient map equality, selected inverses, ambient completeness, compactness, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distortion-ledger routing, Real distance sealing, uniform handoff, transport, replay, provenance, and local naming for BEDC.Derived.QuasisymmetricMapUp.

Chapter 7042

A Concrete Naming Certificate for $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$

The $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$ packet realizes a finite residual-pair witness surface from the confluence questions of chapter 9386. It is narrower than the parallel-diamond frontier of chapter 4238: the witness records one displayed local peak, its residual-substitution rows, its candidate join rows, and the bounded checker budget before any local-diamond read is public. It consumes the critical-path packet of chapter 765, the normalization router of chapter 1108, the closed substitution row of chapter 537, the existing frontier of chapter 4238, and the terminal real seal of chapter 38. The packet is a local finite witness, not a global confluence theorem, not a normalization theorem, and not a hidden substitution principle.

Definition 7042.1 (MetaCIC parallel-diamond witness carrier)**.** *A* $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, L, R, S, J, B, A, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed MetaCIC window row, exposing the closed source term, typing context, redex shape, and bounded binder depth inspected by the witness.*
2. *L and R are the two one-step residual rows generated from the same displayed peak.*
3. *S is the closed-term substitution row, read through the* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *surface before either residual row is joined.*
4. *J is the candidate local-join row, recording the displayed common residual target proposed for the pair.*
5. *B is the bounded checker row that verifies the two residual routes inside the shown finite budget.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *audit seal for the numeric fuel, height, and readback bounds used by the checker.*
7. *O is the obstruction-retention row, keeping the residual-substitution premise and subject-reduction boundary visible.*
8. *H records componentwise* hsame *transport for the typed window, residual pair, substitution row, local join, checker, real audit seal, obstruction row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same local witness route.*

10. *P is the* Pkg *provenance over* MetaCICParallelDiamondWitness$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationCer MetaCICParallelDiamondFrontier$^{\uparrow}$, ClosedTermSubstitution$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, L, R, S, J, B, A, O, H, C, P, N$. It has no coordinate for arbitrary closed-term Church–Rosser, global confluence, normalization for all closed terms, unrestricted substitution compatibility, quotient syntax equality, or a hidden evaluator oracle.

Theorem 7042.2 (MetaCIC parallel-diamond witness obligations)**.** *For every accepted* MetaCICParallelDiamondWi *carrier $W = (T, L, R, S, J, B, A, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}}$ *surface consists exactly of typed-window exposure, two residual rows from the same peak, closed-term substitution exposure, candidate local-join admission, bounded checker accounting,* Real$^{\uparrow}$ *audit sealing, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is supplied by global confluence, normalization, unrestricted substitution compatibility, quotient syntax equality, host evaluator soundness, or kernel consistency.*

Proof. Project the carrier of Definition 7042.1. The rows $T, L, R, S, J, B, A, O, H, C, P, N$ are the only visible coordinates of an accepted witness packet.

The local peak is held by T, L, R. Both residual rows are tied to the same typed window, so a consumer cannot splice in a detached critical pair or an untyped residual route.

The join and checking data are S, J, B, A, O. The closed substitution row remains visible before the candidate join is read, the bounded checker records the finite verification budget, the real audit seal accounts for numeric bounds, and the obstruction row preserves the residual-substitution premise.

Finally H, C, P, N transport, replay, source, and name the same local route. Since none of the excluded global or host principles occurs as a coordinate, the listed rows exhaust the witness NameCert surface. □

Theorem 7042.3 (MetaCIC parallel-diamond witness handoff)**.** *Every confluence or normalization-router consumer of an accepted* MetaCICParallelDiamondWitness$^{\uparrow}$ *packet factors through the finite handoff*

$$T, L, R \longmapsto S, J, B, A, O \longmapsto H, C, P, N.$$

The handoff supplies one residual-pair local-diamond witness to the critical-path and frontier readers. It does not discharge the parallel-diamond frontier, close arbitrary confluence, prove normalization, or erase the closed-substitution boundary.

Proof. Begin with an accepted consumer request. Carrier admission exposes the typed window T and the two residual rows L, R before a local-diamond read is available.

Read the middle rows next. The closed substitution row S binds the residual routes, the candidate join row J records the proposed common target, the bounded checker B checks the displayed routes, the real audit seal A records the finite budget, and the obstruction row O keeps the unclosed premises visible.

The structural rows H, C, P, N then carry the witness to the critical-path and frontier consumers. They preserve the local route without converting it into a global confluence theorem, a normalization theorem, or an unrestricted substitution result. □

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondWitnessUp.MetaCICParallelDiamondWitnessHandoff`

Falsifiable prediction: An accepted $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$ consumer that cites metaciccriticalpath, metacicnormalizationcertificaterouter, metacic-parallel-diamond-frontier, closedtermsubstitution, or real without displaying $T, L, R, S, J, B, A, O, H, C, P, N$ refutes the witness carrier surface.

Independence witness: metaciccriticalpath, metacicnormalizationcertificaterouter, metacic-parallel-diamond-frontier, closedtermsubstitution, real: no listed sibling is bijective to $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$ because this packet jointly carries one typed local peak, two residual rows, closed substitution exposure, a candidate join, bounded checker budget, Real audit seal, obstruction retention, transport, replay, provenance, and local naming.

The Lean follow-up surface is `lean4/BEDC/Derived/MetaCICParallelDiamondWitnessUp/TasteGate.lean`; it must provide `Nontrivial MetaCICParallelDiamondWitnessUp` and `FieldFaithful MetaCICParallelDiamond` instances for the same carrier rows before any TasteGate claim is recorded.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}$ packet is generated from a typed MetaCIC window, two residual rows, a closed-term substitution row, a candidate local join, a bounded checker, a RealUp audit seal, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7042.1, Definition 7042.2, and Definition 7042.3 seed the finite residual-pair local-diamond witness consumed by MetaCIC confluence and normalization-router readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global confluence, arbitrary closed-term Church–Rosser, normalization for all closed terms, unrestricted substitution compatibility, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-pair routing, closed-substitution exposure, local-join accounting, bounded-checker accounting, Real audit sealing, obstruction retention, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICParallelDiamondWitnessU

Chapter 7043

A Concrete Naming Certificate for $\mathsf{MetrizableUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces induced by metrics, the $\mathsf{MetrizableUniformity}^{\uparrow}$ packet records the finite certificate by which a displayed $\mathsf{Met}^{\uparrow}$ row supplies the entourage basis of a $\mathsf{UniformSpace}^{\uparrow}$ row. It sits between the metric certificate of chapter 57, the uniform-space discipline of chapter 339, the entourage-basis surface of $\mathsf{EntourageBase}^{\uparrow}$, and the L10 tolerance route through chapter 291, chapter 126, chapter 301, and chapter 38. The packet names the metric-to-uniformity bridge as finite NameCert data; it does not assert an ambient topology theorem, quotient completion equality, selected radii beyond the displayed dyadic rows, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7043.1 (Metrizable uniformity carrier)**.** *A* $\mathsf{MetrizableUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, U, E, B, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the named distance comparison and triangle rows used by all later requests.*
2. *U is the target* $\mathsf{UniformSpace}^{\uparrow}$ *row whose entourages are read only through the displayed metric basis.*
3. *E is the* $\mathsf{EntourageBase}^{\uparrow}$ *row listing the metric-ball entourage predicates E_ε for the finite dyadic tolerances requested by the consumer.*
4. *B is the basis-refinement ledger: for each displayed tolerance request it records a smaller dyadic tolerance whose metric ball refines the requested entourage row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying positivity, halving, comparison, and addition data for the radii used by E and B.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which endpoint and tolerance observations are read before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
7. *H records componentwise* hsame *transport for metric, uniformity, entourage, dyadic, window, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniformity request through the metric-ball basis and the dyadic refinement ledger.*

9. *P is the* Pkg *provenance over* $\mathsf{MetrizableUniformity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetrizableUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, E, B, D, W, H, C, P, N$. It has no coordinate for host topological equality, quotient completion equality, a selected infinite radius schedule, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7043.2 (Metrizable uniformity entourage basis)**.** *For every accepted* $\mathsf{MetrizableUniformity}^{\uparrow}$ *carrier*

$$K = (M, U, E, B, D, W, H, C, P, N),$$

the metric balls displayed by E generate the exported $\mathsf{UniformSpace}^{\uparrow}$ *entourage basis exactly through the dyadic tolerance rows in D and the refinement ledger B. A consumer that asks for a uniform entourage must first read the* $\mathsf{Met}^{\uparrow}$ *source row, choose one of the displayed dyadic tolerance rows, refine it through B, and replay the request through H, C, P, N before the* $\mathsf{UniformSpace}^{\uparrow}$ *row is exposed.*

Proof. Project the carrier of Definition 7043.1. The only analytic source row is M, and the only target uniformity row is U; the route between them is the displayed entourage basis E with dyadic bookkeeping D and refinement ledger B.

Read a requested entourage through D and B. Positivity and halving in $\mathsf{DyadicRatCore}^{\uparrow}$ supply the smaller tolerance, while the metric row supplies the ball relation used by E. Thus every displayed basis element is a metric-ball entourage and every requested basis refinement is recorded inside the packet.

The structural rows H, C, P, N then transport, replay, source, and name the same route. Since no host topology theorem, quotient completion equality, selected infinite schedule, countable-choice selector, or ambient completeness coordinate occurs in K, the exported uniformity basis is exactly the finite metric-ball basis named by the carrier. □

Theorem 7043.3 (Metrizable uniformity NameCert obligations)**.** *For the carrier of Definition 7043.1, the local* $\mathsf{NameCert}_{\mathsf{MetrizableUniformity}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source admission, uniform-target admission, entourage-basis admission, dyadic basis refinement, symmetry and triangle refinement through the displayed metric rows, finite-window routing through* $\mathsf{StreamName}^{\uparrow}$, *regular readback through* $\mathsf{RegSeqRat}^{\uparrow}$, *terminal sealing through* $\mathsf{Real}^{\uparrow}$, *componentwise transport, displayed replay, provenance, and local naming. The obligation surface exports no host uniformization theorem, quotient completion equality, selected representative, countable-choice selector, or ambient Real completeness principle.*

Proof. Carrier admission is tuple projection from Definition 7043.1. The metric and uniformity rows are M and U, while E, B, D provide the only basis and tolerance evidence.

Symmetry and triangle refinement are inherited only through the displayed metric rows of M. The dyadic ledger D supplies the smaller positive radii needed for refinement, and B records the exact basis step consumed by the uniformity request.

Finally W, H, C, P, N route finite observations, componentwise transport, replay, provenance, and local naming through the same packet, with $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reached only along the L10 readback seal. The excluded host objects are absent from the carrier, so the listed families exhaust the NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{MetrizableUniformity}^{\uparrow}$ packet exposes a uniform entourage only after a displayed $\mathsf{Met}^{\uparrow}$ row, a dyadic tolerance row, an $\mathsf{EntourageBase}^{\uparrow}$ metric-ball row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming have been read.

Independence witness: metricspace, uniformspace, entouragebase, dyadicratcore, streamname, regseqrat, real: $\mathsf{MetrizableUniformity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the metric source, uniform target, metric-ball entourage basis, dyadic refinement ledger, finite window, transport, replay, provenance, and local NameCert rows as one finite bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetrizableUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetrizableUniformity}^{\uparrow}$ packet is generated from a MetricSpace row, a UniformSpace target row, an EntourageBase metric-ball basis, DyadicRatCore tolerance refinement, finite StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetrizableUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7043.1, Definition 7043.2, and Definition 7043.3 seed the finite metric-to-uniformity route over MetricSpace, UniformSpace, EntourageBase, DyadicRatCore, StreamName, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetrizableUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topological equality, quotient completion equality, selected radius schedules, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, metric-source admission, uniform-target admission, entourage-basis refinement, symmetry and triangle refinement through metric rows, transport, replay, provenance, and local naming for BEDC.Derived.MetrizableUniformityUp.

Chapter 7044

A Concrete Naming Certificate for $\mathsf{CousinFiniteCover}^{\uparrow}$

Following Cousin and Bishop-style gauge integration, the $\mathsf{CousinFiniteCover}^{\uparrow}$ packet records the finite subcover extraction surface between the local gauge cover of chapter 3426 and the tagged-subdivision output of chapter 1861. It reuses the positive lower-bound control of chapter 452, the bisection rows of chapter 1711, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names finite extraction data only; it is not a synonym for the cover chapter, the lemma chapter, open-cover compactness, selected subcover choice, or host real completeness.

Definition 7044.1 (Cousin finite-cover carrier)**.** *A* $\mathsf{CousinFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, G, C, T, M, W, R, E, H, P, N)$$

with the following displayed rows:

1. *I is the located closed interval row.*
2. *G is the positive gauge row over I.*
3. *C is the finite gauge-cover row read from* $\mathsf{CousinCover}^{\uparrow}$*.*
4. *T is the finite tagged-subinterval extraction row consumed by* $\mathsf{CousinLemma}^{\uparrow}$*.*
5. *M is the mesh and adjacency ledger for the extracted finite cover cells.*
6. *W is the located bisection witness row, recording the finite search tree used when a displayed cell is refined.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational radii and mesh bounds.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint and radius comparisons.*
9. *H records componentwise* hsame *transport for interval, gauge, cover, tagged-cell, mesh, bisection, readback, Real-seal, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CousinFiniteCover}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinFiniteCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, C, T, M, W, R, E, H, P, N$. It contains no open-cover compactness theorem, selected subcover oracle, quotient cover equality, ambient completeness assertion, or evidence-erased passage from local gauge data to a tagged partition.

Lean4 ✓ `BEDC.Derived.CousinFiniteCoverUp.CousinFiniteCoverTasteGate single carrier alignment`

Theorem 7044.2 (Cousin finite-cover NameCert obligations)**.** *For every accepted* $\mathsf{CousinFiniteCover}^{\uparrow}$ *carrier*

$$K = (I, G, C, T, M, W, R, E, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CousinFiniteCover}^{\uparrow}}$ *surface consists exactly of located interval admission, positive gauge admission, finite* $\mathsf{CousinCover}^{\uparrow}$ *cover exposure, finite tagged-cell extraction toward* $\mathsf{CousinLemma}^{\uparrow}$*, mesh and adjacency exactness, located bisection support,* $\mathsf{RegSeqRat}^{\uparrow}$ *radius readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay through the named rows,* Pkg *provenance, and local naming. No obligation is discharged by open-cover compactness, hidden subcover selection, quotient cover equality, host real completeness, or a direct lemma-level subdivision endpoint.*

Proof. Project the carrier of Definition 7044.1. The only visible analytic rows are I, G, C, T, M, W, R, E, followed by the structural rows H, P, N and displayed replay through the same packet.

The cover row C supplies the finite local cover data, while the extraction row T records the finite tagged cells that can be handed to $\mathsf{CousinLemma}^{\uparrow}$. The mesh ledger M and bisection row W keep the extraction finite and interval-local.

The readback and seal rows R, E attach rational radius comparisons and endpoint comparisons to the same packet. Transport, replay, provenance, and local naming then preserve the row order. Since the carrier has no coordinate for compactness, selected subcovers, quotient cover equality, host completeness, or a direct subdivision endpoint, the listed rows exhaust the NameCert surface. □

Theorem 7044.3 (Cousin finite-cover gauge-refinement route)**.** *Let* $K = (I, G, C, T, M, W, R, E, H, P, N)$ *be an accepted* $\mathsf{CousinFiniteCover}^{\uparrow}$ *carrier. Any consumer that refines a gauge cover toward a Cousin subdivision must expose the* $\mathsf{CousinCover}^{\uparrow}$ *cover row C, the finite bisection row W, the mesh and adjacency ledger M, the rational readback row R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the tagged extraction row T before it may cite the* $\mathsf{CousinLemma}^{\uparrow}$ *handoff. The route exports a finite extraction packet and cannot replace it by open-cover compactness, an arbitrary dense real argument, host subcover choice, quotient cover equality, or ambient completeness.*

Proof. Start with the finite cover row C and the displayed gauge row G. These rows identify the local cover cells over the located interval I, but they do not by themselves export a tagged partition.

Use the bisection row W together with the mesh ledger M. The finite bisection search refines failed cells, records adjacency and mesh comparisons, and keeps each accepted cell attached to the same gauge row.

The rational readback R and Real seal E verify the radius and endpoint comparisons used by the refinement. The tagged extraction row T is therefore a finite handoff row rather than a selected cover family.

Finally replay the packet through H, P, N. Componentwise transport and provenance keep the consumer on the same I, G, C, T, M, W, R, E rows, so no compactness theorem, dense real argument, host selector, quotient cover equality, or ambient completeness coordinate is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinFiniteCover}^{\uparrow}}$.

Constructive story: A $\mathsf{CousinFiniteCover}^{\uparrow}$ packet is generated from a located interval row, a positive gauge row, a CousinCover finite cover row, a tagged extraction row, mesh and bisection ledgers, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7044.1, Definition 7044.2, and Definition 7044.3 seed the finite subcover extraction packet between CousinCoverUp and CousinLemmaUp over FiniteLebesgueNumberUp, LocatedIntervalBisectionUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CousinFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, hidden subcover selection, quotient cover equality, host real completeness, full CousinLemma verification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover exposure, finite extraction exactness, bisection support, mesh and radius readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CousinFiniteCoverU

Chapter 7045

A Concrete Naming Certificate for $\mathsf{PowerSeries}^{\uparrow}$

Following Hardy and Knopp on power series over the reals, the $\mathsf{PowerSeries}^{\uparrow}$ packet names the finite route by which coefficient data, a displayed radius, and Cauchy-controlled partial sums reach a real-facing value. It anchors the route in the real-name seal of chapter 38, the regular rational readback of chapter 301, the uniform tail discipline of chapter 1736, the criterion surface of chapter 2015, and the product-radius budget of **??**. The packet records only the displayed coefficient, radius, partial-sum, Cauchy-control, transport, replay, provenance, and local naming rows; it does not import a host analytic function, a selected infinite sum, quotient function equality, or ambient completeness.

Definition 7045.1 (Power-series NameCert carrier)**.** *A* $\mathsf{PowerSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, R, S, C, E, H, T, P, N)$$

with the following displayed rows:

1. *A is the coefficient stream row, exposing coefficients only through finite windows with their* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
2. *R is the radius ledger row, recording the displayed positive radius or local subradius used by the evaluation request;*
3. *S is the partial-sum stream row, naming the finite polynomial sums associated to the selected coefficient windows and argument row;*
4. *C is the Cauchy estimate row, giving the tail-control route used to accept the partial sums as a Cauchy stream;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the accepted partial-sum stream;*
6. *H records componentwise* hsame *transport for coefficient, radius, partial-sum, Cauchy-control, real-seal, provenance, and local naming rows;*
7. *T records the displayed* Cont *replay by which a consumer reuses the same radius and partial-sum route;*
8. *P is the* Pkg *provenance over* $\mathsf{PowerSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchySeriesCriterion}^{\uparrow}$, $\mathsf{CauchyProductModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{PowerSeries}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these nine rows. It has no coordinate for a host convergence theorem, analytic continuation, quotient function space, selected infinite sum, or ambient $\mathsf{Real}^\uparrow$ *completeness proof.*

Theorem 7045.2 (Power-series NameCert obligations)**.** *For an accepted* $\mathsf{PowerSeries}^\uparrow$ *carrier* $K = (A, R, S, C, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PowerSeries}^\uparrow}$ *surface consists exactly of carrier admission for the coefficient stream, displayed radius ledger, partial-sum stream, Cauchy estimate, terminal* $\mathsf{Real}^\uparrow$ *seal, componentwise transport, continuation replay, provenance, and local naming rows; exactness of the radius row used by the evaluation request; Cauchy control of the displayed partial-sum stream; handoff of the accepted stream to the* $\mathsf{Real}^\uparrow$ *seal; stability under* hsame *transport; replay through the same* Cont *route; and provenance over the named RegSeqRat, uniform Cauchy series, Cauchy-series criterion, and Cauchy-product modulus dependencies.*

Theorem 7045.3 (Power-series Cauchy-radius handoff)**.** *Every real-facing read from an accepted* $\mathsf{PowerSeries}^\uparrow$ *carrier* $K = (A, R, S, C, E, H, T, P, N)$ *factors through the displayed radius row* R *and the partial-sum Cauchy route* S, C *before the terminal* $\mathsf{Real}^\uparrow$ *seal* E *may be consumed. Thus the read is a finite BEDC route through* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{UniformCauchySeries}^\uparrow$*,* $\mathsf{CauchySeriesCriterion}^\uparrow$*,* $\mathsf{CauchyProductModulus}^\uparrow$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a host power-series evaluation, selected infinite sum, analytic continuation, quotient function value, or ambient completeness theorem.*

Closure status for $\mathsf{NameCert}_{\mathsf{PowerSeries}^\uparrow}$**.**

Constructive story: A $\mathsf{PowerSeries}^\uparrow$ packet is generated from a coefficient stream, displayed radius ledger, partial-sum stream, Cauchy estimate, terminal Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PowerSeries}^\uparrow}$, seed)

Scope closed: Definition 7045.1, Definition 7045.2, and Definition 7045.3 seed the real-facing power-series route over RegSeqRat, uniform Cauchy series, Cauchy-series criterion, Cauchy-product modulus, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PowerSeries}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a standard bridge, host analytic continuation, selected infinite sums, quotient function equality, or ambient completeness.

Upgrade path: Obligation closure requires a Lean carrier and obligation package for the same coefficient, radius, partial-sum, Cauchy-estimate, Real-seal, transport, replay, provenance, and local NameCert route.

Chapter 7046

A Concrete Naming Certificate for $\mathsf{MengerCompactum}^{\uparrow}$

Following Menger's universal curve and compactum tradition, and the compact-metric dimension treatment in Engelking, the $\mathsf{MengerCompactum}^{\uparrow}$ packet records a finite compact-metric and covering-dimension naming surface. It sits after the metric carrier of chapter 57, the compact metric source of chapter 129, the covering-dimension row of chapter 1853, the finite metric entropy budget of $\mathsf{FiniteMetricEntropy}^{\uparrow}$, and the terminal real-name boundary of chapter 38. The packet names a displayed finite-cover compactum carrier for downstream compact-metric consumers; it does not import point-set topology primitives, arbitrary open-cover choice, quotient compacta, ambient Euclidean universality, or host completeness.

Definition 7046.1 (Menger compactum carrier)**.** *A* $\mathsf{MengerCompactum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, K, D, E, R, H, C, P, N)$$

with the following rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *carrier and distance classifier.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover compactness row.*
3. *D is the* $\mathsf{CoveringDimension}^{\uparrow}$ *row recording the displayed finite-order cover refinement boundary.*
4. *E is the* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *budget row for the finite net sizes attached to inspected radii.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback seal for the distance and radius data exposed by M, K, D, E.*
6. *H records componentwise* hsame *transport for the metric source, compactness row, covering-dimension row, entropy budget, Real readback, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through the finite compact-metric and dimension rows.*
8. *P is the* Pkg *provenance over* $\mathsf{MengerCompactum}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{FiniteMetricEntropy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerCompactum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, D, E, R, H, C, P, N$. It has no coordinate for an ambient topological space, a selected infinite cover, quotient compactness, Euclidean universality, countable choice, or host real completeness.

Theorem 7046.2 (Menger compactum NameCert obligations)**.** *For every accepted* $\mathsf{MengerCompactum}^{\uparrow}$ *carrier* $U = (M, K, D, E, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MengerCompactum}^{\uparrow}}$ *surface consists exactly of metric source exposure through* M*; compact metric finite-cover exposure through* K*; covering-dimension refinement exposure through* D*; finite metric entropy budget exposure through* E*; terminal Real readback through* R*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*;* Pkg *provenance through* P*; and local naming through* N*.*

Proof. Project the accepted carrier of Definition 7046.1. The source rows M, K, D, E are the only rows that expose the compact-metric, finite-cover, covering-dimension, and finite-net budget data.

The readback row R is downstream of those displayed rows. It seals only the distance and radius data already exposed by the metric, compactness, dimension, and entropy surfaces.

Transport, replay, provenance, and local naming through H, C, P, N preserve that finite route. Since the carrier has no row for ambient topology, selected infinite covers, quotient compacta, Euclidean universality, countable choice, or host completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7046.3 (Menger compactum finite cover handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CoveringDimension}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *consumer of an accepted* $\mathsf{MengerCompactum}^{\uparrow}$ *packet must read the finite route*

$$M, K, D, E, R$$

in order: first the metric source, then the compact metric finite-cover row, then the covering-dimension refinement row, then the finite metric entropy budget, and finally the Real readback seal. No consumer can use the compactum as a point-set object without first exposing those finite certificate rows.

Proof. The metric source M supplies the carried distance interface, and the compact metric row K supplies the finite-cover compactness boundary. These two rows determine the compact-metric source seen by downstream consumers.

The covering-dimension row D refines the same finite-cover surface, while E records the finite net budget attached to inspected radii. Thus the dimension and entropy reads occur before the terminal Real seal is public.

Finally R, H, C, P, N read, transport, replay, source, and name that finite compactum route. The packet therefore exports no ambient point-set topology primitive, no arbitrary open-cover selector, no quotient compactum, no Euclidean universality theorem, and no host completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{MengerCompactum}^{\uparrow}$ compact-metric, covering-dimension, Real, or finite-entropy consumer exposes the $\mathsf{Met}^{\uparrow}$ source, $\mathsf{CompactMetric}^{\uparrow}$ finite-cover row, $\mathsf{CoveringDimension}^{\uparrow}$ refinement row, $\mathsf{FiniteMetricEntropy}^{\uparrow}$ budget, $\mathsf{Real}^{\uparrow}$ readback seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before using the compactum.

Independence witness: metric, compactmetric, covering-dimension, finite-metric-entropy, real: $\mathsf{MengerCompactum}^{\uparrow}$ is not bijective to any listed sibling because this packet binds a metric source, compact metric finite-cover row, covering-dimension refinement row, finite metric entropy budget, Real readback seal, transport, replay, provenance, and local naming into one finite compactum certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerCompactum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MengerCompactum}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite-cover compactness row, a CoveringDimensionUp refinement row, a FiniteMetricEntropyUp budget row, a RealUp readback seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MengerCompactum}\uparrow}$, seed)

Scope closed: Definition 7046.1, Definition 7046.2, and Definition 7046.3 seed the finite compact-metric and covering-dimension route for compactum consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{MengerCompactum}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim point-set topology primitives, arbitrary open-cover compactness, quotient compacta, ambient Euclidean universality, countable choice, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, covering-dimension refinement exposure, finite metric entropy budget exposure, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.MengerCompactumUp.

Chapter 7047

A Concrete Naming Certificate for $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$

Following Banach and Bishop–Bridges, the $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ packet records the finite endpoint-seal surface carried by a contraction orbit before any selected limit or quotient equality is read. It consumes the contraction theorem route of chapter 396, the iterate-window route of chapter 2359, the contraction data of chapter 3819, the completion admission rows of CompleteMetric packets, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the finite orbit-to-endpoint handoff only; it does not assert an ambient selected fixed point, quotient Cauchy equality, or host-level limit choice.

Definition 7047.1 (Cauchy fixed-point orbit-seal carrier)**.** *A* $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, O, M, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the contractive source row, read through* $\mathsf{BanachFixedPoint}^{\uparrow}$ *and the displayed* $\mathsf{ContractiveMap}^{\uparrow}$ *data.*
2. *F is the contraction map row, including the displayed contraction ledger and Lipschitz-bound witness used by the orbit.*
3. *O is the Picard iterate-window row inherited from* $\mathsf{PicardIteration}^{\uparrow}$*.*
4. *M is the Cauchy-modulus row for the displayed orbit windows.*
5. *E is the* $\mathsf{CompleteMetric}^{\uparrow}$ *endpoint-admission row for the Cauchy orbit.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of endpoint approximants.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the endpoint comparison data.*
8. *H records componentwise* hsame *transport for the contraction, orbit, modulus, endpoint, readback, Real-seal, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay through the same finite packet.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$*,* $\mathsf{BanachFixedPoint}^{\uparrow}$*,* $\mathsf{PicardIteration}^{\uparrow}$*,* $\mathsf{ContractiveMap}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, O, M, E, R, S, H, C, P, N$. It contains no selected fixed point, no quotient equality of Cauchy orbits, no ambient completeness oracle, and no hidden route from contraction data to a Real endpoint.

Theorem 7047.2 (Cauchy fixed-point orbit-seal NameCert obligations)**.** *For every accepted* CauchyFixedPointOrbit *carrier*

$$K = (B, F, O, M, E, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}}$ *surface consists exactly of contractive-source admission, contraction-map exposure, Picard iterate-window exposure, Cauchy-modulus exactness,* $\mathsf{CompleteMetric}^{\uparrow}$ *endpoint admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read an endpoint seal only after all those rows are displayed in the same packet.*

Proof. Project the carrier of Definition 7047.1. The analytic rows are B, F, O, M, E, R, S, and the structural rows are H, C, P, N.

The contraction and Picard rows B, F, O supply the finite orbit source. The modulus row M records the Cauchy rate, while E admits the orbit to the completion-facing endpoint route.

The readback and seal rows R, S turn endpoint approximants into regular rational and Real-facing data. Transport, replay, provenance, and local naming then preserve the same row order. Since the carrier has no further coordinate, the listed obligations exhaust the NameCert surface. □

Theorem 7047.3 (Cauchy fixed-point orbit-seal completion handoff)**.** *Every endpoint read from an accepted* $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ *packet factors through the route*

$$\mathsf{BanachFixedPoint}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{ContractiveMap}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exposes the contractive source, finite Picard windows, Cauchy modulus, completion admission, regular rational readback, and Real seal as separate rows. It cannot replace those rows by a selected limit, quotient Cauchy equality, ambient host completeness, or an unrecorded fixed-point oracle.

Proof. Start with the accepted carrier $K = (B, F, O, M, E, R, S, H, C, P, N)$. The only contraction-facing rows are B and F, and the only orbit-facing row is the displayed Picard window O.

Use the Cauchy-modulus row M to pass from finite orbit windows to the completion-facing admission row E. This keeps the endpoint request attached to the same contraction and iterate data.

The completion handoff then passes through R and S. The regular rational readback records endpoint approximants before the terminal Real seal is consumed.

Finally replay the packet through H, C, P, N. Componentwise transport and provenance keep the consumer on the same finite route, so no selected limit, quotient equality, host completeness oracle, or hidden fixed-point coordinate is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ packet is generated from contractive source data, Picard iterate windows, a Cauchy modulus, CompleteMetric endpoint admission, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}}$, seed)

Scope closed: Definition 7047.1, Definition 7047.2, and Definition 7047.3 seed the finite contraction-orbit endpoint handoff through BanachFixedPointUp, PicardIterationUp, ContractiveMapUp, CompleteMetricUp, RegSeqRatUp, RealUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected fixed points, quotient Cauchy orbit equality, ambient host completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction exposure, Picard-window compatibility, Cauchy-modulus exactness, CompleteMetric endpoint admission, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFixedPointOrbitSealUp.

Chapter 7048

A Concrete Naming Certificate for $\mathsf{FiniteProductUniformModulus}^{\uparrow}$

Following Kelley and Bishop–Bridges on finite product compactness and uniform continuity, the $\mathsf{FiniteProductUniformModulus}^{\uparrow}$ packet records the finite product-level route from coordinate local moduli to a tolerance-only uniform-modulus handoff. It consumes the finite product compactness routes of **??** and chapter 1822, the compact metric source of chapter 129, the continuous-map source of chapter 130, and the target uniform-modulus surface of chapter 369. The packet names finite product modulus data only; it does not assert arbitrary product compactness, selected Lebesgue numbers, quotient function equality, or an external modulus selector.

Definition 7048.1 (Finite product uniform-modulus carrier)**.** *A* $\mathsf{FiniteProductUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, C, M, F, L, D, R, E, U, H, P, N)$$

with the following displayed rows:

1. *T is the finite product compactness row read through* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$*.*
2. *C is the compact metric product row read through* $\mathsf{CompactMetricProduct}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$*.*
3. *M is the* $\mathsf{ProductMetric}^{\uparrow}$ *triangle-routing row for product distances and projections.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *source row exposing coordinate local-modulus data.*
5. *L is the coordinate modulus exposure row, listing the finitely many local coordinate tolerances inspected by the product route.*
6. *D is the Rat and Dyadic lower-bound folding row that turns the finite coordinate list into one product tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for folded tolerances and product-distance comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product estimate.*
9. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *export row.*
10. *H records componentwise* hsame *transport for the product, compactness, metric, map, fold, readback, Real-seal, export, provenance, and local naming rows.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteProductUniformModulus}^{\uparrow}$, $\mathsf{TychonoffFiniteProduct}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteProductUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, M, F, L, D, R, E, U, H, P, N$. It has no coordinate for arbitrary products, selected finite nets, hidden compactness, quotient maps, or host-level modulus choice.

Theorem 7048.2 (Finite product uniform-modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteProductUniformModulus}^{\uparrow}$ *carrier*

$$K = (T, C, M, F, L, D, R, E, U, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteProductUniformModulus}^{\uparrow}}$ *surface consists exactly of finite product compactness admission, compact metric product exposure, product-metric triangle routing, continuous-map coordinate modulus exposure, finite coordinate lower-bound folding through Rat and Dyadic tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing,* $\mathsf{UniformModulus}^{\uparrow}$ *export, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7048.1. The product and compactness rows T, C, M expose the finite product source and its metric route.

The continuous-map and coordinate modulus rows F, L provide the local tolerance data. The fold row D is the only row that combines the finitely many coordinate lower bounds into one product tolerance.

The readback, Real seal, and export rows R, E, U then present the tolerance-only uniform-modulus handoff. Transport, replay, provenance, and local naming preserve the same packet, and the carrier has no hidden coordinate for the excluded product or choice principles. □

Theorem 7048.3 (Finite product uniform-modulus export)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteProductUniformModulus}^{\uparrow}$ *packet reads the product modulus through the route*

$$\mathsf{TychonoffFiniteProduct}^{\uparrow},\ \mathsf{CompactMetricProduct}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{ProductMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

Coordinate local moduli remain visible until the Rat and Dyadic lower-bound fold has produced the product tolerance; the product-metric triangle row remains visible until the Real-sealed estimate is exported. No consumer may replace this route by arbitrary product compactness, a selected finite-net oracle, quotient function equality, or an external modulus selector.

Proof. Start with the finite product compactness rows T and C. They expose a finite compact product route and do not supply an arbitrary product theorem.

Use the product-metric row M together with the continuous-map row F. Projection and triangle reads are kept visible while the coordinate local moduli are inspected through L.

Fold the coordinate tolerances through D and read the resulting comparisons through R and E. This produces one product tolerance inside the same finite packet.

Finally export through U, H, P, N. The $\mathsf{UniformModulus}^{\uparrow}$ row is therefore a handoff from displayed product, compactness, coordinate-modulus, lower-bound, readback, Real-seal, transport, replay, provenance, and naming rows, not an external selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteProductUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteProductUniformModulus}^{\uparrow}$ packet is generated from finite product compactness, compact metric product, product-metric triangle routing, continuous-map coordinate modulus rows, Rat and Dyadic lower-bound folding, RegSeqRat readback, a Real seal, UniformModulus export, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteProductUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7048.1, Definition 7048.2, and Definition 7048.3 seed the finite product uniform-modulus packet over TychonoffFiniteProductUp, CompactMetricProductUp, CompactMetricUp, ProductMetricUp, ContinuousMapUp, RegSeqRatUp, RealUp, UniformModulusUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteProductUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary product compactness, selected finite nets, quotient function equality, external modulus selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite product compactness exposure, compact metric product routing, coordinate local-modulus exposure, lower-bound fold exactness, RegSeqRat readback, Real sealing, UniformModulus export, transport, replay, provenance, and local naming for BEDC.Derived.FiniteProductUniformModulusUp.

Chapter 7049

A Concrete Naming Certificate for $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$

Following Bishop and Bridges on total boundedness in constructive metric spaces, the $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ packet records the finite-net bridge needed by completion-facing consumers. It sits after the metric carrier of chapter 57, the metric-space surface of chapter 4230, the finite-net row of chapter 1767, the dyadic tolerance source of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a metric-space-specific total-bounded handoff without treating total boundedness as a free abstract parameter and without importing open-cover compactness, quotient point equality, selected centers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7049.1 (MetricSpaceTotalBounded carrier)**.** *A* $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, B, E, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row for the source under inspection.*
2. *S is the* $\mathsf{Met}^{\uparrow}$ *carrier row, recording the point surface on which the distance row is read.*
3. *B is the* $\mathsf{TotallyBounded}^{\uparrow}$ *budget row, displayed as finite requests rather than as an ambient compactness principle.*
4. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, naming the centers and cover checks actually consumed at the requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to read the positive precision and comparison cells.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular-rational precision data downstream from D.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the metric, total-bounded, finite-net, dyadic, and regular-rational rows have been displayed.*
8. *H records componentwise* hsame *transport for $M, S, B, E, D, R, L, C, P, N$.*
9. *C records displayed* Cont *replay from a finite precision request through the same rows.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{MetricSpaceTotalBounded}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, S, B, E, D, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, quotient point equality, selected centers outside E, countable choice, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7049.2 (MetricSpaceTotalBounded NameCert obligations)**.** *For every accepted* MetricSpaceTotalBounde *carrier*

$$T = (M, S, B, E, D, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricSpaceTotalBounded}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric row, metric-space source, total-bounded budget, finite-epsilon-net row, dyadic tolerance ledger, regular-rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The finite-net exactness clause requires every cover read to expose E under the precision named by D and R. The metric-source clause requires M and S to be consumed before any total-bounded budget is read. The non-escape clause refuses open-cover compactness, quotient point equality, selected centers outside the displayed net, countable choice, host compactness, and ambient Real completeness.*

Proof. Project the carrier of Definition 7049.1. The rows $M, S, B, E, D, R, L, H, C, P, N$ are the complete visible inventory of the accepted packet.

Metric-source exposure is the first projection step. A consumer receives the distance row M only together with the metric-space source row S, so the total-bounded budget cannot be read as an untyped external predicate.

Finite-net exactness is the next projection step. The budget row B reaches a concrete finite net only through E, and the positive precision is read through the dyadic row D and regular-rational row R before any Real seal L is available.

Transport, replay, provenance, and local naming preserve the same route. Since the carrier contains no coordinate for open-cover compactness, quotient point equality, selected centers outside the finite net, countable choice, host compactness, or ambient Real completeness, none of those data can be exported by the local NameCert surface. □

Theorem 7049.3 (MetricSpaceTotalBounded finite-net handoff)**.** *Every compact-metric or completion-facing consumer of an accepted* $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ *carrier reaches finite total-bounded data only through the displayed route*

$$M, S, B, E, D, R, L, H, C, P, N.$$

In particular, the consumer receives a metric-space-specific finite-net handoff: the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row E is read under the* $\mathsf{Met}^{\uparrow}$ *row M,* $\mathsf{Met}^{\uparrow}$ *row S,* $\mathsf{TotallyBounded}^{\uparrow}$ *budget B,* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row D, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal L may be consumed.*

Proof. Begin with the obligation surface of Definition 7049.2. It places the metric and metric-space rows before the total-bounded budget and finite-net rows.

Read the finite net through E. The centers and cover checks are visible only at the displayed precision, so the dyadic and regular-rational rows D and R remain upstream of the terminal Real seal.

Finally use the structural rows H, C, P, N. They replay the same finite route for compact-metric and completion-facing consumers and do not add open-cover compactness, quotient point equality, a selected center outside E, countable choice, host compactness, or ambient completeness. The handoff is therefore the concrete finite-net surface over the named metric space. □

Falsifiable prediction: Every accepted $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ read is refuted if it reaches a finite-net, compact-metric, or completion-facing consumer without displaying the metric row, metric-space source row, total-bounded budget, finite-epsilon-net row, dyadic tolerance row, regular-rational readback row, Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: metric, metric-space, total-bounded, finite-epsilon-net, dyadicratcore, regseqrat, real: $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric-space source, a total-bounded budget, a finite-net row, dyadic and regular-rational precision rows, a Real seal, transport, replay, provenance, and local naming as one finite handoff packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSpaceTotalBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSpaceTotalBounded}^{\uparrow}$ packet is generated from a metric row, metric-space source row, total-bounded budget, finite-epsilon-net row, dyadic tolerance row, regular-rational readback row, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSpaceTotalBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7049.1, Definition 7049.2, and Definition 7049.3 seed the metric-space finite-net route over total-bounded, dyadic, regular-rational, and Real-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSpaceTotalBounded}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient point equality, selected centers outside the displayed net, countable choice, host compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric-source admission, total-bounded budget exactness, finite-net handoff, dyadic and regular-rational precision nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricSpaceTo

Chapter 7050

A Concrete Naming Certificate for $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$ packet records a finite choice-sequence observation only through the located window currently displayed to a construction. It refines the finite-prefix carrier of chapter 1482 by requiring a located readback route through chapter 126, chapter 301, chapter 38, chapter 3857, and chapter 6030. The packet supports finite-window Real and completion arguments while refusing countable-choice, selected-tail, quotient-equality, and ambient-completeness coordinates.

Definition 7050.1 (Located finite choice-sequence carrier)**.** *A* $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite choice index row, naming the inspected choice-sequence positions and no tail positions outside the packet.*
2. *W is the located window row, recording the finite located interval, ball, or tolerance window attached to the indices of I.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the same finite choice-index and located-window data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed* $\mathsf{StreamName}^{\uparrow}$ *window and located tolerance data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback has been displayed.*
6. *H records componentwise* hsame *transport for the finite choice index, located window,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for the finite route*

$$I, W \longmapsto S \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LocatedSequence}^{\uparrow}$, $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets whose finite choice index, located window, stream window, regular rational readback, and real seal describe one displayed finite route. It exports no countable-choice functional, no selected infinite sequence, no quotient real equality, no host real equality, and no $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 7050.2 (Located finite choice-sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$ *carrier* $L = (I, W, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}$ *surface consists exactly of finite choice-index admission through* I*, located-window admission through* W*,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every consumer from the finite-window Real route or a completion-facing route must read those rows in the displayed order

$$\mathsf{ChoiceSequence}^{\uparrow},\ \mathsf{LocatedSequence}^{\uparrow},\ \mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow} \longmapsto I, W \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The packet exports no countable-choice functional, selected infinite sequence, quotient real equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7050.1. The rows I and W are finite displayed rows, so the choice-sequence input is visible only at the named inspected positions and only through the located window recorded in the packet.

The stream and analytic readback rows are downstream of that finite surface. The row S opens the $\mathsf{StreamName}^{\uparrow}$ window from I, W, the row R repacks the same displayed window as $\mathsf{RegSeqRat}^{\uparrow}$ data, and the row E seals only that readback as $\mathsf{Real}^{\uparrow}$ data.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. They do not add a new analytic coordinate; they keep the finite choice index, located window, stream window, regular rational readback, and real seal in one packet.

The sibling chapters supply the boundary discipline. chapter 3857 fixes the located-window route, and chapter 6030 fixes the finite choice-sequence modulus route into $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Since the carrier has no coordinate for a countable-choice functional, a selected infinite sequence, quotient real equality, host real equality, or ambient real completeness, no local $\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}$ consumer can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}$ packet is generated from a finite choice index, located window, StreamName window, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7050.1 and Definition 7050.2 seed the finite located choice-sequence surface for Real and completion finite-window consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedFiniteChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected infinite sequences, quotient real equality, host real equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window stability, StreamName exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedFiniteChoiceSequenceUp

Chapter 7051

A Concrete Naming Certificate for $\mathsf{RationalBallBasis}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalBallBasis}^{\uparrow}$ packet records the finite rational centre-radius basis used before a metric completion or Real-facing consumer reads a regular Cauchy name. It sits between the rational carrier of chapter 37, the dyadic tolerance surface of chapter 291, the metric test interface of chapter 57, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet is a finite rational-ball basis; it is not a host open-ball family, a quotient completion point, a selected limit, a countable-choice basis, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 7051.1 (Rational ball basis carrier)**.** *A* $\mathsf{RationalBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, R, D, M, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the centre row, naming a finite list of* $\mathsf{Rat}^{\uparrow}$ *centres.*
2. *R is the positive rational radius row attached to those centres.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare a requested precision with the rational radius data.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-test row, reading only centre-radius comparisons displayed by C, R, D.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the same rational-ball tests.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular handoff has been displayed.*
7. *H records componentwise* hsame *transport for centres, radii, dyadic tolerances, metric tests, regular handoff, Real seal, continuation, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from a rational-ball request through C, R, D, M, Q, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RationalBallBasis}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBallBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, R, D, M, Q, E, H, T, P, N$. It has no coordinate for host open balls, selected subbases, quotient completion points, chosen limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7051.2 (Rational ball basis NameCert obligations)**.** *For every accepted* $\mathsf{RationalBallBasis}^{\uparrow}$ *carrier* $B = (C, R, D, M, Q, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalBallBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, centre exposure, positive-radius exposure, dyadic tolerance exactness, metric distance-test factorization, regular rational handoff, Real seal nonescape, componentwise transport, continuation replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7051.1. The centre row C and positive radius row R display the rational balls being inspected, and the dyadic tolerance row D supplies the finite precision comparison used by those balls.

The metric row M reads only the displayed centre-radius-tolerance data. The regular handoff Q is therefore reached through the same finite distance tests, and the Real seal E is downstream of Q.

The structural rows H, T, P, N preserve, replay, source, and name that route. Since the excluded objects are absent from the carrier inventory, no additional NameCert obligation can expose host open balls, selected subbases, quotient completion points, chosen limits, countable choice, or ambient Real completeness. □

Theorem 7051.3 (Rational ball basis regular Cauchy handoff)**.** *Every metric-completion, Real-separability, or finite-cover consumer of an accepted* $\mathsf{RationalBallBasis}^{\uparrow}$ *carrier reaches its regular Cauchy data only through the displayed route*

$$\mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the carrier rows $C, R, D, M, Q, E, H, T, P, N$*. The exported basis supplies rational centres, positive rational radii, dyadic tolerances, metric distance tests, a regular rational handoff, and a Real seal, but it supplies no arbitrary open-ball basis, quotient completion point, selected real point, countable-choice enumeration, or ambient compactness principle.*

Proof. Project the accepted carrier and read the rational-ball rows C, R, D. These rows bind the centres, positive radii, and dyadic tolerances before any metric consumer can inspect the packet.

Next use the metric test row M. It factors every distance read through the displayed rational data, so the regular handoff Q receives only centre-radius tests already present in the carrier.

Finally consume the Real seal E through the same handoff and structural rows. The transport, replay, provenance, and local naming rows keep the route attached to the displayed rational-ball packet, so completion, separability, and finite-cover consumers obtain the finite basis rather than any of the excluded host principles. □

Falsifiable prediction: Every $\mathsf{RationalBallBasis}^{\uparrow}$ metric-completion, Real-separability, or finite-cover read factors through rational centres, positive rational radii, a DyadicRatCore tolerance row, a MetricUp distance-test row, RegSeqRat handoff, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: rat, dyadicratcore, metric, regseqrat, real: $\mathsf{RationalBallBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rational centres, positive rational radii, dyadic tolerances, metric distance tests, regular handoff, Real seal, transport, replay, provenance, and local naming as one finite basis packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBallBasis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalBallBasis}^{\uparrow}$ packet is generated from RatUp centres, positive rational radii, a DyadicRatCore tolerance row, MetricUp distance tests, RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalBallBasis}\uparrow}$, seed)

Scope closed: Definition 7051.1, Definition 7051.2, and Definition 7051.3 seed the finite rational-ball route consumed by completion, separability, and finite-cover reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalBallBasis}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-ball bases, quotient completion points, selected real points, countable-choice enumerations, ambient compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked centre exposure, positive-radius exposure, dyadic tolerance exactness, metric distance-test factorization, regular handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalBallBasisUp.

Chapter 7052

A Concrete Naming Certificate for $\mathsf{CatZeroProjection}^{\uparrow}$

Following Bridson–Haefliger on complete CAT(0) spaces and Bacak on projection methods in Hadamard geometry, the $\mathsf{CatZeroProjection}^{\uparrow}$ packet records the finite BEDC surface for projecting a displayed point onto a displayed closed convex subset of a complete CAT(0) carrier. It sits after the CAT(0) comparison packet of chapter 6336, the Hadamard-space carrier of chapter 6734, the convex-set rows of chapter 208, the existing projection endpoint surface of chapter 2356, the complete-metric service of chapter 128, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the closest-point route as finite NameCert data; it does not import a host projection map, selected minimizer, quotient point equality, Hilbert embedding, local compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7052.1 (CatZeroProjection carrier)**.** *A* $\mathsf{CatZeroProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (Z, H, V, M, C, R, A, T, K, P, N)$$

with the following displayed rows:

1. *Z is the* $\mathsf{CATZeroMetric}^{\uparrow}$ *comparison row, carrying the finite nonpositive-curvature inequality ledger read before projection convexity is used.*
2. *H is the* $\mathsf{HadamardSpace}^{\uparrow}$ *complete geodesic source row over the same displayed metric carrier.*
3. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row naming the closed convex target subset and its admitted geodesic-combination reads.*
4. *M is the* $\mathsf{MetricProjection}^{\uparrow}$ *endpoint row, recording the closest-point candidate only after the source point, convex target, and distance ledger have been displayed.*
5. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used by minimizing-window and Cauchy-readback consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for finite distance budgets and approximation windows.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for distance values, squared-distance comparisons, and endpoint-error budgets.*
8. *T records componentwise* hsame *transport for $Z, H, V, M, C, R, A, P, N$.*
9. *K records displayed* Cont *replay from CAT(0) comparison and convexity rows through the projection endpoint, complete-metric service, rational readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CatZeroProjection}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{HadamardSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CatZeroProjection}^{\uparrow}}$ *row.*

The classifier compares accepted projection packets only through these rows. It has no coordinate for a host nearest-point map, selected minimizer, quotient point equality, Hilbert-space replacement, local compactness, or ambient real completeness.

Theorem 7052.2 (CatZeroProjection NameCert obligations)**.** *For every accepted* $\mathsf{CatZeroProjection}^{\uparrow}$ *carrier*

$$Q = (Z, H, V, M, C, R, A, T, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CatZeroProjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the CAT(0) comparison row Z, Hadamard source row H, convex target row V, metric-projection endpoint row M, complete-metric service row C, regular rational readback R, Real seal A, componentwise transport T, displayed replay K, provenance P, and local naming N. Every projection-facing consumer must first display the CAT(0) and Hadamard rows, then the convex target, then the projection endpoint and completion/readback rows, before the local naming row is available.*

Proof. Project the carrier of Definition 7052.1. Its geometric source data are exactly Z, H, V: the CAT(0) comparison row, the complete geodesic Hadamard row, and the closed convex target row.

The endpoint stage is M, C, R, A. The projection endpoint is read only through the existing $\mathsf{MetricProjection}^{\uparrow}$ surface, while complete-metric service, regular rational readback, and the terminal Real seal expose the finite approximation and distance-budget rows needed by projection consumers.

Finally T, K, P, N transport, replay, source, and name the same displayed route. Since no other carrier coordinate exists, the local NameCert surface cannot export a host projection map, selected minimizer, quotient equality, Hilbert embedding, local compactness, or ambient real-completeness theorem. □

Theorem 7052.3 (CatZeroProjection nonescape)**.** *Every convex-projection or completion-facing read exported by an accepted* $\mathsf{CatZeroProjection}^{\uparrow}$ *packet factors through*

$\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{HadamardSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hs

The read exposes a displayed CAT(0) closest-point route and no host nearest-point selector, quotient point equality, selected minimizing sequence, Hilbert-space embedding, local compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 7052.2. Any consumer must first read the CAT(0) comparison row Z, Hadamard source row H, and convex target row V before it can inspect the projection endpoint row M.

The endpoint row is then routed through completion and readback data. The rows C, R, A keep the minimizing-window and distance-budget evidence inside displayed complete-metric, regular-rational, and Real-seal surfaces.

Transport, replay, provenance, and local naming are restricted to T, K, P, N. Therefore every exported projection read remains inside the finite route listed above, and the excluded host selector, quotient equality, chosen minimizing sequence, Hilbert embedding, local compactness, and ambient-completeness data have no projection from the packet. □

Falsifiable prediction: Every accepted $\mathsf{CatZeroProjection}^{\uparrow}$ consumer that reaches a closest-point or completion-facing projection read while omitting the CAT(0) comparison row, Hadamard source row, convex target row, MetricProjection endpoint row, complete-metric service, regular rational readback, Real seal, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cat-zero-metric, hadamard-space, convexset, metricprojection, completemetric, regseqrat, real: $\mathsf{CatZeroProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries CAT(0) comparison, complete geodesic source, convex target, projection endpoint, completion service, rational readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CatZeroProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CatZeroProjection}^{\uparrow}$ packet is generated from CatZeroMetric comparison rows, HadamardSpace complete geodesic rows, ConvexSet target rows, MetricProjection endpoint rows, CompleteMetric service rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CatZeroProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7052.1, Definition 7052.2, and Definition 7052.3 seed the finite CAT(0) closest-point projection route from displayed comparison and convexity data through completion, rational readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CatZeroProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host projection map, selected minimizer, quotient point equality, Hilbert embedding, local compactness, Lean verification, standard bridge closure, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, CAT(0) comparison exposure, Hadamard source exposure, convex target routing, MetricProjection endpoint admission, completion/readback exactness, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CatZeroProjectionUp.

Chapter 7053

A Concrete Naming Certificate for $\mathsf{EuclideanDomain}^{\uparrow}$

Following Bourbaki and Lang, the $\mathsf{EuclideanDomain}^{\uparrow}$ packet records the traditional division-with-remainder interface as finite BEDC NameCert data over a displayed ring carrier. It sits after the algebraic operation rows of chapter 43, the ideal membership and absorption surface of chapter 81, and the Euclidean backtracking witness of chapter 6993. The packet names the carrier, the divisor admissibility row, the Euclidean measure row, the quotient and remainder rows, the strict descent check, transport, replay, provenance, and local naming; it does not import a principal-ideal theorem, a selected gcd normalization, host quotient equality, Noetherian induction, or a classical field-of-fractions argument.

Definition 7053.1 (Euclidean domain carrier)**.** *A* $\mathsf{EuclideanDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, D, \nu, Q, M, I, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the consumed* $\mathsf{Ring}^{\uparrow}$ *row, naming the carrier, zero, one, addition, additive inverse, multiplication, and classifier used by the division read.*
2. *D is the nonzero-divisor admissibility row, recording that division requests are made only for displayed divisors outside the zero row.*
3. *ν is the Euclidean measure row, read as a value in a displayed well-founded comparison ledger.*
4. *Q is the quotient row for a requested pair (a, b) with b admitted by D.*
5. *M is the remainder row for the same request.*
6. *I is the* $\mathsf{Ideal}^{\uparrow}$ *dependency row, exposing the ring-indexed membership and absorption surface consumed by ideal-facing consequences.*
7. *B is the* $\mathsf{BezoutIdentity}^{\uparrow}$ *backtracking row reached by iterating the displayed descent trace.*
8. *H records componentwise* hsame *transport for $R, D, \nu, Q, M, I, B, C, P, N$.*
9. *C records displayed* Cont *replay from a division request through the quotient, remainder, descent, ideal, and Bezout rows.*
10. *P is the* Pkg *provenance over* $\mathsf{EuclideanDomain}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{BezoutIdentity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EuclideanDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host quotient equality, selected gcd normalization, a principal-ideal theorem, Noetherian induction, field-of-fractions construction, or ambient classical Euclidean-domain schema.

Theorem 7053.2 (Euclidean domain NameCert obligations)**.** *For every accepted* $\mathsf{EuclideanDomain}^{\uparrow}$ *carrier*

$$E = (R, D, \nu, Q, M, I, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EuclideanDomain}^{\uparrow}}$ *surface consists exactly of ring-source admission, nonzero-divisor admission, Euclidean-measure exposure, quotient-row exposure, remainder-row exposure, strict descent for nonzero remainders, ideal dependency exposure, Bezout backtracking exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every division read must expose $R, D, \nu, Q, M, I, B, H, C, P, N$ before any ideal-facing or Bezout-facing consumer can use the packet.*

Proof. Project the carrier of Definition 7053.1. The rows $R, D, \nu, Q, M, I, B, H, C, P, N$ are the complete visible inventory, so the ring operations and classifier are read through the consumed $\mathsf{Ring}^{\uparrow}$ surface before any division request is admitted.

For a request with divisor admitted by D, the quotient row Q and remainder row M are displayed together. The Euclidean measure row ν supplies the comparison ledger used to state that a nonzero remainder descends below the divisor.

The ideal and Bezout rows are dependency surfaces, not consequences imported from outside the packet. The $\mathsf{Ideal}^{\uparrow}$ row I exposes membership and absorption data over the same ring source, while the $\mathsf{BezoutIdentity}^{\uparrow}$ row B records the finite backtracking route reached by iterating the displayed descent.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no row names host quotient equality, selected gcd normalization, a principal-ideal theorem, Noetherian induction, a field of fractions, or an ambient classical schema, none of those objects belongs to the local NameCert surface. □

Theorem 7053.3 (Euclidean domain division descent)**.** *Every accepted* $\mathsf{EuclideanDomain}^{\uparrow}$ *carrier exposes division with remainder as the finite route*

$$R, D, \nu, Q, M \longmapsto I, B, H, C, P, N.$$

For each admitted divisor request, the quotient and remainder rows are read under the ring classifier from R; if the remainder is not the displayed zero row, the measure row ν records a strict descent comparison for that remainder. Iterating only these displayed descent rows may feed the Bezout backtracking row B and the ideal dependency row I.

Proof. Start with a division request admitted by D. The ring row R supplies the operations and classifier used to read the requested dividend, divisor, quotient, and remainder.

The quotient and remainder are the rows Q and M. The measure row ν is the only row that compares a nonzero remainder with its divisor, so descent cannot be replaced by a host termination argument or by an external Euclidean algorithm.

The iteration route is then confined to the displayed packet. The Bezout-facing row B may be read only after the descent trace is visible, and the ideal-facing row I may be read only as an $\mathsf{Ideal}^{\uparrow}$ dependency over the same ring row.

Transport and replay through H and C, with provenance and naming from P and N, preserve this finite route. Therefore the division-descent surface exports division data and its finite downstream dependencies without exporting host quotient equality, selected gcd normalization, a principal-ideal theorem, Noetherian induction, or field-of-fractions reasoning. □

Falsifiable prediction: Every accepted $\mathsf{EuclideanDomain}^{\uparrow}$ read is refuted if it reaches a division, ideal-facing, or Bezout-facing consumer without displaying the ring row, nonzero-divisor admission row, Euclidean measure row, quotient row, remainder row, ideal dependency row, Bezout backtracking row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: ring, ideal, bezout-identity: $\mathsf{EuclideanDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a ring source, a nonzero-divisor admission row, a Euclidean measure, quotient and remainder rows, ideal dependency rows, Bezout backtracking, transport, replay, provenance, and local naming as one finite division-with-remainder packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EuclideanDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{EuclideanDomain}^{\uparrow}$ packet is generated from a ring row, nonzero-divisor admission row, Euclidean measure row, quotient row, remainder row, ideal dependency row, Bezout backtracking row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EuclideanDomain}^{\uparrow}}$, seed)

Scope closed: Definition 7053.1, Definition 7053.2, and Definition 7053.3 seed the finite division-with-remainder route over RingUp, IdealUp, and BezoutIdentityUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EuclideanDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close principal ideal generation, unique gcd normalization, host quotient equality, Noetherian induction, field-of-fractions construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked division admission, quotient and remainder exactness, strict descent, ideal dependency exposure, Bezout backtracking, transport, replay, provenance, and local naming for BEDC.Derived.EuclideanDomainUp

Chapter 7054

A Concrete Naming Certificate for $\mathsf{ConstantRealStream}^{\uparrow}$

Following Bishop and Bridges on real numbers as regular Cauchy data, the $\mathsf{ConstantRealStream}^{\uparrow}$ packet records the constant certified point stream generated by one finite rational or dyadic point certificate. It sits between the rational source of chapter 37, the dyadic tolerance ledger of chapter 291, the finite stream naming surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The certificate names a constant stream route inside BEDC; it does not import quotient streams, host functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7054.1 (Constant real stream carrier)**.** *A* $\mathsf{ConstantRealStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *point row naming the rational source value.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the finite precision scale at which the same point is read.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose every displayed window names the same source point.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the constant stream windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback row is displayed.*
6. *H records componentwise* hsame *transport for the rational source, dyadic tolerance, stream window, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the constant point row through the stream and regular readback rows to the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstantRealStream}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstantRealStream}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, D, S, R, E, H, C, P, N$. It has no coordinate for host functions $\mathbb{N} \to \mathbb{Q}$, quotient stream equality, selected limits, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ConstantRealStreamUp`

Theorem 7054.2 (Constant real stream obligations)**.** *For every accepted* $\mathsf{ConstantRealStream}^{\uparrow}$ *carrier* $K = (Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstantRealStream}^{\uparrow}}$ *surface consists exactly of rational point exposure, dyadic tolerance exposure, constant* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, continuation replay, package provenance, and local naming. Every consumer must read these rows before using the constant certified point stream.*

Proof. Project the accepted carrier of Definition 7054.1. The rational source row Q and dyadic tolerance row D are the only point and precision rows visible to the local classifier.

The stream and readback rows preserve that same source. The $\mathsf{StreamName}^{\uparrow}$ row S exposes finite windows that all name the point displayed by Q, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is reached only through those windows.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since quotient streams, host functions, selected limits, countable-choice schedules, and ambient Real completeness are absent from the carrier, the listed rows exhaust the NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.ConstantRealStreamUp.ConstantRealStreamTasteGate single carrier alignment`

Theorem 7054.3 (Constant real stream regular-rational route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{ConstantRealStream}^{\uparrow}$ *packet reaches regular rational data only through the finite route*

$$\mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The route supplies a displayed rational point, a dyadic tolerance ledger, finite constant windows, and regular readback; it exports no quotient stream object, host sequence function, selected limiting process, or countable-choice schedule.

Proof. Read the source rows Q and D first. They determine the rational point and the finite tolerance scale available to every later window.

Next inspect the stream row S. A consumer receives constant windows only as displayed $\mathsf{StreamName}^{\uparrow}$ data, so any regular readback must be downstream of that finite schedule.

Finally project the readback row R and the structural rows. They keep the handoff attached to Q, D, S, and no carrier coordinate can provide an external sequence function, quotient stream equality, a selected limit, or countable choice. □

Theorem 7054.4 (Constant real stream Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{ConstantRealStream}^{\uparrow}$ *packet must pass through the displayed route*

$$Q, D, S, R, E, H, C, P, N.$$

The terminal seal E *certifies only the constant stream route already displayed by* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. It cannot be used as quotient stream equality, a host constant function, a selected limit, countable choice, or an ambient Real-completeness theorem.*

Proof. Start from Definition 7054.3. It places every regular rational readback downstream of the rational point, dyadic tolerance, and constant stream-window rows.

The Real seal row E is listed after that readback in the carrier. Therefore a Real-facing consumer cannot inspect the terminal seal before Q, D, S, R are exposed.

Transport, replay, provenance, and local naming preserve the same finite packet. Since the refused objects are not rows of K, the seal has no route by which to export quotient stream equality, a host function, a selected limit, choice, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{ConstantRealStream}^{\uparrow}$ Real-facing read factors through a rational point row, dyadic tolerance row, constant StreamName row, RegSeqRat readback row, Real seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{ConstantRealStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the constant-window route from a displayed point to a regular rational readback and terminal Real seal as one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstantRealStream}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstantRealStream}^{\uparrow}$ packet is generated from a RatUp point row, a DyadicRatCore tolerance row, a StreamName constant-window row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstantRealStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7054.1, Definition 7054.2, Definition 7054.3, and Definition 7054.4 seed the constant certified point stream route from finite rational and dyadic rows to StreamName, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstantRealStream}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host sequence functions, selected limits, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rational point exposure, dyadic tolerance exactness, constant-window exposure, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstantRealStrea

Chapter 7055

A Concrete Naming Certificate for $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$

Following Bishop and Bridges on monotone bounded real sequences and constructive Cauchy completion, the $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ packet records a finite L10 route by which a monotone regular Cauchy stream receives a terminal real-name readback. It sits after the dyadic rational core of chapter 291, the stream-window interface of chapter 126, the regular rational sequence route of chapter 301, and the terminal real-name boundary of chapter 38. The packet names the monotone limit route through displayed order, tolerance, and seal rows; it does not import a host least-upper-bound theorem, selected supremum, quotient equality of reals, countable choice, or ambient completeness.

Definition 7055.1 (Monotone regular Cauchy limit carrier)**.** *A* $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, O, M, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the source stream, exposing only finite windows requested by the current tolerance read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same stream windows.*
3. *D is the dyadic tolerance ledger read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row on the regular rational readbacks.*
5. *M is the monotonicity ledger, recording that every displayed later window is O-above the earlier window after both have been read through R.*
6. *T is the regular Cauchy tail ledger, routing each dyadic tolerance through a finite tail threshold and a displayed R, D, O comparison.*
7. *L is the candidate limit row, read only as a* $\mathsf{Real}^{\uparrow}$ *name assembled from the displayed regular tail route.*
8. *E is the exactness row saying that every visible tolerance request for L factors through S, R, D, O, M, T before the real seal is exposed.*
9. *H records componentwise* hsame *transport for stream, readback, tolerance, order, monotonicity, tail, limit, exactness, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tolerance request through* $S, R, D, O, M, T, L, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, O, M, T, L, E, H, C, P, N$. It has no coordinate for a host supremum, selected limit, quotient equality of real names, trichotomy, countable choice, or ambient real completeness.

Theorem 7055.2 (Monotone regular Cauchy limit NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ *carrier $K = (S, R, D, O, M, T, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonotoneRegularCauc}}$ *surface consists exactly of stream-window admission, regular rational readback, dyadic tolerance admission, total-order comparison exposure, monotonicity-ledger exactness, regular Cauchy tail exposure, terminal* $\mathsf{Real}^{\uparrow}$ *limit admission, limit-exactness replay, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted carrier. The only analytic rows are S, R, D, O, M, T, L, E, so every limit read starts with finite stream windows and regular rational readback.

The order and monotonicity rows are downstream of that readback. Row O compares only displayed regular rational windows, while M records the monotone direction for those same windows.

The tail ledger T then routes a dyadic tolerance through a finite threshold, and E checks that the candidate $\mathsf{Real}^{\uparrow}$ row L is exposed only after this route has been replayed. Thus the real-facing output is a finite NameCert route rather than a host supremum.

Finally H, C, P, N transport, replay, source, and name the same row surface. Since the excluded objects are absent from K, the listed obligation families exhaust the local certificate. □

Theorem 7055.3 (Monotone regular Cauchy limit induction route)**.** *Every finite induction over an accepted* $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ *read proceeds by extending the stream-window row S, transporting the regular rational readback R, preserving the dyadic tolerance ledger D, appending one* $\mathsf{TotalOrder}^{\uparrow}$ *comparison in O, and replaying the monotonicity and tail rows M, T before the* $\mathsf{Real}^{\uparrow}$ *limit row L may be consumed. The route exports no host least-upper-bound theorem, no selected supremum, no quotient equality of real names, no countable-choice selector, and no ambient completeness principle.*

Proof. Use induction on the displayed window extension in S. The base window exposes the initial regular rational readback through R and records the initial dyadic tolerance cell through D.

For the successor window, append the next readback cell and compare it with the previous visible cell through O. The monotonicity ledger M stores only this displayed comparison, so the induction step has no access to a global order theorem.

The regular Cauchy row T is then replayed at the requested tolerance. It supplies the finite tail threshold needed by the candidate real-name row L, and E records that the real seal is reached only through the same finite route.

The structural rows H, C, P, N preserve the induction route under hsame, replay it under Cont, record its provenance, and name the local certificate. Therefore every induction step remains inside the visible BEDC rows. □

Falsifiable prediction: An accepted $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ consumer is refuted if it reads the $\mathsf{Real}^{\uparrow}$ limit row without first displaying StreamName windows, RegSeqRat readback, DyadicRatCore tolerance cells, TotalOrder comparisons, monotonicity, regular Cauchy tail routing, transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real: none is bijective with $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$, because this packet alone binds stream windows, regular rational readback, dyadic tolerance cells, total-order comparisons, monotonicity, a regular Cauchy tail ledger, a terminal real-name row, exactness, transport, replay, provenance, and local naming into one finite limit route.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance cells, TotalOrder comparisons, monotonicity ledgers, regular Cauchy tail ledgers, a RealUp limit row, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7055.1, Definition 7055.2, and Definition 7055.3 seed the finite monotone regular Cauchy limit readback over StreamName, RegSeqRat, DyadicRatCore, TotalOrder, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneRegularCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host supremum theorem, selected limit, quotient equality of real names, trichotomy, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked stream-window admission, regular rational readback, dyadic tolerance exactness, total-order comparison exposure, monotonicity, regular Cauchy tail routing, RealUp limit admission, exactness, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneRegularCauchyLimitUp.

Chapter 7056

A Concrete Naming Certificate for $\mathsf{HyperbolicShearCoordinate}^{\uparrow}$

The $\mathsf{HyperbolicShearCoordinate}^{\uparrow}$ packet realizes the boundary-fixed disk-transport and observer-expansion comparison branch of chapter 9463 as finite NameCert data. It sits after the boundary-transport route of chapter 6267, the hyperbolic metric row of chapter 3474, the Busemann boundary row of chapter 3732, the boundary-shadow route of chapter 3813, and the observer-expansion packet of chapter 4167. The packet names a finite shear-coordinate read; it is not a completed Teichmuller space, a host geodesic lamination, a quotient boundary, or a physical observer trajectory.

Definition 7056.1 (Hyperbolic shear-coordinate carrier)**.** *A* $\mathsf{HyperbolicShearCoordinate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, B, M, U, S, E, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-surface row, read through Poincare disk boundary-transport data rather than a host disk.*
2. *B is the boundary-pair row naming the two visible boundary endpoints whose separation is inspected.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row on displayed disk and boundary data.*
4. *U is the Busemann or horofunction comparison row, read through the boundary metric route.*
5. *S is the shear-coordinate row, recording the signed comparison budget between the two boundary-fixed transports.*
6. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *comparison row, exposing how the same shear budget is consumed by observer-expansion reads.*
7. *R is the refusal row excluding completed boundary quotients, global geodesic laminations, physical trajectories, and selected ideal endpoints.*
8. *H records componentwise* hsame *transport for disk, boundary, metric, Busemann, shear, observer-expansion, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a shear request through $D, B, M, U, S, E, R, H, P, N$.*

10. *P is the* Pkg *provenance over* HyperbolicShearCoordinate$^{\uparrow}$, PoincareDiskBoundaryTransport$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{HyperbolicShearCoordinate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, B, M, U, S, E, R, H, C, P, N$. It has no coordinate for a completed ideal boundary, unlogged geodesic lamination, ambient Teichmuller coordinate, physical observer path, or selected boundary endpoint.

Theorem 7056.2 (Hyperbolic shear-coordinate NameCert obligations)**.** *For every accepted* HyperbolicShearCoordin *carrier $Q = (D, B, M, U, S, E, R, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicShearCoordinate}^{\uparrow}}$ *surface consists exactly of disk-surface admission, boundary-pair admission, hyperbolic-metric comparison, Busemann boundary comparison, shear-coordinate exactness, observer-expansion comparison, refusal exactness, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier onto D, B, M, U, S, E. These are the only geometric rows available to the packet, and each is a displayed NameCert row rather than an ambient hyperbolic object.

The shear row S is downstream from the boundary and metric rows. It may compare two boundary-fixed transports only after D, B, M, U have been displayed, so the coordinate is a finite ledger read rather than a selected ideal endpoint.

The observer-expansion row E consumes that same finite shear budget. The refusal row R then blocks any rereading as a completed boundary quotient, geodesic-lamination coordinate, Teichmuller parameter, physical trajectory, or selected endpoint.

Finally H, C, P, N transport, replay, source, and name the displayed route. Since the excluded host objects are not coordinates of Q, the listed obligation families exhaust the local certificate. □

Theorem 7056.3 (Hyperbolic shear-coordinate observer route)**.** *Every observer-expansion consumer of an accepted* HyperbolicShearCoordinate$^{\uparrow}$ *packet must first expose the disk row D, boundary-pair row B, hyperbolic metric row M, Busemann comparison row U, shear-coordinate row S, refusal row R, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The route exports no completed boundary quotient, no host geodesic lamination, no ambient Teichmuller coordinate, no physical observer trajectory, and no selected ideal endpoint.*

Proof. Begin with the NameCert obligations. They make D, B, M, U, S, E, R the complete geometric surface of the packet.

Read the boundary-fixed transport first. The disk row D and boundary-pair row B determine which finite boundary comparison is visible, while M and U attach metric and Busemann data to that same comparison.

The shear row S and observer-expansion row E are then replayed over the same displayed route. The refusal row R stays attached to the handoff, so no consumer may erase the boundary and metric rows into a global coordinate theorem.

The structural rows H, C, P, N preserve this route and add no hidden geometric object. Thus the observer-facing handoff is finite and classifier-based. □

Falsifiable prediction: An accepted HyperbolicShearCoordinate$^{\uparrow}$ read is refuted if an observer-expansion consumer obtains a shear coordinate without displaying Poincare disk boundary transport, boundary endpoints, HyperbolicMetric comparison, Busemann boundary comparison, shear exactness, refusal, transport, replay, provenance, and local naming rows.

Independence witness: poincare-disk-boundary-transport, hyperbolic-metric, hyperbolic-busemann-boundary, hyperbolic-boundary-shadow, hyperbolic-observer-expansion: no listed sibling is bijective with $\mathsf{HyperbolicShearCoordinate}^{\uparrow}$, because this packet alone binds boundary-fixed disk transport, boundary-pair comparison, metric readback, Busemann comparison, shear-coordinate exactness, observer-expansion comparison, refusal, transport, replay, provenance, and local naming into one finite row surface.

The Lean TasteGate route must provide explicit $\mathsf{Nontrivial\ HyperbolicShearCoordinate}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicShearCoordinate}^{\uparrow}$ instances before the packet can serve as a checked carrier-alignment witness. The instances must keep $D, B, M, U, S, E, R, H, C, P, N$ as separate fields and must distinguish a changed shear row from a changed metric or boundary row.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicShearCoordinate}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicShearCoordinate}^{\uparrow}$ packet is generated from Poincare disk boundary-transport rows, boundary-pair rows, HyperbolicMetric comparisons, Busemann boundary comparisons, shear-coordinate exactness, HyperbolicObserverExpansion comparisons, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicShearCoordinate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7056.1, Definition 7056.2, and Definition 7056.3 seed the finite boundary-fixed shear-coordinate route over disk, boundary, metric, Busemann, observer-expansion, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicShearCoordinate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary quotients, host geodesic laminations, ambient Teichmuller coordinates, physical observer trajectories, selected ideal endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-surface admission, boundary-pair admission, hyperbolic-metric comparison, Busemann comparison, shear exactness, observer-expansion comparison, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicShearCoordinateUp.

Chapter 7057

A Concrete Naming Certificate for $\mathsf{DoubleArrowSpace}^{\uparrow}$

Following Alexandroff and Urysohn's split-interval construction and Kelley-style order topology, the $\mathsf{DoubleArrowSpace}^{\uparrow}$ packet records the lexicographically ordered two-copy real interval as finite NameCert data. It sits beside the endpoint seal of chapter 38, the order-topology surface of chapter 1890, the topology carrier of chapter 88, the order-facing row family of chapter 1378, and the observable-open classifier of chapter 1860. The packet names the finite carrier and generated interval windows; it does not import a host compactness theorem, quotient point equality, selected cuts, or an unrestricted open-set lattice.

Definition 7057.1 (DoubleArrowSpace carrier)**.** *A* $\mathsf{DoubleArrowSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, T, L, W, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row, carrying the displayed left and right real endpoints currently inspected by the packet.*
2. *T is the split-tag row, with each inspected endpoint marked by a left or right copy of the interval.*
3. *L is the lexicographic order row: real endpoints are compared first, and equal endpoints are then compared by the split tag.*
4. *W is the generated order-topology interval-window row, listing the finite open rays and bounded order intervals actually exposed to topology consumers.*
5. *O is the observable-open row, read through* $\mathsf{SierpinskiSpace}^{\uparrow}$*, that records truth values for membership in the displayed interval windows.*
6. *H records componentwise* hsame *transport for endpoint rows, split tags, lexicographic order rows, interval windows, observable-open truth rows, continuation rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from R, T to L, from L to the generated windows W, and from W to the observable-open row O.*
8. *P is the* Pkg *provenance over* $\mathsf{DoubleArrowSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{DoubleArrowSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, T, L, W, O, H, C, P, N$. It has no coordinate for host point equality on a classical split interval, open-cover compactness, a quotient by adjacent endpoints, selected suprema, or an ambient topology.

Theorem 7057.2 (DoubleArrowSpace NameCert obligations)**.** *For every accepted* $\mathsf{DoubleArrowSpace}^{\uparrow}$ *packet $D = (R, T, L, W, O, H, C, P, N)$, the local NameCert surface consists exactly of endpoint admission through* $\mathsf{Real}^{\uparrow}$*, split-tag admission, lexicographic order admission through* $\mathsf{Order}^{\uparrow}$*, generated order-topology window exposure through* $\mathsf{RealOrderTopology}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$*, observable-open truth exposure through* $\mathsf{SierpinskiSpace}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. A consumer may read a double-arrow open only after those finite rows are displayed; it cannot replace them by a host split-interval topology, quotient point equality, open-cover compactness, selected cuts, or ambient real-order completeness.*

Proof. Project the carrier of Definition 7057.1. The rows $R, T, L, W, O, H, C, P, N$ are the whole accepted packet, so every double-arrow read must expose endpoints, split tags, lexicographic order, interval windows, and observable-open truth before the structural rows are consumed.

The order-topology route is finite. The row L is built from the displayed real endpoint comparison and split-tag comparison; W then lists only the generated order intervals currently inspected; and O records their observable truth values through $\mathsf{SierpinskiSpace}^{\uparrow}$.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores host point equality, compactness, selected cuts, quotient identification, or an unrestricted open-set lattice, none of those objects can appear among the local NameCert obligations. □

Theorem 7057.3 (DoubleArrowSpace order-topology route)**.** *Every* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{RealOrderTopology}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*, or* $\mathsf{SierpinskiSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{DoubleArrowSpace}^{\uparrow}$ *packet receives the generated open data only through the route*

$$\mathsf{Real}^{\uparrow},\ T,\ L,\ W,\ O,\ H,\ C,\ P,\ N.$$

The endpoint row supplies the real comparison boundary, the split-tag row separates the two copies at equal endpoints, the lexicographic order row supplies the order-facing classifier, and the interval-window row supplies the generated order-topology opens. The route exports no host compactness theorem, quotient equality between adjacent split points, selected endpoint cuts, or ambient open-set lattice.

Proof. Read the carrier in the order displayed by Definition 7057.1. The endpoint row R and tag row T determine the lexicographic order row L, and the generated windows W are the only topology-facing opens exposed by the packet.

Apply Definition 7057.2. It identifies the observable-open row O and the structural rows H, C, P, N as part of the same displayed certificate rather than as an external topology.

Thus every order-topology consumer receives the double-arrow data through the finite route above. No additional host topology, quotient point equality, compactness proof, selected cut, or open-set lattice is available from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{DoubleArrowSpace}^{\uparrow}$ observable-open read factors through the $\mathsf{Real}^{\uparrow}$ endpoint row, split-tag row, lexicographic order row, generated interval-window row, $\mathsf{SierpinskiSpace}^{\uparrow}$ observable truth row, hsame, Cont, Pkg, and local NameCert rows before any topology-facing consumer reads it.

Independence witness: real, real-order-topology, topology, order, sierpinski-space: $\mathsf{DoubleArrowSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries real endpoints, left/right split tags, lexicographic order, generated order-topology interval windows, observable-open truth values, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DoubleArrowSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{DoubleArrowSpace}^{\uparrow}$ packet is generated from a RealUp endpoint row, left/right split tags, a lexicographic order row, generated order-topology interval windows, a SierpinskiSpace observable-open row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoubleArrowSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7057.1, Definition 7057.2, and Definition 7057.3 seed the split-interval order-topology packet over RealUp, RealOrderTopologyUp, TopologyUp, OrderUp, SierpinskiSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoubleArrowSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host compactness, quotient point equality, selected cuts, a global open-set lattice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, split-tag stability, lexicographic order exactness, generated interval-window exposure, observable-open replay, transport, provenance, local naming, and nonescape for BEDC.Derived.DoubleArrowSpaceUp.

Chapter 7058

A Concrete Naming Certificate for $\mathsf{CountableBaseMetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CountableBaseMetric}^{\uparrow}$ packet records the finite interface by which metric balls are named through a countable basis schedule. It sits after the metric carrier of chapter 57 and the topology naming surface of chapter 88, while feeding the L10 Real and Completion route through chapter 1809, chapter 126, chapter 301, and chapter 38. The packet names only displayed center windows, radius ledgers, basis indices, containment witnesses, transport, replay, provenance, and local NameCert rows; it does not assert open-cover compactness, countable choice, selected limits, host powersets, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7058.1 (CountableBaseMetric carrier)**.** *A* $\mathsf{CountableBaseMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, T, I, R, S, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the displayed point windows and finite distance-comparison ledger.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-row readback, naming the topology generated from metric balls after the same metric source has been shown.*
3. *I is the countable basis index row, recording the finite portion of the basis schedule inspected by the current request.*
4. *R is the radius ledger row. It stores the positive rational or dyadic radius windows used to name balls around displayed centers.*
5. *S is the stability row, recording that a smaller displayed radius refines the same basis read without changing the center window or escaping the metric ledger.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row by which L10 consumers request finite basis schedules.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the radius comparisons attached to W and R.*
8. *H records componentwise* hsame *transport for metric, topology, index, radius, stability, stream, rational readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite ball request through M, T, I, R, S, W, Q, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CountableBaseMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableBaseMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, I, R, S, W, Q, H, C, P, N$. It has no coordinate for a host enumeration of all opens, a selected dense point, countable choice, quotient stream equality, open-cover compactness, selected completion limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7058.2 (CountableBaseMetric radius stability)**.** *For every accepted* $\mathsf{CountableBaseMetric}^{\uparrow}$ *packet $B = (M, T, I, R, S, W, Q, H, C, P, N)$, any basis read that narrows a displayed radius remains inside the same finite metric-ball route*

$$M, T, I, R \longmapsto S \longmapsto W, Q, H, C, P, N.$$

The stability row S preserves the metric center window, countable basis index, radius ledger, StreamName schedule, and RegSeqRat readback under componentwise hsame *transport and displayed* Cont *replay.*

Proof. Project the carrier of Definition 7058.1. The metric source row M, topology row T, basis index row I, and radius ledger R are the only rows visible before a stability request can be read.

A radius narrowing is therefore interpreted by S as a refinement of the displayed ball data, not as a new open set drawn from a host powerset. The center window and basis index remain attached to M, T, I, R.

The L10 rows W and Q read the same narrowed radius through finite stream windows and regular rational comparisons. Transport through H and replay through C preserve the named route, while P, N record provenance and local naming.

Since selected dense points, countable choice, quotient stream equality, open-cover compactness, and ambient completeness do not occur among the coordinates of B, radius stability is confined to the displayed finite route. □

Theorem 7058.3 (CountableBaseMetric ledger coverage)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CountableBaseMetric}^{\uparrow}$ *packet receives countable metric-basis information only through the displayed ledger rows $M, T, I, R, S, W, Q, H, C, P, N$. In particular, the packet covers metric balls, basis indices, radius refinements, finite stream schedules, regular rational readback, componentwise transport, continuation replay, provenance, and local naming; it exports no open-cover compactness theorem, countable-choice selector, selected completion limit, host equality of opens, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use the carrier route from Definition 7058.1. A downstream consumer first reads the metric ledger M and topology row T, then the countable basis index I and radius ledger R.

The stability row S records the finite containment relation needed to treat metric balls as basis data. It is downstream of the displayed metric and radius rows, so a consumer cannot replace it by a global topology or powerset assertion.

The L10 route then passes through W and Q, exposing only finite StreamName windows and RegSeqRat comparisons for the inspected basis schedule. Componentwise hsame transport and displayed Cont replay preserve that schedule under H and C.

Finally P and N source and name the same packet. The absent objects are not carrier coordinates, so no consumer can extract compactness, choice, selected limits, host open equality, quotient stream equality, or Real completeness from the ledger. □

Falsifiable prediction: Every accepted $\mathsf{CountableBaseMetric}^{\uparrow}$ L10 read exposes a MetricUp ledger, a TopologyUp generated-open row, countable basis indices, radius-refinement rows, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any Real or Completion consumer can inspect countable-basis data.

Independence witness: metric, topology, real-separability, streamname, regseqrat, real: none is bijective with $\mathsf{CountableBaseMetric}^{\uparrow}$ because this packet jointly carries metric-ball data, topological basis readback, countable indices, radius stability, finite schedules, rational readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CountableBaseMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CountableBaseMetric}^{\uparrow}$ packet is generated from a MetricUp source row, TopologyUp open-row readback, countable basis indices, radius ledgers, radius-stability rows, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableBaseMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7058.1, Definition 7058.2, and Definition 7058.3 seed the countable metric-basis route through metric balls, topology readback, StreamName schedules, RegSeqRat comparisons, and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableBaseMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected dense points, selected completion limits, host equality of opens, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-index exposure, radius-stability exactness, ledger coverage, StreamName and RegSeqRat readback, transport, replay, provenance, and local naming for BEDC.Derived.CountableBaseMetricUp.

Chapter 7059

A Concrete Naming Certificate for $\mathsf{LocatedDyadicApproximation}^{\uparrow}$

Following Bishop and Bridges on located constructive reals, the $\mathsf{LocatedDyadicApproximation}^{\uparrow}$ packet records the finite route from displayed located real bounds to dyadic approximation windows before a $\mathsf{Real}^{\uparrow}$ consumer reads the result. It sits beside the real-name seal of chapter 38, the regular rational readback of chapter 301, the stream naming surface of chapter 126, the dyadic precision ledger of chapter 291, and the Archimedean dyadic approximation route of chapter 6845. The chapter names a finite located approximation carrier; it does not quotient real names, choose limits, assert trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7059.1 (LocatedDyadicApproximation carrier)**.** *A* $\mathsf{LocatedDyadicApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, S, Q, D, A, W, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the located* $\mathsf{Real}^{\uparrow}$ *bounds row, exposing lower and upper rational guards for the named real source before any dyadic window is read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row that records the finite stream-name schedule through which the located bounds are inspected.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same source schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row, recording the dyadic denominator and tolerance ledger used by the approximation request.*
5. *A is the* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ *handoff row, displaying the bounded dyadic approximation discipline already licensed by the Archimedean route.*
6. *W is the approximation-window row, carrying the finite dyadic lower and upper endpoints visible to downstream consumers.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for the same located source, available only after R, S, Q, D, A, W have been displayed.*
8. *H records componentwise* hsame *transport for the located bounds, stream schedule, regular rational readback, dyadic precision, Archimedean handoff, approximation window, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a located approximation request through $R, S, Q, D, A, W, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedDyadicApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient reals, equality of real names outside the displayed seal, trichotomy, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7059.2 (LocatedDyadicApproximation NameCert obligations)**.** *For every accepted* LocatedDyadicApprox *carrier $L = (R, S, Q, D, A, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the located bounds, stream source, regular rational readback, dyadic precision, Archimedean handoff, approximation window, real seal, transport, replay, provenance, and local naming rows; source exposure requiring R, S, Q before any dyadic precision row is read; precision exposure requiring D, A before W; real handoff exactness requiring W before E; and ledger exhaustion excluding quotient reals, trichotomy, selected limits, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 7059.1. The rows $R, S, Q, D, A, W, E, H, C, P, N$ are displayed as separate coordinates, so the located bounds, stream schedule, regular rational readback, dyadic precision, and approximation window cannot be hidden inside the terminal real seal.

The source rows are read first. The located real bounds R are inspected through the stream-name row S and regular rational readback row Q, so the approximation request is tied to the displayed name rather than to a quotient real.

Next read the dyadic precision and Archimedean handoff rows D, A. They license the finite approximation window W before the Real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no quotient-real coordinate, trichotomy row, selected-limit row, choice schedule, or ambient-completeness witness, the listed rows exhaust the local NameCert surface. □

Theorem 7059.3 (Located dyadic approximation real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedDyadicApproximation}^{\uparrow}$ *packet receives the finite route*

$$R, S, Q, D, A, W \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer reads located real bounds, a stream-name schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row, an* $\mathsf{ArchimedeanDyadicApproximation}^{\uparrow}$ *handoff row, and a dyadic approximation window before the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no quotient real, trichotomy principle, selected limit, countable-choice schedule, or ambient completeness principle.*

Proof. Use Definition 7059.2. Its source exposure, precision exposure, and real handoff exactness clauses force the route R, S, Q, D, A, W, E before any real-facing read occurs.

After the seal is read, the remaining rows are H, C, P, N. They preserve the same finite packet by componentwise transport, displayed replay, provenance, and local naming.

Therefore a $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed located-bounds-to-window route with the terminal seal and structural rows. Since quotient reals, trichotomy, selected limits, countable choice, and ambient completeness are absent from both the carrier and the obligation surface, the handoff cannot export them. □

Lean4 (statement) `BEDC.Derived.LocatedDyadicApproximationUp.LocatedDyadicApproximationUp`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicApproximation}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{LocatedDyadicApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 7059.1, Definition 7059.2, and Definition 7059.3 seed the finite route from located real bounds to dyadic approximation windows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedDyadicApproximation}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.LocatedDyadicApproximationUp.LocatedDyadicApproximationUp`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, regular rational readback, dyadic precision exposure, Archimedean handoff, approximation-window exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDyadicApproximationUp.

Constructive story: A LocatedDyadicApproximation$^{\uparrow}$ packet is generated from located RealUp bounds, StreamName schedule rows, RegSeqRat readback rows, DyadicRatCore precision rows, an ArchimedeanDyadicApproximation handoff, a dyadic approximation window, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7060

A Concrete Naming Certificate for $\mathsf{FiniteGameMinimax}^{\uparrow}$

Following von Neumann's finite zero-sum game theorem, the $\mathsf{FiniteGameMinimax}^{\uparrow}$ packet names a saddle-certificate surface for finite payoff tables. The BEDC packet consumes the finite ordered-field duality rows of chapter 235, the simplex and convexity rows of chapter 208, the finite payoff-table rows of chapter 49, the scalar readback rows of chapter 38, and the ordered-field operations of chapter 45.

Definition 7060.1 (FiniteGameMinimax carrier)**.** *A* $\mathsf{FiniteGameMinimax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, J, A, \Delta_I, \Delta_J, L, V, S, H, C, P, N)$$

with the following displayed rows:

1. *I and J are finite row-player and column-player pure-strategy carriers.*
2. *$A : I \times J \to F$ is the payoff table, read as a finite* $\mathsf{Mat}^{\uparrow}$ *row over the ordered-field scalar carrier F.*
3. *Δ_I and Δ_J are the row and column mixed-strategy simplex rows, read through* $\mathsf{ConvexSet}^{\uparrow}$ *and the scalar* $\mathsf{Field}^{\uparrow}$ *operations.*
4. *L is the finite* $\mathsf{LPDuality}^{\uparrow}$ *primal-dual row whose primal variables are row-player mixture and lower-value coordinates, and whose dual variables are column-player mixture and upper-value coordinates.*
5. *V is the value row, recording the scalar endpoint selected by the displayed primal-dual equality in L.*
6. *S is the saddle-inequality ledger, listing the finite comparisons*

$$\sum_{i \in I} p_i A_{ij} \preceq_F V \quad \text{and} \quad V \preceq_F \sum_{j \in J} A_{ij} q_j$$

   for every displayed pure column $j \in J$ and pure row $i \in I$.
7. *H records componentwise* hsame *transport for the strategy carriers, payoff table, simplex rows, duality row, value row, saddle ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a saddle-value request through $A, \Delta_I, \Delta_J, L, V, S, H, P, N$.*

9. *P is the* Pkg *provenance over* FiniteGameMinimax$^{\uparrow}$, LPDuality$^{\uparrow}$, ConvexSet$^{\uparrow}$, Mat$^{\uparrow}$, Real$^{\uparrow}$, Field$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{FiniteGameMinimax}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host minimax selector, a hidden optimization solver, quotient equality of mixed strategies, an ambient completed real space, or a selected saddle pair outside the displayed finite packet.

Theorem 7060.2 (FiniteGameMinimax obligation surface)**.** *For every accepted* FiniteGameMinimax$^{\uparrow}$ *carrier $G = (I, J, A, \Delta_I, \Delta_J, L, V, S, H, C, P, N)$, the local* NameCert$_{\text{FiniteGameMinimax}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier comparison, finite simplex feasibility, finite* LPDuality$^{\uparrow}$ *handoff, saddle-inequality stability under* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A saddle-value read is admitted only after the payoff table, both mixed-strategy simplex rows, the duality row, the value row, and the saddle ledger are all visible in the same finite packet.*

Proof. Project the carrier of Definition 7060.1. The finite strategy carriers and payoff table are supplied by I, J, A, while Δ_I and Δ_J give the mixed-strategy feasibility rows.

The duality handoff is exactly the row L. It supplies the finite primal-dual comparison whose common scalar endpoint is recorded in V.

The saddle ledger S lists the pure-row and pure-column comparisons obtained from that endpoint. Componentwise transport through H preserves the strategy carriers, payoff entries, simplex coordinates, value endpoint, and comparison rows, while C, P, N replay, source, and name the same packet.

Since no row in the carrier stores a host solver, a quotient of strategy distributions, or an ambient completeness principle, the displayed fields exhaust the local NameCert obligations. □

Theorem 7060.3 (FiniteGameMinimax LPDuality handoff)**.** *Every saddle-value read from an accepted* FiniteGameMinimax$^{\uparrow}$ *packet factors through the displayed finite ordered-field* LPDuality$^{\uparrow}$ *row L of Definition 7060.1. The exported data are the payoff table, simplex feasibility rows, value endpoint, saddle comparisons, transport, replay, provenance, and local naming rows. The packet exports no optimization solver, host min/max selector, quotient strategy equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Read an accepted saddle-value request through the replay row C. By definition it must pass through the payoff table A, the two simplex rows Δ_I, Δ_J, the duality row L, the value row V, and the saddle ledger S.

The row L is already an LPDuality$^{\uparrow}$ finite ordered-field row. Its equality of primal and dual objective endpoints is the only source for the value coordinate V.

The saddle comparisons are then read from S as finite ordered-field comparisons against the same endpoint. Transport by H and provenance by P, N keep those comparisons in the named finite packet.

Thus every public saddle-value read factors through L and the listed BEDC rows, with no additional solver, selector, quotient, or completeness coordinate available. □

Closure status for NameCert$_{\text{FiniteGameMinimax}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{FiniteGameMinimax}^{\uparrow}}$, seed)

Scope closed: Definition 7060.1, Definition 7060.2, and Definition 7060.3 seed the finite zero-sum game saddle-certificate packet through displayed payoff, simplex, duality, value, and saddle-ledger rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteGameMinimax}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host minimax theorem, an optimizer-selection procedure, quotient equality of mixed strategies, real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, simplex feasibility, ordered-field duality handoff, saddle-inequality stability, replay, provenance, and local NameCert registration for BEDC.Derived.FiniteGameMinimaxUp.

Constructive story: A FiniteGameMinimax$^{\uparrow}$ packet is generated from finite pure-strategy carriers, a payoff table, mixed-strategy simplex rows, an LPDuality row, a value endpoint, saddle-inequality ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7061

A Concrete Naming Certificate for $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$

The $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ packet realizes the theorem-DNA complexity-scoring vision of chapter 9408 as finite NameCert data. It packages role-locus completeness, mass-vector extraction, weighted-score rows, refusal boundaries, transport, replay, provenance, and local naming so downstream discovery classifiers can inspect score reads without treating theorem text or proof length as a complete theorem object.

The score packet is consumed beside chapter 6358 and chapter 5914: theorem-genome supplies the lineage rows, while discovery-pipeline supplies the classifier-shift consumer route.

Definition 7061.1 (TheoremDnaComplexityScoring carrier)**.** *A* $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, R, D, M, W, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the theorem-candidate event-flow row.*
2. *R is the recognizer row that assigns theorem roles to subflows.*
3. *D is the eight-locus theorem-DNA row, displaying statement, dependencies, proof, certificates, ledger, status, canonical-site, and closing-seal loci.*
4. *M is the mass-vector row, recording the nine coordinates for the eight role masses and recognizer overhead.*
5. *W is the public weight row, listing the nine nonnegative weights used to turn M into a scalar score.*
6. *Q is the weighted-score comparison row, computing the accepted scalar score only from M and W.*
7. *B is the refusal-boundary row, rejecting score reads that omit any role locus, mass coordinate, weight coordinate, recognizer row, transport row, replay row, provenance row, or local name row.*
8. *H records componentwise* hsame *transport for theorem flow, recognizer, theorem-DNA loci, mass vector, weight profile, score comparison, refusal boundary, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a scoring request through $T, R, D, M, W, Q, B, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$*, chapter 9408, theorem-flow recognition, theorem-genome evolution, classifier-level information, theorem-DNA taste gates,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoremDnaComplexityScoring}^{\uparrow}}$ *row.*

The classifier compares accepted scoring packets only through these rows. It has no coordinate for statement-only theorem identity, proof-length-only scoring, hidden weights, unanchored recognizer overhead, prose novelty, or a score read outside the displayed finite packet.

Theorem 7061.2 (TheoremDnaComplexityScoring obligation surface)**.** *For every accepted* TheoremDnaComplexity *carrier $S = (T, R, D, M, W, Q, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TheoremDnaComplexityScoring}^{\uparrow}}$ *surface consists exactly of theorem-flow exposure, role-locus completeness, mass-vector extraction, public weight exposure, weighted-score computation, refusal-boundary exactness, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. A score read is accepted only after all eight role loci and all nine mass and weight coordinates are visible.*

Proof. Project the carrier of Definition 7061.1. The theorem object is not read from a sentence alone; it is read through the candidate row T, recognizer row R, and eight-locus DNA row D.

The vector part is supplied by M, and the scalarization is supplied by W and Q. Since Q is defined only from the displayed vector and weight rows, no hidden score coordinate is available.

The refusal boundary B records exactly the missing-row cases. Transport through H preserves the role, vector, weight, and score coordinates together, while C, P, N replay, source, and name the same finite packet.

Thus the NameCert obligation surface is exhausted by the listed finite rows. □

Theorem 7061.3 (TheoremDnaComplexityScoring refusal boundary)**.** *Any attempted* TheoremDnaComplexityScor *score read that omits one of the eight theorem-DNA role loci, one of the nine mass coordinates, one of the nine public weight coordinates, the recognizer row, the refusal-boundary row, componentwise transport, replay, provenance, or local naming is rejected by the carrier of Definition 7061.1. In particular, proof shortening, statement length, recognizer overhead, or score comparison alone is not an accepted theorem-DNA complexity score.*

Proof. Read a scoring request through C. The replay route reaches $T, R, D, M, W, Q, B, H, P, N$ before any scalar score can be exported.

If a role locus is missing, then D is not the eight-locus theorem-DNA row required by the carrier. If a mass or weight coordinate is missing, then Q has no displayed input from which to compute the scalar score.

If the recognizer, boundary, transport, replay, provenance, or local naming row is missing, then the score is detached from the finite NameCert packet. Each case is listed by B as a refusal case.

Therefore the accepted score read requires the full displayed route and cannot be obtained from any single theorem-locus or scalar comparison alone. □

Falsifiable prediction: Any accepted $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ score read without all eight role loci, all nine mass coordinates, all nine weight coordinates, the refusal boundary, transport, replay, provenance, and local naming rows refutes the carrier of Definition 7061.1.

Independence witness: theorem-genome-evolution, discovery-pipeline-calculus, theorem-dna-scoring-taste-gate, classifier-level-information: $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ is not bijective to these siblings because it jointly binds role-locus completeness, mass-vector extraction, public weight

rows, weighted-score comparison, refusal exactness, transport, replay, provenance, and local naming in one scoring packet.

The Lean follow-up for $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ must provide explicit Nontrivial TheoremDnaComplexityS and $\mathsf{FieldFaithful\ TheoremDnaComplexityScoring}^{\uparrow}$ TasteGate instances, keeping role loci, mass vectors, weights, score comparisons, and refusal rows as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{TheoremDnaComplexityScoring}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoremDnaComplexityScoring}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7061.1, Definition 7061.2, and Definition 7061.3 seed the theorem-DNA complexity-scoring packet as finite role-locus, mass-vector, weight, score, refusal, transport, provenance, and NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoremDnaComplexityScoring}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive discovery, theorem-generation power, score optimality, hidden weighting, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked role-locus completeness, mass-vector extraction, public weight exposure, weighted-score computation, refusal exactness, replay, provenance, local NameCert registration, and TasteGate compatibility for BEDC.Derived.TheoremDnaComplexityScoringUp.

Constructive story: A $\mathsf{TheoremDnaComplexityScoring}^{\uparrow}$ packet is generated from theorem-candidate flow, recognizer flow, eight theorem-DNA role loci, a nine-coordinate mass vector, a public nine-coordinate weight profile, a weighted-score comparison, refusal boundaries, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7062

A Concrete Naming Certificate for $\mathsf{BaireOnePointContinuity}^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the $\mathsf{BaireOnePointContinuity}^{\uparrow}$ packet records the finite certificate route by which a Baire-one real-valued function exposes one inspected continuity point. It consumes the pointwise-limit ledger of chapter 1604, the dense-open schedule of chapter 1542, the oscillation window of chapter 1839, and the epigraph handoff of chapter 1536. The packet names a local pointwise-continuity witness over displayed Real and LocatedReal rows; it does not assert a host theorem that every Baire-one map has a continuity point, and it does not choose a point outside the named certificate.

Definition 7062.1 (BaireOnePointContinuity carrier)**.** *A* $\mathsf{BaireOnePointContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, D, O, L, Z, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *source row, exposing the pointwise-limit ledger whose value reads are being tested for local continuity.*
2. *D is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, recording the finite neighborhood refinement requests available around the inspected point.*
3. *O is the* $\mathsf{RealOscillation}^{\uparrow}$ *oscillation-window row, recording the displayed tail-variation and clamped-window evidence used to test a local modulus.*
4. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph handoff row, recording the lower-bound route through which oscillation thresholds are made visible to a located consumer.*
5. *Z is the* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected source point and its terminal value read.*
6. *R is the local point-continuity row, recording the finite neighborhood, tolerance window, oscillation bound, and value-seal handoff accepted by the packet.*
7. *H records componentwise* hsame *transport for the Baire-one source, dense-open schedule, oscillation window, lower-semicontinuous handoff, located-real and real seals, local continuity row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point-continuity request through $B, D, O, L, Z, R, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BaireOnePointContinuity}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireOnePointContinuity}^{\uparrow}}$ *row.*

The carrier exports no global Baire theorem, no selected point outside Z, no quotient equality of real names, no countable choice over dense-open schedules, no open-cover compactness principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 7062.2 (BaireOnePointContinuity NameCert obligations)**.** *For every accepted* $\mathsf{BaireOnePointContinuity}^{\uparrow}$ *carrier*

$$Q = (B, D, O, L, Z, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireOnePointContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire-one source row, dense-open schedule, oscillation window, lower-semicontinuous epigraph handoff, located-real and real seals, local continuity row, transport, replay, provenance, and local naming rows; dense-open exposure requiring D to be displayed before a local neighborhood is used; oscillation-window exposure requiring O to be displayed before a tolerance claim is read; epigraph-handoff exposure requiring L to route every lower-bound comparison through the existing lower-semicontinuous packet; point-seal exposure requiring Z and R to be displayed before the local continuity claim is public; and ledger nonescape excluding choice, compactness, quotient real equality, semantic liminf data, and ambient completeness.*

Proof. Project the carrier of Definition 7062.1. The packet displays exactly the source, dense-open, oscillation, epigraph, point-seal, local-continuity, transport, replay, provenance, and local naming rows.

The dense-open and oscillation obligations are read from D and O. The dense-open schedule supplies the neighborhood refinement surface, while the oscillation row supplies the finite variation window that can be tested at the inspected point.

The lower-semicontinuous handoff is read from L. It keeps lower-bound comparisons on the same finite epigraph route already named by the source packet, so a consumer cannot replace the row by semantic liminf data or a host topology.

Finally the local point seal Z, R is reached through the structural rows H, C, P, N. Since none of these rows contains a selector over all dense-open schedules, a quotient equality of real names, open-cover compactness, or ambient Real completeness, the displayed obligations exhaust the local NameCert surface. □

Theorem 7062.3 (BaireOnePointContinuity oscillation window)**.** *Every local point-continuity read from an accepted* $\mathsf{BaireOnePointContinuity}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{BaireOneFunction}^{\uparrow}, \mathsf{BaireCategory}^{\uparrow}, \mathsf{RealOscillation}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{LocatedReal}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto R.$$

The $\mathsf{RealOscillation}^{\uparrow}$ *row must expose a finite oscillation window at the inspected point, and the* $\mathsf{BaireCategory}^{\uparrow}$ *row must expose the dense-open neighborhood schedule used to shrink the window, before the local continuity row R can be consumed.*

Proof. Start with a consumer of the local continuity row R. The carrier places R downstream of the Baire-one source row B, the dense-open schedule D, the oscillation row O, the lower-semicontinuous handoff L, and the point seal Z.

Apply the Baire-one pointwise schedule route from chapter 1604 and the dense-thread schedule of chapter 1542. These rows make the source value and neighborhood refinement visible before the oscillation request is inspected.

Now use the oscillation packet of chapter 1839. It supplies only a finite clamped window and readback route, not an ambient equality of functions or a completed real-space theorem.

The lower-semicontinuous handoff and point seal then preserve that same route. Hence the local continuity row is available only after the displayed oscillation window and dense-open refinement have both been exposed. □

Theorem 7062.4 (BaireOnePointContinuity local point handoff)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$*-facing read of an accepted* $\mathsf{BaireOnePointContinuity}^{\uparrow}$ *packet must expose the inspected point seal* Z*, the Baire-one source row* B*, the dense-open schedule* D*, the oscillation window* O*, the lower-semicontinuous handoff* L*, and the local continuity row* R *before any point-continuity claim is used. The handoff is a finite local NameCert route, not a choice of a continuity point from an external set and not a proof of global continuity.*

Proof. Project the carrier. The located point and terminal real value are recorded in Z, but Z is not a standalone selector; it is carried with B, D, O, L, R in the same finite packet.

Use the oscillation-window theorem above. It forces any local continuity read through the Baire-one source, dense-open schedule, oscillation window, lower-semicontinuous handoff, and point seal before R is available.

Transport and replay through H and C preserve exactly those rows, while P and N only record provenance and local naming. Therefore a LocatedReal-facing consumer receives a displayed local point handoff and no external point choice or global continuity theorem. □

Theorem 7062.5 (BaireOnePointContinuity nonescape)**.** *Every Real-facing, LocatedReal-facing, or lower-semicontinuous consumer of an accepted* $\mathsf{BaireOnePointContinuity}^{\uparrow}$ *packet remains inside the finite route*

$$B, D, O, L, Z, R, H, C, P, N.$$

The route can certify only the displayed Baire-one source, dense-open schedule, oscillation window, lower-semicontinuous epigraph handoff, located point seal, local continuity row, transport, replay, provenance, and local naming data. It cannot export a host theorem choosing a point of continuity, quotient equality of real names, countable choice, open-cover compactness, semantic liminf data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligation surface. It accounts for all rows of the accepted packet and names no consumer route outside $B, D, O, L, Z, R, H, C, P, N$.

The oscillation-window and local point handoff theorems keep continuity reads downstream of the Baire-one, dense-open, oscillation, lower-semicontinuous, and point-seal rows. These rows are finite certificate data, not host analytic principles.

The structural rows add no hidden coordinates. Componentwise transport, displayed replay, provenance, and local naming preserve the same packet, so the refused choice, compactness, quotient-equality, semantic-liminf, and ambient-completeness endpoints cannot be projected from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOnePointContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BaireOnePointContinuity}^{\uparrow}$ packet is generated from a BaireOneFunction source row, BaireCategory dense-open schedule, RealOscillation window, LowerSemicontinuous epigraph handoff, LocatedReal and Real seal row, local point-continuity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireOnePointContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7062.1, Definition 7062.2, Definition 7062.3, Definition 7062.4, and Definition 7062.5 seed the finite local point-continuity route from Baire-one pointwise data through dense-open, oscillation, lower-semicontinuous, located-real, and real seal rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireOnePointContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a global Baire-one continuity theorem, point selection outside the displayed seal, quotient real equality, countable choice, open-cover compactness, semantic liminf data, ambient Real completeness, Lean theorem checking, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dense-open exposure, oscillation-window exposure, lower-semicontinuous handoff, local point seal, transport, replay, provenance, and nonescape for BEDC.Derived.BaireOnePointContinuityU

Chapter 7063

A Concrete Naming Certificate for $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis and the classical dyadic basis used in differentiation theory, the $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$ packet records the finite interval basis read before an L10 Real or Completion consumer asks for a local refinement. It sits over the dyadic arithmetic surface of chapter 291, the rational interval locator of chapter 4202, the dyadic interval cover of chapter 1428, and the terminal real-name seal of chapter 38. The packet names only finite dyadic interval packets, rational locator rows, finite cover refinement rows, and Real seal handoffs; it is not a measure-theoretic differentiation theorem, a Vitali selection principle, quotient real equality, or ambient completeness.

Definition 7063.1 (Dyadic differentiation basis carrier)**.** *A* $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, L, I, C, R, E, F, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, recording the admitted dyadic center, dyadic radius, positive tolerance, and endpoint comparisons for the requested interval.*
2. *L is the* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *row, locating the same request inside a rational interval window before any Real-facing read occurs.*
3. *I is the finite dyadic interval packet: a displayed lower endpoint, upper endpoint, radius, center, and containment witness over D and L.*
4. *C is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *row that stores the finite family of dyadic intervals covering the inspected rational window.*
5. *R is the refinement ledger, recording finite subinterval choices, containment checks, radius shrinkage, and common-refinement comparisons inside the cover row C.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is opened only after D, L, I, C, R have displayed the dyadic interval packet and its finite refinement route.*
7. *F is the consumer handoff row for L10 Real and Completion routes; it records that local basis reads pass through the displayed interval packet before any dense, cover, or completion-facing consumer sees a basis element.*

8. *H records componentwise* hsame *transport for dyadic endpoints, rational locator rows, interval packets, cover rows, refinement rows, Real seals, consumer handoffs, provenance, and local naming.*
9. *K records displayed* Cont *replay from a local Real request through D, L, I, C, R and then to E, F.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDifferentiationBasis}^{\uparrow}}$ *row.*

The classifier compares accepted basis packets only through these rows. It has no coordinate for a Vitali selector, an ambient sigma-algebra, a measure derivative, quotient real equality, countable choice, or Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicDifferentiationBasisUp.DyadicDifferentiationBasisTasteGate single carrier alignment`

Theorem 7063.2 (Dyadic differentiation basis NameCert obligations)**.** *For every accepted* DyadicDifferentiationBas *carrier*

$$B = (D, L, I, C, R, E, F, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicDifferentiationBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic endpoint and radius admission, rational interval location, finite dyadic interval packet formation, finite cover admission, refinement exactness, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, L10 consumer handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The analytic rows are exactly the dyadic ledger D, rational locator L, interval packet I, finite cover C, refinement ledger R, terminal Real seal E, and consumer handoff F.

The dyadic endpoint and radius data are read first. The row D fixes the scale and endpoint comparisons, while L locates the request inside a rational interval window before any basis element can be exported.

The interval and cover rows then provide the basis surface. The packet row I gives a displayed dyadic interval, and C records the finite cover family in which refinements are searched.

The refinement ledger R is the only row that may shrink or compare basis intervals. It records finite subinterval choices and common-refinement checks before the terminal seal E and L10 handoff F are public.

Finally H, K, P, N transport, replay, source, and name the same route. Since the carrier has no row for Vitali selection, measure derivatives, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7063.3 (Dyadic differentiation basis finite refinement)**.** *Every accepted* $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$ *read that refines a local Real or Completion request factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RationalIntervalLocator}^{\uparrow} \longmapsto \mathsf{DyadicIntervalCover}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For a carrier $B = (D, L, I, C, R, E, F, H, K, P, N)$, the consumer first reads the dyadic endpoint and radius ledger D, then the rational locator L, then the interval packet I, then the finite cover and refinement rows C, R, and only then the terminal Real seal E and L10 handoff F.

Proof. Begin with the carrier projection of Definition 7063.1. The request has no public basis element before D, L, I, C, R are displayed.

Read the dyadic and rational rows first. The dyadic ledger D fixes the scale of the endpoint and radius comparisons, and the rational locator L places the same request in a displayed rational interval window.

Next open the finite cover. The interval packet I is one displayed dyadic interval over the locator row, and the cover row C is the finite family from which refinements are selected.

Use the refinement ledger R for shrinkage and common-refinement comparisons. Because R records only finite subinterval choices inside C, the route remains a finite dyadic basis route rather than a Vitali or measure-theoretic selection.

The terminal Real seal E and handoff F are therefore downstream of the finite basis packet. The structural rows H, K, P, N preserve that order and add no quotient real equality, countable choice, ambient completeness, or measure derivative coordinate. □

Falsifiable prediction: Every accepted $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$ L10 Real or Completion read exposes a DyadicRatCore endpoint-radius ledger, a RationalIntervalLocator row, a finite dyadic interval packet, a DyadicIntervalCover row, a refinement ledger, a RealUp seal, a consumer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any basis element is public.

Independence witness: dyadicratcore, rationalintervallocator, dyadicintervalcover, real: DyadicDifferentiationBas is not bijective to any listed sibling because it jointly carries endpoint-radius ledgers, locator rows, finite interval packets, finite cover rows, refinement ledgers, Real seal handoff, L10 consumer routing, transport, replay, provenance, and local naming as one finite basis certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDifferentiationBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDifferentiationBasis}^{\uparrow}$ packet is generated from DyadicRatCore endpoint and radius rows, a RationalIntervalLocator row, a finite dyadic interval packet, a DyadicIntervalCover row, finite refinement rows, a Real seal, an L10 consumer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDifferentiationBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7063.1, Definition 7063.2, and Definition 7063.3 seed the finite dyadic basis route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDifferentiationBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic differentiation, Vitali selection, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint-radius exactness, rational locator compatibility, finite cover refinement, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicDifferentiationBasisUp.

Chapter 7064

A Concrete Naming Certificate for $\mathsf{DyadicDenominatorClearance}^{\uparrow}$

Following Bishop and Bridges on constructive real-number calculation and the Hardy–Wright denominator-clearing discipline for rational arithmetic, the $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ packet records the finite common-scale step used before regular Cauchy consumers compare dyadic approximants. It sits after the rational and dyadic source rows of chapter 291, the signed normal-form rows of chapter 1865, the valuation ledger of chapter 2106, the finite stream interface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name boundary of chapter 38. The packet names common-denominator clearing as a displayed finite route; it does not import quotient rational equality, host integer normalization, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7064.1 (Dyadic denominator clearance carrier)**.** *A* $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, V, M, E, T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational and dyadic source row, read through* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, exposing only the finitely presented numerator and dyadic-scale cells admitted by the source certificate.*
2. *V is the* $\mathsf{DyadicValuation}^{\uparrow}$ *row that records the visible dyadic exponents and the divisibility checks used to compare source scales.*
3. *M is the signed mantissa row, read through* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$*, carrying sign, finite mantissa, and zero-status data for each displayed source cell.*
4. *E is the common exponent row, generated from the displayed V cells and naming the shared dyadic scale to which the visible source cells are lifted.*
5. *T is the dyadic tolerance row, recording that the common exponent is fine enough for the current local tolerance request.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists exactly the finite source windows whose entries have been lifted to the common exponent.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that consumes the common-scale windows without receiving a quotient rational object or a host sequence function.*

8. *H records componentwise* hsame *transport for source, valuation, mantissa, exponent, tolerance, stream-window, regular handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through $Q, V, M, E, T, S, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicDenominatorClearance}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicValuation}^{\uparrow}$, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDenominatorClearance}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, V, M, E, T, S, R, H, C, P, N$. It has no coordinate for quotient rational equality, host integer normalization, trichotomy, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7064.2 (Dyadic denominator clearance NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ *carrier $D = (Q, V, M, E, T, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicDenominatorClearan}}$ *surface consists exactly of rational and dyadic source exposure, valuation exposure, signed mantissa exposure, common exponent admission, dyadic tolerance admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted carrier. The only arithmetic source rows are Q, V, M, E, T, so every denominator-clearing read starts with finitely displayed rational and dyadic cells, their visible valuations, their signed mantissas, a named common exponent, and the tolerance row that licenses that exponent.

The stream and regular rows are downstream of that arithmetic surface. Row S records the finite windows lifted to the common scale, and row R is available only after those windows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since quotient rational equality, host integer normalization, trichotomy, selected representatives, countable choice, and ambient Real completeness are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7064.3 (Dyadic denominator clearance scale locality)**.** *Every common-scale comparison exposed by an accepted* $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ *packet factors through the finite route*

$$Q, V, M \longmapsto E, T \longmapsto S, R.$$

The common exponent row E is determined only from displayed valuation and mantissa rows, and the resulting comparison exports no quotient rational equality, no host normalization algorithm, no trichotomy, and no selected representative.

Proof. Read Q, V, M first. These rows expose the source cells, the dyadic exponents visible for the current comparison, and the signed mantissas attached to those cells.

The common exponent row E is then admitted as a finite row over the displayed valuation data, and T checks that this exponent is fine enough for the local tolerance. No unlisted normalization routine is consulted.

Finally S and R consume only the lifted finite windows. Because the refused objects are not carrier rows, the scale-clearing comparison remains local to the displayed BEDC data. □

Theorem 7064.4 (Dyadic denominator clearance regular handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ *packet receives regular rational data only through displayed common-scale* $\mathsf{StreamName}^{\uparrow}$ *windows and the structural rows* H, C, P, N*. The handoff cannot be read as a proof of quotient rational equality, a host integer normal form, a trichotomy decision, a selected representative, a countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 7064.3. It places every common-scale comparison downstream of the finite source, valuation, mantissa, exponent, and tolerance rows.

The $\mathsf{StreamName}^{\uparrow}$ row S records exactly the finite windows produced by that route. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer therefore reaches R only after the common-scale windows have been displayed.

Transport, replay, provenance, and local naming keep the handoff attached to the same carrier. Since the excluded principles are not rows of D, the regular handoff has no route by which to export them. □

Falsifiable prediction: Every $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ regular-rational read factors through rational and dyadic source rows, DyadicValuation rows, SignedDyadicNormalForm mantissas, a common exponent row, dyadic tolerance, StreamName windows, RegSeqRat handoff, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, signed-dyadic-normal-form, dyadic-valuation, streamname, regseqrat, real: $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source dyadic cells, visible valuations, signed mantissas, common exponent clearance, tolerance admission, finite stream windows, and regular rational handoff as one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDenominatorClearance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicDenominatorClearance}^{\uparrow}$ packet is generated from rational and dyadic source rows, DyadicValuation rows, SignedDyadicNormalForm mantissas, a common exponent row, dyadic tolerance, StreamName windows, RegSeqRat handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDenominatorClearance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7064.1, Definition 7064.2, Definition 7064.3, and Definition 7064.4 seed the finite denominator-clearing route from DyadicRatCore valuation data and signed normal forms to StreamName windows and RegSeqRat consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDenominatorClearance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, host integer normalization, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source exposure, valuation exposure, signed mantissa exposure, common exponent admission, dyadic tolerance exactness, StreamName window exposure, RegSeqRat handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicDenominatorClearanceUp.

Chapter 7065

A Concrete Naming Certificate for $\mathsf{FiniteDifferenceTable}^{\uparrow}$

Following Newton's finite-difference calculus and Bishop–Bridges constructive analysis, the $\mathsf{FiniteDifferenceTable}^{\uparrow}$ packet records a finite triangular table of successive differences as displayed BEDC rows. It sits over the finite window interface of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the finite-difference surface of chapter 5135, the terminal real-name seal of chapter 38, and the completion-facing boundary of chapter 95. The packet supplies calculus, Real, and completion consumers with a finite table surface; it does not import a host derivative operator, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7065.1 (Finite difference table carrier)**.** *A* $\mathsf{FiniteDifferenceTable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row listing the finite source-name windows whose values seed the first table row.*
2. *W is the window ledger recording the finite index interval, row height, mesh step, and endpoint reuse schedule for the displayed table.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every source value and every derived table entry.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for adjacent subtraction, row-to-row error budgeting, and terminal seal precision.*
5. *L is the table ledger, listing the displayed rows*

   $$L_0, L_1, \ldots, L_k$$

   where L_0 is the read source row and each L_{i+1} is obtained from adjacent finite differences in L_i under the same dyadic budget.
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the table entries and the completion-facing readback endpoints.*
7. *H records componentwise* hsame *transport for source windows, regular rational readbacks, dyadic tolerance rows, table rows, Real seals, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a table request through*

$$S, W \longmapsto R \longmapsto D \longmapsto L \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FiniteDifferenceTable}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDifferenceTable}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source windows, regular readbacks, dyadic budgets, table rows, Real seals, transport, replay, provenance, and local naming rows. It has no coordinate for a host derivative operator, quotient stream equality, selected limiting process, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteDifferenceTableUp.FiniteDifferenceTableTasteGate single carrier alignment`

Theorem 7065.2 (Finite difference table NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDifferenceTable}^{\uparrow}$ *carrier*

$$T = (S, W, R, D, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDifferenceTable}^{\uparrow}}$ *surface consists exactly of source-window admission, table-window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for every displayed entry,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness for adjacent differences, table-ledger exactness for $L_0, \ldots, L_k$, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, completion-facing endpoint exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.FiniteDifferenceTableUp.FiniteDifferenceTableCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 7065.1. The rows S and W are the only source of inspected names, finite windows, mesh requests, row height, and endpoint reuse.

Proceed by finite induction on the displayed table rows. The base row L_0 is read from S through R under the dyadic budget D. In the induction step, L_{i+1} is obtained only from adjacent entries of L_i, again through R and D, so no host derivative operator or selected limiting process is introduced.

The terminal rows E, H, C, P, N seal, transport, replay, source, and name the same finite table route. Since the carrier has no coordinate for quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local certificate surface. □

Theorem 7065.3 (Finite difference table RegSeqRat handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{FiniteDifferenceTable}^{\uparrow}$ *packet receives the table only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R has read each displayed source entry and each displayed adjacent-difference entry under the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget D. The handoff reaches* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *only through the finite route*

$$S, W \longmapsto R \longmapsto D \longmapsto L \longmapsto E,$$

not through derivative semantics, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7065.2. They force every accepted table read to expose the source windows, table-window ledger, regular rational readback, dyadic tolerance row, table ledger, and Real seal before a downstream consumer can use the result.

Use finite induction on the displayed table rows. The base row is a finite $\mathsf{RegSeqRat}^{\uparrow}$ readback of the source windows. Each successor row is a finite adjacent-difference pass through the same readback and dyadic tolerance rows, so the table never becomes an infinite stream quotient or a selected limit.

Finally read the handoff rows. The $\mathsf{Real}^{\uparrow}$ seal and the $\mathsf{Completion}^{\uparrow}$ endpoint exposure are downstream of L and preserve the same finite route through H, C, P, N. Thus a Real or completion consumer receives only the displayed table data and not a host derivative, countable-choice schedule, quotient equality, selected limiting process, or ambient completeness claim. □

Theorem 7065.4 (Finite difference table calculus and completion route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Completion}^{\uparrow}$ *consumer citing an accepted* $\mathsf{FiniteDifferenceTable}^{\uparrow}$ *packet must first read the source windows* S*, the table-window ledger* W*, the regular rational readback row* R*, the dyadic tolerance row* D*, and the full table ledger* $L_0, \ldots, L_k$*. Only then may it consume the terminal seal* E *with transport, replay, provenance, and local naming rows still visible.*

Proof. Project the table carrier. The route to any calculus, Real, or completion consumer is displayed as $S, W, R, D, L, E, H, C, P, N$.

Apply the RegSeqRat handoff. It places every source entry and adjacent-difference entry behind R and D before the terminal seal can be read.

The finite induction on $L_0, \ldots, L_k$ accounts for the whole table ledger. The structural rows then transport and replay the same finite object, so the downstream route cannot add derivative operators, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDifferenceTable}^{\uparrow}$ read factors through source StreamName windows, RegSeqRat readbacks, DyadicRatCore adjacent-difference budgets, displayed table rows, a RealUp seal, completion-facing endpoint exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: finite-difference, streamname, regseqrat, dyadicratcore, real, completion: $\mathsf{FiniteDifferenceTable}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source windows, finite table ledger, rowwise adjacent-difference route, rational readback, dyadic budgets, Real seal, completion endpoint exposure, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDifferenceTable}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteDifferenceTable}^{\uparrow}$ packet is generated from finite StreamName source windows, RegSeqRat readbacks, DyadicRatCore adjacent-difference budgets, a displayed table ledger, a RealUp seal, completion-facing endpoint exposure, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteDifferenceTable}^{\uparrow}}$, seed)

Scope closed: Definition 7065.1, Definition 7065.2, Definition 7065.3, and Definition 7065.4 seed the finite table carrier, certificate surface, rational readback handoff, and calculus/Real/completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteDifferenceTable}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host derivative operators, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window admission, rowwise difference induction, RegSeqRat readback exactness, DyadicRatCore tolerance exactness, Real sealing, completion endpoint exposure, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDifferenceTableUp.

Chapter 7066

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ packet records the finite window-selection surface used when a regular Cauchy name is inspected at a displayed dyadic tolerance. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window route of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the tail selector of chapter 6941. The packet selects one finite inspection window already justified by the displayed regular-Cauchy data; it is not a quotient stream equality, a selected limit, a host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7066.1 (Regular Cauchy window selector carrier)**.** *A* $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the finite precision demand.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row inspected for that demand.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to S.*
4. *T is the* $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ *tail-window row certifying that S is admitted by the displayed regular-Cauchy route.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R, T have been displayed.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, tail-selector, and seal rows.*
7. *C records displayed* Cont *replay of the same finite selection route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, T, E, H, C, P, N$. It contains no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7066.2 (Regular Cauchy window selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ *carrier* $W = (D, S, R, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowSelector}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission, finite stream-window exposure, regular rational readback, tail-window selection, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7066.1. The rows $D, S, R, T, E, H, C, P, N$ are the whole selector packet, so the local naming surface is finite and displayed.

The selection route is read in order. The dyadic row fixes the demanded tolerance, the stream row exposes the inspected finite window, and the regular rational row reads the associated rational data.

The tail-window row T then records that this finite window is admitted by the existing $\mathsf{RegularCauchyTailWindowSelector}^{\uparrow}$ route. The rows E, H, C, P, N seal, transport, replay, source, and name the same finite route, leaving no separate coordinate for a choice schedule or hidden limit. □

Theorem 7066.3 (Regular Cauchy window selector Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ *packet reads the seal row* E *only after the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{RegularCauchyTailWindowSelector}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

has exposed the rows $D, S, R, T, E, H, C, P, N$*. The seal is a finite BEDC-native window-selection certificate for downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *arguments; it does not claim quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7066.2. They place the dyadic tolerance, stream window, regular rational readback, and tail-window selector before any Real-facing row can be consumed.

Read the tail selector through chapter 6941. That sibling supplies the finite tail admission used by T, while the present packet binds the admitted window to the single Real seal row E.

The carrier is exhausted by $D, S, R, T, E, H, C, P, N$. Since no row names a quotient stream, selected limit, host real equality, countable-choice enumeration, trichotomy proof, or ambient completeness theorem, the seal remains a finite window-selection certificate. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ consumer is refuted if it reaches a Real-facing seal before displaying the dyadic tolerance, StreamName window, RegSeqRat readback, RegularCauchyTailWindowSelector row, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, dyadicratcore, regseqrat, real, regular-cauchy-tail-window-selector: $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ is not bijective to any listed sibling because it binds a precision demand, one finite stream window, regular rational readback, tail-window admission, Real sealing, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted window-selection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowSelector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyWindowSelector}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RegularCauchyTailWindowSelector row, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyWindowSelector}\uparrow}$, seed)

Scope closed: Definition 7066.1, Definition 7066.2, and Definition 7066.3 seed the finite window-selection route for regular Cauchy names.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyWindowSelector}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, dyadic tolerance coverage, RegSeqRat readback, tail-window admission, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindowSelectorUp.

Chapter 7067

A Concrete Naming Certificate for $\mathsf{F\Sigma Set}^{\uparrow}$

Following descriptive set theory and classical topology, the $\mathsf{F\Sigma Set}^{\uparrow}$ packet records the traditional countable-union-of-closed-sets interface as finite BEDC NameCert data. It sits after the generated-set boundary of chapter 3384, the zero-locus surface of chapter 1858, the dual open-schedule route of chapter 3547, and the indexed open and closed-row discipline of chapter 88. The packet names the finite surface that Borel hierarchy consumers need before they read a displayed closed-row schedule; it does not import host subsets, quotient set equality, a selected countable enumeration, ambient countable choice, or a classical descriptive-set theorem.

Definition 7067.1 (F-sigma set carrier)**.** *An* $\mathsf{F\Sigma Set}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, Z, G, S, W, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the named open and closed-row vocabulary visible to the current consumer.*
2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set row that receives the handoff after the displayed closed schedule has been inspected.*
3. *Z is the* $\mathsf{ZeroSet}^{\uparrow}$ *closed-locus row used when a closed component is supplied as the zero set of a displayed continuous* $\mathsf{Real}^{\uparrow}$*-valued map.*
4. *G is the* $\mathsf{GDeltaSet}^{\uparrow}$ *sibling row used only as a dual-route comparison for open schedules and Borel handoff discipline.*
5. *S is the countable closed-row schedule. Each visible prefix names the closed components, the schedule position, and the topology row over which that component is read.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row recording exactly the schedule positions, component indices, and finite membership queries visible before the Borel handoff.*
7. *H records componentwise* hsame *transport for the topology source, Borel row, zero-set row, GDeltaSet sibling row, schedule, windows, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a topology or Borel query through T, S, W before the generated-set row B is consumed.*

9. *P is the* Pkg *provenance over* $\mathsf{F\Sigma Set}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{ZeroSet}^{\uparrow}$, $\mathsf{GDeltaSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{F\Sigma Set}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, Z, G, S, W, H, C, P, N$. It has no coordinate for host powersets, quotient equality of subsets, selected countable enumerations, countable choice, or ambient Borel hierarchy completeness.

Lean4 (statement) `BEDC.Derived.FSigmaSetUp`

Theorem 7067.2 (F-sigma set NameCert obligations)**.** *For every accepted* $\mathsf{F\Sigma Set}^{\uparrow}$ *carrier* $F = (T, B, Z, G, S, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{F\Sigma Set}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, Borel row, zero-set row, GDeltaSet sibling row, closed-row schedule, finite windows, transport, replay, provenance, and local naming;*
2. *closed-schedule exposure, requiring every inspected union stage to appear as a displayed finite prefix of S before it reaches B;*
3. *finite-window stability, requiring each schedule position, component index, and membership query to be read through W;*
4. *closed-component readback, requiring every scheduled component to pass through the topology source T and, when supplied by a zero-locus presentation, through the* $\mathsf{ZeroSet}^{\uparrow}$ *row Z;*
5. *Borel handoff, requiring the generated-set row B to consume only the displayed route through T, S, W and the closed-component readback;*
6. *sibling compatibility, requiring the* $\mathsf{GDeltaSet}^{\uparrow}$ *row G to serve only as the dual open-schedule comparison surface and not as a complement-selection principle;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N;*
8. *ledger refusal, excluding host powersets, selected enumerations, countable choice, quotient set equality, classical complement selection, and ambient descriptive-set theorems.*

No $\mathsf{F\Sigma Set}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.FSigmaSetUp.FSigmaSetCarrier namecert obligations`

Proof. Project the carrier of Definition 7067.1. The tuple displays exactly the topology source, Borel row, zero-set row, GDeltaSet sibling row, countable closed-row schedule, finite windows, transport, replay, provenance, and local naming.

The schedule and window obligations are read through S and W. A consumer sees only a finite prefix of the countable closed schedule, and every component index and membership query is inspected through the same displayed finite window.

Closed-component readback is forced by T and Z. The topology source supplies the closed-row vocabulary, and the zero-set row supplies the zero-locus presentation when the component is offered in that form.

Finally H, C, P, N transport, replay, source, and name the same route. Since no listed row stores a host powerset, selected enumeration, countable-choice selector, quotient set equality, classical complement selector, or ambient descriptive-set theorem, the enumerated obligations exhaust the local NameCert surface. □

Theorem 7067.3 (F-sigma set Borel handoff)**.** *Every Borel consumer of an accepted* $\mathsf{F\Sigma Set}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{ZeroSet}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BorelSet}^{\uparrow}.$$

More precisely, the consumer first reads the topology source row T*, then the closed-row schedule and finite windows* S, W*, then the zero-set row* Z *when a scheduled component is presented as a zero locus, and only then the generated-set row* B*. The exported object is a finite NameCert packet for the displayed countable closed-row route; it is not a host subset, a chosen enumeration, a complement-selected* $\mathsf{GDeltaSet}^{\uparrow}$ *packet, an ambient descriptive-set theorem, or a quotient set equality.*

Proof. Begin with a Borel consumer of the accepted packet. Carrier admission exposes the topology source row T, but the generated-set row B is not available until the closed schedule and finite windows have been displayed.

Replay the continuation row C. The read moves through S and W, so every countable-union stage is consumed only as a finite visible prefix with finite component and query observations.

When a closed component is presented by a zero locus, the read passes through Z and therefore through the zero-set discipline of chapter 1858. The sibling $\mathsf{GDeltaSet}^{\uparrow}$ row G supplies only a comparison route for the dual schedule shape; it does not choose complements.

The remaining rows H, P, N preserve, source, and name the same route. Therefore every Borel consumer factors through the displayed topology, zero-set, schedule, and window route, with no host subset, selected enumeration, complement-selected GDeltaSet packet, ambient descriptive-set theorem, or quotient set equality exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{F\Sigma Set}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{F\Sigma Set}^{\uparrow}}$, seed)

Scope closed: Definition 7067.1, Definition 7067.2, and Definition 7067.3 seed the finite F-sigma route over topology, zero-set, GDeltaSet, StreamName, Borel, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{F\Sigma Set}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close host powersets, selected countable enumerations, countable choice, quotient set equality, classical complement selection, ambient descriptive-set theorems, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-schedule exposure, finite-window stability, closed-component readback, Borel handoff, sibling compatibility, transport, replay, provenance, local naming, and ledger-refusal proofs for BEDC.Derived.FSigmaSetUp.

Constructive story: An $\mathsf{F\Sigma Set}^{\uparrow}$ packet is generated from a $\mathsf{Topology}^{\uparrow}$ source row, a $\mathsf{BorelSet}^{\uparrow}$ row, a $\mathsf{ZeroSet}^{\uparrow}$ component row, a $\mathsf{GDeltaSet}^{\uparrow}$ sibling row, a countable closed-row schedule, finite StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7068

A Concrete Naming Certificate for $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$

Following Birkhoff and James on normed-space orthogonality, the $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ packet records the finite norm-variation surface by which one displayed vector is orthogonal to another in the Birkhoff–James sense. It sits after the normed and Hilbert source rows of chapter 89 and chapter 91, the metric and projection rows of chapter 57 and chapter 2356, and the scalar readback rows of chapter 38. The packet names only a finite norm-comparison ledger for scalar perturbations; it does not import a host inner-product characterization, a selected projection map, quotient equality of vectors, trichotomy, or ambient completeness.

Definition 7068.1 (BirkhoffJamesOrthogonality carrier)**.** *A* $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, H, x, y, \Lambda, L, R, M, Q, T, C, P, N)$$

with the following displayed rows:

1. *X is the normed-source row, exposing the carrier, addition, scalar action, zero vector, and norm read through* $\mathsf{Norm}^{\uparrow}$*.*
2. *H is the optional Hilbert-source comparison row, present only when an inner-product route is being compared with the norm-variation route.*
3. *x and y are the two displayed vector rows whose relation is being named.*
4. *Λ is the scalar perturbation window, listing the finite scalar samples and tolerance requests used to test $x + \lambda y$.*
5. *L is the norm-comparison ledger, recording the displayed inequalities $\|x\| \leq \|x + \lambda y\|$ for the perturbation window and its stated tolerance refinements.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the norm values and comparison outcomes in L.*
7. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row induced by the norm source, keeping every distance read tied to the same displayed norm ledger.*
8. *Q is the projection-facing handoff row, recording what a* $\mathsf{MetricProjection}^{\uparrow}$ *consumer may read from the orthogonality ledger before adding its own endpoint evidence.*
9. *T records componentwise* hsame *transport for source, Hilbert comparison, vector, scalar-window, norm-ledger, Real readback, metric, projection-handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an orthogonality request through* $X, H, x, y, \Lambda, L, R, M, Q, T, P, N$.
11. *P is the* Pkg *provenance over* BirkhoffJamesOrthogonality$^{\uparrow}$, Norm$^{\uparrow}$, Hilbert$^{\uparrow}$, MetricProjection$^{\uparrow}$, Real$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{BirkhoffJamesOrthogonality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host orthogonality predicate, an inner-product theorem, a selected best approximation, quotient vector equality, trichotomy of scalar comparisons, countable choice over perturbations, or ambient completion.

Theorem 7068.2 (BirkhoffJamesOrthogonality NameCert obligations)**.** *For every accepted* BirkhoffJamesOrthogon *carrier $O = (X, H, x, y, \Lambda, L, R, M, Q, T, C, P, N)$, the local* NameCert$_{\text{BirkhoffJamesOrthogonality}^{\uparrow}}$ *surface consists exactly of normed-source exposure, optional Hilbert comparison exposure, two-vector exposure, scalar perturbation-window exposure, norm-comparison ledger exactness, Real readback, induced metric readback, projection-facing handoff, componentwise transport, displayed replay, provenance, and local naming. A Birkhoff–James orthogonality read is admitted only after x, y, Λ, L, and R are visible in the same packet.*

Proof. Project the carrier of Definition 7068.1. The source norm, optional Hilbert comparison, and two vector rows are exactly X, H, x, y.

The scalar-variation data are supplied by Λ and L. The ledger L is the only row that records the comparison between $\|x\|$ and $\|x + \lambda y\|$, and R is the only row that exports these comparisons as Real-facing reads.

The metric and projection-facing rows M, Q remain downstream of the same norm ledger. They may expose distance and projection-consumer data, but they do not add a nearest-point selector or replace the norm-variation test by a host projection theorem.

Transport through T preserves the displayed coordinates, while C, P, N replay, source, and name the same packet. Since the refused objects are not carrier rows, the listed fields exhaust the local NameCert obligations. □

Theorem 7068.3 (BirkhoffJamesOrthogonality norm-variation handoff)**.** *Every projection-facing, Hilbert-facing, metric-facing, or Real-facing consumer of an accepted* BirkhoffJamesOrthogonality$^{\uparrow}$ *packet factors through the finite norm-variation route*

$$X, x, y, \Lambda, L, R \longmapsto M, Q.$$

The consumer receives a displayed scalar perturbation window, norm-comparison ledger, Real readback, induced metric row, and projection-facing handoff. It receives no host inner-product characterization, selected projection endpoint, quotient vector equality, trichotomy branch, countable-choice perturbation family, or ambient completeness theorem.

Proof. Start with an accepted consumer read. By the obligation surface, the read must first expose the normed source X, the two vectors x, y, the perturbation window Λ, the comparison ledger L, and the Real readback R.

The Hilbert row H can only compare an existing Hilbert presentation with that same norm-variation ledger. It is not a replacement for the ledger and does not export the inner-product characterization as a separate host theorem.

The metric and projection stages are M and Q. They read the norm ledger as distance data and hand it to projection-facing consumers, but endpoint selection belongs to the downstream MetricProjection$^{\uparrow}$ packet.

Thus every public handoff remains inside the displayed route and cannot project any of the refused host principles from the $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ read displays a normed source, optional Hilbert comparison row, two vector rows, a scalar perturbation window, a norm-comparison ledger, Real readback, induced Metric row, projection-facing handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row before it is consumed.

Independence witness: norm, hilbert, metricprojection, real, metric: $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two vectors, scalar perturbation windows, norm-comparison ledgers, Real readback, metric readback, projection-facing handoff, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}$ packet is generated from a Norm source, optional Hilbert comparison row, two vector rows, scalar perturbation windows, norm-comparison ledgers, Real readback, induced Metric rows, MetricProjection-facing handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7068.1, Definition 7068.2, and Definition 7068.3 seed the finite Birkhoff–James norm-variation packet through displayed normed-source, two-vector, scalar-window, comparison-ledger, Real, Metric, and projection-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffJamesOrthogonality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host inner-product characterization, selected projection endpoint, quotient vector equality, trichotomy, countable choice over scalar perturbations, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-window exposure, norm-comparison ledger exactness, Real readback, Metric readback, projection-facing handoff, transport, replay, provenance, and local NameCert registration for BEDC.Derived.BirkhoffJamesOrthogonalityUp.

Chapter 7069

A Concrete Naming Certificate for $\mathsf{Arc}^{\uparrow}$

Following Munkres and Bishop–Bridges, an arc is read as a certified image of a closed interval under a displayed continuous map. The $\mathsf{Arc}^{\uparrow}$ packet is anchored by the closed interval source of chapter 1591, the map graph and modulus rows of chapter 130, the compact-metric source of chapter 129, the connected-space boundary of chapter 2353, and the terminal real-name seal of chapter 38. It records the interval-image route through finite BEDC rows; it is not a host subset of a topological space, a quotient path image, or a selected point-set equality.

Definition 7069.1 (Arc carrier)**.** *An* $\mathsf{Arc}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, M, K, C, R, H, T, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, naming the endpoint interval, order witnesses, and endpoint real seals inspected by the packet.*
2. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording the displayed map from I into the target metric carrier together with the finite continuity or modulus ledger used for image reads.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *target-source row through which finite nets, distance tests, and compact image windows are exposed.*
4. *C is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *image-continuity row, recording the connected interval-image boundary visible to continuum-facing consumers.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for endpoints, radii, modulus budgets, and target-distance seals.*
6. *H records componentwise* hsame *transport for the interval, map, compact-metric, connectedness, real-readback, provenance, and naming rows.*
7. *T records displayed* Cont *replay from interval requests through map evaluation, compact-metric windows, connected-image reads, and real seals.*
8. *P is the* Pkg *provenance over* $\mathsf{Arc}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Arc}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, K, C, R, H, T, P, N$. It has no coordinate for host subset equality, quotient path images, arbitrary open-cover compactness, selected representatives, or ambient real completeness.

Theorem 7069.2 (Arc NameCert obligations)**.** *For every accepted $\mathsf{Arc}^{\uparrow}$ carrier $A = (I, M, K, C, R, H, T, P, N)$, the local $\mathsf{NameCert}_{\mathsf{Arc}^{\uparrow}}$ surface consists exactly of closed-interval source admission through I, displayed continuous-map admission through M, compact-metric image-window admission through K, connected-image exposure through C, real endpoint and tolerance readback through R, componentwise transport through H, displayed replay through T, provenance through P, and local naming through N.*

Every arc-facing consumer must read these rows before it may treat the packet as interval-image data. No obligation is discharged by a host point-set topology theorem, quotient equality of images, selected image representatives, arbitrary open covers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. Project the carrier of Definition 7069.1. The closed interval row I and the continuous-map row M are the only source rows through which an interval image can be read.

The compact and connected image faces are then finite rows of the same packet. The row K supplies compact-metric windows and distance tests for the target image, while C records the connected-image boundary consumed by continuum-facing readers. The row R supplies the endpoint and tolerance seals needed by both faces.

Transport and replay remain componentwise. The rows H and T move and expose exactly the interval, map, compact-metric, connectedness, and real-readback rows, while P and N record provenance and local naming.

Since the carrier has no coordinate for host subset equality, quotient image equality, arbitrary open-cover compactness, selected representatives, countable choice, or ambient real completeness, the NameCert obligations are exhausted by the displayed rows. □

Theorem 7069.3 (Arc interval image nonescape)**.** *Every public interval-image read of an accepted $\mathsf{Arc}^{\uparrow}$ packet factors through the displayed route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ConnectedSpace}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the rows $I, M, K, C, R, H, T, P, N$. The route exposes a closed interval, a displayed continuous map, compact-metric image windows, connected-image data, and real tolerance seals; it exports no host image subset, quotient path-image equality, selected representative, open-cover compactness theorem, or ambient real-completeness witness.

Proof. Start from the carrier and read I before M. The packet names an interval-image object only after the closed interval and displayed continuous map are both visible.

Next read K, C, R. Compact-metric image windows, connected-image data, and real tolerance seals are downstream of the same map row, so a consumer cannot replace them by a detached host subset or by a quotient image object.

Finally use H, T, P, N. These rows transport, replay, source, and name the finite interval-image route without adding another image coordinate. Therefore every public interval-image read remains inside the displayed BEDC rows. □

Falsifiable prediction: Every accepted $\mathsf{Arc}^{\uparrow}$ read exposes a closed interval source, a displayed continuous-map row, compact-metric image windows, connected-image data, real endpoint and

tolerance seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before an arc-facing consumer can inspect the packet.

Independence witness: closedboundedinterval, continuousmap, compactmetric, connected-space, real: $\mathsf{Arc}^{\uparrow}$ is not bijective to any listed sibling because it binds interval-source rows, map rows, compact image windows, connected-image exposure, real readback, transport, replay, provenance, and local naming in one interval-image carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{Arc}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Arc}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a ContinuousMapUp display row, CompactMetricUp image windows, a ConnectedSpaceUp image boundary, RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Arc}^{\uparrow}}$, seed)

Scope closed: Definition 7069.1, Definition 7069.2, and Definition 7069.3 seed the finite interval-image boundary over closed intervals, displayed continuous maps, compact-metric windows, connected-image rows, real readback, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Arc}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host subset equality, quotient image equality, arbitrary open-cover compactness, selected representatives, countable choice, Lean-checked obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, continuous-map admission, compact image-window admission, connected-image exposure, real readback, transport, replay, provenance, and local naming for BEDC.Derived.ArcUp.

Chapter 7070

A Concrete Naming Certificate for $\mathsf{RationalCauchyTailBound}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalCauchyTailBound}^{\uparrow}$ packet records the finite rational tail-bound ledger used before a regular Cauchy read is handed to a Real or Completion consumer. It sits after the interval localization of chapter 4202, the dyadic tolerance core of chapter 291, the finite stream-window route of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name boundary of chapter 38. The packet names only the displayed tail-bound evidence; it is not quotient equality of rational streams, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7070.1 (Rational Cauchy tail-bound carrier)**.** *A* $\mathsf{RationalCauchyTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *source row, exposing the rational center, interval endpoints, and the finite index threshold at which the tail read begins.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the displayed dyadic radius, slack row, and comparison budget used to test late rational observations.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by L and budgeted by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational tail estimate obtained from the displayed windows and dyadic budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after L, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the locator, dyadic tolerance ledger, finite windows, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a requested precision threshold through L, D, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalCauchyTailBound}^{\uparrow}$, $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyTailBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, D, W, R, E, H, C, P, N$. It has no coordinate for quotient equality of rational streams, a selected real limit, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7070.2 (Rational Cauchy tail-bound NameCert obligations)**.** *For every accepted* RationalCauchyTailBour *carrier*

$$T = (L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCauchyTailBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational locator row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *locator exactness, requiring L to display the rational center, interval endpoints, and finite tail threshold that the bound is allowed to inspect;*
3. *dyadic tolerance exactness, requiring D to store the radius, slack, and comparison budget used for late observations;*
4. *window compatibility, requiring W to expose only the StreamName observations opened by L and budgeted by D;*
5. *regular readback exactness, requiring R to consume L, D, W before any Real seal is visible;*
6. *Real-seal nonescape, requiring E to be terminal with respect to the route L, D, W, R and to export no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy, ambient completeness theorem, Lean verification, or bridge data;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No rational Cauchy tail-bound obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7070.1. The accepted packet displays exactly the rational locator, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows.

The analytic clauses are forced by the route. The row L is the only coordinate that carries the rational center, interval endpoints, and finite tail threshold, while D is the only coordinate that carries the dyadic radius, slack, and comparison budget.

The window and readback clauses are downstream. The row W reads only the observations opened by L and budgeted by D, and R is the first row that repacks those finite observations as regular rational tail evidence.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, and bridge data are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7070.3 (Rational Cauchy tail-bound regular Real handoff)**.** *Every accepted rational tail-bound read that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer must factor through the finite route*

$$L \longmapsto D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives the rational locator row, dyadic tolerance ledger, finite StreamName windows, regular rational readback, terminal Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives no quotient equality of rational streams, selected limit, countable-choice schedule, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Project the accepted carrier $T = (L, D, W, R, E, H, C, P, N)$. The tail-bound data starts in L, and the only tolerance data available to the read is the dyadic ledger D.

The observation route is finite. The StreamName row W exposes the bounded windows licensed by L and D, and the RegSeqRat row R is the first point where those windows are repacked as regular rational tail evidence.

The Real-facing consumer reaches the terminal seal only through E and then through the structural rows H, C, P, N. These rows preserve and replay the same route; they do not add quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every $\mathsf{RationalCauchyTailBound}^{\uparrow}$ read into RegSeqRat, Real, or Completion factors through one RationalIntervalLocator row, one DyadicRatCore tolerance ledger, one StreamName finite-window row, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: rational-interval-locator, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalCauchyTailBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a rational interval locator, finite tail threshold, dyadic tolerance ledger, finite stream windows, regular rational readback, and the terminal RealUp seal that witnesses a bounded tail route rather than a completed quotient object.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCauchyTailBound}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalCauchyTailBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7070.1, Definition 7070.2, and Definition 7070.3 seed the finite rational tail-bound route through RationalIntervalLocator source data, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalCauchyTailBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of rational streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locator exactness, dyadic tolerance exactness, window compatibility, regular readback exactness, Real-seal nonescape, transport, replay, provenance, and handoff factorization for BEDC.Derived.RationalCauchyTailBoundUp.

Constructive story: A $\mathsf{RationalCauchyTailBound}^{\uparrow}$ packet is generated from one RationalIntervalLocator row, one DyadicRatCore tolerance ledger, one StreamName finite-window row, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7071

A Concrete Naming Certificate for $\mathsf{FinitePrefixBar}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bishop–Bridges on spreads, bars, and finite choice-sequence observation, the $\mathsf{FinitePrefixBar}^{\uparrow}$ packet records the constructive L10 interface by which a finite bar over admitted spread prefixes is named before any countable-name or real-facing consumer can read it. It sits after the spread-prefix carrier of chapter 1677, the spread-bar route of chapter 5976, and the finite fan packet of chapter 400. The packet is a finite certificate surface, not a completed branch, countable choice principle, quotient stream equality, fan compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness row.

Definition 7071.1 (Finite prefix bar carrier)**.** *A* $\mathsf{FinitePrefixBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, B, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *row, including the decidable prefix tree, StreamName schedule, and admitted finite-prefix window.*
2. *B is the* $\mathsf{BrouwerSpreadBar}^{\uparrow}$ *row restricted to the prefixes visible in S, with decidable bar decisions and monotone prefix-extension checks.*
3. *A is the* $\mathsf{FanTheorem}^{\uparrow}$ *comparison row, used only as a finite bounded-bar witness when the displayed spread is binary or finitely branching.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *read window through which countable-name consumers request the displayed prefixes.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to every analytic observation generated from the finite prefix window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed to real-facing consumers after the prefix, bar, stream-name, and regular-rational rows have already been displayed.*
7. *H records componentwise* hsame *transport for spread prefixes, bar decisions, fan comparison, stream windows, regular-rational readback, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite prefix-bar request through*

$$S, B, A \longmapsto W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FinitePrefixBar}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{BrouwerSpreadBar}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixBar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, A, W, R, E, H, C, P, N$. It has no coordinate for a completed branch, a selector over all bars, quotient stream equality, fan compactness as an ambient theorem, selected limits, or Real completeness.

Theorem 7071.2 (Finite prefix bar NameCert obligations)**.** *For every accepted* $\mathsf{FinitePrefixBar}^{\uparrow}$ *carrier*

$$F = (S, B, A, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitePrefixBar}^{\uparrow}}$ *surface consists exactly of carrier admission for the Brouwer spread prefix row, restricted spread-bar row, finite fan comparison row, StreamName window, RegSeqRat readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier agreement on the same rows; and nonescape requiring every public read to factor through $S, B, A, W, R, E, H, C, P, N$.*

Proof. Project the carrier of Definition 7071.1. The tuple exposes the spread prefix row S, the restricted bar row B, the finite fan comparison row A, the stream-name window W, the regular-rational readback R, and the terminal Real seal E before the structural rows are considered.

Classifier agreement is rowwise agreement over that same tuple. The spread component is read through chapter 1677, the finite bar component through chapter 5976, and the finite fan comparison through chapter 400.

The structural rows close the certificate surface. The transport row H compares only displayed coordinates, the continuation row C replays only the finite route, and P, N source and name the same packet. Since no extra coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 7071.3 (Finite prefix bar nonescape)**.** *Every countable-name, completion-facing, or real-facing read of an accepted* $\mathsf{FinitePrefixBar}^{\uparrow}$ *packet must first pass through the finite route*

$$S, B, A \longmapsto W \longmapsto R \longmapsto E, H, C, P, N.$$

The read receives only admitted spread prefixes, restricted bar decisions, finite fan comparison data, StreamName windows, RegSeqRat readbacks, a terminal Real seal, transport, replay, provenance, and local naming. It does not obtain a completed branch, countable choice, quotient stream equality, fan compactness as a global theorem, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a public read of the packet. The only route to the analytic rows passes through the displayed prefix and bar rows S, B and, when available, the finite fan comparison row A.

Replay the continuation row C. It opens the StreamName window W, then reads the regular-rational row R, and only after those rows have been displayed does it expose the terminal Real seal E.

Finally attach transport, provenance, and naming through H, P, N. The packet has no field for a branch selector, quotient stream equality, an external compactness theorem, selected limits, or Real completeness. Therefore every downstream read remains inside the stated finite route. □

Falsifiable prediction: Every accepted $\mathsf{FinitePrefixBar}^{\uparrow}$ public StreamName, RegSeqRat, Real, or Completion-facing read factors through a Brouwer spread prefix row, restricted Brouwer spread-bar row, finite FanTheorem comparison row, StreamName window, RegSeqRat readback, RealUp

seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: brouwer-spread, brouwer-spread-bar, fan-theorem, streamname, regseqrat, real: $\mathsf{FinitePrefixBar}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite spread-prefix admission, restricted bar decisions, bounded fan comparison, stream-name windows, regular-rational readback, terminal Real sealing, transport, replay, provenance, and local naming while refusing completed branches, countable choice, quotient streams, ambient compactness, selected limits, and Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixBar}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePrefixBar}^{\uparrow}$ packet is generated from Brouwer spread prefixes, restricted Brouwer spread-bar decisions, finite FanTheorem comparison data, StreamName windows, RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixBar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7071.1, Definition 7071.2, and Definition 7071.3 seed the constructive L10 finite-bar interface for countable-name and real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixBar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed branches, countable choice, quotient stream equality, fan compactness, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier agreement, finite-bar route exactness, StreamName and RegSeqRat handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FinitePrefixBarUp.

Chapter 7072

A Concrete Naming Certificate for $\mathsf{AxisCarryCriticalPair}^{\uparrow}$

Following Newman-style local confluence analysis for finite rewrite systems, the $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ packet records the displayed critical pair formed by two overlapping Zeckendorf carry windows. It sits between the carry classifier of chapter 544, the carry-normalization handoff of chapter 608, the normal-word boundary of chapter 278, the route-separation surface of chapter 1049, the diamond-route packet of chapter 1118, and the confluence packet of chapter 713. The packet names a finite local overlap only; it is not a kernel rewrite, a quotient by history sameness, a global normalizer, or an unbounded confluence theorem.

Definition 7072.1 (Axis carry critical-pair carrier)**.** *An* $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (s, \ell, r, j, n, V, B, H, C, P, N)$$

with the following displayed rows:

1. *s is the overlapping Zeckendorf source-window row containing two visible carry redexes.*
2. *ℓ is the left-first carry target row reached by applying the left window through* $\mathsf{ZCarry}^{\uparrow}$*.*
3. *r is the right-first carry target row reached by applying the right window through* $\mathsf{ZCarry}^{\uparrow}$*.*
4. *j is the shared join row obtained by normalizing the remaining visible carry window from both ℓ and r.*
5. *n is the terminal* $\mathsf{ZNormal}^{\uparrow}$ *row reached from j by the local carry-normalization handoff.*
6. *V is the Fibonacci value ledger, recording that s, ℓ, r, j, n display the same finite value sum.*
7. *B is the boundary row recording the non-*hsame *status of the two carry edges and the fact that the join is a route comparison rather than a history identification.*
8. *H records componentwise* hsame *transport for s, ℓ, r, j, n, V, B and the structural rows.*
9. *C records displayed* Cont *replay through the source window, the two one-step branches, the shared join, the normal target, the value ledger, the boundary row, provenance, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{AxisCarryCriticalPair}^{\uparrow}$*,* $\mathsf{ZCarry}^{\uparrow}$*,* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$*,* $\mathsf{ZNormal}^{\uparrow}$*,* $\mathsf{AxisCarryRouteSeparation}^{\uparrow}$*,* $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$*,* $\mathsf{AxisCarryConfluence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisCarryCriticalPair}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, \ell, r, j, n, V, B, H, C, P, N$. It has no coordinate for source-target history sameness, quotient normal forms, host arithmetic equality, unbounded critical-pair search, global confluence, Lean verification, or bridge checking.

Theorem 7072.2 (Axis carry critical-pair NameCert obligations)**.** *For every accepted* $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ *carrier*

$$K = (s, \ell, r, j, n, V, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AxisCarryCriticalPair}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the overlapping source window, two one-step carry branches, shared join, terminal normal row, value ledger, non-*hsame *boundary, transport, replay, provenance, and local naming;*
2. *left-branch exactness, requiring ℓ to be read only through the displayed left carry window of s;*
3. *right-branch exactness, requiring r to be read only through the displayed right carry window of s;*
4. *join exactness, requiring j to be reached by normalizing the remaining visible carry window from both ℓ and r;*
5. *value-ledger exactness, requiring V to record the same finite Fibonacci value for s, ℓ, r, j, n;*
6. *boundary nonescape, requiring B to keep both carry edges outside* hsame *and to export no quotient normal form, host arithmetic equality, unbounded search, global confluence, Lean verification, or bridge data;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No local critical-pair obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7072.1. The accepted packet displays exactly the overlapping source window, the two one-step carry targets, the shared join, the terminal normal row, the value ledger, the boundary row, transport, replay, provenance, and local naming.

The two branch clauses are forced by the route. The row ℓ is reachable only from the left carry window of s, while r is reachable only from the right carry window of s.

The join and value clauses are downstream. The row j is the common target of the remaining visible carry window from both branches, and V records the finite Fibonacci value read through all displayed windows.

Finally, B, H, C, P, N keep the comparison local. Since history-sameness collapse, quotient normal forms, host arithmetic equality, unbounded critical-pair search, global confluence, Lean verification, and bridge data are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7072.3 (Axis carry critical-pair local diamond)**.** *Every accepted* $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ *packet exposes the local diamond route*

$$s \longmapsto \ell \longmapsto j \qquad \text{and} \qquad s \longmapsto r \longmapsto j$$

before any $\mathsf{AxisCarryConfluence}^{\uparrow}$ *consumer may read the overlap as confluence evidence. The shared target j is accompanied by the terminal normal row n, the value ledger V, the non-*hsame *boundary B, and the structural rows H, C, P, N; the handoff exports no source-target history sameness, quotient normal form, global normalizer, unbounded critical-pair search, or host arithmetic equality.*

Proof. Start from the NameCert obligations of Definition 7072.2. They force any accepted read to expose $s, \ell, r, j, n, V, B, H, C, P, N$ as one finite packet.

The two branches are visible before the join. The left route reads s then ℓ then j, while the right route reads s then r then j; both routes are replayed through the same continuation row C and transported through H.

The confluence-facing handoff is therefore the displayed diamond plus the value and boundary rows. Since B keeps the carry edges outside hsame and no omitted endpoint is a carrier row, an $\mathsf{AxisCarryConfluence}^{\uparrow}$ consumer receives only the finite local critical-pair packet stated above. □

Falsifiable prediction: Every $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ overlap read exposes one source window, two one-step carry branches, one shared join, one terminal normal row, one Fibonacci value ledger, one non-hsame boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any confluence-facing handoff is visible.

Independence witness: zcarry, zeckendorf-carry-normalization, zeckendorf-normal, axis-carry-route-separation, axis-carry-diamond-route, axis-carry-confluence: $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the overlapping source, both one-step carry branches, the shared join, the terminal normal row, the value ledger, the non-hsame boundary, and the local confluence-facing handoff as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AxisCarryCriticalPair}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisCarryCriticalPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7072.1, Definition 7072.2, and Definition 7072.3 seed the finite local critical-pair route for overlapping Zeckendorf carry windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxisCarryCriticalPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close source-target history sameness, quotient normal forms, host arithmetic equality, unbounded critical-pair search, global confluence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch exactness, join exactness, value-ledger exactness, boundary nonescape, transport, replay, provenance, and local-diamond handoff for BEDC.Derived.AxisCarryCriticalPairUp.

Constructive story: An $\mathsf{AxisCarryCriticalPair}^{\uparrow}$ packet is generated from one overlapping Zeckendorf source window, two one-step carry branches, one shared join, one terminal normal row, one Fibonacci value ledger, one non-hsame boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7073

A Concrete Naming Certificate for $\mathsf{DualityMapping}^{\uparrow}$

Following Browder and Cioranescu, the $\mathsf{DualityMapping}^{\uparrow}$ packet records the normalized duality mapping of a displayed normed or Hilbert-facing carrier as finite NameCert data. It sits over the Banach carrier of **??**, the dual-row interface of chapter 2265, the norm readback surface of chapter 89, the Hilbert specialization of chapter 91, and convex admissible domains named by chapter 208. The packet names the pairing and norm-equality obligations without importing a Hahn–Banach selector or a host dual-space theorem.

Definition 7073.1 (Duality mapping carrier)**.** *A* $\mathsf{DualityMapping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, X^*, N, J, E, Q, H, C, P, L)$$

with the following displayed rows:

1. *X is the* $\mathsf{BanachSpace}^{\uparrow}$ *source row, or the* $\mathsf{Hilbert}^{\uparrow}$ *source row when the normalized duality map is read through the Hilbert inner-product specialization.*
2. *X^* is the* $\mathsf{BanachDual}^{\uparrow}$ *row naming the admitted dual observations available to the packet.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row for source and dual norms, including the Real-valued norm endpoints used by the normalized condition.*
4. *J is the graph row of the normalized duality mapping, relating each displayed source point to the admitted dual row that witnesses the mapping read.*
5. *E is the norm-pairing exactness row, recording the displayed equalities between the pairing, the source norm square, and the dual norm product.*
6. *Q is the admissible-domain row, read through* $\mathsf{ConvexSet}^{\uparrow}$ *when the mapping is consumed by monotonicity, variational-inequality, or convex-analysis packets.*
7. *H records componentwise* hsame *transport for source, dual, norm, graph, exactness, admissible-domain, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source and dual rows through the normalized graph and exactness checks.*
9. *P is the* Pkg *provenance over* $\mathsf{DualityMapping}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. L is the local $\mathsf{NameCert}_{\mathsf{DualityMapping}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected norming functional, a global Hahn–Banach extension, quotient dual equality, weak-star compactness, or an ambient Banach-dual existence theorem.

Theorem 7073.2 (Duality mapping NameCert obligations)**.** *For every accepted* $\mathsf{DualityMapping}^{\uparrow}$ *carrier* $D = (X, X^*, N, J, E, Q, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{DualityMapping}^{\uparrow}}$ *surface consists exactly of carrier admission, dual-row admission, norm-pairing exactness, transport, replay, provenance, and non-escape of Hahn–Banach choice.*

Proof. Carrier admission projects the source row X and the norm row N. The packet may be consumed through $\mathsf{BanachSpace}^{\uparrow}$ or through the $\mathsf{Hilbert}^{\uparrow}$ specialization, but the source endpoint and norm readback remain displayed rows of the same finite carrier.

Dual-row admission projects X^* and the graph row J. A consumer reads only the dual observations already named in the $\mathsf{BanachDual}^{\uparrow}$ row, and the normalized mapping graph records which admitted dual row is paired with the displayed source point.

Norm-pairing exactness is the role of E. The pairing value, source norm, and dual norm are all read through N and sealed as displayed Real-valued data, so exactness is a packet row rather than a host dual-space equality.

Transport, replay, and provenance are carried by H, C, P, L. They preserve the same source, dual, norm, graph, exactness, and admissible-domain rows when a $\mathsf{ConvexSet}^{\uparrow}$ or Hilbert-facing consumer reuses the mapping.

Finally, Hahn–Banach choice is outside the carrier. Since no row of D stores a selected norming functional, an extension selector, quotient dual equality, weak-star compactness, or an ambient Banach-dual existence theorem, the local NameCert surface cannot export any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{DualityMapping}^{\uparrow}$ read must expose the $\mathsf{BanachSpace}^{\uparrow}$ or $\mathsf{Hilbert}^{\uparrow}$ source row, the $\mathsf{BanachDual}^{\uparrow}$ dual row, the $\mathsf{Norm}^{\uparrow}$ exactness row, the graph row, the $\mathsf{ConvexSet}^{\uparrow}$ admissible-domain row when used, BHist transport, hsame, Cont, Pkg, and local NameCert rows before any normalized duality-mapping conclusion is public.

Independence witness: banachspace, banachdual, norm, hilbert, convexset: $\mathsf{DualityMapping}^{\uparrow}$ is not bijective to any listed sibling because it jointly records source admission, dual-row admission, normalized norm-pairing exactness, graph replay, convex-domain handoff, transport, provenance, and local naming while refusing choice-selected norming functionals and Hahn–Banach extension data.

Closure status for $\mathsf{NameCert}_{\mathsf{DualityMapping}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DualityMapping}^{\uparrow}$ packet is generated from a displayed BanachSpaceUp or HilbertUp source row, a BanachDualUp row, NormUp source and dual norm readbacks, a normalized graph row, a norm-pairing exactness row, an optional ConvexSetUp admissible-domain row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DualityMapping}^{\uparrow}}$, seed)

Scope closed: Definition 7073.1 and Definition 7073.2 seed the finite normalized duality-mapping packet over displayed Banach, dual, norm, Hilbert, and convex-domain rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DualityMapping}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close maximal monotonicity, smoothness of Banach spaces, strict convexity, reflexivity, weak-star compactness, Hahn–Banach extension, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dual-row admission, norm-pairing exactness, convex-domain handoff, transport, replay, provenance, and Hahn–Banach non-escape for BEDC.Derived.DualityMappingUp.

Chapter 7074

A Concrete Naming Certificate for $\mathsf{HadamardMatrix}^{\uparrow}$

Following Hadamard and the finite orthogonal-design tradition, the $\mathsf{HadamardMatrix}^{\uparrow}$ packet records a finite square matrix whose entries are displayed signs and whose distinct rows have zero inner product. It consumes the matrix carrier of chapter 49, the finite endpoint discipline of $\mathsf{Bool}^{\uparrow}$ in chapter 32, the size and row-index ledger of $\mathsf{Nat}^{\uparrow}$ in chapter 28, the vector carrier of $\mathsf{VecSpace}^{\uparrow}$ in chapter 47, and the inner-product rows of $\mathsf{InnerProduct}^{\uparrow}$ in chapter 86. The packet names only this finite sign-matrix certificate; it is not an infinite design theorem, a classification of all Hadamard orders, a selected normalization, a quotient matrix equality, or an ambient orthogonal-array construction.

Definition 7074.1 (Hadamard matrix carrier)**.** *A* $\mathsf{HadamardMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (N, M, S, R, O, V, I, T, C, P, L)$$

with the following displayed rows:

1. N *is the* $\mathsf{Nat}^{\uparrow}$ *size row, naming the common row and column index boundary.*
2. M *is the* $\mathsf{Mat}^{\uparrow}$ *row, exposing the finite square table and its row-column readback.*
3. S *is the sign ledger, recording for each displayed entry a Boolean endpoint whose decoded value is* $+1$ *or* -1*.*
4. R *is the row-vector ledger, reading every matrix row as a finite* $\mathsf{VecSpace}^{\uparrow}$ *vector over the same index boundary.*
5. O *is the orthogonality ledger, recording that the inner product of any two distinct displayed rows is zero.*
6. V *is the* $\mathsf{VecSpace}^{\uparrow}$ *source row for the row vectors and finite sign combinations.*
7. I *is the* $\mathsf{InnerProduct}^{\uparrow}$ *source row used by the pairwise orthogonality checks.*
8. T *records componentwise* hsame *transport for* N, M, S, R, O, V, I *and the local naming row.*
9. C *records displayed* Cont *replay from the matrix and sign rows through row-vector readback, inner-product checks, provenance, and local naming.*
10. P *is the* Pkg *provenance over* $\mathsf{HadamardMatrix}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

11. L is the local $\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$ *row.*

The classifier compares accepted packets only through $N, M, S, R, O, V, I, T, C, P, L$. It has no coordinate for a host classification theorem, quotient matrix equality, selected equivalence normalization, infinite orthogonal design, or ambient completion.

Theorem 7074.2 (Hadamard matrix NameCert obligations)**.** *For every accepted* $\mathsf{HadamardMatrix}^\uparrow$ *carrier $H = (N, M, S, R, O, V, I, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$ *surface consists exactly of size exposure through N, finite square-table exposure through M, sign-entry exposure through S, row-vector readback through R, V, pairwise orthogonality exposure through O, I, componentwise transport through T, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 7074.1. The rows $N, M, S, R, O, V, I, T, C, P, L$ are the only displayed rows available to a local $\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$ read.

The size and table rows are read first. The size row N fixes the finite index boundary, while M exposes the square matrix entries and row-column readback.

The sign ledger S then restricts every entry to a displayed Boolean endpoint decoded as $+1$ or -1. The row-vector ledger R reads those entries through the $\mathsf{VecSpace}^\uparrow$ source row V before any orthogonality check is available.

Finally the orthogonality ledger O reads pairwise row products through the $\mathsf{InnerProduct}^\uparrow$ row I, and the structural rows T, C, P, L transport, replay, source, and name the same finite packet. Since no classification, normalization selector, quotient matrix equality, infinite design, or ambient completion coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7074.3 (Hadamard matrix orthogonality handoff)**.** *Every* $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, *or* $\mathsf{InnerProduct}^\uparrow$ *consumer that reads orthogonality from an accepted* $\mathsf{HadamardMatrix}^\uparrow$ *carrier must first pass through*

$$N, M, S \longmapsto R, V \longmapsto O, I \longmapsto T, C, P, L.$$

Thus the exported orthogonality datum is a finite sign-row handoff, not a host matrix classification theorem, selected equivalence normalization, quotient equality, or ambient orthogonal-design principle.

Proof. Start with a consumer request for row orthogonality. By Definition 7074.1, the request first sees the finite size row N, the square table row M, and the sign ledger S.

The row-vector readback is next. Each row is exposed through R and the vector source V, so a consumer cannot inspect an inner-product row before the finite sign rows have been translated into displayed vector rows.

Only then does the orthogonality ledger O read the $\mathsf{InnerProduct}^\uparrow$ source row I. The structural rows T, C, P, L keep the same matrix, vector, and inner-product data attached to the local name.

Therefore the handoff exports exactly finite sign-row orthogonality. It does not provide a classification theorem, a selected normalized representative, quotient matrix equality, an infinite design, or an ambient orthogonal-array construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$, seed)

Scope closed: Definition 7074.1, Definition 7074.2, and Definition 7074.3 seed the finite sign-matrix route over Matrix, Bool, Nat, VecSpace, InnerProduct, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HadamardMatrix}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Hadamard-order existence, classification, selected normalization, quotient matrix equality, orthogonal-array construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-entry exposure, row-vector readback, pairwise orthogonality exposure, transport, replay, provenance, and local naming for BEDC.Derived.HadamardMatrixUp.

Constructive story: A HadamardMatrix$^{\uparrow}$ packet is generated from a Nat size row, Matrix square-table row, Bool sign ledger, VecSpace row-vector ledger, InnerProduct orthogonality checks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7075

A Concrete Naming Certificate for $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ packet records total boundedness through displayed finite epsilon-net data rather than through open-cover compactness or a selected choice principle. It sits over the metric and total-bounded sources of chapter 127, the finite epsilon-net carrier of chapter 1767, the compact-metric consumer surface of chapter 129, and the terminal real-name seal of chapter 38.

Definition 7075.1 (Finite total bounded net carrier)**.** *A* $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, T, E, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier, distance classifier, and metric-distance ledger read by the net.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *source row, exposing the total-bounded request at the displayed tolerance.*
3. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row, naming the finite center bundle, center-admission ledger, and coverage-distance ledger.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used when compact-metric or uniform-continuity consumers inspect the net through finite source schedules.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the positive dyadic tolerance and comparison cells used by the coverage ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational coverage values and tolerance comparisons.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for downstream real-valued distance reads.*
8. *H records componentwise* hsame *transport for the metric source, total-bounded source, finite-net witness, stream windows, dyadic tolerance, regular-rational readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a total-bounded coverage request through $M, T, E, W, D, R, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTotalBoundedNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, E, W, D, R, S, H, C, P, N$. It has no coordinate for an ambient finite subset, an open cover, a selected compactness proof, countable choice, quotient metric equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7075.2 (Finite total bounded net NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ *carrier $F = (M, T, E, W, D, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteTotalBoundedNet}^{\uparrow}}$ *surface consists exactly of metric-source admission, total-bounded request exposure, finite epsilon-net witness exposure, finite stream-window inspection, dyadic tolerance comparison, regular-rational readback, Real seal handoff, componentwise transport, displayed replay, provenance, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 7075.1. The analytic source rows are the metric source M, the total-bounded request T, and the finite epsilon-net witness E.

The tolerance and readback route is finite. The stream windows W restrict the inspected source schedules, the dyadic row D supplies the positive tolerance comparisons, and the regular-rational row R records the coverage readback before the Real seal S is offered.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. Since no additional coordinate is present, the local NameCert obligations are exhausted by the listed rows and cannot import an open cover, a selected compactness proof, countable choice, quotient metric equality, or ambient Real completeness. □

Theorem 7075.3 (Finite total bounded net Real handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or real-valued uniform-continuity consumer of an accepted* $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through*

$$M, T, E, W, D, R \longmapsto S.$$

Thus each Real-facing distance or tolerance read factors through the displayed metric source, total-bounded source, finite epsilon-net witness, stream window, dyadic tolerance row, and regular-rational readback before any sealed Real output is public.

Proof. Start from the obligation surface of Definition 7075.2. It places M, T, E before any coverage consumer can read a total-bounded net.

Use the finite epsilon-net route of chapter 1767. The row E supplies center admission and coverage-distance data, while W restricts the inspection to displayed finite source schedules.

The tolerance rows are then forced by D and R. Dyadic comparisons are read before regular-rational coverage values, and the Real seal S receives only that already displayed readback.

Therefore a compact-metric or uniform-continuity consumer reaches $\mathsf{Real}^{\uparrow}$ through the finite route shown above. The carrier has no coordinate from which it could export open-cover compactness, countable choice, quotient equality, or ambient completeness. □

Theorem 7075.4 (Finite total bounded net sibling dependency)**.** *The* $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ *packet is consumed only as a sibling route tying* $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows together. Dropping any one of M, T, E, W, D, R, S leaves the read before the finite total-bounded net boundary and cannot be retyped as a* $\mathsf{CompactMetric}^{\uparrow}$ *coverage witness.*

Proof. Read the carrier rows in Definition 7075.1. The packet has separate coordinates for the metric source, total-bounded source, finite epsilon-net witness, stream windows, dyadic tolerance, regular-rational readback, and Real seal.

Apply Definition 7075.2. Each coordinate is a local NameCert obligation, so none can be inferred from a different sibling row after the packet has been accepted.

Use Definition 7075.3. Compact-metric and Real-facing consumers read the route through M, T, E, W, D, R, S in order. If any one of these rows is absent, the consumer has not reached the finite total-bounded net handoff and receives no compact-metric coverage witness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ consumer exposes metric, total-bounded, finite epsilon-net, stream-window, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows before reading a compact-metric or uniform-continuity coverage route.

Independence witness: totallybounded, compactmetric, finite-epsilon-net, real: $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ is not bijective to any listed sibling because it binds total-bounded requests to finite epsilon-net data, finite source windows, dyadic tolerances, regular rational readbacks, Real seals, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTotalBoundedNet}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTotalBoundedNet}^{\uparrow}$ packet is generated from Metric, TotallyBounded, FiniteEpsilonNet, StreamName, DyadicRatCore, RegSeqRat, and Real rows, plus componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTotalBoundedNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7075.1, Definition 7075.2, Definition 7075.3, and Definition 7075.4 seed the finite total-bounded net packet over displayed metric, total-bounded, finite epsilon-net, stream-window, dyadic tolerance, regular-rational readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTotalBoundedNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient metric equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite epsilon-net exposure, tolerance readback, Real seal nonescape, sibling dependency, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTotalBoundedNetUp.

Chapter 7076

A Concrete Naming Certificate for $\mathsf{RealInverseModulus}^{\uparrow}$

The $\mathsf{RealInverseModulus}^{\uparrow}$ packet realizes the guarded reciprocal-modulus route of the Real and Completion vision as finite BEDC data. It sits after the reciprocal-window budget of chapter 703, the regular Cauchy reciprocal surface of chapter 1471, the real reciprocal handoff of chapter 6251, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records a reciprocal modulus for guarded reads; it is not a total inverse on $\mathsf{Real}^{\uparrow}$ and not a completeness theorem.

Definition 7076.1 (Real inverse-modulus carrier)**.** *A* $\mathsf{RealInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, A, B, Q, R, G, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{Real}^{\uparrow}$ *source row whose guarded reciprocal modulus is being read.*
2. *A is the displayed apartness or lower-bound row that keeps the source away from the zero boundary.*
3. *B is the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row selecting the finite inverse-window budget before any reciprocal readback is public.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by that finite budget.*
5. *R is the* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *sibling route, retaining the regular-source, apartness, window, dyadic, budget, readback, and seal rows.*
6. *G is the* $\mathsf{RealReciprocal}^{\uparrow}$ *guarded reciprocal handoff row.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the reciprocal-modulus candidate.*
8. *H records componentwise* hsame *transport for the Real source, apartness row, inverse budget, regular readback, regular-Cauchy reciprocal route, Real reciprocal handoff, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a reciprocal-modulus request through $X, A, B, Q, R, G, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealInverseModulus}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}$ $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{RealInverseModulus}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, A, B, Q, R, G, E, H, C, P, N$. It has no coordinate for total inversion, trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 7076.2 (Real inverse-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RealInverseModulus}^\uparrow$ *carrier $I = (X, A, B, Q, R, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealInverseModulus}^\uparrow}$ *surface consists exactly of source Real admission, apartness exposure, finite Cauchy inverse budget selection,* $\mathsf{RegSeqRat}^\uparrow$ *reciprocal-modulus readback,* $\mathsf{RegularCauchyReciprocal}^\uparrow$ *sibling routing,* $\mathsf{RealReciprocal}^\uparrow$ *guarded handoff, terminal* $\mathsf{Real}^\uparrow$ *sealing, componentwise transport, displayed replay, provenance, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 7076.1. The source and domain rows are X and A, and the first reciprocal-facing row is the finite inverse budget B.

The readback route is ordered. The budget B produces the regular rational row Q, the sibling route R retains the regular-Cauchy reciprocal window, and the guarded Real reciprocal handoff G precedes the terminal Real seal E.

The rows H, C, P, N transport, replay, source, and name the same packet. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations and exclude total inversion, trichotomy, quotient equality, selected representatives, countable choice, and ambient Real completeness. □

Theorem 7076.3 (Real inverse-modulus Cauchy inverse budget handoff)**.** *Every accepted* $\mathsf{RealInverseModulus}^\uparrow$ *read factors through*

$$X, A \longmapsto B \longmapsto Q \longmapsto R, G \longmapsto E.$$

In particular, the $\mathsf{CauchyInverseBudget}^\uparrow$ *row is consumed before any* $\mathsf{RealReciprocal}^\uparrow$ *seal is public, and the reciprocal-modulus route remains guarded by the displayed apartness row.*

Proof. Apply Definition 7076.2. It exposes the source and apartness rows before any inverse budget or reciprocal readback can be consumed.

Use the budget route from chapter 703. The row B is the finite window selector for the guarded source, and Q is read only after that budget is present.

Next use the sibling reciprocal surfaces of chapter 1471 and chapter 6251. They keep the reciprocal route downstream of the budgeted regular readback and refuse a total inverse on arbitrary Real names.

The terminal seal E therefore receives only the guarded route displayed above. Because the carrier has no coordinate for a total inverse, trichotomy, quotient equality, selected representative, or completeness principle, none of those endpoints is exported. □

Falsifiable prediction: Every accepted $\mathsf{RealInverseModulus}^\uparrow$ read factors through $\mathsf{CauchyInverseBudget}^\uparrow$ before any $\mathsf{RealReciprocal}^\uparrow$ seal is public.

Independence witness: cauchyinversebudget, regular-cauchy-reciprocal, real-reciprocal, regseqrat, real: $\mathsf{RealInverseModulus}^\uparrow$ carries the guarded modulus route tying these siblings together, while none of them alone records the full inverse-modulus packet.

The Lean TasteGate file for this packet must supply $\mathsf{Nontrivial}$ and $\mathsf{FieldFaithful}$ instances by exhibiting accepted guarded reciprocal-modulus packets and refused boundary packets over $X, A, B, Q, R, G, E, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RealInverseModulus}^\uparrow}$.

Constructive story: A RealInverseModulus$^{\uparrow}$ packet is generated from a Real source, apartness row, CauchyInverse-Budget row, RegSeqRat readback, RegularCauchyReciprocal route, RealReciprocal handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7076.1, Definition 7076.2, and Definition 7076.3 seed the guarded reciprocal-modulus packet over finite inverse budgets, regular rational readbacks, reciprocal sibling routes, and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{RealInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a total inverse on Real$^{\uparrow}$, trichotomy, quotient real equality, selected representatives, countable choice, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget-first readback, guarded reciprocal handoff, Real seal nonescape, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.RealInverseModulusUp.

Chapter 7077

A Concrete Naming Certificate for $\mathsf{RegulatedSequence}^{\uparrow}$

Following Bishop and Bridges on regulated and Cauchy sequence moduli, the $\mathsf{RegulatedSequence}^{\uparrow}$ packet records the finite window discipline that turns a displayed stream name into regular rational readback data before the terminal real-name seal is read. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window route of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ admission surface of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite means object for the L10 corridor; it is not a quotient stream representative, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7077.1 (Regulated sequence carrier)**.** *A* $\mathsf{RegulatedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing the inspected sequence indices and the visible window schedule.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the dyadic precision demand attached to that window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readiness row, recording that the displayed window and tolerance data are acceptable for regular rational readback.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-refusal row, available only after S, D, Q have been displayed and refusing use as ambient completeness.*
5. *H records componentwise* hsame *transport for the stream, tolerance, readiness, and seal rows.*
6. *C records displayed* Cont *replay from stream windows through dyadic tolerance rows to regular readback and Real sealing.*
7. *P is the* Pkg *provenance over* $\mathsf{RegulatedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, D, Q, E, H, C, P, N. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegulatedSequenceUp`

Theorem 7077.2 (Regulated sequence window obligations)**.** *For every accepted* $\mathsf{RegulatedSequence}^{\uparrow}$ *carrier* $G = (S, D, Q, E, H, C, P, N)$*, any consumer that asks for regulated-sequence data must expose the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readiness row* Q *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal-refusal row* E *is available.*

Proof. Project the carrier of Definition 7077.1. The rows S, D, Q, E, H, C, P, N are the whole accepted packet, so no consumer has a route to E outside the displayed finite rows.

Read the dependency order in the carrier. The stream row supplies the inspected finite window, the dyadic row supplies the precision demand, and the readiness row records that the pair can be consumed by the regular rational readback surface.

The structural rows preserve the same route. Transport, replay, provenance, and local naming do not add a hidden limit, quotient representative, choice schedule, trichotomy proof, or ambient Real-completeness coordinate. □

Theorem 7077.3 (Regulated sequence shift stability)**.** *If two accepted* $\mathsf{RegulatedSequence}^{\uparrow}$ *packets have componentwise* hsame *stream-window, dyadic-tolerance, readiness, and Real-seal rows, then every shifted finite-window read admitted by the first packet is transported to the corresponding shifted finite-window read of the second packet through the displayed* H *and replayed through* C*. The transported read remains a* $\mathsf{RegulatedSequence}^{\uparrow}$ *read and does not become quotient stream equality or a selected limit.*

Proof. Start with the componentwise transport row H. It identifies the displayed stream-window, tolerance, readiness, and seal rows under hsame, so the shifted window read is transported only along rows already present in the two carriers.

Next use the replay row C. The same finite route from S through D and Q to E is replayed after the shift, preserving the order in which the tolerance and readiness data are exposed.

Since the carrier has no row for quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the shifted read remains a regulated finite-window certificate. □

Theorem 7077.4 (Regulated sequence Real source route)**.** *Every Real-facing source read of an accepted* $\mathsf{RegulatedSequence}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The $\mathsf{Real}^{\uparrow}$ *seal is downstream of finite stream windows, dyadic tolerance rows, and RegSeqRat readiness rows, and it exports no quotient real equality, selected limit, countable-choice schedule, trichotomy, or ambient completeness.*

Proof. Use Definition 7077.2 to expose the rows S, D, Q, E. This places the stream window and dyadic tolerance before the regular rational readiness row and before the Real seal.

The route is grounded in the existing L10 chapters: $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational admission surface, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal boundary.

Finally project H, C, P, N. These rows transport, replay, source, and name the same route, so a Real-facing read cannot bypass the finite sequence corridor or obtain any of the refused host principles. □

Falsifiable prediction: An accepted $\mathsf{RegulatedSequence}^{\uparrow}$ read is refuted if it reaches a Real-facing seal before displaying the StreamName window, DyadicRatCore tolerance, RegSeqRat readiness, Real seal-refusal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegulatedSequence}^{\uparrow}$ is not bijective to any listed sibling because it binds finite stream windows, dyadic tolerance demands, RegSeqRat-readiness rows, terminal Real seal refusal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one L10 intermediate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedSequence}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance rows, RegSeqRat-readiness rows, Real seal refusal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedSequence}^{\uparrow}}$, seed)

Scope closed: Definition 7077.1, Definition 7077.2, Definition 7077.3, and Definition 7077.4 seed the finite regulated-sequence route between stream windows and regular rational readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient Real completeness, checked obligation coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window obligations, shift stability, Real source routing, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedSequenceUp.

Chapter 7078

A Concrete Naming Certificate for $\mathsf{SupplyTaxonomy}^{\uparrow}$

The $\mathsf{SupplyTaxonomy}^{\uparrow}$ packet realizes the source, audit, and refusal surface of chapter 9394 as finite NameCert data. It sits beside the wider socket-kind classifier of chapter 523, the socket ledger of chapter 528, the audit route of chapter 1107, and the legitimate-claim ledger of chapter 5122. Those siblings classify socket kinds, record socket use, audit external routes, or police claim status; this packet binds the source taxonomy row that a classifier must inspect before it decides whether a supplied item is internal, external, auditable, or refused.

Definition 7078.1 (Supply taxonomy carrier)**.** *A* $\mathsf{SupplyTaxonomy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, E, A, R, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the internal-source row, naming the displayed constructor, definition, setup field, certificate row, or generated carrier that supplies the item.*
2. *E is the external-supply row, naming the socket, boundary, corpus, observer, or apophatic source position from which a non-internal item is requested.*
3. *A is the audit-gate row, recording the local gate that decides whether I is sufficient, whether E is only named, and which refusal is active.*
4. *R is the refusal row, recording the exact missing witness, unrestricted induction step, quotient equality, empirical persistence, inter-history recognition, inscription source, or self-reference closure that is not supplied.*
5. *K is the classifier-decision row, read only after I, E, A, R have been exposed.*
6. *H records componentwise* hsame *transport for source, external supply, audit gate, refusal, and classifier rows.*
7. *C records displayed* Cont *replay of the same finite source-audit-refusal route.*
8. *P is the* Pkg *provenance over* $\mathsf{SupplyTaxonomy}^{\uparrow}$, $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SupplyTaxonomy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, E, A, R, K, H, C, P, N$. It has no coordinate for hidden theorem supply, unledgered external import, global truth validation, quotient collapse, or a silent conversion from named supply to internal evidence.

Theorem 7078.2 (Supply taxonomy decision obligations)**.** *Every accepted* $\mathsf{SupplyTaxonomy}^{\uparrow}$ *classifier decision must expose the internal-source row I, external-supply row E, audit-gate row A, and refusal row R before the decision row K is read. A decision that sees only a socket kind, only a claim marker, or only a boundary name is not an accepted supply-taxonomy read.*

Proof. Project the carrier of Definition 7078.1. The rows $I, E, A, R, K, H, C, P, N$ are the entire accepted packet, so the classifier decision K has no route outside the displayed source, audit, and refusal rows.

The source split is visible before the decision. The internal row I records what is supplied by a displayed substrate route, while the external row E records the requested non-internal position without consuming it.

The audit and refusal rows then govern the decision. Row A names the local gate and row R names the missing or refused content; only after those rows are present may K classify the read.

Transport, replay, provenance, and local naming preserve the same finite route. They cannot turn a socket kind, marker, or boundary name into supplied content. □

Theorem 7078.3 (Supply taxonomy audit refusal nonescape)**.** *For an accepted* $\mathsf{SupplyTaxonomy}^{\uparrow}$ *packet $S = (I, E, A, R, K, H, C, P, N)$, any downstream consumer that reads K must retain the audit gate A and refusal row R in the same carrier. Thus an external supply request can be classified, transported, replayed, and named, but it cannot escape as internal evidence unless a separate displayed internal-source row is present.*

Proof. Start from the decision obligations of Definition 7078.2. They require I, E, A, R before K, so the decision is never detached from the audit gate or refusal row.

If the item is internal, the row I names the substrate route that supplies it. If the item is external, the row E names the supply position and R records the content that remains refused.

The structural rows do not change that split. Componentwise hsame transport and displayed Cont replay preserve the rows already present, while provenance and local naming record the same packet for later consumers. □

Theorem 7078.4 (Supply taxonomy sibling separation)**.** *The* $\mathsf{SupplyTaxonomy}^{\uparrow}$ *packet is not reducible to* $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, *or* $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$. *It jointly carries internal source, external supply, audit gate, refusal, classifier decision, transport, replay, provenance, and local naming rows, whereas each listed sibling omits at least one of the internal-source or classifier-decision coordinates required by Definition 7078.1.*

Proof. Compare the carrier rows. The wider transcendental taxonomy classifies socket kinds, and the socket ledger records socket use, but neither one by itself binds the internal-source row I to a classifier decision K.

The external-supply audit route records an audit path for a boundary request, while the legitimate-claim audit ledger records claim status. Those siblings supply important gates, but they do not jointly carry I, E, A, R, K as one source/audit/refusal classifier surface.

Therefore no listed sibling can be a bijective replacement for the present packet. A consumer that needs the supply decision must read the full $\mathsf{SupplyTaxonomy}^{\uparrow}$ carrier. □

The Lean follow-up for $\mathsf{SupplyTaxonomy}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ SupplyTaxonomy}^{\uparrow}$ and $\mathsf{FieldFaithful\ SupplyTaxonomy}^{\uparrow}$ instances. The field-faithfulness witness must keep $I, E, A, R, K, H, C, P, N$ as distinct rows so internal source, external supply, audit gate, refusal, and classifier decision cannot collapse.

Falsifiable prediction: Every accepted $\mathsf{SupplyTaxonomy}^{\uparrow}$ read exposes internal source, external supply, audit gate, and refusal rows before a classifier decision.

Independence witness: theorem-dna-complexity-scoring, trigger-blocker-duality, legitimate-claim-audit-ledger, external-supply-boundary: $\mathsf{SupplyTaxonomy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries internal source, external supply, audit gate, refusal, classifier decision, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SupplyTaxonomy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SupplyTaxonomy}^{\uparrow}$ packet is generated from internal-source rows, external-supply rows, audit gates, refusal rows, classifier decisions, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupplyTaxonomy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7078.1, Definition 7078.2, Definition 7078.3, and Definition 7078.4 seed the finite source/audit/refusal classifier surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupplyTaxonomy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, hidden theorem supply, unledgered external import, global truth validation, quotient collapse, or apophatic discharge.

Upgrade path: Obligation closure requires checked carrier admission, decision obligations, audit-refusal retention, sibling separation, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.SupplyTaxonomyUp.

Chapter 7079

A Concrete Naming Certificate for $\mathsf{HadamardDeterminantInequality}^{\uparrow}$

Following Hadamard and classical finite-dimensional matrix theory, the $\mathsf{HadamardDeterminantInequality}^{\uparrow}$ packet records the finite row-norm surface behind the determinant bound as BEDC NameCert data. It consumes the matrix carrier of chapter 49, the determinant boundary of chapter 84, the norm rows of chapter 89, the inner-product surface of chapter 86, the vector-space rows of chapter 47, the terminal scalar seal of chapter 38, and the Gram determinant packet of chapter 5127. The packet names a finite determinant-estimate route; it does not import host spectral theory, selected orthonormal bases, quotient matrix equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7079.1 (Hadamard determinant inequality carrier)**.** *A* $\mathsf{HadamardDeterminantInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (V, M, D, I, N, G, R, B, E, T, C, P, L)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row carrying the finite row-vector source.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *row carrying the displayed square matrix and its finite row ledger.*
3. *D is the* $\mathsf{Determinant}^{\uparrow}$ *row naming the determinant read for the same displayed matrix.*
4. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row used to read row inner products and orthogonality comparisons when they are displayed.*
5. *N is the* $\mathsf{Norm}^{\uparrow}$ *row listing the finite row-norm reads and their* $\mathsf{Real}^{\uparrow}$ *scalar seals.*
6. *G is the* $\mathsf{GramDeterminant}^{\uparrow}$ *handoff row, recording the Gram matrix and determinant surface obtained from the displayed row vectors.*
7. *R is the row-product budget, the finite product of the displayed row-norm bounds in N.*
8. *B is the determinant-bound ledger, relating the determinant row D to the Gram determinant handoff G and then to the product budget R.*
9. *E is the equality-case exposure row: it records only displayed orthogonality or zero-row data already present in I, N, G, not an ambient spectral argument.*
10. *T records componentwise* hsame *transport for the vector, matrix, determinant, inner-product, norm, Gram, budget, bound, and equality-case rows.*

11. *C records displayed* Cont *replay from matrix rows to determinant rows, row-norm rows, Gram determinant rows, and the final bound ledger.*
12. *P is the* Pkg *provenance over* $\mathsf{HadamardDeterminantInequality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{GramDeterminant}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *L is the local* $\mathsf{NameCert}_{\mathsf{HadamardDeterminantInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host spectral theory, selected orthonormal bases, quotient matrix equality, hidden row operations, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7079.2 (Hadamard determinant inequality NameCert obligations)**.** *For every accepted* $\mathsf{HadamardDeterminantInequality}^{\uparrow}$ *carrier*

$$H = (V, M, D, I, N, G, R, B, E, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HadamardDeterminantInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the vector-space, matrix, determinant, inner-product, norm, Gram determinant, row-product, bound, equality-case, transport, replay, provenance, and local naming rows; row-norm exposure through N and* $\mathsf{Real}^{\uparrow}$*; determinant exposure through M, D; Gram handoff through I, G; row-product budget exposure through R; determinant-bound exactness through B; equality-case exposure through E; and non-escape from host spectral theory, selected orthonormal bases, quotient matrix equality, hidden row operations, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7079.1. The displayed matrix row M is read together with the vector-space row V, and the determinant row D is attached to that same square matrix.

The row-norm data are then read through N and sealed through $\mathsf{Real}^{\uparrow}$. The inner-product row I supplies only displayed scalar products, while the Gram handoff G packages those reads as a finite Gram determinant surface.

The bound ledger B is downstream of D, G, R. It can compare the displayed determinant read with the Gram determinant route and the finite row-product budget, but it cannot add spectral data or a selected orthonormal basis.

Finally, T, C, P, L transport, replay, source, and name the same finite packet. Since no remaining coordinate stores quotient matrix equality, hidden row operations, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7079.3 (Hadamard row-norm determinant-bound route)**.** *Every determinant-estimate consumer of an accepted* $\mathsf{HadamardDeterminantInequality}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Mat}^{\uparrow} \longmapsto \mathsf{Determinant}^{\uparrow} \longmapsto (\mathsf{InnerProduct}^{\uparrow}, \mathsf{Norm}^{\uparrow}) \longmapsto \mathsf{GramDeterminant}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $V, M, D, I, N, G, R, B, E, T, C, P, L$. The consumer receives the row-norm determinant bound only after the matrix row, determinant row, row-norm ledger, Gram determinant handoff, row-product budget, bound ledger, transport, replay, provenance, and local naming row have all been displayed. No accepted route exports host spectral theory, selected orthonormal bases, quotient matrix equality, hidden row operations, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a determinant-estimate consumer. Carrier projection exposes the matrix row M and determinant row D before any bound can be read.

Next read the row geometry. The inner-product row I and norm row N expose only the finite scalar products, row norms, and real scalar seals attached to the displayed rows of M.

The Gram determinant row G is then the only bridge from row geometry to a determinant comparison. The product budget R and bound ledger B record the finite row-norm product and the displayed comparison with the determinant row.

The equality-case row E remains internal to displayed zero-row or orthogonality evidence. It does not select a basis or invoke a spectral decomposition. Transport, replay, provenance, and local naming through T, C, P, L preserve this route, so the consumer receives exactly the finite row-norm determinant-bound surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardDeterminantInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardDeterminantInequality}^{\uparrow}$ packet is generated from displayed VecSpaceUp, MatrixUp, DeterminantUp, InnerProductUp, NormUp, RealUp, GramDeterminantUp, row-product, determinant-bound, equality-case, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardDeterminantInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7079.1, Definition 7079.2, and Definition 7079.3 seed the finite row-norm determinant-bound surface over displayed matrix, determinant, norm, inner-product, vector-space, real, and Gram determinant rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardDeterminantInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host spectral theory, selected orthonormal bases, quotient matrix equality, hidden row operations, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, determinant exposure, row-norm exposure, Gram determinant handoff, row-product budget exactness, bound-ledger non-escape, transport, replay, provenance, and local naming for BEDC.Derived.HadamardDeterminantInequalityUp.

Chapter 7080

A Concrete Naming Certificate for $\mathsf{HalfOpenRealInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{HalfOpenRealInterval}^{\uparrow}$ packet records a half-open real interval $[a, b)$ as finite endpoint, containment, and boundary data rather than as a host subset of the classical real line. It sits between the dyadic endpoint ledger of chapter 291, the located endpoint surface of chapter 1709, the closed interval boundary of chapter 1591, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the traditional interval shape with an included left boundary and an excluded right boundary, while keeping topology, quotient real equality, selected endpoint representatives, trichotomy, and ambient real completeness outside the carrier.

Definition 7080.1 (HalfOpenRealInterval carrier)**.** *A* $\mathsf{HalfOpenRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, O, D, W, R, E, A, B, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint* $\mathsf{Real}^{\uparrow}$ *seal rows.*
2. *O is the endpoint-order row, recording the displayed route from L to U through finite comparison data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *boundary ledger, carrying dyadic lower cells, upper-cut cells, positive-radius rows, and one-sided comparison budgets.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect endpoint and candidate observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for candidate membership reads.*
7. *A is the left-admission row, saying that the lower endpoint is included by the displayed boundary route.*
8. *B is the right-refusal row, saying that the upper endpoint is excluded unless a stricter displayed left-of-upper witness is present.*
9. *H records componentwise* hsame *transports for endpoint seals, order, dyadic boundary data, finite windows, readback, seal handoff, one-sided boundary rows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a membership request through $L, U, O, D, W, R, E, A, B, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{HalfOpenRealInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It has no coordinate for a host topology, quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7080.2 (HalfOpenRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{HalfOpenRealInterval}^{\uparrow}$ *carrier*

$$I = (L, U, O, D, W, R, E, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two endpoint* $\mathsf{Real}^{\uparrow}$ *seals, endpoint-order row, dyadic boundary ledger, finite windows, regular rational readback, terminal seal handoff, one-sided boundary rows, transport, replay, provenance, and local naming;*
2. *located endpoint exactness, requiring endpoint and containment reads to pass through the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *and* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *sibling surfaces before the half-open boundary is consumed;*
3. *dyadic boundary exactness, requiring the left-inclusion and right-exclusion reads to use the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D;*
4. *regular readback exactness, requiring every membership observation to pass through the finite window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before reaching the* $\mathsf{Real}^{\uparrow}$ *seal handoff E;*
5. *left-boundary admission, requiring A to display that L belongs to the interval boundary;*
6. *right-boundary refusal, requiring B to display that U is not admitted as a member unless a stricter below-U witness is supplied;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rows above.*

No obligation is stored in a host subset predicate, quotient real equality proof, selected endpoint representative, trichotomy branch, choice functional, or ambient completeness theorem.

Proof. Project the accepted carrier. The projection exposes exactly the endpoint seals, endpoint-order row, dyadic boundary ledger, finite windows, regular readback, terminal seal handoff, one-sided boundary rows, and structural rows.

The located and closed-bounded sibling surfaces supply the endpoint discipline. They ensure that endpoint, order, and containment reads begin from displayed interval rows rather than from a host real-line subset.

The dyadic and regular-rational rows then determine the one-sided boundary. The ledger D supplies the lower cells, upper-cut cells, and positive comparison budgets, while W and R expose the finite observations used by the candidate and endpoint seals.

Finally A admits the left endpoint and B refuses the right endpoint as a member of the half-open boundary. The rows H, C, P, N preserve and name the same packet, so no host topology, quotient equality, selected representative, trichotomy branch, choice functional, or ambient completeness theorem can be read from the certificate. □

Lean4 ✓ `BEDC.Derived.HalfOpenRealIntervalUp.TasteGate.HalfOpenRealIntervalTasteGate single carrier alignment`

Theorem 7080.3 (HalfOpenRealInterval dyadic boundary induction)**.** *Let*

$$I = (L, U, O, D, W, R, E, A, B, H, C, P, N)$$

be an accepted $\mathsf{HalfOpenRealInterval}^{\uparrow}$ *carrier. Any finite extension of the dyadic boundary ledger* D *by one displayed comparison row preserves the half-open membership route: left-boundary reads still factor through* A*, right-boundary reads still factor through* B*, and every* $\mathsf{Real}^{\uparrow}$*-facing membership consumer still reads through* W, R, E, H, C, P, N *after the extended dyadic ledger is displayed.*

Proof. Proceed by finite induction on the displayed dyadic comparison extension. The base route is the carrier route of Definition 7080.1: membership reads pass from endpoint seals and endpoint order through $D, W, R, E, A, B, H, C, P, N$.

For the successor step, add one comparison row to the same dyadic ledger. Since the row is displayed inside D, it can refine the finite boundary budget but cannot introduce a second endpoint, a second membership classifier, or a host order predicate.

The left and right boundary rows are preserved separately. The left-admission row A continues to identify the included lower endpoint, while the right-refusal row B continues to require a stricter below-U witness before a point is admitted on the upper side.

The regular readback and Real seal rows are unchanged by the extension. Thus every Real-facing consumer still reaches the membership route through W, R, E, H, C, P, N, and no quotient real equality, selected endpoint representative, trichotomy branch, or ambient completeness coordinate is exported. □

Theorem 7080.4 (HalfOpenRealInterval sibling route)**.** *Every accepted* $\mathsf{HalfOpenRealInterval}^{\uparrow}$ *membership read factors through the finite sibling route*

$$\mathsf{LocatedRealInterval}^{\uparrow},\ \mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route first reads located and closed-bounded endpoint data, then the dyadic one-sided boundary ledger, then finite stream windows and regular rational readback, and only then the terminal real-name seal and local $\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}$ *rows.*

Proof. Start with the endpoint rows L, U, O. Their interval meaning is inherited from the located and closed-bounded sibling surfaces cited above, so endpoint order and containment are visible before the half-open boundary rows are consumed.

Next read the dyadic ledger D. It records the one-sided lower inclusion and upper exclusion budget that distinguishes $[a, b)$ from an open or closed interval over the same endpoints.

Finally pass through W, R, E, H, C, P, N. The finite windows and regular rational readback precede the terminal $\mathsf{Real}^{\uparrow}$ seal, and the structural rows transport, replay, source, and name only that displayed route. Hence the sibling route supplies the whole membership surface without importing host topology, quotient real equality, selected representatives, trichotomy, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted HalfOpenRealIntervalUp membership read passes through located endpoint rows, a closed-bounded endpoint boundary, a DyadicRatCoreUp one-sided comparison ledger, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any half-open membership result is visible.

Independence witness: dyadicratcore, locatedrealinterval, closedboundedinterval, regseqrat, real: $\mathsf{HalfOpenRealInterval}^{\uparrow}$ is not bijective to any listed sibling because it binds left endpoint admission, right endpoint refusal, dyadic one-sided comparison rows, finite readback, terminal seal handoff, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{HalfOpenRealInterval}^{\uparrow}$ packet is generated from lower and upper RealUp endpoint seals, an endpoint-order row, a DyadicRatCore one-sided boundary ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal handoff, left-admission and right-refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7080.1, Definition 7080.2, Definition 7080.3, and Definition 7080.4 seed the half-open interval boundary over located endpoint, closed-bounded interval, dyadic, regular-rational, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HalfOpenRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topology, quotient real equality, selected endpoint representatives, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.HalfOpenRealIntervalUp covering carrier admission, dyadic boundary exactness, one-sided boundary stability, regular readback, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 7081

A Concrete Naming Certificate for $\mathsf{StreamReindexing}^{\uparrow}$

Following Bishop and Bridges on constructive sequence names, the $\mathsf{StreamReindexing}^{\uparrow}$ packet records the finite schedule operation that reads one displayed $\mathsf{StreamName}^{\uparrow}$ observation plan through another. It sits before the regular-rational and real-facing reindexing routes of chapter 301, chapter 38, chapter 1568, and chapter 6960, and it uses the dyadic tolerance core of chapter 291 only after the reindexed stream windows have been displayed. The packet names a finite schedule certificate; it does not quotient streams, choose a tail representative, import countable choice, identify host stream equality, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7081.1 (Stream reindexing carrier)**.** *A* $\mathsf{StreamReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, I, J, W, D, R, E, H, C, P, N)$$

with the following rows:

1. *X is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite observations are being reread;*
2. *I is the displayed input schedule window listing the source indices available to the request;*
3. *J is the displayed reindexing schedule, assigning each requested output window to a finite input-window read;*
4. *W is the synchronized finite window ledger certifying that every output request has a displayed source read;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached only to the observed window requests;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after X, I, J, W, D have been displayed;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available to real-facing consumers of the reindexed name;*
8. *H records componentwise* hsame *transport for the stream row, schedules, windows, dyadic tolerance row, readback, and seal;*
9. *C records displayed* Cont *replay of the same finite reindexing route;*
10. *P is the* Pkg *provenance over* $\mathsf{StreamReindexing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{RegularCauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamReindexing}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected tails, unbounded schedule choice, host stream equality, countable choice, or ambient Real completeness.

Theorem 7081.2 (Stream reindexing NameCert obligations)**.** *For an accepted* $\mathsf{StreamReindexing}^{\uparrow}$ *carrier* $S = (X, I, J, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StreamReindexing}^{\uparrow}}$ *surface consists exactly of source-stream admission through* X*, input-window admission through* I*, reindexing-schedule admission through* J*, synchronized finite-window coverage through* W*, dyadic tolerance compatibility through* D*, regular-rational readback through* R*, terminal real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7081.1. The source stream, input schedule, reindexing schedule, synchronized windows, dyadic tolerance ledger, readback row, and terminal seal are all displayed coordinates.

Carrier admission reads only those coordinates. The synchronized window row W accounts for every output request by a finite source-window read, so the local certificate has no detached schedule object to compare.

Transport and replay are then the structural rows H and C, with provenance and local naming supplied by P and N. Since no excluded object is a carrier coordinate, the obligation surface cannot export quotient stream equality, selected tails, countable choice, host stream equality, or ambient completeness. □

Theorem 7081.3 (Stream reindexing window stability)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*, or* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *consumer of an accepted* $\mathsf{StreamReindexing}^{\uparrow}$ *packet must first read the displayed rows*

$$X, I, J, W, D, R, E, H, C, P, N.$$

In particular, the consumer receives the reindexed stream only through finite $\mathsf{StreamName}^{\uparrow}$ *windows, the synchronized schedule ledger, the dyadic tolerance row, the regular-rational readback, and the terminal real seal. It receives no quotient stream, selected tail, countable-choice schedule, host stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 7081.2. They place the source stream row X, schedule rows I, J, synchronized window row W, and dyadic tolerance row D before any readback row can be consumed.

Next read the regular-rational and real-facing rows. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is downstream of the finite schedule window, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of that handoff.

Finally apply componentwise transport and displayed replay through H and C. The provenance and local naming rows keep the same finite packet attached to the downstream $\mathsf{CauchyReindexing}^{\uparrow}$ route and its $\mathsf{RegularCauchyReindexing}^{\uparrow}$ specialization, so no hidden quotient, selected tail, countable-choice schedule, host equality, or ambient completeness coordinate is available. □

Falsifiable prediction: Every consumer of a $\mathsf{StreamReindexing}^{\uparrow}$ packet must expose the source $\mathsf{StreamName}^{\uparrow}$ row, finite input schedule, reindexing schedule, synchronized window ledger, dyadic tolerance row, regular-rational readback, and Real seal before it can use the reindexed stream.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-reindexing: $\mathsf{StreamReindexing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source stream row, input schedule, reindexing schedule, synchronized finite windows, dyadic tolerance ledger, regular-rational handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamReindexing}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamReindexing}^{\uparrow}$ packet is generated from a source StreamName row, finite input schedule, reindexing schedule, synchronized window ledger, dyadic tolerance row, RegSeqRat handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamReindexing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7081.1, Definition 7081.2, and Definition 7081.3 seed the finite schedule-reindexing surface used by regular-rational, real, and Cauchy reindexing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamReindexing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected tails, countable-choice schedules, host stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window coverage, dyadic tolerance compatibility, regular-rational handoff, Real seal nonescape, and local NameCert registration for BEDC.Derived.StreamReindexingUp.

Chapter 7082

A Concrete Naming Certificate for $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$

Following Bourbaki and Kelley-style ordered-field completeness, and Bishop–Bridges constructive located-supremum discipline, the $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$ packet records Dedekind completeness only as a finite NameCert boundary for displayed located, bounded field-family windows. It sits after the complete ordered-field interface of chapter 6012, the located upper-and-lower witness surface of chapter 1427, the terminal real-name seal of chapter 38, the algebraic order carrier of chapter 766, and the rowwise comparison boundary of chapter 3754. The packet is an L10 completion-boundary object; it does not construct $\mathsf{Real}^{\uparrow}$ by Dedekind cuts, quotient real names, choose suprema, import classical trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7082.1 (Dedekind-complete ordered-field carrier)**.** *A* $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, A, L, R, U, G, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CompleteOrderedField}^{\uparrow}$ *source row naming the ordered-field operations and completion-facing service only through finite BEDC data;*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row exposing addition, multiplication, positivity, and order comparison as displayed algebraic-order evidence;*
3. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row for located inhabited bounded family windows over the displayed field carrier;*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the ordered-field, algebraic-order, and located-supremum rows have been exposed;*
5. *U is the* $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ *comparison row, used when a consumer compares this boundary with a Cauchy-complete ordered-field presentation;*
6. *G is the Dedekind-boundary ledger listing the inspected lower-family rows, upper-bound rows, locatedness requests, and lower-approximation witnesses currently in scope;*
7. *W is the finite precision window controlling which field-family observations may be read through L and G;*
8. *H records componentwise* hsame *transport for $F, A, L, R, U, G, W, C, P, N$;*

9. *C records displayed* Cont *replay for the ordered-field, algebraic-order, located-supremum, Real-seal, comparison, boundary-ledger, and precision-window rows;*
10. *P is the* Pkg *provenance over* $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$, $\mathsf{CompleteOrderedField}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCompleteOrderedField}^{\uparrow}}$ *row.*

The accepted packet compares only $F, A, L, R, U, G, W, H, C, P, N$. It has no coordinate for a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient equality of real names, classical trichotomy, selected suprema, countable choice, host field equality, or ambient Real completeness.*

Theorem 7082.2 (Dedekind-complete ordered-field NameCert obligations)**.** *For an accepted packet $D = (F, A, L, R, U, G, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindCompleteOrderedField}^{\uparrow}}$ *obligations consist exactly of carrier admission for those rows, algebraic-order compatibility through A, ordered-field source exposure through F, located-supremum witnessing through L, terminal Real sealing through R, comparison routing through U, boundary-ledger exactness through G, finite precision control through W, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows F, A, L, R, U, G, W are the only mathematical coordinates, and H, C, P, N are the structural certificate rows.

Ordered-field and algebraic-order reads are forced through F and A. Located upper-bound and lower-approximation reads are forced through L and the finite window W, with the inspected lower-family and upper-bound rows recorded in G before the terminal Real seal R is available.

When a consumer compares the packet with a Cauchy-complete ordered-field presentation, the only available comparison surface is U. Transport and replay preserve the same rows componentwise.

Since the carrier contains no Dedekind-cut constructor for $\mathsf{Real}^{\uparrow}$, no quotient-real equality row, no trichotomy oracle, no selected-supremum coordinate, no countable-choice schedule, no host field equality, and no ambient Real-completeness coordinate, none of those objects can be a local NameCert obligation. □

Theorem 7082.3 (Dedekind-complete ordered-field L10 boundary)**.** *Every L10 consumer of an accepted* $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$ *packet must pass through the sibling route*

$$\mathsf{CompleteOrderedField}^{\uparrow} \longmapsto \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{LocatedSupremum}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and, for comparison with Cauchy-complete presentations, through $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ *before reading Dedekind-complete ordered-field service. The exported service is only the finite boundary ledger G over the precision window W together with structural transport, replay, provenance, and local naming; it exports no Dedekind-cut construction of* $\mathsf{Real}^{\uparrow}$*, quotient equality, classical trichotomy, selected supremum, countable choice, host field equality, or ambient Real completeness.*

Proof. Start with the source row F. It exposes complete ordered-field service through chapter 6012, not through a host complete ordered field.

Next read the algebraic order and located-supremum rows. The row A supplies only displayed $\mathsf{RealAlgOrder}^{\uparrow}$ operations and comparisons, while L supplies the upper-bound and lower-approximation faces already licensed by chapter 1427.

The terminal real-facing observation is then R, which is downstream of F, A, L, G, W. If the consumer compares this boundary with a Cauchy-complete presentation, it must additionally read U through chapter 3754.

The structural rows H, C, P, N preserve and name exactly that route. Because the excluded objects are not carrier coordinates, the L10 boundary cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCompleteOrderedField}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCompleteOrderedField}^{\uparrow}$ packet is generated from a CompleteOrderedField source, a RealAlgOrder row, a LocatedSupremum handoff, a terminal Real seal, a CauchyCompleteOrderedFieldUniqueness comparison row, a Dedekind-boundary ledger, a finite precision window, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCompleteOrderedField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7082.1, Definition 7082.2, and Definition 7082.3 seed the L10 Dedekind-complete ordered-field boundary over displayed ordered-field, algebraic-order, located-supremum, Real-seal, and comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCompleteOrderedField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a Dedekind-cut construction of RealUp, quotient equality, classical trichotomy, selected suprema, countable choice, host field equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal work can encode carrier admission, NameCert obligations, and the L10 boundary route for BEDC.Derived.DedekindCompleteOrderedFieldUp.

Chapter 7083

A Concrete Naming Certificate for $\mathsf{RegularRealLineAtlas}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{RegularRealLineAtlas}^{\uparrow}$ packet is a finite atlas of dyadic chart rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{Real}^{\uparrow}$ seals. It sits after the dyadic tolerance carrier of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the modulus schedule of chapter 292. The packet names finite real-line chart reads without importing a host real line, quotient stream equality, selected limits, countable choice, or open-cover compactness.

Definition 7083.1 (Regular real-line atlas carrier)**.** *A* $\mathsf{RegularRealLineAtlas}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *chart ledger, carrying the dyadic center, radius, overlap tolerance, and one-step refinement budget for each displayed chart row.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite stream entries inspected by the chart read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window and dyadic tolerance data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic chart row, finite window, and regular rational readback are visible.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *scheduling row that bounds the chart-window read by a finite precision request.*
6. *H records componentwise* hsame *transport for dyadic chart rows, stream windows, regular readback, Real seals, modulus schedules, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the finite atlas route*

$$D, S, M \longmapsto R \longmapsto E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularRealLineAtlas}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularRealLineAtlas}^{\uparrow}}$ *row.*
The classifier compares accepted atlas packets only through $D, S, R, E, M, H, C, P, N$. It has no coordinate for a host real line, quotient stream equality, selected limit, countable-choice selector, global open cover, compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7083.2 (Regular real-line atlas NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealLineAtlas}^{\uparrow}$ *carrier $A = (D, S, R, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRealLineAtlas}^{\uparrow}}$ *surface consists exactly of dyadic chart admission through D, finite stream-window exposure through S, regular rational readback through R, Real seal handoff through E, finite modulus scheduling through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7083.1. The displayed rows are exactly $D, S, R, E, M, H, C, P, N$, so every chart read begins from finite dyadic and stream data.

The scheduling and readback rows determine the chart obligation surface. Row M bounds the requested precision, row S displays the inspected window, and row R supplies the regular rational readback attached to that same window.

The Real seal row E is downstream of those finite rows. Transport, replay, provenance, and local naming preserve the same route, so no host real line, quotient stream equality, selected limit, countable-choice selector, open-cover compactness, or ambient completeness theorem can be exported. □

Theorem 7083.3 (Regular real-line atlas window handoff)**.** *Every window read from an accepted* $\mathsf{RegularRealLineAtlas}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The atlas can therefore hand a chart window to a downstream real-facing consumer only after the dyadic chart ledger, finite stream window, modulus schedule, regular rational readback, and terminal seal are all displayed.

Proof. Start with the dyadic chart row D and the schedule row M. Together they name the finite tolerance and precision bound for the chart window.

Read the window row S next. Since S exposes only finitely many stream observations, the regular rational readback R is a finite handoff rather than a quotient stream.

Finally attach the Real seal row E and the structural rows H, C, P, N. The resulting route is exactly the displayed atlas route, and it contains no selected limit, global cover, countable-choice enumeration, host equality, or ambient completeness coordinate. □

Theorem 7083.4 (Regular real-line atlas Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{RegularRealLineAtlas}^{\uparrow}$ *packet must pass through the rows*

$$D, S, R, E, M, H, C, P, N.$$

The terminal seal certifies only the finite atlas route already displayed by $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{CauchyModulus}^{\uparrow}$ *rows; it cannot be read as Real completeness, quotient stream equality, host real equality, a selected limit, countable choice, or open-cover compactness.*

Proof. Use Definition 7083.3. It places every atlas window read downstream of dyadic chart data, finite stream observations, modulus scheduling, and regular rational readback.

The Real seal row E occurs only after that route in the carrier. A Real-facing consumer therefore receives a sealed finite chart read, not an external real-line object.

Rows H, C, P, N preserve and name the same packet. Since completeness, quotient equality, host equality, selected limits, countable choice, and open-cover compactness are absent from the carrier, the Real seal has no escape route beyond the displayed finite atlas. □

Falsifiable prediction: Every accepted $\mathsf{RegularRealLineAtlas}^{\uparrow}$ chart read displays a DyadicRatCoreUp chart ledger, StreamNameUp finite window, RegSeqRatUp readback, RealUp seal, CauchyModulusUp schedule, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before any Real-facing consumer can use the atlas.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchymodulus: $\mathsf{RegularRealLineAtlas}^{\uparrow}$ is not bijective to any listed sibling because it binds finite dyadic chart rows, stream windows, regular rational readbacks, Real seals, modulus schedules, transport, replay, provenance, and local naming into one atlas packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealLineAtlas}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealLineAtlas}^{\uparrow}$ packet is generated from DyadicRatCore chart rows, StreamName finite windows, RegSeqRat readback, RealUp seal rows, a CauchyModulus schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealLineAtlas}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7083.1, Definition 7083.2, Definition 7083.3, and Definition 7083.4 seed the finite dyadic chart-window atlas route from displayed tolerances and windows to regular readback and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealLineAtlas}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, host real equality, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.RegularRealLineAtlasUp covering dyadic chart admission, finite window exposure, regular readback, Real seal nonescape, modulus scheduling, transport, replay, provenance, and local naming.

Chapter 7084

A Concrete Naming Certificate for $\mathsf{RealCompletionUnit}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the packet records the unit map from dyadic and rational data into the $\mathsf{Real}^{\uparrow}$ seal as finite BEDC data. It sits over the dyadic tolerance core of chapter 291, the finite stream-window row of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 95. The packet names the unit-map surface available to Real and Completion consumers without quotienting Cauchy names, selecting representatives, importing countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7084.1 (Real completion unit carrier)**.** *A* $\mathsf{RealCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, W, R, E, H, C, P, N)$$

with the following rows:

1. *D is the dyadic constant row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, that supplies the rational or dyadic source data for the unit map.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite observations by which the source data are presented to a regular-name consumer.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback has been displayed.*
5. *H records componentwise* hsame *transport for the dyadic row, stream-window row, regular readback, and Real seal.*
6. *C records displayed* Cont *replay of the same finite unit-map route.*
7. *P is the* Pkg *provenance over* $\mathsf{RealCompletionUnit}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected Cauchy representatives, host real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7084.2 (Real completion unit NameCert obligations)**.** *For an accepted* $\mathsf{RealCompletionUnit}^{\uparrow}$ *carrier $U = (D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}$ *surface consists of five*

obligation rows: dyadic source admission through D, finite stream-window admission through W, regular rational readback through R, terminal Real sealing through E, and structural certification through H, C, P, N. Every Real or Completion consumer of the unit map must read those rows on the same finite BEDC surface.

Proof. Project the carrier of Definition 7084.1. The dyadic source row D, stream-window row W, regular readback row R, and terminal seal row E are displayed coordinates, and the structural rows H, C, P, N transport, replay, source, and name exactly that route.

The first four obligations are therefore the analytic face of the unit map: source admission, window admission, readback, and sealing. The fifth obligation is the structural face binding those reads into one local NameCert surface.

No excluded shortcut is a coordinate of the carrier. Hence a consumer receives the unit map only through the displayed rows and cannot replace the route by quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient completeness. □

Theorem 7084.3 (Real completion unit nonescape)**.** *Any* $\mathsf{Real}^{\uparrow}$ *endpoint consumer of an accepted* $\mathsf{RealCompletionUnit}^{\uparrow}$ *carrier* $U = (D, W, R, E, H, C, P, N)$ *that drops one of* D*,* W*,* R*,* E*,* H*,* C*,* P*, or* N *leaves the accepted unit-map surface. In particular, after dropping any one of those rows, the consumer cannot read the* $\mathsf{Real}^{\uparrow}$ *endpoint, cannot replace the finite route with quotient equality of completed names, cannot choose a representative Cauchy name, cannot import external real completeness, and cannot appeal to countable choice.*

Proof. Start with the carrier projection of Definition 7084.1. The endpoint row E is downstream of the dyadic source row D, the stream-window row W, and the regular readback row R.

Apply the obligation surface of Definition 7084.2. It binds the analytic rows D, W, R, E and the structural rows H, C, P, N into one local $\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}$ surface, so a consumer that drops any row no longer has the displayed certificate route.

The excluded replacements are not rows of U. Thus dropping a coordinate cannot be repaired by quotient equality, a selected representative, external completeness, or countable choice; it simply removes the consumer from the accepted finite unit-map packet. □

Falsifiable prediction: Every consumer of a $\mathsf{RealCompletionUnit}^{\uparrow}$ packet must expose the dyadic source row, stream-window row, regular rational readback, Real seal, componentwise transport, replay, provenance, and local NameCert row before it can use the unit map into $\mathsf{Real}^{\uparrow}$.

Independence witness: dyadicratcore, streamname, regseqrat, real, completion: $\mathsf{RealCompletionUnit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the dyadic source row, finite stream-window row, regular readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCompletionUnit}^{\uparrow}$ packet is generated from a dyadic constant row, StreamName window row, RegSeqRat readback row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7084.1, Definition 7084.2, and Definition 7084.3 seed the finite unit-map surface from dyadic and rational data into the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected Cauchy representatives, host real equality, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic source admission, stream-window admission, regular readback, Real seal nonescape, and local NameCert registration for BEDC.Derived.RealCompletionUnitUp.

Chapter 7085

A Concrete Naming Certificate for $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$

Following Bishop–Bridges on Archimedean bounds and regular Cauchy real names, the RegularCauchyArchimedeanB packet records the finite threshold handoff from Archimedean evidence to regular rational and real-facing reads. It sits between chapter 430, chapter 1672, chapter 126, chapter 301, and chapter 38. The packet is a finite L10 bridge, not a supremum principle, trichotomy row, quotient stream, countable-choice schedule, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness object.

Definition 7085.1 (Regular Cauchy Archimedean bridge carrier)**.** *A* $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row, including the rational bound witness and public threshold surface.*
2. *D is the* $\mathsf{DyadicArchimedean}^{\uparrow}$ *scale row that turns the Archimedean bound into finite dyadic threshold data.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family used by threshold and regular-Cauchy consumers.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed Archimedean, dyadic, and stream-window rows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the regular rational readback row R has been displayed.*
6. *H records componentwise* hsame *transport for the Archimedean bound, dyadic scale, stream windows, regular readback, Real seal, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the finite route*

$$A, D, S \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{DyadicArchimedean}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, D, S, R, E, H, C, P, N$. It has no coordinate for a supremum, trichotomy, quotient stream equality, countable-choice schedule, selected completed limit, or ambient Real completeness.

Theorem 7085.2 (Regular Cauchy Archimedean bridge NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ *carrier*

$$B = (A, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}}$ *surface consists exactly of Archimedean bound admission, dyadic scale admission, finite StreamName window admission, RegSeqRat readback, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read of the packet factors through these rows and receives no supremum, trichotomy, quotient stream equality, countable-choice schedule, selected limit, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 7085.1. The rows A, D, S, R, E expose the Archimedean bound, dyadic scale, finite stream windows, regular rational readback, and terminal Real seal.

The first three rows are source rows. The Archimedean part is read through chapter 430, the dyadic scale through chapter 1672, and the finite windows through chapter 126.

The readback rows close the bridge. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of A, D, S, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of R.

Finally the structural rows H, C, P, N transport, replay, source, and name only the displayed route. Since no excluded object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7085.3 (Regular Cauchy Archimedean threshold handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{ArchimedeanReal}^{\uparrow},\ \mathsf{DyadicArchimedean}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the rows $A, D, S, R, E, H, C, P, N$. The handoff exports Archimedean threshold evidence to regular rational and real-facing consumers without importing a supremum, trichotomy, quotient stream equality, countable-choice schedule, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7085.2. They place the Archimedean bound row A, dyadic scale row D, and finite stream-window row S before any readback endpoint is available.

Read the regular-Cauchy endpoint next. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is produced only from the displayed A, D, S threshold route, so no quotient stream or countable schedule can be substituted for the finite window family.

Only after that row is visible may the terminal $\mathsf{Real}^{\uparrow}$ seal E be inspected. Transport and replay through H, C preserve the same finite route, while P, N source and name it.

Thus every downstream regular-rational or Real-facing consumer receives the stated threshold handoff and no excluded host or completeness object. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ RegSeqRat or Real-facing read factors through an ArchimedeanRealUp bound row, a DyadicArchimedeanUp scale row, finite StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: archimedeanreal, dyadic-archimedean, streamname, regseqrat, real: $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Archimedean bound evidence, dyadic threshold scale data, finite stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming while refusing supremum, trichotomy, quotient streams, selected limits, countable choice, and Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}$ packet is generated from ArchimedeanReal bound evidence, a DyadicArchimedean scale row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7085.1, Definition 7085.2, and Definition 7085.3 seed the finite Archimedean-threshold bridge from bound rows to regular-Cauchy and Real-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyArchimedeanBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close supremum existence, trichotomy, quotient stream equality, countable-choice schedules, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier agreement, dyadic threshold exactness, StreamName window coverage, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyArchimedeanBridgeUp.

Chapter 7086

A Concrete Naming Certificate for $\mathsf{AbelUniformConvergence}^{\uparrow}$

Following Abel, Hardy, and Bishop–Bridges constructive analysis, the $\mathsf{AbelUniformConvergence}^{\uparrow}$ packet records the finite interface by which Abel radial convergence is read uniformly over displayed radius windows. It sits after the Abel theorem route of chapter 1922, the uniform limit packet of chapter 463, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter packages traditional Abel uniform convergence as finite BEDC rows; it does not assert ambient real completeness, countable choice, quotient stream equality, or selected infinite sums.

Definition 7086.1 (AbelUniformConvergence carrier)**.** *An* $\mathsf{AbelUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, W, S, U, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite radius-window row listing the dyadic subradius cells and boundary-approach windows inspected by the Abel consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite coefficient and partial-sum prefixes requested by R.*
3. *S is the Abel partial-sum row whose observations are read through the same finite windows.*
4. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *ledger row, assigning one displayed tail threshold to every radius cell in R.*
5. *T is the* $\mathsf{AbelTheorem}^{\uparrow}$ *boundary row recording the radial handoff after R, W, S, U have been accepted.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the synchronized Abel windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the Abel boundary and regular rational readback have been displayed.*
8. *H records componentwise* hsame *transport for radius windows, stream windows, partial sums, uniform-tail data, Abel boundary data, readback, seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each Abel uniform convergence request through R, W, S, U, T, Q, E.*

10. *P is the* Pkg *provenance over* $\mathsf{AbelUniformConvergence}^{\uparrow}$, $\mathsf{AbelTheorem}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelUniformConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, S, U, T, Q, E, H, C, P, N$. It has no coordinate for ambient real completeness, countable choice, quotient stream equality, selected infinite sums, or a Real seal detached from the displayed Abel uniform route.

Theorem 7086.2 (AbelUniformConvergence NameCert obligations)**.** *For every accepted* $\mathsf{AbelUniformConvergence}^{\uparrow}$ *carrier*

$$A = (R, W, S, U, T, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelUniformConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, radius-window admission through R, finite prefix exposure through W, Abel partial-sum alignment through S, uniform tail exactness through U, Abel boundary handoff through T, regular rational readback and Real seal nonescape through Q and E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation row supplies ambient real completeness, countable choice, quotient stream equality, selected infinite sums, host analytic equality, or an unbounded stream selector.*

Proof. Project the carrier of Definition 7086.1. Its mathematical rows are the radius-window row R, the stream window row W, the partial-sum row S, the uniform-tail ledger U, the Abel boundary row T, the regular rational readback Q, and the Real seal E.

The radius and stream rows fix the finite inspection surface. Every coefficient or partial-sum observation is read through W only after a displayed radius cell in R has been selected.

The uniform row supplies the common tail budget. Because U is a $\mathsf{UniformLimit}^{\uparrow}$ ledger over the same windows, the Abel boundary read in T cannot choose a different threshold for each point of the displayed radius window.

The readback and seal rows are downstream. The row Q repacks the synchronized Abel windows as regular rational data, and E exposes the terminal Real seal only after R, W, S, U, T, Q have been read.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for ambient real completeness, countable choice, quotient stream equality, selected infinite sums, host analytic equality, or an unbounded selector, the listed rows exhaust the local NameCert obligations. □

Theorem 7086.3 (AbelUniformConvergence Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{AbelUniformConvergence}^{\uparrow}$ *packet reads the endpoint only through the finite route*

$$\mathsf{AbelTheorem}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer first receives displayed radius windows, finite stream prefixes, one uniform tail ledger, and the Abel boundary handoff; only then may it consume the regular rational readback and terminal Real seal. The route exports no ambient real completeness theorem, countable-choice schedule, quotient stream equality, selected infinite sum, or host equality of boundary values.

Proof. Start from the accepted packet and read the displayed radius and stream rows R, W. These rows expose only finite windows, so the consumer has no independent stream quotient or unbounded selector.

Apply the uniform-tail row U. It supplies one threshold discipline for the whole displayed radius window, and therefore binds the Abel boundary read to the same finite window surface.

Read the Abel boundary row T through the sibling theorem route. The row records the radial handoff only after R, W, S, U have fixed the finite convergence surface.

Finally pass through $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal $\mathsf{Real}^{\uparrow}$ seal E, with H, C, P, N preserving the same route. The refused objects are absent from the carrier, so every Real-facing read remains inside the displayed finite package. □

Falsifiable prediction: Every accepted AbelUniformConvergenceUp read exposes a finite radius-window row, StreamNameUp prefix row, Abel partial-sum row, UniformLimitUp tail ledger, AbelTheoremUp boundary row, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a Real-facing consumer can inspect the endpoint.

Independence witness: abel-theorem, uniformlimit, streamname, regseqrat, real: $\mathsf{AbelUniformConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries radius-window admission, finite prefix exposure, Abel partial-sum alignment, one uniform tail ledger, Abel boundary handoff, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelUniformConvergence}^{\uparrow}}$.

Constructive story: An $\mathsf{AbelUniformConvergence}^{\uparrow}$ packet is generated from radius-window rows, StreamName finite prefixes, Abel partial sums, a UniformLimit tail ledger, an AbelTheorem boundary row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7086.1, Definition 7086.2, and Definition 7086.3 seed the finite Abel uniform convergence route over AbelTheorem, UniformLimit, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient real completeness, countable choice, quotient stream equality, selected infinite sums, host analytic equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.AbelUniformConvergenceUp covering carrier admission, radius-window admission, finite prefix exposure, uniform-tail exactness, Abel boundary handoff, readback, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 7087

A Concrete Naming Certificate for $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy sequences and finite extraction from regular data, the $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ packet records the finite subsequence route used by Real and Completion consumers. It reads the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert carrier for extracting and replaying a monotone subsequence window; it is not a quotient stream, a selected host limit, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7087.1 (Finite regular Cauchy subsequence carrier)**.** *A* $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, M, O, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite index-window row, naming the inspected source positions and the target subsequence positions visible to the packet.*
2. *M is the monotone index ledger, recording that the selected target positions are strictly ordered inside the displayed finite window.*
3. *O is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing only the stream-name observations read at the displayed indices.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the observations in O and the monotone ledger M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget used to compare the source window, selected positions, and regular rational readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after W, M, O, R, D have been displayed.*
7. *H records componentwise* hsame *transport for finite windows, monotone ledgers, StreamName observations, RegSeqRat readback, dyadic budgets, Real seals, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite extraction route*

$$W \longmapsto M \longmapsto O \longmapsto R \longmapsto D \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, M, O, R, D, E, H, C, P, N$. It has no coordinate for an infinite choice function, quotient stream equality, selected host limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FiniteRegularCauchySubsequenceUp.FiniteRegularCauchySubsequenceTasteGate`
`single carrier alignment`

Theorem 7087.2 (Finite regular Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ *carrier*

$$S = (W, M, O, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index window, monotone index ledger, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance budget, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; monotone-window exactness for the selected indices; regular readback compatibility through O, R, D; Real-seal routing through E; and nonescape for every excluded ambient sequence or completeness object.*

Proof. Carrier admission is projection from Definition 7087.1. The visible packet is the tuple $W, M, O, R, D, E, H, C, P, N$, and every obligation row names one of those coordinates.

The monotone-window obligation is local. The row M is the only place where index order is recorded, and it is attached to the finite window W before any StreamName observation can be read.

Regular readback compatibility then follows the displayed route. The row O supplies the inspected stream observations, R reads them as regular rational data, and D provides the dyadic tolerance budget used by the same finite extraction.

The Real seal is terminal. The row E is exposed only after W, M, O, R, D have been displayed, while H, C, P, N transport, replay, source, and name that same route. Since no excluded ambient object is a coordinate of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7087.3 (Finite regular Cauchy subsequence extraction route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ *carrier*

$$S = (W, M, O, R, D, E, H, C, P, N)$$

must factor its subsequence read through the finite route

$$W \longmapsto M \longmapsto O \longmapsto R \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a finite index window, a monotone index ledger, StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance budget, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It receives no infinite selector, quotient stream, selected host limit, countable choice, host-level sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. Before any analytic read is available, the packet exposes only the finite window W and the monotone ledger M for the selected indices.

Replay the observation route through C. The StreamName row O can be inspected only at the displayed indices, the RegSeqRat row R is read from those observations, and the DyadicRatCore budget D compares only the same finite data.

The terminal seal is then read from E. This seal is attached to the route already displayed by W, M, O, R, D, so it is a Real-facing seal of the finite extraction rather than a selected host limit.

Finally, H, C, P, N transport, replay, source, and name the same packet. Because the carrier contains no infinite selector, quotient stream, countable-choice row, host equality service, or ambient completeness coordinate, every public subsequence read factors through the stated finite route. □

Falsifiable prediction: Every accepted $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ Real or Completion-facing read factors through a finite index window, monotone index ledger, $\mathsf{StreamName}^{\uparrow}$ observation row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, real, bishop-completion-induction: $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite selected-index window, monotone ledger, observation row, regular readback, dyadic tolerance budget, terminal Real seal, transport, replay, provenance, and local naming while refusing infinite selectors, quotient streams, selected host limits, countable choice, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}$ packet is generated from a finite index window, monotone index ledger, StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance budget, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7087.1, Definition 7087.2, and Definition 7087.3 seed the finite subsequence extraction route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRegularCauchySubsequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteRegularCauchySubsequenceUp.FiniteRegularCauchySubsequenceTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close infinite subsequence selection, quotient stream equality, selected host limits, countable choice, host-level sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window exactness, StreamName observation routing, RegSeqRat readback, DyadicRatCore tolerance budgeting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRegularCauchySubsequenceUp.

Chapter 7088

A Concrete Naming Certificate for MetaCICTypedResidualBudget$^{\uparrow}$

The MetaCICTypedResidualBudget$^{\uparrow}$ packet realizes the MetaCIC residual-checker frontier as finite typed residual-budget data. It sits beside the MetaCIC residual-substitution boundary, confluence residual-checker, confluence residual envelope, local diamond witness, and substitution-diamond packets, but its own carrier is the typed budget ledger that a bounded residual checker consumes before it may report a local join or retain an obstruction. The packet does not claim subject reduction, a full parallel diamond, arbitrary Church–Rosser, or unrestricted residual-substitution compatibility.

The dependency route is row-sensitive. The typed-window row T is read with the residual-substitution premise visible through chapter 2181; the residual-frontier row R is checked against the bounded redex and checker inventory of chapter 3827; the checker-route row Q consumes the residual envelope of chapter 3921; the local-join row J is compared with the finite peak geometry of chapter 7042; and the obstruction row O retains the substitution boundary of chapter 6419 together with the frontier budget of chapter 4238. Thus the carrier records the typed windows, bounded checker routes, local joins, and obstruction rows that those sibling packets must read before any residual-diamond handoff is visible to a downstream confluence consumer.

Definition 7088.1 (MetaCIC typed residual-budget carrier)**.** *A* MetaCICTypedResidualBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, R, Q, J, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, naming the finite source term, target type annotation, context fragment, and closedness boundary visible to the residual checker.*
2. *R is the residual-frontier row, listing the redex and residual pairs that the bounded checker is allowed to inspect inside T.*
3. *Q is the checker-route row, recording the finite fuel, residual scheduling order, and local substitution-boundary reads used by the checker.*
4. *J is the local-join row, storing each displayed join candidate accepted by the bounded checker for the inspected residual pairs.*
5. *O is the obstruction row, retaining the exact typed residual pair, substitution-boundary failure, or fuel boundary when the checker does not close a local join.*

6. *H records componentwise* hsame *transport for typed windows, residual frontiers, checker routes, local joins, obstruction rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a residual-checker request through*

$$T \longmapsto R \longmapsto Q \longmapsto J, O.$$

8. *P is the* Pkg *provenance over* $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$*, the surrounding MetaCIC confluence and substitution-boundary packets,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedResidualBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, Q, J, O, H, C, P, N$. It has no coordinate for a subject-reduction theorem, full parallel diamond, arbitrary Church–Rosser theorem, unrestricted substitution compatibility, Lean verification, or bridge checking.

Theorem 7088.2 (MetaCIC typed residual-budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ *carrier*

$$B = (T, R, Q, J, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedResidualBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the typed window, residual frontier, checker route, local join row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; typed-window exactness for the source and type boundary; residual-frontier coverage for the inspected pairs; checker-route boundedness; local-join or obstruction retention for each inspected pair; and nonescape from the named residual-budget rows.*

Proof. Carrier admission is projection from Definition 7088.1. The accepted packet exposes exactly $T, R, Q, J, O, H, C, P, N$.

Typed-window exactness is read from T. The checker has no access to a larger context, unrecorded closedness proof, or hidden subject-reduction theorem.

Residual-frontier coverage and checker-route boundedness are local. The row R lists the residual pairs under inspection, and Q records the finite route and fuel used to inspect them.

Each inspected pair then lands in either J or O. A successful local check is stored as a displayed join candidate in J, while an unclosed pair, substitution-boundary failure, or fuel boundary remains visible in O.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the stronger confluence and substitution principles are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7088.3 (MetaCIC typed residual-budget residual-checker route)**.** *Every MetaCIC residual-checker consumer of an accepted* $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ *carrier*

$$B = (T, R, Q, J, O, H, C, P, N)$$

must factor its read through the finite typed-budget route

$$T \longmapsto R \longmapsto Q \longmapsto J, O \longmapsto H, C, P, N.$$

The consumer may export the displayed typed window, inspected residual frontier, bounded checker route, local join rows, retained obstruction rows, componentwise transport, replay, provenance, and local naming. It may not export subject reduction, a full parallel diamond, arbitrary Church–Rosser, unrestricted residual-substitution compatibility, Lean verification, or bridge checking.

Proof. Start with a residual-checker consumer. The typed window T is the only source of term, type, context, and closedness-boundary data available to the packet.

The residual frontier is then fixed by R. The checker cannot inspect a pair outside that displayed frontier, and the route row Q records the finite schedule and fuel used for the inspected pairs.

For each inspected pair, the packet records the outcome locally. A successful bounded local join is placed in J, while a residual pair that is blocked by substitution, typing, or fuel remains visible in O.

The structural rows add transport and replay without adding a stronger theorem. Thus every consumer export remains inside $T, R, Q, J, O, H, C, P, N$, and the excluded MetaCIC endpoints are not obtained from the typed residual-budget packet. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ residual-checker read factors through a typed-window row, residual-frontier row, bounded checker-route row, local-join row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-residual-checker, metacic-confluence-residual-envelope, metacic-parallel-diamond-witness, metacic-substitution-diamond: $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries typed-window data, residual-frontier data, checker-route budget, local joins, retained obstructions, transport, replay, provenance, and local naming while refusing subject reduction, full parallel diamond, arbitrary Church–Rosser, and unrestricted residual-substitution compatibility.

Later Lean TasteGate work for $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ must include explicit Nontrivial and FieldFaithful instances for the typed-window, residual-frontier, checker-route, local-join, obstruction, transport, replay, provenance, and NameCert rows before any discovery or public closure claim is made.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedResidualBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTypedResidualBudget}^{\uparrow}$ packet is generated from a typed-window row, residual-frontier row, bounded checker-route row, local-join row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedResidualBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7088.1, Definition 7088.2, and Definition 7088.3 seed the finite typed residual-budget surface for the MetaCIC residual-checker frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedResidualBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, full parallel diamond, arbitrary Church–Rosser, unrestricted residual-substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exactness, residual-frontier coverage, bounded checker-route exactness, local-join or obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedResidualBudgetUp.

Chapter 7089

A Concrete Naming Certificate for $\mathsf{RademacherFiniteDifference}^{\uparrow}$

Following Rademacher's finite-difference preparation for Lipschitz maps, the $\mathsf{RademacherFiniteDifference}^{\uparrow}$ packet isolates the endpoint-radius, quotient-window, and Lipschitz-bound data consumed by the $\mathsf{Rademacher}^{\uparrow}$ route of chapter 5146. It sits over the Lipschitz carrier of chapter 3762, the Real seal of chapter 38, the dyadic ledger of chapter 291, and the metric carrier of chapter 57. The chapter names a finite substrate for difference quotients; it does not claim almost-everywhere differentiability, measurable exceptional sets, selected derivatives, quotient completion, or a completed derivative space.

Definition 7089.1 (Rademacher finite-difference carrier)**.** *A* $\mathsf{RademacherFiniteDifference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, E, D, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LipschitzFunction}^{\uparrow}$ *row whose source metric, target metric, graph ledger, and Lipschitz bound are read by the finite-difference packet.*
2. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for base points, endpoint values, quotient values, and finite window radii.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, recording dyadic offsets, halved-window refinements, and tolerance comparisons.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *neighborhood row used to test source and target distance requests for the displayed Lipschitz window.*
5. *Q is the finite difference-quotient row, pairing displayed endpoint values with the dyadic denominator and the Real seal row that receives the quotient window.*
6. *H records componentwise* hsame *transport for L, E, D, M, Q, C, P, N.*
7. *C records displayed* Cont *replay from a finite Rademacher-facing request through D, M, L, Q and into E.*
8. *P is the* Pkg *provenance over* $\mathsf{RademacherFiniteDifference}^{\uparrow}$, $\mathsf{Rademacher}^{\uparrow}$, $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RademacherFiniteDifference}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, E, D, M, Q, H, C, P, N$. It has no coordinate for Lebesgue almost-everywhere differentiability, measurable null sets, selected derivatives, quotient completion points, countable-choice schedules, ambient measure completion, or completed derivative carriers.

Theorem 7089.2 (Rademacher finite-difference NameCert obligations)**.** *For every accepted* RademacherFiniteDiffer *carrier*

$$R = (L, E, D, M, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RademacherFiniteDifference}^{\uparrow}}$ *surface consists exactly of carrier admission, Lipschitz-row exposure, Real seal routing, dyadic endpoint-radius accounting, metric-neighborhood inspection, finite difference-quotient exposure, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 7089.1. The analytic rows are L, E, D, M, Q, and the structural rows are H, C, P, N.

The finite-difference route is row-bounded. The dyadic row supplies endpoint radii and denominator tolerances, the metric row supplies the neighborhoods in which the distance comparisons are read, and the Lipschitz row supplies the graph and bound ledger for the same window.

The quotient row Q only repackages the displayed endpoint values into a Real-facing window. Transport, replay, provenance, and local naming preserve the same coordinates, so the NameCert surface exports none of the excluded measurable, derivative-selection, quotient-completion, choice, or ambient-completion data. □

Theorem 7089.3 (Rademacher finite-difference Lipschitz window route)**.** *Every Lipschitz-window read exported by an accepted* $\mathsf{RademacherFiniteDifference}^{\uparrow}$ *carrier factors through the finite route*

$$D, M, L, Q \longmapsto E$$

before the $\mathsf{Rademacher}^{\uparrow}$ *consumer reads a derivative-candidate window. The route displays endpoint radii, metric neighborhoods, Lipschitz distance bounds, finite quotient rows, and terminal Real seals, but it exports no selected derivative, null exceptional set, quotient completion point, arbitrary measurable representative, or completed derivative space.*

Proof. Start with an accepted carrier $R = (L, E, D, M, Q, H, C, P, N)$. By Definition 7089.2, the Real-facing row E is reached only after the dyadic, metric, Lipschitz, and quotient rows have been displayed.

The Lipschitz window needs both endpoint control and distance control. The rows D and M supply those finite tests, while L supplies the bound that makes the quotient row Q readable as a finite window rather than as a global derivative.

Finally H, C, P, N transport, replay, source, and name the same route. Thus the downstream $\mathsf{Rademacher}^{\uparrow}$ chapter may consume a named finite-difference substrate without receiving any almost-everywhere, quotient-completion, selection, or ambient measure principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RademacherFiniteDifference}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RademacherFiniteDifference}^{\uparrow}}$, seed)

Scope closed: Definition 7089.1, Definition 7089.2, and Definition 7089.3 seed the finite-difference substrate over Lipschitz, Real, dyadic endpoint-radius, metric, quotient-window, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RademacherFiniteDifference}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lebesgue almost-everywhere differentiability, measurable exceptional sets, selected derivatives, quotient completion points, countable choice, ambient measure completion, completed derivative spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz-row exposure, Real seal routing, dyadic endpoint-radius accounting, metric-neighborhood inspection, finite quotient-window exposure, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RademacherFiniteDifferenceUp.

Constructive story: A RademacherFiniteDifference$^{\uparrow}$ packet is generated from a LipschitzFunction row, RealUp seal row, DyadicRatCore endpoint-radius ledger, MetricSpace neighborhood row, finite difference-quotient row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7090

A Concrete Naming Certificate for $\mathsf{RademacherTheorem}^{\uparrow}$

Following Rademacher and standard constructive real analysis, the $\mathsf{RademacherTheorem}^{\uparrow}$ packet records the theorem-level boundary at which a Lipschitz map may expose a weak differentiability route only after the finite derivative-candidate rows of chapter 5146 and the finite quotient-window rows of chapter 7089 have been displayed. It sits over the Lipschitz carrier of chapter 3762, the weak-derivative row of chapter 2210, the measure row of chapter 92, and the terminal real-name seal of chapter 38. The packet names a finite NameCert boundary for the theorem route; it does not select a derivative, identify quotient representatives, complete an ambient measure algebra, or assert a host-level differentiability theorem outside the displayed rows.

Definition 7090.1 (Rademacher theorem carrier)**.** *A* $\mathsf{RademacherTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, R, F, W, M, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LipschitzFunction}^{\uparrow}$ *source row, exposing the graph ledger, source and target metric reads, and Lipschitz bound that the theorem-level packet may inspect.*
2. *R is the* $\mathsf{Rademacher}^{\uparrow}$ *derivative-candidate row, recording only the finite dyadic and metric windows already exposed by the candidate packet.*
3. *F is the* $\mathsf{RademacherFiniteDifference}^{\uparrow}$ *quotient-window row, recording endpoint radii, finite difference quotients, and Real-facing window seals.*
4. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *row, naming the weak-derivative consumer boundary reached by the theorem packet after R and F have been displayed.*
5. *M is the* $\mathsf{Measure}^{\uparrow}$ *row, recording the displayed finite measure-facing exceptional-set interface without installing an ambient completed measure algebra.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for base values, quotient windows, weak-derivative test values, and finite tolerance comparisons.*
7. *H records componentwise* hsame *transport for $L, R, F, W, M, E, C, P, N$.*
8. *C records displayed* Cont *replay from a theorem-level request through the Lipschitz row, finite candidate row, quotient-window row, weak-derivative row, measure row, and Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RademacherTheorem}^{\uparrow}$, $\mathsf{Rademacher}^{\uparrow}$, $\mathsf{RademacherFiniteDifference}^{\uparrow}$, $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RademacherTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, F, W, M, E, H, C, P, N$. It has no coordinate for a selected pointwise derivative, quotient equality of representatives, completed measure spaces, arbitrary measurable representatives, countable-choice schedules, or host-level differentiability outside the displayed theorem boundary.

Theorem 7090.2 (Rademacher theorem NameCert obligations)**.** *For every accepted* $\mathsf{RademacherTheorem}^{\uparrow}$ *carrier*

$$T = (L, R, F, W, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RademacherTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission, Lipschitz-row exposure, finite derivative-candidate readback, finite difference-window readback, weak-derivative boundary routing, displayed measure-interface routing, Real seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 7090.1. The theorem-level analytic rows are L, R, F, W, M, E, and the support rows are H, C, P, N.

The Lipschitz source row L supplies only the displayed graph, metric, and bound ledger. The candidate row R and finite-difference row F then expose the dyadic windows, metric neighborhoods, endpoint radii, quotient windows, and Real seals already named by chapter 5146 and chapter 7089.

The weak-derivative row W is a downstream boundary row, not a chosen pointwise derivative. The measure row M records only the finite measure-facing interface needed to state where the weak route is read, not an ambient completed measure object.

The rows H and C transport and replay the same finite route, while P and N source and name it. Since no other coordinate occurs in the carrier, the certificate exports no selected derivative, quotient equality, arbitrary measurable representative, countable choice, completed measure algebra, or host differentiability theorem outside the displayed rows. □

Theorem 7090.3 (Rademacher Lipschitz weak-derivative route)**.** *Every weak-derivative-facing read exported by an accepted* $\mathsf{RademacherTheorem}^{\uparrow}$ *carrier factors through the finite route*

$$L \longmapsto R, F \longmapsto W, M, E \longmapsto H, C, P, N.$$

The route displays the Lipschitz source, finite derivative-candidate windows, finite quotient-window data, weak-derivative boundary, measure-facing interface, Real seal, transport, replay, provenance, and local naming rows before any analysis or PDE consumer may cite the theorem packet.

Proof. Start with an accepted carrier $T = (L, R, F, W, M, E, H, C, P, N)$. The carrier definition places the Lipschitz row first, so a consumer cannot ask for theorem-level weak differentiability before reading the source graph, metric, and bound data.

The next reads are R and F. They force the consumer through the existing Rademacher candidate packet and the finite-difference quotient-window packet, so the theorem route is downstream of displayed finite windows rather than a direct appeal to a host almost-everywhere theorem.

Only after those rows are present may the consumer read W, M, E. The weak-derivative boundary, measure-facing interface, and Real seal are therefore tied to the same finite route and do not install a selected derivative, quotient equality, arbitrary measurable representative, or completed measure algebra.

Finally H, C, P, N transport, replay, source, and name the route. Thus any downstream analysis or PDE chapter receives a theorem-level NameCert boundary, not data beyond the displayed Lipschitz, Rademacher, finite-difference, weak-derivative, measure, and Real rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RademacherTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RademacherTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7090.1, Definition 7090.2, and Definition 7090.3 seed the theorem-level boundary over Lipschitz, Rademacher candidate, finite-difference, weak-derivative, measure, Real, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RademacherTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected pointwise derivatives, quotient equality, arbitrary measurable representatives, countable choice, ambient measure completion, completed derivative spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz-row exposure, finite derivative-candidate readback, finite difference-window readback, weak-derivative routing, displayed measure-interface routing, Real seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RademacherTheoremUp.

Constructive story: A $\mathsf{RademacherTheorem}^{\uparrow}$ packet is generated from a LipschitzFunction row, Rademacher finite candidate row, RademacherFiniteDifference quotient-window row, WeakDerivative boundary row, Measure interface row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7091

A Concrete Naming Certificate for $\mathsf{CauchyRealAddition}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealAddition}^{\uparrow}$ packet records addition of Cauchy real names as a finite BEDC naming surface before any $\mathsf{Real}^{\uparrow}$ consumer is allowed to read a sum. It sits over the synchronized addition route of chapter 1695, the regular rational sequence surface of chapter 301, the finite window schedule of chapter 126, the dyadic tolerance arithmetic of chapter 291, and the terminal seal of chapter 38. The packet gives Real-facing algebra consumers a finite route from two Cauchy names to one sealed sum name; it does not assert quotient equality of real names, selected host limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a completed field operation.

Definition 7091.1 (Cauchy real addition carrier)**.** *A* $\mathsf{CauchyRealAddition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathsf{Real},+} = (A, R_0, R_1, W, D, S, E, Z, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *source certificate for the synchronized sum route.*
2. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *input-name rows read by the source certificate.*
3. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by the addition request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint-addition row.*
5. *S is the summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed by the synchronized route.*
6. *E is the finite tail ledger recording the two source tolerances, the dyadic addition tolerance, and the output tolerance.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, R_0, R_1, W, D, S, E are displayed.*
8. *H records componentwise* hsame *transport for the source certificate, input names, window schedule, dyadic row, summed readback, tail ledger, seal row, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from every Real-facing addition request through $A, R_0, R_1, W, D, S, E, Z, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyRealAddition}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealAddition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, R_0, R_1, W, D, S, E, Z, H, C, P, N$. The carrier has no coordinate for quotient equality of Cauchy names, host real addition, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completed-real algebra operation.*

Theorem 7091.2 (Cauchy real addition NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealAddition}^{\uparrow}$ *carrier*

$$A_{\mathsf{Real},+} = (A, R_0, R_1, W, D, S, E, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealAddition}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *source certificate A, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *input-name rows R_0, R_1, the paired* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint row D, the summed readback row S, the tail ledger E, the* $\mathsf{Real}^{\uparrow}$ *seal row Z, and the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 7091.1. Its analytic rows are $A, R_0, R_1, W, D, S, E, Z$, and its structural rows are H, C, P, N.

Read the source certificate first. The $\mathsf{RegularCauchyAddition}^{\uparrow}$ row A supplies the synchronized-window addition route, while R_0, R_1, W, D expose the two regular rational names, the paired window schedule, and the dyadic endpoint arithmetic used by that route.

The summed readback and seal rows are downstream. The ledger E records the finite tail accounting before S is consumed, and the $\mathsf{Real}^{\uparrow}$ row Z is exposed only after the summed regular rational name has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no other coordinate appears in the carrier, the local certificate has no hidden quotient-equality, host-arithmetic, selected-limit, trichotomy, or ambient-completeness obligation. □

Theorem 7091.3 (Cauchy real addition tail stability)**.** *Let two accepted* $\mathsf{CauchyRealAddition}^{\uparrow}$ *packets have matching* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *source certificates, input-name rows, paired finite-window schedules, dyadic tolerance rows, summed readbacks, tail ledgers, and* $\mathsf{Real}^{\uparrow}$ *seal rows through their displayed* hsame *data. Then every Real-facing sum read of the first packet transports to the corresponding sum read of the second packet through the same finite route $A, R_0, R_1, W, D, S, E, Z, H, C, P, N$.*

Proof. Start with the componentwise transport row H. It matches the source addition certificate A, the two input rows R_0, R_1, the window row W, and the dyadic tolerance row D.

Use the synchronized route supplied by chapter 1695. The row A keeps the pointwise sum tied to the paired windows and dyadic tolerance arithmetic, so transport cannot replace the finite tail route by host real addition.

The tail ledger and readback rows are preserved next. The ledger E remains the only displayed place where source and output tolerances are combined before the summed readback row S and the seal row Z are consumed.

Replay through C and repack through P, N. Every accepted Real-facing sum read is one of these displayed routes, so no quotient equality, selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle can enter under transport. □

Theorem 7091.4 (Cauchy real addition Real seal handoff)**.** *Every accepted* $\mathsf{CauchyRealAddition}^{\uparrow}$ *packet hands a sum to a* $\mathsf{Real}^{\uparrow}$ *consumer only through the finite route*

$$A \longmapsto (R_0, R_1) \longmapsto W \longmapsto D \longmapsto E \longmapsto S \longmapsto Z.$$

Thus the consumer receives a $\mathsf{RegularCauchyAddition}^{\uparrow}$ *source certificate, two* $\mathsf{RegSeqRat}^{\uparrow}$ *input-name rows, a paired* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, a finite*

tail ledger, a summed regular rational readback, and a terminal $\mathsf{Real}^{\uparrow}$ *seal; it receives no quotient real equality, host addition, selected host limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the carrier projection. The Real-facing addition request has no source rows other than $A, R_0, R_1, W, D, E, S, Z$.

Apply the source route of Definition 1695.5. It already orders the regular Cauchy addition read from input rows and paired windows through dyadic endpoint addition and finite regularity accounting before a $\mathsf{Real}^{\uparrow}$ seal is exposed.

The present packet records that route as A and exposes the same rows as R_0, R_1, W, D, E, S, Z. The structural rows H, C, P, N preserve the order of the handoff for downstream Real algebra consumers.

Therefore the packet supplies a finite Cauchy-name addition seal and nothing stronger. Quotient equality, host arithmetic, selected limits, trichotomy, and ambient completeness remain outside the exported surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealAddition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealAddition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7091.1, Definition 7091.2, Definition 7091.3, and Definition 7091.4 seed the finite Cauchy-real addition surface over a $\mathsf{RegularCauchyAddition}^{\uparrow}$ source certificate, two $\mathsf{RegSeqRat}^{\uparrow}$ names, paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance arithmetic, a summed readback row, and a terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealAddition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, host real addition, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-certificate exposure, tail-stability transport, Real seal handoff, replay, provenance, and local naming for BEDC.Derived.CauchyRealAdditionUp.

Constructive story: A $\mathsf{CauchyRealAddition}^{\uparrow}$ packet is generated from a $\mathsf{RegularCauchyAddition}^{\uparrow}$ source certificate, two $\mathsf{RegSeqRat}^{\uparrow}$ input rows, paired $\mathsf{StreamName}^{\uparrow}$ finite-window schedules, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance arithmetic, a tail ledger, a summed readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7092

A Concrete Naming Certificate for $\mathsf{EulerianGraph}^{\uparrow}$

Following Euler and standard finite graph theory, the $\mathsf{EulerianGraph}^{\uparrow}$ packet names the finite parity and edge-traversal ledger behind Eulerian closed trails. It reads its graph carrier from chapter 118, tree-style finite support traversal from chapter 119, path and endpoint rows from chapter 236, spanning support bookkeeping from chapter 237, matching-style incidence pairing from chapter 234, and residual edge-consumption ledgers from chapter 233. The packet is finite: it records vertices, edges, incidence parity, non-isolated connected support, a closed edge-traversal schedule, structural transport, replay, provenance, and local naming, without importing an ambient infinite graph, a host theorem of Eulerianity, quotient equality of trails, or a selected external traversal.

Definition 7092.1 (EulerianGraph finite trail carrier)**.** *An* $\mathsf{EulerianGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (G, I, P, S, T, M, H, C, Q, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *carrier row, including finite vertex rows, finite edge rows, endpoint readback, and componentwise graph classification.*
2. *I is the incidence ledger, listing for each vertex the incident edge occurrences and their endpoint orientation as finite rows.*
3. *P is the even-degree parity row, pairing the incident edge occurrences at every vertex and exposing the empty-pair branch for isolated vertices.*
4. *S is the connected non-isolated support row, recording finite reachability among all vertices incident to at least one edge.*
5. *T is the edge-traversal schedule, a finite ordered row of edge occurrences whose edge projection lists every edge exactly once.*
6. *M is the closed-trail endpoint-match row, recording that consecutive entries of T share the displayed vertex endpoint and that the final endpoint returns to the initial endpoint.*
7. *H is componentwise* hsame *transport for graph, incidence, parity, support, schedule, endpoint-match, replay, provenance, and local naming rows.*
8. *C is the displayed* Cont *replay row forcing every Eulerian read through G, I, P, S, T, M, H.*

9. *Q is the* Pkg *provenance over* $\mathsf{EulerianGraph}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{ShortestPath}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerianGraph}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows and their displayed transports.

Theorem 7092.2 (EulerianGraph NameCert obligations)**.** *For every accepted* $\mathsf{EulerianGraph}^{\uparrow}$ *carrier* $E = (G, I, P, S, T, M, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EulerianGraph}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the graph carrier, incidence ledger, even-degree parity row, connected non-isolated support row, traversal schedule, endpoint-match row, transport, replay, provenance, and local name;*
2. *incidence exactness, requiring every edge occurrence in I to read its two displayed endpoints from the* $\mathsf{Graph}^{\uparrow}$ *row G;*
3. *parity exactness, requiring the incident edge occurrences at each vertex to be partitioned into displayed pairs by P;*
4. *support connectedness, requiring every non-isolated vertex read to factor through the finite support row S;*
5. *traversal coverage, requiring T to list every displayed edge exactly once and no unlisted edge;*
6. *closed endpoint matching, requiring M to match consecutive endpoint rows in T and to identify the final endpoint with the initial endpoint;*
7. *structural stability, requiring H, C, Q, N to transport, replay, source, and name only the rows already listed.*

No Eulerian trail obligation for this packet is stored outside those finite rows.

Proof. Project the carrier of Definition 7092.1. The certificate can inspect only $G, I, P, S, T, M, H, C, Q, N$.

The graph and incidence obligations read G and I. Each incidence entry is tied to a displayed edge and its two endpoint rows, so the certificate cannot consult a hidden adjacency relation.

The parity and support obligations read P and S. The row P pairs incident occurrences at each vertex, while S supplies finite connected support for the non-isolated vertices.

The traversal obligations read T and M. Coverage is a finite ledger comparison between the edge projection of T and the edge rows of G, and closedness is the displayed endpoint match along consecutive entries and the terminal return.

Finally H, C, Q, N only transport, replay, source, and name the same finite rows. Thus the local NameCert surface exports no host Euler theorem, hidden graph equality, quotient trail, selected external traversal, or infinite graph data. □

Theorem 7092.3 (EulerianGraph even-degree trail exhaustion)**.** *Let $E = (G, I, P, S, T, M, H, C, Q, N)$ be an accepted* $\mathsf{EulerianGraph}^{\uparrow}$ *carrier. Every edge-traversal read of E is exhausted by finite induction on the schedule row T: each step consumes one displayed edge occurrence from I, preserves the parity pairing recorded by P at the two touched endpoints, and leaves the remaining unread occurrences inside the same connected non-isolated support row S until the final endpoint-match row M closes the trail.*

Proof. Induct over the finite schedule row T. In the empty branch, traversal coverage says that no displayed edge remains unread, and the endpoint-match row supplies the degenerate closed endpoint read.

For the successor branch, read the head edge occurrence through I. Its two endpoints are displayed by G, and its incidence memberships are part of the paired local parity ledger P.

Remove the head occurrence from the unread traversal ledger. The remaining entries are still rows of T, and support connectedness keeps every non-isolated endpoint inside S; no external search over graph edges is available.

At the terminal branch, the endpoint-match row M identifies the last endpoint with the first. The induction has consumed each scheduled edge exactly once, and traversal coverage identifies the scheduled edge projection with the displayed edge ledger of G. □

Theorem 7092.4 (EulerianGraph sibling dependency route)**.** *Every downstream consumer of an accepted* $\mathsf{EulerianGraph}^{\uparrow}$ *packet factors through the finite sibling route*

$$\mathsf{Graph}^{\uparrow},\ \mathsf{Tree}^{\uparrow},\ \mathsf{ShortestPath}^{\uparrow},\ \mathsf{SpanningTree}^{\uparrow},\ \mathsf{Matching}^{\uparrow},\ \mathsf{NetworkFlow}^{\uparrow} \longmapsto \mathsf{EulerianGraph}^{\uparrow}.$$

The graph row supplies endpoints and incidence, the tree and spanning-tree rows supply finite support traversal, the shortest-path row supplies endpoint reachability windows, the matching row supplies local parity pairings, and the network-flow row supplies residual edge-consumption bookkeeping. The consumer receives no host Eulerian graph object, hidden connectedness proof, selected external tour, quotient trail equality, or ambient infinite graph.

Proof. Begin with a consumer read of the Eulerian schedule. By Definition 7092.2, the read first exposes $G, I, P, S, T, M, H, C, Q, N$.

The row G is the entry point for finite vertices, edges, and endpoints. The rows S and M then use tree, spanning-tree, and shortest-path style support data only as finite traversal and endpoint windows over that same graph carrier.

The row P is the matching-style parity ledger: incident edge occurrences are paired locally before the schedule is accepted. The row T is the residual-consumption ledger: each edge is consumed exactly once, in the same finite bookkeeping style used by the network-flow surface.

Since H, C, Q, N preserve only this displayed route, every downstream read factors through the named sibling surfaces and cannot obtain any of the excluded host-level data. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerianGraph}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerianGraph}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7092.1, Definition 7092.2, Definition 7092.3, and Definition 7092.4 seed the finite Eulerian graph certificate route over graph, incidence, parity, support, traversal, endpoint-match, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerianGraph}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, standard bridge checking, infinite graph theory, selected external tours, quotient trail equality, host Eulerianity, or ambient connectedness principles.

Upgrade path: Obligation closure requires checked carrier admission, incidence exactness, parity exactness, support connectedness, traversal coverage, closed endpoint matching, sibling dependency routing, transport, replay, provenance, and local naming for BEDC.Derived.EulerianGraphUp.

Constructive story: An $\mathsf{EulerianGraph}^{\uparrow}$ packet is generated from a displayed GraphUp carrier, incidence rows, even-degree parity pairs, connected non-isolated support, an edge-traversal schedule, closed endpoint matching, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7093

A Concrete Naming Certificate for $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$

Following Vitali and Bishop–Bridges constructive covering practice, the $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ packet records the finite gauge-controlled Vitali selection surface used before a compactness, differentiation, or integration consumer reads a covering theorem. It sits after the gauge presentation of chapter 4302, the finite subcover ledger of chapter 6125, the finite stream windows of chapter 126, the regular rational readback rows of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite gauge requests, candidate cells, disjoint selected subfamilies, finite cover handoff rows, rational readback, real sealing, transport, replay, provenance, and local naming; it does not import countable choice, measure completion, host selection, an ambient Vitali theorem, quotient point equality, or a selected infinite subcover.

Definition 7093.1 (Vitali covering finite-gauge carrier)**.** *A* $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (X, G, W, D, A, S, L, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-history source row whose local neighborhoods are inspected.*
2. *G is the* $\mathsf{GaugeSpace}^{\uparrow}$ *row naming the finite gauge family and tolerance requests.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the displayed request schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare gauge radii and window bounds.*
5. *A is the admissible candidate-cell row: each candidate carries its point, gauge row, positive radius, and membership witness.*
6. *S is the selected disjoint finite subfamily row, stored as a closed* $\mathsf{ProbeBundle}$ *over candidates from A.*
7. *L is the local coverage ledger, recording which displayed source windows are covered by the selected subfamily after the gauge comparison.*
8. *F is the* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *handoff row for compact or finite-cover consumers.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational radius and tolerance comparisons.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for any real-valued radius or size read.*

11. *H records componentwise* hsame *transport for source, gauge, window, tolerance, candidate, selection, coverage, subcover, readback, seal, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from a finite Vitali request through the gauge and selected-subfamily rows into the finite-cover handoff.*
13. *P is the* Pkg *provenance over* $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$, $\mathsf{GaugeSpace}^{\uparrow}$, $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{VitaliCover}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for countable selections, measure completion, host-level Vitali choice, quotient point equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7093.2 (Vitali covering finite-gauge NameCert obligations)**.** *For every accepted* VitaliCoveringFiniteGaug *carrier*

$$V = (X, G, W, D, A, S, L, F, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the point source, gauge row, finite stream windows, dyadic tolerance ledger, admissible candidate cells, selected disjoint finite subfamily, local coverage ledger, finite subcover handoff, regular-rational readback, real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is discharged by countable choice, measure completion, host selection, quotient point equality, selected representatives, or ambient real completeness.*

Proof. Project the carrier of Definition 7093.1. The source row X, gauge row G, finite windows W, and dyadic ledger D are the only inputs available before candidate cells can be admitted.

Read the candidate and selection rows. The row A lists displayed gauge-admissible cells, while S stores only a finite disjoint subfamily drawn from those cells. The local coverage ledger L records what the selected subfamily covers in the displayed finite windows.

The handoff rows F, R, E then expose finite subcover consumption, rational readback, and real sealing. The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains none of the excluded choice, measure-completion, host-selection, quotient, representative, or completeness coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 7093.3 (Vitali covering finite-gauge disjoint-selection handoff)**.** *Every* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{VitaliCover}^{\uparrow}$, *or* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *consumer of an accepted* $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{GaugeSpace}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the selected disjoint subfamily row S and the coverage handoff row L are public. The consumer receives a displayed finite selection-and-coverage packet, not a countable-choice selector, measure completion, host Vitali theorem, quotient point equality, or ambient compactness principle.

Proof. Begin with a finite-cover consumer. By Definition 7093.2, the consumer must inspect the gauge, window, dyadic, readback, and real-seal rows before the selected subfamily and coverage ledger are available.

The Vitali-facing siblings enter only through provenance and handoff rows. $\mathsf{VitaliCover}^{\uparrow}$ supplies the disjoint-selection discipline, $\mathsf{FiniteVitaliCover}^{\uparrow}$ supplies the finite shadow-cover route, and $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ consumes the displayed finite coverage row. None of these siblings replaces the carrier rows G, W, D, A, S, L, R, E.

Thus the handoff is the finite route displayed above. Because the packet has no coordinate for a countable selector, measure completion, host theorem, quotient point equality, or ambient compactness, no consumer can read such data from the $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ surface. □

Falsifiable prediction: Every accepted $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ read displays one source row, one GaugeSpaceUp row, one StreamNameUp window route, one DyadicRatCoreUp tolerance ledger, one admissible candidate-cell row, one selected disjoint finite subfamily, one local coverage ledger, one FiniteSubcoverLedgerUp handoff, one RegSeqRatUp readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: gauge-space, finite-subcover-ledger, vitali-cover, finite-vitali-cover, stream-name, regseqrat, real: $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite gauge requests, candidate cells, disjoint selected subfamilies, finite coverage handoff, rational readback, real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}$ packet is generated from a point source, GaugeSpace row, StreamName finite windows, DyadicRatCore tolerance ledger, admissible candidate cells, selected disjoint finite subfamily, local coverage ledger, FiniteSubcoverLedger handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7093.1, Definition 7093.2, and Definition 7093.3 seed the finite gauge-controlled Vitali covering packet over GaugeSpaceUp, FiniteSubcoverLedgerUp, VitaliCoverUp, FiniteVitaliCoverUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VitaliCoveringFiniteGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable Vitali selection, measure completion, host selection, quotient point equality, selected representatives, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge admissibility, finite-window routing, dyadic tolerance exactness, disjoint selected-subfamily exposure, coverage handoff, finite-subcover consumption, readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.VitaliCoveringFiniteGaugeUp.

Chapter 7094

A Concrete Naming Certificate for $\mathsf{BishopRealArithmetic}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealArithmetic}^{\uparrow}$ packet records the constructive arithmetic interface for regular Cauchy real names as finite BEDC NameCert data. It sits after the dyadic rational ledger of chapter 291, the finite stream-window route of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, the regular-Cauchy addition route of chapter 1695, and the regular-Cauchy multiplication route of chapter 1702. The packet gives $\mathsf{RealAlgOrder}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ consumers one public arithmetic route over displayed regular Cauchy names, not separate host operations.

Definition 7094.1 (Bishop real arithmetic carrier)**.** *A* $\mathsf{BishopRealArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, S_X, S_Y, R_X, R_Y, D, +_A, \cdot_A, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two source* $\mathsf{Real}^{\uparrow}$ *name rows, each read only through a displayed regular Cauchy route.*
2. *S_X and S_Y are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows requested for the two source names.*
3. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and error ledger used by both arithmetic operations.*
5. *$+_A$ is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *row, available only after $X, Y, S_X, S_Y, R_X, R_Y, D$ have been displayed.*
6. *$\cdot_A$ is the* $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ *row, available only after the same source rows and the displayed product-bound entries in D have been displayed.*
7. *E is the output* $\mathsf{Real}^{\uparrow}$ *seal for the requested arithmetic read.*
8. *H records componentwise* hsame *transport for the source names, windows, readbacks, dyadic ledger, addition row, multiplication row, output seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from an arithmetic request through the source rows, finite windows, readbacks, dyadic budget, operation row, output seal, transport, package provenance, and local name.*

10. *P is the* Pkg *provenance over* BishopRealArithmetic$^\uparrow$, DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, RegularCauchyAddition$^\uparrow$, RegularCauchyMultiplication$^\uparrow$, RealAlgOrder$^\uparrow$, Met$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BishopRealArithmetic}^\uparrow}$ *row.*

The carrier predicate asserts exactly that these rows form one regular Cauchy arithmetic packet. It has no coordinate for quotient real equality, selected limits, trichotomy, ambient Real$^\uparrow$ *completeness, host real addition, or host real multiplication.*

Theorem 7094.2 (Bishop real arithmetic NameCert obligations)**.** *For every accepted* BishopRealArithmetic$^\uparrow$ *carrier*

$$A = (X, Y, S_X, S_Y, R_X, R_Y, D, +_A, \cdot_A, E, H, C, P, N),$$

the local NameCert$_{\text{BishopRealArithmetic}^\uparrow}$ *surface consists exactly of carrier admission for the two displayed* Real$^\uparrow$ *source rows, finite* StreamName$^\uparrow$ *windows,* RegSeqRat$^\uparrow$ *readbacks, the* DyadicRatCore$^\uparrow$ *tolerance ledger, the* RegularCauchyAddition$^\uparrow$ *row, the* RegularCauchyMultiplication$^\uparrow$ *row, the output* Real$^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* RealAlgOrder$^\uparrow$ *or* Met$^\uparrow$ *consumer of arithmetic on regular Cauchy real names must read through these rows before it consumes an operation result.*

Proof. Project the accepted carrier of Definition 7094.1. The source data are the two real-name rows, their finite stream windows, their regular rational readbacks, and the shared dyadic tolerance ledger.

The addition and multiplication rows are downstream of those finite rows. The addition row cites the regular-Cauchy addition chapter, while the multiplication row also uses the displayed product-bound entries in the dyadic ledger before the output seal is available.

The output seal E is therefore reached only through the finite regular Cauchy arithmetic route. Componentwise transport, continuation replay, provenance, and local naming preserve the same displayed rows for downstream RealAlgOrder$^\uparrow$ and Met$^\uparrow$ consumers.

Since the carrier has no coordinate for quotient real equality, selected limits, trichotomy, ambient Real$^\uparrow$ completeness, host real addition, or host real multiplication, those objects cannot become local NameCert obligations. □

Theorem 7094.3 (Bishop real arithmetic downstream route)**.** *Every* RealAlgOrder$^\uparrow$ *or* Met$^\uparrow$ *consumer of an accepted* BishopRealArithmetic$^\uparrow$ *packet reads arithmetic data only through the finite route*

$$X, Y, S_X, S_Y, R_X, R_Y, D, +_A, \cdot_A, E, H, C, P, N.$$

The route supplies regular Cauchy addition and multiplication over displayed real-name rows, and exports no quotient real equality, selected limits, trichotomy, ambient Real$^\uparrow$ *completeness, or host arithmetic operation.*

Proof. Start with an arithmetic consumer. The consumer first sees the displayed real-name sources and their finite windows, not host representatives of completed real numbers.

The regular rational readbacks and dyadic tolerance ledger are read before either arithmetic row. Thus the addition and multiplication entries are finite regular Cauchy operation rows rather than detached operations on an ambient real carrier.

Finally the output seal and structural rows package the same finite route for RealAlgOrder$^\uparrow$ and Met$^\uparrow$. No carrier row supplies quotient equality, selected limits, trichotomy, ambient completeness, or host arithmetic. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealArithmetic}^{\uparrow}$ arithmetic read displays two Real source rows, two finite StreamName windows, two RegSeqRat readbacks, one DyadicRatCore tolerance ledger, a RegularCauchyAddition row, a RegularCauchyMultiplication row, one output Real seal, and local transport, replay, provenance, and naming rows before a RealAlgOrder or MetricSpace consumer receives the result.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-addition, regular-cauchy-multiplication: $\mathsf{BishopRealArithmetic}^{\uparrow}$ is not bijective to any listed sibling because it binds both regular Cauchy operation rows to one displayed dyadic budget, output real seal, structural transport, continuation replay, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealArithmetic}^{\uparrow}$ packet is generated from two Real name rows, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, RegularCauchyAddition and RegularCauchyMultiplication rows, an output Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7094.1, Definition 7094.2, and Definition 7094.3 seed the public arithmetic interface over regular Cauchy real names for RealAlgOrder and MetricSpace consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealArithmetic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, ambient Real completeness, host arithmetic, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for the finite arithmetic NameCert surface over BEDC.Derived.BishopRealArithmeticUp.

Chapter 7095

A Concrete Naming Certificate for $\mathsf{WeakStarFiniteWindow}^{\uparrow}$

Following Banach–Alaoglu and Kelley functional analysis, the $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ packet records the finite observation surface that a weak-star compactness route may inspect before any full compactness theorem is invoked. It sits between the weak-star topology window of chapter 5192, the finite dual-ball handoff of chapter 3420, the locally convex source rows of chapter 3386, the compact metric consumer rows of chapter 129, and the scalar seal of chapter 38. The packet names only finite dual-ball probes, sampled weak-star coordinates, and tolerance windows; it does not assert full weak-star compactness, choose subnets, construct quotient dual spaces, or import ambient completeness.

Definition 7095.1 (Weak-star finite-window carrier)**.** *A* $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, E, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite dual-ball probe row, listing the displayed bounded functional probes and source points admitted from* $\mathsf{BanachAlaoglu}^{\uparrow}$ *and* $\mathsf{WeakStarTopology}^{\uparrow}$*.*
2. *S is the sampled weak-star coordinate row, recording only the scalar values read from the finite probes in D.*
3. *E is the* $\mathsf{Real}^{\uparrow}$ *tolerance-window row, containing the finite scalar radii and comparison bounds used by the observation.*
4. *T records componentwise* hsame *transport for the dual-ball probes, sampled coordinates, tolerance windows, replay, provenance, and local naming rows.*
5. *H records the displayed finite-window history, including the locally convex source and compact-metric consumer context visible to the packet.*
6. *C records displayed* Cont *replay from a weak-star observation request through*

$$D \longmapsto S \longmapsto E \longmapsto T, H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{WeakStarFiniteWindow}^{\uparrow}$*,* $\mathsf{BanachAlaoglu}^{\uparrow}$*,* $\mathsf{WeakStarTopology}^{\uparrow}$*,* $\mathsf{LocallyConvexSpace}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{WeakStarFiniteWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through D, S, E, T, H, C, P, N. It has no coordinate for full weak-star compactness, choice-selected subnets, quotient dual spaces, or ambient completeness.

Theorem 7095.2 (Weak-star finite-window NameCert obligations)**.** *For every accepted* $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ *carrier $W = (D, S, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakStarFiniteWindow}^{\uparrow}}$ *surface consists exactly of finite dual-ball probe admission through D, sampled weak-star coordinate admission through S,* $\mathsf{Real}^{\uparrow}$ *tolerance-window admission through E, componentwise transport through T, finite-window history exposure through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The finite rows D, S, E, T, H, C, P, N are the entire public inventory of the packet.

The dual-ball and topology data enter only through D. The sampled coordinates are then read through S, and every scalar comparison must pass through the $\mathsf{Real}^{\uparrow}$ tolerance row E.

The structural rows do not add compactness or selection data. Transport, history, replay, provenance, and naming preserve the same finite route, so the NameCert surface is exhausted by the listed rows. □

Theorem 7095.3 (Weak-star finite-window Banach–Alaoglu route)**.** *Every Banach–Alaoglu, weak-star topology, locally convex, compact metric, or Real-and-Completion consumer that reads an accepted* $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ *carrier must factor its request through*

$$D \longmapsto S \longmapsto E \longmapsto T, H, C, P, N.$$

The consumer receives finite dual-ball probes, sampled weak-star coordinates, $\mathsf{Real}^{\uparrow}$ *tolerance windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It receives no full weak-star compactness theorem, no choice-selected subnet, no quotient dual space, and no ambient completeness principle.*

Proof. Start with a compactness-facing or duality-facing request. The finite probe row D must be exposed before any scalar coordinate is visible, so the consumer cannot read an unrestricted dual ball or a completed function space.

The sampled weak-star coordinate row S and the tolerance row E then determine the only scalar comparisons available to the request. These rows are finite and pass through the displayed $\mathsf{Real}^{\uparrow}$ seal.

Finally, T, H, C, P, N transport, replay, source, and name the same finite observation. Since none of the carrier rows is a subnet selector, quotient-dual construction, full weak-star compactness theorem, or ambient completeness principle, the route remains the finite Banach–Alaoglu window named in the statement. □

Falsifiable prediction: Every public $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ read factors through finite dual-ball probes, sampled weak-star coordinates, a $\mathsf{Real}^{\uparrow}$ tolerance window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming. A public read that exports full weak-star compactness, a choice-selected subnet, quotient dual space data, or ambient completeness from this packet refutes the carrier boundary.

Independence witness: banach-alaoglu, weak-star-topology, locally-convex-space, compactmetric, real: $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries

finite dual-ball probes, sampled weak-star coordinates, tolerance windows, history, transport, replay, provenance, and local naming while refusing full compactness, subnet choice, quotient dual spaces, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakStarFiniteWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakStarFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7095.1, Definition 7095.2, and Definition 7095.3 seed the finite weak-star observation surface for dual-ball and compactness-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakStarFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full weak-star compactness, choice-selected subnets, quotient dual spaces, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite dual-ball probe admission, sampled coordinate exactness, Real tolerance-window routing, transport, replay, provenance, and local naming for BEDC.Derived.WeakStarFiniteWindowUp.

Constructive story: A $\mathsf{WeakStarFiniteWindow}^{\uparrow}$ packet is generated from finite dual-ball probes, sampled weak-star coordinates, Real tolerance windows, componentwise hsame transport, finite-window history, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7096

A Concrete Naming Certificate for BetaSubstitutionPreservationFrontier$^{\uparrow}$

Following the standard subject-reduction theorem pattern for typed lambda calculi, the BetaSubstitutionPreservatior packet records the finite frontier at which the beta-redex case must present substitution-preservation data to the MetaCIC subject-reduction interface. It refines the beta obligation of Definition 9377.1, reads the concrete beta-row packet of chapter 716, and sits beside the four-row discharge surfaces of chapter 718 and chapter 712. The packet names a frontier and handoff ledger; it does not prove host substitution semantics, unrestricted conversion, the other three discharge obligations, or closed subject reduction.

Definition 7096.1 (Beta-substitution preservation frontier carrier)**.** *A* BetaSubstitutionPreservationFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, A, R, C, L, D, H, K, P, N)$$

with the following displayed rows:

1. *B is the beta-redex source row, recording the lambda body, shifted-domain context, argument, and displayed beta site.*
2. *A is the accepted* BetaSubstitutionPreservation$^{\uparrow}$ *row, read only through the carrier and handoff theorems of chapter 716.*
3. *R is the substituted-body and substituted-codomain readback row exposed at the frontier.*
4. *C is the codomain-boundary row, keeping the pre-substitution and post-substitution codomain reads paired.*
5. *L is the frontier ledger tying B, A, R, C to the beta row of the parameterised subject-reduction discharge interface.*
6. *D is the discharge-slot row, recording that the frontier supplies only the beta slot of the four-obligation interface named in section 9377.3.*
7. *H records componentwise* hsame *transport for beta source, accepted beta packet, readback, codomain boundary, frontier ledger, discharge slot, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from a subject-reduction beta request through*

$$B \longmapsto A \longmapsto R, C \longmapsto L, D.$$

9. *P is the* Pkg *provenance over* BetaSubstitutionPreservationFrontier$^{\uparrow}$, BetaSubstitutionPreservation$^{\uparrow}$, SubjectReduction$^{\uparrow}$, SubjectReductionDischarge$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BetaSubstitutionPreservationFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, R, C, L, D, H, K, P, N$. It has no coordinate for application-argument stability, lambda-domain preservation, Pi-domain preservation, unrestricted conversion, host substitution semantics, or closed subject reduction.

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationFrontierUp.TasteGate.BetaSubstitutionPreservationFront`
`single carrier alignment`

Theorem 7096.2 (Beta-substitution preservation frontier NameCert obligations)**.** *For every accepted* BetaSubstitutionPreservationFrontier$^{\uparrow}$ *carrier $F = (B, A, R, C, L, D, H, K, P, N)$, the local* NameCert$_{\text{BetaSubstitutionPreservationFrontier}^{\uparrow}}$ *surface consists exactly of beta-redex source admission through B, accepted beta-packet admission through A, substituted readback exposure through R, codomain-boundary pairing through C, frontier-ledger exactness through L, beta discharge-slot exposure through D, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The visible rows are precisely $B, A, R, C, L, D, H, K, P, N$.

The beta source row B supplies the redex boundary, while A must be read through the existing BetaSubstitutionPreservation$^{\uparrow}$ packet. The readback row R and codomain-boundary row C are therefore frontier projections of the accepted beta packet, not host substitution semantics.

The ledger L and discharge row D place this frontier in the beta slot of the four-obligation interface. Transport, replay, provenance, and naming preserve the same finite route. Since no row names the other three discharge obligations or closed subject reduction, the listed obligations exhaust the local NameCert surface. □

Theorem 7096.3 (Beta-substitution preservation subject-reduction frontier route)**.** *Every subject-reduction consumer that reads an accepted* BetaSubstitutionPreservationFrontier$^{\uparrow}$ *carrier must factor the beta-redex request through*

$$B \longmapsto A \longmapsto R, C \longmapsto L, D \longmapsto H, K, P, N.$$

The route supplies only the beta-substitution preservation frontier row of the MetaCIC four-obligation interface. It does not supply application-argument stability, lambda-domain subject reduction, Pi-domain subject reduction, unrestricted conversion, host substitution semantics, or the closed subject-reduction theorem blocked by chapter 9377.

Proof. Begin with a subject-reduction beta request. The request first exposes B, so the lambda body, shifted-domain context, argument, and beta site are displayed before any handoff is visible.

Next read A through chapter 716. The rows R and C are the substituted readback and codomain-boundary projections of that accepted beta packet.

The frontier ledger L and discharge-slot row D identify the supplied row as the beta row of the four-obligation interface. They do not contain the application-argument, lambda-domain, or Pi-domain rows.

Finally H, K, P, N transport, replay, source, and name this finite frontier. The absence of host substitution semantics, unrestricted conversion, and closed subject reduction from the carrier prevents any stronger read. □

Falsifiable prediction: Every public $\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}$ read factors through a beta-redex source row, an accepted $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ row, substituted readback, codomain-boundary pairing, frontier ledger, beta discharge slot, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming. A public read that exports the other discharge rows, host substitution semantics, unrestricted conversion, or closed subject reduction from this packet refutes the frontier boundary.

Independence witness: betasubstitutionpreservation, subjectreductiondischarge, metacic-subject-reduction-obligation: $\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the beta-redex source, accepted beta packet, substituted readback, codomain-boundary pairing, frontier ledger, discharge-slot row, transport, replay, provenance, and local naming while refusing the other discharge obligations and closed subject reduction.

Closure status for $\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7096.1, Definition 7096.2, and Definition 7096.3 seed the finite beta-redex frontier consumed by the MetaCIC subject-reduction discharge interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close application-argument stability, lambda-domain subject reduction, Pi-domain subject reduction, host substitution semantics, unrestricted conversion, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, beta-packet admission, substituted readback exposure, codomain-boundary pairing, frontier-ledger exactness, beta discharge-slot routing, transport, replay, provenance, and local naming for BEDC.Derived.BetaSubstitutionPreservationFrontierUp.

Constructive story: A $\mathsf{BetaSubstitutionPreservationFrontier}^{\uparrow}$ packet is generated from a beta-redex source row, an accepted BetaSubstitutionPreservation row, substituted readback, codomain-boundary pairing, a frontier ledger, beta discharge-slot exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7097

A Concrete Naming Certificate for $\mathsf{MatroidDual}^{\uparrow}$

Following Whitney, the $\mathsf{MatroidDual}^{\uparrow}$ packet records matroid duality as finite complement-basis and rank-nullity NameCert data. It reads the independent-set and exchange rows of chapter 202, the meet/join dependency surface of chapter 55, and the finite algebraic transport discipline of chapter 41. The packet names a dual carrier over the same finite ground; it does not import a host duality theorem, quotient ground equality, selected bases outside the displayed finite rows, or an ambient rank oracle.

Definition 7097.1 (Matroid dual carrier)**.** *A* $\mathsf{MatroidDual}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, M, B, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite ground row, read through the same* $\mathsf{FinSet}^{\uparrow}$ *ground discipline used by* $\mathsf{Matroid}^{\uparrow}$*.*
2. *M is the source* $\mathsf{Matroid}^{\uparrow}$ *row, including independent-set, hereditary, and exchange data.*
3. *B is the finite basis row for the source matroid, recording maximal independent subsets over the displayed ground.*
4. *Q is the complement-basis row, assigning to each displayed source basis its finite complement in E and reading that complement as a dual basis.*
5. *R is the rank/nullity ledger, recording the finite rank of source basis rows and the nullity read used by the dual packet.*
6. *U is the dual independence row, generated from finite subsets of complement-basis rows and transported through the source exchange route.*
7. *H records componentwise* hsame *transport for ground, source matroid, basis, complement-basis, rank/nullity, dual independence, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for every consumer that reads the dual packet.*
9. *P is the* Pkg *provenance over* $\mathsf{MatroidDual}^{\uparrow}$*,* $\mathsf{Matroid}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{Lattice}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host dual matroid, a quotient ground set, a selected global basis family, an external rank function, or an ambient linear representation.

Theorem 7097.2 (Matroid dual NameCert obligations)**.** *For every accepted* $\mathsf{MatroidDual}^{\uparrow}$ *carrier*

$$D = (E, M, B, Q, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}$ *surface consists exactly of carrier admission for* D*, finite ground exposure through* E*, source* $\mathsf{Matroid}^{\uparrow}$ *exposure through* M*, source-basis exposure through* B*, complement-basis exposure through* Q*, rank/nullity ledger exposure through* R*, dual independence exposure through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The rows E, M, B, Q, R, U are the complete finite duality inventory: ground, source matroid, source bases, complement bases, rank/nullity data, and dual independence.

Read the source matroid through chapter 202. The hereditary and exchange rows are available only as displayed $\mathsf{Matroid}^{\uparrow}$ certificate rows, while the ground and finite subset reads remain inside $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ dependency surfaces.

The complement-basis row Q is then checked against E and B. Each dual basis is the finite complement of a displayed source basis, and the rank/nullity row R records the finite handoff by which source rank and complement size determine the dual rank read.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no coordinate stores a host dual matroid, quotient ground set, selected global basis family, external rank oracle, or ambient representation, none of those objects can discharge a local $\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}$ obligation. □

Theorem 7097.3 (Matroid dual rank-nullity handoff)**.** *Every rank-facing consumer of an accepted* $\mathsf{MatroidDual}^{\uparrow}$ *packet must read the finite route*

$$E, M, B, Q, R \longmapsto U, H, C, P, N$$

before exporting a dual rank or independence claim. The source rank row is read from the displayed $\mathsf{Matroid}^{\uparrow}$ *basis data, the nullity row is read from finite complements inside* E*, and the dual independence row is admitted only after the complement-basis and rank/nullity rows have been displayed.*

Proof. Start with the source $\mathsf{Matroid}^{\uparrow}$ row. The basis row B is meaningful only after the finite ground E and the source independent-set certificate M have been displayed.

Next open the complement-basis row. The row Q compares each displayed basis with its finite complement in E, so the dual basis data is a BHist complement ledger rather than a selected basis outside the packet.

The rank/nullity row R is downstream of this complement read. It records the finite handoff from source rank to dual rank by counting displayed basis and complement rows, not by invoking an ambient rank theorem.

The dual independence row U and the structural rows H, C, P, N preserve exactly this route. Thus a rank-facing consumer cannot export a dual rank or independence claim before reading the source basis, complement basis, rank/nullity, transport, replay, provenance, and local NameCert rows. □

Falsifiable prediction: A consumer that decodes a $\mathsf{MatroidDual}^{\uparrow}$ packet without displaying the source basis row, complement-basis row, rank/nullity ledger, hsame, Cont, Pkg, and local NameCert rows is not reading the carrier.

Independence witness: matroid, lattice, group: $\mathsf{Matroid}^{\uparrow}$ supplies the source independence rows, $\mathsf{Lattice}^{\uparrow}$ supplies finite meet/join dependency discipline, and $\mathsf{Group}^{\uparrow}$ supplies algebraic transport neighbours, but none carries the complement-basis plus rank/nullity dual packet as a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}$.

Constructive story: A $\mathsf{MatroidDual}^{\uparrow}$ packet is generated from a finite ground row, source MatroidUp row, finite basis rows, complement-basis rows, rank/nullity ledgers, dual independence rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7097.1, Definition 7097.2, and Definition 7097.3 seed the finite complement-basis and rank-nullity packet for matroid duality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MatroidDual}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host matroid duality, quotient ground equality, selected global bases, external rank oracles, ambient linear representation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complement-basis exactness, rank/nullity handoff, dual independence transport, replay, provenance, and local naming for BEDC.Derived.MatroidDualUp.

Chapter 7098

A Concrete Naming Certificate for $\mathsf{CertifiedEventFlow}^{\uparrow}$

The $\mathsf{CertifiedEventFlow}^{\uparrow}$ packet realizes the channel-termination boundary of chapter 9369 as finite source/channel endpoint data. It sits beside the local channel boundary of chapter 3996, the budget certificate of chapter 5967, the GroundCompiler event-flow audit of chapter 829, and the hendeca event-flow certificate of chapter 6332. The packet names source events, legal channel bodies, terminators, endpoints, transport, replay, provenance, and local naming; it does not assert global halting, compiler correctness, or a host stream parser.

Definition 7098.1 (CertifiedEventFlow carrier)**.** *A* $\mathsf{CertifiedEventFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, B, T, E, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-event row naming the finite event before channel encoding.*
2. *B is the legal channel body row, recording the finite encoded body and its body-safety classifier.*
3. *T is the terminator row, displaying the local channel endpoint token.*
4. *E is the endpoint row, recording the first legal endpoint read and the decoded source event.*
5. *L is the local termination row, separating channel-level endpoint detection from certificate-level halting.*
6. *A is the audit handoff row, exposing the event-flow packet to GroundCompiler audit and hendeca event-flow consumers only through displayed rows.*
7. *H records componentwise* hsame *transport for source event, channel body, terminator, endpoint, local termination, audit handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for endpoint-facing and audit-facing consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{CertifiedEventFlow}^{\uparrow}$, $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$, $\mathsf{ChannelTerminationBud}$ $\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}$, $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedEventFlow}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden source text, an unrestricted evaluator, a global halting oracle, compiler correctness, carried metadata, or a host channel parser.

Theorem 7098.2 (CertifiedEventFlow NameCert obligations)**.** *For every accepted* $\mathsf{CertifiedEventFlow}^{\uparrow}$ *carrier*

$$F = (S, B, T, E, L, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertifiedEventFlow}^{\uparrow}}$ *surface consists exactly of carrier admission for* F*, source-event exposure through* S*, legal channel-body exposure through* B*, terminator exposure through* T*, endpoint exposure through* E*, local termination separation through* L*, audit handoff through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The data rows S, B, T, E, L, A are the complete source/channel endpoint inventory, and the structural rows H, C, P, N only transport, replay, source, and name those rows.

Read the source and channel side first. The source event S is separated from the legal channel body B, the terminator T, and the endpoint read E before a local termination row can be exported.

Next apply the channel-termination boundary and budget siblings. They keep the endpoint row local to the channel and prevent it from becoming a certificate-level halting classifier.

Finally read the audit handoff through the GroundCompiler event-flow audit and hendeca event-flow neighbours. The handoff row A exposes the packet as finite event-flow data, while no coordinate supplies hidden source text, unrestricted evaluation, global halting, compiler correctness, carried metadata, or a host parser. □

Theorem 7098.3 (CertifiedEventFlow channel-termination handoff)**.** *Every endpoint-facing consumer of an accepted* $\mathsf{CertifiedEventFlow}^{\uparrow}$ *carrier must read*

$$S, B, T, E \longmapsto L, A, H, C, P, N$$

before it treats the event as locally terminated. The row E *is certified only as the endpoint of the legal channel body* B *at terminator* T*, and the row* L *records that this local endpoint is not a global halting classifier for arbitrary certificate continuations.*

Proof. Begin with the source/channel separation. The source event S and legal body B are distinct rows, so a consumer cannot identify channel bytes with source identity before the endpoint has been decoded.

Open the terminator and endpoint rows. The terminator T marks the local channel endpoint, and E records the first legal endpoint read together with the decoded source event.

Now attach the local termination row L. It reads the endpoint as local channel termination while preserving the refusal boundary against certificate-level halting.

The audit, transport, replay, provenance, and naming rows A, H, C, P, N carry that same finite route to downstream consumers. Since the packet contains no global halting oracle or host parser, the handoff exports source/channel endpoint separation and nothing stronger. □

Falsifiable prediction: A consumer that decodes a $\mathsf{CertifiedEventFlow}^{\uparrow}$ endpoint without displaying the source event, legal channel body, terminator, and endpoint rows is not reading the carrier.

Independence witness: channel-termination-halting-boundary, channel-termination-budget-certificate, groundcompilereventflowaudit, b-hist-hendeca-event-flow-name-cert: none has the same source-event/channel-event/endpoint triad plus local NameCert row.

The later Lean `TasteGate.lean` development for $\mathsf{CertifiedEventFlow}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{CertifiedEventFlow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CertifiedEventFlow}^{\uparrow}$ instances preserving source event,

legal channel body, terminator, endpoint, local termination, audit handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CertifiedEventFlow}^{\uparrow}}$.

Constructive story: A $\mathsf{CertifiedEventFlow}^{\uparrow}$ packet is generated from source-event rows, legal channel-body rows, terminator rows, endpoint rows, local termination rows, audit handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertifiedEventFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7098.1, Definition 7098.2, and Definition 7098.3 seed the finite source/channel endpoint packet for certified event flows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertifiedEventFlow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global halting, compiler correctness, hidden source recovery, host stream parsing, carried metadata evidence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, local termination separation, audit handoff, transport, replay, provenance, and local naming for BEDC.Derived.CertifiedEventFlowUp.

Chapter 7099

A Concrete Naming Certificate for $\mathsf{FareyMediantInterval}^{\uparrow}$

Following Hardy and Wright's treatment of Farey sequences and mediants, the $\mathsf{FareyMediantInterval}^{\uparrow}$ packet records the finite interval-refinement surface where two determinant-neighbor rational endpoints generate a displayed mediant and a dyadic subinterval before any Real or Completion consumer reads the result. It sits over the rational carrier of chapter 37, the dyadic arithmetic core of chapter 291, the terminal real-name surface of chapter 38, the Stern–Brocot branch row of chapter 1704, and the Farey adjacency ledger of chapter 1759. The packet is finite interval data for rational refinement; it is not a quotient-real equality theorem, a selected representative, a trichotomy principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7099.1 (Farey mediant interval carrier)**.** *A* $\mathsf{FareyMediantInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, M, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the left rational endpoint row, read through* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *numerator, denominator, sign, and scale data.*
2. *R is the right rational endpoint row, read through the same rational and dyadic arithmetic surface.*
3. *M is the mediant row. It records the rational*

$$\frac{p_L + p_R}{q_L + q_R}$$

   formed from the displayed endpoint numerators and denominators, together with the determinant-neighbor and Farey adjacency checks for L and R.
4. *D is the dyadic-refinement row. It records the chosen child interval, the finite dyadic tolerance budget, and the gap comparison used by downstream Real and Completion consumers.*
5. *H records componentwise* hsame *transport for endpoint, determinant-neighbor, Farey, mediant, dyadic-refinement, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the finite route*

$$L, R \longmapsto M \longmapsto D \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{FareyMediantInterval}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FareyMediantInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through L, R, M, D, H, C, P, N. It has no coordinate for quotient real equality, selected endpoint representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7099.2 (Farey mediant interval NameCert obligations)**.** *For every accepted* $\mathsf{FareyMediantInterval}^{\uparrow}$ *carrier*

$$I = (L, R, M, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FareyMediantInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two rational endpoint rows, determinant-neighbor evidence, Farey adjacency row, mediant row, dyadic-refinement row, transport, replay, provenance, and local naming;*
2. *endpoint exactness, requiring L and R to be displayed* $\mathsf{Rat}^{\uparrow}$ *rows with dyadic arithmetic support before a mediant is formed;*
3. *determinant-neighbor exactness, requiring the endpoint rows to expose the finite determinant-one check used by the Farey and Stern–Brocot sibling routes;*
4. *mediant exactness, requiring M to use only the displayed endpoint numerators and denominators;*
5. *dyadic-refinement exactness, requiring D to choose a displayed child interval and a finite dyadic tolerance budget downstream of M;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rows above.*

No obligation is stored in a quotient real equality proof, selected representative, host real order, countable-choice functional, or ambient completeness theorem.

Proof. Project the accepted carrier. The endpoint data are exactly L and R, and the arithmetic information available to the packet is the rational and dyadic data displayed in those rows.

The determinant-neighbor and Farey checks are finite rows attached to the endpoint pair. They identify the pair as an adjacent rational bracket for the sibling Stern–Brocot and Farey routes without adding a host order, trichotomy branch, or selected Real representative.

The mediant row is then forced by the endpoint numerators and denominators. Since M contains no other arithmetic coordinate, the inserted rational is read only from the displayed endpoint rows.

Finally D records the child interval and dyadic budget, while H, C, P, N preserve the same packet by componentwise transport, replay, provenance, and naming. These rows exhaust the local certificate surface. □

Theorem 7099.3 (Farey mediant dyadic refinement)**.** *Let*

$$I = (L, R, M, D, H, C, P, N)$$

be an accepted $\mathsf{FareyMediantInterval}^{\uparrow}$ *carrier. Every dyadic refinement read factors through the finite chain*

$$L, R \longmapsto M \longmapsto D.$$

Thus the child interval used by a Real or Completion consumer is selected only after the determinant-neighbor endpoint rows, the Farey mediant row, and the dyadic tolerance budget have been displayed.

Proof. Begin with the endpoint rows L and R. Their determinant-neighbor evidence gives the finite rational bracket that may be refined.

Read the mediant row M. The inserted rational is computed from the displayed endpoint numerators and denominators, so the refinement has no independent representative or host-order coordinate.

The dyadic row D then compares the resulting child interval with the finite tolerance budget. A downstream Real or Completion request can read only this displayed child interval, not a quotient real equality, a selected endpoint representative, or an ambient completeness witness. □

Theorem 7099.4 (Farey mediant interval Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{FareyMediantInterval}^{\uparrow}$ *carrier reads the displayed route*

$$\mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{SternBrocot}^{\uparrow},\ \mathsf{FareySequence}^{\uparrow} \longmapsto \mathsf{FareyMediantInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes rational endpoint rows, determinant-neighbor and Farey rows, a mediant row, a dyadic-refinement row, componentwise transport, replay, provenance, and local naming before any Real-facing handoff is visible.

Proof. The rational and dyadic sources supply the two endpoint rows. The Stern–Brocot and Farey sibling rows then witness that the endpoints form the finite determinant-neighbor bracket consumed by this packet.

Apply Definition 7099.3. The mediant and dyadic rows are displayed before the child interval can be handed to a Real-facing consumer.

The structural rows H, C, P, N keep the handoff inside the same finite packet. Therefore the route exports finite rational interval refinement only, with no quotient real equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FareyMediantInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{FareyMediantInterval}^{\uparrow}$ packet is generated from left and right rational endpoint rows, determinant-neighbor and Farey adjacency evidence, a mediant row, a dyadic-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FareyMediantInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7099.1, Definition 7099.2, Definition 7099.3, and Definition 7099.4 seed the finite rational interval-refinement packet over Rat, DyadicRatCore, SternBrocot, FareySequence, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FareyMediantInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Real equality, selected rational or Real representatives, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, determinant-neighbor exactness, Farey route compatibility, mediant exactness, dyadic-refinement exactness, Real handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FareyMediantIntervalUp.

Chapter 7100

A Concrete Naming Certificate for $\mathsf{PorousSet}^{\uparrow}$

Following Zajicek and standard metric porosity theory, the $\mathsf{PorousSet}^{\uparrow}$ packet records a displayed hole schedule around a located subset of a metric space as finite BEDC NameCert data. It sits over the parent metric surface of chapter 57, the located-subset distance interface of chapter 1541, the terminal real-name seal of chapter 38, and the finite epsilon-net witness discipline of chapter 1767. The packet names the finite boundary where a consumer can read a center, a scale, and a displayed empty ball before using porous-set data; it does not assert measure-nullness, Baire meagreness, open-cover compactness, choice-selected holes, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7100.1 (Porous set carrier)**.** *A* $\mathsf{PorousSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, L, R, E, H, W, D, S, T, C, Q, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row, carrying the point classifier, distance classifier, and metric-distance ledger used by every hole request.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the admitted subset and its distance-to-set readback surface.*
3. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for distances, radii, and scale comparisons visible to the porous-set consumer.*
4. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *comparison row used only to expose finite center and tolerance windows around the inspected scale.*
5. *H is the hole-schedule row. Each visible entry records a base point, an outer scale, a candidate hole center, an inner radius, and the finite witness that the named inner ball is disjoint from the located subset at the inspected tolerance.*
6. *W is the finite window row for the base point, center point, outer-radius request, inner-radius request, and distance comparisons currently read by the consumer.*
7. *D records the dyadic or rational comparison cells that witness the positive scale relation between the outer request and the inner hole.*
8. *S records componentwise* hsame *transport for the metric source, located-set row, Real seal, finite-net comparison row, hole schedule, finite windows, comparison cells, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a porous-set query through* $M, L, R, E, H, W, D, S, C, Q, N$.
10. *C is the* Pkg *provenance over* $\mathsf{PorousSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q is the local refusal ledger for measure, category, open-cover, selected-hole, and ambient-completion claims.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PorousSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, R, E, H, W, D, S, T, C, Q, N$. It has no coordinate for a measure-null theorem, Baire-category theorem, open-cover compactness proof, global hole selector, quotient subset equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PorousSetUp.TasteGate.PorousSetTasteGate single carrier alignment`

Theorem 7100.2 (Porous set NameCert obligations)**.** *For every accepted* $\mathsf{PorousSet}^{\uparrow}$ *carrier* $P = (M, L, R, E, H, W, D, S, T, C, Q, N)$, *the local* $\mathsf{NameCert}_{\mathsf{PorousSet}^{\uparrow}}$ *surface consists exactly of ambient metric admission, located-set admission, Real distance seal handoff, finite epsilon-net comparison exposure, hole-schedule exposure, finite-window inspection, positive-scale comparison, componentwise transport, displayed replay, provenance, refusal-ledger recording, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 7100.1. The analytic source rows are the ambient metric row M, the located-set row L, the Real seal R, and the finite epsilon-net comparison row E.

The hole data are finite. The schedule row H names only the currently inspected base point, scale, hole center, inner radius, and disjointness witness, while W restricts each read to the displayed finite window.

The comparison and structural rows then fix the same route. The row D records the positive relation between outer scale and inner hole, and S, T, C, N transport, replay, source, and name the packet.

Finally Q records the refused exits. Since no carrier coordinate stores measure-nullness, Baire meagreness, open-cover compactness, a global selector for holes, quotient subset equality, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7100.3 (Porous set metric hole route)**.** *Every metric or located-set consumer of an accepted* $\mathsf{PorousSet}^{\uparrow}$ *packet reads porous-set hole data through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{LocatedSet}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto H, W, D.$$

Thus the consumer must first read the ambient metric row, located-set distance surface, Real distance seal, and finite epsilon-net comparison row before it can inspect a displayed hole schedule, finite window, and positive-scale comparison.

Proof. Start from a porous-set query. The ambient metric row M supplies the point and distance observations, and the located-set row L supplies the admitted subset boundary through the distance interface of chapter 1541.

The query next reaches the Real and finite-net rows. The seal R carries the distance and radius readbacks, while E exposes only the finite center and tolerance windows relevant to the inspected scale.

Only after those rows are displayed can the hole schedule be read. The rows H, W, D name the base point, candidate center, inner radius, finite disjointness witness, finite observation window, and positive-scale comparison.

The structural rows S, T, C, Q, N preserve the same route and record the refused exits. Therefore the exported hole datum is a finite metric route, not a measure theorem, category theorem, open-cover compactness proof, choice-selected family of holes, or ambient completeness principle. □

Theorem 7100.4 (Porous set finite-hole dependency route)**.** *Every displayed hole read from an accepted* $\mathsf{PorousSet}^{\uparrow}$ *carrier factors through the finite dependency route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{LocatedSet}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto H, W, D.$$

The $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row is the sibling dependency that makes the packet non-orphan: it exposes the finite center and tolerance windows before the hole schedule* H*, finite window row* W*, and positive-scale comparison row* D *can be read.*

Proof. Project the carrier of Definition 7100.1. The rows M, L, R, E, H, W, D are the only analytic rows visible before transport, replay, provenance, refusal, and local naming.

Apply Definition 7100.3. It already forces the consumer to read the ambient metric row, located-set row, Real seal, and finite epsilon-net row before any displayed hole datum is exposed.

The refusal ledger Q closes the remaining exits. It records that the packet does not export measure-nullness, Baire meagreness, open-cover compactness, choice-selected holes, quotient subset equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed hole remains finite dependency data over the named sibling route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PorousSet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PorousSet}^{\uparrow}$ packet is generated from a Metric source row, a LocatedSet distance row, a Real seal, a FiniteEpsilonNet comparison row, a displayed hole schedule, finite windows, positive-scale comparison cells, componentwise hsame transport, displayed Cont replay, Pkg provenance, a refusal ledger, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PorousSet}^{\uparrow}}$, seed)

Scope closed: Definition 7100.1, Definition 7100.2, Definition 7100.3, and Definition 7100.4 seed the finite porous-set packet over displayed metric, located-set, Real, finite epsilon-net, hole-schedule, window, comparison, transport, replay, provenance, refusal-ledger, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PorousSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure-nullness, Baire meagreness, open-cover compactness, choice-selected holes, quotient subset equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-set distance compatibility, Real seal handoff, finite epsilon-net exposure, hole-schedule exposure, finite-window inspection, positive-scale comparison, transport, replay, provenance, refusal-ledger, and local naming for BEDC.Derived.PorousSetUp.

Chapter 7101

A Concrete Naming Certificate for $\mathsf{BoundedRealLimsup}^{\uparrow}$

Following Bishop and Bridges on bounded real sequences and constructive upper envelopes, the $\mathsf{BoundedRealLimsup}^{\uparrow}$ packet records the finite surface by which a bounded real sequence exposes limsup data before any ambient supremum theorem is available. It sits over the bounded-sequence source of chapter 1415, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the general limsup boundary of chapter 768, and the dual liminf comparison of chapter 1553. The packet names the upper-envelope read for a displayed bounded sequence; it does not assert arbitrary set suprema, host order completeness, selected subsequences, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7101.1 (Bounded real limsup carrier)**.** *A* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, W, Q, U, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row, carrying the bounded real sequence and its visible bound ledger.*
2. *W is the finite upper-envelope window row, listing the inspected tail windows and rational upper estimates read from the bounded source.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tail estimates and tolerance requests visible inside W.*
4. *U is the* $\mathsf{Limsup}^{\uparrow}$ *upper-envelope row, storing the limsup request as a bounded tail-envelope handoff rather than as an arbitrary supremum.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, finite windows, rational readback, and upper-envelope handoff have been displayed.*
6. *H records componentwise* hsame *transport for the bounded source, windows, rational readback, upper envelope, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the same bounded-source, window, readback, upper-envelope, and seal route.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealLimsup}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{Liminf}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRealLimsup}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, W, Q, U, E, H, C, P, N$. It has no coordinate for arbitrary subset suprema, Dedekind completeness, selected maximizing subsequences, open-cover compactness, quotient real equality, Lean verification, or bridge checking.

Theorem 7101.2 (Bounded real limsup NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *carrier*

$$L = (B, W, Q, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedRealLimsup}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded source, finite upper-envelope windows, regular rational readback, limsup handoff, real seal, componentwise transport, displayed replay, provenance, and local naming row; window coverage requiring every limsup read to inspect W before Q; upper-envelope exactness requiring U to consume only bounded tail data from B, W, Q; and nonescape from the named rows.*

Proof. Project the accepted carrier of Definition 7101.1. The rows B, W, Q, U, E are the only analytic rows available to a limsup consumer.

The bounded source is read first through B. Each tail-envelope request then enters the finite window row W, and the rational estimates visible in that window are read back through Q.

The upper-envelope handoff U is therefore sequence-local. It can compare displayed tail estimates and pass a limsup request onward, but it has no coordinate for an arbitrary subset supremum or a host completeness principle.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded endpoints are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7101.3 (Bounded real limsup Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *carrier*

$$L = (B, W, Q, U, E, H, C, P, N)$$

must factor its read through the finite upper-envelope route

$$B \longmapsto W \longmapsto Q \longmapsto U \longmapsto E \longmapsto H, C, P, N.$$

The consumer may export the bounded sequence source, inspected tail windows, regular rational readback, limsup handoff, real seal, transport, replay, provenance, and local naming. It may not export arbitrary real suprema, selected subsequences, open-cover compactness, quotient equality of real names, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Begin with a Real-facing consumer of the accepted carrier. The bounded real sequence is exposed through B, and every inspected tail request is restricted to the finite windows carried by W.

The rational observations are then read through Q. The $\mathsf{Limsup}^{\uparrow}$ row U can consume only those bounded tail-envelope observations, so the handoff is an upper-envelope route over a displayed bounded sequence rather than a general completeness theorem.

The terminal row E seals the resulting real-name request after B, W, Q, U have been exposed. The structural rows H, C, P, N preserve that route without adding a selected subsequence, host supremum, quotient equality, or ambient real completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{BoundedRealLimsup}^{\uparrow}$ Real-facing read factors through a bounded real sequence row, finite upper-envelope windows, RegSeqRat readback, a Limsup handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: boundedrealsequence, regseqrat, real, limsup, liminf: $\mathsf{BoundedRealLimsup}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded-source data, tail-window data, rational readback, upper-envelope handoff, Real sealing, transport, replay, provenance, and local naming while refusing arbitrary supremum, selected subsequences, quotient real equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealLimsup}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealLimsup}^{\uparrow}$ packet is generated from a bounded real sequence row, finite upper-envelope windows, RegSeqRat readback, a Limsup handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealLimsup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7101.1, Definition 7101.2, and Definition 7101.3 seed the bounded sequence upper-envelope surface for Real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealLimsup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary subset suprema, Dedekind completeness, selected maximizing subsequences, open-cover compactness, quotient real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite upper-envelope window coverage, RegSeqRat readback, Limsup handoff exactness, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRealLimsup

Chapter 7102

A Concrete Naming Certificate for $\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}$

Following Heine–Cantor and Bishop compact metric practice, the $\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}$ packet records the finite route by which a uniform modulus over a compact metric source restricts to a displayed compact subspace. It concretizes the compact-uniform-continuity route of chapter 9237 and sits over the compact metric source of chapter 129, the continuous-map graph row of chapter 130, the uniform-modulus handoff of chapter 369, and the subspace admission boundary of chapter 1519. The packet exposes inherited modulus data only after compact-source, subspace, graph, and tolerance rows have been displayed; it does not assert host compactness, arbitrary subset continuity, selected finite covers, quotient subspace equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7102.1 (Compact subspace modulus-restriction carrier)**.** *A* $\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, S, F, M, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the compact metric finite-net and located-distance data available before restriction.*
2. *S is the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row, naming the displayed subspace predicate, inherited metric row, and inclusion route consumed by the packet.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, exposing the pointwise continuity data of the map before the restricted read reaches a modulus.*
4. *M is the subspace metric-comparison row, recording that source distances used by the restricted consumer are inherited from K through S.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row whose tolerance-to-precision ledger is read only after K, S, F, M have been displayed.*
6. *H records componentwise* hsame *transport for compact source, subspace admission, continuous graph, metric comparison, uniform modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the same compact-subspace, graph, metric, and modulus route.*

8. *P is the* Pkg *provenance over* CompactSubspaceModulusRestriction$^{\uparrow}$, CompactMetric$^{\uparrow}$, MetricSubspace$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{CompactSubspaceModulusRestriction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, F, M, U, H, C, P, N$. It has no coordinate for an external compactness theorem, unrestricted subspace equality, selected cover refinement, host function extensionality, arbitrary Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompactSubspaceModulusRestrictionUp.CompactSubspaceModulusRestrictionTasteGate`
`single carrier alignment`

Theorem 7102.2 (Compact subspace modulus-restriction NameCert obligations)**.** *For every accepted* CompactSubspaceModulusRestriction$^{\uparrow}$ *carrier*

$$R = (K, S, F, M, U, H, C, P, N),$$

the local NameCert$_{\text{CompactSubspaceModulusRestriction}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, metric subspace admission, continuous-map graph row, inherited metric-comparison row, uniform-modulus handoff, componentwise transport, displayed replay, provenance, and local naming row; restriction exactness requiring every subspace read to pass through S before M; modulus exactness requiring U to consume only the displayed graph and inherited metric rows; and nonescape from the named rows.*

Proof. Project the accepted carrier of Definition 7102.1. The compact source, subspace admission, graph, metric comparison, and modulus rows are the only analytic rows present.

The restricted consumer first reads the subspace admission row S against the compact source row K. Thus each source-distance comparison visible in the subspace route is inherited through M rather than supplied by a separate host metric.

The continuous-map row F then feeds the uniform-modulus row U. Since U is downstream from K, S, F, M, the packet restricts an already displayed modulus surface and does not choose a new compact cover or use arbitrary subspace equality.

Finally H, C, P, N transport, replay, source, and name the same finite route. The excluded compactness, equality, choice, and completeness endpoints have no carrier coordinate, so the listed rows exhaust the local NameCert obligations. □

Theorem 7102.3 (Compact subspace modulus-restriction uniform handoff)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* CompactSubspaceModulusRestriction$^{\uparrow}$ *carrier*

$$R = (K, S, F, M, U, H, C, P, N)$$

must factor its restricted read through the finite route

$$K \longmapsto S \longmapsto F, M \longmapsto U \longmapsto H, C, P, N.$$

The consumer may export the compact metric source, displayed subspace admission, continuous-map graph row, inherited metric comparison, uniform-modulus ledger, transport, replay, provenance, and local naming. It may not export host compactness, unrestricted subspace equality, selected finite covers, host function extensionality, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Start with a uniform-modulus consumer. The compact metric source K and subspace admission row S determine which source points and inherited distances may be inspected.

The graph and metric rows are then read together. The continuous-map row F supplies the displayed pointwise graph data, while M records that each restricted distance comparison is inherited from the parent compact metric source through S.

Only after those rows are visible may the consumer read U. The resulting modulus is therefore a restriction of the displayed compact-source modulus route, not a new external selector or cover-refinement theorem.

The structural rows H, C, P, N preserve the same route. Since no other coordinates are present, the packet exports exactly the compact-subspace uniform handoff named in the statement. □

Falsifiable prediction: Every accepted $\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}$ uniform-modulus read factors through a CompactMetric source row, MetricSubspace admission row, ContinuousMap graph row, inherited metric-comparison row, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric-subspace: CompactSubspaceM is not bijective to any listed sibling because it jointly carries compact-source data, subspace admission, graph data, inherited metric comparisons, a uniform-modulus handoff, transport, replay, provenance, and local naming while refusing host compactness, unrestricted subspace equality, selected cover refinement, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}$ packet is generated from a CompactMetric source row, MetricSubspace admission row, ContinuousMap graph row, inherited metric-comparison row, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7102.1, Definition 7102.2, and Definition 7102.3 seed the compact-subspace uniform-modulus restriction surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactSubspaceModulusRestriction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, unrestricted subspace equality, selected finite covers, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subspace admission exactness, inherited metric-comparison exactness, ContinuousMap graph exposure, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactSubspaceModulusRestrictionUp.

Chapter 7103

A Concrete Naming Certificate for $\mathsf{CauchyProductSequence}^{\uparrow}$

Following Bishop and Bridges on constructive real multiplication and the Mertens product discipline for Cauchy data, the $\mathsf{CauchyProductSequence}^{\uparrow}$ packet records the finite sequence-level product route before any completed-real multiplication theorem is available. It sits over the product source row of chapter 464, the finite stream windows of chapter 126, the dyadic product ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The sibling commutativity packet of chapter 2238 reads the same product surface after the two source windows are swapped; this chapter names the underlying product-sequence carrier itself.

Definition 7103.1 (Cauchy product sequence carrier)**.** *A* $\mathsf{CauchyProductSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Y, W_X, W_Y, D, M, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *X and Y are the two accepted source rows whose product sequence is being read.*
2. *W_X and W_Y are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows exposing the source terms used by the product request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-observation ledger, containing the displayed dyadic products and tolerance comparisons for the retained windows.*
4. *M is the finite Cauchy-product budget row licensed by* $\mathsf{CauchyProduct}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W_X, W_Y, D, M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular readback is available.*
7. *H records componentwise* hsame *transport for source rows, windows, dyadic product observations, product budget, readback, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from product-sequence requests through $X, Y, W_X, W_Y, D, M, R, E, H, Q, N$.*
9. *Q is the* Pkg *provenance over* $\mathsf{CauchyProductSequence}^{\uparrow}$*,* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W_X, W_Y, D, M, R, E, H, C, Q, N$. It has no coordinate for host multiplication, quotient stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7103.2 (Cauchy product sequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductSequence}^{\uparrow}$ *carrier*

$$P = (X, Y, W_X, W_Y, D, M, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductSequence}^{\uparrow}}$ *surface consists exactly of source admission for X, Y, paired* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-observation exposure,* $\mathsf{CauchyProduct}^{\uparrow}$ *budget admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host multiplication, quotient streams, selected limits, trichotomy, countable choice, and ambient completeness.*

Proof. Project the carrier in Definition 7103.1. The only analytic rows are the two source rows, their finite stream windows, the dyadic product ledger, the product budget, the regular readback, and the Real-facing seal.

The product obligation is read through the displayed budget row M. It consumes the paired windows W_X, W_Y and the dyadic ledger D, so no product consumer can obtain a sequence product directly from host multiplication or from a quotient stream.

The rows R and E are downstream of that same finite route. Componentwise transport, replay, provenance, and local naming preserve exactly the displayed packet, so the listed obligations exhaust the local certificate surface. □

Theorem 7103.3 (Cauchy product sequence regular readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProductSequence}^{\uparrow}$ *packet must first read*

$$\mathsf{CauchyProduct}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $M, D, W_X, W_Y, R, E, H, C, Q, N$. The readback may transport and replay the finite product-sequence route, but it cannot export host multiplication, quotient stream equality, a selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7103.2. They force the product request to expose the $\mathsf{CauchyProduct}^{\uparrow}$ budget row M, the dyadic ledger D, and the paired stream windows W_X, W_Y before the regular readback row R is consumed.

The Real-facing seal is later in the same carrier. Since E is downstream of R, any consumer reaching E has already passed through the product budget, dyadic observations, stream windows, and regular rational readback.

Finally inspect H, C, Q, N. These structural rows transport, replay, source, and name the same finite packet. None contains host multiplication, quotient stream equality, selected limits, trichotomy, countable choice, or ambient completeness, so the readback cannot export those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductSequence}^{\uparrow}$ packet is generated from two source rows, paired StreamName windows, a DyadicRatCore product ledger, a CauchyProduct budget row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductSequence}^{\uparrow}}$, seed)

Scope closed: Definition 7103.1, Definition 7103.2, and Definition 7103.3 seed the finite product-sequence route for later Real multiplication and regular Cauchy arithmetic consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyProductSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host multiplication, quotient stream equality, selected limits, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, paired window exposure, dyadic product ledger exposure, CauchyProduct budget admission, regular readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductSequenceUp.

Chapter 7104

A Concrete Naming Certificate for $\mathsf{CauchyRealMultiplication}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRealMultiplication}^{\uparrow}$ packet records multiplication of Cauchy real names as a finite BEDC naming surface before any $\mathsf{Real}^{\uparrow}$ consumer is allowed to read a product. It sits over the product route of chapter 464, the regular rational sequence surface of chapter 301, the finite window schedule of chapter 126, the dyadic tolerance arithmetic of chapter 291, the terminal seal of chapter 38, and the addition-facing neighbor of chapter 7091. The packet gives Real-facing algebra consumers a finite route from two Cauchy names to one sealed product name; it does not assert quotient equality of real names, selected host limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, host multiplication, or a completed field operation.

Definition 7104.1 (Cauchy real multiplication carrier)**.** *A* $\mathsf{CauchyRealMultiplication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathsf{Real},\times} = (R_0, R_1, W_0, W_1, D_0, D_1, D_\times, C_\times, S, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *factor-name rows whose product is being read.*
2. W_0 *and* W_1 *are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules used to inspect the two factors.*
3. D_0 *and* D_1 *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *factor-tolerance ledgers for the two source windows.*
4. $D_\times$ *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-tolerance ledger, recording the dyadic product observation and the output tolerance budget.*
5. $C_\times$ *is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product row that consumes* $R_0, R_1, W_0, W_1, D_0, D_1, D_\times$ *before any regular product readback is exposed.*
6. S *is the product* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplied by the displayed* $\mathsf{CauchyProduct}^{\uparrow}$ *route.*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *multiplication seal row reached only after the factor rows, window rows, dyadic ledgers, product row, and product readback have been displayed.*
8. H *records componentwise* hsame *transport for the two factor rows, two window rows, dyadic ledgers, product row, readback row, seal row, replay row, provenance row, and local naming row.*
9. C *records displayed* Cont *replay from every Real-facing multiplication request through* $R_0, R_1, W_0, W_1, D_0, D_1, D_\times$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRealMultiplication}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealAddition}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealMultiplication}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for quotient equality of Cauchy names, host real multiplication, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completed-real algebra operation.*

Theorem 7104.2 (Cauchy real multiplication NameCert obligations)**.** *For every accepted* CauchyRealMultiplication *carrier*

$$M_{\mathsf{Real},\times} = (R_0, R_1, W_0, W_1, D_0, D_1, D_\times, C_\times, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealMultiplication}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the two factor rows, paired windows, dyadic ledgers,* $\mathsf{CauchyProduct}^{\uparrow}$ *product row, product readback,* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows;*
2. *factor classifier stability, requiring the two* $\mathsf{RegSeqRat}^{\uparrow}$ *factor rows and their* $\mathsf{StreamName}^{\uparrow}$ *windows to transport with the same displayed* hsame *data;*
3. $\mathsf{CauchyProduct}^{\uparrow}$ *regularity, requiring the product row to supply the product* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any Real-facing seal is visible;*
4. *Real-seal nonescape, requiring the multiplication seal E to be downstream of $C_\times$ and S;*
5. *ledger exactness, requiring $D_0, D_1, D_\times$ to account for every source tolerance, product observation, and output tolerance used by the read.*

Proof. Project the carrier of Definition 7104.1. Its analytic rows are the two factor rows, two window rows, dyadic ledgers, product row, readback row, and seal row; its structural rows are H, C, P, N.

The factor rows are read first. The $\mathsf{RegSeqRat}^{\uparrow}$ rows R_0, R_1 and $\mathsf{StreamName}^{\uparrow}$ rows W_0, W_1 fix the finite factor observations, while D_0, D_1 record the factor tolerance ledgers that travel with those observations.

The product route is downstream of that data. The row $C_\times$ is a $\mathsf{CauchyProduct}^{\uparrow}$ read through the displayed windows and dyadic product ledger $D_\times$, and it supplies the product $\mathsf{RegSeqRat}^{\uparrow}$ row S before the seal row E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no other coordinate appears in the carrier, the listed five obligations exhaust the local NameCert surface and exclude host multiplication, quotient equality, selected limits, trichotomy, and ambient completeness. □

Theorem 7104.3 (Cauchy real multiplication RegSeqRat handoff)**.** *Every Real-facing multiplication consumer of an accepted* $\mathsf{CauchyRealMultiplication}^{\uparrow}$ *carrier factors through the finite route*

$$(R_0, R_1) \longmapsto (W_0, W_1) \longmapsto (D_0, D_1, D_\times) \longmapsto C_\times \longmapsto S \longmapsto E.$$

Thus the consumer must read the $\mathsf{RegSeqRat}^{\uparrow}$ *factor rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and* $\mathsf{CauchyProduct}^{\uparrow}$ *product row before the product* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *multiplication seal are visible.*

Proof. Begin with carrier admission. The Real-facing multiplication request has no source rows other than $R_0, R_1, W_0, W_1, D_0, D_1, D_\times, C_\times, S, E$ and the structural rows H, C, P, N.

Apply the product route of Definition 464.33. It places the real-facing product read downstream of paired windows, dyadic product observations, a product ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row supplied by $\mathsf{CauchyProduct}^{\uparrow}$.

The present packet records those dependencies as $W_0, W_1, D_0, D_1, D_\times, C_\times, S$ and exposes the terminal seal as E. The structural rows preserve the same order for every multiplication consumer.

Therefore the multiplication handoff is a finite regular-rational route to a Real seal and nothing stronger. It exports no host multiplication primitive, quotient stream equality, selected product, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealMultiplication}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealMultiplication}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7104.1, Definition 7104.2, and Definition 7104.3 seed the finite Cauchy-real multiplication surface over two $\mathsf{RegSeqRat}^{\uparrow}$ names, paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a $\mathsf{CauchyProduct}^{\uparrow}$ product row, product readback, and a terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealMultiplication}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, host real multiplication, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, factor classifier stability, CauchyProduct regularity, Real-seal nonescape, ledger exactness, replay, provenance, and local naming for BEDC.Derived.CauchyRealMultiplicationUp.

Constructive story: A $\mathsf{CauchyRealMultiplication}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ factor rows, paired $\mathsf{StreamName}^{\uparrow}$ finite-window schedules, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a $\mathsf{CauchyProduct}^{\uparrow}$ product row, a product readback row, a $\mathsf{Real}^{\uparrow}$ multiplication seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7105

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real constructions and Mac Lane on functorial action, the $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ packet records the finite BEDC route by which a uniformly continuous map on regular rational Cauchy sources induces a completion-map row through bounded windows. It sits after the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion-map route of chapter 1937. The packet names only the finite functoriality surface visible to completion consumers; it does not form quotient streams, choose limit representatives, assert host function equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7105.1 (Regular Cauchy completion-functoriality carrier)**.** *A* $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, B, W, D, F, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, exposing only the regular rational Cauchy data accepted by the source packet.*
2. *B is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *row, exposing the regular rational Cauchy data produced after the source-map read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected source and target prefixes.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bounds used to compare the displayed windows.*
5. *F is the uniformly continuous source-map row, carrying the finite modulus and graph evidence that transports the source window to the target window.*
6. *M is the induced completion-map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *after A, B, W, D, F have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the endpoint readback produced by the regular Cauchy completion route.*
8. *H records componentwise* hsame *transport for the source row, target row, stream windows, tolerance ledger, source map, completion map, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a regular Cauchy map request through* $A, B, W, D, F, M, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, D, F, M, E, H, C, P, N$. *It has no coordinate for quotient stream equality, selected representatives, countable choice, host function equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Theorem 7105.2 (Regular Cauchy completion-functoriality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ *carrier* $R = (A, B, W, D, F, M, E, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, uniformly continuous source-map row, induced* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *completion-map row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7105.1. The accepted packet displays exactly the rows $A, B, W, D, F, M, E, H, C, P, N$.

The regular Cauchy source and target data are visible before any completion-map row is consumed. The stream-window row W restricts the read to finite prefixes, and D records the dyadic tolerance cells used to compare those prefixes.

The uniformly continuous source-map row F is then the only source-side map evidence. The induced completion-map row M is read through the existing $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ surface, and the endpoint is sealed by E as a finite $\mathsf{Real}^{\uparrow}$ readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no listed row stores quotient streams, selected representatives, countable choice, host function equality, or ambient Real completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 7105.3 (Regular Cauchy completion-map stability)**.** *Let* $R = (A, B, W, D, F, M, E, H, C, P, N)$ *and* $R' = (A', B', W', D', F', M', E', H', C', P', N')$ *be accepted* $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ *carriers whose source regular-rational rows, finite windows, dyadic tolerance ledgers, and uniformly continuous source-map rows compare componentwise by* hsame. *Then the induced completion-map row and terminal* $\mathsf{Real}^{\uparrow}$ *seal compare by the same transported route. A completion consumer must read*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ F \longmapsto \mathsf{CauchyCompletionFunctoriality}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it can consume the completed map.

Proof. Begin with the componentwise comparison of A, W, D, F against A', W', D', F'. The source regular-rational row, finite windows, tolerance ledger, and uniformly continuous map evidence are therefore the same visible route for both packets.

Project the completion-map rows M and M'. Each is generated only after the displayed regular source, stream-window, dyadic tolerance, and source-map data pass through the $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ route.

Transport the resulting endpoint through H and H'. The terminal rows E and E' are finite $\mathsf{Real}^{\uparrow}$ seals attached to the same completion-map route, so the componentwise comparison of the source route determines the map-and-seal comparison.

The replay and provenance rows C, P, N and C', P', N' repack the same transported route. They add no quotient stream equality, selected representative, countable-choice schedule, host function equality, or ambient Real-completeness principle. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ consumer is refuted if it reaches an induced completion-map row or terminal $\mathsf{Real}^{\uparrow}$ seal without displaying a $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and a uniformly continuous source-map row.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchy-completion-functoriality: no listed sibling is bijective to $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ because this packet jointly binds regular rational source and target rows, finite window inspection, dyadic tolerance accounting, source-map evidence, induced completion-map transport, Real sealing, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}$ packet is generated from RegSeqRat source and target rows, StreamName finite windows, a DyadicRatCore tolerance ledger, a uniformly continuous source-map row, an induced CauchyCompletionFunctoriality completion-map row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7105.1, Definition 7105.2, and Definition 7105.3 seed the finite regular Cauchy completion-map packet over displayed regular sources, finite windows, tolerance ledgers, source-map evidence, completion-map routing, Real sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctoriality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target regular-rational visibility, stream-window routing, dyadic tolerance accounting, source-map exactness, completion-map stability, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionFunctorialityUp.

Chapter 7106

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the compact-metric finite-net presentation of uniform convergence, the $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$ packet records the finite family certificate by which compact finite nets, shared modulus rows, uniform Cauchy family readbacks, and Real-seal handoff are made into NameCert data. It sits after the compact metric source of chapter 129, the finite epsilon-net row of chapter 1767, the shared family route of chapter 1555, the compact uniform Cauchy packet of chapter 6264, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The packet gives compact-uniform-continuity consumers a finite family object; it does not select a completed limit, assert open-cover compactness, identify quotient streams, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7106.1 (Compact uniform Cauchy family carrier)**.** *A* $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, N, F, \sigma, R, D, U, Q, E, H, C, P, L)$$

with the following displayed rows:

1. *X is the compact metric source row, read through finite compact-net data rather than through an open-cover choice principle.*
2. *N is the finite epsilon-net row whose cells cover the compact source for the inspected tolerance.*
3. *F is the displayed finite family row of regular Cauchy names over the compact source.*
4. *σ is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule used by every family member over every displayed net cell.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *family readback row for the scheduled component windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the compact-net radius, family tail radius, and Real-seal tolerance rows.*
7. *U is the accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *row using the same family, schedule, readback, and dyadic ledger.*
8. *Q is the compact-uniform Cauchy row that binds X, N, F, σ, R, D, U into one finite net-and-tail route.*

9. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after the compact finite-net row and shared family readback have been displayed.*
10. *H records componentwise* hsame *transport for compact source, net, family, schedule, readbacks, dyadic ledger, uniform-family row, compact-uniform row, seal, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from compact finite-net inspection through shared family readback, dyadic tolerance, uniform-Cauchy family routing, compact-uniform row, and Real-seal delivery.*
12. *P records* Pkg *provenance over* $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{UniformCauchyFamily}^{\uparrow}$, $\mathsf{CompactUniformCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyFamily}^{\uparrow}}$ *row.*

The carrier contains no hidden coordinate for an open-cover choice, quotient stream, selected limit, host real equality, component-specific modulus, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 7106.2 (Compact uniform Cauchy family NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$ *carrier*

$$K = (X, N, F, \sigma, R, D, U, Q, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact source, finite net, family row, shared schedule, regular readbacks, dyadic tolerance ledger, uniform-family row, compact-uniform row, Real seal, transport, replay, provenance, and local naming; compact-net exactness requiring N to remain attached to X; shared-modulus exactness requiring σ, D, U to be common to every displayed family member and net cell; Real-seal nonescape requiring E to be downstream of Q and U; and ledger exhaustion excluding open-cover choice, selected limits, quotient streams, component-specific moduli, countable choice, host real equality, and ambient completeness.*

Proof. Project the carrier of Definition 7106.1. The compact source, finite net, family row, shared schedule, regular readbacks, dyadic ledger, uniform-family row, compact-uniform row, and seal row are all explicit entries of the finite packet.

The compact-net and shared-modulus clauses are forced by the single rows N, σ, D, U. The net row remains attached to X, while the family route uses the same schedule and dyadic ledger for every displayed component and every displayed net cell.

The Real seal is downstream of this data. The row Q binds the compact-net and uniform-family routes, and E appears only after Q and U have been displayed.

Finally H, C, P, L transport, replay, source, and name the same rows. Since no additional coordinate appears in the carrier, the listed obligations exhaust the local NameCert surface and exclude the refused compactness, choice, quotient, and completeness routes. □

Theorem 7106.3 (Compact uniform Cauchy family vision route)**.** *Every compact-uniform-continuity consumer that reads an accepted* $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$ *packet factors through the finite route*

$$X, N \longmapsto F, \sigma, R, D \longmapsto U \longmapsto Q \longmapsto E.$$

Thus the consumer receives compact metric finite-net data, a shared $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *family readbacks, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *row, a compact-uniform Cauchy row, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal before any compact-uniform output is visible.*

Proof. Begin with the compact finite-net side. The source row X and net row N are the only compactness coordinates in the carrier, so a consumer must read compact coverage as displayed finite data.

Next use the family route. The rows F, σ, R, D are the same finite family, schedule, regular readback, and dyadic tolerance rows consumed by $\mathsf{UniformCauchyFamily}^{\uparrow}$, so the shared modulus cannot split into component-specific hidden thresholds.

Apply the compact-uniform row Q. It binds the finite net route to the uniform-family route before the Real seal E is exposed, matching the compact finite-net and shared-modulus route described by chapter 9237.

Transport, replay, provenance, and local naming preserve the same finite route. Therefore no downstream consumer obtains open-cover choice, a selected limit, quotient-stream equality, countable choice, host real equality, or ambient completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7106.1, Definition 7106.2, and Definition 7106.3 seed the finite compact-net and shared-modulus family route over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{UniformCauchyFamily}^{\uparrow}$, $\mathsf{CompactUniformCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient-stream completion, selected limits, component-specific hidden moduli, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exactness, shared-modulus exactness, Real-seal nonescape, ledger exhaustion, replay, provenance, and local naming for BEDC.Derived.CompactUniformCauchyFamilyUp.

Constructive story: A $\mathsf{CompactUniformCauchyFamily}^{\uparrow}$ packet is generated from compact metric finite-net data, a displayed finite family row, one shared $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ family readbacks, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{UniformCauchyFamily}^{\uparrow}$ row, a compact-uniform Cauchy row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, displayed replay, provenance, and local naming.

Chapter 7107

A Concrete Naming Certificate for $\mathsf{OneOverNZeroLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{OneOverNZeroLimit}^{\uparrow}$ packet records the elementary convergence of the reciprocal natural-number row to zero as finite BEDC NameCert data. It sits beside the unary $\mathsf{Nat}^{\uparrow}$ index surface of chapter 28, the rational readback surface of chapter 37, the Archimedean tail-bound route of chapter 3372, the stream-window surface of chapter 126, the regular rational sequence surface of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes the displayed threshold route for the classical sequence $n \mapsto 1/(n+1)$ without importing host limits, quotient reals, trichotomy, countable choice, or ambient real completeness.

Definition 7107.1 (One-over-n zero-limit carrier)**.** *A* $\mathsf{OneOverNZeroLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (N, Q, A, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *N is the unary* $\mathsf{Nat}^{\uparrow}$ *index row, read as the finite index window for the reciprocal sequence.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *reciprocal row, displaying the rational value $1/(n+1)$ on the inspected index.*
3. *A is the* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *tail-bound row, assigning a displayed unary threshold to every positive tolerance read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite requests for indices and tolerances are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational reciprocal sequence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the Archimedean threshold and reciprocal readback rows are displayed as a zero-limit witness.*
7. *H records componentwise* hsame *transport for the index, reciprocal, threshold, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through A, W, R, Q, N and into the terminal seal E.*
9. *P is the* Pkg *provenance over* $\mathsf{OneOverNZeroLimit}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RealArchimedeanProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{OneOverNZeroLimit}^{\uparrow}}$ *row naming exactly this reciprocal zero-limit packet.*

The carrier admits no coordinate for a host limit operator, quotient real equality, trichotomy over reals, countable choice over tolerances, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7107.2 (One-over-n zero-limit NameCert obligations)**.** *For every accepted* $\mathsf{OneOverNZeroLimit}^{\uparrow}$ *carrier* $Z = (N, Q, A, W, R, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{OneOverNZeroLimit}^{\uparrow}}$ *surface consists of five obligation rows:*

1. *carrier admission for the displayed unary index row* N *and reciprocal rational row* Q*;*
2. *positive-tolerance routing through the* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *threshold row* A*;*
3. *finite-window and regular-rational readback exactness through* W *and* R*;*
4. *terminal zero-limit sealing through the* $\mathsf{Real}^{\uparrow}$ *row* E *only after* A, W, R, Q, N *are visible;*
5. *componentwise transport, replay, provenance, and local naming closure through* H, C, P, L*.*

Every accepted zero-limit read must display these rows before a consumer may use the reciprocal sequence as a real zero-limit witness.

Proof. Project the carrier of Definition 7107.1. The only index and value data are the unary natural row N and the rational reciprocal row Q, so carrier admission is finite.

The tolerance request is then routed through A. That row is an accepted $\mathsf{RealArchimedeanProperty}^{\uparrow}$ read, so it supplies a displayed unary threshold before any reciprocal tail claim can be consumed.

The window and readback rows W and R keep the sequence inspection finite. They expose the same reciprocal row Q on the inspected indices and do not add a selected infinite subsequence or a host limit.

Finally E seals the displayed tail witness as a $\mathsf{Real}^{\uparrow}$ zero-limit read, while H, C, P, L transport, replay, source, and name the same packet. Since no other coordinate is present, the five listed rows are the complete local NameCert surface. □

Theorem 7107.3 (One-over-n Archimedean tail)**.** *Every positive tolerance read accepted by a* $\mathsf{OneOverNZeroLimit}^{\uparrow}$ *packet receives a displayed unary threshold through* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *before the reciprocal row is sealed as a* $\mathsf{Real}^{\uparrow}$ *zero-limit witness. In particular, the consumer reads the route*

$$\mathsf{RealArchimedeanProperty}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as finite BEDC data, not as a host limit theorem.

Proof. Start with a positive tolerance request. The continuation row C sends that request to the Archimedean row A, where the tolerance receives a displayed unary threshold.

The threshold is read together with N and Q. Past that unary index, the inspected rational reciprocal values are routed through the finite window W and regular readback R.

Only after this route is visible does the seal row E present the zero-limit witness to a $\mathsf{Real}^{\uparrow}$ consumer. The carrier lacks host limits, quotient reals, trichotomy, countable choice, and ambient completeness, so the Archimedean tail is the only exported route. □

Closure status for $\mathsf{NameCert}_{\mathsf{OneOverNZeroLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{OneOverNZeroLimit}^{\uparrow}$ packet is generated from a unary Nat index row, a Rat reciprocal row, a RealArchimedeanProperty tail-bound row, a StreamName window row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneOverNZeroLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7107.1, Definition 7107.2, and Definition 7107.3 seed the finite reciprocal zero-limit route through Nat, Rat, RealArchimedeanProperty, StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{OneOverNZeroLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient reals, trichotomy, countable choice, RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-tolerance threshold routing, finite-window readback exactness, RealUp zero-limit sealing, and transport-replay-provenance naming for $N, Q, A, W, R, E, H, C, P, L$.

Chapter 7108

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$

The $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ packet realizes the phase-space branch of chapter 9463 as finite reversible disk-phase transport, boundary refusal, and provenance rows. It sits after the disk carrier of chapter 377, the Poincare disk metric of chapter 6209, the hyperbolic metric row of chapter 3474, and the reversible transport route of chapter 3455. The packet names a finite phase-space carrier for observer-expansion reads; it does not complete the disk, select boundary representatives, identify a physical observer trajectory, or import a host symplectic phase space.

Definition 7108.1 (Hyperbolic phase-space carrier)**.** *A* $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, D, M, T, B, G, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *source row for the inspected disk observations.*
2. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *metric row over the same displayed disk points.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *readback row used to compare phase displacement through hyperbolic distance.*
4. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *reversible transport row.*
5. *B is the boundary-refusal row, recording that interior-center or selected-boundary exits are unavailable to the packet.*
6. *G is the observer-phase groupoid row: it records finite source, target, inverse, and composition windows for reversible phase moves without asserting a host groupoid of all disk paths.*
7. *R is the route-stability row, preserving U, D, M, T, B, G under the local classifier.*
8. *H records componentwise* hsame *transport for disk, metric, transport, refusal, groupoid, stability, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a phase-space request through $U, D, M, T, B, G, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{PoincareDiskMetric}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, M, T, B, G, R, H, C, P, N$. It has no coordinate for completed disk geometry, selected ideal-boundary points, global geodesic flow, host symplectic form, physical cosmology, or observer substance.

Theorem 7108.2 (Hyperbolic phase-space NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ *carrier* $S = (U, D, M, T, B, G, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseSpace}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, Poincare disk metric visibility, hyperbolic metric readback, reversible phase-transport routing, boundary-refusal recording, finite observer-phase groupoid windows, route stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7108.1. The accepted packet displays exactly $U, D, M, T, B, G, R, H, C, P, N$

The disk and metric rows are read first. The rows U, D, M keep the packet inside displayed disk observations and metric readbacks rather than a completed hyperbolic surface.

The reversible phase data are read next. The row T supplies the phase transport, B records the boundary refusal, and G exposes only finite source, target, inverse, and composition windows for observer-phase movement.

Finally R, H, C, P, N preserve, transport, replay, source, and name that same route. Since none of those rows contains completed disk geometry, selected ideal-boundary points, global geodesic flow, a host symplectic form, physical cosmology, or observer substance, the displayed rows exhaust the local NameCert obligations. □

Theorem 7108.3 (Hyperbolic phase-space boundary refusal stability)**.** *Every phase-space consumer of an accepted* $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ *packet must pass through*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto B, G, R, H, C, P, N.$$

Thus reversible disk-phase transport and finite observer-phase groupoid windows remain available, while interior-center collapse, selected ideal-boundary representatives, completed disk geometry, and host phase-space structure remain outside the exported surface.

Proof. Begin with a phase-space read. The disk carrier U supplies the inspected disk observations, and the metric rows D, M provide the only distance readbacks available to the route.

Read the reversible phase row T. It transports the displayed phase data across the disk surface, but it does not create a selected boundary representative or a global path space.

Use the refusal and groupoid rows B, G. The refusal row blocks interior-center collapse and selected-boundary exits, while the groupoid row records only finite source, target, inverse, and composition windows.

The stability and structural rows R, H, C, P, N preserve the same finite route. They therefore export reversible disk-phase transport and provenance without adding completed disk geometry, host symplectic data, physical cosmology, or observer substance. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ consumer is refuted if it obtains reversible phase transport, boundary refusal, or observer-phase composition without displaying UnitDisk observations, PoincareDiskMetric and HyperbolicMetric reads, PoincareDiskPhaseTransport routing, componentwise transport, replay, provenance, and local naming.

Independence witness: hyperbolic-metric, poincare-disk-metric, poincare-disk-phase-transport, observer-phase-groupoid: no listed sibling is bijective to $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ because this packet jointly carries disk observations, two metric readback rows, reversible phase transport, boundary

refusal, finite groupoid windows, route stability, componentwise transport, replay, provenance, and local naming.

The proposed Lean TasteGate surface for this packet must include explicit Nontrivial HyperbolicPhaseSpace$^{\uparrow}$ and FieldFaithful HyperbolicPhaseSpace$^{\uparrow}$ instances. Those instances must keep $U, D, M, T, B, G, R, H, C, P, N$ as separate rows before any checked carrier-alignment theorem can serve as the public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseSpace}\uparrow}$.

Constructive story: A HyperbolicPhaseSpace$^{\uparrow}$ packet is generated from UnitDisk observations, PoincareDiskMetric and HyperbolicMetric readbacks, PoincareDiskPhaseTransport rows, a boundary-refusal ledger, finite observer-phase groupoid windows, route stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseSpace}\uparrow}$, seed)

Scope closed: Definition 7108.1, Definition 7108.2, and Definition 7108.3 seed the finite hyperbolic phase-space packet over disk observations, metric readbacks, reversible phase transport, boundary refusal, finite groupoid windows, route stability, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed disk geometry, selected ideal-boundary representatives, global geodesic flow, host symplectic structure, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk admission, Poincare and hyperbolic metric visibility, reversible phase routing, boundary-refusal exactness, finite observer-phase groupoid windows, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseSpaceUp.

Chapter 7109

A Concrete Naming Certificate for $\mathsf{UniformLocalConnectedness}^{\uparrow}$

Following Hausdorff, Bourbaki, and Bishop–Bridges metric topology, the $\mathsf{UniformLocalConnectedness}^{\uparrow}$ packet records a finite uniform local-connectedness certificate by which a metric source, local connected basis rows, epsilon-chain rows, compact finite-window compatibility, continuous-map consumers, and Real readbacks are made into NameCert data. It sits over the metric surface of chapter 57, the local basis row of chapter 5809, the chain-connected metric route of chapter 2050, the compact metric finite-window surface of chapter 129, the continuous-map consumer interface of chapter 130, the stream-name schedule of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet gives downstream metric-topology consumers a finite route from local connected neighborhoods to uniform chain windows; it does not assert arbitrary open-cover local connectedness, choose components, import path connectedness, select moduli by choice, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7109.1 (Uniform local connectedness carrier)**.** *A* $\mathsf{UniformLocalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, L, A, K, F, C, \sigma, R, Q, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier, distance classifier, and metric-distance ledger used by every local window.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *local basis row, naming the displayed connected neighborhood family around each inspected point.*
3. *A is the* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *epsilon-chain row, storing the finite chain witnesses available inside the displayed metric ball.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-window compatibility row, restricting compact reads to finite net cells and bounded metric windows.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer row, recording how a continuous-map request may read the local basis and chain rows without selecting a hidden modulus.*
6. *C is the compact local-window row binding the local basis row L, chain row A, and compact window row K to the same inspected metric scale.*
7. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for visible local-window, chain, compact-window, and consumer reads.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for metric radii, chain tolerances, local-basis thresholds, and consumer moduli.*
9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for distance and tolerance comparisons after the finite rows have been displayed.*
10. *H records componentwise* hsame *transport for metric source, local basis, chain row, compact window, continuous-map consumer, compact local-window row, stream schedule, regular readbacks, Real seal, replay, provenance, and local naming.*
11. *T records displayed* Cont *replay from a uniform local-connectedness query through $M, L, A, K, F, C, \sigma, R, Q, H, P,$*
12. *P records* Pkg *provenance over* $\mathsf{UniformLocalConnectedness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{ChainConnectedMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLocalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, A, K, F, C, \sigma, R, Q, H, T, P, N$. It has no coordinate for arbitrary open-cover local connectedness, selected components, path connectedness, choice-selected moduli, quotient equality of neighborhoods, or ambient Real completeness.

Theorem 7109.2 (Uniform local connectedness NameCert obligations)**.** *For every accepted* $\mathsf{UniformLocalConnectedn}$ *carrier*

$$U = (M, L, A, K, F, C, \sigma, R, Q, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformLocalConnectedness}^{\uparrow}}$ *surface consists exactly of metric-source admission, local-basis admission, epsilon-chain admission, compact finite-window compatibility, continuous-map consumer compatibility, compact local-window binding, stream-schedule readback, regular rational comparison, Real seal handoff, componentwise transport, displayed replay, provenance, and local naming for the same finite packet. It also contains the refusal ledger excluding arbitrary open-cover local connectedness, selected components, path connectedness, choice-selected moduli, quotient neighborhood equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7109.1. The metric source, local basis, chain, compact-window, continuous-map consumer, and compact local-window rows are explicit entries of the finite packet.

The uniformity requirement is witnessed by shared rows. The same schedule σ, regular rational readback R, and Real seal Q are used for metric radii, local-basis thresholds, chain tolerances, compact windows, and consumer moduli.

The structural rows preserve the same route. The transport row H identifies corresponding displayed entries, T replays the route through Cont, and P, N record provenance and local naming.

Since no additional coordinate appears in the carrier, the listed rows exhaust the NameCert surface. The omitted open-cover, component-selection, path, choice-modulus, quotient, and ambient-completeness exits are therefore not available to a consumer. □

Theorem 7109.3 (Uniform local connectedness chain-modulus handoff)**.** *Every continuous-map or compact-metric consumer that reads an accepted* $\mathsf{UniformLocalConnectedness}^{\uparrow}$ *packet factors its local connectedness request through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{LocallyConnected}^{\uparrow} \longmapsto \mathsf{ChainConnectedMetric}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{C}$$

Thus the consumer sees a displayed local basis, an epsilon-chain witness, a compact finite-window compatibility row, a stream schedule, regular rational modulus readbacks, and a Real-facing comparison seal before it can export a uniform local-connectedness modulus.

Proof. Begin with the metric and local rows. The source row M supplies distances, while the local basis row L supplies the connected neighborhood family visible at the inspected point.

Next use the chain and compact rows. The epsilon-chain row A records finite chain witnesses inside the displayed metric ball, and the compact row K restricts the read to finite windows compatible with chapter 129.

The readback rows then bind the modulus. The schedule σ, regular rational row R, and Real seal Q expose only displayed thresholds and comparisons, matching the chain-connected handoff of chapter 2050.

Finally the consumer row F may replay the same data through $\mathsf{ContinuousMap}^{\uparrow}$. The rows H, T, P, N preserve transport, replay, provenance, and local naming, so the exported modulus is finite packet data rather than an open-cover, component-selection, path, choice, quotient, or ambient-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLocalConnectedness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLocalConnectedness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7109.1, Definition 7109.2, and Definition 7109.3 seed the finite uniform local-connectedness route over $\mathsf{Met}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{ChainConnectedMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLocalConnectedness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover local connectedness, selected components, path connectedness, choice-selected moduli, quotient neighborhood equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-basis exactness, epsilon-chain exactness, compact finite-window compatibility, continuous-map consumer compatibility, Real-seal handoff, replay, provenance, and local naming for BEDC.Derived.UniformLocalConnectednessUp.

Constructive story: A $\mathsf{UniformLocalConnectedness}^{\uparrow}$ packet is generated from a Metric source row, a LocallyConnected local basis row, a ChainConnectedMetric epsilon-chain row, CompactMetric finite-window compatibility, a ContinuousMap consumer row, a compact local-window binding, StreamName scheduling, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7110

A Concrete Naming Certificate for $\mathsf{GelfandTransform}^{\uparrow}$

Following Gelfand's transform for commutative Banach algebras, the $\mathsf{GelfandTransform}^{\uparrow}$ packet records the finite NameCert surface that evaluates an algebra element against displayed character rows before any spectrum-level quotient or maximal-ideal selection is used. It sits after the algebraic normed source of chapter 2189, the involutive boundary of chapter 211, the duality-facing package of chapter 262, the compact character-window source of chapter 60, the topology carrier of chapter 88, and the scalar seal of chapter 38. The packet names the constructive evaluation route from a displayed element and displayed character row to a displayed scalar read; it does not choose maximal ideals, quotient functions, unrestricted spectra, or ambient completeness.

Definition 7110.1 (Gelfand transform carrier)**.** *A* $\mathsf{GelfandTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, X, T, K, E, V, H, C, P, N)$$

with the following displayed rows:

1. *A is the commutative* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row, exposing the algebra operations, norm ledger, and displayed element under inspection.*
2. *X is the* $\mathsf{CStarAlg}^{\uparrow}$ *compatibility row when the source is read through an involutive normed boundary.*
3. *T is the* $\mathsf{GelfandDuality}^{\uparrow}$ *handoff row, keeping the transform on the same character-facing duality surface.*
4. *K is the compact character-window row read through* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$*, naming only the finite observation windows of displayed characters.*
5. *E is the element row, naming the algebra element whose transform is evaluated.*
6. *V is the scalar evaluation row, landing in the displayed* $\mathsf{Real}^{\uparrow}$ *scalar seal for the value of the character on E.*
7. *H records componentwise* hsame *transport for algebra, involution, duality, compact-window, topology, element, value, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a character-window request through $A, X, T, K, E, V, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{GelfandTransform}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{CStarAlg}^{\uparrow}$, $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{GelfandTransform}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for an unrestricted spectrum, quotient function equality, selected maximal ideals, host algebra homomorphism equality, selected character spaces, or ambient completeness.

Theorem 7110.2 (Gelfand transform NameCert obligations)**.** *For every accepted* $\mathsf{GelfandTransform}^{\uparrow}$ *carrier $G = (A, X, T, K, E, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GelfandTransform}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for $A, X, T, K, E, V, H, C, P, N$;*
2. *Banach-algebra source exposure, requiring all operation, norm, and element reads to pass through the displayed* $\mathsf{BanachAlgebra}^{\uparrow}$ *row A;*
3. *involutive compatibility, requiring any* $\mathsf{CStarAlg}^{\uparrow}$ *read to remain a displayed boundary row X and not an ambient spectral theorem;*
4. *duality handoff, requiring the transform to use the same* $\mathsf{GelfandDuality}^{\uparrow}$ *character-facing row T;*
5. *compact-topology windowing, requiring character observations to pass through K as a finite* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *window;*
6. *character evaluation exactness, requiring the scalar value V to be read only as evaluation of the displayed element E by the displayed character row;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No Gelfand-transform obligation for this packet is stored outside those rows.

Proof. Project the accepted carrier $G = (A, X, T, K, E, V, H, C, P, N)$. The algebraic source is A, with X available only as an involutive compatibility row and T as the existing Gelfand-duality handoff.

The character side is then forced through K. Its compact and topology rows expose finite character windows rather than a selected character space or unrestricted spectrum.

The element and value rows E, V record the actual evaluation. Since V is downstream of the displayed algebra element and displayed character window, it is a finite scalar read rather than quotient function equality.

Finally H, C, P, N preserve, replay, package, and name the same finite surface. The excluded spectrum, quotient, maximal-ideal, host-equality, selected-character, and completeness objects are absent from the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 7110.3 (Gelfand transform character-evaluation route)**.** *Every character-evaluation consumer of an accepted* $\mathsf{GelfandTransform}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{BanachAlgebra}^{\uparrow},\ \mathsf{CStarAlg}^{\uparrow},\ \mathsf{GelfandDuality}^{\uparrow} \longmapsto \mathsf{Compact}^{\uparrow},\ \mathsf{Topology}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For $G = (A, X, T, K, E, V, H, C, P, N)$, the consumer must expose the Banach-algebra source row, optional involutive boundary, Gelfand-duality handoff, compact character-window row, topology row, displayed element, scalar evaluation row, transport, replay, provenance, and local naming before reading the transform value. The route exports no unrestricted spectrum, quotient function equality, selected maximal ideal, selected character space, host homomorphism equality, or ambient completeness.

Proof. Start with a transform-value request. The only algebraic source available to the packet is A, with X and T recording the compatible involutive and Gelfand-duality rows when those surfaces are consumed.

Next expose the character window. The row K is read through the compact and topology sources, so the consumer sees finite character observations instead of a selected maximal ideal or unrestricted spectrum.

Now evaluate the displayed element. The rows E and V bind the scalar read to the element and character window already displayed.

The structural rows H, C, P, N keep the same route under transport, replay, provenance, and local naming. Since the carrier contains no quotient function equality, selected character space, host homomorphism equality, selected maximal ideal, or ambient completeness row, no consumer can obtain those objects from the $\mathsf{GelfandTransform}^{\uparrow}$ packet. □

Falsifiable prediction: An accepted $\mathsf{GelfandTransform}^{\uparrow}$ read is refuted if a character-evaluation consumer obtains a transform value without displaying the BanachAlgebra source, CStarAlgebra compatibility row when used, GelfandDuality handoff, Compact character-window row, Topology row, Real scalar seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: banachalgebra, cstaralgebra, gelfandduality, compact, topology, and real: none is bijective to $\mathsf{GelfandTransform}^{\uparrow}$ because only this packet jointly records displayed algebra-element evaluation, finite character windows, scalar readback, transport, replay, provenance, and local NameCert rows as one transform surface.

Closure status for $\mathsf{NameCert}_{\mathsf{GelfandTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{GelfandTransform}^{\uparrow}$ packet is generated from a commutative BanachAlgebra source row, optional CStarAlg compatibility row, GelfandDuality handoff, Compact and Topology character-window rows, displayed algebra element, Real scalar evaluation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GelfandTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7110.1, Definition 7110.2, and Definition 7110.3 seed the finite character-evaluation route from BanachAlgebra and CStarAlgebra rows through GelfandDuality, Compact and Topology character windows, and Real scalar readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GelfandTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an unrestricted spectrum, quotient function equality, selected maximal ideals, selected character spaces, host homomorphism equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Banach-algebra exposure, involutive compatibility, duality handoff, compact-topology windowing, character-evaluation exactness, transport, replay, provenance, and local naming for BEDC.Derived.GelfandTransformUp.

Chapter 7111

A Concrete Naming Certificate for $\mathsf{UniformLipschitzModulus}^{\uparrow}$

Following Bishop–Bridges constructive metric analysis and Kelley-style uniform-space bookkeeping, the $\mathsf{UniformLipschitzModulus}^{\uparrow}$ packet records the textbook passage from a displayed Lipschitz bound to a uniform modulus as finite certificate data. It reads the Lipschitz modulus route of chapter 3724, the constant handoff of chapter 2328, and the surrounding metric, real, dyadic, uniform-modulus, and uniform-continuity rows. The chapter names the carrier that makes this handoff local to BEDC; it is not a host continuity theorem, a quotient-map equality, a hidden reciprocal selection, or an ambient completeness principle.

Definition 7111.1 (Uniform Lipschitz modulus carrier)**.** *A* $\mathsf{UniformLipschitzModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, F, K, L, E, D, U, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *F is the* $\mathsf{LipschitzMap}^{\uparrow}$ *graph row whose source and target reads use X and Y.*
3. *K is the displayed* $\mathsf{LipschitzConstant}^{\uparrow}$ *row for the same source, target, and graph data.*
4. *L is the accepted* $\mathsf{LipschitzModulus}^{\uparrow}$ *row that turns the displayed bound into a tolerance route.*
5. *E is the positive target-tolerance row, read through* $\mathsf{Real}^{\uparrow}$ *before any source-tolerance request is exported.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division and rounding ledger used by K, L, E.*
7. *U is the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row.*
8. *W is the* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer row that may read U only after X, Y, F, K, L, E, D have been displayed.*
9. *H records componentwise* hsame *transport for metric rows, graph rows, bound rows, modulus rows, tolerance rows, dyadic ledger rows, uniform-continuity consumers, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a target-tolerance request through E, K, L, D, U, W.*

11. *P is the* Pkg *provenance over* $\mathsf{UniformLipschitzModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{LipschitzModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLipschitzModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for host continuity, quotient function equality, implicit reciprocal choice, selected limits, countable choice, or ambient metric or real completeness.

Theorem 7111.2 (Uniform Lipschitz modulus NameCert obligations)**.** *For every accepted* UniformLipschitzModulus *carrier $Q = (X, Y, F, K, L, E, D, U, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformLipschitzModulus}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the two metric rows, Lipschitz graph row, displayed constant row, Lipschitz modulus row, positive tolerance row, dyadic division and rounding ledger, exported uniform modulus, uniform-continuity consumer, transport, replay, provenance, and local naming;*
2. *source-target alignment, requiring F, K, L to read the same X and Y before any tolerance conversion is exported;*
3. *tolerance route exactness, requiring E to be displayed before D and requiring D to be the only division and rounding ledger used by U;*
4. *uniform-consumer handoff, requiring every* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{UniformContinuity}^{\uparrow}$ *read to pass through K, L, E, D, U in that order;*
5. *ledger non-escape, excluding host continuity, quotient map equality, hidden reciprocal choice, selected limits, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 7111.1. The displayed packet exposes exactly the metric, graph, constant, Lipschitz-modulus, tolerance, dyadic, exported-modulus, consumer, transport, replay, provenance, and naming rows.

The source-target alignment obligation follows from the shared positions of X and Y in F, K, L. A consumer cannot read a bound or modulus row for a different metric pair because no such coordinate occurs in Q.

The tolerance and handoff obligations use the same finite route as chapter 3724 and the displayed constant handoff of chapter 2328. The positive tolerance E is read before the dyadic ledger D, and U is the only exported $\mathsf{UniformModulus}^{\uparrow}$ row available to W.

The refusal clause is an inventory check on the carrier. None of the displayed rows supplies host continuity, quotient equality, hidden reciprocal selection, selected limits, countable choice, or ambient completeness. □

Theorem 7111.3 (Uniform Lipschitz modulus consumer boundary)**.** *Let $Q = (X, Y, F, K, L, E, D, U, W, H, C, P, N$ be an accepted* $\mathsf{UniformLipschitzModulus}^{\uparrow}$ *carrier. Every downstream* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer receives the source-tolerance route only through the displayed Lipschitz constant K, the Lipschitz modulus row L, the positive target tolerance E, the dyadic ledger D, and the exported row U. The consumer boundary is therefore a finite BEDC handoff rather than a rereading of a host Lipschitz proof.*

Proof. Project the carrier. The only graph row is F, the only displayed constant row is K, and the only Lipschitz-modulus row is L; all three read the same source and target metric rows.

The target-tolerance request is E. It reaches a source-tolerance request only through the dyadic division and rounding ledger D, so no hidden reciprocal or selected real approximation can replace the finite arithmetic row.

The exported row U is then the sole $\mathsf{UniformModulus}^{\uparrow}$ coordinate available to W. Componentwise transport through H and replay through C preserve this order, while P and N record the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLipschitzModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLipschitzModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7111.1, Definition 7111.2, and Definition 7111.3 seed the finite route from displayed Lipschitz constant and Lipschitz modulus data to a uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLipschitzModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host continuity, quotient map equality, hidden reciprocal choice, selected limits, countable choice, or ambient metric or real completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-target alignment, tolerance route exactness, uniform-consumer handoff, transport stability, and replay stability for BEDC.Derived.UniformLipschitzModulusUp.

Constructive story: A $\mathsf{UniformLipschitzModulus}^{\uparrow}$ packet is generated from source and target metric rows, a Lipschitz graph row, a displayed Lipschitz constant row, an accepted Lipschitz modulus row, a positive tolerance row, a DyadicRatCore division and rounding ledger, a UniformModulus export, a UniformContinuity consumer, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 7112

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverPhase}^{\uparrow}$

The $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ packet realizes the observer phase route of chapter 9463 as a canonical finite carrier. It binds boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, provenance, hsame, Cont, Pkg, and local NameCert rows before any observer-expansion conclusion can be consumed. The packet is BEDC-native phase data: it is not a host hyperbolic manifold, not empirical cosmology, and not a hidden observer center.

The packet consumes the observer-expansion rows of chapter 4167 and the shell aggregation rows of chapter 5180 before its phase row is read. This is a finite boundary-shadow and observer-shell route: the phase carrier records how displayed boundary-facing data pass through shell aggregation, not a geometric primitive and not a host phase space.

Definition 7112.1 (Hyperbolic observer phase carrier)**.** *A* $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, V, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-shadow row read through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*.*
2. *V is the visual-boundary row read through* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*.*
3. *D is the phase-distance row read through* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$*.*
4. *T is the reversible-transport row read through* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*, and the observer phase groupoid surface.*
5. *R is the refusal row blocking hidden interior centers, unlogged global geometry, empirical expansion laws, and private phase-distance oracles.*
6. *H records componentwise* hsame *transport for B, V, D, T, R.*
7. *C records displayed* Cont *replay from boundary-shadow reads through visual-boundary reads, phase-distance reads, reversible transport, refusal, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverPhase}^{\uparrow}$*,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*,* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*,* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for an ambient hyperbolic space, a selected ideal boundary, a physical expansion law, a global geodesic completion, or an interior observer center.

Theorem 7112.2 (Hyperbolic observer phase NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverPhase}$ *carrier* $Q = (B, V, D, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, transport, replay, provenance, and local naming rows;*
2. *boundary coverage, requiring* B *and* V *to be displayed before any phase-distance row is read;*
3. *phase-distance and transport exactness, requiring* D *and* T *to be the only comparison and reversible-transport rows exported by the carrier;*
4. *refusal coverage, requiring* R *to block hidden center, selected boundary, global-geometry, empirical-cosmology, and private-oracle interpretations;*
5. *ledger non-escape, requiring downstream consumers to pass through* H, C, P, N *rather than rereading the vision text as a proof surface.*

Proof. Carrier admission is projection from Definition 7112.1. The geometric rows are exactly B, V, D, T, R, and the structural rows are exactly H, C, P, N.

Boundary coverage follows because B and V are the only boundary-facing rows. A phase-distance read therefore has no route into the carrier until boundary-shadow and visual-boundary data have been displayed.

The exactness and refusal clauses are row-order checks. The phase-distance row D and transport row T provide the only comparison and reversible movement data, while R blocks hidden center, selected boundary, global geometry, empirical law, and private-oracle readings.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further coordinate, every downstream observer-phase read is confined to the displayed BEDC route. □

Theorem 7112.3 (Hyperbolic observer phase downstream boundary)**.** *Every downstream read of an accepted* $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ *packet factors through boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}$ *row. It exports a finite observer-phase interface, not an ambient hyperbolic geometry or a total observer-expansion law.*

Proof. Apply Definition 7112.2. The accepted packet exposes only the nine rows $B, V, D, T, R, H, C, P, N$.

The boundary-shadow and visual-boundary rows keep the read boundary-facing. The phase-distance and reversible-transport rows provide only displayed comparisons and movement, and the refusal row prevents those readings from becoming global geometry or empirical cosmology.

Transport, replay, provenance, and naming preserve the same finite route. A downstream consumer therefore receives the observer-phase interface and no stronger object. □

The later Lean TasteGate for this packet must provide Nontrivial and FieldFaithful instances for the boundary-shadow, visual-boundary, phase-distance, reversible-transport, and refusal rows before any stronger classifier claim is recorded.

Falsifiable prediction: An accepted $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ consumer that exports observer phase data without first displaying boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, hsame, Cont, Pkg, and local NameCert rows refutes this carrier.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-visual-boundary, phase-distance-observer-expansion, poincare-boundary-fixed-transport, observer-phase-groupoid: none is bijective to $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ because this packet jointly carries boundary-facing reads, phase-distance comparison, reversible transport, refusal, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7112.1, Definition 7112.2, and Definition 7112.3 seed the finite observer-phase carrier over boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient hyperbolic geometry, empirical cosmology, selected ideal boundaries, global geodesic completion, or observer-center existence.

Upgrade path: Obligation closure requires checked carrier admission, boundary coverage, phase-distance exactness, reversible-transport exactness, refusal coverage, Nontrivial and FieldFaithful TasteGate instances, transport stability, and downstream nonescape for BEDC.Derived.HyperbolicObserverPhaseUp.

Constructive story: A $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ packet is generated from boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 7113

A Concrete Naming Certificate for $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$ packet records the interval-local regular Cauchy field rows available on a displayed closed bounded interval. It sits after the closed interval source of chapter 1591, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the local Cauchy-field corridor without claiming general $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, or an ambient field-completion principle.

Definition 7113.1 (Closed-interval Cauchy-field carrier)**.** *A* $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed interval source row, exposing the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *lower endpoint, upper endpoint, endpoint-order, containment, finite-window, readback, and Real-seal boundary.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect only the displayed interval-local observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested precision, endpoint containment, and local comparison budget.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same finite window and dyadic tolerance route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-facing value only after I, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, stream window, dyadic tolerance ledger, regular rational readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from interval-source admission through finite windows, tolerance checks, readback, and terminal sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyField}^{\uparrow}}$ *row.*
The local classifier compares accepted carriers only by $I, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient streams, selected limit points, countable choice, host real-field completion, or an ambient proof that every Cauchy family on $\mathsf{Real}^{\uparrow}$ *converges.*

Theorem 7113.2 (Closed-interval Cauchy-field NameCert obligations)**.** *For every accepted* ClosedIntervalCauchyFi *carrier*

$$F = (I, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyField}^{\uparrow}}$ *surface consists exactly of closed interval source exposure, finite StreamName window admission, DyadicRatCore tolerance accounting, RegSeqRat readback, terminal RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local naming row. Every interval-local Cauchy-field read must pass through $I, S, D, Q, E, H, C, P, N$ before it can be consumed.*

Proof. Project the carrier of Definition 7113.1. The rows $I, S, D, Q, E, H, C, P, N$ are the whole accepted packet.

The source and analytic rows are exposed first. The closed interval source I fixes the endpoint and containment boundary, the window row S exposes the finite observations, the dyadic row D records the requested tolerance, and Q reads those observations as regular rational data.

The remaining rows preserve the route. The Real seal E is downstream of I, S, D, Q, while H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, quotient streams, selected limits, countable choice, and ambient completeness cannot be exported by the local NameCert. □

Theorem 7113.3 (Closed-interval Cauchy-field window readback)**.** *Every accepted* $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$ *read that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal first displays the finite lattice*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The readback is interval-local and finite; it does not identify quotient streams, select a global limit, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let $F = (I, S, D, Q, E, H, C, P, N)$ be an accepted carrier. The source row I is the only interval entry point, so the read begins with the closed bounded interval boundary.

The finite window and tolerance rows then determine the local request. The row S exposes the inspected StreamName observations, D records the DyadicRatCore precision and containment budget, and Q replays the same data as RegSeqRat readback.

Only after this route is visible can the seal row E be read. Transport, replay, provenance, and naming do not add a hidden stream quotient or chosen limit; they preserve the displayed finite lattice. □

Theorem 7113.4 (Closed-interval Cauchy-field Real-seal nonescape)**.** *An accepted* $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$ *carrier exports the terminal* $\mathsf{Real}^{\uparrow}$ *row only as the seal of the displayed closed-interval Cauchy-field route. No consumer may read that seal as a proof of general Real completeness, a selected completion point for an arbitrary Cauchy stream, a countable-choice family, or a host field-completion operator.*

Proof. Use Definition 7113.2. Every consumer reaches the local naming row through the closed interval source, finite window, dyadic tolerance ledger, regular readback, and terminal Real seal.

Apply Definition 7113.3. The exported Real row is tied to the finite interval-local lattice and has no coordinate for a quotient stream, arbitrary Cauchy family, selected limit, or countable-choice schedule.

Thus the seal is a terminal row of this packet, not a global completeness theorem. Any consumer that treats it as an ambient field-completion principle leaves the displayed carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyField}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalCauchyField}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7113.1, Definition 7113.2, Definition 7113.3, and Definition 7113.4 seed the interval-local Cauchy-field packet over finite readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, general Real completeness, quotient stream equality, selected limits, countable choice, or an ambient field-completion operator.

Upgrade path: Obligation closure requires checked carrier admission, window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ClosedIntervalCauchyFieldUp.

Chapter 7114

A Concrete Naming Certificate for ProofRelevantNormalJoin$^{\uparrow}$

The ProofRelevantNormalJoin$^{\uparrow}$ packet realizes the checked closed-normal and atom confluence fragments of chapter 9378 as finite NameCert data. It records the proof-relevant join witness available for closed normal sources, the atom handoff, and the beta-star route while keeping the residual-substitution compatibility boundary visible. The proof-relevant join witness consumes the confluence-audit rows of chapter 735, the closed-normal seal of chapter 741, the critical-path pressure rows of chapter 765, the audit-witness boundary of chapter 745, and the closed-normal join certificate of chapter 6833. These sibling routes supply the closed-normal and atom fragment rows while the packet retains the residual-substitution compatibility boundary and the conditional ledger for full confluence. The packet is not a full confluence theorem for arbitrary closed terms and does not discharge the conditional parallel-diamond premise.

Definition 7114.1 (Proof-relevant normal-join carrier)**.** *A* ProofRelevantNormalJoin$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, W, B, L, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the closed-normal source row, exposing the closed term, normal-source classifier, and atom cases admitted by the checked fragment.*
2. *W is the proof-relevant join witness row, recording the finite common reduct evidence for the closed-normal or atom branch.*
3. *B is the beta-star handoff row, carrying the transport from the parallel or atom join witness into the multi-step beta-star route.*
4. *L is the conditional-boundary ledger, recording that the full substitution-compatibility input and arbitrary closed-source confluence remain outside this packet.*
5. *T records componentwise* hsame *transport for source, witness, beta-star handoff, boundary ledger, replay, provenance, and local naming rows.*
6. *H records displayed* Cont *replay from closed-normal source admission through join witness construction, beta-star handoff, and boundary retention.*
7. *C is the* Pkg *provenance over* ProofRelevantNormalJoin$^{\uparrow}$*, MetaCIC closed-normal confluence, atom joins, beta-star transport,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

8. *P records the public readback route for the proof-relevant normal join, separated from the conditional full-confluence interface.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofRelevantNormalJoin}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only by $S, W, B, L, T, H, C, P, N$. It has no coordinate for unrestricted substitution compatibility, full parallel diamond, arbitrary closed Church–Rosser, subject reduction, host conversion equality, or an erased proof-irrelevant confluence oracle.

Theorem 7114.2 (Proof-relevant normal-join NameCert obligations)**.** *For every accepted* ProofRelevantNormalJoin *carrier*

$$J = (S, W, B, L, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProofRelevantNormalJoin}^{\uparrow}}$ *surface consists exactly of closed-normal source exposure, proof-relevant join witness, beta-star handoff, conditional-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, public readback, and local naming. Every public read factors through those rows before it can cite the closed-normal or atom confluence fragment.*

Proof. Project the carrier of Definition 7114.1. The rows $S, W, B, L, T, H, C, P, N$ are the whole packet.

The source and witness rows carry the positive fragment. The row S admits only the closed-normal and atom cases, W records the finite common-reduct witness, and B transports that witness into the beta-star route.

The boundary and structural rows preserve the refusal. The ledger L keeps the substitution-compatibility and arbitrary closed-source boundaries visible, while T, H, C, P, N transport, replay, source, read back, and name exactly the displayed rows. □

Theorem 7114.3 (Proof-relevant normal-join diamond fragment)**.** *Every accepted* $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ *public read exposes a closed-normal or atom source row, a proof-relevant common-reduct witness, and a beta-star handoff before it cites the checked confluence fragment. The read is a finite closed-fragment diamond packet and not a proof of full parallel diamond.*

Proof. Use Definition 7114.2. A public read must first pass through S, W, B before reaching transport, replay, provenance, readback, and local naming.

The witness row W is proof-relevant: it names the common reduct evidence for the admitted closed-normal or atom branch rather than merely asserting existence. The beta-star handoff B records the route by which that witness is consumed by the star relation.

Since the ledger L remains a required row, the read cannot be promoted into a full parallel diamond. It is exactly the displayed finite diamond fragment. □

Theorem 7114.4 (Proof-relevant normal-join full-confluence nonescape)**.** *An accepted* $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ *carrier cannot export unrestricted MetaCIC confluence. Any public consumer must retain the conditional-boundary ledger separating closed-normal and atom joins from arbitrary closed-source confluence, residual substitution compatibility, subject reduction, and host conversion equality.*

Proof. Start from the carrier rows of Definition 7114.1. The only positive confluence data are the closed-normal source, proof-relevant witness, and beta-star handoff rows.

Apply Definition 7114.3. The exported diamond fragment is available only for the admitted closed-normal and atom branches.

The boundary ledger L is part of every accepted read by Definition 7114.2. Therefore a consumer that drops residual-substitution compatibility, arbitrary closed-source confluence, subject reduction, or host conversion equality leaves the packet. □

Definition 7114.5 (MetaCIC proof-relevant candidate-diamond carrier)**.** *A MetaCIC proof-relevant candidate-diamond carrier is a finite* BHist *packet*

$$D = (R, L_1, L_2, Q, B, E, H, C, P, O, N)$$

with the following displayed rows:

1. *R is the candidate source row, naming the inspected MetaCIC expression, its closedness boundary, and the classifier row used by the confluence audit.*
2. *L_1 and L_2 are the left and right one-step rows from the same source row.*
3. *Q is the proof-relevant common-reduct witness row for the candidate pair.*
4. *B is the beta-star handoff from L_1, L_2, Q into the multi-step reduction route.*
5. *E is the residual-boundary ledger, recording the exact residual-substitution, subject-reduction, arbitrary closed-source confluence, and host-conversion boundaries still outside the carrier.*
6. *H records componentwise* hsame *transport for source, one-step, witness, beta-star, boundary, replay, provenance, public readback, and local naming rows.*
7. *C records displayed* Cont *replay from the candidate source request through the two one-step rows, witness row, beta-star handoff, and boundary ledger.*
8. *P is the* Pkg *provenance over* ProofRelevantNormalJoin$^{\uparrow}$*, MetaCIC one-step reduction, beta-star handoff, confluence-audit rows, residual-boundary rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *O is the public readback row, exposing the candidate-mediated local diamond only after the proof-relevant witness and the residual-boundary ledger are both visible.*
10. *N is the local* NameCert *row for the candidate-diamond packet.*

The classifier compares accepted candidate-diamond packets only by $R, L_1, L_2, Q, B, E, H, C, P, O, N$. It has no coordinate for arbitrary closed-source confluence, residual-substitution compatibility, subject reduction, host conversion equality, or a proof-erased diamond oracle.

Theorem 7114.6 (MetaCIC proof-relevant candidate-diamond NameCert obligations)**.** *For every accepted MetaCIC proof-relevant candidate-diamond carrier*

$$D = (R, L_1, L_2, Q, B, E, H, C, P, O, N),$$

the local NameCert *surface consists exactly of candidate source exposure, left and right one-step row exposure, proof-relevant common-reduct witness exposure, beta-star handoff, residual-boundary ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, public readback, local naming, and nonescape from arbitrary closed-source confluence, residual-substitution compatibility, subject reduction, and host conversion equality.*

Proof. Project the carrier of Definition 7114.5. The source and two one-step rows are R, L_1, L_2, so any candidate pair is read through a single displayed source classifier and two displayed one-step reductions.

The proof-relevant join data are Q and B. The witness row Q names the common reduct for this candidate pair, and the beta-star handoff B records how the two one-step rows reach that common reduct in the multi-step route.

The boundary and structural rows are E, H, C, P, O, N. The ledger E keeps the residual-substitution, subject-reduction, arbitrary closed-source, and host-conversion boundaries visible, while H, C, P, O, N transport, replay, source, read back, and name only the displayed rows.

Because no further coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations and no proof-erased diamond oracle is exported. □

Theorem 7114.7 (MetaCIC proof-relevant candidate-diamond boundary)**.** *An accepted MetaCIC proof-relevant candidate-diamond packet exports a candidate-mediated local diamond only when the proof-relevant common-reduct witness Q and the residual-boundary ledger E are both public rows. The packet may hand the displayed candidate pair to the beta-star route through B and public readback row O, but it still refuses arbitrary closed-source confluence, residual-substitution compatibility, subject reduction, and host conversion equality.*

Proof. Use Definition 7114.6. A public candidate-diamond read must expose the source row, the two one-step rows, the proof-relevant witness, the beta-star handoff, and the residual-boundary ledger.

The local diamond is therefore candidate-mediated. It is witnessed by Q for the displayed pair L_1, L_2 and is transported into beta-star by B, then exposed by the readback row O.

The same read must retain E. Since E names the residual-substitution, subject-reduction, arbitrary closed-source, and host-conversion boundaries as still outside the packet, removing those boundaries would leave the accepted candidate-diamond carrier. □

Falsifiable prediction: Every public $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ read factors through closed-normal source, join witness, beta-star handoff, conditional-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any confluence-fragment output is visible.

Independence witness: metacic-confluence-audit-packet, closednormalconfluenceseal, metaciccriticalpath, metacicconfluenceauditwitness: $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a proof-relevant join witness, beta-star handoff, conditional-boundary ledger, transport, replay, provenance, public readback, and local naming while refusing arbitrary closed-source confluence and unrestricted substitution compatibility.

Later Lean TasteGate work for $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ must provide explicit Nontrivial ProofRelevantNormalJo and FieldFaithful $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ instances, preserving the source, witness, beta-star handoff, conditional-boundary ledger, transport, replay, provenance, readback, and NameCert rows as distinct fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofRelevantNormalJoin}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofRelevantNormalJoin}^{\uparrow}$ packet is generated from a closed-normal source row, proof-relevant join witness, beta-star handoff, conditional-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, public readback, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProofRelevantNormalJoin}^{\uparrow}}$, seed)

Scope closed: Definition 7114.1, Definition 7114.2, Definition 7114.3, Definition 7114.4, and Definition 7114.7 seed the proof-relevant closed-normal, atom join, and candidate-diamond boundary packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProofRelevantNormalJoin}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, full parallel diamond, arbitrary closed Church–Rosser, residual substitution compatibility, subject reduction, or host conversion equality.

Upgrade path: Obligation closure requires checked carrier admission, proof-relevant witness readback, beta-star handoff, conditional-boundary nonescape, Nontrivial, FieldFaithful, transport, replay, provenance, and local naming for BEDC.Derived.ProofRelevantNormalJoinUp.

Chapter 7115

A Concrete Naming Certificate for McShaneExtensionReal$^{\uparrow}$

Following McShane and Bishop–Bridges constructive real analysis, the McShaneExtensionReal$^{\uparrow}$ packet records the finite BEDC surface for reading a real-valued Lipschitz map on a displayed subset of a displayed metric carrier as an ambient extension candidate. It sits after the metric source of chapter 57, the terminal real-name seal of chapter 38, the reusable Lipschitz extension interface of chapter 3650, the uniformly continuous extension handoff of chapter 769, and the dyadic tolerance core of chapter 291. The chapter isolates the Real-valued McShane handoff as a finite NameCert packet rather than a host supremum formula, quotient function, selected family of extrema, countable-choice construction, or ambient completeness theorem.

Definition 7115.1 (McShane real-valued extension carrier)**.** *A* McShaneExtensionReal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, d, G, L, E, S, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the subset-source row, recording admitted source points, inclusion into the ambient metric carrier, and the finite membership evidence used by source-value reads.*
2. *d is the* Met$^{\uparrow}$ *distance row for the ambient carrier and the source subset, exposing only displayed distance observations.*
3. *G is the Real-valued source graph row, read through* Real$^{\uparrow}$ *seals and carrying the displayed source values without a host function object.*
4. *L is the Lipschitz-constant budget row, recording the finite bound ledger consumed by* LipschitzExtension$^{\uparrow}$ *and the dyadic tolerance requests mediated by* DyadicRatCore$^{\uparrow}$*.*
5. *E is the ambient extension graph row, recording the finite McShane envelope comparisons available at ambient inputs without selecting a supremum or infimum.*
6. *S is the subset-agreement row, requiring every source-admitted ambient read to return through the same A, d, G, L, E route before it can be identified with the source graph.*
7. *R is the Real seal row for source values, envelope comparisons, and ambient extension reads.*
8. *H records componentwise* hsame *transport for the subset source, metric distance, graph, budget, extension graph, agreement, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through $A, d, G, L, E, S, R, H, P, N$.*
10. *P is the* Pkg *provenance over* McShaneExtensionReal$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, LipschitzExtension$^{\uparrow}$, UniformContinuousExten DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{McShaneExtensionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for host function equality, selected suprema, selected infima, quotient functions, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, or a public bridge credential.*

Theorem 7115.2 (McShane real-valued NameCert obligations)**.** *For every accepted* McShaneExtensionReal$^{\uparrow}$ *carrier*

$$M = (A, d, G, L, E, S, R, H, C, P, N),$$

the local NameCert$_{\text{McShaneExtensionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the subset source, metric distance, Real-valued graph, Lipschitz-constant budget, extension graph, subset agreement, Real seal, transport, replay, provenance, and local naming rows;*
2. *source-domain compatibility, requiring every source-value read to pass through A and d before it is compared with an ambient extension read;*
3. *Real graph admission, requiring G and R to be the only rows that expose source values and ambient extension values;*
4. *Lipschitz-budget transport, requiring L and* DyadicRatCore$^{\uparrow}$ *to mediate every tolerance transfer used by the extension graph row;*
5. *extension-graph exactness, requiring E to store only finite McShane comparison data and no selected supremum, selected infimum, quotient function, or host function object;*
6. *subset-agreement exactness, requiring S to identify the extension graph with the source graph only through the same displayed A, d, G, L, E, R route;*
7. *ledger non-escape, requiring H, C, P, N to preserve and replay the listed rows without adding host Real equality, selected extrema, quotient functions, countable choice, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7115.1. The accepted packet displays exactly the subset source, metric distance, Real-valued graph, Lipschitz budget, extension graph, subset agreement, Real seal, and structural rows.

Read the source-domain and graph clauses next. The row A admits source points, the distance row d supplies the metric observations, and G is the only Real-valued source graph row. Hence a source comparison cannot be formed before the packet has opened A, d, G, R.

The Lipschitz-budget clause is forced by L together with the dyadic tolerance core. The reusable interface of chapter 3650 and the dyadic route of chapter 291 keep every tolerance transfer finite and visible.

Project the extension and agreement rows E and S. The extension row stores finite McShane envelope comparisons, while the agreement row checks those comparisons on inputs admitted by A. Neither row selects an extremum, forms a quotient function, or imports a host equality of functions.

Finally inspect H, C, P, N. Transport is componentwise, replay follows the displayed request route, and provenance cites only the packet and its sibling dependencies, including chapter 769. Therefore the local NameCert obligations are exactly the listed finite rows. □

Theorem 7115.3 (McShane real-valued Lipschitz handoff)**.** *Let $M = (A, d, G, L, E, S, R, H, C, P, N)$ be an accepted* $\mathsf{McShaneExtensionReal}^{\uparrow}$ *carrier. Every* $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the ambient extension read must first traverse the finite route*

$$A, d \longmapsto G \longmapsto L \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto E \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

Along that route the displayed Lipschitz budget remains the only bound used for the Real-valued ambient extension graph. The handoff exports no host function equality, selected supremum, selected infimum, quotient function, countable-choice selector, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an ambient extension read. The subset row A and distance row d provide the only source-domain and metric observations available to the consumer, while G supplies the source Real-valued graph.

Read the Lipschitz budget and dyadic tolerance rows. The row L records the displayed bound, and the $\mathsf{DyadicRatCore}^{\uparrow}$ route turns that bound into finite precision data before any ambient comparison is inspected.

The extension graph row E is then opened as finite McShane comparison data, and the subset-agreement row S checks that the same route restricts back to G on inputs admitted by A. The Real seal R is therefore reached only after the source graph, budget, extension graph, and agreement rows are visible.

Transport, replay, provenance, and local naming preserve the route through H, C, P, N. Since no listed row contains host function equality, selected extrema, quotient functions, countable choice, trichotomy, or ambient Real completeness, the $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff cannot export any such object. □

Falsifiable prediction: An accepted $\mathsf{McShaneExtensionReal}^{\uparrow}$ consumer that reaches an ambient Real-valued extension read without first reading the subset source, metric distance, source graph, Lipschitz budget, dyadic tolerance, extension graph, subset agreement, and Real seal rows refutes the carrier surface.

Independence witness: metric, real, lipschitz-extension, uniform-continuous-extension, dyadicratcore: no listed sibling supplies the full eleven-row packet $M = (A, d, G, L, E, S, R, H, C, P, N)$, because each sibling contributes only a metric source, Real seal, reusable extension interface, continuity handoff, or dyadic tolerance route while this chapter binds the finite Real-valued McShane extension certificate into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{McShaneExtensionReal}^{\uparrow}}$.

Constructive story: A $\mathsf{McShaneExtensionReal}^{\uparrow}$ packet is generated from a subset-source row, a MetricUp distance row, a Real-valued graph row, a Lipschitz-constant budget, an ambient extension graph, a subset-agreement row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{McShaneExtensionReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7115.1, Definition 7115.2, and Definition 7115.3 seed the finite Real-valued McShane extension certificate over displayed subset, metric, Lipschitz-budget, dyadic-tolerance, and Real-seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{McShaneExtensionReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected suprema, selected infima, quotient functions, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-domain compatibility, Real graph admission, Lipschitz-budget transport, extension-graph exactness, subset-agreement exactness, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.McShaneExtensionRealUp.

Chapter 7116

A Concrete Naming Certificate for $\mathsf{BanachSaks}^{\uparrow}$

Following Banach and Saks on Cesaro-convergent subsequences in Banach spaces, the $\mathsf{BanachSaks}^{\uparrow}$ packet records the finite NameCert surface by which a bounded Banach-space sequence, a weak-compactness handoff, a Cesaro block schedule, and normed real readback are exposed before any consumer treats the route as a Banach–Saks conclusion. It sits after the Banach carrier of chapter 90, the weak-compactness packet of chapter 6851, the bounded real sequence source of chapter 1415, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Cesaro-subsequence certificate; it does not choose an infinite subsequence in the host, quotient sequence equality, countable choice, weak compactness itself, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7116.1 (Banach–Saks carrier)**.** *A* $\mathsf{BanachSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, X, W, A, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row, naming the normed complete carrier through the already displayed Banach packet.*
2. *X is the bounded Banach sequence source row, read through the bounded sequence frontier and not through a host sequence object.*
3. *W is the* $\mathsf{WeakCompactness}^{\uparrow}$ *handoff row, recording the weakly compact source surface that licenses a subsequence obligation without selecting a host subsequence.*
4. *A is the finite Cesaro block schedule, listing the inspected block indices and averaging windows.*
5. *Q is the norm and* $\mathsf{Real}^{\uparrow}$ *readback row for the displayed Cesaro averages.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *route by which the real-valued norm observations are read as regular rational windows.*
7. *T records transport from the bounded-source and weak-compactness rows to the Cesaro schedule and real readback rows.*
8. *H records componentwise* hsame *transport for B, X, W, A, Q, R, T and the local naming row.*
9. *C records displayed* Cont *replay through the bounded source, weak-compactness handoff, Cesaro schedule, norm readback, regular rational route, transport, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BanachSaks}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachSaks}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, X, W, A, Q, R, T, H, C, P, N$. It has no coordinate for host subsequence choice, quotient sequence equality, countable choice, weak compactness as a theorem, completed Cesaro convergence outside the displayed windows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7116.2 (Banach–Saks NameCert obligations)**.** *For every accepted* $\mathsf{BanachSaks}^{\uparrow}$ *carrier $S = (B, X, W, A, Q, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachSaks}^{\uparrow}}$ *surface consists exactly of Banach source exposure through B; bounded sequence exposure through X; weak-compactness handoff through W; finite Cesaro block scheduling through A; norm and* $\mathsf{Real}^{\uparrow}$ *readback through Q;* $\mathsf{RegSeqRat}^{\uparrow}$ *routing through R; transport through T and H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7116.1. The rows $B, X, W, A, Q, R, T, H, C, P, N$ are exactly the Banach source, bounded source, weak-compactness handoff, Cesaro schedule, readback, route, transport, replay, provenance, and naming coordinates.

The analytic content is split between the source rows and the schedule rows. The rows B, X, W expose the Banach, boundedness, and weak-compactness data, while A, Q, R expose only finite Cesaro averages and finite real readback.

Finally T, H, C, P, N preserve and name the same rows. Since no coordinate names a host subsequence selector, quotient sequence equality, countable-choice object, independent weak-compactness theorem, or ambient completeness principle, the displayed rows exhaust the local NameCert surface. □

Theorem 7116.3 (Banach–Saks Cesaro subsequence route)**.** *Every Cesaro-subsequence consumer of an accepted* $\mathsf{BanachSaks}^{\uparrow}$ *packet must read the route*

$$B, X, W, A, Q, R, T, H, C, P, N$$

in that order: the $\mathsf{Banach}^{\uparrow}$ *source row and bounded sequence row precede the* $\mathsf{WeakCompactness}^{\uparrow}$ *handoff, the handoff precedes the displayed Cesaro block schedule, and the schedule precedes normed* $\mathsf{Real}^{\uparrow}$ *readback and* $\mathsf{RegSeqRat}^{\uparrow}$ *routing. The route exports no host subsequence choice, quotient sequence equality, countable choice, weak-compactness theorem, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the source rows. The Banach carrier B and bounded sequence row X are the only sequence-source coordinates available to the packet.

Next read the weak-compactness handoff W and the finite schedule A. The handoff names the obligation surface, while A records only displayed block indices and averaging windows.

The readback rows Q and R then expose norm observations as finite real and regular-rational windows. Transport, replay, provenance, and local naming through T, H, C, P, N preserve the same route. Therefore the accepted packet supplies a finite Cesaro-subsequence route and none of the refused host principles. □

Falsifiable prediction: Every accepted $\mathsf{BanachSaks}^{\uparrow}$ consumer reaches Cesaro-subsequence data only through the displayed Banach source, bounded sequence source, weak-compactness handoff,

Cesaro block schedule, norm and Real readback, RegSeqRat route, transport, replay, provenance, and local NameCert rows.

Independence witness: banach, weakcompactness, boundedrealsequence, regseqrat, real: none is bijective with $\mathsf{BanachSaks}^{\uparrow}$ because this packet alone binds a bounded Banach sequence to a weak-compactness handoff and a finite Cesaro block schedule before the normed Real readback route is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachSaks}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7116.1, Definition 7116.2, and Definition 7116.3 seed the finite Cesaro-subsequence NameCert route over $\mathsf{Banach}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachSaks}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsequence choice, quotient sequence equality, countable choice, weak compactness as a theorem, selected Cesaro limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exactness, weak-compactness handoff, Cesaro schedule exactness, norm and Real readback, RegSeqRat routing, transport, replay, provenance, and local naming for BEDC.Derived.BanachSaksUp.

Constructive story: A $\mathsf{BanachSaks}^{\uparrow}$ packet is generated from a Banach source row, bounded sequence source, weak-compactness handoff, finite Cesaro block schedule, normed Real readback, RegSeqRat route, transport rows, displayed replay, provenance, and local naming.

Chapter 7117

A Concrete Naming Certificate for $\mathsf{CauchyNullModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyNullModulus}^{\uparrow}$ packet records the explicit modulus route by which a displayed regular-Cauchy difference is shown to be a null sequence before a $\mathsf{Real}^{\uparrow}$ equality seal is read. It sits between the zero-convergence classifier of chapter 380, the equality-facing difference route of chapter 1758, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational source of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the explicit threshold row consumed by Real and Completion readers; it is not a quotient of streams, a selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7117.1 (Cauchy null-modulus carrier)**.** *A* $\mathsf{CauchyNullModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, T, W, B, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row whose inspected observations are the rational difference stream being tested for zero convergence.*
2. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request row, including the positive dyadic budget requested by a downstream equality or completion-facing consumer.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window in which the difference row is inspected for the requested tolerance.*
4. *B is the explicit* $\mathsf{ZeroTailModulus}^{\uparrow}$ *threshold row, binding each tolerance request in T to a finite tail index and the corresponding window in W.*
5. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *row that consumes D, T, W, B as the displayed zero-tail witness.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal, readable only after Z has consumed the explicit threshold row.*
7. *H records componentwise* hsame *transport for difference rows, tolerance requests, windows, thresholds, null-sequence rows, equality seals, replay rows, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the consumer request to the tolerance row, from the tolerance row to the threshold and window rows, from those rows to the null-sequence classifier, and from the classifier to the equality seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyNullModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ZeroTailModulus}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNullModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, W, B, Z, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7117.2 (Cauchy null-modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNullModulus}^{\uparrow}$ *carrier $M = (D, T, W, B, Z, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyNullModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed difference, tolerance, window, threshold, NullSequence, Real seal, transport, replay, provenance, and naming rows; threshold exactness from each tolerance request in T to the finite row B; window compatibility between B and W; NullSequence consumption of D, T, W, B through Z; and Real-seal nonescape through E.*

Proof. Carrier admission is projection from Definition 7117.1. The accepted packet displays exactly the difference row, tolerance request, finite window, explicit threshold, null-sequence consumer, Real equality seal, transport, replay, provenance, and local naming rows.

Threshold exactness is the next obligation. The row T supplies the requested dyadic tolerance, while B is the only coordinate that can answer that request with a finite tail index. The window row W is then the finite observation surface attached to that threshold, so the route from tolerance to inspection is visible before zero-convergence is consumed.

Now apply the null-sequence row. The row Z receives the difference row D only together with the tolerance, window, and threshold rows T, W, B, and therefore records explicit zero-tail evidence rather than a host limit.

Finally inspect the carrier inventory. Since quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking are absent from M, none becomes a local NameCert obligation. □

Theorem 7117.3 (Cauchy null-modulus Real equality route)**.** *Let $M = (D, T, W, B, Z, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyNullModulus}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *equality read that uses the displayed regular-Cauchy difference must first expose the finite route*

$$D, T \longrightarrow B, W \longrightarrow Z \longrightarrow E$$

as replayed by H, C, P, N. The equality seal is therefore reached through an explicit null-modulus threshold and not through quotient stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the equality-facing consumer to the rows of M. The consumer tolerance is visible as the dyadic row T, and the difference stream under test is visible as the regular rational difference row D.

The explicit modulus route then supplies the threshold row B and finite window row W. These rows bind the requested tolerance to a displayed tail index and to the inspected finite stream window.

Only after that route is exposed can Z certify the displayed difference as a null sequence. The transport, replay, provenance, and local naming rows H, C, P, N preserve the same finite route when the terminal seal E is read.

The refused alternatives are absent from the carrier. Thus the $\mathsf{Real}^{\uparrow}$ equality seal is a consumer of the explicit null-modulus route, not a quotient equality, selected limit, trichotomy branch, countable-choice schedule, or ambient completeness theorem. □

Theorem 7117.4 (Cauchy null-modulus tail-budget scope)**.** *Every tail-budget read from an accepted* $\mathsf{CauchyNullModulus}^{\uparrow}$ *packet remains inside the rows*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{ZeroTailModulus}^{\uparrow}, \mathsf{NullSequence}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The read may replay displayed difference observations, dyadic tolerances, finite windows, explicit thresholds, null-sequence consumption, Real equality seals, transport, provenance, and local naming, but it cannot import a quotient stream, a host limit, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge certificate.*

Proof. Begin with the carrier inventory of Definition 7117.1. The only analytic source row is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ difference row D, and the only tolerance source is the $\mathsf{DyadicRatCore}^{\uparrow}$ row T.

The tail budget is then exhausted by B and W. The row B gives the explicit threshold for the tolerance request, while W names the finite window in which the tail observation is inspected.

The downstream rows Z and E consume this evidence without adding a new source of analytic data. Transport, replay, provenance, and local naming rows preserve the same finite package.

Because no quotient stream, host limit, selected representative, countable-choice schedule, trichotomy row, ambient completeness theorem, or bridge certificate is present, every tail-budget read remains inside the stated scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNullModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNullModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7117.1, Definition 7117.2, Definition 7117.3, and Definition 7117.4 seed the explicit threshold route from a regular-Cauchy difference row through DyadicRatCore tolerance requests, StreamName windows, a ZeroTailModulus threshold, NullSequence consumption, and a terminal Real equality seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNullModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, window compatibility, NullSequence consumption, Real-seal nonescape, tail-budget scope, and TasteGate field faithfulness for BEDC.Derived.CauchyNullModulusUp.

Constructive story: A $\mathsf{CauchyNullModulus}^{\uparrow}$ packet is generated from one RegSeqRat difference row, one DyadicRatCore tolerance request row, one StreamName finite window, one explicit ZeroTailModulus threshold row, one NullSequence row, one Real equality seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7118

A Concrete Naming Certificate for $\mathsf{DyadicCauchyStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicCauchyStream}^{\uparrow}$ packet records a dyadic-valued Cauchy stream as finite $\mathsf{StreamName}^{\uparrow}$ observations with an explicit dyadic Cauchy modulus. It sits between the dyadic arithmetic core of chapter 291, the finite stream observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite Cauchy-stream handoff for the L10 Real and Completion spine; it does not quotient streams, select a limit, invoke countable choice, use trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7118.1 (Dyadic Cauchy stream carrier)**.** *A* $\mathsf{DyadicCauchyStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, M, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the finite observation schedule for the stream.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic observation row recording the inspected values, arithmetic comparisons, and tolerance entries.*
3. *M is the dyadic Cauchy modulus row assigning each requested dyadic tolerance a displayed tail window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream observations, dyadic arithmetic rows, and modulus windows.*
5. *H records componentwise* hsame *transport for S, D, M, R and for the replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the route*

$$S, D, M \longmapsto R.$$

7. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyStream}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows S, D, M, R, H, C, P, N. It has no coordinate for quotient stream equality, a selected limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7118.2 (Dyadic Cauchy stream NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyStream}^{\uparrow}$ *carrier*

$$K = (S, D, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyStream}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, dyadic Cauchy modulus row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same displayed rows; exactness of every requested dyadic tail window through M and D; and ledger nonescape for quotient streams, selected limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7118.1. The projection exposes the stream schedule, dyadic observation row, Cauchy modulus row, regular rational readback, and structural rows as finite coordinates.

Classifier stability is componentwise on those displayed rows. Exactness is read by pairing a requested tolerance in M with the dyadic comparisons recorded in D and the finite observation window named by S.

The transport, replay, provenance, and naming rows preserve the same route. Since none of the excluded coordinates occurs in the carrier, the local NameCert obligations are precisely the displayed rows and their finite ledger checks. □

Theorem 7118.3 (Dyadic Cauchy stream regularity stability)**.** *Let $K = (S, D, M, R, H, C, P, N)$ be an accepted* $\mathsf{DyadicCauchyStream}^{\uparrow}$ *carrier. Any finite dyadic Cauchy modulus window displayed by M and checked against D transports through the componentwise* hsame *rows in H and replays through the displayed* Cont *row C as the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback window R. The transported window remains a finite dyadic observation route and does not become a quotient stream or selected limit.*

Proof. Read the modulus window from M together with the dyadic comparisons in D. The source of every inspected value is the finite schedule S, so the window is already a finite observation route.

The row H moves S, D, M, R componentwise. Thus the tolerance request, dyadic comparisons, and readback row keep the same roles after transport.

The continuation row C replays only the displayed route from stream observations and dyadic windows to regular rational readback. Therefore the replayed object is still the same finite $\mathsf{RegSeqRat}^{\uparrow}$ window, not a quotient stream, selected limit, or ambient completeness witness. □

Theorem 7118.4 (Dyadic Cauchy stream Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{DyadicCauchyStream}^{\uparrow}$ *packet receives only the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and terminal seal rows named by H, C, P, N. No consumer receives a quotient stream, selected limit, countable-choice schedule, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle from this packet.*

Proof. Apply Definition 7118.2. A Real-facing consumer can inspect the source schedule, dyadic observation row, modulus row, regular rational readback, transport, replay, provenance, and local naming rows, and no other coordinate.

The handoff to the Real boundary is therefore mediated by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows together with the structural seal rows. The packet has no coordinate for quotienting streams or selecting a completed limit.

The excluded operations also cannot be produced by transport or replay: H is componentwise, and C replays only the displayed finite route. Hence the handoff is exactly the finite Real-and-Completion surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyStream}^{\uparrow}$ packet is generated from a StreamName source row, DyadicRatCore observation rows, an explicit dyadic Cauchy modulus, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7118.1, Definition 7118.2, Definition 7118.3, and Definition 7118.4 seed the finite Cauchy-stream bridge through dyadic observations, stream windows, regular rational readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic modulus exactness, regularity stability, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCauchyStreamUp.

Chapter 7119

A Concrete Naming Certificate for $\mathsf{TailWindowFilter}^{\uparrow}$

Following Bishop's filter language and the Real completeness route of chapter 9238, the $\mathsf{TailWindowFilter}^{\uparrow}$ packet records cofinal finite tail windows for completion-facing Cauchy data. It sits over the finite stream observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only a finite tail-filter carrier; it does not quotient filters, choose a limiting representative, invoke countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7119.1 (Tail-window filter carrier)**.** *A* $\mathsf{TailWindowFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite tail-index row naming the inspected lower bounds for the filter.*
2. *W is the cofinal finite-window row assigning each requested tail bound a displayed window beyond it.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested bounds and window estimates.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed after the tail window has been displayed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the observation schedule for the window.*
6. *H records componentwise* hsame *transport for T, W, D, R, S and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*
$$T, W, D, S \longmapsto R.$$
8. *P is the* Pkg *provenance over* $\mathsf{TailWindowFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TailWindowFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by $T, W, D, R, S, H, C, P, N$. It has no coordinate for a completed filter quotient, selected limit, countable-choice schedule, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7119.2 (Tail-window filter NameCert obligations)**.** *For every accepted* $\mathsf{TailWindowFilter}^{\uparrow}$ *carrier*

$$F = (T, W, D, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TailWindowFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite tail-index row, cofinal finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *source row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same rows; cofinality exactness for* W *over* T*; Real-completion handoff through* S, D, R*; and ledger nonescape for completed filter quotients, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7119.1. The finite tail-index, cofinal window, dyadic tolerance, regular rational readback, stream source, and structural rows are all displayed coordinates.

Classifier stability is componentwise on those coordinates. Cofinality exactness is read from W: each tail request named in T receives a displayed finite window beyond the request, with the dyadic ledger D recording the relevant comparison.

The handoff rows S, D, R, H, C, P, N preserve and replay the same finite route. Since no completed filter quotient, selected limit, choice schedule, or ambient completeness row occurs in the carrier, the NameCert surface has exactly the listed obligations. □

Theorem 7119.3 (Tail-window filter cofinality)**.** *Let* $F = (T, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{TailWindowFilter}^{\uparrow}$ *carrier. For every tail request displayed in* T*, the row* W *supplies a finite window beyond that request, and the dyadic ledger* D *records the comparison that makes the window cofinal. This cofinality is stable under the componentwise* hsame *transport row* H *and replay through the displayed* Cont *row* C*.*

Proof. Read a tail request from T. The cofinality row W is part of the carrier, so it supplies the finite window assigned to that request before any regular rational readback is opened.

The comparison that the window lies beyond the request is recorded in D. The stream source S names the finite observations used by that comparison, and R is read only after those rows have been displayed.

Transport through H moves the request, window, dyadic comparison, stream row, and readback row componentwise. Replay through C follows the same displayed route, so cofinality remains a finite tail-window property. □

Theorem 7119.4 (Tail-window filter Real completion handoff)**.** *Every Real-completion consumer admitted by an accepted* $\mathsf{TailWindowFilter}^{\uparrow}$ *packet receives only the displayed finite tail row* T*, cofinal window row* W*,* $\mathsf{StreamName}^{\uparrow}$ *source row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and seal rows* H, C, P, N*. It receives no completed filter quotient, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7119.2. A completion-facing consumer can inspect exactly the carrier rows listed there, and those rows expose a finite tail request, a cofinal finite window, a stream schedule, a dyadic comparison ledger, and regular rational readback.

The transport and replay rows do not add analytic content. They preserve the same displayed route from tail request and window through stream and dyadic data to the regular rational readback.

Thus the Real-completion handoff is the finite row package $T, W, S, R, D, H, C, P, N$. The carrier inventory excludes completed filter quotients, selected limits, countable choice, and ambient real completeness, so none of them is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailWindowFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{TailWindowFilter}^{\uparrow}$ packet is generated from a finite tail-index row, a cofinal finite-window row, a DyadicRatCore tolerance row, RegSeqRat readback, a StreamName source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailWindowFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7119.1, Definition 7119.2, Definition 7119.3, and Definition 7119.4 seed the finite cofinal tail-window route for Real completeness through StreamName, RegSeqRat, DyadicRatCore, and Real sealing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailWindowFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed filter quotients, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, cofinality exactness, Real-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.TailWindowFilterUp.

Chapter 7120

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ packet records the finite-window theorem surface by which two regular Cauchy rational names are read as the same real only after a displayed modulus window has compared their rational tails. It sits after the dyadic arithmetic core of chapter 291, the stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite L10 uniqueness route for Real and Completion consumers; it is not quotient stream equality, a selected canonical representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7120.1 (Regular Cauchy uniqueness-window carrier)**.** *A* $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, W, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, exposing dyadic addition, subtraction, absolute-value, and comparison data used inside the finite window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, recording the two inspected rational streams and the finite indices requested by the consumer.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, exposing the displayed Cauchy moduli and rational readbacks for the two names.*
4. *W is the uniqueness-window row, listing the finite tail bound, the two tail indices, and the dyadic tolerance at which the names are compared.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording only the terminal real-name read permitted after the displayed comparison has passed through D, S, R, W.*
6. *H records componentwise* hsame *transport for D, S, R, W, E, C, P, N.*
7. *C records displayed* Cont *replay from a Real or Completion uniqueness request through*

$$D, S, R \longmapsto W \longmapsto E, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, W, E, H, C, P, N$. It has no coordinate for quotient streams, hidden representatives, unrestricted equality of host reals, countable choice, or ambient completeness.

Theorem 7120.2 (Regular Cauchy uniqueness-window NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ *carrier $U = (D, S, R, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniquenessW}}$ *surface consists exactly of dyadic arithmetic admission through D, stream-name exposure through S, regular rational readback through R, uniqueness-window comparison through W, terminal real-name sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $D, S, R, W, E, H, C, P, N$ are the entire inventory exposed to a Real or Completion consumer.

The arithmetic, stream, and regularity data are read before any equality-facing conclusion is available. The row D supplies the dyadic comparisons, S supplies the inspected stream windows, and R supplies the regular Cauchy moduli.

The uniqueness step is finite. The window row W records the tail indices and tolerance that allow the two regular names to be compared at the requested precision, and the real-name seal E is reached only after that row is displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of these rows contains quotient stream equality, a hidden canonical representative, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7120.3 (Regular Cauchy uniqueness-window route)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ *packet must factor its uniqueness request through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto W \longmapsto \mathsf{Real}^{\uparrow}, H, C, P, N.$$

The consumer receives dyadic arithmetic, two finite stream windows, regular Cauchy readback, a displayed uniqueness window, a terminal real-name seal, componentwise transport, replay, provenance, and local naming. It receives no quotient stream equality, no selected representative, no countable-choice branch, and no ambient completeness principle.

Proof. Begin with a uniqueness-facing read of two regular Cauchy names. The source streams are visible only through S, and their rational arithmetic is visible only through D.

Use the regularity row R to obtain the displayed tail moduli. The packet then exposes W, which stores the finite tolerance and the two tail indices used by the comparison.

After this finite comparison, the route may pass to the real-name seal E. Transport, replay, provenance, and local naming preserve the same route through H, C, P, N.

Thus the exported uniqueness read is exactly the finite window named in the statement. Any read requiring quotient streams, selected representatives, countable choice, or ambient completeness would require a carrier coordinate that U does not contain. □

Falsifiable prediction: Every public $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ uniqueness read factors through dyadic arithmetic, two finite stream windows, regular rational readback, a displayed tail-comparison window, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming. A public read that exports quotient stream equality,

a selected representative, countable choice, or ambient completeness from this packet refutes the carrier boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic arithmetic, two inspected stream windows, regular Cauchy moduli, a uniqueness-window comparison, real-name sealing, transport, replay, provenance, and local naming while refusing quotient equality, selected representatives, choice, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7120.1, Definition 7120.2, and Definition 7120.3 seed the finite uniqueness-window theorem surface over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected canonical representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic arithmetic admission, stream-window exposure, regular Cauchy readback, tail-comparison exactness, real-name sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyUniquenessWindowUp.

Constructive story: A $\mathsf{RegularCauchyUniquenessWindow}^{\uparrow}$ packet is generated from dyadic arithmetic rows, two stream-name windows, regular rational readbacks, a uniqueness-window comparison row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7121

A Concrete Naming Certificate for HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$

The HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ packet realizes the Poincare disk geodesic-midpoint reading of chapter 9463 as finite phase, metric, boundary-transport, and midpoint rows. It sits after the disk metric of chapter 6209, the hyperbolic metric row of chapter 3474, the boundary-transport surface of chapter 6267, and the geodesic-flow packet of chapter 3856. The packet names a finite midpoint certificate for phase transport along one displayed geodesic chord; it does not select a completed geodesic space, identify physical observer motion, or export a host Riemannian midpoint theorem.

Definition 7121.1 (Hyperbolic phase geodesic-midpoint carrier)**.** *A* HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, B, F, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare disk row, storing the two displayed interior endpoints and the inspected disk coordinate window.*
2. *M is the hyperbolic metric row, exposing only the finite metric comparisons used to certify the midpoint read.*
3. *B is the Poincare disk boundary-transport row, recording the boundary endpoints that determine the displayed chord without selecting an ideal-boundary representative outside the packet.*
4. *F is the hyperbolic geodesic-flow row, storing the finite chord parameter window and the reversible transport read along that row.*
5. *Q is the midpoint row, recording the displayed midpoint candidate, the two equal-distance checks, and the dyadic tolerance used by the comparison.*
6. *T is the phase-transport row, preserving the phase label attached to the chord before and after the midpoint read.*
7. *H records componentwise* hsame *transport for $D, M, B, F, Q, T, C, P, N$.*
8. *C records displayed* Cont *replay from a geodesic-midpoint request through*

$$D, M, B, F \longmapsto Q, T, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B, F, Q, T, H, C, P, N$. It has no coordinate for completed hyperbolic geometry, selected boundary representatives, global geodesic existence, physical observer trajectories, or host midpoint uniqueness.

Theorem 7121.2 (Hyperbolic phase geodesic-midpoint NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}$ *carrier $G = (D, M, B, F, Q, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhas}}$ *surface consists exactly of disk-coordinate admission through D, hyperbolic metric readback through M, boundary-transport exposure through B, geodesic-flow windowing through F, midpoint comparison through Q, phase-transport preservation through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $D, M, B, F, Q, T, H, C, P, N$ are the entire public inventory of the packet.

The disk and metric rows are read before any midpoint claim is visible. The rows D and M expose only displayed disk coordinates and finite hyperbolic metric comparisons.

The boundary and geodesic data are then routed through B and F. These rows determine the inspected chord and its finite transport window without selecting an ideal-boundary representative outside the packet.

The midpoint row Q stores the candidate and the two equal-distance checks, while T preserves the phase label attached to the chord. The rows H, C, P, N transport, replay, source, and name that same finite route.

Since none of the carrier rows contains completed hyperbolic geometry, global geodesic existence, physical observer motion, or host midpoint uniqueness, the displayed rows exhaust the local NameCert obligations. □

Theorem 7121.3 (Hyperbolic phase geodesic-midpoint route)**.** *Every phase-geodesic consumer that reads an accepted* $\mathsf{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}$ *packet must factor its midpoint request through*

$$\mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{PoincareDiskBoundaryTransport}^{\uparrow},\ \mathsf{HyperbolicGeodesicFlow}^{\uparrow} \longmapsto Q, T, H, C,$$

The consumer receives displayed disk coordinates, finite hyperbolic metric comparisons, boundary transport, a geodesic-flow window, midpoint checks, phase transport, componentwise transport, replay, provenance, and local naming. It receives no completed geodesic space, no selected boundary representative, no physical observer trajectory, and no host midpoint theorem.

Proof. Begin with a phase-geodesic midpoint request. The disk row D and metric row M provide the only coordinate and distance reads available to the request.

Use the boundary and flow rows B, F to expose the displayed chord. This determines the finite transport window but not a completed geodesic space or a selected ideal-boundary object.

Read the midpoint and phase rows Q, T. The candidate midpoint is accepted only through the recorded equal-distance checks, and the phase label is preserved through the same finite chord route.

Finally H, C, P, N transport, replay, source, and name the route. Thus the exported midpoint surface is the finite packet named in the statement, with no additional host geometry or physical-observer content. □

Falsifiable prediction: Every accepted HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ midpoint read factors through displayed Poincare disk coordinates, hyperbolic metric comparisons, boundary transport, a geodesic-flow window, a midpoint row, phase transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming. A public read that exports a midpoint without one of those rows refutes the carrier boundary.

Independence witness: hyperbolic-metric, poincare-disk-boundary-transport, hyperbolic-geodesic-flow, real, metric: HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk coordinates, metric readback, boundary transport, geodesic-flow windowing, midpoint equal-distance checks, phase transport, replay, provenance, and local naming.

The Lean TasteGate surface for this packet must provide explicit Nontrivial HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ and FieldFaithful HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ instances before a checked carrier-alignment theorem can witness the public route.

Closure status for NameCert$_{\text{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}}$.

Constructive story: A HyperbolicPhaseGeodesicMidpoint$^{\uparrow}$ packet is generated from Poincare disk coordinates, hyperbolic metric comparisons, boundary transport, geodesic-flow windows, midpoint equal-distance checks, phase transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}}$, seed)

Scope closed: Definition 7121.1, Definition 7121.2, and Definition 7121.3 seed the finite phase-geodesic midpoint packet over disk, metric, boundary-transport, geodesic-flow, midpoint, phase, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicPhaseGeodesicMidpoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic geometry, selected ideal-boundary representatives, global geodesic existence, physical observer motion, host midpoint uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk admission, metric readback, boundary-transport exposure, geodesic-flow windowing, midpoint exactness, phase preservation, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseGeodesicMidpointUp.

Chapter 7122

A Concrete Naming Certificate for $\mathsf{BishopModulusCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopModulusCompletion}^{\uparrow}$ packet records the finite BEDC surface by which a modulus-controlled regular sequence enters the constructive completion route and reaches the terminal real-name seal. It sits after the finite observation schedule of chapter 126, the regular rational source of chapter 301, the dyadic tolerance ledger of chapter 291, the completion admission criterion of chapter 1476, and the terminal Real packet of chapter 38. The packet names a modulus-governed handoff as finite NameCert data; it does not select a quotient limit, compare host real equality, invoke countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7122.1 (Bishop modulus completion carrier)**.** *A* $\mathsf{BishopModulusCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, M, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window ledger naming the observations inspected by the completion request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational values are read only through the displayed stream windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same requests, carrying the dyadic scales used by the modulus schedule.*
4. *M is the modulus schedule row, assigning each requested tolerance to a displayed tail bound over S, R, D.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, exposed only after the schedule has met the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *admission surface.*
6. *H records componentwise* hsame *transport for the window, source, tolerance, modulus, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a dyadic tolerance request through the same rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopModulusCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopModulusCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, R, D, M, L, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, arbitrary Cauchy completion, or ambient completeness of the Real carrier.

Theorem 7122.2 (Bishop modulus completion NameCert obligations)**.** *For every accepted* BishopModulusCompleti *carrier $K = (S, R, D, M, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopModulusCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName windows, RegSeqRat source, DyadicRatCore tolerances, modulus schedule, Real seal handoff, classifier transport, displayed replay, provenance, and local naming; modulus coverage requiring every tolerance request to read M after S, R, D; regular-tail compatibility requiring the scheduled tail to preserve the same RegSeqRat readback; Real seal handoff through the completion criterion; componentwise transport and replay through H, C; provenance through P; and nonescape excluding quotient limits, host real equality, countable choice, arbitrary completion, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7122.1. The rows $S, R, D, M, L, H, C, P, N$ are exactly the visible StreamName, RegSeqRat, DyadicRatCore, modulus, Real seal, transport, replay, provenance, and naming coordinates.

The modulus coverage is read from M only after the window, source, and tolerance rows are displayed. Thus each tolerance request is tied to the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ source, and $\mathsf{DyadicRatCore}^{\uparrow}$ scale.

The regular-tail compatibility and Real seal handoff are local to the displayed route. The scheduled tail preserves the source readback, and the seal L is exposed only through the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ admission surface.

Finally H, C, P, N transport, replay, package, and name the same finite rows. Since the carrier has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, arbitrary completion, or ambient completeness, the listed obligations exhaust the NameCert surface. □

Theorem 7122.3 (Bishop modulus completion Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopModulusCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyCompletionCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The modulus row M binds the displayed windows, regular rational source, and dyadic tolerances before the Real seal L can be read.

Proof. Start with an accepted carrier $K = (S, R, D, M, L, H, C, P, N)$. The consumer first reads the displayed window row S, the regular rational source row R, and the dyadic tolerance row D.

Next the modulus row M supplies the tail bound for the requested tolerance over those same rows. This prevents the read from replacing the displayed regular source by a host sequence or by a quotient representative.

The seal row L is downstream of the criterion-facing admission surface, and H, C, P, N preserve the same finite data during transport, replay, provenance, and naming. Therefore the Real-facing consumer receives only the displayed modulus-controlled completion handoff. □

Falsifiable prediction: Every accepted $\mathsf{BishopModulusCompletion}^{\uparrow}$ consumer reaches the Real seal only through displayed StreamName windows, a RegSeqRat source, DyadicRatCore tolerances, a modulus schedule, completion-criterion admission, componentwise transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, dyadicratcore, cauchycompletioncriterion, real: none is bijective with $\mathsf{BishopModulusCompletion}^{\uparrow}$ because this packet alone binds finite windows, regular rational source data, dyadic tolerances, and a modulus schedule before exposing the Real seal handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopModulusCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopModulusCompletion}^{\uparrow}$ packet is generated from StreamName windows, a RegSeqRat source, DyadicRatCore tolerance rows, a modulus schedule, RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopModulusCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7122.1, Definition 7122.2, and Definition 7122.3 seed the modulus-controlled completion handoff over StreamName, RegSeqRat, DyadicRatCore, CauchyCompletionCriterion, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopModulusCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, arbitrary completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus coverage, regular-tail compatibility, Real seal handoff, classifier transport, replay, provenance, and nonescape for BEDC.Derived.BishopModulusCompletionUp.

Chapter 7123

A Concrete Naming Certificate for BishopDyadicLocatedReal$^{\uparrow}$

Following Bishop and Bridges constructive analysis, located real data are read through explicit rational approximations, apartness tests, and finite tolerance responses rather than through quotient reals, selected limits, countable choice, or an ambient completeness theorem. The BishopDyadicLocatedReal$^{\uparrow}$ packet records the dyadic located-real route as finite NameCert data over the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the rational interval locator of chapter 4202, the locatedness surface of **??**, the terminal seal of chapter 38, and the threshold selector of chapter 6642.

Definition 7123.1 (Bishop dyadic located-real carrier)**.** *A* BishopDyadicLocatedReal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, Q, I, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row naming the positive dyadic radius and comparison budget requested by the located read;*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row opened for that tolerance;*
3. *Q is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows;*
4. *I is the* RationalIntervalLocator$^{\uparrow}$ *row exposing lower and upper rational brackets for the same dyadic request;*
5. *L is the* LocatedReal$^{\uparrow}$ *row deciding the displayed overlap or separation test between the readback and the rational interval at the named tolerance;*
6. *E is the* Real$^{\uparrow}$ *seal row exposed only after D, W, Q, I, L have been displayed;*
7. *H records componentwise* hsame *transport for the dyadic tolerance, stream-window, readback, interval-locator, locatedness, Real seal, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from dyadic tolerance through finite windows, regular rational readback, rational interval location, locatedness, Real seal, and the structural rows;*
9. *P is the* Pkg *provenance over* BishopDyadicLocatedReal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RationalIntervalLocator$^{\uparrow}$, LocatedReal$^{\uparrow}$, Real$^{\uparrow}$, LocatedCauchyModulusSelector$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedReal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a selected host limit, countable choice, an unlocated Cauchy witness, an ambient completeness theorem, or a Real seal read that bypasses the dyadic locatedness data.

Theorem 7123.2 (Bishop dyadic located-real NameCert obligations)**.** *For every accepted* BishopDyadicLocatedReal *carrier*

$$B = (D, W, Q, I, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance, finite stream-window, regular rational readback, rational interval locator, locatedness, Real seal, transport, replay, provenance, and local naming rows; dyadic tolerance exactness; window and readback compatibility; rational bracket exposure; locatedness exactness at the displayed tolerance; Real-seal nonescape; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; and local naming for the same finite packet.*

Proof. Carrier admission is projection from Definition 7123.1. The accepted packet exposes exactly the rows $D, W, Q, I, L, E, H, C, P, N$.

The analytic route is finite. The dyadic tolerance row D opens the stream window W, the readback row Q records the regular rational data read from that window, and the locator row I supplies the rational bracket for the same request.

Locatedness is then local to the displayed tolerance. The row L decides the overlap or separation test after D, W, Q, I are visible, and the Real seal E is downstream of that locatedness row.

Finally H, C, P, N transport, replay, source, and name the same data. Since the carrier contains no quotient equality, selected host limit, choice schedule, unlocated witness, ambient completion theorem, or bypass seal coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 7123.3 (Bishop dyadic located-real Real handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{BishopDyadicLocatedReal}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RationalIntervalLocator}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For a carrier $B = (D, W, Q, I, L, E, H, C, P, N)$, the consumer receives the dyadic tolerance request, finite window, regular rational readback, rational interval locator, locatedness row, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. It receives no quotient real, selected limit, countable-choice schedule, unlocated Cauchy witness, or ambient completion datum.*

Proof. Start with the NameCert obligations of Definition 7123.2. They force any Real-facing read to stay on the displayed carrier rows.

Read the finite analytic rows in order. The dyadic request D opens W, the regular readback Q is obtained from the window, and the rational interval locator I brackets the same request.

The locatedness row L is then read before the seal. By the carrier definition, E is exposed only after D, W, Q, I, L have been displayed, and the structural rows H, C, P, N preserve that route for downstream consumers.

Thus the Real handoff is a dyadic located-real certificate, not a quotient real construction, a selected host limit, a countable-choice route, an unlocated Cauchy witness, or an ambient completion theorem. □

Falsifiable prediction: Every accepted $\mathsf{BishopDyadicLocatedReal}^{\uparrow}$ Real-facing read exposes one DyadicRatCore tolerance row, one StreamName finite-window row, one RegSeqRat readback row, one RationalIntervalLocator bracket row, one LocatedReal locatedness row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the Real seal is public.

Independence witness: dyadicratcore, streamname, regseqrat, rational-interval-locator, locatedreal, real, located-cauchy-modulus-selector: $\mathsf{BishopDyadicLocatedReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance data, finite stream windows, regular rational readback, rational brackets, locatedness at the displayed tolerance, Real seal handoff, transport, replay, provenance, and local naming as one finite located-real certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDyadicLocatedReal}^{\uparrow}$ packet is generated from dyadic tolerance requests, finite stream windows, regular rational readback, rational interval locator rows, locatedness rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7123.1, Definition 7123.2, and Definition 7123.3 seed the finite Bishop dyadic located-real route over DyadicRatCore, StreamName, RegSeqRat, RationalIntervalLocator, LocatedReal, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, selected host limits, countable choice, unlocated Cauchy witnesses, ambient completion, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.BishopDyadicLocatedReal

Chapter 7124

A Concrete Naming Certificate for $\mathsf{CauchyBicategory}^{\uparrow}$

Following Lawvere on enriched metric completion and Street on bicategorical completion, the $\mathsf{CauchyBicategory}^{\uparrow}$ packet records the finite BEDC bookkeeping by which Cauchy-completion routes compose without turning into quotient representatives or chosen limits. It sits over the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the metric-completion carrier of chapter 1506. Its role is to name the identity, composition, and 2-cell rows used by L10 completion consumers, not to export a completed object in a host category.

Definition 7124.1 (Cauchy bicategory carrier)**.** *A* $\mathsf{CauchyBicategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, M, I, O, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by every displayed Cauchy comparison.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row serving the inspected prefixes.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-request row read only after D, S, R are visible.*
5. *I is the identity-route ledger for an object row and its finite Cauchy window.*
6. *O is the composition-route ledger for two composable completion arrows.*
7. *T is the 2-cell ledger comparing two displayed completion routes through the same tolerance, window, and readback rows.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from a completion-composition request through the same rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyBicategory}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBicategory}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, M, I, O, T, H, C, P, N$. It has no coordinate for quotient completion equality, selected limit representatives, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host category equivalence, or a global choice of Cauchy representatives.*

Theorem 7124.2 (Cauchy bicategory NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBicategory}^{\uparrow}$ *carrier* $K = (D, S, R, M, I, O, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyBicategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, finite stream windows, regular rational readback, metric-completion request, identity ledger, composition ledger, 2-cell comparison ledger, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7124.1. The rows $D, S, R, M, I, O, T, H, C, P, N$ are exactly the visible identity, composition, and comparison surface.

The completion-facing rows are finite. The dyadic tolerance row D, stream windows S, and regular rational readback R must be displayed before the metric-completion request M can be consumed, so no quotient representative or selected limit is introduced at the entrance.

The identity row I, composition row O, and 2-cell row T are themselves finite BHist routes. They name the identity, composite, and comparison ledgers by replaying the same tolerance-window-readback surface, rather than by appealing to an ambient bicategory of completed objects.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient completion equality, selected limits, host category equivalence, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7124.3 (Cauchy bicategory completion-composition handoff)**.** *Every completion-composition consumer of an accepted* $\mathsf{CauchyBicategory}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ I,\ O,\ T,\ H,\ C,\ P,\ N.$$

The exported handoff is the displayed identity, composition, and 2-cell BHist *ledger. It exports no quotient representative, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, host bicategorical equivalence, or choice-selected Cauchy object.*

Proof. Start with the NameCert obligations of Definition 7124.2. They force every accepted consumer to read the dyadic tolerance, stream-window, regular rational, metric-completion, identity, composition, and 2-cell rows before the structural rows are consulted.

The identity ledger I is tied to the same finite Cauchy window surface as the source object. The composition ledger O is tied to two displayed completion-arrow routes, and the 2-cell ledger T compares only those displayed routes.

The structural rows H, C, P, N preserve that route under componentwise transport, replay, provenance, and local naming. They do not add a host completed category, choose representatives, quotient streams, or import ambient completeness.

Therefore the completion-composition handoff is exactly the finite BHist route shown above. □

Falsifiable prediction: An accepted $\mathsf{CauchyBicategory}^{\uparrow}$ consumer is refuted if it reaches an identity, composition, or 2-cell completion route without displaying the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and $\mathsf{MetricCompletion}^{\uparrow}$ request row.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletion: no listed sibling is bijective to $\mathsf{CauchyBicategory}^{\uparrow}$ because this packet jointly binds identity ledgers, composition ledgers, 2-cell comparison ledgers, tolerance rows, finite windows, regular readback, completion requests, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBicategory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyBicategory}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a MetricCompletion request, identity ledgers, composition ledgers, 2-cell comparison ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyBicategory}^{\uparrow}}$, seed)

Scope closed: Definition 7124.1, Definition 7124.2, and Definition 7124.3 seed the finite Cauchy-bicategory bookkeeping surface over displayed tolerance, window, readback, completion, identity, composition, 2-cell, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyBicategory}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CauchyBicategoryUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limit representatives, ambient Real$^{\uparrow}$ completeness, host bicategorical equivalence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, identity-route exactness, composition-route exactness, 2-cell comparison stability, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyBicategoryUp

Chapter 7125

A Concrete Naming Certificate for $\mathsf{CantorNormalForm}^{\uparrow}$

Following Cantor on ordinal normal forms and Bishop–Bridges constructive ordinal scheduling, this packet records finite descending normal-form rows used by Real and Completion proof schedules. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window service of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the Cauchy schedule surfaces of chapter 292, chapter 1968, chapter 2096, and chapter 7122. The packet names a finite descent schedule only; it does not import quotient ordinals, transfinite choice, ordinal comparison as a host primitive, or a completed Real construction.

Definition 7125.1 (Cantor normal form carrier)**.** *A* $\mathsf{CantorNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (E, K, D, W, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the exponent-descent row, listing a finite strictly descending sequence of ordinal-style exponent tags used only as schedule indices.*
2. *K is the finite coefficient-window row, attaching a positive finite coefficient budget to each exponent tag.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row consumed by Real and Completion estimates at each displayed descent stage.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite stream observations opened by the same descent and coefficient windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to D and W.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational readback has been displayed.*
7. *S is the schedule handoff row for* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$, *and* $\mathsf{BishopModulusCompletio}$ *consumers.*
8. *H records componentwise* hsame *transport for descent, coefficient, dyadic-budget, stream-window, regular-readback, real-seal, schedule, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay for Real-facing and Completion-facing schedule consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$, $\mathsf{BishopModulusCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, K, D, W, Q, R, S, H, C, P, N$. It contains no coordinate for quotient ordinal equality, infinite ordinal induction, selected limit ordinals, trichotomy over all ordinals, or a host completion operator.

Theorem 7125.2 (Cantor normal form obligation surface)**.** *For every accepted* $\mathsf{CantorNormalForm}^{\uparrow}$ *carrier*

$$O = (E, K, D, W, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorNormalForm}^{\uparrow}}$ *surface consists exactly of exponent-descent exposure through E, coefficient-window exposure through K, dyadic tolerance exposure through D, stream-window exposure through W, regular rational readback through Q, terminal Real sealing through R, schedule handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 7125.1. The rows $E, K, D, W, Q, R, S, H, C, P, N$ are exactly the displayed schedule inventory.

Read the descent rows first. The exponent tags in E and coefficient windows in K are finite schedule data, so each later budget request is attached to a named position in the normal-form row.

Next read the analytic rows. The dyadic tolerance ledger D and stream windows W feed the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and only then can the terminal $\mathsf{Real}^{\uparrow}$ seal R be consumed by a Real or Completion schedule.

Finally use S, H, C, P, N to transport, replay, source, and name the same packet. Since the carrier has no quotient ordinal, infinite induction, selected-limit, trichotomy, or host-completion coordinate, the listed rows exhaust the obligation surface. □

Theorem 7125.3 (Cantor normal form Real completion route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CantorNormalForm}^{\uparrow}$ *packet must factor through the finite route*

$$E, K \longmapsto D, W \longmapsto Q \longmapsto R \longmapsto S, H, C, P, N.$$

The route provides ordinal-style descent as displayed schedule data for Real and Completion proofs; it does not provide quotient ordinal equality, transfinite choice, a selected limit ordinal, or an ambient completion theorem.

Proof. Begin with the finite normal-form prefix. Rows E and K expose the descending exponent tags and coefficient windows before any tolerance or stream request can be opened.

The dyadic and stream rows then determine the analytic read. A consumer reads D and W together, obtains the regular rational readback Q, and reaches the Real seal R only after that readback is visible.

The schedule handoff S is therefore downstream of the finite descent and analytic rows. Its Completion-facing use may cite the Cauchy-modulus siblings named in the provenance row, but only through the same displayed packet.

Transport, replay, provenance, and local naming preserve the route. The exclusions follow from carrier inspection: no coordinate names quotient ordinals, transfinite choice, selected limit ordinals, or host completion. □

Falsifiable prediction: Every accepted $\mathsf{CantorNormalForm}^{\uparrow}$ schedule read displays finite exponent descent, coefficient windows, dyadic tolerance budgets, StreamName windows, RegSeqRat readback, and the Real seal before any Completion-facing handoff is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchymodulus, bishop-modulus-completion: none is bijective to $\mathsf{CantorNormalForm}^{\uparrow}$ because this packet jointly binds finite normal-form descent to dyadic budgets, stream windows, regular readback, Real sealing, schedule handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorNormalForm}^{\uparrow}$ packet is generated from finite exponent-descent rows, coefficient windows, dyadic tolerance budgets, StreamName windows, RegSeqRat readback, Real sealing, schedule handoff rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7125.1, Definition 7125.2, and Definition 7125.3 seed the finite normal-form descent route consumed by Real and Completion schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ordinal equality, infinite ordinal induction, transfinite choice, selected limit ordinals, host completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, descent and coefficient exposure, dyadic and stream readback exactness, Real seal nonescape, schedule handoff, transport, replay, provenance, and local naming for BEDC.Derived.CantorNormalFormUp.

Chapter 7126

A Concrete Naming Certificate for $\mathsf{DistinctionCertificate}^{\uparrow}$

The $\mathsf{DistinctionCertificate}^{\uparrow}$ packet realizes the distinction-to-NameCert certification layer as a finite boundary object: a candidate distinction is accepted only when source, pattern, classifier, stability, ledger, transport, replay, provenance, and local naming rows are displayed together. It sits beside the primitive mark distinction ledger of chapter 1244, the observer distinction trace of chapter 882, the halting distinction trace of chapter 492, the certified event-flow packet of chapter 7098, and the supply taxonomy of chapter 7078. Those siblings supply concrete distinction sources or audit routes; this packet certifies the boundary at which a distinction becomes a NameCert object.

Definition 7126.1 (DistinctionCertificate carrier)**.** *A* $\mathsf{DistinctionCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, P_0, K, \Sigma, L, H, C, R, N)$$

with the following displayed rows:

1. *S is the source row naming the generated marks, histories, traces, event flows, or supply records whose distinction is being inspected.*
2. *P_0 is the pattern row, recording the finite feature, trace, boundary, or certificate pattern proposed as evidence for the distinction.*
3. *K is the candidate classifier row that compares source records through the displayed pattern.*
4. *Σ is the stability row, recording transport of the classifier under the admitted* hsame *movements.*
5. *L is the ledger row, naming hidden material, refusal cases, and not-claimed readings.*
6. *H records componentwise* hsame *transport for source, pattern, classifier, stability, ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for distinction-certification consumers.*
8. *R is the* Pkg *provenance over* $\mathsf{DistinctionCertificate}^{\uparrow}$, $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$, $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{CertifiedEventFlow}^{\uparrow}$, $\mathsf{SupplyTaxonomy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DistinctionCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for raw distinction objecthood, uncertified semantic assignment, hidden classifier shift, global explanation, or verifier acceptance detached from the five displayed certificate rows.

Theorem 7126.2 (DistinctionCertificate NameCert boundary)**.** *Every accepted* $\mathsf{DistinctionCertificate}^{\uparrow}$ *packet certifies a candidate distinction as a NameCert boundary only through*

$$S, P_0, K, \Sigma, L, H, C, R, N.$$

The source, pattern, classifier, stability, and ledger rows must all be present before the local naming row can be consumed.

Proof. Project the carrier of Definition 7126.1. Rows S, P_0, K, Σ, L are the five certificate rows, and H, C, R, N only transport, replay, source, and name that same surface.

The candidate classifier row K is not read by itself. It is tied to the source row S and pattern row P_0, then stabilized through Σ and audited through L.

Only after those rows are visible can the local $\mathsf{NameCert}$ row N be consumed. Thus the accepted packet is a certificate boundary, not a raw distinction or a bare classifier. □

Theorem 7126.3 (DistinctionCertificate sibling separation)**.** *The* $\mathsf{DistinctionCertificate}^{\uparrow}$ *packet is not bijective to* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$*,* $\mathsf{ObserverDistinctionTrace}^{\uparrow}$*,* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$*,* $\mathsf{CertifiedEventFlow}^{\uparrow}$*, or* $\mathsf{SupplyTaxonomy}^{\uparrow}$*. Each listed sibling supplies a distinction source, trace, event-flow, or audit route, while* $\mathsf{DistinctionCertificate}^{\uparrow}$ *additionally binds the candidate classifier to the five-field NameCert boundary* S, P_0, K, Σ, L *with transport, replay, provenance, and local naming.*

Proof. Compare the displayed rows. Primitive distinction, observer trace, halting trace, certified event flow, and supply taxonomy each provide a concrete source surface or audit path for distinctions.

The present packet has a different role. It requires the source row, pattern row, classifier row, stability row, and ledger row to be visible together before certification is licensed.

Therefore none of the listed siblings can replace $\mathsf{DistinctionCertificate}^{\uparrow}$ while preserving all fields of Definition 7126.1. A consumer that needs certification must read the full packet. □

Theorem 7126.4 (DistinctionCertificate TasteGate requirements)**.** *The Lean TasteGate surface for* $\mathsf{DistinctionCertificate}^{\uparrow}$ *must provide explicit* $\mathsf{Nontrivial\ DistinctionCertificate}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ DistinctionCertificat}$ *instances. The nontriviality witness must separate a certified classifier packet from a bare candidate distinction, and the field-faithfulness witness must keep* $S, P_0, K, \Sigma, L, H, C, R, N$ *as distinct rows under transport and replay.*

Proof. Nontriviality follows from the intended separation between raw distinction material and a certified NameCert boundary. A bare candidate distinction lacks at least one of S, P_0, K, Σ, L and therefore cannot be the same accepted packet as the full carrier.

Field faithfulness is the preservation requirement for the displayed coordinates. If source, pattern, classifier, stability, ledger, transport, replay, provenance, or naming were collapsed, the packet would no longer expose the certificate boundary of Definition 7126.2.

Thus the TasteGate must preserve both the distinction between bare and certified objects and the componentwise rows of the accepted packet. □

Falsifiable prediction: A consumer that reads $\mathsf{DistinctionCertificate}^{\uparrow}$ without displaying source, pattern, classifier, stability, and ledger rows is reading a bare candidate distinction rather than a certified NameCert boundary.

Independence witness: primitive-distinction-ledger, observerdistinctiontrace, haltingdistinctiontrace, certified-event-flow, supply-taxonomy: none is bijective to $\mathsf{DistinctionCertificate}^{\uparrow}$ because this packet binds candidate classifier certification through source, pattern, classifier, stability, ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DistinctionCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{DistinctionCertificate}^{\uparrow}$ packet is generated from source rows, pattern rows, candidate classifier rows, stability rows, ledger rows, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistinctionCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7126.1, Definition 7126.2, Definition 7126.3, and Definition 7126.4 seed the finite NameCert boundary for certified distinctions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistinctionCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close raw distinction objecthood, uncertified semantic assignment, hidden classifier shift, global explanation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, five-field certificate exposure, sibling separation, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.DistinctionCertificateUp.

Chapter 7127

A Concrete Naming Certificate for $\mathsf{TaoMetastableCauchy}^{\uparrow}$

Following Tao's metastability/no-counterexample interpretation of Cauchy convergence, the $\mathsf{TaoMetastableCauchy}^{\uparrow}$ packet records a finite control-window answer for a Cauchy read before any completed limit, quotient stream, or countable-choice selector is available. It sits on the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing boundary of chapter 1639. The point of the packet is not to prove ambient $\mathsf{Real}^{\uparrow}$ completeness; it is to make the finite Cauchy window demanded by a control function into named BEDC data.

Definition 7127.1 (Tao metastable Cauchy carrier)**.** *A* $\mathsf{TaoMetastableCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (q, F, W, D, S, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *q is the dyadic precision request, read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
2. *F is the finite control row, naming the inspected control-function values only through displayed requests.*
3. *W is the metastable window row, displaying a bound n and the finite interval from n to $F(n)$.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger used by every comparison inside the window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row opened for the named stream positions in W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for the observed values.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after q, F, W, D, S, R have been displayed.*
8. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, recording that the window is completion-facing without selecting a completed limit.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from the precision request and control row through the finite window, dyadic ledger, stream observations, rational readback, terminal seal, and completion boundary.*
11. *P records* Pkg *provenance over* $\mathsf{TaoMetastableCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TaoMetastableCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by $q, F, W, D, S, R, E, B, H, C, P, N$. It has no coordinate for countable choice, quotient stream equality, selected limits, host real equality, trichotomy, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.TaoMetastableCauchyUp.TaoMetastableCauchyTasteGate single carrier alignment`

Theorem 7127.2 (Tao metastable Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{TaoMetastableCauchy}^{\uparrow}$ *carrier*

$$T = (q, F, W, D, S, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaoMetastableCauchy}^{\uparrow}}$ *surface consists exactly of finite carrier admission, precision request exposure, finite control-row exposure, metastable window extraction, dyadic tolerance accounting, StreamName observation, RegSeqRat readback, Real seal boundary, Bishop regular Cauchy completion boundary, componentwise transport, displayed replay, provenance, and local naming. Every public read factors through those rows before it can cite a metastable Cauchy answer.*

Lean4 ✓ `BEDC.Derived.TaoMetastableCauchyUp.TaoMetastableCauchyCarrier namecert obligations`

Proof. Project the carrier of Definition 7127.1. The accepted packet exposes exactly $q, F, W, D, S, R, E, B, H, C, P, N$ so no hidden limit selector or quotient stream coordinate is available.

The request, control, and window rows supply the finite no-counterexample data. The row q fixes the tolerance, F fixes the displayed control values, and W names the finite interval in which all comparisons are inspected.

The analytic rows remain on the same route. The dyadic ledger D, StreamName row S, and RegSeqRat row R provide the only comparison data before the terminal $\mathsf{Real}^{\uparrow}$ seal E and the $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ boundary row B are read.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no row for countable choice, quotient stream equality, selected limits, host real equality, trichotomy, or ambient completeness, the displayed rows exhaust the NameCert obligations. □

Theorem 7127.3 (Tao metastable Cauchy window extraction)**.** *Every accepted* $\mathsf{TaoMetastableCauchy}^{\uparrow}$ *public read extracts a finite window before any real-facing or completion-facing output is consumed: the precision request q and control row F determine the displayed window W, every comparison in W is read through the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the inspected values are read through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, R, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal E and* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row B remain after that finite route. The read exports no countable choice, quotient stream equality, selected limit, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TaoMetastableCauchyUp.TaoMetastableCauchyWindowExtraction`

Proof. Use Definition 7127.2. A public read must open q, F, W before it can reach the analytic rows, so the control request is answered by a displayed finite interval rather than by an infinite selection principle.

The comparison data are then forced through D, S, R. The dyadic ledger records the tolerance, StreamName supplies the finite observations named by the window, and RegSeqRat supplies the rational readbacks actually compared.

The endpoint rows are downstream only. The $\mathsf{Real}^{\uparrow}$ seal E and $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ boundary row B can be cited only after the finite window route has been displayed, while H, C, P, N preserve the same route as transport, replay, provenance, and local naming.

Therefore the extracted object is a finite metastable Cauchy window. Since no carrier row stores a selected limit, quotient equality, countable-choice witness, host equality, trichotomy, or ambient completeness principle, none of those objects is exported by the read. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaoMetastableCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{TaoMetastableCauchy}^{\uparrow}$ packet is generated from a dyadic precision request, finite control row, metastable window row, dyadic tolerance ledger, StreamName observations, RegSeqRat readback, Real seal boundary, Bishop regular Cauchy completion boundary, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaoMetastableCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7127.1, Definition 7127.2, and Definition 7127.3 seed the choice-free metastable Cauchy window route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaoMetastableCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient stream equality, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite control-window extraction, dyadic tolerance accounting, StreamName observation, RegSeqRat readback, Real seal boundary, Bishop regular Cauchy completion boundary, transport, replay, provenance, and local naming for BEDC.Derived.TaoMetastableCauchyUp.

Chapter 7128

A Concrete Naming Certificate for GaugeIntegralCriterion$^{\uparrow}$

Following Henstock, Kurzweil, and Bishop–Bridges on constructive integration, the GaugeIntegralCriterion$^{\uparrow}$ packet records the finite gauge-test surface used to read an integral claim through displayed gauges, tagged sums, dyadic tolerances, regular rational witnesses, and a terminal Real seal. It sits between the gauge integral route of chapter 1999, the tagged-sum surface of RiemannSumGauge$^{\uparrow}$, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal Real row of chapter 38. The packet is a finite BEDC criterion; it is not Lebesgue measure, a selected partition principle, open-cover compactness, countable choice, or ambient Real completeness.

Definition 7128.1 (Gauge integral criterion carrier)**.** *A* GaugeIntegralCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, S, W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* GaugeIntegral$^{\uparrow}$ *row naming the displayed integrand, interval, and gauge-integral claim.*
2. *S is the* RiemannSumGauge$^{\uparrow}$ *row exposing tagged Riemann sums for the same displayed claim.*
3. *W is the finite gauge-window row recording the inspected tagged subintervals and their local gauge bounds.*
4. *T is the tolerance row naming the positive request against which the sums are compared.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *row encoding the dyadic comparison budget for T and W.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row recording the regular rational sum comparisons visible inside the window.*
7. *E is the* Real$^{\uparrow}$ *seal row available only after I, S, W, T, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the gauge-integral, tagged-sum, window, tolerance, dyadic, readback, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the gauge-integral row through tagged sums, gauge windows, tolerance comparison, dyadic readback, Real seal, and the structural rows.*
10. *P is the* Pkg *provenance over* GaugeIntegralCriterion$^{\uparrow}$, GaugeIntegral$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaugeIntegralCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for Lebesgue measure completion, a selected partition, open-cover compactness, countable choice, quotient Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7128.2 (Gauge integral criterion NameCert obligations)**.** *For every accepted* $\mathsf{GaugeIntegralCriterion}^{\uparrow}$ *carrier*

$$G = (I, S, W, T, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GaugeIntegralCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the gauge-integral, tagged-sum, gauge-window, tolerance, dyadic-budget, regular-rational readback, Real-seal, transport, replay, provenance, and local naming rows; gauge-window exposure; tolerance exactness; tagged-sum comparison through the displayed dyadic and regular rational rows; Real-seal nonescape; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; and local naming for the same finite packet.*

Proof. Carrier admission is projection from Definition 7128.1. The accepted packet exposes exactly the rows $I, S, W, T, D, R, E, H, C, P, N$.

The analytic route is finite. The row I names the gauge-integral claim, S exposes the tagged sums, W records the inspected gauge window, and T fixes the tolerance request against which those sums are read.

The comparison rows are also displayed. The dyadic budget D and regular rational readback R are the only numerical comparison rows before the Real seal E is available. Thus a consumer cannot replace the criterion with Lebesgue measure completion, a selected partition, open-cover compactness, countable choice, quotient Real equality, or ambient Real completeness.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate exists in the carrier, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{GaugeIntegralCriterion}^{\uparrow}$ read exposes one GaugeIntegral row, one RiemannSumGauge row, one finite gauge-window row, one tolerance row, one DyadicRatCore comparison row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the criterion is consumed.

Independence witness: gauge-integral, riemann-sum-gauge, dyadicratcore, regseqrat, real: GaugeIntegralCriterio is not bijective to any listed sibling because it jointly carries gauge claims, tagged sums, finite windows, tolerance budgets, dyadic comparison, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{GaugeIntegralCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GaugeIntegralCriterion}^{\uparrow}$ packet is generated from a GaugeIntegral row, a RiemannSumGauge row, finite gauge windows, tolerance requests, DyadicRatCore comparison budgets, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GaugeIntegralCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 7128.1 and Definition 7128.2 seed the finite gauge criterion over GaugeIntegral, RiemannSumGauge, gauge-window, tolerance, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GaugeIntegralCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lebesgue measure completion, selected partitions, open-cover compactness, countable choice, quotient Real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge-window exposure, tolerance exactness, tagged-sum comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.GaugeIntegralCriterionUp.

Chapter 7129

Bishop interval modulus NameCert construction

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalModulus}^{\uparrow}$ packet records the finite modulus data by which a rational interval window controls a regular rational name and its Real-facing readback. It sits after the rational refinement surface of chapter 1432, the dyadic tolerance core of chapter 291, the stream-name windows of chapter 126, the regular rational source of chapter 301, and the terminal Real packet of chapter 38. The packet names interval-local modulus control as finite NameCert data; it does not choose an infinite refinement branch, select a completed real point, import a host interval subset, or claim ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7129.1 (BishopIntervalModulus carrier)**.** *A* $\mathsf{BishopIntervalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, r, W, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *row naming the parent-child rational interval containment and retained interval window.*
2. *r is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used as the requested interval radius or endpoint precision.*
3. *W is the modulus-window row assigning the tolerance r to a displayed retained rational interval window inside I.*
4. *S is the shared source row reading the relevant* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff data through the same finite window.*
5. *H records componentwise* hsame *transport for the interval-refinement row, dyadic tolerance, modulus window, source handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a tolerance request through I, r, W, S, H, P, N without changing the retained window.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalModulus}^{\uparrow}$*,* $\mathsf{RationalIntervalRefinement}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalModulus}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through I, r, W, S, H, C, P, N. It has no coordinate for an infinite interval tree, completed point selection, host subset relation, quotient real equality, countable choice, or standard bridge.

Theorem 7129.2 (BishopIntervalModulus window stability)**.** *Let $M = (I, r, W, S, H, C, P, N)$ be an accepted* $\mathsf{BishopIntervalModulus}^{\uparrow}$ *carrier. Replacing I, r, W, S by componentwise* hsame *rows and replaying the same tolerance request through C preserves the accepted modulus window: the consumer still reads the retained rational interval from W after the refinement row I and dyadic tolerance row r, and the stream, regular-rational, and Real-facing source data remain the row S.*

Proof. Project the carrier of Definition 7129.1. The only interval-refinement coordinate is I, the only dyadic tolerance coordinate is r, the only modulus-window coordinate is W, and the only source handoff coordinate is S.

The componentwise transport row H preserves those coordinates without adding a second window source. The displayed replay row C rereads the same tolerance request through I, r, W, S, H, P, N, so the retained rational interval window remains the one named by W.

Finally P and N only source and name the same finite packet. No row is available for an infinite branch, completed point, host subset, quotient equality, or choice schedule, so the accepted modulus window is stable under the displayed transport and replay. □

Theorem 7129.3 (BishopIntervalModulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalModulus}^{\uparrow}$ *carrier $M = (I, r, W, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the rational interval refinement, dyadic tolerance, modulus window, streamname, regular-rational, and Real-facing source rows; classifier transport through H; displayed continuation replay through C; provenance through P; local naming through N; and nonescape from infinite interval trees, completed point selection, host interval subsets, quotient real equality, countable choice, and standard bridges.*

Proof. Carrier admission is projection from $M = (I, r, W, S, H, C, P, N)$ in Definition 7129.1. These are precisely the rows exposed to a local consumer.

The interval and tolerance obligations are finite because I is read from the $\mathsf{RationalIntervalRefinement}^{\uparrow}$ surface and r is read from the $\mathsf{DyadicRatCore}^{\uparrow}$ surface. The modulus obligation is exactly the retained window row W, with S tying the same request to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback data.

The remaining rows transport, replay, source, and name the same displayed packet. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface and exclude the named nonlocal objects. □

Falsifiable prediction: Any accepted $\mathsf{BishopIntervalModulus}^{\uparrow}$ consumer that reads interval-local modulus control must display the rational refinement row, dyadic tolerance row, retained window row, source handoff row, componentwise transport, replay, provenance, and local NameCert row.

Independence witness: rationalintervalrefinement, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{BishopIntervalModulus}^{\uparrow}$ because this packet alone binds a dyadic tolerance request to a retained rational interval window and the corresponding finite regular-name readback.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalModulus}^{\uparrow}$ packet is generated from a RationalIntervalRefinement row, a DyadicRatCore tolerance, a modulus-window row, StreamName, RegSeqRat, and Real-facing source data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopIntervalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7129.1, Definition 7129.2, and Definition 7129.3 seed the finite interval-local modulus packet over RationalIntervalRefinement, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopIntervalModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite interval trees, completed real point selection, host interval subsets, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window stability, NameCert obligation coverage, source handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BishopIntervalModulusUp.

Chapter 7130

A Concrete Naming Certificate for $\mathsf{RegularCauchyPairModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ packet records finite shared-window modulus evidence for a pair of regular Cauchy rational names before either name is read by a completion-facing consumer. It sits over the stream windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the rate readback of chapter 2085. The packet supplies a finite paired-modulus boundary for equality, null-difference, separability, and completion arguments without turning shared precision evidence into quotient equality or a host-selected limit.

Definition 7130.1 (Regular Cauchy pair modulus carrier)**.** *A* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R_0, R_1, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations are compared only through finite precision requests.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row exposing one finite stage at which both source rows are observed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the shared window, including the two endpoint observations, the requested radius, and the slack cells needed for a paired read.*
4. *M is the paired modulus row assigning each requested joint precision to the displayed shared window and dyadic tolerance data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for any completion-facing read that consumes the paired modulus.*
6. *H records componentwise* hsame *transport for the two regular sources, the shared window, the dyadic ledger, the paired modulus row, the Real seal, provenance, and local naming.*
7. *C records displayed* Cont *replay for the finite route*

$$R_0, R_1 \longmapsto W \longmapsto D \longmapsto M \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPairModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, W, D, M, E, H, C, P, N$. It has no coordinate for quotient equality of regular names, selected limits, trichotomy, countable choice, hidden independent moduli, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7130.2 (Regular Cauchy pair modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ *carrier*

$$X = (R_0, R_1, W, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPairModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, paired modulus row,* $\mathsf{Real}^{\uparrow}$ *seal, transport row, replay row, provenance row, and local naming row;*
2. *shared-window exactness, requiring both source reads to use the same displayed W before any paired modulus entry is consumed;*
3. *dyadic-budget exactness, requiring D to expose the requested radius, endpoint observations, and slack rows used by the paired read;*
4. *paired-modulus exactness, requiring M to send each joint precision request only to the displayed W, D data and not to a detached selector or countable-choice schedule;*
5. *Real-seal nonescape, requiring E to be downstream of R_0, R_1, W, D, M and to export no quotient equality, selected limit, trichotomy, or ambient completeness claim;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name the same finite shared-window route.*

Proof. Carrier admission is projection from Definition 7130.1. The accepted packet displays exactly two regular rational rows, one shared window row, one dyadic ledger, one paired modulus row, one Real seal, and the structural rows H, C, P, N.

Shared-window exactness is read from R_0, R_1, W. Both source rows must be observed at the same displayed finite stage before the packet can answer a joint precision request.

Dyadic-budget exactness is the role of D. The ledger stores the requested radius, endpoint observations, and slack cells for the same shared window, so the paired read cannot appeal to an independent tolerance source.

The paired modulus row M then assigns the joint precision request to the displayed W, D data. Since M is the only modulus coordinate in the carrier, no selected limit, quotient equality, trichotomy, or countable-choice schedule is available.

Finally, E, H, C, P, N close the route. The Real seal is downstream of the paired modulus, while transport, replay, provenance, and naming preserve only the displayed finite packet; hence the listed obligations exhaust the local NameCert surface. □

Theorem 7130.3 (Regular Cauchy pair modulus shared-window handoff)**.** *Every equality, null-difference, separability, or completion-route consumer of an accepted* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ *carrier receives paired regular-Cauchy precision evidence only through*

$$R_0, R_1 \longmapsto W \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

The handoff preserves the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the paired modulus row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data. It exports no quotient equality, selected limit, trichotomy, countable choice, hidden independent modulus, or ambient Real completeness theorem.*

Proof. Begin with a consumer read of an accepted packet. By Definition 7130.2, the read starts from the displayed rows $R_0, R_1, W, D, M, E, H, C, P, N$.

For equality or null-difference uses, the consumer first reaches the two regular rational source rows through the same shared window W. The dyadic ledger D then supplies the radius and slack data at that window, so no quotient equality or detached comparison coordinate is present.

For separability or completion-route uses, the consumer must still pass through M before the terminal Real seal E is visible. Thus the precision evidence is the paired modulus row itself rather than a selected limit or ambient completeness principle.

The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Therefore every downstream handoff factors through the shared-window packet and receives none of the excluded escape objects. □

Falsifiable prediction: Every $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ read factors through two RegSeqRat source rows, one shared StreamName window row, one DyadicRatCore tolerance ledger, a paired modulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-rate-extraction, real: $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, one shared finite window, dyadic paired-tolerance data, a paired modulus row, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPairModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one shared StreamName window, a DyadicRatCore tolerance ledger, a paired modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyPairModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7130.1, Definition 7130.2, and Definition 7130.3 seed the finite shared-window modulus route for paired regular Cauchy reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyPairModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of regular names, selected limits, trichotomy, countable choice, hidden independent moduli, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exactness, dyadic-budget exactness, paired-modulus exactness, Real-seal nonescape, and structural stability for BEDC.Derived.RegularCauchyPairModulusUp.

Chapter 7131

A Concrete Naming Certificate for $\mathsf{DyadicCofinalGrid}^{\uparrow}$

Following Bishop and Bridges constructive analysis, finite dyadic approximation work is read through explicit meshes and tolerances rather than through an ambient dense-subset theorem. The $\mathsf{DyadicCofinalGrid}^{\uparrow}$ packet records the finite cofinal grid surface used when rational sources are approximated by dyadic windows before reaching regular rational sequences and real-name seals. It sits after the rational source of chapter 37, the dyadic ledger of chapter 291, the finite request windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite grid carrier only; it does not assert quotient real equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7131.1 (Dyadic cofinal grid carrier)**.** *A* $\mathsf{DyadicCofinalGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, D, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row whose finite rational samples are the only inputs inspected by the grid.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording dyadic denominators, endpoint comparisons, and tolerance budgets.*
3. *M is the cofinal mesh row: for each displayed tolerance in D, it lists a finite dyadic grid whose spacing refines that tolerance on the displayed rational window.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which precision requests and grid cells are opened.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the rational source, dyadic scale, cofinal mesh, and stream window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, D, M, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the rational source, dyadic scale, mesh, windows, readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a grid request through $Q, D, M, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicCofinalGrid}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCofinalGrid}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, D, M, W, R, E, H, C, P, N$. It has no coordinate for arbitrary real-density completion, quotient equality of real names, countable choice, selected limiting grids, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7131.2 (Dyadic cofinal grid NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCofinalGrid}^{\uparrow}$ *carrier*

$$G = (Q, D, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCofinalGrid}^{\uparrow}}$ *surface consists exactly of rational source exposure, dyadic scale exposure, cofinal mesh admission, StreamName window exposure, RegSeqRat readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may be discharged by an ambient density theorem, quotient real equality, countable choice, a selected limiting grid, or ambient real completeness.*

Proof. Project the carrier of Definition 7131.1. The displayed analytic rows are exactly Q, D, M, W, R, E, and the structural rows are exactly H, C, P, N.

Read the approximation data in dependency order. The rational samples enter through Q, the dyadic denominators and tolerances enter through D, and the cofinal mesh row M records only finite dyadic grids attached to those displayed tolerances.

The window and readback rows are downstream. The row W opens only the requested precision cells, and R reads those cells as a regular rational sequence before the terminal real seal E is visible.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no additional coordinate exists, the excluded density, quotient, choice, selected-limit, and completeness principles are not part of the local NameCert surface. □

Theorem 7131.3 (Dyadic cofinal grid real readback route)**.** *Let $G = (Q, D, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicCofinalGrid}^{\uparrow}$ *carrier. Every Real-facing consumer of the grid must read the rational source row Q, dyadic scale row D, cofinal mesh row M, StreamName window row W, and RegSeqRat readback row R before it reaches the Real seal row E. The route supplies finite dyadic approximation data for displayed tolerances; it does not supply real completeness, quotient equality of real names, a selected limit, or countable choice.*

Proof. Begin with an accepted consumer read of G. Carrier admission exposes the tuple $Q, D, M, W, R, E, H, C, P, N$ and no other analytic row.

The Real-facing route is ordered by the displayed rows. The source row Q supplies finite rational samples, D supplies dyadic scales, M supplies the cofinal finite grids for those scales, and W records the finite precision requests being opened.

Only after those rows are visible can R be read as a regular rational handoff. The terminal seal E is downstream of that handoff, so a consumer cannot reach $\mathsf{Real}^{\uparrow}$ through a hidden density or limit principle.

The transport, replay, provenance, and local naming rows preserve the same route. Therefore every Real-facing read factors through the displayed finite grid surface and leaves real completeness, quotient equality, selected limits, and countable choice outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCofinalGrid}^{\uparrow}$ read factors through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, cofinal mesh data, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCofinalGrid}^{\uparrow}}$.

Constructive story: A DyadicCofinalGrid$^{\uparrow}$ packet is generated from RatUp source rows, DyadicRatCoreUp scale rows, finite cofinal mesh rows, StreamNameUp windows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicCofinalGrid}^{\uparrow}}$, seed)

Scope closed: Definition 7131.1, Definition 7131.2, and Definition 7131.3 seed the finite dyadic-grid route over RatUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicCofinalGrid}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, real completeness, quotient equality of real names, countable choice, selected limiting grids, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exactness, cofinal mesh admission, StreamName window replay, RegSeqRat readback exactness, RealUp seal nonescape, transport stability, and TasteGate field faithfulness for BEDC.Derived.DyadicCofinalGridUp.

Chapter 7132

A Concrete Naming Certificate for $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ packet records a shared-window synchronization certificate for paired regular Cauchy reads before the pair is handed to equality, completion, or Real-facing consumers. It refines the paired-source route of chapter 7130 by naming the synchronization obligation as its own finite carrier surface over the regular rational rows of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet synchronizes two displayed reads at one shared window; it does not identify quotient names, select limits, invoke trichotomy, use countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7132.1 (Regular Cauchy synchronization modulus carrier)**.** *A* $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R_0, R_1, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are to be synchronized.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row at which both reads are inspected for the same requested precision.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested dyadic radius, the two displayed observations, and the slack cells used to compare them inside* W*.*
4. M *is the synchronization modulus row assigning each paired precision request to the displayed window and tolerance data.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available to downstream equality, completion, or Real-facing consumers only after* R_0, R_1, W, D, M *have been read.*
6. H *records componentwise* hsame *transport for the two sources, the shared window, the tolerance ledger, the synchronization modulus, the Real seal, provenance, and local naming.*
7. C *records displayed* Cont *replay for the route*

$$R_0, R_1 \longmapsto W \longmapsto D \longmapsto M \longmapsto E.$$

8. *P is the* Pkg *provenance over* RegularCauchySynchronizationModulus$^{\uparrow}$, RegularCauchyPairModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{RegularCauchySynchronizationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R_0, R_1, W, D, M, E, H, C, P, N$. It has no coordinate for quotient equality of regular Cauchy names, a selected limit, host real equality, trichotomy, countable choice, hidden independent windows, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7132.2 (Regular Cauchy synchronization modulus NameCert obligations)**.** *For every accepted* RegularCauchySynchronizationModulus$^{\uparrow}$ *carrier*

$$S = (R_0, R_1, W, D, M, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchySynchronizationModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* RegSeqRat$^{\uparrow}$ *source rows, shared* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *tolerance ledger, synchronization modulus row,* Real$^{\uparrow}$ *seal, transport row, replay row, provenance row, and local naming row;*
2. *shared-window synchronization, requiring both source reads to use the same displayed W before a paired precision answer is accepted;*
3. *dyadic-budget synchronization, requiring D to expose the requested radius, both endpoint observations, and the slack rows used by the synchronized read;*
4. *modulus exactness, requiring M to route each paired precision request through the displayed W, D data rather than through a detached selector, a second independent window, or a countable-choice schedule;*
5. *Real-seal handoff, requiring E to be downstream of R_0, R_1, W, D, M and to export no quotient equality, selected limit, trichotomy, or ambient completeness claim;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name the same finite synchronization route.*

Proof. Carrier admission is projection from Definition 7132.1. The accepted packet displays exactly two regular rational sources, one shared window, one dyadic ledger, one synchronization modulus row, one Real seal, and the structural rows H, C, P, N.

Shared-window synchronization is read from R_0, R_1, W. Both source rows are inspected at the same finite window before any paired precision answer is accepted, so the packet cannot replace the displayed window by two unrelated observation schedules.

Dyadic-budget synchronization is the role of D. The ledger stores the requested radius, the two endpoint observations, and the slack cells for the same shared window, which prevents the synchronized read from appealing to an independent tolerance source.

The modulus exactness clause is supplied by M. Since M is the only synchronization-modulus coordinate in the carrier, every paired precision request is routed through W, D and not through quotient equality, selected limits, trichotomy, or countable choice.

Finally E, H, C, P, N close the route. The Real seal is downstream of the synchronized finite packet, while transport, replay, provenance, and local naming preserve only that packet; hence the listed obligations exhaust the local NameCert surface. □

Theorem 7132.3 (Regular Cauchy synchronization shared-window handoff)**.** *Every equality, completion, null-difference, or Real-facing consumer of an accepted* $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ *carrier receives synchronized paired-source evidence only through*

$$R_0, R_1 \longmapsto W \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

The handoff preserves the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the synchronization modulus row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data. It exports no quotient equality, selected limit, trichotomy, countable choice, hidden independent window, or ambient Real completeness theorem.*

Proof. Begin with a consumer read of an accepted packet. By Definition 7132.2, the read starts from the displayed rows $R_0, R_1, W, D, M, E, H, C, P, N$.

For equality or null-difference uses, the consumer first reaches the two regular rational source rows through the same shared window W. The dyadic ledger D supplies the requested radius and slack data at that window, so no quotient equality or detached comparison coordinate is present.

For completion or Real-facing uses, the consumer must still pass through the synchronization modulus row M before the terminal Real seal E is visible. Thus the precision evidence is the displayed synchronization certificate rather than a selected limit or ambient completeness principle.

The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Therefore every downstream handoff factors through the shared-window packet and receives none of the excluded escape objects. □

Falsifiable prediction: Every $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ read factors through two RegSeqRat source rows, one shared StreamName window row, one DyadicRatCore tolerance ledger, a synchronization modulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-pair-modulus, regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ is not bijective to any listed sibling because it isolates the shared-window synchronization certificate for paired regular Cauchy reads as its own finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one shared StreamName window, a DyadicRatCore tolerance ledger, a synchronization modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7132.1, Definition 7132.2, and Definition 7132.3 seed the shared-window synchronization route for paired regular Cauchy reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySynchronizationModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality of regular Cauchy names, selected limits, host real equality, trichotomy, countable choice, hidden independent windows, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window synchronization, dyadic-budget synchronization, modulus exactness, Real-seal handoff, and structural stability for BEDC.Derived.RegularCauchySynchronizationModulusUp.

Chapter 7133

A Concrete Naming Certificate for $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$ packet records the finite BEDC route by which a continuous real-valued map on a closed bounded interval is read as Riemann integrable. It sits after the interval carrier of chapter 1591, the map surface of chapter 130, the uniform-modulus row of chapter 369, the Riemann-sum gauge surface of chapter 4115, the gauge-integral packet of chapter 1999, the stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the compact-continuity theorem surface as finite NameCert data; it does not assert a host integration theorem, selected partitions, quotient functions, or ambient completeness.

Definition 7133.1 (Continuous compact Riemann-integrability carrier)**.** *A* $\mathsf{ContinuousCompactRiemannIntegrabilit}$ *carrier is a finite* BHist *packet*

$$I = (J, F, U, G, A, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row, exposing the compact interval endpoints and admitted interval classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, storing the real-valued map read only through its displayed continuity data.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, recording the finite modulus schedule used on the compact interval.*
4. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row, exposing tagged partition, mesh, and gauge-admission data.*
5. *A is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *row, recording the finite gauge-integral acceptance surface.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for finite partition and modulus probes.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row for rational readback of sums, tolerances, and endpoint windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value and comparison tolerances.*
9. *H records componentwise* hsame *transport for $J, F, U, G, A, S, R, E, C, P, N$.*
10. *C records displayed* Cont *replay from a compact-continuity integrability request through*

$$J, F, U \longmapsto G, A, S, R, E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J, F, U, G, A, S, R, E, H, C, P, N$. It has no coordinate for a host Riemann integral, selected partition family, quotient function equality, countable choice, or ambient completeness.

Theorem 7133.2 (Continuous compact Riemann-integrability obligations)**.** *For every accepted* $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$ *carrier $I = (J, F, U, G, A, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of compact interval admission through J, continuous-map readback through F, uniform-modulus exposure through U, Riemann-sum gauge data through G, gauge-integral acceptance through A, finite stream probes through S, regular rational readback through R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $J, F, U, G, A, S, R, E, H, C, P, N$ are the entire inventory visible to a compact-integrability consumer.

The source side is finite. The interval row J supplies only displayed closed-bounded endpoints, the map row F supplies only the continuity readback, and the modulus row U supplies the compact uniformity schedule.

The integration side is also finite. The gauge row G exposes tagged-partition and mesh data, while A records the gauge-integral acceptance surface. The stream row S and regular rational row R give the inspected finite probes and rational tolerances before the terminal real seal E is read.

Finally H, C, P, N transport, replay, source, and name the same route. Since no row contains a host integral, selected partition family, quotient function equality, countable choice, or ambient completeness, the listed rows exhaust the local obligations. □

Theorem 7133.3 (Continuous compact Riemann-integrability gauge route)**.** *Every accepted* ContinuousCompactRi *packet factors a compact-continuity integrability request through*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{RiemannSumGauge}^{\uparrow},\ \mathsf{GaugeIntegral}^{\uparrow},\ \mathsf{StreamName}$$

The consumer receives a displayed compact interval, continuous-map data, a uniform modulus, gauge-controlled Riemann sums, gauge-integral acceptance, finite probes, rational readback, a terminal real seal, transport, replay, provenance, and local naming. It receives no host integral theorem and no selected family of partitions.

Proof. Begin with a compact-continuity read. The interval and map rows are J and F, and compact uniformity is available only through the displayed modulus row U.

Use the Riemann-sum gauge row G to route each inspected partition and tolerance request. The gauge-integral row A records acceptance only after that finite gauge route has been displayed.

Read finite probes through S, rational bounds through R, and the terminal real value through E. The transport and replay rows H, C preserve the same route, and P, N source and name it locally.

Thus the exported integrability surface is exactly the gauge route named in the statement. A stronger read would require a carrier coordinate that I does not contain. □

Theorem 7133.4 (Continuous compact Riemann-integrability L10 non-escape)**.** *An accepted* $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$ *packet remains an L10/L11 finite NameCert route. Downstream Real or GaugeIntegral consumers may inspect* $J, F, U, G, A, S, R, E, H, C, P, N$*, but they cannot replace those rows by host compactness, host uniform continuity, a selected refining partition sequence, quotient-function equality, countable choice, or an ambient completeness theorem.*

Proof. Read the carrier inventory from Definition 7133.1. The only compactness-facing row is J, the only continuity-facing row is F, and the only compact-uniformity row is U.

Apply Definition 7133.3. The integrability read reaches G, A, S, R, E only after the displayed compact-continuity source has been exposed.

The remaining rows H, C, P, N preserve, replay, source, and name the same finite packet. None of them adds a selected refining sequence, host compactness theorem, quotient-function equality, countable-choice selector, or ambient completeness row.

Therefore the packet is a finite L10/L11 dependency route rather than a host theorem about all continuous functions on compact intervals. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$ packet is generated from closed bounded interval data, continuous-map readback, uniform-modulus scheduling, Riemann-sum gauge rows, gauge-integral acceptance, stream probes, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7133.1, Definition 7133.2, Definition 7133.3, and Definition 7133.4 seed the finite compact-continuity Riemann-integrability surface over interval, map, modulus, gauge, integral, stream, rational, Real, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, host uniform continuity, selected partition families, quotient-function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, continuous-map readback, uniform-modulus exposure, Riemann-sum gauge exactness, gauge-integral acceptance, stream probes, rational readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousCompactRiemannIntegrabilityUp.

Chapter 7134

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$

The $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ packet realizes the phase-transport budget surface of chapter 9463 as finite metric, phase, energy, and curvature-budget rows. It sits after the disk metric of chapter 6209, the hyperbolic metric row of chapter 3474, the phase-transport carrier of chapter 6146, and the finite ledger of chapter 6481. The packet names the local transport budget for one accepted observer-phase route; it does not assert a completed hyperbolic manifold, a physical energy law, global phase holonomy, or a host curvature theorem.

Definition 7134.1 (Hyperbolic phase-transport budget carrier)**.** *A* $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, M, G, E, K, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *row, exposing the inspected disk coordinate and disk-distance window.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, recording the finite hyperbolic metric comparison used by the transport budget.*
3. *G is the* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *row, storing the source phase, target phase, inverse route, and compositional boundary for the displayed transport.*
4. *E is the* $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$ *row, recording the finite energy debit, credit, and conservation check associated with the route.*
5. *K is the curvature-budget row, recording the local curvature allowance consumed by the displayed disk and phase transport.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for metric, energy, and curvature tolerances.*
7. *H records componentwise* hsame *transport for $D, M, G, E, K, R, C, P, N$.*
8. *C records displayed* Cont *replay from a phase-transport budget request through*

$$D, M, G, E \longmapsto K, R, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, M, G, E, K, R, H, C, P, N$. It has no coordinate for completed hyperbolic geometry, physical energy conservation, global holonomy, selected observer trajectories, or host curvature integration.

Theorem 7134.2 (Hyperbolic phase-transport budget obligations)**.** *For every accepted* HyperbolicPhaseTransportB *carrier $B = (D, M, G, E, K, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}}$ *surface consists exactly of disk-metric admission through D, hyperbolic metric readback through M, phase groupoid transport through G, energy-ledger exposure through E, curvature-budget recording through K, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $D, M, G, E, K, R, H, C, P, N$ are the entire public inventory of the budget packet.

The metric source is displayed before any phase budget is read. The disk row D supplies the inspected coordinate window, and the hyperbolic metric row M supplies the finite comparison used by the budget.

The phase and ledger rows are then read through G and E. The curvature row K records the local allowance consumed by that route, while R seals the metric, energy, and curvature tolerances as Real-facing data.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row contains completed hyperbolic geometry, physical energy conservation, global holonomy, selected observer trajectories, or host curvature integration, the listed rows exhaust the local obligations. □

Theorem 7134.3 (Hyperbolic phase-transport budget vision route)**.** *Every accepted* HyperbolicPhaseTransportBud *packet factors the vision-level phase-budget request through*

$$\mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{ObserverPhaseGroupoid}^{\uparrow},\ \mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow} \longmapsto K, \mathsf{Real}^{\uparrow}, H, C$$

The consumer receives disk metric data, hyperbolic metric comparison, phase groupoid transport, an energy ledger, a curvature-budget row, a Real seal, componentwise transport, replay, provenance, and local naming. It receives no global phase geometry, no physical observer trajectory, and no host curvature theorem.

Proof. Begin with the disk and hyperbolic metric rows. The route can inspect D and M, but it cannot introduce a completed hyperbolic metric space outside the packet.

Read phase transport through G and the energy ledger through E. These rows supply the local groupoid and ledger data named by the source vision.

The curvature budget is the finite row K, and the Real-facing tolerances are sealed by R. Transport and replay preserve the same row order through H and C, while P, N source and name the route.

Thus the vision-level request is concretized only as the finite packet displayed in the statement. Any stronger geometric or physical reading would require a coordinate absent from B. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ packet exposes disk metric, hyperbolic metric, phase groupoid transport, energy ledger, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows. A public phase-budget read that omits any of those rows, or that exports global phase geometry, physical observer motion, or host curvature integration, refutes the carrier boundary.

Independence witness: poincare-disk-metric, hyperbolic-metric, observer-phase-groupoid, hyperbolic-phase-energy-ledger, hyperbolic-phase-curvature-budget: $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk metric data, hyperbolic metric comparison, phase groupoid transport, energy ledger, curvature-budget recording, Real sealing, transport, replay, provenance, and local naming while refusing completed geometry, physical energy laws, global holonomy, selected trajectories, and host curvature theorems.

The Lean TasteGate surface for this packet must provide explicit $\mathsf{Nontrivial\ HyperbolicPhaseTransportBudget}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicPhaseTransportBudget}^{\uparrow}$ instances before a checked carrier-alignment theorem can witness the public route.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ packet is generated from Poincare disk metric data, hyperbolic metric comparison, observer phase groupoid transport, hyperbolic phase energy ledger rows, a curvature-budget row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7134.1, Definition 7134.2, and Definition 7134.3 seed the finite phase-transport budget packet over disk, metric, phase groupoid, energy ledger, curvature budget, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic geometry, physical energy conservation, global holonomy, selected observer trajectories, host curvature integration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk admission, metric readback, phase groupoid transport, energy-ledger exactness, curvature-budget recording, Real sealing, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseTransportBudgetUp.

Chapter 7135

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ packet records the finite extraction surface by which a regular Cauchy rational name answers a bounded precision request through displayed windows. It sits after the finite stream rows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the choice-free completion route of chapter 4237. The packet is a finite interface for Real and Completion consumers; it is not quotient completion, representative selection, countable choice, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7135.1 (Regular-Cauchy window extraction carrier)**.** *A* $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R, W, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational name, its modulus surface, and the readback rows available to a completion-facing consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *inspected-window row listing the finite windows and indices opened for the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bound, comparison rows, and arithmetic checks attached to the inspected windows.*
4. *Q is the finite query/request row that binds the consumer request to the displayed source, window, and tolerance rows.*
5. *H records componentwise* hsame *transport for R, W, D, Q, C, P, N.*
6. *C records the displayed* Cont *replay route from the query row through the regular source, inspected windows, dyadic tolerance checks, transport, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through R, W, D, Q, H, C, P, N. It has no coordinate for quotient streams, selected limits, countable-choice schedules, host equality of completed objects, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7135.2 (Regular-Cauchy window extraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ *carrier*

$$X = (R, W, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, classifier transport through H, finite-window extraction of the inspected* $\mathsf{StreamName}^{\uparrow}$ *rows, dyadic tolerance exactness through D, regular-rational readback stability through R, displayed replay through C, provenance through P, and local naming through N. Every completion-facing read is routed through the finite query row Q before it can consume a downstream* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the carrier of Definition 7135.1. The rows R, W, D, Q, H, C, P, N are exactly the displayed inventory exposed by an accepted packet.

Carrier admission and classifier transport are componentwise. The row H transports the source, inspected windows, dyadic tolerance ledger, query row, replay row, provenance, and local name without adding a second completion coordinate.

The extraction step is finite. The query row Q names the requested precision, W lists the inspected windows answering that request, and D records the dyadic tolerance comparisons used to certify the answer.

Readback stability is then the regular rational row R reread through the same windows and tolerances. Replay through C follows the displayed route, while P and N bind provenance and local naming.

Since the carrier contains no quotient stream equality, selected representative, countable-choice schedule, host equality, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle, none of those objects can replace the finite extraction surface. □

Theorem 7135.3 (Regular-Cauchy extraction nonescape)**.** *Every consumer-facing route out of an accepted* $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ *packet factors through the finite rows R, W, D, Q, H, C, P, N. In particular, a completion consumer receives the RegSeqRat source, inspected StreamName windows, DyadicRatCore tolerance ledger, finite request row, componentwise transport, replay, provenance, and local NameCert rows, and receives no quotient stream equality, selected limit, countable-choice schedule, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use the obligation surface of Definition 7135.2. It identifies each admissible consumer read with the displayed carrier rows.

The source, window, tolerance, and request rows R, W, D, Q form the public extraction surface. The structural rows H, C, P, N preserve, replay, source, and name that same surface.

Any route exporting quotient streams, selected limits, countable choice, host equality, compactness, or ambient completeness would require an additional carrier coordinate. The carrier has none, so every consumer-facing route remains inside the finite extraction packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ read exposes a RegSeqRat source row, StreamName inspected windows, a DyadicRatCore tolerance ledger, a finite query row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any completion-facing seal is consumed.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-choice-free-completion: $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a regular source, inspected windows, dyadic tolerances, finite precision requests, transport, replay, provenance, and local naming while refusing quotient completion, selected representatives, countable choice, host equality, compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowExtraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7135.1, Definition 7135.2, and Definition 7135.3 seed the finite regular-Cauchy window-extraction surface over RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite-window extraction, tolerance exactness, readback stability, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularCauchyWindowExtractionUp.

Constructive story: A $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName inspected windows, a DyadicRatCore tolerance ledger, a finite query row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7136

A Concrete Naming Certificate for $\mathsf{ChoiceSequenceBar}^{\uparrow}$

Following Brouwer on bars over choice sequences and the Bishop–Bridges use of finite prefix evidence, the $\mathsf{ChoiceSequenceBar}^{\uparrow}$ packet records a bar only as a finite BEDC surface over inspected choice-sequence prefixes. It sits beside the lawless-sequence surface of chapter 1495, the finite $\mathsf{StreamName}^{\uparrow}$ route of chapter 126, the $\mathsf{BaireSpace}^{\uparrow}$ prefix carrier of chapter 334, the $\mathsf{Bool}^{\uparrow}$ hit row of chapter 32, the $\mathsf{Nat}^{\uparrow}$ prefix ledger of chapter 28, and the finite fan/bar boundary of chapter 5863. The packet names prefix-bar coverage and monotone extension evidence; it is not a host infinite choice sequence, not countable choice, and not an arbitrary bar-induction principle.

Definition 7136.1 (Choice-sequence bar carrier)**.** *A* $\mathsf{ChoiceSequenceBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, B, I, M, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite prefix-window row over* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{BaireSpace}^{\uparrow}$*, listing exactly the inspected choice-sequence prefixes.*
2. *B is the* $\mathsf{Bool}^{\uparrow}$ *hit row, recording for each displayed prefix whether the bar has already been met.*
3. *I is the* $\mathsf{Nat}^{\uparrow}$ *prefix ledger, ordering the inspected prefix lengths and finite extension steps.*
4. *M is the monotone-extension evidence row: once a displayed prefix has a positive hit in B, every displayed extension of that prefix in W keeps the same hit.*
5. *H records componentwise* hsame *transport for prefix windows, Bool hit rows, prefix ledgers, monotone-extension evidence, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a prefix-ledger request through the finite window W, the hit row B, and the monotone-extension row M.*
7. *P is the* Pkg *provenance over* $\mathsf{ChoiceSequenceBar}^{\uparrow}$*,* $\mathsf{LawlessSequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceBar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through W, B, I, M, H, C, P, N. It has no coordinate for a completed choice sequence, a branch selector, a countable-choice schedule, quotient equality of streams, or bar induction outside the displayed finite prefix surface.

Theorem 7136.2 (Choice-sequence bar NameCert obligations)**.** *For every accepted* $\mathsf{ChoiceSequenceBar}^{\uparrow}$ *carrier* $Q = (W, B, I, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceBar}^{\uparrow}}$ *surface consists exactly of carrier admission for the eight displayed rows, finite-prefix window coverage through* W*, Bool hit readback through* B*, Nat prefix-ledger ordering through* I*, monotone extension stability through* M*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every accepted bar-facing read remains inside the same finite prefix packet and cannot escape to a host infinite choice sequence or a countable-choice schedule.*

Proof. Project the carrier of Definition 7136.1. The only bar data visible to a consumer are the displayed prefix window W, the Bool hit row B, the prefix ledger I, and the monotone-extension evidence M.

Use finite induction over the $\mathsf{Nat}^{\uparrow}$ ledger I. The base read inspects the displayed empty or initial prefix when it is present in W; each step extends the current displayed prefix by one ledger move and then reads the corresponding hit value from B.

Apply the monotone-extension row M. If a displayed prefix has already hit the bar, every displayed extension keeps that hit, so prefix-bar coverage and extension stability are properties of the finite packet rather than of an unseen branch.

Finally replay through H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve exactly the same rows, and none of them contains a coordinate for a host infinite choice sequence, a branch selector, countable choice, quotient stream equality, or unrestricted bar induction. □

Falsifiable prediction: An accepted ChoiceSequenceBarUp packet that exports a bar hit without displaying the prefix window W, Bool hit row B, Nat prefix ledger I, monotone-extension row M, and structural H, C, P, N route refutes the carrier.

Independence witness: lawlesssequence, streamname, bairespace, bool, nat, finitefantheorem: $\mathsf{ChoiceSequenceBar}^{\uparrow}$ is not bijective to any listed sibling because it binds finite prefix windows, Bool bar hits, Nat prefix ledgers, monotone-extension evidence, transport, replay, provenance, and local NameCert into one prefix-bar packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequenceBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoiceSequenceBar}^{\uparrow}$ packet is generated from finite StreamName and BaireSpace prefix windows, Bool bar-hit rows, Nat prefix ledgers, monotone-extension evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceBar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7136.1 and Definition 7136.2 seed the finite prefix-bar surface over choice-sequence prefixes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceBar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host infinite choice sequences, countable choice, quotient stream equality, selected branches, or arbitrary bar induction.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window coverage, Bool hit readback, Nat ledger ordering, monotone extension stability, transport, replay, provenance, and local naming for BEDC.Derived.ChoiceSequenceBarUp.

Chapter 7137

A Concrete Naming Certificate for $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$

Following Arzela, Ascoli, Kelley, and Bishop–Bridges, the $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$ packet records the constructive compact-open Ascoli route where compact-open source probes, equicontinuity moduli, compact metric source data, complete metric target data, and finite-window diagonal extraction are exposed as NameCert rows. It sits over the compact-open topology certificate of chapter 4138, the equicontinuity-family certificate of chapter 413, and the Ascoli finite-window handoff of chapter 412. The packet is a finite constructive-analysis carrier; it does not assert open-cover compactness, select a subnet, quotient functions by host equality, or import ambient completeness outside the displayed rows.

Definition 7137.1 (Compact-open equicontinuous Ascoli carrier)**.** *A* $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, E, X, Y, D, S, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *source-probe row, listing the compact test windows and finite function-observation requests admitted by the compact-open topology.*
2. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *modulus row, recording the shared finite equicontinuity moduli used on the displayed compact windows.*
3. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, read only through finite compact-metric witnesses and finite net data.*
4. *Y is the* $\mathsf{CompleteMetric}^{\uparrow}$ *target row, exposing the completion-facing target metric data used by limit-window reads.*
5. *D is the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *finite-window diagonal handoff row, binding compact source windows, equicontinuity moduli, and target completeness into one displayed extraction route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the inspected family stages and diagonal prefixes.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerance observations attached to the selected finite windows.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for scalar tolerances, distances, and completion-facing readbacks.*

9. *L records local compact-open and equicontinuity compatibility of O, E, X, Y, D, S, Q, R.*
10. *H records componentwise* hsame *transport for every displayed row and the local naming row.*
11. *C records displayed* Cont *replay for the finite route*

$$O, E, X, Y \longmapsto D \longmapsto S \longmapsto Q \longmapsto R.$$

12. *P is the* Pkg *provenance over* $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, EquicontinuityFamily ArzelaAscoli$^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, E, X, Y, D, S, Q, R, L, H, C, P, N$. It has no coordinate for a selected subnet, host function equality, open-cover compactness, quotient function space, countable choice, or ambient target completeness beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Theorem 7137.2 (Compact-open equicontinuous Ascoli NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$ *carrier*

$$A = (O, E, X, Y, D, S, Q, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}}$ *surface consists exactly of compact-open source-probe exposure through O; equicontinuity-family modulus exposure through E; compact metric source exposure through X; complete metric target exposure through Y; Ascoli finite-window diagonal handoff through D; StreamName window exposure through S; RegSeqRat readback through Q; Real sealing through R; local compatibility through L; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 7137.1. The rows $O, E, X, Y, D, S, Q, R, L, H, C, P, N$ are exactly the displayed compact-open, equicontinuity, compact-source, complete-target, diagonal, stream, rational-readback, Real-seal, compatibility, transport, replay, provenance, and naming coordinates.

The compact-open and equicontinuity inputs are read first. The row O supplies only finite compact-window probes, while E supplies the shared modulus data needed to compare a family on those windows. The compact source row X and complete target row Y are therefore visible before any diagonal extraction is consumed.

The finite-window exhaustion is the role of D, S, Q, R. The Ascoli handoff D names the displayed diagonal route, S records the inspected windows, Q records rational readbacks, and R seals the scalar tolerance and distance data for downstream completion-facing consumers.

Finally L, H, C, P, N bind the route locally. Compatibility, transport, replay, provenance, and naming preserve only the projected finite packet, so no selected subnet, quotient function equality, open-cover compactness, countable choice, or ambient completeness principle can be exported from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenEquicontinuousAscoli}^{\uparrow}$ packet is generated from compact-open source probes, equicontinuity-family moduli, a compact metric source row, a complete metric target row, an Ascoli finite-window diagonal handoff, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactOpenEquicontinuousAscoli}^{\uparrow}}$, seed)

Scope closed: Definition 7137.1 and Definition 7137.2 seed the finite compact-open equicontinuous Ascoli route over CompactOpenTopology$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, ArzelaAscoli$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactOpenEquicontinuousAscoli}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected subnets, quotient function equality, open-cover compactness, countable choice, ambient target completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open probe exposure, equicontinuity modulus exposure, finite-window diagonal handoff, StreamName and RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactOpenEquicontinuousAscoliUp.

Chapter 7138

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$

The $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ packet realizes the closed reversible phase-transport branch of chapter 9463 as finite disk, boundary, metric, holonomy, transport, provenance, and NameCert rows. It sits after the hyperbolic metric route of chapter 3474, the disk-automorphism route of chapter 6776, the reversible observer-transport route of chapter 6146, the closed phase-holonomy route of chapter 6385, and the scalar seal of chapter 38. The packet names torsion only as a finite obstruction row for closed phase transport; it is not a host cohomology class, not a global curvature theorem, and not a selected boundary fixed point.

Definition 7138.1 (Hyperbolic phase torsion carrier)**.** *A* $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, B, M, A, G, H, R, \tau, U, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare-disk readout row, listing the finite disk points and disk-coordinate windows used by the torsion read.*
2. *B is the boundary-fixedness row, recording which displayed boundary directions remain attached to the closed transport route.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row used for distance and displacement comparisons.*
4. *A is the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *reversible disk-automorphism row, including inverse and composition readbacks for the displayed loop.*
5. *G is the* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *row carrying source, target, inverse, and composition data for observer-facing phase transport.*
6. *H is the* $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ *closed holonomy row attached to the displayed loop.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for the phase residue, displacement budget, and tolerance comparisons.*
8. *τ is the local torsion row, recording the finite obstruction residue left after the closed reversible transport is replayed.*
9. *U records componentwise* hsame *transport for disk, boundary, metric, automorphism, groupoid, holonomy, Real, torsion, provenance, and local naming rows.*

10. *C records displayed* Cont *replay for the finite route*

$$D, B, M, A, G, H \longmapsto R \longmapsto \tau.$$

11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTorsion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, B, M, A, G, H, R, \tau, U, C, P, N$. It has no coordinate for host curvature, selected boundary points, quotient phase equality, global disk dynamics, hidden observer state, or a detached holonomy invariant outside the displayed loop.

Theorem 7138.2 (Hyperbolic phase torsion NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ *carrier*

$$T = (D, B, M, A, G, H, R, \tau, U, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTorsion}^{\uparrow}}$ *surface consists exactly of disk-window exposure through D; boundary-fixedness exposure through B; metric comparison through M; reversible disk automorphism through A; observer-phase groupoid transport through G; closed phase holonomy through H; Real sealing through R; finite torsion residue through τ; componentwise transport through U; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 7138.1. The rows $D, B, M, A, G, H, R, \tau, U, C, P, N$ are exactly the displayed disk, boundary, metric, automorphism, groupoid, holonomy, scalar, torsion, transport, replay, provenance, and naming coordinates.

The reversible route is read before the torsion residue. The disk windows D, boundary row B, metric row M, automorphism row A, groupoid row G, and holonomy row H determine the closed transport loop visible to the packet.

The scalar and residue rows are downstream. The Real seal R records the displayed phase and displacement budgets, and τ records only the finite obstruction residue left by the replayed loop.

Finally U, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since no other coordinate appears in the carrier, the NameCert surface exports no host curvature theorem, selected boundary fixed point, quotient phase equality, global disk dynamics, or hidden observer state. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ read factors through displayed disk windows, boundary-fixedness rows, HyperbolicMetricUp comparisons, PoincareDiskAutomorphismUp reversible transport, ObserverPhaseGroupoidUp composition, HyperbolicPhaseHolonomyUp closed-loop readback, a RealUp scalar seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming. A public torsion read that bypasses these rows refutes the carrier.

Independence witness: hyperbolic-metric, poincare-disk-automorphism, observer-phase-groupoid, hyperbolic-phase-holonomy, real: $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk, boundary, metric, reversible transport, groupoid composition, closed holonomy, Real sealing, a local torsion residue, transport, replay, provenance, and naming rows.

The proposed Lean TasteGate surface for $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ asks for a Nontrivial witness that some accepted closed phase-transport packet exposes a nonzero displayed torsion residue, and a FieldFaithful witness that the Real-sealed residue is still read through $D, B, M, A, G, H, R, \tau, U, C, P, N$ rather than through a detached scalar field.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTorsion}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseTorsion}^{\uparrow}$ packet is generated from disk windows, boundary-fixedness rows, hyperbolic metric comparisons, reversible disk automorphisms, observer phase groupoid transport, closed phase holonomy, a RealUp scalar seal, a local torsion residue, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTorsion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7138.1 and Definition 7138.2 seed the finite reversible closed phase-transport torsion route over $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseTorsion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host curvature, global disk dynamics, selected boundary fixed points, quotient phase equality, hidden observer state, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-fixedness exposure, reversible transport replay, holonomy residue sealing, torsion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseTo

Chapter 7139

A Concrete Naming Certificate for $\mathsf{MirskyTheorem}^{\uparrow}$

Following Mirsky's theorem for finite partially ordered sets, the $\mathsf{MirskyTheorem}^{\uparrow}$ packet records the height-dual certificate to chapter 6083: a finite $\mathsf{Poset}^{\uparrow}$ row, a chain-height ledger, an antichain-partition ledger, rank-layer witnesses, and local NameCert data are read as one finite certificate surface. The packet is also adjacent to the finite ordered rows of chapter 2003, the matching discipline of chapter 234, and the ordered witness spines of chapter 36. It exports only displayed finite rows, not a host poset, not quotient order equality, and not a selected partition outside the carrier.

Definition 7139.1 (Mirsky theorem carrier)**.** *A* $\mathsf{MirskyTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, H, A, R, K, S, T, C, P, N)$$

with the following displayed rows:

1. *O is the finite* $\mathsf{Poset}^{\uparrow}$ *order row, including the carried elements, comparison ledger, and classifier for repeated element reads.*
2. *H is the chain-height row, recording a displayed maximal-chain height witness and a finite ledger that bounds all displayed chains by that height.*
3. *A is the antichain-partition row, recording finitely many layers whose union covers the carried elements and whose internal pairs are incomparable according to O.*
4. *R is the rank-layer row, assigning each element to the first displayed height level at which it appears in the finite predecessor recursion.*
5. *K is the chain-antichain consistency row, checking that the height ledger H and partition ledger A meet at the same number of displayed layers.*
6. *S is the sibling route row to* $\mathsf{DilworthDecomposition}^{\uparrow}$*,* $\mathsf{SpernerLemma}^{\uparrow}$*,* $\mathsf{Matching}^{\uparrow}$*, and* $\mathsf{List}^{\uparrow}$*.*
7. *T is componentwise* hsame *transport for the order, height, partition, rank, consistency, sibling route, replay, provenance, and local naming rows.*
8. *C is the displayed* Cont *replay route from O, H, A, R, K, S through T to P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{MirskyTheorem}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*,* $\mathsf{DilworthDecomposition}^{\uparrow}$*,* $\mathsf{SpernerLemma}^{\uparrow}$*,* $\mathsf{Matching}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MirskyTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient host partial order, quotient order equality, unlisted antichain partition, selected rank function outside R, countable choice, or infinite-height theorem.

Lean4 (statement) `BEDC.Derived.MirskyTheoremUp`

Theorem 7139.2 (Mirsky theorem NameCert obligations)**.** *For an accepted* $\mathsf{MirskyTheorem}^{\uparrow}$ *carrier*

$$M = (O, H, A, R, K, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MirskyTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite order, chain-height, antichain-partition, rank-layer, consistency, sibling-route, transport, replay, provenance, and local naming rows; order exposure through O; height exposure through H; antichain-partition exposure through A; rank-layer compatibility through R; height-partition equality through K; and nonescape for host posets, quotient order equality, unlisted partitions, external rank selectors, choice, and infinite-order data.*

Proof. Project the carrier of Definition 7139.1. The finite order is O, the height witness is H, the antichain partition is A, the rank-layer assignment is R, and the equality check between height and layer count is K.

The order exposure obligation is rowwise. Every chain, layer, rank comparison, and sibling read uses the same displayed $\mathsf{Poset}^{\uparrow}$ comparison ledger, so a consumer cannot mix the height of one order with the partition of another.

The height and partition obligations are the two visible halves of Mirsky's finite theorem. Row H records the maximal-chain height boundary, while A records the layer partition into antichains.

The rank and consistency rows then bind those halves. Row R places elements into displayed layers using the finite predecessor recursion, and row K checks that the number of layers agrees with the height ledger.

Finally S, T, C, P, N preserve the sibling route, transport, replay, provenance, and local name. Since the listed rows exhaust the carrier, the excluded host order, quotient equality, unlisted partitions, external rank selectors, choice, and infinite-order principles have no place in the certificate. □

Theorem 7139.3 (Mirsky chain-antichain rank handoff)**.** *Let $M = (O, H, A, R, K, S, T, C, P, N)$ be an accepted* $\mathsf{MirskyTheorem}^{\uparrow}$ *carrier. Every chain-height or antichain-partition consumer exported by M factors through the displayed route*

$$\mathsf{Poset}^{\uparrow},\ \mathsf{DilworthDecomposition}^{\uparrow},\ \mathsf{SpernerLemma}^{\uparrow},\ \mathsf{Matching}^{\uparrow},\ \mathsf{List}^{\uparrow}$$

by reading $O, H, A, R, K, S, T, C, P, N$ in order. The handoff identifies height, rank layers, and antichain partition only through the displayed finite rows, and it exports no host cardinality theorem, quotient order equality, selected external partition, countable choice, or infinite poset theorem.

Proof. First read the finite order row. The $\mathsf{Poset}^{\uparrow}$ packet supplies the carried elements, comparison ledger, and classifier needed before chains, ranks, or antichain layers can be inspected.

Next read the height and partition rows. The chain-height row H records the displayed height witness, while the antichain-partition row A records the covering family of layers.

The rank handoff is supplied by R and K. The rank-layer row assigns elements to finite levels, and the consistency row checks that the layer count and chain-height ledger meet at the same displayed boundary.

Finally route the handoff through S, T, C, P, N. The sibling route may cite Dilworth, Sperner, Matching, and List packets as finite dependency surfaces, but the exported Mirsky read remains the listed BEDC packet rather than an ambient theorem over a host poset. □

Closure status for $\mathsf{NameCert}_{\mathsf{MirskyTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MirskyTheorem}^{\uparrow}$ packet is generated from a finite Poset row, a chain-height ledger, an antichain-partition ledger, rank-layer witnesses, height-partition consistency, sibling route rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MirskyTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7139.1, Definition 7139.2, and Definition 7139.3 seed the finite chain-height and antichain-partition certificate over Poset, DilworthDecomposition, SpernerLemma, Matching, and List rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MirskyTheorem}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim host partial orders, quotient order equality, selected external partitions, countable choice, infinite posets, checked Lean obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, order exposure, height exposure, antichain-partition exposure, rank-layer compatibility, height-partition consistency, transport, replay, provenance, and local naming for BEDC.Derived.MirskyTheoremUp.

Chapter 7140

A Concrete Naming Certificate for $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$

Following Bishop and Bridges on constructive interval compactness, the $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ packet records a finite dyadic endpoint grid over a displayed closed Real interval. It sits over the dyadic arithmetic core of chapter 291, the finite stream-name windows of chapter 126, the regular rational readback rows of chapter 301, the terminal Real seal of chapter 38, and the closed interval boundary of chapter 1591. The packet names a constructive finite grid object for Real and Completion consumers; it is not Heine–Borel compactness, not open-cover compactness, not a selected limit principle, not trichotomy, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7140.1 (CompactIntervalDyadicGrid carrier)**.** *A* $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, \delta, E, M, W, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row naming the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seals, endpoint order data, and interval-containment boundary.*
2. *δ is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row used as the displayed grid precision.*
3. *E is the endpoint-containment row recording dyadic lower and upper endpoint brackets for I at mesh δ.*
4. *M is the finite mesh row listing the dyadic grid cells between the endpoint brackets.*
5. *W is the finite window row exposing the* $\mathsf{StreamName}^{\uparrow}$ *observations used to read the endpoint brackets and each grid cell.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row sealing the endpoint and cell reads as regular rational names and terminal Real-facing rows.*
7. *H records componentwise* hsame *transport for the interval source, dyadic mesh, endpoint containment, finite mesh, finite windows, regular handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interval-and-mesh request through $I, \delta, E, M, W, R, H, P, N$ without adding an off-grid point.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactIntervalDyadicGrid}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $I, \delta, E, M, W, R, H, C, P, N$. It has no coordinate for an arbitrary open cover, a choice-selected finite subcover, a selected limit point, trichotomy for Real endpoints, quotient real equality, or ambient Real completeness.

Theorem 7140.2 (CompactIntervalDyadicGrid NameCert obligations)**.** *For every accepted* CompactIntervalDyadic *carrier $G = (I, \delta, E, M, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalDyadicGrid}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed interval source, dyadic mesh, endpoint-containment, finite mesh, finite window, regular-rational handoff, and Real-facing seal rows; componentwise classifier transport through H; displayed continuation replay through C; provenance through P; local naming through N; and nonescape from arbitrary open covers, selected finite subcovers, selected limits, trichotomy, quotient real equality, and ambient Real completeness.*

Proof. Project the carrier of Definition 7140.1. The only interval source is I, the only mesh source is δ, the only endpoint-containment source is E, and the only finite grid source is M.

The finite observation and completion-facing reads are carried by W and R. Thus every endpoint bracket and cell row is consumed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ after the dyadic mesh has already been displayed.

The transport row H preserves these rows componentwise, and C replays the same interval-and-mesh request without adding an off-grid point. The provenance and local naming rows only source and name the same finite packet, so the listed obligations exhaust the NameCert surface and exclude the nonlocal objects named in the statement. □

Theorem 7140.3 (CompactIntervalDyadicGrid Real handoff)**.** *Every accepted* $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ *read is consumed through the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{ClosedBoundedInterval}^{\uparrow}.$$

In this route the dyadic mesh row δ and endpoint-containment row E are read before finite stream windows W, the regular-rational and Real handoff row R, and the closed interval source row I are consumed by a Real or Completion-facing client. The route exports a finite endpoint grid and nonescape certificate, not Heine–Borel compactness, open-cover compactness, selected limits, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the dyadic mesh row δ and endpoint-containment row E. They are the only rows that determine the grid precision and endpoint brackets, so the finite mesh row M can only be read after the $\mathsf{DyadicRatCore}^{\uparrow}$ source has been exposed.

Next read the finite observation row W. It binds the endpoint brackets and each grid cell to displayed $\mathsf{StreamName}^{\uparrow}$ windows rather than to an ambient subset of a host real line.

Finally pass through R and I. The row R gives the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff for the endpoint and cell reads, while I supplies the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ boundary consumed by the downstream interval client.

No carrier coordinate remains for an open cover, selected subcover, selected limit, trichotomy, quotient equality, or ambient completeness theorem. The handoff is therefore the displayed finite L10 route and nothing stronger. □

Falsifiable prediction: Every $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ consumer that reads a finite endpoint grid over a closed Real interval must display the closed interval source, dyadic mesh, endpoint-containment row, finite mesh row, finite stream windows, RegSeqRat/Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Independence witness: closedboundedinterval, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ because this packet alone binds a displayed closed Real interval to a finite dyadic endpoint grid, finite observation windows, regular-rational handoff, nonescape replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalDyadicGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalDyadicGrid}^{\uparrow}$ packet is generated from a closed bounded interval source, a DyadicRatCore mesh, endpoint-containment rows, finite mesh rows, StreamName observations, RegSeqRat and Real handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalDyadicGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7140.1, Definition 7140.2, and Definition 7140.3 seed the finite dyadic endpoint-grid packet over ClosedBoundedInterval, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalDyadicGrid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, open-cover compactness, selected limits, trichotomy, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, endpoint containment, RegSeqRat and Real handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.CompactIntervalDyadicGridUp.

Chapter 7141

A Concrete Naming Certificate for $\mathsf{RadonPartition}^{\uparrow}$

Following Radon and standard finite-dimensional convex geometry, the $\mathsf{RadonPartition}^{\uparrow}$ packet records a finite affine-dependence split as BEDC naming data. It sits over the ambient affine carrier of chapter 206, the vector-space dependence rows of chapter 47, the convex membership surface of chapter 208, the finite face and witness rows of chapter 209, the finite support route of chapter 2347, and the finite-intersection route of chapter 2309. The packet names only the displayed support family, signed coefficient ledger, convex-hull intersection witness, transport, replay, provenance, and local naming; it does not assert a host separation theorem, a selected intersection point outside the packet, quotient convex-set equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7141.1 (Radon partition carrier)**.** *A* $\mathsf{RadonPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, V, F, C, P, I, H, T, Q, N)$$

with the following displayed rows:

1. *A is the finite-dimensional* $\mathsf{AffineSpace}^{\uparrow}$ *ambient row through which all point histories and affine combinations are read;*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *translation row that records the vector-dependence read underlying the affine relation;*
3. *F is the finite support family, listing the displayed point histories whose affine dependence is being split;*
4. *C is the signed coefficient ledger, recording positive-side and negative-side coefficient rows, zero-sum balance, and equal normalized total mass;*
5. *P is the partition row, assigning each displayed support history to the positive side, the negative side, or a zero-coefficient boundary row;*
6. *I is the convex-hull intersection witness, stored as finite replay data over the two convex combinations and read through* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$*, and* $\mathsf{HellyTheorem}^{\uparrow}$ *sibling routes;*
7. *H is the componentwise* hsame *transport for ambient, vector, support, coefficient, partition, and intersection rows;*

8. *T is the displayed* Cont *route that replays the affine-dependence split without adding host convexity data;*
9. *Q is the* Pkg *provenance over* $\mathsf{RadonPartition}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{CaratheodoryConvexHull}^{\uparrow}$, $\mathsf{HellyTheorem}^{\uparrow}$, BHist, hsame, Cont, *and local* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RadonPartition}^{\uparrow}}$ *row.*

The carrier exposes no ambient theorem that every large finite point family admits such a split, no quotient equality of convex hulls, no selected host point, and no standard bridge row.

Theorem 7141.2 (Radon partition NameCert obligations)**.** *For the carrier of Definition 7141.1, the local* $\mathsf{NameCert}_{\mathsf{RadonPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the affine ambient row, vector-dependence row, finite support family, signed coefficient ledger, partition row, convex-hull intersection witness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Classifier stability is tested only by replaying the same support family, coefficient balance, side assignment, and intersection witness through transported affine, vector, convex, polytope, Caratheodory, and Helly rows. Ledger exactness says that every accepted Radon-partition read exposes precisely $A, V, F, C, P, I, H, T, Q, N$ and no external convex-geometry datum.*

Theorem 7141.3 (Radon affine-dependence route)**.** *Every accepted* $\mathsf{RadonPartition}^{\uparrow}$ *packet routes a finite convex-geometry consumer through*

$$\mathsf{AffineSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{Polytope}^{\uparrow},\ \mathsf{CaratheodoryConvexHull}^{\uparrow},\ \mathsf{HellyTheorem}^{\uparrow}.$$

The affine and vector rows fix the displayed dependence source, the finite support and signed coefficient rows record the zero-sum balance, the partition row separates the two convex sides, and the intersection witness is readable only after the convex-set, polytope, Caratheodory, and Helly sibling routes have been exposed. A consumer that omits any of those rows is outside the accepted $\mathsf{RadonPartition}^{\uparrow}$ *carrier.*

Closure status for $\mathsf{NameCert}_{\mathsf{RadonPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RadonPartition}^{\uparrow}$ packet is generated from finite-dimensional affine ambient data, vector-dependence data, a finite support family, signed coefficient and partition ledgers, a convex-hull intersection witness, componentwise transport, continuation replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RadonPartition}^{\uparrow}}$, seed)

Scope closed: Definition 7141.1, Definition 7141.2, and Definition 7141.3 seed the finite affine-dependence split, signed-coefficient, convex-hull intersection, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RadonPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host Radon theorem, host separation, selected intersection points, quotient convex-set equality, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, affine-dependence replay, signed-coefficient exactness, partition routing, convex-hull intersection witness, transport, provenance, and local naming for BEDC.Derived.RadonPartitionUp.

Chapter 7142

A Concrete Naming Certificate for $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$

Following Bishop and Bridges constructive interval analysis, the $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$ packet records the finite-window data needed to read a compact real interval through choice-sequence observations without importing open-cover compactness or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits between the located interval routes of chapter 1711 and chapter 6253, the dyadic tolerance core of chapter 1562, the stream-name surface of chapter 126, the regular rational readback of chapter 301, and the real-name boundary of chapter 38. The packet gives downstream L10 Real and Completion consumers a finite interval choice-sequence route over $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Definition 7142.1 (CompactIntervalChoiceSequence carrier)**.** *A* $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed real interval, its endpoint order, and its located membership tests;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the finite dyadic mesh budget used by the interval read;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite choice-sequence prefix of sampled interval names;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational approximants visible in S;*
5. *E is the endpoint-and-enclosure ledger proving that every displayed sample remains inside the located interval named by I with tolerance budget D;*
6. *H records componentwise* hsame *transport for I, D, S, R, E;*
7. *C is the* Cont *replay row for the finite choice-sequence prefix and its dyadic tolerance window;*
8. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for open-cover compactness, selected infinite branches, quotient equality of streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CompactIntervalChoiceSequenceUp.CompactIntervalChoiceSequenceTasteGate single carrier alignment`

Theorem 7142.2 (CompactIntervalChoiceSequence NameCert obligations)**.** *For the carrier of Definition 7142.1, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceSequence}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier exposure for* $K = (I, D, S, R, E, H, C, P, N)$*;*
2. *located interval admission through* I *and its endpoint order;*
3. *dyadic tolerance and regular rational readback through* D, S, R*;*
4. *endpoint and enclosure exactness through* E *and componentwise transport through* H*;*
5. *finite replay, provenance, and naming through* C, P, N*.*

Every accepted compact-interval choice-sequence consumer must read those rows before using the packet. No consumer may replace them by an arbitrary open-cover compactness theorem, a selected infinite branch, quotient stream equality, a hidden choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the displayed carrier. The rows $I, D, S, R, E, H, C, P, N$ are the complete local inventory, so the certificate cannot read an endpoint, tolerance, stream prefix, rational approximant, or enclosure fact outside that finite tuple.

The located interval admission is carried by I. The dyadic and rational data are then read through D, S, R, with S exposing only a finite choice-sequence prefix and R attaching regular rational readback to the observations already visible in that prefix.

Next inspect endpoint and enclosure exactness. The row E binds every displayed sample to the located interval named by I under the tolerance budget in D, and H transports exactly those rows componentwise.

Finally C, P, N replay, source, and name the same finite data. Since no carrier coordinate supplies an open cover, infinite branch selector, quotient stream witness, or completeness theorem for $\mathsf{Real}^{\uparrow}$, the five obligation rows exhaust the local NameCert surface. □

Theorem 7142.3 (CompactIntervalChoiceSequence L10 handoff)**.** *Every L10 Real and Completion consumer that reads a compact interval through* $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$ *factors through the finite chain*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longrightarrow E, H, C, P, N \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff gives the consumer a located real interval, a dyadic tolerance budget, a finite choice-sequence prefix, regular rational readback, endpoint enclosure, componentwise transport, replay, provenance, and local naming. It gives no open-cover compactness theorem, selected infinite branch, quotient stream equality, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the interval and tolerance rows. The L10 consumer receives the located interval only as the displayed row I, and its finite tolerance budget only through D.

The choice-sequence information is then finite. The row S displays the observed prefix, while R gives regular rational readback for the samples already exposed by that prefix.

Use the enclosure and structural rows next. The row E keeps the samples inside the located interval under the dyadic budget, and H, C, P, N transport, replay, source, and name the same finite packet.

Thus the route into $\mathsf{Real}^{\uparrow}$ is exhausted by the displayed chain. Because the carrier contains no open-cover row, branch selector, quotient stream witness, countable-choice coordinate, or completeness proof for $\mathsf{Real}^{\uparrow}$, the handoff cannot export any of those principles. □

Falsifiable prediction: Removing the stream-prefix row or the endpoint-and-enclosure ledger lets the consumer confuse finite interval observations with a selected infinite branch, so the L10 handoff no longer has a finite NameCert surface.

Independence witness: Nearest siblings: located-interval-bisection, located-interval-completion, dyadic-rat-core-arithmetic, stream-name, regseqrat, real. This packet combines a located interval row, dyadic tolerance budget, finite choice-sequence prefix, regular rational readback, and endpoint enclosure ledger as one carrier; each named sibling lacks at least one of those coordinates as a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalChoiceSequence}^{\uparrow}$ packet is generated from a located interval row, dyadic tolerance budget, finite choice-sequence prefix, regular rational readback, endpoint enclosure ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7142.1, Definition 7142.2, and Definition 7142.3 state the finite carrier, local NameCert surface, and L10 handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, quotient streams, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: The next step is a Lean carrier and NameCert obligation surface for the finite interval choice-sequence packet.

Chapter 7143

A Concrete Naming Certificate for $\mathsf{ClosedMetricGraph}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{ClosedMetricGraph}^{\uparrow}$ packet reconstructs sequential graph closure as finite BEDC NameCert data. It sits between the metric-graph source of chapter 6788, the closed-ball boundary of chapter 1451, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names one displayed graph row together with source and target convergence windows; it does not import the closed graph theorem, a Banach-space argument, quotient graph equality, or a choice-selected subsequence.

Definition 7143.1 (Closed metric graph carrier)**.** *A* $\mathsf{ClosedMetricGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, F, S, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric carrier, read through the* $\mathsf{MetricGraph}^{\uparrow}$ *or sibling* $\mathsf{Met}^{\uparrow}$ *source surface.*
2. *Y is the target metric carrier whose endpoint comparisons are read through displayed* $\mathsf{MetricClosedBall}^{\uparrow}$ *rows.*
3. *F is the graph row, naming the finite membership relation between source rows of X and target rows of Y.*
4. *S is the source convergent-window ledger, recording the displayed source approximation windows before any limit packet is read.*
5. *T is the target convergent-window ledger, recording the displayed target approximation windows paired with S.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the target window ledger into regular rational comparison data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the target limit comparison exposed by R.*
8. *H records componentwise* hsame *transport for X, Y, F, S, T, R, E and the structural rows.*
9. *C records displayed* Cont *replay from paired graph rows through the source and target windows to the terminal seal.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedMetricGraph}^{\uparrow}$, $\mathsf{MetricGraph}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricGraph}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient closed subset of a product, quotient equality of graph points, a selected subsequence, a Banach-space closed graph theorem, or a host completeness principle.

Theorem 7143.2 (Closed metric graph sequential closure)**.** *Let $G = (X, Y, F, S, T, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedMetricGraph}^{\uparrow}$ *packet. If the displayed source graph rows converge through S and the displayed target graph rows converge through T, R, E, then the graph membership row F transports through H and replays through C to the limit packet named by the same local* NameCert *row N. The transport uses only the metric source rows, closed-ball target comparisons, regular rational readback, real seal, replay, provenance, and local naming rows already present in G; it does not use host quotient equality, external sequence choice, or ambient closed-subset semantics.*

Lean4 ✓ `BEDC.Derived.ClosedMetricGraphUp.ClosedMetricGraphCarrier sequential closure`

Proof. Project the carrier of Definition 7143.1. The only graph membership source is F, and the only convergence ledgers are the source window S and target window T.

Read the target limit route next. The regular rational readback R and terminal real seal E keep every target comparison inside the displayed $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ surfaces, while the closed-ball boundary keeps the comparison non-strict and packet-local.

Transport is then componentwise. The row H moves the source carrier, target carrier, graph row, source window, target window, readback row, and real seal together, and C replays the same route before the provenance row P and local name N expose the limit packet.

The exclusions follow from the finite inventory. No row of G stores a host product closed subset, a quotient equality test for graph points, a subsequence selector, a Banach-space theorem, or an ambient completeness principle, so the sequential closure read remains the displayed NameCert transport. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedMetricGraph}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedMetricGraph}^{\uparrow}$ packet is generated from source and target metric carriers, a graph membership row, source and target convergent windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedMetricGraph}^{\uparrow}}$, seed)

Scope closed: Definition 7143.1 and Definition 7143.2 seed the finite graph-closure source packet over $\mathsf{MetricGraph}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedMetricGraph}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.ClosedMetricGraphUp`

Bridge status: none

Not claimed: This chapter does not close the full closed graph theorem, any Banach-space theorem, quotient graph equality, choice-selected subsequences, ambient closed-subset semantics, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window convergence, target-window readback, Real seal transport, graph-membership replay, provenance, refusal, and local naming for BEDC.Derived.ClosedMetricGraphUp.

Chapter 7144

A Concrete Naming Certificate for $\mathsf{LocatedApartnessField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedApartnessField}^{\uparrow}$ packet records the ordered-field-facing real interface whose comparisons are read through located apartness data rather than through a trichotomy principle. It consumes the constructive real-line surface of chapter 4041, the located real boundary of **??**, the located apartness boundary of chapter 1380, the positive-order handoff of chapter 1905, and the L10 source route through chapter 291, chapter 126, chapter 301, and chapter 38. The chapter names the finite field-facing packet; it does not claim quotient real equality, selected limits, trichotomy, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7144.1 (Located apartness field carrier)**.** *A* $\mathsf{LocatedApartnessField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, L, A, O, M, I, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for radii, slack comparisons, and field-operation error bounds.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect each real-name input and each operation output.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows into regular rational approximants.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *interval-location row for the same readback surface.*
6. *A is the* $\mathsf{LocatedApartness}^{\uparrow}$ *positive-separation row used when a consumer requests apartness.*
7. *O records the finite addition, negation, multiplication, and unit rows whose outputs are again read through D, S, R, E.*
8. *M records the inverse row, available only with an A-witness separating the input from zero.*
9. *I is the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *comparison handoff from positive apartness data to order-facing finite comparison cells.*
10. *H records componentwise* hsame *transport for $D, S, R, E, L, A, O, M, I, C, P, N$.*

11. *C records displayed* Cont *replay from a field-facing request to the finite tolerance, window, readback, seal, locatedness, apartness, operation, inverse, and order-handoff rows.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedApartnessField}^{\uparrow}$, $\mathsf{ConstructiveRealLine}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedApartnessField}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $D, S, R, E, L, A, O, M, I, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7144.2 (Located apartness field NameCert obligations)**.** *For every accepted* $\mathsf{LocatedApartnessField}^{\uparrow}$ *carrier*

$$F = (D, S, R, E, L, A, O, M, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessField}^{\uparrow}}$ *surface consists exactly of carrier admission; L10 source admission through D, S, R, E; locatedness admission through L; apartness admission through A; operation admission through O; inverse admission through M only under displayed apartness from zero; order-handoff admission through I; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7144.1. The L10 source rows D, S, R, E are visible before any field-facing output can be read.

Read the located and apartness boundary rows next. The row L supplies interval-location data, while A supplies positive separation evidence for apartness-sensitive consumers.

The algebra rows are then forced to remain on the same route. The operation row O reads its inputs and outputs through D, S, R, E, and the inverse row M is available only after the zero-separation evidence is displayed through A.

Finally the order bridge I and the structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7144.3 (Located apartness field real nonescape)**.** *Every field-facing consumer exported by an accepted* $\mathsf{LocatedApartnessField}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, *and* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *boundary rows. The consumer receives a finite located-apartness field interface, not trichotomy, quotient real equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the L10 source route D, S, R, E in Definition 7144.1. Every operation output and every comparison endpoint must pass through the dyadic tolerance ledger, stream windows, regular rational readback, and terminal real seal.

The located and apartness reads remain boundary reads. A location request must use L, a positive-separation request must use A, and an order-facing comparison must use the handoff row I from chapter 1905.

The operation and inverse rows do not add a host real object. They only repack the same inspected rows and, in the inverse case, require displayed apartness from zero.

Transport and replay through H and C, then read provenance and local naming through P and N. The route contains no trichotomy branch, quotient real constructor, selected limit, or completeness witness, so none of those objects can escape through the field-facing interface. □

Falsifiable prediction: Every $\mathsf{LocatedApartnessField}^{\uparrow}$ field-facing read factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: constructive-real-line, locatedreal, located-apartness, real-apartness-order-bridge, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedApartnessField}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries field-operation rows, inverse rows guarded by apartness from zero, locatedness rows, positive-separation rows, order-facing comparison rows, L10 readback rows, transport, replay, provenance, and local naming as one field-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessField}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedApartnessField}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a terminal RealUp seal, LocatedReal interval rows, LocatedApartness positive-separation rows, field operation rows, inverse rows guarded by apartness from zero, RealApartnessOrderBridge comparison handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartnessField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7144.1, Definition 7144.2, and Definition 7144.3 seed the finite located-apartness field interface over ConstructiveRealLineUp, LocatedRealUp, LocatedApartnessUp, RealApartnessOrderBridgeUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedApartnessField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter refuses trichotomy, quotient real equality, selected limits, and ambient Real completeness.

Upgrade path: A formal route can encode the displayed carrier and prove source admission, locatedness and apartness boundaries, field operation closure, inverse guarding by apartness from zero, order-handoff discipline, transport, replay, provenance, and ledger nonescape for BEDC.Derived.LocatedApartnessFieldUp.

Chapter 7145

A Concrete Naming Certificate for $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ packet records the finite binary comparison surface by which two regular Cauchy windows are compared through one displayed dyadic tolerance ledger. It sits beside the finite stream rows of chapter 126, the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the window-step route of chapter 1020. The packet is a finite comparison interface for Real and Completion consumers; it is not quotient stream equality, selected limits, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7145.1 (Cauchy-window comparison principle carrier)**.** *A* $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, D, R_0, R_1, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened for the compared regular Cauchy names.*
2. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested dyadic radius, the two displayed observations, and the arithmetic comparison budget used by the binary read.*
3. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from* S_0, S_1 *through the same dyadic ledger.*
4. E *is the comparison-evidence row consumed before any downstream* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *handoff can be read.*
5. H *records componentwise* hsame *transport for* $S_0, S_1, D, R_0, R_1, E, C, P, N$.
6. C *records the displayed* Cont *replay route from the two stream windows through the dyadic ledger, regular rational readbacks, comparison evidence, transport, provenance, and local naming rows.*
7. P *is the* Pkg *provenance over* $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyWindowTransducer}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these ten rows. It has no coordinate for quotient stream equality, a selected limit, host equality of completed reals, countable choice, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyWindowComparisonPrincipleUp.CauchyWindowComparisonPrincipleTasteGate single carrier alignment`

Theorem 7145.2 (Cauchy-window comparison principle NameCert obligations)**.** *For every accepted* $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ *carrier*

$$K = (S_0, S_1, D, R_0, R_1, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the two displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, comparison-evidence row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *consumer must read the comparison-evidence row only after the two source windows, the dyadic ledger, and both regular rational readbacks have been displayed.*

Proof. Project the carrier of Definition 7145.1. Its visible inventory is precisely $S_0, S_1, D, R_0, R_1, E, H, C, P, N$.

The binary comparison is finite because S_0 and S_1 list the two inspected windows and D records the single dyadic tolerance ledger used to compare the displayed observations. The readback rows R_0 and R_1 are then the regular rational data available to downstream consumers.

The comparison-evidence row E is not a completed real equality. It is only the local row consumed after the stream windows, dyadic ledger, and readbacks have been exposed.

Finally H and C transport and replay the same rows, while P and N bind provenance and local naming. Since no other coordinate is present, the listed rows exhaust the NameCert obligation surface. □

Theorem 7145.3 (Cauchy-window comparison principle L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ *packet receives a binary regular-Cauchy comparison only through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ E, H, C, P, N.$$

If the consumer later calls a $\mathsf{Real}^{\uparrow}$ *row or a* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *row, that call must be supplied by the downstream packet after this route has been displayed. The handoff exports no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, no quotient stream equality, no selected limit, no host equality, and no countable-choice schedule.*

Proof. Start from an accepted packet $K = (S_0, S_1, D, R_0, R_1, E, H, C, P, N)$. The only source rows are the two $\mathsf{StreamName}^{\uparrow}$ windows, and the only tolerance row is the shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

The two $\mathsf{RegSeqRat}^{\uparrow}$ rows are read from those displayed windows through the same ledger. Thus the comparison-evidence row E is already tied to finite window observations before any Real-facing or transducer-facing consumer can use it.

Transport, replay, provenance, and local naming preserve the route but do not add another endpoint. A later $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{CauchyWindowTransducer}^{\uparrow}$ handoff must therefore come from its own downstream packet, and the comparison principle itself exports none of the excluded completion, quotient, selection, host-equality, or choice data. □

Falsifiable prediction: Every accepted $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ read exposes two StreamName finite-window rows, one DyadicRatCore tolerance ledger, two RegSeqRat readback rows, one comparison-evidence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any RealUp or CauchyWindowTransducer handoff is consumed.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchy-window-transducer: $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds two finite windows, one shared dyadic comparison ledger, two regular rational readbacks, comparison evidence, transport, replay, provenance, and local naming while refusing quotient stream equality, selected limits, host equality, countable choice, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7145.1, Definition 7145.2, and Definition 7145.3 seed the finite binary comparison surface over two StreamName windows, one DyadicRatCore ledger, two RegSeqRat readbacks, comparison evidence, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyWindowComparisonPrincipleUp.CauchyWindowComparisonPrincipleTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host equality, countable choice, standard bridges, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic comparison exactness, readback stability for both rows, transducer handoff nonescape, replay, provenance, local naming, and L10 route exhaustion for BEDC.Derived.CauchyWindowComparisonPrincipleUp.

Constructive story: A $\mathsf{CauchyWindowComparisonPrinciple}^{\uparrow}$ packet is generated from two StreamName finite-window rows, one DyadicRatCore tolerance ledger, two RegSeqRat readbacks, comparison evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7146

A Concrete Naming Certificate for $\mathsf{WhitneyCoveringFinite}^{\uparrow}$

Following Whitney's covering decomposition and constructive metric-analysis finite-net practice, the $\mathsf{WhitneyCoveringFinite}^{\uparrow}$ packet records a finite local covering family as BEDC NameCert data. It sits over the metric source of chapter 57, the doubling control of chapter 2035, the finite-net handoff of chapter 1767, the total-bounded consumer surface of chapter 127, and the multiscale dyadic skeleton of chapter 6909. The packet names only the displayed finite covering route; it does not import measure theory, open-cover compactness, selected infinite covers, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7146.1 (WhitneyCoveringFinite carrier)**.** *A* $\mathsf{WhitneyCoveringFinite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, O, D, F, B, S, E, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, recording the carrier, distance-reading route, and local metric naming surface.*
2. *O is the open-region request row, naming the displayed region whose finite Whitney cells are being read.*
3. *D is the dyadic scale ledger, read through the* $\mathsf{DyadicNetTree}^{\uparrow}$ *scale and parent-child rows.*
4. *F is the selected finite cell family, storing only the displayed cells and their attached centers, radii, and scale indices.*
5. *B is the bounded-overlap ledger, recording the finite overlap budget available for the selected cells.*
6. *S is the shadow-coverage row, asserting only the displayed coverage witnesses for the region requested by O.*
7. *E is the finite-epsilon-net handoff row, read through* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *and supplying the center witnesses consumed by F and S.*
8. *T is the total-bounded consumer row, read through* $\mathsf{TotallyBounded}^{\uparrow}$ *and used only as a finite coverage consumer.*

9. *H records componentwise* hsame *transport for the metric source, open-region request, dyadic ledger, selected cells, bounded-overlap ledger, coverage shadow, finite-net handoff, total-bounded consumer row, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite Whitney-covering request through $M, O, D, F, B, S, E, T, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{WhitneyCoveringFinite}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicNetTree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyCoveringFinite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, O, D, F, B, S, E, T, H, C, P, N$. It has no coordinate for a measure algebra, open-cover compactness principle, selected infinite Whitney family, quotient equality of covers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7146.2 (WhitneyCoveringFinite NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyCoveringFinite}^{\uparrow}$ *carrier*

$$W = (M, O, D, F, B, S, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WhitneyCoveringFinite}^{\uparrow}}$ *surface consists exactly of carrier admission through M, open-region request visibility through O, dyadic scale visibility through D, selected finite-cell admission through F, bounded-overlap accounting through B, shadow coverage through S, finite-epsilon-net handoff through E, total-bounded consumer routing through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 7146.1. The metric source is M, the requested open region is O, and the scale information is the dyadic ledger D read through the finite net-tree route.

Next read the finite covering data. The selected cell family F is available only with its displayed centers, radii, and scale indices. The bounded-overlap ledger B and shadow-coverage row S are tied to that same finite family.

The consumer handoff is finite. The row E supplies the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ witnesses, and the row T supplies the $\mathsf{TotallyBounded}^{\uparrow}$ consumer route, with $\mathsf{DoublingMetric}^{\uparrow}$ and $\mathsf{DyadicNetTree}^{\uparrow}$ only serving the displayed scale and overlap controls.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no measure-theoretic coordinate, open-cover compactness principle, infinite-cover selector, quotient-cover equality, or ambient real-completeness row, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{WhitneyCoveringFinite}^{\uparrow}$ read exposes the MetricUp source, open-region request, dyadic scale ledger, selected finite cell family, bounded-overlap ledger, shadow-coverage row, finite-epsilon-net handoff, total-bounded consumer row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Whitney-covering consumer reads the packet.

Independence witness: metric, doublingmetric, finite-epsilon-net, totallybounded, dyadic-net-tree: no listed sibling is bijective with $\mathsf{WhitneyCoveringFinite}^{\uparrow}$ because this packet jointly stores an open-region request, selected finite cells, bounded-overlap ledger, shadow coverage, finite-net handoff, total-bounded consumer row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyCoveringFinite}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyCoveringFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7146.1 and Definition 7146.2 seed the finite Whitney-covering packet over displayed metric, open-region, dyadic-scale, finite-cell, bounded-overlap, shadow-coverage, finite-net, total-bounded, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{WhitneyCoveringFinite}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim measure theory, open-cover compactness, selected infinite covers, quotient equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, finite-cell selection, bounded-overlap accounting, shadow coverage, finite-net handoff, total-bounded routing, transport, replay, provenance, and local naming for BEDC.Derived.WhitneyCoveringFiniteUp.

Constructive story: A WhitneyCoveringFinite$^{\uparrow}$ packet is generated from a MetricUp source row, an open-region request row, a dyadic scale ledger, selected finite cells, a bounded-overlap ledger, a shadow-coverage row, a FiniteEpsilonNetUp handoff, a TotallyBoundedUp consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7147

A Concrete Naming Certificate for $\mathsf{LebesgueNumberModulus}^{\uparrow}$

Bishop–Bridges style compact metric analysis treats a Lebesgue-number argument as a finite-net calculation once the compact source, metric scale, cover membership rows, and real-valued tolerance readbacks are displayed. The $\mathsf{LebesgueNumberModulus}^{\uparrow}$ packet records that finite route for the L10 compact/metric handoff. It sits over the compact finite-net source of chapter 129, the metric row of chapter 57, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite modulus packet only; it does not assert open-cover compactness, selected covers, ambient completeness, or a host Lebesgue-number theorem.

Definition 7147.1 (Lebesgue-number modulus carrier)**.** *A* $\mathsf{LebesgueNumberModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, U, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the consumed* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the displayed net cells and scale bound available to the compact route.*
2. *M is the consumed* $\mathsf{Met}^{\uparrow}$ *row used to read the distances between a requested point and the finite net witnesses.*
3. *U is the finite cover-admission ledger. It records which displayed cover row receives each named finite-net cell, without asserting open-cover compactness.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the positive radius, the mesh comparison, and the dyadic slack used by the handoff.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the metric and cover requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows and dyadic tolerances into regular rational comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the radius and comparison readbacks consumed by downstream uniform-modulus routes.*
8. *H records componentwise* hsame *transport for $K, M, U, D, S, R, E, C, P, N$.*
9. *C records displayed* Cont *replay from the compact finite-net source through metric, cover, dyadic, StreamName, RegSeqRat, and Real rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LebesgueNumberModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, U, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary cover refinement, a selected global subcover, host real completeness, quotient metric equality, Lean verification, or bridge checking.

Theorem 7147.2 (Lebesgue-number modulus NameCert obligations)**.** *For every accepted* LebesgueNumberModulus *carrier*

$$L = (K, M, U, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueNumberModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net source, metric distance row, finite cover-admission ledger, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming row; scale exactness requiring every radius read to pass through D, S, R, E; cover exactness requiring every cover read to pass through U after the compact finite-net row K; and nonescape from the named rows.*

Proof. Project the carrier of Definition 7147.1. The rows $K, M, U, D, S, R, E, H, C, P, N$ are exactly the inventory available to an accepted packet.

The compact route first reads the finite-net row K and the metric row M. The cover ledger U is then attached to those displayed net cells, so a cover read never becomes an external open-cover compactness principle.

The radius route is finite. The dyadic tolerance row D fixes the mesh and slack comparisons, S lists the inspected windows, R performs regular rational readback, and E seals the comparison as a $\mathsf{Real}^{\uparrow}$ row.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no additional carrier coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7147.3 (Lebesgue-number modulus finite-net handoff)**.** *Every L10 compact-uniform or metric-continuity consumer of an accepted* $\mathsf{LebesgueNumberModulus}^{\uparrow}$ *carrier*

$$L = (K, M, U, D, S, R, E, H, C, P, N)$$

must factor its modulus read through the finite route

$$K, M \longmapsto U \longmapsto D, S, R, E \longmapsto H, C, P, N.$$

The consumer may export only the compact finite-net row, metric distance row, finite cover ledger, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, terminal Real seal, transport, replay, provenance, and local naming rows. It may not export open-cover compactness, a selected subcover, a host Lebesgue-number theorem, quotient metric equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the compact finite-net and metric rows. They determine which displayed finite witnesses and distance comparisons may be inspected by a downstream L10 consumer.

Next read the finite cover ledger U. Its role is only to attach displayed cover rows to the finite net cells already exposed by K; it does not choose a cover or prove compactness for arbitrary open covers.

The modulus data are then read through D, S, R, E. Dyadic tolerance, finite StreamName windows, regular rational readback, and the Real seal give a finite positive-radius handoff to the consumer.

The structural rows H, C, P, N preserve and name the same route. Therefore every permitted read factors through the displayed finite-net handoff, and every excluded endpoint would require a carrier coordinate that L does not contain. □

Falsifiable prediction: Every accepted $\mathsf{LebesgueNumberModulus}^{\uparrow}$ read exposes a CompactMetric finite-net row, Metric distance row, finite cover ledger, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before any compact-uniform consumer reads the modulus.

Independence witness: compactmetric, metric, dyadicratcore, streamname, regseqrat, real: $\mathsf{LebesgueNumberModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact finite-net cells, metric distance reads, finite cover admission, dyadic tolerance, finite windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming while refusing open-cover compactness, selected covers, quotient metric equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueNumberModulus}^{\uparrow}$ packet is generated from a CompactMetric finite-net source, Metric distance row, finite cover ledger, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7147.1, Definition 7147.2, and Definition 7147.3 seed the finite Lebesgue-number modulus route over compact metric finite-net, metric, dyadic, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host Lebesgue-number theorems, quotient metric equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scale exactness, cover exactness, finite-net handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.LebesgueNumberModulusUp.

Chapter 7148

A Concrete Naming Certificate for HyperbolicWhitneyShadowCover$^{\uparrow}$

The HyperbolicWhitneyShadowCover$^{\uparrow}$ packet realizes the finite shadow-cover branch of chapter 9463 as concrete NameCert data. It sits between the compactification surface of chapter 7019, the Gromov-boundary rows of chapter 6470, the visual-boundary packet of chapter 4181, the visual-metric rows of chapter 3866, and the boundary-shadow route of chapter 3813. The packet names a finite boundary shadow cover only; the Lean TasteGate side later needs Nontrivial and FieldFaithful instances before it can certify the intended hyperbolic comparison surface.

Definition 7148.1 (HyperbolicWhitneyShadowCover carrier)**.** *A* HyperbolicWhitneyShadowCover$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, V, G, B, T, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the hyperbolic metric source row, recording the displayed space, basepoint, coarse distance route, and local metric naming surface.*
2. *V is the visual-boundary row, read through* HyperbolicVisualBoundary$^{\uparrow}$ *and naming the boundary side visible to the observer.*
3. *G is the* GromovBoundary$^{\uparrow}$ *row, exposing the ray-equivalence and product data used by the visual boundary.*
4. *B is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row, recording the accepted finite shadow window.*
5. *T is the visual-metric tolerance row, read through* HyperbolicVisualMetric$^{\uparrow}$ *and the* Real$^{\uparrow}$ *seal.*
6. *F is the selected finite shadow family, storing only displayed shadows, centers, apertures, and scale indices.*
7. *R is the Real seal row, recording the finite real-valued tolerance read and no ambient real-completeness principle.*
8. *H records componentwise* hsame *transport for the metric source, visual boundary, Gromov boundary, boundary shadow, visual metric tolerance, selected shadow family, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite hyperbolic shadow-cover request through M, V, G, B, T, F, R, H.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}$, $\mathsf{HyperbolicBoundaryCompactification}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, G, B, T, F, R, H, C, P, N$. It has no coordinate for selected infinite shadows, quotient boundary equality, ambient real completeness, measure theory, ergodic density, host group action, or an unsealed visual metric.

Theorem 7148.2 (HyperbolicWhitneyShadowCover NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}$ *carrier*

$$S = (M, V, G, B, T, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}}$ *surface consists exactly of hyperbolic metric admission through M, visual-boundary exposure through V, Gromov-boundary exposure through G, boundary-shadow routing through B, visual-metric tolerance through T, selected finite shadow-family admission through F, Real sealing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7148.1. The hyperbolic metric source is M, while V and G expose the boundary rows before any selected shadow can be read.

Next read the covering data. The boundary-shadow row B and visual-metric tolerance row T admit the shadow windows and tolerance, and the selected finite shadow family F stores only displayed centers, apertures, and scale indices.

The Real seal is explicit. The row R records the finite tolerance read through $\mathsf{Real}^{\uparrow}$, so the packet has a real-valued comparison row without importing ambient real completeness or a host visual-metric theorem.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no selected infinite-shadow family, quotient-boundary equality, measure-theoretic density, ergodic theorem, host group action, or unsealed metric coordinate, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}$ read exposes the gromov-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, hyperbolic-visual-boundary, and real rows before any finite shadow-cover consumer reads the packet.

Independence witness: gromov-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, hyperbolic-visual-boundary, real: no listed sibling is bijective with $\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}$ because this packet jointly stores a hyperbolic metric source, visual boundary, Gromov boundary, boundary shadow, visual-metric tolerance, selected finite shadow family, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7148.1 and Definition 7148.2 seed the finite hyperbolic shadow-cover packet over displayed metric, visual-boundary, Gromov-boundary, boundary-shadow, visual-metric, finite-shadow, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicWhitneyShadowCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite shadows, quotient boundary equality, ambient Real completeness, measure theory, ergodic density, host group action, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, boundary exposure, shadow-window routing, visual-metric tolerance, selected finite shadows, Real sealing, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.HyperbolicWhitneyShadowCoverUp.

Constructive story: A HyperbolicWhitneyShadowCover$^{\uparrow}$ packet is generated from a hyperbolic metric source, visual-boundary row, Gromov-boundary row, boundary-shadow row, visual-metric tolerance row, selected finite shadow family, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7149

A Concrete Naming Certificate for $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$

Following Schoenberg on conditionally positive definite kernels, the $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$ packet records the finite centered-Gram route as BEDC NameCert data. It sits next to the positive finite Gram packet of chapter 6513, but its classifier is centered: coefficient rows must pass through a zero-sum ledger before a quadratic-form read is admissible. The packet consumes the finite matrix readback of chapter 49, the inner-product row of chapter 86, the Hilbert-facing handoff of chapter 91, and the real scalar seal of chapter 38; it does not assert a full Schoenberg embedding, a negative-type completion theorem, quotient functions, countable choice, or ambient Hilbert completeness.

Definition 7149.1 (Conditionally positive kernel carrier)**.** *A* $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, A, Z, Q, M, I, H, R, T, C, P, N)$$

with the following displayed rows:

1. *V is the kernel-value row. It records the finite source list, displayed pairwise kernel values, diagonal reads, and symmetry requests inspected by the packet.*
2. *A is the centered coefficient row. It records the finite coefficient windows whose scalar entries are allowed to test conditional positivity.*
3. *Z is the zero-sum ledger. It records that the displayed coefficient row is centered before any Gram quadratic-form read is admitted.*
4. *Q is the centered Gram quadratic-form row. It records the finite expression $\sum_{i,j} a_i a_j K(x_i, x_j)$ together with the nonnegative comparison requested only after Z is displayed.*
5. *M is the $\mathsf{Mat}^{\uparrow}$ readback row. It records row-column indexing, entry readbacks, centered submatrix projections, and the finite matrix-vector products used by Q.*
6. *I is the $\mathsf{InnerProduct}^{\uparrow}$ handoff row. It records the inner-product readback surface through which a later Hilbert-facing consumer may interpret a centered Gram form.*
7. *H is the $\mathsf{Hilbert}^{\uparrow}$ handoff row. It records only the displayed finite route from centered Gram data toward Hilbert rows, without selecting an ambient feature map.*
8. *R is the $\mathsf{Real}^{\uparrow}$ scalar seal. It records real scalar values, zero-sum comparisons, nonnegative quadratic-form witnesses, and terminal real-name boundaries.*

9. *T records componentwise* hsame *transport for V, A, Z, Q, M, I, H, R.*
10. *C records displayed* Cont *replay for the route*

$$V, A, Z, Q, M, I, H \longmapsto R.$$

11. *P is the* Pkg *provenance over* ConditionallyPositiveKernel$^{\uparrow}$, MooreAronszajnKernel$^{\uparrow}$, Mat$^{\uparrow}$, InnerProduct$^{\uparrow}$, Hilbert$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{ConditionallyPositiveKernel}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $V, A, Z, Q, M, I, H, R, T, C, P, N$ and has no coordinate for uncentered positive-kernel positivity, a selected Hilbert embedding, quotient function equality, negative-type completion, countable choice, or ambient Hilbert completeness.

Theorem 7149.2 (Conditionally positive kernel NameCert obligations)**.** *For the carrier of Definition 7149.1, the local* NameCert$_{\text{ConditionallyPositiveKernel}^{\uparrow}}$ *surface consists exactly of carrier admission, kernel-value exposure, centered coefficient exposure, zero-sum ledger validation, centered Gram quadratic-form exposure,* Mat$^{\uparrow}$ *readback compatibility,* InnerProduct$^{\uparrow}$ *handoff,* Hilbert$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *scalar sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The surface exports no uncentered positive-kernel certificate, selected feature map, quotient function space, countable-choice schedule, negative-type completion, or ambient Hilbert theorem.*

Proof. Carrier admission is projection from Definition 7149.1. The packet displays exactly the finite kernel-value row, centered coefficient row, zero-sum ledger, centered quadratic-form row, matrix readback, inner-product and Hilbert handoff rows, real scalar seal, transport, replay, provenance, and local naming row.

The conditional positivity obligation is centered. The coefficient row A can reach the quadratic-form row Q only through the zero-sum ledger Z, and the scalar comparison at R records nonnegativity only for those displayed centered windows.

The analytic handoff remains local to the sibling certificates. The matrix row supplies finite readback, the inner-product row supplies the readout surface, and the Hilbert row records only the handoff boundary rather than a completed host space.

Finally T and C preserve the same route under hsame transport and Cont replay. Since P and N name no off-packet coordinate, the listed obligations exhaust the local NameCert surface and none of the excluded Hilbert or negative-type objects can be read from the packet. □

Theorem 7149.3 (Conditionally positive kernel centered Gram route)**.** *Every centered-Gram read of an accepted* ConditionallyPositiveKernel$^{\uparrow}$ *packet factors through the kernel-value row V, the centered coefficient row A, the zero-sum ledger Z, the centered quadratic-form row Q, the* Mat$^{\uparrow}$ *readback row M, the* InnerProduct$^{\uparrow}$ *and* Hilbert$^{\uparrow}$ *handoff rows I, H, the* Real$^{\uparrow}$ *scalar seal R,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The route is distinct from chapter 6513: it requires zero-sum centered coefficients before positivity is read, and it does not export full positive finite Gram rows or RKHS completion.*

Proof. Project the carrier rows. The only route from displayed kernel values to a real scalar comparison is

$$V, A, Z, Q, M, I, H \longmapsto R.$$

The zero-sum ledger is therefore unavoidable. An uncentered coefficient row has no path to Q, and the matrix readback row M exposes only the finite entries and products already attached to the centered window.

The sibling handoff is also finite. Rows I and H point to the inner-product and Hilbert surfaces, while R seals the scalar comparison; none of these rows selects a Hilbert embedding or a completed reproducing-kernel object.

Transport, replay, provenance, and local naming preserve that centered route. Thus every accepted read remains inside the centered finite Gram packet and the Moore–Aronszajn positive-kernel surface remains a sibling rather than a duplicate. □

Falsifiable prediction: Every accepted $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$ consumer that reads a centered quadratic-form comparison must first expose the centered coefficient row, zero-sum ledger, Matrix readback, InnerProduct and Hilbert handoff rows, Real scalar seal, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert row.

Independence witness: moore-aronszajn-kernel, matrix, innerproduct, hilbert, real: $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries centered coefficient windows, a zero-sum ledger, a centered Gram quadratic-form row, Matrix readback, InnerProduct and Hilbert handoff rows, a Real scalar seal, transport, replay, provenance, and local naming as one finite conditional-positivity certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ConditionallyPositiveKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{ConditionallyPositiveKernel}^{\uparrow}$ packet is generated from finite kernel-value rows, centered coefficient rows, a zero-sum ledger, centered Gram quadratic-form rows, Matrix readback, InnerProduct and Hilbert handoff rows, Real scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConditionallyPositiveKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7149.1, Definition 7149.2, and Definition 7149.3 seed the finite centered-Gram certificate for conditionally positive kernels.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConditionallyPositiveKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full Schoenberg embedding, negative-type completion, selected feature maps, quotient function spaces, countable choice, ambient Hilbert completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, centered coefficient exposure, zero-sum ledger validation, centered Gram quadratic-form readback, Matrix readback, InnerProduct and Hilbert handoff, Real scalar sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConditionallyPositiveKernelUp.ConditionallyPositiveKernelNameCertObligat

Chapter 7150

A Concrete Naming Certificate for $\mathsf{ContractionMappingPicard}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{ContractionMappingPicard}^{\uparrow}$ packet records the finite route from displayed contraction data through Picard iteration into completion-facing regular-real seals. It sits over the contraction-map surface of chapter 1418, the finite iterate ledger of chapter 2359, the complete-metric service row of chapter 128, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the route that a constructive fixed-equation consumer may inspect; it does not select a fixed point, quotient Picard streams, use countable choice, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, or supply host equality of maps.

Definition 7150.1 (ContractionMappingPicard carrier)**.** *A* $\mathsf{ContractionMappingPicard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (K, I, C_m, R, E, H, Q, S, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *source row, including the metric carrier, graph-exact self-map, rational bound below one, and one-step contraction surface.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *finite orbit row generated from the displayed initial point by repeated reads of the same contraction graph.*
3. *C_m is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, kept separate from the contraction packet and consumed only after the finite Picard ledger is visible.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the Picard residual and tail-modulus route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational readback has been displayed.*
6. *H stores componentwise* hsame *transport for the contraction, Picard, complete-metric, regular-rational, real-seal, replay, provenance, and local naming rows.*
7. *Q stores displayed* Cont *replay from contraction admission through Picard iteration, complete-metric service, regular-rational readback, and real-seal exposure.*
8. *S is the* Pkg *provenance over* $\mathsf{ContractionMappingPicard}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContractionMappingPicard}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected fixed point, quotient equality of Picard streams, countable-choice schedules, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge data.*

Theorem 7150.2 (ContractionMappingPicard NameCert obligations)**.** *For an accepted* $\mathsf{ContractionMappingPicard}^{\uparrow}$ *carrier* $P = (K, I, C_m, R, E, H, Q, S, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContractionMappingPicard}^{\uparrow}}$ *surface consists exactly of contraction-source admission, finite Picard-iteration visibility, complete-metric service separation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback visibility,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming for the carrier of Definition 7150.1.*

Proof. Project the carrier rows. The contraction source K supplies the graph-exact self-map, rational contraction bound, and one-step Picard comparison through chapter 1418.

Read the finite iteration row next. The $\mathsf{PicardIteration}^{\uparrow}$ sibling supplies only displayed finite iterates, residual comparisons, and a regular-rational handoff route; it does not supply an endpoint or a quotient stream.

The complete-metric row C_m remains a separate service coordinate. It may be consumed only after the contraction and Picard rows are visible, and it does not turn the contraction packet itself into an ambient completeness theorem.

Finally inspect R, E, H, Q, S, N. These rows read the regular-rational tail, expose the real seal, transport and replay the same route, record provenance, and name the local packet. Since no listed row is a selector, quotient, choice schedule, host equality proof, or ambient real-completeness coordinate, the displayed rows exhaust the local NameCert surface. □

Theorem 7150.3 (ContractionMappingPicard fixed-equation route)**.** *Every fixed-equation consumer of an accepted* $\mathsf{ContractionMappingPicard}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $K, I, C_m, R, E, H, Q, S, N$ *before it may cite endpoint-facing data. The route exposes contraction data, Picard iteration, complete-metric service, regular-rational readback, and a terminal real seal boundary; it does not assert that a selected fixed point exists, that two Picard streams are quotient-equal, that countable choice is available, or that* $\mathsf{Real}^{\uparrow}$ *is ambiently complete.*

Proof. Start from the contraction row K. The consumer first reads the accepted $\mathsf{ContractionMapping}^{\uparrow}$ surface: metric carrier, graph-exact self-map, rational contraction bound, and adjacent Picard comparison data.

Next apply the Picard iteration row I. The iteration sibling turns the contraction data into a finite orbit ledger and residual-distance route, so the consumer receives only displayed finite approximants and their regular-tail budget.

Now read C_m and R. The complete-metric service is separate from the contraction source, and the regular-rational handoff is downstream of the Picard residual route. Thus completion-facing information enters only through the named service and readback rows.

Finally expose the terminal seal and structural rows E, H, Q, S, N. They preserve the same route and mark where real-facing consumers may read a seal after the regular handoff. Because no row selects an endpoint, quotients streams, invokes countable choice, imports host equality, or asserts ambient $\mathsf{Real}^{\uparrow}$ completeness, the fixed-equation consumer receives exactly the displayed route and no stronger theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractionMappingPicard}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractionMappingPicard}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7150.1, Definition 7150.2, and Definition 7150.3 seed the finite route from contraction data through Picard iteration, complete-metric service, regular-rational readback, and real-seal exposure.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractionMappingPicard}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected fixed point, quotient equality of Picard streams, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-source admission, Picard-iteration visibility, complete-metric service separation, RegSeqRat readback visibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContractionMappingPicardUp.

Constructive story: A $\mathsf{ContractionMappingPicard}^{\uparrow}$ packet is generated from a ContractionMapping source row, PicardIteration finite orbit row, CompleteMetric service row, RegSeqRat readback, RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7151

A Concrete Naming Certificate for $\mathsf{MittagLefflerBishop}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley on inverse-limit compactness, the $\mathsf{MittagLefflerBishop}^{\uparrow}$ packet records the finite-window surface of a Mittag–Leffler stabilization argument without quotient completions or selected limit representatives. It sits beside the nested interval carrier of chapter 320, the closed bounded interval carrier of chapter 1591, the dyadic arithmetic core of chapter 291, the stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the displayed finite stabilization rows consumed by L10 completion readers.

Definition 7151.1 (Mittag–Leffler Bishop carrier)**.** *A* $\mathsf{MittagLefflerBishop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, W, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested closed-window row, read through* $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*, naming the finite prefix of closed nonempty windows under inspection.*
2. *W is the finite cofinal-window ledger recording which later windows are required to meet each displayed earlier window.*
3. *M is the mesh-budget row, with dyadic radii read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and ordered by the local decrease ledger.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations exposed by the stabilization request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S through the dyadic mesh budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, W, M, S, R have been displayed.*
7. *H records componentwise* hsame *transport for $I, W, M, S, R, E, C, P, N$.*
8. *C records the displayed* Cont *replay route from nested windows through mesh budgets, StreamName windows, RegSeqRat readback, the Real seal, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{MittagLefflerBishop}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MittagLefflerBishop}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of completions, a countable-choice schedule, a selected point in the inverse limit, host equality of completed reals, or an ambient compactness theorem.

Lean4 (statement) `BEDC.Derived.MittagLefflerBishopUp`

Theorem 7151.2 (Mittag–Leffler Bishop NameCert obligations)**.** *For every accepted* $\mathsf{MittagLefflerBishop}^{\uparrow}$ *carrier*

$$K = (I, W, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MittagLefflerBishop}^{\uparrow}}$ *surface consists exactly of carrier admission for the nested closed windows, stability of the displayed cofinal-window ledger, preservation of nonempty windows along the displayed prefix, dyadic mesh decrease,* $\mathsf{StreamName}^{\uparrow}$ *exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may replace these rows by countable choice, quotient equality, host completion equality, or selected limit representatives.*

Proof. Project the carrier of Definition 7151.1. The accepted packet exposes exactly $I, W, M, S, R, E, H, C, P, N$.

The finite-window obligations are the rows that can be read from that projection. The row I supplies the nested closed-window prefix, W supplies the stabilization ledger, and M supplies the decreasing dyadic mesh budget. Nonempty-window preservation is read only along the displayed prefix.

The rows S and R then provide the finite observation and regular rational readback used by the completion-facing consumer. The row E is a Real seal reached after those rows, so it cannot detach from the displayed route.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for countable choice, quotient equality, host completion equality, or a selected limit representative, none of those objects is part of the local NameCert surface. □

Theorem 7151.3 (Mittag–Leffler Bishop L10 handoff)**.** *Every L10 completion consumer of an accepted* $\mathsf{MittagLefflerBishop}^{\uparrow}$ *packet must read the route*

$$\mathsf{NestedInterval}^{\uparrow} \text{ or } \mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $I, W, M, S, R, E, H, C, P, N$ *before any terminal seal is used. The handoff exports no quotient completion, no selected inverse-limit point, no countable-choice schedule, and no host equality of completed reals.*

Proof. Let a completion consumer request the terminal seal from an accepted packet K. The only interval source is the displayed row I, and the only mesh source is the dyadic row M.

The finite observations are therefore obtained through S, and their regular rational readback is the row R. The seal row E occurs after those rows in the carrier definition, so a terminal Real read cannot bypass the nested-window, dyadic, stream, and regular-rational route.

Transport, replay, provenance, and local naming preserve that same route without adding a selected endpoint. Thus every L10 completion consumer receives only the displayed finite handoff and none of the excluded quotient, choice, selected-representative, or host-equality data. □

Lean4 ✓ `BEDC.Derived.MittagLefflerBishopUp.MittagLefflerBishopTasteGate single carrier alignment`

Falsifiable prediction: Every accepted $\mathsf{MittagLefflerBishop}^{\uparrow}$ read exposes nested closed-window data, a cofinal-window ledger, a dyadic mesh budget, StreamName windows, RegSeqRat readback,

a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing terminal seal is consumed.

Independence witness: nestedinterval, closedboundedinterval, dyadicratcore, streamname, regseqrat, real: $\mathsf{MittagLefflerBishop}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds nested closed windows, cofinal-window stabilization, mesh decrease, finite observations, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming while refusing countable choice, quotient equality, host completion equality, and selected limit representatives.

Closure status for $\mathsf{NameCert}_{\mathsf{MittagLefflerBishop}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MittagLefflerBishop}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7151.1, Definition 7151.2, and Definition 7151.3 seed the finite Mittag–Leffler stabilization surface over nested windows, dyadic mesh budgets, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MittagLefflerBishop}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MittagLefflerBishopUp.MittagLefflerBishopTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected inverse-limit points, countable choice, host equality of completed reals, standard bridges, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window stability, nonempty-window preservation, dyadic mesh decrease, StreamName exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, local naming, and L10 route exhaustion for BEDC.Derived.MittagLefflerBishopUp.

Constructive story: A $\mathsf{MittagLefflerBishop}^{\uparrow}$ packet is generated from nested closed windows, a cofinal-window ledger, dyadic mesh budgets, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7152

A Concrete Naming Certificate for $\mathsf{MittagLeffler}^{\uparrow}$

Following Mittag–Leffler and standard constructive complex analysis, the $\mathsf{MittagLeffler}^{\uparrow}$ packet records the finite principal-part and approximation-window surface by which a meromorphic prescription can be read without importing host meromorphic-function theory. It sits beside the complex carrier of chapter 39, the holomorphic source of chapter 68, the argument-principle meromorphic boundary of chapter 7193, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed finite rows for pole loci, local principal parts, approximation windows, rational error budgets, analytic replay, transport, provenance, and local naming.

Definition 7152.1 (Mittag–Leffler carrier)**.** *A* $\mathsf{MittagLeffler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Z, U, A, E, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Z is the finite pole-locus row, listing the inspected* $\mathsf{Complex}^{\uparrow}$ *centres and separation witnesses for the poles visible to the packet.*
2. *U is the approximation-window row, listing the finite compact windows on which the partial meromorphic expression is tested.*
3. *A is the local principal-part row, assigning to each displayed centre a finite Laurent tail with a bounded negative-degree range.*
4. *E is the finite error-budget row, recorded in* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances and attached to the displayed windows.*
5. *Q is the* $\mathsf{Holomorphic}^{\uparrow}$ *correction row on the complement of the displayed pole loci inside the inspected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximants and error budgets seen on the displayed windows.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for scalar error reads obtained after Z, U, A, E, Q, R have been exposed.*
8. *H records componentwise* hsame *transport for Z, U, A, E, Q, R, L.*

9. *C records the displayed* Cont *replay route from complex centres and holomorphic correction rows through approximation windows, rational readback, real sealing, transport, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{MittagLeffler}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MittagLeffler}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a global meromorphic-function object, analytic continuation outside the inspected windows, a selected infinite pole enumeration, countable choice, quotient equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7152.2 (Mittag–Leffler NameCert obligations)**.** *For every accepted* $\mathsf{MittagLeffler}^{\uparrow}$ *carrier* $M = (Z, U, A, E, Q, R, L, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{MittagLeffler}^{\uparrow}}$ *surface consists exactly of pole-locus admission, approximation-window admission, finite principal-part admission, dyadic error-budget exposure, holomorphic-correction exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer receives no host meromorphic-function object, no analytic-continuation oracle, no selected infinite pole enumeration, no countable-choice schedule, no quotient equality of functions, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 7152.1. The visible analytic rows are precisely Z, U, A, E, Q, R, L, and the structural rows are H, C, P, N.

The principal-part obligations come from Z and A. The pole-locus row fixes the centres under inspection, while the principal-part row supplies only finite negative-degree tails at those centres. No further pole or coefficient can be obtained without a displayed row.

The approximation obligations come from U, E, Q, R, L. The windows in U determine where the correction row Q is read, the dyadic row E fixes the allowed error budgets, and the regular rational row R is the only readback route before the terminal real seal L.

Transport, replay, provenance, and local naming preserve this finite route. Since the carrier contains no coordinate for host meromorphic functions, analytic continuation outside the inspected windows, selected infinite pole enumerations, countable choice, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects belongs to the local NameCert surface. □

Theorem 7152.3 (Mittag–Leffler RegSeqRat and Real handoff)**.** *Every completion-facing or contour-facing consumer of an accepted* $\mathsf{MittagLeffler}^{\uparrow}$ *packet must read the route*

$$\mathsf{Complex}^{\uparrow},\ \mathsf{Holomorphic}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $Z, U, A, E, Q, R, L, H, C, P, N$ before it treats the packet as a meromorphic approximation boundary. The handoff exports finite principal parts and inspected approximation windows; it exports no global meromorphic-function object, no unbounded pole search, no selected limiting function, no quotient equality of analytic functions, and no ambient completeness claim.

Proof. Let a consumer request the scalar error seal or contour-facing boundary from an accepted packet M. The complex source is displayed by the pole-locus row Z, and the holomorphic correction is displayed by Q only over the windows named in U.

The error comparison cannot be read before E and R are present. The row E supplies the finite dyadic tolerances, and R converts the visible approximation data into regular rational readback. The terminal real row L is downstream of those rows.

The structural rows H, C, P, N then transport, replay, source, and name exactly this route. Thus the consumer receives a finite approximation-window handoff through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, not a host meromorphic object or a selected limit outside the displayed carrier. □

Falsifiable prediction: Every accepted $\mathsf{MittagLeffler}^{\uparrow}$ read exposes pole loci, finite principal parts, approximation windows, dyadic error budgets, holomorphic correction rows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any contour-facing or completion-facing consumer uses the packet.

Independence witness: complex, holomorphic, argument-principle, regseqrat, real: $\mathsf{MittagLeffler}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite pole loci, principal-part tails, approximation windows, dyadic budgets, holomorphic correction, regular rational readback, Real sealing, transport, replay, provenance, and local naming while refusing host meromorphic functions, analytic-continuation oracles, selected infinite pole enumerations, countable choice, quotient equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{MittagLeffler}^{\uparrow}}$.

Constructive story: A $\mathsf{MittagLeffler}^{\uparrow}$ packet is generated from displayed Complex pole centres, finite principal-part tails, inspected approximation windows, dyadic error budgets, Holomorphic correction rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MittagLeffler}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7152.1, Definition 7152.2, and Definition 7152.3 seed the finite Mittag–Leffler NameCert surface over principal parts, approximation windows, RegSeqRat readback, and Real-facing seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MittagLeffler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global meromorphic-function existence, analytic continuation outside inspected windows, selected infinite pole enumerations, quotient equality of analytic functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite pole-locus admission, principal-part exactness, approximation-window coverage, dyadic error-budget exposure, holomorphic correction routing, RegSeqRat readback, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.MittagLefflerUp.

Chapter 7153

A Concrete Naming Certificate for $\mathsf{Hausdorffization}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on separated reflection and constructive completion, the $\mathsf{Hausdorffization}^{\uparrow}$ packet records the finite non-quotient boundary that turns a pseudometric name into the separated metric route consumed by completion. It sits between the separated-reflection surface of chapter 4151, the pseudometric source of chapter 1614, the separated metric consumer of chapter 1411, and the completion route of chapter 1506. Its readback is exposed through the finite stream windows of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names the constructive separated-reflection boundary; it does not assert host quotient equality, representative selection, ambient Hausdorff completion, trichotomy, countable choice, or a completed Real object.

Definition 7153.1 (Hausdorffization carrier)**.** *A* $\mathsf{Hausdorffization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (P, S, M, C, W, R, E, T, K, G, N)$$

with the following displayed rows:

1. *P is the consumed* $\mathsf{Pseudometric}^{\uparrow}$ *source row, including the displayed zero-distance witness surface and finite distance comparisons.*
2. *S is the* $\mathsf{SeparatedReflection}^{\uparrow}$ *boundary row that records which source names are identified by displayed zero-distance evidence.*
3. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *target row available only after the separated-reflection boundary has been read.*
4. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer handoff row, recording that completion can read only the separated metric route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which pseudometric and separated-reflection observations are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite distance and zero-distance windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed distance readbacks.*
8. *T is componentwise* hsame *transport for $P, S, M, C, W, R, E, K, G, N$.*
9. *K is displayed* Cont *replay of the same boundary route.*

10. *G is the* Pkg *provenance over* Hausdorffization$^{\uparrow}$, SeparatedReflection$^{\uparrow}$, Pseudometric$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{Hausdorffization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, S, M, C, W, R, E, T, K, G, N$. It has no coordinate for quotient-set equality, selected representatives, ambient Hausdorff reflection, host metric completion, trichotomy, countable choice, or external Real completeness.

Lean4 (definition) `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier`

Theorem 7153.2 (Hausdorffization NameCert obligations)**.** *For every accepted* Hausdorffization$^{\uparrow}$ *carrier*

$$H = (P, S, M, C, W, R, E, T, K, G, N),$$

the local NameCert$_{\text{Hausdorffization}^{\uparrow}}$ *surface consists exactly of carrier admission for the pseudometric source, separated-reflection boundary, separated metric target row, metric-completion consumer handoff, finite StreamName windows, RegSeqRat readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming; zero-distance exactness requiring every separated identification to pass through P, S, W, R, E; completion-boundary exactness requiring every completion read to pass through M, C after the separated reflection; and nonescape from the named rows.*

Proof. Project the carrier of Definition 7153.1. Its rows are precisely $P, S, M, C, W, R, E, T, K, G, N$, so carrier admission supplies the full local inventory.

The zero-distance route begins in the pseudometric row P. A separated identification is admitted only when the boundary row S displays the corresponding finite evidence, and that evidence is inspected through the windows W, read back by R, and sealed by E.

The separated metric and completion handoff are downstream. The row M is exposed only after the separated-reflection boundary, and the completion row C can consume only that separated metric route.

Transport, replay, provenance, and local naming are the rows T, K, G, N. Because no further carrier coordinate exists, the listed obligations exhaust the NameCert$_{\text{Hausdorffization}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.HausdorffizationCarrier namecert obligations`

Theorem 7153.3 (Hausdorffization separated-reflection boundary)**.** *Every* SeparatedMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer of an accepted* Hausdorffization$^{\uparrow}$ *packet factors through the finite route*

$$P \longmapsto S \longmapsto M, C \longmapsto W, R, E \longmapsto T, K, G, N.$$

The consumer may read pseudometric zero-distance evidence, the separated-reflection boundary, the separated metric target row, the completion handoff, StreamName windows, RegSeqRat readback, a terminal Real seal, transport, replay, provenance, and local naming. It may not read host quotient equality, a selected representative, an ambient Hausdorff reflection object, a quotient completion, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Start with the pseudometric source row P. The only equality-like information available to a consumer is the displayed zero-distance evidence carried there.

The separated-reflection boundary S records how that evidence is admitted by the BEDC packet. It is a finite boundary row, not a quotient set or a selector for representatives.

The separated metric and completion-facing rows are then M and C. They consume the boundary already displayed by S, so no completion reader can bypass the separated-reflection route.

Finally W, R, E, T, K, G, N give the window, readback, seal, transport, replay, provenance, and naming surface for the same packet. The absence of quotient, selector, trichotomy, choice, and ambient-completeness coordinates forces every consumer to remain on the displayed route. □

Lean4 ✓ `BEDC.Derived.HausdorffizationCarrier separated reflection boundary`

Theorem 7153.4 (Hausdorffization ledger non-quotient boundary)**.** *An accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet records a separated-reflection ledger, not a host quotient. Any downstream read by* $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *must retain the same finite rows*

$$P, S, M, C, W, R, E, T, K, G, N.$$

In particular, the read cannot collapse source names by host equality, choose representatives, import quotient-real equality, appeal to Dedekind completion, select an infimum, use countable choice, or invoke ambient completeness.

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier ledger nonquotient boundary`

Proof. The carrier definition gives one finite packet and no quotient carrier. Thus a downstream read must start from the displayed pseudometric row P and the separated-reflection ledger S.

The metric and completion rows M, C are consumer-facing handoffs from that ledger. They do not add a host quotient, selected representative, or completed object.

The analytic readback rows W, R, E expose finite windows, regular rational readback, and the terminal Real seal for the same displayed comparisons. Transport and replay through T, K preserve the packet rather than enlarging it.

The provenance and naming rows G, N name the dependencies and the local certificate. Since quotient equality, Dedekind completion, selected infimum, countable choice, and ambient completeness are absent from the packet, no downstream consumer can recover them from the ledger. □

Theorem 7153.5 (Hausdorffization separated-reflection scope)**.** *For every accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet* $H = (P, S, M, C, W, R, E, T, K, G, N)$*, the separated-reflection read is scoped by the finite rows*

$$\mathsf{SeparatedReflection}^{\uparrow}, \mathsf{Pseudometric}^{\uparrow}, \mathsf{SeparatedMetric}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{h}$$

The row S may compare pseudometric names only through displayed zero-distance evidence from P, finite stream windows W, regular-rational readback R, and the terminal real seal E. It cannot expose host quotient equality, representatives, an external separated reflection, or a completion object outside M, C, T, K, G, N.

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated reflection scope`

Proof. Inspect the carrier rows of Definition 7153.1. The separated-reflection row S is present only as part of the same BHist packet as the pseudometric source P and the downstream separated metric and completion rows M, C.

The evidence read by S is finite. It is opened through W, read through R, and sealed by E, so the route has the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ witnesses before any separated-metric consumer can read it.

Transport, replay, provenance, and local naming are exactly T, K, G, N. Since no carrier coordinate names a quotient object, representative selector, or external completion, the separated-reflection scope is exhausted by the listed rows. □

Theorem 7153.6 (Hausdorffization completion consumer route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet must first read the route*

$$P \longmapsto S \longmapsto M \longmapsto C$$

and then authenticate it through W, R, E, T, K, G, N*. Thus completion-facing data are visible only after the* $\mathsf{Pseudometric}^{\uparrow}$ *source,* $\mathsf{SeparatedReflection}^{\uparrow}$ *boundary,* $\mathsf{SeparatedMetric}^{\uparrow}$ *target row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *carrier,* hsame *transport,* Cont *replay,* Pkg *provenance, and* $\mathsf{NameCert}$ *naming rows have been inspected.*

Proof. Begin with a completion consumer. The carrier gives it no direct completion coordinate; its only completion-facing row is C, and C is recorded as a handoff from the separated metric row M.

The row M is itself downstream of the separated-reflection ledger S. The ledger is admitted from the pseudometric source P by finite zero-distance observations, so the completion route cannot start after S or bypass P.

The analytic readback rows W, R, E and the structural rows T, K, G, N authenticate the same path. Non-quotient ledger exclusion in Definition 7153.4 rules out quotient completion, selected representatives, countable choice, and ambient real completeness as alternate completion inputs. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier completion consumer route`

Theorem 7153.7 (Hausdorffization NameCert scope package)**.** *The local* $\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}$ *package for an accepted packet* $H = (P, S, M, C, W, R, E, T, K, G, N)$ *is exactly the finite scope package over the rows*

$$P, S, M, C, W, R, E, T, K, G, N.$$

It packages carrier admission, separated-reflection evidence, separated-metric handoff, completion-consumer handoff, stream-window inspection, regular-rational readback, real sealing, component-wise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming, with no quotient equality or representative-selection clause.*

Proof. Project the local certificate row N. Its admissible dependencies are named by the provenance row G and are precisely the carrier rows displayed in Definition 7153.1.

Carrier admission and separated-reflection evidence are the rows P, S. The separated metric and completion handoffs are M, C, and their finite readback is supplied by W, R, E.

The remaining authentication data are T, K, G, N. These rows transport, replay, source, and name the same finite package. Because the ledger theorem excludes quotient and representative data, the NameCert package has no further clause beyond the displayed scope. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier namecert scope package`

Theorem 7153.8 (Hausdorffization zero-distance window determinacy)**.** *Let* $H = (P, S, M, C, W, R, E, T, K, G, N)$ *be an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet. Any two separated-identification reads that share the* $\mathsf{Pseudometric}^{\uparrow}$ *row* P*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *determine the same separated-reflection boundary* S *before the separated metric row* M *or completion row* C *is visible.*

Proof. Project the carrier of Definition 7153.1. The separated-reflection row S is the only boundary row between the pseudometric source P and the downstream rows M, C.

Use the separated-reflection scope of Definition 7153.5. It states that S compares source names only through zero-distance evidence from P, finite windows W, regular-rational readback R, and the Real seal E.

Finally apply the ledger non-quotient boundary of Definition 7153.4. Since there is no quotient-equality or representative-selection coordinate, two reads sharing P, W, R, E have no alternate boundary data to distinguish them before M, C are opened. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier zero distance window determinacy`

Theorem 7153.9 (Hausdorffization zero-distance exactness)**.** *Let $H = (P, S, M, C, W, R, E, T, K, G, N)$ be an accepted* Hausdorffization$^{\uparrow}$ *packet. The zero-distance evidence displayed by P, W, R, E determines exactly the separated-reflection row S: every separated identification read by S must be witnessed by the pseudometric row P, the finite window row W, the regular-rational readback row R, and the terminal real seal E, and no downstream row M, C, T, K, G, N can add a further separated-identification clause.*

Proof. Use Definition 7153.8. It states that shared P, W, R, E data determine the same boundary row S before M or C is visible.

The separated-reflection scope theorem, Definition 7153.5, supplies the converse route. The row S compares pseudometric names only through zero-distance evidence from P, finite windows W, regular-rational readback R, and the terminal seal E.

Finally apply Definition 7153.4. Since the carrier has no quotient-equality, representative-selection, trichotomy, choice, or ambient-completeness coordinate, the downstream rows M, C, T, K, G, N authenticate the same boundary rather than extending it. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationZeroDistanceExactness`

Theorem 7153.10 (Hausdorffization completion route stability)**.** *Let $H = (P, S, M, C, W, R, E, T, K, G, N)$ be an accepted* Hausdorffization$^{\uparrow}$ *packet. Every completion consumer preserves the route*

$$P \longmapsto S \longmapsto M \longmapsto C$$

under componentwise hsame *transport T and displayed* Cont *replay K. Thus completion-facing reads may transport or replay the finite route, but they cannot reorder it, bypass S, or replace it by quotient completion data.*

Proof. Begin with Definition 7153.6. It forces each MetricCompletion$^{\uparrow}$ consumer to read $P \longmapsto S \longmapsto M \longmapsto C$ and then authenticate the same path through W, R, E, T, K, G, N.

Transport is componentwise. The row T can identify corresponding pseudometric, separated-reflection, separated-metric, completion, readback, provenance, and naming rows, but it has no coordinate that turns a later completion row into an earlier source row.

Replay is displayed by K. It repeats the same boundary route through the finite packet, and Definition 7153.13 rules out any completion-facing factorization other than $P \longmapsto S \longmapsto M \longmapsto C$. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCompletionRouteStability`

Theorem 7153.11 (Hausdorffization separated-reflection nonescape)**.** *For an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet* $H = (P, S, M, C, W, R, E, T, K, G, N)$*, every separated-reflection or completion-facing read is exhausted by the rows* $P, S, M, C, W, R, E, T, K, G, N$*. In particular, quotient equality, representative selection, trichotomy, countable choice, host separated-reflection objects, quotient completion, and ambient completeness cannot escape the packet as readable data.*

Proof. The NameCert scope package of Definition 7153.7 identifies the local certificate with the finite rows $P, S, M, C, W, R, E, T, K, G, N$. The separated-reflection boundary theorem Definition 7153.3 and completion handoff exactness theorem Definition 7153.12 place every separated metric and completion consumer inside that inventory.

Apply zero-distance exactness, Definition 7153.9. The separated-reflection row is fixed by displayed zero-distance evidence, so there is no remaining boundary slot for host quotient equality, representative choice, or trichotomy.

Apply completion route stability, Definition 7153.10. Completion consumers preserve the finite $P \longmapsto S \longmapsto M \longmapsto C$ route under T and K, so quotient completion, countable choice, and ambient completeness are not readable exports of the packet. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationSeparatedReflectionNonescape`

Theorem 7153.12 (Hausdorffization completion handoff exactness)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet receives the completion row* C *only after the rows* $P, S, M, W, R, E, T, K, G, N$ *have been exposed. Thus completion-facing data are exact over the displayed pseudometric source, separated-reflection boundary, separated metric row, finite windows, regular readback, Real seal, componentwise transport, replay, provenance, and local naming.*

Proof. Start with the completion consumer route of Definition 7153.6. It forces a completion consumer to read $P \longmapsto S \longmapsto M \longmapsto C$ and then authenticate that route through W, R, E, T, K, G, N.

The NameCert scope package of Definition 7153.7 records the same finite inventory as the local certificate. Therefore the row C is not an independent completion object; it is the handoff row reached after the displayed boundary and readback rows.

The non-quotient ledger theorem, Definition 7153.4, excludes quotient completion, selected representatives, countable choice, and ambient real completeness as alternate inputs. Hence the completion handoff is exact over precisely the displayed rows. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCompletionHandoffExactness`

Theorem 7153.13 (Hausdorffization separated completion factorization)**.** *Every completion-facing route out of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet factors through the finite chain*

$$P \longmapsto S \longmapsto M \longmapsto C.$$

No route can bypass this chain through quotient equality, selected representatives, host separated-reflection objects, or quotient completion equality.

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated completion factorization`

Proof. Read the separated-reflection boundary theorem Definition 7153.3. It already identifies every $\mathsf{SeparatedMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer route as passing through the pseudometric source P, separated-reflection boundary S, and separated metric and completion rows M, C.

Apply zero-distance window determinacy, Definition 7153.8. The boundary row S is fixed by the displayed zero-distance window route before M or C is visible, so a completion consumer cannot choose a second separated boundary downstream.

Finally apply completion handoff exactness, Definition 7153.12. It places C after $P, S, M, W, R, E, T, K, G, N$, while the ledger non-quotient boundary rules out quotient equality and selected representatives. The only completion-facing factorization is therefore $P \longmapsto S \longmapsto M \longmapsto C$. □

Theorem 7153.14 (Hausdorffization separated-completion obligation)**.** *Every accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet exposes the* $\mathsf{Pseudometric}^{\uparrow}$ *source,* $\mathsf{SeparatedReflection}^{\uparrow}$ *boundary,* $\mathsf{SeparatedMetric}^{\uparrow}$ *target,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before a separated-completion consumer reads it. In carrier notation, the consumer route is exhausted by*

$$P, S, M, C, W, R, E, T, K, G, N.$$

Proof. Project the carrier of Definition 7153.1. The displayed rows give exactly the pseudometric source P, separated-reflection boundary S, separated metric target M, completion handoff C, window, readback, seal, transport, replay, provenance, and local naming rows.

Use the separated-reflection scope of Definition 7153.5. It forces separated reads to begin at P and S and to authenticate zero-distance evidence through W, R, E before the separated metric row can be consumed.

Then apply the completion consumer route of Definition 7153.6. It places every completion-facing read after $P \longmapsto S \longmapsto M \longmapsto C$ and preserves the route through T, K, G, N.

Finally use the non-quotient boundary of Definition 7153.4. Since no extra completion coordinate exists, the listed rows are all that a separated-completion consumer can read. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated completion obligation`

Theorem 7153.15 (Hausdorffization completion-boundary exhaustion)**.** *Every completion-facing read of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet is exhausted by the displayed rows*

$$P, S, M, C, W, R, E, T, K, G, N.$$

The route combines zero-distance exactness from Definition 7153.9, completion-route stability from Definition 7153.10, and non-quotient nonescape from Definition 7153.18. Thus a downstream separated-metric or completion consumer may project only the pseudometric source, separated-reflection boundary, separated metric target, completion handoff, finite windows, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier completion boundary exhaustion`

Proof. Project the accepted carrier through Definition 7153.1. The row projection gives exactly $P, S, M, C, W, R, E, T, K, G, N$, so any completion-facing read must begin from that finite packet.

Apply Definition 7153.9. It fixes the separated-reflection boundary S by the displayed P, W, R, E zero-distance evidence before the separated metric row M or completion handoff C can add any further read.

Apply Definition 7153.10. It preserves the route $P \longmapsto S \longmapsto M \longmapsto C$ under componentwise hsame transport T and displayed Cont replay K, so the completion reader cannot reorder the rows or bypass the boundary.

Finally use Definition 7153.18. Quotient equality, selected representatives, quotient completion, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate. The remaining provenance and naming rows G, N therefore certify the displayed packet rather than extending it. □

Theorem 7153.16 (Hausdorffization Real-completion consumer factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$*-and-*$\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet factors through the displayed chain*

$$\mathsf{Pseudometric}^{\uparrow} \longmapsto \mathsf{SeparatedReflection}^{\uparrow} \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

before any separated-completion conclusion is read. In carrier notation, the consumer may project only $P, S, M, C, W, R, E, T, K, G, N$*, with* S *fixed by zero-distance evidence,* C *stable under the displayed route, and* W, R, E *supplying the finite observation, readback, and Real seal.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationRealCompletionConsumerFactorization`

Proof. Start with zero-distance exactness from Definition 7153.9. It fixes the separated-reflection row S from the pseudometric source P and the finite W, R, E evidence before either the separated metric row M or the completion row C can be consumed.

Next apply completion-route stability, Definition 7153.10. The completion handoff remains downstream of $P \longmapsto S \longmapsto M \longmapsto C$ under the same hsame transport and Cont replay rows T, K.

Use the non-quotient nonescape obligation Definition 7153.18. It rules out quotient equality, selected representatives, quotient completion, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness as alternate routes to a Real or completion read.

Finally apply completion-boundary exhaustion, Definition 7153.15. The remaining consumer surface is precisely $P, S, M, C, W, R, E, T, K, G, N$, so every Real-and-Completion consumer reads the displayed chain before any separated-completion conclusion is available. □

Theorem 7153.17 (Hausdorffization separated Real-completion scope)**.** *Every separated* $\mathsf{Real}^{\uparrow}$*-completion read of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Pseudometric}^{\uparrow}$*,* $\mathsf{SeparatedReflection}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before any completion conclusion is available. In carrier notation the only readable rows are* $P, S, M, C, W, R, E, T, K, G, N$*: zero-distance evidence fixes* S*, the separated metric and completion route reads* M, C*, finite windows and regular rational readback expose* W, R*, and the Real seal is* E*. The scope contains no quotient completion, representative selection, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use zero-distance exactness, Definition 7153.9. It fixes the separated-reflection boundary from the pseudometric source and the finite readback rows before any separated metric consumer can inspect the packet.

Apply completion-route stability and the Real-completion consumer factorization, Definition 7153.10 and Definition 7153.16. These place the separated metric row and completion handoff downstream of P, S and upstream of the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback rows.

Finally combine non-quotient nonescape with completion-boundary exhaustion, Definition 7153.18 and Definition 7153.15. The refused quotient, representative, trichotomy, countable-choice, and ambient-completeness data are absent from the carrier, so the separated Real-completion scope is exactly the displayed non-quotient row route. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated real completion scope`

Theorem 7153.18 (Hausdorffization non-quotient nonescape obligation)**.** *An accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet has no coordinate for host quotient equality, selected representatives, quotient completion, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every separated-reflection or completion-facing read remains inside the finite carrier rows* $P, S, M, C, W, R, E, T, K, G, N$*.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationNonquotientNonescapeObligation`

Proof. Begin with the carrier definition Definition 7153.1. Its coordinates are the pseudometric source, separated-reflection boundary, separated metric target, completion handoff, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

The separated-reflection scope theorem Definition 7153.5 keeps zero-distance comparisons inside P, S, W, R, E. The completion consumer route Definition 7153.6 keeps completion-facing reads inside $P, S, M, C, W, R, E, T, K, G, N$.

The ledger non-quotient boundary Definition 7153.4 explicitly excludes host quotient equality, selected representatives, quotient completion, trichotomy, countable choice, and ambient Real completeness. Therefore none of those data can escape through the local packet. □

Theorem 7153.19 (Hausdorffization separated quotient row)**.** *For every accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet* $H = (P, S, M, C, W, R, E, T, K, G, N)$*, the separated-quotient information visible to a consumer is only the displayed* $\mathsf{SeparatedReflection}^{\uparrow}$ *row* S *read from the* $\mathsf{Pseudometric}^{\uparrow}$ *source row* P*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The packet refuses quotient-space primitives, selected representatives, and any ambient* $\mathsf{Real}^{\uparrow}$ *completeness clause; separated-metric consumers may read only the handoff through* S *into* $\mathsf{SeparatedMetric}^{\uparrow}$ *row* M *and* $\mathsf{MetricCompletion}^{\uparrow}$ *row* C*.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated quotient row`

Proof. Project the carrier of Definition 7153.1. The only boundary row between pseudometric source data and separated-metric data is S.

The evidence for S is finite. It must be inspected through the StreamName row W, read back by the RegSeqRat row R, and sealed by the Real row E before M or C can consume it.

Apply the non-quotient nonescape obligation of Definition 7153.18. Since quotient primitives, representatives, and ambient Real completeness are absent, S is the full separated quotient row available to the packet. □

Theorem 7153.20 (Hausdorffization metric-completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet receives its handoff only through the route*

$$\mathsf{Pseudometric}^{\uparrow} \to \mathsf{SeparatedReflection}^{\uparrow} \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow}$$

as displayed by P, S, M, C *and authenticated by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows* W, R, E, T, K, G, N*. The handoff is not a quotient-space completion, not a selected representative construction, and not a use of ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationMetricCompletionHandoff`
↪ variant `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier metriccompletion handoff`

Proof. Use the completion consumer route Definition 7153.6. It places every completion-facing read after $P \longmapsto S \longmapsto M \longmapsto C$.

Use the separated quotient row theorem Definition 7153.19. It identifies the boundary S as the only separated-reflection row available from the finite pseudometric evidence.

The authentication rows W, R, E, T, K, G, N preserve and name the same route. The nonescape theorem excludes quotient-space primitives, selected representatives, and ambient Real completeness, so the displayed route is the full metric-completion handoff. □

Theorem 7153.21 (Hausdorffization NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}$ *obligation surface consists exactly of the* $\mathsf{Pseudometric}^{\uparrow}$ *source row,* $\mathsf{SeparatedReflection}^{\uparrow}$ *boundary row,* $\mathsf{SeparatedMetric}^{\uparrow}$ *target row,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every separated-reflection or metric-completion consumer must factor through those rows while refusing quotient-space primitives, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier namecert obligation surface`

Proof. Apply the separated quotient row theorem Definition 7153.19. It fixes the boundary read at P, S, W, R, E and excludes quotient-space data.

Apply the metric-completion handoff theorem Definition 7153.20. It places every completion consumer on the route through P, S, M, C with the same readback, transport, replay, provenance, and naming rows.

The NameCert scope package Definition 7153.7 then names exactly that carrier inventory. No selected representative, quotient completion, or ambient Real-completeness coordinate remains for the local certificate to expose. □

Theorem 7153.22 (Hausdorffization complete-metric limit uniqueness route)**.** *Let* $H = (P, S, M, C, W, R, E, T, K,$ *be an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *packet whose separated metric-completion handoff is consumed by* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$*. The consumer can read the handoff only through displayed zero-distance evidence* P, W, R, E*, the separated-classifier row* S*, the separated metric and completion rows* M, C*, and the structural rows* T, K, G, N*. It cannot read quotient equality, selected host representatives, quotient completion, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier complete metric limit uniqueness route`

Proof. Start with zero-distance exactness, Definition 7153.9. It fixes the separated-reflection boundary S from the displayed P, W, R, E evidence before any completion-facing row is available.

Use separated-completion factorization, Definition 7153.13. It keeps every completion read downstream of the boundary S and the separated metric row M.

Apply the metric-completion handoff theorem Definition 7153.20. The handoff to C remains attached to the same finite readback, transport, replay, provenance, and naming rows.

Finally route the consumer through chapter 5983. That packet reads limit uniqueness only after its zero-distance and separated-classifier rows are displayed, so the Hausdorffization handoff supplies exactly those visible rows and no quotient equality, selected representative, quotient completion, or ambient completeness data. □

Theorem 7153.23 (Hausdorffization separated-reflection obligation)**.** *Every accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *carrier exposes its separated read only after the pseudometric, separated-reflection, separated metric,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows are displayed. The separated classifier is therefore a row-projection boundary from* $P, S, M, W, R, E, T, K, G, N$*, not quotient equality of representatives and not an ambient Hausdorff-reflection principle.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier separated reflection obligation`

Proof. Project the accepted carrier of Definition 7153.1. The source pseudometric row P and separated-reflection boundary row S are visible before the separated metric row M can be read.

Apply zero-distance exactness and separated-completion factorization, Definition 7153.9 and Definition 7153.13. They keep the separated classifier attached to displayed zero-distance evidence, finite windows, regular rational readback, and the Real seal.

Finally use the NameCert obligation surface Definition 7153.21. The rows T, K, G, N transport, replay, source, and name the same packet. Since the carrier has no quotient-representative coordinate or ambient reflection row, separated reads can use only the displayed row-projection boundary. □

Theorem 7153.24 (Hausdorffization completion-consumer nonescape)**.** *Let $H = (P, S, M, C, W, R, E, T, K, G, N)$ be an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *carrier. Any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *consumer can read only the displayed zero-distance evidence rows P, W, R, E, the separated-classifier row S, the separated metric row M, the completion handoff C, and the structural rows T, K, G, N. It cannot read quotient representatives, quotient completion, selected host representatives, or ambient completion data.*

Proof. Start with the metric-completion handoff theorem Definition 7153.20. It places completion-facing reads after the pseudometric source, separated-reflection boundary, separated metric row, finite windows, regular rational readback, Real seal, and structural rows.

Use the complete-metric limit uniqueness route Definition 7153.22. That route permits the uniqueness consumer to see only zero-distance evidence, the separated classifier, separated metric data, completion handoff, transport, replay, provenance, and naming.

The same row projection applies to $\mathsf{MetricCompletion}^{\uparrow}$ consumers. Because quotient representatives, quotient completion, selected host representatives, and ambient completion objects are not carrier coordinates, no completion consumer can escape the displayed boundary. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier completion consumer nonescape`

Theorem 7153.25 (Hausdorffization scoped completion readback)**.** *Let $H = (P, S, M, C, W, R, E, T, K, G, N)$ be an accepted* $\mathsf{Hausdorffization}^{\uparrow}$ *carrier. Before any separated completion consumer reads H, the readback route must expose the zero-distance classifier row P, W, R, E, the separated-reflection row S, the metric-completion handoff M, C, the non-quotient ledger boundary, componentwise transport T, replay row K, provenance row G, and local-name row N. The route refuses quotient equality, host equality, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as completion data.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier scoped completion readback`

Proof. Start with zero-distance exactness, Definition 7153.9. It fixes the classifier read at the displayed pseudometric, finite-window, regular-rational, and Real-seal rows P, W, R, E.

Use separated-reflection scope, Definition 7153.5. That theorem places the separated boundary S downstream of the same zero-distance rows and upstream of any separated metric or completion consumer.

Apply the metric-completion handoff theorem Definition 7153.20. The handoff to completion passes through M, C and remains authenticated by transport, replay, provenance, and local naming rows T, K, G, N.

Finally apply completion-consumer nonescape, Definition 7153.24, together with the non-quotient ledger boundary Definition 7153.4. Quotient equality, host equality, selected representatives, and ambient Real completeness are not carrier coordinates, so no separated completion consumer can read them through the scoped route. □

Theorem 7153.26 (Hausdorffization public export)**.** *The public* $\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}$ *export is exactly the composed route*

$$P, S, M, C, W, R, E, T, K, G, N$$

from $\mathsf{Pseudometric}^{\uparrow}$ *zero-distance evidence through* $\mathsf{SeparatedReflection}^{\uparrow}$ *into separated* $\mathsf{MetricCompletion}^{\uparrow}$ *consumption. A separated-metric or completion reader may consume that route as the public certificate surface, but the export excludes quotient equality, selected representatives, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationPublicExport`

Proof. Start with the completion consumer route of Definition 7153.6. It forces completion-facing reads to pass through $P \longmapsto S \longmapsto M \longmapsto C$ and then authenticate the same route through W, R, E, T, K, G, N.

Use the NameCert scope package of Definition 7153.7. It identifies the local certificate with precisely the finite rows $P, S, M, C, W, R, E, T, K, G, N$, so the public export has no additional coordinate beyond the displayed route.

Finally apply separated-reflection nonescape and scoped completion readback, Definition 7153.11 and Definition 7153.25. These theorems rule out quotient equality, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness as readable data, leaving exactly the stated public certificate surface. □

Lean4 ✓ `BEDC.Derived.HausdorffizationUp.HausdorffizationCarrier public export`

Theorem 7153.27 (Hausdorffization public export status)**.** *A public* $\mathsf{Hausdorffization}^{\uparrow}$ *export may expose the separated quotient reading only after it displays the pseudometric row, separated-reflection boundary, separated metric row, metric-completion handoff,* $\mathsf{Real}^{\uparrow}$ *zero-distance seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The export status is therefore the finite route*

$$P, S, M, C, W, R, E, T, K, G, N$$

from pseudometric names through separated reflection into completion-facing data, not a quotient primitive, host equality, selected representative, or ambient completion theorem.

Proof. Inspect the public export theorem and the scoped completion readback. They order the pseudometric source, separated-reflection boundary, separated metric row, completion handoff, finite StreamName windows, RegSeqRat readback, Real zero-distance seal, and structural rows before any separated completion consumer reads the packet.

The public status is exactly that displayed route. The hsame, Cont, Pkg, and NameCert rows keep transport, replay, provenance, and local naming attached to the same Hausdorffization carrier, so the separated quotient reading is only an exported interpretation of the finite certificate surface.

Because the route is read from the displayed rows, it does not create a quotient primitive or appeal to host equality. Selected representatives and ambient completion data are also absent from the carrier, so they cannot be consumed by the public export. □

Falsifiable prediction: Every accepted $\mathsf{Hausdorffization}^{\uparrow}$ read exposes a Pseudometric source row, SeparatedReflection boundary, SeparatedMetric target row, MetricCompletion handoff, StreamName windows, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any separated metric or completion consumer reads it.

Independence witness: separated-reflection, pseudometric, separatedmetric, metriccompletion, streamname, regseqrat, real: $\mathsf{Hausdorffization}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds zero-distance evidence, separated-reflection boundary, separated metric handoff, completion-facing handoff, finite windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming while refusing quotient equality, selected representatives, quotient completion, trichotomy, countable choice, and ambient completeness.

Lean4 (statement) `BEDC.Derived.HausdorffizationUp`

Closure status for $\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}$.

Constructive story: A $\mathsf{Hausdorffization}^{\uparrow}$ packet is generated from a Pseudometric source row, SeparatedReflection boundary, SeparatedMetric target row, MetricCompletion handoff, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 7153.1, Definition 7153.2, Definition 7153.3, Definition 7153.4, and Definition 7153.6 display the carrier and NameCert obligation surface for the finite separated-reflection boundary from pseudometric names into separated metric completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Hausdorffization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality, representative selection, ambient Hausdorff reflection, quotient completion, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance exactness, completion-boundary exactness, non-quotient ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffizationUp.

Chapter 7154

A Concrete Naming Certificate for $\mathsf{FriendlyMetricCompletion}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley-style separated metric language, the $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ packet records the finite handoff by which a metric source can be read as completion-ready without quotienting Cauchy names or selecting representatives. It sits after the separated-reflection boundary of chapter 7153, the separated metric surface of chapter 1411, and the completion handoff of chapter 1506. Its analytic readback is the L10 finite route through chapter 126, chapter 301, and chapter 38; the packet names a friendly completion carrier, not a host completed metric space.

Definition 7154.1 (Friendly metric completion carrier)**.** *A* $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, S, M, C, W, R, E, H, K, G, N)$$

with the following displayed rows:

1. *P is the metric-source row, including the finite distance comparisons opened by completion-facing consumers.*
2. *S is the separatedness boundary row, recording the zero-distance identification test exposed by* $\mathsf{Hausdorffization}^{\uparrow}$ *before completion reads begin.*
3. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row obtained from the displayed boundary rather than from a host quotient.*
4. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row for downstream completion consumers.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which Cauchy observations are opened.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed distance and Cauchy readbacks.*
8. *H is componentwise* hsame *transport for $P, S, M, C, W, R, E, G, N$.*
9. *K is displayed* Cont *replay of the same friendly route.*
10. *G is the* Pkg *provenance over* $\mathsf{FriendlyMetricCompletion}^{\uparrow}$, $\mathsf{Hausdorffization}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FriendlyMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, S, M, C, W, R, E, H, K, G, N$. It has no coordinate for quotient Cauchy equality, selected representatives, host completion points, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7154.2 (Friendly metric completion NameCert obligations)**.** *For every accepted* $\mathsf{FriendlyMetricCompletion}$ *carrier $F = (P, S, M, C, W, R, E, H, K, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{FriendlyMetricCompletion}^{\uparrow}}$ *surface consists exactly of metric-source admission, separatedness-boundary exposure, separated metric handoff, metric-completion handoff, finite StreamName windows, RegSeqRat readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing consumer must read the displayed boundary route*

$$P \longmapsto S \longmapsto M \longmapsto C \longmapsto W, R, E \longmapsto H, K, G, N$$

before it can use the packet as completion data.

Proof. Project the carrier of Definition 7154.1. The displayed rows are exactly $P, S, M, C, W, R, E, H, K, G, N$, so carrier admission supplies the local inventory.

The separatedness boundary is read before the completion handoff. The row S records the zero-distance test visible through the Hausdorffization route, and M is the separated metric row built from that boundary.

The completion row C is downstream of M, while W, R, E provide the finite Cauchy-window, regular-rational, and Real-seal readback. The structural rows H, K, G, N transport, replay, source, and name the same finite route.

Since the carrier has no quotient, selected-representative, host-completion, choice, or ambient-completeness coordinate, the displayed route exhausts the NameCert obligations. □

Theorem 7154.3 (Friendly metric completion finite Cauchy handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Hausdorffization}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ *packet receives Cauchy-facing data only through the finite rows $P, S, M, C, W, R, E, H, K,$ In particular, the handoff exposes the source metric row, separatedness boundary, separated metric row, completion row, finite Cauchy windows, regular rational readback, Real seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *before any completion consumer reads it.*

Proof. Start with the NameCert obligations of Definition 7154.2. They force every completion-facing read to pass through the source row P, separatedness boundary S, separated metric row M, and completion handoff C.

Next read the L10 finite Cauchy surface. The rows W, R, E expose the finite StreamName window, RegSeqRat readback, and terminal Real seal used by the completion consumer.

Finally replay the structural rows H, K, G, N. They preserve the same route under componentwise hsame, displayed Cont, package provenance, and local naming. The carrier lacks quotient stream equality, selected representatives, host completion points, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so no other Cauchy handoff is available. □

Falsifiable prediction: Every accepted $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ read exposes a metric source row, Hausdorffization separatedness boundary, SeparatedMetric row, MetricCompletion handoff, StreamName windows, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any completion-facing consumer reads it.

Independence witness: hausdorffization, separatedmetric, metriccompletion, streamname, regseqrat, real: $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ is not identical to any listed sibling because it jointly binds

the non-quotient separatedness boundary, separated metric handoff, completion handoff, finite Cauchy windows, regular rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FriendlyMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FriendlyMetricCompletion}^{\uparrow}$ packet is generated from a metric source row, separatedness boundary, SeparatedMetric row, MetricCompletion handoff, finite StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FriendlyMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7154.1, Definition 7154.2, and Definition 7154.3 seed the finite non-quotient completion-friendly route from separated metric data to completion-facing Cauchy consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FriendlyMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected representatives, host completion points, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separatedness-boundary exactness, completion-handoff exactness, finite Cauchy readback, transport, replay, provenance, and local naming for BEDC.Derived.FriendlyMetricCompletionUp.

Chapter 7155

A Concrete Naming Certificate for LocatedIntermediateValueModulus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the LocatedIntermediateValueModulus$^{\uparrow}$ packet records the finite real-window modulus used when a sign-change interval is inspected by bisection before a Real-facing consumer reads the terminal seal. It sits after the finite-window naming discipline of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, the real-name boundary of chapter 38, the located real intermediate-value route of chapter 3394, and the constructive IVT modulus route of chapter 6431. The packet names the finite modulus surface consumed by L10 real-window arguments; it does not select a zero, decide trichotomy, import open-cover compactness, invoke choice, or assert ambient completeness of Real$^{\uparrow}$.

Definition 7155.1 (LocatedIntermediateValueModulus carrier)**.** *A* LocatedIntermediateValueModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, D, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the bracket row, recording the located real interval endpoints, the retained subinterval, and the finite sign-change boundary consumed by the route.*
2. *S is the sign-change row, recording the inspected endpoint signs and the midpoint sign reads used by the retained branch prefix.*
3. *D is the dyadic refinement row, recording the* DyadicRatCore$^{\uparrow}$ *mesh, halving, radius comparison, and retained-width budget.*
4. *B is the bisection-modulus row, recording the finite number of branch reads required before the requested located window is exposed.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row for endpoint, midpoint, and retained-interval observations.*
6. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the same windows.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row used only after the bracket, sign-change, dyadic refinement, bisection modulus, finite windows, and readback rows have been displayed.*
8. *H stores componentwise* hsame *transport for I, S, D, B, W, R, E and the structural rows.*

9. *C stores displayed* Cont *replay for bracket inspection, sign readback, dyadic refinement, modulus consumption, regular rational readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedIntermediateValueModulus}^{\uparrow}$, $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$, $\mathsf{ConstructiveIVTModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValueModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected zero, real trichotomy, open-cover compactness, countable choice, quotient real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7155.2 (LocatedIntermediateValueModulus NameCert obligations)**.** *For every accepted carrier $M = (I, S, D, B, W, R, E, H, C, P, N)$ of Definition 7155.1, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValueModulus}^{\uparrow}}$ *surface consists exactly of bracket admission, sign-change exposure, dyadic refinement, bisection-modulus consumption, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier rows in order. The bracket and sign-change rows I, S provide the only source of opposite-sign information, while the dyadic row D and bisection-modulus row B record the finite refinement budget.

The observation and readback rows W, R expose only the windows requested by that finite budget. The terminal seal E is therefore downstream of bracket inspection, sign readback, dyadic refinement, and regular rational readback.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no further coordinate appears in M, the NameCert surface exports exactly the displayed rows and excludes selected zeros, trichotomy, choice, open-cover compactness, quotient equality, and ambient completeness. □

Theorem 7155.3 (LocatedIntermediateValueModulus source stability)**.** *Let $M = (I, S, D, B, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedIntermediateValueModulus}^{\uparrow}$ *carrier. Transporting M along its displayed* hsame *row keeps the bracket row I, sign-change row S, dyadic refinement row D, bisection-modulus row B, finite windows W, regular readback R, and Real seal E in the same packet. A consumer cannot replace the transported source by* $\mathsf{Real}^{\uparrow}$ *completeness, real trichotomy, a host root selector, or an off-packet intermediate-value theorem.*

Proof. Read the componentwise transport row H. Its source and target components are indexed by the rows I, S, D, B, W, R, E together with the structural rows, so transport preserves the same bracket, sign-change, refinement, window, readback, and seal inventory.

Replay through C does not add a new source row. It only reopens the displayed bracket, sign, dyadic, modulus, window, readback, and seal routes already present in the packet.

The provenance row P names the $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ and $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ routes as sources, but it names them through this finite carrier. Thus no transported read can escape to a selected zero, trichotomy, open-cover compactness, choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 7155.4 (LocatedIntermediateValueModulus ledger exactness)**.** *For an accepted* LocatedIntermediateValue *carrier $M = (I, S, D, B, W, R, E, H, C, P, N)$, the bisection-modulus row B records exactly the finite branch budget justified by the bracket row I, sign-change row S, dyadic refinement row D, finite observation row W, regular rational readback row R, and Real seal row E. The ledger does not add open-cover compactness, a selected zero, trichotomy, choice, quotient real equality, or a completed* $\mathsf{Real}^{\uparrow}$ *object.*

Proof. The row B is read only after I, S, D have fixed the source bracket, sign boundary, and dyadic refinement budget. It therefore records the modulus for that displayed sign-change route rather than an ambient existence theorem.

The rows W, R, E give the finite observation, regular rational readback, and terminal Real seal attached to the same branch budget. The replay row C may present these reads to a consumer, but it cannot enlarge B into a completed choice sequence or a selected root.

Package provenance P and local naming N keep the ledger attached to $\mathsf{LocatedIntermediateValueModulus}^{\uparrow}$. Since the carrier has no row for open-cover compactness, trichotomy, choice, quotient equality, or complete $\mathsf{Real}^{\uparrow}$, the ledger is exact for the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValueModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntermediateValueModulus}^{\uparrow}$ packet is generated from bracket, sign-change, dyadic refinement, bisection-modulus, finite StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValueModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7155.1, Definition 7155.2, Definition 7155.3, and Definition 7155.4 bind the finite L10 real-window modulus surface over $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$, $\mathsf{ConstructiveIVTModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValueModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected zeros, real trichotomy, countable choice, quotient real equality, bridge checking, or complete $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure asks for checked carrier admission, sign-change stability, dyadic refinement exactness, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntermediateValu

Chapter 7156

A Concrete Naming Certificate for $\mathsf{CantorSeriesExpansion}^{\uparrow}$

Following Cantor and Hardy–Wright, the $\mathsf{CantorSeriesExpansion}^{\uparrow}$ packet records a variable-base digit expansion as finite denominator-product windows before the expansion can enter the regular rational and Real seal pipeline. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the sibling Engel denominator route of chapter 2007. The packet names only finite base, digit, partial-sum, and tail-bound data; it does not quotient digit streams, select an infinite sum, invoke trichotomy, assume countable choice, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7156.1 (Cantor series expansion carrier)**.** *A* $\mathsf{CantorSeriesExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, D, S, T, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the variable-base row, listing the accepted finite base prefix and the denominator products generated from it.*
2. *D is the digit-bound row, recording for each displayed base entry the corresponding finite digit and its admissible bound.*
3. *S is the finite partial-sum row, displaying the rational sums obtained from the visible base and digit windows.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row, bounding the unread tail of the denominator-product expansion for each precision request.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, selecting exactly the base, digit, partial-sum, and tail-ledger prefixes inspected by a read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from B, D, S, T, W and no other numeric coordinate.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback row is displayed.*
8. *H records componentwise* hsame *transport for the base, digit, partial-sum, tolerance-ledger, stream-window, readback, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through*

$$B, D, W \longmapsto S \longmapsto T \longmapsto R \longmapsto L.$$

10. *P is the* Pkg *provenance over* $\mathsf{CantorSeriesExpansion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{EngelExpansion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorSeriesExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient digit-stream equality, selected infinite sums, arbitrary base-choice functions, trichotomy, countable-choice selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7156.2 (CantorSeriesExpansion NameCert obligations)**.** *For every accepted* $\mathsf{CantorSeriesExpansion}^{\uparrow}$ *carrier $Q = (B, D, S, T, W, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorSeriesExpansion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the variable-base row, digit-bound row, finite partial-sum row, dyadic tolerance ledger, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *base-prefix visibility, requiring each denominator product consumed by a partial sum or tail bound to be generated from the displayed row B;*
3. *digit-bound exactness, requiring every displayed digit in D to be checked against the corresponding displayed base entry before S is read;*
4. *finite partial-sum exactness, requiring S to expose only* $\mathsf{Rat}^{\uparrow}$ *sums generated from the base and digit windows selected by W;*
5. *dyadic tail control, requiring every unread-tail claim consumed by R or L to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger T;*
6. *readback order, requiring the finite route $B, D, W \longmapsto S \longmapsto T \longmapsto R \longmapsto L$ before any Real-facing consumer receives the packet;*
7. *transport, continuation replay, provenance, and local naming for exactly the rows recorded in H, C, P, N.*

No CantorSeriesExpansion NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier $Q = (B, D, S, T, W, R, L, H, C, P, N)$. The projection exposes the base prefix, digit bounds, partial sums, tolerance ledgers, stream windows, readback rows, and structural rows as finite BEDC coordinates.

The base and digit data are read before the analytic handoff. The row B supplies denominator products, while D supplies only digits whose bounds have been displayed against those bases. The row W fixes the finite window in which both rows may be inspected.

The summation and tail steps are separate. The row S records rational partial sums generated from the displayed base and digit windows, and the row T records dyadic bounds for every unread tail before a regular rational or Real-facing consumer can use the expansion.

Finally R and L are downstream of T. Componentwise transport in H and replay in C preserve the same displayed route, while P and N only source and name it. Since the excluded host and choice data are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7156.3 (CantorSeriesExpansion Real handoff)**.** *Let* $Q = (B, D, S, T, W, R, L, H, C, P, N)$ *be an accepted* $\mathsf{CantorSeriesExpansion}^{\uparrow}$ *carrier. Every Real-facing read of the Cantor series expansion factors through*

$$B, D, W \longmapsto S \longmapsto T \longmapsto R \longmapsto L$$

with structural support H, C, P, N*. Thus the consumer receives variable-base rows, digit-bound rows, finite rational partial sums,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal, all as finite BEDC data.*

The handoff supplies no quotient stream, no selected infinite sum, no arbitrary base-choice function, no trichotomy branch, no countable-choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle. It is therefore a traditional Cantor-series representation route in the L10 Real and Completion neighborhood, parallel to the Engel route but not identified with it as a host real theorem.*

Proof. Start with a downstream consumer of R or L. By Definition 7156.1, neither row is an independent real object; both are downstream of the displayed base, digit, partial-sum, stream-window, and dyadic tail-ledger rows.

The consumer first reads B, D, and W. These rows expose the accepted denominator-product prefix, the digit bounds, and the finite window in which the expansion is being inspected.

Next the consumer reads S and T. The partial-sum row records finite rational sums, while the dyadic tolerance ledger records the unread-tail control needed before regular rational readback is public.

The row R then repacks the displayed sums and bounds as a $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal row L seals only that regular rational route as a $\mathsf{Real}^{\uparrow}$ presentation. The structural rows transport, replay, package, and name the same route, so none of the excluded principles can be recovered from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSeriesExpansion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSeriesExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7156.1, Definition 7156.2, and Definition 7156.3 seed the finite variable-base digit-expansion route through dyadic tail ledgers, stream windows, regular rational readback, and terminal Real seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSeriesExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient digit streams, selected infinite sums, arbitrary base-choice functions, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or equivalence with sibling representations as host real objects.

Upgrade path: Obligation closure requires checked carrier admission, base-prefix visibility, digit-bound exactness, finite partial-sum exactness, dyadic tail control, readback order, and Real handoff proofs for BEDC.Derived.CantorSeriesExpansionUp.

Constructive story: A $\mathsf{CantorSeriesExpansion}^{\uparrow}$ packet is generated from variable-base rows, digit-bound rows, finite partial-sum windows, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7157

A Concrete Naming Certificate for $\mathsf{KuratowskiFiniteSet}^{\uparrow}$

Following Kuratowski's finite-set presentation, the $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ packet records a finite family by the displayed membership tree generated from the empty set and singleton adjoining. It sits over the endpoint discipline of chapter 32, the finite spine of chapter 36, the unary length row of chapter 28, the membership interface of chapter 3668, the finite observation windows of chapter 126, and the regular rational ledger route of chapter 301. The packet names finite set data only through displayed rows; it does not import host finite sets, quotient extensionality, selected representatives, powerset choice, or ambient equality of subsets.

Definition 7157.1 (Kuratowski finite set carrier)**.** *A* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, A, L, M, B, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the empty-set seed row, naming the initial Kuratowski stage.*
2. *A is the singleton-adjoining row, recording the displayed element row and the previous finite family row for each adjoining step.*
3. *L is the* $\mathsf{List}^{\uparrow}$ *spine row listing the adjoining schedule in construction order.*
4. *M is the* $\mathsf{Nat}^{\uparrow}$ *length row recording the finite number of adjoining steps.*
5. *B is the* $\mathsf{Bool}^{\uparrow}$ *membership-decision row for inspected entries.*
6. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership surface exposed to downstream consumers.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a consumer inspects only a bounded prefix of the adjoining schedule.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger row used by real-facing finite-window consumers that record scheduled rational readbacks beside the finite family.*
9. *H records componentwise* hsame *transport for $E, A, L, M, B, S, W, R, C, P, N$.*
10. *C records displayed* Cont *replay from a Kuratowski finite-set request through the empty seed, adjoining steps, list spine, length, membership decision, SetLike surface, finite window, and regular-rational ledger rows.*

11. *P is the* Pkg *provenance over* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, A, L, M, B, S, W, R, H, C, P, N$. It has no coordinate for a host finite-set type, quotient extensionality of subsets, selected representatives, powerset choice, hidden enumeration equality, or ambient subset equality.

Theorem 7157.2 (Kuratowski finite set NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ *carrier*

$$K = (E, A, L, M, B, S, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteSet}^{\uparrow}}$ *surface consists exactly of empty-stage admission through E, singleton-adjoining admission through A, list-spine exactness through L, unary length exposure through M, membership-decision readback through B, SetLike membership exposure through S, finite-window inspection through W, RegSeqRat ledger handoff through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 7157.1. The rows $E, A, L, M, B, S, W, R, H, C, P, N$ are the full displayed inventory.

The finite-set construction is read from E, A, L, M. The empty row starts the construction, each adjoining row records a singleton step, the list row fixes the finite spine of those steps, and the natural-number row records the length available to bounded consumers.

Membership is then read through B and S. The Boolean row records the inspected membership decision, and the SetLike row exposes that decision as the public membership surface without identifying subsets by quotient extensionality.

Finite-window consumers use W and R. StreamName opens the bounded prefix that is actually inspected, while the RegSeqRat row records the finite ledger used by real-facing schedules such as finite observations and regular rational readbacks.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no host finite-set coordinate, no powerset selector, no representative choice, and no ambient subset-equality row, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ read exposes the empty-stage row, singleton-adjoining row, List spine, Nat length, Bool membership decision, SetLike membership surface, StreamName finite window, RegSeqRat ledger handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any finite-window consumer reads the packet.

Independence witness: bool, list, nat, setlike, streamname, regseqrat: $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds empty-stage generation, singleton adjoining, finite spine, length, membership decision, SetLike exposure, finite-window inspection, regular-rational ledger handoff, transport, replay, provenance, and local naming while refusing host finite sets and quotient extensionality.

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteSet}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ packet is generated from an empty-stage row, singleton-adjoining rows, a List spine, a Nat length row, Bool membership decisions, a SetLike membership surface, StreamName finite windows, a RegSeqRat ledger handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KuratowskiFiniteSet}^{\uparrow}}$, seed)

Scope closed: Definition 7157.1 and Definition 7157.2 seed the Kuratowski finite-set packet over empty-stage, adjoining, finite-spine, length, membership, SetLike, StreamName, RegSeqRat, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KuratowskiFiniteSet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host finite sets, quotient extensionality, powerset choice, selected representatives, hidden enumeration equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, empty-stage generation, singleton adjoining, list-spine exactness, length exposure, membership readback, finite-window handoff, ledger handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.KuratowskiFiniteSetUp.

Chapter 7158

A Concrete Naming Certificate for $\mathsf{KernelCategoryFunctor}^{\uparrow}$

The $\mathsf{KernelCategoryFunctor}^{\uparrow}$ packet realizes the functorial transport branch of chapter 9363 as finite BEDC data. It sits between two $\mathsf{KernelCategory}^{\uparrow}$ packets, reads object and morphism rows through the concrete $\mathsf{Cat}^{\uparrow}$ and $\mathsf{Functor}^{\uparrow}$ surfaces, and replays transport through $\mathsf{ContinuationMonad}^{\uparrow}$-style continuation rows. The packet names functorial transport without importing host function equality, quotient morphisms, external categorical semantics, or a global category library.

Definition 7158.1 (KernelCategoryFunctor carrier)**.** *A* $\mathsf{KernelCategoryFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, O, M, I, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{KernelCategory}^{\uparrow}$ *row, exposing only its finite object, hom, identity, composition, law-transport, provenance, and local naming data.*
2. *T is the target* $\mathsf{KernelCategory}^{\uparrow}$ *row with the same displayed row discipline.*
3. *O is the object-row transport ledger, assigning each visible source object row to a visible target object row.*
4. *M is the morphism-row transport ledger, assigning each visible source continuation-ledger hom row to a visible target continuation-ledger hom row.*
5. *I records identity preservation: empty-continuation identity rows in S transport to empty-continuation identity rows in T.*
6. *Q records composition preservation: each displayed source composite transports to the target composite named by the transported hom rows.*
7. *H records componentwise* hsame *transport for the source, target, object transport, morphism transport, identity, composition, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the object and morphism transport ledgers through the source and target continuation rows.*
9. *P is the* Pkg *provenance over* $\mathsf{KernelCategoryFunctor}^{\uparrow}$, $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{ContinuationMonad}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelCategoryFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, M, I, Q, H, C, P, N$. It has no coordinate for host function equality, quotient morphism equality, external category objects, semantic functor laws outside the displayed rows, or imported category-library data.

Theorem 7158.2 (KernelCategoryFunctor transport obligations)**.** *For every accepted* $\mathsf{KernelCategoryFunctor}^{\uparrow}$ *carrier* $F = (S, T, O, M, I, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KernelCategoryFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, source kernel-category exposure, target kernel-category exposure, object-row transport, morphism-row transport, identity preservation, composition preservation, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier rows. The source and target packets are the only category-shaped inputs, and the object and morphism transport ledgers are exactly O and M.

Identity preservation is read from I. Since source identities are empty continuation rows and target identities are empty continuation rows, the preservation claim is a displayed row comparison rather than a host identity-function assertion.

Composition preservation is read from Q. Each preserved composite cites the source composite row, the two transported morphism rows, and the target composite row before any functor-facing consumer can use the transported hom.

Rows H, C, P, N then transport, replay, source, and name the same packet. Because the carrier contains no host function equality, quotient morphism equality, external category object, or imported category-library coordinate, the listed rows exhaust the finite certificate surface. □

Theorem 7158.3 (KernelCategoryFunctor continuation handoff)**.** *Every* $\mathsf{Functor}^{\uparrow}$*-facing read from an accepted* $\mathsf{KernelCategoryFunctor}^{\uparrow}$ *carrier factors through the source* $\mathsf{KernelCategory}^{\uparrow}$ *row, target* $\mathsf{KernelCategory}^{\uparrow}$ *row, displayed object transport, displayed morphism transport, identity preservation, composition preservation,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The handoff exports no host function equality, quotient morphism equality, external category object, detached naturality principle, or ambient category semantics.*

Proof. Start from Definition 7158.1. The only source of object data is O, and the only source of hom data is M.

Use Definition 7158.2. It places identity and composition preservation on the displayed rows I and Q, downstream of the source and target kernel-category packets.

The continuation handoff is then C together with provenance and naming through P, N. Componentwise transport through H preserves the same finite route.

Thus a $\mathsf{Functor}^{\uparrow}$ consumer receives functorial transport only as a finite BEDC packet over BHist, hsame, Cont, Pkg, and NameCert, with no additional host categorical coordinate. □

Theorem 7158.4 (KernelCategoryFunctor sibling route)**.** *Every accepted* $\mathsf{KernelCategoryFunctor}^{\uparrow}$ *read factors through the source and target* $\mathsf{KernelCategory}^{\uparrow}$ *packets of chapter 488, the* $\mathsf{Cat}^{\uparrow}$ *object and hom discipline of chapter 61, and the* $\mathsf{Functor}^{\uparrow}$ *transport surface of chapter 62 before* H, C, P, N *can expose local naming. The route is a finite sibling dependency over* $S, T, O, M, I, Q, H, C, P, N$*, not a host categorical semantics principle.*

Proof. Project S and T from the carrier of Definition 7158.1. These are the only rows that expose the source and target kernel-category packets, so the read begins with the sibling surface named by chapter 488.

Next project O, M, I, Q. The object and hom transport rows use the category object and hom discipline, while the identity and composition rows keep the transported data inside the displayed functor surface named by chapter 61 and chapter 62.

Finally project H, C, P, N. These rows transport, replay, source, and name the already displayed sibling route. Since the carrier has no coordinate for ambient categories, quotient morphisms, host functions, or external functor laws, the dependency is finite BEDC sibling routing rather than imported categorical semantics. □

Falsifiable prediction: An accepted $\mathsf{KernelCategoryFunctor}^{\uparrow}$ read is refuted if a functor-facing consumer obtains object transport, morphism transport, identity preservation, or composition preservation without first displaying the source and target KernelCategory packets, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: kernelcategory, category, functor, continuationmonad: $\mathsf{KernelCategoryFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds two kernel-category packets together with object-row transport, morphism-row transport, identity preservation, composition preservation, hsame transport, Cont replay, Pkg provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{KernelCategoryFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelCategoryFunctor}^{\uparrow}$ packet is generated from source and target KernelCategory rows, object-row transport, morphism-row transport, identity and composition preservation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelCategoryFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7158.1, Definition 7158.2, and Definition 7158.3 seed the finite functorial transport surface over source and target kernel-category packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelCategoryFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient morphism equality, external category objects, detached naturality, imported category-library semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target kernel-category exposure, object-row transport, morphism-row transport, identity preservation, composition preservation, transport, replay, provenance, and local naming for BEDC.Derived.KernelCategoryFunctorUp.

Chapter 7159

A Concrete Naming Certificate for $\mathsf{ConstructiveCantorBaire}^{\uparrow}$

Following Brouwer and Bishop on choice sequences and finite-prefix spaces, the $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ packet records the finite-prefix embedding route by which Cantor-space binary observations and Baire-space natural-number observations are read as one BEDC naming surface. It consumes the Cantor prefix row of chapter 1475, the Baire prefix row of chapter 334, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite bridge for countable names only; it does not add countable choice, completed stream quotients, host equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7159.1 (ConstructiveCantorBaire carrier)**.** *A* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, B, S, R, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *row exposing binary prefixes, Boolean branch observations, and the finite cylinder classifier;*
2. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *row exposing natural-number finite prefixes, common-prefix comparison, and the Baire cylinder classifier;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row that records every coordinate inspected by the Cantor-to-Baire and Baire-to-Cantor finite-prefix reads;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *window row, used only after the finite prefix route has exposed a regular rational readback and the terminal Real seal;*
5. *H is the componentwise* hsame *transport row for the Cantor prefix, Baire prefix, stream observation, regular-Cauchy readback, Real seal, replay, provenance, and local naming rows;*
6. *K is the displayed* Cont *replay row forcing each consumer to read finite binary prefixes and finite Baire prefixes before any countable-name or Real-facing observation is exported;*
7. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCantorBaire}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through these displayed finite-prefix, stream-window, regular-readback, transport, replay, provenance, and local naming rows.

Theorem 7159.2 (ConstructiveCantorBaire NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ *carrier* $E = (C, B, S, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCantorBaire}^{\uparrow}}$ *surface consists exactly of these obligations:*

1. *carrier admission for the Cantor prefix row, Baire prefix row, stream observation row, regular-rational and Real readback row, transport row, replay row, provenance row, and local naming row;*
2. *binary-to-Baire exposure, requiring each Cantor binary prefix consumed by the classifier to be replayed as a finite Baire prefix before any countable-name read is public;*
3. *Baire-to-binary exposure, requiring each finite Baire prefix consumed by a Cantor-facing client to expose the finite binary observations used by the handoff;*
4. *regular-Cauchy landing, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read to pass first through the displayed finite-prefix and* $\mathsf{StreamName}^{\uparrow}$ *rows;*
5. *classifier transport through componentwise* hsame *and displayed* Cont *replay;*
6. *non-escape of countable-choice selectors, completed stream quotients, host function equality, selected infinite branches, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No obligation is stored outside the finite packet E*.*

Proof. Project the carrier of Definition 7159.1. Carrier admission exposes exactly the rows C, B, S, R, H, K, P, N, so every local obligation must be read from those coordinates.

For binary-to-Baire exposure, read the $\mathsf{CantorSpace}^{\uparrow}$ prefix row C and replay its inspected bits through the finite observation row S. The $\mathsf{BaireSpace}^{\uparrow}$ row B receives only the finite prefix data recorded by this replay.

For Baire-to-binary exposure, read the finite Baire prefix row B and then inspect only the binary observations displayed by C and S. The route never chooses an infinite branch or a host function representative.

The regular-Cauchy landing is forced by R. A $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can inspect the readback only after the finite prefix rows and stream window have been exposed, and H, K, P, N preserve and name that same route.

The non-escape clauses follow by exhaustion of the carrier. There is no coordinate for countable choice, quotient streams, host equality, selected infinite branches, or ambient Real completeness. □

Theorem 7159.3 (ConstructiveCantorBaire regular-Cauchy induction)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *observation exported by an accepted* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ *packet is induced from a finite sequence of prefix reads*

$$\mathsf{CantorSpace}^{\uparrow}, \quad \mathsf{BaireSpace}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}$$

inside $E = (C, B, S, R, H, K, P, N)$*. The induction step extends the displayed prefix window by one Cantor bit or one Baire entry, replays the enlarged finite window through* S*, and only then permits the regular-rational readback in* R*.*

Proof. The base window is the empty prefix displayed simultaneously by the Cantor and Baire rows. It has no Real-facing content until the stream observation row records the finite request.

For a Cantor-bit extension, append the displayed bit to the binary prefix in C and replay the corresponding finite request through S. The Baire-facing row B can read only the finite prefix obtained from that replay.

For a Baire-entry extension, append the displayed natural entry to the prefix in B and inspect the finite binary observations recorded by C and S. The replay row K keeps the extension finite.

After either step, R may read the regular-rational window and terminal Real seal. Since the readback is reached only after the finite prefix rows and stream window, the induction exports no selected infinite branch or ambient completeness datum. □

Theorem 7159.4 (ConstructiveCantorBaire Cantor-space handoff)**.** *Every Cantor-facing consumer of a* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ *packet first reads the* $\mathsf{CantorSpace}^{\uparrow}$ *row of chapter 1475, then the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row of chapter 334, and then the* $\mathsf{StreamName}^{\uparrow}$ *observation row of chapter 126. Only after those rows are displayed may the consumer pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *as in chapter 301 and chapter 38.*

Proof. Start with the Cantor prefix row C. It supplies binary cylinder data and no Baire or Real-facing value by itself.

Next expose the Baire prefix row B. The handoff compares only finite prefixes, so the consumer receives a Baire cylinder route rather than an infinite branch or a selected sequence.

The stream observation row S records the actual finite coordinates inspected by both sides. The regular-rational and Real row R is available only after that finite route is displayed, and H, K, P, N preserve the same route for downstream consumers. □

Theorem 7159.5 (ConstructiveCantorBaire L10 status)**.** *The* $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ *packet is a seed-level L10 countable-name bridge. It displays the finite route from Cantor and Baire prefixes to stream observations and regular-Cauchy readback, but it does not close a public countable-name theorem, a Lean verification target, a standard bridge, or a completed Real-space compactness statement.*

Proof. The carrier and obligations display a finite route through C, B, S, R, H, K, P, N. This is enough to name the bridge object and to state how countable-name consumers must enter the Real-facing readback.

The same inventory also fixes the boundary. Public closure would require a downstream exported object surface for countable names, Lean closure would require a checked BEDC target, and bridge closure would require a standard bridge instance. None of those rows is part of this seed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCantorBaire}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveCantorBaire}^{\uparrow}$ packet is generated from CantorSpace binary prefixes, BaireSpace finite prefixes, StreamName observations, RegSeqRat and Real windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCantorBaire}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7159.1, Definition 7159.2, Definition 7159.3, Definition 7159.4, and Definition 7159.5 seed the finite-prefix bridge from Cantor and Baire rows to StreamName, RegSeqRat, and Real readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCantorBaire}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a public countable-name theorem, Lean verification, standard bridge checking, countable choice, completed stream quotients, host function equality, selected infinite branches, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix handoff, regular-Cauchy landing, classifier transport, and non-escape for BEDC.Derived.ConstructiveCantorBaireUp.

Chapter 7160

A Concrete Naming Certificate for $\mathsf{CauchyCriterionSeries}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyCriterionSeries}^{\uparrow}$ packet records the finite tail criterion for convergence of a displayed series as a BEDC NameCert surface. It reads dyadic tolerances through chapter 291, partial-sum readbacks through chapter 301, and terminal endpoint data through chapter 38. The packet is the local tail-Cauchy criterion consumed by product and uniform series routes such as chapter 6968 and $\mathsf{UniformCauchySeries}^{\uparrow}$; it names only finite windows, budgets, transport, replay, provenance, and local certificate rows.

Definition 7160.1 (Cauchy criterion series carrier)**.** *A* $\mathsf{CauchyCriterionSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, P, T, D, R, H, C, G, N)$$

with the following displayed rows:

1. *A is the summand source row, exposing only the inspected series terms and their local arithmetic provenance.*
2. *P is the partial-sum window row, naming the finite prefix sums and the tail windows compared by the criterion.*
3. *T is the tail-budget row, recording the displayed request that every inspected tail beyond the named bound stays inside the requested tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the dyadic radius, comparison slack, and arithmetic budget used by the tail estimate.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed partial sums after P, T, D have been inspected.*
6. *H records componentwise* hsame *transport for the summand source, partial-sum windows, tail budget, dyadic tolerance, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of a precision request through the source, window, tail-budget, tolerance, and readback rows.*
8. *G is the* Pkg *provenance over* $\mathsf{CauchyCriterionSeries}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyProductSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, P, T, D, R, H, C, G, N$. It has no coordinate for arbitrary choice of a threshold, quotient equality of series, a selected completed limit, unconditional rearrangement, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7160.2 (Cauchy criterion series obligations)**.** *For every accepted* $\mathsf{CauchyCriterionSeries}^{\uparrow}$ *carrier $S = (A, P, T, D, R, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionSeries}^{\uparrow}}$ *surface consists exactly of carrier admission, tail-Cauchy exactness, partial-sum stability, dyadic tolerance transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, and ledger nonescape through transport, replay, provenance, and local naming.*

Proof. Project the carrier rows. Carrier admission exposes $A, P, T, D, R, H, C, G, N$, with the summand source and partial-sum windows available before any readback.

Tail-Cauchy exactness is the joint read of P, T, D. The row P supplies the displayed partial-sum windows, T supplies the finite tail bound for the requested tolerance, and D supplies the dyadic comparison budget used by the tail estimate.

Partial-sum stability and dyadic tolerance transport are read before the endpoint handoff. Componentwise hsame transport in H preserves the same windows and dyadic budget, while C replays the same precision request through those rows.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback R is downstream of the finite tail rows, and the $\mathsf{Real}^{\uparrow}$ handoff is available only through the provenance route G after R has been displayed. The same route is the one consumed by $\mathsf{CauchyProductSeries}^{\uparrow}$ and $\mathsf{UniformCauchySeries}^{\uparrow}$ when they need a tail-Cauchy series criterion.

Ledger nonescape follows because G and N source and name only the displayed packet. Since the carrier contains no arbitrary choice coordinate, quotient series equality, selected completed limit, or ambient completeness row, the listed obligations exhaust the certificate surface. □

Theorem 7160.3 (Cauchy criterion series finite-tail handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCriterionSeries}^{\uparrow}$ *packet first reads the summand source A, partial-sum windows P, tail-budget T, and dyadic tolerance row D, then the readback row R, and only then the transport, replay, provenance, and local naming rows H, C, G, N. The handoff exports no arbitrary threshold choice, quotient series equality, selected limit, unconditional rearrangement theorem, or ambient completeness principle.*

Proof. Start from Definition 7160.1. The only numerical rows before readback are A, P, T, D, and together they display the finite tail-Cauchy comparison requested by the consumer.

Apply Definition 7160.2. It places partial-sum stability and dyadic tolerance transport upstream of R, so the regular rational readback cannot be inspected before the finite tail window and tolerance budget have been displayed.

The terminal Real-facing handoff is sourced by G and named by N, with transport and replay through H, C. These structural rows preserve the same packet and add no new analytic coordinate.

Thus the consumer receives a finite tail criterion route rather than a host convergence theorem, arbitrary selected bound, quotient-series equality, selected completed limit, or ambient completeness witness. □

Falsifiable prediction: An accepted $\mathsf{CauchyCriterionSeries}^{\uparrow}$ read is refuted if a series convergence consumer reaches RegSeqRat or Real data without first displaying the summand source, partial-sum windows, tail-budget, dyadic tolerance, transport, replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, cauchyproductseries, uniformcauchyseries: $\mathsf{CauchyCriterionSeries}^{\uparrow}$ is not bijective to any listed sibling because it binds summand source rows, partial-sum windows, finite tail budgets, dyadic tolerances, regular readback, Real handoff, transport, replay, provenance, and local naming as one finite criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterionSeries}^{\uparrow}$ packet is generated from summand source rows, partial-sum windows, finite tail budgets, dyadic tolerance rows, RegSeqRat readback, Real handoff provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCriterionSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7160.1, Definition 7160.2, and Definition 7160.3 seed the finite tail-Cauchy criterion surface for series convergence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCriterionSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary choice of tail thresholds, quotient series equality, a selected completed limit, unconditional rearrangement, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-Cauchy exactness, partial-sum stability, dyadic tolerance transport, RegSeqRat and Real handoff, ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCriterionSeriesUp.

Chapter 7161

A Concrete Naming Certificate for $\mathsf{DarbouxOscillation}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxOscillation}^{\uparrow}$ packet records the finite upper-minus-lower ledger read by partition-refinement arguments before Darboux and regulated integration consume a gap bound. It sits after the Darboux integral route of chapter 1460, the finite Darboux sums of chapter 6436, the finite partition mesh of chapter 3496, the bounded-variation source rows of chapter 398, and the terminal real-name seal of chapter 38. The packet names only finite oscillation data; it does not select partitions by choice, complete a real function space, or quotient functions by external equality.

Definition 7161.1 (Darboux oscillation carrier)**.** *A* $\mathsf{DarbouxOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (P, M, U, L, G, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition row, recording the ordered cell list, endpoint coverage, and adjacency route used by the Darboux read.*
2. *M is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *row, recording mesh width, refinement comparison, and the finite cell-width budget.*
3. *U and L are the upper and lower Darboux rows over the cells of P, read through the same bounded-function and partition source.*
4. *G is the oscillation-difference row displaying the bound $U - L$ after P, M, U, L have been projected.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed upper-minus-lower comparison.*
6. *H records componentwise* hsame *transport for the partition, mesh, upper row, lower row, oscillation difference, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through finite refinement, Darboux row projection, oscillation comparison, regular readback, and Real sealing.*
8. *Q is the* Pkg *provenance over* $\mathsf{DarbouxOscillation}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{DarbouxSums}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, M, U, L, G, E, H, C, Q, N$. It has no coordinate for host completeness, quotient equality of functions, choice-selected partitions, selected tags, or a detached supremum principle.

Lean4 (statement) `BEDC.Derived.DarbouxOscillationUp`

Theorem 7161.2 (Darboux oscillation refinement bound)**.** *Let $O = (P, M, U, L, G, E, H, C, Q, N)$ be an accepted* $\mathsf{DarbouxOscillation}^{\uparrow}$ *carrier. Every accepted refinement read preserves the displayed upper-minus-lower oscillation bound by first transporting the finite partition row P and mesh row M, then replaying the upper and lower Darboux rows U, L, and only then reading the difference row G and terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The exported surface is the finite oscillation ledger over the displayed mesh route; it exports no host completeness, quotient equality, choice-selected partition, selected tag family, or detached supremum.*

Lean4 ✓ `BEDC.Derived.DarbouxOscillationUp.DarbouxOscillationCarrier refinement bound`

Proof. Project the carrier rows. The only analytic data available before the oscillation comparison are the partition row P, mesh row M, and Darboux rows U, L.

Transport through H preserves these rows component by component. A refinement therefore changes only the displayed finite mesh route and keeps the upper and lower Darboux ledgers attached to the same cell history.

Replay through C exposes the comparison row G only after the transported partition, mesh, upper, and lower rows are visible. The Real seal E is downstream of that comparison, and provenance Q names only the Darboux, finite-mesh, bounded-variation, regulated-integral, and Real rows already displayed.

Since the packet has no coordinate for host completeness, quotient equality, selected partitions, tags, or detached suprema, a refinement consumer can export only the finite oscillation bound carried by G and sealed by E. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxOscillation}^{\uparrow}$ packet is generated from finite partition, mesh, upper Darboux row, lower Darboux row, oscillation difference, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7161.1 and Definition 7161.2 seed the finite upper-minus-lower oscillation ledger over DarbouxIntegralUp, DarbouxSumsUp, FinitePartitionMeshUp, BoundedVariationUp, RegulatedIntegralUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxOscillation}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host completeness, quotient function equality, choice-selected partitions, selected tags, detached suprema, theorem checking, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, refinement transport, mesh exactness, oscillation-difference nonescape, Real-seal nonescape, replay, provenance, and local naming for BEDC.Derived.DarbouxOscillationUp.

Chapter 7162

A Concrete Naming Certificate for $\mathsf{FiniteFanCompactness}^{\uparrow}$

Following Bishop–Bridges compact metric analysis, the $\mathsf{FiniteFanCompactness}^{\uparrow}$ packet records the finite fan of cover probes used by the compact-uniform-continuity construction of chapter 9237. It sits after the compact carrier of chapter 60, the metric distance rows of chapter 57, the continuous pointwise-modulus surface of chapter 59, the uniform-modulus consumer of chapter 369, the rational tolerance rows of chapter 37, and the finite lower-bound fold of chapter 619. The packet names finite cover and uniform-modulus data only; it does not expose open-cover compactness, external choice, or ambient completeness.

Definition 7162.1 (Finite fan compactness carrier)**.** *A* $\mathsf{FiniteFanCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, M, A, U, R, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite* $\mathsf{ProbeBundle}$ *fan row, recording the bundled probe names, center histories, and coverage witnesses.*
2. *K is the consumed* $\mathsf{Compact}^{\uparrow}$ *row, restricted to the displayed finite fan and its located cover evidence.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row used to read source distances from points to fan centers and between source points.*
4. *A is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row at each bundled center.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after the finite fan and centerwise modulus rows have been read.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *tolerance and positive-radius ledger attached to the same fan.*
7. *L is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound fold row, recording primitive recursion over the finite fan and the certified minimum radius used by the uniform-modulus route.*
8. *H records componentwise* hsame *transport for fan, compact, metric, continuous, uniform-modulus, rational, lower-bound, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through finite cover recursion, centerwise modulus reads, rational lower-bound folding, metric triangle estimates, and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* FiniteFanCompactness$^\uparrow$, Compact$^\uparrow$, Met$^\uparrow$, Continuous$^\uparrow$, UniformModulus$^\uparrow$, Rat$^\uparrow$, FiniteNetMinimumFold$^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{FiniteFanCompactness}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, K, M, A, U, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers, countable choice, quotient metric equality, or ambient completion of the metric space.

Lean4 (statement) `BEDC.Derived.FiniteFanCompactnessUp`

Theorem 7162.2 (Finite fan compactness uniform-modulus route)**.** *Let $F = (B, K, M, A, U, R, L, H, C, P, N)$ be an accepted* FiniteFanCompactness$^\uparrow$ *carrier. Every compact-uniform-continuity consumer must read the finite* ProbeBundle *fan B, the compact and metric rows K, M, the centerwise continuous rows A, the rational tolerance ledger R, and the finite lower-bound fold L before the* UniformModulus$^\uparrow$ *handoff U is exported. The route is constructive finite ProbeBundle recursion and exports no open-cover compactness, selected subcover, countable choice, quotient metric equality, or ambient completeness.*

Proof. Project the finite fan row B. It lists the probe names and center histories consumed by the compact row K, so the cover data are already finite before any modulus is read.

For each bundled probe, replay C through the metric row M and continuous row A. These rows produce only centerwise pointwise-modulus data attached to the displayed fan.

The rational ledger R and finite lower-bound fold L then perform primitive recursion over the probes. The fold keeps a certified positive lower bound for the displayed center radii and supplies the tolerance-only precision needed by U.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no row stores an arbitrary open cover, selected subcover, choice principle, quotient metric equality, or ambient completion, the uniform-modulus handoff can only be read through the finite fan route. □

Closure status for NameCert$_{\text{FiniteFanCompactness}^\uparrow}$.

Constructive story: A FiniteFanCompactness$^\uparrow$ packet is generated from a finite ProbeBundle fan, CompactUp cover row, MetricUp distance row, ContinuousUp centerwise modulus rows, UniformModulusUp handoff, RatUp tolerance ledger, FiniteNetMinimumFoldUp lower-bound fold, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{FiniteFanCompactness}^\uparrow}$, seed)

Scope closed: Definition 7162.1 and Definition 7162.2 seed the finite fan compactness route feeding compact-uniform-continuity through CompactUp, MetricUp, ContinuousUp, UniformModulusUp, RatUp, and FiniteNetMinimumFoldUp rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteFanCompactness}^\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient metric equality, ambient completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure asks for checked fan admission, compact-cover exactness, metric triangle replay, centerwise-modulus extraction, rational fold stability, uniform-modulus nonescape, transport, provenance, and local naming for BEDC.Derived.FiniteFanCompactnessUp.

Chapter 7163

A Concrete Naming Certificate for $\mathsf{UniformConvergenceTheorem}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ packet records the finite certificate surface for the standard theorem that a uniformly convergent family of continuous displayed maps has a continuous Real-facing limit read. It sits after the continuous-map source of chapter 130, the uniform-limit endpoint route of chapter 463, the shared uniform-Cauchy family row of chapter 1555, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet exposes the theorem as a finite BEDC handoff; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, host equality of functions, quotient equality of maps, countable choice, trichotomy, or a selected external limit.

Definition 7163.1 (Uniform convergence theorem carrier)**.** *A* $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, F, U, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the source-domain row inspected by the continuous maps and by the limit read.*
2. *F is the finite family row of* $\mathsf{ContinuousMap}^{\uparrow}$ *carriers, each read through the same source-domain surface.*
3. *U is the* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *or* $\mathsf{UniformLimit}^{\uparrow}$ *ledger row that supplies one displayed uniform tail discipline over the family row.*
4. *L is the limiting graph row whose endpoint reads are admitted only after F and U have been displayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for source points, family indices, continuity-modulus requests, and value names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the value windows and their rational tolerance budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the uniform ledger, stream windows, and regular rational readback have been consumed.*
8. *H records componentwise* hsame *transport for source rows, continuous-map rows, uniform-tail rows, limit graph rows, windows, readbacks, Real seals, provenance, and local names.*

9. *C records displayed* Cont *replay from source admission through continuity inspection, uniform-tail inspection, finite-window readback, graph comparison, and Real-seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformConvergenceTheorem}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformCauchyFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceTheorem}^{\uparrow}}$ *row.*

The classifier relates accepted packets only by the displayed componentwise rows $D, F, U, L, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient map equality, a selected limit outside the packet, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7163.2 (Uniform convergence theorem NameCert obligations)**.** *For every accepted* $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ *carrier*

$$T = (D, F, U, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformConvergenceTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, continuous-source compatibility for D and F, uniform-tail exactness through U, limit-graph admission through L, finite-window readback through W and R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation row supplies host equality of functions, quotient equality of maps, a selected external limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7163.1. The visible rows are the source-domain row, the continuous-map family row, the uniform-tail row, the limiting graph row, the stream-window row, the regular rational readback, the Real seal, transport, replay, provenance, and local naming.

Continuous-source compatibility is read from D and F. Each family member is a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row over the same source-domain surface, so a downstream continuity consumer has no separate host function space from which to read equality.

Uniform-tail exactness is read from U. Whether the row is consumed through the $\mathsf{UniformCauchyFamily}^{\uparrow}$ surface or through the endpoint-facing $\mathsf{UniformLimit}^{\uparrow}$ surface, the same finite ledger supplies the shared threshold discipline for every displayed source window.

The limit graph and endpoint read are downstream. The row L is admitted only after the continuous family and the uniform ledger have been exposed, while W, R, E turn the scheduled value windows into regular rational data and then into the terminal Real seal.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the excluded principles are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7163.3 (Uniform convergence theorem Real completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformCauchyFamily}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer first reads the displayed continuous-map family over the common source row, then the shared uniform-tail ledger, then the finite stream windows and regular rational readbacks, and only then the terminal Real seal. The handoff exports no ambient completeness theorem, no host equality of functions, no quotient equality of maps, no countable-choice schedule, no trichotomy principle, and no selected external limit.

Proof. Begin with a completion-facing consumer asking for the limiting endpoint. The packet admits such a read only through the displayed source and continuous-map rows D, F, so the consumer must inspect the same finite source surface used by the family.

Next use the uniform ledger U. It binds all requested source windows to one displayed tail discipline, matching the uniform-limit endpoint route of chapter 463 and the shared-family route of chapter 1555.

The limiting graph row L then records the theorem handoff: continuity data from the family and the uniform tail data are read before any endpoint row is visible. The stream windows W and regular rational row R provide the only value readback accepted by the packet.

Finally the Real seal E is reached through the same transport, replay, provenance, and local naming rows H, C, P, N. Because no refused object is a coordinate of the carrier, every Real or completion-facing read remains inside this finite route. □

Falsifiable prediction: Every accepted UniformConvergenceTheoremUp read exposes a source-domain row, a ContinuousMapUp family row, a UniformCauchyFamilyUp or UniformLimitUp tail ledger, a limiting graph row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a completion-facing consumer can inspect the endpoint.

Independence witness: continuousmap, uniformlimit, uniform-cauchy-family, streamname, regseqrat, real: $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries continuous-map source compatibility, shared uniform-tail exactness, limiting graph admission, finite-window readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformConvergenceTheorem}^{\uparrow}$ packet is generated from a common source-domain row, ContinuousMap family rows, a UniformCauchyFamily or UniformLimit tail ledger, a limiting graph row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7163.1, Definition 7163.2, and Definition 7163.3 seed the finite theorem route from displayed continuous-map families and uniform-tail ledgers to Real-facing completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, host function equality, quotient map equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.UniformConvergenceTheoremUp covering carrier admission, continuous-source compatibility, uniform-tail exactness, limit-graph admission, readback, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 7164

A Concrete Naming Certificate for $\mathsf{HamiltonianFlow}^{\uparrow}$

Following Arnold and Abraham–Marsden on Hamiltonian mechanics, the $\mathsf{HamiltonianFlow}^{\uparrow}$ packet records the finite BEDC surface where a phase-state carrier, Hamiltonian scalar readback, symplectic variation row, ODE vector-field row, dynamic-flow replay, and energy ledger are exposed before a mechanics consumer may cite Hamiltonian flow. It sits beside the Lagrangian mechanics packet of chapter 337, the ODE vector-field surface of chapter 193, the routed dynamics surface of chapter 195, the symplectic source of chapter 138, the terminal scalar seal of chapter 38, and the conservation-current boundary of chapter 269. The packet names a finite Hamiltonian-flow obligation surface; it does not select a global trajectory, complete a smooth phase manifold, identify Lagrangian and Hamiltonian formalisms, or assert Liouville volume preservation outside displayed rows.

Definition 7164.1 (HamiltonianFlowUp carrier)**.** *A* $\mathsf{HamiltonianFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, A, V, Y, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the phase-state carrier row, naming the finite phase coordinates or state tokens available to the flow packet.*
2. *A is the Hamiltonian scalar readback row, recording the displayed* $\mathsf{Real}^{\uparrow}$ *value route for the energy function on the phase-state rows.*
3. *V is the symplectic and variation row, recording the finite* $\mathsf{Symplectic}^{\uparrow}$ *source, admissible variation probes, and the pairing used to read the Hamiltonian vector field.*
4. *Y is the* $\mathsf{Ode}^{\uparrow}$ *vector-field row, exposing the finite right-hand-side data accepted for the Hamiltonian flow request.*
5. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *flow-replay row, recording how the displayed vector-field request is replayed as routed state evolution.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *energy-ledger row, comparing scalar reads along the displayed replay route and exposing only the finite conservation ledger admitted by the packet.*
7. *H records componentwise* hsame *transport for phase states, Hamiltonian scalar reads, symplectic and variation rows, ODE vector-field rows, dynamic replay, energy ledgers, and local naming.*

8. *C records displayed* Cont *replay from state and scalar rows through symplectic pairing, vector-field admission, dynamic-flow replay, and energy-ledger inspection.*
9. *P is the* Pkg *provenance over* $\mathsf{HamiltonianFlow}^{\uparrow}$, $\mathsf{LagrangianMechanics}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NoetherSymmetry}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HamiltonianFlow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, V, Y, D, E, H, C, P, N$. It has no coordinate for a host smooth phase space, selected integral curve, quotient trajectory equality, unrecorded conservation law, global Liouville theorem, or ambient completeness principle.

Theorem 7164.2 (HamiltonianFlowUp NameCert obligation surface)**.** *For every accepted* $\mathsf{HamiltonianFlow}^{\uparrow}$ *carrier $F = (S, A, V, Y, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HamiltonianFlow}^{\uparrow}}$ *surface consists exactly of phase-state carrier exposure through S, Hamiltonian scalar readback through A, symplectic and variation exposure through V, ODE vector-field admission through Y, dynamic-system flow replay through D,* $\mathsf{Real}^{\uparrow}$ *energy-ledger inspection through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every Hamiltonian consumer must inspect those rows before reading a flow, an energy comparison, or a conservation-current handoff. The surface exports no selected trajectory, no global existence theorem for solutions, no quotient equality of flows, no unledgered Liouville-volume assertion, and no equivalence with a host Lagrangian model.

Proof. Project the carrier of Definition 7164.1. The rows S, A, V, Y, D, E are the only coordinates storing phase-state data, Hamiltonian scalar reads, symplectic and variation probes, vector-field admission, dynamic replay, and energy-ledger observations.

The structural rows then preserve the same route. Componentwise transport through H keeps comparisons on the displayed rows, while replay through C records how the scalar, symplectic, vector-field, flow, and energy reads are consumed. Provenance through P names the Lagrangian, ODE, dynamic-system, symplectic, Real, and Noether-facing siblings, and N registers the local packet.

Since no additional coordinate stores a smooth ambient phase space, a selected integral curve, a quotient of trajectories, a global conservation theorem, or a host equivalence with Lagrangian mechanics, the stated rows exhaust the NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{HamiltonianFlow}^{\uparrow}$ consumer must expose phase-state, Hamiltonian scalar, symplectic-variation, ODE vector-field, DynSystem replay, Real energy-ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before it can read Hamiltonian flow data.

Independence witness: lagrangianmechanics, ode, dynsystem, symplectic, real, noethersymmetry: HamiltonianFlowUp is not bijective with any listed sibling because it binds Hamiltonian scalar readback, symplectic vector-field admission, routed flow replay, and the energy ledger in one finite NameCert packet while those siblings expose only their own source or consumer surfaces.

Closure status for $\mathsf{NameCert}_{\mathsf{HamiltonianFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HamiltonianFlow}^{\uparrow}$ packet is generated from phase-state rows, Hamiltonian scalar readback, symplectic and variation rows, ODE vector-field admission, DynSystem flow replay, RealUp energy-ledger inspection, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HamiltonianFlow}^{\uparrow}}$, seed)

Scope closed: Definition 7164.1 and Definition 7164.2 seed the finite Hamiltonian flow surface over LagrangianMechanics, Ode, DynSystem, Symplectic, Real, and NoetherSymmetry rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HamiltonianFlow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected trajectories, global solution theory, quotient flow equality, Liouville volume preservation, or Hamiltonian–Lagrangian equivalence.

Upgrade path: Obligation closure asks for checked carrier admission, flow replay, scalar energy ledger, transport, replay, provenance, and local naming for BEDC.Derived.HamiltonianFlowUp.

Chapter 7165

A Concrete Naming Certificate for $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite fan barriers, the $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ packet records the finite boundary object that prevents an L10 located-interval cover route from being read as open-cover compactness. It sits beside the located interval source of chapter 394, the closed bounded interval fan surface of chapter 3939, the finite fan compactness packet of chapter 7162, the dyadic tolerance core of chapter 291, the finite window source of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the barrier row used before finite-cover handoff; it does not import open-cover compactness, selected branches, quotient streams, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7165.1 (Located interval fan barrier carrier)**.** *A* $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, K, F, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row carrying located endpoint comparison and interval-membership data.*
2. *K is the closed bounded interval fan row that supplies the finite binary-prefix and bar boundary.*
3. *F is the finite fan compactness row read only as a bounded fan of cover probes, not as an open-cover compactness oracle.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare barrier prefixes and interval endpoints.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint names, barrier prefixes, and finite cover probes.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and dyadic tolerance checks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, K, F, D, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for interval, fan, dyadic, stream, rational, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a located-interval request through the finite fan barrier and then through D, S, Q, E.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$, $\mathsf{FiniteFanCompactness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, K, F, D, S, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected infinite branches, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.LocatedIntervalFanBarrierUp`

Theorem 7165.2 (Located interval fan barrier NameCert obligations)**.** *For every accepted* LocatedIntervalFanBarrie *carrier*

$$B = (I, K, F, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located interval admission through I, closed bounded interval fan admission through K, finite fan compactness admission through F, dyadic tolerance admission through D, stream-window admission through S, regular rational readback through Q, terminal real sealing through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7165.1. Its public rows are exactly $I, K, F, D, S, Q, E, H, C, P, N$, so every local obligation is attached to one displayed row.

The located interval row is read before the fan boundary rows, and the fan boundary rows are read before dyadic tolerance, finite stream, regular rational, and real-seal rows. Thus a finite-cover consumer cannot skip from an interval request directly to a completed cover theorem.

Transport, replay, provenance, and local naming preserve the same packet route. Since no excluded coordinate occurs in the carrier, the NameCert surface cannot call on open-cover compactness, selected branches, quotient streams, countable choice, or ambient real completeness. □

Theorem 7165.3 (Located interval fan barrier finite-cover handoff)**.** *Every L10 finite-cover consumer of an accepted* $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ *packet factors through the route*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{ClosedBoundedIntervalFan}^{\uparrow},\ \mathsf{FiniteFanCompactness}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},$$

In carrier notation, every cover handoff first reads I, K, F, D, S, Q, E and then the structural rows H, C, P, N. The handoff exports only the displayed finite barrier route and exports no open-cover compactness, branch selector, quotient stream equality, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the located interval row I. It supplies the endpoint comparison and interval-membership surface consumed by the barrier.

The closed bounded interval fan row K and finite fan compactness row F then expose a finite prefix and probe boundary. These rows are barrier data: they bound the cover handoff before any real-facing conclusion is read.

The dyadic ledger D, stream windows S, regular rational readback Q, and terminal real seal E are downstream of that barrier. The structural rows H, C, P, N replay and name the same route. Therefore the finite-cover handoff remains inside the displayed packet and cannot export any excluded compactness or choice principle. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalFanBarrierUp.LocatedIntervalFanBarrierFiniteCoverHandoff`

Theorem 7165.4 (Located interval fan barrier obligation package)**.** *For every accepted* $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ *carrier*

$$B = (I, K, F, D, S, Q, E, H, C, P, N),$$

the rows $I, K, F, D, S, Q, E, H, C, P, N$ *project the full local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$ *obligation surface. The surface contains carrier admission, located-interval admission, fan-barrier admission, finite-cover handoff, dyadic tolerance, StreamName window, regular-rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming obligations, and it contains no open-cover compactness, branch selector, quotient stream, countable-choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the displayed carrier of Definition 7165.1. The first seven rows I, K, F, D, S, Q, E are the only mathematical rows available to a cover-facing consumer: located interval data, closed bounded interval fan data, finite fan compactness data, dyadic tolerance data, stream-window data, regular rational readback, and terminal real sealing.

The structural rows then exhaust the remaining NameCert clauses. The row H supplies componentwise transport, C supplies displayed replay, P supplies provenance, and N supplies local naming for exactly the same packet.

Since every obligation is attached to one of the displayed rows, there is no residual coordinate from which open-cover compactness, a selected branch, quotient stream equality, countable choice, or ambient completeness could be read. Thus the carrier rows project the stated obligation package. □

Theorem 7165.5 (Located interval fan barrier dyadic-window induction)**.** *Every finite consumer induction over an accepted* $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ *packet reads dyadic tolerance,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the located interval row* I *and finite fan rows* K, F *have been exposed. In particular, no induction step may consume* D, S, Q, E *as an initial cover premise independent of the located interval and finite barrier rows.*

Proof. Use induction over the displayed finite row list of the carrier. The base segment consists of I, K, F: first the located interval row supplies endpoint and membership data, then the closed bounded interval fan row and finite fan compactness row expose the finite barrier boundary.

For the downstream segment, the next rows D, S, Q, E are introduced only after that base segment has been displayed. The dyadic row D compares barrier prefixes and endpoints, the window row S names the finite observations, the readback row Q records the regular rational values, and the seal row E closes the real-facing boundary.

The structural rows H, C, P, N preserve the same order because they transport, replay, source, and name the already displayed packet. Hence every finite row induction consumes dyadic tolerance, window, readback, and seal data downstream of I, K, F, and the finite-cover consumer cannot treat them as independent initial cover premises. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalFanBarrier}^{\uparrow}$ packet is generated from a located interval row, closed bounded interval fan row, finite fan compactness row, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$, obligation)

Scope closed: Definition 7165.1, Definition 7165.2, Definition 7165.3, Definition 7165.4, and Definition 7165.5 bind the carrier, full NameCert obligation package, finite-cover handoff, and dyadic-window induction order for L10 located-interval finite-cover consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalFanBarrier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected branches, quotient streams, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, fan-barrier admission, finite-cover handoff, dyadic tolerance, stream-window, regular-rational, real-seal, transport, replay, provenance, and local naming obligations for BEDC.Derived.LocatedIntervalFanBarrierUp.

Chapter 7166

A Concrete Naming Certificate for $\mathsf{CompactEmbedding}^{\uparrow}$

Following Rellich–Kondrachov compactness in the form used by Brezis and Adams–Fournier, the $\mathsf{CompactEmbedding}^{\uparrow}$ packet records the finite certificate surface for a compact embedding from displayed Sobolev source windows into a compact complete metric target. It sits after the Sobolev embedding carrier of chapter 2217, the compact and complete metric surfaces of chapter 60 and chapter 128, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the finite embedding, readback, and compact-window rows available to BEDC consumers.

Definition 7166.1 (Compact embedding carrier)**.** *A* $\mathsf{CompactEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, S, K, M, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite source-function window family naming the Sobolev functions, source-domain observations, target observations, and boundedness gauges inspected by this packet.*
2. *S is the* $\mathsf{SobolevEmbedding}^{\uparrow}$ *row that records the displayed embedding comparison for the source windows, including the exponent and codomain route visible to the consumer.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *row exposing the finite compactness window used for target observations.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row used only to read Cauchy and limit-window data inside the target surface.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance and regular readback row attached to the source-function and target-metric windows.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after F, S, K, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for source windows, Sobolev-embedding rows, compact rows, complete-metric rows, regular rational readbacks, Real seals, continuation rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from source-function windows through embedding comparison, compact target inspection, complete-metric readback, regular rational tolerance, Real sealing, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactEmbedding}^{\uparrow}$, $\mathsf{SobolevEmbedding}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected subsequence, open-cover compactness, quotient equality of functions, host representatives, countable choice, trace theory, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the displayed seal row.*

Theorem 7166.2 (Compact embedding NameCert obligations)**.** *For every accepted* $\mathsf{CompactEmbedding}^{\uparrow}$ *carrier*

$$E = (F, S, K, M, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite source-function windows, SobolevEmbedding row, Compact row, CompleteMetric row, RegSeqRat tolerance readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These rows bind the compact-embedding handoff without adding selected subsequences, open-cover compactness, quotient function equality, host representatives, countable choice, trace theory, or ambient Real completeness.*

Proof. Project the carrier of Definition 7166.1. The accepted packet exposes exactly $F, S, K, M, Q, R, H, C, P, N$.

The analytic route is then read row by row. The source-function windows are in F, the embedding comparison is in S, the compact target window is in K, and the complete-metric readback is in M.

The tolerance and endpoint observations are finite. They are read through Q and then sealed through R, so a consumer cannot replace the displayed RegSeqRat route by an ambient completeness theorem or a selected limit.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic principle is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7166.3 (Compact embedding finite route)**.** *Every consumer read exported by an accepted* $\mathsf{CompactEmbedding}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{SobolevEmbedding}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

using the displayed rows $F, S, K, M, Q, R, H, C, P, N$. The route first exposes source-function windows and the SobolevEmbedding comparison, then compact and complete-metric target windows, then regular rational tolerance readback, and only then the terminal Real seal. It supplies no selected subsequence, open-cover compactness principle, quotient function space, host representative, countable-choice schedule, trace theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with a consumer asking for compact-embedding data. Carrier admission confines the request to the finite source-function windows F and the embedding comparison row S, so the consumer first sees the same SobolevEmbedding surface cited in chapter 2217.

Next read the target side. The rows K and M expose only the compact and complete-metric windows named by this packet; the regular rational tolerances in Q are the only numerical readbacks supplied to the consumer.

The terminal seal R is downstream of those rows. Componentwise transport, displayed replay, provenance, and local naming preserve the same finite route and do not add a second analytic endpoint.

The L10 completion anchor is therefore the RegSeqRat tolerance readback of chapter 301 together with the displayed Real seal. No quotient equality, choice schedule, selected subsequence,

open-cover compactness principle, trace theorem, or Real completeness object can be read from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{CompactEmbedding}^{\uparrow}$ read exposes finite source-function windows, a SobolevEmbeddingUp row, CompactUp target windows, CompleteMetricUp readback, RegSeqRatUp tolerances, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any compact-embedding consumer can inspect the endpoint.

Independence witness: sobolev-embedding, compact, completemetric, regseqrat, real: $\mathsf{CompactEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source-function windows, embedding comparison, compact target windows, complete-metric readback, regular rational tolerances, Real sealing, transport, replay, provenance, and local naming while refusing selected subsequences, quotient function equality, choice, trace theory, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactEmbedding}^{\uparrow}$ packet is generated from finite source-function windows, a SobolevEmbedding row, Compact and CompleteMetric target rows, RegSeqRat tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7166.1, Definition 7166.2, and Definition 7166.3 seed the finite compact-embedding handoff over SobolevEmbedding, Compact, CompleteMetric, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subsequences, open-cover compactness, quotient function equality, host representatives, trace theory, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration covering carrier admission, embedding comparison access, compact target access, complete-metric readback, RegSeqRat tolerance readback, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 7167

A Concrete Naming Certificate for $\mathsf{FiniteProofTraceSieve}^{\uparrow}$

The $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ packet realizes the MetaCIC proof-search frontier as a finite sieve over candidate proof traces, bounded normal endpoints, discharge sockets, obstruction rows, and replayable audit data. It sits after the proof-search frontier of chapter 4284, the critical-path and bounded equality-checking surfaces of chapter 765 and chapter 752, the normalization boundary of chapter 892, and the closure-preservation witness of chapter 3344. The chapter records only finite candidate, endpoint, discharge, obstruction, transport, replay, provenance, and NameCert rows; it does not claim global normalization, confluence, or subject reduction.

Definition 7167.1 (Finite proof-trace sieve carrier)**.** *A* $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, E, D, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite candidate trace table, listing proof-search candidates, local rule tags, dependency windows, and the bounded endpoints to be checked.*
2. *E is the bounded normal endpoint row, read through* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *and the MetaCIC normalization frontier, that records which displayed candidates have finished endpoint witnesses.*
3. *D is the discharge-socket row, recording the subject-reduction, closedness, and proof-obligation sockets that a candidate must fill before it can be consumed.*
4. *O is the obstruction row, naming the residual open frontier when a candidate endpoint, discharge socket, or closure-preservation trace is missing.*
5. *H records componentwise* hsame *transport for trace rows, endpoint rows, discharge sockets, obstruction rows, replay rows, provenance, and local naming.*
6. *C records displayed* Cont *replay from candidate enumeration through endpoint checking, discharge inspection, obstruction retention, transport, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteProofTraceSieve}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, *proof-trace audit witness rows, closure-preservation trace rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteProofTraceSieve}^{\uparrow}}$ *row.*

The carrier has no coordinate for a complete proof-search procedure, global normalization, unrestricted confluence, closed subject reduction, quotient equality of proof traces, hidden oracle discharge, or host equality of MetaCIC terms.

Theorem 7167.2 (Finite proof-trace sieve NameCert obligations)**.** *For every accepted* $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ *carrier*

$$S = (T, E, D, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteProofTraceSieve}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite candidate table, bounded normal endpoint row, discharge-socket row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations keep each proof-search candidate inside the displayed sieve and expose any missing endpoint, discharge, or closure-preservation witness as an obstruction rather than a hidden success.*

Proof. Project the carrier of Definition 7167.1. The packet exposes only T, E, D, O, H, C, P, N.

The finite candidate table T supplies the traces that may be inspected. Endpoint checking is confined to E, so a consumer sees only bounded normal endpoint witnesses and cannot infer a global normalization theorem.

The discharge row D records which subject-reduction, closedness, and proof-obligation sockets are filled. If any socket or endpoint is absent, the obstruction row O is the exported frontier, not a replacement proof.

Finally H, C, P, N transport, replay, source, and name the same finite sieve. Because the carrier contains no global confluence, subject-reduction, quotient trace equality, or oracle-discharge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7167.3 (Finite proof-trace sieve frontier handoff)**.** *Every consumer read from an accepted* $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ *packet factors through the route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow},\ \mathsf{BoundedNormalEqualityChecker}^{\uparrow},\ \mathsf{MetaCICNormalizationFrontier}^{\uparrow},\ T, E, D, O, H, C, P, N.$$

The route first enumerates finite proof-trace candidates, then checks bounded normal endpoints, then inspects discharge sockets and closure-preservation traces, and finally exports either a replayable candidate row or the displayed obstruction row. It does not export global normalization, unrestricted confluence, closed subject reduction, quotient equality of proof traces, hidden oracle discharge, or host equality of MetaCIC terms.

Lean4 ✓ `BEDC.Derived.FiniteProofTraceSieveUp.FiniteProofTraceSieveFrontierHandoff`

Proof. Begin with a consumer asking whether a proof-search candidate can be consumed. Carrier admission restricts the request to the finite table T, so only the listed trace candidates and dependency windows are visible.

Next the endpoint row E is read through the bounded normal-equality checker and normalization frontier. This gives bounded endpoint evidence for displayed candidates only, not a normalization theorem for all closed terms.

The discharge row D is then inspected together with the closure-preservation trace. A filled socket can be replayed as part of the finite candidate route; an unfilled socket is recorded by O as the frontier handoff.

Transport, replay, provenance, and naming preserve this same sieve. Since the packet contains no global confluence, subject-reduction, quotient trace equality, oracle discharge, or host equality coordinate, every consumer receives either the finite replayable rows or the explicit obstruction row. □

Falsifiable prediction: Every accepted $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ read exposes finite candidate proof traces, bounded normal endpoints, discharge sockets, obstruction rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any proof-search consumer can inspect a candidate endpoint.

Independence witness: metaciccriticalpath, bounded-normal-equality-checker, metacicnormalizationfrontier, prooftraceauditwitness, closurepreservationtrace: $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite candidate trace tables, bounded endpoint checks, discharge sockets, obstruction retention, closure-preservation replay, transport, provenance, and local naming while refusing global normalization, confluence, subject reduction, quotient trace equality, and hidden oracle discharge.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteProofTraceSieve}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ packet is generated from finite candidate proof traces, bounded normal endpoint rows, discharge sockets, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteProofTraceSieve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7167.1, Definition 7167.2, and Definition 7167.3 seed the finite MetaCIC proof-search sieve over candidate, endpoint, discharge, obstruction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteProofTraceSieve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, unrestricted confluence, closed subject reduction, quotient equality of proof traces, hidden oracle discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration covering carrier admission, finite candidate enumeration, bounded endpoint checking, discharge-socket inspection, obstruction retention, transport, replay, provenance, and local naming.

Chapter 7168

A Concrete Naming Certificate for $\mathsf{BishopIntervalBisection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalBisection}^{\uparrow}$ packet names the finite constructive bisection route by which a located rational interval is repeatedly halved, inspected through rational locator rows, repacked as a regular rational readback, and sealed only at the terminal real-name boundary. It is the Bishop-style neighbor of chapter 394, chapter 1711, and chapter 4202: the packet consumes located endpoint rows, rational endpoint bracketing, dyadic midpoint cells, finite window schedules, and regular rational readbacks before any $\mathsf{Real}^{\uparrow}$ consumer may inspect the result.

Definition 7168.1 (Bishop interval bisection carrier)**.** *A* $\mathsf{BishopIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row. It supplies the located endpoint surface and the endpoint-window discipline consumed from chapter 394.*
2. *Q is the* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *bracketing row. It records the rational endpoint cells and comparison boundary consumed from chapter 4202.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and radius ledger for the requested finite bisection depth.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that records which endpoint and midpoint observations are inspected at each finite branch.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained rational cells and stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is read only after L, Q, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the located interval, rational locator, dyadic midpoint ledger, stream windows, readback, and seal rows.*
8. *C records displayed* Cont *replay from the bisection request through rational bracketing, midpoint halving, retained-window readback, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalBisection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalBisection}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite Bishop interval bisection packet. It has no coordinate for a selected completed endpoint, quotient real equality, host interval equality, open-cover compactness, countable choice, or an ambient completeness theorem.

Theorem 7168.2 (Bishop interval bisection halving stability)**.** *Let $B = (L, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopIntervalBisection}^{\uparrow}$ *carrier. Every finite halving step requested by the packet is stable under the displayed* hsame *transport and* Cont *replay rows: the located interval row L, rational locator row Q, dyadic midpoint ledger D, stream-window schedule S, regular readback R, and terminal seal E move as one packet, and the retained child interval remains attached to the same rational bracketing route.*

Proof. Project the packet through Definition 7168.1. The halving step has no independent host interval coordinate; it can read only the located endpoint row L, the rational locator row Q, and the dyadic midpoint ledger D.

Transport is componentwise. The row H moves L, Q, D, S, R, E together, while the row C replays the same bisection route after transport. Thus the retained child interval is still the child selected by the displayed rational bracketing and midpoint rows.

The stream and readback rows are downstream of that retained child. Since S and R are transported with the same history and provenance, the finite halving route remains stable before the terminal $\mathsf{Real}^{\uparrow}$ seal is inspected. □

Theorem 7168.3 (Bishop interval bisection Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopIntervalBisection}^{\uparrow}$ *carrier factors through the finite route*

$$L, Q, D, S, R, E, H, C, P, N.$$

The consumer first reads the located interval surface, then the rational interval locator, then the dyadic halving ledger, then the finite stream windows and $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The packet exports no selected limit, no quotient equality of real names, no countable-choice sequence, no open-cover compactness witness, and no ambient real-completeness coordinate.*

Proof. Begin with the displayed carrier. The only Real-facing coordinate is E, and the definition places E after the located interval, rational locator, dyadic midpoint, stream-window, and regular-readback rows.

The rational locator row Q supplies endpoint bracketing before any midpoint branch is retained. The dyadic ledger D records the finite halving data, and the stream schedule S records exactly which finite observations are inspected.

The regular readback R is therefore the last non-Real row on the route. Componentwise transport, replay, provenance, and local naming are exactly H, C, P, N, so no host analytic object can be inserted between R and the terminal seal E. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalBisection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7168.1, Definition 7168.2, and Definition 7168.3 seed the finite Bishop interval bisection surface over located interval, rational locator, dyadic midpoint, stream-window, regular-readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient Real equality, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, halving stability, regular-readback routing, Real-seal nonescape, and the corresponding BEDC.Derived.BishopIntervalBisectionUp target.

Constructive story: A BishopIntervalBisection$^{\uparrow}$ packet is generated from a LocatedInterval$^{\uparrow}$ source row, RationalIntervalLocator$^{\uparrow}$ rational bracketing, DyadicRatCore$^{\uparrow}$ midpoint data, StreamName$^{\uparrow}$ finite windows, RegSeqRat$^{\uparrow}$ readback, a terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7169

A Concrete Naming Certificate for $\mathsf{RegulatedStepFunction}^{\uparrow}$

Following Bishop and Bridges on regulated functions and Riemann integration, the $\mathsf{RegulatedStepFunction}^{\uparrow}$ packet records the finite step ledger that sits between the single-step partition surface and the regulated or Riemann integration consumers. It refines the finite $\mathsf{StepFunction}^{\uparrow}$ route of chapter 3780, remains readable by the regulated-function approximation surface of chapter 1554, and supplies the step-integral primitive consumed by chapter 1557 and chapter 1526.

Definition 7169.1 (Regulated step function carrier)**.** *A* $\mathsf{RegulatedStepFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, A, J, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval source row, naming the closed domain, endpoint order data, and finite cell boundary reads used by every step request.*
2. *S is a finite* $\mathsf{StepFunction}^{\uparrow}$ *schedule, read through chapter 3780, listing the displayed partition cells, carried* $\mathsf{Real}^{\uparrow}$ *values, and constant-on-cell admission data.*
3. *A is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *approximation row, read through chapter 1554, stating that the same finite step schedule is accepted as a regulated-function approximation surface.*
4. *J is the primitive step-integral ledger, computing finite cell sums from I and S without importing a limiting integration theorem.*
5. *D is the Darboux/Riemann compatibility row, recording that the primitive ledger fits the Darboux gap route and the tagged-mesh route before any integration-facing handoff is exported.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed finite sums and tolerance windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, S, A, J, D, Q are displayed.*
8. *H records componentwise* hsame *transport for interval, schedule, regulated approximation, primitive step-integral, compatibility, readback, seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from interval reads to step schedule reads, from step schedules to primitive sums, from compatibility rows to Darboux and Riemann consumers, and from readback rows to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RegulatedStepFunction}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedStepFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these rows. It has no coordinate for a host function quotient, selected refinement limit, measure completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7169.2 (Regulated step function NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedStepFunction}^{\uparrow}$ *carrier $K = (I, S, A, J, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedStepFunction}^{\uparrow}}$ *surface consists exactly of interval source admission, finite* $\mathsf{StepFunction}^{\uparrow}$ *schedule exposure,* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *approximation admission, primitive step-integral computation, Darboux/Riemann compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7169.1. The rows I, S, A display the interval source, the finite step schedule, and the regulated-function approximation surface before any integration-facing consumer can inspect a value.

The integration rows are downstream of that displayed surface. The primitive ledger J computes only finite cell sums from I and S, while D records that the same ledger is readable by Darboux and Riemann consumers.

The readback and structural rows preserve the same route. The rows Q, E, H, C, P, N add regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming, but no hidden function quotient, selected refinement, measure-completion object, compactness proof, or ambient completeness coordinate. □

Theorem 7169.3 (Regulated step function Riemann handoff)**.** *Every Riemann-facing or regulated-integral-facing read of an accepted* $\mathsf{RegulatedStepFunction}^{\uparrow}$ *packet factors through*

$$\mathsf{StepFunction}^{\uparrow}, \quad \mathsf{RegulatedRealFunction}^{\uparrow}, \quad \mathsf{RegulatedIntegral}^{\uparrow}, \quad \mathsf{RiemannIntegral}^{\uparrow}$$

by first reading the finite interval and step schedule rows I, S, then the regulated approximation row A, then the primitive step-integral ledger J, then the Darboux/Riemann compatibility row D, and finally the $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal Q, E. The packet exports a finite regulated step primitive, not a quotient function, a selected mesh limit, a measure-completion integral, or ambient Real completeness.*

Proof. Start with the finite step side. The $\mathsf{StepFunction}^{\uparrow}$ schedule S exposes the cell partition and carried values over the interval row I, so the primitive sum ledger has a finite source.

Next read the regulated-function side. The row A makes the same finite schedule available to the $\mathsf{RegulatedRealFunction}^{\uparrow}$ approximation surface, and no integration row is present before this approximation row is displayed.

Finally read the integration side. The primitive ledger J supplies finite sums, the compatibility row D routes those sums through the regulated-integral and Riemann boundaries, and Q, E provide regular rational readback and the terminal Real seal. Since every remaining coordinate is structural, no consumer can replace this handoff by a host quotient, selected mesh limit, measure completion, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedStepFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegulatedStepFunction}^{\uparrow}$ packet is generated from an interval source row, a finite StepFunction schedule, a RegulatedRealFunction approximation row, a primitive step-integral ledger, a Darboux/Riemann compatibility

row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegulatedStepFunction}\uparrow}$, seed)

Scope closed: Definition 7169.1, Definition 7169.2, and Definition 7169.3 seed the finite regulated step primitive between step functions and regulated or Riemann integration.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegulatedStepFunction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a measure-theoretic integral, quotient function equality, selected refinement limit, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite step schedule exposure, regulated approximation routing, primitive step-integral computation, Darboux/Riemann compatibility, RegSeqRat readback, Real sealing, and structural transport for BEDC.Derived.RegulatedStepFunctionUp.

Chapter 7170

A Concrete Naming Certificate for HyperbolicShadowBoundary$^{\uparrow}$

Following Gromov boundary and visual shadow constructions in hyperbolic geometry, the HyperbolicShadowBounda packet records the observer-facing boundary window as finite shadow and visual-boundary NameCert data. It sits between the boundary-shadow packet of chapter 3813, the visual-boundary readback of chapter 4181, the visual-shadow metric surface of chapter 4142, and the observer-expansion route of chapter 4167. This realizes the observer-shadow portion of chapter 9463 without adding a completed ideal boundary or a host hyperbolic space.

Definition 7170.1 (Hyperbolic shadow boundary carrier)**.** *A* HyperbolicShadowBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, O, W, V, S, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the hyperbolic metric source row, naming the finite metric comparison data used by every shadow read.*
2. *O is the observer window row, recording the displayed basepoint, aperture, and finite visible-shadow request.*
3. *W is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row that supplies the boundary-shadow window.*
4. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *row that reads the same shadow window as visible boundary data.*
5. *S is the* HyperbolicVisualShadowMetric$^{\uparrow}$ *row that records visual-shadow metric comparison before observer expansion consumes the boundary read.*
6. *R is the* HyperbolicObserverExpansion$^{\uparrow}$ *handoff row, read only after M, O, W, V, S are displayed.*
7. *H records componentwise* hsame *transport for metric source, observer window, boundary shadow, visual boundary, visual-shadow metric, observer handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from observer-window reads to boundary-shadow windows, visual-boundary reads, visual-shadow metric comparisons, and observer-expansion handoffs.*
9. *P is the* Pkg *provenance over* HyperbolicShadowBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicVisualBounda HyperbolicVisualShadowMetric$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicShadowBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by these rows. It has no coordinate for a completed ideal boundary, quotient geodesic-end equality, selected geodesic ray, global compactification, or physical observer trajectory.

Theorem 7170.2 (Hyperbolic shadow boundary NameCert obligations)**.** *For every accepted* HyperbolicShadowBou *carrier $B = (M, O, W, V, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicShadowBoundary}^{\uparrow}}$ *surface consists exactly of metric source admission, observer-window exposure,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *window admission,* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *readback,* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *comparison,* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7170.1. The rows M, O, W, V, S, R are the only analytic rows, and H, C, P, N are the only structural rows.

The boundary side is finite. The observer row O exposes the window request, the boundary-shadow row W supplies the visible shadow, and the visual-boundary row V reads that same window as boundary data.

The observer side is downstream. The visual-shadow metric row S compares the visible shadow before the observer-expansion handoff R is available. Transport, replay, provenance, and local naming preserve this route without adding a completed ideal boundary, quotient ray equality, selected geodesic, compactification theorem, or physical observer path. □

Theorem 7170.3 (Hyperbolic shadow boundary observer handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicShadowBoundary}^{\uparrow}$ *packet must read the route*

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualShadowMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicObs}$$

through the displayed observer window and structural rows of Definition 7170.1. The handoff exports finite shadow-window and visual-boundary data for chapter 9463; it exports no completed ideal boundary, quotient geodesic equality, selected ray, global compactification, or physical observer trajectory.

Proof. Begin with the boundary-shadow row. It records the finite visible window, and the visual-boundary row reads the same window as boundary data rather than as a completed ideal boundary.

Next read the visual-shadow metric row. It compares only the displayed shadow and visible-boundary rows, so the observer-expansion handoff receives finite metric readback data rather than a global compactification.

Finally consume the structural rows. Componentwise hsame transport and displayed Cont replay carry the same window through provenance and local naming. Since the carrier has no coordinate for quotient ray equality, selected geodesics, completed boundary objects, or physical observer paths, the observer handoff remains exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicShadowBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicShadowBoundary}^{\uparrow}$ packet is generated from hyperbolic metric source rows, observer-window rows, HyperbolicBoundaryShadow rows, HyperbolicVisualBoundary rows, HyperbolicVisualShadowMetric rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicShadowBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 7170.1, Definition 7170.2, and Definition 7170.3 seed the finite observer-shadow boundary packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicShadowBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed ideal-boundary construction, quotient geodesic equality, selected geodesic rays, global compactification, physical observer dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric source admission, observer-window exposure, boundary-shadow admission, visual-boundary readback, visual-shadow metric comparison, observer-expansion handoff, and structural transport for BEDC.Derived.HyperbolicShadowBoundaryUp.

Chapter 7171

A Concrete Naming Certificate for $\mathsf{RationalNullSequence}^{\uparrow}$

Following Bishop and Bridges on constructive real sequences, the $\mathsf{RationalNullSequence}^{\uparrow}$ packet records a rational sequence whose tail is certified to converge to zero by finite tolerance windows rather than by a selected limit or quotient stream. It sits between the rational source of chapter 37, the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a finite null-tail interface without importing countable choice, quotient sequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7171.1 (Rational null sequence carrier)**.** *A* $\mathsf{RationalNullSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (A, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Rat}^{\uparrow}$ *value row, naming the rational source values used by the sequence.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the positive dyadic precision requests used to test the tail.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, displaying the inspected indices and the tail bound witness for each requested precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same rational values, dyadic tolerances, and finite windows.*
5. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, exposed only after A, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the rational values, tolerance ledger, finite windows, readback, seal boundary, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from the rational source and dyadic request through the finite tail window, regular rational readback, and terminal seal.*
8. *P records* Pkg *provenance over* $\mathsf{RationalNullSequence}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalNullSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, D, W, R, E, H, C, P, N$. It has no coordinate for quotient sequence equality, a chosen limiting index function, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7171.2 (Rational null sequence NameCert obligations)**.** *For every accepted* $\mathsf{RationalNullSequence}^{\uparrow}$ *carrier $Z = (A, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalNullSequence}^{\uparrow}}$ *surface consists exactly of five displayed obligations:*

1. *rational-source admission through A;*
2. *dyadic tolerance admission through D;*
3. *finite tail-window exposure through W;*
4. *regular rational readback through R before the* $\mathsf{Real}^{\uparrow}$ *seal E is visible;*
5. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

Every public read of the rational null sequence must pass through these obligations in this order.

Proof. Project the accepted carrier of Definition 7171.1. The rational values appear only in A, and the requested tail precisions appear only in the dyadic ledger D.

The finite tail evidence is then forced through W. A consumer sees the inspected indices and the displayed tail bound witness before any regular rational readback is available.

Rows R and E remain downstream. The regular rational readback is attached to the same A, D, W route, and the Real seal is exposed only after that readback has been named.

Finally H, C, P, N preserve, replay, source, and name the same packet. Since the carrier has no row for quotient sequence equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the five obligations exhaust the public NameCert surface. □

Theorem 7171.3 (Rational null sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalNullSequence}^{\uparrow}$ *packet must read the rational source row A, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{StreamName}^{\uparrow}$ *finite tail windows W before it receives the readback row R. The handoff exports no quotient sequence equality, chosen tail selector, countable-choice schedule, or ambient completeness principle.*

Proof. Use the obligation surface of Definition 7171.2. It places R after the rational source, dyadic tolerance ledger, and finite tail-window rows.

The handoff is therefore finite at each request. The consumer reads the specific rational values and the displayed dyadic precision rows used by the tail window; it cannot replace them by a quotient stream or by a selected global limiting witness.

The transport and provenance rows keep the same order. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ readback is a finite rational-tail interface and not an import of countable choice or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7171.4 (Rational null sequence Real seal nonescape)**.** *An accepted* $\mathsf{RationalNullSequence}^{\uparrow}$ *packet exposes its* $\mathsf{Real}^{\uparrow}$ *seal boundary only after the rational source row, dyadic tolerance ledger, finite StreamName tail windows, and RegSeqRat readback have been displayed. Any consumer route that tries to read the Real seal while omitting one of A, D, W, R leaves the accepted NameCert surface.*

Proof. Apply Definition 7171.1. The Real-facing row is E, and the carrier definition places it downstream of the rational, dyadic, StreamName, and RegSeqRat rows.

The NameCert obligations of Definition 7171.2 make that order public: A, D, W are exposed before R, and R is exposed before E.

If a consumer omits any of A, D, W, R, the structural rows H, C, P, N have no displayed packet to transport, replay, source, or name. The route therefore cannot export a valid Real seal, and it cannot smuggle host real equality, selected limits, or ambient completeness through the seal boundary. □

Falsifiable prediction: A route that reads a Real seal for a rational null sequence while omitting the finite StreamName tail window or the RegSeqRat readback is rejected by the RationalNullSequenceUp NameCert surface.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalNullSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds rational values, dyadic tolerances, finite tail windows, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming as one null-tail packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalNullSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalNullSequence}^{\uparrow}$ packet is generated from rational source values, DyadicRatCore tolerance requests, StreamName finite tail windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalNullSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7171.1, Definition 7171.2, Definition 7171.3, and Definition 7171.4 seed the rational null-tail route over $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalNullSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rational-source admission, dyadic tolerance admission, finite tail-window exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalNullSequenceUp.

Chapter 7172

A Concrete Naming Certificate for $\mathsf{CauchyOrderReflection}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of Cauchy real names, the $\mathsf{CauchyOrderReflection}^{\uparrow}$ packet records the finite route by which Cauchy-tail comparisons are reflected into Real-facing order reads. It sits after the Cauchy criterion surface of chapter 392, the regular rational sequence row of chapter 301, the order interfaces of chapter 1378 and chapter 54, the dyadic arithmetic core of chapter 291, and the terminal Real seal of chapter 38. The packet exposes only displayed tail windows, dyadic comparison ledgers, order rows, and Real-facing seals; it does not assert trichotomy, Dedekind completeness, quotient real classes, selected representatives, or host $\mathsf{Real}^{\uparrow}$ equality.

Definition 7172.1 (CauchyOrderReflection carrier)**.** *A* $\mathsf{CauchyOrderReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (C, T, O, D, R, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *row recording the accepted tail-window request and its finite Cauchy evidence.*
2. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row that turns the accepted Cauchy window into regular rational read-back data.*
3. *O is the* $\mathsf{Order}^{\uparrow}$ *or* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row used only after C and T have been displayed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger for the inspected dyadic tolerances and rational cuts.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row that records the Real-facing order read after the finite tail comparison has passed through O and D.*
6. *H records componentwise* hsame *transport for Cauchy criteria, regular-tail rows, order rows, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a Real-facing order request back through R, D, O, T, C and then forward through the same rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyOrderReflection}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*,* $\mathsf{TotalOrder}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyOrderReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed coordinates. It contains no host quotient of Cauchy names, no chosen representative, no trichotomy branch, no Dedekind-completeness row, and no ambient host $\mathsf{Real}^{\uparrow}$ *equality.*

Theorem 7172.2 (CauchyOrderReflection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyOrderReflection}^{\uparrow}$ *carrier*

$$Q = (C, T, O, D, R, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyOrderReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy criterion row, RegSeqRat tail row, order comparison row, dyadic comparison ledger, Real seal row, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming row; tail-stability of accepted comparisons; and ledger nonescape for quotient real classes, selected representatives, trichotomy, Dedekind completeness, and host Real equality.*

Proof. Project the carrier of Definition 7172.1. The visible rows are $C, T, O, D, R, H, K, P, N$, and every NameCert obligation is read from one of those rows.

Carrier admission records the Cauchy criterion, regular-tail readback, order comparison, dyadic ledger, Real seal, transport, replay, provenance, and local name. No other analytic row is present in the packet.

Tail-stability is the compatibility of C, T, O, D, R under the componentwise transport H and replay K. The nonescape clauses follow by inventory: quotient classes, selected representatives, trichotomy, Dedekind completeness, and host Real equality are not coordinates of the carrier. □

Theorem 7172.3 (CauchyOrderReflection tail stability)**.** *Let* $Q = (C, T, O, D, R, H, K, P, N)$ *be an accepted* $\mathsf{CauchyOrderReflection}^{\uparrow}$ *carrier. Any accepted replacement of a Cauchy tail window transported by* H *and replayed by* K *preserves the displayed order read only through the same* C, T, O, D, R *route. The reflected Real-facing comparison is stable as finite tail and dyadic comparison data, not as host equality of completed Real objects.*

Proof. Start with the Cauchy criterion row C. It names the accepted tail window and supplies the finite Cauchy evidence used by the comparison request.

Read the regular-tail row T and the dyadic ledger D. These rows translate the tail replacement into finite regular rational and dyadic comparison data before any order row is consumed.

Now inspect O and R. The order comparison row reads only the displayed finite comparison, and the Real seal records the reflected comparison only after that route is visible.

Transport by H is componentwise and replay by K follows the same displayed route. Since no coordinate stores host equality of completed Real objects, the stable comparison remains a finite $\mathsf{CauchyOrderReflection}^{\uparrow}$ read. □

Theorem 7172.4 (CauchyOrderReflection nonescape)**.** *Every Real-facing order consumer of an accepted* $\mathsf{CauchyOrderReflection}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Order}^{\uparrow},\ \mathsf{TotalOrder}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer may inspect Cauchy-tail windows, regular rational readback, order comparison rows, dyadic comparison ledgers, Real seals, transport, replay, provenance, and local naming, but it cannot obtain trichotomy, Dedekind completeness, quotient classes, selected representatives, or host Real equality from this packet.

Proof. Apply Definition 7172.2. It lists the admitted rows of the carrier and the only stability and nonescape obligations attached to the local name.

Use Definition 7172.3. A tail replacement reaches a Real-facing comparison only through the displayed C, T, O, D, R route, so an order consumer cannot bypass the Cauchy and dyadic windows by invoking a host quotient or selected representative.

The remaining exclusions are carrier inventory. The packet has no trichotomy branch, Dedekind-completeness coordinate, quotient-class coordinate, representative selector, or host Real-equality row, so no Real-facing order read can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyOrderReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyOrderReflection}^{\uparrow}$ packet is generated from a CauchyCriterion row, a RegSeqRat tail row, an Order or TotalOrder comparison row, a DyadicRatCore comparison ledger, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyOrderReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7172.1, Definition 7172.2, Definition 7172.3, and Definition 7172.4 seed the finite route from Cauchy-tail comparisons to Real-facing order reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyOrderReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, Dedekind completeness, quotient classes, selected representatives, host Real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-stability, order-read reflection, dyadic comparison exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyOrderReflectionUp.

Chapter 7173

A Concrete Naming Certificate for $\mathsf{CompactOscillationSelector}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompactOscillationSelector}^{\uparrow}$ packet records the finite selector route behind compact uniform continuity as one BEDC NameCert carrier. It realizes the compact-uniform-continuity vision of chapter 9237 by reading compact finite nets from chapter 129, pointwise radius rows from chapter 130, rational lower-bound folds from chapter 37, metric distance rows from chapter 57, terminal Real seals from chapter 38, finite stream windows from chapter 126, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. The packet selects only by finite displayed rows; it does not expose host open-cover compactness, semantic continuity, selected centers outside the finite net, a quotient point equality, an external modulus selector, or ambient completeness.

Definition 7173.1 (Compact oscillation selector carrier)**.** *A* $\mathsf{CompactOscillationSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, L, M, R, E, W, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying bundled centers and located coverage witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-radius row, attaching a local oscillation estimate and source radius to each bundled center.*
3. *L is the rational lower-bound fold row over the displayed local radii, read through* $\mathsf{Rat}^{\uparrow}$*.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to route source and target triangle estimates.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing distance and oscillation reads.*
6. *E is the dyadic or rational tolerance ledger that connects requested oscillation bounds to the lower-bound fold.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for center names, source points, metric reads, and Real seals.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported only after K, F, L, M, R, E, W have been displayed.*
9. *H records componentwise* hsame *transport for compact-net, pointwise-radius, lower-bound, metric, Real, tolerance, stream-window, uniform-modulus, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tolerance request through finite net selection, pointwise-radius rows, rational lower-bound fold, metric triangle estimates, Real sealing, and the uniform-modulus handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationSelector}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationSelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host compactness, semantic continuity, quotient point equality, choice-selected centers outside the displayed net, an external modulus selector, or ambient completeness.

Theorem 7173.2 (Compact oscillation selector NameCert obligations)**.** *For every accepted* CompactOscillationSele *carrier*

$$S = (K, F, L, M, R, E, W, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationSelector}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, pointwise-radius admission, rational lower-bound fold exactness, metric triangle-read admission, Real-seal nonescape, tolerance-ledger visibility, finite stream-window readback, UniformModulus handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. The obligations do not include host open-cover compactness, semantic continuity, quotient point equality, choice-selected centers outside the finite net, external modulus selection, or ambient completeness.*

Proof. Project the carrier of Definition 7173.1. The displayed rows $K, F, L, M, R, E, W, U, H, C, P, N$ are the whole accepted packet.

Compact finite-net admission comes from K, and pointwise-radius admission comes from F. The lower-bound row L folds the finitely many local radii read from those rows, while E records the requested tolerance scale.

The metric row M, Real seal R, and stream-window row W provide the only route for source and target triangle estimates. The uniform-modulus row U is exported only after those finite rows are visible.

The structural rows H, C, P, N preserve, replay, source, and name the same finite selector route. Since the refused objects are not carrier coordinates, the listed obligations exhaust the local NameCert surface. □

Theorem 7173.3 (Compact oscillation selector finite-net fold)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationSelector}^{\uparrow}$ *packet receives a precision row only after the finite net K, pointwise-radius row F, rational lower-bound fold L, metric row M, Real seal R, tolerance ledger E, and stream-window row W have been displayed. The handoff is the finite net, pointwise-radius, rational lower-bound fold, and Real-seal route, not an external compactness or continuity theorem.*

Proof. Start with the compact finite-net row K. It supplies the finite bundled centers and coverage witnesses available to the selector.

Read the pointwise-radius row F at each bundled center. These rows are finite local oscillation records, so they do not invoke semantic continuity outside the displayed carrier.

Fold the displayed local radii through L using the tolerance ledger E. The result is a rational lower-bound history carried by the packet rather than a chosen external modulus.

Finally replay the metric and Real-facing estimates through M, R, W and hand them to U. Transport and replay through H and C keep the same finite route, so the consumer receives only the displayed selector data. □

Theorem 7173.4 (Compact oscillation selector nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{CompactOscillationSelector}^{\uparrow}$ *packet remains inside the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow}.$$

The consumer may inspect finite nets, pointwise radii, rational lower-bound folds, metric triangle rows, Real seals, finite stream windows, uniform-modulus handoff rows, transport, replay, provenance, and local naming, but it cannot obtain host open-cover compactness, semantic continuity, quotient point equality, choice-selected centers outside the displayed net, an external modulus selector, or ambient completeness from this packet.

Proof. Apply Definition 7173.2. It lists the finite analytic and structural rows that form the local NameCert surface.

Use Definition 7173.3. Every UniformModulus-facing read consumes the compact net, pointwise-radius rows, rational lower-bound fold, metric row, Real seal, tolerance ledger, and stream-window row before the handoff row is public.

The nonescape conclusion is the carrier inventory. None of the refused objects is a coordinate of S, and the transport and replay rows are componentwise over the displayed route, so no downstream consumer can acquire them from $\mathsf{CompactOscillationSelector}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationSelector}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap pointwise-radius rows, a RatUp lower-bound fold, Metric triangle rows, RealUp seals, tolerance and StreamName windows, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactOscillationSelector}^{\uparrow}}$, seed)

Scope closed: Definition 7173.1, Definition 7173.2, Definition 7173.3, and Definition 7173.4 seed the finite selector route from compact nets and pointwise radii to UniformModulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactOscillationSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, semantic continuity, quotient point equality, choice-selected centers outside the finite net, external modulus selection, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, pointwise-radius admission, lower-bound fold exactness, metric triangle reads, Real-seal nonescape, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationSelectorUp.

Chapter 7174

A Concrete Naming Certificate for $\mathsf{CompactOscillationVariation}^{\uparrow}$

Following Bishop–Bridges compact metric analysis and Jordan's finite-variation ledger, the CompactOscillationVariat packet records the finite route by which compact finite-net data synchronizes bounded oscillation windows with bounded-variation partition rows. It sits over the compact metric source of chapter 129, the bounded oscillation ledger of chapter 6787, the bounded-variation real-function route of chapter 3654, the Real oscillation interface of chapter 1839, the variation-measure ledger of chapter 3593, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The chapter names a compact finite-net bridge for Darboux and Riemann–Stieltjes consumers; it does not assert open-cover compactness, host total-variation suprema, measure completion, quotient function equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7174.1 (CompactOscillationVariation carrier)**.** *A* $\mathsf{CompactOscillationVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, O, V, R, E, H, C, P, N),$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the center-radius windows through which compact consumers read the inspected domain;*
2. *O is the* $\mathsf{BoundedOscillation}^{\uparrow}$ *row, exposing bounded oscillation windows over the same compact finite-net cells;*
3. *V is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *partition ledger, read as finite variation data over the displayed compact source;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the oscillation and variation windows;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal attached to endpoint and value reads;*
6. *H records the componentwise* hsame *transport rows for compact-net, oscillation, variation, readback, and seal coordinates;*
7. *C records the displayed* Cont *replay rows through which compact, oscillation, variation, RegSeqRat, and Real consumers read the same packet;*
8. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BoundedOscillation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactOscillationVariation}^{\uparrow}}$ *row.*
The classifier accepts exactly those packets whose compact finite-net row, oscillation windows, variation partition ledger, regular-rational readback, Real seal, transport, replay, provenance, and local naming rows are all displayed in these positions. It exports no open-cover compactness theorem, total-variation supremum, measure completion, quotient equality of functions, selected limit, or ambient Real-completeness row.

Theorem 7174.2 (CompactOscillationVariation NameCert obligations)**.** *For every accepted* CompactOscillationVar *carrier $Q = (K, O, V, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationVariation}^{\uparrow}}$ *surface consists exactly of the finite compact-net source row K, bounded oscillation window row O, bounded-variation partition ledger V, regular-rational readback row R, terminal Real seal E, componentwise transport row H, replay row C, package provenance row P, and local naming row N.*

Every Darboux-facing, variation-facing, or Riemann–Stieltjes-facing consumer must read the compact finite-net row before reading the oscillation and variation rows, and must read R, E before treating endpoint or value data as Real-facing output. No consumer receives a host compactness theorem, a total-variation supremum, a completed measure object, quotient function equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7174.1. Its rows list exactly the compact source, oscillation windows, variation partition ledger, readback, Real seal, transport, replay, provenance, and local naming coordinates available to the classifier.

The compact row K is the only source of center-radius finite-net data. Thus any oscillation read through O and any variation read through V must remain attached to the displayed compact finite-net route before a downstream consumer can inspect the ledger.

The rows R and E then provide the regular-rational and Real-facing boundary for window values and endpoints. Since the carrier has no coordinate for an open-cover theorem, total-variation supremum, completed measure, quotient function equality, selected limit, or ambient completeness, the local NameCert obligations are exhausted by the stated finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationVariation}^{\uparrow}$ packet is generated from compact finite-net rows, bounded oscillation windows, bounded-variation partition ledgers, RegSeqRat readback, Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactOscillationVariation}^{\uparrow}}$, seed)

Scope closed: Definition 7174.1 and Definition 7174.2 seed the finite compact-net bridge from bounded oscillation windows to bounded-variation partition ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactOscillationVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host total-variation suprema, measure completion, quotient function equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net synchronization, oscillation-window confinement, variation-ledger readback, RegSeqRat and Real seal exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationVariationUp.

Chapter 7175

A Concrete Naming Certificate for $\mathsf{FiniteOscillationVariation}^{\uparrow}$

Following Jordan and Bishop-style finite variation arguments, the $\mathsf{FiniteOscillationVariation}^{\uparrow}$ packet records the finite route by which a displayed partition controls both oscillation windows and variation budgets before any compact or completion-facing consumer reads the data. It sits beside the bounded-variation surface of chapter 398, the compact synchronization route of chapter 7174, the bounded oscillation ledger of chapter 6787, the bounded-variation real-function route of chapter 3654, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names only the finite partition, local oscillation windows, edge-variation ledger, readback, Real seal, transport, replay, provenance, and local naming rows; it does not assert a host total-variation supremum, selected refinements outside the displayed partition, function quotient equality, measure completion, compact open-cover extraction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7175.1 (FiniteOscillationVariation carrier)**.** *A* $\mathsf{FiniteOscillationVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, O, V, R, E, H, C, S, N),$$

with the following visible rows:

1. *P is the finite partition row, including the displayed endpoints and adjacent edge windows inspected by the packet.*
2. *O is the finite oscillation row, recording an oscillation budget for each displayed partition cell.*
3. *V is the finite edge-variation ledger, recording the adjacent value changes consumed by bounded-variation arguments.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the oscillation and variation budgets.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint, value, oscillation, and variation reads.*
6. *H records componentwise* hsame *transport for partition, oscillation, variation, readback, seal, replay, provenance, and local naming coordinates.*
7. *C records displayed* Cont *replay through which finite-partition consumers read the same packet.*
8. *S is the* Pkg *provenance over* $\mathsf{FiniteOscillationVariation}^{\uparrow}$, $\mathsf{BoundedOscillation}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationVariation}^{\uparrow}}$ *row.*

The classifier accepts exactly those packets whose finite partition, oscillation budget, variation ledger, readback, Real seal, transport, replay, provenance, and local naming rows are displayed in these positions. It exports no host total-variation supremum, unlisted partition refinement, quotient equality of functions, measure completion, compact open-cover theorem, selected limit, or ambient Real-completeness row.

Theorem 7175.2 (FiniteOscillationVariation NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationVariation}^{\uparrow}$ *carrier* $F = (P, O, V, R, E, H, C, S, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationVariation}^{\uparrow}}$ *surface consists exactly of the finite partition row* P*, oscillation budget row* O*, edge-variation ledger* V*, regular-rational readback row* R*, terminal Real seal* E*, componentwise transport row* H*, replay row* C*, package provenance row* S*, and local naming row* N*.*

Every bounded-variation, Darboux-facing, or compact-oscillation consumer must read the finite partition row before reading O *or* V*, and must read* R, E *before treating any endpoint, value, oscillation, or variation data as Real-facing output. No consumer receives a host total-variation supremum, an unlisted refinement selector, quotient equality of functions, measure completion, compact open-cover extraction, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7175.1. Its rows list exactly the finite partition, oscillation budget, edge-variation ledger, readback, Real seal, transport, replay, provenance, and local naming coordinates available to the classifier.

The partition row P is the only source of displayed edge windows. Thus any oscillation read through O and any variation read through V remains attached to the same finite partition before a downstream bounded-variation or Darboux-facing consumer can inspect the ledger.

The rows R and E then provide the regular-rational and Real-facing boundary for endpoint and value data. Since the carrier has no coordinate for a host supremum, refinement selector, quotient function equality, completed measure, compact open-cover theorem, selected limit, or ambient completeness, the local NameCert obligations are exhausted by the stated finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationVariation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOscillationVariation}^{\uparrow}$ packet is generated from finite partition rows, oscillation budgets, edge-variation ledgers, RegSeqRat readback, Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteOscillationVariation}^{\uparrow}}$, seed)

Scope closed: Definition 7175.1 and Definition 7175.2 seed the finite partition route from oscillation budgets to variation ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteOscillationVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host total-variation suprema, unlisted refinement selectors, quotient equality of functions, measure completion, compact open-cover extraction, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-window synchronization, oscillation-budget confinement, variation-ledger readback, RegSeqRat and Real seal exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOscillationVariationUp.

Chapter 7176

A Concrete Naming Certificate for $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Heine–Cantor finite-net route, the $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ packet records the finite oscillation-to-uniform-modulus handoff for compact sources. It sits after the compact finite-net surfaces of chapter 60 and chapter 129, the pointwise continuity surfaces of chapter 59 and chapter 130, the oscillation rows of chapter 1839 and chapter 2224, the tolerance readbacks of chapter 291, chapter 301, and chapter 38, and the downstream uniform-modulus packet of chapter 369. The packet names the router by which compact finite-net data and pointwise oscillation ledgers become a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ row; it does not assert open-cover compactness, semantic continuity, quotient function equality, selected centers outside the finite net, an external modulus oracle, or ambient Real completeness.

Definition 7176.1 (Compact modulus oscillation router carrier)**.** *A* $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, M, F, O, Q, D, S, A, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *admission row naming the compact source as BEDC data.*
2. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying bundled centers, coverage witnesses, and the finite net inspected by the router.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise continuity row at each displayed center.*
4. *O is the* $\mathsf{RealOscillation}^{\uparrow}$ *pointwise oscillation ledger read from the same center and tolerance data.*
5. *Q is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row converting the displayed oscillation ledger into a tolerance-to-window modulus route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the pointwise oscillation and modulus rows.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the rational windows opened by D, Q.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for source and target distance or oscillation comparisons.*
9. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, exposed only after K, M, F, O, Q, D, S, A have been displayed.*

10. *H records componentwise* hsame *transport for compact, compact-metric, continuity, oscillation, modulus, dyadic, regular, Real, uniform-modulus, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a target tolerance request through the compact finite net, pointwise continuity rows, oscillation ledger, oscillation modulus, dyadic and regular readback, Real seal, and uniform-modulus handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusOscillationRouter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, semantic continuity outside the displayed pointwise rows, quotient equality of functions or points, selected centers outside M, an external modulus oracle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactModulusOscillationRouterUp.TasteGate.CompactModulusOscillationRouterTasteGa`
`single carrier alignment`

Theorem 7176.2 (Compact modulus oscillation router NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ *carrier*

$$R = (K, M, F, O, Q, D, S, A, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusOscillationRouter}^{\uparrow}}$ *surface consists exactly of compact admission, compact-metric finite-net admission, pointwise continuity admission, pointwise oscillation exposure, oscillation-modulus exactness, dyadic tolerance visibility, regular readback stability, Real-seal nonescape, UniformModulus handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 7176.1. The rows $K, M, F, O, Q, D, S, A, U, H, C, P, N$ are the whole accepted packet.

Compact admission and finite-net admission are read from K and M. The pointwise continuity row F is then inspected only at the displayed centers and windows supplied by those compact rows.

The oscillation route is carried by O, Q, D, S, A. The ledger O records the pointwise oscillation data, Q turns it into a displayed modulus route, D supplies the dyadic tolerance, S supplies regular rational readback, and A seals the Real-facing comparison.

The uniform-modulus handoff U is downstream of those rows. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, so the listed obligations exhaust the local NameCert surface and none of the refused host objects is exported. □

Theorem 7176.3 (Compact modulus oscillation router handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ *packet receives a tolerance-only ledger only after reading the finite route*

$\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

The handoff is the displayed compact finite-net and pointwise oscillation route, not an open-cover compactness theorem, semantic continuity principle, quotient equality, selected external center, external modulus oracle, countable-choice schedule, or ambient Real-completeness claim.

Proof. Start with the compact and metric rows K, M. They provide the finite centers and coverage witnesses available to the router.

Read the pointwise continuity and oscillation rows F, O at those centers. The oscillation-modulus row Q converts the displayed oscillation ledger into a tolerance-to-window route over the dyadic row D and regular readback row S.

The Real seal A records the numerical comparison reached by the same route. Only then is the $\mathsf{UniformModulus}^{\uparrow}$ row U visible to a downstream consumer.

The structural rows H, C, P, N transport, replay, source, and name that finite route. Since the packet contains no coordinate for the excluded host objects, every uniform-modulus consumer remains inside the displayed compact finite-net and pointwise oscillation handoff. □

Falsifiable prediction: Every accepted $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ read factors through CompactUp admission, CompactMetric finite-net rows, Continuous or ContinuousMap pointwise rows, RealOscillation and OscillationModulus ledgers, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seals, UniformModulusUp handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: compact, compactmetric, continuous, continuousmap, real-oscillation, oscillation-modulus, dyadicratcore, regseqrat, real, uniformmodulus: none is bijective with CompactModulusOscillat because this packet alone binds the compact finite-net route to pointwise oscillation and a downstream tolerance-only uniform-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusOscillationRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusOscillationRouter}^{\uparrow}$ packet is generated from Compact and CompactMetric rows, Continuous and ContinuousMap pointwise rows, RealOscillation and OscillationModulus ledgers, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seals, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusOscillationRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7176.1, Definition 7176.2, and Definition 7176.3 seed the compact finite-net and pointwise oscillation route before UniformModulus consumers inspect the tolerance-only ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusOscillationRouter}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactModulusOscillationRouterUp.TasteGate.CompactModulusOscillationRouterTasteGate\_s`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, semantic continuity, quotient function or point equality, selected centers outside the finite net, external modulus selection, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact admission, finite-net admission, pointwise continuity admission, oscillation exposure, oscillation-modulus exactness, tolerance visibility, regular readback stability, Real-seal nonescape, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusOscillationRouterUp

Chapter 7177

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ packet records the object-level completion carrier before theorem-facing dense-embedding, universal-property, functoriality, or $\mathsf{Real}^{\uparrow}$ sealing routes are consumed. It sits over the regular rational source of chapter 301, the finite observation surface of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the completion object as displayed regular Cauchy data with transport, replay, provenance, and local naming; it does not assert quotient completion equality, select limits, identify host real equality, invoke countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7177.1 (Regular Cauchy completion object carrier)**.** *A* $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R, W, D, E, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name and its regularity readback surface.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row, naming the windows opened by completion-facing requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, slack cells, and comparison rows used by the regular Cauchy read.*
4. *E is the object-admission row saying that R, W, D jointly form the displayed completion carrier before any theorem route is inspected.*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary row available to Real-facing consumers only as a downstream seal over R, W, D, E.*
6. *H records componentwise* hsame *transport for the regular source, window, dyadic tolerance, object admission, Real boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through W, D, R, E, A and the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionObject}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, W, D, E, A, H, C, P, N$. It has no coordinate for quotient streams, selected representatives, selected limits, countable-choice schedules, host real equality, theorem-level universal properties, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7177.2 (Regular Cauchy completion object NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ *carrier*

$$X = (R, W, D, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionObject}^{\uparrow}}$ *surface consists exactly of regular rational source admission, finite StreamName observation, DyadicRatCore tolerance visibility, object-admission exactness, Real-boundary nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows. Dense-embedding, universal-property, functoriality, and Real-sealing chapters may cite this object only through those rows.*

Proof. Project the carrier of Definition 7177.1. The rows $R, W, D, E, A, H, C, P, N$ are the full accepted packet.

The object rows are R, W, D, E. The regular rational source, finite observation windows, and dyadic tolerance ledger are displayed before the admission row E may name the completion carrier.

The Real-facing row A is downstream of the same displayed data. It seals the route for consumers of chapter 38 without adding quotient equality, a selected representative, a selected limit, or ambient completeness.

The structural rows H, C, P, N transport, replay, source, and name only that finite route. Therefore the listed obligations exhaust the local NameCert surface and give theorem-facing consumers a single object-level carrier to cite. □

Theorem 7177.3 (Regular Cauchy completion object dense-embedding handoff)**.** *Every dense-embedding, universal-property, functoriality, or* $\mathsf{Real}^{\uparrow}$ *sealing consumer that cites an accepted* $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ *packet first reads the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff is the object-level completion carrier $R, W, D, E, A, H, C, P, N$, not a theorem route itself, not quotient completion equality, not a selected limit, not countable choice, and not ambient Real completeness.

Proof. Begin with an accepted carrier $X = (R, W, D, E, A, H, C, P, N)$. The completion-facing consumer first opens the finite StreamName windows W and dyadic tolerance ledger D attached to the regular rational source R.

The admission row E then names those displayed rows as the completion object. This is the object-level boundary that theorem chapters may consume before proving dense embedding, universal mapping, or functorial transport.

If a Real-facing seal is needed, it is read through A after the same regular source, window, tolerance, and admission rows. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming.

Since no carrier coordinate stores a quotient completion, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem, the consumer receives only the displayed completion object. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ read factors through a RegSeqRat source row, StreamName finite observation row, DyadicRatCore tolerance ledger, object-admission row, RealUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any theorem-facing completion route is consumed.

Independence witness: regseqrat, streamname, dyadicratcore, real: none is bijective with $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ because this packet alone binds regular rational source data, finite observation windows, dyadic tolerance ledgers, object admission, Real boundary sealing, transport, replay, provenance, and local naming as one completion object carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionObject}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite observation rows, DyadicRatCore tolerance rows, an object-admission row, a RealUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionObject}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7177.1, Definition 7177.2, and Definition 7177.3 seed the object-level regular Cauchy completion carrier for later dense-embedding, universal-property, functoriality, and RealUp sealing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionObject}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close dense embedding, universal properties, functoriality, quotient completion equality, selected representatives, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked regular source admission, finite observation, dyadic tolerance visibility, object-admission exactness, Real-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionObjectUp.

Chapter 7178

A Concrete Naming Certificate for $\mathsf{YoungTableau}^{\uparrow}$

Following Fulton, *Young Tableaux*, the $\mathsf{YoungTableau}^{\uparrow}$ packet records the finite row-and-column filling surface of a Young diagram as BEDC NameCert data. It consumes the partition-shape discipline of chapter 201, supplies a tableau row ledger usable by character-theoretic consumers such as chapter 245, and exposes a representation-ring-facing finite classifier route for chapter 246. The packet names only displayed shape, alphabet, row, column, transport, provenance, and local certificate rows; it does not import a host set of boxes, quotient equality of tableaux, a Schur polynomial calculation, or a Littlewood–Richardson rule.

Definition 7178.1 (Young tableau carrier)**.** *A* $\mathsf{YoungTableau}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (S, A, R, C, T, P, N)$$

with the following displayed rows:

1. *S is the consumed* $\mathsf{Partition}^{\uparrow}$ *shape row, including the finite Young-diagram box list and its row-length ledger.*
2. *A is the entry-alphabet row, recording the finite alphabet visible to the tableau and the admissible entry names used by downstream readers.*
3. *R is the row ledger, listing each displayed row, its ordered boxes, and the weak row comparison data carried by the packet.*
4. *C is the column ledger, listing each displayed column, its ordered boxes, and the strict column comparison data carried by the packet.*
5. *T records componentwise* hsame *transport for the shape, alphabet, row, column, replay, provenance, and local naming rows.*
6. *P records displayed* Cont *replay and* Pkg *provenance over* $\mathsf{YoungTableau}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungTableau}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, A, R, C, T, P, N. It has no coordinate for host box equality, quotient tableau equality, an ambient symmetric-function algebra, a Schur functor construction, or a plactic monoid quotient.

Theorem 7178.2 (Young tableau NameCert obligations)**.** *For every accepted* $\mathsf{YoungTableau}^{\uparrow}$ *carrier* $Y = (S, A, R, C, T, P, N)$ *of Definition 7178.1, the local* $\mathsf{NameCert}_{\mathsf{YoungTableau}^{\uparrow}}$ *surface consists exactly of carrier admission, partition-shape exposure through* S*, entry-alphabet exposure through* A*, row-ledger exposure through* R*, column-ledger exposure through* C*, componentwise* hsame *transport through* T*, displayed* Cont *replay and* Pkg *provenance through* P*, and local naming through* N*. These obligations do not include a host tableau set, quotient equality of fillings, a Schur-polynomial identity, a Littlewood–Richardson coefficient, or a representation-theoretic completeness theorem.*

Lean4 ✓ `BEDC.Derived.YoungTableauUp.YoungTableauCarrier namecert obligations`

Proof. Project the carrier of Definition 7178.1. The packet exposes only the seven displayed rows S, A, R, C, T, P, N.

The combinatorial surface is read from S, A, R, C. The shape row fixes the finite box list, the alphabet row fixes the admissible entry names, and the row and column ledgers display the only comparison data available to the classifier.

The structural surface is then T, P, N. Componentwise transport, replay, provenance, and local naming preserve the already displayed rows and add no ambient tableau semantics. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 7178.3 (Young tableau shape-row-column stability)**.** *Let* $Y = (S, A, R, C, T, P, N)$ *and* $Y' = (S', A', R', C', T', P', N')$ *be accepted* $\mathsf{YoungTableau}^{\uparrow}$ *carriers. If the shape rows* S *and* S' *are equal as displayed partition-shape rows, the alphabet rows* A *and* A' *expose the same entry names, and the row and column ledgers* R, C *and* R', C' *agree componentwise under the displayed* hsame *transport, then the tableau classifier gives the same read to* Y *and* Y'*. The stability claim is internal to the finite rows* S, A, R, C, T, P, N *and does not assert equality of host fillings.*

Proof. Begin with the shape rows. Equality of S and S' identifies the same finite Young-diagram box list and row-length ledger, so both packets expose the same positions before any tableau read begins.

Use the displayed alphabet, row, and column ledgers next. The alphabet agreement identifies the admissible entry names, while the componentwise agreement of R, C with R', C' preserves the weak row and strict column comparison data carried by the packets.

Finally apply the structural rows. The transports T, T' and the replay/provenance rows P, P' preserve only the displayed shape, alphabet, row, and column data, and N, N' locally name that same finite route. Thus the classifier has no remaining coordinate on which to distinguish the two accepted reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{YoungTableau}^{\uparrow}}$**.**

Constructive story: A $\mathsf{YoungTableau}^{\uparrow}$ packet is generated from a partition-shape row, a finite entry-alphabet row, displayed row and column ledgers, componentwise hsame transport, displayed Cont replay with Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{YoungTableau}^{\uparrow}}$, seed)

Scope closed: Definition 7178.1, Definition 7178.2, and Definition 7178.3 seed the finite tableau carrier, obligation surface, and classifier-stability route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{YoungTableau}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Schur polynomial identities, Littlewood–Richardson coefficients, plactic monoid quotients, Schur functor exactness, representation-theoretic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shape exposure, alphabet exposure, row-ledger exposure, column-ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.YoungTableauUp.

Chapter 7179

A Concrete Naming Certificate for $\mathsf{CauchyTailSpace}^{\uparrow}$

Following Bishop and Bridges on Cauchy sequences and Kelley on filters, the $\mathsf{CauchyTailSpace}^{\uparrow}$ packet records the finite tail-stability carrier that sits between the stream-name schedule of chapter 126, the regular rational-name surface of chapter 301, the dyadic tolerance core of chapter 291, and the real-seal boundary of chapter 38. It is the local object read by the Cauchy-name map surface of chapter 1401 when a consumer needs tail truncation, not selected limits or host equality of reals.

Definition 7179.1 (CauchyTailSpace carrier)**.** *A* $\mathsf{CauchyTailSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, K, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the base regular Cauchy name row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and its underlying* $\mathsf{StreamName}^{\uparrow}$ *window schedule;*
2. *K is the tail cutoff schedule, listing the finite windows after which the carrier permits a tail read;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed tail windows;*
4. *Q is the tail-equivalence classifier, comparing two accepted packets only by the regular-name windows and dyadic tolerances visible past the displayed cutoffs;*
5. *H records componentwise* hsame *transport for R, K, D, Q and for the local naming row;*
6. *C records the displayed* Cont *replay routes by which a consumer reopens the same regular-name, cutoff, tolerance, and classifier rows;*
7. *P is the* Pkg *provenance over the stream-name, regular-rational, dyadic-tolerance, real-seal boundary, transport, replay, and local naming rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailSpace}^{\uparrow}}$ *row.*

The classifier has no coordinate for quotient streams, selected limits, host real equality, completeness, countable choice, or a hidden real-valued representative.

Theorem 7179.2 (Tail truncation stability)**.** *For every accepted* $\mathsf{CauchyTailSpace}^{\uparrow}$ *carrier $T = (R, K, D, Q, H, C, P, N)$, any consumer that discards an initial finite prefix must still expose the base*

regular Cauchy name row R, the cutoff schedule K, and the dyadic tolerance ledger D before it may read the tail-equivalence classifier Q. The truncation therefore preserves the finite $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *tail surface, without choosing a limit or quotient-stream representative.*

Proof. Project the carrier of Definition 7179.1. The only rows that can witness a tail read are the base regular name R, the cutoff schedule K, and the tolerance ledger D.

After a finite prefix is ignored, the displayed continuation route C still reopens R, K, D before Q is visible. The prefix removal changes which finite windows are inspected, but it does not add a new carrier coordinate.

Transport through H preserves the same rows componentwise. Hence the accepted tail read remains a finite regular-Cauchy read and cannot export a selected limit, quotient stream, or host real value. □

Theorem 7179.3 (Tail classifier transport)**.** *If two accepted* $\mathsf{CauchyTailSpace}^{\uparrow}$ *carriers are related by the displayed* hsame *rows of their packets, then every tail-equivalence read through Q transports along the same R, K, D rows. A transported classifier cannot replace those rows by host real equality, ambient completeness, or an unrecorded filter equality.*

Proof. Read the transport row H. It stores componentwise transport for the base regular name, cutoff schedule, dyadic ledger, classifier, and local naming row.

Replay the classifier through C. The classifier row Q is available only after the transported R, K, D rows are displayed, so its comparison remains tied to the finite packet.

Since the carrier has no coordinate for host real equality, filter equality, quotient streams, completeness, or choice, the transported classifier cannot use any of those objects as a substitute for the displayed tail rows. □

Theorem 7179.4 (Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyTailSpace}^{\uparrow}$ *packet must factor its read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the tail cutoff schedule K, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the classifier row Q, displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The read does not create a selected limit, a host real equality proof, a completeness theorem, a quotient-stream function, or a choice principle.*

Proof. Start with the carrier inventory. The only route from a tail packet toward $\mathsf{Real}^{\uparrow}$ is the displayed regular-rational and dyadic-tolerance route through R, K, D, Q.

The replay row C forces a consumer to reopen that route before any real-seal boundary can be cited. The transport row H and provenance row P keep the same finite data attached to the local naming row N.

The refused objects are absent from the carrier. Therefore a $\mathsf{Real}^{\uparrow}$-facing read may consume the finite tail-stability packet, but it cannot manufacture selected limits, host real equality, completeness, quotient-stream functions, or choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailSpace}^{\uparrow}$ packet is generated from a base regular Cauchy name, a finite tail cutoff schedule, a dyadic tolerance ledger, a tail-equivalence classifier, displayed transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7179.1, Definition 7179.2, Definition 7179.3, and Definition 7179.4 bind the finite Cauchy-tail carrier and its nonescape surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, completeness, or choice.

Upgrade path: A later verification route can encode BEDC.Derived.CauchyTailSpaceUp and prove the carrier, classifier transport, truncation stability, and Real-seal nonescape rows.

Chapter 7180

CandidateNormalizationDischargeUp NameCert Construction

The $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite discharge data. It reads the normalization audit map of chapter 9383 beside the frontier carrier of chapter 892, the candidate-normalization witness of chapter 6577, the normal-endpoint budget of chapter 4292, and the SN candidate interface of chapter 3567. The packet binds the discharge row connecting candidate evidence to endpoint use; it is not a blanket normalization theorem for arbitrary closed typed terms.

Definition 7180.1 (Candidate-normalization discharge carrier)**.** *A* $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, K, E, S, O, H, C, P, N)$$

with the following displayed rows:

1. *A is the audit-map anchor row for chapter 9383;*
2. *K is the candidate-evidence row, reading candidate, closed-candidate, or candidate-set evidence through the existing MetaCIC normalization frontier;*
3. *E is the endpoint-seal row, recording the normal endpoint that may be consumed only after K is displayed;*
4. *S is the discharge row connecting the candidate evidence in K to the endpoint use in E;*
5. *O is the obstruction-retention row, keeping absent candidate evidence, subject-reduction obstruction, and evidence-free endpoint requests visible;*
6. *H records componentwise* hsame *transport for A, K, E, S, O and for the local naming row;*
7. *C records the displayed* Cont *replay routes for audit, candidate, endpoint, discharge, and obstruction reads;*
8. *P is the* Pkg *provenance over* $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$, $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$, $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateNormalizationDischarge}^{\uparrow}}$ *row.*

Theorem 7180.2 (Candidate discharge obligation surface)**.** *Every accepted* $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ *packet exposes exactly the candidate-evidence row* K*, endpoint-seal row* E*, discharge row* S*, obstruction-retention row* O*, displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows needed before a normalization-facing consumer may use a strong-normalization endpoint. No consumer can reach the endpoint seal through the packet while omitting the candidate evidence or the discharge row.*

Proof. Project the carrier of Definition 7180.1. The endpoint row E is stored beside K and S, not as an independent normalization theorem.

Replay the read through C. A normalization-facing consumer must display the audit anchor, candidate evidence, endpoint seal, discharge link, and obstruction row in the carrier order before using the endpoint.

Transport and provenance through H and P preserve those same rows. Thus the local NameCert surface is exhausted by the displayed finite packet. □

Theorem 7180.3 (Endpoint seal nonescape)**.** *An accepted* $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ *packet is refuted as a discharge certificate if a normalization-facing consumer obtains a strong-normalization endpoint without displaying candidate evidence, endpoint seal, obstruction retention,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The packet does not export evidence-free strong normalization, host evaluation soundness, unrestricted subject reduction, quotient syntax equality, or closed consistency.*

Proof. Use the obligation surface of Definition 7180.2. It places candidate evidence and the discharge row before endpoint use.

The obstruction row O records exactly the refused bypasses. If the consumer reaches a strong-normalization endpoint while erasing K, S, O, H, C, P, N, then the read no longer factors through the accepted carrier.

Because the carrier has no row for evidence-free strong normalization, host evaluation soundness, unrestricted subject reduction, quotient syntax equality, or closed consistency, those endpoints remain outside the packet. □

Theorem 7180.4 (Sibling separation)**.** *The* $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ *carrier is not bijective, row for row, with* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$*,* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$*, or* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$*. It binds the discharge row connecting candidate evidence to endpoint use, while the siblings respectively expose frontier, witness, endpoint budget, or candidate interface rows without the same discharge packet.*

Proof. Compare first with $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$. The frontier exposes candidate and endpoint boundary rows, but it does not by itself store the local discharge row S together with this packet's obstruction-retention and endpoint-use contract.

Compare next with $\mathsf{MetaCICCandidateNormalizationWitness}^{\uparrow}$ and $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$. The witness records evidence for a candidate-facing route, and the budget records endpoint consumption constraints; neither is the single carrier that binds both sides through S.

Finally compare with $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$. The interface describes candidate admission, while $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ records the finite discharge packet through which an endpoint read is allowed. A row-preserving bijection would have to preserve $A, K, E, S, O, H, C, P, N$ simultaneously, which no listed sibling carries. □

Falsifiable prediction: An accepted $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ packet is refuted if a normalization-facing consumer obtains a strong-normalization endpoint without displaying candidate evidence, endpoint seal, obstruction retention, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: metacicnormalizationfrontier, metacic-candidate-normalization-witness, metacic-normal-endpoint-budget, metacic-sn-candidate-interface: $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ is not bijective to these siblings because it binds the discharge row connecting candidate evidence to endpoint use, while the siblings respectively expose frontier, witness, endpoint budget, or candidate interface rows without the same discharge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateNormalizationDischarge}^{\uparrow}}$.

Constructive story: A $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$ packet is generated from audit-map anchoring, candidate evidence, endpoint seal, discharge, obstruction retention, displayed transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateNormalizationDischarge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7180.1, Definition 7180.2, Definition 7180.3, and Definition 7180.4 bind the finite candidate-mediated endpoint discharge packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateNormalizationDischarge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, unrestricted subject reduction, host evaluation soundness, quotient syntax equality, or closed consistency.

Upgrade path: A later Lean route should encode BEDC.Derived.CandidateNormalizationDischargeUp and provide Nontrivial and FieldFaithful instances in BEDC.Derived.CandidateNormalizationDischargeUp.TasteGate.

Chapter 7181

A Concrete Naming Certificate for $\mathsf{ExactCouple}^{\uparrow}$

Following Cartan–Eilenberg and McCleary, the $\mathsf{ExactCouple}^{\uparrow}$ packet records the finite precursor by which graded-object data, two degree-preserving maps, and one connecting map generate the page handoff later consumed by $\mathsf{SpectralSeq}^{\uparrow}$. It sits over the abelian-category surface of chapter 176, the homology and cohomology rows of chapter 98 and chapter 99, and before the spectral-sequence packet of chapter 178. The certificate names the finite exact-couple ledger only; it does not export an ambient derived-category calculus, quotient page object, or unrecorded convergence theorem.

Definition 7181.1 (Exact-couple carrier)**.** *An* $\mathsf{ExactCouple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, G, D, I, J, K, W, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row in which the displayed graded objects and maps are read.*
2. *G is the graded-object row, carrying the inspected bidegree window and its neighboring degree shifts.*
3. *D is the degree-preserving self-map row on the displayed graded object data.*
4. *I is the second degree-preserving map row, recording the visible inclusion or transition map used by the exact couple.*
5. *J is the connecting-map row from the companion graded object into the shifted inspected object.*
6. *K is the* $\mathsf{Homology}^{\uparrow}$ *or* $\mathsf{Cohomology}^{\uparrow}$ *kernel-image readback row attached to the same degree window.*
7. *W records the finite exactness windows for the rows D, I, J, K, with every kernel and image comparison read through the displayed certificate rows.*
8. *R is the derived-page handoff row, exposing the finite page data later consumed by* $\mathsf{SpectralSeq}^{\uparrow}$*.*
9. *H records componentwise* hsame *transport for the graded-object row, the two degree-preserving maps, the connecting map, exactness windows, derived-page handoff, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an exact-couple read through $A, G, D, I, J, K, W, R, H$.*

11. *P is the* Pkg *provenance over* $\mathsf{ExactCouple}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{SpectralSeq}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ExactCouple}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient page equality, host exactness outside the finite windows, derived-category localization, selected representatives, countable choice, or convergence data beyond the handoff row R.

Theorem 7181.2 (Exact-couple NameCert obligations)**.** *For every accepted* $\mathsf{ExactCouple}^{\uparrow}$ *carrier $Q = (A, G, D, I, J, K, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ExactCouple}^{\uparrow}}$ *surface is exhausted by the abelian-category source row, graded-object data, two degree-preserving maps, one connecting map, kernel-image readback, finite exactness windows, derived-page handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may not infer an ambient exact-couple object, quotient page equality, host exactness theorem, selected cycle representative, hidden derived-category localization, or spectral-sequence convergence claim.*

Proof. Project the carrier of Definition 7181.1. The rows A, G, D, I, J expose the abelian source, graded data, two degree-preserving maps, and the connecting map before any exactness or page read is available.

Next read K and W. These rows attach kernel-image comparisons to the same finite degree window and keep exactness inside the displayed $\mathsf{Homology}^{\uparrow}$ or $\mathsf{Cohomology}^{\uparrow}$ readback rather than in a host theorem.

Finally R, H, C, P, N package the derived-page handoff, transport, replay, provenance, and local naming rows. Since none of these rows adds quotient page equality, a selected representative, derived localization, countable choice, or convergence data, the listed rows exhaust the local NameCert obligations. □

Theorem 7181.3 (Exact-couple spectral-sequence handoff)**.** *Every* $\mathsf{SpectralSeq}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ExactCouple}^{\uparrow}$ *packet must reach the page-generation boundary through the finite route*

$$\mathsf{AbelianCat}^{\uparrow},\ G,\ D,\ I,\ J,\ K,\ W \longmapsto R \longmapsto \mathsf{SpectralSeq}^{\uparrow}.$$

The route exports the derived-page handoff row R together with H, C, P, N, not an ambient exact-couple object, not a quotient page primitive, and not convergence data outside the spectral-sequence packet that consumes the handoff.

Proof. Start with the carrier rows A, G, D, I, J, K, W. They display the exact-couple data and the finite exactness windows before any page-generation read can be replayed.

The handoff is then the row R. By definition R exposes only finite page data derived from the same graded-object and exactness-window ledger, and the structural rows H, C, P, N transport and replay that same handoff.

Therefore a $\mathsf{SpectralSeq}^{\uparrow}$ consumer receives a finite precursor route rather than a primitive page oracle. Any quotient page equality, ambient exact-couple object, hidden representative choice, or convergence assertion lies outside the accepted $\mathsf{ExactCouple}^{\uparrow}$ carrier. □

Falsifiable prediction: A spectral-sequence page-generation read is rejected as an $\mathsf{ExactCouple}^{\uparrow}$ handoff when it preserves R but omits the graded-object row G, degree-preserving map rows D, I, connecting map row J, kernel-image readback K, or exactness-window row W.

Independence witness: abeliancat, homology, cohomology, spectralseq: none is bijective with $\mathsf{ExactCouple}^{\uparrow}$ because the packet jointly carries graded-object data, two degree-preserving maps,

one connecting map, exactness windows, derived-page handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ExactCouple}^{\uparrow}}$.

Constructive story: An $\mathsf{ExactCouple}^{\uparrow}$ packet is generated from an abelian-category source row, displayed graded-object data, two degree-preserving maps, one connecting map, kernel-image readback, finite exactness windows, a derived-page handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExactCouple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7181.1, Definition 7181.2, and Definition 7181.3 seed the finite exact-couple precursor to the spectral-sequence page handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExactCouple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient exact-couple objects, quotient page equality, selected representatives, derived-category localization, countable choice, or convergence theorems.

Upgrade path: Obligation closure requires checked graded-object rows, two degree-preserving map rows, connecting-map rows, finite exactness-window compatibility, derived-page handoff compatibility, and structural transport for BEDC.Derived.ExactCoupleUp

Chapter 7182

A Concrete Naming Certificate for $\mathsf{SeminormFamily}^{\uparrow}$

Following Bourbaki and Treves on locally convex spaces and families of seminorms, the $\mathsf{SeminormFamily}^{\uparrow}$ packet isolates the finite seminorm-family ledger consumed by chapter 5868, chapter 3386, chapter 205, chapter 47, and chapter 38. It names the family before downstream packets read convex balanced neighbourhoods, induced uniform probes, seminormed-space comparisons, or completion-facing data.

Definition 7182.1 (SeminormFamily carrier)**.** *A* $\mathsf{SeminormFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, B, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *vector source, exposing vector zero, addition, negation, and scalar action only through the displayed vector-space packet.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar source, exposing nonnegative value rows, scalar absolute values, addition, multiplication, and order budgets used by the seminorm reads.*
3. *I is the finite family-index row listing exactly the seminorm labels admitted by the packet.*
4. *F is the seminorm-row ledger; for each displayed index in I, it records the value probe and the corresponding nonnegativity, zero, triangle, and scalar rows.*
5. *Z is the zero-compatibility row, comparing the vector-zero classifier of V with every zero-value row in F.*
6. *T is the triangle-budget row, routing each displayed read of $p_i(x+y)$ through vector addition in V, the selected seminorm row in F, and the Real addition and order rows in R.*
7. *A is the scalar-budget row, routing each displayed read of $p_i(ax)$ through scalar action in V, the absolute-value and multiplication rows in R, and the selected seminorm row in F.*
8. *B is the generated-neighbourhood probe row, recording the finite seminorm balls and finite intersections handed to topological-vector-space and locally convex consumers.*
9. *H records componentwise* hsame *transport for $V, R, I, F, Z, T, A, B, C, P, N$.*
10. *C records displayed* Cont *replay through vector source, Real source, family index, seminorm rows, zero, triangle, scalar, generated-neighbourhood probes, transport, provenance, and local naming.*

11. *P is the* Pkg *provenance over* $\mathsf{SeminormFamily}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an unbounded family, a hidden host topology, a selected quotient completion, a continuous dual, or a norm that replaces the family ledger.

Theorem 7182.2 (SeminormFamily NameCert obligations)**.** *For every accepted* $\mathsf{SeminormFamily}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed vector source, Real scalar source, finite family index, seminorm rows, zero compatibility, triangle budget, scalar budget, generated neighbourhood probes, transport, replay, provenance, and local naming; index exposure through I before any seminorm row in F is consumed; zero compatibility through Z; triangle and scalar budget routing through T and A; neighbourhood handoff through B; and transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Carrier admission is projection from Definition 7182.1. The accepted packet displays $V, R, I, F, Z, T, A, B, H, C, P, N$ and no further coordinate.

Index exposure is row exhaustion. A seminorm read first names an index from I, then reads the matching entry of F together with its zero, triangle, and scalar evidence.

The compatibility budgets are the displayed rows Z, T, A. The row Z compares zero-value data with the vector-zero classifier, T routes sums through vector addition and Real addition-order data, and A routes scalar multiplication through the Real absolute-value and multiplication rows.

Finally B is the only generated-neighbourhood handoff, while H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no hidden topology, unbounded index set, selected completion, continuous-dual selector, or replacement norm coordinate, these rows exhaust the NameCert surface. □

Theorem 7182.3 (SeminormFamily locally convex handoff)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer and every Frechet-facing consumer that reads convex balanced neighbourhoods or completion-facing data from a finite seminorm family must first pass through the* $\mathsf{SeminormFamily}^{\uparrow}$ *rows*

$$V, R, I, F, Z, T, A, B, H, C, P, N.$$

The handoff exports finite indexed seminorm probes and generated neighbourhood rows; it exports no unbounded seminorm family, no hidden host topology, no selected quotient completion, and no detached norm substitute.

Proof. Start with a locally convex or Frechet-facing request. The vector and scalar sources are V and R, while the admissible seminorm labels are exactly the finite row I.

Use the NameCert obligations of Definition 7182.2. A consumer reaches any seminorm value only through F after index exposure, and it reaches zero, triangle, and scalar budgets only through Z, T, A.

The neighbourhood route is then B. Convex balanced neighbourhoods and completion-facing probes consume the generated finite seminorm balls and finite intersections recorded there, with transport, replay, provenance, and local naming preserved by H, C, P, N.

Thus a downstream consumer receives only the displayed finite family route. With no coordinate for an unbounded family, host topology, selected quotient completion, or detached norm substitute, the named handoff is exactly the packet displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{SeminormFamily}^{\uparrow}$ packet is generated from displayed VecSpace and Real source rows, a finite family-index row, seminorm rows, zero, triangle, and scalar budgets, generated-neighbourhood probes, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeminormFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7182.1, Definition 7182.2, and Definition 7182.3 seed the finite seminorm-family ledger consumed by seminormed-space, locally convex, topological-vector, vector-space, Real, and completion-facing packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeminormFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unbounded seminorm families, host locally convex topology, quotient completion, selected continuous duals, or norm replacement.

Upgrade path: Scoped closure requires checked preservation of the displayed rows $V, R, I, F, Z, T, A, B, H, C, P, N$ through every locally convex and completion-facing consumer route.

Chapter 7183

A Concrete Naming Certificate for $\mathsf{BihariLaSalle}^{\uparrow}$

Following Bihari and LaSalle, the $\mathsf{BihariLaSalle}^{\uparrow}$ packet records the finite nonlinear integral-inequality surface used when a nonnegative comparison function is controlled by an Osgood-type modulus before a Real-facing stability consumer reads the bound. It sits after the linear Gronwall route of chapter 2034, the Bellman integral-comparison route of chapter 6270, the finite integration rows of chapter 1526, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only interval, unknown-envelope, window, Osgood-modulus, inverse-modulus, readback, Real-seal, transport, replay, provenance, and local naming rows.

Definition 7183.1 (Bihari–LaSalle carrier)**.** *A* $\mathsf{BihariLaSalle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, U, W, O, G, R, Z, H, C, P, N)$$

with the following visible rows:

1. *I is the interval row, recording the displayed time window, endpoint names, and the finite partition cells on which the nonlinear comparison is inspected.*
2. *U is the unknown-envelope row, storing the nonnegative comparison quantity and its initial bound as finite* $\mathsf{Real}^{\uparrow}$*-readable data.*
3. *W is the weight row, recording the displayed integral coefficient, cellwise majorant, and finite error budget used by the comparison.*
4. *O is the Osgood-modulus row, displaying the nondecreasing positive modulus and the finite inverse-modulus windows on which the Bihari separation read is made.*
5. *G is the Gronwall/Osgood handoff row, identifying the linear Gronwall route when O is linear and the nonlinear Osgood route when the inverse-modulus ledger is required.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the interval endpoints, envelope samples, coefficient bounds, Osgood-modulus values, inverse-modulus windows, and comparison budget.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row for the final Bihari–LaSalle bound after the finite rows have been replayed.*
8. *H records componentwise* hsame *transport for the interval, envelope, weight, Osgood-modulus, handoff, readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through*

$$I, U, W, O, G, R \longmapsto Z \longmapsto \mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}.$$

10. *P is the* Pkg *provenance over* $\mathsf{BihariLaSalle}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{GronwallBellman}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host nonlinear integral operator, selected inverse function, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or analytic ODE solver.*

Lean4 (statement) `BEDC.Derived.BihariLaSalleUp`

Theorem 7183.2 (Bihari–LaSalle NameCert obligations)**.** *For every accepted* $\mathsf{BihariLaSalle}^{\uparrow}$ *carrier* $B = (I, U, W, O, G, R, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval, unknown-envelope, weight, Osgood-modulus, Gronwall/Osgood handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. No obligation is discharged by a host integral theorem, selected inverse-modulus function, quotient real equality, countable-choice schedule, ambient Real completeness, or external ODE solver.*

Proof. Project the accepted carrier of Definition 7183.1. The rows I, U, W, O, G, R, Z are the only analytic-facing coordinates, and the structural rows H, C, P, N only transport, replay, source, and name those coordinates.

The interval, envelope, and weight rows determine the finite nonlinear comparison problem. The Osgood-modulus row supplies the displayed modulus and inverse-window ledger, while the handoff row identifies the linear Gronwall branch as a special finite route and the nonlinear branch as the inverse-modulus route.

The regular-rational readback and Real seal bind the comparison to displayed observations. Since every endpoint of the comparison passes through R before Z is consumed, the certificate cannot import a host integral theorem, selected inverse function, quotient equality, choice schedule, completeness theorem, or external solver. □

Theorem 7183.3 (Bihari–LaSalle Gronwall handoff)**.** *Every accepted* $\mathsf{BihariLaSalle}^{\uparrow}$ *carrier routes a nonlinear integral-inequality read through the finite handoff*

$$\mathsf{GronwallBellman}^{\uparrow} \longmapsto \mathsf{BihariLaSalle}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

When the Osgood-modulus row O is linear, the handoff row G exposes the Gronwall comparison of chapter 2034. When O is nonlinear, G keeps the read on the displayed inverse-modulus windows before the regular-rational readback R and Real seal Z are available.

Proof. Begin with an accepted carrier B. The comparison can inspect the interval row I, unknown envelope U, and weight row W before it reaches any bound.

Read the Osgood-modulus row next. If the displayed modulus is the linear one, the row G cites the Gronwall and Bellman comparison routes already present in the sibling chapters. Otherwise the same row keeps the comparison inside the inverse-modulus windows listed by O.

Finally expose R and Z. The regular-rational row records the finite observations from both branches, and the Real seal is terminal for those observations. Thus the handoff cannot bypass the displayed rows through host integration, selected inverse functions, quotient equality, countable choice, or ambient completeness. □

Theorem 7183.4 (Bihari–LaSalle Real seal nonescape)**.** *For every accepted* $\mathsf{BihariLaSalle}^{\uparrow}$ *carrier* $B = (I, U, W, O, G, R, Z, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* Z *exposes only the bound obtained from the interval, unknown-envelope, weight, Osgood-modulus, Gronwall/Osgood handoff, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows. It exports no host nonlinear integral value, selected inverse-modulus function, quotient stream equality, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness result, or analytic ODE continuation theorem.*

Proof. Use Definition 7183.2 to identify the local obligation surface. The seal row is listed there only as the terminal read of the finite carrier rows.

Use Definition 7183.3 to locate the route into Z. The handoff opens I, U, W, O, G, R before the Real seal is available, so the seal cannot be an independent host integral value or selected inverse-modulus function.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Therefore a consumer of Z receives only the finite Bihari–LaSalle comparison packet and none of the excluded host analytic objects. □

Falsifiable prediction: Every accepted $\mathsf{BihariLaSalle}^{\uparrow}$ consumer that reads a Real nonlinear comparison bound without the displayed interval, unknown-envelope, weight, Osgood-modulus, Gronwall/Osgood handoff, RegSeqRat readback, transport, replay, provenance, and local NameCert rows refutes the packet surface.

Independence witness: gronwall-inequality, gronwall-bellman, riemannintegral, regseqrat, real: none is bijective to $\mathsf{BihariLaSalle}^{\uparrow}$ because this packet binds finite nonlinear modulus data, inverse-window readback, Gronwall/Osgood handoff, Real seal, transport, replay, provenance, and local naming into one integral-inequality certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BihariLaSalle}^{\uparrow}$ packet is generated from one interval row, one unknown-envelope row, one weight row, one Osgood-modulus row, one Gronwall/Osgood handoff row, one RegSeqRat readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7183.1, Definition 7183.2, Definition 7183.3, and Definition 7183.4 seed the finite nonlinear integral-inequality packet over interval, envelope, coefficient, Osgood-modulus, inverse-window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BihariLaSalle}^{\uparrow}}, \mathsf{encodedDef})$

Bridge status: none

Not claimed: This chapter does not claim host nonlinear integral operators, selected inverse-modulus functions, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, analytic ODE continuation, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Osgood-modulus stability, Gronwall/Osgood handoff, inverse-window accounting, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BihariLaSalleUp.

Chapter 7184

A Concrete Naming Certificate for $\mathsf{RegularRealTailSelector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularRealTailSelector}^{\uparrow}$ packet records the finite tail-selection surface used when a regular rational name is read through a terminal real-name seal. It sits over the dyadic tolerance core of chapter 291, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the equality-boundary discipline of chapter 1820. The packet selects a displayed finite tail window for one regular real-name read; it is not countable choice, quotient real equality, a representative selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7184.1 (Regular real tail selector carrier)**.** *A* $\mathsf{RegularRealTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, W, D, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the tail-threshold row naming the first index admitted for the regular real-name read.*
2. *W is the finite window row inspected beyond the displayed threshold.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the tail request.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected regular rational window.*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, W, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the threshold, window, tolerance, readback, and seal rows.*
7. *C records displayed* Cont *replay of the same tail-selection route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularRealTailSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, W, D, Q, A, H, C, P, N$. It has no coordinate for countable choice, quotient equality of real names, a selected representative, hidden completeness, or an external real-number selector.

Theorem 7184.2 (RegularRealTailSelector tail-window stability)**.** *Let $S = (T, W, D, Q, A, H, C, P, N)$ be an accepted* $\mathsf{RegularRealTailSelector}^{\uparrow}$ *carrier. Any read of the selected tail window is stable under the displayed* hsame *transports H and the displayed* Cont *replay C: the threshold row T, finite window W, dyadic tolerance D, regular rational readback Q, and real seal A remain in the same local packet. A transported read cannot replace these rows by countable choice, quotient real equality, a selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7184.1. The tail read is exhausted by $T, W, D, Q, A, H, C, P, N$, with T naming the threshold and W naming the finite window actually inspected.

Use the componentwise transport row H. It moves the threshold, window, tolerance, regular rational readback, and real seal together, so a transported read cannot keep the seal while changing the displayed tail data.

Replay is displayed through C and sourced by P. These rows keep the route inside the local $\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$ packet. Since the carrier has no coordinate for countable choice, quotient real equality, selected representatives, or ambient completeness, the transported tail-window read remains stable only as finite packet data. □

Theorem 7184.3 (RegularRealTailSelector NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealTailSelector}^{\uparrow}$ *carrier $S = (T, W, D, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$ *surface consists of carrier admission, tail-window stability, regular readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, Real-seal non-escape through* $\mathsf{Real}^{\uparrow}$*, componentwise transport, displayed replay, provenance, and local naming. The same surface refuses countable choice, quotient real equality, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as available outputs.*

Proof. Carrier admission is the inventory of Definition 7184.1. The rows $T, W, D, Q, A, H, C, P, N$ are the complete local packet.

Tail-window stability is supplied by Definition 7184.2. The regular readback row Q is consumed only with the dyadic tolerance row D and the inspected window W, and the Real seal row A appears only after those rows have been displayed.

The non-escape clauses are inspection of the same carrier. No row is a countable-choice schedule, quotient equality over real names, representative selector, or ambient completeness theorem; therefore no accepted local read can export one. □

Falsifiable prediction: An accepted $\mathsf{RegularRealTailSelector}^{\uparrow}$ consumer is refuted if it reaches a Real seal before displaying the tail threshold, finite window, dyadic tolerance, RegSeqRat readback, hsame transport, Cont replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-equivalence-setoid: $\mathsf{RegularRealTailSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a tail threshold, inspected finite window, dyadic tolerance, regular rational readback, Real seal, transport, replay, provenance, and local naming in one accepted tail-selection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularRealTailSelector}^{\uparrow}$ packet is generated from tail-threshold rows, finite window rows, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$, seed)

Scope closed: Definition 7184.1, Definition 7184.2, and Definition 7184.3 seed the finite regular real tail-selection surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularRealTailSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient real equality, selected representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window stability, regular readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularRealTailSelectorUp.

Chapter 7185

A Concrete Naming Certificate for RegularCauchyDoubleLimitInterchange$^{\uparrow}$

Following Moore and Osgood, the RegularCauchyDoubleLimitInterchange$^{\uparrow}$ packet records the finite constructive data needed to compare two iterated regular Cauchy reads of a two-parameter rational grid. It sits over the dyadic tolerance core of chapter 291, the rectangular observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the scheduled two-axis route of chapter 460. The packet names a finite-window double-limit interchange surface; it does not assert quotient equality of arrays, select either iterated limit, use countable choice, or assume ambient completeness of Real$^{\uparrow}$.

Definition 7185.1 (Regular Cauchy double-limit interchange carrier)**.** *A* RegularCauchyDoubleLimitInterchange$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the two-parameter* RegSeqRat$^{\uparrow}$ *grid row, recording the row and column rational reads admitted by the packet.*
2. *W is the rectangular* StreamName$^{\uparrow}$ *window row, naming the finite row, column, and diagonal observation rectangles opened by an interchange request.*
3. *R is the row-column regularity ledger, recording the two iterated* RegSeqRat$^{\uparrow}$ *readback routes and their shared diagonal handoff.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *rectangle-budget row, carrying dyadic tolerances, slack cells, and comparison entries for every admitted rectangle.*
5. *E is the* Real$^{\uparrow}$ *seal row visible to downstream real consumers only after G, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for G, W, R, D, E and for replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route from rectangular windows and dyadic budgets to the two iterated readbacks and the shared seal.*
8. *P is the* Pkg *provenance over* RegularCauchyDoubleLimitInterchange$^{\uparrow}$*,* CauchyDoubleSequence$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $G, W, R, D, E, H, C, P, N$. It has no coordinate for a quotient double sequence, a selected row limit, a selected column limit, a global choice schedule, or an ambient real-completeness principle.

Theorem 7185.2 (Regular Cauchy double-limit interchange NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}$ *carrier*

$$I = (G, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed grid, window, readback, budget, seal, transport, replay, provenance, and naming rows;*
2. *classifier stability under componentwise transport of $G, W, R, D, E, H, C, P, N$;*
3. *rectangle stability saying that each row, column, and diagonal observation used by R is named by W and bounded by the dyadic ledger D;*
4. *diagonal handoff saying that the common diagonal read factors through the same scheduled window route as the underlying* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *packet;*
5. *nonescape for quotient arrays, selected iterated limits, countable choice, hidden real completeness, and bridge data.*

Proof. Project the carrier of Definition 7185.1. The rows $G, W, R, D, E, H, C, P, N$ are the full accepted packet, so carrier admission and classifier stability are componentwise on those rows.

Rectangle stability is read from the pair W, D. The window row names every finite row, column, and diagonal rectangle inspected by the two iterated readbacks, while the dyadic budget row supplies the visible tolerance and slack comparisons for those same rectangles.

The diagonal handoff uses the scheduled two-axis route of chapter 460. The packet can reuse the diagonal route only after the rectangular StreamName windows, RegSeqRat grid rows, and DyadicRatCore budget entries have been displayed.

The rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Since the carrier contains no quotient array, selected limit, choice schedule, hidden completeness field, or bridge row, the nonescape clauses follow from absence of those coordinates. □

Theorem 7185.3 (Regular Cauchy double-limit interchange consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}$ *packet first reads the finite route*

$$\mathsf{CauchyDoubleSequence}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The consumer receives a rectangular window comparison surface and a diagonal seal handoff, not a theorem of host double-limit convergence and not a selected value for either iterated limit.

Proof. Let $I = (G, W, R, D, E, H, C, P, N)$ be an accepted carrier. Any consumer first opens the rectangular StreamName windows W and dyadic budget D attached to the regular rational grid G.

The iterated readbacks and common diagonal handoff are then read from R. The seal row E becomes visible only after those finite rows have been displayed, so the Real-facing route is mediated by the same finite packet.

Transport and replay do not add coordinates: H is componentwise and C replays only the displayed rectangle-to-seal route. Thus the consumer boundary is exactly the listed dependency route and cannot export a host convergence theorem, a quotient array, or a selected iterated limit. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}$ read factors through a RegSeqRat grid row, rectangular StreamName window row, DyadicRatCore rectangle-budget row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any Real-facing comparison is visible.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-double-sequence: none is bijective with $\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}$ because this packet alone binds rectangular finite windows, two iterated regular readbacks, a shared diagonal handoff, dyadic rectangle budgets, a Real seal, transport, replay, provenance, and local naming as one interchange carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}$ packet is generated from a two-parameter RegSeqRat grid, rectangular StreamName windows, dyadic rectangle budgets, two iterated readback routes, a shared diagonal handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7185.1, Definition 7185.2, and Definition 7185.3 seed the finite-window double-limit interchange packet over rectangular observations, regular rational grid rows, dyadic budgets, diagonal handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDoubleLimitInterchange}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of arrays, selected row or column limits, countable choice, host double-limit convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, rectangle stability, diagonal handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyDoubleLimitInterchangeUp.

Chapter 7186

A Concrete Naming Certificate for $\mathsf{FiniteKernelClosureGate}^{\uparrow}$

The $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ packet concretizes the closure-admission discipline of chapter 9459. It binds a candidate kernel surface to the finite rows that decide whether an AI-origin chapter may report closure: a TasteGate admission row, a closure-status reference row, an axiom-purity audit row, transport, replay, provenance, and local naming. The packet is a gate over displayed evidence, not a semantic approval oracle, not a proof of the candidate theorem, and not a bridge checker.

The finite gate consumes the admission row of chapter 545, the closure-ledger row of chapter 1265, and the dependency-purity row of chapter 1259 before a closure-admission decision is visible. Those siblings supply the displayed evidence read by A, S, Q; this packet only binds them into the finite kernel gate surface.

Definition 7186.1 (Finite kernel closure gate carrier)**.** *A* $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, A, S, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the candidate kernel-surface row, recording the chapter label, origin row, finite carrier surface, and requested closure grade.*
2. *A is the consumed* $\mathsf{TasteGateAdmission}^{\uparrow}$ *row, exposing the candidate's ChapterTasteGate admission evidence and refused-case ledger.*
3. *S is the consumed* $\mathsf{ClosureStatusReference}^{\uparrow}$ *row, recording the current theory-closure grade, formal-status grade, bridge status, not-claimed surface, and next-grade route.*
4. *Q is the consumed* $\mathsf{AxiomPurityGate}^{\uparrow}$ *row, recording the finite dependency audit and forbidden-dependency refusal surface for the candidate target.*
5. *F is the field ledger for the gate itself, listing which visible fields of X, A, S, Q are read by closure admission and which fields remain outside the gate.*
6. *H records componentwise* hsame *transport for X, A, S, Q, F and for replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the route from candidate surface through admission evidence, closure-status reference, axiom-purity audit, and field ledger.*

8. *P is the* Pkg *provenance over* $\mathsf{FiniteKernelClosureGate}^{\uparrow}$, $\mathsf{TasteGateAdmission}^{\uparrow}$, $\mathsf{ClosureStatusReference}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteKernelClosureGate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, A, S, Q, F, H, C, P, N$. It has no coordinate for hidden prompt state, host-side approval, theorem truth, bridge checking, global horizon admission, or a closure grade stronger than the displayed status row.

Theorem 7186.2 (Finite kernel closure gate NameCert obligations)**.** *For every accepted* $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ *carrier*

$$K = (X, A, S, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteKernelClosureGate}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; classifier stability under componentwise transport of $X, A, S, Q, F, H, C, P, N$; TasteGate-admission exposure through A; closure-status reference exposure through S; axiom-purity exposure through Q; field-ledger exactness through F; displayed replay through C; provenance through P; local naming through N; and nonescape for hidden prompt state, host-side approval, theorem truth, bridge checking, and unreported closure strength.*

Proof. Project the carrier of Definition 7186.1. The finite packet exposes exactly the candidate surface, admission row, closure-status row, axiom-purity row, field ledger, transport row, replay row, provenance row, and local naming row.

The admission, status, and audit obligations are read from A, S, Q. The row A supplies the TasteGate admission surface, S supplies the two-axis closure-status reference, and Q supplies the forbidden-dependency audit surface.

The field ledger F records which visible fields the gate reads. Transport through H and replay through C preserve those fields componentwise, while P and N record source and local naming.

Since the carrier has no hidden prompt, host approval, theorem-truth, bridge-checking, global-admission, or stronger-status coordinate, no consumer can extract one from the accepted gate packet. □

Theorem 7186.3 (Finite kernel closure gate TasteGate expectation)**.** *A future Lean realization of* $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ *must expose a nontrivial carrier and a field-faithful carrier: the nontrivial witness separates an accepted gate row from a refused gate row, and the field-faithful witness proves that equality of the displayed rows $X, A, S, Q, F, H, C, P, N$ determines equality of the carrier token. These are TasteGate expectations for the finite gate surface, not paper claims that the Lean declarations already exist.*

Proof. The carrier in Definition 7186.1 has visible admission and refusal branches because A and S record accepted and refused gate reads and Q records forbidden-dependency refusals. A nontrivial Lean instance should therefore choose one accepted row and one refused row from that displayed surface.

Field faithfulness is also determined by the displayed carrier. The rows $X, A, S, Q, F, H, C, P, N$ are the entire classifier surface, so a Lean proof may compare carriers by projecting and comparing exactly those fields.

The theorem records only this expected shape. Until those Lean declarations are present, the paper chapter remains at seed closure and does not add a Lean marker. □

Falsifiable prediction: Every accepted $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ report factors through a TasteGateAdmission row, ClosureStatusReference row, AxiomPurityGate row, finite field ledger, com-

ponentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any closure-admission decision is visible.

Independence witness: tastegateadmission, closurestatusreference, axiompuritygate, namecert: none is bijective with $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ because this packet alone binds candidate closure request, TasteGate admission, closure-status reference, axiom-purity audit, gate-local field ledger, transport, replay, provenance, and local naming as one finite closure-admission carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteKernelClosureGate}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{FiniteKernelClosureGate}^{\uparrow}$ packet is generated from a candidate kernel-surface row, a TasteGateAdmission row, a ClosureStatusReference row, an AxiomPurityGate row, a gate-local field ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteKernelClosureGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7186.1, Definition 7186.2, and Definition 7186.3 seed the finite closure-admission gate over TasteGate admission, closure-status reference, axiom-purity audit, field ledger, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteKernelClosureGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove the candidate theorem, certify host semantics, close bridge checking, assert a Lean declaration, or authorize closure stronger than the displayed status and audit rows.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, TasteGate-admission exposure, closure-status exposure, axiom-purity exposure, field-ledger exactness, nonescape, and TasteGate nontrivial and field-faithful instances for BEDC.Derived.FiniteKernelClosureGateUp.

Chapter 7187

A Concrete Naming Certificate for $\mathsf{AbelDirichletTest}^{\uparrow}$

Following Abel, Dirichlet, and Hardy-style summation tests, the $\mathsf{AbelDirichletTest}^{\uparrow}$ packet records the combined convergence-test surface in which an Abel summation row, an Abel-test face, and a Dirichlet-test face share one completion-facing tail route. It sits beside the summation-by-parts certificate of chapter 1763, the Abel-test certificate of chapter 1496, the Dirichlet-test certificate of chapter 1497, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the comparison packet of chapter 2230. The packet is a finite naming certificate for the shared Abel–Dirichlet convergence route; it is not a selected infinite sum, a quotient equality of series streams, a countable-choice selector, an order-trichotomy principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7187.1 (Abel–Dirichlet test carrier)**.** *An* $\mathsf{AbelDirichletTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, A, B, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source series row, exposing only the finite term and partial-sum windows read by the test.*
2. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, carrying the finite summation-by-parts window and boundary-term data.*
3. *B is the* $\mathsf{AbelTest}^{\uparrow}$ *face row, exposing the bounded partial-sum, monotone null coefficient, and Abel tail ledger visible to the combined packet.*
4. *D is the* $\mathsf{DirichletTest}^{\uparrow}$ *face row, exposing the bounded partial-sum, coefficient-window, null-tail, and Dirichlet tail-schedule rows visible to the combined packet.*
5. *Q is the* $\mathsf{AbelDirichletComparison}^{\uparrow}$ *row, forcing the two test faces to meet at one displayed comparison and shared-tail request before readback.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed source, summation, test-face, and comparison windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after S, A, B, D, Q, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, summation, Abel face, Dirichlet face, comparison, readback, endpoint, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the finite route*

$$S \longmapsto A \longmapsto B, D \longmapsto Q \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{AbelDirichletTest}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{AbelDirichletComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelDirichletTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven displayed rows. It has no coordinate for a selected infinite sum, quotient series equality, countable choice, trichotomy, or ambient real completeness.

Theorem 7187.2 (Abel–Dirichlet test NameCert obligations)**.** *For every accepted* $\mathsf{AbelDirichletTest}^{\uparrow}$ *carrier $T = (S, A, B, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelDirichletTest}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *source-window exposure through S;*
3. *Abel summation exposure through A, preserving the finite summation-by-parts and boundary-term rows;*
4. *Abel-test face exposure through B, preserving the bounded partial-sum, monotone null coefficient, and tail ledger rows;*
5. *Dirichlet-test face exposure through D, preserving the bounded partial-sum, coefficient-window, null-tail, and tail-schedule rows;*
6. *comparison exactness through Q, requiring the two test faces to meet at the displayed* $\mathsf{AbelDirichletComparison}^{\uparrow}$ *row before completion-facing readback;*
7. $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *endpoint exposure through R and E;*
8. *componentwise transport, displayed replay, provenance, and local naming through exactly H, C, P, N.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.AbelDirichletTestUp.AbelDirichletTestTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 7187.1. The source, summation, test-face, comparison, readback, endpoint, transport, replay, provenance, and local naming rows are exactly the coordinates of T.

Read the analytic rows in order. The Abel summation row A supplies the finite summation-by-parts window. The Abel face B and Dirichlet face D then contribute their displayed bounded-partial-sum and tail-control data.

The common convergence-test obligation is the row Q. It forces the two faces to pass through the existing Abel–Dirichlet comparison surface before R and E can be read by a completion-facing consumer.

Finally read H, C, P, N. These structural rows transport, replay, source, and name the same finite route, so the listed obligations exhaust the local certificate surface. □

Theorem 7187.3 (Abel–Dirichlet test tail handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{AbelDirichletTest}^{\uparrow}$ *carrier must factor through*

$$S \longmapsto A \longmapsto B, D \longmapsto Q \longmapsto R \longmapsto E$$

together with H, C, P, N. Thus the combined Abel–Dirichlet test exports one finite tail handoff to Real-and-Completion consumers, not two independent informal convergence claims.

Proof. Start with a completion-facing read. The source row S is the only source entry point, so the consumer first sees only the displayed finite term and partial-sum windows.

The summation row A supplies the shared Abel summation-by-parts data. The Abel-test face B and Dirichlet-test face D may then be read, but neither face can export a real endpoint directly from this carrier.

The comparison row Q is the unique handoff from the two test faces to later regular readback. Therefore R and E are public only after the displayed Abel–Dirichlet comparison route has been read.

Transport, replay, provenance, and local naming through H, C, P, N preserve that same route. Since the carrier has no excluded host-side coordinate, no consumer can bypass the finite tail handoff. □

Falsifiable prediction: An accepted $\mathsf{AbelDirichletTest}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, or downstream summability handoff without first reading the source row, $\mathsf{AbelSummation}^{\uparrow}$ row, $\mathsf{AbelTest}^{\uparrow}$ face, $\mathsf{DirichletTest}^{\uparrow}$ face, and $\mathsf{AbelDirichletComparison}^{\uparrow}$ row refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, abel-dirichlet-comparison, regseqrat, real: no listed sibling supplies the whole combined packet $T = (S, A, B, D, Q, R, E, H, C, P, N)$, because each sibling contributes only a summation row, a test face, a comparison handoff, a readback, or a seal while this chapter binds them as one finite convergence-test certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelDirichletTest}^{\uparrow}}$.

Constructive story: An $\mathsf{AbelDirichletTest}^{\uparrow}$ packet is generated from source windows, an $\mathsf{AbelSummation}^{\uparrow}$ summation-by-parts row, $\mathsf{AbelTest}^{\uparrow}$ and $\mathsf{DirichletTest}^{\uparrow}$ face rows, an $\mathsf{AbelDirichletComparison}^{\uparrow}$ shared-tail row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelDirichletTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7187.1, Definition 7187.2, and Definition 7187.3 seed the finite combined convergence-test surface over $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{AbelDirichletComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelDirichletTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Abel summation exposure, Abel and Dirichlet face exposure, comparison exactness, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbelDirichletTestUp.

Chapter 7188

A Concrete Naming Certificate for $\mathsf{ResidualSet}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{ResidualSet}^{\uparrow}$ packet records the positive side of a constructive Baire argument: a residual evidence carrier is accepted only when the dense-open schedule and the meagre-complement ledger are both displayed as finite BEDC rows. It sits beside the meagre-set carrier of chapter 2267, the nowhere-dense component carrier of chapter 4186, the dense-open schedule of chapter 6670, and the Baire-category theorem packet of chapter 3570. The packet names residual evidence; it does not choose a generic point, assert arbitrary countable intersections, or import a host category theorem.

Definition 7188.1 (ResidualSet carrier)**.** *A* $\mathsf{ResidualSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, D, M, N, B, H, C, P, L)$$

with the following displayed rows:

1. *T is the ambient topology or complete-metric context in which the residual read is made.*
2. *D is the* $\mathsf{BaireDenseOpen}^{\uparrow}$ *dense-open schedule row whose finite requests are visible to the Baire argument.*
3. *M is the* $\mathsf{MeagreSet}^{\uparrow}$ *complement ledger, decomposed through the* $\mathsf{NowhereDense}^{\uparrow}$ *component rows named by the meagre packet.*
4. *N records the nowhere-dense component interface read from chapter 4186.*
5. *B is the* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *handoff row certifying that the displayed dense-open and meagre-complement evidence are the rows consumed by the Baire route.*
6. *H stores componentwise* hsame *transport for T, D, M, N, B, C, P, L.*
7. *C stores displayed* Cont *replay of the dense-open schedule, meagre-complement ledger, Baire handoff, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ResidualSet}^{\uparrow}$, $\mathsf{BaireDenseOpen}^{\uparrow}$, $\mathsf{MeagreSet}^{\uparrow}$, $\mathsf{NowhereDense}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{ResidualSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen generic point, an unrestricted set-theoretic complement, a host countable-intersection principle, or a classical comeagre-subset theorem outside the displayed BEDC rows.

Theorem 7188.2 (ResidualSet Baire handoff)**.** *Every accepted* $\mathsf{ResidualSet}^{\uparrow}$ *packet read by a Baire-category consumer factors through the displayed dense-open schedule* D*, meagre-complement ledger* M*, nowhere-dense component interface* N*, and Baire handoff row* B*. Thus a residual conclusion is handed to downstream constructive Baire arguments only as the positive face of the same finite evidence that records the meagre complement.*

Proof. Project the carrier of Definition 7188.1. The only analytic rows are the dense-open schedule, meagre-complement ledger, nowhere-dense component interface, and Baire handoff.

Read the sibling packets in order. chapter 6670 supplies the finite dense-open requests, chapter 2267 supplies the complement ledger, and chapter 4186 supplies the component interface used by that ledger.

The Baire-category row B can therefore export residual evidence only through those displayed rows. Since R has no coordinate for a selected generic point or host countable intersection, the handoff is the finite BEDC residual carrier and nothing stronger. □

Theorem 7188.3 (ResidualSet NameCert obligations)**.** *For every accepted* $\mathsf{ResidualSet}^{\uparrow}$ *carrier* $R = (T, D, M, N, B, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{ResidualSet}^{\uparrow}}$ *surface consists exactly of ambient context admission through* T*, dense-open schedule exposure through* D*, meagre-complement exposure through* M*, nowhere-dense component visibility through* N*, Baire-category handoff through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* L*.*

Proof. Inspect the displayed tuple. Each accepted row of the residual packet is one of $T, D, M, N, B, H, C, P, L$, and the structural rows only transport, replay, source, and name those coordinates.

The negative side is exhausted by M and N: the meagre ledger supplies the complement surface, while the nowhere-dense component rows record the finite decomposition interface consumed by that ledger. The positive side is exhausted by D and B: the dense-open schedule supplies the Baire requests and the Baire-category handoff records the route by which the residual read reaches a consumer.

No listed coordinate stores a generic point, an arbitrary complement, or a host countable-intersection theorem. The local NameCert obligations are therefore precisely the displayed rows and their structural support. □

Theorem 7188.4 (ResidualSet dense G-delta boundary)**.** *Every accepted* $\mathsf{ResidualSet}^{\uparrow}$ *packet whose dense-open schedule and meagre-complement ledger are displayed through a* $\mathsf{DenseGDelta}^{\uparrow}$ *carrier factors through chapter 7204, chapter 3547, chapter 1542, and chapter 3570 before a residual consumer reads the positive side.*

Proof. Project the carrier of Definition 7188.1 as $R = (T, D, M, N, B, H, C, P, L)$. The residual packet exposes its dense-open schedule through D, its meagre-complement ledger through M, its nowhere-dense interface through N, and its Baire handoff through B.

When the positive side is displayed through a $\mathsf{DenseGDelta}^{\uparrow}$ carrier, read that carrier's D, M, N, B-facing route as the corresponding countable open schedule, dense-open schedule, Baire route, and residual handoff. The route first passes through the dense G_δ certificate, then through the $\mathsf{GDeltaSet}^{\uparrow}$ schedule, then through the Baire-category surface, and then through the Baire-category theorem packet.

The consumer therefore receives residual evidence only after the positive dense-open face has been displayed. No step chooses a generic point, forms a host intersection, imports countable choice, or identifies quotient sets outside the displayed BEDC rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ResidualSet}^{\uparrow}}$.

Constructive story: A $\mathsf{ResidualSet}^{\uparrow}$ packet is generated from ambient topology or complete-metric context rows, a $\mathsf{BaireDenseOpen}^{\uparrow}$ schedule, a $\mathsf{MeagreSet}^{\uparrow}$ complement ledger, $\mathsf{NowhereDense}^{\uparrow}$ component rows, a $\mathsf{BaireCategoryTheorem}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResidualSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7188.1, Definition 7188.2, and Definition 7188.3 bind the residual evidence carrier to the displayed Baire dense-open, meagre-complement, nowhere-dense, and Baire-category rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResidualSet}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close generic-point choice, arbitrary countable intersections, unrestricted set-theoretic complements, host category theorems, or a Lean carrier for $\mathsf{ResidualSet}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, meagre-complement exactness, nowhere-dense component visibility, Baire handoff, transport, replay, provenance, and local naming for the displayed residual evidence packet.

Chapter 7189

A Concrete Naming Certificate for CompactUniformLocalToGlobalModulus$^{\uparrow}$

Following Bishop–Bridges compact metric analysis, the CompactUniformLocalToGlobalModulus$^{\uparrow}$ packet records the finite local-to-global route that turns compact finite-net coverage and local pointwise modulus rows into a tolerance-only UniformModulus$^{\uparrow}$ handoff. It realizes the compact finite-net lower-bound route of chapter 9237 through the compact carrier of chapter 60, the compact-metric finite-net surface of chapter 129, the continuous-map rows of chapter 130, the rational lower-bound fold of chapter 619, the metric-triangle estimate of chapter 1159, and the uniform-modulus consumer row of chapter 369. The packet is finite NameCert data; it is not an open-cover compactness principle, a selected global minimum outside the fold, or a host modulus function.

Definition 7189.1 (CompactUniformLocalToGlobalModulus carrier)**.** *A* CompactUniformLocalToGlobalModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, L, R, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, including the compact source and the finite* ProbeBundle *coverage surface.*
2. *F is the* ContinuousMap$^{\uparrow}$ *local pointwise-modulus row read at the bundled centers.*
3. *L is the local-radius ledger that ties each bundled center to its pointwise modulus request.*
4. *R is the* Rat$^{\uparrow}$ *lower-bound fold row over the finite radius ledger.*
5. *T is the metric-triangle row that transports local estimates from centers to arbitrary source points.*
6. *U is the resulting* UniformModulus$^{\uparrow}$ *consumer handoff.*
7. *H stores componentwise* hsame *transport for $K, F, L, R, T, U, C, P, N$.*
8. *C stores displayed* Cont *replay of compact-net extraction, local-modulus reading, lower-bound folding, metric-triangle transport, uniform-modulus export, provenance, and local naming.*
9. *P is the* Pkg *provenance over* CompactUniformLocalToGlobalModulus$^{\uparrow}$, Compact$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Rat$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, Pkg, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, countable choice, a selected minimum outside the finite fold, quotient function equality, or an ambient Heine–Cantor theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformLocalToGlobalModulusUp.CompactUniformLocalToGlobalModulusTasteGate`
`single carrier alignment`

Theorem 7189.2 (CompactUniformLocalToGlobalModulus obligations)**.** *For every accepted* CompactUniformLoca *carrier $G = (K, F, L, R, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, local pointwise-modulus exposure through F, local-radius ledger visibility through L, rational lower-bound fold exactness through R, metric-triangle transport through T, uniform-modulus export through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The compact side is exhausted by K: it supplies the finite ProbeBundle coverage row over the compact metric source.

The local side is exhausted by F and L. These rows record the pointwise modulus read at each bundled center and the radius ledger attached to that read. The global precision row is then computed only by R, the finite rational lower-bound fold over the displayed ledger.

The handoff to a consumer uses T and U. The metric-triangle row transports center estimates to arbitrary source points, and the uniform-modulus row records the tolerance-only output. Since H, C, P, N only preserve, replay, source, and name the displayed rows, the listed obligations exhaust the packet. □

Theorem 7189.3 (CompactUniformLocalToGlobalModulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{FiniteNetMinimumFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longmapsto$$

In particular, the exported modulus is a finite lower-bound and metric-triangle handoff, not a host-selected modulus or an open-cover compactness theorem.

Proof. Read the carrier rows in order. The compact metric row K supplies the finite net, and the continuous-map row F supplies local pointwise-modulus evidence at the bundled centers.

The local radii do not become a global modulus until the rational fold row R is displayed. This row is finite because it folds over the ProbeBundle ledger named by K and the local-radius row L.

Finally the metric-triangle row T is the only row that transports the local estimates to arbitrary source points, and U is the only uniform-modulus consumer row. No coordinate in G stores open-cover compactness, countable choice, or a host-selected modulus, so every accepted handoff remains on the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}$ packet is generated from compact finite-net rows, continuous-map local modulus rows, local-radius ledgers, a RatUp lower-bound fold, metric-triangle transport, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7189.1, Definition 7189.2, and Definition 7189.3 bind the compact finite-net, local-modulus, RatUp fold, metric-triangle, and UniformModulus rows of the local-to-global route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformLocalToGlobalModulus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformLocalToGlobalModulusUp.CompactUniformLocalToGlobalModulusTasteGate\_singl`
Bridge status: none
Origin: human
Not claimed: This chapter does not close open-cover compactness, selected global minima outside the finite fold, countable choice, quotient function equality, ambient Heine–Cantor semantics, or a Lean carrier for CompactUniformLocalToGlobalModulus$^{\uparrow}$.
Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, local-modulus exposure, lower-bound fold exactness, metric-triangle transport, uniform-modulus export, replay, provenance, and local naming for the displayed local-to-global packet.

Chapter 7190

A Concrete Naming Certificate for $\mathsf{LocatedCauchyLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchyLimit}^{\uparrow}$ packet records the finite located-limit boundary that sits between regular Cauchy readback and the terminal real-name seal. It reads StreamName windows from chapter 126, RegSeqRat readback from chapter 301, dyadic tolerance ledgers from chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal from chapter 38, and uniqueness discipline from chapter 3942. The packet names a located candidate-limit route; it does not assert quotient completion, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7190.1 (LocatedCauchyLimit carrier)**.** *A* $\mathsf{LocatedCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, T, L_0, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the Cauchy source stream.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested precision with the displayed rational windows.*
4. *T is the located tail schedule, recording the finite threshold and neighbourhood data that make the candidate limit inspectable.*
5. *L_0 is the candidate-limit row consumed by the terminal real-name seal.*
6. *U is the uniqueness boundary, read through* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *before any two candidate-limit rows are classified together.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed candidate limit.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from stream windows through readback, tolerance, located tail schedule, candidate limit, uniqueness boundary, real seal, provenance, and naming.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyLimit}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient limits, selected representatives, countable-choice schedules, host real equality, arbitrary completion objects, or ambient real completeness.

Lean4 (statement) `BEDC.Derived.LocatedCauchyLimitUp`

Theorem 7190.2 (LocatedCauchyLimit NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCauchyLimit}^{\uparrow}$ *carrier* $L = (S, R, D, T, L_0, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyLimit}^{\uparrow}}$ *surface consists exactly of StreamName window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance comparison, located tail schedule admission, candidate-limit row admission, uniqueness-boundary routing, terminal RealUp sealing, componentwise transport, displayed replay, provenance, and local naming. No obligation is stored outside* $S, R, D, T, L_0, U, E, H, C, P, N$*.*

Proof. Project the accepted carrier. The analytic rows are S, R, D, T, L_0, U, E, and the structural rows are H, C, P, N.

The source route is forced by the carrier order: stream windows are read by S, regular rational data by R, and dyadic tolerance comparison by D. The located tail schedule T is the only row that can expose the neighbourhood threshold needed by the candidate-limit row L_0.

The uniqueness row U and the terminal seal E are downstream of the candidate-limit row. Transport, replay, provenance, and naming preserve exactly this finite route.

Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7190.3 (LocatedCauchyLimit RegSeqRat route)**.** *Every accepted* $\mathsf{LocatedCauchyLimit}^{\uparrow}$ *read reaches a candidate-limit row only through the finite route*

$$\mathsf{StreamName}^{\uparrow}(S) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(R) \longmapsto \mathsf{DyadicRatCore}^{\uparrow}(D) \longmapsto T \longmapsto L_0.$$

The route then passes through the uniqueness boundary U and the terminal $\mathsf{Real}^{\uparrow}$ *seal E before any consumer reads H, C, P, N.*

Proof. Read S first. It is the only stream-window source in the carrier, so a regular rational readback cannot bypass it.

The row R is then compared against the requested precision by the dyadic ledger D. The located schedule T records the finite tail threshold that lets the candidate-limit row L_0 be inspected.

The remaining analytic rows U and E place uniqueness and terminal real-name sealing after the candidate-limit read. The structural rows preserve the same route and add no second source. □

Theorem 7190.4 (LocatedCauchyLimit nonescape)**.** *No accepted* $\mathsf{LocatedCauchyLimit}^{\uparrow}$ *packet exports quotient completion, selected representatives, countable choice, host real equality, arbitrary completion data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every completion-facing consumer receives only* $S, R, D, T, L_0, U, E, H, C, P, N$*.*

Proof. Apply Definition 7190.2. It identifies the full local obligation surface as the displayed finite rows.

Apply Definition 7190.3. It orders the analytic source rows before the candidate-limit, uniqueness, and RealUp seal rows.

The carrier definition has no field for the excluded objects. Thus transport, replay, provenance, and naming cannot synthesize them for a completion-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyLimit}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledgers, a located tail schedule, candidate-limit row, uniqueness boundary, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7190.1, Definition 7190.2, Definition 7190.3, and Definition 7190.4 seed the finite located Cauchy limit route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, arbitrary completion objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, RegSeqRat route exactness, uniqueness-boundary routing, terminal RealUp seal routing, and nonescape for BEDC.Derived.LocatedCauchyLimitUp.

Chapter 7191

A Concrete Naming Certificate for $\mathsf{EisensteinCriterion}^{\uparrow}$

Following Eisenstein's irreducibility criterion, the $\mathsf{EisensteinCriterion}^{\uparrow}$ packet records a finite prime-divisibility route for a displayed polynomial before any irreducibility-facing consumer may use it. It sits over the polynomial carrier of chapter 50, the ring carrier of chapter 43, the integer row of chapter 31, and the prime row of chapter 64. The packet names coefficient inspection, prime divisibility, primitive-content control, and irreducibility handoff as displayed BHist and $\mathsf{NameCert}$ data; it is not a host factorization oracle, a quotient polynomial equality, a selected algebraic root, or an ambient unique-factorization theorem.

Definition 7191.1 (Eisenstein criterion carrier)**.** *An* $\mathsf{EisensteinCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, p, V, L, Q, G, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the polynomial coefficient window, listing the finitely many inspected coefficients of the displayed* $\mathsf{Poly}^{\uparrow}$ *row over the* $\mathsf{Ring}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$ *interfaces.*
2. *p is the displayed* $\mathsf{Prime}^{\uparrow}$ *row naming the prime used by the divisibility test.*
3. *V is the divisibility ledger requiring every non-leading coefficient inspected by F to be divisible by p.*
4. *L is the leading-coefficient nondivisibility row, recording that the leading coefficient is not divisible by p.*
5. *Q is the constant-coefficient square-nondivisibility row, recording that p^2 does not divide the constant coefficient.*
6. *G is the primitive-content boundary, exposing the displayed content data needed before irreducibility may be read.*
7. *I is the irreducibility handoff row, available only after F, p, V, L, Q, G have been displayed.*
8. *H records componentwise* hsame *transport for the coefficient, prime, divisibility, nondivisibility, content, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the coefficient window through the prime-divisibility route and the irreducibility handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{EisensteinCriterion}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EisensteinCriterion}^{\uparrow}}$ *row.*

The carrier exposes only the polynomial coefficient window, displayed prime, divisibility ledger, leading nondivisibility, constant square-nondivisibility, primitive-content boundary, irreducibility handoff, transport, replay, provenance, and local naming. It has no coordinate for host factorization, quotient polynomial equality, selected roots, ambient unique factorization, or algebraic-closure reasoning.

Theorem 7191.2 (Eisenstein criterion NameCert obligations)**.** *For every accepted* $\mathsf{EisensteinCriterion}^{\uparrow}$ *carrier*

$$E = (F, p, V, L, Q, G, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EisensteinCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the coefficient window, prime row, divisibility ledger, nondivisibility rows, primitive-content boundary, irreducibility handoff, transport, replay, provenance, and local name;*
2. *coefficient-window exactness, requiring F to expose the inspected polynomial coefficients through the* $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, *and* $\mathsf{Int}^{\uparrow}$ *rows;*
3. *prime-row exactness, requiring the divisibility ledger to read the displayed* $\mathsf{Prime}^{\uparrow}$ *row p;*
4. *divisibility exactness, requiring V to cover every non-leading coefficient and no coefficient outside the displayed window;*
5. *leading and constant boundary checks, requiring L and Q to record respectively leading nondivisibility and constant square-nondivisibility;*
6. *primitive-content routing, requiring G to be displayed before the irreducibility handoff I is consumed;*
7. *structural stability through H, C, P, N for the same finite rows;*
8. *refusal of host factorization, quotient polynomial equality, selected roots, ambient unique factorization, and algebraic-closure reasoning as local certificate data.*

No Eisenstein criterion obligation is stored outside these rows.

Proof. Project the carrier of Definition 7191.1. The visible rows are exactly $F, p, V, L, Q, G, I, H, C, P, N$, so carrier admission and local naming can mention no further algebraic data.

The coefficient and prime obligations are read first. The row F supplies the displayed polynomial coefficients through the polynomial, ring, and integer interfaces, while p supplies the displayed prime row used by all divisibility checks.

The local divisibility surface is then exhausted by V, L, Q. The row V covers the non-leading coefficients, L covers the leading coefficient, and Q covers the constant coefficient against p^2.

Finally G must be displayed before I, and H, C, P, N only transport, replay, source, and name the same route. Since the excluded host principles do not occur as carrier coordinates, they cannot discharge a local NameCert obligation. □

Theorem 7191.3 (Eisenstein criterion prime-divisibility route)**.** *Every irreducibility-facing consumer of an accepted* $\mathsf{EisensteinCriterion}^{\uparrow}$ *carrier*

$$E = (F, p, V, L, Q, G, I, H, C, P, N)$$

must read the finite route

$$F, p \longmapsto V, L, Q \longmapsto G \longmapsto I \longmapsto H, C, P, N$$

before any irreducibility handoff is exported. Thus the consumer receives the coefficient window, displayed prime, non-leading divisibility ledger, leading nondivisibility row, constant square-nondivisibility row, and primitive-content boundary before it can inspect the handoff row.

Proof. Start with an irreducibility-facing read of the row I. By Definition 7191.1, the row I has no independent source of polynomial or prime data; it is downstream of F, p, V, L, Q, G.

Apply Definition 7191.2. The certificate obligations force the coefficient window and prime row to be exposed before the divisibility ledger, force the leading and constant boundary rows to be exposed beside that ledger, and force the primitive-content row before the handoff.

The structural rows do not insert a second handoff. The rows H, C, P, N preserve and name the same finite route, so any irreducibility-facing consumer reads only the displayed coefficient, prime, divisibility, nondivisibility, content, and handoff rows.

Because the carrier contains no host factorization oracle, quotient polynomial equality, selected root, ambient unique-factorization theorem, or algebraic-closure coordinate, none of those principles can be exported with the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EisensteinCriterion}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EisensteinCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7191.1, Definition 7191.2, and Definition 7191.3 seed the finite prime-divisibility certificate over polynomial coefficients, a displayed prime, divisibility and nondivisibility ledgers, primitive-content routing, irreducibility handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EisensteinCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host factorization, quotient polynomial equality, selected roots, ambient unique factorization, algebraic-closure reasoning, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window exactness, prime-row exactness, divisibility exactness, primitive-content routing, structural stability, and local naming for BEDC.Derived.EisensteinCriterionUp.

Constructive story: An $\mathsf{EisensteinCriterion}^{\uparrow}$ packet is generated from a polynomial coefficient window, displayed prime row, non-leading divisibility ledger, leading nondivisibility row, constant square-nondivisibility row, primitive-content boundary, irreducibility handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7192

A Concrete Naming Certificate for $\mathsf{CauchyRealModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{CauchyRealModulus}^{\uparrow}$ packet records a real-name carrier whose convergence evidence is an explicit modulus row rather than a host completeness principle. It sits between the dyadic tolerance core of chapter 291, the stream schedule of chapter 126, the regular rational-name surface of chapter 301, and the real-seal boundary of chapter 38. The packet gives downstream completion consumers a finite place to cite the modulus evidence attached to a Cauchy real name; it does not select a limit, quotient streams, or import countable choice.

Definition 7192.1 (CauchyRealModulus carrier)**.** *A* $\mathsf{CauchyRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, \mu, Q, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are available to a real-name consumer.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reading the rational approximants through the displayed stream schedule.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the regular-name comparisons.*
4. μ *is the modulus row assigning each displayed dyadic tolerance request to a visible stream window.*
5. Q *is the Cauchy coherence classifier, comparing accepted packets only through* S, R, D, μ.
6. H *stores componentwise* hsame *transport for* S, R, D, μ, Q, C, P, N.
7. C *stores displayed* Cont *replay of the stream schedule, regular rational readback, dyadic tolerance ledger, modulus row, coherence classifier, provenance, and local naming.*
8. P *is the* Pkg *provenance over* $\mathsf{CauchyRealModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected real limit, host real equality, quotient-stream equality, ambient completeness, or countable choice.

Theorem 7192.2 (CauchyRealModulus obligation surface)**.** *For every accepted* $\mathsf{CauchyRealModulus}^{\uparrow}$ *carrier* $M = (S, R, D, \mu, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealModulus}^{\uparrow}}$ *surface consists exactly of stream-window admission through* S*, regular rational-name exposure through* R*, dyadic tolerance exposure through* D*, explicit modulus visibility through* μ*, Cauchy coherence classification through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The analytic rows are exhausted by the stream schedule, the regular rational readback, the dyadic tolerance ledger, the modulus row, and the coherence classifier.

The modulus evidence is not inferred from ambient completeness. It is a row of the same packet, so a consumer must expose μ together with S, R, D before reading the classifier Q.

The structural rows H, C, P, N only transport, replay, source, and name those displayed coordinates. Since the carrier lists no selected limit, quotient-stream equality, host real equality, completeness theorem, or choice principle, the obligation surface is exactly the finite inventory above. □

Theorem 7192.3 (CauchyRealModulus Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyRealModulus}^{\uparrow}$ *packet factors its read through the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*, the explicit modulus row* μ*, the coherence classifier* Q*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The read cannot manufacture a selected limit, quotient equality of streams, host real equality, ambient completeness, or countable choice.*

Proof. Begin with the carrier inventory of Definition 7192.1. The only route from the real-name packet toward a $\mathsf{Real}^{\uparrow}$ consumer is the displayed chain S, R, D, μ, Q.

The replay row C forces the same stream, regular-name, tolerance, and modulus rows to be reopened before the real-seal boundary is consumed. Transport through H preserves those rows componentwise, and provenance through P records the same source route for the local naming row N.

The excluded objects are absent from the packet. Therefore the $\mathsf{Real}^{\uparrow}$ read may cite an explicitly modulated Cauchy real name, but it cannot create a selected limit, quotient-stream equality, host equality of reals, completeness theorem, or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealModulus}^{\uparrow}$ packet is generated from stream-window rows, regular rational-name rows, dyadic tolerance rows, an explicit modulus row, a Cauchy coherence classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7192.1, Definition 7192.2, and Definition 7192.3 bind the explicit-modulus Cauchy real-name carrier to the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealModulus}^{\uparrow}}$, formalTarget)

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected limits, quotient equality of streams, host real equality, ambient completeness, countable choice, or a Lean carrier for $\mathsf{CauchyRealModulus}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular rational readback, dyadic tolerance exposure, explicit modulus visibility, classifier coherence, replay, provenance, and local naming for the displayed packet.

Chapter 7193

A Concrete Naming Certificate for $\mathsf{ArgumentPrinciple}^{\uparrow}$

Following Cauchy's argument principle, the $\mathsf{ArgumentPrinciple}^{\uparrow}$ packet records the finite contour-and-winding ledger by which a holomorphic input with displayed zero and pole boundary data can be read as a certified counting relation. It sits after the complex carrier of chapter 39, the holomorphic source of chapter 68, the contour-integral operation of chapter 521, the residue-facing handoff of chapter 3978, and the reparameterization ledger of chapter 7037. The packet exposes only finite contour, meromorphic boundary, winding-count, residue-handoff, transport, replay, provenance, and local naming rows; it does not import a host path quotient, an analytic-continuation principle, or an unregistered zero-counting theorem.

Definition 7193.1 (Argument principle carrier)**.** *An* $\mathsf{ArgumentPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, F, Z, P_0, W, I, R, Q, H, C, K, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{PLContour}$ *row with* $\mathsf{Complex}^{\uparrow}$ *endpoints, segment rows, and a displayed non-boundary locus for zeros and poles.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *row on the displayed contour neighbourhood outside the recorded pole locus.*
3. *Z is the finite zero ledger, listing local zero witnesses and multiplicity rows that are visible to the packet.*
4. *P_0 is the finite pole ledger, listing local pole witnesses and order rows that are visible to the packet.*
5. *W is the winding-number row computed from the contour row and the displayed zero-pole boundary data.*
6. *I is the* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row for the logarithmic-derivative contour read.*
7. *R is the* $\mathsf{ContourResidueHandoff}^{\uparrow}$ *row that exposes the corresponding residue-facing data.*
8. *Q is the* $\mathsf{ContourReparameterizationLedger}^{\uparrow}$ *row recording that allowed contour refinements preserve the same visible winding and integral rows.*
9. *H records componentwise* hsame *transport for G, F, Z, P_0, W, I, R, Q.*

10. *C records displayed* Cont *replay for contour, holomorphic, zero, pole, winding, integral, residue, reparameterization, provenance, and naming reads.*
11. *K is* Pkg *provenance over* ArgumentPrinciple$^{\uparrow}$, Complex$^{\uparrow}$, Holomorphic$^{\uparrow}$, ContourIntegralOperation$^{\uparrow}$, ContourResidueHandoff$^{\uparrow}$, ContourReparameterizationLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ArgumentPrinciple}^{\uparrow}}$ *row.*

The classifier accepts only packets with these displayed rows. It has no coordinate for quotient path equality, analytic continuation outside the displayed domain, unbounded zero searches, or a host meromorphic-function object.

Lean4 (definition) `BEDC.Derived.ArgumentPrincipleUp.ArgumentPrincipleUp`

Theorem 7193.2 (Argument principle NameCert obligations)**.** *For the carrier of Definition 7193.1, the local* $\mathsf{NameCert}_{\mathsf{ArgumentPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission, contour exposure, holomorphic-source exposure, zero-ledger exposure, pole-ledger exposure, winding-number exposure, contour-integral operation exposure, residue-handoff exposure, reparameterization-ledger exposure, componentwise transport, displayed replay, provenance, and local naming. An argument-principle consumer receives no host path quotient, analytic-continuation oracle, unbounded zero search, or global meromorphic-function principle.*

Proof. Project the carrier. The contour row G, holomorphic row F, zero ledger Z, pole ledger P_0, winding row W, integral row I, residue handoff R, and reparameterization ledger Q are the only analytic coordinates available to the local certificate.

The counting read is therefore forced through the displayed zero and pole ledgers before it may inspect the winding row. The contour-integral operation and residue handoff expose the corresponding logarithmic-derivative route, while the reparameterization ledger keeps permitted contour refinements attached to the same visible rows.

Transport, replay, provenance, and local naming preserve this finite inventory. Since the packet contains no coordinate for quotient path equality, analytic continuation outside the displayed domain, unbounded zero search, or a host meromorphic-function object, none of those objects can be exported by the NameCert surface. □

Theorem 7193.3 (Argument principle winding-number route)**.** *Let $A = (G, F, Z, P_0, W, I, R, Q, H, C, K, N)$ be an accepted* ArgumentPrinciple$^{\uparrow}$ *carrier. Every consumer that reads the argument-principle count must first expose the* Complex$^{\uparrow}$ *contour endpoints in G, the* Holomorphic$^{\uparrow}$ *source row F, the finite zero ledger Z, the finite pole ledger P_0, the winding-number row W, the* ContourIntegralOperation$^{\uparrow}$ *row I, the* ContourResidueHandoff$^{\uparrow}$ *row R, the* ContourReparameterizationLedger$^{\uparrow}$ *row Q, and the structural rows H, C, K, N. The resulting route identifies the visible winding count with the displayed zero-minus-pole ledger count without using a host path quotient or analytic-continuation principle.*

Proof. Begin with the NameCert obligations. They expose $G, F, Z, P_0, W, I, R, Q, H, C, K, N$ as the complete carrier surface for any public count read.

The zero and pole ledgers provide the finite counting side, while the winding row records the contour-facing count. The contour-integral operation row supplies the logarithmic-derivative read, and the residue handoff ties that read to the displayed local residue data.

Finally use the reparameterization ledger to transport allowed refinements back to the same displayed contour surface. Componentwise hsame transport and Cont replay keep all rows under the same provenance and local naming certificate, so the consumer obtains only the finite winding-count route displayed by the packet. □

Falsifiable prediction: An accepted $\mathsf{ArgumentPrinciple}^{\uparrow}$ count read is refuted if it omits any displayed contour row, holomorphic row, zero ledger, pole ledger, winding row, contour-integral operation row, residue-handoff row, reparameterization ledger, transport row, replay row, provenance row, or local NameCert row.

Independence witness: complex, holomorphic, contourintegraloperation, contour-residue-handoff, contour-reparameterization-ledger: none is bijective to $\mathsf{ArgumentPrinciple}^{\uparrow}$ because this packet jointly binds the visible zero and pole ledgers, winding row, logarithmic-derivative contour operation, residue handoff, contour-reparameterization ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArgumentPrinciple}^{\uparrow}}$.

Constructive story: An $\mathsf{ArgumentPrinciple}^{\uparrow}$ packet is generated from a displayed PL-contour row, holomorphic source row, finite zero ledger, finite pole ledger, winding-number row, contour-integral operation row, residue-handoff row, reparameterization ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArgumentPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7193.1, Definition 7193.2, and Definition 7193.3 seed the finite argument-principle carrier, obligation surface, and winding-count consumer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArgumentPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient path equality, analytic continuation outside the displayed domain, unbounded zero search, or a host meromorphic-function theorem for $\mathsf{ArgumentPrinciple}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, zero-ledger exposure, pole-ledger exposure, winding-row exactness, contour-integral routing, residue handoff, reparameterization stability, transport, replay, provenance, and local naming for BEDC.Derived.ArgumentPrincipleUp.

Chapter 7194

A Concrete Naming Certificate for $\mathsf{UniformModulusErrorBudget}^{\uparrow}$

Following Bishop and Bridges compact-uniform continuity, the $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ packet records the finite error-budget ledger used to pass from compact finite nets and pointwise moduli to a tolerance-only uniform-modulus handoff. It reads the compact metric source of chapter 129, the continuous-map graph of chapter 130, the uniform-modulus consumer of chapter 369, the rational lower-bound surface of chapter 37, the metric triangle rows of chapter 57, and the Real seal of chapter 38. The packet names only finite epsilon-third, compact-net, rational-fold, metric-triangle, transport, replay, provenance, and local naming rows.

Definition 7194.1 (Uniform modulus error-budget carrier)**.** *A* $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, F, U, Q, M, R, B, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the same metric source.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that receives the tolerance-only precision history.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius and lower-bound fold row over the displayed compact-net probes.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *source-and-target triangle row used to split and recombine the error estimate.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed source and target distance estimates.*
7. *B is the epsilon-third budget row, recording the three visible tolerances, their rational comparisons, and the folded precision consumed by U.*
8. *T is the compact-net triangle ledger tying X, F, Q, M, R, B to one finite estimate route.*
9. *H records componentwise* hsame *transport for X, F, U, Q, M, R, B, T.*
10. *C records displayed* Cont *replay for compact-net reads, pointwise-modulus reads, rational-fold reads, metric-triangle reads, Real-seal reads, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{UniformModulusErrorBudget}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{UniformModulusErrorBudget}^{\uparrow}}$ *row.*

Accepted packets are compared only through $X, F, U, Q, M, R, B, T, H, C, P, N$. They have no coordinate for a selected Lebesgue number, an external modulus selector, quotient equality of metric points, sequential compactness, countable choice, or ambient Real completeness.

Theorem 7194.2 (Uniform modulus error-budget NameCert obligations)**.** *For the carrier of Definition 7194.1, the local* $\mathsf{NameCert}_{\mathsf{UniformModulusErrorBudget}^{\uparrow}}$ *surface consists exactly of compact-metric finite-net exposure, continuous-map pointwise-modulus exposure, uniform-modulus consumer exposure, rational lower-bound fold exposure, metric triangle exposure, Real-seal exposure, epsilon-third budget exposure, compact-net triangle-ledger exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier. The compact finite-net row X, continuous-map row F, rational fold Q, metric row M, Real seal R, and budget row B are the only rows from which a tolerance estimate can be read.

The budget row splits the requested tolerance into three displayed rational comparisons. The compact-net ledger T then routes the source point to a displayed center, applies the pointwise-modulus row at that center, and uses the metric triangle row to recombine the three estimates before the uniform-modulus consumer row U is exposed.

Transport, replay, provenance, and local naming keep this same finite inventory. Since the carrier contains no selected Lebesgue number, external modulus selector, quotient point equality, sequence extractor, choice principle, or ambient Real-completeness row, none of those objects can be exported by the NameCert surface. □

Theorem 7194.3 (Uniform modulus error-budget triangle route)**.** *Let $E = (X, F, U, Q, M, R, B, T, H, C, P, N)$ be an accepted* $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ *carrier. Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer must read the epsilon-third budget B, the compact-net row X, the continuous-map row F, the* $\mathsf{Rat}^{\uparrow}$ *fold row Q, the* $\mathsf{Met}^{\uparrow}$ *triangle row M, and the* $\mathsf{Real}^{\uparrow}$ *seal row R before reaching U. The handoff realizes the compact-uniform-continuity estimate as a finite triangle budget and exports no host compactness theorem or external modulus function.*

Proof. Begin with the accepted carrier and expose the budget row B. Its three rational comparisons are displayed histories, so the tolerance split is part of the packet rather than a hidden arithmetic choice.

Next read the compact-net route. The row X supplies the finite centers, F supplies pointwise-modulus data at those centers, and Q folds the visible positive radii to a common precision history.

Finally apply the metric triangle ledger. The row M combines the source-center, source-pair, and target-distance estimates, while R seals the displayed Real distance rows before U consumes the resulting tolerance-only modulus. The structural rows H, C, P, N transport and replay this route without adding host compactness, quotient equality, or an external modulus function. □

Falsifiable prediction: An accepted $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ consumer is refuted if it reaches the UniformModulusUp handoff without displaying the epsilon-third budget, CompactMetric row, ContinuousMap row, RatUp fold, Metric triangle row, RealUp seal, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: compactmetric, continuousmap, uniformmodulus, rat, metric, real: none is bijective to $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ because this packet jointly carries the epsilon-third budget, compact finite-net route, pointwise-modulus rows, rational lower-bound fold, metric triangle estimates, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformModulusErrorBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformModulusErrorBudget}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, a UniformModulus consumer row, Rat positive-radius folds, Metric triangle rows, Real distance seals, an epsilon-third budget row, a compact-net triangle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformModulusErrorBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7194.1, Definition 7194.2, and Definition 7194.3 seed the finite epsilon-third and metric-triangle budget consumed by compact-uniform-continuity routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformModulusErrorBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected Lebesgue numbers, external modulus selection, quotient metric equality, sequential compactness, countable choice, or ambient Real completeness for $\mathsf{UniformModulusErrorBudget}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, budget-row exactness, compact-net exposure, continuous-map exposure, rational-fold exactness, metric-triangle compatibility, Real-seal nonescape, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformModulusErrorBudgetUp.

Chapter 7195

A Concrete Naming Certificate for $\mathsf{RegularSequenceCompletion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and constructive completion, the $\mathsf{RegularSequenceCompletion}^{\uparrow}$ packet records the finite BEDC route by which a regular rational sequence is admitted to a real-name seal without quotient streams, countable choice, or an ambient completeness theorem. It sits after the stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the cofinal tail-budget surface of chapter 698, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a displayed completion route for regular sequences, not a selected host limit.

Definition 7195.1 (Regular sequence completion carrier)**.** *A* $\mathsf{RegularSequenceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, D, R, T, E, H, K, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing finite windows of the sequence name.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row exposing dyadic tolerances and arithmetic comparison data.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reading the regular rational approximants through S and D.*
4. *T is the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *row recording which tail windows are cofinal for each displayed tolerance request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal route, allowing a completion-facing consumer to read a real-name boundary only through S, D, R, T.*
6. *H stores componentwise* hsame *transport for S, D, R, T, E, K, P, N.*
7. *K stores displayed* Cont *replay of the stream, dyadic, regular-rational, tail-budget, real-seal, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, T, E, H, K, P, N$. It has no coordinate for a quotient of streams, a selected representative, host real equality, countable choice, or ambient completeness.

Theorem 7195.2 (Regular sequence completion NameCert obligations)**.** *For every accepted* RegularSequenceComp *carrier* $C = (S, D, R, T, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceCompletion}^{\uparrow}}$ *surface consists of the following five obligation rows:*

1. *carrier admission for the displayed* BHist *packet* C*;*
2. *window and tolerance exposure through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. *regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*;*
4. *cofinal tail-budget exposure through* $\mathsf{CofinalTailBudget}^{\uparrow}$*;*
5. *real-seal replay through* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7195.1. The analytic data are exactly the finite stream row, dyadic tolerance row, regular-rational row, tail-budget row, and real-seal row.

The structural rows do not add analytic content. Transport through H preserves the displayed rows componentwise, replay through K reopens the same finite route, and P, N record provenance and local naming.

Since the packet contains no quotient-stream coordinate, selected representative, host real equality, choice functional, or ambient completeness field, the NameCert obligations are exhausted by the five rows above. □

Theorem 7195.3 (Regular sequence completion Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularSequenceCompletion}^{\uparrow}$ *packet must factor its read through the finite window row* S*, dyadic tolerance row* D*, regular rational readback row* R*, cofinal tail-budget row* T*, real-seal route* E*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The consumer cannot extract quotient-stream equality, a selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Begin with the obligation surface of Definition 7195.2. It requires the stream, dyadic, regular-rational, tail-budget, and real-seal rows before the packet can be consumed as a completion route.

The sibling route is finite: chapter 301 supplies the regular rational readback, chapter 698 supplies the cofinal tail budget, and chapter 38 supplies the terminal seal. The present packet binds those rows to one completion-facing carrier.

Therefore a consumer may cite a displayed regular-sequence completion route, but it cannot manufacture any of the refused quotient, selection, equality, choice, or ambient-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularSequenceCompletion}^{\uparrow}$ packet is generated from stream-window rows, dyadic tolerance rows, regular rational readback, a cofinal tail-budget row, a real-seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularSequenceCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7195.1, Definition 7195.2, and Definition 7195.3 bind the regular-sequence completion carrier to the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CofinalTailBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularSequenceCompletion}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close quotient equality of streams, selected limits, host real equality, countable choice, ambient completeness, or a Lean carrier for $\mathsf{RegularSequenceCompletion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance exposure, regular rational readback, cofinal tail-budget exposure, real-seal replay, provenance, and local naming for the displayed packet.

Chapter 7196

A Concrete Naming Certificate for $\mathsf{DyadicNewtonSqrt}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Newton method for square roots, the $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ packet records the finite dyadic data needed to run a guarded Newton square-root computation before any completed real quotient, selected limit, or host square-root function is available. It sits between the dyadic arithmetic surfaces of chapter 1658 and chapter 291, the regular rational readback of chapter 301, the Cauchy schedule of chapter 292, the terminal seal of chapter 38, and the Heron square-root consumer of chapter 3818. The carrier exposes a Newton-specific square-residual and quadratic-window ledger as named rows; it does not assert a total square-root operation on arbitrary $\mathsf{Real}^{\uparrow}$ data.

Definition 7196.1 (Dyadic Newton square-root carrier)**.** *A* $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (A, G, W, R, Q, M, S, E, H, C, P, L)$$

with the following displayed rows:

1. *A is the radicand* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, carrying the dyadic rational value, denominator witness, square ledger, and comparison data used by every Newton step.*
2. *G is the positive-window ledger, recording the lower bound, upper guard, and nonzero denominator witnesses that keep each displayed iterate inside the positive dyadic window.*
3. *W is the Newton iterate window, a finite row of dyadic iterates computed by the visible step $x \mapsto (x + A/x)/2$ from A and G.*
4. *R is the residual-square tolerance row, storing the finite comparisons between x_i^2 and A together with the requested dyadic error budget.*
5. *Q is the quadratic contraction budget, recording the local guard under which the next residual is bounded by the displayed square of the previous residual.*
6. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff row, translating a requested output precision into the displayed iterate threshold and residual tolerance.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by replaying A, G, W, R, Q, M as a regular rational name.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the dyadic Newton ledger and regular readback row have been displayed.*

9. *H records componentwise* hsame *transport for the radicand, positive-window ledger, iterate window, residual tolerance, contraction budget, modulus handoff, readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of each square-root precision request through $A, G, W, R, Q, M, S, E, H, P, L$.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicNewtonSqrt}^{\uparrow}$, $\mathsf{Dyadic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeronSquareRoot}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{DyadicNewtonSqrt}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, G, W, R, Q, M, S, E, H, C, P, L$. It has no coordinate for quotient real equality, selected completed limits, arbitrary trichotomy, total square root on all $\mathsf{Real}^{\uparrow}$ *names, ambient completeness, or a host square-root theorem.*

Theorem 7196.2 (Dyadic Newton square-root NameCert obligations)**.** *For every accepted* $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ *carrier*

$$N = (A, G, W, R, Q, M, S, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicNewtonSqrt}^{\uparrow}}$ *surface consists exactly of the displayed carrier admission row, radicand* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness, positive-window exactness, Newton iterate-window stability, residual-square tolerance accounting, quadratic contraction budget visibility,* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The radicand row A and positive-window row G supply the dyadic arithmetic and positivity witnesses before any iterate is visible.

The numerical work is finite. The window row W lists the displayed Newton iterates, the residual row R stores the square-error comparisons, and the contraction row Q records the guarded quadratic estimate for the next residual.

The output rows are downstream of that ledger. The modulus row M selects a finite threshold for the requested precision, the $\mathsf{RegSeqRat}^{\uparrow}$ row S is obtained by replaying the displayed dyadic rows, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after S has been displayed.

Finally H, C, P, L transport, replay, source, and name the same finite packet. Since the carrier contains no host square-root coordinate, selected limit, quotient equality, trichotomy classifier, or ambient completeness row, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7196.3 (Dyadic Newton square-root quadratic window route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer that reads a square-root approximation from an accepted* $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ *carrier must factor through the finite route*

$$A, G \longmapsto W, R, Q \longmapsto M \longmapsto S \longmapsto E.$$

Thus the consumer first reads the $\mathsf{Dyadic}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic surface, then the positive Newton window, residual-square tolerance, quadratic contraction budget,* $\mathsf{CauchyModulus}^{\uparrow}$ *precision handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no host square-root function, quotient real equality, selected completed limit, arbitrary trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier inventory of Definition 7196.1. The rows A, G are the only source of radicand, positivity, and denominator data, so every Newton iterate must read them before any residual comparison can be formed.

The finite window is then inspected through W, R, Q. The row W stores the displayed Newton iterates, R stores the square-residual tolerance checks, and Q stores the guarded quadratic contraction estimate that makes the window a Cauchy-facing ledger.

The precision handoff uses M. It turns the requested output tolerance into a finite iterate threshold, after which S supplies the regular rational readback and E supplies the terminal Real seal.

The structural rows H, C, P, L preserve the same dependency order under transport, replay, provenance, and naming. Therefore a downstream $\mathsf{HeronSquareRoot}^{\uparrow}$ or Real-facing consumer receives only the displayed DyadicRatCore-to-RegSeqRat-to-Real route, not an external square-root value or ambient completeness principle. □

Falsifiable prediction: Every accepted DyadicNewtonSqrtUp read factors through a DyadicRatCore radicand row, a positive-window ledger, a finite Newton iterate window, residual-square tolerance checks, a quadratic contraction budget, a CauchyModulus handoff, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadic, dyadicratcore, cauchymodulus, regseqrat, real, heronsquareroot: $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ is not bijective to any listed sibling because it binds a positive dyadic radicand, Newton iterate windows, residual-square tolerances, a quadratic contraction budget, CauchyModulus handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming as one finite square-root certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNewtonSqrt}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicNewtonSqrt}^{\uparrow}$ packet is generated from a DyadicRatCore radicand row, a positive-window ledger, finite Newton iterate windows, residual-square tolerance checks, a quadratic contraction budget, CauchyModulus handoff, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicNewtonSqrt}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7196.1, Definition 7196.2, and Definition 7196.3 seed the finite Newton square-root packet over $\mathsf{Dyadic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeronSquareRoot}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicNewtonSqrt}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total square root on arbitrary $\mathsf{Real}^{\uparrow}$ names, quotient real equality, selected completed limits, arbitrary trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radicand exactness, positive-window exactness, Newton iterate stability, residual-square tolerance accounting, quadratic contraction visibility, CauchyModulus handoff, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicNewtonSqrtUp.

Chapter 7197

A Concrete Naming Certificate for $\mathsf{CantorBendixsonRank}^{\uparrow}$

Following Cantor and Bendixson, the $\mathsf{CantorBendixsonRank}^{\uparrow}$ packet records a finite derivative-rank certificate for the displayed compact-metric part of a Cantor–Bendixson analysis. It sits after the derivative ledger of chapter 2024, the reusable perfect-kernel face of chapter 3863, the nowhere-dense boundary of chapter 4186, the compact metric carrier of chapter 129, and the source metric row of chapter 57. The packet names only a finite rank budget, its inspected derivative rows, and its residual-kernel handoff; it does not assert a host ordinal rank, transfinite powerset construction, selected enumeration of isolated points, quotient closed-set equality, countable choice, or ambient compactness theorem.

Definition 7197.1 (Cantor–Bendixson rank carrier)**.** *A* $\mathsf{CantorBendixsonRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, K, D, r, I, P, N_0, W, H, C, Q, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, carrying the metric observation surface used by the derivative requests.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, exposing the finite compact inspection and cover budgets on which the derivative search is allowed to depend.*
3. *D is the* $\mathsf{CantorBendixson}^{\uparrow}$ *derivative ledger, listing the displayed derivative stages already admitted by the sibling Cantor–Bendixson packet.*
4. *r is the finite rank budget row, recording the displayed natural-number depth at which the inspected derivative route stops.*
5. *I is the isolated-remnant ledger, recording only those isolated reads witnessed before the budget row r is exhausted.*
6. *P is the* $\mathsf{PerfectSet}^{\uparrow}$ *residual-kernel row, read only for the part that survives the displayed derivative budget.*
7. *N_0 is the* $\mathsf{NowhereDense}^{\uparrow}$ *boundary row, recording the nowhere-dense refusal surface consumed by scattered-remnant or category-facing reads.*
8. *W is the finite window schedule for metric neighborhoods, compact inspections, derivative rows, isolated-remnant reads, perfect-kernel reads, and nowhere-dense boundary reads.*

9. *H records componentwise* hsame *transport for M, K, D, r, I, P, N_0, W and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of every accepted rank request through*

$$M, K, D, r, I, P, N_0, W \longmapsto H, C, Q, N.$$

11. *Q is the* Pkg *provenance over* $\mathsf{CantorBendixsonRank}^{\uparrow}$, $\mathsf{CantorBendixson}^{\uparrow}$, $\mathsf{PerfectSet}^{\uparrow}$, $\mathsf{NowhereDense}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBendixsonRank}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host ordinals, transfinite derivative intersections, ambient powersets, quotient closed-set equality, a selected enumeration of isolated points, countable choice, or compactness outside the displayed compact-metric route.

Theorem 7197.2 (Cantor–Bendixson rank NameCert obligations)**.** *For every accepted* $\mathsf{CantorBendixsonRank}^{\uparrow}$ *carrier*

$$R = (M, K, D, r, I, P, N_0, W, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorBendixsonRank}^{\uparrow}}$ *surface is exhausted by the following finite obligation rows:*

1. *metric and compact admission through M and K, exposing only displayed metric observations, compact inspection rows, and finite cover budgets;*

2. *derivative-ledger admission through D, requiring every rank read to cite the displayed* $\mathsf{CantorBendixson}^{\uparrow}$ *derivative stages it consumes;*

3. *finite-rank exactness through r and W, requiring the rank budget to name the inspected derivative depth and the windows used at that depth;*

4. *isolated-remnant exactness through I, requiring every scattered read to carry a displayed isolation witness before the rank budget is exhausted;*

5. *residual-kernel and nowhere-dense compatibility through P and N_0, requiring perfect-kernel reads and nowhere-dense refusals to remain attached to the same displayed derivative route;*

6. *structural nonescape through H, C, Q, N, requiring transport, replay, provenance, and local naming to export no host ordinal rank, transfinite powerset construction, selected isolated-point enumeration, quotient closed-set equality, countable choice, or ambient compactness theorem.*

No finite derivative-rank obligation is stored outside these rows.

Proof. Project the carrier of Definition 7197.1. The metric and compact inputs are exactly M and K, so every rank request begins with displayed observations and compact inspection data rather than with a host powerset.

The rank work is finite. The derivative ledger D supplies the Cantor–Bendixson stages already visible to the packet, while r and W record the finite depth and the windows inspected at that depth.

The two outputs stay attached to that same route. The isolated-remnant row I records only witnessed scattered reads before the budget is exhausted, and the residual rows P, N_0 record the perfect-kernel and nowhere-dense boundaries that survive the displayed route.

Finally H, C, Q, N transport, replay, source, and name those rows. Since the carrier has no coordinate for host ordinal ranks, transfinite intersections, selected isolated-point enumerations,

quotient closed-set equality, countable choice, or external compactness, the six obligation rows exhaust the local certificate. □

Theorem 7197.3 (Cantor–Bendixson rank finite derivative budget)**.** *Every accepted* $\mathsf{CantorBendixsonRank}^{\uparrow}$ *consumer that asks for a derivative-rank read must first display the route*

$$M, K, D, r, W \longmapsto I, P, N_0 \longmapsto H, C, Q, N.$$

Thus the exported rank is a finite BEDC budget over the $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CantorBendixson}^{\uparrow}$*,* $\mathsf{PerfectSet}^{\uparrow}$*, and* $\mathsf{NowhereDense}^{\uparrow}$ *rows. The consumer receives no host ordinal, transfinite derivative chain, powerset decomposition, selected isolated-point enumeration, quotient closed-set equality, countable-choice schedule, or ambient compactness theorem.*

Proof. Begin with the metric and compact rows. The source metric observations and compact inspection budgets are the only neighborhood data available to the derivative read.

Next read the derivative ledger and budget. The row D supplies the displayed Cantor–Bendixson stages, while r and W identify the finite depth and windows at which the rank request is inspected.

The scattered and residual faces are downstream of that budget. Row I records isolated remnants only when they have displayed witnesses, and rows P, N_0 record the retained perfect-kernel and nowhere-dense boundaries without detaching from the same derivative ledger.

The structural rows preserve the route and add no new mathematical coordinates. Therefore the exported object is exactly a finite derivative-rank budget over the named sibling packets, not any of the excluded host-level or choice-dependent data. □

Falsifiable prediction: Every accepted $\mathsf{CantorBendixsonRank}^{\uparrow}$ consumer that reads a derivative-rank packet while omitting one of the Metric, CompactMetric, CantorBendixson derivative, finite rank budget, isolated-remnant, PerfectSet residual, NowhereDense boundary, window, hsame, Cont, Pkg, or NameCert rows has left the packet.

Independence witness: cantorbendixson, perfectset, nowheredense, compactmetric, metric: $\mathsf{CantorBendixsonRank}^{\uparrow}$ is not bijective with any listed sibling because it joins metric observations, compact inspection, derivative stages, a finite rank budget, isolated remnants, residual perfect-kernel data, nowhere-dense refusals, windows, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBendixsonRank}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorBendixsonRank}^{\uparrow}$ packet is generated from MetricUp observations, CompactMetricUp inspection rows, CantorBendixsonUp derivative ledgers, a finite rank budget, isolated-remnant witnesses, PerfectSetUp residual rows, NowhereDenseUp boundary rows, finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorBendixsonRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7197.1, Definition 7197.2, and Definition 7197.3 seed the finite derivative-rank surface for compact-metric Cantor–Bendixson packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorBendixsonRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host ordinal rank, transfinite powerset decomposition, selected isolated-point enumeration, quotient closed-set equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-ledger admission, finite-rank exactness, isolated-remnant exactness, residual-kernel compatibility, nowhere-dense boundary compatibility, structural nonescape, and route replay for BEDC.Derived.CantorBendixsonRankUp.

Chapter 7198

A Concrete Naming Certificate for ConstructiveCauchyTailOperator$^{\uparrow}$

Following Bishop and Bridges, the ConstructiveCauchyTailOperator$^{\uparrow}$ packet names the finite tail-shift operation needed by constructive Cauchy completion without selecting a hidden quotient representative. It sits over StreamName$^{\uparrow}$ of chapter 126, RegSeqRat$^{\uparrow}$ of chapter 301, Real$^{\uparrow}$ of chapter 38, DyadicRatCore$^{\uparrow}$ of chapter 291, FiniteTailFilter$^{\uparrow}$ of chapter 1119, SelectedTailSeed$^{\uparrow}$ of chapter 1207, and TailCofinalityBudget$^{\uparrow}$ of chapter 1231. Those siblings expose streams, regular rational readback, dyadic tolerances, finite tail admission, selected seeds, and cofinal threshold budgets; this chapter packages the operator that transports a displayed Cauchy tail across those rows before the terminal Real seal consumes it.

Definition 7198.1 (Constructive Cauchy tail-operator carrier)**.** *A* ConstructiveCauchyTailOperator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, k, W, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* StreamName$^{\uparrow}$ *row naming the rational stream whose tail is being read;*
2. *k is the unary tail threshold row, exposed as finite data rather than as a hidden natural-number choice;*
3. *W is the shifted stream-window row, recording the finite read of S after threshold k;*
4. *B is the* TailCofinalityBudget$^{\uparrow}$ *row certifying that the requested tail is cofinal for the displayed Cauchy use;*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row over the shifted window;*
6. *E is the* DyadicRatCore$^{\uparrow}$ *error ledger used by the regularity estimate;*
7. *H is the componentwise* hsame *transport row for replacing S by an extensionally same stream window;*
8. *C is the displayed* Cont *replay route through S, k, W, B, R, E, H;*
9. *P is the* Pkg *provenance over* ConstructiveCauchyTailOperator$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, FiniteTailFilter$^{\uparrow}$, SelectedTailSeed$^{\uparrow}$, TailCofinalityBudget$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, countable choice, an ambient completeness theorem, a selected limit outside the packet, or host-side equality of real numbers.

Theorem 7198.2 (Constructive Cauchy tail-operator NameCert obligations)**.** *For the carrier of Definition 7198.1, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}}$ *surface consists of carrier admission for $T = (S, k, W, B, R, E, H, C, P, N)$, source stream exposure through S, finite threshold exposure through k, shifted window exposure through W, cofinal budget exposure through B, regular-rational readback through R, dyadic error exposure through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier row of Definition 7198.1. The admitted object is exactly the ten-row packet $T = (S, k, W, B, R, E, H, C, P, N)$.

The source and shift rows expose the operator input. The stream row S, threshold row k, and shifted window row W are the only data from which a consumer may read the tail.

The Cauchy rows expose the constructive estimate. The cofinal budget B, regular-rational readback R, and dyadic error ledger E are displayed before any Real-facing handoff can be made.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier admits no other coordinates, these rows exhaust the local NameCert obligations. □

Theorem 7198.3 (Constructive Cauchy tail-operator shift stability)**.** *Let $T = (S, k, W, B, R, E, H, C, P, N)$ be an accepted* $\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}$ *packet. If a later finite threshold k' refines k through the same* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *route, then the shifted-window readback can be repacked as another accepted* $\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}$ *packet with the same source stream, dyadic ledger discipline, componentwise transport surface, replay route, provenance, and local naming format.*

Proof. Begin with the finite threshold rows. A refinement of k through the same cofinal budget row B gives a later finite tail read of the same source stream S.

The shifted window is then rebuilt by reading S after k'. The regular-rational readback row R and dyadic error ledger E are reused at the later threshold because the budget row records cofinality rather than a single isolated prefix.

The remaining rows are structural. Componentwise hsame transport still acts on the displayed stream window, the Cont route replays the same operator shape, and the Pkg and local $\mathsf{NameCert}$ rows name the repacked packet. Therefore the tail operator is stable under displayed tail refinement. □

Theorem 7198.4 (Constructive Cauchy tail-operator Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{FiniteTailFilter}^{\uparrow} \longrightarrow \mathsf{TailCofinalityBudget}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only the shifted window, cofinal budget, regular-rational readback, dyadic error ledger, componentwise transport, replay, provenance, and local NameCert rows; it does not export quotient streams, host real equality, countable choice, detached limits, or ambient completeness.

Proof. Read the operator packet through its displayed rows. The only route from the source stream to a Real-facing seal passes through the finite threshold row, shifted window row, cofinal budget row, regular-rational readback row, and dyadic error ledger.

The terminal consumer sees the Real seal through those rows, not through an external equality principle. The componentwise transport and replay rows show how the same displayed data is carried along the handoff.

Because the carrier has no quotient, choice, detached-limit, host-equality, or ambient-completeness coordinate, no such datum can be exported by the handoff. The Real-facing read is therefore exactly the displayed finite route. □

Falsifiable prediction: No accepted ConstructiveCauchyTailOperatorUp packet exports quotient stream equality, host real equality, countable choice, a detached limit, or ambient completeness outside the displayed finite tail-operator rows.

Independence witness: streamname, regseqrat, real, dyadicratcore, finite-tail-filter, selected-tail-seed, tail-cofinality-budget: none is bijective to ConstructiveCauchyTailOperatorUp because the operator packet combines a source stream, finite threshold, shifted window, cofinal budget, regular-rational readback, dyadic error ledger, componentwise transport, replay, provenance, and local NameCert row in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}$ packet is generated from a source stream, unary tail threshold, shifted stream window, cofinal budget, regular-rational readback, dyadic error ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7198.1, Definition 7198.2, Definition 7198.3, and Definition 7198.4 give the seed finite tail-operator carrier and its displayed Real-seal handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyTailOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, host real equality, countable choice, detached limits, or ambient completeness.

Upgrade path: Scoped closure requires Lean-backed NameCert obligations, shift stability, and Real-seal handoff evidence for the displayed tail-operator rows.

Chapter 7199

A Concrete Naming Certificate for $\mathsf{HyperbolicConvexDisk}^{\uparrow}$

The $\mathsf{HyperbolicConvexDisk}^{\uparrow}$ packet realizes the finite geodesic-window convexity route suggested by chapter 9463 as standard hyperbolic convexity inside a displayed Poincare disk surface. It sits over $\mathsf{PoincareDiskMetric}^{\uparrow}$ of chapter 6209, $\mathsf{HyperbolicMetric}^{\uparrow}$ of chapter 3474, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ of chapter 3491, $\mathsf{Real}^{\uparrow}$ of chapter 38, and $\mathsf{StreamName}^{\uparrow}$ of chapter 126. These siblings expose disk metric reads, hyperbolic length, boundary endpoints, Real sealing, and finite stream windows; this chapter packages the finite convexity certificate consumed by geodesic and observer-phase routes.

Definition 7199.1 (Hyperbolic convex disk carrier)**.** *A* $\mathsf{HyperbolicConvexDisk}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, M, G, B, W, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite Poincare-disk window row naming the disk points under inspection;*
2. *M is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *metric-read row on U;*
3. *G is the finite geodesic-window row, listing the displayed geodesic segments whose endpoints lie in U;*
4. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *endpoint and boundary-transport row for the same windows;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *sampling row used to enumerate the inspected endpoints and interior witnesses;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite metric and parameter reads;*
7. *H is the componentwise* hsame *transport row for replacing the displayed disk window by an extensionally same one;*
8. *C is the displayed* Cont *replay route through U, M, G, B, W, R, H;*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicConvexDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicConvexDisk}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary host subset of the disk, an infinite convex hull, quotient boundary equality, hidden geodesic existence, or an ambient completeness principle.

Theorem 7199.2 (Hyperbolic convex disk NameCert obligations)**.** *For the carrier of Definition 7199.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicConvexDisk}^{\uparrow}}$ *surface consists of carrier admission for* $D = (U, M, G, B, W, R, H, C, P, N)$*, disk-window exposure through* U*, metric-read exposure through* M*, finite geodesic-window exposure through* G*, boundary endpoint exposure through* B*, stream sampling through* W*, Real sealing through* R*, componentwise transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7199.1. The admitted object is exactly the ten-row finite packet $D = (U, M, G, B, W, R, H, C, P, N)$.

The geometric rows expose the convexity test. The disk window U, metric row M, geodesic-window row G, and boundary row B are the only rows from which a consumer can read hyperbolic convexity.

The sampling and seal rows make the finite measurement explicit. The stream row W enumerates the inspected endpoints and witnesses, while R records the Real-facing metric and parameter seal.

Finally H, C, P, N transport, replay, source, and name the displayed rows. Since no host subset, infinite hull, hidden geodesic, quotient boundary equality, or ambient completeness coordinate is admitted, these rows exhaust the local NameCert obligations. □

Theorem 7199.3 (Hyperbolic convex disk geodesic stability)**.** *Let* $D = (U, M, G, B, W, R, H, C, P, N)$ *be an accepted* $\mathsf{HyperbolicConvexDisk}^{\uparrow}$ *packet. If a displayed geodesic segment has endpoints in the finite disk window and its sampled interior witnesses are carried by the same stream and Real-seal rows, then replacing the disk window by an extensionally same window preserves the accepted convex disk packet through the same boundary, transport, replay, provenance, and local naming rows.*

Proof. Begin with the geodesic-window row G. Its endpoints are recorded inside the finite disk window U, and its metric read is governed by M.

The interior witness read is finite. The stream row W lists the sampled witnesses, while the Real seal R records the metric and parameter values needed by the displayed segment check.

Now apply the componentwise transport row H to replace U by an extensionally same disk window. The boundary row B transports the endpoint read, the replay row C repeats the same finite route, and the provenance and NameCert rows keep the packet identity fixed. Therefore the convex disk certificate is stable under displayed geodesic-window transport. □

Falsifiable prediction: No accepted HyperbolicConvexDiskUp packet exports an arbitrary host disk subset, infinite convex hull, quotient boundary equality, hidden geodesic existence, or ambient completeness principle outside the displayed finite geodesic-window rows.

Independence witness: poincare-disk-metric, hyperbolic-metric, hyperbolic-geodesic-boundary, real, streamname: none is bijective to HyperbolicConvexDiskUp because the convex disk packet combines disk-window rows, metric reads, finite geodesic windows, boundary endpoints, stream sampling, Real sealing, transport, replay, provenance, and local NameCert rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicConvexDisk}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicConvexDisk}^{\uparrow}$ packet is generated from a finite Poincare-disk window, metric read, geodesic-window list, boundary endpoint row, stream sampling row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicConvexDisk}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7199.1, Definition 7199.2, and Definition 7199.3 give the seed finite convex disk carrier and its geodesic-window stability route.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicConvexDisk}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary host disk subsets, infinite convex hulls, quotient boundary equality, hidden geodesic existence, or ambient completeness.
Upgrade path: Scoped closure requires Lean-backed NameCert obligations and geodesic-window stability evidence for the displayed convex disk rows.

Chapter 7200

A Concrete Naming Certificate for $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$ packet records the finite compact-interval modulus route used by Real-and-Completion consumers. It sits between the closed bounded interval source of chapter 1591, the completion-modulus handoff of chapter 1469, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic threshold ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the displayed finite cover and modulus rows; it does not assert open-cover compactness, choose a subcover from an ambient family, quotient real names, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7200.1 (Bishop compact interval modulus carrier)**.** *A* $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, M, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including lower endpoint, upper endpoint, endpoint-order, containment, and terminal interval seal data.*
2. *F is the dyadic finite-cover row over I, listing the inspected dyadic subintervals and their finite containment evidence.*
3. *M is the compact interval modulus row, assigning each displayed precision request a finite cover depth and a dyadic threshold.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the endpoint names, finite cover cells, and modulus request.*
5. *W is the StreamName window selected by M at the requested depth.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger containing dyadic scale, endpoint comparison, finite-cover containment, and tolerance rows for W.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected window and dyadic ledger.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, F, M, S, W, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the interval source, finite cover, modulus, window rows, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the interval source through the finite cover and modulus row to the selected StreamName window, then through the dyadic ledger and regular readback to the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BishopCompletionModulu $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for an ambient open cover, selected subcover, host uniform-continuity theorem, quotient real equality, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7200.2 (Bishop compact interval modulus obligations)**.** *For every accepted* BishopCompactIntervalModul *carrier $K = (I, F, M, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; closed bounded interval source exposure through I; dyadic finite-cover exposure through F; compact interval modulus readback through M; StreamName window selection through S, W; DyadicRatCore threshold compatibility through D; RegSeqRat readback through R; RealUp seal non-escape through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7200.1. The rows $I, F, M, S, W, D, R, E, H, C, P, N$ are the full public packet, so a consumer cannot introduce an unlisted open cover, selected subcover, quotient real representative, or completeness witness.

Read the analytic rows in order. The closed bounded interval source I comes first, the finite dyadic cover F is attached to that same source, and the modulus row M selects the cover depth and threshold used for the requested precision.

The completion-facing rows are downstream. The selected window W is read through S, the dyadic ledger D records the same threshold and cover scale, and the regular readback R is available before the seal E.

The structural rows close the NameCert surface. Componentwise transport, replay, provenance, and local naming preserve exactly the displayed route, so the listed obligations exhaust the carrier. □

Theorem 7200.3 (Bishop compact interval modulus completion handoff)**.** *Every accepted* BishopCompactIntervalM *carrier hands a compact interval modulus to Real-and-Completion consumers through the finite route*

$$I, F, M \longmapsto S, W, D \longmapsto R \longmapsto E.$$

The handoff first exposes the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source and dyadic finite cover, then the compact interval modulus row, then the StreamName window, DyadicRatCore threshold ledger, RegSeqRat readback, and RealUp seal. It exports no ambient open-cover compactness, selected subcover, quotient real equality, countable-choice row, or Real completeness principle.*

Proof. Begin with the source and cover rows. The accepted packet displays I and F before the modulus row can be consumed, so the interval and finite dyadic cover are fixed by carrier data rather than by an ambient compactness theorem.

Next read the modulus and window rows. The row M selects the requested finite cover depth and threshold, S names the finite observation schedule, and W is the selected window attached to that request.

Finally read the completion-facing rows. The dyadic ledger D keeps the threshold and cover scale visible, R is the regular rational readback obtained from the displayed window, and E is downstream from that readback. Since the carrier has no coordinate for the refused principles, the handoff exports only the finite route above. □

Falsifiable prediction: An accepted $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$ read fails if a Real-and-Completion consumer reaches a compact interval modulus without displaying the ClosedBoundedInterval source, dyadic finite cover, StreamName window, RegSeqRat readback, DyadicRatCore threshold ledger, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: closedboundedinterval, bishopcompletionmodulus, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$ because this packet binds the closed interval source, finite dyadic cover, compact interval modulus row, completion window, threshold ledger, regular readback, Real seal, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactIntervalModulus}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source, a dyadic finite cover, a compact interval modulus row, StreamName windows, a DyadicRatCore threshold ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactIntervalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7200.1, Definition 7200.2, and Definition 7200.3 seed the compact-interval modulus route through the closed interval source, finite cover, completion window, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactIntervalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient Real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover exposure, compact modulus readback, StreamName and DyadicRatCore threshold compatibility, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactIntervalModulusUp.

Chapter 7201

A Concrete Naming Certificate for TailWindowModulusSynchronization$^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences, the TailWindowModulusSynchronization$^{\uparrow}$ packet realizes the regular-Cauchy route of section 9238.1 as a finite synchronization certificate. It sits after the StreamName window surface of chapter 126, the RegSeqRat readback of chapter 301, the DyadicRatCore threshold ledger of chapter 291, the terminal RealUp seal of chapter 38, and the shared tail threshold discipline of chapter 1231. The packet names the finite synchronization between a tail-window request and a displayed modulus row; it does not choose an infinite subsequence, quotient regular Cauchy names, select a limit, invoke countable choice, or assume ambient Real completeness.

Definition 7201.1 (Tail-window modulus synchronization carrier)**.** *A* TailWindowModulusSynchronization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, B, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the displayed modulus row assigning a unary precision request to a tail threshold.*
2. *B is the* TailCofinalityBudget$^{\uparrow}$ *row recording that later tail requests refine the same finite threshold route.*
3. *S is the* StreamName$^{\uparrow}$ *schedule for the regular Cauchy family being inspected.*
4. *W is the finite tail window selected by M and admitted by B.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *ledger containing the tolerance, threshold comparison, and synchronization scale for W.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the synchronized window and dyadic ledger.*
7. *E is the* Real$^{\uparrow}$ *seal row reached only after the modulus, cofinality, window, dyadic, and regular readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the modulus, cofinality budget, stream schedule, selected window, dyadic ledger, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M, B, S, W, D \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* TailWindowModulusSynchronization$^{\uparrow}$, TailCofinalityBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{TailWindowModulusSynchronization}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, B, S, W, D, R, E, H, C, P, N$. It has no coordinate for host sequence equality, quotient real equality, selected limits, countable choice, or ambient completeness of Real$^{\uparrow}$.

Theorem 7201.2 (Tail-window modulus synchronization obligations)**.** *For every accepted* TailWindowModulusSync *carrier $T = (M, B, S, W, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{TailWindowModulusSynchronization}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; modulus threshold exposure through M; cofinal tail-budget exposure through B; StreamName window selection through S, W; DyadicRatCore synchronization through D; RegSeqRat readback through R; RealUp seal non-escape through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier. The rows $M, B, S, W, D, R, E, H, C, P, N$ are the only classifier-visible coordinates, so no host sequence equality, quotient real equality, selected limit, choice schedule, or completeness principle can be read from the packet.

Read the synchronization rows in order. The modulus row M chooses the threshold for the displayed precision request, and the tail cofinality budget B records how later requests remain on the same finite route.

The window and readback rows then consume that shared threshold. The StreamName schedule S supplies W, the dyadic ledger D records the synchronized tolerance comparison, and the regular readback R is downstream from those rows.

Finally the Real seal and structural rows preserve the route. The row E is available only after R, while H, C, P, N transport, replay, source, and name the same finite packet. □

Theorem 7201.3 (Tail-window modulus synchronization handoff)**.** *Every Real-completion consumer admitted by an accepted* TailWindowModulusSynchronization$^{\uparrow}$ *packet reads a single finite synchronization route*

$$M, B \longmapsto S, W, D \longmapsto R \longmapsto E.$$

The route synchronizes the modulus threshold with the tail-window budget before any RegSeqRat readback or RealUp seal is available. It exports no quotient regular Cauchy equality, hidden subsequence selector, selected limit, countable-choice row, or ambient Real completeness principle.

Proof. Start from the regular Cauchy vision site section 9238.1. A completion-facing read asks for a finite tail window at a displayed precision, so the packet first exposes the modulus row M and the cofinal budget row B.

Next the window is synchronized with those rows. The schedule S names the finite observations, W is the selected tail window, and D records the dyadic threshold comparison that keeps the modulus and budget aligned.

The readback and seal are downstream. The row R is obtained from the displayed window and ledger, and the terminal row E is available only after R. Since the carrier contains no refused coordinate, the consumer receives exactly the finite synchronization route. □

Falsifiable prediction: An accepted TailWindowModulusSynchronization$^{\uparrow}$ read fails if a Real-completion consumer reaches RegSeqRat readback or a RealUp seal before displaying the modulus row, TailCofinalityBudget row, StreamName window, DyadicRatCore synchronization ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: tail-cofinality-budget, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{TailWindowModulusSynchronization}^{\uparrow}$ because this packet binds a modulus threshold and a cofinal tail budget to one synchronized StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{TailWindowModulusSynchronization}^{\uparrow}}$.

Constructive story: A $\mathsf{TailWindowModulusSynchronization}^{\uparrow}$ packet is generated from a modulus threshold row, a TailCofinalityBudget row, StreamName windows, a DyadicRatCore synchronization ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailWindowModulusSynchronization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7201.1, Definition 7201.2, and Definition 7201.3 seed the finite tail-window modulus synchronization route for regular-Cauchy Real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailWindowModulusSynchronization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient regular Cauchy equality, hidden subsequence selection, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus threshold exposure, tail cofinality exposure, StreamName window selection, DyadicRatCore synchronization, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.TailWindowModulusSynchronizationUp.

Chapter 7202

A Concrete Naming Certificate for $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ packet records the finite modulus data used to answer an absolute-continuity request on a displayed located real interval. It sits after the real-name seal of chapter 38, the dyadic arithmetic core of chapter 291, the regular rational readback route of chapter 301, and the located interval surface of chapter 1709. The packet gives later integration and uniform-continuity consumers a choice-free finite budget for controlled variation; it does not import Lebesgue measure, measurable sets, null-set completion, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7202.1 (Absolute-continuity modulus carrier)**.** *An* $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E, D, P, R, H, C, G, N)$$

with the following displayed rows:

1. *E is the epsilon request row, naming the positive* $\mathsf{Real}^{\uparrow}$ *tolerance asked of the controlled-variation read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *delta-budget row, recording the dyadic size bound offered in response to E.*
3. *P is the finite partition ledger over a displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source, listing the disjoint subinterval reads whose total dyadic length is compared with D.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the epsilon request, dyadic delta budget, and partition ledger have been displayed.*
5. *H records componentwise* hsame *transport for E, D, P, R and for the replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of every accepted modulus request through*

$$E, D, P, R \longmapsto H, C, G, N.$$

7. *G is the* Pkg *provenance over* $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. N is the local $\mathsf{NameCert}_{\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for Lebesgue measure, measurable-set completion, almost-everywhere equality, quotient function equality, selected covers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7202.2 (Absolute-continuity modulus NameCert obligations)**.** *For every accepted* AbsoluteContinuityMod *carrier*

$$A = (E, D, P, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}}$ *surface is exhausted by epsilon-request exposure through E, dyadic delta-budget exposure through D, finite partition-ledger exposure through P,* $\mathsf{Real}^{\uparrow}$ *seal exposure through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N. Every variation read must pass through the displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows before it may consume the terminal Real seal.*

Proof. Project the carrier of Definition 7202.1. The epsilon request, dyadic budget, finite partition ledger, and terminal Real seal are the only analytic coordinates available to a consumer.

The partition row is finite. Its interval reads are attached to the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ source and its dyadic length comparisons are read through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the regular-rational handoff is replayed.

Finally H, C, G, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for measure completion, almost-everywhere equality, selected covers, countable choice, or ambient Real completeness, the listed rows exhaust the local certificate surface. □

Theorem 7202.3 (Absolute-continuity modulus partition stability)**.** *Let $A = (E, D, P, R, H, C, G, N)$ be an accepted* $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ *carrier. Replacing P by a displayed refinement whose subinterval ledger has the same* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row and whose dyadic total remains bounded by D yields an accepted partition read with the same epsilon request E, Real seal R, transport row H, replay row C, provenance row G, and local name N.*

Proof. Begin with the finite partition ledger. A displayed refinement only subdivides rows already named by P, so every new subinterval still reads from the same located real interval source.

The dyadic comparison is preserved by the budget row D. The refined ledger is accepted only when its displayed total remains below the same delta budget attached to E.

The structural rows then replay the same route. Componentwise hsame transport keeps E, D, R, G, N fixed, and the Cont row replays the refined partition ledger through the same budgeted Real-facing surface. □

Theorem 7202.4 (Absolute-continuity modulus RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ *packet reaches the terminal seal only through the route*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports a finite epsilon–delta partition budget and not a Lebesgue-measure theorem, a null-set completion, a selected cover, quotient function equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Read the carrier in order. The epsilon request E is answered by the dyadic budget D, and the finite partition ledger P is checked against that same budget before any terminal seal is available.

The regular-rational readback is the only completion-facing bridge in the packet. It carries the dyadic ledger to the terminal $\mathsf{Real}^{\uparrow}$ row R without adding measure-theoretic or host-side data.

Thus R is downstream of E, D, P, H, C, G, N and of the displayed sibling rows. A consumer that omits this route is not reading the accepted $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ consumer that reads a terminal Real seal while omitting the epsilon request, dyadic delta budget, finite partition ledger, hsame transport, Cont replay, Pkg provenance, or local NameCert row has left the packet.

Independence witness: real, dyadicratcore, regseqrat, locatedrealinterval: $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ is not bijective with any listed sibling because it joins an epsilon request, a dyadic delta budget, a finite partition ledger, a terminal Real seal, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}}$.

Constructive story: An $\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}$ packet is generated from an epsilon request, a DyadicRatCore delta budget, a finite partition ledger over a LocatedRealInterval source, a RegSeqRat readback route, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7202.1, Definition 7202.2, Definition 7202.3, and Definition 7202.4 seed the finite epsilon–delta partition-modulus surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuityModulusReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lebesgue measure, measurable-set completion, almost-everywhere equality, quotient function equality, selected covers, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, epsilon-request exposure, dyadic delta-budget exactness, partition-ledger stability, RealUp handoff, structural nonescape, and route replay for BEDC.Derived.AbsoluteContinuityModulusReal

Chapter 7203

A Concrete Naming Certificate for $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ packet records the finite enclosure data used to pass from dyadic interval evidence to a regular rational readback and then to a terminal real-name seal. It sits after the dyadic arithmetic core of chapter 291, the dyadic number surface of chapter 1658, the finite stream-window route of chapter 126, the regular rational sequence route of chapter 301, and the real-name seal of chapter 38. The packet gives completion consumers a non-quotient interval interface; it does not assert arbitrary Cauchy completeness, selected limits, trichotomy, host real equality, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7203.1 (Dyadic interval enclosure carrier)**.** *A* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, N, S, R, E, H, C, P, A)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
2. *U is the upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
3. *N is the nesting and order ledger, recording $L \leq U$ and the finite refinement relation between successive enclosure rows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing exactly the finite interval windows consumed by the readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed dyadic endpoints and stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, N, S, R have been displayed.*
7. *H records componentwise* hsame *transport for all analytic rows and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of every accepted enclosure read through*

$$L, U, N, S, R, E \longmapsto H, C, P, A.$$

9. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Dyadic}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. A is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalEnclosure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a quotient of Cauchy streams, a selected real representative, trichotomy, host real equality, countable choice, or an ambient completeness principle.

Theorem 7203.2 (Dyadic interval enclosure NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ *carrier*

$$I = (L, U, N, S, R, E, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalEnclosure}^{\uparrow}}$ *surface is exhausted by lower endpoint exposure through L, upper endpoint exposure through U, nesting and order exposure through N, finite window exposure through S, regular rational readback through R, terminal real seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through A. Every completion-facing read must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Dyadic}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows before it may consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the carrier of Definition 7203.1. The only endpoint data available to a consumer are L and U, and both are dyadic-core rows. The only order and refinement evidence is the finite ledger N.

The readback path is then fixed. The finite stream window S exposes the displayed enclosure windows, R reads them as regular rational data, and E is downstream of that readback.

Finally H, C, P, A transport, replay, source, and name the same finite rows. Since the carrier has no coordinate for quotient stream equality, selected representatives, trichotomy, host equality, countable choice, or ambient Real completeness, the listed rows exhaust the local certificate surface. □

Theorem 7203.3 (Dyadic interval enclosure nesting stability)**.** *Let $I = (L, U, N, S, R, E, H, C, P, A)$ be an accepted* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ *carrier. Replacing the displayed window row S by a finite subwindow whose endpoint rows are transported by H and whose refinement relation is recorded by N yields an accepted enclosure read with the same readback row R, terminal seal E, provenance row P, and local name A.*

Proof. Begin with the nesting ledger. A finite subwindow is accepted only when N records its lower and upper endpoints as refinements of the displayed dyadic rows.

The transport row H keeps the endpoint identity visible while the window changes. Thus the refined lower and upper rows still read through the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint surface.

The downstream rows are structural. The Cont replay C sends the accepted subwindow through the same regular rational readback R, and the terminal seal E, provenance row P, and local name A remain attached to that replay. □

Theorem 7203.4 (Dyadic interval enclosure Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ *packet reaches the terminal seal only through the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Dyadic}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff exports a finite nested dyadic interval interface and not a quotient stream, a selected limit, host real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Read the carrier in order. The dyadic endpoint rows L and U are exposed before the nesting ledger N can certify the interval relation.

The stream row S and regular rational row R are the only readback bridge to the terminal real seal. They carry finite dyadic window evidence to E without adding quotient or host-side equality data.

Thus any accepted terminal seal read is attached to the displayed route through the dyadic core, dyadic surface, stream window, regular rational readback, and local structural rows. A consumer that omits this route is not reading the accepted $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ carrier. □

Theorem 7203.5 (Dyadic interval enclosure classifier stability)**.** *Two accepted* $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ *packets are classifier-equivalent whenever their lower endpoint rows, upper endpoint rows, nesting ledgers, stream windows, regular rational readbacks, terminal real seals, transport rows, replay rows, provenance rows, and local NameCert rows are pairwise* hsame*. No extra classifier coordinate may distinguish them by a host representative, a selected limit, a quotient stream class, trichotomy data, countable choice data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The classifier of Definition 7203.1 reads only the ten displayed rows. Pairwise hsame therefore identifies the endpoint surface, nesting ledger, finite window, readback, terminal seal, transport, replay, provenance, and local naming data.

There is no hidden coordinate left to inspect. The excluded host representative, selected limit, quotient stream class, trichotomy, countable-choice, and ambient-completeness data are not fields of the carrier.

Consequently the two packets have the same local classifier read precisely when the listed finite rows agree componentwise. □

Falsifiable prediction: Every accepted $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ consumer that reads a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a lower endpoint row, upper endpoint row, nesting ledger, StreamName window, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, or local NameCert row has left the packet.

Independence witness: dyadicratcore, dyadic, streamname, regseqrat, real: $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ is not bijective with any listed sibling because it joins dyadic endpoint rows, a finite nesting ledger, stream windows, regular rational readback, a terminal real seal, transport, replay, provenance, and local naming as one packet.

Lean4 (statement) `BEDC.Derived.DyadicIntervalEnclosureUp`

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalEnclosure}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalEnclosure}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, a finite nesting and order ledger, a StreamName window row, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalEnclosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7203.1, Definition 7203.2, Definition 7203.3, Definition 7203.4, and Definition 7203.5 seed the finite dyadic interval enclosure surface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalEnclosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary Cauchy completeness, selected limits, quotient stream equality, trichotomy, host real equality, countable choice, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, nesting stability, regular rational readback, RealUp handoff, classifier stability, structural nonescape, and route replay for BEDC.Derived.DyadicIntervalEnclosureUp.

Chapter 7204

A Concrete Naming Certificate for $\mathsf{DenseGDelta}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{DenseGDelta}^{\uparrow}$ packet records the standard dense G_δ surface as finite BEDC NameCert data. It sits between the countable open-row schedule of chapter 3547, the residual evidence carrier of chapter 7188, the dense-open Baire route of chapter 1542, and the complete-metric inspection surface of chapter 3535. The packet names the positive dense-open face of residual evidence; it does not choose a generic point, assert a host intersection, import countable choice, or identify quotient subsets.

Definition 7204.1 (DenseGDelta carrier)**.** *A* $\mathsf{DenseGDelta}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, G, S, O, B, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the ambient topology or complete-metric source row over which density is read.*
2. *G is the* $\mathsf{GDeltaSet}^{\uparrow}$ *countable open-row schedule, displayed as finite prefix data.*
3. *S is the dense-open schedule row, read through the Baire-category surface before any residual consumer receives the packet.*
4. *O records the open-row readback and local density witnesses for the schedule positions visible through G and S.*
5. *B is the Baire route row linking the dense-open schedule to the inspected complete-metric Baire surface.*
6. *R is the* $\mathsf{ResidualSet}^{\uparrow}$ *handoff row exposed only after the positive dense G_δ side has been displayed.*
7. *H records componentwise* hsame *transport for $T, G, S, O, B, R, C, P, N$.*
8. *C records displayed* Cont *replay of the schedule, density, Baire, residual, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DenseGDelta}^{\uparrow}$, $\mathsf{GDeltaSet}^{\uparrow}$, $\mathsf{ResidualSet}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurfa}$ BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseGDelta}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected generic point, a host-level countable intersection, unrestricted set equality, quotient subset equality, or countable-choice selector outside the displayed BEDC rows.

Theorem 7204.2 (DenseGDelta NameCert obligations)**.** *For every accepted* $\mathsf{DenseGDelta}^{\uparrow}$ *carrier* $D = (T, G, S, O, B, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DenseGDelta}^{\uparrow}}$ *surface consists exactly of ambient source admission through* T*, countable open-schedule exposure through* G*, dense-open schedule exposure through* S*, open-row and density readback through* O*, Baire route handoff through* B*, residual handoff through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7204.1. The accepted packet displays exactly the ambient source row, countable open-row schedule, dense-open schedule, open-row readback, Baire route, residual handoff, transport, replay, provenance, and local name.

The positive topological content is exhausted by G, S, O, B. The G row supplies the G_δ schedule, the S row supplies the dense-open requests, O records the local readback and density witnesses, and B routes the read through the Baire surface.

The residual side is only a downstream handoff. The row R may be consumed after the positive dense G_δ evidence has been displayed, while H, C, P, N transport, replay, source, and name the same finite route.

No coordinate stores a chosen generic point, a host intersection object, countable-choice data, or quotient subset equality. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 7204.3 (DenseGDelta residual handoff)**.** *Every residual consumer of an accepted* $\mathsf{DenseGDelta}^{\uparrow}$ *packet factors through chapter 3547, chapter 1542, and chapter 3535 before it reads the* $\mathsf{ResidualSet}^{\uparrow}$ *row. Thus the residual route receives a positive dense-open* G_δ *certificate, not a hidden host intersection or selected generic point.*

Proof. Project $D = (T, G, S, O, B, R, H, C, P, N)$. A residual consumer cannot read R until the preceding rows expose the countable open schedule, dense-open schedule, local density readback, and Baire route.

The G row routes through the $\mathsf{GDeltaSet}^{\uparrow}$ certificate, while S and O display the dense-open and local readback data. The Baire row B then passes through the complete-metric Baire surface before the residual handoff is available.

Since the carrier contains no host intersection coordinate, no selected generic point, no countable-choice selector, and no quotient subset equality, the residual consumer receives only the displayed positive dense G_δ route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DenseGDelta}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DenseGDelta}^{\uparrow}}$, seed)

Scope closed: Definition 7204.1, Definition 7204.2, and Definition 7204.3 seed the positive dense G_δ route over GDeltaSet, BaireCategory, CompleteMetricBaireSurface, ResidualSet, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DenseGDelta}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close generic-point choice, host intersections, countable choice, quotient subset equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exactness, GDeltaSet exposure, residual handoff, Baire routing, transport, replay, provenance, local naming, and refusal of hidden host intersections for BEDC.Derived.DenseGDeltaUp.

Constructive story: A $\mathsf{DenseGDelta}^{\uparrow}$ packet is generated from an ambient source row, a $\mathsf{GDeltaSet}^{\uparrow}$ schedule, a dense-open schedule, local open-row and density readback, a Baire route, a $\mathsf{ResidualSet}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7205

A Concrete Naming Certificate for DyadicRoundingProjection$^{\uparrow}$

Following Bishop and Bridges, the DyadicRoundingProjection$^{\uparrow}$ packet records the finite floor-and-ceiling projection surface by which a dyadic approximation request is bracketed before any RegSeqRat$^{\uparrow}$ readback or terminal Real$^{\uparrow}$ seal is consumed. It sits over the arithmetic carrier of chapter 291, the interval endpoint ledger of chapter 306, the finite window source of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names only a finite rounding projection route; it does not assert quotient real equality, selected limits, omniscient rounding, or ambient completeness of Real$^{\uparrow}$.

Definition 7205.1 (Dyadic rounding projection carrier)**.** *A* DyadicRoundingProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, I, S, R, E, F, G, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *arithmetic and comparison row for the inspected dyadic precision.*
2. *I is the* DyadicInterval$^{\uparrow}$ *endpoint row containing the lower and upper dyadic cells visible at that precision.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row whose inspected values are rounded.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row that receives the rounded dyadic bracket.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row read only after D, I, S, R have been displayed.*
6. *F is the floor projection row selecting the greatest displayed dyadic cell not above the inspected window value.*
7. *G is the ceiling projection row selecting the least displayed dyadic cell not below the inspected window value.*
8. *H is the componentwise* hsame *transport row for $D, I, S, R, E, F, G, C, P, N$.*
9. *C is the displayed* Cont *replay row forcing every downstream consumer through the same floor, ceiling, interval, stream, readback, seal, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* DyadicRoundingProjection$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRoundingProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an exact host real value, a selected limit, a quotient real representative, countable choice, or ambient completeness.

Theorem 7205.2 (Dyadic rounding projection NameCert obligations)**.** *For every accepted* DyadicRoundingProjectio *carrier $Q = (D, I, S, R, E, F, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicRoundingProjection}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the dyadic arithmetic row, dyadic interval row, stream window row, regular rational readback row, real seal row, floor row, ceiling row, transport row, replay row, provenance row, and local name;*
2. *floor exactness, requiring F to expose a displayed dyadic endpoint from I whose comparison is certified by D and whose inspected value is read from S;*
3. *ceiling exactness, requiring G to expose a displayed dyadic endpoint from I whose comparison is certified by D and whose inspected value is read from S;*
4. *bracket replay, requiring the ordered pair (F, G) to be consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *through R before the* $\mathsf{Real}^{\uparrow}$ *seal row E is visible;*
5. *classifier transport and non-escape, requiring componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and local naming while excluding host real values, selected limits, quotient representatives, countable choice, and ambient completeness.*

No rounding obligation is stored outside this finite carrier.

Proof. Project the carrier of Definition 7205.1. The arithmetic, interval, stream, readback, seal, floor, ceiling, transport, replay, provenance, and local naming rows are exactly $D, I, S, R, E, F, G, H, C, P, N$.

The floor and ceiling rows are read through the same dyadic interval row I and the same arithmetic comparison row D. The finite window row S supplies the inspected values, while F and G choose only displayed dyadic cells.

The bracket replay row then routes (F, G) into the regular rational readback R and only afterward into the real seal E. Transport through H and replay through C keep the classifier attached to the same displayed rows.

Since the packet has no coordinate for exact host real values, selected limits, quotient representatives, countable choice, or ambient completeness, the five listed rows exhaust the local certificate. □

Theorem 7205.3 (Dyadic rounding projection floor-ceiling bracket)**.** *Every accepted* $\mathsf{DyadicRoundingProjection}^{\uparrow}$ *carrier exposes a finite bracket in which the floor row F and ceiling row G are dyadic endpoints from the displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *row I, both comparisons are witnessed by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, and both endpoints are tied to the same* $\mathsf{StreamName}^{\uparrow}$ *window row S. A downstream consumer cannot replace the bracket by a host real comparison, selected representative, quotient equality, or ambient completeness principle.*

Proof. Use the obligation surface of Definition 7205.2. The floor row and ceiling row are admitted only as cells in the same displayed interval row I.

The comparison evidence is supplied by D, and the inspected source value is supplied by S. Thus the two endpoints form one finite bracket over the same dyadic precision and the same stream window.

The non-escape row of the certificate rules out a second bracket supplied by host real comparison, selected representatives, quotient equality, or ambient completeness. □

Theorem 7205.4 (Dyadic rounding projection RegSeqRat readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicRoundingProjection}^{\uparrow}$ *packet reads the rounded approximation through the ordered route*

$$(D, I, S) \longmapsto (F, G) \longmapsto R \longmapsto E$$

with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row attached. The route exports a finite dyadic bracket for regular rational readback and does not export a selected completed real, quotient equality of real names, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer read. By Definition 7205.2, the read must first expose the dyadic arithmetic row, interval row, and stream window row.

Next apply the floor-ceiling bracket theorem. It identifies F and G as the only rounded endpoints available to the consumer.

Finally replay the bracket through R and then through E. The transport, replay, provenance, and local naming rows H, C, P, N keep the route finite, and the carrier has no coordinate for selected completed reals, quotient equality, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRoundingProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRoundingProjection}^{\uparrow}$ packet is generated from dyadic arithmetic rows, dyadic interval endpoints, stream windows, floor and ceiling projection rows, regular rational readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRoundingProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7205.1, Definition 7205.2, Definition 7205.3, and Definition 7205.4 seed the finite floor-ceiling rounding projection surface for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRoundingProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, selected real limits, quotient equality of real names, countable choice, omniscient rounding, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, floor exactness, ceiling exactness, bracket replay, classifier transport, and non-escape for BEDC.Derived.DyadicRoundingProjectionUp.

Chapter 7206

A Concrete Naming Certificate for $\mathsf{BishopRealRegularization}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealRegularization}^{\uparrow}$ packet records the terminal real-name regularization route for the L10 Real and Completion corridor. It sits after the raw-to-regular service of chapter 4100, the sequential tightening surface of chapter 3615, the located regularization boundary of chapter 1935, the finite window service of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed regularization certificate from raw Cauchy data to a sealed Real row; it does not quotient streams, choose a host limit, use countable choice, invoke trichotomy, identify host sequences, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7206.1 (Bishop real regularization carrier)**.** *A* $\mathsf{BishopRealRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the raw Cauchy source row, exposing the finite source requests and their Cauchy-window evidence.*
2. *T is the Bishop regularization threshold row, recording the finite stage at which each requested tolerance is served by a regularized window.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, naming the source windows and the regularized output windows inspected at the threshold supplied by T.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name produced by T and W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording source tolerances, threshold tolerances, readback tolerances, and the Real-facing output budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after Q, T, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for $Q, T, W, R, D, E, C, P, N$.*
8. *C records the displayed* Cont *replay from a Bishop regularization request through $Q, T, W, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealRegularization}^{\uparrow}$, $\mathsf{RegularCauchyRegularization}^{\uparrow}$, SequentialCauchyRegulari $\mathsf{LocatedRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealRegularization}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite Bishop real-name regularization history. It has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, host sequence equality, or ambient Real completeness.

Theorem 7206.2 (Bishop real regularization NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealRegularization}^{\uparrow}$ *carrier* $B = (Q, T, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealRegularization}^{\uparrow}}$ *surface consists exactly of raw Cauchy source exposure, Bishop threshold exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No consumer receives quotient streams, selected host limits, countable-choice schedules, trichotomy decisions, host sequence equality, or ambient Real completeness from the packet.*

Proof. Project the carrier of Definition 7206.1. Its analytic rows are Q, T, W, R, D, E, so every accepted regularization read must expose the raw Cauchy source, threshold, finite windows, rational readback, tolerance ledger, and terminal seal.

The regularization obligations are read from T, W, R, D. The threshold row supplies the finite stage, the window row displays the source and output observations at that stage, the readback row packages those observations as a regular rational name, and the dyadic ledger records the tolerances used by the whole route.

The structural rows preserve the same route. Transport H is componentwise, replay C consumes the displayed rows in dependency order, and P, N record provenance and local naming.

Since the carrier contains no quotient-stream, selected-limit, choice, trichotomy, host-equality, or completeness coordinate, none of those objects can be exported through the local NameCert surface. □

Theorem 7206.3 (Bishop real regularization Real-seal handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{BishopRealRegularization}^{\uparrow}$ *packet factors through the finite route*

$$Q \longmapsto T \longmapsto W, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the consumer reads the raw Cauchy source, then the Bishop regularization threshold, then finite stream windows and dyadic tolerance data, then the regular rational readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no quotient stream, selected limit, countable-choice selector, trichotomy row, host sequence equality, or ambient completeness theorem.*

Proof. Begin with an accepted carrier $B = (Q, T, W, R, D, E, H, C, P, N)$. The source row Q is the only raw Cauchy source coordinate, so a Real-facing read starts there.

Use the threshold and window rows next. The row T supplies the finite Bishop regularization stage, while W and D record the finite observations and tolerances checked at that stage.

The readback row R is downstream of the threshold and window data. It is the only regular rational name available before the terminal seal is inspected.

Finally read E, H, C, P, N. These rows seal, transport, replay, source, and name the same finite route. They do not add quotient, selection, choice, trichotomy, host-equality, or completeness data. □

Theorem 7206.4 (Bishop real regularization no-choice boundary)**.** *An accepted* $\mathsf{BishopRealRegularization}^{\uparrow}$ *carrier is a finite regularization packet over* $Q, T, W, R, D, E, H, C, P, N$ *and not a countable-choice principle. Any attempt to read a chosen limit, a selected representative stream, a host sequence-equality decision, or a trichotomy comparison must leave the displayed carrier surface before the* $\mathsf{Real}^{\uparrow}$ *seal can be used.*

Proof. Project the carrier through Definition 7206.1. The rows available to a consumer are exactly $Q, T, W, R, D, E, H, C, P, N$.

Apply the NameCert obligations of Definition 7206.2. They force every accepted read to expose the raw source, threshold, windows, readback, tolerance ledger, seal, and structural rows.

Use the Real-seal handoff of Definition 7206.3. The terminal seal is downstream of the finite route and cannot be reached from a selected limit or representative stream.

Therefore countable choice, trichotomy, host sequence equality, selected limits, and quotient streams are outside the packet boundary. A consumer that uses any of them is no longer reading the displayed $\mathsf{BishopRealRegularization}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealRegularization}^{\uparrow}$ packet is generated from a raw Cauchy source row, Bishop regularization threshold, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7206.1, Definition 7206.2, Definition 7206.3, and Definition 7206.4 seed the finite route from raw Cauchy source data through Bishop regularization, regular rational readback, dyadic tolerance accounting, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold stability, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealRegularizationUp.

Chapter 7207

A Concrete Naming Certificate for $\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}$

Bishop-style constructive compact metric analysis bounds the image of a uniformly continuous map by reading finite source nets through a displayed modulus and then sealing the resulting radius as a real-name row. The $\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}$ packet records that finite-net bounded-image route as NameCert data. It consumes the compact source surface of chapter 129, the compact interval source route of chapter 3986, the graph data of chapter 130, the tolerance ledger of chapter 369, the dyadic-radius arithmetic of chapter 291, the rational readback of chapter 301, and the terminal seal of chapter 38. The packet names a bounded-image certificate; it does not introduce open-cover compactness, a host image subset, selected extrema, quotient real equality, or ambient completeness.

Definition 7207.1 (Constructive uniform bounded-image carrier)**.** *A* $\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, U, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is a compact finite-net source row, read either through* $\mathsf{CompactMetric}^{\uparrow}$ *or through the interval finite-net source data of* $\mathsf{CompactInterval}^{\uparrow}$*.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from the displayed source carrier to the target metric carrier.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row that sends a target-radius request to a source-net radius request for the same graph row.*
4. *M is the target* $\mathsf{Met}^{\uparrow}$ *distance row used to compare all displayed image centers with the chosen base image center.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger containing the source radius, target tolerance, finite maximum-radius fold, and comparison scale.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that packages the dyadic radius ledger as a regular rational bound schedule.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *bound seal reached only after the compact source row, map graph, uniform modulus, target metric distance row, dyadic ledger, and rational readback have been displayed.*
8. *H records componentwise* hsame *transport for K, F, U, M, D, R, E and the local structural rows.*

9. *C records displayed* Cont *replay from a bounded-image request through source nets, graph evaluation, uniform-modulus routing, dyadic radius folding, rational readback, and the terminal bound seal.*

10. *P is the* Pkg *provenance over* ConstructiveUniformBoundedImage$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, Met$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ConstructiveUniformBoundedImage}^{\uparrow}}$ *row.*

The carrier contains no coordinate for open-cover compactness, host image subsets, selected extrema, quotient equality of real names, choice-selected image centers, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7207.2 (Constructive uniform bounded-image NameCert obligations)**.** *For every accepted* ConstructiveUniformBoundedImage$^{\uparrow}$ *carrier $B = (K, F, U, M, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveUniformBoundedImage}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net source admission through K, continuous-map graph admission through F, uniform-modulus compatibility through U, target metric-distance admission through M, dyadic radius-ledger admission through D, regular rational readback through R, terminal* Real$^{\uparrow}$ *bound sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 7207.1. The packet displays the compact source row, graph row, modulus row, target metric row, dyadic ledger, rational readback, real seal, transport, replay, provenance, and local naming as the only available coordinates.

The analytic rows are read in finite order. First the compact source row K supplies finite net centers, then the graph row F evaluates the map on those centers, and the uniform modulus row U ties the requested target radius to the displayed source coverage radius.

The boundedness rows are then forced through M, D, R, E. The target metric row measures finite image-center distances, the dyadic ledger records the finite radius fold, the regular rational row reads the radius schedule, and the Real$^{\uparrow}$ row seals the resulting bound.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no excluded host-image, selected-extremum, quotient, choice, open-cover, or completeness coordinate occurs in the carrier, those data cannot be exported by the local NameCert surface. □

Theorem 7207.3 (Constructive uniform bounded-image finite-net bound)**.** *Every bounded-image consumer of an accepted* ConstructiveUniformBoundedImage$^{\uparrow}$ *carrier receives only the finite route*

$$K, F, U, M, D, R, E, H, C, P, N.$$

The bound is obtained by evaluating the displayed ContinuousMap$^{\uparrow}$ *graph on the compact finite-net centers from* CompactMetric$^{\uparrow}$ *or* CompactInterval$^{\uparrow}$*, transporting the tolerance request through* UniformModulus$^{\uparrow}$*, folding the resulting target distances in the* DyadicRatCore$^{\uparrow}$ *ledger, reading the radius through* RegSeqRat$^{\uparrow}$*, and sealing it through* Real$^{\uparrow}$*. The route is a finite bounded-image modulus, not an open-cover compactness theorem, a host image subset, a selected maximum, quotient real equality, or an ambient completeness claim.*

Proof. Start with the compact source row K. It exposes finite source centers and coverage radii only; if the source is an interval packet, the interval row is read as the same finite-net source surface.

Read the map and modulus rows F, U. The graph row supplies the image centers, while the uniform modulus converts the requested target radius into the source-net radius used by K.

Now compute the displayed radius ledger. The target metric row M measures distances from the chosen base image center to the finite list of image centers, and the dyadic ledger D records the finite maximum-radius fold together with the target tolerance scale.

Finally read R and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row packages the dyadic radius as a regular rational bound schedule, and the $\mathsf{Real}^{\uparrow}$ row seals that schedule as the bounded-image radius. Transport, replay, provenance, and local naming preserve this route without adding open-cover compactness, host image subsets, selected extrema, quotient real equality, choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}$ packet is generated from CompactMetric or CompactInterval finite-net source data, ContinuousMap graph rows, a UniformModulus ledger, target Metric distance rows, a DyadicRatCore radius ledger, RegSeqRat readback, a RealUp bound seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7207.1, Definition 7207.2, and Definition 7207.3 seed the finite-net bounded-image carrier and its radius-seal route over compact source data, continuous maps, uniform moduli, dyadic ledgers, regular rational readback, and RealUp seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformBoundedImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host image subsets, selected extrema, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net source admission, graph and modulus compatibility, dyadic radius folding, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local NameCert registration for BEDC.Derived.ConstructiveUniformBoundedImageUp.

Chapter 7208

A Concrete Naming Certificate for $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$

Following Bishop–Bridges compact metric analysis and the Arzela–Ascoli family-modulus route, the $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$ packet records the finite envelope by which compact finite nets, equicontinuity rows, and rational lower-bound folds produce one uniform-modulus handoff for a displayed family. It realizes the compact-uniform-continuity vision of chapter 9237 through the compact metric source of chapter 129, the equicontinuity surface of chapter 4076, the uniform-modulus consumer of chapter 369, the continuous-map rows of chapter 130, the metric-distance rows of chapter 57, the rational fold interface of chapter 37, and the terminal real-name seal of chapter 38. The packet names a finite family envelope; it does not import open-cover compactness, a selected subcover, quotient equality of maps, a host compact-image theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7208.1 (CompactUniformEquicontinuityEnvelope carrier)**.** *A* $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, M, Q, U, D, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including the compact source, center rows, coverage radii, and located metric data.*
2. *F is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family row, exposing the shared family modulus requests read at the bundled centers.*
3. *M is the family-modulus envelope row, tying each center and family observation to one displayed radius ledger.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the finite family-radius ledger.*
5. *U is the resulting* $\mathsf{UniformModulus}^{\uparrow}$ *handoff consumed by compact-uniform family arguments.*
6. *D is the* $\mathsf{Met}^{\uparrow}$ *source and target distance row used for the triangle transport of the shared radius.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for real-valued family reads after the finite fold and metric rows have been displayed.*
8. *H records componentwise* hsame *transport for $K, F, M, Q, U, D, R, C, P, N$.*

9. *C records displayed* Cont *replay from compact-net extraction to family-modulus reads, rational lower-bound folding, metric-triangle transport, uniform-modulus export, Real sealing, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected subcovers, unrestricted host equality of maps, a quotient function space, countable choice, or ambient Real completeness.

Theorem 7208.2 (CompactUniformEquicontinuityEnvelope NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$ *carrier* $E = (K, F, M, Q, U, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, equicontinuity family exposure through F, family-modulus envelope visibility through M, rational lower-bound fold exactness through Q, uniform-modulus export through U, metric-distance transport through D, Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7208.1. The compact source is exhausted by K, which supplies the finite net and located metric coverage data.

The family side is exhausted by F and M. These rows display the equicontinuity request and the family-radius envelope at the compact-net centers. The single global radius can only be read after Q folds those finitely many rational rows.

The consumer side is exhausted by U, D, R. The uniform-modulus row exports the shared radius, the metric row performs the source and target triangle transports, and the Real row seals the real-valued reads after those finite data are public. The structural rows H, C, P, N only transport, replay, source, and name the same packet, so the listed obligations exhaust the local certificate. □

Theorem 7208.3 (CompactUniformEquicontinuityEnvelope family-modulus handoff)**.** *Every compact-uniform, equicontinuity, or Arzela–Ascoli consumer of an accepted* $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Equicontinuity}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives compact finite nets, displayed family moduli, rational lower-bound folds, metric-triangle transport, uniform-modulus export, and the Real seal, but it receives no host compactness theorem, selected subcover, quotient equality of maps, family choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the compact and family rows first. The compact metric row K supplies the finite net, and the equicontinuity row F supplies the shared family-modulus requests visible at those centers.

Next read the envelope and fold rows. The row M binds the finitely many center and family observations to radius ledgers, and Q is the only coordinate that turns those ledgers into one rational lower bound. Thus the proposed family modulus is a finite fold, not a selected external function.

Finally read the consumer rows. The metric row D transports the center estimates to arbitrary source points, U records the tolerance-only uniform-modulus handoff, and R seals the Real-facing

reads. Since the carrier has no coordinate for host compactness, selected subcovers, quotient equality, family choice, or ambient completeness, every accepted handoff remains on the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, Equicontinuity family-modulus rows, a displayed family envelope, a RatUp lower-bound fold, UniformModulus export, Metric transport, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7208.1, Definition 7208.2, and Definition 7208.3 seed the compact family-modulus envelope over compact finite nets, equicontinuity rows, RatUp folds, UniformModulus handoff, Metric transport, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, selected subcovers, quotient equality of maps, family choice principles, Arzela–Ascoli compactness, standard bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, family-modulus envelope exactness, rational lower-bound fold exactness, uniform-modulus export, metric transport, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformEquicontinuityEnvelopeUp.

Chapter 7209

A Concrete Naming Certificate for $\mathsf{CofinalReindexing}^{\uparrow}$

Following Bishop and Bridges on cofinal subsequences of regular Cauchy names, the $\mathsf{CofinalReindexing}^{\uparrow}$ packet records the finite reindexing route by which a displayed tail request is moved along a monotone cofinal index row before any regular rational readback or Real seal is consumed. It sits between the general reindexing surface of chapter 1568, the regular Cauchy specialization of chapter 6960, the finite tail-map packet of chapter 2141, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite cofinal-tail reindex certificate; it is not quotient stream equality, not a selected subsequence limit, not countable choice, not host real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7209.1 (Cofinal reindexing carrier)**.** *A* $\mathsf{CofinalReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the cofinal monotone index row, recording the finite target-to-source request map, its displayed order comparisons, and the finite reachability witnesses for the target requests under inspection.*
2. *T is the tail-window row, recording the target tail cuts and the source cuts selected by M for the current finite request window.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row read after T has fixed the target and source windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the stream windows have been routed through M and T.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular rational readback has been displayed.*
6. *H records componentwise* hsame *transport for M, T, S, R, E, C, P, N.*
7. *C records displayed* Cont *replay from a target precision request through the cofinal index row, tail-window row, StreamName window, RegSeqRat readback, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalReindexing}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{RegularCauchyReindexing}^{\uparrow}$, $\mathsf{CofinalTailMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CofinalReindexing}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7209.2 (Cofinal reindexing NameCert obligations)**.** *For every accepted* $\mathsf{CofinalReindexing}^{\uparrow}$ *carrier $G = (M, T, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalReindexing}^{\uparrow}}$ *surface consists exactly of carrier admission for the cofinal monotone index row, tail-window row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. The obligations require each target request to reach a displayed source cut through M and T before any stream window, regular readback, or Real seal is consumed.*

Proof. Project the carrier of Definition 7209.1. The rows M, T, S, R, E are the complete analytic route, while H, C, P, N transport, replay, source, and name that route.

The index row M supplies only finite monotone comparisons and cofinal reachability witnesses for the requests under inspection. The tail-window row T records the target cuts and the corresponding source cuts selected by that row.

Once M and T are fixed, the $\mathsf{StreamName}^{\uparrow}$ row S may be read, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of the same finite windows. The terminal seal E is therefore a Real-facing read of the displayed reindexing route, not a host-selected subsequence, quotient stream, countable-choice schedule, or ambient completeness claim. □

Theorem 7209.3 (Cofinal reindexing tail-window stability)**.** *Let $G = (M, T, S, R, E, H, C, P, N)$ and $G' = (M', T', S', R', E', H', C', P', N')$ be accepted* $\mathsf{CofinalReindexing}^{\uparrow}$ *carriers whose cofinal index rows and tail-window rows agree through the displayed classifier. Then every* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *read transported from G to G' uses the same finite target cuts, source cuts, monotone comparisons, and cofinal reachability witnesses before reaching the Real seal.*

Proof. Use the classifier of Definition 7209.1. Agreement of M, T fixes the target requests, source cuts, order comparisons, and cofinal witnesses visible to both packets.

Apply Definition 7209.2. The stream window S and regular readback R cannot be read until the index and tail-window rows have been displayed, so their transported versions remain attached to the same finite route.

The structural rows H, C, P, N preserve only componentwise transport, replay, provenance, and naming. They do not add a selected subsequence, quotient equality, countable-choice schedule, host equality, or ambient completeness row, so the transported Real seal remains downstream of the same tail-window data. □

Theorem 7209.4 (Cofinal reindexing Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CofinalReindexing}^{\uparrow}$ *carrier factors through the finite route*

$$M \longmapsto T \longmapsto S \longmapsto R \longmapsto E.$$

Thus the terminal seal is reached only after the cofinal monotone index row, tail-window row, $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed. The read exports no quotient stream equality, selected subsequence limit, countable-choice schedule, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the seal row E. By Definition 7209.2, the seal is downstream of R, and R is downstream of the stream window S.

The stream window is not free-standing. It is read only after M has selected displayed source cuts for the target requests and T has recorded the corresponding tail windows.

Componentwise transport, replay, provenance, and local naming keep that route visible. Since none of the excluded host or quotient objects is a carrier coordinate, the Real-facing consumer cannot bypass the finite cofinal reindexing packet. □

Theorem 7209.5 (Cofinal reindexing seed scope)**.** *The* $\mathsf{CofinalReindexing}^{\uparrow}$ *seed closes exactly the finite cofinal reindexing route from displayed target requests to source cuts, StreamName windows, RegSeqRat readback, and the terminal Real seal. It supplies a handoff surface for Real and Completion consumers using chapter 1568, chapter 6960, and chapter 2141; it does not close quotient stream equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.*

Proof. The seed route is the carrier route of Definition 7209.1. It displays the cofinal index row, tail windows, stream window, regular readback, Real seal, transport, replay, provenance, and local naming.

Use Definition 7209.3 for stability of the finite source and target windows, and use Definition 7209.4 for the terminal Real-facing boundary.

These results are exactly the finite handoff needed by the surrounding reindexing, tail-map, StreamName, RegSeqRat, and Real chapters. They do not add any row for the excluded quotient, selection, choice, completeness, verification, or bridge claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalReindexing}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalReindexing}^{\uparrow}$ packet is generated from a cofinal monotone index row, a tail-window row, a StreamName window, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalReindexing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7209.1, Definition 7209.2, Definition 7209.3, Definition 7209.4, and Definition 7209.5 seed the finite cofinal reindexing route from displayed target requests through StreamName windows and RegSeqRat readback to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalReindexing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal monotone index exposure, tail-window stability, regular readback ordering, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CofinalReindexingU

Chapter 7210

A Concrete Naming Certificate for BishopCompletionModulusSelection$^{\uparrow}$

Following Bishop and Bridges, the BishopCompletionModulusSelection$^{\uparrow}$ packet names the finite threshold-selection surface that sits between a displayed completion modulus and the terminal Real-facing seal. It refines the finite completion route of chapter 1469, reads monotone thresholds from chapter 292, opens finite windows through chapter 126, accounts for dyadic tolerances through chapter 291, repacks the selected window through chapter 301, and hands the result to the terminal boundary of chapter 38. The packet names only a displayed finite threshold selection; it does not select a completed limit, quotient stream representative, host real equality witness, countable-choice schedule, or ambient completeness principle.

Definition 7210.1 (Bishop completion modulus selection carrier)**.** *A* BishopCompletionModulusSelection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, n, k, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* BishopCompletionModulus$^{\uparrow}$ *row whose visible modulus route is being queried.*
2. *n is the displayed precision request.*
3. *k is the finite threshold row selected from M at request n.*
4. *W is the* StreamName$^{\uparrow}$ *finite window read at the selected threshold.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to the same threshold and window.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed window and dyadic ledger.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row available only after M, n, k, W, D, R are displayed.*
8. *H records componentwise* hsame *transport for the modulus, request, threshold, window, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the precision request through the selected threshold to the window, dyadic ledger, readback, and seal.*
10. *P is the* Pkg *provenance over* BishopCompletionModulusSelection$^{\uparrow}$, BishopCompletionModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient stream equality, a selected completed limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BishopCompletionModulusSelectionUp.BishopCompletionModulusSelectionUp`
Lean4 ✓ `BEDC.Derived.BishopCompletionModulusSelectionUp.BishopCompletionModulusSelectionTasteGate`
`single carrier alignment`

Theorem 7210.2 (Bishop completion modulus selection NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionModulusSelection}^{\uparrow}$ *carrier* $S = (M, n, k, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-modulus row exposure, precision request exposure, finite threshold selection,* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside those rows.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusSelectionUp.BishopCompletionModulusSelectionNameCertObligat`

Proof. Project the carrier of Definition 7210.1. The analytic route rows are M, n, k, W, D, R, E, and the structural rows are H, C, P, N.

The threshold obligation is local. The row k is read from the displayed completion-modulus row M at the request n, and the window row W is read only after that threshold is visible.

The dyadic and regular rows are downstream from the same threshold. The ledger D records the tolerance data for W, while R repacks W, D as the regular rational handoff before the seal E is available.

Finally H, C, P, N transport, replay, source, and name exactly that route. Since no other carrier coordinate exists, quotient stream equality, selected limits, countable choice, host real equality, and ambient completeness cannot be local obligations of the packet. □

Theorem 7210.3 (Bishop completion modulus selection threshold choice)**.** *Every accepted* BishopCompletionModul *carrier selects its finite threshold through the displayed route*

$$M, n \longmapsto k \longmapsto W, D.$$

The selected threshold is therefore a row of the packet rather than a hidden choice function: downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers may inspect only the displayed threshold, the finite window opened at that threshold, and the dyadic tolerance ledger attached to the same window.*

Proof. Start from the carrier projection. The request n and threshold k are displayed before the stream window and dyadic ledger rows.

Apply the obligation surface of Definition 7210.2. It forces the threshold read to pass through M, n, k before W or D can be consumed.

The structural rows preserve the same finite route. Thus a downstream consumer cannot replace k by an external choice function or by a selected limit without leaving the accepted carrier surface. □

Theorem 7210.4 (Bishop completion modulus selection Real-seal nonescape)**.** *Every Real-facing read of an accepted* $\mathsf{BishopCompletionModulusSelection}^{\uparrow}$ *carrier factors through*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E.$$

Consequently the terminal $\mathsf{Real}^{\uparrow}$ *seal receives only the displayed completion-modulus threshold, finite stream window, dyadic tolerance ledger, and regular rational handoff. It does not export quotient stream equality, selected completed limits, countable-choice schedules, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7210.3. It places the selected threshold and finite window before the dyadic ledger is read.

The regular handoff is then forced by the carrier order. The row R is downstream from W, D, and the seal row E is downstream from R.

Finally apply the non-escape part of Definition 7210.2. The excluded quotient, selection, choice, host-equality, and completeness data have no carrier projection, so the terminal seal cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionModulusSelection}^{\uparrow}$ packet is generated from a displayed BishopCompletionModulus row, a precision request, a selected finite threshold, a StreamName window, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7210.1, Definition 7210.2, Definition 7210.3, and Definition 7210.4 seed the finite threshold-selection surface between completion-modulus data and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, quotient stream equality, selected completed limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite threshold selection, StreamName window readback, DyadicRatCore tolerance routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionModulusSelectionUp.

Chapter 7211

A Concrete Naming Certificate for $\mathsf{ConstructiveRealNormalization}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ConstructiveRealNormalization}^{\uparrow}$ packet records the finite normal-form handoff from a raw Cauchy source through Bishop real regularization to the public constructive real-line interface. It reads the regularization row of chapter 7206, the public line row of chapter 4041, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the non-quotient route by which regularized Cauchy data becomes a public $\mathsf{Real}^{\uparrow}$ row without importing selected limits, trichotomy, countable choice, quotient streams, or host real equality.

Definition 7211.1 (ConstructiveRealNormalization carrier)**.** *A* $\mathsf{ConstructiveRealNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (Q, B, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the raw Cauchy source row, exposing the finite source requests and their Cauchy-window evidence.*
2. *B is the* $\mathsf{BishopRealRegularization}^{\uparrow}$ *row that regularizes the raw source before any public real-line consumer can read it.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row shared by the source, regularized output, and public interface read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name delivered by the regularization route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording source budgets, regularization budgets, readback budgets, and Real-facing output budgets.*
6. *L is the* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *interface row through which public real-line consumers inspect the same regularized readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after Q, B, W, R, D, L have been displayed.*
8. *H records componentwise* hsame *transport for $Q, B, W, R, D, L, E, C, P, N$.*

9. *C records the displayed* Cont *replay from a normalization request through* $Q, B, W, R, D, L, E, H, P, N$.
10. *P is the* Pkg *provenance over* ConstructiveRealNormalization$^{\uparrow}$, BishopRealRegularization$^{\uparrow}$, ConstructiveRealLine$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ConstructiveRealNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7211.2 (ConstructiveRealNormalization window determinacy)**.** *Let $Z = (Q, B, W, R, D, L, E, H, C, P, N$ be an accepted* ConstructiveRealNormalization$^{\uparrow}$ *packet. Every public* ConstructiveRealLine$^{\uparrow}$ *or* Real$^{\uparrow}$ *read of E determines the same finite normalization window*

$$W_{\text{norm}}(Z) = (Q, B, W, R, D, L)$$

through the raw Cauchy source, Bishop regularization row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and constructive real-line interface row. No second source window, quotient stream, selected limit, countable-choice schedule, trichotomy branch, host real equality, or ambient completeness principle can supply an alternate public read of the same seal.

Proof. Project the carrier of Definition 7211.1. The only analytic rows before the seal are Q, B, W, R, D, L, so every public read starts with the raw source, regularization packet, finite windows, readback row, tolerance ledger, and real-line interface.

Read chapter 7206. Its regularization handoff forces the raw source through the displayed Bishop row before the terminal real-name seal can be inspected.

Read chapter 4041 next. The public interface row consumes the same StreamName, RegSeqRat, and DyadicRatCore route before exposing a Real$^{\uparrow}$ consumer surface.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host equality, or ambient completeness, none of those objects can determine a second window for the seal. □

Closure status for NameCert$_{\text{ConstructiveRealNormalization}^{\uparrow}}$.

Constructive story: A ConstructiveRealNormalization$^{\uparrow}$ packet is generated from a raw Cauchy source row, Bishop real regularization row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, ConstructiveRealLine interface row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveRealNormalization}^{\uparrow}}$, seed)

Scope closed: Definition 7211.1 and Definition 7211.2 seed the finite route from raw Cauchy source data through Bishop regularization, public real-line interface, and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveRealNormalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Bishop regularization exposure, StreamName window exactness, RegSeqRat readback, DyadicRatCore tolerance routing, ConstructiveRealLine interface handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveRealNormalizationUp.

Chapter 7212

A Concrete Naming Certificate for $\mathsf{LawvereMetric}^{\uparrow}$

Following Lawvere's enriched-category account of generalized metric spaces, the $\mathsf{LawvereMetric}^{\uparrow}$ packet records a directed distance interface as finite BEDC NameCert data rather than as an ambient completed metric object. It sits between the directed comparison surface of chapter 2158, the Yoneda-facing route of chapter 2148, the formal-ball surface of chapter 1510, the symmetric metric sibling of chapter 57, the dyadic and regular rational readbacks of chapter 291 and chapter 301, the terminal real seal of chapter 38, and the completion consumer of chapter 3947. The packet names the base directed-distance ledger; it does not claim Real completeness, Cauchy completion, selected colimits, quotient equality of points, symmetry, or separatedness.

Definition 7212.1 (Lawvere metric carrier)**.** *A* $\mathsf{LawvereMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Z, T, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the directed source row naming the finite object-window on which directed distances are read.*
2. *Z is the zero/self ledger, recording that each displayed source entry carries the directed self-distance row expected by a Lawvere-style metric interface.*
3. *T is the triangle ledger, recording finite composable directed-distance triples before any completion or formal-ball consumer sees them.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *distance budget row used for finite tolerance reads.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row converting displayed dyadic distance windows into regular rational observations.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for directed distance values after D and Q have been exposed.*
7. *H records componentwise* hsame *transport for X, Z, T, D, Q, R and for the structural rows.*
8. *C records displayed* Cont *replay from the source row through*

$$X, Z, T, D, Q \longmapsto R \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{LawvereMetric}^{\uparrow}$, $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LawvereMetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LawvereMetric}^{\uparrow}}$ *row.*

The carrier contains no coordinate for symmetry, separatedness, selected Yoneda limit, completed formal-ball object, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification row, or bridge credential.*

Lean4 (statement) `BEDC.Derived.LawvereMetricUp`

Theorem 7212.2 (Lawvere metric triangle ledger stability)**.** *For every accepted* $\mathsf{LawvereMetric}^{\uparrow}$ *carrier* $L = (X, Z, T, D, Q, R, H, C, P, N)$*, any* $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *consumer of a displayed directed-distance comparison must read the same directed source row X, zero/self ledger Z, triangle ledger T, dyadic budget D, regular rational readback Q, and real seal R before it treats the comparison as stable under* hsame *transport and* Cont *replay.*

Proof. Project the carrier of Definition 7212.1. The directed source row X fixes the finite window of points, while Z and T are the only rows naming self-distance and triangle data.

The numeric side is not read directly as host real equality. The dyadic row D supplies finite tolerance data, Q repacks it as regular rational readback, and R is the terminal $\mathsf{Real}^{\uparrow}$ seal visible to distance consumers.

The structural rows preserve that route. Componentwise hsame transport in H carries the same displayed rows, and Cont replay in C permits the consumer to replay the path from directed source data to real-sealed distance data.

Thus any sibling consumer must factor through $X, Z, T, D, Q, R, H, C, P, N$. The carrier has no additional coordinate from which to extract symmetry, separatedness, selected completion data, quotient point equality, or ambient Real completeness. □

Theorem 7212.3 (Lawvere metric directed source exposure)**.** *For every accepted* $\mathsf{LawvereMetric}^{\uparrow}$ *carrier* $L = (X, Z, T, D, Q, R, H, C, P, N)$*, any* $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *consumer must expose the directed source row X before it reads D, Q, or R as distance data.*

Proof. Project the carrier of Definition 7212.1. The row X is the only displayed source row, while D, Q, and R are downstream numeric rows: the dyadic budget, the regular rational readback, and the terminal Real seal.

Apply Definition 7212.2. Every listed consumer of a displayed directed-distance comparison must read the same X, Z, T, D, Q, R surface before treating the comparison as stable under hsame transport and Cont replay. Hence the numeric read of D, Q, R has no independent source; it is exposed only after the directed source row X is fixed. □

Theorem 7212.4 (Lawvere metric zero-self exactness)**.** *For every accepted* $\mathsf{LawvereMetric}^{\uparrow}$ *carrier* $L = (X, Z, T, D, Q, R, H, C, P, N)$*, the self-distance row Z and the triangle row T are mandatory finite ledger rows. They are not consequences of symmetry, separatedness, quotient point equality, selected completion data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier in Definition 7212.1 lists Z and T as displayed rows of the packet. The row Z records the directed self-distance ledger for the displayed source entries, and T records finite composable directed-distance triples before any completion or formal-ball consumer sees them.

The same definition also exhausts the excluded coordinates. The carrier contains no row for symmetry, separatedness, selected Yoneda limit, completed formal-ball object, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge credential. Therefore Z and T are finite NameCert obligations carried by L itself, not consequences imported from any excluded structure. □

Theorem 7212.5 (Lawvere metric NameCert obligation surface)**.** *Every accepted* $\mathsf{LawvereMetric}^{\uparrow}$ *carrier* $L = (X, Z, T, D, Q, R, H, C, P, N)$ *exposes the following finite NameCert obligation rows:*

1. *directed source exposure through* X*;*
2. *zero/self ledger exactness through* Z *and triangle ledger stability through* T*;*
3. *dyadic-RegSeqRat-Real sealing through* D, Q, R*;*
4. *structural replay through* H *and* C*;*
5. *provenance exhaustion and local naming through* P *and* N*.*

Proof. The carrier definition supplies the rows $X, Z, T, D, Q, R, H, C, P, N$ and assigns each row its role. This gives the finite object-window, self-distance ledger, triangle ledger, dyadic budget, regular rational readback, Real seal, componentwise transport, replay route, provenance, and local NameCert name.

The directed source row is exposed by Definition 7212.3. The exactness of the mandatory zero/self and triangle rows follows from Definition 7212.4, while the consumer-facing stability of the same directed-distance surface follows from Definition 7212.2.

The remaining rows are structural rather than optional. The definition routes D, Q, R as the dyadic-RegSeqRat-Real distance seal, H and C as transport and replay, and P, N as provenance and local naming. Since the carrier has no coordinate for the excluded symmetry, separatedness, completion, quotient, Real-completeness, Lean, or bridge data, the five displayed clauses are exactly the finite obligation surface presented by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawvereMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LawvereMetric}^{\uparrow}$ packet is generated from a directed source row, zero/self ledger, triangle ledger, DyadicRatCore distance budget, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LawvereMetric}^{\uparrow}}$, seed)

Scope closed: Definition 7212.1, Definition 7212.2, and Definition 7212.5 seed the finite directed-distance interface consumed by QuasiMetric, FormalBall, YonedaCompletion, Metric, and LawvereMetricCompletion packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawvereMetric}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.LawvereMetricUp`

Bridge status: none

Not claimed: This chapter does not close symmetry, separatedness, Cauchy completion, selected Yoneda limits, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed source exposure, zero/self ledger exactness, triangle ledger stability, dyadic and regular rational readback, RealUp sealing, replay stability, provenance exhaustion, and local naming for BEDC.Derived.LawvereMetricUp.

Chapter 7213

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$ packet records the finite comparison object that lets a completion-facing modulus row be checked against a regular Cauchy readback before any Real or Completion consumer reads a terminal seal. It sits between the completion-modulus discipline of chapter 1469, the generic Cauchy threshold surface of chapter 292, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names a comparison route for L10 Real and Completion work; it does not select limits, quotient streams, host representatives, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7213.1 (Regular Cauchy modulus comparison carrier)**.** *A* $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row, recording the completion-facing modulus budget that later consumers may reuse.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, recording the finite tail request attached to the same tolerance.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observations inspected after B and M have fixed the requested budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational data obtained from those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic comparison ledger shared by B, M, W, R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the displayed completion-modulus, threshold, window, readback, and dyadic rows have been read.*
7. *H stores componentwise* hsame *transport for the analytic rows and the structural rows.*
8. *C stores displayed* Cont *replay from a tolerance request through B, M, W, R, D, E.*

9. *P records* Pkg *provenance over* RegularCauchyModulusComparison$^{\uparrow}$, BishopCompletionModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *data.*

10. *N is the local* NameCert$_{\text{RegularCauchyModulusComparison}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $B, M, W, R, D, E, H, C, P, N$. No coordinate stores a quotient stream, selected limit, host real equality, ambient completion object, or Real$^{\uparrow}$ *completeness theorem.*

Theorem 7213.2 (Regular Cauchy modulus comparison NameCert obligations)**.** *For every accepted* RegularCauchyModulusComparison$^{\uparrow}$ *carrier $K = (B, M, W, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyModulusComparison}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the completion modulus B, Cauchy threshold M, finite windows W, regular readback R, dyadic tolerance ledger D, Real seal E, transport H, replay C, provenance P, and local naming N;*

2. *modulus compatibility, requiring B and M to answer the same displayed dyadic tolerance request before any window row is inspected;*

3. *readback comparison, requiring R to be obtained only from the* StreamName$^{\uparrow}$ *windows W under the dyadic ledger D and the threshold discipline supplied by B and M;*

4. *Real-seal nonescape, requiring the terminal row E to be downstream of B, M, W, R, D and to export no selected limit, quotient stream, host equality, countable choice, or ambient completeness;*

5. *structural preservation, requiring H, C, P, N to transport, replay, source, and name the same finite comparison packet.*

No additional NameCert obligation is available from the comparison carrier.

Proof. Carrier admission is the projection of Definition 7213.1. The tuple lists exactly the completion-modulus, Cauchy-threshold, finite-window, readback, dyadic, Real-seal, transport, replay, provenance, and naming rows.

For modulus compatibility, the only tolerance rows are B, M, and D. A consumer must therefore read the dyadic request through D and then expose both the completion-facing modulus row and the Cauchy-threshold row before it can inspect any StreamName window.

Readback comparison is downstream of the same rows. The row W supplies the finite observations, and R repacks those observations as regular rational data under the displayed threshold and tolerance discipline.

Finally E is the terminal Real-facing row, while H, C, P, N preserve the same packet. Since no quotient stream, selected limit, host equality, choice schedule, or ambient completeness coordinate occurs in the carrier, those objects cannot be exported by the local certificate. □

Theorem 7213.3 (Regular Cauchy modulus comparison stability)**.** *Let $K = (B, M, W, R, D, E, H, C, P, N)$ be an accepted* RegularCauchyModulusComparison$^{\uparrow}$ *carrier. If a later tolerance request is discharged by the same displayed* BishopCompletionModulus$^{\uparrow}$ *row B and* CauchyModulus$^{\uparrow}$ *row M with a refined* DyadicRatCore$^{\uparrow}$ *ledger still recorded in D, then every accepted Real-facing read factors through the same* StreamName$^{\uparrow}$ *window route, the same* RegSeqRat$^{\uparrow}$ *readback row, and the same terminal* Real$^{\uparrow}$ *seal row. Thus refinement preserves the comparison between the completion modulus and the regular Cauchy readback.*

Proof. Start with the refined tolerance request. By the NameCert obligations of Definition 7213.2, the request is still read through D and must expose the same completion-modulus and Cauchy-threshold rows B and M.

The finite window row is downstream of those rows. Hence the refined request reaches observations only through W, and the regular rational readback can only be the displayed row R attached to those windows.

The terminal Real-facing consumer then reads E after B, M, W, R, D. Componentwise transport, replay, provenance, and local naming preserve the same packet, so the refined request stays on the original comparison route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first displaying $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows refutes the carrier.

Independence witness: bishopcompletionmodulus, cauchymodulus, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$ because this packet jointly carries completion-modulus reuse, Cauchy-threshold exposure, finite windows, regular readback, dyadic tolerance comparison, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7213.1, Definition 7213.2, and Definition 7213.3 seed the finite comparison route tying completion moduli to regular Cauchy readback over displayed tolerance, window, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, host real equality, countable choice, ambient completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus compatibility, readback comparison, Real-seal nonescape, and stability for BEDC.Derived.RegularCauchyModulusComparisonUp.

Constructive story: A $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$ packet is generated from BishopCompletionModulus, CauchyModulus, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7214

A Concrete Naming Certificate for $\mathsf{ClopenAlgebra}^{\uparrow}$

Following Stone and Birkhoff on Boolean algebras of clopen sets, the $\mathsf{ClopenAlgebra}^{\uparrow}$ packet records the finite BEDC surface by which a displayed topology row supplies clopen rows together with Boolean operations on those rows. It sits beside the topology carrier of chapter 88, the Boolean algebra carrier of chapter 3733, the Stone-duality clopen ledger of chapter 261, the connectedness refusal surface of chapter 2353, and the total-disconnectedness separation surface of chapter 2271. The packet names a local finite clopen algebra ledger; it does not assert a global powerset algebra, selected clopen basis, ultrafilter representation, host subset extensionality, or excluded-middle separation.

Definition 7214.1 (Clopen algebra carrier)**.** *A* $\mathsf{ClopenAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, O, \bar{O}, B, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row. It records the open-set classifier, displayed empty and whole opens, and finite inclusion requests inspected by the packet.*
2. *O is the displayed open row for each clopen candidate under inspection.*
3. *$\bar{O}$ is the displayed complement-open row attached to the same candidate. A row is admitted as clopen only when both O and $\bar{O}$ are present over the same T.*
4. *B is the* $\mathsf{BooleanAlgebra}^{\uparrow}$ *operation row. It records finite union, intersection, complement, zero, one, and order reads on the admitted clopen rows.*
5. *L is the* $\mathsf{StoneDuality}^{\uparrow}$ *clopen ledger row. It records the same clopen observations as the Stone-duality surface of chapter 261 without adding ultrafilter points or a completed Stone space.*
6. *H records componentwise* hsame *transport for the topology row, open row, complement-open row, Boolean operation row, Stone ledger row, replay, provenance, and local name.*
7. *C records displayed* Cont *replay for the route*

$$T, O, \bar{O} \longmapsto B, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{ClopenAlgebra}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{StoneDuality}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ClopenAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, O, \bar{O}, B, L, H, C, P, N$. It has no coordinate for arbitrary subsets, a selected clopen basis, ultrafilter choice, quotient topology equality, or host powerset Boolean operations.

Theorem 7214.2 (Clopen algebra NameCert obligations)**.** *For the carrier of Definition 7214.1, the local* $\mathsf{NameCert}_{\mathsf{ClopenAlgebra}^{\uparrow}}$ *surface consists exactly of topology-source exposure through T, displayed open-row exposure through O, displayed complement-open exposure through $\bar{O}$, Boolean operation exposure through B, Stone-duality clopen-ledger exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every connectedness or total-disconnectedness consumer reads the clopen algebra only through those rows and receives no excluded-middle split, selected clopen basis, host subset equality, arbitrary powerset algebra, or ultrafilter representation.*

Proof. Carrier admission is projection from Definition 7214.1. The packet displays exactly the topology source, open row, complement-open row, Boolean operation row, Stone clopen ledger, transport, replay, provenance, and local naming row.

The clopen condition is local to $T, O, \bar{O}$. A consumer may call a row clopen only after both the open row and its complement-open row have been displayed over the same topology classifier.

The Boolean operations are then read through B. Union, intersection, complement, zero, one, and order requests are operations on the displayed clopen rows, not on an ambient powerset.

Finally L, H, C, P, N preserve and name the same route. The Stone-duality ledger records the clopen observations already visible in the packet, while transport and replay cannot create a selected basis, ultrafilter point, host subset equality, or external Boolean algebra. □

Theorem 7214.3 (Clopen algebra Boolean-topology handoff)**.** *Every Boolean-topology consumer of an accepted* $\mathsf{ClopenAlgebra}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow O, \bar{O} \longrightarrow \mathsf{BooleanAlgebra}^{\uparrow} \longrightarrow \mathsf{StoneDuality}^{\uparrow} \longrightarrow H, C, P, N.$$

The handoff gives connectedness consumers the displayed clopen-refusal boundary used by $\mathsf{ConnectedSpace}^{\uparrow}$ *and gives total-disconnectedness consumers the displayed clopen-separation ledger used by* $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$ *It gives neither open-cover compactness, nor host subset extensionality, nor an ultrafilter representation, nor a selected global clopen basis.*

Proof. Start with the NameCert obligations of Definition 7214.2. They require every clopen read to expose the topology source together with both the open row and the complement-open row before the Boolean operation row is consumed.

The Boolean handoff is finite. The row B supplies only the displayed operations on admitted clopen rows, and L identifies those rows with the Stone-duality clopen ledger rather than with a completed point space.

Connectedness and total-disconnectedness consume different faces of the same packet. The connectedness side reads clopen attempts as refusal rows, while the total-disconnectedness side reads clopen separations as displayed separation rows.

Since H, C, P, N only transport, replay, source, and name the displayed rows, every Boolean-topology consumer remains inside the route shown above. The excluded powerset, basis-selection, ultrafilter, and host-extensionality objects are absent from the carrier. □

Lean4 (statement) `BEDC.Derived.ClopenAlgebraUp`

Closure status for $\mathsf{NameCert}_{\mathsf{ClopenAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{ClopenAlgebra}^{\uparrow}$ packet is generated from a Topology source row, displayed open and complement-open rows, BooleanAlgebra operation rows, a StoneDuality clopen ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClopenAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7214.1, Definition 7214.2, and Definition 7214.3 seed the finite Boolean-topology clopen algebra ledger over displayed topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClopenAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ClopenAlgebraUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary powerset Boolean algebras, selected clopen bases, ultrafilter representations, host subset extensionality, open-cover compactness, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open and complement-open exposure, Boolean operation exactness, Stone clopen-ledger compatibility, transport, replay, provenance, and local naming for BEDC.Derived.ClopenAlgebraUp.

Chapter 7215

A Concrete Naming Certificate for $\mathsf{DoobMaximalInequality}^{\uparrow}$

Following Doob, the $\mathsf{DoobMaximalInequality}^{\uparrow}$ packet records the finite maximal-window inequality surface used when a bounded martingale prefix is read before any convergence or almost-sure quotient consumer sees it. It sits after the finite martingale carrier of chapter 190, the bounded stopped-window route of chapter 383, the Doob upcrossing ledger of chapter 1630, the convergence-preparation packet of chapter 2246, and the terminal real-name seal of chapter 38. The certificate names only finite probability, martingale, maximal-window, stopping, inequality-bound, Real-boundary, transport, replay, provenance, and local naming rows.

Definition 7215.1 (Doob maximal inequality carrier)**.** *A* $\mathsf{DoobMaximalInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Omega, M, W, A, S, U, B, R, H, C, P, N)$$

with the following visible rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *probability row, carrying only the finite event and expectation observations used by the bound.*
2. M *is the* $\mathsf{Martingale}^{\uparrow}$ *source row, exposing the adapted finite prefix and conditional-expectation faces.*
3. W *is the maximal-window row, recording the displayed finite horizon and the inspected prefix maximum.*
4. A *is the threshold row, naming the positive* $\mathsf{Real}^{\uparrow}$ *boundary at which the maximal event is queried.*
5. S *is the bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *route used to read first-hit and terminal stopped values inside the displayed horizon.*
6. U *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *sibling row, recording that crossing-count control is available through the same martingale and stopped-window surface.*
7. B *is the finite Doob-bound row, relating the maximal event and threshold to the endpoint expectation read.*
8. R *is the* $\mathsf{Real}^{\uparrow}$ *boundary row sealing threshold, expectation, and bound comparisons after the finite rows have been replayed.*

9. *H records componentwise* hsame *transport for the probability, martingale, maximal-window, threshold, optional-stopping, upcrossing, bound, Real-boundary, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through*

$$\Omega, M, W, A, S, U, B, R \longmapsto \mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}.$$

11. *P is the* Pkg *provenance over* $\mathsf{DoobMaximalInequality}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{MartingaleConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a completed filtration, an unbounded stopping time, almost-sure quotient equality, a selected exceptional set, ambient measure completion, or host L^p maximal theorem.

Theorem 7215.2 (Doob maximal inequality NameCert obligations)**.** *For every accepted* $\mathsf{DoobMaximalInequality}^{\uparrow}$ *carrier $D = (\Omega, M, W, A, S, U, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}$ *surface consists exactly of probability exposure, finite martingale-prefix exposure, maximal-window admission, threshold-boundary admission, bounded optional-stopping handoff, Doob-upcrossing compatibility, finite Doob-bound admission,* $\mathsf{Real}^{\uparrow}$ *boundary sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. No obligation is discharged by a completed filtration, an unbounded stopping theorem, almost-sure quotient equality, a selected exceptional set, ambient measure completion, or a host L^p maximal theorem.*

Proof. Project the accepted carrier of Definition 7215.1. The analytic-facing rows are exactly $\Omega, M, W, A, S, U, B, R$, and the structural rows H, C, P, N only transport, replay, source, and name those rows.

The maximal-window row W and threshold row A determine the finite event being bounded. The bounded optional-stopping row S keeps first-hit and terminal reads inside the displayed horizon, while U records that the same martingale surface is compatible with the Doob-upcrossing ledger.

The bound row B and Real-boundary row R are then terminal for the inequality read. Since every comparison passes through R after the finite rows have been replayed, the certificate cannot import completed filtrations, unbounded stopping, almost-sure quotient equality, selected exceptional sets, ambient measure completion, or host maximal theorems. □

Theorem 7215.3 (Doob maximal inequality convergence handoff)**.** *Every accepted* $\mathsf{DoobMaximalInequality}^{\uparrow}$ *carrier routes a convergence-facing maximal read through the finite dependency chain*

$$\mathsf{Martingale}^{\uparrow} \longmapsto \mathsf{OptionalStopping}^{\uparrow} \longmapsto \mathsf{DoobUpcrossing}^{\uparrow} \longmapsto \mathsf{DoobMaximalInequality}^{\uparrow} \longmapsto \mathsf{MartingaleConvergence}^{\uparrow}$$

The exported object is only the displayed maximal-bound row B together with its $\mathsf{Real}^{\uparrow}$ *boundary seal R; it is not a martingale convergence theorem, not an almost-sure maximal process, and not a completed measure-space argument.*

Proof. Begin with the martingale source row M. It supplies the finite adapted prefix and conditional-expectation faces consumed by the maximal-window row W.

Next read the bounded optional-stopping row S and the Doob-upcrossing sibling row U. These rows keep first-hit, terminal, and crossing-control reads inside the same displayed horizon and prevent the maximal event from becoming an unbounded stopping assertion.

Finally expose the bound row B and the Real-boundary row R. A convergence-facing consumer may cite this finite maximal-bound surface before reading chapter 2246, but receives no convergence theorem or almost-sure quotient from this chapter. □

Falsifiable prediction: Every accepted $\mathsf{DoobMaximalInequality}^{\uparrow}$ consumer that reads a maximal-event bound without displaying the probability source, finite martingale prefix, maximal-window row, threshold boundary, bounded $\mathsf{OptionalStopping}^{\uparrow}$ route, $\mathsf{DoobUpcrossing}^{\uparrow}$ compatibility row, finite bound row, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: martingale, optionalstopping, doob-upcrossing, martingale-convergence, real: none is bijective to $\mathsf{DoobMaximalInequality}^{\uparrow}$ because this packet jointly carries the finite maximal-window row, threshold boundary, bounded stopping route, upcrossing compatibility, Doob-bound row, Real seal, transport, replay, provenance, and local naming needed by maximal-inequality consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{DoobMaximalInequality}^{\uparrow}$ packet is generated from one probability row, one finite martingale row, one maximal-window row, one threshold row, one bounded optional-stopping row, one Doob-upcrossing compatibility row, one finite bound row, one RealUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7215.1, Definition 7215.2, and Definition 7215.3 seed the finite maximal-inequality packet over probability, martingale, bounded stopping, Doob-upcrossing, convergence-facing, Real-boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoobMaximalInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filtrations, unbounded stopping, almost-sure quotient equality, selected exceptional sets, ambient measure completion, martingale convergence, host L^p maximal theorems, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, maximal-window exactness, bounded optional-stopping handoff, Doob-upcrossing compatibility, finite bound admission, Real-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DoobMaximalInequalityUp.

Chapter 7216

A Concrete Naming Certificate for $\mathsf{BishopCompletionModulusStability}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionModulusStability}^{\uparrow}$ packet names the finite stability surface for the L10 completion-modulus chain. It sits after the displayed threshold route of chapter 1469 and the selection surface of chapter 7210, reads Cauchy thresholds through chapter 292, opens observation windows through chapter 126, records dyadic tolerance rows through chapter 291, repacks the window through chapter 301, and reaches only the terminal seal of chapter 38. The packet names stability of the displayed finite route; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, or host real equality.

Definition 7216.1 (Bishop completion modulus stability carrier)**.** *A* $\mathsf{BishopCompletionModulusStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, A, n, k, W, D, R, E, H, C, P, N)$$

with the following rows:

1. *M is the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier row whose threshold route is being stabilized.*
2. *A is the* $\mathsf{CauchyModulus}^{\uparrow}$ *source row read by M for the displayed precision request.*
3. *n is the finite precision request and k is the selected threshold row.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window opened at k.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same threshold and window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff obtained from W and D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after M, A, n, k, W, D, R are displayed.*
8. *H records componentwise* hsame *transport for the modulus, source, request, threshold, window, dyadic ledger, handoff, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay of the same route under accepted carrier comparison.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionModulusStability}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusStability}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, countable choice, or host real equality.*

Theorem 7216.2 (Bishop completion modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionModulusStability}^{\uparrow}$ *carrier* $T = (M, A, n, k, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusStability}^{\uparrow}}$ *surface consists exactly of carrier admission, displayed* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *route exposure,* $\mathsf{CauchyModulus}^{\uparrow}$ *source exposure, finite precision and threshold exposure,* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *stability transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The analytic rows are M, A, n, k, W, D, R, E, while H, C, P, N are the structural rows that transport, replay, source, and name the same finite route.

Read the completion-modulus and Cauchy-modulus faces first. The selected threshold k is exposed as a row attached to the request n, and the stream window W is opened only after that threshold is visible.

The dyadic and regular rows are downstream from the same threshold. The ledger D records the tolerance data for W, and the handoff row R repacks W, D before the Real-facing seal E is consumed.

Finally the stability rows H, C, P, N preserve only the displayed coordinates. Since no other carrier row exists, $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, and host real equality cannot be local obligations of the packet. □

Theorem 7216.3 (Bishop completion modulus stability route)**.** *Every accepted* $\mathsf{BishopCompletionModulusStability}^{\uparrow}$ *read preserves the displayed L10 route*

$$\mathsf{BishopCompletionModulus}^{\uparrow}, \mathsf{CauchyModulus}^{\uparrow}, n \longmapsto k \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

If two accepted packets are compared by the classifier, their componentwise transport rows preserve the same threshold, window, dyadic ledger, regular rational handoff, and Real seal before any downstream completion-modulus consumer reads the packet.

Proof. Start with the NameCert obligations of Definition 7216.2. They expose the completion-modulus row, Cauchy-modulus source row, request, threshold, stream window, dyadic ledger, regular handoff, and Real seal as separate finite rows.

Use the scoped dependency surface of Definition 1469.10. It binds the underlying $\mathsf{BishopCompletionModulus}^{\uparrow}$ route to carrier admission, threshold coherence, heredity, Real seal ordering, and same-threshold transport.

Now replay that route inside the stability carrier. The rows H, C, P, N compare only the displayed carrier coordinates, so classifier-related packets keep the threshold, window, dyadic ledger, handoff, and seal aligned.

The excluded principles have no carrier coordinate and no route theorem in the packet. Thus a stability read cannot replace the finite L10 source chain by $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionModulusStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCompletionModulusStability}^{\uparrow}$ packet is generated from a displayed BishopCompletionModulus row, a CauchyModulus source row, a finite precision request, a selected threshold, a StreamName window, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, a RealUp seal, componentwise stability transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopCompletionModulusStability}^{\uparrow}}$, seed)

Scope closed: Definition 7216.1, Definition 7216.2, and Definition 7216.3 seed the finite stability surface for the L10 completion-modulus chain.

Formal verification: FormalStatus(NameCert$_{\text{BishopCompletionModulusStability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold stability, StreamName window readback, DyadicRatCore tolerance routing, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionModulusStabilityUp.

Chapter 7217

A Concrete Naming Certificate for $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive analysis, the $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$ packet records the finite route by which one displayed point defect is repaired before an integration or completion-facing consumer reads the function. It sits beside the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, the continuous-map carrier of chapter 130, the finite integrability witness of chapter 1960, and the Cauchy-sum completion boundary of chapter 5903. The packet repairs only the named finite defect; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, select hidden limits, identify quotient functions, use real trichotomy, invoke countable choice, or prove a full integration theorem.

Definition 7217.1 (Riemann removable-discontinuity carrier)**.** *A* $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, F, A, W, R, E, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite defect row, naming the single point or finite displayed defect window at which the source function is not read through the ordinary continuous-map route.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *off-defect row, exposing the continuous-map evidence on the displayed punctured window or finite complement route.*
3. *A is the approach ledger, recording the visible two-sided or one-sided approximation windows by which the repaired value is compared with nearby off-defect reads.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the samples, punctured-neighbourhood observations, and requested precisions inspected before the repaired value is sealed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, so every numeric comparison has a regular rational source before the terminal real-name seal is read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed replacement value and the repaired function values visible to downstream consumers.*
7. *J is the integration and completion handoff row, allowing* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ *consumers to read the repaired packet only after D, F, A, W, R, E have been displayed.*

8. *H records componentwise* hsame *transport for the defect, off-defect continuity, approach ledger, finite windows, regular readback, Real seal, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a repair request through $D, F, A, W, R, E, J, H, P, N$.*
10. *P is the* Pkg *provenance over* RiemannRemovableDiscontinuity$^\uparrow$, ContinuousMap$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, RiemannIntegrable$^\uparrow$, RiemannIntegrationCompletion$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RiemannRemovableDiscontinuity}^\uparrow}$ *row.*

The classifier compares accepted packets only through $D, F, A, W, R, E, J, H, C, P, N$. It has no coordinate for ambient Real$^\uparrow$ *completeness, selected limits, quotient function equality, real trichotomy, countable choice, a host integration functional, or a full Riemann integration theorem.*

Lean4 (statement) `BEDC.Derived.RiemannRemovableDiscontinuityUp`

Theorem 7217.2 (Riemann removable-discontinuity NameCert obligations)**.** *For every accepted* RiemannRemovableDiscontinuity$^\uparrow$ *carrier*

$$M = (D, F, A, W, R, E, J, H, C, P, N),$$

the local NameCert$_{\text{RiemannRemovableDiscontinuity}^\uparrow}$ *surface consists exactly of carrier admission for the displayed defect row, off-defect continuous-map row, approach ledger, finite StreamName windows, RegSeqRat readback, terminal Real seal, integration and completion handoff, componentwise transport, displayed replay, provenance, and local naming rows.*

The obligation surface requires the defect to be named before the off-defect continuity row is consumed, the approach ledger to expose only finite comparison windows, the StreamName$^\uparrow$ *and* RegSeqRat$^\uparrow$ *rows to precede any terminal* Real$^\uparrow$ *seal, and the handoff row J to export only the repaired finite route to* RiemannIntegrable$^\uparrow$ *or* RiemannIntegrationCompletion$^\uparrow$ *consumers.*

Proof. Project the carrier of Definition 7217.1. The analytic rows are D, F, A, W, R, E, J, and the structural rows are H, C, P, N.

The defect row D fixes the only place where the ordinary continuous-map route is not read directly. The off-defect row F supplies the continuous-map evidence away from that displayed window, while the approach ledger A records the finite comparisons used to justify the replacement value.

The numeric readback is W, R, E. The stream row W lists the inspected windows, the regular rational row R supplies the readback for those windows, and the Real row E seals the displayed replacement value only after those rows are present.

Finally J, H, C, P, N preserve the same finite route for integration and completion consumers. Since no excluded principle is a carrier coordinate, none is a local NameCert obligation of the packet. □

Theorem 7217.3 (Riemann removable-discontinuity completion handoff)**.** *Every* RiemannIntegrable$^\uparrow$ *or* RiemannIntegrationCompletion$^\uparrow$ *consumer of an accepted* RiemannRemovableDiscontinuity$^\uparrow$ *carrier must first read the finite route*

$$D, F, A \longmapsto W, R, E \longmapsto J \longmapsto H, C, P, N.$$

The consumer receives a displayed repaired-function boundary with one named removable defect, finite off-defect continuity evidence, finite approach windows, regular rational readback, and a terminal Real seal. It receives no ambient Real$^\uparrow$ *completeness theorem, no selected limiting value,*

no quotient equality of functions, no real trichotomy, no countable-choice schedule, and no full integration theorem.

Proof. Use Definition 7217.2. It places the defect row, off-defect continuity row, and approach ledger before any consumer-facing handoff.

The completion-facing readback is forced through W, R, E. Thus a requested repaired value is seen through finite StreamName observations and RegSeqRat data before the Real seal is public.

The handoff row J can then be read by $\mathsf{RiemannIntegrable}^{\uparrow}$ or $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ consumers, but only as a boundary row attached to the already displayed repair packet. It does not supply a separate proof of integrability, a Cauchy-sum convergence theorem, or a host integral.

The rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no coordinate for the excluded principles, the completion handoff cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$ read must expose a finite defect row, an off-defect $\mathsf{ContinuousMap}^{\uparrow}$ row, an approach ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, an integration and completion handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: continuousmap, streamname, regseqrat, real, riemann-integrable, riemann-integration-completion: $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$ is not bijective to any listed sibling because it binds a named removable defect, off-defect continuity, finite approach readback, and the integration-completion handoff in one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}$ packet is generated from a finite defect row, an off-defect ContinuousMap row, an approach ledger, StreamName finite windows, RegSeqRat readback, a Real seal row, a RiemannIntegrable and RiemannIntegrationCompletion handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7217.1, Definition 7217.2, and Definition 7217.3 seed the finite removable-discontinuity repair surface over $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannRemovableDiscontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient equality of functions, real trichotomy, countable choice, or a full integration theorem.

Upgrade path: Obligation closure requires checked carrier admission, defect exposure, off-defect continuity, approach-ledger exactness, StreamName and RegSeqRat readback, Real-seal nonescape, integration-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRemovableDiscontinuityUp.

Chapter 7218

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ packet records the finite domination route by which one regular-Cauchy modulus may be replaced by a cofinal dominating modulus without changing the displayed tail-window readback. It sits after the completion-modulus packet of chapter 1469, the Cauchy threshold surface of chapter 292, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational handoff of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a finite L10 route for changing modulus representatives along a displayed tail comparison; it does not select limits, quotient regular Cauchy names, countable-choice schedules, host real equality, or ambient completeness.

Definition 7218.1 (Regular Cauchy modulus domination carrier)**.** *A* $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M_0, M_1, D, W_0, W_1, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. M_0 *is the source* $\mathsf{CauchyModulus}^{\uparrow}$ *row attached to the original regular-Cauchy tail request.*
2. M_1 *is the dominating* $\mathsf{CauchyModulus}^{\uparrow}$ *row whose displayed threshold is used by the replacement route.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, certifying that* M_1 *dominates* M_0 *on the displayed tail and tolerance scale.*
4. W_0 *and* W_1 *are the corresponding* $\mathsf{StreamName}^{\uparrow}$ *finite windows read from the source modulus and the dominating modulus.*
5. T *is the cofinal tail comparison row, recording that every source-tail read used by* M_0 *is met by a dominating-tail read through* M_1*.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the dominated window comparison and the dyadic ledger.*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the two moduli, the domination ledger, both windows, the cofinal tail row, and the regular readback have been displayed.*

8. *H stores componentwise* hsame *transport for the two modulus rows, domination ledger, windows, cofinal tail comparison, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M_0, M_1, D \longmapsto W_0, W_1, T \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusDomination}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M_0, M_1, D, W_0, W_1, T, R, E, H, C, P, N$. It has no coordinate for an infinite modulus choice function, selected limit, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 (statement) `BEDC.Derived.RegularCauchyModulusDominationUp`

Theorem 7218.2 (Regular Cauchy modulus domination NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ *carrier $K = (M_0, M_1, D, W_0, W_1, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusDomination}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; source and dominating modulus exposure; dyadic domination through D; paired StreamName window exposure through W_0, W_1; cofinal tail comparison through T; RegSeqRat handoff through R; RealUp seal non-escape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier. The tuple in Definition 7218.1 lists exactly the two modulus rows, the dyadic domination ledger, the two window rows, the cofinal tail comparison, the regular readback, the Real seal, and the structural rows.

The domination obligation is forced by the order of the rows. A consumer first reads M_0 and M_1, then reads D as the finite certificate that the displayed M_1 threshold dominates the displayed M_0 threshold on the inspected tail.

The handoff rows are downstream. The windows W_0, W_1 and the cofinal comparison T supply the only route to R, and E is available only after R. Transport, replay, provenance, and naming preserve the same packet, so no unlisted modulus selector or completion principle can be exported. □

Theorem 7218.3 (Regular Cauchy modulus domination tail stability)**.** *Let $K = (M_0, M_1, D, W_0, W_1, T, R, E, H, C$ be an accepted* $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ *carrier. Any later tail request transported by the displayed* hsame *rows and still certified by the domination ledger D reads the same cofinal comparison row T, the same dominating window route W_1, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R. Thus replacing M_0 by the dominating modulus M_1 is stable for every Real-facing read that remains inside the displayed tail.*

Proof. Start with the transported tail request. Componentwise hsame transport preserves the visible modulus, ledger, window, cofinal comparison, readback, and seal rows rather than creating a second tail source.

Read the dyadic domination ledger next. Since D is still the certificate for the transported request, the route from the source window to the dominating window must pass through T and then through W_1.

Finally replay the continuation route. The row R is downstream from the two windows and T, and the Real seal E is downstream from R. Hence every transported Real-facing read stays on the original dominated tail route. □

Theorem 7218.4 (Regular Cauchy modulus domination Real seal route)**.** *Every accepted* RegularCauchyModulusDo *carrier routes a Real-completion consumer through the finite chain*

$$M_0, M_1, D \longmapsto W_0, W_1, T \longmapsto R \longmapsto E.$$

The terminal $\mathsf{Real}^{\uparrow}$ *seal records that the dominated cofinal tail window has already been repacked as a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. It exports no selected real limit, quotient regular Cauchy equality, hidden modulus choice function, countable choice, host real equality, or ambient completeness principle.*

Proof. The displayed Cont row supplies the chain. A Real-completion consumer must first expose the two modulus rows and the domination ledger, then the paired windows and cofinal tail comparison.

The regular rational handoff is the next visible row. By the carrier definition, R is obtained only after W_0, W_1, T have been displayed, so the handoff cannot be read from an unregistered modulus or a hidden tail selector.

The Real seal is terminal. Since E appears after R and the carrier contains no selected limit, quotient equality, choice schedule, host equality, or completeness coordinate, the seal exports only the finite dominated-tail route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying M_0, M_1, the dyadic domination ledger D, both $\mathsf{StreamName}^{\uparrow}$ windows, the cofinal tail comparison T, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows refutes the packet.

Independence witness: bishopcompletionmodulus, cauchymodulus, streamname, dyadicratcore, regseqrat, real: no listed sibling is bijective with $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ because this packet jointly carries two modulus rows, a domination ledger, paired windows, cofinal tail comparison, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusDomination}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyModulusDomination}^{\uparrow}$ packet is generated from two CauchyModulus rows, a DyadicRatCore domination ledger, paired StreamName windows, a cofinal tail comparison row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusDomination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7218.1, Definition 7218.2, Definition 7218.3, and Definition 7218.4 seed the finite domination route between source and dominating regular-Cauchy moduli over displayed tail windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusDomination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient regular Cauchy equality, infinite modulus choice, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic domination, paired window exposure, cofinal tail comparison, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyModulusDominationUp.

Chapter 7219

A Concrete Naming Certificate for $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$

The $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ packet realizes one bounded MetaCIC confluence frontier from section 9386.3 as finite residual-pair local-diamond data. It sits after the residual-substitution boundary of chapter 2181, the bounded residual-substitution frontier of chapter 3808, the residual-checker packet of chapter 3827, the typed residual-budget packet of chapter 7088, and the closed parallel-diamond obstruction of chapter 6169. The packet names a bounded local-diamond witness for a displayed residual pair; it does not claim full closed confluence, arbitrary parallel diamond, unrestricted substitution compatibility, global subject reduction, or a completed Church–Rosser theorem.

Definition 7219.1 (MetaCIC residual diamond witness carrier)**.** *A* $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, F, L, S, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, recording the bounded closed MetaCIC source, type window, and residual-pair observation range.*
2. *F is the residual-frontier row, naming the two residual steps whose local join is being checked.*
3. *L is the bounded local-join checker row, recording the finite local-diamond search and its accepted join endpoint for this residual pair.*
4. *S is the substitution-boundary row, exposing the exact residual-substitution interface consumed by the checker.*
5. *O is the obstruction-retention row, recording the remaining frontier outside this bounded residual pair.*
6. *H stores componentwise* hsame *transport for the typed window, residual frontier, local checker, substitution boundary, obstruction row, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the route*

$$T, F, S \longmapsto L \longmapsto O.$$

8. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$*, residual-substitution boundary data, bounded checker data, typed-window data,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, F, L, S, O, H, C, P, N$. It has no coordinate for full confluence, arbitrary residual closure, unrestricted substitution compatibility, subject-reduction discharge, hidden evaluator state, or host syntax quotienting.

Lean4 (statement) `BEDC.Derived.MetaCICResidualDiamondWitnessUp`

Theorem 7219.2 (MetaCIC residual diamond witness obligations)**.** *For every accepted* MetaCICResidualDiamondW *carrier $J = (T, F, L, S, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}}$ *surface consists exactly of typed-window exposure, residual-frontier exposure, bounded local-join checking, substitution-boundary exposure, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is discharged by full confluence, arbitrary parallel diamond, global substitution compatibility, subject reduction, hidden evaluator state, or host syntax quotienting.*

Proof. Project the carrier. The analytic rows are T, F, L, S, O, and the structural rows H, C, P, N only transport, replay, source, and name those rows.

The local-diamond witness is bounded. The residual frontier F and typed window T determine the residual pair under inspection, while S exposes the substitution boundary available to the checker.

The checker row L records only the finite local join for that pair. The obstruction row O remains visible, so the packet cannot be read as an arbitrary confluence theorem or as unrestricted substitution compatibility. □

Theorem 7219.3 (MetaCIC residual diamond witness local join route)**.** *Every accepted* MetaCICResidualDiamondW *carrier routes its bounded local join through*

$$T, F, S \longmapsto L \longmapsto O.$$

The row L is available only after the typed window, residual frontier, and substitution boundary have been displayed, and the row O records that the surrounding confluence frontier remains outside this bounded witness.

Proof. Read the typed window and residual frontier first. These rows fix the source, the observed residual pair, and the finite range over which the checker may operate.

Next read the substitution boundary. The checker receives exactly the boundary row S and no hidden substitution theorem, evaluator state, or subject-reduction discharge.

Finally expose the local checker and obstruction rows. The row L supplies the bounded join for the displayed pair, while O retains the unclosed frontier beyond that pair. □

Theorem 7219.4 (MetaCIC residual diamond witness transport stability)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ *carrier preserves the typed window, residual frontier, substitution boundary, bounded local-join checker, and obstruction-retention rows as one finite witness packet. A transported consumer may replay the local diamond route, but it cannot strengthen the packet into full confluence or erase the retained obstruction boundary.*

Proof. Transport acts componentwise on the displayed rows. Hence the typed window, residual pair, substitution boundary, checker result, obstruction row, replay, provenance, and local naming remain the classifier-visible coordinates.

Replay then follows the same continuation route. The transported checker row is still downstream from the typed window, residual frontier, and substitution boundary, and the obstruction row is still downstream from the checker.

The excluded endpoints are absent from the carrier. Since there is no row for arbitrary confluence, unrestricted substitution compatibility, subject reduction, hidden evaluator state, or host quotienting, transport cannot manufacture one. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ consumer that reads a bounded local join without displaying the typed window, residual frontier, substitution boundary, bounded checker, obstruction-retention row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: metacic-residual-substitution-boundary, bounded-residual-substitution-frontier, metacic-confluence-residual-checker, metacic-typed-residual-budget, closed-parallel-diamond-obstruction: no listed sibling is bijective with $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ because this packet jointly carries a typed window, residual pair, bounded local-join checker, substitution boundary, retained obstruction, transport, replay, provenance, and local naming.

The Lean TasteGate for $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ will include Nontrivial and Field-Faithful instances for the typed-window row, residual-frontier row, substitution-boundary row, bounded local-join checker, and obstruction-retention row, so a future checked surface can distinguish a genuine local witness from a carrier-only smoke packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$ packet is generated from a typed-window row, a residual-frontier row, a bounded local-join checker, a substitution-boundary row, an obstruction-retention row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7219.1, Definition 7219.2, Definition 7219.3, and Definition 7219.4 seed one bounded residual-pair local-diamond witness for the MetaCIC confluence frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full closed confluence, arbitrary parallel diamond, unrestricted substitution compatibility, subject reduction, hidden evaluator state, host syntax quotienting, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, residual-frontier exposure, bounded local-join checking, substitution-boundary exposure, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualDiamondWitnessUp.

Chapter 7220

A Concrete Naming Certificate for MetaCICCriticalOverlapEnvelope$^{\uparrow}$

The MetaCICCriticalOverlapEnvelope$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as finite candidate-overlap envelope data. It sits beside the parallel-diamond frontier of chapter 4238, the confluence diamond ledger of chapter 6547, the residual-substitution boundary packets cited in the same audit map, the normalization frontier of chapter 892, and the critical-path packet of chapter 765. The envelope names bounded local overlap rows only; it does not promote those rows to full confluence, kernel consistency, decidable conversion, or closed-normal completeness.

Definition 7220.1 (MetaCIC critical overlap envelope carrier)**.** *A* MetaCICCriticalOverlapEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, K, J, R, C, B, N, O, H, Q, P, L)$$

with the following displayed rows:

1. *T is the typed-window row, recording the bounded context, source term, target term window, and typing side conditions under inspection.*
2. *K is the critical-pair row, recording the two local reduction choices and the overlap cell where they compete.*
3. *J is the candidate-join row, recording the proposed common reducts and the finite route by which each branch reaches them.*
4. *R is the residual-frontier row, recording the substitution-sensitive residual data that remains visible after the candidate join is proposed.*
5. *C is the closed-checker row, recording whether the bounded window is closed enough for the closed normal and atom fragments cited by the confluence audit map.*
6. *B is the bounded normal/check row, recording finite fuel, normal-form comparison, and local conversion checks for the candidate envelope.*
7. *N is the bounded normal endpoint row, recording any finished-normal or atom-fragment endpoint used by the checker.*
8. *O is the obstruction row, retaining the residual-substitution, closed-diamond, or local-check failure when the candidate envelope cannot discharge the overlap.*

9. *H records componentwise* hsame *transport for typed windows, critical pairs, candidate joins, residual frontiers, closed checks, bounded normal rows, obstruction rows, replay, provenance, and local naming.*
10. *Q records displayed* Cont *replay from a confluence-overlap request through $T, K, J, R, C, B, N, O, H, P, L$.*
11. *P is the* Pkg *provenance over* MetaCICCriticalOverlapEnvelope$^{\uparrow}$, MetaCICRedexFrontier$^{\uparrow}$, MetaCICNormalizationFr MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalOverlapEnvelope}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $T, K, J, R, C, B, N, O, H, Q, P, L$. No coordinate stores full confluence, subject reduction, kernel consistency, unbounded proof search, decidable conversion, or a hidden substitution oracle.

Theorem 7220.2 (MetaCIC critical overlap envelope obligations)**.** *For every accepted* MetaCICCriticalOverlapEnvel *carrier $E = (T, K, J, R, C, B, N, O, H, Q, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalOverlapEnvelope}^{\uparrow}}$ *surface consists exactly of typed-window admission through T; critical-pair visibility through K; candidate-join exposure through J; residual-frontier retention through R; closed-checker visibility through C; bounded normal and check rows through B, N; obstruction retention through O; and componentwise transport, replay, provenance, and local naming through H, Q, P, L.*

Proof. Project the carrier of Definition 7220.1. The typed window T, critical pair K, candidate join J, and residual frontier R are the only rows that describe the local overlap geometry.

Read the checker rows next. The closed-checker row C and bounded normal/check rows B, N record finite inspection of the proposed envelope, while the obstruction row O keeps any residual-substitution or closed-diamond failure visible rather than erasing it.

Finally use the structural rows H, Q, P, L. They transport, replay, source, and name the same bounded envelope, so the accepted packet exports only the listed finite rows and not full confluence, kernel consistency, unbounded proof search, decidable conversion, or a hidden substitution oracle. □

Falsifiable prediction: Any confluence consumer omitting typed-window, candidate-join, residual-frontier, obstruction, transport, replay, provenance, or local naming rows is not an accepted MetaCICCriticalOverlapEnvelope$^{\uparrow}$ packet.

Independence witness: metacic-parallel-diamond-frontier, metacicconfluenceauditpacket, metacicnormalizationfrontier, metaciccriticalpath: the parallel-diamond frontier records a broader premise frontier without this packet's typed-window candidate envelope, the confluence audit packet records map-level boundary status without bounded overlap rows, the normalization frontier records candidate evidence and closed-normal use without confluence critical pairs, and the critical-path packet records programme routing without local candidate joins. None is bijective to MetaCICCriticalOverlapEnvelope$^{\uparrow}$.

The Lean follow-up must provide Nontrivial MetaCICCriticalOverlapEnvelope$^{\uparrow}$ and FieldFaithful MetaCICCriticalOv instances in `BEDC.Derived.MetaCICCriticalOverlapEnvelopeUp.TasteGate`. Those instances must preserve $T, K, J, R, C, B, N, O, H, Q, P, L$ as distinct finite rows before a checked carrier-alignment theorem can serve as the public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalOverlapEnvelope}^{\uparrow}}$.

Constructive story: A MetaCICCriticalOverlapEnvelope$^{\uparrow}$ packet is generated from typed-window, critical-pair, candidate-join, residual-frontier, closed-checker, bounded normal/check, endpoint, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICCriticalOverlapEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 7220.1 and Definition 7220.2 seed the finite bounded-overlap envelope over typed windows, critical pairs, candidate joins, residual frontiers, check rows, obstruction retention, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICCriticalOverlapEnvelope}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full MetaCIC confluence, subject reduction, kernel consistency, unbounded proof search, decidable conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window admission, critical-pair visibility, candidate-join exposure, residual-frontier retention, checker rows, obstruction retention, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MetaCICCriticalOverlapEnvelopeUp.

Chapter 7221

A Concrete Naming Certificate for $\mathsf{SuslinLine}^{\uparrow}$

Following Kelley and Engelking's treatment of ordered spaces and separability, the $\mathsf{SuslinLine}^{\uparrow}$ packet records the traditional Suslin-line boundary as finite order-topology NameCert data. It reads the total-order source of chapter 54, the topology surface of chapter 88, the ordered-real comparison of chapter 6850, and the separability boundary of chapter 5718. The packet names a dense linear order without endpoints, the order-topology base read, a countable-chain-condition row, a nonseparability boundary row, transport, replay, provenance, and local naming; it does not construct a host Suslin line, decide Suslin's hypothesis, import a choice-selected dense set, or identify the line with the Real carrier.

Definition 7221.1 (Suslin line carrier)**.** *A* $\mathsf{SuslinLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, T, C, B, H, K, P, N)$$

with the following displayed rows:

1. *O is the total-order source row, naming a dense linear order without endpoints and exposing the comparison ledger inherited from chapter 54.*
2. *T is the topology row, recording the order-topology base and its local open-interval read through chapter 88.*
3. *C is the countable-chain-condition row, stating that pairwise-disjoint nonempty open intervals are admitted only through the displayed countability ledger.*
4. *B is the separability-boundary row, recording that no countable dense subset is exported by the packet.*
5. *H stores componentwise* hsame *transport for the order source, topology row, countable-chain-condition row, separability-boundary row, continuation, provenance, and local naming rows.*
6. *K records displayed* Cont *replay from order and topology requests through O, T, C, B.*
7. *P is the* Pkg *provenance over* $\mathsf{SuslinLine}^{\uparrow}$*, the total-order, topology, ordered-real, separability,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SuslinLine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through O, T, C, B, H, K, P, N. It has no coordinate for host set-theoretic existence, a forcing extension, a selected dense enumeration, quotient Real equality, or a proof or refutation of Suslin's hypothesis.

Theorem 7221.2 (Suslin line NameCert obligations)**.** *For every accepted* $\mathsf{SuslinLine}^{\uparrow}$ *carrier* $S = (O, T, C, B, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SuslinLine}^{\uparrow}}$ *surface consists exactly of carrier admission for the total-order source, order-topology base, countable-chain-condition row, separability-boundary row, transport, replay, provenance, and local naming rows; order visibility through O; topology visibility through T; countable-chain-condition visibility through C; nonseparability-boundary visibility through B; componentwise transport through H; displayed continuation through K; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7221.1. The accepted inventory is exactly O, T, C, B, H, K, P, N, so every consumer must read the order source, topology row, countable-chain-condition row, and separability-boundary row before the local name is available.

The sibling anchors supply only their displayed surfaces. The total-order and topology chapters provide comparison and open-interval data, the ordered-real chapter supplies a comparison reference rather than an identification with $\mathsf{Real}^{\uparrow}$, and the separable-space chapter provides the boundary against a displayed countable dense subset.

Rows H, K, P, N preserve and replay that same finite route. Since no carrier coordinate names a host construction, forcing extension, selected dense enumeration, quotient Real equality, or set-theoretic decision principle, none of those endpoints can be exported by the NameCert obligations. □

Theorem 7221.3 (Suslin line separability boundary)**.** *Every topology, total-order, or ordered-real consumer of an accepted* $\mathsf{SuslinLine}^{\uparrow}$ *carrier must read the separability-boundary row B after the displayed order, topology, and countable-chain-condition rows O, T, C. The packet therefore records the Suslin-line boundary as a finite order-topology naming certificate: countable-chain-condition data may be consumed as a displayed row, but no countable dense subset, selected dense enumeration, quotient Real equality, host Suslin-line construction, or decision of Suslin's hypothesis is exported.*

Proof. Start with Definition 7221.2. It requires the order row O, topology row T, and countable-chain-condition row C before the separability-boundary row can be consumed.

Now compare with the separable-space anchor. The row B records the absence of an exported countable dense subset in the same finite packet that carries the order topology and countable-chain-condition rows, so a consumer cannot replace the boundary by a selected dense enumeration.

Finally apply the structural rows of Definition 7221.1. Transport, replay, provenance, and local naming preserve only the displayed order-topology packet, and they add no host construction, quotient Real equality, forcing decision, or ambient choice principle. □

Falsifiable prediction: Every accepted $\mathsf{SuslinLine}^{\uparrow}$ topology-facing read factors through a total-order source row, topology row, countable-chain-condition row, separability-boundary row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: totalorder, topology, linearly-ordered-real, separable-space: $\mathsf{SuslinLine}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a dense linear-order source, order-topology base, countable-chain-condition row, nonseparability-boundary row, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SuslinLine}^{\uparrow}}$.

Constructive story: A SuslinLine$^{\uparrow}$ packet is generated from a displayed total-order source, order-topology row, countable-chain-condition row, separability-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SuslinLine}\uparrow}$, seed)

Scope closed: Definition 7221.1, Definition 7221.2, and Definition 7221.3 seed the finite Suslin-line order-topology packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{SuslinLine}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Suslin's hypothesis, host construction of a Suslin line, forcing independence, a choice-selected dense subset, quotient Real equality, or ambient set-theoretic completeness.

Upgrade path: Obligation closure requires checked carrier admission, order-source discipline, topology-base exactness, countable-chain-condition visibility, separability-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SuslinLineUp.

Chapter 7222

A Concrete Naming Certificate for $\mathsf{DyadicBracketing}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{DyadicBracketing}^{\uparrow}$ packet records the finite bracket data that feeds L10 real names before any completion-facing consumer reads a $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic arithmetic surface of chapter 291, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the located interval surface of chapter 1709, and the terminal real-name seal of chapter 38. The packet names finite lower and upper dyadic brackets with tolerance and nesting rows; it is not a quotient real construction, a selected completed limit, trichotomy on arbitrary reals, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7222.1 (Dyadic bracketing carrier)**.** *A* $\mathsf{DyadicBracketing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, G, S, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
2. *U is the upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
3. *G is the finite gap and tolerance ledger, recording $L \leq U$ and the dyadic scale at which the bracket is accepted.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose observations request and replay the bracket.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed endpoints, gap ledger, and stream window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback row R is visible.*
7. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row that packages the same bracket as a located endpoint interval.*
8. *H records componentwise* hsame *transport for endpoint, gap, stream, readback, seal, interval, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through L, U, G, S, I to R and then to E.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicBracketing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBracketing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a chosen real representative, arbitrary real trichotomy, countable choice, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicBracketingUp.DyadicBracketingTasteGate single carrier alignment`

Theorem 7222.2 (Dyadic bracketing NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBracketing}^{\uparrow}$ *carrier*

$$B = (L, U, G, S, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicBracketing}^{\uparrow}}$ *surface consists exactly of lower endpoint exposure, upper endpoint exposure, dyadic gap and tolerance compatibility, finite stream-window readback, located-interval packaging, regular rational handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every Real or Completion consumer must read the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows before treating the bracket as a real-name object.*

Proof. Project the carrier of Definition 7222.1. The endpoint rows L, U and the gap ledger G are the only dyadic data visible to the local classifier.

Read the observation and interval rows next. The stream window S requests the bracket at a finite tolerance, and the located interval row I repackages the same displayed endpoints without introducing a host interval object.

The consumer-facing handoff is downstream. The row R is the regular rational readback from L, U, G, S, I, and the $\mathsf{Real}^{\uparrow}$ seal E is exposed only after R.

Finally the structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the excluded quotient, selected-limit, trichotomy, choice, and completeness data are not carrier coordinates, the listed obligations exhaust the local NameCert surface. □

Theorem 7222.3 (Dyadic bracketing finite-bracket induction)**.** *Let* $B = (L, U, G, S, R, E, I, H, C, P, N)$ *be an accepted* $\mathsf{DyadicBracketing}^{\uparrow}$ *carrier. Any finite refinement history of the displayed bracket preserves the same route*

$$L, U, G, S, I \longmapsto R \longmapsto E$$

by induction on the bracket ledger. The base bracket reads the displayed lower and upper dyadic endpoints; each successor refinement tightens the gap ledger through $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, replays the finite* $\mathsf{StreamName}^{\uparrow}$ *window, preserves the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *packaging, and leaves the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the accepted carrier. The rows L, U, G, S, I are visible before the regular rational readback and the Real seal, so the initial bracket is a finite dyadic object rather than a completed real.

Induct on the finite bracket ledger recorded by G. In the base case the ledger contains the displayed endpoint order and tolerance scale, and the stream window S reads that bracket directly.

For a successor refinement, the new endpoint rows are obtained through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ comparisons in G. The stream window is replayed through S, and the located interval row I records the same refined endpoints as a finite interval package.

Apply Definition 7222.2 to the consumer-facing rows. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R remains downstream of the bracket ledger, and E remains downstream of R, so the induction cannot export quotient real equality, selected limits, trichotomy, countable choice, or ambient completeness. □

Theorem 7222.4 (Dyadic bracketing L10 real-name handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{DyadicBracketing}^{\uparrow}$ *carrier must pass through the finite dyadic bracket rows before reading the regular rational handoff and terminal real-name seal. In carrier notation, the dependency order is* L, U, G, S, I *before* R*, and* R *before* E*, with* H, C, P, N *preserving the same route under transport, replay, provenance, and local naming.*

Proof. Start from the finite-bracket induction of Definition 7222.3. It keeps every accepted refinement inside the displayed dyadic endpoint, gap, stream-window, and located-interval rows.

Use the obligation theorem Definition 7222.2. It identifies the regular rational readback as a handoff from those rows rather than as a hidden Cauchy quotient or selected limit.

The terminal row is therefore ordered after R. Componentwise transport and continuation replay preserve this order, and the provenance and local naming rows do not add any alternate Real-facing coordinate. □

Falsifiable prediction: Every accepted $\mathsf{DyadicBracketing}^{\uparrow}$ Real-facing read factors through lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, a finite gap ledger, a $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{LocatedRealInterval}^{\uparrow}$ package, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, locatedrealinterval, real: $\mathsf{DyadicBracketing}^{\uparrow}$ is not bijective to any listed sibling because it binds two dyadic endpoints, one finite gap ledger, a stream window, located interval packaging, regular readback, Real seal, transport, replay, provenance, and local naming as one bracket certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBracketing}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicBracketing}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, a finite gap and tolerance ledger, StreamName windows, LocatedRealInterval packaging, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBracketing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7222.1, Definition 7222.2, Definition 7222.3, and Definition 7222.4 seed the finite dyadic bracketing route feeding L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBracketing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected completed limits, arbitrary real trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, dyadic gap compatibility, stream-window readback, located-interval packaging, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicBracketingUp.

Chapter 7223

A Concrete Naming Certificate for $\mathsf{CompactOscillationEnvelope}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor finite-net reading of compact uniform continuity, the $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ packet records a finite oscillation envelope before any uniform-modulus consumer reads its tolerance output. It sits after compact finite-net data from chapter 60, metric distance rows from chapter 57, pointwise continuity rows from chapter 59, rational and dyadic tolerance ledgers from chapter 291, Real-facing bounds from chapter 38, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. The packet is a finite envelope over displayed compact-net cells and oscillation bounds; it is not open-cover compactness, semantic continuity, an external modulus selector, quotient point equality, countable choice, or ambient completeness.

Definition 7223.1 (Compact oscillation envelope carrier)**.** *A* $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, M, F, D, R, B, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, displaying finitely many source cells and coverage witnesses.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare source points, centers, and target values.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise oscillation row attached to each displayed net cell.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested oscillation scale and cell radii.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing distance and oscillation reads.*
6. *B is the finite envelope row, recording the maximum displayed cell oscillation and the lower-bound fold used by the packet.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported only after K, M, F, D, R, B have been read.*
8. *H records componentwise* hsame *transport for compact-net, metric, continuity, dyadic, Real, envelope, uniform-modulus, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through the finite net, local oscillation rows, metric estimates, dyadic ledger, Real seals, and envelope fold to the uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationEnvelope}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host compactness, semantic continuity outside the displayed pointwise rows, an external selector for a modulus, quotient point equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactOscillationEnvelopeUp.CompactOscillationEnvelopeTasteGate single carrier alignment`

Theorem 7223.2 (Compact oscillation envelope NameCert obligations)**.** *For every accepted* CompactOscillationEnv *carrier*

$$O = (K, M, F, D, R, B, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationEnvelope}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, metric distance-read admission, pointwise oscillation admission, dyadic tolerance compatibility, Real-seal nonescape, finite envelope fold exactness, UniformModulus handoff ordering, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7223.1. The analytic rows K, M, F, D, R, B, U and the structural rows H, C, P, N are the only classifier-visible data.

The compact and continuity faces are finite. The row K supplies the displayed net cells, while F supplies pointwise oscillation rows on those cells and M supplies the metric reads used by the triangle estimates.

The tolerance and envelope rows are also displayed. The dyadic ledger D records the requested scale, the Real row R seals only the exposed distance and oscillation reads, and B folds those finitely many cell bounds into one envelope.

Finally U is downstream of the envelope. The rows H, C, P, N transport, replay, source, and name the same packet, and no excluded host compactness, semantic continuity, selector, quotient equality, choice, or completeness object is a carrier coordinate. □

Theorem 7223.3 (Compact oscillation envelope uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ *carrier must first read the finite route*

$$K, M, F, D, R, B \longmapsto U.$$

Thus compact finite-net rows, metric reads, pointwise oscillation rows, dyadic tolerance ledgers, Real-facing bounds, and the finite envelope fold are visible before any uniform-modulus row is exported. The handoff supplies a tolerance-only modulus packet and not host compactness, semantic continuity, quotient point equality, countable choice, or ambient completeness.

Proof. Start with the carrier inventory. The row K exposes finite compact-source coverage, and the rows M, F expose only displayed metric and pointwise oscillation data on that source.

Read the tolerance and bound rows next. The dyadic ledger D names the requested oscillation scale, while R seals the Real-facing distance and oscillation reads used by the envelope.

The finite envelope row B folds those displayed bounds before any uniform-modulus handoff is available. Hence U is downstream of K, M, F, D, R, B.

Apply the nonescape part of Definition 7223.2. Transport and replay preserve the finite route, and the packet has no row for the refused host or choice principles. □

Theorem 7223.4 (Compact oscillation envelope finite-cell stability)**.** *Let $O = (K, M, F, D, R, B, U, H, C, P, N)$ be an accepted* $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ *carrier. Restricting to any displayed finite subfamily of the compact-net cells preserves the same oscillation-envelope route through M, F, D, R, B before the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row U is consumed. Componentwise transport may compare the restricted route, but it cannot introduce a second modulus source or an external compactness theorem.*

Proof. Project the finite net row K. A displayed subfamily is still a finite list of cells from the same compact-source packet.

Restrict the pointwise and metric rows to that subfamily. The rows M and F supply exactly the metric reads and local oscillation rows attached to those cells, while D keeps the same tolerance scale.

The envelope row B is then read on the restricted finite family, and the Real row R seals only those displayed bounds. The handoff theorem Definition 7223.3 keeps U downstream of that envelope route.

The structural rows H, C, P, N preserve, replay, source, and name the same finite data. Since no carrier coordinate stores a second selector or host compactness theorem, finite-cell restriction remains internal to the packet. □

Falsifiable prediction: Every accepted $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ UniformModulus-facing read factors through CompactUp finite-net rows, MetricUp distance reads, ContinuousUp pointwise oscillation rows, DyadicRatCoreUp tolerance data, RealUp bounds, a finite envelope fold, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact, metric, continuous, uniformmodulus, real, dyadicratcore: $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ is not bijective to any listed sibling because it binds compact finite-net cells, metric reads, pointwise oscillation rows, dyadic tolerances, Real bounds, one finite envelope fold, UniformModulus handoff, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationEnvelope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactOscillationEnvelope}^{\uparrow}$ packet is generated from CompactUp finite-net rows, MetricUp distance reads, ContinuousUp pointwise oscillation rows, DyadicRatCoreUp tolerance ledgers, RealUp bounds, a finite envelope fold, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOscillationEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7223.1, Definition 7223.2, Definition 7223.3, and Definition 7223.4 seed the finite oscillation-envelope route for compact uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOscillationEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, semantic continuity, external modulus selection, quotient point equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, metric reads, pointwise oscillation rows, dyadic tolerance compatibility, Real-seal nonescape, finite envelope fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationEnvelopeUp.

Chapter 7224

A Concrete Naming Certificate for BishopIntervalNestingMetricCompletion$^{\uparrow}$

Following Bishop and Bridges on nested intervals and constructive metric completion, the BishopIntervalNestingMetr packet records the handoff from displayed interval nesting to a metric-completion-facing real name as finite BEDC data. It sits after the located nested-interval route of chapter 1713, the dyadic arithmetic core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet names only the displayed nesting-to-completion carrier; it does not assert ambient Real$^{\uparrow}$ completeness, choose an interval intersection point, quotient endpoint names, or select a metric-completion representative.

Definition 7224.1 (Bishop interval nesting metric-completion carrier)**.** *A* BishopIntervalNestingMetricCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, S, N, W, R, M, E, H, C, P, L)$$

with the following visible rows:

1. *I is the located interval row, displaying lower and upper endpoint witnesses and the locatedness comparison needed for retained windows.*
2. *D is the dyadic bisection and midpoint row, read through* DyadicRatCore$^{\uparrow}$ *and carrying the parent-to-child radius-halving ledger.*
3. *S is the shrinking-diameter budget row recording the finite request-to-radius schedule.*
4. *N is the nested containment ledger, including the retained child relation and the proof that later windows remain inside earlier windows.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row for the retained interval sequence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the stream window and dyadic tolerance data have been displayed.*
7. *M is the* MetricCompletion$^{\uparrow}$ *handoff row through which completion-facing consumers read the same regular Cauchy route without choosing a quotient representative.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row, downstream of W, R, M.*
9. *H records componentwise* hsame *transport for the interval, bisection, diameter, nesting, stream, readback, completion, and seal rows.*

10. *C records displayed* Cont *replay from a precision request through retained windows and into completion-facing readback.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}$, $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *L is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an ambient completeness theorem, a selected interval-intersection point, quotient equality of endpoint names, countable choice, trichotomy, or a detached real seal.

Theorem 7224.2 (Bishop interval nesting metric-completion obligations)**.** *For every accepted* $\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}$ *carrier* $K = (I, D, S, N, W, R, M, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the displayed located interval, dyadic midpoint, shrinking-diameter, containment, stream, readback, completion, real-seal, transport, replay, provenance, and local naming rows;*

2. *dyadic contraction, requiring each retained child interval to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and radius-halving ledger before the next window is admitted;*

3. *nested containment, requiring the row N to display that every later retained interval remains inside the earlier window named by the same stream prefix;*

4. *regular readback, requiring W and R to expose the retained windows as a* $\mathsf{RegSeqRat}^{\uparrow}$ *route before M or E is read;*

5. *metric-completion handoff, requiring every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer to read I, D, S, N, W, R on the same packet before using M;*

6. *real-seal non-escape, requiring E to be downstream of the displayed stream, regular-readback, and completion rows and excluding all hidden completeness or quotient coordinates.*

Proof. Project the carrier of Definition 7224.1. This exposes exactly the rows $I, D, S, N, W, R, M, E, H, C, P, L$ and no additional host-level interval intersection, quotient representative, choice functional, trichotomy, or completeness coordinate.

The dyadic contraction obligation is read from D and S. The midpoint and radius-halving data are carried as dyadic rows, while the shrinking schedule tells which finite budget is attached to the retained child interval.

The nesting and readback obligations are then rowwise. The containment ledger N records inclusion between retained windows, W names the finite stream windows, and R repacks those windows as regular rational readbacks.

Finally, M and E are downstream rows on the same packet. The transport, replay, provenance, and local naming rows H, C, P, L preserve the route, so every completion-facing or real-seal read is forced through the displayed finite obligations and cannot export an excluded host object. □

Theorem 7224.3 (Bishop interval nesting metric-completion point seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{MetricCompletion}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}$ *carrier factors through*

$$I, D, S, N, W, R, M, E, H, C, P, L.$$

In particular, the point seal is the displayed route from located nested intervals through $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff before the terminal* $\mathsf{Real}^{\uparrow}$ *row is exposed; it is not an ambient intersection theorem or a selected completed point.*

Proof. Begin with the accepted carrier and read its located interval row I. The dyadic row D supplies midpoint and child-window data, while S supplies the shrinking diameter budget that controls the requested precision.

The retained windows cannot be consumed until the nested containment ledger N is displayed. Once N is present, the stream row W names the finite window sequence and the regular row R provides the $\mathsf{RegSeqRat}^{\uparrow}$ readback seen by downstream real and completion consumers.

The metric-completion handoff M is therefore not a selected quotient representative; it is the same finite regular route repackaged for $\mathsf{MetricCompletion}^{\uparrow}$. The real seal E is downstream of that handoff and remains transported by H, C, P, L.

Since the carrier has no field for an ambient completeness theorem, selected intersection point, quotient endpoint equality, countable choice, or detached real seal, every point-seal read is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}$ packet is generated from located interval rows, dyadic midpoint and bisection rows, shrinking diameter budgets, nested containment ledgers, StreamName windows, RegSeqRat readback, a MetricCompletion handoff, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7224.1, Definition 7224.2, and Definition 7224.3 seed the finite handoff from Bishop-style interval nesting to MetricCompletion and Real consumers through displayed dyadic, stream, and regular-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestingMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, selected interval intersections, quotient equality of endpoint names, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, dyadic contraction, nested containment, StreamName window exposure, RegSeqRat readback, MetricCompletion handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalNestingMetricCompletionUp.

Chapter 7225

A Concrete Naming Certificate for $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$

Following Bishop–Bridges on the uniform Cauchy criterion for functions, the $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$ packet records the finite function-window data by which a real-valued function family is read as uniformly Cauchy before any completed function object is available. It sits beside the already named $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ packet of chapter 1830, the continuous-map rows of chapter 130, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the criterion as BEDC rows for later uniform-convergence and continuous-map consumers; it is not host function equality, quotient equality of streams, countable choice, selected pointwise limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7225.1 (Cauchy criterion for functions carrier)**.** *A* $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, T, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the source function-window ledger. It lists the accepted finite argument windows, the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *rows available on those windows, and the paired value rows whose tails are compared.*

2. *T is the uniform tolerance request row. It records one displayed precision request and the single tail threshold demanded for all windows in W.*

3. *L is the pairwise tail-comparison ledger. For every pair of function indices beyond the threshold in T and every displayed source window in W, it stores the finite comparison row witnessing that the two value observations are within the requested dyadic tolerance.*

4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every rational observation exposed by the value comparisons in L.*

5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the function windows, tolerance request, comparison ledger, and regular rational readback have been displayed.*

6. *H records componentwise* hsame *transport for the function-window ledger, tolerance request, pairwise comparison ledger, readback row, real seal, replay, provenance, and local naming row.*

7. *C records displayed* Cont *replay from a uniform Cauchy request through W, T, L, R, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionForFunctions}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $W, T, L, R, E, H, C, P, N$. The carrier has no coordinate for host function equality, completed function-space equality, quotient streams, selected pointwise limits, countable-choice schedules, ambient real completeness, Lean verification, or bridge checking.

Theorem 7225.2 (Cauchy criterion for functions NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$ *carrier*

$$K = (W, T, L, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCriterionForFunctions}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed function-window ledger, uniform tolerance request, pairwise tail-comparison ledger, regular rational readback, terminal real seal, transport, replay, provenance, and local name;*
2. *source-window exposure, requiring every function value used by the criterion to be read from the finite windows listed in W;*
3. *uniform threshold accountability, requiring the threshold row in T to serve all accepted source windows and all compared function-index pairs beyond that threshold;*
4. *pairwise tail-comparison coverage, requiring every comparison consumed by the criterion to appear in L with its requested dyadic tolerance;*
5. *regular rational readback, requiring each numeric observation in L to pass through R before the terminal real seal is inspected;*
6. *Real-seal non-escape, requiring E to remain downstream of W, T, L, R and the structural rows;*
7. *componentwise stability, requiring replacement along* hsame *and displayed* Cont *replay to preserve the same finite criterion surface;*
8. *provenance exhaustion, requiring P to source only the listed rows and to export no host function equality, quotient stream, selected pointwise limit, countable-choice schedule, or ambient real-completeness principle;*
9. *local naming, requiring N to name exactly this finite criterion surface and no external function-space object.*

No local NameCert obligation is stored outside these rows.

Theorem 7225.3 (Cauchy criterion for functions uniform-window handoff)**.** *Let $K = (W, T, L, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$ *carrier. Every* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$*, uniform-convergence, or continuous-map consumer that reads the criterion must first expose the finite route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the source function windows W, the uniform tolerance request T, the pairwise tail-comparison ledger L, the readback row R, and the terminal seal E. Thus the consumer receives a uniform

Cauchy criterion only as displayed function-window data with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It cannot replace the hand-off by host function equality, quotient stream equality, countable choice, selected pointwise limits, completed function-space equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionForFunctions}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForFunctions}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7225.1, Definition 7225.2, and Definition 7225.3 seed the finite function-window route from the uniform Cauchy criterion to RealFunctionUniformCauchy, ContinuousMap, StreamName, RegSeqRat, DyadicRatCore, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForFunctions}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, completed function-space equality, quotient streams, countable choice, selected pointwise limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, uniform threshold accountability, pairwise tail-comparison coverage, regular rational readback, Real-seal non-escape, componentwise stability, provenance exhaustion, and local naming for BEDC.Derived.CauchyCriterionForFunctionsUp.

Constructive story: A $\mathsf{CauchyCriterionForFunctions}^{\uparrow}$ packet is generated from finite function-window rows, one uniform tolerance request, a pairwise tail-comparison ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7226

A Concrete Naming Certificate for $\mathsf{WhitneyApproximation}^{\uparrow}$

Following Whitney and standard smooth approximation theory, the $\mathsf{WhitneyApproximation}^{\uparrow}$ packet records a finite NameCert route for replacing a displayed continuous map on a finite manifold window by a smooth-row handoff at named tolerances. It sits over the finite-atlas source of chapter 96, the continuous-map source of chapter 130, the local smoothing cover of chapter 3369, smooth-row consumers such as chapter 4291 or chapter 136, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the finite approximation certificate; it does not assert a host smooth-function space, select a smoothing operator, identify quotient maps, or use ambient completeness.

Definition 7226.1 (Whitney approximation carrier)**.** *A* $\mathsf{WhitneyApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, F, U, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *finite-atlas source row, exposing only the displayed chart domains, overlap rows, and local coordinate names.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *source map row to be approximated on the displayed finite window.*
3. *U is the* $\mathsf{PartitionOfUnity}^{\uparrow}$ *local cover and weight row used to assemble chartwise smooth pieces.*
4. *L is the smooth-row ledger, naming the local* $\mathsf{SmoothManifold}^{\uparrow}$ *handoff or* $\mathsf{DiffForm}^{\uparrow}$ *consumer-facing row reached after the partitioned construction.*
5. *S is the finite approximation schedule, listing the chart windows, requested tolerances, and overlap checks inspected by the packet.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the displayed tolerances and chartwise comparison data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite error bounds visible to downstream consumers.*
8. *H records componentwise* hsame *transport for the manifold, source map, partition, smooth ledger, schedule, readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the continuous source through the finite cover, local smooth rows, rational readback, and terminal Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{WhitneyApproximation}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{SmoothManifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host smooth-function space, a choice-selected smoothing operator, quotient map equality, a global approximation theorem, or ambient completeness.

Theorem 7226.2 (Whitney approximation NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyApproximation}^{\uparrow}$ *carrier* $W = (M, F, U, L, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WhitneyApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed manifold, continuous-map, partition, smooth-ledger, schedule, regular-readback, Real-seal, transport, replay, provenance, and local naming rows; finite-atlas exposure through M; source-map exposure through F; local smoothing cover exposure through U and L; tolerance readback through S, R, E; componentwise stability through H; displayed replay through C; and ledger exhaustion through P, N.*

Proof. Project the carrier of Definition 7226.1. The only analytic rows are the finite-atlas source, continuous source map, partition-of-unity cover, smooth ledger, approximation schedule, regular rational readback, and Real seal.

The local smoothing cover is visible before any smooth handoff is public. The partition row U and smooth ledger L name the chartwise route, while S, R, E record the finite tolerance checks that a downstream smooth-row consumer may inspect.

The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. Since no host smooth-function space, selected smoothing operator, quotient map equality, global theorem, or ambient completeness row is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7226.3 (Whitney approximation smooth handoff)**.** *Every* $\mathsf{SmoothManifold}^{\uparrow}$ *or* $\mathsf{DiffForm}^{\uparrow}$ *consumer of an accepted* $\mathsf{WhitneyApproximation}^{\uparrow}$ *packet must first read*

$$M, F, U, L, S, R, E$$

before it receives a smooth-row handoff. The consumer receives a finite atlas, a displayed continuous source map, a partition-of-unity assembly row, local smooth rows, finite tolerance windows, regular rational readback, and a terminal Real seal; it receives no host smooth-function space, selected smoothing operator, quotient map equality, global Whitney theorem, or ambient completeness principle.

Proof. Start with Definition 7226.2. It requires carrier admission and finite-atlas exposure before any approximation-facing consumer can inspect the packet.

The smooth handoff is downstream of U and L. These rows display the partitioned local construction and the smooth-row ledger, so a $\mathsf{SmoothManifold}^{\uparrow}$ or $\mathsf{DiffForm}^{\uparrow}$ consumer cannot enter through an untracked host smoothing operator.

The tolerance read is S, R, E. The finite schedule lists the inspected windows, the regular rational row supplies the readback, and the Real seal records the displayed error bound. The remaining rows H, C, P, N only transport, replay, source, and name this route, leaving no coordinate for quotient equality, a global theorem, or ambient completeness. □

Theorem 7226.4 (Whitney approximation nonescape)**.** *Every accepted* $\mathsf{WhitneyApproximation}^{\uparrow}$ *public read factors through the displayed rows*

$$M, F, U, L, S, R, E, H, C, P, N.$$

Thus the packet can expose only a finite NameCert approximation route over $\mathsf{Manifold}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{SmoothManifold}^{\uparrow}$ *or* $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. It cannot export a host smooth-function space, a choice-selected smoothing operator, quotient map equality, ambient completeness, or a completed global approximation theorem.*

Proof. Use Definition 7226.1. The carrier displays exactly the eleven rows $M, F, U, L, S, R, E, H, C, P, N$.

Apply Definition 7226.2 and Definition 7226.3. They force every smooth-facing read to pass through the finite-atlas, continuous-map, partition, smooth-ledger, tolerance, readback, and Real-seal rows before structural transport and provenance are consumed.

The excluded host objects are not coordinates of the carrier and are not introduced by the structural rows. Therefore every public read remains inside the displayed finite NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{WhitneyApproximation}^{\uparrow}$ packet is generated from a finite ManifoldUp atlas row, a ContinuousMap source row, a PartitionOfUnity local cover, SmoothManifold or DiffForm handoff rows, a finite approximation schedule, RegSeqRat readback, a RealUp error seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7226.1, Definition 7226.2, Definition 7226.3, and Definition 7226.4 seed the finite smooth-approximation route over manifold, continuous-map, partition, smooth-row, regular-readback, and Real-seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhitneyApproximation}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a host smooth-function space, choice-selected smoothing operator, quotient map equality, ambient completeness, or a completed global Whitney approximation theorem.

Upgrade path: Obligation closure requires checked carrier admission, local cover exposure, smooth-row handoff, tolerance readback, transport, replay, provenance, and local naming for BEDC.Derived.WhitneyApproximationUp.

Chapter 7227

A Concrete Naming Certificate for $\mathsf{CompactCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges on compact metric sequence arguments, the $\mathsf{CompactCauchySubsequence}^{\uparrow}$ packet records the finite subsequence-window route by which compact-metric data can be consumed as a Cauchy subsequence surface. It sits beside the sequential compactness window packet of chapter 2372, the compact metric source of chapter 129, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies RegSeqRat and Real seal arguments with a finite subsequence-window carrier, not host subsequence choice, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7227.1 (Compact Cauchy subsequence carrier)**.** *A* $\mathsf{CompactCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, J, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the compact metric carrier and the finite cover or total-bounded window that bounds the inspected sequence.*
2. *J is the* $\mathsf{SequentialCompact}^{\uparrow}$ *subsequence-window row, recording the finite increasing index windows supplied to the compact source.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row for the original sequence observations.*
4. *W is the selected finite subsequence-window ledger, displaying the inspected indices and their monotonicity evidence.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for pairwise tail comparisons inside W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the subsequence observations after the stream and dyadic rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, J, S, W, D, R have been exposed.*
8. *H records componentwise* hsame *transport for the compact source, sequential window, stream window, subsequence window, dyadic tolerance, regular readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact Cauchy-subsequence request through* $K, J, S, W, D, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactCauchySubsequence}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $K, J, S, W, D, R, E, H, C, P, N$. It has no coordinate for host subsequence choice, quotient stream equality, selected limits, countable-choice schedules, open-cover compactness as a hidden theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7227.2 (Compact Cauchy subsequence obligation surface)**.** *For every accepted* CompactCauchySubsequen *carrier $Q = (K, J, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, sequential compactness window, stream observation window, selected finite subsequence window, dyadic tolerance ledger, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows.*

Consequently every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $K, J, S, W, D, R, E, H, C, P, N$ and cannot replace it by host subsequence choice, quotient stream equality, countable choice, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7227.1. The displayed rows expose the compact metric source, the sequential compactness window, the original stream observations, the selected finite subsequence window, the dyadic tolerance ledger, the regular readback, and the terminal real seal.

The compact and sequential rows are finite. The $\mathsf{CompactMetric}^{\uparrow}$ source supplies only compact-metric carrier and finite cover data, while the $\mathsf{SequentialCompact}^{\uparrow}$ row supplies only the subsequence-window surface already named by the sibling chapter.

The Cauchy readback rows are downstream. The stream row S opens the inspected observations, the window row W records the selected finite indices, D records the dyadic comparison budget, and R is the regular rational readback available to later consumers.

Finally the Real seal E and the structural rows H, C, P, N preserve the same route. Since the carrier contains no field for host subsequence selection, quotient stream equality, selected limits, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCauchySubsequence}^{\uparrow}$ packet is generated from compact metric source rows, SequentialCompact finite subsequence-window rows, StreamName observations, DyadicRatCore tolerance comparisons, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCauchySubsequence}^{\uparrow}}$, seed)

Scope closed: Definition 7227.1 and Definition 7227.2 seed the finite compact Cauchy subsequence route through compact metric, sequential-window, StreamName, dyadic tolerance, RegSeqRat, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCauchySubsequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subsequence choice, quotient stream equality, selected real limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, subsequence-window monotonicity, stream-window exactness, dyadic Cauchy comparison, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CompactCauchySubsequenceUp.

Chapter 7228

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}$

Following Bishop and Bridges on compact uniform continuity and Cauchy convergence, the CompactUniformCauchyT packet records the finite route that turns compact finite-net modulus data into a Cauchy-tail handoff for regular real-name consumers. It realizes the compact-uniform-continuity route of chapter 9237 through the compact metric source of chapter 129, the compact uniform modulus packet of chapter 2299, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The router is a displayed finite tail packet; it is not a host uniform-continuity theorem, selected global modulus, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7228.1 (Compact uniform Cauchy-tail router carrier)**.** *A* $\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, M, S, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row.*
2. *M is the* $\mathsf{CompactUniformModulus}^{\uparrow}$ *row that converts compact finite-net and pointwise-modulus data into a tolerance-only modulus ledger.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row for the source and target Cauchy windows opened by the router.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested tail precision and compact-net radius budget.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream and dyadic rows.*
6. *L is the Cauchy-tail ledger, recording the finite threshold and the pairwise tail comparisons consumed by downstream regular real-name readers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, M, S, D, R, L have been exposed.*
8. *H records componentwise* hsame *transport for the compact source, modulus, stream windows, dyadic ledger, regular readback, tail ledger, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a compact-uniform Cauchy-tail request through $K, M, S, D, R, L, E, H, P, N$*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, S, D, R, L, E, H, C, P, N$. It has no coordinate for a host uniform-continuity theorem, selected global modulus, quotient stream equality, selected limits, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7228.2 (Compact uniform Cauchy-tail router obligation surface)**.** *For every accepted* $\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}$ *carrier $T = (K, M, S, D, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCa}}$ *surface consists exactly of carrier admission for the compact metric source, compact uniform modulus row, stream windows, dyadic tolerance ledger, regular rational readback, Cauchy-tail ledger, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Consequently every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the router must read*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactUniformModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $K, M, S, D, R, L, E, H, C, P, N$ and cannot replace the route by a selected global modulus, quotient stream equality, selected real limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7228.1. The compact source and uniform-modulus rows are exactly K and M, so no host compactness theorem or external modulus is available to the consumer.

The tail route is then finite. The row S names the inspected stream windows, D records the dyadic tolerance budget for the same request, R provides the regular rational readback, and L records the threshold and pairwise tail comparisons read by downstream consumers.

The terminal seal E is downstream of those rows. Componentwise transport, displayed replay, provenance, and local naming preserve the same finite packet, and the carrier has no additional coordinate for quotient stream equality, selected limits, countable choice, or ambient Real completeness. Thus the listed rows exhaust the obligation surface. □

Theorem 7228.3 (Compact uniform Cauchy-tail router obligation package)**.** *For the carrier of Definition 7228.1, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}$ *route assembles one obligation package consisting of carrier admission, compact finite-net exposure, uniform-modulus handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Cauchy-tail ledger coverage,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Every consumer of the package must therefore read the finite route

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactUniformModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $K, M, S, D, R, L, E, H, C, P, N$ and cannot replace it by a host compactness theorem, selected global modulus, quotient stream equality, selected real limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7228.1. The rows $K, M, S, D, R, L, E, H, C, P, N$ are exactly the compact source, uniform-modulus handoff, stream-window, dyadic-tolerance, regular-readback, tail-ledger, Real-seal, transport, replay, provenance, and local naming fields available to the certificate.

Apply Definition 7228.2. It identifies those same rows as the whole local obligation surface and orders every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer through the displayed finite route.

Since no carrier coordinate stores a host compactness theorem, selected global modulus, quotient stream equality, selected real limit, choice schedule, or ambient completeness oracle, the projected rows and the obligation-surface theorem compose as one obligation package for the local $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}$ route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}$ packet is generated from compact metric finite-net rows, CompactUniformModulus rows, StreamName windows, DyadicRatCore tolerance budgets, RegSeqRat readback, a Cauchy-tail ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7228.1 and Definition 7228.2 seed the compact-uniform Cauchy-tail route through compact metric, uniform-modulus, StreamName, dyadic tolerance, RegSeqRat, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyTailRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host uniform-continuity semantics, selected global moduli, quotient stream equality, selected real limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, uniform-modulus handoff, stream-window exactness, dyadic tail-budget exposure, RegSeqRat readback, Cauchy-tail threshold coverage, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformCauchyTailRouterUp.

Chapter 7229

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$ packet records the finite route by which regular Cauchy evidence is admitted as a Real-facing limit theorem without importing a global completeness principle. It sits after the finite observation discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a theorem-facing carrier for the displayed tail-window route; it is not a selected limit, quotient equality of regular names, arbitrary trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7229.1 (Regular Cauchy limit theorem carrier)**.** *A* $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the tail-modulus row, displaying the finite threshold schedule used by the accepted regular Cauchy evidence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose observations request the tail after the displayed threshold.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic scale at which the tail comparison is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the tail modulus, stream window, and dyadic tolerance rows are visible.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular rational readback row R.*
6. *H records componentwise* hsame *transport for modulus, window, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTheorem}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $M, S, D, R, E, H, C, P, N$. It has no coordinate for selecting a completed limit, quotienting regular Cauchy names, arbitrary real trichotomy, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.RegularCauchyLimitTheoremUp`

Theorem 7229.2 (Regular Cauchy limit theorem obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$ *carrier $L = (M, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission; tail-modulus visibility through M; finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window visibility through S;* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through D;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *seal handoff through E; componentwise stability under* hsame *through H; displayed* Cont *replay through C;* Pkg *provenance through P; local naming through N; and ledger non-escape for all Real-facing reads.*

Proof. Project the carrier. The rows M, S, D are the only source rows visible before the regular rational readback, so tail-modulus visibility, window visibility, and dyadic tolerance exposure are all finite carrier clauses.

Read the handoff rows next. The row R is downstream of M, S, D, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream of R.

The structural rows H, C, P, N transport, replay, source, and name the same displayed route. Since no excluded selected-limit, quotient-equality, trichotomy, choice, or completeness coordinate occurs in the carrier, ledger non-escape follows for every Real-facing read. □

Theorem 7229.3 (Regular Cauchy limit theorem tail-seal handoff)**.** *Every Real-facing read of an accepted* $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$M, S, D \longmapsto R \longmapsto E$$

before the terminal seal is consumed. Transport, replay, provenance, and local naming preserve this route and do not add an alternate limit selector or completeness-facing coordinate.

Proof. Start with the obligation theorem Definition 7229.2. It identifies M, S, D as the complete finite source surface for the theorem-facing tail read.

The regular readback row is next in the carrier order. A Real-facing consumer therefore reaches R only after the tail modulus, stream window, and dyadic tolerance rows have been displayed.

Finally consume the seal row E. The rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and naming, so no consumer can bypass the tail-window, readback, and tolerance route before reading the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitTheorem}^{\uparrow}$ packet is generated from a tail-modulus row, a StreamName tail-window row, a DyadicRatCore tolerance row, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 7229.1, Definition 7229.2, and Definition 7229.3 seed the finite theorem-facing tail-window route from regular Cauchy evidence to the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, selected limits, quotient regular Cauchy equality, arbitrary trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-modulus visibility, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLimitTheoremUp.

Chapter 7230

A Concrete Naming Certificate for $\mathsf{RemovableDiscontinuity}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RemovableDiscontinuity}^{\uparrow}$ packet records the finite pointwise repair route by which a function value at one displayed puncture is replaced by a displayed limiting value before a continuity-facing consumer reads the repaired function. It sits beside the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the finite limiting surface of $\mathsf{Limit}^{\uparrow}$, the continuous-function surfaces $\mathsf{Continuous}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$, and the Riemann-specific repair packet of chapter 7217. The packet repairs only the named local defect; it does not select global functions, identify quotient real functions, invoke countable choice, assume ambient $\mathsf{Real}^{\uparrow}$ completeness, or prove a host extensionality principle.

Definition 7230.1 (Removable-discontinuity carrier)**.** *A* $\mathsf{RemovableDiscontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, W, L, V, E, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the punctured point row, naming the source point, the displayed domain side, and the fact that the ordinary value at the point is not the row read by the repair certificate.*
2. *W is the punctured-window row, listing the finite neighbourhood windows and exclusions used to inspect the source function away from P.*
3. *L is the limiting-value row, read through* $\mathsf{Limit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data attached to the punctured windows.*
4. *V is the repaired-value row, inserting the displayed limiting value as the value assigned at the puncture.*
5. *E is the continuous-extension row, recording the repaired local function surface through* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *after P, W, L, V have been displayed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for all numeric observations used by W, L, V, E.*
7. *H records componentwise* hsame *transport for the puncture, punctured windows, limiting value, repaired value, extension row, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a repair request through $P, W, L, V, E, R, H, Q, N$.*
9. *Q is the* Pkg *provenance over* $\mathsf{RemovableDiscontinuity}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RemovableDiscontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, W, L, V, E, R, H, C, Q, N$. It has no coordinate for host function extensionality, selected global functions, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7230.2 (Removable-discontinuity NameCert obligations)**.** *For every accepted* $\mathsf{RemovableDiscontinuity}^{\uparrow}$ *carrier*

$$M = (P, W, L, V, E, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{RemovableDiscontinuity}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier admission for the displayed packet M;*
2. *punctured-window exposure, requiring the source read to pass through P and W before any repair value is consumed;*
3. *limit-value readback, requiring L and R to expose the* $\mathsf{Limit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route used for the replacement value;*
4. *repaired-value insertion and continuous-extension handoff, requiring V to be inserted before E is exported to* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumers;*
5. *ledger nonescape, meaning every consumer-visible repair claim is projected from $P, W, L, V, E, R, H, C, Q, N$ and from no host extensionality, selected global function, quotient real equality, choice principle, or ambient completeness datum.*

Proof. Project the carrier of Definition 7230.1. The pointwise repair rows are P, W, L, V, E, and the structural rows are R, H, C, Q, N.

The puncture row P fixes the only point at which the ordinary source value is not read directly. The punctured-window row W lists the finite neighbourhood observations away from that point, so a consumer cannot infer a global function extension before the local windows are displayed.

The limiting value is read through L together with the numeric route R. These rows bind the $\mathsf{Limit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data used for the replacement value, while V records that this displayed value is inserted at the puncture.

The handoff row E is therefore available only after P, W, L, V, R have been displayed. Transport and replay use H and C, and provenance and local naming use Q and N.

Since no excluded principle is a coordinate of the carrier, none can be exported by the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{RemovableDiscontinuity}^{\uparrow}$ read must expose a punctured point row, a punctured-window row, a limiting-value row, a repaired-value row, a continuous-extension row, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, limit, continuous, continuousmap, riemann-removable-discontinuity: $\mathsf{RemovableDiscontinuity}^{\uparrow}$ is not bijective to any listed sibling because it binds the punctured point, punctured-window route, limiting-value readback, repaired-value insertion, and continuous-extension handoff in one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RemovableDiscontinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RemovableDiscontinuity}^{\uparrow}$ packet is generated from a punctured point row, a punctured-window row, a limiting-value row, a repaired-value row, a continuous-extension row, regular rational and Real readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RemovableDiscontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7230.1 and Definition 7230.2 seed the finite pointwise repair surface over $\mathsf{Limit}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RemovableDiscontinuity}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, selected global functions, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, punctured-window exposure, limit-value readback, repaired-value insertion, continuous-extension handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.RemovableDiscontinuityUp.

Chapter 7231

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}$ packet realizes the boundary-distortion budget route of chapter 9463 as finite NameCert data. It sits after the finite disk-map rows of chapter 6776, the boundary-shadow rows of chapter 6678, the visual-metric rows of chapter 3866, and the cross-ratio rows of chapter 3948. The budget row consumes those finite disk-map, boundary-shadow, visual-metric, and cross-ratio rows before the Real-facing distortion seal is exposed. The packet is not a completed hyperbolic plane, a selected ideal endpoint, quotient boundary equality, an external quasi-isometry theorem, or an ambient completeness principle.

Definition 7231.1 (Hyperbolic boundary-distortion budget carrier)**.** *A* $\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, V, X, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-map row, displaying the Poincare disk map whose boundary distortion is being read.*
2. *S is the boundary-shadow row, recording the finite boundary arc or endpoint shadow consumed by the observer-expansion route.*
3. *V is the visual-metric row attached to the same boundary shadow.*
4. *X is the cross-ratio or metric-comparison row that compares the displayed before-and-after boundary shadows.*
5. *M is the metric-comparison budget row, recording the finite distortion allowance for the displayed disk map.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the distortion threshold.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after disk-map, boundary-shadow, visual-metric, cross-ratio or metric-comparison, dyadic tolerance, and budget rows are visible.*
8. *H records componentwise* hsame *transport for disk-map, boundary-shadow, visual-metric, comparison, tolerance, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$D, S, V, X, M, Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* HyperbolicBoundaryDistortionBudget$^{\uparrow}$, PoincareDiskAutomorphism$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{HyperbolicBoundaryDistortionBudget}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, V, X, M, Q, E, H, C, P, N$. It has no coordinate for a completed hyperbolic surface, selected boundary endpoint, quotient boundary equality, hidden global distortion theorem, or ambient completeness principle.

Lean4 (statement) `BEDC.Derived.HyperbolicBoundaryDistortionBudgetUp`

Theorem 7231.2 (Hyperbolic boundary-distortion budget obligations)**.** *For every accepted* HyperbolicBoundaryDis *carrier $B = (D, S, V, X, M, Q, E, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicBoundaryDistortionBudget}^{\uparrow}}$ *surface consists exactly of disk-map admission through D; boundary-shadow exposure through S; visual-metric exposure through V; cross-ratio or metric-comparison exposure through X; finite distortion-budget exposure through M; dyadic tolerance exposure through Q;* Real$^{\uparrow}$ *seal non-escape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier. The rows D, S, V, X, M, Q are the complete boundary-distortion source surface: a disk map, its boundary shadow, the visual metric, the comparison read, the finite budget, and the dyadic tolerance.

The Real-facing row is downstream. The seal E is exposed only after the displayed source rows have been read, so a boundary-distortion consumer cannot reach a Real value from an unregistered disk map or hidden comparison theorem.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no completed surface, selected endpoint, quotient boundary equality, hidden global distortion theorem, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7231.3 (Hyperbolic boundary-distortion budget nonescape)**.** *Every accepted* HyperbolicBoundaryDistorti *Real-facing read factors through the displayed disk-map, boundary-shadow, visual-metric, cross-ratio or metric-comparison, dyadic tolerance, transport, replay, provenance, and local NameCert rows before it consumes the seal. No consumer may export boundary distortion from a hidden hyperbolic metric, an unstated cross-ratio, a selected ideal endpoint, quotient boundary equality, or ambient completeness.*

Proof. Start from Definition 7231.2. It places the disk map D, boundary shadow S, visual metric V, comparison row X, distortion budget M, and dyadic tolerance Q before the Real seal.

Transport and replay preserve the displayed order. The rows H and C may replace the packet by componentwise same rows and replay the route, but neither adds a new source for distortion.

Provenance and local naming close the finite route. Since P and N name only the displayed rows, the refused hidden metric, unstated comparison, selected endpoint, quotient equality, and completeness principles have no export coordinate. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}$ boundary-distortion read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying disk-map, boundary-shadow, visual-metric, cross-ratio or metric-comparison, dyadic tolerance, transport, replay, provenance, and local NameCert rows.

Independence witness: hyperbolic-metric, hyperbolic-visual-metric, hyperbolic-cross-ratio, hyperbolic-boundary-shadow, poincare-disk-automorphism: no listed sibling is bijective to HyperbolicBoundaryDistortionBudget because the budget packet binds disk-map, boundary-shadow, visual-metric, comparison, dyadic tolerance, Real seal, transport, replay, provenance, and local naming as one finite distortion certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}$ packet is generated from disk-map rows, boundary-shadow rows, visual-metric rows, cross-ratio or metric-comparison rows, dyadic tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7231.1, Definition 7231.2, and Definition 7231.3 seed the finite boundary-distortion budget route for observer-expansion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryDistortionBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic surfaces, selected ideal endpoints, quotient boundary equality, hidden global distortion theorems, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, boundary-shadow exposure, visual-metric exposure, cross-ratio or metric-comparison exposure, dyadic tolerance, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBoundaryDistortionBudgetUp.

Chapter 7232

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$

The $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ packet realizes the observer-horizon branch of chapter 9463 as finite NameCert data. It sits over the visual boundary surface of chapter 4181, the Gromov-product comparison route of chapter 6073, the Busemann cocycle route of chapter 4134, the horoball-shadow row of chapter 6569, observer-window sources such as chapter 4154, and the terminal real-name seal of chapter 38. The packet names the finite horizon-facing route; it does not assert a global boundary compactification, a completed spacetime horizon, a selected geodesic ray, quotient boundary equality, or ambient hyperbolic completeness.

Definition 7232.1 (Hyperbolic observer-horizon carrier)**.** *A* $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, G, U, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row naming the visible boundary window under inspection.*
2. *G is the* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *comparison row for the same observer-side basepoint and boundary window.*
3. *U is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *cocycle row used to orient the boundary-facing displacement read.*
4. *S is the* $\mathsf{HyperbolicHoroballShadow}^{\uparrow}$ *row naming the horoball-shadow window that a horizon consumer may inspect.*
5. *W is the observer-window row, read through the observer-shadow or observer-expansion surface, listing the finite records visible to the focused observer.*
6. *R is the regular readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for horizon depth, shadow threshold, or visual-radius comparisons.*
7. *E is the refusal row excluding interior far-end selection, quotient boundary equality, global compactification, and completed spacetime-horizon semantics.*
8. *H records componentwise* hsame *transport for the boundary, product, cocycle, horoball-shadow, observer-window, Real-seal, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from observer-window inspection through boundary, product, cocycle, horoball-shadow, Real-seal, and refusal rows.*

10. *P is the* Pkg *provenance over* HyperbolicObserverHorizon$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicGromovProduct HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicHoroballShadow$^{\uparrow}$, *observer-window rows,* Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicObserverHorizon}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global compactification object, selected geodesic ray, quotient boundary equality, completed spacetime horizon, hidden observer synchronization, or ambient hyperbolic completeness.

Theorem 7232.2 (Hyperbolic observer-horizon NameCert obligations)**.** *For every accepted* HyperbolicObserverHori *carrier $O = (B, G, U, S, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicObserverHorizon}^{\uparrow}}$ *surface consists exactly of carrier admission for the visual boundary, Gromov-product, Busemann-cocycle, horoball-shadow, observer-window, Real-seal, refusal, transport, replay, provenance, and local naming rows; boundary exposure through B; product comparison through G; cocycle orientation through U; shadow exposure through S; observer-window accountability through W; Real-seal readback through R; refusal exactness through E; and componentwise stability through H, C, P, N.*

Proof. Project the carrier of Definition 7232.1. The horizon-facing rows are exactly B, G, U, S, W, R, E, and the structural rows are H, C, P, N.

The boundary route is visible before any observer-horizon read is public. The visual boundary row B, Gromov product row G, Busemann cocycle row U, and horoball-shadow row S fix the boundary, comparison, orientation, and shadow data.

The observer and Real-facing rows are then W, R, E. They expose the finite observer window, seal the requested real comparison, and refuse the excluded global or quotient objects. The structural rows transport and replay only those displayed rows, so the listed obligations exhaust the local NameCert surface. □

Theorem 7232.3 (Hyperbolic observer-horizon boundary handoff)**.** *Every horizon-facing consumer of an accepted* HyperbolicObserverHorizon$^{\uparrow}$ *packet must first read*

$$B, G, U, S \longmapsto W, R, E \longmapsto H, C, P, N.$$

The consumer receives visual boundary, Gromov-product, Busemann-cocycle, horoball-shadow, observer-window, Real-seal, refusal, transport, replay, provenance, and local naming rows. It receives no selected geodesic ray, quotient boundary equality, global boundary compactification, completed spacetime-horizon object, hidden synchronized observer coordinate, or ambient hyperbolic completeness principle.

Proof. Use Definition 7232.2. It places the visual boundary, product comparison, Busemann cocycle, and horoball-shadow rows before any observer-window or Real-seal consumer can read the horizon packet.

The observer read is finite. The row W lists the inspected observer records, R seals the real comparison requested by that read, and E records the refusal of interior far-end selection and quotient boundary equality.

Finally H, C, P, N preserve the same finite route. Since the carrier contains no global compactification, selected ray, quotient equality, synchronized observer coordinate, completed horizon, or ambient completeness row, no consumer can obtain one from the packet. □

Falsifiable prediction: Every accepted HyperbolicObserverHorizon$^{\uparrow}$ read must expose HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicGromovProduct$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicHoroballShadow$^{\uparrow}$, observer-window,

terminal $\mathsf{Real}^{\uparrow}$ seal, refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any horizon-facing conclusion is visible.

Independence witness: hyperbolic-visual-boundary, hyperbolic-gromov-product, hyperbolic-busemann-cocycle, hyperbolic-horoball-shadow, hyperbolic-observer-shadow-metric: none is bijective to $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ because the horizon packet jointly binds boundary readback, product comparison, Busemann orientation, horoball-shadow exposure, observer-window accountability, Real sealing, refusal, transport, replay, provenance, and local naming.

The Lean follow-up for this packet must provide explicit Nontrivial $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ TasteGate instances. Those instances must keep $B, G, U, S, W, R, E, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizon}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverHorizon}^{\uparrow}$ packet is generated from a HyperbolicVisualBoundary row, a HyperbolicGromovProduct comparison row, a HyperbolicBusemannCocycle orientation row, a HyperbolicHoroballShadow window, an observer-window row, a RealUp seal, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizon}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7232.1, Definition 7232.2, and Definition 7232.3 seed the finite observer-horizon route over visual boundary, Gromov product, Busemann cocycle, horoball shadow, observer-window, Real-seal, and refusal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizon}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global boundary compactification, selected geodesic rays, quotient boundary equality, completed spacetime horizons, synchronized observer coordinates, or ambient hyperbolic completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary exposure, product comparison, Busemann orientation, horoball-shadow exposure, observer-window accountability, Real-seal readback, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicObserverHorizonUp.

Chapter 7233

A Concrete Naming Certificate for $\mathsf{CompactNetSelection}^{\uparrow}$

Following Bishop and Bridges on compact metric finite nets, the $\mathsf{CompactNetSelection}^{\uparrow}$ packet records the displayed finite center list and hit ledger that Real-and-Completion consumers may read from a compact metric source. It sits after the compact finite-net surface of chapter 129, the metric-distance classifier of chapter 57, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet selects no host point and invokes no open-cover compactness principle; it only names the finite rows already displayed by the compact metric source.

Definition 7233.1 (CompactNetSelection carrier)**.** *A* $\mathsf{CompactNetSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (K_0, M, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. K_0 *is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, carrying the compact metric source, the displayed center list, the net radius, and the finite coverage witness table.*
2. M *is the* $\mathsf{Met}^{\uparrow}$ *distance and hit row, recording the inspected point names, center names, distance comparisons, and hit witnesses used by the selected net.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing exactly the finite observations of source points and centers read by the compact-net query.*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distance and tolerance data opened by* M *and* S*.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for distance or tolerance comparisons, visible only after* K_0, M, S, Q *have been displayed.*
6. H *records componentwise* hsame *transport for the compact source, metric hit ledger, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a requested finite-net read through the center list, hit witnesses, StreamName windows, RegSeqRat readback, and Real seal.*
8. P *is the* Pkg *provenance over* $\mathsf{CompactNetSelection}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactNetSelection}^{\uparrow}}$ *row.*
The classifier accepts a packet only when the center list is displayed in K_0, every hit row is read through M, every observation window is exposed by S, every rational tolerance readback factors through Q, and every Real-facing comparison reaches E after the finite rows are visible. It has no coordinate for a host selector, an open-cover compactness theorem, a choice-selected center, quotient metric equality, selected Cauchy limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7233.2 (CompactNetSelection NameCert obligations)**.** *For every accepted* $\mathsf{CompactNetSelection}^{\uparrow}$ *carrier $K = (K_0, M, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactNetSelection}^{\uparrow}}$ *obligations consist exactly of carrier admission for the compact finite-net source, metric distance and hit ledger, StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows; center-list visibility through K_0; hit-row exactness through M; finite-window exactness through S; rational readback through Q; Real-seal nonescape through E; and preservation of the same route through H, C, P, N.*

Proof. Project the carrier of Definition 7233.1. Its visible rows are precisely $K_0, M, S, Q, E, H, C, P, N$, so every local obligation must be read from one of those rows.

The compact finite-net source row K_0 displays the center list and coverage table before the metric hit row is inspected. The metric row M then records the distance comparisons and hit witnesses attached to those displayed centers.

The Real-facing side is downstream. StreamName windows S expose the finite observations, RegSeqRat readback Q records the rational tolerance data, and the Real seal E is terminal.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no further carrier coordinate exists, host selectors, open-cover compactness, choice-selected centers, quotient equality, selected limits, and ambient Real completeness cannot be local NameCert obligations. □

Theorem 7233.3 (CompactNetSelection Real-and-Completion handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CompactNetSelection}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $K_0, M, S, Q, E, H, C, P, N$ before it consumes a compact-net center, hit witness, rational tolerance, or Real seal. The handoff exports a finite selected-net packet only; it does not export a host selector, an open-cover compactness theorem, quotient metric equality, a selected Cauchy limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with K_0. The compact metric source supplies the finite center list and coverage witnesses as displayed BEDC data, not as a host compactness principle.

Read M next. It ties every selected center use to a metric distance comparison and hit witness, so a downstream consumer cannot detach the selected-net row from the source metric classifier.

The readback rows S, Q, E then expose the finite observation windows, regular rational tolerance data, and terminal Real seal. These rows are the only route by which the selected net becomes visible to Real or Completion consumers.

The structural rows H, C, P, N preserve that route under transport and replay. Therefore every accepted handoff is the finite packet displayed in the carrier, with no coordinate for the excluded host selector, compactness, quotient, selected-limit, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{CompactNetSelection}^{\uparrow}$ Real-and-Completion read factors through CompactMetric finite-net source data, Metric distance and hit rows, StreamName windows, RegSeqRat readback, RealUp seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: compactmetric, metric, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{CompactNetSelection}^{\uparrow}$ because this packet alone binds the displayed finite center list and hit ledger to downstream Real-and-Completion readback through transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactNetSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactNetSelection}^{\uparrow}$ packet is generated from a CompactMetric finite-net source row, Metric distance and hit rows, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactNetSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7233.1, Definition 7233.2, and Definition 7233.3 seed the finite selected-net route for compact metric Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactNetSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host selector extraction, open-cover compactness, choice-selected centers, quotient metric equality, selected Cauchy limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactNetSelection}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, center-list visibility, hit-row exactness, finite-window exactness, rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactNetSelec

Chapter 7234

A Concrete Naming Certificate for $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$ packet records the finite subcover extraction step only after the cover has already been presented as displayed compact-metric, interval, dyadic, stream, regular-rational, and real-seal data. It sits beside the finite Lebesgue-number route of chapter 452, the compact source certificate of chapter 129, the closed interval carrier of chapter 1591, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names an extraction certificate for a displayed finite cover candidate; it does not assert general open-cover compactness, choose from an arbitrary cover family, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7234.1 (Finite subcover extraction carrier)**.** *A* $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, read only through finite-net and coverage data already admitted by the compact-metric packet.*
2. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *interval row when the cover is interval-facing; otherwise it records the absence of an interval specialization.*
3. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger tying the displayed cover candidates to a finite lower-bound row.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and radius-comparison ledger used to compare cover radii with requested windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly which cover candidates are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every rational endpoint, center, or radius consumed by the extraction.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, downstream of W, R, D and never a source of cover selection.*
8. *H records componentwise* hsame *transport for the compact, interval, radius, dyadic, stream, readback, and seal rows.*

9. *C records displayed* Cont *replay from a finite cover request through the extracted subcover rows.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple S. The carrier has no coordinate for an arbitrary open cover, selected infinite family, countable choice, host compactness theorem, quotient real equality, ambient completeness, Lean verification, or bridge checking.

Theorem 7234.2 (Finite subcover extraction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$ *carrier $S = (K, I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-metric source exposure through K, interval specialization through I when present, finite Lebesgue-number ledger exposure through L, dyadic tolerance comparison through D, finite-window inspection through W, regular rational readback through R, terminal Real-seal discipline through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7234.1. The accepted packet exposes exactly the rows $K, I, L, D, W, R, E, H, C, P, N$.

The extraction rows are finite before any consumer reads the result. The compact-metric row supplies the source coverage data, the interval row supplies the optional closed-bounded specialization, the finite Lebesgue-number row supplies a positive-radius ledger, and the dyadic row compares those radii with the requested windows.

The stream and readback rows then name the inspected cover candidates and rational observations. The terminal Real seal and structural rows preserve the same finite route, so no arbitrary open cover, countable-choice selector, host compactness theorem, quotient real equality, or ambient completeness coordinate is introduced. □

Theorem 7234.3 (Finite subcover extraction finite-cover route)**.** *Every finite-cover consumer of an accepted* $\mathsf{FiniteSubcoverExtraction}^{\uparrow}$ *packet must first read*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, RegSe

through the displayed rows $K, I, L, D, W, R, E, H, C, P, N$. The resulting subcover is therefore an extraction from already displayed finite rows, not a general open-cover compactness principle or a selected family outside the packet.

Proof. Start from the NameCert obligations of Definition 7234.2. They force every accepted read through the compact source, optional interval, finite-radius, dyadic comparison, finite-window, readback, and real-seal rows.

The finite-cover consumer can use only the rows named by the carrier. Thus the extracted list of cover candidates is read from W and checked against K, L, D, R before the seal row E is available.

Transport, replay, provenance, and local naming preserve that route for downstream compact-metric and interval consumers. Since no excluded compactness or choice object is a carrier coordinate, the route cannot be read as arbitrary open-cover compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverExtraction}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSubcoverExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7234.1, Definition 7234.2, and Definition 7234.3 seed the finite subcover extraction certificate over compact-metric, closed-interval, finite-radius, dyadic, stream, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteSubcoverExtraction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, arbitrary cover selection, countable choice, quotient Real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric source exposure, interval specialization, finite-radius ledger exposure, dyadic tolerance comparison, finite-window inspection, regular rational readback, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubcoverExtractionUp.

Constructive story: A FiniteSubcoverExtraction$^{\uparrow}$ packet is generated from compact-metric coverage data, an optional closed-bounded interval row, a finite Lebesgue-number radius ledger, dyadic tolerance comparisons, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7235

A Concrete Naming Certificate for $\mathsf{MazurLemma}^{\uparrow}$

Following Mazur's lemma in the standard Banach-space literature, the $\mathsf{MazurLemma}^{\uparrow}$ packet records the finite convex-combination ledger by which a weakly convergent sequence yields strong norm approximants in the convex hull of a tail. It sits over the vector-space source of chapter 47, the norm carrier of chapter 89, the convex-set rows of chapter 208, the bounded-linear-observer surface of chapter 1644, and the finite separation witness of chapter 3905. The certificate names only finite convex weights, tail windows, observer comparisons, separation-refusal rows, transport, replay, provenance, and local naming; it does not import a host weak topology, a selected subsequence, quotient equality of vectors, or a choice principle over all tails.

Definition 7235.1 (Mazur lemma finite convex-combination carrier)**.** *A* $\mathsf{MazurLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, N, C, T, W, O, S, A, R, H, Q, P, L)$$

with the following displayed rows:

1. *X is the normed vector-space source row, read through* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Norm}^{\uparrow}$*.*
2. *N is the weakly convergent sequence row, exposed only through finite bounded-linear-observer tests.*
3. *C is the convex target row, read through* $\mathsf{ConvexSet}^{\uparrow}$ *as the finite convex hull generated by a displayed tail window.*
4. *T is the tail-window row specifying the finite source terms available for one convex approximant.*
5. *W is the convex-weight ledger, listing rational nonnegative weights and the displayed sum-one certificate for the window T.*
6. *O is the bounded-linear-observer ledger, read through* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *for the finite family of observer tests used by the weak comparison.*
7. *S is the finite-dimensional separation row, routed through* $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ *to refute a displayed norm ball separation from the convex hull window.*
8. *A is the strong norm-approximation row, recording the norm bound obtained after the separation refusal.*
9. *R is the readback row that exposes the convex combination as a normed vector-space element.*

10. *H stores componentwise* hsame *transport for the source, sequence, convex, tail, weight, observer, separation, approximation, readback, replay, provenance, and local naming rows.*
11. *Q stores displayed* Cont *replay from a weak-to-strong request through $X, N, C, T, W, O, S, A, R, H, P, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{MazurLemma}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *L is the local* $\mathsf{NameCert}_{\mathsf{MazurLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host weak topology, an infinite convex hull, a selected subsequence, quotient vector equality, countable choice, or ambient Banach-space completeness.

Theorem 7235.2 (Mazur lemma NameCert obligations)**.** *For every accepted* $\mathsf{MazurLemma}^{\uparrow}$ *carrier*

$$M = (X, N, C, T, W, O, S, A, R, H, Q, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MazurLemma}^{\uparrow}}$ *surface consists exactly of normed vector-space admission through X, weak sequence exposure through the bounded-linear-observer ledger N, O, convex-hull admission through C, T, W, finite separation refusal through S, strong norm-approximation readback through A, R, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 7235.1. The normed source X, sequence row N, convex row C, tail row T, and weight ledger W are the only rows from which a convex approximant can be built.

The weak information is finite. The observer ledger O lists exactly the bounded-linear tests read from the sequence row, and the separation row S is the only route that turns the failure of those finite tests to separate the convex hull from a norm ball into an admissible approximation claim.

The strong side is downstream of that refusal. The row A records the norm estimate and R records the convex-combination readback, while H, Q, P, L transport, replay, source, and name the same packet. Since the excluded host objects are not carrier coordinates, the listed rows exhaust the local certificate. □

Theorem 7235.3 (Mazur lemma weak-to-strong convex handoff)**.** *Every strong approximation exported by an accepted* $\mathsf{MazurLemma}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{VecSpace}^{\uparrow}, \mathsf{Norm}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow} \longmapsto$$

Thus a consumer may read a norm-close convex combination only after the finite tail window, convex-weight ledger, bounded-linear observer tests, separation-refusal row, readback row, and norm-approximation row have all been displayed.

Proof. Start with a strong-approximation request. The carrier first exposes the normed vector-space row X, the weak sequence row N, the convex set row C, and the finite tail window T.

The convex combination is then determined by W. Because W records only finitely many rational nonnegative weights and their displayed sum-one certificate, the approximant is internal to the finite convex hull named by C and T.

Weak convergence is used only through the observer ledger O. The separation row S turns the finite observer comparison into a refusal of a displayed norm-ball separation, and only after that refusal may the readback row R and approximation row A be inspected.

The structural rows H, Q, P, L keep the handoff inside the same BHist packet. No host weak topology, infinite convex hull, selected subsequence, quotient equality, countable choice, or ambient completeness row is exported. □

Falsifiable prediction: Every accepted $\mathsf{MazurLemma}^{\uparrow}$ strong-approximation read factors through a displayed normed vector-space source, convex tail window, finite rational weight ledger, bounded-linear observer ledger, finite separation-refusal row, convex-combination readback, norm-approximation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: vecspace, norm, convexset, boundedlinearoperator, finite-dimensional-hahn-banach-separation: none is bijective with $\mathsf{MazurLemma}^{\uparrow}$ because this packet binds weak observer data, finite convex-combination ledgers, separation refusal, strong norm readback, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MazurLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{MazurLemma}^{\uparrow}$ packet is generated from a normed vector-space source, weak sequence row, convex tail window, rational weight ledger, bounded-linear observer ledger, finite separation-refusal row, convex-combination readback, norm-approximation row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MazurLemma}^{\uparrow}}$, seed)

Scope closed: Definition 7235.1, Definition 7235.2, and Definition 7235.3 seed the finite convex-combination ledger route from weak observer rows to strong norm approximants.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MazurLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host weak topology, an infinite convex hull, selected subsequences, quotient vector equality, countable choice, ambient Banach-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite convex-weight ledger exactness, bounded-linear observer exposure, finite separation-refusal replay, strong norm readback, transport, provenance, and local naming for BEDC.Derived.MazurLemmaUp.

Chapter 7236

A Concrete Naming Certificate for $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$

Following Adams–Fournier and Evans on Sobolev trace theorems, the $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ packet records the finite boundary-read window through which a Sobolev-class object on a compact metric domain exposes trace data to Real-facing consumers. It sits after the compact source of chapter 129, the continuous representative route of chapter 130, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only a compact-window trace carrier; it does not assert a full Sobolev embedding theorem, a measure-theoretic trace space, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7236.1 (Sobolev trace compact-window carrier)**.** *A* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-window domain row, carrying the* $\mathsf{CompactMetric}^{\uparrow}$ *source, the finite boundary neighbourhoods inspected by the trace request, and the compact metric window in which all reads occur.*
2. *B is the boundary trace row, recording the finite side of the trace request, the boundary labels, and the local restriction data consumed by the packet.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the domain and boundary observations opened before trace data is read.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximations attached to the interior and boundary observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the dyadic precision requests and comparisons used by the trace window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the trace values and tolerance comparisons visible after K, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the compact window, trace row, stream windows, rational readback, dyadic tolerances, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a trace-window request through the same compact, boundary, stream, rational, dyadic, and Real rows.*
9. *P is the* Pkg *provenance over* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SobolevTraceCompactWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, S, Q, D, E, H, C, P, N$. It has no coordinate for a measure-theoretic trace space, Sobolev embedding, quotient function equality, open-cover compactness, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SobolevTraceCompactWindowUp.SobolevTraceCompactWindowTasteGate single carrier alignment`

Theorem 7236.2 (Sobolev trace compact-window NameCert obligations)**.** *For every accepted* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ *carrier $T = (K, B, S, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SobolevTraceCompactWindow}}$ *surface consists exactly of carrier admission for the compact-window domain, boundary trace row, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise transport, displayed replay, provenance, and local naming rows; compact-window exposure through K; boundary-row exposure through B; finite-window and rational readback through S, Q; dyadic tolerance accounting through D; Real-seal nonescape through E; and preservation of the same route through H, C, P, N.*

Proof. Project the carrier of Definition 7236.1. Its visible rows are exactly $K, B, S, Q, D, E, H, C, P, N$, so every local obligation is read from one of those rows.

The analytic boundary side is finite. The compact metric window K and boundary trace row B are exposed before any rational or Real-facing consumer reads trace values.

The readback side is downstream. Stream windows S, regular rational readback Q, dyadic tolerances D, and the Real seal E supply the only numerical data available to the trace request.

Finally H, C, P, N transport, replay, source, and name the same route. Because the carrier stores no measure-theoretic trace space, Sobolev embedding theorem, quotient function equality, open-cover compactness principle, selected representative, or ambient completeness row, the listed rows exhaust the local NameCert obligations. □

Theorem 7236.3 (Sobolev trace compact-window trace route)**.** *Every boundary trace read accepted by a* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ *packet must first expose*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $K, B, S, Q, D, E, H, C, P, N$. The trace consumer receives a compact metric domain window, a boundary trace row, finite StreamName observations, regular rational readback, dyadic tolerance comparisons, and a terminal Real seal; it receives no measure-theoretic trace space, global Sobolev embedding, quotient function equality, selected representative, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 7236.2. It requires carrier admission and compact-window exposure before any boundary trace read can inspect the packet.

The boundary side is the pair K, B. These rows bind the trace request to the displayed compact metric source and to the finite boundary labels, so a consumer cannot enter through an untracked host trace operator.

The numerical side is S, Q, D, E. These rows expose finite windows, rational readback, dyadic tolerances, and the terminal Real seal in that order. The remaining rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. □

Theorem 7236.4 (Sobolev trace compact-window compact handoff)**.** *Every compact-metric or Real-facing consumer of an accepted* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ *packet reads the compact-window route*

$$K, B, S, Q, D, E$$

before consuming a boundary trace value. The handoff remains a finite NameCert route over $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows; it does not open an ambient compactness theorem, a measure-theoretic trace construction, quotient equality of representatives, countable choice, or ambient Real completeness.*

Proof. Use Definition 7236.1. The compact-facing side is the displayed domain window K together with the boundary trace row B.

Apply Definition 7236.3. It orders the finite StreamName observations, RegSeqRat readback, dyadic tolerance row, and Real seal after the compact and boundary rows.

The structural rows H, C, P, N do not add analytic content; they only preserve the already displayed packet. Hence every compact handoff remains inside the named finite route and cannot import the excluded host objects. □

Theorem 7236.5 (Sobolev trace compact-window nonescape)**.** *Every accepted* $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ *public read factors through the displayed rows*

$$K, B, S, Q, D, E, H, C, P, N.$$

Thus the packet exposes only a compact-window trace NameCert over $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. It cannot export a full Sobolev embedding theorem, a measure-theoretic trace space, open-cover compactness, quotient function equality, countable choice, a selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7236.1. The public packet has exactly the ten displayed rows $K, B, S, Q, D, E, H, C, P, N$.

Apply Definition 7236.2 and Definition 7236.4. They force every trace-facing read to pass through the compact, boundary, finite-window, rational-readback, dyadic-tolerance, and Real-seal rows before structural transport and provenance are consumed.

The excluded analytic objects are not carrier coordinates and are not introduced by the structural rows. Therefore every public read remains inside the displayed compact-window trace route. □

Falsifiable prediction: Every accepted $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ consumer displays a CompactMetric domain window, a boundary trace row, StreamName finite observations, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any boundary trace value is consumed.

Independence witness: compactmetric, continuousmap, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{SobolevTraceCompactWindow}^{\uparrow}$ because this packet alone binds a compact-window boundary trace row to the finite Real-facing readback route.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevTraceCompactWindow}^{\uparrow}}$.

Constructive story: A SobolevTraceCompactWindow$^{\uparrow}$ packet is generated from a compact metric domain window, a boundary trace row, finite StreamName observations, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SobolevTraceCompactWindow}^{\uparrow}}$, seed)

Scope closed: Definition 7236.1, Definition 7236.2, Definition 7236.3, Definition 7236.4, and Definition 7236.5 seed the finite compact-window trace route over compact, continuous-map, stream, rational-readback, dyadic-tolerance, and Real-seal data.

Formal verification: FormalStatus(NameCert$_{\mathsf{SobolevTraceCompactWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, full Sobolev embedding, measure-theoretic trace theory, open-cover compactness, quotient function equality, countable choice, selected representatives, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, boundary trace exactness, finite-window readback, dyadic tolerance accounting, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SobolevTraceCompactWindowUp.

Chapter 7237

A Concrete Naming Certificate for $\mathsf{DarstSelection}^{\uparrow}$

Following Darst and Bishop–Bridges constructive analysis, the $\mathsf{DarstSelection}^{\uparrow}$ packet records regulated bounded-variation compactness selection as finite NameCert data. It sits over the regulated source surface of chapter 1554, the bounded-variation budget of chapter 5933, the compact oscillation handoff of chapter 6818, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a sibling of $\mathsf{HellySelection}^{\uparrow}$ and $\mathsf{HellySelectionCompactVariation}^{\uparrow}$: it uses the same finite selection discipline, but its public route is regulated-function oscillation plus bounded real variation rather than a free host compactness theorem.

Definition 7237.1 (Darst selection carrier)**.** *A* $\mathsf{DarstSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (G, V, O, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the regulated source row, read through* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *and exposing only displayed step-window and endpoint-control data.*
2. *V is the* $\mathsf{BoundedRealVariation}^{\uparrow}$ *budget row, carrying the finite partition variation bound consumed by the selection route.*
3. *O is the* $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$ *handoff row, recording the compact oscillation windows that bind the regulated source to the bounded-variation budget.*
4. *W is the selector-window row. It stores the finite retained windows used by the Darst selection read, not a host-selected subsequence.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained selector windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the selected real-name windows.*
7. *H records componentwise* hsame *transport for G, V, O, W, Q, E.*
8. *C records displayed* Cont *replay from regulated source exposure through bounded-variation budgeting, compact oscillation handoff, selector windows, regular readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DarstSelection}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{BoundedRealVariation}^{\uparrow}$, $\mathsf{RegulatedFunctionCompactOscillation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DarstSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, V, O, W, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, a host-selected subsequence, a completed function-space quotient, detached real equality, or an ambient compactness principle.

Theorem 7237.2 (Darst selection NameCert obligations)**.** *For every accepted* $\mathsf{DarstSelection}^{\uparrow}$ *carrier*

$$D = (G, V, O, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarstSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated source exposure through G, bounded-variation budget exposure through V, compact oscillation handoff through O, selector-window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7237.1. The regulated input row G, variation budget row V, compact oscillation row O, selector window row W, regular readback row Q, and Real seal row E are the only analytic rows visible to the classifier.

The structural rows then close the certificate surface. The row H transports the displayed coordinates componentwise, C replays the route in the same order, P records the package provenance, and N names the local packet.

Since no carrier coordinate stores an open cover, a choice-selected subsequence, a host function-space compactness theorem, detached equality of real functions, or an ambient compactness principle, the listed rows exhaust the NameCert obligations. □

Theorem 7237.3 (Darst selection regulated compact handoff)**.** *Every compact selection read from an accepted* $\mathsf{DarstSelection}^{\uparrow}$ *packet factors through the ordered route*

$$\mathsf{RegulatedRealFunction}^{\uparrow} \longmapsto \mathsf{BoundedRealVariation}^{\uparrow} \longmapsto \mathsf{RegulatedFunctionCompactOscillation}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

Concretely, the consumer must first expose the regulated source row G, then the bounded-variation budget V, then the compact oscillation handoff O, then the selector windows W, then the regular rational readback Q, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal E with H, C, P, N attached.*

Proof. Begin with the source side. The regulated source row G supplies finite step-window and endpoint-control data, while the bounded-variation budget V supplies only the displayed partition budget needed by the selection route.

The compactness handoff is the row O. It binds the regulated source and variation budget to compact oscillation windows before any selector window can be consumed.

The selected output is then read through W, Q, E in order. The selector row W records retained finite windows, Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E seals the corresponding $\mathsf{Real}^{\uparrow}$ windows.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Because the carrier has no additional analytic coordinate, every compact selection read stays inside the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{DarstSelection}^{\uparrow}$ consumer must expose the regulated source row, bounded-variation budget, compact oscillation handoff, selector window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal before reading a Darst selection output.

Independence witness: helly-selection, helly-selection-compact-variation, bounded-real-variation, regulated-real-function, regulated-function-compact-oscillation: $\mathsf{DarstSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries regulated source data, bounded-variation budgeting,

compact oscillation handoff, selector windows, regular readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DarstSelection}\uparrow}$.

Constructive story: A $\mathsf{DarstSelection}^{\uparrow}$ packet is generated from a RegulatedRealFunction source row, a BoundedRealVariation budget row, a RegulatedFunctionCompactOscillation handoff row, selector windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarstSelection}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7237.1, Definition 7237.2, and Definition 7237.3 seed the finite regulated bounded-variation selection packet over compact oscillation handoff, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarstSelection}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequence choice, host function-space compactness, quotient equality of functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated source exposure, bounded-variation budget exposure, compact oscillation handoff, selector-window exactness, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.DarstSelectionUp.

Chapter 7238

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$

Following the Bishop–Bridges finite-net proof of compact uniform continuity, the CompactUniformModulusNormalFo packet records the normal-form route that turns compact coverage, local continuity rows, and a rational lower-bound fold into a tolerance-only uniform-modulus handoff. It realizes the compact-uniform-continuity route of chapter 9237 through the metric source of chapter 57, the compact carrier of chapter 60, the continuous-map rows of chapter 130, the rational fold substrate of chapter 37, and the consumer handoff of chapter 369. The certificate is finite NameCert data; it is not an open-cover compactness theorem, a host Heine–Cantor principle, a selected global minimum outside the fold, quotient equality of functions, or a hidden modulus function.

Definition 7238.1 (Compact uniform modulus normal-form carrier)**.** *A* $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, K, G, N, L, R, E, U, H, C, P, A)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row for the displayed domain and codomain distances.*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, exposing a finite* ProbeBundle *coverage window.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, read only through local pointwise-modulus data at the bundled centers.*
4. *N is the finite-net normal-form row, pairing each bundled center with a local radius and target tolerance row.*
5. *L is the local-radius ledger indexed by the same finite net.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the nonzero local-radius ledger.*
7. *E is the metric-triangle error-budget row that moves from an arbitrary point to its displayed center.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced by R and E.*
9. *H stores componentwise* hsame *transport for metric, compact, continuous, finite-net, local-radius, rational-fold, error-budget, uniform-modulus, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a compact-uniform request through $M, K, G, N, L, R, E, U, H, P, A$.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *A is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open-cover proof, countable choice, a host-selected minimum, quotient function equality, an ambient compactness predicate, or a hidden modulus.

Theorem 7238.2 (Compact uniform modulus normal-form obligations)**.** *For every accepted* CompactUniformModu *carrier*

$$F = (M, K, G, N, L, R, E, U, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNormalForm}^{\uparrow}}$ *surface consists exactly of metric admission through M, compact finite-net admission through K, continuous-map local-modulus exposure through G, finite-net normal-form alignment through N, L, rational lower-bound exactness through R, metric error-budget transport through E, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through A.*

Proof. Project the carrier of Definition 7238.1. The metric and compact rows M, K supply the finite net, while the continuous-map row G supplies only pointwise local-modulus data at the displayed centers.

The normal-form rows are N, L, R. They align the centers with local radii, keep the radius ledger finite, and fold the displayed positive rational radii to one global lower-bound row. No selected minimum outside the fold is present.

The consumer handoff is then forced through E and U. The error-budget row transports the center estimates to arbitrary points, and the uniform-modulus row records the tolerance-only output. Rows H, C, P, A only transport, replay, source, and name the same packet, so the listed obligations exhaust the certificate. □

Theorem 7238.3 (Compact uniform finite-net normal-form exhaustion)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$ *packet factors through the finite normal-form route*

$$\mathsf{Met}^{\uparrow}, \mathsf{Compact}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The route exposes a finite net, local pointwise moduli at the net centers, a rational lower-bound fold, metric-triangle transport, and a tolerance-only uniform-modulus row before any consumer may read the output.

Proof. Begin with a compact-uniform request. The metric row M fixes the distance comparisons, and the compact row K exposes the finite ProbeBundle coverage window.

Read the continuous-map row G only at the bundled centers. The normal-form rows N and L pair those center reads with local-radius data, so every local modulus request has a visible finite-net index.

Apply the rational fold row R. It is the only row that turns the finite list of local radii into a global lower bound, and it is still a $\mathsf{Rat}^{\uparrow}$ ledger rather than a host minimum.

Finally use the error-budget row E and the uniform-modulus row U. The structural rows H, C, P, A preserve this route, so no open-cover compactness, selected minimum, quotient equality, ambient compactness theorem, or hidden modulus can be exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$ uniform-modulus read factors through displayed Metric, Compact, ContinuousMap, Rat lower-bound fold, UniformModulus, ProbeBundle, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: metric, compact, continuousmap, rat, uniformmodulus: none is bijective with $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$ because this packet binds finite-net coverage, center-local modulus rows, rational lower-bound folding, metric error-budget transport, uniform-modulus export, replay, provenance, and local naming as one normal-form certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusNormalForm}^{\uparrow}$ packet is generated from Metric source rows, Compact finite-net rows, ContinuousMap local-modulus rows, finite-net normal-form ledgers, Rat lower-bound folds, metric error-budget transport, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7238.1, Definition 7238.2, and Definition 7238.3 seed the finite-net lower-bound fold normal form for compact uniform modulus extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, host Heine–Cantor semantics, selected minima outside the fold, quotient function equality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net normal-form alignment, local-modulus exposure, rational lower-bound fold exactness, metric error-budget transport, uniform-modulus export, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusNormalFormUp.

Chapter 7239

A Concrete Naming Certificate for $\mathsf{RationalDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges, the rational dense embedding into the constructive real line is read through rational source data, finite tolerance windows, and regular Cauchy handoff rows rather than through an ambient density theorem. The $\mathsf{RationalDenseEmbedding}^{\uparrow}$ packet sits between the rational carrier of chapter 37, the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the equality discipline of chapter 1820. It supplies the finite connector by which rational approximation data are consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ without importing host real equality, quotient representatives, selected limits, or ambient completeness.

Definition 7239.1 (Rational dense embedding carrier)**.** *A* $\mathsf{RationalDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, D, W, S, R, H, T, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row, recording the rational candidate and the rational order data visible to the embedding read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic budget used to compare the rational source against a requested real window.*
3. *W is the dense-witness window row, recording the finite interval request and the displayed rational candidate admitted inside that window.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row through which the finite window is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attaching the rational and dyadic observations to a regular rational name surface.*
6. *H is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, D, W, S, R have been displayed.*
7. *T is the componentwise* hsame *transport row for the rational source, dyadic tolerance, witness window, stream observation, regular readback, real seal, and structural rows.*
8. *C is the displayed* Cont *replay route by which a consumer rereads the same dense-embedding packet through the same rows.*
9. *P is the* Pkg *provenance row over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalDenseEmbedding}^{\uparrow}}$ *row.*

The carrier exports no host real equality, no quotient representative, no trichotomy branch, no selected limit, no countable-choice schedule, and no ambient completeness coordinate.

Theorem 7239.2 (Rational dense embedding NameCert obligations)**.** *For an accepted* $\mathsf{RationalDenseEmbedding}^{\uparrow}$ *carrier* $E = (Q, D, W, S, R, H, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalDenseEmbedding}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the* $\mathsf{Rat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, dense-witness window,* $\mathsf{StreamName}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming data;*
2. *dense-window exactness, stating that each accepted rational candidate is read inside the displayed finite request window and not from a hidden density oracle;*
3. *regular-rational handoff, stating that each accepted read reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the displayed rational, dyadic, witness-window, and stream-observation rows;*
4. *real-seal ordering, stating that the* $\mathsf{Real}^{\uparrow}$ *seal is consumed only after* Q, D, W, S, R *have been displayed;*
5. *nonescape, stating that the packet contains no host real equality, quotient representative, trichotomy branch, selected limit, countable-choice schedule, or ambient completeness row.*

Proof. Carrier admission is the tuple displayed in Definition 7239.1. Its rows name one rational source, one dyadic tolerance, one dense-witness window, one finite stream observation, one regular rational handoff, one Real seal, and the structural rows for transport, replay, provenance, and local naming.

Dense-window exactness is read from W together with Q and D. The rational candidate and dyadic tolerance are displayed before the finite request window is certified, so the packet has no hidden density oracle or host completeness row.

Regular-rational handoff and real-seal ordering are the roles of S, R, H. The stream observation exposes the finite window, the RegSeqRat row records the regular rational readback, and only then may the Real seal be consumed.

The nonescape clause follows from row exhaustion. The provenance may cite $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ as an equality-discipline boundary, but no row stores host real equality, quotient representatives, trichotomy, selected limits, countable choice, or ambient completeness. □

Theorem 7239.3 (Rational dense embedding RegSeqRat handoff)**.** *Every accepted dense-embedding read into* $\mathsf{Real}^{\uparrow}$ *factors through the displayed* $\mathsf{Rat}^{\uparrow}$ *source row* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, dense-witness window* W*,* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* H *is consumed. The exported route is finite certificate data for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers; it exports no host real equality, quotient representative, trichotomy branch, selected limit, countable-choice schedule, or ambient completeness principle.*

Proof. Project the accepted carrier. The first available source data are the rational candidate in Q and the dyadic tolerance budget in D.

Next read the dense witness through W and S. The window row records the finite interval request and accepted rational candidate, while the StreamName observation exposes only the displayed finite window.

Now pass through R before consuming H. This records the regular rational name surface used by $\mathsf{RegSeqRat}^{\uparrow}$ and makes the terminal $\mathsf{Real}^{\uparrow}$ seal a consequence of the displayed certificate route rather than of an external density theorem.

The remaining rows transport, replay, source, and name the same route. Since the carrier has no host equality, quotient, trichotomy, selected-limit, choice, or completeness coordinate, every accepted dense-embedding read remains inside the Rat/DyadicRatCore/StreamName/RegSeqRat/Real corridor. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalDenseEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7239.1, Definition 7239.2, and Definition 7239.3 seed the rational dense embedding route through Rat, DyadicRatCore, StreamName, RegSeqRat, Real, CauchyEquivalenceSetoid, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient real representatives, trichotomy, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-window exactness, RegSeqRat handoff, Real seal ordering, and local NameCert registration for BEDC.Derived.RationalDenseEmbeddingUp.

Constructive story: A $\mathsf{RationalDenseEmbedding}^{\uparrow}$ packet is generated from a Rat source row, a DyadicRatCore tolerance row, a dense-witness window, a StreamName observation row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7240

A Concrete Naming Certificate for $\mathsf{ProofEntropyLedger}^{\uparrow}$

The $\mathsf{ProofEntropyLedger}^{\uparrow}$ packet realizes the theorem-DNA scoring surface of chapter 9460 as finite ledger data. It sits beside the score carrier of chapter 6243, the theorem-genome evolution packet of chapter 6358, the proof trace normal form of chapter 5157, the audit-gate boundary of chapter 526, and the certificate audit gate of chapter 1002. The packet exists so scalar theorem quality reads cannot detach from theorem-flow locus masses, recognizer mass, a public weight profile, and a displayed score-bound ledger.

Definition 7240.1 (Proof entropy ledger carrier)**.** *A* $\mathsf{ProofEntropyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, R, W, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the theorem-flow source row, exposing the canonical theorem candidate and its site before any scalar read is public.*
2. *M is the theorem-DNA locus-mass row, recording the eight displayed masses for statement, dependencies, proof, certificates, ledger, status, canonical site, and closing seal.*
3. *R is the recognizer-mass row attached to the generated theorem recognizer.*
4. *W is the public weight-profile row, a nine-coordinate profile for the eight theorem-DNA loci together with recognizer mass.*
5. *S is the score-profile row, recording the componentwise products and the displayed scalar total.*
6. *A is the score-bound audit row, exposing the accepted bound, comparison result, and refusal of any scalar read that omits M, R, W, S.*
7. *E is the closing seal row, tying the score-bound ledger to the theorem-DNA closing seal and audit status.*
8. *H records componentwise* hsame *transport for T, M, R, W, S, A, E.*
9. *C records displayed* Cont *replay from theorem-flow exposure through locus masses, recognizer mass, public weights, score profile, score-bound audit, and closing seal.*

10. *P is the* Pkg *provenance over* ProofEntropyLedger$^{\uparrow}$, TheoremDnaScore$^{\uparrow}$, TheoremGenomeEvolution$^{\uparrow}$, ProofTraceNormalForm$^{\uparrow}$, AuditGateBoundary$^{\uparrow}$, CertificateAuditGate$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofEntropyLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, R, W, S, A, E, H, C, P, N$. It has no coordinate for a private weight profile, hidden recognizer mass, score-only theorem quality, unledgered audit acceptance, or an external theorem-value oracle.

Theorem 7240.2 (Proof entropy ledger NameCert obligations)**.** *For every accepted* ProofEntropyLedger$^{\uparrow}$ *carrier*

$$L = (T, M, R, W, S, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProofEntropyLedger}^{\uparrow}}$ *surface consists exactly of theorem-flow exposure through T, theorem-DNA locus masses through M, recognizer mass through R, public weight profile through W, score profile through S, score-bound audit through A, closing seal through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7240.1. The theorem-flow source T, locus-mass row M, recognizer-mass row R, public weight profile W, score profile S, score-bound audit A, and closing seal E are the only scoring rows.

The support rows then preserve the same route. Componentwise transport is H, displayed replay is C, provenance is P, and local naming is N.

Since the carrier has no private-weight, score-only, hidden-recognizer, or external theorem-value coordinate, every certified scalar read must expose the listed rows. □

Theorem 7240.3 (Proof entropy ledger weight-profile nonescape)**.** *Every scalar score read from an accepted* ProofEntropyLedger$^{\uparrow}$ *packet must factor through the ordered ledger route*

$$T, M, R, W \longmapsto S \longmapsto A \longmapsto E.$$

In particular, the score profile S is public only together with the theorem-DNA locus masses M, recognizer mass R, and public weight profile W, and the score-bound audit A cannot be replayed as a scalar theorem-quality judgment after erasing the weight profile.

Proof. Start with the theorem-flow source row T. It fixes the theorem candidate whose locus masses are being read, so the score cannot float free of a canonical theorem site.

Next expose M, R, W. These rows give the eight theorem-DNA masses, the recognizer mass, and the public nine-coordinate weight profile before the scalar score profile is formed.

The row S is therefore downstream from the mass and weight rows. The audit row A can compare the displayed total with the displayed bound only after that downstream score profile is present, and the seal row E closes the same ledger.

Transport, replay, provenance, and local naming preserve this order through H, C, P, N. A consumer that erases W no longer has the row required to reconstruct or audit the scalar score, so no weight-detached score read escapes the packet. □

Falsifiable prediction: Every accepted ProofEntropyLedger$^{\uparrow}$ read must expose theorem-flow locus masses, recognizer mass, public weight profile, score bound, audit status, closing seal, transport, replay, provenance, and local naming before any scalar quality read is public.

Independence witness: theorem-dna-score, theorem-genome-evolution, proof-trace-normal-form, auditgateboundary, certificateauditgate: $\mathsf{ProofEntropyLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a public weight profile and score-bound ledger as first-class rows.

The Lean follow-up must provide `Nontrivial` and `FieldFaithful` instances in `lean4/BEDC/Derived/ProofEnt` preserving the theorem-flow source, locus-mass profile, recognizer-mass profile, public weight profile, score-profile row, score-bound audit row, and closing seal as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofEntropyLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofEntropyLedger}^{\uparrow}$ packet is generated from theorem-flow source data, theorem-DNA locus masses, recognizer mass, a public weight profile, a score profile, a score-bound audit row, a closing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofEntropyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7240.1, Definition 7240.2, and Definition 7240.3 seed the finite theorem-DNA score ledger whose scalar read remains attached to public mass and weight rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofEntropyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close theorem-DNA scoring optimality, classifier-shift discovery, hidden recognizer soundness, global theorem quality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-flow exposure, locus-mass exactness, recognizer-mass exactness, public weight-profile exactness, score-bound nonescape, audit status, closing seal, transport, replay, provenance, local naming, and TasteGate instances for BEDC.Derived.ProofEntropyLedgerUp.

Chapter 7241

A Concrete Naming Certificate for $\mathsf{CompactIntervalIntermediateValue}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalIntermediateValue}^{\uparrow}$ packet records the constructive intermediate-value bisection route over a closed real interval as finite NameCert data. It consumes the compact interval finite-net surface of chapter 3969, the dyadic sign-change route of chapter 2203, the finite sign ledger of chapter 1422, the located real interval bisection route of chapter 5546, the displayed graph surface of chapter 130, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite bisection certificate; it does not assert classical Heine–Borel compactness, selected roots, quotient equality of real names, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7241.1 (Compact interval intermediate-value carrier)**.** *A* $\mathsf{CompactIntervalIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, F, B, D, R, E, H, C, P, N)$$

with the following rows:

1. *I is the closed bounded interval source row, read through* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *and located endpoint data;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, exposing only finite graph windows and a displayed continuity budget;*
3. *B is the sign-bracket and bisection row, consuming* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *and* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *branch data;*
4. *D is the dyadic tolerance row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and sign-read budget;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained midpoint and interval-window schedule;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed limiting name;*
7. *H records componentwise* hsame *transport of I, F, B, D, R, E;*
8. *C records displayed* Cont *replay of the same bisection route;*
9. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalIntermediateValue}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactIntervalIntermediateValue}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open-cover compactness principle, a selected root, quotient real equality, excluded middle over signs of arbitrary reals, or an ambient completeness theorem.

Lean4 (statement) `BEDC.Derived.CompactIntervalIntermediateValueUp`

Theorem 7241.2 (Compact interval intermediate-value NameCert obligations)**.** *For the carrier of Definition 7241.1, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalIntermediateValue}^{\uparrow}}$ *surface consists exactly of carrier admission for $V = (I, F, B, D, R, E, H, C, P, N)$, interval-source admission through I, continuous graph admission through F, bracket stability through B, dyadic descent and tolerance control through D, regular rational readback through R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the finite packet. The interval row I supplies only the closed bounded interval and its finite compactness-facing data, while the graph row F supplies the displayed continuity budget needed to interpret finite sign reads.

Read the bisection rows next. The sign-bracket row B keeps the lower and upper sign entries attached to the retained subinterval, and the dyadic row D records the mesh, midpoint, and tolerance descent used by the branch ledger. These rows are finite and do not select an unbounded branch outside the displayed schedule.

The readback and seal rows then close the real-facing surface. Row R turns the retained windows into a $\mathsf{RegSeqRat}^{\uparrow}$ schedule, and row E seals only the resulting displayed real name. Finally H, C, P, N transport, replay, source, and name the same packet, so no excluded host principle can be introduced after the finite bisection route has been named. □

Falsifiable prediction: Every accepted compact-interval intermediate-value read must display a closed interval source, continuous graph row, sign-bracket bisection row, dyadic tolerance row, regular rational readback, Real seal, transport, replay, provenance, and local NameCert evidence before any zero-facing real name is exported.

Independence witness: closed-bounded-interval-compact, dyadic-intermediate-value, intermediate-value-bisection, located-real-interval-bisection, continuousmap, regseqrat, real, dyadicratcore: CompactIntervalInter is not bijective to any listed sibling because it jointly carries interval source, graph, bracket, dyadic descent, readback, seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalIntermediateValue}^{\uparrow}$ packet is generated from a closed interval source, continuous graph row, sign-bracket bisection row, dyadic tolerance row, regular rational readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactIntervalIntermediateValue}^{\uparrow}}$, seed)

Scope closed: Definition 7241.1 and Definition 7241.2 seed the finite compact-interval bisection certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactIntervalIntermediateValue}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.CompactIntervalIntermediateValueUp`

Bridge status: none

Not claimed: This chapter does not close classical Heine–Borel compactness, selected roots, quotient equality of real names, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, bracket stability, bisection descent, dyadic readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalIntermediateValueUp.

Chapter 7242

A Concrete Naming Certificate for MetaCICNormalizationSubstitutionBoundary$^{\uparrow}$

The MetaCICNormalizationSubstitutionBoundary$^{\uparrow}$ packet realizes the substitution edge of the MetaCIC normalization audit map as finite NameCert rows. It sits at the conditional boundary of chapter 9383, beside the normalization frontier of chapter 892, the candidate-mediated strong-normalization route of chapter 5163, the substitution audit map of chapter 9380, the confluence diamond ledger of chapter 6547, and the subject-reduction obstruction of chapter 9377. The packet names the visible substitution-boundary row that must be consumed before candidate-mediated normalization can be read as a closed-term route; it does not assert evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, or host evaluator soundness.

Definition 7242.1 (MetaCIC normalization substitution-boundary carrier)**.** *A* MetaCICNormalizationSubstitutionB *carrier is a finite* BHist *packet*

$$S = (F, C, S_b, D, O, H, R, P, N)$$

with the following rows:

1. *F is the normalization frontier row, retaining the candidate-mediated strong-normalization guard;*
2. *C is the candidate-evidence row, recording the candidate, closed-candidate, or candidate-set interface being consumed;*
3. *S_b is the visible substitution-boundary row, recording the substitution audit data that cannot be erased;*
4. *D is the confluence diamond ledger row, exposing the residual-substitution and local-join data used at the boundary;*
5. *O is the subject-reduction obstruction row, retaining the dependent-codomain and closedness-preservation obstruction data;*
6. *H records componentwise* hsame *transport of F, C, S_b, D, O;*
7. *R records displayed* Cont *replay of the same boundary route;*
8. *P is the* Pkg *provenance over* MetaCICNormalizationSubstitutionBoundary$^{\uparrow}$*, chapter 9383, chapter 9380,* MetaCICNormalizationFrontier$^{\uparrow}$*,* CandidateMediatedSN$^{\uparrow}$*,* MetaCICConfluenceDiamondLedger$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}}$ *row.*
The carrier has no coordinate for unconditional normalization, evidence-free closed-term strong normalization, unrestricted substitution compatibility, completed subject reduction, global confluence, or host quotient equality.

Lean4 (statement) `BEDC.Derived.MetaCICNormalizationSubstitutionBoundaryUp`

Theorem 7242.2 (MetaCIC normalization substitution-boundary NameCert obligations)**.** *For the carrier of Definition 7242.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* $S = (F, C, S_b, D, O, H, R, P, N)$*, normalization-frontier retention through* F*, candidate-interface admission through* C*, visible substitution-boundary admission through* S_b*, confluence-diamond ledger exposure through* D*, subject-reduction obstruction retention through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* R*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The frontier row F and candidate row C keep the normalization read inside the candidate-mediated interface recorded by the audit map.

Read the boundary rows next. The substitution row S_b is visible before the confluence row D or obstruction row O can be consumed, so residual-substitution and subject-reduction data cannot be hidden behind a global normalization statement.

The structural rows H, R, P, N then transport, replay, source, and name the same finite packet. Since no carrier coordinate supplies evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, or host evaluator soundness, the NameCert surface remains exactly the displayed substitution-boundary route. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}$ read must expose the visible substitution-boundary row S_b before candidate-mediated normalization can consume confluence or subject-reduction obstruction data.

Independence witness: metacicnormalizationfrontier, candidate-mediated-sn, metacic-substitution-audit-map, metacic-confluence-diamond-ledger, metacic-subject-reduction-obstruction: MetaCICNormalizationSubs is not bijective to any listed sibling because it jointly carries frontier, candidate, substitution-boundary, confluence-ledger, subject-reduction-obstruction, transport, replay, provenance, and local naming rows.

The Lean boundary must eventually provide $\mathsf{Nontrivial}\ \mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}$ instances. These instances must keep $F, C, S_b, D, O, H, R, P, N$ as separate displayed fields before a checked theorem may replace the statement marker.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}$ packet is generated from normalization frontier, candidate evidence, visible substitution-boundary, confluence diamond ledger, subject-reduction obstruction, componentwise transport, displayed continuation replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7242.1 and Definition 7242.2 seed the finite substitution-boundary packet for candidate-mediated normalization reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationSubstitutionBoundary}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MetaCICNormalizationSubstitutionBoundaryUp`

Bridge status: none

Not claimed: This chapter does not close evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked frontier retention, candidate-interface admission, substitution-boundary visibility, confluence-ledger exposure, subject-reduction obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalizationSubstitutionBoundaryUp.

Chapter 7243

A Concrete Naming Certificate for $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$

Following Bishop and Bridges on constructive compact spaces and located subsets, the $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ packet records Cantor space as a prefix-located binary stream object whose finite cylinders can be inspected before any compactness consumer reads a cover. It sits beside the finite stream-window interface of chapter 126, the binary prefix carrier of chapter 1475, the Baire prefix route of chapter 334, and the Cantor compactness surface of chapter 6999. The packet names a traditional constructive Cantor-space object; it is not an ambient powerset, a selected infinite branch, a quotient stream equality, an open-cover compactness theorem outside displayed cylinders, or a countable-choice principle.

Definition 7243.1 (Bishop located Cantor space carrier)**.** *A* $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, S, C, L, V, R, H, Q, P, N)$$

with the following displayed rows:

1. *B is the binary-prefix row, naming the finite words in* $\mathsf{Bool}^{<\omega}$ *inspected by the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, recording the finite stream windows whose prefixes are being compared.*
3. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *cylinder row, recording the basic clopen cylinder determined by each displayed prefix.*
4. *L is the located-decision row, exposing finite evidence that a queried prefix cylinder is inside, outside, or still separated from the displayed located subobject.*
5. *V is the prefix-stability row, requiring the located answer to be preserved by every displayed extension of the queried prefix.*
6. *R is the located cylinder-coverage row, listing the finite cylinder cover visible at the current precision request.*
7. *H records componentwise* hsame *transport for binary prefixes, stream windows, cylinder rows, located decisions, prefix stability, coverage, replay, provenance, and local naming.*
8. *Q records displayed* Cont *replay of the route*

$$B, S, C, L, V \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCantorSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, C, L, V, R, H, Q, P, N$. It has no coordinate for host stream equality, selected branches, open-cover compactness, powerset membership, quotient cylinders, or countable choice.

Theorem 7243.2 (Bishop located Cantor space NameCert obligations)**.** *For every accepted* BishopLocatedCantorSp *carrier $K = (B, S, C, L, V, R, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCantorSpace}^{\uparrow}}$ *surface consists exactly of binary-prefix admission, finite stream-window exposure, Cantor cylinder exposure, located-decision display, prefix-stability display, located cylinder coverage, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7243.1. The rows $B, S, C, L, V, R, H, Q, P, N$ are the complete local inventory.

The located decision is prefix-local. The binary row B, stream row S, and cylinder row C fix the finite prefix window before L is read, and V keeps the answer stable under displayed prefix extensions.

The coverage row R is downstream of the same finite route. The structural rows H, Q, P, N transport, replay, source, and name that route, and none supplies selected branches, quotient stream equality, powerset membership, open-cover compactness outside the displayed cylinders, or countable choice. □

Theorem 7243.3 (Bishop located Cantor space prefix stability)**.** *Every located-cylinder read of an accepted* $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ *packet factors through*

$$B, S, C, L, V.$$

If two displayed stream windows have the same queried prefix, then the located answer is transported only by the prefix-stability row V and the componentwise row H. The read cannot appeal to host stream equality, a selected infinite branch, or quotient equality of cylinders.

Proof. Start from the accepted carrier. The queried prefix is named in B, the finite stream window is named in S, and the corresponding cylinder is named in C before any located answer is available.

The row L supplies the located decision for that displayed cylinder. Prefix stability V then records the permitted transport along displayed extensions of the same prefix.

Finally H preserves the same rows componentwise. Since the carrier has no coordinate for host stream equality, selected branches, or quotient cylinder equality, the located-cylinder read remains prefix-local. □

Theorem 7243.4 (Bishop located Cantor space located cylinder coverage)**.** *Every compactness-facing consumer of an accepted* $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ *packet receives only the finite located cylinder route*

$$B, S, C, L, V \longmapsto R \longmapsto H, Q, P, N.$$

The consumer may read the displayed CantorSpace and StreamName rows, the located decision, the prefix-stability row, and the finite coverage row. It receives no arbitrary open cover, selected subcover outside the displayed prefixes, host powerset, quotient stream equality, or countable-choice schedule.

Proof. Use Definition 7243.3. It places every located-cylinder read through the displayed binary prefix, stream window, Cantor cylinder, located decision, and stability rows.

The coverage row R is formed only after those rows are visible. Thus a compactness-facing consumer can inspect a finite cylinder list at the current precision request, but cannot replace it by an arbitrary open cover or a selected subcover supplied externally.

The replay, provenance, and local naming rows preserve the same finite packet. Therefore the public read remains a located cylinder coverage route over $\mathsf{CantorSpace}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$, with no hidden branch selector or quotient stream equality. □

Theorem 7243.5 (Bishop located Cantor space ledger refusal)**.** *No accepted* $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ *packet exports host stream equality, arbitrary powerset membership, selected infinite branches, open-cover compactness detached from displayed cylinders, quotient equality of cylinders, or countable choice. Every exported read is confined to the rows* $B, S, C, L, V, R, H, Q, P, N$.

Proof. Project the carrier. Its rows are the binary prefix, stream window, Cantor cylinder, located decision, prefix-stability, located coverage, transport, replay, provenance, and local naming rows.

Apply the located cylinder coverage theorem. It exhausts compactness-facing reads by the displayed finite cylinder route, so none of the refused host principles has an export coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCantorSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCantorSpace}^{\uparrow}$ packet is generated from binary prefixes, StreamName windows, CantorSpace cylinders, located-decision rows, prefix-stability rows, finite located cylinder coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCantorSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7243.1, Definition 7243.2, Definition 7243.3, Definition 7243.4, and Definition 7243.5 seed the finite located Cantor-space cylinder route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCantorSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host stream equality, arbitrary powerset membership, selected branches, open-cover compactness outside displayed cylinders, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, prefix stability, located cylinder coverage, refusal exactness, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCantorSpaceUp.

Chapter 7244

A Concrete Naming Certificate for $\mathsf{PoincareHorocycleTransport}^{\uparrow}$

The $\mathsf{PoincareHorocycleTransport}^{\uparrow}$ packet realizes the horocycle transport branch of chapter 9463 as finite disk, boundary-shadow, horocycle, transport, Real-seal, provenance, and NameCert rows. It sits after the disk automorphism surface of chapter 6776, the horocycle carrier of chapter 3534, the boundary-shadow route of chapter 3813, the visual-boundary readout of chapter 4181, and the observer-expansion packet of chapter 4167. The packet names finite horocycle transport; it is not a completed hyperbolic plane, quotient ideal boundary, selected geodesic ray, physical observer trajectory, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7244.1 (Poincare horocycle transport carrier)**.** *A* $\mathsf{PoincareHorocycleTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, B, O, Y, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *disk-transport row.*
2. *B is the boundary-shadow row, read through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *and the visible boundary packet.*
3. *O is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *row, recording the boundary-tangent horocycle being transported.*
4. *Y is the observer-expansion row that records the finite observer-facing read of the transported horocycle.*
5. *M is the metric or visual-shadow comparison row that binds the disk transport, boundary shadow, and horocycle rows to one finite budget.*
6. *R is the refusal row excluding quotient boundary equality, selected rays, unlogged physical observer trajectories, host isometry semantics, and hidden completeness.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed comparison budget.*
8. *H records componentwise* hsame *transport for disk, boundary-shadow, horocycle, observer-expansion, comparison, refusal, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through*

$$D, B, O, Y, M, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* PoincareHorocycleTransport$^{\uparrow}$, PoincareDiskAutomorphism$^{\uparrow}$, HyperbolicBoundaryShad HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicHorocycle$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{PoincareHorocycleTransport}^{\uparrow}}$ *row.*

The classifier accepts only packets whose rows $D, B, O, Y, M, R, E, H, C, P, N$ remain visible on the same finite route.

Theorem 7244.2 (Poincare horocycle transport NameCert obligations)**.** *For every accepted* PoincareHorocycleTran *carrier $T = (D, B, O, Y, M, R, E, H, C, P, N)$, the local* NameCert$_{\text{PoincareHorocycleTransport}^{\uparrow}}$ *surface consists exactly of disk-transport admission, boundary-shadow exposure, horocycle exposure, observer-expansion exposure, comparison-budget exposure, refusal exactness, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7244.1. The rows $D, B, O, Y, M, R, E, H, C, P, N$ are the whole local inventory.

The geometric source rows are D, B, O, Y, M. They bind the Poincare disk transport, boundary shadow, horocycle, observer-facing read, and comparison budget before the Real-facing seal can be inspected.

The refusal row R blocks quotient boundary equality, selected rays, physical observer trajectories, host isometry semantics, and hidden completeness. The structural rows H, C, P, N preserve the same finite route, so the listed obligations exhaust the certificate. □

Theorem 7244.3 (Poincare horocycle transport boundary-shadow handoff)**.** *Every accepted* PoincareHorocycleTran *Real-facing read factors through*

$$D, B, O, Y, M, R \longmapsto E \longmapsto H, C, P, N.$$

The boundary-shadow row B and horocycle row O must be displayed before the comparison budget M and the seal E are read. A consumer missing either row is outside the accepted horocycle transport packet.

Proof. Start from the carrier. The disk row D supplies the Poincare transport, but it does not by itself identify the boundary-shadow or horocycle data.

Read the rows B and O before M. These rows attach the visible boundary shadow and the boundary-tangent horocycle to the same transport budget, while Y records the observer-facing read.

The refusal and Real-seal rows are downstream. Transport, replay, provenance, and local naming preserve the same route, so omitting B or O leaves the carrier rather than producing a weaker accepted packet. □

Theorem 7244.4 (Poincare horocycle transport nonescape)**.** *No accepted* PoincareHorocycleTransport$^{\uparrow}$ *packet exports a completed hyperbolic plane, quotient ideal boundary, selected geodesic ray, physical observer trajectory, host isometry semantics, or ambient* Real$^{\uparrow}$ *completeness. Every exported read is confined to $D, B, O, Y, M, R, E, H, C, P, N$.*

Proof. Project the accepted carrier. Its rows are exactly the disk transport, boundary shadow, horocycle, observer expansion, comparison budget, refusal, Real seal, transport, replay, provenance, and local naming rows.

Apply the boundary-shadow handoff theorem. It forces every Real-facing consumer through the displayed finite route before the seal is read, so the refused host objects have no carrier coordinate and cannot be exported. □

Falsifiable prediction: A claimed $\mathsf{PoincareHorocycleTransport}^{\uparrow}$ packet that reaches the $\mathsf{Real}^{\uparrow}$ seal without displaying both the boundary-shadow row B and the horocycle row O is rejected as a horocycle transport packet.

Independence witness: poincare-disk-automorphism, hyperbolic-horocycle, hyperbolic-boundary-shadow, hyperbolic-visual-boundary, hyperbolic-observer-expansion: none is bijective to PoincareHorocycleTranspor because the transport packet binds disk transport, boundary shadow, horocycle, observer-expansion read, comparison budget, refusal, Real seal, transport, replay, provenance, and local NameCert rows as one finite route.

The Lean TasteGate route must provide explicit $\mathsf{Nontrivial\ PoincareHorocycleTransport}^{\uparrow}$ and $\mathsf{FieldFaithful\ PoincareHorocycleTransport}^{\uparrow}$ instances before the chapter can be treated as a checked discovery-facing surface. The nontrivial witness separates a packet with both B and O from a refused disk-only transport, and field faithfulness compares exactly $D, B, O, Y, M, R, E, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareHorocycleTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareHorocycleTransport}^{\uparrow}$ packet is generated from Poincare disk transport, boundary-shadow rows, horocycle rows, observer-expansion rows, comparison-budget rows, refusal exactness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareHorocycleTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7244.1, Definition 7244.2, Definition 7244.3, and Definition 7244.4 seed the finite Poincare horocycle transport route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareHorocycleTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic planes, quotient ideal boundaries, selected geodesic rays, physical observer trajectories, host isometry semantics, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-transport admission, boundary-shadow exposure, horocycle exposure, observer-expansion exposure, comparison-budget exactness, refusal, RealUp sealing, TasteGate nontriviality, field faithfulness, replay, provenance, and local naming for BEDC.Derived.PoincareHorocycleTransportUp.

Chapter 7245

A Concrete Naming Certificate for $\mathsf{LocallyFiniteCover}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{LocallyFiniteCover}^{\uparrow}$ packet records the finite BEDC ledger by which a displayed open-cover family is read through local finite windows. It sits over the topology carrier of chapter 88, the open-cover request surface of chapter 6731, the finite refinement comparison of chapter 3430, the paracompact refinement carrier of chapter 1963, and the partition-of-unity consumer of chapter 3369. The packet names only displayed cover cells, local neighbourhood windows, finite incidence reads, refinement assignments, transport, replay, provenance, and local naming; it does not choose an arbitrary cover, infer paracompactness, assert compactness of open covers, or export partition-of-unity weights.

Definition 7245.1 (Locally finite cover carrier)**.** *A* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, A, U, W, I, R, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the open-row operations and finite neighbourhood reads on which the cover ledger is interpreted.*
2. *A is the* $\mathsf{OpenCover}^{\uparrow}$ *request row, listing the displayed cover cells and the source region they are meant to cover.*
3. *U is the neighbourhood-window row. For each displayed point or source window it records an open neighbourhood row over T before any local incidence check is read.*
4. *W is the finite-window row, listing the finitely many cover cells from A allowed to meet the neighbourhood row in U.*
5. *I is the incidence ledger, recording the displayed nonempty-meet checks between the neighbourhood windows and the cover cells named by W.*
6. *R is the refinement-assignment row, tying the local finite ledger either to the original cover request A or to a displayed refinement route read through* $\mathsf{Paracompact}^{\uparrow}$*.*
7. *F is the finite-refinement handoff row consumed by* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *when a compact or completion-facing argument needs a finite comparison window.*

8. *H records componentwise* hsame *transport for the topology source, open-cover request, neighbourhood windows, finite windows, incidence ledger, refinement assignments, finite-refinement handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a cover request through $T, A, U, W, I, R, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocallyFiniteCover}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{OpenCover}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyFiniteCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-indexed cover family, a selected infinite refinement, arbitrary subcover choice, open-cover compactness, partition-of-unity weights, quotient open equality, or ambient point-set topology.

Lean4 (statement) `BEDC.Derived.LocallyFiniteCoverUp`

Theorem 7245.2 (Locally finite cover NameCert obligations)**.** *For every accepted* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *carrier*

$$L = (T, A, U, W, I, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocallyFiniteCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, open-cover request, neighbourhood windows, finite windows, incidence ledger, refinement assignment, finite-refinement handoff, transport, replay, provenance, and local naming;*
2. *open-cover exposure, requiring every local finiteness read to start with the displayed cover request A over the topology row T;*
3. *neighbourhood-window exactness, requiring every finite incidence claim to be read through a displayed open neighbourhood row U;*
4. *finite-window locality, requiring W to list all and only the cover cells inspected by the local incidence ledger I for the corresponding neighbourhood window;*
5. *refinement handoff exactness, requiring R and F to expose only assignments back to A, paracompact refinement rows, or finite refinement comparison rows already displayed in the packet;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve exactly the listed rows.*

No local locally-finite-cover obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 7245.1. The accepted tuple displays exactly the topology source, open-cover request, neighbourhood windows, finite windows, incidence ledger, refinement assignment, finite-refinement handoff, transport, replay, provenance, and local naming.

Open-cover exposure is forced by row order. A local finiteness request first reads T and then the displayed cover row A, so the packet never begins from an unlogged host-indexed family.

Neighbourhood-window exactness is the projection to U. The finite incidence row cannot be inspected until the corresponding open neighbourhood has been displayed over the same topology source.

Finite-window locality is the projection to W and I. The row W lists the finite cells allowed to meet the displayed neighbourhood, and I records only the incidence checks for that finite list.

The handoff rows R and F keep the ledger usable by the neighbouring refinement packets. They expose assignments back to A, paracompact refinement rows when present, and finite comparison windows for compact or completion-facing consumers.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores host-indexed covers, selected infinite refinements, arbitrary subcover choice, open-cover compactness, partition-of-unity weights, quotient open equality, or ambient point-set topology, no such obligation can be exported by the local certificate. □

Theorem 7245.3 (Locally finite cover finite-window locality)**.** *Every accepted* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *carrier exports each local incidence read through the finite route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{OpenCover}^{\uparrow} \longrightarrow U \longrightarrow W \longrightarrow I.$$

Thus a point-neighbourhood consumer sees only the displayed neighbourhood row, the finite list of cover cells meeting that neighbourhood, and the recorded incidence checks for those cells. It does not receive an arbitrary host-indexed cover, a selected infinite subfamily, or an open-cover compactness theorem.

Proof. Project the carrier of Definition 7245.1. The rows T and A expose the topology and cover request before any local window is available.

The local window is U, and the finite list attached to it is W. The incidence ledger I is meaningful only after both rows have been displayed.

The structural rows H, C, P, N only transport, replay, source, and name this same route. The excluded host data are absent from the carrier, so every point-neighbourhood consumer receives exactly the finite route above. □

Theorem 7245.4 (Locally finite cover refinement handoff)**.** *Every refinement-facing consumer of an accepted* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *packet must read the assignment rows*

$$A,\ U,\ W,\ I,\ R,\ F$$

before citing the ledger as a paracompact refinement source, a finite cover comparison source, or a partition-of-unity support source. The handoff records displayed local finite windows and assignments only; it does not create paracompactness, finite subcover extraction, or partition-of-unity weights.

Proof. Begin with the local route from Definition 7245.3. It exposes the source cover, neighbourhood windows, finite visible cells, and incidence checks.

The assignment row R ties those local windows back to the cover request or to a paracompact refinement row already present in the packet. The finite-refinement row F exports only the finite comparison surface needed by compact or completion-facing consumers.

A partition-of-unity consumer may therefore cite the locally finite ledger as support data, but only after the finite windows and assignments have been displayed. Since no row stores paracompactness as an external theorem, finite subcover extraction, or weights, the handoff cannot export those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyFiniteCover}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7245.1, Definition 7245.2, Definition 7245.3, and Definition 7245.4 seed the finite topology, open-cover, neighbourhood-window, incidence, refinement-assignment, finite-handoff, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-indexed covers, selected infinite refinements, arbitrary subcover choice, open-cover compactness, partition-of-unity weights, quotient open equality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-cover exposure, neighbourhood-window exactness, finite-window locality, refinement handoff exactness, transport stability, replay stability, and non-escape proofs for BEDC.Derived.LocallyFiniteCoverUp.

Constructive story: A LocallyFiniteCover$^{\uparrow}$ packet is generated from a Topology$^{\uparrow}$ source row, an OpenCover$^{\uparrow}$ request row, displayed neighbourhood windows, finite visible-cell windows, an incidence ledger, refinement-assignment rows, a FiniteCoverRefinement$^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7246

A Concrete Naming Certificate for $\mathsf{KannanContractionFixedPoint}^{\uparrow}$

Following Kannan's fixed point theorem and Bishop–Bridges constructive metric analysis, the $\mathsf{KannanContractionFixedPoint}^{\uparrow}$ packet records a non-Lipschitz contraction endpoint route as finite NameCert data. It sits next to the complete-metric carrier of chapter 128, the ordinary contraction-map packet of chapter 1418, the Picard orbit ledger of chapter 2359, the fixed-point endpoint discipline of chapter 396, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the finite Kannan adjacent-distance handoff; it does not select a host fixed point, identify quotient orbit tails, use countable choice, assume trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7246.1 (Kannan contraction fixed-point carrier)**.** *A* $\mathsf{KannanContractionFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, A, O, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the pointed complete metric source, displaying the carrier, metric comparison rows, chosen starting point, and complete-metric handoff.*
2. *F is the self-map row, recording the displayed endomap whose Picard orbit is inspected.*
3. *A is the Kannan adjacent-distance inequality row, exposing a rational coefficient below one half and the comparison*
$$d(Fx, Fy) \leq a \cdot (d(x, Fx) + d(y, Fy))$$
only through displayed metric rows.
4. *O is the Picard orbit window row, displaying the finite orbit windows generated by repeatedly reading F from the chosen point.*
5. *U is the Cauchy modulus row obtained from the adjacent-distance ledger and the visible Picard windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational tail estimates consumed by the completion-facing route.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the complete-metric source, self-map row, adjacent-distance inequality, orbit windows, Cauchy modulus, and regular-rational readback are visible.*

8. *H records componentwise* hsame *transport for source, map, inequality, orbit, modulus, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M, F, A, O, U, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* KannanContractionFixedPoint$^{\uparrow}$, CompleteMetric$^{\uparrow}$, ContractionMapping$^{\uparrow}$, PicardIteration$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{KannanContractionFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, F, A, O, U, R, E, H, C, P, N$. It has no coordinate for selected host fixed points, quotient orbit equality, countable choice, trichotomy, or ambient real completeness.

Theorem 7246.2 (Kannan contraction fixed-point obligations)**.** *For every accepted* KannanContractionFixedPoint$^{\uparrow}$ *carrier $K = (M, F, A, O, U, R, E, H, C, P, N)$, the local* NameCert$_{\text{KannanContractionFixedPoint}^{\uparrow}}$ *surface consists exactly of pointed complete-metric source exposure through M; self-map admission through F; Kannan adjacent-distance inequality exposure through A; finite Picard orbit windows through O; Cauchy modulus extraction through U;* RegSeqRat$^{\uparrow}$ *readback through R;* Real$^{\uparrow}$ *seal non-escape through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier. The rows M, F, A, O are the complete source surface: a pointed complete metric object, the self-map, the Kannan adjacent-distance inequality, and the finite Picard windows.

The Cauchy handoff is displayed rather than chosen. The row U records the modulus obtained from the adjacent-distance ledger, and R reads it as regular rational tail data before the terminal seal E is available.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no selected host fixed point, quotient orbit equality, countable-choice selector, trichotomy principle, or ambient real completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7246.3 (Kannan Picard-to-Cauchy handoff)**.** *Every accepted* KannanContractionFixedPoint$^{\uparrow}$ *endpoint read factors through the displayed Picard orbit windows, the adjacent-distance Kannan ledger, the Cauchy modulus row, the* RegSeqRat$^{\uparrow}$ *readback row, and the* Real$^{\uparrow}$ *seal before it can be consumed by a fixed-point endpoint discipline. The route is adjacent to chapter 2359 and chapter 396, but its contraction evidence is the Kannan inequality row rather than a Lipschitz ratio row.*

Proof. Start from Definition 7246.2. It orders the complete-metric source, self-map, adjacent-distance inequality, and finite Picard windows before the modulus and regular-rational readback rows.

The endpoint read is therefore completion-facing only after U and R are visible. Transport and replay preserve this order, and provenance names the route as a KannanContractionFixedPoint$^{\uparrow}$ packet rather than as an ordinary ContractionMapping$^{\uparrow}$ packet.

Thus any consumer of the seal must pass through the displayed Picard-to-Cauchy handoff. The refused selected fixed point, quotient orbit equality, countable-choice selector, trichotomy principle, and ambient completeness principle have no export coordinate. □

Falsifiable prediction: An accepted $\mathsf{KannanContractionFixedPoint}^{\uparrow}$ endpoint read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the pointed complete-metric source, self-map row, Kannan adjacent-distance inequality, Picard windows, Cauchy modulus, $\mathsf{RegSeqRat}^{\uparrow}$ readback, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: complete-metric, contractionmapping, picard-iteration, banach-fixed-point, regseqrat, real: no listed sibling is bijective to $\mathsf{KannanContractionFixedPoint}^{\uparrow}$ because the Kannan packet binds the adjacent-distance inequality, Picard orbit windows, Cauchy modulus handoff, regular-rational readback, Real seal, transport, replay, provenance, and local naming as one fixed-point certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{KannanContractionFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{KannanContractionFixedPoint}^{\uparrow}$ packet is generated from a pointed complete metric source, a self-map row, a Kannan adjacent-distance inequality row, Picard orbit windows, a Cauchy modulus row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KannanContractionFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7246.1, Definition 7246.2, and Definition 7246.3 seed the finite Kannan fixed-point handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KannanContractionFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected host fixed points, quotient orbit equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked complete-metric source exposure, self-map admission, Kannan adjacent-distance inequality, Picard orbit windows, Cauchy modulus extraction, RegSeqRat readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.KannanContractionFixedPointUp.

Chapter 7247

A Concrete Naming Certificate for $\mathsf{FiniteOpenCover}^{\uparrow}$

Following Bourbaki and Kelley on finite open covers, the $\mathsf{FiniteOpenCover}^{\uparrow}$ packet records a cover whose indexing family is already finite before any compactness or subcover-selection consumer inspects it. It sits between the general open-cover surface of chapter 6731, the open-set rows of chapter 88, the compact-metric source rows of chapter 129, the finite-subcover criterion of chapter 3503, and the dyadic interval cover carrier of chapter 1428. The packet is not a selected subcover and not an open-cover compactness theorem; it is the finite carrier row through which already finite cover data is named.

Definition 7247.1 (Finite open-cover carrier)**.** *A* $\mathsf{FiniteOpenCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, U, M, R, K, D, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row, listing exactly the cover indices available to the current consumer.*
2. *U is the* $\mathsf{Topology}^{\uparrow}$ *open-set row, assigning an open certificate to each displayed index of I.*
3. *M is the membership and coverage row, tying each covered point read to one displayed index and one open row.*
4. *R is the finite refinement row, comparing finite cover windows by finite index maps and inclusion witnesses.*
5. *K is the compact-source compatibility row, read only when a* $\mathsf{CompactMetric}^{\uparrow}$ *consumer supplies a compact metric source.*
6. *D is the dyadic-interval handoff row, present only when the finite cover is displayed through a* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *endpoint, mesh, or membership ledger.*
7. *H records componentwise* hsame *transport for index, open-set, membership, refinement, compact-source, dyadic-handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite cover request through $I, U, M, R, K, D, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenCover}^{\uparrow}$, $\mathsf{OpenCover}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteSubcoverCriterion}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, U, M, R, K, D, H, C, P, N$. It has no coordinate for an arbitrary open-cover compactness theorem, a selected finite subcover, a host powerset of opens, quotient cover equality, countable choice, or ambient compactness outside the displayed finite rows.

Theorem 7247.2 (Finite open-cover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOpenCover}^{\uparrow}$ *carrier* $F = (I, U, M, R, K, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index, topology open-set, membership, refinement, compact-source, dyadic-handoff, transport, replay, provenance, and local naming rows; finite-index exactness through I; open-set exactness through U; coverage and refinement exactness through M and R; compact-metric compatibility through K; dyadic-interval handoff through D; and non-escape through H, C, P, N.*

Proof. Project the carrier of Definition 7247.1. The rows I, U, M, R expose the only finite cover window, open predicates, coverage witnesses, and refinement data available to the packet.

The compatibility rows are downstream. A compact-metric consumer reads K only after the finite cover rows are displayed, while a dyadic-interval consumer reads D only as the finite endpoint, mesh, or membership handoff already named by $\mathsf{DyadicIntervalCover}^{\uparrow}$.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for arbitrary open-cover compactness, selected subcovers, host powersets, quotient cover equality, countable choice, or ambient compactness, the listed rows exhaust the local NameCert obligations. □

Theorem 7247.3 (Finite open-cover dyadic interval handoff)**.** *Every* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *consumer that reads a* $\mathsf{FiniteOpenCover}^{\uparrow}$ *packet must display the finite index row I, the* $\mathsf{Topology}^{\uparrow}$ *open row U, the membership row M, the refinement row R, and the dyadic handoff row D before any interval-cover endpoint, mesh, or membership read is exported. The handoff factors through the finite route*

$$\mathsf{FiniteOpenCover}^{\uparrow} \longrightarrow \mathsf{DyadicIntervalCover}^{\uparrow} \longrightarrow \mathsf{FiniteSubcoverCriterion}^{\uparrow}$$

only as displayed cover data. It exports no selected finite subcover, open-cover compactness theorem, host compactness principle, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite cover rows of Definition 7247.1. The interval consumer first sees I, U, M, R, so the cover window is finite before any dyadic endpoint or mesh information is consulted.

Read the dyadic handoff through D. This row may point to the $\mathsf{DyadicIntervalCover}^{\uparrow}$ endpoint, mesh, or membership ledger, but it does so only after the finite open-cover rows have been displayed and transported by H.

The finite-subcover criterion can then consume the displayed cover data as a finite route. The packet contains no selector row and no compactness theorem row, so the handoff cannot export selected subcovers, host compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOpenCover}^{\uparrow}$ packet is generated from finite index rows, Topology open-set rows, membership and refinement rows, compact-source compatibility, dyadic interval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteOpenCover}^{\uparrow}}$, seed)

Scope closed: Definition 7247.1, Definition 7247.2, and Definition 7247.3 seed the finite open-cover carrier and its dyadic interval handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteOpenCover}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, host compactness, quotient cover equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-index exactness, open-set exactness, coverage and refinement exactness, compact-metric compatibility, dyadic-interval handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOpenCoverUp.

Chapter 7248

A Concrete Naming Certificate for $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$ packet records the finite midpoint subdivision route for a displayed dyadic interval before regular-Cauchy, finite-net, and real-completion consumers may inspect a smaller mesh. It sits over the dyadic arithmetic core of chapter 291, the dyadic subdivision ledger of chapter 366, the dyadic interval tree route of **??**, the dyadic interval net of chapter 5966, the finite-net mesh refinement of chapter 6168, and the terminal real-name boundary of chapter 38. The packet is a finite L10 mesh-refinement certificate; it is not open-cover compactness, a selected infinite refinement path, quotient real equality, a host barycentric subdivision theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7248.1 (Dyadic barycentric subdivision carrier)**.** *A* $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, L, R, D, T, C, P, N)$$

with the following displayed rows:

1. *I is the parent dyadic interval row, exposing the lower endpoint, upper endpoint, endpoint order, and dyadic denominator budget through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *M is the midpoint row, recording the common-denominator normalization and the displayed barycentric midpoint of the two endpoint rows.*
3. *L and R are the left-child and right-child interval rows, with L using the lower endpoint and midpoint, and R using the midpoint and upper endpoint.*
4. *D is the mesh-decay row, recording that each child mesh bound is at most one half of the parent bound through the same dyadic denominator ledger.*
5. *T is the finite-net transport row, exposing the handoff from the two child intervals to* $\mathsf{DyadicIntervalNet}^{\uparrow}$*,* $\mathsf{DyadicSubdivision}^{\uparrow}$*, and* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ *consumers.*
6. *C records displayed* Cont *replay from a parent interval request through the midpoint, child intervals, mesh-decay, finite-net transport, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicSubdivision}^{\uparrow}$*,* $\mathsf{DyadicIntervalTree}^{\uparrow}$*,* $\mathsf{DyadicIntervalNet}^{\uparrow}$*,* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBarycentricSubdivision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, L, R, D, T, C, P, N$ and their componentwise hsame *transports. The carrier has no coordinate for an infinite subdivision sequence, selected limiting point, arbitrary open cover, quotient real equality, host interval compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7248.2 (Dyadic barycentric subdivision NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$ *carrier $B = (I, M, L, R, D, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicBarycentricSubdivision}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the parent dyadic interval, midpoint row, two child intervals, mesh-decay row, finite-net transport, replay, provenance, and local naming;*
2. *midpoint closure, requiring M to compute the displayed midpoint by* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-denominator arithmetic from the endpoints of I;*
3. *parent-cover exactness, requiring L and R to share the midpoint boundary and to cover exactly the parent interval row I;*
4. *mesh shrinkage, requiring D to bind the two child mesh bounds to the parent mesh by the displayed halving comparison;*
5. *finite-net handoff, requiring T to expose the two child intervals to the same dyadic subdivision, dyadic interval net, and total-boundedness mesh consumers;*
6. *transport, requiring componentwise* hsame *to preserve $I, M, L, R, D, T, C, P, N$ and no further interval object;*
7. *ledger, requiring C, P, N to replay, source, and name exactly the displayed packet; and*
8. *non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer to pass through the finite rows before reading any real-name seal.*

No barycentric-subdivision obligation is stored outside these rows, and the packet refuses open-cover compactness, quotient real equality, selected infinite refinement, host barycentric-subdivision semantics, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness as local data.*

Proof. Project the carrier of Definition 7248.1. The visible analytic rows are the parent interval I, midpoint row M, child intervals L, R, mesh-decay row D, and finite-net transport row T.

Midpoint closure is read from M. The row exposes the common denominator and the dyadic arithmetic ledger inherited from $\mathsf{DyadicRatCore}^{\uparrow}$, so the midpoint is not a host real midpoint or a selected point of a completed interval.

Parent-cover exactness is then a finite endpoint check. The left child reuses the lower endpoint and midpoint, the right child reuses the midpoint and upper endpoint, and the only shared boundary is the displayed midpoint row.

Mesh shrinkage is the row D. It records the halving comparison for the two children against the parent mesh bound, while T hands those children to the existing finite dyadic subdivision and finite-net routes.

Finally replay the structural rows C, P, N and transport componentwise. Since the packet has no additional coordinate for compactness, quotient equality, infinite refinement, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7248.3 (Dyadic barycentric subdivision mesh stability)**.** *Let $B = (I, M, L, R, D, T, C, P, N)$ and $B' = (I', M', L', R', D', T', C', P', N')$ be accepted* $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$ *carriers related componentwise by* hsame*. If B displays a midpoint choice, parent-cover exactness, and a mesh*

bound through M, L, R, D, then B' displays the transported midpoint choice, transported parent-cover exactness, and transported child mesh bound through M', L', R', D'.

Proof. Read the componentwise hsame transport between the two accepted carriers. The transport relates the parent interval rows, midpoint rows, child interval rows, mesh-decay rows, finite-net transport rows, replay rows, provenance rows, and local naming rows in the same order.

The midpoint choice is preserved first. Since the midpoint is named only by the row M, the transported carrier can read it only through M', with the same common-denominator dyadic arithmetic transported from the endpoint rows.

The coverage rows are preserved next. The left and right children of B share the midpoint boundary and cover the parent row; componentwise transport sends these endpoint incidences to L' and R' over the transported parent I'.

The mesh comparison is preserved by the transported decay row. Row D' is the image of D, so the halving bound attached to the children remains the child mesh bound available to any dyadic subdivision, interval-net, or total-boundedness consumer.

Thus B' exposes exactly the transported midpoint, coverage, and mesh data. It gains no selected limiting point, open-cover compactness, quotient real equality, or ambient completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBarycentricSubdivision}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBarycentricSubdivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7248.1, Definition 7248.2, and Definition 7248.3 seed the finite barycentric mesh-refinement route over parent dyadic intervals, midpoint rows, child intervals, mesh-decay rows, finite-net handoff, componentwise transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBarycentricSubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite refinement paths, quotient real equality, host barycentric-subdivision semantics, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint closure, parent-cover exactness, mesh shrinkage, finite-net handoff, transport, ledger, non-escape, and local naming for BEDC.Derived.DyadicBarycentricSubdivisionUp.

Constructive story: A $\mathsf{DyadicBarycentricSubdivision}^{\uparrow}$ packet is generated from a parent dyadic interval row, a DyadicRatCore midpoint row, two child interval rows, a mesh-decay row, finite-net transport toward DyadicSubdivision, DyadicIntervalNet, and RealIntervalTotalBoundedness consumers, displayed Cont replay, Pkg provenance, componentwise hsame transport, and a local NameCert row.

Chapter 7249

A Concrete Naming Certificate for CantorSpaceBaireCategory$^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the CantorSpaceBaireCategory$^{\uparrow}$ packet records the finite-cylinder face of the Baire-category argument on Cantor space. It sits beside the Cantor-space carrier of chapter 1475, the Baire-category theorem packet of chapter 3570, the choice-free Baire route of chapter 4123, the finite stream-window discipline of chapter 126, and the complete metric surface of chapter 128. The chapter names only the finite cylinder and dense-window handoff; it does not import countable choice, open-cover compactness, quotient equality, or ambient Real completeness.

Definition 7249.1 (CantorSpaceBaireCategory carrier)**.** *A* CantorSpaceBaireCategory$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, T, F, S, M, H, A, P, N)$$

with the following displayed rows:

1. *C is the* CantorSpace$^{\uparrow}$ *cylinder row, exposing finite Boolean prefixes, prefix projection, and cylinder stability;*
2. *T is the* BaireCategoryTheorem$^{\uparrow}$ *dense-open theorem row, exposing only the dense-window rows consumed by the category argument;*
3. *F is the* BaireCategoryChoiceFree$^{\uparrow}$ *refusal row, recording that the route is read without a countable-choice selector or chosen generic point;*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row used to list exactly the Cantor coordinates and dense-open requests inspected by a consumer;*
5. *M is the* CompleteMetric$^{\uparrow}$ *service row for the Cantor metric surface, available only as the displayed complete-metric handoff used by T;*
6. *H records componentwise* hsame *transport for the cylinder, dense-window, choice-free, stream-window, complete-metric, replay, provenance, and local naming rows;*
7. *A records displayed* Cont *replay from cylinder admission through dense-window inspection and terminal NameCert publication;*
8. *P is the* Pkg *provenance over* CantorSpaceBaireCategory$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireCategoryTheorem$^{\uparrow}$, BaireCategoryChoiceFree$^{\uparrow}$, StreamName$^{\uparrow}$, CompleteMetric$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{CantorSpaceBaireCategory}^{\uparrow}}$ *row.*
The carrier has no coordinate for a selected generic point, a countable-choice schedule, an open-cover compactness theorem, quotient equality of streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7249.2 (CantorSpaceBaireCategory NameCert obligations)**.** *For every accepted* CantorSpaceBaireCatego *carrier $K = (C, T, F, S, M, H, A, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorSpaceBaireCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cantor cylinder row, Baire-category dense-open row, choice-free refusal row, finite stream-window row, complete-metric service row, componentwise transport, displayed replay, provenance, and local naming. Every dense-window read must expose the finite cylinder row C, the theorem row T, the refusal row F, and the window row S before it can publish a Baire-category conclusion.*

Proof. Project the carrier of Definition 7249.1. The accepted packet displays exactly the rows $C, T, F, S, M, H, A, P, N$, so the local NameCert surface can read only those coordinates.

The analytic obligations are the first five rows. The Cantor cylinder row supplies finite Boolean prefixes, the Baire-category theorem row supplies dense-window evidence, the choice-free row refuses a generic-point selector, the StreamName row records inspected coordinates, and the complete-metric row supplies the service handoff consumed by the theorem route.

The structural obligations are H, A, P, N. They transport, replay, source, and name the same finite packet. Since no coordinate stores countable choice, open-cover compactness, quotient equality, or ambient Real completeness, those principles cannot discharge a local obligation. □

Theorem 7249.3 (CantorSpaceBaireCategory cylinder route)**.** *Every accepted* $\mathsf{CantorSpaceBaireCategory}^{\uparrow}$ *packet routes a Baire-category dense-window read through the finite path*

$$\mathsf{CantorSpace}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{CompleteMetric}^{\uparrow},\quad \mathsf{BaireCategoryTheorem}^{\uparrow},\quad \mathsf{BaireCategoryChoiceFree}^{\uparrow},\quad \mathsf{Name}$$

The route may shrink Cantor cylinders and inspect dense-open rows along the finite window S, but it cannot export countable choice, open-cover compactness, quotient equality, selected generic points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cantor cylinder row C. A consumer sees only finite Boolean prefix data, prefix projection, and cylinder stability inherited from the $\mathsf{CantorSpace}^{\uparrow}$ surface.

Next read the StreamName window S and the complete-metric service row M. The window records the coordinates and dense-open requests inspected by the consumer, while M provides only the complete-metric handoff used by the Baire-category theorem row.

Finally apply the theorem row T together with the choice-free row F. The Baire-category conclusion is published through H, A, P, N only after the finite cylinder, stream window, and complete-metric handoff remain visible. The refused principles are absent from the route, so the packet exports a finite-cylinder Baire-category handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{CantorSpaceBaireCategory}^{\uparrow}$ dense-window read factors through one $\mathsf{CantorSpace}^{\uparrow}$ finite cylinder row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CompleteMetric}^{\uparrow}$ service row, one $\mathsf{BaireCategoryTheorem}^{\uparrow}$ dense-open row, one $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cantorspace, bairecategorytheorem, bairecategorychoicefree, streamname, completemetric: $\mathsf{CantorSpaceBaireCategory}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cantor finite-cylinder row, the dense-open theorem row, the choice-free refusal row, the finite StreamName window, the complete-metric service handoff, transport, replay, provenance, and local NameCert row as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpaceBaireCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorSpaceBaireCategory}^{\uparrow}$ packet is generated from CantorSpace finite cylinders, BaireCategoryTheorem dense-open rows, BaireCategoryChoiceFree refusal evidence, StreamName finite windows, CompleteMetric service rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSpaceBaireCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7249.1, Definition 7249.2, and Definition 7249.3 seed the finite-cylinder Baire-category route from CantorSpace cylinders to dense-open Baire rows without choice or compactness imports.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSpaceBaireCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a public Cantor-space category theorem, Lean verification, a standard bridge, countable choice, open-cover compactness, quotient equality, selected generic points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dense-window exposure, cylinder-route determinacy, choice-free refusal, transport, replay, provenance, and local naming for BEDC.Derived.CantorSpaceBaireCategoryUp.

Chapter 7250

A Concrete Naming Certificate for BishopHeineBorelInterval$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the BishopHeineBorelInterval$^{\uparrow}$ packet records the finite-net interval compactness read that is available for a located real interval when the endpoints, tolerance rows, and net witnesses are all displayed. It sits on the finite route through DyadicRatCore$^{\uparrow}$ of chapter 291, RegSeqRat$^{\uparrow}$ of chapter 301, Real$^{\uparrow}$ of chapter 38, LocatedCompact$^{\uparrow}$ of chapter 1488, and TotallyBounded$^{\uparrow}$ of chapter 127: dyadic tolerances name the requested mesh, regular rational rows read endpoint and distance windows, the Real seal closes the endpoint names, located compactness supplies the inside-or-apart boundary, and total boundedness supplies the finite net. The packet is not the general open-cover Heine–Borel theorem; it exports no countable choice, quotient real equality, selected finite subcover, or ambient Real$^{\uparrow}$ completeness.

Definition 7250.1 (Located compactness finite-net extraction carrier)**.** *A* BishopHeineBorelInterval$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, D, R, L, T, F, H, C, P, N)$$

with the following displayed rows:

1. *E is the located interval endpoint row, naming the two* Real$^{\uparrow}$ *endpoint seals and the interval orientation check;*
2. *D is the positive* DyadicRatCore$^{\uparrow}$ *tolerance and mesh ledger used to request a finite net;*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row for endpoint windows, mesh comparisons, and distance estimates;*
4. *L is the* LocatedCompact$^{\uparrow}$ *locatedness row, exposing the finite inside-or-apart tests for the interval boundary;*
5. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row, exposing a closed* ProbeBundle *of interval centers,* InBundle *membership evidence, and coverage radii;*
6. *F is the extraction row pairing each displayed tolerance request with the corresponding finite list of centers and the coverage comparison read through D, R, L, T;*
7. *H records componentwise* hsame *transports for endpoints, dyadic tolerances, regular rational readbacks, locatedness, finite nets, extraction, replay, provenance, and naming;*

8. *C records displayed* Cont *replay from a mesh request through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, *and* $\mathsf{TotallyBounded}^{\uparrow}$ *before the extraction row is consumed;*
9. *P is the* Pkg *provenance over* $\mathsf{BishopHeineBorelInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopHeineBorelInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for an arbitrary open cover, a choice-selected subcover, quotient real equality, a host compact interval, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7250.2 (Located compactness finite-net extraction obligations)**.** *For every accepted* $\mathsf{BishopHeineBorelInterval}^{\uparrow}$ *carrier* $I = (E, D, R, L, T, F, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{BishopHeineBorelInterval}^{\uparrow}}$ *surface consists exactly of finite endpoint admission, dyadic mesh admission, regular rational readback, located-boundary testing, total-bounded finite-net exposure, extraction exactness, componentwise transport, displayed replay, provenance, and local naming. Thus the interval compactness read available to Real and Completion consumers is the finite-net extraction surface over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, *and* $\mathsf{TotallyBounded}^{\uparrow}$, *not an open-cover compactness principle, countable-choice schedule, quotient real equality, selected finite subcover, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted packet $I = (E, D, R, L, T, F, H, C, P, N)$. The endpoint row E, dyadic ledger D, and regular rational readback R name the interval and the requested mesh before any compactness consumer reads data from the packet.

Use the locatedness row L and finite-net row T. The located boundary is inspected by finite inside-or-apart comparisons, and the total-bounded surface supplies only closed ProbeBundle centers, InBundle membership rows, and coverage radii.

The extraction row F pairs those displayed requests with finite center data. Componentwise transport in H and displayed replay in C preserve the same route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, and $\mathsf{TotallyBounded}^{\uparrow}$.

Finally, P and N record provenance and local naming for exactly that route. Since the carrier contains no open-cover, choice, quotient-equality, selected-subcover, or ambient-completeness coordinate, the listed obligations exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopHeineBorelInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopHeineBorelInterval}^{\uparrow}$ packet is generated from displayed Real endpoint seals, a DyadicRatCoreUp mesh ledger, RegSeqRatUp readbacks, a LocatedCompactUp located-boundary row, a TotallyBoundedUp finite-net row, an extraction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopHeineBorelInterval}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 7250.1 and Definition 7250.2 close the paper-side finite-net extraction surface for located real intervals over the displayed Real, RegSeqRat, DyadicRatCore, LocatedCompact, and TotallyBounded route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopHeineBorelInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general open-cover Heine–Borel compactness, countable choice, quotient real equality, selected finite subcovers, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite-net extraction obligation theorem.

Chapter 7251

A Concrete Naming Certificate for $\mathsf{FiniteShiftAverage}^{\uparrow}$

Following Banach, Lorentz, and Bishop–Bridges finite averaging practice, the $\mathsf{FiniteShiftAverage}^{\uparrow}$ packet records a finite shift-window average over L10 real-name data as one NameCert carrier. It consumes the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy-name route of chapter 3413. The packet names only the finite averaging route for a displayed window; it does not select a Banach limit, quotient sequence class, ultrafilter, countable-choice subsequence, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness.

Definition 7251.1 (Finite shift average carrier)**.** *A* $\mathsf{FiniteShiftAverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, A, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the finite stream windows whose shifted entries are inspected.*
2. *W is the finite shift-window row, recording the window length, shift offsets, and local inclusion of every sampled entry.*
3. *A is the average ledger, storing the finite rational sum and denominator schedule before any real seal is consumed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision and rounding budget for the average readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the average ledger into a regular rational name.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed averaged name.*
7. *H records componentwise* hsame *transport for S, W, A, D, R, E.*
8. *C records displayed* Cont *replay from stream windows through finite averaging, dyadic tolerance, regular readback, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteShiftAverage}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteShiftAverage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an invariant mean, a Banach-limit value, quotient equality of sequences, an ultrafilter, a choice-selected subsequence, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.FiniteShiftAverageUp.FiniteShiftAverageTasteGate single carrier alignment`

Theorem 7251.2 (Finite shift average NameCert obligations)**.** *For every accepted* $\mathsf{FiniteShiftAverage}^{\uparrow}$ *carrier* $F = (S, W, A, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteShiftAverage}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *source row, finite shift-window admission through* W*, finite average ledger exactness through* A*, dyadic tolerance control through* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal discipline through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7251.1. The stream source S and shift-window row W list the only entries that can be sampled, so every average is finite before any real-facing read is available.

Read the average ledger next. Row A records the rational sum and denominator schedule, while D supplies the dyadic tolerance budget used to compare the finite sum with the requested precision.

The readback and seal rows then expose the L10 real-name route. Row R turns the finite ledger into a regular rational name, and E seals only that displayed name. Transport, replay, provenance, and local naming through H, C, P, N preserve the same rows, so no invariant mean or completion principle can enter after the finite average has been named. □

Theorem 7251.3 (Finite shift average Banach-limit boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteShiftAverage}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{StreamName}^{\uparrow},\ W,\ A,\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the rows $S, W, A, D, R, E, H, C, P, N$ *of Definition 7251.1. The consumer may use the finite shift-window average as L10 real-name data, but it cannot replace that route by a Banach limit, quotient sequence class, ultrafilter, countable-choice subsequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteShiftAverageUp.FiniteShiftAverageCarrier banach limit boundary`

Proof. Start with the NameCert obligations of Definition 7251.2. They require the stream window, shift window, finite average ledger, dyadic tolerance row, regular rational readback, and real seal to be displayed before any consumer reads the averaged name.

The finite route is already closed under the sibling readbacks. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{Cauchy}^{\uparrow}$ source rows expose the sampled entries, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance budget, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal.

Since the carrier has no coordinate for an invariant mean, quotient sequence class, ultrafilter, selected subsequence, or ambient completeness witness, none of those objects can be substituted for the finite ledger. The boundary is therefore a finite shift-window average route rather than a Banach-limit selection principle. □

Falsifiable prediction: Every accepted $\mathsf{FiniteShiftAverage}^{\uparrow}$ consumer must expose StreamName windows, the finite shift-window row, average ledger, DyadicRatCore tolerance row, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert evidence before any averaged real-name read is public.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy: $\mathsf{FiniteShiftAverage}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds source windows, finite shifts, averaging, dyadic tolerance, regular readback, real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteShiftAverage}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteShiftAverage}^{\uparrow}$ packet is generated from StreamName source windows, a finite shift-window row, a finite average ledger, a DyadicRatCore tolerance row, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteShiftAverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7251.1, Definition 7251.2, and Definition 7251.3 seed the finite shift-average certificate for L10 real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteShiftAverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Banach-limit existence, quotient sequence equality, ultrafilters, countable-choice subsequences, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, average-ledger exactness, dyadic tolerance control, regular readback, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.FiniteShiftAverageUp.

Chapter 7252

A Concrete Naming Certificate for $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$ packet names the finite oscillation criterion that a Darboux-integral consumer reads before accepting an integrability boundary. It sits next to the Darboux integral route of chapter 1460, the finite sum interfaces $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, and **??** of chapter 1880, chapter 5858, and chapter 4083, the oscillation packet of chapter 7161, and the regular rational and real-name endpoints $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ of chapter 301 and chapter 38. The packet records finite partition oscillation ledgers; it does not choose partitions by host choice, identify functions by quotient equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7252.1 (DarbouxOscillationCriterion carrier)**.** *A* $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, L, U, O, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *consumer row that requests a finite upper-minus-lower gap before reading an integral seal;*
2. *S is the common* $\mathsf{DarbouxSum}^{\uparrow}$ *row over the displayed partition surface;*
3. *L and U are the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *and* ***??*** *rows, both tied to the same cells and interval source;*
4. *O is the finite oscillation ledger, read through* $\mathsf{DarbouxOscillation}^{\uparrow}$*, that records the displayed cellwise upper-minus-lower bounds;*
5. *M is the finite partition and refinement row, naming the cells on which S, L, U, O are compared;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the tolerance and gap windows;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Darboux rows and regular rational readback have been displayed;*
8. *H records componentwise* hsame *transports for integral, sum, lower, upper, oscillation, partition, readback, seal, continuation, provenance, and naming rows;*
9. *C records displayed* Cont *replay for the same ordered route;*
10. *P is the* Pkg *provenance over* $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* $\mathsf{DarbouxSum}^{\uparrow}$*,* $\mathsf{LowerDarbouxSum}^{\uparrow}$*,* ***??****,* $\mathsf{DarbouxOscillation}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host suprema, quotient function equality, selected refining partitions, countable choice, or ambient real completeness.

Lean4 (statement) `BEDC.Derived.DarbouxOscillationCriterionUp`

Theorem 7252.2 (DarbouxOscillationCriterion NameCert obligations)**.** *For every accepted* DarbouxOscillationCrit *carrier $K = (I, S, L, U, O, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Darboux integral, common sum, lower-sum, upper-sum, oscillation, partition, regular readback, Real seal, transport, replay, provenance, and naming rows; lower-and-upper coherence over the same partition row M; oscillation-ledger exactness for O; regular tolerance readback through R; terminal Real-seal nonescape through E; and provenance exactness through P, N.*

Lean4 (definition) `BEDC.Derived.DarbouxOscillationCriterionUp.darbouxOscillationCriterionToEventFlow`

Proof. Project the carrier of Definition 7252.1. The displayed tuple is the whole certificate surface, so every obligation begins with one of $I, S, L, U, O, M, R, E, H, C, P, N$.

The Darboux rows are coherent because S, L, U, O are all read over the same partition and refinement row M. The lower and upper sum rows therefore produce a finite upper-minus-lower comparison before the integral row I may consume the criterion.

The readback and seal obligations are terminal. The row R supplies the regular rational tolerance windows, and E supplies the $\mathsf{Real}^{\uparrow}$ seal only after the Darboux and oscillation rows have been displayed.

Finally, H, C, P, N preserve, replay, source, and name the same route. Since no host supremum, selected refinement, quotient function equality, countable choice, or ambient completeness coordinate appears in the carrier, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}$ surface. □

Theorem 7252.3 (DarbouxOscillationCriterion partition route)**.** *Every* $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, *or ?? consumer of an accepted* $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$ *packet must read the finite route*

$$M \longmapsto (S, L, U, O) \longmapsto R \longmapsto E$$

with structural support H, C, P, N before it uses the criterion. Thus the criterion exposes a finite partition oscillation ledger and regular rational gap readback, not a selected mesh sequence, host supremum, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DarbouxOscillationCriterionUp.DarbouxOscillationCriterionPartitionRoute`

Proof. Start with the displayed partition row M. It is the only row that names the cells and refinements on which the Darboux rows can be compared.

Read the sum and oscillation data next. The rows S, L, U, O expose the common Darboux sum boundary, the lower and upper sides, and the upper-minus-lower oscillation ledger over that same finite partition surface.

Then pass through R and E. The regular rational readback supplies the finite tolerance windows, and the Real seal is reached only after those windows and Darboux rows have been consumed.

The structural rows H, C, P, N transport, replay, source, and name this route without adding any hidden analytic object. Therefore the partition route is a finite BEDC criterion surface rather than a host-level compactness, choice, supremum, or quotient principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxOscillationCriterion}^{\uparrow}$ packet is generated from DarbouxIntegral, DarbouxSum, LowerDarbouxSum, UpperDarbouxSum, DarbouxOscillation, finite partition, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7252.1, Definition 7252.2, and Definition 7252.3 seed the finite Darboux oscillation criterion over displayed partition, sum, oscillation, regular rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxOscillationCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host Darboux integrability theorem, selected refining partition sequence, quotient equality of functions, countable choice, ambient RealUp completeness, Lean verification of the obligation rows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper coherence, oscillation-ledger exactness, partition-route nonescape, regular readback, Real-seal nonescape, replay, provenance, and local naming for BEDC.Derived.DarbouxOscillationCriterionUp.

Chapter 7253

A Concrete Naming Certificate for $\mathsf{FinitePrefixStream}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FinitePrefixStream}^{\uparrow}$ packet records the finite prefix of a countable name inspected at a displayed unary depth. It sits between the countable-name surface of chapter 126 and the regular rational readback of chapter 301, while keeping its dyadic tolerance row anchored in chapter 291, its terminal real-facing seal anchored in chapter 38, and its completion-facing use anchored in chapter 1639. The diagonal-selection corridor of chapter 4080 consumes these prefix windows before selecting a regular Cauchy diagonal readback, so the diagonal handoff is fed by displayed finite observations rather than by a detached tail selector. The packet gives the Real and Completion route a named finite observation-prefix object, not quotient stream equality, selected limits, countable choice, host real equality, or ambient real completeness.

Definition 7253.1 (Finite prefix stream certificate carrier)**.** *A* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (k, W, D, R, H, C, P, N)$$

with the following displayed rows:

1. *k is the unary depth row. It fixes the largest inspected index and contains only the finite successor spine needed to reach that depth.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It displays exactly the stream-name observations at indices $0, \ldots, k$ and no infinite representative.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the finite dyadic radii, slack bounds, and comparison budgets used by the displayed windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the same finite windows and dyadic budgets as regular rational data before any real-facing seal can be consumed.*
5. *H records componentwise* hsame *transport for the unary-depth, stream-window, dyadic, regular-rational, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a finite prefix request through k, W, D, R, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{FinitePrefixStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixStream}^{\uparrow}}$ *row naming exactly this finite prefix certificate.*

The classifier compares accepted packets only through k, W, D, R, H, C, P, N. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.TasteGate.FinitePrefixStreamTasteGate single carrier alignment`

Theorem 7253.2 (Finite prefix stream NameCert obligations)**.** *For every accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier $F = (k, W, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixStream}^{\uparrow}}$ *surface consists exactly of carrier admission for the unary depth, finite StreamName windows, dyadic tolerance rows, RegSeqRat handoff, componentwise transport, displayed replay, provenance, and local naming. The obligation surface may cite the StreamName source of chapter 126, the RegSeqRat source of chapter 301, and the dyadic arithmetic core of chapter 291; it cannot add quotient stream equality, a selected limit, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge certificate.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier obligation surface`

Proof. Project the carrier of Definition 7253.1. The visible analytic rows are exactly the unary depth k, finite stream window W, dyadic tolerance ledger D, and regular rational handoff R.

Induct on the finite successor spine displayed by k. At depth 0, the certificate reads only the initial window cell, its dyadic budget, and the matching regular rational observation. In the successor step, append one displayed window row and one matching dyadic budget, then transport the already-read prefix through H and replay it through C.

The rows P and N source and name only that finite packet. Since the excluded principles are not fields of the carrier and are not produced by the induction, no local NameCert obligation can export quotient stream equality, a selected limit, countable choice, host real equality, ambient real completeness, or bridge checking. □

Theorem 7253.3 (Finite prefix stream window readback)**.** *Every RegSeqRat-facing consumer of an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier reads the displayed prefix by the finite route*

$$k \longmapsto W \longmapsto D \longmapsto R \longmapsto H, C, P, N.$$

The exported read is a prefix-bounded regular-rational handoff, not an equality class of streams, not a chosen infinite tail, not a countable-choice schedule, not host equality of real names, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier window readback`

Proof. Fix a consumer of the accepted carrier. The first row available to the consumer is the unary depth k, so the consumer receives a finite bound before any stream observation is inspected.

The window read is then bounded by W. For each displayed index at or below k, the consumer reads the corresponding dyadic budget from D and the matching regular rational row from R.

The structural rows H, C, P, N preserve this same finite route under transport, replay, provenance, and naming. They do not add an infinite tail, an equality class of streams, a choice schedule, host equality of real names, or completeness data. □

Theorem 7253.4 (Finite prefix stream observation induction)**.** *Let $F = (k, W, D, R, H, C, P, N)$ be an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier. For every displayed unary index $i \leq k$, the observation at i is obtained by induction on the successor spine of k: the route reads exactly the corresponding*

StreamName$^{\uparrow}$ *window row in* W*, the matching* DyadicRatCore$^{\uparrow}$ *tolerance row in* D*, the* RegSeqRat$^{\uparrow}$ *handoff row in* R*, and the structural* H, C, P, N *rows. The induction exports a finite observation proof only; it does not export quotient stream equality, a selected tail, countable choice, host real equality, ambient* Real$^{\uparrow}$ *completeness, or a bridge certificate.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier observation induction`

Proof. Use Definition 7253.1 to project the unary depth row k and the finite rows W, D, R, H, C, P, N.

The base depth reads the initial window entry from W, the matching dyadic tolerance entry from D, and the corresponding regular-rational handoff from R. The structural rows then record the displayed transport, replay, provenance, and local naming for that initial observation.

For the successor step, extend the already-read finite route by one displayed observation. Apply Definition 7253.3 to force the new observation to pass through the same W, D, R rows before it reaches the structural rows H, C, P, N.

Thus every index $i \leq k$ is read by finite induction on the displayed spine. Since no excluded object is a carrier coordinate or a product of the induction, the observation proof remains a finite StreamName-to-RegSeqRat handoff. □

Definition 7253.5 (Finite prefix stream observation cut)**.** *Let* $F = (k, W, D, R, H, C, P, N)$ *be an accepted* FinitePrefixStream$^{\uparrow}$ *carrier and let* $j \leq k$ *be a displayed unary cut. The observation cut is the restricted packet*

$$F_{\leq j} = (j, W_{\leq j}, D_{\leq j}, R_{\leq j}, H, C, P, N)$$

where $W_{\leq j}$ *is the first* j *StreamName window row segment,* $D_{\leq j}$ *is the first* j *DyadicRatCore tolerance row segment, and* $R_{\leq j}$ *is the first* j *RegSeqRat handoff row segment read from the same finite prefix carrier. The cut preserves only the displayed unary prefix, not any omitted tail coordinate.*

Lean4 (definition) `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCertificateObservationCut`

Theorem 7253.6 (Finite prefix stream observation cut stability)**.** *Let* $F = (k, W, D, R, H, C, P, N)$ *be an accepted* FinitePrefixStream$^{\uparrow}$ *carrier and let* $j \leq k$ *be a displayed unary cut. Then the cut packet* $F_{\leq j}$ *of Definition 7253.5 is again an accepted* FinitePrefixStream$^{\uparrow}$ *carrier, and every RegSeqRat or Real-facing consumer of* $F_{\leq j}$ *factors through the same route*

$$j \longmapsto W_{\leq j} \longmapsto D_{\leq j} \longmapsto R_{\leq j} \longmapsto H, C, P, N.$$

The cut exposes only the displayed finite prefix route; it does not export a selected tail, quotient stream equality, countable choice, host real equality, ambient Real$^{\uparrow}$ *completeness, or bridge checking.*

Proof. Project the accepted carrier of Definition 7253.1. The unary cut $j \leq k$ simply shortens the displayed prefix window, dyadic tolerance, and RegSeqRat handoff rows to the visible initial segment.

Now apply Definition 7253.4. The proof already reads each displayed index by induction on the successor spine of k, so the same finite induction reruns on the truncated prefix without exposing any omitted tail coordinate.

Finally replay Definition 7253.3. The RegSeqRat and Real-facing consumers still see the same finite route through $W_{\leq j}, D_{\leq j}, R_{\leq j}$ before the structural rows H, C, P, N can be read. Since the omitted tail is not a carrier coordinate, the cut remains a finite observation prefix and nothing more. □

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier observation cut stability`

Theorem 7253.7 (Finite prefix stream cofinal reindex stability)**.** *Let $F = (k, W, D, R, H, C, P, N)$ be an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier. Suppose a displayed monotone cofinal reindex route sends a target prefix request m to a source cut $j \leq k$ and exposes that route through the cofinal and stream-reindexing rows of chapter 7209 and chapter 7081. Then every visible prefix observation in the reindexed request still factors through the same* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff after the unary depth k is replaced by the displayed reindexed prefix m. The exported route is finite reindex data followed by finite prefix data; it does not use quotient stream equality, selected tails, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCofinalReindexStability`

Proof. Project the carrier by Definition 7253.1. The source-side prefix observations available to the reindex route are exactly the displayed rows k, W, D, R, H, C, P, N.

Apply Definition 7253.4. Each source observation at or below k is read by finite induction through the matching $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff row.

The displayed cofinal route supplies the target prefix m and the source cut $j \leq k$. By Definition 7253.6, the source-side read may be restricted to $F_{\leq j}$ without exposing any omitted tail coordinate.

Finally use the sibling reindexing surfaces of chapter 7209 and chapter 7081. They place the monotone cofinal route and the stream schedule before any regular-rational or real-facing read, so replacing the visible unary depth by m changes only the displayed prefix request. The downstream StreamName window, DyadicRatCore tolerance row, and RegSeqRat handoff remain the same finite rows. □

Theorem 7253.8 (Finite prefix stream regular Cauchy reindex handoff)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads a reindexed* $\mathsf{FinitePrefixStream}^{\uparrow}$ *prefix must pass first through the displayed cofinal reindex route, then through the finite prefix rows*

$$m \longmapsto W_{\leq j} \longmapsto D_{\leq j} \longmapsto R_{\leq j} \longmapsto H, C, P, N,$$

and only after that route may it consume the terminal $\mathsf{Real}^{\uparrow}$ *seal or Bishop regular Cauchy completion seal. Thus a regular Cauchy or Real handoff from a reindexed prefix is a composed finite route, not quotient stream equality, a selected subsequence limit, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Begin with the cofinal reindex route and apply Definition 7253.7. This forces the reindexed prefix request to expose the finite StreamName window, dyadic tolerance, and RegSeqRat handoff rows before any downstream consumer reads the packet.

Next apply Definition 7253.11. The Real-facing or completion-facing budget can inspect only the prefix rows $W_{\leq j}, D_{\leq j}, R_{\leq j}$ and the structural rows H, C, P, N before it reaches a terminal seal.

Finally use Definition 7253.12. Any completion-facing section is pulled back to that same finite route, so the Bishop regular Cauchy or Real consumer receives a composed reindex-and-prefix handoff rather than a host-level equality or selected limiting object. □

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier regular cauchy reindex handoff`

Theorem 7253.9 (Finite prefix stream certificate prefix concatenation)**.** *Let*

$$F_0 = (k_0, W_0, D_0, R_0, H_0, C_0, P_0, N_0)$$

and

$$F_1 = (k_1, W_1, D_1, R_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{FinitePrefixStream}^{\uparrow}$ *carriers whose unary-depth rows are adjacent. Suppose the shared boundary observation is classified by the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, and suppose both prefixes use the same displayed dyadic tolerance ledger shape and compatible structural rows. Then the two prefixes concatenate to a larger finite prefix certificate whose visible rows are exactly the merged* W, D, R, H, C, P, N *rows. The concatenated certificate exports no infinite tail, selected limit, quotient stream equality, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCertificatePrefixConcatenation`

Proof. Project both carriers using Definition 7253.1. The only analytic rows available for concatenation are the unary depths, StreamName windows, dyadic ledgers, and RegSeqRat handoff rows.

Apply observation induction on the left prefix. This reads every displayed observation up to the boundary through W_0, D_0, R_0 and the structural rows H_0, C_0, P_0, N_0.

Use one successor step for the right boundary. The shared boundary hypothesis identifies the boundary observation through the same $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, so the first row of the right prefix attaches to the already-read left prefix without changing the classifier route.

Merge only the displayed window, dyadic, regular-rational, transport, replay, provenance, and local naming rows. Because both carriers expose the same ledger shape and compatible structural rows, the merged packet is again a finite prefix certificate.

No carrier coordinate in either prefix stores an infinite tail, selected limit, quotient stream equality, host real equality, choice schedule, or ambient completeness. The concatenation therefore preserves the finite StreamName-to-RegSeqRat boundary and exports none of the excluded objects. □

Theorem 7253.10 (Finite prefix stream nonescape)**.** *Every Real or Completion-facing use of an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier remains downstream of the displayed prefix window and the RegSeqRat handoff. Thus the packet may be used by the real-name and Bishop regular Cauchy completion routes only as finite prefix evidence; it does not export quotient stream equality, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier nonescape`

Proof. Start with a Real or Completion-facing use of F. By Definition 7253.1, the only analytic route into such a use is through k, W, D, R.

Apply Definition 7253.3. The consumer obtains only prefix-bounded regular rational readback, and the structural rows keep the same prefix route visible.

Consequently any downstream $\mathsf{Real}^{\uparrow}$ or Bishop regular Cauchy completion argument can cite this packet only as finite observation evidence. The missing objects are absent from the route, so no quotient stream equality, selected limit, countable-choice schedule, host real equality, ambient real completeness, or checked bridge can escape from the certificate. □

Theorem 7253.11 (Finite prefix stream real-completion budget route)**.** *Any completion-facing use of an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier must expose the unary depth row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and structural* H, C, P, N *rows before reading a* $\mathsf{Real}^{\uparrow}$ *seal or a Bishop regular Cauchy completion seal. The route is the finite budget*

$$k \longmapsto W \longmapsto D \longmapsto R \longmapsto H, C, P, N,$$

not quotient stream equality, selected tails, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier using Definition 7253.1. The only analytic rows available before a completion-facing handoff are the unary depth k, the finite window row W, the dyadic tolerance row D, and the regular rational handoff R.

Apply Definition 7253.3. It forces the consumer to read the prefix route through k, W, D, R before the structural rows can replay, transport, source, and name the packet.

Now use Definition 7253.10. Any $\mathsf{Real}^{\uparrow}$ or Bishop regular Cauchy completion read remains downstream of that prefix route, so the completion budget is exactly the displayed finite window and tolerance budget.

Since no carrier coordinate stores quotient stream equality, a selected tail, countable choice, host real equality, or ambient Real completeness, none of those principles can be read by the completion-facing consumer. □

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCarrier real completion budget route`

Theorem 7253.12 (Finite prefix stream completion-section pullback)**.** *Let* $F = (k, W, D, R, H, C, P, N)$ *be an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier. Any completion-facing section that consumes* F *factors through the same finite route*

$$k \longmapsto W \longmapsto D \longmapsto R \longmapsto H, C, P, N.$$

The four visible analytic faces of this pullback are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*: the stream window is read from* W*, its dyadic tolerance budget is read from* D*, the regular-rational handoff is read from* R*, and every real-facing seal remains downstream of those finite rows. The pullback exports no quotient stream equality, selected tail, countable-choice schedule, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Project the accepted carrier by Definition 7253.1. The completion-facing section can inspect only the displayed unary depth k, window row W, dyadic row D, regular-rational row R, and structural rows H, C, P, N.

Apply Definition 7253.3 to force every stream observation used by the section through k, W, D, R. Then apply Definition 7253.4 to see that each displayed index is obtained by finite induction on the successor spine of k, with no appeal to an infinite representative or selected tail.

Finally use Definition 7253.11. The real and completion-facing read is therefore pulled back to the same finite budget route before any $\mathsf{Real}^{\uparrow}$ seal is consumed. Since quotient stream equality, countable choice, host real equality, ambient Real completeness, and bridge checking are absent from the carrier coordinates and from the cited finite route, none of them is exported by the section. □

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamCompletionSectionPullback`

Theorem 7253.13 (Finite prefix stream L10 exit observation route)**.** *Every accepted Real-and-Completion observation route served by a* $\mathsf{FinitePrefixStream}^{\uparrow}$ *packet factors through the displayed finite prefix rows*

$$k, W, D, R, H, C, P, N.$$

The four analytic faces of the route are $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*: the StreamName prefix is read from* W*, the dyadic tolerance budget is read from* D*, the regular rational handoff is read from* R*, and the Real-facing seal remains downstream of those finite rows.*

The exit observation route exports no quotient stream equality, selected tail, countable-choice schedule, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Apply finite prefix window readback, Definition 7253.3. Any RegSeqRat-facing consumer first reads the displayed route through k, W, D, R and only then uses the structural rows H, C, P, N.

Use observation induction, Definition 7253.4, to identify each displayed prefix index with the corresponding StreamName row, dyadic tolerance row, and regular rational handoff row. This supplies the finite observation route for the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ faces.

Next apply the real-completion budget route, Definition 7253.11, and the completion-section pullback, Definition 7253.12. Together they force every Real or Completion-facing use to pass through the same finite prefix budget before a $\mathsf{Real}^{\uparrow}$ seal can be consumed.

All cited routes use only the displayed rows k, W, D, R, H, C, P, N. Since none of those rows stores quotient stream equality, a selected tail, countable choice, host real equality, ambient Real completeness, or bridge checking, the L10 exit route exports none of those objects. □

Lean4 ✓ `BEDC.Derived.FinitePrefixStreamUp.FinitePrefixStreamL10ExitObservationRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixStream}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7253.1, Definition 7253.2, Definition 7253.3, and Definition 7253.10 seed the finite observation-prefix certificate between StreamName and RegSeqRat real-name reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unary-depth prefix induction, finite StreamName window readback, DyadicRatCore tolerance exactness, RegSeqRat handoff, transport, replay, provenance, and local naming for BEDC.Derived.FinitePrefixStreamUp.

Constructive story: A $\mathsf{FinitePrefixStream}^{\uparrow}$ packet is generated from a unary depth, finite StreamName windows, DyadicRatCore tolerance rows, a RegSeqRat handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7254

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$ packet records the finite route from an explicit-modulus Cauchy rational name to a sealed $\mathsf{Real}^{\uparrow}$ boundary. It sits over the dyadic tolerance core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the seal route of chapter 386. The packet names an effective completion surface for displayed Cauchy data; it does not quotient Cauchy classes, select host limits, use countable choice, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7254.1 (Effective Cauchy real completion carrier)**.** *An* $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the explicit modulus row, assigning to each requested dyadic tolerance a finite stream window large enough for the Cauchy comparison.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the entries selected by M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the selected rational entries inside each modulus window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from M, W, D after the Cauchy window comparison has been displayed.*
5. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback row R is visible.*
6. *H records componentwise* hsame *transport for M, W, D, R, S, C, P, N.*
7. *C records displayed* Cont *replay from a Cauchy request through the modulus, window, dyadic, readback, seal, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, W, D, R, S, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative, host real equality, countable choice, an independent limit object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7254.2 (Effective Cauchy real completion obligation surface)**.** *For every accepted* EffectiveCauchyRealCo *carrier*

$$E = (M, W, D, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier admission for the displayed* BHist *packet E;*
2. *modulus-window exactness, requiring M to choose the same finite windows read by W and checked by D;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *readback discipline, requiring R to be produced only from the displayed M, W, D route;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *row S to be read only after R is visible;*
5. *ledger exhaustion through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Project the carrier of Definition 7254.1. The analytic rows are exactly the explicit modulus M, finite stream window W, dyadic tolerance ledger D, regular rational readback R, and terminal seal row S.

The modulus-window row is exact because W exposes only the entries selected by M, while D checks those entries against the requested dyadic tolerance. Thus no Cauchy comparison can be imported from an unlisted stream coordinate.

Next read the $\mathsf{RegSeqRat}^{\uparrow}$ handoff. The row R is downstream of M, W, D, so the regular rational readback discipline is the same finite route used in chapter 301 and cannot be replaced by host sequence equality.

The seal row is then consumed through chapter 386 and chapter 38. Since S appears only after R, the $\mathsf{Real}^{\uparrow}$ endpoint is a routed seal surface rather than a selected limit or a completeness theorem.

Finally H, C, P, N exhaust the ledger. They transport, replay, package, and name the same displayed rows, and the carrier has no coordinate for quotient streams, selected representatives, countable choice, host equality, independent limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7254.3 (Effective Cauchy real completion seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$M, W, D, R, S, H, C, P, N.$$

In particular, the read first displays the explicit modulus, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal before the* $\mathsf{Real}^{\uparrow}$ *boundary is public. It exports no quotient completion, selected host limit, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 7254.2. It places the modulus, window, dyadic, readback, and seal rows in the required order for every accepted packet.

Use the sibling routes named in the chapter introduction. $\mathsf{StreamName}^{\uparrow}$ supplies only finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies only displayed tolerance checks, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ supplies the routed seal consumed by $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N preserve that same route for downstream consumers. Because no row stores quotient classes, selected limits, countable choice, or ambient completeness, the exported completion read is exactly the displayed finite seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}$ packet is generated from an explicit modulus row, finite StreamName windows, DyadicRatCore tolerance ledgers, a RegSeqRat readback row, a CauchyLimitSeal and RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7254.1, Definition 7254.2, and Definition 7254.3 seed the explicit-modulus Cauchy-to-Real completion surface over displayed DyadicRatCore, StreamName, RegSeqRat, CauchyLimitSeal, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyRealCompletion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected host limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-window exactness, RegSeqRat readback discipline, RealUp seal nonescape, ledger exhaustion, provenance, and local naming for BEDC.Derived.EffectiveCauchyRealCompletionUp.

Chapter 7255

A Concrete Naming Certificate for $\mathsf{CompactModulusSelection}^{\uparrow}$

Following Bishop and Bridges on constructive compactness and uniform continuity, the $\mathsf{CompactModulusSelection}^{\uparrow}$ packet records the finite-net route by which a compact metric source and a pointwise continuity ledger yield a displayed modulus-selection row. It sits over the compact-metric source of chapter 129, the uniform-modulus boundary of chapter 369, the finite stream window surface of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite compact-net and modulus-selection data only; it does not assert open-cover compactness, selected subsequence choice, quotient $\mathsf{Real}^{\uparrow}$ equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7255.1 (Compact modulus selection carrier)**.** *A* $\mathsf{CompactModulusSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, N, L, U, S, R, E, H, C, P, A)$$

with the following displayed rows:

1. *K is the compact-metric source row read from chapter 129.*
2. *N is the finite-net row, recording the displayed compact net, tolerance budget, and cover membership checks used by the selector.*
3. *L is the local pointwise continuity ledger, naming the inspected centers and local continuity budgets before any uniform row is exposed.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, available only after K, N, L have been displayed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to replay center and tolerance requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the selected rational tolerance.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for Real-facing consumers.*
8. *H records componentwise* hsame *transport for compact source, finite net, continuity ledger, uniform handoff, stream, regular-rational, Real, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from compact-net requests through local continuity budgets into the uniform handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactModulusSelection}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *A is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, N, L, U, S, R, E, H, C, P, A$. It has no coordinate for an open-cover compactness theorem, selected subsequence functional, quotient Real representative, or ambient completeness principle.

Theorem 7255.2 (Compact modulus selection carrier boundary)**.** *Every accepted* $\mathsf{CompactModulusSelection}^{\uparrow}$ *carrier is read through the finite rows*

$$M = (K, N, L, U, S, R, E, H, C, P, A)$$

over BHist*, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelecti}}$ *surface. Any consumer of the carrier sees compact-metric data, a finite-net selector, a pointwise continuity ledger, a* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, stream windows, regular-rational readback, Real sealing, and the structural rows, and it sees no open-cover compactness, selected subsequence choice, quotient* $\mathsf{Real}^{\uparrow}$ *equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7255.1. The displayed tuple lists exactly the compact source, finite-net, pointwise continuity, uniform-modulus, stream, regular-rational, Real, and structural rows.

The transport and replay rows preserve those coordinates componentwise. Thus a read of the carrier cannot introduce a host compactness proof, a selected subsequence object, a quotient representative, or an ambient completeness datum not already present in M. □

Theorem 7255.3 (Compact modulus selection finite-net selector)**.** *Let $M = (K, N, L, U, S, R, E, H, C, P, A)$ be an accepted* $\mathsf{CompactModulusSelection}^{\uparrow}$ *packet. The finite-net selector row N must be consumed before the* $\mathsf{UniformModulus}^{\uparrow}$ *row U is public: every modulus request first reads the compact-metric source K, the finite net N, the pointwise continuity ledger L, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The selector does not choose a subsequence, infer open-cover compactness, identify quotient Real names, or use ambient Real completeness.*

Proof. Read the source side first. The compact-metric row K supplies the displayed compact source, while N supplies the finite net and tolerance ledger on that same source.

Next read the local continuity ledger L. The uniform handoff U is available only after the finite centers and local budgets have been replayed through C and transported through H.

Finally pass to the stream, regular-rational, and Real rows. They record finite request replay and readback only, so the selector remains a finite-net route rather than any of the refused choice, open-cover, quotient, or completeness principles. □

Theorem 7255.4 (Compact modulus selection NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusSelecti}$ *packet $M = (K, N, L, U, S, R, E, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelection}^{\uparrow}}$ *surface consists of the following finite obligation rows:*

1. *carrier admission for compact-metric source, finite-net selector, pointwise continuity ledger, uniform-modulus handoff, stream windows, regular-rational readback, Real seal, transport, replay, provenance, and local naming;*
2. *finite-net coverage, requiring N to be read from the compact source row K before any uniform handoff is exported;*
3. *local-budget compatibility, requiring L to assign pointwise continuity budgets to the centers named by N;*

4. *modulus handoff exactness, requiring U to be computed only from K, N, L and the structural rows H, C, P, A;*
5. *terminal non-escape, requiring S, R, E to replay and seal the selected tolerance without adding selected subsequences, quotient Real equality, open-cover compactness, or ambient Real completeness.*

Thus the certificate packages compact finite-net modulus selection as displayed BEDC NameCert data.

Proof. Project the accepted packet through the carrier boundary. The first obligation records exactly the displayed rows in Definition 7255.1.

For coverage and compatibility, read K, N, L in order. The finite net is anchored in the compact source, and the local continuity budgets are attached to the centers displayed by that net.

For handoff and non-escape, read U, S, R, E, H, C, P, A. The uniform row is downstream of the finite net and local budgets, while stream replay, regular-rational readback, and Real sealing preserve the same route. Since no further coordinate remains, the certificate refuses the open-cover, subsequence, quotient, and completeness endpoints named in the statement. □

Falsifiable prediction: A $\mathsf{CompactModulusSelection}^{\uparrow}$ consumer that exports a $\mathsf{UniformModulus}^{\uparrow}$ handoff without displaying the compact-metric row K, finite-net row N, pointwise continuity ledger L, and structural BHist, hsame, Cont, Pkg, and NameCert rows refutes the carrier of Definition 7255.1.

Independence witness: compactmetric, uniformmodulus, streamname, regseqrat, real: CompactModulusSelectio is not bijective to any listed sibling because it jointly carries compact-metric source data, finite-net selector rows, pointwise continuity budgets, uniform-modulus handoff, stream replay, regular-rational readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusSelection}^{\uparrow}$ packet is generated from compact-metric source data, finite-net selector rows, pointwise continuity budgets, a UniformModulus handoff, stream replay, regular-rational readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7255.1, Definition 7255.2, Definition 7255.3, and Definition 7255.4 seed the finite compact-net modulus-selection carrier and its NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly refuses open-cover compactness, selected subsequence choice, quotient $\mathsf{Real}^{\uparrow}$ equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, local-budget compatibility, modulus handoff exactness, terminal non-escape, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.CompactModulusSelectionUp.

Chapter 7256

A Concrete Naming Certificate for $\mathsf{LocatedDedekindCut}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers and Troelstra–van Dalen on located cuts, the $\mathsf{LocatedDedekindCut}^{\uparrow}$ packet records the traditional two-sided located Dedekind cut as finite NameCert data before any real-seal consumer reads it. It sits between the located supremum surface of chapter 1427, the located real boundary of **??**, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a narrow L10 carrier for Real and Completion consumers: it exposes cut location and readback rows without identifying cuts by quotient equality or asserting completeness of the ambient real surface.

Definition 7256.1 (Located Dedekind cut carrier)**.** *A* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, G, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-cut row, carrying inhabited lower observations, downward closure witnesses, rounded lower-side requests, and the finite rational or dyadic cells actually inspected.*
2. *U is the upper-cut row, carrying inhabited upper observations, upward closure witnesses, rounded upper-side requests, and the matching finite comparison cells.*
3. *G is the located comparison row. For each displayed interval request $a < b$, it records either a lower-side observation below b or an upper-side observation above a inside the same finite packet.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by Real and Completion consumers that inspect the cut at a requested precision.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed lower, upper, locatedness, and window rows.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint comparison, gap narrowing, radius accounting, and lower-bound folds used by the readback route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, G, W, R, Q have been displayed.*
8. *H records componentwise* hsame *transport for lower-cut rows, upper-cut rows, located comparisons, windows, readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a cut request through the same ordered tuple $L, U, G, W, R, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedDedekindCut}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, G, W, R, Q, E, H, C, P, N$. It has no coordinate for quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7256.2 (Located Dedekind cut NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *carrier $D = (L, U, G, W, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower-cut row, upper-cut row, located comparison row, StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance ledger, terminal Real seal, componentwise transport, displayed replay, package provenance, and local naming. Every Real, Completion, or* $\mathsf{LocatedSupremum}^{\uparrow}$ *consumer must read those rows before using the packet as located cut evidence.*

Proof. Project the accepted carrier. The lower and upper cut rows are the only cut data present, and the located comparison row G is the only answer to interval requests $a < b$.

The observation and readback rows are downstream of those cut rows. A consumer first opens finite windows through W, then reads regular rational data through R, and then checks dyadic tolerance and lower-bound accounting through Q.

The terminal Real seal E is reached only after this displayed route. The structural rows H, C, P, N transport, replay, source, and name the same finite packet, so they do not add quotient cuts, selected suprema, trichotomy, host equality, choice, or ambient completeness. □

Theorem 7256.3 (Located Dedekind cut Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *packet factors through*

$$L, U, G \longmapsto W \longmapsto R \longmapsto Q \longmapsto E.$$

The route exposes lower-cut observations, upper-cut observations, located comparison, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, and the terminal Real seal before any consumer may read real-name evidence.

Proof. Read the cut rows L, U, G from Definition 7256.1. They supply the lower side, upper side, and located comparison surface available to a Real-facing request.

Next pass through the finite observation rows. The window row W lists exactly the finite observations opened by the request, while R repacks those observations as regular rational readback and Q records the dyadic tolerance ledger used to compare them.

The seal row E is terminal for this route. Componentwise transport and continuation replay preserve only the displayed rows, so the seal cannot export quotient cut equality, selected suprema, host Real equality, countable choice, trichotomy, or ambient completeness. □

Theorem 7256.4 (Located Dedekind cut supremum handoff)**.** *Every* $\mathsf{LocatedSupremum}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *packet receives only the displayed lower row L, upper row U, located comparison row G, finite windows W, readback row R, tolerance ledger Q, Real seal E, and structural rows H, C, P, N. The handoff may use the packet as finite located-cut evidence, but it does not select a supremum, prove trichotomy, identify host Real equality, invoke countable choice, or assert ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7256.2. They force every consumer, including the located-supremum consumer, to read the lower, upper, locatedness, window, readback, tolerance, seal, and structural rows.

The handoff is therefore finite. The rows L, U, G provide the cut evidence, W, R, Q provide precision-facing readback, and E gives the terminal real-seal boundary.

Since no row contains a selected least upper bound, a quotient cut representative, a trichotomy branch, a choice schedule, or a host completeness proof, the $\mathsf{LocatedSupremum}^{\uparrow}$ consumer receives only the displayed finite located-cut packet. □

Falsifiable prediction: A $\mathsf{LocatedDedekindCut}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{LocatedSupremum}^{\uparrow}$ handoff while bypassing the lower-cut row, upper-cut row, located comparison row, finite windows, regular readback, or dyadic tolerance ledger refutes the carrier of Definition 7256.1.

Independence witness: locatedsupremum, locatedreal, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedDedekindCut}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower and upper cut rows, located comparison, finite windows, regular rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDedekindCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDedekindCut}^{\uparrow}$ packet is generated from lower-cut rows, upper-cut rows, located comparison rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDedekindCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7256.1, Definition 7256.2, and Definition 7256.3 seed the narrow L10 located-Dedekind-cut carrier between located cut evidence and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDedekindCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly refuses quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, ambient Real completeness, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-cut and upper-cut row exposure, located comparison exactness, StreamName window replay, RegSeqRat readback, DyadicRatCore tolerance accounting, Real-seal ordering, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.LocatedDedekindCutUp.

Chapter 7257

Shape Theory NameCert Construction

Following Borsuk and Mardesic–Segal, the $\mathsf{ShapeTheory}^{\uparrow}$ packet encodes compact-metric shape through finite pro-homotopy expansion rows. It sits beside the compact metric carrier of chapter 129, the hyperspace packet of chapter 1660, the absolute-neighborhood-retract packet of chapter 6562, and the homotopy carrier of chapter 156. The chapter records the finite expansion, comparison, and nonescape rows that a shape consumer may inspect, not a host pro-category, quotient compactum, selected inverse system, or ambient homotopy classifier.

Definition 7257.1 (Shape theory carrier)**.** *A* $\mathsf{ShapeTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, H, A, T, E, M, R, U, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *surface.*
2. *H is the hyperspace row recording finite compact-subset windows used by expansion stages.*
3. *A is the absolute-neighborhood-retract row naming the finite ANR approximants admitted by the packet.*
4. *T is the homotopy row recording finite homotopy comparisons between adjacent approximants.*
5. *E is the expansion-index row, listing the finite prefix of pro-homotopy stages visible to the consumer.*
6. *M is the bonding-map row, recording the displayed maps and homotopy commutation checks inside that prefix.*
7. *R is the refinement row, pairing later expansion prefixes with earlier displayed stages.*
8. *U records componentwise* hsame *transport for compact-metric, hyperspace, ANR, homotopy, expansion, bonding-map, refinement, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a compact-metric read through hyperspace, ANR, and homotopy rows before a shape comparison is exposed.*
10. *P is the* Pkg *provenance over* $\mathsf{ShapeTheory}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ShapeTheory}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an unbounded pro-object, host compactum quotient, arbitrary shape category, selected inverse-system representative, or ambient homotopy equivalence outside the displayed prefix.

Theorem 7257.2 (Shape theory NameCert obligations)**.** *For every accepted* $\mathsf{ShapeTheory}^{\uparrow}$ *carrier* $S = (K, H, A, T, E, M, R, U, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ShapeTheory}^{\uparrow}}$ *surface consists exactly of compact-metric admission, hyperspace-window admission, ANR approximant admission, finite homotopy comparison, expansion-prefix exposure, bonding-map exactness, refinement-row visibility, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 7257.1. The compact metric row K and hyperspace row H name the compact source and finite compact-subset windows before any shape read is available.

The ANR row A and homotopy row T supply only displayed approximants and finite comparison data. The expansion row E, bonding row M, and refinement row R bind those comparisons into a finite pro-homotopy prefix.

Transport, replay, provenance, and naming through U, C, P, N preserve that same prefix. Since no carrier coordinate stores a host pro-category, quotient compactum, selected inverse system, or ambient homotopy classifier, the listed rows exhaust the local NameCert obligations. □

Theorem 7257.3 (Shape theory pro-homotopy handoff)**.** *Every shape comparison read from an accepted* $\mathsf{ShapeTheory}^{\uparrow}$ *packet factors through the compact-metric source, hyperspace windows, ANR approximant rows, homotopy comparison rows, expansion prefix, bonding-map row, refinement row, transport, replay, provenance, and local naming. The handoff is finite pro-homotopy NameCert data, not an ambient pro-category theorem or an unrestricted shape-equivalence principle.*

Proof. Read the source through K and H, then expose ANR approximants through A. Homotopy comparisons are visible only through T.

The expansion prefix E determines which stages are in scope. The bonding and refinement rows M, R then route every comparison through a displayed finite row before it reaches a consumer.

The structural rows U, C, P, N transport and replay the same data. Thus the exported handoff remains the finite pro-homotopy prefix named by the packet. □

Theorem 7257.4 (Shape theory nonescape)**.** *An accepted* $\mathsf{ShapeTheory}^{\uparrow}$ *carrier exports no host compactum quotient, selected inverse system, arbitrary pro-object equality, open-cover compactness theorem, or ambient homotopy classifier. Any consumer that cites a shape comparison must exhibit the rows* $K, H, A, T, E, M, R, U, C, P, N$ *of the same packet.*

Proof. The only comparison rows in Definition 7257.1 are T, E, M, R together with the compact, hyperspace, and ANR sources K, H, A.

No row stores a host quotient, selected inverse system, arbitrary pro-object equality, open-cover compactness proof, or ambient classifier. Therefore any such read is outside the carrier unless it is first reduced to the displayed rows and replayed through U, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{ShapeTheory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ShapeTheory}^{\uparrow}$ packet is generated from compact metric, hyperspace, absolute-neighborhood-retract, homotopy, expansion-prefix, bonding-map, refinement, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ShapeTheory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7257.1, Definition 7257.2, Definition 7257.3, and Definition 7257.4 seed the finite pro-homotopy expansion surface for compact-metric shape data.

Formal verification: FormalStatus(NameCert$_{\mathsf{ShapeTheory}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host pro-categories, quotient compacta, selected inverse systems, arbitrary shape equivalence, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite pro-homotopy handoff and nonescape obligations.

Chapter 7258

A Concrete Naming Certificate for MetaCICResidualJoinGrid$^{\uparrow}$

The MetaCICResidualJoinGrid$^{\uparrow}$ packet realizes the confluence audit route of chapter 9378 as a finite residual-join-grid NameCert carrier. It sits beside the parallel-diamond frontier of chapter 4238, the closed parallel-diamond premise ledger of chapter 2325, and the residual handoff of chapter 5126. The packet records residual vertices, candidate joins, premise rows, and obstruction rows before any confluence handoff can be consumed; it does not assert global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Definition 7258.1 (MetaCIC residual join grid carrier)**.** *A* MetaCICResidualJoinGrid$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, L, R, V, J, D, O, H, C, P, N)$$

with the following displayed rows:

1. *F is the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *row retaining the conditional residual-substitution frontier.*
2. *L is the* ClosedParallelDiamondPremiseLedger$^{\uparrow}$ *row recording the closed premise ledger a confluence route must cite.*
3. *R is the* ParallelDiamondResidual$^{\uparrow}$ *row exposing the residual local-join packet.*
4. *V is the finite residual-vertex row, listing the bounded residual endpoints inspected by the grid.*
5. *J is the candidate-join row, pairing residual vertices with displayed local joins.*
6. *D is the diamond-cell row, recording which adjacent residual vertices share a displayed join cell.*
7. *O is the obstruction row retaining missing residual-substitution, subject-reduction, evaluator-equality, quotient-equality, and kernel-consistency endpoints.*
8. *H records componentwise* hsame *transport for frontier, ledger, residual, vertex, join, diamond-cell, obstruction, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a frontier read through premise ledger and residual rows before any join cell is consumed.*
10. *P is the* Pkg *provenance over* MetaCICResidualJoinGrid$^{\uparrow}$*,* MetaCICParallelDiamondFrontier$^{\uparrow}$*,* ClosedParallelDiamon ParallelDiamondResidual$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for premise-free confluence, global normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Theorem 7258.2 (MetaCIC residual join grid NameCert obligations)**.** *For every accepted* MetaCICResidualJoinGric *carrier $G = (F, L, R, V, J, D, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}$ *surface consists exactly of frontier retention, premise-ledger exposure, residual-handoff exposure, finite residual-vertex exposure, candidate-join exposure, diamond-cell exactness, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 7258.1. The rows F, L, R expose the conditional frontier, closed premise ledger, and residual handoff before any grid cell is visible.

The vertex row V, join row J, and diamond-cell row D record only finite residual endpoints and displayed local joins. The obstruction row O remains part of the same packet, so the grid cannot be read as premise-free confluence.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite row surface. Since no coordinate stores the refused global endpoints, the listed rows exhaust the local NameCert obligations. □

Theorem 7258.3 (MetaCIC residual join grid confluence handoff)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ *packet receives a finite residual join grid only after the parallel-diamond frontier, closed premise ledger, residual handoff, residual vertices, candidate joins, diamond cells, obstruction row, transport, replay, provenance, and local naming have been displayed. The handoff can feed the audit route of chapter 9378, but it does not discharge residual-substitution compatibility, subject reduction, global confluence, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Begin with F, L, R. These rows retain the conditional frontier, premise ledger, and residual packet named by the sibling chapters.

The grid rows V, J, D then expose only bounded vertices, local joins, and finite diamond cells. The obstruction row O remains visible at the same handoff.

The structural rows H, C, P, N replay the route without adding any refused endpoint. Thus the consumer receives a finite residual join grid, not a completed confluence theorem. □

Theorem 7258.4 (MetaCIC residual join grid nonescape)**.** *An accepted* $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ *carrier exports no premise-free parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, global normalization theorem, or kernel-consistency certificate. Any accepted residual-grid read must remain inside $F, L, R, V, J, D, O, H, C, P, N$.*

Proof. The grid carrier contains frontier, premise-ledger, residual, vertex, join, diamond-cell, and obstruction rows only. Each row is finite and replayed through the same structural route.

No coordinate removes O or supplies a global endpoint. Therefore a consumer that reads one of the refused endpoints has left the displayed $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}$ surface. □

Falsifiable prediction: Every closed $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ residual join grid forces a finite confluence handoff row through the displayed frontier, premise ledger, residual, vertex, join, diamond-cell, obstruction, transport, replay, provenance, and local NameCert rows.

Independence witness: metacic-parallel-diamond-frontier, closed-parallel-diamond-premise-ledger, parallel-diamond-residual, metacic-residual-join-determinacy: none is bijective with $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ because this packet simultaneously carries the frontier row, closed premise ledger, residual handoff, finite grid vertices, candidate joins, diamond cells, retained obstructions, transport, replay, provenance, and local naming.

The expected Lean TasteGate surface is that lean4/BEDC/Derived/MetaCICResidualJoinGridUp/TasteGate.lean supplies Nontrivial $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ and FieldFaithful $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ witnesses for the displayed rows $F, L, R, V, J, D, O, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualJoinGrid}^{\uparrow}$ packet is generated from parallel-diamond frontier, closed premise ledger, residual handoff, residual-vertex, candidate-join, diamond-cell, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7258.1, Definition 7258.2, Definition 7258.3, and Definition 7258.4 seed the finite residual-join-grid route for the MetaCIC confluence audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGrid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close premise-free parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient term equality, global normalization, kernel consistency, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the residual-grid obligation, confluence handoff, nonescape, nontriviality, and field-faithfulness rows.

Chapter 7259

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$ packet records the regular-Cauchy schedule surface used when a Cauchy modulus family is read back as rational windows and then sealed as a real-facing name. It refines the schedule route of chapter 5887 through the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The chapter names the finite handoff from abstract schedule rows to regular rational readback; it does not select a limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7259.1 (Regular-Cauchy modulus space carrier)**.** *A* $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *schedule-family row that names the precision requests available to the packet.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger inspected by those schedules.*
3. *W is the regular-window alignment row, pairing each requested schedule entry with the stream window used for readback.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the aligned windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic bound compared against the regular readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after S, W, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for $M, S, W, Q, D, E, C, P, N$.*
8. *C records displayed* Cont *replay from a schedule request through the stream window, regular readback, dyadic tolerance, and terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for an unscheduled Cauchy oracle, quotient stream equality, a selected limit, countable choice, or an ambient completeness theorem.

Theorem 7259.2 (Regular-Cauchy modulus space NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$ *carrier*

$$R = (M, S, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusSpace}^{\uparrow}}$ *surface consists exactly of CauchyModulusSpace schedule admission, finite StreamName window admission, regular-window alignment, RegSeqRat readback, DyadicRatCore tolerance compatibility, terminal Real seal compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Thus the regular Cauchy schedule may be consumed by Real and Completion routes only as finite readback data over* $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 7259.1. The analytic rows are M, S, W, Q, D, E, and the structural rows are H, C, P, N.

The schedule row M names only the displayed requests already available from $\mathsf{CauchyModulusSpace}^{\uparrow}$. The stream row S and alignment row W attach each request to a finite window before any rational readback is inspected.

The readback and tolerance rows are then forced through Q and D. The real seal row E is available only after those regular and dyadic ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7259.3 (Regular-Cauchy RegSeqRat handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$ *carrier receives the regular Cauchy schedule only through the finite handoff*

$$M, S, W, Q, D, E, H, C, P, N.$$

In particular, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row is consumed before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparison and before the terminal* $\mathsf{Real}^{\uparrow}$ *seal, so the handoff exports no quotient stream equality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real or Completion read of the accepted packet. The carrier definition exposes the CauchyModulusSpace schedule row M before the finite stream window row S and the regular-window alignment row W.

The regular rational readback Q is the only row from which the consumer may read the scheduled windows. The dyadic ledger D compares those readings with displayed tolerances, and the terminal seal E is consumed only after that comparison has been named.

Componentwise transport H and displayed replay C preserve the same order of rows, while P and N record provenance and local naming. Hence the consumer receives a finite regular readback handoff, not any quotient, choice, selected limit, or ambient completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusSpace}^{\uparrow}$ packet is generated from a CauchyModulusSpace schedule row, finite StreamName windows, regular-window alignment, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyModulusSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7259.1, Definition 7259.2, and Definition 7259.3 seed the regular-Cauchy schedule handoff from CauchyModulusSpace rows to RegSeqRat readback and Real-seal consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyModulusSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the schedule-admission and RegSeqRat-handoff obligations.

Chapter 7260

A Concrete Naming Certificate for $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$ packet records the last finite estimate in the compact-uniform-continuity route: source compact-net coverage, source metric triangle control, pointwise modulus rows at the selected center, and the target metric triangle are displayed as one ledger before the $\mathsf{UniformModulus}^{\uparrow}$ handoff is consumed. It is anchored in the compact finite net route of chapter 60, the source and target metric surfaces of chapter 4230, the rational tolerance rows of chapter 37, and the downstream uniform-modulus row of chapter 369. The packet does not claim host compactness, a selected center outside the finite bundle, semantic continuity, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7260.1 (Compact-uniform source-target triangle ledger carrier)**.** *A* CompactUniformSourceTargetTriang *carrier is a finite* BHist *packet*

$$T = (K, X, Y, C, R, A, S, U, H, G, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, exposing the selected* ProbeBundle *center and the coverage witness for the source point.*
2. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the distance witness from the source point to the selected center and the source triangle witness for the second source point.*
3. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including the triangle witness that combines the two pointwise target estimates.*
4. *C is the pointwise-modulus row at the selected compact-net center.*
5. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance ledger, carrying the $\varepsilon/3$ target budgets and the source precision budget.*
6. *A is the center-alignment row, tying the compact-net center used by K to the center consumed by C, X, and Y.*
7. *S is the final source-triangle row, showing that the second source point lies inside the same pointwise-modulus window.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row.*

9. *H records componentwise* hsame *transport for $K, X, Y, C, R, A, S, U, G, P, N$.*
10. *G records displayed* Cont *replay from compact-net coverage through source triangle control, pointwise modulus, target triangle control, and the uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host compactness, an external center selector, semantic continuity, quotient point equality, or ambient completeness.

Theorem 7260.2 (Compact-uniform triangle ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$ *carrier*

$$T = (K, X, Y, C, R, A, S, U, H, G, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, source metric admission, target metric admission, pointwise-modulus admission, positive RatUp tolerance admission, center alignment, final source-triangle admissibility, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.*

Proof. Project the accepted carrier of Definition 7260.1. The estimate rows are K, X, Y, C, R, A, S, U, and the structural rows are H, G, P, N.

The compact finite-net row K supplies the selected center only as displayed bundle data. The alignment row A keeps that center identical to the one used by the source metric row X and the pointwise-modulus row C.

The source triangle row S is then forced through X and the RatUp ledger R. The target triangle row in Y can be consumed only after the two pointwise target estimates from C have been displayed.

Finally the handoff row U receives the same finite estimate by the displayed transport, replay, provenance, and naming rows. Since no excluded host object is a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 7260.3 (Compact-uniform final source-target estimate)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$ *carrier receives the final compact-uniform estimate only through the finite route*

$$K, X, Y, C, R, A, S, U, H, G, P, N.$$

When K selects a center c, X displays source distances from x and x' to c, C displays the two pointwise target estimates at c, and Y displays the target triangle row, the handoff U certifies the tolerance-only uniform-modulus ledger without adding a host compactness theorem, a separate center selector, quotient point equality, semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the compact-net row K and the alignment row A. They give one displayed center for the whole route, so the source triangle and target triangle cannot choose different compact-net witnesses.

Use the source metric row X and the source-triangle row S. The source point x is covered by the compact net, and the displayed source distance from x to x' keeps x' inside the same pointwise-modulus window named by R.

Apply the pointwise-modulus row C at the selected center. It supplies the two target estimates from the center to the two source points, using only the positive RatUp tolerance ledger.

The target metric row Y composes those two target estimates by its triangle row. The continuation row G and provenance row P then deliver exactly this finite estimate to the $\mathsf{UniformModulus}^{\uparrow}$ handoff U, while H and N record classifier-stable transport and local naming. No absent host object can be read from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}$ packet is generated from compact finite-net coverage, source MetricSpace triangle rows, target MetricSpace triangle rows, pointwise-modulus rows, positive RatUp tolerance ledgers, center alignment, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7260.1, Definition 7260.2, and Definition 7260.3 seed the finite source-and-target triangle ledger used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformSourceTargetTriangleLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host compactness, semantic continuity, quotient point equality, selected centers outside the finite bundle, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the twelve-row carrier and prove the NameCert obligation and final-estimate theorems.

Chapter 7261

A Concrete Naming Certificate for $\mathsf{TightProbabilityMeasure}^{\uparrow}$

Following Prokhorov and Billingsley, the $\mathsf{TightProbabilityMeasure}^{\uparrow}$ packet records the finite tightness surface of a probability measure before weak-convergence or compactness consumers inspect it. It starts from the probability source of chapter 184, the event-surface row of chapter 92, and the pushforward law row of chapter 186; it then hands the same finite outside-mass and compact-test route to the Prokhorov-facing surface of chapter 4008. The packet names tight probability-measure data through displayed certificates, not through measure completion, selected compact sets, countable choice, host weak compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7261.1 (Tight probability measure carrier)**.** *A* $\mathsf{TightProbabilityMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (P, M, D, K, E, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the total-mass and event-family surface.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *event-surface row through which the inspected outside event and its mass are read.*
3. *D is the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row used when the same probability mass is consumed as a law on a target measurable surface.*
4. *K is the compact-test row naming the finite compactness request whose outside event is being measured.*
5. *E is the real-valued outside-mass error row, read through the displayed* $\mathsf{Real}^{\uparrow}$ *comparison surface.*
6. *R is the finite tolerance ledger aligning the compact-test request, the outside event in M, the pushforward row in D, and the error row E.*
7. *H records componentwise* hsame *transport for the probability source, measure event, distribution law, compact test, outside-mass row, tolerance ledger, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from probability-source admission to event-surface inspection, distribution readback, compact-test exposure, outside-mass comparison, and tolerance-ledger readback.*
9. *Q is the* Pkg *provenance over* $\mathsf{TightProbabilityMeasure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{ProkhorovTightProbability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightProbabilityMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, M, D, K, E, R, H, C, Q, N$. It has no coordinate for measure completion, selected compact subsets, countable choice, host weak compactness, quotient events, or ambient real completeness.

Theorem 7261.2 (Tight probability measure NameCert obligations)**.** *For every accepted* $\mathsf{TightProbabilityMeasure}^{\uparrow}$ *carrier*

$$T = (P, M, D, K, E, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{TightProbabilityMeasure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the probability source, measure event surface, distribution law, compact-test request, outside-mass error row, tolerance ledger, transport, replay, provenance, and local naming;*
2. *probability-measure compatibility, requiring the* $\mathsf{ProbSpace}^{\uparrow}$ *row P and* $\mathsf{Measure}^{\uparrow}$ *row M to name the same mass surface before a tightness read is accepted;*
3. *distribution compatibility, requiring the* $\mathsf{Distribution}^{\uparrow}$ *row D to consume the same measured event through its pushforward law route rather than through a host law equality;*
4. *compact outside-mass exposure, requiring K, E, R to display the compact-test request, real outside-mass bound, and tolerance ledger before any Prokhorov-facing consumer inspects the packet;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through H, C, Q, N;*
6. *nonescape, excluding measure completion, selected compact subsets, countable choice, host weak compactness, quotient events, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No tight probability-measure obligation is stored outside these rows.

Proof. Project the accepted carrier. The probability source P, event-surface measure row M, and distribution law row D are the only probability-facing coordinates.

The compact side is finite. The compact-test request K, outside-mass row E, and tolerance ledger R must be displayed before a tightness consumer can inspect the packet.

Transport and replay preserve the same route through H and C. Provenance Q and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, the local NameCert surface cannot supply measure completion, selected compact subsets, countable choice, host weak compactness, quotient events, or ambient real completeness. □

Theorem 7261.3 (Tight probability measure Prokhorov handoff)**.** *Every Prokhorov-facing consumer of an accepted* $\mathsf{TightProbabilityMeasure}^{\uparrow}$ *carrier reads only the finite route*

$$\mathsf{ProbSpace}^{\uparrow} \longmapsto \mathsf{Measure}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto K, E, R \longmapsto \mathsf{ProkhorovTightProbability}^{\uparrow}$$

together with H, C, Q, N. Thus the handoff to chapter 4008 is a displayed probability-source, measure-event, distribution-law, compact-test, outside-mass, tolerance-ledger, transport, replay, provenance, and local-naming route, with no measure completion, selected subsequence, countable-choice selector, host weak compactness theorem, quotient event, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the NameCert obligations of Definition 7261.2. They force the probability source, measure event surface, and distribution law to be displayed before the compact outside-mass route is consumed.

Next read the compact and tolerance rows. The rows K, E, R are the only compact-test, outside-mass, and tolerance coordinates available to the carrier, so the Prokhorov-facing consumer receives those rows rather than a host compactness theorem.

Finally pass the displayed route to the Prokhorov tight probability surface. The structural rows H, C, Q, N transport, replay, source, and name the same finite packet, and the carrier has no remaining coordinate from which a consumer could extract measure completion, selected subsequences, countable choice, weak compactness, quotient events, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{TightProbabilityMeasure}^{\uparrow}$ tightness read factors through $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, a displayed compact-test row, an outside-mass $\mathsf{Real}^{\uparrow}$ row, and a finite tolerance ledger before any $\mathsf{ProkhorovTightProbability}^{\uparrow}$ consumer can inspect it.

Independence witness: probspace, measure, distribution, prokhorov-tight-probability: $\mathsf{TightProbabilityMeasure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries probability-source admission, event-surface measure access, pushforward-law compatibility, compact outside-mass data, tolerance alignment, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TightProbabilityMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{TightProbabilityMeasure}^{\uparrow}$ packet is generated from ProbSpaceUp source rows, MeasureUp event-surface rows, DistributionUp pushforward rows, compact-test rows, outside-mass RealUp rows, a finite tolerance ledger, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightProbabilityMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7261.1, Definition 7261.2, and Definition 7261.3 seed the finite tight probability-measure certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightProbabilityMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, weak compactness, selected compact sets, selected subsequences, countable choice, quotient events, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, probability-measure compatibility, distribution compatibility, compact outside-mass exposure, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TightProbabilityMeasureU

Chapter 7262

A Concrete Naming Certificate for CauchyLimitUniquenessModulus$^{\uparrow}$

Following Bishop and Bridges, the CauchyLimitUniquenessModulus$^{\uparrow}$ packet records the finite modulus data used to compare two regular Cauchy real-limit routes without quotienting streams or appealing to host real equality. It sits after the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the regular Cauchy limit uniqueness packet of chapter 3942, and the real-limit uniqueness seal of chapter 1039. The packet names a displayed finite-means uniqueness modulus for Real and Completion consumers; it is not a completed-real equality theorem, a quotient-stream principle, a selected limit, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 7262.1 (Cauchy limit uniqueness modulus carrier)**.** *A* CauchyLimitUniquenessModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L_0, L_1, B, W, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two regular Cauchy limit source rows whose terminal reads are being compared.*
2. B *is the* DyadicRatCore$^{\uparrow}$ *tail-budget row naming the requested dyadic radius and its common comparison scale.*
3. W *is the paired* StreamName$^{\uparrow}$ *finite tail-window row, including the two observations used after the displayed unary threshold.*
4. R *is the* RegSeqRat$^{\uparrow}$ *readback row for the two scheduled tails.*
5. U *is the* RegularCauchyLimitUniqueness$^{\uparrow}$ *route row binding the paired tails to one uniqueness read.*
6. E *is the terminal* Real$^{\uparrow}$ *seal row, available only after* B, W, R, U *have been displayed.*
7. H *records componentwise* hsame *transport for* $L_0, L_1, B, W, R, U, E, C, P, N$.
8. C *records displayed* Cont *replay through the tail-budget, window, readback, uniqueness, and terminal seal rows.*
9. P *is the* Pkg *provenance over* CauchyLimitUniquenessModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyLimitUniqueness$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for host real equality, quotient stream equality, a selected limiting representative, a hidden modulus oracle, countable choice, trichotomy, or ambient completion.

Lean4 ✓ `BEDC.Derived.CauchyLimitUniquenessModulusUp.CauchyLimitUniquenessModulusTasteGate`
`single carrier alignment`

Definition 7262.2 (Cauchy limit uniqueness modulus classifier)**.** *The* $\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}$ *classifier identifies accepted packets only through displayed agreement of* $L_0, L_1, B, W, R, U, E, H, C, P, N$*. In particular, the classifier refuses equality that preserves the two source rows but changes the dyadic budget, the finite StreamName observations, the RegSeqRat readback, the uniqueness route, the terminal Real seal, or any structural transport, replay, provenance, or local naming row.*

Theorem 7262.3 (Cauchy limit uniqueness modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}$ *carrier* $M = (L_0, L_1, B, W, R, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitUniquen}}$ *surface consists exactly of carrier admission, classifier exactness, dyadic tail-budget stability, finite StreamName window stability, RegSeqRat readback compatibility, regular Cauchy uniqueness routing, terminal Real-seal exposure, componentwise transport, displayed replay, package provenance, and local naming. Every uniqueness read for a requested dyadic radius factors through these rows.*

Proof. Project the carrier of Definition 7262.1. The analytic rows are the paired source rows L_0, L_1, dyadic tail budget B, finite window W, regular readback R, uniqueness route U, and terminal seal E.

Apply the classifier of Definition 7262.2. It makes the five NameCert obligations visible: carrier admission for the displayed tuple, exact comparison through the classifier row, stability of the dyadic and StreamName windows, ledger compatibility for the RegSeqRat and Real-seal route, and nonescape from the structural rows H, C, P, N.

The regular Cauchy uniqueness route is read through chapter 3942 and the terminal seal is read through chapter 1039. Since neither route supplies host real equality, quotient stream equality, a selected limit, or ambient completion, every requested dyadic radius remains inside the displayed finite modulus ledger. □

Falsifiable prediction: A Real-facing uniqueness consumer that identifies two regular Cauchy limit reads while omitting the dyadic budget row, the paired StreamName observations, the RegSeqRat readback, or the regular Cauchy uniqueness route is rejected by the $\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}$ classifier.

Independence witness: dyadicratcore, streamname, regseqrat, regular-cauchy-limit-uniqueness, real-limit-uniqueness-seal, real: no listed sibling is bijective to $\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}$ because this packet jointly carries two source rows, one dyadic tail budget, paired finite observations, regular readback, uniqueness routing, terminal Real sealing, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}$ packet is generated from paired regular Cauchy limit rows, a DyadicRatCore tail budget, finite StreamName observations, RegSeqRat readback, RegularCauchyLimitUniqueness routing, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7262.1, Definition 7262.2, and Definition 7262.3 seed the finite uniqueness-modulus route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitUniquenessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, host real equality, selected limits, countable choice, trichotomy, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic budget, paired finite observations, RegSeqRat readback, uniqueness route, terminal Real seal, transport, replay, provenance, and local naming for BEDC.Derived.CauchyLimitUniqueness

Chapter 7263

A Concrete Naming Certificate for $\mathsf{PoincareDiskIsometry}^{\uparrow}$

The $\mathsf{PoincareDiskIsometry}^{\uparrow}$ packet realizes the reversible disk-transport branch of chapter 9463 as finite Poincare-disk isometry data. It sits after the disk carrier of chapter 377, the metric rows of chapter 3474 and chapter 6209, the disk-automorphism route of chapter 6776, and the boundary-transport packet of chapter 6267. The packet names reversible hyperbolic disk transport; it is not a completed hyperbolic plane, a global automorphism group, quotient boundary equality, a non-invertible disk map, or a physical observer law.

Definition 7263.1 (Poincare disk isometry carrier)**.** *A* $\mathsf{PoincareDiskIsometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, F, G, A, B, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *source-and-target disk row for the displayed points under inspection.*
2. *F is the disk-map row naming the finite transport on those displayed points.*
3. *G is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *or* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *metric-preservation row for every displayed source-target pair.*
4. *A is the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *reversible-map row, including inverse and composition read-backs for F.*
5. *B is the* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *boundary row showing that ideal-boundary reads are transported only through displayed disk data.*
6. *Q is the refusal row for non-invertible disk maps, metric-changing maps, omitted boundary evidence, and host geometry shortcuts.*
7. *H records componentwise* hsame *transport for $D, F, G, A, B, Q, C, P, N$.*
8. *C records displayed* Cont *replay through disk coordinates, map rows, metric preservation, automorphism, boundary transport, and refusal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskIsometry}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskIsometry}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, F, G, A, B, Q, H, C, P, N$ and has no coordinate for a non-invertible transport, hidden metric-preservation proof, completed disk model, quotient boundary object, or physical observer trajectory.

Theorem 7263.2 (Poincare disk isometry NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDiskIsometry}^{\uparrow}$ *carrier* $I = (D, F, G, A, B, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskIsometry}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, reversible map admission, metric-preservation exactness, boundary-transport compatibility, refusal of non-invertible or metric-changing witnesses, componentwise transport, displayed replay, package provenance, and local naming. A disk-map witness that lacks any one of F, G, A, B, Q is not an accepted isometry packet.*

Proof. Project the carrier of Definition 7263.1. The geometric rows are the disk observations D, disk-map row F, metric-preservation row G, automorphism row A, boundary-transport row B, and refusal row Q.

Metric preservation is read through the existing hyperbolic and Poincare disk metric packets, while reversibility is read through the disk-automorphism packet. The boundary row keeps ideal-end transport attached to the same displayed disk data rather than to a quotient boundary.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a non-invertible map, hidden metric proof, completed disk model, physical observer law, or quotient boundary object, every accepted isometry read remains inside the displayed NameCert surface. □

Falsifiable prediction: A disk-map witness that preserves the displayed point row but fails metric preservation, lacks an inverse, or omits the boundary-transport row is rejected by the $\mathsf{PoincareDiskIsometry}^{\uparrow}$ classifier.

Independence witness: unitdisk, hyperbolic-metric, poincare-disk-automorphism, hyperbolic-boundary-transport: no listed sibling is bijective to $\mathsf{PoincareDiskIsometry}^{\uparrow}$ because UnitDiskUp supplies disk coordinates without reversible metric preservation, HyperbolicMetricUp supplies metric reads without a disk-map inverse, PoincareDiskAutomorphismUp supplies reversible maps without the isometry refusal row, and boundary transport supplies boundary routing without the full disk-map, metric, inverse, refusal, transport, replay, provenance, and local NameCert surface.

The Lean `TasteGate.lean` boundary for $\mathsf{PoincareDiskIsometry}^{\uparrow}$ must provide explicit Nontrivial PoincareDiskIsom and FieldFaithful $\mathsf{PoincareDiskIsometry}^{\uparrow}$ instances preserving the displayed rows $D, F, G, A, B, Q, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskIsometry}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PoincareDiskIsometry}^{\uparrow}$ packet is generated from UnitDisk coordinates, a displayed disk map, HyperbolicMetric or PoincareDiskMetric preservation rows, PoincareDiskAutomorphism inverse rows, boundary transport, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PoincareDiskIsometry}^{\uparrow}}$, seed)

Scope closed: Definition 7263.1 and Definition 7263.2 seed the finite reversible hyperbolic disk-transport route over $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, and $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PoincareDiskIsometry}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed hyperbolic disk geometry, global automorphism groups, quotient boundary equality, non-invertible disk maps, or physical observer dynamics.

Upgrade path: Obligation closure requires checked carrier admission, disk coordinates, reversible map rows, metric preservation, boundary transport, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDiskIsometryUp

Chapter 7264

A Concrete Naming Certificate for $\mathsf{NewtonCotes}^{\uparrow}$

Following Newton–Cotes quadrature in classical numerical analysis, the $\mathsf{NewtonCotes}^{\uparrow}$ packet records a finite interpolation-node quadrature rule as BEDC NameCert data before any integration-facing consumer may read an approximate integral. It sits after the quadrature carrier of chapter 227, the Darboux-sum surface of chapter 1880, the polynomial source of chapter 50, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names only a finite node-and-weight rule with a displayed error budget; it does not import host integration, measure theory, selected limits, quotient function equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7264.1 (Newton–Cotes carrier)**.** *A* $\mathsf{NewtonCotes}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, N, W, S, A, D, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the interval endpoint row, naming the displayed left and right endpoints and the orientation data used by the quadrature rule.*
2. *N is the finite node ledger. It records the node order, endpoint inclusion flags, and node positions inside the same interval row.*
3. *W is the rational weight ledger, assigning one rational weight to each displayed node and recording the finite denominator schedule.*
4. *S is the sample row, exposing the inspected function values only at the nodes listed by N.*
5. *A is the finite weighted-sum row that combines S with W before any regular-rational or Real-facing readback is opened.*
6. *D is the dyadic error-budget row, recording the tolerance request, mesh width, and finite remainder budget available to the rule.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the weighted-sum row and the dyadic budget.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, N, W, S, A, D, R have been displayed.*
9. *H records componentwise* hsame *transport for endpoint, node, weight, sample, weighted-sum, budget, handoff, seal, provenance, replay, and naming rows.*

10. *C records displayed* Cont *replay from an interval request through nodes, weights, samples, weighted sum, dyadic budget, regular-rational handoff, and Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{NewtonCotes}^{\uparrow}$, $\mathsf{Quadrature}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *L is the local* $\mathsf{NameCert}_{\mathsf{NewtonCotes}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, N, W, S, A, D, R, E, H, C, P, L$. It has no coordinate for a host integral, measure-theoretic completion, an unbounded function space, quotient function equality, selected limiting partitions, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7264.2 (Newton–Cotes NameCert obligations)**.** *For every accepted* $\mathsf{NewtonCotes}^{\uparrow}$ *carrier*

$$Q = (I, N, W, S, A, D, R, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{NewtonCotes}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the endpoint, node, weight, sample, weighted-sum, dyadic-budget, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *node-and-weight coverage, requiring N and W to list the same finite node positions and rational weights over the endpoint row I;*
3. *sample exactness, requiring each entry of S to be a displayed sample at a node named by N;*
4. *weighted-sum and dyadic-budget exactness, requiring A to use only the samples and rational weights in S, W and requiring D to bind the requested quadrature tolerance before readback;*
5. *RegSeqRat and Real-seal non-escape, requiring R and E to be exposed only after I, N, W, S, A, D and the structural rows H, C, P, L are visible.*

No Newton–Cotes obligation is stored outside these finite rows.

Proof. Project the accepted carrier. The tuple lists exactly the interval endpoint, finite node ledger, rational weight ledger, sample row, weighted-sum row, dyadic budget, regular-rational handoff, Real seal, and structural rows.

Node-and-weight coverage is read from N and W. The two rows have the same finite index surface over the endpoint row I, so there is no hidden node, extra weight, or external partition.

Sample exactness is the row S. Since S is indexed by N, every sampled value is attached to a displayed node before the weighted sum can be inspected.

The numerical handoff is then finite. Row A combines only S and W, row D binds the dyadic error request, and R, E expose the regular-rational and Real-facing readback only after those rows are visible. Transport, replay, provenance, and local naming through H, C, P, L preserve the same route and add no host integration or completeness principle. □

Theorem 7264.3 (Newton–Cotes RegSeqRat quadrature handoff)**.** *Every quadrature-facing read of an accepted* $\mathsf{NewtonCotes}^{\uparrow}$ *packet factors through the finite route*

$$I, N, W, S, A, D \longmapsto R \longmapsto E.$$

In particular, a consumer must expose the endpoint row, node ledger, rational weight ledger, sample row, finite weighted-sum row, and dyadic error-budget row before the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal are available. The packet exports no host integral, measure-theoretic completion, selected limiting partition, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7264.2. They place every consumer read downstream of the carrier rows $I, N, W, S, A, D, R, E, H, C, P, L$.

Read the quadrature data in order. The endpoint row I fixes the interval, N lists the nodes, W lists the rational weights on those nodes, and S supplies only the corresponding samples.

The finite weighted sum and error budget are the last numerical rows before readback. Row A combines the displayed samples and weights, while D records the dyadic tolerance and finite remainder budget attached to the same rule.

Only then may the consumer inspect R and E. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal are therefore exposed as a finite Newton–Cotes quadrature route, not as a host integration theorem, measure-theoretic limit, quotient function equality, compactness principle, or ambient completeness claim. □

Falsifiable prediction: A $\mathsf{NewtonCotes}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without displaying the interval endpoint row, node ledger, rational weight ledger, sample row, weighted-sum row, and dyadic error-budget row refutes the carrier of Definition 7264.1.

Independence witness: quadrature, darbouxsum, polynomial, regseqrat, real: $\mathsf{NewtonCotes}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, node rows, rational weights, sample rows, finite weighted sums, dyadic error budgets, regular-rational handoff, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonCotes}^{\uparrow}}$.

Constructive story: A $\mathsf{NewtonCotes}^{\uparrow}$ packet is generated from an interval endpoint row, finite node ledger, rational weight ledger, sample row, finite weighted-sum row, dyadic error-budget row, RegSeqRat handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NewtonCotes}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7264.1, Definition 7264.2, and Definition 7264.3 seed the finite Newton–Cotes quadrature carrier and its RegSeqRat-to-Real handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NewtonCotes}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly refuses host integration, measure theory, selected limits, quotient function equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, node-and-weight coverage, sample exactness, finite weighted-sum exactness, dyadic error-budget exposure, RegSeqRat handoff, Real-seal ordering, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.NewtonCotesUp.

Chapter 7265

A Concrete Naming Certificate for $\mathsf{SimpsonRule}^{\uparrow}$

Following Simpson and the Newton–Cotes quadrature tradition, the $\mathsf{SimpsonRule}^{\uparrow}$ packet records the finite three-point closed quadrature rule as BEDC NameCert data before an integration-facing consumer may read its approximation or error row. It sits after the finite exactness interface of chapter 227, the Newton–Cotes node-and-weight packet of chapter 7264, the Taylor remainder packet of chapter 4732, the tagged integral route of chapter 1526, and the terminal real-name seal of chapter 38. The packet names only the midpoint/endpoints rule with displayed weights, a displayed Taylor or Peano error row, and the regular rational to Real handoff; it does not import host integration, measure theory, selected limits, quotient function equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7265.1 (Simpson rule carrier)**.** *A* $\mathsf{SimpsonRule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, M, E, W, Q, T, R, D, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval row, naming the displayed left and right endpoints, the orientation data, and the compact-interval source used by the rule.*
2. *M is the midpoint row, recording the finite construction of the midpoint between the two endpoint names and its membership in the same interval row.*
3. *E is the endpoint sample row, exposing the two endpoint samples and the midpoint sample as the only function reads used by the rule.*
4. *W is the Simpson weight row, recording the finite rational weights $1, 4, 1$ together with the common denominator and interval-width factor.*
5. *Q is the finite quadrature row, combining E and W into the displayed three-point weighted sum.*
6. *T is the Taylor or Peano error row, recording the displayed fourth-order remainder budget or Peano kernel budget used by the packet.*
7. *R is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *and Darboux handoff row, exposing the tagged-integral or Darboux-facing consumer route only after I, M, E, W, Q, T have been displayed.*
8. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal row, recording the regular-rational read-back, dyadic tolerance schedule, and terminal Real seal for the weighted sum and error row.*

9. *H records componentwise* hsame *transport for interval, midpoint, sample, weight, quadrature, error, handoff, seal, provenance, replay, and naming rows.*
10. *C records displayed* Cont *replay from the interval request through midpoint construction, endpoint and midpoint samples, Simpson weights, finite quadrature row, error row, integral handoff, regular-rational readback, dyadic tolerance, and Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{SimpsonRule}^{\uparrow}$, $\mathsf{Quadrature}^{\uparrow}$, $\mathsf{NewtonCotes}^{\uparrow}$, $\mathsf{TaylorRemainder}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SimpsonRule}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, E, W, Q, T, R, D, H, C, P, N$. It has no coordinate for a host integral, an undisplayed mesh limit, a selected antiderivative, quotient function equality, a hidden fourth derivative, an open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7265.2 (Simpson rule NameCert obligations)**.** *For every accepted* $\mathsf{SimpsonRule}^{\uparrow}$ *carrier*

$$S = (I, M, E, W, Q, T, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SimpsonRule}^{\uparrow}}$ *surface consists exactly of compact-interval admission, midpoint construction, endpoint and midpoint sample exposure, Simpson weight exposure, finite quadrature-row exactness, Taylor or Peano error-row exposure, Riemann and Darboux handoff, RegSeqRat/Dyadic/Real sealing, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host integration, undisplayed mesh limits, selected antiderivatives, quotient function equality, hidden differentiability data, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7265.1. The interval row I and midpoint row M are the only source rows from which the three sampling sites can be read.

The numerical rule is then finite. Row E exposes exactly the two endpoint samples and the midpoint sample, while W exposes the rational weight pattern $1, 4, 1$ with its denominator and interval-width factor.

The quadrature and error rows are downstream of those displayed samples. Row Q combines only E and W, and row T records the Taylor or Peano error budget without adding a hidden derivative or host analytic function.

Finally, R, D, H, C, P, N transport, replay, source, seal, and name the same route. Since no carrier coordinate stores a host integral, selected limit, quotient function equality, compactness principle, or ambient real completeness, the listed rows exhaust the local certificate surface. □

Theorem 7265.3 (Simpson rule quadrature-error non-escape)**.** *Every Riemann-facing, Darboux-facing, or real-seal read of an accepted* $\mathsf{SimpsonRule}^{\uparrow}$ *packet factors through*

$$I, M, E, W, Q, T \longmapsto R \longmapsto D.$$

In particular, a consumer must expose the compact interval, midpoint row, endpoint and midpoint samples, Simpson weights, finite quadrature row, and Taylor or Peano error row before the integral handoff and RegSeqRat/Dyadic/Real seal are available. The packet exports no host integration theorem, undisplayed limiting partition, quotient function equality, hidden differentiability assumption, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer read of the accepted packet. The NameCert obligations of Definition 7265.2 place every read downstream of $I, M, E, W, Q, T, R, D, H, C, P, N$.

The endpoint and midpoint data must be displayed first. The interval row fixes the endpoints, the midpoint row fixes the middle sample site, and the sample row exposes only those three function reads.

The Simpson weights and finite weighted sum are then the only quadrature rows. The error row is attached to that same three-point rule, so an integral-facing read cannot bypass Q or T.

Only after those rows are present may R and D be inspected. The Riemann/Darboux handoff and the RegSeqRat/Dyadic/Real seal are therefore finite readback rows, not host analytic integration, quotient function equality, selected limiting partitions, compactness, or ambient completeness. □

Falsifiable prediction: A $\mathsf{SimpsonRule}^{\uparrow}$ consumer that reaches a $\mathsf{RiemannIntegral}^{\uparrow}$ handoff, Darboux-facing read, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, or $\mathsf{Real}^{\uparrow}$ seal without displaying I, M, E, W, Q, T refutes the carrier of Definition 7265.1.

Independence witness: quadrature, newton-cotes, taylor-remainder, riemannintegral, real: none is bijective to $\mathsf{SimpsonRule}^{\uparrow}$ because this packet jointly carries the compact interval, midpoint, endpoint and midpoint samples, Simpson weights, finite quadrature row, Taylor or Peano error row, Riemann/Darboux handoff, RegSeqRat/Dyadic/Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SimpsonRule}^{\uparrow}}$.

Constructive story: A $\mathsf{SimpsonRule}^{\uparrow}$ packet is generated from a compact interval row, midpoint row, endpoint and midpoint sample row, Simpson weight row, finite quadrature row, Taylor or Peano error row, Riemann/Darboux handoff row, RegSeqRat/Dyadic/Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimpsonRule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7265.1, Definition 7265.2, and Definition 7265.3 seed the finite three-point Simpson quadrature carrier and its error-to-readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimpsonRule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host integration, measure theory, selected limits, quotient function equality, hidden differentiability data, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint construction, three-sample exposure, Simpson weight exposure, quadrature-row exactness, Taylor or Peano error exposure, Riemann/Darboux handoff, RegSeqRat/Dyadic/Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.SimpsonRuleUp.

Chapter 7266

A Concrete Naming Certificate for $\mathsf{FiniteJacobian}^{\uparrow}$

Following Cartan and Spivak on finite-dimensional differential calculus, the $\mathsf{FiniteJacobian}^{\uparrow}$ packet records the derivative matrix of a finite coordinate-window presentation as BEDC NameCert data. It sits over the matrix surface of chapter 49, the linear-map handoff of chapter 48, the calculus route of chapter 3632, the real readback seal of chapter 38, and the metric neighbourhood discipline of chapter 57. The finite derivative table is consumed as a calculus-facing bridge from the $\mathsf{Mat}^{\uparrow}$ table through the $\mathsf{LinMap}^{\uparrow}$ handoff before $\mathsf{Real}^{\uparrow}$ error budgets and $\mathsf{Met}^{\uparrow}$ neighbourhood budgets are exposed. The packet is a finite derivative table and error-budget certificate; it does not import host differentiability, quotient function equality, selected limits, inverse-function data, implicit-function data, or countable choice.

Definition 7266.1 (FiniteJacobian carrier)**.** *A* $\mathsf{FiniteJacobian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (W, M, L, E, V, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite coordinate-window row: it records the source coordinate labels, target coordinate labels, base point window, and the finite list of directions inspected by the derivative read.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *derivative table row whose entries are the displayed coordinatewise linear coefficients for the window W.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *handoff row reading the same table as a finite linear map between the displayed coordinate windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *finite-difference error budget row: for each displayed direction it records the rational or dyadic tolerance by which the table M approximates the observed difference quotient.*
5. *V is the* $\mathsf{Met}^{\uparrow}$ *neighbourhood budget row that bounds the domain window on which the finite-difference observations may be replayed.*
6. *H records componentwise* hsame *transport for W, M, L, E, V, C, P, N.*
7. *C records displayed* Cont *replay from a coordinate-window read through the matrix table, linear-map handoff, real error budget, and metric neighbourhood budget.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteJacobian}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteJacobian}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient differentiability predicate, host function equality, selected limit, inverse map, implicit solver, quotient real equality, or countable-choice schedule.

Lean4 ✓ `BEDC.Derived.FiniteJacobianUp.FiniteJacobianTasteGate single carrier alignment`

Theorem 7266.2 (FiniteJacobian NameCert obligations)**.** *For every accepted* $\mathsf{FiniteJacobian}^{\uparrow}$ *carrier*

$$J = (W, M, L, E, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteJacobian}^{\uparrow}}$ *surface consists exactly of finite coordinate-window admission, Matrix derivative-table admission, LinearMap handoff, Real finite-difference error-budget compatibility, Metric neighbourhood-budget compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Thus a derivative consumer may read the finite Jacobian only as displayed matrix and linear-map data over* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Calculus}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 7266.1. The analytic rows are W, M, L, E, V, and the structural rows are H, C, P, N.

The coordinate-window row W fixes the finite source and target faces before any derivative entry is inspected. The matrix row M is therefore a finite table over that window rather than an ambient derivative operator.

The linear-map row L reads the same table as a finite handoff. The real error row E records the finite-difference tolerance for each displayed direction, and the metric row V records the neighbourhood on which those observations may be replayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since every obligation is one of the displayed rows and none of the excluded host objects is a coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7266.3 (FiniteJacobian matrix derivative handoff)**.** *Every inverse-function, implicit-function, ODE, PDE, or linearisation consumer of an accepted* $\mathsf{FiniteJacobian}^{\uparrow}$ *carrier receives the derivative only through the finite handoff*

$$W, M, L, E, V, H, C, P, N.$$

In particular, the $\mathsf{Mat}^{\uparrow}$ *derivative table is consumed before the* $\mathsf{LinMap}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *finite-difference error budget, and the* $\mathsf{Met}^{\uparrow}$ *neighbourhood budget. The handoff exports no host differentiability theorem, selected derivative limit, quotient function equality, inverse-function conclusion, implicit-function conclusion, ODE solution, PDE solution, or countable choice.*

Proof. Begin with a downstream analytic consumer of the accepted packet. The carrier first exposes the coordinate-window row W, so every later derivative read is attached to the same finite list of source directions and target coordinates.

The consumer then reads the matrix table M and only afterward the linear-map handoff L. This order makes the linear map a replay of the displayed derivative table, not a hidden ambient derivative.

The finite-difference budget E and neighbourhood budget V bound the table as real and metric data before any consumer can reuse it in an inverse, implicit, evolution, or linearisation route.

Componentwise transport H and displayed replay C preserve this order, while P and N record provenance and local naming. Hence the consumer receives a finite matrix derivative packet and none of the stronger analytic endpoints refused in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteJacobian}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteJacobian}^{\uparrow}$ packet is generated from finite coordinate-window rows, a displayed MatrixUp derivative table, a LinearMapUp handoff, RealUp finite-difference error budgets, MetricUp neighbourhood budgets, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteJacobian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7266.1, Definition 7266.2, and Definition 7266.3 seed the finite matrix derivative packet over coordinate windows, Matrix, LinearMap, Real error budgets, and Metric neighbourhood budgets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteJacobian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host differentiability, selected derivative limits, quotient function equality, inverse-function results, implicit-function results, ODE or PDE solvability, countable choice, Lean verification, or a standard bridge.

Upgrade path: Scoped treatment asks for checked carrier admission, coordinate-window exactness, matrix-table compatibility, linear-map handoff, finite-difference error control, neighbourhood-budget compatibility, transport, replay, provenance, and local naming.

Chapter 7267

A Concrete Naming Certificate for $\mathsf{BishopRegularRealLocator}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRegularRealLocator}^{\uparrow}$ packet records the locator surface carried by a regular Cauchy real name before any Real-facing approximation consumer reads it. It sits over the dyadic tolerance ledger of chapter 291, the finite window service of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies located approximation windows for regular Cauchy data; it is not quotient real equality, countable choice, trichotomy, host equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7267.1 (Bishop regular real locator carrier)**.** *A* $\mathsf{BishopRegularRealLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, A, E, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and rational-probe ledger, carrying the requested dyadic radius, lower and upper rational guards, and comparison slack.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from the regular Cauchy name at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational approximants attached to the same finite windows.*
4. *A is the located approximation-window row, recording the finite rational interval or apartness window accepted for the requested dyadic tolerance.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after D, S, R, A have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, stream window, regular rational readback, approximation-window, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an approximation request through D, S, R, A, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularRealLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealLocator}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, A, E, H, C, P, N$. It has no coordinate for quotient real equality, countable-choice schedules, trichotomy of arbitrary real comparisons, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BishopRegularRealLocatorUp.BishopRegularRealLocatorUp`

Theorem 7267.2 (Bishop regular real locator NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularRealLocato}$ *carrier $L = (D, S, R, A, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealLocator}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, finite stream window, regular rational readback, located approximation window, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *source agreement, requiring D, S, and R to refer to the same displayed regular Cauchy name before A is read;*
3. *locator exactness, requiring the approximation window in A to be bounded by the dyadic tolerance and rational guards named in D;*
4. *regular readback binding, requiring A to be justified by the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and not by a host real representative;*
5. *Real seal non-escape, requiring E to be downstream of the finite dyadic, stream, regular-rational, and approximation-window rows;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
7. *refusal of quotient real equality, countable choice, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No locator obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 7267.1. The only analytic rows exposed before the Real seal are the dyadic tolerance ledger D, the finite stream window S, the regular rational readback R, and the approximation window A.

Source agreement is read from the shared route D, S, R. The locator row A can answer a rational approximation request only after the requested dyadic tolerance and the inspected regular-rational window have both been displayed.

The Real-facing row E is downstream of that finite route. Transport through H and replay through C preserve the same dyadic, stream, readback, and approximation coordinates, while P and N package the local certificate.

Since the carrier has no coordinate for quotient equality, choice, trichotomy, host equality, or ambient completeness, none of those principles can enter the NameCert surface. □

Theorem 7267.3 (Bishop regular real locator approximation handoff)**.** *Every approximation consumer reading an accepted* $\mathsf{BishopRegularRealLocator}^{\uparrow}$ *carrier factors through the finite route*

$$D, S, R \longmapsto A \longmapsto E \longmapsto H, C, P, N.$$

The consumer receives a displayed rational window over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data. It receives no quotient real equality witness, no countable-choice schedule, no trichotomy oracle, no host equality comparison, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Begin with an approximation request for the accepted carrier. The request first reads D for the dyadic tolerance and rational guards, then opens the finite stream window S and the regular rational readback R attached to the same regular Cauchy name.

The located approximation window A is therefore a finite consequence of displayed regular-rational data. It is not a selected representative of a quotient real and does not require a comparison of completed host objects.

The final Real-facing handoff passes through E, H, C, P, N. These rows seal, transport, replay, source, and name the same route, so a downstream consumer can inspect the approximation window without importing countable choice, trichotomy, host equality, or ambient completeness. □

Theorem 7267.4 (Bishop regular real locator window composition)**.** *Suppose two accepted* BishopRegularRealLocat *reads expose the same* DyadicRatCore$^{\uparrow}$ *tolerance row and compatible* StreamName$^{\uparrow}$ *finite windows. Their locator-facing reads compose into a single finite route whose* RegSeqRat$^{\uparrow}$ *readback and terminal* Real$^{\uparrow}$ *seal remain downstream from the displayed windows. The composed route uses only* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows, and it exports no quotient real equality, countable-choice schedule, selected limit, trichotomy oracle, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 7267.1 for both reads. The visible analytic rows are the dyadic tolerance ledger, the finite stream windows, the regular rational readback, the located approximation window, and the terminal Real seal.

Apply the NameCert obligations of Definition 7267.2. They force source agreement and locator exactness through the displayed D, S, R, A rows before any Real-facing row can be consumed.

Use the approximation handoff of Definition 7267.3. Compatible stream windows over the same dyadic tolerance can be concatenated only at the finite-window level; the RegSeqRat$^{\uparrow}$ readback and the Real seal remain downstream from that display.

The structural rows then transport and replay the same composed route through H, C, P, N. Since no carrier coordinate stores quotient equality, choice, selected limits, trichotomy, or ambient completeness, the composition is only a locator-facing finite-window handoff. □

Falsifiable prediction: Every accepted BishopRegularRealLocator$^{\uparrow}$ packet exposing a Real-facing approximation handoff must display dyadic tolerance, StreamName window, RegSeqRat readback, located approximation, Real seal, hsame transport, Cont replay, Pkg provenance, and NameCert rows.

Independence witness: dyadicratcore, streamname, regseqrat, real: BishopRegularRealLocator$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a dyadic tolerance request, finite stream observation, regular rational readback, located approximation window, terminal Real seal, and local naming rows.

Closure status for NameCert$_{\text{BishopRegularRealLocator}^{\uparrow}}$.

Constructive story: A BishopRegularRealLocator$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, a StreamName finite-window row, a RegSeqRat regular readback row, a located approximation-window row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopRegularRealLocator}^{\uparrow}}$, seed)

Scope closed: Definition 7267.1, Definition 7267.2, Definition 7267.3, and Definition 7267.4 seed the finite locator surface for regular Cauchy real names and Real-facing approximation consumers.

Formal verification: FormalStatus(NameCert$_{\text{BishopRegularRealLocator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, countable choice, trichotomy, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, locator exactness, regular readback binding, Real seal non-escape, transport, replay, provenance, local naming, and approximation handoff for BEDC.Derived.BishopRegularRealLocatorUp.

Chapter 7268

A Concrete Naming Certificate for $\mathsf{ConstructiveLocatedReal}^{\uparrow}$

From Bishop and Bridges, the $\mathsf{ConstructiveLocatedReal}^{\uparrow}$ packet records a located real as finite dyadic locator data tied to regular rational readback and a terminal real-name seal. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Bishop regular real packet of chapter 4058, and the positive apartness witness of chapter 6450. The packet names the constructive located-real boundary needed by Real and Completion consumers; it does not claim quotient equality of Cauchy streams, ambient real completeness, real trichotomy, countable choice, selected representatives, or a chosen limit.

Definition 7268.1 (Constructive located real carrier)**.** *A* $\mathsf{ConstructiveLocatedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, E, A, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *locator row, carrying the dyadic radius, comparison slack, and positive tolerance budget used by each located read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at the dyadic addresses requested by D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after the dyadic locator, stream window, and regular rational readback rows have been displayed.*
5. *A is the apartness and refusal row, recording either a displayed positive separation through* $\mathsf{LocatedRealApartness}^{\uparrow}$ *or a finite refusal to provide trichotomy, quotient equality, countable choice, selected representatives, or ambient completeness.*
6. *F is the Bishop regular row, binding the* $\mathsf{BishopRegularReal}^{\uparrow}$ *route to the same dyadic windows and regular rational readback.*
7. *H records componentwise* hsame *transport for the dyadic locator, stream window, regular readback, Real seal, apartness/refusal row, Bishop regular row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located-real request through D, S, R, E, A, F, H.*

9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveLocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{BishopRegularReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, A, F, H, C, P, N$. It has no coordinate for host real equality, quotient real equality, a trichotomy decision, a choice schedule, a selected Cauchy representative, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ConstructiveLocatedRealUp`

Theorem 7268.2 (Constructive located real apartness stability)**.** *Let $L = (D, S, R, E, A, F, H, C, P, N)$ and $L' = (D', S', R', E', A', F', H', C', P', N')$ be accepted* $\mathsf{ConstructiveLocatedReal}^{\uparrow}$ *carriers. If H records componentwise* hsame *transport from each displayed row of L to the corresponding row of L', then every apartness or refusal read of L transports to L' only through*

$$D, S, R, E, A, F, H, C, P, N.$$

In particular, positive separation evidence remains attached to the same dyadic locator, finite StreamName window, RegSeqRat readback, and Real seal route, while refusal rows remain refusals of trichotomy, quotient equality, choice, selected representatives, selected limits, and ambient completeness.

Proof. Project the two accepted carriers. The analytic rows are the dyadic locator D, the finite stream window S, the regular rational readback R, and the terminal Real seal E; the apartness/refusal row A is downstream of those rows.

Componentwise transport through H carries D, S, R, E, A, F to the corresponding rows of L'. Since C only replays the displayed route and P, N only source and name the same packet, no new comparison principle or equality test is introduced during transport.

Therefore a positive separation witness remains a finite located-apartness witness, and a refusal row remains an explicit refusal of non-displayed real principles. The transported packet can be consumed as constructive located-real data only through the stated rows. □

Theorem 7268.3 (Constructive located real regular NameCert)**.** *For every accepted* $\mathsf{ConstructiveLocatedReal}^{\uparrow}$ *carrier $L = (D, S, R, E, A, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedReal}^{\uparrow}}$ *surface consists exactly of dyadic locator admission, finite StreamName window admission, RegSeqRat readback, Real seal admission, apartness/refusal admission, Bishop regular route admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the displayed rows $D, S, R, E, A, F, H, C, P, N$.*

Proof. Read the carrier of Definition 7268.1. The regular readback row R is available only after the dyadic locator D and finite stream window S have been displayed, and the terminal Real seal E is downstream of that readback.

The apartness/refusal row A is row-local: it records positive separation when such a finite witness is displayed, and otherwise records refusals of trichotomy, quotient equality, choice, selected representatives, selected limits, and ambient completeness. The Bishop regular row F binds the same dyadic-window and regular-readback route to the classical Bishop regular-real surface without adding a host real oracle.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Hence the local NameCert obligations are exhausted by the listed rows, and no unregistered equality, completeness, or selection principle can enter the constructive located-real boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveLocatedReal}^{\uparrow}$ packet is generated from a DyadicRatCore locator row, StreamName finite windows, RegSeqRat readback, a RealUp seal, apartness and refusal rows, a BishopRegularReal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveLocatedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7268.1, Definition 7268.2, and Definition 7268.3 seed the finite constructive located-real boundary over DyadicRatCore, StreamName, RegSeqRat, Real, located-apartness, Bishop regular, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveLocatedReal}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.ConstructiveLocatedRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy streams, ambient real completeness, real trichotomy, countable choice, selected representatives, selected limits, theorem checking, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic locator exactness, StreamName window admission, RegSeqRat readback, Real seal non-escape, apartness and refusal stability, Bishop regular route admission, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveLocatedRealUp.

Chapter 7269

A Concrete Naming Certificate for $\mathsf{BarrelledSpace}^{\uparrow}$

Following Schaefer and Bourbaki on locally convex spaces, the $\mathsf{BarrelledSpace}^{\uparrow}$ packet records the traditional barrel condition as finite BEDC NameCert data. It sits after the locally convex carrier of chapter 3386, the finite seminorm-family surface of chapter 5868, the Banach-space boundary of **??**, and the theorem-level uniform boundedness route of chapter 3375. The packet gives Banach–Steinhaus and locally convex consumers a named barrelledness certificate without importing full Banach–Steinhaus, countable choice, Hahn–Banach separation, quotient dual equality, or bridge checking.

Definition 7269.1 (BarrelledSpace carrier)**.** *A* $\mathsf{BarrelledSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, S, A, D, V, C, K, F, U, H, T, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *source row supplying the topological-vector, convex-neighbourhood, and local-convexity surface.*
2. *S is the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite seminorm-family row through which boundedness probes and neighbourhood reads are inspected.*
3. *A is the absorbing test row for candidate barrels in the displayed locally convex source.*
4. *D is the balanced test row.*
5. *V is the convex test row.*
6. *C is the closed test row.*
7. *K is the barrel predicate row, admitting a candidate precisely when A, D, V, C are displayed for the same neighbourhood carrier.*
8. *F is the bounded-family test row, reading pointwise bounded operator-family windows through the same locally convex and seminorm rows.*
9. *U is the* $\mathsf{BanachSteinhaus}^{\uparrow}$ *consumer row that may receive a uniform-boundedness handoff only after the barrel predicate and bounded-family rows have been exposed.*
10. *H records componentwise* hsame *transport for $L, S, A, D, V, C, K, F, U$.*

11. *T records displayed* Cont *replay from barrel reads and bounded-family requests through the same rows.*
12. *P is the* Pkg *provenance over* $\mathsf{BarrelledSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{BanachSteinhaus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BarrelledSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host topological-vector-space equality, a selected dual, a quotient completion, a full Banach–Steinhaus theorem, a Hahn–Banach separator, or a choice-selected family of barrels.

Theorem 7269.2 (BarrelledSpace NameCert obligations)**.** *For every accepted* $\mathsf{BarrelledSpace}^{\uparrow}$ *carrier* $B = (L, S, A, D, V, C, K, F, U, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BarrelledSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the locally convex source row L and finite seminorm-family row S;*
2. *barrel recognition through the absorbing, balanced, convex, and closed tests A, D, V, C and the predicate row K;*
3. *bounded-family compatibility through F, with all pointwise bounded windows read over the same L, S surface;*
4. *Banach–Steinhaus consumer routing through U, available only after K and F have been displayed;*
5. *nonescape through H, T, P, N, excluding host Banach–Steinhaus, countable choice, Hahn–Banach strong forms, selected duals, quotient completions, and bridge checking.*

Proof. Project the carrier of Definition 7269.1. The tuple $L, S, A, D, V, C, K, F, U, H, T, P, N$ is the whole accepted packet, so carrier admission and local naming are finite row reads.

The barrel predicate is not primitive. It is assembled from the absorbing, balanced, convex, and closed tests A, D, V, C over the locally convex source L and seminorm-family row S, so a consumer cannot detach barrelledness from the displayed locally convex carrier.

The bounded-family test F uses the same L, S surface. The $\mathsf{BanachSteinhaus}^{\uparrow}$ row U is therefore a downstream consumer of the barrel and bounded-family rows rather than a proof of the full Banach–Steinhaus theorem.

Finally the rows H, T, P, N transport, replay, source, and name the same packet. Since no row names choice, Hahn–Banach separation, quotient completion, selected dual data, or bridge checking, the five listed rows exhaust the NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{BarrelledSpace}^{\uparrow}$ packet must expose the locally convex source, finite seminorm-family row, absorbing, balanced, convex, and closed tests, barrel predicate, bounded-family test, and Banach–Steinhaus consumer row before any barrelledness read is public.

Independence witness: locally-convex-space, seminormed-space, banachspace, banach-steinhaus: $\mathsf{BarrelledSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the locally convex source, finite seminorm family, four barrel tests, barrel predicate, bounded-family test, Banach–Steinhaus consumer row, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BarrelledSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarrelledSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7269.1 and Definition 7269.2 seed the finite barrelled-space route over locally convex, seminormed-space, Banach-space, Banach–Steinhaus, barrel-test, bounded-family, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BarrelledSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full Banach–Steinhaus, countable choice, Hahn–Banach strong forms, selected duals, quotient completions, Lean verification, or bridge checking for BarrelledSpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, barrel-test compatibility, bounded-family exactness, Banach–Steinhaus consumer routing, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.BarrelledSpaceUp.

Constructive story: A BarrelledSpace$^{\uparrow}$ packet is generated from locally convex source rows, finite seminorm-family rows, absorbing, balanced, convex, closed, barrel-predicate, bounded-family, Banach–Steinhaus consumer, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 7270

A Concrete Naming Certificate for $\mathsf{RealApartnessLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealApartnessLocator}^{\uparrow}$ packet records the finite located-apartness locator surface needed when Real and Completion consumers compare two displayed real names without asking for trichotomy. It sits after the apartness witness surface of chapter 304, the located-real boundary of **??**, the dyadic tolerance core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a locator interface for positive apartness evidence; it does not assert quotient real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7270.1 (Real apartness locator carrier)**.** *A* $\mathsf{RealApartnessLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, A, D, S, R, O, G, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *request row, exposing the two real-name sides and the located comparison request read by the consumer.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *witness row, carrying the positive apartness evidence made available to the locator.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that fixes the positive dyadic gap scale used by the comparison.*
4. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the two real names.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row computed from the displayed windows and dyadic tolerance.*
6. *O is the located-observation row that records which finite read of L, A, D, S, R produces the positive locator answer.*
7. *G is the positive gap row; it records the witnessed dyadic separation without turning the packet into an exhaustive order decision.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after L, A, D, S, R, O, G are displayed.*

9. *H stores componentwise* hsame *transport for the located request, apartness witness, dyadic tolerance, stream windows, regular readback, locator observation, gap, seal, replay, provenance, and naming rows.*
10. *C stores displayed* Cont *replay from a locator request through $L, A, D, S, R, O, G, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RealApartnessLocator}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessLocator}^{\uparrow}}$ *row.*

The carrier has no coordinate for total order trichotomy, quotient real equality, a selected representative of a real name, countable choice, an independent completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7270.2 (Real apartness locator NameCert obligations)**.** *For every accepted* $\mathsf{RealApartnessLocator}^{\uparrow}$ *carrier*

$$Q = (L, A, D, S, R, O, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealApartnessLocator}^{\uparrow}}$ *surface consists exactly of carrier admission, located-real request exposure, apartness-witness exposure, dyadic tolerance exactness, paired stream-window exposure, regular-rational readback, located-observation display, positive-gap display, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7270.1. The analytic rows are L, A, D, S, R, O, G, E, and the structural rows are H, C, P, N.

The located-real row L supplies only the comparison request. The apartness row A supplies positive evidence, while the dyadic row D fixes the tolerance at which that evidence is read.

The stream windows S and regular readback R make the inspected comparison finite. The observation row O and positive gap row G then display the locator answer without adding a global order branch.

Finally E, H, C, P, N route, transport, replay, source, and name the same finite packet. Since no excluded equality, representative choice, countable-choice schedule, trichotomy, independent completion object, or ambient-completeness coordinate appears in Q, the local certificate obligations are exhausted by the stated rows. □

Theorem 7270.3 (Real apartness locator window exactness)**.** *Every accepted* $\mathsf{RealApartnessLocator}^{\uparrow}$ *locator answer is read through the displayed finite window route*

$$L, A, D, S, R, O, G.$$

The $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used by O are exactly the rows recorded in the same carrier as the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and* $\mathsf{ApartnessReal}^{\uparrow}$ *witness. No consumer may replace them by a quotient real-name comparison, hidden representative, or external trichotomy test.*

Proof. Start from the NameCert obligations of Definition 7270.2. They require the located request, apartness witness, dyadic tolerance, stream windows, regular readback, observation, and gap rows to be displayed in one carrier.

The observation row O is downstream of L, A, D, S, R. Thus the window and readback consumed by the locator are the same S and R stored with the tolerance row D and apartness row A.

The positive gap row G records the finite result of that route. Because the carrier contains no separate equality or order oracle, a consumer cannot replace the route by quotient real comparison, a hidden representative, or trichotomy. □

Theorem 7270.4 (Real apartness locator Real-seal handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing consumer of an accepted* $\mathsf{RealApartnessLocator}^{\uparrow}$ *packet may read the seal-facing row* E *only after the located request, apartness witness, dyadic tolerance, stream windows, regular readback, observation row, and positive gap row are visible. The handoff therefore factors through*

$$L, A, D, S, R, O, G, E, H, C, P, N$$

before the consumer reaches the Real seal.

Proof. Use window exactness, Definition 7270.3. It places L, A, D, S, R, O, G before any locator answer is public.

The seal-facing row E is a downstream row of the same carrier. It is reached only after the located-apartness answer has been displayed, so the Real-facing read cannot skip the dyadic tolerance, window, readback, observation, or gap rows.

The structural rows H, C, P, N preserve the same route. They do not create a selected representative or a completion endpoint, and hence the Real-seal handoff remains a finite NameCert route rather than an assertion of $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7270.5 (Real apartness locator nonescape)**.** *No accepted* $\mathsf{RealApartnessLocator}^{\uparrow}$ *packet exports trichotomy, quotient real equality, selected representatives, countable choice, an independent completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exported read remains confined to the displayed rows* $L, A, D, S, R, O, G, E, H, C, P, N$ *and to their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure.*

Proof. Project the carrier. The only analytic rows are the located request, apartness witness, dyadic tolerance, stream windows, regular readback, observation, positive gap, and seal-facing row.

Apply the Real-seal handoff theorem. It shows that even Real-facing and Completion-facing consumers reach the endpoint only through the displayed locator route.

The remaining rows transport, replay, package, and name the same route. Since none of the refused objects is a carrier coordinate or a structural row, it cannot escape from an accepted packet. □

Lean4 (statement) `BEDC.Derived.RealApartnessLocatorUp`

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{RealApartnessLocator}^{\uparrow}$ packet is generated from a LocatedReal request row, an ApartnessReal witness row, a DyadicRatCore tolerance row, paired StreamName windows, paired RegSeqRat readback, a located observation row, a positive gap row, a Real seal-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealApartnessLocator}^{\uparrow}}$, seed)

Scope closed: Definition 7270.1, Definition 7270.2, Definition 7270.3, Definition 7270.4, and Definition 7270.5 seed the finite located-apartness locator route over LocatedReal, ApartnessReal, DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealApartnessLocator}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, independent completion objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-real request exposure, apartness-witness exposure, dyadic tolerance exactness, paired stream-window exposure, regular-rational readback, located-observation display, positive-gap display, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealApartnessLocatorUp.

Chapter 7271

A Concrete Naming Certificate for $\mathsf{KoopmanOperator}^{\uparrow}$

Following Koopman and von Neumann on operator-theoretic dynamics, the $\mathsf{KoopmanOperator}^{\uparrow}$ packet records the observable-side pullback of a displayed dynamical system as finite NameCert data. It sits after the routed flow carrier of chapter 195, the continuous-map graph surface of chapter 130, the transfer-operator and Perron–Frobenius density routes of chapter 456 and chapter 5123, the mean-ergodic averaging route of chapter 1840, and the finite StreamName-to-RegSeqRat-to-Real readback chain of chapter 126, chapter 301, and chapter 38. The packet names a finite observable pullback route; it is not a host Hilbert-space operator, an invariant measure theorem, a selected spectral decomposition, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7271.1 (Koopman operator carrier)**.** *A* $\mathsf{KoopmanOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, T, F, O, W, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{DynSystem}^{\uparrow}$ *state-evolution row, naming the displayed source states and the single-step or finite-step transition being inspected.*
2. *T is the* $\mathsf{ContinuousMap}^{\uparrow}$ *transition row attached to the same displayed dynamical system.*
3. *F is the observable row, recording the finite graph values of the displayed observable on the target of T.*
4. *O is the Koopman pullback row, whose displayed value at a source state is read as the observable row F after the transition row T.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row through which the pulled-back observable is inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to those finite windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, T, F, O, W, R have been displayed.*
8. *A is the averaging or transfer-neighbour row, recording which reads may be compared with* $\mathsf{TransferOperator}^{\uparrow}$, $\mathsf{PerronFrobenius}^{\uparrow}$, *and* $\mathsf{MeanErgodic}^{\uparrow}$ *consumers without identifying the packet with any one of them.*
9. *H records componentwise* hsame *transport for the dynamical-system, transition, observable, pullback, window, readback, seal, neighbour, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through*

$$X, T, F, O, W, R \longmapsto E$$

and through the neighbour row A when an averaging or density-facing consumer is present.

11. *P is the* Pkg *provenance over* $\mathsf{KoopmanOperator}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TransferOperator}^{\uparrow}$, $\mathsf{PerronFrobenius}^{\uparrow}$, $\mathsf{MeanErgodic}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{KoopmanOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, F, O, W, R, E, A, H, C, P, N$. It has no coordinate for an ambient function space, selected invariant measure, spectral theorem, quotient observable equality, host integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KoopmanOperatorUp.KoopmanOperatorTasteGate single carrier alignment`

Theorem 7271.2 (Koopman operator NameCert obligations)**.** *For every accepted* $\mathsf{KoopmanOperator}^{\uparrow}$ *carrier*

$$K = (X, T, F, O, W, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KoopmanOperator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dynamical-system row, continuous transition row, observable row, pullback row, observation windows, regular-rational readback, Real seal, neighbour route, transport, replay, provenance, and local naming;*
2. *pullback exactness, requiring each O read to be the displayed observable row F consumed after the transition row T on the same* $\mathsf{DynSystem}^{\uparrow}$ *state row X;*
3. *readback ordering, requiring W and R to be exposed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is available;*
4. *neighbour separation, requiring the* $\mathsf{TransferOperator}^{\uparrow}$, $\mathsf{PerronFrobenius}^{\uparrow}$, *and* $\mathsf{MeanErgodic}^{\uparrow}$ *rows in A to remain comparison or averaging consumers rather than replacements for the observable pullback row;*
5. *structural stability under the displayed* hsame, Cont, Pkg, *and local* NameCert *rows.*

No Koopman-operator obligation is stored outside these rows.

Proof. Project the accepted carrier. The tuple lists the dynamical-system row, transition row, observable row, pullback row, observation-window and readback rows, terminal seal, neighbour row, and structural rows.

Pullback exactness is local. Row O is admitted only as the observable row F read after the displayed transition T on the same state-evolution row X, so the packet has no coordinate for an ambient function-space equality.

Readback ordering then follows the StreamName and RegSeqRat rows. The finite window W is inspected before the regular-rational row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream of both.

Finally read the neighbour row A. It permits comparison with transfer, Perron–Frobenius, and mean-ergodic consumers while keeping the observable-side pullback distinct. Transport, replay, provenance, and local naming preserve that same finite route and add no invariant-measure theorem, spectral decomposition, host integration, or ambient completeness principle. □

Theorem 7271.3 (Koopman operator dynamics-observable route)**.** *Every observable-facing read of an accepted* $\mathsf{KoopmanOperator}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DynSystem}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer must expose the state-evolution row X, transition row T, observable row F, pullback row O, observation window W, and readback row R before the terminal seal E or any neighbour comparison row A is consumed. The packet exports no invariant measure, density pushforward, selected spectral datum, quotient observable equality, host integration theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7271.2. They place every accepted consumer read inside the displayed carrier rows $X, T, F, O, W, R, E, A, H, C, P, N$.

Read the dynamics-to-observable route first. The $\mathsf{DynSystem}^{\uparrow}$ row X and $\mathsf{ContinuousMap}^{\uparrow}$ row T determine where the observable is sampled, and the row O records that the observable value is pulled back from F along that transition.

The numerical readback is downstream. The finite $\mathsf{StreamName}^{\uparrow}$ window W feeds the $\mathsf{RegSeqRat}^{\uparrow}$ row R, and only then can the terminal $\mathsf{Real}^{\uparrow}$ seal E be read.

The neighbour row A cannot reverse the route. It may point to transfer, Perron–Frobenius, or mean-ergodic consumers, but those rows do not replace X, T, F, O, W, R, E by a measure-theoretic, spectral, quotient, or host-completeness object. Therefore the displayed dynamics-observable route exhausts the accepted packet. □

Falsifiable prediction: A $\mathsf{KoopmanOperator}^{\uparrow}$ consumer that reads a RealUp seal or an averaging neighbour without displaying the DynSystem transition, ContinuousMap row, observable pullback row, StreamName window, and RegSeqRat readback refutes the carrier of Definition 7271.1.

Independence witness: dynsystem, continuousmap, transferoperator, perronfrobenius, mean-ergodic, streamname, regseqrat, real: $\mathsf{KoopmanOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state evolution, transition, observable pullback, finite observation windows, regular-rational readback, Real sealing, neighbour separation, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KoopmanOperator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KoopmanOperator}^{\uparrow}$ packet is generated from DynSystem state-evolution rows, ContinuousMap transition rows, observable pullback rows, StreamName windows, RegSeqRat readback, a RealUp seal, transfer and mean-ergodic neighbour rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KoopmanOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7271.1, Definition 7271.2, and Definition 7271.3 seed the finite observable-side dynamics route over DynSystem, ContinuousMap, StreamName, RegSeqRat, Real, transfer, Perron–Frobenius, and mean-ergodic rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KoopmanOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close invariant measures, density pushforwards, spectral theory, quotient observable equality, host integration, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pullback exactness, readback ordering, neighbour separation, transport, replay, provenance, and local naming for BEDC.Derived.KoopmanOperatorUp.

Chapter 7272

A Concrete Naming Certificate for $\mathsf{LocatedUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures, and Bishop–Bridges on located metric comparison and regular real names, the $\mathsf{LocatedUniformity}^{\uparrow}$ packet records the finite interface where metric locatedness is read as a uniform-entourage basis before any Cauchy-filter, stream-name, regular-rational, or real-seal consumer may use it. It sits between the ambient uniform-space surface of chapter 339, the real-entourage route of chapter 1624, the regular-Cauchy entourage route of chapter 1628, the Cauchy filter boundary of chapter 333, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located compact consumer surface of chapter 1488. The packet names a located uniformity as displayed finite data; it does not select a completion point, quotient filters, import an open-cover compactness principle, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7272.1 (Located uniformity carrier)**.** *A* $\mathsf{LocatedUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, A, E, F, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, naming the* $\mathsf{Met}^{\uparrow}$ *or real-facing metric source whose entourage reads are admitted by the packet.*
2. *A is the located comparison row, exposing finite inside-or-apart comparison data for point-pairs and positive rational radii.*
3. *E is the entourage-basis row, recording the displayed basis used by* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{RealUniformStructure}^{\uparrow}$*, and* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *consumers.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, available only after M, A, E have been displayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by filter and completion consumers.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the windows named by W.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located comparison, entourage basis, Cauchy-filter handoff, stream windows, and regular rational readback are visible.*
8. *H records componentwise* hsame *transport for $M, A, E, F, W, Q, R, C, P, N$.*

9. *C records displayed* Cont *replay from a metric locatedness request through the entourage basis, Cauchy-filter handoff, StreamName windows, RegSeqRat readback, Real seal, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a hidden completion point, quotient filter equality, host uniform-space equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7272.2 (Located uniformity NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformity}^{\uparrow}$ *carrier*

$$L = (M, A, E, F, W, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedUniformity}^{\uparrow}}$ *surface consists exactly of metric source admission, located comparison admission, entourage-basis admission, Cauchy-filter handoff, StreamName window admission, RegSeqRat readback, Real seal nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.*

Proof. Project the carrier of Definition 7272.1. The analytic rows are exactly M, A, E, F, W, Q, R, and the structural rows are H, C, P, N.

The metric source and located comparison rows are read first. They expose the finite point-pair and rational-radius comparisons from which the entourage basis row E may be inspected by $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, and $\mathsf{RegularCauchyUniformity}^{\uparrow}$ consumers.

The Cauchy-filter and completion-facing rows are downstream. A consumer reaches F only after the located entourage basis is visible, then reads W as finite windows, Q as regular rational readback, and R as the terminal real seal.

Finally H, C, P, N transport, replay, package, and name the same route. Since no excluded host object is a carrier coordinate, these rows exhaust the local NameCert obligations. □

Theorem 7272.3 (Located uniformity Cauchy-filter handoff)**.** *Every Cauchy-filter or completion consumer of an accepted* $\mathsf{LocatedUniformity}^{\uparrow}$ *carrier receives the located uniformity only through the finite route*

$$M, A, E, F, W, Q, R, H, C, P, N.$$

In particular, the $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row is consumed after the metric located comparison and entourage basis rows, and before the StreamName, RegSeqRat, and Real-seal rows. The handoff exports no quotient filter equality, selected completion point, countable-choice schedule, or ambient completeness principle.*

Proof. Begin with a Cauchy-filter or completion read of the accepted packet. The carrier first exposes the metric source M, the located comparison row A, and the entourage-basis row E.

The handoff row F is the only Cauchy-filter coordinate of the carrier. The consumer may inspect F only along the route already displaying M, A, E.

The read then proceeds through W, Q, R: finite stream windows, regular rational readback, and the terminal real seal. Componentwise transport and continuation replay preserve the same order, while P and N record provenance and local naming.

Because no quotient filter, selected completion point, choice schedule, or ambient completeness coordinate occurs in the packet, the consumer receives exactly the displayed finite handoff. □

Theorem 7272.4 (Located uniformity Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedUniformity}^{\uparrow}$ *carrier factors through the displayed located-uniformity route*

$$M, A, E, F, W, Q, R, H, C, P, N.$$

Thus the terminal seal row R *is reached only after metric source admission, located comparison, entourage-basis admission, Cauchy-filter handoff, finite StreamName windows, and RegSeqRat readback. It does not export host real equality, quotient stream equality, quotient filter equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7272.2 to identify the complete local obligation surface. The route to the real seal contains the source, comparison, entourage, filter, window, and rational-readback rows before R.

Next apply the handoff theorem Definition 7272.3. It fixes the order in which Cauchy-filter and completion consumers may read the packet, so the terminal real seal cannot be reached from an unlisted source.

The structural rows preserve this route without adding host data: H transports the displayed coordinates, C replays the same finite path, P records only the named sibling packets, and N names the local certificate. Hence the real-facing read is a finite seal boundary, not a quotient, selection, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedUniformity}^{\uparrow}$ packet is generated from a metric source row, located comparison row, entourage-basis row, CauchyFilter handoff row, StreamName finite windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7272.1, Definition 7272.2, Definition 7272.3, and Definition 7272.4 seed the located-uniformity interface from metric locatedness to uniform, Cauchy-filter, StreamName, RegSeqRat, and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected completion points, open-cover compactness, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the located comparison, Cauchy-filter handoff, Real seal nonescape, transport, replay, provenance, and local naming obligations.

Chapter 7273

A Concrete Naming Certificate for $\mathsf{DyadicIntervalBisection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicIntervalBisection}^{\uparrow}$ packet records the finite act of bisecting a dyadic interval before any infinite branch or real-completion argument is available. It sits between the dyadic midpoint arithmetic of chapter 445, the arithmetic carrier of chapter 291, the finite located-real interval route of chapter 5546, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite bisection ledger; it does not select an infinite branch, construct a Dedekind cut, assert trichotomy, quotient reals, or close ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7273.1 (Dyadic interval bisection carrier)**.** *A* $\mathsf{DyadicIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, R, M, S, W, J, H, C, P, N)$$

with the following displayed rows:

1. *L is the left dyadic endpoint row.*
2. *R is the right dyadic endpoint row.*
3. *M is the midpoint row consumed from the dyadic midpoint certificate and checked in* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *S is the branch-choice row, recording whether the finite bisection retains the left half, the right half, or both halves as named subintervals.*
5. *W is the width-budget row, recording the dyadic width before and after the bisection.*
6. *J is the nestedness ledger, recording that every retained subinterval has endpoints between L and R and is read as a subinterval of the source packet.*
7. *H is the componentwise* hsame *transport row over L, R, M, S, W, J.*
8. *C is the displayed* Cont *replay row for the finite bisection step.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalBisection}^{\uparrow}$*, the dyadic midpoint certificate of chapter 445,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalBisection}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite branch selector, Dedekind cut, real trichotomy, quotient-real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Theorem 7273.2 (Dyadic interval bisection NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalBisection}^{\uparrow}$ *carrier* $B = (L, R, M, S, W, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalBisection}^{\uparrow}}$ *surface consists exactly of endpoint exposure through* L, R*, midpoint exposure through* M*, branch-choice exposure through* S*, width-budget exposure through* W*, nestedness exposure through* J*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every downstream finite located-interval consumer must read those rows before it can hand bisection data to $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalBisectionUp.DyadicIntervalBisectionNamecertObligations`

Proof. Project the carrier of Definition 7273.1. Its rows are exactly $L, R, M, S, W, J, H, C, P, N$, so endpoint, midpoint, branch, width, and nestedness data are all displayed before any consumer can read the bisection.

The midpoint row is checked against the dyadic midpoint certificate and $\mathsf{DyadicRatCore}^{\uparrow}$, while S and J record which finite subintervals are retained. The transport row H and replay row C preserve those finite rows componentwise.

Finally read the provenance and local naming rows P, N. They identify the route into $\mathsf{LocatedRealIntervalBisectio}$ $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ without adding an infinite selector, cut object, quotient equality, trichotomy decision, or completeness witness. Hence the displayed rows exhaust the local NameCert obligations. □

Theorem 7273.3 (Dyadic interval bisection nested width)**.** *Every retained subinterval of an accepted* $\mathsf{DyadicIntervalBisection}^{\uparrow}$ *packet is nested inside the source interval and has dyadic width bounded by the width-budget row* W*. In the exact bisection case, the retained width row is read as one half of the source width through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint calculation.*

Proof. Use the endpoint rows L, R and midpoint row M from Definition 7273.1. The midpoint is a dyadic rational row, so the two candidate subintervals have dyadic endpoints already visible in the packet.

Read the branch-choice row S together with the nestedness ledger J. These rows say that a retained left or right half uses the source endpoint and the midpoint, hence every retained endpoint lies between the original endpoints.

The width-budget row W records the source width and the retained width. Since the midpoint row is checked in $\mathsf{DyadicRatCore}^{\uparrow}$, the exact bisection case reads the retained width as half the source width, while the structural rows H, C, P, N preserve and name that finite calculation for downstream consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalBisection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicIntervalBisection}^{\uparrow}$ packet is generated from left and right dyadic endpoint rows, a dyadic midpoint row, a branch-choice row, a width-budget row, a nestedness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7273.1, Definition 7273.2, and Definition 7273.3 seed the finite dyadic bisection carrier, local NameCert surface, and nested-width ledger used by located real interval and regular rational readback packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, Dedekind cuts, trichotomy, quotient reals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic midpoint exactness, branch-row stability, nested-width preservation, readback handoff, and nonescape for BEDC.Derived.DyadicIntervalBisectionUp.

Chapter 7274

A Concrete Naming Certificate for $\mathsf{BeltramiKleinDisk}^{\uparrow}$

Following Beltrami, Klein, and the projective disk model of hyperbolic geometry, the $\mathsf{BeltramiKleinDisk}^{\uparrow}$ packet records a chord-geodesic disk reading as finite NameCert data. It sits beside the disk carrier of chapter 377, the metric surfaces of chapter 6209 and chapter 3474, the boundary-geodesic route of chapter 3491, and the terminal real-name seal of chapter 38. The packet names the projective comparison surface of the disk model; it is not a completed hyperbolic plane, quotient ideal boundary, host projective space, selected geodesic family, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7274.1 (Beltrami–Klein disk carrier)**.** *A* $\mathsf{BeltramiKleinDisk}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, P, K, G, M, Q, R, H, C, S, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the displayed interior disk points and boundary-facing observations.*
2. *P is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *comparison row on the same displayed disk observations.*
3. *K is the projective chord row, recording the finite chord segment through which the Beltrami–Klein reading treats geodesics as straight disk chords.*
4. *G is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary-incidence row attached to the displayed chord endpoints.*
5. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row used to compare the chord reading with the Poincare-disk metric surface.*
6. *Q is the refusal row excluding a completed projective plane, quotient ideal-boundary equality, selected global geodesics, host cross-ratio semantics, and physical observer trajectories.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed metric or incidence comparison.*
8. *H records componentwise* hsame *transport for disk coordinates, Poincare comparison, chord rows, boundary incidence, metric reads, refusal rows, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through*

$$D, P, K, G, M, Q \longmapsto R.$$

10. *S is the* Pkg *provenance over* $\mathsf{BeltramiKleinDisk}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BeltramiKleinDisk}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, P, K, G, M, Q, R, H, C, S, N$. It has no coordinate for a completed projective model, quotient boundary object, selected infinite geodesic family, host analytic cross-ratio, physical observer law, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7274.2 (Beltrami–Klein disk NameCert obligations)**.** *For every accepted* $\mathsf{BeltramiKleinDisk}^{\uparrow}$ *carrier*

$$B = (D, P, K, G, M, Q, R, H, C, S, N),$$

the local $\mathsf{NameCert}_{\mathsf{BeltramiKleinDisk}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for disk coordinates, Poincare metric comparison, projective chord rows, geodesic-boundary incidence, hyperbolic metric reads, refusal rows, Real seals, transport, replay, provenance, and local naming;*
2. *chord-geodesic exactness, requiring K to be read only over the displayed disk row D and boundary-incidence row G;*
3. *Poincare comparison exactness, requiring the Poincare metric row P and hyperbolic metric row M to be displayed before any Real-facing comparison R is consumed;*
4. *refusal preservation, requiring Q to stay visible whenever a consumer asks for completed projective geometry, quotient boundary equality, selected global geodesics, host cross-ratio semantics, or physical observer trajectories;*
5. *structural stability under H, C, S, N.*

No Beltrami–Klein disk obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 7274.1. The packet exposes the disk row D, Poincare comparison row P, chord row K, boundary-incidence row G, hyperbolic metric row M, refusal row Q, Real seal R, and structural rows H, C, S, N.

Chord-geodesic exactness is finite. The row K can be read only as a displayed chord over the disk observations in D and the boundary-incidence evidence in G, so no selected global geodesic family or host projective plane enters the packet.

The comparison with the Poincare disk model is downstream of the metric rows. Rows P and M must both be visible before R is read as a Real-facing metric or incidence comparison.

Finally the refusal row remains attached under transport and replay. The structural rows preserve the same finite route and add no completed hyperbolic plane, quotient ideal boundary, host cross-ratio semantics, physical observer law, or ambient completeness principle. □

Theorem 7274.3 (Beltrami–Klein disk Poincare comparison route)**.** *Every accepted* $\mathsf{BeltramiKleinDisk}^{\uparrow}$ *comparison with the Poincare disk model factors through the finite route*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicGeodesicBoundary}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer must expose disk coordinates D, Poincare metric comparison P, chord row K, geodesic-boundary incidence G, hyperbolic metric read M, and refusal row Q before any terminal Real seal R is consumed. The packet exports no completed projective model, quotient ideal-boundary equality, selected global geodesics, host analytic cross-ratio, physical observer trajectory, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7274.2. They keep every accepted comparison inside the displayed rows $D, P, K, G, M, Q, R, H, C, S, N$.

Read the disk-model data first. The $\mathsf{UnitDisk}^{\uparrow}$ row D supplies the displayed interior and boundary-facing observations, while K records the projective chord only over that finite disk surface.

The comparison rows then attach the Poincare and hyperbolic metric readings. Row P supplies the Poincare-disk metric surface, row M supplies the hyperbolic metric read, and row G keeps boundary incidence visible before the Real seal R is inspected.

The refusal row Q blocks the common overreadings of the projective model. Transport, replay, provenance, and local naming preserve the displayed route, so the comparison remains a finite disk-model packet rather than a completed projective, quotient-boundary, host analytic, physical, or completeness claim. □

Falsifiable prediction: A $\mathsf{BeltramiKleinDisk}^{\uparrow}$ consumer that reaches a RealUp comparison without displaying UnitDisk coordinates, the PoincareDiskMetric row, the chord-geodesic row, HyperbolicGeodesicBoundary incidence, HyperbolicMetric comparison, and the refusal row refutes the carrier of Definition 7274.1.

Independence witness: unitdisk, poincare-disk-metric, hyperbolic-metric, hyperbolic-geodesic-boundary, real: $\mathsf{BeltramiKleinDisk}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk coordinates, projective chord rows, Poincare comparison, hyperbolic metric reads, boundary incidence, refusal, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BeltramiKleinDisk}^{\uparrow}}$.

Constructive story: A $\mathsf{BeltramiKleinDisk}^{\uparrow}$ packet is generated from UnitDisk coordinates, PoincareDiskMetric comparison rows, projective chord rows, HyperbolicGeodesicBoundary incidence, HyperbolicMetric reads, refusal rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BeltramiKleinDisk}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7274.1, Definition 7274.2, and Definition 7274.3 seed the finite projective disk-model comparison route over UnitDisk, PoincareDiskMetric, HyperbolicMetric, HyperbolicGeodesicBoundary, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BeltramiKleinDisk}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed projective disk geometry, quotient ideal-boundary equality, selected global geodesics, host analytic cross-ratio semantics, physical observer trajectories, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chord-geodesic exactness, Poincare comparison exactness, boundary-incidence exposure, refusal preservation, transport, replay, provenance, and local naming for BEDC.Derived.BeltramiKleinDiskUp.

Chapter 7275

A Concrete Naming Certificate for $\mathsf{MetricRetraction}^{\uparrow}$

Following Borsuk and Dugundji on retractions and constructive metric topology, the $\mathsf{MetricRetraction}^{\uparrow}$ packet records the finite BEDC surface for an idempotent retraction on a displayed metric carrier. It sits after the metric source of chapter 57, the continuous-map route of chapter 130, the complete-service surface of chapter 128, the terminal real comparison seal of chapter 38, and the stronger Lipschitz retraction packet of chapter 5915. The packet names the general idempotent continuous metric retraction route; it does not assert a Lipschitz constant, selected nearest point, quotient subspace equality, ambient completeness of the source, or host fixed-point theorem.

Definition 7275.1 (Metric retraction carrier)**.** *A* $\mathsf{MetricRetraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, G, I, T, M, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row supplying the finite distance witnesses used by the retraction request.*
2. *A is the displayed subcarrier row, recording membership, inclusion into X, and the inherited distance reads inspected by the packet.*
3. *G is the retraction graph row, whose source observations lie in X and whose values are admitted by A.*
4. *I is the identity-on-subcarrier row, requiring every displayed point already admitted by A to replay through G as the same subcarrier point.*
5. *T is the idempotence row, recording that a second read through the same graph contributes no new value row after the first retraction read.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row available only as a downstream consumer service for the displayed subcarrier route.*
7. *E is the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row tying the graph G to the finite continuity route used by metric consumers.*
8. *H records componentwise* hsame *transport for the metric source, subcarrier, graph, identity, idempotence, complete-service, continuous-map replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a retraction request through $X, A, G, I, T, M, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricRetraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LipschitzRetraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricRetraction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a Lipschitz constant, selected best approximation, quotient subspace equality, ambient source completeness theorem, host real equality, or global fixed-point theorem.

Theorem 7275.2 (Metric retraction NameCert obligations)**.** *For every accepted* $\mathsf{MetricRetraction}^{\uparrow}$ *carrier $R = (X, A, G, I, T, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricRetraction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, metric-source visibility through X, subcarrier admission and inherited-distance visibility through A, graph admission through G, identity-on-subcarrier replay through I, idempotent replay through T, complete-metric service access through M, continuous-map replay through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7275.1. The displayed tuple lists exactly the metric source, subcarrier, graph, identity, idempotence, complete-service, continuous-map, transport, replay, provenance, and naming rows.

The source-facing obligations are exhausted by X, A, G, I, T. The metric row supplies distance witnesses, the subcarrier row supplies admissible values, the graph row supplies finite retraction observations, the identity row fixes points already in the subcarrier, and the idempotence row prevents a repeated graph read from adding a second value.

The consumer-facing obligations are exhausted by M, E. A complete-space consumer can inspect only the displayed $\mathsf{CompleteMetric}^{\uparrow}$ service row, and a continuity consumer can inspect only the $\mathsf{ContinuousMap}^{\uparrow}$ replay attached to the same graph.

Finally H, C, P, N transport, replay, package, and name this same finite route. Since the carrier has no Lipschitz constant, selected nearest point, quotient equality, ambient source-completeness theorem, host real equality, or global fixed-point coordinate, no such obligation is present in the local certificate. □

Theorem 7275.3 (Metric retraction idempotent handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{MetricRetraction}^{\uparrow}$ *carrier factors through the finite route*

$$X, A \longmapsto G, I, T \longmapsto E, M \longmapsto H, C, P, N.$$

Thus source distances are read in the metric row X, subcarrier values are admitted through A, the retraction graph and identity rows are checked before any continuous-map or complete-service handoff, and idempotence is recorded by the displayed row T. The handoff is the general metric retraction surface; the stronger $\mathsf{LipschitzRetraction}^{\uparrow}$ *packet may consume it, but this packet exports no Lipschitz constant, selected projection, quotient subspace equality, or ambient completeness claim.*

Proof. Start with the NameCert obligations of Definition 7275.2. Any consumer first receives the metric source row X and the subcarrier row A, since these are the only rows that can supply source distances and admitted retraction values.

The retraction data then pass through G, I, T. The graph row records finite observations, the identity row verifies that subcarrier points replay as themselves, and the idempotence row records that replaying the graph again stays on the same displayed subcarrier value.

Only after those rows are displayed can a consumer read E, M. The continuous-map row is induced by the same graph route, and the complete-metric service row is a displayed downstream service rather than an ambient completeness theorem for the source.

Transport, replay, provenance, and local naming are then carried by H, C, P, N. These rows preserve the factorization but add no Lipschitz constant, selected projection, quotient equality, host equality, or classical branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricRetraction}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricRetraction}^{\uparrow}$ packet is generated from a Metric source row, a displayed subcarrier row, a retraction graph, an identity-on-subcarrier row, an idempotence row, a CompleteMetric service row, a ContinuousMap replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricRetraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7275.1, Definition 7275.2, and Definition 7275.3 seed the finite metric retraction route over source, subcarrier, graph, identity, idempotence, continuous-map replay, and complete-service rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricRetraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lipschitz constants, selected nearest points, quotient subspace equality, ambient source completeness, global fixed-point existence, Lean verification, bridge checking, or host real equality.

Upgrade path: Obligation closure requires checked carrier admission, metric-source visibility, subcarrier admission, graph admission, identity replay, idempotent replay, continuous-map replay, complete-metric service access, transport, replay, provenance, and local naming for BEDC.Derived.MetricRetractionUp.

Chapter 7276

A Concrete Naming Certificate for $\mathsf{DyadicCauchyApartness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicCauchyApartness}^{\uparrow}$ packet records a positive separation witness between regular Cauchy real names as finite BEDC NameCert data. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located real apartness boundary of chapter 6450. The packet supplies a constructive apartness boundary for Real and Completion consumers without importing quotient stream equality, trichotomy, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7276.1 (Dyadic Cauchy apartness carrier)**.** *A* $\mathsf{DyadicCauchyApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, W_X, W_Y, D, R_X, R_Y, A, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy name rows whose observations are read only through displayed finite windows.*
2. *W_X and W_Y are the* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected for the two names at the separating precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-bound row; it records the dyadic radius and strict positive gap used by the apartness witness.*
4. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the two stream windows and the dyadic tolerance ledger.*
5. *A is the* $\mathsf{LocatedRealApartness}^{\uparrow}$ *handoff row reading the displayed positive gap as located apartness evidence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the two readbacks and the positive dyadic gap have been displayed.*
7. *H records componentwise* hsame *transport for the two names, windows, dyadic gap, readbacks, apartness handoff, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a precision request through $W_X, W_Y, D, R_X, R_Y, A, E$.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyApartness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyApartness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for quotient streams, a selected limit, real trichotomy, host real equality, or an ambient completeness principle.

Theorem 7276.2 (Dyadic Cauchy apartness NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyApartness}^{\uparrow}$ *carrier*

$$K = (X, Y, W_X, W_Y, D, R_X, R_Y, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy names, paired* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-gap exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback stability,* $\mathsf{LocatedRealApartness}^{\uparrow}$ *handoff compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7276.1. The analytic rows are the two names X, Y, the two finite windows W_X, W_Y, the positive dyadic row D, the two readbacks R_X, R_Y, the apartness handoff A, and the Real seal E.

The positive witness is forced through D. Since D is the only row carrying a strict dyadic gap, no apartness consumer can read a separation assertion before the finite windows and dyadic positive-bound data have been displayed.

The regular readbacks R_X, R_Y attach the two inspected windows to the regular rational source surface, and A reads exactly that displayed gap as located real apartness evidence. The Real seal E is downstream of those rows rather than a source of equality or trichotomy.

Finally H, C, P, N transport, replay, source, and name the same route. No further coordinate is present in the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 7276.3 (Dyadic Cauchy apartness located-comparison boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicCauchyApartness}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LocatedRealApartness}^{\uparrow} \longmapsto \mathsf{Real}$$

The consumer may use the displayed positive dyadic gap as located apartness evidence, but it cannot extract quotient stream equality, trichotomy, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Begin with a Real or Completion consumer of the accepted packet. The carrier exposes no comparison row before the two finite stream windows and the positive dyadic ledger have been read.

The consumer then obtains R_X and R_Y from the displayed windows. These rows keep the comparison attached to regular rational readback rather than to a quotient of completed streams.

Next the row A turns the same positive dyadic separation into located apartness evidence, and the row E seals the route for Real-facing consumers. The seal is therefore reached only after the window, gap, and readback rows have been displayed.

The structural rows H, C, P, N preserve this ordering under transport, replay, provenance, and local naming. Since the carrier has no row for the excluded host objects, the consumer receives only the stated finite located-comparison boundary. □

Theorem 7276.4 (Dyadic Cauchy apartness window induction)**.** *Let*

$$K = (X, Y, W_X, W_Y, D, R_X, R_Y, A, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicCauchyApartness}^{\uparrow}$ *carrier. If the paired finite windows* W_X, W_Y *are extended by one displayed* $\mathsf{StreamName}^{\uparrow}$ *observation and the positive dyadic gap row* D *transports through the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_X, R_Y*, then the same* $\mathsf{DyadicCauchyApartness}^{\uparrow}$ *packet remains accepted and the located-apartness handoff still reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

Proof. Proceed by finite induction on the displayed paired window extension. The base case is the accepted carrier itself, where $W_X, W_Y, D, R_X, R_Y, A, E$ already expose the positive dyadic gap, the two regular readbacks, the located-apartness handoff, and the Real seal.

For the successor step, append one displayed $\mathsf{StreamName}^{\uparrow}$ observation to each of W_X and W_Y. The transported positive dyadic gap remains the row D, and its two readbacks remain the corresponding R_X and R_Y rows after the window extension.

The handoff row A reads that same transported positive gap as located apartness evidence. Since the extension changes only displayed finite windows and their readbacks, A still feeds the terminal seal row E.

Finally H, C, P, N transport, replay, source, and name the enlarged finite route. Thus the accepted packet is stable under the displayed one-step window extension, and the Real-facing consumer still receives the located-apartness handoff only through E. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyApartness}^{\uparrow}$ packet is generated from two regular Cauchy name rows, paired StreamName windows, a DyadicRatCore positive-gap row, RegSeqRat readbacks, a LocatedRealApartness handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicCauchyApartness}^{\uparrow}}$, seed)

Scope closed: Definition 7276.1, Definition 7276.2, Definition 7276.3, and Definition 7276.4 seed the finite constructive apartness boundary over dyadic positive gaps, stream windows, regular rational readbacks, located apartness, window extension, and Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicCauchyApartness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, trichotomy, selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped treatment asks for checked carrier admission, dyadic positive-gap exactness, paired window readback, located-apartness compatibility, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 7277

A Concrete Naming Certificate for $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$

Following Bishop and Bridges on separated constructive completions, the $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ packet records the finite L10 surface where a located Cauchy-filter completion comparison is allowed to read uniqueness only after the separated completion route, completion uniqueness ledger, stream windows, regular rational readback, dyadic tolerance data, and terminal real seal have all been displayed. It sits over the located filter boundary of chapter 1661, the separated-completion boundary of chapter 1612, the completion-uniqueness ledger of chapter 1463, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the located uniqueness boundary; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient completion equality, selected limits, or host equality of completed points.

Definition 7277.1 (Located completion uniqueness carrier)**.** *A* $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, S, Q, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *source row, exposing the located tail and basis-refinement data that any completion comparison must first read.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row, recording that equality-facing completion reads pass through separated zero-distance evidence rather than host equality.*
3. *Q is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison ledger for the two located completion presentations over the same displayed source route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the comparison ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those windows.*
6. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two located tails at the requested precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, S, Q, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for $F, S, Q, W, R, D, E, C, P, N$.*

9. *C records displayed* Cont *replay from a located completion uniqueness request back through the same filter, separated-completion, completion-uniqueness, window, readback, tolerance, and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7277.2 (Located completion uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ *carrier*

$$U = (F, S, Q, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompletionUniqueness}^{\uparrow}}$ *surface consists exactly of located Cauchy-filter source admission, separated-completion boundary admission, completion-uniqueness ledger exposure, StreamName window admission, RegSeqRat readback, DyadicRatCore tolerance comparison, Real seal nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.*

Proof. Project the carrier of Definition 7277.1. The analytic rows are exactly F, S, Q, W, R, D, E, and the structural rows are H, C, P, N.

The located source and separated boundary rows are read first. Row F exposes the located filter data, while row S forces equality-facing completion reads to pass through the separated completion boundary.

The uniqueness comparison is then a displayed finite ledger. Row Q records the completion-uniqueness comparison, and rows W, R, D provide the stream windows, regular rational readback, and dyadic tolerance data consumed by that ledger.

Only after those rows are visible may the route reach the real seal E. The rows H, C, P, N transport, replay, source, and name the same route, so the local certificate cannot export quotient completion equality, selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 7277.3 (Located completion uniqueness Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{LocatedCauchyFilter}^{\uparrow},\ \mathsf{SeparatedCompletion}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow}$$

Thus located completion uniqueness is available only as a displayed NameCert route through located filter data, separated completion, the completion uniqueness ledger, finite windows, regular readback, and dyadic tolerances. The packet exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient completion equality, selected limit, host equality of completed points, or countable-choice schedule.*

Proof. Use Definition 7277.2 to identify the complete local obligation surface. A real-facing read must first consume F, S, Q, W, R, D before the seal row E is available.

The separated and uniqueness rows are both visible. Row S supplies the separated completion boundary, and row Q supplies the completion uniqueness ledger over the same located route rather than replacing it by a quotient object.

The window, readback, and tolerance rows make the comparison finite. Rows W, R, D expose the StreamName, RegSeqRat, and DyadicRatCore data used by the ledger, after which E is only a terminal seal.

Finally H, C, P, N preserve that route without adding host data. Since no excluded object is a carrier coordinate, every exported real-facing read remains inside the displayed located completion uniqueness boundary. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ uniqueness read must expose a $\mathsf{LocatedCauchyFilter}^{\uparrow}$ source row, a $\mathsf{SeparatedCompletion}^{\uparrow}$ boundary row, a $\mathsf{CompletionUniqueness}^{\uparrow}$ ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: locatedcauchyfilter, separatedcompletion, completionuniqueness, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located filter source, separated completion boundary, completion uniqueness ledger, finite windows, regular readback, dyadic tolerance comparison, Real seal, transport, replay, provenance, and local naming while refusing quotient completion equality and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$ packet is generated from a LocatedCauchyFilter row, SeparatedCompletion boundary, CompletionUniqueness ledger, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7277.1, Definition 7277.2, and Definition 7277.3 seed the located uniqueness boundary over located Cauchy filters, separated completions, completion uniqueness, finite stream windows, regular rational readback, dyadic tolerances, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient completion equality, selected limits, host equality of completed points, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located filter exposure, separated completion boundary, completion uniqueness ledger exactness, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparison, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 7278

A Concrete Naming Certificate for $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the uniform-completion practice of Bourbaki and Kelley, the $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ packet records the finite unit-map modulus route used by L10 completion consumers. It sits over the uniform-completion carrier of chapter 1473, the completion universal-property route of chapter 397, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38. The packet names the displayed modulus by which the unit map enters completion; it does not quotient Cauchy names, select representatives, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7278.1 (Uniform completion unit modulus carrier)**.** *A* $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, I, M, D, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source uniform-space row whose unit map is being read by a completion consumer.*
2. *I is the unit-map row from X into the displayed* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary.*
3. *M is the modulus ledger assigning each requested uniform tolerance a finite dyadic control row.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the ledger M.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by M and checked against D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed window and tolerance data.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R is visible.*
8. *H records componentwise* hsame *transport for $X, I, M, D, W, R, S, C, P, N$.*
9. *C records displayed* Cont *replay from a unit-map request through the modulus ledger, dyadic tolerance, finite window, readback, seal, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, CompletionUniversalProperty $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionUnitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient completion equality, selected representatives, host uniform-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7278.2 (Uniform completion unit modulus NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ *carrier*

$$U = (X, I, M, D, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCompletionUnitModulus}^{\uparrow}}$ *surface consists exactly of source uniform admission, unit-map admission, modulus-ledger exactness, dyadic tolerance admission, StreamName window admission, RegSeqRat readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7278.1. The analytic rows are X, I, M, D, W, R, S, and the structural rows are H, C, P, N.

The source row X and unit-map row I place the read inside the $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ corridor. The modulus row M is the only coordinate that answers a tolerance request, and it answers only by displaying the dyadic row D and the finite window row W.

The readback row R is downstream of M, D, W. Thus a completion consumer may read regular rational data only after the finite tolerance and window ledger is visible.

Finally the seal row S is downstream of R, while H, C, P, N transport, replay, package, and name the same route. Since no excluded host object appears as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7278.3 (Uniform completion unit modulus RegSeqRat route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ *carrier factors through the finite route*

$$X, I, M, D, W, R, H, C, P, N.$$

The route first displays the unit map, the modulus ledger, the dyadic tolerance row, and the StreamName window row; it exports no quotient stream equality, selected Cauchy representative, countable-choice schedule, or host equality of completion points.

Proof. Use Definition 7278.2 to identify the accepted obligation surface. The only row with regular rational readback content is R.

The rows M, D, W are ordered before R by the carrier itself. They provide the displayed modulus, dyadic tolerance, and finite window needed for the readback, so a consumer cannot reach R from an unlisted stream coordinate or selected representative.

The structural rows H, C, P, N preserve that same route for transport, replay, provenance, and naming. Hence the RegSeqRat-facing read is exactly the displayed finite unit-modulus route. □

Theorem 7278.4 (Uniform completion unit modulus Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ *carrier factors through the finite route*

$$X, I, M, D, W, R, S, H, C, P, N.$$

Thus the Real seal is reached only after the unit map, modulus ledger, DyadicRatCore tolerance, StreamName window, and RegSeqRat readback rows are visible. It does not export host real equality, quotient completion equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the RegSeqRat route of Definition 7278.3. It places the readback row R after the displayed unit-map, modulus, tolerance, and window rows.

The seal row S is the only Real-facing coordinate in the carrier, and it is downstream of R. Therefore the Real read cannot bypass the finite unit-modulus ledger.

Componentwise transport, continuation replay, provenance, and local naming preserve the same finite chain. Since the carrier has no row for the excluded objects, the Real-facing endpoint is a seal boundary, not an ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionUnitModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionUnitModulus}^{\uparrow}$ packet is generated from a source uniform-space row, unit-map row, modulus ledger, DyadicRatCore tolerance row, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionUnitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7278.1, Definition 7278.2, Definition 7278.3, and Definition 7278.4 seed the finite unit-map modulus route from uniform completion to StreamName, RegSeqRat, and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionUnitModulus}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host uniform-space equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-map exactness, modulus-ledger exhaustion, RegSeqRat readback discipline, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCompletionUni

Chapter 7279

A Concrete Naming Certificate for $\mathsf{DiscreteGronwall}^{\uparrow}$

Following Bellman, Gronwall, and Bishop–Bridges finite induction practice, the $\mathsf{DiscreteGronwall}^{\uparrow}$ packet records the stepwise inequality ledger used when a Cauchy-rate or Picard-style error estimate is bounded by a finite product-and-sum majorant. It sits beside the continuous envelope route of chapter 2034, the Picard iteration surfaces of chapter 2359, chapter 2233, and chapter 5794, the Cauchy-rate criteria of chapter 3343 and chapter 6981, the summable-tail ledger of chapter 1799, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite dyadic induction boundary; it is not a continuous differential inequality, not a selected infinite product, and not an ambient completeness principle.

Definition 7279.1 (DiscreteGronwall finite-step carrier)**.** *A* $\mathsf{DiscreteGronwall}^{\uparrow}$ *finite-step carrier is a finite* BHist *packet*

$$G = (I, U, A, B, T, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *I is a finite initial segment of step indices with a distinguished first step and successor relation;*
2. *U is the nonnegative error ledger recording the displayed value at each step of I;*
3. *A is the dyadic multiplicative coefficient ledger, with each coefficient read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *B is the dyadic additive inhomogeneity ledger;*
5. *T is the finite product-and-sum majorant row generated by primitive recursion along I;*
6. *Q is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row used when the additive ledger is read as a Cauchy tail;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic bounds;*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed to Real or Completion consumers;*
9. *H records componentwise* hsame *transport for the index, error, coefficient, tail, readback, seal, continuation, package, and local naming rows;*
10. *C records the displayed* Cont *replay of the finite induction;*
11. *P is the* Pkg *provenance over* $\mathsf{DiscreteGronwall}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DiscreteGronwall}^{\uparrow}}$ *row.*

The local classifier compares two accepted carriers only through the finite index segment, displayed error ledger, dyadic coefficient rows, primitive-recursive majorant, summable-tail read, regular-rational readback, Real seal, transport, continuation replay, provenance, and local naming.

Theorem 7279.2 (DiscreteGronwall NameCert obligations)**.** *For a carrier $G = (I, U, A, B, T, Q, R, S, H, C, P, N)$ of Definition 7279.1, the local* $\mathsf{NameCert}_{\mathsf{DiscreteGronwall}^{\uparrow}}$ *surface has six obligations:*

1. *carrier exposure of the finite tuple $I, U, A, B, T, Q, R, S, H, C, P, N$;*
2. *step admission for the first step and successor reads in I;*
3. *dyadic coefficient admission for every row of A and B;*
4. *primitive induction exactness for the majorant row T;*
5. *Cauchy-rate handoff through Q, R, S when a consumer reads a tail estimate;*
6. *non-escape from continuous Gronwall arguments, selected infinite products, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiscreteGronwallUp.DiscreteGronwallNamecertObligations`

Proof. Carrier exposure is coordinate projection from the displayed finite tuple. The classifier has no source outside $I, U, A, B, T, Q, R, S, H, C, P, N$.

Step admission and dyadic coefficient admission are read before the induction starts. The first-step row and successor relation bound the recursion to the finite segment I, while A and B force each multiplicative and additive coefficient through the dyadic ledger.

Primitive induction exactness is then obtained by induction over I. At the first step, T records the displayed initial bound. At a successor step, the previous majorant is multiplied by the visible coefficient row and then enlarged by the visible additive row, so the next error entry of U is bounded by the next entry of T.

For a Cauchy-rate consumer, the terminal read is through Q, R, S. The summable-tail row supplies the finite tail modulus, the regular-rational row repacks the dyadic estimates, and the Real seal exposes the same bound to Real and Completion consumers.

Since no other coordinate is present, the packet cannot export a continuous Gronwall theorem, a selected infinite product, countable choice, quotient real equality, or ambient Real completeness. □

Theorem 7279.3 (DiscreteGronwall finite induction boundary)**.** *Every accepted* $\mathsf{DiscreteGronwall}^{\uparrow}$ *packet bounds its displayed error ledger only by primitive induction over the finite step segment I. Picard-style iteration and L10 Cauchy-rate consumers must read the dyadic coefficient ledgers A, B, the recursive majorant T, the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{Real}^{\uparrow}$ *seal S before exporting a rate bound. The packet exports no continuous-time envelope, no hidden infinite product, no selected limit, and no ambient completeness theorem.*

Proof. Begin with the carrier. The only rows available to the consumer are the finite step segment, error ledger, dyadic coefficient ledgers, primitive-recursive majorant, tail modulus, readback, Real seal, transport, replay, provenance, and local naming rows.

Induct over the finite segment I. The base step reads the initial error row and the initial majorant entry. The successor step applies the displayed dyadic multiplicative row and additive row to the previous majorant entry, producing the next entry of T and bounding the corresponding entry of U.

Now route the bound to the downstream ledgers. Picard consumers read the same finite estimate as an iteration budget through $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, and $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, while Cauchy-rate consumers read the tail through $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The continuous $\mathsf{GronwallInequality}^{\uparrow}$ chapter remains a sibling reference rather than a source of this proof. Because the finite packet has no coordinate for a continuous envelope, hidden infinite product, selected limit, or ambient completeness theorem, every exported bound stays inside the displayed induction route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiscreteGronwall}^{\uparrow}}$.

Constructive story: A $\mathsf{DiscreteGronwall}^{\uparrow}$ packet is generated from a finite step segment, error ledger, dyadic multiplicative and additive coefficients, primitive-recursive majorant, summable-tail row, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiscreteGronwall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7279.1, Definition 7279.2, and Definition 7279.3 seed the finite dyadic induction boundary for Cauchy-rate and Picard-style error ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiscreteGronwall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close continuous Gronwall or Bihari inequalities, selected infinite products, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Further closure requires Lean encoding of the finite-step carrier and a checked primitive-induction theorem for the displayed dyadic majorant.

Chapter 7280

A Concrete Naming Certificate for $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$

The $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ packet realizes the refusal taxonomy of chapter 9412 as finite NameCert data. It sits after the generic refusal packet of chapter 6558, the pseudotheory-specific gate of chapter 6560, the discovery-certificate discipline of Definition 9410.5, and the TasteGate boundary of chapter 9459. The packet does not certify positive discovery; it records the finite countermodel rows that block an asserted positive payload when its evidence is unpaid.

Definition 7280.1 (Pseudotheory refutation countermodel carrier)**.** *A* $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, C, A, D, T, B, X, R, V, H, K, P, N)$$

with the following displayed rows:

1. *S is the structural-DNA collision row, naming the candidate payload, a prior payload, and the checked equality of their structural fingerprints.*
2. *C is the classifier-shift failure row, recording the asserted before and after classifiers together with the missing or neutral disagreement support.*
3. *A is the axiom-scan row, carrying the source-token and kernel-dependency evidence for forbidden axiom, sorry, or dependency exposure.*
4. *D is the certificate-debt row, recording unpaid semantic naming, row-evidence, ledger-membership, cost, debt, or scope-seal obligations.*
5. *T is the TasteGate admission-boundary row, marking the human-only value and stability judgment that is not computed by the cheap refusal gate.*
6. *B is the transport row carrying componentwise* hsame *comparisons for $S, C, A, D, T, X, R, V, K, P, N$.*
7. *X is the replay row, recording displayed* Cont *replay from the positiveDiscovery payload through the countermodel rows.*
8. *R is the provenance row tying the packet to the refutation-gate, pseudotheory-refutation-gate, discovery-certificate, theory-stability-taste-gate, and NameCert surfaces.*
9. *V is the verdict row, storing which of S, C, A, D is displayed against the asserted positive payload.*

10. *H is the admission-history row preserving the candidate's local chapter, carrier, classifier, and evidence ledger names.*
11. *K is the local audit-key row used to replay the refusal without reading unlisted corpus data.*
12. *P is the* Pkg *provenance over* $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$, $\mathsf{RefutationGate}^{\uparrow}$, PseudotheoryRefutationGat BHist, hsame, Cont, Pkg, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a positive-discovery witness, an automatic TasteGate admission, an unscanned axiom surface, or a hidden corpus oracle.

Theorem 7280.2 (Pseudotheory refutation countermodel NameCert obligations)**.** *For every accepted* $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ *carrier*

$$Q = (S, C, A, D, T, B, X, R, V, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}}$ *surface consists exactly of structural-DNA collision admission, classifier-shift failure admission, axiom-scan admission, certificate-debt admission, TasteGate boundary marking, componentwise* hsame *transport, displayed* Cont *replay, provenance, verdict selection, admission-history preservation, audit-key replay,* Pkg *source, and local naming.*

Proof. Project the carrier of Definition 7280.1. The refusal rows are S, C, A, D, the taste boundary is T, and the structural rows are B, X, R, V, H, K, P, N.

Rows S, C, A, D correspond to the finite countermodel classes: structural-DNA collision, classifier-shift failure, axiom exposure, and unpaid certificate debt. The verdict row V records which displayed row is being used against the asserted positive payload.

The TasteGate boundary T separates cheap refusal from human admission. It can mark where value and stability judgment is required, but it is not a positive discovery witness and does not convert the absence of a countermodel row into admission.

Finally B, X, R, H, K, P, N transport, replay, source, preserve, audit, package, and name the same finite refusal route. Since the carrier has no coordinate for the excluded host objects, these rows exhaust the local NameCert obligations. □

Theorem 7280.3 (Pseudotheory refutation countermodel audit route)**.** *Every audit read of an accepted* $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ *carrier factors through the finite route*

$$S, C, A, D, T, B, X, R, V, H, K, P, N.$$

If the verdict row displays a structural-DNA collision row, classifier-shift failure row, axiom-scan row, or certificate-debt row, then the audit blocks the asserted positive payload unless the payload supplies the corresponding paid evidence.

Proof. Begin with Definition 7280.2. It identifies S, C, A, D as the only rows that can carry a cheap finite refusal.

The verdict row V reads one of those rows against the asserted payload. The replay row X and audit-key row K make the read repeatable without importing unlisted corpus data or a hidden oracle.

The TasteGate boundary T is read separately from the cheap refusal rows. It marks human admission requirements and therefore cannot be used as automatic positive evidence.

Componentwise transport, provenance, admission history, package source, and local naming preserve the same finite route. Hence every accepted audit read is the displayed countermodel route, and a positive payload without the matching paid evidence is blocked by the displayed verdict. □

Falsifiable prediction: A positive discovery claim is refuted when a $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ packet displays a structural-DNA collision row, classifier-shift failure row, axiom-scan row, or certificate-debt row while the positiveDiscovery payload lacks the corresponding paid evidence.

Independence witness: refutation-gate, pseudotheory-refutation-gate, discovery-certificate, theory-stability-taste-gate, namecert: $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries concrete countermodel rows, TasteGate boundary marking, verdict replay, provenance, audit key, transport, and local naming as one packet.

The Lean TasteGate surface for this packet must later provide explicit Nontrivial PseudotheoryRefutationCounterm and FieldFaithful $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ instances. Those instances must keep $S, C, A, D, T, B, X, R,$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}}$.

Constructive story: A $\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}$ packet is generated from structural-DNA collision, classifier-shift failure, axiom-scan, certificate-debt, TasteGate boundary, transport, replay, provenance, verdict, admission-history, audit-key, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7280.1, Definition 7280.2, and Definition 7280.3 seed the finite countermodel route for pseudotheory refusal audits.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PseudotheoryRefutationCountermodel}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim positive discovery certification, automatic TasteGate admission, Lean verification, bridge checking, or a hidden corpus oracle.

Upgrade path: Obligation closure requires checked carrier admission, countermodel-row exactness, verdict replay, TasteGate boundary separation, transport, provenance, audit-key replay, and local naming for BEDC.Derived.PseudotheoryRefutationCounterm

Chapter 7281

A Concrete Naming Certificate for $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$ packet records the finite operation row by which a regular Cauchy name is read as a Real-facing limit without importing an ambient completeness theorem. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names an operation-facing carrier for the displayed limit route; it is not quotient equality of streams, a selected host limit, arbitrary trichotomy, countable choice, or completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7281.1 (Regular Cauchy Bishop limit carrier)**.** *A* $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular-Cauchy row, displaying the finite modulus and tail-comparison evidence accepted by the operation.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source name is inspected at the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the exact dyadic scale used by the window comparison.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the source row, stream window, and dyadic ledger are visible.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular-rational readback row R.*
6. *H records componentwise* hsame *transport for source, window, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, W, D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBishopLimit}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, arbitrary real trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.RegularCauchyBishopLimitUp`

Theorem 7281.2 (Regular Cauchy Bishop limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$ *carrier $L = (S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBishopLimit}^{\uparrow}}$ *surface consists exactly of carrier admission; source regular-Cauchy visibility through S; finite-window coverage through W; dyadic exactness through D;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. A limit-operation witness missing any one of these rows is not an accepted packet.*

Proof. Project the carrier of Definition 7281.1. The rows S, W, D are the complete finite source surface for the regular-Cauchy operation, so carrier admission, finite-window coverage, and dyadic exactness are all visible before any Real-facing read.

Read the handoff rows next. The regular rational row R is downstream of S, W, D, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream of R.

Finally the structural rows H, C, P, N transport, replay, source, and name the same displayed dependency route. Since no excluded quotient equality, selected limit, trichotomy, choice, or completeness row occurs in the carrier, every accepted obligation stays inside the listed NameCert surface. □

Theorem 7281.3 (Regular Cauchy Bishop limit Real-seal handoff)**.** *Every Real-facing limit read of an accepted* $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. The handoff exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, arbitrary trichotomy, countable choice, or selected host limit.*

Proof. Apply Definition 7281.2. It identifies the source row S, stream-window row W, and dyadic tolerance ledger D as the only finite input surface for the operation-facing limit read.

The readback row R is therefore available only through that finite input surface. A Real-facing consumer reaches the seal row E only after this $\mathsf{RegSeqRat}^{\uparrow}$ readback has been displayed.

The transport, replay, provenance, and local naming rows preserve the same dependency order. No carrier row can supply a host completeness theorem, quotient stream equality, trichotomy, countable choice, or selected limit, so the terminal handoff remains a finite seal rather than an ambient Real construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBishopLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBishopLimit}^{\uparrow}$ packet is generated from a source regular-Cauchy row, a StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyBishopLimit}^{\uparrow}}$, seed)

Scope closed: Definition 7281.1, Definition 7281.2, and Definition 7281.3 seed the finite Bishop-style limit-operation route from regular Cauchy evidence to the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyBishopLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient RealUp completeness, quotient stream equality, arbitrary trichotomy, countable choice, selected host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, dyadic exactness, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBishopLimitUp

Chapter 7282

A Concrete Naming Certificate for $\mathsf{FuchsianGroupAction}^{\uparrow}$

Following Poincare and Klein on discrete groups of hyperbolic disk transformations, the $\mathsf{FuchsianGroupAction}^{\uparrow}$ packet records a finite NameCert surface for a Fuchsian-style action on displayed Poincare disk data. It sits after the disk metric route of chapter 6209, the hyperbolic metric rows of chapter 3474, the reversible disk automorphism packet of chapter 6776, the boundary-shadow handoff of chapter 3813, and the visual-boundary readback of chapter 4181. The chapter names a finite action packet; it does not construct an orbit quotient, a global hyperbolic surface, a Kleinian boundary object, or a selected fundamental polygon.

Definition 7282.1 (Fuchsian group action carrier)**.** *A* $\mathsf{FuchsianGroupAction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, W, M, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite disk-point row, recording the displayed source and target points on the Poincare disk.*
2. *W is the Mobius-action word row, recording the finite generator word whose partial products are inspected.*
3. *M is the metric-preservation row over* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *and* $\mathsf{HyperbolicMetric}^{\uparrow}$*, attached to every displayed action step.*
4. *B is the boundary-handoff row through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *and* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *for every boundary-facing read of the same action word.*
5. *H records componentwise* hsame *transport for disk points, action words, metric preservation, boundary handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay through disk-point reads, word reads, metric-preservation checks, and boundary handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{FuchsianGroupAction}^{\uparrow}$*,* $\mathsf{PoincareDiskMetric}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$*,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FuchsianGroupAction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through D, W, M, B, H, C, P, N. It has no coordinate for a quotient orbit space, a global hyperbolic surface theorem, a chosen fundamental polygon, an ambient Kleinian group object, or host-level boundary equality.

Lean4 (statement) `BEDC.Derived.FuchsianGroupActionUp`

Theorem 7282.2 (Fuchsian group action NameCert obligations)**.** *For every accepted* $\mathsf{FuchsianGroupAction}^{\uparrow}$ *carrier* $A = (D, W, M, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FuchsianGroupAction}^{\uparrow}}$ *surface consists exactly of finite disk-point admission; finite Mobius-action word admission through the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *route; metric-preservation exactness through* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *and* $\mathsf{HyperbolicMetric}^{\uparrow}$*; boundary handoff through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *and* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$*; componentwise transport; displayed replay; provenance; and local naming. An action witness missing any one of* D, W, M, B, H, C, P, N *is not an accepted packet.*

Proof. Project the carrier of Definition 7282.1. The disk row D and word row W are the only displayed action source rows, so admission of the finite action is scoped before any metric or boundary read.

The metric row M is then read through the existing Poincare disk and hyperbolic metric packets, while the action word is routed through the disk-automorphism packet. Boundary-facing reads pass through B, not through a quotient boundary object.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no orbit quotient, global surface theorem, chosen polygon, ambient Kleinian object, or host boundary equality row occurs in the carrier, the listed clauses exhaust the local NameCert surface. □

Theorem 7282.3 (Fuchsian group action boundary handoff)**.** *Every boundary-facing read of an accepted* $\mathsf{FuchsianGroupAction}^{\uparrow}$ *carrier factors through the finite disk-point row, the Mobius-action word row, the metric-preservation row, and the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *and* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *boundary rows. The handoff exports no quotient orbit space, global hyperbolic surface theorem, selected fundamental polygon, or host boundary equality.*

Proof. Apply Definition 7282.2. It identifies D, W, M, B as the complete geometric surface before any boundary-facing consumer can read the action.

The boundary row B is downstream of the same displayed disk points, action word, and metric-preservation evidence. Thus a boundary consumer cannot bypass the finite action packet and cannot replace it by an ambient quotient or surface construction.

The transport, replay, provenance, and local naming rows preserve this dependency order. Because the carrier has no row for a quotient orbit space, global surface theorem, selected polygon, or host boundary equality, the boundary handoff remains a finite NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FuchsianGroupAction}^{\uparrow}}$.

Constructive story: A $\mathsf{FuchsianGroupAction}^{\uparrow}$ packet is generated from finite disk-point rows, finite Mobius-action word rows, PoincareDiskMetric and HyperbolicMetric preservation rows, HyperbolicBoundaryShadow and HyperbolicVisualBoundary handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FuchsianGroupAction}^{\uparrow}}$, seed)

Scope closed: Definition 7282.1, Definition 7282.2, and Definition 7282.3 seed the finite Fuchsian-style action route over disk, metric, automorphism, and boundary rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FuchsianGroupAction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient orbit spaces, global hyperbolic surface theorems, selected fundamental polygons, Kleinian boundary theory, host boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-point admission, Mobius-action word admission, metric-preservation exactness, boundary handoff, transport, replay, provenance, and local naming for BEDC.Derived.FuchsianGroupActionUp.

Chapter 7283

A Concrete Naming Certificate for $\mathsf{ProofNet}^{\uparrow}$

Following Girard proof nets, the $\mathsf{ProofNet}^{\uparrow}$ packet records proof objects as finite link-and-switching data rather than as host proof trees or quotient proof identities. It sits between the syntax and deduction rows of chapter 197, the proof-term corridor of chapter 200, the normalization-by-evaluation surface of chapter 6305, the logical-relation route of chapter 6767, and the MetaCIC critical path of chapter 765. The packet names finite link tables, switching choices, cut interfaces, transport, replay, provenance, and local naming rows; it does not assert full normalization, confluence, quotient proof equality, a completed semantic model, or a global cut-elimination theorem.

Definition 7283.1 (Proof-net carrier)**.** *A* $\mathsf{ProofNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, S, C, B, R, H, Q, N)$$

with the following displayed rows:

1. *L is the link table, listing axiom, tensor, par, unit, dereliction, contraction, weakening, and cut links only as finite syntax rows.*
2. *S is the switching row, recording the finite switching choices and connectedness checks exposed to a consumer.*
3. *C is the cut-interface row, naming the displayed cut pairs and the local redex boundary without collapsing proof nets by cut conversion.*
4. *B is the boundary row, tying conclusions, formula occurrences, and sequent-facing endpoints to the* $\mathsf{FirstOrder}^{\uparrow}$ *and* $\mathsf{LambdaCalc}^{\uparrow}$ *source surfaces.*
5. *R is the proof-net route row consumed by* $\mathsf{NormalizationByEvaluation}^{\uparrow}$*,* $\mathsf{LogicalRelation}^{\uparrow}$*, and* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *readers before they may cite proof-object data.*
6. *H records componentwise* hsame *transport for link tables, switching rows, cut interfaces, boundaries, routes, provenance, and local naming.*
7. *Q is the* Pkg *provenance over* $\mathsf{ProofNet}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{LambdaCalc}^{\uparrow}$*,* $\mathsf{NormalizationByEvaluation}^{\uparrow}$*,* $\mathsf{LogicalRelation}^{\uparrow}$*,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for completed proof normalization, Church–Rosser confluence, quotient proof equality, a semantic completeness theorem, or a host evaluator.

Theorem 7283.2 (Proof-net NameCert obligations)**.** *For every accepted* $\mathsf{ProofNet}^{\uparrow}$ *carrier* $P = (L, S, C, B, R, H, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProofNet}^{\uparrow}}$ *surface consists exactly of carrier admission for the link table, switching row, cut-interface row, boundary row, proof-net route row, componentwise* hsame *transport,* Pkg *provenance, and local naming. A consumer may read proof-net data only after the finite links, switchings, cuts, boundaries, and route rows are displayed in the same packet.*

Proof. Project the carrier of Definition 7283.1. The packet exposes only L, S, C, B, R, H, Q, N.

The proof-object obligations are exhausted by L, S, C, B, R. Link data are finite syntax rows, switching data are finite checks, cut data are local interfaces, and boundary data keep the proof-net endpoints attached to the $\mathsf{FirstOrder}^{\uparrow}$ and $\mathsf{LambdaCalc}^{\uparrow}$ surfaces.

The rows H, Q, N transport, source, and name the same finite packet. Since no carrier coordinate stores global normalization, confluence, quotient proof equality, semantic completeness, or a host evaluator, the listed rows are exactly the local certificate obligations. □

Theorem 7283.3 (Proof-net cut-elimination route)**.** *Every cut-elimination-facing read of an accepted* $\mathsf{ProofNet}^{\uparrow}$ *packet must first expose*

$$L, S, C, B, R, H, Q, N$$

and may then hand the displayed local cut-interface route to $\mathsf{NormalizationByEvaluation}^{\uparrow}$, $\mathsf{LogicalRelation}^{\uparrow}$, *or* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumers. The handoff is a finite proof-net carrier route over displayed links, switchings, and cut interfaces; it is not a proof of full normalization, unrestricted confluence, quotient proof equality, or a completed semantic model.*

Proof. Read the link and switching rows L, S first. They display the finite graph and switching data available to a proof-net consumer.

Next read the cut-interface row C and boundary row B. These rows identify the local redex boundary and the sequent or proof-term endpoints without quotienting proof nets by cut conversion.

Finally read the route and structural rows R, H, Q, N. They allow the normalization-by-evaluation, logical-relation, and MetaCIC critical-path consumers to cite the same finite route, while the carrier inventory excludes full normalization, confluence, quotient proof equality, and completed semantic-model coordinates. □

Falsifiable prediction: Every accepted $\mathsf{ProofNet}^{\uparrow}$ consumer displays finite link tables, switching rows, cut-interface rows, boundary rows, proof-net route rows, componentwise hsame transport, Pkg provenance, and a local NameCert row before citing proof-object data.

Independence witness: firstorder, lambdacalc, normalization-by-evaluation, logical-relation, metaciccriticalpath: $\mathsf{ProofNet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite link tables, switching rows, cut-interface rows, proof-object boundaries, proof-net route rows, transport, provenance, and local naming while refusing full normalization, confluence, quotient proof equality, and completed semantic-model claims.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProofNet}^{\uparrow}$ packet is generated from finite link tables, switching rows, cut-interface rows, boundary endpoints, proof-net route rows, componentwise hsame transport, Pkg provenance, and local naming evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProofNet}^{\uparrow}}$, seed)

Scope closed: Definition 7283.1, Definition 7283.2, and Definition 7283.3 seed the finite proof-net carrier surface for proof-object consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProofNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, full normalization, unrestricted confluence, quotient proof equality, semantic completeness, or a completed MetaCIC model.

Upgrade path: Obligation closure requires checked carrier admission, switching exposure, cut-interface route preservation, boundary compatibility, transport, provenance, and local naming for BEDC.Derived.ProofNetUp.

Chapter 7284

A Concrete Naming Certificate for LocatedCauchyIntervalCompletion$^{\uparrow}$

Following Bishop and Bridges on located intervals and regular Cauchy completion, the LocatedCauchyIntervalComple packet records the finite BEDC route by which a located interval source produces a regular Cauchy interval-name surface and then reaches the Real-facing seal. It sits after the located interval carrier of chapter 1709, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real$^{\uparrow}$ boundary of chapter 38, and the completion criterion of chapter 3404. The packet is a located interval completion certificate, not quotient real equality, open-cover compactness, countable choice, a selected limit, or ambient Real$^{\uparrow}$ completeness.

Definition 7284.1 (Located Cauchy interval completion carrier)**.** *A* LocatedCauchyIntervalCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, W, D, Q, S, A, H, C, P, N)$$

with the following rows:

1. *I is the* LocatedRealInterval$^{\uparrow}$ *source row, exposing endpoints, radius data, and locatedness probes as finite interval evidence.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row through which interval observations are requested.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for interval widths, endpoint comparison slack, and refinement budgets.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerances.*
5. *S is the* BishopCompletionCriterion$^{\uparrow}$ *row recording that the regular interval windows meet the displayed Cauchy completion criterion.*
6. *A is the terminal* Real$^{\uparrow}$ *seal row reached only after I, W, D, Q, S have been displayed.*
7. *H records componentwise* hsame *transport for the interval source, windows, dyadic ledger, regular readback, completion criterion, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion request through the same ordered rows.*
9. *P is the* Pkg *provenance over* LocatedCauchyIntervalCompletion$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BishopCompletionCriterion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyIntervalCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these ten rows form one located Cauchy interval completion packet. It has no coordinate for quotient reals, selected representatives, host interval equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedCauchyIntervalCompletionUp.LocatedCauchyIntervalCompletionUp`
Lean4 ✓ `BEDC.Derived.LocatedCauchyIntervalCompletionUp.LocatedCauchyIntervalCompletionTasteGate`
`single carrier alignment`

Theorem 7284.2 (Located Cauchy interval completion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyIntervalCompletion}^{\uparrow}$ *carrier* $T = (I, W, D, Q, S, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyIr}}$ *surface consists exactly of carrier admission for the located interval source, finite windows, dyadic tolerance ledger, regular readback, Bishop completion criterion, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows. It also includes route exactness from I through W, D, Q, S, A and ledger exhaustion excluding quotient reals, selected limits, open-cover compactness, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 7284.1. The projected rows are exactly the located interval source, finite window, dyadic ledger, regular readback, completion criterion, Real seal, and structural rows.

Route exactness follows from the tuple order. A public completion read begins with I, opens only the finite windows in W, measures them by the dyadic ledger D, reads them through Q, and then checks the displayed Bishop completion criterion S before A is available.

The transport, replay, provenance, and local naming obligations are the rows H, C, P, N. Since the carrier contains no further coordinate, the excluded quotient, choice, compactness, host equality, selected-limit, and ambient-completeness principles cannot be exported. □

Theorem 7284.3 (Located Cauchy interval completion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{LocatedCauchyIntervalCompletion}^{\uparrow}$ *carrier is downstream of the displayed route*

$$\mathsf{LocatedRealInterval}^{\uparrow}(I) \longmapsto \mathsf{StreamName}^{\uparrow}(W) \longmapsto \mathsf{DyadicRatCore}^{\uparrow}(D) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(Q) \longmapsto \mathsf{BishopCompletionC}$$

The seal therefore names only the finite located Cauchy interval completion read and exports no quotient stream equality, selected host limit, countable-choice schedule, open-cover compactness witness, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source interval row I. It supplies endpoint, radius, and locatedness data through the located real interval certificate rather than through a host closed interval.

Read the finite observation route. The stream row W schedules the inspected windows, the dyadic ledger D records the tolerance and refinement budget, and Q supplies the regular rational readback for the requested precision.

Apply the completion criterion row S. The Real seal A is public only after that displayed criterion has been read from the same finite route.

The structural rows H, C, P, N transport and replay this route without adding analytic data. Thus the listed quotient, choice, compactness, host, selected-limit, and ambient-completeness principles remain outside the packet. □

Theorem 7284.4 (Located Cauchy interval completion L10 handoff)**.** *Any Phase Real and Completion consumer of* $\mathsf{LocatedCauchyIntervalCompletion}^{\uparrow}$ *must consume the L10 handoff through the*

four displayed source faces DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *with the located interval row and Bishop completion criterion kept visible as source and acceptance rows. Agreement on a terminal Real seal alone cannot replace the displayed interval source, tolerance ledger, regular readback, or completion criterion.*

Proof. Project the L10 faces from Definition 7284.1. The rows W, D, Q, A are exactly the StreamName, DyadicRatCore, RegSeqRat, and Real faces available to a Real and Completion consumer.

Keep the source and acceptance rows visible. The located interval row I supplies the finite interval data, while S records the Bishop completion criterion that permits the terminal seal to be read as a completion handoff.

Use Definition 7284.2 and Definition 7284.3. Together they force every accepted handoff through $I, W, D, Q, S, A, H, C, P, N$ and rule out terminal-seal agreement as a substitute for the displayed route. □

Falsifiable prediction: Every accepted LocatedCauchyIntervalCompletion$^{\uparrow}$ read must expose a located interval source, StreamName finite windows, DyadicRatCore tolerance ledger, RegSeqRat readback, BishopCompletionCriterion acceptance, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real-interval, streamname, dyadicratcore, regseqrat, bishop-completion-criterion, real: LocatedCauchyIntervalCompletion$^{\uparrow}$ is not bijective to any listed sibling because it binds located interval source data to the regular Cauchy completion handoff and terminal Real seal in one finite NameCert packet.

Closure status for NameCert$_{\text{LocatedCauchyIntervalCompletion}^{\uparrow}}$.

Constructive story: A LocatedCauchyIntervalCompletion$^{\uparrow}$ packet is generated from a located interval source row, finite StreamName windows, DyadicRatCore tolerance data, RegSeqRat readback, a BishopCompletionCriterion row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCauchyIntervalCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7284.1, Definition 7284.2, Definition 7284.3, and Definition 7284.4 seed the located Cauchy interval completion route through located interval, StreamName, DyadicRatCore, RegSeqRat, BishopCompletionCriterion, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyIntervalCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, open-cover compactness, countable choice, selected host limits, host interval equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route exactness, Real-seal nonescape, L10 handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyIntervalCompletionUp.

Chapter 7285

A Concrete Naming Certificate for $\mathsf{PrincipalBundle}^{\uparrow}$

Following Steenrod and Kobayashi–Nomizu, the $\mathsf{PrincipalBundle}^{\uparrow}$ packet records principal-bundle data as finite BEDC NameCert rows. It sits over the displayed bundle carrier of chapter 97, the charted base surface of chapter 96, the smooth structure-group rows of chapter 140, the algebraic action boundary of chapter 41, and the continuous operation ledger of chapter 204. The packet names only the visible total space, base manifold, projection, local trivializations, transition cocycle, group action, transport, replay, provenance, and local certificate rows; it does not construct quotient bundle equality, choose a global section, assert a host smooth structure, or provide bridge checking.

Definition 7285.1 (Principal bundle carrier)**.** *A* $\mathsf{PrincipalBundle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (E, B, \pi, G, U, \tau, \gamma, \alpha, H, C, \rho, N)$$

with the following displayed rows:

1. *E is the total-space row, read through the upstream* $\mathsf{Bundle}^{\uparrow}$ *carrier surface.*
2. *B is the base-manifold row, read through the chart carrier of* $\mathsf{Manifold}^{\uparrow}$*.*
3. *π is the projection row from E to B, displayed as bundle data rather than as a host map equality.*
4. *G is the structure-group row, read through the* $\mathsf{LieGroup}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*, and* $\mathsf{TopGroup}^{\uparrow}$ *operation surfaces.*
5. *U is the finite local-trivialization cover row, including the displayed local product windows used by the bundle packet.*
6. *τ is the transition-cocycle row over overlapping local trivialization windows.*
7. *γ is the group-action row, naming the right action of the displayed structure group on the displayed total-space fibres.*
8. *α is the* hsame *transport row for the total space, base, projection, structure group, local trivializations, transition cocycle, and group action.*
9. *H is the displayed* Cont *replay row for reading the same local data after transport.*
10. *C is the* Pkg *provenance over* $\mathsf{PrincipalBundle}^{\uparrow}$*,* $\mathsf{Bundle}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* $\mathsf{LieGroup}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* $\mathsf{TopGroup}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

11. *ρ records the finite dependency order in which the upstream rows are exposed to a principal-bundle consumer.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PrincipalBundle}^\uparrow}$ *row.*

The carrier has no coordinate for quotient bundle equality, selected global sections, ambient smooth structures, external atlas maximality, or detached host principal-bundle objects.

Theorem 7285.2 (Principal bundle NameCert obligations)**.** *For the carrier of Definition 7285.1, the local* $\mathsf{NameCert}_{\mathsf{PrincipalBundle}^\uparrow}$ *surface consists exactly of the following displayed obligations:*

1. *carrier admission for the packet $P = (E, B, \pi, G, U, \tau, \gamma, \alpha, H, C, \rho, N)$;*
2. *bundle exposure through the total-space, projection, and local-trivialization rows E, π, U;*
3. *base exposure through the* $\mathsf{Manifold}^\uparrow$ *row B;*
4. *structure-group exposure through the* $\mathsf{LieGroup}^\uparrow$, $\mathsf{Group}^\uparrow$, *and* $\mathsf{TopGroup}^\uparrow$ *readings of G;*
5. *action and transition exposure through γ and τ;*
6. *componentwise transport, replay, dependency provenance, and local naming through α, H, C, ρ, N.*

Ledger exactness says that every accepted principal-bundle read exposes precisely these rows before a principal-bundle consumer may inspect the transition or action data. The obligations do not add quotient bundle equality, selected global sections, host smooth structures, atlas maximality, or bridge checking.

Theorem 7285.3 (Principal bundle transition cocycle)**.** *Let $P = (E, B, \pi, G, U, \tau, \gamma, \alpha, H, C, \rho, N)$ be an accepted* $\mathsf{PrincipalBundle}^\uparrow$ *carrier. Every transition-cocycle read from P factors through the displayed base-manifold row B, the local-trivialization row U, and the structure-group row G before the cocycle row τ is public:*

$$\mathsf{Manifold}^\uparrow,\ \mathsf{Bundle}^\uparrow,\ \mathsf{LieGroup}^\uparrow,\ \mathsf{Group}^\uparrow,\ \mathsf{TopGroup}^\uparrow \longrightarrow \tau.$$

On triple overlaps, the cocycle comparison is recorded only as finite NameCert data transported by α and replayed by H. It cannot identify quotient bundles, choose a section, import a maximal smooth atlas, or replace the displayed structure-group row by an ambient host group.

Closure status for $\mathsf{NameCert}_{\mathsf{PrincipalBundle}^\uparrow}$.

Constructive story: A $\mathsf{PrincipalBundle}^\uparrow$ packet is generated from total-space, base-manifold, projection, structure-group, local-trivialization, transition-cocycle, and group-action rows, together with componentwise hsame transport, displayed Cont replay, Pkg provenance, dependency ordering, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PrincipalBundle}^\uparrow}$, seed)

Scope closed: Definition 7285.1, Definition 7285.2, and Definition 7285.3 seed the finite principal-bundle NameCert surface over displayed bundle, manifold, Lie-group, group, topological-group, action, and transition-cocycle rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PrincipalBundle}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient bundle equality, selected global sections, host smooth structures, atlas maximality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-cocycle exposure, group-action exposure, dependency exactness, transport, replay, provenance, and local naming for BEDC.Derived.PrincipalBundleUp.

Chapter 7286

A Concrete Naming Certificate for $\mathsf{DyadicCauchyStaircase}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ packet records the finite precision staircase by which dyadic tolerance rows are opened in monotone order before a regular rational name is read back and sealed as a Real-facing row. It sits between the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, the effective-rate schedule of chapter 415, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet is a finite precision-schedule certificate; it is not a quotient completion, a selected limit, a countable-choice operator, trichotomy, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7286.1 (Dyadic Cauchy staircase carrier)**.** *A* $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected windows of the source rational stream.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row ledger, listing the dyadic tolerance rows requested by the consumer.*
3. *L is the monotone staircase ledger, recording that later displayed rows of D refine earlier rows and that each read of S is attached to the current step.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed only after the relevant staircase step and stream window have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the staircase and regular readback route has been consumed.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic precision rows, staircase ledger, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through D, L, S, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyStaircase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyStaircase}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected limits, countable choice, trichotomy, host real equality, hidden stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicCauchyStaircaseUp.DyadicCauchyStaircaseTasteGate single carrier alignment`

Theorem 7286.2 (Dyadic Cauchy staircase NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ *carrier* $K = (S, D, L, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyStaircase}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger, monotone staircase ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *precision visibility, requiring each consumed tolerance request to appear as a displayed row of D before L advances the staircase;*
3. *monotone-step admission, requiring L to record that later displayed precision rows refine earlier displayed rows and that no step is skipped by the readback route;*
4. *window-readback routing, requiring S to be opened at the current staircase step before R is consumed;*
5. *real-seal order, requiring E to be read only after D, L, S, R have all been displayed in the same finite route;*
6. *transport and replay stability, requiring H and C to preserve the same precision, staircase, window, readback, and seal rows componentwise;*
7. *ledger nonescape from quotient completion, selected limits, countable choice, trichotomy, host real equality, hidden stream equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7286.1. The accepted packet displays exactly the rows $S, D, L, R, E, H, C, P, N$.

Precision visibility and monotone-step admission are the role of D and L. The dyadic rows in D name the requested tolerances, while L records their displayed refinement order and attaches each opened stream window to the current step.

Window-readback routing follows from the route order. A readback row R can be consumed only after the current dyadic precision row and its staircase step have opened the finite window S.

The real seal is downstream. The row E is read after D, L, S, R, and the structural rows H, C, P, N transport, replay, source, and name that same finite path.

The nonescape clause follows because none of the excluded objects is a coordinate of the carrier. The local certificate therefore exports only the displayed precision-staircase surface. □

Theorem 7286.3 (Dyadic Cauchy staircase L10 handoff)**.** *Every accepted* $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ *carrier* $K = (S, D, L, R, E, H, C, P, N)$ *hands any L10 Real or Completion consumer the ordered finite route*

$$D \longrightarrow L \longrightarrow S \longrightarrow R \longrightarrow E.$$

The route first displays the $\mathsf{DyadicRatCore}^{\uparrow}$ *precision rows, then the monotone staircase ledger, then the* $\mathsf{StreamName}^{\uparrow}$ *window, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal. A consumer*

receives no quotient completion, selected limit, countable-choice selector, trichotomy datum, host real equality, hidden stream equality, or ambient Real-completeness row.

Proof. Start with the accepted carrier $K = (S, D, L, R, E, H, C, P, N)$. Projection gives the dyadic precision ledger D, the staircase ledger L, the stream-window row S, the regular readback row R, and the terminal seal E.

The precision rows are consumed first. The staircase ledger L records their monotone refinement order and fixes which window of S is attached to each requested tolerance.

The regular readback R is then read through the opened window, and E is reached only after that readback has been displayed. The rows H, C, P, N preserve, replay, source, and name the same finite route.

Since the route has no carrier coordinate beyond $D, L, S, R, E, H, C, P, N$, the handoff cannot export quotient completion, selected limits, countable choice, trichotomy, host real equality, hidden stream equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: An accepted $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the DyadicRatCore precision rows, monotone staircase ledger, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming rows.

Independence witness: dyadicratcore, streamname, cauchy-rate, regseqrat, real: none alone is bijective with $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ because the packet jointly records finite precision rows, a monotone staircase ledger, stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyStaircase}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyStaircase}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore precision rows, a monotone staircase ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyStaircase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7286.1, Definition 7286.2, and Definition 7286.3 seed the finite precision-staircase certificate between dyadic tolerance rows, stream windows, regular rational readback, and the Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyStaircase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, hidden stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, precision visibility, monotone-step admission, window-readback routing, real-seal order, transport and replay stability, and ledger nonescape for BEDC.Derived.DyadicCauchyStaircaseUp.

Chapter 7287

A Concrete Naming Certificate for $\mathsf{SobolevDual}^{\uparrow}$

Following Sobolev and Adams–Fournier, the $\mathsf{SobolevDual}^{\uparrow}$ packet records the finite dual-test-action surface over the Sobolev carrier of chapter 5911. It reads the weak-derivative rows of $\mathsf{WeakDerivative}^{\uparrow}$, the norm budgets of $\mathsf{Norm}^{\uparrow}$, the scalar seals of $\mathsf{Real}^{\uparrow}$, and the completion handoff of $\mathsf{MetricCompletion}^{\uparrow}$ before any dual-facing consumer can inspect a test action or a budgeted functional value. The packet gives PDE, variational, and weak-form consumers a displayed BEDC route rather than a host distribution-dual quotient, a Hahn–Banach extension, or a selected representative of a function class.

Definition 7287.1 (SobolevDual carrier)**.** *A* $\mathsf{SobolevDual}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D^* = (S, T, A, R, B, M, H, C, P, L)$$

with the following displayed rows:

1. *S is the source Sobolev carrier row, read from the* $\mathsf{SobolevSpace}^{\uparrow}$ *packet of chapter 5911 after its source-function,* $\mathsf{WeakDerivative}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, integrability,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed.*
2. *T is the test-action family row. It records the finite test windows, derivative-coordinate tags, support tags, and admissible probe names on which the dual action may be evaluated.*
3. *A is the action ledger row. It binds each displayed test window in T to a finite action value request over the source Sobolev data in S.*
4. *R is the scalar* $\mathsf{Real}^{\uparrow}$ *readback row for action values, norm estimates, and dual-budget comparisons.*
5. *B is the norm and dual-budget row. It records the finite bound, tolerance, and comparison data by which the action ledger is admitted as a bounded dual-facing read.*
6. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, available only after S, T, A, R, B have been displayed and only for the completion route named by this dual packet.*
7. *H stores componentwise* hsame *transport for Sobolev source, test-action family, action ledger, scalar readback, norm and dual budget, completion handoff, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from every dual-facing request through $S, T, A, R, B, M, H, P, L$.*

9. *P is the* Pkg *provenance over* $\mathsf{SobolevDual}^{\uparrow}$, $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{SobolevDual}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, A, R, B, M, H, C, P, L$. It has no coordinate for host distribution-dual quotients, quotient equality of functions, Hahn–Banach extension, choice, PDE solvability, Lean verification, or bridge checking.

Theorem 7287.2 (SobolevDual NameCert obligations)**.** *For every accepted* $\mathsf{SobolevDual}^{\uparrow}$ *carrier*

$$D^* = (S, T, A, R, B, M, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SobolevDual}^{\uparrow}}$ *surface consists exactly of carrier admission, test-action coverage, dual-budget exactness, SobolevSpace handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No dual obligation is stored outside those finite rows, and no consumer may escape to a host distribution-dual quotient before reading the displayed Sobolev source and test-action route.*

Proof. Carrier admission is projection from Definition 7287.1. The accepted packet displays the Sobolev source row S, test-action family T, action ledger A, scalar row R, dual-budget row B, completion handoff M, and structural rows H, C, P, L.

Test-action coverage is finite. Every action value must first name a test window in T and then read the action ledger A against the displayed $\mathsf{SobolevSpace}^{\uparrow}$ source row S; there is no separate coordinate for a host distribution space or a quotient representative.

Dual-budget exactness is read through R and B. The scalar seal R is the only real-valued readback, while B records the bound and tolerance data that make the action available to dual-facing consumers.

The handoff to SobolevSpace is therefore prior to every dual read. The rows H, C, P, L transport, replay, source, and name the same finite route, so replay and provenance cannot add Hahn–Banach, choice, PDE-solvability, verification, or bridge data. □

Falsifiable prediction: Every accepted $\mathsf{SobolevDual}^{\uparrow}$ dual-facing consumer reads SobolevSpaceUp, WeakDerivativeUp, NormUp, RealUp, MetricCompletionUp, BHist, hsame, Cont, Pkg, and NameCert rows before receiving a test-action value or dual budget.

Independence witness: sobolev-space, weak-derivative, norm, real, metric-completion: no listed sibling is bijective to $\mathsf{SobolevDual}^{\uparrow}$ because this packet jointly binds a Sobolev source row, test-action family, action ledger, scalar readback, dual budget, completion handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevDual}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevDual}^{\uparrow}$ packet is generated from a displayed SobolevSpaceUp source row, finite test-action family, action ledger, RealUp scalar readbacks, norm and dual-budget data, MetricCompletionUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SobolevDual}^{\uparrow}}$, seed)

Scope closed: Definition 7287.1 and Definition 7287.2 seed the finite Sobolev-dual carrier from source Sobolev rows through test actions, scalar readback, dual budget, completion handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SobolevDual}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach duality, quotient functions, Hahn–Banach extension, choice, PDE solvability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-action coverage, dual-budget exactness, SobolevSpace handoff, transport, replay, provenance, and local naming for BEDC.Derived.SobolevDualUp.

Chapter 7288

A Concrete Naming Certificate for $\mathsf{CauchyFilterCompleteness}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ packet records the finite L10 route by which a located Cauchy filter may be read toward a Real-facing completion seal. It sits after the Cauchy-filter carrier of chapter 333, the located-filter boundary of chapter 1661, the Real-induced uniformity of chapter 1624, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the displayed rows a completion consumer may inspect; it does not quotient filters, select limits, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7288.1 (Cauchy filter completeness carrier)**.** *A* $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, L, U, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing the filter base, directed refinements, and Cauchy request ledger.*
2. *L is the located-tail row, read through* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *and carrying the finite interval or apartness evidence needed by a completion-facing request.*
3. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *uniform-entourage row used to compare the displayed tails by Real-induced metric balls.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and uniform requests.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, L, U, S, R have been displayed.*
7. *H records componentwise* hsame *transport of F, L, U, S, R, E.*
8. *C records displayed* Cont *replay of the same route.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCompleteness}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, U, S, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host equality of completed points, or ambient Real completeness.

Theorem 7288.2 (Cauchy filter completeness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCompletenes}$ *carrier $Q = (F, L, U, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy-filter exposure through F, located-tail exposure through L, Real-uniform entourage exposure through U, finite StreamName windows through S, RegSeqRat readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7288.1. The available analytic rows are exactly F, L, U, S, R, E, and the available structural rows are H, C, P, N.

The source side starts with F, L, U. The Cauchy-filter row supplies the displayed filter requests, the located-tail row supplies finite interval or apartness evidence, and the Real-uniform row supplies the entourage comparison used by the completion-facing read.

The numerical readback is downstream. Rows S and R expose only finite windows and regular rational observations, and the seal row E is available only after those rows have been displayed.

Transport, replay, provenance, and local naming preserve the same packet. Since quotient filters, selected limits, countable choice, host equality, and ambient Real completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7288.3 (Cauchy filter completeness located Real extraction)**.** *Every Real-facing extraction from an accepted* $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow},\ \mathsf{LocatedCauchyFilter}^{\uparrow},\ \mathsf{RealUniformStructure}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, H, C, P, N.$$

The extraction may expose a terminal Real seal only as the end of this displayed route; it does not provide a quotient filter, selected limit, countable-choice schedule, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the obligation surface of Definition 7288.2. It orders every Real-facing read through the Cauchy-filter row, located-tail row, Real-uniform entourage row, finite windows, regular rational readback, and terminal seal.

The located-tail and uniform-entourage rows are both visible before the seal. Thus the consumer can inspect how the filter request is located and how the Real-induced uniformity compares the inspected tails, but it cannot bypass those rows.

The seal row E is terminal. Rows H, C, P, N transport, replay, source, and name the same finite route, so the Real-facing extraction remains a displayed packet rather than a host completion construction. □

Theorem 7288.4 (Cauchy filter completeness nonescape)**.** *Every completion consumer of an accepted* $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ *packet receives only the displayed rows*

$$F, L, U, S, R, E, H, C, P, N.$$

Consequently the packet refuses quotient filters, selected representatives, selected limits, countable choice, host equality of completed points, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7288.1 to read the full carrier inventory. The only source rows are the Cauchy-filter, located-tail, uniform-entourage, StreamName, RegSeqRat, and Real-seal rows.

Apply Definition 7288.3. It shows that a Real-facing extraction is downstream from exactly those rows and the structural route H, C, P, N.

The refused objects are not carrier coordinates and are not produced by the structural rows. Therefore no completion consumer can obtain quotient-filter equality, a selected limit, a countable-choice schedule, host equality of completed points, or ambient Real completeness from this packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ read must expose a $\mathsf{CauchyFilter}^{\uparrow}$ row, a $\mathsf{LocatedCauchyFilter}^{\uparrow}$ located-tail row, a $\mathsf{RealUniformStructure}^{\uparrow}$ entourage row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCompleteness}^{\uparrow}$ packet is generated from a CauchyFilter row, located-tail row, RealUniformStructure entourage row, StreamName finite windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7288.1, Definition 7288.2, Definition 7288.3, and Definition 7288.4 seed the L10 Cauchy-filter completeness route over located filter, Real-uniform, StreamName, RegSeqRat, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyFilterCompletenessUp`

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, selected representatives, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification of the obligation rows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-tail exposure, Real-uniform entourage exposure, StreamName window inspection, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterCompletenessUp.

Chapter 7289

A Concrete Naming Certificate for $\mathsf{BishopCompletionFilter}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionFilter}^{\uparrow}$ packet records the constructive completion route in which a Cauchy filter is read through located filter data, regular Cauchy object rows, uniform completion-filter rows, dense embedding data, and the terminal real-name spine before any quotient completion or selected limit is available. It sits beside the Cauchy-filter carrier of chapter 333, the located filter boundary of chapter 1661, the object-level regular completion of chapter 7177, the uniform-space completion-filter route of chapter 4108, and the dense embedding surface of chapter 3617. Its finite readback path uses the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the $\mathsf{StreamName}^{\uparrow}$ window service of chapter 126.

Definition 7289.1 (Bishop completion-filter carrier)**.** *A* $\mathsf{BishopCompletionFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, L, O, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, exposing the filter-level Cauchy data before any completion read is attempted.*
2. *L is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, recording the located basis-refinement and tail-decision data consumed by the Bishop-style completion route.*
3. *O is the* $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$ *row, naming the regular Cauchy completion object reached from the located filter source.*
4. *U is the* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *row, recording the uniform entourage and filter-completion handoff for the same displayed source.*
5. *D is the* $\mathsf{DenseEmbedding}^{\uparrow}$ *row, exposing the source-to-completion dense image route before a completion-facing consumer reads the endpoint.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the filter, object, uniform, and dense-image ledgers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, L, O, U, D, W, R are visible.*
9. *H records componentwise* hsame *transport for $F, L, O, U, D, W, R, E, C, P, N$.*

10. *C records displayed* Cont *replay from a completion request through the filter, locatedness, object, uniform, dense-image, window, readback, seal, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyCompletionObject}^{\uparrow}$, $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$, $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient filters, selected representatives, countable choice, host equality of completion points, hidden cofinal subfilters, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7289.2 (Bishop completion-filter NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionFilter}^{\uparrow}$ *carrier*

$$B = (F, L, O, U, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionFilter}^{\uparrow}}$ *surface consists exactly of Cauchy-filter source admission, located-filter exposure, regular completion-object admission, uniform completion-filter handoff, dense-embedding route exposure, StreamName window admission, RegSeqRat readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7289.1. The analytic rows are F, L, O, U, D, W, R, E, and the structural rows are H, C, P, N.

Read the completion source first. The Cauchy filter row F is refined by the located filter row L, and the regular completion-object row O can be consumed only after those source rows are visible.

Next read the completion-facing handoffs. The uniform completion-filter row U and dense embedding row D preserve the same displayed source while the finite window row W and regular rational readback row R provide the observable data used by completion consumers.

Finally the Real seal row E is downstream of F, L, O, U, D, W, R. The transport, replay, provenance, and local naming rows preserve the same finite route, so no quotient filter, selected representative, countable-choice schedule, hidden cofinal subfilter, host equality, or ambient completeness theorem can be exported by the local certificate. □

Theorem 7289.3 (Bishop completion-filter dense completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCompletionFilter}^{\uparrow}$ *packet factors through*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{LocatedCauchyFilter}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionObject}^{\uparrow} \longmapsto \mathsf{UniformSpaceCompletionFilter}^{\uparrow} \vdash$$

The route is a finite Bishop-style completion-filter NameCert route; it does not export quotient completion equality, a selected limit representative, countable choice, host filter equality, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7289.2. They require $F, L, O, U, D, W, R, E, H, C, P, N$ to be read as one packet before a completion-facing consumer may inspect the endpoint.

Use the located and object rows. The located filter row L keeps the completion request attached to finite basis-refinement data, while the regular completion-object row O names the object-level completion carrier without adding a quotient representative.

Apply the uniform completion-filter and dense embedding rows. They route the same source through uniform entourage data and dense-image approximation before the StreamName and RegSeqRat rows expose the finite readback.

The terminal Real seal is then only a downstream seal over that route. Since the carrier has no excluded coordinate, every completion-facing read remains inside the displayed Bishop completion-filter packet. □

Falsifiable prediction: An accepted $\mathsf{BishopCompletionFilter}^{\uparrow}$ read is refuted if a completion-facing consumer reaches a Real seal or dense embedding handoff without first displaying the Cauchy filter row, located filter row, regular completion-object row, uniform completion-filter row, finite StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming rows.

Independence witness: cauchyfilter, locatedcauchyfilter, regularcauchycompletionobject, uniformspacecompletionfilter, denseembedding, real, regseqrat, streamname: none is bijective with $\mathsf{BishopCompletionFilter}^{\uparrow}$ because this packet jointly carries the Bishop-style filter source, located refinement, regular object, uniform completion route, dense image, finite windows, readback, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionFilter}^{\uparrow}$ packet is generated from CauchyFilter, LocatedCauchyFilter, RegularCauchyCompletionObject, UniformSpaceCompletionFilter, DenseEmbedding, StreamName, RegSeqRat, Real, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7289.1, Definition 7289.2, and Definition 7289.3 seed the Bishop-style completion-filter route over Cauchy filters, located filters, regular completion objects, uniform completion filters, dense embeddings, finite windows, regular readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host filter equality, hidden cofinal subfilters, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-filter exposure, regular completion-object admission, uniform completion-filter handoff, dense embedding route exposure, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionFilterUp.

Chapter 7290

A Concrete Naming Certificate for $\mathsf{NestedIntervalPrinciple}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ packet records the finite nested-interval principle as a displayed route from dyadic interval refinement data to regular rational names and the terminal real-name seal. It sits after the finite tree ledger of **??**, the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the explicit modulus route of chapter 5548. The packet names the finite L10 handoff for nested closed intervals; it is not selected limits, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7290.1 (Nested interval principle carrier)**.** *A* $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, D, W, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{DyadicIntervalTree}^{\uparrow}$ *source row, carrying the finite nested interval tree and its visible branch-depth ledger.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and containment row, recording the dyadic shrinkage budget used by every interval read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row extracted from the displayed nested tree and dyadic budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after T, D, W, R have been displayed.*
6. *M is the* $\mathsf{CantorIntersectionModulus}^{\uparrow}$ *handoff row that records the explicit modulus by which nested intervals become regular readback data.*
7. *H records componentwise* hsame *transport for the tree, dyadic budget, stream windows, regular readback, real seal, modulus handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from nested interval requests through T, D, W, R, E, M, H.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalPrinciple}^{\uparrow}$, $\mathsf{DyadicIntervalTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CantorIntersectionModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for selected limits, open-cover compactness, countable choice, quotient real equality, host interval intersection, or ambient Real completeness.

Theorem 7290.2 (Nested interval principle NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ *carrier* $J = (T, D, W, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval tree, dyadic budget, finite windows, regular rational readback, terminal real seal, modulus handoff, transport, replay, provenance, and local naming rows; tree-depth exactness through* T*; dyadic shrinkage exactness through* D*; window generation through* W*; regular readback through* R*; Real sealing through* E*; Cantor-modulus handoff through* M*; and structural preservation through* H, C, P, N*.*

Proof. Project the carrier of Definition 7290.1. The analytic rows are T, D, W, R, E, M, and the structural rows are H, C, P, N.

The tree row T exposes only a finite nested interval ledger. The dyadic row D supplies the diameter and containment budget used by that ledger, so no read can appeal to an ambient intersection theorem.

The window and readback rows W, R turn the displayed finite data into regular rational observations. The real seal E is terminal and may be read only after those rows and the modulus row M have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no excluded object appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7290.3 (Nested interval principle Real-seal handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ *carrier must read the finite route*

$$T, D \longmapsto W, R \longmapsto E, M, H, C, P, N$$

before it can consume the terminal $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports dyadic interval tree data, dyadic shrinkage budgets, finite StreamName windows, RegSeqRat readback, Cantor-intersection modulus data, transport, replay, provenance, and local naming; it exports no selected limit, open-cover compactness theorem, countable-choice schedule, quotient real representative, host interval intersection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the source rows. The interval evidence is carried by the displayed finite tree T, and the shrinkage evidence is carried by the dyadic budget row D.

Next read the completion-facing rows. The finite windows W are generated from the same tree and budget, and the regular rational readback R records the observations that the terminal real seal E may consume.

The modulus row M is the only Cantor-intersection handoff in the packet. It records an explicit route into regular readback data rather than a completeness principle.

Transport, replay, provenance, and local naming preserve the same rows. Thus every Real-facing consumer receives exactly the displayed finite route and none of the refused selected-limit, compactness, choice, quotient, host-intersection, or ambient-completeness data. □

Falsifiable prediction: An accepted $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the rows $T, D, W, R, E, M, H, C, P, N$ refutes the carrier surface.

Independence witness: dyadicintervaltree, dyadicratcore, streamname, regseqrat, real, cantor-intersection-modulus: no listed sibling supplies the joint packet containing a finite dyadic interval

tree, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, Cantor-intersection modulus handoff, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalPrinciple}^{\uparrow}$ packet is generated from a finite dyadic interval tree, dyadic shrinkage budget, StreamName windows, RegSeqRat readback, terminal Real seal, Cantor-intersection modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7290.1, Definition 7290.2, and Definition 7290.3 seed the finite nested-interval route into Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, open-cover compactness, countable choice, quotient real equality, host interval intersection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{NestedIntervalPrinciple}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tree-depth exactness, dyadic shrinkage exactness, window generation, regular readback, Real sealing, Cantor-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalPrincipleUp.

Chapter 7291

A Concrete Naming Certificate for $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$

The $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ packet realizes the MetaCIC confluence audit map as finite residual-critical-pair data. It sits after the critical-path source of chapter 765, the normalization frontier of chapter 892, the parallel-diamond frontier of chapter 4238, the confluence audit packet of chapter 735, and the normal-form consistency seal of chapter 740. The packet records residual-substitution, local-join, closed-checker, obstruction, transport, replay, provenance, and NameCert rows; it does not claim global confluence, substitution compatibility, closed Church–Rosser, subject reduction, quotient syntax equality, or a hidden reduction oracle.

Definition 7291.1 (MetaCIC residual critical-pair carrier)**.** *A* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, P, J, D, O, T, R, V, H, C, G, N)$$

with the following displayed rows:

1. *S is the residual-substitution row, naming the substitution-sensitive residual window that must be displayed before a peak can be consumed.*
2. *P is the local peak row, recording the two one-step residual branches and their shared typed source window.*
3. *J is the candidate local-join row, naming the finite common-reduct attempt associated to the displayed peak.*
4. *D is the closed-checker row, restricting the read to the closed or shape-limited fragment currently admitted by the audit map.*
5. *O is the obstruction row, preserving any missing residual-substitution compatibility, parallel diamond, or closed-confluence input as visible boundary data.*
6. *T is the transport row carrying componentwise* hsame *comparisons for $S, P, J, D, O, R, V, G, N$.*
7. *R is the replay row, recording displayed* Cont *replay from residual-substitution inspection through local join checking and obstruction retention.*
8. *V is the verification-frontier row tying the packet to the critical path, normalization frontier, parallel-diamond frontier, confluence audit packet, and normal-form consistency seal.*

9. *H is the audit-history row preserving source labels, shape bounds, peak identifiers, local-join attempts, and obstruction names.*
10. *C is the closed-fragment handoff row consumed by closed-normal or shape-limited confluence readers after S, P, J, D, O have been displayed.*
11. *G is the* Pkg *provenance over* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}$ $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPair}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose residual-substitution, local peak, candidate join, closed-checker, obstruction, transport, replay, frontier, history, provenance, and local naming rows agree componentwise. It has no coordinate for a global confluence theorem, arbitrary substitution compatibility, a hidden parallel diamond, subject reduction, quotient syntax equality, or off-packet oracle joins.

Theorem 7291.2 (MetaCIC residual critical-pair NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ *carrier $K = (S, P, J, D, O, T, R, V, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalP}}$ *surface consists exactly of carrier admission for the displayed rows; residual-substitution admission through S; local peak admission through P; candidate join admission through J; closed or shape-limited checker admission through D; obstruction retention through O; componentwise transport through T; displayed replay through R; critical-path, normalization-frontier, parallel-diamond-frontier, confluence-audit, and normal-form-seal anchoring through V, H, C, G; and local naming through N.*

Proof. Project the carrier of Definition 7291.1. The confluence-facing rows are S, P, J, D, O, and the structural rows are T, R, V, H, C, G, N.

The residual-substitution row S is read before the local peak P and candidate join J. This keeps the packet inside the substitution-sensitive audit map instead of treating a local join as an unconditional diamond.

The closed-checker row D and obstruction row O are then inspected. They separate the available closed or shape-limited fragment from the missing residual-substitution compatibility, parallel-diamond, and closed-confluence inputs.

Finally T, R, V, H, C, G, N transport, replay, anchor, preserve, hand off, source, and name the same finite route. Since the carrier contains no global confluence, subject-reduction, quotient-syntax, or hidden-oracle coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7291.3 (MetaCIC residual critical-pair frontier handoff)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ *packet receives the finite route*

$$S, P, J, D, O, T, R, V, H, C, G, N.$$

The route may be handed to $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}$ $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *consumers only as residual-critical-pair evidence with an explicit obstruction boundary. It does not export global confluence, arbitrary residual-substitution compatibility, an unconditional parallel diamond, closed Church–Rosser, subject reduction, quotient syntax equality, or a hidden reduction oracle.*

Proof. Start with the residual-substitution and local-peak rows. The packet exposes S before P, J, so the consumer sees the substitution-sensitive residual window together with the candidate join attempt.

Read the closed-checker and obstruction rows next. The row D confines the consumer to the closed or shape-limited fragment, while O names the missing compatibility, parallel-diamond, or closed-confluence input that remains outside the carrier.

The structural rows T, R, V, H, C, G, N then preserve the same route for the critical-path, normalization-frontier, parallel-diamond, confluence-audit, and normal-form-seal consumers. The handoff is therefore residual-critical-pair evidence with a visible obstruction boundary, not a confluence theorem. □

Theorem 7291.4 (MetaCIC residual critical-pair obstruction preservation)**.** *Let $K = (S, P, J, D, O, T, R, V, H, C, G$ be an accepted* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ *packet. Any handoff from K to a confluence-facing consumer preserves the obstruction row O until residual-substitution compatibility and parallel-diamond inputs are displayed through the same residual-critical-pair route. In particular, the rows $S, P, J, D, T, R, V, H, C, G, N$ may transport and replay the local peak and candidate join, but they cannot erase O or convert the packet into a global confluence export.*

Proof. Begin with the frontier handoff of Definition 7291.3. It exports the route $S, P, J, D, O, T, R, V, H, C, G, N$ with the obstruction boundary still visible.

The residual-substitution row S, local peak row P, and candidate join row J expose only the local critical-pair data. They do not contain the missing residual-substitution compatibility or parallel-diamond input named by O.

The closed-checker row D then restricts the admissible fragment, while T, R, V, H, C, G, N preserve, replay, anchor, hand off, source, and name the same displayed route. Since O is one of the displayed rows transported by T and replayed by R, every confluence-facing consumer must keep it until the missing inputs are separately displayed. □

Theorem 7291.5 (MetaCIC residual critical-pair closed-checker monotonicity)**.** *For an accepted* $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ *packet $K = (S, P, J, D, O, T, R, V, H, C, G, N)$, strengthening the closed or shape-limited checker row D can only restrict the admitted local-join route. It may remove a candidate read of S, P, J from the closed-fragment handoff C, but it cannot create residual-substitution compatibility, erase the obstruction row O, produce a parallel diamond, or export global confluence.*

Proof. Use the frontier handoff of Definition 7291.3 as the ambient route. The consumer receives $S, P, J, D, O, T, R, V, H, C, G, N$, not an unrestricted reduction relation or a global confluence theorem.

Strengthening D narrows the closed or shape-limited fragment inspected by the packet. Therefore any local-join read through J that remains admitted still passes through the same residual-substitution row S, local peak row P, obstruction row O, and structural rows T, R, V, H, C, G, N.

If a candidate route is excluded by the stronger checker, it simply leaves the handoff surface. If it remains, the obstruction row is still transported and replayed with the route, so the stronger checker cannot manufacture compatibility, a parallel diamond, subject reduction, quotient syntax equality, or global confluence. □

Falsifiable prediction: An accepted $\mathsf{MetaCICResidualCriticalPair}^{\uparrow}$ consumer that reaches a confluence-facing handoff without displaying residual-substitution row S, local peak row P, candidate join row J, closed-checker row D, obstruction row O, and structural rows T, R, V, H, C, G, N refutes the carrier surface.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-parallel-diamond-frontier, metacicconfluenceauditpacket, normalformconsistencyseal: no listed sibling is bijective to

MetaCICResidualCriticalPair$^{\uparrow}$ because this packet jointly carries residual-substitution windows, local peaks, candidate joins, closed-checker rows, obstruction retention, transport, replay, audit history, closed-fragment handoff, provenance, and local naming. The Lean TasteGate follow-up must provide Nontrivial and FieldFaithful instances for the packet surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPair}^{\uparrow}}$.

Constructive story: A MetaCICResidualCriticalPair$^{\uparrow}$ packet is generated from residual-substitution windows, local peaks, candidate joins, closed-checker rows, obstruction rows, componentwise hsame transport, displayed Cont replay, audit-frontier anchoring, history preservation, closed-fragment handoff, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7291.1, Definition 7291.2, Definition 7291.3, Definition 7291.4, and Definition 7291.5 seed the finite residual critical-pair route for the MetaCIC confluence audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, arbitrary residual-substitution compatibility, unconditional parallel diamond, closed Church–Rosser, subject reduction, quotient syntax equality, Lean verification, or bridge checking for MetaCICResidualCriticalPair$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, residual-substitution admission, local peak admission, candidate join admission, closed-checker admission, obstruction retention, transport, replay, audit-frontier anchoring, history preservation, closed-fragment handoff, provenance, and local naming for BEDC.Derived.MetaCICResidualCriticalPairUp.

Chapter 7292

A Concrete Naming Certificate for BishopCompactIntervalCauchyCompleteness$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopCompactIntervalCauchyCompleteness$^{\uparrow}$ packet records compact-interval Cauchy completeness as finite BEDC NameCert data. It reads the nested-interval completion surface of chapter 6091, the Bishop complete-metric packet of chapter 3424, the compact-metric source of chapter 129, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dyadic tolerance core of chapter 291. The packet names the classical compact-interval Cauchy route without turning it into global Real$^{\uparrow}$ completeness, a selected limit operator, quotient equality of Cauchy names, trichotomy, or countable choice.

Definition 7292.1 (Bishop compact-interval Cauchy-completeness carrier)**.** *A* BishopCompactIntervalCauchyComp *carrier is a finite* BHist *packet*

$$K = (I, S, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, read through the* CompactMetric$^{\uparrow}$ *and* BishopCompleteMetric$^{\uparrow}$ *surfaces for the displayed closed bounded interval.*
2. *S is the* StreamName$^{\uparrow}$ *Cauchy stream row, storing only the inspected finite windows of the sequence in the compact interval.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *dyadic mesh row, recording the tolerance and interval-mesh comparisons requested by the compactness argument.*
4. *L is the nested interval lock, routing the Cauchy stream and dyadic mesh through the* NestedIntervalCompleteness$^{\uparrow}$ *shrinking-chain surface.*
5. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback obtained after the stream windows and nested lock have been displayed.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after I, S, D, L, R are read in the same finite route.*
7. *H records componentwise* hsame *transport for the compact interval source, Cauchy stream, dyadic mesh, nested lock, readback, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy tolerance request through D, S, L, R, E.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}$, $\mathsf{NestedIntervalCompleteness}^{\uparrow}$, $\mathsf{BishopCompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}}$ *row.*

The carrier has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient equality of Cauchy names, trichotomy, countable choice, hidden stream equality, or host interval compactness.*

Theorem 7292.2 (Bishop compact-interval Cauchy-completeness obligations)**.** *For every accepted* $\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}$ *carrier* $K = (I, S, D, L, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact interval admission for the displayed interval source I through the compact-metric and Bishop complete-metric sibling packets;*
2. *Cauchy stream visibility, requiring each stream window consumed by the argument to appear as a displayed row of S;*
3. *dyadic mesh visibility, requiring every tolerance and endpoint comparison to pass through the displayed row D;*
4. *nested interval locking, requiring L to bind the Cauchy stream and mesh rows to the shrinking interval route before any limit readback is exposed;*
5. *regular rational readback, requiring R to be read only after I, S, D, L are displayed;*
6. *terminal Real sealing, requiring E to factor through the same compact interval, stream, mesh, nested lock, and readback rows;*
7. *transport, replay, provenance, and local naming stability through H, C, P, N;*
8. *ledger nonescape from ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient equality of Cauchy names, trichotomy, countable choice, hidden stream equality, and host interval compactness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7292.1. The accepted packet displays exactly the rows $I, S, D, L, R, E, H, C, P, N$.

The Cauchy and mesh obligations are read before the lock. The row S exposes only finite stream windows, and the row D exposes only the dyadic tolerance and endpoint comparisons needed to place those windows in a displayed interval mesh.

The nested lock L then binds the displayed windows and mesh rows to the $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ route. After that lock is visible, R may read the regular rational name and E may seal the same route as a $\mathsf{Real}^{\uparrow}$-facing row.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no additional coordinate, the excluded principles cannot appear as local obligations. □

Theorem 7292.3 (Bishop compact-interval Cauchy limit route)**.** *Every accepted* BishopCompactIntervalCauchyCom *carrier* $K = (I, S, D, L, R, E, H, C, P, N)$ *hands a completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer the ordered finite route*

$$I \longrightarrow S \longrightarrow D \longrightarrow L \longrightarrow R \longrightarrow E.$$

The consumer first reads the compact interval source, then the Cauchy stream windows, then the dyadic mesh comparisons, then the nested interval lock, then the regular rational readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The route exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, selected limit, quotient equality of Cauchy names, trichotomy datum, countable-choice selector, or host interval compactness principle.*

Proof. Begin with the accepted carrier $K = (I, S, D, L, R, E, H, C, P, N)$. Projection supplies the compact interval row I, Cauchy stream row S, dyadic mesh row D, nested lock L, regular readback row R, and terminal seal E.

The compact interval source is read first because the stream windows and dyadic comparisons are interpreted inside that displayed interval. The rows S and D then expose exactly the finite Cauchy and mesh data consumed by the nested lock.

Once L is displayed, the route has the shrinking interval evidence needed for the regular rational readback R. The seal E is downstream of that readback, and the rows H, C, P, N keep the same route stable under transport, replay, provenance, and local naming.

Nothing in this ordered projection adds another coordinate. The consumer therefore receives the finite compact-interval Cauchy route and none of the excluded global or choice-based principles. □

Theorem 7292.4 (Bishop compact-interval Cauchy nonescape)**.** *For every accepted* $\mathsf{BishopCompactIntervalCauchyC}$ *carrier* $K = (I, S, D, L, R, E, H, C, P, N)$*, every public read of a Cauchy limit in the compact interval factors through* $I, S, D, L, R, E, H, C, P, N$*. In particular the packet explicitly refuses ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient equality of Cauchy names, trichotomy, and countable choice; none of these may be recovered from the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}}$ *surface.*

Proof. The public read surface is the carrier surface. By Definition 7292.2, the local NameCert obligations are exhausted by the compact interval source, stream windows, dyadic mesh, nested lock, regular readback, terminal seal, and the structural rows H, C, P, N.

By Definition 7292.3, any completion-facing or Real-facing limit read must pass through the ordered route I, S, D, L, R, E before the structural rows are consulted. The route is finite and displayed at each step.

Ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient equality of Cauchy names, trichotomy, and countable choice are not rows of the carrier and are named as excluded coordinates in Definition 7292.1. Hence no public read of the packet can export them. □

Falsifiable prediction: An accepted $\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}$ limit read is refuted if a completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer reaches the terminal seal without displaying the compact interval source, Cauchy stream windows, dyadic mesh, nested interval lock, regular rational readback, transport, replay, provenance, and local naming rows.

Independence witness: nested-interval-completeness, bishop-complete-metric, compactmetric, streamname, regseqrat, real, dyadicratcore: none alone is bijective with $\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}$ because the packet jointly records the compact interval source, Cauchy stream, dyadic mesh, nested lock, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCompactIntervalCauchyCompleteness}^{\uparrow}$ packet is generated from a compact interval source, finite Cauchy stream windows, dyadic mesh comparisons, a nested interval lock, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopCompactIntervalCauchyCompleteness}\uparrow}$, seed)

Scope closed: Definition 7292.1, Definition 7292.2, Definition 7292.3, and Definition 7292.4 seed the finite compact-interval Cauchy-completeness certificate over compact metric, nested interval, stream, dyadic, regular-rational, and Real-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopCompactIntervalCauchyCompleteness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient Real$^{\uparrow}$ completeness, selected limits, quotient equality of Cauchy names, trichotomy, countable choice, host interval compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval admission, Cauchy stream visibility, dyadic mesh visibility, nested interval locking, regular rational readback, terminal Real sealing, transport and replay stability, and ledger nonescape for BEDC.Derived.BishopCompactIntervalCauchyCompletenessUp.

Chapter 7293

A Concrete Naming Certificate for $\mathsf{UrysohnRationalMetric}^{\uparrow}$

Following Urysohn and Katetov on finite metric extensions, the $\mathsf{UrysohnRationalMetric}^{\uparrow}$ packet records the rational finite one-point extension substrate before any universal metric-space carrier is selected. It sits over the metric carrier of chapter 57, the separable-metric source of chapter 1458, the complete-metric boundary of chapter 128, the completion handoff of chapter 1506, the terminal real-distance seal of chapter 38, and the larger Urysohn finite-extension packet of chapter 6554. The packet isolates rational support distances, rational requested distances, and finite Katetov inequalities as NameCert rows without constructing a host Urysohn space, a completion quotient, a dense extension chain, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7293.1 (Urysohn rational metric carrier)**.** *A* $\mathsf{UrysohnRationalMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, D, F, A, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *source row whose finite support points and support distances are inspected.*
2. *S is the* $\mathsf{SeparableMetric}^{\uparrow}$ *finite support schedule naming exactly the support points used by the extension request.*
3. *D is the rational distance table row. It records the already displayed support distances and the proposed rational distance from the new point to each support point.*
4. *F is the finite rational Katetov request row. It lists the triangle and reverse-triangle inequalities checked for each displayed support pair.*
5. *A is the admission ledger row. It records that every inequality in F is verified using only the rational table D and the source metric row M.*
6. *E is the one-point extension handoff row. It allows a consumer to read the rational extension request after M, S, D, F, A have been exposed.*
7. *H records componentwise* hsame *transport for the source metric, support schedule, rational table, Katetov request, admission ledger, extension handoff, replay, provenance, and local name.*
8. *C records displayed* Cont *replay for the same finite request and admission rows.*
9. *P is the* Pkg *provenance over* $\mathsf{UrysohnRationalMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{SeparableMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, rational finite metric tables,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UrysohnRationalMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, D, F, A, E, H, C, P, N$. It has no coordinate for a host universal metric space, a selected completion point, quotient completion equality, an arbitrary countable extension schedule, choice, Lean verification, a standard bridge, or ambient real completeness.

Theorem 7293.2 (Urysohn rational metric NameCert obligations)**.** *For every accepted* $\mathsf{UrysohnRationalMetric}^{\uparrow}$ *carrier*

$$Q = (M, S, D, F, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UrysohnRationalMetric}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the metric source, finite support schedule, rational distance table, finite Katetov request, admission ledger, extension handoff, transport, replay, provenance, and local name;*
2. *common-source discipline, requiring S, D, F, A to be read over the same metric source row M;*
3. *rational support visibility, requiring every distance in D to be attached to a support point named by S;*
4. *Katetov inequality exposure, requiring F and A to display each triangle and reverse-triangle budget over the rational table;*
5. *extension-handoff order, requiring E to be consumed only after M, S, D, F, A have been exposed;*
6. *nonescape, excluding host Urysohn spaces, quotient completion equality, arbitrary extension chains, selected completion points, countable choice, Lean verification, bridge credentials, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Urysohn rational metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 7293.1. The tuple $M, S, D, F, A, E, H, C, P, N$ is the whole public surface, so carrier admission is exactly admission of those rows.

The common-source and support-visibility obligations follow from M, S, D. The support schedule names the finite points, and the rational distance table is read only over those points and the same source metric row.

The Katetov obligation is recorded by F and A. The request row lists the triangle and reverse-triangle comparisons, while the admission ledger records that each comparison has been checked against the rational table.

Finally E, H, C, P, N preserve only that displayed finite route. Since none of the carrier coordinates stores a host universal space, quotient completion equality, countable extension chain, selected completion point, choice functional, Lean credential, bridge credential, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 7293.3 (Urysohn rational metric one-point extension)**.** *Every finite rational one-point extension request accepted by a* $\mathsf{UrysohnRationalMetric}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{SeparableMetric}^{\uparrow} \longmapsto D, F, A \longmapsto E.$$

The request is therefore a displayed rational Katetov route over the current finite support, not a point selected from a universal metric space, a quotient class in a completed space, or an ambient $\mathsf{Real}^{\uparrow}$ *limit.*

Proof. Begin with the metric source row M and finite support schedule S. They expose the only support points and support distances available to the extension request.

Read the rational table D next. It supplies the proposed rational distances from the new point to the displayed support and records the already visible support-distance comparisons.

The rows F and A then check the finite Katetov inequalities. Each compatibility budget is a rational row attached to the displayed support table, so no hidden real-completion datum is introduced.

The handoff row E exports only this finite rational route. Host universal metric spaces, quotient completion representatives, selected completion points, arbitrary countable extension chains, choice functionals, and ambient Real completeness remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{UrysohnRationalMetric}^{\uparrow}$ extension read exposes the $\mathsf{Met}^{\uparrow}$ or $\mathsf{Met}^{\uparrow}$ source, $\mathsf{SeparableMetric}^{\uparrow}$ support schedule, rational distance table, finite Katetov request, and admission ledger before any $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer may inspect the handoff row.

Independence witness: metric, separablemetric, completemetric, metriccompletion, real, urysohnuniversalmetric: $\mathsf{UrysohnRationalMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite rational support tables, finite Katetov inequality rows, admission ledgers, one-point extension handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows while refusing host universal spaces and completion quotients.

Closure status for $\mathsf{NameCert}_{\mathsf{UrysohnRationalMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{UrysohnRationalMetric}^{\uparrow}$ packet is generated from a displayed metric source, finite separable support schedule, rational distance table, finite Katetov request, admission ledger, one-point extension handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UrysohnRationalMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7293.1, Definition 7293.2, and Definition 7293.3 seed the finite rational one-point metric extension substrate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UrysohnRationalMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Urysohn-space existence, quotient completion equality, countable choice, arbitrary extension chains, selected completion points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-source discipline, rational support visibility, Katetov inequality exposure, handoff order, and nonescape for BEDC.Derived.UrysohnRationalMetricUp.

Chapter 7294

A Concrete Naming Certificate for DirichletAbelSummability$^{\uparrow}$

Following Hardy's summability treatment and the Bishop–Bridges constructive reading of real series, the DirichletAbelSummability$^{\uparrow}$ packet records the finite handoff in which a displayed real series exposes both Abel-test and Dirichlet-test faces before a completion-facing summability read is allowed to reach a uniform Cauchy route. It sits over the RealSeries$^{\uparrow}$ source of chapter 1581, the AbelTest$^{\uparrow}$ face of chapter 1496, the DirichletTest$^{\uparrow}$ face of chapter 1497, the UniformCauchySeries$^{\uparrow}$ handoff of chapter 1736, the RegSeqRat$^{\uparrow}$ readback of chapter 301, and the terminal Real$^{\uparrow}$ seal of chapter 38. The packet is a Real-and-Completion summability boundary; it does not import host summability, select an infinite sum, quotient series names, invoke countable choice, or assert ambient Real$^{\uparrow}$ completeness.

Definition 7294.1 (Dirichlet–Abel summability carrier)**.** *A* DirichletAbelSummability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, D, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* RealSeries$^{\uparrow}$ *source row, exposing the real-valued term and partial-sum observations only through the series packet.*
2. *A is the* AbelTest$^{\uparrow}$ *face row, recording the Abel-style monotone coefficient and tail-control evidence available for the same source.*
3. *D is the* DirichletTest$^{\uparrow}$ *face row, recording bounded partial-sum data, null-tail data, and the finite Dirichlet estimate schedule for the same source.*
4. *U is the shared* UniformCauchySeries$^{\uparrow}$ *handoff row reached only after the Abel and Dirichlet faces have both displayed their finite evidence.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the common summability windows.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row available only after S, A, D, U, R are visible.*
7. *H records componentwise* hsame *transport for $S, A, D, U, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a summability request through the source, the two test faces, the uniform handoff, readback, seal, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* DirichletAbelSummability$^{\uparrow}$, RealSeries$^{\uparrow}$, AbelTest$^{\uparrow}$, DirichletTest$^{\uparrow}$, UniformCauchySeries$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DirichletAbelSummability}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through these rows. It has no coordinate for host convergence of the series, selected infinite sums, quotient equality of series names, countable choice, detached Abel or Dirichlet witnesses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7294.2 (Dirichlet–Abel summability NameCert obligations)**.** *For every accepted* DirichletAbelSummabilit *carrier*

$$D = (S, A, D, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DirichletAbelSummability}^{\uparrow}}$ *surface consists exactly of RealSeries source admission, Abel-test face exposure, Dirichlet-test face exposure, shared UniformCauchySeries handoff, RegSeqRat readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7294.1. The analytic rows are S, A, D, U, R, E, and the structural rows are H, C, P, N.

Read the source and the two test faces first. The RealSeries row S fixes the displayed term and partial-sum data, while A and D record the Abel and Dirichlet evidence for that same source.

Next read the completion-facing handoff. The row U is shared by the two faces, so a consumer cannot reach the uniform Cauchy surface from only one detached test witness. The readback row R and terminal seal row E remain downstream of the shared handoff.

Finally the transport, replay, provenance, and local naming rows preserve the same finite route. Since no excluded object is a carrier coordinate, the local certificate cannot export host summability, a selected infinite sum, quotient series equality, countable choice, or ambient real completeness. □

Theorem 7294.3 (Dirichlet–Abel L10 summability handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{DirichletAbelSummability}^{\uparrow}$ *packet factors through*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{AbelTest}^{\uparrow} \longmapsto \mathsf{DirichletTest}^{\uparrow} \longmapsto \mathsf{UniformCauchySeries}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route is a finite L10 handoff from two convergence-test faces to one completion-facing readback; it does not export a host series-summability theorem, selected infinite sum, quotient equality of series names, countable-choice schedule, detached test witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7294.2. They require the rows $S, A, D, U, R, E, H, C, P, N$ to be read as one packet before a Real-and-Completion consumer inspects the endpoint.

The Abel and Dirichlet faces are both attached to the same RealSeries source. Thus the consumer first reads S, then the Abel face A and the Dirichlet face D as sibling evidence for the same displayed series.

The shared uniform handoff U is downstream of both faces. Once U is visible, the regular rational readback R and the Real seal E can be consumed as completion-facing data, but neither row supplies an independent summability theorem or selected limit.

The structural rows H, C, P, N transport, replay, source, and name the same route. Therefore every L10 read remains inside the displayed summability packet. □

Falsifiable prediction: An accepted $\mathsf{DirichletAbelSummability}^{\uparrow}$ read is refuted if a Real-and-Completion consumer reaches the $\mathsf{UniformCauchySeries}^{\uparrow}$ handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, or $\mathsf{Real}^{\uparrow}$ seal without first displaying the $\mathsf{RealSeries}^{\uparrow}$ source together with both the $\mathsf{AbelTest}^{\uparrow}$ and $\mathsf{DirichletTest}^{\uparrow}$ face rows.

Independence witness: realseries, abel-test, dirichlet-test, uniform-cauchy-series, regseqrat, real: none is bijective with $\mathsf{DirichletAbelSummability}^{\uparrow}$ because this packet jointly carries a RealSeries source, Abel and Dirichlet test faces, one shared UniformCauchySeries handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletAbelSummability}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletAbelSummability}^{\uparrow}$ packet is generated from RealSeries source data, AbelTest face data, DirichletTest face data, a shared UniformCauchySeries handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletAbelSummability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7294.1, Definition 7294.2, and Definition 7294.3 seed the Real-and-Completion summability handoff from Abel and Dirichlet test faces to one uniform Cauchy readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletAbelSummability}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DirichletAbelSummabilityUp`

Bridge status: none

Not claimed: This chapter does not close host series summability, selected infinite sums, quotient equality of series names, countable choice, detached Abel or Dirichlet witnesses, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Abel-test exposure, Dirichlet-test exposure, shared UniformCauchySeries handoff, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DirichletAbelSummabilityUp.

Chapter 7295

A Concrete Naming Certificate for $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$

Following Arzela, Ascoli, Kelley, and Bishop–Bridges, the $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ packet records the finite-subfamily surface by which compact-metric finite nets, bounded-family ledgers, and equicontinuity rows are inspected before an Arzela–Ascoli consumer may read compactness-facing data. It sits after the Arzela–Ascoli packet of chapter 412, the equicontinuity-family surface of chapter 413, the bounded-family packet of chapter 3522, the compact-metric source of chapter 129, and the finite epsilon-net rows of chapter 1767. The packet names only the displayed finite subfamily and net-selection readback; it does not claim full Arzela–Ascoli compactness, selected subsequences, quotient function spaces, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7295.1 (Arzela–Ascoli finite-subfamily carrier)**.** *An* $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, F, B, E, Q, W, R, Y, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the compact-metric finite-net schedule on which the subfamily will be inspected.*
2. *F is the finite function-family row. It lists the selected family members as displayed* BHist *entries rather than as a host function space.*
3. *B is the bounded-family ledger read through* $\mathsf{BoundedFunctionFamily}^{\uparrow}$*, recording the pointwise or sup-bound rows needed by each selected member.*
4. *E is the equicontinuity ledger read through* $\mathsf{EquicontinuityFamily}^{\uparrow}$*, exposing the shared finite modulus requests used by the selected members.*
5. *Q is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, recording the finite source-net centers, tolerance rows, and coverage checks used for the selection surface.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite inspection windows and rational observations.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row attached to the displayed readback, not a global completeness principle.*

8. *Y is the Arzela–Ascoli handoff row, read only by* $\mathsf{ArzelaAscoli}^{\uparrow}$ *or* $\mathsf{AscoliDiagonal}^{\uparrow}$ *consumers after X, F, B, E, Q, W, R are present.*
9. *H records componentwise* hsame *transport of the compact source, finite family, boundedness, equicontinuity, finite-net, readback, seal, and handoff rows.*
10. *C records displayed* Cont *replay of the same finite route.*
11. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{AscoliDiagonal}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BoundedFunctionFamily}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, B, E, Q, W, R, Y, H, C, P, N$. It has no coordinate for a host compact-open topology, an arbitrary subsequence selector, quotient equality of functions, open-cover compactness, countable choice, or ambient Real completeness.

Theorem 7295.2 (Arzela–Ascoli finite-subfamily NameCert obligations)**.** *For every accepted* $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ *carrier $S = (X, F, B, E, Q, W, R, Y, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteSubfami}}$ *surface consists exactly of compact-source admission through X, finite-family admission through F, boundedness admission through B, equicontinuity admission through E, finite epsilon-net admission through Q, StreamName and RegSeqRat readback through W, terminal Real sealing through R, Arzela–Ascoli or Ascoli-diagonal handoff through Y, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7295.1. The analytic rows are exactly X, F, B, E, Q, W, R, Y, and the structural rows are exactly H, C, P, N.

The source and family rows are read first. Row X supplies the compact-metric finite-net schedule, row F supplies the selected finite family, row B supplies the bounded-family ledger, and row E supplies the shared equicontinuity ledger.

The finite inspection rows are downstream from those ledgers. Row Q exposes the finite epsilon-net centers and tolerance checks, row W exposes only finite StreamName windows and RegSeqRat observations, and row R seals that readback as a displayed Real-facing row.

The handoff row Y is therefore a consumer boundary, not a compactness theorem. Rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local NameCert surface. □

Theorem 7295.3 (Arzela–Ascoli finite-subfamily handoff)**.** *Every* $\mathsf{ArzelaAscoli}^{\uparrow}$ *or* $\mathsf{AscoliDiagonal}^{\uparrow}$ *consumer of an accepted* $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ *carrier must read the finite subfamily row F, the finite-net row Q, the equicontinuity row E, the boundedness row B, and the Real-seal row R before any compactness-facing conclusion. The handoff factors through*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{BoundedFunctionFamily}^{\uparrow},\ \mathsf{EquicontinuityFamily}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{R}$$

It exports no full Arzela–Ascoli compactness theorem, selected subsequence, quotient function space, open-cover compactness proof, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7295.2. It orders each consumer read through the compact source, finite family, boundedness, equicontinuity, finite epsilon-net, readback, Real seal, and handoff rows.

The finite subfamily and finite-net checks are visible before the consumer boundary. Thus an $\mathsf{ArzelaAscoli}^{\uparrow}$ or $\mathsf{AscoliDiagonal}^{\uparrow}$ consumer can inspect which family members and net centers were used, together with the boundedness and equicontinuity rows that constrain them.

The Real seal is also a displayed coordinate of the same packet. Since the refused compactness, subsequence, quotient-space, open-cover, choice, and ambient-completeness objects are not carrier coordinates, the handoff cannot export them. □

Falsifiable prediction: An accepted $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ handoff is refuted if an $\mathsf{ArzelaAscoli}^{\uparrow}$ or $\mathsf{AscoliDiagonal}^{\uparrow}$ consumer reaches a compactness-facing read without displaying the finite subfamily, finite epsilon-net, equicontinuity, boundedness, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local NameCert rows.

Independence witness: arzelaascoli, ascoli-diagonal, equicontinuityfamily, bounded-function-family, compactmetric, finite-epsilon-net: none alone is bijective with $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ because the packet jointly records the compact source, finite function family, bounded-family ledger, equicontinuity ledger, finite epsilon-net rows, StreamName and RegSeqRat readback, Real seal, handoff boundary, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}$ packet is generated from a compact source, finite function family, bounded-family ledger, equicontinuity ledger, finite epsilon-net rows, StreamName and RegSeqRat readback, a Real seal, an Arzela–Ascoli or Ascoli-diagonal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7295.1, Definition 7295.2, and Definition 7295.3 seed the finite-subfamily selection surface over compact-metric, bounded-family, equicontinuity, finite-net, StreamName, RegSeqRat, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteSubfamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Arzela–Ascoli compactness, selected subsequences, quotient function spaces, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, finite-family exposure, boundedness exposure, equicontinuity exposure, finite epsilon-net inspection, StreamName and RegSeqRat readback, Real-seal nonescape, handoff ordering, transport, replay, provenance, and local naming for BEDC.Derived.ArzelaAscoliFiniteSubfamilyUp.

Chapter 7296

A Concrete Naming Certificate for $\mathsf{WallmanCompactification}^{\uparrow}$

Following Wallman on compactification by closed-set lattices, the $\mathsf{WallmanCompactification}^{\uparrow}$ packet records the finite route by which a displayed normal lattice of closed rows names boundary filters and their compactification-facing topology. It sits beside the one-point compactification of chapter 2049, the Freudenthal end boundary of chapter 6396, the compact carrier of chapter 60, the indexed topology surface of chapter 88, and the Real/L10 readback route through chapter 38 and chapter 301. The packet names only the finite closed-set lattice and boundary-filter readback; it is not a host powerset of closed sets, a selected ultrafilter, a quotient compactification space, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7296.1 (Wallman compactification carrier)**.** *A* $\mathsf{WallmanCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, L, F, B, J, T, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Topology}^{\uparrow}$ *row, exposing the indexed open and closed-set membership reads available to the packet.*
2. *L is the finite closed-set lattice row. It records displayed finite meets, finite joins, inclusion comparisons, and the normal-separation ledger used by the Wallman route.*
3. *F is the boundary-filter row. Each displayed boundary request names a finite consistent family of closed rows from L together with its finite-refinement witness.*
4. *B is the compact-boundary admission row, read through the compact carrier of chapter 60 without importing an ambient compact host.*
5. *J is the source-inclusion row, sending a displayed source point to the finite closed-neighbourhood filter it determines.*
6. *T is the topology handoff row for the compactification carrier. It translates finite closed-lattice and boundary-filter requests into indexed topology reads.*
7. *R is the L10 readback row, routing finite boundary observations through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals when a real-valued consumer inspects the compactification.*
8. *H records componentwise* hsame *transport for $X, L, F, B, J, T, R, C, P, N$.*

9. *C records displayed* Cont *replay of closed-lattice, boundary-filter, topology, compactness, and readback requests.*
10. *P is the* Pkg *provenance over* $\mathsf{WallmanCompactification}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WallmanCompactification}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for a full host closed-set powerset, selected ultrafilter, quotient compactification point, open-cover compactness proof, countable-choice selector, or ambient complete real line.

Theorem 7296.2 (Wallman compactification NameCert obligations)**.** *For every accepted* WallmanCompactification *carrier*

$$K = (X, L, F, B, J, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WallmanCompactification}^{\uparrow}}$ *surface consists exactly of source topology admission, finite closed-lattice admission, boundary-filter consistency, compact-boundary admission, source-inclusion display, topology handoff, L10 readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compactification-facing read must expose X, L, F, J, T before B or R can be consumed.*

Proof. Project the carrier of Definition 7296.1. The only source data are the topology row X and the finite closed-lattice row L, so every boundary request begins with displayed closed-set membership and finite lattice operations.

The boundary-filter row F is then a finite consistency ledger over L. It records which closed rows are jointly requested and which finite refinement witnesses keep the request admissible, rather than selecting an ultrafilter or forming a quotient point.

Read the source-inclusion and topology handoff rows next. The row J ties displayed source points to their closed-neighbourhood filters, and T converts lattice and filter requests into indexed topology reads before any compact-boundary row is public.

Finally consume B, R, H, C, P, N. The compact-boundary and L10 readback rows are downstream of the same finite route, while transport, replay, provenance, and local naming add no host powerset, selected boundary point, countable-choice selector, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Theorem 7296.3 (Wallman compactification L10 boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reading a* $\mathsf{WallmanCompactification}^{\uparrow}$ *packet receives only the finite route*

$$X, L, F, J, T \longmapsto B, R, H, C, P, N.$$

Thus an L10 readback may inspect finite closed-set lattice requests, boundary-filter consistency, source inclusion, topology handoff, compact-boundary admission, and the real-valued readback row, but it cannot export a second compactification point, host closed-set powerset, quotient filter equality, selected ultrafilter, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a real-valued or regular-rational consumer. By carrier projection, the consumer first sees the topology source X, closed-lattice row L, boundary-filter row F, source-inclusion row J, and topology handoff row T.

The compact-boundary admission row B and L10 readback row R are downstream of that prefix. They may cite the compact, RegSeqRat, and Real sibling chapters named in the provenance row, but they receive only the displayed finite route rather than an ambient compactification object.

Transport and replay preserve the same route through H and C, and P, N only package and name it. Hence the L10 boundary cannot produce quotient filter equality, selected ultrafilters, countable choice, a second compactification point, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{WallmanCompactification}^{\uparrow}$ consumer that reaches compact-boundary or Real readback data without displaying the source topology, finite closed-set lattice, boundary-filter consistency, source inclusion, and topology handoff rows refutes the carrier surface.

Independence witness: one-point-compactification, freudenthal-compactification, compact, topology, regseqrat, real: no listed sibling supplies the whole closed-lattice boundary packet $K = (X, L, F, B, J, T, R, H, C, P, N)$, because each sibling contributes only a point-boundary, end-boundary, compactness, topology, or L10 readback surface while this chapter binds all of them through one Wallman closed-set lattice route.

Closure status for $\mathsf{NameCert}_{\mathsf{WallmanCompactification}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WallmanCompactification}^{\uparrow}$ packet is generated from a source topology row, a finite closed-set lattice row, a boundary-filter consistency row, compact-boundary admission, source inclusion, topology handoff, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WallmanCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7296.1, Definition 7296.2, and Definition 7296.3 seed the Wallman compactification packet over displayed topology, finite closed-set lattice, boundary-filter, compact-boundary, and L10 readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WallmanCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host closed-set powersets, selected ultrafilters, quotient compactification points, open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source topology admission, finite closed-lattice operations, boundary-filter consistency, compact-boundary admission, source inclusion, topology handoff, L10 readback, transport, replay, provenance, and local naming for BEDC.Derived.WallmanCompactificationUp.

Chapter 7297

A Concrete Naming Certificate for $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$

Following Ciarlet and Brenner–Scott, the $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ packet records the finite stiffness matrix obtained by reading local element energy rows, assembling them over a displayed mesh, and exposing the result as BEDC NameCert data. It sits after the matrix carrier of chapter 49, the linear-map handoff of chapter 48, the finite-difference stencil surface of chapter 5135, the PDE relation carrier of chapter 194, the terminal real-name seal of chapter 38, and the metric-space source of chapter 4230. The packet names the finite energy matrix and its Galerkin-facing handoff; it does not assert a host weak solution, mesh-limit convergence, global coercivity theorem, quotient function equality, selected basis outside the displayed rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7297.1 (Finite element stiffness matrix carrier)**.** *A* $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, E, B, A, L, R, D, V, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite mesh row, listing element names, incidence data, boundary tags, and the finite trial-test basis labels inspected by the assembly route.*
2. *E is the element-energy row, recording the local bilinear energy contributions read on each displayed element.*
3. *B is the boundary-condition row, separating constrained degrees of freedom, interior degrees of freedom, and boundary load routes.*
4. *A is the* $\mathsf{Mat}^{\uparrow}$ *assembly row whose entries are obtained by summing only the displayed element contributions from E over the incidence row M.*
5. *L is the* $\mathsf{LinMap}^{\uparrow}$ *handoff row reading the same assembled matrix as a finite operator on the displayed trial-test coordinate window.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for matrix entries, local energy values, symmetry comparisons, and positive-energy tests.*
7. *D is the* $\mathsf{FiniteDifference}^{\uparrow}$ *or quadrature-facing local-difference row used when the element energy is read from displayed finite stencil data.*

8. *V is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *geometric row naming the finite metric observations used to orient element neighbourhoods and boundary faces.*
9. *H records componentwise* hsame *transport for $M, E, B, A, L, R, D, V, C, P, N$.*
10. *C records displayed* Cont *replay from mesh and local-energy reads through assembly, linear-map interpretation, scalar sealing, local-difference data, and metric geometry.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient finite-element space, a selected external basis, host weak-form equality, mesh-limit convergence, PDE solvability, global coercivity, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7297.2 (Finite element stiffness matrix NameCert obligations)**.** *For every accepted* $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ *carrier*

$$K = (M, E, B, A, L, R, D, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}}$ *surface consists exactly of finite mesh admission, element-energy admission, boundary-condition separation, Matrix assembly, LinearMap handoff, Real scalar sealing, local finite-difference or quadrature readback, metric-geometric orientation, componentwise transport, displayed replay, provenance, and local naming through $M, E, B, A, L, R, D, V, H, C, P, N$.*

Proof. Project the carrier of Definition 7297.1. The mesh row M, element-energy row E, and boundary row B are the only source rows from which the assembly route may draw element and boundary data.

The assembled row A is then a $\mathsf{Mat}^{\uparrow}$ readback of those displayed local contributions. The row L reads exactly the same table as a finite $\mathsf{LinMap}^{\uparrow}$ handoff, while R seals every entry and comparison as displayed $\mathsf{Real}^{\uparrow}$ data.

The local-difference row D and metric row V keep the finite stencil, quadrature, and geometric observations visible. They do not supply a host weak form or a mesh-limit object.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every obligation is one of the displayed rows and no excluded analytic endpoint is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7297.3 (Finite element stiffness matrix symmetric positive ledger)**.** *Every symmetry or positive-energy read of an accepted* $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ *packet factors through the same displayed rows $M, E, B, A, L, R, D, V, H, C, P, N$. Symmetry is read as equality of the paired finite matrix-entry ledgers in A after the element incidences in M and local energies in E have been displayed; positive-energy evidence is read as a finite* $\mathsf{Real}^{\uparrow}$ *comparison over the displayed coordinate window and does not assert global coercivity, spectral positivity, elliptic regularity, or PDE solvability.*

Proof. Start with the matrix assembly row A. Each entry comparison in a symmetry read is attached to the mesh incidence row M and the element-energy row E, so the comparison is a finite ledger over displayed element contributions.

For positive-energy evidence, the consumer reads the coordinate window through L and the scalar comparisons through R. Boundary rows B and metric or local-difference rows D, V remain visible before the comparison is exported.

Transport and replay through H, C preserve the same rows, and P, N record provenance and local naming. The packet therefore supplies only a finite symmetric positive ledger for the displayed matrix, not a global coercivity theorem, spectral theorem, regularity theorem, PDE solution, or mesh-limit conclusion. □

Theorem 7297.4 (Finite element stiffness matrix Galerkin handoff)**.** *Every Galerkin-facing or PDE-facing consumer of an accepted* $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ *carrier must read the assembly route*

$$M, E, B, A, L, R, D, V, H, C, P, N$$

before it may use the stiffness matrix as input to a finite variational calculation. The handoff reaches $\mathsf{Pde}^{\uparrow}$ *only through the displayed Matrix, LinearMap, Real, finite-difference, and metric rows, and exports no weak solution, convergence theorem, quotient finite-element function, selected external basis, unbounded solver trace, or ambient completeness principle.*

Proof. Begin with a Galerkin-facing consumer. The consumer first sees the finite mesh row M, element-energy row E, and boundary row B, so the variational data are tied to displayed element and boundary coordinates.

Next it reads the assembled matrix A and the linear-map handoff L. This makes the stiffness object a finite matrix and operator row, rather than an ambient bilinear form over an unspecified function space.

The scalar, local-difference, and metric rows R, D, V then expose the real comparisons, stencil or quadrature evidence, and geometric observations that a PDE-facing read may consume. The PDE route is therefore a provenance and replay handoff, not a proof of a host solution theorem.

Finally H, C, P, N preserve and name the same finite route. Since the carrier has no coordinate for weak solutions, convergence, quotient functions, selected external bases, solver traces, or ambient completeness, none of those endpoints can be exported by the Galerkin handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}$ packet is generated from finite mesh rows, local element-energy rows, boundary-condition rows, a displayed MatrixUp assembly table, a LinearMapUp handoff, RealUp scalar seals, finite-difference or quadrature readbacks, MetricSpace or Metric geometry rows, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7297.1, Definition 7297.2, Definition 7297.3, and Definition 7297.4 seed the finite stiffness-matrix carrier, its symmetric positive ledger, and its Galerkin-facing handoff over displayed Matrix, LinearMap, Real, finite-difference, PDE, and metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteElementStiffnessMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host weak solutions, mesh-limit convergence, global coercivity, spectral positivity, elliptic regularity, quotient finite-element functions, selected external bases, solver traces, Lean verification, or a standard bridge.

Upgrade path: Scoped treatment asks for checked carrier admission, element-energy admission, boundary separation, matrix assembly, linear-map handoff, scalar sealing, local-difference or quadrature readback, metric geometry, transport, replay, provenance, and local naming.

Chapter 7298

Baire open-dense intersection NameCert construction

Following Baire and Bishop–Bridges, the $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$ packet records the complete-metric nested-ball construction for an open-dense intersection as finite BEDC NameCert data. It sits beside the theorem-level category route of chapter 3570, the schedule surface of chapter 6530, the complete-metric source of chapter 128, the metric carrier of chapter 57, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names only the displayed nested-ball route; it is not a host comeager set, a countable-choice selector, a quotient of Cauchy streams, or an ambient real-completeness principle.

Definition 7298.1 (Baire open-dense intersection carrier)**.** *A* $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, D, B, \rho, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *source row, naming the complete metric service and the metric ball classifier used by the construction;*
2. *D is the* $\mathsf{OpenDenseSequence}^{\uparrow}$ *schedule row, listing the finite dense-open requests and predecessor-ball refinements visible to the packet;*
3. *B is the nested-ball choice row, a displayed finite window of centers, radii, containment certificates, and dense-open admissions obtained from the rows M and D;*
4. *ρ is the radius-budget row, recording the positive rational bounds and shrinkage comparisons used to keep the nested balls Cauchy-facing;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row that names the finite prefix windows through which centers, radii, and dense-open requests are inspected;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the Cauchy thread assembled from B, ρ, S;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has been displayed;*
8. *H records componentwise* hsame *transport for the complete-metric source, schedule, nested-ball choices, radius budget, stream observations, regular readback, real seal, continuation rows, provenance row, and local naming row;*

9. *C records the displayed* Cont *replay routes by which a consumer rereads the complete-metric source, each dense-open request, each nested-ball refinement, the radius budget, the stream windows, the regular readback, and the real seal without changing the finite packet;*
10. *P is the* Pkg *provenance over* $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{OpenDenseSequence}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireOpenDenseIntersection}^{\uparrow}}$ *row.*

The carrier has no coordinate for a countable-choice selector, a host intersection object, a comeager-set predicate, quotient equality of names, an arbitrary completed point, open-cover compactness, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 7298.2 (Baire open-dense intersection NameCert obligations)**.** *For an accepted* BaireOpenDenseIntersect *packet $I = (M, D, B, \rho, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireOpenDenseIntersection}^{\uparrow}}$ *obligation surface is exhausted by five rows: source exposure through M, D, nested construction exposure through B, ρ, observation and readback exposure through S, Q, E, componentwise transport and replay through H, C, and provenance with local naming through P, N.*

Proof. Project the carrier of Definition 7298.1. The complete-metric and dense-open source obligations are exactly the rows M, D.

The constructive route is the displayed nested-ball row B together with the radius-budget row ρ. These rows record the finite predecessor/successor-ball data and the positive rational shrinkage comparisons needed by the open-dense intersection read.

The terminal observation surface is S, Q, E. The stream windows expose the inspected finite data, the regular rational row reads back the Cauchy thread, and the real seal is available only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no selector, host intersection, comeager predicate, quotient equality, selected completed point, open-cover compactness, or ambient real-completeness coordinate, the five listed rows exhaust the local NameCert obligations. □

Theorem 7298.3 (Baire open-dense intersection nested-ball route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{CompleteMetric}^{\uparrow}, \quad \mathsf{OpenDenseSequence}^{\uparrow}, \quad B, \rho, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The route is transported and replayed only by H, C, P, N. No accepted read may replace it by a countable-choice selector, host comeager set, quotient equality of Cauchy names, arbitrary completed point, open-cover compactness, or ambient real completeness.

Proof. Read the source rows first. The complete-metric and metric-ball classifier row M supplies the ball service, while the schedule row D supplies the finite open-dense requests.

The nested construction then proceeds through B and ρ. Each visible refinement reads the next dense-open request, records the retained successor ball, and displays the positive rational radius comparison before any Cauchy-facing read is exported.

The consumer reaches the completion-facing surface only through S, Q, E. The finite stream row exposes the inspected window, the regular rational row reads the assembled Cauchy thread, and the real seal is the terminal value-facing row.

The structural rows H, C, P, N preserve that same route under componentwise transport, displayed replay, provenance, and local naming. Therefore every completion-facing read passes through the complete-metric, dense-open, nested-ball, radius-budget, StreamName, RegSeqRat, and Real-seal rows, with no hidden selector or host topological object. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOpenDenseIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$ packet is generated from a complete-metric source row, an open-dense schedule, displayed nested-ball choices, a positive radius budget, finite StreamName observations, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireOpenDenseIntersection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7298.1, Definition 7298.2, and Definition 7298.3 seed the finite complete-metric open-dense intersection route for completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireOpenDenseIntersection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host comeager set, arbitrary countable intersections, countable choice, quotient equality of Cauchy names, selected completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BaireOpenDenseIntersection}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, nested-ball refinement, radius-budget exactness, finite-window readback, regular rational readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BaireOpenDenseIntersectionUp.

Chapter 7299

A Concrete Naming Certificate for $\mathsf{SequentialLimitPoint}^{\uparrow}$

Following Kelley and Bourbaki on limit points and filters, together with Bishop–Bridges constructive sequence practice, the $\mathsf{SequentialLimitPoint}^{\uparrow}$ packet names the sequence-specific route by which a displayed sequence supplies a limit-point candidate through finite tail evidence. It consumes the tail filter of chapter 1564, the cluster handoff of chapter 6155, the finite source windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite NameCert interface for sequence-limit arguments; it does not import a selected host limit, quotient filter equality, ambient compactness, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7299.1 (Sequential limit-point carrier)**.** *A* $\mathsf{SequentialLimitPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, F, Q, R, E, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *sequence row, exposing only finite observation windows of the named sequence;*
2. *F is the* $\mathsf{FrechetFilter}^{\uparrow}$ *tail row, recording the eventual-tail requests through which subsequence or neighbourhood reads are inspected;*
3. *Q is the* $\mathsf{ClusterFilter}^{\uparrow}$ *handoff row, used only after the Frechet tail row has been displayed;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite source and tail windows;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after S, F, Q, R have been exposed;*
6. *H records componentwise* hsame *transport of S, F, Q, R, E;*
7. *C records displayed* Cont *replay from a sequence-window request through the tail and cluster rows to the terminal seal;*
8. *P records* Pkg *provenance over* $\mathsf{SequentialLimitPoint}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{ClusterFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialLimitPoint}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected host limit, quotient filter equality, arbitrary subnet selection, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7299.2 (Sequential limit-point NameCert obligations)**.** *For the carrier of Definition 7299.1, the local* $\mathsf{NameCert}_{\mathsf{SequentialLimitPoint}^{\uparrow}}$ *surface consists exactly of source StreamName admission through* S*, Frechet tail admission through* F*, ClusterFilter handoff through* Q*, RegSeqRat readback through* R*, Real seal handoff through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The source sequence row S and Frechet tail row F are the only rows that expose sequence observations and eventual-tail requests.

Read the cluster handoff and regular readback next. The row Q is downstream of the tail filter, and R reads only the displayed finite windows supplied by S and F before any real-facing seal is available.

Finally the seal and structural rows E, H, C, P, N replay, transport, source, and name the same finite route. Since the carrier contains no host-limit, quotient-filter, subnet-choice, compactness, countable-choice, or completeness coordinate, none of those principles can be exported by the local NameCert surface. □

Theorem 7299.3 (Sequential limit-point Frechet cluster route)**.** *Every accepted* $\mathsf{SequentialLimitPoint}^{\uparrow}$ *packet reaches a cluster-facing or real-facing consumer only through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{ClusterFilter}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, a consumer must first inspect the source sequence windows, then the Frechet tail row, then the cluster handoff, then the regular rational readback, and only then the terminal real seal.

Proof. Start with the carrier row S. It supplies finite sequence windows and no infinite representative.

The Frechet tail row F then fixes the eventual-tail request, and the cluster row Q may be consumed only after that tail request is displayed. Thus cluster use remains sequence-specific rather than an arbitrary filter import.

The readback row R and seal row E finish the route through regular rational data and the terminal $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve that order, so the route cannot bypass the Frechet tail or cluster handoff by selecting a host limit or using quotient filter equality. □

Falsifiable prediction: Every $\mathsf{SequentialLimitPoint}^{\uparrow}$ read displays one StreamName sequence row, one Frechet tail row, one ClusterFilter handoff row, one RegSeqRat readback row, one Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any limit-point consumer is public.

Independence witness: streamname, frechetfilter, cluster-filter, regseqrat, real: $\mathsf{SequentialLimitPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a sequence source, a Frechet tail row, a cluster-filter handoff, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming as one sequence-specific limit-point certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialLimitPoint}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SequentialLimitPoint}^{\uparrow}$ packet is generated from source StreamName sequence rows, Frechet tail rows, ClusterFilter handoff rows, RegSeqRat readback, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialLimitPoint}^{\uparrow}}$, seed)

Scope closed: Definition 7299.1, Definition 7299.2, and Definition 7299.3 seed the sequence-specific limit-point route over StreamName, FrechetFilter, ClusterFilter, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialLimitPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected host limits, quotient filter equality, arbitrary subnet selection, open-cover compactness, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.SequentialLimitPointUp.

Chapter 7300

A Concrete Naming Certificate for $\mathsf{PeetreKFunctional}^{\uparrow}$

Following Peetre and the real interpolation tradition, the $\mathsf{PeetreKFunctional}^{\uparrow}$ packet names the finite carrier that exposes a K-functional request before any interpolation-scale consumer reads it. It sits between the $\mathsf{NormedSpace}^{\uparrow}$ pair boundary of chapter 1950, the $\mathsf{Real}^{\uparrow}$ value seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ finite-window discipline of chapter 126, the positive-radius ledger of chapter 291, and the regularity-scale siblings $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, and $\mathsf{RieszThorinInterpolation}^{\uparrow}$ of chapter 4204, chapter 6330, and chapter 5725. The packet records a displayed decomposition budget and value-readback route; it does not assert a quotient interpolation space, selected infimum, countable choice, host norm equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7300.1 (Peetre K-functional carrier)**.** *A* $\mathsf{PeetreKFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, X, T, B, V, Q, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row naming the interpolation request and the finite expression whose decomposition is inspected.*
2. *X is the* $\mathsf{NormedSpace}^{\uparrow}$ *pair row, displaying the two normed sources used by the request.*
3. *T is the positive tolerance row for the $K(t, x)$ read, exposed as a finite request rather than a completed infimum.*
4. *B is the decomposition-budget row listing the finite candidate splits visible to the certificate.*
5. *V is the value-readback row for the displayed norm-sum comparisons attached to the visible splits.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the finite value windows.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which coordinates of the value stream have been inspected.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger controlling tolerance and comparison requests.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, X, T, B, V, Q, W, D are visible.*
10. *H records componentwise* hsame *transport for all displayed rows.*

11. *C records displayed* Cont *replay from the interpolation request through decomposition budget, value readback, regular rational stream, dyadic ledger, Real seal, provenance, and local naming.*
12. *P is the* Pkg *provenance over* $\mathsf{PeetreKFunctional}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, $\mathsf{RieszThorinInterpolation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient interpolation spaces, selected infima, countable choice, host norm equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.PeetreKFunctionalUp`

Theorem 7300.2 (Peetre K-functional NameCert obligations)**.** *For every accepted* $\mathsf{PeetreKFunctional}^{\uparrow}$ *carrier*

$$K = (S, X, T, B, V, Q, W, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}$ *surface consists exactly of source admission,* $\mathsf{NormedSpace}^{\uparrow}$ *pair exposure, positive tolerance exposure, finite decomposition-budget display, value-readback routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{StreamName}^{\uparrow}$ *finite-window inspection,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius control,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7300.1. The analytic rows are $S, X, T, B, V, Q, W, D, E$, and the structural rows are H, C, P, N.

Read the normed pair and tolerance rows first. The row X displays the two normed sources, while T records the positive request scale. The decomposition row B is therefore a finite budget over displayed candidate splits, not a hidden selector for an extremal split.

Next read the value and completion-facing rows. The rows V, Q, W, D, E carry finite norm-sum comparisons through regular rational readback, stream windows, dyadic radii, and the terminal Real seal. None of these rows supplies quotient equality or ambient completeness.

Finally the transport, replay, provenance, and local naming rows preserve the same finite route. Since no excluded object is a coordinate of the carrier, the local certificate cannot export quotient interpolation spaces, selected infima, countable choice, host norm equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7300.3 (Peetre interpolation-scale handoff)**.** *Every Besov, Triebel–Lizorkin, or real-interpolation consumer of an accepted* $\mathsf{PeetreKFunctional}^{\uparrow}$ *packet must read the* $\mathsf{NormedSpace}^{\uparrow}$ *pair row, positive tolerance row, finite decomposition-budget row, value-readback row,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows before it can route data to* $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, *or* $\mathsf{RieszThorinInterpolation}^{\uparrow}$. *The handoff exports the finite K-functional value route only; it exports no quotient interpolation space, selected infimum, countable-choice schedule, host norm equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7300.2. They require the rows $S, X, T, B, V, Q, W, D, E, H$ to be read as one packet before any interpolation-scale consumer inspects the endpoint.

The decomposition budget B is downstream of the normed-pair row X and the tolerance row T. Thus a consumer first reads the finite pair and request scale, then inspects only the displayed candidate splits.

The value route is completion-facing but finite. The rows V, Q, W, D, E carry the displayed comparisons through regular rational readback, stream windows, dyadic-radius control, and the terminal Real seal before any regularity-scale sibling reads the result.

The structural rows H, C, P, N transport, replay, source, and name the same route. Therefore the consumer receives only the finite K-functional handoff and none of the excluded host interpolation claims. □

Theorem 7300.4 (Peetre K-functional decomposition-window route)**.** *For every accepted* $\mathsf{PeetreKFunctional}^{\uparrow}$ *carrier*

$$K = (S, X, T, B, V, Q, W, D, E, H, C, P, N),$$

any read of a K-functional value routes through the displayed $\mathsf{NormedSpace}^{\uparrow}$ *pair row X, the positive tolerance row T, the finite decomposition-budget row B, the value-readback row V, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger D, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}$ *row N. The route exports only this finite decomposition window; it exports no selected infimum, quotient interpolation space, countable-choice schedule, host norm equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7300.1. The value read has no independent coordinate outside $S, X, T, B, V, Q, W, D, E, H, C, P, N$, so the normed pair, tolerance, decomposition budget, readback, window, radius, and Real-seal rows are visible before the structural rows are consumed.

Apply the obligation surface of Definition 7300.2. It identifies the local NameCert rows as source admission, $\mathsf{NormedSpace}^{\uparrow}$ pair exposure, positive tolerance exposure, finite budget display, value-readback routing, regular rational readback, finite-window inspection, dyadic-radius control, Real sealing, transport, replay, provenance, and local naming.

Apply the interpolation handoff of Definition 7300.3. It ensures that any regularity-scale consumer reads the same rows before routing data to $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, or $\mathsf{RieszThorinInterpolation}^{\uparrow}$.

The carrier contains no coordinate for an extremal split, quotient interpolation object, countable-choice schedule, host norm equality, or ambient completeness principle. Thus the decomposition-window route is exactly the finite route displayed by the carrier and its NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{PeetreKFunctional}^{\uparrow}$ read is refuted if a Besov, Triebel–Lizorkin, or real-interpolation consumer reaches the $\mathsf{Real}^{\uparrow}$ value seal without first displaying the $\mathsf{NormedSpace}^{\uparrow}$ pair, positive tolerance, finite decomposition budget, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger.

Independence witness: normedspace, real, regseqrat, streamname, dyadicratcore, besovspace, triebel-lizorkin-space, riesz-thorin-interpolation: none is bijective with $\mathsf{PeetreKFunctional}^{\uparrow}$ because this packet jointly carries a normed-pair source, tolerance request, decomposition budget, value readback, regular rational stream, dyadic radius ledger, Real seal, interpolation-scale handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PeetreKFunctional}^{\uparrow}$ packet is generated from a source request, NormedSpace pair row, positive tolerance row, finite decomposition-budget row, value-readback row, RegSeqRat row, StreamName finite-window row, DyadicRatCore radius ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7300.1, Definition 7300.2, and Definition 7300.3 seed the finite Peetre K-functional carrier and its interpolation-scale handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PeetreKFunctional}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.PeetreKFunctionalUp`

Bridge status: none

Not claimed: This chapter does not claim quotient interpolation spaces, selected infima, countable choice, host norm equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, decomposition-budget exposure, value-readback routing, interpolation-scale handoff, transport, replay, provenance, and local naming for BEDC.Derived.PeetreKFunctionalUp.

Chapter 7301

A Concrete Naming Certificate for $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$

Following Bishop–Bridges constructive analysis, a regulated real function on a compact interval is controlled by finite step approximations and dyadic mesh refinements before a Riemann-sum consumer may read an integral value. The $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ packet records that integrability certificate as BEDC NameCert data. It sits beside the regulated-function schedule of chapter 1906, the Darboux finite-sum surface of chapter 1880, the Riemann-integral readback of chapter 1526, the Real seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet names finite mesh-and-sum evidence only; it does not import measure completion, selected limits, quotient function equality, host integration, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7301.1 (Regulated Riemann integrability carrier)**.** *A* $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, M, G, D, T, H, C, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{Real}^{\uparrow}$ *interval-domain row, naming the left endpoint, right endpoint, orientation data, and the compact interval on which the certificate is read.*
2. *M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh ledger, recording the dyadic subdivision request, refinement order, and finite mesh width used by the certificate.*
3. *G is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *step-approximation schedule, listing the step functions and oscillation bounds inspected on the dyadic mesh.*
4. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *finite-sum handoff row, exposing lower and upper finite sums computed from the displayed step-approximation data.*
5. *T is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *tagged-mesh read, available only after A, M, G, D have been displayed and the Darboux gap is inside the requested dyadic tolerance.*
6. *H records componentwise* hsame *transport for interval domain, mesh ledger, step schedule, finite-sum handoff, tagged-mesh read, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an interval request through dyadic mesh refinement, regulated step approximation, Darboux finite sums, and tagged-mesh readback.*

8. *P is the* Pkg *provenance over* $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *L is the local* $\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, G, D, T, H, C, P, L$. It has no coordinate for a measure algebra, host function quotient, selected limiting partition, ambient compactness proof, or external completeness principle.

Theorem 7301.2 (Regulated Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (A, M, G, D, T, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the Real interval domain, dyadic mesh ledger, regulated step-approximation schedule, Darboux finite-sum handoff, tagged-mesh read, transport, replay, provenance, and local naming rows;*

2. *mesh coverage, requiring M to refine the interval row A before the step schedule G is inspected;*

3. *regulated approximation exactness, requiring the entries of G to be step data over the mesh cells named by M;*

4. *Darboux handoff exactness, requiring D to compute finite lower and upper sums only from A, M, G and to expose the dyadic gap budget;*

5. *tagged-mesh non-escape, requiring T to be read only after A, M, G, D and the structural rows H, C, P, L are visible.*

Thus an accepted regulated-integrability certificate is a finite NameCert packet over interval data, mesh data, regulated approximation data, finite Darboux sums, tagged-mesh readback, transport, replay, provenance, and local naming.

Proof. Project the displayed carrier of Definition 7301.1. The tuple lists exactly the interval, mesh, approximation, finite-sum, tagged-read, and structural rows.

First read the source side. The interval row A supplies the $\mathsf{Real}^{\uparrow}$ domain, while M supplies the dyadic mesh ledger on that same interval. The regulated schedule G is therefore inspected only cellwise over displayed mesh data.

Next read the finite-sum side. The Darboux row D is computed from the step schedule and mesh cells already displayed, so its lower and upper sums cannot be replaced by a hidden host integral or selected limiting partition.

Finally read T, H, C, P, L. The tagged-mesh row T is downstream of the finite Darboux gap, and the transport, replay, provenance, and local-name rows preserve the same route. Since no further coordinate remains, the five obligations exhaust the local $\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}$ surface. □

Falsifiable prediction: A $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ consumer that reaches a $\mathsf{RiemannIntegral}^{\uparrow}$ tagged-mesh read without displaying the $\mathsf{Real}^{\uparrow}$ domain row, $\mathsf{DyadicMesh}^{\uparrow}$ mesh ledger, $\mathsf{RegulatedFunction}^{\uparrow}$ step schedule, and $\mathsf{DarbouxSum}^{\uparrow}$ finite-sum handoff refutes the carrier of Definition 7301.1.

Independence witness: regulatedfunction, darbouxsum, riemannintegral, real, dyadicmesh: $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval-domain rows, dyadic mesh rows, regulated step-approximation rows, Darboux finite-sum rows,

tagged-mesh readback, hsame transport, Cont replay, Pkg provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedRiemannIntegrability}^{\uparrow}$ packet is generated from a Real interval-domain row, DyadicMesh ledger, RegulatedFunction step-approximation schedule, DarbouxSum finite-sum handoff, RiemannIntegral tagged-mesh read, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7301.1 and Definition 7301.2 seed the finite regulated-integrability carrier and its mesh-to-Darboux-to-tagged-read route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly refuses measure completion, selected limits, quotient function equality, host integration, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh coverage, regulated approximation exactness, Darboux finite-sum exactness, tagged-mesh ordering, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.RegulatedRiemannIntegrabilityUp.

Chapter 7302

A Concrete Naming Certificate for $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$

Following Dunford–Schwartz and Bishop–Bridges normed-operator analysis, the $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$ packet records finite iterates of a bounded linear operator before spectral-radius or Neumann-series consumers inspect the bound. It sits between the bounded-operator route of chapter 1644, the scalar-bound surface of chapter 3603, the finite inverse-series route of chapter 3917, the spectral-radius boundary of chapter 2352, and the rational-to-real readback chain of chapter 301 and chapter 38. The packet names a finite power corridor; it does not assert a spectral-radius limit, a Neumann inverse, a quotient equality of operators, or an ambient completeness theorem.

Definition 7302.1 (Spectral-radius finite-power carrier)**.** *A* $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, n, A, B, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the source endpoint, target endpoint, endpoint action, linearity ledger, and displayed norm-bound surface.*
2. *n is the unary power-depth row specifying the finite iterate window that may be inspected.*
3. *A is the finite iterate row. It records the accepted powers $T^0, T^1, \ldots, T^n$ only as displayed operator rows and product ledgers.*
4. *B is the displayed norm-bound ledger for the same finite window. Each entry reads its scalar bound through the* $\mathsf{OperatorNorm}^{\uparrow}$ *surface and keeps the product-bound ancestry visible.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for the rational bound entries and terminal real seals used by downstream comparisons.*
6. *H records componentwise* hsame *transport for T, n, A, B, R and the structural rows.*
7. *C records displayed* Cont *replay from a bounded-operator power request through the same finite iterate and norm-bound rows.*
8. *P is the* Pkg *provenance over* $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{NeumannSeries}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePowerBoundedOperator}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through the bounded-operator source row, unary depth, finite iterate ledger, displayed norm-bound ledger, rational and real readback, transport, replay, provenance, and local naming.

Theorem 7302.2 (Spectral-radius finite-power bound)**.** *For every accepted* $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$ *carrier*

$$F = (T, n, A, B, R, H, C, P, N),$$

all public finite-power and norm-bound information is exposed only through the rows $T, n, A, B, R, H, C, P, N$*. A* $\mathsf{SpectralRadius}^{\uparrow}$ *consumer may read the finite iterate window and displayed norm bounds, and a* $\mathsf{NeumannSeries}^{\uparrow}$ *consumer may read the same finite corridor before checking residual rows, but neither consumer receives a spectral-radius limit, a Neumann-series inverse, quotient operator equality, selected representative data, or an ambient completeness coordinate from this packet.*

Proof. Project the carrier of Definition 7302.1. The bounded operator is the row T, and the only iteration budget is the unary depth row n.

The finite powers are then exactly the entries of A. Their scalar controls are exactly the entries of B, with rational and real readback passing through R.

Transport and replay do not add analytic coordinates. The rows H and C move and replay the same finite iterate and norm-bound ledger, while P and N only source and name that surface.

Thus a spectral-radius or Neumann-series consumer can inspect the finite corridor but cannot recover a limiting radius, an inverse, quotient equality of operators, selected representatives, or ambient completeness data from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$ read exposes a BoundedLinearOperatorUp row, unary power depth, finite iterate row, displayed norm-bound ledger, RegSeqRatUp and RealUp readback, hsame transport, Cont replay, Pkg provenance, and local NameCert before any spectral-radius or Neumann-series consumer may cite the corridor.

Independence witness: boundedlinearoperator, operatornorm, neumann-series, spectral-radius, regseqrat, real: BoundedLinearOperatorUp supplies the operator row without a finite iterate corridor; OperatorNormUp supplies scalar bound rows without a power-depth ledger; NeumannSeriesUp supplies residual and inverse-check rows only after a separate finite corridor is read; SpectralRadiusUp supplies a radius boundary without exporting a limiting theorem from this packet; RegSeqRatUp and RealUp seal bounds without naming operator powers.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePowerBoundedOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePowerBoundedOperator}^{\uparrow}$ packet is generated from a BoundedLinearOperatorUp row, unary power depth, finite iterate rows, displayed norm-bound ledgers, RegSeqRatUp and RealUp readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePowerBoundedOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7302.1, Definition 7302.2, the falsifiable prediction, and the independence witness seed the finite bounded-operator power corridor consumed by spectral-radius and Neumann-series packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePowerBoundedOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a spectral-radius limit, Neumann-series inverse, quotient operator equality, selected representative data, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite iterate stability, norm-bound ledger readback, rational and real bound sealing, transport and replay stability, and consumer nonescape for BEDC.Derived.FinitePowerBoundedOperatorU

Chapter 7303

A Concrete Naming Certificate for $\mathsf{RealTree}^{\uparrow}$

Following Tits and Morgan on real trees and Bridson–Haefliger on geodesic metric spaces, the $\mathsf{RealTree}^{\uparrow}$ packet records the finite metric-geometry route by which a uniquely tripod-shaped geodesic metric object is named inside BEDC. It sits beside the metric source of chapter 57, the geodesic segment surface of chapter 2213, the hyperconvex comparison packet of chapter 4013, the completion boundary of chapter 128, and the distance readback seal of chapter 38. The packet names only displayed distance rows, segment witnesses, and branch-point uniqueness ledgers; it is not a host path space, a selected geodesic family, an ambient dendrite, or a global completion theorem.

Definition 7303.1 (Real-tree carrier)**.** *A* $\mathsf{RealTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, G, R, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the distance observations available to the packet.*
2. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *segment-witness row, recording the finite segment reads that support the tree route.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-readback row for the displayed segment lengths and comparison budgets.*
4. *B is the branch-point and tripod-uniqueness ledger. It records, for each displayed triple of endpoints, the finite witness that the three segment reads meet in one branch row and that any competing branch row is componentwise identical under the packet classifier.*
5. *H records componentwise* hsame *transport for metric source, segment witnesses, Real readbacks, branch ledgers, continuation rows, provenance, and local naming.*
6. *C records displayed* Cont *replay for distance reads, segment reads, tripod reads, uniqueness checks, and downstream handoffs.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTree}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{HyperconvexMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, G, R, B, H, C, P, N. It has no coordinate for a host path space, selected global geodesic family, ambient continuum, quotient tree model, or completion principle.

Theorem 7303.2 (Real-tree NameCert obligations)**.** *For every accepted* $\mathsf{RealTree}^{\uparrow}$ *carrier* $T = (M, G, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealTree}^{\uparrow}}$ *surface consists exactly of metric-source admission, geodesic-segment witness exposure, Real distance readback, branch-point and tripod-uniqueness ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the real-tree packet must factor through*

$$M, G, R, B, H, C, P, N$$

before it reads a tree-shaped metric datum.

Proof. Project the carrier of Definition 7303.1. The metric source row M supplies the only distance observations, while the segment row G supplies the only geodesic witnesses visible to the packet.

Read the Real distance row R and the branch ledger B next. The row R seals the finite distance budgets, and B records the displayed tripod meeting row and uniqueness comparison for each named triple.

The structural rows H, C, P, N transport, replay, source, and name only those displayed rows. Since the carrier has no coordinate for host paths, selected geodesics, quotient tree models, or ambient completion, the listed rows exhaust the local NameCert obligations. □

Theorem 7303.3 (Real-tree geodesic-tripod handoff)**.** *Every boundary, completion, or hyperconvex-style consumer of an accepted* $\mathsf{RealTree}^{\uparrow}$ *packet receives only the finite handoff*

$$\mathsf{Met}^{\uparrow}, \mathsf{GeodesicMetric}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto B, H, C, P, N.$$

Thus it may inspect displayed distance observations, segment witnesses, Real-valued length reads, and branch-point uniqueness rows, but it cannot import a host path space, a selected family of geodesics, a global hyperconvex hull, or an ambient completion object.

Proof. Start with the metric row M and the geodesic row G. These rows are the only source of distance and segment information available to a consumer.

The Real readback row R turns the visible distances into the same finite Real-sealed comparisons used by the sibling $\mathsf{Real}^{\uparrow}$ route. The branch ledger B then supplies the tripod meeting row and uniqueness check for each displayed endpoint triple.

Finally route the handoff through H, C, P, N. The hyperconvex and completion siblings may consume the same finite metric rows, but the carrier never exports a host path space, selected geodesic family, global hull, or completion object. □

Falsifiable prediction: An accepted $\mathsf{RealTree}^{\uparrow}$ consumer that uses a branch point without displaying the metric row, geodesic segment witnesses, Real distance readbacks, and tripod-uniqueness ledger refutes the carrier surface.

Independence witness: metric, geodesic-metric, hyperconvex-metric, completemetric, real: no listed sibling supplies the whole packet $T = (M, G, R, B, H, C, P, N)$, because the real-tree route binds metric distance, segment witnesses, Real readback, branch-point uniqueness, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealTree}^{\uparrow}$ packet is generated from MetricUp distance rows, GeodesicMetricUp segment witnesses, RealUp distance readbacks, a branch-point and tripod-uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTree}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7303.1, Definition 7303.2, and Definition 7303.3 seed the real-tree packet over displayed metric, geodesic, Real-distance, branch-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTree}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, selected global geodesics, ambient dendrites, quotient tree models, global hyperconvex hulls, ambient completion objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, segment-witness exposure, Real distance readback, tripod uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.RealTreeUp.

Chapter 7304

A Concrete Naming Certificate for $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$

Doob's martingale convergence and upcrossing tradition isolates finite stopped windows before any limiting path statement is read. The $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ packet records that finite route between the martingale, stopping, optional-stopping, upcrossing, Doob-upcrossing, and convergence-preparation certificates of chapter 190, chapter 344, chapter 383, chapter 1629, chapter 1630, and chapter 2246. It names the finite stopping-window carrier consumed by later convergence preparation, not an almost-sure quotient, completed filtration, unbounded stopping theorem, or measure-completion object.

Definition 7304.1 (Martingale stopping-convergence carrier)**.** *A* $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\Omega, X, E, M, T, O, U, D, W, A, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *probability row.*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint-family row for the inspected martingale prefix.*
3. E *is the* $\mathsf{CondExp}^{\uparrow}$ *row exposing the successor faces used by the martingale law.*
4. M *is the consumed* $\mathsf{Martingale}^{\uparrow}$ *finite adapted-sequence packet.*
5. T *is the* $\mathsf{StoppingTime}^{\uparrow}$ *bounded decision row.*
6. O *is the* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window readback row.*
7. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count ledger for displayed lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoints.*
8. D *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *inequality row bounding the displayed crossing-count ledger.*
9. W *is the finite stopping-window ledger naming the inspected horizon, stopped endpoints, crossing counts, and boundary endpoints available to convergence preparation.*
10. A *is the* $\mathsf{MartingaleConvergence}^{\uparrow}$ *admission row that may consume only the finite window named by* W*.*
11. H *records componentwise* hsame *transport for* $\Omega, X, E, M, T, O, U, D, W, A, C, P, N$*.*

12. *C records displayed* Cont *replay through the finite endpoint, stopped-window, upcrossing, Doob-bound, and convergence-admission rows.*
13. *P is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{MartingaleConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their componentwise transports. It has no coordinate for an almost-sure quotient, selected exceptional set, completed sigma-algebra, unbounded stopping-time family, or ambient convergence theorem.

Theorem 7304.2 (Martingale stopping-convergence NameCert obligations)**.** *For every accepted* $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ *carrier*

$$S = (\Omega, X, E, M, T, O, U, D, W, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}$ *surface consists exactly of probability exposure, endpoint exposure, conditional-expectation exposure, martingale-prefix exposure, bounded stopping-decision exposure, optional-stopping stopped-window exposure, upcrossing ledger exposure, Doob-bound exposure, finite stopping-window exactness through W, convergence-admission routing through A, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7304.1. The rows $\Omega, X, E, M, T, O, U, D$ are the only stochastic and upcrossing inputs, while W and A are the only convergence-facing rows.

The stopped-window side is finite. The row T supplies bounded decisions, O supplies stopped endpoint reads, U supplies crossing counts, and D supplies the Doob-bound row over the same displayed horizon.

The convergence-facing row A is downstream of W. Since W lists only the finite horizon, stopped endpoints, crossing counts, and boundary endpoints already displayed by T, O, U, D, the local certificate cannot import an almost-sure quotient, a selected exceptional set, completed sigma-algebras, unbounded stopping, or an ambient convergence theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Therefore the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}$ surface. □

Theorem 7304.3 (Stopping-window upcrossing route)**.** *Every accepted* $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ *carrier routes convergence preparation through the finite chain*

$$\mathsf{Martingale}^{\uparrow}, \mathsf{StoppingTime}^{\uparrow}, \mathsf{OptionalStopping}^{\uparrow} \longmapsto \mathsf{Upcrossing}^{\uparrow}, \mathsf{DoobUpcrossing}^{\uparrow} \longmapsto \mathsf{MartingaleStoppingConvergen}$$

The exported object is the displayed stopping-window ledger W and the admission row A over the same endpoint, stopped-window, crossing-count, and Doob-bound rows. It is not an almost-sure convergence theorem, a completed-measure argument, an unbounded stopping theorem, or a quotient equality of stochastic processes.

Proof. Start with the martingale source row M and the stopping row T. They expose only a finite adapted prefix and a bounded decision row.

Next read O, U, D. Optional stopping supplies the stopped endpoints, the upcrossing row supplies the lower-to-upper crossing ledger, and the Doob row supplies the finite bound for that ledger.

The stopping-window row W is then a repacking of the same displayed horizon, endpoint, stopped-value, boundary, crossing, and bound rows. The convergence-admission row A may cite W, but no carrier coordinate contains an almost-sure event, a completed measure space, an unbounded stopping family, or a quotient of processes. □

Falsifiable prediction: Every accepted $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ consumer displays $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, finite stopping-window, convergence-admission, hsame, Cont, Pkg, and local NameCert rows before reading convergence preparation.

Independence witness: martingale, stoppingtime, optionalstopping, upcrossing, doob-upcrossing, martingale-convergence: none is bijective to $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ because this packet jointly carries bounded stopping decisions, stopped endpoints, crossing counts, Doob bounds, a stopping-window ledger, convergence admission, transport, replay, provenance, and local naming as one finite route.

Lean4 (statement) `BEDC.Derived.MartingaleStoppingConvergenceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{MartingaleStoppingConvergence}^{\uparrow}$ packet is generated from probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, Doob-upcrossing, finite stopping-window, convergence-admission, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7304.1, Definition 7304.2, and Definition 7304.3 seed the finite stopping-window route from optional stopping and upcrossing control to martingale-convergence preparation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MartingaleStoppingConvergence}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, completed filtrations, measure completion, unbounded stopping, quotient process equality, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stopped-window exactness, upcrossing handoff exactness, Doob-bound consumption, convergence-admission nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MartingaleStoppingConvergenceUp.

Chapter 7305

A Concrete Naming Certificate for $\mathsf{ObserverExperienceTrace}^{\uparrow}$

The $\mathsf{ObserverExperienceTrace}^{\uparrow}$ packet realizes the consciousness-observer vision hub of section 9417.8 as finite observer-history, locality, classifier, transport, replay, provenance, and NameCert rows. It sits beside the finite multi-history packet of chapter 468, the locality-cell route of chapter 509, the no-global-synchronization ledger of chapter 1106, and the cross-history causal surface of chapter 478. The packet names a finite trace surface for observer-facing reads; it does not assert a private mental-state oracle, privileged observer frame, global synchronization map, host phenomenology predicate, or physical consciousness law.

Definition 7305.1 (Observer experience trace carrier)**.** *An* $\mathsf{ObserverExperienceTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H, L, I, Q, R, V, A, S, C, P, N)$$

with the following displayed rows:

1. *H is the finite observer-history row, listing the displayed local histories being traced.*
2. *L is the inter-history locality ledger, recording which observations may be compared without a global synchronization row.*
3. *I is the observer-identity and perspective-classifier row, separating local identity reads from cross-history alignment reads.*
4. *Q is the finite query row for the experience trace: displayed observation sites, local classifier tags, and allowed read windows.*
5. *R is the replay row tying each query in Q back to the displayed observer-history and locality rows.*
6. *V is the provenance row from the consciousness-observer hub, recording which finite vision-site claims are being concretized.*
7. *A is the TasteGate-facing audit row requiring a future Lean* `Nontrivial` *instance and a* `FieldFaithful` *instance to expose at least two distinguishable trace fields and to account for all public carrier rows.*
8. *S records componentwise* hsame *transport for $H, L, I, Q, R, V, A, C, P, N$.*
9. *C records displayed* Cont *replay from observer-history reads through locality, perspective classification, trace queries, audit rows, and local naming.*

10. *P is the* Pkg *provenance over* ObserverExperienceTrace$^{\uparrow}$, FiniteMultiHistPacket$^{\uparrow}$, LocalityCell$^{\uparrow}$, NoGlobalSynchronizationLedger$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{ObserverExperienceTrace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed histories, locality rows, identity-classifier rows, query rows, replay rows, audit rows, structural transports, provenance, and local name.

Theorem 7305.2 (Observer experience trace boundary)**.** *Every accepted* ObserverExperienceTrace$^{\uparrow}$ *carrier exposes observer-facing trace data only through the finite rows*

$$H, L, I, Q, R, V, A, S, C, P, N.$$

The public boundary permits local observer-history reads, inter-history locality checks, perspective-classifier reads, replay of finite trace queries, vision provenance, and TasteGate-facing `Nontrivial` *and* `FieldFaithful` *obligations. It exports no private Self access, host phenomenology predicate, privileged global observer, global synchronization map, physical consciousness law, or quotient equality of observer histories.*

Proof. Project the carrier of Definition 7305.1. The trace source is H, L, I, Q, and the only replay route is R together with the displayed Cont row C.

The audit row A is explicit. It requires future Lean evidence that the carrier has a nontrivial trace distinction and that the displayed fields are faithful, but it does not itself add an external consciousness predicate.

The structural rows S, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for private Self access, host phenomenology, global synchronization, physical consciousness laws, or quotient observer-history equality, the stated boundary is exact. □

Theorem 7305.3 (Observer experience trace sibling independence)**.** *The* ObserverExperienceTrace$^{\uparrow}$ *carrier is not bijective to any single sibling among* FiniteMultiHistPacket$^{\uparrow}$, LocalityCell$^{\uparrow}$, NoGlobalSynchronizationLed *or* CrossHistCausal$^{\uparrow}$. *It jointly carries finite observer histories, inter-history locality, perspective-classifier rows, finite trace queries, replay provenance, TasteGate-facing audit fields, transport, continuation, package provenance, and local naming as one packet.*

Proof. The finite multi-history packet supplies displayed histories but not the trace query row Q or TasteGate-facing audit row A.

The locality-cell and no-global-synchronization packets supply locality and refusal rows, but they do not jointly carry the observer identity classifier I, replay row R, or vision provenance row V.

The cross-history causal packet supplies a causal comparison surface, while ObserverExperienceTrace$^{\uparrow}$ records a finite trace read with its own query, replay, audit, provenance, and local naming rows. Thus no listed sibling is bijective to the full carrier. □

Falsifiable prediction: An accepted ObserverExperienceTrace$^{\uparrow}$ consumer that reads an observer experience trace without displaying finite observer histories, inter-history locality rows, perspective-classifier rows, finite trace queries, replay provenance, TasteGate-facing audit rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows refutes the carrier boundary.

Independence witness: finite-multihist-packet, localitycell, noglobalsynchronizationledger, crosshist-causal: no listed sibling is bijective because ObserverExperienceTrace$^{\uparrow}$ jointly carries observer-history, locality, perspective-classifier, finite query, replay, provenance, TasteGate-audit, transport, continuation, package, and local NameCert rows.

Lean4 (statement) `BEDC.Derived.ObserverExperienceTraceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverExperienceTrace}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverExperienceTrace}^{\uparrow}$ packet is generated from finite observer histories, inter-history locality rows, perspective-classifier rows, trace queries, replay provenance, TasteGate-facing audit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverExperienceTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7305.1, Definition 7305.2, Definition 7305.3, the falsifiable prediction, and the independence witness seed a finite observer-experience trace carrier with explicit locality, classifier, replay, provenance, audit, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverExperienceTrace}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close private Self access, host phenomenology, privileged observer frames, global synchronization, physical consciousness laws, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observer-history exposure, locality exposure, classifier exactness, replay provenance, TasteGate Nontrivial and FieldFaithful instances, transport, continuation, package provenance, and local naming for BEDC.Derived.ObserverExperienceTraceUp.

Chapter 7306

Regular Cauchy Finite Splice NameCert Construction

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$ packet records finite prefix replacement of a regular rational Cauchy name while preserving the displayed tail route. It sits over the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the tail-shift route of chapter 1578, and the finite prefix certificate of chapter 7253. The packet gives Real and Completion consumers a finite means of changing observation windows without exporting quotient stream equality, a selected host limit, countable choice, host real equality, or ambient Real completeness.

Definition 7306.1 (Regular Cauchy finite splice carrier)**.** *A* $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (k, X, Y, W, D, R, H, C, P, N)$$

with the following displayed rows:

1. *k is the unary splice-depth row naming the last replaced prefix index.*
2. *X is the original* $\mathsf{RegSeqRat}^{\uparrow}$ *stream row whose tail regularity is being retained.*
3. *Y is the replacement finite-prefix row, read only through displayed cells at indices $0, \ldots, k$.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that aligns the original and replacement observations at the splice boundary.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed prefix cells and the retained tail comparison.*
6. *R is the tail* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed after depth k.*
7. *H records componentwise* hsame *transport for the splice depth, original row, replacement prefix, window, dyadic ledger, tail readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the finite splice request through $k, X, Y, W, D, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{FinitePrefixStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSplice}^{\uparrow}}$ *row naming exactly this finite splice packet.*

The classifier compares accepted packets only through $k, X, Y, W, D, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyFiniteSpliceUp.RegularCauchyFiniteSpliceTasteGate single carrier alignment`

Theorem 7306.2 (Regular Cauchy finite splice NameCert obligations)**.** *For every accepted* RegularCauchyFiniteSpli *carrier $S = (k, X, Y, W, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSplice}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *splice-depth exposure: the unary row k is visible before any replacement cell is read;*
2. *prefix replacement locality: the replacement row Y supplies only the displayed cells at indices $0, \ldots, k$;*
3. *boundary-window alignment: the* $\mathsf{StreamName}^{\uparrow}$ *row W and dyadic ledger D expose the finite splice boundary before the retained tail is consumed;*
4. *tail readback preservation: the row R reads the post-k tail as regular rational data downstream of the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *route;*
5. *structural locality: the rows H, C, P, N preserve exactly the same splice packet under transport, replay, provenance, and naming.*

The obligation surface may cite the finite prefix certificate of chapter 7253 and the regular Cauchy shift route of chapter 1578; it cannot add quotient stream equality, selected host limits, countable choice, host real equality, ambient Real completeness, or a bridge certificate.

Proof. Project the carrier of Definition 7306.1. The finite analytic rows are exactly the splice depth k, the source row X, the replacement prefix Y, the window W, the dyadic ledger D, and the retained tail readback R.

Read the finite prefix certificate and the regular Cauchy shift route. The prefix certificate bounds the replacement cells by k, while the shift route ensures that all post-k reads are taken from the displayed tail rather than from an external selector.

The rows H, C, P, N then repack the same finite data under transport, replay, provenance, and naming. Since no excluded principle is a coordinate of the carrier or a result of the cited sibling routes, no local NameCert obligation can export quotient stream equality, selected host limits, countable choice, host real equality, ambient Real completeness, or bridge checking. □

Theorem 7306.3 (Regular Cauchy finite splice tail stability)**.** *Let $S = (k, X, Y, W, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$ *carrier. Every Real or Completion-facing consumer of the spliced name reads the post-k tail by the displayed route*

$$X, Y, k \longmapsto W \longmapsto D \longmapsto R \longmapsto H, C, P, N.$$

Thus finite prefix replacement preserves the regular Cauchy tail route after depth k: the consumer may inspect the replacement prefix and the retained tail readback, but it obtains no quotient equality of streams, no host sequence equality, no selected limiting value, no countable-choice tail selector, and no ambient completeness row.

Proof. Start with the NameCert obligations of Definition 7306.2. They force the splice depth, replacement prefix, boundary window, dyadic ledger, and retained readback to remain visible before any downstream real-name use.

Use the finite prefix certificate for the cells up to k. Those cells are the only part of the name supplied by Y, so the replacement cannot affect a post-k read except through the displayed boundary rows W and D.

Use the regular Cauchy shift route for the tail after k. The readback row R is downstream of that shift route and therefore preserves the regular tail as displayed RegSeqRat data.

Finally replay the structural rows H, C, P, N. They transport and name the same finite route, so the Real or Completion-facing consumer receives a stable tail route and none of the excluded quotient, equality, choice, limit-selection, or completeness principles. □

Theorem 7306.4 (Regular Cauchy finite splice nonescape)**.** *Every exported use of an accepted* $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$ *packet remains downstream of the displayed finite splice rows* $k, X, Y, W, D, R, H, C, P, N$ *The packet exports finite prefix replacement and retained tail regularity only; it does not export quotient stream equality, a selected host limit, countable choice, host real equality, ambient Real completeness, or a bridge certificate.*

Proof. Apply the carrier definition. The accepted packet has no field for quotient stream equality, selected host limits, countable choice, host real equality, ambient completeness, or bridge checking.

Apply tail stability from Definition 7306.3. Any Real or Completion-facing read passes through the finite replacement prefix and retained regular tail route before it reaches the structural rows.

The structural rows preserve the same displayed packet under hsame, Cont, Pkg, and local NameCert naming. They do not add analytic principles absent from the carrier, so the exported surface remains the finite splice certificate alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSplice}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFiniteSplice}^{\uparrow}$ packet is generated from a unary splice depth, original RegSeqRat row, replacement finite-prefix row, StreamName boundary window, DyadicRatCore tolerance ledger, retained RegSeqRat tail readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSplice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7306.1, Definition 7306.2, Definition 7306.3, and Definition 7306.4 seed the finite prefix-replacement route that preserves the displayed regular Cauchy tail after the splice depth.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSplice}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyFiniteSpliceUp.RegularCauchyFiniteSpliceTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked splice-depth admission, finite-prefix replacement locality, boundary-window alignment, dyadic tolerance exactness, tail readback stability, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFiniteSpliceUp.

Chapter 7307

A Concrete Naming Certificate for $\mathsf{SequentialFilterBridge}^{\uparrow}$

Following Bourbaki on Cauchy filters and Bishop–Bridges on regular Cauchy sequences, the $\mathsf{SequentialFilterBridge}^{\uparrow}$ packet records the finite bridge by which sequence-indexed Cauchy data is read as filter-style completion data without quotienting representatives. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window route of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational tail route of chapter 301, the saturated $\mathsf{CauchyFilter}^{\uparrow}$ surface of chapter 333, the $\mathsf{CauchyFilterBase}^{\uparrow}$ basis route of chapter 1915, and the separated completion boundary of chapter 402. The bridge exposes only displayed tails, finite windows, basis refinements, Cauchy-filter reads, separated-completion checks, transport, replay, provenance, and local naming rows; it does not add a quotient stream, selected representative, countable-choice schedule, host equality of limits, or ambient completeness.

Definition 7307.1 (Sequential-filter bridge carrier)**.** *A* $\mathsf{SequentialFilterBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, T, F, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly which sequence indices are inspected.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the inspected tail values and tolerances.*
3. *T is the sequential tail ledger recording unary tail bounds, tail inclusions, and finite requests made after a chosen depth.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated from those tail requests.*
5. *A is the* $\mathsf{CauchyFilterBase}^{\uparrow}$ *row listing the finite basis refinements used by F.*
6. *Q is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion boundary read after S, R, T, F, A are visible.*
7. *H records componentwise* hsame *transport for the stream-window, regular-readback, tail, filter, base, and separated-completion rows.*
8. *C records displayed* Cont *replay from a completion request through the sequence windows, regular tails, generated filter, basis refinements, separated boundary, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialFilterBridge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialFilterBridge}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient representatives, selected subsequences, selected filter points, countable choice, host equality of Cauchy names, or ambient completion.

Theorem 7307.2 (Sequential-filter bridge NameCert obligations)**.** *For every accepted* $\mathsf{SequentialFilterBridge}^{\uparrow}$ *carrier*

$$B = (S, R, T, F, A, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialFilterBridge}^{\uparrow}}$ *surface consists exactly of StreamName finite-window admission through* S*, RegSeqRat regular-tail readback through* R*, sequential tail-budget exposure through* T*, CauchyFilter generation through* F*, CauchyFilterBase refinement through* A*, HausdorffCompletion separated-boundary exposure through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7307.1. The sequence-facing rows are S, R, T, the filter-facing rows are F, A, Q, and the structural rows are H, C, P, N.

The sequence rows expose the finite means of the bridge. A completion request can inspect only the StreamName windows named by S, the regular rational readback supplied by R, and the tail-budget ledger T that says which later requests those windows answer.

The filter rows then consume that same displayed tail data. The generated Cauchy filter row F and basis row A are downstream of S, R, T, while the separated completion boundary Q is downstream of the filter-style read. The structural rows preserve the same dependency, so no quotient representative, selected filter point, countable-choice schedule, host equality, or ambient completeness object can appear in the local certificate. □

Theorem 7307.3 (Sequential-filter finite-window induction)**.** *Let* $B = (S, R, T, F, A, Q, H, C, P, N)$ *be an accepted* $\mathsf{SequentialFilterBridge}^{\uparrow}$ *carrier. Every Cauchy-filter or Cauchy-filter-base read exported by* B *is obtained by induction on finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *tail rows: the base read uses the displayed initial window and tail budget, and each successor read extends the same tail ledger before exposing the corresponding* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyFilterBase}^{\uparrow}$ *rows. The terminal* $\mathsf{HausdorffCompletion}^{\uparrow}$ *boundary is readable only after that finite induction has displayed* S, R, T, F, A*.*

Proof. The base step reads the first displayed finite window in S together with the matching regular tail row in R. The tail ledger T records that this finite observation already answers the initial filter request, so the generated F and basis row A have a visible source.

For the successor step, extend the inspected window by one finite tail request. The $\mathsf{StreamName}^{\uparrow}$ row records the new index set, the $\mathsf{RegSeqRat}^{\uparrow}$ row records the corresponding regular readback and tolerance, and T appends the tail-budget comparison without changing the earlier rows.

The filter-style read is then rebuilt from the extended finite data. The $\mathsf{CauchyFilter}^{\uparrow}$ row F consumes the extended tail request, the $\mathsf{CauchyFilterBase}^{\uparrow}$ row A records the basis refinement, and transport through H plus replay through C preserve the same finite route.

Thus every exported filter-style completion read is reached by finite induction on displayed sequence windows and regular tail rows. Since Q is downstream of F and A, the $\mathsf{HausdorffCompletion}^{\uparrow}$ boundary cannot be read before the induction has exposed the sequence-to-filter route. □

Theorem 7307.4 (Sequential-filter representative nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentialFilterBridge}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBase}^{\uparrow} \longmapsto \mathsf{HausdorffCompletion}^{\uparrow}.$$

The bridge exports a finite route from sequential Cauchy data to filter-style completion data, not a quotient class, selected representative, hidden limit, host equality proof, countable-choice schedule, or ambient completeness theorem.

Proof. Apply the NameCert obligations and the finite-window induction theorem. They force any completion-facing read to expose S, R, T before F, A, Q are public.

The displayed route is therefore sequence first, filter second, and separated completion last. The carrier has no row that can store a quotient class, selected representative, hidden limit, host equality proof, choice schedule, or ambient completeness witness.

Consequently every completion consumer receives only the finite bridge rows and their structural transport, replay, provenance, and local naming data. □

Falsifiable prediction: An accepted $\mathsf{SequentialFilterBridge}^{\uparrow}$ read is refuted if a $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, or $\mathsf{HausdorffCompletion}^{\uparrow}$ consumer reaches its row without first displaying the corresponding $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ tail readback, and finite tail ledger.

Independence witness: streamname, regseqrat, cauchyfilter, cauchyfilterbase, hausdorffcompletion: none is bijective with $\mathsf{SequentialFilterBridge}^{\uparrow}$ because the bridge jointly carries finite sequence windows, regular tail readback, tail budgets, generated Cauchy-filter rows, basis refinements, separated-completion boundary data, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialFilterBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialFilterBridge}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular tail rows, sequential tail budgets, CauchyFilter generation, CauchyFilterBase refinements, HausdorffCompletion separated boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialFilterBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7307.1, Definition 7307.2, Definition 7307.3, and Definition 7307.4 seed the finite sequence-to-filter completion bridge over StreamName windows, RegSeqRat tails, CauchyFilter rows, CauchyFilterBase refinements, and HausdorffCompletion boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialFilterBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host equality of limits, countable choice, ambient completeness, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{SequentialFilterBridge}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window exposure, RegSeqRat tail readback, finite-window induction, CauchyFilter generation, CauchyFilterBase refinement, HausdorffCompletion boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialFilterBridgeUp.

Chapter 7308

A Concrete Naming Certificate for $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the fan-theorem route for compact metric sources, the $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$ packet records the finite branch data by which compact-uniform continuity is read as a bounded finite-window induction. It sits between the fan-uniform route of chapter 6152, the compact metric source of chapter 129, the map graph of chapter 130, the downstream modulus surface of chapter 369, and the real-name seal of chapter 38. The packet names the constructive fan ledger; it does not assert an ambient open-cover theorem, a choice-selected branch, quotient function equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7308.1 (Constructive compact uniform fan carrier)**.** *A* $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, B, T, W, D, M, U, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact finite-net and metric-window data inspected by the fan.*
2. *B is the finite branch ledger, listing the branch prefixes over which the fan induction ranges.*
3. *T is the terminal-bar row, recording that every inspected branch reaches a displayed terminal prefix.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to the inspected branch prefixes.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and diameter row for the finite windows.*
6. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-continuity row whose local data is read at each branch prefix.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced after the branch ledger and tolerance rows have been exhausted.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for codomain distance and tolerance comparisons.*
9. *H records componentwise* hsame *transport for K, B, T, W, D, M, U, E and the local packet shape.*
10. *C records displayed* Cont *replay from compact-metric source data through the branch ledger, terminal-bar row, windows, dyadic tolerances, continuous-map reads, uniform-modulus handoff, and real seal.*

11. *P is the* Pkg *provenance over* ConstructiveCompactUniformFan$^{\uparrow}$, FanUniformContinuity$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{ConstructiveCompactUniformFan}^{\uparrow}}$ *row.*

The carrier exports no open-cover compactness theorem, infinite bar object, branch selector, quotient equality of functions, global choice coordinate, or ambient Real$^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.ConstructiveCompactUniformFanUp.ConstructiveCompactUniformFanTasteGate`
`single carrier alignment`

Theorem 7308.2 (Constructive compact uniform fan NameCert obligations)**.** *For every accepted* ConstructiveCompactUniformFan$^{\uparrow}$ *carrier*

$$F = (K, B, T, W, D, M, U, E, H, C, P, N),$$

the local NameCert$_{\text{ConstructiveCompactUniformFan}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through K, finite branch-ledger exposure through B, terminal-bar exposure through T, finite-window exposure through W, dyadic tolerance exposure through D, pointwise continuous-map exposure through M, uniform-modulus handoff through U, real sealing through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7308.1. The finite packet exposes exactly the compact source, branch ledger, terminal bar, windows, dyadic tolerances, continuous-map rows, uniform-modulus row, real seal, and structural rows.

Read the compact and fan rows first. The compact-metric row K supplies the inspected finite-net source, while B and T record the branch prefixes and the terminal-bar stops used by the fan-uniform route of chapter 6152.

The analytic rows are downstream of that finite branch data. The window row W and dyadic row D supply the finite observations and tolerance budgets; the continuous-map row M supplies pointwise continuity data at the displayed prefixes; the uniform-modulus row U is exposed only after those rows have been read.

Finally E, H, C, P, N seal, transport, replay, package, and name the same route. Since no additional coordinate exists, the listed rows exhaust the local NameCert surface and cannot export open-cover compactness, a selected branch, quotient function equality, global choice, or ambient Real completeness. □

Theorem 7308.3 (Constructive compact uniform fan finite-branch induction)**.** *Let $F = (K, B, T, W, D, M, U, E, H$ be an accepted* ConstructiveCompactUniformFan$^{\uparrow}$ *carrier. Every fan-uniform read accepted by F is exhausted by induction over the finite branch ledger B. At each branch prefix, the induction preserves the compact-metric source row K, the terminal-bar row T, the window row W, the dyadic tolerance row D, the continuous-map row M, the uniform-modulus handoff U, the real seal E, and the structural rows H, C, P, N.*

Consequently a downstream UniformModulus$^{\uparrow}$ *consumer receives only the displayed finite route*

$$K, B, T, W, D, M, U, E, H, C, P, N$$

and receives no branch selector, infinite fan object, host open-cover compactness theorem, quotient function equality, global choice coordinate, or ambient Real$^{\uparrow}$ *completeness witness.*

Proof. Induct on the displayed finite branch ledger B.

For the empty ledger, no branch prefix is available to inspect. The packet can therefore expose only the compact source and structural rows already present in the carrier, and no dyadic tolerance, continuous-map read, or uniform-modulus handoff can be detached from the displayed route.

For a successor branch prefix, first read the terminal-bar row T and the finite-window row W. These rows keep the successor prefix inside the same compact-metric source K and prevent the consumer from replacing it by a selected external branch.

Next consume the dyadic tolerance row D and the pointwise continuous-map row M for that prefix. The row D records the local diameter and tolerance budget, while M supplies only the displayed continuity data attached to the prefix.

The induction hypothesis preserves every earlier prefix with the same rows $K, T, W, D, M, U, E, H, C, P, N$. Combining the earlier route with the successor prefix exhausts the finite branch ledger, and the handoff row U reaches the $\mathsf{Real}^{\uparrow}$ seal E without importing open-cover compactness, quotient function equality, global choice, or ambient real completeness. □

Falsifiable prediction: A compact-uniform consumer that claims a $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$ read while omitting the compact-metric source row, finite branch ledger, terminal-bar row, finite-window row, dyadic tolerance row, continuous-map row, uniform-modulus handoff, Real seal, transport, replay, provenance, or local naming row refutes the carrier of Definition 7308.1.

Independence witness: fanuniformcontinuity, compactmetric, continuousmap, uniformmodulus, real: $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-metric source data, finite branch prefixes, terminal-bar stops, finite windows, dyadic tolerance budgets, pointwise continuous-map rows, a uniform-modulus handoff, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCompactUniformFan}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCompactUniformFan}^{\uparrow}$ packet is generated from a CompactMetric source row, a finite branch ledger, a terminal-bar row, StreamName finite windows, DyadicRatCore tolerance rows, ContinuousMap pointwise rows, a UniformModulus handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveCompactUniformFan}^{\uparrow}}$, seed)

Scope closed: Definition 7308.1, Definition 7308.2, and Definition 7308.3 seed the L10 compact-uniform fan route over compact-metric, finite-branch, continuous-map, uniform-modulus, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveCompactUniformFan}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, infinite fan objects, branch selectors, quotient function equality, global choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, terminal-bar exposure, finite-window coverage, dyadic tolerance exactness, pointwise continuous-map routing, uniform-modulus handoff, RealUp seal nonescape, and branch-selector refusal for BEDC.Derived.ConstructiveCompactUniformFanUp.

Chapter 7309

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ packet records the constructive located-real modulus surface as finite BEDC data before any $\mathsf{Real}^{\uparrow}$ or Completion consumer treats a located Cauchy name as a real object. It sits beside the located Cauchy interface of chapter 1850, the sequence-modulus surface of chapter 1801, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite modulus route for located real reads; it is not trichotomy, a Dedekind cut, quotient real equality, a selected limit, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 7309.1 (Bishop located real modulus carrier)**.** *A* $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *source row, exposing the located Cauchy real interface before any modulus request is read.*
2. *L is the* $\mathsf{LocatedSequenceModulus}^{\uparrow}$ *row, carrying the finite tolerance-to-window schedule for located real sequence reads.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the locatedness request and the regular Cauchy readback.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the modulus schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows and dyadic tolerances.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, L, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for B, L, D, S, R, E and the local packet shape.*
8. *C records displayed* Cont *replay from a located modulus request through the source row, modulus row, tolerance ledger, windows, readback, seal, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealModulus}^{\uparrow}$, $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$, $\mathsf{LocatedSequenceModulus}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, D, S, R, E, H, C, P, N$. It has no coordinate for trichotomy, Dedekind cuts, quotient real equality, selected limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7309.2 (Bishop located real modulus NameCert obligations)**.** *For every accepted* BishopLocatedRealMod *carrier $M = (B, L, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located Cauchy source row, the located sequence modulus row, the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, displayed replay, package provenance, and local naming.*

Every accepted obligation is read through those rows and through the sibling chapters named above. No obligation is stored in trichotomy, Dedekind cuts, quotient real equality, a selected limit, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the carrier of Definition 7309.1. The accepted tuple lists exactly the source, modulus, dyadic, stream, regular-readback, seal, transport, replay, provenance, and naming rows.

Read B and L first. The source row exposes the Bishop located Cauchy real interface, while the modulus row gives the finite tolerance-to-window schedule already displayed by the located sequence modulus chapter.

The analytic readback rows are downstream of that schedule. The dyadic row D, stream row S, and regular rational row R supply the finite data that a $\mathsf{Real}^{\uparrow}$ consumer may inspect before the terminal seal E is available.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no additional coordinate occurs in the carrier, the local NameCert surface cannot export any excluded real-completeness, quotient, selection, trichotomy, compactness, verification, or bridge object. □

Theorem 7309.3 (Bishop located real modulus RegSeqRat readback)**.** *Let $M = (B, L, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exported by M is reached through the finite route*

$$B, L, D, S \longmapsto R \longmapsto E, H, C, P, N.$$

Consequently the readback remains attached to the located Cauchy source, the displayed modulus schedule, the dyadic tolerance ledger, and the finite stream window; it cannot be replaced by a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the source row B and the modulus row L. A located real modulus request first fixes the located Cauchy interface and the finite schedule that tells which stream window is inspected.

The rows D and S then expose the tolerance ledger and the finite window. The regular rational readback R is read only after those rows are fixed, so it is not a detached representative or a quotient stream.

The terminal seal E and the structural rows H, C, P, N only preserve and replay this displayed route. Thus every $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains finite and carries no selected limit, host equality, countable choice, or ambient completeness coordinate. □

Theorem 7309.4 (Bishop located real modulus Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read by an accepted* $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ *carrier is downstream of B, L, D, S, R. The seal therefore records a terminal real-name boundary for the displayed located-modulus route and does not create trichotomy, Dedekind cuts, quotient real equality, selected limits, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Apply Definition 7309.2. The only seal coordinate in the accepted packet is E, and it is listed after the source, modulus, dyadic, stream, and regular-readback rows.

Apply Definition 7309.3. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff already forces the route through B, L, D, S, R before any Real-facing read may occur.

The structural rows H, C, P, N do not add a second seal or an external real object. Since the carrier contains no excluded coordinate, the terminal seal cannot escape to trichotomy, Dedekind cuts, quotient real equality, selected limits, open-cover compactness, ambient completeness, Lean verification, or bridge checking. □

Falsifiable prediction: A consumer that claims a $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ read while omitting the located Cauchy source row, located sequence modulus row, dyadic tolerance ledger, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, or local naming row refutes the carrier of Definition 7309.1.

Independence witness: bishoplocatedcauchyreal, locatedsequencemodulus, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located Cauchy source, reusable located sequence modulus, shared dyadic tolerance ledger, finite windows, regular rational readback, terminal seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealModulus}^{\uparrow}$ packet is generated from a BishopLocatedCauchyReal row, a LocatedSequenceModulus row, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7309.1, Definition 7309.2, Definition 7309.3, and Definition 7309.4 seed the finite located real modulus route over BishopLocatedCauchyReal, LocatedSequenceModulus, DyadicRatCore, StreamName, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, Dedekind cuts, quotient real equality, selected limits, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located sequence modulus exposure, dyadic tolerance exactness, RegSeqRat readback, RealUp seal nonescape, transport stability, replay preservation, provenance, and local naming targets for BEDC.Derived.BishopLocatedRealModulusUp.

Chapter 7310

Bishop Regular Cauchy Reindexing NameCert Construction

Following Bishop and Bridges, the $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ packet records the finite certificate by which a regular Cauchy name may be read along a monotone cofinal unary route without changing its Real-facing finite windows. It sits beside the regular rational readback of chapter 301, the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers stable reindexing data before a $\mathsf{Real}^{\uparrow}$ name consumes the source as public finite-window evidence; it does not import quotient representatives, countable choice, a selected host limit, host stream equality, or ambient completeness.

Definition 7310.1 (Bishop regular Cauchy reindexing carrier)**.** *A* $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, I, M, C, D, T, R, H, A, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *name, including its regular Cauchy modulus and finite rational readback rows.*
2. *I is the unary index-route row, displaying the finite graph of the reindexing map on every requested window.*
3. *M is the monotonicity row, certifying inside the packet that the displayed route is nondecreasing on the inspected unary indices.*
4. *C is the cofinality row, giving for every requested tail depth a displayed later source index reached by the route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic bounds used to compare the source and reindexed windows.*
6. *T is the tail-stability row, showing that every post-depth read of the reindexed name is backed by a post-depth read of S.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *finite-window read surface exported only after S, I, M, C, D, T are visible.*
8. *H records componentwise* hsame *transport for source, index route, monotonicity, cofinality, dyadic tolerance, tail stability, read surface, replay, provenance, and local naming rows.*

9. *A records displayed* Cont *replay from a reindexing request through* $S, I, M, C, D, T, R, H, P, N$.
10. P *is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyReindexing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, I, M, C, D, T, R, H, A, P, N$. *It has no coordinate for quotient stream representatives, selected limits, countable choice, host equality of sequences, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Theorem 7310.2 (Bishop regular Cauchy reindexing NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ *carrier* $J = (S, I, M, C, D, T, R, H, A, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyRe}}$ *surface consists exactly of eight obligation groups:*

1. *carrier admission for the source name, displayed index route, monotonicity row, cofinality row, dyadic tolerance row, tail-stability row, Real-facing read surface, transport, replay, provenance, and local naming;*
2. *source regularity readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S;
3. *classifier exactness through the displayed finite index route* I;
4. *monotonicity exposure through* M *before any tail comparison is consumed;*
5. *cofinality exposure through* C *before any downstream real-name read is exported;*
6. *dyadic tolerance preservation through* D *for every inspected source and reindexed window;*
7. *tail stability through* T, *binding each reindexed tail read to a later source tail read;*
8. *ledger, transport, replay, provenance, and local naming through* R, H, A, P, N.

Thus the packet names stable monotone cofinal reindexing of regular Cauchy names as finite BEDC data, not as quotient equality of streams or as a selected limit operation.

Proof. Project the carrier of Definition 7310.1. The analytic rows are the source regular Cauchy name S, the displayed index route I, the monotonicity row M, the cofinality row C, the dyadic tolerance row D, and the tail-stability row T.

Read the classifier and route obligations first. The route I is finite on each requested window, while M and C certify that the displayed route is monotone and eventually reaches every required source tail depth.

Next read the dyadic and tail rows. The row D preserves the requested tolerances under the source and reindexed windows, and T ensures that a reindexed tail read is backed by a later tail read of the original $\mathsf{RegSeqRat}^{\uparrow}$ name.

Finally R, H, A, P, N expose the Real-facing finite-window surface and preserve the same packet under transport, replay, provenance, and local naming. Since no excluded quotient, choice, selected-limit, host-equality, or completeness principle is a carrier coordinate, the listed rows exhaust the local obligation surface. □

Theorem 7310.3 (Bishop regular Cauchy reindexing tail stability)**.** *Let* $J = (S, I, M, C, D, T, R, H, A, P, N)$ *be an accepted* $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *finite-window consumer of the reindexed regular Cauchy name reads the same public surface through*

$$S, I, M, C \longmapsto D \longmapsto T \longmapsto R, H, A, P, N.$$

Consequently stable monotone cofinal reindexing preserves the $\mathsf{Real}^{\uparrow}$ *finite-window read surface of the source name: the consumer may inspect the reindexed windows and their displayed source tail witnesses, but it obtains no quotient representative, countable-choice selector, host stream equality, selected host limit, or ambient completeness row.*

Proof. Start with the NameCert obligations of Definition 7310.2. They require source regularity, route exactness, monotonicity, cofinality, dyadic tolerance preservation, and tail stability to be displayed before any Real-facing read.

Use monotonicity and cofinality together. Monotonicity prevents the displayed route from returning to earlier source windows, while cofinality supplies a later source index for every requested tail depth.

Use dyadic tolerance preservation next. The row D records the same requested tolerance against the source and reindexed windows, so the reindexed read has no independent tolerance scale.

Finally apply tail stability and the structural rows. The row T binds every reindexed tail read to a later source tail read, and R, H, A, P, N replay and name that finite route for the $\mathsf{Real}^{\uparrow}$ consumer. Since quotient representatives, choice selectors, host stream equality, selected host limits, and ambient completeness are absent from the carrier, the public read surface is preserved without importing them. □

Falsifiable prediction: A claimed $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ read that exports a $\mathsf{Real}^{\uparrow}$ finite-window surface without displaying monotonicity, cofinality, dyadic tolerance preservation, and tail stability is outside the carrier.

Independence witness: regseqrat, streamname, real: $\mathsf{RegSeqRat}^{\uparrow}$ supplies source regularity but not the monotone cofinal route, $\mathsf{StreamName}^{\uparrow}$ supplies finite observation windows but not tail cofinality, and $\mathsf{Real}^{\uparrow}$ supplies the terminal read surface but not the reindexing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyReindexing}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyReindexing}^{\uparrow}$ packet is generated from a source RegSeqRat row, a displayed unary index route, monotonicity, cofinality, dyadic tolerance preservation, tail stability, a RealUp finite-window read surface, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyReindexing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7310.1, Definition 7310.2, and Definition 7310.3 seed the finite certificate for stable monotone cofinal reindexing of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyReindexing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream representatives, countable choice, selected host limits, host stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, monotonicity, cofinality, dyadic tolerance preservation, tail stability, ledger, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyReir

Chapter 7311

A Concrete Naming Certificate for $\mathsf{BalancedTruncation}^{\uparrow}$

Following Moore's balanced realization construction and Antoulas's reduced-order modeling treatment, the $\mathsf{BalancedTruncation}^{\uparrow}$ packet records the finite certificate surface that turns reachability and observability energy rows into a reduced state-space model with an explicit error ledger. It sits after the state-space presentation of chapter 329, the Gramian surfaces of chapter 455 and chapter 5787, the Hankel window of chapter 6634, and the Riccati/control consumers of chapter 307, chapter 296, and chapter 348.

Definition 7311.1 (Balanced truncation carrier)**.** *A* $\mathsf{BalancedTruncation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, G_r, G_o, H, k, E, T, R, Q, C, P, N)$$

with the following displayed rows:

1. *S is the finite state-space row read through* $\mathsf{StateSpaceModel}^{\uparrow}$*, including the state, input, output, and transition matrices used by the reduction request.*
2. *G_r is the* $\mathsf{ReachabilityGramian}^{\uparrow}$ *row and G_o is the* $\mathsf{ObservabilityGramian}^{\uparrow}$ *row, both attached to the same displayed state-space row S.*
3. *H is the* $\mathsf{HankelOperator}^{\uparrow}$ *window comparing the reachable and observable energy coordinates visible in G_r and G_o.*
4. *k is the truncation-order row, bounded by the displayed state dimension and recorded as finite BEDC data rather than as an ambient spectral projection.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *error-bound row for the discarded Hankel window.*
6. *T is the reduced state-space row obtained by reading S, G_r, G_o, H, k, E and exposing only the retained coordinates.*
7. *R is the* $\mathsf{Riccati}^{\uparrow}$ *dependency row available to control consumers that compare reduced Riccati data with the source model.*
8. *Q records componentwise* hsame *transport for the state-space, Gramian, Hankel, order, error, reduced-model, Riccati, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every reduction request through $S, G_r, G_o, H, k, E, T, R, Q, P, N$.*

10. *P is the* Pkg *provenance over* BalancedTruncation$^{\uparrow}$, StateSpaceModel$^{\uparrow}$, ReachabilityGramian$^{\uparrow}$, ObservabilityGramian$^{\uparrow}$, HankelOperator$^{\uparrow}$, Riccati$^{\uparrow}$, LQR$^{\uparrow}$, ModelPredictiveControl$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BalancedTruncation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, G_r, G_o, H, k, E, T, R, Q, C, P, N$. It has no coordinate for an unlogged infinite-dimensional Hankel operator, hidden singular-vector basis, host norm equality, external optimality theorem, selected controller, or ambient completeness principle.

Theorem 7311.2 (Balanced truncation NameCert obligations)**.** *For every accepted* BalancedTruncation$^{\uparrow}$ *carrier $B = (S, G_r, G_o, H, k, E, T, R, Q, C, P, N)$, the local* NameCert$_{\text{BalancedTruncation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *common state-space admission, requiring G_r and G_o to be read as Gramian rows over the same displayed* StateSpaceModel$^{\uparrow}$ *row S;*
2. *Hankel-window exposure, requiring H to display the finite comparison window between reachable and observable energy rows before any truncation order is consumed;*
3. *order and error binding, requiring k and the* Real$^{\uparrow}$ *error-bound row E to be visible before the reduced row T is exported;*
4. *downstream control handoff, requiring Riccati, LQR, and model-predictive-control consumers to read only the displayed reduced row and its source rows;*
5. *ledger non-escape, requiring Q, C, P, N to transport, replay, source, and name only the displayed packet, with no hidden singular-vector basis, host norm equality, detached optimality theorem, selected controller, or ambient completeness coordinate.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 7311.1. The reachability and observability rows are both tied to the same state-space row S, so the energy comparison cannot be read from two unrelated models.

The Hankel row H is downstream of G_r and G_o. Thus the truncation order k and the error-bound row E are attached to a displayed finite Hankel window rather than to an unlogged operator or hidden basis.

The reduced row T is exported only after S, G_r, G_o, H, k, E are visible. Control consumers then receive a finite handoff through R and the provenance rows, not an external controller or an ambient optimality principle.

Finally, Q, C, P, N preserve and name the same coordinates. Since the refused objects do not occur in the carrier, the listed obligations exhaust the certificate surface. □

Theorem 7311.3 (Balanced truncation Hankel-Riccati handoff)**.** *Every downstream* Riccati$^{\uparrow}$, LQR$^{\uparrow}$, *or* ModelPredictiveControl$^{\uparrow}$ *consumer of an accepted* BalancedTruncation$^{\uparrow}$ *packet must pass through the displayed* HankelOperator$^{\uparrow}$ *window H, truncation-order row k,* Real$^{\uparrow}$ *error-bound row E, and reduced state-space row T before reading any Riccati or finite-horizon control data. The handoff exports no hidden singular-vector basis, detached Hankel norm equality, selected controller, infinite-horizon optimality theorem, or ambient state-space completion.*

Proof. Project the rows $S, G_r, G_o, H, k, E, T, R, Q, C, P, N$ from Definition 7311.1. The Riccati dependency row R appears only after the Hankel window, truncation order, error-bound row, and reduced model have been displayed.

Use the obligation surface of Definition 7311.2. Hankel-window exposure binds H to the two Gramian rows, while order and error binding force k, E, T to precede any exported reduced-control read.

The downstream handoff clause then restricts Riccati, LQR, and model-predictive-control consumers to the displayed reduced row and its provenance. Transport and replay through Q and C keep the handoff inside the same finite packet.

Because no hidden basis, detached norm equality, selected controller, infinite-horizon optimality theorem, or ambient completion coordinate is present in B, none can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BalancedTruncation}^{\uparrow}}$.

Constructive story: A $\mathsf{BalancedTruncation}^{\uparrow}$ packet is generated from a finite StateSpaceModelUp row, ReachabilityGramianUp and ObservabilityGramianUp rows over the same state-space source, a HankelOperatorUp comparison window, a truncation-order row, a RealUp error-bound row, a reduced state-space row, a RiccatiUp dependency row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BalancedTruncation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7311.1, Definition 7311.2, and Definition 7311.3 seed the finite reduced-order control route through state-space, Gramian, Hankel, truncation-order, error-bound, Riccati, LQR, and model-predictive-control rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BalancedTruncation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close analytic balanced realization existence, singular-vector computation, host Hankel norm equality, optimal reduced-order control, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked common state-space admission, Hankel-window exposure, order and error binding, downstream control handoff, transport, replay, provenance, and local naming for BEDC.Derived.BalancedTruncationUp.

Chapter 7312

A Concrete Naming Certificate for $\mathsf{ObserverModulusClock}^{\uparrow}$

The $\mathsf{ObserverModulusClock}^{\uparrow}$ packet realizes the modulus-as-temporal-rate route of Definition 9418.17 as finite NameCert data: a regular convergence request exposes a stream window, a modulus threshold, a regular rational readback, and a terminal real seal, and the observer reads the modulus as the local rate of the associated Hist time stream rather than as a global clock.

Definition 7312.1 (Observer modulus clock carrier)**.** *An* $\mathsf{ObserverModulusClock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, S, M, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the local* $\mathsf{HistTimeStream}^{\uparrow}$ *row, exposing the observer's finite Cont-step window and no global synchronization coordinate.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose observations are read by the regular convergence request.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, interpreted as the number of local time steps required before the requested precision window stabilizes.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same S, M route.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream window, modulus threshold, and regular readback have been displayed.*
6. *A is the observer-selection row, recording which local Hist time stream reads the modulus threshold.*
7. *H records componentwise* hsame *transport for the local time stream, stream window, modulus threshold, regular readback, real seal, observer-selection, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every observer precision request through $L, S, M, Q, R, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ObserverModulusClock}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverModulusClock}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, S, M, Q, R, A, H, C, P, N$. It has no coordinate for a global observer clock, synchronized Hist stream, hidden convergence oracle, selected real limit, quotient stream equality, host real equality, or ambient completeness principle.

Theorem 7312.2 (Observer modulus clock NameCert obligations)**.** *For every accepted* $\mathsf{ObserverModulusClock}^{\uparrow}$ *carrier $O = (L, S, M, Q, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ObserverModulusClock}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *local-time admission, requiring L and A to name the observer's finite Hist time stream before any modulus rate is read;*
2. *stream-window exposure, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row S to display the finite observations inspected by the request;*
3. *modulus-rate binding, requiring M to be read as a local temporal threshold for S before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q is consumed;*
4. *real-seal handoff, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row R to be downstream of L, S, M, Q, A;*
5. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet, with no global clock, synchronized Hist stream, hidden convergence oracle, selected real limit, quotient stream equality, host real equality, or ambient completeness coordinate.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 7312.1. The observer-selection row A chooses a local Hist time stream L, so the modulus read is attached to a finite Cont-step window rather than to a global clock.

The stream row S supplies the observations inspected by the regular convergence request. The modulus row M is consumed as the local threshold for that displayed window, and the regular readback Q is downstream of the same S, M route.

The real seal R is available only after the local time stream, observer selection, stream window, modulus threshold, and readback are visible. Thus a Real-facing consumer receives a finite temporal-rate route, not a selected limit or quotient stream.

Finally, H, C, P, N preserve and name exactly these rows. Since no global clock, synchronized Hist stream, hidden convergence oracle, host equality, or ambient completeness coordinate appears in the carrier, the listed obligations exhaust the certificate surface. □

Theorem 7312.3 (Observer modulus clock sibling route)**.** *Every accepted* $\mathsf{ObserverModulusClock}^{\uparrow}$ *read consumes the sibling rows of chapter 603, chapter 126, chapter 292, chapter 301, and chapter 38 before the observer-selection and temporal-rate rows can be read.*

Proof. Project the accepted carrier $O = (L, S, M, Q, R, A, H, C, P, N)$. The carrier of Definition 7312.1 displays L as the local Hist time stream, S as the finite StreamName window, M as the CauchyModulus threshold, Q as the RegSeqRat readback, and R as the terminal Real seal.

Apply the obligation surface of Definition 7312.2. Its local-time, stream-window, modulus-rate, and real-seal clauses force those sibling rows to be read before A may select the observer route or before M may be interpreted as a temporal rate.

The remaining rows H, C, P, N transport, replay, source, and name the same route. Since no global clock, hidden convergence oracle, selected limit, quotient stream equality, host real equality, or ambient completeness coordinate occurs in O, the accepted read cannot bypass the five sibling chapters listed in the statement. □

Falsifiable prediction: An accepted $\mathsf{ObserverModulusClock}^{\uparrow}$ read is refuted if a $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{RegSeqRat}^{\uparrow}$ readback is reached without displaying the local $\mathsf{HistTimeStream}^{\uparrow}$ row, observer-selection row, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{CauchyModulus}^{\uparrow}$ threshold, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: histtimestream, streamname, cauchymodulus, regseqrat, real: no listed sibling is bijective with $\mathsf{ObserverModulusClock}^{\uparrow}$ because this packet alone binds a local Hist time stream, observer selection, finite stream window, modulus-as-rate threshold, regular readback, Real seal, transport, replay, provenance, and local naming as one temporal-rate carrier.

The Lean follow-up for this chapter must provide explicit Nontrivial $\mathsf{ObserverModulusClock}^{\uparrow}$ and FieldFaithful $\mathsf{ObserverModulusClock}^{\uparrow}$ instances preserving $L, S, M, Q, R, A, H, C, P, N$ as distinct rows before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverModulusClock}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverModulusClock}^{\uparrow}$ packet is generated from a local HistTimeStreamUp row, StreamNameUp finite-window row, CauchyModulusUp threshold row, RegSeqRatUp readback row, terminal RealUp seal, observer-selection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverModulusClock}^{\uparrow}}$, seed)

Scope closed: Definition 7312.1 and Definition 7312.2 seed the finite observer-history route that reads a modulus as a local temporal rate over StreamName, CauchyModulus, RegSeqRat, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverModulusClock}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a global observer clock, inter-Hist synchronization, selected real limits, quotient stream equality, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked local-time admission, stream-window exposure, modulus-rate binding, real-seal handoff, ledger non-escape, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.ObserverModulusClockUp.

Chapter 7313

RiemannPartitionUp NameCert construction

Following Riemann and Darboux, the $\mathsf{RiemannPartition}^{\uparrow}$ packet records a finite subdivision of a displayed $\mathsf{Real}^{\uparrow}$ interval as NameCert data before tags, sums, or integrability claims are available. It sits after the dyadic partition route of chapter 1863, the equal-mesh route of chapter 2033, and the finite mesh-refinement surface of chapter 3496. The packet gives L10 interval consumers a standard finite-subdivision interface without importing host finite lists, quotient partition equality, selected mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7313.1 (RiemannPartition carrier)**.** *A* $\mathsf{RiemannPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, D, U, M, R, H, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval source row, carrying the named left and right endpoint seals through which the subdivision is read.*
2. *D is the* $\mathsf{DyadicPartition}^{\uparrow}$ *row, listing the dyadic breakpoint cells that refine the displayed interval.*
3. *U is the* $\mathsf{UniformPartition}^{\uparrow}$ *row, recording any equal-mesh window used by the same subdivision request.*
4. *M is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *row, exposing the mesh bound, adjacency, endpoint coverage, and common-refinement witness for the displayed cells.*
5. *R is the downstream readback row for Darboux, Riemann-sum, regulated-function, or interval-estimate consumers that inspect the subdivision only after I, D, U, M have been displayed.*
6. *H records componentwise* hsame *transport for interval endpoints, dyadic cells, uniform mesh data, finite mesh witnesses, downstream readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an interval-subdivision request through I, D, U, M, R, H, Q, N.*
8. *Q is the* Pkg *provenance over* $\mathsf{RiemannPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, U, M, R, H, C, Q, N$. It has no coordinate for tagged Riemann sums, Darboux equality, host finite-list equality, quotient partitions, selected mesh limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7313.2 (RiemannPartition mesh refinement)**.** *Every accepted* $\mathsf{RiemannPartition}^{\uparrow}$ *carrier* $P = (I, D, U, M, R, H, C, Q, N)$ *refines its displayed* $\mathsf{Real}^{\uparrow}$ *interval through the existing* $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{UniformPartition}^{\uparrow}$, *and* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *carrier rows before any subdivision consumer can read the packet. The refinement route exposes dyadic cells through* D*, equal-mesh windows through* U *when present, and mesh-bound and common-refinement witnesses through* M*; it exports no Riemann integrability theorem, Darboux equality, host finite-list presentation, quotient partition equality, or selected limiting partition.*

Proof. Project the accepted carrier of Definition 7313.1. The interval source I is the only endpoint row, while D, U, M are the only subdivision and mesh rows available to a consumer.

Read the dyadic side through chapter 1863. The row D supplies the finite dyadic cells and their ordered interval coverage, so a consumer cannot replace the subdivision by a host list of breakpoints.

Next read the equal-mesh and mesh-refinement rows through chapter 2033 and chapter 3496. These rows expose uniform windows, mesh bounds, adjacency, endpoint coverage, and common-refinement data before R is public.

Finally replay the structural rows H, C, Q, N. Transport, continuation replay, provenance, and local naming preserve the same finite subdivision packet, so the accepted route remains a NameCert interface for interval work rather than an integrability theorem, Darboux equality, quotient partition principle, selected mesh-limit construction, or host finite-list assertion. □

Falsifiable prediction: A $\mathsf{RiemannPartition}^{\uparrow}$ consumer that reads a subdivision without exposing the $\mathsf{Real}^{\uparrow}$ interval row, dyadic partition row, optional uniform-mesh row, finite mesh-refinement row, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: dyadic-partition, uniform-partition, finite-partition-mesh: $\mathsf{RiemannPartition}^{\uparrow}$ is not bijective with any listed sibling because it joins the interval source, dyadic cells, optional uniform window, and common mesh-refinement handoff as one subdivision packet for downstream interval consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannPartition}^{\uparrow}$ packet is generated from a RealUp interval row, DyadicPartition cells, optional UniformPartition mesh windows, FinitePartitionMesh refinement witnesses, downstream readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannPartition}^{\uparrow}}$, seed)

Scope closed: Definition 7313.1 and Definition 7313.2 seed the finite Real interval subdivision interface through DyadicPartitionUp, UniformPartitionUp, and FinitePartitionMeshUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Riemann integrability, equality of Darboux upper and lower values, host finite-list presentation, quotient partition equality, selected mesh limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-cell coverage, uniform-window compatibility, mesh-refinement exactness, downstream readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannPartitionUp.

Chapter 7314

A Concrete Naming Certificate for $\mathsf{CountableProductMetric}^{\uparrow}$

Following Bourbaki and Kelley on product uniformities and Bishop–Bridges constructive metric practice, the $\mathsf{CountableProductMetric}^{\uparrow}$ packet records the finite-prefix product metric used when certified metric streams are read by Real and Completion consumers. It sits over the finite schedule of chapter 126, the metric carrier surface of chapter 57, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the finite-prefix weighted route only; it is not a host countable product, a selected family of points, a quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7314.1 (Countable product metric carrier)**.** *A* $\mathsf{CountableProductMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, M, D, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-prefix schedule. It records which coordinate windows of the certified stream are visible to the consumer.*
2. *M is the finite family of* $\mathsf{Met}^{\uparrow}$ *rows indexed by the visible prefix of S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *weighted finite-prefix tolerance ledger. It records coordinate weights, requested coordinate tolerances, the finite product-radius budget, and the comparison allowance used by the packet.*
4. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$ *seal handoff that reads the weighted finite-prefix sum as a regular rational name before any terminal real seal is consumed.*
5. *H records componentwise* hsame *transport for the stream schedule, metric rows, dyadic ledger, regular-real handoff, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a product-metric request through S, M, D, W.*
7. *P is the* Pkg *provenance over* $\mathsf{CountableProductMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableProductMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, M, D, W, H, C, P, N. It has no coordinate for host countable products, a choice-selected stream of points, infinite sums, quotient completion equality, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7314.2 (Countable product metric NameCert obligations)**.** *For every accepted* $\mathsf{CountableProductMetric}^{\uparrow}$ *carrier* $Q = (S, M, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CountableProductMetric}^{\uparrow}}$ *surface consists exactly of finite-prefix schedule admission, metric-row admission for the visible coordinates, weighted dyadic tolerance accounting, regular-rational readback, terminal Real seal handoff, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of host countable products, choice-selected point streams, infinite sums, quotient completion equality, detached real carriers, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 7314.1. The accepted packet exposes exactly the finite schedule S, the metric rows M over the visible prefix, the dyadic ledger D, the regular-real handoff W, and the structural rows H, C, P, N.

The finite-prefix route reads S before it reads M or D. Thus the product metric read is bounded by the displayed prefix and cannot import a host countable product or a choice-selected infinite point stream.

The weighted ledger D is then read before W. It supplies the coordinate weights, coordinate tolerances, and product-radius budget that the $\mathsf{RegSeqRat}^{\uparrow}$ readback turns into the terminal $\mathsf{Real}^{\uparrow}$ seal handoff.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores infinite sums, quotient completion equality, detached real carriers, ambient Real completeness, Lean verification, or bridge checking, the listed obligations exhaust the local certificate surface. □

Theorem 7314.3 (Countable product metric finite-prefix stability)**.** *Let* $Q = (S, M, D, W, H, C, P, N)$ *be an accepted* $\mathsf{CountableProductMetric}^{\uparrow}$ *packet. If* $Q' = (S', M', D', W', H', C', P', N')$ *is obtained by componentwise* hsame *transport of the finite schedule, visible metric rows, weighted dyadic ledger, regular-real handoff, continuation, provenance, and local naming rows, then* Q' *preserves the same finite-prefix product metric read. The transported packet may rename the displayed prefix and rows, but it may not add an unseen coordinate, an infinite product metric, quotient completion equality, a selected limit, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with carrier admission for Q. The finite schedule S determines exactly which coordinate metric rows can be inspected, and the ledger D is attached to that same prefix.

Apply componentwise hsame transport. The transported schedule S' names the same visible prefix up to the history classifier, while M' and D' are the transported metric rows and weighted ledger over that prefix.

Replay the displayed continuation. The row C' reads the transported route through S', M', D', W' in the same order, so the regular-real handoff preserves the product metric value exposed by the finite prefix.

The exclusions follow from the carrier shape. No coordinate of Definition 7314.1 stores unseen coordinates, infinite product metrics, quotient completion equality, selected limits, or ambient Real completeness, and transport cannot create rows absent from the packet. □

Theorem 7314.4 (Countable product metric Cauchy consumer route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion consumer of an accepted* $\mathsf{CountableProductMetric}^{\uparrow}$ *packet must first expose the finite-prefix weighted ledger* D *together with the schedule* S *and metric rows* M *before reading the handoff row* W*. The Cauchy-facing seal is therefore a routed finite-prefix weighted read, not a host completion of a countable product, a quotient of streams, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing read of $Q = (S, M, D, W, H, C, P, N)$. Carrier admission makes the schedule and visible metric rows available before any regular-rational or Real-facing seal can be read.

The dyadic ledger D records the weights and tolerance budgets for that displayed prefix. A consumer that omits D has no certified product-radius budget and therefore has no $\mathsf{CountableProductMetric}^{\uparrow}$ metric read to pass to $\mathsf{RegSeqRat}^{\uparrow}$.

Now read the handoff row W. It turns the finite weighted ledger into the regular rational route and then into the terminal $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep the readback tied to the same displayed rows.

Thus every Cauchy or Real completion consumer factors through S, M, D, W, H, C, P, N in order. The packet exports only that finite-prefix weighted route and none of the host product, quotient, selected-limit, choice, or ambient-completeness exits. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableProductMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CountableProductMetric}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ finite-prefix schedule, visible $\mathsf{Met}^{\uparrow}$ rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ weighted tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableProductMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7314.1, Definition 7314.2, Definition 7314.3, and Definition 7314.4 seed the finite-prefix weighted product metric route for certified metric streams.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableProductMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host countable products, choice-selected point streams, infinite sums, quotient completion equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weighted finite-prefix ledger exactness, finite-prefix transport stability, Cauchy consumer routing, replay, provenance, and local naming for BEDC.Derived.CountableProductMetricUp.

Chapter 7315

A Concrete Naming Certificate for $\mathsf{BauerSimplex}^{\uparrow}$

Following Bauer and Choquet compact-convex theory, the $\mathsf{BauerSimplex}^{\uparrow}$ packet records the finite NameCert surface where a Choquet simplex is read with a closed extreme-boundary handoff. It sits over the Choquet simplex carrier of chapter 6611, the compact source of chapter 60, the convex-set interface of chapter 208, the Radon-measure readback route of chapter 2263, and the finite extreme-point witness surface of chapter 3610. The chapter names the Bauer-simplex packet only as a finite compact-convex certificate with a closed extreme-boundary route; it does not assert selected representing measures, host simplex equality, quotient measure equality, ambient locally convex completeness, or a bridge theorem.

Definition 7315.1 (Bauer simplex carrier)**.** *A* $\mathsf{BauerSimplex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, K, V, R, E, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{ChoquetSimplex}^{\uparrow}$ *source row, carrying the compact-convex probability-boundary interface already named by the Choquet simplex packet.*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *row whose compact source observations bound the ambient simplex and the extreme-boundary request.*
3. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row exposing affine-combination and convex subcarrier data available to the packet.*
4. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *readback row for finite probability-window and boundary-facing measure data.*
5. *E is the* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *finite extreme-witness row, opened only as displayed exposed-face and candidate-extreme evidence.*
6. *Z is the closed extreme-boundary ledger, recording finite closedness tests for the boundary route after S, K, V, R, E have been displayed.*
7. *H records componentwise* hsame *transport for Choquet, compact, convex, Radon, finite extreme, closed-boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-convex probability requests through the finite extreme-boundary route.*

9. *P is the* Pkg *provenance over* $\mathsf{BauerSimplex}^{\uparrow}$, $\mathsf{ChoquetSimplex}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BauerSimplex}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, K, V, R, E, Z, H, C, P, N$. It has no coordinate for a host Bauer simplex object, selected barycentric measure, quotient measure equality, ambient locally convex completeness, choice-selected extreme point, or bridge witness.

Theorem 7315.2 (Bauer simplex NameCert obligations)**.** *For every accepted* $\mathsf{BauerSimplex}^{\uparrow}$ *carrier $B = (S, K, V, R, E, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BauerSimplex}^{\uparrow}}$ *surface consists exactly of Choquet-simplex source admission through S, compact-source exposure through K, convex-carrier exposure through V, Radon-measure readback through R, finite extreme-witness admission through E, closed extreme-boundary ledger visibility through Z, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every compact-convex probability-boundary consumer of the packet must read

$$S, K, V, R, E, Z, H, C, P, N$$

before treating the packet as Bauer-simplex evidence. The route exports no selected representing measure, quotient measure equality, host closed-boundary theorem, ambient locally convex completeness, or bridgeChecked evidence.

Proof. Project the carrier of Definition 7315.1. The Choquet source, compact source, convex source, Radon readback, finite extreme witness, and closed extreme-boundary ledger are exactly the rows S, K, V, R, E, Z.

Read the source rows first. The Choquet simplex row already packages compact convex probability-boundary data, while K and V keep the compact and affine convex observations displayed as separate finite rows.

Next attach measure and extreme-boundary data. The Radon row R supplies only finite regular-measure readback, the Krein–Milman row E supplies only finite exposed-face and candidate-extreme evidence, and Z records closed extreme-boundary tests over those displayed rows.

Finally H, C, P, N transport, replay, source, and name the same route. Since no remaining coordinate stores a selected representing measure, quotient measure equality, host closed-boundary theorem, ambient locally convex completeness, or bridge witness, the listed rows exhaust the local NameCert obligations. □

Theorem 7315.3 (Bauer simplex closed extreme-boundary handoff)**.** *Every closed extreme-boundary read exported by an accepted* $\mathsf{BauerSimplex}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{ChoquetSimplex}^{\uparrow}, \quad \mathsf{Compact}^{\uparrow}, \quad \mathsf{ConvexSet}^{\uparrow}, \quad \mathsf{RadonMeasure}^{\uparrow}, \quad \mathsf{KreinMilmanFiniteExtreme}^{\uparrow}.$$

The handoff reaches the boundary ledger Z only after the Choquet source, compact source, convex affine row, Radon readback row, and finite Krein–Milman extreme-witness row have been displayed. It exports no selected extreme point, no global representation theorem, no quotient measure equality, no host closed-subset equality, and no ambient compact-convex completeness principle.

Proof. Start with the Choquet source. chapter 6611 supplies the compact-convex probability-boundary surface but refuses representing-measure uniqueness and selected extreme points.

Next read the compact and convex rows. The compact source of chapter 60 and the convex carrier of chapter 208 remain visible as K and V, so the handoff cannot replace them by a host compact convex body.

Now attach measure and finite extreme evidence. chapter 2263 contributes only the displayed regular-measure readback, while chapter 3610 contributes only finite exposed-face and candidate-extreme rows.

Therefore the closed extreme-boundary ledger Z is downstream from the listed route. Because the carrier has no selected extreme point, global representation theorem, quotient measure equality, host closed-subset equality, or ambient compact-convex completeness row, the handoff is exactly the finite boundary route stated above. □

Falsifiable prediction: An accepted $\mathsf{BauerSimplex}^{\uparrow}$ consumer that reaches a closed extreme-boundary read without first displaying the $\mathsf{ChoquetSimplex}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, and $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: choquet-simplex, compact, convexset, radon-measure, krein-milman-finite-extreme: no listed sibling is bijective to $\mathsf{BauerSimplex}^{\uparrow}$, because the packet jointly carries Choquet simplex source data, compact and convex rows, Radon readback, finite extreme-witness data, a closed extreme-boundary ledger, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BauerSimplex}^{\uparrow}}$.

Constructive story: A $\mathsf{BauerSimplex}^{\uparrow}$ packet is generated from a Choquet simplex source row, compact source row, convex-set row, Radon-measure readback, finite Krein–Milman extreme-witness row, closed extreme-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BauerSimplex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7315.1, Definition 7315.2, and Definition 7315.3 seed the finite Bauer-simplex certificate over Choquet, compact, convex, Radon, finite extreme-witness, closed-boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BauerSimplex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected representing measures, quotient measure equality, host closed-boundary theorems, ambient locally convex completeness, Lean verification, or bridge checking for $\mathsf{BauerSimplex}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, Choquet-source exposure, compact and convex source visibility, Radon readback, finite extreme-witness admission, closed extreme-boundary exactness, transport, replay, provenance, and local naming for BEDC.Derived.BauerSimplexUp.

Chapter 7316

A Concrete Naming Certificate for $\mathsf{TopologistSineCurve}^{\uparrow}$

Following Kelley and classical point-set topology, the $\mathsf{TopologistSineCurve}^{\uparrow}$ packet records the finite NameCert surface by which the oscillatory graph of $x \mapsto \sin(1/x)$ and its limiting vertical segment are read as Real- and topology-facing boundary data. It sits after the Real carrier of chapter 38, the finite stream window of chapter 126, the regular rational readback of chapter 301, the topology handoff of chapter 88, and the continuous-map route of chapter 130. The packet names only the finite boundary example; it does not assert connectedness, compactness, quotient equality, host subset equality, Real completeness, or a bridge theorem.

Definition 7316.1 (Topologist sine curve carrier)**.** *A* $\mathsf{TopologistSineCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, V, W, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *G is the oscillatory graph-window row, recording finitely many rational observation windows for the graph-side read of $x \mapsto \sin(1/x)$ away from the limiting axis.*
2. *V is the limiting vertical-segment row, recording the displayed endpoint and interval observations used by the boundary read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window that names the graph and segment samples before any Real readback is consumed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the finite stream observations into regular rational approximants.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the graph and limiting-segment coordinates read through W and R.*
6. *T is the topology and continuous-map handoff row, available only after the graph window, segment row, stream window, regular-rational readback, and Real seal have been displayed.*
7. *H records componentwise* hsame *transport for graph-window, segment, stream, regular-rational, Real, topology, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from graph-window and segment requests through $G, V, W, R, E, T, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{TopologistSineCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{TopologistSineCurve}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, V, W, R, E, T, H, C, P, N$. It has no coordinate for a host subset of $\mathbb{R}^2$, quotient equality, Real completeness, compactness, connectedness, path-connectedness, or ambient topological closure.

Theorem 7316.2 (Topologist sine curve NameCert obligations)**.** *For every accepted* $\mathsf{TopologistSineCurve}^{\uparrow}$ *carrier $S = (G, V, W, R, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TopologistSineCurve}^{\uparrow}}$ *surface consists exactly of oscillatory graph-window admission through G, limiting vertical-segment visibility through V, finite stream observation through W, regular rational readback through R, terminal Real sealing through E, topology and continuous-map handoff through T, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every boundary consumer of the packet must read

$$G, V, W, R, E, T, H, C, P, N$$

before treating the packet as topologist-sine-curve evidence. The route exports no connectedness theorem, compactness theorem, quotient equality, host subset equality, Real completeness principle, or bridgeChecked evidence.

Proof. Project the carrier of Definition 7316.1. The graph-window row G and vertical-segment row V are separate displayed coordinates, so the limiting segment is not inferred from an ambient subset closure.

Read the finite observation rows next. The stream window W supplies only finite samples, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R converts those samples into regular rational readback before the terminal Real seal E is available.

The topology and continuous-map handoff row T is downstream from G, V, W, R, E. Thus a consumer may cite the topology-facing boundary route only after the Real-facing finite data has already been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores host subset equality, quotient equality, Real completeness, connectedness, compactness, or bridge evidence, the listed rows exhaust the local NameCert obligations. □

Theorem 7316.3 (Topologist sine curve Real-topology handoff)**.** *Every topology-facing read of an accepted* $\mathsf{TopologistSineCurve}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{Topology}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}.$$

The handoff reaches the topology and continuous-map row T only after the oscillatory graph window G, limiting vertical-segment row V, finite stream window W, regular rational readback row R, and terminal Real seal E have been displayed. It exports no connectedness claim, compactness claim, quotient equality, host subset equality, or Real completeness principle.

Proof. Start with the graph and segment rows. The graph-window coordinate G records only finite oscillatory observations, while V records the displayed limiting-segment observations; neither row is a host subset of the plane.

Next read the Real-facing route. chapter 126 supplies the finite observation window, chapter 301 supplies the regular rational readback, and chapter 38 supplies the terminal Real seal.

Only after those rows are visible may the packet reach chapter 88 and chapter 130. The topology-facing row is therefore a handoff from finite Real data, not a replacement for it.

Because the carrier has no row for connectedness, compactness, quotient equality, host subset equality, or Real completeness, the handoff cannot export any of those endpoints. □

Falsifiable prediction: An accepted $\mathsf{TopologistSineCurve}^{\uparrow}$ consumer that reads the topology handoff without first displaying G, V, W, R, E refutes the carrier surface.

Independence witness: real, streamname, regseqrat, topology, continuousmap: none carries the oscillatory graph-window row, limiting vertical-segment row, finite stream window, regular rational readback, terminal Real seal, topology handoff, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TopologistSineCurve}^{\uparrow}}$.

Constructive story: A $\mathsf{TopologistSineCurve}^{\uparrow}$ packet is generated from an oscillatory graph-window row, limiting vertical-segment row, StreamName finite observation window, RegSeqRat readback, terminal RealUp seal, topology and continuous-map handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TopologistSineCurve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7316.1, Definition 7316.2, and Definition 7316.3 seed the finite Real/topology boundary certificate over graph-window, segment, stream, regular-rational, Real, topology, continuous-map, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TopologistSineCurve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close connectedness, compactness, quotient equality, host subset equality, Real completeness, Lean verification, or bridge checking for $\mathsf{TopologistSineCurve}^{\uparrow}$.

Upgrade path: Obligation closure requires checked graph-window admission, limiting vertical-segment visibility, finite stream observation, regular rational readback, terminal Real sealing, topology and continuous-map handoff, transport, replay, provenance, and local naming for BEDC.Derived.TopologistSineCurveUp.

Chapter 7317

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy construction carries its usable modulus as displayed finite evidence, not as a selected limit or a quotient completion. The $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ packet records that evidence on the L10 Real and Completion route: it reads dyadic tolerances from chapter 291, finite windows from chapter 126, regular rational schedules from chapter 301, and the terminal Real seal from chapter 38. The packet is a constructive Cauchy-modulus witness; it is not a quotient, selected limit, countable-choice object, or $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7317.1 (Bishop Cauchy modulus witness carrier)**.** *A* $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, Q, A, R, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic precision and the rational comparisons used to inspect it.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, recording the finite windows opened on the source stream before any modulus claim is read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row, recording the regular rational readbacks whose tails must be controlled.*
4. *A is the* $\mathsf{Rat}^{\uparrow}$ *arithmetic row, carrying the rational addition, subtraction, and bound comparisons used inside the modulus witness.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *handoff row, recording only the terminal seal route available after the displayed rational schedule has been inspected.*
6. *M is the modulus-witness row, assigning each displayed dyadic tolerance in D a finite index boundary for the schedule in Q.*
7. *H records componentwise* hsame *transport for D, S, Q, A, R, M and the structural rows.*
8. *C records displayed* Cont *replay from a tolerance request through stream windows, rational readbacks, rational arithmetic, the modulus witness, and the terminal seal route.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7317.2 (Bishop Cauchy modulus witness obligation surface)**.** *For the carrier of Definition 7317.1, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite stream-window exposure, regular rational schedule readback, rational arithmetic accounting, terminal Real-seal handoff, modulus-witness admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier $W = (D, S, Q, A, R, M, H, C, P, N)$. The rows D, S, Q, A, R, M expose the tolerance request, source windows, regular rational schedule, rational arithmetic checks, terminal Real handoff, and displayed modulus witness.

The structural rows H, C, P, N preserve that same finite inventory. Since the carrier lists no quotient, selected limit, choice schedule, host real equality, or ambient completeness row, the named rows exhaust the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}$ obligation surface. □

Theorem 7317.3 (Bishop Cauchy modulus witness induction)**.** *Every accepted* $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ *read is determined by finite-window induction on the displayed tolerance requests and the stream windows opened for those requests. At each induction step, the witness row M may inspect only the current* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, the current* $\mathsf{StreamName}^{\uparrow}$ *window, the current* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule segment, and the* $\mathsf{Rat}^{\uparrow}$ *arithmetic row before extending the same packet through H, C, P, N.*

Proof. The base case is the empty inspection window. The displayed rows carry no quotient class, limit point, or host real equality, so the empty route is the unique admitted packet prefix.

For the induction step, extend the finite window by one displayed tolerance request. The row D supplies the dyadic precision, S supplies the newly opened stream window, Q supplies the matching regular schedule segment, and A supplies the rational comparisons needed by the modulus witness.

The row M records the finite index boundary for that request. The rows H, C, P, N then transport, replay, source, and name the same packet prefix, so the induction step stays inside the displayed carrier. □

Theorem 7317.4 (Bishop Cauchy modulus witness Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ *packet must first read the dyadic tolerance row, finite stream windows, regular rational schedule, rational arithmetic row, and modulus-witness row. The terminal Real seal row is therefore a handoff from displayed modulus evidence, not a completeness principle and not a selected limit.*

Proof. Use Definition 7317.2. Its obligation surface places dyadic tolerance exposure, stream-window exposure, regular rational schedule readback, rational arithmetic accounting, and modulus-witness admission before the terminal Real-seal handoff.

Apply Definition 7317.3. Every finite consumer route is built by extending displayed tolerance and stream windows, so the Real-facing row can only be reached after those finite rows have been inspected.

Finally inspect the carrier inventory. The packet has no row for ambient completeness, selected limits, quotient real equality, countable choice, or host real equality; the Real handoff is exactly the displayed seal route R preserved by H, C, P, N. □

Falsifiable prediction: A consumer that reads a $\mathsf{Real}^{\uparrow}$ seal from a regular Cauchy schedule while bypassing the displayed dyadic tolerance row, stream-window row, rational arithmetic row, or modulus-witness row is not reading a $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ packet.

Independence witness: dyadicratcore, streamname, regseqrat, real: none is bijective to BishopCauchyModulusWi because the witness packet jointly carries tolerance requests, finite windows, regular rational schedules, rational arithmetic checks, a displayed modulus row, terminal seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyModulusWitness}^{\uparrow}$ packet is generated from dyadic tolerance requests, finite stream windows, regular rational schedule rows, rational arithmetic checks, a displayed modulus-witness row, terminal Real-seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7317.1, Definition 7317.2, Definition 7317.3, and Definition 7317.4 seed the displayed Cauchy-modulus witness over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion, selected limits, countable choice, host real equality, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exposure, stream-window exposure, regular schedule readback, rational arithmetic accounting, modulus-witness admission, terminal seal handoff, transport, replay, provenance, and local naming.

Chapter 7318

A Concrete Naming Certificate for $\mathsf{ClaimAuditBoundary}^{\uparrow}$

The $\mathsf{ClaimAuditBoundary}^{\uparrow}$ packet realizes the legitimate-claim audit surface of chapter 9456 as a finite NameCert boundary. It sits beside the writing-item packet of chapter 1222, the named externality boundary of chapter 543, the registry layer of chapter 1220, and the object-knowledge certificate of chapter 1218. Those siblings regulate item-level claim text, apophatic externality, registry admission, and licensed object knowledge; this packet binds the finite expression, claim-status row, refusal or gap/socket row, and permitted-claim row before a theorem-shaped consumer may publish the claim.

Definition 7318.1 (Claim audit boundary carrier)**.** *A* $\mathsf{ClaimAuditBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, G, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite communicable-expression row, recording the encoded expression being audited.*
2. *S is the claim-status row, read from the theorem, conditional, route, evidence, non-evidence, meta, refusal,* GAP*, or socket statuses of Definition 9456.6.*
3. *G is the gap or socket row, recording the boundary position whenever the expression requests supply not produced by the closed substrate.*
4. *R is the refusal row, recording the visible refusal that prevents hidden supply from being read as theorem content.*
5. *Q is the permitted-claim row, recording the strongest claim text licensed after E, S, G, R have been displayed.*
6. *H records componentwise* hsame *transport for expression, status, gap or socket, refusal, permitted claim, and the structural rows.*
7. *C records displayed* Cont *replay from expression admission through status assignment, boundary exposure, refusal readback, and permitted-claim publication.*
8. *P is the* Pkg *provenance over* $\mathsf{ClaimAuditBoundary}^{\uparrow}$*,* $\mathsf{ClaimStatusAudit}^{\uparrow}$*,* $\mathsf{WritingItemAuditPacket}^{\uparrow}$*,* $\mathsf{ApophaticName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClaimAuditBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a theorem row that bypasses the displayed status, refusal, gap or socket, and permitted-claim rows.

Theorem 7318.2 (Claim audit boundary NameCert obligations)**.** *For the carrier of Definition 7318.1, the local* $\mathsf{NameCert}_{\mathsf{ClaimAuditBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-expression readback, claim-status readback, gap or socket exposure, refusal exposure, permitted-claim readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the packet $B = (E, S, G, R, Q, H, C, P, N)$. The rows E, S, G, R, Q expose the expression under audit, the status assigned to it, the gap or socket boundary, the refusal row, and the permitted claim.

The rows H, C, P, N preserve that displayed boundary by transport, replay, provenance, and local naming. Since the carrier has no hidden theorem, status-free claim, refusal-free read, or gap-erasing coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7318.3 (Claim audit boundary status nonescape)**.** *No accepted* $\mathsf{ClaimAuditBoundary}^{\uparrow}$ *packet permits a theorem row, route row, evidence row, or object-knowledge claim to escape the displayed claim-status row S, refusal row R, gap or socket row G, and permitted-claim row Q. If a consumer publishes a claim stronger than Q or bypasses G or R, the consumer is no longer reading the accepted packet.*

Proof. Begin with the obligation surface of Definition 7318.2. Every accepted read exposes S, G, R, Q before local naming, so the status assignment, boundary row, refusal row, and permitted text are part of the public packet.

Read the replay row C. It orders expression admission, status assignment, boundary exposure, refusal readback, and permitted-claim publication; therefore claim publication cannot be separated from the displayed status and boundary rows.

Finally inspect the carrier inventory. There is no row for a hidden theorem, unmarked object-knowledge export, gap-free theorem upgrade, or refusal-free claim. The structural rows H, P, N only preserve and name the same packet, so any stronger or bypassing read exits $\mathsf{ClaimAuditBoundary}^{\uparrow}$. □

Falsifiable prediction: If a consumer publishes a theorem row while bypassing the displayed refusal or GAP row, it is not a $\mathsf{ClaimAuditBoundary}^{\uparrow}$ packet.

Independence witness: writing-item-audit-packet, apophaticname, registrylayer, objectknowledgecertificate: none is bijective to $\mathsf{ClaimAuditBoundary}^{\uparrow}$ because the writing-item packet centers item text, the apophatic name centers externality sockets, the registry layer centers admission rows, and the object-knowledge certificate centers licensed object witnesses, while this packet jointly carries expression, status, gap or socket, refusal, permitted claim, transport, replay, provenance, and local naming rows.

The Lean follow-up must supply explicit Nontrivial $\mathsf{ClaimAuditBoundary}^{\uparrow}$ and FieldFaithful $\mathsf{ClaimAuditBoundary}^{\uparrow}$ instances in lean4/BEDC/Derived/ClaimAuditBoundaryUp/TasteGate.lean. The field-faithfulness witness must keep $E, S, G, R, Q, H, C, P, N$ distinct before the claim-audit boundary can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{ClaimAuditBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClaimAuditBoundary}^{\uparrow}$ packet is generated from a finite expression row, claim-status row, gap or socket row, refusal row, permitted-claim row, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{ClaimAuditBoundary}\uparrow}$, seed)

Scope closed: Definition 7318.1, Definition 7318.2, and Definition 7318.3 seed the finite expression, status, gap or socket, refusal, permitted-claim, hsame, Cont, Pkg, and NameCert boundary for legitimate-claim reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClaimAuditBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, registry-wide claim correctness, object-knowledge truth, hidden supply discharge, bridge checking, or theorem status outside the displayed packet rows.

Upgrade path: Obligation closure requires checked carrier admission, status readback, gap or socket exposure, refusal exposure, permitted-claim readback, transport, replay, provenance, and local naming.

Chapter 7319

A Concrete Naming Certificate for $\mathsf{RegularSequenceRegularizer}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ packet records the finite route that turns a stream-window Cauchy witness into a regular rational sequence before the terminal real-name seal is consumed. It sits between the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the uniform-modulus surface of chapter 4158, and the terminal real-name seal of chapter 38. The packet names a displayed regularization interface for Real and Completion consumers; it does not choose a limit, quotient streams, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7319.1 (Regular sequence regularizer carrier)**.** *A* $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, D, K, U, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the Cauchy request;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic endpoint and comparison budgets used by the regularization;*
3. *K is the stream-window Cauchy evidence row, read only through S and D;*
4. *U is the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *row that supplies one displayed modulus schedule for the regularization route;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from S, D, K, U;*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback row has been displayed;*
7. *H records componentwise* hsame *transport for $S, D, K, U, Q, R, C, P, N$;*
8. *C records displayed* Cont *replay through the stream window, dyadic tolerance row, Cauchy evidence, uniform modulus, regular readback, real seal, transport, provenance, and local naming rows;*
9. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceRegularizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceRegularizer}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen limit, quotient stream equality, host real equality, countable-choice schedule, hidden convergence oracle, or ambient completeness principle.

Theorem 7319.2 (Regular sequence regularizer NameCert obligations)**.** *For every accepted* RegularSequenceRegula *carrier* $G = (S, D, K, U, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceRegularizer}^{\uparrow}}$ *surface consists exactly of carrier admission for* G*, stream-window exposure through* S*, dyadic tolerance exposure through* D*, Cauchy evidence exposure through* K*, uniform-modulus exposure through* U*, regular rational readback through* Q*, terminal real-seal handoff through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 7319.1. The only observation data are the finite stream windows in S, and the only tolerance arithmetic available to the regularizer is the dyadic ledger D.

The row K records that the inspected windows satisfy the Cauchy request, while U supplies the displayed uniform schedule that converts the request into a regular route. Thus the regular rational readback Q is downstream of S, D, K, U.

The seal R is reached only after the regular readback is visible. The structural rows H, C, P, N transport, replay, source, and name that same finite route, so the NameCert surface cannot add a selected limit, quotient stream equality, host real equality, countable-choice schedule, hidden convergence oracle, or ambient completeness coordinate. □

Theorem 7319.3 (Regularizer RegSeqRat readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exported by an accepted* $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ *packet factors through the displayed route*

$$S, D, K, U \longmapsto Q.$$

Consequently a Real-facing or Completion-facing consumer cannot read the regular sequence as a free stream, a selected limit, or a quotient representative outside the StreamName, DyadicRatCore, Cauchy evidence, and regular uniform-modulus rows.

Proof. Begin with $G = (S, D, K, U, Q, R, H, C, P, N)$. The carrier definition presents Q as the regular rational row produced from the finite-window row S, dyadic tolerance row D, Cauchy evidence row K, and uniform-modulus row U.

Use Definition 7319.2. Its readback clause exposes Q only after the four source rows have been displayed, and its transport and replay clauses preserve that dependency under hsame and Cont.

Therefore any consumer of Q receives a regular readback route rather than an independent stream or chosen representative. The packet carries no coordinate that could supply a selected limit, quotient stream equality, host real equality, countable-choice schedule, or hidden convergence oracle. □

Theorem 7319.4 (Regularizer terminal seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ *packet is downstream of the rows*

$$S, D, K, U, Q, H, C, P, N.$$

It cannot be consumed as ambient real completeness, host real equality, quotient stream equality, a selected real limit, or a completion seal independent of the displayed regularization route.

Proof. Project the accepted packet. The carrier row R is the only terminal $\mathsf{Real}^{\uparrow}$ seal coordinate, and it appears after the stream window, dyadic tolerance, Cauchy evidence, uniform modulus, and regular readback rows.

Apply Definition 7319.2. Its terminal handoff clause binds R to Q, while the preceding clauses bind Q to S, D, K, U.

The structural rows H, C, P, N keep the same route visible under transport, replay, provenance, and naming. Since no ambient-completeness or quotient coordinate is stored in the carrier, the terminal seal cannot escape the displayed regularization route. □

Falsifiable prediction: A consumer that claims a $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ read while omitting the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, Cauchy evidence row, RegularCauchyUniformMo schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert rows refutes the carrier of Definition 7319.1.

Independence witness: streamname, dyadicratcore, regseqrat, regularcauchyuniformmodulus, real: no listed sibling is bijective to $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ because this packet alone binds finite stream windows, dyadic tolerance data, Cauchy evidence, a regular uniform modulus, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming as one regularization interface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceRegularizer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceRegularizer}^{\uparrow}$ packet is generated from StreamName finite-window rows, DyadicRatCore tolerance rows, stream-window Cauchy evidence, a RegularCauchyUniformModulus schedule, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceRegularizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7319.1, Definition 7319.2, Definition 7319.3, and Definition 7319.4 seed the finite regularization route between stream-window Cauchy evidence and RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceRegularizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected real limits, host real equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance exposure, Cauchy evidence exposure, uniform-modulus exposure, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceRegularizerUp.

Chapter 7320

A Concrete Naming Certificate for $\mathsf{NakayamaLemma}^{\uparrow}$

Following Atiyah–Macdonald and Matsumura, the $\mathsf{NakayamaLemma}^{\uparrow}$ packet names the finite commutative-algebra handoff by which an ideal action trapped in the Jacobson-radical direction forces a finitely generated module witness to collapse. In BEDC form it sits after the $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ carrier of chapter 46, the $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ carrier of chapter 81, the ring and commutative-ring rows of chapter 43 and chapter 44, and the finite-generation surface of $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$ (chapter 3648). The certificate records only the displayed algebraic rows needed for the lemma-shaped non-escape argument; it does not import quotient-module equality, host module equality, full local algebra, or a standard bridge.

7320.1 Carrier

Definition 7320.1 (Nakayama lemma carrier)**.** *A* $\mathsf{NakayamaLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{Nak}(M, R, C, I, F, J, A, H, Q, P, N)$$

with the following displayed rows:

1. M*, the* $\mathsf{Module}^{\uparrow}$ *source surface for the module whose generators are being inspected;*
2. R*, the ambient* $\mathsf{Ring}^{\uparrow}$ *scalar row read by the module action;*
3. C*, the* $\mathsf{CommRing}^{\uparrow}$ *specialization used by the ideal-action rows;*
4. I*, the* $\mathsf{Ideal}^{\uparrow}$ *source surface acting on* M*;*
5. F*, a finite-generator ledger for the displayed module endpoints, read through the* $\mathsf{NoetherianModule}^{\uparrow}$ *finite-generation surface;*
6. J*, a Jacobson-radical containment row for the ideal action;*
7. A*, the action-collapse row recording that the ideal action carries each displayed generator back into the generated submodule boundary;*
8. H*, componentwise* hsame *transport for the module, scalar, ideal, finite-generation, radical-containment, and action-collapse rows;*
9. Q*, displayed* Cont *replay for the finite generator and ideal-action checks;*

10. *P, Pkg provenance for the module, ring, commutative-ring, ideal, finite-generation, radical, and action rows;*
11. *N, the local $\mathsf{NameCert}_{\mathsf{NakayamaLemma}^{\uparrow}}$ row binding these rows as one certificate.*

The packet contains no quotient-module equality, no host equality of module carriers, no local-ring structure theorem, and no chosen maximal ideal.

Theorem 7320.2 (Nakayama NameCert obligations)**.** *For an accepted $\mathsf{NakayamaLemma}^{\uparrow}$ carrier $\mathsf{Nak}(M, R, C, I, F, J, A, H, Q, P, N)$, the local $\mathsf{NameCert}_{\mathsf{NakayamaLemma}^{\uparrow}}$ surface consists exactly of carrier admission for the displayed rows, source compatibility with $\mathsf{Module}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, and $\mathsf{NoetherianModule}^{\uparrow}$, finite-generation access through F, radical-containment access through J, ideal-action collapse through A, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Proof. Project the carrier of Definition 7320.1. The projection exposes exactly the module row, scalar ring row, commutative specialization, ideal row, finite-generation ledger, radical-containment row, action-collapse row, transport, replay, provenance, and local name.

The source compatibility obligations are read componentwise from the first four rows, with F tied to the $\mathsf{NoetherianModule}^{\uparrow}$ finite-generation surface. The radical and action rows are separate finite ledgers, so the certificate does not need a host quotient module or a selected maximal ideal to state the handoff.

Transport and replay preserve only these rows. Thus H, Q, P, and N record movement, finite observation, provenance, and local naming without exporting quotient equality, host module equality, or full local algebra. □

Theorem 7320.3 (Nakayama finite-generation handoff)**.** *Every accepted $\mathsf{NakayamaLemma}^{\uparrow}$ carrier exports finite-generation evidence only through the displayed ledger F over the $\mathsf{Module}^{\uparrow}$ row M and the $\mathsf{NoetherianModule}^{\uparrow}$ finite-generation surface. Downstream algebra consumers may reuse this evidence together with R, C, I, H, Q, P, N, but they cannot read a chosen basis, a quotient-module representative, or a host equality of modules from the packet.*

Proof. Read F from the carrier. It is attached to the module row M and to the finite-generation surface already provided by $\mathsf{NoetherianModule}^{\uparrow}$, so it is the only generator ledger available to a consumer.

The scalar and ideal rows R, C, I give the algebraic context in which the generators are inspected. They do not choose a basis or identify quotient representatives; they only supply the displayed ring, commutative-ring, and ideal surfaces.

Componentwise transport through H, replay through Q, provenance through P, and local naming through N keep the finite-generation row attached to the same packet. Hence the handoff is finite-generation evidence, not host module equality or quotient-module data. □

Theorem 7320.4 (Nakayama nilpotent radical non-escape)**.** *In an accepted $\mathsf{NakayamaLemma}^{\uparrow}$ carrier, radical-action information is confined to the displayed rows J and A: J records the Jacobson-radical containment surface for the ideal row I, and A records the finite action-collapse check over the module row M. Any nilpotent or radical-action read transported by H and replayed by Q remains inside the same $M, R, C, I, F, J, A, H, Q, P, N$ packet and cannot escape into a local-algebra oracle, a maximal-ideal selector, quotient-module equality, or host module equality.*

Proof. Project the radical-facing part of the packet. The row J is the only radical-containment source, and A is the only row that connects the ideal action back to the finite module generator boundary.

The surrounding algebraic rows keep this read finite. The $\mathsf{Module}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Ideal}^{\uparrow}$ sources specify the carrier and action surfaces, while F supplies the finite-generation ledger needed by the collapse argument.

Transport and replay do not add structure. The rows H and Q move and replay the displayed radical and action ledgers componentwise, with P and N recording provenance and local naming. Therefore every radical-action read stays inside the named packet and exports none of the excluded host algebra. □

Closure status for $\mathsf{NameCert}_{\mathsf{NakayamaLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{NakayamaLemma}^{\uparrow}$ packet is generated from ModuleUp, RingUp, CommRingUp, IdealUp, NoetherianModuleUp finite-generation, radical-containment, action-collapse, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NakayamaLemma}^{\uparrow}}$, seed)

Scope closed: Definition 7320.1, Definition 7320.2, Definition 7320.3, and Definition 7320.4 seed the finite Nakayama handoff over ModuleUp, IdealUp, RingUp, CommRingUp, NoetherianModuleUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NakayamaLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient module equality, host module equality, full local algebra, maximal-ideal choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-generation handoff, radical-containment access, action-collapse access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.NakayamaLemmaUp.

Chapter 7321

UniversalCoefficientUp NameCert Construction

Following Eilenberg–Steenrod and Mac Lane, the $\mathsf{UniversalCoefficient}^{\uparrow}$ packet records the finite homological-algebra interface by which a chain-complex homology read and the corresponding cohomology read share coefficient data through displayed module or abelian-group rows. It sits beside the homology carrier of chapter 98, the cohomology substrate of chapter 99, the module source of chapter 46, the abelian-group carrier of chapter 42, and the Mayer–Vietoris sequence route of chapter 6740. The packet names coefficient, chain-complex, exact-sequence, transport, replay, provenance, and local naming rows; it does not assert a host derived functor, selected chain representatives, quotient-class equality, spectral-sequence convergence, or ambient exactness outside the displayed finite window.

Definition 7321.1 (Universal coefficient carrier)**.** *A* $\mathsf{UniversalCoefficient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, A, M, H_*, H^*, E, T, R, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{ChainComplex}^{\uparrow}$ *source row, naming the inspected boundary windows and the displayed square-zero compatibility rows.*
2. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *coefficient row, read as the only coefficient carrier available to both homology and cohomology consumers.*
3. *M is the finite coefficient-action ledger, recording the module action or additive group operation rows consumed by the inspected degree.*
4. *H_* is the* $\mathsf{Homology}^{\uparrow}$ *row for the displayed chain-complex degree window.*
5. *H^* is the* $\mathsf{Cohomology}^{\uparrow}$ *row for the matching cochain degree window.*
6. *E is the* $\mathsf{ExactSequence}^{\uparrow}$ *row that records the displayed boundary order, kernel-image contact, and connecting slot for the inspected universal-coefficient window.*
7. *T records componentwise* hsame *transport for chain-complex, coefficient, homology, cohomology, exact-sequence, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay for consumers that ask for the same coefficient window and degree route.*

9. *P is the* Pkg *provenance over* UniversalCoefficient$^\uparrow$, ChainComplex$^\uparrow$, Homology$^\uparrow$, Cohomology$^\uparrow$, Module$^\uparrow$, AbGroup$^\uparrow$, ExactSequence$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniversalCoefficient}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a global derived functor, a selected cycle or cocycle representative, quotient-class equality, an unrestricted splitting, spectral-sequence data, or ambient exactness beyond E.

Theorem 7321.2 (Universal coefficient NameCert obligations)**.** *For every accepted* UniversalCoefficient$^\uparrow$ *carrier $U = (C, A, M, H_*, H^*, E, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniversalCoefficient}^\uparrow}$ *surface consists exactly of carrier admission for the displayed chain-complex source C, coefficient row A, coefficient-action ledger M, homology row H_*, cohomology row H^*, exact-sequence row E, componentwise transport T, displayed replay R, provenance P, and local naming N. Every consumer must read the coefficient row before either the* Homology$^\uparrow$ *or* Cohomology$^\uparrow$ *face, and must read the displayed* ExactSequence$^\uparrow$ *row before using the homology-to-cohomology handoff.*

Proof. Project the carrier of Definition 7321.1. The tuple lists the only rows available to the classifier: chain-complex source, coefficient carrier, coefficient action, homology, cohomology, exact-sequence, transport, replay, provenance, and local naming.

The coefficient row is shared. A homology-facing read and a cohomology-facing read both pass through A and M, so the packet cannot silently change coefficient systems between H_* and H^*.

The exact-sequence row is then the unique handoff route. The row E records the displayed kernel-image contact and connecting slot for the inspected degree window, while T and R transport and replay that same finite route. Thus no selected representative, unrestricted splitting, quotient equality, or ambient exactness principle can be exported. □

Theorem 7321.3 (Universal coefficient homology-cohomology handoff)**.** *Let $U = (C, A, M, H_*, H^*, E, T, R, P, N)$ be an accepted* UniversalCoefficient$^\uparrow$ *packet. Any downstream consumer that moves from the* Homology$^\uparrow$ *row H_* to the* Cohomology$^\uparrow$ *row H^* must first expose the common coefficient row A, the coefficient-action ledger M, the* ChainComplex$^\uparrow$ *source C, and the* ExactSequence$^\uparrow$ *row E. The handoff supplies a finite displayed universal-coefficient route for the inspected degree window, not a chosen splitting, selected chain representative, quotient-class equality, host derived-functor computation, or spectral-sequence convergence claim.*

Proof. Begin with the shared coefficient surface. The rows A and M are the only coefficient data named by the packet, so both homology and cohomology reads must use the same displayed AbGroup$^\uparrow$ or Module$^\uparrow$ source.

Read the chain-complex and exact-sequence rows next. The row C fixes the inspected boundary windows, and E records the finite exact-sequence contact through which the homology and cohomology rows are related.

The structural rows preserve this route for downstream consumers. Componentwise transport T, replay R, provenance P, and local naming N carry the displayed C, A, M, H_*, H^*, E surface without adding a selected representative, a split sequence, quotient equality, a host derived functor, or spectral-sequence data. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniversalCoefficient}^\uparrow}$.

Constructive story: A UniversalCoefficient$^\uparrow$ packet is generated from a displayed ChainComplexUp source row, a shared AbGroupUp or ModuleUp coefficient row, a coefficient-action ledger, HomologyUp and CohomologyUp degree-window rows, an ExactSequenceUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniversalCoefficient}^\uparrow}$, seed)

Scope closed: Definition 7321.1, Definition 7321.2, and Definition 7321.3 seed the finite carrier and handoff surface for universal-coefficient packets over homology, cohomology, chain-complex, module, abelian-group, and exact-sequence rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{UniversalCoefficient}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a global derived functor, selected chain representatives, quotient-class equality, unrestricted splitting, spectral-sequence convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier, coefficient, homology, cohomology, exact-sequence, transport, replay, provenance, and local naming rows for BEDC.Derived.UniversalCoefficientUp.

Chapter 7322

A Concrete Naming Certificate for $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$

Following Bishop and Bridges on choice sequences and constructive real numbers, the ConstructiveChoiceSequenceSp packet names the finite-window space in which a displayed stream name is read together with a regular Cauchy rational row and a terminal Real seal. It sits between the finite stream discipline of chapter 126, the regular Cauchy rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real boundary of chapter 38. The chapter records only the finite rows needed for a named constructive choice-sequence space; it does not add countable choice, quotient stream equality, selected limits, trichotomy, or ambient Real completeness.

Definition 7322.1 (Constructive choice-sequence space carrier)**.** *A* $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, Q, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing only the finite windows of the sequence name that the packet inspects.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, reading those windows as regular Cauchy rational approximants.*
3. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S and Q have been displayed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested windows and regularity bounds.*
5. *W is the finite window-budget row, recording which stream positions, rational approximants, and dyadic tolerances are visible to the packet.*
6. *H records componentwise* hsame *transport for S, Q, R, D, W and for the replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a finite stream-window request through regular rational readback and the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, R, D, W, H, C, P, N$. It has no coordinate for countable choice, quotient stream equality, selected limits, trichotomy, or ambient Real completeness.

Theorem 7322.2 (Constructive choice-sequence space NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$ *carrier* $X = (S, Q, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveChoiceSequenc}}$ *surface consists exactly of carrier admission for the finite stream windows, regular Cauchy rational readback, terminal Real seal, dyadic tolerance ledger, finite window budget, componentwise transport, displayed replay, provenance, and local naming rows. Every consumer must read the route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *before treating the packet as a constructive choice-sequence space.*

Proof. Project the carrier of Definition 7322.1. The visible analytic rows are exactly the stream-window row S, regular rational readback Q, Real seal R, dyadic tolerance ledger D, and finite window budget W.

The NameCert obligations are therefore local. A request first exposes the finite stream positions in S and the budget row W; it then reads regular rational approximants through Q and the dyadic tolerance ledger D before the Real seal R is available.

Transport and replay do not create new sequence data. Componentwise hsame in H preserves the displayed rows, while Cont in C repeats the same finite-window route for every accepted request.

Finally, provenance and naming are exhausted by P and N. Since countable choice, quotient stream equality, selected limits, trichotomy, and ambient Real completeness are not carrier rows, the certificate cannot export them. □

Theorem 7322.3 (Constructive choice-sequence space finite-window law)**.** *Every read of an accepted* $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$ *packet is determined by a finite window budget W and factors through the sibling rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *named by chapter 126, chapter 301, chapter 291, and chapter 38. Enlarging the budget may expose additional displayed windows, but no read bypasses the same* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *route.*

Proof. Start with an accepted carrier $X = (S, Q, R, D, W, H, C, P, N)$. The row W lists the only stream positions, rational approximants, and dyadic tolerances visible to the packet, so a consumer begins with a finite budget rather than an infinite selector.

Apply the carrier route. The stream row S supplies the displayed finite windows, Q reads them as regular Cauchy rational approximants, and D supplies the dyadic tolerance comparisons needed before R seals the data as $\mathsf{Real}^{\uparrow}$.

If the budget is enlarged, the same construction is repeated with a larger displayed row W'. The transport row H and replay row C preserve the componentwise route, and P, N name no additional source.

Thus the finite-window law is stable under budget enlargement but never becomes countable choice, quotient stream equality, selected limit data, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveChoiceSequenceSpace}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a Real seal, a DyadicRatCore tolerance ledger, a finite window budget, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ConstructiveChoiceSequenceSpace}\uparrow}$, seed)

Scope closed: The chapter seeds the finite-window carrier and NameCert obligation surface for Bishop-style constructive choice-sequence spaces.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConstructiveChoiceSequenceSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim countable choice, quotient stream equality, selected limits, trichotomy, or ambient Real completeness.

Upgrade path: A Lean carrier and checked finite-window gate would move the verification axis beyond the unformalized seed.

Chapter 7323

A Concrete Naming Certificate for $\mathsf{FiniteProductCover}^{\uparrow}$

Following Bishop and Bridges and Kelley on finite products of compact metric covers, the $\mathsf{FiniteProductCover}^{\uparrow}$ packet records the finite product-cover route used by compact metric and uniform-modulus consumers. It sits between the product metric carrier of chapter 424, the compact product route of chapter 1822, the compact metric source surface of chapter 129, the finite refinement route of chapter 3430, and the product constructor surface of chapter 34. The packet names a finite cover of a displayed product metric object; it does not choose subcovers, assert open-cover compactness, import countable choice, or identify quotient product points.

Definition 7323.1 (Finite product cover carrier)**.** *A* $\mathsf{FiniteProductCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, R, J, M, K, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the left finite cover row, a finite list of* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *cells over the left* $\mathsf{CompactMetric}^{\uparrow}$ *source;*
2. *R is the right finite cover row, a finite list of* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *cells over the right* $\mathsf{CompactMetric}^{\uparrow}$ *source;*
3. *J is the product pairing row, recording the* $\mathsf{Prod}^{\uparrow}$ *cell pairs obtained from the displayed left and right cover cells;*
4. *M is the* $\mathsf{ProductMetric}^{\uparrow}$ *ledger that reads distances only through the displayed left and right metric rows;*
5. *K records the* $\mathsf{CompactMetricProduct}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *source rows consumed by the product-cover request;*
6. *B is the finite radius budget row, recording the product-radius allowance and the component radius budgets assigned to L and R;*
7. *H records componentwise* hsame *transport for $L, R, J, M, K, B, C, P, N$;*
8. *C records displayed* Cont *replay through the left cover, right cover, product pairing, product metric ledger, compact metric sources, radius budget, transport, provenance, and local naming rows;*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteProductCover}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteProductCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, a selected subcover, countable choice, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7323.2 (Finite product cover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteProductCover}^{\uparrow}$ *carrier $F = (L, R, J, M, K, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteProductCover}^{\uparrow}}$ *surface consists exactly of left finite cover exposure, right finite cover exposure, product pairing exposure, product metric ledger exposure, compact metric source exposure, finite radius budget exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7323.1. The cover data are exactly the finite left and right rows L, R, and the product-cover cells are exactly the product pairing row J.

The metric and compactness-facing reads are displayed separately. The row M supplies the $\mathsf{ProductMetric}^{\uparrow}$ ledger, K supplies the $\mathsf{CompactMetricProduct}^{\uparrow}$ and $\mathsf{CompactMetric}^{\uparrow}$ source rows, and B records the finite radius budget used by the product-cover request.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier stores no arbitrary open cover, selected subcover, countable-choice schedule, quotient point equality, ambient Real completeness, Lean theorem, or bridge row, the listed rows exhaust the local NameCert obligations. □

Theorem 7323.3 (Finite product cover product-net route)**.** *Every product-net consumer of an accepted* $\mathsf{FiniteProductCover}^{\uparrow}$ *packet factors through the displayed route*

$$L, R, B \longmapsto J \longmapsto M, K.$$

It first reads the left and right finite cover rows and the finite radius budget, then forms only the displayed $\mathsf{Prod}^{\uparrow}$ *pair cells, and only then consumes the* $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, *and* $\mathsf{CompactMetric}^{\uparrow}$ *rows.*

Proof. Begin with $F = (L, R, J, M, K, B, H, C, P, N)$. The left and right finite cover rows are the only available component cover data, and B is the only radius budget attached to them.

Read the product pairing row J. It pairs cells already present in L and R, so it cannot introduce an unseen component cover cell, an arbitrary open cover, or a selected subcover.

Now consume the product metric ledger M and the compact metric source row K. These rows read the paired cells as a product metric cover over the displayed compact metric sources, while H, C, P, N keep the route stable under transport, replay, provenance, and naming.

Thus every product-net route is componentwise and finite. No step supplies countable choice, quotient product point equality, ambient Real completeness, Lean verification, or bridge checking. □

Falsifiable prediction: A consumer that claims a $\mathsf{FiniteProductCover}^{\uparrow}$ read while omitting the left finite cover row, right finite cover row, product pairing row, product metric ledger, compact metric source rows, finite radius budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 7323.1.

Independence witness: productmetric, compactmetricproduct, compactmetric, finitecoverrefinement, prod: no listed sibling is bijective to $\mathsf{FiniteProductCover}^{\uparrow}$ because this packet alone binds

paired finite cover rows, a product pairing row, a product metric ledger, compact metric source rows, finite radius budgeting, transport, replay, provenance, and local naming as one product-cover certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteProductCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteProductCover}^{\uparrow}$ packet is generated from left and right finite cover rows, product pairing rows, a ProductMetric ledger, CompactMetricProduct and CompactMetric source rows, a finite radius budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteProductCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7323.1, Definition 7323.2, and Definition 7323.3 seed the finite product-cover route for compact metric and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteProductCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, component cover exposure, product pairing exactness, product metric ledger exposure, compact metric source exposure, finite radius budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteProductCoverUp.

Chapter 7324

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailLimit}^{\uparrow}$

Following Bishop–Bridges regular Cauchy sequences, the $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ packet records the finite tail-window limit route from inspected regular rational data to a terminal real-name seal. It sits between the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-certificate route of chapter 1230, and the terminal real-name seal of chapter 38. The packet names a finite tail-limit handoff; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, select a quotient representative, or import countable choice.

Definition 7324.1 (Regular Cauchy tail limit carrier)**.** *A* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, Q, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row inspected by the limit request;*
2. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational sequence visible on that window;*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested precision, tail index budget, and comparison allowance;*
4. R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only through the displayed* S, Q, D *route;*
5. A *is the tail-stability row, recording that later inspected windows replay the same regular-Cauchy tail route at the requested tolerance;*
6. H *records componentwise* hsame *transport for* S, Q, D, R, A, C, P, N*;*
7. C *records displayed* Cont *replay through the stream window, regular rational readback, dyadic tolerance row, real seal, tail-stability row, transport, provenance, and local naming rows;*
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden limit choice, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7324.2 (Regular Cauchy tail limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *carrier* $T = (S, Q, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailLimit}^{\uparrow}}$ *surface consists exactly of finite StreamName window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance exposure, Real seal handoff, tail-stability exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7324.1. The finite observation is S, the regular rational readback is Q, and the tolerance budget is D.

The terminal real seal R is downstream of those three rows. The tail-stability row A records that later inspected windows replay the same route at the displayed tolerance; it is not a hidden choice of a limit.

The structural rows H, C, P, N preserve, replay, source, and name this finite route. Because the carrier stores no quotient stream equality, countable-choice schedule, ambient Real completeness, Lean theorem, or bridge row, the listed rows exhaust the local NameCert obligations. □

Theorem 7324.3 (Regular Cauchy tail limit nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *packet factors through*

$$S, Q, D, A, H, C, P, N$$

before it can read the seal row R*. The terminal seal is therefore a finite tail-window limit handoff, not an ambient completeness principle, selected quotient representative, countable-choice output, or detached host real.*

Proof. Start with $T = (S, Q, D, R, A, H, C, P, N)$. Carrier admission makes the finite tail window, regular rational readback, dyadic tolerance row, and tail-stability row visible before a consumer can reach the terminal seal.

Use Definition 7324.2. Its Real seal handoff clause binds R to the displayed S, Q, D, A route, while the transport and replay clauses keep that route componentwise stable.

The packet has no coordinate for quotient stream equality, selected representatives, countable choice, detached host reals, or ambient completeness. Thus the terminal seal cannot escape the finite StreamName, RegSeqRat, DyadicRatCore, tail-stability, transport, replay, provenance, and local naming rows. □

Theorem 7324.4 (Regular Cauchy tail limit window exposure)**.** *Every accepted* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *carrier* $T = (S, Q, D, R, A, H, C, P, N)$ *must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *before any tail-limit read can consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R*.*

Proof. Project the carrier of Definition 7324.1. The rows S and Q are displayed coordinates, so a consumer cannot read the tail-limit packet without naming both the inspected finite stream window and its regular rational readback.

Apply Definition 7324.3. The seal row R is available only after the route through S, Q, D, A, H, C, P, N is visible, hence the window and readback rows precede every terminal tail-limit read. □

Theorem 7324.5 (Regular Cauchy tail limit tolerance exactness)**.** *For every accepted* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *carrier* $T = (S, Q, D, R, A, H, C, P, N)$*, the dyadic tolerance row* D *and tail-stability row* A *are the complete finite control of the tail budget. No hidden choice of a tail index, quotient representative, or ambient completion witness is available outside* D, A, H, C, P, N*.*

Proof. Use the carrier definition to isolate the tolerance coordinates. The dyadic row D records the requested precision, tail index budget, and comparison allowance, while A records replay of later inspected windows at that displayed tolerance.

The structural rows H, C, P, N transport, replay, source, and name this same budget. Since the carrier has no coordinate for hidden limit choice, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the pair D, A is the exact tolerance surface. □

Theorem 7324.6 (Regular Cauchy tail limit Real seal handoff)**.** *In every accepted* $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ *carrier* $T = (S, Q, D, R, A, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *is reachable only through the displayed finite route*

$$S, Q, D, A, H, C, P, N.$$

The handoff exports no detached host real, quotient-stream equality, selected representative, countable-choice schedule, or ambient completeness theorem.

Proof. Start from the nonescape theorem, Definition 7324.3. It forces every Real-facing consumer through the StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, and tail-stability row before R is read.

Now apply the NameCert obligation surface of Definition 7324.2. The transport, replay, provenance, and local naming rows preserve that finite route, so the seal cannot be consumed through any unlisted completion, representative-selection, or quotient-equality channel. □

Falsifiable prediction: A consumer that claims a $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ read while omitting the $\mathsf{StreamName}^{\uparrow}$ finite tail window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{Real}^{\uparrow}$ seal, tail-stability row, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 7324.1.

Independence witness: streamname, regseqrat, dyadicratcore, real, regularcauchytailcertificate: no listed sibling is bijective to $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ because this packet alone binds finite tail windows, regular rational readbacks, dyadic tolerance budgets, terminal Real seals, tail-stability, transport, replay, provenance, and local naming as one tail-limit certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailLimit}^{\uparrow}$ packet is generated from StreamName tail-window rows, RegSeqRat readback rows, DyadicRatCore tolerance rows, a Real seal, tail-stability evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailLimit}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 7324.1, Definition 7324.2, Definition 7324.3, Definition 7324.4, Definition 7324.5, and Definition 7324.6 expose the finite regular-Cauchy tail-window limit obligations for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailLimit}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected representatives, countable choice, hidden limit selection, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance exactness, Real seal handoff, tail-stability exactness, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailLimitUp.

Chapter 7325

A Concrete Naming Certificate for $\mathsf{FanTheoremBoundary}^{\uparrow}$

Following Brouwer and Bishop–Bridges finite-fan analysis, the $\mathsf{FanTheoremBoundary}^{\uparrow}$ packet records the L10 boundary row that lets finite fan and bar evidence be consumed as displayed BEDC data before any compactness theorem is read. It sits beside the finite prefix-bar certificate of chapter 5863, the finite fan-bar modulus packet of chapter 4297, the stream-window carrier of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the boundary object that prevents finite bar evidence from being re-read as open-cover compactness, a chosen branch, countable choice, quotient stream equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7325.1 (Fan theorem boundary carrier)**.** *A* $\mathsf{FanTheoremBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *finite prefix-bar row, exposing the admitted fan depth and the retained branch-prefix window.*
2. *B is the* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *bar-modulus row, exposing the finite lower-bound handoff available to uniform-modulus consumers.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row read by downstream Real and Completion consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the finite prefix and bar windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that keeps rational sequence data inside the displayed finite window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for any real-name comparison reached from the finite bar data.*
7. *H records componentwise* hsame *transport for T, B, S, D, Q, E and the local packet shape.*
8. *C records displayed* Cont *replay from finite fan evidence through the bar-modulus row and into the analytic readback rows.*
9. *P is the* Pkg *provenance over* $\mathsf{FanTheoremBoundary}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{FiniteFanBarModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FanTheoremBoundary}^\uparrow}$ *row.*
The carrier has no coordinate for open-cover compactness, selected branches, countable choice, quotient stream equality, selected limits, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 7325.2 (Fan theorem boundary NameCert obligations)**.** *For every accepted* $\mathsf{FanTheoremBoundary}^\uparrow$ *carrier*

$$F = (T, B, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FanTheoremBoundary}^\uparrow}$ *surface consists exactly of finite prefix-bar admission through T, finite fan-bar modulus admission through B, stream-window exposure through S, dyadic tolerance exposure through D, regular-rational readback through Q, real-name sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Thus a Real or Completion consumer may use the packet only after the finite fan theorem row, finite bar-modulus row, observation window, tolerance row, rational readback row, real seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Project the carrier of Definition 7325.1. The tuple displays exactly the finite fan theorem row, finite fan-bar modulus row, stream window, dyadic tolerance row, regular-rational readback row, real seal, transport, replay, provenance, and local naming row.

The fan and bar evidence is finite because T and B are the only rows through which branch-facing information enters the packet. The row T exposes a retained prefix window, while B exposes the lower-bound handoff available after the finite bar rows have been inspected.

The analytic readback is downstream of those finite rows. The rows S, D, Q, E record the observation window, tolerance ledger, rational readback, and terminal real seal; none of them contains a selected branch or an open-cover subcover.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for open-cover compactness, selected branches, countable choice, quotient stream equality, selected limits, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7325.3 (Fan theorem boundary uniform-bar readback)**.** *Every Real or Completion read of an accepted* $\mathsf{FanTheoremBoundary}^\uparrow$ *packet factors through*

$$\mathsf{FiniteFanTheorem}^\uparrow \longrightarrow \mathsf{FiniteFanBarModulus}^\uparrow \longrightarrow \mathsf{StreamName}^\uparrow \longrightarrow \mathsf{DyadicRatCore}^\uparrow \longrightarrow \mathsf{RegSeqRat}^\uparrow \longrightarrow \mathsf{Real}^\uparrow$$

and then through H, C, P, N. The readback is a boundary read: it may consume the finite bar row and the finite modulus handoff, but it exports no open-cover compactness, selected branch, countable-choice functional, quotient stream equality, selected limit, or ambient $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. Start from the NameCert obligations of Definition 7325.2. They force every accepted consumer to expose T before B, because the bar-modulus handoff is meaningful only after the finite prefix-bar row has been displayed.

After T and B are visible, the read proceeds through the analytic rows S, D, Q, E. The stream window supplies the finite observation schedule, the dyadic row supplies the tolerance ledger, the regular-rational row supplies sequence readback, and the real seal records the terminal comparison.

The structural rows H, C, P, N do not add new analytic principles. They preserve and name the same finite route under componentwise transport, displayed replay, provenance, and local NameCert packaging.

Therefore the read remains a boundary read. It carries finite fan and bar evidence into Real and Completion consumers, while the excluded compactness, selection, quotient, choice, limit, and completeness data remain outside the carrier. □

Falsifiable prediction: Every $\mathsf{FanTheoremBoundary}^{\uparrow}$ read factors through one $\mathsf{FiniteFanTheorem}^{\uparrow}$ prefix-bar row, one $\mathsf{FiniteFanBarModulus}^{\uparrow}$ finite bar-modulus row, one $\mathsf{StreamName}^{\uparrow}$ observation window, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: finite fan theorem, finite fan-bar modulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{FanTheoremBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the boundary between finite fan/bar evidence and Real-or-Completion readback while refusing compactness, selection, quotient, choice, limit, and completeness data.

Closure status for $\mathsf{NameCert}_{\mathsf{FanTheoremBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FanTheoremBoundary}^{\uparrow}$ packet is generated from finite fan theorem rows, finite fan-bar modulus rows, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanTheoremBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7325.1, Definition 7325.2, and Definition 7325.3 seed the L10 boundary route from finite fan and bar evidence into Real and Completion readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanTheoremBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected branches, countable choice, quotient stream equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite fan theorem exposure, finite bar-modulus exposure, stream-window coverage, dyadic tolerance exactness, rational readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FanTheoremBoundaryUp.

Chapter 7326

A Concrete Naming Certificate for $\mathsf{SobolevExtensionOperator}^{\uparrow}$

Following Adams–Fournier and Evans-style Sobolev extension theory, the $\mathsf{SobolevExtensionOperator}^{\uparrow}$ packet records the finite BEDC rows needed by a boundary-to-interior extension read. It sits beside the Sobolev source of chapter 5911, the trace packet of chapter 3647, the boundary package of chapter 6445, the compact-metric support of chapter 129, and the complete-metric readback surface of chapter 128. The packet names only a finite extension route from displayed boundary data back to displayed Sobolev rows; it does not select a global host operator, identify quotient function spaces, or assert an unrestricted trace theorem.

Definition 7326.1 (Sobolev extension operator carrier)**.** *A* $\mathsf{SobolevExtensionOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, T, B, K, M, R, X, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SobolevSpace}^{\uparrow}$ *or* $\mathsf{Sobolev}^{\uparrow}$ *source row, exposing only the interior Sobolev carrier data already named by the corresponding packet.*
2. *T is the* $\mathsf{SobolevTrace}^{\uparrow}$ *row, recording the boundary trace read that the extension route is allowed to address.*
3. *B is the* $\mathsf{SobolevTraceBoundary}^{\uparrow}$ *row. It fixes the boundary carrier, trace-read arrow, compact boundary window, and complete-metric readback needed by the extension request.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row, used only as a finite boundary-window and compact-source ledger.*
5. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *readback row for the interior target and boundary compatibility checks.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for norm budgets, boundary radii, and extension estimates.*
7. *X is the extension-route classifier. It distinguishes source Sobolev reads, boundary trace reads, compact-window reads, complete-metric readbacks, and scalar budget reads, but has no coordinate for a selected global operator or quotient equality of functions.*
8. *H stores componentwise* hsame *transport for S, T, B, K, M, R, X and the structural rows.*
9. *C stores displayed* Cont *replay from an extension request through the Sobolev source, trace row, boundary row, compact support, complete-metric readback, and scalar seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SobolevExtensionOperator}^{\uparrow}$, $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, $\mathsf{SobolevTraceBoundary}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SobolevExtensionOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, B, K, M, R, X, H, C, P, N$. It does not contain an unrestricted host trace theorem, a selected extension operator on all boundary data, quotient function-space equality, open-cover compactness, countable choice, or ambient completeness.

Theorem 7326.2 (Sobolev extension operator NameCert obligations)**.** *For every accepted* SobolevExtensionOperato *carrier*

$$E = (S, T, B, K, M, R, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SobolevExtensionOperator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Sobolev source, trace row, boundary row, compact-metric support, complete-metric readback, scalar seal, classifier, transport, replay, provenance, and local naming rows;*
2. *trace-source compatibility, requiring every extension request to read T and B before it can claim a boundary value belongs to the extension route;*
3. *compact and complete-metric exactness, requiring K and M to expose only finite boundary-window, compact-source, distance, and Cauchy-readback rows;*
4. *scalar budget exactness, requiring norm and radius comparisons to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row R;*
5. *structural stability under X, H, C, P, N, which classifies, transports, replays, sources, and names only the displayed finite route.*

No $\mathsf{SobolevExtensionOperator}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7326.1. The analytic rows visible to the classifier are exactly S, T, B, K, M, R, and the structural rows are X, H, C, P, N.

The trace-source compatibility obligation is forced before any extension read. A consumer must first expose the Sobolev trace row T and the boundary package B, so the boundary datum is read through the finite trace and boundary ledgers already present in the carrier.

The compact and complete-metric obligations are then attached to K and M. These rows provide only finite boundary-window support and displayed Cauchy or distance readbacks, while scalar budget comparisons reach the Real seal only through R.

Finally X, H, C, P, N classify, transport, replay, source, and name the same row order. Since the carrier has no coordinate for a host operator, quotient equality, open-cover compactness, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7326.3 (Sobolev trace-extension handoff)**.** *Every downstream trace or boundary-analysis consumer citing an accepted* $\mathsf{SobolevExtensionOperator}^{\uparrow}$ *carrier receives only the finite handoff*

$$\mathsf{SobolevSpace}^{\uparrow}, \mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{SobolevTrace}^{\uparrow} \longmapsto \mathsf{SobolevTraceBoundary}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{Real}$$

followed by the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the same packet. The handoff is a finite extension-route certificate; it is not a host extension theorem, a selected linear operator, or an equality of completed Sobolev quotients.*

Proof. Start with an accepted trace-extension request. The request first reads the Sobolev source row S, so it inherits only the interior Sobolev data displayed by the source packet.

Next it reads T and B. These rows fix the boundary trace and boundary carrier before compact support or complete-metric readback can be consulted.

The remaining analytic read passes through K, M, R. Thus compact-window data, Cauchy or distance readback, and scalar estimates are finite rows of the same packet.

The structural rows X, H, C, P, N only classify, transport, replay, source, and name this route. No carrier coordinate remains from which a host extension theorem, selected operator, or quotient-space equality could be read. □

Theorem 7326.4 (Sobolev extension operator nonescape boundary)**.** *Every accepted* $\mathsf{SobolevExtensionOperator}^{\uparrow}$ *consumer that reaches the complete-metric or scalar extension estimate must first display the boundary route T, B and the compact support row K. Consequently the consumer cannot replace the boundary route by an unrestricted host trace theorem, a detached boundary representative, open-cover compactness, ambient completeness, or a selected global extension operator.*

Proof. Project an accepted consumer of the carrier. The complete-metric row M and scalar seal R occur downstream of S, T, B, K, so neither readback can be reached before the boundary trace and compact support rows are fixed.

The classifier row X records this row order as part of the accepted packet. Componentwise transport H and replay C preserve that order, while P and N source and name the same route.

Any consumer omitting T, B, or K would be reading a coordinate absent from the displayed carrier. Hence complete-metric and scalar extension estimates cannot escape the finite boundary route. □

Falsifiable prediction: Every accepted $\mathsf{SobolevExtensionOperator}^{\uparrow}$ extension read factors through displayed SobolevSpaceUp or SobolevUp, SobolevTraceUp, SobolevTraceBoundaryUp, CompactMetricUp, CompleteMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any downstream consumer receives an extension estimate.

Independence witness: sobolev-space, sobolev, sobolev-trace, sobolev-trace-boundary, compactmetric, completemetric, real: $\mathsf{SobolevExtensionOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source Sobolev data, trace data, boundary data, compact support, complete-metric readback, scalar estimates, classifier, transport, replay, provenance, and local naming in one finite extension-route carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevExtensionOperator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevExtensionOperator}^{\uparrow}$ packet is generated from a Sobolev source row, Sobolev trace row, Sobolev trace boundary row, compact-metric support row, complete-metric readback row, Real scalar seal, extension-route classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobolevExtensionOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7326.1, Definition 7326.2, Definition 7326.3, and Definition 7326.4 seed the finite Sobolev extension-route surface over Sobolev, trace, boundary, compact-metric, complete-metric, and real neighbours.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobolevExtensionOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an unrestricted host trace theorem, quotient function-space equality, a selected global extension operator, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trace-source compatibility, compact and complete-metric exactness, scalar budget exactness, classifier nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SobolevExtensionOperatorUp.

Chapter 7327

A Concrete Naming Certificate for $\mathsf{LocatedUniformExtension}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedUniformExtension}^{\uparrow}$ packet records the finite surface by which a uniformly continuous source map on a located dense domain is read as an extension candidate through explicit modulus, Cauchy handoff, window, readback, and real-seal rows. It refines the uniformly continuous extension route of chapter 769 and the Cauchy-controlled extension route of chapter 1632 by requiring the source domain to be displayed as located data before any completion-facing extension read is exposed.

The packet is a non-quotient L10 extension certificate: it keeps the modulus and located-domain rows visible, routes Cauchy evidence through finite StreamName windows and RegSeqRat readback, and reaches $\mathsf{Real}^{\uparrow}$ only through a terminal seal. It does not identify host functions, select representatives, import ambient completeness, or replace the displayed domain by an impredicative subset.

Definition 7327.1 (Located uniform extension carrier)**.** *A* $\mathsf{LocatedUniformExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, M, E, K, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the located domain row, naming the source subcarrier, its located comparison data, and the finite membership boundary read by the extension.*
2. *M is the modulus row, recording the uniform continuity tolerance-to-radius data consumed from the source map before completion-facing reads are available.*
3. *E is the extension candidate row, aligned with the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff of chapter 769.*
4. *K is the Cauchy handoff row, aligned with the* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *route of chapter 1632.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which source and target observations are requested at finite precision.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the regular rational data used by the Cauchy and extension rows.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the located domain, modulus, extension candidate, Cauchy handoff, window, and readback rows have been replayed.*

8. *H records componentwise* hsame *transport for $D, M, E, K, W, Q, R, C, P, N$.*
9. *C records displayed* Cont *replay from located-domain reads to modulus reads, from modulus reads to extension-candidate reads, and from Cauchy handoff through windows and readback to the Real seal.*
10. *P is the* Pkg *provenance over* LocatedUniformExtension$^{\uparrow}$, UniformContinuousExtension$^{\uparrow}$, CauchyContinuousExtensi StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocatedUniformExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, E, K, W, Q, R, H, C, P, N$. There is no coordinate for quotient function equality, selected completion representatives, host subset extensionality, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7327.2 (Located uniform extension NameCert obligations)**.** *For every accepted* LocatedUniformExtension *carrier*

$$L = (D, M, E, K, W, Q, R, H, C, P, N),$$

the local NameCert$_{\text{LocatedUniformExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located domain, modulus, extension candidate, Cauchy handoff, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *located-domain visibility, requiring every extension read to project D before using the modulus row;*
3. *modulus exactness, requiring M to be the only tolerance-to-radius source for the extension candidate E;*
4. *Cauchy handoff ordering, requiring K to consume the same located-domain and modulus data before window or readback rows are public;*
5. *readback and seal exactness, requiring W, Q, R to be consumed in the displayed finite order before any Real-facing value is exported;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet.*

No obligation row stores a host function, quotient completion point, selected representative, ambient completeness principle, or choice functional.

Proof. Project the carrier of Definition 7327.1. Carrier admission is the display of the rows $D, M, E, K, W, Q, R, H, C, P, N$ as one finite BHist packet.

The located-domain and modulus clauses are read from D and M. The row D is the only source-domain coordinate, and M is the only modulus coordinate, so the extension candidate E cannot be read before those two rows have been consumed.

The Cauchy handoff clause is read from K. It shares the same located-domain and modulus data as E, matching the sibling extension surfaces cited above while retaining finite window and readback access through W and Q.

Finally inspect R, H, C, P, N. The Real seal is terminal for the displayed window-readback route, and the structural rows only transport, replay, source, and name that route. Since the excluded objects have no carrier coordinate, the local NameCert obligations are exactly the listed finite rows. □

Theorem 7327.3 (Located uniform extension Cauchy handoff)**.** *Every Cauchy-facing read of an accepted* $\mathsf{LocatedUniformExtension}^{\uparrow}$ *packet factors through the finite route*

$$D \longmapsto M \longmapsto E, K \longmapsto W, Q \longmapsto R, H, C, P, N.$$

Thus the Cauchy handoff row cannot bypass the located domain, cannot replace the uniform modulus by a hidden host continuity assertion, and cannot reach the $\mathsf{Real}^{\uparrow}$ *seal without the StreamName and RegSeqRat rows.*

Proof. Begin with the NameCert obligations of Definition 7327.2. They force each extension read to project the located domain row D and the modulus row M before the extension candidate or Cauchy handoff is public.

The sibling uniform-extension route supplies the role of E, while the Cauchy-continuous extension route supplies the role of K. Both rows are attached to the same located source and modulus data, so neither can serve as an independent completion source.

The readback rows W, Q mediate all finite observation requests before the terminal seal R is available. The structural rows H, C, P, N preserve this order by componentwise transport, displayed replay, provenance, and local naming.

Therefore every Cauchy-facing read remains on the displayed finite route, with no quotient function equality, selected representative, hidden host continuity assertion, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformExtension}^{\uparrow}$ packet is generated from a located domain row, a uniform-modulus row, an extension candidate, a Cauchy handoff, finite StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7327.1, Definition 7327.2, and Definition 7327.3 give the seed carrier and Cauchy handoff surface for located uniform extension packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient extension equality, selected completion representatives, full extension uniqueness, Lean verification, axiom-purity certification, or bridge checking for $\mathsf{LocatedUniformExtension}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, located-domain visibility, modulus exactness, Cauchy handoff ordering, StreamName-RegSeqRat-Real readback, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformExtensionUp.

Chapter 7328

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$

Following Bishop–Bridges regular Cauchy real construction, the $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ packet records the finite BEDC rows needed to replace a displayed Cauchy modulus by a normalized dyadic schedule. It sits beside the stream naming surface of chapter 126, the regular rational sequence packet of chapter 301, the dyadic rational core of chapter 291, the real seal of chapter 38, and the diagonal selector of chapter 4080. The packet names only the finite normalization route from a stream-indexed modulus to a regular dyadic handoff; it does not assert located order, completion uniqueness, countable choice, or a global real-equality theorem.

Definition 7328.1 (Regular Cauchy modulus-normalization carrier)**.** *A* $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, Q, D, R, \Delta, A, X, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row. It supplies the stream-indexed name whose successive entries are allowed to be queried by the normalization route.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row. It stores the regular rational sequence read that receives the normalized modulus schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row. It supplies dyadic radii, dyadic comparison rows, and denominator-depth data used by the normalized schedule.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal. It records only the scalar real-read interface reached after the regular rational sequence row has accepted the normalized schedule.*
5. *Δ is the* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *row. It records the finite diagonal read that chooses a common index for the stream name, dyadic depth, and regular rational row.*
6. *A is the admission ledger. It records the original Cauchy modulus row, the normalized dyadic modulus row, and the finite proof that the normalized row is at least as demanding as the original row at every displayed depth.*
7. *X is the normalization classifier. It distinguishes stream-name reads, regular rational reads, dyadic-depth reads, real-seal reads, diagonal-selection reads, and admission-ledger reads, but has no coordinate for global choice or completed real equality.*

8. *H stores componentwise* hsame *transport for S, Q, D, R, Δ, A, X and the structural rows.*
9. *C stores displayed* Cont *replay from a modulus-normalization request through the stream name, dyadic-depth row, diagonal selector, regular rational target, and real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, R, \Delta, A, X, H, C, P, N$. It does not contain a located-order decision, a completed-real quotient equality, a countable-choice principle, or an unrestricted completeness theorem.

Theorem 7328.2 (Regular Cauchy modulus-normalization NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ *carrier*

$$M = (S, Q, D, R, \Delta, A, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-name, regular rational sequence, dyadic rational core, real seal, diagonal-selection, admission-ledger, classifier, transport, replay, provenance, and local naming rows;*
2. *modulus dominance, requiring the normalized dyadic row in A to dominate the original Cauchy modulus row at each displayed depth before Q may read the schedule;*
3. *diagonal compatibility, requiring Δ to expose a common finite index for the stream row S, dyadic-depth row D, and regular rational target Q;*
4. *scalar exactness, requiring every real-read budget to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *seal R after the normalized schedule has been accepted by Q;*
5. *structural stability under X, H, C, P, N, which classifies, transports, replays, sources, and names only the displayed finite route.*

No $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7328.1. The analytic rows visible to the classifier are exactly S, Q, D, R, Δ, A, and the structural rows are X, H, C, P, N.

The modulus-dominance obligation is attached to A. A regular rational consumer may read Q only after the normalized dyadic row has been shown to dominate the displayed Cauchy modulus row at the relevant depth.

The diagonal obligation then passes through Δ. This row exposes the common finite index needed by the stream name S, dyadic-depth row D, and regular rational target Q.

Finally scalar reads reach R only after the normalized schedule has passed through Q. Since X, H, C, P, N classify, transport, replay, source, and name the same row order, and since the carrier has no coordinate for located order, quotient equality, countable choice, or unrestricted completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7328.3 (Regular Cauchy modulus-normalization handoff)**.** *Every downstream regular-sequence or real-construction consumer citing an accepted* $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ *carrier receives only the finite handoff*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{CauchyDiagonalSelection}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

followed by the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the same packet. The handoff is a finite modulus-normalization certificate; it is not located order, completed-real equality, countable choice, or a standalone completeness theorem.*

Proof. Start with an accepted normalization request. The request first reads the stream-name row S and dyadic-depth row D, so it inherits only the displayed stream entries and dyadic budget data.

Next it reads the diagonal selector Δ and admission ledger A. These rows fix a common finite index and require dominance of the original Cauchy modulus by the normalized dyadic schedule before the regular rational target can be consulted.

The remaining analytic read passes through Q and then R. Thus the regular rational sequence and real seal receive only the normalized finite schedule named by the same packet.

The structural rows X, H, C, P, N only classify, transport, replay, source, and name this route. No carrier coordinate remains from which located order, completed-real equality, countable choice, or unrestricted completeness could be read. □

Theorem 7328.4 (Regular Cauchy modulus-normalization nonescape boundary)**.** *Every accepted* $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ *consumer that reaches the regular rational target or real seal must first display the stream-name row S, dyadic-depth row D, diagonal selector Δ, and admission ledger A. Consequently the consumer cannot replace the finite normalization route by a detached modulus choice, a located-order decision, a completed-real quotient equality, countable choice, or an unrestricted completeness theorem.*

Proof. Project an accepted consumer of the carrier. The regular rational row Q and real seal R occur downstream of S, D, Δ, A, so neither read can be reached before the stream, dyadic-depth, diagonal, and admission rows are fixed.

The classifier row X records this row order as part of the accepted packet. Componentwise transport H and replay C preserve that order, while P and N source and name the same route.

Any consumer omitting S, D, Δ, or A would be reading a coordinate absent from the displayed carrier. Hence the regular rational and real reads cannot escape the finite modulus-normalization route. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ normalization read factors through displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows before any downstream consumer receives a regular rational or real read.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-diagonal-selection: $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries stream-indexed modulus data, dyadic-depth data, diagonal index selection, regular rational target data, real-seal readback, classifier, transport, replay, provenance, and local naming in one finite normalization-route carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusNormalization}^{\uparrow}$ packet is generated from a StreamName row, RegSeqRat row, DyadicRatCore row, Real seal, CauchyDiagonalSelection row, admission ledger, normalization classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalization}^{\uparrow}}$, seed)

Scope closed: Definition 7328.1, Definition 7328.2, Definition 7328.3, and Definition 7328.4 seed the finite modulus-normalization surface over StreamName, RegSeqRat, DyadicRatCore, Real, and CauchyDiagonalSelection neighbours.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close located order, completed-real quotient equality, countable choice, unrestricted completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus dominance, diagonal compatibility, scalar exactness, classifier nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyModulusNormalization

Chapter 7329

A Concrete Naming Certificate for $\mathsf{PolynomialResultant}^{\uparrow}$

Following Sylvester and van der Waerden, the $\mathsf{PolynomialResultant}^{\uparrow}$ packet records the finite elimination row by which two displayed univariate polynomial coefficient lists expose a determinant obstruction to a shared root. It sits after the coefficient carrier of chapter 50, the finite matrix carrier of chapter 49, and the scalar arithmetic surface of chapter 45. The packet names only the Sylvester matrix assembly, determinant readback, degree ledger, transport, replay, provenance, and local naming rows; it is not an algebraic closure theorem, a selected common root, a quotient-polynomial equality, a Euclidean division oracle, or a host elimination procedure.

Definition 7329.1 (Polynomial resultant carrier)**.** *A* $\mathsf{PolynomialResultant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, G, D, S, M, \Delta, H, C, P, N)$$

with the following displayed rows:

1. *F and G are* $\mathsf{Poly}^{\uparrow}$ *coefficient-list rows for the two input polynomials.*
2. *D is the finite degree and leading-coefficient ledger that fixes the Sylvester block sizes.*
3. *S is the Sylvester assembly row, listing the shifted coefficient rows used to build the matrix.*
4. *M is the* $\mathsf{Mat}^{\uparrow}$ *row that exposes the assembled Sylvester matrix as finite matrix data.*
5. *Δ is the* $\mathsf{Field}^{\uparrow}$ *scalar determinant readback row for the matrix M.*
6. *H records componentwise* hsame *transport for F, G, D, S, M, Δ and the local packet shape.*
7. *C records displayed* Cont *replay from polynomial coefficient reads through Sylvester assembly and determinant readback.*
8. *P is the* Pkg *provenance over* $\mathsf{PolynomialResultant}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PolynomialResultant}^{\uparrow}}$ *row.*

The carrier has no coordinate for algebraic closure, selected roots, quotient-polynomial equality, Euclidean division, subresultant chains, ambient field completion, or host elimination.

Theorem 7329.2 (Polynomial resultant Sylvester determinacy)**.** *For every accepted* $\mathsf{PolynomialResultant}^{\uparrow}$ *carrier*

$$R = (F, G, D, S, M, \Delta, H, C, P, N),$$

the determinant row Δ is determined only after the two polynomial rows F, G, the degree ledger D, the Sylvester assembly row S, and the finite matrix row M have all been displayed. A consumer may therefore read the resultant scalar only along the route

$$\mathsf{Poly}^{\uparrow}, \mathsf{Poly}^{\uparrow} \longmapsto D \longmapsto S \longmapsto \mathsf{Mat}^{\uparrow} \longmapsto \mathsf{Field}^{\uparrow},$$

followed by H, C, P, N; no accepted packet reads Δ from a selected common root, a quotient-polynomial equality, a Euclidean algorithm transcript, or an ambient elimination theorem.

Proof. Project the carrier of Definition 7329.1. The scalar row Δ appears only after the two coefficient rows, the degree ledger, the Sylvester assembly row, and the matrix row have been named.

The Sylvester row S is finite because it is built from shifted copies of the displayed coefficient rows with block sizes fixed by D. The matrix row M is therefore a displayed $\mathsf{Mat}^{\uparrow}$ read, not a host array or an implicit linear operator.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no selected root, quotient-polynomial equality, Euclidean transcript, or host elimination theorem, the determinant row is determined exactly by the displayed Sylvester route. □

Theorem 7329.3 (Polynomial resultant NameCert obligations)**.** *For every accepted* $\mathsf{PolynomialResultant}^{\uparrow}$ *carrier*

$$R = (F, G, D, S, M, \Delta, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolynomialResultant}^{\uparrow}}$ *surface consists exactly of polynomial coefficient exposure through F, G, degree and leading-coefficient exposure through D, Sylvester block assembly through S, matrix readback through M, field determinant readback through Δ, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Thus every downstream elimination, discriminant, or algebraic-geometry consumer receives a finite resultant packet only through the sibling surfaces of chapter 50, chapter 49, and chapter 45.

Proof. Start from the carrier projection. The polynomial source rows F, G and the ledger D provide the only coefficient and degree data available to the packet.

Next apply Definition 7329.2. It forces every determinant read to pass through S and M before the scalar row Δ is exposed.

The structural rows do not add algebraic content. They only transport, replay, source, and name the same finite packet, so the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{PolynomialResultant}^{\uparrow}$ read displays two PolynomialUp coefficient rows, a finite degree ledger, a Sylvester assembly row, a MatrixUp row, a FieldUp determinant readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before any resultant scalar is consumed.

Independence witness: polynomial, matrix, field: $\mathsf{PolynomialResultant}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two polynomial coefficient rows, the Sylvester block assembly, the finite matrix readback, the determinant scalar, and the local NameCert ledger in one elimination packet.

Lean4 (statement) `BEDC.Derived.PolynomialResultantUp`

Closure status for $\mathsf{NameCert}_{\mathsf{PolynomialResultant}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PolynomialResultant}^{\uparrow}$ packet is generated from two PolynomialUp coefficient rows, a degree and leading-coefficient ledger, a Sylvester assembly row, a MatrixUp readback, a FieldUp determinant row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PolynomialResultant}\uparrow}$, seed)

Scope closed: Definition 7329.1, Definition 7329.2, and Definition 7329.3 seed the finite Sylvester-resultant route over polynomial, matrix, and field sibling surfaces.

Formal verification: FormalStatus(NameCert$_{\text{PolynomialResultant}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.PolynomialResultantUp`

Bridge status: none

Not claimed: This chapter does not close algebraic closure, selected roots, quotient-polynomial equality, Euclidean division, subresultant chains, ambient field completion, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sylvester assembly determinacy, matrix readback exactness, determinant row exposure, transport, replay, provenance, and local naming for BEDC.Derived.PolynomialResultantUp.

Chapter 7330

A Concrete Naming Certificate for CompactUniformContinuityFiniteModulus$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor theorem, the CompactUniformContinuityFinite packet names the finite modulus route that turns compact finite-net data and continuous pointwise-modulus data into a single tolerance-only uniform-modulus row. It sits after the compact carrier of chapter 60, the continuous-map surface of chapter 59, the metric comparison rows of chapter 57, the rational and dyadic tolerance ledgers of chapter 37 and chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite fold described in chapter 9237. The packet records a finite ProbeBundle rational-fold handoff only; it does not assert open-cover compactness, choose a pointwise modulus from a host theorem, import Real$^{\uparrow}$ completeness, or identify quotient functions.

Definition 7330.1 (Compact uniform continuity finite-modulus carrier)**.** *A* CompactUniformContinuityFiniteModul *carrier is a finite* BHist *packet*

$$U = (K, F, M, E, B, R, D, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row read through* Compact$^{\uparrow}$*;*
2. *F is the continuous pointwise-modulus row read through* Continuous$^{\uparrow}$*;*
3. *M is the* Met$^{\uparrow}$ *distance-comparison row used by the finite-net and pointwise-modulus reads;*
4. *E is the positive* Rat$^{\uparrow}$ *tolerance request;*
5. *B is the finite* ProbeBundle *of centers, radii, and subordinate pointwise-modulus witnesses;*
6. *R is the* Rat$^{\uparrow}$ *lower-bound fold over the displayed bundle radii;*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row that converts the rational lower bound into the requested dyadic budget;*
8. *Q is the* RegSeqRat$^{\uparrow}$ *readback row used when the modulus is consumed by regular rational names;*
9. *S is the* Real$^{\uparrow}$ *seal reached only after the finite compact, continuous, metric, rational-fold, dyadic, and regular-readback rows are visible;*
10. *H records componentwise* hsame *transport for $K, F, M, E, B, R, D, Q, S, C, P, N$;*

11. *C records displayed* Cont *replay through the compact source, continuous pointwise-modulus source, metric comparison, rational fold, dyadic tolerance row, regular readback, real seal, transport, provenance, and local naming rows;*
12. *P is the* Pkg *provenance over* $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open-cover compactness theorem, a selected global modulus, ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of functions, Lean verification, or bridge checking.*

Theorem 7330.2 (Compact uniform continuity finite-modulus obligations)**.** *For every accepted* $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$ *carrier* $U = (K, F, M, E, B, R, D, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure, continuous pointwise-modulus exposure, metric comparison, positive rational tolerance exposure, finite* ProbeBundle *exposure, rational lower-bound fold exposure, dyadic tolerance handoff, regular rational readback, real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7330.1. The compact source row K, continuous row F, metric row M, tolerance row E, and finite bundle row B are the only data from which the lower-bound fold can be read.

The rational fold row R is computed over the displayed bundle; the dyadic row D and regular-readback row Q only transport that finite lower bound to the consumer side. The real seal S is therefore downstream of the same finite compact-continuous route.

The structural rows H, C, P, N preserve, replay, source, and name the packet. Since the carrier stores no open-cover compactness theorem, selected global modulus, ambient completeness claim, quotient function equality, Lean theorem, or bridge row, the listed rows exhaust the NameCert obligations. □

Theorem 7330.3 (Compact uniform continuity finite-modulus nonescape)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$ *packet must first read*

$$K, F, M, E, B, R, D, Q, H, C, P, N$$

before it can consume the real-seal row S. The finite modulus is therefore the ProbeBundle rational-fold route of the compact-uniform-continuity construction, not an open-cover compactness principle, selected host modulus, quotient-function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with $U = (K, F, M, E, B, R, D, Q, S, H, C, P, N)$. Carrier admission makes the compact finite net, continuous pointwise-modulus row, metric comparison, positive rational tolerance request, finite ProbeBundle, rational lower-bound fold, dyadic tolerance handoff, and regular rational readback visible before the seal is available.

Use Definition 7330.2. Its handoff clauses bind the consumer route to the displayed K, F, M, E, B, R, D, Q rows, while the transport and replay clauses keep the route componentwise stable.

The packet has no coordinate for open-cover compactness, a selected host modulus, quotient equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$. Thus no downstream consumer can detach the seal from the finite ProbeBundle rational-fold route. □

Falsifiable prediction: A consumer that claims a $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$ read while omitting the $\mathsf{Compact}^{\uparrow}$ finite-net row, $\mathsf{Continuous}^{\uparrow}$ pointwise-modulus row, $\mathsf{Met}^{\uparrow}$ comparison

row, positive $\mathsf{Rat}^{\uparrow}$ tolerance row, finite ProbeBundle, rational lower-bound fold, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 7330.1.

Independence witness: compact, continuous, metric, rat, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$ because this packet alone binds compact finite-net data, continuous pointwise-modulus data, metric comparison, rational ProbeBundle folding, dyadic tolerance routing, regular readback, real sealing, transport, replay, provenance, and local naming as one finite modulus certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}$ packet is generated from compact finite-net rows, continuous pointwise-modulus rows, metric comparison rows, positive rational tolerance rows, finite ProbeBundle rows, rational lower-bound fold rows, dyadic tolerance rows, regular rational readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7330.1, Definition 7330.2, and Definition 7330.3 seed the finite ProbeBundle rational-fold modulus route for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityFiniteModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, a selected global modulus, quotient equality of functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous pointwise-modulus exposure, metric comparison, positive rational tolerance exposure, finite ProbeBundle exposure, rational lower-bound fold exactness, dyadic tolerance handoff, regular rational readback, real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformContinuityFiniteModulusUp.

Chapter 7331

A Concrete Naming Certificate for $\mathsf{DyadicMidpointInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicMidpointInterval}^{\uparrow}$ packet records the finite dyadic interval-halving data used when a Real and Completion construction inspects a retained interval before passing to regular rational names. It sits over the dyadic arithmetic core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested interval route of chapter 320. The packet names only midpoint-halving rows in finite BHist data; it does not import Dedekind cuts, host interval equality, open-cover compactness, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7331.1 (Dyadic midpoint interval carrier)**.** *A* $\mathsf{DyadicMidpointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, M, W, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the left endpoint row, read as a dyadic rational endpoint supplied by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
2. *R is the right endpoint row, read as a dyadic rational endpoint in the same ledger and displayed with $L \leq R$ by finite comparison data.*
3. *M is the midpoint row, recording the dyadic value obtained from the displayed halving operation on L and R.*
4. *W is the width and tolerance row. It records the original width, the halved width, and the tolerance budget consumed by Real and Completion requests.*
5. *D is the dyadic ledger. It contains the addition, comparison, halving, and denominator-clearance rows needed to read L, R, M, W without leaving* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for endpoint, midpoint, width, dyadic-ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an interval-halving request through L, R, M, W, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicMidpointInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicMidpointInterval}^{\uparrow}}$ *row naming exactly this midpoint-halving certificate. The classifier compares accepted packets only through $L, R, M, W, D, H, C, P, N$. It has no coordinate for Dedekind cuts, host interval equality, open-cover compactness, selected limits, quotient equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7331.2 (Dyadic midpoint interval halving)**.** *For every accepted* $\mathsf{DyadicMidpointInterval}^{\uparrow}$ *carrier $I = (L, R, M, W, D, H, C, P, N)$, the midpoint row M and width row W display a finite halving route: the original endpoint interval $[L, R]$ is read through the dyadic ledger D, the midpoint is the dyadic half-sum recorded in M, and each retained child interval has width bounded by the halved width recorded in W. Any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of this halving route must pass through D, H, C, P, N before reading the resulting finite window or real-name seal.*

Proof. Project the carrier of Definition 7331.1. The only endpoint, midpoint, and width data available to a consumer are the displayed rows L, R, M, W together with the dyadic ledger D.

The halving step is read inside $\mathsf{DyadicRatCore}^{\uparrow}$. The ledger D supplies the endpoint comparison, half-sum row, denominator-clearance row, and width comparison row, so the retained left or right child interval is certified by finite dyadic arithmetic rather than by host interval equality.

Next attach the Real and Completion handoff. The finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38 may consume the halved interval only after the rows L, R, M, W, D are displayed.

Finally compare with the nested route of chapter 320. A retained nested cell reads this midpoint packet as one finite refinement step, and the structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object is a carrier coordinate, no Dedekind cut, host interval equality, open-cover compactness, selected limit, quotient equality, countable choice, or ambient completeness principle is exported. □

Theorem 7331.3 (Dyadic midpoint interval NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMidpointInterval}^{\uparrow}$ *carrier $I = (L, R, M, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpointInterval}^{\uparrow}}$ *surface consists exactly of endpoint admission, midpoint-halving admission, width and tolerance admission, dyadic-ledger exactness, componentwise transport, displayed replay, provenance, and local naming. The obligation surface may cite the dyadic arithmetic core of chapter 291, the stream window surface of chapter 126, the regular rational handoff of chapter 301, the terminal real-name surface of chapter 38, and the nested interval route of chapter 320; it cannot add Dedekind cuts, host interval equality, open-cover compactness, selected limits, quotient equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Carrier admission is Definition 7331.1. It exposes exactly $L, R, M, W, D, H, C, P, N$ and no host interval object.

The halving obligation is Definition 7331.2. It forces every midpoint and child-width read through the dyadic ledger before any downstream finite window, regular rational row, or real seal can be consumed.

The remaining obligations are structural. Componentwise hsame transport is the row H, displayed replay is the row C, provenance is the row P, and local naming is the row N. These rows preserve the same finite packet, so the local certificate has no route to Dedekind cuts, host interval equality, open-cover compactness, selected limits, quotient equality, countable choice, ambient completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMidpointInterval}^{\uparrow}}$.

Constructive story: A DyadicMidpointInterval$^{\uparrow}$ packet is generated from left and right dyadic endpoint rows, a midpoint row, a width and tolerance row, a dyadic arithmetic ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicMidpointInterval}\uparrow}$, seed)

Scope closed: Definition 7331.1, Definition 7331.2, and Definition 7331.3 seed the finite midpoint-halving certificate used by Real and Completion construction routes.

Formal verification: FormalStatus(NameCert$_{\text{DyadicMidpointInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, host interval equality, open-cover compactness, selected limits, quotient equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, midpoint-halving exactness, width and tolerance readback, dyadic-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMidpointIntervalUp.

Chapter 7332

A Concrete Naming Certificate for $\mathsf{CantorCylinderCompactness}^{\uparrow}$

Following Brouwer and Bishop–Bridges, the $\mathsf{CantorCylinderCompactness}^{\uparrow}$ packet records the compact finite-prefix surface of Cantor cylinders as displayed NameCert data. It refines the cylinder basis of chapter 2153, reads its ambient Cantor carrier through chapter 1475, uses the finite request windows of chapter 126, and routes tolerance comparisons through chapter 291. The packet gives later StreamName and completion consumers a finite-net handoff surface; it does not assert open-cover compactness, choice-selected branches, quotient stream equality, or ambient Real completeness.

Definition 7332.1 (Cantor cylinder compactness carrier)**.** *A* $\mathsf{CantorCylinderCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, C, R, W, D, F, T, H, K, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{CantorCylinderBasis}^{\uparrow}$ *source row, including the finite binary prefix basis and refinement data.*
2. *C is the ambient* $\mathsf{CantorSpace}^{\uparrow}$ *carrier row read only through finite prefix observations.*
3. *R is the finite binary prefix row family inspected by the compactness request.*
4. *W is the finite cylinder-cover row family, listing the displayed cylinders whose prefixes cover the inspected window.*
5. *D is the* $\mathsf{StreamName}^{\uparrow}$ *request-window row that bounds which prefix lengths and stream positions are visible to the consumer.*
6. *F is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and tolerance ledger attached to the same finite request.*
7. *T is the compact finite-net handoff row, giving the finite prefix representatives delivered to StreamName and completion-facing consumers.*
8. *H stores componentwise* hsame *transport for the basis, Cantor carrier, prefix rows, cover rows, request windows, tolerance ledger, finite-net handoff, replay route, provenance, and local naming.*
9. *K is the displayed* Cont *replay route from request windows to cylinder-cover rows, from cover rows to dyadic tolerance rows, and from those rows to the finite-net handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorCylinderCompactness}^{\uparrow}$, $\mathsf{CantorCylinderBasis}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorCylinderCompactness}^{\uparrow}}$ *row.*

The carrier compares compactness reads only through these finite rows. It has no coordinate for arbitrary open covers, selected infinite branches, quotient equality of streams, or a completeness principle for ambient real numbers.

Theorem 7332.2 (Cantor cylinder compactness NameCert obligations)**.** *For every accepted* CantorCylinderCompac *carrier*

$$Q = (B, C, R, W, D, F, T, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorCylinderCompactness}^{\uparrow}}$ *surface consists exactly of the finite carrier admission, prefix-cover visibility, StreamName request-window discipline, dyadic tolerance discipline, finite-net handoff, componentwise transport, continuation replay, provenance, and nonescape obligations displayed by*

$$B, C, R, W, D, F, T, H, K, P, N.$$

Every obligation is checked inside the finite-prefix rows inherited from $\mathsf{CantorCylinderBasis}^{\uparrow}$ *and* $\mathsf{CantorSpace}^{\uparrow}$ *and the finite request and tolerance rows inherited from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$.

Proof. Carrier admission is projection of the displayed tuple in Definition 7332.1. The rows B, C, R, W supply the finite cylinder basis, Cantor source, inspected prefix rows, and finite cover rows before any compactness-facing handoff is available.

The request and tolerance obligations are then forced by D and F. A consumer may ask only for the prefix lengths and stream positions exposed by the $\mathsf{StreamName}^{\uparrow}$ row D, and each radius or tolerance comparison must pass through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger F.

The finite-net handoff is the row T. It packages exactly the displayed prefix representatives selected by the finite cover rows and tolerance ledger, without adding a branch selector or an infinite product coordinate.

Finally H, K, P, N transport, replay, source, and locally name the same route. Since no displayed row contains an arbitrary open-cover principle, quotient stream equality, selected branch, or ambient Real-completeness certificate, the local NameCert surface cannot export one. □

Theorem 7332.3 (Cantor cylinder compactness prefix-net handoff)**.** *Every StreamName-facing read of an accepted* $\mathsf{CantorCylinderCompactness}^{\uparrow}$ *packet reaches the compact finite-net handoff row T only after the finite binary prefix rows R, finite cylinder-cover rows W, request-window row D, and dyadic radius or tolerance ledger F have been displayed over the source rows B and C. The handoff delivered through H, K, P, N is a finite-prefix net surface, not an open-cover compactness theorem, choice-selected branch, quotient stream equality, or ambient Real-completeness claim.*

Proof. Start with a StreamName-facing request. The source cylinder basis B and Cantor carrier C expose only finite prefix observations, so the request cannot inspect an infinite stream as a host function.

Read the finite rows in order. The prefix family R supplies the visible binary prefixes, the cover family W records which displayed cylinders cover the inspected window, the request row D bounds the visible stream positions, and the tolerance ledger F records the dyadic radius data attached to that same request.

Only then can the compact handoff row T be consumed. It exports the finite representatives already displayed by R, W, D, F to later StreamName and completion-facing consumers.

The structural rows H, K, P, N preserve this route componentwise. They do not add open-cover compactness, a branch selector, quotient stream equality, or ambient Real completeness, so the exported handoff remains a finite-prefix compactness packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorCylinderCompactness}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorCylinderCompactness}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7332.1, Definition 7332.2, and Definition 7332.3 seed the StreamName-facing finite-prefix compactness surface over Cantor cylinder basis, CantorSpace, StreamName windows, DyadicRatCore tolerance rows, compact finite-net handoff rows, transport, replay, provenance, and local NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorCylinderCompactness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim open-cover compactness, choice-selected branches, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: The next step is obligationClosure: carrier admission, prefix-cover visibility, StreamName request-window discipline, dyadic tolerance discipline, finite-net handoff, and nonescape must be discharged as local rows.

Constructive story: The construction starts from finite Cantor cylinder prefixes, displays the requested StreamName window, attaches DyadicRatCore tolerance rows, and exports only the finite-net handoff preserved by hsame transport, Cont replay, Pkg provenance, and local NameCert.

Chapter 7333

Rational squeeze NameCert construction

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalSqueeze}^{\uparrow}$ packet records the finite rational lower–upper squeeze data used by the L10 regular-Cauchy and Real completion route. It sits between the dyadic arithmetic core of chapter 291, the regular rational sequence surface of chapter 301, the terminal Real seal of chapter 38, the located comparison route of chapter 1730, and the Cauchy-tail basis of chapter 1868. The packet is a finite means certificate for reading a rational squeeze as a regular rational name and then as a sealed Real row; it is not a Dedekind cut, quotient real equality, selected limit, open-cover compactness principle, or ambient completeness axiom.

Definition 7333.1 (Rational squeeze carrier)**.** *A* $\mathsf{RationalSqueeze}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, M, D, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational bound stream, read as finite rational endpoint rows with monotone lower-side comparison data.*
2. *U is the upper rational bound stream, read as finite rational endpoint rows with monotone upper-side comparison data.*
3. *M is the middle stream row, obtained by finite dyadic midpoint or rational averaging rows from the displayed lower and upper bounds.*
4. *D is the dyadic tolerance row supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*; it records denominator clearance, dyadic width budgets, and the visible tolerance cell inspected by a consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row; it records the regular rational name obtained by reading the middle stream against the dyadic tolerance row.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, displaying the terminal real-name package obtained from the regular rational readback.*
7. *O is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *order route, recording the lower–middle and middle–upper comparisons needed by a located-order consumer.*
8. *H records componentwise* hsame *transport for lower bounds, upper bounds, middle stream, dyadic tolerance, regular readback, Real seal, located-order rows, replay rows, provenance, and local naming.*

9. *C records displayed* Cont *replay for rereading the squeeze request through $L, U, M, D, R, E, O, H, P, N$ without changing the finite packet.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalSqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{CauchyTailBasis}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalSqueeze}^{\uparrow}}$ *row naming exactly this squeeze packet.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host real equality, quotient interval equality, a selected limit point, countable choice, open-cover compactness, or ambient Real completeness.

Lean4 (statement) `BEDC.Derived.RationalSqueezeUp`

Theorem 7333.2 (Rational squeeze NameCert obligations)**.** *For every accepted* $\mathsf{RationalSqueeze}^{\uparrow}$ *carrier $S = (L, U, M, D, R, E, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalSqueeze}^{\uparrow}}$ *obligations are exhausted by lower-bound exposure, upper-bound exposure, middle-stream exposure, dyadic-tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal exposure, located-order route exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7333.1. The only numerical rows available to a consumer are L, U, M, D, R, E, O, and each row is displayed inside the finite packet before any downstream read occurs.

The NameCert surface is therefore rowwise. The lower and upper bounds expose the squeeze interval, the middle row exposes the finite averaging stream, the dyadic tolerance row exposes the visible width budget, the regular readback row exposes the rational name, the Real seal row exposes the terminal package, and the located-order row exposes the comparison route.

The remaining rows are structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve precisely the same packet, so the certificate has no route to host real equality, quotient interval equality, selected limits, countable choice, open-cover compactness, or ambient completeness. □

Theorem 7333.3 (Rational squeeze RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalSqueeze}^{\uparrow}$ *packet must read the middle stream row M through the dyadic tolerance row D and the Cauchy-tail basis supplied by* $\mathsf{CauchyTailBasis}^{\uparrow}$ *before it may consume the readback row R. The handoff exports a regular rational sequence surface, not an arbitrary Cauchy wrapper or a host quotient name.*

Proof. Read the middle stream and tolerance rows of Definition 7333.1. The row M supplies the rational approximants, while D supplies the dyadic width and denominator-clearance data needed by a regular rational name.

Next apply the Cauchy-tail basis route. A consumer may inspect only the finite tolerance cell named by D and the corresponding middle-stream window, so the regularity read is forced through $\mathsf{CauchyTailBasis}^{\uparrow}$ before the row R is available.

Finally attach the structural rows. The transports, replay, provenance, and local name keep the readback componentwise identical to the displayed squeeze packet. Thus the handoff exports the finite $\mathsf{RegSeqRat}^{\uparrow}$ readback row and no host quotient name. □

Theorem 7333.4 (Rational squeeze Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalSqueeze}^{\uparrow}$ *packet must obtain the real-name seal E from the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, with the lower and upper rational rows L, U and the dyadic tolerance row D still visible as the finite witness surface. The seal does not identify quotient reals or select an external limit.*

Proof. First expose the regular rational row R from Definition 7333.3. This row is already tied to M and D, with L and U recording the squeeze bounds that constrain the middle stream.

The Real seal row E is then read as the terminal package over that regular rational name. Because the carrier keeps the squeeze rows displayed, the Real consumer receives the lower bound, upper bound, middle stream, tolerance, and readback evidence together.

The rows H, C, P, N preserve the same package during transport, replay, provenance, and local naming. No quotient real equality, selected external limit, countable choice, or ambient completeness principle is introduced by the seal. □

Theorem 7333.5 (Rational squeeze located-order route)**.** *Every located-order read exported by an accepted* $\mathsf{RationalSqueeze}^{\uparrow}$ *packet factors through the displayed row O: the lower-bound stream is compared with the middle stream, the middle stream is compared with the upper-bound stream, and each comparison is reread through the same dyadic tolerance row D before a* $\mathsf{LocatedRealComparison}^{\uparrow}$ *consumer can inspect it.*

Proof. Project the order route row O. It is displayed as part of the carrier and can mention only the lower, middle, upper, and tolerance rows already present in L, U, M, D.

The located comparison is therefore finite. Lower–middle and middle–upper reads are replayed at the visible dyadic tolerance, and the same tolerance row is transported by H before any consumer sees the route.

The provenance and local naming rows export this comparison as a $\mathsf{LocatedRealComparison}^{\uparrow}$ dependency of the squeeze packet. Since no host interval or quotient real equality is a carrier coordinate, the route remains a finite located-order read. □

Theorem 7333.6 (Rational squeeze Cauchy-window consumer)**.** *Every Cauchy-window consumer of an accepted* $\mathsf{RationalSqueeze}^{\uparrow}$ *packet receives a finite window consisting of the lower and upper rational bounds L, U, the middle stream M, the dyadic tolerance row D, the* $\mathsf{CauchyTailBasis}^{\uparrow}$ *handoff, and the structural rows H, C, P, N. The consumer cannot replace this window by an infinite choice sequence, an open-cover compactness argument, or ambient Real completeness.*

Proof. Read the squeeze window directly from the carrier. The rows L, U, M, D display the lower endpoint, upper endpoint, middle stream, and tolerance budget of the finite window.

The Cauchy-tail handoff is the same handoff used by Definition 7333.3. It lets a consumer reread a visible tolerance cell and its corresponding middle-stream approximant before passing to the regular rational readback.

Finally, H, C, P, N transport, replay, source, and name that finite window. These rows do not add an infinite choice sequence, an open-cover compactness argument, selected limits, quotient equality, or ambient Real completeness, so the Cauchy-window consumer remains bound to the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalSqueeze}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalSqueeze}^{\uparrow}$ packet is generated from lower rational bounds, upper rational bounds, a middle rational stream, dyadic tolerance rows, a RegSeqRat readback, a Real seal, a located-order route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalSqueeze}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7333.1, Definition 7333.2, Definition 7333.3, Definition 7333.4, Definition 7333.5, and Definition 7333.6 seed the finite squeeze certificate used by the L10 regular-Cauchy and Real completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalSqueeze}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient real equality, selected limits, countable choice, open-cover compactness, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked lower-bound exposure, upper-bound exposure, middle-stream readback, dyadic-tolerance exactness, RegSeqRat handoff, Real seal, located-order route, transport, replay, provenance, and local naming for BEDC.Derived.RationalSqueezeUp.

Chapter 7334

A Concrete Naming Certificate for $\mathsf{IsaacsEquation}^{\uparrow}$

Following Isaacs and Fleming–Soner on differential games and viscosity solutions, the $\mathsf{IsaacsEquation}^{\uparrow}$ packet records the finite BEDC surface by which a two-player dynamic-programming boundary can be read as an Isaacs-equation certificate. It sits after the dynamic-programming route of chapter 5875, the viscosity-solution boundary of chapter 2177, the PDE carrier of chapter 194, the ODE evolution surface of chapter 193, and the terminal real-name seal of chapter 38. The packet names a finite game-PDE handoff only; it does not assert comparison, existence, uniqueness, measurable strategy selection, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7334.1 (Isaacs equation carrier)**.** *An* $\mathsf{IsaacsEquation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, A, B, V, D, E, O, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the differential-game state and payoff row;*
2. *A and B are the two finite control-action boundary rows;*
3. *V is the value-function candidate row read through* $\mathsf{DynamicProgramming}^{\uparrow}$*;*
4. *D is the dynamic-programming handoff row recording the local min-max or max-min route;*
5. *E is the* $\mathsf{PDE}^{\uparrow}$ *row that displays the Hamilton-Jacobi-Isaacs expression;*
6. *O is the* $\mathsf{ODE}^{\uparrow}$ *row that records the controlled state-evolution boundary;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the game, control, dynamic-programming, PDE, and ODE rows are visible;*
8. *H records componentwise* hsame *transport for $G, A, B, V, D, E, O, R, C, P, N$;*
9. *C records displayed* Cont *replay through the game, control, dynamic-programming, viscosity, PDE, ODE, real-seal, transport, provenance, and local naming rows;*
10. *P is the* Pkg *provenance over* $\mathsf{IsaacsEquation}^{\uparrow}$, $\mathsf{DynamicProgramming}^{\uparrow}$, $\mathsf{ViscositySolution}^{\uparrow}$, $\mathsf{PDE}^{\uparrow}$, $\mathsf{ODE}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IsaacsEquation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a full PDE comparison theorem, an existence theorem, a uniqueness theorem, selected measurable strategies, quotient equality of value functions, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.IsaacsEquationUp.IsaacsEquationTasteGate single carrier alignment`

Theorem 7334.2 (Isaacs equation NameCert obligations)**.** *For every accepted* $\mathsf{IsaacsEquation}^{\uparrow}$ *carrier* $I = (G, A, B, V, D, E, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IsaacsEquation}^{\uparrow}}$ *surface consists exactly of game-state exposure, both finite control-action rows, value-candidate exposure, dynamic-programming handoff, viscosity-facing PDE exposure, controlled ODE exposure, real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside those rows.*

Proof. Project the carrier of Definition 7334.1. The rows G, A, B, V, D, E, O, R are the only analytic data visible to the local certificate.

Read the dynamic-programming and viscosity boundaries through the named siblings. The dynamic-programming row supplies the finite min-max handoff, while the viscosity-solution, PDE, and ODE rows keep the Hamilton-Jacobi-Isaacs expression attached to displayed evolution data.

The structural rows H, C, P, N preserve, replay, source, and name the packet. Since the carrier contains no comparison theorem, existence theorem, uniqueness theorem, selected strategy, quotient value-function equality, Lean theorem, or bridge row, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.IsaacsEquationUp.IsaacsEquationCarrier namecert obligations`

Theorem 7334.3 (Isaacs equation dynamic-programming route)**.** *Every* $\mathsf{ViscositySolution}^{\uparrow}$*,* $\mathsf{PDE}^{\uparrow}$*,* $\mathsf{ODE}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{IsaacsEquation}^{\uparrow}$ *packet must first read*

$$G, A, B, V, D, E, O, H, C, P, N$$

before it can consume the real-seal row R*. The route is the displayed dynamic-programming handoff for a differential-game equation, not a proof of PDE comparison, existence, uniqueness, selected optimal strategies, or ambient completeness.*

Proof. Start from the accepted packet $I = (G, A, B, V, D, E, O, R, H, C, P, N)$. The carrier orders the value-candidate row and dynamic-programming handoff before the PDE, ODE, and real-seal rows can be consumed.

Use Definition 7334.2. Its obligations bind every downstream read to the same game-state, control-action, dynamic-programming, PDE, ODE, transport, replay, provenance, and local naming rows.

The sibling chapters then supply the horizontal route: dynamic programming supplies the finite handoff, viscosity solutions supply the weak PDE reading discipline, PDE and ODE supply the displayed equation and evolution rows, and Real supplies only the terminal seal. No comparison, existence, uniqueness, strategy-selection, quotient, or ambient-completeness coordinate can be substituted for that route. □

Falsifiable prediction: A consumer that claims an $\mathsf{IsaacsEquation}^{\uparrow}$ read while omitting the differential-game row, one of the two control-action rows, the $\mathsf{DynamicProgramming}^{\uparrow}$ value handoff, the $\mathsf{ViscositySolution}^{\uparrow}$ discipline, the $\mathsf{PDE}^{\uparrow}$ expression row, the $\mathsf{ODE}^{\uparrow}$ evolution row, the $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 7334.1.

Independence witness: dynamic-programming, viscosity-solution, pde, ode, real: no listed sibling is bijective to $\mathsf{IsaacsEquation}^{\uparrow}$ because this packet alone binds two-player game rows, min-max dynamic-programming handoff, viscosity-facing PDE expression, controlled ODE evolution, real sealing, transport, replay, provenance, and local naming as one finite differential-game certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{IsaacsEquation}^{\uparrow}}$.

Constructive story: An $\mathsf{IsaacsEquation}^{\uparrow}$ packet is generated from differential-game source rows, two finite control-action rows, a value-candidate row, a dynamic-programming handoff, viscosity-facing PDE and ODE rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IsaacsEquation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7334.1, Definition 7334.2, and Definition 7334.3 seed the finite differential-game dynamic-programming route for Isaacs-equation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IsaacsEquation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim PDE comparison, PDE existence, uniqueness, measurable strategy selection, quotient equality of value functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked game-row admission, control-action exposure, dynamic-programming handoff, viscosity-facing PDE and ODE routing, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IsaacsEquationUp.

Chapter 7335

A Concrete Naming Certificate for $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$

The $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ packet realizes the missing-certificate analysis of section 9462.7 as finite NameCert data. It sits beside the classifier boundary of chapter 595, the classifier morphism surface of chapter 590, the classifier-shift ledger residue of chapter 6985, and the audit gate boundary of chapter 526. The packet separates supported-but-uncertified behavior from unexplained behavior and ledger-complete explanation; it is not a hidden explanation theorem or a classifier-correctness proof.

Definition 7335.1 (Black-box certificate deficit carrier)**.** *A* $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, S, M, L, F, X, R, V, H, C, P, N)$$

with the following displayed rows:

1. *T is the behavior-target row;*
2. *S is the support-classifier row witnessing that a classifier candidate supports T;*
3. *M is the missing-row inventory for source, pattern, classifier, stability, ledger, or verification rows;*
4. *L is the deficit ledger that records which NameCert rows are absent or incomplete;*
5. *F is the closed-explanation refusal row;*
6. *X is the classifier transport row from* $\mathsf{ClassifierBoundary}^{\uparrow}$ *and* $\mathsf{ClassifierMorphism}^{\uparrow}$ *surfaces;*
7. *R is the replay row for behavior, support, missing-row, ledger, and refusal reads;*
8. *V is the provenance row tying the packet to the black-box missing-certificate vision and the classifier-facing siblings;*
9. *H records componentwise* hsame *transport for $T, S, M, L, F, X, R, V, C, P, N$;*
10. *C records displayed* Cont *replay through the behavior target, support classifier, missing-row inventory, deficit ledger, refusal, transport, replay, provenance, and local naming rows;*
11. *P is the* Pkg *provenance over* $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$, $\mathsf{ClassifierBoundary}^{\uparrow}$, $\mathsf{ClassifierMorphism}^{\uparrow}$, $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}^{\uparrow}}$ *row.*

The carrier has no coordinate for unexplained behavior with no support classifier, ledger-complete certified explanation, global classifier correctness, hidden causal proof, Lean verification, or bridge checking.

Theorem 7335.2 (Black-box certificate deficit NameCert obligations)**.** *For every accepted* $\mathsf{BlackBoxCertificateDefic}$ *carrier* $B = (T, S, M, L, F, X, R, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}^{\uparrow}}$ *surface consists exactly of behavior-target exposure, support-classifier exposure, missing-row inventory, deficit-ledger exposure, closed-explanation refusal, classifier transport, displayed replay, provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7335.1. The behavior target T and support classifier S are visible before any deficit claim can be read.

Read the missing-row inventory and ledger rows next. The inventory M identifies which NameCert rows are absent or incomplete, while L records the deficit as finite ledger data and F refuses a closed-explanation read until those rows are supplied.

The classifier-facing siblings constrain transport. Classifier boundary and classifier morphism rows supply the support route, classifier-shift ledger residue records whether a classifier disagreement is present, and audit gate boundary records the local admission route. The structural rows R, V, H, C, P, N preserve and name exactly that surface. □

Theorem 7335.3 (Black-box certificate deficit closed-explanation refusal)**.** *An accepted* $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ *packet may be read as black-box behavior only when the support-classifier row* S *is displayed and the missing-row inventory* M *or deficit ledger* L *is nonempty. It may not be read as unexplained behavior, because* S *is present, and it may not be read as ledger-complete explanation, because* F *records the closed-explanation refusal.*

Proof. Start with the behavior target and support classifier. The carrier stores T and S as distinct rows, so the packet is not an unexplained-behavior row.

Use the deficit rows M, L, F. Missing or incomplete certificate rows are recorded before the refusal row is consumed, and F blocks any closed-explanation read that lacks the named NameCert rows.

Finally inspect the transport, replay, provenance, and local naming rows. They preserve the supported-but-uncertified status and provide no hidden certificate row, classifier-correctness proof, or ledger-complete explanation outside the displayed packet. □

Falsifiable prediction: Any closed-explanation read of a $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ packet without a local $\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}^{\uparrow}}$ row and the displayed missing-row inventory refutes the carrier of Definition 7335.1.

Independence witness: classifierboundary, classifiermorphism, classifier-shift-ledger-residue, auditgateboundary, namecert-facing siblings: $\mathsf{ClassifierBoundary}^{\uparrow}$ exposes classifier admission without the deficit ledger; $\mathsf{ClassifierMorphism}^{\uparrow}$ transports classifier rows without recording missing explanation rows; $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ records classifier disagreement without forcing a black-box deficit; $\mathsf{AuditGateBoundary}^{\uparrow}$ localizes admission without carrying the behavior-support pair; NameCert-facing siblings close certificates rather than naming the supported-but-uncertified gap. None is bijective to $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ under NameCert-row preservation.

The Lean follow-up for this carrier must provide $\mathsf{Nontrivial}\ \mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ instances in `BEDC.Derived.BlackBoxCertificateDeficitUp.TasteGate` before any checked TasteGate theorem can serve as a closure witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}\uparrow}$.

Constructive story: A $\mathsf{BlackBoxCertificateDeficit}^{\uparrow}$ packet is generated from a behavior target, support classifier, missing-row inventory, deficit ledger, closed-explanation refusal, classifier transport, replay, provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7335.1, Definition 7335.2, and Definition 7335.3 seed the finite certificate-deficit route for supported but uncertified behavior.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BlackBoxCertificateDeficit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unexplained behavior, ledger-complete explanation, classifier correctness, hidden causal explanation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked behavior-target exposure, support-classifier exposure, missing-row inventory, deficit-ledger accounting, closed-explanation refusal, classifier transport, replay, provenance, local naming, Nontrivial, and FieldFaithful rows for BEDC.Derived.BlackBoxCertificateDeficitUp.

Chapter 7336

A Concrete Naming Certificate for $\mathsf{Sheafification}^{\uparrow}$

Following Grothendieck's SGA 4 site formalism and the Mac Lane–Moerdijk account of sheaves in geometry and logic, the $\mathsf{Sheafification}^{\uparrow}$ packet records the finite BEDC surface by which a displayed presheaf is routed to a sheaf-facing carrier. It sits after the categorical source of chapter 61, the topological and site carriers of chapter 88 and chapter 6718, the presheaf restriction surface of chapter 150, and the gluing target of chapter 151. The packet names the separated-refinement, cover-local matching, plus-style gluing, transport, replay, provenance, and local naming rows; it is not a host sheafification functor, not a quotient of compatible families, and not a topos-level reflector.

Definition 7336.1 (Sheafification carrier)**.** *A* $\mathsf{Sheafification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Cat}^{\uparrow}$ *source row naming the displayed objects and arrows over which the presheaf is read.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *or site-open row supplying the indexed open windows used by cover requests.*
3. *J is the* $\mathsf{GrothendieckTopology}^{\uparrow}$ *coverage row, including covering sieves, pullback-stability rows, and local-transitivity rows.*
4. *P is the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row over C, T, J, with section carriers and restriction comparison data.*
5. *L is the locality-refinement ledger, recording matching families after every displayed cover refinement.*
6. *G is the gluing ledger, recording the accepted glued section row for each compatible local family.*
7. *S is the resulting* $\mathsf{Sheaf}^{\uparrow}$ *target row, exposed only through the same locality and gluing data.*
8. *H records componentwise* hsame *transport for the category, topology, site, presheaf, locality, gluing, sheaf, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from presheaf restriction reads through the locality and gluing ledgers to the sheaf target.*

10. *Q is the* Pkg *provenance over* $\mathsf{Sheafification}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, T, J, P, L, G, S, H, R, Q, N$. It has no coordinate for a host reflector, quotient of compatible families, global choice of covers, topos object, ambient powerset equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.SheafificationUp.TasteGate.SheafificationTasteGate single carrier alignment`

Theorem 7336.2 (Sheafification NameCert obligations)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *surface consists exactly of category-source exposure through C, topology-window exposure through T, Grothendieck coverage through J, presheaf restriction exposure through P, locality-refinement rows through L, gluing rows through G, the sheaf target row through S, componentwise transport through H, displayed replay through R, provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCarrier namecert obligations`

Proof. Project the carrier of Definition 7336.1. The source side is the displayed category, topology or site window, Grothendieck coverage row, and presheaf restriction row.

The locality ledger L records the cover-refinement comparisons that make matching families visible. The gluing ledger G records the resulting glued section row only after those matching-family rows have been displayed.

The sheaf target S is therefore read from P, L, G over the same C, T, J source. The structural rows H, R, Q, N transport, replay, source, and name that same route. Since no host reflector, quotient family, selected cover, topos object, or powerset equality is a carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 7336.3 (Sheafification locality-gluing scope)**.** *Every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier routes presheaf restriction, cover-local matching, separated refinement, plus-style gluing,* $\mathsf{Sheaf}^{\uparrow}$ *target exposure,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows through the displayed dependencies*

$$\mathsf{Cat}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{GrothendieckTopology}^{\uparrow}, \mathsf{PreSheaf}^{\uparrow}, \mathsf{Sheaf}^{\uparrow}.$$

Thus any locality or gluing consumer that preserves the local certificate must read the same category source, topology windows, Grothendieck coverage, presheaf restriction, locality ledger, gluing ledger, sheaf target, transport, replay, provenance, and naming rows; it cannot substitute a host reflector, quotient of compatible families, selected cover system, ambient powerset equality, or topos-level object for the displayed BEDC route.

Proof. Project the carrier rows of Definition 7336.1. The source side is fixed by C, T, J, P: category data, topology windows, Grothendieck coverage, and presheaf restrictions are visible before locality can be read.

Apply the NameCert obligations of Definition 7336.2. They identify L as the cover-local matching and separated-refinement ledger, G as the plus-style gluing ledger, and S as the sheaf target row exposed only after those rows are displayed.

The remaining rows H, R, Q, N preserve that same route by componentwise transport, displayed replay, provenance, and local naming. Since the obligation theorem exhausts the accepted certificate surface, a locality or gluing consumer cannot leave the displayed $\mathsf{Cat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, and $\mathsf{Sheaf}^{\uparrow}$ dependencies. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalityGluingScope`

Theorem 7336.4 (Sheafification sheaf handoff)**.** *Every downstream* $\mathsf{Sheaf}^{\uparrow}$ *consumer of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet receives only the displayed sheaf row* S *together with its dependency route*

$$\mathsf{Cat}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{GrothendieckTopology}^{\uparrow}, \mathsf{PreSheaf}^{\uparrow} \longmapsto L \longmapsto G \longmapsto \mathsf{Sheaf}^{\uparrow}.$$

The handoff exposes the category source, topology or site windows, coverage row, presheaf restriction row, locality ledger, gluing ledger, transport row, replay row, provenance row, and local naming row; it does not expose a host sheafification functor, quotient of compatible families, selected cover system, topos object, or ambient powerset equality.

Proof. Start with the carrier projection. A sheaf-facing consumer can read S only after C, T, J, P, L, G have been displayed, because S is the target row of that finite route.

Use the topology and site rows to interpret covers as displayed $\mathsf{Topology}^{\uparrow}$ windows and $\mathsf{GrothendieckTopology}^{\uparrow}$ covering sieves. The presheaf row P supplies restriction data on those windows, while L records locality under cover refinement and G records the glued section row.

The handoff to $\mathsf{Sheaf}^{\uparrow}$ is therefore the same row package, transported by H and replayed by R. The provenance and local naming rows Q, N name the consumed siblings without adding a host reflector or quotient object. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSheafHandoff`

Theorem 7336.5 (Sheafification coverage exactness)**.** *Every cover-local read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N)$$

must expose the category row C*, topology-window row* T*, Grothendieck-coverage row* J*, and presheaf-restriction row* P *before any locality data in* L *may be consumed.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverageExactness`

Proof. Project the displayed carrier tuple. The only cover source coordinates are C, T, J, P: C supplies the $\mathsf{Cat}^{\uparrow}$ objects and arrows, T supplies the $\mathsf{Topology}^{\uparrow}$ windows, J supplies the $\mathsf{GrothendieckTopology}^{\uparrow}$ covering and pullback-stability rows, and P supplies the $\mathsf{PreSheaf}^{\uparrow}$ restriction data over that source.

The locality ledger L is typed by those displayed cover and restriction rows. A read that reaches L without first exposing C, T, J, P has no BEDC row from which cover-locality can be interpreted.

The structural rows H, R, Q, N preserve this order. Componentwise hsame transport, Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row can carry the exposed source route, but none of them supplies an independent coverage coordinate. Thus coverage exactness forces every locality read through C, T, J, P first. □

Theorem 7336.6 (Sheafification locality refinement)**.** *For an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet, every matching-family row recorded in* L *is stable under displayed* $\mathsf{GrothendieckTopology}^{\uparrow}$ *refinement and componentwise* hsame *transport: replacing a cover by a displayed refinement in* J *and transporting the component section rows by* H *yields another locality row still recorded by* L *over the same presheaf row* P*.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalityRefinement`

Proof. Read L from the carrier tuple after C, T, J, P have been exposed. By definition, L is the locality-refinement ledger, so its entries are indexed by the displayed cover-refinement comparisons carried by the Grothendieck-topology row J.

Now apply the transport row H. It records componentwise hsame transport for the category, topology, site, presheaf, locality, gluing, sheaf, replay, provenance, and local naming rows. Transporting the component sections therefore preserves the same P-restriction comparisons rather than changing to a host family.

The replay row R and provenance row Q keep this refinement inside the same $\mathsf{Sheafification}^{\uparrow}$ package, and N names it as part of the local $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ surface. Hence matching-family rows in L are stable under the displayed refinement and transport data. □

Theorem 7336.7 (Sheafification gluing exactness)**.** *The sheaf target row S of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet is consumed only after the locality ledger L and gluing ledger G have displayed compatible local sections for the presheaf row P over the cover rows T, J.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationGluingExactness`

Proof. Begin with the route theorem Definition 7336.4. It places the sheaf-facing row S downstream of the displayed category, topology, Grothendieck coverage, presheaf, locality, and gluing rows.

The ledger L records the matching-family and refinement checks. The ledger G records the accepted glued section row for compatible local families. Therefore S has no independent coordinate from which a sheaf target can be read before L and G have been displayed.

Finally use H, R, Q, N. Componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming preserve the same route from P through L and G to the $\mathsf{Sheaf}^{\uparrow}$ target row. They do not introduce a quotient of compatible families or a host sheafification functor, so the gluing read is exact at the displayed BEDC surface. □

Theorem 7336.8 (Sheafification separated refinement exactness)**.** *Let*

$$A = (C, T, J, P, L, G, S, H, R, Q, N)$$

be an accepted $\mathsf{Sheafification}^{\uparrow}$ *carrier. Every separated-refinement read of A must pass through the displayed category row C, topology-window row T, Grothendieck-coverage row J, presheaf-restriction row P, and locality ledger L before any separated sheaf-facing row may be consumed. The route is transported by* hsame *through H, replayed by* Cont *through R, sourced by* Pkg *through Q, and named only by the local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row N.*

Proof. The carrier definition exposes separated refinement only as a locality entry in L. That row is indexed by the displayed source data C, T, J, P, so a separated read with no prior source exposure has no cover, no restriction comparison, and no matching-family ledger on which to stand.

Now apply the structural rows. The hsame transport H moves the same displayed components, Cont replay R records the same route, Pkg provenance Q identifies the sibling packets, and N names the local certificate. None of these rows creates an independent host reflector or quotient-compatible-family object.

Thus every separated-refinement read factors through C, T, J, P, L before reaching the sheaf-facing surface. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSeparatedRefinementExactness`

Theorem 7336.9 (Sheafification plus gluing replay)**.** *For an accepted* Sheafification$^{\uparrow}$ *carrier*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

every plus-style gluing replay must pass through the locality ledger L, gluing ledger G, sheaf row S, componentwise transport row H, and displayed replay row R. The replay remains a finite BHist *route using* hsame*,* Cont*,* Pkg*, and* NameCert *data; it is not a quotient construction of compatible families.*

Proof. First read L. It records the compatible matching-family rows over the displayed cover and restriction data. Then read G, which records the accepted glued section row for those compatible families.

The sheaf row S is downstream of that locality-and-gluing pair. Transport through H and replay through R carry the same finite route into a sheaf-facing consumer without adding a separate plus object.

Finally Q and N package and name the route. Since the carrier has no coordinate for quotient families or selected covers, the plus-style replay is exactly the displayed path through L, G, S, H, R. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusGluingReplay`

Theorem 7336.10 (Sheafification plus-construction boundary)**.** *For an accepted* Sheafification$^{\uparrow}$ *carrier*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

every plus-construction read factors through $C, T, J, P, L, G, S, H, R, Q, N$ and cannot read quotient compatible-family data. The plus-facing consumer receives the displayed category source, topology window, Grothendieck coverage, presheaf restriction, locality ledger, gluing ledger, sheaf target row, transport, replay, provenance, and local naming rows, but no host quotient object and no selected compatible-family representative.

Proof. Project the carrier from Definition 7336.1. The tuple exposes exactly the rows $C, T, J, P, L, G, S, H, R, Q, N$, so any plus-construction read starts inside that finite surface.

Use Definition 7336.5 and Definition 7336.9. Coverage exactness forces the source rows C, T, J, P before locality is consumed, while plus gluing replay keeps the plus-style route inside L, G, S, H, R.

Finally apply Definition 7336.21. Since quotient compatible-family data and selected representatives are absent from the carrier and excluded by the local certificate boundary, the plus-construction read cannot escape the displayed rows. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusConstructionBoundary`

Theorem 7336.11 (Sheafification separated-reflection route)**.** *For an accepted* Sheafification$^{\uparrow}$ *carrier, separated reflection is visible only through the displayed presheaf restriction row P and locality row L before the sheaf row S is consumed. The route may use C, T, J to interpret covers and H, R, Q, N to transport, replay, source, and name the read, but it has no independent host reflector row.*

Proof. Begin with Definition 7336.8. A separated-refinement read must expose C, T, J, P, L before any sheaf-facing row is available.

The presheaf row P supplies the restriction comparisons, and the locality row L records their cover-local separated behavior. The sheaf row S is downstream of that pair by Definition 7336.7, so it cannot be consumed as an independent reflector.

The structural rows H, R, Q, N preserve the same route. They transport, replay, source, and name the displayed separated-reflection boundary without adding a host reflector or quotient compatible-family object. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSeparatedReflectionRoute`

Theorem 7336.12 (Sheafification cover-gluing nonescape)**.** *Every cover-gluing read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier consumes* $\mathsf{GrothendieckTopology}^{\uparrow}$ *coverage,* $\mathsf{PreSheaf}^{\uparrow}$ *restriction, the locality ledger L, and the gluing ledger G before reaching the sheaf row S. The read exports no host reflector, selected cover family, quotient of compatible families, or topos-level reflection object.*

Proof. First expose the cover source. By Definition 7336.5, the Grothendieck coverage row J and presheaf restriction row P are read before any locality ledger entry in L can be consumed.

Next apply Definition 7336.7. The gluing ledger G is the displayed row that records accepted glued sections, and the sheaf row S is read only after locality and gluing data are available.

The non-escape boundary follows from Definition 7336.21. Since host reflectors, selected cover families, quotient compatible-family objects, and topos-level reflectors are not carrier coordinates, cover gluing remains inside $J, P, L, G, S, H, R, Q, N$. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverGluingNonescape`

Theorem 7336.13 (Sheafification source obligation package)**.** *For an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

every sheaf-facing consumer must expose the $\mathsf{Cat}^{\uparrow}$ *source C, the* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{GrothendieckTopology}^{\uparrow}$ *cover rows T, J, the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row P, locality rows L, gluing rows G, the* $\mathsf{Sheaf}^{\uparrow}$ *target row S, componentwise* hsame *transport H, displayed* Cont *replay R,* Pkg *provenance Q, and the local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row N before it reads S.*

Proof. First apply Definition 7336.5. It forces every locality read through C, T, J, P before the ledger L is available.

Next use Definition 7336.6 and Definition 7336.7. Locality rows in L are stable under displayed refinement, and the sheaf target S is consumed only after L and G have displayed compatible local sections over the presheaf row P.

The structural rows complete the package. The tuple projection supplies H, R, Q, N, and these rows transport, replay, source, and name the same route rather than adding a host reflector, quotient family, selected cover, or topos object. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSourceObligationPackage`

Theorem 7336.14 (Sheafification source exposure obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

any sheafification read must expose the $\mathsf{Cat}^{\uparrow}$ *row C, the* $\mathsf{Topology}^{\uparrow}$ *window row T or* $\mathsf{GrothendieckTopology}^{\uparrow}$ *coverage row J, and the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row P before it consumes locality data in L. The exposure is the finite BEDC source surface for the read, not a host site, chosen cover family, quotient compatible-family object, or reflector.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSourceExposureObligation`

Proof. Project the carrier of Definition 7336.1. The rows C, T, J, P are the only source rows that type a cover-local sheafification request, and L is downstream of those rows in the displayed tuple.

Apply Definition 7336.5 and Definition 7336.13. Coverage exactness forces the coverage and presheaf restriction data to be visible before locality is read, while the source package records the same order for every sheaf-facing consumer.

The structural rows H, R, Q, N only preserve this source route by transport, replay, provenance, and local naming. Since no carrier coordinate supplies a host site, selected cover family, quotient compatible-family object, or reflector, the read remains a finite source exposure through C, T, J, P before L. □

Theorem 7336.15 (Sheafification local gluing obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, a consumer may read the* $\mathsf{Sheaf}^{\uparrow}$ *target row* S *only after the locality ledger* L *and gluing ledger* G *have both been consumed over the displayed source rows* C, T, J, P*. The route records locality and gluing exactness inside the carrier and exports no quotient of compatible families or host reflector.*

Proof. Start with source exposure from Definition 7336.14. It places C, T, J, P before locality, so every gluing request is already typed by the displayed category, topology or coverage, and presheaf restriction rows.

Use Definition 7336.6 to keep locality data in L stable under displayed cover refinement. Then apply Definition 7336.7: the gluing ledger G is consumed after locality and before the target sheaf row S.

Finally invoke Definition 7336.21. The local certificate boundary excludes quotient compatible-family data and host reflectors, so the target row can only be reached through L, G, H, R, Q, N inside the finite carrier. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalGluingObligation`

Theorem 7336.16 (Sheafification target handoff obligation)**.** *For an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

every final target handoff to a $\mathsf{Sheaf}^{\uparrow}$ *consumer factors through the complete route* $C, T, J, P, L, G, S, H, R, Q, N$*. The consumer may inspect the category source, topology or Grothendieck coverage, presheaf restrictions, locality ledger, gluing ledger, sheaf target row, transport, replay, provenance, and local name, but it cannot bypass them through a quotient compatible-family object, a host reflector, or a detached sheafification functor.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationTargetHandoffObligation`

Proof. First expose the source and locality rows. By Definition 7336.14, every target-facing read starts with C, T, J, P and reaches L only through that source surface.

Next consume the gluing rows. Definition 7336.15 places G between L and the $\mathsf{Sheaf}^{\uparrow}$ row S, and Definition 7336.7 identifies this as the displayed gluing exactness boundary.

The handoff is completed by H, R, Q, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same ordered route C, T, J, P, L, G, S for the downstream consumer. Since Definition 7336.21 excludes host reflectors and quotient compatible-family data, the final handoff has no coordinate outside $C, T, J, P, L, G, S, H, R, Q, N$. □

Theorem 7336.17 (Sheafification source cover locality)**.** *Every locality read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *source package exposes the* $\mathsf{Cat}^{\uparrow}$ *row* C*, the* $\mathsf{Topology}^{\uparrow}$ *window row* T*, the* $\mathsf{GrothendieckTopology}^{\uparrow}$ *coverage row* J*, and the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row* P *before any locality row in* L *is consumed. The exposure is a finite source obligation surface, not a host cover selector or ambient reflector.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSourceCoverLocality`

Proof. Project the carrier of Definition 7336.1. The only rows that can type a cover-local request are C, T, J, P.

Apply coverage exactness and the source obligation package, Definition 7336.5 and Definition 7336.13. These force the category source, topology window, Grothendieck coverage, and presheaf restriction rows to be displayed before L is read.

The structural rows H, R, Q, N only transport, replay, source, and name that displayed route. They do not add a host cover choice, so every source-cover locality read remains inside the finite package. □

Theorem 7336.18 (Sheafification presheaf locality transport)**.** *For an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet, locality rows in* L *transport along the displayed* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row* P *and the componentwise* hsame *row* H*. After transport, the locality read still lies over the same* C, T, J, P *source surface and still precedes the gluing row* G*.*

Proof. Read P and L from the carrier. The presheaf row records restriction comparisons over the displayed category, topology, and coverage rows, while L records matching-family rows over those restrictions.

Use locality refinement, Definition 7336.6. Refining a displayed cover and transporting component sections by H preserves locality over the same presheaf row rather than creating a new family outside the packet.

Finally apply gluing exactness, Definition 7336.7. The transported locality rows remain before G and S, so the presheaf-locality transport is still a finite BEDC route through P, L, H, R, Q, N. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPresheafLocalityTransport`

Theorem 7336.19 (Sheafification gluing ledger scope)**.** *Every gluing ledger row* G *of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet is consumed only after the locality ledger* L *has displayed compatible local data and only before the* $\mathsf{Sheaf}^{\uparrow}$ *row* S *is read. A gluing consumer therefore sees the route* $C, T, J, P, L, G, S, H, R, Q, N$ *and no detached quotient of compatible families.*

Proof. Start with Definition 7336.13. It places source exposure and locality before any sheaf-facing consumer reads S.

Next apply Definition 7336.7 and Definition 7336.9. They identify G as the ledger row consumed after locality and before the sheaf target row, with replay still occurring through H and R.

The local non-escape theorem, Definition 7336.21, rules out host reflectors and quotient compatible-family objects. Thus G is scoped exactly between L and S inside the displayed carrier. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationGluingLedgerScope`

Theorem 7336.20 (Sheafification local gluing induction)**.** *Finite induction over displayed cover-refinement rows in an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet preserves the route from locality to gluing: the base cover reads* C, T, J, P, L*, and each displayed refinement transports matching-family data by* H *and replays the same* Cont *route* R *before the gluing row* G *and sheaf target* S *are consumed. The induction exports no quotient of compatible families, selected cover system, host reflector, or ambient powerset equality.*

Proof. For the base cover, use Definition 7336.5. The locality ledger is typed only after C, T, J, P have been displayed, so the base matching-family row is already inside the finite carrier.

For a refinement step, apply Definition 7336.6. The displayed $\mathsf{GrothendieckTopology}^{\uparrow}$ refinement row and the transport row H preserve matching-family data over the same presheaf row P.

Finally use Definition 7336.7 together with Definition 7336.9. The gluing ledger G and target row S are read only after the preserved locality route has been replayed by R, so finite induction stays inside L, G, S, H, R, Q, N and never produces an external quotient or choice coordinate. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalGluingInduction`

Theorem 7336.21 (Sheafification local NameCert nonescape)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet*

$$A = (C, T, J, P, L, G, S, H, R, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row* N *cannot export a host reflector, a quotient compatible-family object, selected covers, or a topos-level reflector. Any consumer of* N *receives only the finite* BHist *route through* $C, T, J, P, L, G, S, H, R, Q$ *together with the displayed* hsame, Cont, *and* Pkg *evidence.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalNameCertNonescape`

Proof. Inspect the carrier coordinates. The exposed rows are exactly the category, topology-window, coverage, presheaf, locality, gluing, sheaf, transport, replay, provenance, and local naming rows.

The excluded objects are absent from that tuple. A host reflector, quotient compatible-family object, selected cover system, or topos-level reflector would require an additional coordinate or an additional classifier comparison.

Because N names only the displayed package and because H, R, Q preserve the same route, the local certificate has no escape path to any of those excluded host constructions. □

Theorem 7336.22 (Sheafification separated-plus carrier obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, the separated-refinement row and the plus-carrier row are admitted only through the displayed* $\mathsf{Cat}^{\uparrow}$, $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{Sheaf}^{\uparrow}$ *source rows already present in Definition 7336.1. No separated or plus-facing read may introduce a host reflector, quotient compatible-family object, selected cover system, or topos-level carrier outside those rows.*

Proof. Project the carrier from Definition 7336.1. The only source rows available to a separated or plus-facing read are $C, T, J, P, L, G, S, H, R, Q, N$, with C, J, P, S providing the category, coverage, presheaf, and sheaf-facing endpoints.

Apply the NameCert obligations of Definition 7336.2. They exhaust the local certificate surface by the displayed source, locality, gluing, target, transport, replay, provenance, and naming rows.

The exactness and replay rows then identify the admission boundary. Coverage exactness from Definition 7336.5 forces source exposure before locality is consumed, plus gluing replay from Definition 7336.9 keeps the plus-style route inside L, G, S, H, R, and local non-escape from Definition 7336.21 excludes host reflectors and quotient family objects. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSeparatedPlusCarrierObligation`

Theorem 7336.23 (Sheafification cover-local gluing obligation)**.** *Every cover-local matching-family read and every gluing read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier replays through componentwise* hsame *transport and displayed* Cont *before any sheaf-facing carrier row is consumed. Thus the route from* P *through* L *and* G *to* S *remains the finite BEDC route named by* H, R, Q, N*.*

Proof. Begin with Definition 7336.5. It forces any locality read to expose C, T, J, P before the ledger L can be used.

Next apply Definition 7336.9. A plus-style gluing read must pass through L, G, S, H, R, so the matching-family rows are transported by H and replayed by R before the sheaf-facing carrier row is consumed.

Finally use Definition 7336.2 and Definition 7336.21. The local certificate names only the displayed route, and the non-escape theorem rules out a shortcut through host quotients, selected covers, or topos-level reflectors. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverLocalGluingObligation`

Theorem 7336.24 (Sheafification sheaf handoff ledger obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier, the sheaf handoff ledger records the non-escape boundary from the displayed source rows to the target row S: compatible-family quotients and topos reflectors are excluded, while the only exported handoff is the finite route through Definition 7336.1, Definition 7336.2, Definition 7336.5, Definition 7336.9, and Definition 7336.21.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSheafHandoffLedgerObligation`

Proof. Read the carrier definition first. The handoff ledger has no coordinate for host quotients of compatible families and no coordinate for a topos reflector; it has only the source, locality, gluing, target, transport, replay, provenance, and naming rows.

The NameCert obligations and coverage exactness determine the source side of the ledger. They require category, topology or site, Grothendieck coverage, and presheaf restriction rows before locality and gluing data can be consumed.

The plus gluing replay theorem supplies the target side, routing through L, G, S, H, R. The local non-escape theorem then states that the local certificate cannot export the excluded host constructions, so the handoff ledger records exactly that boundary. □

Theorem 7336.25 (Sheafification plus-localization carrier admission)**.** *Every accepted plus-localization read of a* $\mathsf{Sheafification}^{\uparrow}$ *packet exposes the separated-refinement rows, cover-local matching rows, plus-style gluing rows, componentwise transport row, displayed replay row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row as finite* BHist *data. The exposed rows are exactly the rows admitted by Definition 7336.1 and sourced through the category, topology, Grothendieck topology, presheaf, and sheaf chapters named in the chapter introduction.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusLocalizationCarrierAdmission`

Proof. Project the carrier $A = (C, T, J, P, L, G, S, H, R, Q, N)$ from Definition 7336.1. The separated-refinement and cover-local matching rows are the locality entries in L, and the plus-style gluing rows are the accepted glued-section entries in G.

The source chapters named in the introduction supply the endpoints of this route: category source data, topology windows, Grothendieck coverage, presheaf restrictions, and the sheaf-facing target row. The NameCert obligations of Definition 7336.2 then package these rows with H, R, Q, N.

No host reflector or quotient family is admitted, because no such coordinate occurs in the displayed carrier. □

Theorem 7336.26 (Sheafification separated-refinement transport)**.** *For an accepted* Sheafification$^{\uparrow}$ *carrier, componentwise* hsame *transport preserves separated-refinement rows and cover-local matching rows before the sheaf handoff is consumed. Thus a transported read still reaches the sheaf-facing row only through the displayed route from C, T, J, P through L and G to S.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSeparatedRefinementTransport`

Proof. Start with a separated-refinement row in L. By Definition 7336.8, that row is indexed by the displayed category, topology-window, Grothendieck-coverage, and presheaf-restriction data C, T, J, P.

Apply the transport row H. It records componentwise hsame transport for the source rows, locality rows, gluing rows, sheaf row, replay row, provenance row, and local naming row, so the transported separated-refinement read remains over the same displayed cover-local matching data.

Finally use Definition 7336.4. The sheaf-facing row is downstream of L and G, hence transport preserves the pre-handoff boundary rather than exposing a separate sheafification object. □

Theorem 7336.27 (Sheafification cover-local gluing replay)**.** *Every displayed* Cont *replay of an accepted* Sheafification$^{\uparrow}$ *packet carries cover-local matching rows through plus-style gluing into the sheaf-facing carrier. The replay route is the finite path through P, L, G, S, H, R, Q, N and exports no quotient of compatible families, selected cover system, or host sheafification functor.*

Proof. Read the cover-local matching data through P and L. Coverage exactness from Definition 7336.5 supplies the displayed cover and restriction rows before the locality ledger is consumed.

Next apply Definition 7336.9. The plus-style gluing step passes through L, G, S, H, R, so the displayed Cont row R carries the matching rows into the sheaf-facing carrier only after the gluing ledger has been read.

The provenance and local naming rows Q, N record the same route. Since Definition 7336.21 excludes host quotients, selected covers, and topos-level reflectors, the replay remains inside the finite BEDC carrier. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverLocalGluingReplay`

Theorem 7336.28 (Sheafification site-plus boundary obligations)**.** *Every accepted* Sheafification$^{\uparrow}$ *carrier records the site-to-plus boundary as finite* BHist *rows: the displayed site rows C, T, J, presheaf restriction row P, separated-plus locality row L, gluing row G, sheaf handoff row S, componentwise* hsame *transport H, displayed* Cont *replay R,* Pkg *provenance Q, and local* NameCert *row N. A plus consumer must read the site and presheaf rows before it may consume separated-plus, locality, or gluing rows.*

Proof. Project the carrier tuple $A = (C, T, J, P, L, G, S, H, R, Q, N)$ from Definition 7336.1. The displayed site is the finite source surface C, T, J, and the presheaf row is P.

Read the site row before plus construction. The coverage and presheaf restriction obligations force every plus request to expose C, T, J, P before any separated-refinement row in L is available.

Use componentwise transport and replay. The rows H and R preserve this order under hsame transport and displayed Cont replay, while Q, N record provenance and local naming for exactly the same finite route. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSitePlusBoundaryObligations`

Theorem 7336.29 (Sheafification plus separation obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *read, separatedness is consumed as a displayed plus row before any gluing handoff is exported. The finite route is*

$$C, T, J, P, L \longmapsto G \longmapsto S, H, R, Q, N,$$

where C, T, J *are site rows,* P *is the presheaf row,* L *is the separated-plus locality row,* G *is the gluing row,* S *is the sheaf-facing target row, and* H, R, Q, N *are the* hsame*,* Cont*,* Pkg*, and* NameCert *structural rows.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusSeparationObligation`

Proof. First project the displayed carrier. The separated-plus information occurs in L, not in a quotient of compatible families and not in a host reflector.

Read separatedness before gluing. By Definition 7336.8 and Definition 7336.26, a consumer reaches G only after L has already been read over the site and presheaf rows.

Finally use the sheaf handoff ledger. The rows S, H, R, Q, N transport, replay, record, and name the post-gluing handoff, so stability is componentwise and no separate sheafification object is introduced. □

Theorem 7336.30 (Sheafification locality gluing obligation)**.** *Every locality-to-gluing read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet is anchored in finite BEDC data:* BHist *site rows,* hsame *transport,* Cont *replay,* Pkg *provenance, the local* NameCert *row, and the displayed site, presheaf, separated-plus, locality, and gluing rows of the carrier. Gluing stability is therefore a componentwise replay property of* $C, T, J, P, L, G, S, H, R, Q, N$*, not a quotient-family or topos-reflection principle.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalityGluingRouteCertificate`

Proof. Start with the locality row L. It is attached to the displayed category, topology or Grothendieck coverage, and presheaf rows C, T, J, P, so local matching has no detached cover selector.

Then read the gluing row G. The plus gluing replay theorem routes the local matching data through G and into the sheaf-facing row S only after the separated-plus boundary has been exposed.

Transport and replay finish the stability claim. Componentwise hsame transport H, displayed Cont replay R, Pkg provenance Q, and local NameCert row N preserve the same finite route for downstream consumers. □

Theorem 7336.31 (Sheafification site source exposure)**.** *Every plus, locality, or gluing read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier must expose the* $\mathsf{Cat}^{\uparrow}$ *row* C*, the* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{GrothendieckTopology}^{\uparrow}$ *cover rows* T, J*, and the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row* P *before any locality ledger* L*, gluing ledger* G*, or sheaf target row* S *is consumed. The source exposure is therefore the finite route through the displayed category, topology or site, Grothendieck coverage, and presheaf rows, not a host reflector or quotient compatible-family object.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSiteSourceExposure`

Proof. Project the carrier of Definition 7336.1. Its plus and locality data occur only in L and G, and those rows are typed by the displayed source rows C, T, J, P.

Apply the coverage exactness theorem Definition 7336.5. It already forces every locality read to expose the category, topology-window, Grothendieck-coverage, and presheaf-restriction rows before the locality ledger can be consumed.

The separated-refinement and plus-carrier obligations preserve the same order. They admit no host reflector and no quotient compatible-family coordinate, so every plus, locality, or gluing read is sourced by the displayed site and presheaf rows. □

Theorem 7336.32 (Sheafification separated-plus site route)**.** *Every accepted* $\mathsf{Sheafification}^{\uparrow}$ *separated-plus read exposes the finite site route*

$$C, T, J, P, L, G, S, H, R, Q, N$$

in that order: $\mathsf{Cat}^{\uparrow}$ *source row* C*,* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{GrothendieckTopology}^{\uparrow}$ *cover rows* T, J*,* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row* P*, separated-plus locality row* L*, gluing row* G*,* $\mathsf{Sheaf}^{\uparrow}$ *handoff row* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* R*,* Pkg *provenance* Q*, and local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row* N*. No separated-plus consumer may read a gluing or sheaf-facing row before the site, presheaf, locality, transport, replay, provenance, and naming rows are displayed.*

Proof. Project the carrier of Definition 7336.1. Its displayed rows already place the category, topology or Grothendieck coverage, presheaf restriction, locality, gluing, sheaf target, transport, replay, provenance, and local naming data in the finite tuple $C, T, J, P, L, G, S, H, R, Q, N$.

Read source exposure through Definition 7336.31. It forces every plus, locality, or gluing read to display C, T, J, P before L, G, S are consumed.

Apply the separated-refinement and plus-gluing transport rows. They keep L before G, keep G before the sheaf-facing handoff S, and keep H, R, Q, N attached to the same displayed route rather than to a host reflector or quotient compatible-family object. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSeparatedPlusSiteRoute`

Theorem 7336.33 (Sheafification Grothendieck locality induction)**.** *Every finite chain of displayed Grothendieck cover refinements read by an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier is handled by induction over the visible refinement rows. The base case reads the source site* C, T, J *and presheaf row* P*; each successor step replays locality refinement through* L*, plus gluing through* G*, componentwise transport through* H*, displayed continuation through* R*, provenance through* Q*, and local naming through* N *before the* $\mathsf{Sheaf}^{\uparrow}$ *handoff row* S *is consumed.*

Proof. Use induction on the finite list of displayed cover-refinement rows in the $\mathsf{GrothendieckTopology}^{\uparrow}$ component J. The empty list is exactly the source exposure route of Definition 7336.31, so the carrier has only C, T, J, P available before locality is read.

For the successor step, assume the previous refinement segment has already been replayed through L, H, R, Q, N. The locality-refinement exactness row supplies the next cover-local read over the same site and presheaf source, and the plus gluing replay row transports that read into G without selecting a host cover system.

The handoff is then the existing sheaf-facing route. The row S is consumed only after the successor locality and gluing rows have been replayed through H and R and recorded by Q, N, so the induction remains finite and internal to the displayed BEDC carrier. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationGrothendieckLocalityInduction`

Theorem 7336.34 (Sheafification presheaf-sheaf route)**.** *The presheaf-to-sheaf handoff of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet is the finite BEDC route*

$$P, L, G, S, H, R, Q, N.$$

Here P is the displayed $\mathsf{PreSheaf}^{\uparrow}$ *restriction row, L is separated-plus locality, G is plus gluing, S is the* $\mathsf{Sheaf}^{\uparrow}$ *target row, H is componentwise* hsame *transport, R is displayed* Cont *replay, Q is* Pkg *provenance, and N is local* NameCert *data. The route excludes host sheafification reflectors, quotient compatible-family objects, selected cover systems, and topos-level carriers.*

Proof. Start from the presheaf row P in the carrier. By source exposure, P is read only together with the displayed site rows C, T, J, so the handoff is not sourced by an ambient presheaf category or a hidden reflector.

Move from P to L and G. The separated-plus and cover-local gluing obligations make locality and gluing into displayed carrier rows, while the plus gluing transport theorem preserves the same route through componentwise transport and replay.

Finally consume S, H, R, Q, N. These rows provide the sheaf-facing target, transport, replay, provenance, and local naming surface for the same packet, so no quotient compatible-family object, selected cover system, or topos-level carrier is exported. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPresheafSheafRoute`

Theorem 7336.35 (Sheafification plus gluing transport)**.** *Every plus-style gluing read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier transports cover-local matching data through componentwise* hsame *and replays it through displayed* Cont *before the* $\mathsf{Sheaf}^{\uparrow}$ *handoff is consumed. The route from P through L and G to S is therefore preserved by H, R, Q, N as finite BEDC data.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusGluingTransport`

Proof. Start with the plus gluing replay theorem Definition 7336.9. It places every plus-style gluing read through the locality ledger L, gluing ledger G, sheaf row S, transport row H, and replay row R.

Use the cover-local gluing obligation Definition 7336.23. It says that matching-family and gluing reads replay through componentwise hsame transport and displayed Cont before a sheaf-facing carrier row is consumed.

Finally apply the separated-refinement transport theorem and the local nonescape theorem. They keep the transported read over the same displayed source rows and exclude quotient families, selected covers, host reflectors, and topos-level reflectors. □

Theorem 7336.36 (Sheafification obligation closure package)**.** *The carrier of Definition 7336.1, the source exposure of Definition 7336.31, the plus gluing transport of Definition 7336.35, the* $\mathsf{Sheaf}^{\uparrow}$ *handoff of Definition 7336.4, the* Pkg *provenance row Q, and the local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row N assemble the obligation package for* $\mathsf{Sheafification}^{\uparrow}$*. The package consists exactly of the displayed category, topology or site, Grothendieck coverage, presheaf restriction, locality, gluing, sheaf target, componentwise transport, replay, provenance, and local naming rows.*

Proof. First assemble the source side. The carrier definition lists $C, T, J, P, L, G, S, H, R, Q, N$, and the site source exposure theorem forces every plus, locality, and gluing read through C, T, J, P before L, G, S are consumed.

Next assemble the transport and replay side. The plus gluing transport theorem preserves cover-local matching and gluing data through H and R, while the sheaf handoff theorem identifies S as the only sheaf-facing row exported by the packet.

Finally use the package and naming rows. The provenance row Q records the sibling sources, and N names the same finite route as the local NameCert surface. Since no host reflector, quotient compatible-family object, selected cover system, topos object, or ambient powerset equality occurs in the carrier, the obligation package is exactly the displayed BEDC route. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationObligationClosurePackage`

Theorem 7336.37 (Sheafification cover-basis stability)**.** *Every accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet exposes its* $\mathsf{Cat}^{\uparrow}$ *source row* C*, its* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{GrothendieckTopology}^{\uparrow}$ *cover rows* T, J*, and its* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row* P *before any locality row in* L *may be consumed. Thus the cover-basis obligation is the finite source route*

$$C, T, J, P \longmapsto L$$

inside the carrier of Definition 7336.1, not a host cover selector or reflector.

Proof. Project the carrier from Definition 7336.1. The only rows that can type a cover-local request are C, T, J, P: category source data, topological or site windows, Grothendieck coverage, and presheaf restrictions.

Apply the site source exposure theorem Definition 7336.31. It already forces every plus, locality, or gluing read to display those source rows before L, G, S are consumed.

The obligation closure package Definition 7336.36 then records the same order as part of the NameCert route. Since the carrier has no host cover selector or reflector coordinate, cover-basis stability is exactly the displayed source-before-locality route. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverBasisStability`

Theorem 7336.38 (Sheafification plus-locality gluing ledger)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, the locality ledger* L *and the gluing ledger* G *form the finite plus-locality ledger that must be displayed before the* $\mathsf{Sheaf}^{\uparrow}$ *target row* S *can be consumed. No sheaf-facing consumer may read* S *from* P *without first exposing both* L *and* G *over the same source rows* C, T, J, P*.*

Proof. Start with the gluing exactness theorem Definition 7336.7. It places the sheaf target row S downstream of the locality ledger L and the gluing ledger G.

Use the plus gluing replay and plus gluing transport theorems, Definition 7336.9 and Definition 7336.35. They keep the plus-style route inside L, G, S, H, R and preserve the same cover-local matching data under componentwise transport.

Finally use the obligation closure package. It identifies the finite package $C, T, J, P, L, G, S, H, R, Q, N$ as the complete route, so the ledger L, G is not replaceable by a quotient of compatible families, host reflector, selected cover system, or topos-level carrier. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusLocalityGluingLedger`

Theorem 7336.39 (Sheafification sheaf target transport)**.** *Let* $A = (C, T, J, P, L, G, S, H, R, Q, N)$ *be an accepted* $\mathsf{Sheafification}^{\uparrow}$ *packet. Componentwise* hsame *transport through* H *and displayed* Cont *replay through* R *preserve the same source-to-sheaf route*

$$C, T, J, P, L, G, S$$

under the provenance row Q *and local naming row* N*. The transported read cannot change the source site, presheaf restriction, locality ledger, gluing ledger, or sheaf target into a host reflector route.*

Proof. Read the transported rows from Definition 7336.1. The transport row H is componentwise: it relates the displayed category, topology or site, presheaf, locality, gluing, sheaf, replay, provenance, and local naming coordinates without adding another coordinate.

Apply the separated-refinement transport theorem Definition 7336.26 and the cover-local gluing replay theorem Definition 7336.27. These theorems keep cover-local matching and plus-style gluing on the same route from P through L and G to S while replaying it through R.

The package rows Q, N then source and name that transported route. By local nonescape, Definition 7336.21, no host reflector, quotient compatible-family object, selected cover system, or topos-level carrier is exported. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSheafTargetTransport`

Theorem 7336.40 (Sheafification restriction compatibility obligation)**.** *Every restriction-compatibility read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$ *keeps each plus and locality row downstream of the displayed* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{GrothendieckTopology}^{\uparrow}$*, and* $\mathsf{PreSheaf}^{\uparrow}$ *source rows. In particular, no read may consume* L*,* G*, or* S *before* C, T, J, P *have exposed the category source, topology or site windows, Grothendieck coverage, and presheaf restriction comparisons.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationRestrictionCompatibilityObligation`

Proof. Project the carrier of Definition 7336.1. The only restriction source rows are C, T, J, P, while L, G, S are locality, gluing, and sheaf-target rows downstream of that source.

Apply coverage exactness and locality refinement, Definition 7336.5 and Definition 7336.6. Coverage exactness forces every locality read through the category, topology or site, coverage, and presheaf rows; locality refinement keeps the same presheaf restriction data when displayed cover refinements are replayed.

Finally use gluing exactness Definition 7336.7. Since S is consumed only after L and G over the same P, restriction compatibility remains downstream of C, T, J, P and cannot become an independent reflector route. □

Theorem 7336.41 (Sheafification source-locality obligation)**.** *Every source-locality read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$ *must expose* C, T, J, P *before any locality row in* L *is consumed. Thus locality is read only after the displayed category source, topology or site window, Grothendieck coverage row, and presheaf restriction comparisons are visible in the same finite packet.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationSourceLocalityObligation`

Proof. Project the carrier of Definition 7336.1. The rows C, T, J, P are the only source rows for category objects, open or site windows, covering data, and presheaf restrictions.

The locality ledger L is indexed by those displayed source rows. A read of L without C, T, J, P would have no BEDC cover row and no presheaf restriction comparison from which locality could be interpreted.

The structural rows preserve the order. Componentwise transport through H, replay through R, provenance through Q, and local naming through N carry the displayed source-locality route, but none of them supplies an independent source coordinate. □

Theorem 7336.42 (Sheafification gluing-target obligation)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, the sheaf target row* S *is reachable only through the locality ledger* L *and gluing ledger* G *over the same displayed cover rows* C, T, J *and presheaf row* P*. No consumer may read* S *as a detached sheafification reflector, quotient family, or selected compatible-family representative.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationGluingTargetObligation`

Proof. Begin with the source rows C, T, J, P. They display the category, topology or site windows, Grothendieck coverage, and presheaf restrictions on which the gluing request is based.

The locality row L records matching-family data over that source, and the gluing row G records the accepted glued section. Therefore S is a target row only after the same cover and presheaf rows have passed through L and G.

Use transport and replay through H and R. These rows preserve the displayed route to S, while Q, N source and name it. Since the carrier has no reflector, quotient-family, or selected-representative coordinate, the gluing target cannot be read through any other path. □

Theorem 7336.43 (Sheafification provenance-naming obligation)**.** *In an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, the provenance row* Q *and local naming row* N *name only the displayed route*

$$C, T, J, P, L, G, S, H, R.$$

They cannot add a host reflector, quotient-family datum, selected cover system, topos object, ambient powerset equality, or separate sheafification functor.

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationProvenanceNamingObligation`

Proof. Read Q from the carrier. Its provenance is over the displayed $\mathsf{Sheafification}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, BHist, hsame, Cont, and $\mathsf{NameCert}$ rows.

Read N next. The local $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ row names the same route from source data through locality and gluing to the sheaf target; it has no separate field for quotient families or host reflectors.

Finally apply the nonescape theorems already present in the chapter. The plus, separated-refinement, and cover-gluing boundaries keep every read inside $C, T, J, P, L, G, S, H, R, Q, N$, so provenance and naming cannot enlarge the packet beyond the displayed route. □

Theorem 7336.44 (Sheafification root-obligation carrier)**.** *Every root-obligation read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$ *must display all eleven carrier rows before the local* $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$ *row is consumed. The sibling route is ordered through* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{GrothendieckTopology}^{\uparrow}$*,* $\mathsf{PreSheaf}^{\uparrow}$*, and* $\mathsf{Sheaf}^{\uparrow}$*, and the BEDC primitives visible along the route are* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Project the carrier of Definition 7336.1. The rows $C, T, J, P, L, G, S, H, R, Q, N$ are the only coordinates available to a root read.

Read the sibling route in order. Category data in C precedes topological windows in T, coverage in J, presheaf restriction data in P, locality and gluing ledgers in L, G, and the sheaf target in S.

The remaining rows are structural projections. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row record the same ordered carrier and do not transport parameters from an outside reflector. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationRootObligationCarrier`

Theorem 7336.45 (Sheafification kernel-scope package)**.** *For every accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, the kernel-scope package exposed to downstream category or sheaf consumers is exactly*

$$C, T, J, P, L, G, S, H, R, Q, N.$$

The package names the BHist *rows for the sibling route through* Cat$^{\uparrow}$, Topology$^{\uparrow}$, GrothendieckTopology$^{\uparrow}$, PreSheaf$^{\uparrow}$, *and* Sheaf$^{\uparrow}$, *the* hsame *and* Cont *replay rows, the* Pkg *source row, and the local* NameCert *row; it exports no detached reflector or quotient-family carrier.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationKernelScopePackage`

Proof. Start with the displayed package. Each coordinate is a carrier projection from Definition 7336.1, so the kernel scope is finite before any consumer reads it.

The route through the siblings is row-ordered rather than parameter-transported. The presheaf row P is read only after C, T, J, the locality and gluing rows L, G are read only after P, and the sheaf row S is read only after the locality and gluing ledgers.

The structural rows H, R, Q, N then package the same route. Since they are hsame, Cont, Pkg, and NameCert projections over the displayed BHist coordinates, they cannot introduce a separate sheafification object. □

Theorem 7336.46 (Sheafification cover-site gluing ledger)**.** *For every accepted* Sheafification$^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, a cover-site gluing read must expose* $C, T, J, P, L, G, S, H, R, Q, N$ *in the displayed carrier order. The category row* C*, topology row* T*, Grothendieck coverage row* J*, presheaf row* P*, locality row* L*, gluing row* G*, sheaf target row* S*,* hsame *transport* H*,* Cont *replay* R*,* Pkg *provenance* Q*, and local* NameCert *row* N *form the ledger; no site consumer may read* G *or* S *before this source route is visible.*

Proof. Apply Definition 7336.5 to force the source rows C, T, J, P before locality is consumed. Then apply Definition 7336.7 and Definition 7336.9; together they place L before G, G before S, and keep replay inside H, R.

The package rows Q, N name the same route by Definition 7336.2. Hence the cover-site ledger is the displayed finite carrier route rather than a host site object or quotient of compatible families. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverSiteGluingLedger`

Theorem 7336.47 (Sheafification refinement stability budget)**.** *Every finite refinement budget used by an accepted* Sheafification$^{\uparrow}$ *carrier is spent inside* $C, T, J, P, L, G, S, H, R, Q, N$*: refinements are read from* J*, presheaf compatibility from* P*, locality stability from* L*, gluing from* G*, transport from* H*, replay from* R*, provenance from* Q*, and naming from* N *before the sheaf target* S *is consumed.*

Proof. Use Definition 7336.6 for each displayed refinement row. It keeps matching-family data over the same presheaf row P and uses componentwise transport H rather than an external cover selector.

Now apply Definition 7336.51. Finite induction preserves the same L, G, H, R, Q, N structure at every refinement step, and Definition 7336.7 keeps S downstream of locality and gluing. Thus the budget is exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationRefinementStabilityBudget`

Theorem 7336.48 (Sheafification target nonescape ledger)**.** *For an accepted* Sheafification$^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, every target-facing read of* S *remains inside the cover, presheaf, locality, gluing, transport, replay, provenance, and local naming rows of that tuple. It exports no host sheafification functor, quotient compatible-family object, selected cover system, topos reflector, or ambient powerset equality.*

Proof. First use Definition 7336.16. The final handoff to a $\mathsf{Sheaf}^{\uparrow}$ consumer factors through $C, T, J, P, L, G, S, H, R, Q, N$.

Then apply Definition 7336.21 and Definition 7336.50. These nonescape boundaries exclude host reflectors, quotient compatible-family data, selected covers, topos objects, and ambient powerset equality because none of them is a carrier coordinate. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationTargetNonescapeLedger`

Theorem 7336.49 (Sheafification locality-gluing ledger readiness)**.** *In an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier* $A = (C, T, J, P, L, G, S, H, R, Q, N)$*, a locality or gluing consumer may read* L *or* G *only after the source route* C, T, J, P *has been displayed and before the sheaf target row* S *is consumed. Thus the ledger is ready exactly when the* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{GrothendieckTopology}^{\uparrow}$*, and* $\mathsf{PreSheaf}^{\uparrow}$ *sibling rows are already visible, and its readback to* $\mathsf{Sheaf}^{\uparrow}$ *remains inside the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet.*

Proof. Inspect the row order. The source rows C, T, J, P provide the category, topology, coverage, and presheaf restriction data on which any locality or gluing ledger entry depends.

The locality row L and gluing row G are therefore ready only after that source route is displayed. Their output is the sheaf target row S, so a consumer cannot read S before the same ledger has been exposed.

Transport, replay, provenance, and naming preserve the ledger. Rows H, R, Q, N record hsame, Cont, Pkg, and NameCert data for the same finite packet, so ledger readiness is a projection and row-order claim rather than parameter transport. □

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalityGluingLedgerReadiness`

Theorem 7336.50 (Sheafification plus nonescape)**.** *Every plus-style read of an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier remains inside the finite route*

$$C, T, J, P, L, G, S, H, R, Q, N.$$

It exports no quotient compatible-family data, selected cover system, host sheafification reflector, topos reflection object, ambient powerset equality, or cover choice principle outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationPlusNonescape`

Proof. Read the carrier in Definition 7336.1. The plus and locality data occur only as the locality ledger L, gluing ledger G, sheaf target S, and structural rows H, R, Q, N over the source C, T, J, P.

Use coverage exactness and gluing exactness, Definition 7336.5 and Definition 7336.7. They force plus-style gluing to consume source coverage and presheaf restrictions before any sheaf-facing row is read.

The excluded objects are absent from the carrier. Componentwise transport, displayed replay, provenance, and local naming preserve the finite route, so they cannot create quotient compatible-family data, selected covers, host reflectors, or topos reflection objects. □

Theorem 7336.51 (Sheafification cover-refinement finite induction)**.** *For every finite displayed chain of cover refinements in an accepted* $\mathsf{Sheafification}^{\uparrow}$ *carrier, induction over the chain preserves locality rows in* L *and gluing rows in* G *through componentwise* hsame *transport* H *and displayed* Cont *replay* R*. The resulting route remains the same finite sheafification packet* $C, T, J, P, L, G, S, H, R, Q, N$ *and never leaves the displayed coverage, presheaf, locality, gluing, provenance, and local naming surface.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverRefinementFiniteInduction`

Proof. For the base refinement, apply coverage exactness Definition 7336.5. The source rows C, T, J, P are displayed before any locality row in L is available.

For the successor step, use locality refinement Definition 7336.6. The next displayed cover refinement preserves matching-family data over the same presheaf row P and transports it componentwise through H.

Finally apply gluing exactness Definition 7336.7. The gluing row G and sheaf target S are consumed only after the preserved locality data are replayed through R and recorded by Q, N, so the induction stays inside the finite carrier. □

Theorem 7336.52 (Sheafification cover-refinement budget)**.** *Every accepted* Sheafification$^{\uparrow}$ *carrier exposes cover rows, plus-locality rows, separated refinement rows, gluing rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *before any sheaf target consumes the refinement budget. The budget is therefore a finite row obligation over* $C, T, J, P, L, G, S, H, R, Q, N$*, not a host quotient of compatible families, selected cover system, ambient sheafification reflector, topos reflection object, or cover-choice principle.*

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationCoverRefinementBudget`

Proof. Start with the source obligation package in Definition 7336.2. It exposes the source category, topology or Grothendieck topology, presheaf row, locality rows, gluing rows, sheaf target, transport, replay, provenance, and local naming as one finite packet.

Apply cover basis stability and the plus-locality and gluing ledger results already established in the chapter. Coverage exactness keeps every refinement request inside the displayed cover basis, while locality refinement and gluing exactness force the plus and gluing reads to remain attached to the same presheaf route.

Finally use cover-refinement finite induction, Definition 7336.51. Since each refinement step preserves the same L, G, H, R, Q, N structure, a downstream sheaf consumer can spend only the displayed refinement budget and cannot import a host reflector, quotient-family construction, selected covers, or topos object. □

Falsifiable prediction: Every accepted Sheafification$^{\uparrow}$ read displays CategoryUp, TopologyUp or GrothendieckTopologyUp, PreSheafUp, locality, gluing, SheafUp, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any sheaf-facing consumer can use the target row.

Independence witness: category, topology, grothendieck topology, presheaf, sheaf: Sheafification$^{\uparrow}$ is not bijective to any listed sibling because it binds locality refinement and gluing handoff rows between a displayed presheaf source and a displayed sheaf target.

Theorem 7336.53 (Sheafification locality scope addendum)**.** *The presheaf restriction rows, cover-refinement rows, locality witnesses, gluing witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows of an accepted* Sheafification$^{\uparrow}$ *carrier form one scoped BEDC route*

$$C, T, J, P, L, G, S, H, R, Q, N.$$

Thus locality and gluing are consumed only through the displayed finite packet, not through a host reflector, quotient compatible-family object, selected cover system, or ambient topos construction.

Lean4 ✓ `BEDC.Derived.SheafificationUp.SheafificationLocalityScopeAddendum`

Proof. Read the carrier of Definition 7336.1. The rows C, T, J, P expose the category, cover, topology, and presheaf restriction data before locality or gluing can be used.

Apply cover-refinement finite induction, Definition 7336.51, and the cover-refinement budget of Definition 7336.52. They keep locality rows L and gluing rows G attached to the same finite refinement route.

The structural rows H, R, Q, N transport, replay, source, and name that route. Since the refused host objects are absent from the carrier, every accepted locality or gluing read remains inside the displayed BEDC packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}$.

Constructive story: A $\mathsf{Sheafification}^{\uparrow}$ packet is generated from CategoryUp source rows, TopologyUp or Grothendieck-TopologyUp cover rows, a PreSheafUp restriction row, locality-refinement rows, gluing rows, a SheafUp target row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 7336.1, Definition 7336.2, Definition 7336.4, and Definition 7336.52 bind the finite sheafification route from presheaf restrictions through locality, gluing, transport, replay, provenance, and local naming obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Sheafification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host sheafification functor, quotient of compatible families, global choice of covers, topos objects, ambient powerset equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source exposure, coverage exactness, presheaf restriction exposure, locality refinement, gluing exactness, sheaf handoff, transport, replay, provenance, and local naming for BEDC.Derived.SheafificationUp.

Chapter 7337

A Concrete Naming Certificate for $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$

Following Doob and Bishop-style finite upcrossing arguments, the $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ packet records the BEDC route by which a finite upcrossing count, a bounded stopping window, and rational tolerance rows become convergence-preparation data for $\mathsf{MartingaleConvergence}^{\uparrow}$. It sits after the probability, random-variable, martingale, upcrossing, Doob-upcrossing, bounded-variation, and convergence surfaces of chapter 184, chapter 185, chapter 190, chapter 1629, chapter 1630, chapter 398, and chapter 2246. The packet is finite evidence for the convergence route; it is not an almost-sure quotient, a selected exceptional set, a completed filtration, or an ambient limit theorem.

Definition 7337.1 (Martingale convergence finite-upcrossing carrier)**.** *A* $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (\Omega, X, M, U, D, S, V, R, T, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *row for the inspected probability context.*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint-family row for the finite martingale window.*
3. M *is the* $\mathsf{Martingale}^{\uparrow}$ *finite adapted-sequence source.*
4. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *completed-crossing count ledger for displayed lower and upper rational boundary rows.*
5. D *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *finite inequality row that bounds* U*.*
6. S *is the bounded stopping-window row recording the inspected horizon and the lower-to-upper hit windows.*
7. V *is the* $\mathsf{BoundedVariation}^{\uparrow}$ *or finite-variation ledger controlling the rational tolerance rows read by the convergence consumer.*
8. R *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the displayed lower and upper windows.*
9. T *is the* $\mathsf{MartingaleConvergence}^{\uparrow}$ *route row that may consume only* U, D, S, V, R *after the martingale source rows have been displayed.*
10. H *records componentwise* hsame *transport for all displayed rows.*

11. *C records displayed* Cont *replay through the martingale, upcrossing, stopping-window, variation, tolerance, and convergence-route rows.*
12. *P is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MartingaleConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows and their componentwise transports.

Theorem 7337.2 (Martingale convergence finite-upcrossing NameCert obligations)**.** *For every accepted carrier*

$$F = (\Omega, X, M, U, D, S, V, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}}$ *surface consists exactly of probability exposure, endpoint exposure, martingale-source exposure, finite upcrossing-count exposure, Doob-bound exposure, bounded stopping-window exposure, bounded-variation or tolerance-ledger exposure, dyadic rational tolerance exposure, convergence-route admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7337.1. The stochastic source rows are Ω, X, M, while the convergence-facing finite evidence is U, D, S, V, R, T.

Read the upcrossing side first. The row U supplies a finite completed-count ledger, D supplies the Doob bound for that ledger, and S records the bounded stopping windows that make the count inspectable.

The tolerance side is also finite. The rows V and R display the bounded variation and dyadic tolerance evidence before the convergence-route row T may be consumed.

The structural rows H, C, P, N transport, replay, source, and name exactly that packet. Since no row stores an almost-sure quotient, selected exceptional set, completed filtration, unbounded stopping family, selected limit, or measure completion, the listed obligations exhaust the local certificate. □

Theorem 7337.3 (Finite upcrossing route to martingale convergence)**.** *Every* $\mathsf{MartingaleConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ *packet must first read the* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, *and* $\mathsf{Martingale}^{\uparrow}$ *source rows, the* $\mathsf{Upcrossing}^{\uparrow}$ *completed-count ledger, the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *bound, the bounded stopping-window row, the* $\mathsf{BoundedVariation}^{\uparrow}$ *or tolerance ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational tolerance row, and the structural* hsame/Cont/Pkg/NameCert *rows. The route exports finite convergence-preparation data only.*

Proof. Start from the martingale source. The rows Ω, X, M display the probability context, endpoint family, and adapted finite window before any crossing count is read.

Apply the upcrossing and Doob-upcrossing routes. The finite count U is read through the displayed lower and upper boundary rows, while the Doob row D records the finite inequality surface that bounds that count.

Now inspect the convergence-facing rows. The bounded stopping window S and the tolerance rows V, R are the only data passed to T, so the downstream $\mathsf{MartingaleConvergence}^{\uparrow}$ packet receives a finite ledger rather than a limiting event.

Transport and replay are componentwise through H and C, with P, N recording provenance and local naming. Therefore the route cannot export an almost-sure quotient, selected exceptional set,

completed filtration, unbounded stopping principle, selected limit, or completed measure carrier. □

Theorem 7337.4 (Martingale convergence finite-upcrossing non-escape)**.** *Every accepted* MartingaleConvergenceFi *packet exports only the probability row, endpoint row, finite martingale row, upcrossing-count row, Doob-bound row, bounded stopping-window row, bounded-variation or tolerance-ledger row, dyadic tolerance row, convergence-route row, componentwise transport, displayed replay, provenance, and local naming rows. It exports no almost-sure quotient, selected exceptional set, completed filtration, unbounded stopping principle, selected limit, measure completion, or ambient convergence theorem.*

Proof. Project the carrier

$$F = (\Omega, X, M, U, D, S, V, R, T, H, C, P, N)$$

from Definition 7337.1. The displayed coordinates are exactly the stochastic source rows, finite upcrossing evidence, tolerance rows, convergence route, and structural rows named in the statement.

Invoke Definition 7337.2. The local NameCert surface admits only probability exposure, endpoint exposure, finite martingale exposure, upcrossing count, Doob bound, bounded stopping window, bounded-variation or tolerance ledger, dyadic tolerance, convergence-route admission, transport, replay, provenance, and naming.

Apply Definition 7337.3. Any $\mathsf{MartingaleConvergence}^{\uparrow}$ consumer receives finite convergence-preparation data only after those rows are displayed.

The refused objects have no carrier coordinate and no structural row. Row exhaustion therefore blocks an almost-sure quotient, selected exceptional set, completed filtration, unbounded stopping principle, selected limit, measure completion, and ambient convergence theorem from escaping the packet. □

Falsifiable prediction: A claimed $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ read that reaches MartingaleConvergenc without displaying the $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, bounded stopping-window, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 7337.1.

Independence witness: probspace, randomvar, martingale, upcrossing, doob-upcrossing, martingale-convergence: no listed sibling is bijective to $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ because this packet alone binds the finite crossing-count ledger, bounded stopping window, Doob-bound row, variation and dyadic tolerance rows, convergence-route admission, transport, replay, provenance, and local naming as one convergence-preparation carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}}$.

Constructive story: A $\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}$ packet is generated from probability, random-variable, finite martingale, upcrossing-count, Doob-bound, bounded stopping-window, bounded-variation, dyadic tolerance, convergence-route, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7337.1, Definition 7337.2, and Definition 7337.3 seed the finite upcrossing route into martingale convergence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MartingaleConvergenceFiniteUpcrossing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure quotient equality, selected exceptional sets, completed filtrations, unbounded stopping, selected limits, measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite upcrossing count, Doob-bound, stopping-window, tolerance-ledger, convergence-route, transport, replay, provenance, and local naming rows for BEDC.Derived.MartingaleConvergenceFi

Chapter 7338

A Concrete Naming Certificate for $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$

Following Doob, the $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ packet extracts the finite witness surface by which upcrossing control is consumed by $\mathsf{MartingaleConvergence}^{\uparrow}$ without importing the whole convergence-preparation packet. It sits after the martingale convergence route of chapter 2246, the upcrossing ledger of chapter 1629, the Doob bound of chapter 1630, the bounded stopping route of chapter 383, the finite schedule of chapter 126, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet is a finite extraction boundary for later martingale convergence consumers; it is not an almost-sure quotient, a selected exceptional set, a completed filtration, a measure-completion object, or an ambient convergence theorem.

Definition 7338.1 (Martingale convergence witness carrier)**.** *A* $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, T, U, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{Martingale}^{\uparrow}$ *finite adapted-source row over the inspected probability and endpoint data.*
2. T *is the bounded stopping-window row consumed through* $\mathsf{StoppingTime}^{\uparrow}$ *and* $\mathsf{OptionalStopping}^{\uparrow}$*.*
3. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *completed-crossing count for the displayed lower and upper boundary rows.*
4. D *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *inequality row bounding the finite count* U*.*
5. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row naming the requested observation windows.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and rational budgets.*
7. L *is the* $\mathsf{Real}^{\uparrow}$ *lower and upper boundary row read by the upcrossing and convergence consumers.*
8. H *records componentwise* hsame *transport for the martingale, stopping, upcrossing, Doob-bound, schedule, readback, Real-boundary, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay through the same finite route.*
10. P *is the* Pkg *provenance over* $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$*,* $\mathsf{MartingaleConvergence}^{\uparrow}$*,* $\mathsf{Martingale}^{\uparrow}$*,* $\mathsf{StoppingTime}^{\uparrow}$*,* $\mathsf{OptionalStopping}^{\uparrow}$*,* $\mathsf{Upcrossing}^{\uparrow}$*,* $\mathsf{DoobUpcrossing}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceWitness}^{\uparrow}}$ *row.*

The classifier compares accepted witness packets only through these displayed rows and their componentwise transports.

Theorem 7338.2 (Martingale convergence witness NameCert obligations)**.** *For every accepted* $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ *carrier*

$$W = (M, T, U, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceWitness}^{\uparrow}}$ *surface consists exactly of martingale-source exposure through M, bounded stopping exposure through T, upcrossing-count exposure through U, Doob-bound exposure through D, finite schedule exposure through S, regular rational readback through R, Real boundary exposure through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7338.1. Its stochastic source row is M, its bounded stopping row is T, and its upcrossing witness rows are U, D.

Read the schedule and boundary side next. The rows S, R, L are the finite observation schedule, regular rational readback, and Real lower-upper boundary rows already required by chapter 2246.

The structural rows H, C, P, N transport, replay, source, and name exactly this finite packet. Since no coordinate stores an almost-sure quotient, a selected exceptional set, a completed filtration, a measure-completion object, or an ambient convergence theorem, the listed rows exhaust the local certificate. □

Theorem 7338.3 (Martingale convergence witness upcrossing factorization)**.** *Every upcrossing read exported by an accepted* $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ *packet factors through*

$$\mathsf{Martingale}^{\uparrow},\ \mathsf{StoppingTime}^{\uparrow},\ \mathsf{OptionalStopping}^{\uparrow},\ \mathsf{Upcrossing}^{\uparrow},\ \mathsf{DoobUpcrossing}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the witness may be consumed by $\mathsf{MartingaleConvergence}^{\uparrow}$ *only after the martingale row, bounded stopping row, completed-crossing count, Doob bound, finite schedule, regular rational readback, and Real boundary rows have all been displayed.*

Proof. Begin with the accepted carrier $W = (M, T, U, D, S, R, L, H, C, P, N)$. The rows M and T expose the finite martingale source and bounded stopping window before any crossing count is read.

Apply the sibling upcrossing packets. The row U is the finite completed-count ledger supplied through chapter 1629, and D is the finite Doob-bound row supplied through chapter 1630.

Now read the consumer-facing rows. The finite schedule S, the regular rational readback R, and the Real boundary row L are precisely the rows through which chapter 2246 admits a convergence-preparation read.

Finally transport and replay are componentwise through H and C, with P, N fixing provenance and local naming. Thus the export route factors through the displayed sibling chapters and cannot bypass them by an ambient limiting event or an almost-sure quotient. □

Theorem 7338.4 (Martingale convergence witness finite extraction)**.** *The finite extraction of an accepted* $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ *packet is exactly the subpacket*

$$(M, T, U, D, S, R, L, H, C, P, N)$$

needed by a $\mathsf{MartingaleConvergence}^{\uparrow}$ *consumer to read a bounded upcrossing witness. The extraction exports no selected limit, no exceptional null set, no completed filtration, no measure completion, no unbounded stopping principle, and no ambient convergence theorem.*

Proof. Project the carrier and use Definition 7338.2. The local NameCert surface admits only the martingale, bounded stopping, upcrossing count, Doob bound, finite schedule, regular rational readback, Real boundary, transport, replay, provenance, and local naming rows.

Use Definition 7338.3. Any convergence-facing read through $\mathsf{MartingaleConvergence}^{\uparrow}$ receives the witness only along the displayed route through martingale, optional-stopping, upcrossing, Doob-upcrossing, StreamName, RegSeqRat, and Real rows.

The refused objects have no coordinate in the extracted packet and no structural row in H, C, P, N. Row exhaustion therefore prevents the extraction from exporting a selected limit, exceptional null set, completed filtration, measure completion, unbounded stopping principle, or ambient convergence theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleConvergenceWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleConvergenceWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7338.1, Definition 7338.2, Definition 7338.3, and Definition 7338.4 seed the finite upcrossing witness boundary consumed by $\mathsf{MartingaleConvergence}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MartingaleConvergenceWitness}^{\uparrow}}, \mathsf{formalTarget})$

Lean4 (statement) `BEDC.Derived.MartingaleConvergenceWitnessUp`

Bridge status: none

Not claimed: This chapter does not close almost-sure quotient equality, selected exceptional sets, completed filtrations, measure completion, selected limits, unbounded stopping principles, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded stopping exposure, upcrossing-count exactness, Doob-bound consumption, finite schedule readback, Real-boundary routing, transport, replay, provenance, and local naming for BEDC.Derived.MartingaleConvergenceWitnessUp.

Constructive story: A $\mathsf{MartingaleConvergenceWitness}^{\uparrow}$ packet is generated from martingale, bounded stopping, finite upcrossing, Doob-bound, StreamName schedule, RegSeqRat readback, Real-boundary, transport, replay, provenance, and local NameCert rows.

Chapter 7339

A Concrete Naming Certificate for MetaCICEtaExpansionAudit$^{\uparrow}$

The MetaCICEtaExpansionAudit$^{\uparrow}$ packet realizes the beta audit map of chapter 9384 as finite eta-expansion boundary data. It sits beside the beta audit packet of chapter 761, the normalization frontier of chapter 892, the critical path of chapter 765, and the closed beta trace envelope of chapter 6692. The packet records eta rows only as auditable finite data adjacent to beta conversion boundaries; it does not prove unrestricted eta conversion, subject reduction, confluence, normalization, or host conversion equality.

Definition 7339.1 (MetaCIC eta-expansion audit carrier)**.** *A* MetaCICEtaExpansionAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, B, N, F, O, T, R, V, H, C, P, Q)$$

with the following displayed rows:

1. *A is the* MetaCICBetaAudit$^{\uparrow}$ *source row naming the audited beta boundary.*
2. *B is the finite beta-conversion-boundary row over the displayed* BetaStep *and* BetaConv *relations.*
3. *N is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row naming the bounded normalization frontier inspected by the packet.*
4. *F is the* MetaCICCriticalPath$^{\uparrow}$ *row recording which syntax boundary or typing route is being audited.*
5. *O is the* MetaCICClosedBetaTraceEnvelope$^{\uparrow}$ *obstruction-retention row.*
6. *T is the eta-row ledger, recording the source term, target term, binder shape, and finite eta-expansion request.*
7. *R is the replay row that rechecks the eta ledger against the beta boundary without turning it into a global conversion rule.*
8. *V is the provenance row tying the packet to the beta audit map and the normalization frontier.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay through the eta-row, beta-boundary, normalization-frontier, obstruction-retention, transport, replay, provenance, and local naming rows.*

11. *P is the* Pkg *provenance over* MetaCICEtaExpansionAudit$^{\uparrow}$, MetaCICBetaAudit$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICClosedBetaTraceEnvelope$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetaCICEtaExpansionAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted eta conversion, subject reduction, confluence, normalization, or host conversion equality.

Theorem 7339.2 (MetaCIC eta-expansion audit NameCert obligations)**.** *For every accepted* MetaCICEtaExpansionAudit$^{\uparrow}$ *carrier*

$$E = (A, B, N, F, O, T, R, V, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICEtaExpansionAudit}^{\uparrow}}$ *surface consists exactly of beta-audit source exposure, beta-conversion-boundary exposure, normalization-frontier exposure, critical-path exposure, obstruction-retention exposure, finite eta-row exposure, replay exposure, provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7339.1. The source audit row A, beta boundary B, normalization frontier N, critical path F, and closed trace envelope O are displayed before the eta ledger can be consumed.

Read the eta row locally. The row T names a finite source, target, binder shape, and eta request, while R replays that request against the displayed beta boundary.

The obstruction row remains present. Since O is carried with T and R, a consumer cannot erase the boundary cases that separate eta-row auditing from global conversion equality.

Finally H, C, P, Q transport, replay, source, and name exactly that packet. No row supplies unrestricted eta conversion, subject reduction, confluence, normalization, or host conversion equality, so the local certificate is limited to the finite audit obligations named above. □

Theorem 7339.3 (MetaCIC eta-expansion audit boundary)**.** *Every consumer of an accepted* MetaCICEtaExpansionAudit$^{\uparrow}$ *packet reads eta expansion only through finite eta rows, the displayed beta-conversion boundary, the normalization-frontier row, the critical-path row, the retained obstruction row, transport, replay, provenance, and local* NameCert *data. The packet does not authorize unrestricted eta conversion, subject reduction, confluence, normalization, or host conversion equality.*

Proof. Start with the beta audit row. It fixes the audited BetaStep and BetaConv boundary before any eta row is inspected.

The normalization frontier and critical path narrow the read to a finite syntax and typing boundary. The retained obstruction row stays in the packet, so the consumer must keep the cases where conversion, typing preservation, and normalization are not globally available.

The eta ledger is therefore a replayable audit row, not a conversion rule. The hsame, Cont, Pkg, and NameCert rows preserve and name that finite route without adding an external equality principle.

Thus the packet exports only the finite eta/beta audit boundary stated above. It cannot prove or imply unrestricted eta conversion, subject reduction, confluence, normalization, or host conversion equality. □

Falsifiable prediction: A claimed MetaCICEtaExpansionAudit$^{\uparrow}$ read that derives host conversion equality or unrestricted eta conversion while bypassing the MetaCICBetaAudit$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$,

MetaCICCriticalPath$^{\uparrow}$, MetaCICClosedBetaTraceEnvelope$^{\uparrow}$, finite eta-row, obstruction-retention, transport, replay, provenance, and local NameCert rows refutes the carrier of Definition 7339.1.

Independence witness: metacicbetaaudit, metacicnormalizationfrontier, metaciccriticalpath, metacic-closed-beta-trace-envelope: no listed sibling is bijective to MetaCICEtaExpansionAudit$^{\uparrow}$ because this packet alone binds finite eta rows to a beta-conversion boundary, normalization frontier, critical path, retained obstruction envelope, transport, replay, provenance, and local naming while refusing global conversion claims.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICEtaExpansionAudit}^{\uparrow}}$.

Constructive story: A MetaCICEtaExpansionAudit$^{\uparrow}$ packet is generated from beta-audit source rows, beta-conversion boundary rows, normalization-frontier rows, critical-path rows, obstruction-retention rows, finite eta-row data, replay, provenance, transport, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICEtaExpansionAudit}^{\uparrow}}$, seed)

Scope closed: Definition 7339.1, Definition 7339.2, and Definition 7339.3 seed the finite eta-expansion audit boundary for MetaCIC beta-conversion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICEtaExpansionAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted eta conversion, subject reduction, confluence, normalization, host conversion equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked beta-audit exposure, eta-row exposure, beta-boundary retention, normalization-frontier routing, obstruction retention, transport, replay, provenance, and local naming rows for BEDC.Derived.MetaCICEtaExpansionAuditUp.

Chapter 7340

A Concrete Naming Certificate for $\mathsf{MackeyTopology}^{\uparrow}$

Following Mackey and Bourbaki on dual pairs and locally convex topologies, the $\mathsf{MackeyTopology}^{\uparrow}$ packet records the finite BEDC rows that name a Mackey-style topology through displayed dual-pair probes, bounded-set compatibility, and seminorm windows. It sits after the weak topology carrier of chapter 3597, the bounded-family surface of chapter 3370, the barrelled-space boundary of chapter 7269, the topological-vector-space handoff of chapter 205, and the linear-map rows of chapter 48. The packet names only finite dual-pair, comparison, bounded-family, seminorm-window, transport, replay, provenance, and local naming rows; it does not import host locally convex space completeness, Hahn–Banach separation, arbitrary choice, quotient functional equality, Lean verification, or bridge checking.

Definition 7340.1 (Mackey topology carrier)**.** *A* $\mathsf{MackeyTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, W, B, S, T, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the dual-pair source row, naming the vector carrier, displayed functional window, evaluation relation, and finite pairing ledger.*
2. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *comparison row over the same dual-pair source, recording which weak-neighbourhood reads are consumed before the Mackey row is exposed.*
3. *B is the bounded-set and* $\mathsf{Bornology}^{\uparrow}$ *row, listing the displayed bounded families against which uniform convergence windows are tested.*
4. *S is the seminorm-window ledger, with each row naming a finite bounded-family probe, its seminorm read, and the neighbourhood row generated by that read.*
5. *T is the* $\mathsf{TopVecSpace}^{\uparrow}$ *handoff row, exposing addition-continuity and scalar-continuity only after D, W, B, S have been displayed.*
6. *L is the* $\mathsf{LinMap}^{\uparrow}$ *handoff row for dual-pair evaluations and bounded-family probes that downstream operator consumers may inspect.*
7. *H records componentwise* hsame *transport for D, W, B, S, T, L.*
8. *C records displayed* Cont *replay from a topology request through the same dual-pair, weak-topology, bornology, seminorm, topological-vector-space, and linear-map rows.*

9. *P is the* Pkg *provenance over* $\mathsf{MackeyTopology}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Bornology}^{\uparrow}$, $\mathsf{BarrelledSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MackeyTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient locally convex topology, a selected full dual, a quotient equality of functionals, a Hahn–Banach separator, a completeness theorem, arbitrary choice, or bridge evidence.

Theorem 7340.2 (Mackey topology NameCert obligations)**.** *For every accepted* $\mathsf{MackeyTopology}^{\uparrow}$ *carrier $M = (D, W, B, S, T, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MackeyTopology}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dual-pair source D;*
2. *weak-topology comparison through W over the same vector and functional windows;*
3. *bounded-family and bornology exposure through B;*
4. *seminorm-window admission through the finite ledger S;*
5. *topological-vector-space and linear-map handoff through T and L;*
6. *nonescape through H, C, P, N, excluding host locally convex space completeness, Hahn–Banach separation, arbitrary choice, quotient functional equality, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 7340.1. The tuple $D, W, B, S, T, L, H, C, P, N$ is the whole accepted packet, so each obligation row is a finite coordinate read.

The topology comparison begins with the dual-pair source D and the weak-topology row W. These rows expose only displayed evaluation and weak-neighbourhood data, so a consumer cannot replace them by an ambient weak topology or by equality of a full dual space.

Next read the bornology and seminorm data. The row B names the bounded-family surface, and every seminorm request must pass through one of the finite windows in S before any topological-vector-space or linear-map handoff is available.

Finally the rows T, L, H, C, P, N transport, replay, source, and name the same finite packet. Since none of those rows stores completeness, Hahn–Banach separation, arbitrary choice, quotient functional equality, Lean verification, or bridge evidence, the listed obligations exhaust the local NameCert surface. □

Theorem 7340.3 (Mackey topology dual-pair finite route)**.** *Every topology read exported by an accepted* $\mathsf{MackeyTopology}^{\uparrow}$ *packet factors through the finite route*

$$D \longmapsto W \longmapsto B \longmapsto S \longmapsto T, L \longmapsto H, C, P, N.$$

Thus a downstream consumer may read the Mackey topology only as displayed dual-pair data compared with weak-topology rows, bounded-family rows, finite seminorm-window rows, and the named topological-vector-space and linear-map handoffs.

Proof. Induct over the displayed row order of the carrier. The base row is the dual-pair source D, which supplies the vector, functional-window, evaluation, and pairing data from which all later rows are read.

For the first step, the weak-topology comparison row W consumes only D. For the next step, the bornology row B consumes the same source and lists the bounded-family tests without selecting a host family or a quotient functional equality.

For the seminorm step, each entry of S names a finite bounded-family probe and its generated neighbourhood row. The handoff rows T and L are therefore reached only after the weak-topology comparison and bounded-family ledger have already been displayed.

The terminal rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Repacking these projections gives the displayed finite route and prevents any consumer from importing completeness, Hahn–Banach separation, arbitrary choice, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{MackeyTopology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MackeyTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7340.1, Definition 7340.2, and Definition 7340.3 seed the finite Mackey topology packet over dual-pair, weak-topology, bornology, seminorm-window, topological-vector-space, linear-map, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MackeyTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex space completeness, Hahn–Banach separation, arbitrary choice, quotient functional equality, Lean verification, or bridge checking for $\mathsf{MackeyTopology}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, weak-topology comparison, bornology exposure, seminorm-window exactness, topological-vector-space and linear-map handoff, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.MackeyTopologyUp.

Constructive story: A $\mathsf{MackeyTopology}^{\uparrow}$ packet is generated from displayed dual-pair source rows, weak-topology comparison rows, bornology and bounded-family rows, finite seminorm-window rows, topological-vector-space and linear-map handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7341

A Concrete Naming Certificate for $\mathsf{CylinderSet}^{\uparrow}$

Following Kolmogorov and standard product-measure constructions, the $\mathsf{CylinderSet}^{\uparrow}$ packet records the finite cylinder-event substrate used before any σ-algebra completion or product-measure existence theorem is invoked. It sits between the finite endpoint discipline of chapter 32, the finite observation windows of chapter 126, and the probability-facing packets of chapter 184, chapter 185, and chapter 186.

Definition 7341.1 (Cylinder set carrier)**.** *A* $\mathsf{CylinderSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, I, S, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite coordinate window, listing the inspected coordinates and their order of access.*
2. *A is the coordinate-event row, assigning to each displayed coordinate a finite* $\mathsf{Bool}^{\uparrow}$*-readable event decision surface.*
3. *I is the Boolean finite-intersection ledger, recording conjunction, empty-window top event, and pairwise merge rows for the displayed coordinate events.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source window, or the product-style source window, from which the finite coordinate reads are replayed.*
5. *H records componentwise* hsame *transport for the coordinate window, coordinate-event rows, Boolean intersection ledger, source window, replay routes, provenance, and local naming.*
6. *C records displayed* Cont *replay for every finite coordinate read and every Boolean intersection read.*
7. *P is the* Pkg *provenance over* $\mathsf{CylinderSet}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CylinderSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for σ-algebra completion, countable unions, product-measure existence, host set membership, selected infinite branch, quotient stream equality, or ambient probability space semantics.

Theorem 7341.2 (Cylinder set prefix stability)**.** *Let $E = (W, A, I, S, H, C, P, N)$ and $E' = (W', A', I', S', H', C', P', N')$ be accepted* $\mathsf{CylinderSet}^{\uparrow}$ *packets. If W' extends W only by coordinates not inspected by the read, and the rows A, I, S, H, C, P, N transport componentwise under* hsame *through the common prefix, then every consumer restricted to W reads the same cylinder classifier from E and E'.*

Proof. Project both packets to the carrier of Definition 7341.1. The consumer restricted to W can inspect only the listed coordinate rows, the Boolean finite-intersection ledger over those rows, and the source replay window visible through S.

The extension coordinates in W' are outside that finite read. Componentwise hsame transport identifies the shared coordinate-event rows and the shared Boolean ledger rows on the prefix, while C replays the same finite access route.

Since the carrier has no host set coordinate or quotient stream equality coordinate, the classifier visible to the restricted consumer is exhausted by the shared prefix rows. Therefore the two packets determine the same prefix cylinder classifier. □

Theorem 7341.3 (Cylinder set finite-intersection ledger)**.** *For every accepted* $\mathsf{CylinderSet}^{\uparrow}$ *packet $E = (W, A, I, S, H, C, P, N)$, the finite intersection of displayed coordinate events is read entirely from the* $\mathsf{Bool}^{\uparrow}$ *ledger I over the same window W. The empty intersection, binary merge, and finite fold rows are all local to the packet, and no countable union, complement-completion, host set operation, or product-measure row is exported by the ledger.*

Proof. Read the finite window W and the coordinate-event row A. These two rows determine the finite list of Boolean event decisions available to the packet.

The ledger I supplies the empty-window top event, the binary merge row, and the finite fold over that displayed list. Induction over the finite BHist window gives the intersection row without asking for a countable family or an external set operation.

The remaining rows H, C, P, N transport, replay, package, and name the same finite ledger. Because the carrier excludes completion and product-measure coordinates, the finite-intersection read cannot export them. □

Theorem 7341.4 (Cylinder set probability-event readiness)**.** *Every* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, *or* $\mathsf{Distribution}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{CylinderSet}^{\uparrow}$ *packet as an event must first expose the finite coordinate window W, the coordinate-event rows A, the Boolean finite-intersection ledger I, the* $\mathsf{StreamName}^{\uparrow}$ *or product-style source window S, and the structural rows H, C, P, N. The handoff supplies a finite event substrate only; it exports no σ-algebra completion, countable additivity, product-measure existence proof, host set semantics, selected infinite branch, or ambient probability object.*

Proof. Begin with the finite source read. The rows W and S display exactly which stream or product coordinates are inspected before a probability-facing consumer may treat the packet as an event.

Next read A and I. The coordinate-event rows are $\mathsf{Bool}^{\uparrow}$-readable, and the finite-intersection ledger is the only event constructor present in the carrier.

Finally pass through H, C, P, N. These rows preserve the finite read, replay the displayed route, record provenance toward $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{Distribution}^{\uparrow}$ consumers, and name the local certificate. Since no completion, countable-additivity, product-measure, or host-set row occurs in the carrier, the handoff is event-ready only at the finite cylinder level. □

Falsifiable prediction: A probability-facing consumer that reads a $\mathsf{CylinderSet}^{\uparrow}$ event without exposing the finite coordinate window, coordinate-event rows, Boolean finite-intersection ledger, $\mathsf{StreamName}^{\uparrow}$ or product-style source window, hsame, Cont, Pkg, and NameCert rows refutes the $\mathsf{CylinderSet}^{\uparrow}$ carrier.

Independence witness: bool, streamname, probspace, randomvar, distribution: $\mathsf{CylinderSet}^{\uparrow}$ binds finite coordinate windows, Boolean coordinate events, finite-intersection rows, source-window replay, and probability-facing provenance in one local NameCert packet; each named sibling lacks at least one of these rows as a local cylinder-event obligation.

Closure status for $\mathsf{NameCert}_{\mathsf{CylinderSet}^{\uparrow}}$.

Constructive story: A $\mathsf{CylinderSet}^{\uparrow}$ packet is generated from a finite coordinate window, coordinate-event rows, a Boolean finite-intersection ledger, a StreamName or product-style source window, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. Its classifier is a finite cylinder-event surface, not a completed measurable set or a product-measure theorem.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CylinderSet}^{\uparrow}}$, seed)

Scope closed: Definition 7341.1, Definition 7341.2, Definition 7341.3, and Definition 7341.4 seed the finite cylinder-event route over $\mathsf{Bool}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CylinderSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: It does not claim σ-algebra completion, countable additivity, product-measure existence, host set semantics, selected infinite branches, quotient stream equality, or ambient probability-space semantics.

Upgrade path: To reach obligationClosure, the prefix-stability, finite-intersection, and probability-event-readiness obligations must be enumerated against a future Lean carrier under `BEDC.Derived.CylinderSetUp`.

Chapter 7342

A Concrete Naming Certificate for $\mathsf{IteratedFunctionSystem}^{\uparrow}$

Following Hutchinson on iterated function systems, the $\mathsf{IteratedFunctionSystem}^{\uparrow}$ packet records a finite family of contraction maps on a compact complete metric carrier as BEDC naming data. It sits after the metric source of chapter 57, the complete-metric limit surface of chapter 128, the compact-metric boundary of chapter 129, the map interface of chapter 130, the contraction route of chapter 1418, the tolerance schedules of chapter 292, and the real seal of chapter 38. The packet names only the finite contraction-family surface and its dependency rows; it does not construct an attractor, a hyperspace completion, a Hausdorff metric fixed point, or a measure on the limit set.

Definition 7342.1 (IteratedFunctionSystem carrier)**.** *An* $\mathsf{IteratedFunctionSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, M, K, F, L, R, Q, S, A, H, C, P, N)$$

with the following rows:

- X *is the displayed* $\mathsf{Met}^{\uparrow}$ *source carrier.*
- M *is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness surface for Cauchy reads in* X*.*
- K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *total-bounded and closed-boundary row used by finite cover consumers.*
- F *is a finite ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose sources and targets are* X*.*
- L *is the finite Lipschitz ledger, with each entry routed through a* $\mathsf{ContractionMapping}^{\uparrow}$ *row.*
- R *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold ledger used by the contraction estimates.*
- Q *is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row for the displayed contraction constants.*
- S *is the finite index and family-membership row naming which maps are in the system.*
- A *is the attractor-obligation placeholder row: it records the route a later fixed-set consumer must inspect, but it is not itself an attractor construction.*
- H, C, P, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{IteratedFunctionSystem}^{\uparrow}}$ *rows.*

Two carriers are classified only by the componentwise rows $X, M, K, F, L, R, Q, S, A, H, C, P, N$*. The classifier has no coordinate for quotient hyperspaces, selected fixed compact sets, ambient classical compactness, choice of a point in the attractor, or host equality of functions.*

Lean4 (statement) `BEDC.Derived.IteratedFunctionSystemUp`

Theorem 7342.2 (IteratedFunctionSystem NameCert obligations)**.** *For an accepted* $\mathsf{IteratedFunctionSystem}^{\uparrow}$ *carrier*

$$I = (X, M, K, F, L, R, Q, S, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IteratedFunctionSystem}^{\uparrow}}$ *surface is exhausted by five obligations: the carrier is inhabited by finite displayed rows; classifier equality is reflexive, symmetric, and transitive by componentwise* hsame *over those rows; and classifier transport preserves the metric, complete-metric, compact-metric, continuous-map, contraction, modulus, Real-seal, finite-index, attractor-obligation, replay, provenance, and naming coordinates. No obligation may be discharged by a host hyperspace quotient, selected attractor, global choice principle, or ambient compactness theorem outside the displayed packet.*

Proof. Project the carrier of Definition 7342.1. Every row is a finite BHist coordinate, and the classifier compares only those coordinates through componentwise hsame.

Reflexivity, symmetry, and transitivity are therefore the corresponding componentwise hsame laws for $X, M, K, F, L, R, Q, S, A, H, C, P, N$. The structural rows H, C, P, N then transport, replay, source, and name the same finite packet.

The excluded objects are not carrier coordinates. A NameCert consumer can inspect only the displayed metric, compactness, map-family, contraction, modulus, Real-seal, and structural rows, so no host hyperspace quotient or selected attractor can become a local obligation. □

Theorem 7342.3 (IteratedFunctionSystem contraction route)**.** *Every contraction-family request served by an accepted* $\mathsf{IteratedFunctionSystem}^{\uparrow}$ *carrier*

$$I = (X, M, K, F, L, R, Q, S, A, H, C, P, N)$$

must first read the metric source X, the continuous-map ledger F, the finite family row S, the contraction ledger L, and the modulus threshold ledger R. It may then use the $\mathsf{CompleteMetric}^{\uparrow}$ *row M, the* $\mathsf{CompactMetric}^{\uparrow}$ *row K, and the* $\mathsf{Real}^{\uparrow}$ *seal row Q only through the same componentwise transport, replay, provenance, and naming rows H, C, P, N. Thus the contraction route is a finite displayed BEDC route, not a direct appeal to a Hutchinson attractor theorem.*

Proof. Begin with a contraction-family request. The carrier exposes maps only through the finite continuous-map ledger F and records family membership only through S, both over the metric source X.

Read the contraction ledger L and threshold ledger R next. These rows are the only places where Lipschitz bounds and requested tolerances are stored before a later compact or complete-metric consumer can use them.

Finally pass through M, K, Q, H, C, P, N. The complete-metric, compact-metric, and Real-seal rows are consumed through the same displayed packet, while H, C, P, N preserve and name that packet. Since the carrier does not contain a hyperspace fixed-set row, the route cannot claim an attractor construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{IteratedFunctionSystem}^{\uparrow}}$.

Constructive story: An $\mathsf{IteratedFunctionSystem}^{\uparrow}$ packet is generated from a compact complete metric source, a finite continuous-map family, contraction ledgers, modulus thresholds, Real-seal constants, family-membership rows, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IteratedFunctionSystem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7342.1, Definition 7342.2, and Definition 7342.3 seed the finite contraction-family carrier and its dependency route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IteratedFunctionSystem}^{\uparrow}}, \text{unformalized})$

Lean target: `BEDC.Derived.IteratedFunctionSystemUp`

Bridge status: none

Not claimed: This chapter does not close Hutchinson attractors, hyperspace completion, Hausdorff-metric fixed-set existence, invariant measures, Lean theorem coverage, or bridge checking.

Upgrade path: Scoped closure requires checked carrier obligations for the finite family ledger, contraction-modulus route, compact complete metric consumer boundary, and Real-seal nonescape.

Chapter 7343

A Concrete Naming Certificate for $\mathsf{RealUpperLowerApproximation}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{RealUpperLowerApproximation}^{\uparrow}$ packet records the two-sided approximation surface by which lower and upper rational names are read through a shared finite window before any located-real or completion-facing consumer sees a terminal real seal. It sits between the finite stream discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance budget of chapter 291, the located-real boundary of chapter 1933, and the terminal $\mathsf{Real}^{\uparrow}$ certificate of chapter 38. The packet names a finite upper/lower approximation object only; it does not choose representatives or assert ambient completeness.

Definition 7343.1 (Real upper/lower approximation carrier)**.** *A* $\mathsf{RealUpperLowerApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower approximation row, a finite monotone rational read that may be consumed only as lower evidence.*
2. *U is the upper approximation row, a finite antitone rational read that may be consumed only as upper evidence.*
3. *W is the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window through which both L and U are inspected at the same precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the lower and upper rational observations inside W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap budget recording the displayed tolerance between the lower and upper reads.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the shared window, readback, and gap budget have been displayed.*
7. *H records componentwise* hsame *transport for L, U, W, R, D, E.*
8. *C records displayed* Cont *replay from the two-sided approximation rows through the shared window into the terminal seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RealUpperLowerApproximation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUpperLowerApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for Dedekind cuts, quotient real equality, selected representatives, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7343.2 (Real upper/lower approximation NameCert obligations)**.** *For every accepted* $\mathsf{RealUpperLowerApproximation}^{\uparrow}$ *carrier* $A = (L, U, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUpperLowerApproximatio}}$ *surface consists exactly of carrier admission for the lower approximation, upper approximation, shared finite stream window, regular rational readback, dyadic gap budget, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows; lower/upper polarity exactness for L and U; shared-window exactness for W; gap-budget visibility for D before E; and nonescape of quotient equality, selected representatives, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7343.1. Its analytic rows are exactly the lower read, upper read, shared window, regular rational readback, dyadic budget, and terminal real seal.

The polarity and shared-window clauses are read from L, U, W, R, D. The lower and upper rows are separate coordinates, and both pass through the same $\mathsf{StreamName}^{\uparrow}$ window before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ gap budget are visible.

The structural rows H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier has no coordinate for quotient equality, selected representatives, trichotomy, countable choice, or ambient completeness, the listed clauses exhaust the local NameCert surface. □

Theorem 7343.3 (Real upper/lower approximation window handoff)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUpperLowerApproximation}^{\uparrow}$ *packet reads the finite route*

$$L, U, W, R, D, E, H, C, P, N$$

in that order: lower and upper approximation rows, the shared $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap budget, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The handoff may supply a located two-sided approximation boundary, but it does not supply Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier rows. A consumer sees L and U as two-sided rational approximation data, but both rows are tied to the same finite window W.

Read the regular rational and dyadic rows next. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R and $\mathsf{DyadicRatCore}^{\uparrow}$ budget D expose the finite gap discipline required before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

Finally transport and replay the route through H, C, P, N. These rows preserve the displayed window handoff for $\mathsf{LocatedReal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, while all completeness, quotient, representative, trichotomy, and countable-choice claims remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUpperLowerApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUpperLowerApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7343.1, Definition 7343.2, and Definition 7343.3 seed the finite two-sided approximation route over lower and upper rational reads, shared stream windows, regular rational readback, dyadic gap budgets, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{RealUpperLowerApproximation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower/upper polarity exactness, shared-window exactness, gap-budget visibility, terminal-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealUpperLowerAppro

Constructive story: A RealUpperLowerApproximation$^{\uparrow}$ packet is generated from lower and upper rational approximation rows, a shared StreamName window, RegSeqRat readback, DyadicRatCore gap budget, a terminal Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 7344

A Concrete Naming Certificate for $\mathsf{EquicontinuousCompactImage}^{\uparrow}$

Following Arzela–Ascoli and Bishop–Bridges constructive compact metric analysis, the EquicontinuousCompactImag packet records the finite image-window surface obtained from an equicontinuous family over a compact metric source. It sits after the compact source of chapter 129, the displayed function rows of chapter 130, the family modulus surface of chapter 413, the tolerance-only consumer of chapter 369, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet exposes compact-image data only as finite source nets, synchronized family windows, regular readback, and Real seals.

Definition 7344.1 (Equicontinuous compact image carrier)**.** *An* $\mathsf{EquicontinuousCompactImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, M, U, S, R, Q, H, C, P, N)$$

with the following rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, carrying the displayed centers, coverage witnesses, and metric-window data available to the image computation.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family graph row restricted to the inspected maps and source centers.*
3. *M is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row that synchronizes the family at the displayed source windows.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row receiving the finite tolerance schedule induced by K, F, M.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *image-window row exposed only at listed source centers and displayed family indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the image values and their tolerance schedule.*
7. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed image values.*
8. *H records componentwise* hsame *transport for compact-source, graph, equicontinuity, uniform-modulus, image-window, readback, seal, continuation, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a target image tolerance through*

$$K, F, M \longmapsto U, S, R, Q.$$

10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousCompactImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousCompactImage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host image compactness, open-cover compactness, quotient function equality, selected image points outside the finite net, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7344.2 (Equicontinuous compact image NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousCompactImage}^{\uparrow}$ *carrier $E = (K, F, M, U, S, R, Q, H, C, P, N)$, the local NameCert obligation surface consists exactly of compact finite-net admission through K, displayed continuous-map image admission through F, equicontinuity-family synchronization through M, uniform-modulus handoff through U, finite image-window exposure through S, regular rational readback through R, terminal Real sealing through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

No obligation row asserts host image compactness, an open-cover finite subcover, quotient equality of functions, a selected image point outside the listed windows, countable choice, or ambient Real completeness.

Proof. Project the carrier of Definition 7344.1. The compactness-facing data is the finite-net row K, the function-facing data is the displayed graph row F, and the family-control data is the equicontinuity row M.

The tolerance handoff is finite. A target tolerance request first reads the source net in K, then the graph rows in F at the displayed centers, and then the family synchronization row M. The resulting tolerance schedule is passed to U.

The image rows are downstream of that handoff. The window row S exposes only listed image observations, R supplies regular rational readback for those observations, and Q seals the resulting Real-facing values.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same route, so no host compactness principle, selected subcover, quotient function equality, choice schedule, selected image point, or ambient completeness row is introduced. □

Theorem 7344.3 (Equicontinuous compact image L10 route)**.** *Every L10 consumer of an accepted* $\mathsf{EquicontinuousCompactImage}^{\uparrow}$ *packet must first inspect the finite compact-metric source row, the continuous-map graph row, the equicontinuity-family synchronization row, and the uniform-modulus handoff before reaching image windows, regular rational readback, and the terminal Real seal. Concretely, every such read factors through*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{EquicontinuityFamily}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed carrier rows, not as host compactness or quotient function data.

Proof. Start with an L10 image read of the carrier in Definition 7344.1. The first available source data is K, the compact-metric finite net, so the consumer cannot begin from an open cover or a host image set.

Read the function and family rows next. The graph row F supplies displayed image values at inspected centers, while M forces those reads to share the equicontinuity-family synchronization.

The handoff row U then records the tolerance-only uniform-modulus surface consumed by the downstream route.

Only after K, F, M, U are displayed can the consumer reach S, R, Q. These rows expose finite image windows, regular rational readback, and the terminal Real seal.

The structural rows H, C, P, N preserve this order. Therefore the L10 route exports only the displayed compact-source, map, family-modulus, window, readback, and seal rows, while refusing host image compactness, open-cover compactness, quotient function equality, selected image points, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{EquicontinuousCompactImage}^{\uparrow}$ consumer read factors through CompactMetric finite-net rows, ContinuousMap graph rows, EquicontinuityFamily synchronization rows, UniformModulus handoff, StreamName windows, RegSeqRat readback, RealUp seals, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: compactmetric, continuousmap, equicontinuityfamily, uniformmodulus, streamname, regseqrat, real: no listed sibling supplies the whole eleven-row packet $E = (K, F, M, U, S, R, Q, H, C, P, N)$, because each sibling contributes only a compact source, graph row, family synchronization row, tolerance handoff, finite window, regular readback, or Real seal while this chapter binds the compact-image route through transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuousCompactImage}^{\uparrow}}$.

Constructive story: An $\mathsf{EquicontinuousCompactImage}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap graph rows, EquicontinuityFamily synchronization rows, UniformModulus handoff, StreamName image windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompactImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7344.1, Definition 7344.2, and Definition 7344.3 seed the finite compact-metric equicontinuous image route before L10 consumers inspect image windows and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompactImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host image compactness, open-cover compactness, quotient function equality, selected image points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, continuous-map graph admission, family synchronization, uniform-modulus handoff, image-window exposure, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EquicontinuousCompactImageUp.

Chapter 7345

A Concrete Naming Certificate for $\mathsf{LocallyConnectedContinuum}^{\uparrow}$

Following Hahn–Mazurkiewicz and Kelley on compact metric continua, the $\mathsf{LocallyConnectedContinuum}^{\uparrow}$ packet records the compact, connected, locally connected metric layer that Peano-continuum and dendrite consumers read before importing their richer endpoint or image data. It sits over the compact metric carrier of chapter 129, the finite chain-connected route of chapter 2050, the continuous-map surface of chapter 59, and the downstream continuum consumers of chapter 2160 and chapter 5803. The packet names the local connectedness refinement boundary without asserting a space-filling curve, unique-arc data, open-cover compactness, quotient equality of points, or ambient real completeness.

Definition 7345.1 (Locally connected continuum carrier)**.** *A* $\mathsf{LocallyConnectedContinuum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, C, R, W, M, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric carrier, finite compact-window rows, total-boundedness ledger, and completion-facing comparisons used by continuum consumers.*
2. *C is the connectedness source row. It either records a displayed connectedness witness over K or a finite* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *route whose chain ledgers are read as connectedness-facing data for the same compact metric source.*
3. *R is the local connectedness refinement ledger. For each displayed compact window it records connected neighbourhood rows, inclusion refinements, overlap witnesses, radius comparisons, and the refusal of host open-set lattices outside the packet.*
4. *W is the compact-window row, recording the finite family of windows through which later interval, image, and dendrite reads may inspect the continuum.*
5. *M is the continuous-map consumer route, exposing only the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph, modulus, and window-read rows allowed to consume K, C, R, W.*
6. *H records componentwise* hsame *transport for the compact metric source, connectedness row, local connectedness refinement ledger, compact windows, continuous-map route, replay row, provenance row, and local naming row.*

7. *T records displayed* Cont *replay of the route*

$$K, C, R, W \longmapsto M \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocallyConnectedContinuum}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ChainConnectedMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnectedContinuum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, C, R, W, M, H, T, P, N$. The carrier has no coordinate for an ambient point-set continuum, a selected Peano parametrization, dendrite arc uniqueness, open-cover compactness, quotient point equality, countable choice, or completed real subset selection.

Lean4 (definition) `BEDC.Derived.LocallyConnectedContinuumUp.LocallyConnectedContinuumUp`

Theorem 7345.2 (Locally connected continuum NameCert obligations)**.** *For every accepted* LocallyConnectedCont *carrier*

$$L = (K, C, R, W, M, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocallyConnectedContinuum}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through K, connected or chain-connected source exposure through C, local connectedness refinement through R, compact-window visibility through W, continuous-map consumer routing through M, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The displayed tuple lists every row that may be consumed by a continuum-facing reader, so carrier admission and classifier exposure are confined to $K, C, R, W, M, H, T, P, N$.

The compact and connected source data are read first. The row K supplies the compact metric source, and C supplies either displayed connectedness evidence or the finite $\mathsf{ChainConnectedMetric}^{\uparrow}$ route over that same source. No second host space is introduced.

The local connectedness data are confined to R and W. Each compact-window request is answered by finite connected-neighbourhood, inclusion, overlap, and radius-comparison rows, while W records the windows through which downstream image and continuum readers may inspect the packet.

Finally M, H, T, P, N route, transport, replay, source, and name exactly the displayed rows. Since the carrier has no coordinate for a host continuum, selected parametrization, dendrite arc, open-cover principle, quotient point equality, countable choice, or completed real subset, the listed rows exhaust the local NameCert surface. □

Theorem 7345.3 (Locally connected continuum Peano handoff)**.** *Every* $\mathsf{PeanoContinuum}^{\uparrow}$ *or* $\mathsf{Dendrite}^{\uparrow}$ *consumer that reads a compact metric continuum through* $\mathsf{LocallyConnectedContinuum}^{\uparrow}$ *must first pass through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ChainConnectedMetric}^{\uparrow},\ R,\ W,\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto H, T, P, N.$$

Thus the consumer receives the compact metric source, connectedness route, local connectedness refinement ledger, compact windows, continuous-map route, componentwise transport, displayed replay, provenance, and local naming before reading Peano-continuum or dendrite-specific rows. It receives no host continuum, selected Peano parametrization, unique-arc dendrite structure, open-cover compactness, quotient point equality, countable-choice selector, or ambient real-completeness witness.

Proof. Start with the compact source row K. A Peano-continuum or dendrite consumer cannot read local connectedness, image, endpoint, or arc data before the metric carrier, compact windows, and finite compactness-facing rows have been displayed.

Next read connectedness through C. If the source is supplied by a chain-connected route, the consumer sees the finite endpoint, chain, radius, and tolerance ledgers of $\mathsf{ChainConnectedMetric}^{\uparrow}$ over the same compact metric row.

Now read the local connectedness and window rows R, W. These rows provide finite connected-neighbourhood refinements and compact-window boundaries before any $\mathsf{ContinuousMap}^{\uparrow}$ graph or modulus row in M is made public.

The structural rows close the handoff. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve exactly this finite dependency order, so Peano-continuum and dendrite consumers receive the locally connected continuum layer without receiving any refused host-side object. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnectedContinuum}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyConnectedContinuum}^{\uparrow}$ packet is generated from a compact metric source row, connectedness or ChainConnectedMetric row, finite local connectedness refinement ledger, compact-window row, ContinuousMap consumer route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnectedContinuum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7345.1, Definition 7345.2, and Definition 7345.3 seed the finite compact metric, connected, locally connected continuum packet for Peano-continuum and dendrite consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnectedContinuum}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.LocallyConnectedContinuumUp.LocallyConnectedContinuumUp`

Bridge status: none

Not claimed: This chapter does not close selected Peano parametrizations, unique arcs, host point-set continua, open-cover compactness, quotient point equality, countable choice, ambient real completeness, or standard bridges.

Upgrade path: Obligation closure requires checked carrier admission, connectedness visibility, local connectedness refinement, continuous-map consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.LocallyConnectedContinuumU

Chapter 7346

A Concrete Naming Certificate for MetaCICLocalConfluenceFrontier$^{\uparrow}$

The MetaCICLocalConfluenceFrontier$^{\uparrow}$ packet realizes the conditional local-diamond boundary of chapter 9378 as finite critical-pair, conditional-join, substitution-premise, obstruction, transport, replay, provenance, and local naming data. It sits between the critical-path surface of chapter 765, the normal-form consistency seal of chapter 740, the decidable-conversion boundary of chapter 4097, and the parallel-diamond frontier of chapter 4238. The packet records the conditional local confluence surface without claiming full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, residual-substitution compatibility, or kernel consistency.

Falsifiable prediction: Every accepted MetaCICLocalConfluenceFrontier$^{\uparrow}$ read exposes the critical-pair, conditional-join, residual-substitution premise, obstruction, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any local-diamond handoff can be consumed.

Independence witness: metacic-parallel-diamond-frontier, normal-form-consistency-seal, metaciccriticalpath, metacicdecidableboundary, subject-reduction-discharge-socket: the parallel-diamond frontier retains a stronger residual-diamond route without closing the local confluence premise; the normal-form consistency seal records normal-source consistency without the critical-pair frontier; the critical path records reduction-budget routing without the conditional local-join packet; the decidable boundary records conversion-frontier data without local diamond discharge; and the subject-reduction discharge socket names a typing-preservation boundary rather than a confluence frontier. None is bijective to MetaCICLocalConfluenceFrontier$^{\uparrow}$.

Definition 7346.1 (MetaCIC local confluence frontier carrier)**.** *A* MetaCICLocalConfluenceFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, J, S, O, H, T, P, N)$$

with the following displayed rows:

1. *K is the critical-pair row, exposing the local overlap, source term, two one-step residual branches, and bounded redex-window data consumed by* MetaCICCriticalPath$^{\uparrow}$*.*
2. *J is the conditional-join row, recording the candidate common reduct and the finite local-join check that may be read only after K and S are displayed.*
3. *S is the substitution-premise row, retaining the residual-substitution compatibility boundary required by the conditional local-diamond read.*

4. *O is the obstruction row, recording non-discharge of full parallel diamond, full closed Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, and kernel consistency.*
5. *H records componentwise* hsame *transport for critical pairs, conditional joins, substitution premises, obstruction data, replay, provenance, and local naming.*
6. *T records displayed* Cont *replay of the route*

$$K, S \longmapsto J \longmapsto O, H, P, N.$$

7. *P is the* Pkg *provenance over* MetaCICLocalConfluenceFrontier$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, SubjectReductionDisc BHist, hsame, Cont, *and* NameCert.
8. *N is the local* NameCert$_{\text{MetaCICLocalConfluenceFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through K, J, S, O, H, T, P, N. It has no coordinate for an unconditional diamond theorem, a subject-reduction theorem, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Lean4 (definition) `BEDC.Derived.MetaCICLocalConfluenceFrontierUp.MetaCICLocalConfluenceFrontierUp`

Theorem 7346.2 (MetaCIC local confluence frontier NameCert obligations)**.** *For every accepted* MetaCICLocalConfluenceFrontier$^{\uparrow}$ *carrier*

$$F = (K, J, S, O, H, T, P, N),$$

the local NameCert$_{\text{MetaCICLocalConfluenceFrontier}^{\uparrow}}$ *surface consists exactly of critical-pair exposure through K, conditional-join exposure through J, residual-substitution premise retention through S, obstruction nonescape through O, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The finite tuple displays the critical-pair, conditional-join, substitution-premise, obstruction, transport, replay, provenance, and local naming rows, and the classifier compares only those rows.

The confluence-facing data are K, J, S. A reader first exposes the critical pair K and retained substitution premise S; only then may it inspect the candidate local join J. This preserves the conditional boundary named in chapter 9378.

The obstruction and structural rows prevent nonescape. The row O records that the packet has not supplied full parallel diamond, closed Church–Rosser, subject reduction, evaluator equality, quotient equality, global normalization, or kernel consistency, while H, T, P, N transport, replay, source, and name the same finite rows. □

Theorem 7346.3 (MetaCIC local confluence frontier conditional handoff)**.** *Every confluence-audit consumer that reads an accepted* MetaCICLocalConfluenceFrontier$^{\uparrow}$ *packet must expose the route*

$$\text{MetaCICCriticalPath}^{\uparrow},\ S,\ J,\ O \longmapsto H, T, P, N$$

before using a local-diamond handoff. The handoff may feed the conditional surface of MetaCICParallelDiamondFront *and the consistency-facing rows of* NormalFormConsistencySeal$^{\uparrow}$*, but it does not prove residual-substitution compatibility, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, decidable conversion, or kernel consistency.*

Proof. Start with the critical-pair row K. It exposes only the local overlap and bounded residual branches consumed by the MetaCIC critical path, so the reader has not yet received any global confluence theorem.

Next read the retained substitution premise S and conditional join row J. The join is public only relative to the displayed premise, which keeps the residual-substitution boundary visible at the handoff.

Finally read O, H, T, P, N. The obstruction row records the endpoints that remain outside the packet, and the structural rows transport, replay, source, and name the same conditional frontier. Therefore the consumer receives a local confluence frontier, not a proof of full confluence, subject reduction, evaluator equality, quotient equality, decidable conversion, or kernel consistency. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICLocalConfluenceFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICLocalConfluenceFrontier}^{\uparrow}$ packet is generated from critical-pair rows, conditional-join rows, residual-substitution premise rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICLocalConfluenceFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7346.1, Definition 7346.2, and Definition 7346.3 seed the finite local confluence frontier over critical-pair, conditional-join, substitution-premise, obstruction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICLocalConfluenceFrontier}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.MetaCICLocalConfluenceFrontierUp.MetaCICLocalConfluenceFrontierUp`

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, full parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, decidable conversion, global normalization, kernel consistency, or standard bridges.

Upgrade path: Obligation closure requires checked carrier admission, critical-pair exposure, conditional-join exposure, residual-substitution premise retention, obstruction nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICLocalConfluenceFrontierUp.

Chapter 7347

A Concrete Naming Certificate for $\mathsf{SmallInductiveDimension}^{\uparrow}$

Following Hurewicz–Wallman and Engelking, the $\mathsf{SmallInductiveDimension}^{\uparrow}$ packet records the boundary-recursive small inductive dimension as finite BEDC separation and cover data. It sits beside the finite cover-order carrier of chapter 1853, the Lebesgue refinement route of chapter 4205, the topology source of chapter 88, the normal-space separation packet of chapter 1866, and the finite refinement carrier of chapter 3430. The packet names only displayed boundary neighborhoods, finite boundary-cover rows, refinement comparisons, transport, replay, provenance, and local naming; it does not assert a host inductive-dimension theorem, arbitrary open-cover compactness, selected separators outside the carrier, or ambient quotient topology equality.

Definition 7347.1 (Small inductive dimension carrier)**.** *A* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, N, B, R, F, L, H, C, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the finite point, open-neighborhood, closed-set, and boundary observations inspected by the packet.*
2. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *separation row, recording the closed-pair separation route used to choose a displayed neighborhood before its boundary is inspected.*
3. *B is the boundary-cover row, recording the finite* BHist *boundary entries whose own dimension read is one step lower than the surrounding packet.*
4. *R is the boundary-recursion ledger, recording that each boundary read is consumed only through the displayed topology and normal-space rows.*
5. *F is the* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *comparison row, recording the finite refinement handoff used when boundary covers are compared with cover-order packets.*
6. *L is the dimension-neighbor comparison row, recording the finite comparison with* $\mathsf{CoveringDimension}^{\uparrow}$ *and* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *without identifying the three dimension notions.*
7. *H stores componentwise* hsame *transport for topology, normal-space, boundary, recursion, refinement, comparison, replay, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from a topology-neighborhood request through normal-space separation, boundary cover inspection, finite refinement comparison, and dimension-neighbor handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{SmallInductiveDimension}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{LebesgueCoveringDimension}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *A is the local* $\mathsf{NameCert}_{\mathsf{SmallInductiveDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $X, N, B, R, F, L, H, C, P, A$. It has no coordinate for a host dimension theorem, arbitrary open-cover compactness, a selected separating family outside the packet, quotient-topology equality, countable choice, or an ambient boundary oracle.

Theorem 7347.2 (Small inductive dimension NameCert obligations)**.** *For every accepted* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *carrier*

$$S = (X, N, B, R, F, L, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{SmallInductiveDimension}^{\uparrow}}$ *surface consists exactly of topology-source admission through X, normal-space separation exposure through N, finite boundary-cover admission through B, boundary-recursion exactness through R, finite refinement comparison through F, covering-dimension and Lebesgue-covering comparison through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through A.*

Proof. Project the carrier of Definition 7347.1. The source, separation, boundary, recursion, refinement, comparison, transport, replay, provenance, and naming rows are precisely the displayed coordinates of the finite packet.

The boundary-recursive part is read from X, N, B, R. A consumer first sees the topology source, then the normal-space separation row, and only then the boundary-cover rows whose lower-dimensional reads are stored by the recursion ledger.

The finite comparison part is read from F, L. These rows permit comparison with finite cover-refinement, covering-dimension, and Lebesgue-covering packets, but they do not identify those packets with the small inductive dimension carrier.

Finally H, C, P, A transport, replay, source, and name the same finite route. Since none of the refused host objects is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7347.3 (Small inductive dimension boundary route)**.** *Every boundary-recursive read of an accepted* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *packet must factor through*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{NormalSpace}^{\uparrow},\ B,\ R,\ H,\ C,\ P,\ A$$

before a lower-dimensional boundary claim is consumed. The route exposes a displayed neighborhood-separation row and a finite boundary-cover row; it exports no selected separating family outside the carrier, no host boundary oracle, no quotient-topology equality, and no countable-choice schedule.

Proof. Begin with the topology source row X. It supplies the displayed point, open-neighborhood, closed-set, and boundary observations available to the packet.

Use the normal-space separation row N next. The boundary that enters the recursion ledger is admitted only after the separation route is visible, so a consumer cannot replace it by an unrecorded host separator.

The rows B and R then record the finite boundary-cover entries and their lower-dimensional reads. Structural rows H, C, P, A preserve, replay, source, and name this route, so the boundary claim remains inside the displayed packet. □

Theorem 7347.4 (Small inductive dimension cover comparison)**.** *Every finite-cover comparison exported by an accepted* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *packet must read the route*

$$\mathsf{FiniteCoverRefinement}^{\uparrow},\ \mathsf{CoveringDimension}^{\uparrow},\ \mathsf{LebesgueCoveringDimension}^{\uparrow} \longmapsto F, L, H, C, P, A.$$

The comparison is a finite handoff from boundary-recursive rows to cover-refinement and cover-order rows. It does not identify small inductive dimension with covering dimension, does not claim equality with Lebesgue covering dimension, and does not import arbitrary open-cover compactness or a host dimension theorem.

Proof. Start with the NameCert obligations of Definition 7347.2. They require the topology source, normal-space separation row, finite boundary-cover row, and recursion ledger before any comparison is exported.

Read the finite refinement row F. It supplies only the displayed refinement handoff, so cover comparison is routed through $\mathsf{FiniteCoverRefinement}^{\uparrow}$ rather than through arbitrary open-cover compactness.

Finally inspect the neighbor comparison row L. It permits cover-order and Lebesgue-covering consumers to see the same finite boundary-refinement route while keeping $\mathsf{SmallInductiveDimension}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, and $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ as distinct packet names. The structural rows H, C, P, A then transport, replay, source, and name the comparison without adding a host theorem. □

Falsifiable prediction: An accepted $\mathsf{SmallInductiveDimension}^{\uparrow}$ boundary read is refuted if it exports a lower-dimensional boundary claim without displaying the TopologyUp source row, NormalSpaceUp separation row, finite boundary-cover row, boundary-recursion ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: coveringdimension, lebesgue-covering-dimension, topology, normalspace, finite-cover-refinement: $\mathsf{SmallInductiveDimension}^{\uparrow}$ jointly carries topology-source, normal-space separation, finite boundary-cover, boundary-recursion, finite refinement comparison, dimension-neighbor comparison, transport, replay, provenance, and local naming rows in one packet; each named sibling lacks at least one of these local boundary-recursive coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{SmallInductiveDimension}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SmallInductiveDimension}^{\uparrow}$ packet is generated from a TopologyUp source row, a NormalSpaceUp separation row, finite BHist boundary-cover rows, a boundary-recursion ledger, a FiniteCoverRefinementUp comparison row, CoveringDimensionUp and LebesgueCoveringDimensionUp neighbor rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmallInductiveDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7347.1, Definition 7347.2, Definition 7347.3, and Definition 7347.4 seed the boundary-recursive small inductive dimension surface over topology, normal-space, finite-refinement, covering-dimension, and Lebesgue-covering rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SmallInductiveDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host inductive-dimension theorem, equality with covering dimension, equality with Lebesgue covering dimension, arbitrary open-cover compactness, selected separator choice, quotient topology equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topology-source admission, normal-space separation exactness, finite boundary-cover admission, boundary-recursion exactness, finite refinement comparison, cover-dimension comparison, transport, replay, provenance, and local NameCert rows for BEDC.Derived.SmallInductiveDimensionUp.

Chapter 7348

A Concrete Naming Certificate for $\mathsf{FiniteMetricProjection}^{\uparrow}$

Following Bishop and Bridges on finite metric spaces and Kelley on metric neighbourhoods, the $\mathsf{FiniteMetricProjection}^{\uparrow}$ packet records the finite nearest-point projection interface that is available when all candidate distances are displayed rows. It sits over the parent projection boundary of chapter 2356, the located-distance discipline of chapter 1541, the ambient metric source of chapter 57, the terminal distance seal of chapter 38, and the dyadic comparison core of chapter 291. The packet gives downstream located-set and completion consumers a finite endpoint witness without importing Hilbert projection, closed convexity, quotient subset equality, countable choice, or an ambient nearest-point selector.

Definition 7348.1 (FiniteMetricProjection carrier)**.** *A* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, L, D, E, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the finite metric source.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the displayed finite subset and its located-distance ledger.*
3. *D is the finite table of candidate distance rows from the queried point to the displayed subset.*
4. *E is the endpoint evidence row naming a candidate in the finite subset whose displayed distance is minimal among the rows in D.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger used to compare the finite distance rows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal read after the dyadic comparisons and endpoint evidence have been replayed.*
7. *H records componentwise* hsame *transport for the metric, located-set, distance-table, endpoint, dyadic-comparison, Real-seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a projection request through the finite distance table, dyadic comparisons, endpoint evidence, and terminal distance seal.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricProjection}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricProjection}^{\uparrow}}$ *row.*

The carrier contains no coordinate for Hilbert projection, closed convexity, subset quotient equality, selected minimizers beyond the displayed endpoint row, countable choice, compactness, or ambient completion.

Theorem 7348.2 (FiniteMetricProjection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *carrier* $F = (M, L, D, E, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricProjection}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the displayed metric, located-set, distance-table, endpoint, dyadic-comparison, Real-seal, transport, continuation, provenance, and local naming rows;*
2. *located-set exposure, requiring* L *to be read before any projection consumer may inspect the finite distance table* D*;*
3. *finite comparison exactness, requiring every minimality comparison for* E *to pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* Q*;*
4. *endpoint handoff, requiring the endpoint row* E *and terminal distance seal* R *to be replayed before a* $\mathsf{MetricProjection}^{\uparrow}$ *consumer receives a projection result;*
5. *ledger refusal, excluding Hilbert projection, closed-convex projection, quotient subset equality, hidden minimizer selection, countable choice, compactness, and ambient completion.*

No local NameCert row is stored outside those obligations and the structural rows H, C, P, N*.*

Proof. Project the carrier of Definition 7348.1. Its public rows are exactly $M, L, D, E, Q, R, H, C, P, N$, so carrier admission exposes only the finite metric source, located-set row, candidate distance table, endpoint witness, dyadic comparison ledger, and terminal distance seal.

The located-set and finite-comparison obligations follow from the displayed continuation. A projection request first reads M and L, then the finite table D, and then the dyadic comparison row Q that checks the endpoint row E against every displayed candidate.

The endpoint handoff is the remaining route. The row E is the only projection endpoint visible to the packet, and R is readable only after E, Q and the finite table have been replayed through H, C, P, N. Since the carrier has no further coordinate, none of the refused projection or choice principles can be projected from the NameCert surface. □

Theorem 7348.3 (FiniteMetricProjection nearest-row totality)**.** *Let* $F = (M, L, D, E, Q, R, H, C, P, N)$ *be an accepted* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *packet. Every projection request over the displayed finite subset reaches a nearest-distance row by the finite route*

$$M, L, D, Q \longmapsto E, R \longmapsto H, C, P, N.$$

Thus nearest-row totality is the finite exhaustion of the displayed distance table, with all comparisons read through $\mathsf{DyadicRatCore}^{\uparrow}$ *and the terminal distance exposed as a* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the finite distance table D in the carrier. Because D is a displayed finite row, the request inspects each candidate distance inside the packet rather than searching an ambient subset.

Use the dyadic comparison ledger Q. It compares the finite distance rows and identifies the endpoint evidence row E as minimal among the displayed candidates.

Finally replay the continuation row C and transport row H. The terminal seal R is read only after M, L, D, Q, E are visible, and provenance and local naming preserve that order. The route therefore gives nearest-row totality without adding a hidden selector, quotient equality, choice schedule, compactness theorem, or completion coordinate. □

Theorem 7348.4 (FiniteMetricProjection LocatedSet route)**.** *Every* $\mathsf{LocatedSet}^{\uparrow}$ *consumer that asks an accepted* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *packet for a projection endpoint must first expose the parent* $\mathsf{Met}^{\uparrow}$ *row* M*, the located-set row* L*, the finite distance table* D*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* Q*, the endpoint evidence row* E*, the* $\mathsf{Real}^{\uparrow}$ *seal* R*, and the structural rows* H, C, P, N*. The packet supplies the endpoint evidence required outside chapter 1541; it does not turn the located-set distance interface itself into a nearest-point selector.*

Proof. Read the LocatedSet metric-projection admission boundary in Definition 1541.6. It says that a projection consumer must supply endpoint evidence outside the located-set packet.

The carrier of Definition 7348.1 supplies exactly that outside row as E, together with the finite distance table D, dyadic comparison ledger Q, and Real distance seal R.

Combine this with Definition 7348.3. The endpoint is obtained only through the finite metric and located-set rows, the displayed distance table, and the dyadic comparison route, so the LocatedSet interface remains a distance interface while $\mathsf{FiniteMetricProjection}^{\uparrow}$ carries the separate endpoint witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricProjection}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7348.1, Definition 7348.2, Definition 7348.3, and Definition 7348.4 seed the finite projection route from displayed metric-distance rows to a separate endpoint witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetricProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hilbert projection, closed-convex projection, quotient subset equality, hidden minimizer selection, countable choice, compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite comparison exactness, endpoint handoff, located-set compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMetricProjectionUp.

Constructive story: A $\mathsf{FiniteMetricProjection}^{\uparrow}$ packet is generated from a Metric row, a LocatedSet row, a finite distance table, endpoint evidence, a DyadicRatCore comparison ledger, a Real distance seal, componentwise hsame transport, continuation replay, provenance, and a local NameCert row.

Chapter 7349

A Concrete Naming Certificate for $\mathsf{MilnorFibration}^{\uparrow}$

Following Milnor, the $\mathsf{MilnorFibration}^{\uparrow}$ packet records the finite fibration boundary visible near an isolated complex hypersurface singularity before any ambient analytic triviality theorem is imported. It sits after the charted source carrier of chapter 96, the local-trivialization rows of chapter 97, the circle boundary source of chapter 58, the horizontal-lift ledger of chapter 6815, and the cycle-read surface of chapter 98. The chapter names only displayed singular chart rows, punctured-neighbourhood rows, circle-base rows, fibre-bundle rows, connection-lift rows, homology readback, monodromy replay, provenance, transport, and local naming; it does not assert analytic Milnor existence, choose a global tube, identify fibres by host diffeomorphism, or import a Picard–Lefschetz formula.

Definition 7349.1 (Milnor fibration carrier)**.** *A* $\mathsf{MilnorFibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, U, S, F, B, E, H, L, T, R, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *charted source row carrying the inspected singular neighbourhood and its chart ledger.*
2. *U is the punctured-neighbourhood boundary row, naming the deleted singular point and the finite boundary window used by the read.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *base-circle row through which phase or argument reads are exposed.*
4. *F is the fibre row over the displayed base-circle observation, with no host fibre quotient or selected representative.*
5. *B is the* $\mathsf{Bundle}^{\uparrow}$ *local-trivialization row for the displayed map over the visible circle window.*
6. *E is the* $\mathsf{EhresmannConnection}^{\uparrow}$ *horizontal-lift row used only after the bundle and base-circle rows have been read.*
7. *H is the* $\mathsf{Homology}^{\uparrow}$ *cycle-read row on the displayed fibre boundary.*
8. *L is the monodromy ledger, recording the finite continuation route from one circle read to the next through E and H.*
9. *T records componentwise* hsame *transport for X, U, S, F, B, E, H, L.*

10. *R records displayed* Cont *replay for the fibration, lift, cycle, and monodromy reads.*
11. *P is the* Pkg *provenance over* $\mathsf{MilnorFibration}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{EhresmannConnection}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MilnorFibration}^{\uparrow}}$ *row.*

The carrier has no coordinate for analytic existence of the Milnor tube, host smooth-fibre equality, selected global trivialization, quotient vanishing-cycle object, or Picard–Lefschetz transformation.

Theorem 7349.2 (Milnor fibration NameCert obligations)**.** *For an accepted* $\mathsf{MilnorFibration}^{\uparrow}$ *packet $M = (X, U, S, F, B, E, H, L, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MilnorFibration}^{\uparrow}}$ *surface is exhausted by carrier admission for M, manifold-source exposure through X, U, circle-base exposure through S, fibre and local-trivialization exposure through F, B, horizontal-lift exposure through E, homology-cycle exposure through H, monodromy-ledger exposure through L, componentwise transport through T, displayed replay through R, provenance through P, and local naming through N. A consumer cannot inspect a monodromy or vanishing-cycle boundary before those rows have been displayed.*

Proof. Project the accepted packet to the carrier of Definition 7349.1. Its coordinates list exactly the manifold, punctured-neighbourhood, circle-base, fibre, bundle, connection, homology, monodromy, transport, replay, provenance, and naming rows.

The NameCert surface is therefore the local package over those displayed rows. The bundle row cannot be read before the manifold and circle rows, the connection row cannot be read before the bundle row, and the homology row cannot be consumed by monodromy before the fibre boundary has been displayed.

Since the carrier contains no host analytic tube, no global trivialization coordinate, and no quotient vanishing-cycle object, any accepted downstream monodromy read is exhausted by the listed rows. □

Theorem 7349.3 (Milnor fibration monodromy route)**.** *Every monodromy-facing consumer of an accepted* $\mathsf{MilnorFibration}^{\uparrow}$ *packet must read the dependency route*

$$\mathsf{Manifold}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Bundle}^{\uparrow} \longmapsto \mathsf{EhresmannConnection}^{\uparrow} \longmapsto \mathsf{Homology}^{\uparrow}.$$

The manifold and punctured-neighbourhood rows delimit the inspected singular boundary, the circle row supplies the base loop, the bundle row supplies the local fibre handoff, the Ehresmann row supplies the finite horizontal-lift replay, and the homology row supplies the cycle endpoint read by the monodromy ledger. The route exports no arbitrary analytic fibration theorem, no selected fibre equivalence, and no Picard–Lefschetz formula.

Proof. Begin with the source boundary. The rows X and U expose the charted neighbourhood and deleted-point boundary before any circle or fibre read is available.

Next read the base and bundle rows. The $\mathsf{S}^{1\uparrow}$ row S supplies the visible loop coordinate, and the $\mathsf{Bundle}^{\uparrow}$ row B supplies only the local fibre handoff over that displayed loop.

Finally route continuation through E, H, L. The $\mathsf{EhresmannConnection}^{\uparrow}$ row gives the horizontal-lift replay, the $\mathsf{Homology}^{\uparrow}$ row gives the cycle endpoint on the displayed fibre, and the monodromy ledger records only this finite continuation. Therefore the consumer sees exactly the stated dependency route and none of the excluded host analytic data. □

Falsifiable prediction: A monodromy consumer that reads a $\mathsf{MilnorFibration}^{\uparrow}$ packet without exposing the $\mathsf{Manifold}^{\uparrow}$ source, punctured-neighbourhood boundary, $\mathsf{S}^{1\uparrow}$ base loop, $\mathsf{Bundle}^{\uparrow}$ fibre

handoff, $\mathsf{EhresmannConnection}^{\uparrow}$ lift row, $\mathsf{Homology}^{\uparrow}$ cycle row, hsame, Cont, Pkg, and NameCert rows refutes the carrier.

Independence witness: manifold, bundle, s1, ehresmann-connection, homology: $\mathsf{MilnorFibration}^{\uparrow}$ binds singular-neighbourhood source rows, circle-base rows, local fibre handoff, horizontal-lift replay, homology cycle readback, and monodromy provenance in one local NameCert packet; each named sibling lacks at least one of these rows as a Milnor-fibration obligation.

Closure status for $\mathsf{NameCert}_{\mathsf{MilnorFibration}^{\uparrow}}$.

Constructive story: A $\mathsf{MilnorFibration}^{\uparrow}$ packet is generated from a charted singular-neighbourhood row, punctured-boundary row, circle-base row, fibre and bundle rows, Ehresmann horizontal-lift row, homology cycle row, monodromy ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MilnorFibration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7349.1, Definition 7349.2, and Definition 7349.3 seed the finite Milnor-fibration NameCert surface over $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{EhresmannConnection}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MilnorFibration}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MilnorFibrationUp`

Bridge status: none

Not claimed: This chapter does not close analytic Milnor-fibration existence, selected global tubes, host smooth-fibre equality, Picard–Lefschetz formulas, Lean NameCert verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, singular-boundary exposure, circle-base exposure, bundle-fibre handoff, horizontal-lift replay, homology-cycle readback, monodromy-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.MilnorFibrationUp.

Chapter 7350

A Concrete Naming Certificate for $\mathsf{BoundedConvergence}^{\uparrow}$

Following Bishop–Bridges constructive real analysis and the Lebesgue bounded-convergence theorem, the $\mathsf{BoundedConvergence}^{\uparrow}$ packet records only the finite-window BEDC surface needed when a uniformly bounded real sequence has displayed pointwise convergence windows. It sits over the real-name seal of chapter 38, the regular rational readback of chapter 301, and the bounded real sequence surface of chapter 1415. The packet is a finite observation and averaging ledger for a bounded-convergence consumer; it does not assert a general measure-theoretic dominated-convergence theorem, choose pointwise limits, quotient functions, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7350.1 (Bounded convergence carrier)**.** *A* $\mathsf{BoundedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, L, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row, carrying the bounded real observation family inspected by the packet.*
2. *W is the finite observation-window ledger, listing the indices and sample names available to the current tolerance request.*
3. *L is the pointwise convergence-window row, recording finite evidence that the displayed samples eventually enter the requested rational tolerance window.*
4. *M is the finite measure or finite averaging ledger, naming the finite weights or finite sample average through which the consumer reads the bounded source.*
5. *A is the bounded-average comparison row, certifying that bounded tails outside the active window remain inside the displayed bound budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows produced by W, L, M, A.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has been displayed.*
8. *H records componentwise* hsame *transport for the bounded source, windows, convergence ledger, finite measure or averaging row, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through the same finite-window route.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedConvergence}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host almost-everywhere relation, a selected limiting function, countable choice, quotient equality of functions, arbitrary dominated convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7350.2 (Bounded convergence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedConvergence}^{\uparrow}$ *carrier $B = (S, W, L, M, A, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedConvergence}^{\uparrow}}$ *surface consists exactly of bounded source admission through S, finite observation-window coverage through W, pointwise convergence-window exposure through L, finite measure or finite averaging admission through M, bounded-average comparison through A, regular rational readback through R,* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7350.1. The analytic rows are the bounded source S, the finite windows W, the pointwise convergence ledger L, the finite measure or averaging row M, and the bounded-average comparison row A.

The readback rows come next. The regular rational row R records the finite rational comparison assembled from those windows, and the real seal E is downstream of that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate names an external limit selector, arbitrary dominated convergence theorem, quotient function equality, countable choice, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 7350.3 (Bounded convergence finite-measure handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedConvergence}^{\uparrow}$ *packet reaches the convergence readback only through the finite route*

$$\mathsf{BoundedRealSequence}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The finite measure or finite averaging ledger M and bounded-average comparison row A must be read before R or E is public, and the consumer receives no general dominated-convergence theorem, selected pointwise limit, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with the accepted carrier $B = (S, W, L, M, A, R, E, H, C, P, N)$. The bounded source row S and finite windows W identify the only observations available to the packet.

Read L, M, A before the readback rows. The pointwise convergence windows, finite measure or averaging ledger, and bounded comparison row are the data that justify the finite regular-rational readback.

The row R is then the only $\mathsf{RegSeqRat}^{\uparrow}$ surface exposed to consumers, and the real seal E is downstream of R. Transport and replay through H, C, P, N preserve that route without adding a selected limit or an external convergence principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedConvergence}^{\uparrow}$ packet is generated from a bounded real sequence row, finite observation windows, pointwise convergence windows, a finite measure or averaging ledger, bounded-average comparison, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 7350.1, Definition 7350.2, and Definition 7350.3 seed the finite-window bounded-convergence packet over RealUp and RegSeqRat readback rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BoundedConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general measure-theoretic dominated convergence, selected pointwise limits, quotient function equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window convergence exposure, finite measure or averaging exactness, bounded comparison, RegSeqRat readback, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.BoundedConvergenceUp.

Chapter 7351

A Concrete Naming Certificate for $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$

Following the compact-uniform-continuity construction of chapter 9237, the $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$ packet records the finite cover handoff that turns pointwise modulus rows into a tolerance-only uniform-modulus export. It sits over the compact finite-cover surface of chapter 60, the metric source rows of chapter 57, the continuous-map pointwise rows of chapter 130, the rational lower-bound readback of chapter 37, and the downstream uniform-modulus consumer of chapter 369. The packet realizes the subordinate finite cover and pointwise-modulus handoff as finite Compact, ContinuousMap, RatUp, and UniformModulus NameCert rows; it does not export open-cover compactness, an external modulus selector, host minima, quotient point equality, or ambient real analysis.

Definition 7351.1 (Subordinate modulus uniform handoff carrier)**.** *A* $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, X, F, R, Q, V, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-cover row, exposing the finite center or cover spine inspected by the handoff.*
2. *X is the* $\mathsf{Met}^{\uparrow}$ *source-distance row, recording the finite distance comparisons used by the cover.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row, assigning local precision windows to the displayed compact-cover centers.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *subordinate radius row, recording positive rational radii below the local pointwise-modulus budgets.*
5. *Q is the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, producing a single tolerance-only precision from the displayed subordinate radii.*
6. *V is the* $\mathsf{UniformModulus}^{\uparrow}$ *export row, public only after K, X, F, R, Q have been read.*
7. *H records componentwise* hsame *transport for compact-cover, metric, pointwise-modulus, rational-fold, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through compact cover, pointwise modulus, rational fold, and uniform-modulus export.*

9. *P is the* Pkg *provenance over* $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite subcover, host Lebesgue-number minimum, external uniform-modulus function, quotient point equality, countable choice, Lean verification, or bridge credentials.

Theorem 7351.2 (Subordinate modulus uniform handoff NameCert obligations)**.** *For every accepted* $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$ *carrier* $U = (K, X, F, R, Q, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}}$ *surface consists exactly of compact finite-cover admission through K, metric comparison admission through X, continuous pointwise-modulus admission through F, subordinate positive-radius exposure through R, finite rational lower-bound folding through Q, uniform-modulus export through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7351.1. The compact-cover row K and metric row X provide the finite source geometry available to the handoff.

Read the pointwise-modulus row F and subordinate radius row R. Each radius is attached to a displayed cover center and is certified below the corresponding pointwise precision budget.

The fold row Q is a finite $\mathsf{Rat}^{\uparrow}$ lower-bound computation over those radii. The export row V may be read only after that fold has produced a tolerance-only precision for the uniform-modulus consumer.

Finally H, C, P, N transport, replay, source, and name the same finite route. Because no carrier coordinate names an infinite subcover, host minimum, external modulus selector, quotient equality, countable choice, Lean verification, or bridge credentials, the listed rows exhaust the local NameCert obligations. □

Theorem 7351.3 (Subordinate modulus uniform handoff uniform-modulus export)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$ *packet must first read the compact finite-cover row, metric distance comparisons, continuous-map pointwise-modulus rows, positive* $\mathsf{Rat}^{\uparrow}$ *subordinate radii, and finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold. Only then may the consumer read the export row V as a tolerance-only uniform-modulus handoff.*

Proof. Start with the accepted packet $U = (K, X, F, R, Q, V, H, C, P, N)$. The compact and metric rows K, X supply the finite cover and distance data; no open-cover theorem or point quotient is available.

Read F, R, Q in order. The pointwise-modulus row attaches a local precision to each displayed cover center, the subordinate radius row keeps that radius below the local budget, and the rational fold converts the finite family of radii into one precision history.

The uniform-modulus export V is downstream of that fold. Transport and replay through H, C, P, N preserve the same finite route, so the consumer receives a $\mathsf{UniformModulus}^{\uparrow}$ handoff rather than a host modulus selector or ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}$ packet is generated from compact finite-cover rows, metric distance rows, continuous-map pointwise modulus rows, RatUp subordinate radii, a RatUp finite lower-bound fold, a UniformModulus export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubordinateModulusUniformHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 7351.1, Definition 7351.2, and Definition 7351.3 seed the finite compact-cover handoff from pointwise moduli to a UniformModulus export.

Formal verification: FormalStatus(NameCert$_{\text{SubordinateModulusUniformHandoff}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite subcover selection, host Lebesgue-number minima, external modulus selectors, quotient point equality, ambient real analysis, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover exposure, metric comparison, pointwise-modulus admission, subordinate radius exactness, RatUp lower-bound folding, UniformModulus export, transport, replay, provenance, and local naming for BEDC.Derived.SubordinateModulusUniformHandoffUp.

Chapter 7352

A Concrete Naming Certificate for $\mathsf{RegularCauchyEmbedding}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ packet records the finite constant-entry route by which a displayed dyadic or rational point is read as a regular Cauchy rational name and then sealed for Real-facing consumers. It sits after the dyadic core of chapter 291, the rational carrier of chapter 37, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dyadic embedding surface of chapter 467. The chapter names one L10 constant embedding boundary; it does not identify quotient streams, choose limits, assert trichotomy, use countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7352.1 (Regular Cauchy embedding carrier)**.** *A* $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, naming the dyadic endpoint when the embedding starts from dyadic data.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row, naming the rational endpoint when the same route is read through rational data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *constant-window row, recording that every inspected finite window returns the displayed entry supplied by D or Q.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying the stationary regular Cauchy checks for the constant windows in W.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source, window, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for the source rows, constant-window row, regular readback row, real-seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from the dyadic or rational source through the constant windows, regular readback, and terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, Q, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, trichotomy of rationals, countable-choice schedules, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7352.2 (Regular Cauchy embedding NameCert obligations)**.** *For every accepted* RegularCauchyEmbeddir *carrier $K = (D, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEmbedding}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed packet K;*
2. *constant-window exactness for the* $\mathsf{StreamName}^{\uparrow}$ *row W over the displayed source row D or Q;*
3. *regularity handoff from W to the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R;*
4. *real-seal nonescape for the terminal* $\mathsf{Real}^{\uparrow}$ *row E;*
5. *classifier transport through H;*
6. *displayed replay through C;*
7. *provenance and local naming through P and N.*

Proof. Project the carrier of Definition 7352.1. Its analytic rows are precisely the dyadic source D, rational source Q, constant-window row W, regular readback row R, and real-seal row E.

The constant-window obligation is local to W. Every finite observation named by W returns the same displayed source entry, so the regularity handoff can read R as the stationary $\mathsf{RegSeqRat}^{\uparrow}$ witness attached to that entry.

The terminal seal is downstream of those rows. A Real-facing read reaches E only after the source, window, and regularity rows have been displayed, and the transport, replay, provenance, and naming rows H, C, P, N preserve that same route.

No remaining coordinate exists in the carrier. Hence quotient stream equality, selected limits, trichotomy, countable choice, host real equality, and ambient completeness cannot supply further local obligations, and the seven listed groups exhaust the NameCert surface. □

Theorem 7352.3 (Regular Cauchy embedding constant handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*- or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ *carrier receives only the displayed constant-entry route*

$$D, Q \longmapsto W \longmapsto R \longmapsto E$$

preserved by H, C, P, N. In particular, the consumer cannot replace the route by quotient stream equality, a selected limit, a trichotomy branch not recorded in the source rows, a countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 7352.2. A consumer must first read carrier admission and the displayed dyadic or rational source row before the constant-window row is meaningful.

The StreamName handoff is constant. Since W repeats the displayed entry on each inspected finite window, the regular readback R can only be consumed as the stationary $\mathsf{RegSeqRat}^{\uparrow}$ row attached to those windows.

The Real seal is last. The row E is exposed after the constant-window and regularity rows, while H, C, P, N transport, replay, source, and name exactly that finite route.

Therefore every RegSeqRat- or Real-facing read factors through $D, Q, W, R, E, H, C, P, N$. The carrier contains no separate coordinate for quotient stream equality, selected limits, trichotomy,

countable-choice schedules, host equality, or ambient completeness, so none of those objects can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ consumer must expose the dyadic or rational source row, the constant StreamName window row, the stationary RegSeqRat readback row, the Real seal row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any RegSeqRat- or Real-facing constant-entry read is available.

Independence witness: Nearest siblings: DyadicRatCore, Rat, StreamName, RegSeqRat, Real, and DyadicEmbedding. $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ is not bijective to any one of them because it jointly carries source rows, constant windows, stationary regular readback, a terminal seal, transport, replay, provenance, and local naming for the same constant-entry embedding route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEmbedding}^{\uparrow}$ packet is generated from displayed DyadicRatCore or Rat source rows, constant StreamName windows, stationary RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7352.1, Definition 7352.2, and Definition 7352.3 seed the finite constant-entry route from DyadicRatCore or Rat data into RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constant-window exactness, regularity handoff, Real-seal nonescape, classifier transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEmbeddingUp.

Chapter 7353

A Concrete Naming Certificate for $\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}$

Following Dunford and Pettis, with the weak-topology and Banach-space background read through Bourbaki and Schaefer, the $\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}$ packet records the finite BEDC window in which a Dunford–Pettis handoff can be inspected as weak-compactness evidence without importing the full compactness theorem. It sits between the Dunford–Pettis route of chapter 6346, the weak compactness surface of chapter 6851, the weak-topology probe window of chapter 3597, the Banach source of chapter 90, and the scalar readback discipline of chapter 38. The chapter names only a finite window object: it does not claim Banach–Alaoglu, full weak compactness, selected weakly convergent subsequences, quotient function-space equality, or choice.

Definition 7353.1 (Dunford–Pettis finite weak-compactness carrier)**.** *A* $\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, K, T, B, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DunfordPettis}^{\uparrow}$ *source row, exposing only the already displayed uniform-integrability, weak-test, bounded-probe, scalar-budget, handoff, transport, replay, provenance, and local naming rows.*
2. *K is the* $\mathsf{WeakCompactness}^{\uparrow}$ *request row, read as a finite compactness-facing interface rather than as a completed weak-compactness theorem.*
3. *T is the* $\mathsf{WeakTopology}^{\uparrow}$ *probe-window row, listing the finitely many weak probes and tolerances visible in the present handoff.*
4. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row, naming the ambient linear and norm structure only through displayed finite functional-analysis data.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for probe values, bounds, tolerances, and comparison ledgers.*
6. *A is the finite acceptance-window row. It records that the Dunford–Pettis handoff, weak-compactness request, weak-topology probes, Banach source, and scalar readback have been displayed in the same packet before a consumer can read weak-compactness evidence.*
7. *H records componentwise* hsame *transport for D, K, T, B, R, A, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$D, T, B, R \longmapsto A \longmapsto K, H, P, N.$$

9. *P is the* Pkg *provenance over* DunfordPettisFiniteWeakCompactness$^{\uparrow}$, DunfordPettis$^{\uparrow}$, WeakCompactness$^{\uparrow}$, WeakTopology$^{\uparrow}$, Banach$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{DunfordPettisFiniteWeakCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, K, T, B, R, A, H, C, P, N$. It has no coordinate for Banach–Alaoglu, a full weak-compactness theorem, selected weakly convergent subsequences, quotient function-space equality, an ambient dual-ball compactness theorem, countable choice, or an unrestricted weak-probe family.

Theorem 7353.2 (Dunford–Pettis finite weak-compactness obligations)**.** *For every accepted* DunfordPettisFiniteWe *carrier*

$$W = (D, K, T, B, R, A, H, C, P, N),$$

the local NameCert$_{\text{DunfordPettisFiniteWeakCompactness}^{\uparrow}}$ *surface consists exactly of Dunford–Pettis source admission through D, weak-compactness request admission through K, finite weak-topology probe admission through T, Banach source admission through B, scalar readback through R, finite acceptance-window exactness through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Every consumer must expose D, T, B, R, A before reading K as weak-compactness evidence. The row K is therefore a finite window handoff, not Banach–Alaoglu, full weak compactness, a selected weakly convergent subsequence, quotient function-space equality, or a choice principle.

Proof. Project the carrier of Definition 7353.1. Its public rows are exactly $D, K, T, B, R, A, H, C, P, N$, so the NameCert surface can expose only the Dunford–Pettis source, weak-compactness request, weak-topology probe window, Banach source, scalar readback, finite acceptance window, and structural rows.

The window exactness is forced by the order of the replay row. A consumer first reads the Dunford–Pettis source D, the weak-topology probes T, the Banach source B, and the scalar readback R; only then does the acceptance-window row A allow the weak-compactness request K to be inspected.

Transport and provenance do not add compactness data. The rows H, C, P, N move, replay, source, and name the same finite route, while the classifier has no coordinate for Banach–Alaoglu, selected subsequences, quotient function-space equality, unrestricted weak compactness, or choice. □

Theorem 7353.3 (Dunford–Pettis finite weak-compactness window handoff)**.** *Every weak-compactness-facing consumer of an accepted* DunfordPettisFiniteWeakCompactness$^{\uparrow}$ *packet must pass through the finite route*

$$\text{DunfordPettis}^{\uparrow}, \text{WeakTopology}^{\uparrow}, \text{Banach}^{\uparrow}, \text{Real}^{\uparrow} \longmapsto A \longmapsto \text{WeakCompactness}^{\uparrow}.$$

The handoff may feed the finite weak-compactness interface of chapter 6851, but it exports no Banach–Alaoglu theorem, full weak compactness, selected weakly convergent subsequence, quotient function-space equality, countable-choice schedule, or ambient dual-ball compactness theorem.

Proof. Begin with the Dunford–Pettis source row D. It carries only the finite handoff described in chapter 6346, so the consumer has not obtained a selected subsequence or ambient compactness theorem.

Next read the weak-topology, Banach, and Real rows T, B, R. These rows expose the finite probe window, source structure, scalar values, bounds, and tolerances needed for the present request.

The acceptance-window row A then binds those reads to the weak-compactness request K. Since A is a displayed finite row, it can hand the request to the sibling weak-compactness surface without promoting the route to Banach–Alaoglu, full weak compactness, quotient equality, or choice.

Finally the structural rows H, C, P, N transport, replay, source, and name the same route. No additional carrier coordinate remains through which a consumer could extract a selected weakly convergent subsequence, an ambient dual-ball compactness theorem, or unrestricted function-space equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7353.1, Definition 7353.2, and Definition 7353.3 seed the finite window from Dunford–Pettis source data through weak-topology probes, Banach source rows, scalar readback, acceptance-window exactness, and weak-compactness request rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Banach–Alaoglu, full weak compactness, selected weakly convergent subsequences, quotient function-space equality, countable choice, ambient dual-ball compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-window admission, Banach source compatibility, scalar readback, acceptance-window exactness, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DunfordPettisFiniteWeakCompactnessUp.

Constructive story: A $\mathsf{DunfordPettisFiniteWeakCompactness}^{\uparrow}$ packet is generated from a DunfordPettis source row, a WeakCompactness request row, a WeakTopology finite probe window, a Banach source row, Real scalar readback, an acceptance-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7354

A Concrete Naming Certificate for $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ packet records the finite away-from-zero modulus data needed to read a reciprocal of a regular Cauchy rational name. It sits after the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located positive lower-bound surface of chapter 2029, the terminal real-name seal of chapter 38, and the reciprocal handoff of chapter 6251. The packet names the local modulus route for a source already displayed away from zero; it is not RealUp completeness, field completeness, quotient equality of representatives, or a selector for inverses.

Definition 7354.1 (Regular Cauchy reciprocal modulus carrier)**.** *A* $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, A, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, displaying the regular rational name whose reciprocal modulus is being read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested precision, dyadic comparison budget, and denominator-safety scale.*
3. *A is the located positive away-from-zero witness, read through* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *as a positive lower bound for the source seal.*
4. *K is the reciprocal modulus calculation row, turning the lower-bound witness and the requested tolerance into the finite denominator budget used by the reciprocal readback.*
5. *R is the* $\mathsf{RealReciprocal}^{\uparrow}$ *handoff row, available only after S, D, A, K have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the reciprocal name.*
7. *H records componentwise* hsame *transport for source, tolerance, located gap, modulus calculation, reciprocal handoff, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, D, A, K \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, D, A, K, R, E, H, C, P, N$. It has no coordinate for RealUp completeness, field completeness, quotient representatives, selected inverses, arbitrary trichotomy, or countable choice.

Lean4 (statement) `BEDC.Derived.RegularCauchyReciprocalModulusUp`

Theorem 7354.2 (Regular Cauchy reciprocal modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ *carrier $M = (S, D, A, K, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReciproca}}$ *surface consists exactly of carrier admission;* $\mathsf{RegSeqRat}^{\uparrow}$ *source visibility through S;* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness through D; away-from-zero located positivity through A; reciprocal modulus calculation through K;* $\mathsf{RealReciprocal}^{\uparrow}$ *handoff through R;* $\mathsf{Real}^{\uparrow}$ *seal handoff through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7354.1. The source row S, dyadic tolerance ledger D, and located positive row A are the complete finite input surface for the reciprocal modulus request.

Read the calculation row next. The row K is downstream of the displayed tolerance and away-from-zero data, so the reciprocal handoff R can be consumed only after the denominator-safety budget has been named.

Finally the seal row E and the structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded completeness theorem, quotient representative, selected inverse, trichotomy, or choice row occurs in the carrier, every accepted obligation stays inside the listed NameCert surface. □

Theorem 7354.3 (Regular Cauchy reciprocal modulus Real seal boundary)**.** *Every Real-facing reciprocal read of an accepted* $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. The route exports no RealUp completeness, field completeness, quotient representative equality, selected inverse, arbitrary trichotomy, or countable choice.*

Proof. Apply Definition 7354.2. It identifies the source row S, dyadic ledger D, located positive row A, and reciprocal modulus calculation row K as the only finite input surface for the reciprocal read.

The $\mathsf{RealReciprocal}^{\uparrow}$ handoff row R is therefore available only through that displayed input surface. A Real-facing consumer reaches the seal row E only after the reciprocal handoff has been exposed.

The transport, replay, provenance, and local naming rows preserve the same dependency order. No carrier row supplies completeness, quotient representatives, selected inverses, trichotomy, or countable choice, so the terminal handoff remains a finite seal rather than an ambient field operation. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ packet is generated from a RegSeqRat source, a DyadicRatCore tolerance ledger, a located positive away-from-zero witness, a reciprocal modulus calculation row, a RealReciprocal handoff, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7354.1, Definition 7354.2, and Definition 7354.3 seed the finite Bishop-style away-from-zero reciprocal-modulus route from regular Cauchy evidence to the RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyReciprocalModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, field completeness, quotient representatives, selected inverses, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, located positive away-from-zero input, reciprocal modulus calculation, RealReciprocal handoff, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyReciprocalModulusUp.

Chapter 7355

A Concrete Naming Certificate for $\mathsf{EgyptianFraction}^{\uparrow}$

Following Hardy–Wright style unit-fraction decomposition, the $\mathsf{EgyptianFraction}^{\uparrow}$ packet records the finite greedy route by which a positive rational window is decomposed into displayed unit-denominator rows before it is consumed by real-approximation data. It sits after the rational carrier of chapter 37, the dyadic tolerance core of chapter 291, the pointwise rational schedule of chapter 350, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only a finite decomposition packet; it does not assert host rational trichotomy, select an infinite expansion, quotient real names, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7355.1 (Egyptian fraction carrier)**.** *An* $\mathsf{EgyptianFraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, R, D, G, T, S, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the initial positive* $\mathsf{Rat}^{\uparrow}$ *source row to be decomposed.*
2. *$R = (r_0, \ldots, r_m)$ is the residual rational row, with $r_0 = Q$ and each displayed residual read through* $\mathsf{Rat}^{\uparrow}$*.*
3. *$D = (d_1, \ldots, d_m)$ is the finite unit-denominator list, recording the unit rows $1/d_i$ consumed by the decomposition.*
4. *G is the greedy-step ledger. Its ith row records that d_i is a displayed positive denominator with $1/d_i \leq r_{i-1}$ and that the next residual is $r_i = r_{i-1} - 1/d_i$.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-budget row, recording dyadic windows for the residuals and for the finite partial sums $\sum_{j \leq i} 1/d_j$.*
6. *S is the* $\mathsf{RationalStream}^{\uparrow}$ *handoff row, exposing the finite partial-sum schedule as rational observations.*
7. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, reading the same finite schedule as a regular rational approximation window for downstream Real and Completion consumers.*
8. *H records componentwise* hsame *transport for Q, R, D, G, T, S, A and the local naming rows.*

9. *C records displayed* Cont *replay from the initial rational row through the greedy ledger, dyadic budget, rational-stream handoff, and regular-rational readback.*
10. *P is the* Pkg *provenance over* $\mathsf{EgyptianFraction}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EgyptianFraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed finite rows $Q, R, D, G, T, S, A, H, C, P, N$. It has no coordinate for an infinite unit-fraction tail, a chosen host expansion, quotient equality of real names, rational trichotomy beyond the displayed comparisons, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7355.2 (Egyptian fraction greedy step)**.** *For an accepted* $\mathsf{EgyptianFraction}^{\uparrow}$ *carrier $E = (Q, R, D, G, T, S, A, H, C, P, N)$, every displayed greedy row in G decreases the residual inside the rational-and-dyadic surface. More precisely, if the ith ledger row records $1/d_i \le r_{i-1}$ and $r_i = r_{i-1} - 1/d_i$ with $r_{i-1} > 0$, then the next residual row satisfies*

$$0 \le r_i < r_{i-1}.$$

The comparison is read only through the $\mathsf{Rat}^{\uparrow}$ *residual rows and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row T.*

Proof. Project the carrier of Definition 7355.1. The relevant data for one step are the residual pair r_{i-1}, r_i, the denominator row d_i, the ledger equation in G, and the dyadic comparison budget in T.

The lower bound is ledger exactness. Since G records $1/d_i \le r_{i-1}$ and $r_i = r_{i-1} - 1/d_i$, the displayed $\mathsf{Rat}^{\uparrow}$ subtraction row presents r_i as a nonnegative residual.

The strict upper bound uses only the positive unit row. The same ledger records a positive denominator, so the unit row $1/d_i$ is a positive rational. Subtracting a displayed positive rational from the positive residual gives $r_i < r_{i-1}$, and the dyadic budget row T supplies the finite comparison window used by downstream consumers. □

Theorem 7355.3 (Egyptian fraction NameCert obligations)**.** *For every accepted* $\mathsf{EgyptianFraction}^{\uparrow}$ *carrier $E = (Q, R, D, G, T, S, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EgyptianFraction}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for the displayed packet E;*
2. *residual exactness for the* $\mathsf{Rat}^{\uparrow}$ *rows Q, R and the unit-denominator list D;*
3. *greedy-step exactness for G, including the residual decrease of Definition 7355.2;*
4. *approximation handoff through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row T, the* $\mathsf{RationalStream}^{\uparrow}$ *schedule row S, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row A;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Proof. Carrier admission is projection of the finite tuple E. The rational source, residual list, denominator list, greedy ledger, dyadic budget, rational-stream handoff, regular readback, transport, replay, provenance, and naming rows are all coordinates of the same BHist packet.

Residual exactness and greedy-step exactness are local to Q, R, D, G. Each accepted step stores the unit denominator and the resulting residual equation, while Definition 7355.2 supplies the displayed decrease used to keep the finite ledger ordered.

The approximation handoff is downstream of the greedy rows. The dyadic budget row T bounds the finite partial sums, S exposes those sums as rational-stream observations, and A reads the same schedule as a regular rational approximation row. The remaining rows H, C, P, N transport and name exactly this route, so the five listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{EgyptianFraction}^{\uparrow}$ consumer must expose the initial Rat row, residual rational row, unit-denominator list, greedy-step ledger, dyadic error budget, RationalStream handoff, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Real-facing rational approximation read is available.

Independence witness: Nearest siblings: Rat, DyadicRatCore, RationalStream, RegSeqRat, and Real. $\mathsf{EgyptianFraction}^{\uparrow}$ is not bijective to any one of them because it binds a finite unit-denominator decomposition ledger to dyadic budget, rational-stream, and regular-rational readback rows in one local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EgyptianFraction}^{\uparrow}}$.

Constructive story: An $\mathsf{EgyptianFraction}^{\uparrow}$ packet is generated from a positive Rat source row, residual rational rows, a finite unit-denominator list, a greedy-step ledger, dyadic error budgets, RationalStream handoff, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EgyptianFraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7355.1, Definition 7355.2, and Definition 7355.3 seed the finite unit-fraction decomposition surface for L10 rational approximation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EgyptianFraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close existence of a complete expansion for arbitrary rationals, uniqueness of decomposition, host rational trichotomy, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual exactness, greedy-step decrease, dyadic budget handoff, RationalStream handoff, RegSeqRat readback, transport, replay, provenance, and local naming for BEDC.Derived.EgyptianFractionUp.

Chapter 7356

A Concrete Naming Certificate for $\mathsf{CompactMetricEquicontinuity}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactMetricEquicontinuity}^{\uparrow}$ packet records the finite BEDC carrier that connects a compact metric source, an equicontinuity-family modulus ledger, and the finite-modulus extraction interface used by Arzela–Ascoli. It sits after the compact finite-net source route of chapter 6598, the family-radius ledger of chapter 413, and the extraction consumer of chapter 3568. The packet names the finite-net handoff only; it does not import open-cover compactness, a choice-selected center family, host function-space compactness, selected subsequences, quotient equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7356.1 (CompactMetricEquicontinuity carrier)**.** *A* $\mathsf{CompactMetricEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, Y, F, R, M, V, S, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, read through finite net centers, cover radii, and located membership witnesses;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance comparisons are used by the family ledger;*
3. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row over the source and target rows;*
4. *R records pointwise continuity radius rows for the displayed family members and source centers;*
5. *M is the shared equicontinuity modulus candidate read through* $\mathsf{EquicontinuityFamily}^{\uparrow}$*;*
6. *V is the finite-net cover ledger that records which compact-source centers and radii are consumed by the handoff;*
7. *S is the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *handoff row that receives the compact finite-net and shared-modulus data;*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *tolerance seal for the displayed target-distance budget;*
9. *H records componentwise* hsame *transport for the compact source, target metric, family rows, radius rows, modulus candidate, cover ledger, handoff row, tolerance seal, replay, provenance, and local naming;*

10. *C records displayed* Cont *replay from a requested tolerance through $K, Y, F, R, M, V, A, H, P, N$ to the finite-modulus handoff S;*
11. *P is the* Pkg *provenance over* $\mathsf{CompactMetricEquicontinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricEquicontinuity}^{\uparrow}}$ *row.*

The carrier exports only these finite rows. It contains no arbitrary open-cover compactness proof, no selected center family outside V, no host function-space topology, no selected subsequence operator, no quotient equality of maps, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 7356.2 (CompactMetricEquicontinuity Arzela–Ascoli handoff)**.** *Every accepted* CompactMetricEquiconti *carrier $E = (K, Y, F, R, M, V, S, A, H, C, P, N)$ supplies the finite-modulus handoff consumed by* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$. *The handoff exposes the compact metric source row K, target metric row Y, finite family row F, pointwise radius rows R, shared modulus candidate M, finite-net cover ledger V, terminal* $\mathsf{Real}^{\uparrow}$ *tolerance seal A, and structural rows H, C, P, N before the consumer reads S. It supplies no open-cover compactness, selected centers outside the displayed ledger, host function-space compactness, selected subsequences, quotient equality of maps, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the compact source row K and the finite-net cover ledger V. Finite induction over the displayed net centers reduces a requested tolerance to finitely many center reads, and each step keeps the membership witness and cover radius inside the same BHist packet.

Use the equicontinuity rows next. The pointwise continuity radius rows R are transported through the shared modulus candidate M, so every family member in F reads the same source-radius budget for the displayed tolerance. The target metric row Y supplies only the distance comparisons attached to those finite rows.

Finally compose the tolerance ledger. The $\mathsf{Real}^{\uparrow}$ seal A records the target-distance budget after the finite-net and equicontinuity-radius transports have been displayed, while H, C, P, N preserve, replay, source, and name the same packet. Therefore the handoff row S gives $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ exactly the finite modulus data it expects, with no hidden compactness, choice, subsequence, quotient, function-space, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricEquicontinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricEquicontinuity}^{\uparrow}$ packet is generated from compact metric finite-net rows, target MetricUp rows, finite ContinuousMapUp family rows, pointwise continuity radius rows, EquicontinuityFamilyUp shared-modulus candidates, finite-net cover ledgers, ArzelaAscoliFiniteModulusUp handoff rows, RealUp tolerance seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactMetricEquicontinuity}^{\uparrow}}$, seed)

Scope closed: Definition 7356.1 and Definition 7356.2 seed the compact-metric equicontinuity carrier and its finite-modulus Arzela–Ascoli handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactMetricEquicontinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected centers outside the displayed ledger, host function-space compactness, selected subsequences, quotient equality of maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net coverage, pointwise radius transport, shared-modulus exactness, finite-cover ledger exhaustion, Real tolerance sealing, and Arzela–Ascoli finite-modulus handoff for BEDC.Derived.CompactMetricEquicontinuityUp.

Chapter 7357

A Concrete Naming Certificate for $\mathsf{MilmanConverse}^{\uparrow}$

Following Milman's converse direction in compact convexity, the $\mathsf{MilmanConverse}^{\uparrow}$ packet records the finite boundary route by which a compact convex carrier reads an extreme-window source before exporting a closure-hull handoff. It sits after the compact carrier of chapter 60, the convex-set surface of chapter 208, the locally convex observation rows of chapter 3386, the Choquet boundary packet of chapter 6611, and the finite Krein–Milman extreme witness of chapter 3610. The chapter names only the compact-convex boundary object needed for a Milman-converse read; it does not assert selected extreme points, strong Hahn–Banach separation, quotient convex-set equality, countable choice, or ambient locally convex completeness.

Definition 7357.1 (Milman converse carrier)**.** *A* $\mathsf{MilmanConverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, V, L, S, E, W, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row naming the compact carrier inspected by the boundary request.*
2. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row exposing affine-combination and convex-subcarrier data.*
3. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *observation row for seminorm and neighbourhood windows used by compact-convex consumers.*
4. *S is the* $\mathsf{ChoquetSimplex}^{\uparrow}$ *boundary row when the read is routed through a simplex-facing compact convex source.*
5. *E is the* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *finite extreme-window row, opened only as displayed candidate-extreme and exposed-face evidence.*
6. *W is the finite extreme-window ledger recording which boundary observations have been inspected.*
7. *G is the closure-hull handoff row, available only after K, V, L, S, E, W are visible.*
8. *H records componentwise* hsame *transport for compact, convex, locally convex, Choquet, finite extreme-window, closure-hull, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a compact-convex boundary request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{MilmanConverse}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{ChoquetSimplex}^{\uparrow}$, $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, L, S, E, W, G, H, C, P, N$. It has no coordinate for a selected extreme point, a strong Hahn–Banach separator, quotient convex-set equality, countable-choice search, or ambient locally convex completeness.

Theorem 7357.2 (Milman converse NameCert obligations)**.** *For every accepted* $\mathsf{MilmanConverse}^{\uparrow}$ *carrier $M = (K, V, L, S, E, W, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, convex-carrier admission through V, locally convex observation through L, Choquet-boundary admission through S, finite Krein–Milman extreme-window admission through E, boundary ledger visibility through W, closure-hull handoff visibility through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7357.1. The compact source, convex carrier, locally convex observations, Choquet boundary row, finite extreme-window row, boundary ledger, and closure-hull handoff are exactly the rows K, V, L, S, E, W, G.

Read the source rows first. The compact row K and convex row V supply the compact-convex object, while L keeps the locally convex observations finite and visible.

Next attach the boundary rows. The Choquet row S supplies the simplex-facing boundary packet when present, the finite Krein–Milman row E supplies only candidate-extreme and exposed-face data, and W records the inspected extreme-window ledger.

Finally the handoff row G may be read, and H, C, P, N transport, replay, source, and name the same route. Since no remaining coordinate stores selected extreme points, a strong Hahn–Banach separator, quotient convex-set equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7357.3 (Milman converse compact-convex handoff)**.** *Every compact-convex boundary read exported by an accepted* $\mathsf{MilmanConverse}^{\uparrow}$ *packet factors through*

$$\mathsf{Compact}^{\uparrow}, \quad \mathsf{ConvexSet}^{\uparrow}, \quad \mathsf{LocallyConvexSpace}^{\uparrow}, \quad \mathsf{ChoquetSimplex}^{\uparrow}, \quad \mathsf{KreinMilmanFiniteExtreme}^{\uparrow}.$$

The closure-hull handoff row G is available only after the compact carrier, convex carrier, locally convex observation row, Choquet boundary row, finite extreme-window row, and boundary ledger have all been displayed. The handoff exports no selected extreme point, strong Hahn–Banach separation, quotient convex-set equality, countable choice, or ambient locally convex completeness.

Proof. Start with Definition 7357.2. It forces any accepted boundary consumer to expose $K, V, L, S, E, W, G, H, C, P$ before the local $\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}$ row can be used.

Read the supplying chapters in order. chapter 60 supplies compact observations, chapter 208 supplies convex carrier data, and chapter 3386 supplies locally convex observation windows.

Then use the boundary sources. chapter 6611 supplies only the displayed Choquet boundary packet, and chapter 3610 supplies only finite exposed-face and candidate-extreme evidence.

Therefore the closure-hull handoff is downstream from the displayed compact-convex boundary route. Since the carrier has no selected extreme point, strong Hahn–Banach separator, quotient convex-set equality, countable-choice selector, or ambient completeness row, the handoff cannot export any of those objects. □

Falsifiable prediction: An accepted $\mathsf{MilmanConverse}^{\uparrow}$ consumer that reaches the closure-hull handoff without first displaying the $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{ChoquetSimplex}^{\uparrow}$, and $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: compact, convexset, locally-convex-space, choquet-simplex, krein-milman-finite-extreme: no listed sibling is bijective to $\mathsf{MilmanConverse}^{\uparrow}$, because the packet jointly carries compact and convex rows, locally convex observations, a Choquet boundary row, a finite extreme-window row, a closure-hull handoff, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}$.

Constructive story: A $\mathsf{MilmanConverse}^{\uparrow}$ packet is generated from compact source rows, convex carrier rows, locally convex observation rows, Choquet boundary rows, finite Krein–Milman extreme-window rows, a closure-hull handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7357.1, Definition 7357.2, and Definition 7357.3 seed the compact-convex boundary certificate over compact, convex, locally convex, Choquet, finite extreme-window, closure-hull, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MilmanConverse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected extreme points, strong Hahn–Banach separation, quotient convex-set equality, countable choice, ambient locally convex completeness, Lean verification, or bridge checking for $\mathsf{MilmanConverse}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, compact and convex source visibility, locally convex observation, Choquet boundary admission, finite extreme-window exactness, closure-hull handoff visibility, transport, replay, provenance, and local naming for BEDC.Derived.MilmanConverseUp.

Chapter 7358

A Concrete Naming Certificate for $\mathsf{HellingerDistance}^{\uparrow}$

Following Hellinger and Kakutani on affinity-style probability distances, the $\mathsf{HellingerDistance}^{\uparrow}$ packet records the finite square-root overlap ledger between two displayed probability or distribution rows. It sits after the probability packet of chapter 184, the pushforward distribution ledger of chapter 186, the metric comparison surface of chapter 57, the Wasserstein transport-plan packet of chapter 5197, and the terminal real-name seal of chapter 38. The chapter names only a finite statistical-distance packet; it does not import measure completion, Radon–Nikodym representatives, selected densities, or a host integration theorem.

Definition 7358.1 (Hellinger distance carrier)**.** *A* $\mathsf{HellingerDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, \mu, \nu, A, B, O, R, M, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-row ledger whose cells are read through an accepted* $\mathsf{ProbSpace}^{\uparrow}$ *carrier or an accepted* $\mathsf{Distribution}^{\uparrow}$ *pushforward surface.*
2. *μ and ν are the two probability or distribution source rows over S.*
3. *A and B are marginal readback rows, exposing the displayed nonnegative weights $\mu(s_i)$ and $\nu(s_i)$ on each listed cell of S.*
4. *O is the square-root overlap ledger, recording the finite entries $\sqrt{\mu(s_i)}\sqrt{\nu(s_i)}$ and their displayed sum.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the terminal Hellinger value, reached only after A, B, and O have been read.*
6. *M is the optional* $\mathsf{Met}^{\uparrow}$ *comparison context in which the resulting real distance may be consumed.*
7. *W is the optional* $\mathsf{Wasserstein}^{\uparrow}$ *comparison context, present only when a consumer compares the overlap distance with a finite transport-plan row over the same source.*
8. *H records componentwise* hsame *transport for $S, \mu, \nu, A, B, O, R, M, W$ and the local naming rows.*
9. *C records displayed* Cont *replay from the source rows through marginal readback, square-root overlap, real sealing, and optional comparison contexts.*

10. *P is the* Pkg *provenance over* $\mathsf{HellingerDistance}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Wasserstein}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HellingerDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for measure completion, a chosen density representative, an ambient integral, a quotient of probability laws, or a selected coupling.

Theorem 7358.2 (Hellinger distance NameCert obligations)**.** *For every accepted* $\mathsf{HellingerDistance}^{\uparrow}$ *carrier*

$$L = (S, \mu, \nu, A, B, O, R, M, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellingerDistance}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier admission for the displayed* BHist *packet L;*
2. *marginal readback through S, μ, ν, A, B, with every public weight attached to a listed finite source cell;*
3. *square-root overlap exactness through O, whose entries and finite sum are the only overlap ledger exported by the packet;*
4. $\mathsf{Real}^{\uparrow}$ *distance sealing through R, downstream from the marginal readbacks and overlap ledger;*
5. *componentwise transport, displayed replay, and provenance through H, C, P;*
6. *local naming through N, with no exported coordinate for measure completion, selected densities, ambient integration, quotient probability laws, or hidden couplings.*

Proof. Project the carrier of Definition 7358.1. The visible coordinates are the finite source ledger, two probability or distribution rows, marginal readbacks, the square-root overlap ledger, the real seal, optional comparison contexts, transport, replay, provenance, and local naming row.

Carrier admission is exactly this projection. Marginal readback is exhausted by A and B: a consumer can inspect only weights attached to the listed cells of S, and each such read is mediated by the displayed $\mathsf{ProbSpace}^{\uparrow}$ or $\mathsf{Distribution}^{\uparrow}$ source row.

The overlap obligation is local to O. Its entries are the displayed products of square-rooted marginal weights, and its finite sum is the only affinity-style scalar exported before the real seal is reached.

Finally read R, H, C, P, N. The $\mathsf{Real}^{\uparrow}$ seal is downstream from the marginal and overlap rows, while transport, replay, provenance, and local naming preserve that same route. Since no other coordinate is present in the packet, the six listed rows exhaust the local NameCert surface and leave measure completion, selected densities, ambient integration, quotient laws, and hidden couplings outside the carrier. □

Theorem 7358.3 (Hellinger finite probability route)**.** *Every public consumer of an accepted* $\mathsf{HellingerDistance}^{\uparrow}$ *packet reads the distance through the finite probability route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{Distribution}^{\uparrow} \longmapsto S, \mu, \nu, A, B, O \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{Wasserstein}^{\uparrow}, H, C, P, N.$$

In particular, a downstream comparison may cite $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Wasserstein}^{\uparrow}$ *only after the probability or distribution source rows, square-root overlap ledger, and* $\mathsf{Real}^{\uparrow}$ *seal have been displayed.*

Proof. Start with the NameCert obligations of Definition 7358.2. They force every public read to expose the finite source ledger, the two probability or distribution rows, the marginal readbacks, and the square-root overlap ledger before the real-valued distance row is available.

The optional comparison rows M and W are downstream coordinates of the same carrier. Thus a $\mathsf{Met}^{\uparrow}$ consumer receives only the real-sealed distance, and a $\mathsf{Wasserstein}^{\uparrow}$ comparison receives only a route already tied to the displayed finite source and overlap ledger.

The structural rows H, C, P, N preserve and name this route without adding analytic data. Since measure completion, selected densities, ambient integration, quotient probability laws, and hidden couplings are absent from the carrier, no public consumer can bypass the finite probability, overlap, and real-seal rows. □

Falsifiable prediction: Every accepted $\mathsf{HellingerDistance}^{\uparrow}$ consumer must expose finite ProbSpace or Distribution source rows, two marginal readback rows, the square-root overlap ledger, the RealUp seal, optional MetricUp or WassersteinUp comparison rows, hsame transport, Cont replay, Pkg provenance, and the local NameCert row before any statistical-distance read is public.

Independence witness: Nearest siblings: ProbSpace, Distribution, Metric, Wasserstein, and Real. $\mathsf{HellingerDistance}^{\uparrow}$ is not bijective to any one of them because it binds two finite marginal rows, a square-root overlap ledger, a terminal RealUp seal, optional metric or transport comparison contexts, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HellingerDistance}^{\uparrow}}$.

Constructive story: A $\mathsf{HellingerDistance}^{\uparrow}$ packet is generated from finite ProbSpace or Distribution source rows, two marginal readback rows, a square-root overlap ledger, a RealUp distance seal, optional MetricUp or WassersteinUp comparison contexts, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellingerDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7358.1, Definition 7358.2, and Definition 7358.3 seed the finite statistical-distance NameCert surface over probability, distribution, metric, Wasserstein, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellingerDistance}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.HellingerDistanceUp`

Bridge status: none

Not claimed: This chapter does not close measure completion, Radon–Nikodym representatives, selected densities, ambient integration, quotient probability laws, optimal transport comparison, Lean theorem verification, or bridge checking.

Upgrade path: Lean verification requires checked carrier admission, marginal readback, square-root overlap exactness, RealUp seal routing, MetricUp and WassersteinUp comparison handoff, transport, replay, provenance, and local naming for BEDC.Derived.HellingerDistanceUp.

Chapter 7359

A Concrete Naming Certificate for $\mathsf{TreeAutomaton}^{\uparrow}$

Following Thatcher and Wright's finite tree automata and the standard automata-theory presentation of regular tree languages, the $\mathsf{TreeAutomaton}^{\uparrow}$ packet records bottom-up recognition of finite ranked trees as BEDC NameCert data. It sits beside the word-facing $\mathsf{FiniteAutomaton}^{\uparrow}$ packet of chapter 2364, the accepted-language surface of $\mathsf{RegularLanguage}^{\uparrow}$ in chapter 351, the grammar-side syntax carrier $\mathsf{ContextFreeGrammar}^{\uparrow}$ of chapter 356, and the finite classifier discipline of $\mathsf{MyhillNerode}^{\uparrow}$ in chapter 5729. The point is a finite syntax-tree recognition packet, not a determinization theorem or a complement-closure result.

Definition 7359.1 (Tree automaton carrier)**.** *A* $\mathsf{TreeAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Sigma, Q, F, \Delta, R, A, H, C, P, N)$$

with the following displayed rows:

1. Σ *is a finite ranked alphabet row, recording each constructor symbol together with its arity.*
2. Q *is a finite state row.*
3. F *is the accepting-state row.*
4. Δ *is the bottom-up transition ledger, assigning a target state to every displayed constructor together with the finite tuple of child states matching its rank.*
5. R *is the finite input-tree row, carrying a rooted syntax tree whose node labels are drawn from* Σ *with matching arities.*
6. A *is the acceptance row, recording the root state reached by replaying* Δ *from leaves to root and checking membership in* F*.*
7. H *records componentwise* hsame *transport for* $\Sigma, Q, F, \Delta, R, A$ *and the local naming row.*
8. C *records displayed* Cont *replay for reading child-state rows, transition rows, and the root acceptance row in the same finite packet.*
9. P *is the* Pkg *provenance over* $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{TreeAutomaton}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite trees, unranked branching, hidden state choice, determinization, complement closure, or an external quotient of regular tree languages.

Theorem 7359.2 (Tree automaton NameCert obligations)**.** *For the carrier of Definition 7359.1, the local* $\mathsf{NameCert}_{\mathsf{TreeAutomaton}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (\Sigma, Q, F, \Delta, R, A, H, C, P, N),$$

ranked-alphabet exposure through Σ*, finite-state exposure through* Q*, accepting-state exposure through* F*, transition-ledger exactness through* Δ*, finite-tree admission through* R*, root-recognition readback through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The finite ranked alphabet, state row, accepting row, transition ledger, input tree, and root-recognition row are the only data from which a $\mathsf{TreeAutomaton}^{\uparrow}$ certificate can be read.

The transport and replay rows do not add a second recognizer. They only preserve the same finite constructor ranks, child-state tuples, transition entries, and root-state check under hsame and Cont.

Finally read P and N. Provenance names the surrounding automata and language packets, while the local NameCert row registers this packet as the tree-recognition surface. No further NameCert obligation remains outside the displayed rows. □

Theorem 7359.3 (Tree automaton recognition stability)**.** *Every accepted* $\mathsf{TreeAutomaton}^{\uparrow}$ *replay that preserves the ranked alphabet row* Σ*, the transition ledger* Δ*, and the finite input-tree row* R *also preserves the root-recognition row* A*. Thus a syntax-tree consumer may export a recognized-tree result only after reading the constructor ranks, child-state transitions, and root accepting-state test from the same packet.*

Proof. Begin with the carrier of Definition 7359.1. Each node of R has a constructor from Σ, and the rank recorded in Σ fixes the number of child states that may be supplied to Δ.

Replay the tree from leaves to root. Since Δ is preserved, each displayed child-state tuple determines the same parent state in the replayed packet. Induction over the finite tree therefore preserves the state assigned to every node.

At the root, the acceptance row A reads exactly that preserved root state and the displayed accepting-state row F. Hence any replay preserving Σ, Δ, R and the displayed packet rows preserves the recognized-tree result seen by the downstream consumer. □

Falsifiable prediction: A $\mathsf{TreeAutomaton}^{\uparrow}$ read is refuted if a consumer exports recognized-tree acceptance without displaying the finite ranked alphabet, child-state transition ledger, input-tree row, root-state calculation, transport, replay, provenance, and local naming row.

Independence witness: finiteautomaton, regularlanguage, contextfreegrammar, myhillnerode: FiniteAutomatonUp recognizes linear words rather than ranked trees; RegularLanguageUp names word-language acceptance rather than tree-recognition replay; ContextFreeGrammarUp records derivation syntax but not bottom-up ranked-tree state propagation; MyhillNerodeUp records right-congruence classifiers for regular word languages. None is bijective to $\mathsf{TreeAutomaton}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{TreeAutomaton}^{\uparrow}}$**.**

Constructive story: A TreeAutomaton$^{\uparrow}$ packet is generated from a finite ranked alphabet, finite states, accepting states, a bottom-up transition ledger, a finite input tree, a root-recognition row, componentwise transport, displayed continuation replay, provenance over the automata-language siblings, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TreeAutomaton}^{\uparrow}}$, seed)

Scope closed: Definition 7359.1, Definition 7359.2, and Definition 7359.3 seed the finite ranked-tree recognition certificate.

Formal verification: FormalStatus(NameCert$_{\text{TreeAutomaton}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim determinization, complement closure, minimization, infinite-tree acceptance, or a checked Lean implementation for TreeAutomaton$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, ranked-alphabet exactness, transition-ledger replay, root-recognition stability, transport, continuation, provenance, and local naming for BEDC.Derived.TreeAutomatonUp.

Chapter 7360

A Concrete Naming Certificate for $\mathsf{PositiveDyadicRadius}^{\uparrow}$

Following Bishop and Bridges on constructive real estimates, the $\mathsf{PositiveDyadicRadius}^{\uparrow}$ packet names a strictly positive dyadic tolerance before a regular Cauchy or Real-facing consumer may use it as an error budget. It sits on the dyadic arithmetic rows of chapter 291, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the halving ladder of chapter 6842. The packet records positivity as displayed finite data; it does not import trichotomy of reals, a host positive cone, quotient equality of names, countable choice, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7360.1 (Positive dyadic radius carrier)**.** *A* $\mathsf{PositiveDyadicRadius}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, E, W, S, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa row, naming the nonzero dyadic numerator visible to the current tolerance request.*
2. *E is the exponent row, naming the finite power-of-two scale used to read the radius as a dyadic rational.*
3. *W is the strict positivity witness row, recording that the displayed mantissa and exponent form a radius above zero inside the dyadic core comparison surface.*
4. *S is the halving successor row, pointing to the next positive dyadic radius consumed by chapter 6842.*
5. *T is the tolerance-consumer row, recording the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *requests allowed to read this positive radius.*
6. *H stores componentwise* hsame *transport for mantissa, exponent, strict positivity, halving successor, tolerance consumer, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a tolerance request to the mantissa, exponent, positivity witness, halving successor, and consumer rows.*
8. *P is the* Pkg *provenance over* $\mathsf{PositiveDyadicRadius}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRadiusHalvingLadder}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PositiveDyadicRadius}^{\uparrow}}$ *row.*
The classifier compares packets only through $M, E, W, S, T, H, C, P, N$. It has no coordinate for host real positivity, arbitrary order trichotomy, quotient Cauchy equality, a selected limiting radius, or an ambient completeness theorem.

Theorem 7360.2 (Positive dyadic radius halving)**.** *Let $R = (M, E, W, S, T, H, C, P, N)$ be an accepted* $\mathsf{PositiveDyadicRadius}^{\uparrow}$ *carrier. Any halving read exported from R first exposes the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa row M, exponent row E, and strict positivity witness W, and then reaches the successor row S. The successor is therefore a positive dyadic radius in the same finite tolerance route, not a host arithmetic choice or a real-order inference.*

Proof. Project the carrier of Definition 7360.1. The positivity data are the rows M, E, W, so a consumer cannot inspect S without first reading the dyadic representation and strict positivity witness.

Read the halving ladder through chapter 6842. Its successor-halving row consumes displayed dyadic radius data and preserves the finite ladder route; here the row S is the local successor coordinate attached to the same mantissa and exponent surface.

Finally replay the request through H, C, P, N. Transport and continuation replay preserve the same packet before the successor is exported, while provenance and local naming keep the read inside the positive-radius certificate. No host positive cone, selected limit, quotient equality, countable-choice schedule, or ambient completeness row is added. □

Theorem 7360.3 (Positive dyadic radius regular-sequence budget)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance consumer of an accepted* $\mathsf{PositiveDyadicRadius}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{DyadicRadiusHalvingLadder}^{\uparrow}.$$

The consumer must read the dyadic mantissa and exponent rows, the strict positivity witness, the finite stream request, the regular-rational budget row, the halving successor, componentwise transport, displayed replay, package provenance, and local naming before using the radius as a Cauchy or Real-facing error budget.

Proof. Begin with the tolerance-consumer row T. It records exactly which $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ budget, and $\mathsf{Real}^{\uparrow}$ seal request may inspect the positive radius.

Use the positivity rows M, E, W. They place the radius inside the dyadic core before any regular-sequence estimate is read, so the budget is a displayed positive dyadic row rather than an inferred real-order fact.

Apply Definition 7360.2. If the consumer refines the tolerance, the successor row S remains attached to the same positive-radius packet and to the halving ladder route.

The structural rows H, C, P, N close the handoff. They preserve the mantissa, exponent, positivity, successor, and consumer rows under transport and replay, so downstream regular Cauchy and Real consumers receive a finite positive-radius budget and nothing from quotient equality, selected limits, countable choice, host trichotomy, or ambient completeness. □

Falsifiable prediction: A regular Cauchy or Real-facing consumer refutes the PositiveDyadicRadiusUp packet if it uses a tolerance as positive without displaying the dyadic mantissa, exponent, strict positivity witness, halving successor, tolerance-consumer row, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, and dyadicradiushalvingladder: none is bijective to $\mathsf{PositiveDyadicRadius}^{\uparrow}$ because only this packet binds strict dyadic positivity, halving successor, tolerance-consumer routing, transport, replay, provenance, and local NameCert evidence as one certificate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveDyadicRadius}^{\uparrow}}$.

Constructive story: A $\mathsf{PositiveDyadicRadius}^{\uparrow}$ packet is generated from a DyadicRatCore mantissa row, exponent row, strict positivity witness, halving successor, StreamName and RegSeqRat tolerance-consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveDyadicRadius}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7360.1, Definition 7360.2, and Definition 7360.3 seed the positive dyadic tolerance surface consumed by regular Cauchy and Real budget arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveDyadicRadius}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real positivity, arbitrary order trichotomy, quotient Cauchy equality, selected limiting radii, ambient completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, strict positivity exactness, halving successor preservation, tolerance-consumer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PositiveDyadicRadiusUp.

Chapter 7361

A Concrete Naming Certificate for $\mathsf{RegularLanguageContinuation}^{\uparrow}$

Following Kleene's regular-language theorem and the automata presentation of Hopcroft and Ullman, the $\mathsf{RegularLanguageContinuation}^{\uparrow}$ packet records regular-language recognition as finite continuation-recognizer rows rather than as a host set of words. It sits beside the transition-as-continuation surface of chapter 482, the bounded run surface of chapter 1016, the machine-specific prefix packet of chapter 1030, and the terminal behaviour boundary of chapter 597. The carrier names a bounded recognition packet; it does not assert a language quotient, minimization theorem, regular-expression equivalence, unbounded evaluator, or host powerset semantics.

Definition 7361.1 (Regular-language continuation carrier)**.** *A* $\mathsf{RegularLanguageContinuation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, I, A, T, W, R, X, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite state-row inventory.*
2. *I is the initial continuation row.*
3. *A is the accepting-row inventory.*
4. *T is the transition table, whose entries are displayed* Cont *steps between state rows.*
5. *W is the finite input-prefix row inspected by the recognizer.*
6. *R is the bounded replay row, recording the run through W using only entries of T.*
7. *X is the acceptance classifier row, reading the endpoint of R against A.*
8. *H stores componentwise* hsame *transport for state, initial, accepting, transition, prefix, replay, acceptance, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from an input-prefix request to the transition and endpoint classifier rows.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularLanguageContinuation}^{\uparrow}$*,* $\mathsf{ContinuationAutomaton}^{\uparrow}$*,* $\mathsf{FinitePrefixAutomaton}^{\uparrow}$*,* $\mathsf{FinitePrefixMachine}^{\uparrow}$*,* $\mathsf{ContinuationTermination}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularLanguageContinuation}^{\uparrow}}$ *row.*

The classifier compares packets only through $Q, I, A, T, W, R, X, H, C, P, N$. It has no coordinate for a host language subset, quotient of recognizers, minimized automaton, regular-expression conversion, unbounded run, or external evaluator.

Theorem 7361.2 (Regular-language continuation closure)**.** *Every accepted* $\mathsf{RegularLanguageContinuation}^{\uparrow}$ *packet presents bounded regular-language recognition as the finite continuation surface*

$$Q, I, A, T, W, R, X, H, C, P, N.$$

Any acceptance consumer must read the continuation automaton rows, finite prefix run, machine-prefix comparison row when present, continuation-termination boundary, transport rows, replay rows, provenance, and local naming before it can consume the classifier X. The closure is a finite NameCert closure of the recognizer packet, not a theorem identifying all regular-language presentations.

Proof. Project the carrier of Definition 7361.1. The state inventory, initial row, accepting rows, transition table, input prefix, replay ledger, and classifier are the only recognition coordinates available.

Read the transition surface through chapter 482. Each transition row is a displayed Cont step, so the recognizer is already a continuation automaton rather than a separate host operational semantics.

Next read the bounded prefix surfaces of chapter 1016 and chapter 1030. These siblings keep W and R finite and prevent the classifier from escaping to an unbounded evaluator or hidden machine trace.

Finally use chapter 597 for the terminal behaviour boundary. The rows H, C, P, N transport, replay, package, and name the same packet, so the acceptance classifier X remains a finite continuation-recognition row and not a quotient language, minimized automaton, regular-expression equivalence, host powerset object, or unbounded computation theorem. □

Falsifiable prediction: A regular-language continuation consumer refutes the packet if it accepts a word without displaying finite state rows, initial row, accepting rows, continuation transition table, input prefix, bounded replay, endpoint classifier, transport, replay, provenance, and local naming.

Independence witness: continuationautomaton, finiteprefixautomaton, finiteprefixmachine, continuationtermination, and regularlanguage: none is bijective to $\mathsf{RegularLanguageContinuation}^{\uparrow}$ because only this packet binds finite continuation transitions, bounded prefix replay, acceptance classification, terminal behaviour boundary, transport, replay, provenance, and local NameCert rows as one recognizer surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularLanguageContinuation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularLanguageContinuation}^{\uparrow}$ packet is generated from finite state rows, an initial continuation row, accepting rows, continuation transition rows, a finite input-prefix row, bounded replay, acceptance classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularLanguageContinuation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7361.1 and Definition 7361.2 seed finite continuation recognition for regular-language packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularLanguageContinuation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimization, regular-expression equivalence, quotient language equality, host powerset semantics, unbounded evaluation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition replay exactness, bounded prefix acceptance, terminal behaviour nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularLanguageContinuationUp.

Chapter 7362

A Concrete Naming Certificate for $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and the standard Kleisli reading of monads, the $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$ packet records the finite route by which a regular-Cauchy completion arrow is composed without quotienting the underlying name data. It sits between the source $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the dyadic tolerance budget of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the Kleisli law surface of chapter 3834, and the stabilization surface of **??**. The packet is a composition interface for regular-Cauchy routes, not a selected limit, quotient completion object, countable-choice schedule, host equality proof, or ambient Real-completeness theorem.

Definition 7362.1 (Regular Cauchy Kleisli extension carrier)**.** *A* $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, F, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *route whose regular rational name is the input to the completion composition.*
2. *F is the Kleisli arrow and modulus row, recording the finite rule that sends each admitted regular-Cauchy request to the next regular-Cauchy route.*
3. *W is the intermediate* $\mathsf{StreamName}^{\uparrow}$ *window through which the source route and the Kleisli arrow are jointly inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget used to synchronize the source request, the intermediate window, and the target readback.*
5. *R is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the dyadic budget and StreamName window have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback row has been displayed.*
7. *H records componentwise* hsame *transport for the source route, Kleisli arrow, window, tolerance budget, target readback, Real seal, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from A through F, W, D, R, E and back through the same transport and naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyKleisliExtension}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite Kleisli-composition packet. It has no coordinate for a quotient sequence, selected limit, countable choice, host equality of completed reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7362.2 (Regular Cauchy Kleisli extension NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$ *carrier* $K = (A, F, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyKleisliExtensio}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission, requiring the displayed source* $\mathsf{RegSeqRat}^{\uparrow}$ *route, Kleisli arrow and modulus row, StreamName window, dyadic tolerance budget, target readback, Real seal, transport, replay, provenance, and local naming row to inhabit one* BHist *packet;*

2. *source-window compatibility, requiring each request read from A to pass through W before the Kleisli arrow F is applied;*

3. *tolerance synchronization, requiring D to bind the source request, the intermediate StreamName window, and the target* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in one finite budget;*

4. *transport and replay stability, requiring H and C to preserve the same displayed rows under componentwise* hsame *transport and* Cont *replay;*

5. *non-escape, excluding quotient sequences, selected limits, countable-choice functionals, host equality of completed reals, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the exported certificate.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 7362.1. The tuple displays the source route A, Kleisli row F, window W, tolerance budget D, target readback R, Real seal E, and structural rows H, C, P, N.

The source-window obligation reads A only through W. Thus a consumer cannot apply the Kleisli arrow to an unlisted source request or infer a hidden completion route.

The tolerance synchronization obligation reads D before R and E. The dyadic budget is the finite ledger tying the source request, intermediate window, and target readback together.

Transport and replay are componentwise. The rows H and C move only the displayed packet, while P and N record the provenance and local certificate. Since the carrier contains no quotient sequence, selected limit, countable-choice functional, host equality proof, or ambient completeness theorem, the local NameCert surface exports none of those objects. □

Theorem 7362.3 (Regular Cauchy Kleisli extension composition law)**.** *Every composed read from an accepted* $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$ *carrier factors through the displayed route*

$$A \longmapsto W \longmapsto F \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the Kleisli composition exports only the finite source route, arrow/modulus row, StreamName window, dyadic budget, target readback, Real seal, transport, replay, provenance, and local NameCert rows. It exports no quotient sequence, selected limit, countable-choice functional, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier $K = (A, F, W, D, R, E, H, C, P, N)$. A composed read first inspects the source regular-Cauchy route A through the finite StreamName window W.

The Kleisli arrow F is then applied only under the displayed modulus row. Before a target readback is exposed, the route reads the dyadic tolerance budget D, so the target request remains tied to the same finite budget.

The row R is the only target $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E is the only $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N preserve and name this same route rather than replacing it by an ambient completion principle.

By Definition 7362.2, the carrier inventory has no coordinate for quotient sequences, selected limits, countable-choice functionals, host equality, or ambient Real completeness. The composed read therefore factors through the displayed rows and exports none of the excluded data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyKleisliExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$ packet is generated from a source $\mathsf{RegSeqRat}^{\uparrow}$ route, a Kleisli arrow and modulus row, an intermediate $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, a target $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyKleisliExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7362.1, Definition 7362.2, and Definition 7362.3 seed the finite Kleisli-composition route for regular-Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyKleisliExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host equality of completed reals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires carrier admission, source-window compatibility, dyadic tolerance synchronization, transport replay, and composition non-escape proofs for BEDC.Derived.RegularCauchyKleisliExtensionUp.

Chapter 7363

A Concrete Naming Certificate for $\mathsf{DiscreteLyapunov}^{\uparrow}$

Following Kalman on sampled linear systems and the Anderson–Moore treatment of finite-dimensional linear quadratic control, the $\mathsf{DiscreteLyapunov}^{\uparrow}$ packet records the finite certificate surface for a displayed one-step Lyapunov recursion. It sits after the energy certificate of chapter 313, the finite descent ledger of chapter 324, the algebraic Riccati rows of chapter 307, the finite-horizon control ledger of chapter 296, and the matrix readback rows of chapter 49. The packet names only the displayed finite recursion, residual ledger, transport, replay, provenance, and local naming rows; it does not assert spectral-radius completeness, host matrix analysis, selected infinite limits, or an analytic convergence theorem.

Definition 7363.1 (DiscreteLyapunov finite recursion carrier)**.** *A* $\mathsf{DiscreteLyapunov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, Q, P, R, E, L, Y, H, C, G, N)$$

with the following displayed rows:

1. *A is the one-step state-transition matrix row read through* $\mathsf{Mat}^{\uparrow}$*.*
2. *Q is the displayed positive cost or dissipation matrix row read through the Lyapunov and LQR source packets.*
3. *P is the candidate Lyapunov matrix row whose quadratic readback is consumed by stability and control consumers.*
4. *R is the finite residual row recording the displayed recursion comparison between P, the transported successor read through A, and the source row Q.*
5. *E is the exactness ledger saying that every accepted recursion read is the same displayed residual row R.*
6. *L is the local dependency row tying Q and P to the* $\mathsf{Lyapunov}^{\uparrow}$*,* $\mathsf{LyapunovStability}^{\uparrow}$*,* $\mathsf{Riccati}^{\uparrow}$*, and* $\mathsf{LQR}^{\uparrow}$ *surfaces.*
7. *Y records the finite row and column indices used by the matrix readback of A, Q, P, R.*
8. *H records componentwise* hsame *transport for A, Q, P, R, E, L, Y.*
9. *C records displayed* Cont *replay from the transition row and matrix rows to the residual comparison.*

10. *G is the* Pkg *provenance over* DiscreteLyapunov$^{\uparrow}$, Lyapunov$^{\uparrow}$, LyapunovStability$^{\uparrow}$, Riccati$^{\uparrow}$, LQR$^{\uparrow}$, Mat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DiscreteLyapunov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed transition, matrix, residual, exactness, dependency, index, transport, replay, provenance, and naming rows. It has no coordinate for a spectral-radius test, a host normed-space theorem, an infinite series selection, or an ambient analytic solver.

Theorem 7363.2 (DiscreteLyapunov NameCert obligations)**.** *For the carrier of Definition 7363.1, the local* $\mathsf{NameCert}_{\mathsf{DiscreteLyapunov}^{\uparrow}}$ *surface consists exactly of carrier admission, transition-matrix exposure, cost-row exposure, Lyapunov-matrix exposure, finite residual exposure, residual exactness through E, dependency exposure through L, finite matrix-index exposure through Y, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that replaces one of these rows by spectral-radius completeness, host matrix analysis, selected infinite limits, or an analytic convergence theorem leaves the accepted carrier.*

Proof. Project the carrier of Definition 7363.1. The rows A, Q, P, R display the transition matrix, cost row, Lyapunov matrix, and residual comparison before any structural transport or control dependency is read.

The dependency row L ties those displayed rows to the sibling Lyapunov, Lyapunov-stability, Riccati, LQR, and matrix surfaces. This row is only a finite handoff; it does not import a spectral-radius criterion, an infinite sum, or a host analytic matrix theorem.

The rows E, Y, H, C, G, N then record exactness, indices, transport, replay, provenance, and local naming for the same finite residual route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the NameCert obligation surface. □

Theorem 7363.3 (DiscreteLyapunov residual route)**.** *Let $D = (A, Q, P, R, E, L, Y, H, C, G, N)$ be an accepted* DiscreteLyapunov$^{\uparrow}$ *packet. Every downstream read of the discrete Lyapunov recursion factors through the finite route*

$$A, Q, P \longrightarrow R \longrightarrow E, L, Y \longrightarrow H, C, G, N.$$

The read first displays the transition and matrix rows, then the residual comparison, then the exactness, dependency, and index rows, and only then the transport, replay, provenance, and local naming rows. The route exports no spectral-radius completeness claim, no host matrix-analysis theorem, no selected infinite limit, and no analytic solver semantics.

Proof. Start with the displayed matrix rows A, Q, P. Their source and target slots are the finite row and column indices later recorded by Y.

The residual row R is the only carrier coordinate that compares the source Lyapunov row, the successor read through A, and the cost row Q. Exactness through E says that any accepted recursion read returns to that displayed residual row.

Finally L, H, C, G, N attach dependency, transport, replay, provenance, and naming data to the same residual route. No remaining coordinate can supply spectral completeness, host matrix analysis, an infinite limit selection, or analytic solver data, so every accepted read factors through the displayed finite route. □

Falsifiable prediction: An accepted DiscreteLyapunov$^{\uparrow}$ read is refuted if a recursion conclusion is reached without displaying the transition matrix row, cost row, Lyapunov matrix row, residual

comparison row, exactness ledger, dependency row, finite matrix-index row, transport, replay, provenance, and local naming rows.

Independence witness: lyapunov, lyapunovstability, riccati, lqr, and matrix: none is bijective to $\mathsf{DiscreteLyapunov}^{\uparrow}$ because only the discrete Lyapunov packet simultaneously records the one-step transition row, cost row, candidate Lyapunov row, finite residual ledger, dependency handoff, matrix indices, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DiscreteLyapunov}^{\uparrow}}$.

Constructive story: A $\mathsf{DiscreteLyapunov}^{\uparrow}$ packet is generated from a one-step transition matrix row, cost row, candidate Lyapunov matrix row, finite residual ledger, residual exactness row, Lyapunov, Lyapunov-stability, Riccati, LQR, and Matrix dependency row, finite matrix indices, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiscreteLyapunov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7363.1, Definition 7363.2, Definition 7363.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite discrete Lyapunov recursion certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiscreteLyapunov}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral-radius completeness, host matrix analysis, selected infinite limits, analytic convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual exactness, dependency-row exposure, finite matrix-index exposure, non-escape for analytic Lyapunov primitives, transport, replay, provenance, and local naming for BEDC.Derived.DiscreteLyapunovUp.

Chapter 7364

A Concrete Naming Certificate for $\mathsf{StrongLawFiniteWindow}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on sums of independent random variables and constructive probability estimates, the $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ packet names the finite-window certificate surface that sits between weak-law tail control and Borel–Cantelli event budgeting. It reads the weak-law finite-average route of chapter 4373, the Chebyshev tail-bound carrier of chapter 6592, the first-lemma event budget of chapter 2357, the normalized probability source of chapter 184, the finite random-variable rows of chapter 185, and the terminal real-name seal of chapter 38. The packet records only a displayed finite schedule of windows and budgets; it is not the full strong law of large numbers.

Definition 7364.1 (Strong law finite-window carrier)**.** *A* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, R, W, C, B, E, H, T, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-event normalization. The row R is the finite* $\mathsf{RandomVar}^{\uparrow}$ *family over P. The row W is the finite window schedule, recording the displayed index blocks whose average rows are inspected. The row C is the* $\mathsf{ChebyshevInequality}^{\uparrow}$ *tail table, with one finite tail row for each displayed window. The row B is the* $\mathsf{BorelCantelli}^{\uparrow}$ *finite-event budget that packages the visible tail rows into a repeated-hit ledger. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the terminal tail budget. The row H records componentwise* hsame *transport for the probability, random-variable, window, Chebyshev, Borel–Cantelli, epsilon, replay, provenance, and local naming rows. The row T records* Cont *replay of the finite-window route, Q records* Pkg *provenance over* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$, $\mathsf{WeakLawLargeNumbers}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{BorelCantelli}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{StrongLawFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted packets only through these ten rows.*

Lean4 ✓ `BEDC.Derived.StrongLawFiniteWindowUp.StrongLawFiniteWindowTasteGate single carrier alignment`

Theorem 7364.2 (Strong law finite-window NameCert obligations)**.** *For every accepted* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *carrier $S = (P, R, W, C, B, E, H, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{StrongLawFiniteWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the probability source, finite random-variable family, finite window schedule, Chebyshev tail rows, Borel–Cantelli finite-event budget,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted read*

must expose $P, R, W, C, B, E, H, T, Q, N$ in that order before a finite-window convergence consumer receives data, and no obligation is stored outside those rows.

Proof. Project the carrier of Definition 7364.1. The rows P, R, W expose the probability source, random-variable family, and finite schedule before any tail estimate can be read.

The rows C, B, E then route the finite-window estimate through the displayed Chebyshev tail table, the Borel–Cantelli finite-event budget, and the positive real tolerance seal. Each row is finite and remains tied to the visible windows.

Finally H, T, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for an infinite almost-sure quotient, countable-choice subsequence, global limit theorem, central-limit normalization, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7364.3 (Strong law finite-window Borel–Cantelli handoff)**.** *Every accepted* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *packet hands a repeated-hit request to* $\mathsf{BorelCantelli}^{\uparrow}$ *only after the finite route*

$$P \longmapsto R \longmapsto W \longmapsto C \longmapsto B \longmapsto E$$

has been displayed. Thus the Borel–Cantelli consumer receives a probability source, finite random-variable family, finite window schedule, Chebyshev tail rows, finite event budget, and real epsilon seal, together with transport, replay, provenance, and local naming; it receives no full strong-law theorem, almost-sure quotient equality, countable-choice subsequence, central limit theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a repeated-hit request from an accepted carrier. Projection exposes P and R as the only probability and random-variable sources available to the request.

The request next reads the window schedule W and tail table C. These rows determine which finite averages are inspected and which Chebyshev estimates are available for those windows.

Only then can B and E be consumed as the finite-event budget and tolerance seal. The structural rows H, T, Q, N preserve that same route for later readers, so the handoff is exactly the finite Borel–Cantelli budget over displayed Chebyshev tails. □

Theorem 7364.4 (Strong law finite-window nonescape)**.** *Every accepted* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *finite-window read remains inside the carrier surface of Definition 7364.1. A consumer can inspect a convergence-facing budget only through the source probability row, finite random-variable rows, finite window schedule, Chebyshev tail rows, Borel–Cantelli finite-event budget,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by the full strong law, almost-sure quotient equality, countable-choice subsequences, a central limit theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a finite-window consumer read an accepted carrier. The definition lists the only coordinates available to the consumer, so the read starts at P, R, W and has no independent limit or quotient coordinate.

The tail-control segment C, B, E is downstream from the displayed schedule and upstream from the structural rows. Hence the consumer receives only a finite table of tail estimates and a finite event-budget seal.

The rows H, T, Q, N add no analytic content outside the displayed packet. They transport equal histories, replay the finite route, record package provenance, and attach the local name. Therefore every accepted read remains inside the finite-window certificate surface and cannot escape to the excluded convergence, quotient, subsequence, limit-law, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{StrongLawFiniteWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ packet is generated from a ProbSpace source row, finite RandomVar family, finite window schedule, Chebyshev tail rows, a BorelCantelli finite-event budget, a RealUp epsilon seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StrongLawFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7364.1, Definition 7364.2, Definition 7364.3, and Definition 7364.4 seed the finite-window probability route through WeakLawLargeNumbers, ChebyshevInequality, BorelCantelli, ProbSpace, RandomVar, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StrongLawFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full strong law, almost-sure quotient equality, countable-choice subsequences, the central limit theorem, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window schedule stability, Chebyshev tail-row exactness, Borel–Cantelli finite-event budget routing, epsilon-seal nonescape, hsame transport, Cont replay, Pkg provenance, and local naming for BEDC.Derived.StrongLawFiniteWindowUp.

Chapter 7365

A Concrete Naming Certificate for $\mathsf{SnowflakeMetric}^{\uparrow}$

Following Assouad and Heinonen on snowflake metrics, the $\mathsf{SnowflakeMetric}^{\uparrow}$ packet records the finite BEDC route that transforms a displayed metric distance by a dyadic exponent $0 < q \leq 1$. It sits over the metric source of chapter 57, the dyadic rational core of chapter 291, the terminal real-name seal of chapter 38, the uniform-continuity route of chapter 391, and the Lipschitz transport discipline of chapter 3762 and chapter 3724. The packet names a finite snowflake-distance readback; it does not import host real powers, quotient metric spaces, choice-selected representatives, or ambient completeness.

Definition 7365.1 (Snowflake metric carrier)**.** *A* $\mathsf{SnowflakeMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, q, D, T, L, H, C, P, N)$$

with the following rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the displayed point pair and distance read;*
2. *q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponent row, carrying the finite inequalities $0 < q$ and $q \leq 1$;*
3. *D is the* $\mathsf{Real}^{\uparrow}$ *distance readback row for the snowflaked distance produced from the displayed source distance and q;*
4. *T is the Holder/snowflake transport row, read through the uniform-continuity surface of chapter 391 and the Lipschitz siblings chapter 3762 and chapter 3724;*
5. *L is the triangle ledger, recording the finite dyadic comparison that keeps the snowflake distance subadditive for $0 < q \leq 1$;*
6. *H records componentwise* hsame *transport of M, q, D, T, L;*
7. *C records displayed* Cont *replay of the same source metric, exponent, readback, transport, and triangle-ledger rows;*
8. *P is the* Pkg *provenance over* $\mathsf{SnowflakeMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SnowflakeMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real exponentiation, quotient equality of metric spaces, selected completions, or choice principles.

Theorem 7365.2 (Snowflake metric NameCert obligations)**.** *For every accepted* $\mathsf{SnowflakeMetric}^{\uparrow}$ *carrier* $S = (M, q, D, T, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SnowflakeMetric}^{\uparrow}}$ *surface consists exactly of source* $\mathsf{Met}^{\uparrow}$ *admission through* M*, dyadic exponent admission through* q *with* $0 < q \leq 1$*, terminal* $\mathsf{Real}^{\uparrow}$ *distance readback through* D*, Holder/snowflake transport through* T*, triangle-ledger accountability through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7365.1. The source metric row, dyadic exponent row, readback row, transport row, triangle ledger, and structural rows are the whole packet.

Read the analytic rows in order. The $\mathsf{Met}^{\uparrow}$ source supplies the distance being transformed, q supplies the positive dyadic exponent bounded by 1, and D records the resulting $\mathsf{Real}^{\uparrow}$ distance readback.

The transport and triangle rows then bind the construction to the existing continuity surface. The rows H, C, P, N transport, replay, source, and name the same finite route, so no host real-power operation, quotient metric-space equality, selected completion, or choice principle is exported. □

Theorem 7365.3 (Snowflake metric Holder route)**.** *Every continuity consumer of an accepted* $\mathsf{SnowflakeMetric}^{\uparrow}$ *packet reads the transformed metric only through the displayed route*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{UniformContinuity}^{\uparrow},\quad \mathsf{LipschitzMap}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{Name}$$

The Holder-facing row T *consumes the dyadic exponent and the source metric distance before exporting uniform-continuity or Lipschitz-style data; it exports no ambient real powers, quotient metric spaces, selected representatives, or untracked completeness facts.*

Proof. Begin with the source metric and exponent rows. The consumer can inspect only the displayed $\mathsf{Met}^{\uparrow}$ distance and the $\mathsf{DyadicRatCore}^{\uparrow}$ proof that $0 < q \leq 1$.

Next read the terminal distance readback and the triangle ledger. These rows attach the snowflaked distance to a $\mathsf{Real}^{\uparrow}$ seal and to the finite subadditivity comparison needed by metric consumers.

Finally pass through the uniform-continuity and Lipschitz sibling routes. They consume the Holder/snowflake transport row T only after M, q, D, L are visible, and the structural rows H, C, P, N keep the handoff inside the same finite NameCert packet. □

Falsifiable prediction: An accepted $\mathsf{SnowflakeMetric}^{\uparrow}$ consumer that obtains a snowflaked distance without reading the source $\mathsf{Met}^{\uparrow}$ row, the positive dyadic exponent row, the $\mathsf{Real}^{\uparrow}$ readback row, the Holder/snowflake transport row, the triangle ledger, and the structural hsame, Cont, Pkg, and NameCert rows refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, uniformcontinuity, lipschitz-function, and lipschitz-modulus: no listed sibling jointly carries the source metric, positive dyadic exponent, snowflaked real readback, Holder transport, triangle ledger, transport, replay, provenance, and local naming rows of $S = (M, q, D, T, L, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{SnowflakeMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SnowflakeMetric}^{\uparrow}$ packet is generated from a source MetricUp row, a positive DyadicRatCoreUp exponent bounded by one, a RealUp distance readback, a Holder/snowflake transport row, a triangle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SnowflakeMetric}^{\uparrow}}$, seed)

Scope closed: Definition 7365.1, Definition 7365.2, and Definition 7365.3 seed the finite snowflake-metric transformation surface over $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, and $\mathsf{LipschitzMap}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SnowflakeMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real powers, quotient metric spaces, selected completions, Lean verification, or bridge checking for SnowflakeMetric$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dyadic exponent bounds, Real distance readback, triangle-ledger accountability, Holder transport, replay, provenance, and local naming for BEDC.Derived.SnowflakeMetricUp.

Chapter 7366

A Concrete Naming Certificate for $\mathsf{DistanceToCompact}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis, the $\mathsf{DistanceToCompact}^{\uparrow}$ packet records the finite certificate surface for the standard distance from a point to a compact metric source. It sits over the metric source of chapter 57, the compact finite-net data of chapter 129, the located comparison function of chapter 6402, the displayed endpoint rows of chapter 7348, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite compact-distance certificate only; it does not choose a nearest point, import open-cover compactness, quotient metric equality, countable choice, or ambient metric completeness.

Definition 7366.1 (Distance to compact carrier)**.** *A* $\mathsf{DistanceToCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, K, L, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *point-data row naming the ambient metric source and the queried point.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row naming the compact metric source by displayed finite covers and tolerance schedules.*
3. *L is the* $\mathsf{LocatedDistanceFunction}^{\uparrow}$ *comparison row that answers point-to-compact distance requests by located real comparisons.*
4. *F is the* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *witness table, recording finite net candidates and the endpoint rows used only as witnesses for bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning the displayed comparison windows into a regular rational name.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance value after the regular readback is replayed.*
7. *H records componentwise* hsame *transport for the metric point data, compact finite-net data, located comparison, finite witness table, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a distance request through X, K, L, F, R, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToCompact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocatedDistanceFunction}^{\uparrow}$, $\mathsf{FiniteMetricProjection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToCompact}^{\uparrow}}$ *row.*

The classifier accepts only packets whose distance value is read through $X, K, L, F, R, E, H, C, P, N$. It has no coordinate for a nearest-point choice, open-cover compactness proof, quotient metric equality, countable-choice schedule, or ambient metric completion.

Lean4 ✓ `BEDC.Derived.DistanceToCompactUp.DistanceToCompactTasteGate single carrier alignment`

Theorem 7366.2 (Distance to compact NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToCompact}^{\uparrow}$ *carrier $D = (X, K, L, F, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DistanceToCompact}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric point row, compact finite-net row, located-distance comparison, finite witness table, regular rational readback, real seal, transport, replay, provenance, and local naming;*
2. *compact-source exposure, requiring K to be read through the finite-net surface of chapter 129 before any distance value is exported;*
3. *located comparison exactness, requiring L to consume the located function route of chapter 6402;*
4. *finite witness discipline, requiring F to provide only finite projection witness rows from chapter 7348, not a global nearest-point selector;*
5. *regular readback and real sealing, requiring R and E to pass through chapter 301 and chapter 38;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DistanceToCompactUp.DistanceToCompactCarrier namecert obligations`

Proof. Project the carrier of Definition 7366.1. The public rows are exactly $X, K, L, F, R, E, H, C, P, N$, so carrier admission exposes only metric point data, compact finite-net data, located comparison, finite witness rows, regular readback, and the terminal real seal.

The compact-source and located-comparison obligations are forced by the replay order. A distance request first reads X and the finite-net row K, then uses L to compare the requested point against the compact source through displayed located-distance windows.

The finite witness table F contributes only finite candidate rows. It can bound and certify the readback route, but it has no coordinate that selects a nearest point in the compact source.

Finally R and E seal the value through the regular rational and Real rows, while H, C, P, N preserve the same row family under transport, replay, provenance, and naming. Since the carrier has no further coordinate, the listed obligations exhaust the NameCert surface and none of the excluded principles can be projected from it. □

Theorem 7366.3 (Distance to compact located readback)**.** *Every accepted* $\mathsf{DistanceToCompact}^{\uparrow}$ *packet $D = (X, K, L, F, R, E, H, C, P, N)$ exports a located distance value only through the displayed route*

$$X, K, L, F \longmapsto R, E \longmapsto H, C, P, N.$$

Thus downstream projection and Hausdorff-distance consumers receive a regular rational readback and a real seal for the distance-to-compact value, while nearest-point choice, open-cover compactness, quotient metric equality, countable choice, and ambient metric completeness remain outside the packet.

Proof. Begin with the compact finite-net row K and the located comparison row L. Together they supply the finite tolerance windows and located comparisons needed to answer a distance request without appealing to an open-cover theorem or a completed ambient metric.

Next read the finite witness table F. The witness rows certify the displayed bounds used by the comparison route, but the endpoint evidence remains a finite table row and is not retyped as a nearest-point choice.

Now replay the readback rows. The regular rational row R records the tolerance-indexed value route, and the Real seal E is readable only after the metric, compact-source, comparison, and finite-witness rows have been displayed.

Finally transport and replay through H, C, P, N. The same packet is delivered to projection and Hausdorff-distance consumers, and row exhaustion excludes quotient metric equality, countable choice, ambient completeness, and any hidden nearest-point principle. □

Lean4 ✓ `BEDC.Derived.DistanceToCompactUp.DistanceToCompactCarrier located readback`

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToCompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7366.1, Definition 7366.2, and Definition 7366.3 seed the finite compact-metric distance certificate through displayed metric, compact-net, located-comparison, finite-witness, regular-readback, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim nearest-point choice, open-cover compactness, quotient metric equality, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, located comparison exactness, finite witness discipline, regular readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.DistanceToCompactUp.

Constructive story: A $\mathsf{DistanceToCompact}^{\uparrow}$ packet is generated from Metric point data, CompactMetric finite-net data, LocatedDistanceFunction comparison, FiniteMetricProjection witness rows, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7367

A Concrete Naming Certificate for $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$ packet records the finite unit-law surface by which a regular Cauchy rational name enters the reflector and completion interface without selecting a quotient representative or invoking ambient completeness. It sits after the regular rational sequence surface of chapter 301 and before the reflector handoff of chapter 1585; the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed only through displayed completion rows.

Definition 7367.1 (Regular Cauchy reflection unit carrier)**.** *A* $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, S, D, F, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, naming the regular rational sequence to be reflected.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source sequence is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger bounding every inspected window.*
4. *F is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *handoff row that reads the regular Cauchy data as reflector input.*
5. *M is the* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *service row used when completion-facing consumers require an existence seal.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the reflector and completion rows have been read.*
7. *H records componentwise* hsame *transport for the regular sequence, windows, dyadic ledger, reflector handoff, completion service, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$R, S, D \longmapsto F \longmapsto M \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionUnit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay route. It has no coordinate for quotient-stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7367.2 (Regular Cauchy reflection unit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$ *carrier* $U = (R, S, D, F, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionUnit}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed regular rational sequence, finite stream windows, dyadic tolerance ledger, reflector handoff, completion-existence service row, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The unit law is visible only as this finite route from* $\mathsf{RegSeqRat}^{\uparrow}$ *through* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *and* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$*; it does not export quotient representatives, selected limits, countable choice, host equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 7367.1. The source row R admits the regular Cauchy rational name, while S and D expose the finite windows and dyadic tolerance ledger through which the source may be inspected.

The unit visibility is the ordered handoff. The reflector row F can be read only after the regular source rows have been displayed, the completion service row M can be read only through that reflector handoff, and the terminal seal E is reached only after the completion-facing route has been exposed.

The Real-seal nonescape follows from the carrier inventory. No row supplies quotient-stream equality, a selected representative, a selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of these can be exported by the certificate.

Finally, H and C preserve and replay the same row order, and P, N record its provenance and local name. Transport or replay therefore cannot replace the unit-law packet by any object outside the displayed finite surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionUnit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$ packet is generated from RegSeqRat source rows, StreamName finite windows, DyadicRatCore tolerance ledgers, a CauchyCompletionReflector handoff, a MetricCompletionExistence service row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionUnit}^{\uparrow}}$, seed)

Scope closed: Definition 7367.1 and Definition 7367.2 seed the finite reflector-unit packet over RegSeqRat, StreamName, DyadicRatCore, CauchyCompletionReflector, MetricCompletionExistence, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionUnit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, countable choice, host equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit visibility, reflector handoff, Real-seal nonescape, transport preservation, replay preservation, provenance, and local naming for BEDC.Derived.RegularCauchyReflectionUnitUp.

Chapter 7368

A Concrete Naming Certificate for $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and Kelley–Bourbaki reflection terminology, the $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$ packet records the counit route from the completed reflector surface back to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows. It is paired with the unit packet of chapter 7367 and reads the reflector boundary of chapter 3398, the regular source of chapter 301, the finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the readback half of the reflector interface without selecting quotient representatives, host limits, countable schedules, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7368.1 (Regular Cauchy reflection counit carrier)**.** *A* $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (E, R, S, D, F, U, H, K, P, N)$$

with the following displayed rows:

1. *E is the completed reflector and* $\mathsf{Real}^{\uparrow}$ *seal row from which a Real-facing consumer begins.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed by the counit.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the readback is inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to every displayed window.*
5. *F is the* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *row that supplies the completion-reflector source of the counit read.*
6. *U is the unit and counit compatibility row pairing this packet with the displayed* $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$ *route.*
7. *H records componentwise* hsame *transport for the completed seal, readback, finite windows, dyadic ledger, reflector row, compatibility row, replay, provenance, and local naming.*
8. *K records displayed* Cont *replay of the route*

$$E \longmapsto R, S, D \longmapsto F, U \longmapsto H, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$, $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$, $\mathsf{RegularCauchyReflectionUnit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionCounit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay order. It has no coordinate for quotient stream equality, a selected representative, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7368.2 (Regular Cauchy reflection counit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$ *carrier* $C = (E, R, S, D, F, U, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionCou}}$ *surface consists exactly of carrier admission for the completed-source and* $\mathsf{Real}^{\uparrow}$ *seal row, regular readback row, finite window row, dyadic tolerance ledger, completion-reflector row, unit and counit compatibility row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The counit handoff is visible only as the finite route from E through R, S, D and then through F, U, H, K, P, N; it does not export quotient stream equality, selected representatives, countable choice, host equality, trichotomy, or ambient Real completeness.*

Proof. Project the carrier of Definition 7368.1. The completed-source row E is the only Real-facing entry point, while R, S, D expose the regular readback, finite observation windows, and dyadic tolerance ledger that the counit may return.

The handoff is ordered by the displayed rows. A consumer first reads the completed seal, then the RegSeqRat readback and its inspection data, and only then reaches the reflector and compatibility rows F and U.

The Real-seal nonescape follows from the carrier inventory. No row supplies quotient stream equality, a selected representative, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of these can be exported by the local certificate.

Finally, H and K preserve and replay the same route, while P and N record provenance and local naming. Transport, replay, and naming therefore cannot replace the counit packet by any object outside the displayed finite surface. □

Theorem 7368.3 (Regular Cauchy reflection counit Real handoff)**.** *Every Real-facing counit read from an accepted* $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$ *carrier factors through the finite route*

$$E \longmapsto R, S, D \longmapsto F, U \longmapsto H, K, P, N.$$

The handoff returns from the completed reflector surface to displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows before any reflector or unit-compatibility row may be consumed. It exports no quotient stream equality, selected representative, countable choice, host equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the Real-facing row E. By definition of the carrier, the counit has no other completed-source coordinate and no hidden host completion object.

The readback rows R, S, D are next in the displayed route. They expose the regular rational name, the finite windows through which it is read, and the dyadic tolerance ledger attached to those windows.

Only after that readback may the reflector row F and compatibility row U be consumed. The structural rows H, K, P, N then transport, replay, record, and name the same finite route.

Since the carrier contains no quotient equality, selected representative, countable-choice schedule, host equality, trichotomy, or ambient completeness row, the Real-facing handoff cannot export any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionCounit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReflectionCounit}^{\uparrow}$ packet is generated from a completed reflector and Real seal row, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyCompletionReflector row, unit and counit compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionCounit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7368.1, Definition 7368.2, and Definition 7368.3 seed the finite reflector-counit packet over the completed source, RegSeqRat, StreamName, DyadicRatCore, RegularCauchyCompletionReflector, RegularCauchyReflectionUnit, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionCounit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completed-source visibility, regular readback, counit handoff, Real-seal nonescape, transport preservation, replay preservation, provenance, and local naming for BEDC.Derived.RegularCauchyReflectionCounitUp.

Chapter 7369

A Concrete Naming Certificate for $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$ packet records the compact sequence-modulus route behind chapter 9237 as finite NameCert data. It sits over the compact metric source of chapter 129, the continuous-map row of chapter 130, the uniform-modulus consumer of chapter 369, and the terminal real-name seal of chapter 38.

Definition 7369.1 (Compact uniform sequence-modulus carrier)**.** *A* $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, U, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. X *is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, including the finite compact windows used by the packet.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence row whose pointwise modulus data are inspected on the compact source.*
3. U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row, exposed only after the compact windows and pointwise rows have been read.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested precision and radius comparisons.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected sequence windows.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the resulting real-valued estimates.*
7. H *records componentwise* hsame *transport for compact source, continuous-map sequence, uniform modulus, dyadic tolerance, regular readback, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay of the route*

$$X, F, D, R \longmapsto U \longmapsto E.$$

9. P *is the* Pkg *provenance over* $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformSequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed compact windows, pointwise sequence-modulus rows, dyadic tolerance ledger, regular readback, uniform-modulus handoff, Real seal, and structural rows. It has no coordinate for open-cover compactness, host-selected subcovers, semantic continuity, countable choice, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7369.2 (Compact uniform sequence-modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$ *carrier* $K = (X, F, U, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUnifo}}$ *surface consists exactly of carrier admission for the displayed compact metric source, continuous-map sequence row, uniform-modulus output, dyadic tolerance ledger, regular rational readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-uniform consumer must read the compact window and pointwise sequence-modulus rows before the uniform-modulus row is exposed; the route exports no open-cover compactness theorem, selected subcover, semantic continuity principle, quotient function equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 7369.1. The compact source X supplies finite compact windows, and the continuous-map sequence row F supplies the pointwise modulus observations inspected on those windows.

The modulus handoff is ordered by the carrier. The dyadic tolerance row D and regular readback row R are read before the $\mathsf{UniformModulus}^{\uparrow}$ output U, and the terminal Real seal E is downstream of that same route.

Transport and replay preserve the compact-uniform surface. The row H moves each coordinate componentwise, and C repeats the inspection route for downstream consumers without adding a second source of compactness or continuity.

Finally, P and N record provenance and local naming. Since the carrier contains no open-cover compactness theorem, selected subcover, semantic continuity principle, quotient function equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness row, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformSequenceModulus}^{\uparrow}$ packet is generated from CompactMetric source windows, ContinuousMap sequence rows, UniformModulus output rows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSequenceModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7369.1 and Definition 7369.2 seed the finite compact sequence-modulus packet over CompactMetric, ContinuousMap, UniformModulus, DyadicRatCore, RegSeqRat, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSequenceModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, semantic continuity, quotient function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, pointwise sequence-modulus visibility, dyadic tolerance readback, uniform-modulus handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSequenceModulusUp.

Chapter 7370

A Concrete Naming Certificate for $\mathsf{FiniteChainDecomposition}^{\uparrow}$

Following Dilworth and Mirsky on finite partially ordered sets, the $\mathsf{FiniteChainDecomposition}^{\uparrow}$ packet records a standalone finite chain-cover carrier for order-theoretic decomposition work. It sits between the finite order surface of chapter 53, the width-cover route of chapter 6083, the height-layer route of chapter 7139, the matching dual certificate of chapter 234, and the ordered spine discipline of chapter 36. The packet names the chain decomposition as displayed finite rows, not as a host poset cover, a quotient order equality, a selected global partition, or an infinite-order theorem.

Definition 7370.1 (Finite chain-decomposition carrier)**.** *A* $\mathsf{FiniteChainDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, L, C, U, A, D, H, T, P, N)$$

with the following displayed rows:

1. *O is the finite* $\mathsf{Poset}^{\uparrow}$ *order row, including the carried elements, comparison ledger, and classifier for repeated element reads.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *chain-spine row, storing a finite list of finite chains as displayed ordered spines.*
3. *C is the chain-certification row, requiring each spine in L to be internally comparable through the same order row O.*
4. *U is the coverage row, recording that every carried element of O appears in at least one displayed chain spine.*
5. *A is the antichain-width boundary row, exposing the width pressure read by* $\mathsf{DilworthDecomposition}^{\uparrow}$ *before a minimum-chain-cover consumer inspects the packet.*
6. *D is the dual handoff row to* $\mathsf{DilworthDecomposition}^{\uparrow}$, $\mathsf{MirskyTheorem}^{\uparrow}$, *and* $\mathsf{Matching}^{\uparrow}$, *binding chain covers, height layers, and matching certificates to the same finite order row.*
7. *H records componentwise* hsame *transport for the order, chain spine, certification, coverage, width boundary, dual handoff, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from O, L, C, U, A, D through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteChainDecomposition}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{DilworthDecomposition}^{\uparrow}$, $\mathsf{MirskyTheorem}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteChainDecomposition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, L, C, U, A, D, H, T, P, N$. It has no coordinate for an ambient host poset, quotient order equality, unlisted chain cover, selected maximal antichain, countable choice, or infinite decomposition principle.

Theorem 7370.2 (Finite chain-decomposition NameCert obligations)**.** *For an accepted* $\mathsf{FiniteChainDecomposition}^{\uparrow}$ *carrier*

$$F = (O, L, C, U, A, D, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteChainDecomposition}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite order row, list-of-chains row, chain-certification row, coverage row, antichain-width boundary row, dual handoff row, componentwise transport, displayed replay, provenance, and local naming; order exposure through O; finite spine exposure through L; chain validity through C; cover exactness through U; Dilworth and Mirsky compatibility through A, D; and nonescape for host posets, quotient order equality, unlisted chain covers, selected external antichains, choice, and infinite-order data.*

Proof. Project the carrier of Definition 7370.1. The finite order is O, the displayed chain family is L, chain validity is C, cover exactness is U, and the width and dual handoff data are A and D.

The order exposure obligation is rowwise. Every chain comparison in C, every coverage read in U, and every width or dual-certificate read in A, D uses the same $\mathsf{Poset}^{\uparrow}$ comparison ledger, so a consumer cannot combine chains from one order with antichain data from another.

The finite-spine and cover obligations are carried by L, C, U. The $\mathsf{List}^{\uparrow}$ row supplies only displayed finite spines, the certification row checks comparability inside each spine, and the coverage row records the carried elements reached by those spines.

Finally A, D, H, T, P, N preserve the chain-cover route. The Dilworth, Mirsky, and Matching siblings may consume the packet only after the finite order, chain spines, and coverage rows are visible, and the structural rows transport, replay, source, and name that same finite surface. □

Theorem 7370.3 (Finite chain-decomposition Dilworth handoff)**.** *Every* $\mathsf{DilworthDecomposition}^{\uparrow}$ *or* $\mathsf{MirskyTheorem}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteChainDecomposition}^{\uparrow}$ *carrier must read the finite order row O, chain-spine row L, chain-certification row C, coverage row U, antichain-width boundary row A, and dual handoff row D before any chain-cover or height-layer conclusion is exported. The handoff may cite the* $\mathsf{Matching}^{\uparrow}$ *dual certificate and the* $\mathsf{List}^{\uparrow}$ *spine discipline, but it exports no host partial order, quotient order equality, selected global decomposition, or infinite poset theorem.*

Proof. Begin with the NameCert obligations of Definition 7370.2. They make $O, L, C, U, A, D, H, T, P, N$ the complete public surface of the finite chain-decomposition packet.

The Dilworth-facing read uses A and D only after the chain family is visible. Thus the width boundary and matching handoff remain attached to the same O, L, C, U rows that certify the chain cover.

The Mirsky-facing read uses the same finite chain data as height-layer evidence. It may route through the sibling packet of chapter 7139, but it cannot replace the displayed chain spines by a selected partition outside the carrier.

Finally H, T, P, N transport, replay, source, and name the handoff. Since neither the carrier nor the cited siblings add a host poset, quotient equality, external decomposition selector, or infinite-order coordinate, the handoff remains a finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteChainDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteChainDecomposition}^{\uparrow}$ packet is generated from a finite Poset row, a List row of displayed chains, chain-certification rows, a coverage row, an antichain-width boundary, a Dilworth-Mirsky-Matching handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteChainDecomposition}\uparrow}$, seed)

Scope closed: Definition 7370.1, Definition 7370.2, and Definition 7370.3 seed the finite chain-cover carrier over Poset, List, DilworthDecomposition, MirskyTheorem, and Matching rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteChainDecomposition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host partial orders, quotient order equality, selected external chain covers, countable choice, infinite posets, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order exposure, finite-spine exposure, chain validity, cover exactness, Dilworth-Mirsky compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FiniteChainDecomposi

Chapter 7371

A Concrete Naming Certificate for $\mathsf{LargeInductiveDimension}^{\uparrow}$

Following Hurewicz–Wallman and Engelking, the $\mathsf{LargeInductiveDimension}^{\uparrow}$ packet records the separator-recursive large inductive dimension as finite BEDC topology and boundary-cover data. It sits beside the small boundary route of chapter 7347, the finite cover-order carrier of chapter 1853, the topology source of chapter 88, the normal-space separation packet of chapter 1866, and the finite refinement carrier of chapter 3430. The packet names a displayed separator-boundary ledger; it does not assert a host dimension theorem, arbitrary open-cover compactness, selected separators outside the carrier, or a choice principle for boundary recursion.

Definition 7371.1 (Large inductive dimension carrier)**.** *A* $\mathsf{LargeInductiveDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, N, B, R, S, C, H, T, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the finite open-set, closed-set, neighborhood, and boundary observations inspected by the packet.*
2. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *closed-set separation row, recording the displayed separation route between closed sets before a separator boundary is read.*
3. *B is the boundary-cover data row, recording finite boundary-cover entries attached to the separator produced by N.*
4. *R is the recursive large-dimension ledger, recording that each separator-boundary read is consumed only through the displayed topology, normal-space, boundary-cover, and comparison rows.*
5. *S is the* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *comparison row, recording the finite handoff to the point-neighborhood boundary route without identifying the two inductive dimension packets.*
6. *C is the* $\mathsf{CoveringDimension}^{\uparrow}$ *comparison row, recording the finite cover-order handoff used when separator-boundary covers are compared with finite refinement data.*
7. *H stores componentwise* hsame *transport for topology, separation, boundary cover, recursion, comparison, replay, provenance, and local naming rows.*
8. *T stores displayed* Cont *replay from a closed-set separation request through separator-boundary inspection, recursive ledger readback, small-dimension comparison, covering-dimension comparison, provenance, and local name.*

9. *P is the* Pkg *provenance over* $\mathsf{LargeInductiveDimension}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{SmallInductiveDimension}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *A is the local* $\mathsf{NameCert}_{\mathsf{LargeInductiveDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, N, B, R, S, C, H, T, P, A$. It has no coordinate for open-cover compactness, selected separator choice, quotient-topology equality, or an ambient theorem equating large inductive dimension with a host cover dimension.

Theorem 7371.2 (Large inductive dimension NameCert obligations)**.** *For every accepted* $\mathsf{LargeInductiveDimension}^{\uparrow}$ *carrier*

$$L = (X, N, B, R, S, C, H, T, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{LargeInductiveDimension}^{\uparrow}}$ *surface consists exactly of five obligation groups:*

1. *carrier admission for the displayed finite packet L;*
2. *separator exactness through the* $\mathsf{Topology}^{\uparrow}$ *source row X and* $\mathsf{NormalSpace}^{\uparrow}$ *closed-set separation row N;*
3. *recursive boundary ledger exactness through the boundary-cover row B and large-dimension recursion row R;*
4. *comparison-row stability through the* $\mathsf{SmallInductiveDimension}^{\uparrow}$ *row S,* $\mathsf{CoveringDimension}^{\uparrow}$ *row C,* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *provenance inside P, componentwise* hsame *transport H, and displayed* Cont *replay T;*
5. *non-escape from host dimension theorems, arbitrary open-cover compactness, selected separator choice, quotient topology equality, and external choice principles.*

Proof. Project the carrier of Definition 7371.1. The topology source, closed-set separator, boundary-cover data, recursive ledger, comparison rows, transport, replay, provenance, and naming row are exactly the displayed coordinates of the finite packet.

The separator obligation is read first from X, N. A consumer sees only the topology observations and the normal-space closed-set separation route before any separator boundary is inspected.

The recursive obligation is then read from B, R. These rows record finite boundary-cover entries and the large-dimension recursion ledger, so the boundary read remains inside the displayed packet.

Finally use S, C, H, T, P, A. The small-dimension and covering-dimension rows are comparison surfaces only, and the structural rows transport, replay, source, and name the same packet. Since the refused host objects are not carrier coordinates, the listed obligation groups exhaust the local NameCert surface. □

Theorem 7371.3 (Large inductive dimension separator route)**.** *Every separator-boundary read of an accepted* $\mathsf{LargeInductiveDimension}^{\uparrow}$ *packet factors through*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{NormalSpace}^{\uparrow},\ \mathsf{SmallInductiveDimension}^{\uparrow},\ \mathsf{CoveringDimension}^{\uparrow},\ \mathsf{FiniteCoverRefinement}^{\uparrow}$$

before the recursive large-dimension ledger is consumed. The route exposes closed-set separation, finite boundary-cover data, small-inductive comparison, covering-dimension comparison, finite refinement provenance, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it exports no host separator choice, no open-cover compactness theorem, and no ambient equality of dimension notions.*

Proof. Begin with the topology source and normal-space separation rows. The read must expose X and N before it can speak about a separator boundary, so there is no channel for an unrecorded host separator.

Next read the boundary-cover and recursion rows B, R. They provide the finite boundary data and the displayed recursive ledger attached to the separator.

Then use the comparison rows. The $\mathsf{SmallInductiveDimension}^{\uparrow}$ row S links the separator boundary to the point-neighborhood boundary route, while the $\mathsf{CoveringDimension}^{\uparrow}$ row C and the $\mathsf{FiniteCoverRefinement}^{\uparrow}$ provenance inside P link the same finite boundary data to cover-order refinement.

Finally project H, T, P, A. These rows transport, replay, source, and name the displayed route, so every separator-boundary read factors through the named topology, normal-space, small-inductive, covering-dimension, and finite-refinement surfaces. □

Falsifiable prediction: An accepted $\mathsf{LargeInductiveDimension}^{\uparrow}$ separator-boundary read is refuted if it consumes a recursive large-dimension ledger without first displaying the TopologyUp source row, NormalSpaceUp closed-set separation row, boundary-cover row, SmallInductiveDimensionUp comparison row, CoveringDimensionUp comparison row, FiniteCoverRefinementUp provenance, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: small-inductive-dimension, coveringdimension, topology, normalspace, finite-cover-refinement: $\mathsf{LargeInductiveDimension}^{\uparrow}$ jointly carries topology-source, closed-set separation, separator-boundary cover data, recursive large-dimension ledger, small-inductive comparison, covering-dimension comparison, finite refinement provenance, transport, replay, provenance, and local naming rows in one packet; each named sibling lacks at least one of these separator-recursive coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{LargeInductiveDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{LargeInductiveDimension}^{\uparrow}$ packet is generated from a TopologyUp source row, a NormalSpaceUp closed-set separation row, finite separator-boundary cover rows, a recursive large-dimension ledger, SmallInductiveDimensionUp and CoveringDimensionUp comparison rows, FiniteCoverRefinementUp provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeInductiveDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7371.1, Definition 7371.2, and Definition 7371.3 seed the separator-recursive large inductive dimension surface over topology, normal-space, small-inductive, covering-dimension, and finite-refinement rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeInductiveDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a host large-inductive-dimension theorem, equality with small inductive dimension, equality with covering dimension, arbitrary open-cover compactness, selected separator choice, quotient topology equality, or external choice principles.

Upgrade path: Obligation closure requires checked carrier admission, separator exactness, recursive boundary ledger exactness, comparison-row stability, transport, replay, provenance, and local NameCert rows for BEDC.Derived.LargeInductiveDimensionUp.

Chapter 7372

A Concrete Naming Certificate for $\mathsf{HeineBorelFiniteCover}^{\uparrow}$

Following Borel and Bishop and Bridges, the $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ packet records the constructive closed-interval finite-cover route by which displayed dyadic meshes and radius ledgers produce a finite cover bundle before any Real or compact-metric consumer reads compactness-facing data. It sits over the closed-bounded interval net of chapter 1250, the compact metric handoff of chapter 129, the dyadic rational arithmetic core of chapter 291, the rational comparison row of chapter 37, and the terminal real-name seal of chapter 38. The packet is a finite closed-interval certificate: it exposes the displayed bundle and its extraction route without asserting open-cover compactness for arbitrary $\mathsf{Real}^{\uparrow}$ subsets.

Definition 7372.1 (Heine–Borel finite-cover carrier)**.** *A* $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, M, U, L, E, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval source row, read through* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$*, with named endpoint seals, boundedness data, and the interval membership route.*
2. *M is the dyadic mesh row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$*, recording the mesh scale, rational cell endpoints, and cell-containment comparisons.*
3. *U is the finite cover bundle, listing the displayed cover entries, their inspected cells, and the finite membership witnesses used by the certificate.*
4. *L is the Lebesgue-radius ledger, recording the positive radius comparisons that make each mesh cell small enough for one displayed cover entry.*
5. *E is the compact-net extraction row, packaging the finite net centers and coverage witnesses handed to* $\mathsf{CompactMetric}^{\uparrow}$ *consumers.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the interval endpoints, inspected cell representatives, and covered points.*
7. *H records componentwise* hsame *transport for the interval source, dyadic mesh, finite cover bundle, radius ledger, compact-net extraction, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from closed-interval inspection through dyadic mesh selection, finite cover bundle inspection, Lebesgue-radius comparison, compact-net extraction, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{HeineBorelFiniteCover}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for arbitrary $\mathsf{Real}^{\uparrow}$ *subsets, unrestricted open covers, trichotomy, host completeness, quotient equality, selected subcovers outside the displayed finite bundle, or bridge data.*

Theorem 7372.2 (Heine–Borel finite-cover NameCert obligations)**.** *For every accepted* $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ *carrier $F = (I, M, U, L, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteCover}^{\uparrow}}$ *obligation surface consists exactly of closed bounded interval source admission, dyadic mesh admission, finite cover bundle exposure, Lebesgue-radius ledger exactness, compact-net extraction, Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7372.1. The interval source I is read before the dyadic mesh M, so the mesh cells are attached to a displayed closed bounded interval rather than to an arbitrary subset of $\mathsf{Real}^{\uparrow}$.

Read the finite cover bundle U and the radius ledger L. These rows state which finitely many cover entries are inspected and which positive radius comparisons certify that the displayed cells fit inside those entries.

The extraction row E and Real seal R are downstream of the same displayed rows. Componentwise transport H, replay C, provenance P, and local naming N preserve that route and add no open-cover compactness, trichotomy, host completeness, quotient equality, or hidden subcover selector. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 7372.3 (Heine–Borel finite-net extraction)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ *packet must read the compact-net extraction row through the finite route*

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Rat}^{\uparrow},\quad U,\quad L,\quad \mathsf{Real}^{\uparrow},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The extracted object is the displayed finite cover bundle together with its mesh and Lebesgue-radius ledger; it is not an open-cover compactness theorem for arbitrary $\mathsf{Real}^{\uparrow}$ *subsets and not a selected subcover outside that displayed bundle.*

Proof. Start with the obligation surface of Definition 7372.2. It places compact-net extraction after the interval source, dyadic mesh, finite cover bundle, radius ledger, and Real seal have all been displayed.

The rows H, C, P, N preserve and name exactly this route. A compact-metric or Real-facing consumer can therefore receive E only as the extraction attached to I, M, U, L, R.

Since no carrier coordinate stores arbitrary subsets, unrestricted open covers, trichotomy, host completeness, quotient equality, or selected subcovers outside U, the extraction exports only the finite closed-interval cover certificate. □

Falsifiable prediction: Every $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ consumer read factors through a closed bounded interval source row, a dyadic mesh row, a displayed finite cover bundle, a Lebesgue-radius

ledger, compact-net extraction, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closedboundedintervalnet, compactmetric, dyadicratcore, rat, real: HeineBorelFiniteCove is not bijective to any listed sibling because it jointly carries the closed-interval source, dyadic mesh, finite cover bundle, Lebesgue-radius ledger, compact-net extraction, Real seal, transport, replay, provenance, and local NameCert rows as one finite-cover packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteCover}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelFiniteCover}^{\uparrow}$ packet is generated from a closed bounded interval source, dyadic mesh comparisons, a displayed finite cover bundle, a Lebesgue-radius ledger, compact-net extraction, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7372.1, Definition 7372.2, and Definition 7372.3 seed the constructive closed-interval finite-cover certificate surface over the displayed finite bundle.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness for arbitrary $\mathsf{Real}^{\uparrow}$ subsets, trichotomy, host completeness, quotient equality, selected subcovers outside the displayed finite bundle, Lean verification, or bridge checking.

Upgrade path: A formal route can encode BEDC.Derived.HeineBorelFiniteCoverUp and prove the finite-cover carrier, NameCert obligation, and net-extraction theorems.

Chapter 7373

A Concrete Naming Certificate for $\mathsf{BoundedContinuousImage}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BoundedContinuousImage}^{\uparrow}$ packet records the finite BEDC carrier for the familiar fact that a continuous real-valued map on a compact metric source has a displayed bound. It sits over the compact metric finite-net source of chapter 129, the continuity-radius surface of chapter 130, the bounded source exactness of chapter 1415, the stream-name handoff of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names only the finite route from compact source observations to a regular rational bound and a sealed real output; it does not import open-cover compactness, selected suprema, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7373.1 (Bounded continuous image carrier)**.** *A* $\mathsf{BoundedContinuousImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, W, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, read through finite net centers, cover radii, and located membership witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from the compact metric source to the real-name target surface.*
3. *W is the finite source-window ledger listing the compact-source centers and local continuity windows consumed by the bound search.*
4. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the finitely many real-name samples produced by F on the displayed windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximations of those finitely many samples.*
6. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *bound row asserting that the displayed finite readback family lies under a rational bound budget.*
7. *H records componentwise* hsame *transport for the compact source, map row, source windows, stream observations, readback, bound row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a requested tolerance through the same finite-window route.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedContinuousImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedContinuousImage}^{\uparrow}}$ *row.*
The carrier contains no coordinate for an open-cover compactness proof, a selected supremum, quotient equality of maps, countable choice, ambient real completeness, or a host-level bounded-image theorem outside the displayed finite rows.

Theorem 7373.2 (Bounded continuous image NameCert obligations)**.** *For every accepted* BoundedContinuousImag *carrier $I = (K, F, W, Q, R, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedContinuousImage}^{\uparrow}}$ *surface consists exactly of compact source admission through K, continuous-map admission through F, finite source-window coverage through W, stream observation through Q, regular rational readback through R, bounded-source exactness through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7373.1. The source and map rows are K and F, and the finite window ledger W records the only compact-source observations available to the packet.

Read the observation rows next. The stream row Q records the finitely many real-name samples produced by the map row on the displayed windows, and R is the regular rational readback for those samples.

The bound row B is downstream of R, so boundedness is asserted only for the displayed finite readback family. The structural rows H, C, P, N transport, replay, source, and name the same finite route; no excluded compactness, supremum, quotient, choice, or ambient completeness principle has a carrier coordinate. □

Theorem 7373.3 (Bounded continuous image compact-bound route)**.** *Every accepted* $\mathsf{BoundedContinuousImage}^{\uparrow}$ *packet routes a compact metric source and a continuous real-valued map through the finite chain*

$$\mathsf{CompactMetric}^{\uparrow} \to \mathsf{ContinuousMap}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{BoundedRealSequence}^{\uparrow}.$$

The bound row B may be read only after the finite source-window ledger W, stream observation row Q, and regular rational readback row R have been displayed.

Proof. Start with K and F. Compactness is used only through the finite net and located membership data recorded in the source row, while continuity is used only through the displayed map row and the windows listed in W.

The image samples are then named by Q and read back by R. Since B is a bounded real sequence row for that finite readback family, a consumer cannot read the bound before the compact-window and continuity observations have been exposed.

Transport and replay through H and C keep the same route under componentwise equality and tolerance requests. Provenance and local naming through P and N seal the route as the stated compact-bound surface. □

Theorem 7373.4 (Bounded continuous image real seal nonescape)**.** *For every accepted* $\mathsf{BoundedContinuousImage}^{\uparrow}$ *carrier $I = (K, F, W, Q, R, B, H, C, P, N)$, any* $\mathsf{Real}^{\uparrow}$ *read of an image value or displayed bound must pass through the stream-name row Q, the regular rational readback row R, and the bounded real sequence row B. The carrier supplies no direct* $\mathsf{Real}^{\uparrow}$ *completeness principle, selected supremum, quotient equality of maps, or host real equality.*

Proof. Inspect the real-facing rows of the carrier. The map row F produces image observations only through the stream-name row Q, and those observations become rationally readable only through R.

The bound is recorded by B after the readback. Therefore a $\mathsf{Real}^{\uparrow}$ consumer can read an image value or bound only through the displayed stream, regular-rational, and bounded-sequence route.

The remaining coordinates H, C, P, N preserve and name that route. They add no coordinate for a completeness principle, supremum selector, quotient map equality, or host equality, so the real seal cannot escape the finite packet. □

Theorem 7373.5 (Bounded continuous image public export)**.** *The public surface of an accepted* $\mathsf{BoundedContinuousImage}^{\uparrow}$ *packet is the finite bound certificate carried by* $K, F, W, Q, R, B, H, C, P, N$*. A downstream compact-continuous-real consumer may import the displayed compact-bound route and real-seal nonescape result, but receives no open-cover compactness theorem, selected supremum, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean credential, or bridge claim.*

Proof. Combine Definition 7373.2 with Definition 7373.3. The first theorem lists the complete local NameCert surface, and the second orders the compact, continuity, stream, readback, and boundedness rows.

Use Definition 7373.4 for the real-facing boundary. It keeps every real read downstream of Q, R, B and excludes completeness, supremum, quotient, and host-equality readings.

Thus a public consumer receives exactly the finite certificate named by the carrier and no stronger analytic or verification principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedContinuousImage}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedContinuousImage}^{\uparrow}$ packet is generated from compact metric finite-net rows, continuous map rows, finite source windows, StreamName observations, RegSeqRat readback, a BoundedRealSequence bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedContinuousImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7373.1, Definition 7373.2, Definition 7373.3, Definition 7373.4, and Definition 7373.5 seed the finite compact-continuous-image bound certificate over compact metric, continuous map, stream-name, regular rational, bounded sequence, and real-name rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedContinuousImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected suprema, quotient equality of maps, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite source-window coverage, stream observation exactness, RegSeqRat readback, bounded-source exactness, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.BoundedContinuousImageUp.

Chapter 7374

A Concrete Naming Certificate for $\mathsf{WellFoundedTreeRank}^{\uparrow}$

Following Brouwer-style finite-bar arguments and Bishop's use of induction over finitely generated construction trees, the $\mathsf{WellFoundedTreeRank}^{\uparrow}$ packet records the bounded tree-rank surface needed by prefix-tree reasoning in the L10 stream and completion route. It sits after the finitely branching tree carrier of chapter 6488 and the Brouwer bar-induction boundary of chapter 1675. The packet names only a finite rank row for displayed prefix trees; it does not import choice, infinite branch selection, host well-founded recursion, or quotient tree equality.

Definition 7374.1 (Well-founded tree-rank carrier)**.** *A* $\mathsf{WellFoundedTreeRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, R, L, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet or constructor row for prefix labels;*
2. *R is the rank row, listing a natural height bound for each displayed node in the finite prefix tree;*
3. *L is the leaf row, recording the frontier nodes where the rank is 0;*
4. *W is the window row, recording the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *prefix windows served by the tree;*
5. *H is the componentwise* hsame *transport row for A, R, L, W;*
6. *C is the displayed* Cont *replay row that extends a prefix window without changing the already exposed rank data;*
7. *P is the* Pkg *provenance over* $\mathsf{WellFoundedTreeRank}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{WellFoundedTreeRank}^{\uparrow}}$ *row.*

The carrier has no coordinate for choosing an infinite branch, identifying distinct tree presentations by quotient equality, or invoking an ambient well-founded-recursion principle outside the displayed finite rank row.

Theorem 7374.2 (Well-founded tree-rank NameCert obligations)**.** *For the carrier of Definition 7374.1, the local* $\mathsf{NameCert}_{\mathsf{WellFoundedTreeRank}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (A, R, L, W, H, C, P, N)$, rank-row exposure through R, leaf-frontier exposure through L, finite*

prefix-window readback through W, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The finitely branching predecessor surface is supplied by chapter 6488, while the bar-facing boundary of chapter 1675 explains why a finite frontier may be read without selecting an infinite branch.

Induct on the displayed finite rank row R. At rank 0, the leaf row L supplies the frontier certificate, and the window row W reads only the finite $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ prefix already present at that leaf.

For the successor rank, each immediate subtree has a strictly smaller displayed rank. Apply the induction hypothesis to those rows, then repack the parent by H, C, P, N. Since every step reads only the finite coordinates A, R, L, W, H, C, P, N, the NameCert obligations are exhausted by the stated surface. □

Theorem 7374.3 (Well-founded tree-rank induction route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *prefix argument served by an accepted* $\mathsf{WellFoundedTreeRank}^{\uparrow}$ *packet may use induction only over the displayed finite rank row R. The route first exposes the finite branching data, then the leaf frontier, then the prefix window W, and only then replays continuation through C; it exports no choice of an infinite branch and no host recursion over an ambient tree type.*

Proof. Begin with Definition 7374.1. The only induction coordinate is R, and every consumer-visible prefix window is recorded in W before continuation replay is available.

For the base rank, the leaf-frontier row L gives the terminal prefix window. The consumer reads the finite $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ prefix directly from W, so no branch selection or quotient tree equality is needed.

For the successor rank, use the induction hypothesis on each displayed predecessor row. Finitely branching exposure supplies the finite list of predecessors, and the Brouwer-style bar boundary supplies the reason that the read remains frontier-local. Transport and replay through H and C preserve the same packet, while P and N keep the route inside the local NameCert. □

Falsifiable prediction: A prefix-tree proof that deletes the rank row R can no longer distinguish finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ window induction from an illicit infinite-branch selection.

Independence witness: finitelybranchingtree, brouwerbarinduction, streamname, regseqrat: FinitelyBranchingTreeUp supplies finite branching but not a displayed decreasing rank row; BrouwerBarInductionUp supplies the bar-facing boundary but not the L10 prefix-window packet; StreamNameUp and RegSeqRatUp supply finite windows but not the tree-rank induction carrier. None is bijective to $\mathsf{WellFoundedTreeRank}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{WellFoundedTreeRank}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WellFoundedTreeRank}^{\uparrow}$ packet is generated from a finite label row, displayed rank row, leaf frontier, finite stream or regular-rational prefix window, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WellFoundedTreeRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite rank carrier, NameCert obligation surface, and prefix-window induction route for $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WellFoundedTreeRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim choice, infinite branch selection, host well-founded recursion, quotient tree equality, or ambient completion equality.

Upgrade path: Reaching obligation closure requires checked carrier admission, rank-row exposure, leaf-frontier readback, finite prefix-window preservation, transport, replay, provenance, and local naming for BEDC.Derived.WellFoundedTreeRankUp.

Chapter 7375

A Concrete Naming Certificate for BanachFixedPointConstructive$^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the BanachFixedPointConstructi packet records the finite choice-free route from a displayed contraction to a fixed-point seal. It sits after the contraction-map surface of chapter 1418, the Picard iteration ledger of chapter 2359, the complete-metric boundary of chapter 128, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Banach fixed-point modulus route of chapter 4188. The packet names only the displayed contraction data, Picard Cauchy schedule, completion service, regular-rational handoff, fixed seal, transports, replay, provenance, and local naming; it is not an ambient completeness theorem, selected host fixed point, quotient orbit equality, trichotomy principle, or countable-choice construction.

Definition 7375.1 (Constructive Banach fixed point carrier)**.** *A* BanachFixedPointConstructive$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, T, L, I, M, C, R, E, H, Q, P, N)$$

with the following displayed rows:

1. *X is the* CompleteMetric$^{\uparrow}$ *source row naming the metric carrier whose completeness is available only through the displayed service boundary.*
2. *T is the* ContractionMapping$^{\uparrow}$ *row for the self-map and its contraction replay surface.*
3. *L is the rational contraction ledger below one, including the displayed positivity and unit-bound rows consumed by the Picard estimate.*
4. *I is the* PicardIteration$^{\uparrow}$ *orbit row, recording the finite iterate schedule and adjacent-step residuals.*
5. *M is the* BanachFixedPointModulus$^{\uparrow}$ *or* CauchyModulus$^{\uparrow}$ *threshold row extracted from T, L, I.*
6. *C is the completion-facing service row read from X only after the Picard Cauchy threshold has been displayed.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row reached from I, M, C.*
8. *E is the terminal* Real$^{\uparrow}$ *fixed seal row reached only after R and C have been read.*

9. *H records componentwise* hsame *transports for the complete-metric, contraction, contraction-ledger, Picard, modulus, completion-service, readback, seal, replay, provenance, and local naming rows.*

10. *Q records displayed* Cont *replay from the contraction and Picard rows to the modulus threshold, from the modulus threshold to regular-rational readback, and from readback plus completion service to the fixed seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BanachFixedPointConstructive}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointConstructive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, L, I, M, C, R, E, H, Q, P, N$. It has no coordinate for a host-selected fixed point, quotient equality of iterates or reals, ambient $\mathsf{Real}^{\uparrow}$ *completeness, unrestricted contraction theorem, trichotomy, analytic continuation, or countable choice.*

Theorem 7375.2 (Constructive Banach fixed point NameCert obligations)**.** *For every accepted* $\mathsf{BanachFixedPointConstructive}^{\uparrow}$ *carrier $B = (X, T, L, I, M, C, R, E, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointCon}}$ *surface consists exactly of carrier admission for the complete-metric source X, contraction row T, rational contraction ledger L, Picard orbit I, Cauchy threshold M, completion service C, regular-rational readback R, terminal fixed seal E, componentwise transport H, displayed replay Q, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 7375.1. The analytic rows visible to the certificate are exactly X, T, L, I, M, C, R, E, followed by transport, replay, provenance, and local naming.

The contraction side is exhausted by T, L, I. The self-map row supplies the displayed contraction replay, the rational ledger supplies the bound below one, and the Picard row supplies the finite orbit and residual schedule.

The completion side is downstream from the same finite data. The modulus row M is extracted before the completion service C, regular-rational readback R, or terminal fixed seal E can be read.

Finally H, Q, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7375.3 (Constructive Banach Picard-Cauchy route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachFixedPointConstructive}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{BanachFixedPointModulus}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The route exposes only the displayed complete-metric source, contraction row, rational contraction ledger, Picard orbit, Cauchy threshold, completion service, regular-rational readback, terminal fixed seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Proof. Start with the NameCert obligations of Definition 7375.2. They place every read of the packet behind the complete-metric source, contraction row, rational contraction ledger, Picard orbit, and explicit threshold.

Next read the Picard-Cauchy handoff. The contraction and Picard rows supply the finite residual schedule, while the modulus row M is the only threshold coordinate available before regular-rational readback.

The completion service is confined to the displayed $\mathsf{CompleteMetric}^{\uparrow}$ row X. Once M and C are visible, the certificate may read R and then the fixed seal E.

Thus every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the displayed route and cannot bypass it through a selected fixed point, quotient equality, ambient completeness, or countable choice. □

Theorem 7375.4 (Constructive Banach fixed seal nonescape)**.** *Let $B = (X, T, L, I, M, C, R, E, H, Q, P, N)$ be an accepted* $\mathsf{BanachFixedPointConstructive}^{\uparrow}$ *carrier. Every fixed-point consumer that reads the terminal seal E must first expose the complete-metric source X, contraction row T, rational contraction ledger L, Picard orbit I, Cauchy threshold M, completion service C, and regular-rational readback R.*

Consequently the terminal fixed seal exports no selected host fixed point, quotient orbit equality, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, unrestricted contraction theorem, trichotomy, analytic continuation, or countable-choice schedule.*

Proof. Project the carrier of Definition 7375.1. The seal coordinate E occurs only after the contraction, Picard, modulus, completion-service, and regular-rational rows in the displayed packet.

Apply Definition 7375.3. It forces every regular-rational or real-facing read through T, L, I before M, through M before C and R, and through R before E.

The structural rows H, Q, P, N preserve this order under transport, replay, provenance, and local naming. They add no coordinate for host selection, quotient equality, ambient completeness, analytic continuation, trichotomy, or choice.

Therefore the fixed seal is a nonescape boundary for the displayed constructive route only. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointConstructive}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointConstructive}^{\uparrow}$ packet is generated from a CompleteMetric source, ContractionMapping row, rational contraction ledger, PicardIteration orbit, BanachFixedPointModulus or CauchyModulus threshold, completion service, RegSeqRat readback, RealUp fixed seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointConstructive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7375.1, Definition 7375.2, Definition 7375.3, and Definition 7375.4 seed the finite choice-free contraction fixed-point route through Picard, modulus, complete-metric, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointConstructive}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected host fixed points, quotient orbit equality, quotient real equality, ambient RealUp completeness, unrestricted contraction theorems, trichotomy, analytic continuation, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, contraction replay, rational contraction ledger exactness, Picard orbit ordering, Cauchy threshold extraction, complete-metric service confinement, RegSeqRat readback, RealUp fixed-seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointConstru

Chapter 7376

A Concrete Naming Certificate for $\mathsf{FixedPointIndex}^{\uparrow}$

Following Dold and Brown on fixed-point index and Lefschetz theory, the $\mathsf{FixedPointIndex}^{\uparrow}$ packet records the finite BEDC surface by which a displayed compact-metric neighborhood, a continuous self-map row, homology trace data, and a Lefschetz fixed-point handoff are inspected before an index value is exported. It sits after the compact source row of chapter 129, the map row of chapter 130, the homology ledger of chapter 98, the Lefschetz fixed-point packet of chapter 6557, and the Brouwer fixed-point metric reconstruction of chapter 5941. The chapter records only finite carrier rows and NameCert obligations; it does not assert a host fixed-point theorem or a global index existence theorem.

Definition 7376.1 (Fixed-point index carrier)**.** *A* $\mathsf{FixedPointIndex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, U, F, H, L, B, V, T, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through* $\mathsf{CompactMetric}^{\uparrow}$*, naming the inspected compact-metric neighborhood and its finite cover interface.*
2. *U is the isolating neighborhood row, recording the finite boundary and exclusion witnesses used before any fixed-point index read is admitted.*
3. *F is the continuous-map row, read through* $\mathsf{ContinuousMap}^{\uparrow}$*, for the displayed self-map on the inspected source.*
4. *H is the homology ledger row, read through* $\mathsf{Homology}^{\uparrow}$*, recording the finite chain, trace, and boundary data available to the index packet.*
5. *L is the Lefschetz handoff row, read through* $\mathsf{LefschetzFixedPoint}^{\uparrow}$*, attaching the displayed trace expression to the fixed-point-facing consumer route.*
6. *B is the Brouwer metric fixed-point comparison row, read through* $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$*, used only as a finite local comparison surface.*
7. *V is the integer-valued index row, containing the displayed sign, degree, or trace readback accepted by the packet.*
8. *T records componentwise* hsame *transport for the compact source, isolating row, continuous-map row, homology ledger, Lefschetz handoff, Brouwer comparison, index value, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-source inspection through isolating-neighborhood admission, continuous-map exposure, homology ledger readback, Lefschetz handoff, Brouwer comparison, and index-value export.*
10. *P is the* Pkg *provenance over* $\mathsf{FixedPointIndex}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{LefschetzFixedPoint}^{\uparrow}$, $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FixedPointIndex}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for an ambient fixed-point set, unrestricted homotopy invariance, selected zero, global Brouwer theorem, quotient space equality, or bridge data.

Theorem 7376.2 (Fixed-point index NameCert obligations)**.** *For every accepted* $\mathsf{FixedPointIndex}^{\uparrow}$ *carrier* $I = (K, U, F, H, L, B, V, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FixedPointIndex}^{\uparrow}}$ *obligation surface consists exactly of compact-source admission, isolating-neighborhood exposure, continuous-map exposure, homology ledger exactness, Lefschetz handoff, Brouwer metric comparison, index-value readback, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7376.1. The compact source K and isolating row U are displayed before the map row F, so the index read is attached to a visible local neighborhood rather than to an ambient fixed-point set.

Read the homology ledger H and Lefschetz handoff L. These rows expose the finite chain and trace data that downstream consumers may inspect, while the Brouwer comparison row B remains only a local metric comparison surface.

Finally V, T, C, P, N record the index value, transport, replay, provenance, and local name for that same route. Since none of the carrier rows stores unrestricted homotopy invariance, selected zeros, global fixed-point existence, quotient equality, or bridge data, the listed rows exhaust the NameCert obligations. □

Theorem 7376.3 (Fixed-point index dependency route)**.** *Every accepted* $\mathsf{FixedPointIndex}^{\uparrow}$ *consumer read of the index row V must factor through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Homology}^{\uparrow},\ \mathsf{LefschetzFixedPoint}^{\uparrow},\ \mathsf{BrouwerFixedPointMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Co}$$

The route exports the displayed local index packet and its homology trace ledger. It does not export a global host fixed-point theorem, unrestricted Lefschetz theorem, selected fixed point, quotient-space equality, or bridge certificate.

Proof. Start with Definition 7376.2. It places the index value after the compact source, isolating row, displayed self-map, homology ledger, Lefschetz handoff, and Brouwer metric comparison have all been exposed.

The componentwise transport, replay, and provenance rows preserve this ordering. Thus a consumer cannot read V without also reading the finite topology and homology rows that generated it.

Because the classifier has no coordinate for a host fixed-point set, unrestricted Lefschetz theorem, selected fixed point, quotient equality, or bridge certificate, the exported route is exactly the displayed local index packet. □

Falsifiable prediction: Every $\mathsf{FixedPointIndex}^{\uparrow}$ consumer read factors through a compact source row, an isolating neighborhood row, a displayed $\mathsf{ContinuousMap}^{\uparrow}$ self-map, a $\mathsf{Homology}^{\uparrow}$ trace

ledger, a $\mathsf{LefschetzFixedPoint}^{\uparrow}$ handoff, a $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ comparison row, an index-value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuous-map, homology, lefschetz-fixed-point, brouwer-fixed-point-metric: $\mathsf{FixedPointIndex}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an isolating neighborhood, displayed self-map, finite homology ledger, Lefschetz handoff, Brouwer comparison, and local index row as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FixedPointIndex}^{\uparrow}}$.

Constructive story: A $\mathsf{FixedPointIndex}^{\uparrow}$ packet is generated from a CompactMetricUp source row, isolating neighborhood row, ContinuousMapUp self-map row, HomologyUp ledger, LefschetzFixedPointUp handoff, BrouwerFixedPointMetricUp comparison, index-value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FixedPointIndex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7376.1, Definition 7376.2, and Definition 7376.3 seed the local fixed-point index certificate surface over displayed compact, map, homology, Lefschetz, and Brouwer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FixedPointIndex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global host fixed-point theorem, unrestricted homotopy invariance, selected fixed points, quotient-space equality, Lean verification, or bridge checking.

Upgrade path: A formal route can encode BEDC.Derived.FixedPointIndexUp and prove the carrier, NameCert obligation, and dependency-route theorems.

Chapter 7377

A Concrete Naming Certificate for $\mathsf{StratifiedSpace}^{\uparrow}$

Following Whitney and Thom, the $\mathsf{StratifiedSpace}^{\uparrow}$ packet records a finite stratum ledger for a displayed topological space whose local pieces are read through manifold, simplicial, or cellular charts. It sits after the indexed-open surface of chapter 88, the deformation route of chapter 156, the finite face inventory of chapter 238, and the cellular attachment surface of chapter 6235. The packet names incidence, frontier, local chart, and consumer handoff rows without importing host stratification theory, selected tubular neighborhoods, quotient spaces, or ambient smooth transversality.

Definition 7377.1 (Stratified space carrier)**.** *A* $\mathsf{StratifiedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, L, F, C, A, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the named open indices and finite membership reads for the ambient observable space;*
2. *L is the finite stratum-label row, listing the displayed strata and their dimension or depth tags;*
3. *F is the frontier-incidence row, recording which stratum closures may meet lower-depth strata;*
4. *C is the local chart row, read through* $\mathsf{SimplicialComplex}^{\uparrow}$ *or* $\mathsf{CWComplex}^{\uparrow}$ *finite cells when a consumer needs combinatorial support;*
5. *A is the deformation-admissibility row, read through* $\mathsf{Homotopy}^{\uparrow}$ *for stratum-preserving deformation requests;*
6. *H is the componentwise* hsame *transport row for T, L, F, C, A;*
7. *R is the displayed* Cont *replay row for refining a finite chart or frontier read without changing the named stratum ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{StratifiedSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{CWComplex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StratifiedSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host Whitney condition theorem, a selected normal slice, quotient topological equality, global triangulation choice, or ambient smooth transversality.

Theorem 7377.2 (Stratified space NameCert obligations)**.** *For the carrier of Definition 7377.1, the local* $\mathsf{NameCert}_{\mathsf{StratifiedSpace}^{\uparrow}}$ *surface consists exactly of ambient topology exposure through* T*, finite stratum inventory through* L*, frontier incidence through* F*, local chart support through* C*, stratum-preserving deformation admissibility through* A*, componentwise transport through* H*, displayed replay through* R*, provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The topology source supplies open-membership rows, the stratum row lists only finitely many inspected pieces, and the frontier row records only the displayed incidence relation among those pieces.

Read the sibling chart rows next. The simplicial-complex and CW-complex chapters supply finite face and cell inventories, while the homotopy chapter supplies only the deformation route attached to those displayed strata.

Finally attach transport, replay, provenance, and naming. Since every consumer-visible read is one of $T, L, F, C, A, H, R, P, N$, the packet cannot export host stratification theory, selected tubular neighborhoods, quotient spaces, global triangulation choice, or transversality data outside the stated NameCert surface. □

Theorem 7377.3 (Stratified space cellular handoff)**.** *Every homology, homotopy, or topology consumer of an accepted* $\mathsf{StratifiedSpace}^{\uparrow}$ *packet must expose the finite route*

$$\mathsf{Topology}^{\uparrow} \longmapsto L, F \longmapsto \mathsf{SimplicialComplex}^{\uparrow}, \mathsf{CWComplex}^{\uparrow} \longmapsto \mathsf{Homotopy}^{\uparrow}$$

before it reads a stratum-preserving boundary. The route is a finite chart-and-frontier handoff, not a host Whitney stratification theorem or a quotient realization of a stratified space.

Proof. Start with Definition 7377.1. The ambient topology row T is read before any stratum label, and the frontier row F is attached to the same finite stratum inventory L.

Next read the chart row C. Any simplicial or cellular support is supplied by the sibling $\mathsf{SimplicialComplex}^{\uparrow}$ and $\mathsf{CWComplex}^{\uparrow}$ packets, so the handoff has finite face or cell data before the deformation row is inspected.

The row A then records only stratum-preserving deformation admissibility through $\mathsf{Homotopy}^{\uparrow}$. Transport and replay preserve the same route through H, R, P, N, so the consumer receives the displayed packet and no additional global stratification principle. □

Falsifiable prediction: A $\mathsf{StratifiedSpace}^{\uparrow}$ consumer that deletes the frontier row F can no longer distinguish a stratum-preserving chart handoff from an arbitrary topological decomposition.

Independence witness: topology, homotopy, simplicialcomplex, cw-complex: TopologyUp supplies open-membership rows but not a stratum frontier ledger; HomotopyUp supplies deformation rows but not finite strata; SimplicialComplexUp and CWComplexUp supply finite face or cell inventories but not the ambient topology and deformation handoff together. None is bijective to $\mathsf{StratifiedSpace}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{StratifiedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StratifiedSpace}^{\uparrow}$ packet is generated from a Topology row, finite stratum labels, frontier incidence, simplicial or cellular chart support, Homotopy deformation admissibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StratifiedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite carrier, NameCert obligation surface, and cellular handoff route for displayed stratified-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StratifiedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: It does not claim host Whitney conditions, selected tubular neighborhoods, quotient realization, global triangulation choice, smooth transversality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite stratum inventory, frontier-incidence exactness, chart support, deformation admissibility, transport, replay, provenance, and local naming for BEDC.Derived.StratifiedSpaceUp.

Chapter 7378

A Concrete Naming Certificate for MetaCICResidualJoinBudget$^{\uparrow}$

The MetaCICResidualJoinBudget$^{\uparrow}$ packet realizes the confluence remaining-question surface of section 9386.3 as finite residual-join budget data. It sits beside the parallel-diamond frontier of chapter 4238, the residual-substitution boundary of chapter 2181, the closed checker seal of chapter 2182, and the confluence audit packet of chapter 735. The packet records typed-window, residual-pair, join-candidate, checker-status, premise-retention, obstruction, transport, replay, provenance, and local naming rows before any confluence consumer reads a local join.

Definition 7378.1 (MetaCIC residual-join budget carrier)**.** *A* MetaCICResidualJoinBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (W, P, R, Q, S, B, O, H, C, G, N)$$

with the following displayed rows:

1. *W is the typed-window row for the closed MetaCIC source under inspection;*
2. *P is the retained residual-substitution premise row read through* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*;*
3. *R is the residual-pair row, naming the two residual branches whose local join is budgeted;*
4. *Q is the proposed join-candidate row, kept visible until the bounded checker row accepts it;*
5. *S is the* ClosedTypeCheckerSeal$^{\uparrow}$ *row for the closed type-checker and finished-normal endpoint data used by the budget;*
6. *B is the bounded local-join checker row, recording the finite checker status rather than an unrestricted confluence theorem;*
7. *O is the obstruction-retention row for residual-substitution compatibility, premise-free parallel diamond, closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency endpoints;*
8. *H is the componentwise* hsame *transport row for W, P, R, Q, S, B, O;*
9. *C is the displayed* Cont *replay row for refining the same residual budget;*
10. *G is the* Pkg *provenance over* MetaCICResidualJoinBudget$^{\uparrow}$*,* MetaCICParallelDiamondFrontier$^{\uparrow}$*,* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*,* ClosedTypeCheckerSeal$^{\uparrow}$*,* MetaCICConfluenceAuditPacket$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinBudget}^{\uparrow}}$ *row.*
The carrier has no coordinate for residual-substitution compatibility, premise-free parallel diamond, arbitrary closed confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Theorem 7378.2 (MetaCIC residual-join budget NameCert obligations)**.** *For the carrier of Definition 7378.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinBudget}^{\uparrow}}$ *surface consists exactly of typed-window exposure through W, residual-substitution premise retention through P, residual-pair exposure through R, join-candidate exposure through Q, closed-checker sealing through S, bounded local-join status through B, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N.*

Proof. Project the displayed carrier. The typed window, residual-substitution premise, residual pair, join candidate, closed checker, bounded checker, and obstruction rows are all explicit coordinates of J.

Read the sibling confluence rows next. The residual-substitution boundary supplies the retained premise row P, the parallel-diamond frontier supplies the conditional local-diamond context, the closed checker seal supplies S, and the confluence audit packet supplies the consumer-facing audit boundary.

Finally attach transport, replay, provenance, and local naming. Since the carrier contains no row for residual-substitution compatibility, premise-free parallel diamond, arbitrary closed confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency, the stated NameCert surface exhausts every accepted residual-join budget read. □

Theorem 7378.3 (MetaCIC residual-join budget confluence route)**.** *Every confluence consumer of an accepted* $\mathsf{MetaCICResidualJoinBudget}^{\uparrow}$ *packet must first expose the finite route*

$$W, P, R, Q, S, B, O \longmapsto H, C, G, N$$

before reading any local join handoff. The route may feed $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *and* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *consumers, but it does not discharge residual substitution or turn the bounded checker row into a closed Church–Rosser theorem.*

Proof. Start with Definition 7378.1. The typed window W and residual-substitution premise P are visible before the residual pair R and join candidate Q are inspected.

Next read the checker rows. The closed type-checker seal S and bounded local-join checker B record only finite checker status for the displayed residual pair. The obstruction row O remains visible at the same stage.

Transport and replay preserve this route through H and C, while G and N keep the read inside the local package and NameCert rows. Thus the consumer receives a residual-join budget surface, not a premise-free confluence theorem. □

Falsifiable prediction: Every $\mathsf{MetaCICResidualJoinBudget}^{\uparrow}$ local-join read that omits the retained premise row P fails to distinguish bounded residual checking from a premise-free parallel diamond claim.

Independence witness: metacic-parallel-diamond-frontier, metacic-residual-substitution-boundary, closed-type-checker-seal, metacic-confluence-audit-packet: $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ carries the conditional diamond context but not this budget's closed checker and bounded status

rows; $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ carries the retained premise but not the join candidate; $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ carries checker sealing but not the confluence audit boundary; $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ carries audit status but not the local residual-pair budget. None is bijective to $\mathsf{MetaCICResidualJoinBudget}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualJoinBudget}^{\uparrow}$ packet is generated from a typed MetaCIC window, residual-substitution premise, residual-pair row, join-candidate row, closed type-checker seal, bounded local-join checker, obstruction-retention row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite carrier, NameCert obligation surface, and confluence route for bounded MetaCIC residual-join budgets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: It does not claim residual-substitution compatibility, premise-free parallel diamond, arbitrary closed confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, retained residual-substitution premise, residual-pair coverage, join-candidate exactness, closed-checker sealing, bounded local-join status, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualJoinBudgetUp.

Chapter 7379

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$ packet records the finite diagonal route by which a bounded real sequence is inspected through regular Cauchy subsequence windows before a $\mathsf{Real}^{\uparrow}$ cluster-facing seal is exposed. It sits over the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the bounded source surface of chapter 1415, and the Bolzano–Weierstrass route of chapter 1254. The packet names a displayed diagonal schedule and its finite readback; it is not countable choice, a host-selected subsequence, quotient stream equality, trichotomy of reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7379.1 (Regular Cauchy subsequence diagonal carrier)**.** *A* $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, S, J, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded sequence source row, read as a* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet and exposing only the inspected sequence indices, finite bounds, and local source route.*
2. *S is the subsequence schedule row, listing the finite initial segment of the increasing index schedule used by the diagonal read.*
3. *J is the diagonal window selector, assigning each requested regular-Cauchy tolerance to the finite source window and subsequence index already displayed in S.*
4. *W is the family of* $\mathsf{StreamName}^{\uparrow}$ *windows inspected along the displayed schedule and selector rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the bounded source, schedule, selector, windows, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the bounded source, schedule, selector, StreamName windows, RegSeqRat readback, Real seal, continuation row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay of the route*

$$B, S, J, W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets exactly through $B, S, J, W, R, E, H, C, P, N$. It has no coordinate for countable choice, a host subsequence selector, quotient equality of streams, trichotomy, or ambient Real completeness.

Theorem 7379.2 (Regular Cauchy subsequence diagonal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$ *carrier $D = (B, S, J, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows; bounded-source visibility through B; finite subsequence scheduling through S; diagonal window selection through J; StreamName window exposure through W; RegSeqRat readback through R; Real seal handoff through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7379.1. The bounded source row B is the only source-side input, so every sequence read starts from the finite $\mathsf{BoundedRealSequence}^{\uparrow}$ surface.

Read the schedule and selector next. The row S lists the finite inspected subsequence indices, while J ties each requested diagonal tolerance to one of the displayed windows. No row can request an unlisted tail or an externally selected subsequence.

The analytic handoff then passes through W, R, E. StreamName windows are inspected before the regular rational readback, and the Real seal is downstream of that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, a host subsequence selector, quotient stream equality, trichotomy, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7379.3 (Regular Cauchy subsequence diagonal Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$ *carrier factors through*

$$B, S, J, W, R, E, H, C, P, N.$$

In particular, the Real seal is exposed only after the bounded sequence source, finite subsequence schedule, diagonal selector, StreamName windows, and RegSeqRat readback have been displayed. The handoff exports no countable choice, selected host subsequence, quotient stream equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7379.2. They place the bounded source row B, schedule row S, selector row J, and StreamName windows W before any regular readback can be consumed.

The readback row R is therefore the only regular rational route to the terminal seal. A $\mathsf{Real}^{\uparrow}$ consumer reaches E only after this displayed readback has been named.

The structural rows H, C, P, N preserve the same route under transport and replay. Because no excluded selector, quotient, trichotomy, choice, or completeness row occurs in the carrier, every Real-facing read remains inside the displayed diagonal packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}$ packet is generated from a bounded real sequence source, a finite subsequence schedule, a diagonal window selector, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7379.1, Definition 7379.2, and Definition 7379.3 seed the finite diagonal route from bounded source windows to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected host subsequences, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source visibility, finite schedule exactness, diagonal selector stability, StreamName window coverage, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubsequenceDiagonalUp.

Chapter 7380

A Concrete Naming Certificate for $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$

Following Koksma and Hlawka, the $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$ packet records a finite numerical-integration error certificate whose only analytic inputs are a displayed discrepancy ledger and a displayed bounded-variation ledger. It sits after the low-discrepancy sequence surface of chapter 5173, the bounded-variation family surface of chapter 398, the finite epsilon-net tolerance row of chapter 1767, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite error handoff only; it does not import measure completion, selected representatives, quotient equality of functions, host total variation, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7380.1 (Koksma–Hlawka finite-error carrier)**.** *A* $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, V, E, Q, T, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{LowDiscrepancySequence}^{\uparrow}$ *source row, carrying the finite point schedule and the inspected star-discrepancy windows.*
2. *D is the discrepancy ledger, recording the finite dyadic upper bound read from S.*
3. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *row for the integrand, including the finite partition-variation ledger and endpoint readback surface.*
4. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *tolerance row that names the finite comparison scale used by the error request.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row joining the displayed discrepancy and variation budgets.*
6. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the certified quadrature error bound.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the finite error estimate.*
8. *H records componentwise* hsame *transport for the sequence, discrepancy, variation, epsilon-net, dyadic-budget, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from discrepancy and variation rows to the dyadic product budget, then to regular-rational readback and the terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$, $\mathsf{LowDiscrepancySequence}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, V, E, Q, T, R, H, C, P, N$. It has no coordinate for a measure-completion object, a selected representative of the integrand, quotient function equality, a host supremum over all partitions, or an ambient completeness theorem.

Theorem 7380.2 (Koksma–Hlawka NameCert obligations)**.** *For every accepted* $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$ *carrier $K = (S, D, V, E, Q, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}}$ *surface consists exactly of carrier admission for the low-discrepancy source S, discrepancy ledger D, bounded-variation ledger V, finite epsilon-net tolerance row E, dyadic product budget Q, regular-rational readback T, terminal real seal R, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 7380.1. The analytic rows visible to the certificate are the source schedule, discrepancy ledger, variation ledger, tolerance row, product budget, readback row, and terminal seal.

The sequence side is exhausted by S, D, E. The low-discrepancy row supplies the inspected point schedule, the discrepancy ledger supplies the dyadic upper bound, and the finite epsilon-net row supplies the tolerance scale at which the error request is read.

The integrand side is exhausted by V. Its bounded-variation ledger exposes only finite partition and endpoint-readback data before the product budget is available.

Finally Q, T, R, H, C, P, N multiply the displayed budgets, read the product as regular-rational data, seal the finite estimate as $\mathsf{Real}^{\uparrow}$, transport the same rows, replay the route, record provenance, and name the packet. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7380.3 (Koksma–Hlawka error handoff)**.** *Every quadrature-error consumer of an accepted* $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LowDiscrepancySequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

The route exports only the displayed discrepancy budget, bounded-variation budget, tolerance scale, dyadic product ledger, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Start with the NameCert obligations of Definition 7380.2. They force every consumer read to expose the sequence schedule, discrepancy ledger, variation ledger, epsilon-net tolerance, and dyadic product budget before any error estimate can be read.

Next read the dyadic budget row Q. It is the only coordinate where discrepancy and variation interact, so the consumer cannot bypass it through a measure-completion object, a hidden partition supremum, or a selected representative of the integrand.

The regular-rational row T and terminal seal R are downstream from Q. The structural rows H, C, P, N preserve that order under transport, replay, provenance, and local naming.

Thus the finite error handoff is exactly the displayed Koksma–Hlawka route and carries no additional analytic principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}}$.

Constructive story: A $\mathsf{KoksmaHlawkaFiniteError}^{\uparrow}$ packet is generated from a LowDiscrepancySequence source, a displayed discrepancy ledger, a BoundedVariation ledger, a FiniteEpsilonNet tolerance row, a DyadicRatCore product

budget, RegSeqRat readback, a RealUp error seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KoksmaHlawkaFiniteError}\uparrow}$, seed)

Scope closed: Definition 7380.1, Definition 7380.2, and Definition 7380.3 seed the finite Koksma–Hlawka error route over discrepancy, bounded-variation, epsilon-net, dyadic, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{KoksmaHlawkaFiniteError}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, measure completion, selected integrand representatives, quotient function equality, host total variation, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, discrepancy-ledger exactness, bounded-variation ledger exactness, epsilon-net tolerance routing, dyadic product-budget routing, RegSeqRat readback, RealUp error sealing, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.KoksmaHlawkaFiniteErrorUp.

Chapter 7381

A Concrete Naming Certificate for $\mathsf{VisionCoverageIndex}^{\uparrow}$

The $\mathsf{VisionCoverageIndex}^{\uparrow}$ packet realizes the top-level vision routing surface of chapter 9236 as finite BEDC data. It records which exploratory source label is covered by which concrete-instance target, which marker row witnesses the coverage, and which replay/provenance rows make the route auditable. The replay row is meant to inspect concrete target packets such as chapter 7380 and chapter 7378 before a coverage row is counted. The packet is an index certificate for vision coverage accounting; it is not a new mathematical theorem about the source chapters, not a semantic endorsement of every vision paragraph, and not a replacement for the concrete-instance chapters named by its rows.

Definition 7381.1 (Vision coverage index carrier)**.** *A* $\mathsf{VisionCoverageIndex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S, T, M, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-label row naming a vision chapter, section, or routing paragraph.*
2. *T is the concrete-target row naming the realized chapter label under* `concrete_instances/`*.*
3. *M is the marker ledger row containing the displayed* `\concretizedIn` *route from S to T.*
4. *L is the local coverage ledger, recording whether the row is root-index coverage, sibling coverage, bridge-facing coverage, or replay-only coverage.*
5. *R is the replay row that lets an audit reread the target chapter without relying on cached coverage counts.*
6. *H records componentwise* hsame *transport for source labels, target labels, marker rows, coverage ledgers, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from source label to target label through the marker ledger and the replay row.*
8. *P is the* Pkg *provenance over* $\mathsf{VisionCoverageIndex}^{\uparrow}$*, the referenced vision source, the referenced concrete-instance target,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{VisionCoverageIndex}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, M, L, R, H, C, P, N$. It has no coordinate for hidden coverage totals, cached root counts, semantic approval of source prose, or a target chapter that lacks a concrete label.

Theorem 7381.2 (Vision coverage index NameCert obligations)**.** *For every accepted* $\mathsf{VisionCoverageIndex}^{\uparrow}$ *carrier* $V = (S, T, M, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{VisionCoverageIndex}^{\uparrow}}$ *surface consists exactly of carrier admission for the source-label row* S*, concrete-target row* T*, marker ledger* M*, coverage ledger* L*, replay row* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming row* N*.*

Proof. Project the carrier of Definition 7381.1. The coverage data visible to the certificate are exactly the source label, target label, marker ledger, local coverage ledger, and replay row.

The marker row binds source to target. It records the displayed concretization route and therefore cannot be replaced by a cached count or by prose that merely resembles a coverage claim.

The replay row keeps the certificate auditable. It points back to the target chapter so that a consumer can inspect the finite NameCert rows named by T.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no hidden total or semantic-approval coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7381.3 (Vision coverage replay handoff)**.** *Every coverage consumer of an accepted* $\mathsf{VisionCoverageIndex}^{\uparrow}$ *packet factors through*

$$S \longmapsto M \longmapsto T \longmapsto R \longmapsto H, C, P, N.$$

Thus a vision source is counted as concretely covered only through a displayed marker ledger and a replayable concrete-instance target; the packet exports no hidden coverage count, no semantic endorsement of the source prose, and no target without a concrete chapter label.

Proof. Start with the NameCert obligations of Definition 7381.2. They require the source label, target label, marker ledger, coverage ledger, and replay row before any coverage consumer reads the packet.

The marker ledger M is the only source-to-target edge. The target row T is therefore not inferred from the source title or from a cached index; it is displayed by the packet.

The replay row R makes the target inspectable. The structural rows H, C, P, N preserve that route under transport, continuation replay, provenance, and local naming.

Consequently coverage accounting is finite BEDC data routed through the displayed marker and replay rows. □

Falsifiable prediction: Every accepted $\mathsf{VisionCoverageIndex}^{\uparrow}$ coverage read names a source label, concrete target label, marker ledger, replay row, transport row, continuation row, provenance row, and local NameCert row before it is counted.

Independence witness: kernelacceptancewitness, inscriptionevent, closure-capstone-realization, finitekernelcategory, koksma-hlawka-finite-error: $\mathsf{VisionCoverageIndex}^{\uparrow}$ is not bijective to any listed sibling because it indexes coverage routes among source labels and concrete targets rather than carrying the target's mathematical or audit payload.

Lean Nontrivial and FieldFaithful TasteGate instances for $\mathsf{VisionCoverageIndex}^{\uparrow}$ should inspect the carrier rows directly: Nontriviality is witnessed by distinct source-label and concrete-target

rows, while field faithfulness requires projections of $S, T, M, L, R, H, C, P, N$ to preserve the displayed marker-ledger and replay route. Neither instance may read hidden coverage totals, cached routing weights, or source prose outside the displayed marker row.

Closure status for $\mathsf{NameCert}_{\mathsf{VisionCoverageIndex}^{\uparrow}}$.

Constructive story: A $\mathsf{VisionCoverageIndex}^{\uparrow}$ packet is generated from source-label rows, concrete-target rows, marker-ledger rows, local coverage ledgers, replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VisionCoverageIndex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7381.1, Definition 7381.2, and Definition 7381.3 seed the finite routing packet that turns vision coverage accounting into displayed source-label, target-label, marker-ledger, replay, transport, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VisionCoverageIndex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, source-prose semantics, hidden coverage totals, cached routing weights, or the NameCert obligations of any target chapter named by the index.

Upgrade path: Obligation closure requires checked carrier admission, source-label exactness, concrete-target exactness, marker-ledger routing, coverage-ledger classification, replay handoff, transport stability, continuation replay, provenance nonescape, and local naming for BEDC.Derived.VisionCoverageIndexUp.

Chapter 7382

A Concrete Naming Certificate for $\mathsf{HawaiianEarring}^{\uparrow}$

Following Eilenberg–Steenrod and classical topology, the $\mathsf{HawaiianEarring}^{\uparrow}$ packet records the compact metric boundary surface of the countable family of tangent circles whose radii converge to zero. It reads the S1Up circle certificate of chapter 58, the compact-metric surface of chapter 129, the topological carrier of chapter 88, the stream-name route of chapter 126, the regular rational sequence route of chapter 301, and the real-coordinate seal of chapter 38. The chapter names the displayed compact metric wedge boundary only; it does not compute a fundamental group or identify a quotient wedge.

Definition 7382.1 (Hawaiian earring carrier)**.** *A* $\mathsf{HawaiianEarring}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, R, S, T, K, Q, W, H, D, P, N)$$

with the following displayed rows:

1. *C is the circle-source row, read through S1Up and its* $\mathsf{Real}^{\uparrow}$ *coordinate surface, naming the reusable unit-circle certificate.*
2. *R is the radius schedule row, read through* $\mathsf{RegSeqRat}^{\uparrow}$*, displaying a positive rational radius for each inspected circle and its convergence toward the common tangent point.*
3. *S is the stream-name row, read through* $\mathsf{StreamName}^{\uparrow}$*, recording the finite stream observations by which requested circle indices and rational radii are presented.*
4. *T is the topology row, read through* $\mathsf{Topology}^{\uparrow}$*, naming the subspace and neighborhood interface available at the wedge point and at displayed circle points.*
5. *K is the compact-metric row, read through* $\mathsf{CompactMetric}^{\uparrow}$*, exposing only the finite compact-metric observations attached to the displayed radius schedule.*
6. *Q is the circle-wedge incidence row, recording the shared tangent point and the finite membership routes for the inspected circle pieces.*
7. *W is the wedge-boundary readback row, recording which displayed circle and radius row supplies a requested local observation.*
8. *H records componentwise* hsame *transport for the circle source, radius schedule, stream name, topology, compact-metric row, incidence row, wedge-boundary readback, replay, provenance, and local naming.*

9. *D records displayed* Cont *replay from stream-name admission through regular-rational radius readback, circle-source inspection, topology exposure, compact-metric observation, and wedge-boundary readback.*

10. *P is the* Pkg *provenance over* $\mathsf{HawaiianEarring}^{\uparrow}$*, S1Up,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HawaiianEarring}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a full fundamental-group computation, quotient wedge equality, selected points from infinitely many circles, ambient compactness beyond the displayed compact-metric rows, Lean verification, or bridge data.

Theorem 7382.2 (Hawaiian earring NameCert obligations)**.** *For every accepted* $\mathsf{HawaiianEarring}^{\uparrow}$ *carrier $E = (C, R, S, T, K, Q, W, H, D, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HawaiianEarring}^{\uparrow}}$ *obligation surface consists exactly of circle-source admission, regular rational radius scheduling, stream-name readback, topology exposure, compact-metric observation, circle-wedge incidence, wedge-boundary readback, componentwise transport, displayed replay, provenance, and local naming. A consumer may inspect the compact metric boundary packet only after those rows have been exposed from the same* BHist *carrier.*

Proof. Project the carrier of Definition 7382.1. The rows $C, R, S, T, K, Q, W, H, D, P, N$ are the only rows exposed by an accepted packet.

Read the source and schedule rows first. The S1Up circle row C, regular rational sequence row R, and stream-name row S jointly determine which displayed circles and radii may be inspected before any topology or compact-metric consumer enters.

Next read T, K, Q, W. These rows expose the topology, compact-metric observation, shared tangent point, finite incidence data, and wedge-boundary readback, while keeping all observations attached to the same displayed schedule.

Finally H, D, P, N transport, replay, source, and name that packet. Since no carrier coordinate stores a fundamental group, quotient wedge equality, a choice schedule over all circles, ambient compactness beyond K, Lean verification, or bridge data, the listed rows exhaust the NameCert obligations. □

Theorem 7382.3 (Hawaiian earring circle wedge route)**.** *Every accepted* $\mathsf{HawaiianEarring}^{\uparrow}$ *consumer read of the wedge-boundary row W must factor through*

S1Up, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The route exports the displayed compact metric boundary packet for the tangent circle family. It does not export a full fundamental-group computation, quotient wedge equality, selected representatives from infinitely many circles, ambient compactness beyond the displayed compact-metric rows, Lean verification, or bridge checking.

Proof. Start with the NameCert obligation surface of Definition 7382.2. It requires the circle source, rational radius schedule, and stream-name rows before the topology and compact-metric rows can be inspected.

The incidence and wedge-boundary rows Q, W are downstream of that schedule. They identify the shared tangent point and the displayed circle piece used by a local observation, but they do not add a quotient-space equality or an infinite choice principle.

Transport, replay, provenance, and local naming preserve the same finite route. Therefore every wedge-boundary read remains a compact metric/topology boundary read over the displayed circle family and cannot be reinterpreted as any of the refused endpoints. □

Falsifiable prediction: Every $\mathsf{HawaiianEarring}^{\uparrow}$ consumer read factors through a S1Up circle-source row, a $\mathsf{RegSeqRat}^{\uparrow}$ radius schedule, a $\mathsf{StreamName}^{\uparrow}$ index-read row, a $\mathsf{Topology}^{\uparrow}$ neighborhood row, a $\mathsf{CompactMetric}^{\uparrow}$ observation row, a circle-wedge incidence row, a wedge-boundary readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: s1, compactmetric, topology, streamname, regseqrat, real: $\mathsf{HawaiianEarring}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a countable tangent-circle schedule, compact metric observations, topology exposure, stream-name readback, and wedge-boundary incidence as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HawaiianEarring}^{\uparrow}}$.

Constructive story: A $\mathsf{HawaiianEarring}^{\uparrow}$ packet is generated from a S1Up circle source, RegSeqRatUp radius schedule, StreamNameUp index-read row, TopologyUp neighborhood row, CompactMetricUp observation row, circle-wedge incidence row, wedge-boundary readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HawaiianEarring}^{\uparrow}}$, seed)

Scope closed: Definition 7382.1, Definition 7382.2, and Definition 7382.3 seed the compact metric/topology boundary packet for the displayed tangent circle family.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HawaiianEarring}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full fundamental-group computation, quotient wedge equality, choice over infinitely many circles, ambient compactness beyond displayed compact-metric rows, Lean verification, or bridge checking.

Upgrade path: A formal route can encode BEDC.Derived.HawaiianEarringUp and prove the carrier, NameCert obligation, and circle-wedge route theorems.

Chapter 7383

A Concrete Naming Certificate for RiemannDarbouxCriterionFinite$^{\uparrow}$

Following Bishop and Bridges on constructive Riemann integration, the RiemannDarbouxCriterionFinite$^{\uparrow}$ packet records the finite criterion that identifies a Riemann-sum Cauchy route with an upper/lower Darboux squeeze before a Real$^{\uparrow}$ integral seal is exported. It sits between the finite Darboux boundary of chapter 1880, the tagged Riemann-sum route of chapter 1615, the dyadic mesh source of chapter 358, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives RiemannIntegrable$^{\uparrow}$ and regulated-integral consumers a finite bridge from partition-sum criteria to real-sealed integration data, not a host theorem about arbitrary partitions, quotient function equality, measure completion, or ambient Real$^{\uparrow}$ completeness.

Definition 7383.1 (Riemann–Darboux finite criterion carrier)**.** *A* RiemannDarbouxCriterionFinite$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, U, L, S, R, D, O, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *M is the tagged partition mesh row, read through* DyadicMesh$^{\uparrow}$ *and the finite tag ledger used by the* RiemannSum$^{\uparrow}$ *consumer.*
2. *U is the upper Darboux-sum row, read through* ***??*** *or the upper side of the* DarbouxSum$^{\uparrow}$ *packet over the same mesh row M.*
3. *L is the lower Darboux-sum row, read through* LowerDarbouxSum$^{\uparrow}$ *or the lower side of the* DarbouxSum$^{\uparrow}$ *packet over the same mesh row M.*
4. *S is the Riemann-sum Cauchy-net read, exposing only displayed tagged sums and finite refinement witnesses.*
5. *R is the dyadic mesh refinement row, requiring every refinement consumed by S, U, L to factor through the same* DyadicMesh$^{\uparrow}$ *surface.*
6. *D is the oscillation ledger, recording the upper-minus-lower squeeze and the cellwise oscillation comparisons that license the finite criterion.*
7. *O is the* RegSeqRat$^{\uparrow}$ *readback row for the gap, tagged-sum, and tolerance windows.*
8. *Q is the* Real$^{\uparrow}$ *seal row, reached only after the Darboux squeeze and regular rational readback have been displayed.*

9. *E records hsame transport for the mesh, Darboux rows, Riemann-sum Cauchy-net read, refinement row, oscillation ledger, readback, seal, replay, provenance, and local name.*
10. *H records the displayed* Cont *replay from mesh rows to Darboux sums, from mesh rows to tagged sums, from refinement rows to the squeeze ledger, and from readback rows to the Real seal.*
11. *T is the* Pkg *provenance over* $\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, *??*, $\mathsf{LowerDarbouxSum}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *P is the local integration-boundary provenance row naming the downstream* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and regulated-integral consumers that may read the criterion.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected refining sequence, quotient equality of integrands, measure-theoretic integration, host compactness, or ambient real completeness.

Theorem 7383.2 (Riemann–Darboux finite criterion NameCert obligations)**.** *For every accepted* $\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}$ *carrier*

$$C = (M, U, L, S, R, D, O, Q, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed tagged partition mesh, upper and lower Darboux rows, Riemann-sum Cauchy-net read, refinement row, oscillation ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *mesh-refinement classifier transport, requiring every refinement used by the Darboux rows or Riemann-sum row to be compared through the same* $\mathsf{DyadicMesh}^{\uparrow}$ *row M and refinement route R;*
3. *upper/lower squeeze exactness, requiring the rows U, L, D to expose the finite Darboux upper-minus-lower comparison before any integral-facing consumer reads the criterion;*
4. *Riemann-sum Cauchy handoff, requiring the row S to pass through the same mesh-refinement surface and then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row O;*
5. *Real seal nonescape, requiring Q, E, H, T, P, N to preserve, replay, source, and name only the displayed finite rows before a* $\mathsf{Real}^{\uparrow}$ *consumer receives the seal.*

Proof. Project the carrier of Definition 7383.1. The rows $M, U, L, S, R, D, O, Q, E, H, T, P, N$ are the whole packet, so carrier admission exposes exactly the finite mesh, Darboux, Riemann-sum, readback, seal, transport, replay, provenance, and naming data.

Mesh-refinement transport is forced by M and R. Both the Darboux rows and the tagged Riemann-sum row must read their cells through the same displayed $\mathsf{DyadicMesh}^{\uparrow}$ source, so a refinement changes the packet only by replaying the common finite route.

The squeeze rows U, L, D are read before the Cauchy handoff is accepted. They provide the upper and lower Darboux comparison over the displayed mesh, while S supplies the tagged-sum Cauchy-net read over the same refinement surface.

Finally O and Q form the terminal readback and seal. The structural rows E, H, T, P, N transport, replay, source, and name that route without adding a selected refining sequence, quotient integrand equality, measure object, host compactness theorem, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}$ packet is generated from a tagged DyadicMesh row, upper and lower Darboux-sum rows, a Riemann-sum Cauchy-net read, a mesh-refinement row, an oscillation ledger, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, downstream integration-boundary provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7383.1 and Definition 7383.2 seed the finite Riemann–Darboux criterion over $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, **??**, $\mathsf{LowerDarbouxSum}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Riemann integrability, selected refining sequences, quotient equality of integrands, measure-theoretic integration, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh-refinement classifier transport, Darboux squeeze exactness, Riemann-sum Cauchy handoff, Real seal nonescape, replay, provenance, and local naming for BEDC.Derived.RiemannDarbouxCriterionFiniteUp.

Chapter 7384

A Concrete Naming Certificate for $\mathsf{CauchyModulusThinning}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusThinning}^{\uparrow}$ packet records the finite route by which an explicit Cauchy modulus is thinned to a monotone cofinal schedule before a regular rational or real-name consumer reads it. It sits over the modulus surface of chapter 292, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet is a non-quotient thinning certificate; it supplies no selected limit, quotient stream equality, countable-choice schedule, host real equality, or ambient completeness theorem.

Definition 7384.1 (Cauchy modulus thinning carrier)**.** *A* $\mathsf{CauchyModulusThinning}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (M, T, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{CauchyModulus}^{\uparrow}$ *row, carrying the original explicit modulus request and its finite threshold answers.*
2. *T is the monotone thinning route, recording the retained threshold schedule and the finite proof that each retained threshold dominates the source answer needed at the same dyadic depth.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic precision, scale, and comparison rows used to read the source and thinned schedules.*
4. *W is the retained* $\mathsf{StreamName}^{\uparrow}$ *window opened at the thinned threshold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the retained window and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, T, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the modulus, thinning, tolerance, window, regular handoff, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source modulus through the thinned threshold, dyadic ledger, retained window, regular handoff, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusThinning}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusThinning}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, D, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7384.2 (Cauchy modulus thinning NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusThinning}^{\uparrow}$ *carrier $X = (M, T, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusThinning}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, source-modulus exposure through M, monotone thinning dominance through T, dyadic tolerance exactness through D, retained stream-window visibility through W, regular rational handoff through R, real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7384.1. The analytic rows are exactly M, T, D, W, R, E, and the structural rows are exactly H, C, P, N.

The source-modulus and thinning obligations are read first. Row M supplies the explicit threshold answer, while row T records the retained monotone schedule and its finite dominance comparison against the source answer at the displayed dyadic depth.

The tolerance and window obligations then force the read through D and W. The dyadic ledger fixes the precision scale, and the retained stream window is opened only at the thinned threshold named by T.

Finally the regular and real-facing rows are downstream. The row R is obtained from the retained window and tolerance ledger, and the seal row E is reachable only after that handoff. Since H, C, P, N preserve the same route and the carrier lists no quotient, selected limit, choice schedule, host equality, or ambient completeness row, the displayed obligations exhaust the local NameCert surface. □

Theorem 7384.3 (Cauchy modulus thinning Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusThinning}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{CauchyModulus}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the monotone thinning row T retained before any stream window is opened. Thus downstream regular-Cauchy and completion packets may consume a thinned explicit modulus without importing quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness.

Proof. Begin with an accepted packet and read the source modulus. The row M answers the displayed dyadic request, while T records a monotone retained threshold that dominates the source answer for that request.

Read the dyadic and stream rows next. The ledger D fixes the tolerance scale, and the stream window W is opened only at the retained threshold. Therefore a regular rational consumer cannot bypass the thinning route.

The handoff to R and then E is forced by the same finite rows. Componentwise transport, replay, provenance, and local naming preserve this route, but they do not add a selected real, a quotient equality, countable choice, host equality, or an ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{CauchyModulusThinning}^{\uparrow}$ real-facing read displays a source Cauchy modulus row, a monotone thinning row, a dyadic tolerance ledger, a retained StreamName window, a RegSeqRat handoff, and a RealUp seal before any downstream consumer receives the packet.

Independence witness: cauchymodulus, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{CauchyModulusThinning}^{\uparrow}$ because the thinning packet jointly carries source modulus data, a monotone retained-threshold route, dyadic tolerance accounting, stream-window retention, regular rational handoff, terminal real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusThinning}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusThinning}^{\uparrow}$ packet is generated from a source CauchyModulus row, a monotone thinning route, a DyadicRatCore tolerance ledger, a retained StreamName window, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusThinning}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7384.1, Definition 7384.2, and Definition 7384.3 seed the non-quotient modulus-thinning route over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusThinning}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close obligation coverage, Lean theorem verification, quotient stream equality, selected limits, countable choice, host real equality, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-modulus exposure, thinning dominance, dyadic tolerance exactness, retained-window visibility, regular handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusThinningUp.

Lean4 (statement) `BEDC.Derived.CauchyModulusThinningUp`

Chapter 7385

A Concrete Naming Certificate for $\mathsf{CousinPartition}^{\uparrow}$

Following Cousin and Bishop–Bridges gauge integration, the $\mathsf{CousinPartition}^{\uparrow}$ packet records the finite gauge-partition object that sits between the existence route of chapter 1861 and the tagged-sum consumers of $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, and $\mathsf{RiemannSumGauge}^{\uparrow}$. It names a located interval, a positive gauge, finitely many tagged subintervals, mesh and refinement rows, gauge-fineness witnesses, finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, displayed replay, provenance, and local naming. The packet does not assert open-cover compactness, an arbitrary host partition object, selected representatives outside the displayed ledger, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7385.1 (Cousin partition carrier)**.** *A* $\mathsf{CousinPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, G, T, M, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located closed interval row, including endpoint names, interval containment, and the finite source boundary inherited from the Cousin route.*
2. *G is the positive gauge row, recording local positive-radius data as displayed rational or regular-rational comparisons rather than as an ambient open cover.*
3. *T is the finite tagged-subinterval ledger. Each entry carries a cell, a tag, endpoint reads, tag-in-cell containment, and its position in the ordered partition.*
4. *M is the mesh and refinement ledger, recording adjacency, common refinements, and the comparison route to the tagged-real-partition mesh.*
5. *F is the gauge-fineness row, certifying for every displayed tagged cell that the cell lies inside the local gauge radius attached to its tag.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for endpoints, tags, gauge-radius reads, mesh comparisons, and fineness checks.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and positive-radius comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the interval, gauge, tagged-cell, mesh, fineness, window, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for I, G, T, M, F, S, R, E and for replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a Cousin-partition request through*

$$I, G, T, M, F, S, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CousinPartition}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed interval, gauge, tagged-subinterval, mesh, fineness, readback, seal, transport, replay, provenance, and local naming rows.

Theorem 7385.2 (Cousin partition NameCert obligations)**.** *For every accepted* $\mathsf{CousinPartition}^{\uparrow}$ *carrier $K = (I, G, T, M, F, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CousinPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval exposure through I, positive gauge admission through G, finite tagged-subinterval exposure through T, mesh and refinement accountability through M, gauge-fineness certification through F, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seals through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7385.1. The tuple lists the only rows available to a Cousin-partition consumer: interval source, positive gauge, tagged cells, mesh and refinement data, gauge-fineness witnesses, finite windows, regular readback, Real seals, transport, replay, provenance, and local naming.

Read the partition-facing rows in order. The Cousin lemma chapter supplies the existence-facing route to finite tagged subintervals, but the present packet records the displayed ledger T together with the mesh row M and fineness row F before any tagged-sum consumer can inspect it.

Finally read S, R, E, H, C, P, N. These rows keep endpoint, tag, and gauge-radius observations inside finite StreamName windows and regular-rational readback before the Real seals are exposed. Since no carrier coordinate names an open cover, host partition, selected representative family, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7385.3 (Cousin partition gauge-refinement handoff)**.** *Let $K = (I, G, T, M, F, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CousinPartition}^{\uparrow}$ *carrier. Any consumer that refines a Cousin subdivision toward* $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, *or* $\mathsf{RiemannSumGauge}^{\uparrow}$ *must first expose the located interval row I, positive gauge row G, finite tagged-subinterval ledger T, mesh and refinement ledger M, gauge-fineness row F, regular readback row R, and terminal Real seal row E. The handoff exports a finite gauge-fine partition route only; it does not export an ambient open-cover theorem, an arbitrary host mesh, quotient equality of partitions, selected tag values outside T, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the Cousin source of chapter 1861. It supplies the finite subdivision route over the displayed interval and positive gauge, but the consumer still receives only the rows named in I, G, T, M, F.

Next route the tagged cells through the sibling tagged-partition and fine-gauge surfaces. The tagged row T is the only source for $\mathsf{TaggedRealPartition}^{\uparrow}$ reads, while F is the row that permits $\mathsf{FinePartitionGauge}^{\uparrow}$ to see the partition as gauge-fine.

Finally route a Riemann-sum or gauge-integration consumer through $\mathsf{RiemannSumGauge}^{\uparrow}$. The mesh row M, readback row R, Real seal row E, and structural rows H, C, P, N preserve the same finite partition ledger, so the downstream consumer receives a displayed gauge-refinement handoff rather than hidden compactness, quotient equality, selected tags, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{CousinPartition}^{\uparrow}$ packet is generated from a located interval row, positive gauge row, finite tagged-subinterval ledger, mesh and refinement ledger, gauge-fineness row, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7385.1, Definition 7385.2, and Definition 7385.3 seed the finite gauge-partition route from CousinLemmaUp to TaggedRealPartitionUp, FinePartitionGaugeUp, RiemannSumGaugeUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CousinPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, host partition equality, selected tags outside the displayed ledger, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive gauge admission, tagged-subinterval exposure, mesh refinement, gauge-fineness, readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CousinPartitionUp.

Chapter 7386

A Concrete Naming Certificate for $\mathsf{BishopIntervalOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalOrder}^{\uparrow}$ packet records constructive interval-order data over displayed real-name endpoints without importing a trichotomy oracle. It sits after the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the finite stream-window discipline of chapter 126. The packet gives Real and Completion consumers an apartness-routed interval comparison boundary; it is not a quotient interval construction, not host real equality, not ambient $\mathsf{Real}^{\uparrow}$ completeness, and not a proof that every two intervals are classically comparable.

Definition 7386.1 (Bishop interval order carrier)**.** *A* $\mathsf{BishopIntervalOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, A, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{Real}^{\uparrow}$ *endpoint row, exposed only as a named real boundary.*
2. *U is the upper* $\mathsf{Real}^{\uparrow}$ *endpoint row, paired with L by the same interval carrier.*
3. *A is the located-apartness witness row, recording positive separation evidence when the order comparison is read constructively.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare finite endpoint observations.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite endpoint reads inspected by the tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the lower and upper endpoints after S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for endpoints, apartness, dyadic tolerances, stream windows, regular readback, real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from an interval-order request through L, U, A, D, S, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalOrder}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, U, A, D, S, R, E, H, C, P, N$. It has no coordinate for real trichotomy, quotient interval equality, selected representatives, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7386.2 (Bishop interval order NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalOrder}^{\uparrow}$ *carrier $I = (L, U, A, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower endpoint, upper endpoint, located-apartness witness, dyadic tolerance ledger, stream-window row, regular rational readback, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 7386.1. Its interval data are exactly the endpoint rows L, U, the apartness row A, the dyadic tolerance row D, the finite window row S, the regular readback row R, and the terminal seal row E.

The endpoint rows are read through the same finite observation route. The stream windows and regular rational readback provide the displayed data that the dyadic tolerance ledger may compare, and the apartness row records only positive separation evidence visible through that route.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for trichotomy, quotient interval equality, host real equality, selected representatives, or ambient completeness, no further local NameCert clause can be read from it. □

Theorem 7386.3 (Bishop interval order apartness route)**.** *Every constructive order read exported by an accepted* $\mathsf{BishopIntervalOrder}^{\uparrow}$ *carrier factors through*

$$L, U \longmapsto S, R, D \longmapsto A \longmapsto E \longmapsto H, C, P, N.$$

Thus a consumer may use displayed located-apartness data to separate endpoint intervals, but it cannot replace that route by trichotomy, quotient intervals, host real equality, selected endpoint representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7386.2. Any interval-order consumer must first read the lower and upper endpoint rows together with their finite stream windows and regular rational readback.

The dyadic tolerance ledger is the comparison budget for those reads. A positive order separation is therefore available only through the located-apartness row A, which is downstream of the displayed endpoint observations.

The terminal real seal and structural rows do not add a classical comparison principle. They only seal, transport, replay, source, and name the same apartness-routed interval packet.

Consequently every exported constructive order read factors through the displayed route, and all non-displayed real-comparison principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopIntervalOrder}^{\uparrow}$ consumer must expose lower and upper Real endpoint rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a located-apartness row, Real seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before reading an interval-order comparison.

Independence witness: Nearest siblings: Real, RegSeqRat, DyadicRatCore, and StreamName. $\mathsf{BishopIntervalOrder}^{\uparrow}$ is not bijective to any one sibling because it jointly carries paired endpoint rows, finite windows, regular readback, tolerance ledgers, located-apartness evidence, terminal

seals, transport, replay, provenance, and local naming for the same constructive interval-order route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalOrder}^{\uparrow}$ packet is generated from lower and upper Real endpoint rows, located-apartness evidence, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback, Real seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7386.1, Definition 7386.2, and Definition 7386.3 seed the finite apartness-routed interval-order boundary over displayed Real endpoints, regular readback, dyadic tolerances, stream windows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, quotient intervals, host real equality, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located-apartness routing, dyadic tolerance visibility, StreamName window admission, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalOrderUp.

Chapter 7387

A Concrete Naming Certificate for $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy estimates, the $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$ packet records the finite tail-estimate route by which two regular rational Cauchy sources, their aligned tail windows, and the displayed pointwise-sum budget become a real-facing seal. It sits over the Cauchy carrier of chapter 3413, the regular rational source of chapter 301, the terminal real-name boundary of chapter 38, the dyadic tolerance arithmetic of chapter 291, and the binary sum surface of chapter 417. The packet names finite tail estimates for the sum route; it is not a host limit theorem, quotient stream equality, selected convergence witness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7387.1 (Regular Cauchy sum tail-estimate carrier)**.** *A* $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, W_A, W_B, D_A, D_B, S, E, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose tails are compared on the same precision request.*
2. *W_A and W_B are the aligned* $\mathsf{StreamName}^{\uparrow}$ *tail windows opened for the two sources.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows attached to the two tail estimates.*
4. *S is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row that combines the two displayed source windows through pointwise dyadic addition.*
5. *E is the finite error ledger recording the displayed tail bound for the sum after both source bounds have been read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the sum window and its tail ledger.*
7. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after $A, B, W_A, W_B, D_A, D_B, S, E, R$ have been displayed.*
8. *H records componentwise* hsame *transport for the two sources, windows, dyadic rows, sum row, ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tail request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySumTailEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient real completeness.

Theorem 7387.2 (Regular Cauchy sum tail-estimate NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$ *carrier*

$$T = (A, B, W_A, W_B, D_A, D_B, S, E, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySumTailEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular rational sources, aligned tail windows, dyadic tolerance rows, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row, the finite error ledger, regular rational readback, real seal, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7387.1. The accepted packet exposes precisely the rows $A, B, W_A, W_B, D_A, D_B, S, E, R, Q, H, C, P, N$, so carrier admission and local naming have no hidden coordinates.

The tail estimate is then read in order. The two source rows A, B are inspected through the aligned windows W_A, W_B and the dyadic tolerance rows D_A, D_B before the pointwise-sum row S is available.

The finite ledger E records the displayed output bound for the summed tail, and the readback row R is downstream of that ledger. The real seal Q is therefore reached only through the displayed regular rational route.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no row stores quotient streams, selected limits, host equality, trichotomy, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7387.3 (Regular Cauchy sum tail-estimate real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$ *packet reaches the seal row Q only after the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, aligned* $\mathsf{StreamName}^{\uparrow}$ *tail windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegularCauchySum}^{\uparrow}$ *row, finite error ledger, and output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row have been displayed. The route exports no quotient stream equality, selected limit, host real equality, trichotomy, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 7387.2. They force the two source rows and their aligned windows to be visible before the sum row or output readback is consumed.

Read the sum route through chapter 417. The binary sum packet supplies only the displayed pointwise-addition and finite-budget surface, so the tail estimate remains tied to D_A, D_B, S, E.

The output readback R is therefore a regular rational row over the displayed sum ledger. The real seal Q is downstream of R, while H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

No carrier coordinate supplies the excluded host or choice principles. Hence every Real-facing read of the packet is the finite route through Cauchy, RegSeqRat, DyadicRatCore, RegularCauchySum, and Real rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySumTailEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySumTailEstimate}^{\uparrow}$ packet is generated from two RegSeqRat source rows, aligned StreamName tail windows, DyadicRatCore tolerance rows, a RegularCauchySum row, a finite error ledger, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySumTailEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7387.1, Definition 7387.2, and Definition 7387.3 seed the finite L10 tail-estimate route for regular Cauchy sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySumTailEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, aligned tail-window coverage, dyadic tolerance exactness, sum-ledger stability, readback handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySumTailEstimateUp.

Chapter 7388

A Concrete Naming Certificate for $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$

The $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ packet realizes the bounded observation branch of the cellular substrate reading as finite Rule 110 NameCert data. It records a local window, a bounded time depth, the selected Rule 110 update ledger, and the observed halt tag for that finite window only. It sits beside the pattern classifier of chapter 5844, the causal-cone route of chapter 5846, the local-stencil observation of chapter 6004, and the Rule 110 cellular trust substrate of chapter 6517. The packet is a bounded local observation surface, not a global halting theorem, not a universal decidability claim, and not an oracle for infinite Rule 110 evolutions.

Definition 7388.1 (Rule 110 halting-window carrier)**.** *A* $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (W, T, L, C, O, B, K, S, R, P, N)$$

with the following displayed rows:

1. *W is the finite Rule 110 row window being inspected.*
2. *T is the bounded time-depth row, giving the finite number of update steps admitted by the observation.*
3. *L is the local transition-table ledger for the Rule 110 update rule on the cells that can influence W within depth T.*
4. *C is the* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *row naming the finite cone through which the observation is replayed.*
5. *O is the orbit-window row containing only the finite states reached before the depth bound is exhausted.*
6. *B is the boundary row recording the left and right finite context cells required by the cone.*
7. *K is the classifier row, read through* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *or* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$*, that tags the final bounded observation as halted, active, boundary-blocked, or unresolved inside the displayed window.*
8. *S records componentwise* hsame *transport for the window, depth, ledger, cone, orbit, boundary, classifier, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from the initial window through the bounded cone and classifier rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$, $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$, $\mathsf{Rule110CellularTrustSubstrate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenHaltingWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an unbounded orbit, a global halting predicate, an undecidable continuation oracle, or a hidden host-side simulation result.

Theorem 7388.2 (Rule 110 halting-window NameCert obligations)**.** *For every accepted* RuleOneTenHaltingWindow *carrier*

$$H = (W, T, L, C, O, B, K, S, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenHaltingWindow}^{\uparrow}}$ *surface consists exactly of finite-window admission, bounded-depth admission, local transition-table exposure, causal-cone replay, orbit-window exposure, boundary-cell exposure, classifier-tag exactness, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7388.1. The rows $W, T, L, C, O, B, K, S, R, P, N$ are the whole packet, so every obligation is attached to a displayed finite coordinate.

The local update obligation reads L only over the cells admitted by W, T, C, and B. Thus the orbit row O contains bounded states of the selected cone rather than an unbounded evolution.

The classifier row K is downstream of that bounded orbit. It may tag the finite observation as halted, active, boundary-blocked, or unresolved inside the window, but it has no row from which to decide a global halting question.

Finally, S, R, P, N transport, replay, source, and name the same finite surface. Since no carrier coordinate stores an unbounded orbit, global halting predicate, continuation oracle, or hidden simulation result, the listed obligations exhaust the local NameCert surface. □

Theorem 7388.3 (Rule 110 halting-window causal-cone route)**.** *Every accepted* $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ *observation factors through the finite causal cone*

$$W, T, L, B \longmapsto C \longmapsto O \longmapsto K$$

before its local NameCert *row is read. The route records a bounded halting-window observation for Rule 110; it does not decide whether an arbitrary Rule 110 configuration eventually halts outside the displayed window and depth.*

Proof. Begin with the finite window W and the depth row T. The local transition ledger L and boundary row B determine exactly which cells can influence the displayed observation before the bound is exhausted.

Apply the causal-cone sibling of chapter 5846. The cone row C repacks the window, depth, transition ledger, and boundary rows as a finite replay route.

The orbit row O is read only after the cone is displayed, and the classifier row K is read only after O. Thus the packet can report the bounded observation state without asserting anything about cells or times outside the carrier.

The structural rows S, R, P, N preserve this bounded route. Therefore the packet supplies a finite halting-window observation and no global halting theorem. □

Falsifiable prediction: Every accepted $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ read exposes a finite row window, a bounded time-depth row, the local Rule 110 transition ledger, a causal-cone row, boundary cells, the orbit window, a classifier tag, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a bounded halt tag is visible.

Independence witness: rule110-pattern-classifier, rule-one-ten-causal-cone, rule110-local-stencil-observation, rule110-cellular-trust-substrate, rule110-cook-frontier-budget: $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded time-depth row, causal cone, orbit window, boundary context, halt-state classifier, replay, provenance, and local naming for one finite observation packet.

The Lean TasteGate follow-up for this packet must provide Nontrivial $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ instances for the displayed rows $W, T, L, C, O, B, K, S, R, P, N$ before the bounded observation can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenHaltingWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenHaltingWindow}^{\uparrow}$ packet is generated from a finite Rule 110 row window, bounded time-depth row, local transition ledger, causal cone, orbit window, boundary context, classifier tag, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenHaltingWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7388.1, Definition 7388.2, and Definition 7388.3 seed bounded finite-window halting observations for Rule 110.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenHaltingWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global halting decidability, unbounded orbit classification, universal continuation prediction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, bounded-depth admission, transition-table exposure, causal-cone replay, orbit-window exposure, classifier exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenHaltingWindowUp.

Chapter 7389

A Concrete Naming Certificate for $\mathsf{KTopology}^{\uparrow}$

Following Kelley and Munkres on point-set topology, the $\mathsf{KTopology}^{\uparrow}$ packet records the finite naming surface for the real line with the deleted-neighbourhood discipline around a distinguished closed subset K. It sits over the real order topology of chapter 1890, the neighbourhood interface of chapter 88, the observable-open handoff of $\mathsf{SierpinskiSpace}^{\uparrow}$ in chapter 1860, and the finite-window route of chapter 126. The packet names the displayed K-topology carrier only; it does not import host powersets, quotient point equality, choice over neighbourhood bases, or ambient real-line separation.

Definition 7389.1 (K-topology carrier)**.** *A* $\mathsf{KTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, D, L, O, W, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealOrderTopology}^{\uparrow}$ *source row, naming the order-neighbourhood surface inherited from the real-line order packet.*
2. *D is the distinguished closed-subset row, recording the visible closed set K as a finite membership and boundary ledger rather than as an ambient powerset.*
3. *L is the deleted-neighbourhood ledger. A neighbourhood request at a displayed point may inspect ordinary order intervals from R and, when the request meets the distinguished row D, the ledger records the deletion of the named K-part before the open row is exported.*
4. *O is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open handoff, exposing accepted open observations as Sierpinski-valued reads instead of host predicates on points.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing the finite point, interval, boundary, and observable-open reads admitted in the present packet.*
6. *H records componentwise* hsame *transport for the real-order source, closed-subset ledger, deleted-neighbourhood ledger, observable-open handoff, finite windows, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite neighbourhood request through R, D, L, O, W, H.*
8. *P is the* Pkg *provenance over* $\mathsf{KTopology}^{\uparrow}$, $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{KTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host subset equality, quotient point equality, arbitrary powersets, choice-selected bases, or a completed real-line topology beyond the displayed source rows.

Theorem 7389.2 (K-topology NameCert obligations)**.** *For every accepted* $\mathsf{KTopology}^{\uparrow}$ *carrier* $K = (R, D, L, O, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the real-order topology source, distinguished closed subset, deleted-neighbourhood ledger, Sierpinski observable-open handoff, finite StreamName windows, componentwise transport, displayed continuation replay, package provenance, and local naming. A consumer may read a K-topology open observation only after those rows have been exposed from the same finite packet.*

Proof. Project the carrier of Definition 7389.1. The projection exposes the source row R, the distinguished closed-subset row D, the deleted-neighbourhood ledger L, the observable-open handoff O, the finite-window row W, and the structural rows H, C, P, N.

Read neighbourhood requests through R, D, L. Ordinary order-neighbourhood data comes from $\mathsf{RealOrderTopology}^{\uparrow}$, while any request touching the distinguished closed row must display the deletion ledger before an open observation is accepted.

The observable-open handoff O and finite-window row W then route the accepted request through $\mathsf{SierpinskiSpace}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$. This makes the open read finite and observable, not a host predicate over all real points.

Finally transport through H, replay through C, and attach P, N. Since the carrier contains no powerset coordinate, quotient point equality, choice-selected basis, or ambient real-line separation oracle, the displayed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{KTopology}^{\uparrow}$ open-observation read factors through a $\mathsf{RealOrderTopology}^{\uparrow}$ source row, a distinguished closed-subset ledger, a deleted-neighbourhood ledger, a $\mathsf{SierpinskiSpace}^{\uparrow}$ observable-open handoff, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real-order-topology, topology, sierpinski-space, streamname, subspace-topology: $\mathsf{KTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a real-order source, a distinguished closed-subset row, deleted-neighbourhood routing, observable-open handoff, finite windows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KTopology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KTopology}^{\uparrow}$ packet is generated from a $\mathsf{RealOrderTopology}^{\uparrow}$ source row, a distinguished closed-subset ledger, deleted-neighbourhood rows, a $\mathsf{SierpinskiSpace}^{\uparrow}$ observable-open handoff, $\mathsf{StreamName}^{\uparrow}$ finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KTopology}^{\uparrow}}$, seed)

Scope closed: Definition 7389.1 and Definition 7389.2 seed the finite K-topology NameCert surface over real-order, deleted-neighbourhood, observable-open, finite-window, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host powersets, quotient point equality, choice-selected neighbourhood bases, ambient real-line separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deleted-neighbourhood exactness, observable-open handoff, finite-window replay, provenance exhaustion, and TasteGate nontriviality with field faithfulness for $R, D, L, O, W, H, C, P, N$.

Chapter 7390

A Concrete Naming Certificate for $\mathsf{ProofTraceCompactness}^{\uparrow}$

The $\mathsf{ProofTraceCompactness}^{\uparrow}$ packet realizes the finite proof-trace compact replay route as NameCert data. It sits after the finite proof-trace sieve of chapter 7167, the MetaCIC proof-search frontier of chapter 4284, the critical-path packet of chapter 765, the bounded endpoint checker of chapter 752, and the normalization frontier of chapter 892. The packet records compactness only as finite replay stability over displayed trace windows; it does not claim global normalization, confluence, subject reduction, quotient equality of proof traces, or hidden oracle discharge.

Definition 7390.1 (Proof-trace compactness carrier)**.** *A* $\mathsf{ProofTraceCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, W, E, R, O, H, Q, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ *source row, listing the candidate trace table, bounded endpoint requests, discharge sockets, and obstruction rows available to the compact replay.*
2. *W is the finite trace-window cover row, naming the displayed subwindows whose replay suffices for the present consumer read.*
3. *E is the bounded endpoint row, read through* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *and the MetaCIC normalization frontier.*
4. *R is the replay compactness row, recording that every accepted consumer read factors through one of the displayed trace windows before it reaches endpoint or discharge data.*
5. *O is the obstruction retention row, preserving missing endpoint, subject-reduction, closedness, and closure-preservation sockets as visible frontier data.*
6. *H records componentwise* hsame *transport for the sieve source, trace-window cover, endpoint row, compact replay row, obstruction row, replay, provenance, and local naming rows.*
7. *Q records displayed* Cont *replay from proof-search request through S, W, E, R, O, H.*
8. *P is the* Pkg *provenance over* $\mathsf{ProofTraceCompactness}^{\uparrow}$, $\mathsf{FiniteProofTraceSieve}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofTraceCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite proof-search tree, a global normalizer, unrestricted confluence, closed subject reduction, quotient equality of traces, host equality of terms, or oracle discharge.

Theorem 7390.2 (Proof-trace compactness NameCert obligations)**.** *For every accepted* $\mathsf{ProofTraceCompactness}^{\uparrow}$ *carrier* $C = (S, W, E, R, O, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProofTraceCompactness}^{\uparrow}}$ *surface consists exactly of sieve-source admission, finite trace-window cover exposure, bounded endpoint readback, compact replay routing, obstruction retention, componentwise transport, displayed continuation replay, package provenance, and local naming. A consumer may treat a proof-trace read as compact only by displaying the finite window through which that read is replayed.*

Proof. Project the carrier of Definition 7390.1. The rows $S, W, E, R, O, H, Q, P, N$ are the only rows exposed by an accepted packet.

Read the sieve source and trace-window cover first. The source row S lists the candidate traces and sockets, while W names the finite windows that may cover the present replay request.

The endpoint and replay rows E, R are downstream of that cover. A consumer reaches bounded endpoint data only after selecting a displayed trace window, and compactness means that this finite replay route is visible in the packet.

The obstruction row O remains part of the same read. Missing endpoint, subject-reduction, closedness, or closure-preservation data is retained as frontier information rather than converted into success.

Finally H, Q, P, N transport, replay, source, and name the same finite surface. Since the carrier contains no global normalizer, confluence oracle, subject-reduction theorem, quotient trace equality, host term equality, or hidden discharge coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 7390.3 (Proof-trace compact replay route)**.** *Every accepted* $\mathsf{ProofTraceCompactness}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{FiniteProofTraceSieve}^{\uparrow},\ \mathsf{MetaCICCriticalPath}^{\uparrow},\ \mathsf{BoundedNormalEqualityChecker}^{\uparrow},\ \mathsf{MetaCICNormalizationFrontier}^{\uparrow},\ W$$

The route exports a finite trace-window replay surface and an obstruction-retaining endpoint read. It does not export global normalization, unrestricted confluence, closed subject reduction, quotient equality of proof traces, host equality of MetaCIC terms, or hidden oracle discharge.

Proof. Begin with the proof-trace sieve source. The accepted read is confined to candidate traces and discharge sockets already displayed by $\mathsf{FiniteProofTraceSieve}^{\uparrow}$.

Next inspect the finite window cover W. The compact replay row R can be read only after the consumer identifies the displayed trace window whose replay carries the present request.

Endpoint data is then read through E and the bounded normal-equality checker. Any missing endpoint or discharge row is exported through O as an obstruction, not as a completed proof.

Transport through H, replay through Q, and attach P, N. Thus the consumer receives a finite compact replay route over the displayed trace windows, with all refused global endpoints still outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{ProofTraceCompactness}^{\uparrow}$ read displays a $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ source row, a finite trace-window cover, bounded endpoint readback, compact replay routing, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a proof-search consumer may treat the read as compact.

Independence witness: finite-proof-trace-sieve, metacic-proof-search-frontier, metaciccritical-path, bounded-normal-equality-checker, metacicnormalizationfrontier: $\mathsf{ProofTraceCompactness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite trace-window cover data, compact replay routing, endpoint readback, obstruction retention, transport, replay, provenance, and local naming as one packet.

Future TasteGate discharge for $\mathsf{ProofTraceCompactness}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ProofTraceCompactness}$ and $\mathsf{FieldFaithful\ ProofTraceCompactness}^{\uparrow}$ instances. The field-faithfulness witness must preserve $S, W, E, R, O, H, Q, P, N$ as distinct rows before any checked carrier-alignment theorem can serve as a public gate.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTraceCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTraceCompactness}^{\uparrow}$ packet is generated from a $\mathsf{FiniteProofTraceSieve}^{\uparrow}$ source row, finite trace-window cover rows, bounded endpoint readbacks, compact replay routing, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofTraceCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7390.1, Definition 7390.2, and Definition 7390.3 seed the finite proof-trace compact replay surface over displayed trace windows, endpoint rows, obstruction retention, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofTraceCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, unrestricted confluence, closed subject reduction, quotient trace equality, host term equality, hidden oracle discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite trace-window cover exactness, compact replay routing, obstruction retention, and TasteGate nontriviality with field faithfulness for $S, W, E, R, O, H, Q, P, N$.

Chapter 7391

A Concrete Naming Certificate for $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$ packet records the finite service that refines located rational interval cells into dyadic readback windows and regular-Cauchy request rows. It sits between the located-cell surface of chapter 394, the rational and dyadic interval rows of chapter 354 and chapter 291, and the regular-Cauchy consumer of chapter 301. The packet is a readback service, not a quotient real construction, a selected endpoint operator, a trichotomy principle, or an ambient completeness assumption.

Definition 7391.1 (Bishop located interval refinement carrier)**.** *A* $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, Q, D, S, G, A, H, C, P, N)$$

with the following rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, containing the displayed interval cell, locatedness witness, and finite request boundary.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint ledger read from the same displayed cell. It records rational lower and upper witnesses only as named rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger. It supplies dyadic subcells, midpoint rows, and positive dyadic radii used by finite readback.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule row. It lists the finite observation windows through which downstream names ask for interval refinement.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy handoff row. It records the finite modulus requests and tail windows accepted by the regular sequence consumer.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the real-name boundary used to certify that the handoff is a readback of the same located interval packet.*
7. *H records componentwise* hsame *transport for $L, Q, D, S, G, A, C, P, N$.*
8. *C records displayed* Cont *replay of the finite route*

$$L, Q \longmapsto D \longmapsto S, G \longmapsto A.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares only the displayed rows above. It contains no coordinate for quotient reals, selected endpoints, trichotomy, arbitrary choice of nested cells, or ambient completeness.

Theorem 7391.2 (Bishop located interval refinement NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$ *carrier*

$$R = (L, Q, D, S, G, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, rational interval ledger, dyadic refinement ledger, stream request schedule, regular-Cauchy handoff, real seal, transport, replay, provenance, and local naming rows;*
2. *endpoint readback, requiring each rational endpoint row in Q to be read through the located interval cell L rather than through a selected host endpoint;*
3. *dyadic refinement, requiring every finite subcell read to pass through the dyadic ledger D before it reaches a stream or regular-Cauchy consumer;*
4. *request compatibility, requiring the schedule row S and the regular-Cauchy row G to expose the same finite tolerance windows;*
5. *nonescape, requiring downstream Real, stream, dyadic, and regular-Cauchy consumers to read only $L, Q, D, S, G, A, H, C, P, N$.*

No BishopLocatedIntervalRefinement obligation is stored outside these rows.

Proof. Project the carrier of Definition 7391.1. The displayed tuple contains the located source cell, rational endpoint ledger, dyadic refinement ledger, stream schedule, regular-Cauchy handoff, real seal, and structural rows.

Endpoint readback uses L and Q. The rational endpoint rows are named observations of the located cell, so no host endpoint is selected and no trichotomy branch is imported.

Dyadic refinement is mediated by D. Each finite subcell and radius read passes through the dyadic ledger before S or G may inspect it.

Finally compare the request rows S and G through H and replay them through C. Provenance and local naming are restricted to P and N, so downstream consumers cannot extract quotient reals, selected endpoints, arbitrary nested-cell choice, or ambient completeness from the packet. □

Theorem 7391.3 (Bishop located interval refinement Cauchy route)**.** *Let*

$$R = (L, Q, D, S, G, A, H, C, P, N)$$

be an accepted $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$ *carrier. Every finite readback from a located interval cell to a regular-Cauchy request factors through*

$$L, Q \longmapsto D \longmapsto S, G \longmapsto A, H, C, P, N.$$

Thus the route converts located interval evidence into dyadic and regular-Cauchy finite obligations over $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *without importing quotient real equality, selected endpoints, trichotomy, or ambient completeness.*

Proof. Start with the located interval source row L and rational endpoint ledger Q. They display the cell to be refined and the rational boundary observations consumed by the finite route.

Apply the dyadic ledger row D. It refines the displayed rational cell into dyadic subcells and radius rows, giving the finite windows read by the stream schedule S.

Use the regular-Cauchy handoff row G together with the real seal A. The handoff exports only finite modulus requests and tail windows, and the seal records that these requests read the same displayed source rather than a quotient real.

The rows H, C, P, N transport, replay, package, and name exactly this chain. Since no other coordinate is present, the readback cannot select endpoints, invoke trichotomy, identify quotient reals, or assume ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7391.1, Definition 7391.2, and Definition 7391.3 define the finite located-interval refinement service and its dyadic and regular-Cauchy readback obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected endpoints, trichotomy, arbitrary nested-cell choice, ambient completeness, Lean verification, or standard bridge closure.

Upgrade path: Scoped closure requires a Lean-backed carrier and obligation theorem for BEDC.Derived.BishopLocatedIntervalRefinement

Constructive story: A $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$ packet is generated from a LocatedInterval source row, RationalInterval endpoint ledger, DyadicRatCore refinement ledger, StreamName request schedule, RegSeqRat handoff, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Chapter 7392

A Concrete Naming Certificate for $\mathsf{CauchySeparableCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySeparableCompletion}^{\uparrow}$ packet records the constructive completion route in which a countable dense boundary, finite dyadic tolerances, stream windows, and regular Cauchy readback are assembled before the terminal Real seal is visible. It consumes the separability witness of chapter 1809, the finite-window discipline of chapter 126, the regular rational readback of chapter 301, the completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The chapter gives the L10 Real and Completion route a single human-known NameCert packet rather than separate density and completion fragments.

Definition 7392.1 (Cauchy separable completion carrier)**.** *A* $\mathsf{CauchySeparableCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, T, W, R, C, E, H, K, P, N)$$

with the following displayed rows:

1. *D is the dense enumeration row, naming the countable dense source exposed by* $\mathsf{RealSeparability}^{\uparrow}$*.*
2. *T is the dyadic tolerance row, recording the finite precision requests used to compare dense approximants.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the stream fragments inspected for the displayed tolerance requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, repacking the inspected windows as a regular Cauchy handoff.*
5. *C is the regular Cauchy handoff row consumed by the* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary.*
6. *E is the completion seal, visible only after D, T, W, R, C have been displayed.*
7. *H is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the completion seal and not from a host equality of completed names.*
8. *K records componentwise* hsame *transport and displayed* Cont *replay for the dense enumeration, tolerance, window, regular readback, completion seal, and Real seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySeparableCompletion}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySeparableCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, T, W, R, C, E, H, K, P, N$. It has no coordinate for quotient streams, selected dense points, chosen limits, host equality, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.CauchySeparableCompletionUp`

Theorem 7392.2 (Cauchy separable completion NameCert obligations)**.** *For every accepted* CauchySeparableComp *carrier*

$$Q = (D, T, W, R, C, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySeparableCompletion}^{\uparrow}}$ *surface consists exactly of dense enumeration admission, dyadic tolerance accounting, stream-window inspection, regular rational readback, regular Cauchy handoff, completion-seal discipline, Real-seal routing, componentwise transport and replay, provenance, and local naming. Every completion-facing consumer must read the displayed route*

$$\mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it may export the completed real-name boundary.

Proof. Project the carrier of Definition 7392.1. The projection exposes the dense enumeration row, dyadic tolerance row, stream windows, regular rational readback, regular Cauchy handoff, completion seal, Real seal, and structural rows.

The route is finite at each consumer request. The dense row D supplies only the displayed enumeration boundary, while T supplies the finite tolerance budget used to inspect the requested approximants.

The stream and regular rows then determine the handoff surface. The window row W records the inspected fragments, and R repacks them as the regular Cauchy readback consumed by C.

Finally E and H are exposed only after that handoff, and K, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient streams, selected dense points, chosen limits, host equality, countable choice, open-cover compactness, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7392.3 (Cauchy separable completion dense regular handoff)**.** *Every accepted* CauchySeparableCompletio *packet hands a completion-facing consumer from the dense enumeration row D to the* $\mathsf{MetricCompletion}^{\uparrow}$ *seal only through the dyadic tolerance row T, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row R. The handoff supplies a displayed regular Cauchy boundary; it does not supply a selected dense point, a quotient stream, a chosen limit, or host equality of completed names.*

Proof. Begin with the NameCert obligations of Definition 7392.2. They order every completion-facing read through the dense enumeration, tolerance, stream-window, regular readback, and handoff rows before the completion seal is available.

The dense row is only an enumeration boundary. A consumer may request finite approximants from D, but those requests are interpreted through the dyadic budget T and the stream windows W.

The regular Cauchy handoff is therefore the row R together with its continuation into C. No selected dense point or quotient stream is present in the carrier, so the completion-facing consumer receives exactly the displayed regular route. □

Theorem 7392.4 (Cauchy separable completion no-choice boundary)**.** *No accepted* $\mathsf{CauchySeparableCompletion}^{\uparrow}$ *carrier exports Real completeness, open-cover compactness, quotient streams, selected dense points, host equality of completed names, or countable choice. Any Real-facing read must factor through the dense enumeration row, dyadic tolerance row, stream-window row, regular Cauchy handoff, completion seal, Real seal, transport and replay row, provenance row, and local NameCert row displayed in Definition 7392.1.*

Proof. Use Definition 7392.3. It forces every completion-facing read through D, T, W, R, C before the completion seal can be inspected.

The Real-facing seal adds no new analytic principle. It is the row H reached after E, with K transporting and replaying only the same finite dependency surface.

Thus the excluded principles are absent by carrier projection. The packet records separable completion data as finite NameCert rows and not as a choice principle, quotient construction, open-cover theorem, host equality, or ambient completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySeparableCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySeparableCompletion}^{\uparrow}$ packet is generated from a dense enumeration row, dyadic tolerance row, StreamName windows, RegSeqRat regular readback, regular Cauchy handoff, completion seal, Real seal, componentwise hsame transport and Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySeparableCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7392.1, Definition 7392.2, Definition 7392.3, and Definition 7392.4 seed the finite separable-completion route over RealSeparabilityUp, StreamNameUp, RegSeqRatUp, MetricCompletionUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySeparableCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient streams, selected dense points, host equality of completed names, countable choice, Lean theorem verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove dense enumeration admission, dyadic tolerance accounting, stream-window inspection, regular Cauchy handoff, completion-seal discipline, Real-seal routing, transport, replay, provenance, and ledger non-escape.

Chapter 7393

A Concrete Naming Certificate for $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$ packet records the finite regular-Cauchy squeeze route by which a middle rational name is trapped between lower and upper regular rational names before a terminal Real-facing seal is exposed. It sits over the dyadic arithmetic core of chapter 291, the finite observation rows of chapter 126, the regular rational sequence surface of chapter 301, and the completed real-name seal of chapter 38. The certificate is a displayed sandwich-limit route for Phase Real and Completion consumers; it does not assert quotient real equality, choose a host limit, use countable choice, or invoke ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7393.1 (Regular-Cauchy sandwich-limit carrier)**.** *A* $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, M, U, B, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{RegSeqRat}^{\uparrow}$ *row, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and monotone lower-side rational observations.*
2. *M is the middle* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose Real-facing read is to be certified by the sandwich route.*
3. *U is the upper* $\mathsf{RegSeqRat}^{\uparrow}$ *row, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and monotone upper-side rational observations.*
4. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *two-sided bound ledger recording, for each inspected tolerance cell, the lower–middle and middle–upper inequalities and the dyadic width budget between L and U.*
5. *S is the squeeze-to-*$\mathsf{Real}^{\uparrow}$ *seal row, exposing the terminal real-name boundary only after L, M, U and B have been read.*
6. *H records componentwise* hsame *transport for the three regular-Cauchy rows, the dyadic bound ledger, the seal row, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay for rereading the sandwich request through the same finite observation windows and dyadic tolerance cells.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwichLimit}^{\uparrow}}$ *row naming exactly this finite packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected representatives, a host least upper bound, countable choice, trichotomy, or ambient Real completeness.

Theorem 7393.2 (Regular-Cauchy sandwich-limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$ *carrier* $K = (L, M, U, B, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwichLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower, middle, and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *rows; exposure of the finite* $\mathsf{StreamName}^{\uparrow}$ *windows used by those rows; visibility of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *two-sided bound ledger; squeeze-to-*$\mathsf{Real}^{\uparrow}$ *seal exposure; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; and local naming. No obligation may replace these rows by quotient real equality, a selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7393.1. The only analytic rows available to a consumer are the three regular rational rows L, M, U, the dyadic bound ledger B, and the terminal seal S.

The obligation surface is rowwise. The lower and upper rows supply the bounding regular-Cauchy names, the middle row supplies the candidate read, and B supplies the two-sided dyadic inequalities at the inspected tolerance. The seal row S is therefore available only after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name precisely the same finite packet. Since the carrier contains no quotient equality, selected limit, countable-choice selector, trichotomy branch, or ambient completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7393.3 (Regular-Cauchy sandwich-limit squeeze route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of the middle row* M *in an accepted* $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$ *packet factors through the two bounding* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* L, U *and the dyadic tolerance ledger* B *before the seal row* S *is available. The route reads lower–middle and middle–upper comparison cells at the same displayed precision and then exports the terminal* $\mathsf{Real}^{\uparrow}$ *seal as finite BEDC data, not as a host-limit assertion.*

Proof. Start with the accepted packet $K = (L, M, U, B, S, H, C, P, N)$. A middle read is not independent carrier data: it is the row M inside the same packet that also contains lower and upper regular-Cauchy rows.

Next inspect the ledger B. For each visible tolerance cell, B records the lower–middle comparison, the middle–upper comparison, and the dyadic width budget between L and U. Thus a consumer cannot reach the middle Real-facing read without seeing both bounding rows and the same tolerance ledger.

Finally read S through H, C, P, N. Transport and replay keep the same sandwich route visible when the terminal seal is consumed, and provenance and local naming record that no selected host limit, quotient stream equality, countable choice, trichotomy, or ambient completeness principle has been added. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySandwichLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySandwichLimit}^{\uparrow}$ packet is generated from lower, middle, and upper RegSeqRat rows, finite StreamName windows, a DyadicRatCore two-sided bound ledger, a squeeze-to-Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySandwichLimit}^{\uparrow}}$, seed)

Scope closed: Definition 7393.1, Definition 7393.2, and Definition 7393.3 seed the finite sandwich-limit certificate over lower, middle, and upper regular-Cauchy rows, dyadic tolerance cells, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySandwichLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper regular-Cauchy exposure, middle-row readback, dyadic two-sided bound visibility, squeeze-to-Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySandwichLimitUp.

Chapter 7394

A Concrete Naming Certificate for $\mathsf{MiddleThirdCantorSet}^{\uparrow}$

Following Cantor and Bishop–Bridges constructive analysis, the $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ packet records the middle-third Cantor subset of the real line as finite ternary-stage NameCert data. It reads the binary prefix discipline of $\mathsf{CantorSpace}^{\uparrow}$ through chapter 1475, the compact finite-prefix surface of $\mathsf{CantorSpaceCompactness}^{\uparrow}$ through chapter 3798, the rational endpoint arithmetic of $\mathsf{Rat}^{\uparrow}$ through chapter 37, the finite observation schedule of $\mathsf{StreamName}^{\uparrow}$ through chapter 126, the regular rational readback of $\mathsf{RegSeqRat}^{\uparrow}$ through chapter 301, and the terminal real-name seal of $\mathsf{Real}^{\uparrow}$ through chapter 38. The carrier names only finite retained-third prefixes, finite deleted-gap ledgers, regular rational endpoints, stream windows, transport, replay, provenance, and local naming; it does not import host subsets, quotient real equality, selected infinite branches, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7394.1 (Middle-third Cantor set carrier)**.** *A* $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, T, G, Q, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row, recording the finite binary choices that are translated into retained left-third and right-third ternary stages.*
2. *T is the finite ternary-stage ledger, listing the retained closed intervals visible at the inspected depths.*
3. *G is the deleted-middle gap ledger, listing exactly the middle-third intervals removed before the displayed retained stages are read.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *endpoint row, giving rational endpoints for every retained interval and deleted gap named by T and G.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule that bounds which stage numbers, prefix positions, and endpoint rows a consumer may inspect.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the displayed nested endpoint approximations toward a real-name consumer.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite prefix, gap, endpoint, stream, and regular-rational rows have been displayed.*

8. *H stores componentwise* hsame *transport for the Cantor prefix row, ternary-stage ledger, deleted-gap ledger, rational endpoints, stream schedule, regular-rational readback, Real seal, replay route, provenance, and local naming.*
9. *K is the displayed* Cont *replay route from finite Cantor prefixes to retained ternary stages, from retained stages to endpoint rows, and from endpoint rows to the regular-rational and Real-facing handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{MiddleThirdCantorSet}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{CantorSpaceCompactness}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MiddleThirdCantorSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host subset of the real line, quotient equality of real names, an infinite branch selector, an open-cover compactness theorem, or ambient real completeness.

Theorem 7394.2 (Middle-third Cantor set NameCert obligations)**.** *For every accepted* $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ *carrier*

$$C = (B, T, G, Q, S, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MiddleThirdCantorSet}^{\uparrow}}$ *surface consists exactly of carrier admission, finite Cantor-prefix translation, retained ternary-stage visibility, deleted-middle gap exclusion, rational endpoint exactness, StreamName schedule discipline, regular-rational readback, Real seal handoff, componentwise transport, continuation replay, provenance, and finite-stage nonescape. Every obligation is read through*

$$B, T, G, Q, S, R, E, H, K, P, N$$

and through no additional host-level membership or compactness coordinate.

Proof. Carrier admission is projection of Definition 7394.1. The rows B, T, G expose the finite Cantor prefix, the retained ternary-stage ledger, and the deleted-middle ledger before any endpoint or Real-facing read is available.

Endpoint exactness is then forced by Q. Each retained interval and each deleted gap named by T or G receives rational endpoints in the same displayed packet, so the consumer cannot replace the finite ternary-stage route by a host subset of $\mathsf{Real}^{\uparrow}$.

The observation and readback obligations are S, R, E. The StreamName row S bounds the visible stages, the regular-rational row R records the endpoint approximations, and the Real row E is reached only after those finite rows have been consumed.

Finally H, K, P, N preserve, replay, source, and locally name the same route. Since no displayed row contains quotient real equality, selected infinite branches, open-cover compactness, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7394.3 (Middle-third Cantor set finite-stage nonescape)**.** *Every Real-facing read of an accepted* $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ *carrier factors through the finite route*

$$B, S \longmapsto T \longmapsto G \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, K, P, N.$$

The route exports a finite retained-stage and deleted-gap certificate over rational endpoint rows. It exports no host subset membership predicate, no quotient equality of real names, no selected infinite branch, no open-cover compactness principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with a Real-facing consumer. The only available stage request is the $\mathsf{StreamName}^{\uparrow}$ schedule S applied to the finite $\mathsf{CantorSpace}^{\uparrow}$ prefix row B, so the consumer sees finitely many binary choices rather than an infinite branch.

Read the ternary-stage rows next. The retained intervals in T and the deleted gaps in G are displayed together, and the rational endpoint row Q gives the finite endpoint data attached to those rows.

Only after that endpoint data is visible may the packet expose R and E. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R carries the displayed endpoint approximations, and the $\mathsf{Real}^{\uparrow}$ seal E is the terminal row of that same finite route.

The structural rows H, K, P, N cannot add omitted host principles; they transport, replay, source, and name the route already displayed. Thus the Real-facing handoff remains a finite-stage middle-third Cantor certificate. □

Falsifiable prediction: Every accepted $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ Real-facing read exposes finite Cantor prefixes, retained ternary stages, deleted-middle gaps, rational endpoints, StreamName schedules, RegSeqRat readbacks, Real seals, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any membership-shaped output is available.

Independence witness: cantorspace, cantor-space-compactness, rat, streamname, regseqrat, real: $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Cantor-prefix translation, retained ternary-stage ledgers, deleted-middle gap rows, rational endpoint exactness, finite observation schedules, regular-rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MiddleThirdCantorSet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MiddleThirdCantorSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7394.1, Definition 7394.2, and Definition 7394.3 seed the finite ternary-stage route over $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{CantorSpaceCompactness}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MiddleThirdCantorSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim Lean verification, bridge checking, host subset membership, quotient equality of real names, selected infinite branches, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix translation, retained-stage visibility, deleted-gap exclusion, rational endpoint exactness, StreamName schedule discipline, regular-rational readback, Real seal handoff, and nonescape under BEDC.Derived.MiddleThirdCantorSetUp.

Constructive story: A $\mathsf{MiddleThirdCantorSet}^{\uparrow}$ packet is generated from finite Cantor-prefix choices, retained ternary-stage rows, deleted-middle gap rows, rational endpoints, StreamName schedules, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Chapter 7395

A Concrete Naming Certificate for $\mathsf{FrechetFilterCompletion}^{\uparrow}$

Following Frechet, Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{FrechetFilterCompletion}^{\uparrow}$ packet records the L10 completion frontier where cofinite-tail filter data is read as a completion-facing Cauchy filter without selecting a completed point. It sits after the cofinite-tail filter surface of chapter 1564, the completion consumer of chapter 1569, the sequential route of chapter 3958, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed Frechet-tail, Cauchy-filter, sequential-completion, window, readback, seal, transport, replay, provenance, and local certificate rows; it does not assert quotient filter equality, a selected limit, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7395.1 (Frechet filter completion carrier)**.** *A* $\mathsf{FrechetFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, Q, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite-tail row whose threshold and finite-complement data must be consumed first.*
2. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-facing row that receives the displayed filter route.*
3. *S is the* $\mathsf{MetricSequentialCompletion}^{\uparrow}$ *handoff row by which sequential completion consumers read the same cofinite-tail route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for tail and completion requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, Q, S, W, R have been replayed.*
7. *H records componentwise* hsame *transport for the Frechet filter, Cauchy-filter completion, sequential-completion handoff, windows, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through the cofinite-tail filter, Cauchy-filter completion row, sequential handoff, finite windows, regular readback, and real seal.*
9. *P records* Pkg *provenance over* $\mathsf{FrechetFilterCompletion}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{MetricSequentialCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FrechetFilterCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $F, Q, S, W, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected completion points, countable-choice tail selection, host equality of completed objects, or ambient real completeness.

Theorem 7395.2 (Frechet filter completion NameCert obligations)**.** *For every accepted* $\mathsf{FrechetFilterCompletion}^{\uparrow}$ *carrier*

$$G = (F, Q, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Frechet filter row, Cauchy-filter completion row, metric sequential-completion handoff, StreamName window schedule, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming row. Every completion-facing obligation is read through those rows and no obligation is stored in quotient filter equality, selected limits, countable choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 7395.1. The tuple displays the Frechet filter source, the Cauchy-filter completion consumer, the sequential-completion handoff, the window schedule, the regular rational readback, the terminal real seal, and the structural rows.

The filter side is exhausted by F and Q. The Frechet row exposes only cofinite-tail threshold and finite-complement data, and the Cauchy-filter completion row receives that displayed route without adding a quotient filter or selected completion point.

The sequential and real-name side is exhausted by S, W, R, E. The sequential handoff reads the same cofinite-tail route, the finite windows in W schedule the requested observations, the regular rational row R is read from those windows, and the real seal E appears only after the readback row has been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is present, the excluded quotient, selector, choice, host-equality, and ambient-completeness data cannot become NameCert obligations. □

Theorem 7395.3 (Frechet filter completion Cauchy handoff)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{MetricSequentialCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetFilterCompletion}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{MetricSequentialCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The route carries a cofinite-tail filter into the completion frontier as displayed data only. It exports no quotient filter, selected limit, countable-choice tail selector, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7395.2. It fixes the public completion surface at the rows $F, Q, S, W, R, E, H, C, P, N$.

Read the Frechet and Cauchy-filter rows first. The cofinite-tail row F supplies the finite threshold and tail-neighborhood information, and the completion row Q receives exactly that displayed filter route.

Next read the sequential-completion and real-name rows. The handoff row S presents the same route to sequential completion consumers, W schedules the finite observations, R supplies the regular rational readback, and E seals the route for real-facing consumers.

The structural rows keep the handoff confined to the displayed packet. Therefore a consumer that reaches the completion or real boundary without the stated route is not reading the accepted $\mathsf{FrechetFilterCompletion}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{FrechetFilterCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the $\mathsf{FrechetFilter}^{\uparrow}$ cofinite-tail row, the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ completion row, the $\mathsf{MetricSequentialCompletion}^{\uparrow}$ handoff, the $\mathsf{StreamName}^{\uparrow}$ finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: frechetfilter, cauchyfiltercompletion, metric-sequential-completion, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{FrechetFilterCompletion}^{\uparrow}$ because this packet binds the cofinite-tail filter source, Cauchy-filter completion consumer, sequential-completion handoff, finite window schedule, regular rational readback, real seal, transport, replay, provenance, and local naming as one L10 completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetFilterCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetFilterCompletion}^{\uparrow}$ packet is generated from a FrechetFilterUp cofinite-tail row, CauchyFilterCompletionUp completion row, MetricSequentialCompletionUp handoff, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7395.1, Definition 7395.2, and Definition 7395.3 seed the L10 Frechet filter completion route over FrechetFilterUp, CauchyFilterCompletionUp, MetricSequentialCompletionUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetFilterCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable-choice tail selectors, host equality of completed points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove Frechet-tail admission, Cauchy-filter completion exposure, sequential-completion handoff, window scheduling, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.FrechetFilterCompletionUp.

Chapter 7396

A Concrete Naming Certificate for $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$

Following Dedekind and Bishop–Bridges constructive analysis, the $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$ packet records the finite boundary data through which located cut-style order-completion requests may be read before any ambient completeness theorem for $\mathsf{Real}^{\uparrow}$ is available. It sits after the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dense Real route of chapter 1809. The packet is a constructive boundary object for order-completion data, not a Dedekind-cut quotient completion or a proof that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 7396.1 (Real order-completion boundary carrier)**.** *A* $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, D, S, R, E, H, C, P, N)$$

with the following rows:

1. *L is the located lower rational-cut row. It records finite rational lower witnesses and located comparison requests.*
2. *U is the located upper rational-cut row. It records finite rational upper witnesses and the displayed separation boundary from L.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket-window ledger. It stores dyadic lower and upper brackets, positive dyadic widths, and the finite refinement cells used by order reads.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It lists the finite requests through which a consumer asks for bracket refinement.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational approximants obtained from the displayed brackets and stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after L, U, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for $L, U, D, S, R, E, C, P, N$.*
8. *C records displayed* Cont *replay of the finite route*

$$L, U \longmapsto D \longmapsto S, R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealOrderCompletionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed located cut rows, dyadic bracket windows, finite observations, regular readback rows, terminal Real seals, transport, replay, provenance, and local names. It has no coordinate for Dedekind-cut completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, quotient real equality, or choice-selected suprema.*

Lean4 ✓ `BEDC.Derived.RealOrderCompletionBoundaryUp.RealOrderCompletionBoundaryTasteGate single carrier alignment`

Theorem 7396.2 (Real order-completion boundary obligations)**.** *For every accepted* $\mathsf{RealOrderCompletionBoundary}$ *carrier*

$$B = (L, U, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealOrderCompletionBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located lower row, located upper row, dyadic bracket ledger, finite stream observations, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *located cut exposure, requiring every order request to read both L and U before it may inspect a bracket window;*
3. *dyadic bracket refinement, requiring every finite order window to pass through D before it reaches* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*;*
4. *readback compatibility, requiring S and R to expose the same finite tolerance windows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is available;*
5. *nonescape, requiring downstream Real, separability, stream, regular-sequence, and dyadic consumers to read only $L, U, D, S, R, E, H, C, P, N$.*

No $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7396.1. The displayed tuple contains the located lower and upper cut rows, dyadic bracket ledger, finite stream observations, regular rational readback, terminal Real seal, and structural rows.

The located cut exposure obligation uses L and U together. A consumer cannot turn one side into a selected supremum or a trichotomy branch because the packet only displays finite lower and upper witnesses.

The dyadic bracket obligation uses D before any stream or regular-rational read. Thus every finite order-completion request is mediated by the same dyadic window route that already governs chapter 291, chapter 126, and chapter 301.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for Dedekind-cut completeness, ambient Real completeness, trichotomy, quotient real equality, or choice-selected suprema, the obligation surface is exactly the displayed boundary. □

Theorem 7396.3 (Real order-completion located supremum window)**.** *Every located supremum read admitted by an accepted* $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$ *carrier factors through the finite window*

$$L, U \longmapsto D \longmapsto S, R \longmapsto E, H, C, P, N.$$

The read may use located rational-cut rows, dyadic bracket windows, finite $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. It may not infer Dedekind-cut completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, quotient real equality, or a choice-selected supremum.*

Proof. Start with the located rows L and U. They expose the lower and upper rational-cut evidence as finite rows rather than as a completed cut quotient.

Read the dyadic bracket ledger D next. Its finite brackets and positive widths are the only windows through which the order-completion request can be refined.

Pass from D to S and R. The finite stream observations determine which regular rational approximants are read back, and the terminal Real seal E is visible only after those rows have been displayed.

The support rows H, C, P, N preserve, replay, source, and name this route. Hence the located supremum window is a boundary certificate over finite BEDC rows, not a proof of Dedekind-cut completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient real equality, or choice-selected suprema. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOrderCompletionBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealOrderCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7396.1, Definition 7396.2, and Definition 7396.3 seed the finite located order-completion boundary over rational-cut rows, dyadic bracket windows, stream observations, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealOrderCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient real equality, choice-selected suprema, Lean verification, or standard bridge closure.

Upgrade path: Scoped closure requires a Lean-backed carrier and obligation theorem for BEDC.Derived.RealOrderCompletionBoundaryUp

Constructive story: A $\mathsf{RealOrderCompletionBoundary}^{\uparrow}$ packet is generated from located lower and upper rational-cut rows, dyadic bracket windows, finite StreamName observations, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7397

A Concrete Naming Certificate for $\mathsf{RapidlyCauchySequence}^{\uparrow}$

Following Bishop and Bridges on constructive real names with explicit rates, the $\mathsf{RapidlyCauchySequence}^{\uparrow}$ packet records the fixed dyadic-rate sibling of $\mathsf{FastCauchySequence}^{\uparrow}$ as finite BEDC NameCert data. It sits on the Real and Completion route through the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names the special schedule 2^{-n} directly; it is not a quotient stream, a chosen limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7397.1 (Rapidly Cauchy sequence carrier)**.** *A* $\mathsf{RapidlyCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the sequence is inspected.*
2. *D is the fixed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row for the dyadic tolerance schedule $n \mapsto 2^{-n}$, including the finite radius and comparison rows read at each displayed window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the same stream windows and dyadic tolerance rows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R have been displayed.*
5. *H records componentwise* hsame *transport for the stream schedule, dyadic schedule, regular readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from every finite precision request through S, D, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RapidlyCauchySequence}^{\uparrow}$, $\mathsf{FastCauchySequence}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RapidlyCauchySequence}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary modulus, quotient stream equality, selected limits, countable-choice schedules, host real equality, or a detached completeness principle.

Theorem 7397.2 (Rapidly Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{RapidlyCauchySequence}^{\uparrow}$ *carrier $K = (S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RapidlyCauchySequence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed stream schedule, fixed dyadic-rate ledger, regular rational readback, Real seal, transport, replay, provenance, and local name;*
2. *fixed-rate discipline, requiring every precision request to read the dyadic schedule row D for the tolerance 2^{-n} before R or E may be consumed;*
3. *StreamName-to-RegSeqRat readback, requiring S and D to determine the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R without an arbitrary modulus parameter;*
4. *Real-seal terminality, requiring E to remain downstream of S, D, R and to export no completed host real;*
5. *ledger nonescape, requiring H, C, P, N to preserve only the displayed packet rows and no quotient stream, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 7397.1. The packet displays exactly the stream windows, fixed dyadic-rate ledger, regular readback, Real seal, transport, replay, provenance, and local name.

For fixed-rate discipline, read a precision request. The only tolerance row in the packet is D, so the request must use the dyadic schedule 2^{-n} before the regular readback R or seal row E can be reached.

The readback obligation follows by the same route. The stream windows S and the dyadic ledger D are the only sources for R, and the structural rows H, C, P, N preserve that route componentwise.

Finally, the excluded coordinates are absent from the carrier. Since K has no arbitrary modulus row, quotient-stream coordinate, choice schedule, selected limit, host equality, or completeness witness, no local NameCert obligation can read one. □

Theorem 7397.3 (Rapidly Cauchy dyadic schedule nonescape)**.** *Every consumer of an accepted* $\mathsf{RapidlyCauchySequence}^{\uparrow}$ *carrier $K = (S, D, R, E, H, C, P, N)$ must read the same displayed route*

$$S \longrightarrow D \longrightarrow R \longrightarrow E$$

with structural support from H, C, P, N. Hence the consumer cannot replace the fixed dyadic schedule by quotient stream equality, a selected limit, a countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7397.1. A downstream read sees only the finite stream schedule S, the dyadic tolerance ledger D, the regular readback R, the seal row E, and the structural rows H, C, P, N.

Apply the obligation surface of Definition 7397.2. Its fixed-rate and readback clauses require every precision request to pass through D before R or E can be consumed.

The nonescape conclusion is then immediate by absence of coordinates. None of the listed forbidden principles is a row of K, and the transport, replay, provenance, and local name rows preserve only the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{RapidlyCauchySequence}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read without the displayed $\mathsf{StreamName}^{\uparrow}$ schedule and fixed $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic-rate row refutes the carrier surface.

Independence witness: fastcauchysequence, fastcauchymodulus, streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole eight-row packet $K = (S, D, R, E, H, C, P, N)$, because this chapter binds the fixed 2^{-n} schedule, finite stream windows, regular readback, seal, transport, replay, provenance, and local naming as one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RapidlyCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RapidlyCauchySequence}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ schedule, a fixed $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic-rate ledger for 2^{-n}, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RapidlyCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7397.1, Definition 7397.2, and Definition 7397.3 seed the fixed dyadic-rate NameCert surface for rapidly Cauchy sequence packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RapidlyCauchySequence}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.RapidlyCauchySequenceUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, arbitrary moduli, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fixed-rate discipline, StreamName-to-RegSeqRat readback, Real-seal terminality, transport, replay, provenance, and local naming for the same displayed packet.

Lean4 (statement) `BEDC.Derived.RapidlyCauchySequenceUp`

Chapter 7398

A Concrete Naming Certificate for $\mathsf{BishopRealRegularCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealRegularCompletion}^{\uparrow}$ packet records the regular completion route for rational names as finite BEDC data. It reads dyadic tolerance arithmetic from chapter 291, finite observation windows from chapter 126, regular rational readback from chapter 301, effective tail schedules from chapter 415, and the terminal real-name seal from chapter 38. The packet names a finite means of completion; it does not identify quotient streams, select limits, use countable choice, assert host real equality, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7398.1 (Bishop real regular completion carrier)**.** *A* $\mathsf{BishopRealRegularCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic cells used to compare rational observations.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface for the inspected rational name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from those windows.*
4. *M is the* $\mathsf{CauchyRate}^{\uparrow}$ *effective tail schedule that bounds all admitted completion requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after D, S, R, M have been displayed.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, rate, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from tolerance cells and stream windows to regular readback, effective tail schedule, and terminal seal.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRealRegularCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealRegularCompletion}^{\uparrow}}$ *row naming exactly this finite completion packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRealRegularCompletionUp.BishopRealRegularCompletionTasteGate single carrier alignment`

Theorem 7398.2 (Bishop real regular completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealRegularCompletion}^{\uparrow}$ *carrier*

$$B = (D, S, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealRegularCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance visibility, finite stream-window admission, regular rational readback, effective tail scheduling, terminal real-name sealing, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing read must pass through*

$$D, S, R, M, E, H, C, P, N$$

and through no quotient stream equality, selected limit, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7398.1. The rows D, S, R, M, E expose the dyadic tolerance ledger, finite stream windows, regular rational readback, effective tail schedule, and terminal real seal before any completion-facing consumer is available.

Read the analytic route in order. The $\mathsf{StreamName}^{\uparrow}$ windows are interpreted through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells, the $\mathsf{RegSeqRat}^{\uparrow}$ row provides the regular rational readback, and the $\mathsf{CauchyRate}^{\uparrow}$ row bounds admitted tail requests before E is exposed.

The structural rows H, C, P, N preserve, replay, source, and locally name the same finite route. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness, the listed rows exhaust the NameCert obligation surface. □

Theorem 7398.3 (Bishop real regular completion finite-means route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopRealRegularCompletion}^{\uparrow}$ *packet receives a finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyRate}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

implemented by the displayed rows $D, S, R, M, E, H, C, P, N$*. The route is a finite regular-completion certificate, not a quotient of streams, a selected limit, a countable-choice construction, a host real equality theorem, or a proof that all* $\mathsf{Real}^{\uparrow}$ *names are complete.*

Proof. Start with the displayed rows D and S. They determine which dyadic tolerances and finite observation windows are visible to the consumer.

Next read R and M. The regular rational row R is downstream of the displayed windows, and the effective rate row M bounds the admitted tail requests that may ask for a completion-facing seal.

Finally read E through H, C, P, N. Transport and replay preserve the same row order, while provenance and local naming record that the packet contains no additional quotient, selector, choice, host-equality, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealRegularCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealRegularCompletion}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, CauchyRate tail schedules, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{BishopRealRegularCompletion}\uparrow}$, seed)

Scope closed: Definition 7398.1, Definition 7398.2, and Definition 7398.3 seed the finite regular-completion route over dyadic tolerance rows, stream windows, regular rational readback, effective tail schedules, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{BishopRealRegularCompletion}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.BishopRealRegularCompletionUp.BishopRealRegularCompletionTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: The chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance visibility, finite window admission, regular rational readback, effective tail scheduling, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealRegularCompletionUp.

Chapter 7399

A Concrete Naming Certificate for $\mathsf{FiniteObservationSheaf}^{\uparrow}$

Following the sheaf-theoretic locality tradition and the auditable-generativity reading of chapter 9462, the $\mathsf{FiniteObservationSheaf}^{\uparrow}$ packet records compatible finite observation windows as a NameCert object. It reads finite observation prefixes from chapter 7253, stream windows from chapter 126, indexed restriction rows from chapter 150, and finite overlap gluing from chapter 151. The packet is an auditable finite sheaf of observations over BHist windows; it does not assert an external site of all observations, host sheaf equality, unrestricted gluing, quotient sections, or a black-box generativity principle.

Definition 7399.1 (Finite observation sheaf carrier)**.** *A* $\mathsf{FiniteObservationSheaf}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, W, A, R, G, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the inspected observation windows.*
2. *W is the* $\mathsf{FinitePrefixStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ *window row for the finite observations assigned to each index.*
3. *A is the assignment row sending each displayed window to its local observation section.*
4. *R is the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction row for every displayed inclusion of finite windows.*
5. *G is the* $\mathsf{Sheaf}^{\uparrow}$ *finite-overlap gluing row for compatible displayed sections.*
6. *H records componentwise* hsame *transport of indices, windows, assignments, restrictions, gluing rows, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay from window inclusion to restriction and from pairwise compatibility to finite overlap gluing.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationSheaf}^{\uparrow}$, $\mathsf{FinitePrefixStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$ *row naming exactly this finite observation-sheaf packet.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host observation topos, equality of external sheaves, quotient sections, unrestricted covers, or opaque generative output.

Theorem 7399.2 (Finite observation sheaf NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationSheaf}^{\uparrow}$ *carrier*

$$O = (I, W, A, R, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$ *surface consists exactly of finite index admission, finite observation-window admission, local section assignment, presheaf restriction, finite-overlap compatibility, finite gluing, componentwise transport, displayed replay, provenance, and local naming. Every auditable-generativity read of the packet must pass through*

$$I, W, A, R, G, H, C, P, N$$

and through no external site of all observations, host sheaf equality, quotient section, unrestricted cover, or opaque generator.

Proof. Project the carrier of Definition 7399.1. The rows I, W, A expose the finite observation indices, finite windows, and local sections before any restriction or gluing row is available.

Read R and G next. The presheaf restriction row handles displayed window inclusions, while the sheaf row glues only compatible finite overlaps already named by the packet.

The structural rows H, C, P, N transport, replay, source, and locally name the same finite observation surface. Since the carrier has no coordinate for an external observation site, host sheaf equality, quotient sections, unrestricted covers, or an opaque generator, the listed rows exhaust the NameCert obligations. □

Theorem 7399.3 (Finite observation sheaf auditable gluing)**.** *Every generativity-facing read of an accepted* $\mathsf{FiniteObservationSheaf}^{\uparrow}$ *packet is a compatible finite gluing of displayed* BHist *window sections. The read factors through* $\mathsf{FinitePrefixStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{Sheaf}^{\uparrow}$ *rows before the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$ *row is consumed, and it exports auditable finite overlap data rather than opaque generated content.*

Proof. Start with the finite window rows I, W, A. They determine the observation windows and the local sections available to the consumer.

Apply the restriction row R. It compares local sections only along displayed inclusions of finite windows, so compatibility is a finite presheaf condition over the carried observations.

Finally apply G and then H, C, P, N. The gluing row is available only for those compatible finite overlaps, and the structural rows preserve the same route. Thus the generativity-facing read is auditable as finite window data rather than as a black-box output. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationSheaf}^{\uparrow}$ packet is generated from finite observation indices, finite StreamName and FinitePrefixStream windows, local section assignments, PreSheaf restriction rows, Sheaf finite-overlap gluing rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$, seed)

Scope closed: Definition 7399.1, Definition 7399.2, and Definition 7399.3 seed the finite observation-sheaf route over finite indices, observation windows, local section assignments, presheaf restrictions, finite gluing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteObservationSheaf}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim Lean verification, bridge checking, an external site of all observations, host sheaf equality, quotient sections, unrestricted covers, or opaque generative output.

Upgrade path: Obligation closure requires checked carrier admission, finite observation-window admission, assignment rows, restriction rows, finite-overlap gluing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteObservationSheafUp.

Chapter 7400

A Concrete Naming Certificate for $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ packet records the finite route by which dyadic tolerance requests are compared with Archimedean rational bounds before regular Cauchy and Real-facing consumers read a completion seal. It sits between the dyadic tolerance surface of chapter 291, the rational bound carrier of chapter 37, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The bridge is a finite NameCert route from dyadic radii to rational Archimedean bounds; it is not host real completeness, quotient equality of real names, trichotomy, or a selected limit.

Definition 7400.1 (Dyadic Archimedean bridge carrier)**.** *A* $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (N, Q, D, S, R, E, H, C, P, M)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *scale row naming the finite denominator exponent, mesh depth, and comparison stage used by the tolerance request.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *Archimedean-bound row, recording the rational upper bound that dominates the displayed dyadic tolerance at scale N.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic numerator, exponent, positivity witness, and radius ledger consumed by the comparison.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the completion request before any regular rational readback is exposed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and the rational bound row.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic tolerance, Archimedean rational bound, finite windows, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the scale, rational bound, dyadic tolerance, windows, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion tolerance request through $N, Q, D, S, R, E, H, P, M$.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *M is the local* $\mathsf{NameCert}_{\mathsf{DyadicArchimedeanBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, Q, D, S, R, E, H, C, P, M$. The carrier has no coordinate for host real completeness, quotient equality of real names, trichotomy, countable choice, selected limits, or a hidden Archimedean theorem outside the displayed rational-bound row.

Theorem 7400.2 (Dyadic Archimedean bridge NameCert obligations)**.** *For every accepted* DyadicArchimedeanBrid *carrier*

$$K = (N, Q, D, S, R, E, H, C, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicArchimedeanBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Nat}^{\uparrow}$ *scale row,* $\mathsf{Rat}^{\uparrow}$ *Archimedean-bound row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; rational-bound exactness requiring Q to dominate the dyadic tolerance displayed by D at the scale named by N; window locality requiring S and R to be read before E; and ledger non-escape excluding host completeness, quotient equality, trichotomy, countable choice, selected limits, and hidden Archimedean witnesses.*

Proof. Carrier admission is projection from Definition 7400.1. The visible packet contains exactly the scale row N, rational bound row Q, dyadic tolerance row D, finite windows S, regular readback R, Real seal E, and structural rows H, C, P, M.

The rational-bound obligation is local to N, Q, D. The scale row names the finite comparison stage, the dyadic row supplies the displayed tolerance, and the rational row supplies the Archimedean upper bound for that tolerance.

The completion-facing readback is local to S, R, E. A consumer opens finite StreamName windows, reads them through RegSeqRat, and reaches the terminal Real seal only after the rational and dyadic comparison rows have been displayed.

The remaining rows transport, replay, source, and name the same route. Since no excluded object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7400.3 (Dyadic Archimedean bridge Real tolerance route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ *packet factors a dyadic tolerance request through*

$$\mathsf{Nat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before the terminal seal is read. The route converts the displayed dyadic radius into a rational Archimedean bound at the finite scale N, opens the permitted StreamName windows, performs the RegSeqRat readback, and only then exposes the Real seal. It exports no host real completeness, quotient equality of real names, trichotomy, countable-choice schedule, selected limit, or hidden Archimedean witness.

Proof. Start with the accepted carrier $K = (N, Q, D, S, R, E, H, C, P, M)$. The dyadic tolerance request is not a free analytic assertion; it is the displayed row D read together with the scale row N.

Read the rational bound row Q. It is the only Archimedean comparison coordinate in the packet, so the dyadic tolerance can reach a rational upper bound only through N, Q, D.

Next open the completion-facing rows S, R, E. The finite StreamName windows determine the observations, RegSeqRat performs the regular rational readback, and the terminal Real seal is exposed only after those rows and the dyadic-to-rational comparison have been displayed.

Finally replay the route through H, C, P, M. Transport and continuation preserve the same finite packet, while provenance and local naming record that the Real tolerance route does not import completeness, quotient equality, trichotomy, choice, selected limits, or hidden Archimedean data. □

Falsifiable prediction: Every $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ read factors through a NatUp scale row, RatUp Archimedean-bound row, DyadicRatCoreUp tolerance row, finite StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: nat, rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite scale, rational Archimedean bound, dyadic tolerance, observation windows, regular readback, Real seal, transport, replay, provenance, and local NameCert row as one bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicArchimedeanBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicArchimedeanBridge}^{\uparrow}$ packet is generated from a NatUp scale row, a RatUp Archimedean-bound row, a DyadicRatCoreUp tolerance row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicArchimedeanBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7400.1, Definition 7400.2, and Definition 7400.3 seed the finite dyadic-to-Archimedean rational tolerance route from displayed scale and dyadic rows through rational bounds, finite windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicArchimedeanBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real completeness, quotient equality of real names, trichotomy, countable choice, selected limits, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, rational Archimedean-bound exactness, finite-window locality, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicArchimedeanBridgeUp.

Chapter 7401

A Concrete Naming Certificate for $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$

Following Bishop–Bridges compact-uniform continuity and Kelley-style function-space usage, the $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ packet names a finite BEDC carrier for compact-source continuous maps under uniform finite observation. It sits after the compact-uniform-continuity packet of chapter 388, the compact metric source rows of chapter 129, the continuous-map graph rows of chapter 130, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The packet is a finite function-space carrier consumed by compact-uniform continuity and Arzela–Ascoli style rows; it does not import host function equality, open-cover compactness, selected subsequences, or ambient Real completeness.

Definition 7401.1 (Compact-uniform function-space carrier)**.** *A* $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{F} = (K, M, U, W, R, D, E, H, C, P, N)$$

with the following displayed coordinates:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite* $\mathsf{ProbeBundle}$ *net and compact-metric distance observations available to function-space reads.*
2. *M is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger of displayed maps over the compact source.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *handoff row that turns pointwise map data into compact-source uniform windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for source probes, map indices, tolerance requests, and target observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational precision histories attached to W.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for finite target-distance and modulus comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by sampled target-distance rows after the window and readback coordinates have been displayed.*
8. *H records componentwise* hsame *transport for $K, M, U, W, R, D, E, C, P, N$.*

9. *C records displayed* Cont *replay from a function-space request through compact finite-net inspection, continuous-map graph sampling, compact-uniform handoff, stream scheduling, rational readback, dyadic tolerance accounting, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFunctionSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, U, W, R, D, E, H, C, P, N$. There is no coordinate for host equality of functions, open-cover compactness, a selected subsequence, a quotient map space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7401.2 (Compact-uniform function-space NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ *carrier*

$$\mathcal{F} = (K, M, U, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformFunctionSpace}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, displayed continuous-map admission through M, compact-uniform handoff through U, finite scheduling through W, regular-rational readback through R, dyadic tolerance accounting through D, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7401.1. Its coordinates list exactly the compact source, displayed maps, compact-uniform handoff, finite windows, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows.

The compact-source and map clauses are read through K and M. The source side is a finite compact-metric observation surface, while the map side is a displayed continuous-map ledger; neither clause exposes host equality of functions or a topology on all maps.

The uniform and numerical clauses are then read through U, W, R, D, E. The compact-uniform row orders the source windows, the stream and regular-rational rows schedule and read them back, the dyadic row accounts for finite tolerances, and the Real row seals sampled target distances only after those finite rows have appeared.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for open-cover compactness, selected subsequences, quotient map equality, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7401.3 (Compact-uniform function-space finite-net handoff)**.** *Every compact-uniform or Arzela–Ascoli style consumer of an accepted* $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ *packet must read the displayed finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{CompactUniformContinuity}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCo}$$

The handoff supplies compact finite-net observations, displayed map rows, uniform source-window control, finite precision schedules, rational readback, dyadic tolerance accounting, and terminal Real seals. It supplies no host function equality, open-cover compactness, selected subsequence, quotient function space, or ambient Real-completeness datum.

Proof. Start with an accepted carrier $\mathcal{F} = (K, M, U, W, R, D, E, H, C, P, N)$. The compact source is exposed only by K, and the maps available to the consumer are the displayed rows in M.

Next read the compact-uniform handoff U. It routes every tolerance request through the compact finite-net surface before any target observation can be sealed.

The schedule and numerical readback are then forced by W, R, D, E. The stream row names the finite windows, the regular rational row reads the precision histories, the dyadic row records the tolerance ledger, and the Real row seals only the sampled target-distance data produced by those finite rows.

The structural rows H, C, P, N preserve exactly that route under transport and replay. Because no further coordinate is present, a downstream compact-uniform or Arzela–Ascoli style packet cannot replace the finite-net handoff by host function equality, open-cover compactness, selected subsequences, quotient map equality, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ read factors through one $\mathsf{CompactMetric}^{\uparrow}$ finite-net source row, one finite $\mathsf{ContinuousMap}^{\uparrow}$ ledger, one $\mathsf{CompactUniformContinuity}^{\uparrow}$ handoff row, one $\mathsf{StreamName}^{\uparrow}$ schedule, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactuniformcontinuity, compactmetric, continuousmap, streamname, regseqrat, real, dyadicratcore: $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net source data, displayed continuous-map rows, compact-uniform handoff, finite schedules, rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local NameCert rows as one function-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFunctionSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFunctionSpace}^{\uparrow}$ packet is generated from a CompactMetricUp finite-net source, ContinuousMapUp displayed map rows, a CompactUniformContinuityUp handoff, StreamNameUp windows, RegSeqRatUp readback, DyadicRatCoreUp tolerance accounting, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFunctionSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7401.1, Definition 7401.2, and Definition 7401.3 seed the compact-uniform function-space packet over finite-net, map, uniform-handoff, window, readback, dyadic, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFunctionSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, open-cover compactness, selected subsequences, quotient function spaces, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source coverage, displayed-map admission, compact-uniform handoff, window and readback exactness, dyadic tolerance exactness, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 7402

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ packet records the finite shared-tail modulus join used when two regular Cauchy rational names must be read at one downstream tolerance. It sits beside the regular rational source of chapter 301, the finite window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the meet-side comparison packet of chapter 462. The packet names the selected common threshold as displayed BEDC data; it does not import host maxima, quotient Cauchy classes, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7402.1 (Regular Cauchy modulus join carrier)**.** *A* $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M_0, M_1, T, W_0, W_1, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M_0 and M_1 are the two modulus-input rows whose tail demands must be served by one shared threshold.*
2. *T is the selected join-threshold row, recording a displayed bound large enough for both modulus inputs at the requested dyadic tolerance.*
3. *W_0 and W_1 are the* $\mathsf{StreamName}^{\uparrow}$ *window rows opened only after T is displayed.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested radius, the two inherited tolerance budgets, and the slack accounting for the shared threshold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the two windows and the dyadic ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the modulus inputs, join threshold, windows, tolerance ledger, and readback row have been replayed.*
7. *H records componentwise* hsame *transport for $M_0, M_1, T, W_0, W_1, Q, R, E$ and the local naming row.*
8. *C records displayed* Cont *replay from a downstream tolerance request through the two modulus inputs, selected threshold, windows, ledger, readback, seal, provenance, and local name.*
9. *P records* Pkg *provenance over* $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusJoin}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $M_0, M_1, T, W_0, W_1, Q, R, E, H, C, P, N$. It has no coordinate for an ambient maximum operation, quotient-stream equality, a selected limit, trichotomy, countable choice, or real completeness outside the displayed packet.

Theorem 7402.2 (Regular Cauchy modulus join NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ *carrier*

$$J = (M_0, M_1, T, W_0, W_1, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusJoin}^{\uparrow}}$ *surface consists exactly of carrier admission for the two modulus-input rows, the selected join threshold, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming row. Every obligation is read through those rows and no obligation is stored in host maxima, quotient streams, selected limits, trichotomy, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 7402.1. The tuple displays the two modulus inputs, the selected threshold, the two window rows, the dyadic ledger, the regular rational readback, the terminal real seal, and the structural rows.

The tail side is exhausted by M_0, M_1, T, W_0, W_1, Q. The threshold row T is the only common bound available to both modulus inputs, and the window rows are opened only through the displayed tolerance ledger.

The readback side is exhausted by R and E. The regular rational row receives the paired windows and the ledger, while the real seal appears only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no additional coordinate, the excluded host maximum, quotient, selector, choice, trichotomy, and ambient-completeness data cannot become NameCert obligations. □

Theorem 7402.3 (Regular Cauchy modulus join shared-tail handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ *carrier reads the finite route*

$$(M_0, M_1) \longmapsto T \longmapsto (W_0, W_1) \longmapsto Q \longmapsto R \longmapsto E.$$

The route exposes a single shared threshold before either stream window or regular rational readback is consumed. It exports no host maximum operation, quotient Cauchy class, selected limit, trichotomy, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7402.2. It fixes the public handoff surface at the rows $M_0, M_1, T, W_0, W_1, Q, R, E, H, C, P, N$.

Read the modulus side first. The two input rows state the tail demands, and the threshold row T is the displayed common bound consumed by both of them.

Next read the window and real-name side. The rows W_0, W_1 open the finite observations under that threshold, Q records the tolerance ledger, R supplies the regular rational readback, and E seals the route for real-facing consumers.

The structural rows keep the handoff confined to the displayed packet. Therefore a consumer that reaches a stream, rational readback, or real seal without the stated route is not reading the accepted $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ carrier. □

Theorem 7402.4 (Regular Cauchy modulus join real seal route)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ *packet is downstream of the two modulus inputs, the selected*

join threshold, the paired $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. Hence later L10 regular-Cauchy comparison routes may cite this packet as the concrete shared-tail join certificate, but they receive no additional real-completeness witness from it.*

Proof. Use the shared-tail handoff of Definition 7402.3. The displayed route reaches the real seal only after the threshold, windows, ledger, and readback rows have been consumed in order.

The meet-side sibling chapter 462 supplies the comparison-facing common-modulus row, while this packet supplies the join-facing selected threshold used by downstream L10 regular Cauchy routes. Both are read as finite NameCert rows, not as ambient host operations.

Thus the real seal is a terminal boundary for the displayed shared-tail certificate. It cannot be used as a hidden proof of real completeness, selected limits, quotient equality, or trichotomy. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading both modulus inputs, the selected join threshold, paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchymodulusmeet: no listed sibling is bijective to $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ because this packet binds two modulus inputs, one selected shared threshold, paired finite windows, a tolerance ledger, regular rational readback, real seal, transport, replay, provenance, and local naming as one L10 shared-tail route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusJoin}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusJoin}^{\uparrow}$ packet is generated from two modulus-input rows, one selected join-threshold row, paired StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusJoin}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7402.1, Definition 7402.2, Definition 7402.3, and Definition 7402.4 seed the L10 shared-tail modulus join route over StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusJoin}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host maxima, quotient Cauchy classes, selected limits, trichotomy, countable-choice selectors, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove two-input admission, selected-threshold exposure, paired window scheduling, tolerance-ledger readback, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyModulusJoinUp.

Chapter 7403

A Concrete Naming Certificate for CompactLocatedMaximumPrinciple$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the CompactLocatedMaximumPrinciple$^{\uparrow}$ packet names the finite-net route by which a located real-valued continuous map on a compact metric source exposes a maximum witness as BEDC certificate data. It sits after the compact metric source of chapter 129, the finite epsilon-net schedule of chapter 1767, the located real continuity modulus of chapter 3809, and the terminal real seal of chapter 38. The packet is a finite witness ledger for one displayed maximum read; it is not open-cover compactness, Heine–Borel compactness for arbitrary subsets, choice, quotient completion, or ambient Real$^{\uparrow}$ completeness.

Definition 7403.1 (Compact located maximum principle carrier)**.** *A* CompactLocatedMaximumPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, E, C, R, W, H, T, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing the compact metric packet only through its displayed metric, total-bounded, and finite-net consumer coordinates.*
2. *E is the* FiniteEpsilonNet$^{\uparrow}$ *schedule row, containing the finite centers, tolerance request, and coverage ledger consumed by the maximum search.*
3. *C is the* LocatedRealContinuity$^{\uparrow}$ *modulus row, read over the same compact metric source and the displayed finite-net tolerance.*
4. *R is the* Real$^{\uparrow}$ *seal row for all located real comparisons reached by the finite schedule.*
5. *W is the maximum-witness ledger, recording the selected displayed center, its located real value seal, and the comparison rows showing that every inspected finite-net center is bounded by that value within the consumed modulus budget.*
6. *H records componentwise* hsame *transport for the compact metric packet, finite epsilon-net schedule, continuity modulus, real comparison seals, witness ledger, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from a maximum request through K, E, C, R, W, H, P, N.*
8. *P is the* Pkg *provenance over* CompactLocatedMaximumPrinciple$^{\uparrow}$, CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, LocatedRealContinuity$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, C, R, W, H, T, P, N$. It has no coordinate for open-cover compactness, Heine–Borel compactness for arbitrary subsets, choice, quotient completion, ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host-level supremum, or an unrestricted global maximum theorem.*

Theorem 7403.2 (Compact located maximum principle NameCert obligations)**.** *For every accepted* $\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}$ *carrier*

$$M = (K, E, C, R, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric packet, finite epsilon-net schedule, located real continuity modulus, real comparison seals, maximum witness ledger, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* hsame *to move K, E, C, R, W, T, P, N componentwise without replacing any row by host compactness or host real equality;*
3. *finite-net coverage, requiring every maximum read to consume the displayed* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *centers and coverage ledger before inspecting the witness row;*
4. *located-continuity modulus consumption, requiring the modulus row C to translate the finite-net tolerance into the comparison budget used by W;*
5. *maximum-witness ledger nonescape, requiring W to record only the displayed center, its real seal, and the finite comparison rows certified through R, with no host supremum, choice-selected maximizer, quotient completion, open-cover compactness, Heine–Borel compactness for arbitrary subsets, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No compact located maximum obligation is stored outside these displayed rows.

Proof. Project the carrier of Definition 7403.1. The compact metric packet K, finite epsilon-net schedule E, located real continuity modulus C, real seal row R, and witness ledger W are explicit coordinates of the accepted packet.

Carrier admission and classifier transport are therefore finite-row projections. The transport row H moves the displayed coordinates componentwise, and the replay row T replays the same route; no host compactness theorem or host real equality appears as an alternate classifier coordinate.

Finite-net coverage is the projection to E. A maximum request can inspect the witness ledger only after the displayed centers, tolerance, and coverage ledger have been read, so the route is the finite schedule of chapter 1767 rather than an open-cover argument.

Located-continuity modulus consumption is the projection to C together with the real seal row R. The modulus row supplies the comparison budget used by the finite search, and the real seals record the located comparisons exposed to the witness ledger.

Finally, nonescape follows from row exhaustion. The carrier contains no coordinate for a host supremum, choice-selected maximizer, quotient completion, open-cover compactness, Heine–Borel compactness for arbitrary subsets, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the local $\mathsf{NameCert}$ can certify only the displayed finite witness ledger. □

Falsifiable prediction: A consumer that claims a $\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}$ maximum read while omitting the $\mathsf{CompactMetric}^{\uparrow}$ packet, $\mathsf{FiniteEpsilonNet}^{\uparrow}$ schedule, $\mathsf{LocatedRealContinuity}^{\uparrow}$ modulus, $\mathsf{Real}^{\uparrow}$ comparison seal, maximum witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 7403.1.

Independence witness: compactmetric, finite-epsilon-net, located-real-continuity, real: no listed sibling is bijective to $\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}$ because this packet alone binds a compact metric source, a finite epsilon-net schedule, a located real continuity modulus, real comparison seals, a maximum witness ledger, transport, replay, provenance, and local naming as one finite maximum-principle certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}$ packet is generated from compact metric rows, finite epsilon-net schedule rows, located real continuity modulus rows, RealUp comparison seals, a maximum witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7403.1 and Definition 7403.2 seed the finite-net located maximum witness route for compact metric real-valued consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLocatedMaximumPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel compactness for arbitrary subsets, choice, quotient completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite-net coverage, located-continuity modulus consumption, real comparison sealing, maximum-witness ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactLocatedMaximumPrincipleUp.

Chapter 7404

A Concrete Naming Certificate for $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy moduli, the $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$ packet records the finite comparison surface through which two displayed Cauchy-modulus witnesses may be related before any quotient stream, selected limit, or ambient completeness theorem is used. It sits beside the observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a comparison certificate for finite modulus schedules, not an equality principle for completed real numbers.

Definition 7404.1 (Finite Cauchy modulus comparison carrier)**.** *A* $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M_0, M_1, W_0, W_1, D, R, E, H, C, P, N)$$

with the following rows:

1. *M_0 is the first displayed modulus schedule, recording a finite list of precision requests and the index bounds assigned to them:*

$$n_0 \mapsto k^0_{n_0}, \quad n_1 \mapsto k^0_{n_1}, \quad \ldots, \quad n_s \mapsto k^0_{n_s}.$$

2. *M_1 is the second displayed modulus schedule on the same precision list:*

$$n_0 \mapsto k^1_{n_0}, \quad n_1 \mapsto k^1_{n_1}, \quad \ldots, \quad n_s \mapsto k^1_{n_s}.$$

3. *W_0 and W_1 are the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows. They expose exactly the stream observations demanded by M_0 and M_1 on the displayed precision list.*

4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-comparison ledger. It records the dyadic tolerance cells in which the two schedules are compared and the positive margin used by the comparison route.*

5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational approximants obtained from the two windows after the dyadic tolerance comparison has been displayed.*

6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after M_0, M_1, W_0, W_1, D, R have been read.*

7. *H records componentwise* hsame *transport for the two modulus schedules, the shared windows, the dyadic tolerance ledger, the regular readback row, the Real seal, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$M_0, M_1 \longmapsto W_0, W_1 \longmapsto D \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyModulusComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the two displayed schedules, their shared finite windows, the dyadic tolerance-comparison ledger, regular rational readback, the terminal Real seal, transport, replay, provenance, and local names. It has no coordinate for quotient stream equality, selected limits, ambient completeness, or host real equality.

Theorem 7404.2 (Finite Cauchy modulus comparison NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$ *carrier*

$$B = (M_0, M_1, W_0, W_1, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyModulusComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two modulus schedules, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparison,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*

2. *common-window exposure, requiring both schedules to be read on the same finite precision list before any tolerance comparison is available;*

3. *dyadic comparison exactness, requiring the tolerance ledger D to mediate every comparison between schedule bounds;*

4. *readback compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to use exactly the stream windows and dyadic tolerance cells displayed by W_0, W_1, D;*

5. *Real-seal nonescape, requiring any downstream Real consumer to pass through E, H, C, P, N after the comparison rows have been displayed.*

No finite Cauchy-modulus comparison obligation is stored outside these rows.

Proof. Project the carrier of Definition 7404.1. The two schedules M_0 and M_1 expose only finite precision requests and their assigned bounds.

The common-window obligation reads W_0 and W_1 on the same displayed precision list. This keeps the comparison inside the StreamName windows rather than moving to a quotient stream.

The dyadic comparison row D is then the only route from schedule data to tolerance data. The readback row R receives exactly those windows and tolerance cells, matching the regular rational reconstruction discipline of chapter 301.

Finally the Real seal E and the structural rows H, C, P, N transport, replay, source, and name the displayed route. Since no carrier coordinate supplies selected limits, ambient completeness, or host real equality, the listed rows exhaust the local NameCert surface. □

Lean4 (statement) `BEDC.Derived.FiniteCauchyModulusComparisonUp`

Falsifiable prediction: A $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$ consumer that compares two Cauchy-modulus witnesses without reading the two displayed schedules, the shared $\mathsf{StreamName}^{\uparrow}$ windows,

the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal refutes the carrier surface.

Independence witness: streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole comparison packet because each sibling contributes observations, tolerance arithmetic, readback, or sealing, while this chapter binds the finite comparison route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyModulusComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyModulusComparison}^{\uparrow}$ packet is generated from two finite Cauchy-modulus schedules, shared StreamName windows, a dyadic tolerance-comparison ledger, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyModulusComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7404.1 and Definition 7404.2 seed the finite comparison surface over two Cauchy-modulus schedules, shared StreamName windows, dyadic tolerance comparison, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyModulusComparison}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FiniteCauchyModulusComparisonUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, ambient Real completeness, host real equality, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window exposure, dyadic comparison exactness, readback compatibility, Real-seal nonescape, and finite NameCert coverage for BEDC.Derived.FiniteCauchyModulusComparisonUp.

Chapter 7405

A Concrete Naming Certificate for MetaCICResidualSubstitutionComparison$^{\uparrow}$

Following Barendregt-style residual and confluence analysis, the MetaCICResidualSubstitutionComparison$^{\uparrow}$ packet records the finite comparison surface through which residual-substitution data may be audited before a confluence consumer cites a conditional diamond route. It sits beside the residual-substitution boundary of chapter 2181, the critical-pair ledger of chapter 3437, and the local confluence frontier of chapter 7346. The packet compares displayed residual-substitution rows; it does not prove global confluence or discharge substitution compatibility.

Definition 7405.1 (MetaCIC residual-substitution comparison carrier)**.** *A* MetaCICResidualSubstitutionComparison *carrier is a finite* BHist *packet*

$$B = (T_0, T_1, S_0, S_1, J, O, Q, H, C, P, N)$$

with the following rows:

1. *T_0 and T_1 are the typed-window rows. They display the two finite MetaCIC term windows whose residual substitution traces are compared.*
2. *S_0 and S_1 are the residual-substitution trace rows. They record the two substituted residual branches and the binder windows used by the comparison.*
3. *J is the local-join row. It records the bounded local join attempt between the two residual traces, without promoting the attempt to a full parallel diamond.*
4. *O is the obstruction-retention row. It keeps the named substitution-compatibility boundary visible whenever the local join is conditional.*
5. *Q is the bounded-checker row. It records the finite checker requests used to compare the typed windows, residual traces, local join, and obstruction row.*
6. *H records componentwise* hsame *transport for the typed windows, residual traces, local join, obstruction retention, bounded checker, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$T_0, T_1 \longmapsto S_0, S_1 \longmapsto J, O, Q \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* MetaCICResidualSubstitutionComparison$^{\uparrow}$*, MetaCIC typed windows, residual substitution traces, local join data, obstruction rows, bounded checker data,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed typed-window, residual-trace, local-join, obstruction-retention, bounded-checker, transport, replay, provenance, and local naming rows. It has no coordinate for global confluence, premise-free substitution compatibility, arbitrary closed Church–Rosser, quotient syntax, or host evaluator soundness.

Theorem 7405.2 (MetaCIC residual-substitution comparison NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}$ *carrier*

$$B = (T_0, T_1, S_0, S_1, J, O, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the typed windows, residual-substitution traces, local join, obstruction retention, bounded checker, transport, replay, provenance, and local naming rows;*
2. *typed-window exposure, requiring T_0 and T_1 to be read before either residual trace is compared;*
3. *residual-trace comparison, requiring S_0 and S_1 to remain paired with their typed windows and binder windows;*
4. *local-join boundedness, requiring J to record only the displayed local join attempt inspected by Q;*
5. *obstruction retention, requiring O to keep the residual-substitution compatibility boundary visible whenever the comparison is conditional;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows.*

No MetaCIC residual-substitution comparison obligation is stored outside these rows.

Proof. Project the carrier of Definition 7405.1. The typed windows T_0, T_1 are finite term-and-binder windows, and the residual traces S_0, S_1 are read only through those windows.

The local join row J is bounded by the checker row Q. Thus the packet can record a finite comparison between two residual traces without claiming a global parallel diamond.

The obstruction row O remains visible whenever the join is conditional. This matches the premise-retention discipline of chapter 2181 and keeps the confluence audit map from silently treating a named input as a theorem.

Finally H, C, P, N transport, replay, source, and name the same route. Since no coordinate supplies global confluence, premise-free substitution compatibility, arbitrary closed Church–Rosser, quotient syntax, or host evaluator soundness, the obligation surface is exactly the finite comparison packet displayed above. □

Lean4 (statement) `BEDC.Derived.MetaCICResidualSubstitutionComparisonUp`

Falsifiable prediction: A $\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}$ confluence-audit read factors through finite typed-window rows, residual-substitution trace rows, local-join data, obstruction retention, bounded checker data, transport, replay, provenance, and local NameCert rows before any conditional diamond handoff is cited.

Independence witness: metacic-residual-substitution-boundary, metacic-critical-pair-ledger, metacic-local-confluence-frontier: the comparison packet is not bijective to any listed sibling because it

jointly carries two typed windows, two residual traces, a bounded local join, obstruction retention, bounded checking, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}$ packet is generated from two finite typed-window rows, two residual-substitution trace rows, a bounded local-join row, obstruction-retention data, a bounded-checker row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7405.1 and Definition 7405.2 seed the finite residual-substitution comparison packet over typed windows, residual traces, local joins, obstruction retention, bounded checking, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionComparison}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MetaCICResidualSubstitutionComparisonUp`

Bridge status: none

Not claimed: This chapter does not close global confluence, premise-free substitution compatibility, arbitrary closed Church–Rosser, quotient syntax, host evaluator soundness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-trace comparison, local-join boundedness, obstruction retention, provenance exhaustion, and finite NameCert coverage for BEDC.Derived.MetaCICResidualS

Chapter 7406

A Concrete Naming Certificate for $\mathsf{Kirchberger}^{\uparrow}$

Following Kirchberger's finite convex separation theorem, the $\mathsf{Kirchberger}^{\uparrow}$ packet records the finite BEDC surface for detecting separation of two finitely displayed point families by checking bounded small subfamilies. It sits over the affine-coordinate carrier of chapter 206, the convex-set carrier of chapter 208, the polytope row of chapter 209, and the finite-dimensional separating-functional row of chapter 3905. The packet names only the displayed finite subfamily ledger and its separation obligations; it does not assert an ambient compactness theorem, selected global hyperplane, quotient convex hull equality, or infinite Helly-family principle.

Definition 7406.1 (Kirchberger finite separation carrier)**.** *A* $\mathsf{Kirchberger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, C, D, P, S, L, E, H, T, Q, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AffineSpace}^{\uparrow}$ *coordinate row, fixing the finite-dimensional affine window in which the two point families are read.*
2. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *row, recording the two displayed convex hull requests generated by the finite point families.*
3. *D is the* $\mathsf{Polytope}^{\uparrow}$ *row, naming the finite hull presentations and their face or vertex ledgers.*
4. *P is the point-family row, listing the two color classes and the finite subfamily windows inspected by the certificate.*
5. *S is the small-subset obligation row, recording the bounded subfamilies whose convex hulls must be separated.*
6. *L is the local separation row supplied by* $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ *for each displayed small-subset request.*
7. *E is the exactness row, asserting that every accepted small-subset request is routed through the same affine, convex, polytope, and local-separation data.*
8. *H records componentwise* hsame *transport for $A, C, D, P, S, L, E, T, Q, N$.*
9. *T records displayed* Cont *replay through the finite point-family, subset, convex-hull, and separation rows.*

10. *Q is the* Pkg *provenance over* Kirchberger$^{\uparrow}$, AffineSpace$^{\uparrow}$, ConvexSet$^{\uparrow}$, Polytope$^{\uparrow}$, FiniteDimensionalHahnBanachSe BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{Kirchberger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a selected global separating hyperplane, quotient convex hull equality, ambient compactness, infinite choice over subfamilies, or an unbounded Helly-style search.

Theorem 7406.2 (Kirchberger finite subset obligations)**.** *For every accepted* Kirchberger$^{\uparrow}$ *carrier $K = (A, C, D, P, S, L, E, H, T, Q, N)$, the local* NameCert$_{\text{Kirchberger}^{\uparrow}}$ *surface consists exactly of affine-coordinate admission, convex-hull readback, polytope finite-hull presentation, two-color point-family exposure, small-subset obligation accounting, finite-dimensional local separation, exactness through the same displayed rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7406.1. The rows $A, C, D, P, S, L, E, H, T, Q, N$ are the complete visible inventory.

The affine, convex, and polytope rows A, C, D fix the ambient finite-dimensional window and the finite hull presentations. The point-family and small-subset rows P, S then identify the bounded subfamilies whose separation must be displayed.

The local separation row L is read only through FiniteDimensionalHahnBanachSeparation$^{\uparrow}$, and the exactness row E binds those local reads to the same subfamily requests. Transport, replay, provenance, and naming through H, T, Q, N preserve that finite route, so no excluded global or quotient object is exported. □

Theorem 7406.3 (Kirchberger NameCert nonescape)**.** *An accepted* Kirchberger$^{\uparrow}$ *packet exports no selected global separating hyperplane, no quotient equality of convex hulls, no ambient compactness theorem, no infinite subfamily choice schedule, and no unbounded Helly-family search. Every public read factors through the displayed affine window, convex and polytope hull rows, two-color point-family row, small-subset obligation ledger, finite-dimensional local-separation row, exactness row, componentwise transport, replay, provenance, and local NameCert row.*

Proof. Inspect the carrier coordinates. They list only the finite affine, convex, polytope, point-family, subset-obligation, local-separation, exactness, transport, replay, provenance, and naming rows.

A global separation read must therefore pass through the bounded subfamily ledger S and the local separation row L. The carrier has no coordinate from which a host hyperplane, quotient convex hull identity, or infinite family selection can be read.

Thus any downstream convexity consumer receives Kirchberger's finite obstruction surface as NameCert data. The excluded principles require rows absent from Definition 7406.1, so they cannot escape the packet. □

Theorem 7406.4 (Kirchberger seed scope)**.** *The seed scope of* Kirchberger$^{\uparrow}$ *is the finite convex separation route*

$$A, C, D, P \longmapsto S, L, E \longmapsto H, T, Q, N.$$

It binds the affine-coordinate window, convex-hull and polytope rows, point-family ledger, bounded small-subset obligations, finite-dimensional local-separation witnesses, exactness, componentwise transport, replay, provenance, and local naming. It does not close Lean verification, bridge checking, public registry export, or any ambient global separation theorem.

Proof. Read the route displayed in the statement from the carrier definition. The left side supplies the finite ambient and point-family data, the middle supplies the small-subset and local-separation obligations, and the right side supplies transport, replay, provenance, and naming.

Definition 7406.2 identifies this route as the local NameCert obligation surface. Definition 7406.3 gives the boundary: consumers cannot replace the finite route by host separation, quotient hull equality, compactness, or unbounded choice.

Consequently the chapter seeds exactly the displayed finite convex-separation packet. Higher closure grades require checked carrier admission, classifier stability, local-separation exactness, and a public export surface. □

Falsifiable prediction: If the small-subset obligation row S is removed, Definition 7406.2 no longer distinguishes a Kirchberger finite-obstruction packet from a single finite-dimensional separation certificate.

Independence witness: affinespace, convexset, polytope, finite-dimensional-hahn-banach-separation: $\mathsf{Kirchberger}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds two-color point-family data, bounded small-subset obligations, finite hull presentations, and local finite-dimensional separation rows into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{Kirchberger}^{\uparrow}}$.

Constructive story: A $\mathsf{Kirchberger}^{\uparrow}$ packet is generated from affine-coordinate rows, convex-set rows, polytope finite-hull rows, two-color point-family rows, small-subset obligation rows, finite-dimensional local-separation rows, exactness data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Kirchberger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7406.1, Definition 7406.2, Definition 7406.3, and Definition 7406.4 seed the finite Kirchberger separation packet over AffineSpaceUp, ConvexSetUp, PolytopeUp, FiniteDimensionalHahnBanachSeparationUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Kirchberger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public registry export, selected global hyperplanes, quotient convex hull equality, ambient compactness, infinite subfamily choice, or unbounded Helly-family search.

Upgrade path: Obligation closure requires checked carrier admission, finite point-family exposure, bounded small-subset accounting, local finite-dimensional separation exactness, classifier stability, and nonescape for BEDC.Derived.KirchbergerUp.

Chapter 7407

A Concrete Naming Certificate for MetaCICResidualDiamondBudget$^{\uparrow}$

The MetaCICResidualDiamondBudget$^{\uparrow}$ packet realizes one finite residual-substitution confluence frontier from section 9386.3. It sits beside the residual-substitution boundary of chapter 2181, the residual checker of chapter 3827, the residual-diamond scheduler of chapter 6857, the residual-diamond witness of chapter 7219, and the residual-join budget of chapter 7378. The packet records a finite typed window, residual-pair budget, local-diamond candidate ledger, checker fuel, obstruction retention, transport, replay, provenance, and local naming; it does not claim unrestricted residual substitution, arbitrary closed parallel diamond, Church–Rosser, quotient conversion equality, or a host confluence oracle.

Definition 7407.1 (MetaCIC residual diamond budget carrier)**.** *A* MetaCICResidualDiamondBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, R, D, J, C, O, H, K, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, fixing the closed source term, local context fragment, redex shape, and observation depth inspected by the packet.*
2. *R is the residual-frontier row, naming the residual substitutions visible inside the typed window.*
3. *D is the residual-pair budget row, listing the finitely many residual pairs for which the local-diamond question is inspected.*
4. *J is the join-candidate ledger, recording the proposed common reducts for the residual pairs in D.*
5. *C is the bounded checker row, recording the finite fuel, replay trace, and checker status used to inspect the candidate joins.*
6. *O is the obstruction-retention row, preserving any residual-substitution or subject-reduction boundary that the bounded checker has not discharged.*
7. *H records componentwise* hsame *transport for $T, R, D, J, C, O, K, P, N$.*
8. *K records displayed* Cont *replay from a residual-diamond request through the typed window, frontier, pair budget, join candidates, checker row, and obstruction row.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetacicResidualDiamondScheduler}^{\uparrow}$, $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$, $\mathsf{MetaCICResidualJoinBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, D, J, C, O, H, K, P, N$. It has no coordinate for unrestricted residual substitution, an arbitrary closed parallel diamond, a global Church–Rosser endpoint, quotient conversion equality, unbounded checking, or a host confluence oracle.

Theorem 7407.2 (MetaCIC residual diamond budget obligations)**.** *For every accepted* MetaCICResidualDiamondBu *carrier $B = (T, R, D, J, C, O, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}}$ *surface consists exactly of typed-window admission, residual-frontier exposure, residual-pair budget accounting, join-candidate ledger visibility, bounded-checker exactness, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7407.1. The visible rows are exactly $T, R, D, J, C, O, H, K, P, N$.

The typed window T and residual frontier R fix the local MetaCIC source in which residuals are meaningful. The residual-pair budget D narrows the local-diamond request to finitely many inspected pairs, and J records only the proposed joins for those pairs.

The bounded checker row C evaluates the same finite ledger, while the obstruction row O keeps undisclosed residual-substitution or subject-reduction boundaries visible. Transport, replay, provenance, and naming through H, K, P, N preserve this route and introduce no additional confluence endpoint. □

Theorem 7407.3 (MetaCIC residual diamond budget nonescape)**.** *An accepted* $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$ *packet exports no unrestricted residual-substitution principle, no arbitrary closed parallel diamond, no global Church–Rosser theorem, no quotient conversion equality, no unbounded checker, and no host confluence oracle. Every confluence-facing read factors through the displayed typed window, residual-frontier row, residual-pair budget, join-candidate ledger, bounded checker, obstruction row, componentwise transport, replay, provenance, and local NameCert row.*

Proof. Inspect the carrier coordinates. They list only the finite typed-window, residual-frontier, residual-pair, join-candidate, bounded-checker, obstruction, transport, replay, provenance, and naming rows.

Any diamond-facing read must pass through the pair budget D, the candidate ledger J, and the bounded checker C before it reaches a consumer. The obstruction row O remains present whenever the bounded check has not discharged a residual-substitution or subject-reduction boundary.

The excluded endpoints require coordinates absent from the carrier. Since no row supplies unrestricted substitution compatibility, arbitrary closed parallel diamonds, quotient conversion equality, unbounded checking, or a host oracle, those objects cannot escape this NameCert packet. □

Theorem 7407.4 (MetaCIC residual diamond budget confluence frontier)**.** *The* $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$ *packet records the residual-substitution confluence frontier as the finite route*

$$T, R \longmapsto D, J, C, O \longmapsto H, K, P, N.$$

It turns a candidate local-diamond request into a bounded residual-pair budget with explicit join candidates, checker status, and obstruction retention. It does not remove the residual-substitution boundary named in section 9386.3, and it does not upgrade a finite local budget into arbitrary closed-term confluence.

Proof. Read the route from Definition 7407.1. The rows T, R locate the typed residual frontier, while D, J, C, O record the finite pair budget, candidate joins, bounded checking, and retained obstruction.

Definition 7407.2 identifies these rows as the local NameCert obligation surface. Definition 7407.3 prevents a downstream confluence consumer from replacing the finite budget by unrestricted residual substitution, arbitrary closed parallel diamonds, or quotient conversion equality.

Thus the packet is a concrete finite frontier object for later Lean work: it makes the residual-diamond budget inspectable without pretending that the open confluence endpoint has already been discharged. □

Falsifiable prediction: If the residual-pair budget row D is removed, Definition 7407.4 can no longer distinguish a bounded residual-diamond packet from a single local-join witness.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-residual-checker, metacic-residual-diamond-scheduler, metacic-residual-diamond-witness, metacic-residual-join-budget: $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a typed residual frontier, residual-pair budget, join-candidate ledger, bounded checker, obstruction retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$ packet is generated from typed-window rows, residual-frontier rows, residual-pair budgets, join-candidate ledgers, bounded-checker rows, obstruction-retention rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7407.1, Definition 7407.2, Definition 7407.3, and Definition 7407.4 seed the finite residual-diamond budget packet for MetaCIC confluence-frontier consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, unrestricted residual substitution, arbitrary closed parallel diamond, global Church–Rosser, quotient conversion equality, unbounded checking, host confluence oracles, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window coverage, residual-frontier exposure, residual-pair budget accounting, join-candidate ledger visibility, bounded-checker exactness, obstruction retention, classifier stability, and nonescape for BEDC.Derived.MetaCICResidualDiamondBudgetUp.

Chapter 7408

A Concrete Naming Certificate for $\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}$

Bishop–Bridges constructive analysis treats quotient estimates for regular Cauchy names as guarded calculations: the numerator tail, denominator tail, and denominator away-from-zero witness must all remain visible before a real quotient seal is read. The $\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}$ packet records that finite guarded quotient-tail budget as a NameCert route between the regular rational sources of chapter 301, the dyadic tolerance ledger of chapter 291, the reciprocal modulus surface of chapter 7354, the guarded division handoff of chapter 1943, and the terminal real seal of chapter 38. The packet names only the displayed quotient-tail route; it does not supply total division, quotient representative equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7408.1 (Regular Cauchy quotient tail estimate carrier)**.** *A* $\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (N, D, A, M, S, L, V, E, H, C, P, K)$$

with the following displayed rows:

1. *N is the numerator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displaying the regular rational name whose quotient tail is being estimated.*
2. *D is the denominator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displayed as a separate regular rational name.*
3. *A is the located away-from-zero lower-bound row for the denominator, recording the positive guard that makes quotient-tail control meaningful.*
4. *M is the* $\mathsf{RegularCauchyReciprocalModulus}^{\uparrow}$ *row, turning D and A into a finite reciprocal modulus budget.*
5. *S is the quotient tail schedule, recording the numerator tail index, denominator tail index, reciprocal index, and final quotient precision request.*
6. *L is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic error split for numerator tail, denominator tail, reciprocal budget, and multiplication by the guarded reciprocal.*
7. *V is the* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *handoff row, available only after N, D, A, M, S, L have been displayed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary for the quotient tail estimate.*

9. *H records componentwise* hsame *transport for the two sources, away-from-zero guard, reciprocal modulus, tail schedule, dyadic ledger, division handoff, seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay of the route*

$$N, D, A, M, S, L \longmapsto V \longmapsto E.$$

11. *P is the* Pkg *provenance over* RegularCauchyQuotientTailEstimate$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyReciprocalModulu RegularCauchyDivision$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *K is the local* NameCert$_{\text{RegularCauchyQuotientTailEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $N, D, A, M, S, L, V, E, H, C, P, K$. It has no coordinate for total division, quotient representative equality, hidden denominator choice, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7408.2 (Regular Cauchy quotient tail estimate NameCert obligations)**.** *For every accepted* RegularCauchyQuotientTailEstimate$^{\uparrow}$ *carrier $Q = (N, D, A, M, S, L, V, E, H, C, P, K)$, the local* NameCert$_{\text{RegularCauchyQuotientTailEstimate}^{\uparrow}}$ *surface consists exactly of numerator and denominator* RegSeqRat$^{\uparrow}$ *source visibility, located denominator away-from-zero lower-bound visibility, reciprocal modulus routing, quotient tail scheduling, dyadic tolerance-ledger exposure,* RegularCauchyDivision$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7408.1. The source rows N, D and the away-from-zero guard A are the complete displayed input surface for a guarded quotient estimate.

Read the reciprocal and tolerance rows next. The modulus row M, schedule row S, and dyadic ledger L record the finite budgets needed before a quotient tail can be passed to a division consumer.

Finally the division handoff V, seal boundary E, and structural rows H, C, P, K preserve, replay, source, and name that same finite route. Since no excluded total-division, quotient-representative, hidden-choice, trichotomy, countable-choice, or ambient-completeness row appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7408.3 (Regular Cauchy quotient tail estimate Real seal boundary)**.** *Every Real-facing quotient-tail read of an accepted* RegularCauchyQuotientTailEstimate$^{\uparrow}$ *carrier factors through the displayed numerator and denominator* RegSeqRat$^{\uparrow}$ *sources, the denominator away-from-zero lower bound, the* RegularCauchyReciprocalModulus$^{\uparrow}$ *row, the quotient tail schedule, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* RegularCauchyDivision$^{\uparrow}$ *handoff, and the terminal* Real$^{\uparrow}$ *seal boundary. The route exports no total division, quotient representative equality, hidden denominator choice, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Apply Definition 7408.2. It identifies N, D, A, M, S, L as the only finite input surface before the guarded division handoff can be read.

The RegularCauchyDivision$^{\uparrow}$ row V is therefore downstream of both regular rational sources, the denominator guard, the reciprocal modulus budget, the tail schedule, and the dyadic tolerance ledger. A Real-facing consumer reaches E only through that displayed handoff.

The transport, replay, provenance, and local naming rows preserve the same dependency order. Because the carrier has no row for total division, quotient representative equality, hidden denominator choice, trichotomy, countable choice, or ambient completeness, the terminal seal remains a finite guarded quotient-tail boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}$ packet is generated from numerator and denominator RegSeqRat sources, a located away-from-zero lower bound for the denominator, a RegularCauchyReciprocalModulus row, a quotient tail schedule, a DyadicRatCore tolerance ledger, a RegularCauchyDivision handoff, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7408.1, Definition 7408.2, and Definition 7408.3 seed the finite guarded quotient-tail budget for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientTailEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total division, quotient representative equality, hidden denominator choice, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, numerator and denominator RegSeqRat source visibility, away-from-zero lower-bound visibility, reciprocal modulus routing, quotient tail scheduling, dyadic tolerance exactness, RegularCauchyDivision handoff, RealUp seal boundary, transport, replay, provenance, and local naming.

Chapter 7409

A Concrete Naming Certificate for $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ packet records the bounded monotone convergence route as finite BEDC data. It consumes the $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ source of chapter 535, the convergence-seal boundary of chapter 918, the finite window surface of chapter 126, the dyadic tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the constructive route from bounded monotone data to a Cauchy regular rational handoff and then to a Real seal; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, choose a host limit, identify quotient streams, or import countable choice.

Definition 7409.1 (Monotone convergence theorem carrier)**.** *A* $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, W, \mu, D, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *row, carrying the bounded monotone source, tail witness, located trap, and Real-seal precursor.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger listing the inspected sequence windows and bound requests.*
3. *μ is the extracted Cauchy modulus row, displayed as finite threshold data read from B and W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the modulus and by the regular-rational readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row produced from W, μ, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the bounded source, modulus, tolerance, and regular readback rows have been displayed.*
7. *L is the accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *row binding the same witness, modulus, readback, and Real seal route.*
8. *H records componentwise* hsame *transport for the bounded source, windows, modulus, dyadic ledger, readback, seal, convergence-seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a precision request through the same bounded monotone route.*

10. *P is the* Pkg *provenance over* $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$, $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceTheorem}^{\uparrow}}$ *row naming exactly this finite convergence packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for selected limits, quotient stream equality, countable-choice schedules, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7409.2 (Monotone convergence theorem carrier surface)**.** *Every accepted* MonotoneConvergenceTheorem *carrier exposes exactly the finite route*

$$\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneC}$$

through the rows $B, W, \mu, D, R, L, E, H, C, P, N$. A consumer receives bounded monotone convergence data only as this displayed BEDC packet, not as a host supremum theorem, a selected limit, quotient real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 7409.1. The rows B, W, μ, D, R, E, L display the bounded witness, finite windows, extracted modulus, tolerance ledger, regular readback, terminal Real seal, and convergence-seal boundary.

Read the analytic route in order. The bounded witness row B supplies the monotone source and trap data, W lists the finite requests, μ records the threshold extracted from those requests, and D supplies the dyadic tolerance cells used by the threshold.

The rows R, L, E then package the regular readback, convergence-seal compatibility, and terminal Real seal. The structural rows H, C, P, N transport, replay, source, and name the same packet, so there is no carrier coordinate from which a selected limit, quotient equality, host supremum, or ambient completeness principle can be exported. □

Theorem 7409.3 (Monotone convergence bounded carrier admission)**.** *For every accepted* MonotoneConvergenceTh *carrier $M = (B, W, \mu, D, R, E, L, H, C, P, N)$, the bounded monotone source is admitted only through the accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *row B and the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger W. No later modulus, regular-rational, convergence-seal, or Real-facing read may bypass B, W.*

Proof. Start with the carrier projection. The first analytic rows are B and W, and every other analytic coordinate is downstream of them.

The modulus row μ is not an independent tail oracle; it is displayed as threshold data extracted from the bounded witness and the finite window ledger. Thus the dyadic, regular-rational, convergence-seal, and Real rows can be inspected only after B, W have been read.

Transport and replay preserve that order through H and C, while P, N only source and name the same route. Hence a consumer that avoids B, W is not reading the accepted packet. □

Theorem 7409.4 (Monotone convergence Cauchy modulus extraction)**.** *Every Cauchy modulus exported by an accepted* $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ *packet is the displayed row μ extracted from the bounded monotone witness B, the finite window ledger W, and the dyadic tolerance row D. The extraction exports no detached tail oracle, countable-choice schedule, trichotomy branch, host order completeness, or selected limit.*

Proof. Project $M = (B, W, \mu, D, R, E, L, H, C, P, N)$. The row μ occurs between the bounded-window rows and the regular readback row.

The bounded monotone witness supplies the source and trap data, while W records the finite precision request being served. The dyadic row D supplies the tolerance cell for that request, so the threshold in μ is a displayed packet row rather than a function chosen from outside the carrier.

Since no row contains a detached selector, a trichotomy branch, host order completeness, or a selected limit, the exported modulus is exactly the finite extraction recorded by μ. □

Theorem 7409.5 (Monotone convergence RegSeqRat handoff)**.** *Every regular-rational name produced by an accepted* $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ *packet is read through*

$$W, \quad \mu, \quad D, \quad R.$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R is downstream of the finite windows, extracted modulus, and dyadic tolerance ledger, and it exports no quotient stream, selected representative, hidden Cauchy family, or ambient Real-completeness witness.*

Proof. Begin with bounded carrier admission. It places the finite windows W behind the bounded witness row B.

Apply the modulus extraction route. The row μ is read from B, W, D, so the regular-rational request has a displayed threshold and tolerance ledger before R is available.

The handoff row R therefore packages the regular rational readback from finite data only. The structural rows preserve this order and have no coordinate for quotient streams, selected representatives, hidden Cauchy families, or ambient completeness. □

Theorem 7409.6 (Monotone convergence Real seal nonescape)**.** *For every accepted* $\mathsf{MonotoneConvergenceTheorem}$ *packet, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is exposed only after the route*

$$B, W, \mu, D, R, L$$

has been displayed. The seal may be consumed by Real and Completion routes as the named endpoint of the bounded monotone construction, but it cannot be read as ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, a selected supremum, quotient stream equality, or a choice-selected limit.*

Proof. The carrier places E after the bounded witness, finite windows, extracted modulus, dyadic ledger, regular readback, and convergence-seal boundary.

Use the RegSeqRat handoff to keep R downstream of W, μ, D. The convergence-seal row L then binds the same route before the Real seal is visible.

Rows H, C, P, N cannot add analytic data; they only preserve, replay, source, and name the displayed route. Since the carrier has no coordinate for ambient completeness, host equality, selected suprema, quotient streams, or choice-selected limits, the Real seal cannot escape the finite packet. □

Theorem 7409.7 (Monotone convergence ledger exactness)**.** *The local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceTheorem}^{\uparrow}}$ *surface consists exactly of bounded monotone carrier admission, Cauchy modulus extraction, dyadic tolerance readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *compatibility,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming. Equivalently, every obligation-facing consumer of M reads only*

$$B, W, \mu, D, R, E, L, H, C, P, N.$$

Proof. Combine bounded carrier admission with Cauchy modulus extraction. These two routes fix the source, window, threshold, and dyadic rows available to an obligation-facing consumer.

Add the RegSeqRat handoff and Real seal nonescape. They force regular readback and terminal sealing through the same finite route and through the convergence-seal boundary L.

The remaining rows H, C, P, N are structural. They transport, replay, source, and locally name the packet without adding selected limits, quotient equality, choice, trichotomy, host real equality, or ambient completeness. Thus the displayed rows exhaust the local NameCert ledger. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ Real-facing read factors through an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a displayed Cauchy modulus row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a BoundedMonotoneConvergence boundary, componentwise transport, displayed replay, provenance, and local NameCert before the $\mathsf{Real}^{\uparrow}$ seal is public.

Independence witness: boundedmonotonecauchywitness, streamname, dyadicratcore, regseqrat, bounded-monotone-convergence-seal, real: $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly records the bounded monotone source, finite windows, extracted Cauchy modulus, dyadic tolerance ledger, regular rational handoff, convergence-seal boundary, terminal Real seal, transport, replay, provenance, and local naming as one theorem packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneConvergenceTheorem}^{\uparrow}$ packet is generated from a BoundedMonotoneCauchyWitness source row, finite StreamName windows, an extracted Cauchy modulus row, DyadicRatCore tolerance data, RegSeqRat readback, a BoundedMonotoneConvergenceSeal boundary, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7409.1, Definition 7409.2, and Definition 7409.7 seed the constructive bounded monotone convergence route over finite windows, dyadic tolerances, regular rational readback, convergence sealing, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, trichotomy, countable-choice schedules, host real equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus extraction, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneConvergenceTheoremUp.

Chapter 7410

A Concrete Naming Certificate for $\mathsf{RationalFloorBound}^{\uparrow}$

Bishop–Bridges constructive analysis treats integer-part extraction for rationals as bounded search once an explicit Archimedean bound has been displayed. The $\mathsf{RationalFloorBound}^{\uparrow}$ packet records that finite bounded-search floor witness as a NameCert route between the rational carrier of chapter 37, the signed-integer interface of chapter 31, the dyadic scaling ledger of chapter 291, the finite stream and regular-rational readbacks of chapter 126 and chapter 301, and the terminal real-name boundary of chapter 38. The packet names only the displayed rational floor route; it does not import a host floor function, least-number operator, quotient representative, choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7410.1 (RationalFloorBound carrier)**.** *A* $\mathsf{RationalFloorBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, U, B, F, D, T, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *input row, naming the rational whose integer part is being bounded and inspected.*
2. *U is a unary Archimedean bound row, displaying a finite natural bound large enough for the bounded search interval around Q.*
3. *B is the bounded-search trace, recording the inspected integer candidates and the first adjacent pair where the rational input changes from being below the upper candidate to being at least the lower candidate.*
4. *F is the* $\mathsf{Int}^{\uparrow}$ *floor witness row, storing the integer candidate certified by the search trace.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scaling ledger, recording denominator clearing, dyadic comparison cells, and tolerance-free rational comparison data used by the trace.*
6. *T records componentwise* hsame *transport for the rational input, unary bound, search trace, integer witness, dyadic scaling ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$Q, U, D \longmapsto B \longmapsto F.$$

8. *P is the* Pkg *provenance over* $\mathsf{RationalFloorBound}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RationalFloorBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, U, B, F, D, T, C, P, N$. It has no coordinate for host floor, least-number search, quotient representative equality, countable choice, hidden real completeness, or an unbounded integer-part oracle.

Theorem 7410.2 (RationalFloorBound NameCert obligations)**.** *For every accepted* $\mathsf{RationalFloorBound}^{\uparrow}$ *carrier $L = (Q, U, B, F, D, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalFloorBound}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the* $\mathsf{Rat}^{\uparrow}$ *input, unary bound, bounded-search trace,* $\mathsf{Int}^{\uparrow}$ *witness, dyadic scaling ledger, transport, replay, provenance, and local naming rows;*
2. *bounded-search visibility, requiring every inspected candidate and stopping comparison in B to remain displayed under the unary bound U;*
3. *floor inequality, requiring the witness row F to certify $F \leq Q < F + 1$ through rational comparisons read from Q and D;*
4. *dyadic scale compatibility, requiring denominator clearing and comparison cells in D to agree with the rational input and the integer candidates inspected by B;*
5. *ledger nonescape, stating that no host floor function, least-number operator, quotient representative, choice principle, hidden real-completeness row, or unbounded search oracle is exported by the packet.*

Proof. Project the carrier of Definition 7410.1. The rows $Q, U, B, F, D, T, C, P, N$ are the full visible surface of the packet, so a consumer must read the rational input and unary bound before it can inspect any search trace or integer witness.

Read the bounded search trace under the dyadic ledger. The row B records only finitely many inspected integer candidates bounded by U, while D supplies the rational comparison data used to verify the adjacent inequalities around F.

Finally replay through T, C, P, N. Transport, continuation, provenance, and local naming preserve the same finite route and introduce no coordinate for host floor, least-number selection, quotient representatives, choice, hidden completeness, or unbounded search. The listed rows therefore exhaust the NameCert obligations. □

Theorem 7410.3 (RationalFloorBound Archimedean step)**.** *Every accepted* $\mathsf{RationalFloorBound}^{\uparrow}$ *packet exports a finite unary bound and an* $\mathsf{Int}^{\uparrow}$ *floor witness for the displayed rational input Q. The export is the route*

$$Q, U, D \longmapsto B \longmapsto F$$

inside $L = (Q, U, B, F, D, T, C, P, N)$: U supplies the finite search limit, B records the bounded trace, D supplies the dyadic rational comparison ledger, and F is the displayed integer satisfying $F \leq Q < F + 1$. The route uses no host floor function, least-number operator, quotient representative, choice principle, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier admission row. It displays the rational input Q, unary bound U, dyadic scaling ledger D, bounded-search trace B, and integer witness F as coordinates of the same finite packet.

Use the NameCert obligations of Definition 7410.2. Bounded-search visibility keeps every inspected candidate inside the unary bound, and dyadic scale compatibility makes the rational comparisons in the trace read from Q and D rather than from a host floor oracle.

The floor inequality row then identifies the adjacent certified candidates. The witness F is exported only with the displayed inequalities $F \leq Q < F + 1$, while T, C, P, N preserve and name the same finite route. Since no excluded floor, least-number, quotient, choice, or real-completeness coordinate exists in the carrier, the Archimedean step is constructively bounded. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalFloorBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalFloorBound}^{\uparrow}$ packet is generated from a Rat input, unary Archimedean bound, bounded-search trace, Int floor witness, DyadicRatCore scaling ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalFloorBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7410.1, Definition 7410.2, and Definition 7410.3 seed the finite rational floor witness over Rat, Int, DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalFloorBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host floor function, least-number operator, quotient representative equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-search visibility, floor-inequality exactness, dyadic scale compatibility, transport, replay, provenance, local naming, and ledger nonescape for BEDC.Derived.RationalFloorBoundUp.

Chapter 7411

A Concrete Naming Certificate for $\mathsf{OscillationSeminorm}^{\uparrow}$

Following Bishop and Bridges on constructive function spaces and oscillation estimates, the $\mathsf{OscillationSeminorm}^{\uparrow}$ packet records finite seminorm data for displayed real-valued function windows. It sits after the real-name boundary of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the bounded oscillation ledger of chapter 6787, and the oscillation modulus route of chapter 2224. The packet names only the finite window, tolerance, readback, and oscillation-bound rows used to compare seminorm requests; it does not claim a host supremum over all points, quotient equality of functions, selected function representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7411.1 (Oscillation seminorm carrier)**.** *An* $\mathsf{OscillationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, O, M, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite function-window row, naming the source window, target window, and the finite observation schedule used by the seminorm request.*
2. *W is the* $\mathsf{Real}^{\uparrow}$ *window row for the displayed source and target comparisons.*
3. *O is the* $\mathsf{BoundedOscillation}^{\uparrow}$ *row recording the finite oscillation ledger attached to the same window.*
4. *M is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row converting the displayed oscillation ledger into a tolerance-to-window route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for radius, slack, and comparison budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values opened by the function-window and oscillation rows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the seminorm comparison.*
8. *B is the displayed seminorm-bound row, asserting that the finite oscillation ledger is below the carried bound under the same tolerance row D.*
9. *H records componentwise* hsame *transport for function-window, Real, bounded-oscillation, oscillation-modulus, dyadic, regular-readback, seal, bound, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a seminorm request through the finite window, oscillation ledger, modulus row, dyadic budget, regular readback, Real seal, and bound row.*
11. *P is the* Pkg *provenance over* $\mathsf{OscillationSeminorm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedOscillation}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host supremum, quotient equality of functions, selected representatives, an unrestricted function-space norm, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7411.2 (Oscillation seminorm NameCert obligations)**.** *For every accepted* $\mathsf{OscillationSeminorm}^{\uparrow}$ *carrier $S = (F, W, O, M, D, R, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OscillationSeminorm}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite function-window row, Real window row, bounded-oscillation ledger, oscillation-modulus route, dyadic tolerance row, regular rational readback, Real seal, seminorm-bound row, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7411.1. The analytic rows are F, W, O, M, D, R, E, B, and the structural rows are H, C, P, N.

The oscillation side is finite. The row O supplies the bounded oscillation ledger for the displayed window, while M records the modulus route by which a tolerance request opens only the corresponding finite windows.

The numerical side is also finite. The dyadic row D fixes the comparison budget, R supplies regular rational readback, E seals the Real-facing comparison, and B records the seminorm bound against the same rows.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no further analytic coordinate exists, the listed rows exhaust the local NameCert surface. □

Theorem 7411.3 (Oscillation seminorm RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{OscillationSeminorm}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{BoundedOscillation}^{\uparrow},\ \mathsf{OscillationModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before the seminorm-bound row is public. The handoff exports regular rational readback of the displayed oscillation data, not a host supremum, unrestricted norm, selected function representative, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the bounded oscillation and modulus rows O, M. They determine which finite window and tolerance route may be inspected for the seminorm request.

Read the dyadic budget D next. It fixes the comparison precision before any regular rational endpoint can be consumed.

The row R then supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E supplies the terminal $\mathsf{Real}^{\uparrow}$ seal. Only after those rows are displayed may the bound row B be read as seminorm data.

The structural rows H, C, P, N preserve the same route. Because the carrier lacks host suprema, quotient function equality, selected representatives, unrestricted norm data, countable choice, and ambient Real-completeness coordinates, none of those objects can be exported by the handoff. □

Theorem 7411.4 (Oscillation seminorm window nonescape)**.** *Every accepted* $\mathsf{OscillationSeminorm}^{\uparrow}$ *comparison remains inside the finite function-window, bounded-oscillation, oscillation-modulus, dyadic, regular-readback, Real-seal, and bound rows of Definition 7411.1. In particular, a consumer cannot replace the packet by an ambient function-space norm or a supremum over unlisted points.*

Proof. By carrier projection, the only window data is F, W, and the only oscillation data is O, M. Thus every comparison begins with the displayed finite window and the oscillation rows attached to it.

By Definition 7411.2, all accepted reads must then pass through D, R, E, B with the same transport, replay, provenance, and naming rows.

The excluded objects require coordinates absent from the packet. Since no carrier row stores all host points, quotient function equality, selected representatives, an unrestricted norm, countable choice, or ambient Real-completeness data, no accepted comparison can escape the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{OscillationSeminorm}^{\uparrow}$ read factors through a finite function-window row, bounded-oscillation ledger, oscillation-modulus row, DyadicRatCore tolerance row, RegSeqRat readback, Real seal, seminorm-bound row, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: bounded-oscillation, oscillation-modulus, regseqrat, dyadicratcore, real: none is bijective with $\mathsf{OscillationSeminorm}^{\uparrow}$ because this packet jointly carries function-window data, finite oscillation ledgers, modulus routing, tolerance accounting, regular readback, Real sealing, a seminorm-bound row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationSeminorm}^{\uparrow}}$.

Constructive story: An $\mathsf{OscillationSeminorm}^{\uparrow}$ packet is generated from finite function-window rows, Real window rows, BoundedOscillation rows, OscillationModulus rows, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seals, displayed seminorm bounds, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7411.1, Definition 7411.2, Definition 7411.3, and Definition 7411.4 seed the finite oscillation seminorm packet over displayed Real-facing function windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationSeminorm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host suprema, quotient function equality, selected function representatives, unrestricted function-space norms, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, bounded-oscillation ledger exactness, oscillation-modulus routing, dyadic tolerance accounting, RegSeqRat readback, Real-seal nonescape, seminorm-bound confinement, transport, replay, provenance, and local naming for BEDC.Derived.OscillationSeminormUp.

Chapter 7412

A Concrete Naming Certificate for $\mathsf{AbelMeanBoundary}^{\uparrow}$

Following Hardy's summability theory, the $\mathsf{AbelMeanBoundary}^{\uparrow}$ packet records the finite boundary at which Abel means of a displayed real series become available to Real and Completion consumers. It sits after the real-series source of chapter 1581, the Abel partial-summation route of chapter 2125, the Abel-test surface of chapter 1496, the uniform Cauchy series ledger of chapter 1736, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Abel-mean boundary; it does not import host infinite sums, quotient streams, selected radial limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7412.1 (Abel mean boundary carrier)**.** *An* $\mathsf{AbelMeanBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, A, T, U, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing finite partial sums and tail requests for the underlying real series.*
2. *A is the* $\mathsf{AbelPartialSummation}^{\uparrow}$ *row, carrying the finite summation-by-parts ledger used to control Abel means.*
3. *T is the* $\mathsf{AbelTest}^{\uparrow}$ *row, recording the bounded-partial-sum and monotone-coefficient route relevant to the Abel boundary.*
4. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *row, giving the shared tail discipline used before a boundary read reaches a real seal.*
5. *W records the inspected coefficient, radius, partial-sum, and tail windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for radial parameters, tail bounds, and endpoint comparisons.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed Abel-mean windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after S, A, T, U, W, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the source, partial-summation, Abel-test, uniform-tail, window, dyadic, readback, seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay of the route*

$$S, A, T, U, W, D \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* AbelMeanBoundary$^{\uparrow}$, RealSeries$^{\uparrow}$, AbelPartialSummation$^{\uparrow}$, AbelTest$^{\uparrow}$, UniformCauchySeries$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{AbelMeanBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, T, U, W, D, R, E, H, C, P, N$. It has no coordinate for host summability semantics, a selected radial limit, quotient equality of series names, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7412.2 (Abel mean boundary NameCert obligations)**.** *For every accepted* AbelMeanBoundary$^{\uparrow}$ *carrier*

$$M = (S, A, T, U, W, D, R, E, H, C, P, N),$$

the local NameCert$_{\text{AbelMeanBoundary}^{\uparrow}}$ *surface consists exactly of source-series admission through S; Abel partial-summation admission through A; Abel-test admission through T; uniform-tail admission through U; finite coefficient, radius, partial-sum, and tail-window exposure through W; dyadic tolerance accounting through D; regular rational readback through R; terminal Real seal ordering through E; and componentwise transport, continuation replay, provenance, and local naming through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.AbelMeanBoundaryUp.AbelMeanBoundaryTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 7412.1. The tuple displays exactly the real-series source, Abel partial-summation row, Abel-test row, uniform-tail row, finite windows, dyadic ledger, regular readback, Real seal, and structural rows.

The analytic controls are finite. Rows S, A, T, U name the partial sums, summation-by-parts ledger, Abel-test constraints, and shared tail discipline that a boundary read may inspect, while W and D record the concrete windows and tolerances.

The regular readback R repacks those same displayed rows before the terminal seal E is visible. Transport through H and replay through C preserve the same route, and P, N only source and name it. Since no further carrier coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 7412.3 (Abel mean boundary Real-seal route)**.** *Every Real or Completion consumer of an accepted* AbelMeanBoundary$^{\uparrow}$ *carrier must read the finite route*

$$\text{RealSeries}^{\uparrow},\quad \text{AbelPartialSummation}^{\uparrow},\quad \text{AbelTest}^{\uparrow},\quad \text{UniformCauchySeries}^{\uparrow},\quad W, D,\quad \text{RegSeqRat}^{\uparrow},\quad \text{Real}^{\uparrow}$$

through the same packet rows $S, A, T, U, W, D, R, E, H, C, P, N$. Thus an Abel-mean endpoint reaches the Real$^{\uparrow}$ *seal only as a displayed regular-rational handoff, not as a host infinite sum, selected radial limit, quotient stream equality, countable-choice schedule, trichotomy branch, or ambient Real-completeness theorem.*

Proof. Start with an accepted carrier and project the source rows S, A, T, U. The real-series row supplies finite partial sums, the Abel partial-summation row supplies the by-parts ledger, the Abel-test row supplies the boundary control, and the uniform Cauchy series row supplies the shared tail discipline.

Read the finite Abel-mean window through W and D. These rows fix the inspected coefficients, radius requests, partial-sum windows, tail thresholds, and dyadic tolerances that determine the regular rational readback R.

Finally pass through E, H, C, P, N. The seal row receives its analytic input only from R, and the structural rows transport, replay, source, and name that same path. Therefore a downstream Real or Completion consumer obtains only the displayed finite handoff. □

Falsifiable prediction: An accepted $\mathsf{AbelMeanBoundary}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the RealSeries source row, Abel partial-summation row, Abel-test row, uniform Cauchy series row, finite Abel-mean windows, dyadic tolerance ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: realseries, abel-partial-summation, abel-test, uniform-cauchy-series, regseqrat, real: $\mathsf{AbelMeanBoundary}^{\uparrow}$ is not bijective to any listed sibling because it binds source partial sums, summation-by-parts data, Abel-test control, one uniform tail ledger, Abel-mean windows, regular readback, and Real sealing into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelMeanBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelMeanBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7412.1, Definition 7412.2, and Definition 7412.3 seed the finite Abel-mean boundary over RealSeries partial sums, Abel partial summation, Abel-test control, uniform Cauchy series tails, RegSeqRat readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelMeanBoundary}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AbelMeanBoundaryUp`

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected radial limits, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Abel partial-summation admission, Abel-test admission, uniform-tail admission, finite-window exhaustion, regular readback, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.AbelMeanBoundaryUp.

Constructive story: An $\mathsf{AbelMeanBoundary}^{\uparrow}$ packet is generated from RealSeries source data, Abel partial-summation data, Abel-test data, uniform Cauchy series tail data, finite Abel-mean windows, dyadic tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7413

A Concrete Naming Certificate for $\mathsf{HenstockLemma}^{\uparrow}$

Following Henstock–Kurzweil and Bishop–Bridges gauge integration, the $\mathsf{HenstockLemma}^{\uparrow}$ packet records the finite gauge-partition control lemma as NameCert data. It sits after the fine-partition surface of chapter 6467 and before the gauge-integral consumer route of chapter 1999: a finite tagged partition is read together with local oscillation and mesh ledgers, then passed to regular-rational and Real-facing rows without importing host choice, open-cover compactness, quotient equality, or ambient Real completeness.

Definition 7413.1 (Henstock lemma carrier)**.** *A* $\mathsf{HenstockLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, G, F, O, M, R, E, T, C, P, N)$$

whose rows expose:

1. *I, the interval source row carrying the displayed compact or located interval domain on which the gauge partition is inspected;*
2. *G, the positive gauge row assigning a strictly positive local radius to every admitted tag;*
3. *F, the accepted* $\mathsf{FinePartitionGauge}^{\uparrow}$ *row carrying the Cousin/fine tagged partition, cell coverage, tag ledger, and fineness evidence;*
4. *O, the oscillation ledger recording the local upper and lower oscillation bounds attached to each tagged cell of F;*
5. *M, the mesh row recording finite cell widths, finite comparison bounds, and the mesh budget consumed by later sums;*
6. *R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite oscillation and mesh estimates;*
7. *E, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular rational readback;*
8. *T, the componentwise* hsame *transport row for interval, gauge, fine partition, oscillation, mesh, readback, seal, replay, provenance, and local naming data;*
9. *C, the displayed* Cont *replay row preserving the same finite gauge-partition route;*
10. *P, the* Pkg *provenance over* $\mathsf{HenstockLemma}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. N, the local $\mathsf{NameCert}_{\mathsf{HenstockLemma}^\uparrow}$ *row.*

The classifier compares accepted packets only through $I, G, F, O, M, R, E, T, C, P, N$. It has no coordinate for a choice-selected partition, a host open-cover compactness theorem, quotient equality of functions, an external Henstock integral, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 7413.2 (Henstock lemma NameCert obligations)**.** *For every accepted* $\mathsf{HenstockLemma}^\uparrow$ *carrier $H = (I, G, F, O, M, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HenstockLemma}^\uparrow}$ *surface is exhausted by interval-source admission, positive-gauge admission, Cousin/fine tagged-partition admission, oscillation-ledger exactness, mesh-row readback,* $\mathsf{RegSeqRat}^\uparrow$ *readback, terminal* $\mathsf{Real}^\uparrow$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{RiemannSumGauge}^\uparrow$ *or* $\mathsf{GaugeIntegral}^\uparrow$ *consumer must read these rows before using the finite Henstock control packet.*

Proof. Project the carrier of Definition 7413.1. The visible analytic rows are I, G, F, O, M, R, E, and the structural rows are T, C, P, N.

The partition admission obligation is the route through F. It is already a $\mathsf{FinePartitionGauge}^\uparrow$ row, so the Cousin/fine tagged partition is displayed before the oscillation or mesh ledgers can be consumed.

The estimate obligation is the route through O, M, R, E. The oscillation ledger and mesh row are finite rows attached to the same tagged cells, the regular readback row R is downstream of those estimates, and the Real seal E is visible only after R.

Transport, replay, provenance, and local naming preserve these coordinates. Since the carrier has no host choice, open-cover compactness, quotient equality, external integral, or ambient completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 7413.3 (Henstock lemma finite gauge control)**.** *Every* $\mathsf{GaugeIntegral}^\uparrow$ *consumer of an accepted* $\mathsf{HenstockLemma}^\uparrow$ *carrier factors through the finite route*

$$I, G, F, O, M \longmapsto R \longmapsto E.$$

The consumer receives interval source data, positive gauge data, a Cousin/fine tagged partition, local oscillation control, mesh readback, regular-rational readback, and a terminal Real seal; it receives no choice-selected partition, no host open-cover compactness theorem, no quotient equality of functions, and no ambient $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. Start with the fine-partition row F. It binds the displayed tagged partition to the interval source I and positive gauge G, so a consumer cannot replace the partition by a host-selected one.

Read the estimate rows next. The oscillation ledger O and mesh row M are attached cellwise to the same finite partition, and the $\mathsf{RegSeqRat}^\uparrow$ row R is the only readback route for those estimates.

Finally consume the Real seal row E and the structural rows T, C, P, N. They seal and replay the displayed finite packet without adding compactness, quotient equality, or completeness data. Hence every gauge-integral use of the Henstock lemma remains inside the route displayed above. □

Falsifiable prediction: Every accepted $\mathsf{HenstockLemma}^\uparrow$ packet must expose interval source, positive gauge, Cousin/fine tagged partition, oscillation ledger, mesh row, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows before a gauge-integral consumer reads it.

Independence witness: cousinlemma, finepartitiongauge, taggedrealpartition, riemannsumgauge, gaugeintegral, regseqrat, real: $\mathsf{HenstockLemma}^\uparrow$ is not bijective to any listed sibling because it

jointly carries local oscillation control and mesh readback over a displayed Cousin/fine tagged partition before regular-rational and Real sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockLemma}\uparrow}$.

Constructive story: A $\mathsf{HenstockLemma}^{\uparrow}$ packet is generated from an interval source, positive gauge, Cousin/fine tagged partition, oscillation ledger, mesh row, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockLemma}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7413.1, Definition 7413.2, and Definition 7413.3 seed the finite gauge-partition control route between Cousin/fine partitions and gauge-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HenstockLemma}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected partitions, host open-cover compactness, quotient equality of functions, external Henstock integrals, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, positive-gauge admission, Cousin/fine partition admission, oscillation exactness, mesh readback, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HenstockLemmaUp.

Chapter 7414

A Concrete Naming Certificate for $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$

The $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ packet realizes the cost-protocol and critical-debt rows of section 9462.13 as finite NameCert data. It sits beside the positive interpretability classifier of chapter 6202, the classifier novelty witness of chapter 6138, the discovery pipeline calculus of chapter 5914, the explanation-complexity information packet of chapter 6181, and the effective replacement ledger of chapter 568. The packet names the debt-accounting surface used by a positive discovery claim; it does not by itself certify classifier novelty, positive net information, global interpretability quality, hidden semantic truth, or a prose-only discovery verdict.

Definition 7414.1 (Positive discovery debt ledger carrier)**.** *A* $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E_0, E_1, J, W, D_c, D_l, D_s, D_v, D_i, D_g, Q, U, H, C, P, N)$$

with the following displayed rows:

1. E_0 *is the source explanation-state row before the proposed discovery transition.*
2. E_1 *is the target explanation-state row after the proposed transition.*
3. J *is the classifier-novelty witness row, naming the common source and changed classifier judgment.*
4. W *is the public cost-protocol row for benefit weights, score weights, debt weights, source scope, target scope, and not-claimed boundary.*
5. $D_c, D_l, D_s, D_v, D_i, D_g$ *are the critical-debt rows for certificate debt, ledger debt, semantic debt, verification debt, intervention debt, and generalization debt.*
6. Q *is the net-information readback row inherited from the explanation-complexity information surface under the same protocol row* W*.*
7. U *is the downstream-use ledger row recording which classifier, cost, and debt rows are actually consumed by later reports.*
8. H *records componentwise* hsame *transport for explanation states, novelty witness, cost protocol, critical-debt rows, net-information readback, downstream-use ledger, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay through classifier novelty, cost-protocol inspection, critical-debt accounting, net-information readback, downstream-use inspection, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$, $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$, $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$, $\mathsf{ExplanationComplexityInformation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden benefit weights, uncounted critical debt, classifier novelty without J, positive net information without Q, downstream use without U, or a positive-discovery verdict without the displayed protocol and debt rows.

Theorem 7414.2 (Positive discovery debt ledger NameCert obligations)**.** *For every accepted* $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ *carrier* $L = (E_0, E_1, J, W, D_c, D_l, D_s, D_v, D_i, D_g, Q, U, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}}$ *surface consists exactly of before-state admission, after-state admission, classifier-novelty witness exposure, public cost-protocol exposure, six critical-debt rows, net-information readback, downstream-use ledger exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7414.1. The before and after states are E_0, E_1, the classifier-shift evidence is J, and the public accounting convention is W.

The debt surface is the six-row family $D_c, D_l, D_s, D_v, D_i, D_g$. These rows separate certificate, ledger, semantic, verification, intervention, and generalization debt before any positive discovery claim may consume the ledger.

The net-information and downstream-use rows are Q, U. They ensure that the cost protocol and debt rows are read by the same route that later reports consume.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate exists, the listed rows exhaust the local NameCert surface. □

Theorem 7414.3 (Cost protocol critical-debt route)**.** *Every accepted* $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ *read that supports a positive discovery classification must expose the public cost-protocol row W and all six critical-debt rows $D_c, D_l, D_s, D_v, D_i, D_g$ before it may consume the net-information row Q or the downstream-use ledger U.*

Proof. Begin with the NameCert obligations of Definition 7414.2. They place W and the six debt rows in the admitted surface before replay can certify the packet.

Read Q only after that accounting surface is present. The row Q compares benefit against score and debt under the same protocol, so it cannot be detached from W or from the critical-debt rows.

Read U last. The downstream-use ledger records which displayed rows are consumed by later reports; it cannot witness use for a hidden protocol or an omitted debt row. □

Theorem 7414.4 (Debt ledger separation from classifier verdict)**.** *An accepted* $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ *packet supplies debt-accounting evidence for the positive-discovery route, but it does not itself classify the transition as* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$. *A downstream positive classifier must still read the classifier-novelty witness, positive net-information row, public cost protocol, debt rows, and not-claimed boundary through its own carrier.*

Proof. Use Definition 7414.3. The ledger packet exposes the cost protocol, critical debts, net-information readback, and downstream-use rows as an auditable accounting surface.

The positive interpretability classifier of chapter 6202 has its own carrier and acceptance route. The debt ledger here names the accounting rows that route may consume; it is not a replacement for the classifier's verdict row.

Thus the packet can support a certified-discovery route by making cost and debt public, while still refusing any positive-discovery classification that skips the classifier surface. □

Falsifiable prediction: Every accepted $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ read exposes before and after explanation states, a classifier-novelty witness, a public cost protocol, six critical-debt rows, net-information readback, downstream-use ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: positive-interpretability-classifier, classifier-novelty-witness, discovery-pipeline-calculus, explanation-complexity-information, effective-replacement-ledger: none is bijective with $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ because this packet jointly carries public cost protocol, critical-debt accounting, net-information readback, downstream-use evidence, transport, replay, provenance, and local naming as one debt ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}$ packet is generated from before and after explanation states, a classifier-novelty witness, public cost-protocol rows, certificate, ledger, semantic, verification, intervention, and generalization debt rows, net-information readback, downstream-use ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7414.1, Definition 7414.2, Definition 7414.3, and Definition 7414.4 seed the finite debt-ledger packet for positive discovery accounting.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveDiscoveryDebtLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close positive-discovery classification, global interpretability quality, hidden semantic truth, uncounted costs, undisplayed debts, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked before and after state admission, classifier-novelty exposure, public cost-protocol exposure, critical-debt row accounting, net-information readback, downstream-use ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.PositiveDiscoveryDebtLedgerUp.

Chapter 7415

A Concrete Naming Certificate for $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$

Following Jordan's bounded-variation theory and the Riemann finite-sum construction, the BoundedVariationRiemar packet records the finite partition route already used when integration-facing chapters read a variation ledger and a tagged sum in the same breath. It sits between the bounded-variation certificate of chapter 398, the tagged finite-sum certificate of chapter 1615, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the common finite partition, variation ledger, tagged summation row, dyadic observations, regular readback, Real seal, transport, replay, provenance, and local naming data; it does not assert an integral limit, a supremum over all partitions, a host function quotient, selected tags outside the displayed packet, or ambient completeness.

Definition 7415.1 (Bounded-variation Riemann-sum carrier)**.** *A* $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, \Pi, V, T, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval row over which all partition, endpoint, variation, and tagged-sum reads are made.*
2. *Π is the finite interval-partition row, carrying ordered endpoints, subinterval cells, and the common mesh ledger used by both the variation and tagged-sum faces.*
3. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger, read through the finite endpoint and edge-difference rows of chapter 398.*
4. *T is the tag-containment row, assigning a displayed tag to each cell of Π without exporting a choice-selected tag family outside the packet.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row, computing the finite tagged sum from Π, T, the displayed value reads, and the cell-width ledger of chapter 1615.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-observation row, recording endpoint differences, tag-value observations, widths, and finite variation or sum arithmetic at the dyadic level.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the displayed dyadic variation and tagged-sum data as regular rational observations.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after* I, Π, V, T, S, D, R *have been displayed.*

9. *H records componentwise* hsame *transport for the interval, partition, variation, tag, sum, dyadic, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the common partition through*

$$I, \Pi \longmapsto V, T, S, D \longmapsto R \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRiemannSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host integrability, a limiting process over all refinements, a supremum variation value, quotient equality of functions, a choice-selected tag family, an ambient measure, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7415.2 (Bounded-variation Riemann-sum NameCert obligations)**.** *For every accepted* $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$ *carrier* $J = (I, \Pi, V, T, S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRiema}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the displayed interval, partition, bounded-variation ledger, tag-containment row, finite Riemann-sum row, dyadic observation row, regular readback, Real seal, transport, replay, provenance, and local naming;*

2. *common finite-partition exposure, requiring both the* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger* V *and the* $\mathsf{RiemannSum}^{\uparrow}$ *row* S *to read the same interval row* I *and partition row* Π *before either face is consumed;*

3. *dyadic edge and tag compatibility, requiring the edge-difference observations, tag-value reads, widths, variation arithmetic, and finite-sum arithmetic in* D *to be attached to the same cells of* Π*;*

4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring* R *to be obtained only from the displayed variation and tagged-sum dyadic rows rather than from a hidden sequence or external function;*

5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring* E *to be downstream from* R *and to export no host integrability theorem, limiting process over refinements, supremum over partitions, quotient function equality, choice-selected tag family, ambient measure, or completeness principle.*

Proof. Project the carrier of Definition 7415.1. The displayed rows are exactly the interval, common partition, bounded-variation ledger, tag-containment data, finite Riemann sum, dyadic observations, regular readback, terminal Real seal, and structural rows.

The common-partition obligation is forced by the order of the carrier. The rows V and S are both downstream from I and Π, so a consumer cannot read the variation budget from one partition and the tagged sum from another hidden mesh.

The dyadic compatibility obligation is the row D. It attaches endpoint differences, tag values, widths, variation arithmetic, and finite-sum arithmetic to the same displayed cells before any regular rational readback is available.

Finally the readback and seal obligations are downstream. The row R repacks only the displayed dyadic data, and the row E is exposed only after R. Since no carrier coordinate stores integrability, refinement limits, variation suprema, quotient function equality, external tag choice, measures, or completeness, the five obligations exhaust the local certificate. □

Theorem 7415.3 (Bounded-variation Riemann-sum finite interface)**.** *Every integration-facing consumer of an accepted* $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{BoundedVariation}^{\uparrow},\ \mathsf{RiemannSum}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through one displayed interval-partition row Π*. The bounded-variation face supplies the finite endpoint and edge ledger, the Riemann-sum face supplies tags, widths, and finite summands on the same cells, and the regular rational readback and Real seal are downstream from both faces.*

Proof. Begin with the accepted carrier $J = (I, \Pi, V, T, S, D, R, E, H, C, P, N)$. The rows I and Π are displayed before the variation ledger V and before the tag-containment and finite-sum rows T, S.

Read the two analytic faces over that same partition. The bounded-variation face contributes endpoint and edge-difference rows through V, while the Riemann-sum face contributes the tag, value, width, and summand rows through T and S.

The dyadic row D then records both faces at the finite arithmetic level. Only after that common dyadic ledger has been displayed may the packet expose the regular rational readback R and the terminal Real seal E.

Thus integration-facing consumers receive a named finite interface between variation control and tagged summation. The interface contains no hidden limiting process, no ambient integrability theorem, and no external tag or partition selection principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationRiemannSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedVariationRiemannSum}^{\uparrow}$ packet is generated from a finite interval row, a common partition row, a bounded-variation ledger, a tag-containment row, a finite Riemann-sum row, dyadic edge and tag observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationRiemannSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7415.1, Definition 7415.2, and Definition 7415.3 seed the finite common-partition interface between bounded-variation ledgers and tagged Riemann sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationRiemannSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host integrability, limiting processes over all refinements, variation suprema over arbitrary partitions, quotient equality of functions, selected tag families, ambient measures, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common finite-partition exposure, dyadic edge and tag compatibility, RegSeqRat readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BoundedVariationRiemannSumUp.

Chapter 7416

A Concrete Naming Certificate for $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$

Following the finite nerve tradition for compact covers and the Bishop–Bridges proof of uniform continuity on compact metric spaces, the $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ packet records finite cover-nerve incidence together with local oscillation control and a rational lower-bound fold. It refines the compact-cover nerve surface of chapter 6263 into a consumer-facing packet for the $\mathsf{UniformModulus}^{\uparrow}$ route of chapter 9237, without passing through a host open cover, selected Lebesgue number, quotient equality of metric points, or ambient compactness theorem.

Definition 7416.1 (Compact uniform oscillation nerve carrier)**.** *A* $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, N, O, L, F, M, R, E, H, C, P, A)$$

whose rows expose:

1. K*, the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact metric carrier, finite probe spine, and located finite-net coverage;*
2. N*, the* $\mathsf{CompactCoverNerve}^{\uparrow}$ *row carrying cover-member incidence and finite overlap data;*
3. O*, the local oscillation-control ledger attached to each displayed cover member and overlap row;*
4. L*, the rational lower-bound fold row over the positive cover and oscillation radii;*
5. F*, the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row receiving the tolerance-only precision extracted by* L*;*
6. M*, the* $\mathsf{Met}^{\uparrow}$ *distance row used for source-side triangle and overlap comparisons;*
7. R*, the* $\mathsf{Real}^{\uparrow}$ *seal row for real-valued oscillation or target-distance bounds after the uniform-modulus handoff is displayed;*
8. E*, the finite incidence exactness row, recording that every oscillation read is attached to a cover member or nerve edge of* N*;*
9. H*, the componentwise* hsame *transport row for compact source, nerve, oscillation, lower-bound fold, uniform-modulus handoff, metric, Real, incidence, replay, provenance, and local naming data;*
10. C*, the displayed* Cont *replay row preserving the same finite compact-uniform route;*

11. *P, the* Pkg *provenance over* $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactCoverNerve}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *A, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationNerve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, N, O, L, F, M, R, E, H, C, P, A$. It has no coordinate for a host nerve complex, an open-cover compactness theorem, an external lower-bound selector, quotient equality of metric points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7416.2 (Compact uniform oscillation nerve NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ *carrier $U = (K, N, O, L, F, M, R, E, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationNerve}^{\uparrow}}$ *surface is exhausted by compact-metric source admission, compact-cover-nerve incidence admission, local oscillation-control exactness, rational lower-bound fold readback,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff exactness,* $\mathsf{Met}^{\uparrow}$ *distance compatibility,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, incidence exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7416.1. The compact source, cover nerve, oscillation ledger, lower-bound fold, uniform-modulus handoff, metric row, Real seal, and incidence row are exactly K, N, O, L, F, M, R, E.

The compact-cover obligation is the route through K, N, E. The compact metric source supplies the finite probe spine, the cover nerve records displayed member and overlap incidence, and incidence exactness keeps every oscillation row attached to that finite nerve.

The uniform-modulus obligation is the route through O, L, F, M, R. Local oscillation control supplies the positive candidate rows, the lower-bound fold extracts the tolerance-only precision, the metric row supplies source comparisons, and the Real seal is consumed only after the handoff is displayed.

The rows H, C, P, A transport, replay, source, and name the same packet. Since no coordinate stores a host nerve complex, open-cover compactness theorem, external lower-bound selector, quotient point equality, or ambient completeness, the listed obligations exhaust the NameCert surface. □

Theorem 7416.3 (Compact uniform oscillation nerve handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{CompactCoverNerve}^{\uparrow}, \quad O, \quad L, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The consumer receives finite cover-nerve incidence, local oscillation control, rational lower-bound fold data, metric compatibility, and a Real seal boundary, but it receives no host open cover, no selected Lebesgue number, no quotient equality of metric points, no host nerve complex, and no ambient compactness theorem.

Proof. Begin with the compact source and nerve rows. The route through K and N fixes the finite cover-member and overlap data before any oscillation estimate is read.

Next use the oscillation and lower-bound rows. The ledger O attaches each local oscillation bound to the displayed nerve incidence, and L folds the positive radius data into the precision row consumed by F.

Finally read the metric and Real-facing rows. The $\mathsf{Met}^{\uparrow}$ row M supplies source-distance compatibility for the handoff, while the $\mathsf{Real}^{\uparrow}$ seal R is downstream of the displayed uniform-modulus route. The structural rows H, C, P, A preserve the same finite packet, so none of the excluded host objects can be exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ packet must expose CompactMetric source rows, CompactCoverNerve incidence, local oscillation control, rational lower-bound fold data, UniformModulus handoff, MetricSpace distance compatibility, Real seal, transport, replay, provenance, and local NameCert rows before a compact-uniform consumer reads it.

Independence witness: compactmetric, compactcovernerve, uniformmodulus, metricspace, real: $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries cover-nerve incidence, local oscillation control, rational lower-bound fold data, metric compatibility, and uniform-modulus handoff as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationNerve}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CompactUniformOscillationNerve}^{\uparrow}$ packet is generated from CompactMetric finite-net data, CompactCoverNerve incidence, local oscillation rows, rational lower-bound folds, UniformModulus handoff, MetricSpace distance compatibility, Real sealing, incidence exactness, componentwise transport, displayed replay, provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillationNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7416.1, Definition 7416.2, and Definition 7416.3 seed the finite oscillation-nerve route from compact cover incidence to uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillationNerve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host nerve complexes, open-cover compactness, selected Lebesgue numbers, quotient equality of metric points, ambient compactness theorems, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, cover-nerve incidence admission, oscillation exactness, lower-bound fold readback, UniformModulus handoff, metric compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformOscillationNerveUp.

Chapter 7417

A Concrete Naming Certificate for $\mathsf{OstrowskiInequality}^{\uparrow}$

Following Ostrowski and Bishop–Bridges on constructive real estimates, the $\mathsf{OstrowskiInequality}^{\uparrow}$ packet records a finite error-bound certificate for a function on a displayed closed bounded interval. It sits over the dyadic rational arithmetic surface of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the continuous-map surface of chapter 130, the calculus ledger of chapter 3632, and the interval carrier of chapter 1591. The chapter names only a finite constructive error-bound route; it does not import a host real floor, trichotomy, quotient equality, selected limits, or ambient Real completeness.

Definition 7417.1 (Ostrowski inequality carrier)**.** *An* $\mathsf{OstrowskiInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, f, x, q, W, D, A, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row naming endpoints $a \leq b$ and the inspected dyadic interval windows.*
2. *f is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the interval function, read only through finite continuity witnesses on the displayed windows.*
3. *x is the inspected point row in I, represented through the interval membership and dyadic-window data.*
4. *q is the requested error precision.*
5. *W is the finite dyadic interval window containing the endpoint, midpoint, and point-readback cells used by the estimate.*
6. *D is the* $\mathsf{Calculus}^{\uparrow}$ *derivative-bound ledger. It records the finite rows witnessing the derivative bound M on the displayed interval windows.*
7. *A is the average-value row for the finite integral approximation over I.*
8. *B is the dyadic error-budget row combining the distance of x from the interval midpoint, the interval length, and the derivative-bound row D.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two regular rational names whose difference is bounded by B at precision q.*

10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed error estimate.*
11. *H records componentwise* hsame *transport for the interval, map, inspected point, precision, dyadic window, derivative-bound ledger, average row, budget row, readback row, seal, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay of the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Calculus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

13. *P is the* Pkg *provenance over* $\mathsf{OstrowskiInequality}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{OstrowskiInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $I, f, x, q, W, D, A, B, R, E, H, C, P, N$. It has no coordinate for host integration, a selected representative of a real function, quotient equality of real names, trichotomy, a host floor, or ambient completeness.

Lean4 ✓ `BEDC.Derived.OstrowskiInequalityUp.OstrowskiInequalityTasteGate single carrier alignment`

Theorem 7417.2 (Ostrowski NameCert obligations)**.** *For every accepted* $\mathsf{OstrowskiInequality}^{\uparrow}$ *carrier $O = (I, f, x, q, W, D, A, B, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OstrowskiInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval row I, continuous-map row f, inspected point row x, precision row q, finite dyadic window W, derivative-bound ledger D, average-value row A, dyadic error budget B, regular-rational readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 7417.1. The analytic data visible to the certificate are exactly the interval, map, point, precision, finite window, derivative-bound ledger, average row, dyadic budget, readback row, and terminal seal.

The interval and continuity rows are read before the calculus row. Thus the derivative-bound ledger D is attached to the displayed closed bounded interval windows rather than to an ambient differentiability principle over host reals.

The budget row B is the only place where the Ostrowski estimate is assembled. It receives the point-location and interval-length cells from W, the derivative bound from D, and the average row A before R and E are available.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no excluded object appears in the carrier, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7417.3 (Ostrowski dyadic error bound)**.** *Every accepted* $\mathsf{OstrowskiInequality}^{\uparrow}$ *packet exposes a finite dyadic inequality row*

$$|f(x) - A| \leq B_q$$

at the requested precision q, where A is the displayed average-value row, B_q is read from the dyadic budget row B, and every component of B_q is obtained from the finite interval window W and derivative-bound ledger D.

Proof. Start with the finite window W. It displays the endpoint cells, the midpoint cell, the inspected point cell, and the interval-length cells used by the Ostrowski estimate.

Read the calculus ledger D next. Its derivative-bound rows supply the finite bound used in the estimate, and the classifier admits no separate derivative source.

The dyadic budget B combines exactly those window and derivative rows with the average-value row A. The regular-rational row R then reads the difference estimate at precision q, and the terminal seal E records the resulting Real-facing bound without adding a new analytic principle. □

Theorem 7417.4 (Ostrowski Real-seal handoff)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{OstrowskiInequality}^{\uparrow}$ *packet is reachable only after the finite dyadic interval windows, derivative-bound rows, average-value row, dyadic error-budget row, and regular-rational readback have been displayed. Hence the packet exports no host real floor, no trichotomy decision, no quotient equality of real names, no selected limit, and no ambient Real completeness witness.*

Proof. Inspect the route stored by C. It lists the interval and map rows before the calculus ledger, the calculus ledger before the dyadic budget, and the dyadic budget before regular-rational readback.

The terminal seal E is downstream from R. A consumer therefore cannot reach the Real-facing estimate until the finite windows W, derivative-bound ledger D, average row A, budget row B, and readback row R have all been exposed.

The structural rows H, P, N preserve and name this same route. They do not insert host comparison, quotient equality, selected representatives, or completeness data. □

Theorem 7417.5 (Ostrowski local naming surface)**.** *The local naming row N of an accepted* $\mathsf{OstrowskiInequality}^{\uparrow}$ *packet is stable under componentwise* hsame *transport and displayed* Cont *replay: any named consumer of the estimate must preserve the same interval row, map row, inspected point, precision request, dyadic windows, derivative-bound ledger, average row, budget row, readback row, Real seal, and* Pkg *provenance.*

Proof. Use the transport row H component by component. It relates only the rows already present in the carrier and therefore cannot rename the packet through a detached real estimate.

Replay row C fixes the order in which the finite observations are consumed. Any continuation that names the estimate must follow the displayed interval, map, calculus, dyadic, regular-rational, and Real-seal route.

The package row P records the sibling provenance and the local row N names the packet over that provenance. Thus local naming is stable exactly over the finite Ostrowski surface and no larger host-real interface. □

Falsifiable prediction: An accepted $\mathsf{OstrowskiInequality}^{\uparrow}$ consumer that reads an error estimate without the displayed interval row I, map row f, inspected point x, precision q, finite dyadic window W, derivative-bound ledger D, average row A, dyadic budget B, regular-rational readback R, and terminal seal E refutes the carrier surface.

Independence witness: dyadicratcore, regseqrat, real, continuousmap, calculus, closedbounded-interval: no listed sibling supplies the whole Ostrowski packet because each sibling contributes arithmetic, readback, sealing, continuity, derivative bounds, or interval windows, while this chapter binds the finite error-budget route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OstrowskiInequality}^{\uparrow}}$.

Constructive story: An $\mathsf{OstrowskiInequality}^{\uparrow}$ packet is generated from a ClosedBoundedInterval row, a ContinuousMap row, an inspected point, a precision request, finite dyadic windows, a Calculus derivative-bound ledger, an average-value row, a DyadicRatCore error budget, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OstrowskiInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7417.1, Definition 7417.2, Definition 7417.3, Definition 7417.4, and Definition 7417.5 seed the finite constructive Ostrowski error-bound route over interval, continuity, calculus, dyadic, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{OstrowskiInequality}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host integration, host real floor, trichotomy, quotient equality of real names, selected limits, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, interval-window exactness, derivative-bound exactness, dyadic error-budget routing, RegSeqRat readback, RealUp sealing, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.OstrowskiInequalityUp.

Chapter 7418

A Concrete Naming Certificate for $\mathsf{PackingDimension}^{\uparrow}$

Following Falconer and Mattila's packing-side metric dimension, the $\mathsf{PackingDimension}^{\uparrow}$ packet records the finite disjoint-ball schedules that measure a compact metric object from the separated-cover side. It is the packing neighbour of the Hausdorff, Minkowski, and Assouad metric-dimension certificates in chapter 4119, chapter 3878, and chapter 6325. The carrier is grounded in the metric and compact metric surfaces of chapter 57 and chapter 129, the dyadic radius ledger of chapter 291, and the threshold readback of chapter 38. It names only finite packing evidence; it does not assert a host supremum over all packings, quotient equality of real names, selected limits, or ambient completeness.

Definition 7418.1 (Packing dimension carrier)**.** *A* $\mathsf{PackingDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, K, C, R, N, H, Q, S, T, U, L, O, A)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *distance surface used to test separated centers.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source for the compact metric object whose packing behaviour is being inspected.*
3. *C is the finite list of packing centers, each read through the compact metric carrier rather than through a host point set.*
4. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, with one positive dyadic radius attached to each packing window.*
5. *N records the disjointness checks: the closed metric balls centered at the displayed entries of C and using the radii in R are pairwise separated inside the metric row M.*
6. *H records the Hausdorff comparison row, naming the covering-side threshold from* $\mathsf{HausdorffDimension}^{\uparrow}$ *that is compared with the packing schedule.*
7. *Q records the Minkowski comparison row, naming the finite covering and box-counting ledger from* $\mathsf{MinkowskiDimension}^{\uparrow}$*.*
8. *S records the Assouad comparison row, naming the local scale comparison from* $\mathsf{AssouadDimension}^{\uparrow}$*.*
9. *T is the* $\mathsf{Real}^{\uparrow}$ *threshold readback for the displayed exponent bound associated to the packing schedule.*

10. *U records componentwise* hsame *transport for $M, K, C, R, N, H, Q, S, T$ and for the structural rows.*
11. *L records displayed* Cont *replay of the same finite route from metric data through disjoint packing centers, dyadic radii, comparison rows, and the real threshold.*
12. *O is the* Pkg *provenance over* $\mathsf{PackingDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{HausdorffDimension}^{\uparrow}$, $\mathsf{MinkowskiDimension}^{\uparrow}$, $\mathsf{AssouadDimension}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{PackingDimension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers by transporting these displayed metric, compact-metric, packing-center, dyadic-radius, comparison, threshold, continuation, provenance, and local naming rows to a common finite surface. It has no coordinate for a host supremum, arbitrary set-family quantification, quotient real equality, selected limits, or ambient completeness.

Theorem 7418.2 (Packing dimension NameCert obligations)**.** *For every accepted* $\mathsf{PackingDimension}^{\uparrow}$ *carrier $P = (M, K, C, R, N, H, Q, S, T, U, L, O, A)$, the local* $\mathsf{NameCert}_{\mathsf{PackingDimension}^{\uparrow}}$ *surface consists exactly of carrier admission, separated-center admission, dyadic radius positivity, disjoint ball checking, Hausdorff/Minkowski/Assouad comparison readback,* $\mathsf{Real}^{\uparrow}$ *threshold readback, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 7418.1. The rows M, K, C, R, N expose the metric source, compact finite-net data, packing centers, radius ledger, and pairwise disjointness checks.

Read the comparison rows H, Q, S next. They place the packing schedule beside the existing Hausdorff, Minkowski, and Assouad metric-dimension certificates without replacing any of those carrier surfaces.

The real-facing row is T. It reads only the displayed threshold associated to the finite packing schedule, so the route reaches $\mathsf{Real}^{\uparrow}$ after the metric and radius rows have already been replayed.

Finally U, L, O, A transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, the NameCert cannot expose host suprema, arbitrary set-family quantification, quotient real equality, selected limits, or ambient completeness. □

Theorem 7418.3 (Packing dimension scale-packing route)**.** *Every scale-facing consumer of an accepted* $\mathsf{PackingDimension}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto (C, R, N) \longmapsto (\mathsf{HausdorffDimension}^{\uparrow}, \mathsf{MinkowskiDimension}^{\uparrow}, \mathsf{AssouadDimension}^{\uparrow}) \longmapsto$$

Thus the packing-side dimension evidence is exported only after disjoint packing centers, dyadic radii, sibling dimension comparisons, and the real threshold have been displayed.

Proof. Start with the metric and compact metric rows. The metric row gives the distance test, and the compact metric row gives the finite-net source through which centers are admitted.

The packing rows C, R, N are then replayed. They list the centers, attach positive dyadic radii, and check pairwise disjointness inside the displayed metric surface.

Next the sibling comparison rows are read. The Hausdorff row supplies the covering-side threshold, the Minkowski row supplies the finite box-counting ledger, and the Assouad row supplies the local scale comparison.

Only then is the $\mathsf{Real}^{\uparrow}$ threshold row T available. The structural rows U, L, O, A preserve and name this order of dependency, so no scale-facing consumer can bypass the finite packing route through a host supremum, quotient real equality, selected limit, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PackingDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{PackingDimension}^{\uparrow}$ packet is generated from MetricUp distance tests, CompactMetricUp finite-net source data, disjoint packing centers, a positive DyadicRatCore radius ledger, Hausdorff/Minkowski/Assouad comparison rows, a RealUp threshold readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PackingDimension}^{\uparrow}}$, seed)

Scope closed: Definition 7418.1, Definition 7418.2, and Definition 7418.3 seed the packing-side metric-dimension certificate for compact metric carriers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PackingDimension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host suprema over all packings, arbitrary set-family quantification, quotient real equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-center admission, dyadic radius positivity, disjoint ball checking, sibling dimension comparison readback, Real-consumer factorization, and TasteGate faithfulness for $M, K, C, R, N, H, Q, S, T, U, L, O, A$.

Chapter 7419

A Concrete Naming Certificate for $\mathsf{FrechetCauchyNet}^{\uparrow}$

Following Frechet, Moore–Smith, Kelley, and Bishop–Bridges constructive completion practice, the $\mathsf{FrechetCauchyNet}^{\uparrow}$ packet records a directed Cauchy-net completion surface as finite BEDC NameCert data. It sits beside the finite window schedule of chapter 126, the Cauchy-filter boundary of chapter 333, the completion carrier of chapter 1506, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the directed-index, tail-cofinal, Cauchy-filter, metric-completion, regular-readback, Real-seal, transport, replay, provenance, and local certificate rows; it does not assert quotient-net equality, a selected subnet, a selected limit, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7419.1 (Frechet Cauchy net carrier)**.** *A* $\mathsf{FrechetCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, T, F, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the directed-index row, recording only the finite portion of the Moore–Smith direction inspected by the packet.*
2. *T is the tail-cofinal ledger, listing the displayed upper-bound and tail-neighborhood requests attached to I.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row obtained by reading the tail ledger as a Cauchy-filter boundary.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that receives the displayed Cauchy-filter route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to inspect the net tail at requested tolerances.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and tail budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, T, F, M, S, R have been displayed.*
8. *H records componentwise* hsame *transport for directed-index, tail-cofinal, Cauchy-filter, metric-completion, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion request through the directed tail, Cauchy-filter row, metric-completion handoff, finite windows, regular readback, and Real seal.*
10. *P records* Pkg *provenance over* $\mathsf{FrechetCauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{FrechetCauchyNet}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $I, T, F, M, S, R, E, H, C, P, N$. It has no coordinate for quotient nets, selected subnets, selected limits, countable-choice tail selection, host equality of completed objects, or ambient real completeness.

Theorem 7419.2 (Frechet Cauchy net NameCert obligations)**.** *For every accepted* $\mathsf{FrechetCauchyNet}^{\uparrow}$ *carrier*

$$G = (I, T, F, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetCauchyNet}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed directed-index row, tail-cofinal ledger, Cauchy-filter row, metric-completion handoff, StreamName window schedule, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming row. Every completion-facing obligation is read through those rows and no obligation is stored in quotient-net equality, selected subnets, selected limits, countable choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 7419.1. The tuple displays the directed-index source, tail-cofinal ledger, Cauchy-filter boundary, metric-completion handoff, window schedule, regular rational readback, terminal real seal, and structural rows.

The directed-net side is exhausted by I and T. The index row exposes only the finite directed fragment being inspected, and the tail ledger records only the cofinal tail requests attached to that fragment.

The completion side is exhausted by F, M, S, R, E. The Cauchy-filter row reads the tail ledger without quotienting nets, the metric-completion row receives that displayed boundary, the StreamName windows schedule the requested observations, the RegSeqRat row reads back the regular rational name, and the Real seal appears only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is present, the excluded quotient, subnet, selector, choice, host-equality, and ambient-completeness data cannot become NameCert obligations. □

Theorem 7419.3 (Frechet Cauchy net completion handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetCauchyNet}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route carries a directed Cauchy-net tail into the completion frontier as displayed data only. It exports no quotient net, selected subnet, selected limit, countable-choice tail selector, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7419.2. It fixes the public completion surface at the rows $I, T, F, M, S, R, E, H, C, P, N$.

Read the directed tail first. The index row I and tail-cofinal row T expose the inspected Moore–Smith direction and the finite cofinal tail data required by the completion request.

Next read the completion and real-name rows. The Cauchy-filter row F receives the tail route, M hands it to metric completion, S schedules the finite observations, R supplies the regular rational readback, and E seals the route for real-facing consumers.

The structural rows keep the handoff confined to the displayed packet. Therefore a consumer that reaches the completion or real boundary without the stated route is not reading the accepted $\mathsf{FrechetCauchyNet}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{FrechetCauchyNet}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the directed-index row, tail-cofinal ledger, $\mathsf{CauchyFilter}^{\uparrow}$ boundary, $\mathsf{MetricCompletion}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: streamname, cauchyfilter, metriccompletion, regseqrat, real: no listed sibling is bijective to $\mathsf{FrechetCauchyNet}^{\uparrow}$ because this packet binds the directed-index source, tail-cofinal ledger, Cauchy-filter boundary, metric-completion handoff, finite window schedule, regular rational readback, real seal, transport, replay, provenance, and local naming as one directed Cauchy-net completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetCauchyNet}^{\uparrow}$ packet is generated from a directed-index row, tail-cofinal ledger, CauchyFilterUp boundary, MetricCompletionUp handoff, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7419.1, Definition 7419.2, and Definition 7419.3 seed the directed Cauchy-net completion route over StreamNameUp, CauchyFilterUp, MetricCompletionUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient-net equality, selected subnets, selected limits, countable-choice tail selectors, host equality of completed points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove directed-index admission, tail-cofinal exposure, Cauchy-filter boundary routing, metric-completion handoff, window scheduling, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.FrechetCauchyNetUp.

Chapter 7420

A Concrete Naming Certificate for $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$

Following Hausdorff and Blaschke, the $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$ packet records the distance-ledger specialization of the compact-metric hyperspace: nonempty compact-subset names over one displayed compact metric source are compared by finite directed Hausdorff covers before a hyperspace consumer reads the result. It sits beside the compact metric hyperspace of chapter 5184, the scalar Hausdorff comparison surface of chapter 2086, the Vietoris hyperspace of chapter 6507, and the Blaschke selection route of chapter 2370. The packet names a finite Hausdorff metric ledger, not a host powerset metric, quotient equality of compact subsets, selected nearest points, open-cover compactness, or ambient hyperspace completion.

Definition 7420.1 (Hausdorff metric hyperspace carrier)**.** *A* $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, K, A, B, D, S, V, L, T, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the metric, finite-net, total-bounded, and complete boundary rows used by compact-subset reads.*
2. *K is the* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *row naming the nonempty compact-subset carrier over X.*
3. *A and B are the two displayed compact-subset names being compared inside K.*
4. *D is the directed Hausdorff coverage ledger, carrying one finite witness from A to B and one finite witness from B to A at every admitted tolerance.*
5. *S is the symmetric scalar readback row obtained from the two directed coverage ledgers.*
6. *V is the* $\mathsf{VietorisHyperspace}^{\uparrow}$ *compatibility row, recording that the same finite hit-and-miss windows are read when a topology-facing consumer inspects A and B.*
7. *L is the Blaschke handoff row, exposing only the distance ledger and compact source rows needed by chapter 2370.*
8. *T records componentwise* hsame *transport for $X, K, A, B, D, S, V, L, C, P, N$.*
9. *C records displayed* Cont *replay of the compact-source, compact-subset, directed-coverage, symmetric-readback, Vietoris, and Blaschke-handoff route.*

10. *P is the* Pkg *provenance over* $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$, $\mathsf{CompactMetricHyperspace}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{VietorisHyperspace}^{\uparrow}$, $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMetricHyperspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, A, B, D, S, V, L, T, C, P, N$. It has no coordinate for host powersets, quotient compacta, selected nearest points, open-cover compactness, or a hidden hyperspace completion object.

Theorem 7420.2 (Hausdorff metric hyperspace NameCert obligations)**.** *For every accepted* HausdorffMetricHypers *carrier*

$$H = (X, K, A, B, D, S, V, L, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffMetricHyperspace}^{\uparrow}}$ *surface is exhausted by compact metric source admission, compact-metric hyperspace admission, two compact-subset rows, bidirectional finite Hausdorff coverage, symmetric scalar readback, Vietoris compatibility, Blaschke handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7420.1. The source row X and hyperspace row K are the only places where compact metric and compact-subset data enter the packet.

The two compact-subset names are the rows A and B. The directed ledger D then supplies both finite coverage directions before the symmetric readback row S can be consumed.

The topology and compactness handoffs are V and L. They reuse the same directed coverage rows for Vietoris-facing and Blaschke-facing consumers rather than importing host open-cover compactness or selected representatives.

Finally T, C, P, N transport, replay, source, and name the same finite route. Since no further coordinate exists in the carrier, the listed rows exhaust the local naming surface. □

Theorem 7420.3 (Hausdorff metric hyperspace compact-set handoff)**.** *Every compact-set consumer of an accepted* $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$ *packet factors through*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactMetricHyperspace}^{\uparrow} \longmapsto \mathsf{HausdorffMetric}^{\uparrow} \longmapsto \mathsf{VietorisHyperspace}^{\uparrow}.$$

The consumer receives compact source rows, compact-subset names, two directed finite Hausdorff coverage ledgers, symmetric readback, Vietoris compatibility, Blaschke handoff, transport, replay, provenance, and local naming; it receives no host powerset, quotient compactum, selected nearest point, open-cover compactness theorem, or ambient completion witness.

Proof. Use Definition 7420.2. It forces every public read through the compact metric source X, the compact-metric hyperspace row K, and the compact-subset rows A, B.

The Hausdorff metric read is then the finite route through D and S. Both directed coverage directions are displayed before the scalar comparison can be exported.

Vietoris and Blaschke consumers receive only the rows V and L attached to the same finite ledger. The structural rows T, C, P, N preserve and name that route, so compact-set handoff cannot escape to host powersets, quotient compacta, selected nearest points, open-cover compactness, or ambient completion. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$ compact-set read that exposes symmetric distance data without first displaying $X, K, A, B, D, S, V, L, T, C, P, N$ refutes the carrier surface.

Independence witness: compact-metric-hyperspace, hausdorff-metric, vietoris-hyperspace, blaschke-selection-compact-hyperspace: the compact-metric hyperspace names the carrier without isolating the symmetric distance ledger, the Hausdorff metric chapter names directed comparison without the hyperspace handoff, the Vietoris hyperspace names hit-and-miss compatibility without the Blaschke distance ledger, and the Blaschke route consumes compact-set selection data without owning the two directed Hausdorff rows. None is bijective to $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMetricHyperspace}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMetricHyperspace}^{\uparrow}$ packet is generated from a CompactMetricUp source, a CompactMetricHyperspaceUp compact-subset carrier, two displayed compact-subset names, bidirectional finite Hausdorff coverage, symmetric readback, Vietoris compatibility, Blaschke handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMetricHyperspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7420.1, Definition 7420.2, and Definition 7420.3 seed the finite distance-ledger specialization of compact metric hyperspace data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMetricHyperspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host powersets, quotient compacta, selected nearest points, open-cover compactness, or ambient hyperspace completion.

Upgrade path: Obligation closure requires checked carrier admission, bidirectional Hausdorff coverage, symmetric readback, Vietoris compatibility, Blaschke handoff, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffMetricHyperspa

Chapter 7421

A Concrete Naming Certificate for $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ packet names the finite witness surface by which a subsequence ledger is read as a cofinal filter without importing countable choice or quotient stream equality. It sits over the displayed window rows of chapter 126, the regular rational readback of chapter 301, the finite filter surface of chapter 333, and the terminal real-name seal of chapter 38. The packet records cofinality as certificate data: every requested tail is answered by a displayed subsequence index and then transported through the same filter and Real-facing rows.

Definition 7421.1 (Cofinal subsequence-filter carrier)**.** *A* $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, R, F, J, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row for the original sequence and for every displayed subsequence read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row whose finite tails are generated from the displayed subsequence windows.*
4. *J is the cofinal-index ledger. For each finite tail request already displayed by the consumer, J records a subsequence index whose source window in S lies past that tail and whose rational readback is the row in R.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *consumer seal, reachable only after the rows S, R, F, J have been displayed.*
6. *H records componentwise* hsame *transport for the stream, regular readback, filter, cofinal-index, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, R \longmapsto J \longmapsto F \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CofinalSubsequenceFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, F, J, E, H, C, P, N$. It has no coordinate for countable choice, quotient equality of streams, a selected limit, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7421.2 (Cofinal subsequence-filter NameCert obligations)**.** *For every accepted* $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ *carrier*

$$X = (S, R, F, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CofinalSubsequenceFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, CauchyFilter tails, cofinal-index ledger, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *subsequence readback exactness, requiring every subsequence value to be read from a displayed window in S and the matching regular rational row in R;*
3. *cofinality exactness, requiring every displayed tail request to be answered by a finite index stored in J before the filter row F is consumed;*
4. *filter compatibility, requiring F to be generated only from the displayed subsequence-tail rows and not from an external filter base;*
5. *Real-seal nonescape, requiring E to be downstream of S, R, J, F and to export no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness claim;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed cofinal subsequence-filter route.*

No $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 7421.1. The accepted packet displays exactly the stream, regular readback, filter, cofinal-index, Real seal, transport, replay, provenance, and local naming rows.

Subsequence readback exactness is read from S and R. A subsequence value can only use a source window already displayed in S and the matching regular rational readback already displayed in R.

Cofinality exactness is the role of J. For each displayed tail request, J stores a finite index past that tail before the filter row F is available, so no countable-choice selector or hidden tail oracle is present.

The filter compatibility row is then read through F. Since F is downstream of S, R, J, the filter base is generated from the displayed subsequence-tail witnesses rather than from an external filter object.

Finally, E, H, C, P, N close the route. The Real seal and structural rows preserve only the displayed packet coordinates, so the excluded objects are absent from every accepted carrier and the listed obligations exhaust the local NameCert surface. □

Theorem 7421.3 (Cofinal subsequence-filter nonescape)**.** *Every consumer read of an accepted* $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ *packet factors through the finite route*

$$S, R \longmapsto J \longmapsto F \longmapsto E \longmapsto H, C, P, N.$$

The consumer receives displayed StreamName windows, RegSeqRat readback, a finite cofinal-index ledger, a CauchyFilter row generated from those witnesses, a Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data, and receives no countable choice, quotient stream equality, selected limit, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

Proof. Begin with a consumer read of an accepted packet. Projection exposes the ordered carrier rows $S, R, F, J, E, H, C, P, N$.

The read starts at the source surface. The row S displays the source and subsequence windows, and R displays the regular rational values attached to those windows.

The cofinal witness is then read from J. Each finite tail request is answered by an index already stored in the packet, so the consumer cannot replace the ledger by countable choice, quotient stream equality, or a host-side selector.

The filter row F and Real seal E are downstream of the same ledger. They can consume the displayed cofinal subsequence witnesses, but they cannot add a selected limit or an ambient completeness theorem.

The structural rows H, C, P, N transport, replay, source, and name that same route. Therefore every consumer factors through the displayed packet and receives none of the excluded escape objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalSubsequenceFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalSubsequenceFilter}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, a CauchyFilter row, a cofinal-index ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalSubsequenceFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7421.1, Definition 7421.2, and Definition 7421.3 seed the finite route from displayed subsequence windows through cofinal-index witnesses, CauchyFilter tails, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalSubsequenceFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient equality of streams, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subsequence readback exactness, cofinality exactness, filter compatibility, Real-seal nonescape, and structural stability for BEDC.Derived.CofinalSubsequenceFilterUp.

Chapter 7422

A Concrete Naming Certificate for MetaCICResidualCandidateSelector$^{\uparrow}$

The MetaCICResidualCandidateSelector$^{\uparrow}$ packet realizes the bounded residual-confluence frontier of chapter 9378 as a finite candidate-selection surface. It sits beside the residual diamond budget of chapter 7407, the candidate-normalization witness of chapter 6577, the parallel-diamond frontier of chapter 4238, and the residual envelope of chapter 3921. The packet records typed windows, residual-pair budgets, candidate-normalization evidence, local-join checking, obstruction retention, transport, replay, provenance, and local naming without claiming residual-substitution compatibility, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Definition 7422.1 (MetaCIC residual candidate selector carrier)**.** *A* MetaCICResidualCandidateSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, B, K, J, O, H, C, P, R, N)$$

with the following displayed rows:

1. *T is the typed-window row, fixing the closed MetaCIC source, local context fragment, redex shape, residual frontier, and observation depth inspected by the selector.*
2. *B is the residual-pair budget row, read through* MetaCICResidualDiamondBudget$^{\uparrow}$*, listing the finitely many residual pairs for which candidate selection is meaningful.*
3. *K is the candidate-normalization evidence row, read through* MetaCICCandidateNormalizationWitness$^{\uparrow}$*, naming the finite candidate evidence and endpoint data available to the selector.*
4. *J is the local-join checker row, recording the proposed common reducts, bounded checker status, and replay fuel for the residual pairs admitted by B.*
5. *O is the obstruction-retention row for residual-substitution compatibility, premise-free parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.*
6. *H records componentwise* hsame *transport for $T, B, K, J, O, C, P, R, N$.*
7. *C records displayed* Cont *replay from the typed window through the residual-pair budget, candidate-normalization evidence, local-join checker, and obstruction row.*

8. *P is the* Pkg *provenance over* MetaCICResidualCandidateSelector$^{\uparrow}$, MetaCICResidualDiamondBudget$^{\uparrow}$, MetaCICCandidateNormalizationWitness$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, MetaCICConfluenceResidualEnvelope BHist, hsame, Cont, *and* NameCert.
9. *R is the replay ledger that compares the selector's candidate row with the conditional handoff exposed by* MetaCICParallelDiamondFrontier$^{\uparrow}$ *and the residual envelope exposed by* MetaCICConfluenceResidualEnvelo
10. *N is the local* NameCert$_{\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, K, J, O, H, C, P, R, N$. It has no coordinate for an unbounded residual search, a premise-free diamond theorem, a global confluence proof, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Theorem 7422.2 (MetaCIC residual candidate selector NameCert obligations)**.** *For every accepted* MetaCICResidualCandidateSelector$^{\uparrow}$ *carrier*

$$S = (T, B, K, J, O, H, C, P, R, N),$$

the local NameCert$_{\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}}$ *surface consists exactly of typed-window admission, residual-pair budget exposure, candidate-normalization evidence, local-join checker status, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, replay comparison, and local naming.*

Proof. Project the carrier of Definition 7422.1. The visible rows are precisely $T, B, K, J, O, H, C, P, R, N$.

The typed window T and residual-pair budget B determine which residual branches are inspected. The candidate-normalization row K supplies the finite candidate evidence available for those branches, and the local-join checker J records only bounded checker status for the displayed budget.

The obstruction row O keeps the residual-substitution and confluence endpoints outside the packet, while H, C, P, R, N transport, replay, source, compare, and name the same rows. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7422.3 (MetaCIC residual candidate selector frontier route)**.** *Every confluence-audit consumer of an accepted* MetaCICResidualCandidateSelector$^{\uparrow}$ *packet must expose the finite route*

$$T, B, K, J, O \longmapsto H, C, P, R, N$$

before reading a candidate-mediated local-join handoff. The handoff may be consumed by MetaCICParallelDiamondFr *and* MetaCICConfluenceResidualEnvelope$^{\uparrow}$ *routes, but it does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Read T and B first. They fix the typed residual frontier and the finite residual-pair budget, so the selector is not an unbounded search over reductions.

Next read K and J. The candidate-normalization evidence names the finite candidate data available to the selector, and the local-join checker records only bounded checker status for the residual pairs in B.

Finally read O, H, C, P, R, N. The obstruction row remains visible, and the structural rows preserve the same candidate-mediated route for downstream frontier and envelope consumers. Thus the packet exports a bounded selector surface rather than a completed confluence theorem. □

Falsifiable prediction: An accepted $\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}$ packet with no residual-pair budget row B cannot export a local-join certificate, because the candidate row K and checker row J would no longer be tied to a finite residual frontier.

Independence witness: metacic_residual_diamond_budget, metacic_candidate_normalization_witness, metacic_parallel_diamond_frontier, metacic_confluence_residual_envelope: none is bijective to $\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}$ because this packet jointly carries typed-window data, residual-pair budget rows, candidate-normalization evidence, local-join checker status, obstruction retention, transport, replay comparison, provenance, and local naming.

Lean support for $\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}$ should provide Nontrivial and FieldFaithful instances in `TasteGate.lean` so the typed-window, budget, candidate, checker, obstruction, replay, provenance, and local naming rows remain distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}$ packet is generated from typed-window rows, residual-pair budgets, candidate-normalization evidence, local-join checker rows, obstruction-retention rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, replay comparison, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7422.1, Definition 7422.2, and Definition 7422.3 seed the finite residual-candidate selector over typed windows, budgets, candidates, checkers, obstruction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualCandidateSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, residual-substitution compatibility, premise-free parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Nontrivial or FieldFaithful instances, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window coverage, residual-pair budget exactness, candidate-normalization evidence, local-join checker status, obstruction retention, transport, replay comparison, provenance, local naming, Nontrivial, and FieldFaithful instances for BEDC.Derived.MetaCICResidualCandidateSelectorUp.

Chapter 7423

A Concrete Naming Certificate for $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$

The $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ packet realizes the geodesic endpoint and cross-ratio branch of chapter 9463 as finite NameCert data. It sits after the endpoint route of chapter 3491, the four-boundary comparison packet of chapter 3948, the noncrossing geodesic-lamination surface of chapter 4203, the disk phase-transport route of chapter 3455, and the metric readback of chapter 3474. The packet names a finite phase-facing endpoint comparison, not a completed hyperbolic plane, quotient ideal boundary, selected geodesic representative, host Mobius invariant, or physical observer dynamics.

Definition 7423.1 (Phase geodesic cross-ratio carrier)**.** *A* $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, E_0, E_1, E_2, E_3, L, X, \Phi, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *disk and phase-transport row.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row used to read endpoint and geodesic comparisons.*
3. *E_0, E_1, E_2, E_3 are four boundary endpoint rows, each attached to the displayed disk route rather than to a host ideal boundary.*
4. *L is the* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *compatibility row, recording that the endpoint pairs form the finite geodesic leaves consumed by lamination-facing readers.*
5. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row comparing the four endpoints through the finite cross-ratio packet.*
6. *Φ is the phase-readout row, attaching the cross-ratio comparison to the visible phase-transport ledger without treating phase equality as source equality.*
7. *R is the refusal row excluding completed hyperbolic planes, quotient ideal-boundary equality, selected geodesic representatives, host Mobius invariants, and physical observer dynamics.*
8. *H records componentwise* hsame *transport for disk, metric, endpoint, lamination, cross-ratio, phase, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from disk phase transport through metric, endpoints, lamination, cross-ratio, phase readout, and refusal.*

10. *P is the* Pkg *provenance over* $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, E_0, E_1, E_2, E_3, L, X, \Phi, R, H, C, P, N$. It has no coordinate for fewer than four endpoint rows, a detached phase scalar, a host global geometry, quotient boundary equality, or unrecorded geodesic selection.

Theorem 7423.2 (Phase geodesic cross-ratio NameCert obligations)**.** *For every accepted* $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ *carrier*

$$G = (D, M, E_0, E_1, E_2, E_3, L, X, \Phi, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}}$ *surface is exhausted by disk phase transport, hyperbolic metric readback, four displayed endpoint rows, geodesic-lamination compatibility, hyperbolic cross-ratio comparison, phase readout, explicit refusal, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7423.1. The disk and metric rows D, M fix the displayed geometry before any endpoint or phase comparison is visible.

The four endpoint rows E_0, E_1, E_2, E_3 are read next. Lamination compatibility L ensures that the endpoint pairs are geodesic-facing finite leaves, and the cross-ratio row X compares the same endpoints through chapter 3948.

The phase row Φ is downstream from those geometric rows. It records only the visible phase readout attached to the same disk transport, while R refuses host global geometry, quotient boundary equality, selected geodesics, detached Mobius invariance, and physical observer dynamics.

Finally H, C, P, N transport, replay, source, and name the same packet. Since there is no further carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7423.3 (Phase geodesic cross-ratio route)**.** *Every phase-facing consumer of an accepted* $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ *packet factors through*

$$\mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicLamination}^{\uparrow} \longmapsto \mathsf{HyperbolicC}$$

The consumer receives disk phase transport, hyperbolic metric readback, four endpoint rows, lamination compatibility, cross-ratio comparison, phase readout, refusal, transport, replay, provenance, and local naming; it receives no completed hyperbolic plane, quotient ideal boundary, selected geodesic representative, host Mobius invariant, or physical observer-dynamics row.

Proof. Use Definition 7423.2. It places the disk transport and metric rows before the four endpoint rows can be inspected.

The endpoint rows then feed the lamination and cross-ratio rows L and X. The lamination row keeps the geodesic leaves finite and displayed, while the cross-ratio row keeps the four-boundary comparison inside the existing hyperbolic cross-ratio packet.

Only after those rows are visible may Φ expose a phase readout. The refusal row R and structural rows H, C, P, N keep that readout attached to the displayed route, so the phase-facing consumer cannot escape to host global geometry, quotient boundary equality, selected geodesics, detached Mobius invariance, or physical observer dynamics. □

Falsifiable prediction: An accepted $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ phase read that omits disk transport, hyperbolic metric readback, four endpoint rows, lamination compatibility, cross-ratio comparison, refusal, transport, replay, provenance, or local NameCert rows refutes the carrier surface.

Independence witness: hyperbolic-geodesic-boundary, hyperbolic-geodesic-lamination, hyperbolic-cross-ratio, poincare-disk-phase-transport, hyperbolic-metric: the geodesic-boundary packet names endpoints without phase cross-ratio readout, the lamination packet names noncrossing leaves without the four-point phase row, the cross-ratio packet compares endpoints without owning the phase transport, the disk phase-transport packet names reversible phase surfaces without the geodesic endpoint comparison, and the metric packet supplies distance readback without the endpoint-phase carrier. None is bijective to $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}}$.

Constructive story: A $\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}$ packet is generated from PoincareDiskPhaseTransportUp rows, HyperbolicMetricUp readback, four HyperbolicGeodesicBoundaryUp endpoint rows, HyperbolicGeodesicLaminationUp compatibility, HyperbolicCrossRatioUp comparison, a phase readout row, explicit refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7423.1, Definition 7423.2, and Definition 7423.3 seed the finite phase-facing route from geodesic endpoints to cross-ratio readout.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhaseGeodesicCrossRatio}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed hyperbolic planes, quotient ideal-boundary equality, selected geodesic representatives, host Mobius invariants, or physical observer dynamics.

Upgrade path: Obligation closure requires checked carrier admission, endpoint display, lamination compatibility, cross-ratio comparison, phase-readout nonescape, refusal, transport, replay, provenance, and local naming for BEDC.Derived.PhaseGeodesicCrossRat

Chapter 7424

A Concrete Naming Certificate for $\mathsf{StoneSpace}^{\uparrow}$

Following Stone, Birkhoff, and Kelley, the $\mathsf{StoneSpace}^{\uparrow}$ packet records the finite BEDC carrier by which a Boolean algebra is read as a topological clopen ledger. It sits beside the Boolean algebra carrier of chapter 3733, the clopen algebra carrier of chapter 7214, the Stone-duality bridge of chapter 261, the topology carrier of chapter 88, and the local compact Hausdorff surface of chapter 4029. The packet names the Stone-space observation surface available to later clopen-ledger and bridge work; it does not assert ultrafilter choice, host powersets, quotient topology equality, or a checked standard bridge.

Definition 7424.1 (Stone space carrier)**.** *A* $\mathsf{StoneSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, L, T, K, Z, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BooleanAlgebra}^{\uparrow}$ *source row. It records the displayed zero, one, meet, join, complement, order, and distributivity observations that generate the Stone-space interface.*
2. *L is the* $\mathsf{ClopenAlgebra}^{\uparrow}$ *ledger. It records the clopen rows named by the Boolean observations and keeps their finite Boolean operations visible as clopen operations.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *public-open row. It exposes the indexed open rows generated by the clopen ledger, including the displayed empty, whole, finite-meet, complement-open, and finite-union reads.*
4. *K is the compact Hausdorff readback row. It records only the displayed* $\mathsf{CompactHausdorff}^{\uparrow}$ *separation and compactness obligations available from the finite clopen ledger, not an external point-set compactification theorem.*
5. *Z is the zero-dimensional readback row. It says that the public topology rows read by T are generated through clopen rows from L, so the topology-facing basis remains the displayed clopen ledger.*
6. *H records componentwise* hsame *transport for the Boolean source, clopen ledger, topology rows, compact Hausdorff readback, zero-dimensional readback, replay, provenance, and local name.*
7. *C records displayed* Cont *replay for the route*

$$B \longmapsto L \longmapsto T, K, Z.$$

8. *P is the* Pkg *provenance over* $\mathsf{StoneSpace}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{ClopenAlgebra}^{\uparrow}$, $\mathsf{StoneDuality}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactHausdorff}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, T, K, Z, H, C, P, N$. It has no coordinate for selected ultrafilters, host powersets, quotient topology equality, arbitrary opens outside the displayed $\mathsf{Topology}^{\uparrow}$ *rows, or bridge checking.*

Theorem 7424.2 (Stone space NameCert obligations)**.** *For the carrier of Definition 7424.1, the local* $\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}$ *surface consists exactly of Boolean-algebra source exposure through B, clopen-ledger exposure through L, topology public-open exposure through T, compact Hausdorff and zero-dimensional readback through K and Z, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every Stone-duality or clopen-ledger consumer reads the Stone space only through those rows and receives no ultrafilter choice, host powerset, quotient topology equality, arbitrary-open completion, or checked bridge.*

Proof. Carrier admission is projection from Definition 7424.1. The packet displays exactly the Boolean source, clopen ledger, topology public rows, compact Hausdorff readback, zero-dimensional readback, transport, replay, provenance, and local naming row.

The topology surface is generated before any point-space readback is considered. The row L turns Boolean observations into clopen observations, and T exposes only the indexed open rows generated from that clopen ledger.

The compact Hausdorff and zero-dimensional rows are readbacks from the same displayed route. They can certify the finite separation, compactness, and clopen-basis obligations visible in K and Z, but they do not choose ultrafilters or equate an external quotient topology with the internal classifier.

Finally H, C, P, N preserve and name the same route. Since no field of the carrier stores host powersets, selected ultrafilters, arbitrary opens, quotient topology equality, or a standard bridge witness, no consumer can extract such data from the local $\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}$ surface. □

Theorem 7424.3 (Stone space clopen-ledger boundary)**.** *Every accepted* $\mathsf{StoneSpace}^{\uparrow}$ *packet factors its topology-facing observations through the finite route*

$$\mathsf{BooleanAlgebra}^{\uparrow} \longrightarrow \mathsf{ClopenAlgebra}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{CompactHausdorff}^{\uparrow} \longrightarrow H, C, P, N.$$

The route is compatible with the $\mathsf{StoneDuality}^{\uparrow}$ *clopen classifier of chapter 261: a Boolean row may be replayed as a clopen topology row and then read back as compact Hausdorff and zero-dimensional structure. The boundary gives later bridge work a concrete* $\mathsf{StoneSpace}^{\uparrow}$ *target while excluding ultrafilter selection, host powerset extensionality, quotient topology equality, and bridge-checked equivalence.*

Proof. Start with the Boolean source row B. The clopen ledger L admits only the finite clopen observations generated from the Boolean operations, so each topology-facing read has a displayed clopen origin.

The topology handoff is T. It records the public opens generated by the clopen rows, while the compact Hausdorff and zero-dimensional readbacks K and Z inspect only the same public rows and the same clopen generator provenance.

Compatibility with $\mathsf{StoneDuality}^{\uparrow}$ is therefore a ledger comparison, not a point-space construction. The Stone-duality classifier already sends Boolean observations to clopen topology rows, and the present carrier exposes that route as the named $\mathsf{StoneSpace}^{\uparrow}$ packet.

Transport, replay, provenance, and local naming keep the route finite. Since every accepted read factors through the displayed chain, the excluded ultrafilter, powerset, quotient-equality, and bridge-checking objects remain outside the carrier boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{StoneSpace}^{\uparrow}$ packet is generated from BooleanAlgebra source rows, a ClopenAlgebra ledger, Topology public-open rows, CompactHausdorff and zero-dimensional readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7424.1, Definition 7424.2, and Definition 7424.3 seed the finite Stone-space carrier over BooleanAlgebraUp, ClopenAlgebraUp, StoneDualityUp, TopologyUp, CompactHausdorffUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilter choice, host powersets, quotient topology equality, arbitrary-open completion, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, clopen-ledger exactness, topology public-row exposure, compact Hausdorff and zero-dimensional readback, transport, replay, provenance, and local naming for BEDC.Derived.StoneSpaceUp.

Chapter 7425

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, the $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$ packet records the finite seal between a selected cofinal subsequence window and the Real-facing regular read used by L10 completion consumers. It sits after the bounded selector surface of chapter 1408, the regular Cauchy subsequence limit packet of chapter 1862, the broader subsequence limit boundary of chapter 1467, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert boundary for subsequence limits; it does not choose a subsequence by host choice, quotient a stream, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7425.1 (Cauchy subsequence limit seal carrier)**.** *A* $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence window, carrying the inspected regular Cauchy source values before any subsequence read is formed.*
2. *I is the cofinal subsequence index row, recording monotonicity, cofinal reach, and the finite selected indices used by the route.*
3. *D is the dyadic tolerance data, including the requested precision and the finite comparison ledger that justifies the selected tail window.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and selected subsequence observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and dyadic tolerance data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the selected window and regular rational readback rows have been displayed.*
7. *H records componentwise* hsame *transport for source windows, cofinal index rows, dyadic tolerance data, stream windows, regular readbacks, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the same finite subsequence-limit route.*
9. *P records the* Pkg *provenance over* $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$, $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$, $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$, $\mathsf{SubsequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, I, D, W, R, E, H, C, P, N$. It has no coordinate for a host-selected infinite subsequence, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7425.2 (Cauchy subsequence limit seal NameCert obligations)**.** *For every accepted* $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$ *carrier $L = (S, I, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the source sequence window S, cofinal subsequence index row I, dyadic tolerance data D,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, package provenance P, and local naming N.*

Proof. Project the accepted carrier of Definition 7425.1. Each row named in the statement is a displayed coordinate of the finite packet, and the carrier lists no hidden coordinate for a selected infinite subsequence, quotient stream, or ambient completion theorem.

The source and index obligations are S and I. The dyadic tolerance and finite-window obligations are D and W, and the Real-facing route reaches R before it reaches the terminal seal E.

Finally use the structural rows. Componentwise transport H, replay C, provenance P, and local naming N preserve the same route for the selector, regular-subsequence-limit, subsequence-limit, StreamName, RegSeqRat, DyadicRatCore, and Real siblings named above. □

Theorem 7425.3 (Cauchy subsequence limit seal nonescape)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$ *packet must read the route*

$$S, I, D, W, R, E, H, C, P, N$$

before it consumes any subsequence-limit seal. The route exports no host-selected infinite subsequence, quotient stream equality, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7425.2. They force every consumer to display the source window, cofinal index row, dyadic tolerance data, finite StreamName windows, regular rational readback, and Real seal as one carrier-local route.

The sibling handoffs preserve the same order. The selector boundary of chapter 1408, the regular Cauchy subsequence limit packet of chapter 1862, and the subsequence limit boundary of chapter 1467 can feed the row I or the readback row R, but none supplies a hidden quotient stream or host choice coordinate.

Since the classifier has only $S, I, D, W, R, E, H, C, P, N$, any completion-facing read that asks for host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness asks for data outside the packet. The Real seal is therefore a terminal finite seal for the displayed subsequence route, not an escape hatch into host completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}$ packet is generated from source sequence windows, cofinal subsequence index rows, dyadic tolerance data, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}}$, seed)

Scope closed: Definition 7425.1, Definition 7425.2, and Definition 7425.3 seed the finite subsequence-limit seal between selected cofinal windows and Real-facing regular reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySubsequenceLimitSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove host subsequence choice, quotient stream equality, countable choice, host real equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal index exposure, dyadic tolerance exactness, StreamName window coverage, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySubsequenceLimitSealUp.

Chapter 7426

A Concrete Naming Certificate for $\mathsf{FrechetCauchyFilter}^{\uparrow}$

Following Bourbaki and Kelley on filter convergence and Bishop–Bridges constructive analysis, the $\mathsf{FrechetCauchyFilter}^{\uparrow}$ packet records the L10 route where cofinite sequence tails are read as a displayed Cauchy-filter completion surface. It sits beside the cofinite-tail filter of chapter 1564, the Cauchy-filter boundary of chapter 333, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite tail-to-filter certificate only; it does not quotient filters, select limits, invoke countable choice, use host equality of completed objects, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7426.1 (Frechet Cauchy filter carrier)**.** *A* $\mathsf{FrechetCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite tail-threshold row, carrying the displayed finite complement and the tail request through which a sequence is inspected.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *meet row, recording the common-tail refinement required before two tail requests are read as one filter event.*
3. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row receiving the displayed cofinite-tail filter route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that lists the sequence coordinates inspected by tail and Cauchy requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the displayed tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Frechet tail, FilterBase meet, Cauchy-filter handoff, stream windows, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for $T, B, Q, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a completion request through the cofinite-tail threshold, FilterBase meet, Cauchy-filter handoff, finite windows, regular readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetCauchyFilter}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, B, Q, W, R, E, H, C, P, N$. It has no coordinate for quotient filter equality, selected limits, countable-choice tail selection, host completion equality, or ambient real completeness.

Theorem 7426.2 (Frechet Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{FrechetCauchyFilter}^{\uparrow}$ *carrier*

$$F = (T, B, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the Frechet tail threshold, FilterBase meet, Cauchy-filter handoff, StreamName finite windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming row.*

Proof. Project the carrier of Definition 7426.1. The visible rows are precisely $T, B, Q, W, R, E, H, C, P, N$.

The tail side is exhausted by T and B. The Frechet row exposes the finite complement and tail request, while the FilterBase row records the meet that makes several tail requests a single displayed filter event.

The completion-facing side is exhausted by Q, W, R, E. The Cauchy-filter row receives the displayed tail route, the StreamName row lists the inspected coordinates, the RegSeqRat row reads those windows as regular rational data, and the Real seal appears only after the readback has been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no other coordinate is present, quotient filter equality, selected limits, countable choice, host completion equality, and ambient real completeness cannot become local NameCert obligations. □

Theorem 7426.3 (Frechet Cauchy filter completion handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetCauchyFilter}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff carries cofinite sequence tails into Cauchy-filter completion consumers as displayed data only. It exports no quotient filter, selected limit, countable-choice tail selector, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7426.2. It fixes the public completion surface at the rows $T, B, Q, W, R, E, H, C, P, N$.

Read the Frechet and FilterBase rows first. The cofinite-tail threshold row T supplies the finite tail request, and the meet row B displays the common-tail refinement required before the request is treated as a filter event.

Next read the Cauchy-filter and real-name rows. The handoff row Q receives the displayed filter route, W schedules the finite observations, R supplies regular rational readback, and E seals the route for real-facing consumers.

The structural rows keep the handoff confined to the displayed packet. Therefore a consumer that reaches a Cauchy-filter, regular-rational, or real boundary without the stated route is not reading the accepted $\mathsf{FrechetCauchyFilter}^{\uparrow}$ carrier. □

Theorem 7426.4 (Frechet Cauchy filter scope ledger)**.** *For every accepted* $\mathsf{FrechetCauchyFilter}^{\uparrow}$ *carrier*

$$F = (T, B, Q, W, R, E, H, C, P, N),$$

finite induction over the displayed carrier rows exposes, in order, the cofinite-tail threshold, FilterBase meet, Cauchy-filter handoff, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

Every $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of* F *must read that prefix order; no consumer can skip an earlier row and still consume the accepted* $\mathsf{FrechetCauchyFilter}^{\uparrow}$ *packet.*

Proof. Project the carrier tuple from Definition 7426.1. The tuple has exactly the finite row list $T, B, Q, W, R, E, H, C, P, N$, so the base read starts at the cofinite-tail threshold row T and no hidden completion coordinate is available.

Induct along the displayed list. After T has exposed the finite tail request, the row B records the FilterBase meet; after B, the handoff row Q may receive the displayed filter route; after Q, the finite window row W, regular readback row R, and Real seal row E are read in that order.

Use Definition 7426.2 for row exactness. It identifies H, C, P, N as transport, replay, provenance, and local naming for the same finite route, not as alternative entrances into the completion surface.

Finally apply Definition 7426.3. Any downstream Cauchy-filter, regular-rational, or real-facing consumer receives only the displayed route from $\mathsf{FrechetFilter}^{\uparrow}$ through $\mathsf{Real}^{\uparrow}$, so skipping an earlier row would leave the consumer outside the accepted carrier. □

Falsifiable prediction: Every accepted $\mathsf{FrechetCauchyFilter}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the $\mathsf{FrechetFilter}^{\uparrow}$ cofinite-tail threshold, $\mathsf{FilterBase}^{\uparrow}$ meet, $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: frechetfilter, cauchyfilter, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{FrechetCauchyFilter}^{\uparrow}$ because this packet binds the cofinite-tail threshold, FilterBase meet, Cauchy-filter handoff, finite window schedule, regular rational readback, real seal, transport, replay, provenance, and local naming as one L10 completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetCauchyFilter}^{\uparrow}$ packet is generated from a FrechetFilterUp cofinite-tail threshold, FilterBaseUp meet, CauchyFilterUp handoff, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 7426.1, Definition 7426.2, Definition 7426.3, and Definition 7426.4 seed the L10 Frechet Cauchy filter route over FrechetFilterUp, FilterBaseUp, CauchyFilterUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable-choice tail selectors, host equality of completed points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove Frechet-tail admission, FilterBase meet exposure, Cauchy-filter handoff, window scheduling, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.FrechetCauchyFilterUp.

Chapter 7427

A Concrete Naming Certificate for $\mathsf{ModulusCauchyReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ModulusCauchyReal}^{\uparrow}$ packet records a regular Cauchy real read whose convergence evidence is indexed by a displayed modulus ledger rather than by quotient classes of Cauchy sequences. It sits after the dyadic rational core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the choice-free completion route of chapter 4237. The packet names the finite route from modulus-indexed rational requests to the $\mathsf{Real}^{\uparrow}$ seal without adding a selected limit, quotient representative, host real equality, or countable-choice schedule.

Definition 7427.1 (Modulus-Cauchy real carrier)**.** *A* $\mathsf{ModulusCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus ledger, listing the inspected precision requests and the finite convergence bounds admitted for those requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite stream windows read at the bounds named by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic radii, slack data, and comparisons used by the modulus-indexed read.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus ledger, stream windows, regular readback, dyadic tolerance, real seal, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from a modulus-indexed request through M, S, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, S, R, D, E, H, C, P, N$. It has no coordinate for quotient Cauchy classes, selected real limits, host equality of real numbers, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7427.2 (Modulus-Cauchy real NameCert obligations)**.** *For every accepted* $\mathsf{ModulusCauchyReal}^{\uparrow}$ *carrier*

$$K = (M, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusCauchyReal}^{\uparrow}}$ *surface consists exactly of five obligation rows: modulus-ledger exposure through M, finite-window and regular-readback exposure through S, R, dyadic tolerance exactness through D, terminal real sealing through E, and structural preservation through H, C, P, N.*

Proof. Project the carrier of Definition 7427.1. Rows M, S, R, D, E are the analytic route, while H, C, P, N transport, replay, source, and name that route.

The modulus ledger is read first. It supplies the precision requests and finite bounds, so the stream windows in S and the regular rational readback in R are inspected only along displayed modulus-indexed data.

The dyadic tolerance row D fixes the comparison budget before the terminal row E can be consumed as a $\mathsf{Real}^{\uparrow}$ seal. Since the carrier has no quotient, selected-limit, host-equality, trichotomy, choice, or ambient-completeness coordinate, the five listed rows exhaust the local NameCert surface. □

Theorem 7427.3 (Modulus-Cauchy real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{ModulusCauchyReal}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the displayed modulus ledger before the terminal real seal is consumed. Equivalently, the read follows the finite route*

$$M,\ S,\ R,\ D,\ E,\ H,\ C,\ P,\ N$$

from modulus-indexed precision data to the $\mathsf{Real}^{\uparrow}$ *seal.*

The route exports no quotient representative, selected limit, host real equality, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 7427.2. They place every real-facing read downstream of the modulus ledger, the finite stream windows, the regular rational readback, and the dyadic tolerance row.

Use the sibling anchors chapter 126, chapter 301, and chapter 291 for the finite observation route. The terminal seal is the $\mathsf{Real}^{\uparrow}$ row of chapter 38, and the surrounding completion discipline is the choice-free route of chapter 4237.

Finally H, C, P, N preserve the same displayed route by transport, replay, provenance, and local naming. Thus every real-facing consumer receives the modulus-indexed finite packet before the seal and cannot obtain any excluded host-level completion data from it. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCauchyReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7427.1, Definition 7427.2, and Definition 7427.3 seed the modulus-indexed regular Cauchy route from finite stream windows and dyadic tolerances to the terminal real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusCauchyReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy completions, selected real limits, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, modulus coverage, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusCauchyRealUp.

Constructive story: A ModulusCauchyReal$^{\uparrow}$ packet is generated from a displayed modulus ledger, finite StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerance rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7428

A Concrete Naming Certificate for $\mathsf{BornologicalSpace}^{\uparrow}$

Following Hogbe-Nlend and Bourbaki on bounded structures in functional analysis, the $\mathsf{BornologicalSpace}^{\uparrow}$ packet records a finite bounded-family carrier rather than an ambient collection of all bounded subsets. It sits after the bounded-set packet of chapter 1757, the locally convex family surface of chapter 3386, the seminorm family route of chapter 5868, and the normed-space comparison row of chapter 1950. The packet names the rows needed by bounded-map and functional-analysis consumers; it does not assert a host bornology, all subset closure, quotient completion, selected absorbing disk, or classical duality.

Definition 7428.1 (BornologicalSpace carrier)**.** *A* $\mathsf{BornologicalSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, \mathcal{B}, L, S, V, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying carrier row whose admissible points and finite classifier reads are the only source data available to the packet.*
2. *$\mathcal{B}$ is the finite bounded-set family row, listing the displayed* $\mathsf{BoundedSet}^{\uparrow}$ *certificates accepted as bounded disks or bounded subsets.*
3. *L is the locally convex control row imported from* $\mathsf{LocallyConvexSpace}^{\uparrow}$*, exposing only the finite neighborhood and seminorm-family data used to test boundedness.*
4. *S is the seminorm control row imported from* $\mathsf{SeminormedSpace}^{\uparrow}$*, recording finite seminorm budgets for every displayed bounded-family read.*
5. *V is the normed-space comparison row imported from* $\mathsf{NormedSpace}^{\uparrow}$ *when a single norm comparison is part of the carrier.*
6. *M is the bounded-map route row, saying that a consumer may read bounded-map data only after the source bounded-family row and the target control row are displayed.*
7. *H records componentwise* hsame *transport for the carrier, bounded-family, locally convex, seminorm, normed comparison, bounded-map, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through $X, \mathcal{B}, L, S, V, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BornologicalSpace}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{LocallyConvexSpace}^{\uparrow}$*,* $\mathsf{SeminormedSpace}^{\uparrow}$*,* $\mathsf{NormedSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BornologicalSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host bornology, an unbounded subset universe, a selected disk basis, quotient completion, or classical duality.

Theorem 7428.2 (BornologicalSpace NameCert obligations)**.** *For every accepted* $\mathsf{BornologicalSpace}^{\uparrow}$ *carrier* $B = (X, \mathcal{B}, L, S, V, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BornologicalSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, finite bounded-family admission, locally convex control, seminorm control, normed-space comparison when present, bounded-map route exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7428.1. The analytic rows are the source carrier X, the bounded-family row $\mathcal{B}$, the locally convex row L, the seminorm row S, the normed comparison row V, and the bounded-map route M.

The bounded-family obligation is finite. Each boundedness read names a displayed $\mathsf{BoundedSet}^{\uparrow}$ certificate before any locally convex, seminorm, or normed-space consumer can use it.

The control obligations are rowwise. The row L supplies locally convex neighborhood data, S supplies finite seminorm budgets, and V supplies the optional single-norm comparison without replacing the finite bornological carrier by a host topology.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores a host bornology, unbounded subset universe, selected disk basis, quotient completion, or duality theorem, the listed rows exhaust the NameCert surface. □

Theorem 7428.3 (BornologicalSpace bounded-map route)**.** *Every bounded-map consumer of an accepted* $\mathsf{BornologicalSpace}^{\uparrow}$ *carrier must first read*

$$X, \mathcal{B}, L, S, V, M, H, C, P, N.$$

Thus the consumer receives finite bounded-set family rows together with locally convex, seminorm, and normed comparison control before the bounded-map row is exported. It receives no host bornology, all-subsets closure principle, selected disk basis, quotient completion, or classical duality object.

Proof. Start with a bounded-map request. By carrier admission, the request cannot reach M before the source carrier X and the bounded-family row $\mathcal{B}$ are displayed.

Use the control rows next. Locally convex data in L, seminorm budgets in S, and the normed comparison row V describe exactly how bounded family reads are tested by downstream functional-analysis packets.

The row M is therefore a bounded-map route rather than an independent host theorem. Componentwise transport and replay through H and C preserve the same finite path, while P, N only record provenance and local naming.

Because the carrier contains no coordinate for the excluded host objects, the displayed route is the entire bounded-map surface. □

Falsifiable prediction: A consumer that claims a $\mathsf{BornologicalSpace}^{\uparrow}$ bounded-map read while omitting the finite bounded-family row, locally convex control, seminorm control, normed comparison when present, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local $\mathsf{NameCert}$ row refutes the carrier of Definition 7428.1.

Independence witness: boundedset, locally-convex-space, seminormed-space, normedspace: no listed sibling is bijective to $\mathsf{BornologicalSpace}^{\uparrow}$ because this packet jointly binds finite bounded-family rows, locally convex control, seminorm budgets, normed comparison, bounded-map routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BornologicalSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BornologicalSpace}^{\uparrow}$ packet is generated from carrier rows, finite bounded-family rows, locally convex control rows, seminorm control rows, normed-space comparison rows, bounded-map route rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BornologicalSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7428.1, Definition 7428.2, and Definition 7428.3 seed the finite bornological bounded-family packet for functional-analysis consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BornologicalSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host bornologies, all-subsets closure, selected disk bases, quotient completion, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-family admission, locally convex control, seminorm control, normed comparison, bounded-map routing, transport, replay, provenance, and local naming for BEDC.Derived.BornologicalSpaceUp.

Chapter 7429

A Concrete Naming Certificate for $\mathsf{CompactOscillationMesh}^{\uparrow}$

Following Bishop–Bridges compact metric uniform continuity, the $\mathsf{CompactOscillationMesh}^{\uparrow}$ packet realizes the compact-uniform-continuity route of chapter 9237 as a finite oscillation mesh. It sits after the compact metric source of chapter 129, the displayed continuous-map rows of chapter 130, the uniform-modulus consumer of chapter 369, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite mesh of compact probes and oscillation bounds; it is not an open-cover compactness theorem, selected Lebesgue number, host modulus function, quotient equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7429.1 (Compact oscillation mesh carrier)**.** *A* $\mathsf{CompactOscillationMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, Q, M, D, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, exposing centers, mesh radius, and coverage witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise inspection row attached to the displayed compact source.*
3. *Q is the finite oscillation ledger, recording for each mesh cell the source-radius and target-oscillation bound read from F.*
4. *M is the rational lower-bound mesh fold, extracting a positive source precision from the finite oscillation ledger.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-conversion row used by downstream dyadic budgets.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the folded mesh precision.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for consumers that read the modulus through regular rational names.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, F, Q, M, D, U, R are displayed.*
9. *H records componentwise* hsame *transport for compact source, continuous-map, oscillation, fold, dyadic, uniform-modulus, readback, Real, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay through $K, F, Q, M, D, U, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationMesh}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for selected covers, selected Lebesgue numbers, host moduli, quotient map equality, countable choice, or ambient compactness.

Theorem 7429.2 (Compact oscillation mesh NameCert obligations)**.** *For every accepted* $\mathsf{CompactOscillationMesh}^{\uparrow}$ *carrier* $O = (K, F, Q, M, D, U, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationMesh}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, continuous-map pointwise inspection, finite oscillation-ledger admission, rational lower-bound mesh folding, dyadic tolerance conversion,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7429.1. The compact source row K, continuous-map row F, oscillation ledger Q, mesh fold M, dyadic row D, uniform-modulus row U, readback row R, and Real seal E are the analytic coordinates.

The compact and continuity obligations are finite. The mesh centers and coverage witnesses come from K, while each oscillation row in Q is attached to an inspected pointwise row in F.

The modulus obligation is the route through Q, M, D, U. The fold M extracts one positive source precision from the finite oscillation ledger, D converts it into the dyadic tolerance budget, and U exports the tolerance-only uniform-modulus row.

Finally R, E, H, C, P, N read, seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert surface. □

Theorem 7429.3 (Compact oscillation mesh uniform-modulus route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationMesh}^{\uparrow}$ *carrier must first read*

$$K, F, Q, M, D, U, H, C, P, N.$$

If the consumer later reaches $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *data, the read factors through the same mesh route before R and E are available. Thus the compact-uniform-continuity output is a finite oscillation-mesh realization, not a selected Lebesgue number, host modulus, open-cover compactness theorem, quotient map equality, or ambient completeness principle.*

Proof. Begin with the compact source row. The finite mesh data in K fixes the centers, mesh radius, and coverage witnesses before any oscillation estimate can be consumed.

Use the continuous-map inspection row F and the oscillation ledger Q. Each target-oscillation bound is attached to a displayed mesh cell, so the consumer cannot replace the ledger by a host continuity theorem.

The lower-bound fold M and dyadic row D supply the source precision consumed by U. Componentwise transport and replay through H and C preserve exactly that route, and P, N only source and name it.

When regular rational or Real-facing consumers appear, R and E are downstream of the same rows. No selected cover, selected Lebesgue number, host modulus, quotient equality, countable choice, or ambient completeness coordinate is present in the carrier. □

Falsifiable prediction: A consumer that claims a $\mathsf{CompactOscillationMesh}^{\uparrow}$ uniform-modulus read while omitting compact finite-net rows, continuous-map pointwise inspection, finite oscillation ledger, rational mesh fold, dyadic tolerance conversion, $\mathsf{UniformModulus}^{\uparrow}$ handoff, transport, replay, provenance, or local NameCert rows refutes the carrier of Definition 7429.1.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{CompactOscillationMesh}^{\uparrow}$ because this packet jointly binds compact finite-net data, pointwise oscillation rows, finite mesh folding, dyadic tolerance conversion, uniform-modulus handoff, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationMesh}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise inspection rows, finite oscillation ledger rows, rational mesh folds, DyadicRatCore tolerance conversion, UniformModulus handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOscillationMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7429.1, Definition 7429.2, and Definition 7429.3 seed the finite oscillation-mesh route from compact metric pointwise inspection to uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOscillationMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, host modulus functions, quotient map equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, continuous-map pointwise inspection, oscillation-ledger admission, rational mesh fold exactness, dyadic tolerance conversion, UniformModulus handoff, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationMeshUp.

Chapter 7430

A Concrete Naming Certificate for $\mathsf{DetachableBar}^{\uparrow}$

Following Bishop and Troelstra–van Dalen on bars over spreads and choice sequences, the $\mathsf{DetachableBar}^{\uparrow}$ packet records a finite-prefix bar predicate whose inspected prefixes are decidable inside the displayed BEDC rows. It sits beside the finite fan surface of chapter 400, the barred-spread induction route of chapter 1675, the choice-sequence prefix rows of chapter 1482, the Baire-space cylinder surface of chapter 334, and the spread-law admission route of chapter 1532. Downstream analytic consumers reach it only through the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38; they receive no countable-choice selector, quotient stream equality, open-cover compactness, selected branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7430.1 (DetachableBar carrier)**.** *A* $\mathsf{DetachableBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, I, Q, B, A, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row or finite fan-boundary row through which bounded prefix inspection is requested;*
2. *I is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *barred-spread row, exposing only the finite bar decisions and induction replay visible at the inspected prefixes;*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix-answer row, recording the finite answers whose prefixes are being tested;*
4. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row that names the ambient countable-prefix window without naming a completed branch;*
5. *A is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row, deciding which displayed prefixes are legal spread extensions;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by downstream analytic consumers;*
7. *W is the detachable bar-decision row, recording for each displayed prefix whether the bar predicate accepts it;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after the finite window schedule and bar-decision row have been displayed;*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for real-facing consumers that pass through S, W, R;*
10. *H records componentwise* hsame *transport for the fan, bar-induction, choice-sequence, Baire, spread-law, StreamName, bar-decision, regular-rational, and Real rows;*
11. *C records displayed* Cont *replay of the same finite-prefix route;*
12. *P is the* Pkg *provenance over* $\mathsf{DetachableBar}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DetachableBar}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed finite-prefix rows. It has no coordinate for a host function on completed streams, a selected infinite branch, countable choice, quotient stream equality, an open-cover subcover, or ambient real completeness.

Theorem 7430.2 (DetachableBar NameCert obligations)**.** *For every accepted* $\mathsf{DetachableBar}^{\uparrow}$ *carrier*

$$D = (F, I, Q, B, A, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DetachableBar}^{\uparrow}}$ *surface consists exactly of carrier admission for those rows, fan-boundary exposure through F, barred-spread replay through I, choice-sequence prefix exposure through Q, Baire-cylinder exposure through B, spread-law admission through A, finite StreamName scheduling through S, detachable bar-decision exposure through W, regular-rational readback through R, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7430.1. The projection lists exactly the fan, bar-induction, choice-sequence, Baire, spread-law, finite-window, bar-decision, readback, Real-seal, transport, replay, provenance, and naming rows.

The bar predicate is detachable because W is a displayed decision row over the finite prefixes already exposed by F, I, Q, B, A, S. A consumer cannot ask about an unseen tail or a completed branch before those rows have been displayed.

The analytic handoff is downstream. Rows R and E may be read only after the finite StreamName schedule and bar-decision row have been exposed, and the structural rows H, C, P, N preserve the same route without adding a choice principle or quotient equality. □

Theorem 7430.3 (DetachableBar finite-prefix decision)**.** *Every decision exported by an accepted* $\mathsf{DetachableBar}^{\uparrow}$ *packet is attached to a displayed finite prefix admitted through*

$$\mathsf{FanTheorem}^{\uparrow},\ \mathsf{BrouwerBarInduction}^{\uparrow},\ \mathsf{ChoiceSequence}^{\uparrow},\ \mathsf{BaireSpace}^{\uparrow},\ \mathsf{SpreadLaw}^{\uparrow},\ \mathsf{StreamName}^{\uparrow}.$$

If two accepted packets agree componentwise on those prefix rows and on their detachable bar-decision rows under the displayed hsame *transport, then they expose the same accepted-or-refused bar decision to every downstream consumer.*

Proof. Begin with the prefix-facing rows F, I, Q, B, A, S. They determine the finite branch window, barred-spread ancestry, choice-sequence answers, Baire cylinder, spread-law admission, and StreamName schedule available to the packet.

The detachable decision row W is indexed only by that displayed window. Componentwise hsame transport therefore identifies the visible prefix and the visible decision together, so no consumer can distinguish transported packets by asking about an unlisted tail.

Replay through C and provenance through P, N preserve the same finite decision surface. Thus the exported classifier is a finite-prefix decision, not an extensional equality test on completed streams. □

Theorem 7430.4 (DetachableBar real-completion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{DetachableBar}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DetachableBar}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

together with the sibling fan, bar-induction, choice-sequence, Baire, and spread-law rows. The handoff exports finite bar evidence for real-name readback, not countable choice, open-cover compactness, quotient stream equality, selected branch data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the NameCert obligations of Definition 7430.2. They force the StreamName window S and detachable decision row W to be displayed before R or E is available.

The regular-rational row R reads the same finite window, and the Real row E seals only that readback. The fan, bar-induction, choice-sequence, Baire, and spread-law rows remain prerequisites for the prefix to which W attaches.

Since no carrier coordinate stores a selected branch, open-cover subcover, quotient stream equality, countable-choice functional, or ambient completeness theorem, the analytic consumer receives only the displayed finite handoff. □

Theorem 7430.5 (DetachableBar prefix transport)**.** *Let* $D = (F, I, Q, B, A, S, W, R, E, H, C, P, N)$ *and* $D' = (F', I', Q', B', A', S', W', R', E', H', C', P', N')$ *be accepted* $\mathsf{DetachableBar}^{\uparrow}$ *packets. If the fan, barred-spread, choice-sequence, Baire, spread-law, StreamName, bar-decision, regular-rational, and Real rows transport componentwise under* H, H' *and replay through* C, C'*, then every consumer sees the same detachable-bar prefix route from* D *and* D'*.*

Proof. The componentwise transport identifies the prefix-facing rows F, I, Q, B, A, S with F', I', Q', B', A', S'. Hence both packets expose the same finite prefix window and the same legal spread or Baire admission surface.

The same transport identifies W with W' over that window. Consequently the accepted-or-refused bar decision is transported together with its prefix, not detached from it as a host predicate on completed streams.

Finally R, E, P, N and R', E', P', N' are replayed through C, C'. Any real-facing or completion-facing consumer therefore reads the same transported finite-prefix route. □

Theorem 7430.6 (DetachableBar obligation package)**.** *The preceding carrier, NameCert, finite-prefix decision, real-completion handoff, and prefix-transport clauses assemble the seed obligation package for* $\mathsf{DetachableBar}^{\uparrow}$*. The package is exhausted by the finite rows*

$$F, I, Q, B, A, S, W, R, E, H, C, P, N$$

and by the sibling route through $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{BrouwerBarInduction}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{SpreadLaw}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. The carrier definition supplies exactly the listed rows, and Definition 7430.2 makes them the local NameCert surface.

Definition 7430.3 binds the detachable decision to displayed finite prefixes. Definition 7430.4 places analytic readback after that decision, and Definition 7430.5 shows that componentwise transport preserves the whole route.

Thus the package names the finite-prefix bar predicate needed by downstream Real and Completion work while refusing countable choice, open-cover compactness, quotient stream equality, selected branch data, and ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DetachableBar}^{\uparrow}}$.

Constructive story: A $\mathsf{DetachableBar}^{\uparrow}$ packet is generated from FanTheoremUp finite fan rows, BrouwerBarInductionUp barred-spread rows, ChoiceSequenceUp prefix answers, BaireSpaceUp prefix cylinders, SpreadLawUp admission, StreamNameUp finite windows, detachable bar-decision rows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DetachableBar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7430.1, Definition 7430.2, Definition 7430.3, Definition 7430.4, Definition 7430.5, and Definition 7430.6 seed the finite-prefix detachable-bar route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DetachableBar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, open-cover compactness, quotient stream equality, selected branch data, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean carrier and checked obligation package for BEDC.Derived.DetachableBarUp.

Chapter 7431

A Concrete Naming Certificate for $\mathsf{CourantFischerMinimax}^{\uparrow}$

Following Courant and Fischer on the finite-dimensional min-max principle, the $\mathsf{CourantFischerMinimax}^{\uparrow}$ packet names the finite spectral min-max certificate without importing a host eigenspace selector. It consumes the matrix carrier of chapter 49, the inner-product rows of chapter 86, the Hermitian surface of chapter 6403, the normal-operator routing of chapter 5807, and the finite gap window of chapter 5136.

Definition 7431.1 (CourantFischerMinimax carrier)**.** *A* $\mathsf{CourantFischerMinimax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, O, I, W, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite* $\mathsf{Mat}^{\uparrow}$ *row, recording the coordinate table used by the spectral read.*
2. *O is the* $\mathsf{NormalOperator}^{\uparrow}$ *or* $\mathsf{HermitianMatrix}^{\uparrow}$ *operator row attached to the same matrix table.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row through which vector lengths and orthogonality tests are read.*
4. *W is the finite subspace-window row, listing the inspected dimensions, inclusions, and complement windows.*
5. *R is the Rayleigh quotient ledger, recording numerator and denominator reads through M, O, I, W.*
6. *L is the min-max comparison row, displaying the finite lower-window and upper-window comparisons for the same dimension index.*
7. *E is the extremal ledger, recording which displayed subspace windows witness the lower and upper sides of the comparison.*
8. *H records componentwise* hsame *transport for the matrix, operator, inner-product, subspace-window, Rayleigh, comparison, extremal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a spectral min-max request through $M, O, I, W, R, L, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CourantFischerMinimax}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{HermitianMatrix}^{\uparrow}$, $\mathsf{NormalOperator}^{\uparrow}$, $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CourantFischerMinimax}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host eigenvector choice, a spectral-completeness theorem, an ambient quotient of subspaces, selected extrema outside E, or a hidden diagonalization procedure.

Theorem 7431.2 (CourantFischerMinimax NameCert obligations)**.** *For every accepted* $\mathsf{CourantFischerMinimax}^{\uparrow}$ *carrier*

$$K = (M, O, I, W, R, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CourantFischerMinimax}^{\uparrow}}$ *surface consists exactly of carrier admission, Rayleigh quotient readback, finite subspace-window stability, min-max ledger exactness, sibling transport through* $\mathsf{Mat}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{HermitianMatrix}^{\uparrow}$, $\mathsf{NormalOperator}^{\uparrow}$, *and* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$, *displayed* Cont *replay,* Pkg *provenance, and local naming. A spectral min-max read is admitted only after the same matrix, operator, inner-product, subspace-window, Rayleigh, comparison, and extremal rows are visible in the packet.*

Proof. Project the carrier of Definition 7431.1. The matrix row M, operator row O, and inner-product row I supply the finite algebraic surface on which Rayleigh reads can be formed.

The subspace-window row W fixes the finite inspection domain. The Rayleigh ledger R is read only through M, O, I, W, so every quotient claim remains tied to the displayed matrix and inner-product coordinates.

The min-max comparison row L records the lower-window and upper-window comparisons for the same finite dimension index. The extremal ledger E names the displayed windows that witness the two sides, without selecting host eigenvectors.

Finally, componentwise transport by H preserves the sibling rows, while C, P, N replay, source, and name the same packet. Since the carrier has no field for spectral completeness, diagonalization, quotient subspace equality, or external extrema, these rows exhaust the local NameCert obligations. □

Lean4 (statement) `BEDC.Derived.CourantFischerMinimaxUp`

Closure status for $\mathsf{NameCert}_{\mathsf{CourantFischerMinimax}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CourantFischerMinimax}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7431.1 and Definition 7431.2 seed the finite spectral min-max certificate through displayed matrix, operator, inner-product, subspace-window, Rayleigh, comparison, and extremal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CourantFischerMinimax}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host eigenvector choice, spectral completeness, quotient subspace equality, selected extrema outside the packet, or ambient diagonalization.

Upgrade path: Obligation closure requires checked carrier admission, Rayleigh readback, finite subspace-window stability, min-max ledger exactness, sibling transport, replay, provenance, and non-escape for BEDC.Derived.CourantFischerMinimaxUp.

Constructive story: A $\mathsf{CourantFischerMinimax}^{\uparrow}$ packet is generated from a finite matrix row, a Hermitian or normal-operator row, an inner-product row, a finite subspace-window row, a Rayleigh quotient ledger, a min-max comparison row, an extremal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7432

A Concrete Naming Certificate for $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ packet records the finite L10 seal by which two interleaved regular Cauchy reads are accepted as the same real-facing completion route. It sits after the finite windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, the comparison criterion of chapter 3405, and the binary projection ledger of chapter 3864. The packet names the displayed interleaving seal only; it does not quotient streams, select limits, invoke countable choice, compare host real equality, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7432.1 (Real Cauchy interleaving seal carrier)**.** *A* $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W_0, W_1, B, I, Q, R_0, R_1, D, E, H, C, P, N)$$

with the following displayed rows:

1. W_0 *and* W_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules inspected by the equality or completion request.*
2. B *is the* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *projection ledger that records how even and odd windows are recovered from one interleaved read.*
3. I *is the* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *comparison row, carrying the finite criterion that licenses the two projected reads to share one tolerance discipline.*
4. Q *is the dyadic tolerance row used by the comparison, the two projected reads, and the final seal.*
5. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the two projected regular rational names after* W_0, W_1, B, I, Q *are displayed.*
6. D *is the diagonal tolerance row showing that the readbacks meet the same requested dyadic bound.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the finite windows, interleaving ledger, comparison row, dyadic tolerance row, readbacks, and diagonal row have been consumed.*
8. H *records componentwise* hsame *transport for* $W_0, W_1, B, I, Q, R_0, R_1, D, E, C, P, N$.

9. *C records displayed* Cont *replay from an equality or completion request through the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$, $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyInterleavingSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W_0, W_1, B, I, Q, R_0, R_1, D, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7432.2 (Real Cauchy interleaving seal NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ *carrier*

$$S = (W_0, W_1, B, I, Q, R_0, R_1, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyInterleavingSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the two StreamName windows, binary interleaving projection ledger, Cauchy interleaving criterion row, dyadic tolerance row, two RegSeqRat readbacks, diagonal tolerance row, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming row.*

Proof. Project the carrier of Definition 7432.1. The finite windows W_0, W_1 are the only observation schedules available to an equality or completion request.

Read the interleaving and comparison rows next. The binary ledger B supplies the even–odd projection surface, while I and Q attach the Cauchy interleaving criterion to a displayed dyadic tolerance row.

The two regular rational readbacks are then R_0 and R_1. The diagonal row D records that the two readbacks meet the same requested tolerance before the terminal $\mathsf{Real}^{\uparrow}$ seal E is exposed.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier lists no quotient stream, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate, those objects are not part of the local NameCert surface. □

Theorem 7432.3 (Real Cauchy interleaving seal regular readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *equality or completion consumer of an accepted* $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ *packet must read the route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegularCauchyBinaryInterleaving}^{\uparrow},\ \mathsf{CauchyInterleavingCriterion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through $W_0, W_1, B, I, Q, R_0, R_1, D, E, H, C, P, N$ before it consumes the terminal seal. The route is a finite regular-readback seal, not a quotient of streams, selected limit, countable-choice construction, host real equality, or ambient completeness theorem.

Proof. Start with Definition 7432.2. It makes the StreamName windows, binary projection ledger, criterion row, tolerance row, readbacks, diagonal row, and seal the whole analytic surface of the packet.

Use the sibling routes in order. The StreamName rows expose the finite windows, the RegularCauchyBinaryInterleaving row supplies the even–odd projection ledger, and the CauchyInterleavingCriterion row fixes the comparison before either regular rational readback is consumed.

The readback step is carried by R_0, R_1 under the shared dyadic tolerance row Q. The diagonal row D is the last finite check before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

Transport, replay, provenance, and local naming preserve this same route. Thus a Real or Completion consumer that bypasses any of the listed rows is asking for data outside the accepted seal packet. □

Falsifiable prediction: An accepted $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ consumer is refuted if it reaches the $\mathsf{Real}^{\uparrow}$ seal without first reading the two $\mathsf{StreamName}^{\uparrow}$ finite-window rows, the $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ projection ledger, the $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ comparison row, dyadic tolerance data, both $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, diagonal tolerance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, regseqrat, real, cauchy-interleaving-criterion, regular-cauchy-binary-interleaving: none is bijective with $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ because this packet jointly binds two finite window schedules, one binary interleaving ledger, one criterion row, one dyadic tolerance row, two regular rational readbacks, a diagonal tolerance row, a terminal Real seal, transport, replay, provenance, and local naming as one completion-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyInterleavingSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyInterleavingSeal}^{\uparrow}$ packet is generated from two StreamName finite-window rows, a RegularCauchyBinaryInterleaving projection ledger, a CauchyInterleavingCriterion row, dyadic tolerance data, two RegSeqRat readbacks, a diagonal tolerance row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyInterleavingSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7432.1, Definition 7432.2, and Definition 7432.3 seed the finite Real Cauchy interleaving seal over StreamNameUp, RegularCauchyBinaryInterleavingUp, CauchyInterleavingCriterionUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyInterleavingSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable-choice schedules, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove window admission, binary projection exposure, criterion routing, dyadic tolerance exactness, regular readback, diagonal tolerance, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyInterleavingSealUp.

Chapter 7433

A Concrete Naming Certificate for KuratowskiClosureComplement$^{\uparrow}$

Following Kuratowski's closure-complement theorem, the KuratowskiClosureComplement$^{\uparrow}$ packet records the finite alternation surface generated by a closure row and a complement row before any host powerset equality is read. It is grounded in the topology carrier of chapter 88, the closed-set boundary of chapter 4229, the closure operator packet of chapter 2145, and the two-point specialization witness of chapter 1860. The chapter names the closure/complement alternation route as BEDC certificate data only; it does not assert a host set algebra, quotient equality of subsets, Boolean powerset extensionality, or a completed lattice of all opens.

Definition 7433.1 (Kuratowski closure-complement carrier)**.** *A* KuratowskiClosureComplement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, L, C, A, B, H, R, P, N)$$

with the following visible rows:

1. *T is the ambient* Topology$^{\uparrow}$ *carrier whose finite open, closed, and specialization rows provide the inspection context.*
2. *F is the* ClosedSet$^{\uparrow}$ *boundary row naming the closed-stable subsets allowed in the certificate.*
3. *L is the* KuratowskiClosure$^{\uparrow}$ *closure row, including extensivity, monotonicity, and idempotence reads already displayed by the closure packet.*
4. *C is the complement row, recording only the formal exchange between a named subset row and its displayed complement row inside the finite topology packet.*
5. *A is the alternation ledger for words in the two symbols closure and complement, with every accepted word reduced through the displayed finite route.*
6. *B is the fourteen-boundary row, recording that no accepted alternation consumer can distinguish more than the finite Kuratowski boundary packets generated by L and C.*
7. *H is componentwise* hsame *transport for the topology, closed-set, closure, complement, alternation, and boundary rows.*
8. *R is displayed* Cont *replay for reading T, F, L, C, A, B, H in that order.*
9. *P is* Pkg *provenance over* KuratowskiClosureComplement$^{\uparrow}$, Topology$^{\uparrow}$, ClosedSet$^{\uparrow}$, KuratowskiClosure$^{\uparrow}$, SierpinskiSpace$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{KuratowskiClosureComplement}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by the finite alternation ledger and its displayed transports. It has no coordinate for host powerset equality, quotient subset equality, Boolean algebra completion, arbitrary choice of representatives, or a full topology of all opens.

Lean4 (definition) `BEDC.Derived.KuratowskiClosureComplementUp.KuratowskiClosureComplementCarrier`

Theorem 7433.2 (Kuratowski closure-complement NameCert obligations)**.** *For an accepted* KuratowskiClosureCom *carrier $K = (T, F, L, C, A, B, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiClosureComplement}^{\uparrow}}$ *surface consists exactly of carrier admission, closure-row exposure, complement-row exposure, finite alternation replay, fourteen-boundary retention, componentwise transport, provenance, and non-escape from host powerset principles.*

Proof. Project the carrier of Definition 7433.1. The visible rows are the topology context, closed-set boundary, closure operator row, complement row, alternation ledger, fourteen-boundary row, transport, replay, provenance, and local name.

The closure obligation is read from L together with the topology context T and the closed-set boundary F. It exposes only the finite closure operator behavior already admitted by the Kuratowski closure packet.

The complement and alternation obligations are read from C and A. Each alternation word is replayed through the displayed closure/complement rows, so the certificate tracks a finite route rather than an ambient subset function space.

Finally, B, H, R, P, N record the boundary, transport, replay, source, and naming rows. Since no row contains host powerset equality, quotient subset equality, Boolean algebra completion, representative choice, or an all-opens carrier, none of those principles is exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.KuratowskiClosureComplementUp.KuratowskiClosureComplementCarrier namecert obligations`

Theorem 7433.3 (Kuratowski closure-complement fourteen route)**.** *Every accepted* KuratowskiClosureComplement *consumer reading a closure/complement word factors through the finite route*

$$\mathsf{Topology}^{\uparrow} \longmapsto \mathsf{ClosedSet}^{\uparrow} \longmapsto \mathsf{KuratowskiClosure}^{\uparrow} \longmapsto C \longmapsto A \longmapsto B \longmapsto (H, R, P, N).$$

The route records the traditional Kuratowski fourteen-set boundary as finite NameCert data, not as host-level equality of all subsets or as an ambient Boolean powerset algebra.

Proof. Begin with the ambient topology and closed-set rows. They provide the only admissible inspection context for subset, closedness, and specialization reads.

Read the closure row next. The $\mathsf{KuratowskiClosure}^{\uparrow}$ packet supplies the closure behavior used in each alternation step, and the complement row supplies the other generator of the word ledger.

The alternation ledger A records the finite word route, while B stores the displayed fourteen-boundary retention. Transport and replay through H and R keep the same finite rows under componentwise comparison, and P, N source and name the packet.

Thus the consumer receives only finite closure/complement alternation evidence. The read cannot turn into host powerset equality, quotient subset equality, Boolean completion, arbitrary representative choice, or a topology-of-all-opens principle because no such row appears in the carrier. □

Lean4 ✓ `BEDC.Derived.KuratowskiClosureComplementUp.KuratowskiClosureComplementCarrier fourteen route`

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiClosureComplement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiClosureComplement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7433.1, Definition 7433.2, and Definition 7433.3 seed the finite closure/complement alternation packet and its fourteen-boundary route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiClosureComplement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, host powerset equality, quotient subset equality, Boolean algebra completion, representative choice, or a full topology of all opens.

Upgrade path: Obligation closure requires checked carrier admission, closure-row exposure, complement-row exposure, alternation replay, fourteen-boundary retention, transport, provenance, and non-escape for BEDC.Derived.KuratowskiClosureComplement

Constructive story: A $\mathsf{KuratowskiClosureComplement}^{\uparrow}$ packet is generated from finite topology, closed-set, Kuratowski-closure, complement, alternation-ledger, fourteen-boundary, transport, replay, provenance, and local NameCert rows.

Chapter 7434

A Concrete Naming Certificate for $\mathsf{FinitePhaseCylinder}^{\uparrow}$

The $\mathsf{FinitePhaseCylinder}^{\uparrow}$ packet realizes the phase-surface and Poincare disk readings of chapter 9463 as finite NameCert data. It consumes the complex coordinate surface of chapter 39, the boundary-circle carrier of chapter 58, the disk row of chapter 377, the Poincare metric route of chapter 6209, and the reversible transport row of chapter 219.

Definition 7434.1 (FinitePhaseCylinder carrier)**.** *A* $\mathsf{FinitePhaseCylinder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Z, S, D, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *Z is the finite* $\mathsf{Complex}^{\uparrow}$ *phase-coordinate row used for local readout.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *boundary-circle row naming the phase boundary.*
3. *D is a* $\mathsf{UnitDisk}^{\uparrow}$ *or* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk row that records the displayed interior and boundary read.*
4. *U is the* $\mathsf{UnitaryGroup}^{\uparrow}$ *reversible-transport row, read only as a classifier-preserving finite transport.*
5. *W is the finite phase-window ledger, listing the inspected phase samples, boundary tags, and disk-position reads.*
6. *R is the refusal row excluding a global universe digest, a global clock, an unlogged Hilbert-space operator, and an interior point named* T*.*
7. *H records componentwise* hsame *transport for the complex phase, circle, disk, transport, window, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a phase-cylinder request through $Z, S, D, U, W, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FinitePhaseCylinder}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{PoincareDiskMetric}^{\uparrow}$*,* $\mathsf{UnitaryGroup}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePhaseCylinder}^{\uparrow}}$ *row.*

The future TasteGate surface for this packet must expose $\mathsf{Nontrivial}$ *and* $\mathsf{FieldFaithful}$ *instances for the displayed phase and field-facing readback rows before downstream consumers use the cylinder as more than a finite representation packet.*

Theorem 7434.2 (FinitePhaseCylinder NameCert obligations)**.** *For every accepted* $\mathsf{FinitePhaseCylinder}^{\uparrow}$ *carrier*

$$F = (Z, S, D, U, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitePhaseCylinder}^{\uparrow}}$ *surface consists exactly of carrier admission, complex phase readout, boundary-circle exposure, disk or Poincare-metric readback, reversible transport, finite phase-window stability, refusal-row preservation, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A phase-cylinder read may consume* $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, *and* $\mathsf{UnitaryGroup}^{\uparrow}$ *only through the displayed rows of the same packet.*

Proof. Project the carrier of Definition 7434.1. The phase coordinate Z, boundary circle S, disk row D, and reversible transport U give the only geometric rows exposed to a consumer.

The finite window W fixes which phase samples, boundary tags, and disk positions are readable. The refusal row R travels with the same window, so no consumer can reinterpret the cylinder as a global digest or an unlogged ambient operator.

Transport by H is componentwise across $Z, S, D, U, W, R, C, P, N$. Replay through C and provenance through P, N preserve the same finite packet, exhausting the local NameCert obligations. □

Lean4 (statement) `BEDC.Derived.FinitePhaseCylinderUp`

Falsifiable prediction: Every accepted $\mathsf{FinitePhaseCylinder}^{\uparrow}$ consumer must display ComplexUp phase coordinates, S1Up boundary tags, a UnitDiskUp or PoincareDiskMetricUp disk read, a UnitaryGroupUp reversible-transport row, a finite phase-window ledger, refusal rows, hsame transport, Cont replay, Pkg provenance, and local NameCert before using a phase-cylinder read.

Independence witness: complex, s1, unitdisk, poincare-disk-metric, unitarygroup: $\mathsf{FinitePhaseCylinder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries phase readout, boundary circle, disk metric, reversible transport, finite window, and refusal rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePhaseCylinder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePhaseCylinder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7434.1 and Definition 7434.2 seed the finite phase-cylinder packet through displayed phase, boundary, disk, reversible-transport, window, refusal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePhaseCylinder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, a global universe digest, a global clock, a hidden Hilbert-space operator, field-faithful TasteGate instances, or an interior point named T.

Upgrade path: Obligation closure requires checked carrier admission, phase readout, boundary-circle exposure, disk readback, reversible transport, refusal preservation, replay, provenance, and local naming for BEDC.Derived.FinitePhaseCylinderUp.

Constructive story: A $\mathsf{FinitePhaseCylinder}^{\uparrow}$ packet is generated from a ComplexUp phase row, an SOneUp boundary-circle row, a UnitDiskUp or PoincareDiskMetricUp disk row, a UnitaryGroupUp reversible-transport row, a finite phase-window ledger, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7435

A Concrete Naming Certificate for $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$

Following Hausdorff and Bishop–Bridges finite-cover analysis, the $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ packet records the displayed bridge where Hausdorff gauge covers are read as finite cover-refinement data before any covering-dimension consumer inspects them. It sits between the gauge-cover packet of chapter 4119, the finite-refinement surface of chapter 1853, the metric observation route of chapter 57, the compact source of chapter 129, and the dyadic scale ledger of chapter 291. The packet names a finite gauge-to-refinement handoff only; it does not assert Hausdorff measure existence, exact dimension values, open-cover compactness, countable cover choice, selected optimal covers, host logarithms, or ambient Euclidean embeddings.

Definition 7435.1 (Hausdorff dimension finite-cover carrier)**.** *A* $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, K, D, G, C, R, O, W, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *observation row used to read ball centers, radii, and diameter comparisons.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite-net data before any cover member is admitted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, recording the finite dyadic radius and tolerance ledger used by the gauge.*
4. *G is the inherited* $\mathsf{HausdorffDimension}^{\uparrow}$ *gauge row, including the positive scale and rational weight expression.*
5. *C is the finite visible cover row, listing cover members as* BHist *entries with center, radius, membership, and diameter data.*
6. *R is the refinement map row, assigning each covering-dimension cover member to a displayed Hausdorff cover member.*
7. *O is the overlap-order row consumed by* $\mathsf{CoveringDimension}^{\uparrow}$ *when the finite cover is read as a refinement surface.*
8. *W is the gauge-weight and finite-sum row attached to the same listed cover members.*

9. *H records componentwise* hsame *transport for $M, K, D, G, C, R, O, W, T, P, N$.*
10. *T records displayed* Cont *replay from the Hausdorff gauge request through the finite cover, refinement, overlap, and weight rows.*
11. *P is the* Pkg *provenance over* HausdorffDimensionFiniteCover$^{\uparrow}$, HausdorffDimension$^{\uparrow}$, CoveringDimension$^{\uparrow}$, Met$^{\uparrow}$, CompactMetric$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{HausdorffDimensionFiniteCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, D, G, C, R, O, W, H, T, P, N$. It has no coordinate for outer measures, countable cover families, selected optimal covers, host logarithmic dimension calculations, open-cover compactness, or quotient compacta.

Theorem 7435.2 (Hausdorff dimension finite-cover NameCert obligations)**.** *For every accepted* HausdorffDimensionFiniteCover$^{\uparrow}$ *carrier*

$$F = (M, K, D, G, C, R, O, W, H, T, P, N),$$

the local NameCert$_{\text{HausdorffDimensionFiniteCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric observation row, compact metric source row, dyadic scale ledger, Hausdorff gauge row, finite visible cover row, refinement map, overlap-order row, gauge-weight finite sum, componentwise transport, displayed replay, provenance, and local naming. No obligation is stored outside $M, K, D, G, C, R, O, W, H, T, P, N$.*

Proof. Project the carrier of Definition 7435.1. The source and scale side is exhausted by M, K, D, G: metric observations, compact finite-net data, dyadic radii, and the Hausdorff gauge row.

The finite-cover side is exhausted by C, R, O, W. The cover row lists exactly the displayed members, the refinement row attaches covering-dimension members to those displayed parents, the overlap row records the finite order read, and the weight row keeps the Hausdorff gauge sum attached to the same cover list.

Finally H, T, P, N transport, replay, source, and name that finite route. Since the carrier has no outer-measure, countable-cover, selected-cover, host-logarithm, open-cover-compactness, or quotient-compactum coordinate, none can become a local NameCert obligation. □

Theorem 7435.3 (Hausdorff dimension covering-dimension handoff)**.** *Every* CoveringDimension$^{\uparrow}$ *consumer of an accepted* HausdorffDimensionFiniteCover$^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{HausdorffDimension}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto C, R, O, W \longmapsto \mathsf{CoveringDimension}^{\uparrow}.$$

The handoff exposes a displayed finite cover, its refinement map, its overlap-order row, and its gauge-weight ledger before the covering-dimension surface is public. It exports no Hausdorff measure theorem, open-cover compactness principle, countable cover choice, selected optimal cover, host logarithm, or ambient Euclidean embedding.

Proof. Apply Definition 7435.2. It fixes the readable surface at the rows $M, K, D, G, C, R, O, W, H, T, P, N$.

Read the source and scale rows first. The metric and compact-metric rows anchor the visible cover in displayed finite-net data, while the dyadic row and Hausdorff gauge row determine the finite scale and weight expression.

Next read the covering-dimension interface rows. The cover row C supplies the displayed members, R supplies the refinement map, O supplies the overlap-order ledger, and W keeps the gauge sum attached to the same finite cover.

The structural rows preserve the same route for downstream consumers. Therefore a covering-dimension read that bypasses the metric, compact, dyadic, gauge, cover, refinement, overlap, and weight rows is not reading the accepted $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ consumer that reaches $\mathsf{CoveringDimension}^{\uparrow}$ without first exposing the $\mathsf{HausdorffDimension}^{\uparrow}$ gauge row, $\mathsf{Met}^{\uparrow}$ observation row, $\mathsf{CompactMetric}^{\uparrow}$ source row, $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, finite cover row, refinement map, overlap row, gauge-weight row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: hausdorff-dimension, coveringdimension, metric, compactmetric, dyadicratcore: no listed sibling is bijective to $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ because this packet binds the gauge-cover, finite-refinement, compact-source, metric-observation, dyadic-scale, overlap-order, and gauge-weight rows as one finite handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}$ packet is generated from HausdorffDimensionUp gauge rows, CoveringDimensionUp finite-refinement rows, MetricUp observations, CompactMetricUp finite-net support, DyadicRatCoreUp scales, finite cover membership, overlap order, gauge weights, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7435.1, Definition 7435.2, and Definition 7435.3 seed the finite Hausdorff gauge-cover route into covering-dimension refinement consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffDimensionFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff measure existence, exact dimension values, countable cover choice, selected optimal covers, open-cover compactness, host logarithm analysis, ambient Euclidean embeddings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric and compact source exposure, dyadic scale readback, finite cover membership, refinement routing, overlap order, gauge-weight summation, transport, replay, provenance, and ledger nonescape for BEDC.Derived.HausdorffDimensionFiniteCoverUp.

Chapter 7436

A Concrete Naming Certificate for $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$

Bishop–Bridges constructive analysis treats regular Cauchy rational names through explicit tail estimates, and the $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ packet records the finite tail-replacement route used before a real-name seal is exposed. It sits after the finite-window discipline of chapter 126, the tolerance ledger of chapter 291, the regular rational readback of chapter 301, the threshold normalizer of chapter 1299, and the terminal seal of chapter 38. The packet names canonical tail replacement as displayed NameCert data; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7436.1 (RegSeqRat tail canonical carrier)**.** *A* $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (i, \theta, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *i is the finite tail index at which the replacement route begins.*
2. *θ is the cofinal threshold row showing that every later request is routed through the same displayed tail window.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule inspected at and beyond i.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the tail window and to the requested precision.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W, D after the tail threshold is displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after i, θ, W, D, R have been read.*
7. *H records componentwise* hsame *transport for the tail index, threshold, window, tolerance, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion request through the canonical tail route.*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatTailCanonical}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $i, \theta, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7436.2 (RegSeqRat tail canonical NameCert obligations)**.** *For every accepted* $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ *carrier*

$$K = (i, \theta, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegSeqRatTailCanonical}^{\uparrow}}$ *surface consists exactly of carrier admission, tail cofinality, classifier stability, readback preservation, Real-seal nonescape, and local naming over the displayed finite rows.*

Proof. Carrier admission is projection from Definition 7436.1. The public coordinates are the finite tail index, cofinal threshold, finite-window schedule, dyadic tolerance ledger, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Tail cofinality is the role of θ. It ensures that later precision requests are read through the same tail window W rather than through a private side threshold.

Classifier stability and readback preservation are componentwise through H and C. The route transports i, θ, W, D, R together, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback remains attached to the displayed tail threshold before E is exposed.

Finally P, N source and name this route. Because no carrier coordinate stores quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient completeness, those objects are outside the local NameCert obligations. □

Theorem 7436.3 (RegSeqRat tail canonical Real seal handoff)**.** *Every Real-facing read of an accepted* $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ *packet factors through the canonical tail route*

$$i, \theta, W, D, R, E, H, C, P, N.$$

The handoff exports no quotient stream equality, selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7436.2. They place the tail index, cofinal threshold, StreamName window, dyadic ledger, and RegSeqRat readback before the Real seal.

Use the sibling routes named in the chapter opening. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite inspection window, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the rational readback, and the $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ route records the shared threshold discipline.

The terminal $\mathsf{Real}^{\uparrow}$ seal is therefore downstream of i, θ, W, D, R. Transport, replay, provenance, and local naming preserve that same finite route, leaving no coordinate for the excluded quotient, choice, host-equality, trichotomy, or completeness principles. □

Falsifiable prediction: An accepted $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ consumer is refuted if it reaches the $\mathsf{Real}^{\uparrow}$ seal without first reading the finite tail index, cofinal threshold, $\mathsf{StreamName}^{\uparrow}$ window schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Independence witness: streamname, dyadicratcore, regseqrat, cauchy-tail-threshold-normalizer, real: none is bijective with $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ because this packet jointly binds one finite tail index, one cofinal threshold, one finite-window schedule, one tolerance ledger, one regular readback, one Real seal, transport, replay, provenance, and local naming as a canonical tail route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRatTailCanonical}^{\uparrow}}$.

Constructive story: A $\mathsf{RegSeqRatTailCanonical}^{\uparrow}$ packet is generated from a finite tail index, a cofinal threshold row, StreamName finite-window schedules, DyadicRatCore tolerance ledgers, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegSeqRatTailCanonical}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7436.1, Definition 7436.2, and Definition 7436.3 seed the canonical tail-replacement surface over StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, CauchyTailThresholdNormalizerUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegSeqRatTailCanonical}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove tail cofinality, classifier stability, readback preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegSeqRatTailCanonicalUp.

Chapter 7437

A Concrete Naming Certificate for $\mathsf{RaabeTest}^{\uparrow}$

Following Raabe and Duhamel on the ratio refinement for positive-term series, the $\mathsf{RaabeTest}^{\uparrow}$ packet records the finite ratio-window evidence by which a displayed first-order quotient defect is routed into constructive real-series Cauchy data. It sits beside the ratio corridor of chapter 2244, the dyadic-block comparison of chapter 1478, the Abel boundary of chapter 1496, and the root-test majorant of chapter 6709. The packet is a finite NameCert surface for Real and Completion consumers; it is not a selected sum, a countable-choice schedule, quotient stream equality, trichotomy, host summability, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7437.1 (Raabe test carrier)**.** *A* $\mathsf{RaabeTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, Q, J, M, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive real-series source row, exposing only the inspected term windows and partial-sum windows used by the current tail request.*
2. *Q is the adjacent positive ratio-window row, recording the finite quotient cells a_{n+1}/a_n on the inspected tail.*
3. *J is the unary index-threshold row, displaying the natural cutoff after which the Raabe defect is read.*
4. *M is the first-order defect-margin row, displaying the rational margin by which $n(1 - a_{n+1}/a_n)$ stays beyond the unit boundary on the opened window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that turns the finite defect row and threshold row into a tail-comparison budget.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source terms, quotient cells, partial sums, and comparison majorants.*
7. *T is the ratio-window route row, binding Q, J, M, D, W to the existing real-series ratio-test and adjacent comparison surfaces before endpoint data may be read.*
8. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row, exposed only after the ratio-window route row has been displayed.*

9. *H records componentwise* hsame *transport for the source, quotient, threshold, defect, dyadic, stream, route, endpoint, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from each tail request through Q, J, M, D, W, T, E.*
11. *P is the* Pkg *provenance over* $\mathsf{RaabeTest}^{\uparrow}$, $\mathsf{RealSeriesRatioTest}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{WeierstrassRootTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RaabeTest}^{\uparrow}}$ *row naming exactly this finite ratio-refinement packet.*

The classifier compares accepted packets only through $A, Q, J, M, D, W, T, E, H, C, P, N$. It has no coordinate for selected sums, countable choice, quotient stream equality, trichotomy, host summability, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean theorem checking, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RaabeTestUp.RaabeTestTasteGate single carrier alignment`

Theorem 7437.2 (Raabe test NameCert obligations)**.** *For every accepted* $\mathsf{RaabeTest}^{\uparrow}$ *carrier*

$$R = (A, Q, J, M, D, W, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RaabeTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the positive source row, adjacent ratio windows, unary index threshold, first-order defect margin, dyadic tolerance ledger, finite stream windows, ratio-window route, regular rational and Real handoff, componentwise transport, displayed replay, package provenance, and local naming. Every obligation-facing read must pass through these rows and may not replace them by selected sums, countable choice, quotient stream equality, trichotomy, host summability, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7437.1. The displayed rows are exactly $A, Q, J, M, D, W, T, E, H, C, P, N$, so carrier admission and local naming have no hidden analytic coordinate.

Read the finite ratio evidence first. The source row A supplies only the visible positive-term and partial-sum windows, the quotient row Q supplies adjacent positive ratio cells, and the threshold row J records the unary cutoff at which the defect row is allowed to be inspected.

The defect and comparison route is then forced through M, D, W, T. The row M records the first-order Raabe margin, D translates that margin into dyadic tolerance cells, W fixes the stream windows, and T binds the route to the surrounding ratio-test, condensation, Abel, and root-test surfaces.

Finally, E supplies the endpoint handoff through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The structural rows H, C, P, N transport, replay, source, and name the same finite packet, so none of them can introduce selected sums, countable choice, quotient stream equality, trichotomy, host summability, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge data. □

Theorem 7437.3 (Raabe test ratio-window route)**.** *Every accepted* $\mathsf{RaabeTest}^{\uparrow}$ *read reaching a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$A, Q, J \longmapsto M \longmapsto D, W \longmapsto T \longmapsto E \longmapsto (H, C, P, N).$$

Thus a constructive real-series Cauchy consumer receives ratio-refinement data only after the positive source windows, adjacent quotient windows, unary threshold, first-order defect margin, dyadic tolerance ledger, finite stream windows, ratio-window route, endpoint handoff, transport, replay, provenance, and local NameCert *row have all been displayed.*

Proof. Begin with an endpoint-facing read of an accepted carrier. The rows A, Q, J are the only data available before the defect margin is read, so the consumer first sees the positive source windows, adjacent quotient windows, and the displayed tail threshold.

The margin row M then records the Raabe first-order defect on the opened tail. It is a finite row tied to the current request, not a global choice of a sum or a detached convergence theorem.

Next inspect D, W, T. The dyadic ledger and stream window rows convert the displayed defect margin into the same finite-window discipline read by the ratio-test corridor and its condensation, Abel, and root-test sibling surfaces. The route row T is therefore the only bridge from quotient evidence to the endpoint handoff.

Only after T has been displayed may the packet expose E, the regular rational and Real-facing handoff. Transport, replay, provenance, and local naming preserve this order, so no endpoint consumer can bypass the route through selected sums, quotient stream equality, countable choice, trichotomy, host summability, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{RaabeTest}^{\uparrow}$ endpoint-facing read factors through positive source windows, adjacent quotient windows, a unary tail threshold, a first-order defect-margin row, DyadicRatCore tolerance cells, StreamName windows, a ratio-window route row, RegSeqRat and RealUp endpoint routing, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: real-series-ratio-test, cauchy-condensation-test, abel-test, weierstrass-root-test, streamname, regseqrat, real: no listed sibling supplies the whole packet $R = (A, Q, J, M, D, W, T, E, H, C,$ because each sibling contributes only a ratio corridor, dyadic block comparison, Abel boundary, root majorant, finite window surface, regular readback, or endpoint seal while this chapter binds the Raabe threshold, first-order defect margin, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RaabeTest}^{\uparrow}}$.

Constructive story: A $\mathsf{RaabeTest}^{\uparrow}$ packet is generated from positive source windows, adjacent quotient windows, a unary index threshold, a first-order defect margin, DyadicRatCore tolerance data, StreamName windows, a ratio-window route row, RegSeqRat and RealUp endpoint routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RaabeTest}^{\uparrow}}$, seed)

Scope closed: Definition 7437.1, Definition 7437.2, and Definition 7437.3 seed the finite Raabe ratio-refinement route from displayed positive-series quotient windows through regular rational readback and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RaabeTest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected sums, countable choice, quotient stream equality, trichotomy, host summability, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, quotient-window exactness, unary threshold stability, defect-margin coverage, dyadic tolerance routing, StreamName and RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RaabeTestUp.

Chapter 7438

A Concrete Naming Certificate for $\mathsf{CauchyModulusFan}^{\uparrow}$

Following Bishop and Bridges on explicit Cauchy moduli, the $\mathsf{CauchyModulusFan}^{\uparrow}$ packet names a finite family of modulus schedules under one shared tolerance ledger. It sits after the single-schedule certificate of chapter 292, the finite meet discipline of chapter 462, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. Its arithmetic consumers read the same packet through chapter 1485; the chapter records a finite fan of schedules, not a selected limit, a countable family, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7438.1 (CauchyModulusFan carrier)**.** *A* $\mathsf{CauchyModulusFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, M, D, W, Q, A, R, E, H, C, P, N)$$

with the following rows:

1. *I is the finite index ledger naming the source schedules in the fan.*
2. *M lists one accepted* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule for each index in I.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which all schedules are compared.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, equivalently a finite* ProbeBundle *of observed index-window requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached componentwise to the schedules named by M.*
6. *A is the componentwise arithmetic and meet surface read through* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, *and* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$.
7. *R is the fan-coverage row, asserting that every displayed index in I has its schedule, tolerance, observation window, and readback row present in the same finite packet.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, M, D, W, Q, A, R have been displayed.*
9. *H records componentwise* hsame *transport for the index ledger, schedules, tolerance ledger, window rows, readback rows, arithmetic rows, fan coverage, Real seal, continuation rows, provenance, and local naming row.*
10. *C records displayed* Cont *replay for a request that changes precision or index window without leaving the same finite fan.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusFan}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusFan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, D, W, Q, A, R, E, H, C, P, N$. It has no coordinate for an infinite family, a selected common limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7438.2 (CauchyModulusFan NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusFan}^{\uparrow}$ *carrier $F = (I, M, D, W, Q, A, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusFan}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite index ledger I, component schedules M, shared tolerance ledger D, observation windows W, readback rows Q, arithmetic rows A, fan-coverage row R, Real seal E, and structural rows H, C, P, N;*

2. *fan coverage, requiring each index named by I to expose a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, a shared dyadic tolerance read, a finite* $\mathsf{StreamName}^{\uparrow}$ *or* ProbeBundle *window, and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in the same packet;*

3. *shared-threshold stability, requiring tolerance refinements in D to replay componentwise through every schedule in M and through the meet and arithmetic rows in A;*

4. *RegSeqRat readback, requiring each component schedule to reach Q before a Real-facing consumer can inspect the fan;*

5. *Real-seal nonescape, requiring the terminal row E to be read only after the finite fan rows I, M, D, W, Q, A, R have been displayed.*

The obligation surface exports no selected common limit, no countable family of schedules, no quotient stream equality, no countable-choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 7438.1. The displayed rows already separate the finite index ledger, component schedules, shared tolerance ledger, observation windows, regular readbacks, arithmetic surface, coverage row, and Real seal.

For carrier admission and fan coverage, read I, M, D, W, Q, A, R in order. Each index in I is accepted only when its schedule, tolerance request, finite window, and readback are present in the same BHist packet.

For shared-threshold stability, replay a refinement of D through C and transport it through H. The meet and arithmetic rows in A keep the request componentwise, so no schedule can leave the finite fan.

For RegSeqRat readback and the Real seal, every component reaches Q before E is visible. Since E is terminal and the carrier has no row for a selected common limit, countable family, quotient stream equality, choice selector, or ambient completeness, the five obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusFan}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusFan}^{\uparrow}}$, seed)

Scope closed: Definition 7438.1 and Definition 7438.2 seed the finite fan surface over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyModulusArithmetic}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusFan}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite schedule families, selected common limits, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan coverage, shared-threshold stability, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFanUp.

Constructive story: A CauchyModulusFan$^{\uparrow}$ packet is generated from a finite index ledger, component CauchyModulus schedules, a shared DyadicRatCore tolerance ledger, StreamName or ProbeBundle windows, RegSeqRat readbacks, meet and arithmetic rows, fan coverage, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7439

A Concrete Naming Certificate for $\mathsf{QuasiBanach}^{\uparrow}$

Following Aoki–Rolewicz renorming and Kalton-style quasi-Banach analysis, the $\mathsf{QuasiBanach}^{\uparrow}$ packet records a finite quasi-normed completion surface rather than a Banach-space norm surface. It sits beside the normed-space carrier of chapter 1950, the metric-space carrier of chapter 57, the Cauchy-filter basis route of chapter 1501, the completion-embedding row of chapter 1512, and the constructive metric completion packet of chapter 1556. The carrier names a displayed quasi-triangle modulus and completion-facing Cauchy route; it does not assert local convexity, a norm equivalent to the quasi-norm, Hahn–Banach extension, quotient completion, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7439.1 (QuasiBanach carrier)**.** *A* $\mathsf{QuasiBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, A, Z, M, T, F, E, C, H, R, P, N)$$

with the following visible rows:

1. *X is the underlying additive carrier row, listing the accepted point histories and finite classifier reads used by all algebraic consumers.*
2. *A is the addition and zero row, recording the displayed finite additive operations available to the packet.*
3. *Z is the scalar action row when scalar multiplication is part of the displayed quasi-normed structure.*
4. *M is the metric-facing row induced by the quasi-norm distance ledger.*
5. *T is the quasi-triangle row, storing a displayed constant K and the finite obligation that each inspected triple satisfies $\rho(x+z) \leq K(\rho(x+y) + \rho(y+z))$ through named rational bounds.*
6. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row generated from the quasi-norm metric ledger and the visible tolerance schedule.*
7. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that embeds source observations into the completion-facing boundary without replacing the quasi-norm by a Banach norm.*
8. *C is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route through regular Cauchy names, dyadic tolerances, Cauchy-filter basis data, dense embedding, and separated comparison.*

9. *H records componentwise* hsame *transport for the carrier, algebraic rows, quasi-triangle row, Cauchy-filter basis, embedding, constructive completion route, replay, provenance, and local naming.*
10. *R records displayed* Cont *replay through the same rows.*
11. *P is the* Pkg *provenance over* $\mathsf{QuasiBanach}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiBanach}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a Banach norm, local convexity, Hahn–Banach separation, quotient completion, selected limits, or ambient real completeness.

Theorem 7439.2 (QuasiBanach NameCert obligations)**.** *For every accepted* $\mathsf{QuasiBanach}^{\uparrow}$ *carrier* $Q = (X, A, Z, M, T, F, E, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{QuasiBanach}^{\uparrow}}$ *surface consists exactly of carrier admission, additive and scalar-row admission, metric-facing quasi-norm readback, quasi-triangle budget exposure, Cauchy-filter basis generation, completion embedding, constructive metric completion routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7439.1. The rows $X, A, Z, M, T, F, E, C, H, R, P, N$ are the only available coordinates of the accepted packet.

The algebraic and metric rows are displayed before any completion-facing read. Addition, zero, and scalar action are read through A and Z, while the quasi-norm distance ledger is read through M.

The non-Banach obligation is concentrated in T. Each inspected triple carries the displayed quasi-triangle constant and rational bound rows, so a consumer may use quasi-triangle control but cannot replace it by a norm triangle inequality or local convexity theorem.

The completion-facing rows are downstream. The Cauchy-filter basis F, embedding row E, and constructive completion route C may be read only after the quasi-norm metric and quasi-triangle rows are visible.

Finally H, R, P, N transport, replay, source, and name the same finite packet. Since no row stores a Banach norm, Hahn–Banach separation, quotient completion, selected limit, or ambient real-completeness datum, the listed rows exhaust the NameCert surface. □

Theorem 7439.3 (QuasiBanach metric completion handoff)**.** *Every completion-facing read from an accepted* $\mathsf{QuasiBanach}^{\uparrow}$ *packet must expose the quasi-norm metric row M, the quasi-triangle budget row T, the Cauchy-filter basis row F, the completion embedding row E, and the constructive completion route C before any separated completion or uniformly continuous consumer can inspect the result. The handoff is a finite quasi-normed completion route through $Q = (X, A, Z, M, T, F, E, C, H, R, P, N)$, not a Banach-space completion, norm-equivalence theorem, quotient completion, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.QuasiBanachUp.QuasiBanachCarrier metric completion handoff`

Proof. Begin with the NameCert obligations of Definition 7439.2. They require every completion-facing consumer to read the accepted carrier rows and to keep the quasi-triangle budget visible.

The metric handoff starts at M and T. The row M supplies the quasi-norm distance ledger, while T records the displayed quasi-triangle constant and finite rational witnesses needed by the Cauchy route.

Next the consumer reads F, E, C in order. The Cauchy-filter basis is generated from the quasi-norm metric data, the embedding row places source observations into the completion boundary, and the constructive completion route records the finite regular-name handoff.

Transport and replay through H and R preserve this same route, and P, N only source and name it. Therefore no downstream consumer can read the handoff as Banach completion, norm equivalence, quotient completion, selected limit, or ambient real completeness. □

Falsifiable prediction: A completion-facing consumer that reads a $\mathsf{QuasiBanach}^{\uparrow}$ route without the quasi-norm metric row, quasi-triangle budget row, Cauchy-filter basis row, completion embedding row, constructive completion route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the carrier of Definition 7439.1.

Independence witness: normedspace, metricspace, cauchy-filter-basis, completionembedding, constructive-metric-completion: no listed sibling jointly carries the finite additive rows, scalar rows, quasi-norm metric ledger, non-Banach quasi-triangle budget, Cauchy-filter basis, completion embedding, constructive completion route, transport, replay, provenance, and local naming rows of $Q = (X, A, Z, M, T, F, E, C, H, R, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiBanach}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuasiBanach}^{\uparrow}$ packet is generated from additive carrier rows, scalar-action rows when present, a quasi-norm metric ledger, a displayed quasi-triangle budget, Cauchy-filter basis data, completion embedding rows, constructive metric completion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuasiBanach}^{\uparrow}}$, seed)

Scope closed: Definition 7439.1, Definition 7439.2, and Definition 7439.3 seed the finite quasi-normed completion-facing packet for functional-analysis consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{QuasiBanach}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach-space completion, norm equivalence, local convexity, Hahn–Banach separation, quotient completion, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, additive and scalar-row admission, quasi-norm metric readback, quasi-triangle budget exposure, Cauchy-filter generation, embedding, constructive completion routing, transport, replay, provenance, and local naming for BEDC.Derived.QuasiBanachUp.

Chapter 7440

A Concrete Naming Certificate for $\mathsf{CofinalityFilter}^{\uparrow}$

Following Kelley and Bourbaki on filters and Bishop–Bridges on constructive Cauchy tails, the $\mathsf{CofinalityFilter}^{\uparrow}$ packet records the finite tail-cobase interface by which a sequence-indexed completion route is read as a filter without importing quotient filters, selected limits, or countable choice. It sits after the directed-tail route of chapter 3898, the cofinal sequence interface of $\mathsf{CofinalSequence}^{\uparrow}$, the Cauchy-filter basis surface of $\mathsf{CauchyFilterBasis}^{\uparrow}$, the finite Cauchy-filter row of chapter 333, the stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite cobase corridor for Real and Completion consumers; it is not a quotient filter, a chosen convergent subsequence, a selected completed point, or an ambient completeness principle.

Definition 7440.1 (CofinalityFilter carrier)**.** *A* $\mathsf{CofinalityFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, B, F, U, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the unary tail-index row, recording finite natural-number tail requests for the sequence-indexed route.*
2. *L is the cofinal lower-bound ledger. For every displayed tail request in I, it stores a later index whose inspected window is still inside the same requested tail.*
3. *B is the cobase row, listing the finite family of tail neighborhoods generated from I and refined by L.*
4. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row assembled only from the cobase row B.*
5. *U is the Completion corridor row, available only after the directed and filter-facing tail data have been replayed through I, L, B, F.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached through* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the same cobase route.*
7. *H records componentwise* hsame *transport for the tail index, lower-bound ledger, cobase, filter-basis handoff, completion corridor, Real seal, replay, provenance, and local name rows.*

8. *C records displayed* Cont *replay of the finite route*

$$I, L \longmapsto B \longmapsto F \longmapsto U, E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CofinalityFilter}^{\uparrow}$, $\mathsf{DirectedFilter}^{\uparrow}$, $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalityFilter}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $I, L, B, F, U, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable-choice schedules, host equality of filters, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7440.2 (CofinalityFilter NameCert obligations)**.** *For every accepted* $\mathsf{CofinalityFilter}^{\uparrow}$ *carrier $K = (I, L, B, F, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalityFilter}^{\uparrow}}$ *surface consists exactly of the finite obligations for tail-index admission, cofinal lower-bound exactness, cobase generation, Cauchy-filter basis handoff, completion-corridor nonescape, Real-seal readback, componentwise transport, displayed replay, provenance, and local naming. Every obligation is read from $I, L, B, F, U, E, H, C, P, N$, and no obligation is stored in a quotient filter, selected limit, countable-choice schedule, host equality relation, quotient real equality row, or ambient completeness witness.*

Proof. Project the carrier of Definition 7440.1. The displayed analytic rows are I, L, B, F, U, E, and the structural rows are H, C, P, N.

Tail-index admission is supplied by I. The lower-bound ledger L is then read against the same finite tail requests, so a later window is displayed before any cobase or filter consumer inspects the packet.

The cobase and filter rows are downstream. The row B records exactly the tail neighborhoods generated from the displayed sequence indices, while F presents those neighborhoods through the $\mathsf{CauchyFilterBasis}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$ handoff surface.

Finally U, E, H, C, P, N preserve the same route for Completion and Real-facing consumers. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7440.3 (CofinalityFilter tail cobase handoff)**.** *Every* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, *Completion, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalityFilter}^{\uparrow}$ *carrier must read the unary tail-index row I, the cofinal lower-bound ledger L, and the cobase row B before the handoff row F is available. The handoff presents finite tail neighborhoods, cofinal lower-bound witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it does not present a quotient filter, selected limit, countable-choice selector, host filter equality, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.CofinalityFilterUp.CofinalityFilterCarrier tail cobase handoff`

Proof. Begin with the NameCert obligations of Definition 7440.2. They place every filter-facing read after the displayed tail-index row I and cofinal lower-bound ledger L.

Read B next. Because B is assembled only from the finite tail neighborhoods witnessed by I and refined by L, the consumer receives a displayed cobase rather than an external filter object.

The row F hands that cobase to $\mathsf{CauchyFilterBasis}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$, and the rows U, E keep Completion and Real-facing reads downstream of the same cobase. The structural rows H, C, P, N transport, replay, source, and name that route, so none of the excluded objects can be recovered from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalityFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalityFilter}^{\uparrow}$ packet is generated from a unary tail-index row, a cofinal lower-bound ledger, a finite cobase row, a CauchyFilterBasis and CauchyFilter handoff, a Completion corridor row, a RealUp seal through StreamName and RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalityFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7440.1, Definition 7440.2, and Definition 7440.3 seed the finite cofinality-filter cobase interface for sequence-indexed Completion and Real routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalityFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host filter equality, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal lower-bound exactness, cobase generation, Cauchy-filter basis handoff, completion-corridor nonescape, Real-seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CofinalityFilterUp.

Chapter 7441

A Concrete Naming Certificate for $\mathsf{ExtendedMetricSpace}^{\uparrow}$

Following Lawvere and Kelley-style extended metrics, the $\mathsf{ExtendedMetricSpace}^{\uparrow}$ packet records a finite BEDC carrier for distances that may leave the finite-real window while still exposing every finite-radius read through the ordinary metric and real-number siblings. It sits beside $\mathsf{Met}^{\uparrow}$ and $\mathsf{LawvereMetric}^{\uparrow}$ as the distance-interface packet for radius-bounded arguments, and it reads finite radii through chapter 38, chapter 301, chapter 126, and chapter 291. The chapter supplies only the finite NameCert route for extended distances and truncations; it does not import quotient metrics, ambient infinity arithmetic, or a completeness theorem.

Definition 7441.1 (Extended metric space carrier)**.** *An* $\mathsf{ExtendedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (P, D, F, T, M, L, H, C, G, N)$$

with the following displayed rows:

1. *P is the point-name row, naming the finite observation surface on which distance probes are evaluated.*
2. *D is the extended-distance row, recording a nonnegative distance comparison that may be finite or declared outside the finite-radius window.*
3. *F is the finite-radius window row, naming the radii for which D is read through* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$.
4. *T is the finite truncation row, recording the bounded read $d_T(x, y) = \min(d(x, y), r)$ for a displayed finite radius r.*
5. *M is the ordinary metric handoff row, connecting every finite-radius truncation to the sibling* $\mathsf{Met}^{\uparrow}$ *packet.*
6. *L is the Lawvere metric sibling row, recording that the extended-distance orientation is a Lawvere-style distance surface, not a quotient metric.*
7. *H stores componentwise* hsame *transport for point names, extended-distance reads, finite windows, truncations, sibling handoffs, provenance, and local naming.*
8. *C stores displayed* Cont *replay for finite-radius reads and sibling handoffs.*
9. *G is the* Pkg *provenance over* $\mathsf{ExtendedMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LawvereMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ExtendedMetricSpace}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient metric identification, host-level infinity arithmetic, hidden completeness, selected limit points, or an ambient real-number object outside the displayed finite-radius route.

Theorem 7441.2 (Extended metric space NameCert obligations)**.** *For the carrier of Definition 7441.1, the local* $\mathsf{NameCert}_{\mathsf{ExtendedMetricSpace}^{\uparrow}}$ *surface consists exactly of point-name admission through P, extended-distance admission through D, finite-radius admission through F, truncation admission through T, ordinary metric handoff through M, Lawvere sibling orientation through L, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ExtendedMetricSpaceUp.ExtendedMetricSpaceCarrier namecert obligations`

Proof. Project the accepted carrier $X = (P, D, F, T, M, L, H, C, G, N)$. Each listed obligation is one displayed coordinate of the carrier, and the carrier definition assigns no additional row to quotient metrics, completeness, host infinity arithmetic, selected limits, or ambient real equality.

The finite-radius obligations are the only rows that read numerical distance data as finite. Row F routes those reads through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$, while T records the radius-bounded truncation attached to the same finite window.

The sibling rows preserve the intended interface. Row M hands finite-radius truncations to $\mathsf{Met}^{\uparrow}$, and row L records the Lawvere-style extended-distance orientation. Transport, replay, provenance, and local naming then preserve the same displayed packet, so the listed rows exhaust the NameCert surface. □

Theorem 7441.3 (Extended metric space finite-radius truncation)**.** *Every accepted finite-radius read of an* $\mathsf{ExtendedMetricSpace}^{\uparrow}$ *carrier factors through*

$$F, T, M, \mathsf{Real}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}.$$

The truncation may expose a bounded ordinary metric read, but it cannot use an unlisted infinity calculus, quotient completion witness, selected limit, or ambient completeness claim.

Proof. Start with the finite-radius row F. By definition it is the only row that certifies a distance read as finite, and it routes the displayed radius through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Next use the truncation row T. A bounded read is recorded as a finite-radius truncation of the extended-distance row D, so it remains tied to the same radius window rather than to host infinity arithmetic.

Finally apply the ordinary metric handoff row M. It sends the bounded truncation to the sibling $\mathsf{Met}^{\uparrow}$ surface. Since the carrier contains no coordinate for quotient completion, selected limits, or ambient completeness, the finite-radius truncation exports only the displayed route. □

Theorem 7441.4 (Extended metric space metric-space handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer of an* $\mathsf{ExtendedMetricSpace}^{\uparrow}$ *packet must first restrict to a displayed finite-radius truncation. The consumer may read point names, the finite radius, the truncated distance, componentwise transport, replay, provenance, and local naming, but it may not read the raw extended-distance row as an ordinary metric without the F, T, M handoff.*

Proof. Read the carrier $X = (P, D, F, T, M, L, H, C, G, N)$. The raw extended-distance row D is allowed to sit outside the finite-radius window, so it is not by itself an ordinary metric-space distance.

Apply the finite-radius truncation theorem. It supplies the route through F, T, M and through the real, regular-rational, stream, and dyadic siblings before the bounded distance is exported.

The $\mathsf{Met}^{\uparrow}$ handoff is therefore exactly the row M applied after F and T. The structural rows H, C, G, N preserve that same bounded route, while the Lawvere row L records the sibling orientation. No consumer can bypass those rows and reread D as an ordinary metric. □

Falsifiable prediction: An ordinary metric consumer that reads an $\mathsf{ExtendedMetricSpace}^{\uparrow}$ distance without exposing F, T, M and the $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ route is not reading the local carrier.

Independence witness: metricspace, lawveremetric, real, regseqrat, streamname, dyadicratcore: no listed sibling jointly carries extended-distance admission, finite-radius truncation, ordinary metric handoff, Lawvere orientation, real and rational source rows, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ExtendedMetricSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ExtendedMetricSpace}^{\uparrow}$ packet is generated from point-name rows, extended-distance rows, finite-radius windows, truncation rows, MetricSpace and LawvereMetric sibling handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtendedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7441.1, Definition 7441.2, Definition 7441.3, and Definition 7441.4 seed the finite-radius certificate for extended distances and ordinary metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtendedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient metric identification, ambient infinity arithmetic, completeness, selected limits, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-radius truncation exactness, MetricSpace handoff exactness, Lawvere sibling separation, transport, replay, provenance, and local naming for BEDC.Derived.ExtendedMetricSpaceUp.

Chapter 7442

A Concrete Naming Certificate for $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$

Following Bishop and Bridges on constructive interval arguments, the $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$ packet records a located nested-interval route as finite BEDC data. It sits between the located interval carrier of chapter 394, the nested-chain surface of chapter 320, the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names the finite route by which a nested located interval argument can feed Real and Completion consumers; it does not assert open-cover compactness, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7442.1 (Bishop located interval nest carrier)**.** *A* $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, K, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite family of* $\mathsf{LocatedInterval}^{\uparrow}$ *rows inspected by the consumer.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-margin ledger recording positive dyadic margins for the displayed intervals.*
3. *K is the* $\mathsf{NestedInterval}^{\uparrow}$ *containment spine, stating which displayed interval row is retained at each inspected stage.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window surface used to read the retained endpoints and margin requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite endpoint windows selected by S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after I, D, K, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for located intervals, dyadic margins, containment-spine rows, stream windows, readbacks, seal rows, provenance, replay, and local naming.*
8. *C records displayed* Cont *replay from located-interval reads through endpoint margins, nested containment, stream windows, regular readbacks, and the terminal seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalNest}^{\uparrow}}$ *row naming exactly this finite located-nest packet.*

The local classifier compares accepted packets only through these rows. It has no coordinate for an infinite host intersection, a chosen real point, quotient equality of real names, open-cover compactness, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7442.2 (Bishop located interval nest NameCert obligations)**.** *For every accepted* BishopLocatedIntervalNe *carrier*

$$L = (I, D, K, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalNest}^{\uparrow}}$ *surface consists exactly of located-interval exposure through* I*, dyadic endpoint-margin accounting through* D*, nested-containment spine exposure through* K*, finite window selection through* S*, regular rational readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary through* E*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*. A consumer cannot read an interval-nest convergence route without first exposing those rows.*

Proof. Project the carrier of Definition 7442.1. The located intervals, dyadic endpoint margins, nested containment spine, stream windows, regular readbacks, Real seal boundary, and structural rows are the complete visible packet.

The analytic route is componentwise. The located interval rows in I provide endpoint and interval-cell data; the dyadic margin ledger D records positive endpoint margins; and the containment spine K states the finite nesting relations inspected by the consumer.

Next read S, Q, E. The stream row selects the finite endpoint windows, the regular rational row supplies their readback, and the Real seal row is exposed only after those windows and margins have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier lacks any coordinate for host interval compactness, quotient real equality, selected limits, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7442.3 (Bishop located interval nest regular Cauchy consumer)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$ *packet must factor through the displayed route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the finite rows $I, D, K, S, Q, E, H, C, P, N$*. The route may be transported and replayed, but it cannot export a selected point of the infinite nest, quotient real equality, countable-choice data, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the located interval and nesting rows. A consumer first reads I and K, so the retained interval stages and their containment spine are visible before any regular Cauchy or Real-facing handoff is available.

Then read the dyadic margin and window rows. The ledger D supplies the positive endpoint-margin data, while S determines the finite windows on which those margins and endpoint reads are inspected.

Now pass to regular rational data and the Real seal. The row Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ readback for the same finite windows, and E is downstream of that readback. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route.

No row remains for a host-selected nest point, quotient equality of real names, countable choice, open-cover compactness, or ambient completeness. The consumer therefore receives only the displayed regular Cauchy handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalNest}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedIntervalNest}^{\uparrow}$ packet is generated from displayed located interval rows, dyadic endpoint-margin rows, a nested containment spine, StreamName windows, RegSeqRat readback, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalNest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7442.1, Definition 7442.2, and Definition 7442.3 seed the finite located-interval-nest route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalNest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval exposure, dyadic endpoint-margin accounting, nested containment, StreamName window selection, RegSeqRat readback, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedIntervalNestUp.

Chapter 7443

A Concrete Naming Certificate for $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$

Following Bishop–Bridges style located compact interval enclosures, the $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$ packet records a finite boundary around one displayed located interval cell before any compactness or continuity consumer reads it. It sits after the endpoint-cell certificate of chapter 394, the finite-net route of chapter 1433, and the bisection handoff of chapter 1465. The packet is a local compact-enclosure boundary over finite observation rows; it is not open-cover compactness, a selected subcover, a host interval object, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7443.1 (Localized compact enclosure carrier)**.** *A* $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, K, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, carrying the located endpoint-cell boundary to be enclosed.*
2. *K is the* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *finite-net row for the displayed interval cell.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row through which downstream names request finite windows inside the enclosure.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by Real and completion-facing consumers of the same enclosure.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite interval and readback rows have been displayed.*
6. *H records componentwise* hsame *transport for I, K, S, Q, E and the structural rows.*
7. *C records displayed* Cont *replay of the route from the located interval row through the finite-net row to stream, regular-rational, and real-seal consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalizedCompactEnclosure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, K, S, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers, host interval equality, quotient real equality, arbitrary choice of a mesh, or ambient completeness.

Theorem 7443.2 (Localized compact enclosure NameCert obligations)**.** *For every accepted* LocalizedCompactEnclo *carrier* $L = (I, K, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocalizedCompactEnclosure}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval boundary exposure, finite-net admission, stream-window exposure, regular-rational readback, Real seal factorization, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the enclosure must read these rows before it can consume the terminal seal.*

Proof. Project the carrier of Definition 7443.1. The rows $I, K, S, Q, E, H, C, P, N$ expose the located interval source, finite-net enclosure, stream windows, regular readbacks, Real seal, transport, replay, provenance, and local name.

Use the sibling endpoint-cell route of chapter 394. It supplies the finite located interval boundary read through $\mathsf{LocatedInterval}^{\uparrow}$ rather than through a host interval or selected endpoint.

Apply the finite-net route of chapter 1433. The row K is the only compact-enclosure row, and it is displayed before any stream, regular-rational, or real-seal consumer can read the packet.

Finally replay the structural rows H, C, P, N. They transport, route, source, and name the same finite enclosure surface. Since the carrier contains no open-cover, selected-subcover, quotient-real, mesh-choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7443.3 (Localized compact enclosure finite-net handoff)**.** *Every accepted* $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$ *carrier* $L = (I, K, S, Q, E, H, C, P, N)$ *hands a downstream compactness or uniform-continuity consumer the same finite route*

$\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The consumer receives the located interval row I *and the finite-net row* K *before it may read the stream windows* S*, regular readbacks* Q*, or seal row* E*. The handoff supplies a finite enclosure boundary only; it does not claim open-cover compactness, selected subcovers, host function equality, quotient real equality, or ambient completeness.*

Proof. Start with the located interval row I. It is the same endpoint-cell boundary governed by chapter 394, so the enclosure begins with displayed finite interval data.

Read the compact-net row K next. By chapter 1433, this row is an explicit finite-net handoff for the displayed interval cell, not a general open-cover compactness principle.

The stream and regular-rational rows S, Q then expose the finite observation windows consumed by downstream names. The seal row E is reachable only after those finite rows are displayed.

Transport and replay through H, C, then attach P, N. The resulting L10 dependency route is the same boundary used by interval bisection in chapter 1465, but the present packet exports only the localized enclosure surface and not a branch selector or selected subcover. □

Falsifiable prediction: Every accepted $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$ consumer displays a $\mathsf{LocatedInterval}^{\uparrow}$ source row, a $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ finite-net row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any terminal compactness or Real-facing read.

Independence witness: locatedinterval, closed-interval-compact-net, streamname, regseqrat, real: $\mathsf{LocalizedCompactEnclosure}^{\uparrow}$ is not bijective to any listed sibling because it packages one localized finite enclosure boundary between a displayed located interval cell and later compactness or uniform-continuity consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalizedCompactEnclosure}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocalizedCompactEnclosure}^{\uparrow}}$, seed)

Scope closed: Definition 7443.1, Definition 7443.2, and Definition 7443.3 define the finite localized compact-enclosure boundary over located intervals and finite-net consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocalizedCompactEnclosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, host interval equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, finite-net handoff, consumer nonescape, transport replay, and local NameCert exactness for BEDC.Derived.LocalizedCompactEnclosureUp.

Constructive story: A LocalizedCompactEnclosure$^{\uparrow}$ packet is generated from a LocatedIntervalUp source row, a ClosedIntervalCompactNetUp finite-net row, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7444

A Concrete Naming Certificate for $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$

Following Bishop and Bridges constructive metric-space practice, the $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$ packet records a metric carrier whose distinct points are separated by a displayed positive rational floor. It refines the $\mathsf{Met}^{\uparrow}$ surface of chapter 57 and the $\mathsf{LocatedMetric}^{\uparrow}$ distance-comparison surface of chapter 1533, reads the dyadic lower-bound arithmetic through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$, and exposes a $\mathsf{StreamName}^{\uparrow}$ scheduling row for later Cauchy-stabilization consumers. The packet is a finite separation ledger; it is not a decidable equality principle, an excluded-middle split on points, a host discrete topology, or an ambient completeness theorem.

Definition 7444.1 (Uniformly discrete metric carrier)**.** *A* $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, L, D, R, W, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point histories, distance-witness histories, symmetry, and triangle ledgers consumed by metric users.*
2. *L is the* $\mathsf{LocatedMetric}^{\uparrow}$ *row exposing located distance comparisons for the same source metric.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive separation ledger. It stores a displayed dyadic rational $\eta > 0$ and the finite comparisons used by lower-bound reads.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *positivity and order row associated to the same separation floor, so consumers can read the floor through rational order rather than through host arithmetic.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule for distance observations. It records which finite distance windows are inspected before a separation claim is consumed.*
6. *H stores componentwise* hsame *transport for the metric source, located-distance row, dyadic floor, rational positivity row, request schedule, provenance, and local name.*
7. *C stores the displayed* Cont *routes for reading a distance request, comparing it with the dyadic floor, transporting the lower-bound row, and exposing the consumer handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteMetric}^{\uparrow}}$ *row.*

The classifier on such packets is componentwise replay through H and C. A consumer may read the displayed positive separation floor and the scheduled metric-distance rows, but it receives no host-level point decision procedure and no completed-space datum.

Theorem 7444.2 (Uniformly discrete metric NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$ *carrier* $U = (X, L, D, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteMetric}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed metric source, located metric row, dyadic positive floor, rational order row, stream request schedule, transport rows, continuation routes, provenance, and local naming data;*
2. *separation readability, requiring every distinct-point lower-bound read to consume* X, L, D, R, W *before it exposes the positive floor;*
3. *classifier stability, requiring replacement by classified point histories to transport the metric, locatedness, dyadic floor, rational order, and stream schedule componentwise through* H *and* C*;*
4. *finite ledger exactness, requiring the separation floor to be exactly the dyadic-rational ledger* D *with its* $\mathsf{Rat}^{\uparrow}$ *positivity row* R*;*
5. *downstream handoff, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer to read the finite separation ledger before using the metric as a stabilized discrete source.*

No obligation exports decidable equality, excluded middle, a host discrete topology, selected representatives, quotient equality, or ambient completeness.

Proof. Carrier admission is projection from Definition 7444.1. The packet lists the metric source, located comparison row, dyadic floor, rational positivity row, stream schedule, transport and continuation rows, provenance, and local name.

Separation readability is the route order. A consumer first reads the metric distance witness in X, then the located comparison in L, then the positive dyadic floor in D and its rational order row in R, with W recording the finite observation window used by that read.

Classifier stability is componentwise. The transport row H identifies the source, located-distance, dyadic, rational, and stream rows after classifier replacement, while C replays the same lower-bound comparison. The read therefore preserves the finite floor ledger rather than creating a point-decision principle.

The downstream handoff is just the exposed ledger $X, L, D, R, W, H, C, P, N$. Since no coordinate stores decidable equality, a selected representative, quotient equality, or a completion witness, no consumer can obtain such data from the local $\mathsf{NameCert}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteMetric}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyDiscreteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7444.1 and Definition 7444.2 seed the finite positive-separation ledger for metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyDiscreteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim decidable equality, excluded middle, host discreteness, quotient equality, selected representatives, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, separation readability, classifier stability, floor exactness, and downstream handoff for the same packet.

Constructive story: A $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$ packet is generated from a MetricSpace source, a LocatedMetric comparison row, a DyadicRatCore positive floor, a RatUp positivity row, a StreamName observation schedule, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Chapter 7445

A Concrete Naming Certificate for CompactUniformDyadicRadiusLadder$^{\uparrow}$

The CompactUniformDyadicRadiusLadder$^{\uparrow}$ packet realizes the compact-uniform-continuity route of chapter 9237 as finite NameCert data. The ladder consumes the compact source of chapter 129, the pointwise-radius interface of chapter 130, the finite lower-bound row of chapter 452, the dyadic comparison substrate of chapter 291, and the downstream handoff of chapter 369. It packages compact finite-net rows, continuous-map pointwise radii, finite Lebesgue-number lower-bound ledgers, dyadic radius comparisons, and the final UniformModulus$^{\uparrow}$ handoff. The carrier is a finite dyadic radius ladder for compact metric sources; it is not an external modulus selector, an open-cover compactness theorem, a sequential compactness argument, or a host minimum over a set of radii.

Definition 7445.1 (Compact uniform dyadic radius ladder)**.** *A* CompactUniformDyadicRadiusLadder$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, G, L, D, U, H, C, P, N)$$

with the following visible rows:

1. *X is the* CompactMetric$^{\uparrow}$ *source row, including the finite compact-net and located metric data consumed by the uniform-continuity route.*
2. *F is the compact finite-net ledger, with its* ProbeBundle *centers, membership constructors, coverage rows, and finite radius requests.*
3. *G is the* ContinuousMap$^{\uparrow}$ *pointwise-radius row for the map under consideration, including the local modulus witnesses attached to bundled centers.*
4. *L is the* FiniteLebesgueNumber$^{\uparrow}$ *lower-bound ledger obtained by folding the bundled pointwise radii into one positive cover radius.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *radius ladder. It records the dyadic refinements and comparisons used to pass from local radii to the global uniform precision.*
6. *U is the* UniformModulus$^{\uparrow}$ *handoff row whose tolerance-only precision is read from L and D.*
7. *H stores componentwise* hsame *transport for the compact source, finite net, continuous-map radii, Lebesgue-number ledger, dyadic ladder, uniform-modulus row, provenance, and local name.*
8. *C stores the displayed* Cont *routes for reading compact coverage, pointwise radii, finite lower-bound folds, dyadic refinements, and the final uniform-modulus handoff.*

9. *P is the* Pkg *provenance over* CompactUniformDyadicRadiusLadder$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CompactUniformDyadicRadiusLadder}^{\uparrow}}$ *row.*

The carrier exports exactly the finite ladder from compact finite nets and pointwise continuous radii to a tolerance-only uniform-modulus row. It stores no external minimum, no sequential compactness witness, and no selected modulus function.

Theorem 7445.2 (Compact uniform dyadic radius ladder NameCert obligations)**.** *For every accepted* CompactUniformDyadicRadiusLadder$^{\uparrow}$ *carrier $K = (X, F, G, L, D, U, H, C, P, N)$, the local* NameCert$_{\text{CompactUniformDyadicRadiusLadder}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the compact metric source, finite-net ledger, continuous-map pointwise-radius row, finite Lebesgue-number row, dyadic radius ladder, uniform-modulus handoff, transports, continuations, provenance, and local name;*
2. *finite-net coverage, requiring every uniform-modulus read to begin with the compact* ProbeBundle *centers and coverage witnesses in X and F;*
3. *pointwise-radius attachment, requiring each bundled center to carry its* ContinuousMap$^{\uparrow}$ *local radius row G before the lower-bound fold is consumed;*
4. *lower-bound exactness, requiring the folded positive radius to pass through the* FiniteLebesgueNumber$^{\uparrow}$ *row L and the dyadic comparison ladder D before U is exposed;*
5. *downstream handoff, requiring every* UniformModulus$^{\uparrow}$ *consumer to read $X, F, G, L, D, U, H, C, P, N$ and no external compactness, selected minimum, or host modulus function.*

Proof. Carrier admission is projection from Definition 7445.1. The packet displays the compact source, finite-net rows, continuous-map local radii, finite lower-bound row, dyadic ladder, uniform-modulus handoff, transport and continuation rows, provenance, and local name.

The finite-net coverage and pointwise-radius obligations follow the compact-uniform route in chapter 9237. Each uniform-modulus read first opens the compact finite net, then reads the local radius data for the bundled centers through the continuous-map row.

The lower-bound step is finite. The bundled radii are folded through the FiniteLebesgueNumber$^{\uparrow}$ ledger, and the resulting positive radius is refined through the dyadic ladder D before the UniformModulus$^{\uparrow}$ row can be exposed.

Transport and continuation replay preserve the same row order. Since the carrier contains no external minimum, sequential compactness witness, or selected modulus function, the downstream consumer receives only the finite NameCert ladder. □

Lean4 ✓ `BEDC.Derived.CompactUniformDyadicRadiusLadderUp.CompactUniformDyadicRadiusLadderNameCertObligat`

Closure status for NameCert$_{\text{CompactUniformDyadicRadiusLadder}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformDyadicRadiusLadder}^{\uparrow}}$, seed)

Scope closed: Definition 7445.1 and Definition 7445.2 seed the finite compact-net, pointwise-radius, lower-bound, dyadic-ladder, and uniform-modulus route.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformDyadicRadiusLadder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim external modulus selection, sequential compactness, host minima over radius sets, open-cover compactness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, pointwise-radius attachment, lower-bound exactness, and uniform-modulus handoff for the same packet.

Constructive story: A CompactUniformDyadicRadiusLadder$^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-radius rows, a FiniteLebesgueNumber lower-bound ledger, DyadicRatCore radius comparisons, a UniformModulus handoff, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 7446

A Concrete Naming Certificate for $\mathsf{CompactChoiceSequence}^{\uparrow}$

Following Brouwer's choice sequences, Troelstra–van Dalen's fan and bar practice, and Bishop–Bridges constructive analysis, the $\mathsf{CompactChoiceSequence}^{\uparrow}$ packet records the finite-prefix compactness surface used when a choice sequence is read through a fan bound and then handed to Real-facing consumers. It sits beside the located prefix route of chapter 2088, the finite bar packet of chapter 400, the schedule discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal seal of chapter 38. The chapter names only displayed prefixes, finite fan bounds, tolerance rows, readback rows, and local naming data; it does not select an infinite branch or import host compactness.

Definition 7446.1 (Compact choice-sequence carrier)**.** *A* $\mathsf{CompactChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, F, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row whose visible answers are available to this packet.*
2. *F is the fan and bar finite-bound row, read through* $\mathsf{FanTheorem}^{\uparrow}$*, naming the bounded prefix depth and the decidable bar data used by compact consumers.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing exactly the finite prefix positions inspected under the bound F.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the inspected prefixes and to the finite bar depth.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after Q, F, W, D have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular rational readback R.*
7. *H stores componentwise* hsame *transport for the prefix row, fan bound, schedule, tolerance row, readback, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a compactness request through the finite prefix bound, the scheduled windows, the dyadic tolerance row, the regular rational readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactChoiceSequence}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactChoiceSequence}^{\uparrow}}$ *row.*
The dependency route is the finite handoff

$$\begin{aligned} Q, F &\longmapsto W \\ &\longmapsto D \\ &\longmapsto R \\ &\longmapsto E. \end{aligned}$$

Thus a compact choice-sequence consumer receives a finite prefix and fan-bound interface before any regular rational or real-seal row can be read.

Theorem 7446.2 (Compact choice-sequence NameCert obligations)**.** *The packet of Definition 7446.1 satisfies the seed local* $\mathsf{NameCert}_{\mathsf{CompactChoiceSequence}^{\uparrow}}$ *obligation surface. Carrier admission is the displayed ten-row packet K; stability is componentwise* hsame *transport of $Q, F, W, D, R, E, H, C, P, N$; exactness says that every accepted read factors through the finite route from choice-sequence prefix and fan bound to stream schedule, dyadic tolerance, regular rational readback, real seal, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7446.1. The rows $Q, F, W, D, R, E, H, C, P, N$ are all displayed coordinates of one finite BHist packet.

For stability, transport each coordinate through H. The transported packet keeps the same order of dependence: the choice prefix and fan bound determine the scheduled prefix window, the scheduled window carries the dyadic tolerance row, and the readback and seal are reached only after those rows have been displayed.

For exactness, replay the consumer request through C and then repack the projected rows with P and N. Coordinate exhaustion gives the local naming surface and leaves no row for selected infinite branches, countable choice, quotient stream equality, open-cover compactness, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7446.3 (Compact choice-sequence fan-prefix handoff)**.** *Every compactness-facing read of an accepted* $\mathsf{CompactChoiceSequence}^{\uparrow}$ *packet must pass through the fan/bar row F and the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule W before it can inspect the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D or the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The handoff is a finite route from* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefixes and* $\mathsf{FanTheorem}^{\uparrow}$ *bounds to scheduled regular rational data, not a selected branch, countable-choice functional, open-cover compactness theorem, quotient stream equality, or ambient real-completeness claim.*

Proof. Begin with a consumer read of K. The carrier exposes Q as the only choice-sequence source and F as the only fan/bar finite-bound row, so the request has no route to a tail value outside the displayed prefix surface.

Next read the schedule W. It names exactly the prefix positions inspected under F, and the dyadic row D is attached only to those scheduled positions.

Finally read R and then any real-facing seal E. Since both rows occur after Q, F, W, D in the carrier route, the handoff factors through finite prefix and fan data. None of the excluded branch-selection, choice, quotient, open-cover, or completeness principles is a coordinate of the packet. □

Theorem 7446.4 (Compact choice-sequence Real-seal nonescape)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactChoiceSequence}^{\uparrow}$ *packet can read the seal row E only after the route*

$$Q, F, W, D, R$$

has been displayed. The seal exports the finite regular rational readback obtained from the compact prefix surface, but it cannot be used to recover selected infinite branches, countable choice, quotient stream equality, open-cover compactness, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the fan-prefix handoff of Definition 7446.3. It forces every compactness-facing read through the choice prefix, fan bound, schedule, dyadic tolerance row, and regular rational readback before any real-facing row is available.

Now project the seal coordinate E from Definition 7446.1. The seal is terminal in the displayed route and records only the real-facing presentation of the already exposed regular rational readback.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because the carrier has no selected branch row, countable-choice row, quotient stream equality row, open-cover compactness row, trichotomy row, or ambient-completeness row, the $\mathsf{Real}^{\uparrow}$ seal cannot escape the finite compact choice-sequence packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactChoiceSequence}^{\uparrow}$ packet is generated from a displayed ChoiceSequence prefix row, a FanTheorem finite bar-bound row, a StreamName schedule, a DyadicRatCore tolerance row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactChoiceSequence}^{\uparrow}}$, seed)

Scope closed: Definition 7446.1, Definition 7446.2, Definition 7446.3, and Definition 7446.4 seed the finite compact choice-sequence route from displayed prefixes and fan bounds through regular rational readback to a Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactChoiceSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, countable choice, quotient stream equality, open-cover compactness, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-prefix handoff, Real-seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 7447

A Concrete Naming Certificate for $\mathsf{HadamardBoundary}^{\uparrow}$

Following Bridson–Haefliger and Ballmann on CAT(0) and Hadamard boundaries, the $\mathsf{HadamardBoundary}^{\uparrow}$ packet records the finite readback by which geodesic-ray, Busemann, visual-boundary, metric, and Real-seal rows expose a boundary direction without selecting a point at infinity from a host completion. It sits beside the hyperbolic metric surface of chapter 3474, the Busemann boundary route of chapter 3732, the visual boundary route of chapter 4181, the geodesic-flow route of chapter 3856, and the terminal scalar seal of chapter 38. The packet names a traditional CAT(0)/Hadamard boundary read as displayed NameCert data; it does not import a quotient boundary, selected asymptotic ray, global compactification, countable choice, or ambient completeness.

Definition 7447.1 (Hadamard boundary carrier)**.** *A* $\mathsf{HadamardBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, M, R, G, U, V, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the CAT(0) or Hadamard source surface, read only as finite metric and convexity observations.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row used for distance and thinness-style readback when a hyperbolic model supplies the visible metric window.*
3. *R is the finite geodesic-ray window, recording initial segments and comparison requests without choosing a global representative ray.*
4. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *transport row attached to the displayed ray window.*
5. *U is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, recording Busemann-level boundary values over the same source window.*
6. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row, recording the visual boundary face exposed by the ray and Busemann rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for the scalar comparison values carried by M, U, V.*
8. *H records componentwise* hsame *transport for the source, metric, ray, flow, Busemann, visual-boundary, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$X, R \longmapsto M, G \longmapsto U, V \longmapsto E.$$

10. *P is the* Pkg *provenance over* HadamardBoundary$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{HadamardBoundary}^{\uparrow}}$ *row.*

The carrier stores no quotient ideal boundary, selected asymptotic ray, host CAT(0) compactification, countable-choice schedule, or ambient completeness coordinate.

Theorem 7447.2 (Hadamard boundary NameCert obligations)**.** *For every accepted* HadamardBoundary$^{\uparrow}$ *carrier $B = (X, M, R, G, U, V, E, H, C, P, N)$, the local* NameCert$_{\text{HadamardBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite ray-window visibility, metric readback, geodesic-flow transport, Busemann boundary exposure, visual-boundary exposure, Real-seal discipline, componentwise transport, displayed replay, provenance, and local naming. The certificate refuses quotient boundary equality, selected ray representatives, host compactification, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 7447.1. The displayed rows list the source surface, metric readback, finite geodesic-ray window, geodesic-flow transport, Busemann row, visual-boundary row, Real seal, transport, replay, provenance, and local name.

The obligations follow the carrier order. A boundary-facing request first exposes the finite ray window over the source surface, then reads metric and geodesic-flow rows, then reaches Busemann and visual-boundary rows, and only then consumes the Real-valued seal.

Transport and replay do not add a second boundary object. They preserve the same rows componentwise and repeat the same route for downstream consumers.

Since no quotient boundary, selected asymptotic ray, host compactification, countable-choice schedule, or ambient completeness witness is a coordinate of the packet, the listed obligations exhaust the local NameCert surface. □

Theorem 7447.3 (Hadamard boundary finite ray-window handoff)**.** *Every CAT(0)/Hadamard boundary read exported by an accepted* HadamardBoundary$^{\uparrow}$ *carrier must pass through the finite geodesic-ray row R, the* HyperbolicMetric$^{\uparrow}$ *metric comparison row M, the* HyperbolicGeodesicFlow$^{\uparrow}$ *transport row G, the* HyperbolicBusemannBoundary$^{\uparrow}$ *row U, the* HyperbolicVisualBoundary$^{\uparrow}$ *row V, and the* Real$^{\uparrow}$ *seal E before any boundary consumer may inspect it.*

Proof. Start with the accepted carrier $B = (X, M, R, G, U, V, E, H, C, P, N)$. The source boundary request is attached to X only through the finite ray-window row R.

Read the metric and flow rows next. The row M supplies the displayed metric comparison, while G records the geodesic-flow transport attached to the same ray window.

Then consume the boundary rows U and V. The Busemann row and visual-boundary row are the only boundary-facing coordinates of the packet, and their scalar comparisons are sealed by E.

The replay row C records the route order, and H, P, N transport, source, and name the same rows. Hence every boundary consumer sees a finite ray-window handoff rather than a selected ideal representative or host compactification. □

Theorem 7447.4 (Hadamard boundary sibling nonescape)**.** *The sibling routes to* HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, *and* Real$^{\uparrow}$ *are*

consumed by HadamardBoundary$^{\uparrow}$ *only through the carrier rows* M, U, V, G, E *and the shared transport, replay, provenance, and local naming rows. Thus a Hadamard boundary read exports no global CAT(0) compactification, quotient ideal-boundary equality, selected geodesic ray, countable-choice selector, or ambient completeness theorem.*

Proof. Inspect the provenance row of Definition 7447.1. It names the sibling packets available to the boundary read, but each sibling appears as a displayed row in the same finite carrier.

Apply Definition 7447.3. It orders metric, flow, Busemann, visual-boundary, and Real-seal reads after the finite ray-window row and before any boundary consumer can inspect the packet.

The nonescape claim is then a carrier inventory. Since B contains no separate global compactification, quotient ideal-boundary equality, selected geodesic ray, choice selector, or completeness witness, none of those endpoints can be exported by a downstream consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardBoundary}^{\uparrow}}$.

Constructive story: A HadamardBoundary$^{\uparrow}$ packet is generated from a CAT(0)/Hadamard source surface, finite geodesic-ray windows, hyperbolic metric readback, geodesic-flow transport, Busemann boundary rows, visual-boundary rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HadamardBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 7447.1, Definition 7447.2, Definition 7447.3, and Definition 7447.4 seed the finite CAT(0)/Hadamard boundary route over geodesic-ray, Busemann, visual-boundary, metric, flow, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HadamardBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient ideal-boundary equality, selected geodesic rays, global compactification, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite ray-window visibility, metric readback, geodesic-flow transport, Busemann and visual-boundary exposure, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.HadamardBoundaryUp.

Chapter 7448

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ packet records the finite route by which a displayed Cauchy modulus, a finite stream schedule, dyadic tolerances, and a regular rational readback are sealed as a $\mathsf{Real}^{\uparrow}$ name. It sits beside the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the located modulus route of chapter 1035. The packet names a constructive limit certificate for the Real and Completion corridor without quotient streams, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7448.1 (Bishop Cauchy modulus limit carrier)**.** *A* $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, D, R, M, E, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row, listing the stream windows inspected by the modulus-guided read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, slack comparisons, and precision budgets used by the read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S and the tolerances in D.*
4. *M is the displayed Cauchy-modulus row, naming the finite threshold assignments that make the regular readback stable at the requested precisions.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after S, D, R, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream schedule, dyadic ledger, regular readback, modulus row, real seal, replay, provenance, and local name.*
7. *C records displayed* Cont *replay for precision refinements that remain inside the same finite packet.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, D, R, M, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7448.2 (Bishop Cauchy modulus limit NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ *carrier*

$$B = (S, D, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusLimit}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite stream schedule S, dyadic tolerance ledger D, regular rational readback R, displayed modulus row M, terminal real seal E, and structural rows H, C, P, N;*
2. *modulus coverage, requiring every precision request displayed in M to have a matching window in S, a matching dyadic tolerance in D, and a matching regular readback in R;*
3. *tolerance stability, requiring refinements in D to replay through C and transport through H without changing the displayed modulus row;*
4. *regular handoff, requiring R to be consumed before the terminal real seal E is read;*
5. *real-seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to export only the named seal and no host-level completion data.*

The obligation surface exports no quotient stream, selected limit, host real equality, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 7448.1. Rows S, D, R, M, E are the analytic route, and rows H, C, P, N provide transport, replay, source, and local naming.

For carrier admission and modulus coverage, read M against the finite stream schedule S, the dyadic ledger D, and the regular readback R. A precision request is accepted only when all three finite witnesses are present in the same BHist packet.

For tolerance stability, replay a refinement through C and transport the same refinement through H. The modulus row remains displayed, so the accepted packet does not switch to an external selector or a hidden completion principle.

For the regular handoff and the real seal, R must be consumed before E becomes visible. Since the carrier has no row for quotient streams, selected limits, host equality, trichotomy, choice, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7448.3 (Bishop Cauchy modulus regular handoff)**.** *Every regular rational read of an accepted* $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ *packet factors through the finite schedule, dyadic tolerance, and displayed modulus rows:*

$$S,\ D,\ M,\ R.$$

Consequently a $\mathsf{RegSeqRat}^{\uparrow}$ *consumer receives only the displayed readback certified by the finite packet and cannot infer a quotient stream, selected limit, countable-choice selector, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the handoff.*

Proof. Use the obligation surface of Definition 7448.2. Modulus coverage places every accepted precision request in M beside a stream window in S and a dyadic tolerance entry in D.

The regular handoff obligation then forces the $\mathsf{RegSeqRat}^{\uparrow}$ row R to be read after those finite witnesses. Transport and replay through H and C preserve the same rows, so the handoff cannot introduce an unlisted stream quotient or selector.

The excluded principles are absent from the carrier of Definition 7448.1. They therefore cannot be exported by a consumer whose only access is the displayed route S, D, M, R. □

Theorem 7448.4 (Bishop Cauchy modulus real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ *packet factors through*

$$S,\ D,\ R,\ M,\ E,\ H,\ C,\ P,\ N$$

before the terminal seal is consumed. The read exports the local $\mathsf{Real}^{\uparrow}$ *seal and no quotient-stream representative, selected host limit, host real equality, countable-choice schedule, trichotomy witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the NameCert obligations of Definition 7448.2. They place the terminal row E downstream of the finite stream schedule, dyadic tolerance ledger, regular rational readback, and displayed modulus row.

The surrounding anchors provide the route: chapter 126 supplies the finite schedule surface, chapter 291 supplies the tolerance ledger, chapter 301 supplies the regular readback, and chapter 38 supplies the terminal seal.

Rows H, C, P, N preserve that route by transport, replay, provenance, and local naming. Because no excluded host-level datum is a row of the carrier, a real-facing consumer cannot escape the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7448.1, Definition 7448.2, Definition 7448.3, and Definition 7448.4 seed the constructive Cauchy-modulus limit route from finite stream schedules and dyadic tolerances to the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus coverage, tolerance stability, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyModulusLimitUp.

Constructive story: A $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$ packet is generated from a StreamName finite schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a displayed Cauchy-modulus row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7449

A Concrete Naming Certificate for $\mathsf{TaggedPartition}^{\uparrow}$

Following Riemann, Henstock, and Kurzweil, the $\mathsf{TaggedPartition}^{\uparrow}$ packet records a finite interval subdivision whose cells carry displayed tags before any finite sum, gauge integral, or completion-facing read is exported. It sits after the dyadic cell discipline of chapter 1863, the interval-subdivision carrier of chapter 7313, and the gauge-window route of chapter 4180. The packet gives constructive integration consumers a finite tagged-subinterval interface without importing host finite sets, quotient partition equality, selected mesh limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7449.1 (Tagged partition carrier)**.** *A* $\mathsf{TaggedPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, B, S, E, L, D, G, H, C, P, N)$$

with the following visible rows:

1. *I is the* $\mathsf{RiemannPartition}^{\uparrow}$ *interval-subdivision row. It displays the ordered subinterval spine, adjacency ledger, endpoint coverage, and mesh-refinement data that every tagged read must preserve.*
2. *B is the boundary row for the inspected interval, with left and right endpoint names read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *witnesses.*
3. *S is the finite subinterval row. Each entry stores a left endpoint, a right endpoint, and the local inclusion route tying the cell back to I.*
4. *E is the endpoint exactness row. It records that consecutive subintervals meet at the displayed boundary and that the first and last endpoints match B.*
5. *L is the tag ledger. For each displayed subinterval it names one tag and the finite proof row that the tag lies in that same subinterval.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row used to read endpoint order, cell width, and mesh-window bounds through dyadic rational data.*
7. *G is the gauge-window row. It records the local radius request attached to each tag and the finite test that the tagged cell is contained in the displayed gauge window.*
8. *H records componentwise* hsame *transport for interval source, boundary, subintervals, endpoint exactness, tags, dyadic comparisons, gauge windows, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a tagged-partition request through $I, B, S, E, L, D, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{TaggedPartition}^{\uparrow}$, $\mathsf{RiemannPartition}^{\uparrow}$, $\mathsf{HenstockKurzweil}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TaggedPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, S, E, L, D, G, H, C, P, N$. There is no coordinate for host finite-set membership, quotient partition equality, an untagged mesh limit, countable choice, a selected refinement family, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7449.2 (Tagged partition NameCert obligations)**.** *For every accepted* $\mathsf{TaggedPartition}^{\uparrow}$ *carrier $T = (I, B, S, E, L, D, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TaggedPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, interval-subdivision exposure through I, boundary exposure through B, finite subinterval exposure through S, endpoint exactness through E, tag-in-cell exactness through L, dyadic endpoint and mesh comparison through D, gauge-window compatibility through G, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by a host finite set of cells, quotient equality of partitions, selected mesh limits, countable choice, a detached gauge refinement theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7449.1. The displayed rows are exactly $I, B, S, E, L, D, G, H, C, P, N$, so a local certificate can read only the interval subdivision, boundary, subinterval, endpoint, tag, dyadic, gauge-window, transport, replay, provenance, and naming data.

The subdivision side starts with I, B, S, E. The Riemann partition row supplies the ordered cell spine, while the boundary and endpoint rows force the displayed subintervals to remain attached to the same interval.

Next read L, D, G. The tag ledger attaches one visible tag to each displayed cell, the dyadic comparison row provides the endpoint and mesh comparisons, and the gauge-window row tests the same tagged cell against its local radius request.

Finally replay H, C, P, N. These structural rows preserve the same finite packet, and the carrier has no coordinate for the excluded host set, quotient, selected-limit, choice, detached-refinement, or ambient-completeness data. The listed rows therefore exhaust the local NameCert surface. □

Theorem 7449.3 (Tagged partition gauge-window handoff)**.** *Every* $\mathsf{HenstockKurzweil}^{\uparrow}$, *Riemann-sum, or completion-facing consumer of an accepted* $\mathsf{TaggedPartition}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{RiemannPartition}^{\uparrow} \longmapsto \mathsf{TaggedPartition}^{\uparrow} \longmapsto \mathsf{HenstockKurzweil}^{\uparrow}$$

only after reading the interval source I, boundary row B, subinterval row S, endpoint exactness row E, tag ledger L, dyadic comparison row D, gauge-window row G, transports H, replay C, provenance P, and local name N. The handoff exports a tagged gauge window for the displayed partition, not integrability, Darboux equality, selected refinements, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted tagged-partition read. By Definition 7449.2, the consumer first receives the interval-subdivision rows I, B, S, E and cannot replace them by a host list or quotient partition.

Read the tag and gauge side through L, D, G. The tag ledger binds each tag to the same displayed subinterval, the dyadic row supplies the endpoint and mesh comparisons, and the gauge-window row records the local radius containment test before the handoff is public.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. Since the carrier has no integrability, Darboux-equality, selected-refinement, or ambient-completeness coordinate, the exported route is exactly a local tagged gauge-window handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaggedPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{TaggedPartition}^{\uparrow}$ packet is generated from a RiemannPartition row, interval boundary rows, finite subinterval rows, endpoint exactness, tag-in-cell ledgers, DyadicRatCore comparisons, gauge-window rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaggedPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7449.1, Definition 7449.2, and Definition 7449.3 seed the finite tagged-subinterval interface from RiemannPartitionUp through TaggedPartitionUp to HenstockKurzweilUp gauge-window consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaggedPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim integrability, Darboux equality, host finite-set presentation, quotient partition equality, selected mesh limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, tag-in-cell exactness, dyadic comparison, gauge-window compatibility, consumer handoff, transport, replay, provenance, and local naming for BEDC.Derived.TaggedPartitionUp.

Chapter 7450

A Concrete Naming Certificate for $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$

Following Bishop and Bridges on constructive uniform convergence and regular real completions, the $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$ packet records the finite located-modulus route by which a uniformly Cauchy family yields a Real-facing limit certificate. It sits after the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the uniform Cauchy limit packet of chapter 3980, and the completion-modulus handoff of chapter 7448. The packet names a located limit route with an explicit common modulus; it does not assert quotient equality of limits, select a host real, use countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7450.1 (Located uniform Cauchy limit carrier)**.** *A* $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (U, M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *source row, including the inspected function-window family and the uniform Cauchy comparison ledger.*
2. *M is the located common modulus row, read through the completion-modulus surface and binding each tolerance request to a finite tail threshold before any limit seal is exposed.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row selected by the same tolerance and modulus request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and tail threshold.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that compares the requested radius, the selected threshold, and the displayed readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the located limit row, available only after U, M, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the uniform Cauchy source, located modulus, windows, readback, dyadic ledger, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located tolerance request through U, M, S, R, D and then to E.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$, $\mathsf{UniformCauchyLimit}^{\uparrow}$, $\mathsf{BishopCauchyModulusLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host limit selection, quotient equality of functions or reals, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyLimitUp.LocatedUniformCauchyLimitTasteGate single carrier alignment`

Theorem 7450.2 (Located uniform Cauchy limit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$ *carrier* $L = (U, M, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyLimit}^{\uparrow}}$ *surface consists exactly of carrier admission, located-modulus exposure, finite-window readback, dyadic tolerance accounting, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming for the rows of Definition 7450.1. Every completion-facing consumer must read* U *and* M *before* S, R, D, E *are available, and no obligation exports quotient limits, selected host reals, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 7450.1. The accepted packet displays the uniform Cauchy source row, located common modulus, finite windows, regular readback, dyadic tolerance ledger, Real seal, and structural rows.

The located-modulus obligation is the first analytic route. A tolerance request reads the uniform Cauchy source U and then the common modulus row M, so the later window and readback rows are tied to a displayed threshold rather than to a host-selected limit.

Finite-window readback and dyadic accounting then proceed through S, R, D. The stream window is opened at the selected threshold, the regular rational readback is extracted from that window, and the dyadic ledger records the tolerance comparison before the terminal Real seal can be consumed.

Finally, H, C, P, N transport, replay, source, and name only the displayed packet. Since quotient limits, selected host reals, countable choice, open-cover compactness, and ambient completeness are absent from the carrier, no local obligation can export them. □

Theorem 7450.3 (Located uniform Cauchy limit completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{UniformCauchyLimit}^{\uparrow} \longmapsto \mathsf{BishopCauchyModulusLimit}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto$$

The handoff supplies a located modulus, finite windows, regular rational readback, dyadic tolerance evidence, terminal Real sealing, componentwise transport, replay, provenance, and local naming; it supplies no quotient completion, selected representative, host equality of limits, countable-choice schedule, or ambient completeness theorem.

Proof. Open the carrier of Definition 7450.1. Its analytic rows are U, M, S, R, D, E, and the structural rows H, C, P, N preserve only that displayed order.

Apply the NameCert obligations above. They force the uniform Cauchy source and located modulus to be read before the finite window, regular readback, dyadic tolerance, and Real-seal rows are available.

The stated route is exactly the row order of the packet. Because the refused objects have no carrier coordinate and no provenance row, every completion-facing read remains a finite located-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyLimit}^{\uparrow}$ packet is generated from a UniformCauchyLimit row, a BishopCauchyModulusLimit row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7450.1, Definition 7450.2, and Definition 7450.3 seed the located uniform Cauchy limit route over common moduli, finite windows, regular readback, dyadic tolerance accounting, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of limits, selected host reals, countable choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-modulus exposure, finite-window readback, dyadic accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyLimitUp.

Chapter 7451

A Concrete Naming Certificate for $\mathsf{RealLocatedInfimum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLocatedInfimum}^{\uparrow}$ packet names the lower-bound dual of a located real-order construction as finite BEDC data. It sits after the terminal real-name seal of chapter 38, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the modulus neighbor of chapter 6478. The packet gives Real and Completion consumers a displayed lower-bound infimum route without claiming Dedekind cuts, trichotomy, quotient equality, selected infima, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7451.1 (Real located infimum carrier)**.** *A* $\mathsf{RealLocatedInfimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, D, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite* $\mathsf{Real}^{\uparrow}$ *lower-bound-candidate row, listing the inspected real-name lower surfaces and the comparison observations that certify them as lower-bound candidates in the current window.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation-window row, recording dyadic precisions, lower-window widths, and the finite tolerance comparisons used to read the candidates.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the dyadic windows and candidate comparisons as regular rational observations before the terminal real seal is consumed.*
4. *M is the consumed* $\mathsf{LocatedInfimumModulus}^{\uparrow}$ *row, supplying the displayed modulus discipline that turns lower-window requests into finite dyadic approximation obligations.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the lower-bound candidates, dyadic windows, regular readback, and located-infimum modulus have all been displayed.*
6. *H records componentwise* hsame *transport for B, D, R, M, E, C, P, N.*
7. *C records displayed* Cont *replay for lower-bound-candidate reads, dyadic approximation reads, regular rational readback, modulus replay, real-seal access, and transported consumer return.*
8. *P is the* Pkg *provenance over* $\mathsf{RealLocatedInfimum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInfimumModulus}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocatedInfimum}^{\uparrow}}$ *row.*

The carrier exposes only the finite route $B, D, R, M, E, H, C, P, N$. It has no coordinate for Dedekind cuts, trichotomy, quotient equality of real names, a host infimum operator, countable choice, selected witnesses from completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7451.2 (Real located infimum NameCert obligations)**.** *For every accepted* $\mathsf{RealLocatedInfimum}^{\uparrow}$ *carrier*

$$J = (B, D, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLocatedInfimum}^{\uparrow}}$ *surface consists exactly of Real lower-bound candidate exposure through B, dyadic approximation-window exactness through D, regular rational readback through R, located-infimum modulus replay through M, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7451.1. The analytic rows are exactly the Real lower-bound-candidate row, dyadic approximation-window row, regular rational readback row, located-infimum modulus row, and terminal real seal row.

The lower-bound read is forced through B, D, R, M. Candidate observations begin as displayed $\mathsf{Real}^{\uparrow}$ lower surfaces, are inspected only at the dyadic windows stored in D, are repacked by R, and are governed by the modulus route M.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no row stores Dedekind cuts, trichotomy, quotient equality, host infimum selection, countable choice, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7451.3 (Real located infimum L10 handoff)**.** *Every Real or Completion consumer read exported by an accepted* $\mathsf{RealLocatedInfimum}^{\uparrow}$ *packet factors through the displayed L10 route*

$$B, D \longmapsto R \longmapsto M \longmapsto E \longmapsto C, P, N.$$

Thus the packet hands off lower-bound infimum data only as regular rational readback plus a located-infimum modulus and a terminal $\mathsf{Real}^{\uparrow}$ *seal, not as Dedekind-cut data, trichotomy, quotient equality, selected infima, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real or Completion-facing request. The carrier contains one terminal real seal row E, and the definition makes it reachable only after the lower-bound candidates, dyadic windows, regular readback, and modulus row have been displayed.

The request first reads B through the dyadic approximation windows in D. The row R then turns those finite observations into regular rational readback, while M supplies the located-infimum modulus discipline that keeps the lower-window request finite.

Finally C, P, N replay, source, and name the same route before the consumer sees E. No step has a coordinate for Dedekind cuts, trichotomy, quotient equality, selected infima, or ambient completeness, so the L10 handoff remains inside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocatedInfimum}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealLocatedInfimum}^{\uparrow}}$, seed)

Scope closed: Definition 7451.1, Definition 7451.2, and Definition 7451.3 seed the finite lower-bound located-infimum route over Real lower-bound candidates, dyadic approximation windows, RegSeqRat readback, LocatedInfimumModulus replay, and a terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealLocatedInfimum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind cuts, trichotomy, quotient real equality, selected infima, countable choice, host infimum operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Real lower-bound candidate exposure, dyadic approximation-window exactness, RegSeqRat readback, LocatedInfimumModulus replay, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealLocatedInfimumUp.

Constructive story: A RealLocatedInfimum$^{\uparrow}$ packet is generated from Real lower-bound candidates, dyadic approximation windows, RegSeqRat readback, LocatedInfimumModulus replay, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 7452

A Concrete Naming Certificate for $\mathsf{ProbeBundleCoverFold}^{\uparrow}$

The $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ packet realizes the finite cover-fold branch of chapter 9237. It records the displayed step that folds compact finite-cover probes, local pointwise-modulus rows, and rational lower-bound ledgers into a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff before any compact-uniform-continuity consumer reads the route. The packet sits beside the finite minimum fold of chapter 619, the compact-net selector of chapter 914, the compact uniform-continuity carrier of chapter 388, the compact source of chapter 60, the metric source of chapter 4230, the continuous-map source of chapter 130, and the uniform-modulus consumer of chapter 369. It is a finite $\mathsf{ProbeBundle}$ cover fold, not an open-cover compactness theorem, selected subcover, host modulus oracle, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7452.1 (ProbeBundle cover-fold carrier)**.** *A* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, G, B, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-cover or finite-net source row.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to test membership in the displayed cover cells.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to each retained probe.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *of centers, cover cells, local radii, and pointwise modulus witnesses.*
5. *R is the rational positive-radius ledger folded across the displayed bundle.*
6. *L is the lower-bound output row, compatible with* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *and the* $\mathsf{CompactRadiusChain}^{\uparrow}$ *cover-chain ledger.*
7. *U is the tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row read by compact-uniform-continuity consumers.*
8. *H records componentwise* hsame *transport for $K, M, G, B, R, L, U, C, P, N$.*
9. *C records displayed* Cont *replay from compact cover rows through metric tests, pointwise modulus rows, finite bundle folding, lower-bound output, and uniform-modulus handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{CompactRadiusChain}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProbeBundleCoverFold}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers outside the displayed bundle, a host modulus function, quotient point equality, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Theorem 7452.2 (ProbeBundle cover-fold obligations)**.** *For every accepted* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ *carrier* $F = (K, M, G, B, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProbeBundleCoverFold}^{\uparrow}}$ *surface consists exactly of compact finite-cover exposure, metric membership exposure, continuous pointwise-modulus exposure, finite* $\mathsf{ProbeBundle}$ *exposure, rational radius-ledger exposure, lower-bound fold output, tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7452.1. The compact source row K, metric row M, pointwise-modulus row G, finite bundle row B, rational radius ledger R, lower-bound output L, and uniform-modulus row U are the whole mathematical surface.

The fold is finite because B is a displayed $\mathsf{ProbeBundle}$ and R is read only over the probes retained in that bundle. The lower-bound row L is therefore a fold output rather than an external minimum.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier has no open-cover theorem, selected subcover, host modulus, quotient equality, choice, completeness, Lean theorem, or bridge coordinate, the stated rows exhaust the local NameCert obligations. □

Theorem 7452.3 (ProbeBundle cover-fold uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{UniformModulus}^{\uparrow}$ *row from an accepted* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ *packet must first read*

$$\mathsf{Compact}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{ProbeBundle} \longmapsto \mathsf{FiniteNetMinimumFold}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}$$

through $K, M, G, B, R, L, U, H, C, P, N$*. The handoff is a finite cover fold over displayed probes; it is not a selected global modulus, open-cover compactness theorem, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Start with Definition 7452.2. It forces compact-cover data, metric membership tests, pointwise modulus rows, the finite probe bundle, and the rational radius ledger to be displayed before the lower-bound row and the uniform-modulus handoff can be consumed.

The lower-bound step is the finite fold layer. It is compatible with the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ and $\mathsf{CompactNetModulusSelector}^{\uparrow}$ siblings because the consumer sees the same finite probe inventory and the same rational radius ledger before it sees U.

The rows H, C, P, N preserve the route as a displayed BEDC packet. No excluded host principle is stored in the carrier, so the consumer receives only the tolerance-only handoff generated by the finite cover fold. □

Theorem 7452.4 (ProbeBundle cover-fold compact-radius route)**.** *Every* $\mathsf{CompactRadiusChain}^{\uparrow}$ *or* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ *carrier must read the same* K, M, G, B, R, L, U *route before the tolerance-only handoff is public. The compact radius chain is therefore a displayed ledger over the finite probe bundle, not an implicit compactness theorem or a selected cover refinement outside the packet.*

Proof. Project $F = (K, M, G, B, R, L, U, H, C, P, N)$. The compact source, metric tests, continuous-map local moduli, finite bundle, radius ledger, lower-bound output, and uniform-modulus row appear as separate coordinates.

Apply Definition 7452.3. It shows that the uniform-modulus read passes through the finite fold before any compact-uniform-continuity consumer can use it.

The compact radius chain can therefore be read only as provenance for the same finite route. Transport, replay, provenance, and local naming preserve the route but do not add a selected cover refinement or background compactness principle. □

Falsifiable prediction: If the finite ProbeBundle cover-fold row is removed, the displayed route cannot derive a tolerance-only UniformModulus$^{\uparrow}$ handoff from compact finite-cover rows, pointwise modulus rows, and rational lower-bound ledgers inside the same carrier.

Independence witness: finitenetminimumfold, compactnetmodulusselector, uniformmodulus, compactradiuschain: no listed sibling is bijective to ProbeBundleCoverFold$^{\uparrow}$ because this packet jointly carries compact cover rows, metric membership tests, pointwise modulus rows, the finite ProbeBundle inventory, rational radius ledgers, lower-bound fold output, tolerance-only UniformModulus handoff, transport, replay, provenance, and local naming.

The Lean TasteGate file for ProbeBundleCoverFold$^{\uparrow}$ must provide explicit Nontrivial and FieldFaithful instances before a checked paper marker can cite a carrier-level TasteGate target.

Closure status for NameCert$_{\text{ProbeBundleCoverFold}^{\uparrow}}$.

Constructive story: A ProbeBundleCoverFold$^{\uparrow}$ packet is generated from Compact finite-cover rows, MetricSpace membership rows, ContinuousMap pointwise-modulus rows, finite ProbeBundle rows, rational radius ledgers, lower-bound fold output, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ProbeBundleCoverFold}^{\uparrow}}$, seed)

Scope closed: Definition 7452.1, Definition 7452.2, Definition 7452.3, and Definition 7452.4 seed the finite cover-fold route from compact probes to tolerance-only uniform-modulus data.

Formal verification: FormalStatus(NameCert$_{\text{ProbeBundleCoverFold}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host modulus functions, quotient point equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite ProbeBundle exposure, rational fold exactness, compact-radius routing, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.ProbeBundleCoverF

Chapter 7453

A Concrete Naming Certificate for $\mathsf{RegularCauchyVariationBound}^{\uparrow}$

Following Jordan's finite variation ledgers and the Bishop–Bridges treatment of regular Cauchy real names, the $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ packet records a displayed variation budget for a regular rational Cauchy stream before the terminal real-name seal is consumed. It sits after the regular sequence surface of chapter 301, the dyadic tolerance arithmetic of chapter 291, the finite stream-window interface of chapter 126, the Cauchy modulus route of chapter 292, and the real-name boundary of chapter 38. The packet names only a finite variation-budget route; it does not import host sequence equality, a selected limit representative, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7453.1 (Regular Cauchy variation-bound carrier)**.** *A* $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S, Q, D, B, M, E, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of finite windows inspected from the regular Cauchy name.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reading the same windows as regular rational observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row of adjacent dyadic difference cells and finite partial variation sums.*
4. *B is the displayed dyadic variation-budget row bounding every inspected partial variation sum.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff row extracted from the displayed variation budget and regularity data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, Q, D, B, M have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, dyadic, budget, modulus, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from stream windows to regular rational readbacks, dyadic variation sums, the modulus handoff, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyVariationBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyVariationBound}^{\uparrow}}$ *row.*

The classifier compares carriers only through $S, Q, D, B, M, E, H, C, P, N$. It has no coordinate for host sequence equality, quotient real equality, a chosen limit representative, unbounded variation search, countable choice, or ambient real completeness.

Theorem 7453.2 (Regular Cauchy variation-bound NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ *carrier* $V = (S, Q, D, B, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyVariationBoun}}$ *surface consists exactly of stream-window exposure, regular rational readback, dyadic adjacent-difference exposure, finite variation-budget exposure, Cauchy-modulus handoff, real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7453.1. The rows S, Q, D, B, M, E are the only mathematical rows visible before the structural transport, replay, provenance, and naming rows.

The stream and regular rational rows determine which finite observations are being compared. The dyadic row D records adjacent differences and finite partial variation sums, while B records the displayed budget bounding those sums.

The modulus row M is downstream of the same data, and the real seal E is downstream of M. The rows H, C, P, N preserve, replay, source, and name that finite route.

Since the carrier has no host equality, quotient equality, selected-limit, unbounded-search, choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7453.3 (Regular Cauchy variation-bound Cauchy handoff)**.** *Every* $\mathsf{CauchyModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ *packet must first read the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow}$$

through the same displayed rows $S, Q, D, B, M, H, C, P, N$. The handoff is a finite tail-control obligation extracted from the variation budget, not a selected limit, quotient equality of names, or ambient completeness principle.

Proof. Start with the obligation surface of Definition 7453.2. It places stream windows, regular rational readbacks, dyadic adjacent-difference rows, and the displayed variation budget before the modulus row.

The row M is therefore read only after S, Q, D, B have been displayed. The replay row C records the same order, and the transport row H keeps the handoff componentwise stable.

The rows P, N source and name the route. Because no carrier row stores a chosen limit, quotient equality, or ambient completeness claim, the consumer receives only the finite Cauchy-modulus handoff. □

Theorem 7453.4 (Regular Cauchy variation-bound Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ *packet must read the stream windows, regular rational readbacks, dyadic variation sums, displayed variation budget, and* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff before the seal row E is available. Thus the real-facing boundary is the finite variation-budget packet, not host real equality, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $V = (S, Q, D, B, M, E, H, C, P, N)$. The seal row E appears only after the stream, regular rational, dyadic, budget, and modulus rows in the displayed packet.

Apply Definition 7453.3. A completion-facing read of the modulus already factors through the finite variation-budget route before it can reach a real-name consumer.

Finally inspect H, C, P, N. They transport, replay, source, and name the same route without adding another real equality or completion coordinate. The stated exclusions are therefore unavailable inside the accepted packet. □

Falsifiable prediction: A $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ consumer is refuted if it reaches a $\mathsf{CauchyModulus}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without displaying the stream windows, regular rational readback, dyadic adjacent-difference ledger, finite variation budget, transport, replay, provenance, and local NameCert rows.

Independence witness: regseqrat, dyadicratcore, streamname, cauchymodulus, real: no listed sibling is bijective to $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ because this packet jointly binds finite windows, regular rational observations, dyadic variation sums, a displayed budget, Cauchy-modulus handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyVariationBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyVariationBound}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular rational readbacks, DyadicRatCore adjacent-difference ledgers, a displayed dyadic variation budget, a CauchyModulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyVariationBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7453.1, Definition 7453.2, Definition 7453.3, and Definition 7453.4 seed the finite variation-budget route over RegSeqRat, DyadicRatCore, StreamName, CauchyModulus, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyVariationBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence equality, quotient real equality, selected limit representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-budget exactness, Cauchy-modulus handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyVariationBoundUp.

Chapter 7454

A Concrete Naming Certificate for $\mathsf{DetachableSubset}^{\uparrow}$

Following Bishop and Bridges on constructive subsets with decidable membership, the $\mathsf{DetachableSubset}^{\uparrow}$ packet records the finite BEDC surface by which a displayed subset may be read through a Boolean membership classifier. It sits beside the Boolean carrier of chapter 32, the set-like naming surface of chapter 3668, the semidecidable-open classifier of chapter 6464, the finite stream-name route of chapter 126, and the terminal real-name seal of chapter 38. The packet supplies a reusable finite decidable-membership carrier for constructive subset reads without importing host powersets, excluded middle, quotient subset equality, selected witnesses, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7454.1 (DetachableSubset carrier)**.** *A* $\mathsf{DetachableSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, M, Q, B, S, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{SetLike}^{\uparrow}$ *ambient carrier row whose elements are allowed to be inspected;*
2. *M is the detachable membership row, returning a displayed* $\mathsf{Bool}^{\uparrow}$ *answer for each inspected element row;*
3. *Q is the complementary refusal row, recording the negative membership answer when M returns false;*
4. *B is the* $\mathsf{Bool}^{\uparrow}$ *decision boundary through which both positive and negative membership answers are read;*
5. *S is the* $\mathsf{SemidecidableOpen}^{\uparrow}$ *exposure row available to open-facing consumers after the Boolean decision has been displayed;*
6. *R is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which sequential or real-facing consumers request inspected elements;*
7. *H records componentwise* hsame *transport for the ambient, membership, refusal, Boolean, semidecidable-open, and stream rows;*
8. *C records displayed* Cont *replay of the same finite route;*
9. *P is the* Pkg *provenance over* $\mathsf{DetachableSubset}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{SemidecidableOpen}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{DetachableSubset}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host powerset, an external predicate with excluded middle, quotient subset equality, selected element witness, or ambient real-completion principle.

Lean4 (definition) `BEDC.Derived.DetachableSubsetUp`

Theorem 7454.2 (DetachableSubset NameCert obligations)**.** *For every accepted* $\mathsf{DetachableSubset}^{\uparrow}$ *carrier*

$$D = (A, M, Q, B, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DetachableSubset}^{\uparrow}}$ *surface consists exactly of carrier admission for those rows, ambient set-like exposure through A, detachable membership through M, negative-membership refusal through Q, Boolean readback through B, semidecidable-open exposure through S, finite StreamName scheduling through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7454.1. The projection lists exactly the ambient, membership, refusal, Boolean, semidecidable-open, stream-window, transport, replay, provenance, and naming rows.

The subset is detachable because each inspected element reaches the membership row M only through the displayed ambient row A and receives its answer through the Boolean boundary B. The refusal row Q is part of the same packet, so a negative answer is not supplied by an external excluded-middle principle.

The downstream handoff is finite. Open-facing consumers may read S only after the Boolean membership boundary has been displayed, and sequential or real-facing consumers may request inspected elements only through R, H, C, P, N. Thus the local NameCert surface is the displayed decidable-membership route itself. □

Theorem 7454.3 (DetachableSubset finite membership route)**.** *Every membership answer exported by an accepted* $\mathsf{DetachableSubset}^{\uparrow}$ *packet is attached to a displayed element row in A and a displayed Boolean row in B. If two accepted packets agree componentwise on A, M, Q, B, S, R under the displayed* hsame *transport, then every downstream consumer sees the same accepted-or-refused membership answer.*

Proof. Begin with the ambient carrier row A. It is the only source from which an inspected element may be named, so a consumer cannot ask the detachable subset about an element outside the displayed carrier route.

The membership and refusal rows M, Q are read through the Boolean boundary B. Componentwise hsame transport therefore identifies the visible element and its visible truth value together.

Replay through C and provenance through P, N preserve the same finite route. Hence transported packets expose the same membership answer to open-facing, stream-facing, and real-facing consumers. □

Theorem 7454.4 (DetachableSubset obligation package)**.** *The carrier, NameCert, and finite membership route clauses assemble the seed obligation package for* $\mathsf{DetachableSubset}^{\uparrow}$*. The package is exhausted by the rows*

$$A, M, Q, B, S, R, H, C, P, N$$

and by the sibling route through $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{SemidecidableOpen}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. The carrier definition supplies exactly the listed rows, and Definition 7454.2 makes them the local NameCert surface.

Definition 7454.3 binds each membership answer to a displayed ambient element and Boolean decision boundary. The semidecidable-open and StreamName rows can only consume that displayed boundary; they do not add a host powerset or a choice principle.

Thus the package names the finite decidable-membership surface needed by constructive subset consumers while refusing excluded middle, quotient subset equality, selected witnesses, host powersets, and ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DetachableSubset}^{\uparrow}}$.

Constructive story: A $\mathsf{DetachableSubset}^{\uparrow}$ packet is generated from a SetLikeUp ambient row, detachable membership and refusal rows, a BoolUp decision boundary, a SemidecidableOpenUp exposure row, a StreamNameUp finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DetachableSubset}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7454.1, Definition 7454.2, Definition 7454.3, and Definition 7454.4 seed the finite decidable-membership route for constructive subset consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DetachableSubset}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.DetachableSubsetUp`

Bridge status: none

Not claimed: This chapter does not claim host powersets, excluded middle, quotient subset equality, selected element witnesses, ambient Real completeness, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier and NameCert obligations for BEDC.Derived.DetachableSubsetUp.

Chapter 7455

A Concrete Naming Certificate for $\mathsf{BishopModulusConvergence}^{\uparrow}$

Following Bishop and Bridges on convergence with displayed moduli, the $\mathsf{BishopModulusConvergence}^{\uparrow}$ packet records the finite route by which a stream of rational approximants is read with a Bishop modulus before any real-facing consumer sees a limit seal. It sits after the finite observation discipline of chapter 126, the dyadic tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the displayed limit handoff of chapter 7448. The packet names convergence through finite rows only; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7455.1 (Bishop modulus convergence carrier)**.** *A* $\mathsf{BishopModulusConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(S, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row scheduling the finite observation windows of the approximating sequence.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic requests and arithmetic comparisons for the convergence budget.*
3. *M is the Bishop modulus row: for each displayed tolerance request it supplies a finite stage bound, together with the statement that all later scheduled windows lie inside the requested dyadic tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S, D, and M.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the stream schedule, dyadic tolerance row, modulus row, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the stream, dyadic, modulus, regular-readback, real-seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from tolerance request through finite window inspection, modulus-bound lookup, regular rational readback, and real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopModulusConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopModulusConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopModulusConvergenceUp.BishopModulusConvergenceTasteGate single carrier alignment`

Theorem 7455.2 (BishopModulusConvergence NameCert obligations)**.** *For an accepted* BishopModulusConvergenc *carrier* $(S, D, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopModulusConvergence}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the stream schedule, dyadic tolerance row, Bishop modulus row, regular rational readback, real seal, componentwise transport, continuation replay, provenance, and local naming row;*
2. *tolerance discipline, requiring every convergence request to enter through D before a stage bound in M is read;*
3. *modulus exactness, requiring the bound supplied by M to control all later windows exposed by S at the requested tolerance;*
4. *regular readback, requiring R to be obtained from the displayed stream, dyadic, and modulus rows before the* $\mathsf{Real}^{\uparrow}$ *seal E is public;*
5. *transport and replay, requiring H and C to preserve the same ordered route under* hsame *transport and displayed* Cont *replay;*
6. *ledger non-escape, excluding quotient streams, selected host limits, countable choice, trichotomy, host real equality, and ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted packet. The carrier inventory is exactly the stream schedule S, dyadic tolerance row D, Bishop modulus row M, regular readback R, real seal E, and structural rows H, C, P, N.

Tolerance discipline follows from the order of the packet. A convergence request first becomes a dyadic row in D, and only then can the displayed modulus row M supply the finite stage bound.

Modulus exactness is the content of M: after the bound is read, every later scheduled window supplied by S remains within the requested dyadic tolerance. The regular readback row R packages that controlled tail as a $\mathsf{RegSeqRat}^{\uparrow}$ name before E seals it as a $\mathsf{Real}^{\uparrow}$ row.

The rows H and C transport and replay the same route, while P and N record provenance and local naming. Since no remaining coordinate stores quotient stream equality, a selected host limit, countable choice, trichotomy, host equality, or ambient completeness, those data cannot be projected from the packet. □

Theorem 7455.3 (BishopModulusConvergence regular handoff)**.** *Every accepted* $\mathsf{BishopModulusConvergence}^{\uparrow}$ *packet hands its convergence data to regular rational and real-name consumers only through the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The Bishop modulus row supplies the finite stage bound used on this route, and the handoff remains stable under the displayed hsame *transport and* Cont *replay rows. No consumer can bypass the regular readback by asking for a quotient stream, selected host limit, countable-choice selector, trichotomy split, host real equality, or ambient completeness witness.*

Proof. Read a consumer request through the continuation ledger C. The first public data are the finite stream windows in S and the dyadic tolerance request in D.

The modulus row M then returns a stage bound for that request. This bound controls the later stream windows, so the controlled tail is available to R as regular rational readback rather than as an external limit selection.

Finally the real-facing row E is read only after R is present. Componentwise hsame transport in H moves the same stream, dyadic, modulus, regular-readback, and real-seal rows together, so the handoff remains a displayed finite route. The excluded host and quotient data are absent from the carrier inventory. □

Theorem 7455.4 (BishopModulusConvergence tail transport)**.** *For every accepted* BishopModulusConvergence$^{\uparrow}$ *packet* $(S, D, M, R, E, H, C, P, N)$*, each tail-bound supplied by the Bishop modulus row* M *remains valid after componentwise* hsame *transport through* H *and displayed* Cont *replay through* C*. Consequently the* RegSeqRat$^{\uparrow}$ *readback* R *and the* Real$^{\uparrow}$ *seal* E *receive the same tolerance-controlled tail that was certified before transport.*

No quotient stream, selected limit, countable-choice selector, trichotomy branch, or ambient completeness witness is introduced by this transport route.

Proof. Project the transport and replay rows H and C from the accepted carrier. The transport row is componentwise, so it moves the stream schedule, dyadic tolerance row, modulus row, regular readback, and real seal together rather than replacing the tail certificate by an external equality.

Read the analytic rows in the carrier order S, D, M, R, E. A tolerance request is first registered in D, then the modulus row M supplies the finite tail-bound for all later windows of the stream schedule S.

Replay through C preserves the same request-to-bound route. Since H transports the same rows componentwise, the tail-bound still controls the later scheduled windows after transport, and R receives the identical controlled tail before E is exposed.

The carrier has no coordinate for quotient streams, selected limits, countable-choice selectors, trichotomy, or ambient completeness. The transported theorem therefore remains a finite tail handoff through S, D, M, R, E, H, C. □

Closure status for NameCert$_{\text{BishopModulusConvergence}\uparrow}$.

Theory closure: TheoryGate(NameCert$_{\text{BishopModulusConvergence}\uparrow}$, seed)

Scope closed: Definition 7455.1, Definition 7455.2, and Definition 7455.3 seed the convergence-modulus packet through StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{BishopModulusConvergence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance discipline, modulus exactness, regular readback, transport replay, provenance, and non-escape for BEDC.Derived.BishopModulusConvergenceUp.

Constructive story: A BishopModulusConvergence$^{\uparrow}$ packet is generated from a StreamName schedule, DyadicRatCore tolerance row, Bishop modulus row, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7456

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusCertificate}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive compact analysis, the $\mathsf{CompactUniformModulusCertificate}^{\uparrow}$ packet records the finite-net route by which compactness and pointwise continuity produce a tolerance-only uniform modulus. It realizes the compact uniform-continuity vision of chapter 9237 as concrete NameCert data, sitting over the compact source of chapter 60, the continuous-map rows of chapter 130, the source and target metric rows of chapter 57, the rational lower-bound fold of chapter 37, the downstream uniform-modulus consumer of chapter 369, and the terminal real seal of chapter 38. The packet names the finite proof object; it is not open-cover compactness, a host continuity theorem, quotient function equality, a selected external modulus, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7456.1 (Compact modulus fan carrier)**.** *A* $\mathsf{CompactUniformModulusCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(K, F, L, Q, S, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, exposing bundled centers and coverage witnesses from* $\mathsf{Compact}^{\uparrow}$*.*
2. *F is the continuous-map row, exposing pointwise modulus ledgers from* $\mathsf{ContinuousMap}^{\uparrow}$ *over the same source metric.*
3. *L is the local pointwise modulus ledger: each bundled center carries a positive rational radius and the target-tolerance row that the continuous-map certificate supplies there.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, obtained by primitive recursion over the compact finite net and certified below every local radius.*
5. *S is the source* $\mathsf{Met}^{\uparrow}$ *triangle row, used to move an arbitrary source point and its nearby neighbour into the same local ball around a bundled center.*
6. *T is the target* $\mathsf{Met}^{\uparrow}$ *triangle row, used to combine the two pointwise target estimates into the requested output tolerance.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, reading only the tolerance request and the folded precision from Q.*

8. *H records componentwise* hsame *transport for the compact net, continuous-map row, pointwise ledgers, rational fold, source and target metric rows, uniform-modulus row, provenance, and local naming row.*

9. *C records displayed* Cont *replay from tolerance request through finite-net coverage, pointwise modulus lookup, rational lower-bound fold, source triangle, target triangle, and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusCertificate}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, host function extensionality, quotient map equality, a selected external modulus, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusCertificateUp.CompactUniformModulusCertificateTasteGate`
`single carrier alignment`

Theorem 7456.2 (Compact modulus fan NameCert obligations)**.** *For an accepted* CompactUniformModulusCertific *carrier* $(K, F, L, Q, S, T, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCertificate}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for compact finite-net rows, continuous-map rows, pointwise modulus ledgers, rational lower-bound fold rows, source and target metric triangle rows, uniform-modulus consumer rows, transport, continuation, provenance, and local naming;*

2. *finite-net coverage, requiring every source-point request to choose a displayed bundled center through K rather than through an external selector;*

3. *pointwise-modulus lookup, requiring each bundled center to read its local radius and target-tolerance row through F and L;*

4. *rational-fold exactness, requiring Q to be certified below every local radius by* $\mathsf{Rat}^{\uparrow}$ *comparisons;*

5. *source-target triangle handoff, requiring S and T to turn finite-net coverage plus the folded radius into the final uniform target estimate;*

6. *uniform-modulus exposure, requiring U to expose only the tolerance request and folded precision as the public* $\mathsf{UniformModulus}^{\uparrow}$ *row;*

7. *ledger non-escape, excluding open-cover compactness, host function extensionality, quotient map equality, selected external moduli, countable choice, and ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier. Its public coordinates are exactly the compact finite-net row K, continuous-map row F, local modulus ledger L, rational fold Q, source and target metric rows S, T, uniform-modulus row U, and structural rows H, C, P, N.

Finite-net coverage is read from K. Each source point is routed to a bundled center that is already part of the finite compact packet, so no external center selector is introduced.

Pointwise-modulus lookup uses F and L at each bundled center. The local radius and target-tolerance estimate are stored as displayed rows, and the lower-bound fold Q is obtained by primitive recursion over those finitely many rational radius rows.

The source metric row S moves a nearby pair of source points into the same local ball around the selected center. The target metric row T then combines the two pointwise estimates, and U exposes the resulting tolerance-only precision as the uniform-modulus consumer row.

The transport, continuation, provenance, and local naming rows preserve this same route. Since no carrier coordinate stores open-cover compactness, host extensionality, quotient equality, an external modulus selector, countable choice, or ambient real completeness, those data cannot be read from the packet. □

Theorem 7456.3 (Compact modulus fan uniform-continuity handoff)**.** *Every accepted* CompactUniformModulusCe *packet hands compact-uniform-continuity data to* $\mathsf{UniformModulus}^{\uparrow}$ *consumers by the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{UniformModulus}^{\uparrow}.$$

The $\mathsf{Real}^{\uparrow}$ *row is consumed only as the terminal scalar seal for the displayed metric and tolerance estimates. The handoff remains stable under the componentwise* hsame *transport and* Cont *replay rows, and no consumer can bypass it by asking for open-cover compactness, host function extensionality, quotient map equality, selected external moduli, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a tolerance request for the uniform-modulus consumer. The compact finite-net row K supplies the bundled centers and coverage rows, while F and L supply the pointwise modulus data attached to those centers.

Use the $\mathsf{Rat}^{\uparrow}$ fold row Q to obtain one precision below every local radius. This is a finite recursive read over the displayed net, not a selection from an ambient family of radii.

Apply the source metric triangle row S to place both source points in the same local modulus ball. Apply the target metric triangle row T to combine the two pointwise estimates into the requested target tolerance, and then expose the result through U as a $\mathsf{UniformModulus}^{\uparrow}$ row.

The real scalar row is sealed only at the end of this displayed estimate route. Componentwise transport in H and replay in C preserve the same carrier coordinates, so the excluded host, quotient, choice, and completeness data remain unavailable. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCertificate}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7456.1, Definition 7456.2, and Definition 7456.3 seed the finite-net, pointwise-modulus, rational-fold, metric-triangle, and uniform-modulus handoff packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host function extensionality, quotient map equality, selected external moduli, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-net coverage, pointwise-modulus lookup, rational-fold exactness, source-target triangle handoff, uniform-modulus exposure, transport replay, provenance, and non-escape for BEDC.Derived.CompactUniformModulusCertificateUp.

Constructive story: A $\mathsf{CompactUniformModulusCertificate}^{\uparrow}$ packet is generated from compact finite-net rows, continuous-map pointwise-modulus rows, RatUp lower-bound folds, Metric source and target triangle rows, UniformModulus handoff, Real scalar seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7457

A Concrete Naming Certificate for $\mathsf{BourbakiCompletionFilter}^{\uparrow}$

Following Bourbaki and Bishop constructive completion practice, the $\mathsf{BourbakiCompletionFilter}^{\uparrow}$ packet records the filter-based completion surface as finite BEDC data before any quotient-filter carrier or ambient completeness theorem is available. It reads the Cauchy-filter basis of chapter 1501, the reflector discipline of chapter 2149, the constructive completion route of chapter 1556, the finite stream window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a seed object for the L10 Real and Completion route; it does not construct quotient filters, select limits, use countable choice, assert selected representatives, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7457.1 (Bourbaki completion filter carrier)**.** *A* $\mathsf{BourbakiCompletionFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, M, C, S, R, E, H, T, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row naming the finite Cauchy-filter basis packets and their common-refinement reads.*
2. *M is the* $\mathsf{MetricCompletionReflector}^{\uparrow}$ *row that records the reflector-facing completion request for the same metric source.*
3. *C is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route through regular Cauchy names, dyadic tolerances, Cauchy-filter basis data, separated comparison, dense embedding, and uniformly continuous handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the displayed basis through finite observation schedules.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows and tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after B, M, C, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the basis, reflector, constructive completion route, stream windows, regular readback, and Real seal.*
8. *T records the displayed* Cont *replay from the Cauchy-filter basis to the reflector request, from the reflector request to the constructive completion route, from stream windows to the regular readback, and from the regular readback to the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{MetricCompletionReflector}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BourbakiCompletionFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient filters, selected limits, selected representatives, countable choice, host real equality, or an ambient completion object.

Theorem 7457.2 (BourbakiCompletionFilter NameCert obligations)**.** *For every accepted* BourbakiCompletionFilter *carrier $F = (B, M, C, S, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BourbakiCompletionFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy-filter basis, metric-completion reflector, constructive completion route, StreamName window, RegSeqRat readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert row. No obligation is stored in a quotient-filter object, selected limit, countable-choice schedule, selected representative, host real equality row, or ambient completion theorem.*

Proof. Project the carrier of Definition 7457.1. The displayed rows $B, M, C, S, R, E, H, T, P, N$ are the whole finite packet, and the local NameCert row names only those coordinates.

The route rows bind the basis to the completion surface. The Cauchy-filter basis B feeds the reflector request M and the constructive completion route C; the finite StreamName row S and RegSeqRat row R then provide the readback through which the terminal Real seal E is visible.

Transport, replay, and provenance add no hidden coordinate. The rows H, T, P, N only move, replay, source, and name the same displayed packet, so quotient filters, selected limits, countable choice, selected representatives, host real equality, and ambient completion have no row from which a NameCert obligation could be exported. □

Theorem 7457.3 (BourbakiCompletionFilter Cauchy nonescape)**.** *Every completion-facing read of an accepted* $\mathsf{BourbakiCompletionFilter}^{\uparrow}$ *packet factors through*

$$\mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{MetricCompletionReflector}^{\uparrow} \longmapsto \mathsf{ConstructiveMetricCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The read stays inside the displayed rows $B, M, C, S, R, E, H, T, P, N$ and exports no quotient filters, countable choice, selected limits, selected representatives, detached $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completion.*

Proof. Begin with the basis row B. It is the only Cauchy-filter source carried by the packet, and its common-refinement reads are the inputs to the reflector-facing row M.

The constructive completion route C is downstream of that basis and reflector surface. The finite window row S and regular rational readback R make the completion read visible as a BEDC route before the terminal seal E is exposed.

Finally, the structural rows H, T, P, N preserve the same sequence of displayed coordinates. Since the carrier excludes quotient-filter objects, choice schedules, selected limits, selected representatives, detached Real completeness, and ambient completion objects, no completion-facing consumer can escape the finite route named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{BourbakiCompletionFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BourbakiCompletionFilter}^{\uparrow}$ packet is generated from a Cauchy-filter basis, metric-completion reflector row, constructive completion route, finite StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BourbakiCompletionFilter}^{\uparrow}}$, seed)

Scope closed: Definition 7457.1, Definition 7457.2, and Definition 7457.3 seed the filter-based completion surface over Cauchy-filter basis, reflector, constructive completion, StreamName, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BourbakiCompletionFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim quotient filters, countable choice, selected limits, selected representatives, detached $\mathsf{Real}^{\uparrow}$ completeness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter basis handoff, reflector replay, constructive completion replay, StreamName and RegSeqRat readback, Real-seal nonescape, and transport provenance for the same packet.

Chapter 7458

A Concrete Naming Certificate for $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$

Following Hardy and Knopp on products of positive convergent series, the $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$ packet records the finite NameCert consumer that reads two positive real-series rows, forms their Cauchy product through bounded convolution windows, and hands the resulting product row to the constructive real-series surface. It sits over the real-series carrier of chapter 1581, the positivity route of chapter 2080, the convolution carrier of chapter 6968, the majorant discipline of chapter 1700, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38. The packet is a consumer of displayed finite Real and completion rows; it is not an ambient summation operator, a quotient equality of infinite series, or a host completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7458.1 (Positive series Cauchy product carrier)**.** *A* $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (A, B, W, M, T, S, D, E, H, C, Q, N)$$

with the following visible rows:

1. *A and B are positive* $\mathsf{RealSeries}^{\uparrow}$ *source rows, each exposing finite coefficient windows, partial-sum windows, and the positivity ledgers read by* $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ *consumers.*
2. *W is the bounded convolution-window row. For a requested product coefficient or partial product sum it lists only the finitely many pairs $i + j = n$ and the finite rectangular windows needed for the requested tail budget.*
3. *M is the absolute or majorant tail-control row. It records the positive majorant comparison that bounds the remaining convolution tail before a product read is exposed.*
4. *T is the* $\mathsf{StreamName}^{\uparrow}$ *replay row for the two source streams and the product stream windows.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the partial sums, majorant bounds, and product-tail estimates.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare source budgets, convolution-window budgets, and the final product tolerance.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product value, available only after S and D have been displayed.*

8. *H records componentwise* hsame *transport for the two source series, convolution window, majorant tail row, stream replay, rational readback, dyadic tolerance, Real seal, provenance, and local name.*
9. *C records displayed* Cont *replay for reading source coefficients, convolution windows, tail bounds, rational tolerances, dyadic comparisons, and the terminal Real seal.*
10. *Q is the* Pkg *provenance over* $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$, $\mathsf{CauchyProductSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed finite rows. It has no coordinate for an external infinite sum, quotient equality of series, selected completion representative, countable choice, or ambient real completeness.

Theorem 7458.2 (Positive series Cauchy product NameCert obligations)**.** *For every accepted* $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$ *carrier* $P = (A, B, W, M, T, S, D, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PositiveSeriesCauchyPr}}$ *surface consists exactly of carrier admission for the two positive real-series rows, convolution-window row, majorant tail-control row, StreamName replay row, RegSeqRat readback row, DyadicRatCore tolerance row, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. Every product-facing consumer must read* $A, B, W, M, T, S, D, E, H, C, Q, N$ *in that finite route before it may consume the product row.*

Proof. Project the carrier of Definition 7458.1. The only source series available to the packet are the positive real-series rows A and B, and the only product coefficients available to a consumer are those exposed by the bounded convolution window W.

The tail step is finite. The row M records the absolute or positive majorant comparison, while T, S, D replay the stream windows, rational readback, and dyadic tolerance comparisons needed to bound the requested product tail.

The terminal product value is then sealed by E. Transport, replay, provenance, and local naming are exactly H, C, Q, N, so the packet cannot export an external infinite sum, quotient equality of series, selected completion representative, countable choice, or ambient real completeness. □

Theorem 7458.3 (Positive series Cauchy product Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$ *packet reaches the product Real seal only through the displayed route*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{PringsheimPositiveRealSeries}^{\uparrow},\ \mathsf{CauchyProductSeries}^{\uparrow},\ \mathsf{AbsoluteConvergence}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRa}$$

The handoff supplies a finite product-series readback over convolution windows and majorant tails. It does not supply ambient Real completeness, a host infinite-sum operator, quotient real equality, selected limit representatives, or a completion object outside the packet.

Proof. Begin with the NameCert obligations of Definition 7458.2. They force any product-facing read to expose the two positive source rows and the convolution-window row before the product is read.

Next read the majorant and tolerance rows. The absolute or positive majorant row M bounds the remaining convolution tail, the StreamName row T fixes the finite windows, and the RegSeqRat and DyadicRatCore rows S, D provide the tolerance comparison consumed by the product request.

Finally consume the Real seal E. Since every completion-facing read has already passed through A, B, W, M, T, S, D, the handoff is a displayed finite route to $\mathsf{Real}^{\uparrow}$, not an ambient summation or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}$ packet is generated from two positive RealSeries rows, bounded convolution windows, absolute or majorant tail-control rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparison, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7458.1, Definition 7458.2, and Definition 7458.3 seed the finite product-series route from positive source rows to the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveSeriesCauchyProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient summation, quotient equality of series, selected completion representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convolution-window exactness, majorant tail control, tolerance readback, Real-seal nonescape, transport, provenance, refusal, and local naming for BEDC.Derived.PositiveSeriesCauchyProd

Chapter 7459

A Concrete Naming Certificate for $\mathsf{UpperDiniDerivative}^{\uparrow}$

Following Saks and Bishop–Bridges constructive analysis, the $\mathsf{UpperDiniDerivative}^{\uparrow}$ packet separates the one-sided upper Dini derivative carrier from the combined Dini-window packet of chapter 6110. It sits beside the Denjoy–Young–Saks boundary of chapter 5829, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The chapter names the upper one-sided limsup surface as finite BEDC data; it does not assert differentiability, a selected derivative value, almost-everywhere alternatives, or ambient real completeness.

Definition 7459.1 (UpperDiniDerivative carrier)**.** *An* $\mathsf{UpperDiniDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, F, X, Q, S, R, D, W, E, H, C, P, N)$$

with the following rows:

1. *I is the interval-cell row on which the one-sided derivative request is displayed.*
2. *F is the real-valued function-observation row over the inspected interval cells.*
3. *X is the base-point row, read only through the same real-name and interval surface as I.*
4. *Q is the positive approach-direction row, recording the finite side from which quotients are sampled.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row for the inspected function and base-point observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream windows in S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the radii, slack comparisons, and quotient-window refinements.*
8. *W is the sibling* $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *row whose upper Dini coordinate is being exposed.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed upper-bound read.*
10. *H records componentwise* hsame *transport for the interval, function, base-point, direction, stream, readback, dyadic, window, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay for refinements of the same one-sided quotient window.*

12. *P is the* Pkg *provenance over* $\mathsf{UpperDiniDerivative}^{\uparrow}$, $\mathsf{DiniDerivativeWindow}^{\uparrow}$, $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperDiniDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, F, X, Q, S, R, D, W, E, H, C, P, N$. It has no coordinate for a lower Dini row, a two-sided derivative, a selected limiting slope, a measure-theoretic alternative, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7459.2 (UpperDiniDerivative window stability)**.** *For every accepted* $\mathsf{UpperDiniDerivative}^{\uparrow}$ *carrier*

$$U = (I, F, X, Q, S, R, D, W, E, H, C, P, N),$$

any refinement of the displayed one-sided quotient window is accepted only when it factors through the same interval row I, function-observation row F, base-point row X, approach-direction row Q, stream schedule S, regular readback R, dyadic tolerance ledger D, and sibling $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *row W. Consequently the public upper-Dini read is stable under finite window refinement and exports no lower-Dini coordinate, two-sided derivative, selected limiting slope, quotient equality of real names, or ambient completeness principle.*

Proof. Project the carrier of Definition 7459.1. The one-sided quotient surface is not an isolated real row; it is tied to the interval, function observation, base point, approach direction, stream schedule, regular readback, dyadic ledger, and sibling Dini window.

For a refinement, replay through C and transport through H. The replay can change only the inspected finite window while preserving the rows $I, F, X, Q, S, R, D, W, E, P, N$ as the same displayed packet.

The excluded data are absent from the carrier. Since no coordinate names a lower-Dini export, two-sided derivative, selected limiting slope, quotient equality, choice schedule, or ambient completeness theorem, a refinement-stable consumer cannot read any of those objects. □

Theorem 7459.3 (UpperDiniDerivative NameCert obligations)**.** *For every accepted* $\mathsf{UpperDiniDerivative}^{\uparrow}$ *carrier $U = (I, F, X, Q, S, R, D, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UpperDiniDerivative}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the interval cell, function-observation row, base-point row, approach-direction row, stream schedule, regular readback, dyadic tolerance ledger, sibling Dini-window row, upper real seal, and structural rows H, C, P, N;*

2. *one-sided quotient coverage, requiring each upper-Dini request to display the positive approach side Q together with matching stream windows in S, readbacks in R, and dyadic tolerance entries in D;*

3. *sibling-window compatibility, requiring the upper coordinate read from W to be the same finite upper-Dini row routed by the carrier;*

4. *window-stability replay, requiring refinements to replay through C and transport through H without changing the carrier-local upper row;*

5. *non-escape from differentiability, lower-Dini export, selected limiting slopes, quotient real equality, countable choice, measure-theoretic alternatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission reads the packet named in Definition 7459.1. The analytic rows are $I, F, X, Q, S, R, D, W, E$, while H, C, P, N provide transport, replay, provenance, and local naming.

One-sided quotient coverage is finite because every request is displayed through the approach-direction row Q, the stream schedule S, the regular readback R, and the dyadic ledger D. A request lacking one of those rows is not an accepted $\mathsf{UpperDiniDerivative}^{\uparrow}$ read.

Sibling-window compatibility uses the $\mathsf{DiniDerivativeWindow}^{\uparrow}$ row W. The upper coordinate is consumed from that sibling window and then sealed by E only after the same route has been displayed.

Finally apply the stability theorem Definition 7459.2. Refinement replay remains inside the same carrier, and the listed exclusions are not rows of the packet, so the NameCert surface is exhausted by the stated obligations. □

Falsifiable prediction: Every accepted $\mathsf{UpperDiniDerivative}^{\uparrow}$ consumer reaching an upper-Dini read without the displayed one-sided direction, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, and DiniDerivativeWindow row refutes the carrier.

Independence witness: dini-derivative-window, denjoy-young-saks, real, regseqrat, streamname, dyadicratcore: no listed sibling supplies the whole UpperDiniDerivative packet, because each sibling contributes only the combined Dini window, broader differentiability boundary, terminal real seal, rational readback, observation schedule, or tolerance ledger while this chapter binds the one-sided upper Dini carrier in one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{UpperDiniDerivative}^{\uparrow}}$.

Constructive story: An $\mathsf{UpperDiniDerivative}^{\uparrow}$ packet is generated from interval cells, function-observation rows, a base-point row, a positive approach-direction row, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance rows, a DiniDerivativeWindow upper row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperDiniDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7459.1, Definition 7459.2, and Definition 7459.3 seed the one-sided upper Dini derivative carrier and its finite window-stability obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpperDiniDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim differentiability, lower-Dini export, two-sided derivative existence, selected limiting slopes, almost-everywhere alternatives, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, one-sided quotient coverage, sibling-window compatibility, finite window stability, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UpperDiniDeriv

Chapter 7460

A Concrete Naming Certificate for KreinMilmanCompactConvex$^{\uparrow}$

Following Krein and Milman, the KreinMilmanCompactConvex$^{\uparrow}$ packet records the finite NameCert surface for a compact convex body inspected inside a locally convex real vector environment. It sits over the compact carrier of chapter 60, the convex-set interface of chapter 208, the locally convex space source of chapter 3386, the scalar Real$^{\uparrow}$ seal of chapter 38, and the finite extreme-point sibling of chapter 3610. The chapter names the compact-convex dependency surface; it does not assert a host theorem that the body is the closed convex hull of all extreme points.

Definition 7460.1 (Krein–Milman compact-convex carrier)**.** *A* KreinMilmanCompactConvex$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, V, X, W, E, F, H, C, P, N)$$

with the following displayed rows:

1. *A is the* Compact$^{\uparrow}$ *row for the inspected compact subset.*
2. *V is the* LocallyConvexSpace$^{\uparrow}$ *row carrying the ambient vector-space, seminorm-window, and weak-neighborhood data available to the packet.*
3. *X is the* ConvexSet$^{\uparrow}$ *row for the same subset, including displayed affine-combination and convex-membership reads.*
4. *W is the weak-window row through which continuous linear observations and finite neighborhood tests are opened.*
5. *E is the* KreinMilmanFiniteExtreme$^{\uparrow}$ *sibling row, exposing only the finite extreme-point witness surface licensed by that chapter.*
6. *F records finite face, hull, and scalar-comparison rows, with all scalar values sealed through* Real$^{\uparrow}$.
7. *H records componentwise* hsame *transport for the compact, locally convex, convex-set, weak-window, extreme-sibling, face, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-convex reads through weak windows and finite face ledgers to the extreme-sibling handoff.*
9. *P is the* Pkg *provenance over* KreinMilmanCompactConvex$^{\uparrow}$, Compact$^{\uparrow}$, ConvexSet$^{\uparrow}$, LocallyConvexSpace$^{\uparrow}$, Real$^{\uparrow}$, KreinMilmanFiniteExtreme$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $A, V, X, W, E, F, H, C, P, N$. It has no coordinate for a completed dual space, unrestricted separation, a chosen list of all extreme points, quotient convex-body equality, countable choice, or ambient real completeness.

Theorem 7460.2 (Krein–Milman compact-convex NameCert obligations)**.** *For every accepted* $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$ *carrier*

$$K = (A, V, X, W, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}$ *surface consists exactly of compact admission for A, locally convex ambient admission for V, convex-set admission for X, weak-window visibility for W, finite extreme-sibling readback for E, finite face and scalar-comparison visibility for F, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every compact-convex consumer must read the dependency route

$$A, V, X \longmapsto W, F \longmapsto E \longmapsto H, C, P, N$$

before using the packet as a Krein–Milman compact-convex name. The route exports finite certificate rows only; it exports no host closed-convex-hull equality, no unrestricted compactness principle, no global supporting-functional selector, no quotient convex body, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 7460.1. The compact source is A, the ambient locally convex source is V, and the convex subset source is X.

Open the weak and scalar windows next. The row W records only finite weak-neighborhood and functional-observation tests, while F records finite face, hull, and scalar-comparison data whose scalar reads pass through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Read the finite extreme-point sibling through E. By chapter 3610, this handoff supplies a finite exposed-face and candidate-extreme witness route, not a chosen set of all extreme points or a completed dual-space theorem.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no row for unrestricted separation, quotient convex-body equality, countable choice, host compact-convex hull equality, or ambient real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 7460.3 (Krein–Milman compact-convex finite extreme-window handoff)**.** *Every accepted* $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$ *carrier*

$$K = (A, V, X, W, E, F, H, C, P, N)$$

must expose the finite extreme-point row E through chapter 3610 before any compact-convex hull read, weak-window read, scalar comparison, or consumer export is public. The exported route is the finite window

$$A, V, X, W, F \longmapsto E \longmapsto H, C, P, N,$$

not a selected set of all extreme points, a completed dual-space theorem, or a host closed-convex-hull equality.

Proof. Project the carrier in Definition 7460.1. The compact, locally convex, and convex-set rows are A, V, X, and the weak-window and finite scalar rows are W, F.

Apply Definition 7460.2. That obligation surface requires finite extreme-sibling readback for E as part of the same local $\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}$ packet.

Read the existing sibling route of chapter 3610. Because E is the only extreme-point row carried by the packet, every compact-convex hull or scalar export must pass through it before H, C, P, N repack the public route. The carrier has no coordinate for a selected set of all extreme points, a completed dual space, unrestricted separation, or a classical Krein–Milman theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}$.

Constructive story: A $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$ packet is generated from compact, locally convex, convex-set, weak-window, finite extreme-sibling, face, scalar, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7460.1 and Definition 7460.2 seed the compact-convex NameCert surface over compact, convex-set, locally convex, real scalar, weak-window, and finite extreme-point sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KreinMilmanCompactConvex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Krein–Milman theorem, closed-convex-hull equality, unrestricted separation, selected extreme-point sets, quotient convex bodies, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact routing, convex-set admission, locally convex weak-window routing, finite extreme-sibling readback, scalar face visibility, transport, replay, provenance, and local naming for the same packet.

Chapter 7461

A Concrete Naming Certificate for $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$ packet records the separated completion route that remains visible through finite windows before any terminal real seal is consumed. It sits between the separated completion boundary of chapter 402, the finite-window completion handoff of chapter 1559, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name surface of chapter 38. The packet names a constructive L10 completion surface without quotient representatives, countable choice, host equality of completed points, or an ambient completeness theorem.

Definition 7461.1 (Bishop separated finite-window completion carrier)**.** *A* BishopSeparatedFiniteWindowCompletio *carrier is a finite* BHist *packet*

$$B = (H, F, S, D, R, E, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion boundary, read as finite seal data rather than as a quotient class.*
2. *F is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *finite-window completion row that fixes the completion-facing request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened by the finite request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the radii and slack comparisons attached to the visible windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed after the windows and dyadic tolerances have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after H, F, S, D, R have been replayed.*
7. *T records componentwise* hsame *transport for the separated boundary, finite-window completion row, stream schedule, dyadic ledger, regular readback, real seal, continuation, provenance, and local name.*
8. *C records displayed* Cont *replay for refinements that remain inside the same separated finite-window packet.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletionFinite}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $H, F, S, D, R, E, T, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, host equality of completed points, or ambient completeness.

Theorem 7461.2 (Bishop separated finite-window completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$ *carrier*

$$B = (H, F, S, D, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the separated completion boundary H, finite-window completion row F, stream schedule S, dyadic ledger D, regular readback R, terminal real seal E, and structural rows T, C, P, N;*
2. *separated-window compatibility, requiring F to consume the separated boundary H before any stream window or rational readback is public;*
3. *finite coverage, requiring every visible completion request in F to have a matching window in S, dyadic tolerance in D, and regular readback in R;*
4. *stability under transport and replay, requiring refinements to move through T and C without replacing the displayed separated boundary or finite-window row;*
5. *real-seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to export only the named seal reached through H, F, S, D, R.*

No local obligation exports quotient representatives, host equality of completed points, countable choice, or an ambient completeness theorem.

Proof. Project the carrier of Definition 7461.1. Its visible coordinates are exactly $H, F, S, D, R, E, T, C, P, N$, so carrier admission can expose only those rows.

Separated-window compatibility reads H before F. The finite-window row is therefore attached to a displayed Hausdorff completion boundary, not to a quotient representative or host completion object.

Finite coverage then reads S, D, R from the same packet. Each completion request visible in F has its stream window, dyadic tolerance, and regular rational readback present before the terminal seal is reachable.

Transport and replay use T and C componentwise. Since those rows move the displayed packet rather than replacing it, the route remains inside $H, F, S, D, R, E, T, C, P, N$.

The real-seal clause is the final projection. The carrier has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, host equality, or ambient completeness, so none of those objects can be exported by the local NameCert surface. □

Theorem 7461.3 (Bishop separated finite-window completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$ *packet reaches a regular rational readback or terminal real seal only through the finite separated route*

$$H,\ F,\ S,\ D,\ R,\ E,\ T,\ C,\ P,\ N.$$

Consequently the handoff to $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *exports separated finite-window completion data, not quotient completion equality, selected representatives, countable-choice schedules, host equality of completed points, or ambient completeness.*

Proof. Begin with the obligation surface of Definition 7461.2. It forces any completion-facing consumer to read the separated boundary H and finite-window row F before inspecting the downstream readback.

The finite coverage obligation supplies the window schedule, dyadic tolerance ledger, and regular rational readback as S, D, R. These rows are displayed inside the same BHist packet and are preserved by the transport and replay rows T, C.

The terminal seal E is downstream of that route. Package provenance P and local naming N record the same finite packet, so the consumer has no access path outside $H, F, S, D, R, E, T, C, P, N$.

The excluded completion principles are absent from the carrier of Definition 7461.1. Therefore the handoff can export only the separated finite-window completion surface named by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7461.1, Definition 7461.2, and Definition 7461.3 seed the separated finite-window completion route from Hausdorff completion and finite-window completion rows to RegSeqRat readback and RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-window compatibility, finite coverage, transport replay stability, RegSeqRat handoff, RealUp seal nonescape, provenance, and local naming for BEDC.Derived.BishopSeparatedFiniteW

Constructive story: A $\mathsf{BishopSeparatedFiniteWindowCompletion}^{\uparrow}$ packet is generated from a HausdorffCompletion separated boundary, a MetricCompletionFinite finite-window row, a StreamName schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7462

A Concrete Naming Certificate for $\mathsf{CantorIntervalTrap}^{\uparrow}$

Following Bishop and Bridges on nested intervals, the $\mathsf{CantorIntervalTrap}^{\uparrow}$ packet records the finite trap by which a decreasing displayed interval chain catches a single Real-and-Completion boundary without invoking open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits over the dyadic nested selector of chapter 6784, the Cantor intersection theorem packet of chapter 2318, the dyadic radius core of chapter 291, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The originating L10 route is the Real and Completion construction order from section 9467.12, where $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are admitted by finite observation and readback rows.

Definition 7462.1 (Cantor interval trap carrier)**.** *A* $\mathsf{CantorIntervalTrap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *I is the nested interval row, listing a finite prefix of closed dyadic intervals together with the inclusion checks read from the* $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$ *and* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *siblings.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording endpoint comparison, midpoint, and radius-shrink data for each displayed interval row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which finite interval prefixes are inspected for each requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed interval centers and dyadic radius bounds.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the nested interval prefix, dyadic radius ledger, stream window, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for nested interval rows, dyadic radius rows, window rows, readback rows, Real seal rows, provenance rows, and the local naming row.*
7. *C records displayed* Cont *replay from interval-prefix requests to dyadic radius checks, from those checks to stream windows, and from stream windows through the regular readback to the Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{CantorIntervalTrap}^{\uparrow}$, $\mathsf{DyadicNestedIntervalSelector}^{\uparrow}$, $\mathsf{CantorIntersectionTheorem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntervalTrap}^{\uparrow}}$ *row.*

The classifier compares accepted traps only through these displayed rows. It has no coordinate for a selected host limit, quotient equality of streams, trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7462.2 (Cantor interval trap NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntervalTrap}^{\uparrow}$ *carrier $T = (I, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorIntervalTrap}^{\uparrow}}$ *surface consists exactly of carrier admission for the nested interval prefix, dyadic radius ledger, finite StreamName windows, RegSeqRat readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing consumer must read $I, D, W, R, E, H, C, P, N$ in that finite route before it can consume the trapped Real boundary.*

Proof. Project the carrier of Definition 7462.1. The nested interval data are exactly I, and every radius or endpoint comparison available to the packet is stored in the dyadic ledger D.

Next read the finite observation route. The window row W determines the inspected interval prefix for a requested tolerance, while R records the regular rational readback obtained from the displayed centers and radius bounds.

Finally expose the Real seal E through the structural rows. Transport, replay, provenance, and local naming are exactly H, C, P, N, so the certificate cannot export a selected host limit, quotient stream equality, trichotomy, open-cover compactness, countable choice, or ambient Real completeness. □

Theorem 7462.3 (Cantor interval trap Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CantorIntervalTrap}^{\uparrow}$ *packet reaches the trapped point only through the displayed route*

$$\mathsf{DyadicNestedIntervalSelector}^{\uparrow} \longmapsto \mathsf{CantorIntersectionTheorem}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRa}$$

The handoff supplies nested interval rows, dyadic radii, finite windows, regular rational readback, transport, replay, provenance, and local naming. It does not supply ambient interval compactness, a selected host point, quotient equality of Cauchy streams, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the nested interval and Cantor-intersection sibling rows. They provide only the displayed finite interval prefix and its inclusion route, not a completed host intersection object.

Read the dyadic and stream rows next. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies the endpoint and radius comparisons, and the $\mathsf{StreamName}^{\uparrow}$ window row fixes which finite prefix is inspected for the requested tolerance.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal then expose the Real-facing boundary. Because the carrier has no coordinate for open-cover compactness, choice, trichotomy, quotient stream equality, selected host limits, or ambient completeness, every Real consumer remains inside the displayed trap route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntervalTrap}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntervalTrap}^{\uparrow}$ packet is generated from nested interval rows, DyadicRatCore radius ledgers, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntervalTrap}^{\uparrow}}$, seed)

Scope closed: Definition 7462.1, Definition 7462.2, and Definition 7462.3 seed the finite nested-interval trap route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CantorIntervalTrap}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ambient interval compactness, selected host limits, quotient stream equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked nested-prefix exposure, dyadic radius exactness, StreamName window inspection, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntervalTrapUp.

Chapter 7463

A Concrete Naming Certificate for $\mathsf{InscriptionEventNovelty}^{\uparrow}$

Following the hard-problems inscription-event reading of Definition 9415.5, the $\mathsf{InscriptionEventNovelty}^{\uparrow}$ packet records a finite prior-record boundary together with the new record whose origin is not reducible to the displayed closed-generation route. It sits beside the free-will inscription commitment of chapter 592, the classifier novelty witness of chapter 6138, the finite witnessed refutation packet of chapter 576, and the GAP closure boundary of chapter 1006. These siblings expose commitment, classifier shift, contrary-record, and boundary rows; this packet binds the inscription event itself as finite NameCert data.

Definition 7463.1 (Inscription-event novelty carrier)**.** *An* $\mathsf{InscriptionEventNovelty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, A, J, G, L, H, C, P, N)$$

with the following visible rows:

1. *B is the prior-record boundary row, listing the records visible before the inscription event.*
2. *A is the after-record row, displaying the record that appears at the inscription point.*
3. *J is the novelty witness row, recording that A is not presented as a forced consequence of B under the displayed closed-generation route.*
4. *G is the* GAP *row that keeps the non-reduction boundary visible instead of turning the event into a hidden theorem about transcendental supply.*
5. *L is the local ledger row tying the carrier to Definition 9415.5 and to the sibling packets for free-will inscription commitment, classifier novelty, witnessed refutation, and GAP closure.*
6. *H records componentwise* hsame *transport for prior-record, after-record, novelty, GAP, ledger, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for prior-record inspection, after-record exposure, novelty checking, GAP non-escape, and local ledger readback.*
8. *P is the* Pkg *provenance over* $\mathsf{InscriptionEventNovelty}^{\uparrow}$, BHist, GAP, hsame, Cont, Pkg, *and* NameCert *together with the named sibling rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionEventNovelty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for private access to T*, a proof of metaphysical freedom, a hidden host-causality theorem, a global classifier-shift verdict, or a positive-discovery claim.*

Theorem 7463.2 (Inscription-event novelty NameCert obligations)**.** *For every accepted* $\mathsf{InscriptionEventNovelty}^{\uparrow}$ *carrier* $E = (B, A, J, G, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{InscriptionEventNovelty}^{\uparrow}}$ *surface consists exactly of carrier admission for the prior-record boundary, after-record row, novelty witness, GAP non-escape row, local ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every downstream read of the inscription event must expose* $B, A, J, G, L, H, C, P, N$ *before it may treat the new record as an inscription-event novelty row.*

Proof. Project the carrier of Definition 7463.1. The prior records are B, the new displayed record is A, and the novelty witness is J.

The non-reduction boundary is explicit. The row G records the GAP side of the event, while L names the hard-problems definition and the sibling packets that can consume the event without replacing it by commitment, classifier shift, refutation, or closure-boundary data.

Finally replay through H, C, P, N. These rows transport, replay, source, and name the same finite packet, so no accepted read obtains private access to T, a metaphysical-freedom theorem, a hidden host-causality theorem, a global classifier-shift verdict, or a positive-discovery claim. □

Theorem 7463.3 (Inscription-event novelty nonescape)**.** *Every consumer of an accepted* $\mathsf{InscriptionEventNovelty}^{\uparrow}$ *packet factors through*

$$B \longmapsto A \longmapsto J \longmapsto G \longmapsto L \longmapsto H, C, P, N.$$

The read may cite the hard-problems inscription-event definition and the sibling rows for free-will inscription commitment, classifier novelty, finite witnessed refutation, and GAP closure, but it may not collapse the packet into any one of those siblings or into a hidden host theorem.

Proof. Begin with the prior-record and after-record rows. They fix the event boundary as a finite record transition rather than as a global statement about all possible conscious accumulation.

Next read the novelty and GAP rows. The novelty witness J keeps the new record distinct from the displayed closed-generation route, and G keeps that non-reduction boundary visible to downstream consumers.

The ledger and structural rows then preserve the event as its own packet. Since L, H, C, P, N only cite and replay the displayed rows, the packet cannot become a free-will commitment, classifier novelty verdict, witnessed refutation, GAP closure theorem, private access route, or hidden host-causality theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionEventNovelty}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionEventNovelty}^{\uparrow}$ packet is generated from prior-record boundary rows, after-record rows, novelty witnesses, GAP non-escape rows, local ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InscriptionEventNovelty}^{\uparrow}}$, seed)

Scope closed: Definition 7463.1, Definition 7463.2, and Definition 7463.3 seed the finite inscription-event novelty packet over prior-record, after-record, GAP, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InscriptionEventNovelty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim private access to T, metaphysical freedom, hidden host causality, global classifier shift, positive discovery, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked prior-record exposure, after-record exposure, novelty witnessing, GAP non-escape, sibling-ledger routing, transport, replay, provenance, and local naming for BEDC.Derived.InscriptionEventNoveltyUp.

Chapter 7464

A Concrete Naming Certificate for $\mathsf{PositiveRealNormalization}^{\uparrow}$

Bishop and Bridges constructive analysis treats a positive radius as usable only after its lower-bound witness, rational readback, and consumer boundary have been made explicit. The $\mathsf{PositiveRealNormalization}^{\uparrow}$ packet records this normalization handoff over the positive-real source of chapter 1587, the dyadic comparison substrate of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the metric-radius consumers of chapter 57. It names a finite choice-free route from an apartness-backed positive real to a normalized dyadic minorant without importing host positivity, quotient equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7464.1 (Positive-real normalization carrier)**.** *A* $\mathsf{PositiveRealNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, D, W, Q, M, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{PositiveReal}^{\uparrow}$ *packet, including its sealed* $\mathsf{Real}^{\uparrow}$ *row and apartness-from-zero evidence.*
2. *A is the displayed apartness row read from R; it is the only positivity source admitted by the carrier.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger carried by A.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window used to inspect the real-name source.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream window W.*
6. *M is the normalized dyadic minorant row, recording one positive dyadic radius below the apartness radius exposed by D.*
7. *S is the downstream scale row that lets metric or radius-ledger consumers read M as a finite radius token.*
8. *H records componentwise* hsame *transport for $R, A, D, W, Q, M, S, C, P, N$.*
9. *C records displayed* Cont *replay from a positive-real request through the dyadic ledger, finite window, readback, normalized minorant, and downstream scale row.*

10. *P is the* Pkg *provenance over* $\mathsf{PositiveRealNormalization}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveRealNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, A, D, W, Q, M, S, H, C, P, N$. It has no coordinate for host positivity, trichotomy, quotient real equality, selected positive representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7464.2 (Positive-real normalization NameCert obligations)**.** *For every accepted* PositiveRealNormalizatio *carrier*

$$K = (R, A, D, W, Q, M, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PositiveRealNormalization}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the positive-real source, apartness row, dyadic radius ledger, stream window, regular readback, normalized dyadic minorant, downstream scale row, transport, replay, provenance, and local name;*
2. *positivity-source confinement, requiring every normalization read to enter through the displayed* $\mathsf{PositiveReal}^{\uparrow}$ *packet R and its apartness row A;*
3. *dyadic minorant exactness, requiring M to be read from the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D transported with the carrier;*
4. *regular-readback stability, requiring W and Q to remain attached to the real-name source whenever M is transported or replayed;*
5. *consumer boundary, requiring metric or radius-ledger consumers to receive only the normalized dyadic token M together with S, H, C, P, N.*

Proof. Project the carrier of Definition 7464.1. The analytic rows are R, A, D, W, Q, M, S, and the structural rows are H, C, P, N.

The positivity-source obligation follows from the first two coordinates: the source is the displayed $\mathsf{PositiveReal}^{\uparrow}$ packet and the apartness row A is read from it. There is no separate host-order or trichotomy coordinate to inspect.

The dyadic minorant obligation reads the positive radius ledger D before the normalized row M is exposed. Thus the minorant is a finite $\mathsf{DyadicRatCore}^{\uparrow}$ comparison token, not a selected real representative.

The stability and boundary obligations use H and C. Transport and replay move R, A, D, W, Q, M, S together, so downstream metric or radius-ledger consumers see only the normalized dyadic token and its displayed support rows. The excluded host principles are absent from the carrier inventory. □

Theorem 7464.3 (Positive-real normalization dyadic minorant)**.** *Every accepted* $\mathsf{PositiveRealNormalization}^{\uparrow}$ *packet exposes a positive dyadic minorant only through the route*

$$\mathsf{PositiveReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ M.$$

The minorant row M is below the apartness radius displayed by D and is transported only with the same positive-real source, finite stream window, regular readback, replay, provenance, and local name. It exports no host positivity predicate, quotient real equality, selected representative, trichotomy split, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use the obligation surface of Definition 7464.2. Positivity-source confinement places every read at R and A, while dyadic minorant exactness places M downstream of D.

The finite observation rows W and Q keep the source real-name read attached to the same packet. Componentwise transport through H and replay through C preserve the displayed route, so the minorant cannot be detached from its source rows.

Since no excluded host-level datum is a carrier coordinate, the minorant row exports only the finite dyadic token named by M and its support rows. □

Theorem 7464.4 (Positive-real normalization RegSeqRat stability)**.** *For every accepted* $\mathsf{PositiveRealNormalization}^{\uparrow}$ *carrier* $K = (R, A, D, W, Q, M, S, H, C, P, N)$*, replacing the real-name observations by componentwise* hsame*-transported* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows preserves the same normalized minorant whenever the transported packet keeps* $R, A, D, M, S, H, C, P, N$ *displayed. The stabilized read remains a finite route through* $\mathsf{PositiveReal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the regular-readback stability obligation in Definition 7464.2. It requires W and Q to remain attached to the same positive-real source whenever the normalized minorant is moved.

Apply the transport row H to the stream window and regular readback, and use C to replay the same positive-radius request through the transported data. The dyadic ledger D and minorant row M are moved with the packet, so the normalized token is unchanged as a carrier row.

The conclusion is therefore the same finite route, not a quotient assertion about real names or a host equality of positive reals. □

Theorem 7464.5 (Positive-real normalization real-consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, or radius-ledger consumer of an accepted* $\mathsf{PositiveRealNormalization}^{\uparrow}$ *carrier receives only the sealed positive-real source, the apartness row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *minorant ledger, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the normalized radius row, and the support rows* H, C, P, N*. Such a consumer receives no host positivity predicate, quotient real equality, selected positive representative, trichotomy witness, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the consumer boundary obligation from Definition 7464.2. A consumer may inspect the source packet R, the apartness and dyadic rows A, D, the finite observation rows W, Q, the normalized token M, the scale row S, and the structural rows H, C, P, N.

The sibling anchors identify the permitted route: $\mathsf{PositiveReal}^{\uparrow}$ supplies positivity by apartness, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the positive dyadic token, $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ supply finite readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{Met}^{\uparrow}$ consumes only the displayed radius token.

No remaining coordinate stores a host positivity predicate, quotient equality, selected representative, trichotomy, choice selector, or completeness principle. Those data cannot be projected from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveRealNormalization}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveRealNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7464.1, Definition 7464.2, Definition 7464.3, Definition 7464.4, and Definition 7464.5 seed the finite positive-radius normalization route from PositiveRealUp through DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, and MetricSpaceUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveRealNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host positivity, quotient real equality, selected positive representatives, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positivity-source confinement, dyadic minorant exactness, RegSeqRat stability, real-consumer boundary, transport, replay, provenance, and local naming for BEDC.Derived.PositiveRealNormalizationUp.

Constructive story: A PositiveRealNormalization$^{\uparrow}$ packet is generated from a PositiveReal source, an apartness row, a DyadicRatCore radius ledger, a StreamName window, a RegSeqRat readback, a normalized dyadic minorant, a downstream scale row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7465

A Concrete Naming Certificate for DyadicClosedIntervalFiniteNet$^{\uparrow}$

Following Bishop and Bridges, the DyadicClosedIntervalFiniteNet$^{\uparrow}$ packet records the finite dyadic data used to read a closed interval by endpoint rows and then expose a finite net at one unary precision. It sits beside the dyadic arithmetic core of chapter 291, the finite-window reader of chapter 126, the regular rational handoff of chapter 301, the terminal real seal of chapter 38, the metric carrier of chapter 4230, and the compact finite-net route of chapter 3471. The certificate names the interval finite-net surface as displayed BEDC packet data, not as open-cover compactness, quotient real equality, selected endpoints, or ambient completeness.

Definition 7465.1 (Dyadic closed-interval finite-net carrier)**.** *A* DyadicClosedIntervalFiniteNet$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, q, G, F, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *L, U are the lower and upper dyadic closed-interval endpoint rows, sourced through* DyadicRatCore$^{\uparrow}$*.*
2. *q is the unary precision row at which the interval is inspected.*
3. *G is the endpoint-grid row listing the dyadic grid points between L and U at precision q.*
4. *F is the finite-net row naming the selected grid points and their metric-radius witnesses.*
5. *S is the* StreamName$^{\uparrow}$ *readback row for endpoint and grid windows.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row that packages the same endpoint and net observations as regular rational data.*
7. *R is the* Real$^{\uparrow}$ *seal row, available only after L, U, q, G, F, S, Q have been displayed.*
8. *H stores componentwise* hsame *transport for endpoints, precision, endpoint grid, finite net, readback rows, real seal, replay, provenance, and local naming.*
9. *C stores displayed* Cont *replay from the endpoint rows to the grid, from the grid to the finite net, and from the readback rows to the real-facing seal.*
10. *P is the* Pkg *provenance over* DyadicClosedIntervalFiniteNet$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* Met$^{\uparrow}$*,* HeineBorelFiniteNet$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, U, q, G, F, S, Q, R, H, C, P, N$. It has no coordinate for a selected real endpoint outside the dyadic rows, quotient real equality, open-cover compactness, countable choice, or ambient metric completeness.

Theorem 7465.2 (Dyadic closed-interval finite-net NameCert obligations)**.** *For an accepted* DyadicClosedIntervalFi *carrier $I = (L, U, q, G, F, S, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}}$ *obligations are exactly carrier admission, endpoint order visibility, unary precision classification, endpoint-grid coverage, finite-net row coverage,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal factorization, componentwise transport, displayed replay, package provenance, and local naming. Every obligation is discharged by the finite route*

$$L, U \longmapsto q \longmapsto G \longmapsto F \longmapsto S, Q \longmapsto R$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 7465.1. The endpoint rows L, U, precision row q, grid row G, finite-net row F, readback rows S, Q, and seal row R are the only mathematical rows available to the classifier.

Endpoint order visibility and precision classification are read before any net row is consumed. The endpoint rows are dyadic arithmetic data, and the unary precision row fixes the scale at which the endpoint grid is formed.

Endpoint-grid coverage and finite-net row coverage are then finite packet checks. The row G lists the displayed dyadic grid, while F names the finite selected grid points and their metric-radius witnesses.

The readback and seal clauses are factorization clauses. The rows S, Q carry the stream and regular rational readings of the same endpoint-grid packet, and R is reached only through those rows. Transport, replay, provenance, and local naming preserve that packet through H, C, P, N without adding quotient equality, selected endpoints, open-cover compactness, countable choice, or ambient completeness. □

Theorem 7465.3 (Dyadic closed-interval finite-net sibling route)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}$ *packet must first read the dyadic endpoint rows L, U, unary precision q, endpoint-grid row G, finite-net row F,* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q,* $\mathsf{Real}^{\uparrow}$ *seal row R, and the structural rows H, C, P, N. No consumer may replace this route by open-cover compactness, quotient real equality, selected real endpoints, countable choice, or ambient metric completeness.*

Proof. Begin with the accepted carrier of Definition 7465.1. Its sibling-facing rows are exactly the dyadic endpoints, unary precision, endpoint grid, finite net, stream readback, regular rational readback, real seal, and structural rows displayed there.

The metric and finite-net consumers read G and F only after the dyadic endpoints and precision have been exposed. Thus a $\mathsf{Met}^{\uparrow}$ radius witness or $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ handoff is tied to the same endpoint-grid packet.

The regular rational and real consumers read Q and R only after the stream readback row S and finite-net row F are present. Hence the real-facing route factors through the displayed dyadic finite-net data rather than through quotient equality or selected limits.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite packet, so they cannot create an additional endpoint selector, open-cover compactness witness, choice principle, or ambient completeness row. □

Falsifiable prediction: Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ read from a $\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}$ packet factors through the displayed dyadic endpoints, unary precision, endpoint grid, finite net, StreamName readback, RegSeqRat readback, Real seal, componentwise transport, continuation replay, provenance, and local naming before the finite net is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, metricspace, heineborel-finite-net: $\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic endpoints, unary precision, endpoint-grid rows, finite net rows, readback rows, Real seal, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}$ packet is generated from dyadic closed-interval endpoints, a unary precision request, endpoint-grid rows, finite net rows, StreamName and RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7465.1, Definition 7465.2, and Definition 7465.3 seed the finite dyadic closed-interval net packet for one displayed precision.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient real equality, selected endpoints, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint order visibility, precision classification, endpoint-grid coverage, finite-net row coverage, readback factorization, Real-seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.DyadicClosedIntervalFiniteNetUp.

Chapter 7466

A Concrete Naming Certificate for $\mathsf{KreinRutman}^{\uparrow}$

Following Krein and Rutman, the $\mathsf{KreinRutman}^{\uparrow}$ packet records the finite NameCert surface for a compact positive operator acting on a cone inside an ordered Banach-style environment. It sits over the Banach carrier of chapter 90, the compact-operator surface of chapter 1952, the scalar $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the compact-convex spectral sibling of chapter 7460. The packet names the dependency route for a positive eigenvector handoff; it does not assert a host spectral theorem, a chosen spectral radius, or a completed cone interior principle.

Definition 7466.1 (Krein–Rutman carrier)**.** *A* $\mathsf{KreinRutman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, C, T, Q, \rho, V, H, R, P, N)$$

with the following displayed rows:

1. *B is the ordered Banach row, carrying the normed vector carrier, order relation, cone membership, and real scalar reads used by the packet.*
2. *C is the positive cone row, exposing finite cone-membership, nonzero, order, and normalization tests.*
3. *T is the compact positive operator row, read through* $\mathsf{CompactOperator}^{\uparrow}$*, recording the finite operator graph windows and the proof rows that $T(C) \subseteq C$.*
4. *Q is the compact orbit-window row, listing the finitely many normalized iterate, image, and enclosure rows inspected by the certificate.*
5. *ρ is the spectral-bound row, sealed through* $\mathsf{Real}^{\uparrow}$*, naming only the displayed radius comparison and not a selected host supremum.*
6. *V is the positive-eigenvector handoff row, consisting of a displayed nonzero cone element, its normalized readback, and the finite equality route asserting that T carries it to the scalar multiple named by ρ.*
7. *H records componentwise* hsame *transport for B, C, T, Q, ρ, V and for the replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from ordered Banach and cone reads through the compact positive operator, compact orbit window, spectral-bound row, and positive-eigenvector handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{KreinRutman}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KreinRutman}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, C, T, Q, \rho, V, H, R, P, N$. It has no coordinate for an unrestricted spectrum, a host compactness theorem, a selected cone interior point, a quotient eigenline equality, countable choice, or ambient real completeness.

Theorem 7466.2 (Krein–Rutman NameCert obligations)**.** *For every accepted* $\mathsf{KreinRutman}^{\uparrow}$ *carrier*

$$K = (B, C, T, Q, \rho, V, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KreinRutman}^{\uparrow}}$ *surface consists exactly of ordered Banach admission for B, positive cone admission for C, compact positive operator admission for T, compact orbit-window exposure for Q, real spectral-bound sealing for ρ, positive-eigenvector handoff for V, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7466.1. The visible rows are precisely the ordered Banach row, cone row, compact positive operator row, compact orbit window, spectral-bound row, positive-eigenvector handoff row, transport, replay, provenance, and local naming.

The operator route is finite. The row T records only compact positive operator graph windows and cone-preservation rows, while Q records only the normalized orbit and enclosure data admitted by the certificate.

The spectral read is also displayed. The row ρ is sealed through $\mathsf{Real}^{\uparrow}$, and the handoff row V records a nonzero cone element together with the finite route asserting the scalar-multiple read against T and ρ.

Finally H, R, P, N transport, replay, source, and name the same rows. Since the carrier lacks coordinates for unrestricted spectra, host compactness, selected cone interiors, quotient eigenlines, countable choice, and ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 7466.3 (Krein–Rutman positive-eigenvector handoff)**.** *Every consumer that reads an accepted* $\mathsf{KreinRutman}^{\uparrow}$ *packet as positive-eigenvector evidence must first expose the ordered Banach row B, positive cone row C, compact positive operator row T, compact orbit window Q, real spectral-bound row ρ, positive-eigenvector handoff row V, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The handoff exports only this finite BEDC route and does not export a host spectral-radius theorem, an unrestricted eigenbasis, a quotient eigenline, a chosen cone interior, or a completeness principle.*

Proof. Start from the NameCert obligations of Definition 7466.2. They force any accepted read through the rows $B, C, T, Q, \rho, V, H, R, P, N$.

The positive-eigenvector part is reached only after the operator and cone rows are visible. The cone row C supplies nonzero and positivity tests, while T supplies the compact positive operator windows that preserve the displayed cone.

Next read Q, ρ, V. The compact orbit window provides the finite approximation and enclosure route, the real spectral-bound row seals the scalar comparison, and V records the displayed vector handoff against that same operator and scalar.

The structural rows H, R, P, N preserve the same finite route for downstream consumers. No listed coordinate is a host spectrum, unrestricted eigenbasis, quotient eigenline, selected cone interior, or ambient completeness principle, so the handoff is exactly the finite positive-eigenvector certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreinRutman}^{\uparrow}}$.

Constructive story: A $\mathsf{KreinRutman}^{\uparrow}$ packet is generated from ordered Banach, positive cone, compact positive operator, compact orbit-window, real spectral-bound, positive-eigenvector handoff, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KreinRutman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7466.1, Definition 7466.2, and Definition 7466.3 seed the compact positive operator NameCert surface for the finite positive-eigenvector handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KreinRutman}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Krein–Rutman theorem, spectral-radius existence, unrestricted eigenbasis, quotient eigenline, chosen cone interior, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact positive operator routing, cone preservation, spectral-bound sealing, positive-eigenvector handoff, transport, replay, provenance, and local naming for $B, C, T, Q, \rho, V, H, R, P, N$.

Chapter 7467

A Concrete Naming Certificate for $\mathsf{UpperSemicontinuity}^{\uparrow}$

Following Bourbaki and Kelley on topological threshold tests and Bishop–Bridges constructive real analysis, the $\mathsf{UpperSemicontinuity}^{\uparrow}$ packet records upper semicontinuity as finite BEDC NameCert data. It sits beside the real-name carrier of chapter 38, the Real-and-Completion lower-threshold sibling of chapter 3935, and the epigraph boundary of chapter 6224. The packet names only displayed upper-threshold rows for real-valued reads; it is not a host theorem about limits, a selection principle for suprema, a countable-choice schedule, or an ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Definition 7467.1 (UpperSemicontinuity carrier)**.** *An* $\mathsf{UpperSemicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, T, F, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the source row whose inspected points are named through finite BEDC histories rather than host points.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *threshold row whose basic neighborhoods are displayed before an upper cut is accepted.*
3. *F is the value-row handoff for the real-valued map under inspection.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the source and value windows used by each threshold query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the real-name approximants consumed by the threshold test.*
6. *A is the upper-threshold ledger, recording finite evidence for requests of the form $f(x) < q$ and the corresponding open upper-cut route.*
7. *H records componentwise* hsame *transport for source, topology, value, schedule, readback, upper-threshold, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an upper-threshold request through $X, T, F, W, R, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UpperSemicontinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, with sibling pointers to the lower-semicontinuity and epigraph packets.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperSemicontinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for semantic limits, selected suprema, quotient equality of functions or real names, countable choice, host topological completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7467.2 (UpperSemicontinuity NameCert obligations)**.** *For every accepted* $\mathsf{UpperSemicontinuity}^{\uparrow}$ *carrier $U = (X, T, F, W, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UpperSemicontinuity}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier admission for the displayed source, topology, value, schedule, real-readback, and upper-threshold rows;*
2. *upper-threshold exposure through A, requiring each request $f(x) < q$ to be read only after X, T, F, W, R are visible;*
3. *real-name sealing through R and the sibling* $\mathsf{Real}^{\uparrow}$ *boundary of chapter 38;*
4. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
5. Pkg *provenance and local naming through P and N, including the comparison surface beside chapter 3935 and chapter 6224.*

Proof. Project the carrier of Definition 7467.1. The only analytic rows are X, T, F, W, R, A, so carrier admission and threshold exposure must be read from those displayed coordinates.

The upper-threshold row is not a hidden limit principle. A request $f(x) < q$ first exposes the source and topology rows X, T, then the value handoff F, then the stream schedule and regular-rational readback W, R, and only then the upper ledger A.

The structural rows preserve the same route. Transport through H and replay through C move the displayed rows componentwise, while P and N source and name the packet without adding a second mathematical coordinate.

Thus the five listed obligation families exhaust the local NameCert surface. Since the excluded limit, supremum-selection, quotient, choice, host-completeness, and ambient-completeness principles are absent from the carrier, no accepted read may recover them. □

Theorem 7467.3 (UpperSemicontinuity real seal scope)**.** *Every Real-facing consumer of an accepted* $\mathsf{UpperSemicontinuity}^{\uparrow}$ *packet reads the upper-threshold route through*

$$X, T, F, W, R, A \longmapsto H, C, P, N.$$

The route may compare its upper-cut ledger with the lower-threshold and epigraph siblings, but it reaches $\mathsf{Real}^{\uparrow}$ *only through displayed real-name readback rows and local NameCert provenance. It does not supply semantic limsup data, host limits, selected suprema, countable choice, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the threshold request. The source and topology rows X, T fix the finite neighborhood surface, and the value row F fixes the displayed map read.

Next use the Real-facing readback. The stream schedule W selects the finite windows, while R exposes regular-rational approximants at the real-name boundary. The upper-threshold ledger A is read only against those displayed rows.

Finally apply the structural surface. The rows H, C, P, N transport, replay, source, and name the same packet, with sibling comparison pointers to lower semicontinuity and epigraph data only as provenance. Because no excluded principle appears as a row, the Real-facing seal remains scoped to the displayed $\mathsf{UpperSemicontinuity}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UpperSemicontinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{UpperSemicontinuity}^{\uparrow}$ packet is generated from source rows, topology threshold rows, real-valued map handoffs, StreamName schedules, RegSeqRat readbacks, upper-threshold ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperSemicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7467.1, Definition 7467.2, and Definition 7467.3 seed the finite upper-threshold packet over displayed topology, real-name readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpperSemicontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close semantic limsup data, host limits, selected suprema, countable choice, quotient equality of functions or real names, host topological completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-threshold exposure, real-name readback, transport replay, provenance, local naming, and non-escape for BEDC.Derived.UpperSemicontinuityUp.

Chapter 7468

A Concrete Naming Certificate for $\mathsf{ParallelReduction}^{\uparrow}$

Following Barendregt and Takahashi on parallel beta reduction, the $\mathsf{ParallelReduction}^{\uparrow}$ packet records the finite simultaneous-contraction surface used by confluence arguments before any quotient of lambda terms is exposed. It sits after the lambda-calculus carrier of chapter 200 and beside the MetaCIC parallel-diamond frontier of chapter 4238. The packet gives the traditional route through reflexive constructor cases, simultaneous redex contraction, residual bookkeeping, transport, replay, provenance, and local naming; it does not assert full Church–Rosser, subject reduction, quotient equality of terms, global normalization, or kernel consistency.

Definition 7468.1 (Parallel reduction carrier)**.** *A* $\mathsf{ParallelReduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, B, A, S, R, J, H, C, K, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{LambdaCalc}^{\uparrow}$ *term-packet row supplying the source and target term syntax.*
2. *B is the one-step beta row whose redex witnesses may be contracted simultaneously.*
3. *A is the alpha-stability row keeping binder names and capture-avoidance ledgers visible during the contraction.*
4. *S is the simultaneous-contraction schedule, including reflexive constructor cases for variables, abstraction, and application.*
5. *R is the residual ledger recording which redex rows survive, disappear, or are copied across the displayed constructor route.*
6. *J is the local join candidate row used by downstream diamond consumers, without asserting that every peak has already been joined.*
7. *H records componentwise* hsame *transport for the term, beta, alpha, schedule, residual, join, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay through the same rows.*
9. *K is the* Pkg *provenance over* $\mathsf{ParallelReduction}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ParallelReduction}^{\uparrow}}$ *evidence row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for quotient term equality, subject reduction, evaluator equality, global normalization, full Church–Rosser, or MetaCIC kernel consistency.

Theorem 7468.2 (Parallel reduction NameCert obligations)**.** *For every accepted* $\mathsf{ParallelReduction}^{\uparrow}$ *carrier* $P = (T, B, A, S, R, J, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ParallelReduction}^{\uparrow}}$ *surface consists exactly of carrier admission, lambda-term exposure, one-step beta row visibility, alpha-stable simultaneous contraction, residual ledger preservation, local join-candidate exposure, componentwise transport, displayed replay, provenance, and local naming. A MetaCIC confluence consumer may read this packet as the traditional parallel-reduction surface feeding chapter 4238, but it cannot read it as full Church–Rosser, subject reduction, quotient equality of terms, global normalization, or kernel consistency.*

Proof. Carrier admission is projection from Definition 7468.1. The rows $T, B, A, S, R, J, H, C, K, N$ are the whole visible surface, and the lambda-calculus source row keeps the term syntax inside the existing carrier of chapter 200.

The simultaneous-contraction obligation reads B, A, S together. The beta row supplies the redex witnesses, the alpha row keeps binder transport and capture-avoidance visible, and the schedule row records the constructorwise reflexive and contraction cases.

The residual and join-facing obligations then read R and J. These rows expose the data a parallel-diamond consumer may inspect, while retaining the conditional frontier: the packet names candidate residuals and local joins without asserting that all closed peaks have been joined.

Finally, H, C, K, N transport, replay, source, and certify exactly the displayed rows. Since quotient equality, subject reduction, evaluator equality, global normalization, full Church–Rosser, and kernel consistency are absent from the carrier, none of those conclusions is exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ParallelReduction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ParallelReduction}^{\uparrow}$ packet is generated from lambda-term syntax rows, one-step beta rows, alpha-stability rows, simultaneous-contraction schedules, residual ledgers, local join candidates, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ParallelReduction}^{\uparrow}}$, seed)

Scope closed: Definition 7468.1 and Definition 7468.2 seed the finite parallel beta-reduction route consumed by MetaCIC parallel-diamond frontier packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ParallelReduction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full Church–Rosser, subject reduction, quotient equality of terms, global normalization, kernel consistency, Lean verification, or bridge checking for $\mathsf{ParallelReduction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, simultaneous-contraction schedule exactness, residual ledger preservation, local join-candidate confinement, transport, replay, provenance, and local naming for BEDC.Derived.ParallelReductionUp.

Chapter 7469

A Concrete Naming Certificate for $\mathsf{ClosedInductionTrace}^{\uparrow}$

The $\mathsf{ClosedInductionTrace}^{\uparrow}$ packet realizes the closed-observation and induction-boundary surface of chapter 9393 as finite NameCert data. It sits beside the finite trace GAP packet of chapter 3353, the inscription point packet of chapter 610, and the typed GAP socket taxonomy of chapter 1240. The packet records observation rows, input-boundary rows, inscription rows, replay rows, and non-escape rows without installing a substrate-internal induction rule.

Definition 7469.1 (Closed induction trace carrier)**.** *A* $\mathsf{ClosedInductionTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, F, G, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the closed observation row: a finite family of records already present in the observational substrate.*
2. *F is the finite verification trace row, recording observed base and successor checks only as bounded data.*
3. *G is the induction GAP row, marking the missing universal closure without treating it as an accepted substrate rule.*
4. *B is the input-boundary row identifying where a forward-binding belief would be supplied from outside the closed observation surface.*
5. *S is the inscription row connecting that boundary to an inscription-point consumer without adding a new proof rule.*
6. *R is the replay row that shows which finite observations can be read again by a downstream audit consumer.*
7. *E is the non-escape row blocking a read from O, F, G, B, S, R to a universal induction theorem inside the substrate.*
8. *H records componentwise* hsame *transport for the observation, trace, GAP, boundary, inscription, replay, non-escape, provenance, and naming rows.*
9. *C records displayed* Cont *replay through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{ClosedInductionTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the sibling GAP and inscription packets.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedInductionTrace}^{\uparrow}}$ *evidence row.*

The carrier has no coordinate for substrate-internal induction, unbounded universal quantification, countable choice, host natural-number recursion beyond displayed rows, or a hidden source of mathematical belief.

Theorem 7469.2 (Closed induction trace NameCert obligations)**.** *For every accepted* $\mathsf{ClosedInductionTrace}^{\uparrow}$ *carrier* $I = (O, F, G, B, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClosedInductionTrace}^{\uparrow}}$ *surface consists exactly of closed observation exposure, finite trace admission, induction-GAP retention, input-boundary naming, inscription-row visibility, replay confinement, non-escape, componentwise transport, displayed replay, provenance, and local naming. The obligations realize closed observation, input boundary, inscription, replay, and non-escape rows as finite data; they do not provide a substrate-internal induction schema.*

Proof. Carrier admission is projection from Definition 7469.1. The observation row and finite trace row provide only bounded records already present in the closed observational surface.

The boundary step reads G, B, S together. The GAP row keeps the missing universal closure visible, the boundary row names the outside-supply site, and the inscription row records the local reception point without adding a rule to the substrate.

Replay and non-escape then read R, E. A downstream audit may replay displayed observations and trace rows, but the non-escape row blocks replacement of those finite records by a universal induction theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite rows. Since substrate-internal induction, unbounded universal quantification, countable choice, host recursion beyond displayed rows, and hidden belief supply are absent from the carrier, none can be exported by the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{ClosedInductionTrace}^{\uparrow}$ packet is refuted if a consumer erases the induction GAP row or reads the input-boundary row as an internal induction rule while preserving the same finite observation trace.

Independence witness: finite-trace-induction-gap, inscriptionpoint, typedgapsockettaxonomy, closedobservationsystem: none is bijective to $\mathsf{ClosedInductionTrace}^{\uparrow}$ because this carrier jointly binds closed observation, finite trace, induction GAP, input boundary, inscription, replay, non-escape, transport, provenance, and local naming rows.

The Lean follow-up must provide explicit Nontrivial $\mathsf{ClosedInductionTrace}^{\uparrow}$ and FieldFaithful ClosedInductionTrace instances before a TasteGate theorem can accept the displayed carrier. The nontrivial witness must separate a packet retaining G, B, S, E from one that erases the induction boundary, and the field-faithfulness witness must preserve $O, F, G, B, S, R, E, H, C, P, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedInductionTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedInductionTrace}^{\uparrow}$ packet is generated from closed observation rows, finite verification traces, induction GAP rows, input-boundary rows, inscription rows, replay rows, non-escape rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedInductionTrace}^{\uparrow}}$, seed)

Scope closed: Definition 7469.1 and Definition 7469.2 seed the finite closed-induction trace route from observation through boundary, inscription, replay, and non-escape rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedInductionTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close substrate-internal induction, unbounded universal quantification, countable choice, host recursion beyond displayed rows, hidden belief supply, Lean verification, or bridge checking for ClosedInductionTrace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, induction-GAP retention, input-boundary naming, inscription-row visibility, replay confinement, non-escape, Nontrivial and FieldFaithful TasteGate instances, transport, provenance, and local naming for BEDC.Derived.ClosedInductionTraceUp.

Chapter 7470

A Concrete Naming Certificate for MetaCICBetaRedexSpine$^{\uparrow}$

The MetaCICBetaRedexSpine$^{\uparrow}$ packet realizes the beta-redex spine branch of chapter 9384 as finite NameCert data over the checked MetaCIC beta reduction grammar. It sits beside the beta-step boundary of chapter 754, the beta-reduction audit trail of chapter 879, the beta critical-pair packet of chapter 881, and the subject-reduction boundary of chapter 721. The packet records the displayed redex spine and contraction handoff without claiming subject reduction, confluence, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 7470.1 (MetaCIC beta redex spine carrier)**.** *A* MetaCICBetaRedexSpine$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, \Gamma, t, u, \beta, \chi, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC* Term *row naming the source term, its visible application spine, and the binder depth at which the inspected redex occurs.*
2. *Γ is the local context and binder-window row, recording the finite variables, displayed domains, and shift boundary available to the contraction.*
3. *t is the redex row, exposing the displayed* app(lam *dom body*) *arg site, its subterm address, and the surrounding spine segments.*
4. *u is the contracted target row, naming the* substitute 0 *arg body result and the reassembled target term inside the same spine.*
5. *β is the* BetaStep *and* BetaConv *row, recording the one-step contraction and the common-reduct conversion interface available after the step.*
6. *χ is the obstruction-retention row for substitution preservation, dependent codomain stability, binder-domain subject reduction, subject reduction, confluence, evaluator equality, quotient syntax equality, and kernel consistency.*
7. *H records componentwise* hsame *transport for $M, \Gamma, t, u, \beta, \chi$ and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source* Term *and binder window through the redex address, contraction target,* BetaStep *witness,* BetaConv *handoff, and obstruction-retention row.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$, Term, BetaStep, BetaConv, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICBetaRedexSpine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, \Gamma, t, u, \beta, \chi, H, C, P, N$. It has no coordinate for an unbounded evaluator, a subject-reduction theorem, a full confluence proof, quotient equality of terms, or kernel consistency.

Theorem 7470.2 (MetaCIC beta redex spine NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$ *carrier*

$$S = (M, \Gamma, t, u, \beta, \chi, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICBetaRedexSpine}^{\uparrow}}$ *surface consists exactly of source* Term *admission for M, local binder-window admission for Γ, redex-site exposure for t, contraction-target exposure for u,* BetaStep *and* BetaConv *routing through β, obstruction retention through χ, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7470.1. The displayed rows are $M, \Gamma, t, u, \beta, \chi, H, C, P, N$.

The syntax rows come first. The source term row M, context row Γ, redex row t, and target row u name the inspected application-lambda site, the binder window, and the reassembled contracted term.

The reduction row is then read through β. It records the BetaStep contraction and the BetaConv handoff made available by the beta conversion interface, while χ keeps the subject-reduction and confluence endpoints outside the packet.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for an evaluator, quotient syntax equality, subject reduction, full confluence, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 7470.3 (MetaCIC beta redex spine contraction handoff)**.** *Every consumer that reads an accepted* $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$ *packet as a beta contraction handoff must expose the source* Term *row, binder-window row, displayed redex site, contracted target,* BetaStep *witness,* BetaConv *interface, obstruction-retention row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before using the handoff in subject-reduction or confluence work. The handoff supplies a finite contraction spine only; it does not discharge substitution preservation, dependent-codomain stability, binder-domain preservation, subject reduction, confluence, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Start from the NameCert obligations of Definition 7470.2. They force every accepted contraction read through $M, \Gamma, t, u, \beta, \chi, H, C, P, N$.

The redex and contraction rows determine the local spine. The row t names the application-lambda site and its address, while u names the term obtained by the displayed substitution and reassembly route.

The beta interface is finite. The row β records the one-step BetaStep witness and the available BetaConv comparison route, while χ keeps the substitution, dependent-codomain, binder-domain, subject-reduction, confluence, evaluator, quotient, and consistency obligations visible as non-exported endpoints.

Finally the structural rows transport, replay, source, and name the same finite contraction spine. Thus downstream subject-reduction and confluence packets may consume the handoff as a local beta-redex route, but not as any of the stronger typing or semantic theorems. □

Falsifiable prediction: If the redex-site row t is removed, two packets with different application-lambda addresses but the same contracted target can be classified together; if the obstruction-retention row χ is removed, the contraction handoff can be misread as subject-reduction evidence.

Independence witness: betastepboundary, betareductionauditrail, beta_critical_pair, typedbetacongruencetrigger, and metacic_subjectreductionboundary are nearby MetaCIC siblings, but none is bijective to $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$ because this packet jointly carries the source term spine, binder window, redex address, contracted target, BetaStep witness, BetaConv route, obstruction retention, transport, replay, provenance, and local naming.

Lean TasteGate support for $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$ will provide Nontrivial and FieldFaithful instances so the source term, binder window, redex site, contraction target, beta relation, obstruction, transport, replay, provenance, and local naming fields remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICBetaRedexSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICBetaRedexSpine}^{\uparrow}$ packet is generated from MetaCIC Term, binder-window, redex-site, contracted-target, BetaStep, BetaConv, obstruction-retention, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICBetaRedexSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7470.1, Definition 7470.2, and Definition 7470.3 seed the finite beta-redex spine surface over Term, BetaStep, BetaConv, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICBetaRedexSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close substitution preservation, dependent-codomain stability, binder-domain preservation, subject reduction, confluence, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, redex-site exactness, contraction-target exactness, BetaStep routing, BetaConv routing, obstruction retention, transport, replay, provenance, Nontrivial, FieldFaithful, and local naming for $M, \Gamma, t, u, \beta, \chi, H, C, P, N$.

Chapter 7471

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$

Following Bishop and Bridges on metric completions, located subsets, and regular real names, the $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$ packet records the finite route by which a metric source is embedded into a displayed located completion without quotienting Cauchy sequences or selecting host limits. It sits beside the metric carrier of chapter 57, the completion packet of chapter 95, the metric-completion service of chapter 1506, the separated-metric reflection of chapter 1790, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the embedding as finite NameCert data; it is not a quotient completion, a host equality principle, a choice-selected limit, trichotomy, or an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7471.1 (Bishop located completion embedding carrier)**.** *A* $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(M, L, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, exposing the* $\mathsf{Met}^{\uparrow}$ *distance, symmetry, triangle, and zero-distance comparison used by the embedding.*
2. *L is the located positive-radius row, recording the finite rational-radius evidence that a source point has a displayed neighbourhood before any completion read is made.*
3. *D is the dense completion route, carrying the* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *approach map from the source metric into the completion packet.*
4. *E is the embedding graph row, pairing each inspected source point with its completion name and recording graph membership before equality is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row, exposing only regular rational distance reads and the terminal real seal of the embedded point.*
6. *H records componentwise* hsame *transport for the metric source, located-radius row, dense route, embedding graph, readback, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a source-point request through located-radius inspection, dense-route lookup, embedding-graph readback, regular rational read, and real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completions, host equality of representatives, choice-selected limits, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BishopLocatedCompletionEmbeddingUp.BishopLocatedCompletionEmbeddingCarrier`

Theorem 7471.2 (Bishop located completion embedding NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$ *carrier* $(M, L, D, E, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompl}}$ *surface consists exactly of carrier admission for the metric source, located positive-radius evidence, dense completion route, embedding graph, regular rational and real readback, componentwise transport, displayed continuation replay, provenance, and local naming. Every embedding read must pass through* $M, L, D, E, R, H, C, P, N$*, and no obligation is stored outside those finite rows.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionEmbeddingUp.TasteGate.BishopLocatedCompletionEmbeddingTaste`
`single carrier alignment`

Proof. Project the accepted carrier. The displayed rows are precisely the metric source M, located-radius evidence L, dense completion route D, embedding graph E, regular rational and real readback R, and the structural rows H, C, P, N.

The located part is read before the completion part. A source-point request first enters the metric row and the positive-radius ledger, so the later dense-route lookup is tied to displayed local evidence rather than to an external completion representative.

The embedding graph row then records the finite source-to-completion relation, and the readback row exposes only regular rational distance reads and the terminal $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport and displayed replay preserve this same order.

Since the carrier contains no quotient completion row, host equality coordinate, selected-limit coordinate, trichotomy split, or ambient real-completeness row, none of those objects can be projected from the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7471.1 and Definition 7471.2 seed the located completion embedding route over metric source rows, located positive-radius evidence, dense completion routes, embedding graph rows, regular rational readback, real seals, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, host equality of representatives, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-radius exposure, dense-route exactness, embedding-graph readback, regular-real handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompletionE

Constructive story: A $\mathsf{BishopLocatedCompletionEmbedding}^{\uparrow}$ packet is generated from a Metric source row, located positive-radius evidence, a dense Completion or MetricCompletion route, an embedding graph, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7472

A Concrete Naming Certificate for $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges compact metric analysis, the $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$ packet realizes the compact uniform-continuity route of chapter 9237 as graph-level finite NameCert data. It sits beside the compact metric source of chapter 129, the continuous-map packet of chapter 130, the metric-distance rows of chapter 57, the rational tolerance fold of chapter 37, and the uniform-modulus consumer of chapter 369. The packet names the finite graph and modulus ledger; it is not open-cover compactness, host function extensionality, quotient map equality, a selected external modulus, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7472.1 (Uniform-continuity graph modulus carrier)**.** *A* $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(K, G, L, Q, S, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, exposing bundled source centers, coverage witnesses, and finite* ProbeBundle *membership data.*
2. *G is the continuous graph row, pairing source and target metric reads through* $\mathsf{ContinuousMap}^{\uparrow}$ *before any modulus is exported.*
3. *L is the pointwise-modulus ledger attached to the graph row at each bundled center.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the positive radii carried by L.*
5. *S is the source* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *triangle row used to move nearby source points into a common local graph neighbourhood.*
6. *T is the target metric-triangle row used to combine the two pointwise target estimates.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, exposing only a tolerance request and the folded precision obtained from Q.*
8. *H records componentwise* hsame *transport for compact nets, graph rows, local modulus ledgers, rational folds, metric-triangle rows, uniform-modulus rows, replay, provenance, and naming.*
9. *C records displayed* Cont *replay from tolerance request through finite-net coverage, graph lookup, pointwise-modulus readback, rational fold, source and target triangle rows, and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityGraphModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, host function extensionality, quotient maps, selected external moduli, countable choice, or ambient real completeness.

Theorem 7472.2 (Uniform-continuity graph modulus NameCert obligations)**.** *For every accepted* $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$ *carrier* $(K, G, L, Q, S, T, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityGraphModu}}$ *surface consists exactly of carrier admission for compact finite nets, continuous graph rows, pointwise-modulus ledgers, rational lower-bound folds, source and target metric-triangle rows, uniform-modulus handoff, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-uniform-continuity read must pass through those rows before the public* $\mathsf{UniformModulus}^{\uparrow}$ *output is available.*

Proof. Project the accepted packet. Its analytic rows are the compact finite-net row K, continuous graph row G, pointwise-modulus ledger L, rational fold Q, source and target metric rows S, T, and uniform-modulus handoff U.

Finite-net coverage comes first. A source point is matched to a bundled center through K, and the graph row G supplies only the displayed continuous-map relation at that center.

The local modulus ledger L provides positive radii and target estimates for the graph row. The rational fold Q is read by primitive recursion over those finitely many radii, yielding a single precision below every displayed local radius.

The source triangle row S moves a nearby pair of source points into the same local graph neighbourhood, and the target triangle row T combines the resulting target estimates. Only then does U expose the tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Rows H, C, P, N transport, replay, source, and name the same route. Since the packet has no coordinate for open-cover compactness, host function extensionality, quotient maps, selected external moduli, countable choice, or ambient completeness, those objects cannot escape through the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuityGraphModulus}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformContinuityGraphModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7472.1 and Definition 7472.2 seed the compact finite-net, continuous graph, pointwise-modulus, ProbeBundle fold, metric-triangle, and uniform-modulus handoff route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformContinuityGraphModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host function extensionality, quotient map equality, selected external moduli, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net coverage, graph lookup, pointwise-modulus exactness, rational-fold lower bounds, source-target triangle handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformContinuityGraphModulusUp.

Constructive story: A $\mathsf{UniformContinuityGraphModulus}^{\uparrow}$ packet is generated from compact finite-net rows, continuous graph rows, pointwise-modulus ledgers, RatUp lower-bound folds, source and target MetricSpace triangle rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7473

A Concrete Naming Certificate for BishopLocatedIntervalRefinementMap$^{\uparrow}$

Following Bishop and Bridges constructive interval analysis, the BishopLocatedIntervalRefinementMap$^{\uparrow}$ packet names the finite map surface that sends one located interval certificate to a displayed refined interval certificate. It reads the Bishop refinement carrier of chapter 7391, the generic refinement packet of chapter 6199, the located interval source rows of chapter 394, the dyadic endpoint window of chapter 1524, and the terminal real seal of chapter 38. The map is recorded as finite certificate data, not as an ambient interval operation supplied by the host.

Definition 7473.1 (Bishop located interval refinement map carrier)**.** *A* BishopLocatedIntervalRefinementMap$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, J, R, D, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* LocatedInterval$^{\uparrow}$ *row, including lower and upper endpoint names, located membership comparison rows, and the displayed interval non-emptiness witness.*
2. *J is the target* LocatedInterval$^{\uparrow}$ *row obtained after the refinement request is read.*
3. *R is the* LocatedIntervalRefinement$^{\uparrow}$ *row asserting the finite inclusion route from J into I with the displayed endpoint and membership comparisons.*
4. *D is the* BishopLocatedIntervalRefinement$^{\uparrow}$ *row, tying the refinement request to Bishop-style located interval data rather than to a host interval operator.*
5. *W is the* DyadicRoundingWindow$^{\uparrow}$ *row that transports the source and target endpoints through one displayed dyadic precision window.*
6. *S is the StreamName and RegSeqRat readback seal used by endpoint consumers before the Real row is read.*
7. *E is the* Real$^{\uparrow}$ *seal row for the endpoint and width data after dyadic readback.*
8. *H records componentwise* hsame *transport for source interval, target interval, refinement inclusion, dyadic endpoint window, readback seal, real seal, continuation replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from the source interval to the refinement request, from the request to the target interval, from source and target endpoints to dyadic rounding windows, and from dyadic readback to the Real-facing rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedIntervalRefinementMap}^{\uparrow}$, $\mathsf{BishopLocatedIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementMap}^{\uparrow}}$ *row.*

The classifier compares accepted map packets only through $I, J, R, D, W, S, E, H, C, P, N$. It has no coordinate for arbitrary interval choice, quotient real equality, selected limits, host rounding, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BishopLocatedIntervalRefinementMapUp.BishopLocatedIntervalRefinementMapTasteGate`
`single carrier alignment`

Theorem 7473.2 (Bishop located interval refinement map obligations)**.** *For an accepted* BishopLocatedIntervalRefi *packet $M = (I, J, R, D, W, S, E, H, C, P, N)$, the local* NameCert *obligation surface consists exactly of source interval projection, target interval projection, refinement inclusion, dyadic endpoint transport, width decrease, StreamName readback, RegSeqRat handoff, Real seal factorization, componentwise transport, displayed replay, provenance non-escape, and local naming.*

Proof. Project the carrier of Definition 7473.1. The source row I and target row J are both located interval packets, while R records the inclusion of the target interval into the source interval and D records that the route is the Bishop located refinement route.

Endpoint and width data are transported through W. The dyadic window reads the displayed endpoint precision, carries the adjacent endpoint packet, and exposes the finite width decrease before any real-facing consumer is allowed to read E.

The readback handoff is exact. The row S supplies the StreamName and RegSeqRat route for the same endpoint observations, and E seals only those rows as $\mathsf{Real}^{\uparrow}$ data. No selected limit or quotient real equality row is present.

Finally exhaust H, C, P, N. Transport, replay, provenance, and local naming all name the same finite packet, so no arbitrary interval operator, host rounding operation, or ambient completeness principle can enter the local obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementMap}^{\uparrow}}$.

Constructive story: A Bishop located interval refinement map is generated from a source located interval, a target located interval, a displayed refinement inclusion, dyadic endpoint transport, StreamName and RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementMap}^{\uparrow}}$, seed)

Scope closed: Definition 7473.1 and Definition 7473.2 seed the finite refinement-map carrier from source intervals to displayed refined intervals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary interval selection, quotient real equality, selected limits, host rounding, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source and target interval projection, refinement inclusion, dyadic endpoint transport, width decrease, StreamName and RegSeqRat readback, Real seal factorization, provenance non-escape, and local NameCert exactness for BEDC.Derived.BishopLocatedIntervalRefinementMapUp.

Chapter 7474

A Concrete Naming Certificate for $\mathsf{CompactUniformTriangleEstimateLedger}^{\uparrow}$

The $\mathsf{CompactUniformTriangleEstimateLedger}^{\uparrow}$ packet records the constructive Heine–Cantor source-target triangle estimate of chapter 9237 as finite NameCert data. It reads the compact metric source of chapter 129, the continuous map row of chapter 130, the metric law carrier of chapter 57, the tolerance-only handoff of chapter 369, and the compact-uniform-continuity packet of chapter 388. The ledger isolates the final pair of metric-triangle estimates so downstream consumers can cite the estimate without reopening the full compactness proof.

Definition 7474.1 (Compact uniform triangle estimate ledger carrier)**.** *A* $\mathsf{CompactUniformTriangleEstimateLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, U, M_X, M_Y, c, \rho, \varepsilon, \eta, S, Y, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row containing the displayed finite net and center membership ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus rows are read at the chosen compact-net centers.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced after the finite estimate is assembled.*
4. *M_X and M_Y are the source and target* $\mathsf{Met}^{\uparrow}$ *law rows, including the triangle rows used in the estimate.*
5. *c is the compact-net center selected from the finite net row, and ρ is the pointwise radius row attached to c by F.*
6. *ε is the requested target tolerance, and η is the displayed epsilon-third budget used by the ledger.*
7. *S records the source-triangle estimate joining x, the center c, and the comparison point x' inside the same local source ball.*
8. *Y records the target-triangle estimate joining $f(x)$, $f(c)$, and $f(x')$ within the displayed target tolerance.*
9. *H records componentwise* hsame *transport for compact-net rows, pointwise modulus rows, source metric triangle rows, target metric triangle rows, epsilon-third budget rows, UniformModulus handoff rows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from compact-net coverage to pointwise modulus, from source-triangle estimates to target-triangle estimates, and from the target estimate to the UniformModulus handoff.*

11. *P is the* Pkg *provenance over* CompactUniformTriangleEstimateLedger$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CompactUniformTriangleEstimateLedger}^{\uparrow}}$ *row.*

The classifier has no coordinate for sequential compactness, selected subsequences, global choice of centers, quotient equality of metric points, or an ambient continuity principle outside the displayed rows.

Theorem 7474.2 (Compact uniform triangle estimate ledger obligations)**.** *For an accepted* CompactUniformTriangl *packet $T = (K, F, U, M_X, M_Y, c, \rho, \varepsilon, \eta, S, Y, H, C, P, N)$, the local obligation surface is exactly compact net center projection, pointwise modulus readback, source metric triangle transport, target metric triangle transport, epsilon-third splitting, UniformModulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Project the compact metric row K and choose the displayed finite-net center c from its carrier rows. The continuous map row F then supplies the pointwise modulus row ρ at that center, so the source estimate starts with finite compact-net data rather than a selected subsequence.

Read the source metric triangle row S from M_X. It combines the distance from x to c with the distance from x to x' inside the local source ball named by ρ.

Read the target metric triangle row Y from M_Y. The epsilon-third budget η splits the requested tolerance ε into the pointwise modulus contribution and the two triangle contributions, so the target estimate closes by finite rational arithmetic and the displayed metric-law rows.

The row U is the only UniformModulus handoff. The transport row H, replay row C, provenance row P, and local naming row N preserve the same compact-net, pointwise-modulus, source-triangle, target-triangle, and tolerance-budget rows; hence no sequential compactness, global choice of centers, quotient equality, or ambient continuity principle escapes the ledger. □

Closure status for NameCert$_{\text{CompactUniformTriangleEstimateLedger}^{\uparrow}}$.

Constructive story: A compact uniform triangle estimate ledger is generated from compact-net center rows, pointwise modulus rows, source and target metric triangle rows, an epsilon-third tolerance budget, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformTriangleEstimateLedger}^{\uparrow}}$, seed)

Scope closed: Definition 7474.1 and Definition 7474.2 seed the finite source-target triangle estimate ledger for compact-uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformTriangleEstimateLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close sequential compactness, selected subsequences, global choice of centers, quotient metric equality, ambient continuity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-net center projection, pointwise modulus readback, source and target metric triangle transport, epsilon-third split, UniformModulus handoff, provenance non-escape, and local NameCert exactness for BEDC.Derived.CompactUniformTriangleEstimateLedgerUp.

Chapter 7475

A Concrete Naming Certificate for $\mathsf{PlanarGraphMinor}^{\uparrow}$

Following Diestel, the $\mathsf{PlanarGraphMinor}^{\uparrow}$ packet records the finite graph-minor route by which a displayed graph carrier is inspected through edge deletions, vertex deletions, contractions, and a planar embedding ledger. It sits over the finite graph carrier of chapter 118, the independence and rank bookkeeping of chapter 202, the incidence-pairing discipline of chapter 234, and the closed-trail parity surface of chapter 7092. The packet names a finite obstruction-facing carrier for later minor and planar-routing consumers; it is not a host Robertson–Seymour theorem, not a completed obstruction catalogue, and not an ambient topological embedding principle.

Definition 7475.1 (PlanarGraphMinor carrier)**.** *A* $\mathsf{PlanarGraphMinor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, D, V, K, F, E, R, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the source* $\mathsf{Graph}^{\uparrow}$ *row, including finite vertex rows, finite edge rows, endpoint readback, incidence rows, and the local graph classifier.*
2. *D is the edge-deletion ledger, listing the displayed edge rows removed before the minor read is admitted.*
3. *V is the vertex-deletion ledger, listing the displayed vertex rows and incident edge rows removed before the minor read is admitted.*
4. *K is the contraction ledger, recording each contracted edge, the two endpoint rows it identifies, and the resulting endpoint readback in the contracted graph row.*
5. *F is the planar-face ledger, recording finite cyclic boundary walks, incidence checks, and the local face-adjacency rows accepted as planar evidence.*
6. *E is the obstruction-boundary row, retaining the inspected finite subgraph windows used to refuse hidden nonplanar witnesses or unstated contractions.*
7. *R is the* $\mathsf{Matroid}^{\uparrow}$ *support row, reading cycle and rank-style dependency data only through the displayed graph edges.*
8. *A is the* $\mathsf{Matching}^{\uparrow}$ *and* $\mathsf{EulerianGraph}^{\uparrow}$ *sibling row, exposing incidence pairings, parity checks, and closed boundary-walk support used by the planar-face ledger.*

9. *H records componentwise* hsame *transport for G, D, V, K, F, E, R, A and for the replay, provenance, and naming rows.*
10. *C records displayed* Cont *replay from the graph row through deletion, contraction, face, obstruction-boundary, matroid-support, and sibling-support rows.*
11. *P is the* Pkg *provenance over* $\mathsf{PlanarGraphMinor}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{EulerianGraph}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PlanarGraphMinor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite graph, hidden embedding, quotient graph equality, unlisted contraction, host obstruction theorem, or completed minor-closed family.

Theorem 7475.2 (PlanarGraphMinor NameCert obligations)**.** *For every accepted* $\mathsf{PlanarGraphMinor}^{\uparrow}$ *carrier $M = (G, D, V, K, F, E, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PlanarGraphMinor}^{\uparrow}}$ *surface consists exactly of source graph admission through G, edge-deletion exposure through D, vertex-deletion exposure through V, contraction exposure through K, planar-face ledger exposure through F, obstruction-boundary retention through E, matroid-support exposure through R, matching and Eulerian sibling exposure through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7475.1. The visible rows are precisely the source graph, deletion ledgers, contraction ledger, planar-face ledger, obstruction-boundary row, matroid-support row, matching and Eulerian sibling row, transport, replay, provenance, and local naming.

The graph-minor operations are finite. The rows D and V list deleted edges and vertices, while K lists contracted edges together with the endpoint readback used after contraction. Each such read is downstream of the displayed $\mathsf{Graph}^{\uparrow}$ source row.

The planar and obstruction-facing rows are also displayed. The face ledger F records only finite boundary walks and incidence checks, while E retains the inspected windows that prevent a consumer from inserting hidden nonplanar witnesses or unstated contractions.

Finally R, A, H, C, P, N keep the same route attached to matroid support, matching incidence, Eulerian parity, transport, replay, provenance, and local naming. Since the carrier has no coordinate for hidden embeddings, quotient graph equality, infinite graphs, completed obstruction families, or host minor theorems, the listed rows exhaust the NameCert obligations. □

Theorem 7475.3 (PlanarGraphMinor finite contraction route)**.** *Every consumer that reads an accepted* $\mathsf{PlanarGraphMinor}^{\uparrow}$ *packet as planar minor evidence must first expose the finite route*

$$\mathsf{Graph}^{\uparrow} \longmapsto D, V, K \longmapsto F, E \longmapsto R, A \longmapsto H, C, P, N.$$

In particular, a contraction read can identify endpoints only through the displayed edge row in K, can inspect planarity only through the finite face ledger F, and can use matroid, matching, or Eulerian support only after the corresponding sibling rows have been displayed.

Proof. Start with a planar-minor consumer. By Definition 7475.1, the request reaches the minor packet only after the source graph row G has exposed its finite vertices, edges, endpoints, incidence data, and local classifier.

The minor route then passes through the operation ledgers. Edge deletion is read from D, vertex deletion from V, and contraction from K. The contraction row records the contracted edge and the endpoint pair, so no hidden quotient graph can replace the displayed endpoint readback.

Next expose the planar-face and obstruction-boundary rows. The row F supplies finite cyclic boundary walks and incidence checks, while E retains the inspected subgraph windows that delimit the obstruction-facing read.

The sibling support rows are downstream of the same route. The matroid row R reads only displayed graph-edge dependencies, and the matching/Eulerian row A reads only incidence pairings and parity-support data already visible in the packet. Transport, replay, provenance, and naming then preserve the same finite route, so every accepted contraction and planarity read factors through the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{PlanarGraphMinor}\uparrow}$.

Constructive story: A $\mathsf{PlanarGraphMinor}^{\uparrow}$ packet is generated from a finite GraphUp carrier, edge-deletion rows, vertex-deletion rows, contraction rows, planar-face ledgers, obstruction-boundary windows, MatroidUp support, MatchingUp and EulerianGraphUp sibling support, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PlanarGraphMinor}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7475.1, Definition 7475.2, and Definition 7475.3 seed the finite graph-minor and planar-face NameCert route over displayed GraphUp, MatroidUp, MatchingUp, and EulerianGraphUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PlanarGraphMinor}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the Robertson–Seymour theorem, a complete obstruction catalogue, host planar embedding equivalence, quotient graph equality, infinite graph minors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deletion exposure, contraction endpoint readback, planar-face ledger exactness, obstruction-boundary retention, sibling support routing, transport, replay, provenance, and local naming for BEDC.Derived.PlanarGraphMinorUp.

Chapter 7476

A Concrete Naming Certificate for $\mathsf{LebesgueCoveringLemma}^{\uparrow}$

Following Lebesgue and Bishop–Bridges compact metric analysis, the $\mathsf{LebesgueCoveringLemma}^{\uparrow}$ packet records the finite compact-cover route that turns displayed compact metric data into a finite positive-radius witness. It sits between the compact metric source of chapter 129, the metric-space rows of chapter 57, the rational and dyadic tolerance surfaces of chapter 37 and chapter 291, the finite-radius consumer of chapter 452, and compact-uniform-continuity consumers such as chapter 2292 and chapter 7456. The packet names only finite cover data, radius comparisons, componentwise transport, and local provenance; it is not open-cover compactness, a selected subcover principle, countable choice, or an ambient theorem about all metric spaces.

Definition 7476.1 (Lebesgue covering lemma carrier)**.** *A* $\mathsf{LebesgueCoveringLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, U, F, R, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing a named compact metric packet only through finite compact-cover data.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row whose point and ball readings are available to the cover test.*
3. *U is the finite cover row: a finite list of named cover centers, cover radii, and membership tests read from the compact source.*
4. *F is the finite subcover witness already displayed by U; it is a finite list coordinate, not a host-selected subcover.*
5. *R is the positive radius row, recorded through* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons.*
6. *D is the finite radius-bound ledger asserting that every point read through K, M lies in a cover member whose displayed radius is bounded below by the same positive row R.*
7. *H records componentwise* hsame *transport for the compact source, metric distance, finite cover, subcover list, rational radius, dyadic ledger, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point read through K, M to the finite cover row U, the finite list row F, the radius row R, and the bound ledger D.*
9. *P is the* Pkg *provenance over* $\mathsf{LebesgueCoveringLemma}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringLemma}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, a host-chosen subcover, countable choice, quotient metric equality, ambient compactness, or a global Heine–Borel theorem.

Theorem 7476.2 (Lebesgue covering lemma NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueCoveringLemma}^{\uparrow}$ *carrier* $L = (K, M, U, F, R, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringLemma}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier admission for the compact metric source, metric distance row, finite cover row, finite subcover list, positive radius row, and dyadic radius-bound ledger;*
2. *cover exposure, requiring each point read from K, M to enter the finite list row F only through the displayed cover row U;*
3. *finite positive-radius certification through R, D, with all comparisons recorded through* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
5. Pkg *provenance and local naming through P, N, including the handoff to* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *and compact-uniform-continuity consumers.*

No NameCert obligation is stored outside these rows, and no obligation selects a subcover from an arbitrary open cover.

Proof. Project the carrier of Definition 7476.1. The only cover data are U and the finite list coordinate F, so carrier admission and cover exposure are read from displayed finite rows.

The radius witness is read next. Rows R and D provide the positive rational or dyadic lower-bound data and the finite ledger saying that each point read through K, M reaches a cover member with that bound.

The structural rows preserve the same finite route. Transport through H and replay through C move the cover, list, radius, and ledger rows componentwise, while P and N source and name the packet without adding open-cover compactness or a choice-based selector. □

Theorem 7476.3 (Lebesgue covering lemma finite radius bound)**.** *Every finite-cover consumer of an accepted* $\mathsf{LebesgueCoveringLemma}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow} \longrightarrow U, F \longrightarrow R, D \longrightarrow \mathsf{FiniteLebesgueNumber}^{\uparrow}.$$

In particular, compact-uniform-continuity consumers may read the shared positive radius only after the finite cover row, finite list row, rational or dyadic radius row, and bound ledger have all been exposed. The route does not provide arbitrary open-cover compactness, a host-selected subcover, countable choice, quotient metric equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with a consumer read of $L = (K, M, U, F, R, D, H, C, P, N)$. The compact metric and metric-space rows K, M supply only the point and distance data needed to inspect the displayed finite cover.

Next read the finite cover route. The row U exposes the cover members, F records the finite list actually used by the packet, and R, D give the shared positive-radius bound through rational and dyadic comparisons.

Finally hand the same rows to $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ or to a compact-uniform-continuity consumer. Since P names no arbitrary open cover, host selector, countable-choice schedule, quotient metric equality, or ambient completeness row, every accepted consumer remains inside the finite radius route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueCoveringLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueCoveringLemma}^{\uparrow}$ packet is generated from compact metric source rows, metric distance rows, finite cover and finite list rows, rational and dyadic positive-radius rows, radius-bound ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7476.1, Definition 7476.2, and Definition 7476.3 seed the finite compact-cover packet over displayed compact metric, metric-space, rational-radius, dyadic-radius, finite-list, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-based subcover selection, countable choice, quotient metric equality, ambient compactness for all metric spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover exposure, rational and dyadic radius-bound exactness, finite-list handoff, transport replay, provenance, local naming, and non-escape for BEDC.Derived.LebesgueCoveringLemmaUp

Chapter 7477

A Concrete Naming Certificate for $\mathsf{SchurTest}^{\uparrow}$

Following Schur's operator-bound criterion in classical functional analysis, the $\mathsf{SchurTest}^{\uparrow}$ packet records the finite row-bound and column-bound data used to name a bounded matrix operator without importing an infinite measure kernel or an ambient completeness theorem. It sits between the finite matrix surface of chapter 49, the norm readback of chapter 89, the operator seal of chapter 1644, the Hilbert source of chapter 91, and the real tolerance interface of chapter 38.

7477.1 Carrier and finite Schur packet

Definition 7477.1 (Schur test carrier)**.** *A* $\mathsf{SchurTest}^{\uparrow}$ *carrier is a finite packet* $(A, U, V, M, N, B, H, R, T, E, C, P, K$ *with the following displayed rows:*

1. *A, a finite* $\mathsf{Mat}^{\uparrow}$ *window containing the matrix coefficients read by the test;*
2. *U, the row-bound ledger for the displayed rows of A;*
3. *V, the column-bound ledger for the displayed columns of A;*
4. *M, the finite* $\mathsf{Mat}^{\uparrow}$ *support window tying U and V to the same coefficient packet;*
5. *N, a* $\mathsf{Norm}^{\uparrow}$ *readback row for the source and target vector windows;*
6. *B, a* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *seal reached only after the matrix window, norm readback, and two Schur ledgers have been displayed;*
7. *H, the* $\mathsf{Hilbert}^{\uparrow}$ *source row for the finite source and target vector windows;*
8. *R, the* $\mathsf{Real}^{\uparrow}$ *tolerance row carrying the displayed bound budget;*
9. *T, the componentwise* hsame *rows for the matrix window, row ledger, column ledger, norm readback, Hilbert source, real tolerance, and operator seal;*
10. *E, the displayed* Cont *replay rows preserving the finite Schur packet under consumer reads;*
11. *C, the* Pkg *provenance over* $\mathsf{SchurTest}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *P, a refusal ledger excluding infinite measure kernels, quotient operators, selected limits, choice, spectral-theorem data, and ambient Hilbert completeness;*

13. *K, the local* $\mathsf{NameCert}_{\mathsf{SchurTest}^\uparrow}$ *row binding these displayed rows as one Schur-test certificate.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an infinite kernel, a chosen summability representative, a quotient operator equality, a spectral decomposition, or a completeness proof for the ambient Hilbert space.

7477.2 NameCert obligations

Theorem 7477.2 (Schur test NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SchurTest}^\uparrow}$ *surface for Definition 7477.1 consists exactly of carrier admission for the displayed packet* $(A, U, V, M, N, B, H, R, T, E, C, P, K)$*, row-bound exactness for U, column-bound exactness for V, finite matrix-window coherence for A and M, norm readback through N, operator sealing through B, Hilbert-source visibility through H, real-tolerance visibility through R, componentwise transport through T, continuation replay through E, package provenance through C, and refusal of the excluded analytic and choice-theoretic rows through P.*

Proof. Project the carrier of Definition 7477.1. The projection exposes the matrix window, the two Schur ledgers, the finite support window, the norm readback, the operator seal, the Hilbert source, the real tolerance, transport, replay, provenance, refusal, and local naming rows.

The row-bound and column-bound obligations are read separately from U and V. The finite matrix-window row M is the common support that prevents either ledger from being reread as an infinite summability statement or as an unbounded host kernel.

The operator-facing obligations then pass through N, B, H, R. These rows expose only the norm readback, bounded-operator seal, Hilbert source window, and real tolerance budget displayed by the packet. The transport, replay, provenance, refusal, and local naming rows keep the same finite support attached to every consumer read. □

Theorem 7477.3 (Schur test finite row-column bound)**.** *For every accepted* $\mathsf{SchurTest}^\uparrow$ *carrier* $(A, U, V, M, N, B, H, R, T, E, C, P, K)$*, any bounded-operator read of B must first preserve the finite row-bound ledger U, the finite column-bound ledger V, their shared* $\mathsf{Mat}^\uparrow$ *support window M, the* $\mathsf{Norm}^\uparrow$ *readback N, the* $\mathsf{Hilbert}^\uparrow$ *source row H, and the* $\mathsf{Real}^\uparrow$ *tolerance row R. The resulting bound is a finite BEDC-visible row-column budget, not an infinite measure Schur kernel, a choice-selected representative, a quotient operator, a spectral theorem, or an ambient Hilbert-completeness claim.*

Proof. Start from the accepted carrier and isolate the bounded-operator seal B. By Definition 7477.2, this seal is downstream from the matrix window, both row and column ledgers, the norm readback, the Hilbert source, and the real tolerance row.

The row and column ledgers are tied to the same finite support by M. Thus a consumer cannot keep only the operator seal while replacing the displayed estimates by an infinite kernel, a quotient equality of operators, or a selected representative of the matrix coefficients.

Finally replay the transport and provenance rows T, E, C, K. They preserve the same finite packet for every consumer read, while the refusal ledger P records the excluded analytic principles. Hence the exported bound is exactly the finite row-column budget stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurTest}^\uparrow}$**.**

Constructive story: A $\mathsf{SchurTest}^\uparrow$ packet is generated from finite MatrixUp coefficient windows, row-bound and column-bound ledgers, NormUp readback, a BoundedLinearOperatorUp seal, a HilbertUp source row, a RealUp tolerance row, componentwise hsame transport, Cont replay, Pkg provenance, refusal rows, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurTest}^\uparrow}$, seed)

Scope closed: Definition 7477.1, Definition 7477.2, and Definition 7477.3 seed the Schur row-bound, column-bound, finite MatrixUp window, NormUp readback, BoundedLinearOperatorUp seal, HilbertUp source, RealUp tolerance, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurTest}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim infinite measure Schur kernels, choice, quotient operators, spectral theorem data, or ambient Hilbert completeness.

Upgrade path: To reach obligationClosure, the finite row-column packet must close classifier transport and operator-read exactness against a checked formal target.

Chapter 7478

A Concrete Naming Certificate for $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$

Following Bishop and Bridges, a located Dedekind cut and a regular Cauchy real name are compared constructively by explicit rational brackets and moduli, not by identifying $\mathsf{Real}^{\uparrow}$ with a quotient of cuts. The $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$ packet records this comparison as finite BEDC data between the located-cut surface of chapter 404, the located Dedekind comparison boundary of chapter 6423, the dyadic arithmetic rows of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names the bridge surface used by Real and Completion consumers; it does not construct $\mathsf{Real}^{\uparrow}$ from Dedekind cuts and does not claim ambient real completeness.

Definition 7478.1 (Located Dedekind regular-Cauchy bridge carrier)**.** *A* $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *data: lower and upper rational rows, inhabitedness witnesses, located separation tests, and finite cut-stability checks.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger. It records dyadic lower and upper brackets extracted from L at displayed unary precisions, together with the tolerance rows used to compare successive brackets.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that lists exactly which located-cut bracket requests are read before a regular Cauchy row is exposed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy modulus and readback row obtained from the bracket ledger and the stream schedule.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is reached only after the located cut data, dyadic brackets, stream schedule, and regular Cauchy readback have been displayed.*
6. *H records componentwise* hsame *transport for located cut data, bracket ledgers, stream schedules, regular Cauchy rows, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite cut request to its bracket extraction, from brackets to the regular Cauchy modulus, and from the regular row to the terminal Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$, $\mathsf{LocatedDedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, D, S, R, E, H, C, P, N$. It has no coordinate for quotient cut equality, a selected Dedekind representative, trichotomy of reals, host limit selection, countable choice, a construction of $\mathsf{Real}^{\uparrow}$ *from cuts, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7478.2 (Located Dedekind regular-Cauchy bridge NameCert obligations)**.** *For an accepted* $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$ *carrier $B = (L, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}}$ *obligations are exactly carrier admission, bracket extraction, regular Cauchy realization, Real seal factorization, classifier stability, and nonescape. They are discharged by the displayed route*

$$L \longmapsto D \longmapsto S \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 7478.1. The located cut, dyadic bracket ledger, stream schedule, regular Cauchy readback, Real seal, transport, replay, provenance, and local naming rows are the complete public inventory.

Bracket extraction reads L only through the displayed located-cut tests and then records the resulting rational bounds in the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D. The row S lists the finite bracket requests whose answers are used downstream.

Regular Cauchy realization is the handoff from D and S to R. The modulus is visible as part of the $\mathsf{RegSeqRat}^{\uparrow}$ row, so no host limit selector or quotient-cut equality is needed to form the regular readback.

The Real seal factorization reads E only after L, D, S, R have been displayed. The transport and replay rows H and C keep classified packets componentwise aligned, while P and N source and name the same local bridge.

Nonescape follows by packet exhaustion. Since no coordinate stores trichotomy of reals, countable choice, a selected Dedekind representative, a Dedekind construction of $\mathsf{Real}^{\uparrow}$, or ambient completeness, no consumer can extract such data from the bridge. □

Falsifiable prediction: Every $\mathsf{Real}^{\uparrow}$ read from a $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$ packet factors through located cut data, dyadic bracket extraction, a StreamName schedule, RegSeqRat regular Cauchy readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: locatedcut, dyadicratcore, streamname, regseqrat, real: LocatedDedekindRegularCauchyB is not bijective to any listed sibling because it jointly carries located cut data, dyadic brackets, finite schedules, regular Cauchy modulus rows, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}$ packet is generated from located Dedekind cut data, dyadic bracket rows, a StreamName schedule, a RegSeqRat regular Cauchy readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}}$, seed)

Scope closed: Definition 7478.1 and Definition 7478.2 seed the finite located-cut to regular Cauchy comparison bridge for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedDedekindRegularCauchyBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind construction of $\mathrm{Real}^{\uparrow}$, quotient cut equality, selected representatives, trichotomy, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket extraction, regular Cauchy realization, Real seal factorization, classifier stability, and nonescape for BEDC.Derived.LocatedDedekindRegularCauchyBridgeUp.

Chapter 7479

A Concrete Naming Certificate for $\mathsf{FiniteIntervalPartition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteIntervalPartition}^{\uparrow}$ packet records the finite closed-interval subdivision data used by interval arguments before any regular rational, real-name, or compact-cover consumer can read a partition. It sits beside the tagged-cell interface of chapter 7449 and the dyadic refinement route of chapter 1877. The carrier gives $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ interval-cover arguments a displayed finite substrate, not a host finite set of cuts, quotient equality of partitions, a selected limiting mesh, open-cover compactness, or ambient real completeness.

Definition 7479.1 (Finite interval partition carrier)**.** *A* $\mathsf{FiniteIntervalPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, U, K, M, V, R, H, C, P, N)$$

with the following displayed rows:

1. *L, U are the closed interval endpoint rows, sourced through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *witnesses and read as the boundary of the inspected interval.*
2. *K is the ordered cut row. It lists the finite internal cuts between L and U together with adjacency witnesses for consecutive cells.*
3. *M is the dyadic mesh-bound row. It records one displayed dyadic scale bounding the width of every cell determined by L, K, U.*
4. *V is the cell-coverage row. It witnesses that every interval containment read is routed through exactly one displayed cell or through the shared boundary ledger attached to neighboring cells.*
5. *R is the refinement ledger. It records how a later partition read refines the displayed cut row while preserving endpoint rows, coverage rows, and the mesh-bound comparison.*
6. *H records componentwise* hsame *transport for endpoint rows, ordered cut rows, mesh bounds, coverage rows, refinement ledgers, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from endpoints to cuts, from cuts to cells, from cells to coverage, and from coverage to refinement-facing consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalPartition}^{\uparrow}$*,* $\mathsf{TaggedPartition}^{\uparrow}$*,* $\mathsf{DyadicCoverRefinement}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, M, V, R, H, C, P, N$. It has no coordinate for a host list of cuts, quotient equality of partitions, an unbounded refinement family, selected mesh limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7479.2 (Finite interval partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalPartition}^{\uparrow}$ *carrier $F = (L, U, K, M, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalPartition}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite endpoint, cut, mesh, coverage, refinement, transport, replay, provenance, and naming rows;*
2. *endpoint order, requiring L and U to bracket the ordered cut row K and every adjacent cell boundary;*
3. *cell coverage, requiring every interval containment read to pass through the displayed coverage row V before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff;*
4. *mesh and refinement compatibility, requiring M and R to preserve cell width bounds and endpoint coverage under displayed refinement;*
5. *classifier transport, requiring H, C, P, N to preserve the same finite partition packet under* hsame, Cont, *provenance, and local naming.*

The obligations expose a finite interval-subdivision substrate that may be consumed by $\mathsf{TaggedPartition}^{\uparrow}$ *tags,* $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *covers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *interval seals without importing host finite partitions, quotient partition equality, selected limiting meshes, open-cover compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 7479.1. Carrier admission reads exactly the rows $L, U, K, M, V, R, H, C, P, N$ so the local certificate can inspect only the displayed endpoint, cut, mesh, coverage, refinement, transport, replay, provenance, and naming data.

Endpoint order is read next. The boundary rows L, U bracket the ordered cut row K, and every cell boundary is part of the same displayed adjacency ledger rather than a host-side partition object.

The coverage and mesh clauses then consume V, M, R. The coverage row sends every interval containment request through the displayed cells, while the dyadic mesh row and refinement ledger keep later subdivision reads compatible with the same endpoint and coverage data.

Finally replay H, C, P, N. These rows transport, replay, source, and name the same finite packet, so no extra coordinate can supply quotient partition equality, an external refinement family, a selected mesh limit, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The five listed rows therefore exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntervalPartition}^{\uparrow}$ packet is generated from closed interval endpoint rows, ordered finite cut rows, a dyadic mesh bound, cell coverage rows, refinement ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteIntervalPartition}^{\uparrow}}$, seed)

Scope closed: Definition 7479.1 and Definition 7479.2 seed the finite interval-subdivision substrate used by RegSeqRat and Real interval-cover consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteIntervalPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient partition equality, host finite-partition presentation, selected limiting meshes, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint order, cell coverage, mesh-refinement compatibility, classifier transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalPartitionUp.

Chapter 7480

A Concrete Naming Certificate for DyadicIntervalSubdivision$^{\uparrow}$

Following Bishop and Bridges, the DyadicIntervalSubdivision$^{\uparrow}$ packet records the finite dyadic cell subdivision used by compact interval arguments before any finite-cover or uniform-modulus consumer can read a mesh. It refines the interval-net discipline of chapter 1250, the cover-refinement carrier of chapter 1877, the compact metric source of chapter 129, and the tolerance-only handoff of chapter 369. The packet is a concrete dyadic subdivision surface for compact-uniform-continuity arguments, not an open-cover compactness theorem, a selected subcover, quotient real equality, or ambient Real$^{\uparrow}$ completeness.

Definition 7480.1 (Dyadic interval subdivision carrier)**.** *A* DyadicIntervalSubdivision$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, E, Q, G, V, W, M, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the* ClosedBoundedIntervalNet$^{\uparrow}$ *source row exposing the closed interval endpoints and the finite interval-net boundary.*
2. *E is the endpoint-exactness row, tying every dyadic cell endpoint back to the same lower and upper interval endpoints.*
3. *Q is the dyadic scale row, sourced through* DyadicRatCore$^{\uparrow}$*, at which the interval is subdivided.*
4. *G is the ordered dyadic grid row listing the cell endpoints at scale Q.*
5. *V is the cell-coverage row witnessing that every interval read is routed through one displayed dyadic cell or a shared endpoint boundary.*
6. *W is the common-refinement row comparing the displayed grid with a later* DyadicCoverRefinement$^{\uparrow}$ *cover row.*
7. *M is the mesh-bound row recording the dyadic diameter bound exported to compact finite-cover consumers.*
8. *U is the* UniformModulus$^{\uparrow}$ *handoff row, available only after the grid, coverage, common-refinement, and mesh rows have been consumed.*
9. *H records componentwise* hsame *transport for interval-net source, endpoint exactness, dyadic scale, grid, coverage, common-refinement, mesh, uniform-modulus handoff, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from interval endpoints to dyadic grid rows, from grid rows to coverage rows, from coverage to common refinement, and from mesh bounds to the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalSubdivision}^{\uparrow}$, $\mathsf{DyadicCoverRefinement}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalSubdivision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, E, Q, G, V, W, M, U, H, C, P, N$. It has no coordinate for an open-cover subcover selector, quotient real equality, a hidden grid chooser, arbitrary compactness, or ambient metric completeness.

Theorem 7480.2 (Dyadic interval subdivision NameCert obligations)**.** *For every accepted* DyadicIntervalSubdivisio *carrier $D = (I, E, Q, G, V, W, M, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalSubdivision}^{\uparrow}}$ *surface consists exactly of carrier admission, interval-net source exposure, endpoint exactness, dyadic scale classification, ordered grid coverage, cell-coverage exactness, common-refinement compatibility with* $\mathsf{DyadicCoverRefinement}^{\uparrow}$*, mesh-bound export,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The finite route is*

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{DyadicIntervalSubdivision}^{\uparrow} \longmapsto \mathsf{DyadicCoverRefinement}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow},$$

with the $\mathsf{CompactMetric}^{\uparrow}$ *source row read only through the displayed finite interval-net and mesh data.*

Proof. Carrier admission is projection from Definition 7480.1. The source interval row I, endpoint exactness row E, dyadic scale row Q, ordered grid row G, coverage row V, common-refinement row W, mesh row M, and handoff row U are the only mathematical rows available.

Read I, E, Q, G, V first. These rows display the closed interval endpoints, fix the dyadic scale, list the finite grid, and route every interval containment request through a displayed cell or shared boundary.

Next consume W, M, U. The common-refinement row compares the dyadic grid with the $\mathsf{DyadicCoverRefinement}^{\uparrow}$ cover route, the mesh row exports the dyadic diameter bound to compact finite-cover consumers, and the $\mathsf{UniformModulus}^{\uparrow}$ row is visible only after those finite checks.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same subdivision packet, so no open-cover selector, hidden grid choice, quotient real equality, arbitrary compactness principle, or ambient completeness row can enter the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalSubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalSubdivision}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalNet source, endpoint exactness, dyadic scale rows, ordered dyadic grid rows, cell coverage, common-refinement rows, mesh bounds, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicIntervalSubdivision}^{\uparrow}}$, seed)

Scope closed: Definition 7480.1 and Definition 7480.2 seed the finite dyadic subdivision route used by compact finite-cover and uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalSubdivision}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, hidden grid choice, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval-net source exposure, endpoint exactness, dyadic scale classification, grid coverage, common-refinement compatibility, mesh export, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalSubdivisionUp.

Chapter 7481

A Concrete Naming Certificate for FiniteOpenCoverNerve$^{\uparrow}$

Following Alexandroff and the standard nerve construction for finite covers, the FiniteOpenCoverNerve$^{\uparrow}$ packet records the finite simplicial incidence data carried by a displayed open cover without importing open-cover compactness, a selected refinement theorem, or ambient choice. It sits beside the compact source of chapter 60, the compact metric finite-net source of chapter 129, the finite lower-bound fold of chapter 619, and the finite cover-fold route of chapter 7452. The packet is a named nerve carrier over displayed cover rows, nonempty finite-intersection rows, incidence rows, ProbeBundle compatibility, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Definition 7481.1 (Finite open-cover nerve carrier)**.** *A* FiniteOpenCoverNerve$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (K, O, I, J, B, H, C, P, M)$$

with the following displayed rows:

1. *K is the compact source row, read through* Compact$^{\uparrow}$ *or* CompactMetric$^{\uparrow}$*, that bounds the cover inventory as finite data.*
2. *$O = (O_1, \ldots, O_n)$ is the displayed finite open-cover row.*
3. *I is the finite nonempty-intersection ledger, listing exactly those finite subfamilies of O whose displayed intersection witness is present.*
4. *J is the incidence row, recording vertex-to-cover membership and simplex-to-intersection membership for the nerve.*
5. *B is the* ProbeBundle *compatibility row, aligning cover cells, finite probes, and lower-bound folds with the cover-fold surface of chapter 7452.*
6. *H records componentwise* hsame *transport for K, O, I, J, B, C, P, M.*
7. *C records displayed* Cont *replay from compact source rows through cover membership, finite-intersection witnesses, nerve incidence, probe-bundle compatibility, provenance, and local naming.*
8. *P is the* Pkg *provenance over* FiniteOpenCoverNerve$^{\uparrow}$, Compact$^{\uparrow}$, CompactMetric$^{\uparrow}$, ProbeBundleCoverFold$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. M is the local $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNerve}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, a selected subcover theorem, a host simplicial realization, quotient point equality, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.FiniteOpenCoverNerveUp.FiniteOpenCoverNerveTasteGate single carrier alignment`

Theorem 7481.2 (Finite open-cover nerve NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$ *carrier* $N = (K, O, I, J, B, H, C, P, M)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNerve}^{\uparrow}}$ *surface consists exactly of compact-source exposure, finite open-cover exposure, nonempty finite-intersection ledger exposure, nerve incidence exposure,* ProbeBundle *compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is discharged by host open-cover compactness, a selected external refinement, quotient point equality, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 7481.1. The rows K, O, I, J, B are the whole mathematical surface: the compact source, the finite cover, the displayed nonempty intersections, the incidence relation, and the probe-bundle compatibility ledger.

The nerve data are finite because O is displayed as a finite row and I lists only the finite subfamilies whose intersection witness is present. The incidence row J reads from those two rows, so it cannot introduce a host realization or an unlisted refinement.

The structural rows H, C, P, M transport, replay, source, and name the same packet. Since the carrier has no coordinate for the excluded compactness, choice, quotient, completeness, Lean, or bridge data, the listed rows exhaust the local certificate. □

Theorem 7481.3 (Finite open-cover nerve compact-uniform route)**.** *Every compact-uniform consumer that reads an accepted* $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$ *packet must factor through the displayed route*

$$\mathsf{Compact}^{\uparrow}/\mathsf{CompactMetric}^{\uparrow} \longmapsto O \longmapsto I \longmapsto J \longmapsto \mathsf{ProbeBundle} \longmapsto \mathsf{FiniteNetMinimumFold}^{\uparrow}.$$

The nerve row supplies finite cover incidence for the compact-uniform modulus route, including the cover-fold consumer of chapter 7452, but it supplies no open-cover compactness theorem, selected subcover, host geometric realization, quotient point equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 7481.2. A consumer first sees the compact source row K and the finite cover row O, then the nonempty finite-intersection ledger I and incidence row J.

The probe compatibility row B is read after those cover and incidence rows. It aligns the nerve inventory with the same finite probes and lower-bound folds used by the cover-fold route, so a compact-uniform modulus consumer receives finite incidence and fold data rather than a background compactness theorem.

The transport, replay, provenance, and local naming rows preserve this route. No additional row can provide an external subcover selector, host realization, quotient equality, choice principle, or ambient real-completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNerve}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$ packet is generated from a compact source row, a finite open-cover row, nonempty finite-intersection rows, incidence rows, ProbeBundle compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7481.1, Definition 7481.2, and Definition 7481.3 seed the finite cover-nerve carrier used by compact-uniform finite-cover consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNerve}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host simplicial realizations, quotient point equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover exposure, intersection-ledger exactness, incidence exposure, ProbeBundle compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOpenCoverNerve

Chapter 7482

A Concrete Naming Certificate for $\mathsf{MinimizingMovement}^{\uparrow}$

Following De Giorgi, Ambrosio, Gigli, and Savare on minimizing movements, the $\mathsf{MinimizingMovement}^{\uparrow}$ packet records the finite descent surface by which an implicit time-step scheme is read before any gradient-flow or curve-of-maximal-slope consumer receives it. It sits after the variational descent ledger of chapter 3926, the complete-metric carrier of chapter 128, the metric source of chapter 57, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the lower-semicontinuity packet of chapter 1536. The packet names a displayed minimizing-step route only; it does not select minimizers, identify quotient metric points, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7482.1 (Minimizing-movement carrier)**.** *A* $\mathsf{MinimizingMovement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, K, L, Q, R, E, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row for the inspected state addresses.*
2. *d is the metric-comparison row naming the distances used by the step penalty.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *descent-readiness row for the finite Cauchy tails requested by the step scheme.*
4. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *value-stability row for the energy inspected at each step.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for comparing displayed energy values and penalties.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for exported values after the regular-rational readback is visible.*
7. *E is the* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *descent ledger used to certify the approximate minimizing step.*
8. *S is the step-size and budget row, naming the finite time-step schedule, penalty coefficient, and admissible error budget.*
9. *T is the finite step trace, recording the source state, trial state, descent comparison, and accepted next state for each inspected step.*
10. *H records componentwise* hsame *transport for metric, complete-metric, lower-semicontinuity, readback, variational, schedule, and step-trace rows.*

11. *C records displayed* Cont *replay from a minimizing-movement request through $X, d, K, L, Q, R, E, S, T$.*
12. *P is the* Pkg *provenance over* $\mathsf{MinimizingMovement}^{\uparrow}$, $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimizingMovement}^{\uparrow}}$ *row.*

The carrier exports exactly these finite rows. It contains no selected minimizer, quotient equality of states, countable-choice scheduler, completed gradient flow, metric slope theorem, or independent proof that the ambient metric space is complete.

Theorem 7482.2 (Minimizing-movement NameCert obligations)**.** *For every accepted* $\mathsf{MinimizingMovement}^{\uparrow}$ *carrier $M = (X, d, K, L, Q, R, E, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinimizingMovement}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed metric, distance, complete-metric, lower-semicontinuity, regular-rational, real-seal, variational, schedule, trace, transport, continuation, provenance, and naming rows;*
2. *metric-penalty readability, requiring every penalty comparison in T to use the displayed d row on addresses admitted by X;*
3. *complete-metric readiness, requiring every Cauchy-tail read requested by the step route to pass through K;*
4. *lower-semicontinuity and value readback, requiring every energy comparison to pass through L, Q, R before any exported value is read;*
5. *variational-step exposure, requiring the approximate minimizing-step ledger to be read through E before the accepted next-state row of T is consumed;*
6. *schedule and budget coverage, requiring the time-step, penalty, and error data to be named in S before the corresponding trace row is used;*
7. *transport, replay, provenance, and local naming stability through H, C, P, N;*
8. *ledger non-escape, excluding selected minimizers, quotient state equality, countable-choice schedules, completed gradient flows, metric-slope conclusions, and ambient real or metric completeness claims.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier row by row. The metric and distance rows X, d expose exactly the inspected states and penalty comparisons, while K, L, Q, R expose the complete-metric readiness, lower-semicontinuity reads, regular-rational comparisons, and real seals used by the step route.

The variational and schedule rows determine when the trace may be read. The row E supplies the displayed Ekeland-style descent ledger for an approximate step, and S supplies the finite time-step, penalty, and admissible error data before any coordinate of T is accepted as a next-state record.

The trace row T has no hidden coordinates beyond the source state, trial state, comparison, and accepted next-state data displayed in the carrier. Thus each minimizing-step read is exhausted by the metric, value, variational, and schedule rows already listed.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores a selected minimizer, quotient state equality, countable-choice schedule, completed gradient flow, metric slope theorem, or ambient completeness assertion, the listed obligations exhaust the local NameCert surface. □

Theorem 7482.3 (Minimizing-movement real-completion route)**.** *Every Real-facing or Completion-facing consumer reading an accepted* $\mathsf{MinimizingMovement}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{EkelandVariationalPrinciple}^{\uparrow} \longmapsto S, T, H, C, P, N.$$

The consumer therefore receives a finite implicit-step descent packet over displayed metric, value, and variational rows, not a selected minimizer, quotient completion object, countable-choice sequence, completed gradient flow, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier $M = (X, d, K, L, Q, R, E, S, T, H, C, P, N)$. A consumer cannot read a distance penalty without first reading X, d, and cannot read a completion-facing tail request without first reading the complete-metric row K.

The value route is similarly forced. Energy comparisons pass through the lower-semicontinuity row L, the regular-rational readback Q, and the real seal R before they can be exported to a Real-facing consumer.

The minimizing-step data then pass through E, S, T. The variational ledger E supplies the displayed approximate descent route, S names the finite time-step and budget data, and T records only the inspected source, trial, comparison, and accepted next-state rows.

The structural rows H, C, P, N preserve and replay this same route. Because the carrier has no coordinate for selected minimizers, quotient completion, countable-choice schedules, completed flows, metric slopes, or independent completeness, no such datum can be obtained by a downstream Real or Completion consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{MinimizingMovement}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimizingMovement}^{\uparrow}$ packet is generated from a finite metric source, displayed distance penalties, complete-metric descent readiness, lower-semicontinuity value reads, regular-rational and real readback, an Ekeland-style variational ledger, a step-size and budget row, a finite step trace, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimizingMovement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7482.1, Definition 7482.2, and Definition 7482.3 seed the implicit-step descent route consumed by later Real, Completion, gradient-flow, and curve-of-maximal-slope packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimizingMovement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected minimizers, quotient state equality, countable-choice schedules, completed gradient flows, metric-slope theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric readiness, lower-semicontinuity readback, variational-step exposure, schedule coverage, trace nonescape, and consumer route exhaustion for BEDC.Derived.MinimizingMoveme

Chapter 7483

A Concrete Naming Certificate for $\mathsf{RegularSequenceModulus}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy construction needs an explicit tail-modulus packet before a regular rational readback can be handed to a real-name seal. The $\mathsf{RegularSequenceModulus}^{\uparrow}$ packet records that finite tail data over the dyadic arithmetic of chapter 291, the stream-window naming surface of chapter 126, the regular rational route of chapter 301, the terminal real-name surface of chapter 38, and the modulus boundary of chapter 292. It is a displayed modulus object for regular-Cauchy tails, not a quotient of streams, a selected limit, a host real equality, a countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7483.1 (Regular sequence modulus carrier)**.** *A* $\mathsf{RegularSequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, D, Q, M, T, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, listing the finite windows of the sequence name inspected by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the dyadic comparison requests used by the tail test.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations that will be consumed downstream.*
4. *M is the tail-modulus row; for each admitted precision request it displays a finite bound after which all listed readbacks are stable inside the corresponding dyadic tolerance.*
5. *T is the finite tail ledger pairing stream windows, dyadic tolerances, readback rows, and modulus bounds.*
6. *H records componentwise* hsame *transport for S, D, Q, M, T and the structural rows.*
7. *C records displayed* Cont *replay from a tail request through the same finite rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through hsame-stable agreement of $S, D, Q, M, T, H, C, P, N$. Two carriers agree when the same finite windows, dyadic tolerance rows, regular rational readbacks, modulus bounds, tail-ledger entries, transport rows, replay rows, provenance rows, and local naming rows are matched by componentwise hsame*. No coordinate records a selected limit, quotient stream equality, countable choice, host real equality, or ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.RegularSequenceModulusUp.TasteGate.RegularSequenceModulusTasteGate single carrier alignment`

Theorem 7483.2 (Regular sequence modulus carrier admission)**.** *Every accepted* RegularSequenceModulus$^{\uparrow}$ *carrier $R = (S, D, Q, M, T, H, C, P, N)$ admits exactly the finite NameCert surface displayed in Definition 7483.1. A consumer may project stream windows, dyadic tolerances, regular rational readbacks, tail-modulus bounds, tail-ledger entries, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It may not project a selected limit, quotient stream equality, countable-choice schedule, host real equality, or ambient* Real$^{\uparrow}$ *completeness witness.*

Proof. Project the carrier definition. The rows S, D, Q, M, T provide the finite data used by tail reads, while H, C, P, N provide the structural transport, replay, provenance, and local name.

The classifier is hsame-stable only on those displayed rows. Therefore an accepted packet can be read through the listed finite coordinates and through no detached completion coordinate.

Finally inspect the carrier inventory. Since selected limits, quotient stream classes, countable-choice schedules, host real equalities, and ambient completeness witnesses are not fields of R, they cannot be projected from the NameCert surface. □

Theorem 7483.3 (Regular sequence modulus tail exactness)**.** *Let $R = (S, D, Q, M, T, H, C, P, N)$ be an accepted* RegularSequenceModulus$^{\uparrow}$ *carrier. For every admitted dyadic precision row in D, the modulus row M supplies a finite tail bound, and the ledger T records that all corresponding* RegSeqRat$^{\uparrow}$ *readbacks in Q past that bound agree inside the displayed dyadic tolerance. This exactness is internal to the finite packet: it does not choose a limit and does not identify streams by quotient equality.*

Proof. Begin with the dyadic precision row in D. The carrier requires the same precision request to be paired with a displayed bound in M and with readback entries in Q.

Use the tail ledger T. It is the finite row that binds the stream windows from S, the dyadic comparisons from D, the regular rational observations from Q, and the modulus bound from M into one tail assertion.

The structural rows H, C, P, N only transport, replay, source, and name that same finite assertion. Because no limit or quotient coordinate is present, the exactness statement remains the displayed tail-bound claim carried by T. □

Theorem 7483.4 (Regular sequence modulus RegSeqRat handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *consumer that receives a tail witness from an accepted* RegularSequenceModulus$^{\uparrow}$ *packet must pass through the finite route*

StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegularSequenceModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Co

For each admitted unary precision row, the modulus bound in M supplies the closeness witness consumed by the RegSeqRat$^{\uparrow}$ *readback boundary, and the terminal* Real$^{\uparrow}$ *seal may only read that displayed witness. The handoff exports no selected limit, quotient stream equality, countable choice, host real equality, or ambient completeness principle.*

Proof. Apply Definition 7483.2. A downstream read can only project the displayed carrier rows and the structural rows.

Next use Definition 7483.3. The precision row in D and the bound in M are tied by T to the regular rational readback in Q, so the $\mathsf{RegSeqRat}^{\uparrow}$ boundary receives a finite closeness witness rather than an external convergence theorem.

The $\mathsf{Real}^{\uparrow}$ seal is downstream of the same readback route. Since H, C, P, N merely preserve and name the finite packet, the terminal seal cannot add quotient stream equality, a selected limit, choice, host equality, or ambient completeness. □

The regular tail-modulus row is consumed by the Bishop Cauchy calibration route of chapter 7577. The dyadic tolerance, stream window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal are the shared rows, so the modulus packet is a finite source for downstream calibration rather than an isolated kernel float.

Falsifiable prediction: Every accepted $\mathsf{RegularSequenceModulus}^{\uparrow}$ consumer displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, finite tail-ledger, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before it reads a terminal $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchymodulus: $\mathsf{RegularSequenceModulus}^{\uparrow}$ is not bijective to any listed sibling because it packages the finite tail-modulus row and the regular rational handoff ledger as one hsame-stable NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7483.1, Definition 7483.2, Definition 7483.3, and Definition 7483.4 seed the finite tail-modulus packet for regular rational readbacks and terminal real-name reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, tail-bound exactness, RegSeqRat handoff, transport replay, provenance, and non-escape over the same finite packet.

Constructive story: A $\mathsf{RegularSequenceModulus}^{\uparrow}$ packet is generated from finite stream windows, dyadic tolerance rows, regular rational readbacks, tail-modulus bounds, a finite tail ledger, componentwise transport, displayed replay, package provenance, and local naming evidence.

Chapter 7484

A Concrete Naming Certificate for $\mathsf{VisionCoverageLedger}^{\uparrow}$

The $\mathsf{VisionCoverageLedger}^{\uparrow}$ packet realizes the top-level routing surface of chapter 9236 as finite BEDC data. It records which vision label is being covered, which concrete-instance chapter receives the coverage row, which marker was replayed, and which NameCert rows make the routing auditable. The packet sits beside the existing coverage account of chapter 7381 and the concrete tail-modulus surface of chapter 7483; it is a finite ledger of label-to-chapter coverage, not a bijection between visions and constructions and not a global completeness theorem for the visions tree.

Definition 7484.1 (Vision coverage ledger carrier)**.** *A* $\mathsf{VisionCoverageLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (L, T, M, W, H, C, P, N)$$

with the following rows:

1. *L is the source-label row, naming an existing vision label such as the capstone label of chapter 9236.*
2. *T is the concrete-target row, naming an existing concrete-instance chapter label.*
3. *M is the marker row, recording the displayed concretized-in route that connects L to T.*
4. *W is the well-formedness row, checking that both labels are present in the manuscript and that the target label belongs to a concrete-instance NameCert chapter.*
5. *H records componentwise* hsame *transport for the source-label, target-label, marker, and well-formedness rows.*
6. *C records displayed* Cont *replay from a coverage query through L, T, M, W, H.*
7. *P is the* Pkg *provenance over* $\mathsf{VisionCoverageLedger}^{\uparrow}$, $\mathsf{VisionCoverageIndex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{VisionCoverageLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the finite rows L, T, M, W, H, C, P, N. It has no coordinate for a total vision-to-instance bijection, a completeness theorem for all vision files, a semantic equivalence between a vision and a construction, or automatic coverage for unmarked chapters.

Theorem 7484.2 (Vision coverage ledger marker admission)**.** *Every accepted* $\mathsf{VisionCoverageLedger}^{\uparrow}$ *carrier admits a coverage row only when its marker row M names an existing source label L and an existing concrete-instance target label T. The admitted row is exactly the finite marker route displayed in the carrier; it does not infer coverage from prose proximity, filename similarity, or thematic overlap.*

Proof. Project the carrier of Definition 7484.1. The marker row M is stored together with the source-label row L and target-label row T.

Use the well-formedness row W. It checks existence of both labels and checks that T is a concrete-instance NameCert chapter label before transport or replay is consumed.

The structural rows H, C, P, N preserve and name the same marker route. Since no prose-inference, filename, or thematic-overlap coordinate is present, marker admission is exactly the displayed finite row. □

Theorem 7484.3 (Vision coverage ledger label well-formedness)**.** *For every accepted* $\mathsf{VisionCoverageLedger}^{\uparrow}$ *carrier $V = (L, T, M, W, H, C, P, N)$, the well-formedness row W is consumed before any coverage replay. Thus a coverage read can only cite a source label already present in the visions surface and a target label already present in a concrete-instance chapter, including rows such as chapter 9236 and chapter 7381.*

Proof. Start with the carrier projection. The replay row C is downstream of L, T, M, W, H, so it cannot be consumed before the labels and marker are displayed.

Apply Definition 7484.2. Admission requires the marker route to name an existing source label and an existing concrete-instance target label.

Therefore the well-formedness row is the gate for coverage replay. It keeps the ledger finite and prevents a consumer from reading a missing vision label or a missing concrete target as covered. □

Theorem 7484.4 (Vision coverage ledger transport and replay stability)**.** *If two accepted* $\mathsf{VisionCoverageLedger}^{\uparrow}$ *carriers agree by componentwise* hsame *on L, T, M, W, H, C, P, N, then every coverage replay through one carrier replays through the other with the same source label, concrete target label, marker row, and well-formedness check. The transport does not create a new coverage target and does not turn a finite marker row into a global coverage theorem.*

Proof. Use the classifier in Definition 7484.1. Agreement is defined componentwise on the finite rows L, T, M, W, H, C, P, N.

The transport row H preserves the source label, target label, marker row, and well-formedness row. The replay row C is displayed over those same transported rows.

Finally apply Definition 7484.3. Since replay is gated by the same well-formedness row on both carriers, transport cannot add an unlisted coverage target or promote the row to global coverage. □

Theorem 7484.5 (Vision coverage ledger finite-row non-escape)**.** *Every admitted* $\mathsf{VisionCoverageLedger}^{\uparrow}$ *coverage read remains inside the finite rows L, T, M, W, H, C, P, N. It records coverage only for the source and target labels named in the row, and it exports no completeness claim for unmarked vision sections, no bijection between all visions and all concrete instances, and no semantic equality between a vision narrative and a NameCert construction.*

Proof. Project the carrier. The only data available to a coverage consumer are the source-label, target-label, marker, well-formedness, transport, replay, provenance, and local naming rows.

Use marker admission and label well-formedness from Definition 7484.2 and Definition 7484.3. They restrict the read to one existing source label and one existing concrete target label.

Use transport and replay stability from Definition 7484.4. Transport preserves the same finite row, so replay cannot escape to unmarked labels, a total bijection, or semantic equality. □

Falsifiable prediction: Every accepted $\mathsf{VisionCoverageLedger}^{\uparrow}$ coverage row names an existing vision label and an existing concrete-instance chapter label before it may be replayed as a coverage marker.

Independence witness: vision-coverage-index, namecert, package, gap, exact-globalize-base: $\mathsf{VisionCoverageLedger}^{\uparrow}$ is not bijective to any listed sibling because it stores source-label, concrete-target, marker-ledger, well-formedness, transport, replay, provenance, and NameCert rows for coverage accounting without claiming total coverage, semantic equivalence, or a global bijection.

Closure status for $\mathsf{NameCert}_{\mathsf{VisionCoverageLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VisionCoverageLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7484.1, Definition 7484.2, Definition 7484.3, Definition 7484.4, and Definition 7484.5 seed the finite label-to-concrete coverage ledger for the top-level visions surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VisionCoverageLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total vision coverage, vision-to-concrete bijection, semantic equality between narratives and constructions, Lean verification, bridge checking, or coverage for unmarked rows.

Upgrade path: Obligation closure requires checked marker admission, target-label well-formedness, hsame transport, replay stability, provenance, local naming, and finite-row non-escape.

Constructive story: A $\mathsf{VisionCoverageLedger}^{\uparrow}$ packet is generated from a source vision label, a concrete-instance target label, a coverage marker row, a label well-formedness row, componentwise transport, displayed replay, package provenance, and local naming evidence.

Chapter 7485

A Concrete Naming Certificate for $\mathsf{LHopitalRule}^{\uparrow}$

Following de l'Hopital, Bernoulli, and Cauchy's mean-value route, the $\mathsf{LHopitalRule}^{\uparrow}$ packet records the finite BEDC surface behind a derivative-ratio comparison before a limiting quotient is read as a real-facing conclusion. It sits after the derivative quotient interface of chapter 122, the Cauchy mean-value theorem handoff of chapter 6763, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names one displayed calculus route; it does not import a host limit rule, quotient real equality, trichotomy, selected limiting point, or ambient completeness.

Definition 7485.1 (LHopitalRule carrier)**.** *A* $\mathsf{LHopitalRule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, F, G, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row, including the endpoint-side window on which the quotient comparison is requested.*
2. *F and G are the two differentiable function rows whose values and derivative quotients are inspected through* $\mathsf{Derivative}^{\uparrow}$ *data.*
3. *D is the derivative-ratio comparison row, recording the displayed comparison between the derivative quotients of F and G on the same interval window.*
4. *Q is the* $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ *handoff row that turns the paired derivative data into a finite quotient comparison route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the quotient windows and derivative-ratio budgets.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the interval, function rows, derivative-ratio row, Cauchy mean-value handoff, and regular rational readback are displayed.*
7. *H stores componentwise* hsame *transport for the interval, functions, derivative-ratio comparison, Cauchy mean-value handoff, readback, real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from the interval and function rows to the derivative comparison, from the derivative comparison to the Cauchy mean-value handoff, and from that handoff to the readback and real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{LHopitalRule}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LHopitalRule}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, G, D, Q, R, E, H, C, P, N$. It has no coordinate for a host limit theorem, quotient real equality, trichotomy, selected endpoint representative, or ambient real completeness.

Theorem 7485.2 (LHopitalRule NameCert obligations)**.** *For an accepted* $\mathsf{LHopitalRule}^{\uparrow}$ *carrier $L = (I, F, G, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LHopitalRule}^{\uparrow}}$ *obligations are exactly interval admission, paired differentiable-function admission, derivative-ratio comparison, Cauchy mean-value handoff, regular rational readback, Real seal factorization, componentwise transport, replay, provenance, and local naming. These obligations are discharged only through the displayed route*

$$I, F, G \longmapsto D \longmapsto Q \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Project the accepted carrier. The interval row I fixes the side and finite window, while F and G are the only differentiable function rows available to the packet. Thus carrier admission and function admission are projections from Definition 7485.1.

Next inspect the derivative-ratio row. The row D reads derivative quotient data through the $\mathsf{Derivative}^{\uparrow}$ interface and records only a displayed comparison on the same interval window. No independent endpoint limit or host quotient rule is present.

Now apply the Cauchy mean-value handoff row Q. It is the only route from the paired derivative comparison to a quotient comparison, and the later rows R and E can read that comparison only through regular rational readback and the terminal real seal.

Finally read H, C, P, N. These rows transport, replay, source, and name the same finite packet, so no obligation can escape to quotient real equality, trichotomy, selected limiting representatives, host calculus, or ambient completeness. □

Theorem 7485.3 (LHopitalRule Cauchy mean-value handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{LHopitalRule}^{\uparrow}$ *packet must read the Cauchy mean-value handoff row Q after the interval row I, the differentiable function rows F, G, and the derivative-ratio comparison row D have been displayed. The exported route is*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{CauchyMeanValueTheorem}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the same finite packet, with $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets available only through the displayed readback rows.*

Proof. Begin with the obligation route of Definition 7485.2. It places I, F, G before D, and places D before the Cauchy mean-value row Q can be consumed.

Use the sibling Cauchy mean-value packet named in chapter 6763. The row Q is a handoff row, not a host theorem invocation: it records the finite mean-value comparison surface already exposed by the derivative rows.

Next route the readback. The row R gives the regular rational windows for the quotient comparison, with $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ data visible through the displayed finite observations and tolerance ledger. The row E is downstream of R and is the only real-facing seal.

Therefore any consumer reaching $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ has already read I, F, G, D, Q through the same packet. The handoff exports no selected limit, no quotient real equality, no trichotomy, no countable choice, and no ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{LHopitalRule}^{\uparrow}$ read factors through an interval row, two differentiable function rows, one derivative-ratio comparison row, a $\mathsf{CauchyMeanValueTheorem}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ budgets, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: derivative, cauchy-mean-value-theorem, streamname, dyadicratcore, regseqrat, real: $\mathsf{LHopitalRule}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval window, two differentiable function rows, derivative-ratio comparison, Cauchy mean-value handoff, readback, real seal, transport, replay, provenance, and local naming as one calculus certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LHopitalRule}^{\uparrow}}$.

Constructive story: A $\mathsf{LHopitalRule}^{\uparrow}$ packet is generated from an interval row, two differentiable function rows, a derivative-ratio comparison, a CauchyMeanValueTheorem handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LHopitalRule}^{\uparrow}}$, seed)

Scope closed: Definition 7485.1, Definition 7485.2, and Definition 7485.3 seed the finite derivative-ratio and Cauchy mean-value route for one displayed l'Hopital-rule packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LHopitalRule}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host limit rule, quotient real equality, selected limits, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, differentiable-function admission, derivative-ratio comparison, Cauchy mean-value handoff, regular rational readback, Real seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.LHopitalRuleUp.

Chapter 7486

A Concrete Naming Certificate for $\mathsf{VoronoiCellLedger}^{\uparrow}$

Following Aurenhammer and the de Berg–van Kreveld–Overmars–Schwarzkopf treatment of Voronoi diagrams, the $\mathsf{VoronoiCellLedger}^{\uparrow}$ packet records the finite ledger by which nearest-site cells are read from displayed metric comparisons before any host planar subdivision or selected closest-point theorem is available. It sits after the metric carrier of chapter 57, the finite-net routes represented by $\mathsf{FiniteNet}^{\uparrow}$ and chapter 7465, the convex-set carrier of chapter 208, the polytope carrier of chapter 209, and the terminal real-name seal of chapter 38. The chapter names finite site, distance-comparison, bisector, halfspace, cell-boundary, real-comparison, transport, replay, provenance, and local NameCert rows; it does not claim a global Delaunay dual, a host Euclidean order theorem, selected nearest points, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7486.1 (Voronoi cell ledger carrier)**.** *A* $\mathsf{VoronoiCellLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S, M, D, B, A, C, R, H, T, P, N)$$

with the following displayed rows:

1. *S is the finite site-set row, listing the site names and the finite-net address from which every inspected site is read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *metric source row for the displayed distance comparisons.*
3. *D is the distance-comparison ledger, recording for each inspected query row the finite comparison between the chosen site and every competing site in S.*
4. *B is the bisector ledger, naming the equal-distance boundary rows induced by pairs of displayed sites.*
5. *A is the halfspace ledger, recording which side of each bisector keeps the selected site no farther than the competitor.*
6. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{Polytope}^{\uparrow}$ *cell-boundary row obtained by intersecting the listed halfspace rows.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison seal for the distance and boundary inequalities after the finite comparison ledger is visible.*

8. *H records componentwise* hsame *transport for the site set, metric row, comparison ledger, bisector rows, halfspace rows, cell boundary, real seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a nearest-site request through S, M, D, B, A, C, R.*
10. *P is the* Pkg *provenance over* $\mathsf{VoronoiCellLedger}^{\uparrow}$, $\mathsf{FiniteNet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{VoronoiCellLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, M, D, B, A, C, R, H, T, P, N$. It has no coordinate for an infinite site family, a host nearest-neighbor selector, a completed Euclidean diagram, quotient real equality, trichotomy, countable choice, Delaunay duality, or ambient real completeness.

Theorem 7486.2 (Voronoi cell ledger NameCert obligations)**.** *For every accepted* $\mathsf{VoronoiCellLedger}^{\uparrow}$ *carrier $V = (S, M, D, B, A, C, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VoronoiCellLedger}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *site admission for the displayed finite site set and finite-net address row S;*
2. *metric comparison admission for the* $\mathsf{Met}^{\uparrow}$ *source row M and the distance-comparison ledger D;*
3. *bisector exposure, requiring every equal-distance boundary row to be named in B from a displayed pair of sites;*
4. *halfspace exposure, requiring every selected-site inequality to be read through the corresponding row of A;*
5. *convex and polytope boundary exposure, requiring the cell boundary row C to be the listed intersection of the accepted halfspace rows;*
6. *real-comparison sealing, requiring every exported distance or boundary comparison to pass through the* $\mathsf{Real}^{\uparrow}$ *seal R;*
7. *transport, replay, provenance, and local naming stability through H, T, P, N;*
8. *ledger non-escape, excluding selected nearest points, host diagram completion, quotient real equality, trichotomy, countable choice, Delaunay duality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 7486.1. The finite site row S and metric row M are the only sources for the distance comparisons listed in D.

The boundary rows are downstream of those comparisons. A bisector row in B is attached to a displayed pair of sites, and a halfspace row in A records the selected-site side of that same comparison before the convex cell row C can be read.

The real-facing route is sealed only after the finite comparison and boundary rows are visible. Thus R can export distance and boundary inequalities, but cannot add a nearest-neighbor selector or a completed host diagram.

Finally, H, T, P, N transport, replay, source, and name the same displayed packet. Since the carrier has no coordinate for the excluded selector, quotient, choice, duality, trichotomy, or completeness data, the listed rows exhaust the local NameCert obligations. □

Theorem 7486.3 (Voronoi cell nearest-site route)**.** *Every nearest-site or cell-boundary consumer of an accepted* $\mathsf{VoronoiCellLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{FiniteNet}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto D \longmapsto B \longmapsto A \longmapsto \mathsf{ConvexSet}^{\uparrow}/\mathsf{Polytope}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives a displayed ledger of site comparisons, bisectors, halfspaces, convex-cell boundaries, and real-comparison seals; it receives no selected nearest point, host planar subdivision, quotient real equality, countable-choice schedule, Delaunay dual, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the site and metric rows. The finite-net source bounds the site inventory, and the metric row supplies the only distance readings used by the ledger.

Read the comparison and boundary rows next. The ledger D compares the chosen site against the finite competitors, the bisector row B records equal-distance boundaries, and the halfspace row A records the side that keeps the chosen site no farther than each competitor.

The convex-cell row C is read after those halfspaces, as a $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$ boundary assembled from the finite ledger. The real seal R exports only the displayed comparisons already present in D, B, A, C.

Transport, replay, provenance, and local naming preserve the same route. Therefore any nearest-site consumer remains inside the finite ledger and cannot obtain the excluded selector, host diagram, quotient, choice, duality, or completeness data from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{VoronoiCellLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VoronoiCellLedger}^{\uparrow}$ packet is generated from a finite site set, a MetricUp distance row, distance-comparison rows, bisector rows, halfspace rows, ConvexSet and Polytope cell-boundary rows, a RealUp comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VoronoiCellLedger}^{\uparrow}}$, seed)

Scope closed: Definition 7486.1, Definition 7486.2, and Definition 7486.3 seed the finite Voronoi cell ledger over metric, finite-net, convex, polytope, real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{VoronoiCellLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global Voronoi diagram construction, Delaunay duality, selected nearest points, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked site admission, metric-comparison admission, bisector exposure, halfspace exposure, convex-cell boundary exposure, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.VoronoiCellLedgerUp.

Chapter 7487

A Concrete Naming Certificate for $\mathsf{HorosphereShearLedger}^{\uparrow}$

Following the standard hyperbolic-geometry reading of Busemann functions, horospheres, and shear coordinates, the $\mathsf{HorosphereShearLedger}^{\uparrow}$ packet records the finite route by which a boundary point, a horosphere row, and a shear transport comparison are read before any global Teichmuller or measured-lamination theorem is available. It is grounded by the hyperbolic observer-expansion vision of chapter 9463, and it sits after the metric carrier of chapter 3474, the Busemann-boundary packet of chapter 3732, the horosphere-flow ledger of chapter 4051, the boundary-shadow packet of chapter 3813, and the terminal real-name seal of chapter 38. The chapter names finite boundary, horosphere, Busemann-comparison, shear-transport, metric-readback, real-seal, transport, replay, provenance, and local NameCert rows; it does not claim complete hyperbolic geometry, ideal boundary compactness, selected geodesic representatives, quotient boundary equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7487.1 (Horosphere shear ledger carrier)**.** *A* $\mathsf{HorosphereShearLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, O, B, S, M, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the boundary-point row read through* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$*.*
2. *O is the horosphere row, naming the displayed Busemann level set attached to Q.*
3. *B is the Busemann comparison row, recording the finite level and cocycle comparisons used by the shear reading.*
4. *S is the shear-transport row, carrying the displayed displacement between adjacent horosphere reads.*
5. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-readback row for the distance and displacement comparisons used by B and S.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for exported Busemann, metric, and shear comparisons after Q, O, B, S, M are visible.*
7. *H records componentwise* hsame *transport for boundary point, horosphere, Busemann comparison, shear transport, metric readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-shear request through Q, O, B, S, M, R.*

9. *P is the* Pkg *provenance over* $\mathsf{HorosphereShearLedger}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HorosphereShearLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, O, B, S, M, R, H, C, P, N$. It has no coordinate for an ideal-boundary quotient, a selected geodesic representative, global horosphere foliation, measured lamination, completed Teichmuller shear coordinate, countable choice, trichotomy, or ambient real completeness.

Theorem 7487.2 (Horosphere shear ledger NameCert obligations)**.** *For every accepted* $\mathsf{HorosphereShearLedger}^{\uparrow}$ *carrier $L = (Q, O, B, S, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HorosphereShearLedger}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *boundary-point exposure through the displayed* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row Q;*
2. *horosphere exposure through the Busemann level row O;*
3. *Busemann comparison exposure through the finite comparison ledger B;*
4. *shear-transport exposure through the displayed transport row S;*
5. *hyperbolic metric readback through M before any exported displacement is read;*
6. *real-comparison sealing through R;*
7. *transport, replay, provenance, and local naming stability through H, C, P, N;*
8. *ledger non-escape, excluding ideal-boundary quotient equality, selected geodesic representatives, global horosphere foliations, completed shear-coordinate theory, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 7487.1. The boundary point Q and horosphere row O are the only sources for the Busemann comparison ledger B.

The shear row is downstream of that comparison. It records a finite transport between displayed horosphere reads, while the hyperbolic metric row M supplies the distance readback used by the comparison and transport rows.

The real seal R is read after the boundary, horosphere, Busemann, shear, and metric rows. It can export displayed real-valued comparisons, but it cannot introduce an ideal-boundary quotient or a selected geodesic representative.

Finally, H, C, P, N preserve, replay, source, and name the same packet. Since the carrier lacks coordinates for the excluded quotient, global, choice, trichotomy, and completeness data, the listed rows exhaust the local NameCert surface. □

Theorem 7487.3 (Horosphere shear transport route)**.** *Every boundary-shadow or horosphere-flow consumer of an accepted* $\mathsf{HorosphereShearLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto O \longmapsto B \longmapsto S \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may be consumed by $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *and* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *rows, but it supplies only displayed boundary-point, horosphere, Busemann, metric-readback, shear-transport, real-seal, transport, replay, provenance, and local NameCert data.*

Proof. Read the boundary and horosphere rows first. The boundary point Q comes from the Busemann-boundary packet, and the horosphere row O names the displayed level set attached to that boundary point.

The Busemann comparison row B and shear-transport row S are then read over the same displayed boundary and horosphere data. The hyperbolic metric row M supplies the distance readback used by the shear comparison before the terminal real seal R is visible.

Downstream $\mathsf{HorosphereFlowLedger}^{\uparrow}$ and $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ consumers can therefore inspect the shear route only through Q, O, B, S, M, R. The structural rows H, C, P, N keep that route stable under transport and replay, so no global foliation, selected geodesic, quotient boundary equality, choice schedule, trichotomy branch, or ambient real-completeness datum is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{HorosphereShearLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HorosphereShearLedger}^{\uparrow}$ packet is generated from a HyperbolicBusemannBoundary boundary-point row, a horosphere row, a Busemann comparison row, a shear-transport row, a HyperbolicMetric readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HorosphereShearLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7487.1, Definition 7487.2, and Definition 7487.3 seed the finite horosphere-shear ledger over boundary, horosphere, Busemann, metric, real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HorosphereShearLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close complete hyperbolic geometry, ideal-boundary compactness, selected geodesic representatives, quotient boundary equality, global horosphere foliations, measured laminations, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked boundary-point exposure, horosphere exposure, Busemann comparison exposure, shear-transport exposure, hyperbolic metric readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.HorosphereShearLedgerUp.

Chapter 7488

A Concrete Naming Certificate for $\mathsf{BishopCauchyFusion}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy sequence arguments, the $\mathsf{BishopCauchyFusion}^{\uparrow}$ packet records the finite route by which two regular rational Cauchy sources are read through paired stream windows and one shared dyadic budget before a Real-facing fusion seal is exposed. It sits beside the regular rational name surface of chapter 301, the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the reusable tail-modulus fusion route of chapter 1024, and the terminal real-name seal of chapter 38. The packet names a two-witness fusion route for the Real and Completion corridor without quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7488.1 (Bishop Cauchy fusion carrier)**.** *A* $\mathsf{BishopCauchyFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W_A, W_B, D, F, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are to be fused.*
2. *W_A and W_B are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which the two sources are inspected.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, recording the single dyadic tolerance service used by both windows.*
4. *F is the* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *row that binds the two source tails to the shared budget before a seal-facing read is allowed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row produced only after A, B, W_A, W_B, D, F have been displayed.*
6. *H records componentwise* hsame *transport for the two sources, paired windows, shared budget, fusion row, Real seal, continuation row, package row, and local naming row.*
7. *C records displayed* Cont *replay from a requested tolerance through the two windows, the shared budget, the fusion row, and then to the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyFusion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailModulusFusion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyFusion}^{\uparrow}}$ *row.*

The carrier exposes only this finite two-witness route. It has no row for quotient stream equality, selected host limits, host real equality, countable-choice schedules, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7488.2 (Bishop Cauchy fusion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyFusion}^{\uparrow}$ *carrier*

$$K = (A, B, W_A, W_B, D, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyFusion}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget, one* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *row, one terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every seal-facing read must keep both source rows and both window rows attached to the same budget row* D *before* F *and* R *are consumed.*

Proof. Project the carrier of Definition 7488.1. Rows A, B, W_A, W_B, D, F, R are the analytic route, and rows H, C, P, N provide transport, replay, provenance, and local naming.

Carrier admission reads the two regular sources and paired windows first. The budget obligation then identifies D as the only dyadic tolerance row available to both windows, so the two branches cannot be read through private tolerance services.

The fusion obligation is the next step. Row F is consumed only after A, B, W_A, W_B, D are displayed, and the Real seal R is downstream of that same fusion row.

Finally, H, C, P, N preserve the route as finite NameCert data. Since quotient streams, selected host limits, host equality, countable choice, trichotomy, and ambient completeness are absent from the carrier, the listed obligations exhaust the local surface. □

Theorem 7488.3 (Bishop Cauchy fusion single-window handoff)**.** *Every accepted* $\mathsf{BishopCauchyFusion}^{\uparrow}$ *packet exposes a single shared observation window for the two-witness fusion route: a consumer may inspect* W_A *and* W_B *only through the common dyadic budget* D *and the fusion row* F*. Thus the handoff from two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources to the terminal* $\mathsf{Real}^{\uparrow}$ *seal is a single finite route through*

$$A,\ B,\ W_A,\ W_B,\ D,\ F,\ R,$$

not two independent Real-seal routes.

Proof. Start with the NameCert obligations of Definition 7488.2. They keep both source rows attached to the paired stream windows and require one shared dyadic budget row D before the fusion row is available.

The handoff is therefore controlled by D. Any read of W_A or W_B that reaches F has already passed through the same budget service, so the consumer receives one displayed fusion route rather than two detached source reads.

The terminal Real seal R is downstream of F. Because no second Real-seal coordinate appears in the carrier, the handoff remains a single finite route through the displayed rows. □

Theorem 7488.4 (Bishop Cauchy fusion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{BishopCauchyFusion}^{\uparrow}$ *packet factors through the two regular sources, paired stream windows, shared dyadic budget, and fusion row before the terminal seal* R *is consumed. The read exports no quotient stream equality, selected host limit, host real equality, countable-choice row, trichotomy witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the carrier rows of Definition 7488.1. The only Real-facing coordinate is R, and it appears after the two sources A, B, the windows W_A, W_B, the shared budget D, and the fusion row F.

Apply the single-window handoff theorem. It forces the two source branches to meet at the same dyadic budget and the same fusion row before a Real-facing consumer can read the seal.

The structural rows H, C, P, N only transport, replay, source, and name the same finite packet. Since the refused host-level data are not rows of K, a consumer of R cannot escape the displayed fusion route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyFusion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyFusion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7488.1, Definition 7488.2, Definition 7488.3, and Definition 7488.4 seed the finite two-witness Cauchy fusion route through paired windows, one dyadic budget, one fusion row, and one Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyFusion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-window exposure, shared dyadic budget exactness, fusion-row ordering, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyFusionUp

Constructive story: A $\mathsf{BishopCauchyFusion}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two StreamName windows, one DyadicRatCore budget row, a CauchyTailModulusFusion row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7489

A Concrete Naming Certificate for $\mathsf{RealArchimedeanWindow}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArchimedeanWindow}^{\uparrow}$ packet records the finite inspected window by which a displayed $\mathsf{Real}^{\uparrow}$ name is bracketed between dyadic lower and upper rows at a positive radius before any Real or Completion consumer reads an order-facing statement. It sits over the terminal real seal of chapter 38, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic endpoint ledger of chapter 291, the positive-radius surface of chapter 1587, and the Archimedean property route of chapter 3372. The packet names a displayed bracketing carrier; it is not a completeness theorem, trichotomy principle, quotient-stream equality test, selected-limit operator, or open-cover compactness claim.

Definition 7489.1 (RealArchimedeanWindow carrier)**.** *A* $\mathsf{RealArchimedeanWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, S, Q, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row whose inspected name is being bracketed;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule selecting the finite windows consumed by the read;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic-core ledger for the bracket arithmetic;*
5. *L is the lower dyadic bracket row;*
6. *U is the upper dyadic bracket row;*
7. *H records componentwise* hsame *transport for the real seal, schedule, readback, dyadic core, lower and upper brackets, provenance, and local naming row;*
8. *C records displayed* Cont *replay for the same inspected bracketing route;*
9. *P is the* Pkg *provenance over* $\mathsf{RealArchimedeanWindow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}$ *row.*

The positive radius is carried as part of the local dyadic-window read through D, L, U and the sibling $\mathsf{PositiveReal}^{\uparrow}$ *provenance. No coordinate stores a quotient real representative, selected limit point, host trichotomy decision, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RealArchimedeanWindowUp.TasteGate.RealArchimedeanWindowTasteGate single carrier alignment`

Theorem 7489.2 (RealArchimedeanWindow NameCert obligations)**.** *For an accepted* $\mathsf{RealArchimedeanWindow}^{\uparrow}$ *carrier* $W = (R, S, Q, D, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}$ *surface consists exactly of carrier admission, sealed* $\mathsf{Real}^{\uparrow}$ *exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *observation scheduling,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket arithmetic, lower and upper dyadic endpoint exposure, positive-radius provenance, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.RealArchimedeanWindowUp.RealArchimedeanWindowNameCertObligations`

Proof. Carrier admission is projection from Definition 7489.1. The accepted packet exposes only the rows $R, S, Q, D, L, U, H, C, P, N$.

The observation route reads S before Q, and reads D before the lower and upper endpoint rows. Thus the bracket is an inspected dyadic window over a displayed regular rational readback rather than a host comparison of real numbers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-stream equality coordinate, selected-limit coordinate, trichotomy coordinate, or completeness coordinate, the local certificate cannot export any of those principles. □

Theorem 7489.3 (RealArchimedeanWindow dyadic bracket)**.** *Every accepted* $\mathsf{RealArchimedeanWindow}^{\uparrow}$ *packet exposes its lower and upper bracket only through the shared dyadic-core row* D*, the inspected stream schedule* S*, and the regular readback row* Q*. The lower row* L *and upper row* U *are consumed as one finite bracketing surface for the sealed real row* R *and the positive-radius provenance in* P*; they do not supply Real completeness, trichotomy, quotient stream equality, selected limits, or open-cover compactness.*

Proof. Open the carrier $W = (R, S, Q, D, L, U, H, C, P, N)$. The two endpoint rows L and U are not independent order witnesses; they are named after the schedule, readback, and dyadic-core rows have been displayed.

Use the NameCert obligations above. They make the bracketing route pass through R, S, Q, D, L, U before transport, replay, provenance, or local naming is read.

The exclusion follows by coordinate inspection. A completeness theorem, trichotomy decision, quotient stream equality test, selected-limit operator, and open-cover compactness witness are absent from the carrier, so no consumer of the window can project them from the dyadic bracket. □

Lean4 ✓ `BEDC.Derived.RealArchimedeanWindowUp.RealArchimedeanWindowCarrier dyadic bracket handoff`

Theorem 7489.4 (RealArchimedeanWindow bound consumer exactness)**.** *Every order-facing or completion-facing consumer of an accepted* $\mathsf{RealArchimedeanWindow}^{\uparrow}$ *packet must read the* $\mathsf{PositiveReal}^{\uparrow}$ *radius provenance and the* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *bound route together with the displayed rows* $R, S, Q, D, L, U, H, C, P, N$ *before treating the bracket as a bound. The consumer route factors through the positive-radius sibling of chapter 1587, the Archimedean-property sibling of chapter 3372, the local carrier of Definition 7489.1, the NameCert obligations of Definition 7489.2, and the dyadic-bracket handoff of Definition 7489.3.*

Proof. Project the carrier $W = (R, S, Q, D, L, U, H, C, P, N)$ from Definition 7489.1. The only radius-facing entry available to a consumer is the provenance row P, where the $\mathsf{PositiveReal}^{\uparrow}$ dependency is recorded together with the local $\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}$ row.

Use the NameCert obligations from Definition 7489.2. They force the order-facing read to keep $R, S, Q, D, L, U, H, C, P, N$ in the same finite packet, so the lower and upper endpoints cannot be separated from their stream schedule, regular readback, dyadic core, transport, replay, provenance, or local name.

Apply the dyadic-bracket handoff of Definition 7489.3. The bracket is consumed only after D, S, Q, L, U and the positive-radius provenance have been displayed. Therefore an order or completion consumer that treats the bracket as a bound must also carry the sibling $\mathsf{RealArchimedeanProperty}^{\uparrow}$ route named in the chapter interface, rather than projecting a host order axiom, trichotomy decision, quotient-real equality test, selected-limit operator, or ambient completeness witness from the window alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7489.1, Definition 7489.2, and Definition 7489.3 seed the inspected dyadic-window carrier over Real, StreamName, RegSeqRat, DyadicRatCore, and PositiveReal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArchimedeanWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient stream equality, selected limits, open-cover compactness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, inspected-window scheduling, dyadic endpoint exactness, positive-radius provenance, transport, replay, and local naming for BEDC.Derived.RealArchimedeanWindowUp.

Constructive story: A $\mathsf{RealArchimedeanWindow}^{\uparrow}$ packet is generated from a sealed Real row, finite StreamName observation schedule, RegSeqRat readback, DyadicRatCore bracket ledger, lower and upper dyadic endpoint rows, positive-radius provenance, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 7490

A Concrete Naming Certificate for $\mathsf{DyadicMidpointChain}^{\uparrow}$

Following Bishop and Bridges constructive interval analysis, the $\mathsf{DyadicMidpointChain}^{\uparrow}$ packet records finite dyadic midpoint chains as displayed BEDC data. It sits after the dyadic rational core of chapter 291, the midpoint interval carrier of chapter 7331, the bisection branch ledger of chapter 361, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite midpoint-refinement route; it is not a quotient stream equality, selected limit, trichotomy principle, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7490.1 (Dyadic midpoint chain carrier)**.** *A* $\mathsf{DyadicMidpointChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, Q, B, O, W, A, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the endpoint-pair rows, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and displayed as the current closed dyadic interval.*
2. *Q is the midpoint row, recording the dyadic midpoint computed from the displayed endpoint pair.*
3. *B is the retained-side ledger, listing the finite left-or-right midpoint decisions accepted by the refinement.*
4. *O is the finite chain-order row, asserting that the retained intervals are read in the displayed midpoint-refinement order.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing exactly the stages inspected by a downstream consumer.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and midpoint decisions.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after L, R, Q, B, O, W, A have been displayed.*
8. *H records componentwise* hsame *transport for endpoint, midpoint, retained-side, chain-order, stream-window, readback, real-seal, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay from an interval-refinement request through L, R, Q, B, O, W, A, E.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicMidpointChain}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMidpointInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpointChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient equality of streams, choice-selected limits, trichotomy on reals, open-cover compactness, Heine–Borel, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointChainUp.TasteGate.DyadicMidpointChainTasteGate single carrier alignment`

Theorem 7490.2 (Dyadic midpoint chain NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMidpointChain}^{\uparrow}$ *carrier*

$$M = (L, R, Q, B, O, W, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMidpointChain}^{\uparrow}}$ *surface consists exactly of endpoint-pair admission through L, R, midpoint exposure through Q, retained-side ledger exactness through B, finite chain-order exposure through O,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through A, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7490.1. The endpoint, midpoint, retained-side, chain-order, stream-window, readback, real-seal, transport, replay, provenance, and naming rows are the whole packet.

Read the analytic rows in order. The endpoint rows L, R supply the dyadic interval, the midpoint row Q computes the displayed midpoint, and the retained-side ledger B records which side survives each finite refinement.

The chain-order row O then keeps those retained intervals in one displayed midpoint-refinement sequence. The window and readback rows W, A expose only finite observations of that sequence, and the seal row E is downstream of those reads.

Finally H, C, P, N transport, replay, source, and name the same data. Since no carrier coordinate stores quotient stream equality, selected limits, trichotomy, open-cover compactness, Heine–Borel, or ambient real completeness, none of those principles belongs to the local obligation surface. □

Theorem 7490.3 (Dyadic midpoint chain regular-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicMidpointChain}^{\uparrow}$ *packet reads through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{DyadicMidpointInterval}^{\uparrow} \longmapsto \mathsf{DyadicBisection}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using $L, R, Q, B, O, W, A, E, H, C, P, N$. The handoff cannot use quotient stream equality, selected limits, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $M = (L, R, Q, B, O, W, A, E, H, C, P, N)$. A regular-rational or real-facing read must first expose the endpoint and midpoint rows L, R, Q before a retained interval is visible.

The retained-side ledger B and chain-order row O then determine the finite midpoint-refinement chain. These rows are the only source of branch decisions, so a consumer cannot replace them by a selected point or a host trichotomy test.

The window row W and readback row A provide the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observations through which the terminal seal E is consumed. Componentwise transport and replay through H, C preserve that same route, while P, N source and name it.

Thus every downstream read factors through the displayed midpoint chain. The carrier has no coordinate for quotient stream equality, selected limits, open-cover compactness, Heine–Borel, or ambient real completeness, so the handoff exports none of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMidpointChain}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMidpointChain}^{\uparrow}$ packet is generated from dyadic endpoint rows, a midpoint row, a retained-side ledger, finite chain order, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMidpointChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7490.1, Definition 7490.2, and Definition 7490.3 seed the finite midpoint-refinement route for regular-Cauchy and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMidpointChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, open-cover compactness, Heine–Borel, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint-pair admission, midpoint exposure, retained-side ledger exactness, chain-order exposure, finite-window readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMidpointChainUp.

Chapter 7491

A Concrete Naming Certificate for $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$

Following Bishop and Bridges, a located dyadic interval is useful when a finite bracketing object keeps both endpoints, the positive gap, and the comparison route visible before any Real-facing consumer reads a point or completion seal. The $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ packet sits over the dyadic arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the terminal Real seal of chapter 38, the bisection route of chapter 2018, and the located dyadic real packet of chapter 6192. It names the interval-level carrier between endpoint data and Real-facing located completion; it does not select a midpoint limit, quotient Cauchy names, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7491.1 (Bishop located dyadic interval carrier)**.** *A* $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, G, S, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and retained as a displayed dyadic rational.*
2. *U is the upper endpoint row, read through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
3. *G is the positive located-gap row, recording the finite evidence that the lower endpoint lies apart from or below the upper endpoint by a displayed dyadic margin.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite endpoint windows requested by a comparison or bisection consumer.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows.*
6. *B is the* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *bisection row, recording the retained left or right subinterval and the dyadic midpoint read used to make that decision.*
7. *R is the* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff row, available only after L, U, G, S, Q, B have been displayed.*
8. *H records componentwise* hsame *transport for endpoints, gap evidence, stream windows, regular readback, bisection, handoff, provenance, replay, and local naming.*

9. *C records displayed* Cont *replay from endpoint reads through gap checks, finite windows, regular readbacks, bisection decisions, and the Real-facing handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicBinarySearch}^{\uparrow}$, $\mathsf{LocatedDyadicReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDyadicInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, G, S, Q, B, R, H, C, P, N$. It has no coordinate for a hidden midpoint selector, quotient real equality, countable choice, detached trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7491.2 (Bishop located dyadic interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ *carrier*

$$I = (L, U, G, S, Q, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedDyadicInterval}^{\uparrow}}$ *surface consists exactly of lower endpoint exposure, upper endpoint exposure, positive located-gap accounting, finite StreamName window inspection, RegSeqRat endpoint readback, dyadic bisection readback, Real-facing handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may erase an endpoint, replace the gap by detached order trichotomy, select a midpoint limit, quotient Cauchy names, or export ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7491.1. The endpoint rows L, U and the located-gap row G are the only source data for the interval boundary.

The finite observation route is S followed by Q. A comparison or bisection consumer may inspect endpoint windows and regular rational readbacks, but it cannot replace them by quotient real names or by a hidden global order oracle.

The bisection row B is downstream of the same endpoint and gap rows. It records the displayed midpoint read and retained subinterval, so it supplies readback data rather than a selected limit point.

Finally R, H, C, P, N hand off, transport, replay, source, and name the same finite interval packet. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7491.3 (Bishop located dyadic interval bisection readback)**.** *Every accepted* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *consumer of a* $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicBinarySearch}^{\uparrow}.$$

The bisection decision may retain a left or right dyadic subinterval, but it must keep the original endpoint rows, located-gap row, finite window row, regular readback row, transport, replay, provenance, and local naming visible.

Proof. Start with the endpoint and gap rows L, U, G. A binary-search read is an interval read, so it cannot begin from an unlogged midpoint or a host real order comparison.

Read S and Q next. These rows supply the finite endpoint windows and regular rational readback at the dyadic precision requested by the bisection consumer.

The row B then records the actual bisection decision. The retained subinterval is another displayed dyadic interval row attached to the same provenance, not a chosen limit and not a quotient of all later refinements.

Transport, replay, provenance, and local naming through H, C, P, N preserve this route. Thus every bisection consumer factors through the stated finite chain and keeps the interval boundary data visible. □

Theorem 7491.4 (Bishop located dyadic interval Real handoff)**.** *Every* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ *packet must first read*

$$L, U, G, S, Q, B, H, C, P, N.$$

The handoff row R may expose a Real-facing located interval boundary only after the dyadic endpoints, located gap, finite windows, regular rational readback, and bisection route have been displayed. It cannot export quotient real equality, countable choice, detached trichotomy, a selected point of the interval, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the interval boundary rows. The lower endpoint, upper endpoint, and positive gap are dyadic data, so a Real-facing consumer must receive them through $\mathsf{DyadicRatCore}^{\uparrow}$ rather than through ambient real equality.

Replay the observation rows S, Q, B. Stream windows and regular rational readbacks expose the finite endpoint information, while the bisection row records how a retained subinterval was read.

Only after those rows are public can R be consumed. The handoff is therefore a seal on the displayed interval route, not a selected point or a proof of ambient completeness.

The structural rows H, C, P, N preserve the same route under hsame transport, Cont replay, provenance, and local naming. Hence every $\mathsf{LocatedDyadicReal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer remains inside the stated finite packet. □

Falsifiable prediction: A $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ consumer that reaches $\mathsf{LocatedDyadicReal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ after deleting one of L, U, G, S, Q, B refutes the carrier of Definition 7491.1.

Independence witness: Nearest siblings: dyadicratcore, streamname, regseqrat, dyadic-binary-search, located-dyadic-real, and real. $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ is not bijective to any one of them because it jointly carries two dyadic endpoints, positive gap evidence, finite endpoint windows, regular readback, bisection readback, Real-facing handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedDyadicInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedDyadicInterval}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, positive located-gap data, StreamName endpoint windows, RegSeqRat readback, DyadicBinarySearch bisection readback, a LocatedDyadicReal or Real handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedDyadicInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7491.1, Definition 7491.2, Definition 7491.3, and Definition 7491.4 seed the finite located dyadic interval route between endpoint data, bisection readback, and Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedDyadicInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, countable choice, detached trichotomy, a selected interval point, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, positive located-gap accounting, finite window inspection, RegSeqRat readback, bisection readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedDyadicIntervalUp.

Chapter 7492

A Concrete Naming Certificate for $\mathsf{BourbakiFilterLimit}^{\uparrow}$

Following Bourbaki and Kelley on filter convergence, the $\mathsf{BourbakiFilterLimit}^{\uparrow}$ packet records the finite limit-reading surface by which a displayed filter base may be routed to a completion-facing real seal. It sits after the basis carrier of chapter 1601, the Cauchy-filter surface of chapter 333, the convergence-filter packet of chapter 6578, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a Bourbaki–Kelley limit route as finite BEDC data; it does not import quotient filters, selected limits, countable choice, host filter equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7492.1 (Bourbaki filter-limit carrier)**.** *A* $\mathsf{BourbakiFilterLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, C, V, W, R, D, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *basis row, listing the finite basic neighborhoods inspected by the limit read.*
2. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, recording the Cauchy refinement obligations available to the inspected basis windows.*
3. *V is the* $\mathsf{ConvergenceFilter}^{\uparrow}$ *row, recording the convergence-facing neighborhood requests and their finite witnesses.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which sequential observations of the same route are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational names exposed by the stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the Cauchy and convergence requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, C, V, W, R, D have been displayed.*
8. *H is the componentwise* hsame *transport row for basis, Cauchy, convergence, stream, rational, dyadic, and real-seal rows.*
9. *T is the displayed* Cont *replay row for the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{BourbakiFilterLimit}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{ConvergenceFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BourbakiFilterLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, C, V, W, R, D, E, H, T, P, N$. It has no coordinate for quotient filters, a selected limit point, countable choice, host equality of filters, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BourbakiFilterLimitUp.BourbakiFilterLimitTasteGate single carrier alignment`

Theorem 7492.2 (Bourbaki filter-limit NameCert obligations)**.** *For the carrier of Definition 7492.1, the local* $\mathsf{NameCert}_{\mathsf{BourbakiFilterLimit}^{\uparrow}}$ *surface consists exactly of eight obligation groups:*

1. *carrier admission for $L = (B, C, V, W, R, D, E, H, T, P, N)$;*
2. *filter-base coverage through B;*
3. *Cauchy-filter refinement through C;*
4. *convergence-filter neighborhood coverage through V;*
5. *stream and regular-rational readback through W and R;*
6. *dyadic tolerance exactness and real-seal handoff through D and E;*
7. *componentwise transport and replay through H and T;*
8. *package provenance and local naming through P and N.*

Thus the certificate presents the Bourbaki–Kelley filter-limit route as finite BEDC data and cannot be read as quotient filters, selected limits, countable choice, host filter equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is tuple projection from Definition 7492.1. The filter side is exhausted by B, C, V: the basis row supplies inspected basic neighborhoods, the Cauchy row supplies finite refinement obligations, and the convergence row supplies the displayed neighborhood witnesses.

The completion-facing read is separate. Rows W and R expose only finite stream windows and regular rational readback, while D fixes the dyadic tolerance ledger before the terminal real seal E is consumed.

Finally inspect H, T, P, N. These rows transport, replay, package, and name the same displayed route. Since no remaining coordinate stores quotient filters, selected limits, countable choice, host filter equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the eight groups exhaust the local certificate surface. □

Theorem 7492.3 (Bourbaki filter-limit real-completion handoff)**.** *Every accepted* $\mathsf{BourbakiFilterLimit}^{\uparrow}$ *packet that is consumed by a real-completion route reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through*

$$\mathsf{FilterBase}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{ConvergenceFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff exposes basis coverage, Cauchy refinement, convergence neighborhoods, finite stream windows, rational readback, and dyadic tolerances before any real-facing consumer reads the seal. It exports no quotient filter, selected limit, countable-choice schedule, host filter equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the obligation theorem Definition 7492.2. Its first four groups force every accepted packet to display the filter base, Cauchy-filter refinement, and convergence-filter neighborhood rows before the route can be named.

Next read the completion-facing rows. The same obligation surface requires the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows together with the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before the terminal $\mathsf{Real}^{\uparrow}$ seal appears.

The transport, replay, provenance, and local naming rows preserve only that displayed finite route. Because the carrier has no coordinate for quotient filters, selected limits, countable choice, host filter equality, or ambient completeness, none of those objects can be exported by the real-completion handoff. □

Falsifiable prediction: Every accepted $\mathsf{BourbakiFilterLimit}^{\uparrow}$ completion read exposes FilterBase, CauchyFilter, ConvergenceFilter, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any terminal real seal is consumed.

Independence witness: filterbase, cauchyfilter, convergencefilter, streamname, regseqrat, dyadicratcore, real: none carries the joint Bourbaki–Kelley filter-limit packet because this chapter alone binds basis coverage, Cauchy refinement, convergence neighborhoods, finite stream windows, rational readback, dyadic tolerances, transport, replay, provenance, and local naming without quotient filters or choice.

Closure status for $\mathsf{NameCert}_{\mathsf{BourbakiFilterLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BourbakiFilterLimit}^{\uparrow}$ packet is generated from FilterBase basis rows, CauchyFilter refinement rows, ConvergenceFilter neighborhood rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BourbakiFilterLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7492.1, Definition 7492.2, and Definition 7492.3 seed the finite Bourbaki–Kelley filter-limit route from filter-base convergence data to regular rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BourbakiFilterLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BourbakiFilterLimitUp.BourbakiFilterLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host filter equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base coverage, Cauchy-filter refinement, convergence-neighborhood coverage, finite window coverage, rational readback, dyadic tolerance exactness, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BourbakiFilterLimitUp.

Chapter 7493

A Concrete Naming Certificate for $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$

Following Gromov and Bridson–Haefliger, a visual cross-ratio cocycle is the finite boundary packet that keeps four visual boundary shadows, the visual metric comparison, the cross-ratio read, and the Busemann cocycle ledger visible in one route. The $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ packet sits after the boundary-shadow carrier of chapter 3813, the visual metric surface of chapter 3866, the hyperbolic cross-ratio packet of chapter 3948, the Busemann cocycle packet of chapter 4134, and the terminal Real seal of chapter 38. It names a single finite cocycle carrier for visual boundary comparison; it does not complete the boundary, quotient geodesic rays, select ideal representatives, or import host global geometry.

Definition 7493.1 (Hyperbolic visual cross-ratio cocycle carrier)**.** *A* $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S_0, S_1, S_2, S_3, M, X, B, T, R, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1, S_2, S_3 *are four* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *rows read from the same displayed boundary-shadow route.*
2. M *is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual metric row comparing the four boundary shadows through finite disk and metric data.*
3. X *is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row recording the finite four-point cross-ratio read.*
4. B *is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive boundary-cocycle ledger attached to the same visual comparison.*
5. T *is the transport row keeping the four shadows, visual metric, cross-ratio, and Busemann cocycle in one displayed route.*
6. R *is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite visual comparison value, available only after* $S_0, S_1, S_2, S_3, M, X, B, T$ *have been displayed.*
7. H *records componentwise* hsame *transport for shadows, visual metric, cross-ratio, Busemann cocycle, transport, Real seal, provenance, replay, and local naming.*
8. C *records displayed* Cont *replay from boundary-shadow reads through visual metric comparison, cross-ratio readback, Busemann cocycle replay, transport, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, S_2, S_3, M, X, B, T, R, H, C, P, N$. It has no coordinate for a completed boundary compactification, quotient boundary equality, selected geodesic representatives, detached visual metric equality, or a host global hyperbolic plane.

Theorem 7493.2 (Hyperbolic visual cross-ratio cocycle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ *carrier*

$$V = (S_0, S_1, S_2, S_3, M, X, B, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}}$ *surface consists exactly of four boundary-shadow rows, visual metric comparison, hyperbolic cross-ratio readback, Busemann cocycle replay, route transport, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may collapse the four shadows, detach the visual metric from the displayed boundary route, read the cross-ratio without the Busemann ledger, or replace the packet by a completed boundary object.*

Proof. Project the carrier of Definition 7493.1. The four boundary-shadow rows S_0, S_1, S_2, S_3 are simultaneous fields, so a consumer cannot read a visual comparison from fewer boundary positions.

The comparison rows are M, X, B. The visual metric row supplies finite visual distance data, the cross-ratio row records the four-point comparison, and the Busemann cocycle row records the additive boundary ledger on the same route.

The transport and Real-seal rows T, R are downstream of those displayed rows. Transport keeps the comparison inside one boundary-shadow route, and the Real seal names only the finite comparison value reached by that route.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no completed boundary, quotient equality, selected representative, detached visual metric equality, or host global plane appears in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 7493.3 (Hyperbolic visual cross-ratio cocycle boundary handoff)**.** *Every boundary-facing consumer of an accepted* $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ *packet must factor through*

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicCrossRatio}^{\uparrow} \longmapsto \mathsf{HyperbolicBusemannCocyc}$$

The consumer may replay the route under hsame *transport and* Cont *continuation, but it cannot bypass one of the four boundary-shadow rows, read a detached cross-ratio scalar, quotient the boundary, select geodesic representatives, or invoke host global geometry.*

Proof. Begin with the four boundary-shadow rows. Each is read through $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, so the boundary-facing consumer receives displayed finite shadow data rather than a completed ideal-boundary point.

Read the visual metric and cross-ratio rows next. The row M supplies the visual comparison window, and X records the four-point cross-ratio only after all four shadow rows have been displayed.

The Busemann cocycle row B ties the comparison to an additive boundary ledger. This prevents the cross-ratio from being consumed as an isolated scalar detached from the boundary route.

The route then passes through T, R, H, C, P, N. Transport, Real sealing, replay, provenance, and local naming preserve the same finite packet, so the consumer cannot escape to quotient boundary equality, selected representatives, or host global hyperbolic geometry. □

Falsifiable prediction: A $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ consumer that exports a Real-valued cross-ratio without four boundary-shadow rows, the visual metric row, the Busemann cocycle row, transport, provenance, and local naming refutes the carrier of Definition 7493.1.

Independence witness: Nearest siblings: hyperbolic-boundary-shadow, hyperbolic-visual-metric, hyperbolic-cross-ratio, hyperbolic-busemann-cocycle, and real. $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ is not bijective to any one of them because it jointly carries four shadow rows, visual metric comparison, four-point cross-ratio readback, additive Busemann cocycle replay, route transport, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}$ packet is generated from four HyperbolicBoundaryShadow rows, a HyperbolicVisualMetric comparison, a HyperbolicCrossRatio readback, a HyperbolicBusemannCocycle ledger, route transport, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7493.1, Definition 7493.2, and Definition 7493.3 seed the finite visual boundary cocycle route unifying boundary-shadow, visual-metric, cross-ratio, Busemann, transport, Real-seal, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualCrossRatioCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary compactification, quotient boundary equality, selected geodesic representatives, detached visual metric equality, host global hyperbolic geometry, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked four-shadow exposure, visual metric comparison, cross-ratio readback, Busemann cocycle replay, route transport, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicVisualCrossRatioCocycleUp.

Chapter 7494

A Concrete Naming Certificate for $\mathsf{CoordinateFunctional}^{\uparrow}$

Following Schauder and Banach-space basis theory, the $\mathsf{CoordinateFunctional}^{\uparrow}$ packet records the bounded coefficient readback licensed by a displayed Schauder basis without identifying that readback with an arbitrary element of the dual space. It sits beside the basis packet of chapter 3669, the vector-space carrier of chapter 47, the norm row of chapter 89, the Banach completion surface of chapter 90, the regular rational scalar route of chapter 301, and the real-name modulus seal of chapter 38. The certificate separates the Schauder basis that supplies coefficients from the coordinate functional that consumes one index and exports a bounded scalar read.

Definition 7494.1 (Coordinate functional carrier)**.** *A* $\mathsf{CoordinateFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, V, N, C, Q, M, H, T, P, L)$$

with the following displayed rows:

1. *B is the* $\mathsf{SchauderBasis}^{\uparrow}$ *source row naming the basis-vector stream, finite partial sums, coefficient requests, tail estimates, and reconstruction route used by the coefficient read.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming vector addition, scalar action, zero vector, and finite linear-combination syntax for the vectors on which the coordinate functional is read.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Banach}^{\uparrow}$ *budget row, recording the norm estimates and completion-facing bounds used to treat the coefficient read as bounded.*
4. *C is the coordinate-readback row for a displayed basis index i, sending an accepted vector read to the coefficient row $c_i(x)$ licensed by B.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *scalar-name row for the rational approximants used to present the coefficient value.*
6. *M is the* $\mathsf{Real}^{\uparrow}$*-valued continuity-modulus row, recording the local bound that turns the coefficient readback into a bounded scalar output.*
7. *H records componentwise* hsame *transport for the Schauder basis source, vector-space operations, norm budget, coordinate readback, regular scalar name, continuity modulus, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from vector reads through B, V, N, C, Q, M to the scalar output.*

9. *P records* Pkg *provenance over* $\mathsf{CoordinateFunctional}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{CoordinateFunctional}^{\uparrow}}$ *row.*

The classifier compares packets only through $B, V, N, C, Q, M, H, T, P, L$. It has no coordinate for a Hamel basis choice, arbitrary linear functionals, a full dual-space representation, Hahn–Banach extension, countable choice, or a detached coefficient oracle.

Theorem 7494.2 (CoordinateFunctional coefficient readback)**.** *For every accepted* $\mathsf{CoordinateFunctional}^{\uparrow}$ *carrier $F = (B, V, N, C, Q, M, H, T, P, L)$ and displayed basis index i, a coefficient read $x \mapsto c_i(x)$ is admitted only by first exposing the* $\mathsf{SchauderBasis}^{\uparrow}$ *row B, the vector-space row V, the norm budget N, and the coordinate-readback row C. The scalar output is then named through Q and sealed by M as a* $\mathsf{Real}^{\uparrow}$*-valued bounded read. No coefficient can be read from a host dual vector, a Hamel-basis selector, or an undeclared functional row.*

Proof. Project the carrier of Definition 7494.1. The only row that contains basis coefficients is B, and the only row that turns one displayed index into a scalar request is C.

The vector and norm rows are then load-bearing. The row V supplies the finite linear syntax for the input vector, while N supplies the norm budget needed before the read can be treated as a bounded coordinate.

Finally read the scalar rows Q and M. The regular rational row gives the displayed scalar approximants, and the real-valued modulus row gives the continuity budget. Since no host dual, Hamel selector, or undeclared functional row appears in F, the coefficient readback is exactly the displayed route through B, V, N, C, Q, M. □

Theorem 7494.3 (CoordinateFunctional linearity obligation)**.** *Every accepted* $\mathsf{CoordinateFunctional}^{\uparrow}$ *packet exposes the additivity and scalar-compatibility obligations for its displayed coefficient read only through the rows $B, V, N, C, Q, M, H, T, P, L$. Additivity is checked against the* $\mathsf{VecSpace}^{\uparrow}$ *operations in V and the coefficient-readback row C; scalar compatibility is checked against the same vector-space action and the* $\mathsf{RegSeqRat}^{\uparrow}$ *scalar names in Q. The obligation does not assert linearity for arbitrary functionals outside the packet.*

Proof. Start with the vector-space row. Addition and scalar action are named in V, so the two algebraic tests for the coordinate read are formed only in that displayed syntax.

Use the coefficient row C twice. For additivity it compares the readback on a displayed sum with the displayed sum of coefficient reads; for scalar compatibility it compares the readback on a displayed scalar multiple with the scalar action recorded through Q.

The structural rows H, T, P, L preserve the same tests under transport, replay, provenance, and naming. They do not add a new functional row, so the linearity obligation remains local to the coordinate readback carried by the packet. □

Theorem 7494.4 (CoordinateFunctional continuity modulus)**.** *For every accepted* $\mathsf{CoordinateFunctional}^{\uparrow}$ *carrier $F = (B, V, N, C, Q, M, H, T, P, L)$, the boundedness or continuity claim available to a downstream consumer is the displayed* $\mathsf{Real}^{\uparrow}$*-valued modulus row M read after the Schauder source B, vector operations V, norm budget N, coefficient readback C, and regular scalar name Q. The modulus is a finite NameCert row, not a Hahn–Banach extension, an ambient dual-space norm, or a choice-selected bound.*

Proof. Open the carrier and locate the modulus row. The only continuity datum is M, and it is downstream from the basis, vector-space, norm, coordinate, and scalar-name rows.

The coefficient readback theorem supplies the route into M. A consumer first reads B, V, N, C, Q to obtain a scalar output tied to the displayed basis index, and only then reads the real-valued bound stored in M.

The provenance and local naming rows exclude the external principles listed in the statement. Since Hahn–Banach extension, ambient dual-space norms, and choice-selected bounds have no coordinate in F, the continuity claim is only the finite modulus row recorded by the packet. □

Theorem 7494.5 (CoordinateFunctional NameCert obligations)**.** *For every accepted* $\mathsf{CoordinateFunctional}^{\uparrow}$ *carrier* $F = (B, V, N, C, Q, M, H, T, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CoordinateFunctional}^{\uparrow}}$ *surface consists exactly of carrier admission, Schauder basis source exposure, vector-space operation exposure, norm-budget exposure, coordinate-readback exposure, regular scalar-name exposure, real-valued continuity-modulus exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer of* F *receives those finite rows and no Hamel basis choice, arbitrary linear functional, full dual-space representation, Hahn–Banach theorem, countable-choice schedule, or detached coefficient oracle.*

Proof. Carrier admission is projection from Definition 7494.1. It exposes exactly the tuple $B, V, N, C, Q, M, H, T, P,$

The analytic rows are B, V, N, C, Q, M. They identify the Schauder basis source, the vector-space syntax, the norm budget, the coefficient readback, the regular scalar name, and the real-valued modulus in the order required by the coefficient-readback, linearity-obligation, and continuity-modulus theorems above.

The structural rows H, T, P, L preserve and name that same route. Since the carrier inventory omits Hamel basis choice, arbitrary functionals, dual-space representation, Hahn–Banach extension, countable choice, and detached coefficient oracles, no local NameCert obligation can export them. □

Falsifiable prediction: Every accepted $\mathsf{CoordinateFunctional}^{\uparrow}$ coefficient read factors through a SchauderBasis row, VecSpace operations, Norm and Banach budget rows, coordinate readback, RegSeqRat scalar names, a Real-valued continuity modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: schauder-basis, vecspace, norm, banach, regseqrat, real: no listed sibling supplies the whole packet $F = (B, V, N, C, Q, M, H, T, P, L)$, because the coordinate functional carrier binds basis coefficients, vector operations, norm budgets, scalar readback, continuity modulus, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{CoordinateFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{CoordinateFunctional}^{\uparrow}$ packet is generated from a SchauderBasis source row, VecSpace operation row, Norm and Banach budget row, coordinate readback row, RegSeqRat scalar-name row, Real-valued continuity-modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CoordinateFunctional}^{\uparrow}}$, seed)

Scope closed: Definition 7494.1, Definition 7494.2, Definition 7494.3, Definition 7494.4, and Definition 7494.5 seed the bounded coordinate-functional route licensed by a displayed Schauder basis.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CoordinateFunctional}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hamel basis choice, arbitrary linear functionals, full dual-space representation, Hahn–Banach, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-readback exactness, linearity obligations, continuity-modulus exposure, transport, replay, provenance, and local naming for the displayed coordinate functional packet.

Chapter 7495

A Concrete Naming Certificate for $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$ packet records the constructive fusion interface between the product-completion certificate of chapter 3797 and the regular Cauchy coordinate projection route of chapter 2076. It reads finite windows from chapter 126, regular rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. The packet names only displayed product-completion rows; it does not quotient streams, select limits, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7495.1 (CauchyProductCompletionFusion carrier)**.** *A* $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, J, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *product-completion packet whose fused route is being consumed.*
2. *J is the* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *coordinate projection packet exposing the left and right regular Cauchy readbacks of Q.*
3. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window family read before either coordinate projection reaches a real-facing consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing both coordinate windows and the product budget.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after S and D are displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the product-completion packet, projection packet, finite windows, dyadic ledger, and regular readback have all been exposed.*
7. *H stores componentwise* hsame *transport for Q, J, S, D, R, E.*
8. *C stores displayed* Cont *replay through the same product-completion and projection route.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$, $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{RegularCauchyProductProjection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionFusion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $Q, J, S, D, R, E, H, C, P, N$. It contains no quotient stream equality, selected completion representative, host product equality, trichotomy, or ambient completeness row.

Theorem 7495.2 (CauchyProductCompletionFusion laws)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$ *carrier factors through* $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{RegularCauchyProductProjection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *in that order, with* hsame, Cont, Pkg, *and local* NameCert *rows preserving the same finite route. The fusion therefore exposes product-completion and regular-Cauchy projection data as one displayed interface without adding quotient streams, selected limits, host product equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionFusionUp.CauchyProductCompletionFusion laws`

Proof. Project the carrier of Definition 7495.1. The rows Q and J are the only product-completion and projection packets available to the consumer, so a Real-facing read cannot start from an external product object.

Read the finite route next. The shared window row S and dyadic ledger D precede the regular readback row R, and the terminal seal E is downstream of those displayed rows. Thus the route has exactly the dependency order claimed in the statement.

Finally inspect H, C, P, N. They transport, replay, source, and name the same finite packet, but introduce no coordinate for quotient equality, selected limits, host product equality, trichotomy, or ambient completeness. □

Theorem 7495.3 (CauchyProductCompletionFusion NameCert obligations)**.** *For an accepted carrier $F = (Q, J, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionFusion}^{\uparrow}}$ *surface consists exactly of carrier admission, product-completion admission for Q, projection-route admission for J, stream-window exposure for S, dyadic ledger compatibility for D, regular readback handoff for R, Real-seal nonescape for E, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is the projection of the finite packet $F = (Q, J, S, D, R, E, H, C, P, N)$. The product-completion and projection obligations are exactly the rows Q and J, so the fused certificate does not recreate their sibling carriers.

The window, ledger, and readback obligations are the rows S, D, R. They force each coordinate read to expose a StreamName window and DyadicRatCore tolerance before the regular rational handoff is available.

The remaining obligations are structural. The seal row E is the only Real-facing export, while H, C, P, N preserve, replay, source, and name the same route. Since the carrier has no other row, the local NameCert surface is exhausted by the listed obligations. □

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionFusionUp.CauchyProductCompletionFusionTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionFusion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$ packet is generated from a CauchyProductCompletion packet, a RegularCauchyProductProjection packet, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductCompletionFusion}^{\uparrow}}$, seed)

Scope closed: Definition 7495.1, Definition 7495.2, and Definition 7495.3 seed the product-completion fusion interface between product completion and regular Cauchy projection.

Formal verification: FormalStatus(NameCert$_{\text{CauchyProductCompletionFusion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host product equality, trichotomy, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-completion admission, projection-route admission, window exposure, dyadic ledger compatibility, regular readback handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionFusionUp.

Chapter 7496

A Concrete Naming Certificate for $\mathsf{DyadicMeshGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicMeshGauge}^{\uparrow}$ packet records the finite dyadic mesh gauges used to schedule regular Cauchy approximations before a real-name seal is read. It sits after the dyadic arithmetic surface of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name surface of chapter 38. The packet names a finite gauge ledger, not a quotient of streams, a selected limit, countable choice, open-cover compactness, or host real equality.

Definition 7496.1 (Dyadic mesh gauge carrier)**.** *A* $\mathsf{DyadicMeshGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, S, W, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the dyadic precision requests and mesh widths inspected by the gauge.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, listing the finite name windows opened by each precision request.*
3. *W is the finite mesh-window ledger pairing dyadic widths with the stream windows that witness them.*
4. *M is the gauge row, recording the displayed bound beyond which the mesh is read at the requested dyadic width.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D, S, W, M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the regular readback row has been displayed.*
7. *H records componentwise* hsame *transport for dyadic tolerances, stream windows, mesh windows, gauge bounds, readbacks, real seals, replay, provenance, and naming.*
8. *C records displayed* Cont *replay from a precision request through the same finite mesh-gauge route.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshGauge}^{\uparrow}}$ *row.*

The classifier compares accepted gauges only through $D, S, W, M, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable-choice schedule, open-cover compactness, Heine–Borel, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicMeshGaugeUp.TasteGate.DyadicMeshGaugeTasteGate single carrier alignment`

Theorem 7496.2 (Dyadic mesh gauge NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshGauge}^{\uparrow}$ *carrier*

$$G = (D, S, W, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMeshGauge}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission through D, stream-window admission through S, finite mesh-window exposure through W, gauge-bound exactness through M,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7496.1. The rows $D, S, W, M, Q, E, H, C, P, N$ are the whole visible packet, so the local naming surface can only list those rows.

Read the approximation rows in order. The dyadic row D supplies the precision request, the stream row S supplies the finite windows, and the mesh ledger W records how those windows are paired with dyadic widths.

The gauge row M then supplies the finite bound used by the regular readback row Q. The terminal seal E is downstream of that readback, while H, C, P, N transport, replay, source, and name the same finite route.

Since no carrier coordinate stores quotient stream equality, selected limits, countable choice, open-cover compactness, Heine–Borel, or ambient real completeness, none of those principles belongs to the local obligation surface. □

Theorem 7496.3 (Dyadic mesh gauge regular-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicMeshGauge}^{\uparrow}$ *packet reads through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicMeshGauge}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using $D, S, W, M, Q, E, H, C, P, N$. The handoff exports finite dyadic mesh-gauge data for regular Cauchy approximation schedules; it exports no host real equality, quotient stream equality, selected limit, countable-choice schedule, open-cover compactness, or ambient completeness.

Proof. Begin with an accepted carrier $G = (D, S, W, M, Q, E, H, C, P, N)$. A regular-Cauchy read must expose the dyadic tolerance row D and the finite stream-window row S before a mesh window or gauge bound is available.

Use the mesh ledger W and gauge row M. These rows bind each displayed dyadic width to the finite windows and the bound used by the approximation schedule, so a consumer cannot replace the route by a selected host limit or an unrecorded choice function.

The readback row Q is then the only $\mathsf{RegSeqRat}^{\uparrow}$ row visible to downstream consumers, and the Real seal E is reached only after Q. Componentwise transport and replay through H, C preserve that same order, while P, N source and name it.

Thus every regular-Cauchy or Real-facing read factors through the finite dyadic mesh-gauge route. The excluded host equality, quotient, choice, compactness, and completeness principles have no row from which to enter. □

Falsifiable prediction: Every accepted $\mathsf{DyadicMeshGauge}^{\uparrow}$ consumer displays $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, finite mesh-window, gauge-bound, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before it reads a terminal real-name approximation.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-sequence-modulus: $\mathsf{DyadicMeshGauge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic mesh widths, finite stream windows, gauge bounds, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshGauge}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, mesh-window ledgers, gauge bounds, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7496.1, Definition 7496.2, and Definition 7496.3 seed the finite dyadic mesh-gauge route for regular-Cauchy and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, Heine–Borel, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, stream-window admission, mesh-window exposure, gauge-bound exactness, RegSeqRat readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMeshGaugeUp.

Chapter 7497

A Concrete Naming Certificate for $\mathsf{SplitInterval}^{\uparrow}$

Following Alexandroff and Urysohn's split-interval construction, the $\mathsf{SplitInterval}^{\uparrow}$ packet records the constructive two-sided split of a displayed real interval as finite NameCert data. It sits next to the double-arrow packet of chapter 7057 and the real endpoint reconstruction of chapter 38. The chapter names the carrier, local NameCert obligations, and order-topology handoff rows only; it does not assert open-cover compactness, quotient topology, selected endpoint cuts, or a host equality principle for adjacent split points.

Definition 7497.1 (SplitInterval carrier)**.** *A* $\mathsf{SplitInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (x, \sigma, B, O, H, C, P, N)$$

with the following displayed rows:

1. *x is the displayed* $\mathsf{Real}^{\uparrow}$ *endpoint currently inspected by the packet.*
2. *σ is a two-sided split bit, naming the lower or upper copy attached to the same endpoint.*
3. *B is a located half-open interval basis row recording the finite left-open and right-open neighborhoods actually exposed at (x, σ).*
4. *O is the lexicographic order-topology row, comparing endpoints through* $\mathsf{Real}^{\uparrow}$ *and resolving equal endpoints through σ.*
5. *H records componentwise* hsame *transport for endpoint, split-bit, basis, order-topology, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from endpoint and split-bit data to the located half-open basis and then to the order-topology row.*
7. *P is the* Pkg *provenance over* $\mathsf{SplitInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DoubleArrowSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SplitInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $x, \sigma, B, O, H, C, P, N$. It has no coordinate for a quotient topology, open-cover compactness, selected endpoint cuts, unrestricted open-set lattice, or host point equality between the two copies.

Lean4 (definition) `BEDC.Derived.SplitIntervalUp`

Theorem 7497.2 (SplitInterval NameCert obligations)**.** *For every accepted* $\mathsf{SplitInterval}^{\uparrow}$ *packet* $S = (x, \sigma, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SplitInterval}^{\uparrow}}$ *surface consists exactly of endpoint admission through* $\mathsf{Real}^{\uparrow}$*, split-bit admission, located half-open basis exposure, lexicographic order-topology exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. A consumer may read a split-interval neighborhood only after those finite rows have been displayed; it cannot replace them by host compactness, quotient topology, selected endpoint cuts, or ambient open-set data.*

Proof. Project the carrier of Definition 7497.1. The displayed rows $x, \sigma, B, O, H, C, P, N$ are the whole accepted packet, so every local read begins with the real endpoint and the two-sided split bit.

The located half-open basis row B supplies only the finite neighborhoods exposed at the inspected split endpoint. The order-topology row O then reads endpoint comparison through $\mathsf{Real}^{\uparrow}$ and tie-breaking through σ, matching the split-interval surface used by chapter 7057.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for compactness, quotient topology, selected cuts, ambient opens, or host equality between the two copies, none of those objects can appear among the local NameCert obligations. □

Theorem 7497.3 (SplitInterval order-topology handoff)**.** *Every order-topology consumer of an accepted* $\mathsf{SplitInterval}^{\uparrow}$ *packet receives generated split-interval data only through the route*

$$\mathsf{Real}^{\uparrow},\ \sigma,\ B,\ O,\ H,\ C,\ P,\ N.$$

The endpoint row supplies the real comparison boundary, the split bit separates the two sides at equal endpoints, the located half-open basis row supplies the finite neighborhood windows, and the lexicographic order-topology row supplies the order-facing handoff. The route exports no open-cover compactness theorem, quotient-topology presentation, selected cut, unrestricted open-set lattice, or host equality principle.

Proof. Read the carrier in the order displayed by Definition 7497.1. The endpoint x and bit σ determine which side of the split interval is inspected, while B lists the located half-open basis rows made available to a topology consumer.

Apply Definition 7497.2. It forces every order-topology read through endpoint admission, split-bit admission, basis exposure, order-topology exposure, transport, replay, provenance, and local naming.

Thus the handoff is a finite route through $\mathsf{Real}^{\uparrow}$, the split bit, located basis data, and the structural NameCert rows. It does not import compactness, quotient topology, selected endpoint cuts, a global open-set lattice, or host equality between the two displayed copies. □

Falsifiable prediction: Every accepted $\mathsf{SplitInterval}^{\uparrow}$ order-topology read factors through the displayed $\mathsf{Real}^{\uparrow}$ endpoint, split-bit row, located half-open basis row, lexicographic order-topology row, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Independence witness: double-arrow-space, real: $\mathsf{SplitInterval}^{\uparrow}$ is not bijective to either sibling because it records a single endpoint with a two-sided split bit and located half-open basis rows, while $\mathsf{DoubleArrowSpace}^{\uparrow}$ records the two-copy interval packet and $\mathsf{Real}^{\uparrow}$ records endpoint data without the split-bit or half-open basis coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{SplitInterval}^{\uparrow}}$.

Constructive story: A SplitInterval$^{\uparrow}$ packet is generated from a RealUp endpoint, a two-sided split bit, located half-open basis rows, a lexicographic order-topology row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SplitInterval}^{\uparrow}}$, seed)

Scope closed: Definition 7497.1, Definition 7497.2, and Definition 7497.3 seed the finite split-interval order-topology packet over RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SplitInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient topology, selected endpoint cuts, unrestricted open-set lattices, Lean theorem verification, or bridge checking for SplitInterval$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, split-bit stability, located half-open basis exposure, lexicographic order-topology exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.SplitIntervalUp.

Chapter 7498

A Concrete Naming Certificate for $\mathsf{StreamInterleaving}^{\uparrow}$

Following Bishop and Bridges, an interleaving of two lawlike streams is useful only when the merge route keeps the two source windows, the parity selector, and the rational readback visible. The $\mathsf{StreamInterleaving}^{\uparrow}$ packet sits over the finite stream-name surface of chapter 126, the regular rational readback route of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal Real seal of chapter 38. It names the finite merge certificate needed by later regular-Cauchy and Real-facing consumers; it does not identify host sequences, choose a hidden subsequence, or quotient Cauchy names.

Definition 7498.1 (Stream interleaving carrier)**.** *A* $\mathsf{StreamInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, O, W, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the even-window* $\mathsf{StreamName}^{\uparrow}$ *row for the first source stream.*
2. *O is the odd-window* $\mathsf{StreamName}^{\uparrow}$ *row for the second source stream.*
3. *W is the parity-window split row, recording which displayed output positions read from E and which read from O.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every displayed source and output window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger used to compare the rational reads at the requested precision.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after E, O, W, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for both stream windows, the parity split, rational readback, dyadic ledger, Real seal, provenance, replay, and local naming.*
8. *C records displayed* Cont *replay from source-window inspection through parity selection, regular rational readback, dyadic comparison, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamInterleaving}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamInterleaving}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host sequence equality, countable choice over positions, quotient Cauchy names, or a Real value that bypasses the displayed parity split.

Definition 7498.2 (Stream interleaving classifier)**.** *The* $\mathsf{StreamInterleaving}^{\uparrow}$ *classifier compares accepted packets only through the rows* $E, O, W, Q, D, R, H, C, P, N$*. Two packets are classifier-same exactly when their even-window reads, odd-window reads, parity split, RegSeqRat readback, DyadicRatCore comparison rows, Real seal, componentwise transport, replay, provenance, and local naming agree through displayed* hsame *routes.*

Theorem 7498.3 (Stream interleaving NameCert obligations)**.** *For every accepted* $\mathsf{StreamInterleaving}^{\uparrow}$ *carrier*

$$I = (E, O, W, Q, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamInterleaving}^{\uparrow}}$ *surface consists exactly of even-source window exposure, odd-source window exposure, parity-window splitting, RegSeqRat readback, DyadicRatCore comparison, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may erase a source stream, change the parity selector, compare through host sequence equality, or export a quotient Cauchy name.*

Proof. Project the carrier of Definition 7498.1. The source data are exactly the two StreamName rows E and O, so a consumer cannot read an output window without naming which source supplied each displayed position.

The parity selector is the row W. It records the finite split of the displayed output positions into even and odd reads, and it is the only route by which the two sources enter one output packet.

The numeric readback rows are Q and D. They expose regular rational reads and dyadic comparisons at the requested finite precision, without replacing the sources by host sequence equality or by a completed quotient object.

Finally, R, H, C, P, N seal, transport, replay, source, and name the same finite route. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 7498.4 (Stream interleaving parity window split)**.** *Every finite output window read from an accepted* $\mathsf{StreamInterleaving}^{\uparrow}$ *packet factors through the displayed split row* W*. The even positions read through* E*, the odd positions read through* O*, and the packet keeps the same* Q, D, H, C, P, N *rows attached to both source reads.*

Proof. Let an output consumer request a finite window. The carrier exposes only the rows $E, O, W, Q, D, R, H, C, P, N$, so the request first meets the parity split row W.

The even part of the requested window is routed to E, and the odd part is routed to O. These are StreamName rows, so the consumer receives finite source-window data rather than a host sequence.

The shared rows Q, D, H, C, P, N then read, compare, transport, replay, source, and name the two branches as one packet. Thus the interleaved window is the displayed parity merge and nothing else. □

Theorem 7498.5 (Stream interleaving finite-stream source window)**.** *In every accepted* $\mathsf{StreamInterleaving}^{\uparrow}$ *packet, the even and odd source windows may be read as* $\mathsf{FiniteStream}^{\uparrow}$ *request-list-window packets before the parity split row* W *is consumed. The finite-source read is the one exposed by chapter 5916, and the parity merge is then the split of Definition 7498.4.*

Proof. Project the carrier as $I = (E, O, W, Q, D, R, H, C, P, N)$. The rows E and O are the two displayed source windows, so a consumer first reads them as finite request/list/window packets rather than as host streams.

The $\mathsf{FiniteStream}^{\uparrow}$ source discipline supplies exactly those finite requests and lists for the two source windows. Thus E and O are available before any output position is assigned by W.

Finally consume the parity split row W. It merges the two finite source windows into the displayed output route while the shared rows Q, D, R, H, C, P, N retain regular rational readback, dyadic comparison, Real sealing, transport, replay, provenance, and local naming. No host sequence equality or countable choice is introduced. □

Theorem 7498.6 (Stream interleaving RegSeqRat readback)**.** *Every regular-Cauchy consumer of an accepted* $\mathsf{StreamInterleaving}^{\uparrow}$ *packet must read*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}$$

through the rows E, O, W, Q, D. The consumer may combine the two source reads only by the displayed parity route, and it may not invoke countable choice over source positions or host-level equality of streams.

Proof. Begin with E, O, W. The parity row determines which source window supplies each requested output position, so a regular-Cauchy read cannot be detached from the two finite StreamName windows.

Read Q next. This row is the RegSeqRat readback surface for the source and output windows already selected by W.

The row D supplies the dyadic arithmetic checks needed by the finite precision request. It compares displayed rational data only, and therefore it cannot introduce countable choice over all later positions or host equality of streams. □

Theorem 7498.7 (Stream interleaving Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{StreamInterleaving}^{\uparrow}$ *packet factors through $E, O, W, Q, D, H, C, P, N$ before reading the seal row R. The Real-facing handoff exports only the interleaved regular rational route and cannot hide a source stream, select a subsequence, or quotient the two Cauchy names.*

Proof. Project the carrier. The Real seal row R is downstream of the source windows, parity split, regular rational readback, and dyadic comparison rows.

A Real-facing consumer must therefore read E, O, W, Q, D first. These rows expose finite stream windows and rational precision data, not host sequence equality or a completed quotient of Cauchy names.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no hidden selector or quotient coordinate is present, the Real seal cannot escape the displayed interleaving route. □

Falsifiable prediction: A $\mathsf{StreamInterleaving}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without displaying $E, O, W, Q, D, H, C, P, N$ refutes the carrier of Definition 7498.1.

Independence witness: Nearest siblings: streamname, regseqrat, dyadicratcore, and real. $\mathsf{StreamInterleaving}^{\uparrow}$ is not bijective to any one of them because it jointly carries two source windows, a parity split, regular rational readback, dyadic comparison, Real sealing, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamInterleaving}^{\uparrow}}$.

Constructive story: A StreamInterleaving$^{\uparrow}$ packet is generated from two StreamName source windows, a parity-window split, RegSeqRat readback, DyadicRatCore comparison, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{StreamInterleaving}\uparrow}$, seed)

Scope closed: Definition 7498.1, Definition 7498.2, Definition 7498.3, Definition 7498.4, Definition 7498.6, and Definition 7498.7 seed the finite interleaving route between StreamName windows, regular rational readback, dyadic comparison, and Real-facing handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{StreamInterleaving}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sequence equality, countable choice over positions, quotient Cauchy equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked parity splitting, RegSeqRat readback, DyadicRatCore comparison, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.StreamInterleavingUp.

Chapter 7499

A Concrete Naming Certificate for PoincareDiskAutomorphismDerivative$^{\uparrow}$

Following the standard Poincare disk automorphism theory, a derivative packet is the finite local witness that keeps the disk automorphism, complex coordinate read, hyperbolic metric preservation, boundary transport, and Real-valued derivative window visible together. The PoincareDiskAutomorphismDeriva packet sits after the automorphism carrier of chapter 6776, the metric surface of chapter 3474, the complex coordinate surface of chapter 39, the boundary transport route of chapter 6267, and the terminal Real seal of chapter 38. It names the finite derivative-window consumer route; it does not import analytic completeness, host holomorphic equality, or a global disk model outside the displayed rows.

Definition 7499.1 (Poincare disk automorphism derivative carrier)**.** *A* PoincareDiskAutomorphismDerivative$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, Z, J, M, B, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* PoincareDiskAutomorphism$^{\uparrow}$ *row naming the disk-preserving automorphism together with its inverse and composition readback.*
2. *Z is the* Complex$^{\uparrow}$ *coordinate-window row for the source and target disk points used by the derivative read.*
3. *J is the derivative-window row, recording the finite complex-linear or real-scale read attached to the displayed automorphism window.*
4. *M is the* HyperbolicMetric$^{\uparrow}$ *preservation row comparing the source and target metric reads through the same window.*
5. *B is the* PoincareDiskBoundaryTransport$^{\uparrow}$ *row recording the induced boundary transport available to boundary-facing consumers.*
6. *R is the* Real$^{\uparrow}$ *seal row for the finite derivative norm or scale read, available only after A, Z, J, M, B have been displayed.*
7. *H records componentwise* hsame *transport for automorphism data, complex coordinate windows, derivative readback, metric preservation, boundary transport, Real sealing, provenance, replay, and local naming.*

8. *C records displayed* Cont *replay from automorphism inspection through coordinate-window readback, derivative-window readback, metric preservation, boundary transport, and Real sealing.*
9. *P is the* Pkg *provenance over* PoincareDiskAutomorphismDerivative$^{\uparrow}$, PoincareDiskAutomorphism$^{\uparrow}$, Complex$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskBoundaryTransport$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{PoincareDiskAutomorphismDerivative}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host holomorphic equality, a completed tangent bundle, ambient analytic continuation, detached boundary dynamics, or a Real derivative value that bypasses the displayed disk automorphism.

Definition 7499.2 (Poincare disk automorphism derivative classifier)**.** *The* PoincareDiskAutomorphismDerivative$^{\uparrow}$ *classifier compares accepted packets only through $A, Z, J, M, B, R, H, C, P, N$. Classifier sameness requires displayed agreement of the automorphism row, complex coordinate window, derivative-window readback, hyperbolic-metric preservation row, boundary-transport row, Real seal, transport, replay, provenance, and local naming.*

Theorem 7499.3 (Poincare disk automorphism derivative NameCert obligations)**.** *For every accepted* PoincareDiskAutomorphismDerivative$^{\uparrow}$ *carrier*

$$D = (A, Z, J, M, B, R, H, C, P, N),$$

the local NameCert$_{\text{PoincareDiskAutomorphismDerivative}^{\uparrow}}$ *surface consists exactly of automorphism exposure, complex coordinate-window readback, derivative-window readback, hyperbolic metric preservation, boundary transport, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may detach the derivative from the automorphism row, replace metric preservation by host analytic equality, or export a completed tangent object.*

Proof. Project the carrier of Definition 7499.1. The automorphism row A is the only source of disk-map data, and the complex coordinate row Z is the only source of coordinate-window data for the derivative read.

The derivative row J is read with the metric row M and the boundary row B. This keeps local derivative information tied to hyperbolic metric preservation and to the displayed boundary transport route.

The Real seal R is downstream of these rows, while H, C, P, N transport, replay, source, and name the same packet. Since no completed tangent bundle, host holomorphic equality, or detached analytic continuation appears in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 7499.4 (Poincare disk automorphism derivative metric preservation)**.** *Every metric-facing consumer of an accepted* PoincareDiskAutomorphismDerivative$^{\uparrow}$ *packet must factor through*

$$\text{PoincareDiskAutomorphism}^{\uparrow} \longmapsto \text{Complex}^{\uparrow} \longmapsto \text{HyperbolicMetric}^{\uparrow} \longmapsto \text{Real}^{\uparrow}.$$

The derivative-window read may be consumed only as a displayed local row attached to the same automorphism and metric-preservation data.

Proof. Start with A and Z. The automorphism and coordinate-window rows supply the finite disk-map and coordinate data, so the metric consumer cannot begin from a host analytic function.

Read the derivative window J together with the metric preservation row M. The derivative read is meaningful for this packet only because it is attached to the same finite automorphism route whose hyperbolic metric comparison is displayed.

Finally, R, H, C, P, N seal, transport, replay, source, and name the result. Thus the Real-facing metric value remains inside the finite automorphism and metric-preservation route. □

Theorem 7499.5 (Poincare disk automorphism derivative boundary transport)**.** *Every boundary-facing consumer of an accepted* $\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}$ *packet factors through the boundary row B after reading A, Z, J, M. The consumer may cite the derivative window as evidence for local transport behavior, but it cannot construct detached boundary dynamics or a completed ideal-boundary action.*

Proof. The boundary consumer first reads the disk automorphism and coordinate rows A, Z. Without these rows, there is no displayed source for the boundary transport.

The derivative and metric rows J, M are read next. They record the local scale and metric-preservation window attached to the same disk map.

Only then does the consumer read B. The row B is a finite boundary-transport certificate, and H, C, P, N keep that certificate under transport, replay, provenance, and local naming. No row supplies detached boundary dynamics or a completed boundary action. □

Theorem 7499.6 (Poincare disk automorphism derivative Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}$ *packet must read $A, Z, J, M, B, H, C, P, N$ before reading R. The Real seal names only the displayed derivative scale or norm reached through the disk automorphism, complex coordinate window, hyperbolic metric preservation, and boundary transport route.*

Proof. Project the carrier. The Real seal R is not a free scalar; it appears after the automorphism, coordinate, derivative, metric, and boundary rows.

A Real-facing consumer must therefore read A, Z, J, M, B first. These rows display the finite disk-map and derivative route rather than host analytic equality or a completed tangent object.

The structural rows H, C, P, N preserve the same route. Since the carrier has no coordinate for a detached Real derivative value, the seal cannot escape the finite derivative-window packet. □

Falsifiable prediction: A $\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}$ consumer that exports a Real derivative read without displaying $A, Z, J, M, B, H, C, P, N$ refutes the carrier of Definition 7499.1.

Independence witness: Nearest siblings: poincare-disk-automorphism, hyperbolic-metric, complex, poincare-disk-boundary-transport, and real. $\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}$ is not bijective to any one of them because it jointly carries disk automorphism data, complex coordinate windows, derivative readback, metric preservation, boundary transport, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}$ packet is generated from a PoincareDiskAutomorphism row, Complex coordinate windows, derivative-window readback, HyperbolicMetric preservation, PoincareDiskBoundaryTransport, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7499.1, Definition 7499.2, Definition 7499.3, Definition 7499.4, Definition 7499.5, and Definition 7499.6 seed the finite derivative-window route between disk automorphism, coordinate readback, metric preservation, boundary transport, and Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskAutomorphismDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host holomorphic equality, analytic continuation, a completed tangent bundle, detached boundary dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked automorphism exposure, coordinate-window readback, derivative readback, metric preservation, boundary transport, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDiskAutomorphismDerivativeUp.

Chapter 7500

A Concrete Naming Certificate for MetaCICResidualJoinGauge$^{\uparrow}$

The MetaCICResidualJoinGauge$^{\uparrow}$ packet realizes a bounded residual-join gauge for the MetaCIC confluence frontier of section 9386.3. It sits beside the residual-substitution boundary of chapter 2181, the confluence critical-pair boundary of chapter 2212, the parallel-diamond frontier of chapter 4238, the closed type-checker seal of chapter 2182, and the closed-normal confluence fragment of chapter 737. The packet records a finite residual-pair and bounded local-join surface without claiming residual-substitution compatibility, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 7500.1 (MetaCIC residual-join gauge carrier)**.** *A* MetaCICResidualJoinGauge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, R, K, L, B, S, F, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row exposing the finite MetaCIC source, target, and context window inspected by the gauge.*
2. *R is the residual-pair row, recording the two residual contraction paths whose local join is being tested.*
3. *K is the critical-pair boundary row, keeping the inspected peak inside the confluence critical-pair surface.*
4. *L is the bounded local-join row, listing the candidate common reducts and the finite search budget used to inspect them.*
5. *B is the bounded checker row for the local join.*
6. *S is the residual-substitution boundary row, retained as a premise-facing obstruction rather than discharged by the packet.*
7. *F is the closed type-checker seal and closed-normal fragment row used to keep the inspected terms inside the sealed fragment.*
8. *O is the obstruction-retention row for residual substitution, full parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, and kernel consistency.*
9. *H records componentwise* hsame *transport for T, R, K, L, B, S, F, O and for replay, provenance, and naming.*

10. *C records displayed* Cont *replay through the same residual-pair and bounded local-join route.*
11. *P is the* Pkg *provenance over* MetaCICResidualJoinGauge$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{MetaCICResidualJoinGauge}^{\uparrow}}$ *row.*

The classifier compares accepted gauges only through $T, R, K, L, B, S, F, O, H, C, P, N$. It has no coordinate for a premise-free residual-substitution theorem, full closed confluence, evaluator equality, quotient syntax equality, subject reduction, or kernel consistency.

Theorem 7500.2 (MetaCIC residual-join gauge NameCert obligations)**.** *For every accepted* MetaCICResidualJoinG *carrier*

$$J = (T, R, K, L, B, S, F, O, H, C, P, N),$$

the local NameCert$_{\text{MetaCICResidualJoinGauge}^{\uparrow}}$ *surface consists exactly of typed-window admission through T, residual-pair exposure through R, critical-pair boundary retention through K, bounded local-join exposure through L, bounded checker routing through B, residual-substitution boundary retention through S, closed-checker and closed-normal fragment sealing through F, obstruction retention through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7500.1. The confluence-facing rows are T, R, K, L, B, S, F, O, and the structural certificate rows are H, C, P, N.

Read the residual rows first. The typed-window row T scopes the inspected source and target data, while R and K expose the residual pair and its critical-pair boundary before any local-join read is available.

Read the bounded join rows next. The local-join row L and checker row B provide only a finite inspection surface, and the row F keeps the route inside the closed type-checker seal and closed-normal fragment.

Finally S, O, H, C, P, N retain the residual-substitution premise, refused endpoints, transport, replay, provenance, and local naming. Since no carrier coordinate discharges residual substitution or proves full confluence, the listed rows exhaust the local NameCert obligations. □

Theorem 7500.3 (MetaCIC residual-join gauge diamond handoff)**.** *Every confluence consumer of an accepted* MetaCICResidualJoinGauge$^{\uparrow}$ *packet must read the residual pair R, bounded local-join row L, checker row B, residual-substitution boundary S, closed type-checker and closed-normal fragment row F, obstruction row O, and structural rows H, C, P, N before using the packet in a conditional diamond route. The handoff supplies a finite bounded local-join gauge; it does not claim residual-substitution compatibility, full parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Start from the obligation theorem Definition 7500.2. It forces every confluence-facing read through the typed-window, residual-pair, critical-pair, bounded local-join, checker, residual-substitution, sealed-fragment, and obstruction rows.

The residual pair and critical-pair rows scope the peak. A consumer cannot read a local join before R and K identify the finite residual pair under inspection.

The bounded local-join and checker rows then expose only a finite gauge. The closed type-checker seal and closed-normal fragment row F keep the inspected route inside the sealed fragment, while the retained boundary row S prevents the handoff from being retyped as residual-substitution compatibility.

Finally O, H, C, P, N preserve the refused endpoints and the structural certificate. Hence the packet may feed a later conditional diamond argument, but it does not itself close the full confluence or typing endpoints. □

Falsifiable prediction: A $\mathsf{MetaCICResidualJoinGauge}^{\uparrow}$ consumer that exports a diamond handoff without residual-pair, bounded local-join, bounded checker, residual-substitution boundary, closed type-checker seal, closed-normal fragment, obstruction, transport, replay, provenance, and local naming rows refutes the carrier of Definition 7500.1.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, metacic-parallel-diamond-frontier, closed-type-checker-seal, metacicconfluencenormal-fragment: the residual-substitution boundary retains the missing compatibility premise without the bounded local-join gauge; the critical-pair boundary records peaks without the residual-pair gauge; the parallel-diamond frontier records the broader conditional handoff without this bounded local-join packet; the closed type-checker seal records typing admission without residual joins; and the closed-normal fragment records normal-source joins without the residual-pair gauge. None is bijective to $\mathsf{MetaCICResidualJoinGauge}^{\uparrow}$.

Lean TasteGate.lean must later provide Nontrivial and FieldFaithful instances for $\mathsf{MetaCICResidualJoinGauge}^{\uparrow}$ so the typed-window, residual-pair, critical-pair, bounded local-join, checker, residual-substitution boundary, sealed-fragment, obstruction, transport, replay, provenance, and local naming fields remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualJoinGauge}^{\uparrow}$ packet is generated from typed-window, residual-pair, critical-pair boundary, bounded local-join, bounded checker, residual-substitution boundary, closed checker and closed-normal fragment, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7500.1, Definition 7500.2, and Definition 7500.3 seed the finite residual-pair and bounded local-join gauge for conditional MetaCIC diamond consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, full parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window admission, residual-pair exposure, critical-pair boundary retention, bounded local-join exposure, bounded checker routing, residual-substitution boundary retention, sealed-fragment routing, obstruction retention, transport, replay, provenance, Nontrivial, FieldFaithful, and local naming for BEDC.Derived.MetaCICResidualJoinGaugeUp.

Chapter 7501

A Concrete Naming Certificate for $\mathsf{CantorScheme}^{\uparrow}$

Following Bishop and Bridges, a Cantor scheme is the finite binary tree surface used when compact metric and Cantor-space arguments must expose their branch choices without importing countable choice or a hidden limit selector. The $\mathsf{CantorScheme}^{\uparrow}$ packet sits between the Cantor-space cylinder route of chapter 1475, the finite observation rows of chapter 126, the nested interval discipline of chapter 320, and the compact support route of chapter 129. It also shares the regular rational readback surface of chapter 301 when a compact-metric consumer asks for radius or diameter data along a displayed branch.

Definition 7501.1 (Cantor scheme carrier)**.** *A* $\mathsf{CantorScheme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, B, W, R, K, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite binary tree row, listing the nodes, parent links, left and right successor rows, and the level of each displayed node.*
2. *B is the branch-prefix row, recording the finite bit string or cylinder prefix currently read by the consumer.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that says which branch-prefix coordinates have been inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the scheme carries dyadic radii, diameters, or compact-metric comparison data.*
5. *K is the compact support row, read through* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, *or* $\mathsf{CompactMetric}^{\uparrow}$ *according to the consumer-facing route.*
6. *H records componentwise* hsame *transport for tree nodes, branch prefixes, finite windows, readback rows, compact support rows, provenance, replay, and local naming.*
7. *C records displayed* Cont *replay from tree-node requests to branch-prefix reads, from branch-prefix reads to stream windows, and from those windows to regular readback and compact support.*
8. *P is the* Pkg *provenance over* $\mathsf{CantorScheme}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CantorScheme}^\uparrow}$ *row.*

The classifier compares accepted schemes only through $T, B, W, R, K, H, C, P, N$. It has no coordinate for a selected infinite branch, quotient stream equality, detached path choice, open-cover compactness, countable choice, or ambient completion.

Theorem 7501.2 (Cantor scheme NameCert obligations)**.** *For every accepted* $\mathsf{CantorScheme}^\uparrow$ *carrier $S = (T, B, W, R, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorScheme}^\uparrow}$ *surface consists exactly of finite binary tree exposure, branch-prefix exposure, finite* $\mathsf{StreamName}^\uparrow$ *window inspection,* $\mathsf{RegSeqRat}^\uparrow$ *readback, compact support through* $\mathsf{CantorSpace}^\uparrow$*,* $\mathsf{NestedInterval}^\uparrow$*, or* $\mathsf{CompactMetric}^\uparrow$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read a compact cylinder, nested interval prefix, or compact-metric support row only after these displayed rows have been exposed.*

Proof. Project the carrier of Definition 7501.1. The finite tree row T and branch-prefix row B are the only source rows for the scheme.

The observation route is W followed by R. A consumer can inspect only the displayed branch-prefix coordinates and the corresponding regular rational readback data.

The compact support row K is downstream of the same prefix and window rows. It may route to Cantor-space cylinders, nested interval prefixes, or compact-metric comparison data, but it cannot add a selected infinite branch or quotient stream equality.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate appears in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 7501.3 (Cantor scheme branch nonescape)**.** *Every accepted branch read of a* $\mathsf{CantorScheme}^\uparrow$ *packet factors through the displayed route*

$$\mathsf{CantorSpace}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{NestedInterval}^\uparrow \longmapsto \mathsf{CompactMetric}^\uparrow.$$

The read may extend a finite prefix or refine a compact support row, but it must keep the finite tree row, branch-prefix row, inspected StreamName windows, regular readback, compact support, transport, replay, provenance, and local naming visible. It cannot escape to an unlogged infinite path, quotient stream equality, detached path choice, open-cover compactness, countable choice, or ambient completion.

Proof. Begin with the finite tree and branch-prefix rows. They expose a finite cylinder boundary, not a completed infinite path.

Read the finite observation rows next. The $\mathsf{StreamName}^\uparrow$ window row determines which prefix coordinates are inspected, and the $\mathsf{RegSeqRat}^\uparrow$ row records any diameter, radius, or comparison data attached to that inspection.

The compact support row then routes the same prefix through the relevant Cantor-space, nested-interval, or compact-metric sibling. This row refines displayed finite data; it does not introduce a hidden branch selector.

The structural rows H, C, P, N preserve the route under transport, replay, provenance, and local naming. Therefore every branch read remains inside the displayed scheme packet. □

Falsifiable prediction: Every accepted $\mathsf{CantorScheme}^\uparrow$ branch read exposes finite tree, branch-prefix, StreamName window, RegSeqRat readback, compact support, transport, replay, provenance, and NameCert rows before any compact-cylinder or nested-interval consumer reads it.

Independence witness: Nearest siblings: cantorspace, streamname, nestedinterval, compact-metric, and regseqrat. $\mathsf{CantorScheme}^\uparrow$ is not bijective to any one sibling because it jointly carries

finite binary tree structure, branch-prefix inspection, finite observation windows, regular readback, compact support, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorScheme}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorScheme}^{\uparrow}$ packet is generated from finite binary tree rows, branch-prefix rows, StreamName windows, RegSeqRat readback, compact support through CantorSpace, NestedInterval, or CompactMetric rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorScheme}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7501.1, Definition 7501.2, and Definition 7501.3 seed the finite branching-scheme carrier consumed by Cantor-space, nested-interval, and compact-metric reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorScheme}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected infinite branch, quotient stream equality, detached path choice, open-cover compactness, countable choice, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-tree exposure, branch-prefix stability, StreamName window inspection, RegSeqRat readback, compact support nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorSchemeUp.

Chapter 7502

A Concrete Naming Certificate for $\mathsf{LipschitzSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LipschitzSpace}^{\uparrow}$ packet records the finite BEDC surface for a metric-domain real-valued map whose variation is controlled by a displayed rational Lipschitz constant. It sits over the metric source of chapter 57, the terminal real-name seal of chapter 38, the map graph carrier of chapter 130, the tolerance-only modulus consumer of chapter 369, and the Lipschitz-function packet of chapter 3762. The packet names a finite function-space row only; it does not add quotient equality of functions, selected representatives, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7502.1 (LipschitzSpace carrier)**.** *A* $\mathsf{LipschitzSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, K, I, T, C, P, N)$$

with the following displayed rows:

1. *M is the metric-domain row, exposing only finite* $\mathsf{Met}^{\uparrow}$ *distance inspections for the source points read by the packet.*
2. *F is the real-valued map row, routed through* $\mathsf{ContinuousMap}^{\uparrow}$ *graph data before any modulus or inequality ledger is exported.*
3. *K is the positive rational Lipschitz-constant row.*
4. *I is the metric inequality ledger, recording only displayed finite comparisons of the form $d_{\mathsf{Real}^{\uparrow}}(F(x), F(y)) \leq K d_M(x, y)$ at inspected source windows.*
5. *T records componentwise* hsame *transport for the metric row, map row, rational constant, inequality ledger, replay, provenance, and local naming row.*
6. *C records displayed* Cont *replay from metric inspection and graph readback through the inequality ledger.*
7. *P is the* Pkg *provenance over* $\mathsf{LipschitzSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{LipschitzFunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, F, K, I, T, C, P, N. It has no coordinate for quotient equality of functions, selected representatives, open-cover compactness, countable choice, measure-theoretic regularity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7502.2 (LipschitzSpace NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzSpace}^{\uparrow}$ *carrier* $S = (M, F, K, I, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-domain row, real-valued continuous-map graph row, rational Lipschitz-constant row, finite metric inequality ledger, componentwise transport, displayed continuation replay, package provenance, and local naming. Every consumer must read the displayed metric and graph windows before the inequality ledger is available.*

Proof. Project the accepted packet. Its analytic rows are exactly M, F, K, I, followed by the structural rows T, C, P, N.

The metric-domain row supplies finite source-distance inspections, and the map row supplies only the displayed real-valued graph data. The rational row K is read before the inequality ledger I, so every displayed bound is tied to the same finite source and target windows.

Rows T, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for quotient function equality, selected representatives, open-cover compactness, countable choice, measure-theoretic regularity, or ambient Real completeness, none of those principles can be projected from the local NameCert surface. □

Lean4 (statement) `BEDC.Derived.LipschitzSpaceUp`

Theorem 7502.3 (LipschitzSpace uniform-modulus handoff)**.** *Every accepted* $\mathsf{LipschitzSpace}^{\uparrow}$ *carrier exposes a* $\mathsf{UniformModulus}^{\uparrow}$ *handoff only through the displayed route*

$$\mathsf{Met}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{LipschitzFunction}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The handoff may consume the rational constant row K *and the finite inequality ledger* I*, but it cannot export a host-selected modulus, quotient equality of functions, open-cover compactness, countable choice, measure-theoretic regularity, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with $S = (M, F, K, I, T, C, P, N)$. The only rows that can feed a modulus consumer are the metric-domain row M, the real-valued map row F, the rational constant row K, and the inequality ledger I.

The sibling $\mathsf{LipschitzFunction}^{\uparrow}$ surface reads those rows as finite Lipschitz data. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives only the tolerance-facing consequence recorded by that route.

Transport and replay through T and C preserve the same displayed rows, while P and N only record provenance and naming. Therefore the handoff is a finite BEDC route, not a selected external modulus or an ambient compactness or completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7502.1, Definition 7502.2, and Definition 7502.3 seed the finite Lipschitz function-space carrier over Metric, Real, ContinuousMap, UniformModulus, LipschitzFunction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of functions, selected representatives, open-cover compactness, countable choice, measure-theoretic regularity, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-domain readback, real-valued graph exposure, rational Lipschitz-constant exactness, inequality-ledger nonescape, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzSpaceUp.

Constructive story: A $\mathsf{LipschitzSpace}^{\uparrow}$ packet is generated from a metric-domain row, real-valued map row, rational Lipschitz constant row, finite metric inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7503

A Concrete Naming Certificate for $\mathsf{DyadicShadowCone}^{\uparrow}$

The $\mathsf{DyadicShadowCone}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion reading of chapter 9463 as finite dyadic radius windows. It sits after the boundary-shadow carrier of chapter 3813, the observer-expansion route of chapter 4167, the visual metric surface of chapter 3866, the finite observation rows of chapter 126, the regular rational readback rows of chapter 301, and the terminal Real seal of chapter 38. It names the dyadic cone of visible boundary shadows around an observer-side read; it does not complete the boundary, select a global observer horizon, or import host hyperbolic geometry.

Definition 7503.1 (Dyadic shadow cone carrier)**.** *A* $\mathsf{DyadicShadowCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, S, O, V, W, R, E, T, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the dyadic-radius row, listing the finite radius budget and the center of the observer-side window.*
2. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row exposing the boundary shadow read at that radius.*
3. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the observer-expansion source ledger.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row comparing the boundary shadow through a finite visual metric read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which finite observer and boundary coordinates have been inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the dyadic radius and visual metric data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the finite radius or visual metric value.*
8. *T is the transport row keeping the dyadic radius, boundary shadow, observer-expansion, visual metric, stream-window, readback, and Real-seal rows in one displayed route.*
9. *L is the replay ledger that records the accepted read as a cone row rather than as an isolated shadow or isolated expansion row.*
10. *H records componentwise* hsame *transport for the radius, shadow, observer, visual metric, window, readback, seal, transport, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from dyadic-radius requests through boundary-shadow reads, observer-expansion rows, visual-metric comparison, finite windows, regular readback, Real sealing, and transport.*
12. *P is the* Pkg *provenance over* $\mathsf{DyadicShadowCone}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicShadowCone}^{\uparrow}}$ *row.*

The classifier compares accepted cones only through $Q, S, O, V, W, R, E, T, L, H, C, P, N$. It has no coordinate for a completed hyperbolic boundary, selected observer horizon, quotient boundary equality, detached visual metric scalar, global disk model, or host geometric compactness.

Theorem 7503.2 (Dyadic shadow cone NameCert obligations)**.** *For every accepted* $\mathsf{DyadicShadowCone}^{\uparrow}$ *carrier*

$$D = (Q, S, O, V, W, R, E, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicShadowCone}^{\uparrow}}$ *surface consists exactly of dyadic-radius exposure, boundary-shadow exposure, observer-expansion exposure, visual-metric comparison, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, route transport, cone replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may replace the dyadic radius window by a completed boundary point, detach the visual metric from the boundary-shadow route, or read observer expansion without the same finite cone packet.*

Proof. Project the carrier of Definition 7503.1. The dyadic radius row Q, boundary-shadow row S, observer-expansion row O, and visual metric row V are simultaneous source rows for the cone.

Read the finite observation route next. The stream window W fixes which observer and boundary coordinates are inspected, while R and E provide the regular rational and Real-facing values reached by that inspection.

The transport and replay rows T, L keep the read as one cone route. They prevent a consumer from taking an isolated shadow, an isolated expansion row, or a detached visual scalar as the whole carrier.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no completed boundary, selected horizon, quotient equality, detached visual metric scalar, global disk model, or host compactness coordinate appears in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 7503.3 (Dyadic shadow cone observer-window nonescape)**.** *Every accepted observer-expansion read of a* $\mathsf{DyadicShadowCone}^{\uparrow}$ *packet factors through*

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longmapsto \mathsf{HyperbolicObserverExpansion}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto$$

The read may narrow the dyadic radius or refine the finite boundary window, but it must keep the radius, boundary-shadow, observer-expansion, visual-metric, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and NameCert rows visible. It cannot escape to a completed boundary object, selected horizon, quotient boundary equality, detached visual scalar, global disk model, or host hyperbolic compactness.

Proof. Begin with the dyadic radius and boundary-shadow rows. They expose a finite cone window, not a completed boundary point.

Read the observer-expansion and visual-metric rows next. The observer row records the source ledger being expanded, and the visual metric row compares it only through the displayed boundary-shadow route.

The StreamName and RegSeqRat rows then expose the finite observation schedule and the regular rational readback. The Real seal is downstream of those rows and therefore cannot be consumed as an independent geometric scalar.

Transport, replay, provenance, and local naming preserve the same packet through T, L, H, C, P, N. Thus every observer-window read remains inside the displayed dyadic shadow cone. □

The Lean TasteGate file at `lean4/BEDC/Derived/DyadicShadowConeUp/TasteGate.lean` must include explicit `Nontrivial DyadicShadowConeUp` and `FieldFaithful DyadicShadowConeUp` instances preserving $Q, S, O, V, W, R, E, T, L, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can witness the gate.

Falsifiable prediction: Every accepted $\mathsf{DyadicShadowCone}^{\uparrow}$ read exposes dyadic-radius, boundary-shadow, observer-expansion, visual-metric, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and NameCert rows before any boundary-shadow or observer-expansion consumer reads it.

Independence witness: Nearest siblings: hyperbolic-boundary-shadow, hyperbolic-observer-expansion, hyperbolic-visual-metric, streamname, and regseqrat. $\mathsf{DyadicShadowCone}^{\uparrow}$ is not bijective to any one sibling because it jointly carries dyadic radius windows, boundary shadows, observer-expansion rows, visual-metric reads, finite observations, regular readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicShadowCone}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicShadowCone}^{\uparrow}$ packet is generated from dyadic-radius rows, boundary-shadow rows, observer-expansion rows, visual-metric comparison, StreamName windows, RegSeqRat readback, a Real seal, route transport, cone replay, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicShadowCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7503.1, Definition 7503.2, and Definition 7503.3 seed the finite dyadic shadow-cone route for boundary-shadow and observer-expansion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicShadowCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary compactification, selected observer horizons, quotient boundary equality, detached visual metric scalars, global disk models, host hyperbolic compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic-radius exposure, boundary-shadow exposure, observer-expansion exposure, visual-metric comparison, StreamName window inspection, RegSeqRat readback, Real sealing, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.DyadicShadowConeUp.

Chapter 7504

A Concrete Naming Certificate for $\mathsf{DyadicRealDensity}^{\uparrow}$

Following Bishop and Bridges-style constructive real analysis, the $\mathsf{DyadicRealDensity}^{\uparrow}$ packet records the finite route by which dyadic rational approximants become visible to a displayed $\mathsf{Real}^{\uparrow}$ name through regular rational readback, finite stream windows, dyadic tolerance data, transport, replay, provenance, and a local naming row. It sits after the dyadic arithmetic surface of chapter 291, the finite-window naming route of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a density handoff for Real and Completion consumers; it does not choose a representative stream, quotient real names, invoke trichotomy, use countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7504.1 (DyadicRealDensity carrier)**.** *A* $\mathsf{DyadicRealDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (D, W, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row carrying the dyadic rational approximant and its arithmetic certificate.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule at which the Real name is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after the dyadic source, window, and readback rows have been displayed.*
5. *T is the dyadic tolerance row, read from the same dyadic ledger as D, that bounds the approximation request.*
6. *H records componentwise* hsame *transport for D, W, R, E, T and the structural rows.*
7. *C records displayed* Cont *replay from a Real approximation request through D, W, R, E, T, H.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRealDensity}^{\uparrow}}$ *row.*

The carrier stores only displayed finite rows. It has no coordinate for selected representatives, quotient equality of real names, trichotomy, countable choice, host least-upper-bound data, or ambient Real completeness.

Theorem 7504.2 (DyadicRealDensity NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRealDensity}^{\uparrow}$ *carrier*

$$Z = (D, W, R, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRealDensity}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the dyadic source, finite window, regular readback, Real seal, dyadic tolerance, transport, replay, provenance, and local naming rows;*
2. *tolerance compatibility, requiring the tolerance row* T *to be read from the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger as the source row* D*;*
3. *window-readback exactness, requiring every Real approximation request to inspect the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *before the Real seal* E *is consumed;*
4. *transport stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve the route* D, W, R, E, T*;*
5. *ledger non-escape, stating that* P *and* N *export no selected representative stream, quotient real equality, trichotomy, countable-choice selector, host least-upper-bound object, or ambient Real completeness principle.*

Proof. Project the carrier of Definition 7504.1. The rows $D, W, R, E, T, H, C, P, N$ are exactly the available coordinates, so carrier admission is finite.

The tolerance clause reads D and T together. Since both rows come from the dyadic ledger, an approximation request cannot import a separate host radius or a selected rational outside the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ source.

The Real-facing clause then reads W, R, E. A consumer reaches the Real seal only after the finite windows and regular rational readback have been displayed, so the approximation route remains a finite read rather than a quotient stream judgement.

Finally H, C, P, N transport, replay, source, and name the same rows. None of the refused host-side objects is a coordinate of Z, and therefore the listed obligations exhaust the local NameCert surface. □

Theorem 7504.3 (DyadicRealDensity Real approximation)**.** *Every Real-facing approximation exported by an accepted* $\mathsf{DyadicRealDensity}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

with its dyadic tolerance, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The exported approximation cannot inspect selected limits, quotient equality of real names, countable choice, trichotomy, host least-upper-bound data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $Z = (D, W, R, E, T, H, C, P, N)$. The dyadic source row D and tolerance row T provide the only approximation scale visible to the packet.

The finite observation then passes through W and R. The StreamName row opens only the displayed windows, and the RegSeqRat row supplies the regular rational readback attached to those windows.

The Real seal E is downstream of that finite route. Componentwise transport and replay preserve the same coordinates through H and C, while P and N record provenance and local naming.

Thus every Real-facing approximation is a read of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in the displayed order. Since no other coordinate exists in the carrier, the refused host-side principles are unavailable to the approximation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRealDensity}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRealDensity}^{\uparrow}$ packet is generated from a DyadicRatCore source row, finite StreamName windows, a RegSeqRat readback row, a RealUp seal row, a dyadic tolerance row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRealDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7504.1, Definition 7504.2, and Definition 7504.3 seed the finite density route from dyadic rational approximants to Real-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRealDensity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected representatives, quotient equality of real names, trichotomy, countable choice, host least-upper-bound data, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance compatibility, window-readback exactness, transport stability, and ledger non-escape for BEDC.Derived.DyadicRealDensityUp.

Chapter 7505

A Concrete Naming Certificate for $\mathsf{SupportFunction}^{\uparrow}$

Following Schneider on convex bodies, the $\mathsf{SupportFunction}^{\uparrow}$ packet records the finite NameCert surface for support-function rows over $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{ConvexSet}^{\uparrow}$. It sits over the scalar seal of chapter 38, the vector carrier of chapter 47, and the convex membership boundary of chapter 208. The packet names support values only through displayed direction rows, convex witness rows, real comparison rows, and finite supremum budgets; it does not import host compactness, a selected maximizing point, quotient dual spaces, or ambient real completeness.

Definition 7505.1 (Support-function carrier)**.** *A* $\mathsf{SupportFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, V, D, E, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the convex body row, read through* $\mathsf{ConvexSet}^{\uparrow}$*, carrying membership, convex-combination, and located boundary tests for the inspected points.*
2. *V is the ambient vector-space row, read through* $\mathsf{VecSpace}^{\uparrow}$*, carrying vector addition, scalar action, and finite pairing coordinates used by the packet.*
3. *D is the direction row, listing the finitely many inspected linear-functional or dual-direction reads.*
4. *E is the evaluation row, whose entries pair displayed directions from D with displayed convex witnesses from K and seal the values through* $\mathsf{Real}^{\uparrow}$*.*
5. *B is the support-budget row, listing finite upper-bound comparisons and the inspected witnesses that justify each support value read.*
6. *H records componentwise* hsame *transport for K, V, D, E, B and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from vector and convex-set reads to evaluations and from evaluations to budgeted real comparisons.*
8. *P is the* Pkg *provenance over* $\mathsf{SupportFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SupportFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, D, E, B, H, C, P, N$. It has no coordinate for a host supremum operator, a chosen maximizer, quotient dual equality, compactness of K, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SupportFunctionUp.SupportFunctionTasteGate single carrier alignment`

Theorem 7505.2 (Support-function finite supremum budget)**.** *Every support value read from an accepted* $\mathsf{SupportFunction}^{\uparrow}$ *packet factors through the displayed route*

$$K, V, D, E \longmapsto B$$

before any downstream consumer may inspect a real bound. The row B contains the only supremum-facing data: finite inspected witnesses, upper-bound comparisons sealed through $\mathsf{Real}^{\uparrow}$*, and the replay route tying those comparisons to the convex and vector rows. No support read may be interpreted as a host supremum over all points of K, a selected exposed point, an ambient compactness theorem, or a quotient dual-space value.*

Proof. Project the carrier of Definition 7505.1. The only rows that can mention support values are the direction row D, the evaluation row E, and the support-budget row B, all attached to the convex and vector rows K, V.

The evaluation route is finite. Each inspected direction is paired with displayed convex witnesses, and every resulting comparison is sealed through the $\mathsf{Real}^{\uparrow}$ dependency rather than by an external supremum operation.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since the carrier contains no maximizer selector, host compactness row, quotient dual equality, or ambient completeness field, the finite budget B exhausts the support-function supremum surface. □

Theorem 7505.3 (Support-function NameCert obligations)**.** *For every accepted* $\mathsf{SupportFunction}^{\uparrow}$ *carrier*

$$S = (K, V, D, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SupportFunction}^{\uparrow}}$ *surface consists exactly of convex-set admission for K, vector-space admission for V, direction exposure for D, real evaluation sealing for E, finite support-budget exposure for B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. These obligations are sufficient to hand a finite support-bound surface to later convex-duality and compact-convex consumers, but they do not assert classical support-function reconstruction or compact-convex duality.*

Proof. Start from Definition 7505.1. The carrier lists exactly the convex row, vector row, direction row, evaluation row, support-budget row, transport row, replay row, provenance row, and local naming row.

Apply Definition 7505.2. It forces every support-facing read through K, V, D, E, B before a real bound can be consumed, so the local NameCert obligations include budget exposure and real sealing but no host supremum.

The structural rows H, C, P, N preserve the same displayed packet under transport, replay, provenance, and naming. Because no coordinate records a selected maximizer, quotient dual equality, compactness theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the accepted support-function surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SupportFunction}^{\uparrow}}$**.**

Constructive story: A SupportFunction$^{\uparrow}$ packet is generated from convex-set rows, vector-space rows, displayed direction rows, real evaluation rows, finite support-budget comparisons, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SupportFunction}^{\uparrow}}$, seed)

Scope closed: Definition 7505.1, Definition 7505.2, and Definition 7505.3 seed the finite support-function NameCert surface over RealUp, VecSpaceUp, and ConvexSetUp.

Formal verification: FormalStatus(NameCert$_{\mathsf{SupportFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host support-function theorem, compactness of convex bodies, selected maximizers, quotient dual spaces, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, direction exposure, finite evaluation sealing, support-budget nonescape, transport, replay, provenance, and local naming for $K, V, D, E, B, H, C, P, N$.

Chapter 7506

A Concrete Naming Certificate for $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$

Following Clarkson's uniformly convex space theory and the Bishop–Bridges practice of keeping metric and real-name evidence explicit, the $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ packet records a closed fiber inside a uniformly convex carrier as finite NameCert data. It sits over the modulus route of chapter 5831, the complete-metric boundary of chapter 128, and the terminal real-name discipline of chapter 38. The packet gives projection, fixed-point, and completion consumers a named closed-fiber convexity surface without importing quotient subspaces, selected nearest points, weak compactness, or choice.

Definition 7506.1 (Closed fiber uniform convexity carrier)**.** *A* $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, K, \Phi, \Gamma, \Delta, \Pi, E, H, T, P, N)$$

with the following displayed rows:

1. *U is the ambient* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *row, exposing the norm, metric, midpoint, and positive modulus routes only through its local certificate.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row used by Cauchy and fixed-point consumers of the fiber.*
3. *Φ is the fiber membership classifier, read as a displayed predicate on the ambient carrier rather than as a quotient subspace.*
4. *Γ is the closedness row: every finite Cauchy read in the fiber that lands through K must replay Φ at the displayed limit row.*
5. *Δ is the uniform-convexity restriction row, reusing the ambient Clarkson modulus from U for pairs whose membership reads pass through Φ.*
6. *Π is the projection-route row, recording how a consumer may ask for a fiber-facing projection bound through the displayed metric, real, midpoint, and closedness rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for norm gaps, radius comparisons, modulus margins, and projection-distance readback.*
8. *H records componentwise* hsame *transport for $U, K, \Phi, \Gamma, \Delta, \Pi, E$ and the local structural rows.*
9. *T records displayed* Cont *replay from a closed-fiber request through the ambient uniform-convexity, complete-metric, membership, closedness, projection, and real-seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$, $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedFiberUniformConvexity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for quotient fibers, selected nearest points, weak compactness, Hahn–Banach separation, or an ambient choice principle.

Theorem 7506.2 (Closed fiber uniform convexity NameCert obligations)**.** *For every accepted* $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ *packet* $F = (U, K, \Phi, \Gamma, \Delta, \Pi, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClosedFiberUniformConvexity}}$ *surface consists exactly of ambient uniformly convex admission, complete-metric boundary visibility, fiber-membership readback, closedness replay for finite Cauchy reads, restricted modulus admission, projection-route exposure, real-seal comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $U, K, \Phi, \Gamma, \Delta, \Pi, E, H, T, P, N$.

Proof. Carrier admission is projection from Definition 7506.1. The rows $U, K, \Phi, \Gamma, \Delta, \Pi, E, H, T, P, N$ are the whole accepted packet.

Read the ambient and closedness rows first. The row U supplies only the certified uniform-convexity route, while K, Φ, Γ bind completion-facing Cauchy reads to the displayed fiber predicate.

Read the convexity and projection rows next. The row Δ restricts the ambient modulus to pairs whose membership is displayed by Φ, and Π exposes only the metric, midpoint, closedness, and real-comparison route already present in U, K, Γ, E.

Finally H, T, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores quotient fibers, selected nearest points, weak compactness, Hahn–Banach separation, or choice, those objects cannot be obligations of the local certificate. □

Theorem 7506.3 (Closed fiber uniform convexity projection route)**.** *Every accepted* $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ *consumer that asks for a fiber-facing projection or distance estimate factors through*

$$\mathsf{UniformlyConvexSpace}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \Phi, \Gamma, \Delta, \Pi, H, T, P, N.$$

The route may read the restricted Clarkson modulus only after the ambient uniformly convex row, the fiber-membership row, the complete-metric closedness row, and the real-valued comparison seal have been displayed. It exports a finite closed-fiber NameCert surface, not a selected nearest point, quotient fiber equality, weak compactness, Hahn–Banach separation, or choice.

Proof. Begin with a projection or distance request. The ambient row U supplies the only norm, metric, midpoint, and modulus data, and the real seal E supplies the only visible comparison values.

Pass through the fiber rows before reading the modulus. The predicate row Φ displays membership, and the closedness row Γ ensures that completion-facing Cauchy reads landing through K replay the same predicate at the limit row.

Only then may Δ and Π answer the request. The restricted modulus is tied to pairs already admitted by Φ, and the projection route is tied to the displayed metric, midpoint, closedness, and real-comparison data.

Finally H, T, P, N keep the route invariant under transport, continuation replay, provenance, and local naming. The excluded selected nearest point, quotient-fiber equality, weak compactness, Hahn–Banach separation, and choice objects are absent from the carrier, so the finite route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ read is refuted if a consumer obtains a restricted uniform-convexity margin or fiber-facing projection bound without first displaying the ambient $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, fiber-membership, closedness, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: uniformly-convex-space, completemetric, real: no listed sibling is bijective to $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ because the packet jointly carries an ambient uniformly convex route, a complete-metric boundary, fiber membership, closedness replay, restricted modulus access, projection-route exposure, real comparisons, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedFiberUniformConvexity}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedFiberUniformConvexity}^{\uparrow}$ packet is generated from an ambient UniformlyConvexSpace row, a CompleteMetric boundary row, fiber membership, closedness replay, restricted modulus access, projection-route exposure, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedFiberUniformConvexity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7506.1, Definition 7506.2, and Definition 7506.3 seed the finite closed-fiber route over uniformly convex, complete-metric, and real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedFiberUniformConvexity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient fiber equality, selected nearest points, weak compactness, Hahn–Banach separation, choice, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, closedness replay, restricted modulus exactness, projection-route nonescape, and local naming for BEDC.Derived.ClosedFiberUniformConvexityUp.

Chapter 7507

RegularCauchyDistributiveLaw NameCert Construction

From Bishop and Bridges constructive real arithmetic, the $\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}$ packet records the finite window calculation that lets multiplication distribute over addition for regular rational Cauchy names. It consumes the two-source surfaces of $\mathsf{RegularCauchySum}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$, the difference route of $\mathsf{RegularCauchyDifference}^{\uparrow}$, the algebraic operation inventory of $\mathsf{RegularCauchyRing}^{\uparrow}$, and the source and seal layers $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ from chapter 417, chapter 461, chapter 429, chapter 1470, chapter 301, chapter 291, and chapter 38. The packet names only the finite distributive-window proof surface; it does not import host real algebra or equality of completed real quotients.

Definition 7507.1 (Regular Cauchy distributive-law carrier)**.** *A* $\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, B, C, W_A, W_B, W_C, S_L, P_L, P_B, P_C, S_R, \Delta, E, R, H, T, Q, N)$$

with the following displayed coordinates:

1. *A, B, and C are three* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for regular rational Cauchy real names.*
2. *W_A, W_B, and W_C are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened at the same requested precision scale.*
3. *S_L is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row for the source sum $B + C$.*
4. *P_L is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *row for the left route $A \cdot (B + C)$.*
5. *P_B and P_C are the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *rows for the two right-route products $A \cdot B$ and $A \cdot C$.*
6. *S_R is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row for the right route $(A \cdot B) + (A \cdot C)$.*
7. *Δ is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *comparison row between the left-route readback and the right-route readback.*
8. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the source endpoint rows, product endpoint rows, sum endpoint rows, and the finite distributive error budget for the synchronized window.*
9. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the left route, right route, difference comparison, and tolerance ledger have been displayed.*

10. *H stores componentwise* hsame *transport for the three source rows, synchronized windows, sum rows, product rows, difference row, dyadic tolerance ledger, terminal seal, continuation, provenance, and local naming.*

11. *T stores displayed* Cont *replay from the three source windows to the left product route, the two right product routes, the right sum route, the comparison route, and the terminal real-facing route.*

12. *Q is the* Pkg *provenance over* $\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{RegularCauchyRing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient streams, host real equality, trichotomy, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7507.2 (Regular Cauchy distributive-law NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDistributi}}$ *surface for Definition 7507.1 consists exactly of carrier admission, synchronized source-window admission, left-route product admission, right-route product-and-sum admission, difference-row comparison, dyadic tolerance-ledger exactness, terminal real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7507.1. The three source rows, synchronized windows, operation rows, difference row, tolerance ledger, transports, continuations, provenance, and local name are all visible coordinates of D.

The left route uses the sum row S_L and then the product row P_L. Apply Definition 417.2 to expose $B + C$, then apply Definition 461.2 to expose $A \cdot (B + C)$ through the same synchronized window scale.

The right route uses P_B, P_C, and S_R. Apply the product obligation surface twice to expose $A \cdot B$ and $A \cdot C$, and then apply the sum obligation surface to expose their displayed sum. The difference row Δ is then read through the comparison surface of Definition 429.2.

The tolerance and structural rows are packet coordinates. The ledger E records the dyadic endpoint and error-budget rows used by both routes, while H, T, Q, N transport, replay, source, and name the same finite calculation. Since no other coordinate is present, the obligation surface excludes host real algebra, quotient equality, trichotomy, selected limits, and ambient completeness. □

Theorem 7507.3 (Regular Cauchy distributive-window induction)**.** *For every accepted* RegularCauchyDistributiveL *carrier, the equality of the two distributive routes is witnessed by induction on the displayed synchronized window ledger. At each finite precision window, the endpoint calculation reads only the three* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *rows, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *rows, the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *comparison row, the* $\mathsf{RegularCauchyRing}^{\uparrow}$ *operation inventory, and the structural* hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Begin with the empty window ledger. No source endpoint has been read, so both routes expose only the trivial displayed tolerance row in E and no real-facing equality claim is available.

For the induction step, open the next synchronized windows W_A, W_B, W_C. The sum row S_L computes the endpoint for $B + C$, and the product row P_L computes the left-route endpoint from A and that displayed sum.

On the same window, the rows P_B and P_C compute the two product endpoints $A \cdot B$ and $A \cdot C$. The row S_R then sums those two displayed endpoints. The dyadic arithmetic entries in E record the finite distributive identity at this precision and the accumulated error budget for both routes.

The difference row Δ consumes the two route readbacks and records that their displayed tolerance gap is absorbed by the same ledger. Componentwise hsame transport and displayed Cont replay preserve the induction state, so the finite calculation remains inside the named BEDC packet at every step. □

Theorem 7507.4 (Regular Cauchy distributive-law Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}$ *packet reaches the terminal distributive handoff only after reading the three source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the left route* $A \cdot (B + C)$*, the right route* $(A \cdot B) + (A \cdot C)$*, the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *comparison row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The handoff exports no host real equality, quotient stream, trichotomy, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 7507.1. The only real-facing route is the one assembled from the left product readback, the two right product readbacks, their sum, and the difference comparison row.

Apply Definition 7507.2. It forces every consumer to keep the source windows, operation rows, tolerance ledger, transport, replay, provenance, and local naming attached to the route.

Apply Definition 7507.3. The route comparison is established at finite synchronized windows and is carried by the displayed dyadic ledger before any $\mathsf{Real}^{\uparrow}$ seal is read.

Finally use the Real-seal boundaries of the sibling operation chapters and the operation inventory of Definition 1470.9. They expose only regular-Cauchy readbacks and terminal seals. Since the distributive-law carrier adds no further real-facing coordinate, the consumer cannot replace the packet by host equality, quotient streams, trichotomy, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}$ packet is generated from three RegSeqRat source rows, synchronized StreamName windows, RegularCauchySum and RegularCauchyProduct routes for the left and right distributive expressions, a RegularCauchyDifference comparison row, a DyadicRatCore tolerance ledger, componentwise transport, continuation replay, package provenance, terminal RealUp seal access, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}}$, seed)

Scope closed: Definition 7507.1, Definition 7507.2, Definition 7507.3, and Definition 7507.4 seed the finite distributive-window surface consumed by regular-Cauchy real arithmetic.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyDistributiveLaw}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host real equality, quotient streams, trichotomy, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window admission, dyadic tolerance-ledger exactness, distributive-window induction, and Real-seal nonescape for the same finite packet.

Chapter 7508

A Concrete Naming Certificate for $\mathsf{PixleyRoySpace}^{\uparrow}$

Following Pixley and Roy on finite-set hyperspaces, the $\mathsf{PixleyRoySpace}^{\uparrow}$ packet records the finite topology surface whose points are nonempty finite closed-subset rows and whose basic opens are hit windows of the form "the displayed finite row is kept inside a displayed open neighbourhood while meeting the named finite core." It sits next to the indexed-open carrier of chapter 88, the observable-open classifier of chapter 1860, the membership discipline of chapter 3668, the finite-set surface of chapter 112, and the lower-limit topology example chapter 3363. The packet names only the finite hit-and-containment interface; it does not import a host powerset hyperspace, selected compact subsets, Vietoris completeness, countable choice, or ambient separation theorems.

Definition 7508.1 (Pixley–Roy space carrier)**.** *A* $\mathsf{PixleyRoySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, S, F, O, W, H, C, Q, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *indexed-open row for the source space.*
2. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row used to read membership in displayed opens.*
3. *F is the nonempty finite closed-subset row, exposed through* $\mathsf{FinSet}^{\uparrow}$ *and* $\mathsf{SetLike}^{\uparrow}$ *membership data rather than through a host powerset.*
4. *O is the finite basic-open hit window. It displays a finite core row and an ambient open row, and it records that the finite closed-subset row lies inside the open row while hitting every named core coordinate.*
5. *W is the finite observation schedule for the hit and containment reads.*
6. *H records componentwise* hsame *transport for the topology, observable-open, finite closed-subset, hit-window, schedule, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an observable-open request through T, S, F, O, W.*
8. *Q is the* Pkg *provenance over* $\mathsf{PixleyRoySpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{SierpinskiSpace}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{SorgenfreyLine}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PixleyRoySpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary hyperspace powerset, a selected compact-subset representative, countable choice, host equality of closed sets, Vietoris completeness, or an ambient separation theorem.

Lean4 ✓ `BEDC.Derived.PixleyRoySpaceUp.PixleyRoySpaceTasteGate single carrier alignment`

Theorem 7508.2 (Pixley–Roy space NameCert obligations)**.** *For every accepted* $\mathsf{PixleyRoySpace}^{\uparrow}$ *carrier* $P = (T, S, F, O, W, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PixleyRoySpace}^{\uparrow}}$ *surface consists exactly of ambient topology admission, Sierpinski observable-open admission, finite nonempty closed-subset admission through* $\mathsf{FinSet}^{\uparrow}$ *and* $\mathsf{SetLike}^{\uparrow}$*, basic-open hit-window exactness, finite observation scheduling, componentwise transport, displayed replay, provenance, and local naming. Every public neighbourhood read must display* T, S, F, O, W *before the structural rows* H, C, Q, N *are consumed.*

Proof. Project the accepted carrier of Definition 7508.1. The mathematical rows are exactly the ambient topology row T, the observable-open truth row S, the finite closed-subset row F, the hit-window row O, and the finite observation schedule W.

The finite-subset admission is local. The row F is read through the displayed $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{SetLike}^{\uparrow}$ membership surfaces, so nonemptiness, membership, and closed-subset evidence are visible finite data rather than a host powerset element.

The basic-open obligation is also local. The row O records the finite core and ambient open row, while S supplies observable-open truth for the containment and hit reads scheduled by W.

Finally H, C, Q, N transport, replay, source, and name the same rows. Since none of the excluded objects is a coordinate of the carrier, the local NameCert surface is exhausted by the listed finite obligations. □

Theorem 7508.3 (Pixley–Roy finite-open handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$*, compactness-boundary, or lower-limit-topology consumer of an accepted* $\mathsf{PixleyRoySpace}^{\uparrow}$ *packet receives a neighbourhood only through the finite route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{SierpinskiSpace}^{\uparrow} \longrightarrow \mathsf{FinSet}^{\uparrow}, \mathsf{SetLike}^{\uparrow} \longrightarrow O \longrightarrow W.$$

The route exposes finite nonempty closed-subset rows and finite basic-open hit windows before any downstream read is public; it exports no host powerset of closed sets, selected compact-subset representative, countable choice, Vietoris completeness, or ambient separation theorem.

Proof. Start with the NameCert obligations of Definition 7508.2. Any neighbourhood read must first open the ambient indexed topology row and then read observable-open truth through the Sierpinski row.

The finite closed-subset row is then inspected through $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{SetLike}^{\uparrow}$. Only after those membership rows are displayed can the hit-window row O record that the named finite core is hit and that the displayed finite row stays inside the ambient open.

The schedule W lists the finite observations used by the handoff, while H, C, Q, N preserve the same route under transport, replay, provenance, and local naming. Thus a consumer such as the lower-limit topology surface of chapter 3363 receives a finite-open Pixley–Roy packet, not a hidden hyperspace or an external compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PixleyRoySpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PixleyRoySpace}^{\uparrow}$ packet is generated from a Topology indexed-open row, a SierpinskiSpace observable-open row, a finite nonempty closed-subset row, a finite basic-open hit window, a finite observation schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PixleyRoySpace}^{\uparrow}}$, seed)

Scope closed: Definition 7508.1, Definition 7508.2, and Definition 7508.3 seed the finite Pixley–Roy hyperspace interface over Topology, SierpinskiSpace, FinSet, SetLike, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PixleyRoySpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host powerset hyperspace, selected compact-subset representatives, countable choice, host equality of closed sets, Vietoris completeness, ambient separation theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked ambient topology admission, observable-open admission, finite closed-subset admission, basic-open hit-window exactness, schedule exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.PixleyRoySpaceUp.

Chapter 7509

A Concrete Naming Certificate for RegulatedFunctionFiniteJump$^{\uparrow}$

Following Bishop and Bridges on regulated functions, the RegulatedFunctionFiniteJump$^{\uparrow}$ packet records the finite-jump subclass as a bounded local NameCert surface rather than as host-level function equality. It sits beside the regulated-function carrier of chapter 1906, the step-function endpoint rows of chapter 3780, the uniform-limit route of chapter 463, and the dyadic mesh-gauge schedule of chapter 7496. The packet names only finite source intervals, jump partitions, one-sided windows, step approximants, uniform-limit seals, transport, replay, provenance, and local naming rows.

Definition 7509.1 (Finite-jump regulated-function carrier)**.** *A* RegulatedFunctionFiniteJump$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, J, W, A, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the source interval row, naming the displayed closed interval, endpoint order, and observation windows read by the regulated-function consumer.*
2. *J is the finite jump partition row, listing the finitely many jump sites and the adjacent cells on which one-sided data is inspected.*
3. *W is the one-sided window ledger, recording the left and right window rows at each jump site together with ordinary interior windows away from J.*
4. *A is the step approximant ledger, read through the* StepFunction$^{\uparrow}$ *sibling surface and indexed by the finite partition and requested tolerance.*
5. *U is the* UniformLimit$^{\uparrow}$ *seal row, consuming the step approximant ledger and the dyadic mesh-gauge schedule before any regulated-function or integration-facing consumer reads the packet.*
6. *H records componentwise* hsame *transport for intervals, finite jump partitions, one-sided windows, step approximants, uniform-limit seals, replay, provenance, and naming.*
7. *C records displayed* Cont *replay from a source-window request through the partition, one-sided window, step approximant, mesh-gauge, and uniform-limit rows.*
8. *P is the* Pkg *provenance over* RegulatedFunctionFiniteJump$^{\uparrow}$, RegulatedFunction$^{\uparrow}$, StepFunction$^{\uparrow}$, UniformLimit$^{\uparrow}$, DyadicMeshGauge$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $I, J, W, A, U, H, C, P, N$. It has no coordinate for arbitrary discontinuity classification, Lebesgue differentiation, quotient equality of host functions, countable choice, selected point values, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegulatedFunctionFiniteJumpUp.RegulatedFunctionFiniteJumpTasteGate single carrier alignment`

Theorem 7509.2 (Finite-jump regulated-function NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}$ *carrier $F = (I, J, W, A, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}}$ *surface consists exactly of carrier admission, finite jump-row stability, the step approximation route through* $\mathsf{StepFunction}^{\uparrow}$*, the uniform-limit consumer route through* $\mathsf{UniformLimit}^{\uparrow}$ *and* $\mathsf{DyadicMeshGauge}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger non-escape from undisplayed analytic principles.*

Proof. Project the carrier of Definition 7509.1. The rows $I, J, W, A, U, H, C, P, N$ are the whole visible packet.

The finite jump-row stability clause is read from J and W. A consumer may inspect only the displayed finite partition, the adjacent one-sided windows at each listed jump, and the ordinary interior windows away from the listed jump sites.

The approximation route is read from A. Each approximation passes through the sibling $\mathsf{StepFunction}^{\uparrow}$ pattern, while C replays the same source-window request through the selected step row.

The limiting route is read from U. It consumes the displayed approximant ledger together with the dyadic mesh-gauge schedule before exposing a $\mathsf{RegulatedFunction}^{\uparrow}$ or integration-facing handoff.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier row stores arbitrary discontinuity classification, Lebesgue differentiation, host function equality, selected point values, countable choice, or ambient completeness, none of those endpoints belongs to the local obligation surface. □

Theorem 7509.3 (Finite-jump regulated-function UniformLimit consumer)**.** *An accepted* RegulatedFunctionFiniteJ *packet supplies a finite-window consumer route for* $\mathsf{UniformLimit}^{\uparrow}$ *and later regulated-integration chapters:*

$$I, J, W, A \longmapsto U$$

with transport, replay, provenance, and local naming carried by H, C, P, N. The route exposes finite jump partitions, one-sided windows, step approximant rows, and the dyadic mesh-gauge handoff, but it exports no host function equality, arbitrary discontinuity classifier, Lebesgue differentiation theorem, selected countable family, or quotient function-space endpoint.

Proof. Begin with the accepted carrier. A source request first reads the interval row I and the finite jump partition row J, so every later window is attached to a displayed cell or a displayed one-sided jump window.

Use the window ledger W and the step approximant ledger A. The one-sided windows and ordinary interior windows determine which step approximant rows may be inspected, and those rows use the finite-data pattern supplied by $\mathsf{StepFunction}^{\uparrow}$.

Next read the uniform-limit seal U. The seal consumes the step approximant ledger and the dyadic mesh-gauge schedule before a $\mathsf{UniformLimit}^{\uparrow}$ or regulated-integration consumer may use the packet.

Finally apply the structural rows H, C, P, N. They preserve the same finite-window route under transport, replay, provenance, and naming, so no host function equality, countable choice, arbitrary discontinuity classification, Lebesgue differentiation, or quotient function-space endpoint is exported. □

Falsifiable prediction: Every accepted $\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}$ consumer displays a source interval, finite jump partition, one-sided window ledger, step approximant ledger, $\mathsf{UniformLimit}^{\uparrow}$ seal, $\mathsf{DyadicMeshGauge}^{\uparrow}$ schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any regulated-function or integration-facing handoff is read.

Independence witness: regulated-function, step-function, uniformlimit, dyadic-mesh-gauge: $\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite jump partition, one-sided window ledger, step approximant route, uniform-limit seal, dyadic mesh-gauge schedule, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}$ packet is generated from a source interval, finite jump partition, one-sided window rows, step approximant rows, a UniformLimit seal, a DyadicMeshGauge schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7509.1, Definition 7509.2, and Definition 7509.3 seed the finite-jump regulated-function interface over $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, and $\mathsf{DyadicMeshGauge}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedFunctionFiniteJump}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary discontinuity classification, Lebesgue differentiation, host-level function extensionality, quotient function-space equality, countable choice, selected point values, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite jump-row stability, step approximation routing, UniformLimit consumer routing, DyadicMeshGauge scheduling, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.RegulatedFunctionFiniteJumpUp.

Chapter 7510

A Concrete Naming Certificate for $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ packet records the finite Archimedean completion route by which rational regular names are exposed as a located Real-facing completion object. It sits after the dyadic arithmetic ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located completion packet of chapter 2231. The packet names an Archimedean located-completion interface; it is not quotient stream equality, a selected limit representative, countable choice, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

It also consumes the effective completion route of chapter 5930 through visible $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and terminal $\mathsf{Real}^{\uparrow}$ seal data. The Archimedean row therefore attaches to the same L10 completion lattice before its located-completion tests are read.

Definition 7510.1 (BishopArchimedeanCompletion carrier)**.** *A* $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Archimedean tolerance ledger, carrying dyadic scales, rational slack rows, and comparison budgets.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected rational-name windows at the dyadic scales displayed by D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row obtained from the inspected windows and the dyadic tolerance ledger.*
4. *L is the located-completion row, aligned with* $\mathsf{BishopLocatedCompletion}^{\uparrow}$ *and recording the displayed locatedness tests consumed before the terminal seal is visible.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R, L have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic, stream-window, regular-readback, located-completion, Real-seal, replay, provenance, and naming rows.*
7. *C records the displayed* Cont *replay from an Archimedean completion request through D, S, R, L, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopLocatedCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopArchimedeanCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, L, E, H, C, P, N$. It has no coordinate for quotient representatives, hidden limit selection, countable choice, trichotomy, host equality of reals, or an unmediated completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 7510.2 (BishopArchimedeanCompletion NameCert obligations)**.** *For every accepted* $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ *carrier $A = (D, S, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopArchimedeanCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic Archimedean tolerance exposure, finite stream-window exposure, regular rational readback, located-completion handoff, Real-seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier coordinates. The rows D, S, R, L, E display the finite dyadic tolerance data, the inspected name windows, the regular Cauchy readback, the located-completion tests, and the terminal Real-facing seal.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. They do not add a quotient representative, a selected limit, or an ambient completeness witness.

Since the carrier has no coordinate outside $D, S, R, L, E, H, C, P, N$, every accepted local NameCert obligation is one of the listed rows, and every listed row is explicitly present in the accepted packet. □

Theorem 7510.3 (BishopArchimedeanCompletion regular Cauchy handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopLocatedCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the $\mathsf{Real}^{\uparrow}$ *seal is not public until the dyadic tolerance ledger, finite stream windows, regular rational readback, and located-completion row have all been displayed.*

Proof. Begin with a Real-facing consumer of A. Carrier admission gives access to the terminal seal only as the row E in the same finite packet as D, S, R, L.

Replay the continuation row C. The request first reads the Archimedean dyadic scales through D, opens the finite $\mathsf{StreamName}^{\uparrow}$ windows S, obtains the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback R, and then consumes the located-completion row L.

Only after that route is displayed can E be read. Componentwise transport and package provenance preserve the same order, so no host limit operation can bypass the regular Cauchy handoff. □

Theorem 7510.4 (BishopArchimedeanCompletion no quotient nonescape)**.** *No accepted* BishopArchimedeanCompl *packet exports quotient stream equality, a selected representative of a Cauchy class, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any such datum is outside the local carrier unless it is re-expressed through the displayed dyadic, stream-window, regular-readback, located-completion, Real-seal, transport, replay, provenance, and naming rows.*

Proof. Inspect the classifier of Definition 7510.1. It compares accepted packets only through the nine displayed rows $D, S, R, L, E, H, C, P, N$.

The excluded data are not among those coordinates. The handoff theorem also forces every Real-facing read through the dyadic, finite-window, regular-readback, and located-completion route before the terminal seal is visible.

Therefore an export using quotient equality, selected representatives, countable choice, trichotomy, host equality, or ambient completeness has left the accepted $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ surface rather than escaping from it. □

Falsifiable prediction: Every accepted BishopArchimedeanCompletionUp Real-facing read factors through DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat regular readback, BishopLocatedCompletion locatedness tests, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, bishop-located-completion, real: $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic Archimedean tolerances, finite windows, regular readback, located-completion tests, Real sealing, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopArchimedeanCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopArchimedeanCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore Archimedean tolerance ledger, StreamName finite windows, RegSeqRat regular Cauchy readback, BishopLocatedCompletion locatedness tests, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopArchimedeanCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7510.1, Definition 7510.2, Definition 7510.3, and Definition 7510.4 seed the finite Archimedean located-completion route through dyadic tolerance, finite stream-window, regular-readback, located-completion, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopArchimedeanCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Archimedean dyadic tolerance exposure, finite-window readback, regular Cauchy handoff, located-completion tests, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopArchimedeanCompletionUp.

Chapter 7511

A Concrete Naming Certificate for $\mathsf{RegularCauchyInverseBoundary}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyInverseBoundary}^{\uparrow}$ packet records the constructive boundary at which a regular Cauchy real name may be read through a reciprocal row. It sits next to the algebra inventory of chapter 1470, the finite product readback of chapter 461, and the terminal real-name seal of chapter 38. The packet does not import host real inversion; it keeps the apartness precondition, reciprocal budget, product-unit check, and seal handoff as displayed NameCert rows.

Definition 7511.1 (Regular Cauchy inverse-boundary carrier)**.** *A* $\mathsf{RegularCauchyInverseBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, W, D, Q, R, M, U, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row for the denominator regular Cauchy name.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row on which the denominator is read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the endpoint, lower-bound, reciprocal-budget, and product-tolerance entries for the same window.*
4. *Q is the apartness precondition row, read through the positive-gap surfaces used by* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$
5. *R is the* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *readback row exposed only after Q, W, D have been displayed.*
6. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *handoff row for multiplying the source row by the reciprocal readback.*
7. *U is the product-unit ledger row saying that the M route reaches the displayed unit readback at the requested window and budget.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the reciprocal readback and product-unit handoff are public.*
9. *H records componentwise* hsame *transport for the source, window, dyadic ledger, apartness row, reciprocal row, product row, unit ledger, seal row, continuation, provenance, and local name.*
10. *C records displayed* Cont *replay for opening the source window, reading the apartness lower bound, invoking the reciprocal row, checking the product-unit ledger, and exposing the terminal seal.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyInverseBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseBoundary}^{\uparrow}}$ *row.*

The carrier is a reciprocal-boundary packet. It stores no total inverse on $\mathsf{Real}^{\uparrow}$*, no quotient real equality, no selected stream representative, no trichotomy, no countable-choice row, and no ambient completeness principle.*

Theorem 7511.2 (Regular Cauchy inverse-boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInverse}}$ *surface for Definition 7511.1 consists exactly of the following finite obligation rows:*

1. *apartness precondition, requiring the positive-gap row Q to be displayed before the reciprocal branch is readable;*
2. *reciprocal readback, requiring R to consume only the source row A, window row W, dyadic ledger D, and apartness evidence Q;*
3. *product-unit handoff, requiring M and U to witness the finite route from A and R to the displayed unit readback through* $\mathsf{RegularCauchyProduct}^{\uparrow}$*;*
4. *componentwise* hsame *transport, requiring H to move $A, W, D, Q, R, M, U, E, C, P, N$ together;*
5. *Real-seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer of E to read the apartness, reciprocal, product-unit, transport, replay, provenance, and naming rows first.*

These rows are the whole local inverse-boundary surface.

Proof. Carrier admission is projection from Definition 7511.1. The source, window, dyadic, apartness, reciprocal, product, unit, seal, transport, replay, provenance, and naming rows are explicit coordinates.

The apartness and reciprocal obligations are read in order. The positive-gap row Q is the only eligibility coordinate, and the reciprocal row R is listed as downstream of Q, W, D. Thus a consumer cannot read a reciprocal branch from the source row alone.

The product-unit handoff is the displayed algebra check. The row M routes the product through the existing $\mathsf{RegularCauchyProduct}^{\uparrow}$ surface, while U records the finite unit readback at the same window and budget. This is a packet-level check, not host real multiplication.

Finally the transport and seal obligations keep the route closed. The rows H, C, P, N preserve and replay the same displayed data, and E is reached only after the reciprocal and unit rows are public. Therefore the packet exports a partial inverse boundary without total real inversion, quotient equality, selected representatives, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7511.3 (Regular Cauchy inverse-boundary seal discipline)**.** *Every accepted* RegularCauchyInverseBoundar *carrier exposes its terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$A, W, D, Q \longmapsto R \longmapsto M, U \longmapsto E.$$

The route is stable under the stored componentwise hsame *and displayed* Cont *replay rows, and it exports no total inverse operation on all real names.*

Proof. Project an accepted carrier $I = (A, W, D, Q, R, M, U, E, H, C, P, N)$. The only real-facing coordinate is E, and its listed dependencies are the apartness row, reciprocal readback, product row, and product-unit ledger.

Use the NameCert obligations. Apartness precondition and reciprocal readback force the route through Q and R; product-unit handoff forces the route through M and U; componentwise transport and replay force later consumers to keep H and C attached.

Since the carrier has no coordinate for a total inverse, quotient representative, trichotomy decision, choice principle, or ambient completeness theorem, the terminal seal is exactly a partial inverse-boundary seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyInverseBoundary}^{\uparrow}$ packet is generated from a RegSeqRat source row, a StreamName window, a DyadicRatCore reciprocal ledger, a positive apartness precondition, a RegularCauchyReciprocal readback, a RegularCauchyProduct product-unit handoff, a terminal RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInverseBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7511.1, Definition 7511.2, and Definition 7511.3 seed the finite reciprocal-boundary surface consumed before field-like Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInverseBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close total inversion on RealUp, quotient real equality, trichotomy, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness eligibility, reciprocal readback, product-unit exactness, transport stability, and Real-seal nonescape for BEDC.Derived.RegularCauchyInverseBoundaryUp.

Chapter 7512

A Concrete Naming Certificate for $\mathsf{DarbouxIntegrable}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxIntegrable}^{\uparrow}$ packet separates integrability evidence from the $\mathsf{DarbouxIntegral}^{\uparrow}$ value packet. It sits between the Darboux criterion of chapter 5712, the finite lower and upper sum rows of chapter 5858 and chapter 4083, the finite Riemann–Darboux criterion of chapter 7383, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the value-level Darboux integral route of chapter 1460. The chapter records a finite predicate-level carrier for integrability only; it does not select an integral value, quotient integrands by host equality, import measure completion, select a partition by choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7512.1 (Darboux integrable carrier)**.** *A* $\mathsf{DarbouxIntegrable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, P, L, U, G, M, Q, R, H, C, V, N)$$

with the following rows:

1. *F is the bounded function row exposed only through finite interval and value observations.*
2. *P is the finite partition row used by the Darboux criterion.*
3. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row over F and P.*
4. *U is the* ***??*** *row over the same F and P.*
5. *G is the oscillation-gap row comparing U and L at the displayed tolerance request.*
6. *M is the tolerance-modulus row that chooses later partition requests only through finite data.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the gap estimates.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row available after the regular rational readback.*
9. *H is componentwise* hsame *transport for F, P, L, U, G, M, Q, R.*
10. *C is displayed* Cont *replay for the same route.*
11. *V is the* Pkg *provenance over* $\mathsf{DarbouxIntegrable}^{\uparrow}$, $\mathsf{DarbouxCriterion}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, ***??***, $\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegrable}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for a selected integral value, host integrand equality, measure completion, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7512.2 (Darboux integrable NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxIntegrable}^{\uparrow}$ *carrier* $I = (F, P, L, U, G, M, Q, R, H, C, V, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegrable}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, common-partition compatibility for* L *and* U*, oscillation-gap exactness through* G*, tolerance-modulus coverage through* M*,* $\mathsf{DarbouxCriterion}^{\uparrow}$ *and* $\mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* V*, and local naming through* N*.*

Proof. Project the carrier of Definition 7512.1. The rows $F, P, L, U, G, M, Q, R, H, C, V, N$ are the whole certificate surface, so every obligation must begin by reading one of those rows.

The common-partition obligation uses P before either sum row is consumed. The lower and upper sums therefore compare the same bounded function row F on the same finite partition, and the oscillation gap G is a row over that shared surface.

The tolerance-modulus row M supplies the finite requests used by the Darboux and Riemann–Darboux criterion packets. Once those requests are displayed, Q gives regular rational readback for the gap estimates and R supplies the terminal Real seal.

Finally, H, C, V, N transport, replay, source, and name the same finite packet. Because the carrier has no coordinate for a selected integral value, quotient function equality, measure completion, selected partitions by choice, or ambient completeness, those objects cannot be exported by the local NameCert surface. □

Theorem 7512.3 (Darboux integrable oscillation classifier)**.** *An accepted* $\mathsf{DarbouxIntegrable}^{\uparrow}$ *packet classifies integrability by the displayed oscillation-gap route*

$$\mathsf{DarbouxCriterion}^{\uparrow} \longmapsto \mathsf{LowerDarbouxSum}^{\uparrow} \longmapsto ?? \longmapsto \mathsf{RiemannDarbouxCriterionFinite}^{\uparrow}.$$

Every classifier read must use the bounded function row, partition row, lower and upper sum rows, oscillation gap, tolerance modulus, regular rational readback, and Real seal of Definition 7512.1. It cannot read integrability from the value-level $\mathsf{DarbouxIntegral}^{\uparrow}$ *packet alone.*

Proof. Start with the NameCert obligations of Definition 7512.2. They force the lower and upper Darboux rows to share the same bounded function and partition rows before the gap row is inspected.

The oscillation classifier is then the finite route through G and M. The row G records the upper-minus-lower comparison, while M records the tolerance requests under which the comparison is reusable by Darboux and Riemann–Darboux criteria.

The readback rows Q and R only seal this displayed classifier route. Since $\mathsf{DarbouxIntegral}^{\uparrow}$ is named only in provenance as a downstream value consumer, the predicate-level classifier is independent of any selected integral value. □

Theorem 7512.4 (Darboux integrable Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{DarbouxIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{DarbouxIntegrable}^{\uparrow}$ *packet receives the integrability evidence only through the displayed finite route*

$$F \longmapsto P \longmapsto (L, U) \longmapsto G \longmapsto M \longmapsto Q \longmapsto R.$$

The handoff exports integrability-as-carrier data without exporting a chosen integral value, quotient integrand equality, measure completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the oscillation classifier theorem Definition 7512.3. It identifies the route that any integrability consumer must read before the predicate can be used.

Next use the NameCert obligations. The regular rational and Real rows are downstream of the same gap and modulus rows, so a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer cannot bypass F, P, L, U, G, M when reading Q or R.

For a $\mathsf{DarbouxIntegral}^{\uparrow}$ consumer, the packet supplies only the predicate-level evidence needed before a value packet may be read. It does not add a coordinate for the value itself, so the handoff remains a finite integrability carrier rather than a value certificate. □

Falsifiable prediction: Every $\mathsf{DarbouxIntegral}^{\uparrow}$ consumer that cites $\mathsf{DarbouxIntegrable}^{\uparrow}$ must first display the bounded function, partition, lower sum, upper sum, oscillation gap, tolerance modulus, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: darboux-criterion, lower-darboux-sum, upper-darboux-sum, riemann-darboux-criterion-finite, darboux-integral, regseqrat, real: $\mathsf{DarbouxIntegrable}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries predicate-level oscillation-gap evidence and a Real handoff without carrying the selected Darboux integral value.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxIntegrable}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxIntegrable}^{\uparrow}$ packet is generated from a bounded function row, finite partition row, lower and upper Darboux sum rows, oscillation-gap row, tolerance-modulus row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxIntegrable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7512.1, Definition 7512.2, Definition 7512.3, and Definition 7512.4 seed the finite predicate-level Darboux integrability carrier between criterion data and value-level integral packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxIntegrable}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a selected Darboux integral value, host integrand equality, measure completion, selected partitions by choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: The next closure step requires checked carrier admission, oscillation classifier, and Real handoff rows for the displayed finite packet.

Chapter 7513

A Concrete Naming Certificate for $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$

Following Bishop–Bridges constructive sequence bookkeeping, the $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ packet records the finite route by which a regular-Cauchy tail request is routed through stream windows, regular rational readback, dyadic tolerance arithmetic, and the terminal Real seal. It concretizes the regular-Cauchy diagonal-limit route of chapter 9238 and sits over the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The router is a finite precision-and-tail packet; it does not choose a limit, quotient streams by host equality, supply countable choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7513.1 (StreamName tail-window router carrier)**.** *A* $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (r, W, Q, D, E, H, C, P, N)$$

with the following rows:

1. *r is the finite request row naming the precision budget, tail threshold, and diagonal-window demand.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by that request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row for the displayed window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row comparing the requested precision with the inspected tail cells.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after W, Q, D are displayed.*
6. *H is componentwise* hsame *transport for r, W, Q, D, E.*
7. *C is displayed* Cont *replay for the same route.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamNameTailWindowRouter}^{\uparrow}}$ *row.*

The classifier compares accepted routers only through $r, W, Q, D, E, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, hidden tail selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7513.2 (StreamName tail-window router NameCert obligations)**.** *For every accepted* $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ *carrier* $T = (r, W, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StreamNameTailWindowRouter}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed request, StreamName window coverage, RegSeqRat tail readback, DyadicRatCore tolerance exactness, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every regular-Cauchy diagonal-limit consumer must read those rows in that order before it treats* E *as a Real-facing seal.*

Proof. Project the carrier of Definition 7513.1. The request row r names the only precision and tail data available to the router.

The stream-window row W is downstream of r, and the regular rational readback Q is downstream of that displayed window. Thus a tail read cannot reach $\mathsf{RegSeqRat}^{\uparrow}$ without first exposing the finite StreamName window it uses.

The dyadic row D supplies the tolerance arithmetic for the same request and window. Once W, Q, D are present, the Real seal E is available as a terminal handoff rather than as a selected limit or quotient-stream equality.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores a hidden tail selector, countable-choice schedule, host real equality, or ambient completeness, those objects cannot be exported by the local NameCert surface. □

Falsifiable prediction: Every regular-Cauchy diagonal-limit read through $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ must display the finite request, StreamName window, RegSeqRat tail readback, DyadicRatCore tolerance row, RealUp seal, transport, replay, provenance, and local NameCert rows before the Real-facing output is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite tail request, one StreamName window, RegSeqRat tail readback, DyadicRatCore tolerance arithmetic, and a terminal Real seal as one router packet.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamNameTailWindowRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamNameTailWindowRouter}^{\uparrow}$ packet is generated from a finite precision-and-tail request, a StreamName window row, RegSeqRat tail readback, DyadicRatCore tolerance arithmetic, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamNameTailWindowRouter}^{\uparrow}}$, seed)

Scope closed: Definition 7513.1 and Definition 7513.2 seed the regular-Cauchy diagonal-limit router as finite request, window, tail, tolerance, readback, and Real-seal data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamNameTailWindowRouter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, hidden tail selectors, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: The next closure step requires checked carrier admission, window coverage, tolerance exactness, regular readback, and Real-seal handoff for the displayed router packet.

Chapter 7514

Regular Real Located Modulus NameCert Construction

Following Bishop–Bridges constructive analysis, regular real names carry usable real information only through displayed rational windows, dyadic tolerances, and explicit tail moduli. The $\mathsf{RegularRealLocatedModulus}^{\uparrow}$ packet records that finite-means surface over the L10 Real and Completion substrate. It sits after the stream schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, the located real-name route of chapter 1933, and the regular Cauchy located-modulus sibling of chapter 3655.

Definition 7514.1 (Regular real located modulus carrier)**.** *A* $\mathsf{RegularRealLocatedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, D, Q, E, M, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule for the regular real name.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of positive tolerances, endpoints, and radius rows used by located brackets.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback visible on the windows named by S.*
4. *E is the located interval row displaying lower and upper dyadic bracket data over D, S, Q.*
5. *M is the modulus row assigning each displayed dyadic tolerance in D a tail depth after which the Q-readback remains inside the bracket row E.*
6. *W is the Real-facing seal window, recording the point at which the* $\mathsf{Real}^{\uparrow}$ *row may be read only after S, D, Q, E, M have been exposed.*
7. *H records componentwise* hsame *transport for S, D, Q, E, M, W and the structural rows.*
8. *C records displayed* Cont *replay of the same finite schedule, bracket, modulus, and seal-window reads.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularRealLocatedModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRealLocatedModulus}^{\uparrow}}$ *row binding these displayed rows as one certificate.*

The carrier contains no quotient equality of real names, no selected representative, no selected limit, no host real equality, no trichotomy oracle, and no ambient completeness principle.

Theorem 7514.2 (Regular real located modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealLocatedModulus}^{\uparrow}$ *carrier* $R = (S, D, Q, E, M, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularRealLocatedModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the stream schedule, dyadic ledger, regular readback, located bracket, modulus, Real-seal window, transport, replay, provenance, and local naming rows;*
2. *modulus admission, requiring each request in* M *to consume a positive tolerance row already displayed in* D*;*
3. *located bracket witness, requiring* E *to expose lower and upper dyadic bounds over the same* S, D, Q *rows;*
4. *tail localization, requiring the depth supplied by* M *to bind all later* Q*-reads to the bracket row* E *before* W *is available;*
5. *transport and replay stability for* S, D, Q, E, M, W *through* H *and* C*;*
6. *provenance, requiring* P *to name the StreamName, DyadicRatCore, RegSeqRat, LocatedReal, RegularCauchyLocatedModulus, and Real sibling surfaces used by the packet;*
7. *nonescape, excluding quotient completion equality, selected representatives, selected limits, host equality, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local data.*

Proof. Project the carrier of Definition 7514.1. The projection exposes exactly the rows $S, D, Q, E, M, W, H, C, P, N$

The modulus and bracket obligations are row-local. The modulus row M has no tolerance coordinate other than the positive rows of D, and the located bracket row E reads the same stream schedule and regular rational source through S and Q.

Tail localization is the compatibility of M with E. Once a requested dyadic tolerance is fixed, the depth supplied by M determines which later Q-windows remain inside the displayed bracket before the Real seal window W can be read.

Transport, replay, provenance, and local naming are structural rows. Since the carrier has no coordinate for quotient equality, selected representatives, selected limits, host equality, trichotomy, choice, or ambient completeness, the listed obligations exhaust the local certificate surface. □

Theorem 7514.3 (Regular real located modulus tail localization)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularRealLocatedModulus}^{\uparrow}$ *packet must expose the modulus row* M *and the located tail window* (E, W) *before any Real seal is read. Thus a consumer receives the regular real name only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, located bracket* E*, modulus* M*, Real-seal window* W*, transport* H*, replay* C*, provenance* P*, and local name* N*.*

Proof. Begin with an accepted carrier $R = (S, D, Q, E, M, W, H, C, P, N)$. The only Real-facing coordinate is W, and its row is placed after the stream, dyadic, regular readback, located bracket, and modulus rows.

Apply Definition 7514.2. Tail localization forces each requested tolerance to pass through M and the bracket row E before the seal window is exposed.

The transport and replay rows preserve the same order of reads. Therefore every Real-facing consumer must display the modulus and located tail window before reading the $\mathsf{Real}^{\uparrow}$ seal, and no quotient representative, selected limit, host equality, or ambient completeness route can bypass the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealLocatedModulus}^{\uparrow}$ packet is generated from a StreamName schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a located bracket row, an explicit tail modulus, a Real-seal window, componentwise transport, displayed replay, provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7514.1, Definition 7514.2, and Definition 7514.3 name the seed carrier, finite obligation surface, and tail-localized Real-facing route for regular real located modulus data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real names, selected representatives, selected limits, host real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, modulus admission, located bracket witness, tail localization, transport, replay, provenance, and nonescape for BEDC.Derived.RegularRealLocatedModulusUp.

Chapter 7515

Hyperbolic Visual Metric Modulus NameCert Construction

Following Gromov boundary practice and the Poincare disk tradition, visual metrics expose boundary distance only after an observer base, a shadow comparison, and a tolerance modulus are fixed. The $\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}$ packet records this finite surface for the observer-expansion reading of chapter 9463. It sits after the visual metric route of chapter 3866, the boundary-shadow packet of chapter 3813, the observer-expansion packet of chapter 4167, the dyadic shadow cone of chapter 7503, and the Real-and-Completion substrate of chapter 291 and chapter 38.

Definition 7515.1 (Hyperbolic visual metric modulus carrier)**.** *A* $\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (G, B, S, D, M, R, O, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *source row supplying the visual metric read over the displayed hyperbolic boundary surface.*
2. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *boundary-shadow row that records which shadow comparison is being measured.*
3. *S is the finite shadow-window row, read through the observer-expansion surface of* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for visual-radius and shadow-depth requests.*
5. *M is the modulus row assigning each displayed dyadic tolerance in D a finite boundary-shadow depth for the rows G, B, S.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the visual-distance readback, available only after G, B, S, D, M are exposed.*
7. *O is the observer-expansion row tying the visual metric and shadow modulus to the finite observer surface named by chapter 4167.*
8. *W is the nonescape row refusing completed boundary quotients, selected geodesic representatives, global hyperbolic compactifications, and physical horizon semantics.*
9. *H records componentwise* hsame *transport for G, B, S, D, M, R, O, W and the structural rows.*

10. *C records displayed* Cont *replay from a boundary-shadow request through the visual metric, dyadic tolerance, modulus, Real seal, observer-expansion, and nonescape rows.*

11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShad}$ $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{DyadicShadowCone}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}}$ *row.*

The carrier is modulus data for a displayed visual metric route. It is not a completed Gromov boundary, not a quotient boundary equality, not a selected geodesic, not a physical horizon, and not an ambient compactness theorem.

Theorem 7515.2 (Hyperbolic visual metric modulus NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}$ *carrier* $V = (G, B, S, D, M, R, O, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualM}}$ *surface consists exactly of the following finite obligations:*

1. *visual metric admission for the row G and its displayed boundary surface;*
2. *boundary-shadow admission for the row B and its finite shadow comparison;*
3. *shadow-window admission for S through the observer-expansion route;*
4. *dyadic tolerance admission for D and modulus admission for M;*
5. *Real-seal discipline, requiring R to be read only after G, B, S, D, M have been displayed;*
6. *observer-expansion compatibility for O;*
7. *transport, replay, provenance, and local naming through H, C, P, N;*
8. *nonescape through W, excluding completed boundary quotients, selected geodesic representatives, global hyperbolic compactifications, physical horizon semantics, and ambient compactness.*

Proof. Project the carrier of Definition 7515.1. The geometric rows are G, B, S, the tolerance and modulus rows are D, M, and the Real-facing row is R.

The modulus is finite because every request in M is indexed by a displayed dyadic tolerance in D and by the already visible shadow rows G, B, S. No boundary quotient or selected geodesic coordinate is available to answer a request.

The Real-seal and observer-expansion obligations are then forced by the row order. The seal R is downstream of G, B, S, D, M, while O records the observer-expansion route that receives the same finite shadow-depth data.

Finally W, H, C, P, N record refusal, transport, replay, source, and local naming. These rows preserve the same packet and refuse every global or physical endpoint listed in the statement, so the obligations exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}$ packet is generated from visual metric rows, boundary-shadow rows, observer-expansion shadow windows, dyadic tolerance rows, an explicit modulus row, a Real-seal row, nonescape data, componentwise transport, displayed replay, provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7515.1 and Definition 7515.2 name the seed carrier and finite obligation surface for visual-metric modulus data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetricModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed Gromov boundaries, quotient boundary equality, selected geodesic representatives, global hyperbolic compactification, physical horizon semantics, ambient compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked visual metric admission, boundary-shadow admission, shadow-window admission, dyadic tolerance admission, modulus admission, Real-seal discipline, observer-expansion compatibility, transport, replay, provenance, and nonescape for BEDC.Derived.HyperbolicVisualMetricModulusUp.

Chapter 7516

A Concrete Naming Certificate for $\mathsf{TropicalSemiring}^{\uparrow}$

Following Speyer–Sturmfels and standard idempotent algebra, the $\mathsf{TropicalSemiring}^{\uparrow}$ packet records the finite BEDC surface for the max-plus tropical semiring. It sits over the algebraic inventory of chapter 43, the order-facing comparison rows of chapter 1378, and the scalar readback boundary of chapter 38. The packet names tropical addition and multiplication only through displayed carrier, endpoint, order, transport, replay, provenance, and local NameCert rows; it does not import host real completeness, quotient scalar equality, a selected limit, or an unledgered convention for infinity.

Definition 7516.1 (Tropical semiring carrier)**.** *A* $\mathsf{TropicalSemiring}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, A, M, E, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the carrier-source row, read through the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *scalar boundary together with a displayed endpoint symbol for the tropical additive zero.*
2. *A is the idempotent tropical-addition row. On visible scalar entries it records the chosen max-plus operation, so the displayed addition satisfies $x \oplus x = x$ by comparison with the same inspected value.*
3. *M is the tropical-multiplication row. On finite scalar entries it records ordinary scalar addition, with the endpoint row acting as the absorbing boundary for products involving the tropical additive zero.*
4. *E is the endpoint-convention ledger. It records whether the packet is read in the bottom-adjoined max-plus presentation or the dual top-adjoined min-plus presentation, and it binds the additive identity, absorbing multiplication boundary, and excluded unledgered endpoint cases.*
5. *O is the order-compatibility row. It reads the order surface of chapter 1378 and records monotonicity of A and M on the displayed finite comparisons.*
6. *H records componentwise* hsame *transport for S, A, M, E, O and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from scalar and endpoint rows to tropical sums, tropical products, and order-compatible comparisons.*

8. *P is the* Pkg *provenance over* $\mathsf{TropicalSemiring}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{TropicalSemiring}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, M, E, O, H, C, P, N$. It has no coordinate for host-level real completeness, quotient scalar equality, selected extrema, unbounded optimization data, or an endpoint convention not written in E.

Theorem 7516.2 (Tropical semiring NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TropicalSemiring}^{\uparrow}}$ *surface for Definition 7516.1 consists exactly of carrier-source admission, idempotent tropical-addition exposure, tropical-multiplication exposure, endpoint-convention accounting, order compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier $T = (S, A, M, E, O, H, C, P, N)$. The first five rows expose the scalar source, the two tropical operations, the endpoint convention, and the order comparisons available to consumers.

The endpoint row is part of the packet rather than ambient notation. Hence the additive zero, the absorbing multiplication boundary, and the max-plus or dual min-plus orientation are checked before any operation row is consumed.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the displayed rows exhaust the local NameCert obligations and exclude host completeness, quotient scalar equality, selected extrema, and unrecorded endpoint choices. □

Theorem 7516.3 (Tropical idempotent order route)**.** *Every order-facing consumer of an accepted* $\mathsf{TropicalSemiring}^{\uparrow}$ *packet reads tropical idempotence and operation monotonicity through the same finite route*

$$S, E \longmapsto A, M \longmapsto O$$

before it may use the packet for path-weight, optimization, or idempotent-algebra arguments. The route exports no total comparison beyond the displayed order rows and no completed-real supremum principle.

Proof. Begin with the carrier-source and endpoint rows S, E. They determine which scalar entries and which additive boundary are visible to the packet.

Next read A and M. Idempotence of tropical addition is the diagonal projection of the displayed max-plus comparison in A, while multiplication is ordinary scalar addition only on entries admitted by S and the boundary convention recorded by E.

Finally read O. The order-compatibility row is the only place where monotonicity is exported to a consumer, and it is transported and replayed only through H and C. Therefore downstream path-weight or optimization packets receive a finite idempotent order route, not a host supremum theorem, trichotomy principle, selected extremum, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{TropicalSemiring}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TropicalSemiring}^{\uparrow}$ packet is generated from a scalar carrier source, idempotent tropical addition, tropical multiplication, endpoint-convention ledger, order-compatibility rows, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TropicalSemiring}^{\uparrow}}$, seed)

Scope closed: Definition 7516.1, Definition 7516.2, and Definition 7516.3 seed the finite max-plus tropical semiring surface over scalar, endpoint, order, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TropicalSemiring}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host tropical algebra, completed-real suprema, quotient scalar equality, selected extrema, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, idempotent-addition exposure, multiplication exposure, endpoint accounting, order compatibility, transport, replay, provenance, and local naming for the same finite packet.

Chapter 7517

A Concrete Naming Certificate for $\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley metric completion, the TotallyBoundedCompletionCri packet records the finite-metric criterion that a displayed metric source reaches its completion-facing compact metric form by combining finite ε-net data with complete-metric limit data. It consumes the total-boundedness surface of chapter 127, the complete-metric limit surface of chapter 128, the metric-completion handoff of chapter 1506, and the compact-metric certificate surface of chapter 129. The packet is a BEDC criterion row, not an open-cover compactness theorem, a selected-limit principle, a quotient-completion equality, or a host compactness claim.

Definition 7517.1 (Totally bounded completion criterion carrier)**.** *A* $\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, E, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *source row, including its point classifier and* MetricDistanceWitness *surface.*
2. *E is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger, recording for each visible positive* $\mathsf{Rat}^{\uparrow}$ *tolerance a closed* ProbeBundle *of centers, membership rows, coverage rows, and rational bound comparisons.*
3. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit ledger, recording regular* $\mathsf{StreamName}^{\uparrow}$ *Cauchy names, explicit moduli, displayed limit histories, and convergence-distance rows.*
4. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, recording whether a completion-facing consumer reads the source through the Cauchy-filter branch or the Cauchy-net branch before any separated-limit comparison.*
5. *H records componentwise* hsame *transport for X, E, L, Q, the replay row, the provenance row, and the local naming row.*
6. *C records displayed* Cont *replay for finite-net coverage, complete-limit witnesses, completion handoff, separated comparison, and compact-metric consumption.*
7. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected limits, countable choice, quotient completion equality, host compactness, or a detached ambient complete metric space.

Theorem 7517.2 (Totally bounded completion criterion NameCert obligations)**.** *For an accepted* $\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}$ *packet* $T = (X, E, L, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletionCrit}}$ *surface consists exactly of carrier admission for the displayed metric source X, finite-net coverage through E, complete-limit witnesses through L, completion handoff through Q, componentwise transport through H, replay through C, package provenance through P, and local naming through N.*

Every compact-metric consumer of this criterion must read the finite-net row before invoking total boundedness, read the limit row before invoking completeness, and pass through the completion handoff row before comparing displayed completion endpoints. No consumer can project open-cover compactness, selected representatives, countable-choice schedules, quotient equality of completion names, or host compactness from the packet.

Proof. Project the accepted packet through Definition 7517.1. The source metric row is X, and every finite-net request is forced through E because no other row carries a $\mathsf{ProbeBundle}$ center ledger or rational coverage comparison.

The complete-metric part is forced through L. It carries the regular stream name, modulus, displayed limit, and convergence-distance rows, while the completion-facing branch is forced through Q before separated comparison can be replayed.

The structural rows H, C, P, N transport, replay, package, and name exactly the same finite tuple. Since the tuple has no coordinate for open covers, selected limits, countable choice, quotient completion equality, or host compactness, the listed obligations exhaust the local NameCert surface. □

Theorem 7517.3 (Totally bounded completion finite-net route)**.** *Let $T = (X, E, L, Q, H, C, P, N)$ be an accepted* $\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}$ *packet. Every finite-net compactness use of T factors first through the* $\mathsf{TotallyBounded}^{\uparrow}$ *row E, then through the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row L, and then through the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row Q when the consumer asks how a Cauchy-filter or Cauchy-net reading reaches a displayed completion endpoint.*

The resulting route is the finite BEDC chain

$$X \longmapsto E \longmapsto L \longmapsto Q \longmapsto H, C, P, N,$$

not an open-cover compactness proof, not a selected-limit principle, not a quotient of completion names, and not a host compact metric object.

Proof. Start with a compact-metric consumer. The metric carrier and distance rows are projected from X, while total boundedness is available only through the finite $\mathsf{ProbeBundle}$ data in E.

Completeness is read next through L. This row supplies the displayed regular Cauchy stream, modulus, limit, and convergence-distance witnesses; it does not select a host representative or assert ambient completeness outside the packet.

When the consumer asks for the completion-facing route, Q records whether the visible evidence is filter-based or net-based before any separated-limit comparison is used. The transport, replay, provenance, and local naming rows then keep the same finite chain stable, so no open-cover, choice, quotient-equality, or host-compactness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}$ packet is generated from a metric source row, a finite-net ledger, a complete-limit ledger, a metric-completion handoff row, componentwise transport, replay, provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7517.1, Definition 7517.2, and Definition 7517.3 bind the finite NameCert criterion joining total boundedness, complete metrics, and metric completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletionCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected limits, countable choice, quotient completion equality, or host compactness.

Upgrade path: The next paper-side grade requires a scoped dependency theorem over the displayed metric source, finite-net ledger, complete-limit ledger, completion handoff, transport, replay, provenance, and local naming rows.

Chapter 7518

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$ packet records the filter presentation of regular Cauchy completion as finite BEDC data. It sits between the Cauchy-filter basis route of chapter 1501, the sequence-indexed filter route of chapter 6712, finite stream windows from chapter 126, regular rational readback from chapter 301, dyadic tolerance ledgers from chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 filter route for completion without quotienting filters, selecting limits, importing countable choice, using trichotomy, or appealing to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7518.1 (Bishop regular Cauchy filter carrier)**.** *A* $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, S, W, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row generated by regular tail requests and common-refinement witnesses.*
2. *S is the* $\mathsf{SequentialCauchyFilter}^{\uparrow}$ *row tying the basis to a sequence-indexed Cauchy-filter read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by the inspected tail requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row exposed from those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window and readback route.*
6. *Q is the Cauchy-filter saturation row, recording that every completion-facing filter request factors through the displayed basis route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, S, W, R, D, Q have been displayed.*
8. *H is the componentwise* hsame *transport row for the basis, sequential filter, windows, readback, dyadic ledger, saturation row, real seal, replay, provenance, and local naming.*
9. *C is the displayed* Cont *replay row for the same finite filter route.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SequentialCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, W, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient filters, selected limit points, host equality of filters, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7518.2 (Bishop regular Cauchy filter NameCert obligations)**.** *For every accepted* BishopRegularCauchyFil *carrier $F = (B, S, W, R, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; Cauchy-filter basis generation through B; sequence-indexed filter routing through S; finite stream-window exposure through W; regular rational readback through R; dyadic tolerance accounting through D; filter saturation through Q; terminal real-seal boundary through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Carrier admission is tuple projection from Definition 7518.1. The displayed filter data are exactly B, S, Q, the completion-facing readback data are W, R, D, E, and the structural certificate rows are H, C, P, N.

The basis and sequence rows are read first. The row B records the finite common-refinement surface for regular Cauchy tails, while S keeps that basis tied to the sequence-indexed Cauchy-filter route before saturation is consumed.

The completion-facing rows are downstream. Rows W, R, D expose finite observations, regular rational readback, and dyadic tolerance accounting before the real seal E is available, and Q records that the filter request remains attached to the displayed basis.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no remaining coordinate stores quotient filters, selected limits, host filter equality, trichotomy, countable choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 7518.3 (Bishop regular Cauchy filter Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$ *carrier reaches the terminal seal only through*

$$\mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{SequentialCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the saturation row Q, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}$ *row. The packet exports no quotient filter, selected limit, host filter equality, trichotomy, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the obligation surface of Definition 7518.2. It forces every accepted read to expose the finite Cauchy-filter basis, the sequence-indexed route, the stream windows, the regular rational readback, and the dyadic tolerance ledger before the terminal seal appears.

The saturation row Q keeps the filter request attached to the displayed basis route. A completion-facing consumer therefore receives the Cauchy-filter presentation as finite refinement data, not as a quotient filter or selected limit object.

Transport, replay, provenance, and local naming preserve only that route. Because the carrier has no coordinate for quotient filters, selected limits, host filter equality, trichotomy, countable choice, or ambient completeness, none of those endpoints can be exported by a $\mathsf{Real}^{\uparrow}$ seal consumer. □

Falsifiable prediction: A $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$ completion read that exports a terminal $\mathsf{Real}^{\uparrow}$ seal without displaying CauchyFilterBasis, SequentialCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, saturation, transport, replay, provenance, and local NameCert rows refutes the carrier of Definition 7518.1.

Independence witness: cauchy-filter-basis, sequential-cauchy-filter, streamname, regseqrat, dyadicratcore, real: no listed sibling carries the whole Bishop regular Cauchy filter packet because each supplies only a basis generator, sequence-indexed filter route, finite window row, regular readback, dyadic tolerance ledger, or terminal seal while this chapter binds them into one L10 completion-facing filter certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyFilter}^{\uparrow}$ packet is generated from CauchyFilterBasis rows, SequentialCauchyFilter routing, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance rows, a Cauchy-filter saturation row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7518.1, Definition 7518.2, and Definition 7518.3 seed the finite L10 filter presentation of regular Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host filter equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter basis generation, sequence-indexed filter routing, finite window exposure, regular rational readback, dyadic tolerance exactness, saturation, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyFilterUp.

Chapter 7519

A Concrete Naming Certificate for MetaCICTypedSubstitutionWindow$^{\uparrow}$

The MetaCICTypedSubstitutionWindow$^{\uparrow}$ packet realizes the MetaCIC substitution audit boundary of chapter 9380 as a finite typed-window certificate. It sits beside the substitution spine of chapter 6103, the audit packet of **??**, the closed audit map of chapter 779, the route window of chapter 824, the context audit of chapter 840, the generator audit of chapter 910, the boundary audit of chapter 1102, the substitution spine realizer of chapter 6990, and the closed type-checker seal of chapter 2182. The packet records typed context, closedness, substitution, obstruction, transport, replay, provenance, and NameCert rows without claiming subject reduction, definitional equality, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 7519.1 (MetaCIC typed substitution window carrier)**.** *A* MetaCICTypedSubstitutionWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Gamma, A, U, V, C, S, O, B, H, R, P, N)$$

with the following displayed rows:

1. Γ *is the finite typed-context window, listing the context entries inspected by the substitution request.*
2. A *is the typed source-term row and target-type row inside that context window.*
3. U *is the substitute-in term row, carrying its typed boundary and the binder depth at which it is inserted.*
4. V *is the closedness row, recording the relevant* `ClosedAt` *boundaries for the source, inserted term, and substituted output.*
5. C *is the closed type-checker seal row consumed by the typed window.*
6. S *is the substitution audit route, recording shift, substitution, closed-composition, and generator-closure reads.*
7. O *is the obstruction row for missing subject reduction, beta preservation beyond the sealed window, evaluator equality, quotient syntax equality, and kernel consistency.*
8. B *is the binder-transport row that records how the typed context window advances across binders.*
9. H *is the componentwise* hsame *transport row for* $\Gamma, A, U, V, C, S, O, B$ *and for replay, provenance, and local naming.*

10. *R is the displayed* Cont *replay row for the same typed substitution route.*

11. *P is the* Pkg *provenance over* $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$, $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$, $\mathsf{MetaCICSubstitutionAuditPacket}^{\uparrow}$, $\mathsf{SubstitutionAuditMap}^{\uparrow}$, $\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}$, $\mathsf{SubstitutionContext}^{\uparrow}$ $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$, $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for an unrestricted subject-reduction theorem, definitional equality oracle, evaluator equality, quotient syntax equality, or kernel consistency theorem.

Theorem 7519.2 (MetaCIC typed substitution window NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$ *carrier* $T = (\Gamma, A, U, V, C, S, O, B, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}}$ *surface consists exactly of typed-context admission through* Γ*; source and inserted-term typing through* A *and* U*; closedness boundary exposure through* V*; closed type-checker sealing through* C*; shift, substitution, closed-composition, and generator-closure routing through* S*; obstruction retention through* O*; binder-window transport through* B*; componentwise* hsame *transport through* H*; displayed replay through* R*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 7519.1. The typed-window rows are $\Gamma, A, U, V, C, S, O, B$, and the structural certificate rows are H, R, P, N.

Read the context rows first. The finite context window Γ, typed source data A, inserted-term data U, and closedness row V determine the boundary at which substitution is inspected.

Read the audit rows next. The closed type-checker seal C scopes the window, while S records the shift, substitution, closed-composition, and generator-closure route. The binder row B is the only row that advances the typed window across binders.

Finally O, H, R, P, N retain refused endpoints and structural certificate data. Since the carrier has no row for full subject reduction, definitional equality, evaluator equality, quotient syntax equality, or kernel consistency, the listed rows exhaust the local NameCert surface. □

Theorem 7519.3 (MetaCIC typed substitution window subject-reduction handoff)**.** *Every subject-reduction consumer of an accepted* $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$ *packet must first read the typed context window, typed source and inserted-term rows, closedness row, closed type-checker seal, substitution audit route, binder-window transport, obstruction row, componentwise transport, replay, provenance, and local naming. The handoff exposes only a finite typed substitution window; it does not prove unrestricted subject reduction, beta preservation beyond the displayed window, definitional equality, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Start with the obligation theorem Definition 7519.2. It orders every accepted read through the typed context, source typing, inserted-term typing, closedness, closed checker, substitution audit, binder transport, and obstruction rows.

The typed rows scope the handoff. A consumer cannot read the substitution route before the context window, term rows, and closedness boundary are visible, because those rows determine which substitution instance is under inspection.

The audit and obstruction rows then delimit the result. The substitution route gives finite shift, substitution, closed-composition, and generator-closure evidence, while the obstruction row records the endpoints not discharged by the packet.

The structural rows transport, replay, source, and name only that displayed route. Therefore the packet may feed a later subject-reduction proof, but it cannot itself be read as unrestricted subject reduction or as a kernel-consistency theorem. □

Falsifiable prediction: A $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$ consumer that exports a subject-reduction handoff without typed-context, source typing, inserted-term typing, closedness, closed checker, substitution audit, binder transport, obstruction, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 7519.1.

Independence witness: metacic-substitution-spine, metacic-substitution-audit-packet, substitutionauditmap, substitutionauditwindowroute, substitutioncontextaudit, substitutiongeneratoraudit, substitutionboundaryaudit, closed-type-checker-seal: the typed substitution window is not bijective to any listed sibling because it jointly carries typed context rows, typed source and inserted-term rows, ClosedAt boundaries, closed checker sealing, substitution audit routing, binder-window transport, obstruction retention, replay, provenance, and local naming.

Lean TasteGate.lean must later provide Nontrivial and FieldFaithful instances for MetaCICTypedSubstitutionWind so the typed-context, closedness, substitution, obstruction, transport, replay, provenance, and local naming fields remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$ packet is generated from typed-context rows, typed source and inserted-term rows, ClosedAt boundaries, a closed checker seal, substitution audit rows, binder-window transport, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7519.1, Definition 7519.2, and Definition 7519.3 seed the finite typed substitution window for MetaCIC subject-reduction consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted subject reduction, beta preservation beyond the displayed window, definitional equality, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-context admission, source and inserted-term typing, closedness exposure, closed checker sealing, substitution audit routing, binder transport, obstruction retention, transport, replay, provenance, Nontrivial, FieldFaithful, and local naming for BEDC.Derived.MetaCICTypedSubstitutionWindowUp.

Chapter 7520

WeylEquidistribution NameCert Construction

Following Weyl, the $\mathsf{WeylEquidistribution}^{\uparrow}$ packet reads equidistribution through finite rational interval tests over $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not through measure or choice. It sits after the rational interval surface of chapter 354, the finite stream-name surface of chapter 126, the regular rational sequence surface of chapter 301, and the real-name seal of chapter 38. The packet names only the finite test ledger, discrepancy budget, transport, replay, provenance, and local NameCert rows needed before a constructive distribution statement can be consumed.

Definition 7520.1 (Weyl equidistribution finite carrier)**.** *A* $\mathsf{WeylEquidistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, I, K, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence-source row, reading a* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *name indexed by* $\mathsf{Nat}^{\uparrow}$ *and sealed only through the displayed* $\mathsf{Real}^{\uparrow}$ *readback.*
2. *I is the finite rational interval-test ledger. Each entry names rational endpoints from* $\mathsf{Rat}^{\uparrow}$*, a half-open interval in the unit window, and the finite prefix length over which the test is inspected.*
3. *K is the counting row, recording for each entry of I the finite set of indices whose displayed real readback falls in that rational interval.*
4. *D is the discrepancy row, comparing the counted frequency with the rational interval length by a displayed* $\mathsf{Rat}^{\uparrow}$ *tolerance.*
5. *E is the exhaustion row, recording the finite list of interval tests currently exposed to a consumer and the remaining tests not claimed by the packet.*
6. *H records componentwise* hsame *transport for the sequence source, interval ledger, counting row, discrepancy row, exhaustion row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite interval-test request through S, I, K, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{WeylEquidistribution}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WeylEquidistribution}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, I, K, D, E, H, C, P, N$. It has no coordinate for a measure space, an infinite choice of test intervals, a selected subsequence, host real equality, or an unbounded limiting frequency claim.

Lean4 ✓ `BEDC.Derived.WeylEquidistributionUp.WeylEquidistributionTasteGate single carrier alignment`

Theorem 7520.2 (WeylEquidistribution NameCert obligations)**.** *For every accepted* $\mathsf{WeylEquidistribution}^{\uparrow}$ *carrier* $W = (S, I, K, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WeylEquidistribution}^{\uparrow}}$ *surface consists exactly of sequence-source admission, finite rational interval-test exposure, finite counting exposure, discrepancy-budget exposure, test-exhaustion accounting, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier coordinates. The rows S, I, K, D, E expose the inspected sequence name, the finite interval ledger, the counted index set for each ledger entry, the rational discrepancy comparison, and the finite-test exhaustion boundary.

The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. They do not add a measure object, an infinite interval family, or a selected representative outside the displayed sequence row.

Since the carrier has no coordinate outside $S, I, K, D, E, H, C, P, N$, every accepted local NameCert obligation is one of the listed rows, and each listed row is present before the packet is public to a downstream distribution consumer. □

Theorem 7520.3 (WeylEquidistribution interval-test coverage)**.** *Every constructive distribution statement consumed from an accepted* $\mathsf{WeylEquidistribution}^{\uparrow}$ *packet first factors through the finite route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow}\text{-}\textit{interval tests} \longmapsto \mathsf{Rat}^{\uparrow}\text{-}\textit{discrepancy budgets}.$$

The route covers exactly the rational interval tests displayed in I and recorded by E; it does not assert measure-theoretic equidistribution, an unrestricted limiting frequency, countable choice, or host equality of real representatives.

Proof. Start with an accepted packet W. Carrier admission exposes the sequence source only through the displayed natural-indexed stream or regular rational name and the terminal real readback named in S.

Next read the interval ledger I and the counting row K. Each test is finite: it names rational endpoints, a prefix length, and the counted indices visible inside that prefix.

Finally read the discrepancy and exhaustion rows D, E. They record the rational tolerance currently certified and the finite test list currently covered. Thus any downstream constructive distribution statement receives finite interval-test coverage and no hidden measure, choice principle, or unbounded limit assertion. □

Falsifiable prediction: Every accepted WeylEquidistributionUp consumer that exports distribution evidence must expose a finite rational interval-test ledger, finite counts, rational discrepancy budgets, exhaustion accounting, transport, replay, provenance, and local naming before export.

Independence witness: real, rat, streamname, regseqrat, rationalinterval: $\mathsf{WeylEquidistribution}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the natural-indexed sequence source, rational interval ledger, finite counting row, discrepancy budget, exhaustion row, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeylEquidistribution}^{\uparrow}}$.

Constructive story: A $\mathsf{WeylEquidistribution}^{\uparrow}$ packet is generated from a natural-indexed sequence source, rational interval-test ledger, finite counting rows, rational discrepancy budgets, test-exhaustion accounting, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeylEquidistribution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7520.1, Definition 7520.2, and Definition 7520.3 seed the finite rational interval-test surface for constructive equidistribution reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeylEquidistribution}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic equidistribution, unrestricted limiting frequencies, countable choice, selected subsequences, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite interval-test exposure, counting exposure, rational discrepancy accounting, exhaustion accounting, transport, replay, provenance, and local naming for BEDC.Derived.WeylEquidistributionUp.

Chapter 7521

A Concrete Naming Certificate for TotalBoundedCompletionCompactness$^{\uparrow}$

Following Bishop and Bridges on compact metric completions, the TotalBoundedCompletionCompactness$^{\uparrow}$ packet records the finite handoff by which a totally bounded completion route may be read by compact-metric consumers while retaining explicit net and Cauchy rows. It sits after the total-bounded completion route of chapter 1573, the precompact completion route of chapter 1842, and the constructive metric completion route of chapter 1556, and it supplies a compactness-facing boundary for chapter 129. The packet is a finite NameCert surface; it is not an open-cover compactness theorem, a countable-choice principle, a quotient completion, or an ambient Real$^{\uparrow}$ completeness theorem.

Definition 7521.1 (TotalBoundedCompletionCompactness carrier)**.** *A* TotalBoundedCompletionCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, F, C, Q, S, H, R, P, N)$$

with the following displayed rows:

1. *M is the source metric carrier row, including the point histories, metric-distance witnesses, and classifier transport data used by the completion route.*
2. *T is the total-bounded finite-net ledger inherited from* TotallyBoundedCompletion$^{\uparrow}$ *and* PrecompactCompletion$^{\uparrow}$*, naming centers, membership rows, rational radius bounds, and refinement reads for each displayed tolerance request.*
3. *F is the* CauchyFilterBasis$^{\uparrow}$ *or regular Cauchy basis row generated from the same finite-net data.*
4. *C is the* ConstructiveMetricCompletion$^{\uparrow}$ *route through dense embedding, regular Cauchy readback, dyadic tolerance control, and the terminal* Real$^{\uparrow}$ *naming boundary.*
5. *Q is the compact-metric reading row, allowing a downstream* CompactMetric$^{\uparrow}$ *consumer to inspect only the displayed total-bounded completion route.*
6. *S is the seal row recording that every compactness-facing read factors through T, F, C, Q before any completion point, net refinement, or real-name boundary is consumed.*
7. *H records componentwise* hsame *transport for the metric, net ledger, basis row, constructive completion route, compact-metric reading row, seal row, provenance, and local naming rows.*
8. *R records displayed* Cont *replay for the same row order.*

9. *P is the* Pkg *provenance over* TotalBoundedCompletionCompactness$^{\uparrow}$, TotallyBoundedCompletion$^{\uparrow}$, PrecompactCompletion$^{\uparrow}$, ConstructiveMetricCompletion$^{\uparrow}$, CompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{TotalBoundedCompletionCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, T, F, C, Q, S, H, R, P, N$. No coordinate stores open-cover compactness, a countable-choice selector, a quotient completion, selected completion representatives, detached host equality, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7521.2 (TotalBoundedCompletionCompactness NameCert obligations)**.** *For every accepted* TotalBoundedCompletionCompactness$^{\uparrow}$ *carrier*

$$K = (M, T, F, C, Q, S, H, R, P, N),$$

the local NameCert$_{\text{TotalBoundedCompletionCompactness}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed metric row, finite-net ledger, Cauchy basis row, constructive completion route, compact-metric reading row, seal row, transport, replay, provenance, and local naming row;*

2. *finite-net persistence, requiring every compactness-facing tolerance request to expose the centers, membership constructors, rational radius bounds, and refinement reads stored in T;*

3. *completion-route admission, requiring the basis row F and constructive route C to be displayed before the compact-metric reader reaches Q;*

4. *compactness-facing seal discipline, requiring S to record the same T, F, C, Q order for every downstream compact-metric or uniform-continuity consumer;*

5. *nonescape from open-cover compactness, countable choice, quotient completion, selected representatives, host equality, and ambient* Real$^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these displayed rows.

Proof. Carrier admission is projection from Definition 7521.1. The accepted packet displays exactly the metric row, total-bounded finite-net ledger, Cauchy basis row, constructive completion route, compact-metric reading row, seal row, and structural rows.

Finite-net persistence is read from T. A compactness-facing tolerance request first opens the same finite centers, membership rows, radius bounds, and refinement data already named by TotallyBoundedCompletion$^{\uparrow}$ and PrecompactCompletion$^{\uparrow}$; it cannot replace them by an open cover or by a selected convergent subnet.

The completion route is then forced through F and C. The basis row is generated from the displayed finite-net ledger, and the constructive completion row exposes only dense embedding, regular Cauchy readback, dyadic tolerance, and terminal real-name data before Q is visible.

Finally S, H, R, P, N seal, transport, replay, source, and name the same finite route. Since the excluded principles have no carrier coordinate and no replay edge, the five rows exhaust the local NameCert surface. □

Theorem 7521.3 (TotalBoundedCompletionCompactness compact handoff)**.** *Every accepted* TotalBoundedComple *carrier $K = (M, T, F, C, Q, S, H, R, P, N)$ hands a compact-metric or uniform-continuity consumer only the displayed finite route*

$$M, T \longmapsto F \longmapsto C \longmapsto Q \longmapsto S, H, R, P, N.$$

The route reads $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *or* $\mathsf{PrecompactCompletion}^{\uparrow}$ *net data before it reads* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *completion data, and it reads the* $\mathsf{CompactMetric}^{\uparrow}$ *boundary only as a consumer of that route.*

Proof. Begin with the finite-net ledger T over the source metric row M. These rows provide the only tolerance and center data available to the packet.

The basis and completion rows F and C are downstream of that ledger. They expose the constructive completion route without adding quotient representatives, selected limits, countable choice, or an ambient completeness theorem.

The compact-metric row Q is then a reader of the displayed route, not an independent compactness proof. The seal and structural rows S, H, R, P, N preserve the same order, so every downstream consumer receives the finite handoff stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotalBoundedCompletionCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{TotalBoundedCompletionCompactness}^{\uparrow}$ packet is generated from a source metric row, total-bounded finite-net ledger, Cauchy basis row, constructive completion route, compact-metric reading row, seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalBoundedCompletionCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7521.1, Definition 7521.2, and Definition 7521.3 seed the finite completion-to-compactness route over TotallyBoundedCompletionUp, PrecompactCompletionUp, ConstructiveMetricCompletionUp, CompactMetricUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalBoundedCompletionCompactness}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, quotient completion, selected representatives, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove finite-net persistence, completion-route admission, compactness-facing seal discipline, transport, replay, provenance, and nonescape for the same packet.

Chapter 7522

A Concrete Naming Certificate for $\mathsf{SchwartzDistribution}^{\uparrow}$

Following Schwartz, the $\mathsf{SchwartzDistribution}^{\uparrow}$ packet records a finite distribution-action surface over displayed test windows. It sits beside the rapidly decreasing carrier of chapter 3399, the compact-support test probe of chapter 6945, the tempered-distribution handoff of chapter 2310, the weak-action route of chapter 3751, and the terminal scalar seal of chapter 38. The chapter names a finite BEDC carrier for distribution reads on Schwartz-style test windows; it does not import the full locally convex dual, measure-theoretic distribution theory, Fourier transform completeness, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7522.1 (Schwartz distribution carrier)**.** *A* $\mathsf{SchwartzDistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, T, A, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-window row. It records the rapidly decreasing smooth windows, derivative orders, polynomial-weight labels, and finite observation domains visible to the packet.*
2. *T is the* $\mathsf{TestFunction}^{\uparrow}$ *compact-support row. It records the compact test windows through which local reads may be compared with the Schwartz-function row.*
3. *A is the response-ledger row. It lists the finite action values assigned to the displayed test windows and records which response belongs to which test-window request.*
4. *W is the test-window coverage row. It records that every public action read names a test window in S or T before any scalar response in A is exposed.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational approximations to the finite response ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row reached only after the corresponding rational readback in Q has been displayed.*
7. *L is the local classifier row. It admits two packets together exactly when their Schwartz windows, compact-support windows, response ledger, coverage row, rational readback, and Real seal match componentwise.*
8. *H records componentwise* hsame *transport for S, T, A, W, Q, E, L.*
9. *C records displayed* Cont *replay from a distribution-action request through S, T, W, A, Q, E, L, H in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{SchwartzDistribution}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TestFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{DistributionDerivative}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzDistribution}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for the full locally convex dual, measure completion, quotient distribution equality, Fourier transform completeness, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SchwartzDistributionUp.SchwartzDistributionTasteGate single carrier alignment`

Theorem 7522.2 (Schwartz distribution NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzDistribution}^{\uparrow}$ *carrier* $D = (S, T, A, W, Q, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SchwartzDistribution}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed test-window rows, response-ledger rows, test-window coverage, rational readback, Real scalar sealing, local classification, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 7522.1. The rows $S, T, A, W, Q, E, L, H, C, P, N$ are the whole visible packet, so every local obligation starts from one of those displayed rows.

The test-window obligations are the rows S, T, W. The row S exposes Schwartz-function windows, T exposes compact-support test windows, and W records that a public action read first names one of those windows.

The response obligations are the rows A, Q, E. The response ledger A stores only finite action values for displayed windows, Q gives their regular-rational readback, and E supplies the terminal Real scalar seal after the readback has been opened.

Finally, L, H, C, P, N compare, transport, replay, source, and name the same finite distribution-action surface. Since the carrier has no row for a full locally convex dual, measure completion, Fourier transform completeness, quotient distribution equality, selected representatives, or countable choice, none of those objects belongs to the local NameCert surface. □

Theorem 7522.3 (Schwartz distribution test-window nonescape)**.** *Every* $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{DistributionDerivative}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{SchwartzDistribution}^{\uparrow}$ *packet receives a scalar action only through the finite route*

$$S, T \longmapsto W \longmapsto A \longmapsto Q \longmapsto E \longmapsto L, H, C, P, N.$$

Thus no consumer can extract a response without first displaying a Schwartz or compact-support test window, and no consumer can extract the full locally convex dual, measure-theoretic distribution data, Fourier transform completeness, quotient distribution equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Start with an accepted carrier and apply Definition 7522.2. The theorem identifies the visible rows and in particular forces every response read to pass through the coverage row W.

The route to a scalar response is then fixed. A consumer reads either the Schwartz-function window S or the compact-support test row T, checks the coverage row W, and only then reads the response ledger A.

The numeric endpoint remains downstream of A. The regular-rational readback Q and Real seal E expose scalar names for the displayed response, while L, H, C, P, N only compare, transport, replay, source, and name the same route.

Because the carrier inventory contains no dual-space coordinate, no measure-completion coordinate, no Fourier-completeness coordinate, no quotient-distribution equality, and no selected representative, the finite route cannot export those principles to tempered-distribution, derivative, or Real-facing consumers. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzDistribution}^{\uparrow}$ read that reaches a scalar response must first expose a SchwartzFunctionUp or TestFunctionUp window, a coverage row, a response-ledger entry, RegSeqRat readback, RealUp sealing, classifier, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: schwartzfunction, testfunction, tempereddistribution, distributionderivative, real: $\mathsf{SchwartzDistribution}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries test-window rows, response-ledger rows, coverage, readback, scalar seals, transport, replay, provenance, and local naming while refusing full dual-space and measure-theoretic distribution data.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzDistribution}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchwartzDistribution}^{\uparrow}$ packet is generated from SchwartzFunction and TestFunction window rows, a response ledger, a test-window coverage row, RegSeqRat readback, RealUp scalar sealing, local classification, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzDistribution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7522.1, Definition 7522.2, and Definition 7522.3 seed the finite distribution-action surface over Schwartz and compact-support test windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzDistribution}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full locally convex dual, measure-theoretic distribution theory, Fourier transform completeness, quotient distribution equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: The next closure step requires checked carrier admission, test-window coverage, response-ledger stability, scalar readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SchwartzDistributionUp.

Chapter 7523

A Concrete Naming Certificate for $\mathsf{JungTheorem}^{\uparrow}$

Following Jung's theorem and compact metric radius bounds, the $\mathsf{JungTheorem}^{\uparrow}$ packet records the finite BEDC surface by which a displayed compact metric source exports a radius-bound handoff through Chebyshev-center and finite-projection rows. It sits after the compact-metric source of chapter 129, the metric carrier of chapter 57, the Chebyshev-center certificate of chapter 6623, the finite metric projection boundary of chapter 7348, and the terminal real-radius seal of chapter 38. The packet names one constructive radius-bound route; it is not a host inner-product theorem, selected center principle, quotient compactum, ambient completeness claim, or choice-based extremal argument.

Definition 7523.1 (Jung theorem carrier)**.** *A* $\mathsf{JungTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, M, C, F, R, B, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including its finite-net and compactness-facing observation schedule.*
2. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *distance row read by every radius comparison.*
3. *C is the* $\mathsf{ChebyshevCenter}^{\uparrow}$ *center-candidate row, exposing the finite center ledger before any radius claim is read.*
4. *F is the* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *projection row, recording the finite subconfiguration on which the radius comparison is witnessed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius-seal row carrying the displayed bound as real-name data.*
6. *B is the Jung-bound handoff row, recording that K, M, C, F are consumed before R is exported.*
7. *H records componentwise* hsame *transport for the compact source, metric distance row, center row, projection row, radius seal, bound handoff, replay row, provenance row, and local naming row.*
8. *T records displayed* Cont *replay for the same row order.*
9. *P is the* Pkg *provenance over* $\mathsf{JungTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ChebyshevCenter}^{\uparrow}$, $\mathsf{FiniteMetricProjection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{JungTheorem}^\uparrow}$ *row.*
The classifier compares accepted packets only through $K, M, C, F, R, B, H, T, P, N$. It has no coordinate for host Euclidean structure, selected centers, quotient compacta, ambient completeness, or choice.

Theorem 7523.2 (Jung theorem NameCert obligations)**.** *For every accepted* $\mathsf{JungTheorem}^\uparrow$ *carrier*

$$J = (K, M, C, F, R, B, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{JungTheorem}^\uparrow}$ *surface consists exactly of carrier admission for the displayed compact metric source, metric distance row, Chebyshev-center row, finite-projection row, radius seal, bound handoff, transport, replay, provenance, and local naming row; compact-metric dependency exposure through K; distance comparison exposure through M; center-candidate exposure through C; finite witness exposure through F; real-radius sealing through R; and nonescape from host geometry, selected centers, quotient compacta, ambient completeness, and choice.*

Every downstream consumer of this packet must read the compact metric source before reading the metric distance row, read the Chebyshev-center and finite-projection rows before reading the Jung-bound handoff, and read the $\mathsf{Real}^\uparrow$ *radius row only after that finite route is displayed.*

Proof. Project the accepted carrier through Definition 7523.1. The compact source row K and metric row M are the only places where compactness-facing and distance-facing data occur, so every radius request begins with those rows.

The center and witness data are then forced through C and F. The row C exposes the Chebyshev-center candidate ledger, while F records the finite metric projection that witnesses the displayed radius comparison. No other carrier coordinate can supply a center or finite subconfiguration.

Finally B, R, H, T, P, N hand off, seal, transport, replay, source, and name the same finite route. Since the carrier has no row for host Euclidean structure, selected centers, quotient compacta, ambient completeness, or choice, the listed clauses exhaust the local NameCert surface. □

Theorem 7523.3 (Jung theorem radius-bound handoff)**.** *Every accepted* $\mathsf{JungTheorem}^\uparrow$ *carrier hands a compact-metric consumer only the displayed finite route*

$$K, M \longmapsto C, F \longmapsto B \longmapsto R \longmapsto H, T, P, N.$$

The handoff records a radius-bound interface over $\mathsf{CompactMetric}^\uparrow$, $\mathsf{Met}^\uparrow$, $\mathsf{ChebyshevCenter}^\uparrow$, $\mathsf{FiniteMetricProjection}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *rows. It does not export a host geometric proof, a selected minimizing center, a quotient compact object, an ambient completeness theorem, or a choice principle.*

Proof. Start with the compact-metric and metric rows K, M. They provide the displayed source and distance data that a radius-bound consumer is allowed to inspect.

Next read the center and projection rows C, F. These rows make the finite Chebyshev-center candidate and finite metric witness visible before the bound handoff is consumed.

The handoff and real-radius rows B, R then expose the stated bound as $\mathsf{Real}^\uparrow$ data. The structural rows H, T, P, N preserve the same finite route under transport, replay, provenance, and local naming, so no host geometry, selected center, quotient compactum, ambient completeness, or choice-based extremal principle is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{JungTheorem}^\uparrow}$.

Constructive story: A JungTheorem$^{\uparrow}$ packet is generated from a compact metric source row, metric distance row, Chebyshev-center row, finite-projection row, real-radius seal, bound handoff, componentwise transport, displayed replay, package provenance, and local naming row.

Theory closure: TheoryGate(NameCert$_{\mathsf{JungTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 7523.1, Definition 7523.2, and Definition 7523.3 seed the finite compact-metric radius-bound route through CompactMetricUp, MetricUp, ChebyshevCenterUp, FiniteMetricProjectionUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{JungTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host Euclidean geometry, selected centers, quotient compacta, ambient completeness, Lean theorem verification, or bridge checking.

Upgrade path: The next paper-side grade requires a full obligation package over carrier admission, compact-metric dependency exposure, distance comparison exposure, center and projection rows, radius sealing, transport, replay, provenance, and nonescape.

Chapter 7524

A Concrete Naming Certificate for $\mathsf{MetricDenseEmbedding}^{\uparrow}$

Following Bourbaki and Kelley on dense embeddings in metric and uniform spaces, the $\mathsf{MetricDenseEmbedding}^{\uparrow}$ packet records the metric-specific interface by which a source metric carrier enters a target metric carrier as a dense displayed subobject. It sits between the metric embedding surface of chapter 3339, the uniform dense embedding route of chapter 1651, the completion handoff of chapter 1512, the metric source of chapter 57, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet is a finite NameCert interface for metric density plus graph control; it is not a quotient completion, a selected representative, a host equality of functions, countable choice, or an ambient completeness principle.

Definition 7524.1 (MetricDenseEmbedding carrier)**.** *A* $\mathsf{MetricDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, F, D, I, M, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including its metric-distance readback and classifier transport surface.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row in which the dense image is inspected.*
3. *F is the graph row naming the embedding route from displayed source observations to displayed target observations.*
4. *D is the graph-distance control row, recording the metric comparison budget attached to F without asserting host equality of all distances.*
5. *I is the dense-image witness: every displayed target ball request is met by a source observation routed through F within the named metric tolerance.*
6. *M is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *handoff row used by completion-facing consumers after the dense-image witness has been read.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance readback row attached to the metric budget named by D and I.*
8. *H records componentwise* hsame *transport for the metric rows, graph row, distance-control row, dense-image witness, completion handoff, tolerance readback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay for the same row order.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricDenseEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $X, Y, F, D, I, M, R, H, C, P, N$. No coordinate stores quotient completion classes, selected representatives, selected dense points, host function equality, countable choice, or ambient completeness.

Theorem 7524.2 (MetricDenseEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{MetricDenseEmbedding}^{\uparrow}$ *carrier $E = (X, Y, F, D, I, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the two metric rows, graph route, graph-distance control, dense-image witness, completion-embedding handoff, tolerance readback, componentwise transport, displayed replay, provenance, and local naming. Every dense-image read must first inspect X, Y, F, D, I before it can pass to M, R, H, C, P, N, and every completion-facing read must preserve that same order.*

Proof. Project the accepted carrier of Definition 7524.1. The packet exposes the source and target metric rows, graph row, graph-distance control, dense-image witness, completion handoff, tolerance readback, and the structural rows, with no further coordinate available.

The dense-image coverage clause is the row I. A target ball request is opened inside Y, routed back through a source observation in X, and then returned along F with the metric budget named by D and read back by R.

Transport and replay are structural. The row H moves each displayed coordinate componentwise, and C repeats the same graph-control and dense-image route for later consumers. Since the excluded objects have no carrier coordinate and no replay edge, the listed rows exhaust the local NameCert surface. □

Theorem 7524.3 (MetricDenseEmbedding completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricDenseEmbedding}^{\uparrow}$ *packet $E = (X, Y, F, D, I, M, R, H, C, P, N)$ receives the finite route*

$$X, Y \longrightarrow F \longrightarrow D \longrightarrow I \longrightarrow M \longrightarrow R, H, C, P, N.$$

The handoff is therefore a metric dense embedding route through graph control, dense-image coverage, and completion embedding data. It never replaces those rows by quotient completion classes, selected representatives, selected limits, host equality, countable choice, or ambient completeness.

Proof. Start from carrier projection. The only completion-facing row is M, and it is stored after the source and target metric rows, the graph row, the graph-distance control row, and the dense-image witness.

Use the NameCert obligations of Definition 7524.2. They require any dense-image read to consume X, Y, F, D, I before M or the tolerance readback row R can be used by a downstream completion route.

Finally replay the route through H, C, P, N. These rows transport, replay, source, and name the same finite carrier coordinates, so the completion handoff cannot escape into quotient completion equality, selected representatives, host equality of functions, countable-choice selection, or an ambient completeness assumption. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricDenseEmbedding}^{\uparrow}}$.

Constructive story: A MetricDenseEmbedding$^{\uparrow}$ packet is generated from source and target MetricSpace rows, a graph row, graph-distance control, a dense-image witness, a CompletionEmbedding handoff, RegSeqRat and Real tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetricDenseEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 7524.1, Definition 7524.2, and Definition 7524.3 seed the metric dense embedding interface over MetricSpaceUp, MetricEmbeddingUp, DenseUniformEmbeddingUp, CompletionEmbeddingUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{MetricDenseEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, selected dense points, host function equality, countable choice, ambient completeness, Lean theorem verification, or bridge checking.

Upgrade path: The next grade requires Lean encoding of carrier admission, graph-distance control, dense-image coverage, completion handoff, tolerance readback, transport, replay, provenance, local naming, and nonescape for the displayed packet.

Chapter 7525

A Concrete Naming Certificate for $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$

Following Doob and Meyer, the $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$ packet records the finite predictable-bracket data attached to an inspected martingale prefix. It sits over the adapted martingale surface of chapter 190, the random-variable endpoint rows of chapter 185, the probability source of chapter 184, the conditional-expectation route of chapter 188, and the terminal real-name seal of chapter 38. The packet names the bracket as displayed finite BEDC rows; it does not identify quotient stochastic processes, complete a measure space, select limiting variation, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7525.1 (Martingale quadratic-variation carrier)**.** *A* $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, F, X, D, B, A, E, H, C, P, N)$$

with the following displayed coordinates:

1. *M is the consumed* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet over a displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source.*
2. *F is the finite filtration and visibility row inherited from the martingale packet.*
3. *X is the finite endpoint family $X_0, \ldots, X_n$ of* $\mathsf{RandomVar}^{\uparrow}$ *rows read through M.*
4. *D is the finite difference family $\Delta X_k = X_k - X_{k-1}$ recorded as displayed random-variable endpoint rows for $1 \leq k \leq n$.*
5. *B is the predictable-bracket row whose kth face is read through the conditional-expectation packet as the displayed previous-time expectation of the squared difference row.*
6. *A is the accumulated bracket ledger, built by finite prefix summation of the displayed faces of B.*
7. *E stores the* $\mathsf{CondExp}^{\uparrow}$ *rows used to read each previous-time predictable face.*
8. *H stores componentwise* hsame *transport for the martingale row, filtration row, endpoint family, difference rows, bracket faces, accumulated ledger, conditional-expectation rows, replay, provenance, and local naming data.*
9. *C stores displayed* Cont *replay for opening the martingale prefix, reading a difference row, reading the predictable face, and appending it to the accumulated ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{Martingale}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MartingaleQuadraticVariation}^{\uparrow}}$ *row.*

The carrier is a finite predictable-bracket packet over the inspected martingale prefix. It stores no almost-sure quotient equality of processes, no completed-measure representative, no selected infinite bracket limit, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 7525.2 (Martingale quadratic-variation predictable bracket)**.** *For every accepted* MartingaleQuadraticVar *carrier Q, each bracket face in B is admitted only through the displayed predecessor filtration row, the adjacent martingale endpoints in X, the difference row in D, and the corresponding* $\mathsf{CondExp}^{\uparrow}$ *row in E. The accumulated row A may append only those admitted predictable faces, so a bracket consumer cannot replace the finite route by an almost-sure process equality, a completed-measure argument, a selected limiting variation, or ambient real completeness.*

Proof. Project the carrier of Definition 7525.1. The rows M, F, X, D, B, A, E are explicit coordinates of the accepted packet, and the predictable face is listed only beside the previous-time filtration data and the corresponding conditional-expectation row.

Use the martingale finite-prefix and finite-suffix restrictions from Definition 190.17 and Definition 190.20. They ensure the bracket read is made on an inspected finite time spine, with the same consumed $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ rows as the parent martingale packet.

Finally apply componentwise transport through H and replay through C. These rows preserve the displayed difference and predictable-face coordinates while adding no carrier slot for quotient process equality, completed measures, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7525.3 (Martingale quadratic-variation ledger)**.** *The local* $\mathsf{NameCert}_{\mathsf{MartingaleQuadraticVariation}^{\uparrow}}$ *surface for the carrier of Definition 7525.1 consists exactly of carrier admission for Q, predictable-face admission through* $\mathsf{CondExp}^{\uparrow}$*, finite accumulation of the bracket faces, componentwise classifier transport through H, displayed replay through C, package provenance through P, and local naming through N. No obligation row outside $M, F, X, D, B, A, E, H, C, P, N$ is part of the packet.*

Proof. Carrier admission is projection from Definition 7525.1. The martingale row, filtration row, endpoint family, difference family, bracket faces, accumulated ledger, conditional-expectation rows, transport, replay, provenance, and local name are the whole displayed tuple.

Predictable-face admission is supplied by Definition 7525.2. The same theorem fixes the finite accumulation boundary for A: every appended face first passes through the martingale prefix and the conditional-expectation route.

The structural rows H, C, P, N then name the packet without adding semantic coordinates. Therefore the NameCert ledger has no hidden almost-sure quotient equality row, no measure-completion row, no selected-limit row, and no ambient completeness row. □

Falsifiable prediction: Every $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$ bracket-facing read factors through a displayed finite martingale prefix, adjacent random-variable endpoint rows, finite difference rows, predictable $\mathsf{CondExp}^{\uparrow}$ faces, accumulated bracket ledger, componentwise transport, replay, provenance, and local naming.

Independence witness: martingale, randomvar, condexp, probspace, real: $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores martingale prefix data, difference rows, predictable conditional-expectation faces, an accumulated bracket ledger, structural transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleQuadraticVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{MartingaleQuadraticVariation}^{\uparrow}$ packet is generated from a finite Martingale row, filtration row, random-variable endpoints, difference rows, predictable bracket faces, accumulated bracket ledger, conditional-expectation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleQuadraticVariation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7525.1, Definition 7525.2, and Definition 7525.3 seed the finite predictable-bracket packet over martingale prefixes and conditional-expectation faces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MartingaleQuadraticVariation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure quotient equality, measure completion, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, predictable-face admission, finite accumulation, componentwise transport, replay, provenance, and local naming for BEDC.Derived.MartingaleQuadraticVariationUp.

Chapter 7526

A Concrete Naming Certificate for $\mathsf{PseudocompactSpace}^{\uparrow}$

Following Kelley and Engelking, the $\mathsf{PseudocompactSpace}^{\uparrow}$ packet records the finite BEDC surface for spaces whose displayed real-valued continuous reads carry boundedness ledgers. It sits between the topological source of chapter 88, the compactness comparison boundary of chapter 60, the real-valued map route of chapter 130, and the terminal real seal of chapter 38. The chapter names boundedness of displayed continuous real reads; it does not claim open-cover compactness, countable subcover extraction, or a host function-space theorem.

7526.1 Pseudocompact Carrier

Definition 7526.1 (Pseudocompact space finite packet)**.** *A* $\mathsf{PseudocompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, F, B, R, H, C, Q, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row naming the underlying open-set interface.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed real-valued continuous read.*
3. *B is the boundedness ledger for the read carried by F.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the lower and upper real bounds used by B.*
5. *H records componentwise* hsame *transport for the topological, continuous-map, boundedness, real-bound, provenance, and naming rows.*
6. *C records the displayed* Cont *replay from the topology row through the continuous real-valued read to the boundedness ledger.*
7. *Q is the* Pkg *provenance over* $\mathsf{PseudocompactSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PseudocompactSpace}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed rows. In particular, it has no coordinate for finite subcover choice, countable compactness, uniform continuity, a compact-open function space, quotient equality of maps, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

7526.2 Certificate Obligations

Theorem 7526.2 (Pseudocompact space NameCert obligations)**.** *For the carrier of Definition 7526.1, the local* $\mathsf{NameCert}_{\mathsf{PseudocompactSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the rows*

$$T, F, B, R, H, C, Q, N,$$

topology exposure through T*, continuous real-valued read exposure through* F*, boundedness-ledger exactness through* B*, real-bound sealing through* R*, componentwise transport through* H*, displayed replay through* C*, provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 7526.1. The packet lists exactly the topology, continuous-map, boundedness, real-bound, transport, replay, provenance, and naming rows.

The topology-facing coordinate is T, and the real-valued continuous read is exposed only through F. Thus a certificate reader cannot replace the displayed read by an arbitrary host function or by a compact-open function-space object.

The boundedness obligation is the row B sealed by the $\mathsf{Real}^{\uparrow}$ row R. Transport and replay through H and C preserve this route rowwise, while Q and N only record provenance and local naming.

Since the carrier contains no finite-subcover, countable-compactness, uniform-continuity, quotient-map, or ambient real-completeness coordinate, those objects are outside the local certificate surface. □

Theorem 7526.3 (Pseudocompact bounded continuous handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or boundedness-facing consumer of an accepted* $\mathsf{PseudocompactSpace}^{\uparrow}$ *carrier*

$$P = (T, F, B, R, H, C, Q, N)$$

must first expose the $\mathsf{Topology}^{\uparrow}$ *source row* T*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *real-valued read row* F*, and the boundedness ledger* B *before it may consume the real-bound seal* R*. The exported handoff is a finite bounded-continuous-read route over the same topology, continuous-map, boundedness, transport, replay, provenance, and local naming rows; it does not export open-cover compactness, a finite subcover selector, countable compactness, quotient equality of maps, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a boundedness-facing consumer of the accepted carrier. The only real-bound coordinate available to that consumer is the row R, and R is displayed as the seal for the boundedness ledger B.

Apply Definition 7526.2. It forces the topology source to be read through T, the real-valued continuous map through F, and the boundedness ledger through B before the real-bound seal can be consumed.

The replay and transport rows C and H keep this route inside the same finite packet, and the rows Q and N only source and name the route. Hence the handoff cannot introduce open-cover compactness, a finite subcover selector, countable compactness, quotient equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PseudocompactSpace}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PseudocompactSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7526.1, Definition 7526.2, and Definition 7526.3 seed the finite bounded-continuous-read packet over topology, continuous-map, boundedness, Real-bound, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PseudocompactSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, finite subcover extraction, countable compactness, compact-open function spaces, quotient equality of maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous real-valued read exposure, boundedness-ledger exactness, Real-bound sealing, transport stability, replay accountability, and non-escape for BEDC.Derived.PseudocompactSpaceUp.

Constructive story: A PseudocompactSpace$^{\uparrow}$ packet is generated from a topology source row, a continuous real-valued read row, a boundedness ledger, a RealUp bound seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7527

A Concrete Naming Certificate for $\mathsf{CauchyCompletionIntersection}^{\uparrow}$

Following Bishop and Bridges and Kelley-style completion theory, the $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ packet records the finite BEDC route by which two completion-facing Cauchy reads are compared through a shared window before any terminal real seal is consumed. It sits after the regular Cauchy completion route of chapter 1639, the completion monad law surface of chapter 3834, and the L10 readback spine through chapter 126, chapter 291, chapter 301, and chapter 38. The packet names finite intersection stability for completion routes; it is not a quotient completion equality, not a selected representative, not a countable-choice schedule, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7527.1 (CauchyCompletionIntersection carrier)**.** *A* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the first completion-facing Cauchy route, including its source name, regularity request, and completion consumer boundary.*
2. *B is the second completion-facing Cauchy route, with the same kind of source, regularity, and consumer data.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row through which both routes must expose finite observations before either route reaches a completion seal.*
4. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radius, slack, comparison budget, and refinement data used by both window reads.*
5. *R is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after A and B have both passed through W and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the two displayed routes have reached the same finite window, tolerance, and regular-rational readback surface.*
7. *H records componentwise* hsame *transport for $A, B, W, D, R, E, C, P, N$.*
8. *C records displayed* Cont *replay for the route order $A, B, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionIntersection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, B, W, D, R, E, H, C, P, N$. No coordinate stores quotient completion classes, selected stream representatives, a countable-choice selector, detached host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7527.2 (CauchyCompletionIntersection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *carrier $I = (A, B, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionIntersection}}$ *surface consists exactly of carrier admission for the two completion-facing Cauchy routes, shared* $\mathsf{StreamName}^{\uparrow}$ *window exposure, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure, shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every consumer of the packet must read A and B before opening the shared window row W, must read D before comparing regular-rational readbacks through R, and must reach the terminal seal E only after the same finite rows have been displayed for both routes.

Proof. Project the accepted carrier of Definition 7527.1. The two completion-facing Cauchy routes are exactly A and B, and the shared analytic readback surface is exactly W, D, R, E.

The window and tolerance obligations are forced by the carrier order. A consumer must first display both Cauchy routes, then open the common $\mathsf{StreamName}^{\uparrow}$ window and the shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before any $\mathsf{RegSeqRat}^{\uparrow}$ readback can be compared.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since there is no coordinate for quotient classes, selected representatives, countable choice, host equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7527.3 (CauchyCompletionIntersection shared-window route)**.** *Let $I = (A, B, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *packet. Any completion-facing read that claims the two Cauchy routes meet must factor through the finite route*

$$A, B \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the intersection is witnessed by shared StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the Real seal, not by quotienting the two routes or selecting a representative completion point.

Proof. Start with the two route rows A and B. They are separate coordinates of the carrier, so neither one can be erased or replaced by the other before the shared readback surface is opened.

The common window row W and tolerance ledger D provide the finite comparison surface. The regular-rational row R is read only after both routes have passed through those shared rows, and the real seal E is downstream of that readback.

Finally the transport, replay, provenance, and naming rows preserve the displayed order. The carrier therefore supplies a finite shared-window route and no quotient, selected representative, choice schedule, host equality, or ambient-completeness route. □

Theorem 7527.4 (CauchyCompletionIntersection nonescape)**.** *Every accepted* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal has already consumed the finite packet rows*

$$A, B, W, D, R, E, H, C, P, N.$$

The resulting read cannot export quotient completion equality, a selected completion representative, countable choice, detached host equality of completed points, a closed-subspace theorem, a metric-intersection theorem, Lean verification, or bridge checking.

Proof. Apply the NameCert obligations of Definition 7527.2. They force every accepted consumer through the two Cauchy routes, shared window, tolerance ledger, regular-rational readback, real seal, and structural rows.

Use the shared-window route of Definition 7527.3. It identifies the only path to the terminal $\mathsf{Real}^{\uparrow}$ seal as the finite route through $A, B, W, D, R, E, H, C, P, N$.

The excluded conclusions require coordinates not present in the carrier. Since the packet stores no quotient class, selected representative, choice schedule, host equality, closed-subspace certificate, metric-intersection certificate, verification credential, or bridge-checked row, none of those conclusions can escape from the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ packet is generated from two completion-facing Cauchy routes, shared StreamName windows, a shared DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionIntersection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7527.1, Definition 7527.2, Definition 7527.3, and Definition 7527.4 seed the finite intersection route for two completion-facing Cauchy reads over shared StreamName, DyadicRatCore, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionIntersection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected representatives, countable choice, host equality of completed points, closed-subspace closure, metric-intersection closure, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exposure, shared-tolerance exposure, regular-rational readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionInters

Chapter 7528

A Concrete Naming Certificate for $\mathsf{CauchyBornologicalUniformity}^{\uparrow}$

Following Bourbaki on uniform structures and bornologies, and Bishop–Bridges on constructive Cauchy data, the $\mathsf{CauchyBornologicalUniformity}^{\uparrow}$ packet records the finite interface where Cauchy filters, bounded sets, metric-bornological control, and uniform entourages are read as one NameCert surface. It sits after the Cauchy-filter carrier of chapter 333, the bornology carrier of chapter 3370, the metric-bornology comparison of chapter 4091, the uniform-space entourage discipline of chapter 339, and the complete-metric handoff of chapter 128. The packet gives Real and Completion consumers a bounded-uniform Cauchy interface without quotient completions, selected limits, countable choice, trichotomy, or ambient completeness outside the displayed rows.

Definition 7528.1 (Cauchy bornological uniformity carrier)**.** *A* $\mathsf{CauchyBornologicalUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, B, M, U, C, H, R, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *request row, including the finite filter-base and Cauchy-eventuality data consumed from chapter 333;*
2. *B is the* $\mathsf{Bornology}^{\uparrow}$ *bounded-family row, recording the bounded subsets or bounded events that may be inspected by the filter;*
3. *M is the* ***??*** *comparison row, tying metric balls and boundedness requests to the same displayed source;*
4. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, carrying the finite entourage requests through which Cauchy behavior is read;*
5. *C is the compatibility row asserting that every bounded Cauchy-filter read requested by F remains contained in the bornological row B, is measured by M, and is stable under the entourage route U;*
6. *H stores componentwise* hsame *transport for F, B, M, U, C, replay, provenance, and local naming rows;*
7. *R records the displayed* Cont *replay routes used by bounded-filter, bornological-containment, metric-bornology, and entourage-stability consumers;*

8. *P is the* Pkg *provenance over* CauchyFilter$^{\uparrow}$, Bornology$^{\uparrow}$, *??*, UniformSpace$^{\uparrow}$, CompleteMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

9. *N is the local* NameCert$_{\text{CauchyBornologicalUniformity}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completion, selected Cauchy limit, countable-choice schedule, trichotomy over distances, host equality of filters, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyBornologicalUniformityUp.TasteGate.CauchyBornologicalUniformityTasteGate single carrier alignment`

Theorem 7528.2 (Cauchy bornological uniformity NameCert obligations)**.** *For every accepted* CauchyBornologicalUniformity$^{\uparrow}$ *carrier*

$$Q = (F, B, M, U, C, H, R, P, N),$$

the local NameCert$_{\text{CauchyBornologicalUniformity}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded Cauchy-filter compatibility, bornological containment, metric-bornology comparison, uniform entourage stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer that asks for a bounded-uniform Cauchy handoff must read these rows before it can consume a completion-facing seal.*

Proof. Carrier admission is projection from Definition 7528.1. The accepted packet exposes exactly $F, B, M, U, C, H, R, P, N$, so the Cauchy-filter, bornology, metric-bornology, uniform-space, compatibility, transport, replay, provenance, and naming rows are all finite BEDC data.

Bounded Cauchy-filter compatibility is read from F together with C. The Cauchy-filter row supplies the finite eventuality requests, while the compatibility row records that these requests remain bounded by B and measurable through M before any completion consumer can read them.

Bornological containment and uniform entourage stability are read from B, U, C, and R. The bornology row supplies the bounded family, the uniform row supplies entourage requests, and the replay row preserves the displayed route through bounded-filter and entourage consumers.

Finally inspect H, P, N. Componentwise transport, provenance, and local naming preserve the same finite packet. Since no coordinate stores a quotient completion, selected limit, countable-choice schedule, trichotomy row, host filter equality, or ambient completeness theorem, the NameCert obligations are exhausted by the displayed bounded-uniform Cauchy ledger. □

Falsifiable prediction: Every accepted CauchyBornologicalUniformity$^{\uparrow}$ consumer displays a CauchyFilter$^{\uparrow}$ row, a Bornology$^{\uparrow}$ row, a **??** row, a UniformSpace$^{\uparrow}$ row, a bounded-compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any completion-facing seal is consumed.

Independence witness: cauchyfilter, bornology, metric-bornology, uniformspace, completemetric: CauchyBornologicalUniformity$^{\uparrow}$ is not bijective to any listed sibling because it packages bounded Cauchy-filter compatibility, bornological containment, metric-bornology comparison, and uniform entourage stability as one finite NameCert surface.

Closure status for NameCert$_{\text{CauchyBornologicalUniformity}^{\uparrow}}$.
Theory closure: TheoryGate(NameCert$_{\text{CauchyBornologicalUniformity}^{\uparrow}}$, seed)
Scope closed: Definition 7528.1 and Definition 7528.2 seed the bounded-uniform Cauchy interface over CauchyFilter$^{\uparrow}$, Bornology$^{\uparrow}$, **??**, UniformSpace$^{\uparrow}$, CompleteMetric$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.
Formal verification: FormalStatus(NameCert$_{\text{CauchyBornologicalUniformity}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human

Not claimed: This chapter does not close Lean verification, quotient completion, selected limits, countable choice, trichotomy, host equality of filters, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Cauchy-filter compatibility, bornological containment, metric-bornology comparison, uniform entourage stability, transport replay, and ledger non-escape for BEDC.Derived.CauchyBornologicalUniformityUp.

Constructive story: A CauchyBornologicalUniformity$^{\uparrow}$ packet is generated from Cauchy-filter requests, bornological bounded-family rows, metric-bornology comparisons, uniform entourage rows, bounded compatibility, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Chapter 7529

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ packet records the finite handoff by which apartness data already exposed over regular Cauchy names are transported into later real-name and completion-facing consumers. It sits over the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38; it also consumes the order boundary of chapter 1710, the reflector of chapter 6139, the located-apartness route of chapter 6730, and the finite budget row of chapter 991. The packet is a constructive transfer surface for apartness reads, not a trichotomy principle, quotient-real equality test, selected-limit operation, countable-choice schedule, or ambient completeness assertion.

Definition 7529.1 (Regular Cauchy apartness transfer carrier)**.** *A* $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, O, B, F, L, R, W, D, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose apartness relation is being transferred.*
2. *O is the* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *comparison row that supplies the order-facing apartness read.*
3. *B is the* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *positive lower-bound budget required before any strict or apartness-facing read is public.*
4. *F is the* $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ *row that reflects the order read into an inspected apartness surface.*
5. *L is the* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *row that keeps the apartness read located over displayed regular Cauchy data.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational and apartness rows have been displayed.*
7. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the transfer.*
8. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and lower-bound ledgers attached to those windows.*

9. *H stores componentwise* hsame *transport for the source, order, budget, reflector, located-apartness, seal, window, and dyadic rows.*
10. *C stores displayed* Cont *replay for the same transfer route.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$, $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$, RegularCauchyLocated $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, quotient real equality, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7529.2 (Regular Cauchy apartness transfer NameCert obligations)**.** *For the carrier of Definition 7529.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for T, source admission for X, Y, apartness-order admission through O, positive-budget admission through B, reflector admission through F, located-apartness admission through L, terminal real-seal handoff through R, finite window and dyadic ledger admission through W, D, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 7529.1. The listed rows are exactly the available coordinates of T.

The apartness-facing obligations are inherited from the sibling packets. The order row O supplies the comparison surface, the budget row B supplies the positive lower-bound evidence, the reflector row F exposes the inspected apartness read, and the located row L keeps the read attached to displayed regular Cauchy data.

The structural rows W, D, H, C, P, N keep the same finite route under window inspection, dyadic readback, transport, replay, provenance, and local naming. Since no further coordinate is present, the NameCert surface is exactly the displayed transfer packet. □

Theorem 7529.3 (Regular Cauchy apartness transfer real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ *packet reaches the terminal seal R only after the regular rational sources X, Y, the apartness-order row O, the budget row B, the reflector row F, the located-apartness row L, the finite stream windows W, and the dyadic ledger D have been displayed. The handoff exports a real-facing apartness read, but no quotient equality, selected representative, trichotomy decision, countable-choice schedule, or ambient completeness row.*

Proof. Start from the NameCert obligations of Definition 7529.2. They place R downstream of the source, apartness, budget, reflector, located, window, and dyadic rows.

Use the sibling order and budget boundaries. The $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ and RegularCauchyApartnessBu rows force the positive apartness read through displayed finite lower-bound data before any real-facing seal is public.

Finally use the reflector and located-apartness rows. They preserve the inspected apartness route into the real-facing handoff, while the absent objects are not coordinates of T. □

Theorem 7529.4 (Regular Cauchy apartness transfer transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ *carrier preserves the source rows, apartness-order row, budget row, reflector row, located-apartness row, terminal seal, finite windows, dyadic ledger, replay, provenance, and local naming together. A transported packet therefore carries the same apartness-transfer route and cannot detach the reflected apartness read from its displayed budget, window, dyadic, or real-seal boundary.*

Proof. Project the transport row H from the carrier. It is componentwise over every displayed coordinate used by the transfer.

Apply the NameCert obligation theorem. The order, budget, reflector, located-apartness, window, dyadic, and seal rows are admitted as one packet surface, so a transport of the packet transports those rows together.

The replay and provenance rows preserve the same route for downstream consumers. Thus transport may rename the finite packet but cannot split the apartness read from its displayed constructive witnesses. □

Theorem 7529.5 (Regular Cauchy apartness transfer non-escape)**.** *Every downstream read of an accepted* $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ *packet factors through*

$$X, Y, O, B, F, L, R, W, D, H, C, P, N.$$

It may inspect regular rational sources, stream windows, dyadic lower-bound data, apartness-order evidence, the apartness budget, reflector and located-apartness rows, the terminal real seal, transport, replay, provenance, and the local NameCert *row. It cannot inspect trichotomy, quotient real equality, selected limits, countable choice, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier definition and the NameCert obligations. They enumerate all rows available to a consumer of the accepted packet.

Apply the real-seal handoff and transport theorems. The handoff keeps the real-facing read after the displayed apartness route, while transport preserves the same finite route component by component.

The excluded principles are absent from the carrier inventory and from the sibling apartness rows used by the transfer. Therefore no downstream consumer can extract them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}$ packet is generated from regular rational sources, apartness-order evidence, a positive apartness budget, reflector and located-apartness rows, a terminal RealUp seal, finite StreamName windows, DyadicRatCore ledgers, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7529.1, Definition 7529.2, Definition 7529.3, Definition 7529.4, and Definition 7529.5 describe the finite apartness-transfer carrier and its real-seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessTransfer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-order admission, budget admission, reflector and located-apartness handoff, transport, nonescape, and real-seal handoff for the displayed transfer packet.

Chapter 7530

A Concrete Naming Certificate for $\mathsf{ShadowingLemma}^{\uparrow}$

Following Bowen's hyperbolic dynamics and Walters' formulation of shadowing for expansive systems, the $\mathsf{ShadowingLemma}^{\uparrow}$ packet isolates the finite pseudo-orbit data needed to read a shadowing orbit inside BEDC. It sits downstream of chapter 6707, chapter 195, chapter 57, chapter 38, chapter 126, and chapter 301. The packet records a local certificate route; it does not assert a global hyperbolic decomposition, expansivity theorem, invariant measure, selected orbit, or host-level completeness principle.

Definition 7530.1 (ShadowingLemma carrier)**.** *A* $\mathsf{ShadowingLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, T, W, E, Q, M, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{DynSystem}^{\uparrow}$ *row naming the finite system interface and the branch of the orbit map used by the certificate.*
2. *T is the* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *dependency row that names the hyperbolic-shadowing sibling packet consumed by the local route.*
3. *W is the pseudo-orbit window schedule, read as finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *E is the error-modulus row assigning the displayed tolerance budget to each pseudo-orbit window.*
5. *Q is the pseudo-orbit row whose adjacent iterates are compared only through W and E.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row measuring the pseudo-orbit errors and the shadowing errors in the same displayed metric interface.*
7. *S is the shadowing-orbit row, storing the orbit read that shadows Q over the windows W under the modulus E.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row converting the window and error data into regular rational approximants.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the metric-error bounds read from M and R.*
10. *H records componentwise* hsame *transport for the system, dependency, window, modulus, pseudo-orbit, metric comparison, shadowing orbit, readback, and Real-seal rows.*

11. *C records displayed* Cont *replay from a shadowing query through $X, T, W, E, Q, M, S, R, A, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{ShadowingLemma}^{\uparrow}$, $\mathsf{HyperbolicShadowing}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ShadowingLemma}^{\uparrow}}$ *row.*

The carrier stores no selected host orbit, no ambient compactness theorem, no global stability theorem, no real-number trichotomy, and no quotient equality of orbit names.

Theorem 7530.2 (ShadowingLemma NameCert obligations)**.** *For every accepted* $\mathsf{ShadowingLemma}^{\uparrow}$ *carrier*

$$L = (X, T, W, E, Q, M, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ShadowingLemma}^{\uparrow}}$ *surface consists exactly of system admission through X, hyperbolic-shadowing dependency through T, pseudo-orbit windows through W, error-modulus exposure through E, pseudo-orbit readback through Q, metric comparison through M, shadowing-orbit exposure through S, regular-rational readback through R, terminal Real seal through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7530.1. Its dynamical rows are exactly $X, T, W, E, Q, M, S, R, A$, and its structural rows are exactly H, C, P, N.

The pseudo-orbit part is finite. The window row W determines which adjacent iterates can be inspected, and the modulus row E determines which error budget is visible on each window.

The shadowing part is then read through M, S, R, A. The metric row compares pseudo-orbit and shadowing-orbit errors, the regular-rational row gives the finite readback, and the Real row seals the resulting bounds.

Finally H, C, P, N transport, replay, source, and name the same finite data. Since no omitted global theorem is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7530.3 (ShadowingLemma pseudo-orbit handoff)**.** *Every consumer of an accepted* $\mathsf{ShadowingLemma}^{\uparrow}$ *packet must expose the* $\mathsf{DynSystem}^{\uparrow}$ *row X, the* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *sibling row T, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the error-modulus row E, the pseudo-orbit row Q, and the* $\mathsf{Met}^{\uparrow}$ *comparison row M before reading the shadowing orbit row S. The resulting readback must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the* $\mathsf{Real}^{\uparrow}$ *seal A, with* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows preserving the same route.*

Proof. Begin with the obligation surface of Definition 7530.2. It makes X, T, W, E, Q, M visible before the shadowing row can be consumed.

The sibling dependencies keep the route local. The $\mathsf{DynSystem}^{\uparrow}$ row supplies the finite system interface, the $\mathsf{HyperbolicShadowing}^{\uparrow}$ row supplies the traditional shadowing dependency, and the $\mathsf{Met}^{\uparrow}$ row supplies the only comparison surface for the pseudo-orbit and shadowing-orbit errors.

The readback is then forced through R, A, H, C, P, N. The regular-rational row records the finite approximants, the Real row seals the displayed bounds, and the transport, replay, provenance, and naming rows preserve the same packet. Thus the handoff is a finite BEDC route rather than a selected-orbit or ambient-completeness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{ShadowingLemma}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ShadowingLemma}^{\uparrow}}$, seed)

Scope closed: Definition 7530.1, Definition 7530.2, and Definition 7530.3 seed the finite pseudo-orbit, error-modulus, metric comparison, shadowing-orbit, regular-rational readback, Real-seal, transport, replay, provenance, and local naming route.

Formal verification: FormalStatus(NameCert$_{\text{ShadowingLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global hyperbolic decomposition, expansivity, invariant measures, selected host orbits, quotient equality of orbit names, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked system admission, hyperbolic-shadowing dependency, pseudo-orbit window exactness, error-modulus exactness, Metric comparison, StreamName and RegSeqRat readback, Real seal, transport, replay, provenance, and local naming for BEDC.Derived.ShadowingLemmaUp.

Constructive story: A ShadowingLemma$^{\uparrow}$ packet is generated from a DynSystem row, HyperbolicShadowing dependency row, StreamName pseudo-orbit windows, an error-modulus row, a pseudo-orbit row, a Metric comparison row, a shadowing-orbit row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7531

A Concrete Naming Certificate for $\mathsf{KatetovExtension}^{\uparrow}$

Following Katetov and Urysohn on finite metric one-point extensions, the $\mathsf{KatetovExtension}^{\uparrow}$ packet records the standalone finite extension certificate that L10 metric and completion consumers can read before any host Urysohn space is named. It sits over the metric carrier of chapter 57, the separable-metric support schedule of chapter 1458, the isometric handoff surface of chapter 439, the completion-facing route of chapter 1506, the terminal real-distance seal of chapter 38, and the Urysohn finite-extension consumer of chapter 6554. The packet exposes a finite support, Real-valued triangle and reverse-triangle budgets, an isometric handoff, and completion-order obligations without claiming a host Urysohn-space existence theorem or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7531.1 (Katetov extension carrier)**.** *A* $\mathsf{KatetovExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, S, R, F, A, I, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose finite support distances are inspected.*
2. *S is the* $\mathsf{SeparableMetric}^{\uparrow}$ *or finite support schedule row naming exactly the support points used by the one-point request.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row for the support distances and the proposed distances from the new point to each support point.*
4. *F is the finite Katetov budget row. It lists, for every displayed support pair, the triangle and reverse-triangle inequalities that the proposed distances must satisfy.*
5. *A is the admission ledger row recording that the inequalities in F have been checked against the same M, S, R surface.*
6. *I is the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *handoff row. It exports the old support points with their displayed distances unchanged before a consumer reads the new-point distances.*
7. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *order row. It permits completion-facing consumers to read the extension request only after M, S, R, F, A, I have been exposed.*
8. *H records componentwise* hsame *transport for the metric source, support schedule, real-distance seal, Katetov budget, admission ledger, isometric handoff, completion order, replay, provenance, and local name.*

9. *C records displayed* Cont *replay for the same finite one-point extension route.*

10. *P is the* Pkg *provenance over* $\mathsf{KatetovExtension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *finite Katetov requests,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{KatetovExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, R, F, A, I, L, H, C, P, N$. It has no coordinate for a host Urysohn space, an infinite extension chain, quotient completion equality, a selected completion point, countable choice, Lean verification, a standard bridge, or ambient real completeness.

Theorem 7531.2 (Katetov extension NameCert obligations)**.** *For every accepted* $\mathsf{KatetovExtension}^{\uparrow}$ *carrier*

$$K = (M, S, R, F, A, I, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KatetovExtension}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the metric source, support schedule, real-distance seal, finite Katetov budget, admission ledger, isometric handoff, completion order, transport, replay, provenance, and local name;*

2. *common-source discipline, requiring S, R, F, A, I, L to be read over the same metric source row M;*

3. *finite support visibility, requiring every requested new distance to attach to a support point named by S;*

4. *Real-valued Katetov exposure, requiring F and A to display each triangle and reverse-triangle budget through the same* $\mathsf{Real}^{\uparrow}$ *seal row R;*

5. *isometric and completion order, requiring I to preserve the old support distances before L may be read by a completion-facing consumer;*

6. *nonescape, excluding host Urysohn spaces, infinite extension chains, quotient completion equality, selected completion points, countable choice, Lean verification, bridge credentials, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Katetov extension obligation is stored outside these rows.

Proof. Project the carrier of Definition 7531.1. The tuple $M, S, R, F, A, I, L, H, C, P, N$ is the whole public surface, so carrier admission is exactly admission of those rows.

The common-source and finite-support obligations follow from M and S. Every proposed new distance is read through a support name displayed by S and through the same metric source row M.

The Katetov inequalities are recorded by F and A. The budget row lists the triangle and reverse-triangle comparisons, while the admission ledger records that each comparison is checked through the displayed real-distance seal R.

Finally I, L, H, C, P, N preserve only that displayed finite route. The old support distances are exported through the isometric handoff before the completion-order row is available, and no carrier coordinate stores a host Urysohn space, quotient completion equality, selected completion point, infinite extension chain, choice functional, Lean credential, bridge credential, or ambient completeness theorem. □

Theorem 7531.3 (Katetov extension isometric handoff)**.** *Every completion-facing or Urysohn-facing consumer of an accepted* $\mathsf{KatetovExtension}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{SeparableMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto F, A \longmapsto \mathsf{IsometricEmbedding}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

Thus the packet exports a displayed finite one-point metric-extension certificate: the old support is handed off isometrically, the new distances satisfy the displayed Real-valued Katetov budgets, and completion-facing reads occur only after those rows have been exposed.

Proof. Begin with the metric source row M and the support schedule S. These rows expose the only old points and support distances available to the extension request.

Read the real-distance row R, then the finite Katetov rows F and A. They display the proposed new distances and the triangle and reverse-triangle budgets over the same finite support.

The isometric row I then exports the old support without changing its displayed distances. Only after this handoff may the completion-order row L be read.

Consequently any $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives a finite one-point extension route, not a host universal space, quotient completion construction, selected limit point, choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{KatetovExtension}^{\uparrow}$ read exposes the $\mathsf{Met}^{\uparrow}$ source row, finite $\mathsf{SeparableMetric}^{\uparrow}$ support schedule, $\mathsf{Real}^{\uparrow}$ distance seal, Katetov budget, admission ledger, and $\mathsf{IsometricEmbedding}^{\uparrow}$ handoff before any $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{UrysohnUniversalMetric}^{\uparrow}$ consumer may inspect the completion-order row.

Independence witness: metric, separablemetric, isometric-embedding, metriccompletion, real, urysohnuniversalmetric: $\mathsf{KatetovExtension}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite support schedules, Real-valued one-point distance requests, triangle and reverse-triangle budgets, admission ledgers, isometric handoffs, completion-order rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows while refusing host Urysohn spaces and completion quotients.

Closure status for $\mathsf{NameCert}_{\mathsf{KatetovExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{KatetovExtension}^{\uparrow}$ packet is generated from a displayed metric source, finite support schedule, Real-valued distance seal, finite Katetov budget, admission ledger, isometric handoff, completion-order row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KatetovExtension}^{\uparrow}}$, seed)

Scope closed: Definition 7531.1, Definition 7531.2, and Definition 7531.3 seed the finite one-point metric-extension certificate for L10 metric and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KatetovExtension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Urysohn-space existence, quotient completion equality, countable choice, infinite extension chains, selected completion points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-source discipline, finite support visibility, Real-valued Katetov exposure, isometric handoff order, completion order, and nonescape for BEDC.Derived.KatetovExtensionUp.

Chapter 7532

A Concrete Naming Certificate for $\mathsf{CofinalTailMesh}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, the $\mathsf{CofinalTailMesh}^{\uparrow}$ packet records the shared tail-window mesh used when several regular rational names are read before a Real or Completion consumer may inspect a terminal seal. It sits over the finite stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the regular Cauchy tail-meet route of chapter 560, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and completion-facing Cauchy routes. The packet names a common cofinal window surface; it does not quotient streams, select limits, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7532.1 (CofinalTailMesh carrier)**.** *A* $\mathsf{CofinalTailMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, M_0, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule family naming the finite windows inspected for every participating regular Cauchy name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every mesh comparison.*
4. *M_0 is the regular Cauchy tail-meet row, recording the common cofinal tail at which the named reads are compared.*
5. *W is the shared cofinal window row consumed before any* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal is read.*
6. *H records componentwise* hsame *transport for S, R, D, M_0, W and for the structural rows.*
7. *C records displayed* Cont *replay for opening the stream windows, reading regular rational tails, checking dyadic tolerances, and handing the shared window to downstream consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalTailMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *regular Cauchy completion routes,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, R, D, M_0, W, H, C, P, N$. No coordinate stores quotient stream equality, a selected limit, a host cofinal subsequence, countable choice, or ambient completeness.

Theorem 7532.2 (CofinalTailMesh NameCert obligations)**.** *For every accepted* $\mathsf{CofinalTailMesh}^{\uparrow}$ *carrier* $M = (S, R, D, M_0, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalTailMesh}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window scheduling, RegSeqRat readback, DyadicRatCore tolerance comparison, regular Cauchy tail-meet admission, shared cofinal window handoff, componentwise transport, displayed replay, provenance, and local naming. Every consumer must read the stream schedules and dyadic ledger before it can consume the shared window row.*

Proof. Project the carrier of Definition 7532.1. The visible inventory is $S, R, D, M_0, W, H, C, P, N$, and each row has a fixed role in the mesh.

The stream and rational rows are read first. The schedule row S opens the finite windows, while R supplies regular rational readback for those same windows.

The dyadic and tail-meet rows then bind the comparison. The row D supplies the tolerance ledger, and M_0 records the common regular Cauchy tail at which the participating reads are compared before W exposes the shared cofinal window.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since quotient streams, selected limits, host cofinal subsequences, countable choice, and ambient completeness have no carrier coordinate, they are not NameCert obligations of the packet. □

Theorem 7532.3 (CofinalTailMesh Real-completion route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CofinalTailMesh}^{\uparrow}$ *carrier receives the finite route*

$$S \longrightarrow R \longrightarrow D \longrightarrow M_0 \longrightarrow W \longrightarrow H, C, P, N.$$

Thus the consumer reads finite StreamName windows, RegSeqRat tails, a DyadicRatCore tolerance ledger, a regular Cauchy tail meet, and one shared cofinal window before any terminal $\mathsf{Real}^{\uparrow}$ *or completion seal can be consumed.*

Proof. Start from carrier admission in Definition 7532.1. The only completion-facing row is the shared window W, and it is stored after the stream schedule, regular rational readback, dyadic ledger, and tail-meet rows.

Apply Definition 7532.2. It requires the stream schedules and dyadic ledger to be read before the shared window handoff, so a Real-facing read cannot bypass S, R, D, M_0.

The replay row C and transport row H preserve the same route for downstream consumers, while P and N name the provenance and local certificate. Therefore the route exports a finite mesh and not a quotient completion, selected representative, countable-choice tail, or ambient complete-space witness. □

Theorem 7532.4 (CofinalTailMesh nonescape)**.** *Any accepted* $\mathsf{CofinalTailMesh}^{\uparrow}$ *read is exhausted by* $S, R, D, M_0, W, H, C, P, N$*. It cannot produce quotient equality of streams, a selected regular Cauchy representative, a global cofinal subsequence, countable choice, a completed host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition gives the complete coordinate list. A consumer may inspect the finite stream schedule, regular rational readback, dyadic tolerance ledger, regular Cauchy tail meet, shared cofinal window, transport, replay, provenance, and local naming rows.

The Real-completion route of Definition 7532.3 keeps every value-facing handoff inside that list. Its only terminal-facing row is W, and W is reached through S, R, D, M_0.

The excluded objects would require additional carrier coordinates or an external selection principle. Since neither appears in $M = (S, R, D, M_0, W, H, C, P, N)$, the accepted read remains within the finite NameCert packet. □

Falsifiable prediction: Every accepted $\mathsf{CofinalTailMesh}^{\uparrow}$ read factors through displayed StreamName schedules, RegSeqRat readback, a DyadicRatCore tolerance ledger, a regular Cauchy tail meet, a shared cofinal window, componentwise transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-tail-meet: $\mathsf{CofinalTailMesh}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the common cofinal tail-window mesh consumed before Real and Completion routes inspect a seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalTailMesh}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readback, a DyadicRatCore tolerance ledger, a regular Cauchy tail meet, a shared cofinal window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7532.1, Definition 7532.2, Definition 7532.3, and Definition 7532.4 seed the finite shared-tail mesh for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, global cofinal subsequences, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window admission, dyadic tolerance comparison, Real-completion routing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.CofinalTailMeshUp.

Chapter 7533

A Concrete Naming Certificate for $\mathsf{BlackBoxFiberWitness}^{\uparrow}$

The $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ packet realizes the black-box decomposition surface of section 9462.18 as finite NameCert data that keeps visible behavior separate from hidden-fiber opacity. It sits beside classifier-forming learning in chapter 6097, the hard-certificate layer of section 9462.5, the interpretability classifier-novelty route of section 9462.11, formal-verification certified interpretability in section 9462.17, and the certificate-preserving discipline of chapter 481. The packet names a finite witness for behavior support, missing certificate rows, hidden-fiber refusal, and ledger provenance; it is not a semantic oracle for the model internals.

Falsifiable prediction: Deleting the missing-certificate row or identifying the hidden fiber with the visible behavior refutes the $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ carrier.

Independence witness: classifier-forming-learning, hard-certificate-layer, interpretability-classifier-novelty, formal-verification-certified-interpretability, namecert: none is bijective to $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ because this packet simultaneously carries visible behavior, support-candidate, missing-certificate, hidden-fiber, ledger, transport, replay, provenance, and local NameCert rows.

Definition 7533.1 (BlackBoxFiberWitness carrier)**.** *A* $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, S, M, K, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the visible behavior row read from samples, outputs, traces, failure cases, or an observation protocol.*
2. *S is the support-candidate row naming the classifier candidate that supports the behavior.*
3. *M is the missing-certificate row, recording which source, pattern, classifier, stability, or ledger-policy fields are absent or incomplete.*
4. *K is the hidden-fiber row, kept distinct from the visible behavior row and used only as an opacity boundary.*
5. *L is the ledger row recording present evidence, missing fields, and debt for the target (B, S).*
6. *H records componentwise* hsame *transport for the visible behavior, support candidate, missing-certificate, hidden-fiber, ledger, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay for opening behavior evidence, checking support, preserving missing-field data, and refusing hidden-fiber identification.*
8. *P is the* Pkg *provenance over* $\mathsf{BlackBoxFiberWitness}^{\uparrow}$*, classifier-forming learning, hard-certificate layers, interpretability classifier novelty, formal-verification certified interpretability,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BlackBoxFiberWitness}^{\uparrow}}$ *row.*

The classifier compares packets only through $B, S, M, K, L, H, C, P, N$. No coordinate identifies the hidden fiber with visible behavior, supplies a complete explanation, or imports model-internal semantic truth.

Theorem 7533.2 (BlackBoxFiberWitness NameCert obligations)**.** *For every accepted* $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ *carrier $F = (B, S, M, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BlackBoxFiberWitness}^{\uparrow}}$ *surface consists exactly of visible behavior admission, support-candidate admission, missing-certificate preservation, hidden-fiber separation, ledger accounting, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7533.1. The displayed rows are $B, S, M, K, L, H, C, P, N$, with B and K occupying separate coordinates.

The support and certificate rows are read next. The row S records the classifier candidate supporting B, while M records the certificate fields that remain absent or incomplete.

The ledger row L binds present evidence, missing fields, and debt for the same target. Transport through H and replay through C preserve those distinctions, and P, N only source and name the finite packet. □

Theorem 7533.3 (BlackBoxFiberWitness visible-hidden route)**.** *Every accepted* $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ *read reaches the hidden-fiber boundary only through the finite route*

$$B \longrightarrow S \longrightarrow M \longrightarrow K \longrightarrow L \longrightarrow H, C, P, N.$$

The visible behavior row B and the hidden-fiber row K remain distinct throughout the route, and the support candidate may be consumed only together with the missing-certificate row and ledger row.

Proof. Start with carrier projection. The visible behavior B, support candidate S, missing-certificate row M, hidden-fiber row K, and ledger row L are separate fields.

Use Definition 7533.2. It preserves missing-certificate data and hidden-fiber separation as explicit NameCert obligations, so a consumer cannot turn support evidence into a closed explanation.

Finally replay the route through H, C, P, N. These rows transport and name the same finite fields, not a semantic identification between visible behavior and hidden model fiber. □

Theorem 7533.4 (BlackBoxFiberWitness nonescape)**.** *No accepted* $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ *packet exports a hidden model-internal truth, a complete interpretability certificate, an identification of hidden fiber with visible behavior, or a verification-hardening proof. It exports only the finite rows $B, S, M, K, L, H, C, P, N$ and the visible-hidden route stated above.*

Proof. The carrier of Definition 7533.1 lists the entire accepted inventory. Hidden-fiber opacity is represented by the row K, while certificate debt is represented by M and L.

Apply the visible-hidden route of Definition 7533.3. Every consumer reaches K only after B, S, M and before the structural rows, so the packet cannot bypass missing-certificate data.

The hard-certificate and verification siblings may later harden a certificate, and classifier-novelty siblings may compare explanation states. This packet itself supplies neither a complete certificate nor hidden semantic truth, because those objects are absent from $F = (B, S, M, K, L, H, C, P, N)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{BlackBoxFiberWitness}\uparrow}$.

Constructive story: A $\mathsf{BlackBoxFiberWitness}^{\uparrow}$ packet is generated from visible behavior, support-candidate, missing-certificate, hidden-fiber, ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BlackBoxFiberWitness}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7533.1, Definition 7533.2, Definition 7533.3, and Definition 7533.4 seed the finite visible-hidden black-box witness packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BlackBoxFiberWitness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hidden model semantics, complete interpretability, verification hardening, classifier novelty, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, missing-certificate preservation, hidden-fiber separation, visible-hidden routing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BlackBoxFiberWitnessUp.

Chapter 7534

A Concrete Naming Certificate for $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$

Following Busemann and Bridson–Haefliger on boundary functions in hyperbolic geometry, the $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ packet realizes the scalar readback surface requested by chapter 9463. It sits after the hyperbolic metric carrier of chapter 3474, the boundary-point packet of chapter 3732, the additive cocycle packet of chapter 4134, the horocycle packet of chapter 3534, and the terminal real seal of chapter 38. The packet names a finite boundary-point, geodesic-ray, metric-difference, Real-seal, and horosphere-readback route; it does not complete the ideal boundary, quotient geodesic rays, choose a global ray representative, or import an ambient curvature model.

Definition 7534.1 (Hyperbolic Busemann function carrier)**.** *A* $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, \Xi, G, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row for the visible disk, geodesic, or metric source.*
2. *Ξ is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-point row toward which the function is read.*
3. *G is the geodesic-ray row. It records the displayed ray window used to compare the point of evaluation with the boundary point Ξ.*
4. *D is the metric-difference row. It stores the finite difference of metric reads that defines the Busemann-function value over the displayed ray window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the scalar value read from D.*
6. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *readback row. It records the level-set or horosphere-facing surface determined by the same scalar value.*
7. *H records componentwise* hsame *transport for metric, boundary, geodesic-ray, metric-difference, Real-seal, horocycle readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the same function-read route.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, \Xi, G, D, E, Q, H, C, P, N$. It has no coordinate for completed ideal-boundary quotients, selected infinite rays, global horosphere foliations, host curvature analysis, physical observer trajectories, Lean verification, or bridge credentials.

Theorem 7534.2 (Hyperbolic Busemann function NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ *carrier*

$$B = (M, \Xi, G, D, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannFunction}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the hyperbolic metric row, boundary-point row, geodesic-ray window, metric-difference row, Real seal, horocycle readback, transport, replay, provenance, and local name;*
2. *common-boundary discipline, requiring G, D, E, Q to be read toward the same boundary row Ξ;*
3. *metric-difference exposure, requiring D to display the distance comparisons through M before any scalar seal is consumed;*
4. *Real-seal order, requiring E to be downstream of M, Ξ, G, D and to expose only the scalar value named by the function row;*
5. *horosphere readback, requiring Q to be determined by the same Ξ and the same sealed scalar value E;*
6. *nonescape, excluding completed ideal-boundary quotients, selected infinite rays, global horosphere foliations, host curvature analysis, physical observer trajectories, Lean verification, and bridge credentials.*

No Hyperbolic Busemann function obligation is stored outside these rows.

Proof. Project the carrier of Definition 7534.1. The tuple $M, \Xi, G, D, E, Q, H, C, P, N$ is the whole public surface, so carrier admission is exactly admission of those rows.

The common-boundary obligation follows from Ξ and G. The geodesic-ray window is read toward the same boundary point that the function names, so a consumer cannot detach the scalar read from its boundary source.

The metric-difference and Real-seal obligations are recorded by D and E. The row D displays the metric comparisons through M, and E seals only the scalar value supplied by that displayed difference.

Finally Q, H, C, P, N preserve the same finite function read. The horocycle row is downstream of the boundary point and scalar seal, while transport, replay, provenance, and naming add no coordinate for quotient boundaries, selected rays, global horosphere foliations, host curvature theorems, physical observer trajectories, Lean credentials, or bridge credentials. □

Theorem 7534.3 (Hyperbolic Busemann function boundary handoff)**.** *Every boundary-facing or horosphere-facing consumer of an accepted* $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ *carrier factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto G, D \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{HyperbolicHorocycle}^{\uparrow}.$$

The handoff is therefore a displayed scalar readback between Busemann-boundary data, Busemann-cocycle consumers, and horosphere-level consumers, not a replacement for the boundary packet or the additive cocycle packet.

Proof. Begin with the metric row M and boundary row Ξ. They determine the visible source and the boundary point toward which the function is evaluated.

Read the geodesic-ray row G and metric-difference row D next. These rows supply the finite comparison window and the distance difference before any scalar value is public.

The Real seal E exports that scalar value, and the horocycle row Q reads its level surface through the same boundary point. The cocycle packet may consume the scalar read as one boundary comparison, and horosphere-facing consumers may consume the level-set row, but neither route obtains a completed ideal-boundary quotient, selected ray representative, global foliation, host curvature theorem, or physical observer trajectory. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ scalar read exposes the $\mathsf{HyperbolicMetric}^{\uparrow}$ metric row, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ boundary row, geodesic-ray window, metric-difference row, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{HyperbolicHorocycle}^{\uparrow}$ readback before any cocycle-facing or horosphere-facing consumer may inspect the handoff.

Independence witness: hyperbolic-metric, hyperbolic-busemann-boundary, hyperbolic-busemann-cocycle, hyperbolic-horocycle, real: $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a boundary point, a geodesic-ray window, metric-difference rows, a Real scalar seal, horocycle readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows while refusing quotient boundaries, selected rays, and global horosphere foliations.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBusemannFunction}^{\uparrow}$ packet is generated from HyperbolicMetric rows, HyperbolicBusemannBoundary rows, geodesic-ray windows, metric-difference rows, RealUp scalar seals, HyperbolicHorocycle readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7534.1, Definition 7534.2, and Definition 7534.3 seed the finite scalar readback route from boundary-point and geodesic-ray data to Real-sealed horosphere consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed ideal-boundary quotients, selected infinite rays, global horosphere foliations, host curvature analysis, or physical observer dynamics.

Upgrade path: Obligation closure requires checked carrier admission, common-boundary discipline, metric-difference exposure, Real-seal order, horosphere readback, and nonescape for BEDC.Derived.HyperbolicBusemannFunctionUp.

Chapter 7535

A Concrete Naming Certificate for $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Cauchy filters, the $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ packet records the finite uniform-filter diagonal by which Cauchy-filter rows are read through StreamName windows, regular rational readback, dyadic tolerance ledgers, and the Real seal. It sits next to the finite diagonal interface of chapter 6873, the window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the completed real-name seal of chapter 38. The chapter names a non-quotient uniform-filter diagonal interface only; it does not choose a filter representative, select a limit, quotient streams, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7535.1 (Uniform Cauchy-filter diagonal carrier)**.** *A* $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the uniform Cauchy-filter row, recording finite entourage requests and the corresponding Cauchy-filter response rows before any diagonal consumer reads the packet;*
2. *W is the finite diagonal-window row, selecting the bounded family of window indices inspected by the uniform route;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger that exposes the named stream windows used by W;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic error bounds used to compare the uniform-filter rows with the regular rational readback;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and dyadic tolerance ledger are present;*
7. *H records componentwise* hsame *transport for $F, W, S, Q, D, R, C, P, N$;*
8. *C records displayed* Cont *replay from a finite uniform Cauchy-filter request through the diagonal window, StreamName window ledger, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$, $\mathsf{CauchyFilterDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, W, S, Q, D, R, H, C, P, N$. The carrier has no coordinate for quotient filters, selected representatives, chosen limits, countable-choice schedules, quotient stream equality, host real equality, or ambient Real completeness.

Theorem 7535.2 (Uniform Cauchy-filter diagonal NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ *carrier $U = (F, W, S, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}}$ *surface consists exactly of carrier admission, uniform Cauchy-filter exposure, finite diagonal-window routing, StreamName window access, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may read a Real-facing completion seal before these displayed rows are present.*

Proof. Carrier admission is projection from Definition 7535.1. The accepted packet exposes exactly the rows $F, W, S, Q, D, R, H, C, P, N$, so every local obligation must be read from those finite rows.

The uniform-filter and diagonal-window obligations are read from F and W. The row F exposes only displayed uniform Cauchy-filter requests, while W fixes the finite window family through which the diagonal observation is allowed to pass.

The readback route is then forced by S, Q, D, R. StreamName windows are exposed by S, regular rational data are read by Q, dyadic tolerances are accounted for by D, and the Real-facing seal appears only at R after those ledgers are present.

Transport, replay, provenance, and local naming are structural. Componentwise hsame in H preserves the displayed rows, Cont replay in C repeats the same route, and P, N add provenance and local naming without adding a quotient, selector, choice schedule, stream equality, host equality, or ambient completeness row. □

Theorem 7535.3 (Uniform diagonal-window subsequence route)**.** *Every* $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ *or Real-facing completion consumer that reads an accepted* $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ *packet receives the same finite diagonal-window subsequence*

$$F, W, S, Q, D, R$$

before it can consume the displayed $\mathsf{Real}^{\uparrow}$ *seal. The route factors through chapter 126, chapter 301, and chapter 291; it exports no quotient filter, selected representative, selected limit, countable-choice schedule, quotient stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $U = (F, W, S, Q, D, R, H, C, P, N)$. The only rows available to a downstream consumer are the displayed carrier rows and their transport, replay, provenance, and local naming data.

The diagonal-window subsequence is therefore fixed by W after the uniform Cauchy-filter row F is exposed. A consumer cannot bypass W, because the StreamName row S and the regular rational row Q are attached to the windows selected by W.

The tolerance and seal part is read next. The dyadic ledger D records the tolerance bounds used by the regular rational readback, and the Real seal R is consumed only after Q and D are present.

Finally, H, C, P, N preserve this same route under transport, replay, provenance, and local naming. Since no other coordinate exists, the handoff to $\mathsf{CauchyFilterDiagonal}^{\uparrow}$ and to Real-facing completion consumers remains the finite route F, W, S, Q, D, R and cannot export the excluded quotient, choice, equality, or completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ consumer displays a uniform Cauchy-filter row, a finite diagonal-window row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any completion-facing seal is consumed.

Independence witness: cauchy-filter-diagonal, streamname, regseqrat, dyadicratcore, real: UniformCauchyFilterD is not bijective to any listed sibling because it packages uniform Cauchy-filter exposure and a finite diagonal-window subsequence together with dyadic tolerance accounting before the Real seal is public.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7535.1, Definition 7535.2, and Definition 7535.3 seed the uniform Cauchy-filter diagonal route through finite windows, regular rational readback, dyadic tolerance ledgers, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient filters, selected representatives, selected limits, countable choice, quotient stream equality, host real equality, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier exposure, uniform Cauchy-filter routing, diagonal-window subsequence routing, StreamName access, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal admission, transport replay, provenance, and local naming for BEDC.Derived.UniformCauchyFilterDiagonalUp.

Constructive story: A $\mathsf{UniformCauchyFilterDiagonal}^{\uparrow}$ packet is generated from uniform Cauchy-filter rows, a finite diagonal-window row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed continuation replay, package provenance, and local naming evidence.

Chapter 7536

A Concrete Naming Certificate for CauchyBornologyDuality$^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion, and Bourbaki and Hogbe-Nlend on bornological control, the CauchyBornologyDuality$^{\uparrow}$ packet records the finite interface where boundedness tests and Cauchy-completion reads determine one another without passing to quotient completions or selected limits. It sits after the bounded-uniform Cauchy surface of chapter 7528, the Cauchy-filter carrier of chapter 333, the metric-completion surface of chapter 1506, the uniform-space entourage discipline of chapter 339, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the dual passage between bounded Cauchy data and completion-facing reads as finite NameCert evidence, not as ambient Real completeness.

Definition 7536.1 (Cauchy bornology duality carrier)**.** *A* CauchyBornologyDuality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, F, U, M, W, R, A, C, H, P, N)$$

with the following displayed rows:

1. *B is the bornological boundedness row read from bounded subsets or bounded events;*
2. *F is the* CauchyFilter$^{\uparrow}$ *row that turns bounded Cauchy tests into finite eventuality requests;*
3. *U is the* UniformSpace$^{\uparrow}$ *entourage row that supplies the uniform reading of the same Cauchy data;*
4. *M is the* MetricCompletion$^{\uparrow}$ *row through which completion-facing consumers inspect the resulting route;*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row used to read Cauchy observations;*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the window schedule;*
7. *A is the terminal* Real$^{\uparrow}$ *seal for the displayed tolerance and distance reads;*
8. *C is the compatibility row asserting that the bornological boundedness tests and completion-facing Cauchy reads replay through the same bounded-uniform route of chapter 7528;*
9. *H stores componentwise* hsame *transport for the boundedness, Cauchy-filter, uniform, completion, window, readback, Real-seal, and compatibility rows;*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyBornologyDuality}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBornologyDuality}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected limit, countable choice, host equality of filters, trichotomy over distances, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7536.2 (Cauchy bornology duality NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBornologyDuality}^{\uparrow}$ *carrier*

$$D = (B, F, U, M, W, R, A, C, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyBornologyDuality}^{\uparrow}}$ *surface consists exactly of carrier admission, bornological boundedness, Cauchy-filter eventuality, uniform entourage stability, metric-completion routing, finite StreamName windows, RegSeqRat readback, Real-seal tolerance, bounded-completion compatibility, componentwise* hsame *transport,* Pkg *provenance, and local naming. A consumer may pass from boundedness tests to completion reads, or from completion reads back to bounded Cauchy tests, only through these rows.*

Proof. Project the carrier of Definition 7536.1. The packet exposes $B, F, U, M, W, R, A, C, H, P, N$ as the whole local surface, so the boundedness, Cauchy-filter, uniform, completion, finite-window, readback, Real-seal, compatibility, transport, provenance, and naming rows are all displayed finite BEDC data.

The forward direction reads bounded Cauchy tests through B, F, U before any completion-facing row is available. The compatibility row C ties that route to the bounded-uniform Cauchy packet of chapter 7528, so the metric-completion row M can only consume the same displayed bounded-uniform route.

The reverse direction starts from a completion-facing read through M, W, R, A. The finite StreamName windows, RegSeqRat readback, and Real seal recover only the bounded Cauchy observations stored in B, F, U, C, not a chosen limit point or quotient representative.

Finally inspect H, P, N. Componentwise transport, provenance, and local naming preserve the same finite packet. Since the carrier has no quotient-completion, selected-limit, countable-choice, host-filter-equality, trichotomy, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7536.3 (Cauchy bornology duality completion route)**.** *Every completion-facing* $\mathsf{CauchyBornologyDuality}^{\uparrow}$ *consumer factors through*

$\mathsf{CauchyBornologicalUniformity}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, Rea

before it exports a completion seal. The route is dual in the finite BEDC sense: boundedness tests determine exactly the Cauchy-completion reads carried by the packet, and completion reads reflect back only to those bounded Cauchy tests. It does not export quotient completions, selected representatives, countable choice, or ambient Real completeness.

Proof. Use the NameCert obligations of Definition 7536.2. They force every accepted consumer to expose boundedness, Cauchy-filter, uniform, completion, finite-window, readback, Real-seal, compatibility, transport, provenance, and local naming rows.

The bounded-uniform route of chapter 7528 supplies the common interface. Thus a boundedness-to-completion read and a completion-to-boundedness read share the same B, F, U, C spine before the metric-completion row M reaches the stream, regular rational, and Real-seal rows.

The excluded endpoints are absent by carrier inspection. No row stores a quotient completion, selected representative, countable-choice schedule, or ambient completeness theorem, so the completion seal is only the displayed bounded Cauchy route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyBornologyDuality}^{\uparrow}$ completion consumer displays the boundedness row, $\mathsf{CauchyFilter}^{\uparrow}$ row, $\mathsf{UniformSpace}^{\uparrow}$ row, $\mathsf{MetricCompletion}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal, compatibility row, transport, provenance, and local NameCert before a completion seal is exported.

Independence witness: cauchy-bornological-uniformity, cauchyfilter, metriccompletion, uniformspace, streamname, regseqrat, real: $\mathsf{CauchyBornologyDuality}^{\uparrow}$ is not bijective to any listed sibling because it packages the two-way boundedness and completion readback route as one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBornologyDuality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBornologyDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7536.1, Definition 7536.2, and Definition 7536.3 seed the boundedness and completion readback interface over $\mathsf{CauchyBornologicalUniformity}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBornologyDuality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient completion, selected limits, countable choice, host filter equality, trichotomy, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundedness-to-completion routing, completion-to-boundedness readback, compatibility, transport, provenance, local naming, and ledger non-escape for BEDC.Derived.CauchyBornologyDualit

Constructive story: A $\mathsf{CauchyBornologyDuality}^{\uparrow}$ packet is generated from bornological boundedness rows, Cauchy-filter eventuality rows, uniform entourage rows, metric-completion routing, finite stream windows, regular rational readback, Real-seal tolerance, bounded-completion compatibility, componentwise transport, package provenance, and local naming evidence.

Chapter 7537

Birkhoff Distributive Lattice NameCert Construction

Following Birkhoff and Davey–Priestley, the $\mathsf{BirkhoffDistributiveLattice}^{\uparrow}$ packet records the finite representation surface for distributive lattices before any Stone-style, Boolean, or order-facing consumer may read the object as a represented lattice. The packet sits after the order carrier of chapter 53, the meet-join surface of chapter 55, the bounded endpoint rows of chapter 6501, the Boolean comparison envelope of chapter 3733, and the prime-filter dual route of chapter 261. It is a finite NameCert reconstruction of the Birkhoff distributive-lattice handoff, not an import of host algebraic representation theorems, choice-selected prime filters, quotient lattice equality, or classical completeness.

Definition 7537.1 (Birkhoff distributive lattice carrier)**.** *A* $\mathsf{BirkhoffDistributiveLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (P, L, E, J, M, D, F, S, H, C, Q, N)$$

with the following visible rows:

1. *P is the* $\mathsf{Poset}^{\uparrow}$ *order row naming the displayed carrier order.*
2. *L is the* $\mathsf{Lattice}^{\uparrow}$ *meet-join row over the same carrier.*
3. *E is the* $\mathsf{BoundedLattice}^{\uparrow}$ *endpoint row exposing the displayed bottom and top entries.*
4. *J is the finite join-irreducible ledger used by the representation read.*
5. *M is the finite meet-irreducible ledger used for the dual read.*
6. *D is the distributivity witness row, read only through the displayed meet and join coordinates.*
7. *F is the finite down-set representation row attached to J and the order row P.*
8. *S is the* $\mathsf{StoneDuality}^{\uparrow}$ *and* $\mathsf{BooleanAlgebra}^{\uparrow}$ *sibling handoff row for Boolean comparison and prime-filter inspection.*
9. *H records componentwise* hsame *transport for P, L, E, J, M, D, F, S and the structural rows.*
10. *C records displayed* Cont *replay for opening the order, lattice, endpoint, irreducible, representation, and sibling handoff rows.*
11. *Q is the* Pkg *provenance over* $\mathsf{BirkhoffDistributiveLattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{BoundedLattice}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{StoneDuality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{BirkhoffDistributiveLattice}^{\uparrow}}$ *row.*
The carrier stores no host lattice quotient, no choice-selected prime ideal, no ambient powerset equality, no classical completeness theorem, and no Boolean algebra replacement for the displayed distributive-lattice rows.

Theorem 7537.2 (Birkhoff distributive lattice NameCert obligations)**.** *For every accepted* BirkhoffDistributiveLatti *carrier $B = (P, L, E, J, M, D, F, S, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BirkhoffDistributiveLattice}^{\uparrow}}$ *surface consists exactly of carrier admission, order exposure through P, meet-join exposure through L, bounded endpoint exposure through E, finite irreducible ledgers J and M, distributivity inspection through D, finite down-set representation through F, Boolean and Stone sibling handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through Q, and local naming through N.*

No accepted read may replace these rows by host distributive-lattice equality, a selected prime-filter family, classical Stone representation, unbounded ideal completion, or an ambient Boolean algebra quotient.

Proof. Project the carrier of Definition 7537.1. Its algebraic rows are the order row P, the lattice row L, the endpoint row E, the irreducible ledgers J, M, the distributivity row D, the representation row F, and the sibling route S; its structural rows are H, C, Q, N.

The order-facing obligations are supplied by the $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, and $\mathsf{BoundedLattice}^{\uparrow}$ sibling chapters. They make the carrier order, meet-join operations, and endpoints visible before any representation read is consumed.

The representation-facing obligations then read J, M, D, F as finite ledgers. Distributivity is inspected through the displayed meet and join rows, while the down-set row remains attached to the finite irreducible ledger rather than becoming an ambient powerset quotient.

Finally the Boolean and Stone sibling handoff S is only a route through chapter 3733 and chapter 261. Transport, replay, provenance, and local naming preserve that finite route, so the packet exports the listed NameCert surface and no host representation theorem. □

Theorem 7537.3 (Birkhoff distributive lattice sibling handoff)**.** *Every order-facing, Boolean-facing, or Stone-facing consumer of an accepted* $\mathsf{BirkhoffDistributiveLattice}^{\uparrow}$ *carrier must first factor through*

$$\mathsf{Poset}^{\uparrow} \longrightarrow \mathsf{Lattice}^{\uparrow} \longrightarrow \mathsf{BoundedLattice}^{\uparrow} \longrightarrow \mathsf{BooleanAlgebra}^{\uparrow} \longrightarrow \mathsf{StoneDuality}^{\uparrow}$$

as displayed sibling rows attached to P, L, E, S. The finite irreducible ledgers J, M, distributivity row D, down-set representation row F, componentwise transport H, replay C, provenance Q, and local name N remain visible throughout the handoff.

The route does not claim a general representation theorem for all host lattices, a choice-selected spectrum, quotient equality of down-sets, classical prime-filter separation, or Boolean replacement of the original carrier.

Proof. Start with the NameCert obligations of Definition 7537.2. They require the order, lattice, bounded endpoint, irreducible, distributivity, representation, sibling handoff, transport, replay, provenance, and local naming rows to be visible together.

For order-facing consumers, project P, L, E and use the sibling chapters chapter 53, chapter 55, and chapter 6501. These rows expose order, meet, join, bottom, and top data before a representation row can be read.

For Boolean-facing and Stone-facing consumers, project S together with J, M, D, F. The Boolean and Stone sibling chapters provide comparison and duality routes, but the irreducible ledgers and down-set row remain part of the carrier and are not discharged by those routes.

Thus every consumer sees the displayed sibling chain and the local representation data before using the Birkhoff packet. Since no coordinate stores a host quotient, selected spectrum, prime-filter choice, or classical completeness theorem, no such endpoint is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffDistributiveLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffDistributiveLattice}^{\uparrow}$ packet is generated from Poset, Lattice, BoundedLattice, BooleanAlgebra, and StoneDuality sibling rows, finite irreducible ledgers, a distributivity row, a down-set representation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffDistributiveLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7537.1, Definition 7537.2, and Definition 7537.3 seed the finite distributive-lattice representation packet and its order, Boolean, and Stone sibling route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffDistributiveLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distributive-lattice representation, choice-selected spectra, quotient equality of down-sets, classical Stone separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, irreducible-ledger exactness, distributivity-row preservation, down-set representation routing, sibling handoff, transport, replay, provenance, and local naming for BEDC.Derived.BirkhoffDistributiveLatticeUp.

Chapter 7538

A Concrete Naming Certificate for $\mathsf{RoundedRealDomain}^{\uparrow}$

Following Bishop–Bridges constructive analysis and domain-theoretic real approximation, the $\mathsf{RoundedRealDomain}^{\uparrow}$ packet records rounded real approximants as finite NameCert data before any Real or Completion consumer may treat them as completed reals. It sits between the formal-ball surface of chapter 1510, the interval-domain carrier of chapter 1543, the way-below refinement of chapter 1545, the lower-upper located-cut boundary of chapter 2087, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a rounded approximation surface; it is not a quotient real, not a host supremum, not a selected representative, and not an ambient completeness principle.

Definition 7538.1 (Rounded real domain carrier)**.** *A* $\mathsf{RoundedRealDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, I, W, K, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *row, recording the center-radius request and the rounded inclusion budget inspected by the approximant.*
2. *I is the* $\mathsf{IntervalDomain}^{\uparrow}$ *row, displaying the current interval-domain cell to which the formal ball is attached.*
3. *W is the* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *row, recording the strict containment margin by which the rounded cell refines a coarser interval-domain observation.*
4. *K is the* $\mathsf{LowerUpperRealCut}^{\uparrow}$ *row, keeping the rounded interval adjacent to the lower-upper cut boundary used by Real-facing consumers.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, storing the dyadic center, radius, slack, endpoint, and comparison data used by F, I, W, K.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which endpoint and radius observations are inspected.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the same windows and dyadic budgets.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after the formal ball, interval-domain, way-below, cut-boundary, dyadic, stream, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for F, I, W, K, Q, S, R, E and the structural rows.*
10. *C records displayed* Cont *replay for the same rounded approximation route.*
11. *P is the* Pkg *provenance over* $\mathsf{RoundedRealDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{LowerUpperRealCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RoundedRealDomain}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $F, I, W, K, Q, S, R, E, H, C, P, N$. It contains no coordinate for quotient real equality, host supremum, trichotomy, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7538.2 (Rounded real domain NameCert obligations)**.** *For an accepted* $\mathsf{RoundedRealDomain}^{\uparrow}$ *carrier*

$$D = (F, I, W, K, Q, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RoundedRealDomain}^{\uparrow}}$ *surface consists exactly of carrier admission, formal-ball rounding visibility, interval-domain attachment, way-below margin stability, lower-upper cut boundary visibility, dyadic exactness, finite-window scheduling, regular rational readback, terminal real-seal non-escape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7538.1. The accepted packet exposes exactly the formal-ball row, interval-domain row, way-below row, lower-upper cut row, dyadic ledger, stream window, regular readback, real seal, and structural rows.

The rounded approximation rows are read in order. The formal-ball row F supplies the radius request, I attaches it to the interval-domain carrier, and W records the strict way-below refinement that keeps the rounded cell inside the displayed approximation route.

The cut and readback rows then prevent escape. The lower-upper cut row K keeps the approximant adjacent to the two-sided located boundary, while Q, S, R provide the dyadic, finite-window, and regular rational reads consumed before the seal row E.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient, host-supremum, trichotomy, choice, selected-representative, or completeness coordinate, the local certificate cannot export those principles. □

Theorem 7538.3 (Rounded real domain cut handoff)**.** *Let*

$$D = (F, I, W, K, Q, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RoundedRealDomain}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing read of D factors through the displayed route*

$\mathsf{FormalBall}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{LowerUpperRealCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Re

Thus a rounded real-domain approximant reaches the real seal only by passing through formal-ball rounding data, interval-domain containment, way-below margin evidence, the lower-upper cut boundary, dyadic tolerance rows, finite windows, regular rational readback, and the structural naming rows. It does not import quotient reals, host suprema, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the formal-ball and interval-domain rows. The row F records the rounded center-radius request, and the row I supplies the displayed interval-domain cell that the request is allowed to refine.

Next read the way-below and lower-upper cut rows. The row W supplies strict containment for the rounded cell, and the row K keeps the same approximation adjacent to the lower-upper cut boundary before any Real-facing seal is consumed.

Then read the arithmetic and observation rows. The dyadic ledger Q records the endpoint and radius budgets, S schedules the finite observations, and R provides the regular rational readback for the same windows.

The seal row E is downstream of those rows, and H, C, P, N only transport, replay, source, and name the displayed route. Therefore every real or completion handoff factors through the stated finite tuple and cannot project a quotient real, host supremum, selected representative, choice principle, trichotomy decision, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedRealDomain}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedRealDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7538.1, Definition 7538.2, and Definition 7538.3 seed the finite rounded real-domain approximation surface over formal balls, interval-domain cells, way-below refinements, lower-upper cuts, dyadic ledgers, finite windows, regular readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedRealDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, host suprema, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{RoundedRealDomain}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, formal-ball rounding visibility, interval-domain attachment, way-below margin stability, lower-upper cut boundary visibility, dyadic exactness, finite-window scheduling, regular readback, real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RoundedRealDomainUp.

Constructive story: A $\mathsf{RoundedRealDomain}^{\uparrow}$ packet is generated from FormalBall, IntervalDomain, IntervalDomainWayBelow, LowerUpperRealCut, DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 7539

A Concrete Naming Certificate for $\mathsf{MenelausTheorem}^{\uparrow}$

Following Menelaus and Coxeter-style projective geometry, the $\mathsf{MenelausTheorem}^{\uparrow}$ packet records the finite collinearity and directed-ratio certificate dual to the concurrency certificate of chapter 3509. It sits over the affine coordinate carrier of chapter 206, the projective incidence boundary of chapter 207, the rational ratio ledger of chapter 37, and the terminal real comparison surface of chapter 38. The packet names one displayed triangle, one transversal line, and three side-intersection witnesses as finite BEDC data; it is not a host projective theorem, a selected coordinate chart, an unrestricted orientation principle, or an ambient completeness claim for $\mathsf{Real}^{\uparrow}$.

Definition 7539.1 (MenelausTheorem carrier)**.** *A* $\mathsf{MenelausTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, P, L, I, R, Q, H, C, G, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AffineSpace}^{\uparrow}$ *triangle row, including three displayed vertices and the affine side carriers on which ratios are read.*
2. *P is the* $\mathsf{ProjectiveSpace}^{\uparrow}$ *incidence row carrying the projective line and side-completion data needed to treat parallel affine cases uniformly.*
3. *L is the transversal-line row, displayed as a finite incidence object rather than as a host subset of points.*
4. *I records the three side-intersection witnesses, including the permitted at-infinity branch supplied by P.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *directed-ratio ledger for the three side cuts, with numerator, denominator, sign, and nonzero-side rows visible.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *readback row used only to compare the terminal product relation after the rational ledger has been displayed.*
7. *H records componentwise* hsame *transport for the affine triangle, projective incidence, transversal, intersections, directed ratios, and real readback.*
8. *C is the displayed* Cont *replay of the route from affine incidence through projective completion and rational ratio multiplication to the real-facing comparison.*

9. *G is the* Pkg *provenance over* MenelausTheorem$^{\uparrow}$, Ceva$^{\uparrow}$, AffineSpace$^{\uparrow}$, ProjectiveSpace$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MenelausTheorem}^{\uparrow}}$ *row.*

The classifier accepts only packets exposing all ten rows. It has no coordinate for a hidden projective model, selected affine chart, untracked signed length, quotient real equality, or detached Real$^{\uparrow}$ *completeness theorem.*

Theorem 7539.2 (MenelausTheorem NameCert obligations)**.** *For every accepted* MenelausTheorem$^{\uparrow}$ *carrier $M = (A, P, L, I, R, Q, H, C, G, N)$, the local* NameCert$_{\text{MenelausTheorem}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed affine triangle, projective incidence boundary, transversal line, side-intersection witnesses, directed-ratio ledger, real readback, transport, replay, provenance, and local name;*
2. *incidence exactness, requiring the three side intersections in I to be read through A, P, L before any ratio row is consumed;*
3. *ratio exactness, requiring every directed ratio in R to expose its* Rat$^{\uparrow}$ *numerator, denominator, sign, and nonzero-side rows before the product comparison is read;*
4. *real-seal handoff, requiring the* Real$^{\uparrow}$ *row Q to be downstream of R and never a replacement for the rational ratio ledger;*
5. *Ceva companion separation, recording that chapter 3509 supplies a concurrency route while this packet supplies the transversal-collinearity route;*
6. *ledger non-escape, excluding hidden projective models, selected charts, host orientation, quotient real equality, countable choice, and detached ambient completeness.*

No Menelaus obligation is stored outside $A, P, L, I, R, Q, H, C, G, N$.

Proof. Carrier admission is projection from Definition 7539.1. The ten rows list the triangle, projective boundary, transversal, intersections, directed ratios, real readback, transport, replay, provenance, and local naming data available to the certificate.

Incidence exactness is forced before ratio formation. The transversal row L and side-intersection row I can be read only against the affine triangle A and the projective incidence boundary P, so no ratio row is formed from an untracked host line.

The ratio ledger is then local to R. Each directed ratio exposes rational numerator, denominator, sign, and nonzero-side rows before the terminal product comparison reaches Q.

Finally compare with the Ceva companion. The provenance row G points to the concurrency certificate of chapter 3509, while the present packet keeps the transversal-collinearity route separate. Since Q, H, C, G, N only read, transport, replay, source, and name the displayed route, the certificate exports none of the refused hidden projective, chart-selection, quotient-real, choice, or ambient-completeness data. □

Closure status for NameCert$_{\text{MenelausTheorem}^{\uparrow}}$**.**

Constructive story: A MenelausTheorem$^{\uparrow}$ packet is generated from an affine triangle, projective incidence boundary, transversal line, side-intersection witnesses, directed rational ratios, real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MenelausTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 7539.1 and Definition 7539.2 seed the finite transversal-collinearity certificate over Ceva$^{\uparrow}$, AffineSpace$^{\uparrow}$, ProjectiveSpace$^{\uparrow}$, Rat$^{\uparrow}$, and Real$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MenelausTheorem}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Lean verification, bridge checking, full projective incidence geometry, selected affine charts, quotient real equality, countable choice, or ambient Real$^{\uparrow}$ completeness.
Upgrade path: Scoped closure requires checked incidence exactness, directed-ratio exactness, real-seal handoff, Ceva companion separation, transport, replay, provenance, and local naming for BEDC.Derived.MenelausTheoremUp.

Chapter 7540

A Concrete Naming Certificate for $\mathsf{MetricMidpointConvexity}^{\uparrow}$

Following Menger and Busemann, the $\mathsf{MetricMidpointConvexity}^{\uparrow}$ packet records the finite midpoint-convexity surface for a displayed metric carrier before any ambient geodesic or convexity theorem is read. It sits over the metric carrier of chapter 57, the real-valued distance comparison row used by $\mathsf{RealMetric}^{\uparrow}$, the geodesic segment route of chapter 2213, the convex-set membership boundary of chapter 208, and the regular-Cauchy midpoint packet of chapter 1670. The packet names midpoint witnesses and distance ledgers as finite BEDC rows; it does not assert host midpoint uniqueness, open-cover compactness, selected geodesics, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7540.1 (Metric midpoint convexity carrier)**.** *A* $\mathsf{MetricMidpointConvexity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, R, G, V, W, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row, including the point classifier, distance request surface, and metric-law ledger used by all midpoint reads.*
2. *R is the* $\mathsf{RealMetric}^{\uparrow}$ *distance row, recording the real-valued distance comparison only through finite rational or dyadic observations.*
3. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *segment row that displays the two endpoints and the inspected midpoint witness as geodesic data, not as a host path choice.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *membership row for the inspected subset, including endpoint admission and midpoint admission as visible convexity-ledger entries.*
5. *W is the* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *row that supplies the finite regular-Cauchy midpoint window read by Real-facing consumers.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two endpoint distances, the midpoint distance comparison, and the convexity admission budget.*
7. *H records componentwise* hsame *transport for metric, distance, geodesic, convex-membership, midpoint-window, dyadic-ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a midpoint-convexity request through X, R, G, V, W, D and then to H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricMidpointConvexity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RegularCauchyMidpoint}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricMidpointConvexity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, R, G, V, W, D, H, C, P, N$. It has no coordinate for a selected global geodesic, a quotient point equality, a host convexity principle, an open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7540.2 (Metric midpoint convexity NameCert obligations)**.** *For every accepted* MetricMidpointConvexity *carrier $M = (X, R, G, V, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricMidpointConvexity}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric row, real-distance row, geodesic midpoint row, convex-set membership row, regular-Cauchy midpoint window, dyadic tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It also requires endpoint exposure through X, distance exactness through R, D, midpoint witness exposure through G, W, convex-membership exactness through V, and non-escape for selected geodesics, quotient point equality, host convexity, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7540.1. The rows $X, R, G, V, W, D, H, C, P, N$ are the complete accepted inventory, so every local naming obligation must be read from those rows.

The midpoint witness is finite. The metric row X exposes the endpoints and point classifier, the geodesic row G exposes the inspected midpoint witness, and the regular-Cauchy midpoint row W gives the Real-facing finite window for the same midpoint read.

The convexity and distance ledgers are then local. The row V records endpoint and midpoint membership, while R and D record the real-valued distance comparisons and dyadic tolerance budget before any consumer can inspect a terminal real-distance value.

Finally H, C, P, N transport, replay, source, and name the displayed route. Since the packet contains no row for a selected global geodesic, quotient point equality, host convexity theorem, open-cover compactness, or ambient completeness, none of those claims can be exported by the local certificate. □

Falsifiable prediction: Every accepted $\mathsf{MetricMidpointConvexity}^{\uparrow}$ read exposes MetricSpace, RealMetric, GeodesicMetric, ConvexSet, RegularCauchyMidpoint, DyadicRatCore, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a midpoint-convexity consumer can inspect the packet.

Independence witness: metricspace, realmetric, geodesicmetric, convexset, regular-cauchy-midpoint: $\mathsf{MetricMidpointConvexity}^{\uparrow}$ is not bijective to any listed sibling because it binds metric endpoints, real-distance comparisons, a geodesic midpoint witness, convex-set membership, regular-Cauchy midpoint windows, dyadic tolerances, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricMidpointConvexity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricMidpointConvexity}^{\uparrow}$ packet is generated from a MetricSpace carrier row, a RealMetric distance row, a GeodesicMetric midpoint witness, ConvexSet membership rows, a RegularCauchyMidpoint window, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricMidpointConvexity}^{\uparrow}}$, seed)

Scope closed: Definition 7540.1 and Definition 7540.2 seed the finite midpoint-convexity certificate over MetricSpace, RealMetric, GeodesicMetric, ConvexSet, RegularCauchyMidpoint, and DyadicRatCore rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricMidpointConvexity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Real completeness, open-cover compactness, selected geodesics, host midpoint uniqueness, quotient point equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint witness exactness, distance-ledger exactness, convex-membership exactness, transport, replay, provenance, and local naming for BEDC.Derived.MetricMidpointConvexityUp.

Chapter 7541

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$

Following Heine–Cantor finite-cover arguments in constructive analysis, the $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ packet records the finite fold that turns a compact finite cover and pointwise modulus rows into one tolerance-only uniform-modulus ledger. It realizes the finite ProbeBundle lower-bound fold of Definition 9237.5 in chapter 9237. The packet sits after the compact finite-net surface of chapter 60, the continuous-map route of chapter 130, the finite Lebesgue-number ledger of chapter 452, the rational lower-bound route of chapter 37, and the uniform-modulus consumer of chapter 369. It is a finite cover-fold certificate, not a selected global modulus functional, open-cover compactness theorem, quotient function equality, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 7541.1 (Compact uniform finite-cover fold carrier)**.** *A* $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, L, B, R, U, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-cover source row, exposing one finite* ProbeBundle*, its centers, coverage witnesses, and nonempty spine.*
2. *F is the continuous-map pointwise-modulus row over the same source carrier, with local precision histories attached to the bundled centers.*
3. *L is the finite Lebesgue-number or subordinate-radius row tying each bundled center to a positive cover radius.*
4. *B is the finite lower-bound fold row over the bundled positive radius histories.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row used by the fold comparisons.*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *row receiving the tolerance-only precision history produced by B.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for center observations, local-modulus reads, radius comparisons, and uniform-modulus output.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to Real-facing tolerance and output estimates.*

9. *H records componentwise* hsame *transport for the compact cover, continuous-map row, Lebesgue-number row, lower-bound fold, rational readback, uniform-modulus output, finite windows, regular readbacks, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay through $K, F, L, B, R, U, W, Q, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, B, R, U, W, Q, H, C, P, N$. It has no coordinate for an external global-modulus selector, arbitrary open-cover compactness, selected cover refinements, quotient function equality, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Theorem 7541.2 (Compact uniform finite-cover fold NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ *carrier*

$$G = (K, F, L, B, R, U, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite cover, pointwise-modulus rows, subordinate positive-radius ledger, lower-bound fold, rational readbacks, uniform-modulus output, finite windows, regular readbacks, transport, replay, provenance, and local naming;*

2. *cover exactness, requiring every center, coverage, and nonempty-spine read to pass through the displayed compact finite-cover row K;*

3. *local-modulus exactness, requiring every pointwise precision row to pass through F over the centers supplied by K;*

4. *radius-fold exactness, requiring every positive-radius comparison to pass through L, R before the finite lower-bound fold B is read;*

5. *uniform-output ordering, requiring the* $\mathsf{UniformModulus}^{\uparrow}$ *row U to be downstream of B and not an independent modulus selector;*

6. *readback ordering, requiring W and Q to expose finite windows and regular rational readbacks before any Real-facing uniform estimate is consumed;*

7. *nonescape, excluding open-cover compactness, selected global moduli, quotient function equality, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No compact uniform finite-cover fold obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7541.1. The accepted packet displays exactly one compact finite-cover row, one continuous pointwise-modulus row, one subordinate-radius ledger, one lower-bound fold, one rational readback route, one uniform-modulus output, finite windows, regular readbacks, and structural rows.

Cover exactness is supplied by K. The bundle centers, coverage witnesses, and nonempty spine have no second cover coordinate, so every cover read enters the packet through the displayed compact finite-cover source.

Local-modulus exactness is supplied by F over the centers from K. Pointwise precision rows are therefore tied to the finite cover and cannot be replaced by an external continuity theorem.

The fold route is supplied by L, R, B. The positive-radius ledger and rational readback rows expose the finite comparisons before primitive recursion over the bundle records the lower-bound fold.

Finally U, W, Q, H, C, P, N impose output order and local naming. The uniform-modulus row is downstream of the fold, finite windows and regular readbacks precede Real-facing estimates, and the structural rows contain none of the refused compactness, selection, quotient, choice, completeness, verification, or bridge data. □

Theorem 7541.3 (Compact uniform finite-cover fold route)**.** *Every tolerance-only uniform-modulus read exported by an accepted* $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ *packet factors through*

$$K, F, L \longmapsto R, B \longmapsto U, W, Q, H, C, P, N.$$

The read first receives the finite compact cover, pointwise-modulus rows, and subordinate positive-radius ledger; only then may it inspect the rational comparison readback, finite lower-bound fold, uniform-modulus output, finite windows, regular readbacks, and structural NameCert rows. It exports no selected global modulus functional or open-cover compactness theorem.

Proof. Start with an accepted packet $G = (K, F, L, B, R, U, W, Q, H, C, P, N)$. The compact cover row K supplies the finite $\mathsf{ProbeBundle}$, centers, coverage rows, and nonempty spine, while F supplies the local pointwise-modulus rows over those centers.

The subordinate-radius row L records the positive radius attached to each bundled center. The rational readback R exposes the comparison data needed by the finite fold, and B records the lower-bound fold over the displayed finite spine.

The uniform output is then read through U. The finite-window and regular-readback rows W, Q supply the observation and Real-facing estimate boundary, while H, C, P, N transport, replay, source, and name the same route.

Because no row stores a global modulus selector, arbitrary open-cover compactness proof, selected refinement, quotient function equality, countable choice, host real equality, or ambient completeness witness, the displayed factorization is the whole public route. □

Theorem 7541.4 (Compact uniform finite-cover fold vision realization)**.** *The finite lower-bound fold of Definition 9237.5 is realized by an accepted* $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ *carrier exactly at the boundary where compact finite-cover data and continuous pointwise-modulus rows become a* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row. The realization is a concrete NameCert packet over* $K, F, L, B, R, U, W, Q, H, C, P, N$*, not a separate analytic compactness principle.*

Proof. Read the capstone theorem through chapter 9237. Its finite $\mathsf{ProbeBundle}$, center rows, pointwise-modulus rows, subordinate radius rows, and rational lower-bound fold are precisely the rows displayed as K, F, L, R, B.

The theorem's output uniform-modulus ledger is the row U. The remaining rows W, Q, H, C, P, N provide finite windows, regular readbacks, transport, replay, provenance, and local naming for that output.

Thus the vision-side fold lands as one concrete NameCert packet. Since the packet contains no additional compactness, selection, quotient, choice, host equality, or completeness coordinate, the realization does not enlarge the theorem beyond the displayed finite cover-fold boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7541.1, Definition 7541.2, Definition 7541.3, and Definition 7541.4 seed the finite cover-fold route from compact finite covers and pointwise-modulus rows through positive-radius ledgers, rational lower-bound folds, uniform-modulus output, finite windows, readbacks, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, selected global moduli, selected cover refinements, quotient function equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, cover exactness, local-modulus exactness, radius-fold exactness, uniform-output ordering, readback ordering, and nonescape proofs for BEDC.Derived.CompactUniformFiniteCoverFoldUp

Constructive story: A $\mathsf{CompactUniformFiniteCoverFold}^{\uparrow}$ packet is generated from a compact finite-cover row, continuous pointwise-modulus row, subordinate positive-radius ledger, rational readback row, finite lower-bound fold, UniformModulus output, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7542

A Concrete Naming Certificate for $\mathsf{IntervalImageConnected}^{\uparrow}$

Following Bishop and Bridges on constructive interval arguments, the $\mathsf{IntervalImageConnected}^{\uparrow}$ packet records the finite route by which a continuous image of a displayed interval remains connected at the Real-facing certificate boundary. It sits after the continuous image packet of chapter 1897, the located intermediate-value rows of chapter 1402, the located interval boundary of chapter 56, the Darboux-property surface carried by $\mathsf{DarbouxProperty}^{\uparrow}$, and the terminal real-name seal of chapter 38. The packet names a constructive interval-image connectedness carrier; it does not assert open-cover compactness, host image-set connectedness, quotient equality of real names, selected preimages, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7542.1 (Interval image connected carrier)**.** *An* $\mathsf{IntervalImageConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, V, D, L, R, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, including the displayed endpoints, membership ledger, and located interval-window reads.*
2. *F is the* $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *row carrying the continuous map, source interval boundary, image-output seal, and finite image-window route.*
3. *V is the* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ *or* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *route used when an image-connectedness request must pass between two displayed Real-facing image values.*
4. *D is the* $\mathsf{DarbouxProperty}^{\uparrow}$ *sibling row, read only as the interval-value handoff induced by the same continuous image route.*
5. *L records the finite lower and upper image brackets whose endpoints are read through I, F, V, D before a connectedness request is answered.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for image values, reached only after the interval, continuity, intermediate-value, Darboux, and bracket rows are displayed.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for interval endpoints, source probes, image observations, and intermediate-value bisection reads.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and image-value comparisons.*

9. *H records componentwise* hsame *transport for the interval, continuous-image, intermediate-value, Darboux, bracket, Real seal, finite-window, readback, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through $I, F, V, D, L, R, W, Q, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{IntervalImageConnected}^{\uparrow}$, $\mathsf{ContinuousIntervalImage}^{\uparrow}$, $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalImageConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, V, D, L, R, W, Q, H, C, P, N$. It has no coordinate for host image subsets, selected preimages, endpoint-choice functionals, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.IntervalImageConnectedUp.IntervalImageConnectedTasteGate single carrier alignment`

Theorem 7542.2 (Interval image connected NameCert obligations)**.** *For every accepted* $\mathsf{IntervalImageConnected}^{\uparrow}$ *carrier*

$$K = (I, F, V, D, L, R, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalImageConnected}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval row, continuous interval-image row, intermediate-value route, Darboux handoff, bracket row, Real seal, finite windows, regular readbacks, transport, replay, provenance, and local naming;*

2. *interval-source exactness, requiring endpoint, membership, and source-window reads to pass through the displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *row I;*

3. *image-route exactness, requiring image-value reads to pass through the* $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *row F before the connectedness row is inspected;*

4. *intermediate-value exactness, requiring every between-value request to pass through V and the same displayed image route rather than through host order completeness;*

5. *Darboux compatibility, requiring D to be read as a sibling route over the same interval and continuous-image rows;*

6. *Real-seal ordering, requiring W and Q to expose finite StreamName windows and RegSeqRat readbacks before the terminal* $\mathsf{Real}^{\uparrow}$ *seal R is available;*

7. *nonescape, excluding host image subsets, selected preimages, endpoint selectors, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No interval-image connectedness obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7542.1. The accepted packet displays exactly one located interval source, one continuous image row, one intermediate-value route, one Darboux handoff, one bracket row, one Real seal, finite windows, readbacks, and structural rows.

Interval-source exactness is supplied by I. Endpoint, membership, and source-window requests have no second source interval coordinate, so each enters the packet through the displayed $\mathsf{LocatedInterval}^{\uparrow}$ row.

Image-route exactness is supplied by F. The continuous interval-image row carries the map, image-output route, and finite image windows before any connectedness request can read an image value.

The between-value route is supplied by V and checked against D. These rows expose the located or dyadic intermediate-value handoff and the Darboux sibling surface over the same interval data, so no host order-completeness theorem is substituted.

Finally L, R, W, Q, H, C, P, N impose readback order and local naming. The bracket row is read before the Real seal, StreamName windows and RegSeqRat readbacks precede terminal image-value comparison, and the structural rows contain none of the refused host image, selection, quotient, choice, compactness, completeness, verification, or bridge data. □

Theorem 7542.3 (Interval image connected intermediate-value route)**.** *Every connectedness read exported by an accepted* $\mathsf{IntervalImageConnected}^{\uparrow}$ *carrier factors through*

$$I, F \longmapsto V, D, L \longmapsto W, Q, R, H, C, P, N.$$

The read uses the located interval source, continuous interval-image route, located or dyadic intermediate-value row, Darboux handoff, bracket row, finite windows, regular readbacks, Real seal, and structural NameCert rows. It does not introduce host image connectedness, selected preimages, open-cover compactness, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted packet $K = (I, F, V, D, L, R, W, Q, H, C, P, N)$. The source and image sides are fixed by I and F: the located interval row supplies the endpoint and membership boundary, and the continuous interval-image row supplies the image-value route over that same source.

A connectedness request between two displayed image values must then read V and D. The intermediate-value row supplies the finite bisection or located crossing route, while the Darboux row records the same interval-value handoff as a sibling route.

The bracket, window, and readback rows are downstream. The row L identifies the displayed image brackets, W and Q expose finite observations and rational readbacks, and R seals the image values as $\mathsf{Real}^{\uparrow}$ names.

The structural rows H, C, P, N transport, replay, source, and name only this route. Since no row stores a host image set, selected preimage, open-cover compactness theorem, quotient real equality, trichotomy, choice, or ambient completeness witness, every connectedness read remains inside the displayed factorization. □

Theorem 7542.4 (Interval image connected Real seal)**.** *Every Real-facing seal read of an accepted* $\mathsf{IntervalImageConnected}^{\uparrow}$ *packet is downstream of the interval-image connectedness route. A consumer may inspect the terminal* $\mathsf{Real}^{\uparrow}$ *row only after it has read the located interval source* I*, the continuous image row* F*, the intermediate-value row* V*, the Darboux handoff* D*, the bracket row* L*, the finite windows* W*, and the RegSeqRat readback row* Q*. The seal exports no ambient Real completeness theorem.*

Proof. The carrier definition places the terminal Real seal at R and places R after I, F, V, D, L, W, Q in the displayed reading order. Thus a Real-facing consumer must first inspect the source interval, continuous image route, intermediate-value route, Darboux handoff, bracket row, finite windows, and rational readbacks.

The structural rows do not change that order. They transport, replay, source, and name the same packet, but they contain no coordinate for host real equality, quotient representatives, selected preimages, open-cover compactness, countable choice, or ambient completeness.

Therefore the Real seal is exactly a downstream image-value seal for the displayed finite route, not an independent completeness or host-connectedness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalImageConnected}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalImageConnected}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7542.1, Definition 7542.2, Definition 7542.3, and Definition 7542.4 seed the finite interval-image connectedness route from a located interval through continuous interval-image, intermediate-value, Darboux, bracket, finite-window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalImageConnected}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntervalImageConnectedUp.IntervalImageConnectedTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host image-set connectedness, selected preimages, endpoint selection, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exactness, image-route exactness, intermediate-value exactness, Darboux compatibility, Real-seal ordering, and nonescape proofs for BEDC.Derived.IntervalImageConnected

Constructive story: An $\mathsf{IntervalImageConnected}^{\uparrow}$ packet is generated from a located interval row, continuous interval-image row, located or dyadic intermediate-value route, Darboux handoff, bracket row, Real seal, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7543

A Concrete Naming Certificate for $\mathsf{BeurlingLax}^{\uparrow}$

Following Beurling and Lax, the $\mathsf{BeurlingLax}^{\uparrow}$ packet records the finite invariant-subspace rows by which a Hardy-space subspace is read through shift-stability data. It sits beside the Hardy-space boundary carrier of chapter 5108, the bounded-operator surface of chapter 1644, and the operator-neighbour route of chapter 7271. The packet names the displayed Beurling–Lax dependency pattern over finite NameCert rows; it does not import a host Hilbert quotient, a selected inner function, a spectral theorem, orthogonal projection choice, or ambient closed-subspace completeness.

Definition 7543.1 (Beurling–Lax carrier)**.** *A* $\mathsf{BeurlingLax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (H, O, S, I, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{HardySpace}^{\uparrow}$ *source row, exposing the finite coefficient prefix, boundary window, Hardy-control, summation, readback, tolerance, and real-seal rows used by the invariant-subspace read.*
2. *O is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the displayed shift or multiplication action as bounded operator data.*
3. *S is the shift-stability row, recording the finite witnesses that the inspected subspace rows are stable under the displayed operator action.*
4. *I is the inner-divider ledger row. It names a finite divisor or multiplier certificate when such a row is displayed, but it is not a selected inner function.*
5. *W is the finite window row for inspected basis, boundary, or coefficient probes.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficient, boundary, operator, and stability probes.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the Hardy-space, operator, shift-stability, window, and readback rows are visible.*
8. *T records componentwise* hsame *transport for H, O, S, I, W, R, E.*
9. *C records displayed* Cont *replay from Hardy boundary data through bounded shift action, stability, optional divider ledger, readback, and real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BeurlingLax}^{\uparrow}$, $\mathsf{HardySpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{KoopmanOperator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BeurlingLax}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host Hilbert quotient, a chosen orthogonal projection, a selected inner function, full spectral resolution, quotient equality of boundary functions, or ambient Hilbert-space completeness.

Theorem 7543.2 (Beurling–Lax NameCert obligations)**.** *For every accepted* $\mathsf{BeurlingLax}^{\uparrow}$ *carrier* $B = (H, O, S, I, W, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BeurlingLax}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the displayed Hardy-space source, bounded-operator action, shift-stability ledger, optional divider ledger, finite windows, readback, real seal, transport, replay, provenance, and local name;*
2. *Hardy boundary exposure, requiring every invariant-subspace read to consume the* $\mathsf{HardySpace}^{\uparrow}$ *coefficient, boundary-window, Hardy-control, summation, tolerance, and seal rows before it can inspect S;*
3. *bounded shift exposure, requiring the operator row O and the finite stability row S to be read before any invariant-subspace claim is exported;*
4. *readback and seal nonescape, requiring W, R, E to supply the visible coefficient and boundary reads without host quotient equality or selected representatives;*
5. *structural transport, replay, provenance, and naming nonescape through T, C, P, N.*

No obligation is discharged by a host Hilbert-space quotient, a selected inner function, a spectral theorem, orthogonal projection choice, or ambient closed-subspace completeness.

Proof. Project the carrier of Definition 7543.1. The displayed analytic rows are H, O, S, I, W, R, E, and the structural rows are T, C, P, N.

The Hardy-space source H is read first, so the coefficient and boundary data remain those of the finite $\mathsf{HardySpace}^{\uparrow}$ packet. The bounded operator row O and shift-stability row S then record the displayed invariant-subspace action.

The divider ledger I is only a finite certificate row. It cannot choose an inner function because the carrier contains no selector for one. The window, readback, and real-seal rows W, R, E keep the boundary reads inside the L10 surface.

Finally T, C, P, N transport, replay, source, and name the same route. Since no excluded host object appears among the carrier coordinates, the five rows exhaust the local NameCert obligations. □

Theorem 7543.3 (Beurling–Lax Hardy shift route)**.** *Let $B = (H, O, S, I, W, R, E, T, C, P, N)$ be an accepted* $\mathsf{BeurlingLax}^{\uparrow}$ *carrier. Every Hardy-space shift consumer of B first reads the* $\mathsf{HardySpace}^{\uparrow}$ *row H, then reads the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *action row O, then reads the shift-stability row S, and only then may consume the finite divider ledger I, the window and readback rows W, R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E.*

Thus the route

$$\mathsf{HardySpace}^{\uparrow} \longmapsto \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{KoopmanOperator}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRa}$$

is a displayed dependency route for the packet. It is not a claim of host Hilbert quotient reconstruction, selected inner-function factorization, orthogonal projection choice, full spectral theorem, or ambient closed-subspace completeness.

Proof. Start with a shift-facing read of the accepted carrier. The only Hardy-space source is the row H, so the read begins with the finite boundary and coefficient data already exposed by $\mathsf{HardySpace}^{\uparrow}$.

The operator action is then forced through O and its stability evidence through S. The operator-neighbour row is used only as a comparison surface for shift-style reads and does not replace the bounded action row.

Rows I, W, R, E provide the finite divider ledger, inspected windows, regular rational readback, and terminal real seal. Structural rows T, C, P, N preserve this ordering under transport, replay, provenance, and local naming.

Therefore every Hardy-space shift consumer factors through the listed sibling rows, while the excluded Hilbert quotient, selected inner function, projection choice, spectral theorem, and ambient completeness data remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{BeurlingLax}^{\uparrow}$ invariant-subspace read exposes HardySpaceUp source rows, a BoundedLinearOperatorUp action row, shift-stability data, finite windows, RegSeqRatUp readback, DyadicRatCoreUp tolerance, RealUp sealing, transport, replay, provenance, and local NameCert before any shift-stability claim is exported.

Independence witness: hardyspace, boundedlinearoperator, koopmanoperator, streamname, regseqrat, dyadicratcore, real: $\mathsf{BeurlingLax}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Hardy boundary data, bounded shift action, stability, optional divider ledger, finite windows, readback, sealing, transport, replay, provenance, and local naming for an invariant-subspace route.

Closure status for $\mathsf{NameCert}_{\mathsf{BeurlingLax}^{\uparrow}}$.

Constructive story: A $\mathsf{BeurlingLax}^{\uparrow}$ packet is generated from HardySpace rows, a BoundedLinearOperator shift-action row, shift-stability data, an optional divider ledger, finite windows, RegSeqRat readback, DyadicRatCore tolerance, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BeurlingLax}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7543.1, Definition 7543.2, and Definition 7543.3 seed the finite Beurling–Lax invariant-subspace route over Hardy-space, bounded-operator, shift-stability, readback, tolerance, real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BeurlingLax}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Hilbert quotients, selected inner functions, orthogonal projection choice, quotient boundary equality, full spectral theorems, or ambient closed-subspace completeness.

Upgrade path: Obligation closure requires checked carrier admission, Hardy boundary exposure, bounded shift exposure, readback and seal nonescape, structural transport, replay, provenance, and local naming for BEDC.Derived.BeurlingLaxUp.

Chapter 7544

A Concrete Naming Certificate for $\mathsf{CauchyTailStability}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{CauchyTailStability}^{\uparrow}$ packet records the finite-tail stability surface needed before a regular rational stream is admitted to the $\mathsf{Real}^{\uparrow}$ seal. It sits over the regular rational classifier of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the regular Cauchy shift route of chapter 1578. The packet names only finite prefix erasure, tail-window replay, tolerance preservation, and seal handoff; it does not introduce quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7544.1 (Cauchy tail stability carrier)**.** *A* $\mathsf{CauchyTailStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, k, W, R, D, E, H, C, P, N)$$

with the following visible rows:

1. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, read as a regular rational Cauchy name before any finite prefix is ignored.*
2. *k is the displayed finite prefix length whose removal is being certified.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, sending each later window request to the source window beginning after k.*
4. *R is the replayed regular-rational classifier for the tail stream.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger witnessing that the tail replay preserves the same dyadic error budgets.*
6. *E is the exclusion ledger for quotient streams, selected limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
7. *H records componentwise* hsame *transport for source, prefix length, tail window, classifier, dyadic ledger, exclusion ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail-stability request through Q, k, W, R, D, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailStability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, k, W, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, countable-choice tail selectors, trichotomy, host real equality, or an ambient completeness proof for $\mathsf{Real}^{\uparrow}$.

Theorem 7544.2 (Cauchy tail stability NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailStability}^{\uparrow}$ *carrier $T = (Q, k, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailStability}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-prefix exposure, tail-window replay, regular-rational classifier retention, dyadic tolerance preservation, exclusion-ledger nonescape, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the carrier. The source regular rational row, finite prefix length, tail-window row, replayed classifier, dyadic ledger, exclusion ledger, and structural rows are all displayed coordinates.

The finite-prefix exposure and tail-window replay are read through k and W. A consumer cannot inspect the tail before the prefix length is visible, and it cannot detach the replayed window from the source row Q.

The regular-rational classifier and dyadic tolerance obligations are read through R and D. They keep the tail stream attached to the same regular Cauchy route and the same dyadic budget surface.

Finally, E, H, C, P, N record nonescape, transport, replay, provenance, and naming. Since no row stores quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7544.3 (Cauchy tail stability Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyTailStability}^{\uparrow}$ *carrier receives the tail-stable regular Cauchy name only through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegularCauchyShift}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

as displayed by $Q, k, W, R, D, E, H, C, P, N$. Finite prefixes therefore cannot affect the admitted Real seal, and the handoff exports no quotient stream, selected representative, selected limit, countable-choice schedule, trichotomy principle, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the source row Q and prefix length k. The packet exposes the regular rational Cauchy name before it exposes the tail, so the removed prefix is finite and named.

Read the tail-window row W and the replayed classifier row R. Each requested tail window is a displayed source window after k, and the classifier remains the regular-rational classifier attached to that replay.

Read the dyadic ledger D before the real seal. It records the same tolerance budgets for the tail route, so the $\mathsf{Real}^{\uparrow}$ consumer sees a regular Cauchy readback with preserved dyadic control.

The remaining rows transport, replay, source, and name the route, while the exclusion ledger blocks every non-displayed exit. Thus finite prefixes cannot change the admitted Real seal inside this packet. □

Falsifiable prediction: Any consumer that deletes k or W can no longer distinguish a finite-prefix read from an unrecorded quotient-stream identification.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-shift: $\mathsf{CauchyTailStability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source regular rational row, displayed prefix length, tail-window replay, dyadic tolerance ledger, exclusion ledger, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailStability}^{\uparrow}$ packet is generated from a source regular rational Cauchy row, a finite prefix length, StreamName tail windows, a replayed RegSeqRat classifier, a DyadicRatCore tolerance ledger, an exclusion ledger, componentwise transport, displayed replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7544.1, Definition 7544.2, and Definition 7544.3 seed the finite-tail stability route from regular rational names to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix exposure, tail-window replay, dyadic tolerance preservation, Real-seal handoff, and exclusion-ledger nonescape for BEDC.Derived.CauchyTailStabilityUp.

Chapter 7545

A Concrete Naming Certificate for $\mathsf{CriticalStripOscillation}^{\uparrow}$

Following Riemann, Hadamard, de la Vallee Poussin, and the constructive contour discipline used in the RH route, the $\mathsf{CriticalStripOscillation}^{\uparrow}$ packet records a finite zero-window boundary inside the critical strip. It sits beside the RH boundary of chapter 9239, the route-halting boundary of chapter 9336, the zeta-continuation application of chapter 9342, and the critical-strip dependency named in chapter 72. The packet is a local oscillation-window certificate only; it does not prove RH, refute RH, move a zero to the critical line, assert zero-free regions beyond the displayed window, or decide any global zeta-zero statement.

Definition 7545.1 (Critical-strip oscillation carrier)**.** *A* $\mathsf{CriticalStripOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, Z, W, B, A, E, H, C, P, N)$$

with the following visible rows:

1. *S is the displayed critical-strip window row, naming a bounded rational rectangle inside $0 < \Re(s) < 1$ and excluding the pole at $s = 1$.*
2. *Z is the zeta-continuation source row, read only through the finite continuation and factor ledgers already exposed by the RH route.*
3. *W is the finite zero-window row listing inspected subwindows, sample histories, and local sign or argument-variation observations.*
4. *B is the boundary ledger requiring every inspected zero claim to remain inside the displayed window S.*
5. *A is the oscillation ledger recording finite argument-variation or sign-change budget data available on the inspected boundary.*
6. *E is the exclusion ledger refusing RH, $\neg\mathsf{RH}$, global zero-free regions, unrestricted analytic continuation, hidden contour integrals, selected zeros, and oracle decisions.*
7. *H records componentwise* hsame *transport for strip window, zeta source, zero window, boundary, oscillation, exclusion, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a zero-window request through S, Z, W, B, A, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CriticalStripOscillation}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CriticalStripOscillation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, Z, W, B, A, E, H, C, P, N$. It has no coordinate for a proof of RH, a disproof of RH, a global zero classifier, selected zeros, unrestricted contour evaluation, or a decision oracle.

Theorem 7545.2 (Critical-strip oscillation NameCert obligations)**.** *For every accepted* $\mathsf{CriticalStripOscillation}^{\uparrow}$ *carrier $O = (S, Z, W, B, A, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CriticalStripOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission, critical-strip window exposure, zeta-source retention, finite zero-window coverage, boundary nonescape, oscillation-budget replay, exclusion-ledger refusal, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the carrier. The strip window, zeta source, zero-window ledger, boundary ledger, oscillation ledger, exclusion ledger, and structural rows are all displayed coordinates.

The zero-window obligations are local. A consumer first reads the bounded strip window S, then the zeta source Z, and only then the inspected subwindows and observations in W.

Boundary nonescape and oscillation replay are read through B and A. They keep every inspected zero or variation claim attached to the same finite window and prevent it from being promoted to a global zero theorem.

The exclusion and structural rows then close the local NameCert surface. Since no row stores RH, $\neg\mathsf{RH}$, a selected zero, unrestricted contour evaluation, or a decision oracle, the listed obligations exhaust the packet. □

Theorem 7545.3 (Critical-strip zero-window nonescape)**.** *Every zero-window read exported by an accepted* $\mathsf{CriticalStripOscillation}^{\uparrow}$ *carrier remains inside the displayed critical-strip window S and the finite observation ledger W. The packet may report local oscillation, boundary variation, and inspected zero-window data, but it does not prove RH, refute RH, assert that all critical-strip zeros lie on the critical line, claim a global zero-free region, select a zero outside the window, or decide any RH condition chain.*

Proof. Start with the strip row S. It is the only domain row exposed by the carrier, so every zero-window read is bounded by the displayed rational rectangle.

Read the zeta source and zero-window rows next. The route through Z and W supplies only the finite continuation source and inspected subwindows available in the packet.

The boundary and oscillation ledgers B and A keep the read local. They record window containment and finite variation data, not a global classification of zeros.

Finally apply the exclusion ledger E. It refuses RH decisions, selected off-window zeros, global zero-free regions, unrestricted contour evaluation, and oracle routes, so the zero-window read cannot escape the finite oscillation packet. □

Falsifiable prediction: A consumer that removes the boundary ledger B can turn a finite inspected window into an unbounded zero claim, which is rejected by the $\mathsf{CriticalStripOscillation}^{\uparrow}$ carrier.

Independence witness: critical-strip, zeta-continuation, rh-route-halting-boundary, rh-zero-window: $\mathsf{CriticalStripOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded strip window, zeta source, inspected zero-window ledger, oscillation budget, boundary nonescape ledger, exclusion ledger, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CriticalStripOscillation}^{\uparrow}}$.

Constructive story: A CriticalStripOscillation$^{\uparrow}$ packet is generated from a bounded critical-strip window, zeta-continuation source row, finite zero-window ledger, boundary ledger, oscillation budget, exclusion ledger, componentwise transport, displayed replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{CriticalStripOscillation}\uparrow}$, seed)

Scope closed: Definition 7545.1, Definition 7545.2, and Definition 7545.3 seed the finite zero-window boundary used by the RH route.

Formal verification: FormalStatus(NameCert$_{\text{CriticalStripOscillation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RH, ¬RH, global zero-free regions, unrestricted analytic continuation, selected zeros, oracle decisions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, strip-window exposure, zeta-source retention, finite zero-window coverage, oscillation-budget replay, and exclusion-ledger refusal for BEDC.Derived.CriticalStripOscillationUp.

Chapter 7546

A Concrete Naming Certificate for ClassifierFormationTrace$^{\uparrow}$

The ClassifierFormationTrace$^{\uparrow}$ packet realizes the learning classifier shift of Definition 9462.356 as finite before-and-after classifier data. It refines the broader ClassifierFormingLearning$^{\uparrow}$ route of chapter 6097 by naming the trace row where a learning transition exposes source scope, before classifier, after classifier, ledger residue, audit export, and local NameCert obligations. The packet records classifier formation as a finite trace; it does not claim model-family correctness, benchmark validity, causal explanation, positive net information, or metric improvement as learning by itself.

Definition 7546.1 (Classifier-formation trace carrier)**.** *A* ClassifierFormationTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, B, A, L, R, G, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-scope row naming the task, observations, training protocol, and common comparison domain for the before and after classifiers.*
2. *B is the before-classifier row read from the pre-transition learning classifier state.*
3. *A is the after-classifier row read from the post-transition learning classifier state.*
4. *L is the classifier-shift ledger row, recording that B and A do not agree on the displayed common source scope.*
5. *R is the learning-residue row for parameter motion, loss movement, benchmark evidence, memorization, shortcut behavior, and distribution-boundary residue.*
6. *G is the registry and audit export row inherited from* RegistryExportSurface$^{\uparrow}$, MachineReadableAuditInterface$^{\uparrow}$, *and* GroundCompilerAuditMap$^{\uparrow}$.
7. *Q is the TasteGate commitment row requiring later Lean work to provide* Nontrivial ClassifierFormationTrace$^{\uparrow}$ *and* FieldFaithful ClassifierFormationTrace$^{\uparrow}$ *instances for this trace carrier.*
8. *H records componentwise* hsame *transport for S, B, A, L, R, G, Q.*
9. *C records displayed* Cont *replay from source scope through before classifier, after classifier, shift ledger, residue, audit export, TasteGate commitments, provenance, and local naming.*
10. *P is the* Pkg *provenance over* ClassifierFormationTrace$^{\uparrow}$, ClassifierFormingLearning$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, MachineReadableAuditInterface$^{\uparrow}$, GroundCompilerAuditMap$^{\uparrow}$, BHist, hsame, Cont, Pkg, TasteGate, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierFormationTrace}^{\uparrow}}$ *row.*

The carrier compares accepted traces only by $S, B, A, L, R, G, Q, H, C, P, N$. Parameter motion, loss reduction, and benchmark movement remain residue rows unless L displays classifier shift between the before and after rows on the same source scope.

Theorem 7546.2 (Classifier-formation trace NameCert obligations)**.** *For every accepted* $\mathsf{ClassifierFormationTrace}^{\uparrow}$ *carrier $T = (S, B, A, L, R, G, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClassifierFormationTrace}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for source scope, before classifier, after classifier, shift ledger, residue ledger, audit export, TasteGate commitments, transport, replay, provenance, and local naming;*
2. *before/after classifier exposure, requiring B and A to be read on the same displayed source scope S;*
3. *ledger-residue separation, requiring classifier shift through L and keeping parameter motion, loss movement, benchmark evidence, memorization, shortcuts, and distribution residue in R;*
4. *audit and TasteGate handoff through G, Q, with nontriviality and field-faithfulness instances left as explicit Lean obligations for the trace carrier;*
5. *structural transport, replay, provenance, and naming nonescape through H, C, P, N.*

No obligation is discharged by parameter motion alone, lower loss alone, benchmark improvement alone, memorized examples, shortcut performance, or unregistered interpretability prose.

Proof. Project the carrier of Definition 7546.1. The source and classifier rows are S, B, A, while L records the displayed classifier shift on the same scope.

The residue row R is read separately. It may record parameter, loss, benchmark, memorization, shortcut, or distribution-boundary evidence, but none of those rows is the classifier-shift row.

The audit row G and TasteGate row Q make the trace exportable and record the later Lean obligations for nontriviality and field faithfulness. These commitments attach to the same finite carrier rather than to a private training narrative.

Finally H, C, P, N transport, replay, source, and name the same route. Since the excluded metric and prose substitutes have no carrier coordinate that can replace L, the listed rows exhaust the local NameCert surface. □

Theorem 7546.3 (Classifier-formation trace shift witness)**.** *Let $T = (S, B, A, L, R, G, Q, H, C, P, N)$ be an accepted* $\mathsf{ClassifierFormationTrace}^{\uparrow}$ *carrier. A learning transition exports classifier formation through T only when the before classifier B and after classifier A are compared on the source row S, the shift ledger L records nonagreement on that scope, the residue row R keeps training side effects separate, and the audit and TasteGate rows G, Q are displayed before local naming through N.*

Thus the trace realizes the route from Definition 9462.356 as finite NameCert data. It does not assert positive net explanation information, benchmark validity, causal explanation, field faithfulness before the later Lean instance, or nontriviality before the later Lean instance.

Proof. Begin with the accepted trace carrier. The source row S fixes the common domain on which the before and after classifier rows can be compared.

The rows B, A, L then form the classifier-shift witness. Without L, the transition remains a pair of classifier rows and does not export certified classifier formation.

The residue row R keeps parameter motion, loss movement, benchmark evidence, memorization, shortcut behavior, and distribution residue outside the classifier-shift claim. The audit row G and TasteGate row Q preserve the public-checking and later-instance boundary.

Transport and replay through H, C preserve the row order, and P, N source and name the trace. Therefore the accepted packet realizes classifier shift as finite NameCert data without claiming the excluded metric, causal, positive-information, or completed TasteGate facts. □

Falsifiable prediction: A $\mathsf{ClassifierFormationTrace}^{\uparrow}$ packet with before and after classifier rows but no same-scope shift ledger is rejected, even if it records parameter motion, lower loss, benchmark movement, memorization evidence, or audit prose.

Independence witness: classifier-forming-learning, classifier-shift-ledger-residue, namecert, pkg, bhist: $\mathsf{ClassifierFormationTrace}^{\uparrow}$ is not bijective to its listed neighbors because it jointly carries source scope, before classifier, after classifier, classifier-shift ledger, residue separation, audit export, TasteGate commitments, transport, replay, provenance, and local naming as one finite trace.

Later Lean TasteGate work must provide $\mathsf{Nontrivial\ ClassifierFormationTrace}^{\uparrow}$ and $\mathsf{FieldFaithful\ ClassifierFormatio}$ instances for the carrier of Definition 7546.1. Those instances must preserve the rows $S, B, A, L, R, G, Q, H, C, P, N$ so classifier formation remains separated from training residue.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierFormationTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierFormationTrace}^{\uparrow}$ packet is generated from source scope, before classifier, after classifier, classifier-shift ledger, learning-residue rows, registry and audit export, TasteGate commitments, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierFormationTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7546.1, Definition 7546.2, and Definition 7546.3 seed the finite trace route for learning classifier shift over before/after classifier rows, ledger residue, audit export, TasteGate commitments, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierFormationTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, model-family correctness, benchmark validity, causal explanation, positive net information, Nontrivial instances, FieldFaithful instances, or metric improvement as learning by itself.

Upgrade path: Obligation closure requires checked carrier admission, before/after classifier exposure, ledger-residue separation, audit export, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.ClassifierFormationTraceUp.

Chapter 7547

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ packet records the finite cotransitivity route for apartness between regular Cauchy real names before any trichotomy or quotient-real equality is exposed. It sits over the positive apartness boundary of chapter 1710, the located comparison route of chapter 1681, the dyadic rational core of chapter 291, the finite stream windows of chapter 126, the regular rational sequence readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a finite cotransitivity surface for displayed regular Cauchy names; it is not a trichotomy principle, not quotient real equality, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7547.1 (Regular Cauchy apartness cotransitivity carrier)**.** *A* $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, A, O, L, R, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose apartness is already displayed by regular Cauchy data.*
2. *Z is the third* $\mathsf{RegSeqRat}^{\uparrow}$ *row through which cotransitivity is read.*
3. *A is the* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *source-apartness row for X and Y.*
4. *O is the* $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ *comparison row that keeps the branch read located over the same regular Cauchy windows.*
5. *L is the left branch row, witnessing apartness from X to Z when that branch is displayed.*
6. *R is the right branch row, witnessing apartness from Z to Y when that branch is displayed.*
7. *W records the common* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule inspected by the source apartness row and by the active branch.*
8. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-gap ledger, tolerance budget, and slack consumed by the source apartness and branch rows.*
9. *Q records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback handoff for the three regular rational names.*

10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, O, L, R, W, D, Q have been displayed.*
11. *H stores componentwise* hsame *transport for the source, third-name, apartness, located-order, branch, window, dyadic, readback, seal, provenance, and local naming rows.*
12. *C stores displayed* Cont *replay for the same cotransitivity route.*
13. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$, $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$, $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, Z, A, O, L, R, W, D, Q, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected limit representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.RegularCauchyApartnessCotransitivityUp.RegularCauchyApartnessCotransitivityTasteGa
single carrier alignment

Theorem 7547.2 (Regular Cauchy apartness cotransitivity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ *carrier $K = (X, Y, Z, A, O, L, R, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}}$ *surface consists exactly of carrier admission for K, source apartness admission through A, located-order admission through O, one displayed cotransitivity branch through L or R, common window admission through W, dyadic positive-gap admission through D, regular rational readback through Q, terminal Real seal handoff through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 7547.1. The displayed tuple lists every row available to a consumer of the accepted packet.

The apartness and located-order rows supply the constructive middle of the route. The source apartness row A fixes the positive comparison of X and Y, while O keeps the branch comparison located over the same finite regular Cauchy windows.

The branch rows are exclusive as displayed data. A consumer may read the left branch L or the right branch R, but the packet does not export a separate trichotomy branch or a choice function selecting among all future comparisons.

The window, dyadic, readback, and Real-seal rows keep the branch finite. The common schedule W, positive-gap ledger D, regular rational handoff Q, and seal E must be read in the same carrier route before the result is offered to Real or Completion consumers.

Finally, H, C, P, N transport, replay, source, and name the same rows. Since the excluded principles are absent from the carrier inventory, the listed obligations exhaust the local NameCert surface. □

Theorem 7547.3 (Regular Cauchy apartness cotransitivity route)**.** *Every accepted* RegularCauchyApartnessCotrans
read with source apartness between X and Y and third regular Cauchy name Z reaches a Real-facing consumer only through the finite route

$$\mathsf{RegularCauchyApartnessOrder}^{\uparrow} \longmapsto \mathsf{RegularCauchyLocatedOrder}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The consumer receives exactly one displayed branch: either the left branch witnessing X apart from Z, or the right branch witnessing Z apart from Y. The route exports no trichotomy of X, Z, Y, no quotient real equality, no selected limit, no countable-choice branch selector, no host real equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier $K = (X, Y, Z, A, O, L, R, W, D, Q, E, H, C, P, N)$. The source apartness is exposed through A, and the third regular Cauchy name is exposed through Z and the readback row Q.

Use the located-order row O to keep the branch read attached to the same regular Cauchy windows. If L is the displayed branch, the route witnesses apartness from X to Z; if R is the displayed branch, the route witnesses apartness from Z to Y.

The finite evidence is then read through W, D, Q. The stream schedule fixes the inspected windows, the dyadic ledger supplies the positive-gap and slack data, and the regular rational readback keeps all three names inside displayed regular Cauchy data.

Only after those rows are displayed may the terminal Real seal E be consumed. The structural rows H, C, P, N preserve and name the same finite route, so the Real-facing consumer receives a cotransitivity handoff rather than trichotomy, quotient equality, selected limits, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ read factors through source RegSeqRat rows, one third RegSeqRat row, a RegularCauchyApartnessOrder row, a RegularCauchyLocatedOrder row, one displayed cotransitivity branch, StreamName windows, a DyadicRatCore positive-gap ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regular-cauchy-apartness-order, regular-cauchy-located-order, dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source apartness row, a third regular Cauchy name, one displayed cotransitivity branch, the common window and dyadic positive-gap ledgers, regular rational readback, and the Real seal exclusions that keep cotransitivity separate from trichotomy and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}$ packet is generated from source RegSeqRat rows, one third RegSeqRat row, a RegularCauchyApartnessOrder source-apartness row, a RegularCauchyLocatedOrder comparison row, one displayed left-or-right branch, StreamName windows, a DyadicRatCore positive-gap ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7547.1, Definition 7547.2, and Definition 7547.3 seed the finite cotransitivity route for regular Cauchy apartness through displayed branch, located-order, window, dyadic, readback, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCotransitivity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source apartness admission, located-order admission, branch stability, dyadic positive-gap preservation, RegSeqRat readback, Real-seal non-escape, and ledger exclusion for BEDC.Derived.RegularCauchyApartnessCotransitivityUp.

Chapter 7548

A Concrete Naming Certificate for $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$ packet records the finite grid on which bounded variation, Riemann-sum, and compact-interval oscillation consumers inspect an interval function before any ambient real-completeness principle is available. It sits over the compact interval source of chapter 3986, the closed endpoint boundary of chapter 1591, the dyadic tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite variation grid only; it does not choose an optimal partition, quotient real values, or appeal to open-cover compactness.

Definition 7548.1 (Compact interval variation grid carrier)**.** *A* $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, K, M, D, V, W, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval-source row, exposing the closed bounded endpoint data through chapter 1591.*
2. *K is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *row, recording the compact-interval finite inspection surface attached to I.*
3. *M is the finite mesh-grid row, listing ordered cells and adjacent endpoints over the displayed interval source.*
4. *D is the dyadic tolerance row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, for mesh width, local oscillation budgets, and variation comparison budgets.*
5. *V is the finite variation ledger, recording the displayed adjacent changes and the finite sum row that a variation consumer may inspect.*
6. *W is the window-read row, carrying the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used at the mesh endpoints and at any displayed refinement cell.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the terminal scalar comparisons exported by the packet.*
8. *H is the componentwise* hsame *transport row for interval source, compact row, mesh, dyadic tolerances, variation ledger, window reads, real seal, replay, provenance, and local naming.*
9. *C is the displayed* Cont *replay from an interval-grid request through $I, K, M, D, V, W, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalVariationGrid}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected partition outside the displayed grid, quotient equality of real values, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7548.2 (Compact interval variation grid NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$ *carrier*

$$G = (I, K, M, D, V, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalVariationGrid}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *interval and compact-source admission through I, K;*
2. *finite mesh-grid exposure through M, with every endpoint and adjacent cell read over the same displayed interval source;*
3. *dyadic tolerance and variation-ledger exactness through D, V;*
4. *regular readback and real-seal handoff through W, R; and*
5. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No variation-grid obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 7548.1. The interval source and Bishop compact interval row are the first displayed coordinates, so every consumer reads I, K before it can inspect the grid.

The finite mesh obligation is projection to M. Because the mesh row is attached to the same interval source, endpoint and adjacent-cell reads cannot be detached from the closed bounded interval boundary.

The numerical ledger is read from D, V. The dyadic row records the tolerances and comparison budgets, while the variation row records the finite adjacent changes and their finite sum.

The readback and scalar handoff are read from W, R. Regular rational windows supply the finite endpoint observations, and the Real row seals only the terminal scalar comparisons already exposed by those windows.

Finally, H, C, P, N exhaust transport, replay, provenance, and local naming. With no further coordinate, the certificate cannot export open-cover compactness, a selected partition outside the grid, quotient equality of real values, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalVariationGrid}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactIntervalVariationGrid}^{\uparrow}}$, seed)

Scope closed: Definition 7548.1 and Definition 7548.2 seed the finite compact-interval variation grid surface over interval source, compact row, mesh, dyadic tolerances, variation ledger, regular readbacks, real seal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactIntervalVariationGrid}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected partitions outside the displayed mesh, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval-source admission, compact-row attachment, mesh-grid exactness, dyadic tolerance exactness, variation-ledger exactness, regular readback, real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalVariationGridUp.

Constructive story: A $\mathsf{CompactIntervalVariationGrid}^{\uparrow}$ packet is generated from interval source, BishopCompactInterval, mesh-grid, DyadicRatCore, finite variation ledger, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows.

Chapter 7549

A Concrete Naming Certificate for $\mathsf{EuclideanAlgorithm}^{\uparrow}$

Following Euclid and Knuth, the $\mathsf{EuclideanAlgorithm}^{\uparrow}$ packet records the finite descent trace behind a greatest-common-divisor computation as displayed BEDC data. It consumes the unary natural-number surface of chapter 28, the signed coefficient rows of chapter 31, the multiplication-witness divisibility boundary of chapter 331, the backtracking route of chapter 6993, and the division-with-remainder interface of chapter 7053. The chapter names the algorithm only as a finite history packet: input rows, quotient and remainder rows, a strict natural-valued descent budget, terminal divisor data, Bezout backtracking, transport, replay, provenance, and local naming all remain visible.

Definition 7549.1 (Euclidean algorithm carrier)**.** *A* $\mathsf{EuclideanAlgorithm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, S, B, T, Z, H, C, P, N)$$

with the following displayed rows:

1. *I is the input row, naming the two displayed arithmetic inputs through* $\mathsf{Nat}^{\uparrow}$ *rows, with signed readback through* $\mathsf{Int}^{\uparrow}$ *only when coefficient consumers ask for it.*
2. *S is the finite step ledger. Each row of S records one quotient-and-remainder step, its dividend and divisor rows, the quotient row, the remainder row, and the local division-with-remainder evidence read through* $\mathsf{EuclideanDomain}^{\uparrow}$*.*
3. *B is the strict* $\mathsf{Nat}^{\uparrow}$*-valued descent budget. It assigns a displayed natural value to every non-terminal divisor row and records that each nonzero remainder row in S has strictly smaller budget.*
4. *T is the terminal divisor row. It records the first step whose remainder row is the displayed zero row and stores the terminal divisor as the candidate gcd row.*
5. *Z is the Bezout backtracking route. It reads the terminal divisor and the reversed finite step ledger through* $\mathsf{BezoutIdentity}^{\uparrow}$ *and the* $\mathsf{Divisibility}^{\uparrow}$ *boundary.*
6. *H records componentwise* hsame *transport for input rows, step rows, budget rows, terminal divisor rows, Bezout backtracking, continuation replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from the input row through the step ledger, descent budget, terminal divisor, and Bezout backtracking route.*

8. *P is the* Pkg *provenance over* $\mathsf{EuclideanAlgorithm}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Divisibility}^{\uparrow}$, $\mathsf{BezoutIdentity}^{\uparrow}$, $\mathsf{EuclideanDomain}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{EuclideanAlgorithm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by replaying these displayed coordinates. It has no coordinate for a hidden host gcd routine, a selected quotient stream, quotient equality of ideals, a normalized gcd convention, global well-ordering, or an off-packet Euclidean-domain theorem.

Lean4 (definition) `BEDC.Derived.EuclideanAlgorithmUp.EuclideanAlgorithmUp`
Lean4 ✓ `BEDC.Derived.EuclideanAlgorithmUp.EuclideanAlgorithmTasteGate single carrier alignment`

Theorem 7549.2 (Euclidean algorithm NameCert obligations)**.** *For every accepted* $\mathsf{EuclideanAlgorithm}^{\uparrow}$ *carrier*

$$A = (I, S, B, T, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EuclideanAlgorithm}^{\uparrow}}$ *surface consists exactly of input-row admission, quotient-and-remainder step admission through S, strict* $\mathsf{Nat}^{\uparrow}$*-budget admission through B, terminal divisor admission through T, Bezout and divisibility backtracking through Z, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EuclideanAlgorithmUp.EuclideanAlgorithmCarrier namecert obligations`

Proof. Project the carrier of Definition 7549.1. The input row I supplies the two displayed arithmetic inputs, while the step ledger S supplies every quotient-and-remainder row consumed by the computation.

Read the budget and terminal rows next. The row B is the only coordinate that records strict descent, and the row T is the only coordinate that records the terminal zero-remainder step and its divisor output.

Finally read the backtracking and structural rows. The row Z routes the reversed finite trace through $\mathsf{BezoutIdentity}^{\uparrow}$ and $\mathsf{Divisibility}^{\uparrow}$, while H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a host gcd routine, quotient stream, ideal quotient equality, normalized-gcd convention, global well-ordering, or detached Euclidean-domain theorem, none of those objects belongs to the local NameCert surface. □

Theorem 7549.3 (Euclidean algorithm descent budget)**.** *Every accepted* $\mathsf{EuclideanAlgorithm}^{\uparrow}$ *carrier exposes termination only through the finite budget row B. If a nonterminal step of S has divisor row d and nonzero remainder row r, then the budget ledger records a strict* $\mathsf{Nat}^{\uparrow}$ *comparison from the value attached to r below the value attached to d. The terminal row T may be read only after this finite descent ledger reaches a displayed zero-remainder step.*

Proof. Induct over the finite step ledger S. In the base case, the first displayed step is read with its divisor, quotient, remainder, and local division evidence, and the budget row B supplies the comparison attached to that step.

For the induction step, assume the current nonterminal row exposes a nonzero remainder. The next row of S uses that remainder as the new divisor row, and B records the strict $\mathsf{Nat}^{\uparrow}$ descent from the old divisor budget to the new divisor budget.

Because the induction ranges only over the displayed finite ledger, it cannot appeal to a host well-ordering or to an unlisted search procedure. The terminal divisor row T is available exactly when the next inspected remainder is the displayed zero row, and the Bezout row Z can then replay the reversed finite ledger without adding another termination principle. □

Falsifiable prediction: Every accepted $\mathsf{EuclideanAlgorithm}^{\uparrow}$ read must expose the input row, every quotient-and-remainder step row, the strict $\mathsf{Nat}^{\uparrow}$ descent budget, the terminal divisor row, the Bezout backtracking row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before it may be consumed as a gcd or Bezout computation.

Independence witness: nat, int, divisibility, bezout-identity, euclidean-domain: $\mathsf{EuclideanAlgorithm}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the input rows, the whole finite quotient-and-remainder trace, a strict natural-valued descent budget, the terminal divisor row, and the Bezout backtracking route as one algorithm packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EuclideanAlgorithm}^{\uparrow}}$.

Constructive story: A $\mathsf{EuclideanAlgorithm}^{\uparrow}$ packet is generated from input rows, finite quotient-and-remainder step rows, a strict NatUp-valued descent budget, a terminal divisor row, Bezout backtracking, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EuclideanAlgorithm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7549.1, Definition 7549.2, and Definition 7549.3 seed the finite Euclidean descent trace over NatUp, IntUp, DivisibilityUp, BezoutIdentityUp, and EuclideanDomainUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EuclideanAlgorithm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unique gcd normalization, ideal quotient equality, a selected host algorithm, global well-ordering, or a Euclidean-domain theorem beyond the displayed finite trace.

Upgrade path: Obligation closure requires checked input admission, step-ledger exactness, strict descent, terminal divisor exposure, Bezout backtracking, transport, replay, provenance, and local naming for BEDC.Derived.EuclideanAlgorithmUp.

Chapter 7550

A Concrete Naming Certificate for $\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}$ packet records the finite surface by which located interval data yield an explicit dyadic modulus row for later real-name and regular-Cauchy consumers. It sits after the dyadic endpoint ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the located completion boundary of chapter 6253, the compact interval modulus surface of chapter 5937, and the choice-free completion route of chapter 4237. The packet extracts finite modulus rows from displayed located endpoint and tolerance data; it does not assert Heine–Borel compactness, open-cover compactness, selected representatives, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7550.1 (Located interval modulus extraction carrier)**.** *A* $\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, G, W, T, M, H, C, P, N)$$

with the following displayed rows:

1. *E is the located interval endpoint row, listing lower and upper dyadic endpoint reads together with the finite locatedness witnesses that separate an external rational probe from the interval boundary.*
2. *G is the dyadic grid-window row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh levels and endpoint refinements through which the interval is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to request only finitely many endpoint and regular-name observations.*
4. *T is the regular-Cauchy tolerance row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and carrying the requested precision scale for each displayed window.*
5. *M is the extracted modulus row, assigning to each displayed tolerance request a dyadic grid window sufficient for the located interval read.*
6. *H records componentwise* hsame *transport for endpoint rows, dyadic grid windows, StreamName windows, regular-Cauchy tolerance rows, extracted modulus rows, replay, provenance, and local naming.*

7. *C records displayed* Cont *replay from a modulus request through E, G, W, T, M, H, P, N.*

8. *P is the* Pkg *provenance over* LocatedIntervalModulusExtraction$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, LocatedIntervalCompletion$^{\uparrow}$, CompactIntervalModulus$^{\uparrow}$, RegularCauchyChoiceFreeCompletion$^{\uparrow}$ BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{LocatedIntervalModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $E, G, W, T, M, H, C, P, N$. It has no coordinate for open covers, selected real representatives, quotient equality of real names, trichotomy, countable choice, or a completeness principle for Real$^{\uparrow}$.

Theorem 7550.2 (Located interval modulus extraction obligations)**.** *For every accepted* LocatedIntervalModulusEx *carrier*

$$K = (E, G, W, T, M, H, C, P, N),$$

the local NameCert$_{\text{LocatedIntervalModulusExtraction}^{\uparrow}}$ *surface consists exactly of finite located endpoint exposure through E, dyadic grid-window exposure through G, finite observation scheduling through W, regular-Cauchy tolerance exposure through T, extracted modulus rows through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Every consumer of the extracted modulus first reads the sibling route

DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, LocatedIntervalCompletion$^{\uparrow}$, CompactIntervalModulus$^{\uparrow}$, Regula

and receives only the displayed finite rows of that route. The packet does not export Heine–Borel compactness, open-cover compactness, selected representatives, countable choice, quotient real equality, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7550.1. Rows E, G, W, T, M are the endpoint, dyadic-window, finite-observation, tolerance, and modulus coordinates, while H, C, P, N transport, replay, source, and name the same finite packet.

Read the endpoint and grid rows first. The dyadic endpoint ledger is supplied by chapter 291, and the finite observation schedule is supplied by chapter 126; hence a modulus request can inspect only displayed dyadic windows and listed observations.

Next read the regular-Cauchy and real-name rows. The tolerance row T is the RegSeqRat$^{\uparrow}$ source route of chapter 301, and its terminal real-facing seal is the Real$^{\uparrow}$ row of chapter 38. The located interval completion route of chapter 6253 keeps the endpoint data attached to that same finite read.

Finally use the modulus and completion siblings. The finite modulus surface is the one exposed by chapter 5937, and the source-side completion route is the choice-free packet of chapter 4237. Since none of the excluded principles appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: Every accepted LocatedIntervalModulusExtraction$^{\uparrow}$ read exposes located endpoint rows, dyadic grid windows, StreamName observations, RegSeqRat tolerance rows, extracted modulus rows, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing real seal consumes the extracted modulus.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-interval-completion, compact-interval-modulus, regular-cauchy-choice-free-completion: none is bijective with LocatedIntervalModulusExt because this packet alone binds finite located endpoint rows, dyadic grid windows, regular-Cauchy

tolerance requests, extracted modulus rows, and structural transport into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}$ packet is generated from located interval endpoint rows, dyadic grid-window rows, StreamName finite observations, RegSeqRat tolerance rows, extracted modulus rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7550.1 and Definition 7550.2 seed the finite located interval modulus extraction surface over dyadic, stream, regular-Cauchy, real, located-completion, compact-modulus, and choice-free completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, open-cover compactness, selected representatives, countable choice, quotient real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-row exposure, dyadic grid-window coverage, regular-Cauchy tolerance extraction, modulus-row exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalModulusExtractionUp.

Chapter 7551

A Concrete Naming Certificate for $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$

Following Hardy's summability theory, the $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$ packet records the finite Cauchy–Stolz transform route by which a regular rational sequence is inspected through displayed numerator differences, denominator growth rows, and Stolz–Cesaro comparison windows before a real-facing consumer reads the transformed endpoint. It sits after the regular rational readback of chapter 301, the finite observation discipline of chapter 126, the rational arithmetic carrier of chapter 37, the terminal real-name seal of chapter 38, and the Stolz–Cesaro sibling of chapter 6137. The packet names a finite L10 summability transform; it does not claim host quotient equality of real names, selected limits, trichotomy, countable choice, or classical completeness.

Definition 7551.1 (Regular Cauchy–Stolz transform carrier)**.** *A* $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, Q, D, A, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational windows supply the inspected sequence values.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the source indices, successor indices, and transform windows used by the comparison.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *arithmetic row for numerator differences, denominator differences, and quotient comparisons.*
4. *D is the positive denominator-growth row, recording that the inspected denominator windows are ordered and never read through an invisible zero divisor.*
5. *A is the finite difference-ratio row comparing $(a_{n+1} - a_n)$ with $(b_{n+1} - b_n)$ over the displayed windows.*
6. *Z is the* $\mathsf{StolzCesaroTheorem}^{\uparrow}$ *handoff row, used only after R, W, Q, D, A have been exposed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available to downstream summability consumers after the regular source, window, arithmetic, growth, and Stolz rows are visible.*
8. *H records componentwise* hsame *transport for the source, window, rational arithmetic, denominator-growth, difference-ratio, Stolz handoff, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through the same finite transform route.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StolzCesaroTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyStolzTransform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, Q, D, A, Z, E, H, C, P, N$. It has no coordinate for host quotient equality of real names, an unlogged infinite matrix transform, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7551.2 (Regular Cauchy–Stolz transform NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$ *carrier $S = (R, W, Q, D, A, Z, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyStolzTransf}}$ *surface consists exactly of carrier admission for the displayed regular rational source, finite windows, rational arithmetic, denominator-growth row, difference-ratio row, Stolz handoff, real seal, componentwise transport, displayed replay, provenance, and local naming. Every transform read must expose R, W, Q, D, A before the Stolz–Cesaro handoff Z and the real seal E are available.*

Proof. Project the carrier of Definition 7551.1. The analytic rows are R, W, Q, D, A, Z, E, and the structural rows are H, C, P, N.

The source and windows are read first. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular rational values, while the $\mathsf{StreamName}^{\uparrow}$ row fixes the finite source and successor windows used by the transform.

The rational arithmetic route is then forced through Q, D, A. Numerator differences, denominator differences, denominator-growth witnesses, and quotient comparisons are displayed before any Stolz-facing handoff is read.

Finally Z, E, H, C, P, N preserve and name the same route. Since no carrier coordinate stores a selected limit, quotient equality, countable-choice schedule, trichotomy witness, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 7551.3 (Regular Cauchy–Stolz squeeze handoff)**.** *Every completion-facing squeeze read exported by an accepted* $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{StolzCesaroTheorem}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In carrier coordinates, the read exposes R, W, Q, D, A before Z and exposes Z before E. Thus the consumer sees a finite regular-source and difference-ratio certificate, not a host quotient equality, selected endpoint, or classical completeness theorem.

Proof. Start with an accepted carrier $S = (R, W, Q, D, A, Z, E, H, C, P, N)$. The finite source values and observation windows are exactly the rows R, W.

The arithmetic comparison is displayed by Q, D, A. These rows supply the rational differences, denominator-growth evidence, and quotient comparison windows needed for a Stolz–Cesaro squeeze read.

Only then can the handoff row Z be consumed. The terminal row E is reached through the $\mathsf{Real}^{\uparrow}$ seal after the Stolz route is visible, while H, C, P, N preserve transport, replay, provenance, and local naming. The refused host principles are absent from the carrier coordinates. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyStolzTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyStolzTransform}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite windows, Rat arithmetic rows, denominator-growth data, finite difference-ratio comparisons, a StolzCesaroTheorem handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchyStolzTransform}^{\uparrow}}$, seed)

Scope closed: Definition 7551.1, Definition 7551.2, and Definition 7551.3 seed the finite Cauchy–Stolz transform route over regular rational source windows, rational difference rows, Stolz handoff, real sealing, transport, replay, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyStolzTransform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host quotient equality, selected limits, trichotomy, countable choice, classical completeness, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, rational difference-ratio exactness, denominator-growth compatibility, Stolz handoff ordering, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyStolzTransformUp.

Chapter 7552

A Concrete Naming Certificate for $\mathsf{BusemannProjection}^{\uparrow}$

The $\mathsf{BusemannProjection}^{\uparrow}$ packet realizes the hyperbolic observer-expansion route of chapter 9463 as a finite boundary projection surface. It sits after the Busemann-boundary rows of chapter 3732, the visual-boundary rows of chapter 4181, the observer-expansion rows of chapter 4167, the hyperbolic metric rows of chapter 3474, and the disk-metric rows of chapter 6209. The packet records a finite projection from observer windows to Busemann boundary coordinates; it is not a completed ideal boundary, a quotient ray space, a selected geodesic representative, or a host compactification theorem.

Definition 7552.1 (Busemann projection carrier)**.** *A* $\mathsf{BusemannProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, D, \Xi, V, O, W, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row for the inspected disk or boundary-facing source.*
2. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric row supplying the local disk coordinate and distance comparison windows.*
3. *Ξ is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary row that receives the projected direction.*
4. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *visual-boundary row used to compare the projected direction with observer-visible boundary data.*
5. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *observer-window row that supplies finite expansion and phase-readout windows.*
6. *W is the projection-window row, listing the finite metric, boundary, visual, and observer windows consumed by the projection.*
7. *R is the refusal row excluding host compactifications, quotient ray spaces, selected geodesic representatives, and unlogged global boundary models.*
8. *H records componentwise* hsame *transport for the metric, disk, Busemann, visual, observer, projection-window, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through $M, D, \Xi, V, O, W, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BusemannProjection}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BusemannProjection}^{\uparrow}}$ *row.*

The later Lean TasteGate file for $\mathsf{BusemannProjection}^{\uparrow}$ *must provide Nontrivial and FieldFaithful instances before the packet can be treated as a positive discovery gate rather than a finite carrier row.*

Theorem 7552.2 (Busemann projection NameCert obligations)**.** *For every accepted* $\mathsf{BusemannProjection}^{\uparrow}$ *carrier* $B = (M, D, \Xi, V, O, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BusemannProjection}^{\uparrow}}$ *surface consists exactly of metric admission, disk-metric admission, Busemann-boundary admission, visual-boundary comparison, observer-window admission, projection-window exactness, refusal exactness, componentwise transport, displayed replay, provenance, and local naming. Every projection read must expose the finite boundary, Busemann, visual-metric, observer-window, provenance, and NameCert rows before it is consumed as a boundary projection.*

Proof. Project the carrier of Definition 7552.1. The metric and disk rows are M, D, the boundary-facing rows are Ξ, V, O, W, R, and the structural rows are H, C, P, N.

The projection begins with displayed metric data. The hyperbolic metric and disk-metric rows provide the distance windows and disk coordinate comparisons used by the boundary-facing read.

The Busemann, visual, and observer rows are then read together. The boundary target Ξ, the visual comparison V, and the observer-expansion window O determine the finite projection route, while W records which windows are consumed.

Finally the refusal and structural rows prevent escape to host geometry. Since R, H, C, P, N contain no completed ideal boundary, quotient ray space, selected geodesic representative, or global compactification theorem, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{BusemannProjection}^{\uparrow}$ read exposes HyperbolicMetric, PoincareDiskMetric, HyperbolicBusemannBoundary, HyperbolicVisualBoundary, HyperbolicObserverExpansion, projection-window, refusal, transport, replay, provenance, and local NameCert rows before any boundary projection is consumed.

Independence witness: hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-observer-expansion, hyperbolic-metric, poincare-disk-metric: $\mathsf{BusemannProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric admission, disk-metric comparison, Busemann boundary target, visual-boundary comparison, observer-window projection, host-geometry refusal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BusemannProjection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BusemannProjection}^{\uparrow}$ packet is generated from HyperbolicMetric rows, PoincareDiskMetric rows, HyperbolicBusemannBoundary rows, HyperbolicVisualBoundary rows, HyperbolicObserverExpansion windows, finite projection-window data, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BusemannProjection}^{\uparrow}}$, seed)

Scope closed: Definition 7552.1 and Definition 7552.2 seed the finite boundary projection route over hyperbolic metric, disk metric, Busemann boundary, visual boundary, observer-window, refusal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BusemannProjection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed ideal boundaries, quotient ray spaces, selected geodesic representatives, host compactification theorems, or global hyperbolic models.

Upgrade path: Obligation closure requires checked metric admission, disk-metric admission, Busemann-boundary admission, visual-boundary comparison, observer-window admission, projection-window exactness, refusal exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BusemannProjectionUp.

Chapter 7553

A Concrete Naming Certificate for CompactModulusMonotoneConvergence$^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the CompactModulusMonotoneConvergence$^{\uparrow}$ packet records the finite-modulus surface by which a monotone sequence of continuous maps on a compact metric source is read as a uniform convergence candidate. It sits over the compact metric carrier of chapter 129, the continuous-map rows of chapter 130, the uniform-limit consumer of chapter 463, the uniform-modulus route of chapter 369, the stream naming surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a displayed finite-modulus convergence interface only; it does not import open-cover compactness, quotient function equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ completeness.

Falsifiable prediction: Every accepted CompactModulusMonotoneConvergence$^{\uparrow}$ read exposes compact source probes, continuous-map rows, a monotone comparison ledger, a finite modulus row, StreamName windows, RegSeqRat readbacks, and the Real seal before any uniform-limit consumer may inspect the convergence handoff.

Independence witness: compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat: the compact metric source lacks monotone comparison and finite-modulus rows; the continuous-map route lacks a convergence handoff; the uniform-limit consumer lacks the compact source provenance; the uniform-modulus packet lacks monotone sequence data; and the regular rational readback lacks the NameCert rows that bind the whole interface. None is bijective to CompactModulusMonotoneConvergence$^{\uparrow}$.

Definition 7553.1 (Compact modulus monotone convergence carrier)**.** *A* CompactModulusMonotoneConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, M, U, L, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing the compact metric probes and finite inspection windows used by the convergence read.*
2. *F is the* ContinuousMap$^{\uparrow}$ *family row, carrying the displayed continuous-map data for the monotone sequence.*
3. *M is the monotone comparison ledger, recording the finite order checks between adjacent displayed maps at inspected source probes.*

4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, supplying the finite tolerance-to-window modulus consumed by the convergence handoff.*
5. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *row receiving only the convergence candidate produced by K, F, M, U, S, Q, R.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for sequence indices, source probes, and finite comparison reads.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerances and Real-facing estimates.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for terminal scalar comparisons already exposed by Q.*
9. *H records componentwise* hsame *transport for compact source, map family, monotone ledger, modulus, limit handoff, stream windows, regular readbacks, Real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a convergence request through $K, F, M, U, L, S, Q, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, M, U, L, S, Q, R, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, quotient equality of functions, selected limit objects, countable-choice schedules, host supremum principles, ambient Real completeness, Lean verification, or bridge checking.

Theorem 7553.2 (Compact modulus monotone convergence NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}$ *carrier*

$$G = (K, F, M, U, L, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact-source admission through K;*
2. *continuous-family admission through F over the same source row;*
3. *monotone comparison exactness through M, with each displayed adjacent comparison attached to the same finite source and stream windows;*
4. *finite-modulus exactness through U, requiring the tolerance-to-window ledger to be read before the convergence handoff;*
5. *uniform-limit handoff through L, downstream of K, F, M, U, S, Q, R and not an independent selected limit;*
6. *stream, regular readback, and Real-seal ordering through S, Q, R; and*
7. *componentwise transport, replay, provenance, local naming, and nonescape through H, C, P, N.*

No compact modulus monotone convergence obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7553.1. The compact source, continuous family, monotone ledger, finite modulus, uniform-limit handoff, stream windows, regular readbacks, Real seal, and structural rows are all displayed coordinates of the packet.

The compact and continuous-map obligations are read first through K and F. Because the map-family row is attached to the same compact source row, a consumer cannot replace the finite source probes by an ambient open-cover compactness theorem or a quotient function-space equality.

The monotone and modulus obligations are read through M and U. The comparison ledger keeps adjacent-map order checks visible at inspected source probes, while the modulus row supplies only a finite tolerance-to-window route.

Finally L, S, Q, R, H, C, P, N impose output order and local naming. The uniform-limit row receives the candidate after the displayed rows have been exposed, stream and rational readbacks precede the Real seal, and the structural rows contain none of the refused selected-limit, choice, quotient, completeness, verification, or bridge data. □

Theorem 7553.3 (Compact modulus monotone convergence handoff)**.** *Every uniform-limit consumer of an accepted* $\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}$ *carrier must read the route*

$$K, F \longmapsto M, U \longmapsto L, S, Q, R, H, C, P, N.$$

The handoff exposes compact source probes, continuous-map rows, monotone comparisons, a finite modulus, stream windows, regular rational readbacks, and the Real seal before the convergence candidate is consumed. It exports no open-cover compactness theorem, quotient function equality, selected limit, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an accepted packet $G = (K, F, M, U, L, S, Q, R, H, C, P, N)$. The source and map-family rows K, F provide the displayed compact metric and continuous-map data.

Next read M, U. The monotone comparison ledger and finite modulus row are the only coordinates through which adjacent-map order and tolerance-to-window information enter the packet.

The convergence handoff is then read through L, with S, Q, R supplying stream windows, regular rational estimates, and the Real seal. The rows H, C, P, N transport, replay, source, and name the same finite route.

Since no coordinate stores open-cover compactness, quotient function equality, selected limits, countable choice, or ambient Real completeness, the displayed handoff is the whole public convergence interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7553.1, Definition 7553.2, and Definition 7553.3 seed the finite-modulus convergence surface over compact source probes, continuous-map rows, monotone comparisons, uniform modulus, uniform-limit handoff, stream windows, regular readbacks, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient function equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, continuous-family admission, monotone comparison exactness, finite-modulus exactness, uniform-limit handoff ordering, readback ordering, and nonescape proofs for BEDC.Derived.CompactModulusMonotoneConvergenceUp.

Constructive story: A $\mathsf{CompactModulusMonotoneConvergence}^{\uparrow}$ packet is generated from CompactMetric, ContinuousMap, monotone comparison, UniformModulus, UniformLimit, StreamName, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows.

Chapter 7554

A Concrete Naming Certificate for $\mathsf{CauchyCompletionCategory}^{\uparrow}$

Following Mac Lane and Bishop–Bridges on Kleisli composition and constructive completion, the $\mathsf{CauchyCompletionCategory}^{\uparrow}$ packet records the finite category surface whose objects are displayed completion packets and whose morphisms are completion-valued BEDC routes. It sits after the completion monad of chapter 378, the unit row of chapter 1527, the bind row of chapter 3777, the metric-completion boundary of chapter 1506, and the finite real-name route through chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names a completion-category interface as BEDC rows; it is not quotient completion equality, not a selected limit principle, not host equality of completed objects, and not countable choice.

Definition 7554.1 (CauchyCompletionCategory carrier)**.** *A* $\mathsf{CauchyCompletionCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (O, A, U, B, T, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the completion-object row. Each object is read through* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *A is the completion-morphism row. Each morphism is a completion-valued BEDC route whose source and target are displayed by O.*
3. *U is the identity row, routing object identities through* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *and the same source-completion data.*
4. *B is the bind and Kleisli-composition row, exposing composition only through* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *and* $\mathsf{CauchyCompletionBind}^{\uparrow}$.
5. *T is the classifier-transport row, recording componentwise transport for object, morphism, unit, bind, metric-completion, stream, regular-rational, dyadic, and Real-seal reads.*
6. *R is the replay row, forcing each category-facing consumer to replay the displayed completion route before reading identity or composition data.*
7. *H records componentwise* hsame *transport for completion objects, completion morphisms, identities, compositions, classifier transport, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for object reads, morphism reads, identity reads, Kleisli-composition reads, and terminal* $\mathsf{Real}^{\uparrow}$ *seal requests.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionCategory}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCategory}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7554.2 (CauchyCompletionCategory NameCert obligations)**.** *For every accepted* CauchyCompletionCate *carrier $G = (O, A, U, B, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCategory}^{\uparrow}}$ *surface consists exactly of carrier admission, object exposure, morphism exposure, identity exposure, composition exposure, classifier transport, replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient completion equality, selected limits, host equality, countable choice, hidden representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7554.1. The object, morphism, identity, composition, transport, replay, structural transport, displayed replay, provenance, and naming rows are precisely $O, A, U, B, T, R, H, C, P, N$.

The object and morphism clauses come from O and A. The identity clause is read through U, while the composition clause is read through B and therefore through the displayed monad and bind rows rather than through host equality of functions.

Classifier transport and replay are local to T and R. The rows H and C preserve and replay the same carrier coordinates componentwise, and P, N source and name the packet.

Since no carrier coordinate stores quotient completion equality, selected limits, host equality, countable choice, hidden representatives, or ambient completeness, the non-escape obligation follows by exhaustion. □

Theorem 7554.3 (CauchyCompletionCategory Kleisli factorization)**.** *Every morphism read and every composition read of an accepted* $\mathsf{CauchyCompletionCategory}^{\uparrow}$ *packet factors through*

$\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Re

The factorization may inspect $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows along the regular-rational route, but it exports no quotient completion equality, selected limits, host equality of completed objects, countable choice, hidden stream representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a morphism or composition request. By Definition 7554.2, the request first projects the accepted carrier and reads only the displayed object and morphism rows.

For identities, the row U routes the request through $\mathsf{CauchyCompletionUnit}^{\uparrow}$. For composites, the row B routes the request through $\mathsf{CauchyCompletionMonad}^{\uparrow}$ and $\mathsf{CauchyCompletionBind}^{\uparrow}$, so categorical composition is a finite Kleisli-composition read.

The completion-facing part then passes through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with $\mathsf{DyadicRatCore}^{\uparrow}$ supplying the tolerance observations used by the regular-rational readback. These rows are named in the provenance row P and replayed by R and C.

The transport rows T and H preserve the same route componentwise. Because the carrier has no row for quotient completion equality, selected limits, host equality, countable choice, hidden representatives, or ambient completeness, no second route can supply categorical composition data. □

Falsifiable prediction: A $\mathsf{CauchyCompletionCategory}^{\uparrow}$ consumer that reads composition without first exposing the monad, unit, bind, metric-completion, stream-window, regular-rational, Real-seal, transport, replay, provenance, and local NameCert rows refutes the packet.

Independence witness: cauchycompletionmonad, cauchycompletionunit, cauchycompletionbind, metriccompletion, streamname, regseqrat, real: none is bijective to $\mathsf{CauchyCompletionCategory}^{\uparrow}$ because the category packet jointly carries completion objects, completion morphisms, identity rows, bind-composition rows, classifier transport, replay, provenance, and local naming in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionCategory}^{\uparrow}$ packet is generated from completion-object rows, completion-morphism rows, identity rows, bind-composition rows, classifier-transport rows, replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7554.1, Definition 7554.2, and Definition 7554.3 seed the finite completion-category packet for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, hidden stream representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, object exposure, morphism exposure, identity exposure, composition exposure, classifier transport, replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyCompletionCategoryUp.

Chapter 7555

A Concrete Naming Certificate for $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$ packet records the finite modulus data needed to read a nested interval construction as Real-facing and Completion-facing evidence. It sits after the dyadic interval rows of chapter 7491, the subdivision ledger of chapter 7480, the regular rational route of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a finite nested-interval schedule and does not claim open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, a selected limit, countable choice, trichotomy, or host equality of real numbers.

Definition 7555.1 (Bishop interval nesting modulus carrier)**.** *A* $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (D, L, U, M, S, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger for the displayed nested intervals.*
2. *L is the lower-endpoint row, read as a finite dyadic list over D.*
3. *U is the upper-endpoint row, read as a finite dyadic list over D.*
4. *M is the modulus row assigning each requested precision a finite depth after which the interval width is inside the requested dyadic tolerance.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule for the interval endpoints and widths.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed on the windows named by S.*
7. *H records componentwise* hsame *transport for D, L, U, M, S, R and the structural rows.*
8. *C records displayed* Cont *replay of the same endpoint, width, modulus, stream, and regular-rational reads.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingModulus}^{\uparrow}}$ *row binding the displayed interval-nesting surface.*

The carrier has no coordinate for open-cover compactness, ambient real completeness, selected limits, countable choice, trichotomy, or host real equality.

Theorem 7555.2 (BishopIntervalNestingModulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$ *carrier* $J = (D, L, U, M, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingModulus}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic endpoint ledger, lower endpoint row, upper endpoint row, modulus row, stream schedule, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *nestedness admission, requiring each displayed interval row to read the same dyadic ledger* D *and to place the next lower and upper endpoints inside the previous endpoints;*
3. *width-modulus admission, requiring every request in* M *to consume a positive tolerance row already displayed in* D*;*
4. *finite-window admission, requiring the stream schedule* S *to expose the endpoint and width windows before the regular rational row* R *is read;*
5. *Real-facing replay stability, requiring* H *and* C *to preserve the same endpoint, modulus, stream, and readback rows componentwise;*
6. *provenance and local naming through* P *and* N*;*
7. *nonescape from open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, countable choice, trichotomy, and host real equality.*

Proof. Project the carrier of Definition 7555.1. The only available rows are $D, L, U, M, S, R, H, C, P, N$.

Nestedness, width-modulus admission, and finite-window admission are row-local. The lower and upper endpoint rows are read over the same dyadic ledger D, the modulus row M consumes tolerances from D, and the stream row S exposes the finite windows before R is available.

Transport, replay, provenance, and naming are exactly the structural rows H, C, P, N. Since the carrier has no coordinate for open-cover compactness, selected limits, countable choice, trichotomy, host equality, or ambient completeness, the listed obligations exhaust the local certificate surface. □

Theorem 7555.3 (BishopIntervalNestingModulus Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$ *packet must first read the finite route*

$$D,\ L,\ U,\ M,\ S,\ R$$

and then the structural rows H, C, P, N*. The handoff exposes a nested dyadic interval schedule together with a width modulus and regular-rational readback, but it does not expose open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit, countable choice, trichotomy, or host real equality.*

Proof. Start from an accepted carrier $J = (D, L, U, M, S, R, H, C, P, N)$. The endpoint and width information is displayed only through the dyadic ledger D and the lower and upper endpoint rows L, U.

Apply Definition 7555.2. The width-modulus row M and stream row S must be read before the regular rational row R, so the Real-facing consumer receives the nested interval data as a finite schedule.

Finally use the structural rows H, C, P, N. They transport, replay, source, and locally name the same route, leaving no coordinate through which a consumer could import open-cover compactness, ambient completeness, a selected limit, countable choice, trichotomy, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNestingModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalNestingModulus}^{\uparrow}$ packet is generated from a dyadic endpoint ledger, lower and upper endpoint rows, a width modulus, a finite stream schedule, a regular rational readback, componentwise transport, displayed replay, provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestingModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7555.1, Definition 7555.2, and Definition 7555.3 name the seed carrier, finite obligation surface, and Real-facing nested-interval handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestingModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, ambient Real completeness, selected limits, countable choice, trichotomy, host real equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, nestedness admission, width-modulus admission, finite-window admission, Real-facing replay stability, provenance, local naming, and nonescape for BEDC.Derived.BishopIntervalNestingModulu

Chapter 7556

A Concrete Naming Certificate for $\mathsf{ChoiceFreeRealSeal}^{\uparrow}$

Following Bishop and Bridges on constructive real names, the $\mathsf{ChoiceFreeRealSeal}^{\uparrow}$ packet records the terminal finite seal for a real-name route that stays inside displayed name data. It reads stream windows from chapter 126, regular rational readback from chapter 301, dyadic tolerances from chapter 291, and the terminal real-name surface of chapter 38. The packet is a finite NameCert seal for accepted real-name consumers, not a completeness theorem for $\mathsf{Real}^{\uparrow}$ and not a quotient stream presentation.

Definition 7556.1 (Choice-free real seal carrier)**.** *A* $\mathsf{ChoiceFreeRealSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window used by the consumer read.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to that window.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the inspected precision.*
4. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after* W, R, D *have been read.*
5. L *is the nonescape ledger refusing countable-choice, quotient-stream, trichotomy, host-real-equality, and completeness routes.*
6. H *records componentwise* hsame *transport for* W, R, D, E, L, P, N*.*
7. C *records the displayed* Cont *replay from stream window to rational readback, dyadic tolerance, and terminal real seal.*
8. P *is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeRealSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a countable-choice selector, quotient stream equality, trichotomy on real names, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7556.2 (Choice-free real seal NameCert obligations)**.** *For the carrier of Definition 7556.1, the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeRealSeal}^{\uparrow}}$ *surface consists of five finite rows: carrier admission for* S*;*

stream-window, regular-rational, and dyadic-tolerance visibility before the terminal real seal; componentwise hsame *transport for the displayed rows;* Cont *replay and* Pkg *provenance for the same route; and nonescape from countable-choice, quotient-stream, trichotomy, host-real-equality, and completeness routes.*

Proof. Carrier admission is the displayed packet $S = (W, R, D, E, L, H, C, P, N)$. The first read exposes the stream window, regular rational readback, and dyadic tolerance rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be used.

The transport, replay, and provenance rows are H, C, P. They keep every accepted read on the same finite packet and attach the local naming row N to that route.

The ledger row L supplies the refusal boundary. Since the carrier lists no selector, quotient equality, trichotomy, host equality, or completeness coordinate, those principles cannot be exported by the local NameCert surface. □

Lean4 (definition) `BEDC.Derived.ChoiceFreeRealSealUp`

Theorem 7556.3 (Choice-free real seal no-choice route)**.** *An accepted* $\mathsf{ChoiceFreeRealSeal}^{\uparrow}$ *packet exports no countable-choice selector, no quotient stream equality, no trichotomy row for real names, no host real equality, and no* $\mathsf{Real}^{\uparrow}$ *completeness row. Every real-facing consumer must read the finite stream window W, regular rational row R, dyadic tolerance row D, terminal real seal E, nonescape ledger L, transport H, replay C, provenance P, and local name N as its complete surface.*

Proof. Inspect the carrier rows in Definition 7556.1. The only real-facing rows are W, R, D, E together with the ledger, transport, replay, provenance, and naming rows.

Apply Definition 7556.2. It forces every local naming read to keep the same five-row obligation surface and the same nonescape ledger.

The excluded principles have no row in S. A consumer asking for countable choice, quotient stream equality, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness therefore leaves the accepted packet rather than reading an exported coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeRealSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceFreeRealSeal}^{\uparrow}$ packet is generated from a finite StreamNameUp window, RegSeqRatUp readback, DyadicRatCoreUp tolerance, terminal RealUp seal, nonescape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoiceFreeRealSeal}^{\uparrow}}$, seed)

Scope closed: Definition 7556.1, Definition 7556.2, and Definition 7556.3 seed the finite choice-free real-name seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoiceFreeRealSeal}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.ChoiceFreeRealSealUp`

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient stream equality, trichotomy, host real equality, Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and nonescape exactness for the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert route.

Chapter 7557

A Concrete Naming Certificate for $\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}$

Following Bishop-style finite-cover proofs of uniform continuity on compact metric data, the $\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}$ packet records the finite envelope needed by constructive compact-continuous consumers. It reads compact metric finite-net data from chapter 129, total-bounded schedules from chapter 127, continuous-map pointwise rows from chapter 130, and formal-ball tolerances from chapter 1510. The envelope is a displayed finite cover and modulus route, not an open-cover compactness theorem and not a host continuity principle.

Definition 7557.1 (Finite-cover uniform modulus envelope carrier)**.** *A* $\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, T, F, B, M, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite compact source row.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite schedule row selecting the inspected cover cells.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise continuity row read at the selected cells.*
4. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *tolerance and containment row for the cover cells.*
5. *M is the displayed uniform modulus envelope extracted from K, T, F, B.*
6. *W is the finite window ledger tying the cover cells, pointwise moduli, and output modulus to one read.*
7. *H records componentwise* hsame *transport for the displayed rows.*
8. *C records the* Cont *replay from compact source to finite cover, pointwise continuity, formal-ball tolerance, and modulus envelope.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, countable choice, selected subsequences, host semantic continuity, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7557.2 (Finite-cover uniform modulus envelope NameCert obligations)**.** *For the carrier of Definition 7557.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}}$ *surface consists of carrier admission for* U*; finite cover visibility through* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *rows; pointwise continuity and formal-ball tolerance visibility through* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{FormalBall}^{\uparrow}$ *rows; componentwise transport and replay stability through* H, C, P*; and nonescape from open-cover compactness, countable choice, selected-subsequence, host-continuity, and completeness routes.*

Proof. Carrier admission exposes the finite packet $U = (K, T, F, B, M, W, H, C, P, N)$. The compact and total-bounded rows supply only displayed finite cover data, and the continuous-map and formal-ball rows supply only pointwise and tolerance reads.

The modulus envelope M is read through the window ledger W and replay row C. Transport in H and provenance in P keep that read attached to the same finite packet and local name.

No excluded principle has a carrier coordinate. Thus open-cover compactness, countable choice, selected subsequences, host semantic continuity, and $\mathsf{Real}^{\uparrow}$ completeness are outside the exported NameCert surface. □

Lean4 (definition) `BEDC.Derived.FiniteCoverUniformModulusEnvelopeUp`

Theorem 7557.3 (Finite-cover uniform modulus consumer route)**.** *Every constructive compact-continuous consumer of an accepted* $\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}$ *packet must read the compact source row* K*, total-bounded schedule* T*, continuous-map row* F*, formal-ball tolerance* B*, modulus envelope* M*, finite window ledger* W*, transport* H*, replay* C*, provenance* P*, and local name* N *before treating the packet as a uniform-modulus source.*

Proof. Read the carrier rows of Definition 7557.1. The compact source, total-bounded schedule, continuous-map row, and formal-ball tolerance are the only inputs to the displayed envelope.

Apply Definition 7557.2. It binds those rows to M, W, H, C, P, N as one local NameCert route.

Therefore a consumer can use the packet as a uniform-modulus source only by following the displayed route. Any appeal to open-cover compactness, countable choice, selected subsequences, host semantic continuity, or completeness has no exported row. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}$ packet is generated from CompactMetricUp finite cover data, TotallyBoundedUp schedules, ContinuousMapUp pointwise rows, FormalBallUp tolerances, a displayed modulus envelope, finite window ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 7557.1, Definition 7557.2, and Definition 7557.3 seed the finite cover uniform modulus envelope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCoverUniformModulusEnvelope}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.FiniteCoverUniformModulusEnvelopeUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected subsequences, host semantic continuity, Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and consumer-route exactness for the displayed $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert route.

Chapter 7558

A Concrete Naming Certificate for $\mathsf{DarbouxStieltjesSum}^{\uparrow}$

Following Riemann, Stieltjes, and Bishop–Bridges, the $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ packet records the finite upper-lower Stieltjes sum boundary for a regulated integrand read against a bounded-variation integrator. It sits beside the Riemann–Stieltjes packet of chapter 1637, the bounded-variation ledger of chapter 398, the regulated-function surface of chapter 1906, the tagged finite-sum row of chapter 1615, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite partition cells, local oscillation rows, bounded-variation cell contributions, dyadic comparison budgets, rational readback, and the Real-facing seal handoff; it does not import measure integration, quotient functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7558.1 (Darboux–Stieltjes sum carrier)**.** *A* $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, A, T, U, L, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *integrand row, exposing only the finite value windows and one-sided oscillation reads requested by the displayed partition.*
2. *A is the* $\mathsf{BoundedVariation}^{\uparrow}$ *integrator row, carrying the finite partition-variation ledger and transported endpoint changes of the integrator.*
3. *T is the tagged partition row, read through the* $\mathsf{RiemannSum}^{\uparrow}$ *mesh and tag-containment surface.*
4. *U is the upper Darboux–Stieltjes cell row, recording for each carried cell a displayed upper oscillation bound for F multiplied by the corresponding positive variation contribution of A.*
5. *L is the lower Darboux–Stieltjes cell row, recording for each carried cell a displayed lower oscillation bound for F multiplied by the corresponding signed variation contribution of A when that row is licensed by A.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *oscillation budget, fixing the dyadic tolerances used to compare upper and lower cell ledgers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite upper sum, lower sum, and gap comparison ledgers.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after F, A, T, U, L, Q, R have all been displayed.*

9. *H records componentwise* hsame *transport for the integrand, integrator, tagged partition, upper and lower cell rows, dyadic budget, rational readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a partition request through the regulated-function windows, bounded-variation cell contributions, upper and lower cell ledgers, dyadic budget, rational readback, and Real seal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DarbouxStieltjesSum}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxStieltjesSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $F, A, T, U, L, Q, R, E, H, C, P, N$. It has no coordinate for measure integration, a quotient of functions, a selected refinement limit, countable choice, or an ambient Real-completeness theorem.

Theorem 7558.2 (Darboux–Stieltjes NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ *carrier $D = (F, A, T, U, L, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxStieltjesSum}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated integrand, bounded-variation integrator, tagged partition, upper and lower Darboux cell rows, dyadic oscillation budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every accepted read follows the finite route*

$$F, A, T \longrightarrow U, L \longrightarrow Q \longrightarrow R \longrightarrow E$$

together with H, C, P, N.

Proof. Project the carrier of Definition 7558.1. The displayed rows list the regulated integrand windows, bounded-variation cell contributions, tagged partition, upper and lower cell ledgers, dyadic budget, rational readback, Real seal, and structural provenance.

For each carried cell, the upper and lower rows are formed from the same partition row T. The integrand contribution is read through F, while the Stieltjes cell contribution is read through A, so both Darboux rows are finite cellwise records rather than host-level extrema or a measure integral.

The dyadic budget Q and readback row R compare the two finite ledgers before the seal row E is visible. Transport through H and replay through C preserve the same route, and P, N only source and name it.

Since no carrier coordinate names measure integration, quotient functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligation surface. □

Theorem 7558.3 (Darboux–Stieltjes RegSeqRat handoff)**.** *Every* $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ *read that is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *route factors through a finite partition refinement before the numerical seal is inspected. For an accepted carrier $D = (F, A, T, U, L, Q, R, E, H, C, P, N)$, the consumer must first read the common refinement of the tagged partition, the regulated integrand windows, the bounded-variation cell contributions, the upper and lower Darboux cell rows, and the dyadic oscillation budget; only then may it read R and the terminal seal row E.*

Proof. Start with the finite tagged partition row T and refine it inside the displayed partition surface. The refinement changes only carried cells and their tag-containment data, so it remains a finite BHist calculation.

Read F and A on that refinement. The regulated integrand supplies the local value windows and oscillation bounds, while the bounded-variation integrator supplies the endpoint changes and variation ledger needed by each Stieltjes cell.

Rebuild the upper and lower ledgers U, L on the refined cells and compare them through the dyadic budget Q. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads back exactly those finite sums and their gap, after which the $\mathsf{Real}^{\uparrow}$ seal E may be consumed.

The structural rows H, C, P, N transport, replay, source, and name the same refinement route. Therefore the handoff never uses measure integration, quotient functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxStieltjesSum}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ packet is generated from a regulated integrand row, a bounded-variation integrator row, a tagged partition, upper and lower Darboux–Stieltjes cell ledgers, a dyadic oscillation budget, a regular rational readback row, a Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxStieltjesSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7558.1, Definition 7558.2, and Definition 7558.3 specify the finite Darboux–Stieltjes sum carrier, its NameCert obligation rows, and its RegSeqRat-to-Real handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxStieltjesSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure integration, quotient functions, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: The next grade requires Lean encoding of the carrier and checked theorems for finite partition refinement, upper and lower cell comparison, dyadic oscillation budgeting, RegSeqRat readback, and Real seal nonescape.

Chapter 7559

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$

Following Mac Lane's monad multiplication and Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ packet records the finite join surface that flattens a completion-of-completion name into the displayed Cauchy-completion boundary. It sits beside the law ledger of chapter 3834, the multiplication packet of chapter 1829, the unit route of chapter 1527, the bind route of chapter 3777, the idempotence packet of chapter 2381, and the finite readback layers chapter 126, chapter 291, chapter 301, and chapter 38. The chapter names join as a local BEDC carrier for constructive completion; it is not quotient completion equality, not a selected limit, not host equality of completed objects, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7559.1 (Cauchy completion monad join carrier)**.** *A* $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (O, U, B, I, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the completion-of-completion input row. It displays the outer completion name before any flattening data are public.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row, recording how source and already completed inputs enter the join surface.*
3. *B is the* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *row, recording the finite bind schedule used to flatten the outer name.*
4. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *row, requiring an already completed inner name to reuse the displayed completion boundary rather than produce a second quotient object.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed after O, U, B, I have been read.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same flattened windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the flattened name.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after the stream, tolerance, and regular-rational rows have been fixed.*
9. *H records componentwise* hsame *transport for outer completion, unit, bind, idempotence, stream, dyadic, regular-rational, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the outer completion request through the join route and then through S, D, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, CauchyCompletionM $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadJoin}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve displayed rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, hidden stream witnesses, or an ambient completeness theorem.

Theorem 7559.2 (Cauchy completion monad join NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ *carrier $J = (O, U, B, I, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMona}}$ *surface consists exactly of carrier admission, outer-completion exposure, unit exposure, bind-schedule exposure, idempotence reuse, flattened StreamName window exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, terminal Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape of quotient completion equality, selected representatives, countable choice, host equality, hidden stream witnesses, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7559.1. The rows $O, U, B, I, S, D, R, E, H, C, P, N$ are the whole public surface, so carrier admission and row exposure are exactly these projections.

The join-specific obligations come from O, U, B, I. The outer completion row must be displayed first, the unit row records admission into the completion surface, the bind row records the finite flattening schedule, and the idempotence row forces already completed inner data back to the same displayed boundary.

The readback obligations come from S, D, R, E. These rows are downstream of the join route, so a consumer cannot inspect a stream window, dyadic tolerance, regular-rational readback, or Real seal detached from O, U, B, I.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, hidden stream witnesses, or ambient completeness, the nonescape clause follows by exhaustion. □

Theorem 7559.3 (Cauchy completion monad join flattening route)**.** *Every completion-of-completion consumer of an accepted* $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ *packet must read*

$$O \longmapsto \mathsf{CauchyCompletionUnit}^{\uparrow} \longmapsto \mathsf{CauchyCompletionBind}^{\uparrow} \longmapsto \mathsf{CompletionIdempotence}^{\uparrow}$$

before it can consume the finite readback route

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route is the local join/multiplication face consumed by $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *and by the existing* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *packet; it exports no quotient completion, selected representative, countable-choice schedule, host equality, hidden stream witness, or ambient completeness principle.*

Proof. Begin with an accepted carrier $J = (O, U, B, I, S, D, R, E, H, C, P, N)$. Projection exposes the outer completion row O before any readback row is available.

The row U supplies the completion-unit admission used by constant or already displayed inputs. The row B supplies the bind schedule that flattens the outer completion. The row I records that already completed inner data are reused through the same completion boundary.

Only after those rows have been read does the packet expose S, D, R, E. Componentwise transport through H and replay through C preserve this order for every transported packet.

The provenance row P ties the route to the monad-law and multiplication siblings. Because no excluded object is stored in the carrier, the consumer receives only the displayed join route and the finite readback rows named in the statement. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ read exposes completion-of-completion input, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any flattened completion seal can be consumed.

Independence witness: cauchy-completion-monad-law, cauchycompletionmultiplication, cauchy-completionunit, cauchy-completion-bind, completion-idempotence, streamname, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ because this packet jointly binds the outer completion input, unit admission, bind schedule, idempotence reuse, finite readback rows, transport, replay, provenance, and local naming as one join carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadJoin}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ packet is generated from a completion-of-completion input row, CauchyCompletionUnit admission, CauchyCompletionBind scheduling, CompletionIdempotence reuse, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, terminal RealUp seal access, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadJoin}^{\uparrow}}$, seed)

Scope closed: Definition 7559.1, Definition 7559.2, and Definition 7559.3 seed the finite join packet for constructive completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadJoin}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality, hidden stream witnesses, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, outer-completion exposure, unit exposure, bind-schedule exposure, idempotence reuse, readback exactness, transport, replay, provenance, and nonescape for BEDC.Derived.CauchyCompletion

Chapter 7560

A Concrete Naming Certificate for $\mathsf{BoundaryPairing}^{\uparrow}$

The $\mathsf{BoundaryPairing}^{\uparrow}$ packet realizes the paired-boundary handoff surface of chapter 9463 as finite BEDC data. It sits after the Gromov-boundary carrier of chapter 6470, the cross-ratio packet of chapter 3948, the geodesic-flow route of chapter 3856, the Busemann-cocycle row of chapter 4134, and the Poincare-disk boundary transport route of chapter 6267. The packet records a displayed ordered pair of boundary rows before cross-ratio or geodesic-flow consumers may inspect a handoff; it is not quotient ideal-boundary equality, not a selected geodesic, not a completed hyperbolic disk, and not a host compactification theorem.

Definition 7560.1 (Boundary pairing carrier)**.** *A* $\mathsf{BoundaryPairing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, R, G, X, F, B, T, H, C, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{GromovBoundary}^{\uparrow}$ *boundary row.*
2. *R is the right* $\mathsf{GromovBoundary}^{\uparrow}$ *boundary row.*
3. *G is the pair-guard row, recording that L and R are exposed as two boundary rows before any downstream consumer reads a cross-ratio or flow handoff.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *handoff row, available only after the ordered pair L, R and the guard G have been displayed.*
5. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *handoff row, recording the boundary-paired flow route consumed by geodesic-facing readers.*
6. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *comparison row tying the same pair to Busemann-level boundary transport.*
7. *T is the* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *row, recording disk-boundary movement that preserves the displayed pair.*
8. *H records componentwise* hsame *transport for the two boundary rows, pair guard, cross-ratio handoff, geodesic-flow handoff, Busemann comparison, disk-boundary transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a paired-boundary request through $L, R, G, X, F, B, T, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundaryPairing}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlo}$ $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryPairing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, G, X, F, B, T, H, C, P, N$. It has no coordinate for quotient ideal-boundary equality, selected geodesic representatives, completed disks, unlogged cross-ratio shortcuts, physical observer laws, or host compactification data.

Theorem 7560.2 (Boundary pairing NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryPairing}^{\uparrow}$ *carrier $Q = (L, R, G, X, F, B, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryPairing}^{\uparrow}}$ *surface consists exactly of carrier admission, left-boundary exposure, right-boundary exposure, pair-guard exactness, cross-ratio handoff exposure, geodesic-flow handoff exposure, Busemann-cocycle comparison, Poincare-disk boundary-transport exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape of quotient ideal-boundary equality, selected geodesics, completed disks, unlogged cross-ratio shortcuts, physical observer laws, and host compactification data.*

Proof. Project the carrier of Definition 7560.1. The rows $L, R, G, X, F, B, T, H, C, P, N$ are the whole accepted packet, so carrier admission and the two boundary exposures are direct projections.

The guard row G supplies pair exactness. It records that both boundary rows are visible as a pair before any cross-ratio, geodesic-flow, Busemann, or disk-transport row can be consumed.

The downstream handoffs are the rows X, F, B, T. Each handoff is attached to the same displayed pair, so no consumer can replace the pair by a hidden boundary quotient or by a selected geodesic representative.

The structural rows H, C, P, N transport, replay, source, and name the same finite surface. Since the carrier has no coordinate for quotient ideal-boundary equality, selected geodesics, completed disks, unlogged shortcuts, physical observer laws, or host compactification data, the nonescape clause follows by exhaustion. □

Theorem 7560.3 (Boundary pairing cross-ratio handoff)**.** *Every cross-ratio, geodesic-flow, or Busemann-facing consumer of an accepted* $\mathsf{BoundaryPairing}^{\uparrow}$ *packet first reads the ordered boundary pair*

$$L \longmapsto R \longmapsto G$$

and only then reads the handoff route

$\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$.

The handoff exports paired boundary rows and their finite transport provenance; it exports no quotient ideal-boundary equality, selected global geodesic, completed hyperbolic disk, unlogged cross-ratio shortcut, physical observer law, or host compactification theorem.

Proof. Start with an accepted carrier $Q = (L, R, G, X, F, B, T, H, C, P, N)$. Projection exposes the two boundary rows and the pair guard before any handoff row is available.

The cross-ratio handoff X reads the pair only after G has certified that both boundary rows are visible. The geodesic-flow row F and Busemann row B consume the same pair, while T records the disk-boundary transport route that preserves it.

Componentwise transport through H and displayed replay through C keep the order fixed for transported packets. Provenance P names the sibling boundary, cross-ratio, flow, Busemann, and Poincare-disk routes, and N names the local certificate.

Because none of the excluded objects appears as a carrier coordinate, every listed consumer receives only the displayed ordered pair and its finite handoff rows. □

Falsifiable prediction: Every accepted $\mathsf{BoundaryPairing}^{\uparrow}$ read exposes two boundary rows before any cross-ratio handoff can be consumed.

Independence witness: gromov-boundary, hyperbolic-cross-ratio, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, poincare-disk-boundary-transport: none is bijective to $\mathsf{BoundaryPairing}^{\uparrow}$ because this packet alone binds two displayed $\mathsf{GromovBoundary}^{\uparrow}$ rows, a pair guard, cross-ratio handoff, geodesic-flow handoff, Busemann-cocycle comparison, disk-boundary transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

The Lean TasteGate surface for this packet must keep the two boundary rows and the pair guard as separate fields before Nontrivial and FieldFaithful instances for $\mathsf{BoundaryPairing}^{\uparrow}$ can serve downstream cross-ratio or geodesic-flow consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryPairing}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryPairing}^{\uparrow}$ packet is generated from two GromovBoundary rows, a pair-guard row, HyperbolicCrossRatio handoff, HyperbolicGeodesicFlow handoff, HyperbolicBusemannCocycle comparison, PoincareDiskBoundaryTransport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryPairing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7560.1, Definition 7560.2, and Definition 7560.3 seed the finite paired-boundary route consumed before cross-ratio and geodesic-flow handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryPairing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal-boundary equality, selected geodesics, completed disks, physical observer laws, host compactification data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked boundary exposure, pair-guard exactness, cross-ratio handoff exposure, geodesic-flow handoff exposure, Busemann comparison, disk-boundary transport, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryPairingUp.

Chapter 7561

A Concrete Naming Certificate for $\mathsf{AlexiewiczNorm}^{\uparrow}$

Following Alexiewicz and the Henstock–Kurzweil integral literature, the $\mathsf{AlexiewiczNorm}^{\uparrow}$ packet records the finite BEDC surface by which a displayed primitive-window norm is read from gauge-integral data before any real-valued bound is exported. It sits beside the Henstock–Kurzweil route of chapter 4180, the gauge-integral packet of chapter 1999, the regular rational readback of chapter 301, the dyadic endpoint ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names a norm carrier for one displayed integration window; it does not import measure completion, quotient function equality, selected suprema, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7561.1 (AlexiewiczNorm carrier)**.** *An* $\mathsf{AlexiewiczNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, G, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{HenstockKurzweil}^{\uparrow}$ *integrability row that supplies the accepted gauge-integral source.*
2. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *row recording the finite tagged-window integral route used by the primitive read.*
3. *W is the primitive-window row, carrying the displayed interval endpoints and the finite list of subinterval reads on which the norm request is evaluated.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger for those windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembling the visible rational bounds for the primitive-window values.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, G, W, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the Henstock–Kurzweil, gauge-integral, primitive-window, dyadic, readback, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from an Alexiewicz-norm request through $K, G, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{AlexiewiczNorm}^{\uparrow}$, $\mathsf{HenstockKurzweil}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexiewiczNorm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, W, D, R, E, H, C, P, N$. There is no coordinate for a measure space, quotient equality of functions, selected primitive, selected supremum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7561.2 (AlexiewiczNorm NameCert obligations)**.** *For every accepted* $\mathsf{AlexiewiczNorm}^{\uparrow}$ *carrier $A = (K, G, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexiewiczNorm}^{\uparrow}}$ *surface consists exactly of carrier admission, Henstock–Kurzweil exposure through K, gauge-integral exposure through G, primitive-window exactness through W, DyadicRatCore endpoint and tolerance exposure through D, RegSeqRat readback through R, Real seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by measure completion, quotient function equality, selected primitives, selected suprema, countable choice, host almost-everywhere equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7561.1. The displayed rows are exactly $K, G, W, D, R, E, H, C, P, N$, so carrier admission exposes only Henstock–Kurzweil, gauge-integral, primitive-window, dyadic, readback, seal, transport, replay, provenance, and naming data.

The integration side starts with K and G. The Henstock–Kurzweil row supplies the accepted integral source, while the gauge-integral row supplies the finite tagged-window route through which primitive-window values are read.

The numerical side then reads W, D, R, E in order. The primitive-window row fixes the displayed interval reads, the dyadic row supplies endpoint and tolerance data, the regular rational row records the visible bounds, and the real seal is available only after those rows have been replayed.

Finally the structural rows H, C, P, N preserve the same route. Since the excluded measure, quotient, selection, choice, host-equality, and completeness principles have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7561.3 (AlexiewiczNorm Henstock handoff)**.** *Every completion-facing or Real-facing read of an accepted* $\mathsf{AlexiewiczNorm}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{HenstockKurzweil}^{\uparrow} \longmapsto \mathsf{GaugeIntegral}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The primitive-window row W is read before the dyadic tolerance ledger and the real seal, and the route cannot bypass $K, G, W, D, R, E, H, C, P, N$.

Proof. Begin with an accepted carrier read. The carrier exposes the Henstock–Kurzweil source K and the gauge-integral row G before any norm-facing numerical read can be inspected.

Replay the continuation row C. The read moves from the integration rows through the primitive-window row W, then through the dyadic ledger D and regular rational readback R before it reaches the seal E.

The structural rows H, P, N do not add another norm object. They preserve the same finite handoff, and no excluded measure, quotient, selection, choice, host-equality, or completeness coordinate is available in the packet. □

Falsifiable prediction: Every accepted $\mathsf{AlexiewiczNorm}^{\uparrow}$ read factors through $\mathsf{HenstockKurzweil}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexiewiczNorm}^{\uparrow}}$.

Constructive story: An AlexiewiczNorm$^{\uparrow}$ packet is generated from a HenstockKurzweil row, GaugeIntegral row, primitive-window row, DyadicRatCore endpoint and tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AlexiewiczNorm}^{\uparrow}}$, seed)

Scope closed: Definition 7561.1, Definition 7561.2, and Definition 7561.3 seed the finite primitive-window norm route over HenstockKurzweilUp, GaugeIntegralUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{AlexiewiczNorm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, measure completion, quotient function equality, selected primitives, selected suprema, countable choice, host almost-everywhere equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, primitive-window exactness, HenstockKurzweil exposure, GaugeIntegral exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.AlexiewiczNormUp.

Chapter 7562

A Concrete Naming Certificate for $\mathsf{RealCauchyWindow}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealCauchyWindow}^{\uparrow}$ packet records the finite tail-window certificate by which a real-facing or completion-facing read must pass from displayed Cauchy criterion data to a terminal Real seal. It sits under the Real and Completion route of chapter 9238, refines the real-name discipline of chapter 38, and threads together the window, regular-rational, and dyadic surfaces of chapter 126, chapter 301, and chapter 291. It also specializes the finite convergence surfaces of chapter 1509 and chapter 6459. The packet is a tail-window handoff certificate; it is not Real completeness, quotient-stream equality, a selected limit, countable choice, trichotomy, or host equality.

Definition 7562.1 (Real Cauchy window carrier)**.** *A* $\mathsf{RealCauchyWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, Q, D, M, T, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, carrying the finite observation windows and the order in which tail requests are inspected.*
2. *Q is the criterion request row, naming the displayed Cauchy or Bishop-Cauchy criterion to be discharged before any Real seal is read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying requested radii, split tolerances, comparison budgets, and endpoint observations.*
4. *M is the modulus row, recording the finite index bound that answers the criterion request over the schedule S at tolerance D.*
5. *T is the finite tail-window row, listing the inspected tail segment downstream of M and upstream of the Real seal.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after S, Q, D, M, T have been displayed.*
7. *H records componentwise* hsame *transport for the schedule, criterion request, tolerance, modulus, finite tail window, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S \longmapsto Q \longmapsto D \longmapsto M \longmapsto T \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyWindow}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, M, T, R, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host equality, ambient Real completeness, Lean verification, or bridge checking.

Theorem 7562.2 (Real Cauchy window NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyWindow}^{\uparrow}$ *carrier*

$$W = (S, Q, D, M, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName schedule, criterion request, DyadicRatCore tolerance, modulus row, finite tail window, Real seal, transport, replay, provenance, and local naming;*
2. *criterion-to-tail scheduling, requiring the request Q to be read over the finite schedule S before M and T are consumed;*
3. *dyadic tolerance readback, requiring D to display the requested radii, split tolerances, comparison budgets, and endpoint observations before the modulus row answers the request;*
4. *Real seal handoff, requiring R to be downstream of S, Q, D, M, T and never upstream of the finite tail window;*
5. hsame *transport and* Cont *replay, requiring H and C to preserve exactly the displayed route;*
6. Pkg *provenance and local naming, requiring P and N to name the same finite packet and no hidden host construction;*
7. *nonescape from quotient streams, selected limits, countable choice, trichotomy, host equality, and ambient Real completeness.*

No Real Cauchy window obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7562.1. The accepted tuple displays exactly the schedule, criterion request, dyadic tolerance, modulus, finite tail window, Real seal, transport, replay, provenance, and local naming rows.

The criterion-to-tail schedule is read from S, Q, M, T. A consumer first exposes the finite observation schedule and the criterion request, then reads the modulus row, and only then inspects the tail-window row that answers the request.

Dyadic tolerance readback is the role of D. The row displays the requested radii, split tolerances, comparison budgets, and endpoint observations used by the modulus and tail-window rows, so no external real-order or trichotomy principle is introduced.

The Real seal handoff is the projection to R together with the structural rows H, C, P, N. These rows transport, replay, source, and name the same finite route, and the carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, or ambient Real completeness. □

Theorem 7562.3 (Real Cauchy window finite tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RealCauchyWindow}^{\uparrow}$ *packet must pass through the finite tail route*

$$S \longmapsto Q \longmapsto D \longmapsto M \longmapsto T \longmapsto R \longmapsto H, C, P, N$$

before reading the Real seal. The handoff exports a StreamName schedule, criterion request, DyadicRatCore tolerance readback, modulus row, finite tail window, terminal Real seal, componentwise transport, replay, provenance, and local naming; it exports no Real completeness, quotient streams, selected limits, trichotomy, countable choice, host equality, Lean verification, or bridge checking.

Proof. Begin with a Real-facing or completion-facing read of the packet. Projection from Definition 7562.1 exposes the ordered rows $S, Q, D, M, T, R, H, C, P, N$.

Apply Definition 7562.2. The criterion request is tied to the finite StreamName schedule, the dyadic tolerance row is read before the modulus answers the request, and the finite tail window is downstream of that answer.

The terminal Real seal is the row R. Since the carrier admits no other seal coordinate and the transport, replay, provenance, and naming rows preserve only the displayed route, every consumer reaches R only after the finite tail window has been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyWindow}\uparrow}$.

Constructive story: A $\mathsf{RealCauchyWindow}^{\uparrow}$ packet is generated from StreamName schedules, criterion requests, DyadicRatCore tolerance rows, modulus rows, finite tail windows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyWindow}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7562.1, Definition 7562.2, and Definition 7562.3 seed the finite route from displayed Cauchy criterion data through tail-window inspection into a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyWindow}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, trichotomy, countable choice, host equality, Lean verification, or bridge checking.

Upgrade path: Formalization requires carrier admission, criterion-to-tail scheduling, dyadic tolerance readback, Real seal handoff, transport and replay, provenance, naming, and non-escape proofs for BEDC.Derived.RealCauchyWindowUp.

Chapter 7563

A Concrete Naming Certificate for $\mathsf{CauchyConvergenceRate}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyConvergenceRate}^{\uparrow}$ packet records a tolerance-to-index rate ledger for regular Cauchy names. It sits between the finite stream surface of chapter 126, the dyadic tolerance carrier of chapter 291, the regular-rational readback surface of chapter 301, and the terminal Real seal of chapter 38. The chapter names the finite rate object used by Real-and-Completion consumers; it is not a chosen limit, not a quotient of Cauchy names, and not an ambient completeness theorem.

Definition 7563.1 (Cauchy convergence rate carrier)**.** *A* $\mathsf{CauchyConvergenceRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, I, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the tolerance request row. It displays the dyadic error level to be met before any index is exposed.*
2. *I is the unary index row assigned to the tolerance in E.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular Cauchy name at the displayed index.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound row comparing the requested tolerance with the finite tail bound.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after E, I, R, D have been fixed.*
6. *H records componentwise* hsame *transport for tolerance, index, RegSeqRat readback, dyadic bound, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through the unary index, regular-rational readback, dyadic bound, and Real seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyConvergenceRate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for a selected Cauchy limit, quotient equality of names, countable choice, hidden tail witnesses, host equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7563.2 (Cauchy convergence rate NameCert obligations)**.** *For every accepted* $\mathsf{CauchyConvergenceRate}^{\uparrow}$ *carrier* $Q = (E, I, R, D, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceRate}^{\uparrow}}$ *surface consists exactly of carrier admission, tolerance request exposure, total unary-index assignment for displayed tolerances, monotone tolerance refinement, RegSeqRat handoff,* $\mathsf{Real}^{\uparrow}$ *nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7563.1. Carrier admission and tolerance exposure are the projections of Q and E.

The rate obligations are carried by I, D. Totality means that every displayed tolerance row has a displayed unary index row, and monotone refinement means that a sharper displayed tolerance is routed through the same finite ledger before its index can be consumed.

The readback and seal obligations are carried by R, S. The row R is the regular-rational handoff, while S is downstream of E, I, R, D and therefore cannot export a Real-facing use without the rate rows.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since the carrier has no coordinate for selected limits, quotient equality, countable choice, hidden tail witnesses, host equality, or ambient completeness, the nonescape clause follows by exhaustion. □

Theorem 7563.3 (Cauchy convergence rate RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing use of an accepted* $\mathsf{CauchyConvergenceRate}^{\uparrow}$ *packet first reads the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows that expose the stream window, the displayed dyadic tolerance, and the regular-rational readback. No Real-facing consumer receives the seal row* S *from* $Q = (E, I, R, D, S, H, C, P, N)$ *without those finite rows and their transport, replay, provenance, and local naming data.*

Proof. Begin with the accepted carrier $Q = (E, I, R, D, S, H, C, P, N)$. The provenance row P names the stream, dyadic, regular-rational, and Real surfaces that the packet is allowed to consume.

The tolerance row E and index row I expose the rate request. The dyadic row D fixes the finite tolerance comparison, and the regular-rational row R fixes the readback point for the displayed Cauchy name.

The Real seal row S is downstream of those rows by the carrier definition. Componentwise transport through H and replay through C preserve the same order under every accepted presentation, and N names only this local packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyConvergenceRate}^{\uparrow}$ read exposes tolerance, unary index, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any Real-facing convergence-rate use can be consumed.

Independence witness: streamname, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{CauchyConvergenceRate}^{\uparrow}$ because this packet jointly binds a tolerance request, unary index, regular-rational readback, dyadic bound, Real seal, transport, replay, provenance, and local naming as one finite rate ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyConvergenceRate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyConvergenceRate}^{\uparrow}$ packet is generated from a displayed tolerance row, a unary index row, RegSeqRat readback, DyadicRatCore bounds, Real seal access, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyConvergenceRate}^{\uparrow}}$, seed)

Scope closed: Definition 7563.1, Definition 7563.2, and Definition 7563.3 seed the finite rate packet for Real-and-Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyConvergenceRate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected Cauchy limits, quotient equality of names, countable choice, hidden tail witnesses, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rate totality, monotone tolerance refinement, RegSeqRat handoff, RealUp nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyConvergenceRateUp.

Chapter 7564

A Concrete Naming Certificate for $\mathsf{TailWindowFilterRefinement}^{\uparrow}$

Following Bishop and Bridges / Kelley filter terminology, the $\mathsf{TailWindowFilterRefinement}^{\uparrow}$ packet records the finite common-filter refinement by which two displayed Cauchy tail-window schedules can be read through one downstream Real-facing route. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the dyadic tail envelope of chapter 1659, and the terminal real-name seal of chapter 38. The packet names a finite refinement carrier for completion consumers; it is not a quotient stream, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7564.1 (TailWindowFilterRefinement carrier)**.** *A* $\mathsf{TailWindowFilterRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S_0, S_1, R_0, R_1, D, E, W, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are* $\mathsf{StreamName}^{\uparrow}$ *tail-window rows naming the two schedules being compared.*
2. R_0 *and* R_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row requested by the downstream comparison.*
4. E *is a* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *row witnessing that both readbacks enter the same finite tail-envelope route.*
5. W *is the common refinement row, displaying a tail window through which both* S_0, R_0 *and* S_1, R_1 *factor before any* $\mathsf{Real}^{\uparrow}$ *seal is read.*
6. H *records componentwise* hsame *transport for the two schedules, two readbacks, dyadic tolerance, envelope, common refinement, replay, provenance, and naming rows.*
7. C *records displayed* Cont *replay from a refinement request through* $S_0, S_1, R_0, R_1, D, E, W, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{TailWindowFilterRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{TailWindowFilterRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, R_0, R_1, D, E, W, H, C, P, N$. *There is no coordinate for quotient equality of streams, a selected limit representative, countable choice, trichotomy, host real equality, or an ambient Real completeness principle.*

Theorem 7564.2 (TailWindowFilterRefinement NameCert obligations)**.** *For every accepted* TailWindowFilterRefin *carrier* $F = (S_0, S_1, R_0, R_1, D, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TailWindowFilterRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, two StreamName tail-window exposures, two RegSeqRat readback exposures, shared DyadicRatCore tolerance exposure, DyadicCauchyTailEnvelope exposure, common-refinement window exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No obligation may be discharged by quotient stream equality, selected limits, countable-choice selectors, trichotomy, host real equality, detached completion objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7564.1. The displayed rows are exactly $S_0, S_1, R_0, R_1, D, E, W, H, C, P, N$, so carrier admission exposes only the two schedules, two readbacks, shared dyadic tolerance, tail envelope, common refinement, transport, replay, provenance, and local naming data.

The two schedule branches are then read through the same tolerance request. Rows S_0, R_0 and S_1, R_1 provide the two finite tail-window presentations, while D fixes the dyadic scale at which the comparison is made.

The envelope and refinement rows are the only meeting point. Row E certifies that both branches enter a displayed $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ route, and W records the common tail window consumed before any $\mathsf{Real}^{\uparrow}$ seal-facing read.

Finally the structural rows preserve that route. Since the excluded quotient, selection, choice, trichotomy, host-equality, detached-completion, and ambient-completeness principles have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7564.3 (TailWindowFilterRefinement Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of an accepted* $\mathsf{TailWindowFilterRefinement}^{\uparrow}$ *carrier factors through the ordered finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{DyadicCauchyTailEnvelope}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The common refinement row W *is read before the Real seal, and no downstream consumer can bypass* $S_0, S_1, R_0, R_1, D, E, W, H, C, P, N$*.*

Proof. Begin with an accepted carrier read. The two StreamName windows and two RegSeqRat readbacks are exposed as displayed rows before a downstream consumer can inspect the shared dyadic tolerance.

Read the dyadic and envelope rows next. The tolerance row D fixes the comparison scale, and the envelope row E places both branches inside the same finite dyadic tail-envelope surface.

The common window row W supplies the refinement consumed by the Real-facing route. Transport, replay, provenance, and naming preserve that route, and the carrier has no coordinate from which a quotient stream, selected limit, countable-choice schedule, trichotomy datum, host equality, detached completion object, or ambient completeness theorem could be read. □

Falsifiable prediction: Every accepted $\mathsf{TailWindowFilterRefinement}^{\uparrow}$ read factors two StreamName tail windows and two RegSeqRat readbacks through one DyadicRatCore tolerance, one DyadicCauchyTailEnvelope row, one common refinement window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any RealUp seal-facing consumer reads the route.

Closure status for $\mathsf{NameCert}_{\mathsf{TailWindowFilterRefinement}^{\uparrow}}$**.**

Constructive story: A TailWindowFilterRefinement$^{\uparrow}$ packet is generated from two StreamName tail-window rows, two RegSeqRat readback rows, a shared DyadicRatCore tolerance row, a DyadicCauchyTailEnvelope route, a common refinement window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TailWindowFilterRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 7564.1, Definition 7564.2, and Definition 7564.3 seed the finite common-filter refinement surface over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, DyadicCauchyTailEnvelopeUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{TailWindowFilterRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, detached completion objects, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, paired StreamName exposure, RegSeqRat readback, DyadicRatCore tolerance exactness, DyadicCauchyTailEnvelope exposure, common-refinement exactness, and TasteGate field faithfulness for BEDC.Derived.TailWindowFilterRefinementUp.

Chapter 7565

A Concrete Naming Certificate for $\mathsf{LindebergCondition}^{\uparrow}$

Following Lindeberg and Feller, the $\mathsf{LindebergCondition}^{\uparrow}$ packet records the finite replacement-condition surface used by central-limit windows before any asymptotic theorem is visible. It sits after the probability source of chapter 184, the random-variable window of chapter 185, the distribution readback of chapter 186, the regular rational comparison route of chapter 301, the terminal real-name seal of chapter 38, and the finite central-limit window of chapter 5800. The packet names only displayed truncation windows, variance-budget comparisons, distribution and Real seal handoffs, transport, replay, provenance, and local naming; it does not import the asymptotic central limit theorem, characteristic-function convergence, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7565.1 (Lindeberg-condition carrier)**.** *A* $\mathsf{LindebergCondition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, X, V, T, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row X is the triangular-window* $\mathsf{RandomVar}^{\uparrow}$ *display, including the finite row index, the visible summand range, and the local law token read by each displayed summand. The row V is the variance-budget ledger naming the finite total variance request, the component variance rows, and the positive scale comparison used by the window. The row T is the truncation-threshold row, recording the displayed cutoff, the truncated summand window, and the finite exceptional-event ledger. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the truncated and untruncated finite sums. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row used to compare the displayed variance and tail budgets. The row E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after P, X, V, T, D, R have all been exposed. The row H records componentwise* hsame *transport for the probability source, triangular window, variance budget, threshold, distribution readback, RegSeqRat comparison, Real seal, replay, provenance, and local name. The row C records displayed* Cont *replay from a truncation request through P, X, V, T, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{LindebergCondition}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$. *The row N is the local* $\mathsf{NameCert}_{\mathsf{LindebergCondition}^{\uparrow}}$ *row. The classifier compares accepted packets only through these eleven rows.*

Theorem 7565.2 (Lindeberg-condition NameCert obligations)**.** *For every accepted* $\mathsf{LindebergCondition}^{\uparrow}$ *carrier $L = (P, X, V, T, D, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{LindebergCondition}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the probability source, triangular random-variable window, variance budget, truncation threshold, distribution readback, RegSeqRat comparison, Real seal, transport, replay, provenance, and local naming rows;*
2. *truncation-window exactness, requiring every tail event and truncated summand read to pass through the displayed threshold row T over the same finite window X;*
3. *variance-budget compatibility, requiring the scale and component variance reads to pass through V before any RegSeqRat comparison row R is exposed;*
4. *distribution and Real-seal handoff, requiring D and R to precede the terminal seal row E;*
5. *nonescape from the asymptotic central limit theorem, characteristic-function limits, countable choice, quotient distribution equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 7565.1. The probability source P and triangular window X provide the only random-variable data, while V and T provide the only variance and truncation coordinates.

The comparison route is then forced through D, R, E. The distribution row D reads the displayed finite sums, the RegSeqRat row R compares the finite variance and tail budgets, and the Real seal row E is visible only after those rows are already present.

Finally H, C, Q, N transport, replay, source, and name the same finite route. Since no coordinate names an asymptotic theorem, characteristic-function convergence, quotient-law equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7565.3 (Lindeberg-condition central-limit handoff)**.** *Every* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *consumer that uses Lindeberg data must first read an accepted* $\mathsf{LindebergCondition}^{\uparrow}$ *packet* $L = (P, X, V, T, D, R, E, H, C, Q, N)$*. The consumer receives only the finite replacement-condition surface consisting of the probability source, triangular random-variable window, variance budget, truncation threshold, distribution readback, RegSeqRat comparison, Real seal, transport, replay, provenance, and local naming rows; it receives no asymptotic central limit theorem, characteristic-function limit, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a central-limit finite-window consumer that asks for Lindeberg data. The consumer can inspect the probability source and triangular window only through P and X, and it can inspect tail removal only through the displayed threshold row T.

The variance and comparison data are then read through V, D, R, E. These rows provide a finite budget, finite distribution readback, rational comparison, and Real-facing seal, not an asymptotic convergence theorem.

The structural rows H, C, Q, N preserve this same finite packet for the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ route. Therefore the handoff supplies exactly the finite replacement-condition surface and none of the excluded analytic or choice principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{LindebergCondition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LindebergCondition}^{\uparrow}$ packet is generated from a probability source row, triangular random-variable window, variance-budget ledger, truncation-threshold row, distribution readback, RegSeqRat comparison row, Real seal handoff, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LindebergCondition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7565.1, Definition 7565.2, and Definition 7565.3 seed the finite Lindeberg replacement-condition surface for central-limit windows.

Formal verification: FormalStatus(NameCert$_{\text{LindebergCondition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: The next grade requires Lean encoding of the carrier and checked theorems for carrier admission, truncation-window exactness, variance-budget compatibility, distribution-to-RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 7566

A Concrete Naming Certificate for $\mathsf{SeparatedUniformCompletion}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ packet records the finite separated-reflection route by which Cauchy-filter data over a uniform source reaches a completion seal without carrying indistinguishable representatives into the public object. It sits after the uniform completion surface of chapter 3560, the Cauchy-filter interface of chapter 333, the uniform entourage source of chapter 339, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The separated completion route also consumes the Real-induced entourage basis of chapter 1624 and the generated finite subbasis route of chapter 4232 before the separated reflection row can be read. The packet names the L10 separated completion handoff as displayed NameCert data; it is not a quotient-completion construction, selected-limit operator, countable-choice principle, host equality test, trichotomy claim, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7566.1 (Separated uniform completion carrier)**.** *A* $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, F, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage-source row, exposing the finite entourage tests used by the completion consumer.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, recording the Cauchy-filter demands accepted over the displayed uniform source.*
3. *R is the separated-reflection row, identifying only those Cauchy-filter readings that the displayed uniform tests cannot distinguish.*
4. *Q is the* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *handoff row, read only after U, F, R have been displayed.*
5. *E is the completion seal row, with the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback anchors named in the provenance surface.*
6. *H records componentwise* hsame *transport for the uniform source, Cauchy-filter row, separated-reflection row, completion handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Cauchy-filter demand through the separated reflection and into the completion seal.*

8. *P is the* Pkg *provenance over* $\mathsf{SeparatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformSpaceCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedUniformCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, R, Q, E, H, C, P, N$ and their displayed replay paths. It has no coordinate for quotient representatives, hidden filter equality, selected limits, countable choice, host equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7566.2 (Separated uniform completion NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ *carrier $S = (U, F, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedUniformCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, uniform-source exposure through U, Cauchy-filter exactness through F, separated-reflection non-escape through R, completion handoff through Q, completion-seal stability through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 7566.1. The only source and Cauchy rows are U and F, so every completion-facing demand is tested against displayed uniform entourages and displayed Cauchy-filter data.

The separated-reflection row is then read before the completion handoff. The row R is the only place where indistinguishable Cauchy-filter readings are compared, and the row Q can be consumed only after that comparison has been displayed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no row for quotient representatives, hidden filter equality, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can serve as a NameCert obligation. □

Theorem 7566.3 (Separated uniform completion Cauchy-filter route)**.** *Every accepted Cauchy-filter read of a* $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{SeparatedUniformCompletion}^{\uparrow} \longmapsto \mathsf{UniformSpaceCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In the notation of Definition 7566.1, a consumer reads F against U, then the separated-reflection row R, then the completion handoff Q, and only then the completion seal E with its structural rows.

Proof. Begin with a Cauchy-filter read of the separated completion packet. The row F supplies the only Cauchy-filter data, and U supplies the only uniform entourage source available to test it.

The route then passes through separation. The row R records the displayed indistinguishability boundary before any completion handoff can be read, so no consumer can skip directly from a filter row to a completion seal.

The completion-facing rows are downstream from that boundary. The handoff Q reads the $\mathsf{UniformSpaceCompletion}^{\uparrow}$ surface, while E, H, C, P, N keep the Real-facing seal, transport, replay, provenance, and local naming attached to the same finite packet. □

Falsifiable prediction: Every accepted $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ completion read exposes a UniformSpace source row, a CauchyFilter row, a separated-reflection row, a UniformSpaceCompletion handoff, a completion seal, transport, replay, provenance, and local naming.

Independence witness: uniform-space-completion, cauchyfilter, uniformspace, regseqrat, real: $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a uniform

source, Cauchy-filter exactness, separated reflection, completion handoff, terminal seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedUniformCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedUniformCompletion}^{\uparrow}$ packet is generated from a uniform entourage source row, a Cauchy-filter row, a separated-reflection row, a UniformSpaceCompletion handoff, a completion seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7566.1, Definition 7566.2, and Definition 7566.3 seed the finite L10 separated completion route from Cauchy-filter data through uniform-source tests, separated reflection, completion handoff, seal stability, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion classes, selected limits, countable choice, hidden filter equality, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of BEDC.Derived.SeparatedUniformCompletionUp with checked carrier admission, uniform-source exposure, Cauchy-filter exactness, separated-reflection non-escape, completion handoff, seal stability, transport replay, provenance non-escape, and classifier faithfulness.

Chapter 7567

A Concrete Naming Certificate for $\mathsf{ClosedMetricSubspace}^{\uparrow}$

Following Kelley and Bishop–Bridges metric-space practice, the $\mathsf{ClosedMetricSubspace}^{\uparrow}$ packet records the finite closed-subspace route by which a displayed subspace admission row inherits convergence from a parent metric carrier. It refines the subspace surface of chapter 1519, reads the parent metric packet of chapter 57, consumes the complete-metric limit boundary of chapter 128, and keeps its convergence data visible through chapter 126, chapter 301, and chapter 38. The resulting carrier is a closed-subspace certificate, not a host-subset quotient, selected-representative principle, arbitrary completeness theorem, or compactness assertion.

Definition 7567.1 (ClosedMetricSubspace carrier)**.** *A* $\mathsf{ClosedMetricSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, A, D, L, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *M is the parent* $\mathsf{Met}^{\uparrow}$ *row, including its point classifier, distance witness, metric-law ledger, and finite observation histories.*
2. *A is the displayed subspace admission row. A point is readable by the closed-subspace packet only after the parent point classifier of M and the admission row A both accept it.*
3. *D is the inherited-distance row. It exports no independent distance; each closed-subspace distance read is the parent distance row of M applied to two admitted points.*
4. *L is the closed-under-limit witness. It states that a parent limit witness for an admitted Cauchy stream is admitted again by A.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *Cauchy witness row naming the finite observation windows used by the convergence consumer.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row recording the regular rational comparison data attached to the stream.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after the stream, regular-rational, inherited-distance, and closed-limit rows have been displayed.*
8. *H records componentwise* hsame *transport for the parent metric row, admission row, inherited distance, closed-limit witness, stream witness, regular-rational readback, Real seal, replay row, provenance row, and local naming row.*

9. *K records the displayed* Cont *route replaying point admission, subspace distance restriction, Cauchy-window admission, parent limit handoff, closedness admission, and Real-seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedMetricSubspace}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricSubspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $M, A, D, L, S, R, E, H, K, P, N$ and their replay paths. It has no coordinate for host subset equality, quotient membership, selected representatives, arbitrary completeness, open-cover compactness, countable choice, or ambient Real completeness.

Theorem 7567.2 (ClosedMetricSubspace NameCert obligations)**.** *For every accepted* $\mathsf{ClosedMetricSubspace}^{\uparrow}$ *carrier $C = (M, A, D, L, S, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricSubspace}^{\uparrow}}$ *surface consists exactly of carrier admission, inherited metric restriction, closure under parent limit witnesses, convergence handoff through StreamName windows and RegSeqRat readback, transport, replay, provenance, and local naming.*

Equivalently, every completion-facing or Real-facing consumer must read

$$M, \quad A, \quad D, \quad L, \quad S, \quad R, \quad E, \quad H, \quad K, \quad P, \quad N$$

before it can use a closed-subspace conclusion, and no consumer can import host subsets, quotient equality, selected representatives, arbitrary completeness, open-cover compactness, or bridge data.

Proof. Carrier admission is projection from Definition 7567.1. The tuple lists the parent metric, displayed admission, inherited distance, closed-limit witness, StreamName Cauchy witness, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

The inherited metric restriction is forced by D and K. A distance read first admits both endpoints through M and A, then reads the parent distance row of M; the carrier contains no second metric coordinate.

Closure under parent limits is the role of L. A convergence consumer opens the finite StreamName row S, reads the RegSeqRat comparisons in R, uses the parent metric and complete-metric boundary named by M, and then applies L to admit the parent limit back into A.

Finally E, H, K, P, N seal, transport, replay, source, and name the same route. Since the carrier has no row outside $M, A, D, L, S, R, E, H, K, P, N$, host subset equality, quotient equality, selected representatives, arbitrary completeness, open-cover compactness, and bridge checking cannot be exported by the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{ClosedMetricSubspace}^{\uparrow}$ convergence read exposes a parent Metric row, displayed admission row, inherited-distance row, closed-under-limit witness, StreamName Cauchy witness, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming before any closed-subspace conclusion is consumed.

Independence witness: metric-subspace, metric, complete-metric, streamname, regseqrat, real: $\mathsf{ClosedMetricSubspace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries displayed subspace admission, inherited distance, closed-limit admission, Cauchy-window data, regular-rational readback, terminal Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedMetricSubspace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedMetricSubspace}^{\uparrow}$ packet is generated from a parent Metric row, displayed subspace admission row, inherited distance row, closed-under-limit witness, StreamName Cauchy witness, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ClosedMetricSubspace}\uparrow}$, seed)

Scope closed: Definition 7567.1 and Definition 7567.2 seed the finite L10 closed-subspace route from admitted Cauchy-window data through inherited distance, parent limit witnessing, closedness admission, Real seal handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClosedMetricSubspace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subsets, quotient equality, selected representatives, arbitrary completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inherited metric restriction, closure under parent limit witnesses, convergence handoff, transport replay, provenance non-escape, and classifier faithfulness for BEDC.Derived.ClosedMetricSubspaceUp.

Chapter 7568

A Concrete Naming Certificate for $\mathsf{DirichletUniformModulus}^{\uparrow}$

Following Dirichlet and Abel summation-by-parts, the $\mathsf{DirichletUniformModulus}^{\uparrow}$ packet records the finite uniform-tail budget obtained from bounded partial sums and monotone coefficient windows before any Real-or-Completion consumer reads a series endpoint. It sits beside the summation-by-parts certificate of chapter 1763, the Dirichlet convergence-test certificate of chapter 1497, the shared uniform-tail route of chapter 1736, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a constructive modulus object for displayed tails; it is not a host convergence theorem, a selected infinite sum, quotient series equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7568.1 (Dirichlet uniform modulus carrier)**.** *A* $\mathsf{DirichletUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, M, D, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded-partial-sum row, exposing the inspected partial sums and one displayed uniform bound for the relevant source windows.*
2. *A is the Abel summation-by-parts row, carrying the finite boundary and difference-window data used to rewrite the inspected tail.*
3. *M is the monotone coefficient-window row, naming the finite comparisons and endpoint coefficient values visible on each requested tail.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row, recording the dyadic radius, comparison slack, and arithmetic loss used by the displayed estimate.*
5. *T is the exported uniform-tail modulus row, assigning to each displayed dyadic request a named tail window whose later reads must pass through S, A, M, D.*
6. *H records componentwise* hsame *transport for the bounded-sum, summation-by-parts, monotone-window, dyadic-budget, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of a precision request through*

$$S \longmapsto A \longmapsto M \longmapsto D \longmapsto T.$$

8. *P is the* Pkg *provenance over* DirichletUniformModulus$^{\uparrow}$, AbelSummation$^{\uparrow}$, DirichletTest$^{\uparrow}$, UniformCauchySeries$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{DirichletUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for a selected limit, quotient series, countable-choice threshold, host series equality, or ambient real-completeness witness.

Lean4 (definition) `BEDC.Derived.DirichletUniformModulusUp.DirichletUniformModulusCarrier`

Theorem 7568.2 (Dirichlet uniform modulus tail budget)**.** *For every accepted* DirichletUniformModulus$^{\uparrow}$ *carrier $K = (S, A, M, D, T, H, C, P, N)$, the exported row T is a uniform tail modulus only through the displayed summation-by-parts route*

$$\sum_{k=m}^{n} a_k b_k = A_m B_m - A_n B_{n+1} + \sum_{k=m}^{n-1} A_k (b_k - b_{k+1})$$

and the dyadic-budget rows S, M, D. Thus any Real-or-Completion consumer reading the modulus must first display the bounded partial-sum bound, the monotone coefficient window, and the dyadic loss ledger.

Proof. Project the carrier of Definition 7568.1. The only analytic rows upstream of T are S, A, M, D, so every tail request reaches the modulus through those displayed coordinates.

The row A supplies the summation-by-parts identity on the inspected finite window. The row S bounds the partial sums that occur in the boundary and difference terms, while M names the monotone coefficient comparisons that make the boundary and variation terms readable on the same window.

The dyadic budget row D records the requested radius and the arithmetic slack assigned to the boundary and difference contributions. Reading T therefore amounts to reading the same finite estimate after the dyadic losses have been allocated.

The structural rows H, C, P, N transport, replay, source, and name this route. Since the carrier contains no selected-limit, quotient-series, choice, host-equality, or ambient-completeness coordinate, no consumer can export a uniform tail modulus except through the displayed rows. □

Lean4 ✓ `BEDC.Derived.DirichletUniformModulusUp.DirichletUniformModulusTailBudget`

Theorem 7568.3 (Dirichlet uniform modulus NameCert obligations)**.** *For every accepted* DirichletUniformModulus$^{\uparrow}$ *carrier $K = (S, A, M, D, T, H, C, P, N)$, the local* NameCert$_{\text{DirichletUniformModulus}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the finite packet K and its displayed* BHist *rows;*
2. *bounded-partial-sum and Abel summation-by-parts exposure through S and A;*
3. *monotone coefficient-window and dyadic tail-budget exactness through M and D;*
4. *uniform-tail modulus export through T, with every readback to* UniformCauchySeries$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *downstream of S, A, M, D;*
5. *componentwise transport, displayed replay, provenance, and local naming through exactly H, C, P, N.*

No NameCert obligation is stored outside those five groups.

Lean4 ✓ `BEDC.Derived.DirichletUniformModulusUp.DirichletUniformModulusNameCertObligations`

Proof. Carrier admission is the direct read of the tuple K. The accepted packet exposes exactly the nine rows listed in Definition 7568.1.

The analytic evidence is split into two displayed groups. Rows S and A provide the bounded partial-sum data and finite summation-by-parts rewrite, while rows M and D provide the monotone coefficient window and dyadic loss accounting.

The public modulus obligation is row T. By Definition 7568.2, a read of T factors through S, A, M, D before any uniform Cauchy, regular readback, or Real-facing consumer can inspect the tail bound.

Finally read H, C, P, N. These rows transport, replay, source, and name the same finite route, and the carrier supplies no further coordinate. The five listed groups therefore exhaust the local certificate surface. □

Falsifiable prediction: An accepted $\mathsf{DirichletUniformModulus}^{\uparrow}$ read is refuted if a uniform Cauchy, regular readback, Real, or Completion consumer reaches the exported modulus row without first displaying the bounded partial-sum row, summation-by-parts row, monotone coefficient window, dyadic budget, transport, replay, provenance, and local NameCert rows.

Independence witness: abel-summation, dirichlet-test, uniform-cauchy-series, dyadicratcore, regseqrat, real: $\mathsf{DirichletUniformModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded partial sums, summation-by-parts data, monotone coefficient windows, dyadic tail budgets, a named uniform-tail modulus, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletUniformModulus}^{\uparrow}$ packet is generated from bounded partial-sum rows, Abel summation-by-parts windows, monotone coefficient windows, dyadic tail budgets, an exported uniform-tail modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7568.1, Definition 7568.2, and Definition 7568.3 seed the finite constructive uniform-tail budget route over bounded partial sums, summation-by-parts rows, monotone coefficient windows, dyadic budgets, uniform Cauchy series readback, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletUniformModulus}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.DirichletUniformModulusUp.DirichletUniformModulusCarrier`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient series, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, host series equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded partial-sum exposure, summation-by-parts exposure, monotone coefficient-window exactness, dyadic tail-budget exactness, uniform-tail modulus export, transport, replay, provenance, and local naming for BEDC.Derived.DirichletUniformModulusUp.

Chapter 7569

A Concrete Naming Certificate for $\mathsf{KuratowskiEmbedding}^{\uparrow}$

Following Bourbaki's and Kelley's metric-space treatment of distance functions, the $\mathsf{KuratowskiEmbedding}^{\uparrow}$ packet records the classical embedding of a metric space into a space of distance-to-point functions as finite BEDC data. It sits over the metric source of chapter 57, the real-distance rows of chapter 38, and the distance-preserving interface of chapter 439. The packet exposes a selected basepoint or displayed family row, distance-function target rows, and the isometric route; it does not import quotient completion, host function extensionality, or a detached real-valued function space.

Definition 7569.1 (Kuratowski embedding carrier)**.** *A* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, B, D, T, I, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose point and distance-witness data supply the domain of the embedding.*
2. *B is the selected basepoint or displayed point-family row used to normalize and compare distance functions.*
3. *D is the* $\mathsf{Real}^{\uparrow}$ *distance-witness row recording the finite reads $d(x, b)$ and $d(y, b)$ against the selected basepoint or family.*
4. *T is the distance-function target row naming the displayed family of* $\mathsf{Real}^{\uparrow}$ *distance observations rather than a host function space.*
5. *I is the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *route row witnessing that source distances are preserved by the displayed distance-function target.*
6. *H records componentwise* hsame *transport for M, B, D, T, I.*
7. *C records displayed* Cont *replay from metric-distance reads to target distance-function reads and then to the isometric route.*
8. *P is the* Pkg *provenance over* $\mathsf{KuratowskiEmbedding}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{IsometricEmbedding}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, host function extensionality, ambient Banach-space structure, or a selected completion representative.

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingTasteGate single carrier alignment`

Theorem 7569.2 (Kuratowski embedding NameCert obligations)**.** *For the carrier of Definition 7569.1, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$ *surface consists of six finite rows:*

1. *carrier admission: every accepted packet exposes* $K = (M, B, D, T, I, H, C, P, N)$ *and no hidden function-space coordinate;*
2. *metric-source exposure: every source point and source-distance read enters through the* $\mathsf{Met}^{\uparrow}$ *row* M*;*
3. *basepoint or family selection: the row* B *is the only displayed source for the normalizing point or comparison family;*
4. *real-distance target exposure: the rows* D *and* T *turn source-distance reads into* $\mathsf{Real}^{\uparrow}$ *distance-function observations before any target comparison is made;*
5. *isometric route: the row* I *is the only distance-preserving handoff, and it is read after* M, B, D, T*;*
6. *structural replay: the rows* H, C, P, N *preserve, replay, source, and name the same route without exporting quotient completion or host function extensionality.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 7569.1. The accepted packet displays the metric source, basepoint or family row, real-distance witness row, target row, isometric route, transport, continuation, provenance, and local naming row.

Read the dependency rows in order. The metric source row M supplies point and distance-witness data; the row B supplies the selected basepoint or displayed family; and the real-distance row D records the corresponding $\mathsf{Real}^{\uparrow}$ observations.

The target row T is therefore a displayed distance-function surface, not a host function space. The isometric row I can only be read after those source and target observations are visible, and H, C, P, N preserve that same finite route. □

The route consumes the metric source of chapter 57, the real-distance rows of chapter 38, and the distance-preserving handoff of chapter 439.

Theorem 7569.3 (Kuratowski embedding distance route)**.** *Every accepted* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *packet routes source-distance comparison through*

$$M \to B \to D \to T \to I \to H \to C \to P \to N.$$

Thus a consumer reaches the $\mathsf{IsometricEmbedding}^{\uparrow}$ *handoff only after reading the* $\mathsf{Met}^{\uparrow}$ *source, the selected basepoint or family row, and the* $\mathsf{Real}^{\uparrow}$ *distance-function target rows. No consumer receives quotient completion, host function extensionality, ambient Banach-space structure, or a selected completion representative from the packet.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingCarrier distance route`

Proof. Start with an accepted carrier $K = (M, B, D, T, I, H, C, P, N)$. A source-distance query must read M, and a distance-function query must next read the basepoint or family row B.

The real-valued observations are then stored in D and organized by T. Because T is part of the finite carrier, it supplies only displayed $\mathsf{Real}^{\uparrow}$ distance observations and not host-level function extensionality.

Finally apply the isometric route row I. The structural rows H, C, P, N keep the same route under transport and replay, so the only exported embedding data are the displayed metric, basepoint or family, real-distance, target, and isometric rows. □

Theorem 7569.4 (Kuratowski embedding distance readback exactness)**.** *Every accepted* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *packet reads source metric distance exactly through the displayed route*

$$M \to D \to T \to I.$$

The $\mathsf{Met}^{\uparrow}$ *row* M *supplies the source distance query, the* $\mathsf{Real}^{\uparrow}$ *row* D *supplies the distance witnesses used by the distance-function target* T*, and the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *row* I *is the first row from which a completion, hyperspace, or distance-function consumer may inspect the resulting target comparison. No consumer may read a distance-function target as exact before the metric source, real-distance witnesses, and isometric handoff have all been displayed.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbedding distance readback exactness`

Proof. Project the accepted carrier $K = (M, B, D, T, I, H, C, P, N)$ from Definition 7569.1. The source-distance query starts in M, while the basepoint or displayed family row B fixes the finite comparison context.

Apply the NameCert obligations of Definition 7569.2. They force every source-distance read through M and every real-valued distance observation through D before the target row T can be inspected.

Finally use the source-distance route of Definition 7569.3. The isometric handoff I is downstream of M, B, D, T, and the structural rows H, C, P, N only preserve that route. Thus completion, hyperspace, and distance-function consumers receive exactness through the displayed isometric readback rather than from an ambient function space. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KuratowskiEmbedding}^{\uparrow}$ packet is generated from a MetricUp source row, a selected basepoint or displayed family row, RealUp distance-witness rows, a distance-function target row, an IsometricEmbeddingUp route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7569.1, Definition 7569.2, and Definition 7569.3 seed the finite distance-function embedding surface over MetricUp, RealUp, and IsometricEmbeddingUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion, host function extensionality, ambient Banach-space structure, selected completion representatives, or bridge checking.

Upgrade path: Lean closure requires checked carrier admission, basepoint or family exposure, RealUp distance target exposure, isometric-route preservation, and structural replay proofs for BEDC.Derived.KuratowskiEmbeddingUp.

Chapter 7570

A Concrete Naming Certificate for KuratowskiEmbeddingDistanceProfile$^{\uparrow}$

Following Kuratowski's metric embedding of a metric space into distance functions, the KuratowskiEmbeddingDistan packet records the finite BEDC row that turns a source point or compact-source query into a displayed distance-profile classifier. It sits between the embedding carrier of chapter 7569, the finite compact-subset comparison of chapter 2086, the compact-metric source of chapter 129, and the metric-distance surface of chapter 57. The packet names the distance-profile classifier consumed by hyperspace and Hausdorff-facing chapters; it is not a host function space, quotient subset equality, selected completion representative, ambient supremum construction, or open-cover compactness theorem.

Definition 7570.1 (Kuratowski embedding distance-profile packet)**.** *A* KuratowskiEmbeddingDistanceProfile$^{\uparrow}$ *packet is a finite* BHist *carrier*

$$P = (M, K, E, H, D, R, S, C, Q, N)$$

with the following displayed rows:

1. *M is the source* Met$^{\uparrow}$ *row carrying the visible point names, source-distance witnesses, and tolerance requests for the profile read.*
2. *K is the optional* CompactMetric$^{\uparrow}$ *support row used when the profile is read against compact finite-net data.*
3. *E is the accepted* KuratowskiEmbedding$^{\uparrow}$ *row that converts the metric source into displayed distance-function observations.*
4. *H is the* HausdorffMetric$^{\uparrow}$ *comparison row available only when two compact-supported profile reads are compared.*
5. *D is the distance-profile classifier row, listing the finite family of displayed distance observations and the tolerance window used by the consumer.*
6. *R is the readback row that routes each displayed distance observation to the same metric and compact-support data named by M, K, E, H.*
7. *S records componentwise* hsame *transport for the metric, compact-support, embedding, Hausdorff, profile, readback, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay along the route*

$$M, K, E, H \longmapsto D \longmapsto R.$$

9. *Q is the* Pkg *provenance over* KuratowskiEmbeddingDistanceProfile$^{\uparrow}$, KuratowskiEmbedding$^{\uparrow}$, HausdorffMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{KuratowskiEmbeddingDistanceProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host real-valued function extensionality, quotient compact-set equality, selected completion representatives, ambient suprema, open-cover compactness, or a detached hyperspace object.

Theorem 7570.2 (Kuratowski embedding distance-profile NameCert obligations)**.** *For every accepted* KuratowskiEmbeddingDistanceProfile$^{\uparrow}$ *packet $P = (M, K, E, H, D, R, S, C, Q, N)$, the local* NameCert$_{\text{KuratowskiEmbeddingDistanceProfile}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric, compact-support, Kuratowski embedding, Hausdorff comparison, distance-profile, readback, transport, replay, provenance, and local naming rows;*
2. *source exposure, requiring every distance-profile observation in D to be read through the same* Met$^{\uparrow}$ *row M and, when compact support is present, through the same* CompactMetric$^{\uparrow}$ *row K;*
3. *embedding-profile coherence, requiring the profile row D to be downstream of the* KuratowskiEmbedding$^{\uparrow}$ *row E rather than a detached host function;*
4. *Hausdorff compatibility, requiring any compact-supported comparison to read the* HausdorffMetric$^{\uparrow}$ *row H before comparing two profile windows;*
5. *readback and transport, requiring R, S, C to replay and transport only the displayed finite route;*
6. *provenance exhaustion, requiring Q, N to name only the cited BEDC rows and to export no host function extensionality, quotient compact-set equality, selected completion representative, ambient supremum, open-cover compactness, or detached hyperspace object.*

No distance-profile obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7570.1. The accepted packet displays exactly the metric source, compact support, Kuratowski embedding row, Hausdorff comparison row, profile row, readback row, and structural rows.

Source exposure follows by reading M first and K when the profile is compact-supported. The profile row D has no independent source coordinate, so every visible distance observation is tied to the same metric and compact-support data.

Embedding-profile coherence is the projection to E, D. The row E is the only embedding row in the carrier, and D is read after it, so the profile is a displayed Kuratowski distance-function read rather than a host function-space object.

For compact-supported comparison, the row H supplies the Hausdorff finite-net comparison before any two profile windows are compared. The readback and transport rows R, S, C preserve and replay that same finite route.

Finally, Q, N exhaust provenance and local naming. Since the carrier has no coordinate for host function extensionality, quotient compact-set equality, selected completion representatives, ambient suprema, open-cover compactness, or detached hyperspace objects, none of these can be exported by the NameCert surface. □

Theorem 7570.3 (Kuratowski embedding distance-profile handoff)**.** *Every hyperspace-facing, Hausdorff-facing, or compact-metric-facing consumer of an accepted* KuratowskiEmbeddingDistanceProfile$^{\uparrow}$ *packet factors through*

$$M, K, E, H \longmapsto D \longmapsto R \longmapsto S, C, Q, N.$$

Consequently the consumer receives only the displayed metric source, compact-support row, Kuratowski embedding row, Hausdorff comparison row, finite distance-profile classifier, readback, componentwise transport, replay, provenance, and local name. It cannot claim host real-valued function extensionality, quotient compact-set equality, a selected completion representative, an ambient supremum, open-cover compactness, or a detached hyperspace object from the packet.

Proof. Start with an accepted packet $P = (M, K, E, H, D, R, S, C, Q, N)$. A profile read must first expose the metric source M and the compact-support row K when the read is compact-supported.

The Kuratowski embedding row E then supplies the displayed distance-function observations. If the consumer compares compact-supported profiles, it must also read the Hausdorff comparison row H before the profile classifier D is public.

The readback row R ties each displayed profile coordinate to the same source route. Componentwise transport, replay, provenance, and local naming are then supplied by S, C, Q, N without adding any new object coordinate.

Thus the consumer receives exactly the finite rows listed in the statement and none of the excluded host-level function, quotient, representative, supremum, compactness, or hyperspace data. □

Closure status for NameCert$_{\text{KuratowskiEmbeddingDistanceProfile}^{\uparrow}}$.

Constructive story: A KuratowskiEmbeddingDistanceProfile$^{\uparrow}$ packet is generated from MetricUp source rows, optional CompactMetricUp support, KuratowskiEmbeddingUp distance-function observations, HausdorffMetricUp compact-supported comparisons, a finite distance-profile classifier, readback rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KuratowskiEmbeddingDistanceProfile}^{\uparrow}}$, seed)

Scope closed: Definition 7570.1, Definition 7570.2, and Definition 7570.3 seed the finite distance-profile classifier route between Kuratowski embedding, Hausdorff metric comparison, compact-metric support, and metric source data.

Formal verification: FormalStatus(NameCert$_{\text{KuratowskiEmbeddingDistanceProfile}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host function extensionality, quotient compact-set equality, selected completion representatives, ambient suprema, open-cover compactness, detached hyperspace objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, Kuratowski profile coherence, Hausdorff compatibility, readback, transport, replay, provenance, and local NameCert exhaustion for BEDC.Derived.KuratowskiEmbeddingDista

Chapter 7571

A Concrete Naming Certificate for $\mathsf{SequentialCompactnessModulus}^{\uparrow}$

Following Bishop–Bridges compact metric arguments, the $\mathsf{SequentialCompactnessModulus}^{\uparrow}$ packet records the finite modulus data by which a compact metric source exposes bounded sequence windows, nested finite covers, dyadic tolerance requests, regular rational readback, and a terminal Real seal before any cluster-facing consumer may proceed. It sits after the compact metric source of chapter 129, total boundedness data from chapter 127, the completion boundary of chapter 128, finite stream windows from chapter 126, regular rational readback from chapter 301, and the Real naming seal of chapter 38. The packet supplies explicit finite modulus data for sequential compactness evidence consumed by Bolzano–Weierstrass and compact-metric sequence arguments; it is not a proof of general sequential compactness.

Definition 7571.1 (SequentialCompactnessModulus carrier)**.** *A* $\mathsf{SequentialCompactnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, T, C, S, D, R, E, Q, H, U, P, N)$$

with the following rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric carrier used by the sequence argument.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-cover row, listing the cover cells inspected at the displayed tolerances.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row, recording that Cauchy-style readback may be sealed only through the displayed completion boundary.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the sequence being inspected.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the precision requests and dyadic margins used by the finite cover refinement.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected finite windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after K, T, C, S, D, R have been displayed.*
8. *Q is the cluster handoff row, giving the downstream Bolzano–Weierstrass or compact-metric sequence consumer the displayed finite modulus route.*

9. *H records componentwise* hsame *transport for the source, cover, completion-boundary, stream-window, tolerance, readback, seal, and handoff rows.*
10. *U records displayed* Cont *replay for refinements of the finite tolerance and window budget.*
11. *P is the* Pkg *provenance over* $\mathsf{SequentialCompactnessModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompactnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, D, R, E, Q, H, U, P, N$. It has no coordinate for a general sequential compactness theorem, a selected infinite subsequence, countable choice, open-cover compactness, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7571.2 (SequentialCompactnessModulus NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompactnessModulus}^{\uparrow}$ *carrier*

$$M = (K, T, C, S, D, R, E, Q, H, U, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompactnessModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source K, total-bounded finite-cover row T, completion-boundary row C, stream-window row S, dyadic tolerance row D, regular readback row R, Real seal E, cluster handoff Q, and structural rows H, U, P, N;*
2. *finite-modulus exposure, requiring every tolerance request in D to be matched with a displayed finite cover cell in T and a finite stream window in S;*
3. *completion-boundary discipline, requiring every Cauchy-style readback to pass through C before reaching R and E;*
4. *cluster handoff nonescape, requiring any Bolzano–Weierstrass or compact-metric sequence consumer to read Q only after the route*

$$K, T, C, S, D \longmapsto R \longmapsto E$$

   has been displayed;
5. *provenance and local naming, requiring H, U, P, N to transport, replay, source, and name only the displayed finite rows.*

No obligation row contains general sequential compactness, selected subsequences, countable choice, open-cover compactness, quotient stream equality, or ambient Real completeness.

Proof. Project the carrier of Definition 7571.1. The rows $K, T, C, S, D, R, E, Q, H, U, P, N$ are the whole accepted packet.

For finite-modulus exposure, read D together with T and S. Each precision request is paired with a displayed finite cover cell and a finite stream window before any readback row is visible.

For the completion-boundary discipline, route the selected window and tolerance data through C before R and E. The packet therefore uses completion only as the displayed boundary row named by chapter 128.

The cluster handoff Q is downstream of the same finite route. Since H, U, P, N only transport, replay, source, and name the displayed rows, the packet exports finite modulus data and no host sequential compactness, selected subsequence, countable-choice schedule, open-cover theorem, quotient stream equality, or ambient Real completeness. □

Lean4 (statement) `BEDC.Derived.SequentialCompactnessModulusUp`

Theorem 7571.3 (SequentialCompactnessModulus compact-metric sequential-net route)**.** *Every accepted* SequentialCompactnessModulus$^{\uparrow}$ *consumer that supplies a compact-metric sequence argument through a sequential-net packet must read the compact metric source row K, total-bounded finite-cover row T, complete-metric boundary row C,* StreamName$^{\uparrow}$ *finite-window row S,* DyadicRatCore$^{\uparrow}$ *tolerance row D,* RegSeqRat$^{\uparrow}$ *readback row R, terminal* Real$^{\uparrow}$ *seal E, and cluster handoff row Q before consuming* CompactMetricSequentialNet$^{\uparrow}$ *through chapter 4019. The route exports finite modulus data only; it does not supply general sequential compactness, selected subsequences, countable choice, open-cover compactness, quotient stream equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 7571.1. The accepted packet exposes exactly the compact metric source, total-bounded cover, complete-metric boundary, stream-window, dyadic tolerance, readback, Real seal, cluster handoff, and structural rows.

Apply Definition 7571.2. Its finite-modulus exposure reads D together with T and S, while its completion-boundary discipline routes Cauchy-style readback through C before R and E.

Now read the compact-metric sequential-net sibling of chapter 4019. The cluster handoff row Q can feed CompactMetricSequentialNet$^{\uparrow}$ only after the source, finite-cover, boundary, window, tolerance, readback, and seal rows have been displayed.

The structural rows H, U, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores a general sequential-compactness theorem, a selected subsequence, a countable-choice schedule, open-cover compactness, quotient stream equality, or ambient Real completeness, the sibling route remains a finite modulus handoff. □

Falsifiable prediction: A SequentialCompactnessModulus$^{\uparrow}$ consumer that obtains a cluster handoff without reading compact metric source data, total-bounded finite covers, completion-boundary data, finite StreamName windows, dyadic tolerances, RegSeqRat readback, and the Real seal refutes the carrier surface.

Independence witness: compactmetric, totallybounded, completemetric, streamname, regseqrat, real: no listed sibling supplies the whole finite modulus packet because each sibling contributes compact source data, cover finiteness, completion-boundary discipline, observation windows, regular readback, or sealing, while SequentialCompactnessModulus$^{\uparrow}$ binds the cluster-facing route with transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{SequentialCompactnessModulus}^{\uparrow}}$**.**

Constructive story: A SequentialCompactnessModulus$^{\uparrow}$ packet is generated from CompactMetric source data, TotallyBounded finite covers, CompleteMetric boundary rows, StreamName finite windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, cluster handoff data, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SequentialCompactnessModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7571.1 and Definition 7571.2 seed the finite modulus surface for sequential compactness evidence over compact-metric sequence consumers.

Formal verification: FormalStatus(NameCert$_{\text{SequentialCompactnessModulus}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.SequentialCompactnessModulusUp`

Bridge status: none

Not claimed: This chapter does not close general sequential compactness, selected infinite subsequences, countable choice, open-cover compactness, quotient stream equality, ambient RealUp completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-modulus exposure, completion-boundary discipline, cluster handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialCompactnessModulusUp.

Chapter 7572

A Concrete Naming Certificate for HausdorffLocatedUniformReflection$^{\uparrow}$

Following Kelley and Bourbaki, the HausdorffLocatedUniformReflection$^{\uparrow}$ packet records the finite located separated-reflection boundary for uniform data whose completion-facing consumers must keep Real-induced tests visible. It sits after the Real uniformity surface of chapter 1624, the separated completion packet of chapter 7566, the generated Cauchy-filter interface of chapter 4232, the uniform entourage source of chapter 339, and the terminal Real-name seal of chapter 38. The packet names a finite reflection route only; it is not a quotient-completion construction, selected-limit operator, countable-choice principle, trichotomy claim, host equality test, or ambient Real$^{\uparrow}$ completeness theorem.

Definition 7572.1 (Hausdorff located uniform reflection carrier)**.** *A* HausdorffLocatedUniformReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (R, U, S, B, F, L, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *R is the* RealUniformStructure$^{\uparrow}$ *row exposing Real endpoints, metric distances, positive dyadic radii, and generated entourages.*
2. *U is the* UniformSpace$^{\uparrow}$ *entourage-source row read by the reflection consumer.*
3. *S is the* CauchyFilterSubbasis$^{\uparrow}$ *row displaying primitive subbasis tokens, finite intersections, generated events, and the generation ledger.*
4. *B is the generated* FilterBase$^{\uparrow}$ *row obtained from S.*
5. *F is the* CauchyFilter$^{\uparrow}$ *row read through B and U.*
6. *L is the locatedness row, recording the finite apartness and neighbourhood tests used before separation is public.*
7. *Q is the separated-reflection row consumed by* SeparatedUniformCompletion$^{\uparrow}$ *readers.*
8. *E is the completion-facing seal row, naming the retained* RegSeqRat$^{\uparrow}$ *readback and terminal* Real$^{\uparrow}$ *seal.*
9. *T records componentwise* hsame *transport for Real-uniform data, uniform entourages, subbasis generation, generated bases, Cauchy filters, located tests, separated reflection, seal rows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from Real-uniform entourage requests through generated subbasis bases, Cauchy-filter rows, located tests, separated reflection, and the seal.*
11. *P is the* Pkg *provenance over* HausdorffLocatedUniformReflection$^{\uparrow}$, RealUniformStructure$^{\uparrow}$, SeparatedUniformComp CauchyFilterSubbasis$^{\uparrow}$, FilterBase$^{\uparrow}$, CauchyFilter$^{\uparrow}$, UniformSpace$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{HausdorffLocatedUniformReflection}^{\uparrow}}$ *row.*

The classifier reads only $R, U, S, B, F, L, Q, E, T, C, P, N$ and their displayed replay paths. It has no coordinate for quotient completion, selected limits, countable choice, hidden filter equality, host real equality, trichotomy, or ambient completeness of Real$^{\uparrow}$.

Theorem 7572.2 (Hausdorff located uniform reflection NameCert obligations)**.** *For every accepted* HausdorffLocatedUniformReflection$^{\uparrow}$ *carrier $H = (R, U, S, B, F, L, Q, E, T, C, P, N)$, the local* NameCert$_{\text{HausdorffLocatedUniformReflection}^{\uparrow}}$ *surface consists exactly of carrier admission, Real-induced uniformity exposure through R, uniform entourage exposure through U, finite subbasis generation through S, B, Cauchy-filter exactness through F, locatedness testing through L, separated-reflection non-escape through Q, completion-facing seal stability through E, and structural transport, replay, provenance, and local naming through T, C, P, N.*

Proof. Project the carrier of Definition 7572.1. The Real-induced uniformity, uniform entourage, subbasis, generated-base, Cauchy-filter, locatedness, separated-reflection, seal, transport, replay, provenance, and local naming rows are exactly $R, U, S, B, F, L, Q, E, T, C, P, N$.

Read the generation route first. The rows S and B expose the finite subbasis and generated base before the Cauchy-filter row F is available, while R and U keep the Real-induced entourage source visible to the same consumer.

The located and separated rows are downstream from those data. The row L supplies the finite located tests, and Q records the separated reflection consumed by the completion-facing seal E.

Finally T, C, P, N preserve and name the same finite route. Since the carrier has no row for quotient completion, selected limits, countable choice, hidden filter equality, host real equality, trichotomy, or ambient Real$^{\uparrow}$ completeness, none of those principles is a NameCert obligation. □

Theorem 7572.3 (Hausdorff located uniform reflection completion route)**.** *Every completion-facing read of an accepted* HausdorffLocatedUniformReflection$^{\uparrow}$ *carrier factors through*

$$\text{RealUniformStructure}^{\uparrow} \longmapsto \text{CauchyFilterSubbasis}^{\uparrow} \longmapsto \text{FilterBase}^{\uparrow} \longmapsto \text{CauchyFilter}^{\uparrow} \longmapsto \text{SeparatedUniformCompletio}$$

In the notation of Definition 7572.1, a consumer reads Real-uniform and uniform-source rows R, U, finite subbasis and generated-base rows S, B, the Cauchy-filter row F, located and separated rows L, Q, and only then the completion-facing seal E with its structural rows.

Proof. Start with the Real-uniform and generated-base inputs. The rows R, U, S, B are the only displayed sources for entourage and subbasis generation, so a completion-facing read cannot begin from an ambient uniform completion object.

Apply the generated-base discipline of the CauchyFilterSubbasis$^{\uparrow}$ route. It keeps the Cauchy-filter row F downstream from the finite subbasis tokens, finite intersections, generated events, and generated FilterBase$^{\uparrow}$ row.

Now read the separated reflection. The located row L supplies the finite tests used before Q is public, and the seal row E is downstream from that separated-reflection boundary.

The structural rows T, C, P, N transport, replay, source, and name the same path. Because no quotient, selected-limit, choice, trichotomy, host-equality, or ambient-completeness coordinate occurs in the carrier, the displayed route is the complete completion-facing surface. □

Theorem 7572.4 (Hausdorff located uniform reflection Real-uniform handoff)**.** *Every accepted* $\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}$ *consumer that asks for a Real-induced uniformity read must inspect the* $\mathsf{RealUniformStructure}^{\uparrow}$ *row* R*, the* $\mathsf{UniformSpace}^{\uparrow}$ *row* U*, the generated subbasis rows* S, B*, and the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *before the located row* L *or separated-reflection row* Q *can be used. The handoff exports no quotient completion, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier rows in Definition 7572.1. The only Real-uniform input is R, the only independent uniform source is U, and the only generated-base path is S, B.

The Cauchy-filter row is downstream of those rows. A consumer reaches F only after the Real-induced entourage and subbasis-generated base have been displayed.

The located and separated rows are later in the same packet. Rows L and Q therefore refine the displayed uniform data rather than replacing it with an external quotient or selected completion.

The excluded principles have no carrier coordinate. Thus the Real-uniform handoff remains a finite NameCert route through $R, U, S, B, F, L, Q, E, T, C, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}$ completion read exposes RealUniformStructureUp, UniformSpaceUp, CauchyFilterSubbasisUp, generated FilterBaseUp, CauchyFilterUp, locatedness, separated reflection, completion seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: real-uniform-structure, separated-uniform-completion, cauchy-filter-subbasis, uniformspace, real: $\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Real-induced uniformity, uniform-source exposure, finite generated subbasis data, Cauchy-filter exactness, located tests, separated reflection, seal stability, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}$ packet is generated from RealUniformStructureUp data, UniformSpaceUp entourages, CauchyFilterSubbasisUp finite-generation rows, a generated FilterBaseUp row, a CauchyFilterUp row, locatedness tests, separated-reflection data, a completion-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7572.1, Definition 7572.2, Definition 7572.3, and Definition 7572.4 seed the finite located separated-reflection route for Real-uniform completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffLocatedUniformReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, hidden filter equality, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of BEDC.Derived.HausdorffLocatedUniformReflectionUp with checked carrier admission, Real-uniform exposure, subbasis generation, Cauchy-filter exactness, locatedness, separated-reflection non-escape, seal stability, transport, replay, provenance, and classifier faithfulness.

Chapter 7573

A Concrete Naming Certificate for $\mathsf{AffineFilterHull}^{\uparrow}$

Following Kelley on finite filter bases and the standard affine-hull closure discipline for generated neighbourhood systems, the $\mathsf{AffineFilterHull}^{\uparrow}$ packet records a finite hull route from displayed affine generators to a filter-base handoff. It concretizes the vision-routing surface of chapter 9236 through a traditional filter-hull object grounded in the coverage index of chapter 7381, the coverage ledger of chapter 7484, and the filter-base interface of chapter 1601. The packet names the generated affine-filter boundary only; it does not assert convex completeness, select limits, quotient filters, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7573.1 (Affine filter hull carrier)**.** *An* $\mathsf{AffineFilterHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (V, G, M, I, B, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite affine-generator row, listing the displayed generator tokens available to the hull.*
2. *G is the finite generator-family row, recording which finite affine combinations are admitted as local hull requests.*
3. *M is the membership-event row, exposing the* BHist *surface attached to each admitted hull request.*
4. *I is the inclusion ledger from generator tokens and finite affine combinations into generated filter requests.*
5. *B is the generated* $\mathsf{FilterBase}^{\uparrow}$ *row exposed to downstream consumers.*
6. *R is the routing row linking this concrete object back to the vision coverage index and ledger surfaces.*
7. *H records componentwise* hsame *transport for generators, finite affine combinations, membership events, inclusion ledgers, filter-base rows, routing rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from affine-generator requests through finite hull membership and into the generated filter-base handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{AffineFilterHull}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{VisionCoverageIndex}^{\uparrow}$, $\mathsf{VisionCoverageLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{AffineFilterHull}^{\uparrow}}$ *row.*
The classifier reads only $V, G, M, I, B, R, H, C, P, N$. It has no coordinate for an infinite affine span, quotient filters, selected limits, countable choice, host equality of hulls, or ambient completeness.

Theorem 7573.2 (Affine filter hull NameCert obligations)**.** *For every accepted* $\mathsf{AffineFilterHull}^{\uparrow}$ *carrier $A = (V, G, M, I, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AffineFilterHull}^{\uparrow}}$ *surface consists exactly of carrier admission, finite affine-generator exposure through V, finite affine-combination admission through G, membership-event exposure through M, inclusion-ledger coverage through I, generated* $\mathsf{FilterBase}^{\uparrow}$ *handoff through B, vision-routing compatibility through R, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 7573.1. The rows V, G, M, I are the complete finite affine-hull generation surface, and B is the only generated filter-base row.

The routing row is separate from the generated base. It records that the hull object realizes a vision coverage route, but it does not create a second carrier or a hidden completion object.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since the carrier has no row for an infinite affine span, quotient filters, selected limits, countable choice, host equality of hulls, or ambient completeness, those principles cannot appear as local NameCert obligations. □

Theorem 7573.3 (Affine filter hull generated-base route)**.** *Every downstream read of an accepted* $\mathsf{AffineFilterHull}^{\uparrow}$ *carrier reaches the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B only after finite affine generators V, admitted finite combinations G, membership events M, and the inclusion ledger I have been displayed. The route exports no quotient filters, selected limits, countable choice, host equality of generated hulls, infinite affine span, or ambient completeness.*

Proof. Begin with the carrier rows. The only source data for the hull are the finite generator row V and finite-combination row G.

The membership-event and inclusion-ledger rows M, I mediate all access from those generators to the generated base. Thus B is available only through the displayed finite hull route.

Apply the NameCert obligations of Definition 7573.2. They force the generated $\mathsf{FilterBase}^{\uparrow}$ handoff and the routing row to remain inside the same finite packet.

The excluded coordinates are absent from A. Hence a downstream consumer cannot read quotient filters, selected limits, countable choice, host equality, infinite spans, or ambient completeness from this packet. □

Theorem 7573.4 (Affine filter hull vision routing)**.** *The* $\mathsf{AffineFilterHull}^{\uparrow}$ *packet realizes a concrete routing object for the vision coverage surface: a read through R must point to the existing vision coverage index and coverage ledger while the generated* $\mathsf{FilterBase}^{\uparrow}$ *handoff remains the row B of the same carrier. No vision-side marker supplies an independent theorem, hidden carrier, quotient filter, selected limit, or countable-choice schedule.*

Proof. Inspect the routing row R in Definition 7573.1. It names the already displayed coverage index and ledger surfaces as provenance for the concrete hull route.

The generated filter-base handoff is still B. Rows V, G, M, I produce it, and rows H, C, P, N transport, replay, source, and name the same object.

Therefore the vision marker is a pointer to this concrete carrier rather than a separate mathematical payload. Since the carrier has no quotient, selected-limit, choice, hidden-carrier, or ambient-completeness coordinate, the routing read cannot export any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{AffineFilterHull}^{\uparrow}$ read exposes finite affine generators, finite affine-combination rows, membership events, an inclusion ledger, a generated FilterBaseUp row, a vision-routing row, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: filterbase, vision-coverage-index, vision-coverage-ledger: $\mathsf{AffineFilterHull}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite affine generators, generated membership events, inclusion-ledger coverage, a generated filter-base handoff, vision-routing compatibility, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AffineFilterHull}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AffineFilterHull}^{\uparrow}$ packet is generated from finite affine-generator rows, finite affine-combination rows, membership-event rows, an inclusion ledger, a generated FilterBaseUp row, a vision-routing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AffineFilterHull}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7573.1, Definition 7573.2, Definition 7573.3, and Definition 7573.4 seed the finite affine-filter hull route from displayed generators to generated filter-base handoff and vision routing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AffineFilterHull}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite affine spans, quotient filters, selected limits, countable choice, host equality of hulls, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of BEDC.Derived.AffineFilterHullUp with checked carrier admission, finite affine generation, membership-event exposure, inclusion-ledger coverage, FilterBaseUp handoff, vision-routing compatibility, transport, replay, provenance, and classifier faithfulness.

Chapter 7574

A Concrete Naming Certificate for $\mathsf{CauchyApproximationSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationSpace}^{\uparrow}$ packet records the finite approximation interface by which a regular rational source answers completion-facing Real reads through named finite windows. It sits after the StreamName finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the located metric comparison surface of chapter 3685. The chapter packages the four-face route as BEDC-native finite data rather than as informal Cauchy approximation prose.

Definition 7574.1 (Cauchy approximation space carrier)**.** *A* $\mathsf{CauchyApproximationSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the observed unary request spine.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row family attached to those windows, including common-scale comparison and radius rows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that identifies the regular rational source consumed by the window and tolerance data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the finite window, dyadic tolerance, and regular readback rows have been displayed.*
5. *H stores componentwise* hsame *transport for W, D, R, E, C, P, N and for every local comparison row opened by the packet.*
6. *C stores displayed* Cont *replay from StreamName requests through dyadic tolerance checks and RegSeqRat readback into the Real seal row.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedMetricSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected real representative, countable choice, trichotomy, host metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7574.2 (Cauchy approximation space NameCert obligations)**.** *For an accepted* CauchyApproximationSpac *carrier* $A = (W, D, R, E, H, C, P, N)$*, the local* NameCert *surface consists exactly of finite* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-row exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Every completion-facing consumer of the carrier must read these rows before it can use a Real-facing approximation result.*

Proof. Project the accepted carrier through Definition 7574.1. The rows W, D, R, E are the only analytic coordinates, while H, C, P, N supply transport, replay, provenance, and local naming.

The finite window obligation is read from the StreamName chapter chapter 126. The dyadic tolerance obligation is read from chapter 291, and the regular rational readback is read from chapter 301.

The Real-facing endpoint is sealed only after those rows are present. The local NameCert therefore names the displayed four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the located metric surface of chapter 3685 available only as a dependency of the displayed comparison rows. □

Theorem 7574.3 (Cauchy approximation space window stability)**.** *Let two accepted* $\mathsf{CauchyApproximationSpace}^{\uparrow}$ *carriers agree under componentwise* hsame *transport on their finite* $\mathsf{StreamName}^{\uparrow}$ *window rows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows. Then every retained request reads the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the same* $\mathsf{Real}^{\uparrow}$ *seal handoff, the same* Cont *replay, the same* Pkg *provenance, and the same local* NameCert *row. The comparison remains a finite BEDC window comparison rather than quotient stream equality or host real equality.*

Proof. Start with the common transported window and tolerance rows. The StreamName finite observation surface keeps every retained request inside the same displayed finite section, and the DyadicRatCore tolerance ledger keeps the comparison rows on the same finite scale.

Read the RegSeqRat row through that shared window. Since R is a carrier coordinate and not a hidden tail selector, the readback available to a retained request is the same regular rational source row on both carriers.

Finally replay the support rows. The hsame row identifies the displayed coordinates, Cont repeats the same route, and P, N package and name that route without adding quotient stream equality, a selected limit, host real equality, or an ambient completeness principle. □

Theorem 7574.4 (Cauchy approximation space Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyApproximationSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

carried by W, D, R, E, H, C, P, N*. The consumer may use the located metric dependency of chapter 3685 only through the displayed dyadic comparison rows, and it cannot introduce a selected completion point, quotient Cauchy equality, host metric completeness, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7574.2 to expose the carrier rows. A Real-facing consumer sees the finite StreamName window, dyadic tolerance ledger, and regular rational readback before it reaches the seal row E.

Apply Definition 7574.3 to retained requests. Enlarging or replaying a request preserves the old window, tolerance ledger, readback, support rows, provenance, and local name.

The located metric dependency is therefore already internal to the displayed dyadic comparison rows. The final handoff exports only the finite approximation route into $\mathsf{Real}^{\uparrow}$, so no selected completion point, quotient equality, host metric completeness, countable choice, trichotomy, or ambient Real completeness can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyApproximationSpace}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7574.1, Definition 7574.2, Definition 7574.3, and Definition 7574.4 seed the finite L10 approximation route from StreamName windows and dyadic tolerance rows to RegSeqRat readback and RealUp seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completion points, host metric completeness, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window stability, dyadic tolerance exactness, RegSeqRat readback, RealUp seal nonescape, and located metric dependency coverage for BEDC.Derived.CauchyApproximationSpaceUp.

Chapter 7575

A Concrete Naming Certificate for $\mathsf{FreeTopologicalGroup}^{\uparrow}$

Following Markov and Graev on free topological groups, the $\mathsf{FreeTopologicalGroup}^{\uparrow}$ packet records the finite BEDC surface by which a displayed topological source generates reduced-word group rows equipped with Graev uniformity data. It sits over the open-row discipline of chapter 88, the entourage discipline of chapter 339, the continuous operation ledger of chapter 204, the metric witness route of chapter 57, and the algebraic operation surface of chapter 41. The certificate names finite reduced-word rows, Graev entourage rows, continuity rows for multiplication and inverse, componentwise transport, replay, provenance, and local naming; it does not import a host quotient, a universal-property oracle, or an ambient free-object primitive.

Definition 7575.1 (Free topological group carrier)**.** *A* $\mathsf{FreeTopologicalGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, T, U, G, W, E, M, I, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-object row whose points are read only through the displayed* $\mathsf{Topology}^{\uparrow}$ *surface.*
2. *T is the topology row from chapter 88, exposing the open-neighbourhood requests used by the source.*
3. *U is the uniformity row from chapter 339, exposing entourage requests used to compare reduced words.*
4. *G is the algebraic group row from chapter 41, naming multiplication, unit, inverse, and reduced-word comparison rows.*
5. *W is the reduced-word ledger, listing finite alternating source letters and formal inverse tags together with the local cancellation record.*
6. *E is the Graev entourage ledger, recording the finite word-level entourages generated from source entourages and their compatibility with reduction.*
7. *M is the optional* $\mathsf{Met}^{\uparrow}$ *handoff row, read through chapter 57 when the source uniformity is supplied by a displayed metric.*
8. *I is the* $\mathsf{TopGroup}^{\uparrow}$ *continuity row from chapter 204, requiring multiplication and inverse to be continuous for the topology generated by E.*

9. *H stores componentwise* hsame *transport for $X, T, U, G, W, E, M, I, C, P, N$.*
10. *C stores displayed* Cont *replay from source rows through reduced words, Graev uniformity, group operations, metric handoff, and topological-group continuity.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeTopologicalGroup}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FreeTopologicalGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, U, G, W, E, M, I, H, C, P, N$. It has no coordinate for host quotient equality of words, selected representatives, ambient universal arrows, untracked topologies, external completion, Lean verification, or bridge checking.

Theorem 7575.2 (Free topological group NameCert obligations)**.** *For every accepted* $\mathsf{FreeTopologicalGroup}^{\uparrow}$ *carrier*

$$F = (X, T, U, G, W, E, M, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FreeTopologicalGroup}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source topology, source uniformity, algebraic group row, reduced-word ledger, Graev entourage ledger, optional metric handoff, topological-group continuity row, componentwise transport, replay, provenance, and local naming. A consumer may read a free topological group only after these rows have been displayed together.*

Proof. Project the accepted carrier through Definition 7575.1. The source data are X, T, U, the algebraic data are G, W, and the Graev uniform route is stored in E with optional metric support in M.

The topological-group obligation is read through I. This forces the multiplication and inverse rows named by G to be consumed together with the topology generated by the Graev entourage ledger, using the sibling topological-group surface rather than an ambient operation theorem.

Finally H, C, P, N exhaust transport, replay, provenance, and local naming. Since the classifier has no further coordinate, the packet cannot export host quotient equality of words, selected representatives, ambient universal arrows, untracked topologies, external completion, Lean verification, or bridge checking. □

Theorem 7575.3 (Free topological group Graev uniform route)**.** *Every Graev-uniform read exported by an accepted* $\mathsf{FreeTopologicalGroup}^{\uparrow}$ *packet factors through the displayed source route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{UniformSpace}^{\uparrow},\ \mathsf{Group}^{\uparrow} \longrightarrow \mathsf{TopGroup}^{\uparrow}$$

and, when metric data are present, through the optional $\mathsf{Met}^{\uparrow}$ *row before the word-level entourage is consumed. The reduced-word ledger W and the Graev entourage ledger E remain visible at the point of use, so no consumer can replace the packet by a host quotient group, an external free topological group, or an untracked universal map.*

Proof. Begin with $F = (X, T, U, G, W, E, M, I, H, C, P, N)$. The topology row T and uniformity row U expose the source neighbourhood and entourage requests before any generated word can be compared.

The reduced-word and group rows are W and G. They provide the finite word ledger, cancellation record, unit, multiplication, and inverse rows that the Graev route may inspect.

The Graev-uniform route is E together with the continuity row I. If a metric source is available, M records the displayed metric handoff before the word-level entourage is used; otherwise no metric coordinate is exported. Transport and replay through H, C, P, N preserve exactly these rows, so the consumer receives only the finite NameCert route. □

Lean4 (statement) `BEDC.Derived.FreeTopologicalGroupUp`

Closure status for $\mathsf{NameCert}_{\mathsf{FreeTopologicalGroup}^{\uparrow}}$.

Constructive story: A $\mathsf{FreeTopologicalGroup}^{\uparrow}$ packet is generated from a displayed topological source row, uniformity row, algebraic group row, reduced-word ledger, Graev entourage ledger, optional metric handoff, topological-group continuity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeTopologicalGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7575.1, Definition 7575.2, and Definition 7575.3 seed the finite route from displayed topology and uniformity rows through reduced-word group rows into topological-group continuity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeTopologicalGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host quotient group, an ambient universal property, selected word representatives, external completion, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reduced-word exactness, Graev entourage compatibility, multiplication and inverse continuity, metric handoff, transport, provenance, and local naming obligations for BEDC.Derived.FreeTopologicalGroupUp.

Chapter 7576

A Concrete Naming Certificate for $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$

Following Arzela and Ascoli, the $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ packet records the finite-net extraction surface for an equicontinuous family on a compact metric source as BEDC NameCert data. It sits after the compact and metric source rows of chapter 60, chapter 57, and chapter 129, the pointwise continuous-map rows of chapter 59 and chapter 130, the finite-net ledger of chapter 127, the real-name boundary of chapter 38, and the shared family-modulus route of chapter 4076. The packet names the finite extraction boundary only; it is not an open-cover compactness theorem, a selected subsequence, a quotient function-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7576.1 (Arzela–Ascoli finite-net carrier)**.** *An* $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, M, C, T, E, R, W, H, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through the* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *certificates.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used by finite-net comparisons on the source.*
3. *C is the finite* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *family row whose values are inspected only through displayed pointwise windows.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger of centers, membership rows, distance witnesses, and refinement data.*
5. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *shared tolerance-radius row for the same family and source.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *value-readback row used for displayed pointwise distances and oscillation bounds.*
7. *W is the finite extraction window row, recording which center values and tolerance budgets are exposed to a completion or compactness-facing consumer.*
8. *H records componentwise* hsame *transport for the compact, metric, continuity, total-bounded, equicontinuity, Real, window, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, C, T, E, R, W, H, P, N$. It has no coordinate for open-cover compactness, host function-space topology, quotient equality of maps, selected subsequences, countable choice, or ambient Real completeness.

Theorem 7576.2 (Arzela–Ascoli finite-net NameCert obligations)**.** *For every accepted* $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ *carrier* $A = (K, M, C, T, E, R, W, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact source, metric row, continuous-family row, total-bounded finite-net ledger, equicontinuity modulus row, Real value-readback row, extraction window, componentwise transport, provenance, and local naming; finite-net compatibility requiring T to be read before W; equicontinuity compatibility requiring E to be read before the extracted window is consumed; Real readback exactness requiring R to remain attached to the displayed pointwise values; and nonescape excluding open-cover compactness, selected subsequences, quotient function equality, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 7576.1. The accepted packet displays exactly $K, M, C, T, E, R, W, H, P, N$, so every local obligation must be read from those rows.

The finite-net compatibility row is T. It supplies centers, membership rows, distance witnesses, and refinement data before the extraction window W can expose any finite sample of the family.

The equicontinuity compatibility row is E. Its shared tolerance-radius handoff is read after the continuous-family row C and before W, so the extracted window is controlled by the same family modulus rather than by memberwise radius choices.

Real readback exactness is the role of R. The value comparisons consumed by W remain terminal real-name reads for displayed pointwise data, not quotient equality of functions or a host supremum.

Finally H, P, N transport, source, and name only the displayed rows. Since no row stores open-cover compactness, selected subsequences, quotient function equality, countable choice, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7576.3 (Arzela–Ascoli finite-net extraction)**.** *Every completion-facing or compactness-facing read exported by an accepted* $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ *carrier*

$$A = (K, M, C, T, E, R, W, H, P, N)$$

factors through the finite route

$$K, M, C, T, E, R \longmapsto W \longmapsto H, P, N.$$

Thus the exported object is a finite-net extraction window over compact, metric, continuous, totally bounded, Real, and equicontinuity rows. It does not export a selected subsequence, a function-space compactness theorem, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7576.2. They force any accepted read to display the compact source, metric row, continuous-family row, finite-net ledger, shared equicontinuity row, and Real value-readback row before the extraction window is consumed.

Use the cited sibling chapters for the route data. The compact and metric rows supply the source boundary, the continuity rows supply displayed value reads, the total-bounded row supplies finite centers, the Real row supplies terminal value seals, and the equicontinuity row supplies the shared tolerance-radius handoff.

The extraction window W can therefore expose only finite center values and displayed tolerance budgets inherited from K, M, C, T, E, R. The structural rows H, P, N then transport, source, and name that same window without adding another analytic coordinate.

The exclusions follow from the carrier inventory. Since selected subsequences, function-space compactness, quotient equality of maps, countable choice, and ambient Real completeness do not occur in $K, M, C, T, E, R, W, H, P, N$, no completion-facing or compactness-facing read can export them from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7576.1, Definition 7576.2, and Definition 7576.3 seed the finite-net extraction surface for equicontinuous families over compact metric source, total-bounded net, Real readback, transport, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequences, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net compatibility, equicontinuity compatibility, Real readback exactness, extraction-window routing, transport, provenance, and local naming for BEDC.Derived.ArzelaAscoliFiniteNetUp

Constructive story: An $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ packet is generated from compact and metric source rows, continuous-family rows, a total-bounded finite-net ledger, an Equicontinuity shared modulus row, Real value readback, an extraction window, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Chapter 7577

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ packet records the finite calibration row that ties a displayed precision request to the dyadic tolerance, stream window, regular-rational readback, and terminal Real seal used by a Bishop-style Cauchy name. It sits after the dyadic core of chapter 291, the finite stream-name surface of chapter 126, the regular rational sequence route of chapter 301, and the real-name seal of chapter 38. It also sits beside the finite tail-window handoff of chapter 7562. The packet is calibration data for inspected regular Cauchy names; it is not $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, trichotomy, selected representatives, or countable choice.

Definition 7577.1 (Bishop Cauchy modulus calibration carrier)**.** *A* $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying the requested precision, its dyadic radius, and the finite comparison budget used by the name.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected at that requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected window into regular rational observations.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after D, S, R have been displayed.*
5. *M is the calibration row tying the precision request, tolerance row, window row, readback row, and Real seal into one Bishop Cauchy modulus route.*
6. *H records componentwise* hsame *transport for D, S, R, E, M, C, P, N.*
7. *C records displayed* Cont *replay of the route*

$$D \longmapsto S \longmapsto R \longmapsto M \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalibration}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, E, M, H, C, P, N$. It has no coordinate for quotient real equality, trichotomy, countable choice, selected representatives, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7577.2 (Bishop Cauchy modulus calibration NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ *carrier*

$$K = (D, S, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalibration}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{StreamName}^{\uparrow}$ *window row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{Real}^{\uparrow}$ *seal row E, calibration row M, transport row H, replay row C, provenance row P, and local naming row N;*
2. *dyadic-window calibration, requiring D to determine the precision budget under which S is inspected;*
3. *regular readback calibration, requiring R to consume the same S and D before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is visible;*
4. hsame *transport and* Cont *replay, requiring H and C to preserve the displayed route D, S, R, M, E without inserting any hidden selector;*
5. Pkg *provenance and local* $\mathsf{NameCert}$ *naming, requiring P and N to name the same finite packet and to refuse quotient equality, trichotomy, countable choice, selected representatives, host real equality, and ambient Real completeness.*

No calibration obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7577.1. The packet displays exactly the dyadic tolerance, stream window, regular rational readback, terminal Real seal, calibration, transport, replay, provenance, and local naming rows.

The calibration rows are D, S, R, M, E. A precision request first exposes the dyadic budget D, reads the finite window S, and then checks the regular rational observations through R before the calibration row M can hand the request to the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The structural rows are H, C, P, N. They transport, replay, source, and name the same finite route, so a consumer cannot introduce a quotient equality row, trichotomy row, countable-choice schedule, selected representative, host equality, or ambient completeness coordinate through the local certificate. □

Theorem 7577.3 (Bishop Cauchy modulus calibration seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ *carrier reaches the seal row only through the finite chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

as calibrated by M and preserved by H, C, P, N. The route exports a calibrated regular-Cauchy observation packet; it exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient equality of real names, trichotomy, selected representative, host real equality, or countable-choice schedule.*

Proof. Apply Definition 7577.2. The obligation rows force a real-facing read to expose the dyadic tolerance row, the stream window, and the regular rational readback before the calibration row can present the terminal seal.

The terminal seal row is E. Since E is downstream of the same D, S, R, M route and the transport, replay, provenance, and local naming rows preserve only that route, every accepted consumer reaches $\mathsf{Real}^{\uparrow}$ through the displayed finite packet. □

Falsifiable prediction: A $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the DyadicRatCore tolerance row, StreamName window row, RegSeqRat readback row, calibration row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier.

Independence witness: dyadicratcore, streamname, regseqrat, real, realcauchywindow: no listed sibling is bijective to $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ because this packet binds the precision tolerance, inspected stream window, regular rational readback, terminal Real seal, calibration row, transport, replay, provenance, and local NameCert row into one finite Bishop Cauchy route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalibration}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyModulusCalibration}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName window row, RegSeqRat readback row, RealUp seal row, calibration row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalibration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7577.1, Definition 7577.2, and Definition 7577.3 seed the finite Bishop Cauchy modulus calibration route from dyadic precision requests to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalibration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient equality of real names, trichotomy, selected representatives, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-window calibration, regular readback calibration, RealUp seal nonescape, hsame transport, Cont replay, Pkg provenance, and local naming for BEDC.Derived.BishopCauchyModulusCali

Chapter 7578

A Concrete Naming Certificate for $\mathsf{HilbertCubeCompactness}^{\uparrow}$

Following Kelley and Engelking on the Hilbert cube and compact metric products, the $\mathsf{HilbertCubeCompactness}^{\uparrow}$ packet records compactness through finite coordinate-window nets over $\mathsf{HilbertCube}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$. It consumes the finite Hilbert-cube carrier of chapter 1995, the stream and regular-rational readback routes of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, the terminal real seal of chapter 38, and the compact-metric row of chapter 129. The packet names finite coordinate-window net data; it does not claim open-cover compactness, a choice-selected subnet, host product completeness, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7578.1 (Hilbert cube compactness carrier)**.** *A* $\mathsf{HilbertCubeCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (H, W, R, D, E, K, B, T, P, N)$$

with the following displayed rows:

1. *H is the accepted* $\mathsf{HilbertCube}^{\uparrow}$ *source row, read through the finite coordinate-window carrier of chapter 1995.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *coordinate-window schedule naming exactly the coordinates inspected by the current net request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those inspected coordinate windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic mesh radii and comparison cells used by the finite net.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, attaching each displayed coordinate readback to a real interval boundary.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-metric support row for the inspected product window.*
7. *B is the finite coordinate-window net ledger, listing the displayed centers, mesh radii, and containment rows for the inspected window.*
8. *T records componentwise* hsame *transport and displayed* Cont *replay for H, W, R, D, E, K, B.*
9. *P is the* Pkg *provenance over* $\mathsf{HilbertCubeCompactness}^{\uparrow}$, $\mathsf{HilbertCube}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, W, R, D, E, K, B, T, P, N$. It has no coordinate for open-cover compactness, host product completeness, a selected subnet, quotient stream equality, countable choice, Lean verification, or bridge checking.

Theorem 7578.2 (Hilbert cube compactness NameCert obligations)**.** *For every accepted* $\mathsf{HilbertCubeCompactness}^{\uparrow}$ *carrier*

$$C = (H, W, R, D, E, K, B, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, Hilbert-cube source exposure, finite coordinate-window scheduling, regular-rational readback, dyadic mesh tolerance exposure, real interval sealing, compact-metric finite-window support, finite net containment, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7578.1. The accepted tuple exposes the Hilbert-cube source row, stream schedule, regular-rational readback, dyadic tolerance ledger, real seal row, compact-metric support row, finite net ledger, transport and replay row, provenance row, and local name.

The finite coordinate-window obligations are read from H, W, R, D, E. The source row H supplies only the displayed Hilbert-cube carrier, W names the inspected coordinates, R reads their rational data, D fixes the dyadic mesh, and E seals the inspected coordinates as real interval data.

The compact-metric support and net obligations are read from K and B. The row K records the finite metric support for the inspected product window, while B lists the centers, radii, and containment checks used by the net consumer.

Finally, T, P, N exhaust transport, replay, provenance, and local naming. Since the carrier has no additional coordinate, the certificate cannot store open-cover compactness, host product completeness, a selected subnet, quotient stream equality, countable choice, Lean verification, or bridge checking. □

Theorem 7578.3 (Hilbert cube compactness finite coordinate net)**.** *Every accepted* $\mathsf{HilbertCubeCompactness}^{\uparrow}$ *consumer read factors through the finite coordinate-window net route*

$$\mathsf{HilbertCube}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow}.$$

The exported net is therefore a finite ledger over inspected coordinates, dyadic mesh radii, sealed real intervals, compact-metric support, and displayed containment rows, not an open-cover compactness proof or a choice-selected countable product argument.

Proof. Start with an accepted carrier $C = (H, W, R, D, E, K, B, T, P, N)$. The Hilbert-cube source row H can be read only through the finite coordinate-window surface already displayed by chapter 1995.

The consumer then reads W, R, D, E in order. These rows name the inspected coordinates, recover regular rational data, choose the dyadic mesh budget, and seal the inspected coordinates into real interval rows.

The net itself is exported only after K and B are visible. The compact-metric support row fixes the finite metric window, and the net ledger lists the displayed centers, radii, and containment checks for that same window.

Transport, replay, provenance, and local naming preserve this route componentwise through T, P, N. No row remains from which the consumer could obtain an arbitrary open-cover argument, a selected subnet, host product completeness, quotient stream equality, or countable choice. □

Theorem 7578.4 (Hilbert cube compactness real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{HilbertCubeCompactness}^{\uparrow}$ *packet is bounded by the displayed finite route through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh tolerances,* $\mathsf{Real}^{\uparrow}$ *interval seals,* $\mathsf{CompactMetric}^{\uparrow}$ *finite support, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}$ *row. The packet exports no ambient real completeness principle and no equality of host streams.*

Proof. Project the carrier. A real-facing consumer reaches the terminal real rows only after the finite stream schedule, regular rational readback, and dyadic tolerance ledger have all been displayed.

The compactness read does not alter that boundary. The compact-metric support row and finite net ledger consume the sealed interval data already present in E and do not add a separate real-completion or host-product coordinate.

The structural rows T, P, N preserve the same finite route. Therefore every real-facing read remains inside the displayed coordinate-window seal and cannot be reread as ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host product completeness, open-cover compactness, or a selected subnet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7578.1, Definition 7578.2, Definition 7578.3, and Definition 7578.4 seed the finite coordinate-window net route from $\mathsf{HilbertCube}^{\uparrow}$ through stream windows, regular-rational readback, dyadic mesh tolerances, real interval seals, compact-metric support, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertCubeCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host product completeness, quotient stream equality, selected subnets, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coordinate-window scheduling, dyadic mesh exposure, real interval sealing, compact-metric support, finite net containment, transport, replay, provenance, and local naming for BEDC.Derived.HilbertCubeCompactnessUp.

Constructive story: A $\mathsf{HilbertCubeCompactness}^{\uparrow}$ packet is generated from a $\mathsf{HilbertCube}^{\uparrow}$ source row, finite $\mathsf{StreamName}^{\uparrow}$ coordinate windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, $\mathsf{DyadicRatCore}^{\uparrow}$ mesh ledgers, $\mathsf{Real}^{\uparrow}$ interval seals, $\mathsf{CompactMetric}^{\uparrow}$ finite support rows, a finite net containment ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7579

A Concrete Naming Certificate for $\mathsf{CompletionModulusLifting}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completions and Kelley on completion and uniform structures, the $\mathsf{CompletionModulusLifting}^{\uparrow}$ packet records the finite route by which a displayed completion modulus is lifted into StreamName, RegSeqRat, DyadicRatCore, and Real seal rows before any Real or Completion consumer reads it. It sits after the modulus surface of chapter 292, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, the Bishop completion-modulus carrier of chapter 1469, and the threshold-selection surface of chapter 7210. The packet names only displayed finite lifting data; it does not select limits, quotient streams, countable-choice schedules, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7579.1 (Completion modulus lifting carrier)**.** *A* $\mathsf{CompletionModulusLifting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the completion modulus row, read through* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *and* $\mathsf{CauchyModulus}^{\uparrow}$*, assigning a finite threshold to each displayed precision request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened at the thresholds named by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic request, threshold comparison, and radius accounting used by M and R.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus, window, readback, tolerance, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through completion-modulus thresholding, finite windows, rational readback, dyadic tolerance accounting, and the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusLifting}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, BishopCompletionModulus $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompletionModulusLifting}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, S, R, D, E, H, C, P, N$. The carrier has no coordinate for a selected completion point, quotient stream representative, countable-choice schedule, trichotomy, host real equality, or ambient Real completeness.

Theorem 7579.2 (Completion modulus lifting NameCert obligations)**.** *For every accepted* CompletionModulusLiftin *carrier $L = (M, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusLifting}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed completion modulus, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows; threshold coverage requiring every precision request to be discharged by M before S is opened; tolerance exactness requiring D before R is read; Real seal ordering requiring E to be downstream of M, S, R, D; and nonescape for quotient streams, selected limits, countable choice, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7579.1. The only available rows are $M, S, R, D, E, H, C, P, N$, so carrier admission exposes exactly the completion modulus, finite windows, rational readback, dyadic tolerance ledger, Real seal, and structural data.

Threshold coverage is the first lifting step. A precision request is read by the completion-modulus row M, and the finite StreamName windows S are opened only at the threshold displayed by that row.

Tolerance exactness and readback are the next step. The DyadicRatCore ledger D records the requested dyadic radius and the threshold comparison before the RegSeqRat row R is read from the opened windows.

Finally the Real seal E is downstream of M, S, R, D. The structural rows H, C, P, N transport, replay, source, and name the same route, and no carrier coordinate stores quotient streams, selected limits, countable-choice schedules, trichotomy, host equality, or ambient completeness. □

Theorem 7579.3 (Completion modulus lifting Real seal handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{CompletionModulusLifting}^{\uparrow}$ *carrier must factor through the displayed route*

$$\mathsf{BishopCompletionModulus}^{\uparrow},\ \mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{C}$$

In particular, the terminal $\mathsf{Real}^{\uparrow}$ *seal is available only after the completion modulus row, finite windows, regular rational readback, dyadic tolerance ledger, componentwise transport, replay, provenance, and local naming rows have been displayed.*

Proof. Start with the NameCert obligations of Definition 7579.2. They force the consumer to read M before S, to read D before R, and to read M, S, R, D before the terminal seal E.

Use the neighbouring Bishop completion-modulus route of chapter 1469 and the threshold-selection surface of chapter 7210. Those routes supply the finite threshold discipline used by M without turning it into a host-selected limit.

The remaining rows are structural. Componentwise hsame transport and displayed Cont replay preserve the same lifting path, while Pkg provenance and the local NameCert row name it for downstream Real and Completion consumers. Since no listed row contains quotient streams, selected limits, countable choice, trichotomy, host equality, or ambient completeness, the Real seal handoff exports none of those objects. □

Falsifiable prediction: An accepted $\mathsf{CompletionModulusLifting}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the completion modulus row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, bishopcompletionmodulus, bishop-completion-modulus-selection: no listed sibling is bijective to $\mathsf{CompletionModulusLifting}^{\uparrow}$ because this packet jointly carries a completion modulus row, finite windows, regular rational readback, dyadic tolerance accounting, terminal Real seal, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusLifting}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionModulusLifting}^{\uparrow}$ packet is generated from a completion modulus row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusLifting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7579.1, Definition 7579.2, and Definition 7579.3 seed the finite completion-modulus lifting route over completion modulus, stream-window, regular-rational, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusLifting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, threshold coverage, tolerance exactness, RegSeqRat readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusLiftingUp.

Chapter 7580

A Concrete Naming Certificate for $\mathsf{CauchyRealsCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealsCompletion}^{\uparrow}$ packet records the traditional Cauchy-real completion interface as finite BEDC rows rather than as a quotient carrier or a selected representative. It reads the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the displayed monad-multiplication boundary of chapter 1829. The packet is a finite completion-facing interface for the open Real and Completion phase; it does not assert quotient classes, selected representatives, countable choice, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7580.1 (Cauchy reals completion carrier)**.** *A* $\mathsf{CauchyRealsCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, D, M, E, J, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the regular Cauchy input.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same scheduled windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every admitted precision request.*
4. *M is the modulus-readback row that sends each unary precision request to a displayed finite window and dyadic tolerance entry.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after S, Q, D, M have been displayed.*
6. *J is the* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *handoff row used when a completion-of-completion consumer asks to flatten its request before reading the Real-facing seal.*
7. *H records componentwise* hsame *transport for the schedule, readback, tolerance, modulus, seal, join, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

   $$S \longmapsto Q \longmapsto D \longmapsto M \longmapsto E$$

   and of the join route through J when a nested completion request is present.
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealsCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealsCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, Q, D, M, E, J, H, C, P, N$. It has no coordinate for quotient classes, selected representatives, countable choice, host equality, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$*, Lean verification, or bridge checking.*

Theorem 7580.2 (Cauchy reals completion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealsCompletion}^{\uparrow}$ *carrier*

$$R = (S, Q, D, M, E, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealsCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, finite-window scheduling through S, regular-rational readback through Q, dyadic tolerance accounting through D, modulus readback through M, Real-seal handoff through E, nested completion flattening through J, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No local obligation is stored in quotient classes, selected representatives, countable-choice schedules, host equality, ambient completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 7580.1. The tuple displays exactly the StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, modulus row, Real seal, completion-multiplication handoff, transport, replay, provenance, and local name.

The scheduling, readback, tolerance, and modulus obligations are read from S, Q, D, M. The Real-facing obligation is read from E only after those rows have been displayed.

The nested completion obligation is read through J, so a completion-of-completion consumer must use the same finite route already named by chapter 1829. Transport, replay, provenance, and local naming are exhausted by H, C, P, N.

Since the carrier has no further coordinate, the excluded quotient, representative, choice, host-equality, and completeness objects cannot store a local obligation. □

Theorem 7580.3 (Cauchy reals completion modulus readback)**.** *Let $R = (S, Q, D, M, E, J, H, C, P, N)$ be an accepted* $\mathsf{CauchyRealsCompletion}^{\uparrow}$ *carrier. Every unary precision request is answered by the modulus row M as one displayed finite route through a* $\mathsf{StreamName}^{\uparrow}$ *window in S, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in Q, and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance entry in D before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is read.*

The readback exports no quotient class, selected representative, countable-choice selector, host equality, ambient completeness datum, or completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with a precision request at the accepted carrier. The only public request-answering row is M, and it is tied in the carrier to the finite schedule S.

Read the regular rational observation through Q and the corresponding dyadic tolerance entry through D. These rows are the only displayed data that can justify the requested accuracy before the seal row is reached.

The terminal Real-facing read is E. The structural rows H, C, P, N transport, replay, source, and name the same route, while J is used only when a nested completion request must first be flattened.

Thus the precision request is answered by the stated finite route. The excluded quotient, representative, choice, host-equality, and completeness data have no row from which they could be read. □

Theorem 7580.4 (Cauchy reals completion classifier stability)**.** *If two accepted* $\mathsf{CauchyRealsCompletion}^{\uparrow}$ *carriers agree componentwise under displayed* hsame *transport on their* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, modulus-readback row,* $\mathsf{Real}^{\uparrow}$ *seal,* CauchyCompletionMultipli

handoff, replay, provenance, and local naming rows, then they have the same public completion classifier read.

The classifier is stable only over those finite rows and cannot be altered by quotient classes, selected representatives, countable choice, host equality, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the componentwise transport row H for the two accepted packets. It compares exactly the rows named in Definition 7580.1.

Replay the transported rows through C. The finite schedule, regular readback, dyadic tolerance ledger, modulus answer, Real seal, and nested completion handoff remain on the same displayed route.

The provenance row P and local name N do not add classifier data; they only source and name the accepted route. Since the carrier exposes no quotient, representative, choice, host-equality, or completeness coordinate, no such object can change the classifier read. □

Theorem 7580.5 (Cauchy reals completion Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyRealsCompletion}^{\uparrow}$ *carrier reaches the seal row* E *only after reading the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, modulus row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. If the consumer begins with a nested completion request, it must first pass through the* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *handoff row* J*.*

The seal is a terminal BEDC row for the displayed route; it is not a proof of ambient completeness and it is not a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier. The Real-facing seal coordinate is E, and the rows upstream of it are the schedule S, readback Q, tolerance ledger D, and modulus row M.

If the request is nested, the handoff row J supplies the completion-multiplication route before the same Real-facing rows are consumed. This keeps the completion-of-completion case inside the displayed BEDC packet.

Transport, replay, provenance, and local naming are carried by H, C, P, N. These rows preserve and source the same route without adding a host Real-completeness premise.

Therefore every Real-facing consumer reaches E only through the finite surface listed in the statement, and the terminal seal cannot be read as ambient completeness or as completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealsCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealsCompletion}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a modulus-readback row, a terminal RealUp seal, a CauchyCompletionMultiplication handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealsCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7580.1, Definition 7580.2, Definition 7580.3, Definition 7580.4, and Definition 7580.5 seed the finite completion-facing interface for regular Cauchy names through StreamName, RegSeqRat, DyadicRatCore, RealUp, CauchyCompletionMultiplication, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealsCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient classes, selected representatives, countable choice, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus readback, classifier stability, Real-seal nonescape, nested completion handoff, and local NameCert registration for BEDC.Derived.CauchyRealsCompletionUp.

Chapter 7581

A Concrete Naming Certificate for $\mathsf{DependentChoiceBoundary}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DependentChoiceBoundary}^{\uparrow}$ packet records the constructive boundary between finite-prefix diagonal selection and dependent-choice assumptions in the Real and Completion corridor. It sits after the countable-choice refusal packet of chapter 1872, the LLPO boundary of chapter 1978, the diagonal regular-Cauchy route of chapter 3782, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite prefix dependency rows; it is not a dependent-choice functional, not a branch selector through all stages, not quotient stream equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7581.1 (Dependent-choice boundary carrier)**.** *A* $\mathsf{DependentChoiceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, L, G, W, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *Q is the inherited* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *refusal row, keeping countable witness selection outside the carrier.*
2. *L is the* $\mathsf{LLPOBoundary}^{\uparrow}$ *row, recording that binary separation principles are not supplied by the packet.*
3. *G is the* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *source row for the inspected diagonal family.*
4. *W is the finite prefix-dependency window, listing the already inspected source indices and their dependency edges.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite regular-rational data exposed by W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite prefix window and regular-rational readback are displayed.*
7. *F is the nonescape ledger refusing a dependent-choice chain, a hidden global selector, quotient stream equality, selected limits, and ambient completion.*
8. *H records componentwise* hsame *transport for the displayed rows.*
9. *C records displayed* Cont *replay of the same finite prefix route.*
10. *P is the* Pkg *provenance over* $\mathsf{DependentChoiceBoundary}^{\uparrow}$, $\mathsf{CountableChoiceBoundary}^{\uparrow}$, $\mathsf{LLPOBoundary}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{DependentChoiceBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, L, G, W, R, E, F, H, C, P, N$. It has no coordinate for dependent choice, a global branch selector, quotient stream equality, selected limits, trichotomy, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7581.2 (Dependent-choice boundary finite-prefix nonescape)**.** *Every accepted* DependentChoiceBoundary *carrier exposes its diagonal data through the finite route*

$$\mathsf{CountableChoiceBoundary}^{\uparrow},\quad \mathsf{LLPOBoundary}^{\uparrow},\quad \mathsf{RegularCauchyDiagonalization}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Any consumer that asks for an infinite dependent-choice chain, a hidden global selector, quotient stream equality, a selected real limit, or ambient completion must stop at the nonescape ledger F.

Proof. Project the carrier $D = (Q, L, G, W, R, E, F, H, C, P, N)$. The rows Q and L place the request behind the countable-choice and LLPO refusal boundaries before the diagonal source row is read.

Read the finite prefix data next. The diagonal source G supplies only the inspected regular-Cauchy family, while W records the finite dependency window and R reads back the regular rational data visible in that window.

The terminal real-name row E is therefore downstream of finite prefix evidence. The nonescape row F has no field for a dependent-choice chain, a global branch selector, quotient stream equality, selected limits, or ambient completion, and the structural rows H, C, P, N preserve this same finite route. □

Theorem 7581.3 (Dependent-choice boundary Real-completion refusal)**.** *Every Real or Completion consumer reading an accepted* $\mathsf{DependentChoiceBoundary}^{\uparrow}$ *packet must first display the countable-choice boundary row, the LLPO boundary row, the diagonal regular-Cauchy source, the finite prefix-dependency window, the regular rational readback, the terminal real-name seal, and the structural rows H, C, P, N. The consumer may use those rows as a finite diagonal selection certificate, but it receives no dependent-choice assumption and no completed Real object beyond the displayed* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Start from the displayed carrier. The Real-facing consumer reaches $\mathsf{Real}^{\uparrow}$ only through G, W, R, E: the diagonal row gives the source family, the prefix window lists the finite dependencies, the regular-rational row reads the inspected data, and the real seal records the terminal boundary.

Now apply the refusal rows. The inherited $\mathsf{CountableChoiceBoundary}^{\uparrow}$ row blocks countable selection, while the $\mathsf{LLPOBoundary}^{\uparrow}$ row prevents binary separation from being supplied as a hidden principle.

Finally read F, H, C, P, N. These rows refuse dependent choice and replay the same finite packet under transport, continuation, provenance, and local naming. Thus the Real-completion handoff is a finite-prefix route through the cited sibling chapters, not a dependent-choice construction or an ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DependentChoiceBoundary}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DependentChoiceBoundary}^{\uparrow}}, \mathsf{seed})$
Scope closed: Definition 7581.1, Definition 7581.2, and Definition 7581.3 seed the finite-prefix diagonal boundary over countable-choice refusal, LLPO refusal, regular-Cauchy diagonalization, regular-rational readback, Real sealing, transport, replay, provenance, and local naming.
Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DependentChoiceBoundary}^{\uparrow}}, \mathsf{unformalized})$
Bridge status: none

Not claimed: This chapter does not close dependent choice, countable choice, LLPO, selected limits, quotient stream equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix nonescape, Real-completion refusal, transport, replay, provenance, and local naming for BEDC.Derived.DependentChoiceBoundaryUp.

Constructive story: A DependentChoiceBoundary$^{\uparrow}$ packet is generated from countable-choice and LLPO refusal rows, a regular-Cauchy diagonal source, a finite prefix-dependency window, regular-rational readback, a Real seal row, nonescape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7582

A Concrete Naming Certificate for RealityConstrainedSelectorBoundary$^{\uparrow}$

The RealityConstrainedSelectorBoundary$^{\uparrow}$ packet realizes the selector-facing boundary of Definition 9413.89 as finite NameCert data. It sits after the finite observation interface of chapter 1007, the model-audit classifier of chapter 1124, the open-fit carrier of chapter 1132, the certified physical truth surface of chapter 1130, the failure-surface packet of chapter 1163, and the final-synthesis ledger of chapter 1178. The packet names a selector boundary only through finite source records, classifiers, physical-fit rows, failure surfaces, and gap ledgers; it is not a final model, a truth oracle, an unbounded observation selector, or a host reality predicate.

Definition 7582.1 (Reality-constrained selector boundary carrier)**.** *A* RealityConstrainedSelectorBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, K, F, O, G, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-record row, read through finite observation records and certified trace fragments.*
2. *K is the classifier row, exposing the reality-facing audit classifier and its admissible comparison surface.*
3. *F is the physical-fit row, recording finite agreement with the displayed open-fit and certified-truth surfaces.*
4. *O is the failure-surface row, listing defeats, domain exits, classifier mismatches, and continuation limits.*
5. *G is the gap-ledger row, recording omitted selector supply, hidden-source pressure, failed fit claims, and cannot-claim residues.*
6. *B is the boundary-decision row, admitting a selector read only when A, K, F, O, G are all present in the same packet.*
7. *H records componentwise* hsame *transport for the displayed source, classifier, fit, failure, gap, and boundary rows.*
8. *C records displayed* Cont *replay of the same selector-boundary route.*
9. *P is the* Pkg *provenance over* RealityConstrainedSelectorBoundary$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the cited sibling chapters.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, K, F, O, G, B, H, C, P, N$. It has no coordinate for final truth, unbounded observation, model–reality identity, a hidden selector, private source access, Lean verification, or bridge checking.

Theorem 7582.2 (Reality-constrained selector boundary NameCert obligations)**.** *For every accepted* $\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ *carrier*

$$S = (A, K, F, O, G, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}}$ *surface consists exactly of source-record admission, classifier exposure, physical-fit exposure, failure-surface exposure, gap-ledger exposure, boundary-decision admission, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the displayed carrier. The source row A, classifier row K, physical-fit row F, failure row O, gap ledger G, and boundary-decision row B are the complete selector-facing data.

Read the structural rows next. Transport through H preserves each displayed component, replay through C keeps the consumer inside the same finite route, provenance through P records the cited sibling surfaces, and N locally names the packet.

No additional coordinate is available. Therefore a selector read that omits a source record, classifier, fit row, failure surface, gap ledger, or boundary decision is a projection of the packet rather than an accepted $\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ carrier. □

Theorem 7582.3 (Reality-constrained selector boundary failure-surface nonescape)**.** *Every accepted selector read through* $\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ *must expose the failure-surface row O and the gap-ledger row G before the boundary-decision row B is consumed. If a consumer reaches a selector conclusion while hiding a defeat, domain exit, classifier mismatch, omitted selector supply, or cannot-claim residue, the read is rejected by the local classifier.*

Proof. Start from $S = (A, K, F, O, G, B, H, C, P, N)$. The boundary-decision row B is defined only over the five displayed source, classifier, fit, failure, and gap rows.

The failure-surface row O records defeats, exits, mismatches, and continuation limits. The gap-ledger row G records omitted supply and cannot-claim residue. Hiding either row changes the displayed selector boundary before B is read.

Finally read H, C, P, N. These rows transport, replay, source, and name the same packet; they do not add hidden evidence or erase listed failures. Hence a selector conclusion is accepted only while O and G remain visible. □

Theorem 7582.4 (Reality-constrained selector boundary sibling separation)**.** *The* $\mathsf{RealityConstrainedSelectorBound}$ *carrier is not bijective to the finite observation interface, reality-constrained model audit, open-fit packet, certified physical truth packet, failure-surface packet, final-synthesis ledger, or model-selection boundary named above. Each sibling supplies one surface, but none alone carries the joint source-record, classifier, physical-fit, failure-surface, gap-ledger, boundary-decision, transport, replay, provenance, and local naming rows of S.*

Proof. Compare the row roles. The finite observation interface supplies source windows, the model audit supplies classifier comparison, the open-fit and certified-truth chapters supply fit and truth-facing rows, the failure-surface chapter supplies defeats, the final-synthesis ledger supplies synthesis accounting, and the model-selection boundary supplies comparative selection discipline.

The selector boundary requires all of these roles at once, together with G and B. The gap-ledger row records selector pressure that none of the sibling packets alone carries, and the boundary-decision row admits or refuses the selector read only after the other rows are jointly displayed.

Thus no listed sibling is bijective to $\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$. Replacing the packet by any one sibling loses at least one of the source, classifier, fit, failure, gap, decision, transport, replay, provenance, or local naming rows. □

Falsifiable prediction: Any accepted selector-boundary read that omits either the failure-surface row O or the gap-ledger row G is rejected before the boundary-decision row B is consumed.

Independence witness: finite-observation-interface, reality-constrained-model-audit, reality-constrained-open-fit, realityconstrainedfailuresurface, model-selection-boundary: none is bijective because RealityConstrainedSel jointly carries source-record, classifier, physical-fit, failure-surface, gap-ledger, boundary-decision, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ TasteGate instances in `lean4/BEDC/Derived/RealityConstrain` The nontriviality instance must separate an accepted selector-boundary packet from a packet missing O or G, and the field-faithful instance must preserve $A, K, F, O, G, B, H, C, P, N$ as distinct rows before any checked carrier-alignment theorem can serve as a TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7582.1, Definition 7582.2, Definition 7582.3, Definition 7582.4, the falsifiable prediction, and the independence witness seed the finite selector boundary over source records, classifiers, physical-fit rows, failure surfaces, gap ledgers, boundary decisions, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close final truth, model–reality identity, unbounded observation, hidden selectors, private source access, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-record exposure, classifier exposure, physical-fit exposure, failure-surface nonescape, gap-ledger exactness, boundary-decision admission, transport, replay, provenance, and local naming for BEDC.Derived.RealityConstrainedSelectorBoundaryUp.

Constructive story: A $\mathsf{RealityConstrainedSelectorBoundary}^{\uparrow}$ packet is generated from source-record, classifier, physical-fit, failure-surface, gap-ledger, boundary-decision, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 7583

A Concrete Naming Certificate for $\mathsf{KolmogorovWidth}^{\uparrow}$

Following Kolmogorov and Tikhomirov on n-widths, the $\mathsf{KolmogorovWidth}^{\uparrow}$ packet records finite-dimensional approximation width as displayed NameCert data. It sits after the norm and normed-space rows of chapter 89 and chapter 1950, the compact metric source of chapter 129, the metric comparison row of chapter 57, the vector-space and linear-map rows of chapter 47 and chapter 48, and the terminal real-name boundary of chapter 38. The packet names a finite approximation ledger; it does not import a host infimum, choose an optimal subspace, quotient a normed space, invoke compactness by open covers, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7583.1 (Kolmogorov width carrier)**.** *A* $\mathsf{KolmogorovWidth}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, A, E, M, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the source compact family row, read through* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and the surrounding* $\mathsf{VecSpace}^{\uparrow}$ *source.*
2. *A is the finite-dimensional approximant packet, displaying the chosen dimension bound, basis-window ledger,* $\mathsf{LinMap}^{\uparrow}$ *projection or inclusion row, and admissible finite subspace data.*
3. *E is the* $\mathsf{Real}^{\uparrow}$ *error-bound row recording the displayed radius by which A approximates every inspected member of K.*
4. *M is the norm or metric comparison row, naming the* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *faces used to read distances from source elements to the finite-dimensional approximant.*
5. *Q is the classifier row, comparing accepted width packets only through $K, A, E, M, L, H, C, P, N$.*
6. *L is the width ledger, recording inspected precision requests, candidate subspace witnesses, failure rows for too-large errors, and the refusal of a host infimum or selected best subspace.*
7. *H stores componentwise* hsame *transport for the compact source, approximant, error-bound, norm or metric, classifier, ledger, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from compact-family inspection through finite-dimensional approximation rows, norm or metric comparison,* $\mathsf{Real}^{\uparrow}$ *error readback, and refusal boundaries.*
9. *P is the* Pkg *provenance over* $\mathsf{KolmogorovWidth}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{KolmogorovWidth}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host-level infimum over all subspaces, an optimal-subspace selector, quotient norm equality, open-cover compactness, countable choice, or ambient Real completeness.

Theorem 7583.2 (Kolmogorov width NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovWidth}^{\uparrow}$ *carrier* $W = (K, A, E, M, Q, L, H, C, P, N)$*, the local NameCert surface consists exactly of compact-family admission, finite-dimensional approximant admission,* $\mathsf{Real}^{\uparrow}$ *error-bound readback, norm or metric comparison, classifier exactness, width-ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every approximation-width consumer must read these rows before treating the packet as a finite-dimensional approximation certificate.*

Proof. Project the carrier of Definition 7583.1. The analytic rows are K, A, E, M, Q, L, while H, C, P, N provide transport, replay, provenance, and local naming.

Read the sibling sources. The compact metric, metric, norm, vector-space, linear-map, and real-name chapters supply the rows named by K, A, E, M; none supplies a hidden optimal subspace or host infimum.

The classifier and ledger rows then bind the approximation claim to the displayed finite data. Since Q compares only the listed coordinates and L records both candidate witnesses and failure boundaries, every accepted consumer remains inside the stated NameCert surface. □

Theorem 7583.3 (Kolmogorov width finite approximation route)**.** *Every finite approximation, compactness, or completion-facing read exported by an accepted* $\mathsf{KolmogorovWidth}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{Norm}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

carried by $K, A, E, M, Q, L, H, C, P, N$*. The read may expose a displayed finite-dimensional approximant and a displayed error ledger, but it cannot export a host infimum, selected optimal subspace, quotient norm equality, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the NameCert obligations of Definition 7583.2. They force a consumer to display the compact source, finite-dimensional approximant, norm or metric comparison, real error bound, classifier, ledger, transport, replay, provenance, and local name.

The finite-dimensional row A is the only approximant coordinate. It may name a basis-window ledger and linear-map row, but it is not an optimizer over all subspaces.

The width ledger L records requested precision, candidate witnesses, and failure boundaries. Thus the exported read is a finite approximation route over displayed data, with the excluded host infimum, selected optimum, quotient equality, open-cover compactness, countable choice, and ambient completeness objects absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovWidth}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KolmogorovWidth}^{\uparrow}$ packet is generated from a compact metric source family, finite-dimensional approximant data, a RealUp error bound, norm or metric comparison, classifier, width ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovWidth}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7583.1, Definition 7583.2, and Definition 7583.3 seed the finite-dimensional approximation-width route without a host infimum or selected optimal subspace.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovWidth}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host infimum, optimal-subspace selector, quotient norm equality, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximant admission, error-bound readback, classifier exactness, width-ledger exposure, finite approximation routing, and non-escape for BEDC.Derived.KolmogorovWidthUp.

Chapter 7584

A Concrete Naming Certificate for $\mathsf{FiniteApproximationKernel}^{\uparrow}$

Following standard finite approximation theory and the finite relation-certificate discipline of chapter 9416, the $\mathsf{FiniteApproximationKernel}^{\uparrow}$ packet records source, approximant, classifier, error-ledger, failure-boundary, transport, replay, provenance, and local naming rows as a concrete NameCert surface. It sits beside the Cauchy approximation space of chapter 7574, the Arzela–Ascoli finite-net surface of chapter 7576, the compact metric source of chapter 129, the metric and real readback rows of chapter 57 and chapter 38, and the finite-dimensional width packet of chapter 7583. The packet is a finite relation-certificate kernel for approximation reads; it is not a selected limit, universal approximation theorem, host infimum, quotient equality, open-cover compactness, countable choice, or ambient completeness principle.

Definition 7584.1 (Finite approximation kernel carrier)**.** *A* $\mathsf{FiniteApproximationKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, A, K, E, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row naming the observed domain, admissible requests, compact or Cauchy windows, and relation-certificate source boundary.*
2. *A is the finite approximant row naming candidate approximants, finite-net centers, finite-dimensional windows, or Cauchy approximation rows available to the read.*
3. *K is the classifier row, comparing accepted kernels only through S, A, E, B, H, C, P, N.*
4. *E is the error-ledger row recording displayed* $\mathsf{Real}^{\uparrow}$ *bounds, tolerance budgets, residual rows, and inspected failure thresholds.*
5. *B is the failure-boundary row, naming requests that exceed the displayed approximant, hidden-source attempts, classifier mismatches, missing error ledgers, and over-reads into host completeness.*
6. *H records componentwise* hsame *transport for source, approximant, classifier, error-ledger, failure-boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source inspection through approximant selection, classifier comparison, error-ledger readback, and failure-boundary refusal.*

8. *P is the* Pkg *provenance over* FiniteApproximationKernel$^{\uparrow}$, CauchyApproximationSpace$^{\uparrow}$, ArzelaAscoliFiniteNet$^{\uparrow}$, KolmogorovWidth$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteApproximationKernel}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected limits, unrestricted approximation theorems, host infima, quotient equality, open-cover compactness, countable choice, or ambient completion.

Theorem 7584.2 (Finite approximation kernel NameCert obligations)**.** *For every accepted* FiniteApproximationKer *carrier $F = (S, A, K, E, B, H, C, P, N)$, the local NameCert surface consists exactly of finite source admission, finite approximant admission, classifier exactness, error-ledger exposure, failure-boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any finite approximation consumer must display these rows before reading the packet as approximation data.*

Proof. Project the carrier of Definition 7584.1. The source and approximant rows are S, A, the comparison and audit rows are K, E, B, and the structural rows are H, C, P, N.

Read the sibling approximation surfaces. The Cauchy approximation, Arzela–Ascoli finite-net, compact metric, metric, real, and Kolmogorov width chapters provide the finite rows that may populate S, A, E.

The classifier and failure boundary prevent relation-certificate language from becoming object identity or host completeness. Hence a consumer receives exactly the finite approximation kernel rows and none of the excluded selected, quotient, compactness, choice, or completeness endpoints. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteApproximationKernel}^{\uparrow}}$.

Constructive story: A FiniteApproximationKernel$^{\uparrow}$ packet is generated from finite source rows, finite approximant rows, classifier exactness, error-ledger rows, failure-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteApproximationKernel}^{\uparrow}}$, seed)

Scope closed: Definition 7584.1 and Definition 7584.2 seed the finite relation-certificate approximation kernel over source, approximant, classifier, error-ledger, failure-boundary, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteApproximationKernel}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, unrestricted approximation theorems, host infima, quotient equality, open-cover compactness, countable choice, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, error-ledger exposure, failure-boundary exposure, finite approximation consumer routing, and non-escape for BEDC.Derived.FiniteApproximationKernelUp.

Chapter 7585

A Concrete Naming Certificate for $\mathsf{DecidableBar}^{\uparrow}$

Following Brouwer on bars over spreads and Bishop–Bridges on finite binary inspection in constructive analysis, the $\mathsf{DecidableBar}^{\uparrow}$ packet names the finite decidable-bar row used by fan arguments before any real-completion consumer reads it. It sits beside the finite fan surface of chapter 400, the binary prefix carrier of chapter 1475, and the finite observation gateway of chapter 126. The packet records finite prefix windows, decidable bar hits, unary depth ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row. It does not select an infinite branch, assert countable choice, supply open-cover compactness, or close ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7585.1 (Decidable bar carrier)**.** *A* $\mathsf{DecidableBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, S, W, R, D, H, T, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite binary-prefix row inspected by the bar.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row naming the finite prefix window read by the current consumer.*
3. *W is the finite prefix-window ledger, listing the inspected binary prefixes and their parent-prefix inclusions.*
4. *R is the decidable bar-hit row, recording for each displayed prefix whether the bar predicate has fired inside the finite window.*
5. *D is the unary depth ledger bounding all inspected prefixes and bar-hit checks.*
6. *H records componentwise* hsame *transport for C, S, W, R, D, T, P, N.*
7. *T records displayed* Cont *replay of the finite route*

$$C \longmapsto S \longmapsto W \longmapsto R \longmapsto D.$$

8. *P is the* Pkg *provenance over* $\mathsf{DecidableBar}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, S, W, R, D, H, T, P, N$. It has no coordinate for selected infinite branches, countable choice, open-cover compactness, host Cantor-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DecidableBarUp.DecidableBarTasteGate single carrier alignment`

Theorem 7585.2 (Decidable bar NameCert obligations)**.** *For every accepted* $\mathsf{DecidableBar}^{\uparrow}$ *carrier*

$$B = (C, S, W, R, D, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Cantor prefix row, StreamName window row, finite prefix-window ledger, decidable bar-hit row, unary depth ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read these rows as finite bar data, but no obligation is stored outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.DecidableBarUp.DecidableBarCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 7585.1. The accepted tuple exposes exactly the Cantor prefix row, stream observation row, prefix-window ledger, bar-hit row, unary depth ledger, transport row, replay row, provenance row, and local name.

The bar obligations are finite because W, R, D are displayed rows. The window row lists the prefixes under inspection, the hit row gives a decidable answer on that displayed list, and the depth ledger bounds the readback.

The structural rows H, T, P, N preserve that same finite packet. Since the carrier has no further coordinate, the local certificate cannot store a selected infinite branch, a countable-choice schedule, an open-cover compactness witness, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Theorem 7585.3 (Decidable bar stream-window route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer reading an accepted* $\mathsf{DecidableBar}^{\uparrow}$ *packet must first expose the finite route*

$$\mathsf{CantorSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto W \longmapsto R \longmapsto D.$$

Thus the public bar read is a finite prefix-window computation preserved by BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not an infinite branch selection or a host-level choice principle.*

Lean4 ✓ `BEDC.Derived.DecidableBarUp.DecidableBarCarrier stream window route`

Proof. Start with the carrier projection. The stream observation row S is downstream of the Cantor prefix row C, and the prefix-window ledger W is downstream of that displayed stream observation.

The decidable bar-hit row R is read only over the prefixes listed by W. The unary depth ledger D records the bound under which those hits are available, so the route remains finite.

Finally, transport, replay, provenance, and local naming through H, T, P, N preserve the same finite route. They do not add a selector for an infinite binary branch or a countable family of choices. □

Theorem 7585.4 (Decidable bar real-completion boundary)**.** *If a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer cites an accepted* $\mathsf{DecidableBar}^{\uparrow}$ *packet, the citation can use only the finite Cantor prefix row, StreamName window, prefix-window ledger, decidable hit row, unary depth ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}$ *row. The citation cannot be read as selected branch data, countable choice, open-cover compactness, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DecidableBarUp.DecidableBarRealCompletionBoundary`

Proof. Apply Definition 7585.2. It identifies the full public surface of the packet with the finite rows $C, S, W, R, D, H, T, P, N$.

Apply Definition 7585.3. Any real-facing or completion-facing read must pass through the finite StreamName window and the decidable bar-hit ledger before it can cite the packet.

Coordinate exhaustion gives the boundary. No row records a chosen infinite branch, a countable-choice selector, an open-cover compactness proof, quotient equality of real names, or a completeness witness for $\mathsf{Real}^{\uparrow}$. □

Theorem 7585.5 (Decidable bar finite-branch induction)**.** *Every induction over an accepted* $\mathsf{DecidableBar}^{\uparrow}$ *packet is induction over the displayed finite prefix-window ledger* W *and its unary depth ledger* D*. At each stage the consumer exposes only the current Cantor prefix row, StreamName observation row, decidable bar-hit row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.DecidableBarUp.DecidableBarFiniteBranchInduction`

Proof. Induct on the displayed ledger W. The empty window exposes no hidden branch data; it carries only the structural rows already present in the accepted packet.

For a nonempty window, read the head prefix through C and S, inspect its decidable hit row in R, and retain the unary depth bound in D. The induction hypothesis applies to the remaining displayed prefixes because the carrier stores them in the same finite ledger.

Transport and replay through H and T preserve the component currently under inspection, while P and N preserve provenance and local naming. The induction therefore never leaves the finite BEDC packet and cannot produce a selected infinite branch, countable choice, open-cover compactness, or real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7585.1, Definition 7585.2, Definition 7585.3, Definition 7585.4, and Definition 7585.5 seed the finite decidable-bar carrier, NameCert obligations, StreamName window route, real-completion boundary, and finite-branch induction surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecidableBar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, countable choice, Brouwer bar induction, open-cover compactness, quotient equality of real names, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix-window coverage, decidable bar-hit readback, unary depth ledger exposure, stream-window routing, finite-branch induction, transport, replay, provenance, and local naming for BEDC.Derived.DecidableBarUp.

Constructive story: A $\mathsf{DecidableBar}^{\uparrow}$ packet is generated from a finite $\mathsf{CantorSpace}^{\uparrow}$ prefix row, a $\mathsf{StreamName}^{\uparrow}$ observation row, a finite prefix-window ledger, a decidable bar-hit row, a unary depth ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7586

A Concrete Naming Certificate for $\mathsf{CauchyPairLocator}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy data, the $\mathsf{CauchyPairLocator}^{\uparrow}$ packet records the finite locator surface for a paired regular Cauchy read before any Real or Completion consumer may treat the pair as separated, equal, or ready for a downstream handoff. It sits after the dyadic core of chapter 291, the finite stream-name surface of chapter 126, the regular rational sequence route of chapter 301, the terminal real-name seal of chapter 38, and the paired-modulus surface of chapter 7130. The packet is a located pair witness over displayed finite rows; it is not quotient equality, a selected representative pair, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7586.1 (Cauchy pair locator carrier)**.** *A* $\mathsf{CauchyPairLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, S, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *A is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying the regular rational observations read from the first name.*
2. *B is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying the regular rational observations read from the second name.*
3. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite windows inspected from both names before a locator request is answered.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, displaying the dyadic radius, split tolerance, comparison budget, and endpoint accounting used by the pair read.*
5. *R is the* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ *row, recording the finite pair-modulus budget that synchronizes A and B at the tolerance named by D.*
6. *Q is the locator decision and witness row, storing the displayed comparison branch, its finite witness data, and the exact tolerance at which the branch is read.*
7. *H records componentwise* hsame *transport for $A, B, S, D, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto A, B \longmapsto R \longmapsto Q.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyPairLocator}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyPairModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPairLocator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, S, D, R, Q, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable choice, trichotomy, host equality, or ambient Real completeness.

Theorem 7586.2 (Cauchy pair locator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPairLocator}^{\uparrow}$ *carrier*

$$L = (A, B, S, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyPairLocator}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the two regular readback rows, shared stream windows, dyadic tolerance ledger, pair-modulus row, locator row, transport row, replay row, provenance row, and local naming row;*
2. *two-sided regular readback, requiring A and B to be read from the same displayed windows S and the same dyadic budget D before the pair modulus is consumed;*
3. *dyadic locator exactness, requiring Q to name its comparison branch and witness at the exact tolerance displayed by D and synchronized by R;*
4. *Real seal nonescape, requiring every Real-facing or Completion-facing handoff to keep the two readback rows, window row, tolerance ledger, pair modulus, and locator row visible before any seal is read;*
5. *structural stability through H, C, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve the same finite packet.*

No locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7586.1. The accepted packet displays exactly the two regular readback rows, shared StreamName windows, DyadicRatCore tolerance ledger, pair-modulus row, locator row, transport, replay, provenance, and local naming.

The two-sided regular readback is read from A, B, S, D, R. Both regular sequences are inspected through the same finite window and tolerance ledger before the pair modulus row synchronizes them for a locator request.

Dyadic locator exactness is the role of Q. Its branch and witness are tied to the tolerance ledger D and to the synchronized pair-modulus row R, so the locator cannot appeal to an external trichotomy, quotient equality, selected representative, or hidden host comparison.

The Real seal nonescape and structural clauses are carried by H, C, P, N. These rows transport, replay, source, and name the same packet for Real and Completion consumers without adding quotient equality, countable choice, host equality, or ambient completeness. □

Theorem 7586.3 (Cauchy pair locator L10 handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CauchyPairLocator}^{\uparrow}$ *carrier must read the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyPairModulus}^{\uparrow}$ *row, and the locator row before any equality or separation handoff. The handoff exports only the displayed finite locator packet and its structural rows H, C, P, N; it does not claim quotient equality, selected representatives, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 7586.2. The obligation rows force a consumer to expose A and B as regular readbacks over the shared windows S and dyadic ledger D before the pair-modulus row R and locator row Q can be consumed.

The equality or separation handoff is downstream of Q. Since the locator row records only the displayed comparison branch and finite witness at the named tolerance, it cannot supply quotient equality, a selected representative pair, countable choice, trichotomy, or ambient Real completeness.

The rows H, C, P, N preserve the same route. Componentwise transport, displayed replay, provenance, and local naming do not add a second access path to Real or Completion consumers, so every handoff remains confined to the finite locator packet. □

Falsifiable prediction: An accepted $\mathsf{CauchyPairLocator}^{\uparrow}$ consumer that reaches an equality or separation handoff without displaying both RegSeqRat readback rows, the StreamName windows, the DyadicRatCore tolerance ledger, the RegularCauchyPairModulus row, the locator row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-pair-modulus: no listed sibling is bijective to $\mathsf{CauchyPairLocator}^{\uparrow}$ because this packet jointly carries two regular readbacks, shared finite windows, dyadic tolerance accounting, a pair-modulus synchronizer, a locator decision and witness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPairLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyPairLocator}^{\uparrow}$ packet is generated from two RegSeqRat readback rows, shared StreamName windows, a DyadicRatCore tolerance ledger, a RegularCauchyPairModulus row, a locator decision and witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyPairLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7586.1, Definition 7586.2, and Definition 7586.3 seed the finite paired regular Cauchy locator route over readback, window, tolerance, pair-modulus, locator, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyPairLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality, selected representatives, countable choice, trichotomy, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-sided regular readback, dyadic locator exactness, Real seal nonescape, structural stability, provenance, and local naming for BEDC.Derived.CauchyPairLocatorUp.

Chapter 7587

A Concrete Naming Certificate for $\mathsf{ConstructiveRealModulus}^{\uparrow}$

Following Bishop and Bridges on constructive real moduli, the $\mathsf{ConstructiveRealModulus}^{\uparrow}$ packet records the finite modulus surface that a Real and Completion consumer reads before treating a regular rational stream as a completed real name. It sits over the dyadic tolerance rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed modulus schedule only; it does not assert quotient stream equality, select a limit, import countable choice, or close ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7587.1 (Constructive real modulus carrier)**.** *A* $\mathsf{ConstructiveRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the inspected source positions.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii requested by the consumer.*
3. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name inside the displayed window.*
4. R *is the modulus-response row, assigning each requested dyadic radius a displayed tail threshold inside* S*.*
5. A *is the visible* $\mathsf{Real}^{\uparrow}$ *seal available only after* S, D, Q, R *have been displayed.*
6. H *records componentwise* hsame *transport for* S, D, Q, R, A, C, P, N*.*
7. C *records displayed* Cont *replay of the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}.$$

8. P *is the* Pkg *provenance over* $\mathsf{ConstructiveRealModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, D, Q, R, A, H, C, P, N$*. It has no coordinate for quotient equality of real names, a selected limit, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ConstructiveRealModulusUp`

Theorem 7587.2 (Constructive real modulus NameCert obligations)**.** *For every accepted* ConstructiveRealModulus *carrier*

$$M = (S, D, Q, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRealModulus}^{\uparrow}}$ *surface consists exactly of finite StreamName window exposure, DyadicRatCore tolerance readback, RegSeqRat regular-rational readback, modulus-response threshold exposure, visible Real seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside these displayed rows.*

Proof. Project the carrier of Definition 7587.1. The packet exposes exactly the window, tolerance, readback, response, seal, transport, replay, provenance, and naming rows.

The modulus obligations are finite because S, D, Q, R are displayed rows. The window row lists the inspected positions, the dyadic ledger lists the requested tolerances, the regular-rational readback row supplies the inspected values, and the response row records the threshold used by the consumer.

The structural rows H, C, P, N preserve the same packet. Since the carrier has no further coordinate, the local certificate cannot store quotient equality of real names, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7587.3 (Constructive real modulus Real-completion route)**.** *Every Real and Completion consumer reading an accepted* $\mathsf{ConstructiveRealModulus}^{\uparrow}$ *packet must first expose the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the packet supplies a displayed modulus window for the completion exit route through chapter 126, chapter 291, chapter 301, and chapter 38; it does not supply Real completeness, quotient equality, selected limits, or countable choice.

Proof. Start with the carrier projection. The consumer first reads the finite source window S, then the dyadic tolerance ledger D, and then the regular rational readback row Q.

The modulus-response row R is available only over the displayed S, D, Q surface. The Real seal A is therefore downstream of a finite modulus response, not a host completion principle.

Finally H, C, P, N transport, replay, source, and name the same finite route. Because the route is exhausted by $S, D, Q, R, A, H, C, P, N$, it cannot export quotient equality of real names, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealModulus}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveRealModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7587.1, Definition 7587.2, and Definition 7587.3 seed the finite modulus carrier, NameCert obligations, and Real-completion route over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveRealModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real names, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance readback, regular-rational readback, modulus-response threshold exposure, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveRealModulusUp.

Constructive story: A $\mathsf{ConstructiveRealModulus}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, modulus-response thresholds, a visible Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7588

A Concrete Naming Certificate for $\mathsf{BishopIntervalCantor}^{\uparrow}$

Following Bishop and Bridges on nested interval arguments, the $\mathsf{BishopIntervalCantor}^{\uparrow}$ packet records the finite extraction route by which a shrinking nested interval family yields a regular rational name and then a Real-facing seal. It sits beside the dyadic endpoint rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the interval-domain surface of chapter 1543. The packet names the nested-window extraction surface only; it does not assert Heine–Borel compactness, selected limits, quotient Real equality, countable choice, ambient Real completeness, or a host interval theorem.

Definition 7588.1 (Bishop interval Cantor carrier)**.** *A* $\mathsf{BishopIntervalCantor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, N, D, M, R, E, H, C, P, A)$$

with the following displayed rows:

1. *L, U are the lower and upper dyadic endpoint rows, each read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *N is the nested-inclusion ledger, recording that every inspected later window is contained in every earlier displayed interval window.*
3. *D is the diameter-shrink row, giving a dyadic tolerance bound for every inspected precision request.*
4. *M is the located midpoint and refinement ledger, naming the finite midpoint checks and interval-refinement choices already displayed by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *extraction row obtained from the endpoint and midpoint schedules.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after R has displayed the regular rational name.*
7. *H stores componentwise* hsame *transport for endpoint rows, nested inclusion, shrink rows, located midpoint refinements, extraction, Real sealing, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from a precision request through finite endpoint windows, midpoint refinement, regular rational extraction, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalCantor}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{IntervalDomain}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. A is the local $\mathsf{NameCert}_{\mathsf{BishopIntervalCantor}^{\uparrow}}$ *row.*
The carrier has no coordinate for Heine–Borel compactness, an arbitrary interval intersection theorem, a selected limit, quotient Real equality, countable choice, or ambient Real completeness.

Theorem 7588.2 (Bishop interval Cantor nested window)**.** *For every accepted* $\mathsf{BishopIntervalCantor}^{\uparrow}$ *carrier* $I = (L, U, N, D, M, R, E, H, C, P, A)$*, every nested-window read factors through the dyadic endpoint rows* L, U*, the nested-inclusion ledger* N*, the diameter-shrink row* D*, the located midpoint and refinement ledger* M*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local naming* A *before any regular rational extraction can be consumed. The read exports no host interval intersection, selected limit, countable-choice schedule, quotient Real equality, or ambient completeness row.*

Proof. Project the carrier of Definition 7588.1. The endpoint rows are L, U, the nesting and shrink data are N, D, and the located refinement ledger is M.

Read the route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{IntervalDomain}^{\uparrow}$. These rows expose finite endpoint windows and containment evidence, so the consumer receives displayed interval data rather than a host compactness theorem.

The structural rows H, C, P, A keep the read inside the same packet. Since the tuple contains no selected limit, quotient equality, choice schedule, or ambient completeness coordinate, the nested-window read has exactly the displayed finite surface. □

Theorem 7588.3 (Bishop interval Cantor completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{BishopIntervalCantor}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

displayed by $L, U, N, D, M, R, E, H, C, P, A$*. The handoff may expose dyadic endpoints, finite windows, midpoint refinement, a regular rational extraction row, Real sealing, transport, replay, provenance, and local naming; it cannot export Heine–Borel compactness, an arbitrary interval-intersection principle, selected limits, quotient Real equality, countable choice, or ambient Real completeness.*

Proof. Use Definition 7588.2. It forces the completion-facing consumer to display endpoint, nesting, shrink, refinement, transport, replay, provenance, and local naming rows before the extraction row is available.

The extraction row R is the only route to a regular rational name. It is read from the same finite StreamName and dyadic endpoint windows, so the Real-facing row E can only seal that regular rational extraction.

Thus the completion handoff is the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. No excluded compactness, selected-limit, quotient, choice, or ambient-completeness endpoint has a coordinate in the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalCantor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopIntervalCantor}^{\uparrow}$ packet is generated from dyadic endpoint rows, nested-inclusion rows, diameter-shrink rows, located midpoint and refinement rows, RegSeqRat extraction, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopIntervalCantor}^{\uparrow}}$, seed)

Scope closed: Definition 7588.1, Definition 7588.2, and Definition 7588.3 seed the nested-interval extraction route for Real-facing completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopIntervalCantor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, arbitrary interval intersection, selected limits, quotient Real equality, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-inclusion stability, shrink-ledger exactness, midpoint-refinement compatibility, RegSeqRat extraction, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalCantorUp.

Chapter 7589

A Concrete Naming Certificate for $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$

The $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$ packet realizes the Mobius-invariant disk branch of chapter 9463 as finite NameCert data. It sits after the disk-coordinate surface of chapter 377, the Poincare disk metric of chapter 6209, the Schwarz–Pick disk-contraction packet of chapter 3682, the hyperbolic cross-ratio packet of chapter 3948, the hyperbolic metric packet of chapter 3474, and the Poincare disk automorphism packet of chapter 6776. The packet records boundary quadruples, disk transport, cross-ratio readback, metric provenance, refusal rows, and local naming; it does not add a completed boundary quotient, an analytic Mobius group action, a selected geodesic representative, global curvature, or ambient complex-field completeness.

Definition 7589.1 (Mobius disk cross-ratio carrier)**.** *A* $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (B, D, T, Q, M, S, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the ordered boundary-quadruple row, naming four inspected boundary slots and their admissibility evidence.*
2. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row through which the boundary data is read inside a finite Poincare disk window.*
3. *T is the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *transport row, recording the displayed Mobius disk transport and its inverse-facing ledger.*
4. *Q is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *readback row for the transported quadruple.*
5. *M is the* $\mathsf{PoincareDiskMetric}^{\uparrow}/\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-provenance row.*
6. *S is the* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *nonexpansion guard used by downstream disk maps.*
7. *O is the boundary-refusal row, recording over-reads into completed boundary quotients, selected ideal points, host analytic group actions, and global curvature claims.*
8. *H stores componentwise* hsame *transport for boundary, disk, transport, cross-ratio, metric, contraction, refusal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from boundary quadruple inspection through disk transport, metric provenance, cross-ratio readback, and refusal.*

10. *P is the* Pkg *provenance over* MobiusDiskCrossRatio$^{\uparrow}$, UnitDisk$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, SchwarzPickDiskContractio HyperbolicCrossRatio$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskAutomorphism$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{MobiusDiskCrossRatio}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed boundary quotient, selected ideal representative, ambient analytic Mobius group, global curvature theorem, unrestricted Schwarz–Pick theorem, or complex-field completion.

Theorem 7589.2 (Mobius disk cross-ratio invariance)**.** *For every accepted* MobiusDiskCrossRatio$^{\uparrow}$ *carrier $X = (B, D, T, Q, M, S, O, H, C, P, N)$, an invariance read must display the same boundary quadruple row B, disk-coordinate row D, Poincare automorphism transport row T, cross-ratio readback row Q, metric provenance M, Schwarz–Pick guard S, transport H, replay C, provenance P, and local name N. The read is invariant only as this finite transported packet; it exports no host Mobius group action, quotient boundary equality, selected geodesic representative, global curvature theorem, or ambient analytic completion.*

Proof. Project the carrier of Definition 7589.1. The boundary, disk, transport, readback, metric, and contraction rows are B, D, T, Q, M, S.

Read the Poincare disk automorphism row T through the same displayed disk data D. The cross-ratio row Q is then read after transport, with metric provenance supplied by M and nonexpansion compatibility guarded by S.

The structural rows H, C, P, N preserve the displayed route. Since the carrier contains no host group action, quotient boundary equality, selected geodesic representative, global curvature theorem, or analytic completion coordinate, the invariance claim is exactly the finite transported NameCert surface. □

Theorem 7589.3 (Mobius disk cross-ratio boundary refusal)**.** *Every boundary-facing read of an accepted* MobiusDiskCrossRatio$^{\uparrow}$ *carrier must pass through $B, D, T, Q, M, S, O, H, C, P, N$ before exporting cross-ratio data. The refusal row O blocks completed boundary quotients, selected ideal points, unrestricted analytic Mobius actions, global hyperbolic geometry, and ambient complex-field completeness from being read as rows of the packet.*

Proof. Use Definition 7589.2. It already forces the read through the boundary quadruple, disk-coordinate, transport, cross-ratio, metric, contraction, transport, replay, provenance, and local naming rows.

Now inspect the refusal row O. It is attached to the same packet and names exactly the boundary over-reads that are not represented by $B, D, T, Q, M, S, H, C, P, N$.

Therefore a boundary consumer receives only the finite cross-ratio packet. Completed boundary quotients, selected ideal points, unrestricted analytic actions, global geometry, and ambient complex-field completeness remain outside the accepted NameCert surface. □

Falsifiable prediction: Every accepted MobiusDiskCrossRatio$^{\uparrow}$ read factors through boundary quadruple rows, UnitDisk disk-coordinate rows, PoincareDiskAutomorphism transport, HyperbolicCrossRatio readback, PoincareDiskMetric and HyperbolicMetric provenance, SchwarzPick-DiskContraction guards, boundary refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: poincare-disk-metric, schwarz-pick-disk-contraction, hyperbolic-cross-ratio, hyperbolic-metric, poincare-disk-automorphism: MobiusDiskCrossRatio$^{\uparrow}$ is not bijective to

any listed sibling because it jointly carries boundary quadruple rows, disk transport, cross-ratio readback, metric provenance, contraction guard, boundary refusal, transport, replay, provenance, and local NameCert data.

Future Lean TasteGate obligations for $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$ must certify Nontrivial and FieldFaithful instances at `lean4/BEDC/Derived/MobiusDiskCrossRatioUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{MobiusDiskCrossRatio}^{\uparrow}}$.

Constructive story: A $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$ packet is generated from boundary quadruple rows, UnitDisk disk-coordinate rows, PoincareDiskAutomorphism transport, HyperbolicCrossRatio readback, PoincareDiskMetric and HyperbolicMetric provenance, SchwarzPickDiskContraction guards, boundary refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MobiusDiskCrossRatio}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7589.1, Definition 7589.2, and Definition 7589.3 seed the finite Mobius disk cross-ratio route for the hyperbolic phase geometry vision.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MobiusDiskCrossRatio}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary quotients, selected ideal representatives, unrestricted analytic Mobius actions, global curvature, ambient complex-field completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-quadruple admission, disk transport, cross-ratio readback, metric provenance, contraction guarding, boundary refusal, Nontrivial and FieldFaithful TasteGate obligations, transport, replay, provenance, and local naming for BEDC.Derived.MobiusDiskCrossRatioUp.

Chapter 7590

A Concrete Naming Certificate for $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$

Following Frostman and Falconer's finite-cover formulation, the $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ packet records the finite growth-measure witness that links Hausdorff cover budgets to measure-facing readback without assuming an ambient regular Borel measure. It sits after the finite content row of chapter 3765, the measure-facing ledger of chapter 5808, the gauge corridor of chapter 4119, and the metric and dyadic substrate of chapter 57 and chapter 291. The packet names a displayed finite support, ball-growth budget, cover compatibility row, and terminal finite-measure seal; it is not a completed Hausdorff measure construction, a choice-selected mass distribution, countable regularity theorem, quotient measure equality, or ambient compactness principle.

Definition 7590.1 (Frostman finite-measure carrier)**.** *A* $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, A, S, R, G, U, L, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance observations are sealed through finite* $\mathsf{Real}^{\uparrow}$ *readback.*
2. *A is the inspected source or subset row, exposed only by finite membership probes.*
3. *S is the finite support row listing the displayed support probes on which the witness measure is read.*
4. *R is the rational radius and dyadic tolerance row inherited from* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *G is the Frostman growth-budget row, recording the displayed exponent and coefficient used in each ball estimate.*
6. *U is the finite ball-cover row, pairing each listed ball with the support probes it meets.*
7. *L is the finite-measure ledger row, sealing the displayed masses and total mass through the measure-facing* $\mathsf{Real}^{\uparrow}$ *route.*
8. *E is the cover-growth exactness row, asserting that every admitted ball read uses the same support, radius, growth, and finite-measure ledgers.*
9. *H records componentwise* hsame *transport for source, support, radius, growth, cover, measure, exactness, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a cover request through radius selection, growth comparison, finite-measure readback, exactness, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$, $\mathsf{HausdorffContent}^{\uparrow}$, $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{HausdorffDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FrostmanFiniteMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, S, R, G, U, L, E, H, C, P, N$. It has no coordinate for an ambient regular Borel measure, countable cover schedule, selected minimizing cover, quotient measure equality, host compactness, or completed Hausdorff measure existence theorem.

Theorem 7590.2 (Frostman finite-measure NameCert obligations)**.** *For every accepted* $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ *carrier*

$$F = (M, A, S, R, G, U, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrostmanFiniteMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, inspected subset, finite support probes, radius and dyadic tolerance row, growth-budget row, finite ball-cover row, finite-measure ledger, cover-growth exactness row, componentwise transport, displayed replay, provenance, and local naming. No obligation is stored outside $M, A, S, R, G, U, L, E, H, C, P, N$.*

Proof. Project the carrier of Definition 7590.1. Rows M and A give the metric source and inspected subset, while S lists the only support probes available to the witness.

The cover-growth data are displayed before any measure-facing read. The radius row R, growth row G, and ball-cover row U determine which finite balls are being tested, and L records the finite mass and total-mass readback for the same support probes.

Exactness, transport, and replay add no hidden analytic object. Row E binds the cover and growth comparison to the measure ledger, while H, C, P, N preserve, replay, source, and name that same finite packet.

Because the carrier has no ambient regular Borel measure, countable cover schedule, selected minimizing cover, quotient measure equality, host compactness, or completed Hausdorff measure theorem, those objects cannot become local NameCert obligations. □

Theorem 7590.3 (Frostman finite-measure cover-growth route)**.** *Every Hausdorff-cover consumer of an accepted* $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{HausdorffContent}^{\uparrow}, \mathsf{HausdorffMeasure}^{\uparrow}, \mathsf{HausdorffDimension}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto S, R, G, U, L, E$$

together with H, C, P, N. The route exports a finite growth-measure witness for displayed balls and covers. It does not export a completed Hausdorff measure, exact Hausdorff dimension value, countable regularity theorem, selected mass distribution, quotient measure equality, or ambient compactness principle.

Proof. Read the Hausdorff corridor through the sibling rows first. The content, measure-facing, and dimension packets supply only finite cover budgets, finite measure ledgers, and gauge data, and the metric and dyadic rows determine the displayed balls.

The Frostman witness then lives in S, R, G, U, L, E. The support row S, radius row R, growth row G, ball-cover row U, finite-measure ledger L, and exactness row E jointly state how each displayed ball receives a growth-bounded finite mass read.

Finally H, C, P, N transport, replay, source, and name the same route for downstream consumers. A consumer that asks for completed measure existence, selected mass distributions,

countable regularity, quotient measure equality, or ambient compactness is outside the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ read is refuted if it exports a growth-measure witness without displaying $\mathsf{HausdorffContent}^{\uparrow}$, $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{HausdorffDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, finite support probes, radius and growth budgets, finite ball-cover rows, finite-measure ledger, exactness, componentwise hsame transport, Cont replay, provenance, and local NameCert rows.

Independence witness: hausdorff-content, hausdorff-measure, hausdorff-dimension, metric, dyadicratcore: none is bijective to $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ because this packet alone binds finite support probes, radius budgets, growth estimates, ball-cover rows, finite-measure readback, and exactness into one finite witness surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FrostmanFiniteMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{FrostmanFiniteMeasure}^{\uparrow}$ packet is generated from HausdorffContent, HausdorffMeasure, HausdorffDimension, Metric, DyadicRatCore, finite support probes, radius and growth budgets, finite ball-cover rows, finite-measure ledger, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrostmanFiniteMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7590.1, Definition 7590.2, and Definition 7590.3 seed the finite growth-measure witness corridor for Hausdorff cover consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrostmanFiniteMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Hausdorff measure existence, exact Hausdorff dimension values, countable cover schedules, selected mass distributions, quotient measure equality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support admission, radius and growth budget readback, finite ball-cover exposure, finite-measure sealing, exactness, transport, replay, provenance, and local naming for BEDC.Derived.FrostmanFiniteMeasureUp.

Chapter 7591

A Concrete Naming Certificate for $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$ packet records the finite criterion by which displayed Cauchy data becomes completion-facing $\mathsf{Real}^{\uparrow}$ data only after windows, dyadic tolerances, regular rational readback, and an explicit modulus row have all been exposed. It sits after the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a Bishop-style completeness criterion for the L10 Real and Completion route; it is not quotient stream equality, a selected limit, countable choice over tail indices, trichotomy, host real equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7591.1 (Bishop Cauchy completeness criterion carrier)**.** *A* $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming every observation window and precision request used by the Cauchy criterion.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording requested dyadic radii, split tolerances, slack cells, and comparison budgets for those windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from the displayed windows and dyadic tolerances.*
4. *M is the explicit Cauchy modulus row, recording a finite threshold schedule for the criterion before any completion-facing consumer reads R as convergent data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, D, R, M have been displayed.*
6. *H records componentwise* hsame *transport for the window, tolerance, readback, modulus, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S \longmapsto D \longmapsto R \longmapsto M \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $S, D, R, M, E, H, C, P, N$. It has no coordinate for quotienting Cauchy streams, choosing a limiting representative, invoking countable choice, importing trichotomy, using host real equality, or asserting ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7591.2 (Bishop Cauchy completeness criterion obligations)**.** *For every accepted* BishopCauchyCompleten *carrier $K = (S, D, R, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission, exposing the StreamName finite windows, DyadicRatCore tolerances, RegSeqRat regular readback, explicit Cauchy modulus, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *criterion-to-modulus exposure, requiring the Cauchy criterion to display M before any downstream consumer treats R as completion-facing data;*
3. *modulus-to-RegSeqRat readback, requiring each read of M to be tied back to the regular rational observations in R and to the finite windows and dyadic tolerances in S, D;*
4. *terminal Real seal nonescape, requiring E to be downstream of S, D, R, M and to export no quotient stream equality, selected limit, countable choice, trichotomy, host equality, or ambient completeness;*
5. *transport, replay, and provenance stability, requiring H, C, P, N to preserve only the displayed criterion route and the local naming row.*

No Bishop Cauchy completeness criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7591.1. The accepted packet displays exactly the window row, dyadic tolerance row, regular readback row, explicit modulus row, terminal seal, and the structural rows.

Criterion-to-modulus exposure is the role of M. A completion-facing read cannot treat the regular rational row as complete before the threshold schedule has been displayed as finite data.

The modulus-to-readback clause ties M back to R and then to S, D. Thus the criterion is read through finite StreamName windows and DyadicRatCore tolerances rather than through an unbounded host sequence predicate.

The terminal seal row E is downstream of the four analytic rows. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host equality, or ambient completeness, none of those objects can be exported by consuming the seal.

Finally H, C, P, N transport, replay, source, and name the same route. They add no independent completion principle, so the listed rows exhaust the local NameCert surface. □

Theorem 7591.3 (Bishop Cauchy completeness Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto M \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N.$$

The handoff exposes only finite windows, dyadic tolerances, regular rational readback, an explicit Cauchy modulus, componentwise transport, displayed replay, package provenance, and local naming.

Proof. Start with an accepted carrier $K = (S, D, R, M, E, H, C, P, N)$. The only observation rows available to a completion-facing consumer are the StreamName windows S and the DyadicRatCore tolerance ledger D.

The regular readback row R is attached to those rows, and the explicit modulus row M is displayed before the terminal seal can be read. This orders the criterion as finite Cauchy data rather than as quotient equality or a selected limit.

The $\mathsf{Real}^{\uparrow}$ seal row E is reached only after S, D, R, M. The structural rows preserve that route, so every Real or Completion consumer factors through the displayed handoff and receives no ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7591.1, Definition 7591.2, and Definition 7591.3 seed the finite Bishop Cauchy completeness criterion through StreamName windows, DyadicRatCore tolerances, RegSeqRat regular readback, an explicit Cauchy modulus, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exposure, modulus-to-RegSeqRat readback, Real seal nonescape, transport stability, replay stability, and provenance stability for BEDC.Derived.BishopCauchyComple

Constructive story: A $\mathsf{BishopCauchyCompletenessCriterion}^{\uparrow}$ packet is generated from StreamName finite windows, DyadicRatCore tolerances, RegSeqRat regular readback, an explicit Cauchy modulus, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Chapter 7592

A Concrete Naming Certificate for $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$

The $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ packet realizes the fixed-boundary shadow branch of chapter 9463 as finite NameCert data. It sits after the boundary-fixed transport route of chapter 6267, the boundary-shadow carrier of chapter 3813, the metric readback of chapter 3474, the Busemann boundary row of chapter 3732, and the observer-expansion packet of chapter 4167. The packet names a finite shadow window whose boundary far-end is held fixed by displayed transport; it is not an interior-center fixed point, selected ideal endpoint, quotient boundary equality, physical observer motion, or completed hyperbolic compactification.

Definition 7592.1 (Fixed-boundary hyperbolic shadow carrier)**.** *A* $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, B, M, U, V, W, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *row preserving the selected boundary commitment along the displayed disk route.*
2. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row naming the finite shadow window attached to that boundary commitment.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row used for disk and shadow distance readback.*
4. *U is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row recording the Busemann-facing far-end comparison.*
5. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *or* $\mathsf{GromovBoundary}^{\uparrow}$ *readback row that exposes the public boundary token.*
6. *W is the fixed-boundary witness row, asserting that the boundary token read by V is preserved by T while the shadow window B is transported.*
7. *E is the refusal and exactness row excluding interior-center fixed points, selected ideal endpoints, quotient boundary equality, physical observer motion, and global compactification.*
8. *H records componentwise* hsame *transport for transport, shadow, metric, Busemann, visual-boundary, witness, refusal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from boundary transport through shadow readback, metric comparison, Busemann comparison, fixed-boundary witness, refusal, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, M, U, V, W, E, H, C, P, N$. It has no coordinate for a fixed interior disk center, selected ideal endpoint, quotient boundary equality, physical observer trajectory, completed compactification, or unlogged global hyperbolic model.

Theorem 7592.2 (Fixed-boundary hyperbolic shadow NameCert obligations)**.** *For every accepted* $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ *carrier*

$$S = (T, B, M, U, V, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}}$ *surface consists exactly of boundary-transport admission, boundary-shadow admission, hyperbolic metric readback, Busemann-boundary comparison, public boundary-token readback, fixed-boundary witness, refusal exactness, componentwise transport, displayed replay, provenance, and local naming. No obligation is stored outside $T, B, M, U, V, W, E, H, C, P, N$.*

Proof. Project the carrier of Definition 7592.1. The transport row T and shadow row B give the finite route whose boundary commitment is being tested.

The metric and boundary readback rows are downstream. Rows M, U, V expose the hyperbolic metric comparison, the Busemann-facing far-end comparison, and the public boundary token before any fixedness claim is read.

The fixed-boundary content is exactly row W. It states preservation of the displayed boundary token along the transported shadow window, while E records the refusals needed to keep the reading from becoming an interior-center or quotient-boundary claim.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no fixed interior center, selected ideal endpoint, quotient boundary equality, physical observer trajectory, completed compactification, or unlogged global model, none can become a local NameCert obligation. □

Theorem 7592.3 (Fixed-boundary hyperbolic shadow boundary route)**.** *Every boundary-shadow consumer of an accepted* $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ *carrier factors through*

$$\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}, \mathsf{HyperbolicBoundaryShadow}^{\uparrow}, \mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto$$

together with H, C, P, N. The route exports only a finite transported shadow whose public boundary token is preserved. It does not export an interior fixed point, selected ideal endpoint, quotient boundary equality, physical observer motion, completed compactification, or host-level boundary object.

Proof. Begin with the boundary-transport and shadow rows. The transport row T records the fixed boundary commitment, and the shadow row B records which finite shadow window is being moved.

Read the metric and Busemann comparisons next. Rows M and U expose the hyperbolic readbacks needed to interpret the shadow as a boundary-facing packet, and V supplies the public boundary token.

The preservation claim is then exactly W, constrained by the refusal row E. The structural rows H, C, P, N keep the exported route attached to the same finite transport, shadow, metric, Busemann, boundary-token, and witness rows.

Thus any consumer bypassing $T, B, M, U, V, W, E, H, C, P, N$ is not reading the accepted fixed-boundary shadow packet. □

Falsifiable prediction: Every accepted $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ read is refuted if it exports fixed-boundary shadow data without displaying $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ or $\mathsf{GromovBoundary}^{\uparrow}$, the fixed-boundary witness, refusal exactness, componentwise hsame transport, Cont replay, provenance, and local NameCert rows.

Independence witness: poincare-disk-boundary-transport, hyperbolic-boundary-shadow, hyperbolic-metric, hyperbolic-busemann-boundary, hyperbolic-observer-expansion: none is bijective to FixedBoundaryHyperbol because this packet alone binds boundary-fixed transport, finite shadow windows, metric comparison, Busemann readback, public boundary-token preservation, refusal exactness, and observer-expansion provenance into one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}$ packet is generated from PoincareDiskBoundaryTransport, HyperbolicBoundaryShadow, HyperbolicMetric, HyperbolicBusemannBoundary, HyperbolicVisualBoundary or GromovBoundary, HyperbolicObserverExpansion, a fixed-boundary witness, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7592.1, Definition 7592.2, and Definition 7592.3 seed the finite fixed-boundary shadow route for hyperbolic observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FixedBoundaryHyperbolicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close interior fixed points, selected ideal endpoints, quotient boundary equality, physical observer motion, completed compactification, host-level boundary objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary transport preservation, shadow admission, metric readback, Busemann comparison, public boundary-token readback, fixed-boundary witness exactness, refusal coverage, transport, replay, provenance, and local naming for BEDC.Derived.FixedBoundaryHyperbolicShadowUp.

Chapter 7593

A Concrete Naming Certificate for $\mathsf{CauchyTailExtraction}^{\uparrow}$

Following the Bishop–Bridges tail-extraction step for regular Cauchy data, the $\mathsf{CauchyTailExtraction}^{\uparrow}$ packet records the displayed witness carrier that turns a tail threshold into the finite windows, dyadic tolerances, and readback rows consumed by Real and Completion-facing routes. It sits after the tail-budget service of chapter 906, the finite stream interface of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the observation cap of chapter 903. The packet is a finite extracted-tail witness surface; it is not quotient stream equality, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7593.1 (Cauchy tail extraction carrier)**.** *A* $\mathsf{CauchyTailExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, T, O, W, D, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular Cauchy tail is being extracted.*
2. *T is the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row naming the finite tail threshold and its served request.*
3. *O is the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row bounding the number of observations used after T is displayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the limiter for the extracted tail.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window service.*
6. *Q is the extracted-tail witness row, recording the finite tail segment, its endpoint accounting, and the exact tolerance at which it is read back.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, T, O, W, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for $R, T, O, W, D, Q, S, C, P, N$.*
9. *C records displayed* Cont *replay of the route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyTailExtraction}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailExtraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, T, O, W, D, Q, S, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected host limits, countable choice, trichotomy, host equality, or ambient Real completeness.

Theorem 7593.2 (Cauchy tail extraction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailExtraction}^{\uparrow}$ *carrier*

$$E = (R, T, O, W, D, Q, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular source, tail-budget row, observation limiter, finite stream windows, dyadic tolerance ledger, extracted-tail witness, Real seal handoff, componentwise transport, displayed replay, provenance, and local naming.*

Every Real-facing or Completion-facing consumer of the extracted tail must therefore read the finite route

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailBudget}^{\uparrow} \longmapsto \mathsf{ObservationBudgetLimiter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \mapsto$$

through $R, T, O, W, D, Q, S, H, C, P, N$ before treating the tail as sealed. The route exports no quotient stream equality, selected representative, selected host limit, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7593.1. The accepted packet displays exactly the regular source row, tail-budget row, observation limiter, finite windows, dyadic ledger, extracted-tail witness, terminal seal, transport, replay, provenance, and local naming.

The extraction part is finite. The row T names the threshold supplied by the regular Cauchy tail budget, O bounds the observations after that threshold is displayed, and W, D, Q record the inspected windows, tolerance ledger, and extracted witness for the same request.

The seal row S is downstream of those fields, while H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for quotient stream equality, selected representatives, selected limits, choice, trichotomy, host equality, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: An accepted $\mathsf{CauchyTailExtraction}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the RegSeqRat source, RegularCauchyTailBudget threshold, ObservationBudgetLimiter cap, StreamName windows, DyadicRatCore tolerance ledger, extracted-tail witness, transport, replay, provenance, and local NameCert rows refutes the packet.

Independence witness: regular-cauchy-tail-budget, streamname, regseqrat, dyadicratcore, real, observationbudgetlimiter: no listed sibling is bijective to $\mathsf{CauchyTailExtraction}^{\uparrow}$ because this packet jointly stores the extracted-tail witness row Q together with the source, threshold, limiter, window, dyadic, seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailExtraction}^{\uparrow}$ packet is generated from a RegSeqRat source row, RegularCauchyTailBudget threshold row, ObservationBudgetLimiter cap, StreamName window service, DyadicRatCore tolerance ledger, extracted-tail witness, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7593.1 and Definition 7593.2 seed the finite Bishop tail-extraction route from regular Cauchy tail budgets to Real-facing seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-budget threshold exposure, observation-limiter exposure, stream-window service, dyadic tolerance readback, extracted-tail witness coverage, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailExtractionUp.

Chapter 7594

A Concrete Naming Certificate for $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$ packet records located nested closed-set completeness over $\mathsf{Real}^{\uparrow}$ as finite BEDC NameCert data. It sits after the located-real comparison surface of **??**, the nested-interval route of chapter 320, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the interval nesting route of chapter 7588, and the metric-completion handoff of chapter 1506. The packet names a shrinking inhabited located closed family and its completion-facing handoff; it is not open-cover compactness, sequential compactness, a selected point principle, quotient real equality, countable choice, or a host closed-set intersection theorem.

Definition 7594.1 (Bishop nested closed-set carrier)**.** *A* $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, I, D, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the located closed-set row, listing the inspected* $\mathsf{Real}^{\uparrow}$ *predicates together with located distance or apartness witnesses read through* $\mathsf{LocatedReal}^{\uparrow}$*.*
2. *I is the inhabited witness row, giving for each inspected depth a displayed finite witness row rather than a chosen global point.*
3. *D is the diameter-modulus row, assigning every requested dyadic radius a displayed depth after which the inspected closed-set row has that diameter bound.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the diameter and locatedness reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the nested witness windows and the dyadic diameter schedule.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after F, I, D, Q, R have been displayed.*
7. *H records componentwise* hsame *transport for located closed-set rows, inhabited witnesses, diameter moduli, dyadic tolerances, regular rational readback, Real sealing, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from nested located closed-set inspection through witness windows, diameter-modulus comparison, regular rational readback, and Real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{BishopIntervalCantor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, I, D, Q, R, A, H, C, P, N$. It has no coordinate for open-cover compactness, sequential compactness, an unbounded choice schedule, quotient equality of real names, a host closed-set intersection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopNestedClosedSetPrincipleUp`

Theorem 7594.2 (Bishop nested closed-set NameCert obligations)**.** *For every accepted* BishopNestedClosedSetPrin *carrier*

$$K = (F, I, D, Q, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}}$ *surface consists exactly of located closed-set admission, inhabited finite witness exposure, diameter-modulus admission, dyadic tolerance readback, regular rational readback, terminal Real seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside these displayed rows.*

Proof. Project the carrier of Definition 7594.1. The located closed-set row F, witness row I, diameter-modulus row D, dyadic row Q, readback row R, and seal row A are the only mathematical coordinates.

The locatedness and inhabitedness obligations are finite because the packet records inspected rows rather than a host family of all closed subsets. The witness row gives depthwise evidence, and the diameter row supplies a displayed modulus for the requested precision.

The structural rows H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the carrier contains no open-cover compactness, sequential compactness, global choice schedule, quotient equality, host intersection, or ambient completeness coordinate, none can become a local NameCert obligation. □

Theorem 7594.3 (Bishop nested closed-set diameter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$ *carrier must expose*

$$\mathsf{LocatedReal}^{\uparrow}, \quad \mathsf{NestedInterval}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows $F, I, D, Q, R, A, H, C, P, N$ before it reads diameter-modulus data as a completed Real-facing witness. The handoff may cite chapter 7588 and chapter 1506; it does not claim open-cover compactness, sequential compactness, countable choice, quotient real equality, or host closed-set completeness.

Proof. Start with the located closed-set and witness rows. The consumer reads F and I as finite displayed rows, so the nested family is available only through inspected located data and depthwise inhabitants.

Next read the diameter modulus. The row D is measured by the dyadic tolerance row Q, and therefore any precision request reaches the nested family through $\mathsf{DyadicRatCore}^{\uparrow}$ rather than through a host compactness argument.

The regular rational readback row R is the only route into the terminal $\mathsf{Real}^{\uparrow}$ seal A. The same route is preserved by H, C, P, N, so completion consumers receive a finite diameter handoff compatible with the interval nesting and metric-completion siblings.

No additional coordinate remains from which a selected point, open-cover compactness theorem, sequential compactness theorem, countable-choice schedule, quotient equality, or ambient completeness claim could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}$ packet is generated from nested located closed Real rows, inhabited witness rows, diameter-modulus rows, DyadicRatCore tolerance rows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7594.1, Definition 7594.2, and Definition 7594.3 seed the finite located nested closed-set route over Real, LocatedReal, NestedInterval, DyadicRatCore, RegSeqRat, MetricCompletion, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopNestedClosedSetPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, selected intersection points, countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located closed-set exposure, inhabited witness exposure, diameter-modulus exactness, DyadicRatCore tolerance readback, RegSeqRat extraction, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopNestedClosedSetPrincipleUp.

Chapter 7595

A Concrete Naming Certificate for $\mathsf{CartanAngularBoundaryInvariant}^{\uparrow}$

The $\mathsf{CartanAngularBoundaryInvariant}^{\uparrow}$ packet realizes the ordered-boundary-triple branch of chapter 9463 as finite hyperbolic phase data. It sits after the boundary-shadow packet of chapter 3813, the hyperbolic cross-ratio packet of chapter 3948, the Mobius disk cross-ratio packet of chapter 7589, and the complex phase-surface fiber of chapter 6044. The packet records an ordered boundary triple, cross-ratio phase readback, Mobius stability, refusal rows, and provenance; it is not a completed boundary quotient, a selected ideal-triangle object, a host Lie-group invariant, or a global hyperbolic compactification.

Definition 7595.1 (Cartan angular boundary invariant carrier)**.** *A* $\mathsf{CartanAngularBoundaryInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, Q, M, S, \Phi, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the ordered boundary-triple row, naming three oriented boundary slots and their admissibility witnesses.*
2. *Q is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *phase-readback row obtained by inserting the triple into the displayed boundary comparison window.*
3. *M is the* $\mathsf{MobiusDiskCrossRatio}^{\uparrow}$ *transport row, recording the disk-map comparison under which the phase read remains stable.*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row that anchors the triple to visible boundary-shadow data.*
5. *Φ is the* $\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}$ *readback row through which the angular phase is exposed as a finite surface value.*
6. *O is the refusal row excluding completed boundary quotients, selected ideal triangles, hidden host Mobius actions, global curvature, and unlogged complex-field completion.*
7. *H records componentwise* hsame *transport for boundary triples, cross-ratio phase readback, Mobius transport, boundary shadows, phase-surface fibers, refusal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from ordered boundary triple inspection through cross-ratio readback, Mobius transport, boundary-shadow anchoring, phase readback, refusal, provenance, and local naming.*
9. *P is the* Pkg *provenance over* CartanAngularBoundaryInvariant$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, MobiusDiskCrossRatio$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, ComplexPhaseSurfaceFiber$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CartanAngularBoundaryInvariant}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, Q, M, S, \Phi, O, H, C, P, N$. It has no coordinate for completed boundary quotients, selected ideal-triangle representatives, a host Mobius group action, global curvature, or ambient complex-field completion.

Lean4 (statement) `BEDC.Derived.CartanAngularBoundaryInvariantUp`

Theorem 7595.2 (Cartan angular boundary invariant NameCert obligations)**.** *For every accepted* CartanAngularBoundaryInvariant$^{\uparrow}$ *carrier*

$$J = (B, Q, M, S, \Phi, O, H, C, P, N),$$

the local NameCert$_{\text{CartanAngularBoundaryInvariant}^{\uparrow}}$ *surface consists exactly of ordered boundary-triple admission, cross-ratio phase readback, Mobius disk cross-ratio transport, boundary-shadow anchoring, complex phase-surface fiber readback, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside these displayed rows.*

Proof. Project the carrier of Definition 7595.1. The boundary triple, cross-ratio readback, Mobius transport, boundary shadow, phase fiber, and refusal rows are exactly B, Q, M, S, Φ, O.

The invariant is boundary-facing because B is ordered and S anchors it to visible boundary-shadow data. The phase value is not a host angle: it is the finite readback Φ supplied after Q and M are displayed.

The structural rows H, C, P, N preserve the same packet under componentwise transport, replay, provenance, and naming. Since no other coordinate is present, the local certificate cannot store a completed boundary quotient, selected ideal triangle, host Mobius action, global curvature theorem, or ambient complex-field completion. □

Theorem 7595.3 (Cartan angular boundary invariant Mobius stability)**.** *Every Mobius-stability read of an accepted* CartanAngularBoundaryInvariant$^{\uparrow}$ *carrier factors through the displayed route*

$$B \longmapsto Q \longmapsto M \longmapsto S \longmapsto \Phi$$

together with O, H, C, P, N. The future Lean TasteGate *for this packet must include* Nontrivial CartanAngularBoundar *and* FieldFaithful CartanAngularBoundaryInvariant$^{\uparrow}$ *instances preserving the ordered triple, phase readback, Mobius transport, boundary-shadow, and provenance rows.*

Proof. Start from the ordered boundary-triple row B. The cross-ratio row Q reads its phase only after the boundary slots have been displayed.

The Mobius disk cross-ratio row M is then the stability witness. It compares the transported triple without replacing the displayed route by a host Mobius group action or quotient boundary equality.

The boundary-shadow row S and phase-surface row Φ expose the visible boundary and angular readback surfaces. The refusal and structural rows O, H, C, P, N keep this route finite and record the excluded over-reads.

Thus a stable invariant read is exactly the displayed packet. The named TasteGate obligations demand Nontrivial and FieldFaithful evidence for the same finite row separation. □

Falsifiable prediction: Every accepted CartanAngularBoundaryInvariant$^{\uparrow}$ read is refuted if it exports angular boundary-invariant data without displaying ordered boundary triples, HyperbolicCrossRatio$^{\uparrow}$ phase readback, MobiusDiskCrossRatio$^{\uparrow}$ transport, HyperbolicBoundaryShadow$^{\uparrow}$ anchoring, ComplexPhaseSurfaceFibe readback, refusal exactness, componentwise hsame transport, Cont replay, provenance, and local NameCert rows.

Independence witness: hyperbolic-cross-ratio, mobius-disk-cross-ratio, hyperbolic-boundary-shadow, complex-phase-surface-fiber: HyperbolicCrossRatio$^{\uparrow}$ compares boundary data without owning the ordered triple and phase-fiber provenance; MobiusDiskCrossRatio$^{\uparrow}$ supplies transport without the Cartan triple row; HyperbolicBoundaryShadow$^{\uparrow}$ anchors boundary visibility without angular phase readback; ComplexPhaseSurfaceFiber$^{\uparrow}$ reads phase fibers without boundary-triple stability. None is bijective to CartanAngularBoundaryInvariant$^{\uparrow}$.

Closure status for NameCert$_{\text{CartanAngularBoundaryInvariant}^{\uparrow}}$.

Constructive story: A CartanAngularBoundaryInvariant$^{\uparrow}$ packet is generated from ordered boundary-triple rows, HyperbolicCrossRatio phase readback, MobiusDiskCrossRatio transport, HyperbolicBoundaryShadow anchoring, ComplexPhaseSurfaceFiber readback, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CartanAngularBoundaryInvariant}^{\uparrow}}$, seed)

Scope closed: Definition 7595.1, Definition 7595.2, and Definition 7595.3 seed the finite boundary-triple angular invariant route for the hyperbolic phase geometry vision.

Formal verification: FormalStatus(NameCert$_{\text{CartanAngularBoundaryInvariant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed boundary quotients, selected ideal triangles, host Mobius group actions, global curvature, ambient complex-field completion, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ordered boundary-triple exposure, cross-ratio phase readback, Mobius stability, boundary-shadow anchoring, phase-fiber readback, Nontrivial and FieldFaithful TasteGate instances, refusal coverage, transport, replay, provenance, and local naming for BEDC.Derived.CartanAngularBoundaryInvariantUp.

Chapter 7596

A Concrete Naming Certificate for $\mathsf{FiniteProductCompletion}^{\uparrow}$

Following Bourbaki and Kelley on finite products and completions, the $\mathsf{FiniteProductCompletion}^{\uparrow}$ packet records the finite route by which a displayed finite family of metric or uniform sources is completed componentwise and read back as one product-completion object. It sits after the product naming surface of chapter 34, the metric source of chapter 57, the completeness boundary of chapter 128, the product-uniform route of chapter 4040, the Cauchy product-completion carrier of chapter 3797, and the completion universal property of chapter 397. The packet names only the finite componentwise completion surface; it does not import quotient completion equality, selected representatives, countable products, arbitrary limits, choice, or ambient real completeness.

Definition 7596.1 (Finite product completion carrier)**.** *A* $\mathsf{FiniteProductCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, X, M, U, Q, \Pi, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index packet. It records the finite list of component names and the admissible projection indices, with no coordinate for countable or dependent products.*
2. *X is the product carrier row. It reads the* ***??*** *surface componentwise and exposes only tuples whose entries are named by I.*
3. *M and U are the product metric and product uniformity rows. They record the finite maximum, finite sum, or finite entourage route chosen for the displayed sources, as mediated by* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{ProductUniformSpace}^{\uparrow}$*.*
4. *Q is the component completion row. For each index in I, it records the completion-facing packet, the local completeness certificate, and the componentwise terminal seal.*
5. *Π is the projection readback row. Every component projection out of the completed product first reads I, then X, then the corresponding component of Q.*
6. *L is the universal route row. A map into the completed finite product is accepted only by giving all component maps and their completion-compatible routes before the terminal product read.*
7. *H records componentwise* hsame *transport for the index packet, product carrier, metric or uniformity rows, component completions, projections, and universal route.*

8. *C records displayed* Cont *replay from finite source data through product formation, component completion, projection readback, and universal routing.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteProductCompletion}^{\uparrow}$, *??*, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ProductUniformSpace}^{\uparrow}$, $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteProductCompletion}^{\uparrow}}$ *row.*

The accepted carrier contains no coordinate for countable products, quotient completions, selected limits, choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7596.2 (Finite product completion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteProductCompletion}^{\uparrow}$ *carrier $F = (I, X, M, U, Q, \Pi, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteProductCompletion}^{\uparrow}}$ *surface consists exactly of the finite index admission, product carrier admission, product metric or uniformity admission, component completion admission, projection readback coverage, universal route coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows displayed in Definition 7596.1.*

Proof. Project the carrier. The rows $I, X, M, U, Q, \Pi, L, H, C, P, N$ are the only rows available to the local naming surface, so every accepted obligation must be read through this finite inventory.

The index row I and product carrier row X force product formation to be finite and componentwise. The product metric or uniformity rows M, U then supply the only admissible product structure before completion data can be read.

Component completion is confined to Q. Projection reads use Π, and maps into the product use L; both rows refer back to the same finite index packet and component completion rows.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no coordinate for countable products, quotient completions, selected limits, choice, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7596.3 (Finite product completion projection and universal route)**.** *Every projection read or completion-universal-property read exported by an accepted* $\mathsf{FiniteProductCompletion}^{\uparrow}$ *packet factors through the displayed product, completion, and projection rows before the terminal completion seal. More explicitly, the read first consumes the finite index packet I, the product carrier X, the product metric or uniformity rows M, U, and the component completion row Q; it then reaches the projection row Π or universal route row L, and only afterward uses H, C, P, N for transport, replay, provenance, and local naming.*

Proof. Begin with an accepted carrier $F = (I, X, M, U, Q, \Pi, L, H, C, P, N)$. The carrier definition makes I and X prior to every product-facing read, so a consumer cannot inspect a projection or universal route before the finite product data are displayed.

The metric or uniformity rows M, U and the component completion row Q provide the completion-facing surface. Thus a projection read reaches Π only after the component completion row has named the relevant coordinate, and a universal-property read reaches L only after all component routes have been displayed.

The remaining rows preserve the same route. Componentwise hsame transport is recorded in H, replay in C, provenance in P, and local naming in N.

No row supplies countable product formation, quotient completion equality, selected limits or representatives, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every accepted projection or universal-property read factors through the displayed finite product-completion route before the terminal seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteProductCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteProductCompletion}^{\uparrow}$ packet is generated from a finite index packet, product carrier row, product metric or uniformity rows, component completion rows, projection readback rows, a universal route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteProductCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7596.1, Definition 7596.2, and Definition 7596.3 seed the finite product-completion route used by completion-product consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteProductCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable products, quotient completions, selected limits or representatives, choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode finite index admission, product carrier admission, component completion admission, projection readback coverage, universal route coverage, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FiniteProductCompletionUp.

Chapter 7597

A Concrete Naming Certificate for $\mathsf{FiniteCoverNerve}^{\uparrow}$

Following the classical nerve construction for finite covers, and the Bishop–Bridges finite-net reading of compact metric continuity, the $\mathsf{FiniteCoverNerve}^{\uparrow}$ packet records cover membership, overlap incidence, nerve faces, rational radius folds, and the uniform-modulus handoff as displayed BEDC rows. It realizes the finite cover branch of Definition 9237.5 in chapter 9237 without invoking open-cover compactness or a selected subcover theorem. The packet sits after $\mathsf{CompactMetric}^{\uparrow}$ finite-cover data, $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus rows, $\mathsf{Rat}^{\uparrow}$ radius comparisons, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer.

Definition 7597.1 (Finite cover nerve carrier)**.** *A* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (K, V, O, F, R, B, U, H, C, P, Q)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite cover source row, displaying a finite* $\mathsf{ProbeBundle}$ *of centers and cover-radius histories.*
2. *V is the cover-member row, recording which displayed probe covers which inspected source point.*
3. *O is the overlap-incidence row, recording finite pairwise and higher overlap reads among displayed probes.*
4. *F is the nerve-face row, accepting exactly the finite probe lists whose overlap rows are displayed by O.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *radius and lower-bound comparison row attached to the displayed cover radii.*
6. *B is the rational-fold row that computes a finite positive lower-bound ledger from the accepted face and radius rows.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed after the rational fold.*
8. *H records componentwise* hsame *transport for K, V, O, F, R, B, U and the structural rows.*
9. *C records displayed* Cont *replay through cover membership, overlap incidence, nerve faces, rational folding, and uniform-modulus handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverNerve}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, V, O, F, R, B, U, H, C, P, Q$. It has no coordinate for arbitrary open covers, selected subcovers, quotient equality of covers, trichotomy, countable choice, host set equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteCoverNerveUp.FiniteCoverNerveTasteGate single carrier alignment`

Theorem 7597.2 (Finite cover nerve NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *carrier*

$$N = (K, V, O, F, R, B, U, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverNerve}^{\uparrow}}$ *surface consists exactly of carrier admission, cover-member exposure through K, V, overlap-incidence readback through O, nerve-face admission through F, rational radius comparison through R, finite rational folding through B, uniform-modulus handoff through U, and structural transport, replay, provenance, and local naming through H, C, P, Q.*

Lean4 ✓ `BEDC.Derived.FiniteCoverNerveUp.FiniteCoverNerveCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 7597.1. The accepted packet displays the finite cover source, cover-member row, overlap-incidence row, nerve-face row, rational comparison row, finite fold, uniform-modulus handoff, and structural rows.

Cover membership is exposed through K, V. Every membership read is tied to a displayed probe from the finite $\mathsf{ProbeBundle}$, so the packet has no route to an arbitrary open cover.

Overlap and face obligations are exposed through O, F. A nerve face is admitted only when its finite overlap rows are already displayed, and the carrier has no separate coordinate for host-set intersection equality.

The radius and output obligations are exposed through R, B, U. The rational fold reads only the displayed radius comparisons before the uniform-modulus handoff is consumed. □

Theorem 7597.3 (Finite cover nerve compact-uniform handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *row exported by an accepted* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *packet factors through the finite route*

$$K, V, O, F \longmapsto R, B \longmapsto U, H, C, P, Q.$$

The route first reads compact-metric cover membership and overlap incidence, then accepts finite nerve faces, then folds rational radius data, and only then exposes the uniform-modulus handoff. It exports no open-cover compactness theorem, selected subcover, quotient cover equality, trichotomy, countable choice, host set equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteCoverNerveUp.FiniteCoverNerveCompactUniformHandoff`

Proof. Start with the carrier rows K, V, O, F. The compact metric source supplies the finite probe list, the member row records coverage reads, the incidence row records finite overlaps, and the face row records exactly the accepted finite nerve faces.

Next read R, B. The rational comparison row exposes radius histories, and the finite fold row records the lower-bound ledger used by the compact-uniform construction.

Finally read U, H, C, P, Q. The uniform-modulus row is downstream of the fold, while transport, replay, provenance, and local naming repack only the same finite cover-nerve data. □

Theorem 7597.4 (Finite cover nerve vision realization)**.** *The finite cover and rational lower-bound construction used in Definition 9237.5 is realized by an accepted* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *carrier as CompactMetric cover-member rows, overlap-incidence rows, nerve-face rows, rational-fold rows, and a UniformModulus handoff. The realization is a concrete finite NameCert packet, not a separate compactness or selection principle.*

Lean4 ✓ `BEDC.Derived.FiniteCoverNerveUp.FiniteCoverNerveVisionRealization`

Proof. Read the vision theorem through chapter 9237. Its finite ProbeBundle and center-radius coverage data are the rows K, V, while the finite overlap checks and accepted faces are the rows O, F.

The rational lower-bound construction is recorded by R, B. The output tolerance-only uniform-modulus ledger is recorded by U, with H, C, P, Q supplying transport, replay, provenance, and local naming.

Thus the vision-side finite cover branch lands as one concrete NameCert carrier. Since the carrier contains no arbitrary open-cover, selected subcover, quotient-cover, choice, trichotomy, host-equality, or completeness coordinate, the realization remains finite. □

Theorem 7597.5 (Finite cover nerve sibling route)**.** *Let*

$$N = (K, V, O, F, R, B, U, H, C, P, Q)$$

be an accepted $\mathsf{FiniteCoverNerve}^{\uparrow}$ *carrier. Any consumer that reads* N *as a compact-uniform or finite-cover dependency must first expose the compact and metric source rows through chapter 60, chapter 129, and chapter 57; the finite cover-radius and lower-bound rows through chapter 452; and the downstream uniform-modulus rows through chapter 369. The route is substantive:* K, V, O, F *supply the compact finite cover and overlap data,* R, B *supply the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold, and* U, H, C, P, Q *supply the uniform-modulus handoff, transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.FiniteCoverNerveUp.FiniteCoverNerveOrphanSiblingRoute`

Proof. Project the carrier rows $K, V, O, F, R, B, U, H, C, P, Q$. The compact and metric source dependency is read from K together with the cover-member and overlap rows V, O, F, so the finite nerve cannot be inspected before the displayed compact finite-cover surface is present.

The rational dependency is read from R, B. These rows record radius comparisons and the finite lower-bound ledger, using the same finite cover rows rather than a separate host Lebesgue-number theorem.

Finally the compact-uniform dependency is read from U, H, C, P, Q. The uniform-modulus handoff is downstream of the cover and rational-fold rows, while transport, replay, provenance, and local naming keep the consumer attached to the same packet. Thus the cited sibling chapters are load-bearing sources of rows, not decorative references. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverNerve}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7597.1, Definition 7597.2, Definition 7597.3, and Definition 7597.4 seed the finite cover-member, overlap-incidence, nerve-face, rational-fold, uniform-modulus, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverNerve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient equality of covers, trichotomy, countable choice, host set equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-member exactness, overlap-incidence exactness, nerve-face admission, rational-fold stability, uniform-modulus handoff, and nonescape proofs for BEDC.Derived.FiniteCoverNerveUp.

Constructive story: A FiniteCoverNerve$^{\uparrow}$ packet is generated from CompactMetric finite-cover rows, cover-member rows, overlap-incidence rows, nerve-face rows, Rat radius comparisons, rational-fold rows, UniformModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7598

A Concrete Naming Certificate for $\mathsf{ContinuityModulusLattice}^{\uparrow}$

Following Bishop and Bridges on compact metric continuity and constructive moduli, the $\mathsf{ContinuityModulusLattice}^{\uparrow}$ packet records the finite order surface on which displayed local moduli can be compared, combined by finite meet, and handed to a uniform-modulus consumer. It sits after the continuous-map rows of chapter 130, the uniform-modulus consumer of chapter 369, the compact metric finite-net source of chapter 129, the metric source rows of chapter 57, and rational tolerance data from chapter 37. The packet names a finite lattice of modulus rows; it is not open-cover compactness, a selected modulus functional, quotient equality of functions, trichotomy, countable choice, or ambient completeness.

Definition 7598.1 (Continuity modulus lattice carrier)**.** *A* $\mathsf{ContinuityModulusLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, D, F, U, R, O, M, J, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *over the same* $\mathsf{Met}^{\uparrow}$ *carrier.*
2. *D is the metric-distance comparison row used to compare source balls and target tolerances.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus histories are being ordered.*
4. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *row that consumes a tolerance-only modulus history.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *tolerance and lower-bound comparison row.*
6. *O is the finite preorder row on displayed modulus histories.*
7. *M is the finite meet row, recording the common lower precision history for finitely many comparable modulus rows.*
8. *J is the join-readback row used only to report compatibility of already displayed modulus histories.*
9. *H records componentwise* hsame *transport for X, D, F, U, R, O, M, J and the structural rows.*
10. *C records displayed* Cont *replay through compact-source, metric, continuous-map, rational, order, meet, and uniform-modulus reads.*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuityModulusLattice}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{ContinuityModulusLattice}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, D, F, U, R, O, M, J, H, C, P, N$. It has no coordinate for open-cover compactness, selected moduli, quotient equality, trichotomy, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7598.2 (Continuity modulus lattice NameCert obligations)**.** *For every accepted* ContinuityModulusLattice *carrier*

$$L = (X, D, F, U, R, O, M, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuityModulusLattice}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-metric source exposure through X, D, continuous-map modulus exposure through F, rational tolerance comparison through R, finite preorder readback through O, finite meet readback through M, compatibility readback through J, uniform-modulus handoff through U, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Carrier admission is projection from Definition 7598.1. The accepted packet displays the compact metric source, metric comparison row, continuous-map row, uniform-modulus consumer row, rational tolerance row, finite order rows, and structural rows.

The order obligations are read through O and R. A comparison between two modulus histories must expose the rational tolerance comparisons used by the preorder row, so the packet cannot replace finite order by an ambient order theorem.

The meet obligations are read through M. The meet row records only a finite common lower precision history for already displayed modulus histories, and J reports compatibility without adding an independent selector.

Finally U, H, C, P, N keep the handoff local. The uniform-modulus consumer receives only the displayed finite meet route, while the structural rows transport, replay, source, and name the same finite data. □

Theorem 7598.3 (Continuity modulus lattice finite-meet stability)**.** *Every finite meet read exported by an accepted* $\mathsf{ContinuityModulusLattice}^{\uparrow}$ *packet remains stable under* hsame *transport of the displayed rows $X, D, F, U, R, O, M, J, H, C, P, N$. A downstream compact-uniform or completion-modulus consumer may cite the meet as a finite order object over continuous-map and rational tolerance rows, but it may not read the meet as open-cover compactness, a selected global modulus, quotient equality, trichotomy, countable choice, host function equality, or ambient completeness.*

Proof. Project the carrier rows. The finite meet row M is stored together with the rational comparison row R, the preorder row O, the compatibility row J, and the structural transport row H.

Transport first acts componentwise on X, D, F, U, R, O, M, J. Since M is not a detached selector, a transported meet read must carry along the same compact-source, metric, continuous-map, rational, order, and compatibility rows.

The replay and provenance rows C, P, N then repack the transported data as the same local NameCert surface. No refused object is introduced by this repacking, so finite-meet stability is exactly stability of the displayed carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuityModulusLattice}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuityModulusLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7598.1, Definition 7598.2, and Definition 7598.3 seed the finite order surface on compact metric, metric, continuous-map, rational tolerance, finite preorder, finite meet, compatibility, uniform-modulus, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{ContinuityModulusLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected moduli, quotient equality, trichotomy, countable choice, host function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite preorder exactness, finite meet stability, compatibility readback, uniform-modulus handoff, and nonescape proofs for BEDC.Derived.ContinuityModulusLatticeUp.

Constructive story: A ContinuityModulusLattice$^{\uparrow}$ packet is generated from compact metric source rows, metric comparison rows, continuous-map modulus rows, uniform-modulus handoff rows, rational tolerance comparisons, finite preorder and meet rows, compatibility readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7599

A Concrete Naming Certificate for $\mathsf{PointedSet}^{\uparrow}$

Following Bourbaki and standard category-theoretic usage of pointed sets, the $\mathsf{PointedSet}^{\uparrow}$ packet names a finite carrier together with a displayed basepoint row. It sits after the finite Boolean tag discipline of chapter 32, the nullable constructor route of chapter 33, the coproduct sibling surface of chapter 35, the product sibling surface of chapter 34, and the finite list observation route of chapter 36. The packet records basepoint admission, option-style absence routing, product and wedge handoff rows, transport, replay, provenance, and local naming; it is not a host pointed type, a selected global element oracle, a quotient of carriers, or a completed algebraic structure.

Definition 7599.1 (Pointed set carrier)**.** *A* $\mathsf{PointedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, b, O, P, W, H, C, Q, N)$$

with the following displayed rows:

1. *X is the finite carrier row, read as BEDC data rather than as a host set.*
2. *b is the distinguished basepoint row, displayed before any pointed consumer may read a map, product, or wedge route.*
3. *O is the* $\mathsf{Option}^{\uparrow}$ *basepoint route, recording whether a consumer reads the distinguished point directly or reads an optional payload after the basepoint boundary is fixed.*
4. *P is the* $\mathsf{Prod}^{\uparrow}$ *handoff row, recording the ordered pair of carrier and basepoint data consumed by product-facing routes.*
5. *W is the wedge handoff row, built from* $\mathsf{Sum}^{\uparrow}$ *and* $\mathsf{Bool}^{\uparrow}$ *branch marks together with the common basepoint row.*
6. *H records componentwise* hsame *transport for carrier, basepoint, option, product, wedge, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each pointed consumer through b, O, P, W, H.*
8. *Q is the* Pkg *provenance over* $\mathsf{PointedSet}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, b, O, P, W, H, C, Q, N$. It has no coordinate for a host pointed type, global element choice, quotient carrier equality, selected coproduct representative, or ambient algebraic structure.

Theorem 7599.2 (Pointed set NameCert obligations)**.** *For every accepted* $\mathsf{PointedSet}^{\uparrow}$ *carrier* $S = (X, b, O, P, W, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedSet}^{\uparrow}}$ *surface consists exactly of carrier admission, basepoint admission, option-basepoint routing, product handoff, wedge handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows of* S*.*

Proof. Project the carrier of Definition 7599.1. The rows X and b expose the finite carrier and distinguished point before any consumer can read pointed structure.

The option, product, and wedge obligations are the rows O, P, W. The option row uses the nullable sibling only after the basepoint row is visible; the product row packages carrier and basepoint data together; the wedge row uses sum and Boolean branch marks while retaining the common basepoint.

The structural rows add no hidden object. Componentwise hsame transport in H, displayed Cont replay in C, provenance in Q, and the local naming row N carry the same finite packet. Since the excluded host pointed type, choice, quotient, selected representative, and ambient algebraic coordinates are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7599.3 (Pointed set option basepoint route)**.** *Every* $\mathsf{Option}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointedSet}^{\uparrow}$ *packet must read the displayed basepoint row* b *before it may distinguish an absent optional value from a present payload. The route passes through* X, b, O, H, C, Q, N *and exports no host default element, selected payload representative, quotient carrier equality, or global choice of basepoint.*

Proof. Start with the carrier projection. The optional route is not a detached nullable object; it is the row O of the displayed packet $S = (X, b, O, P, W, H, C, Q, N)$.

Read the basepoint first. The row b supplies the distinguished point against which the absent optional case is interpreted, while the finite carrier row X supplies the payload boundary for present cases.

Transport and replay preserve that order. The rows H, C, Q, N carry the basepointed option route without adding a host default element, selected representative, quotient equality, or global choice coordinate. □

Theorem 7599.4 (Pointed set product and wedge handoff)**.** *Every product-facing or wedge-facing consumer of an accepted* $\mathsf{PointedSet}^{\uparrow}$ *carrier reads the finite route*

$$X, b \longmapsto P, W \longmapsto H, C, Q, N.$$

The $\mathsf{Prod}^{\uparrow}$ *handoff may package the carrier and basepoint rows together, and the wedge handoff may read* $\mathsf{Sum}^{\uparrow}$ *and* $\mathsf{Bool}^{\uparrow}$ *branch marks, but neither route identifies carriers by quotient equality, chooses a coproduct representative, imports a host pointed type, or exports an ambient algebraic structure.*

Proof. Project S through Definition 7599.1. The product route is the row P, and the wedge route is the row W; both rows are downstream of the carrier row X and the distinguished basepoint row b.

For product consumers, P packages the finite carrier and basepoint data without introducing a new equality principle. For wedge consumers, W uses the $\mathsf{Sum}^{\uparrow}$ branch surface and $\mathsf{Bool}^{\uparrow}$ marks while keeping the same basepoint row visible across the branches.

The rows H, C, Q, N transport, replay, source, and name the same finite handoff. Since no quotient carrier equality, selected coproduct representative, host pointed type, or ambient algebraic structure appears in the packet, neither product nor wedge consumption can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedSet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7599.1, Definition 7599.2, Definition 7599.3, and Definition 7599.4 seed the finite basepoint certificate over Bool, Option, Sum, Prod, List, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedSet}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.PointedSetUp`

Bridge status: none

Not claimed: This chapter does not close host pointed types, selected global elements, quotient carrier equality, selected coproduct representatives, ambient algebraic structure, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basepoint admission, option-basepoint routing, product and wedge handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.PointedSetUp.

Constructive story: A $\mathsf{PointedSet}^{\uparrow}$ packet is generated from a finite carrier row, a displayed basepoint row, Option, Prod, Sum, and Bool handoff rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7600

A Concrete Naming Certificate for $\mathsf{FrontdoorCriterion}^{\uparrow}$

Following Pearl's front-door criterion, the $\mathsf{FrontdoorCriterion}^{\uparrow}$ packet records the finite mediation certificate used to identify an intervention-facing query through observed mediator rows. It sits after the intervention-row packet of chapter 1290, the normalized probability surface of chapter 184, and the carried random-variable rows of chapter 185. The packet names the front-door surface as BEDC data; it does not claim full do-calculus, back-door completeness, counterfactual semantics, or probability-space completion.

Definition 7600.1 (Front-door criterion carrier)**.** *A* $\mathsf{FrontdoorCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, M, Y, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the treatment* $\mathsf{RandomVar}^{\uparrow}$ *row whose intervention-facing value is read by the query.*
2. *M is the mediator* $\mathsf{RandomVar}^{\uparrow}$ *row through which the treatment effect must pass.*
3. *Y is the outcome* $\mathsf{RandomVar}^{\uparrow}$ *row whose response is queried.*
4. *I is the* $\mathsf{CausalIntervention}^{\uparrow}$ *row that names the finite intervention on T.*
5. *B is the observed-blocking ledger, recording that the mediator row blocks every displayed directed treatment-to-outcome path admitted by the packet.*
6. *A is the adjustment ledger, recording the observed rows used to read the treatment-to-mediator and mediator-to-outcome probability factors.*
7. *H records componentwise* hsame *transport for the treatment, mediator, outcome, intervention, blocking, adjustment, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

   $$T, I \longmapsto M \longmapsto Y$$

   together with the blocking and adjustment ledgers.
9. *P is the* Pkg *provenance over* $\mathsf{FrontdoorCriterion}^{\uparrow}$*,* $\mathsf{CausalIntervention}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FrontdoorCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host structural causal model, selected counterfactual universe, unbounded graph search, hidden back-door completeness proof, or completed probability space.

Lean4 ✓ `BEDC.Derived.FrontdoorCriterionUp.FrontdoorCriterionTasteGate single carrier alignment`

Theorem 7600.2 (Front-door criterion NameCert obligations)**.** *For every accepted* $\mathsf{FrontdoorCriterion}^{\uparrow}$ *carrier* $F = (T, M, Y, I, B, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrontdoorCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed treatment, mediator, outcome, intervention, observed-blocking, adjustment, transport, replay, provenance, and local naming rows. Any consumer that reads a front-door claim must display the treatment-to-mediator route, the mediator-to-outcome route, the observed blocking ledger, the adjustment ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local name before the claim is public.*

Proof. Project the carrier of Definition 7600.1. Its rows expose $T, M, Y, I, B, A, H, C, P, N$ and no other coordinate.

Read the causal route first. The intervention row I is attached to the treatment row T, the mediator row M is the only displayed handoff between treatment and outcome, and the outcome row Y is consumed only after that handoff is visible.

Read the two ledgers next. The blocking row B records the displayed path-blocking surface, while the adjustment row A records the observed probability-facing rows used by the two front-door factors. These rows are supported by the $\mathsf{ProbSpace}^{\uparrow}$ and $\mathsf{RandomVar}^{\uparrow}$ siblings rather than by a host model.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for full do-calculus, back-door completeness, counterfactual semantics, or probability-space completion, those objects are not local NameCert obligations. □

Theorem 7600.3 (Front-door criterion causal handoff)**.** *Every intervention-facing consumer of an accepted* $\mathsf{FrontdoorCriterion}^{\uparrow}$ *packet reaches its causal query only after reading the finite route*

$$T, I, B, A, H, C, P, N \longmapsto M \longmapsto Y.$$

The handoff is a displayed mediator route from $\mathsf{CausalIntervention}^{\uparrow}$ *through* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{RandomVar}^{\uparrow}$ *rows. It exports no host causal graph, full do-calculus derivation, back-door completeness theorem, selected counterfactual, or completed probability space.*

Proof. Start with the intervention row I and treatment row T. The carrier supplies no intervention-facing conclusion until the blocking ledger B and adjustment ledger A have been inspected with the same transport, replay, provenance, and naming rows.

The mediator row M is then the only displayed bridge to the outcome row Y. Thus the consumer can reach the outcome-facing query only through the finite mediator handoff and not by projecting a hidden causal graph or back-door theorem.

Because the probability-facing rows are cited through $\mathsf{ProbSpace}^{\uparrow}$ and $\mathsf{RandomVar}^{\uparrow}$, the route stays inside existing NameCert siblings. The absence of any counterfactual or probability-completion coordinate gives the claimed nonescape boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrontdoorCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrontdoorCriterion}^{\uparrow}$ packet is generated from treatment, mediator, and outcome random-variable rows; a finite intervention row; observed blocking and adjustment ledgers; componentwise hsame transport; displayed Cont replay; Pkg provenance; and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{FrontdoorCriterion}\uparrow}$, seed)

Scope closed: Definition 7600.1, Definition 7600.2, and Definition 7600.3 seed the finite front-door mediation packet over CausalIntervention, ProbSpace, RandomVar, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{FrontdoorCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full do-calculus, back-door completeness, counterfactual semantics, probability-space completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mediator blocking, adjustment route, causal handoff, transport, replay, provenance, and local naming for BEDC.Derived.FrontdoorCriterionUp.

Chapter 7601

A Concrete Naming Certificate for $\mathsf{BishopRealEqualityModulus}^{\uparrow}$

Following Bishop and Bridges constructive real equality, the $\mathsf{BishopRealEqualityModulus}^{\uparrow}$ packet records the finite equality-modulus route by which two regular real names are compared before a terminal $\mathsf{Real}^{\uparrow}$ consumer reads an equality boundary. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It refines the criterion surface of chapter 6461 and the equality packet of chapter 2180 by naming the displayed modulus row that every equality read must traverse.

Definition 7601.1 (Bishop real equality-modulus carrier)**.** *A* $\mathsf{BishopRealEqualityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S_0, S_1, R_0, R_1, D, \mu, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are paired finite* $\mathsf{StreamName}^{\uparrow}$ *windows for the two regular real names being compared.*
2. R_0, R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested radii, split dyadic budgets, endpoint observations, and comparison cells used by the equality read.*
4. μ *is the displayed equality-modulus row. For every admitted tolerance request* ε *in* D*, the row supplies a finite bound after which the paired readbacks* R_0, R_1 *are compared only inside the displayed dyadic budget.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal handoff, exposed only after* $S_0, S_1, R_0, R_1, D, \mu$ *have been displayed.*
6. H *records componentwise* hsame *transport for the paired windows, readbacks, tolerance ledger, equality modulus, Real seal handoff, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto \mu \longmapsto E.$$

8. P *is the* Pkg *provenance over* $\mathsf{BishopRealEqualityModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyEqualityCriterion}^{\uparrow}$, $\mathsf{CauchyRealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealEqualityModulus}^{\uparrow}}$ *row naming exactly this finite equality-modulus packet.*

The classifier compares accepted packets only through $S_0, S_1, R_0, R_1, D, \mu, E, H, C, P, N$. It has no coordinate for quotient real equality, trichotomy, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7601.2 (Bishop real equality-modulus NameCert obligations)**.** *For every accepted* BishopRealEqualityMo *carrier*

$$M = (S_0, S_1, R_0, R_1, D, \mu, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealEqualityModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the paired StreamName windows, paired RegSeqRat readbacks, DyadicRatCore tolerance ledger, equality-modulus row, Real seal handoff, transport, replay, provenance, and local naming;*
2. *paired finite-window exposure, requiring both names to display the inspected windows before any equality-modulus answer is consumed;*
3. *regular readback exactness, requiring R_0, R_1 to be the only regular rational rows read from the paired windows;*
4. *dyadic tolerance visibility, requiring D to display the requested radii and split budgets before μ answers a comparison request;*
5. *equality-modulus exactness, requiring every admitted equality read to pass through μ before the terminal Real seal handoff E is exposed;*
6. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
7. Pkg *provenance and local naming through P and N;*
8. *nonescape from quotient real equality, trichotomy, selected limits, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Bishop real equality-modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7601.1. The accepted tuple displays exactly the paired stream windows, paired regular readbacks, dyadic tolerance ledger, equality modulus, Real seal handoff, transport, replay, provenance, and local naming rows.

The finite-window and readback obligations are read from S_0, S_1, R_0, R_1. A consumer first exposes the inspected coordinates of both names and then reads the regular rational rows obtained from those windows.

The tolerance and equality-modulus obligations are read from D and μ. The dyadic ledger displays the requested radii and split comparison budgets; the equality-modulus row supplies the finite bound that makes the paired readbacks comparable inside that displayed budget.

Finally the structural rows E, H, C, P, N seal, transport, replay, source, and name exactly the same finite route. Since the carrier contains no coordinate for quotient equality, trichotomy, selected limits, countable choice, host equality, or ambient completeness, those principles cannot enter the local NameCert surface. □

Theorem 7601.3 (Bishop real equality-modulus L10 handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing equality read exported by an accepted* $\mathsf{BishopRealEqualityModulus}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BishopRealEqualityModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

implemented by $S_0, S_1, R_0, R_1, D, \mu, E, H, C, P, N$. The route exports a finite equality-modulus certificate for later Real and Completion consumers; it exports no quotient real equality, trichotomy, selected limit, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real-facing equality read of the packet. Projection exposes the paired finite windows S_0, S_1 and their regular rational readbacks R_0, R_1.

Next read the dyadic tolerance ledger D. It fixes the requested radii, split budgets, and comparison cells before the equality-modulus row can answer the read.

Now consume μ and then E. The modulus row is downstream of the displayed windows, readbacks, and tolerance ledger; the terminal Real seal handoff is downstream of the modulus row. Transport, replay, provenance, and local naming preserve this order through H, C, P, N.

Thus every L10 equality-facing consumer sees the displayed modulus route before any Real seal is available. Because no quotient, trichotomy, selected-limit, choice, host-equality, or ambient-completeness coordinate is present in the carrier, none is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealEqualityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealEqualityModulus}^{\uparrow}$ packet is generated from paired finite StreamName windows, paired RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a displayed equality-modulus row, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealEqualityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7601.1, Definition 7601.2, and Definition 7601.3 seed the finite equality-modulus route over paired stream windows, regular rational readbacks, dyadic tolerance ledgers, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealEqualityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, trichotomy, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formalization requires carrier admission, paired finite-window exposure, regular readback exactness, dyadic tolerance visibility, equality-modulus exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealEqualityModulusUp.

Chapter 7602

A Concrete Naming Certificate for $\mathsf{RieszBasis}^{\uparrow}$

Following Young, Christensen, and constructive Hilbert-space analysis, the $\mathsf{RieszBasis}^{\uparrow}$ packet records the finite BEDC naming surface for a sequence that is read as stable Hilbert coordinates through displayed lower and upper frame bounds. It sits beside the Hilbert carrier of chapter 91, the vector and norm rows of chapter 47 and **??**, the finite orthonormal comparison row of chapter 4758, and the Cauchy-name surface of chapter 406. The packet names only the displayed coordinate reconstruction route; it does not import host Hilbert equality, a selected infinite series, ambient completeness, countable choice, or an external basis theorem.

Definition 7602.1 (RieszBasis carrier)**.** *A* $\mathsf{RieszBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (H, V, N, I, L, U, Q, \Gamma, T, C, P, Z)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, exposing only the inner-product and Hilbert-facing carrier reads used by the coordinate packet.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row over the same endpoint, recording vector addition, zero, inverse, scalar action, and finite linear-combination windows.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row induced from the Hilbert source, naming norm evaluations and norm-comparison ledgers used by the bound checks.*
4. *I is the sequence-index carrier, a finite visible row for requested index windows, coefficient locations, and truncation schedules.*
5. *L is the lower frame-bound row, carrying a displayed positive rational or real bound and the finite inequality ledgers that witness lower stability on inspected coefficient windows.*
6. *U is the upper frame-bound row, carrying a displayed finite upper bound and the finite inequality ledgers that witness bounded synthesis on the same windows.*
7. *Q is the coordinate reconstruction ledger, recording the inspected partial sums, coefficient reads, residual windows, and Cauchy-name handoff used before a reconstruction consumer may read the vector endpoint.*
8. *Γ is the local classifier, comparing accepted packets only by the shared Hilbert, vector, norm, index, frame-bound, and reconstruction rows.*

9. *T records componentwise* hsame *transport for $H, V, N, I, L, U, Q, \Gamma, C, P, Z$.*
10. *C records displayed* Cont *replay from a coordinate request through the same source, bound, classifier, and reconstruction rows.*
11. *P is the* Pkg *provenance over* $\mathsf{RieszBasis}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *Z is the local* $\mathsf{NameCert}_{\mathsf{RieszBasis}^{\uparrow}}$ *row binding the packet.*

The carrier has no coordinate for a host basis equality, a chosen infinite synthesis series, a quotient of Hilbert vectors, or an ambient completeness proof.

Theorem 7602.2 (RieszBasis NameCert obligations)**.** *For every accepted* $\mathsf{RieszBasis}^{\uparrow}$ *carrier* $R = (H, V, N, I, L, U, Q, \Gamma, T, C, P, Z)$*, the local* $\mathsf{NameCert}_{\mathsf{RieszBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, Hilbert and norm source compatibility, sequence-index exposure, lower frame-bound display, upper frame-bound display, coordinate reconstruction ledger display, classifier stability, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A coordinate consumer may inspect reconstruction only along the displayed route*

$$H, V, N, I, L, U \longrightarrow Q, \Gamma \longrightarrow T, C, P, Z.$$

It receives no host Hilbert equality, no selected infinite series, no quotient vector equality, no countable-choice selector, and no ambient completeness coordinate.

Proof. Carrier admission is projection from Definition 7602.1. The rows H, V, N expose the Hilbert, vector, and norm sources, while I exposes the only index windows on which coefficient reads may be requested.

The frame-bound rows L and U are displayed beside the same index windows. The lower row records finite stability ledgers, and the upper row records finite bounded-synthesis ledgers, so neither bound can be read as an external Hilbert-space theorem.

The reconstruction row Q is accepted only after those source and bound rows are visible. The classifier Γ compares packets by the same displayed rows, and the structural rows T, C, P, Z preserve that route under transport, replay, provenance, and local naming.

Since the carrier contains no host basis equality, selected infinite synthesis series, quotient vector equality, countable-choice selector, or ambient completeness coordinate, the listed rows exhaust the local naming surface. □

Theorem 7602.3 (RieszBasis frame-coordinate handoff)**.** *Every Hilbert-facing consumer read exported by an accepted* $\mathsf{RieszBasis}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H, the* $\mathsf{VecSpace}^{\uparrow}$ *row V, the* $\mathsf{Norm}^{\uparrow}$ *row N, the sequence-index carrier I, the lower and upper frame-bound rows L, U, the coordinate reconstruction ledger Q, the local classifier Γ, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row Z. Thus a coordinate reconstruction reaches the Hilbert source only through finite frame and coefficient ledgers, not through host Hilbert equality, a selected infinite series, quotient vector equality, countable choice, or ambient completeness.*

Proof. Begin with an accepted packet $R = (H, V, N, I, L, U, Q, \Gamma, T, C, P, Z)$. Projection by Definition 7602.1 identifies the Hilbert, vector, norm, sequence-index, frame-bound, reconstruction, classifier, transport, replay, provenance, and naming rows as the whole carrier inventory.

Use Definition 7602.2. It forces every coordinate request to pass first through the displayed source and index rows, then through the lower and upper frame-bound ledgers, and only then through the reconstruction ledger and classifier.

The structural rows keep the same finite route under hsame transport and Cont replay. Because the packet has no coordinate for host Hilbert equality, chosen infinite synthesis, quotient equality, countable choice, or ambient completeness, the Hilbert-facing read cannot leave the displayed frame-coordinate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszBasis}^{\uparrow}$ packet is generated from a Hilbert source row, vector and norm source rows, sequence-index windows, lower and upper frame-bound ledgers, a coordinate reconstruction ledger, classifier data, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7602.1, Definition 7602.2, and Definition 7602.3 seed the finite Riesz-basis certificate over $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows for later coordinate reconstruction consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert equality, selected infinite synthesis, quotient vector equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Hilbert and norm source compatibility, sequence-index exposure, lower and upper frame-bound display, coordinate reconstruction, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.RieszBasisUp.

Chapter 7603

A Concrete Naming Certificate for $\mathsf{CousinGaugeCover}^{\uparrow}$

Following Bishop–Bridges constructive integration and the Henstock gauge-cover route, the $\mathsf{CousinGaugeCover}^{\uparrow}$ packet records the finite gauge-cover data that sits between the local gauge rows of chapter 3426, the finite gauge cells of chapter 6708, and the tagged partition handoff of chapter 7044. It keeps the traditional Cousin argument as displayed NameCert data: a gauge assignment is inspected through finitely many tagged cells, each cell has a finite refinement witness, the Cousin cover selection is replayed by Cont, and every exported row remains attached to its Pkg provenance and local naming row. The packet is not an open-cover compactness principle, a hidden subcover selector, a quotient equality of covers, or an ambient real-completeness assertion.

$\mathsf{CousinGaugeCover}^{\uparrow}$ hands its finite gauge-cover rows to the theorem-level subdivision route of chapter 1861 before $\mathsf{GaugeIntegral}^{\uparrow}$ or $\mathsf{FiniteGaugeCover}^{\uparrow}$ consumers inspect tagged subdivisions.

Definition 7603.1 (Cousin gauge-cover carrier)**.** *A* $\mathsf{CousinGaugeCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, T, F, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the gauge-assignment row over the displayed interval or compact interval source.*
2. *T is the finite tagged-cell row; each entry records its tag, cell boundary, and local gauge-admission witness.*
3. *F is the finite-refinement row, recording subdivisions or shrinkages used to make each tagged cell subordinate to D.*
4. *S is the Cousin cover-selection row, read through the finite cover surfaces of* $\mathsf{CousinCover}^{\uparrow}$*,* $\mathsf{CousinFiniteCover}^{\uparrow}$*, and* $\mathsf{FiniteGaugeCover}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for D, T, F, S and the structural rows.*
6. *C records displayed* Cont *replay from a gauge-cover request through D, T, F, S, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{CousinGaugeCover}^{\uparrow}$*,* $\mathsf{CousinCover}^{\uparrow}$*,* $\mathsf{CousinFiniteCover}^{\uparrow}$*,* $\mathsf{FiniteGaugeCover}^{\uparrow}$*,* $\mathsf{CousinLemma}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinGaugeCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, T, F, S, H, C, P, N. It contains no coordinate for arbitrary open covers, selected subcovers outside the displayed finite cells, quotient cover equality, countable choice, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CousinGaugeCoverUp.CousinGaugeCoverTasteGate single carrier alignment`

Theorem 7603.2 (Cousin gauge-cover finite refinement)**.** *Let $G = (D, T, F, S, H, C, P, N)$ be an accepted* $\mathsf{CousinGaugeCover}^{\uparrow}$ *carrier. Every finite cover row exported by G factors through the displayed refinement route*

$$D, T \longmapsto F \longmapsto S, H, C, P, N.$$

Thus the cover is assembled from gauge-assigned tagged cells and finite refinement rows before the Cousin selection row is consumed. It exports no open-cover compactness theorem, no selected subcover outside T, F, no quotient equality of covers, no countable choice, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier rows D, T, F, S, H, C, P, N. The gauge row and tagged-cell row expose the only local cover data available to a consumer.

The refinement row F records the finite step that makes the tagged cells subordinate to the gauge assignment. Since F is a displayed row of the same packet, the consumer cannot replace it by an external compactness or subcover principle.

Finally the Cousin selection row S is replayed through H, C, P, N. Transport, replay, provenance, and local naming keep the exported cover attached to the same finite tagged cells, so every accepted cover read follows the displayed route. □

Theorem 7603.3 (Cousin gauge-cover NameCert obligations)**.** *For every accepted* $\mathsf{CousinGaugeCover}^{\uparrow}$ *carrier*

$$G = (D, T, F, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CousinGaugeCover}^{\uparrow}}$ *surface consists exactly of gauge-assignment admission through D, tagged-cell exposure through T, finite-refinement exactness through F, Cousin cover-selection exposure through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No local obligation is discharged by open-cover compactness, hidden subcover choice, quotient cover equality, host interval equality, countable choice, or ambient real completeness.*

Proof. Carrier admission is projection from Definition 7603.1. The accepted packet displays the gauge assignment, tagged cells, finite refinement, Cousin selection, and structural rows.

The cover-facing obligations are exhausted by D, T, F, S. Gauge admission is read from D, the finite cell witnesses are read from T, refinement is read from F, and the Cousin-facing finite cover is read from S.

The structural obligations are exhausted by H, C, P, N. Componentwise transport preserves the displayed rows, replay follows the finite route, provenance names the source packets, and the local NameCert row admits only the listed data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinGaugeCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CousinGaugeCover}^{\uparrow}$ packet is generated from gauge-assignment rows, finite tagged-cell rows, finite-refinement rows, Cousin cover-selection rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinGaugeCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7603.1, Definition 7603.2, and Definition 7603.3 seed the finite Bishop–Bridges and Henstock gauge-cover route over CousinCoverUp, CousinFiniteCoverUp, FiniteGaugeCoverUp, CousinLemmaUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CousinGaugeCover}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers outside the displayed finite cells, quotient cover equality, host interval equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge-assignment exposure, tagged-cell exactness, finite-refinement exactness, Cousin cover-selection exposure, structural replay, provenance exhaustion, and local naming for BEDC.Derived.CousinGaugeCoverUp.

Chapter 7604

A Concrete Naming Certificate for $\mathsf{CompactOscillationModulus}^{\uparrow}$

Following Bishop–Bridges compact metric analysis and the finite-net proof of uniform continuity in chapter 9237, the $\mathsf{CompactOscillationModulus}^{\uparrow}$ packet records compact oscillation control as finite NameCert data. It sits over the compact source of chapter 60, the metric inspection rows of chapter 57, the compact-metric finite-net surface of chapter 129, the finite cover nerve of chapter 7597, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer. The packet names finite compact-net oscillation bounds and the handoff from local oscillation ledgers to a tolerance-only uniform modulus; it is not an external modulus selector, an arbitrary open-cover argument, a host minimum over all points, or an ambient completeness theorem.

Definition 7604.1 (Compact oscillation-modulus carrier)**.** *A* $\mathsf{CompactOscillationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, M, V, L, F, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, carrying finite compact-net coverage data.*
2. *M is the metric observation row used to read source distances and finite ball membership.*
3. *V is the finite cover-nerve row recording the displayed compact-net cover and overlap incidence.*
4. *L is the local oscillation row, assigning to each displayed center a target oscillation bound.*
5. *F is the finite oscillation fold row, computing a compact-net-wide bound from the listed local rows.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *comparison row for the finite radii, tolerances, and oscillation bounds.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exposed only after K, M, V, L, F, R have been displayed.*
8. *H records componentwise* hsame *transport for the compact, metric, cover-nerve, oscillation, fold, rational, handoff, and structural rows.*
9. *C records displayed* Cont *replay from compact-net oscillation requests through $K, M, V, L, F, R, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationModulus}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, V, L, F, R, U, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, a host supremum of oscillations, quotient equality of covers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7604.2 (Compact oscillation-modulus NameCert obligations)**.** *For every accepted* CompactOscillationMo *carrier*

$$O = (K, M, V, L, F, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationModulus}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, metric readback through M, finite cover-nerve exposure through V, local oscillation-bound exposure through L, finite oscillation folding through F, rational comparison through R, uniform-modulus handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 7604.1. The accepted packet displays the compact source, metric observations, finite cover nerve, local oscillation rows, finite fold, rational comparisons, uniform-modulus handoff, and structural rows.

The compact and cover obligations are read through K, M, V. These rows expose finite compact-net coverage and overlap incidence, not an arbitrary open-cover compactness theorem.

The oscillation obligations are read through L, F, R. Local oscillation bounds are folded over the displayed finite net and compared by the rational row, so no host supremum or selected global minimum is available.

Finally U, H, C, P, N expose only the downstream handoff and structural replay of the same packet. This exhausts the local NameCert surface. □

Theorem 7604.3 (Compact oscillation-modulus uniform handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationModulus}^{\uparrow}$ *carrier factors through the finite route*

$$K, M, V \longmapsto L, F, R \longmapsto U, H, C, P, N.$$

The route first exposes compact-net and metric cover data, then folds local oscillation bounds over the finite cover-nerve rows, and only then exports the tolerance-only uniform-modulus handoff. It exports no arbitrary open-cover compactness theorem, selected subcover, host oscillation supremum, quotient cover equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with K, M, V. The compact source and metric rows identify the finite compact-net observations, and the finite cover nerve records the cover and overlap rows that control which local oscillation bounds can be compared.

Next read L, F, R. The local oscillation row supplies the displayed bounds, the fold row combines only the finite list, and the rational row records the tolerance and radius comparisons used by the fold.

The handoff row U is downstream of those finite rows. Componentwise transport, replay, provenance, and local naming through H, C, P, N therefore keep every accepted uniform-modulus read inside the same compact oscillation packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationModulus}^{\uparrow}$ packet is generated from compact finite-net rows, metric observation rows, finite cover-nerve rows, local oscillation-bound rows, finite oscillation folds, RatUp comparisons, UniformModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactOscillationModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7604.1, Definition 7604.2, and Definition 7604.3 seed the finite compact-net oscillation route through CompactUp, CompactMetricUp, MetricUp, FiniteCoverNerveUp, OscillationModulusUp, UniformModulusUp, RatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactOscillationModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host oscillation suprema, quotient cover equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, cover-nerve exactness, local oscillation-bound exposure, finite fold stability, uniform-modulus handoff, structural replay, provenance exhaustion, and local naming for BEDC.Derived.CompactOscillationModulusUp.

Chapter 7605

A Concrete Naming Certificate for $\mathsf{PietschDomination}^{\uparrow}$

Following Pietsch, the $\mathsf{PietschDomination}^{\uparrow}$ packet records the finite domination surface through which an absolutely summing operator request may be read inside BEDC. It sits over displayed Banach rows of chapter 90, bounded operator and norm rows of chapter 1644 and chapter 3603, finite measure rows of chapter 92, normalized probability rows of chapter 184, and terminal scalar rows of chapter 38. The packet names only the BEDC carrier rows and their NameCert obligation surface; it does not assert Hahn–Banach separation, choice of a representing probability measure, or the full Pietsch domination theorem.

Definition 7605.1 (PietschDomination carrier)**.** *A* $\mathsf{PietschDomination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, Y, T, N, A, M, P, R, H, C, Q, L)$$

with the following displayed rows:

1. *X and Y are source and target* $\mathsf{Banach}^{\uparrow}$ *rows;*
2. *T is a* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row from X to Y;*
3. *N is the associated* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T;*
4. *A is a finite absolutely-summing test ledger, listing only displayed finite families of carried source endpoints and their target norm reads;*
5. *M is a* $\mathsf{Measure}^{\uparrow}$ *row exposing the finite event and value windows used by the domination read;*
6. *P is a* $\mathsf{ProbSpace}^{\uparrow}$ *normalization row over the same measure surface;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-bound row through which the domination constant and all endpoint comparisons are read;*
8. *H records componentwise* hsame *transport for the Banach, operator, norm, measure, probability, and Real rows;*
9. *C records displayed* Cont *replay of the finite domination request;*
10. *Q is the* Pkg *provenance over* $\mathsf{PietschDomination}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{PietschDomination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed coordinates. It has no coordinate for a selected Hahn–Banach functional, an external probability measure chosen by choice, quotient equality of operators, a host domination theorem, or bridge checking.

Theorem 7605.2 (PietschDomination NameCert obligations)**.** *For every accepted* $\mathsf{PietschDomination}^{\uparrow}$ *carrier* $(X, Y, T, N, A, M, P, R, H, C, Q, L)$*, the local* $\mathsf{NameCert}_{\mathsf{PietschDomination}^{\uparrow}}$ *surface consists exactly of carrier admission, Banach endpoint exposure, bounded-operator exposure, operator-norm bound exposure, finite absolutely-summing test-window coverage, measure-window coverage, probability-normalization coverage, Real scalar-bound readback, componentwise transport, displayed replay, provenance, and local naming.*

Every domination read by a consumer must factor through

$$\mathsf{Banach}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow},\ \mathsf{OperatorNorm}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \mathsf{ProbSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

The certificate does not export Hahn–Banach separation, countable or dependent choice, a full domination theorem, selected representing measures, quotient operator equality, ambient Banach-space completeness beyond the displayed input rows, or bridge checking data.

Proof. Project the carrier coordinates. The source, target, operator, and norm data are the rows X, Y, T, N, so any operator-facing read is already routed through the existing Banach and bounded-operator packets.

The finite absolutely-summing tests are the entries of A. The measure and probability rows M, P expose only their displayed event, value, and normalization windows, while the scalar endpoint R supplies the Real-facing comparison data.

Transport, replay, provenance, and local naming are exactly H, C, Q, L. Since the carrier has no further field, a consumer cannot recover a selected Hahn–Banach witness, a choice-selected probability measure, a quotient operator equality, or a completed domination theorem from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PietschDomination}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PietschDomination}^{\uparrow}$ packet is generated from displayed Banach source and target rows, a bounded-operator row, an operator-norm row, finite absolutely-summing test windows, measure and probability rows, a Real scalar-bound row, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PietschDomination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7605.1 and Definition 7605.2 seed the finite domination packet through Banach, bounded-operator, operator-norm, MeasureUp, ProbSpaceUp, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PietschDomination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice of a representing probability measure, the full Pietsch domination theorem, quotient operator equality, Lean verification, or bridge checking.

Upgrade path: A stronger grade requires Lean definitions and checked obligations for the displayed finite domination carrier and its NameCert surface.

Chapter 7606

A Concrete Naming Certificate for $\mathsf{RegularRectangularModulus}^{\uparrow}$

Following Pringsheim, Moore–Osgood, and Bishop–Bridges, the $\mathsf{RegularRectangularModulus}^{\uparrow}$ packet records the finite rectangular modulus surface that sits between regular double Cauchy data and the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ completion handoff. It isolates the window-and-tolerance budget used by chapter 4155 before the diagonal consumer reads a seal. The packet is anchored in the uniform-modulus discipline of chapter 4158, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the completion-facing route of chapter 360, and the terminal real-name surface of chapter 38. It names the rectangular budget as finite BEDC data; it does not assert a host double limit, quotient arrays, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7606.1 (Regular rectangular modulus carrier)**.** *A* $\mathsf{RegularRectangularModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, R, Q, T, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *rectangular-window family, giving the finite row and column windows inspected for each rectangular request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for those windows, including the dyadic budget assigned to each row-column face.*
3. *R is the row-column* $\mathsf{RegSeqRat}^{\uparrow}$ *modulus row, recording the regular rational bounds obtained from the displayed windows and tolerances.*
4. *Q is the uniform rectangular comparison row inherited from the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *discipline, restricted to the finite rectangle visible in S, D, R.*
5. *T is the diagonal handoff row, available only after the rectangular surface S, D, R, Q has been displayed.*
6. *W is the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal route, read after the diagonal handoff or by a completion consumer that factors through the same rectangular surface.*
7. *H records componentwise* hsame *transport for the window family, tolerance ledger, row-column modulus, diagonal handoff, seal route, provenance, and local name.*

8. *C records displayed* Cont *replay from a consumer request through S, D, R, Q, T, W and then to the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularRectangularModulus}^{\uparrow}$, $\mathsf{RegularDoubleCauchy}^{\uparrow}$, $\mathsf{RegularCauchyUniformModulu}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRectangularModulus}^{\uparrow}}$ *row.*

No carrier coordinate stores a host double limit, quotient array, selected limiting value, countable-choice schedule, or ambient real-completeness theorem.

Theorem 7606.2 (Regular rectangular modulus obligation surface)**.** *For every accepted* RegularRectangularModulu *carrier $M = (S, D, R, Q, T, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRectangularModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, rectangular window display, dyadic tolerance accounting, row-column* $\mathsf{RegSeqRat}^{\uparrow}$ *modulus readback, uniform rectangular comparison, diagonal handoff discipline,* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must consume the rectangular surface S, D, R, Q, H, C, P, N before it may read either T or W.*

Proof. Project the accepted carrier. The rows S, D, R, Q display the finite rectangular surface: windows, tolerances, regular rational row-column bounds, and uniform comparison over the inspected rectangle.

The diagonal row T and the completion-facing row W appear after that surface in the packet inventory. Transport through H and replay through C preserve the same displayed rows for any accepted consumer request, while P and N keep the route local to the named packet.

Since the carrier has no coordinate for a host double limit, quotient array, selected limit, countable-choice schedule, or ambient completeness theorem, the listed rows exhaust the local obligation surface. □

Theorem 7606.3 (Regular rectangular modulus handoff factorization)**.** *Every* $\mathsf{RegularDoubleCauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularRectangularModulus}^{\uparrow}$ *carrier factors through the finite route*

$$S, D \longmapsto R, Q \longmapsto T \longmapsto W.$$

The diagonal handoff may refine a request to a cofinal line, but it cannot replace the displayed StreamName windows, DyadicRatCore tolerance ledger, or row-column RegSeqRat modulus surface.

Proof. Begin with the consumer request and read the rectangular rows S and D. These rows fix the inspected StreamName windows and dyadic tolerance entries before any regular rational or real-facing seal is visible.

Apply the row-column modulus row R and the uniform comparison row Q. Together they supply the finite regular rational surface that a regular double Cauchy packet or a RegSeqRat consumer may inspect.

Only then may the route read the diagonal handoff T and the completion seal W. The structural rows H, C, P, N replay and name the same path, so no consumer obtains a bypass through a selected double limit, quotient array, choice schedule, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularRectangularModulus}^{\uparrow}$ read displays StreamName rectangular windows, a DyadicRatCore tolerance ledger, row-column RegSeqRat modulus data,

a uniform comparison row, a diagonal handoff, a RealCauchyCompletion and Real seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any completion-facing consumer reads a terminal seal.

Independence witness: regular-double-cauchy, regular-cauchy-uniform-modulus, streamname, regseqrat, dyadicratcore, realcauchycompletion, real: $\mathsf{RegularRectangularModulus}^{\uparrow}$ is not bijective to any listed sibling because it names the finite rectangular modulus surface itself, separating the row-column budget from both the source double Cauchy carrier and the downstream completion seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRectangularModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRectangularModulus}^{\uparrow}$ packet is generated from finite StreamName rectangular windows, a DyadicRatCore tolerance ledger, row-column RegSeqRat modulus data, a uniform comparison row, a diagonal handoff, a RealCauchyCompletion and Real seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRectangularModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7606.1, Definition 7606.2, and Definition 7606.3 seed the finite rectangular modulus surface between regular double Cauchy data and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRectangularModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host double limits, quotient arrays, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, rectangular window display, dyadic tolerance accounting, row-column modulus readback, uniform comparison, diagonal handoff, seal routing, and local NameCert surface proofs for BEDC.Derived.RegularRectangularModulusUp.

Chapter 7607

A Concrete Naming Certificate for $\mathsf{EquivalenceRelation}^{\uparrow}$

Following Bourbaki and standard set-theoretic algebra, the $\mathsf{EquivalenceRelation}^{\uparrow}$ packet records the reusable finite classifier-law surface for reflexive, symmetric, and transitive binary rows. It sits beside the membership discipline of chapter 3668, the order-law contrast of chapter 52, the regular Cauchy comparison packet of chapter 1412, the equality-facing setoid boundary of chapter 1820, and the rational source of chapter 37. The packet gives L10 comparison chapters one named BEDC object for classifier laws; it is not a quotient carrier, a representative selector, or a host equality principle.

Definition 7607.1 (Equivalence relation carrier)**.** *An* $\mathsf{EquivalenceRelation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the carried source data on which the classifier row is displayed; it may be a set-like row, a rational-history row, or an L10 comparison source, but it is read only through its own NameCert surface;*
2. *R is the displayed binary classifier row on accepted source observations;*
3. *L stores the reflexive, symmetric, and transitive law witnesses for R on the accepted displayed rows;*
4. *H transports the source, classifier, law, replay, provenance, and local name rows componentwise by* hsame*;*
5. *C replays the same finite classifier-law packet by displayed* Cont *routes;*
6. *P is the* Pkg *provenance over* $\mathsf{EquivalenceRelation}^{\uparrow}$*, the source packet named by S,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{EquivalenceRelation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting S, selecting canonical representatives, choosing class witnesses, or importing host equality of source objects.

Theorem 7607.2 (Equivalence relation NameCert obligations)**.** *For an accepted* $\mathsf{EquivalenceRelation}^{\uparrow}$ *carrier $E = (S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EquivalenceRelation}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the source, binary classifier, law witnesses, transport, replay, provenance, and local naming rows;*
2. *law-row exactness, requiring L to expose precisely reflexivity, symmetry, and transitivity for R on accepted displayed rows;*
3. *classifier transport, requiring H and C to move the same source, classifier, and law rows without changing their public read;*
4. *non-quotient boundary, excluding quotient carriers, representative selectors, host equality, and selected equivalence classes;*
5. *downstream classifier handoff, allowing comparison packets such as regular Cauchy and rational-history readers to cite S, R, L, H, C, P, N as one finite classifier-law substrate.*

No equivalence-relation NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic data are exactly the source row S, the displayed binary classifier R, and the law row L.

The law row is read without leaving the carrier. Its three fields state reflexivity, symmetry, and transitivity only for displayed accepted rows, so the public classifier law is not a host relation on an ambient set.

The structural rows H, C, P, N transport, replay, source, and locally name the same classifier-law packet. Since no quotient, selector, class representative, or host equality row occurs in E, the listed five obligations exhaust the local NameCert surface. □

Theorem 7607.3 (Equivalence relation classifier laws)**.** *Every public classifier read of an accepted* $\mathsf{EquivalenceRelation}^{\uparrow}$ *packet exposes exactly the reflexive, symmetric, and transitive laws stored in L for the displayed classifier row R on accepted source rows from S. The exported read may be handed to downstream comparison packets through H, C, P, N, but it exports no quotient carrier, representative selector, selected equivalence class, host equality of source objects, or ambient extensionality principle.*

Proof. Begin with a public classifier read. Carrier admission forces the read to pass through S and then through the displayed binary classifier row R.

Next read the law row L. The only laws available there are reflexivity on each accepted displayed row, symmetry for accepted displayed pairs, and transitivity for accepted displayed triples.

Finally transport the read through H and replay it through C, with provenance and local naming supplied by P and N. These rows preserve the displayed classifier-law packet but do not add a quotient carrier or a representative-selection coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{EquivalenceRelation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquivalenceRelation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7607.1, Definition 7607.2, and Definition 7607.3 seed the finite classifier-law substrate for displayed equivalence rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquivalenceRelation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient carriers, representative selectors, selected equivalence classes, host equality, ambient extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, law-row exactness, classifier transport, non-quotient boundary, and downstream classifier handoff for BEDC.Derived.EquivalenceRelationUp.

Constructive story: An $\mathsf{EquivalenceRelation}^{\uparrow}$ packet is generated from carried source data, a displayed binary classifier row, reflexive/symmetric/transitive law witnesses, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7608

A Concrete Naming Certificate for $\mathsf{CriticalPairResidualStream}^{\uparrow}$

The $\mathsf{CriticalPairResidualStream}^{\uparrow}$ packet realizes the residual-join edge of chapter 9463 as finite MetaCIC-facing NameCert data. It sits beside the parallel-diamond frontier of chapter 4238, the normalization frontier of chapter 892, the same-schema certificate of chapter 1093, and the inter-Hist invariant surface of chapter 656. These siblings supply conditional local-diamond rows, candidate-normalization rows, relation-certificate scope, and history-invariance rows; this packet records a displayed stream of residual critical-pair windows without turning the stream into a global confluence theorem.

Definition 7608.1 (Critical-pair residual stream carrier)**.** *A* $\mathsf{CriticalPairResidualStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, N, L, I, R, J, W, E, H, C, P, T)$$

with the following displayed rows:

1. *D is the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *row supplying the conditional local-diamond frontier.*
2. *N is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row supplying candidate-normalization and endpoint-readback evidence.*
3. *L is the* $\mathsf{LayeredRelationCert}^{\uparrow}$ *row naming the same-schema comparison used to keep residual windows in one displayed schema.*
4. *I is the* $\mathsf{InterHistInvariant}^{\uparrow}$ *row preserving the residual stream across the inspected Hist boundaries.*
5. *R is the finite residual-window ledger, listing the critical-pair residuals under inspection.*
6. *J is the candidate-join ledger attached to the residual windows of R.*
7. *W is the stream-order row, recording the displayed predecessor relation among residual windows.*
8. *E is the failure row for windows that cannot be routed through the diamond and normalization frontiers.*
9. *H stores componentwise* hsame *transport for $D, N, L, I, R, J, W, E, C, P, T$.*
10. *C stores displayed* Cont *replay of the route from diamond-frontier evidence through normalization, relation scope, inter-Hist invariance, residual windows, and candidate joins.*

11. *P is the* Pkg *provenance over* $\mathsf{CriticalPairResidualStream}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *T is the local* $\mathsf{NameCert}_{\mathsf{CriticalPairResidualStream}^{\uparrow}}$ *row.*

The carrier has no coordinate for residual-substitution discharge, subject reduction, premise-free parallel diamond, unrestricted Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 7608.2 (Critical-pair residual stream NameCert obligations)**.** *For every accepted* CriticalPairResidualStre *carrier*

$$S = (D, N, L, I, R, J, W, E, H, C, P, T),$$

the local $\mathsf{NameCert}_{\mathsf{CriticalPairResidualStream}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed diamond-frontier row, normalization-frontier row, same-schema relation row, inter-Hist invariant row, residual-window ledger, candidate-join ledger, stream-order row, failure row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7608.1. The accepted tuple displays every row used by the residual-stream read: diamond frontier, normalization frontier, relation scope, inter-Hist invariance, residual windows, joins, order, failure, transport, replay, provenance, and name.

The positive residual read is routed through D, N, L, I, R, J, W. A consumer first exposes the conditional diamond frontier and candidate-normalization rows, then reads the same-schema and inter-Hist rows, and only then inspects the finite residual-window and candidate-join ledgers.

The failure row E remains in the same packet. Thus a window that lacks the displayed frontier, normalization, relation, or invariant support is not retyped as a successful residual join.

Finally H, C, P, T only preserve, replay, package, and name the displayed rows. Since the carrier contains no further coordinate, the NameCert surface exports no hidden confluence theorem or quotient-level equality. □

Theorem 7608.3 (Critical-pair residual stream route)**.** *Every public residual-stream read of an accepted* $\mathsf{CriticalPairResidualStream}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto \mathsf{LayeredRelationCert}^{\uparrow} \longmapsto \mathsf{InterHistInvariar}$$

The route may feed a MetaCIC parallel-diamond consumer as displayed candidate-mediated residual data, but it cannot discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with the carrier projection. The residual-window ledger R is not public until the frontier row D, normalization row N, relation row L, and inter-Hist row I have all been exposed.

The candidate joins in J are read only over the displayed windows of R, and the stream-order row W records only the finite predecessor relation among those windows. Hence the route is a finite stream of inspected residual windows rather than an ambient reduction relation.

Transport and replay through H and C preserve the same displayed route, while P and T preserve provenance and naming. No structural row adds a residual-substitution proof, subject-reduction proof, global confluence theorem, evaluator equality, quotient equality, or kernel endpoint. □

Theorem 7608.4 (Critical-pair residual stream frontier handoff)**.** *Every* MetaCICParallelDiamondFrontier$^{\uparrow}$ *consumer of an accepted* CriticalPairResidualStream$^{\uparrow}$ *packet must first read the rows* D, N, L, I, R, J, W*. The consumer may export only the displayed candidate-mediated residual data carried by the frontier row, normalization row, same-schema relation row, inter-Hist invariant row, residual-window ledger, candidate-join ledger, and stream-order row.*

It cannot discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency.

Proof. Project the carrier of Definition 7608.1. The rows D, N, L, I, R, J, W are the only rows that expose a parallel-diamond frontier, normalization frontier, same-schema relation, inter-Hist invariant, residual-window ledger, candidate-join ledger, and stream order.

Apply Definition 7608.2. The local certificate admits exactly those displayed rows together with the failure, transport, replay, provenance, and naming rows, so a frontier consumer cannot read a residual window before the frontier and normalization rows are exposed.

Now use Definition 7608.3. The route factors any public residual-stream read through D, N, L, I, R, J, W and exports only candidate-mediated residual data along that route.

The refusal boundary is preserved because none of the carrier coordinates stores substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency. Thus the AI chapter remains a residual-frontier handoff rather than a hidden MetaCIC metatheorem. □

Falsifiable prediction: If the residual-window ledger R is read without the displayed MetaCICParallelDiamondFron MetaCICNormalizationFrontier$^{\uparrow}$, LayeredRelationCert$^{\uparrow}$, and InterHistInvariant$^{\uparrow}$ rows, the packet must route the request to the failure row E rather than to a residual-stream certificate.

Independence witness: metacicparallel, metacicnormalization, layeredrelationcert, interhistinvariant: MetaCICParallelDiamondFrontier$^{\uparrow}$ supplies conditional local-diamond rows without the residual stream ledger; MetaCICNormalizationFrontier$^{\uparrow}$ supplies candidate-normalization rows without the critical-pair stream order; LayeredRelationCert$^{\uparrow}$ supplies same-schema relation scope without residual joins; InterHistInvariant$^{\uparrow}$ supplies history-invariance rows without the diamond and normalization handoff. None is bijective to CriticalPairResidualStream$^{\uparrow}$ under NameCert-row preservation.

Lean follow-up requires Nontrivial CriticalPairResidualStream$^{\uparrow}$ and FieldFaithful CriticalPairResidualStream$^{\uparrow}$ TasteGate instances before the packet can be checked as a single-carrier handoff. Those instances must preserve exactly the rows $D, N, L, I, R, J, W, E, H, C, P, T$.

Closure status for NameCert$_{\text{CriticalPairResidualStream}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{CriticalPairResidualStream}^{\uparrow}}$, seed)

Scope closed: Definition 7608.1, Definition 7608.2, Definition 7608.3, the falsifiable prediction, and the independence witness seed the finite residual-stream route over MetaCIC parallel-diamond, normalization-frontier, same-schema relation, inter-Hist invariant, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CriticalPairResidualStream}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim residual-substitution discharge, subject reduction, premise-free parallel diamond, unrestricted Church–Rosser, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-window coverage, candidate-join readback, stream-order visibility, frontier and normalization routing, relation and inter-Hist preservation, transport, replay, provenance, and local naming for BEDC.Derived.CriticalPairResidualStreamUp.

Constructive story: A CriticalPairResidualStream$^{\uparrow}$ packet is generated from MetaCICParallelDiamondFrontierUp and MetaCICNormalizationFrontierUp rows, a LayeredRelationCertUp same-schema row, an InterHistInvariantUp row, residual-window and candidate-join ledgers, a stream-order row, failure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7609

A Concrete Naming Certificate for $\mathsf{NonArchimedeanMetric}^{\uparrow}$

Following Bourbaki and standard ultrametric geometry, the $\mathsf{NonArchimedeanMetric}^{\uparrow}$ packet records the finite strong-triangle metric surface used by constructive ultrametric consumers. It sits over the metric carrier of chapter 57, the finite dyadic tolerance ledger of chapter 291, the terminal real-name boundary of chapter 38, and the observation schedule of chapter 126. The packet names the non-Archimedean comparison route as displayed BEDC data; it is not a quotient metric construction, a selected representative discipline, or a Real completeness theorem.

Definition 7609.1 (Non-Archimedean metric carrier)**.** *A* $\mathsf{NonArchimedeanMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, D, R, S, V, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare distances by finite dyadic bounds.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row reached only after the metric and dyadic rows have been displayed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the point and distance observations read by a consumer.*
5. *V is the strong-triangle ledger, recording that every inspected triple is routed through the displayed maximum-of-two-distance comparison before the third distance is public.*
6. *H records componentwise* hsame *transport for the metric, dyadic, Real-seal, stream-window, strong-triangle, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite triple request through S, M, D, V and then to R.*
8. *P is the* Pkg *provenance over* $\mathsf{NonArchimedeanMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NonArchimedeanMetric}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient metric points, selected representatives, trichotomy of distances, countable choice, host equality of reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7609.2 (Non-Archimedean metric NameCert obligations)**.** *For every accepted* $\mathsf{NonArchimedeanMetric}^{\uparrow}$ *carrier*

$$U = (M, D, R, S, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NonArchimedeanMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; metric admission through the common* $\mathsf{Met}^{\uparrow}$ *row* M*; dyadic tolerance admission through* D*; finite observation scheduling through* S*; Real-distance sealing through* R*; strong-triangle comparison through* V*; componentwise transport and replay through* H, C*; and provenance with local naming through* P, N*. No additional obligation stores a quotient metric construction, selected representatives, host real equality, countable choice, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 7609.1. The accepted packet displays exactly the metric row, dyadic tolerance ledger, Real-distance seal, stream-window schedule, strong-triangle ledger, and the structural rows.

Metric and dyadic admission are read before any strong-triangle comparison. A finite triple request first opens the observation window S, then reads the common metric row M and dyadic tolerance ledger D used to compare the three displayed distances.

The strong-triangle row is V. It records only the finite maximum comparison between the two displayed legs and the third distance, and it reaches the Real-distance seal R only after those rows are public.

Finally inspect H, C, P, N. They transport, replay, source, and name the same displayed route, while the excluded quotient, representative, trichotomy, choice, host-equality, and completeness data have no projection from the packet. □

Theorem 7609.3 (Non-Archimedean ultrametric triangle route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NonArchimedeanMetric}^{\uparrow}$ *carrier reads the strong triangle route through*

$$S \longmapsto M \longmapsto D \longmapsto V \longmapsto R.$$

The route exposes the finite observation window, the common metric row, the dyadic tolerance comparison, the ultrametric strong-triangle ledger, and the terminal Real-distance seal before any consumer treats the distance comparison as public. It exposes no quotient metric, selected representative, trichotomy branch, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the accepted carrier $U = (M, D, R, S, V, H, C, P, N)$. The only consumer-facing observation row is S, and the only metric source is M.

Use Definition 7609.2. It forces each finite triple request to read the dyadic tolerance ledger D before the strong-triangle ledger V can compare the displayed distances.

The terminal Real-facing boundary is R. Since R is reached only after S, M, D, V, a consumer cannot bypass the displayed rows by appealing to a quotient metric, a selected representative, trichotomy, choice, host equality of reals, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{NonArchimedeanMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{NonArchimedeanMetric}^{\uparrow}$ packet is generated from a MetricSpace row, a DyadicRatCore tolerance ledger, a RealUp distance seal, a StreamName observation schedule, a strong-triangle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NonArchimedeanMetric}^{\uparrow}}$, seed)

Scope closed: Definition 7609.1, Definition 7609.2, and Definition 7609.3 seed the finite non-Archimedean metric carrier, NameCert obligations, and strong-triangle route over MetricSpace, DyadicRatCore, Real, StreamName, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{NonArchimedeanMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient metric construction, selected representatives, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance routing, strong-triangle comparison, Real-distance sealing, transport, replay, provenance, and local naming for BEDC.Derived.NonArchimedeanMetricUp.

Chapter 7610

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy real name carries usable information only when its modulus data can be transferred along the displayed stream, rational-readback, and real-seal route. The $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ packet records that transfer as finite BEDC data from the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, to the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It consumes the sibling uniform-tail budget of $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ from chapter 4158 and names the non-quotient handoff that later completion consumers may inspect.

Definition 7610.1 (RegularCauchyModulusTransfer carrier)**.** *A* $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, D, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the source name windows to which the modulus data applies;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic comparison budget for those windows;*
3. *U is the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *uniform-tail row that supplies one visible regular Cauchy modulus schedule;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes the displayed windows and dyadic tolerances as a regular rational schedule;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the same readback route, with no selected limit or quotient representative;*
6. *H records componentwise* hsame *transport for S, D, U, R, E, P, N;*
7. *C records displayed* Cont *replay of the stream-window, dyadic-tolerance, modulus, readback, and real-seal rows;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusTransfer}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through the displayed rows $S, D, U, R, E, H, C, P, N$. It therefore records transfer of modulus data across the StreamName-RegSeqRat-Real route, not equality of quotient completions, choice of a representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7610.2 (RegularCauchyModulusTransfer NameCert obligations)**.** *For the carrier of Definition 7610.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusTransfer}^{\uparrow}}$ *surface consists of carrier admission for*

$$Q = (S, D, U, R, E, H, C, P, N),$$

source-window exposure through S, dyadic-tolerance exposure through D, uniform-modulus exposure through U, regular-rational readback through R, terminal real-seal exposure through E, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 7610.1. The tuple displays the stream windows, dyadic tolerance ledger, uniform-tail modulus, regular rational readback, terminal real seal, transport data, replay data, provenance, and local naming row.

The transfer route is ordered. Row S supplies the finite source windows, row D supplies the dyadic precision ledger for those windows, row U supplies the regular Cauchy tail budget, and row R consumes that visible data as regular rational readback before row E can be used as a real-seal endpoint.

The structural rows H, C, P, N preserve exactly that route. Since the carrier has no coordinate for a quotient stream, selected representative, countable-choice schedule, host equality of completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle, the listed rows are precisely the local NameCert obligations. □

Theorem 7610.3 (RegularCauchyModulusTransfer StreamName-RegSeqRat handoff)**.** *Let $Q = (S, D, U, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ *carrier. Any consumer that reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R must first expose the* $\mathsf{StreamName}^{\uparrow}$ *source windows S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *budget U; only then may it pass the readback to the* $\mathsf{Real}^{\uparrow}$ *seal row E through H, C, P, N.*

Proof. Project the accepted carrier. The readback coordinate R is part of the same finite packet as S, D, U, E, H, C, P, N, and the classifier compares packets only through those displayed rows.

Before R can be consumed, the source windows and tolerance data must be visible. The rows S and D provide that finite input, while U supplies the uniform regular Cauchy modulus that makes the readback route a transfer of displayed data rather than a hidden completion argument.

The real-seal row E is reached only through the structural rows H, C, P, N. These rows transport, replay, source, and name the same packet, so the StreamName-RegSeqRat handoff cannot be replaced by a quotient stream, selected limit, host equality of completions, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ consumer displays the $\mathsf{StreamName}^{\uparrow}$ source windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ budget, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before it claims a modulus-transfer handoff.

Independence witness: StreamName, DyadicRatCore, RegSeqRat, RegularCauchyUniformModulus, Real: $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ is not bijective to any listed sibling because it

packages the displayed transfer route between stream windows, dyadic tolerances, uniform modulus data, regular rational readback, and terminal real seal as one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ packet is generated from stream windows, dyadic tolerance rows, a regular Cauchy uniform-modulus row, regular rational readback, terminal real-seal data, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Only the seed carrier, local NameCert obligations, and StreamName-RegSeqRat handoff route in Definition 7610.1, Definition 7610.2, and Definition 7610.3 are recorded.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusTransfer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Formal closure requires a Lean carrier and obligation theorem for the displayed rows $S, D, U, R, E, H, C, P, N$.

Chapter 7611

A Concrete Naming Certificate for $\mathsf{LocatedSequenceInfimum}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ packet records the lower-envelope certificate for a bounded displayed sequence of real names. It sits beside the upper-envelope route of chapter 6098, the bounded-sequence carrier of chapter 1415, the located-supremum boundary of chapter 1427, the finite window service of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite lower-bound and upper-approximation route; it is not Dedekind completeness, an arbitrary subset infimum, trichotomy, quotient streams, a selected extremum, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7611.1 (Located sequence infimum carrier)**.** *A* $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, L, U, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row naming the bounded displayed* $\mathsf{Real}^{\uparrow}$ *sequence.*
2. *L is the lower-bound ledger, recording the finite lower rows against which every inspected sequence value is compared.*
3. *U is the upper-approximation and cut-response ledger, recording the finite request that produces an inspected sequence value within the requested dyadic tolerance above the lower envelope.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window service used by the lower and upper ledgers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected real-name windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the precision split used by the lower-bound and upper-approximation ledgers.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for the named lower-envelope output, reached only after B, L, U, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the sequence, lower ledger, upper-response ledger, window, readback, dyadic, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$B \longmapsto L \longmapsto U \longmapsto W \longmapsto R \longmapsto D \longmapsto A.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSequenceInfimum}^{\uparrow}$, $\mathsf{LocatedSequenceSupremum}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceInfimum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, U, W, R, D, A, H, C, P, N$. It has no coordinate for Dedekind completeness, arbitrary subset infima, trichotomy, quotient streams, selected extrema, host order equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7611.2 (Located sequence infimum NameCert obligations)**.** *For an accepted* $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ *carrier $I = (B, L, U, W, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceInfimum}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded sequence row, lower-bound ledger, upper-approximation response ledger, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Every lower-envelope consumer must read the displayed route

$$\mathsf{BoundedRealSequence}^{\uparrow} \longmapsto L, U \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $B, L, U, W, R, D, A, H, C, P, N$. The route exports a located lower-envelope certificate for the sequence only, with no arbitrary subset infimum, trichotomy, quotient-stream equality, selected extremum, or ambient completeness principle.

Proof. Project the carrier of Definition 7611.1. The accepted packet displays exactly the bounded sequence row, lower-bound ledger, upper-approximation ledger, finite windows, readback, dyadic tolerance, Real seal, and structural rows.

The lower-bound and upper-response ledgers are the analytic core. The row L records the finite lower comparisons for inspected values, while U records the finite response to a requested cut above the lower envelope. Both rows are read through the same W, R, D window, readback, and tolerance service.

The sibling routes supply only their displayed interfaces. $\mathsf{LocatedSequenceSupremum}^{\uparrow}$ and $\mathsf{LocatedSupremum}^{\uparrow}$ provide the dual located-envelope discipline, $\mathsf{BoundedRealSequence}^{\uparrow}$ supplies the sequence boundary, and $\mathsf{Real}^{\uparrow}$ is reached only as the seal row A.

Since the carrier contains no coordinate for arbitrary subset infima, trichotomy, quotient streams, selected extrema, host order equality, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7611.3 (Located sequence infimum Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ *packet receives the lower-envelope result only after the rows*

$$B, L, U, W, R, D, A, H, C, P, N$$

have been displayed. The handoff preserves the bounded sequence source, lower-bound ledger, upper-approximation response, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert row.*

Proof. Start with the NameCert obligations of Definition 7611.2. They identify $B, L, U, W, R, D, A, H, C, P, N$ as the full route available to a lower-envelope consumer.

The Real handoff is downstream of the finite numerical rows. A consumer reaches the seal A only after the bounded sequence row, the two located ledgers, the StreamName windows, the RegSeqRat readback, and the dyadic tolerance row have already been displayed.

The structural rows preserve the same packet. Componentwise transport, displayed replay, provenance, and local naming cannot add selected extrema, quotient streams, trichotomy, arbitrary infima, or ambient completeness, because none of those objects is a carrier coordinate. □

Falsifiable prediction: If an accepted $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ consumer obtains a lower-envelope $\mathsf{Real}^{\uparrow}$ seal without displaying the bounded sequence row, lower-bound ledger, upper-approximation response, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, transport, replay, provenance, and local NameCert rows, the packet is refuted.

Independence witness: located-sequence-supremum, bounded-real-sequence, located-supremum, streamname, regseqrat, dyadicratcore, real: the packet is not bijective to any listed sibling because it jointly stores the lower-bound ledger and upper-approximation response for one bounded sequence lower envelope.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceInfimum}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSequenceInfimum}^{\uparrow}$ packet is generated from a BoundedRealSequence row, a lower-bound ledger, an upper-approximation response ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequenceInfimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7611.1, Definition 7611.2, and Definition 7611.3 seed the finite lower-envelope route for bounded RealUp sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequenceInfimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, arbitrary subset infima, trichotomy, quotient streams, selected extrema, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-bound ledger exposure, upper-approximation response exposure, StreamName window service, RegSeqRat readback, DyadicRatCore tolerance, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSequenceInfimumUp.

Chapter 7612

A Concrete Naming Certificate for $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy representatives, the $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ packet records the finite comparison witness that two diagonal-limit representatives determine the same named real up to the displayed psame route. It concretizes the uniqueness clause of chapter 9238 through the compatibility surface of chapter 646, the observer surface of chapter 961, the Cauchy limit seal of chapter 386, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet compares paired representatives; it does not quotient streams, select a completed limit, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7612.1 (Diagonal limit uniqueness witness carrier)**.** *A* $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, K, O, S, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the paired diagonal representative rows being compared.*
2. *K is the* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *row, recording the finite compatibility comparison between the paired diagonal routes.*
3. *O is the* $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *row, recording the observer budget and inspected diagonal windows used by the comparison.*
4. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row that seals both representatives through the same displayed limit route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the paired representative reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the synchronized windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the split precision used to compare the two representatives.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after X, Y, K, O, S, W, R, D have been displayed.*
9. *H records componentwise* hsame *transport for the paired representatives, compatibility, observer, seal, window, readback, dyadic, Real, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$X, Y \longmapsto K \longmapsto O \longmapsto S \longmapsto W \longmapsto R \longmapsto D \longmapsto A.$$

11. *P is the* Pkg *provenance over* $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, $\mathsf{DiagonalLimitObserver}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K, O, S, W, R, D, A, H, C, P, N$. It has no coordinate for quotient stream equality, a selected completed limit, host equality of representatives, countable choice, or ambient Real completeness.

Theorem 7612.2 (Diagonal limit uniqueness witness NameCert obligations)**.** *For an accepted* $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ *carrier $U = (X, Y, K, O, S, W, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitU}}$ *surface consists exactly of carrier admission for the paired representatives, compatibility row, observer row, CauchyLimitSeal row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Every uniqueness read must route the paired representatives through

$$\mathsf{DiagonalLimitCompatibility}^{\uparrow} \longmapsto \mathsf{DiagonalLimitObserver}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \mapsto$$

before it exposes a psame *comparison. The route is a finite witness packet and exports no quotient stream equality, selected completed limit, host representative equality, countable choice, or ambient completion principle.*

Proof. Project the carrier of Definition 7612.1. The accepted packet displays exactly the paired representatives, compatibility row, observer row, Cauchy limit seal, finite windows, readback, dyadic tolerance, Real seal, and structural rows.

The compatibility and observer rows are read before the seal. The row K records the finite comparison route for the two diagonal representatives, while O records the observer windows and budgets used by that comparison.

The seal and numerical rows are downstream. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ row S binds the representatives to one displayed limit route, and W, R, D, A provide the finite window, regular readback, dyadic tolerance, and Real seal evidence for the psame comparison.

The carrier has no coordinate for quotient streams, selected completed limits, host representative equality, countable choice, or ambient Real completeness. Therefore the displayed rows exhaust the local NameCert obligation surface. □

Theorem 7612.3 (Diagonal limit uniqueness Real seal)**.** *Every Real-facing comparison exported by an accepted* $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed rows*

$$X, Y, K, O, S, W, R, D, A, H, C, P, N.$$

The resulting comparison is the finite paired diagonal-representative route through compatibility, observer, CauchyLimitSeal, StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, Pkg, *and local NameCert data.*

Proof. Begin with the NameCert obligations of Definition 7612.2. They identify the full route that any uniqueness read may consume.

The Real seal A is reached after the paired representative comparison has passed through K, O, S, W, R, D. Thus the comparison is a sealed finite route rather than equality of host streams or selected completed limits.

Transport, replay, provenance, and local naming preserve the same displayed rows. Since the carrier lacks quotient-stream equality, selected limits, countable choice, host equality, and ambient completeness coordinates, the Real seal cannot export those objects. □

Falsifiable prediction: If a diagonal-limit uniqueness consumer compares two representatives without displaying the compatibility row, observer row, CauchyLimitSeal row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, replay, provenance, and local NameCert rows, it is not a $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ packet.

Independence witness: diagonal-limit-compatibility, diagonal-limit-observer, cauchy-limit-seal, streamname, regseqrat, dyadicratcore, real: the packet is not bijective to any listed sibling because it jointly stores the paired representatives and their finite uniqueness witness route.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}$ packet is generated from paired diagonal representatives, a DiagonalLimitCompatibility row, a DiagonalLimitObserver row, a CauchyLimitSeal row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7612.1, Definition 7612.2, and Definition 7612.3 seed the finite uniqueness-witness route for paired diagonal-limit representatives.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalLimitUniquenessWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completed limits, countable choice, host representative equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compatibility exposure, observer exposure, CauchyLimitSeal exposure, StreamName window service, RegSeqRat readback, DyadicRatCore tolerance, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DiagonalLimitUniquenessWitnessUp.

Chapter 7613

A Concrete Naming Certificate for $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$ packet records the finite certificate surface for the classical theorem that a displayed derivative has the Darboux property. It sits over the difference-quotient discipline of chapter 122, the target-value handoff of chapter 3348, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38. The packet names the derivative-Darboux route as finite NameCert data rather than as host differentiability semantics, quotient equality of functions, a selected intermediate point, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7613.1 (Darboux derivative property carrier)**.** *A* $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, Q, L, U, Y, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval source row, exposing the lower and upper endpoint reads and the finite orientation data used by derivative requests.*
2. *F is the function-source row whose local difference-quotient reads are inspected only through the derivative packet.*
3. *Q is the* $\mathsf{Derivative}^{\uparrow}$ *quotient row, carrying the displayed nonzero-step quotient and metric-distance ledger for derivative observations.*
4. *L and U are the two derivative-value rows read at the endpoint-side observations of I.*
5. *Y is the target derivative value displayed between the endpoint-side derivative rows.*
6. *S records the finite endpoint-side comparison data that place Y between L and U.*
7. *R is the* $\mathsf{DarbouxProperty}^{\uparrow}$ *handoff row reached after the derivative quotient route has exposed the target-value bracket.*
8. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the derivative quotient windows and requested dyadic tolerances.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal exposed only after Q, R, T have been displayed.*

10. *H stores componentwise* hsame *transport for the interval source, function source, derivative quotient, endpoint derivative values, target value, comparison data, Darboux handoff, readback, real seal, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a derivative-Darboux request through $I, F, Q, L, U, Y, S, R, T, E$.*
12. *P is the* Pkg *provenance over* $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxDerivativeProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, Q, L, U, Y, S, R, T, E, H, C, P, N$. It has no coordinate for host differentiability semantics, quotient equality of functions, selected intermediate points, real trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7613.2 (Darboux derivative property NameCert obligations)**.** *For every accepted* DarbouxDerivativeProp *carrier $D = (I, F, Q, L, U, Y, S, R, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxDerivativeProperty}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed packet of Definition 7613.1; interval-source exposure through I; derivative quotient exposure through F, Q; endpoint-side derivative value and target-value exposure through L, U, Y, S; DarbouxProperty handoff through R; regular rational and dyadic readback through T and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance surface named by P; Real-seal exposure through E; componentwise transport through H; displayed continuation replay through C; provenance through P; local naming through N; and non-escape from host differentiability semantics, quotient equality of functions, selected intermediate points, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7613.1. The interval row I, function row F, and derivative quotient row Q expose the only derivative-source data, while L, U, Y, S expose the finite endpoint-side bracket for the requested target derivative value.

The Darboux handoff is the row R. It can be read only after the displayed quotient route and the target-value comparison data have been exposed, so the packet separates the derivative route from the continuous-image Darboux route of chapter 3702.

The readback rows are then T and E. The row T supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback against the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data named by provenance, and E is the terminal $\mathsf{Real}^{\uparrow}$ seal for derivative-Darboux consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier has no coordinate for host differentiability semantics, quotient equality of functions, selected intermediate points, trichotomy, open-cover compactness, or ambient Real completeness, none of those objects can enter the local NameCert surface. □

Theorem 7613.3 (Darboux derivative property Darboux handoff)**.** *Every accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *consumer reading a* $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$ *carrier factors through the finite derivative route*

$$\mathsf{Derivative}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{DarbouxProperty}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

In particular, the consumer first reads the interval source, function source, derivative quotient row, endpoint-side derivative values, target value, and comparison data; then the regular rational and dyadic tolerance readback; and only then the $\mathsf{DarbouxProperty}^{\uparrow}$ *handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start from an accepted carrier $D = (I, F, Q, L, U, Y, S, R, T, E, H, C, P, N)$. The continuation row C exposes derivative-Darboux reads only through the interval source, function source,

quotient row, endpoint-side values, target value, and finite comparison data before any Darboux-facing handoff is available.

The derivative route is forced by Q, T, E. The carrier contains no alternate route from the function source to the target-value handoff, so a consumer cannot replace the displayed quotient windows by host differentiability semantics or quotient equality of functions.

After the derivative quotient and readback rows are visible, the consumer reaches R as the $\mathsf{DarbouxProperty}^{\uparrow}$ handoff. Transport through H and replay through C preserve the same route, while P, N only package and name it.

Thus every accepted Darboux-property read factors through the derivative quotient, regular readback, dyadic tolerance, and real-seal rows, with selected intermediate points, trichotomy, open-cover compactness, and ambient Real completeness remaining outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxDerivativeProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxDerivativeProperty}^{\uparrow}$ packet is generated from an interval source, a function source, a Derivative quotient row, endpoint-side derivative values, target-value comparison data, a DarbouxProperty handoff, RegSeqRat readback, DyadicRatCore tolerance data, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxDerivativeProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7613.1, Definition 7613.2, and Definition 7613.3 seed the derivative-Darboux property as a finite quotient-readback route from displayed derivative observations to DarbouxProperty handoff and Real-seal exposure.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxDerivativeProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability semantics, quotient equality of functions, selected intermediate points, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative quotient exposure, endpoint-side comparison, DarbouxProperty handoff, RegSeqRat and DyadicRatCore readback, Real-seal exposure, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.DarbouxDerivativePropertyUp.

Chapter 7614

A Concrete Naming Certificate for $\mathsf{LipschitzFreeSpace}^{\uparrow}$

Following Weaver and Arens–Eells, the $\mathsf{LipschitzFreeSpace}^{\uparrow}$ packet records the finite molecule-and-norm surface used when pointed metric data is linearized into a normed space. It reads the metric carrier of chapter 57, the Lipschitz seminorm surface of chapter 7502, the function-action route of chapter 3762, and the Arens–Eells norm carrier of chapter 5855. The packet names only the displayed finite molecule ledger and its universal linearization route; it does not assert Banach completion, dual-space representation, quotient completion equality, selected representatives, or a host free Banach space.

Definition 7614.1 (Lipschitz free space carrier)**.** *A* $\mathsf{LipschitzFreeSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, B, U, D, Z, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the pointed* $\mathsf{Met}^{\uparrow}$ *row, including the finite point list and the distinguished basepoint row.*
2. *B records the basepoint anchor used to interpret molecule sums as pointed metric data.*
3. *U is the finitely supported molecule row, listing formal point generators and their support.*
4. *D is the signed dyadic coefficient row for the finite molecule support.*
5. *Z is the zero-sum ledger, recording that the signed dyadic coefficients sum to zero before the molecule is admitted.*
6. *L is the* $\mathsf{LipschitzSpace}^{\uparrow}$ *seminorm-budget row for scalar Lipschitz tests on the pointed metric rows.*
7. *A is the* $\mathsf{ArensEellsSpace}^{\uparrow}$ *norm-witness row, exposing the finite decomposition budget used to read the Arens–Eells norm.*
8. *E is the evaluation row through* $\mathsf{LipschitzFunction}^{\uparrow}$*, recording how a Lipschitz function reads a molecule by the displayed finite coefficient ledger.*
9. *H records componentwise* hsame *transport for $M, B, U, D, Z, L, A, E, C, P, N$.*
10. *C records displayed* Cont *replay from pointed metric rows through molecules, zero-sum admission, Lipschitz seminorm budgets, Arens–Eells norm witnesses, and evaluation by Lipschitz functions.*

11. *P is the* Pkg *provenance over* $\mathsf{LipschitzFreeSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzSpace}^{\uparrow}$, $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzFreeSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for Banach completion, a selected completion representative, a dual-space equality theorem, an ambient quotient of measures, or an unrestricted extension principle.

Theorem 7614.2 (Lipschitz free space NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzFreeSpace}^{\uparrow}$ *carrier*

$$F = (M, B, U, D, Z, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzFreeSpace}^{\uparrow}}$ *surface consists exactly of pointed metric exposure through M, B; finitely supported molecule exposure through U; signed dyadic coefficient exposure through D; zero-sum admission through Z; Lipschitz seminorm budgeting through L; Arens–Eells norm witnessing through A; Lipschitz-function evaluation through E; componentwise transport and replay through H, C; and provenance with local naming through P, N.*

Proof. Project the carrier of Definition 7614.1. The visible mathematical rows are the pointed metric rows, the molecule support rows, the signed dyadic coefficient row, the zero-sum ledger, the seminorm budget, the Arens–Eells norm witness, and the Lipschitz-function evaluation row.

The molecule side is read first. A consumer opens M, B, U, D and may admit a molecule only after the zero-sum ledger Z has displayed the signed dyadic coefficient balance.

The norm and evaluation reads are then forced through L, A, E. The Lipschitz seminorm budget supplies the scalar test boundary, the Arens–Eells row supplies the finite norm witness, and the Lipschitz-function row records the evaluation of the same molecule support.

Finally H, C, P, N only transport, replay, source, and name this displayed route. Since the carrier contains no completion, quotient-measure, dual-space, or representative-selection coordinate, those objects are not local NameCert obligations. □

Theorem 7614.3 (Lipschitz free space universal linearization)**.** *Every accepted* $\mathsf{LipschitzFreeSpace}^{\uparrow}$ *packet exports a universal linearization route only through the finite spine*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{LipschitzSpace}^{\uparrow} \longmapsto \mathsf{LipschitzFunction}^{\uparrow} \longmapsto \mathsf{ArensEellsSpace}^{\uparrow} \longmapsto \mathsf{NormedSpace}^{\uparrow}.$$

For a carrier $F = (M, B, U, D, Z, L, A, E, H, C, P, N)$, any linear read must preserve the pointed metric rows, the finitely supported molecule row, the signed dyadic coefficient row, the zero-sum ledger, the Lipschitz seminorm budget, the Arens–Eells norm witness, and componentwise hsame *transport. It may treat the resulting packet as a normed-space-facing molecule certificate, but it may not claim Banach completion, dual identification, quotient-measure equality, or a selected free-space representative.*

Proof. Begin with the sibling dependency rows named in the chapter introduction. The metric chapter supplies pointed distance data, the Lipschitz space and Lipschitz function chapters supply seminorm and evaluation reads, and the Arens–Eells chapter supplies the finite molecule norm witness.

Next read the carrier rows M, B, U, D, Z, L, A, E. The zero-sum ledger turns signed dyadic coefficients into admitted molecules, while the seminorm and Arens–Eells rows keep the norm read tied to finite tests and finite decompositions.

The structural rows H, C, P, N preserve that same route under componentwise transport, displayed replay, provenance, and local naming. Therefore the exported universal property is only the finite BEDC linearization spine shown above, not a completed Banach-space or dual-space theorem. □

Falsifiable prediction: If an accepted $\mathsf{LipschitzFreeSpace}^{\uparrow}$ packet changes its pointed metric row, signed dyadic coefficient row, zero-sum ledger, Lipschitz seminorm budget, or Arens–Eells norm witness while retaining the same local name, then it is not the same finite NameCert packet.

Independence witness: metric, lipschitz-space, lipschitz-function, arens-eells-space, normed-space: the metric packet supplies pointed distance rows without molecules; the Lipschitz-space packet supplies seminorm budgets without zero-sum molecules; the Lipschitz-function packet supplies evaluation rows without the free carrier; the Arens–Eells packet supplies norm witnesses without this universal linearization route; and the normed-space packet supplies linear norm rows without pointed metric provenance. None is bijective to $\mathsf{LipschitzFreeSpace}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzFreeSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzFreeSpace}^{\uparrow}$ packet is generated from pointed Metric rows, basepoint rows, finitely supported molecule rows, signed dyadic coefficient rows, a zero-sum ledger, Lipschitz seminorm budgets, Arens–Eells norm witnesses, Lipschitz-function evaluation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzFreeSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7614.1, Definition 7614.2, Definition 7614.3, the falsifiable prediction, and the independence witness seed the finite pointed-molecule carrier and universal linearization route over Metric, LipschitzSpace, LipschitzFunction, ArensEellsSpace, NormedSpace, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzFreeSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Banach completion, dual-space representation, quotient-measure equality, selected free-space representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-sum molecule admission, signed dyadic coefficient routing, Lipschitz seminorm budgeting, Arens–Eells norm witnessing, evaluation readback, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzFreeSpaceUp.

Chapter 7615

A Concrete Naming Certificate for $\mathsf{ContractionRealModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Banach's contraction principle, the $\mathsf{ContractionRealModulus}^{\uparrow}$ packet records the finite modulus data by which a displayed contraction on a located interval or complete metric row yields a Cauchy route toward a real-name seal. It sits between the interval contraction surface of chapter 5867, the fixed-point modulus surface of chapter 4188, the theorem-level contraction route of chapter 3931, and the real-completion rows of chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a finite contraction-modulus carrier; it is not an ambient completeness theorem, a selected fixed point, quotient equality of real names, countable choice, or trichotomy.

Definition 7615.1 (Contraction real modulus carrier)**.** *A* $\mathsf{ContractionRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, X, F, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the source located interval, located metric interval, or complete metric row. It gives the domain on which the contraction is read, using only the displayed* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{CompleteMetric}^{\uparrow}$ *surface supplied by the surrounding chapter.*

2. *X is the initial point or initial interval row. It names the finite source datum from which the contraction iteration is inspected.*

3. *F is the contraction self-map row. It records the displayed action of the map on the source row and the finite Lipschitz witness λ with the contraction comparison $\lambda < 1$ available only as a row of the packet.*

4. *Q is the dyadic shrinkage ledger. It records the finite sequence of dyadic radii, slack inequalities, and comparison budgets used to read the contraction decrease.*

5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected iterates.*

6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic budgets.*

7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after the source row, self-map row, dyadic shrinkage ledger, finite windows, and regular rational readback have been displayed.*

8. *H records componentwise* hsame *transport for the source row, initial row, self-map, shrinkage ledger, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a contraction-modulus request through $L, X, F, Q, W, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* ContractionRealModulus$^{\uparrow}$, BishopIntervalContraction$^{\uparrow}$, BanachFixedPointModulus$^{\uparrow}$, ContractionMappingTheorem$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ContractionRealModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, X, F, Q, W, R, E, H, C, P, N$. It has no coordinate for a chosen fixed point, quotient equality of streams or real names, countable choice, trichotomy, or ambient completeness outside the displayed source row.

Theorem 7615.2 (Contraction real modulus NameCert obligations)**.** *For every accepted* ContractionRealModulus$^{\uparrow}$ *carrier $M = (L, X, F, Q, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{ContractionRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the source located or complete metric row, the initial datum, the contraction self-map, the dyadic shrinkage ledger, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The obligation surface may cite the interval-contraction source of chapter 5867, the fixed-point modulus row of chapter 4188, the contraction theorem route of chapter 3931, and the real-name dependencies of chapter 291, chapter 126, chapter 301, and chapter 38. It cannot add a selected fixed point, quotient stream equality, countable choice, trichotomy, host real equality, or a detached completeness principle.*

Proof. Project the accepted carrier. The analytic coordinates are exactly the source row L, initial row X, contraction self-map F, dyadic shrinkage ledger Q, finite stream windows W, regular rational readback R, and terminal Real seal E.

The NameCert obligations follow the displayed order. The self-map row cannot be read without L and X, the shrinkage ledger cannot be consumed without F, and the real-name handoff cannot inspect E before Q, W, R have all been displayed.

Transport and replay are structural. The row H moves every listed coordinate componentwise, while C repeats the same finite route for later contraction or fixed-point consumers.

Finally P and N source and name only that finite packet. Since the excluded objects are not carrier coordinates and are not produced by componentwise transport or replay, no local NameCert obligation exports them. □

Theorem 7615.3 (Contraction real modulus Cauchy Real handoff)**.** *Every accepted* ContractionRealModulus$^{\uparrow}$ *carrier $M = (L, X, F, Q, W, R, E, H, C, P, N)$ gives any Cauchy or Real-facing consumer the finite route*

$$L, X \longrightarrow F \longrightarrow Q \longrightarrow W \longrightarrow R \longrightarrow E$$

with structural rows H, C, P, N attached. Thus the consumer receives a contraction-modulus route through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *before it can invoke the Banach fixed-point or contraction-mapping surface; it receives no selected fixed point, quotient stream, countable-choice schedule, trichotomy row, host real equality, or ambient completeness row outside the displayed carrier.*

Proof. Start with the carrier projection of Definition 7615.1. The only route from the source row to a Real-facing read is through the contraction self-map, dyadic shrinkage ledger, finite stream windows, regular rational readback, and Real seal.

The dyadic ledger Q controls the decrease row before the windows are opened. Once W is fixed, the regular rational row R reads the inspected iterates with the same budgets, and E seals only that displayed route.

Use the structural rows H, C, P, N to transport, replay, source, and locally name the same path. Therefore downstream uses by $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{BanachFixedPointConstructive}^{\uparrow}$, or $\mathsf{ContractionMappingTheorem}^{\uparrow}$ can cite only the finite contraction-modulus handoff and not any excluded ambient principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractionRealModulus}\uparrow}$.

Constructive story: A $\mathsf{ContractionRealModulus}^{\uparrow}$ packet is generated from a displayed source row, initial datum, contraction self-map, dyadic shrinkage ledger, finite StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractionRealModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7615.1, Definition 7615.2, and Definition 7615.3 seed the finite contraction-modulus carrier over displayed source, self-map, dyadic shrinkage, StreamName, RegSeqRat, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractionRealModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, quotient stream equality, countable choice, trichotomy, host real equality, ambient completeness beyond the displayed source row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map exactness, dyadic shrinkage accounting, StreamName window readback, RegSeqRat handoff, Real-seal routing, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.ContractionRealModulusUp.

Chapter 7616

A Concrete Naming Certificate for $\mathsf{LocatedMetricBall}^{\uparrow}$

Following Bishop and Bridges on constructive metric spaces and located subsets, the $\mathsf{LocatedMetricBall}^{\uparrow}$ packet records the finite ball window that a metric consumer may inspect before any total-boundedness, compact-net, or uniform-continuity argument reads open-ball data. It sits over the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, and the regular Cauchy ball-basis sibling of chapter 6762. The packet names a located center-radius-radius-transport surface; it does not assert open-cover compactness, selected centers outside the displayed packet, quotient metric equality, trichotomy of distances, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7616.1 (LocatedMetricBall carrier)**.** *A* $\mathsf{LocatedMetricBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, X, C, R, Q, L, T, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, recording the displayed metric relation and the finite distance windows available to the packet.*
2. *X is the inspected point row, naming the point or finite point-window whose membership in the ball is being read.*
3. *C is the center row, naming the displayed center and its transport data inside the same metric source.*
4. *R is the positive radius row, read through* $\mathsf{Real}^{\uparrow}$ *and carried only as a displayed real-name seal.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that supplies rational tolerance windows for the radius and distance comparison.*
6. *L is the located comparison row, recording the finite alternatives $d_M(X, C) < R$ and $R < d_M(X, C)$ at the requested precision.*
7. *T is the radius-transport row, recording how the same center and radius are preserved when the metric source is transported by* hsame.
8. *H records componentwise* hsame *transport for $M, X, C, R, Q, L, T, P, N$.*
9. *K records displayed* Cont *replay of the metric, point, center, radius, rational-readback, located-comparison, and radius-transport rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedMetricBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyBallBasis}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricBall}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, X, C, R, Q, L, T, H, K, P, N$. It therefore records one located metric-ball packet, not a host open set, a completed metric space, a quotient distance equality, a selected finite cover, or an ambient compactness theorem.

Theorem 7616.2 (LocatedMetricBall NameCert obligations)**.** *For every accepted* $\mathsf{LocatedMetricBall}^{\uparrow}$ *carrier*

$$B = (M, X, C, R, Q, L, T, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMetricBall}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, inspected point row, center row, real-radius row, regular rational readback row, located comparison row, radius-transport row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7616.1. The tuple exposes the metric source M, inspected point X, center C, radius seal R, rational readback Q, located comparison L, radius transport T, structural transport H, replay K, provenance P, and local naming N.

The metric ball is read only after the distance and radius data are visible. Row M supplies the distance windows, C and X supply the two endpoints of the comparison, R supplies the real radius, and Q supplies the rational tolerance windows used to inspect that comparison.

The locatedness row L is the only public membership boundary. It records the displayed comparison alternatives at the requested precision and does not create a host open set or a quotient-level distance judgment.

The rows T, H, K, P, N preserve, replay, source, and name the same finite packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7616.3 (LocatedMetricBall radius transport)**.** *Let $B = (M, X, C, R, Q, L, T, H, K, P, N)$ be an accepted* $\mathsf{LocatedMetricBall}^{\uparrow}$ *carrier. Any consumer that transports the ball along an* hsame *comparison must preserve the metric source row M, center row C, real-radius row R, regular rational readback row Q, and located comparison row L before it may read the transported radius row T as the same located ball.*

Proof. Project the accepted carrier. The radius-transport row T is part of the same packet as the metric, center, radius, rational-readback, and located-comparison rows.

Transport first preserves the metric source and center. Without the displayed M and C rows, the expression $d_M(X, C)$ has no public metric-ball source inside the packet.

The radius and comparison must also be preserved. The row R supplies the real-name radius, Q supplies the rational readback used to compare it with the distance window, and L records the located comparison boundary.

Only after those rows are visible can T be consumed as radius transport for the same ball. The structural rows H, K, P, N do not add a selected center, quotient metric equality, compact cover, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{LocatedMetricBall}^{\uparrow}$ consumer displays the $\mathsf{Met}^{\uparrow}$ metric source, inspected point, center, $\mathsf{Real}^{\uparrow}$ radius seal, $\mathsf{RegSeqRat}^{\uparrow}$ rational readback, located comparison row, radius-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before it claims a located metric-ball read.

Independence witness: metric, real, regseqrat, regular-cauchy-ball-basis: $\mathsf{Met}^{\uparrow}$ supplies the distance source without a located center-radius packet; $\mathsf{Real}^{\uparrow}$ supplies terminal real seals without metric membership; $\mathsf{RegSeqRat}^{\uparrow}$ supplies rational readback without a ball classifier; $\mathsf{RegularCauchyBallBasis}^{\uparrow}$ supplies a ball-basis route without this single located point-center-radius transport surface. None is bijective to $\mathsf{LocatedMetricBall}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetricBall}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedMetricBall}^{\uparrow}$ packet is generated from a metric source row, inspected point row, center row, real-radius seal, regular rational readback, located comparison, radius transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMetricBall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7616.1, Definition 7616.2, and Definition 7616.3 seed the finite located ball surface over Metric, Real, RegSeqRat, regular Cauchy ball-basis, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMetricBall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite covers, quotient metric equality, trichotomy of distances, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located comparison exactness, radius transport, metric-source preservation, rational-readback preservation, transport, replay, provenance, and local naming for BEDC.Derived.LocatedMetricBallUp.

Chapter 7617

A Concrete Naming Certificate for $\mathsf{CandidateFrontierQueue}^{\uparrow}$

The $\mathsf{CandidateFrontierQueue}^{\uparrow}$ packet realizes the automated theory-discovery calculus of chapter 9410 as finite source-state, classifier-shift, score-budget, queue, ledger, replay, provenance, and NameCert rows. It sits beside the candidate frontier of chapter 5994, the before-and-after classifier residue of chapter 6985, the vision-coverage ledger of chapter 7484, and the critical-path source of chapter 765. The packet names an admission queue for candidate theory work; it does not by itself certify positive discovery, semantic novelty, theorem soundness, or bridge checking.

Definition 7617.1 (CandidateFrontierQueue carrier)**.** *A* $\mathsf{CandidateFrontierQueue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, A, F, D, B, U, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the certified source-state row, recording the current theory state, accepted theorem rows, and active boundary conditions read from the discovery calculus.*
2. *A is the automation-record row, naming the generator, admissible moves, replay seed, and bounded search surface used to propose candidates.*
3. *F is the* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *row, supplying candidate-frontier evidence rather than an evidence-free theorem.*
4. *D is the* $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ *row, exposing the before-and-after classifier comparison used to decide whether the queue entry can leave record growth.*
5. *B is the score-budget row, recording public cost, score, and resource bounds for the queued candidate.*
6. *U is the queue-order row, recording the displayed priority, dependency predecessor, and admission status of each candidate.*
7. *L is the ledger row, recording vision coverage, classifier-shift status, score account, replay obligations, and candidate rejection reasons.*
8. *R is the replay row, recording the deterministic route by which the source state, automation record, frontier row, classifier-shift row, budget, and queue order are rechecked.*
9. *H stores componentwise* hsame *transport for $S, A, F, D, B, U, L, R, C, P, N$.*

10. *C records displayed* Cont *replay from the source state through automation, frontier, classifier-shift, score, queue, ledger, and replay rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CandidateFrontierQueue}^{\uparrow}$, $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$, ClassifierShiftLedgerRes VisionCoverageLedger$^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateFrontierQueue}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, B, U, L, R, H, C, P, N$. It therefore records finite queue admission for candidate-frontier work, not a host search process, an unledgered discovery verdict, a hidden semantic novelty score, or unchecked theorem soundness.

Theorem 7617.2 (CandidateFrontierQueue NameCert obligations)**.** *For every accepted* $\mathsf{CandidateFrontierQueue}^{\uparrow}$ *carrier*

$$Q = (S, A, F, D, B, U, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CandidateFrontierQueue}^{\uparrow}}$ *surface consists exactly of source-state admission, automation-record admission, candidate-frontier exposure, classifier-shift ledger exposure, score-budget exposure, queue-order exposure, ledger exposure, replay exposure, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7617.1. The accepted tuple displays the source state S, automation record A, candidate frontier F, classifier-shift ledger D, score budget B, queue order U, ledger L, replay row R, transport H, continuation replay C, provenance P, and local naming N.

The queue read starts from the source state and automation record. These rows fix which candidate was proposed and which bounded search surface produced it.

The frontier and classifier rows then decide admissibility. Row F keeps the candidate attached to a displayed frontier, while row D records the before-and-after classifier evidence required by the discovery calculus.

The budget, queue, ledger, and replay rows make the admission auditable. Since the carrier contains no further coordinate for semantic novelty, theorem soundness, or positive-discovery certification, the listed rows exhaust the local NameCert obligations. □

Theorem 7617.3 (CandidateFrontierQueue classifier-shift nonescape)**.** *Let $Q = (S, A, F, D, B, U, L, R, H, C, P, N)$ be an accepted* $\mathsf{CandidateFrontierQueue}^{\uparrow}$ *carrier. A queued candidate cannot be read as discovery-class work unless the same packet exposes the* $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ *row D, the score-budget row B, the ledger row L, and the replay row R over the source-state and automation rows S, A.*

Proof. Project the accepted carrier. The queue-order row U only records priority and dependency placement; it does not decide claim strength.

The source-state and automation rows S, A fix the candidate's origin. The candidate-frontier row F supplies admissible frontier evidence, but it still does not show classifier change.

Classifier-shift evidence must therefore be read through D. The score-budget row B records the public cost account, the ledger row L records the coverage and rejection boundary, and the replay row R records the deterministic recheck route over the same source and automation surface.

If any of D, B, L, R is absent, the queue entry remains record growth or candidate work inside this packet. The structural rows H, C, P, N preserve that boundary and do not create a hidden discovery verdict. □

Falsifiable prediction: A $\mathsf{CandidateFrontierQueue}^{\uparrow}$ packet that admits a candidate as discovery-class work while omitting the displayed $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ row, score-budget row, ledger

row, or replay row over the same source-state and automation rows fails the carrier of Definition 7617.1.

Independence witness: normalization-candidate-frontier, classifier-shift-ledger-residue, vision-coverage-ledger, metaciccriticalpath: $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ supplies candidate evidence without queue order; $\mathsf{ClassifierShiftLedgerResidue}^{\uparrow}$ supplies before-and-after classifier residue without score-budgeted queue admission; $\mathsf{VisionCoverageLedger}^{\uparrow}$ supplies coverage accounting without replay ordering; $\mathsf{MetaCICCriticalPath}^{\uparrow}$ supplies dependency pressure without classifier-shift queue discharge. None is bijective to $\mathsf{CandidateFrontierQueue}^{\uparrow}$ under NameCert-row preservation.

The Lean-side draft for $\mathsf{CandidateFrontierQueue}^{\uparrow}$ must include $\mathsf{Nontrivial\ CandidateFrontierQueue}^{\uparrow}$ and $\mathsf{FieldFaithful\ CandidateFrontierQueue}^{\uparrow}$ instances for the displayed carrier, with the field-faithful instance preserving exactly the rows $S, A, F, D, B, U, L, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateFrontierQueue}^{\uparrow}}$.

Constructive story: A $\mathsf{CandidateFrontierQueue}^{\uparrow}$ packet is generated from source-state rows, automation-record rows, candidate-frontier evidence, classifier-shift residue, score budgets, queue order, ledger rows, replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateFrontierQueue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7617.1, Definition 7617.2, and Definition 7617.3 seed the finite candidate-frontier queue over discovery-calculus, normalization-frontier, classifier-shift, vision-coverage, critical-path, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateFrontierQueue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive-discovery certification, theorem soundness, semantic novelty scoring, host search completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-frontier exposure, classifier-shift nonescape, score-budget preservation, queue-order preservation, ledger replay, transport, provenance, local naming, Nontrivial, and FieldFaithful rows for BEDC.Derived.CandidateFrontierQueueUp.

Chapter 7618

A Concrete Naming Certificate for $\mathsf{BishopRealSequence}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopRealSequence}^{\uparrow}$ packet records the finite surface of a sequence of real names whose inspected entries are given by stream windows, regular rational readback rows, dyadic tolerances, and terminal real seals. The packet sits after the finite-window discipline of chapter 126, the regular rational schedule of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name surface of chapter 38. It names a sequence object only through displayed finite BEDC rows; it does not choose a limit, quotient sequence representative, countable-choice schedule, host real equality, or ambient completeness theorem.

Definition 7618.1 (Bishop real sequence carrier)**.** *A* $\mathsf{BishopRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family indexed by the visible* $\mathsf{UnaryHistory}$ *entries requested by the consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those windows, recording the regular rational approximants exposed at each inspected index.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite window request.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, R, D have been displayed.*
5. *H records componentwise* hsame *transport for the window family, readback row, tolerance ledger, terminal seal, provenance, and local name.*
6. *C records displayed* Cont *replay from a sequence-entry request through W, R, D, E and then to the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealSequence}^{\uparrow}}$ *row.*

No carrier coordinate stores a selected limit, quotient sequence equality, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 7618.2 (Bishop real sequence pointwise stability)**.** *Let $S = (W, R, D, E, H, C, P, N)$ and $S' = (W', R', D', E', H', C', P', N')$ be accepted* $\mathsf{BishopRealSequence}^{\uparrow}$ *carriers. If the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows agree componentwise under the visible* hsame *transports, then every inspected* $\mathsf{UnaryHistory}$ *entry has the same exported real-name row in S and S'.*

Proof. Project both carriers through Definition 7618.1. The only entry-facing coordinates are W, R, D, E, and each is transported by the displayed rows H, H'.

For a fixed inspected $\mathsf{UnaryHistory}$ entry, read the stream window first. The readback row R is attached to that same finite window, and the tolerance row D records the dyadic budget used by the readback.

The terminal seal E is available only after those rows have been displayed. Since the corresponding rows agree componentwise under hsame, the replay routes C, C' export the same real-name row for the inspected entry. □

Theorem 7618.3 (Bishop real sequence tail window)**.** *Every accepted* $\mathsf{BishopRealSequence}^{\uparrow}$ *consumer that asks for a finite tail window must first display the* $\mathsf{StreamName}^{\uparrow}$ *windows indexed by the requested* $\mathsf{UnaryHistory}$ *tail, then the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, before any terminal* $\mathsf{Real}^{\uparrow}$ *seal can be read.*

Proof. Begin with the consumer request and project the carrier inventory. The request can reach only the displayed rows W, R, D, E, H, C, P, N.

The window row W fixes the finite tail segment being inspected. The readback row R is tied to this window, and the tolerance row D supplies the dyadic budget for the same indices.

Only after this finite surface is visible may the route read E. The structural rows H, C, P, N preserve and name the route without adding a selected tail limit, quotient sequence representative, or choice schedule. □

Theorem 7618.4 (Bishop real sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealSequence}^{\uparrow}$ *carrier $S = (W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealSequence}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{StreamName}^{\uparrow}$ *finite-window display,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7618.1. The tuple displays the finite windows, regular rational readback, dyadic tolerances, terminal real seal, transport data, continuation data, provenance, and local naming row.

The rows W, R, D, E are the complete pointwise read surface. A consumer may ask for more indices only by presenting another finite $\mathsf{UnaryHistory}$ window, not by extracting an infinite sequence object or a selected limit.

The remaining rows H, C, P, N transport, replay, source, and name the same finite surface. Since the carrier has no coordinate for quotient sequence equality, countable choice, host real equality, or completeness, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealSequence}^{\uparrow}$ read displays StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerances, terminal Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a sequence-entry consumer obtains a real-name row.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopRealSequence}^{\uparrow}$ is not bijective to any listed sibling because it packages a finite family of pointwise real-name reads indexed by visible unary history windows, while each listed sibling supplies only one row type or terminal seal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealSequence}^{\uparrow}$ packet is generated from finite StreamName windows indexed by visible unary-history entries, RegSeqRat readback rows, DyadicRatCore tolerance ledgers, terminal Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7618.1, Definition 7618.2, Definition 7618.3, and Definition 7618.4 seed the finite pointwise sequence surface between certified stream windows and terminal Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient sequence equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, pointwise stability, finite tail-window routing, and local NameCert surface proofs for BEDC.Derived.BishopRealSequenceUp.

Chapter 7619

A Concrete Naming Certificate for $\mathsf{JamesCompactnessCriterion}^{\uparrow}$

Following James's weak compactness criterion for bounded closed convex sets in Banach spaces, the $\mathsf{JamesCompactnessCriterion}^{\uparrow}$ packet records the finite BEDC surface by which a displayed weak compactness claim is tested through norm-attaining dual rows. It sits over the Banach carrier of chapter 90, the weak topology surface of chapter 3597, the weak compactness packet of chapter 6851, and the scalar comparison rows of chapter 38. The packet names only the finite criterion ledger and its local NameCert obligations; it does not assert a host dual ball, quotient weak topology, selected maximizing functional, or an ambient compactness theorem outside the displayed rows.

Definition 7619.1 (JamesCompactnessCriterion carrier)**.** *A* $\mathsf{JamesCompactnessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, W, K, D, A, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the ambient* $\mathsf{Banach}^{\uparrow}$ *row, including the finite normed carrier data inspected by the criterion.*
2. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *row on the displayed Banach data.*
3. *K is the candidate* $\mathsf{WeakCompactness}^{\uparrow}$ *row whose acceptance is being routed through the James criterion ledger.*
4. *D is the finite dual-test ledger, listing only the displayed continuous linear functional names and their inspected bounded-set windows.*
5. *A is the norm-attainment row, recording the finite witnesses that each displayed dual test attains its supremum on the candidate set.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-comparison row for the displayed supremum and norm-attainment values.*
7. *H records componentwise* hsame *transport for $B, W, K, D, A, R, C, P, N$.*
8. *C records displayed* Cont *replay from Banach data through weak-topology exposure, finite dual tests, norm-attainment rows, scalar comparisons, and the weak-compactness handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{JamesCompactnessCriterion}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{JamesCompactnessCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected dual-ball point, quotient weak closure, countable choice, a host weak compactness theorem, or bridge checking data.

Theorem 7619.2 (JamesCompactnessCriterion NameCert obligations)**.** *For every accepted* JamesCompactnessCrit *carrier*

$$J = (B, W, K, D, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{JamesCompactnessCriterion}^{\uparrow}}$ *surface consists exactly of Banach carrier exposure through* B*; weak-topology exposure through* W*; candidate weak-compactness exposure through* K*; finite dual-test coverage through* D*; norm-attainment witnesses through* A*; scalar comparison through* R*; componentwise transport and displayed replay through* H, C*; and provenance with local naming through* P, N*.*

Proof. Project the carrier of Definition 7619.1. The analytic rows are precisely B, W, K, D, A, R, so a consumer first sees the Banach carrier, the weak topology, and the candidate weak compactness row.

The criterion-specific read is forced through D, A, R. The dual-test ledger D records the displayed functional names, the attainment row A records the finite witnesses for those tests, and the Real row R supplies the scalar comparison boundary.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no further field, no consumer can extract a selected dual-ball element, quotient weak closure, countable-choice schedule, host compactness theorem, or bridge data from this packet. □

Theorem 7619.3 (JamesCompactnessCriterion weak-compactness route)**.** *Every accepted* JamesCompactnessCrite *packet exports a weak-compactness-facing conclusion only through the finite spine*

$$\mathsf{Banach}^{\uparrow} \longmapsto \mathsf{WeakTopology}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{WeakCompactness}^{\uparrow}.$$

For a carrier $J = (B, W, K, D, A, R, H, C, P, N)$*, any downstream weak-compactness consumer must inspect the Banach row, weak-topology row, finite dual-test ledger, norm-attainment row, scalar comparison row, and componentwise* hsame *transport before it may read* K *as weak-compactness evidence. The route is a BEDC NameCert handoff, not an ambient theorem about all bounded closed convex subsets.*

Proof. Start with the sibling rows named in the chapter introduction. The $\mathsf{Banach}^{\uparrow}$ row supplies the normed carrier, the $\mathsf{WeakTopology}^{\uparrow}$ row supplies the topology-facing exposure, the $\mathsf{Real}^{\uparrow}$ row supplies scalar comparison data, and the $\mathsf{WeakCompactness}^{\uparrow}$ row is the only output coordinate.

Next read the James ledger rows D, A. They ensure that every displayed dual test and every displayed norm-attainment witness is consumed before the weak-compactness row K is exported.

Finally H, C, P, N preserve the same route under componentwise transport, displayed replay, provenance, and local naming. Thus the route reaches weak compactness only through the displayed finite certificate rows and not through a host duality or compactness oracle. □

Falsifiable prediction: An accepted $\mathsf{JamesCompactnessCriterion}^{\uparrow}$ consumer is refuted if it claims weak compactness without displaying the $\mathsf{Banach}^{\uparrow}$ carrier row, $\mathsf{WeakTopology}^{\uparrow}$ exposure row, finite dual-test ledger, norm-attainment row, $\mathsf{Real}^{\uparrow}$ scalar comparison, $\mathsf{WeakCompactness}^{\uparrow}$ handoff, transport, replay, provenance, and local NameCert row.

Independence witness: banach, weak-topology, weak-compactness, real: $\mathsf{Banach}^{\uparrow}$ supplies normed carrier data without the James dual-test ledger; $\mathsf{WeakTopology}^{\uparrow}$ supplies weak-open exposure without norm-attainment witnesses; $\mathsf{WeakCompactness}^{\uparrow}$ supplies a compactness-facing row without the

criterion route; and $\mathsf{Real}^{\uparrow}$ supplies scalar comparisons without Banach or weak-topology provenance. None is bijective to $\mathsf{JamesCompactnessCriterion}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{JamesCompactnessCriterion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{JamesCompactnessCriterion}^{\uparrow}$ packet is generated from Banach carrier rows, weak-topology rows, weak-compactness candidate rows, finite dual-test ledgers, norm-attainment witnesses, Real scalar comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JamesCompactnessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7619.1, Definition 7619.2, Definition 7619.3, the falsifiable prediction, and the independence witness seed the finite James compactness criterion carrier and weak-compactness route over Banach, WeakTopology, WeakCompactness, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JamesCompactnessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host dual ball, quotient weak closure, selected norm-attaining functionals beyond displayed rows, the full ambient James theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite dual-test coverage, norm-attainment routing, scalar comparison, weak-compactness handoff, transport, replay, provenance, and local naming for BEDC.Derived.JamesCompactnessCrite

Chapter 7620

A Concrete Naming Certificate for $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ packet records a located closed real interval by endpoint order, finite window reads, and dyadic net data rather than by an ambient set of completed real points. It consumes the real-name surface of chapter 38, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the located endpoint discipline of chapter 394, and the finite-net interval route of chapter 6500. The packet is a closed-interval carrier for L10 Real and Completion consumers; it does not assert open-cover compactness, choose endpoints as host reals, quotient interval representatives, or import ambient completeness.

Definition 7620.1 (Constructive real closed interval carrier)**.** *A* $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, O, S, R, D, F, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint rows, each read through* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *name windows.*
2. *O is the endpoint-order row, recording that the lower endpoint is admitted before the upper endpoint as a located interval boundary.*
3. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger used by endpoint, membership, and net-cell reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row attached to those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for mesh budgets and endpoint comparisons.*
6. *F is the finite interval-net row, read through the closed-interval finite-net sibling before any completion-facing consumer inspects the interval.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint and cell readbacks.*
8. *H records componentwise* hsame *transport for endpoint, order, window, readback, dyadic, finite-net, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a precision request through the same endpoint and finite-net route.*

10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealClosedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, U, O, S, R, D, F, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected endpoint representatives, quotient interval equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7620.2 (ConstructiveRealClosedInterval NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ *carrier $I = (L, U, O, S, R, D, F, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealClosed}}$ *surface consists exactly of endpoint admission through L, U; endpoint-order admission through O; finite window and rational readback admission through S, R; dyadic tolerance admission through D; closed-interval finite-net admission through F; terminal real-name sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7620.1. The only endpoint rows are L, U, and the only order row is O.

The numerical reads are then forced through S, R, D. A precision request opens a finite stream window, reads regular rational endpoint data in that window, and consumes the dyadic tolerance ledger before reaching the finite-net row.

The finite-net and seal rows are downstream. The row F is the displayed closed-interval net surface, while E is the terminal Real seal reached only after the endpoint, order, window, readback, and dyadic rows have been displayed.

Finally H, C, P, N preserve, replay, source, and locally name that same route. Since the carrier has no coordinate for open covers, selected endpoint representatives, quotient interval equality, countable choice, trichotomy, or ambient completeness, no local NameCert obligation can be stored outside the displayed rows. □

Theorem 7620.3 (ConstructiveRealClosedInterval dyadic net route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ *carrier must read the finite dyadic net route*

$$L, U, O \longmapsto S, R, D \longmapsto F, E \longmapsto H, C, P, N$$

before it can inspect the interval as a closed real interval. The route consumes $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as displayed packet rows, not as an open-cover compactness theorem, a selected subcover, quotient real equality, or a completed host interval.*

Proof. Begin with endpoint admission. The lower and upper endpoint rows, together with the endpoint-order row, are the only source rows for interval membership.

Next read the finite numerical route. The shared $\mathsf{StreamName}^{\uparrow}$ window row S and $\mathsf{RegSeqRat}^{\uparrow}$ readback row R supply endpoint approximations, while the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the mesh tolerance used by the finite net.

The row F is then the only closed-interval finite-net surface visible to the consumer, and the row E is the only Real-facing terminal seal. The structural rows H, C, P, N keep this route invariant under transport, replay, provenance, and local naming.

Therefore a consumer that reaches a closed real interval through the packet receives exactly the displayed dyadic finite-net data and no open-cover compactness proof, selected endpoint representative, selected subcover, quotient interval equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Falsifiable prediction: An accepted $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ consumer is refuted if it reads a closed real interval without first displaying endpoint rows, the endpoint-order row, StreamName and RegSeqRat windows, a DyadicRatCore tolerance row, and the finite closed-interval net row.

Independence witness: real, regseqrat, streamname, dyadicratcore, locatedinterval, and closed-interval-finite-net: none is bijective to the constructive real closed interval packet because only $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ records endpoint order and dyadic finite-net evidence as one local NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealClosedInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ packet is generated from lower and upper Real endpoint rows, endpoint order, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a closed-interval finite-net row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRealClosedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7620.1, Definition 7620.2, and Definition 7620.3 seed the endpoint-order and dyadic finite-net carrier surface over $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRealClosedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected endpoint representatives, quotient interval equality, selected subcovers, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order admission, finite-window readback, dyadic mesh admission, finite-net exactness, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveRealClosedIn

Chapter 7621

A Concrete Naming Certificate for $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$

Following Hadamard and later constructive analysis, the $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ packet records the directional derivative surface in which the direction itself may be approached through a finite perturbation window. It sits between the point-direction interface of chapter 6344 and the full linear approximation interface of chapter 3634. Its displayed route consumes the map surface of chapter 130, the linear direction and response rows of chapter 48, and the terminal real-name seal of chapter 38. The packet names a stable directional read along finite approach windows; it is not a host function-space derivative, a quotient germ equality, a selected limit, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7621.1 (Hadamard directional derivative carrier)**.** *A* $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, V, Q, L, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted source map row read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and the surrounding local domain data;*
2. *A is the base-point and neighbourhood window over which the directional approach is inspected;*
3. *V is the displayed direction row, including the finite perturbation window through which nearby directions are compared;*
4. *Q is the finite difference-quotient schedule, listing the inspected step sizes and perturbed directions;*
5. *L is the* $\mathsf{LinMap}^{\uparrow}$ *response row that records the candidate directional linear value on the displayed direction surface;*
6. *S is the stability row requiring the quotient reads along perturbed directions to agree with the same response row over the finite schedule;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance seal through which the quotient comparison is read;*
8. *H records componentwise* hsame *transport for the source, base window, direction window, quotient schedule, linear response, stability row, Real seal, provenance, and local name;*

9. *C records the displayed* Cont *replay routes for source inspection, direction exposure, quotient readback, stability comparison, linear-response handoff, and Real-seal consumption;*
10. *P is the* Pkg *provenance over* $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$, $\mathsf{GateauxDerivative}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HadamardDirectionalDerivative}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $X, A, V, Q, L, S, R, H, C, P, N$.

Theorem 7621.2 (Hadamard directional derivative NameCert obligations)**.** *For the carrier of Definition 7621.1, the local* $\mathsf{NameCert}_{\mathsf{HadamardDirectionalDerivative}^{\uparrow}}$ *surface consists of carrier admission, direction-window exposure, finite quotient scheduling, perturbation stability, linear-response handoff, classifier stability, and non-escape for the same finite packet $D = (X, A, V, Q, L, S, R, H, C, P, N)$. Direction-window exposure requires V before any quotient read. Finite quotient scheduling requires every inspected step and perturbed direction to be listed in Q. Perturbation stability requires the comparison row S to keep the quotient reads attached to L and R. The non-escape clause excludes host function equality, quotient germs, selected limits, unrestricted Frechet differentiability, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the displayed packet. Carrier admission exposes exactly the rows $X, A, V, Q, L, S, R, H, C, P, N$, so there is no hidden coordinate for a host derivative or a selected limiting value.

The direction-window and quotient clauses follow from the order of the carrier. The direction row V is displayed before Q, and Q is the only finite schedule from which quotient reads may be taken.

The stability and handoff clauses use S, L, R. The row S compares all listed perturbation reads against the same $\mathsf{LinMap}^{\uparrow}$ response L through the $\mathsf{Real}^{\uparrow}$ tolerance seal R.

Finally, H and C transport and replay only those rows. Thus a consumer asking for host function equality, quotient-germ equality, an unrestricted Frechet derivative, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness leaves the accepted packet. □

Theorem 7621.3 (Hadamard directional derivative finite-step stability)**.** *For every accepted* HadamardDirectionalD *carrier*

$$D = (X, A, V, Q, L, S, R, H, C, P, N),$$

each finite quotient read selected by Q is stable only along the displayed direction-window route

$$A, V, Q, L, S, R, H, C, P, N.$$

A $\mathsf{GateauxDerivative}^{\uparrow}$*-facing consumer may read the displayed directional value through V, Q, L, S, R, while a* $\mathsf{FrechetDerivative}^{\uparrow}$*-facing consumer must still supply a separate full linear-approximation packet. The Hadamard carrier itself exports no host derivative operator, no quotient germ, no selected limit, and no ambient completeness theorem.*

Proof. Begin with the finite quotient schedule Q. Its entries are admitted only after the base window A and the direction window V have been displayed.

Apply the NameCert obligations. They attach every quotient read to the stability row S, the linear response L, and the Real tolerance seal R.

The transport and replay rows H and C preserve that same finite route under accepted consumer motion. Provenance and local naming, P and N, record the route without adding derivative data.

Therefore the packet supplies a stable Hadamard directional read for $\mathsf{GateauxDerivative}^{\uparrow}$-style consumers, but it does not by itself provide the global linear approximation required by $\mathsf{FrechetDerivative}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ packet must expose a direction window, a finite quotient schedule, a perturbation-stability row, and a $\mathsf{Real}^{\uparrow}$ tolerance seal before any directional derivative read is public.

Independence witness: Nearest siblings: gateaux-derivative, frechet-derivative, continuousmap, linearmap, real. The carrier is not bijective to GateauxDerivativeUp because it carries a perturbation-stability window; not to FrechetDerivativeUp because it does not require a full norm-linear approximation over all directions; not to ContinuousMapUp because it adds quotient and derivative rows; not to LinearMapUp because it is a directional derivative certificate; not to RealUp because it keeps the source, direction, quotient, transport, replay, provenance, and naming rows visible.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardDirectionalDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ packet is generated from a continuous-map source row, a base-point window, a displayed direction window, finite quotient schedules, a LinearMap response row, a perturbation-stability row, a RealUp tolerance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardDirectionalDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7621.1, Definition 7621.2, Definition 7621.3, the falsifiable prediction, and the independence witness bind the seed surface for Hadamard directional derivative reads between Gateaux and Frechet derivative packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardDirectionalDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, host function equality, quotient germs, selected limits, unrestricted Frechet differentiability, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, direction-window exposure, finite quotient scheduling, perturbation stability, linear-response handoff, classifier stability, non-escape, provenance, and TasteGate alignment for the same finite packet.

Chapter 7622

A Concrete Naming Certificate for $\mathsf{MoreauEnvelope}^{\uparrow}$

Following Moreau's envelope construction, Rockafellar's convex-analysis account of infimal convolution, and Boyd–Vandenberghe's proximal-operator presentation, the $\mathsf{MoreauEnvelope}^{\uparrow}$ packet records the finite lower-envelope surface for a convex functional after a quadratic penalty has been displayed. It sits beside the conjugacy route of chapter 3563, the subgradient support ledger of chapter 3712, the proximal candidate route of chapter 3713, the real seal of chapter 38, and the norm source of chapter 89. The packet lets proximal and Fenchel consumers cite one finite envelope certificate without asserting selected minimizers, host suprema, quotient function equality, Hilbert projection theorems, or ambient completeness.

7622.1 Moreau Envelope Carrier

Definition 7622.1 (Moreau envelope carrier)**.** *A* $\mathsf{MoreauEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, \lambda, x, Y, Q, V, L, A, G, H, C, P, N)$$

with the following visible rows:

1. *X is the normed source row, read through* $\mathsf{Norm}^{\uparrow}$ *and the scalar seal of* $\mathsf{Real}^{\uparrow}$*.*
2. *F is the convex functional row, including the displayed convex-domain, epigraph, and conjugacy surfaces used by the request.*
3. *λ is the positive regularization-scale row carried as displayed real data.*
4. *x is the input point row in the source carrier.*
5. *Y is the finite comparison-window row listing the displayed candidate points inspected by the envelope request.*
6. *Q is the quadratic-penalty ledger recording the visible values $\frac{1}{2\lambda}\|y - x\|^2$ for y in Y.*
7. *V is the value ledger recording the displayed sums $F(y) + \frac{1}{2\lambda}\|y - x\|^2$ after both the functional and penalty rows have been read.*
8. *L is the lower-envelope budget row recording the finite lower comparison available to the packet; it is a BEDC ledger row, not a host infimum operator.*

9. *A is the proximal-candidate handoff row, read only when a displayed* $\mathsf{ProximalMap}^{\uparrow}$ *packet supplies a candidate compatible with the same $X, F, \lambda, x, Q, V, L$ rows.*

10. *G is the convex-conjugate and subdifferential handoff row, read through* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{Subdifferential}^{\uparrow}$ *after the envelope comparison rows have been displayed.*

11. *H records componentwise* hsame *transport for the source, functional, scale, input, comparison window, penalty, value, lower-envelope budget, proximal handoff, conjugacy handoff, replay, provenance, and local naming rows.*

12. *C records displayed* Cont *replay from an envelope request through norm evaluation, real penalty readback, convex-functional comparison, proximal handoff, and conjugacy or subgradient consumers.*

13. *P is the* Pkg *provenance over* $\mathsf{MoreauEnvelope}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{Subdifferential}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{MoreauEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, \lambda, x, Y, Q, V, L, A, G, H, C, P, N$. It has no coordinate for selected minimizers, host suprema, quotient function equality, Hilbert projection theorems, lower-semicontinuity existence theorems, or ambient completeness.

7622.2 Moreau Envelope Certificate

Theorem 7622.2 (Moreau envelope NameCert obligations)**.** *For every accepted* $\mathsf{MoreauEnvelope}^{\uparrow}$ *carrier $M = (X, F, \lambda, x, Y, Q, V, L, A, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MoreauEnvelope}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the normed source, convex functional, positive scale, input point, comparison window, quadratic penalty, value ledger, lower-envelope budget, proximal handoff, conjugacy handoff, transport, replay, provenance, and local naming rows;*

2. *penalty exactness, requiring every envelope comparison to read the same* $\mathsf{Norm}^{\uparrow}$ *source row,* $\mathsf{Real}^{\uparrow}$ *scalar row, scale row λ, and input row x before the penalty row Q is public;*

3. *value-ledger exactness, requiring each displayed value in V to be formed from the same functional row F and penalty row Q for a comparison point listed in Y;*

4. *lower-envelope nonescape, requiring the budget row L to remain a displayed finite ledger rather than a host infimum or host supremum;*

5. *proximal handoff compatibility, requiring any* $\mathsf{ProximalMap}^{\uparrow}$ *consumer to read the same $X, F, \lambda, x, Q, V, L$ rows before it treats a displayed candidate as an envelope witness;*

6. *conjugacy and subgradient compatibility, requiring Fenchel or subdifferential consumers to pass through* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{Subdifferential}^{\uparrow}$ *only after the envelope comparison rows have been displayed.*

Proof. Project the accepted carrier. The rows $X, F, \lambda, x, Y, Q, V, L, A, G$ are the entire envelope surface, and the rows H, C, P, N are the entire structural route that transports, replays, sources, and names that same packet.

Penalty exactness is read first. A comparison point in Y contributes to the envelope request only after the normed source, scalar seal, scale, input, and quadratic penalty value have all been displayed in X, λ, x, Q.

The value ledger is then forced by F, Q, V. Because V records only the displayed sums for listed comparison points, a consumer cannot replace the finite comparison window by a host function space or a quotient-function equality claim.

Finally read L, A, G, H, C, P, N. The row L is a finite lower-envelope budget, A is downstream of the proximal packet, and G is downstream of conjugacy and subgradient packets. Since no carrier coordinate stores a selected minimizer, host supremum, Hilbert projection theorem, or ambient completeness principle, none of those objects can be exported by the local certificate. □

Theorem 7622.3 (Moreau envelope proximal-real route)**.** *Every proximal, Fenchel, or real-valued optimization consumer of an accepted* $\mathsf{MoreauEnvelope}^{\uparrow}$ *packet must factor through the finite route*

$$X, F, \lambda, x, Y, Q, V, L, A, G \longmapsto H, C, P, N.$$

The consumer may read an envelope value only after the norm row, real scalar seal, convex-functional row, finite comparison window, quadratic-penalty ledger, value ledger, lower-envelope budget, and proximal or conjugacy handoff have all been displayed. It receives no selected minimizer, host supremum, quotient function equality, Hilbert projection theorem, or ambient completeness principle.

Proof. Begin with the real-valued read. The row Q supplies the only quadratic penalty values, and those values are available only through the $\mathsf{Norm}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows named by X and λ.

Next inspect the envelope comparison. The row V combines the displayed convex-functional values with the displayed penalty values for the finite window Y, while L records the lower-envelope budget available to the packet.

Now route downstream consumers. A proximal read must pass through A and the $\mathsf{ProximalMap}^{\uparrow}$ packet before a candidate can be treated as compatible with the envelope rows. A Fenchel or subgradient read must pass through G and the $\mathsf{ConvexConjugate}^{\uparrow}$ or $\mathsf{Subdifferential}^{\uparrow}$ rows after the same comparison data have been exposed.

The structural rows H, C, P, N preserve exactly this route. They add no selected minimizer, host supremum, quotient equality of functions, Hilbert projection semantics, or ambient completeness, so the handoff stops at the displayed proximal-real envelope boundary. □

Falsifiable prediction: Every accepted $\mathsf{MoreauEnvelope}^{\uparrow}$ consumer read factors through a NormUp source, RealUp scale and value rows, ConvexConjugateUp and SubdifferentialUp handoff rows when dual data are requested, a ProximalMapUp candidate route when a proximal witness is requested, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convex-conjugate, subdifferential, proximal-map, real, norm: $\mathsf{MoreauEnvelope}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source row, functional row, positive scale, input, comparison window, quadratic penalty ledger, value ledger, lower-envelope budget, proximal handoff, conjugacy handoff, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MoreauEnvelope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MoreauEnvelope}^{\uparrow}$ packet is generated from a normed source row, convex functional row, positive real scale, input point, finite comparison window, quadratic penalty values, displayed value ledger, lower-envelope budget, proximal handoff, convex-conjugate and subdifferential handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MoreauEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7622.1, Definition 7622.2, and Definition 7622.3 seed the finite Moreau-envelope certificate surface over proximal, convex-conjugate, subdifferential, real, and norm rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MoreauEnvelope}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected minimizers, host suprema, quotient function equality, Hilbert projection theorems, lower-semicontinuity existence theorems, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, penalty exactness, value-ledger exactness, lower-envelope nonescape, proximal handoff compatibility, conjugacy and subgradient compatibility, transport, replay, provenance, and local naming for BEDC.Derived.MoreauEnvelopeUp.

Chapter 7623

A Concrete Naming Certificate for $\mathsf{MirrorDescent}^{\uparrow}$

Following Nemirovski and Yudin, Beck and Teboulle, and Bubeck, the $\mathsf{MirrorDescent}^{\uparrow}$ packet records mirror descent as finite convex-dual NameCert data. It sits after the feasible convex source of chapter 208, the convex-conjugate ledger of chapter 3563, the Fenchel-duality route of chapter 321, the proximal-map handoff of chapter 3713, and the linear-programming duality certificate of chapter 235. The chapter names the mirror update as displayed convex optimization data; it does not import Hilbert projection, quotient equality of iterates, a host minimizer choice, or a selected limiting point.

Definition 7623.1 (MirrorDescent carrier)**.** *A* $\mathsf{MirrorDescent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, D, B, G, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the feasible convex row read through the* $\mathsf{ConvexSet}^{\uparrow}$ *surface.*
2. *F is the mirror potential row whose conjugate and subgradient-facing data are read through* $\mathsf{ConvexConjugate}^{\uparrow}$*.*
3. *D is the dual-gradient row carrying the finite dual read used by the update.*
4. *B is the Bregman transport row comparing the old primal row, the dual-gradient row, and the transported next-row candidate.*
5. *G is the step ledger, listing the displayed step size, tolerance, and update cell consumed by the route.*
6. *S is the descent certificate row produced only after X, F, D, B, G have been displayed.*
7. *H records componentwise* hsame *transport for the feasible row, mirror potential, dual-gradient data, Bregman transport, step ledger, descent certificate, provenance, and local naming row.*
8. *C records the displayed* Cont *replay from the feasible row to the mirror potential, from the mirror potential to the dual-gradient row, from the Bregman row to the step ledger, and from the step ledger to the descent certificate.*
9. *P is the* Pkg *provenance over* $\mathsf{MirrorDescent}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{ConvexConjugate}^{\uparrow}$*,* $\mathsf{FenchelDuality}^{\uparrow}$*,* $\mathsf{ProximalMap}^{\uparrow}$*,* $\mathsf{LPDuality}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MirrorDescent}^\uparrow}$ *row.*
The carrier has no coordinate for Hilbert projection, a host-selected minimizer, quotient equality of iterates, countable choice, selected limits, global optimizer existence, or an ambient convergence theorem.

Theorem 7623.2 (MirrorDescent NameCert obligations)**.** *For every accepted* $\mathsf{MirrorDescent}^\uparrow$ *carrier* $M = (X, F, D, B, G, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MirrorDescent}^\uparrow}$ *surface consists exactly of carrier admission for the displayed tuple, classifier transport over* $X, F, D, B, G, S, H, C, P, N$*, Bregman transport stability through* B *and* H*, Fenchel-dual compatibility through* F, D *and the* $\mathsf{FenchelDuality}^\uparrow$ *provenance row, step-ledger nonescape through* G*, descent-certificate handoff through* S*, displayed replay through* C*, dependency provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The feasible convex row, mirror potential, dual-gradient data, Bregman transport row, step ledger, and descent certificate are the only analytic coordinates.

Classifier transport is componentwise through H. Bregman stability can compare only the displayed rows X, F, D, B, G, S, and the replay row C fixes the order in which these rows are consumed.

Fenchel-dual compatibility is read from the same provenance route. The $\mathsf{ConvexConjugate}^\uparrow$, $\mathsf{FenchelDuality}^\uparrow$, $\mathsf{ProximalMap}^\uparrow$, and $\mathsf{LPDuality}^\uparrow$ rows named by P supply the finite convex-dual ledgers, while G and S keep the step and descent handoff inside the packet.

Since no carrier coordinate stores Hilbert projection, a host minimizer, quotient equality, selected limits, global optimizer existence, or an ambient convergence theorem, none of those objects can be exported by the local NameCert surface. □

Theorem 7623.3 (MirrorDescent Bregman transport certificate)**.** *Every mirror update read from an accepted* $\mathsf{MirrorDescent}^\uparrow$ *packet consumes the* $\mathsf{ConvexSet}^\uparrow$ *feasible row,* $\mathsf{ConvexConjugate}^\uparrow$ *mirror-potential row,* $\mathsf{FenchelDuality}^\uparrow$ *dual-compatibility row,* $\mathsf{ProximalMap}^\uparrow$ *update handoff, and* $\mathsf{LPDuality}^\uparrow$ *dual ledger through the displayed packet*

$$X \longrightarrow F \longrightarrow D \longrightarrow B \longrightarrow G \longrightarrow S.$$

The route is a Bregman transport certificate over finite BEDC rows, not a Hilbert projection, host minimizer choice, quotient equality of iterates, or selected limiting argument.

Proof. Start from Definition 7623.1. The feasible row X is the only source row, and the mirror potential and dual-gradient rows F, D are displayed before the Bregman row B can be read.

Apply Definition 7623.2. Bregman transport stability and Fenchel-dual compatibility are obligations of the same carrier, with the replay row C ordering the read through X, F, D, B, G, S.

The proximal and linear-dual surfaces enter only as provenance rows. They certify that the update handoff and dual ledger have been read as finite dependency data, while the step ledger G and descent row S remain inside the displayed packet.

Thus the mirror update route consumes the convex, conjugate, Fenchel-dual, proximal, and linear-dual rows without adding Hilbert projection, host minimizer choice, quotient equality, or a selected limit. □

Closure status for $\mathsf{NameCert}_{\mathsf{MirrorDescent}^\uparrow}$**.**

Constructive story: A $\mathsf{MirrorDescent}^\uparrow$ packet is generated from feasible convex data, a mirror-potential row, dual-gradient data, Bregman transport, a step ledger, a descent certificate, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MirrorDescent}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 7623.1, Definition 7623.2, and Definition 7623.3 seed the finite convex-dual mirror update route.

Formal verification: FormalStatus(NameCert$_{\mathsf{MirrorDescent}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim convergence rates, global optimizer existence, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, Bregman transport stability, Fenchel-dual compatibility, step-ledger nonescape, descent-certificate handoff, replay, provenance, and local naming for BEDC.Derived.MirrorDescentUp.

Chapter 7624

A Concrete Naming Certificate for $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$

The $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ packet realizes the observer-expansion branch of chapter 9463 as a finite ledger that places Busemann, visual-shadow, boundary, metric, observer-window, transport, provenance, and local naming rows on one route. It sits after the cross-ratio boundary comparison of chapter 3948, the geodesic-lamination surface of chapter 4203, the Patterson–Sullivan shadow packet of chapter 6070, the visual-metric certificate of chapter 3866, and the Busemann-boundary certificate of chapter 3732. The packet names a finite horoball-shadow ledger; it is not a quotient ideal boundary, a selected geodesic ray, a completed conformal density, an ambient hyperbolic group action, a physical observer trajectory, or a global measure construction.

Definition 7624.1 (Hyperbolic horoball-shadow ledger carrier)**.** *A* $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (B, V, W, M, O, X, L, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row recording the horoball threshold and boundary-level Busemann read.*
2. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row recording the visible shadow metric used by the boundary read.*
3. *W is the visual-shadow window row, tying the Busemann threshold and visual metric to a finite aperture.*
4. *M is the metric comparison row that keeps the horoball threshold attached to displayed hyperbolic distance data.*
5. *O is the observer-window row recording the finite observer-expansion read that consumes the shadow.*
6. *X is the cross-ratio and lamination sibling row, recording the boundary comparison route through* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *and* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *when those rows are present.*
7. *L is the Patterson–Sullivan shadow sibling row, recording the counting and boundary-weight ledger consumed by the same shadow aperture when present.*

8. *T is the refusal row excluding quotient ideal-boundary equality, selected rays, completed conformal densities, host group actions, physical observer trajectories, and unlogged global measures.*
9. *H records componentwise* hsame *transport for $B, V, W, M, O, X, L, T, C, P, N$.*
10. *C records displayed* Cont *replay through Busemann readback, visual-shadow exposure, metric comparison, observer-window routing, sibling ledger rows, refusal, transport, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$, HyperbolicPattersonSullivanShad BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, V, W, M, O, X, L, T, H, C, P, N$.

Theorem 7624.2 (Hyperbolic horoball-shadow ledger NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ *carrier $Z = (B, V, W, M, O, X, L, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}}$ *surface consists exactly of Busemann-boundary admission through B; visual-shadow metric exposure through V, W; metric comparison through M; observer-window exposure through O; cross-ratio and geodesic-lamination sibling exposure through X; Patterson–Sullivan sibling exposure through L; refusal exactness through T; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7624.1. The displayed tuple exposes the Busemann row, visual metric row, shadow window, metric comparison, observer window, sibling rows, refusal row, and structural rows.

Read the geometric rows first. Rows B, V, W, M, O bind the horoball threshold, visual-shadow metric, finite aperture, metric comparison, and observer-window route to one finite ledger.

The sibling rows X, L then record the boundary comparison, lamination, and Patterson–Sullivan shadow routes without replacing the local carrier. Finally T, H, C, P, N refuse unlogged host geometry, transport the displayed fields, replay the route, record provenance, and name the packet. No further NameCert obligation is stored outside these rows. □

Theorem 7624.3 (Hyperbolic horoball-shadow ledger sibling independence)**.** *The* HyperbolicHoroballShadowLedge *packet is not bijective to any one of the cross-ratio, geodesic-lamination, Patterson–Sullivan shadow, visual-metric, or Busemann-boundary sibling packets. Each sibling may supply one row of Z, but the accepted ledger requires the joint Busemann threshold, visual-shadow window, metric comparison, observer-window route, sibling ledger rows, refusal row, transport, replay, provenance, and local naming.*

Proof. Compare the carrier fields. A $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ packet supplies boundary comparison data but not the Busemann threshold, observer window, or Patterson–Sullivan ledger row. A $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ packet supplies lamination data but not the horoball threshold or visual-shadow aperture.

The $\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}$ sibling supplies a counting and boundary-weight route, while $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ and $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ supply visual and Busemann rows. None of those packets alone contains all rows $B, V, W, M, O, X, L, T, H, C, P, N$.

Therefore each sibling is a dependency row rather than a replacement carrier, and no listed sibling is bijective to the accepted horoball-shadow ledger. □

Theorem 7624.4 (Hyperbolic horoball-shadow ledger observer-window nonescape)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ *packet must read the Busemann row* B*, visual-shadow rows* V, W*, metric row* M*, observer-window row* O*, and sibling ledger rows* X, L *before it can use the structural rows* H, C, P, N*. Consequently the packet exports no quotient ideal-boundary equality, selected geodesic ray, completed conformal density, ambient hyperbolic group action, physical observer trajectory, or global measure construction.*

Proof. Start with the observer-expansion consumer. Projection of Z gives the finite route through B, V, W, M, O, X, L before the structural rows are available.

The refusal row T is part of the same route. It blocks interpretations that replace the displayed horoball-shadow ledger by quotient boundary equality, selected rays, conformal densities, host group actions, physical trajectories, or global measures.

Rows H, C, P, N then transport, replay, source, and name only the displayed ledger. Since none of those rows adds a hidden geometry coordinate, the consumer cannot escape the finite observer-window surface. □

Falsifiable prediction: Any accepted $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ packet that omits the Busemann threshold row, the visual-shadow window row, or the observer-window row fails to produce the observer-window nonescape route of Definition 7624.4.

Independence witness: hyperbolic-cross-ratio, hyperbolic-geodesic-lamination, hyperbolic-patterson-sullivan-shadow, hyperbolic-visual-metric, hyperbolic-busemann-boundary: no listed sibling is bijective to $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ because the ledger jointly carries Busemann threshold, visual-shadow aperture, metric comparison, observer-window, sibling ledger, refusal, transport, replay, provenance, and local NameCert rows.

The future Lean TasteGate route must provide explicit $\mathsf{Nontrivial}\ \mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ instances in `lean4/BEDC/Derived/HyperbolicHoroballShado`
Those instances must preserve $B, V, W, M, O, X, L, T, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}$ packet is generated from Busemann-boundary rows, visual-shadow metric rows, a finite shadow window, metric comparison rows, an observer-window route, cross-ratio and geodesic-lamination sibling rows, a Patterson–Sullivan shadow sibling row, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7624.1, Definition 7624.2, Definition 7624.3, and Definition 7624.4 seed the finite horoball-shadow ledger through Busemann, visual-shadow, metric, observer-window, sibling, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballShadowLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient ideal-boundary equality, selected geodesic rays, completed conformal densities, ambient hyperbolic group actions, physical observer trajectories, global measure constructions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Busemann exposure, visual-shadow exposure, metric comparison, observer-window nonescape, sibling independence, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicHoroballShadowLedgerUp.

Chapter 7625

A Concrete Naming Certificate for $\mathsf{EilenbergZilber}^{\uparrow}$

Following Eilenberg and Zilber, the $\mathsf{EilenbergZilber}^{\uparrow}$ packet records the finite chain-level comparison between the product of simplicial objects and the tensor product of their chain rows. It sits after the categorical source surface of chapter 61, the functorial transport surface of chapter 62, the face-and-degeneracy ledger of chapter 157, the homology consumer of chapter 98, the bilinear source row of chapter 87, and the finite product carrier of chapter 34. The packet names shuffle, Alexander–Whitney, boundary, chain-homotopy, transport, replay, provenance, and local naming rows; it does not choose simplices, quotient chain representatives, a host homotopy equivalence, unrestricted naturality, or ambient homological algebra.

Definition 7625.1 (Eilenberg-Zilber carrier)**.** *An* $\mathsf{EilenbergZilber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (C, F, S, H, T, P, B, A, W, R, Q, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Cat}^{\uparrow}$ *source row naming the finite chain-category and product-category readings used by the packet.*
2. *F is the* $\mathsf{Functor}^{\uparrow}$ *transport row carrying the chain functor readings from simplicial data to chain data.*
3. *S is the* $\mathsf{SimplicialSet}^{\uparrow}$ *row recording the finite face, degeneracy, and product-simplex windows inspected by the consumer.*
4. *H is the* $\mathsf{Homology}^{\uparrow}$ *boundary-consumer row that may be read only after the chain-level comparison rows have been displayed.*
5. *T is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the tensor product of the two displayed chain rows.*
6. *P is the* $\mathsf{Prod}^{\uparrow}$ *row naming the product simplicial source before any tensor comparison is exported.*
7. *B records the boundary-compatibility ledger for the displayed shuffle and Alexander–Whitney rows.*
8. *A records the Alexander–Whitney projection row from the product-chain window to the tensor-chain window.*

9. *W records the shuffle row from the tensor-chain window back to the product-chain window.*
10. *R records componentwise* hsame *transport for categorical, functorial, simplicial, homology, tensor, product, boundary, comparison, provenance, and local naming rows.*
11. *Q records displayed* Cont *replay from a product-simplex request through the product, simplicial, functorial, tensor, boundary, and homology rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EilenbergZilber}^{\uparrow}}$ *row.*

No coordinate stores a selected simplex representative, quotient chain equality, host chain-homotopy equivalence, unrestricted naturality theorem, or ambient homological-algebra principle.

Lean4 (statement) BEDC.Derived.EilenbergZilberUp

Theorem 7625.2 (Eilenberg-Zilber NameCert obligations)**.** *For every accepted* $\mathsf{EilenbergZilber}^{\uparrow}$ *carrier* $Z = (C, F, S, H, T, P, B, A, W, R, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EilenbergZilber}^{\uparrow}}$ *surface consists exactly of carrier admission, product-source display, simplicial face-and-degeneracy display, functorial chain transport, tensor readback, boundary compatibility, Alexander–Whitney route, shuffle route, chain-homotopy ledger, componentwise transport, displayed replay, and local naming. Every homology-facing consumer reads the* $\mathsf{Homology}^{\uparrow}$ *row only through these finite rows.*

Proof. Project the carrier of Definition 7625.1. The displayed tuple lists the categorical, functorial, simplicial, homology, tensor, product, boundary, comparison, transport, replay, and local naming rows.

The product-source display is forced by P and S. A consumer first reads the finite product and simplicial windows before the functor row F can expose chain data.

The tensor readback then passes through T, while A and W record the two comparison directions. Boundary compatibility is confined to B, and chain-homotopy evidence is read only as a ledger row attached to the displayed comparison data.

Finally R, Q, N transport, replay, and name the same packet. Since the carrier has no coordinate for selected simplices, quotient chain equality, host equivalence, unrestricted naturality, or ambient homological algebra, the listed rows exhaust the local NameCert obligations. □

Theorem 7625.3 (Eilenberg-Zilber chain-homotopy route)**.** *Let* $Z = (C, F, S, H, T, P, B, A, W, R, Q, N)$ *be an accepted* $\mathsf{EilenbergZilber}^{\uparrow}$ *carrier. Every chain-homotopy-facing consumer of* Z *factors through the finite route*

$$P \longmapsto S \longmapsto F \longmapsto T \longmapsto A, W \longmapsto B \longmapsto H \longmapsto R, Q, N.$$

Consequently the consumer receives the product-simplex row, the simplicial face-and-degeneracy row, the functorial chain row, the tensor row, the comparison rows, and the boundary ledger before any homology-facing reading is exported.

Proof. Begin with a chain-homotopy-facing request. Carrier admission exposes H only as a downstream row of the finite packet, not as a detached homology assertion.

Replay Q. The request first reads the product row P and the simplicial row S, then uses the functorial transport row F to reach the chain surface.

The tensor row T is consumed before the comparison rows A, W. Boundary compatibility is then checked by B, so the chain-homotopy ledger is attached to displayed chain data rather than to an ambient quotient object.

Only after these rows have been replayed can the consumer read H. The transport and naming rows R, Q, N preserve the same route, so no host chain-homotopy equivalence or unrestricted naturality theorem is exported. □

Falsifiable prediction: Every accepted $\mathsf{EilenbergZilber}^{\uparrow}$ read displays CategoryUp, FunctorUp, SimplicialSetUp, TensorProductUp, ProdUp, boundary, Alexander–Whitney, shuffle, transport, replay, and local naming rows before a HomologyUp consumer obtains a chain-homotopy-facing row.

Independence witness: category, functor, simplicialset, homology, tensorproduct, prod: $\mathsf{EilenbergZilber}^{\uparrow}$ is not bijective to any listed sibling because it packages the finite comparison route between product simplicial chains and tensor chains, while each sibling supplies only one source, transport, consumer, tensor, or product surface.

Closure status for $\mathsf{NameCert}_{\mathsf{EilenbergZilber}^{\uparrow}}$.

Constructive story: An $\mathsf{EilenbergZilber}^{\uparrow}$ packet is generated from displayed CategoryUp and FunctorUp rows, finite SimplicialSetUp product windows, TensorProductUp and ProdUp source rows, boundary compatibility, Alexander–Whitney and shuffle comparison rows, chain-homotopy ledger data, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EilenbergZilber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7625.1, Definition 7625.2, and Definition 7625.3 seed the finite chain-homotopy comparison surface between product simplicial data and tensor-chain homology consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EilenbergZilber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected simplex representatives, quotient chain equality, host chain-homotopy equivalence, unrestricted naturality, ambient homological algebra, bridge checking, or Lean theorem verification.

Upgrade path: Obligation closure requires checked carrier admission, product-source display, tensor readback, boundary compatibility, Alexander–Whitney and shuffle route exactness, chain-homotopy ledger, and local NameCert proofs for BEDC.Derived.EilenbergZilberUp.

Chapter 7626

A Concrete Naming Certificate for $\mathsf{UniformCompletionUniqueness}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley uniform-space completion, the UniformCompletionUni packet records the finite NameCert surface where two uniform-completion presentations over the same displayed uniform source may be compared. It sits after the uniform source of chapter 339, the uniform completion boundary of chapter 1473, the Cauchy-filter route of chapter 333, the separated comparison ledger of chapter 1463, the metric-completion comparison of chapter 1506, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the terminal Real seal of chapter 38. The packet names uniqueness as finite routed comparison data rather than as quotient-completion equality, selected limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7626.1 (Uniform completion uniqueness carrier)**.** *A* $\mathsf{UniformCompletionUniqueness}^{\uparrow}$ *carrier is a finite packet*

$$K = (U, C, F, S, M, R, E, H, T, P, N)$$

with the following displayed rows.

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, exposing the entourage ledger and separatedness data read by both completion presentations.*
2. *C stores the two* $\mathsf{UniformCompletion}^{\uparrow}$ *presentation rows over the same displayed source.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *route by which each presentation reads uniform Cauchy data.*
4. *S is the separated comparison row, read through* $\mathsf{CompletionUniqueness}^{\uparrow}$ *after the two presentations and the Cauchy-filter route have been displayed.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row used only as a metric handoff for consumers that enter through metric completion.*
6. *R is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row visible before regular readback.*
7. *E combines the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the same finite windows.*
8. *H records componentwise* hsame *transport for U, C, F, S, M, R, E.*
9. *T records displayed* Cont *replay from the uniform source through the two presentations, Cauchy-filter route, separated comparison, metric handoff, finite windows, readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionUniqueness}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, C, F, S, M, R, E, H, T, P, N$. It has no coordinate for quotient-completion equality, selected limits, countable choice, host equality of completed points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformCompletionUniquenessUp.TasteGate.UniformCompletionUniquenessTasteGate`
`single carrier alignment`

Theorem 7626.2 (Uniform completion uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompletionUniqueness}^{\uparrow}$ *carrier $K = (U, C, F, S, M, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionUnique}}$ *surface consists exactly of carrier admission for the uniform source row, two uniform-completion presentation rows, Cauchy-filter route, separated comparison row, metric-completion comparison row, StreamName windows, RegSeqRat readback and Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7626.1. The comparison data are the uniform source U, the two presentation rows in C, the Cauchy-filter route F, and the separated comparison row S.

The metric and Real-facing rows are downstream handoffs. Row M lets metric-completion consumers enter the same comparison surface, while R and E expose finite windows, regular readback, and the terminal Real seal.

Finally H, T, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no quotient-completion equality, selected limit, countable-choice schedule, host equality, or ambient completeness row, none of those objects is a local NameCert obligation. □

Theorem 7626.3 (Uniform completion uniqueness separated comparison)**.** *Every uniqueness-facing comparison exported by an accepted* $\mathsf{UniformCompletionUniqueness}^{\uparrow}$ *packet first reads the common* $\mathsf{UniformSpace}^{\uparrow}$ *source row U, the two* $\mathsf{UniformCompletion}^{\uparrow}$ *presentation rows C, the* $\mathsf{CauchyFilter}^{\uparrow}$ *route F, and then the separated comparison row S through* $\mathsf{CompletionUniqueness}^{\uparrow}$. *The comparison is therefore a finite separated-comparison route, not a quotient-completion equality or host equality of completed points.*

Proof. Start with Definition 7626.2. It orders carrier admission through the source, presentation, Cauchy-filter, and separated comparison rows before any downstream handoff can be read.

Read the separated comparison row S. Its role is to compare the two displayed completion presentations after the shared uniform Cauchy route is visible, so it cannot be replaced by equality of quotient representatives.

The structural rows H, T, P, N preserve and replay that order. Thus every uniqueness-facing consumer receives a displayed separated comparison and no host equality witness. □

Theorem 7626.4 (Uniform completion uniqueness Real metric handoff)**.** *Every metric-facing or Real-facing consumer of an accepted* $\mathsf{UniformCompletionUniqueness}^{\uparrow}$ *packet factors through the same comparison route*

$\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Re

The handoff exposes the metric-completion comparison row, finite StreamName windows, RegSeqRat readback, terminal Real seal, componentwise transport, replay, provenance, and local naming,

but it does not export selected limits, quotient completion equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 7626.3. The metric-facing and Real-facing reads cannot reach M, R, E until the common uniform source, the two completion presentations, the Cauchy-filter route, and the separated comparison row have been displayed.

The row M is only a comparison handoff for metric-completion consumers. Rows R and E then expose finite windows, regular rational readback, and the Real seal attached to the same comparison route.

Use Definition 7626.2 for the structural rows. Since transport, replay, provenance, and local naming preserve the same finite route and the excluded objects are absent from the carrier, the handoff remains inside the displayed NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionUniqueness}^{\uparrow}$ packet is generated from a UniformSpace source row, two UniformCompletion presentation rows, a CauchyFilter route, separated CompletionUniqueness comparison, MetricCompletion comparison, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7626.1, Definition 7626.2, Definition 7626.3, and Definition 7626.4 seed the finite uniform-completion uniqueness route from uniform Cauchy data through separated comparison and metric or Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, two-presentation order, Cauchy-filter route exactness, separated comparison, metric and Real handoff, transport replay, provenance non-escape, and local naming for BEDC.Derived.UniformCompletionUniquenessUp.

Chapter 7627

A Concrete Naming Certificate for $\mathsf{FiniteRealPicardIteration}^{\uparrow}$

Following Picard and Bishop–Bridges constructive metric iteration, the $\mathsf{FiniteRealPicardIteration}^{\uparrow}$ packet records the finite Real-facing Picard window used before a contraction, completion, or fixed-point consumer may read a terminal real-name seal. It sits over the broader Picard iteration surface of chapter 2359, the contraction-real modulus row of chapter 7615, the finite completion handoff of chapter 1559, the Bishop comparison route of chapter 1508, and the real-completion spine of chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a finite iteration certificate; it is not a proof of Banach fixed-point existence, quotient equality of Picard streams, a selected limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7627.1 (FiniteRealPicardIteration carrier)**.** *A* $\mathsf{FiniteRealPicardIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, F, D, W, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the initial real or real-window datum from which the finite Picard run starts.*
2. *F is the contractive self-map row, carrying the displayed graph reads and the contraction witness consumed by* $\mathsf{ContractionRealModulus}^{\uparrow}$ *or the Picard sibling route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *contraction budget row, recording the dyadic radius, contraction slack, and finite request schedule used for the window.*
4. *W is the finite iterate-window row. It lists the displayed iterates from the source row through the chosen finite height and records the successor reads of F.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the source, iterate, and terminal-name windows opened by W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same observations and dyadic budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after X, F, D, W, S, R have all been displayed.*
8. *H records componentwise* hsame *transport for the source seal, contractive map, dyadic budget, iterate window, stream observations, regular readback, terminal seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Picard-window request through*

$$X, F, D, W, S \longmapsto R \longmapsto E \longmapsto H, Q, N.$$

10. *Q is the* Pkg *provenance over* FiniteRealPicardIteration$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionRealModulus$^{\uparrow}$, MetricCompletionFinite$^{\uparrow}$, BishopCompletionComparison$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{FiniteRealPicardIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, D, W, S, R, E, H, C, Q, N$. It has no coordinate for a completed Picard limit, fixed-point selection, quotient stream equality, countable-choice schedule, host function extensionality, ambient completeness of Real$^{\uparrow}$*, Lean verification, or bridge checking.*

Theorem 7627.2 (FiniteRealPicardIteration NameCert obligations)**.** *For every accepted* FiniteRealPicardIteration$^{\uparrow}$ *carrier*

$$P = (X, F, D, W, S, R, E, H, C, Q, N),$$

the local NameCert$_{\text{FiniteRealPicardIteration}^{\uparrow}}$ *surface consists exactly of source Real-seal admission, contractive-map exposure, dyadic contraction-budget exposure, finite iterate-window exposure, stream-window readback, regular rational readback, terminal Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier coordinates. The analytic rows are exactly the source seal X, contractive map row F, dyadic budget D, finite iterate window W, stream observations S, regular readback R, and terminal seal E.

The structural rows then preserve the same finite route. The transport row H moves each coordinate componentwise, while C replays only the displayed request path from the source seal through the iterate window and readback.

The provenance and local name rows Q, N source and name this packet inside the cited Picard, contraction, completion, dyadic, stream, regular-readback, and Real chapters. Since no other coordinate is present, the listed rows exhaust the NameCert obligation surface and the excluded limit, fixed-point, quotient, choice, extensionality, completeness, verification, and bridge data have no export path. □

Theorem 7627.3 (FiniteRealPicardIteration contractive window)**.** *Every contraction-facing or completion-facing consumer of an accepted* FiniteRealPicardIteration$^{\uparrow}$ *carrier must read the finite route*

$$X \longrightarrow F \longrightarrow D \longrightarrow W \longrightarrow S \longrightarrow R \longrightarrow E$$

before it may consume the terminal Real$^{\uparrow}$ *seal. The route exports a finite Picard-window certificate only; it does not export a completed fixed point, quotient equality of iterate streams, countable choice, host equality of maps, or an ambient completeness theorem.*

Proof. Start with the terminal seal request. The carrier definition places E after the source seal, contractive map, dyadic budget, finite iterate window, stream observations, and regular rational readback.

The contraction data are finite. The row F is read only against the displayed source row and the row D supplies the dyadic contraction budget for the finite height recorded by W.

The Real-facing handoff is downstream. The rows S and R expose the observed finite windows and regular rational readback before E is available, while H, C, Q, N transport, replay, source, and name the same route.

Thus every consumer receives only the finite Picard-window certificate. The absent fixed point, quotient stream equality, choice schedule, host map equality, and ambient completeness theorem cannot be recovered from the displayed packet. □

Theorem 7627.4 (FiniteRealPicardIteration Real seal nonescape)**.** *In every accepted* $\mathsf{FiniteRealPicardIteration}^{\uparrow}$ *carrier, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is a seal for the displayed finite route*

$$X, F, D, W, S, R, H, C, Q, N.$$

Any downstream Real or Completion consumer may cite E only as the endpoint of that route, and cannot detach it into ambient $\mathsf{Real}^{\uparrow}$ *completeness, host equality of completed reals, a selected Picard limit, or quotient equality of iterate streams.*

Proof. Read E from the carrier. It appears only after the finite Picard data X, F, D, W, S, R and is transported only through the componentwise row H.

Replay is equally local. The continuation row C repeats the same finite order of exposure, and the provenance and local naming rows Q, N record no further Real object.

Therefore the seal is terminal for the displayed finite packet. Since the carrier contains no coordinate for completeness, host equality, selected limits, or quotient stream equality, no consumer can detach E into one of those claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealPicardIteration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteRealPicardIteration}^{\uparrow}$ packet is generated from a source Real seal, a contractive map row, a dyadic contraction budget, a finite Picard iterate window, StreamName observations, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteRealPicardIteration}^{\uparrow}}$, seed)

Scope closed: Definition 7627.1, Definition 7627.2, Definition 7627.3, and Definition 7627.4 seed the finite Picard-window route over Real, contraction, dyadic budget, StreamName, RegSeqRat, completion comparison, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteRealPicardIteration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach fixed-point existence, completed Picard limits, quotient equality of iterate streams, countable choice, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contractive-map exposure, dyadic budget exactness, finite iterate-window replay, StreamName observation, RegSeqRat readback, Real-seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRealPicardIterationUp.

Chapter 7628

A Concrete Naming Certificate for $\mathsf{WeilUniformity}^{\uparrow}$

Following Weil and Kelley on the left and right uniformities of a topological group, the $\mathsf{WeilUniformity}^{\uparrow}$ packet records the finite BEDC surface on which the two entourage directions are read together. It sits between the entourage rows of chapter 339, the multiplication and inversion continuity rows of chapter 204, the open-neighbourhood source of chapter 88, and the completion-facing consumers of chapter 95 and chapter 2211. The packet explains why Raikov-style completion and topological-group completion require a two-sided uniformity surface rather than an anonymous uniform-space row.

Definition 7628.1 (Weil uniformity carrier)**.** *A* $\mathsf{WeilUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (G, T, U, L, R, M, I, C_L, C_R, Q, V, H, C, P, N)$$

with the following displayed rows:

1. *G is the underlying group-object row whose multiplication and unit are read only through the topological-group packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row supplying the open identity-neighbourhood requests.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row exposing finite entourage requests, diagonal rows, symmetry rows, and composition refinements.*
4. *L is the left-entourage ledger generated by rows of the form $x^{-1}y$ lying in a displayed identity neighbourhood.*
5. *R is the right-entourage ledger generated by rows of the form xy^{-1} lying in a displayed identity neighbourhood.*
6. *M records the* $\mathsf{TopGroup}^{\uparrow}$ *multiplication-continuity transport for the left and right ledgers.*
7. *I records the* $\mathsf{TopGroup}^{\uparrow}$ *inversion-continuity transport exchanging the left and right ledgers.*
8. *C_L is the* $\mathsf{Completion}^{\uparrow}$ *consumer row that can read Cauchy data only after the left ledger is displayed.*
9. *C_R is the* $\mathsf{UniformCompleteSpace}^{\uparrow}$ *consumer row that can read completion data only after the right ledger is displayed.*

10. *Q records the comparison boundary with* $\mathsf{HausdorffUniformity}^{\uparrow}$ *when separated completion or uniqueness consumers require diagonal separation.*
11. *V records the two-sided completion handoff used by Raikov-style consumers without selecting a quotient completion point.*
12. *H stores componentwise* hsame *transport for all displayed rows.*
13. *C stores displayed* Cont *replay for neighbourhood admission, left and right entourage generation, multiplication transport, inversion transport, completion handoff, and separated comparison.*
14. *P is the* Pkg *provenance over* $\mathsf{WeilUniformity}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{UniformCompleteSpace}^{\uparrow}$, $\mathsf{HausdorffUniformity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{WeilUniformity}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $G, T, U, L, R, M, I, C_L, C_R, Q, V, H, C, P, N$. It has no coordinate for a host topological group, a chosen completion, quotient equality of Cauchy filters, ambient uniform completeness, or a one-sided replacement for the two-sided uniformity route.

Theorem 7628.2 (Weil uniformity NameCert obligations)**.** *For every accepted* $\mathsf{WeilUniformity}^{\uparrow}$ *carrier*

$$W = (G, T, U, L, R, M, I, C_L, C_R, Q, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeilUniformity}^{\uparrow}}$ *surface consists exactly of carrier admission, identity-neighbourhood access, left-entourage generation, right-entourage generation, multiplication transport, inversion transport, completion-facing handoff, separated-uniform comparison, two-sided completion provenance, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion or topological-group consumer must read both L and R before it may inspect C_L, C_R, or V.*

Proof. Project the accepted carrier through Definition 7628.1. The rows G, T, U supply the group, topology, and ambient uniform-space requests, while L and R are distinct displayed ledgers for the left and right entourages.

Read the topological-group transport rows. Multiplication continuity in M keeps products compatible with both ledgers, and inversion continuity in I exchanges the two directions instead of allowing one direction to stand for the other.

The completion-facing rows C_L, C_R, Q, V occur only after the two ledgers have been displayed. The structural rows H, C, P, N then transport, replay, source, and name the same packet, so the listed obligations exhaust the local NameCert surface. □

Theorem 7628.3 (Weil uniformity topological-group handoff)**.** *Every* $\mathsf{TopGroup}^{\uparrow}$ *consumer of an accepted* $\mathsf{WeilUniformity}^{\uparrow}$ *packet reads multiplication and inversion through the finite route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{UniformSpace}^{\uparrow},\ L,\ R,\ M,\ I \longrightarrow \mathsf{Completion}^{\uparrow},\ \mathsf{UniformCompleteSpace}^{\uparrow}.$$

The handoff keeps the left and right entourage ledgers visible until completion-facing rows are reached. Consequently the packet may feed Raikov-style completion, separated-uniform comparison, and uniform-complete-space consumers, but it does not identify the carrier with a host completion, choose Cauchy representatives, assert ambient completeness, or collapse the two ledgers to a single anonymous uniform-space row.

Proof. Start with the topology and uniform-space source rows. The open identity-neighbourhood requests in T and the finite entourage data in U are the only source rows from which the left and right ledgers can be generated.

Use the $\mathsf{TopGroup}^{\uparrow}$ transport rows M and I. Multiplication reads both directions through the same finite packet, while inversion swaps the direction of an entourage request and therefore requires both L and R to remain present.

Finally inspect $C_L, C_R, Q, V, H, C, P, N$. These rows hand the two-sided packet to completion and separated-uniform consumers while preserving componentwise transport, replay, provenance, and local naming. Since no host completion or quotient representative is a coordinate of the carrier, the handoff cannot bypass the displayed two-sided route. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeilUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{WeilUniformity}^{\uparrow}$ packet is generated from a topological-group source, a topology row, a uniform-space row, left and right entourage ledgers, multiplication and inversion continuity transports, completion-facing handoff rows, separated-uniform comparison, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeilUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7628.1, Definition 7628.2, and Definition 7628.3 seed the two-sided uniformity surface between $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{UniformCompleteSpace}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeilUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, Raikov completion, quotient Cauchy-filter equality, selected completion points, ambient uniform completeness, host topological-group equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left and right entourage exactness, multiplication and inversion transport, completion handoff, separated comparison, two-sided non-escape, transport, replay, provenance, and local naming for BEDC.Derived.WeilUniformityUp.

Chapter 7629

A Concrete Naming Certificate for $\mathsf{CapProduct}^{\uparrow}$

Following Eilenberg–Steenrod and the chain-level treatments in Hatcher, the $\mathsf{CapProduct}^{\uparrow}$ packet records the finite action route by which a displayed cochain row acts on a displayed chain row before the resulting chain class is read. It sits between the homology carrier of chapter 98 and the cohomology carrier of chapter 99. The packet names chain, cochain, degree-window, boundary/coboundary compatibility, transport, replay, provenance, and local naming rows; it is not a full homology operation calculus, not a cup-product theory, not quotient chain equality, and not an ambient algebraic-topology theorem.

Definition 7629.1 (Cap product carrier)**.** *A* $\mathsf{CapProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, D, E, H, G, T, R, P, N)$$

with the following displayed rows:

1. *A is the chain row, recording the finite chain window on which the action is evaluated.*
2. *B is the cochain row, recording the finite cochain window and coefficient readback used by the action.*
3. *D is the degree-window row, naming the source chain degree, cochain degree, and output chain degree inspected by the packet.*
4. *E is the boundary/coboundary compatibility row, recording the displayed sign and face-boundary ledger used by the action check.*
5. *H is the* $\mathsf{Homology}^{\uparrow}$ *row read only after the chain row and compatibility row have been displayed.*
6. *G is the* $\mathsf{Cohomology}^{\uparrow}$ *row read only through the cochain row and coboundary side of E.*
7. *T records componentwise* hsame *transport for chain, cochain, degree-window, compatibility, homology, cohomology, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a cap-product request through $A, B, D, E, H, G, T, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CapProduct}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CapProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, E, H, G, T, R, P, N$. It has no coordinate for chosen chain representatives, quotient chain equality, unrestricted naturality, a cup-product algebra, or ambient homological algebra.

Theorem 7629.2 (Cap product NameCert obligations)**.** *For every accepted* $\mathsf{CapProduct}^{\uparrow}$ *carrier* $Q = (A, B, D, E, H, G, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CapProduct}^{\uparrow}}$ *surface consists exactly of chain-row admission, cochain-row admission, degree-window admission, boundary/coboundary compatibility, the downstream* $\mathsf{Homology}^{\uparrow}$ *read, the downstream* $\mathsf{Cohomology}^{\uparrow}$ *read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7629.1. The displayed tuple is the complete inventory of rows available to a cap-product consumer.

The chain and cochain sides are separated by A and B, while D fixes the degree window in which the action is inspected. The compatibility row E is the only place where the boundary and coboundary ledgers meet.

The rows H and G can be read only after those finite rows have been displayed. Finally T, R, P, N transport, replay, source, and name the same route, so no chosen representative, quotient equality, unrestricted naturality, cup-product algebra, or ambient homological-algebra principle is introduced. □

Theorem 7629.3 (Cap product chain-cochain handoff)**.** *Every accepted* $\mathsf{CapProduct}^{\uparrow}$ *consumer that hands a cohomology-facing row to a homology-facing row factors through*

$$A, B, D, E, T, R, P, N.$$

The exported handoff reads $\mathsf{Cohomology}^{\uparrow}$ *through the displayed cochain and coboundary side, reads* $\mathsf{Homology}^{\uparrow}$ *through the displayed chain and boundary side, and exposes only the finite compatibility route named by the carrier.*

Proof. Begin with a chain-cochain handoff request. Carrier admission exposes the chain row A, cochain row B, degree-window row D, and compatibility row E before either downstream algebraic-topology row may be consumed.

Read $\mathsf{Cohomology}^{\uparrow}$ through B and the coboundary side of E. Read $\mathsf{Homology}^{\uparrow}$ through A and the boundary side of E.

The structural rows T, R, P, N then preserve and replay this finite route. Since the carrier contains no selected representative, quotient equality, full operation calculus, or unrestricted naturality coordinate, the handoff is exactly the displayed chain-cochain route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CapProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{CapProduct}^{\uparrow}$ packet is generated from chain rows, cochain rows, degree-window rows, boundary/coboundary compatibility rows, HomologyUp and CohomologyUp consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CapProduct}^{\uparrow}}$, seed)

Scope closed: Definition 7629.1, Definition 7629.2, and Definition 7629.3 seed the finite chain-cochain action route between displayed HomologyUp and CohomologyUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CapProduct}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full homology operations, cup-product theory, quotient chain equality, unrestricted naturality, ambient homological algebra, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chain-row and cochain-row exposure, degree-window exposure, boundary/coboundary compatibility, homology and cohomology handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.CapProductUp.

Chapter 7630

A Concrete Naming Certificate for $\mathsf{WindowSixDecorationDefect}^{\uparrow}$

The $\mathsf{WindowSixDecorationDefect}^{\uparrow}$ packet realizes the defect-refined decoration polynomial of Definition 9462.1702 inside the DNA window-six private/shared boundary-tile route. It consumes the private/shared decoration packet of chapter 5881, the private/shared model packet of chapter 5896, and the Alexander-dual trigger algebra of chapter 3555. The packet records private/shared decomposition, shared row-set arity, defect exponents, polynomial coefficients, trigger coverage, transport, replay, provenance, and local naming as finite rows; it is not a biological mechanism, stochastic model, hidden enumeration oracle, or quotient codon semantics.

Definition 7630.1 (Window-six decoration defect carrier)**.** *A* $\mathsf{WindowSixDecorationDefect}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, A, E, K, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the private/shared decomposition row, read through the displayed private-block and shared-coordinate split.*
2. *S is the shared row-set arity row, assigning each shared coordinate to a finite row set of size at least two.*
3. *A is the defect exponent ledger, recording the contribution $r - 1$ attached to a shared row set of arity r.*
4. *E is the decoration-polynomial coefficient row, recording the finite coefficient data used by the defect-refined polynomial.*
5. *K is the trigger-coverage row, tying the decomposition to accepted window-six boundary-trigger coverage.*
6. *T records componentwise* hsame *transport for decomposition, arity, exponent, coefficient, trigger, replay, provenance, and local naming rows.*
7. *H records displayed* Cont *replay from a defect-polynomial request through B, S, A, E, K, T, P, N.*
8. *C is the* Pkg *provenance over* $\mathsf{WindowSixDecorationDefect}^{\uparrow}$, $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$, $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$, $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$, *the source theorem of Definition 9462.1702,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. *P is the trigger-polynomial readback row that keeps coefficient reads attached to displayed trigger coverage rather than to a hidden enumeration oracle.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WindowSixDecorationDefect}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, A, E, K, T, H, C, P, N$. It has no coordinate for biological semantics, stochastic fitness, quotient codon equality, hidden global enumeration, or unrecorded trigger search.

Theorem 7630.2 (Window-six decoration defect NameCert obligations)**.** *For every accepted* $\mathsf{WindowSixDecorationDefect}^{\uparrow}$ *carrier $W = (B, S, A, E, K, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WindowSixDecorationDefect}^{\uparrow}}$ *surface consists exactly of private/shared decomposition exposure, shared row-set arity exposure, defect-exponent exposure, decoration-polynomial coefficient exposure, trigger-coverage exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, trigger-polynomial readback, and local naming.*

Proof. Project the accepted carrier of Definition 7630.1. The decomposition, arity, exponent, coefficient, trigger, transport, replay, provenance, readback, and local naming data are precisely $B, S, A, E, K, T, H, C, P, N$.

The rows B, S expose the private/shared split and the finite arities of shared row sets. The row A records the defect contribution for each shared coordinate, and E records the coefficient ledger read by the polynomial.

The trigger row K keeps the polynomial attached to the accepted window-six coverage route. Finally T, H, C, P, N transport, replay, source, read back, and name the same finite packet, leaving no carrier coordinate for biological semantics, stochastic fitness, quotient codon equality, hidden enumeration, or unrecorded trigger search. □

Theorem 7630.3 (Window-six decoration defect polynomial handoff)**.** *Every accepted* WindowSixDecorationDefect *consumer that reads the defect-refined decoration polynomial must first expose*

$$B, S, A, E, K, T, H, C, P, N.$$

The handoff realizes Definition 9462.1702 as finite private/shared row-set arity, defect-exponent, coefficient, trigger-coverage, transport, replay, provenance, readback, and local NameCert rows.

Proof. Read the private/shared decomposition through B and the shared row-set arities through S. These rows determine which shared coordinates contribute to the defect ledger.

Use A to expose the exponent contribution attached to each shared arity and E to expose the corresponding coefficient row. The trigger row K ensures that the coefficient read remains tied to the window-six boundary-trigger surface.

The remaining rows T, H, C, P, N transport, replay, source, read back, and locally name the same finite route. Hence the consumer receives the finite polynomial handoff described by the theorem rather than a hidden enumeration oracle or external biological semantics. □

Falsifiable prediction: Every accepted $\mathsf{WindowSixDecorationDefect}^{\uparrow}$ consumer must display private/shared decomposition rows, shared row-set arity rows, defect-exponent ledger rows, coefficient rows, trigger-coverage rows, hsame transport, Cont replay, Pkg provenance, trigger-polynomial readback, and local NameCert rows before the defect-refined decoration polynomial is read.

Independence witness: window-six-private-shared-decoration, private-shared-decoration-model, alexander-dual-trigger-algebra, dna-window-six-boundary-tile, dna-boundary-trigger-closure: WindowSixDecoratior

is not bijective to any listed sibling because it jointly carries arity, exponent, coefficient, trigger, transport, replay, provenance, readback, and local naming rows for the defect-refined polynomial.

The Lean TasteGate route for `BEDC.Derived.WindowSixDecorationDefectUp` must provide explicit Nontrivial WindowSixDecorationDefect$^{\uparrow}$ and FieldFaithful WindowSixDecorationDefect$^{\uparrow}$ instances preserving $B, S, A, E, K, T, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{WindowSixDecorationDefect}^{\uparrow}}$.

Constructive story: A WindowSixDecorationDefect$^{\uparrow}$ packet is generated from private/shared decomposition rows, shared row-set arity rows, defect-exponent ledgers, decoration-polynomial coefficient rows, trigger-coverage rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, trigger-polynomial readback, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{WindowSixDecorationDefect}^{\uparrow}}$, seed)

Scope closed: Definition 7630.1, Definition 7630.2, and Definition 7630.3 seed the finite defect-refined decoration polynomial packet over displayed private/shared, trigger, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{WindowSixDecorationDefect}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon equality, hidden global enumeration, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private/shared decomposition, shared row-set arity, defect-exponent ledger, coefficient exposure, trigger-coverage routing, TasteGate nontriviality, field faithfulness, transport, replay, provenance, readback, and local naming for BEDC.Derived.WindowSixDecorationDefectUp.

Chapter 7631

A Concrete Naming Certificate for $\mathsf{ConstructiveContractionModulus}^{\uparrow}$

Following Bishop and Bridges constructive metric fixed-point theory, the $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ packet records the finite modulus data a Picard or Banach contraction consumer must display before a Cauchy tail is handed to completion. It sits after the complete-metric boundary of chapter 128, the continuous-map row of chapter 130, the Cauchy-modulus ledger of chapter 292, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The carrier names contraction shrinkage as displayed finite data; it does not select a fixed point, quotient a completion, use countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7631.1 (Constructive contraction-modulus carrier)**.** *A* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, L, Q, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row naming the metric source on which the Picard route is allowed to ask for completion-facing data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row, displayed as a graph or local evaluation ledger rather than as an extensional host function.*
3. *L is the contraction ledger, recording the finite shrinkage factor or stagewise contraction budget consumed by later tail estimates.*
4. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row produced from L and the displayed iterate-distance windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for finite Picard prefixes and tolerance requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distances and tail comparisons named by Q.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after X, F, L, Q, W, R have been replayed.*
8. *H records componentwise* hsame *transport for the metric, self-map, contraction ledger, modulus, window, readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a contraction request through the finite Picard windows, modulus lookup, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveContractionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected fixed point, quotient completion equality, host function extensionality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ConstructiveContractionModulusUp.ConstructiveContractionModulusTasteGate`
`single carrier alignment`

Theorem 7631.2 (Constructive contraction-modulus NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ *carrier* $M = (X, F, L, Q, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveContracti}}$ *surface consists exactly of carrier admission for the complete-metric boundary, continuous self-map row, contraction ledger, CauchyModulus row, finite window row, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming. Every Picard or fixed-point consumer must read* $X, F, L, Q, W, R, E, H, C, P, N$ *in that displayed route before it can consume the contraction modulus.*

Proof. Project the carrier of Definition 7631.1. The displayed tuple contains the complete-metric source, continuous self-map row, contraction ledger, Cauchy modulus, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Read the contraction route. The self-map row F is meaningful only over the metric boundary X, and the shrinkage ledger L is the only row from which the CauchyModulus row Q can be read.

The window and readback rows W, R then expose the finite Picard prefixes and rational tail comparisons. Transport, replay, provenance, and local naming preserve this same route, so no further NameCert obligation is stored outside the listed rows. □

Theorem 7631.3 (Constructive contraction-modulus Picard Cauchy route)**.** *Every Picard or Banach fixed-point consumer of an accepted* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ *packet must factor through*

$$\mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto L \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route supplies a finite Cauchy-tail modulus for downstream Picard iteration and fixed-point packets. It exports no selected fixed point, quotient completion equality, host equality of maps, countable-choice schedule, or ambient completeness theorem beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. Begin with a fixed-point-facing request. The only metric source is X, and the only self-map data is the displayed continuous-map row F.

The contraction ledger L is then read as the finite shrinkage budget. The CauchyModulus row Q is obtained from that ledger and can be inspected only through the finite StreamName windows W and RegSeqRat readback R.

Finally the terminal Real seal E is downstream of the same replay. Since the carrier contains no fixed-point selector, quotient completion equality, host map equality, or choice schedule, the Picard consumer receives only the displayed Cauchy-tail route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveContractionModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ packet is generated from a CompleteMetric boundary, a ContinuousMap self-map row, a finite contraction ledger, a CauchyModulus row, StreamName windows, RegSeqRat

readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveContractionModulus}\uparrow}$, seed)

Scope closed: Definition 7631.1, Definition 7631.2, and Definition 7631.3 seed the finite contraction-modulus route consumed by Picard and fixed-point packets.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveContractionModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close fixed-point existence, selected limits, quotient completion equality, host function extensionality, countable choice, ambient Real completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-ledger exactness, CauchyModulus extraction, StreamName and RegSeqRat readback stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveContractionModulusUp.

Chapter 7632

A Concrete Naming Certificate for $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$

The $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ packet realizes the conditional MetaCIC confluence audit frontier of chapter 9378 as a finite scheduler carrier. It consumes the audit rows of chapter 743, the normal-fragment rows of chapter 737, the closed-normal seal of chapter 741, classifier boundary rows of chapter 595, and subject-reduction rows of chapter 483. The packet separates component diamonds, residual boundaries, atom joins, closed-source guards, and star-confluence handoff rows without exporting closed star confluence as an unconditional theorem.

Definition 7632.1 (MetaCIC parallel-diamond scheduler carrier)**.** *A* $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, B, A, G, T, H, C, P, N)$$

with the following visible rows:

1. *D is the component-diamond row listing the checked local parallel-diamond fragments available to the scheduler.*
2. *B is the residual-substitution boundary row retained from the confluence audit map.*
3. *A is the atom-join row for closed atom fragments already accepted by the normal-fragment and closed-normal seal packets.*
4. *G is the closed-source guard row, ensuring that a star-confluence request is made only after the closed-source hypotheses have been displayed.*
5. *T is the star-confluence handoff row, exposed as a conditional target whose residual-substitution boundary is still visible.*
6. *H records componentwise* hsame *transport for the component-diamond, residual-boundary, atom-join, closed-guard, handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an audit request through the scheduler rows.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$*, the MetaCIC confluence audit packet, the MetaCIC confluence normal-fragment packet,* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$*,* $\mathsf{ClassifierBoundary}^{\uparrow}$*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}}$ *row.*

The classifier compares accepted schedulers only through $D, B, A, G, T, H, C, P, N$. It has no coordinate for unconditional closed star confluence, global substitution compatibility, erased residual data, or a host confluence theorem.

Theorem 7632.2 (MetaCIC parallel-diamond scheduler NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ *carrier* $S = (D, B, A, G, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDia}}$ *surface consists exactly of component-diamond exposure through D, residual-boundary exposure through B, atom-join exposure through A, closed-source guarding through G, star-confluence handoff through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the scheduler carrier. The displayed rows are precisely the component-diamond, residual-boundary, atom-join, closed-source guard, conditional handoff, transport, replay, provenance, and local naming rows.

Read the audit route. Component diamonds in D and atom joins in A may be scheduled only with the residual boundary B still present, and the closed-source guard G controls when the star-confluence handoff T is exposed.

Transport, replay, provenance, and naming preserve those same rows. Since no carrier coordinate erases B or replaces it by a global substitution theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 7632.3 (MetaCIC parallel-diamond residual boundary)**.** *No* $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ *consumer may export closed star confluence unless its route displays an explicit residual-substitution boundary row. The visible handoff is*

$$D, B, A, G \longmapsto T,$$

and the row B remains part of the exported scheduler packet.

Proof. Start from an accepted scheduler $S = (D, B, A, G, T, H, C, P, N)$. The only star-confluence-facing coordinate is T.

The carrier definition places T downstream of the component-diamond row D, residual-boundary row B, atom-join row A, and closed-source guard G. Thus the handoff cannot be read without B.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. They do not add a substitution-compatibility theorem or remove the residual row, so closed star confluence is never exported without the displayed boundary. □

Falsifiable prediction: No $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ scheduler may export closed star confluence without an explicit residual-substitution boundary row.

Independence witness: metacic-confluence-audit, metacicconfluencenormalfragment, closednormalconfluenceseal, classifierboundary, subjectreduction: $\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries component-diamond rows, residual-substitution boundary rows, atom-join rows, closed-source guards, star-confluence handoff, transport, replay, provenance, and local NameCert data.

Lean development for this route must later supply $\mathsf{Nontrivial}\ \mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}$ TasteGate instances after the residual-substitution boundary is represented as a displayed scheduler row.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondScheduler}^{\uparrow}}$.

Constructive story: A MetaCICParallelDiamondScheduler$^{\uparrow}$ packet is generated from component-diamond rows, residual-substitution boundary rows, atom-join rows, closed-source guards, star-confluence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICParallelDiamondScheduler}\uparrow}$, seed)

Scope closed: Definition 7632.1, Definition 7632.2, and Definition 7632.3 seed the finite scheduler surface for the conditional MetaCIC confluence audit map.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICParallelDiamondScheduler}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global substitution compatibility, unconditional closed star confluence, host confluence, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, component-diamond exposure, residual-boundary retention, atom-join exposure, closed-source guarding, star-confluence handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetacicParallelDiamondSchedulerUp.

Chapter 7633

A Concrete Naming Certificate for $\mathsf{SierpinskiGasket}^{\uparrow}$

Following Sierpinski and Hutchinson-style iterated function systems, the $\mathsf{SierpinskiGasket}^{\uparrow}$ packet records the triangular three-map self-similar gasket as finite BEDC naming data. It sits after the finite contraction-family surface of chapter 7342, beside the stagewise fractal routes of chapter 3995 and chapter 6486, and before any consumer asks for a completed attractor subset. The packet names only triangular vertex data, three displayed contraction rows, finite address windows, stage-cell readback, dyadic coordinate seals, transport, replay, provenance, and local naming; it does not import an ambient planar compact set, a hyperspace fixed point, Hausdorff dimension, area measure, selected infinite branch, or host equality of points.

Definition 7633.1 (Sierpinski gasket carrier)**.** *A* $\mathsf{SierpinskiGasket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, I, F, A, S, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the triangular vertex row, recording the three displayed corner seals and the orientation used by stage cells.*
2. *I is the* $\mathsf{IteratedFunctionSystem}^{\uparrow}$ *dependency row. It supplies the compact complete metric source, finite map family, contraction ledger, modulus thresholds, and Real-facing constants inherited from chapter 7342.*
3. *F is the three-map contraction-family row, addressed by the finite indices $0, 1, 2$ and read only through the IFS packet I.*
4. *A is the finite address row for inspected words over the three symbols, with no selected infinite branch.*
5. *S is the stage-cell row. It records the displayed triangular subcell reached by an address word and the three endpoint rows carried by that cell.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the inspected address, stage, and endpoint-coordinate reads.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *stage and midpoint ledger used to route the half-scale triangular contractions and coordinate tolerances.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled coordinate windows.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *coordinate-seal family for the displayed vertices and stage-cell endpoints.*
10. *H records componentwise* hsame *transport for $T, I, F, A, S, W, D, R, E, C, P, N$.*
11. *C records displayed* Cont *replay from a gasket request through the finite address, contraction-family, stage-cell, coordinate, and structural rows.*
12. *P is the* Pkg *provenance over* $\mathsf{SierpinskiGasket}^{\uparrow}$, $\mathsf{IteratedFunctionSystem}^{\uparrow}$, $\mathsf{KochCurve}^{\uparrow}$, $\mathsf{KochSnowflake}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}$ *row.*

The classifier accepts a packet only through the displayed tuple $T, I, F, A, S, W, D, R, E, H, C, P, N$. It has no coordinate for a completed attractor subset, hyperspace quotient, Hausdorff dimension, area computation, selected infinite address, or ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.SierpinskiGasketUp.SierpinskiGasketTasteGate single carrier alignment`

Theorem 7633.2 (Sierpinski gasket NameCert obligations)**.** *For every accepted* $\mathsf{SierpinskiGasket}^{\uparrow}$ *carrier*

$$G = (T, I, F, A, S, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; triangular vertex exposure through T; finite IFS exposure through I; three contraction rows through F; finite address exposure through A; stage-cell exactness through S; finite window scheduling through W; dyadic stage and midpoint exactness through D; regular rational readback through R; terminal* $\mathsf{Real}^{\uparrow}$ *coordinate sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7633.1. The visible rows are exactly $T, I, F, A, S, W, D, R, E, H, C, P, N$, and the classifier compares only those rows by componentwise hsame.

The IFS-facing obligations are exhausted by I and F. The dependency row I supplies the finite contraction-family interface named in chapter 7342, while F fixes the three displayed half-scale maps by finite index.

The gasket-specific readback is exhausted by A, S, W, D, R, E. A consumer first reads a finite address word, then the corresponding displayed stage cell, then the scheduled coordinate windows and dyadic midpoint ledger before the regular rational and Real-seal rows are inspected.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no completed attractor subset, hyperspace quotient, dimension row, area row, selected infinite branch, or ambient compactness row, no such object can become a local $\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}$ obligation. □

Theorem 7633.3 (Sierpinski gasket finite IFS route)**.** *Every accepted stage-cell, address, or coordinate read of a* $\mathsf{SierpinskiGasket}^{\uparrow}$ *packet factors through the finite route*

$$T \longmapsto I, F \longmapsto A, S \longmapsto W, D, R, E \longmapsto H, C, P, N.$$

The route exports only the triangular vertex row, finite IFS dependency, three contraction rows, finite address and stage-cell rows, StreamName windows, dyadic midpoint ledger, regular rational readback, terminal Real seals, and the structural rows. It exports no completed planar fractal, host attractor subset, hyperspace fixed point, selected infinite branch, area computation, Hausdorff dimension, or ambient compactness principle.

Proof. Begin with an accepted consumer of a gasket stage-cell, address, or coordinate read. By Definition 7633.1, the consumer can enter the packet only through the triangular vertex row T, the IFS row I, and the three-map row F.

Apply Definition 7633.2. It forces every finite address read through A, every stage-cell read through S, and every coordinate read through W, D, R, E before the structural rows are inspected.

The IFS packet of chapter 7342 supplies only a finite contraction-family route, and the nearby Koch packets supply only displayed stagewise readbacks. None adds a carrier coordinate for a completed attractor, hyperspace fixed point, selected infinite branch, dimension, area, or compactness theorem. Hence every accepted consumer remains on the finite IFS route. □

Theorem 7633.4 (Sierpinski gasket nonescape)**.** *An accepted* $\mathsf{SierpinskiGasket}^{\uparrow}$ *consumer cannot reach a Real coordinate seal for a stage-cell endpoint without first displaying* $T, I, F, A, S, W, D, R, E$*. In particular, any coordinate read that skips the finite address row, the three-map IFS row, the stage-cell row, or the dyadic midpoint ledger is outside the local NameCert surface.*

Proof. Read the accepted carrier through Definition 7633.1. The terminal coordinate seals occur only in the row E, and E is downstream of the displayed stage and coordinate rows A, S, W, D, R.

Use Definition 7633.3. The route places T, I, F, A, S, W, D, R before E, so a Real coordinate seal has no independent entry point.

The structural rows H, C, P, N can only transport, replay, source, and name the same packet. They do not create a second coordinate-seal route, and the carrier has no host planar subset or selected attractor row that could bypass the finite address and stage-cell data. □

Falsifiable prediction: An accepted $\mathsf{SierpinskiGasket}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ coordinate seal without first displaying the triangular vertex row, IFS row, three contraction rows, finite address row, stage-cell row, StreamName windows, dyadic midpoint ledger, and RegSeqRat readback refutes the carrier surface.

Independence witness: iteratedfunctionsystem, kochcurve, kochsnowflake, streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole $\mathsf{SierpinskiGasket}^{\uparrow}$ packet, because the gasket certificate binds triangular vertices, three half-scale contraction rows, finite address and stage-cell data, coordinate readback, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SierpinskiGasket}^{\uparrow}$ packet is generated from triangular vertex data, an IteratedFunctionSystem dependency row, three contraction rows, finite address and stage-cell rows, StreamName windows, dyadic midpoint ledgers, RegSeqRat readbacks, RealUp coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7633.1, Definition 7633.2, Definition 7633.3, and Definition 7633.4 seed the finite triangular IFS gasket carrier and its displayed coordinate nonescape route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiGasket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host attractor subsets, hyperspace completion, Hausdorff-metric fixed-set existence, invariant measures, area or dimension computation, selected infinite branches, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires a Lean-side SierpinskiGasketUp carrier with checked triangular vertex exposure, three contraction rows, finite address and stage-cell exactness, coordinate readback, Real coordinate sealing, transport, replay, provenance, and local naming.

Chapter 7634

A Concrete Naming Certificate for $\mathsf{DoldKan}^{\uparrow}$

Following Dold and Kan on the equivalence between simplicial abelian groups and nonnegatively graded chain complexes, the $\mathsf{DoldKan}^{\uparrow}$ packet records the finite BEDC naming surface for the normalized-chain handoff. It sits after the face-and-degeneracy ledger of chapter 157, the chain comparison route of chapter 7625, the cycle-boundary consumer of chapter 98, and the additive exactness rows of chapter 176. The packet names only displayed simplicial-object rows, normalized-chain rows, abelian-category rows, homology readback, componentwise transport, replay, provenance, and a local naming row; it does not select quotient representatives, assert a host categorical equivalence, import ambient derived-category machinery, or choose an unbounded normalization procedure.

Definition 7634.1 (Dold-Kan carrier)**.** *A* $\mathsf{DoldKan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, N, C, H, E, R, T, Q, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{SimplicialSet}^{\uparrow}$ *source row, specialized to the visible simplicial abelian-object window inspected by the packet.*
2. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *ambient additive row in which kernels, cokernels, sums, and zero morphisms are read.*
3. *N is the normalized-chain row, recording the finite intersection of displayed degeneracy kernels and the quotient-free normalized endpoint used by the consumer.*
4. *C is the chain-complex row whose differential is read from the displayed face maps and whose square-zero check is routed through the same finite window.*
5. *H is the* $\mathsf{Homology}^{\uparrow}$ *readback row reached only after the normalized-chain and differential rows have been displayed.*
6. *E records the Eilenberg-Zilber compatibility row consumed from chapter 7625 when tensor or product consumers ask for a chain-level comparison.*
7. *R stores componentwise* hsame *transport for the simplicial-object, abelian-category, normalized-chain, differential, homology, comparison, provenance, and local naming rows.*

8. *T stores displayed* Cont *replay from a finite simplicial window through normalization, differential readback, homology handoff, and comparison rows.*
9. *Q is the* Pkg *provenance row over* $\mathsf{DoldKan}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{EilenbergZilber}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *P records the public dependency boundary: every accepted read passes through the sibling chapters named above before a consumer may inspect the normalized-chain handoff.*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{DoldKan}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed tuple $S, A, N, C, H, E, R, T, Q, P, L$.

Theorem 7634.2 (Dold-Kan NameCert obligations)**.** *For every accepted* $\mathsf{DoldKan}^{\uparrow}$ *carrier $D = (S, A, N, C, H, E, R, T, Q, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{DoldKan}^{\uparrow}}$ *surface consists exactly of simplicial-object admission, abelian-category source admission, normalized-chain row admission, differential readback, homology handoff, Eilenberg-Zilber compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, dependency-boundary exposure, and local naming. A consumer cannot read the normalized-chain handoff without first exposing the* $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, *and* $\mathsf{EilenbergZilber}^{\uparrow}$ *sibling surfaces named by the carrier.*

Proof. Project the accepted carrier through Definition 7634.1. The source rows S and A expose the finite simplicial-object window and the additive ambient rows before any normalized chain row is reachable.

Next read N and C. The normalized-chain endpoint is tied to the displayed degeneracy-kernel window, and the differential row is tied to the displayed face maps in the same packet. The homology row H can therefore be consumed only after those rows have been displayed.

Finally inspect E, R, T, Q, P, L. The comparison row E supplies the Eilenberg-Zilber sibling route, while transport, replay, provenance, dependency boundary, and local naming keep all reads on the same finite carrier. Since no coordinate supplies host equivalence, quotient representative choice, derived-category machinery, or unbounded normalization, the listed rows exhaust the local NameCert obligations. □

Theorem 7634.3 (Dold-Kan normalized-chain handoff)**.** *Every accepted* $\mathsf{DoldKan}^{\uparrow}$ *packet exports its normalized-chain handoff through the finite route*

$$\mathsf{SimplicialSet}^{\uparrow},\ \mathsf{AbelianCat}^{\uparrow} \longmapsto N, C \longmapsto \mathsf{Homology}^{\uparrow} \longmapsto \mathsf{EilenbergZilber}^{\uparrow} \longmapsto R, T, Q, P, L.$$

The route reads finite face and degeneracy rows, additive kernel and zero rows, the normalized-chain endpoint, the differential row, homology readback, Eilenberg-Zilber comparison, componentwise transport, replay, provenance, and local naming before any downstream chain or homology consumer receives the handoff.

Proof. Begin with the carrier rows S and A. They are the only source rows for the simplicial object and abelian ambient data, so a normalized-chain request first opens those finite windows.

Use the normalized-chain and chain rows N and C. These rows display the degeneracy-kernel intersection and the differential readback before the homology row H can be inspected.

Attach the comparison and structural rows. The Eilenberg-Zilber row E is downstream of the chain window, and R, T, Q, P, L transport, replay, source, bound, and name the same packet. Thus the handoff remains finite and cannot be replaced by a host Dold-Kan equivalence, a quotient-chain representative, or an unbounded normalization oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DoldKan}^{\uparrow}}$.

Constructive story: A $\mathsf{DoldKan}^{\uparrow}$ packet is generated from displayed SimplicialSetUp face-and-degeneracy rows, AbelianCatUp additive rows, normalized-chain rows, chain differential rows, HomologyUp readback, EilenbergZilberUp comparison, componentwise hsame transport, Cont replay, Pkg provenance, a dependency boundary, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoldKan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7634.1, Definition 7634.2, and Definition 7634.3 seed the finite normalized-chain handoff over $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, and $\mathsf{EilenbergZilber}^{\uparrow}$ sibling surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoldKan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, a host categorical equivalence, quotient chain equality, selected chain representatives, derived-category exactness, ambient homological algebra, unbounded normalization, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, normalized-chain row exactness, differential readback, homology handoff, Eilenberg-Zilber compatibility, transport, replay, provenance, dependency-boundary exposure, and local naming for BEDC.Derived.DoldKanUp.

Chapter 7635

A Concrete Naming Certificate for $\mathsf{StoneCechRemainder}^{\uparrow}$

Following Gillman–Jerison and Engelking on Stone–Cech compactification remainders, the $\mathsf{StoneCechRemainder}^{\uparrow}$ packet records the compactification-boundary object read from a displayed $\mathsf{StoneCechCompactification}^{\uparrow}$ source without forming a host set difference or quotient compactification equality. It sits after the Stone–Cech compactification carrier of chapter 2344, the topology rows of chapter 88, the compact rows of chapter 60, and the compact-metric witness discipline of chapter 129. The packet names only the local boundary ledger for the complement-facing read; it does not assert an ambient corona space, select ultrafilters, identify quotient points, or import standard open-cover compactness.

Definition 7635.1 (Stone–Cech remainder carrier)**.** *A* $\mathsf{StoneCechRemainder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, D, C, Q, T, K, H, L, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *source row, including the bounded-continuous-probe windows and compact-target seal rows already named in chapter 2344.*
2. *D is the dense embedding and source-object row, recording the displayed source carrier, its generated topology, and the embedding route into the compactification packet.*
3. *C is the compactification boundary row, naming the boundary-facing read attached to B and D without identifying it with a host complement.*
4. *Q is the local complement and remainder ledger, recording which finite probes see the source row and which finite probes are routed to the remainder-facing boundary row.*
5. *T is the* $\mathsf{Topology}^{\uparrow}$ *row for the generated neighbourhood and closure tests used by the boundary ledger.*
6. *K is the* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *compact-target row consumed by the compactification source and boundary readback.*
7. *H records componentwise* hsame *transport for the compactification source, dense embedding, boundary row, complement ledger, topology row, compact row, continuation, provenance, and local naming rows.*
8. *L records displayed* Cont *replay from the dense source route through the Stone–Cech compactification row to the boundary and remainder ledger.*

9. *P is the* Pkg *provenance over* StoneCechRemainder$^{\uparrow}$, StoneCechCompactification$^{\uparrow}$, Topology$^{\uparrow}$, Compact$^{\uparrow}$, CompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{StoneCechRemainder}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host set difference, quotient compactification equality, an ultrafilter selector, ambient corona-space equality, selected boundary points, or standard open-cover compactness.

Theorem 7635.2 (Stone–Cech remainder NameCert obligations)**.** *For every accepted* StoneCechRemainder$^{\uparrow}$ *carrier $R = (B, D, C, Q, T, K, H, L, P, N)$, the local* NameCert$_{\text{StoneCechRemainder}^{\uparrow}}$ *surface consists exactly of carrier admission,* StoneCechCompactification$^{\uparrow}$ *source exposure through B, dense embedding and source-object exposure through D, compactification boundary exposure through C, local complement ledger exposure through Q,* Topology$^{\uparrow}$ *and* Compact$^{\uparrow}$ *row exposure through T, K, componentwise* hsame *transport through H, displayed* Cont *replay through L,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7635.1. The only analytic rows are the Stone–Cech compactification source, dense embedding, boundary row, complement ledger, topology row, and compact row.

The source row B is read through the existing compactification certificate of chapter 2344. The embedding row D and boundary row C then display the compactification-boundary route, while Q records the finite local complement ledger used by remainder-facing probes.

Finally inspect H, L, P, N. These rows transport, replay, source, and name the same displayed packet. Since none of them stores host set difference, quotient compactification equality, ultrafilter choice, ambient corona equality, selected boundary points, or standard open-cover compactness, the listed rows exhaust the local NameCert surface. □

Theorem 7635.3 (Stone–Cech remainder boundary non-escape)**.** *Every boundary-facing consumer of an accepted* StoneCechRemainder$^{\uparrow}$ *packet must read the finite route*

$$B, D, T, K \longmapsto C, Q \longmapsto H, L, P, N.$$

Thus the remainder boundary is exposed only through the displayed Stone–Cech compactification source, dense embedding, topology and compact rows, local complement ledger, componentwise transport, continuation replay, provenance, and local naming. It exports no host set difference, quotient compactification equality, ultrafilter selector, ambient corona-space equality, selected boundary point, or standard open-cover compactness theorem.

Proof. Start with the carrier of Definition 7635.1. A boundary-facing read has access to C and Q only as rows of the displayed tuple, so it cannot begin from a host complement or an external corona-space object.

Apply the NameCert obligations of Definition 7635.2. They require the compactification source row B, dense embedding row D, topology row T, and compact row K to be exposed before the boundary row and complement ledger are public.

The structural rows H, L, P, N preserve that route. Transport is componentwise, replay follows the displayed continuation, provenance cites the same source packets, and local naming adds no extra coordinate. Therefore every boundary read remains inside the finite route $B, D, T, K \longmapsto C, Q \longmapsto H, L, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{StoneCechRemainder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneCechRemainder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7635.1, Definition 7635.2, and Definition 7635.3 seed the compactification-boundary packet over Stone–Cech compactification, dense embedding, topology, compactness, local complement ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneCechRemainder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host set difference, quotient compactification equality, ultrafilter selection, ambient corona-space equality, selected boundary points, standard open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, compactification-source exposure, dense-embedding exposure, boundary-ledger exactness, topology and compact row stability, transport, replay, provenance, and local naming for the same packet.

Constructive story: A $\mathsf{StoneCechRemainder}^{\uparrow}$ packet is generated from a StoneCechCompactification row, a dense embedding and source-object row, a compactification boundary row, a local complement ledger, Topology and Compact rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7636

A Concrete Naming Certificate for $\mathsf{NewtonInterpolation}^{\uparrow}$

Following Newton's divided-difference interpolation, the $\mathsf{NewtonInterpolation}^{\uparrow}$ packet records the finite route from ordered interpolation nodes and displayed values to the Newton-basis polynomial row whose coefficients are the divided differences. It sits beside the basis-polynomial route of chapter 4206 and the confluent jet route of chapter 6668, while consuming the scalar, coefficient-list, and finite-table surfaces of chapter 45, chapter 50, and chapter 49. The packet names only finite node, value, divided-difference, polynomial-row, transport, replay, provenance, and local naming data.

Definition 7636.1 (Newton interpolation carrier)**.** *A* $\mathsf{NewtonInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, V, T, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is a nonempty finite node list with an order row and pairwise-distinctness ledger over the displayed* $\mathsf{Field}^{\uparrow}$ *scalar source.*
2. *V is the value list, indexed by the same finite node list and classified together with X under componentwise endpoint transport.*
3. *T is the divided-difference table, stored as a finite triangular* $\mathsf{Mat}^{\uparrow}$ *row whose first column is V and whose higher columns are computed from displayed field subtraction and inverse rows.*
4. *R is the interpolation polynomial row, assembling the Newton basis products*

$$1, \quad (Z - X_0), \quad (Z - X_0)(Z - X_1), \quad \ldots$$

   with coefficients read from the top row of T as a normalized $\mathsf{Poly}^{\uparrow}$ *coefficient list.*
5. *A is the classifier-transport and agreement row, recording that the polynomial row evaluates to the displayed values at the node list and agrees with the sibling* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *route whenever the same finite node-value ledger is read.*
6. *H records componentwise* hsame *transport for nodes, values, table entries, basis products, polynomial coefficients, agreement rows, replay routes, provenance, and local naming.*
7. *C records displayed* Cont *replay for forming table entries, extending the polynomial by one Newton basis factor, evaluating at displayed nodes, and reading the sibling interpolation anchors.*

8. *P is the* Pkg *provenance over* $\mathsf{NewtonInterpolation}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{HermiteInterpolation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{NewtonInterpolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets by transporting exactly these rows through H and replaying the same finite table and polynomial-row routes through C and P. It has no coordinate for a host polynomial function, unstated matrix inverse, quotient polynomial equality, infinite node family, analytic limiting table, or field-completion principle.

Theorem 7636.2 (Newton interpolation NameCert obligations)**.** *For every accepted* $\mathsf{NewtonInterpolation}^{\uparrow}$ *carrier $W = (X, V, T, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NewtonInterpolation}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the finite node list, value list, divided-difference table, interpolation polynomial row, classifier transport, displayed replay, package provenance, and local naming. The table row T must read* $\mathsf{Field}^{\uparrow}$ *subtraction and inverse data before any higher divided difference is exposed; the polynomial row R must read* $\mathsf{Poly}^{\uparrow}$ *coefficient-list assembly before a consumer obtains the interpolant; and the agreement row A may compare with the sibling* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *and* $\mathsf{HermiteInterpolation}^{\uparrow}$ *anchors only through their displayed finite interpolation rows.*

Proof. Project the carrier of Definition 7636.1. The rows X and V supply the ordered node-value ledger, while T stores the triangular divided-difference table as finite matrix data.

The polynomial row R is downstream of T. Each coefficient is read from the displayed top row of the table, and each Newton basis product is assembled as a finite $\mathsf{Poly}^{\uparrow}$ coefficient-list operation.

The agreement row A records only classifier transport and sibling comparison. It may point to the Lagrange node-value route or to the Hermite confluent route, but it does not import a host interpolation theorem or an external analytic table.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no additional coordinate for quotient polynomial equality, hidden matrix inversion, infinite node families, selected Real representatives, or field completion, the listed rows exhaust the local NameCert obligations. □

Theorem 7636.3 (Newton divided-difference stability)**.** *Let $W = (X, V, T, R, A, H, C, P, N)$ be an accepted* $\mathsf{NewtonInterpolation}^{\uparrow}$ *carrier. Any transported read of the node list, value list, or top-row divided-difference coefficients determines the same polynomial row after replaying the displayed* hsame *and* Cont *routes. In particular, extending the finite node-value ledger by one displayed node appends one triangular column of T and one Newton basis factor to R, without changing the already displayed lower-stage polynomial rows.*

Proof. Start with the componentwise transport row H. It transports the node list, value list, table entries, basis products, and polynomial coefficients together, so a transported divided-difference entry remains attached to the same finite table position.

Replay C. The recurrence for the next divided difference uses only displayed field subtraction and inverse rows from the existing node list, and the appended polynomial term uses the next top-row coefficient with the next finite Newton basis product.

The lower-stage rows are already named in T and R. Adding a later table column and basis factor appends data to the finite carrier but does not rewrite the earlier table entries or coefficient-list reads.

The agreement and provenance rows A, P, N then name the transported packet as the same Newton interpolation surface. Thus divided-difference stability is a statement about finite carrier transport and replay, not about host polynomial equality, an analytic limiting process, or a hidden matrix inverse. □

Falsifiable prediction: Every accepted $\mathsf{NewtonInterpolation}^{\uparrow}$ read displays a finite node list, value list, divided-difference table, interpolation polynomial row, classifier transport, $\mathsf{Field}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{HermiteInterpolation}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows before any Newton interpolant is available.

Independence witness: field, polynomial, matrix, lagrange-interpolation, hermite-interpolation: $\mathsf{NewtonInterpolation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an ordered node-value ledger, a triangular divided-difference table, a Newton-basis polynomial row, sibling agreement rows, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{NewtonInterpolation}^{\uparrow}$ packet is generated from an ordered finite node list, displayed value list, triangular divided-difference table, Newton-basis polynomial row, classifier transport, Lagrange and Hermite sibling anchors, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NewtonInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7636.1, Definition 7636.2, and Definition 7636.3 seed the finite Newton interpolation carrier, local NameCert obligation surface, and divided-difference stability route over displayed FieldUp, PolynomialUp, MatrixUp, LagrangeInterpolationUp, HermiteInterpolationUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NewtonInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, quotient polynomial equality, host interpolation functions, hidden matrix inverses, infinite node families, analytic limiting tables, selected Real representatives, or field-completion arguments.

Upgrade path: Obligation closure requires checked carrier admission, divided-difference table formation, Newton basis assembly, node evaluation stability, sibling agreement, transport, replay, provenance, and local naming for BEDC.Derived.NewtonInterpolationUp.NewtonInterpolationUp.

Chapter 7637

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$ packet records the regular completion monad as finite name data over the L10 Real and Completion stack. It reads the dyadic tolerance surface of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet uses the displayed $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ law row only as a finite unit-and-multiplication ledger; it does not introduce quotient completion, countable choice, trichotomy, selected host limits, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7637.1 (Bishop regular completion monad carrier)**.** *A* $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, L, U, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic scales, triangle-budget entries, and composition slack used by every regular completion request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family. It records source windows, unit windows, multiplication windows, and seal-facing observation cuts.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the displayed stream windows and dyadic tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R and the monad-law rows have been displayed.*
5. *L is the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law ledger naming the finite unit, multiplication, left-unit, right-unit, and associativity comparisons.*
6. *U is the unit row. It admits a displayed regular rational name by copying the relevant stream, dyadic, readback, transport, replay, provenance, and local naming rows into the completion surface.*
7. *J is the join or multiplication row. It flattens a displayed completion-of-completion request by replaying the outer and inner windows through the same dyadic ledger before exposing the Real seal.*

8. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readbacks, Real seal, law ledger, unit row, multiplication row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the route*

$$D, S \longmapsto R \longmapsto L, U, J \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, L, U, J, H, C, P, N$. It has no coordinate for quotient completion equality, quotient stream equality, selected representatives, countable-choice schedules, trichotomy, host real equality, or full completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7637.2 (Bishop regular completion monad NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$ *carrier*

$$B = (D, S, R, E, L, U, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionMonad}^{\uparrow}}$ *surface consists of exactly five finite obligation rows:*

1. *carrier admission: the accepted object is the displayed* BHist *packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$*;*
2. *source routing: every request reads the finite stream-window row S, the dyadic tolerance ledger D, and the regular rational readback row R before a Real seal is available;*
3. *unit ledger: the row U copies displayed regular rational names into the completion surface without changing D, S, R, H, C, P, N;*
4. *multiplication ledger: the row J flattens nested completion requests only by replaying the law ledger L and the displayed D, S, R rows;*
5. *structural closure: H, C, P, N provide componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming for the same rows.*

No obligation is stored in quotient completion, quotient stream equality, countable choice, trichotomy, selected host limits, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7637.1. The projection exposes exactly the dyadic ledger, stream windows, regular readbacks, Real seal, monad-law ledger, unit row, multiplication row, transport, replay, provenance, and local naming.

The source-routing row is finite because S lists the inspected windows and D lists the dyadic budgets before R repacks the observations. The seal row E is downstream of those rows and has no independent host-limit coordinate.

The unit and multiplication obligations are read from U, J together with L. The unit row copies the displayed source data, while the multiplication row replays the nested request through the same finite ledger.

Finally H, C, P, N preserve, replay, source, and name the displayed packet. Since the excluded principles have no carrier coordinate, the five listed rows exhaust the local NameCert obligations. □

Theorem 7637.3 (Bishop regular completion monad unit and multiplication ledger)**.** *Every unit, multiplication, left-unit, right-unit, and associativity comparison exported by an accepted* $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$ *carrier is a comparison of the displayed ledger*

$$D, S, R, E, L, U, J, H, C, P, N.$$

The unit row U *embeds source regular rational data through the same* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, and the multiplication row* J *flattens nested completion requests only after the law ledger* L *has replayed the relevant finite windows and dyadic tolerances. The comparison exports no quotient completion, quotient stream equality, countable choice, trichotomy, selected host limit, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the unit side first. The row U is available only after the displayed dyadic ledger D, stream-window row S, and regular readback row R have been projected, so source data enter the completion surface by copying those rows.

For multiplication, read the law ledger L and then the row J. A nested completion request supplies outer and inner finite windows; J flattens them by replaying the same dyadic budget and regular readback route rather than by selecting a quotient representative.

The left-unit and right-unit comparisons are therefore comparisons of copied finite rows. Associativity compares the two finite replays of a three-stage request, and both replays enumerate the same dyadic tolerances, stream windows, readbacks, law ledger rows, and Real-seal handoff.

Transport, replay, provenance, and local naming close the ledger through H, C, P, N. Since the carrier has no row for the excluded principles, none of the exported monad comparisons can depend on them. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletionMonad}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readbacks, a RealUp seal, a CauchyCompletionMonadLaw ledger, unit and multiplication rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRegularCompletionMonad}^{\uparrow}}$, seed)

Scope closed: Definition 7637.1, Definition 7637.2, and Definition 7637.3 seed the finite Bishop regular completion monad surface over the L10 Real and Completion stack.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRegularCompletionMonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, quotient stream equality, countable choice, trichotomy, selected host limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source routing, unit exactness, multiplication exactness, law-ledger replay, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletionMo

Chapter 7638

A Concrete Naming Certificate for $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$

Following Blaschke and Hausdorff, the $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$ packet records the finite-net interface by which compact subsets of a compact metric source are compared inside the compact hyperspace without passing through host powersets, quotient hyperspaces, selected representatives, or choice. It sits after the Hausdorff-distance interface of chapter 2086, the compact metric source of chapter 129, the compact-subset ledger of chapter 1660, the compact metric hyperspace route of chapter 5184, and the Blaschke selection surface of chapter 2370.

Definition 7638.1 (Compact hyperspace finite-net carrier)**.** *A* $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, H, D, E, U, S, T, C, P, N)$$

with the following displayed rows:

1. X *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the metric, finite-net, total-boundedness, and completion data of the ambient compact metric object.*
2. H *is the* $\mathsf{Hyperspace}^{\uparrow}$ *compact-subset ledger over* X*.*
3. D *is the* $\mathsf{HausdorffMetric}^{\uparrow}$ *bidirectional distance ledger for displayed compact-subset names.*
4. E *is the tolerance-indexed finite-net family in the hyperspace: each entry records a positive tolerance, finitely many compact-subset names, and Hausdorff coverage witnesses for the inspected compact-subset rows.*
5. U *is the finite-union row, recording the finite union of selected compact-subset names and its displayed compact-subset readback through* H*.*
6. S *is the stability ledger showing that Hausdorff finite-net matching is preserved when finitely many accepted compact-subset rows are united.*
7. T *records componentwise* hsame *transport for* X, H, D, E, U, S, P, N*.*
8. C *records displayed* Cont *replay through*

$$X, H, D, E \longmapsto U, S.$$

9. P *is the* Pkg *provenance over* $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{CompactMetricHyperspace}^{\uparrow}$, $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $X, H, D, E, U, S, T, C, P, N$. It has no coordinate for host subsets, powerset quotients, open-cover compactness, selected dense families, selected Hausdorff minimizers, or ambient hyperspace completion.

Theorem 7638.2 (Compact hyperspace finite-net NameCert obligations)**.** *For every accepted* $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$ *carrier*

$$F = (X, H, D, E, U, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *source admission, requiring X to expose the compact metric finite-net and completion rows before any hyperspace net is public;*
2. *compact-subset admission, requiring every named compact subset in H and every finite-net center in E to be read through the same* $\mathsf{Hyperspace}^{\uparrow}$ *ledger;*
3. *Hausdorff finite-net matching, requiring D and E to display bidirectional finite coverage at each requested tolerance;*
4. *finite-union stability, requiring U, S to keep finite unions inside the same compact-subset ledger and to preserve the displayed Hausdorff finite-net matching;*
5. *structural exhaustion, requiring T, C, P, N to transport, replay, source, and name only the displayed finite route.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier. The source row X is the only entry point for the compact metric, finite-net, total-boundedness, and completion data, and the compact-subset ledger H is the only entry point for hyperspace names.

Read the Hausdorff and finite-net rows next. The row D supplies bidirectional distance witnesses, while E lists finitely many compact-subset centers and coverage witnesses at the requested tolerances. Both rows are tied to X and H, so the matching cannot range over a host powerset.

The finite-union row U and stability ledger S are then read as the only union-facing coordinates. They certify that finite unions of accepted centers remain named compact-subset rows and that the Hausdorff finite-net matching is preserved under that finite operation.

Finally T, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no additional coordinate for host subsets, quotient hyperspaces, selected dense families, selected minimizers, or ambient hyperspace completion, the five rows exhaust the local NameCert surface. □

Theorem 7638.3 (Compact hyperspace finite-net finite-union stability)**.** *Let $F = (X, H, D, E, U, S, T, C, P, N)$ be an accepted* $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$ *carrier. If a finite family of compact-subset rows is matched by the Hausdorff finite-net row E at a displayed tolerance, then the finite union named by U is matched at the same hyperspace read through the stability ledger S. The stability uses only the* $\mathsf{HausdorffMetric}^{\uparrow}$ *ledger of chapter 2086, the compact metric source of chapter 129, the hyperspace ledger of chapter 1660, the compact metric hyperspace route of chapter 5184, and the Blaschke finite-selection surface of chapter 2370.*

Proof. Start with the finite family named in E. Each member is a displayed compact-subset row in H, and each matching witness is a bidirectional Hausdorff read through D over the compact metric source X.

Apply the finite-union row U. It records the union as another compact-subset row over the same source, rather than as a host subset or quotient object.

Now use the stability ledger S. For each requested tolerance, the matching witnesses for the finitely many members are replayed through the same Hausdorff ledger, and the union read is accepted because every constituent row is already displayed in H.

The structural rows keep the operation inside the packet. Componentwise hsame transport and Cont replay preserve X, H, D, E, U, S, while P, N source and name the route. Thus finite-union stability is a finite NameCert consequence, not an open-cover compactness theorem or a choice-selected hyperspace argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}$ packet is generated from a CompactMetricUp source, a HyperspaceUp compact-subset ledger, HausdorffMetricUp distance rows, hyperspace finite-net centers, finite-union rows, stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7638.1, Definition 7638.2, and Definition 7638.3 seed the finite-net and finite-union stability surface for compact hyperspace packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactHyperspaceFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets, quotient hyperspaces, selected dense families, selected Hausdorff minimizers, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-subset admission, Hausdorff finite-net matching, finite-union stability, transport, replay, provenance, and local naming for BEDC.Derived.CompactHyperspaceFiniteNetUp.

Chapter 7639

A Concrete Naming Certificate for $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$

Following Whitney's stratification conditions and the finite control viewpoint of Goresky–MacPherson stratified Morse theory, the $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ packet records a finite-atlas stratification surface as BEDC NameCert data. It sits over the manifold carrier of chapter 96, the finite jet compatibility rows of chapter 3679, the finite embedding-candidate rows of chapter 6532, and the filtration consumer boundary of chapter 239. The packet names the finite strata, frontier incidences, chart compatibility checks, transport, replay, provenance, and local naming visible to stratified-space consumers; it is not a host Whitney theorem, selected tubular-neighborhood system, quotient realization, global triangulation, or ambient transversality principle.

Definition 7639.1 (Whitney stratification finite-atlas carrier)**.** *A* $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, A, S, F, Q, J, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *source row, carrying the ambient finite-atlas manifold data read by local stratified charts;*
2. *A is the finite atlas row, listing the chart domains and coordinate windows used to inspect each stratum;*
3. *S is the finite stratum-list row, naming the displayed strata, their depth tags, and their admitted chart windows;*
4. *F is the frontier-incidence row, recording which stratum closures may meet lower-depth strata;*
5. *Q is the chart-compatibility row, checking that overlapping chart windows preserve the finite stratum labels and frontier incidences;*
6. *J is the* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *compatibility row, retaining finite jet comparisons as boundary data rather than as a selected smooth extension;*
7. *E is the* $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$ *and* $\mathsf{PersistentHom}^{\uparrow}$ *consumer row, exposing embedding-candidate and filtration-facing reads only after M, A, S, F, Q, J are displayed;*
8. *H is the componentwise* hsame *transport row for the manifold, atlas, stratum, frontier, chart, jet, and consumer rows;*

9. *C records displayed* Cont *replay from a stratification request through M, A, S, F, Q, J, E, H;*
10. *P is the* Pkg *provenance over* $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$, $\mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow}$, $\mathsf{PersistentHom}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, S, F, Q, J, E, H, C, P, N$. It has no coordinate for a host Whitney condition theorem, selected tubular neighborhoods, quotient strata, global triangulation choice, ambient transversality, or persistent-homology completeness.

Theorem 7639.2 (Whitney stratification finite-atlas NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ *carrier $W = (M, A, S, F, Q, J, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WhitneyStratificationFi}}$ *surface consists exactly of carrier admission for the ambient manifold row, finite atlas, finite stratum list, frontier incidence, chart compatibility, finite jet boundary, embedding and filtration consumer row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every stratified-space consumer must read those rows before it may treat the packet as a finite Whitney stratification certificate.*

Proof. Project the carrier of Definition 7639.1. The manifold row M, atlas row A, stratum row S, and frontier row F are displayed coordinates, so the packet exposes the ambient chart source, finite chart inventory, finite stratum list, and allowed closure incidences before any downstream read.

Read the compatibility and consumer rows next. The chart row Q keeps the stratum labels stable across overlaps, the finite jet row J imports only the bounded compatibility surface of chapter 3679, and the consumer row E routes embedding-candidate and filtration-facing reads through chapter 6532 and chapter 239.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier contains no coordinates for host Whitney-condition theorems, selected tubular neighborhoods, quotient strata, global triangulation, ambient transversality, or persistent-homology completeness, the listed rows exhaust the NameCert obligation surface. □

Theorem 7639.3 (Whitney stratification frontier condition route)**.** *Every frontier-condition read exported by an accepted* $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ *packet factors through*

$$\mathsf{Manifold}^{\uparrow} \longmapsto A, S, F, Q \longmapsto \mathsf{WhitneyExtensionFiniteJet}^{\uparrow}, \mathsf{WhitneyEmbeddingFiniteAtlas}^{\uparrow} \longmapsto \mathsf{PersistentHom}^{\uparrow}.$$

The route supplies a finite atlas, finite stratum list, displayed frontier incidence, chart compatibility, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data. It exports no quotient completion of strata, selected limit, choice schedule, host equality of stratified spaces, selected tubular neighborhood, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or global Whitney stratification theorem.*

Proof. Start from the carrier projection. The frontier row F is not available by itself: it is read after the ambient manifold row M, finite atlas row A, and finite stratum-list row S have fixed the displayed chart windows and stratum names.

Use the chart-compatibility row Q to transport the frontier incidence across overlapping atlas windows. The finite jet row J then supplies only the bounded Whitney-extension compatibility surface, while the consumer row E keeps embedding-candidate and filtration reads downstream of the displayed stratification rows.

The structural rows H, C, P, N do not add a second frontier source. They preserve the same finite route by componentwise transport, replay it for consumers, record provenance, and name

the packet locally. The absent quotient, selected-limit, choice, host-equality, tubular-neighborhood, ambient-completeness, and global Whitney-theorem endpoints therefore cannot be read from the packet. □

Falsifiable prediction: Every accepted $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ read exposes the ManifoldUp source, finite atlas, finite stratum list, frontier-incidence row, chart-compatibility row, WhitneyExtensionFiniteJetUp boundary, WhitneyEmbeddingFiniteAtlasUp consumer route, PersistentHomUp filtration-facing route, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a frontier-condition consumer receives the packet.

Independence witness: manifold, whitney-extension-finite-jet, whitney-embedding-finite-atlas, persistenthom: no listed sibling supplies the joint finite-atlas stratification packet $W = (M, A, S, F, Q, J, E, H, C, P,$ because each sibling contributes only an ambient chart carrier, finite jet boundary, embedding-candidate route, or filtration-facing consumer surface while this chapter binds the finite stratum and frontier rows into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}}$.

Constructive story: A $\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}$ packet is generated from a ManifoldUp finite-atlas source, finite stratum list, frontier-incidence row, chart-compatibility row, WhitneyExtensionFiniteJetUp compatibility boundary, WhitneyEmbeddingFiniteAtlasUp and PersistentHomUp consumer route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7639.1, Definition 7639.2, and Definition 7639.3 seed the finite Whitney stratification packet over displayed finite-atlas, stratum, frontier, chart-compatibility, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhitneyStratificationFiniteAtlas}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Whitney-condition theorems, selected tubular neighborhoods, quotient strata, global triangulation choice, ambient transversality, persistent-homology completeness, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, finite-atlas exposure, stratum inventory, frontier-incidence exactness, chart compatibility, finite jet boundary routing, embedding and filtration consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.WhitneyStratificationFiniteAtlasUp.

Chapter 7640

A Concrete Naming Certificate for $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$

Following Haar's invariant integration principle in the finite-group case, the $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ packet records normalized averaging over a displayed finite group action as finite NameCert data. It consumes the finite action rows of chapter 41, the probability-mass interface of chapter 184, and the finite measure ledger of chapter 92. The packet names only the finite invariant average; it does not import sigma-additive Haar measure, a choice of representatives, quotient orbit equality, host sums, or ambient locally compact groups.

Definition 7640.1 (HaarIntegralFiniteGroup averaging carrier)**.** *A* $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, X, F, Q, L, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *action row, including the finite multiplication, unit, inverse, and action table entries needed to translate samples.*
2. *X is the finite support enumeration row listing exactly the points whose values are averaged.*
3. *F is the value row assigning a named scalar or probability-readable value to each enumerated support point.*
4. *Q is the normalized averaging row, recording the finite numerator ledger and denominator schedule read through the* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{Measure}^{\uparrow}$ *interfaces.*
5. *L and R are the left- and right-translation rows, showing how the finite action table permutes the support enumeration before the same average row is read again.*
6. *H records componentwise* hsame *transport for the group action, support enumeration, value, normalized average, translation, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the group action and support enumeration through finite averaging and both translation checks.*
8. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HaarIntegralFiniteGroup}^{\uparrow}}$ *row.*

The carrier has no coordinate for sigma-additive Haar measure, selected orbit representatives, quotient orbit equality, host summation operators, or ambient locally compact groups.

Theorem 7640.2 (HaarIntegralFiniteGroup finite invariance)**.** *For every accepted* $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ *carrier* $A = (G, X, F, Q, L, R, H, C, P, N)$*, the finite average row* Q *is invariant under the displayed left- and right-translation rows* L *and* R*. The comparison is made only by replaying the finite support enumeration* X*, the action table in* G*, and the componentwise* hsame *transport row* H*.*

Proof. Project the carrier of Definition 7640.1. The support row X is finite and displayed, so a translation row can only reorder named support entries already present in the packet.

Read the normalized average row Q. Its numerator ledger and denominator schedule are tied to the displayed support enumeration and to the probability and measure rows consumed by the packet.

Replay the continuation row C first through the left-translation row and then through the right-translation row. In both cases the action table from G permutes the same finite entries, and the transport row H carries the value and average rows componentwise. Hence the translated readback is the same finite average row, without invoking any ambient Haar measure or host summation theorem. □

Theorem 7640.3 (HaarIntegralFiniteGroup NameCert obligations)**.** *For every accepted* $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ *carrier, the local* $\mathsf{NameCert}_{\mathsf{HaarIntegralFiniteGroup}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the displayed group action, support enumeration, value, normalized average, translation, transport, continuation, provenance, and local naming rows;*
2. *finite support exposure, requiring the support enumeration* X *to be public before any average or translation row is consumed;*
3. *averaging normalization, requiring* Q *to display the numerator ledger and denominator schedule read through the finite probability and measure interfaces;*
4. *translation invariance, requiring* L *and* R *to replay the same finite support through the action table of* G*;*
5. *ledger non-escape, excluding sigma-additive Haar measure, choice of representatives, quotient orbit equality, host sums, and ambient locally compact groups.*

Proof. Carrier admission is the tuple of Definition 7640.1; no row outside $G, X, F, Q, L, R, H, C, P, N$ is admitted.

Finite support exposure and averaging normalization are the two rows that make the average readable. The support enumeration fixes the entries, while the normalized average row records the finite ledger and denominator schedule.

Translation invariance is Definition 7640.2. The final ledger row is exact because the carrier contains no sigma-additive, representative-choice, quotient-orbit, host-sum, or locally compact group coordinate. □

Theorem 7640.4 (HaarIntegralFiniteGroup non-escape)**.** *An accepted* $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ *packet exports no sigma-additive Haar measure, no choice of orbit representatives, no quotient orbit equality, no host summation operator, and no ambient locally compact group. Every invariant-average consumer must read only the displayed rows* $G, X, F, Q, L, R, H, C, P, N$*.*

Proof. The carrier of Definition 7640.1 lists the complete public packet. The only average row is the finite normalized row Q, and the only invariance rows are the displayed left and right translation rows.

The obligation theorem Definition 7640.3 requires every consumer to pass through support exposure, normalization, and translation replay. Since no further coordinate is present, the excluded

measure-theoretic, quotient, representative, host-sum, and locally compact group data cannot be projected from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaarIntegralFiniteGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaarIntegralFiniteGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7640.1, Definition 7640.2, and Definition 7640.3 seed the finite invariant-averaging packet over displayed group, probability, and measure rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaarIntegralFiniteGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-additive Haar measure, representative choice, quotient orbit equality, host summation, ambient locally compact groups, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite support exposure, averaging normalization, left and right translation invariance, transport, replay, provenance, local NameCert coverage, and ledger non-escape for the same packet.

Constructive story: A $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$ packet is generated from a finite GroupUp action row, a finite support enumeration row, a value row, a normalized averaging row, left and right translation rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7641

A Concrete Naming Certificate for $\mathsf{CauchyRealNegation}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRealNegation}^{\uparrow}$ packet records additive inverse as a finite Real-seal route rather than as completed-real algebra. It sits above the regular sign-flip surface of chapter 459 and reads the same StreamName, DyadicRatCore, RegSeqRat, and Real rows before exporting a unary $\mathsf{Real}^{\uparrow}$ boundary.

Definition 7641.1 (CauchyRealNegation carrier)**.** *A* $\mathsf{CauchyRealNegation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *row naming the finite sign-flip route of chapter 459;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window surface read by the source regular Cauchy name and by the negated packet;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and signed-observation ledger consumed by the sign-flip route;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the sign-flipped dyadic observations have been routed through the regular rational classifier;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from Q and from no independent completed-real inverse operation;*
6. *H stores componentwise* hsame *transport for R, S, D, Q, E;*
7. *C stores displayed* Cont *replay for the source windows, dyadic sign flip, regular readback, and Real-seal handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealNegation}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealNegation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, trichotomy, ambient completeness, or a completed-real inverse law.

Theorem 7641.2 (CauchyRealNegation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyRealNegation}^{\uparrow}}$ *surface for Definition 7641.1 consists of carrier admission for $G = (R, S, D, Q, E, H, C, P, N)$,*

source binding to the RegularCauchyNegation$^{\uparrow}$ *sign-flip packet, finite-window binding through* StreamName$^{\uparrow}$, *dyadic tolerance binding through* DyadicRatCore$^{\uparrow}$, *regular rational readback through* RegSeqRat$^{\uparrow}$, *Real-seal delivery through* E, *componentwise* hsame *transport through* H, *displayed* Cont *replay through* C, Pkg *provenance through* P, *and local naming through* N.

Proof. Project the displayed carrier. Its first five rows expose the regular sign-flip source, stream-window surface, dyadic observation ledger, regular rational readback, and terminal Real seal.

Bind the source route to chapter 459. That sibling already names the regular sign-flip surface; this packet only records the Real-seal consumer route that reads it through S, D, Q, E.

The transport, replay, provenance, and local naming rows preserve exactly that finite route. Since no remaining carrier coordinate exists, the excluded quotient, host-equality, selected-limit, trichotomy, completeness, and completed inverse objects cannot be read from the certificate. □

Theorem 7641.3 (CauchyRealNegation Real seal route)**.** *Every additive-inverse consumer of an accepted* CauchyRealNegation$^{\uparrow}$ *carrier must read the route*

$$R, S, D, Q, E, H, C, P, N$$

before it can consume the Real$^{\uparrow}$ *seal. The row* R *supplies the regular sign-flip certificate,* S *supplies the finite windows,* D *supplies the dyadic tolerance and sign data,* Q *supplies the* RegSeqRat$^{\uparrow}$ *readback, and* E *is the only Real-seal boundary exported to the consumer.*

Proof. Start with the carrier order in Definition 7641.1. The Real seal E appears only after the regular sign-flip route, finite window surface, dyadic ledger, and regular rational readback rows have been named.

Apply the obligation surface of Definition 7641.2. It binds every consumer route to R, S, D, Q before E is visible, and it keeps H, C, P, N as transport, replay, provenance, and local naming rows for the same data.

Thus the additive-inverse route is a seal-level BEDC handoff over Real$^{\uparrow}$. It does not assert completed real arithmetic, quotient equality of Cauchy names, host sequence equality, selected limits, trichotomy, or ambient completeness. □

Closure status for NameCert$_{\text{CauchyRealNegation}^{\uparrow}}$.

Constructive story: A CauchyRealNegation$^{\uparrow}$ packet is generated from a regular sign-flip row, StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a Real seal, displayed transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRealNegation}^{\uparrow}}$, seed)

Scope closed: Definition 7641.1, Definition 7641.2, and Definition 7641.3 expose the seed carrier, obligation surface, and Real-seal additive-inverse route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRealNegation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close completed real additive-inverse laws, quotient Cauchy equality, host real equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and exact source-row binding for the regular sign-flip, StreamName, DyadicRatCore, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows.

Chapter 7642

A Concrete Naming Certificate for $\mathsf{ToeplitzSilverman}^{\uparrow}$

Following Toeplitz and Silverman on regular matrix summability, the $\mathsf{ToeplitzSilverman}^{\uparrow}$ packet names the finite matrix-window bridge that sits between the Toeplitz transform route of chapter 1682, the regular matrix certificate of chapter 2126, the Hardy–Cesaro averaging row of chapter 1765, the convergence-window row of chapter 1827, the Kronecker tail-control consumer of chapter 1879, the regular rational readback of chapter 301, and the terminal seal of chapter 38. It is a finite NameCert surface for the shared regular-matrix handoff; it is not a host summability theorem, a completed infinite matrix, quotient equality of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7642.1 (Toeplitz–Silverman carrier)**.** *A* $\mathsf{ToeplitzSilverman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, B, V, W, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite matrix-window row, listing the inspected coefficient rows, finite column windows, truncation thresholds, and row-index requests.*
2. *B is the row-bound ledger, recording the dyadic row-sum bounds used before any transformed read is exposed.*
3. *V is the vanishing-column ledger, recording the finite small-entry requests that make the inspected matrix compatible with the convergence-window route.*
4. *W is the source-window row, listing only the finite input observations read from the regular name.*
5. *F is the transformed finite-row surface obtained from M, B, V, W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source windows and transformed rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, B, V, W, F, R have been displayed.*
8. *H stores componentwise* hsame *transport for the matrix, bound, vanishing, source, transformed, readback, seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from matrix inspection to source-window selection, transformed-row formation, regular readback, and real sealing.*

10. *P is the* Pkg *provenance over* ToeplitzSilverman$^{\uparrow}$, SilvermanToeplitz$^{\uparrow}$, ToeplitzLemma$^{\uparrow}$, HardyCesaroMean$^{\uparrow}$, CesaroConvergence$^{\uparrow}$, KroneckerLemma$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ToeplitzSilverman}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, V, W, F, R, E, H, C, P, N$. It has no coordinate for a host summability theorem, completed infinite matrix, selected external limit, quotient equality of real-name streams, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ToeplitzSilvermanUp`

Theorem 7642.2 (Toeplitz–Silverman NameCert obligations)**.** *For every accepted* ToeplitzSilverman$^{\uparrow}$ *carrier $T = (M, B, V, W, F, R, E, H, C, P, N)$, the local* NameCert$_{\text{ToeplitzSilverman}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the matrix-window, row-bound, vanishing-column, source-window, transformed-row, readback, seal, transport, continuation, provenance, and naming rows;*
2. *matrix-window exactness, requiring every transformed row to read coefficient data from M before any regular handoff is available;*
3. *boundedness and vanishing exactness, requiring the row-sum ledger B and the column-vanishing ledger V to be displayed before the transformed finite row F is exposed;*
4. *source-to-transform handoff, requiring W and F to be routed through the same finite matrix window;*
5. *regular readback and real sealing, requiring every completion-facing read to pass through R and then E;*
6. *sibling-route provenance, requiring P to cite the finite summability anchors* ToeplitzLemma$^{\uparrow}$, HardyCesaroMean$^{\uparrow}$, CesaroConvergence$^{\uparrow}$, KroneckerLemma$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *without adding a host analytic principle.*

Proof. Project the carrier of Definition 7642.1. Carrier admission is exactly the displayed inventory $M, B, V, W, F, R, E, H, C, P, N$, so the local certificate has no detached coordinate for a completed matrix or a host summability method.

The matrix-window clause is read from M. The boundedness and vanishing clauses are read from B and V, and the source data are the finite observations in W. Hence the transformed row F is downstream of the same finite window data that supply the regular matrix conditions.

The completion-facing part of the route is forced through R and then E. The rows H, C, P, N transport, replay, source, and name this route, while P records the sibling anchors rather than another analytic path. □

Theorem 7642.3 (Toeplitz–Silverman regular handoff)**.** *Every accepted* ToeplitzSilverman$^{\uparrow}$ *read that reaches a* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer factors through the finite route*

$$M \longmapsto (B, V) \longmapsto W \longmapsto F \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

The route refines the ToeplitzLemma$^{\uparrow}$ *matrix-window discipline, records the* HardyCesaroMean$^{\uparrow}$ *and* CesaroConvergence$^{\uparrow}$ *special surfaces, leaves* KroneckerLemma$^{\uparrow}$ *with a named finite-matrix bridge, and exposes no readback except through* RegSeqRat$^{\uparrow}$ *before the* Real$^{\uparrow}$ *seal.*

Proof. Start with a regular or real-facing consumer. By Definition 7642.1, the only coefficient data visible to the classifier are the finite rows in M, with regularity checks held in B and V.

The source sequence is then inspected through W, and the transformed row is displayed as F. Thus the consumer cannot reach regular rational readback before the matrix window, bounded-row ledger, vanishing-column ledger, source window, and transformed row have all been exposed.

The handoff to regular names is R, and the terminal real-facing seal is E. The structural rows H, C, P, N preserve and name the same path, while the sibling anchors in P identify the Toeplitz, Hardy–Cesaro, Cesaro-convergence, and Kronecker routes as provenance rather than bypasses. □

Falsifiable prediction: Every accepted $\mathsf{ToeplitzSilverman}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ while omitting the finite matrix window, row-bound ledger, vanishing-column ledger, source window, transformed finite row, readback row, real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: silvermantoeplitz, toeplitzlemma, hardycesaromean, cesaroconvergence, kroneckerlemma, regseqrat, real: $\mathsf{ToeplitzSilverman}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite matrix-window bridge, bounded-row and vanishing-column ledgers, source windows, transformed rows, regular readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ToeplitzSilverman}^{\uparrow}}$.

Constructive story: A $\mathsf{ToeplitzSilverman}^{\uparrow}$ packet is generated from a finite matrix window, row-bound ledger, vanishing-column ledger, source windows, transformed finite rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ToeplitzSilverman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7642.1, Definition 7642.2, and Definition 7642.3 seed the finite matrix-window bridge from regular summability data to Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ToeplitzSilverman}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host summability theorems, completed infinite matrices, selected limits, quotient equality of real-name streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, matrix-window exactness, boundedness and vanishing exactness, source-to-transform handoff, regular readback, Real-seal nonescape, sibling-route provenance, and TasteGate field faithfulness for BEDC.Derived.ToeplitzSilvermanUp.

Chapter 7643

A Concrete Naming Certificate for $\mathsf{KrullIntersection}^{\uparrow}$

Following Atiyah–Macdonald style Noetherian commutative algebra, the $\mathsf{KrullIntersection}^{\uparrow}$ packet records the finite BEDC naming surface for the ideal-adic tail condition that a displayed module element disappears from all sufficiently exposed tail-power rows. It sits after the ring, ideal, module, and Noetherian module carriers of chapter 43, chapter 81, chapter 46, and chapter 3648; it is a sibling of the Artin–Rees filtration route in chapter 6899 and a source for completion-facing algebra packets. The packet names only finite I-adic tail-power rows, transport, replay, provenance, and local naming; it does not import quotient completion equality, a selected infinite intersection, separatedness of a completion, or an ambient host theorem.

Definition 7643.1 (Krull intersection carrier)**.** *A* $\mathsf{KrullIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{KrullIntersection}(R, I, M, N, T, G, H, C, P, Z)$$

with the following displayed rows:

1. *R is the* $\mathsf{Ring}^{\uparrow}$ *or* $\mathsf{CommRing}^{\uparrow}$ *source row read by the algebra consumer.*
2. *I is the* $\mathsf{Ideal}^{\uparrow}$ *row over the same ring classifier, carrying the ideal-power requests.*
3. *M is the* $\mathsf{Module}^{\uparrow}$ *row, together with the* $\mathsf{NoetherianModule}^{\uparrow}$ *surface needed by the finite tail-power route.*
4. *N is the Noetherian control row, exposing only the finite generator and stabilization ledgers consumed by ideal-adic powers.*
5. *T is the I-adic tail-power ledger, recording requested powers $I^n M$, later windows $I^{n+k} M$, and the finite comparison rows between them.*
6. *G is the Krull-intersection classifier row, comparing accepted packets by R, I, M, N, T and by the declared tail membership route.*
7. *H records componentwise* hsame *transport for the ring, ideal, module, Noetherian, tail-power, and classifier rows.*
8. *C records displayed* Cont *replay from an algebra or completion-facing request through the same finite tail-power rows.*
9. *P is the* Pkg *provenance over* $\mathsf{KrullIntersection}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{Ideal}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{NoetherianModule}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. Z is the local $\mathsf{NameCert}_{\mathsf{KrullIntersection}^{\uparrow}}$ *row binding the packet.*

The carrier has no coordinate for quotient completion equality, a selected element of an infinite intersection, host separatedness, countable choice, or an external Krull intersection theorem.

Theorem 7643.2 (Krull intersection NameCert obligations)**.** *For every accepted* $\mathsf{KrullIntersection}^{\uparrow}$ *carrier* $\mathsf{KrullIntersection}(R, I, M, N, T, G, H, C, P, Z)$*, the local* $\mathsf{NameCert}_{\mathsf{KrullIntersection}^{\uparrow}}$ *surface consists exactly of ring-source exposure, ideal exposure, module exposure, Noetherian control, finite I-adic tail-power rows, classifier stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion-facing or algebra-facing consumer may inspect only the finite route*

$$R, I, M, N \longrightarrow T, G \longrightarrow H, C, P, Z.$$

It receives no quotient completion equality, no selected infinite intersection element, no host separatedness theorem, no countable-choice selector, and no ambient equality of all ideal-power intersections.

Proof. Carrier admission is projection from Definition 7643.1. The rows R, I, M, N expose the Noetherian ring, ideal, module, and control surfaces before any ideal-power tail can be read.

The row T supplies only finite I-adic tail-power windows. Each inspected tail comparison is therefore a displayed row of the packet, not an appeal to a selected infinite intersection or to quotient completion equality.

The classifier G compares packets through the same source and tail-power rows. Componentwise hsame transport H and displayed Cont replay C preserve those rows, while P records provenance and Z repacks them as the local NameCert.

Since no carrier coordinate names quotient completion equality, a selected infinite intersection element, host separatedness, countable choice, or ambient equality of all ideal-power intersections, the listed rows exhaust the local certificate surface. □

Falsifiable prediction: Any accepted $\mathsf{KrullIntersection}^{\uparrow}$ consumer that cites an ideal-adic tail conclusion without exposing $R, I, M, N, T, G, H, C, P, Z$ refutes the carrier surface of Definition 7643.1.

Independence witness: ring, ideal, module, noetherian-module, artin-rees, completion: no listed sibling is bijective to $\mathsf{KrullIntersection}^{\uparrow}$, because this packet jointly binds finite ideal-power tail rows, Noetherian control, classifier stability, transport, replay, provenance, and local NameCert rows while refusing quotient completion equality and selected infinite intersections.

Closure status for $\mathsf{NameCert}_{\mathsf{KrullIntersection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KrullIntersection}^{\uparrow}$ packet is generated from a ring source, ideal source, module source, Noetherian control row, finite ideal-adic tail-power ledger, classifier row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KrullIntersection}^{\uparrow}}$, seed)

Scope closed: Definition 7643.1 and Definition 7643.2 seed the finite Krull-intersection certificate over RingUp, IdealUp, ModuleUp, NoetherianModuleUp, ArtinReesUp-adjacent filtration rows, completion-facing consumers, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KrullIntersection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected infinite intersections, host separatedness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite ideal-adic tail-power exposure, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.KrullIntersectionUp.

Chapter 7644

A Concrete Naming Certificate for $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ packet records the finite dyadic interval-selection surface used when a Real or completion route needs a displayed local interval but must not import a hidden countable-choice operator. It sits between the dyadic rational core of chapter 291, the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the terminal real-name seals of chapter 38, and the located interval carrier of chapter 394. The packet names only a finite request ledger, a displayed dyadic candidate family, a local selection row, a located-interval handoff, structural transport, replay, provenance, and local naming; it does not choose an infinite subsequence, a global Cauchy selector, a completed real endpoint, or a countable family of intervals.

Definition 7644.1 (Choice-free dyadic interval selection carrier)**.** *A* $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, D, W, R, E, L, H, C, P, N)$$

with the following visible rows:

1. *Q is the finite request row. It records the inspected tolerance, endpoint-side flag, and consumer route that asked for a dyadic interval witness.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *candidate ledger. It supplies only the finitely displayed dyadic endpoints and containment comparisons named by Q.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the inspected real-name or completion-name prefixes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximants exposed by those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected endpoint names, with no selected completed real.*
6. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *handoff row, packaging the displayed dyadic endpoints as a located interval cell when the consumer asks for interval-shaped output.*
7. *H records componentwise* hsame *transport for $Q, D, W, R, E, L, C, P, N$.*
8. *C records displayed* Cont *replay from the interval-selection request through the finite dyadic, window, readback, Real-seal, and located-interval rows.*

9. *P is the* Pkg *provenance over* ChoiceFreeDyadicIntervalSelection$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, LocatedInterval$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}$ *row.*

The classifier accepts a packet only through the displayed tuple $Q, D, W, R, E, L, H, C, P, N$. It has no coordinate for countable choice, dependent choice, a selected infinite branch, a global interval selector, a completed endpoint value, or a quotient of real names.

Theorem 7644.2 (Choice-free dyadic interval selection obligations)**.** *For every accepted* ChoiceFreeDyadicIntervalSe *carrier*

$$S = (Q, D, W, R, E, L, H, C, P, N),$$

the local NameCert$_{\text{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed packet S and its finite request row Q;*
2. *dyadic candidate exposure through D together with window and rational readback exposure through W and R;*
3. *terminal real-name sealing through E and located-interval handoff through L;*
4. *componentwise transport and displayed replay through H and C;*
5. *provenance and local naming through P and N.*

No additional obligation asks for a countable selection function, an infinite chain of interval choices, a quotient-real endpoint, or a host-completion choice principle.

Proof. Project the carrier of Definition 7644.1. Its accepted rows are exactly $Q, D, W, R, E, L, H, C, P, N$, and the classifier compares only these displayed components by hsame.

The computational part of the obligation surface is exhausted by Q, D, W, R, E, L. The request row fixes one finite inspection, the dyadic core supplies the displayed endpoint candidates, the stream and regular rational rows expose the requested prefixes, the Real row seals the inspected endpoint names, and the located interval row repacks the same finite dyadic cell.

The structural part is exhausted by H, C, P, N. These rows transport, replay, source, and name the same packet. Since no row contains a function selecting intervals for all tolerances or all request histories, the local certificate cannot smuggle countable choice into the Real or completion-facing route. □

Theorem 7644.3 (Choice-free dyadic interval selection finite route)**.** *Every accepted* ChoiceFreeDyadicIntervalSelect *read factors through the finite route*

$$Q \longmapsto D \longmapsto W, R, E \longmapsto L \longmapsto H, C, P, N.$$

The route exports only the inspected request, displayed dyadic candidates, finite stream windows, regular rational readbacks, terminal Real seals, located-interval handoff, transport, replay, provenance, and local naming. It exports no countable selector, dependent-choice tree, completed real endpoint, or global interval-selection oracle.

Proof. Begin with an accepted consumer of an interval-selection read. By Definition 7644.1, the consumer enters through the finite request row Q and reaches dyadic data only through D.

Apply Definition 7644.2. The only rows downstream of D are the finite window and readback rows W, R, the terminal seal row E, and the located-interval handoff L before the structural rows are inspected.

The sibling packets $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{LocatedInterval}^{\uparrow}$ provide their own finite NameCert surfaces. None adds a selected infinite interval family or a host choice operator to this packet. Hence every accepted read remains on the displayed finite route. □

Theorem 7644.4 (Choice-free dyadic interval selection nonescape)**.** *An accepted* $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ *consumer cannot obtain a* $\mathsf{LocatedInterval}^{\uparrow}$ *handoff or a* $\mathsf{Real}^{\uparrow}$ *endpoint seal without first displaying* Q, D, W, R, E *as appropriate to the request. In particular, any route that produces an interval cell by choosing a member of an infinite dyadic family is outside the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}$ *surface.*

Proof. Read the accepted carrier through Definition 7644.1. The located-interval output occurs only in L, and L is downstream of the finite request, dyadic candidate, window, readback, and Real-seal rows.

Use Definition 7644.3. The route places Q, D, W, R, E before L and then proceeds only to H, C, P, N, so the carrier has no independent interval-output entry point.

The structural rows can only transport, replay, source, and name the same displayed packet. They do not create a choice function over all tolerances, all histories, or all dyadic refinements. □

Falsifiable prediction: An accepted $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ consumer that reaches a $\mathsf{LocatedInterval}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without the finite request row, dyadic candidate ledger, StreamName windows, and RegSeqRat readback refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedinterval: no listed sibling supplies the whole $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ packet, because the selector certificate binds a finite request row, dyadic candidate ledger, window and readback rows, terminal Real seals, a located-interval handoff, transport, replay, provenance, and local naming without a countable-choice coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}$ packet is generated from one finite request row, displayed DyadicRatCore candidates, StreamName windows, RegSeqRat readbacks, RealUp endpoint seals, a LocatedInterval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7644.1, Definition 7644.2, Definition 7644.3, and Definition 7644.4 seed the finite choice-free dyadic interval-selection carrier and its nonescape route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceFreeDyadicIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, dependent choice, total interval selection over all tolerances, quotient-real endpoint equality, completion existence, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires a Lean-side ChoiceFreeDyadicIntervalSelectionUp carrier with checked request admission, dyadic candidate exposure, finite window readback, Real endpoint sealing, LocatedInterval handoff, transport, replay, provenance, and local naming.

Chapter 7645

A Concrete Naming Certificate for ClosedAtDecisionRefusalBoundary$^{\uparrow}$

The ClosedAtDecisionRefusalBoundary$^{\uparrow}$ packet realizes the Closed-MetaCIC decidability frontier of section 9386.5 as finite NameCert data. It sits beside the decidability witness of chapter 751, the bounded normal-equality checker of chapter 752, the finite beta closedness route of chapter 6876, and the subject-reduction obstruction of chapter 9377. The packet names checker input, bounded normal output, explicit refusal, transport, replay, provenance, and local naming; it does not turn bounded normal equality into a standalone ClosedAt decider or a global closed beta-conversion oracle.

Definition 7645.1 (ClosedAt decision refusal boundary carrier)**.** *A* ClosedAtDecisionRefusalBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, K, N_b, R, T, C, P, G, H, N)$$

with the following visible rows:

1. *Q is the request row, recording whether the consumer asked for standalone* ClosedAt *decision, global closed beta-conversion, or bounded normal comparison.*
2. *K is the finite checker row. It exposes the closed typing, structural equality, and represented-term checks available in the existing MetaCIC decidability surface.*
3. *N_b is the bounded-normal row, carrying only displayed fuel, bounded normalizer outputs, and finished-normal witnesses when those witnesses are explicitly present.*
4. *R is the refusal row. It rejects standalone* ClosedAt *decision and unrestricted closed beta-conversion requests before any bounded checker row is consumed.*
5. *T is the transport row from accepted bounded evidence to downstream normal-form or decidability packets, with the refusal row still visible.*
6. *C records displayed* Cont *replay through the request, refusal, checker, bounded-normal, and transport rows.*
7. *P is the* Pkg *provenance over* ClosedAtDecisionRefusalBoundary$^{\uparrow}$, MetaCICDecidabilityWitness$^{\uparrow}$, BoundedNormalEqualityChecker$^{\uparrow}$, FiniteBetaClosednessIteration$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *G records the later Lean TasteGate surface for* Nontrivial ClosedAtDecisionRefusalBoundary$^{\uparrow}$ *and* FieldFaithful ClosedAtDecisionRefusalBoundary$^{\uparrow}$ *instances.*

9. *H records componentwise* hsame *transport for $Q, K, N_b, R, T, C, P, G, N$.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedAtDecisionRefusalBoundary}^{\uparrow}}$ *row.*

The classifier accepts a packet only through $Q, K, N_b, R, T, C, P, G, H, N$. It has no coordinate for a total closedness decider, fuel-erased normalizer, unrestricted conversion oracle, quotient equality of terms, subject-reduction discharge, or kernel-consistency theorem.

Theorem 7645.2 (ClosedAt decision refusal boundary obligations)**.** *For every accepted* ClosedAtDecisionRefusalBou *carrier*

$$B = (Q, K, N_b, R, T, C, P, G, H, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedAtDecisionRefusalBoundary}^{\uparrow}}$ *surface consists exactly of checker request admission through Q; finite checker exposure through K; bounded-normal exposure through N_b; refusal of standalone* ClosedAt *decision and global closed beta-conversion through R; transport through T; replay through C; provenance through P; later* Nontrivial *and* FieldFaithful *instance obligations through G; componentwise transport through H; and local naming through N.*

Proof. Project the carrier of Definition 7645.1. The visible rows are exactly $Q, K, N_b, R, T, C, P, G, H, N$, and the classifier compares only those rows componentwise.

The boundary row R is part of the accepted packet, not an external warning. Therefore a request for standalone ClosedAt decision or unrestricted closed beta-conversion is classified before K or N_b can be consumed as positive evidence.

Rows K and N_b still expose the bounded checks that the existing MetaCIC decidability surface permits. Rows T, C, P, G, H, N transport, replay, source, record the later Lean instance obligations, and name the same finite packet. None adds a total decider or fuel-erased conversion oracle. □

Theorem 7645.3 (ClosedAt decision refusal first)**.** *Every accepted* $\mathsf{ClosedAtDecisionRefusalBoundary}^{\uparrow}$ *read routes standalone* ClosedAt*-decision and global closed beta-conversion requests to R before any bounded checker row K or bounded-normal row N_b is consumed. Bounded checker evidence may be read only after the request row has not been classified as one of those global requests.*

Proof. Read the request component Q of the accepted carrier in Definition 7645.1. The only row that classifies a standalone closedness decision or unrestricted conversion request is R.

Apply Definition 7645.2. It places the refusal row in the local NameCert surface together with, but logically before use of, the checker and bounded-normal rows for requests that ask for global decision power.

Thus a bounded checker read is available only for bounded requests. A consumer that skips R and reads K or N_b as a total closedness or conversion oracle is outside the packet. □

Theorem 7645.4 (ClosedAt decision refusal nonescape)**.** *An accepted* $\mathsf{ClosedAtDecisionRefusalBoundary}^{\uparrow}$ *packet cannot export a standalone* ClosedAt *decider, global closed beta-conversion oracle, or fuel-erased normalizer. Any downstream use of K or N_b remains bounded by the displayed request, fuel, finished-normal evidence, refusal row, transport, replay, provenance, and local naming.*

Proof. Use Definition 7645.3. Global requests are routed to R, while bounded reads proceed only through the displayed checker and bounded-normal rows.

The transport row T keeps R visible when bounded evidence is forwarded to sibling packets. The replay and provenance rows C and P record the same route, and G, H, N only record later Lean instance obligations, componentwise transport, and local naming.

Since the carrier has no total-decision coordinate and no fuel-erased normalizer coordinate, no accepted read can escape from bounded evidence into a global theorem. □

The later Lean side must provide explicit Nontrivial ClosedAtDecisionRefusalBoundary$^{\uparrow}$ and FieldFaithful ClosedAtDecisionRefusalBoundary$^{\uparrow}$ instances in `lean4/BEDC/Derived/ClosedAtDecisionRefusalBou` These instances must preserve $Q, K, N_b, R, T, C, P, G, H, N$ as distinct rows before any checked marker can cite the packet as a TasteGate witness.

Falsifiable prediction: Every accepted ClosedAtDecisionRefusalBoundary$^{\uparrow}$ read routes standalone ClosedAt-decision and global beta-conversion requests to the refusal row before any bounded checker row is consumed.

Independence witness: metacicdecidableboundary, normalfalseexclusion, closednormalconfluenceseal, metacic-subject-reduction-obstruction: no listed sibling supplies the whole ClosedAtDecisionRefusalBounda packet, because the refusal boundary binds checker request, bounded-normal output, refusal, transport, replay, provenance, later Lean instance obligations, componentwise transport, and local naming as one carrier.

Closure status for NameCert$_{\text{ClosedAtDecisionRefusalBoundary}^{\uparrow}}$.

Constructive story: A ClosedAtDecisionRefusalBoundary$^{\uparrow}$ packet is generated from checker request data, finite MetaCIC checker rows, bounded-normal rows, an explicit refusal row, transport, replay, provenance, later Nontrivial and FieldFaithful instance obligations, componentwise hsame transport, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosedAtDecisionRefusalBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 7645.1, Definition 7645.2, Definition 7645.3, and Definition 7645.4 seed the finite refusal boundary for standalone ClosedAt decision and global closed beta-conversion requests.

Formal verification: FormalStatus(NameCert$_{\text{ClosedAtDecisionRefusalBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close standalone ClosedAt decidability, unrestricted closed beta-conversion decidability, fuel-erased normalization, subject reduction, confluence, kernel consistency, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires a Lean-side ClosedAtDecisionRefusalBoundaryUp carrier with checked request classification, refusal-first routing, bounded checker exposure, bounded-normal exposure, transport, replay, provenance, Nontrivial and FieldFaithful instances, componentwise transport, and local naming.

Chapter 7646

A Concrete Naming Certificate for $\mathsf{RadstromEmbedding}^{\uparrow}$

Following Radstrom's embedding of convex compact sets into an ordered vector group, the $\mathsf{RadstromEmbedding}^{\uparrow}$ packet records the finite BEDC surface on which convex compact-set addition and Hausdorff-distance reads become a cancellative embedding route. It sits beside the compact-subset hyperspace carrier of chapter 1660, the compact metric source of chapter 129, the convex-set carrier of chapter 208, the Hausdorff metric route of chapter 2086, the Vietoris topology route of chapter 2375, and the real seal of chapter 38. The chapter names only finite compact, convex, Hausdorff, Vietoris, and real-valued rows; it is not a host Banach-space completion, quotient equality of compact sets, selected support functional, or ambient compactness theorem.

Definition 7646.1 (Radstrom embedding carrier)**.** *A* $\mathsf{RadstromEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, V, H, T, E, D, S, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the finite compact metric presentation from which compact-set names are read.*
2. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row recording the inspected convex compact-set operations and finite Minkowski-sum windows.*
3. *H is the* $\mathsf{HausdorffMetric}^{\uparrow}$ *row recording the finite Hausdorff comparison ledger on compact subsets.*
4. *T is the* $\mathsf{VietorisTopology}^{\uparrow}$ *row recording the hit–miss compatibility windows used before the embedding row is exposed.*
5. *E is the embedding row, displaying the cancellative difference-class name determined by K, V, H, T.*
6. *D is the distance-readback row into* $\mathsf{Real}^{\uparrow}$*, exposed only after the compact, convex, Hausdorff, Vietoris, and embedding rows have been displayed.*
7. *S stores componentwise* hsame *transport for compact, convex, Hausdorff, Vietoris, embedding, distance, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from compact-set inspection to convex operation windows, Hausdorff comparison, Vietoris compatibility, embedding exposure, and real-valued distance readback.*

9. *P is the* Pkg *provenance over* $\mathsf{RadstromEmbedding}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RadstromEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, H, T, E, D, S, C, P, N$. It has no coordinate for host quotient spaces, selected support functions, external Banach completions, powerset equality, countable choice, or open-cover compactness.

Theorem 7646.2 (Radstrom embedding NameCert obligations)**.** *For every accepted* $\mathsf{RadstromEmbedding}^{\uparrow}$ *carrier $R = (K, V, H, T, E, D, S, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RadstromEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, convex compact-set operations, Hausdorff comparison ledger, Vietoris compatibility row, embedding row, real-valued distance readback, componentwise transport, replay, provenance, and local naming rows. The obligations require the embedding row E to be downstream of K, V, H, T, and they require the* $\mathsf{Real}^{\uparrow}$ *distance readback D to be downstream of E.*

Proof. Project the carrier of Definition 7646.1. Carrier admission is the displayed tuple $K, V, H, T, E, D, S, C, P, N$, so the local certificate has no detached coordinate for an ambient quotient space or external completion.

The compact, convex, Hausdorff, and Vietoris obligations are read from K, V, H, T. Since E is the only embedding row, every embedding exposure is downstream of those four rows.

The real-valued readback is D, and its only source is the displayed embedding row. The rows S, C, P, N transport, replay, source, and name the same route, so the NameCert surface is exhausted by the listed finite rows. □

Theorem 7646.3 (Radstrom embedding hyperspace route)**.** *Every accepted* $\mathsf{RadstromEmbedding}^{\uparrow}$ *consumer that reads the embedding row or the* $\mathsf{Real}^{\uparrow}$ *distance readback factors through the finite hyperspace route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Hyperspace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{HausdorffMetric}^{\uparrow} \longmapsto \mathsf{VietorisTopology}^{\uparrow} \longmapsto E \longmapsto D.$$

The route uses the compact-subset carrier of chapter 1660 as the neighbour through which convex compact-set and Hausdorff-metric requests are inspected. It exposes no quotient equality of compact sets, host linear-space completion, selected support functional, choice principle, or open-cover compactness proof.

Proof. Begin with the compact-subset side. The carrier row K supplies the compact metric presentation, and the provenance row P routes compact-subset inspection through the sibling hyperspace chapter before any convex operation is read.

Next read the convex and metric rows. The rows V, H, T display convex compact-set operations, finite Hausdorff comparisons, and Vietoris compatibility windows. Thus the embedding row E cannot be inspected before the compact hyperspace, convex, Hausdorff, and Vietoris rows are all present.

Finally read the real-facing row D. Since D is downstream of E and the transport and replay rows preserve the same order, any consumer of the Radstrom surface factors through the displayed hyperspace route and cannot recover any excluded host principle. □

Falsifiable prediction: Any accepted $\mathsf{RadstromEmbedding}^{\uparrow}$ carrier that reaches the embedding row or the $\mathsf{Real}^{\uparrow}$ readback while omitting the compact metric row, hyperspace route, convex

compact-set row, Hausdorff metric row, Vietoris compatibility row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: hyperspace, compactmetric, convexset, hausdorffmetric, vietoristopology, real: $\mathsf{RadstromEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries convex compact-set operations, Hausdorff comparison, Vietoris compatibility, a cancellative embedding row, real-valued readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RadstromEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RadstromEmbedding}^{\uparrow}$ packet is generated from CompactMetric, Hyperspace, ConvexSet, HausdorffMetric, VietorisTopology, embedding, Real readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RadstromEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7646.1, Definition 7646.2, and Definition 7646.3 seed the finite convex compact hyperspace route into Radstrom embedding data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RadstromEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of compact sets, host Banach completions, selected support functionals, countable choice, open-cover compactness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convex compact-set operation exposure, Hausdorff comparison exactness, Vietoris compatibility, embedding nonescape, Real readback, provenance, and local NameCert exhaustion for BEDC.Derived.RadstromEmbeddingUp.

Chapter 7647

A Concrete Naming Certificate for $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$ packet records the finite route by which compact located metric data and apartness-sensitive separation data are read before a completion or Real-facing consumer asks for distinctness. It sits beside the located-limit route of chapter 1420 and the separated-metric route of chapter 1411. The packet keeps the constructive distinction between positive apartness evidence and equality refusal: it names finite compact windows, located comparison rows, separated metric tests, transport, replay, provenance, and local NameCert data without importing classical trichotomy, quotient equality, selected limit points, host equality, or an ambient completed metric object.

Definition 7647.1 (Compact apartness located separation carrier)**.** *A* $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, A, S, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the consumed* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{Compact}^{\uparrow}$ *finite-window row whose compact reads are presented by displayed probe or finite-net data.*
2. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *row carrying the located metric-distance comparisons used by the packet.*
3. *A is the positive apartness ledger, recording finite evidence that two displayed compact-window reads are separated by a rational lower-bound row.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row that turns the positive apartness evidence into the separation-facing public test used by downstream consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row for the rational lower bound and the terminal numerical seal.*
6. *H records componentwise* hsame *transport for compact windows, metric rows, apartness ledgers, separated comparisons, readback rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from compact finite windows through located metric comparison, apartness evidence, separated comparison, and rational or Real readback.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}}$ *row.*

The carrier has no coordinate for classical trichotomy of equality and apartness, quotient equality, selected limit representatives, countable choice, host equality, or an ambient completed metric space.

Theorem 7647.2 (CompactApartnessLocatedSeparation NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$ *carrier* $L = (K, M, A, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactApartn}}$ *surface consists exactly of carrier admission, compact finite-window exposure, located metric comparison, positive apartness lower-bound exposure, separated-metric handoff, rational or Real readback, transport, replay, provenance, and local naming. A consumer may read a separated compact comparison only after the compact row K, metric row M, apartness row A, and separated row S have been displayed.*

The same surface exports no classical trichotomy, quotient equality, selected limit point, countable-choice schedule, host equality, ambient completion object, or Real completeness row.

Proof. Project the carrier of Definition 7647.1. The displayed rows $K, M, A, S, R, H, C, P, N$ are the complete public inventory of the packet.

The compact and located rows are read first. The compact finite-window row K supplies the bounded testing window, and the metric row M supplies located distance comparisons internal to that window.

The apartness and separation rows are downstream. The row A records positive rational lower-bound evidence, while S packages the same evidence as the separated-metric handoff consumed by sibling separation routes such as chapter 1411.

Finally R, H, C, P, N read back, transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for trichotomy, quotient equality, selected limits, choice schedules, host equality, ambient completions, or Real completeness, none of those objects can become a NameCert obligation. □

Falsifiable prediction: A compact separation consumer that reads apartness without first displaying the compact finite-window row, located metric comparison row, separated-metric handoff, and rational or Real readback row is not a $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$ packet.

Independence witness: compactmetric, locatedlimit, separatedmetric, regseqrat, real: the packet is not bijective to these siblings because it binds compact finite windows, located metric comparison, positive apartness lower bounds, separated-metric handoff, numerical readback, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$ packet is generated from compact finite-window data, located metric comparisons, positive apartness lower-bound rows, SeparatedMetric handoff data, RegSeqRat or Real readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 7647.1 and Definition 7647.2 seed the compact apartness/separation carrier and its NameCert obligation surface over displayed CompactMetricUp, LocatedLimitUp, SeparatedMetricUp, MetricSpaceUp, RegSeqRatUp, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, classical trichotomy, quotient equality, selected limit representatives, countable choice, host equality, ambient completed metric spaces, or Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-window exposure, located metric comparison, apartness lower-bound exposure, separated-metric handoff, numerical readback, transport, replay, provenance, and local naming for the same packet.

Chapter 7648

A Concrete Naming Certificate for $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$

Following Heine–Cantor and Bishop compact-uniform-continuity, the $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$ packet records a finite oscillation envelope extracted from compact finite-cover data, pointwise continuity moduli, and rational lower-bound folds. It consumes the compact source rows of chapter 60, the metric rows of chapter 57, the continuous-map rows of chapter 130, the uniform-modulus handoff of chapter 369, and the numerical readback routes of chapter 37 and chapter 38. The packet names finite ProbeBundle radii and oscillation bounds without selecting an external modulus function, invoking open-cover compactness, importing host function equality, or asserting an ambient completed metric object.

Definition 7648.1 (Finite oscillation-envelope carrier)**.** *A* $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, X, Y, F, B, R, Q, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *finite cover row for the source object.*
2. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, and Y is the target metric row used to read oscillation bounds.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose local pointwise modulus data are consumed on the compact finite cover.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *of source centers and radius rows read from the compact witness.*
5. *R is the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the positive local radius histories carried by B.*
6. *Q is the oscillation-bound ledger, assigning each displayed probe center a target-distance bound read through F, X, Y, and R.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the finite envelope after the rational fold has been replayed.*
8. *H records componentwise* hsame *transport for compact, metric, continuous-map, ProbeBundle, rational-fold, oscillation-bound, uniform-modulus, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the compact finite-cover row through local modulus reads, rational lower-bound folds, oscillation estimates, and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationEnvelope}^{\uparrow}}$ *row.*

The carrier contains no external modulus selector, no open-cover compactness row, no host function equality, no countable-choice schedule, no quotient metric equality, and no ambient complete metric object.

Lean4 ✓ `BEDC.Derived.FiniteOscillationEnvelopeUp.FiniteOscillationEnvelopeTasteGate single carrier alignment`

Theorem 7648.2 (FiniteOscillationEnvelope NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$ *carrier* $O = (K, X, Y, F, B, R, Q, U, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-cover exposure, source and target metric reads, continuous-map pointwise modulus reads, finite ProbeBundle radius rows,* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold exposure, oscillation-bound ledger rows,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, transport, replay, provenance, and local naming.*

Every uniform-modulus consumer must read the finite route

$$K, X, Y, F, B, R, Q \longmapsto U \longmapsto H, C, P, N$$

before a tolerance-only modulus row is public. The packet exports no external modulus selector, open-cover compactness, host function equality, countable-choice schedule, quotient metric equality, or ambient complete metric object.

Proof. Project the carrier of Definition 7648.1. The rows $K, X, Y, F, B, R, Q, U, H, C, P, N$ are the complete public inventory.

The compact and continuous rows provide the local data. The finite cover row K supplies the displayed ProbeBundle B, while the continuous-map row F supplies pointwise modulus reads over the source and target metric rows X, Y.

The rational fold and oscillation ledger are the finite envelope. The row R folds the positive radius histories through $\mathsf{Rat}^{\uparrow}$, and Q records the resulting oscillation bounds at the displayed probe centers.

Finally the uniform-modulus row U is readable only after the finite envelope has been replayed. The transport, continuation, provenance, and naming rows preserve that dependency order, and the missing coordinates exclude external selectors, open-cover compactness, host function equality, choice schedules, quotient metric equality, and ambient completion objects. □

Falsifiable prediction: A compact-uniform-continuity consumer that reads a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying compact finite-cover data, ContinuousMap pointwise modulus rows, finite ProbeBundle radii, RatUp folds, and oscillation-bound ledgers is not a $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$ packet.

Independence witness: compact, continuousmap, uniformmodulus, metricspace, rat, real: FiniteOscillationEnvelo is not bijective to any listed sibling because it jointly carries finite compact-cover centers, local oscillation bounds, rational lower-bound folds, a uniform-modulus handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationEnvelope}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{FiniteOscillationEnvelope}^{\uparrow}$ packet is generated from compact finite-cover data, source and target MetricSpace rows, a ContinuousMap pointwise-modulus row, finite ProbeBundle radius rows, RatUp lower-bound folds,

oscillation-bound ledger rows, a UniformModulus handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteOscillationEnvelope}\uparrow}$, seed)

Scope closed: Definition 7648.1 and Definition 7648.2 seed the finite oscillation-envelope carrier and NameCert obligation surface over displayed CompactUp, ContinuousMapUp, UniformModulusUp, MetricSpaceUp, RatUp, and RealUp rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteOscillationEnvelope}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, open-cover compactness, host function equality, external modulus selection, countable choice, quotient metric equality, ambient complete metric spaces, or Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite compact-cover exposure, pointwise modulus reads, ProbeBundle radius rows, RatUp lower-bound folds, oscillation-bound ledgers, UniformModulus handoff, transport, replay, provenance, and local naming for the same packet.

Chapter 7649

A Concrete Naming Certificate for $\mathsf{CauchyReblocking}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy tails, the $\mathsf{CauchyReblocking}^{\uparrow}$ packet records finite reblocking of regular rational Cauchy data before any real-name seal is exported. It sits beside the finite-window and tail packets of chapter 126, chapter 301, chapter 291, chapter 5916, and chapter 1420. The packet exposes a finite block schedule and a monotone tail selector; it does not introduce a new real, a quotient completion, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7649.1 (CauchyReblocking carrier)**.** *A* $\mathsf{CauchyReblocking}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, K, T, S, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regular rational readings supply the Cauchy tail data to be reblocked.*
2. *K is the finite block partition row, listing the displayed consecutive blocks and their endpoints.*
3. *T is the monotone tail selector row, assigning each requested tolerance to the block from which all later reads are taken.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing exactly the finite observations consumed by the selected block.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii and comparison cells attached to the selected tail.*
6. *H records componentwise* hsame *transport for the regular source, block partition, tail selector, finite window, dyadic tolerance row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a tail request through R, K, T, S, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyReblocking}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyReblocking}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $R, K, T, S, D, H, C, P, N$. No coordinate stores an independent limit, quotient completion, choice-selected subsequence, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 7649.2 (CauchyReblocking NameCert obligations)**.** *For every accepted* $\mathsf{CauchyReblocking}^{\uparrow}$ *carrier* $B = (R, K, T, S, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyReblocking}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, block-schedule exactness for the finite partition* K*, tail selector monotonicity for* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback preservation for* R*, and Real-seal nonescape for every downstream* $\mathsf{Real}^{\uparrow}$ *consumer. The finite-window and tolerance rows* S, D *are consumed only after* K *and* T *have been displayed, while* H, C, P, N *preserve the same route under transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7649.1. The displayed rows are exactly the regular rational source, finite block partition, monotone selector, finite window, dyadic tolerance row, transport, replay, provenance, and local naming.

Block-schedule exactness is read from K. A downstream consumer may inspect only the finite blocks listed there, and the selected tail is recorded by T rather than by a hidden subsequence or an external choice function.

The regular readback is preserved because R remains the only source row. The finite window S and tolerance row D can be reached only after K and T expose the selected block, so the reblocked read is still a read of the same $\mathsf{RegSeqRat}^{\uparrow}$ source.

Finally, the rows H, C, P, N transport, replay, name, and cite the same finite route. Since the carrier has no coordinate for an independent limit, quotient completion, trichotomy branch, choice schedule, or ambient completeness principle, no downstream $\mathsf{Real}^{\uparrow}$ seal can escape the displayed packet. □

Theorem 7649.3 (CauchyReblocking regular tail selection)**.** *Every tail read exported by an accepted* $\mathsf{CauchyReblocking}^{\uparrow}$ *carrier* $B = (R, K, T, S, D, H, C, P, N)$ *factors through the same finite block schedule before any* $\mathsf{Real}^{\uparrow}$ *seal can be read:*

$$R, K, T, S, D, H, C, P, N.$$

Thus the reblocked tail is a finite reindexing of the source $\mathsf{RegSeqRat}^{\uparrow}$ *row through* K *and* T*, followed by the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, transport, replay, provenance, and local naming rows.*

Proof. Read the carrier of Definition 7649.1. The only rows available before a tail-facing consumer reaches a finite window or tolerance comparison are the source row R, the block partition K, and the selector T.

Use Definition 7649.2. Block-schedule exactness fixes the finite partition, and tail selector monotonicity forces later tail requests to remain inside the scheduled reblocking route.

After the selected block is visible, the consumer may read the finite window S and the dyadic tolerance row D. Transport and replay through H and C preserve the same route, while P and N record provenance and local naming.

Therefore every reblocked tail read factors through $R, K, T, S, D, H, C, P, N$ before any $\mathsf{Real}^{\uparrow}$ seal. The packet supplies finite reindexing data only, not a separate completion or host limit. □

Falsifiable prediction: Every accepted $\mathsf{CauchyReblocking}^{\uparrow}$ read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the $\mathsf{RegSeqRat}^{\uparrow}$ source row, finite block partition, monotone tail selector, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ evidence.

Independence witness: regseqrat, streamname, dyadicratcore, finite-stream, locatedlimit: $\mathsf{CauchyReblocking}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source regular rational row, a

finite block partition, a monotone tail selector, finite-window and tolerance rows, transport, replay, provenance, and local naming as one reblocking packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyReblocking}\uparrow}$.

Constructive story: A $\mathsf{CauchyReblocking}^{\uparrow}$ packet is generated from a RegSeqRat source row, a finite block partition, a monotone tail selector, a StreamName finite window, a DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyReblocking}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7649.1, Definition 7649.2, and Definition 7649.3 seed the finite reblocking route for regular rational Cauchy tails before any Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyReblocking}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected host limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, block-schedule exactness, tail selector monotonicity, RegSeqRat readback preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyReblockingUp.

Chapter 7650

A Concrete Naming Certificate for $\mathsf{RegulatedDarbouxSum}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ packet records finite lower and upper Darboux-sum data for a displayed regulated real function without importing host-level real equality, selected suprema, or measure completion. It sits beside the regulated-function carrier of chapter 1906, the bounded-variation ledger of chapter 398, the dyadic tolerance core of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite integration ledger for later Riemann and Darboux consumers; it names no quotient integral and chooses no ambient completion principle.

Definition 7650.1 (Regulated Darboux-sum carrier)**.** *A* $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, T, L, U, M, O, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, exposing the regulated real function only through its displayed finite window and regulated-approximation interface.*
2. *T is the tagged partition row, listing finitely many subintervals, tags, endpoint order data, and the cell incidence relation read by the Darboux consumer.*
3. *L is the lower-sum row, storing the finite cellwise lower estimates and the displayed addition route by which they are accumulated.*
4. *U is the upper-sum row, storing the finite cellwise upper estimates and the displayed addition route by which they are accumulated.*
5. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, carrying cell widths, requested dyadic tolerance, refinement radius, and endpoint comparison budget.*
6. *O is the oscillation row, recording for each cell the finite readback that bounds the gap between the corresponding upper and lower estimates.*
7. *R is the refinement row, comparing the partition T with a displayed common refinement and transporting the lower, upper, mesh, and oscillation ledgers along the refinement map.*
8. *H records componentwise* hsame *transport for F, T, L, U, M, O, R and for the replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an integration request through the regulated-function source, tagged partition, lower and upper sums, dyadic mesh, oscillation ledger, and refinement row.*
10. *P is the* Pkg *provenance over* $\mathsf{RegulatedDarbouxSum}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedDarbouxSum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, T, L, U, M, O, R, H, C, P, N$. It has no coordinate for selected suprema, Lebesgue measure, host real equality, quotient integration, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7650.2 (Regulated Darboux-sum NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ *carrier $D = (F, T, L, U, M, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the regulated-function source, tagged partition, lower-sum row, upper-sum row, dyadic mesh row, oscillation row, refinement row, transport, replay, provenance, and local naming;*
2. *common-refinement stability, requiring every refinement consumer to read the partition row T and refinement row R before comparing lower and upper sums;*
3. *oscillation-ledger confinement, requiring the gap between the lower and upper rows to be witnessed only by the displayed cellwise oscillation row O;*
4. *dyadic tolerance readback, requiring all mesh widths, slack comparisons, and requested tolerances to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row M;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring every real-facing value read to pass through $F, T, L, U, M, O, R, H, C, P, N$ and exporting no selected supremum, measure completion, host real equality, quotient integral, or choice principle.*

Proof. Project the carrier of Definition 7650.1. The analytic rows are the regulated-function source F, the tagged partition T, the lower and upper sums L, U, the dyadic mesh M, the oscillation ledger O, and the refinement row R; the structural rows are H, C, P, N.

Carrier admission is therefore finite. Each admitted packet exposes exactly the listed rows, so the local NameCert surface has no hidden coordinate from which a selected supremum, measure object, quotient integral, or host equality test could be read.

Common-refinement stability is read through T and R. The refinement row compares two displayed finite partitions and transports only the lower, upper, mesh, and oscillation ledgers already present in the carrier.

The oscillation and tolerance clauses are separate ledger checks. The row O bounds the cellwise upper-lower gap, while the row M carries the dyadic mesh widths, slack, and comparison budget used by that check.

Finally the structural rows H, C, P, N preserve the same order under transport, replay, provenance, and local naming. Any $\mathsf{Real}^{\uparrow}$-facing consumer can therefore see only the displayed finite Darboux-sum route and cannot project selected suprema, Lebesgue measure, host real equality, quotient integration, or countable choice from the packet. □

Theorem 7650.3 (Regulated Darboux-sum finite route)**.** *Every accepted* $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ *read that reaches a real-facing lower or upper value factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}, \quad T, \quad L, \quad U, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad O, \quad R, \quad \mathsf{Real}^{\uparrow}.$$

Thus the handoff to later Darboux or Riemann integration chapters is only a finite ledger over displayed regulated-function, partition, mesh, oscillation, refinement, transport, replay, provenance, and local naming rows.

Proof. Start from an accepted consumer read. By Definition 7650.1, the only source for real-facing lower and upper values is the pair of finite rows L, U attached to the displayed tagged partition T.

The mesh and oscillation controls are forced before the value rows can be used. The dyadic row M records cell widths and tolerance data, while O records the displayed oscillation gap on each cell.

Apply Definition 7650.2. It makes common refinement pass through R and makes the real-facing seal pass through the same finite rows.

The transport, replay, provenance, and naming rows add no analytic coordinates. Hence the consumer receives exactly the finite route stated in the theorem and receives no selected supremum, Lebesgue measure, quotient integral, host real equality, countable choice, or ambient completeness principle. □

Falsifiable prediction: An accepted $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ consumer that reaches a lower or upper real-facing value without displaying the regulated-function source, tagged partition, lower sum, upper sum, dyadic mesh, oscillation ledger, refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: regulated-function, boundedvariation, dyadicratcore, real: $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries tagged partition data, lower and upper sum rows, a dyadic mesh, an oscillation ledger, a common-refinement row, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedDarbouxSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegulatedDarbouxSum}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a tagged partition row, lower and upper finite-sum rows, a DyadicRatCore mesh and tolerance row, a cellwise oscillation ledger, a common-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedDarbouxSum}^{\uparrow}}$, seed)

Scope closed: Definition 7650.1, Definition 7650.2, and Definition 7650.3 seed the finite regulated Darboux-sum ledger over regulated-function, dyadic-mesh, oscillation, refinement, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedDarbouxSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected suprema, Lebesgue measure, host real equality, quotient integration, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement stability, oscillation-ledger confinement, dyadic tolerance readback, RealUp seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.RegulatedDarbouxSumUp.

Chapter 7651

A Concrete Naming Certificate for $\mathsf{ClosedIntervalCompactness}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ packet records the finite dyadic-net test object used by closed real intervals before any open-cover or sequential compactness principle is available. It reads dyadic mesh data from chapter 291, finite name observations from chapter 126, regular rational readback from chapter 301, endpoint seals from chapter 38, and completion-facing finite windows from chapter 1559. The packet names only a finite interval-net route; it is not general Heine–Borel compactness, not open-cover compactness, not sequential compactness, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7651.1 (Closed interval compactness carrier)**.** *A* $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, O, D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper endpoint rows, each sealed through* $\mathsf{Real}^{\uparrow}$ *after finite name inspection.*
2. *O is the endpoint-order row binding the displayed closed interval boundary.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh ledger for the requested tolerance and the finite subdivision cells.*
4. *M is the finite dyadic-net row listing the displayed mesh points and cell radii.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row used to inspect endpoint and mesh-point names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the observed rational windows.*
7. *E is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *handoff row for completion-facing consumers that ask for finite coverage data.*
8. *H records componentwise* hsame *transport for endpoint, order, mesh, window, readback, handoff, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through $A, B, O, D, M, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalCompactness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletionFinite}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for arbitrary open covers, selected subcovers, convergent subsequences, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7651.2 (Closed interval compactness NameCert obligations)**.** *For every accepted* ClosedIntervalCompactn *carrier*

$$K = (A, B, O, D, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactness}^{\uparrow}}$ *surface consists exactly of endpoint admission through* A, B, O*; dyadic mesh admission through* D, M*; finite observation and regular-rational readback through* S, R*; completion-facing finite-window handoff through* E*; componentwise transport through* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 7651.1. The endpoint rows A, B, O are the only interval-boundary data available to the local classifier.

Next read the finite mesh route. The dyadic ledger D fixes the inspected tolerance and subdivision cells, while M lists the displayed mesh points and radii. The window and readback rows S, R are the only way those finite observations reach rational data.

Finally read E, H, C, P, N. The completion-facing handoff is finite because it consumes the same displayed mesh and window rows before any downstream completion read. The structural rows transport, replay, source, and name that route, so no open-cover, subsequence, choice, quotient, or ambient-completeness obligation remains outside the packet. □

Theorem 7651.3 (Closed interval compactness dyadic-net route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ *packet must read the finite route*

$$A, B, O \longmapsto D, M \longmapsto S, R, E \longmapsto H, C, P, N$$

before using the interval as compact finite data. The route exports finite dyadic mesh coverage and readback rows only; it exports no general Heine–Borel theorem, no open-cover compactness, no sequential compactness, no selected subcover, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with endpoint admission. The lower endpoint, upper endpoint, and order row fix the closed interval boundary through displayed $\mathsf{Real}^{\uparrow}$ name seals.

Read the mesh route next. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies the tolerance cells, and the finite net row records the mesh points inspected by the consumer. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows expose only the finite observation windows and rational readbacks attached to those cells.

The $\mathsf{MetricCompletionFinite}^{\uparrow}$ row then repacks the same finite data for completion-facing reads. The transport, continuation, provenance, and local naming rows preserve the displayed route, so the consumer receives a finite dyadic-net packet rather than any excluded compactness or completeness principle. □

Falsifiable prediction: An accepted $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ consumer is refuted if it reaches compact interval data without first displaying endpoint rows, a DyadicRatCore mesh ledger, finite StreamName observations, RegSeqRat readback, and a MetricCompletionFinite handoff.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletionfinite: none is bijective to the closed-interval compactness packet because only $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ binds endpoint Real rows, dyadic mesh coverage, finite observation windows, regular readback, completion-facing handoff, transport, replay, provenance, and local naming as one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalCompactness}^{\uparrow}$ packet is generated from Real endpoint rows, endpoint order, DyadicRatCore mesh ledgers, finite dyadic-net rows, StreamName observations, RegSeqRat readback, MetricCompletionFinite handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7651.1, Definition 7651.2, and Definition 7651.3 seed the finite closed-interval dyadic-net compactness test object.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, open-cover compactness, sequential compactness, selected subcovers, countable choice, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order admission, dyadic mesh coverage, finite observation readback, completion-facing handoff, transport, replay, provenance, and local naming for BEDC.Derived.ClosedIntervalC

Chapter 7652

A Concrete Naming Certificate for $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ packet records real-completeness evidence as finite modulus-ledger data. It consumes the regular Cauchy surface of chapter 1611, the finite completion boundary of chapter 1559, finite stream observations from chapter 126, regular rational readback from chapter 301, and terminal real-name seals from chapter 38. The packet is a ledger for a displayed regular Cauchy request; it is not a selected limit operator, not quotient stream equality, not countable choice, and not an ambient assertion that every host real sequence has a limit.

Definition 7652.1 (Real completeness modulus ledger carrier)**.** *A* $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, K, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row naming the precision and regular Cauchy datum being inspected.*
2. *K is the* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *row that exposes the modulus-controlled tail criterion.*
3. *F is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *row that keeps the completion-facing read inside finite windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the inspected rational windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, K, F, S, R have been displayed.*
7. *H records componentwise* hsame *transport for request, criterion, finite-completion, observation, readback, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a precision request through $Q, K, F, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{MetricCompletionFinite}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host-selected limit, a quotient of streams, a countable-choice schedule, trichotomy, open-cover compactness, or ambient real completeness.

Theorem 7652.2 (Real completeness modulus ledger NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ *carrier*

$$L = (Q, K, F, S, R, E, H, C, P, N),$$

the finite NameCert obligation surface is exactly the RegularCauchyCriterion row K*, MetricCompletionFinite row* F*, RegSeqRat readback row* R*, StreamName observation row* S*, RealUp seal row* E*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}$ *row* N*, all bound to the same request row* Q*. No obligation is discharged by an ambient completeness theorem, quotient stream equality, countable choice schedule, selected limit, or host real-number comparison.*

Proof. Project the carrier of Definition 7652.1. The request row Q fixes the displayed precision and regular Cauchy datum, so every downstream row is read on that same finite ledger entry.

Read K, F, S, R, E in the displayed order. The RegularCauchyCriterion row supplies the tail criterion, the MetricCompletionFinite row keeps the completion read windowed, the StreamName and RegSeqRat rows expose the observed rational data, and the RealUp row seals only that readback.

Finally use H, C, P, N. These rows preserve the request, criterion, completion window, observation, readback, and seal under transport, replay, provenance, and local naming. Because no carrier coordinate stores ambient completeness or selected limits, the named rows exhaust the local obligation surface. □

Theorem 7652.3 (Real completeness modulus ledger obligations)**.** *For every accepted* RealCompletenessModulusLe *carrier*

$$L = (Q, K, F, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}$ *surface consists exactly of finite request admission through* Q*; regular Cauchy criterion exposure through* K*; finite completion-window exposure through* F*; stream observation and regular-rational readback through* S, R*; terminal real-name sealing through* E*; componentwise transport through* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 7652.1. The request row Q fixes one inspected precision and regular Cauchy datum, so the packet begins with finite input rather than an all-sequences completeness principle.

The analytic rows are K, F, S, R, E. The regular Cauchy criterion row supplies the modulus-controlled tail surface, the finite completion row keeps the completion read windowed, the stream and regular-rational rows expose the inspected observations, and the Real row seals only the displayed readback.

The structural rows H, C, P, N transport, replay, source, and name the same ledger. Since the carrier contains no selected limit operator, quotient-stream equality, choice schedule, trichotomy, open-cover compactness, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7652.4 (Real completeness modulus ledger finite route)**.** *Every Real-facing consumer of an accepted* $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ *packet must read the finite ledger route*

$$Q \longmapsto K, F \longmapsto S, R, E \longmapsto H, C, P, N$$

before it can use completion data. The route realizes real-completeness evidence as displayed RegularCauchyCriterion, MetricCompletionFinite, StreamName, RegSeqRat, and RealUp rows, not as a selected limit, quotient stream equality, countable choice, or ambient host completeness theorem.

Proof. Begin with the request row Q. It names the inspected precision and regular Cauchy datum, so the consumer starts from one finite ledger entry.

Next read the criterion and completion boundary. The $\mathsf{RegularCauchyCriterion}^{\uparrow}$ row provides the modulus-controlled tail test, while the $\mathsf{MetricCompletionFinite}^{\uparrow}$ row prevents the completion-facing read from bypassing finite windows.

Finally read the observation, readback, seal, and structural rows. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows expose only finite rational windows before the $\mathsf{Real}^{\uparrow}$ seal is available, and H, C, P, N preserve the same route under transport, replay, provenance, and local naming. □

Falsifiable prediction: An accepted $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ consumer is refuted if it reaches a RealUp seal from completion data without first displaying a regular Cauchy criterion row, finite completion-window row, StreamName observations, and RegSeqRat readback.

Independence witness: regularcauchycriterion, metriccompletionfinite, streamname, regseqrat, real: no listed sibling supplies the full real-completeness modulus ledger, because only RealCompletenessModulusLedg binds a finite request row, regular Cauchy criterion, finite completion window, stream observations, regular readback, Real seal, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletenessModulusLedger}^{\uparrow}$ packet is generated from a finite precision request, RegularCauchyCriterion tail data, MetricCompletionFinite windows, StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7652.1, Definition 7652.3, and Definition 7652.4 seed the finite real-completeness modulus ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessModulusLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, trichotomy, open-cover compactness, ambient host real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked request admission, RegularCauchyCriterion exposure, MetricCompletionFinite handoff, StreamName observation, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessModulusLedgerUp.

Chapter 7653

A Concrete Naming Certificate for $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis on inverse functions over compact real intervals, the $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$ packet records the finite modulus data needed to read a monotone inverse branch without invoking host inverse functions or quotient real equality. It sits over the compact interval route of chapter 3969, the monotone inverse branch of chapter 1929, the uniform-modulus consumer of chapter 369, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a compact-interval inverse-modulus certificate for Real and Completion consumers; it is not an open-cover compactness theorem, a trichotomy principle, a selected inverse function, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 7653.1 (Compact interval inverse modulus carrier)**.** *A* $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, M, F, L, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *compact interval source row, exposing the closed bounded interval, finite-net compactness route, compact-metric row, and endpoint* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *M is the* $\mathsf{MonotoneInverse}^{\uparrow}$ *branch row, naming the monotone inverse branch only through displayed source, target, and order rows.*
3. *F is the finite-net row over K, listing the net centers, interval cells, image comparisons, and radius data inspected by the inverse-modulus route.*
4. *L is the inverse-modulus ledger, recording how target separation requests are discharged by finite source cells, monotone branch data, and compact interval bounds.*
5. *W stores the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened on source endpoints, net centers, branch values, and target requests.*
6. *D stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows for endpoint radii, image gaps, requested target tolerances, and residual readback error.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W and D before any real-facing seal is exposed.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the source interval and the inverse-modulus readback.*

9. *H records componentwise* hsame *transport for the compact interval source, monotone inverse branch, finite-net row, inverse-modulus ledger, stream windows, dyadic tolerance rows, regular readback, Real seal, provenance, and local naming row.*

10. *C records displayed* Cont *replay from compact interval inspection through monotone branch routing, finite-net handoff, tolerance accounting, regular readback, and Real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{MonotoneInverse}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalInverseModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, F, L, W, D, R, S, H, C, P, N$. It has no coordinate for a host inverse function, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7653.2 (Compact interval inverse modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$ *carrier $I = (K, M, F, L, W, D, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalInverseModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact interval exposure, monotone inverse branch exposure, finite-net handoff, inverse-modulus ledger exactness,* $\mathsf{StreamName}^{\uparrow}$ *window visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7653.1. Its analytic rows are K, M, F, L, W, D, R, S, and its structural rows are H, C, P, N.

The compact interval source is upstream. A consumer must first read K, then the monotone branch row M, before the finite-net row F and inverse-modulus ledger L can be inspected.

The finite real-name rows are downstream of that ledger. The stream windows W and dyadic tolerance rows D feed the regular readback R, and the real-facing seal S is reached only after that finite readback is displayed.

Finally H, C, P, N preserve, replay, source, and name the same packet. Since the refused host objects are not carrier coordinates, none can appear as a local NameCert obligation. □

Theorem 7653.3 (Compact interval inverse modulus finite-net handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{ClosedBoundedIntervalCompact}^{\uparrow} \longmapsto \mathsf{MonotoneInverse}^{\uparrow} \longmapsto F, L \longmapsto W, D, R, S \longmapsto H, C, P, N.$$

The handoff supplies compact interval data, monotone inverse branch data, finite-net cells, inverse-modulus ledger rows, stream windows, dyadic tolerances, regular readback, and Real sealing as one finite packet. It exports no host inverse function, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient Real completeness.

Proof. Start with the compact interval source and branch rows. The rows K and M force any inverse-modulus read to expose the compact interval certificate and the monotone inverse branch before a modulus request is evaluated.

Next read the finite-net and ledger rows. The row F lists the finite cells and image comparisons, while L records the inverse-modulus accounting used by the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer.

Then attach the finite real-name handoff. The stream windows W and dyadic tolerances D produce the regular readback R, and the terminal seal S is available only after that readback.

The structural rows H, C, P, N preserve this order. Hence the consumer receives a finite inverse-modulus route, not a host inverse map or an ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalInverseModulus}\uparrow}$.

Constructive story: A $\mathsf{CompactIntervalInverseModulus}^{\uparrow}$ packet is generated from a compact interval source, monotone inverse branch, finite-net row, inverse-modulus ledger, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalInverseModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7653.1, Definition 7653.2, and Definition 7653.3 seed the compact-interval inverse-modulus carrier and its finite-net handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalInverseModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval exposure, monotone branch exposure, finite-net handoff, inverse-modulus ledger exactness, finite real-name readback, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalInverseModulusUp.

Chapter 7654

A Concrete Naming Certificate for $\mathsf{GelfandMazur}^{\uparrow}$

Following Gelfand and Mazur's theorem for complex Banach division algebras, the $\mathsf{GelfandMazur}^{\uparrow}$ packet records the finite certificate surface by which a displayed complex Banach division algebra can be read as scalar-complex data. It sits after the field packet of chapter 45, the complex-number carrier of chapter 39, the norm boundary of chapter 89, the Banach completion surface of chapter 90, and the scalar-linear map rows of chapter 48. The chapter records the visible NameCert rows for that classical route; it does not assert spectral theory, quotient equality, host algebra equality, selected spectrum points, or a complete Banach-algebra library.

Definition 7654.1 (Gelfand-Mazur carrier)**.** *A* $\mathsf{GelfandMazur}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{gm}} = (A, F, C, N, B, I, H, T, P, L)$$

with the following displayed rows:

1. *A is the algebra row carrying the multiplication, unit, distributivity, and scalar-action interface over a displayed complex scalar source.*
2. *F is the* $\mathsf{Field}^{\uparrow}$ *row recording the nonzero division surface for the algebra's invertible part.*
3. *C is the* $\mathsf{Complex}^{\uparrow}$ *scalar row used for scalar embedding, scalar comparison, and terminal complex readback.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *row recording the multiplicative norm bound, unit norm readback, and norm comparison ledger.*
5. *B is the* $\mathsf{Banach}^{\uparrow}$ *row exposing the complete normed carrier surface consumed by completion-facing readers.*
6. *I is the nonzero inverse row, displaying the inverse operation only after the nonzero classifier and field row have been read.*
7. *H records componentwise* hsame *transport for $A, F, C, N, B, I, T, P, L$.*
8. *T records displayed* Cont *replay from the algebra, scalar, norm, Banach, and inverse rows to the scalar-complex readback.*
9. *P is the* Pkg *provenance over* $\mathsf{GelfandMazur}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{GelfandMazur}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and exposes no spectrum selector, host algebra equality, quotient representative, external Banach-algebra library, or detached complex scalar object.

Theorem 7654.2 (Gelfand-Mazur NameCert obligations)**.** *For every accepted* $\mathsf{GelfandMazur}^{\uparrow}$ *carrier* $K_{\mathrm{gm}} = (A, F, C, N, B, I, H, T, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{GelfandMazur}^{\uparrow}}$ *surface consists exactly of algebra admission, field and nonzero-inverse exposure,* $\mathsf{Complex}^{\uparrow}$ *scalar rows, norm and Banach rows, scalar-linear readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Any consumer that reads a Gelfand-Mazur conclusion must first display the sibling routes through chapter 45, chapter 39, chapter 89, chapter 90, and chapter 48. Replacing one of those rows by spectral theory, quotient equality, host algebra equality, selected spectrum points, or a complete Banach-algebra library leaves the accepted carrier.

Proof. Project the carrier of Definition 7654.1. The rows A, F, C, N, B, I display the algebra, division, complex scalar, norm, Banach, and inverse surfaces before any structural row is read.

The sibling dependency surface is explicit. The field row supplies division, the complex row supplies scalar readback, the norm row supplies bounded comparison, the Banach row supplies completion-facing stability, and the linear-map row records the scalar-linear route used by the algebra consumer.

The rows H, T, P, L transport, replay, source, and name the same finite inventory. Since the carrier has no coordinate for spectral theory, quotient equality, host algebra equality, selected spectrum points, or a complete Banach-algebra library, the listed rows exhaust the local NameCert obligations. □

Theorem 7654.3 (Gelfand-Mazur complex scalar collapse)**.** *Every scalar-collapse read from an accepted* $\mathsf{GelfandMazur}^{\uparrow}$ *packet factors through the finite route*

$$A, F, C, N, B, I \longrightarrow H, T, P, L.$$

The read first exposes the algebra and nonzero-inverse rows, then the $\mathsf{Complex}^{\uparrow}$ *scalar row, then the norm and Banach rows, and only then the transport, replay, provenance, and local naming rows. The resulting scalar-complex comparison is a NameCert readback over* K_{gm}*, not a claim of host algebra equality, quotient equality, selected spectrum points, spectral theory, or a complete Banach-algebra library.*

Proof. Start with the algebra row A and the field row F. They display the multiplication, unit, scalar action, and nonzero division surface that a Gelfand-Mazur consumer is allowed to inspect.

Next read the complex, norm, Banach, and inverse rows C, N, B, I. These rows attach scalar-complex comparison to the displayed normed complete carrier and to the nonzero inverse route; they do not introduce a spectrum selector or a host algebra-equality principle.

Finally apply H, T, P, L. Componentwise transport, displayed replay, provenance, and local naming preserve exactly the finite packet, so the scalar-collapse read cannot escape to quotient equality, selected spectrum points, spectral theory, or an external Banach-algebra library. □

Falsifiable prediction: An accepted $\mathsf{GelfandMazur}^{\uparrow}$ read is refuted if a scalar-complex conclusion is reached without displaying the algebra row, FieldUp division row, ComplexUp scalar row, NormUp row, BanachUp row, nonzero inverse row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: field, complex, norm, banach, and linearmap: none is bijective to the Gelfand-Mazur packet because only $\mathsf{GelfandMazur}^{\uparrow}$ simultaneously records algebra admission, complex scalar readback, Banach norm completion, nonzero inverse exposure, scalar-linear comparison, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GelfandMazur}^{\uparrow}}$.

Constructive story: A $\mathsf{GelfandMazur}^{\uparrow}$ packet is generated from an algebra row, FieldUp division row, ComplexUp scalar row, NormUp row, BanachUp row, nonzero inverse row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GelfandMazur}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7654.1, Definition 7654.2, and Definition 7654.3 seed the finite scalar-collapse route for complex Banach division algebra consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GelfandMazur}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, spectral theory, quotient equality, host algebra equality, selected spectrum points, or a complete Banach-algebra library.

Upgrade path: Obligation closure requires checked carrier admission, scalar-collapse route determinacy, nonzero inverse exposure, Banach and norm row compatibility, transport, replay, provenance, and local naming for BEDC.Derived.GelfandMazurUp.

Chapter 7655

A Concrete Naming Certificate for $\mathsf{IntervalArzelaAscoli}^{\uparrow}$

Following Arzela and Ascoli, and Bishop and Bridges, the $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ packet records the closed-interval, compact-metric, and equicontinuity rows needed for a constructive Arzela–Ascoli handoff as finite BEDC data. It consumes the real-name seal of chapter 38, the regular rational readback of chapter 301, the compact-metric finite-net surface of chapter 129, the continuous-map carrier of chapter 130, and the final metric-triangle estimate of chapter 1159. The packet is a compact-uniform handoff for interval-indexed families; it is not an open-cover compactness theorem, not a selected-subsequence construction, not quotient real equality, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7655.1 (Interval Arzela–Ascoli carrier)**.** *An* $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, K, F, W, R, D, T, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the displayed closed interval source row, read through* $\mathsf{Real}^{\uparrow}$ *endpoint seals and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback windows.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row for the interval source.*
3. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose pointwise reads are inspected on the same interval carrier.*
4. *W is the shared finite window ledger for interval points, family indices, and target-value reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *D is the dyadic tolerance row used to compare interval mesh, equicontinuity radius, and target-value error.*
7. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *row that transports local pointwise estimates to a uniform compact-metric estimate.*
8. *U is the compact-uniform handoff row exported to Real or Completion consumers after I, K, F, W, R, D, T have been displayed.*
9. *H records componentwise* hsame *transport for source, compact-metric, map-family, window, readback, tolerance, triangle, handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a precision request through the same interval family route.*
11. *P is the* Pkg *provenance over* $\mathsf{IntervalArzelaAscoli}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalArzelaAscoli}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, K, F, W, R, D, T, U, H, C, P, N$. It has no coordinate for open-cover compactness, selected subsequences, quotient equality of real names or function names, countable choice, host compact-open topology, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7655.2 (Interval Arzela–Ascoli NameCert obligations)**.** *For every accepted* $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ *carrier $A = (I, K, F, W, R, D, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalArzelaAscoli}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, compact-metric finite-net admission through K, continuous-map family admission through F, finite-window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, metric-triangle uniform-estimate exposure through T, compact-uniform handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7655.1. The source rows are the interval row I, compact-metric row K, and continuous-map family row F.

Read the finite numerical surface next. The window row W, readback row R, and dyadic tolerance row D attach each interval point and each inspected map value to finite data before any uniform estimate is considered.

The row T supplies the only metric-triangle uniform estimate, and U is downstream of that row. The structural rows H, C, P, N preserve, replay, source, and name the same packet.

Since the carrier has no open-cover, selected-subsequence, quotient-equality, countable-choice, compact-open host, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7655.3 (Interval Arzela–Ascoli compact-uniform handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ *carrier must read the finite route*

$$I, K, F \longmapsto W, R, D \longmapsto T, U \longmapsto H, C, P, N$$

before treating the interval-indexed family as compact-uniform data. The route consumes $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *and* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *as displayed sibling rows; it exports no open-cover compactness principle, selected subsequence, quotient real equality, host compact-open equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the interval and compact-metric rows. The source interval is read through displayed real-name seals and regular rational endpoint windows, while the compact-metric row supplies only finite-net coverage.

Next inspect the continuous family through F and the shared finite windows W, R, D. These rows keep each pointwise value and tolerance comparison inside the same finite route.

Apply the metric-triangle uniform-estimate row T. It is the only row that transports the pointwise estimates across the compact-metric net into the compact-uniform handoff U.

Thus any consumer that reaches Real or Completion data through the packet receives exactly the displayed finite handoff. A selected subsequence, an open cover, a quotient equality oracle, or an ambient completeness theorem would require a coordinate not present in the carrier. □

Falsifiable prediction: An accepted $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ consumer is refuted if it reads compact-uniform interval-family data without displaying the interval row, compact-metric finite-net row, continuous-map family row, finite windows, RegSeqRat readback, dyadic tolerance row, and metric-triangle uniform-estimate row.

Independence witness: real, regseqrat, compactmetric, continuousmap, metric-triangle-uniform-estimate: none is bijective to the interval Arzela–Ascoli packet because only $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ binds interval source, compact-metric finite nets, continuous family reads, finite windows, rational readback, dyadic tolerances, metric-triangle handoff, transport, replay, provenance, and local naming in one route.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalArzelaAscoli}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalArzelaAscoli}^{\uparrow}$ packet is generated from interval Real rows, RegSeqRat finite readback, compact-metric finite nets, continuous-map family rows, dyadic tolerances, a metric-triangle uniform-estimate row, compact-uniform handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalArzelaAscoli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7655.1, Definition 7655.2, and Definition 7655.3 seed the compact-uniform interval-family carrier surface over Real, RegSeqRat, CompactMetric, ContinuousMap, MetricTriangleUniformEstimate, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalArzelaAscoli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequences, quotient real equality, quotient function equality, host compact-open topology, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source admission, compact-metric finite-net exposure, continuous-map family exposure, finite-window readback, dyadic tolerance exactness, metric-triangle uniform-estimate handoff, structural replay, provenance, and local naming for BEDC.Derived.IntervalArzelaAscoliUp.

Chapter 7656

A Concrete Naming Certificate for DunfordSchwartzMeanErgodicFinite$^{\uparrow}$

Following Dunford–Schwartz and Krengel, the DunfordSchwartzMeanErgodicFinite$^{\uparrow}$ packet records the finite Cesaro-window surface of a contraction-facing mean ergodic route. It sits after the averaging packet of chapter 1840, the observable pullback row of chapter 7271, the finite orbit-average ledger of chapter 6982, the Banach carrier surface of chapter 90, and the terminal real-name seal of chapter 38. The chapter names only finite windows, bounded operator rows, rational Cesaro readbacks, Real sealing, transport, replay, provenance, and local naming; it does not assert strong convergence, invariant projections, quotient ergodic components, almost-everywhere limits, spectral decomposition, or a host Banach-space limit theorem.

Definition 7656.1 (Dunford-Schwartz mean ergodic finite carrier)**.** *A* DunfordSchwartzMeanErgodicFinite$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, O, K, F, W, S, R, A, H, C, P, N)$$

with the following visible rows:

1. *B is the* Banach$^{\uparrow}$ *source row exposing the normed carrier only through displayed finite samples.*
2. *O is the Dunford–Schwartz operator row, recording the contraction and positivity-facing bounds used by the finite averaging request.*
3. *K is the* KoopmanOperator$^{\uparrow}$ *or transfer-facing observable route through which the operator row may be compared with measure-preserving dynamics.*
4. *F is the* BirkhoffErgodicFiniteAverage$^{\uparrow}$ *row carrying finite orbit averages when the request comes from an ergodic observable.*
5. *W is the* MeanErgodic$^{\uparrow}$ *Cesaro-window row, available only as a finite prefix-average ledger.*
6. *S is the* StreamName$^{\uparrow}$ *window row for inspected operator iterates and averaged endpoints.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row for rational norm and endpoint comparisons.*
8. *A is the* Real$^{\uparrow}$ *boundary row sealing the displayed rational comparisons as Real-facing names.*
9. *H records componentwise* hsame *transport for the Banach, operator, Koopman or transfer, finite-average, Cesaro-window, stream, readback, Real, continuation, provenance, and naming rows.*
10. *C records displayed* Cont *replay from an averaging request through $B, O, K, F, W, S, R, A, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}$, $\mathsf{MeanErgodic}^{\uparrow}$, $\mathsf{KoopmanOperator}^{\uparrow}$, $\mathsf{BirkhoffErgodicFiniteAverage}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these twelve rows. It has no coordinate for an invariant projection, selected limit, quotient ergodic component, almost-everywhere convergence theorem, spectral resolution, or ambient Banach-space completeness assertion.

Theorem 7656.2 (Dunford-Schwartz mean ergodic finite NameCert obligations)**.** *For every accepted* $\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}$ *carrier* $G = (B, O, K, F, W, S, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}}$ *surface consists exactly of carrier admission, Banach source exposure, Dunford–Schwartz operator bound exposure, Koopman or transfer comparison, Birkhoff finite-average handoff, MeanErgodic Cesaro-window handoff, StreamName window routing, RegSeqRat readback, Real seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7656.1. Its visible coordinates are precisely the Banach source, operator bound, Koopman or transfer comparison, finite-average handoff, Cesaro-window handoff, stream, readback, Real boundary, transport, replay, provenance, and local naming rows.

Read the analytic rows in order. The Banach row fixes the sampled normed carrier, the operator row records the Dunford–Schwartz finite bound surface, and the Koopman or transfer row explains how observable dynamics feed that operator without identifying the packet with a dynamics chapter.

The averaging rows are downstream of those sources. The Birkhoff row supplies finite orbit averages when present, while the MeanErgodic row supplies the Cesaro prefix ledger. StreamName and RegSeqRat then expose only finite windows and rational readbacks before the Real row seals the displayed endpoint.

Since the carrier has no coordinate for invariant projections, selected limits, quotient ergodic components, almost-everywhere convergence, spectral data, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7656.3 (Dunford-Schwartz mean ergodic finite Cesaro route)**.** *Every Cesaro-facing consumer of an accepted* $\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}$ *carrier factors through the finite route*

$$B \longmapsto O \longmapsto K \longmapsto F \longmapsto W \longmapsto S \longmapsto R \longmapsto A \longmapsto H, C, P, N.$$

Thus a consumer receives the Banach sample row, Dunford–Schwartz operator bounds, Koopman or transfer comparison, Birkhoff finite averages, MeanErgodic Cesaro windows, StreamName windows, RegSeqRat readback, and Real sealing before it can export an averaged endpoint.

Proof. Begin with the carrier definition. The only averaging-facing coordinates are the finite Birkhoff row F and the finite MeanErgodic Cesaro row W, and both are downstream of the Banach, operator, and Koopman or transfer rows.

Apply the NameCert obligations of Definition 7656.2. They force every Cesaro-facing read to expose B, O, K, F, W before the window and readback rows are used.

The StreamName row S and RegSeqRat row R then provide the finite inspected windows and rational comparisons. The Real boundary A is reached only after those finite rows, while H, C, P, N preserve the same route under transport, replay, provenance, and naming.

Therefore the exported endpoint is a finite Cesaro-window readback and not a strong convergence theorem, invariant projection, quotient ergodic component, almost-everywhere limit, spectral decomposition, or host Banach-space limit. □

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}}$.

Constructive story: A $\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}$ packet is generated from a Banach source row, a Dunford–Schwartz finite operator-bound row, a Koopman or transfer comparison row, a Birkhoff finite-average row, a MeanErgodic Cesaro-window row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7656.1, Definition 7656.2, and Definition 7656.3 seed the finite route from Banach and Dunford–Schwartz operator rows through Koopman or transfer comparison, Birkhoff finite averages, MeanErgodic Cesaro windows, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordSchwartzMeanErgodicFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close strong convergence, invariant projections, quotient ergodic components, almost-everywhere limits, spectral decomposition, host Banach-space limit operators, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operator-bound exposure, observable comparison, finite-average handoff, Cesaro-window routing, readback stability, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.DunfordSchwartzMeanErgodicFiniteUp.

Chapter 7657

A Concrete Naming Certificate for $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ packet records finite bilinear product-window data over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyProduct}^{\uparrow}$. It sits beside the product source of chapter 464, the product-stability route of chapter 6271, the finite stream-window surface of chapter 126, the dyadic product ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite bilinear product surface for later Real algebra; it does not use quotient streams, host multiplication, trichotomy, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7657.1 (Regular Cauchy bilinear product carrier)**.** *A* $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, each read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows before a product request is admitted.*
2. *W is the paired product-window row, recording the synchronized finite observations used for the two inputs.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, displaying the requested tolerance, endpoint bounds, cross-error budget, and output tolerance for the paired window.*
4. *Q is the* $\mathsf{CauchyProduct}^{\uparrow}$ *route row that consumes X, Y, W, D as finite product data.*
5. *R is the regular readback row asserting that the product route remains a* $\mathsf{RegSeqRat}^{\uparrow}$ *read rather than a host multiplication result.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the product window, dyadic budget, Cauchy-product route, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the two source rows, product window, dyadic budget, product route, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bilinear product request through $X, Y, W, D, Q, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBilinearProduct}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, Y, W, D, Q, R, E, H, C, P, N$ and has no coordinate for quotient streams, host multiplication, trichotomy, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7657.2 (Regular Cauchy bilinear product NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ *carrier $B = (X, Y, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBilinearPro}}$ *surface consists exactly of carrier admission for the two regular rational source rows, paired product-window exposure, dyadic product-budget exactness, Cauchy-product route admission, regular readback preservation, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7657.1. The analytic rows are exactly the two source rows X, Y, the paired window W, the dyadic budget D, the product route Q, the readback row R, and the seal row E.

Read the product-window side first. The row W exposes only finite observations of the two source names, and D records the product tolerance budget used by that window.

The route row Q consumes those displayed rows as a $\mathsf{CauchyProduct}^{\uparrow}$ handoff. The readback row R keeps the result inside the regular rational surface, and the seal row E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no refused object appears as a coordinate, the listed obligations exhaust the local certificate. □

Theorem 7657.3 (Regular Cauchy bilinear product window bilinearity)**.** *Every accepted* $\mathsf{RegularCauchyBilinearProd}$ *packet reads bilinearity only through the finite route*

$$X, Y, W, D \longmapsto Q \longmapsto R, E, H, C, P, N.$$

Thus a downstream algebra consumer may vary either displayed source row inside the same paired window and dyadic budget only by replaying the corresponding $\mathsf{CauchyProduct}^{\uparrow}$ *route and regular readback; it receives no host multiplication operation, quotient-stream equality, trichotomy branch, selected limit, or ambient completeness principle.*

Proof. Start from the carrier $B = (X, Y, W, D, Q, R, E, H, C, P, N)$. The only product-facing source rows are X and Y, and the only joint observation row is W.

The dyadic budget D records the finite tolerance accounting for the displayed window. Therefore any left-source or right-source product read must pass through the same visible budget before the product route can be consumed.

The row Q is the sole $\mathsf{CauchyProduct}^{\uparrow}$ handoff, while R and E are downstream of that handoff. Transport and replay through H and C preserve this order, and P, N only record provenance and local naming.

Since the carrier has no coordinate for host multiplication or quotient equality, the bilinear behavior available to consumers is exactly the window-level replay route described above. □

Theorem 7657.4 (Regular Cauchy bilinear product Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ *packet factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the paired* $\mathsf{StreamName}^{\uparrow}$ *product window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, the* $\mathsf{CauchyProduct}^{\uparrow}$ *route, and the regular readback row before the terminal seal E is exposed. The packet therefore supplies a non-quotient bilinear product certificate and cannot export quotient streams, host multiplication, trichotomy, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7657.1. The terminal seal E is downstream of X, Y, W, D, Q, R in the displayed tuple.

Apply Definition 7657.2. The local certificate requires paired window exposure, dyadic product-budget exactness, Cauchy-product route admission, and regular readback preservation before any seal handoff.

Use Definition 7657.3. A downstream algebra consumer can replay only the finite window route through Q and R, so it cannot replace the packet by host multiplication or a selected completed product.

The structural rows H, C, P, N preserve the same route. Row exhaustion blocks quotient streams, trichotomy, selected limits, and ambient completeness from crossing the seal boundary. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ read reaching a $\mathsf{Real}^{\uparrow}$ seal must display both $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the paired $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ product budget, the $\mathsf{CauchyProduct}^{\uparrow}$ route, the regular readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: streamname, dyadicratcore, regseqrat, cauchyproduct, real: RegularCauchyBilinearProdu is not bijective to any listed sibling because it jointly carries two source regular rational rows, a paired product window, dyadic product-budget accounting, a Cauchy-product route, regular readback, terminal Real sealing, transport, replay, provenance, and local naming as one bilinear product packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBilinearProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBilinearProduct}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName product windows, a DyadicRatCore product budget, a CauchyProduct route, a regular readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBilinearProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7657.1, Definition 7657.2, Definition 7657.3, and Definition 7657.4 seed the finite bilinear product-window route from regular rational sources to product readback and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBilinearProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, host multiplication, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired product-window exposure, dyadic product-budget exactness, CauchyProduct route admission, regular readback preservation, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBilinearProductUp.

Chapter 7658

A Concrete Naming Certificate for SeparatingHyperplane$^{\uparrow}$

Following Rockafellar and finite-dimensional convex separation theory, the SeparatingHyperplane$^{\uparrow}$ packet records the finite affine-functional surface by which two displayed convex rows are separated inside a finite vector-space window. It sits after the finite alternative row of chapter 6185, the convex carrier of chapter 208, the linear-map row of chapter 48, the vector-space carrier of chapter 47, and the supporting-hyperplane packet of chapter 6895. The packet names only displayed coefficient, inequality, and transport rows; it does not import a host dual space, arbitrary topological separation, selected support points, quotient dual equality, or choice.

Definition 7658.1 (Separating hyperplane carrier)**.** *A* SeparatingHyperplane$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, C_0, C_1, L, a_0, a_1, F, W, H, R, P, N)$$

with the following displayed rows:

1. *V is the finite-dimensional* VecSpace$^{\uparrow}$ *row on which every vector and scalar-action read is performed.*
2. *C_0 and C_1 are the two* ConvexSet$^{\uparrow}$ *rows whose displayed convex-combination operations and inspected point rows are visible to the packet.*
3. *L is the* LinMap$^{\uparrow}$ *functional row, recorded by the finite coefficient list read over V.*
4. *a_0 and a_1 are the scalar level rows marking the two displayed affine sides of the hyperplane comparison.*
5. *F is the* FarkasLemma$^{\uparrow}$ *alternative row used when the separating comparison is obtained from a finite inequality system.*
6. *W is the finite witness ledger saying that every inspected row of C_0 lies on the first side of L and every inspected row of C_1 lies on the second side, with the displayed located slack between a_0 and a_1.*
7. *H records componentwise* hsame *transport for vector, convex, functional, scalar-level, Farkas, witness, replay, package, and local naming rows.*
8. *R records displayed* Cont *replay from a separation request through $V, C_0, C_1, L, a_0, a_1, F, W, H, P, N$.*
9. *P is the* Pkg *provenance over* SeparatingHyperplane$^{\uparrow}$*,* FarkasLemma$^{\uparrow}$*,* ConvexSet$^{\uparrow}$*,* LinMap$^{\uparrow}$*,* VecSpace$^{\uparrow}$*,* FiniteSupportingHyperplane$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SeparatingHyperplane}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a selected ambient dual vector, an arbitrary topological separation theorem, a quotient dual-space equality, or a global family of separating functionals.

Theorem 7658.2 (Separating hyperplane NameCert obligations)**.** *For every accepted* $\mathsf{SeparatingHyperplane}^{\uparrow}$ *carrier* $S = (V, C_0, C_1, L, a_0, a_1, F, W, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatingHyperplane}^{\uparrow}}$ *surface consists exactly of finite vector-space admission, two convex-set admissions, linear-functional display, scalar-level display, finite alternative handoff, one-sided witness exactness, componentwise transport, displayed replay, package provenance, and local naming. The certificate exports no host dual-space object, arbitrary topological separation theorem, selected support point, quotient dual equality, or choice principle.*

Proof. Project the carrier of Definition 7658.1. The rows $V, C_0, C_1, L, a_0, a_1, F, W$ provide the entire geometric and scalar surface of the packet.

The separation read is finite. The functional row L is read through the displayed vector-space row V, the convex information is read through C_0 and C_1, and the finite alternative row F supplies only the multiplier or feasibility ledger already visible to the packet.

Rows H, R, P, N transport, replay, source, and name the same finite data. Since the carrier lacks any coordinate for the excluded ambient or choice-based objects, the listed rows exhaust the local NameCert obligations. □

Theorem 7658.3 (Separating hyperplane Farkas route)**.** *Every Farkas-facing consumer of an accepted* $\mathsf{SeparatingHyperplane}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{FarkasLemma}^{\uparrow} \longmapsto \mathsf{FiniteSupportingHyperplane}^{\uparrow}.$$

The consumer receives the same finite vector window, two convex rows, a displayed linear functional, scalar levels, the finite alternative row, the one-sided witness ledger, transport, replay, provenance, and local naming. It receives no detached separation principle and no selected ambient dual object.

Proof. Begin with a Farkas-facing separation request. The vector and convex data are read through V, C_0, C_1, so all subsequent coefficient rows remain tied to the displayed finite-dimensional surface.

Read L, a_0, a_1 next. These rows provide the affine comparison seen by the consumer, while the finite alternative row F explains the inequality ledger without invoking a topological separation theorem.

Finally compare the witness row W with the supporting-hyperplane sibling surface. The supporting packet supplies the one-sided boundary reading, and the present packet carries the two-sided separation ledger through the same structural rows H, R, P, N.

Therefore the public route is the displayed finite chain above. Any read bypassing those rows would require a carrier coordinate that the packet does not contain. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatingHyperplane}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatingHyperplane}^{\uparrow}$ packet is generated from finite VecSpaceUp data, two ConvexSetUp rows, a LinearMapUp functional row, scalar-level rows, a FarkasLemmaUp alternative handoff, a finite one-sided witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeparatingHyperplane}^{\uparrow}}$, seed)

Scope closed: Definition 7658.1, Definition 7658.2, and Definition 7658.3 seed the finite separating-hyperplane packet over FarkasLemmaUp, ConvexSetUp, LinearMapUp, VecSpaceUp, FiniteSupportingHyperplaneUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{SeparatingHyperplane}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary topological separation, selected ambient dual vectors, quotient dual equality, selected support points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite alternative handoff, one-sided witness exactness, componentwise transport, replay, provenance, and local naming for BEDC.Derived.SeparatingHyperplaneUp.

Chapter 7659

A Concrete Naming Certificate for $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$

The $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ packet realizes the boundary-quadruple branch of chapter 9463 as finite NameCert data. It sits beside the disk carrier of chapter 377, the metric row of chapter 3474, the boundary row of chapter 3491, the cross-ratio packet of chapter 3948, and the ideal-triangle packet of chapter 6544. The packet names four displayed boundary ends, their side and diagonal incidence, a finite cross-ratio read, transport, replay, provenance, and local naming; it does not complete the disk, quotient boundary endpoints, select global geodesic representatives, or identify observer expansion with physical spacetime.

Definition 7659.1 (Hyperbolic ideal quadrilateral carrier)**.** *A* $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, M, B, E, D, X, T, J, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the inspected disk probes and four boundary-end reads.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *row naming four cyclically ordered boundary ends and the finite incidence data used by the packet.*
4. *E records the four boundary-to-boundary side rows and the two diagonal rows as displayed geodesic-boundary data.*
5. *D is the diagonal-splitting row that reads the two ideal triangles cut from the quadrilateral through the sibling* $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$ *surface.*
6. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row carrying the finite boundary cross-ratio comparison attached to the same four-end boundary surface.*
7. *T is the transport row preserving disk coordinates, metric reads, boundary incidence, side rows, diagonal rows, triangle-splitting data, and cross-ratio data.*
8. *J is the classifier-stability row for cyclic order, side incidence, diagonal incidence, cross-ratio readback, and triangle-splitting reads.*
9. *H records componentwise* hsame *transport for disk, metric, boundary, side, diagonal, cross-ratio, classifier, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an ideal-quadrilateral request through* $U, M, B, E, D, X, T, J, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBou}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed hyperbolic disks, quotient ideal-boundary equality, selected global geodesics, host analytic flow, or physical cosmology.

Theorem 7659.2 (Hyperbolic ideal quadrilateral NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ *carrier* $Q = (U, M, B, E, D, X, T, J, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{HyperbolicIdealQuadrilat}}$ *surface is exhausted by disk-coordinate admission, hyperbolic metric visibility, four-end boundary incidence, side and diagonal visibility, triangle-splitting readback, finite cross-ratio readback, transport, classifier stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7659.1. The rows $U, M, B, E, D, X, T, J, H, C, P, N$ are the only displayed coordinates.

The geometric read begins with U, M, B, E. Disk coordinates and metric reads are visible before the four boundary ends, side rows, and diagonal rows are inspected.

The finite boundary-quadruple structure is carried by D, X, T, J. Triangle splitting reads the two diagonal pieces through the ideal-triangle sibling, while the cross-ratio row attaches one finite four-end comparison to the same boundary surface. Transport and classifier stability keep these rows on the same packet.

Rows H, C, P, N preserve, replay, source, and name the finite carrier. Since no row stores the excluded completed or host-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 7659.3 (Hyperbolic ideal quadrilateral boundary-quadruple handoff)**.** *Every boundary-quadruple consumer of an accepted* $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicCrossRatio}^{\uparrow} \longmapsto \mathsf{HyperbolicIdealT}$$

The consumer receives four displayed boundary ends, side and diagonal incidence, the finite cross-ratio row, triangle-splitting readback, transport, classifier stability, componentwise transport, replay, provenance, and local naming. It obtains no completed disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance.

Proof. Begin with a boundary-quadruple request. The disk-coordinate row U and metric row M provide the only geometric source data available to the request.

Read B and E next. The four boundary ends, four sides, and two diagonals are displayed as finite incidence and geodesic-boundary rows, not as quotient endpoints or selected global geodesics.

Use X, D, T, J for the handoff. The cross-ratio row records the finite four-end comparison, the diagonal-splitting row exposes the two ideal-triangle reads, and transport plus classifier stability preserve the same boundary commitments.

Finally replay the structural rows H, C, P, N. They preserve the same finite packet, and the carrier has no coordinate for the excluded completed or host-level objects, so the boundary-quadruple handoff cannot leave the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ consumer that exports a boundary-quadruple comparison without the four-end boundary row B and the cross-ratio row X refutes the carrier surface.

Independence witness: unitdisk, hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, hyperbolic-ideal-triangle: none is bijective to $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ because this packet alone binds four boundary ends, side and diagonal incidence, triangle-splitting readback, cross-ratio comparison, transport, classifier stability, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}$ packet is generated from UnitDiskUp coordinate rows, HyperbolicMetricUp metric reads, HyperbolicGeodesicBoundaryUp four-end boundary incidence, side and diagonal rows, HyperbolicIdealTriangleUp splitting readback, HyperbolicCrossRatioUp comparison, transport, classifier stability, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7659.1, Definition 7659.2, and Definition 7659.3 seed the finite boundary-quadruple route over disk, metric, boundary, side, diagonal, triangle-splitting, cross-ratio, transport, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicIdealQuadrilateral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic disks, quotient ideal-boundary equality, selected global geodesics, host analytic flows, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk visibility, metric-read visibility, four-end boundary incidence, side and diagonal visibility, triangle splitting, cross-ratio readback, transport, classifier stability, ledger nonescape, and local naming for BEDC.Derived.HyperbolicIdealQuadrilateralUp.

Chapter 7660

A Concrete Naming Certificate for $\mathsf{BishopRegularRealLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularRealLimit}^{\uparrow}$ packet records the finite limit data carried by a regular rational real name. It sits over the dyadic tolerance arithmetic of chapter 291, the finite schedule rows of chapter 126, the regular rational source of chapter 301, the uniqueness boundary of chapter 3942, and the terminal real-name seal of chapter 38. The packet names a Bishop-style limit route; it does not quotient streams, select a limit representative, use countable choice, assert host equality of real objects, or claim full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7660.1 (Bishop regular real limit carrier)**.** *A* $\mathsf{BishopRegularRealLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, T, U, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the inspected tail scales.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row for reading rational observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row over the displayed schedule.*
4. *T is the Bishop tail-test row recording that later displayed reads stay within the dyadic tolerance demanded by D.*
5. *U is the* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *comparison row, available only when two displayed candidate-limit reads are compared through the same regular route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R, T, U have been exposed.*
7. *H records componentwise* hsame *transport for the tolerance, schedule, readback, tail-test, uniqueness, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite schedules and dyadic tolerances to the regular readback, tail-test, uniqueness boundary, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularRealLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, host real equality, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 7660.2 (Bishop regular real limit NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularRealLimit}^{\uparrow}$ *carrier* $L = (D, S, R, T, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealLimit}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance visibility, finite schedule admission, regular rational readback, Bishop tail-test exposure, regular Cauchy uniqueness boundary exposure, terminal real sealing, componentwise transport, displayed replay, provenance, and local naming. Every limit-facing read passes through*

$$D, S, R, T, U, E, H, C, P, N$$

and through no quotient stream equality, selected limit, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7660.1. The rows D, S, R display the tolerance ledger, schedule, and regular readback before any limit-facing row is inspected.

Read the Bishop tail test next. The row T is attached to the same displayed schedule and dyadic ledger, so it records only finite tail data. When two candidate-limit reads are compared, U routes the comparison through $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ before the real seal E is public.

The structural rows H, C, P, N transport, replay, source, and locally name this route. Since the excluded principles have no carrier coordinate, the listed rows exhaust the local $\mathsf{NameCert}$ obligations. □

Theorem 7660.3 (Bishop regular real limit regularity induction)**.** *Every accepted* $\mathsf{BishopRegularRealLimit}^{\uparrow}$ *packet supports induction on displayed finite schedules: the base read is the first* $\mathsf{StreamName}^{\uparrow}$ *window checked against* D*, and the induction step appends one later window whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback satisfies the same Bishop tail-test row* T*. The induction exports only the preserved route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyLimitUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

and no quotient, selector, choice, host equality, or ambient completeness data.

Proof. The base case is the displayed schedule row S read against the dyadic ledger D. Its regular rational readback is the row R, and the tail-test row T records the admitted tolerance.

For the induction step, append one finite window to S and read it through the same dyadic ledger. Regularity is preserved because the appended read is admitted only when it satisfies T and therefore remains on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ route.

The uniqueness and seal rows are downstream. The row U compares only displayed candidate-limit reads, and E is reached only through the same finite route. The excluded completion principles are absent from the carrier, so the induction cannot export them. □

Theorem 7660.4 (Bishop regular real limit seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopRegularRealLimit}^{\uparrow}$ *carrier receives the terminal seal* E *only after the displayed dyadic tolerance ledger, stream schedule, regular rational readback, Bishop tail-test row, and* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *boundary have been exposed. The handoff exports finite regular real limit data and not quotient streams, selected limits, countable-choice schedules, host real equality, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the obligation theorem to read D, S, R, T, U before E. This places the terminal real seal behind the same finite tolerance, schedule, readback, and uniqueness route.

Apply regularity induction to the inspected finite schedule. It preserves the route through appended windows and prevents the consumer from replacing the displayed rows by a selected representative or quotient stream.

Finally use H, C, P, N to transport, replay, source, and locally name the same handoff. No excluded completion coordinate is present, so the public seal is only the displayed regular real limit route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularRealLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularRealLimit}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite schedules, RegSeqRat readback, Bishop tail-test data, a RegularCauchyLimitUniqueness boundary, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularRealLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7660.1, Definition 7660.2, Definition 7660.3, and Definition 7660.4 seed the finite Bishop regular real limit route over dyadic tolerances, stream schedules, regular rational readback, uniqueness, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularRealLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule locality, regular rational readback, Bishop tail-test exposure, uniqueness-boundary exposure, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularRealLimitUp.

Chapter 7661

A Concrete Naming Certificate for $\mathsf{LayeredRelationCertificate}^{\uparrow}$

The $\mathsf{LayeredRelationCertificate}^{\uparrow}$ packet realizes the relation-before-identity discipline of chapter 9416 as finite NameCert data. It sits beside the compact same-schema packet of chapter 1090, the gate verdict of chapter 1099, the open-problem ledger of chapter 780, and the observer-history identity surface of chapter 479. The packet is deliberately larger than a same-schema row: it stores source, carrier, classifier, preserved, not-preserved, exactness, failure-boundary, refusal, and local naming data in one auditable surface.

Definition 7661.1 (Layered relation certificate carrier)**.** *A* $\mathsf{LayeredRelationCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, B, K, M, V, W, E, F, Q, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source-domain rows being compared.*
2. *K is the finite carrier ledger for the layers read from both sources.*
3. *M is the classifier row assigning each layer to its displayed relation role.*
4. *V is the preserved-structure row.*
5. *W is the not-preserved row.*
6. *E is the exactness row stating the admitted same-schema strength.*
7. *F is the failure-boundary row naming when the relation weakens or is rejected.*
8. *Q is the refusal row excluding object identity, empirical transfer, subject-substance transfer, global-frame transfer, hidden axioms, and unlisted bridge evidence.*
9. *H records componentwise* hsame *transport for all source, carrier, classifier, preservation, refusal, exactness, boundary, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same rows.*
11. *P is the* Pkg *provenance over* $\mathsf{LayeredRelationCertificate}^{\uparrow}$, $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{LayeredRelationGate}^{\uparrow}$, $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCertificate}^{\uparrow}}$ *row.*

The classifier admits a relation read only when W is present with V, F is present with E, and Q is present with A, B, K, M. It has no coordinate for identity collapse or unrecorded bridge transfer.

Theorem 7661.2 (Layered relation certificate gate obligations)**.** *For every accepted* $\mathsf{LayeredRelationCertificate}^{\uparrow}$ *carrier* $R = (A, B, K, M, V, W, E, F, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCertificate}^{\uparrow}}$ *surface consists exactly of finite source admission, carrier ledger admission, classifier exposure, preserved-row exposure, not-preserved-row exposure, exactness exposure, failure-boundary exposure, refusal exposure, componentwise transport, displayed replay, provenance, and local naming. A packet without the not-preserved row W is rejected by the gate.*

Proof. Project the carrier of Definition 7661.1. The source rows A, B, carrier ledger K, and classifier row M determine which relation is being read.

The gate then reads preservation and refusal together. The row V can be consumed only with the not-preserved row W, and the exactness row E can be consumed only with the failure-boundary row F. The refusal row Q blocks identity collapse and hidden bridge transfer.

Finally H, C, P, N transport, replay, source, and name the same finite packet. If W is absent, the preserved-row read lacks its refusal boundary, so the packet is not an accepted layered relation certificate. □

Theorem 7661.3 (Layered relation certificate exactness boundary)**.** *Every downstream consumer of an accepted* $\mathsf{LayeredRelationCertificate}^{\uparrow}$ *packet reads same-schema exactness only through*

$$A, B, K, M, V, W, E, F, Q, H, C, P, N.$$

The consumer may reuse the packet as relation evidence, but it may not replace either source by the other, infer preservation of a row listed in W, ignore the failure boundary F, bypass the refusal row Q, or treat the certificate as bridge-checked data.

Proof. Start from the gate obligations. They force the consumer to read the two sources, carrier ledger, classifier, preserved row, not-preserved row, exactness row, failure boundary, and refusal row before any relation evidence is exported.

The exactness row is therefore bounded by the not-preserved and failure rows. If a consumer asks for object identity, empirical transfer, subject-substance transfer, global-frame transfer, hidden axiom support, or bridge evidence, the request is routed to Q rather than to E.

The transport and replay rows preserve this boundary. Provenance and local naming record the packet as relation evidence only, so the exactness read cannot become an unlisted bridge or identity claim. □

Theorem 7661.4 (Layered relation certificate sibling independence)**.** *The* $\mathsf{LayeredRelationCertificate}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{LayeredRelationGate}^{\uparrow}$, $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$, *or* $\mathsf{ObserverHistIdentity}^{\uparrow}$. *The simultaneous presence of two source rows, a carrier ledger, a classifier row, preserved and not-preserved rows, exactness, failure boundary, refusal, transport, replay, provenance, and local naming is required for this packet.*

Proof. The compact $\mathsf{LayeredRelationCert}^{\uparrow}$ sibling supplies same-schema relation scope, but it does not by itself store the full gate refusal and open-ledger boundary used here.

The $\mathsf{LayeredRelationGate}^{\uparrow}$ sibling supplies a verdict route, while $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$ supplies unresolved-boundary ledgers and $\mathsf{ObserverHistIdentity}^{\uparrow}$ supplies observer-history identity discipline. Each omits at least one of $A, B, K, M, V, W, E, F, Q, H, C, P, N$.

Therefore no listed sibling is bijective to the packet while preserving all NameCert rows. The independence witness is the combined surface named by the carrier. □

Falsifiable prediction: Any proposed $\mathsf{LayeredRelationCertificate}^{\uparrow}$ packet without a displayed not-preserved row W fails the gate and must be routed to the refusal row rather than consumed as same-schema relation evidence.

Independence witness: layeredrelationcert, layered-relation-gate, metacicopenproblemledger, observerhistidentity: none is bijective to $\mathsf{LayeredRelationCertificate}^{\uparrow}$ because none carries the simultaneous source, carrier, classifier, preserved, not-preserved, exactness, failure-boundary, refusal, transport, replay, provenance, and local NameCert rows.

Future Lean work should provide Nontrivial and FieldFaithful instances for $\mathsf{LayeredRelationCertificate}^{\uparrow}$ inside BEDC.Derived.LayeredRelationCertificateUp.TasteGate.

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{LayeredRelationCertificate}^{\uparrow}$ packet is generated from two source rows, a carrier ledger, a classifier row, preserved and not-preserved rows, exactness, failure-boundary, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LayeredRelationCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7661.1, Definition 7661.2, Definition 7661.3, and Definition 7661.4 seed the finite relation-certificate surface that separates same-schema evidence from identity and hidden bridge transfer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LayeredRelationCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close object identity, empirical transfer, subject-substance transfer, global-frame transfer, hidden axiom support, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, preserved and not-preserved row pairing, exactness and failure-boundary pairing, refusal, transport, replay, provenance, and local naming for BEDC.Derived.LayeredRelation

Chapter 7662

A Concrete Naming Certificate for $\mathsf{DeterministicPushdownAutomaton}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{DeterministicPushdownAutomaton}^{\uparrow}$ packet records the deterministic finite-control stack recognizer surface as displayed BEDC data. It refines the stack-recognizer carrier of chapter 6018, reads terminal words through the $\mathsf{List}^{\uparrow}$ spine of chapter 36, uses the $\mathsf{Nat}^{\uparrow}$ cursor discipline of chapter 28, and sits beside the $\mathsf{ContextFreeGrammar}^{\uparrow}$ and $\mathsf{RegularLanguage}^{\uparrow}$ language interfaces of chapter 356 and chapter 351. The packet names a single deterministic accepting computation; it does not import a host language set, an ambiguity quotient, or an infinite trace.

Definition 7662.1 (Deterministic pushdown automaton carrier)**.** *A* $\mathsf{DeterministicPushdownAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F, w, \rho, \sigma, e, \chi, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite control-state carrier row.*
2. *Σ is the finite input alphabet row.*
3. *Γ is the finite stack alphabet row.*
4. *δ is the deterministic transition ledger. Each row reads one control state, either one input symbol or ε, and one top stack-symbol row, then returns one next-state row and one finite stack-word replacement row.*
5. *q_0 is the start-state row inside Q.*
6. *Z_0 is the initial-stack row inside Γ.*
7. *F is the accepting-state row inside Q.*
8. *w is the finite input word as a $\mathsf{List}^{\uparrow}$ row over Σ.*
9. *ρ is the finite run ledger beginning at (q_0, Z_0) and consuming w through the displayed deterministic transition rows.*
10. *σ is the stack-history ledger whose entries are finite $\mathsf{List}^{\uparrow}$ rows over Γ.*
11. *e is the endpoint readback for the run ledger.*

12. *χ is the acceptance readback, by final state or by the displayed empty-stack convention named by the packet.*
13. *H stores componentwise* hsame *transports for state, input, stack, transition, run, stack-history, endpoint, acceptance, and classifier rows.*
14. *C stores the* Cont *replay routes for reading input symbols, ε choices, stack-top rows, transition entries, and the finite run ledger.*
15. *P stores the* Pkg *provenance over* $\mathsf{PushdownAutomaton}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
16. *N stores the local* $\mathsf{NameCert}_{\mathsf{DeterministicPushdownAutomaton}^{\uparrow}}$ *rows.*

The carrier predicate is $\mathsf{DeterministicPushdownAutomatonPacket}(D)$. *It contains no coordinate for a host language set, nondeterministic branch choice, ambiguity witness, selected parse tree, or infinite accepting trace.*

Lean4 (definition) `BEDC.Derived.DeterministicPushdownAutomatonUp.TasteGate.DeterministicPushdownAutomatonU`
Lean4 ✓ `BEDC.Derived.DeterministicPushdownAutomatonUp.TasteGate.DeterministicPushdownAutomatonTasteGate`
`single carrier alignment`

Theorem 7662.2 (Deterministic pushdown automaton NameCert obligations)**.** *For every accepted* $\mathsf{DeterministicPushdownAutomaton}^{\uparrow}$ *carrier*

$$D = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F, w, \rho, \sigma, e, \chi, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DeterministicPushdownAutomaton}^{\uparrow}}$ *surface consists exactly of carrier admission, deterministic transition uniqueness for each state/input-or-ε/stack-top row, stack-row stability, run replay exactness, acceptance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must read through the finite rows in Definition 7662.1; no consumer obtains a host language set, a nondeterministic branch selector, a selected parse tree, or an infinite trace.*

Proof. Project the accepted carrier D. The rows $Q, \Sigma, \Gamma, \delta, q_0, Z_0, F$ name the finite control, input alphabet, stack alphabet, deterministic transition table, start state, initial stack, and accepting states. The rows w, ρ, σ, e, χ name the inspected input, the unique run ledger, the stack-history ledger, the endpoint, and the acceptance readback.

For transition uniqueness, fix a displayed state, either an input-symbol row or ε, and a top stack-symbol row. The ledger δ has at most one matching entry, and the run row ρ records only that entry at the corresponding cursor and stack position.

Stack-row stability follows by reading each step of ρ together with the matching stack row of σ. The next stack row is obtained by popping the displayed top symbol and pushing the finite $\mathsf{List}^{\uparrow}$ replacement named by the unique transition entry.

Run replay exactness is the finite induction over the displayed word and ε-step cursor ledger. At the endpoint, χ reads only the final control state or the displayed empty-stack row named by the carrier.

The rows H, C, P, N transport, replay, source, and locally name exactly these coordinates. Since the carrier has no coordinate for a host language set, branch selector, parse-tree choice, or infinite trace, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeterministicPushdownAutomaton}^{\uparrow}}$.

Constructive story: A DeterministicPushdownAutomaton$^{\uparrow}$ packet is generated from finite control rows, input alphabet rows, stack alphabet rows, a deterministic transition ledger, start and initial-stack rows, accepting rows, a finite input word, a finite run ledger, a stack-history ledger, an endpoint readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{DeterministicPushdownAutomaton}^{\uparrow}}$, seed)

Scope closed: Definition 7662.1 and Definition 7662.2 seed the deterministic finite stack-recognizer route over PushdownAutomatonUp, ContextFreeGrammarUp, RegularLanguageUp, ListUp, NatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DeterministicPushdownAutomaton}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, equivalence with all context-free grammars, language-set semantics, ambiguity analysis, selected parse trees, nondeterministic branch choice, or infinite trace semantics.

Upgrade path: Obligation closure requires checked carrier admission, deterministic transition uniqueness, stack-row stability, run replay exactness, acceptance readback, and ledger nonescape for BEDC.Derived.DeterministicPushdownAutomatonUp.

Chapter 7663

A Concrete Naming Certificate for $\mathsf{HamiltonJacobi}^{\uparrow}$

Following Evans and Fleming, the $\mathsf{HamiltonJacobi}^{\uparrow}$ packet records a finite dynamic-programming and variational-control surface for Hamilton-Jacobi readback. It sits beside the $\mathsf{Ode}^{\uparrow}$ flow rows of chapter 193, the $\mathsf{DynSystem}^{\uparrow}$ transition rows of chapter 195, the finite quadratic-control ledger of chapter 296, the Riccati dependency row of chapter 307, the $\mathsf{Real}^{\uparrow}$ scalar surface of chapter 38, the $\mathsf{Mat}^{\uparrow}$ coefficient rows of chapter 49, and the $\mathsf{VecSpace}^{\uparrow}$ state rows of chapter 47. The packet names the finite carrier through which later optimal-control consumers may request a Hamiltonian row, value-slope row, and dynamic-programming replay without importing analytic viscosity solutions, global PDE existence, selected optimal controls, host differentiability, or ambient completeness.

Definition 7663.1 (Hamilton-Jacobi carrier)**.** *A* $\mathsf{HamiltonJacobi}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (X, U, F, L, H, V, D, R, Q, A, T, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{VecSpace}^{\uparrow}$ *state row and U is the control-row domain visible to the finite dynamic-programming request.*
2. *F is the* $\mathsf{DynSystem}^{\uparrow}$ *or* $\mathsf{Ode}^{\uparrow}$ *transition row tying a state-control pair to the displayed successor row.*
3. *L is the* $\mathsf{Real}^{\uparrow}$ *running-cost row, and H is the Hamiltonian row computed from the displayed transition and cost rows.*
4. *V is the value-row name and D is the finite slope or subgradient witness row used only as packet data.*
5. *R is the* $\mathsf{Riccati}^{\uparrow}$ *comparison row available when the displayed data come from a quadratic linear-control packet.*
6. *Q is the* $\mathsf{LQR}^{\uparrow}$ *dependency row recording the finite dynamic-programming route for the quadratic specialization.*
7. *A is the* $\mathsf{Mat}^{\uparrow}$ *coefficient row used by the linear and quadratic control fragments.*
8. *T records componentwise* hsame *transport for $X, U, F, L, H, V, D, R, Q, A, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{HamiltonJacobi}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{Riccati}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HamiltonJacobi}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, U, F, L, H, V, D, R, Q, A, T, P, N$ and has no coordinate for analytic viscosity solutions, global PDE existence, selected optimal controls, host differentiability, or ambient completeness.

Theorem 7663.2 (Hamilton-Jacobi NameCert obligations)**.** *For every accepted* $\mathsf{HamiltonJacobi}^{\uparrow}$ *carrier $J = (X, U, F, L, H, V, D, R, Q, A, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HamiltonJacobi}^{\uparrow}}$ *surface consists exactly of carrier admission, state-control exposure, transition or ODE route admission, running-cost and Hamiltonian row exposure, value and slope-row readback, Riccati and LQR dependency availability, matrix-coefficient exposure, componentwise* hsame *transport,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7663.1. The state-control rows are X, U, the transition side is F, the scalar cost and Hamiltonian rows are L, H, and the value-readback rows are V, D.

The finite control dependencies are displayed separately. Rows R and Q expose the Riccati and LQR routes only when a quadratic linear-control packet is being read, while A records the matrix coefficients used by that comparison.

The structural rows T, P, N transport, source, and name the same packet. Since the carrier contains no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7663.3 (Hamilton-Jacobi LQR-Riccati route)**.** *Every downstream* $\mathsf{LQR}^{\uparrow}$ *or* $\mathsf{Riccati}^{\uparrow}$ *consumer of an accepted* $\mathsf{HamiltonJacobi}^{\uparrow}$ *packet must pass through the displayed state row X, control row U, transition row F, cost row L, Hamiltonian row H, value row V, slope row D, matrix row A, transport row T, provenance row P, and local naming row N before it can read the quadratic-control rows Q and R. The route exports no analytic viscosity solution, global PDE existence theorem, selected optimal controller, host differentiability, or ambient completeness principle.*

Proof. Begin with the NameCert obligations of Definition 7663.2. They require the state-control, transition, cost, Hamiltonian, value, slope, matrix, transport, provenance, and local naming rows to be visible in the same finite packet.

The LQR and Riccati rows are dependency rows of that packet, not external solvers. A consumer that wants the quadratic-control face must therefore read Q and R through the displayed Hamilton-Jacobi carrier rather than through a host optimal-control theorem.

Because no carrier coordinate stores analytic viscosity solutions, global PDE existence, selected optimal controls, host differentiability, or ambient completeness, the only available LQR-Riccati route is the finite route displayed by $X, U, F, L, H, V, D, R, Q, A, T, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{HamiltonJacobi}^{\uparrow}$ read reaching an $\mathsf{LQR}^{\uparrow}$ or $\mathsf{Riccati}^{\uparrow}$ dependency must display the state row, control row, transition or ODE row, cost row, Hamiltonian row, value row, slope row, matrix row, componentwise hsame transport, Pkg provenance, and local NameCert row.

Independence witness: ode, dynsystem, lqr, riccati, real, matrix, vecspace: $\mathsf{HamiltonJacobi}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite state-control row, transition route, Hamiltonian readback, value and slope rows, LQR and Riccati dependency rows, matrix coefficient row, transport, provenance, and local naming as one variational-control packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HamiltonJacobi}^{\uparrow}}$.

Constructive story: A $\mathsf{HamiltonJacobi}^{\uparrow}$ packet is generated from finite state-control rows, an Ode or DynSystem transition row, a RealUp running-cost row, a Hamiltonian row, value and slope rows, Riccati and LQR dependency rows, a MatrixUp coefficient row, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{HamiltonJacobi}\uparrow}$, seed)

Scope closed: Definition 7663.1, Definition 7663.2, and Definition 7663.3 seed the finite variational-control carrier, NameCert obligation surface, and LQR-Riccati dependency route.

Formal verification: FormalStatus(NameCert$_{\mathsf{HamiltonJacobi}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic viscosity solutions, global PDE existence, selected optimal controls, host differentiability, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, state-control exposure, transition route admission, running-cost and Hamiltonian row exposure, value and slope-row readback, Riccati and LQR dependency availability, matrix-coefficient exposure, transport, provenance, and local naming for BEDC.Derived.HamiltonJacobiUp.

Chapter 7664

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ packet records the filter-facing universal-property route for regular Cauchy real names in the L10 Real and Completion spine. It sits after the finite observation surface of chapter 126, the regular rational readback of chapter 301, the criterion surface of chapter 1611, the Cauchy-filter source of chapter 333, and the terminal real-name seal of chapter 38. The packet names a universal readout through displayed rows only; it does not quotient streams, select limits, compare host real equality, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7664.1 (RegularCauchyFilterUniversalProperty carrier)**.** *A* $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (W, R, K, F, B, T, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger, recording the inspected stream stages and filter requests.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values observed through W.*
3. *K is the* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *row certifying that the same windows and readbacks meet the regular Cauchy tail discipline.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row reached from W, R, K and used as the filter-facing input to the universal property.*
5. *B is the* $\mathsf{Real}^{\uparrow}$ *boundary row reached only after the regular Cauchy criterion and Cauchy-filter source rows have been displayed.*
6. *T records componentwise* hsame *transport for the window, readback, criterion, filter source, Real boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$W, R, K, F \longmapsto B, T, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, R, K, F, B, T, C, P, N$. It has no coordinate for quotient stream equality, a selected limit representative, host real equality, countable choice, or ambient Real completeness.

Theorem 7664.2 (RegularCauchyFilterUniversalProperty NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ *carrier $U = (W, R, K, F, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchy}}$ *surface consists exactly of StreamName window exposure through W, RegSeqRat readback through R, RegularCauchyCriterion tail testing through K, CauchyFilter source exposure through F, Real boundary exposure through B, componentwise* hsame *transport through T, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. Its rows are precisely the finite window ledger, regular rational readback, regular Cauchy criterion, Cauchy-filter source, Real boundary, transport, replay, provenance, and naming rows.

Read the source route in order. The window row W supplies the inspected stages, R reads their regular rational values, and K checks the regular Cauchy tail discipline before any filter-facing row is available.

The filter and Real rows are then confined to the same route. The source row F is reached from W, R, K, and B is read only as the terminal Real boundary of that displayed source.

Finally T, C, P, N transport, replay, source, and name the same packet. Since the carrier has no quotient stream, selected limit, host equality, countable-choice, or ambient-completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 7664.3 (RegularCauchyFilterUniversalProperty route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ *packet reads the finite route*

$$W, R, K, F \longmapsto B, T, C, P, N.$$

Thus the universal-property handoff starts with StreamName windows, RegSeqRat readback, and RegularCauchyCriterion rows, reaches the CauchyFilter source only after those rows are visible, and reaches the Real boundary only through the same displayed packet.

Proof. Start with the NameCert obligations. They order the consumer read from W to R to K before the Cauchy-filter row F can be inspected.

The terminal boundary is likewise displayed. The Real row B is not an ambient completeness theorem; it is the boundary row reached by replaying the regular Cauchy filter route.

Transport, replay, provenance, and local naming preserve that order. Hence any universal-property handoff exported by the packet factors through $W, R, K, F, B, T, C, P, N$. □

Theorem 7664.4 (RegularCauchyFilterUniversalProperty nonescape)**.** *An accepted* $\mathsf{RegularCauchyFilterUniversalP}$ *packet cannot export quotient stream equality, a selected limit representative, host equality of real names, countable choice, or ambient Real completeness. Any consumer claiming such data must supply it from a different certificate, because the local carrier stores only $W, R, K, F, B, T, C, P, N$.*

Proof. Inspect the carrier coordinates. The source data are finite StreamName windows, RegSeqRat readbacks, a RegularCauchyCriterion row, and a CauchyFilter row; the target data are a displayed Real boundary together with transport, replay, provenance, and naming.

None of these rows is a quotient-stream representative, a selected limit, a host equality proof, a choice selector, or an ambient completeness theorem. Componentwise hsame transport and displayed Cont replay preserve the listed rows rather than adding new host-side objects.

Therefore a consumer can escape the packet only by citing an additional certificate. The local $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ NameCert itself exports exactly the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ consumer that exports a filter-facing universal property while omitting one of the displayed StreamName, RegSeqRat, RegularCauchyCriterion, CauchyFilter, Real boundary, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: streamname, regseqrat, regularcauchycriterion, cauchyfilter, real: none is bijective because $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ jointly carries finite windows, regular rational readback, regular Cauchy tail testing, Cauchy-filter source exposure, terminal Real boundary, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7664.1, Definition 7664.2, Definition 7664.3, and Definition 7664.4 seed the filter-facing universal-property route over StreamName, RegSeqRat, RegularCauchyCriterion, CauchyFilter, Real boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route ordering, transport stability, replay exactness, and non-escape proofs for BEDC.Derived.RegularCauchyFilterUniversalPropertyUp.

Constructive story: A $\mathsf{RegularCauchyFilterUniversalProperty}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, RegularCauchyCriterion rows, a CauchyFilter source row, a RealUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7665

A Concrete Naming Certificate for $\mathsf{PositiveSeriesComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PositiveSeriesComparison}^{\uparrow}$ packet records the comparison test for real series with nonnegative terms as finite BEDC data. It sits after the partial-sum surface of chapter 1581, the tail criterion of chapter 2015, the majorant ledger of chapter 1774, the dyadic tolerance arithmetic of chapter 291, and the terminal real-name seal of chapter 38. The packet names a constructive comparison route; it is not a host infinite-sum theorem, a quotient of term streams, a trichotomy principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7665.1 (Positive series comparison carrier)**.** *A* $\mathsf{PositiveSeriesComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (A, B, S, M, D, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{RealSeries}^{\uparrow}$ *row whose terms are read through finite partial-sum windows.*
2. *B is the comparison* $\mathsf{RealSeries}^{\uparrow}$ *row whose displayed nonnegative terms dominate the source terms on the accepted window.*
3. *S is the* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *row attached to the source partial sums.*
4. *M is the* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *row supplied by the comparison series.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and inequality ledger used to compare the two term rows and their partial-sum tails.*
6. *R is the finite regular readback route through which the compared partial sums are inspected before any real seal is visible.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source series, comparison series, criterion row, majorant row, dyadic ledger, and readback route have been displayed.*
8. *H records componentwise* hsame *transport for the two series rows, criterion row, majorant row, dyadic ledger, readback route, real seal, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from term comparison to partial sums, from partial sums to the Cauchy criterion, from the majorant series to the tail bound, and from those rows to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{PositiveSeriesComparison}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchySeriesCriterion}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveSeriesComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, S, M, D, R, E, H, K, P, N$. It has no coordinate for a host convergence theorem, selected infinite sum, quotient stream equality, order trichotomy, countable-choice schedule, or ambient completeness principle.

Theorem 7665.2 (Positive series comparison NameCert obligations)**.** *For every accepted* $\mathsf{PositiveSeriesComparison}^{\uparrow}$ *carrier $C = (A, B, S, M, D, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PositiveSeriesComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, nonnegative comparison-window admission through A, B, D, source Cauchy-tail exposure through S, majorant-tail exposure through M, regular readback through R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7665.1. The packet displays exactly the two series rows, the criterion row, the majorant row, dyadic comparison data, readback data, real seal, transport, replay, provenance, and naming rows.

The comparison-window obligations are finite. The source and comparison series are inspected only through their displayed windows, and the dyadic ledger D is the only row that records nonnegativity and domination evidence for those windows.

The tail obligations then use the sibling criterion and majorant rows. The row S exposes the source Cauchy-tail request, while M supplies the displayed majorant schedule that bounds the requested tail.

Finally R, E, H, K, P, N read back, seal, transport, replay, source, and name the same route. Since no excluded host-side object is present in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7665.3 (Positive series comparison majorant route)**.** *Every Real- or Completion-facing comparison read from an accepted* $\mathsf{PositiveSeriesComparison}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyMajorantSeries}^{\uparrow} \longmapsto \mathsf{CauchySeriesCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a positive-term source series reaches its terminal real-facing tail conclusion only by displaying the comparison series, the dyadic domination ledger, the majorant tail schedule, the source Cauchy criterion, and the real seal.

Proof. Start with the source and comparison rows A and B. Their term windows are visible only as $\mathsf{RealSeries}^{\uparrow}$ packets, so the comparison read begins with displayed partial-sum and term-window data.

Next read the dyadic ledger D. This row records the nonnegative comparison evidence and the tolerance budget used to compare the source term windows against the majorant windows.

Apply the majorant and criterion rows M and S. The majorant row supplies the tail bound, and the criterion row consumes that bound as the source Cauchy-tail route before the real-facing seal is available.

Finally inspect R, E, H, K, P, N. These rows read back, seal, transport, replay, source, and name the same finite route, so the comparison test exports no host convergence theorem, selected infinite sum, quotient stream equality, trichotomy, countable choice, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{PositiveSeriesComparison}^{\uparrow}$ read displays two RealSeries rows, one CauchySeriesCriterion row, one CauchyMajorantSeries row, one DyadicRatCore comparison ledger, one regular readback route, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a comparison conclusion is consumed.

Independence witness: realseries, cauchy-series-criterion, cauchy-majorant-series, dyadicrat-core, real: $\mathsf{PositiveSeriesComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds positive-term domination, majorant tail control, source Cauchy criterion data, dyadic tolerance evidence, and real-facing sealing in one comparison packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveSeriesComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveSeriesComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7665.1, Definition 7665.2, and Definition 7665.3 seed the finite positive-series comparison route over source series rows, comparison series rows, Cauchy criterion data, majorant tail schedules, dyadic tolerance evidence, readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveSeriesComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host infinite-sum theorems, quotient stream equality, selected sums, order trichotomy, countable choice, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, nonnegative comparison-window admission, majorant-tail exposure, source Cauchy-tail exposure, real-seal nonescape, transport stability, replay stability, and local naming for BEDC.Derived.PositiveSeriesComparisonUp.

Constructive story: A $\mathsf{PositiveSeriesComparison}^{\uparrow}$ packet is generated from source and comparison RealSeries rows, CauchySeriesCriterion data, CauchyMajorantSeries data, a DyadicRatCore comparison ledger, a regular readback route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7666

A Concrete Naming Certificate for HyperbolicDirichletDomain$^{\uparrow}$

Following Poincare, Klein, and classical Fuchsian fundamental-domain theory, the HyperbolicDirichletDomain$^{\uparrow}$ packet records a finite Dirichlet-domain surface inside displayed Poincare disk data. It sits after the metric-read surface of chapter 6209, the action-word surface of chapter 7282, the visual-boundary packet of chapter 4181, and the hyperbolic metric route of chapter 3474. The packet names finite side-pairing, metric-read, boundary-refusal, and local NameCert rows; it does not construct a quotient surface, a selected tessellation, a global moduli object, or host-level discreteness theorem.

Definition 7666.1 (Hyperbolic Dirichlet domain carrier)**.** *A* HyperbolicDirichletDomain$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, G, S, M, B, V, H, C, P, N)$$

with the following displayed rows:

1. *O is the displayed basepoint and disk-coordinate row inside the* PoincareDiskMetric$^{\uparrow}$ *surface.*
2. *G is the finite* FuchsianGroupAction$^{\uparrow}$ *action-word row whose inspected translates determine the displayed side candidates.*
3. *S is the side-pairing ledger, recording the finite bisector or boundary-side rows, their paired action words, and the local incidence data read from the disk model.*
4. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row used to compare the basepoint with the inspected translates along the displayed side candidates.*
5. *B is the boundary-refusal row: ideal endpoints, visual-boundary contacts, and exterior reads are exposed only as finite refusal or boundary-facing data, not as a completed quotient boundary.*
6. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *row available to consumers that inspect visible boundary arcs of the displayed sides.*
7. *H records componentwise* hsame *transport for basepoint, action-word, side-pairing, metric-read, boundary-refusal, visual-boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from basepoint inspection through action words, metric reads, side-pairing checks, boundary refusal, visual-boundary reads, provenance, and local naming.*
9. *P is the* Pkg *provenance over* HyperbolicDirichletDomain$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, FuchsianGroupAction$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicDirichletDomain}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $O, G, S, M, B, V, H, C, P, N$. It has no coordinate for a quotient hyperbolic surface, selected global tessellation, host discreteness proof, moduli point, or completed ideal-boundary equality.

Theorem 7666.2 (Hyperbolic Dirichlet domain NameCert obligations)**.** *For every accepted* HyperbolicDirichletDom *carrier $D = (O, G, S, M, B, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicDirichletDomain}^{\uparrow}}$ *surface consists exactly of basepoint admission through O, finite action-word admission through G, side-pairing ledger exactness through S, hyperbolic metric-read exactness through M, boundary-refusal exposure through B, visual-boundary exposure through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7666.1. The packet displays exactly the basepoint, finite action-word, side-pairing, metric-read, boundary-refusal, visual-boundary, transport, replay, provenance, and naming rows.

The geometric obligations are finite. The basepoint row is read through the Poincare disk metric surface, the action words through the Fuchsian action surface, and the side-pairing ledger through the displayed bisector and incidence rows.

The metric and boundary rows are downstream of those displayed data. The row M gives the hyperbolic metric comparisons, B records refusal of quotient and ideal-boundary data, and V is the only visual-boundary route available to boundary-facing consumers.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded global objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7666.3 (Hyperbolic Dirichlet domain side route)**.** *Every side-pairing or boundary-facing read from an accepted* $\mathsf{HyperbolicDirichletDomain}^{\uparrow}$ *carrier factors through*

$$\mathsf{PoincareDiskMetric}^{\uparrow} \longmapsto \mathsf{FuchsianGroupAction}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualBoundary}^{\uparrow}.$$

Thus the Dirichlet-domain packet exposes finite basepoint, action-word, side-pairing, metric-read, boundary-refusal, and visual-boundary rows before any consumer can inspect a domain side.

Proof. Start with the basepoint row O. It is read through the Poincare disk metric packet, so the domain begins as displayed disk-coordinate and metric-read data rather than as a completed hyperbolic plane.

Next inspect the action row G and side ledger S. The finite Fuchsian action words determine which displayed translates and side candidates are available to the packet.

Read the metric and boundary rows M, B, V. The metric row compares the displayed basepoint with inspected translates, while B and V restrict boundary-facing data to refusal and visual-boundary rows.

The remaining rows H, C, P, N preserve, replay, source, and name the same route. No quotient surface, selected tessellation, host discreteness proof, moduli point, or completed boundary equality can be read from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicDirichletDomain}^{\uparrow}$ side read displays a PoincareDiskMetric basepoint row, a FuchsianGroupAction word row, a side-pairing ledger, a HyperbolicMetric comparison row, a boundary-refusal row, a HyperbolicVisualBoundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: poincare-disk-metric, fuchsian-group-action, hyperbolic-metric, hyperbolic-visual-boundary: $\mathsf{HyperbolicDirichletDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly

binds a basepoint, finite action words, side-pairing ledger, metric comparisons, boundary refusal, and visual-boundary exposure as one finite domain packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicDirichletDomain}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicDirichletDomain}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7666.1, Definition 7666.2, and Definition 7666.3 seed the finite Dirichlet-domain packet over Poincare disk metric rows, Fuchsian action words, side-pairing data, hyperbolic metric reads, boundary refusal, visual-boundary exposure, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicDirichletDomain}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient hyperbolic surfaces, selected tessellations, host discreteness theorems, global moduli objects, bridge checking, or completed ideal-boundary equality.

Upgrade path: Obligation closure requires checked basepoint admission, action-word admission, side-pairing exactness, metric-read exactness, boundary-refusal exposure, visual-boundary exposure, transport stability, replay stability, and local naming for BEDC.Derived.HyperbolicDirichletDomainUp.

Constructive story: A $\mathsf{HyperbolicDirichletDomain}^{\uparrow}$ packet is generated from a PoincareDiskMetric basepoint row, a FuchsianGroupAction word row, a finite side-pairing ledger, HyperbolicMetric comparisons, boundary-refusal data, a HyperbolicVisualBoundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7667

A Concrete Naming Certificate for $\mathsf{LocatedCauchySubnet}^{\uparrow}$

Following Kelley and Moore–Smith subnet terminology together with Bishop–Bridges located Cauchy completion, the $\mathsf{LocatedCauchySubnet}^{\uparrow}$ packet records the finite route by which a cofinal subnet of a Cauchy source remains located before the Real seal is inspected. It consumes the Moore–Smith subnet surface of chapter 5098, the Cauchy restriction surface of chapter 1592, the located-distance request of chapter 2043, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet names a finite completion-neighbourhood route; it is not a selected limit, quotient net, countable-choice schedule, host equality statement, or ambient Real completeness theorem.

Definition 7667.1 (Located Cauchy subnet carrier)**.** *A* $\mathsf{LocatedCauchySubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MooreSmithSubnet}^{\uparrow}$ *cofinal-window row, recording the source index window, subnet index window, and displayed cofinal tail route.*
2. *S is the* $\mathsf{CauchySubnet}^{\uparrow}$ *row, requiring the source Cauchy request to pass through the same cofinal subnet ledger before any completion-facing handoff is read.*
3. *Q is the located-distance request row, read through* $\mathsf{LocatedSpace}^{\uparrow}$ *and carrying the finite source, target, and tolerance request.*
4. *D is the dyadic tolerance ledger attached to the located request and the Cauchy comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the subnet windows and located-distance comparison.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after M, S, Q, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the subnet window, Cauchy request, located request, tolerance ledger, regular readback, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of every accepted read through*

$$M, S, Q, D, R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchySubnet}^{\uparrow}$, $\mathsf{MooreSmithSubnet}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchySubnet}^{\uparrow}}$ *row.*

The carrier exports no selected subnet representative, quotient net, countable-choice selector, host equality row, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness object.*

Theorem 7667.2 (Located Cauchy subnet NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchySubnet}^{\uparrow}$ *carrier* $L = (M, S, Q, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchySubnet}^{\uparrow}}$ *surface consists exactly of the subnet cofinal window, Cauchy request, located-distance request, RegSeqRat readback, Real seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows displayed in Definition 7667.1.*

Proof. Project the carrier. The cofinal-window row M is the only subnet route, and the Cauchy row S is read through that same route before any completion-facing consumer can inspect the packet.

The located-distance request is then exposed by Q together with the tolerance ledger D. The regular rational readback R is the only row translating the displayed subnet windows and located comparison into finite rational data.

Only after those rows have been replayed may the packet reach the Real seal E. Componentwise transport, continuation replay, provenance, and local naming are confined to H, C, P, N, so the listed rows exhaust the NameCert surface and leave no coordinate for selected limits, quotient nets, choice, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchySubnet}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchySubnet}^{\uparrow}}$, seed)

Scope closed: Definition 7667.1 and Definition 7667.2 seed the finite located Cauchy subnet route from cofinal Moore–Smith windows through Cauchy restriction, located-distance request, regular readback, and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCauchySubnet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient nets, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-window exactness, Cauchy request stability, located-distance request stability, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchySubnetUp.

Constructive story: A $\mathsf{LocatedCauchySubnet}^{\uparrow}$ packet is generated from a Moore–Smith subnet window, a CauchySubnet request, a LocatedSpace distance request, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7668

A Concrete Naming Certificate for $\mathsf{MedianDepthSealingCorner}^{\uparrow}$

The $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ packet realizes the two-layer median closure and unique ACG sealing corner of Definition 9462.1329 as finite NameCert data. It consumes the DNA window-six boundary tile packet of chapter 3393, the trigger-blocker route of chapter 3801, the stop/Trp face deficiency packet of chapter 4090, and the Boolean trigger lattice of chapter 3704. The packet records a median-layer route and a unique-corner seal; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unbounded support-closure oracle.

Definition 7668.1 (Median depth sealing corner carrier)**.** *A* $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, U, F, W, H, C, P, N)$$

with the following displayed rows:

1. *B is the DNA window-six boundary-tile source row.*
2. *L is the median-layer ledger, displaying the active reassignment support, the first median layer, and the second median layer.*
3. *U is the unique sealing-corner witness row, naming* ACG *as the sole strict second-layer median point.*
4. *F is the f-vector transition row recording the passage from the punctured main* WNR *4-cube to the sealed cube.*
5. *W is the witness row displaying the median witnesses for the first layer and the* ACG *second-layer witness before F is read.*
6. *H records componentwise* hsame *transport for the boundary tile, median-layer, unique-corner, f-vector, witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through*

$$B, L, U, W \longmapsto F.$$

8. *P is the* Pkg *provenance over* $\mathsf{MedianDepthSealingCorner}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{TriggerBlockerDuality}^{\uparrow}$, ***??***, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealingCorner}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $B, L, U, F, W, H, C, P, N$ and has no coordinate for hidden codon semantics, stochastic biology, unrestricted median search, or host cubical-complex equality.

Theorem 7668.2 (Median depth sealing corner NameCert obligations)**.** *For every accepted* MedianDepthSealingCor *carrier $K = (B, L, U, F, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealingCorner}^{\uparrow}}$ *surface consists exactly of finite median-layer admission, unique* ACG *corner exposure, f-vector transition exposure, witness-row exposure, componentwise transport, continuation replay, provenance, and local naming rows.*

Proof. Project the carrier. The boundary-tile row B is the source, while L records the two median layers and U names the unique strict second-layer corner.

The witness row W must be read before the f-vector transition F. Thus an accepted read displays both median layers and the ACG witness before it consumes the cubical transition from the punctured main WNR cell to the sealed cell.

The structural rows H, C, P, N transport, replay, source, and name only this finite packet. Since no row stores hidden codon semantics, stochastic dynamics, an unrestricted median oracle, or host cubical equality, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ read displays the two median layers and the ACG witness row before the f-vector transition row is consumed.

Independence witness: dna-window-six-boundary-tile, trigger-blocker-duality, stop-trp-face-rank-deficiency, boolean-trigger-lattice: none is bijective with $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ because this packet jointly carries the two median layers, the unique ACG corner, the f-vector transition, witness rows, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit Nontrivial $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ and FieldFaithful MedianDepthSe TasteGate instances so that $B, L, U, F, W, H, C, P, N$ remain separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianDepthSealingCorner}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MedianDepthSealingCorner}^{\uparrow}}$, seed)

Scope closed: Definition 7668.1 and Definition 7668.2 seed the finite median-layer and unique sealing-corner packet for the DNA window-six boundary-tile route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MedianDepthSealingCorner}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological mechanism, stochastic fitness, quotient codon semantics, unrestricted median search, host cubical equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, median-layer exposure, unique-corner exposure, f-vector transition exposure, witness exposure, Nontrivial MedianDepthSealingCornerUp, and FieldFaithful MedianDepthSealingCornerUp for BEDC.Derived.MedianDepthSealingCornerUp.

Constructive story: A $\mathsf{MedianDepthSealingCorner}^{\uparrow}$ packet is generated from a DNA boundary-tile source, median-layer rows, a unique ACG sealing-corner row, f-vector transition data, witness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7669

A Concrete Naming Certificate for $\mathsf{EulerConstant}^{\uparrow}$

Following Euler and Bishop–Bridges constructive analysis, the $\mathsf{EulerConstant}^{\uparrow}$ packet records the Euler–Mascheroni constant as a finite harmonic-log comparison route rather than as a host real constant. It sits after the harmonic-window divergence surface of chapter 3711, the logarithm comparison packet of chapter 2315, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite comparison data for $H_n - \log(n)$ windows; it is not a host limit, quotient real equality, selected convergence oracle, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7669.1 (Euler constant carrier)**.** *An* $\mathsf{EulerConstant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (H, L, D, R, E, T, C, P, N)$$

with the following displayed rows:

1. *H is the harmonic partial-sum window ledger. It records the inspected finite indices, the displayed partial sums $1 + \cdots + 1/n$, and the Oresme-facing comparison evidence inherited from* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$*.*
2. *L is the* $\mathsf{RealLogarithm}^{\uparrow}$ *comparison row for the same displayed indices, read as $\log(n)$ only after the harmonic window has been fixed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the positive dyadic budget used to compare the displayed harmonic-log difference windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows obtained from H, L, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the Euler-constant candidate after the harmonic, logarithm, dyadic, and regular rational rows have been displayed.*
6. *T records componentwise* hsame *transport for the harmonic windows, logarithm comparisons, dyadic budgets, readback rows, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$H, L, D, R \longmapsto E, T, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{EulerConstant}^{\uparrow}$*,* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$*,* $\mathsf{RealLogarithm}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{EulerConstant}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $H, L, D, R, E, T, C, P, N$. It has no coordinate for an ambient real limit, a quotient equality of real names, countable choice, a selected convergence oracle, or host-side logarithm completeness.

Theorem 7669.2 (Euler constant NameCert obligations)**.** *For every accepted* $\mathsf{EulerConstant}^{\uparrow}$ *carrier $G = (H, L, D, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EulerConstant}^{\uparrow}}$ *surface consists exactly of harmonic partial-sum windows through H,* $\mathsf{RealLogarithm}^{\uparrow}$ *comparisons through L, positive dyadic comparison budgets through D, regular rational readback through R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal through E, componentwise* hsame *transport through T, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. Its coordinates are precisely the harmonic window, logarithm comparison, dyadic tolerance, regular readback, real seal, transport, replay, provenance, and naming rows.

The comparison route starts with finite harmonic data. The row H exposes only inspected partial sums and the Oresme-facing harmonic comparison surface; it does not provide a completed infinite series.

The logarithm and tolerance rows are then read against the same windows. The row L supplies the $\mathsf{RealLogarithm}^{\uparrow}$ comparison for the displayed index, and D gives the dyadic budget before R can read the harmonic-log difference as regular rational data.

Finally E, T, C, P, N seal, transport, replay, source, and name the same route. Since the carrier stores no ambient limit selector, quotient real equality, countable-choice schedule, or host completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 7669.3 (Euler constant regular Cauchy handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{EulerConstant}^{\uparrow}$ *carrier reads the Euler-constant candidate only through the finite route*

$$\mathsf{OresmeHarmonicDivergence}^{\uparrow},\ \mathsf{RealLogarithm}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the $\mathsf{Real}^{\uparrow}$ *seal E is available only after the harmonic window H, logarithm comparison L, dyadic tolerance row D, and regular rational readback row R have been displayed through T, C, P, N.*

Proof. Start from an accepted carrier $G = (H, L, D, R, E, T, C, P, N)$. A consumer can read the real seal only at the row E.

The row E is downstream of the finite harmonic-log comparison route. The consumer must first inspect the harmonic window H, compare the same index through L, attach a positive dyadic budget through D, and read the resulting difference through R.

Transport and replay preserve this order. The rows T and C identify and replay the displayed route, while P and N record provenance and local naming. Hence the handoff reaches $\mathsf{Real}^{\uparrow}$ through regular Cauchy readback data rather than through an ambient completeness theorem. □

Theorem 7669.4 (Euler constant nonescape)**.** *An accepted* $\mathsf{EulerConstant}^{\uparrow}$ *packet cannot export a host-level Euler–Mascheroni constant, quotient equality of real names, countable choice, a selected convergence oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer claiming such data must supply it from a different certificate, because the local carrier stores only $H, L, D, R, E, T, C, P, N$.*

Proof. Inspect the carrier coordinates. The source data are finite harmonic windows, logarithm comparisons, dyadic budgets, and regular rational readbacks; the target data are a displayed real seal together with transport, replay, provenance, and naming.

None of these rows is a host real constant, quotient representative, choice selector, convergence oracle, or ambient completeness theorem. Componentwise hsame transport and displayed Cont replay preserve the listed rows rather than adding new host-side objects.

Therefore a consumer can obtain such data only by citing an additional certificate. The local $\mathsf{EulerConstant}^{\uparrow}$ NameCert itself exports exactly the displayed harmonic-log route. □

Falsifiable prediction: Every accepted $\mathsf{EulerConstant}^{\uparrow}$ consumer that exports an Euler-constant real while omitting one of the displayed harmonic-window, RealLogarithm, DyadicRatCore, RegSeqRat, Real seal, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: harmonic, logarithm, dyadic, regseqrat, real: none is bijective because $\mathsf{EulerConstant}^{\uparrow}$ jointly carries harmonic partial-sum windows, RealLogarithm comparisons, DyadicRatCore budgets, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerConstant}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerConstant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7669.1, Definition 7669.2, Definition 7669.3, and Definition 7669.4 seed the finite harmonic-log comparison route over Oresme harmonic windows, RealLogarithm comparisons, DyadicRatCore budgets, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerConstant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host-level Euler–Mascheroni constant, quotient real equality, countable choice, selected convergence oracles, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, harmonic-log route ordering, dyadic tolerance stability, regular Cauchy readback, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.EulerConstantUp.

Constructive story: An $\mathsf{EulerConstant}^{\uparrow}$ packet is generated from harmonic partial-sum windows, RealLogarithm comparison rows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7670

A Concrete Naming Certificate for $\mathsf{AlternatingHarmonic}^{\uparrow}$

Following Hardy's summability treatment of alternating series and Bishop–Bridges constructive analysis, the $\mathsf{AlternatingHarmonic}^{\uparrow}$ packet names the finite endpoint route for the alternating reciprocal series before any ambient real completeness principle is read. It sits after the Leibniz-series certificate of chapter 3723, the regular rational readback of chapter 301, the finite stream naming surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet records the known alternating harmonic endpoint as finite NameCert data; it is not a host analytic theorem, a quotient of series streams, a selected real representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7670.1 (Alternating harmonic carrier)**.** *An* $\mathsf{AlternatingHarmonic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the alternating reciprocal source-window row, listing the inspected signs, reciprocal denominators, partial-sum windows, and tail requests.*
2. *L is the* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *row certifying monotone reciprocal magnitudes and alternating tail control for the inspected windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder and tolerance ledger, recording the dyadic budget attached to each admitted tail request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite partial sums and dyadic remainders exposed by S, L, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, Leibniz row, dyadic ledger, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the source, Leibniz, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, L, D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{AlternatingHarmonic}^{\uparrow}$, $\mathsf{LeibnizSeriesTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{AlternatingHarmonic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, D, R, E, H, C, P, N$. It has no coordinate for host convergence, a selected value of $\log 2$, quotient equality of streams, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.AlternatingHarmonicUp.AlternatingHarmonicTasteGate single carrier alignment`

Theorem 7670.2 (Alternating harmonic NameCert obligations)**.** *For every accepted* $\mathsf{AlternatingHarmonic}^{\uparrow}$ *carrier $A = (S, L, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlternatingHarmonic}^{\uparrow}}$ *surface consists exactly of alternating reciprocal source-window admission, LeibnizSeriesTest row exposure, dyadic remainder and tolerance exactness, RegSeqRat readback, Real seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7670.1. The source row S supplies the inspected alternating reciprocal windows, while L records the Leibniz monotonicity and alternating-tail discipline on those windows.

The dyadic ledger D is then the only tolerance source available to the packet. Regular rational readback R consumes S, L, D, and the Real seal E is downstream from that readback.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7670.3 (Alternating harmonic Leibniz real seal)**.** *Every Real-facing consumer of an accepted* $\mathsf{AlternatingHarmonic}^{\uparrow}$ *packet factors through*

$$S, L, D \longmapsto R \longmapsto E.$$

Thus the alternating reciprocal source windows are first checked by the $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *row, then budgeted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then read through* $\mathsf{RegSeqRat}^{\uparrow}$*, and only then sealed as a* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. Start with a Real-facing request. The carrier gives no direct Real coordinate except E, and E is named as the seal reached after the source, test, ledger, and readback rows.

The Leibniz row L verifies the alternating-tail discipline on the same windows S that feed the dyadic budget D. The readback row R therefore receives exactly the finite partial-sum and remainder data displayed by the packet.

Transport and replay preserve that order through H and C, while P, N source and name it. Hence any Real-facing read must pass through S, L, D, R before it can consume E. □

Theorem 7670.4 (Alternating harmonic ledger nonescape)**.** *An accepted* $\mathsf{AlternatingHarmonic}^{\uparrow}$ *carrier cannot export host convergence, a selected value for the infinite sum, quotient equality of series streams, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any such consumer must cite a separate certificate, because the local carrier stores only $S, L, D, R, E, H, C, P, N$.*

Proof. Inspect the carrier coordinates. The analytic data are finite alternating reciprocal windows, a Leibniz row, a dyadic tolerance ledger, regular rational readback, and a terminal Real seal.

The structural data are componentwise transport, displayed replay, provenance, and local naming. None of these rows stores a host convergence theorem, a chosen real representative, quotient-stream equality, a choice selector, or a completeness principle.

Therefore every accepted read remains inside the finite ledger route. Any stronger assertion must be supplied by another named packet rather than by $\mathsf{AlternatingHarmonic}^{\uparrow}$ itself. □

Falsifiable prediction: Every proposed $\mathsf{AlternatingHarmonic}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ while omitting the alternating reciprocal source windows, LeibnizSeriesTest row, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: leibniz-series-test, streamname, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{AlternatingHarmonic}^{\uparrow}$ because the packet jointly carries alternating reciprocal source windows, Leibniz tail control, dyadic remainders, regular rational readback, terminal sealing, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{AlternatingHarmonic}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlternatingHarmonic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7670.1, Definition 7670.2, Definition 7670.3, and Definition 7670.4 seed the finite alternating-series endpoint packet over Leibniz, dyadic tolerance, regular readback, Real sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlternatingHarmonic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convergence, the equality of the endpoint with $\log 2$, quotient equality of series streams, countable choice, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Leibniz row exposure, dyadic ledger exactness, regular readback, Real-seal routing, transport, replay, provenance, and non-escape proofs for BEDC.Derived.AlternatingHarmonicUp.

Constructive story: An $\mathsf{AlternatingHarmonic}^{\uparrow}$ packet is generated from alternating reciprocal source windows, a LeibnizSeriesTest row, a DyadicRatCore remainder and tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7671

A Concrete Naming Certificate for $\mathsf{UnconditionalBasis}^{\uparrow}$

Following Banach-space basis theory as in Lindenstrauss–Tzafriri and Wojtaszczyk, the $\mathsf{UnconditionalBasis}^{\uparrow}$ packet records the finite BEDC naming surface for basis coordinates whose inspected partial sums remain stable under displayed sign changes. It refines the coordinate route of chapter 3669, sits beside the frame-bound stability of chapter 7602, and uses the finite orthonormal comparison discipline of chapter 4758. The packet names only finite sign-window stability data; it is not a full Banach-space basis theorem, not countable choice, not a selected infinite series, and not quotient equality.

Definition 7671.1 (Unconditional basis carrier)**.** *An* $\mathsf{UnconditionalBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, V, N, W, S, K, Q, \Gamma, H, C, P, Z)$$

with the following displayed rows:

1. *B is the* $\mathsf{SchauderBasis}^{\uparrow}$ *source row, exposing the basis-vector stream, coordinate readback, finite partial sums, tail estimates, and reconstruction route consumed by the packet.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row over the same endpoint, recording the vector operations and finite linear-combination syntax used by signed sums.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row naming norm evaluations, triangle ledgers, and comparison rows for the displayed signed partial sums.*
4. *W is the finite window row, listing the inspected index set, coefficient names, truncation bound, and every sign pattern $\sigma_i \in \{-1, 1\}$ admitted inside the packet.*
5. *S is the signed partial-sum row, recording the displayed sums $\sum_{i \in W} \sigma_i c_i b_i$ and their unsigned comparison sums.*
6. *K is the unconditional-stability budget row, displaying the finite bound or modulus by which every signed sum in S remains controlled by the same coordinate window.*
7. *Q is the coordinate reconstruction ledger, requiring every consumer read to pass through the Schauder source row, the finite sign window, the signed-sum row, and the stability budget before reaching a named vector endpoint.*

8. Γ *is the local classifier, comparing accepted packets only by the shared Schauder source, vector and norm source rows, sign-window rows, signed sums, stability budget, reconstruction ledger, and structural rows.*
9. H *records componentwise* hsame *transport for* $B, V, N, W, S, K, Q, \Gamma, C, P, Z$.
10. C *records displayed* Cont *replay from a coordinate request through the same finite sign-window route.*
11. P *is the* Pkg *provenance over* $\mathsf{UnconditionalBasis}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. Z *is the local* $\mathsf{NameCert}_{\mathsf{UnconditionalBasis}^{\uparrow}}$ *row binding the packet.*

The carrier has no coordinate for a host unconditional convergence theorem, countable-choice schedule, selected infinite series, quotient vector equality, or ambient Banach-space completion proof.

Theorem 7671.2 (Unconditional basis NameCert obligations)**.** *For every accepted* $\mathsf{UnconditionalBasis}^{\uparrow}$ *carrier* $U = (B, V, N, W, S, K, Q, \Gamma, H, C, P, Z)$*, the local* $\mathsf{NameCert}_{\mathsf{UnconditionalBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, Schauder source compatibility, vector and norm source compatibility, finite sign-window exposure, signed partial-sum exposure, unconditional-stability budget display, coordinate reconstruction routing, classifier stability, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A consumer may inspect signed coordinates only along the displayed route*

$$B, V, N, W \longrightarrow S, K \longrightarrow Q, \Gamma \longrightarrow H, C, P, Z.$$

It receives no host unconditional convergence theorem, no countable-choice selector, no selected infinite synthesis series, no quotient vector equality, and no ambient completion coordinate.

Proof. Carrier admission is projection from Definition 7671.1. The rows B, V, N expose the Schauder source, vector operations, and norm comparisons, while W exposes the only finite sign windows on which signed coefficient reads may be requested.

The signed partial-sum row S and the stability budget row K are displayed beside the same window row. Thus a sign-changed read can compare only the inspected signed sums with their named coordinate window, and it cannot appeal to an external unconditional convergence theorem.

The reconstruction row Q is downstream of B, V, N, W, S, K. The classifier Γ compares packets by the same displayed rows, and the structural rows H, C, P, Z preserve that route under transport, replay, provenance, and local naming.

Since the carrier contains no host unconditional convergence coordinate, countable-choice selector, selected infinite synthesis series, quotient vector equality, or ambient completion proof, the listed rows exhaust the local naming surface. □

Theorem 7671.3 (Unconditional basis finite sign-window stability)**.** *Every coordinate consumer exported by an accepted* $\mathsf{UnconditionalBasis}^{\uparrow}$ *carrier reads sign stability through a finite inspected window. For each displayed sign pattern* $\sigma_i \in \{-1, 1\}$ *in* W*, the signed partial sum* $\sum_{i \in W} \sigma_i c_i b_i$ *is admitted only through the signed-sum row* S *and the stability budget row* K*, after the* $\mathsf{SchauderBasis}^{\uparrow}$ *source row* B*, the vector row* V*, and the norm row* N *have been exposed. The exported read is therefore a finite-window stability certificate, not a Banach-space theorem about all signs, not an infinite series selector, not countable choice, and not quotient equality.*

Proof. Start with an accepted carrier $U = (B, V, N, W, S, K, Q, \Gamma, H, C, P, Z)$. The carrier definition places every sign pattern in the finite window row W and every signed partial sum in S, so a consumer has no source for signs outside the inspected window.

Use Definition 7671.2. It forces the consumer to read the Schauder source and the vector and norm rows before any signed sum can be compared, then to pass through the budget row K for the displayed stability check.

The reconstruction ledger Q and classifier Γ keep the read on the same finite surface. The transport, replay, provenance, and local naming rows preserve that surface, while the carrier omits host unconditional convergence, countable choice, selected infinite synthesis, quotient equality, and ambient completion. Hence the exported stability statement is exactly the finite sign-window certificate named by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnconditionalBasis}^{\uparrow}}$**.**

Constructive story: An $\mathsf{UnconditionalBasis}^{\uparrow}$ packet is generated from a SchauderBasisUp source row, VecSpaceUp and NormUp source rows, finite sign-window rows, signed partial-sum rows, a stability budget, a coordinate reconstruction ledger, classifier data, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnconditionalBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7671.1, Definition 7671.2, and Definition 7671.3 seed the finite sign-window stability packet over $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnconditionalBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a full Banach-space basis theorem, unconditional convergence over all signs, countable choice, selected infinite series, quotient vector equality, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Schauder source compatibility, finite sign-window exposure, signed partial-sum exposure, stability-budget display, reconstruction routing, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.UnconditionalBasisUp.

Chapter 7672

A Concrete Naming Certificate for $\mathsf{ForceLikeRelation}^{\uparrow}$

Following Newtonian force language and Cartan-style relation discipline, the $\mathsf{ForceLikeRelation}^{\uparrow}$ packet realizes the finite inter-Hist classifier described in Definition 9416.15 without turning force into a source-free primitive. It sits beside the relation-certificate source of chapter 9416, the MetaCIC relation surface of chapter 7661, the inter-Hist locality discipline of chapter 9419, and the records-side primitives BHist, hsame, Cont, and NameCert. The packet names finite change rows, classifier rows, preservation rows, and failure boundaries; it is not empirical physical force itself, not a hidden cause, not a global field primitive, and not a standard-physics bridge.

Definition 7672.1 (Force-like relation carrier)**.** *A* $\mathsf{ForceLikeRelation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, K, V, W, M, E, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite inter-*Hist *change row, listing source histories, observed changes, and local measurement or continuation requests.*
2. *K is the classifier row assigning admitted changes to the same-force-pattern surface.*
3. *V is the preserved-structure row, recording the relation data stable under the admitted classifier.*
4. *W is the not-preserved row, recording empirical constants, numerical laws, global-field readings, and physical semantics not carried by the records-side packet.*
5. *M is the layer map from source change rows to relation-certificate rows.*
6. *E is the exactness row stating when classifier sameness reflects the displayed inter-*Hist *relation rather than object identity.*
7. *B is the failure-boundary row naming the conditions under which the relation weakens, refuses, or asks for a separate standard bridge.*
8. *H records componentwise* hsame *transport for the change, classifier, preserved, not-preserved, layer, exactness, boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite relation rows.*
10. *P is the* Pkg *provenance over* $\mathsf{ForceLikeRelation}^{\uparrow}$*,* $\mathsf{LayeredRelationCertificate}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{ForceLikeRelation}^{\uparrow}}$ *row.*

The classifier admits force-like language only through $I, K, V, W, M, E, B, H, C, P, N$. It has no coordinate for empirical physical force, a hidden causal substance, global field identity, host physics, or an unrecorded bridge.

Theorem 7672.2 (Force-like relation NameCert obligations)**.** *For every accepted* $\mathsf{ForceLikeRelation}^{\uparrow}$ *carrier $F = (I, K, V, W, M, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ForceLikeRelation}^{\uparrow}}$ *surface consists exactly of inter-*Hist *change-row admission, classifier exposure, preserved-row exposure, not-preserved-row exposure, layer-map exposure, exactness exposure, failure-boundary exposure, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7672.1. The source data are the finite inter-Hist change rows I, and the admissible force-like classifier is K.

The certificate then reads preservation and refusal together. The preserved row V is bounded by the not-preserved row W, while the layer map M and exactness row E are bounded by the failure row B.

Transport, replay, provenance, and naming are exhausted by H, C, P, N. Thus the local certificate has exactly the displayed relation rows and no hidden physics coordinate. □

Theorem 7672.3 (Force-like relation MetaCIC boundary)**.** *Every MetaCIC or relation-certificate consumer of an accepted* $\mathsf{ForceLikeRelation}^{\uparrow}$ *packet reads force-like language only through*

$$I, K, V, W, M, E, B, H, C, P, N.$$

*The consumer may reuse the packet as a finite inter-*Hist *relation certificate, but it may not infer empirical force laws, global field identity, hidden causal substance, object identity between histories, or bridge-checked standard physics.*

Proof. Start with the relation route. The inter-Hist change row I and classifier row K supply the only source from which force-like language can be read.

The preserved row V is paired with the not-preserved row W, so no empirical or global-field content can be silently carried across the relation. The layer map M and exactness row E are likewise constrained by the failure-boundary row B.

The rows H, C, P, N preserve, replay, source, and name the same finite packet. Since the carrier lacks a standard bridge or host-physics coordinate, the consumer remains at relation-certificate strength. □

Falsifiable prediction: Every accepted $\mathsf{ForceLikeRelation}^{\uparrow}$ packet without a displayed not-preserved row or failure-boundary row fails as force-like relation evidence and must be routed to refusal rather than to standard physical force.

Independence witness: layered-relation-certificate, inter-hist-locality, bhist, hsame, cont: none is bijective to $\mathsf{ForceLikeRelation}^{\uparrow}$ because the packet jointly carries inter-Hist change rows, a force-like classifier, preserved and not-preserved rows, a layer map, exactness, failure boundary, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{ForceLikeRelation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForceLikeRelation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7672.1, Definition 7672.2, and Definition 7672.3 seed the finite inter-Hist relation-certificate packet that separates force-like classifier evidence from empirical force and bridge-checked physics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForceLikeRelation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close empirical physical force, global field identity, hidden causal substance, object identity between histories, standard-physics bridge checking, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, preserved and not-preserved row pairing, layer-map exactness, failure-boundary routing, transport, replay, provenance, and non-escape proofs for BEDC.Derived.ForceLikeRelationUp.

Constructive story: A ForceLikeRelation$^{\uparrow}$ packet is generated from finite inter-Hist change rows, classifier rows, preserved and not-preserved rows, a layer map, exactness row, failure-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7673

A Concrete Naming Certificate for $\mathsf{FiniteCardinal}^{\uparrow}$

Following Dedekind and Cantor on finite equipotence, the $\mathsf{FiniteCardinal}^{\uparrow}$ packet records cardinal comparison only through displayed finite bijection and counting ledgers. It sits after the membership discipline of chapter 3668, the natural-number endpoint of chapter 28, the Kuratowski finite-set presentation of chapter 7157, and the finite-net consumer route of chapter 7638. The packet names finite cardinality without importing unrestricted cardinal arithmetic, quotient sets, chosen representatives, powersets, or choice.

Definition 7673.1 (Finite cardinal carrier)**.** *A* $\mathsf{FiniteCardinal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, n, L, R, B, Q, \Gamma, H, C, P, N)$$

with the following displayed rows:

1. *S is the source finite-object row, read through* $\mathsf{SetLike}^{\uparrow}$ *and* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$*, exposing membership, finite generation, and the inspected element names.*
2. *T is the target initial-segment row over* $\mathsf{Nat}^{\uparrow}$*, naming the displayed positions $0, \ldots, n-1$ as finite endpoints rather than as a host ordinal quotient.*
3. *n is the cardinality readback row, a named natural-number endpoint whose only public meaning is the length of the displayed target segment T.*
4. *L is the forward ledger, listing for each displayed source element its target position in T.*
5. *R is the reverse ledger, listing for each displayed target position the source element in S that occupies it.*
6. *B is the bijection ledger, recording the two finite round-trip checks $R(L(s)) = s$ for displayed source entries and $L(R(i)) = i$ for displayed target entries.*
7. *Q is the cardinality-use ledger, forcing every consumer of the number n to pass through S, T, L, R, B before citing the cardinality row.*
8. *Γ is the local classifier, comparing accepted packets only by source membership rows, target segment rows, the named natural endpoint, the two ledgers, the round-trip checks, and the consumer route.*
9. *H records componentwise* hsame *transport for $S, T, n, L, R, B, Q, \Gamma, C, P, N$.*

10. *C records displayed* Cont *replay through*

$$S, T, n \longrightarrow L, R, B \longrightarrow Q, \Gamma.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteCardinal}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{KuratowskiFiniteSet}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCardinal}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted cardinal arithmetic, quotient extensionality of subsets, selected representatives, host powersets, or choice-based enumeration.

Theorem 7673.2 (Finite cardinal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCardinal}^{\uparrow}$ *carrier*

$$F = (S, T, n, L, R, B, Q, \Gamma, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCardinal}^{\uparrow}}$ *surface consists exactly of source finite-object admission, initial-segment admission, natural-number cardinality readback, forward ledger exposure, reverse ledger exposure, finite round-trip verification, cardinality-use routing, classifier stability, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A consumer receives the public cardinality only along the route*

$$S, T, n \longrightarrow L, R, B \longrightarrow Q, \Gamma, H, C, P, N.$$

Proof. Project the carrier F. The rows S, T, n are the only entries that name the source finite object, the target initial segment, and the natural-number readback.

The rows L and R display the two directions of the finite comparison. The row B records the two round-trip checks on exactly the displayed source entries and target positions, so no ambient quotient or selected enumeration is available outside the packet.

The consumer route Q and classifier Γ force every cardinality read to preserve the same finite ledgers. The structural rows H, C, P, N transport, replay, source, and name only those rows, so the listed obligations exhaust the local NameCert surface. □

Theorem 7673.3 (Finite cardinal classifier stability)**.** *Let* $F = (S, T, n, L, R, B, Q, \Gamma, H, C, P, N)$ *and* $F' = (S', T', n', L', R', B', Q', \Gamma', H', C', P', N')$ *be accepted* $\mathsf{FiniteCardinal}^{\uparrow}$ *carriers. If the source rows, target segment rows, named natural endpoints, forward ledgers, reverse ledgers, round-trip ledgers, and consumer routes are componentwise* hsame*, then* Γ *and* Γ' *classify the same finite cardinality packet. Conversely, a classifier agreement for* $\mathsf{FiniteCardinal}^{\uparrow}$ *exposes precisely those displayed rows as its comparison surface.*

Proof. In the forward direction, read the componentwise hsame data through H and H'. The classifier rows are defined over the source row, target segment, natural endpoint, both ledger directions, their round-trip checks, and the consumer route, so agreement on those coordinates gives the same classifier read.

For the reverse direction, open the classifier rows Γ and Γ'. By the carrier definition, they have no coordinates other than S, T, n, L, R, B, Q and their primed counterparts. Therefore classifier agreement can expose only those displayed rows, together with the structural transport and replay rows that preserve them. □

Theorem 7673.4 (Finite cardinal bijection ledger)**.** *Every accepted* $\mathsf{FiniteCardinal}^{\uparrow}$ *carrier exports cardinality as a finite bijection ledger. The number n may be cited only after the packet displays the*

forward assignments L, reverse assignments R, and the round-trip row B over the finite source row S and the initial segment T. Thus the packet supplies a constructive finite-cardinality interface for finite-net, compactness, and bounded-family consumers, but it does not supply unrestricted cardinal arithmetic, quotient-set equality, selected representatives, host powersets, or choice.

Proof. Start with the accepted carrier and use Definition 7673.2. The public route to n passes through S, T, L, R, B before it reaches any consumer ledger.

The bijection row B contains both finite round-trip checks, so each source element and each target position is accounted for inside the displayed packet. A compact finite-net consumer such as chapter 7638 can therefore cite the finite count as a ledger-backed size row.

Since the carrier omits unrestricted cardinal arithmetic, quotient extensionality, selected representatives, powersets, and choice, the exported statement is exactly the finite bijection ledger, not an ambient cardinal theory. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCardinal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCardinal}^{\uparrow}$ packet is generated from a displayed finite source row, an initial-segment row over NatUp, a natural-number readback, forward and reverse finite ledgers, two round-trip checks, a consumer route, classifier data, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCardinal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7673.1, Definition 7673.2, Definition 7673.3, and Definition 7673.4 seed the finite bijection and cardinality-ledger surface for $\mathsf{FiniteCardinal}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCardinal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted cardinal arithmetic, quotient sets, selected representatives, host powersets, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-object admission, initial-segment admission, forward and reverse ledger exposure, round-trip verification, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCardinalUp.

Chapter 7674

A Concrete Naming Certificate for $\mathsf{FiniteLimitCone}^{\uparrow}$

Following Mac Lane's category-theoretic account of limits, the $\mathsf{FiniteLimitCone}^{\uparrow}$ packet records the finite cone data needed by limit arguments as BEDC NameCert rows rather than as an imported universal-property object. It sits after the certified $\mathsf{Cat}^{\uparrow}$ surface of chapter 61, the diagram map discipline of $\mathsf{Functor}^{\uparrow}$ in chapter 62, the componentwise comparison discipline of $\mathsf{NatTrans}^{\uparrow}$ in chapter 63, the binary-product route of **??** in chapter 34, the finite pullback route of $\mathsf{Pullback}^{\uparrow}$ in chapter 2240, and the finite-limit refusal boundary of $\mathsf{FiniteLimitSeal}^{\uparrow}$ in chapter 5722. The packet names cone legs and finite naturality checks; it is not a host finite-limit theorem, a selected terminal cone, quotient equality of diagrams, or an ambient category library object.

Definition 7674.1 (Finite-limit cone carrier)**.** *A* $\mathsf{FiniteLimitCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, J, D, A, L, V, H, R, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Cat}^{\uparrow}$ *codomain certificate, exposing object, hom, identity, composition, associativity, unit, classifier, and transport rows for the category in which the cone is read.*
2. *J is the finite* $\mathsf{Cat}^{\uparrow}$ *shape row, naming the inspected indexing objects and arrows of the diagram shape.*
3. *D is the* $\mathsf{Functor}^{\uparrow}$ *diagram row $J \to C$, including object images, arrow images, identity and composition preservation, and the diagram classifier.*
4. *A is the apex row, naming the candidate cone apex as an object of C through the same* $\mathsf{Cat}^{\uparrow}$ *classifier.*
5. *L is the finite leg row, listing the displayed arrows $A \to D(j)$ for every inspected object j of J.*
6. *V is the cone-visibility row, recording the finite naturality squares for every inspected arrow of J and the adjacent* $\mathsf{NatTrans}^{\uparrow}$ *comparison route when the cone is transported along a certified diagram comparison.*
7. *H is componentwise* hsame *transport for $C, J, D, A, L, V, R, P, N$.*
8. *R is the displayed* Cont *replay route through the category, diagram, apex, leg, and visibility rows.*
9. *P is* Pkg *provenance over* $\mathsf{FiniteLimitCone}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, ***??***, $\mathsf{Pullback}^{\uparrow}$, $\mathsf{FiniteLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitCone}^\uparrow}$ *row.*

The carrier has no coordinate for a universal cone, a selected limiting object, quotient equality of diagrams, all finite shapes at once, ambient finite products, ambient pullbacks, or an external category-theory library.

Theorem 7674.2 (Finite-limit cone NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLimitCone}^\uparrow$ *carrier* $K = (C, J, D, A, L, V, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitCone}^\uparrow}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the codomain* $\mathsf{Cat}^\uparrow$ *row, finite shape* $\mathsf{Cat}^\uparrow$ *row,* $\mathsf{Functor}^\uparrow$ *diagram row, apex row, leg row, and cone-visibility row;*
2. *classifier exposure, comparing cones only through the same codomain, shape, diagram, apex, leg, and visibility data under the displayed category and functor classifiers;*
3. *finite cone exactness, requiring every inspected shape arrow to satisfy its displayed naturality square before the leg family is exported;*
4. *sibling compatibility, allowing* ***??****,* $\mathsf{Pullback}^\uparrow$*, and* $\mathsf{FiniteLimitSeal}^\uparrow$ *consumers to read the cone only as finite cone data and not as a completed universal limit;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, R, P, N*.*

Thus a consumer may inspect the finite route

$$\mathsf{Cat}^\uparrow,\ \mathsf{Functor}^\uparrow,\ \mathsf{NatTrans}^\uparrow,\ \textbf{??},\ \mathsf{Pullback}^\uparrow,\ \mathsf{FiniteLimitSeal}^\uparrow \longmapsto \mathsf{FiniteLimitCone}^\uparrow$$

only through the displayed cone rows. It receives no host finite-limit theorem, selected terminal cone, quotient diagram equality, or ambient category-library object.

Proof. Project the accepted carrier. The rows C, J, D, A, L, V are exactly the codomain, finite shape, diagram, apex, leg family, and cone-visibility data, so every exported cone read starts from the finite category and functor surfaces.

The naturality checks are finite because J supplies only the inspected shape rows. The visibility row V records the squares for those rows and the adjacent $\mathsf{NatTrans}^\uparrow$ comparison route, while the product and pullback siblings may consume only the displayed finite legs and square data.

The structural rows H, R, P, N preserve the same packet by transport, replay, provenance, and local naming. Since the carrier contains no universal-cone coordinate, selected limiting object, quotient diagram equality, all-shapes quantifier, ambient finite products, ambient pullbacks, or external category library, the five listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{FiniteLimitCone}^\uparrow$ consumer that reads a finite-limit cone while omitting the displayed codomain category, finite shape, diagram functor, apex, leg family, cone-visibility, transport, replay, provenance, or local naming rows refutes the carrier of Definition 7674.1.

Independence witness: category, functor, nattrans, product, pullback, finite-limit-seal: none is bijective because $\mathsf{FiniteLimitCone}^\uparrow$ jointly carries codomain, finite shape, diagram, apex, leg, finite naturality, sibling compatibility, transport, replay, provenance, and local NameCert rows. Each listed sibling exposes an adjacent surface without the full cone packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitCone}^\uparrow}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLimitCone}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 7674.1 and Definition 7674.2 seed the finite cone carrier and its five-row NameCert surface over Category, Functor, NatTrans, Product, Pullback, FiniteLimitSeal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a host finite-limit theorem, a selected terminal cone, quotient diagram equality, all finite shapes at once, or an ambient category-library object.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, finite cone exactness, sibling compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitConeUp.

Constructive story: A $\mathsf{FiniteLimitCone}^{\uparrow}$ packet is generated from CategoryUp codomain and finite-shape rows, a FunctorUp diagram row, apex and leg rows, finite NatTrans-facing cone-visibility rows, ProductUp/PullbackUp/FiniteLimitSealUp sibling compatibility, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7675

A Concrete Naming Certificate for $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ packet records the bridge from bounded monotone real windows to located Cauchy-filter completion data as finite BEDC rows. It consumes the monotone-window surface of chapter 3750, the located filter surface of chapter 1661, the basis route of chapter 1501, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet names the bounded monotone route into located Cauchy-filter completion without invoking selected subsequences, quotient filters, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7675.1 (CauchyFilterBoundedMonotone carrier)**.** *A* $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, L, F, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *window row naming the bounded monotone finite observations.*
2. *L is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row that answers located filter requests generated from those windows.*
3. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row used by L for refinement and meet data.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the bounded monotone observations.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the monotone comparisons and located filter balls.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, L, F, S, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for the bounded-window, located-filter, basis, stream, readback, dyadic, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bounded monotone precision request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$, $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, F, S, Q, D, E, H, C, P, N$. It has no coordinate for an externally selected subsequence, quotient filter equality, countable choice, hidden basis selection, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7675.2 (CauchyFilterBoundedMonotone NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ *carrier $M = (B, L, F, S, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBoundedMo}}$ *surface consists exactly of carrier admission, bounded monotone window exposure through B, located filter exposure through L, basis refinement through F, finite stream-window exposure through S, regular rational readback through Q, dyadic tolerance accounting through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7675.1. The analytic rows are exactly the bounded monotone window, located Cauchy filter, filter basis, stream window, regular rational readback, dyadic tolerance ledger, and real seal.

The structural rows H, C, P, N preserve, replay, source, and name that same finite tuple. Since the carrier contains no coordinate for a selected infinite subsequence, quotient filter, countable-choice schedule, hidden basis selector, or ambient completeness theorem, the listed rows exhaust the local certificate surface. □

Theorem 7675.3 (CauchyFilterBoundedMonotone regular extraction)**.** *Every accepted* CauchyFilterBoundedMono *packet extracts regular readback data from bounded monotone windows only through the displayed route*

$$B \longmapsto S \longmapsto D \longmapsto Q.$$

The route may then be consumed by the located filter row L and its basis row F, but it cannot be replaced by a selected subsequence, quotient stream equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the bounded monotone row B. It supplies only finite monotone windows, so any readback request must open the stream-window row S and compare the displayed observations through the dyadic ledger D.

The regular rational row Q is downstream of those windows and comparisons. It is therefore a readback surface for the displayed bounded monotone data, not a selected infinite subsequence.

The located filter rows L and F consume the same readback route. Transport, replay, provenance, and local naming preserve the route through H, C, P, N, and none of those rows adds quotient equality, choice, trichotomy, or ambient completeness. □

Theorem 7675.4 (CauchyFilterBoundedMonotone located-filter handoff)**.** *Every located Cauchy-filter consumer of an accepted* $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ *carrier must read the bounded monotone window row B, the stream-window row S, the regular readback row Q, the dyadic tolerance ledger D, and the basis row F before consuming the located filter row L and terminal seal E. The handoff exports no quotient filter, selected limit, countable-choice schedule, hidden basis selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Apply the NameCert obligations of Definition 7675.2. They place the located filter row L in the same finite packet as the bounded monotone, stream, readback, dyadic, basis, and seal rows.

Use the regular extraction theorem Definition 7675.3. The regular readback is already routed through B, S, D, Q, so a located filter request cannot bypass the bounded monotone windows.

Finally read the basis row F and seal row E from the carrier. The basis row supplies the refinement surface for L, and the seal is available only after the displayed route has been consumed. Thus the handoff remains inside the finite rows $B, L, F, S, Q, D, E, H, C, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ read displays one bounded monotone window row, one located Cauchy filter row, one CauchyFilterBasis row, one StreamName window row, one RegSeqRat readback row, one DyadicRatCore tolerance ledger, one RealUp seal, transport, replay, provenance, and local naming before completion-facing data are consumed.

Independence witness: bounded-monotone-subsequence, located-cauchy-filter, cauchy-filter-basis, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds bounded monotone window extraction, located filter requests, basis refinement, regular readback, dyadic tolerance accounting, and terminal real sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}$ packet is generated from bounded monotone windows, located Cauchy-filter rows, CauchyFilterBasis refinement, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise transport, displayed replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7675.1, Definition 7675.2, Definition 7675.3, and Definition 7675.4 seed the finite bridge from bounded monotone real windows to located Cauchy-filter completion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterBoundedMonotone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subsequences, quotient filters, countable choice, hidden basis selection, host limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone window exposure, located filter exposure, basis refinement, regular readback, dyadic tolerance accounting, seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 7676

A Concrete Naming Certificate for $\mathsf{TychonoffProductWindow}^{\uparrow}$

Following Tychonoff and Kelley, the $\mathsf{TychonoffProductWindow}^{\uparrow}$ packet records only the finite product-window compactness data that can be displayed inside BEDC NameCert rows. It sits after the finite product certificate of **??**, the compact metric product route of chapter 1822, the product-distance ledger of chapter 424, the compact source boundary of chapter 60, and the topology carrier of chapter 88. The packet is a finite inspection window for product compactness consumers, not a proof of full product compactness.

Definition 7676.1 (Tychonoff product-window carrier)**.** *A* $\mathsf{TychonoffProductWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, U, K, M, F, T, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-window row, listing exactly the coordinates visible to the packet.*
2. *U is the topology-window row, reading each listed coordinate through the finite open-row interface of* $\mathsf{Topology}^{\uparrow}$*.*
3. *K is the compact-source row, exposing for each listed coordinate only the displayed compact data admitted by* $\mathsf{Compact}^{\uparrow}$*.*
4. *M is the metric product handoff, using* $\mathsf{CompactMetricProduct}^{\uparrow}$ *and* $\mathsf{ProductMetric}^{\uparrow}$ *only when the displayed window is metric.*
5. *F is the finite-product compactness row inherited from* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$ *for the same listed coordinates.*
6. *T is the finite cover-window ledger, naming the visible cover requests and finite response rows inside the displayed product window.*
7. *L is the local compactness-read ledger, recording which compact-source row is read by each visible product-window request.*
8. *H stores componentwise* hsame *transport for $I, U, K, M, F, T, L, C, P, N$.*
9. *C stores displayed* Cont *replay from a consumer request through the same finite product-window rows.*

10. *P is the* Pkg *provenance over* $\mathsf{TychonoffProductWindow}^{\uparrow}$, $\mathsf{TychonoffFiniteProduct}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TychonoffProductWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an arbitrary product family, a countable product, an ultrafilter, a selected subcover, a host product point, or open-cover compactness beyond the displayed finite window.

Theorem 7676.2 (Tychonoff product-window NameCert obligations)**.** *For an accepted* $\mathsf{TychonoffProductWindow}^{\uparrow}$ *carrier*

$$W = (I, U, K, M, F, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TychonoffProductWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index window, topology rows, compact-source rows, metric product handoff, finite-product compactness row, cover-window ledger, local compactness-read ledger, componentwise transport, replay, provenance, and local naming. Every compactness read must first expose the displayed route*

$$\mathsf{Topology}^{\uparrow}, \mathsf{Compact}^{\uparrow}, \mathsf{TychonoffFiniteProduct}^{\uparrow}, \mathsf{CompactMetricProduct}^{\uparrow}, \mathsf{ProductMetric}^{\uparrow} \longmapsto I, U, K, M, F, T, L, H, C$$

The certificate exports finite product-window compactness data only; it does not assert arbitrary Tychonoff products, countable products, choice-selected subcovers, ultrafilters, host product points, or open-cover compactness beyond displayed finite windows.

Proof. Project the carrier of Definition 7676.1. The index row I fixes the visible coordinates, and the topology, compact, metric-product, finite-product, cover-window, and local compactness rows are all displayed before the structural rows are read.

Read U and K first. They expose the finite topology and compact-source data supplied by chapter 88 and chapter 60, without forming an ambient product of all coordinate points.

Next read F through **??**. In metric windows, the same request may also pass through M using chapter 1822 and chapter 424. These rows provide only the finite product-window route already named by I.

Finally, T, L, H, C, P, N record the cover-window request, local compactness reads, transport, replay, provenance, and local naming. Since the packet contains no row for countable products, arbitrary products, selected subcovers, ultrafilters, host product points, or open-cover compactness outside the finite window, the NameCert surface is exhausted by the displayed data. □

Closure status for $\mathsf{NameCert}_{\mathsf{TychonoffProductWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{TychonoffProductWindow}^{\uparrow}$ packet is generated from a finite index window, topology rows, compact-source rows, metric product handoff, TychonoffFiniteProduct finite-product rows, finite cover-window ledgers, local compactness-read ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TychonoffProductWindow}^{\uparrow}}$, seed)

Scope closed: Definition 7676.1 and Definition 7676.2 define only the finite product-window carrier and its displayed NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TychonoffProductWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary Tychonoff products, countable products, choice-selected subcovers, ultrafilters, host product points, or open-cover compactness beyond displayed finite windows.

Upgrade path: The next step is Lean alignment of the finite index window, topology rows, compact-source rows, metric product handoff, finite-product row, cover-window ledger, local compactness-read ledger, transport, replay, provenance, and local naming for BEDC.Derived.TychonoffProductWindowUp.

Chapter 7677

A Concrete Naming Certificate for $\mathsf{CauchyProductTheorem}^{\uparrow}$

Following Bishop and Bridges on constructive real arithmetic, the $\mathsf{CauchyProductTheorem}^{\uparrow}$ packet records the finite L10 route saying that two displayed regular Cauchy rational sources admit a product name once the source windows, dyadic product ledger, regular readback, and terminal Real seal are all visible. It sits between the local product-budget packet of chapter 676, the regular rational source surface of chapter 301, the finite stream-window discipline of chapter 126, the tolerance arithmetic of chapter 291, and the terminal real-name boundary of chapter 38. The chapter names only the finite product theorem corridor for regular Cauchy names; it does not assert quotient real multiplication, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7677.1 (Cauchy product theorem carrier)**.** *A* $\mathsf{CauchyProductTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, S, T, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows. Each row exposes the finite rational windows and regularity witnesses consumed by the product request.*
2. *S and T are paired* $\mathsf{StreamName}^{\uparrow}$ *window rows for the same two sources, indexed by the visible tolerance requests used by the product route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product row. It records the finite dyadic product cells, denominator refinements, and tolerance comparisons needed to multiply the inspected source windows.*
4. *L is the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger binding the source moduli, dyadic product row, and terminal tolerance request into one displayed error account.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *product handoff row obtained from A, B, S, T, D, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the product handoff row R is visible.*
7. *H records componentwise* hsame *transport for the source rows, stream windows, dyadic product row, error ledger, product handoff, Real seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a product-theorem request through $A, B, S, T, D, L, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductTheorem}^{\uparrow}$, $\mathsf{RegularCauchyProductBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient equality of completed reals, host real multiplication, selected limiting products, countable-choice schedules, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyProductTheoremUp.CauchyProductTheoremUp`
Lean4 ✓ `BEDC.Derived.CauchyProductTheoremUp.CauchyProductTheoremTasteGate single carrier alignment`

Theorem 7677.2 (Cauchy product theorem finite route)**.** *For every accepted* $\mathsf{CauchyProductTheorem}^{\uparrow}$ *carrier* $K = (A, B, S, T, D, L, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing product read factors through the finite route*

$$A, B \longmapsto S, T \longmapsto D, L \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the theorem packet exposes two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *product row, a* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *product handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* hsame/Cont/Pkg/NameCert *support rows before any product name can be consumed.*

Proof. Project the carrier of Definition 7677.1. The accepted packet exposes exactly the two source rows, the paired stream windows, the dyadic product row, the finite error ledger, the product handoff row, and the terminal Real seal.

The source stage is A, B, S, T. The two $\mathsf{RegSeqRat}^{\uparrow}$ rows supply the rational Cauchy data, while the two $\mathsf{StreamName}^{\uparrow}$ rows display the finite windows requested by the product consumer.

The arithmetic stage is D, L. The $\mathsf{DyadicRatCore}^{\uparrow}$ row records the finite product cells and tolerance comparisons, and the $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger binds those comparisons to the source moduli and the requested output tolerance.

The handoff stage is R, E. The product handoff row repacks the displayed budget as a regular rational product name, and the terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after that row is visible.

Finally H, C, P, N transport, replay, package, and locally name the same route. Since no excluded object is a carrier coordinate, a product consumer cannot bypass the displayed regular source rows, stream windows, dyadic product row, error ledger, handoff, or Real seal. □

Theorem 7677.3 (Cauchy product theorem budget support)**.** *Every accepted* $\mathsf{CauchyProductTheorem}^{\uparrow}$ *packet consumes the* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *surface before its product handoff row can be read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data. In the carrier* $K = (A, B, S, T, D, L, R, E, H, C, P, N)$*, the budget surface is read through* A, B, S, T, D, L*, while* R, E, H, C, P, N *only expose the resulting regular product name, Real seal, and structural support.*

Proof. Start with the finite route of Definition 7677.2. Its first two stages contain the source regular Cauchy rows, paired stream windows, dyadic product arithmetic, and finite error ledger.

These are exactly the rows inspected by the $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ packet: source moduli, product cells, and tolerance bookkeeping appear before the product handoff row R is available.

The remaining rows do not create a second product computation. They read back the displayed budget as a regular product name, seal it as $\mathsf{Real}^{\uparrow}$ data, and preserve the route under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductTheorem}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{DyadicRatCore}^{\uparrow}$ product row, the $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ product handoff, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: regularcauchyproductbudget, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole theorem packet $K = (A, B, S, T, D, L, R, E, H, C, P, N)$, because each sibling contributes only the budget, source readback, stream windows, arithmetic ledger, or terminal seal while this chapter binds them as one finite Cauchy product theorem certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductTheorem}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore product row, a FiniteErrorBudget ledger, a RegSeqRat product handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7677.1, Definition 7677.2, and Definition 7677.3 seed the finite Cauchy product theorem route from regular Cauchy source rows to stream windows, dyadic product arithmetic, finite error budgeting, regular product handoff, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real multiplication, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-window stability, dyadic product arithmetic, finite error budgeting, RegSeqRat product handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductTheoremUp.

Chapter 7678

A Concrete Naming Certificate for $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$

Following Kock and Zoberlein on lax idempotent monads, the $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ packet records the finite doctrine-level certificate by which Cauchy completion flattening is consumed as an idempotent handoff rather than as quotient completed objects. It sits beside the Cauchy completion monad join of chapter 7559, the completion idempotence surface of chapter 2381, the Yoneda completion packet of chapter 2148, the Lawvere metric completion profile of chapter 3947, and the finite readback stack of chapter 126, chapter 301, chapter 291, chapter 38, and chapter 360. The packet names the doctrine row that lets downstream completion consumers read repeated completion through the same finite carrier surface, without host equality of completed objects, quotient representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7678.1 (Kock–Zoberlein Cauchy completion carrier)**.** *A* $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, M, J, I, Y, L, S, R, Q, E, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the lax-idempotent doctrine row naming the Kock–Zoberlein orientation of the completion read.*
2. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *source row recording the unit-and-bind schedule read before any flattening.*
3. *J is the* $\mathsf{CauchyCompletionMonadJoin}^{\uparrow}$ *row that exposes the completion-of-completion join.*
4. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *row requiring the repeated completion read to reuse the same finite completion boundary.*
5. *Y is the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row that supplies presheaf-style completion observations.*
6. *L is the* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *directed-profile row used when the doctrine is read over one-sided enriched metrics.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row.*
9. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the completion join and idempotence rows.*

10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*
11. *F is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff row available only after $M, J, I, Y, L, S, R, Q, E$ have been displayed.*
12. *H records componentwise* hsame *transport for the displayed doctrine and completion rows.*
13. *C records displayed* Cont *replay from doctrine orientation through monad join, idempotence, profile, finite-window, readback, tolerance, Real seal, and completion handoff.*
14. *P is the* Pkg *provenance over* $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionMo $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{LawvereMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}}$ *row.*

The carrier exports no quotient completion object, selected completion representative, host equality of completed objects, countable-choice schedule, global colimit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 7678.2 (Kock–Zoberlein Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ *carrier $K = (D, M, J, I, Y, L, S, R, Q, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}}$ *surface consists exactly of doctrine orientation through D, Cauchy completion monad exposure through M, monad join exposure through J, completion idempotence through I, Yoneda profile exposure through Y, Lawvere metric completion exposure through L, finite window exposure through S, regular-rational readback through R, dyadic tolerance accounting through Q, Real sealing through E, Real Cauchy completion handoff through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. Its coordinates display the doctrine row, monad row, join row, idempotence row, Yoneda profile, Lawvere directed profile, finite-window route, regular-rational readback, dyadic ledger, Real seal, Real Cauchy completion handoff, and structural support rows.

The completion rows are consumed in doctrine order. The monad source M is read before the join row J, the idempotence row I constrains the repeated completion read, and the profile rows Y, L provide the Yoneda and Lawvere observations that downstream completion consumers may inspect.

The finite analytic rows S, R, Q, E, F supply the BEDC readback boundary after the doctrine rows have been displayed. Transport, replay, provenance, and local naming preserve that same packet, so the listed rows exhaust the local NameCert obligations. □

Theorem 7678.3 (Kock–Zoberlein Cauchy completion idempotent handoff)**.** *Every completion-of-completion consumer of an accepted* $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ *packet must factor through the finite route*

$$D, M, J, I, Y, L, S, R, Q, E, F, H, C, P, N.$$

The consumer receives the Cauchy completion monad row, monad join row, completion idempotence row, Yoneda profile row, Lawvere metric completion row, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, and RealCauchyCompletion handoff before it can treat the repeated completion read as flattened. The route supplies an idempotent handoff surface, not a quotient completed object, selected representative, host equality of completed objects, countable choice, global colimit, or ambient completeness theorem.

Proof. Begin with the doctrine and monad rows. The Kock–Zoberlein orientation D can be consumed only together with the displayed completion monad row M and the join row J, so no consumer reaches a flattened completion read before the completion-of-completion input has been named.

Next read the idempotence and profile rows. The row I forces the repeated completion request to reuse the displayed completion boundary, while Y and L expose the Yoneda and Lawvere profile data that make the handoff visible to completion-facing readers.

Finally pass through $S, R, Q, E, F, H, C, P, N$. These rows provide the finite observation windows, regular readback, tolerance accounting, terminal Real seal, Real Cauchy completion handoff, transport, replay, provenance, and local naming. Since the carrier contains no quotient object, selected representative, host equality, choice schedule, colimit coordinate, or ambient completeness row, the idempotent handoff cannot export any of those endpoints. □

Falsifiable prediction: Every accepted $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ read exposes the Cauchy completion monad, monad join, completion idempotence, Yoneda profile, Lawvere metric completion profile, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, and RealCauchyCompletion handoff before any repeated completion read is consumed as flattened.

Independence witness: cauchy-completion-monad-join, completion-idempotence, yoneda-completion, lawvere-metric-completion, streamname, regseqrat, dyadicratcore, real, real-cauchy-completion: none is bijective to $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ because this packet jointly binds lax-idempotent doctrine orientation, completion monad source, monad join, idempotence reuse, Yoneda and Lawvere profile rows, finite readback rows, Real sealing, Real Cauchy completion handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}$ packet is generated from a lax-idempotent doctrine row, CauchyCompletionMonad row, CauchyCompletionMonadJoin row, CompletionIdempotence row, YonedaCompletion row, LawvereMetricCompletion row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, RealUp seal, RealCauchyCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7678.1, Definition 7678.2, and Definition 7678.3 seed the finite lax-idempotent completion-flattening route over completion monad, join, idempotence, profile, finite-window, readback, tolerance, Real seal, and Real Cauchy completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KockZoberleinCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.KockZoberleinCauchyCompletionUp.KockZoberleinCauchyCompletionUp`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completed objects, selected representatives, host equality of completed objects, countable choice, global colimits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, doctrine-orientation exactness, completion monad join exposure, idempotent handoff non-escape, Yoneda and Lawvere profile compatibility, finite readback preservation, Real seal preservation, RealCauchyCompletion handoff, transport, replay, provenance, and local naming for BEDC.Derived.KockZoberleinCauchyCompletionUp.

Chapter 7679

A Concrete Naming Certificate for $\mathsf{RiemannStieltjesSum}^{\uparrow}$

Following Riemann and Stieltjes, the $\mathsf{RiemannStieltjesSum}^{\uparrow}$ packet records the finite product-sum ledger used when a displayed integrand window is read against displayed integrator differences. It sits beside the tagged finite-sum surface of chapter 1615, the upper and lower Stieltjes-sum boundary of chapter 7558, and the terminal real-name seal of chapter 38. The packet gives Real-facing finite integration consumers a named row for one displayed Riemann–Stieltjes sum; it does not assert an integral value, bounded-variation completeness, a refinement limit, a quotient of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7679.1 (Riemann–Stieltjes sum carrier)**.** *A* $\mathsf{RiemannStieltjesSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, F, G, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the tagged partition row, exposing ordered cells, endpoints, tags, and tag-containment witnesses through the finite* $\mathsf{RiemannSum}^{\uparrow}$ *surface.*
2. *F is the integrand window row, recording exactly the finitely many displayed integrand reads required at the carried tags.*
3. *G is the integrator-difference row, recording for each carried cell the displayed endpoint difference of the integrator between the cell endpoints.*
4. *L is the finite product-sum ledger, pairing each integrand read in F with the matching integrator difference in G and recording the carried arithmetic sum over T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, fixing the dyadic budget used to compare the product-sum ledger with later regulated-integral or Darboux–Stieltjes consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed finite product-sum and tolerance ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the tagged partition, integrand window, integrator differences, product-sum ledger, dyadic budget, and regular readback are displayed.*
8. *H records componentwise* hsame *transport for the tagged partition, integrand window, integrator differences, product-sum ledger, tolerance row, readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the tagged partition through*

$$T, F, G \longmapsto L \longmapsto D \longmapsto R \longmapsto E$$

and then through the structural rows.

10. *P is the* Pkg *provenance over* $\mathsf{RiemannStieltjesSum}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxStieltjesSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F, G, L, D, R, E, H, C, P, N$. It has no coordinate for a completed integral, a limit over all refinements, bounded-variation completeness, quotient function equality, a selected tag family outside T, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7679.2 (Riemann–Stieltjes sum NameCert obligations)**.** *For every accepted* $\mathsf{RiemannStieltjesSum}^{\uparrow}$ *carrier $S = (T, F, G, L, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesSum}^{\uparrow}}$ *surface consists exactly of carrier admission for the tagged partition, integrand window, integrator-difference row, finite product-sum ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every accepted read follows the finite route*

$$T, F, G \longmapsto L \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Project the carrier of Definition 7679.1. Its displayed rows are exactly the tagged partition, integrand reads, integrator differences, finite product-sum ledger, dyadic tolerance, regular readback, Real seal, and structural rows.

The arithmetic route is cellwise. Each summand is admitted only when the tag comes from T, the value comes from F, and the matching integrator difference comes from G on the same displayed cell.

The ledger L records only the finite product-sum built from those rows. The tolerance row D and readback row R make the numerical comparison visible before the terminal seal E can be consumed.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same route. Since no carrier coordinate names a completed integral, refinement limit, quotient function, hidden tag selector, countable-choice principle, or ambient completeness theorem, the listed rows exhaust the obligation surface. □

Theorem 7679.3 (Riemann–Stieltjes sum Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RiemannStieltjesSum}^{\uparrow}$ *packet must read the tagged partition row, integrand window, integrator-difference row, finite product-sum ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal seal is inspected. The seal exports only the displayed finite Riemann–Stieltjes sum route, not an integral value, a refinement limit, bounded-variation completeness, quotient function equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface of Definition 7679.2. It orders the visible route through T, F, G, L, D, R, E, with the structural rows attached afterward.

The $\mathsf{Real}^{\uparrow}$ seal row E is downstream from the regular readback row R. Thus a consumer cannot inspect the seal until the finite product-sum ledger and dyadic tolerance row have already been displayed.

The sibling $\mathsf{RiemannSum}^{\uparrow}$ surface supplies the finite tagged-partition reading, and the $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ surface supplies a comparison boundary for upper and lower Stieltjes rows. Neither sibling adds a completed integral coordinate to this carrier.

Therefore the terminal seal is only a Real-facing seal for the displayed finite sum route. All completed integration, refinement-limit, quotient-function, choice, and ambient-completeness readings remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{RiemannStieltjesSum}^{\uparrow}$ read exposes tagged partition, integrand window, integrator-difference, finite product-sum, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any Real-facing consumer inspects the seal.

Independence witness: riemannsum, darboux-stieltjes-sum, real: $\mathsf{RiemannSum}^{\uparrow}$ supplies tagged finite-sum rows without integrator differences, $\mathsf{DarbouxStieltjesSum}^{\uparrow}$ supplies upper and lower Stieltjes comparison rows rather than one tagged product-sum ledger, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal. None is bijective with the complete tagged partition, integrand-window, integrator-difference, finite product-sum, tolerance, readback, transport, replay, provenance, and local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannStieltjesSum}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannStieltjesSum}^{\uparrow}$ packet is generated from a tagged partition row, integrand window, integrator differences, finite product-sum ledger, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7679.1, Definition 7679.2, and Definition 7679.3 seed the finite Riemann–Stieltjes product-sum route over tagged partition, integrand-window, integrator-difference, tolerance, readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Riemann–Stieltjes integration, bounded-variation completeness, refinement limits, quotient function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged partition exposure, integrand-window readback, integrator-difference exposure, finite product-sum exactness, DyadicRatCore tolerance control, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannStieltjesSumUp.

Chapter 7680

A Concrete Naming Certificate for $\mathsf{SequentialMetricSpace}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{SequentialMetricSpace}^{\uparrow}$ packet records a sequence-indexed metric interface as finite BEDC NameCert data. It sits between the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the stream naming discipline of chapter 126, the regular rational readback of chapter 301, the $\mathsf{Real}^{\uparrow}$ completion seal of chapter 38, and the filter-side completion route of chapter 333. The packet names sequential Cauchy observations without importing host sequences, quotient completions, or an ambient completeness theorem.

Definition 7680.1 (SequentialMetricSpace carrier)**.** *A* $\mathsf{SequentialMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, M, U, O, C, S, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the named point carrier read from the underlying* $\mathsf{Met}^{\uparrow}$ *surface;*
2. *M is the metric observation row, recording only displayed distance comparisons and no host metric library object;*
3. *U is the finite* UnaryHistory *index row used to address the observed sequence windows;*
4. *O is the observation table pairing each displayed unary index window with metric reads in M;*
5. *C is the sequential Cauchy ledger, expressed as finite tolerance-to-window data over U, O;*
6. *S is the stream-name handoff row consumed by* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readers;*
7. *R is the real-readback row consumed by* $\mathsf{Real}^{\uparrow}$ *and by the* $\mathsf{CauchyFilter}^{\uparrow}$ *comparison route;*
8. *H records componentwise* hsame *transport for point, metric, index, observation, Cauchy, stream, real-readback, continuation, provenance, and naming rows;*
9. *Q records the displayed* Cont *routes from metric observations to stream, regular rational, real, and Cauchy-filter consumers;*
10. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and the local certificate;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialMetricSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host infinite sequence, quotient completion, selected representative of a Cauchy class, classical choice principle, or ambient theorem asserting that every sequential Cauchy object has a limit.

Theorem 7680.2 (SequentialMetricSpace NameCert obligations)**.** *For an accepted* $\mathsf{SequentialMetricSpace}^{\uparrow}$ *carrier* $B = (X, M, U, O, C, S, R, H, Q, P, N)$*, the local* NameCert *surface exposes exactly the point carrier* X*, metric observation row* M*, unary history index row* U*, observation table* O*, sequential Cauchy ledger* C*, stream-name handoff* S*, real-readback row* R*, componentwise transport* H*, displayed continuation routes* Q*, provenance* P*, and local naming* N*. Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer must read through those displayed rows.*

Proof. Project the carrier. The rows X, M, U, O, C give the finite metric and sequential observation surface, while S and R are the only rows that export the sequence-indexed data to downstream completion readers.

The transport and continuation rows preserve the same inventory. Componentwise hsame movement through H can move an accepted packet only row by row, and every Cont route in Q begins with the displayed metric observations and sequential Cauchy ledger.

Finally inspect P and N. Provenance names the metric, stream, regular rational, real, and Cauchy-filter consumers, and local naming records that no host sequence, quotient completion, selected representative, or ambient completeness theorem is part of the certificate. □

Theorem 7680.3 (SequentialMetricSpace Cauchy readback)**.** *For an accepted* $\mathsf{SequentialMetricSpace}^{\uparrow}$ *carrier* $B = (X, M, U, O, C, S, R, H, Q, P, N)$*, every Cauchy readback consumed by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{CauchyFilter}^{\uparrow}$ *factors through the finite ledger*

$$U, O, C, S, R.$$

The readback may use the displayed metric observations and the sequential Cauchy ledger, but it cannot replace them by a host sequence, quotient class, unindexed filter, or a completed real supplied outside the packet.

Proof. Start with the index and observation rows. The row U supplies the displayed unary windows, and O attaches the finite metric observations used by the Cauchy ledger C.

The handoff rows then determine the downstream reads. Stream and regular rational consumers read through S, while real and Cauchy-filter consumers read through R together with the same ledger data.

The support rows H, Q, P, N transport, replay, package, and name that finite route. Since the carrier lists no host sequence, quotient completion, unindexed filter, or external completed real coordinate, every accepted Cauchy readback remains inside U, O, C, S, R. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialMetricSpace}^{\uparrow}$ packet is generated from a point carrier row, metric observation row, finite unary history index row, observation table, sequential Cauchy ledger, stream-name handoff, real-readback row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialMetricSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7680.1, Definition 7680.2, and Definition 7680.3 bind the finite sequence-indexed metric carrier and its Cauchy readback route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialMetricSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove metric completion, quotient completion, selected limits for all Cauchy sequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric observation stability, sequential Cauchy ledger exactness, stream and regular rational handoff, real and Cauchy-filter readback, and non-escape for the displayed finite packet.

Chapter 7681

A Concrete Naming Certificate for $\mathsf{EulerianNumber}^{\uparrow}$

Following Euler and standard enumerative combinatorics, the $\mathsf{EulerianNumber}^{\uparrow}$ packet records descent-count data for finite permutations as a displayed NameCert surface. It sits after the natural-number endpoint of chapter 28, the finite spine discipline of chapter 36, the polynomial table surface of chapter 50, and the Young-ledger partition interface of chapter 201. The packet names Eulerian-number rows without importing host finite sets, quotient permutation equality, selected representatives, infinite generating functions, or ambient representation theory.

Definition 7681.1 (Eulerian number carrier)**.** *An* $\mathsf{EulerianNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (n, k, \Pi, D, A, R, V, H, C, P, N)$$

with the following displayed rows:

1. *n and k are* $\mathsf{Nat}^{\uparrow}$ *rows naming the permutation length and requested descent count.*
2. *Π is the finite permutation-window row, represented as a* $\mathsf{List}^{\uparrow}$ *ledger of displayed one-line permutation names over the initial segment* $0, \ldots, n-1$*.*
3. *D is the descent-position row, listing for each displayed permutation the adjacent positions where the next entry is smaller.*
4. *A is the descent-count row, reading each displayed descent list through* $\mathsf{Nat}^{\uparrow}$ *and comparing it with k.*
5. *R is the recurrence-table row, recording the finite Eulerian recurrence cells used to compute the entry indexed by (n, k).*
6. *V is the polynomial and table provenance row, reading Eulerian polynomial coefficients through* $\mathsf{Poly}^{\uparrow}$ *and recording the compatible triangular table readback.*
7. *H records componentwise* hsame *transport for length, descent count, permutation window, descent positions, count row, recurrence table, provenance row, continuation row, package row, and local naming row.*
8. *C records displayed* Cont *replay from Π to D, from D to A, from the table row R to the polynomial row V, and from all public rows to the local certificate.*
9. *P is the* Pkg *provenance over* $\mathsf{EulerianNumber}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{Partition}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{EulerianNumber}^{\uparrow}}$ *row.*
The carrier has no coordinate for host finite-set quotients, unbounded permutation spaces, selected representative permutations, analytic generating functions, or representation-ring equality.

Lean4 (definition) `BEDC.Derived.EulerianNumberUp.EulerianNumberUp`
Lean4 ✓ `BEDC.Derived.EulerianNumberUp.EulerianNumberTasteGate single carrier alignment`

Theorem 7681.2 (Eulerian number NameCert obligations)**.** *For every accepted* $\mathsf{EulerianNumber}^{\uparrow}$ *carrier*

$$E = (n, k, \Pi, D, A, R, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerianNumber}^{\uparrow}}$ *surface consists exactly of length and descent-count admission, finite permutation-window exposure, descent-position extraction, descent-count comparison, recurrence-table readback, polynomial and table provenance, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every consumer of the number $A(n, k)$ must read*

$$n, k, \Pi, D, A, R, V, H, C, P, N$$

before citing the Eulerian entry.

Proof. Project the carrier E. The rows n, k, Π give the only admissible length, count, and finite permutation-window data.

The rows D and A compute descent positions and descent counts inside the displayed window. The recurrence row R and polynomial provenance row V supply the table-facing and coefficient-facing readbacks for the same entry.

Transport, replay, package provenance, and local naming are confined to H, C, P, N. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7681.3 (Eulerian number descent recurrence)**.** *Let $E = (n, k, \Pi, D, A, R, V, H, C, P, N)$ be an accepted* $\mathsf{EulerianNumber}^{\uparrow}$ *carrier. Every recurrence-facing read of the Eulerian entry indexed by (n, k) factors through the finite table row R and the descent ledger rows Π, D, A. The recurrence table may be replayed as the usual finite split by the last insertion position: entries preserving a descent are read from the cell indexed by $(n-1, k)$, and entries creating a descent are read from the cell indexed by $(n-1, k-1)$. The polynomial provenance row V records the same table as coefficients of the Eulerian polynomial before any* $\mathsf{Partition}^{\uparrow}$ *or representation-facing consumer may cite it.*

Proof. Start with the displayed permutation window Π. The descent rows D and A classify each listed permutation by the number of descents, so the requested entry is counted only through the finite window and the requested $\mathsf{Nat}^{\uparrow}$ row k.

Read the recurrence table R. The insertion-position split is a finite table operation over displayed rows: one branch preserves the descent count and reads $(n-1, k)$, while the other branch creates the displayed descent and reads $(n-1, k-1)$.

Finally read V, H, C, P, N. The polynomial row records the same triangular table as coefficients, while transport, replay, provenance, and naming preserve the displayed recurrence route for downstream $\mathsf{Poly}^{\uparrow}$ and $\mathsf{Partition}^{\uparrow}$ consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerianNumber}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EulerianNumber}^{\uparrow}$ packet is generated from NatUp length and descent-count rows, a ListUp finite permutation window, descent-position and descent-count ledgers, a finite recurrence table, polynomial/table provenance, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{EulerianNumber}\uparrow}$, seed)

Scope closed: Definition 7681.1, Definition 7681.2, and Definition 7681.3 seed the finite Eulerian descent-count certificate over displayed permutation windows and recurrence-table rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{EulerianNumber}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host finite-set quotients, unrestricted permutation equality, selected representatives, infinite generating functions, representation-ring equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, permutation-window admission, descent-position extraction, descent-count comparison, recurrence-table readback, polynomial provenance, transport, replay, package provenance, and local naming for BEDC.Derived.EulerianNumberUp.

Chapter 7682

A Concrete Naming Certificate for MetaCICClosureSubstitutionFrontier$^{\uparrow}$

The MetaCICClosureSubstitutionFrontier$^{\uparrow}$ packet realizes the beta-audit obstruction route of section 9384.4 as finite beta-trigger, closedness, substitution-boundary, and subject-reduction frontier rows. It sits between the beta audit packet of chapter 761, the closure-preservation audit packet of chapter 791, the subject-reduction discharge socket of chapter 719, the confluence substitution boundary of chapter 3791, and the normal-form consistency seal of chapter 740. The packet records the finite frontier where beta movement remains closed but still has to cross typed substitution and subject-reduction boundaries; it does not assert subject reduction, confluence, normalization, conversion typing, quotient syntax equality, or kernel consistency.

Definition 7682.1 (MetaCIC closure-substitution frontier carrier)**.** *A* MetaCICClosureSubstitutionFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, S, O, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the beta-trigger row, exposing the direct beta contraction, application-argument congruence, lambda-domain congruence, and Pi-domain congruence triggers named by the beta audit map.*
2. *K is the* ClosedAt *preservation row, recording the closedness-preservation route through shift, substitution, one-step beta, and finite beta-chain audit rows.*
3. *S is the substitution-boundary row, retaining the point where substitution through binders must be inspected before a typed preservation consumer may proceed.*
4. *O is the obstruction row, recording dependent-codomain, lambda-domain, Pi-domain, and beta-substitution typing boundaries that are not discharged by closedness alone.*
5. *R is the subject-reduction frontier row, exposing the discharge socket route before any subject-reduction consumer may inspect the packet.*
6. *H records componentwise* hsame *transport for beta-trigger, closedness, substitution-boundary, obstruction, subject-reduction, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$B, K, S, O \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* MetaCICClosureSubstitutionFrontier$^{\uparrow}$, MetaCICBetaAudit$^{\uparrow}$, *chapter 791,* SubjectReductionDischargeSocket$^{\uparrow}$, MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$, NormalFormConsistencySe BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{MetaCICClosureSubstitutionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, S, O, R, H, C, P, N$. It has no coordinate for closed subject reduction, full confluence, normalization, conversion typing, quotient equality of terms, hidden substitution admissibility, or kernel consistency.

Theorem 7682.2 (MetaCIC closure-substitution frontier NameCert obligations)**.** *For every accepted* MetaCICClosureSubstitutionFrontier$^{\uparrow}$ *carrier $F = (B, K, S, O, R, H, C, P, N)$, the local* NameCert$_{\text{MetaCICClosureS}}$ *surface consists exactly of carrier admission for beta triggers,* ClosedAt *preservation, substitution-boundary retention, obstruction retention, subject-reduction frontier exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any subject-reduction consumer must read the route*

$$B, K, S, O \longmapsto R$$

before it may consume the socket-facing row.

Proof. Project the carrier of Definition 7682.1. The displayed rows are exactly the beta triggers, closedness-preservation row, substitution-boundary row, obstruction row, subject-reduction frontier row, and structural rows.

The beta audit contributes B. Its direct beta trigger and binder-domain triggers identify where reduction starts, but they do not by themselves provide typed preservation.

The closure-preservation audit contributes K. It keeps ClosedAt stable through shift, substitution, one-step beta, and finite beta chains, while leaving the typed substitution boundary S visible.

The obstruction row O records the dependent-codomain and binder-domain typing boundaries. The subject-reduction frontier row R can therefore be read only after B, K, S, O are displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores subject reduction, confluence, normalization, conversion typing, quotient equality, hidden substitution admissibility, or kernel consistency, the listed rows exhaust the local obligation surface. □

Theorem 7682.3 (MetaCIC closure-substitution frontier subject-reduction boundary)**.** *Every subject-reduction-facing consumer of an accepted* MetaCICClosureSubstitutionFrontier$^{\uparrow}$ *packet must expose the beta-trigger row B, the* ClosedAt *preservation row K, the substitution-boundary row S, and the obstruction row O before the discharge socket row R is available. The packet may hand closed beta-trigger data to the subject-reduction socket, confluence substitution boundary, or normal-form consistency seal, but it does not discharge beta-substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, confluence, normalization, or consistency.*

Proof. Begin with the beta-trigger row B. It names the syntactic redex and congruence sources from the beta audit map, so it fixes where the consumer is allowed to inspect reduction.

Read the closedness row K next. This row supplies ClosedAt preservation through the closure-preservation audit route, but the row is still a closedness statement rather than a typed preservation statement.

The substitution-boundary row S and obstruction row O retain the typed interfaces touched by substitution through binders, dependent codomains, lambda domains, and Pi domains. They must remain visible before R can be read.

The discharge socket row R is therefore only a frontier exposure. It may feed the sibling socket and boundary chapters, but none of the missing preservation, confluence, normalization, or consistency claims is a coordinate of the carrier. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}$ read exposes BetaStep trigger, ClosedAt preservation, substitution-boundary, obstruction, transport, replay, provenance, and local NameCert rows before any subject-reduction consumer inspects the socket-facing row.

Independence witness: metacicbetaaudit, metacicclosurepreservationaudit, subject-reduction-discharge-socket, metacic-confluence-substitution-boundary, normal-form-consistency-seal: the beta audit records reduction triggers without the closedness-to-substitution frontier, the closure-preservation audit records ClosedAt stability without typed discharge, the subject-reduction socket records discharge rows without beta-trigger provenance, the confluence substitution boundary records residual substitution premises without the subject-reduction frontier row, and the normal-form consistency seal records endpoint consistency boundaries without the substitution-frontier carrier. None is bijective with the complete beta-trigger, closedness, substitution-boundary, obstruction, subject-reduction, transport, replay, provenance, and local NameCert packet.

The later Lean TasteGate file for this route must include explicit Nontrivial $\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}$ and FieldFaithful $\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}$ instances whose fields project $B, K, S, O, R, H, C, P, N$ separately.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}$ packet is generated from beta-trigger rows, ClosedAt preservation rows, substitution-boundary rows, obstruction rows, subject-reduction frontier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7682.1, Definition 7682.2, and Definition 7682.3 seed the finite closure-substitution frontier over beta-trigger, ClosedAt, substitution-boundary, obstruction, subject-reduction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta-substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, subject reduction, confluence, normalization, conversion typing, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, BetaStep trigger exposure, ClosedAt preservation, substitution-boundary retention, obstruction retention, subject-reduction frontier exposure, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICClosureSubstitutionFrontierUp.

Chapter 7683

A Concrete Naming Certificate for $\mathsf{VitaliCoveringLemma}^{\uparrow}$

Following Vitali and Bishop–Bridges finite covering practice, the $\mathsf{VitaliCoveringLemma}^{\uparrow}$ packet records the displayed finite-gauge route by which admissible cover rows yield a disjoint finite subfamily and a finite covering handoff. It sits after the gauge presentation of chapter 4302, the disjoint selection surface of chapter 2184, the finite subcover ledger of chapter 6125, the regular rational readback rows of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite gauge requests, candidate cover rows, disjoint selection rows, coverage handoff rows, rational readback, real sealing, componentwise transport, displayed replay, provenance, and local naming; it does not import a host Vitali theorem, countable choice, measure completion, selected infinite subfamilies, quotient point equality, or ambient real completeness.

Definition 7683.1 (Vitali covering lemma carrier)**.** *A* $\mathsf{VitaliCoveringLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, V, A, S, C, F, R, E, H, K, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeSpace}^{\uparrow}$ *row naming the finite gauge family and tolerance request.*
2. *V is the* $\mathsf{VitaliCover}^{\uparrow}$ *cover row that admits candidate cells for the displayed gauge request.*
3. *A is the candidate-admission ledger; each accepted cell carries its source point, gauge radius, and membership witness.*
4. *S is the selected disjoint finite subfamily, stored as a closed* $\mathsf{ProbeBundle}$ *over rows from A.*
5. *C is the coverage ledger recording which displayed request windows are covered after S has been read.*
6. *F is the* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *handoff row exposed to compactness, differentiation, maximal-function, or integration consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for radius and tolerance comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for real-valued radius, size, or bound reads.*
9. *H records componentwise* hsame *transport for gauge, cover, candidate, selection, coverage, subcover, readback, seal, replay, provenance, and naming rows.*

10. *K records displayed* Cont *replay from a finite-gauge Vitali request through G, V, A, S, C, F, R, E in that order.*
11. *P is the* Pkg *provenance over* $\mathsf{VitaliCoveringLemma}^{\uparrow}$, $\mathsf{GaugeSpace}^{\uparrow}$, $\mathsf{VitaliCover}^{\uparrow}$, $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VitaliCoveringLemma}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host-level Vitali selection, measure completion, selected countable subfamilies, quotient point equality, selected representatives, or ambient real completeness.

Theorem 7683.2 (Vitali covering lemma NameCert obligations)**.** *For every accepted* $\mathsf{VitaliCoveringLemma}^{\uparrow}$ *carrier $L = (G, V, A, S, C, F, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VitaliCoveringLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite gauge row, Vitali cover row, candidate-admission ledger, selected disjoint finite subfamily, coverage ledger, finite subcover handoff, regular-rational readback, real seal, componentwise transport, displayed replay, package provenance, and local naming. No obligation is discharged by host Vitali selection, countable choice, measure completion, selected infinite subfamilies, quotient point equality, selected representatives, or ambient real completeness.*

Proof. Project the carrier of Definition 7683.1. The projection exposes exactly the gauge row G, cover row V, candidate ledger A, selected disjoint subfamily S, coverage ledger C, finite subcover handoff F, readback row R, seal row E, and structural rows H, K, P, N.

The continuation row K fixes the order of inspection. A covering request first reads the finite gauge row and the Vitali cover row, then admits candidate cells, selects a displayed disjoint finite subfamily, records its coverage ledger, and only then exposes the finite subcover handoff.

Since the carrier contains no row for host-level Vitali selection, countable choice, measure completion, selected infinite subfamilies, quotient point equality, selected representatives, or ambient real completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7683.3 (Vitali covering lemma finite disjoint handoff)**.** *Every compactness, differentiation, maximal-function, or integration consumer of an accepted* $\mathsf{VitaliCoveringLemma}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{GaugeSpace}^{\uparrow},\ \mathsf{VitaliCover}^{\uparrow},\ A,\ S,\ C,\ \mathsf{FiniteSubcoverLedger}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before it receives a covering handoff. Thus the public output is a displayed disjoint finite subfamily together with a finite coverage ledger, not a host theorem, selected infinite subfamily, measure-completion object, quotient point equality, or ambient completeness assertion.

Proof. Begin with a consumer request. By Definition 7683.2, the request must inspect the gauge row, the Vitali cover row, the candidate ledger, and the selected finite subfamily before any coverage ledger is available.

The coverage row C is the only source for the finite subcover handoff F. The regular rational row R and the real seal E read radius and bound comparisons only after this finite route has been displayed.

The structural rows H, K, P, N preserve the same order under transport, replay, provenance, and local naming. Therefore no downstream consumer can bypass the disjoint finite subfamily or read excluded host-level selection and completeness data from the packet. □

Falsifiable prediction: Every accepted $\mathsf{VitaliCoveringLemma}^{\uparrow}$ read displays one GaugeSpaceUp row, one VitaliCoverUp row, one candidate-admission ledger, one selected disjoint finite ProbeBundle subfamily, one coverage ledger, one FiniteSubcoverLedgerUp handoff, one RegSeqRatUp readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: gauge-space, vitali-cover, finite-subcover-ledger, regseqrat, real: $\mathsf{VitaliCoveringLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite gauge requests, cover candidates, disjoint selected subfamilies, finite coverage handoff, rational readback, real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VitaliCoveringLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{VitaliCoveringLemma}^{\uparrow}$ packet is generated from a finite GaugeSpace row, a VitaliCover row, candidate-admission ledger, selected disjoint finite subfamily, coverage ledger, FiniteSubcoverLedger handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VitaliCoveringLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7683.1, Definition 7683.2, and Definition 7683.3 seed the finite gauge-controlled Vitali covering lemma packet over GaugeSpaceUp, VitaliCoverUp, FiniteSubcoverLedgerUp, RegSeqRatUp, RealUp, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VitaliCoveringLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable Vitali selection, measure completion, selected infinite subfamilies, quotient point equality, selected representatives, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-cell admissibility, disjoint finite selection, coverage exactness, finite-subcover handoff, readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.VitaliCoveringLemmaUp.

Chapter 7684

A Concrete Naming Certificate for $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$

Following Riemann and Darboux, the $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$ packet records the finite discontinuity ledger by which a bounded real function on a displayed compact interval is admitted to the Riemann-integrability route. It sits beside the Riemann integral readback of chapter 1526, the Darboux finite-sum surface of chapter 1880, the regulated Darboux ledger of chapter 7650, and the regular rational seal of chapter 301. The packet names only a finite discontinuity ledger and its Darboux cover; it does not import measure-zero sets, a Lebesgue criterion, quotient function equality, selected partitions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7684.1 (Riemann integrability discontinuity carrier)**.** *A* $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, L, C, U, V, G, R, E, H, Q, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-domain row, naming the endpoints, orientation, and compact interval on which the read is made.*
2. *F is the bounded-function window row, exposing only the finitely many function observations requested by the surrounding Riemann and Darboux consumers.*
3. *L is the finite discontinuity ledger, listing the displayed discontinuity names and the local interval windows assigned to them.*
4. *C is the continuity-off-ledger cover row, recording that every cell outside the windows carried by L is read through a displayed continuity certificate.*
5. *U is the upper-window row, carrying the Darboux upper estimates over the ledger windows and the continuity cells.*
6. *V is the lower-window row, carrying the Darboux lower estimates over the same displayed partition cells.*
7. *G is the finite gap row, comparing the upper and lower Darboux rows against a requested dyadic tolerance.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed finite gap budget.*

9. *E is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *handoff row, exposed only after I, F, L, C, U, V, G, R are visible.*
10. *H records componentwise* hsame *transport for the interval, bounded-function window, discontinuity ledger, cover, Darboux rows, gap, readback, handoff, replay, provenance, and local naming rows.*
11. *Q records displayed* Cont *replay from the interval and bounded-function request through the discontinuity ledger, continuity cover, Darboux finite sums, gap readback, and Riemann handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, C, U, V, G, R, E, H, Q, P, N$. It has no coordinate for measure completion, a countable discontinuity set, selected refinement limits, quotient function equality, a host integral operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegrabilityDiscontinuityUp.RiemannIntegrabilityDiscontinuityTasteGate`
`single carrier alignment`

Theorem 7684.2 (Riemann integrability discontinuity NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$ *carrier*

$$D = (I, F, L, C, U, V, G, R, E, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval-domain, bounded-function window, finite discontinuity ledger, continuity-off-ledger cover, upper and lower Darboux rows, gap row, regular readback, Riemann handoff, transport, replay, provenance, and local naming rows;*
2. *ledger finiteness, requiring L to list only finitely many displayed discontinuity windows over the interval row I;*
3. *continuity cover exactness, requiring C to cover every displayed partition cell not assigned to a row of L;*
4. *Darboux gap control, requiring U, V, G to compare lower and upper finite sums only over the displayed ledger windows and continuity cells;*
5. *Riemann handoff nonescape, requiring E to be read only after the finite ledger, cover, Darboux gap, and regular readback rows are visible.*

Thus the packet supplies a finite discontinuity route into Riemann integrability, not a measure-theoretic criterion, quotient-function theorem, selected partition limit, or ambient completeness principle.

Proof. Project the carrier of Definition 7684.1. The only analytic rows are $I, F, L, C, U, V, G, R, E$; the remaining rows H, Q, P, N transport, replay, source, and name that same packet.

First read the discontinuity ledger. The row L is finite and names exactly the discontinuity windows that may contribute exceptional oscillation. The row C covers the complement of those displayed windows by local continuity data, so no hidden discontinuity set is available.

Next read the Darboux comparison. The rows U and V compute upper and lower finite sums over the same displayed cells, and G compares their gap against the requested dyadic budget. The

regular readback row R makes that comparison visible before any Riemann-facing handoff can be consumed.

Finally read E, H, Q, P, N. The handoff E is downstream of the ledger, cover, Darboux gap, and readback rows, while the structural rows preserve only the same finite route. Since the carrier has no coordinate for measure-zero reasoning, quotient functions, selected limits, countable choice, or ambient completeness, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}$ surface. □

Theorem 7684.3 (Riemann integrability finite-discontinuity cover)**.** *Every accepted* RiemannIntegrabilityDiscontinu *packet that exposes a finite discontinuity ledger L and continuity-off-ledger cover C routes any* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer through the finite Darboux comparison*

$$I, F, L, C \longmapsto U, V \longmapsto G \longmapsto R \longmapsto E.$$

Consequently a finite discontinuity ledger over a displayed compact interval is enough BEDC data for the Riemann handoff row, while measure-zero sets, countable exceptional families, selected mesh limits, quotient function equality, host integration, and ambient $\mathsf{Real}^{\uparrow}$ *completeness remain outside the packet.*

Proof. Start from the NameCert obligations of Definition 7684.2. They force the ledger row L and the continuity cover row C to be visible before the Darboux upper and lower rows are read.

The Darboux rows U, V are finite rows over the same displayed interval and partition cells. Their comparison row G is therefore a finite upper-minus-lower budget, and the regular readback R exposes that budget to the Riemann-facing row.

Apply the Riemann handoff nonescape clause. The row E can be inspected only after I, F, L, C, U, V, G, R have been displayed, with H, Q, P, N preserving the same packet. This proves the stated finite cover route and blocks all measure, quotient, choice, host-integration, and ambient-completeness readings. □

Falsifiable prediction: An accepted $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$ read that reaches a $\mathsf{RiemannIntegral}^{\uparrow}$ handoff without displaying the interval row, bounded-function window, finite discontinuity ledger, continuity-off-ledger cover, Darboux upper and lower rows, finite gap row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 7684.1.

Independence witness: riemannintegral, darbouxsum, regulated-darboux-sum, regseqrat: RiemannIntegrabilityD is not bijective with any listed sibling because it jointly carries a finite discontinuity ledger, an off-ledger continuity cover, paired Darboux rows, a finite gap readback, a Riemann handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$ packet is generated from a Real interval-domain row, bounded-function window, finite discontinuity ledger, continuity-off-ledger cover, Darboux upper and lower rows, finite gap row, RegSeqRat readback, RiemannIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7684.1, Definition 7684.2, and Definition 7684.3 seed the finite discontinuity-ledger route into Riemann integrability over interval, Darboux, regular readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-zero criteria, countable exceptional families, quotient function equality, selected refinement limits, host integration, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite ledger finiteness, continuity-off-ledger cover exactness, Darboux gap control, Riemann handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegrabilityDiscontinuityUp.

Chapter 7685

A Concrete Naming Certificate for $\mathsf{FiniteHausdorffMetric}^{\uparrow}$

Following Hausdorff and Bishop–Bridges constructive metric analysis, the $\mathsf{FiniteHausdorffMetric}^{\uparrow}$ packet records the finite separated metric surface on which a finite point family carries displayed pairwise distance rows before any Hausdorff or completion-facing consumer reads it. It sits over the metric carrier of chapter 57, the separated comparison boundary of chapter 1411, the finite-cardinal row of chapter 7673, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite metric certificate; it is not an open-cover compactness theorem, quotient completion object, selected dense subset, countable-choice schedule, or ambient completeness claim for $\mathsf{Real}^{\uparrow}$.

Definition 7685.1 (Finite Hausdorff metric carrier)**.** *A* $\mathsf{FiniteHausdorffMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, E, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, carrying the point classifier and distance observations used by the finite family.*
2. *E is the finite enumeration row naming the admitted points and the finite-cardinal boundary through which the carrier is inspected.*
3. *D is the full pairwise distance ledger on the enumerated points, with every distance read through the same* $\mathsf{Met}^{\uparrow}$ *row.*
4. *S is the separatedness ledger: every zero-distance row in D is routed back to point-classifier identity on the enumerated carrier.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for positive-radius, equality-threshold, and Hausdorff-facing comparisons.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for X, E, D, S, Q, R and the local certificate row.*
8. *C records displayed* Cont *replay of the finite route from the metric source and enumeration through the pairwise ledger, separatedness ledger, dyadic tolerance ledger, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteHausdorffMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{FiniteCardinal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetric}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient point equality, open-cover compactness, selected dense subsets, host finite subsets, countable choice, completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7685.2 (Finite Hausdorff metric NameCert obligations)**.** *For every accepted* $\mathsf{FiniteHausdorffMetric}^{\uparrow}$ *carrier* $F = (X, E, D, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, finite enumeration, pairwise distance ledger, separatedness ledger, dyadic tolerance ledger, Real seal row, componentwise transport, displayed replay, package provenance, and local naming. Every Hausdorff-facing read must expose* E, D, S, Q *before* R *can be consumed, and the ledger exports no quotient equality, open-cover compactness, selected dense subset, host finite subset, countable-choice schedule, completion class, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 7685.1. The metric row is X, the finite carrier row is E, and the metric data are exactly the pairwise ledger D together with the separatedness ledger S and dyadic tolerance ledger Q.

The finite enumeration is prerequisite data. A Hausdorff-facing or completion-facing consumer can read a point only after it appears in E, and can read a distance only after the corresponding pairwise row in D has been displayed.

Separatedness is internal to the same finite route. The row S routes zero-distance witnesses back through the point classifier of X, while Q records the dyadic tolerance comparisons used by positive-radius and Hausdorff bounds. Thus the route cannot be replaced by quotient equality or a host finite subset.

Finally read the Real seal and structural rows. The $\mathsf{Real}^{\uparrow}$ seal R exposes only the already displayed finite metric data, while H, C, P, N transport, replay, source, and name the same packet. □

Theorem 7685.3 (Finite separated metric Hausdorff handoff)**.** *Every metric, Hausdorff-metric, or completion-facing consumer of an accepted* $\mathsf{FiniteHausdorffMetric}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{Met}^{\uparrow},\ E,\ D,\ S \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives enumerated points, pairwise distance rows, zero-distance classifier return, dyadic tolerance comparisons, and the terminal Real seal as displayed rows. It receives no quotient completion object, selected dense subset, open-cover compactness theorem, countable-choice schedule, host finite subset, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer of the accepted packet. Carrier admission forces the entry point to be the displayed $\mathsf{Met}^{\uparrow}$ row X and the finite enumeration row E.

Open the pairwise ledger next. The row D supplies all distance reads on the enumerated carrier, and S supplies the separatedness return from zero distance to classifier identity on that same finite carrier.

The dyadic and Real rows are downstream. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger Q records the comparison budget used by Hausdorff-facing reads, and the $\mathsf{Real}^{\uparrow}$ seal R is reached only after the finite metric and separatedness rows have already been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetric}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteHausdorffMetric}^{\uparrow}}, \mathsf{seed})$
Scope closed: Definition 7685.1, Definition 7685.2, and Definition 7685.3 seed the finite separated metric route over displayed MetricSpace, separatedness, finite enumeration, dyadic tolerance, and Real seal rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteHausdorffMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality, open-cover compactness, selected dense subsets, host finite subsets, countable choice, completion classes, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, pairwise distance ledger exactness, separatedness return, dyadic tolerance transport, Real seal nonescape, replay, provenance, and local naming for BEDC.Derived.FiniteHausdorffMetricUp.

Constructive story: A FiniteHausdorffMetric$^{\uparrow}$ packet is generated from a MetricSpaceUp row, finite enumeration, pairwise distance ledger, separatedness ledger, DyadicRatCoreUp tolerance row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 7686

A Concrete Naming Certificate for $\mathsf{AsymptoticCone}^{\uparrow}$

Following Gromov's asymptotic-cone large-scale metric construction, the $\mathsf{AsymptoticCone}^{\uparrow}$ packet records the finite scaling-window surface by which a pointed metric family is read at increasing scales. It replaces nonprincipal ultrafilters and selected limiting cones by explicit scale ledgers, finite metric-ball windows, dyadic tolerance rows, regular rational readback, and a terminal Real seal. The packet sits beside the metric carrier of chapter 57, the ball-window surface of chapter 1450, the dyadic ledger of chapter 291, the finite observation discipline of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38.

Definition 7686.1 (AsymptoticCone carrier)**.** *An* $\mathsf{AsymptoticCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, o, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, naming only the displayed distance comparisons available to the packet.*
2. *o is the basepoint row attached to the same metric source.*
3. *S is the scaling ledger, a finite list of scale requests and comparison budgets.*
4. *B is the* $\mathsf{MetricBall}^{\uparrow}$ *window row, recording which bounded metric balls are inspected after rescaling.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the finite window schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare rescaled distances.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distance families.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, B, W, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the metric, basepoint, scale, ball-window, stream, dyadic, regular-rational, Real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a scaling-window request through the same rows.*
11. *P is the* Pkg *provenance over* $\mathsf{AsymptoticCone}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{AsymptoticCone}^\uparrow}$ *row.*
The carrier has no coordinate for a nonprincipal ultrafilter, a selected metric limit, a quotient cone, host real equality, or ambient completeness of the source metric or $\mathsf{Real}^\uparrow$*.*

Theorem 7686.2 (AsymptoticCone NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticCone}^\uparrow$ *carrier*

$$A = (M, o, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AsymptoticCone}^\uparrow}$ *surface consists exactly of metric-source admission through M, o, scaling-window exposure through S, bounded ball-window exposure through B, finite stream observation through W, dyadic tolerance comparison through D, regular rational readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7686.1. The metric and basepoint rows M, o are the only source rows, and the scaling ledger S names exactly the scale requests visible to the packet.

The bounded windows are displayed before any Real-facing read. The row B supplies the metric balls, W lists the finite observation schedule, D supplies dyadic tolerance comparisons, and R supplies the regular rational readback used by the terminal seal E.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Since no coordinate contains an ultrafilter, selected limit, quotient cone, host equality, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 7686.3 (AsymptoticCone scaling window)**.** *Every coarse-metric consumer of an accepted* $\mathsf{AsymptoticCone}^\uparrow$ *carrier must read the finite route*

$$\mathsf{Met}^\uparrow,\ \mathsf{MetricBall}^\uparrow,\ \mathsf{DyadicRatCore}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow$$

through the scale ledger S before treating a rescaled distance family as cone-facing data. The route exposes only displayed scale requests, bounded ball windows, finite stream observations, dyadic tolerances, regular rational readbacks, and terminal Real seals; it exports no nonprincipal ultrafilter, selected limiting point, quotient cone equality, host real equality, or ambient completeness principle.

Proof. Start from the NameCert obligations of Definition 7686.2. They force every accepted read through the metric source, basepoint, scaling ledger, ball-window row, finite observation row, dyadic tolerance row, regular readback row, Real seal, and structural rows.

The scale ledger S controls the order of the read. A consumer can inspect a rescaled distance only after the matching $\mathsf{MetricBall}^\uparrow$ window has been displayed and the dyadic tolerance row has attached a finite comparison budget to that window.

The StreamName, RegSeqRat, and Real rows then provide readback rather than a completed cone object. Because the carrier has no ultrafilter, selected-limit, quotient, host-equality, or completeness coordinate, the scaling-window route cannot export any of those endpoints. □

Falsifiable prediction: Every accepted $\mathsf{AsymptoticCone}^\uparrow$ read exposes MetricUp source data, a basepoint, finite scaling ledgers, MetricBallUp windows, StreamName observations, DyadicRatCore tolerances, RegSeqRat readbacks, RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any cone-facing consumer reads the packet.

Independence witness: metric, metricball, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{AsymptoticCone}^\uparrow$ because this packet alone binds a pointed metric source, finite scaling ledgers, bounded ball windows, dyadic tolerance budgets, regular rational readbacks, terminal Real seals, transport, replay, provenance, and local naming as one large-scale metric surface.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticCone}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticCone}^{\uparrow}$ packet is generated from a MetricUp source row, a basepoint row, finite scaling ledgers, MetricBallUp bounded windows, StreamName finite observations, DyadicRatCore tolerance rows, RegSeqRat readback rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7686.1, Definition 7686.2, and Definition 7686.3 seed the finite scaling-window metric packet over displayed metric, ball-window, dyadic, stream, regular-rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close nonprincipal ultrafilters, selected limits, quotient cones, host real equality, ambient metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scale-ledger exactness, bounded-window coverage, dyadic tolerance preservation, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AsymptoticConeUp.

Chapter 7687

A Concrete Naming Certificate for $\mathsf{RiemannTaggedMesh}^{\uparrow}$

Following Bishop and Bridges on constructive Riemann integration, the $\mathsf{RiemannTaggedMesh}^{\uparrow}$ packet records the finite tagged-mesh surface that sits between displayed interval partitions, tagged cells, Riemann sums, Darboux brackets, and the terminal Riemann-integral handoff. It reads the tagged real partition of chapter 4296, the generic tagged-partition carrier of chapter 7449, the finite Riemann-sum row of chapter 1615, the Darboux bracket row of chapter 1880, the integration endpoint of chapter 1526, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names only a displayed mesh-refinement interface; it does not select a limiting partition family, identify integrands by quotient equality, import measure integration, use countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7687.1 (Riemann tagged mesh carrier)**.** *A* $\mathsf{RiemannTaggedMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, T, U, S, B, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row, exposing the two* $\mathsf{Real}^{\uparrow}$ *endpoints and their displayed orientation.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row, listing the finite cells, endpoint reads, tags, and tag-containment witnesses.*
3. *U is the* $\mathsf{TaggedPartition}^{\uparrow}$ *refinement row, recording the finite subdivision comparison between an incoming tagged partition and the carried mesh.*
4. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row, carrying the finite tagged-sum reads over the cells of T.*
5. *B is the* $\mathsf{DarbouxSum}^{\uparrow}$ *bracket row, carrying the lower and upper finite-sum comparisons over the same displayed mesh.*
6. *G is the gap ledger, comparing the Riemann-sum row and the Darboux bracket row against the displayed mesh refinement and tolerance request.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite gap and sum data.*
8. *E is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *handoff row, exposed only after I, T, U, S, B, G, R are visible.*
9. *H records componentwise* hsame *transport for interval, tagged partition, refinement, sum, bracket, gap, readback, handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the finite route*

$$I, T, U \longmapsto S, B \longmapsto G \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{RiemannTaggedMesh}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{TaggedPartition}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannTaggedMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, U, S, B, G, R, E, H, C, P, N$. It has no coordinate for a selected refinement sequence, a host function space, quotient equality of integrands, measure integration, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7687.2 (Riemann tagged mesh NameCert obligations)**.** *For every accepted* $\mathsf{RiemannTaggedMesh}^{\uparrow}$ *carrier $M = (I, T, U, S, B, G, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannTaggedMesh}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval, tagged real partition, tagged refinement, finite Riemann-sum, Darboux bracket, gap ledger, regular rational readback, Riemann-integral handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every accepted integration-facing read follows the finite route*

$$I, T, U \longmapsto S, B \longmapsto G \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Project the carrier of Definition 7687.1. Its analytic rows are exactly the interval row, tagged real partition, tagged refinement row, finite sum row, Darboux bracket row, gap ledger, regular readback, and integral handoff.

The mesh row is not detached from the partition rows. The row U compares the incoming tagged partition with the carried finite mesh, while T supplies the cells, endpoint reads, and tags on which S and B are evaluated.

The rows S and B are then read over the same displayed cells. The gap row G compares those finite rows against the requested mesh tolerance before R can expose regular rational readback.

Finally E, H, C, P, N preserve the same route toward the Riemann-integral handoff. Since no carrier coordinate stores a selected refinement sequence, host function space, quotient integrand equality, measure integral, choice principle, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 7687.3 (Riemann tagged mesh refinement handoff)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{RiemannTaggedMesh}^{\uparrow}$ *packet must read the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row, the* $\mathsf{TaggedPartition}^{\uparrow}$ *refinement row, the* $\mathsf{RiemannSum}^{\uparrow}$ *finite-sum row, the* $\mathsf{DarbouxSum}^{\uparrow}$ *bracket row, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the handoff row E is public. Thus mesh refinement is exposed as finite tagged data and not as a hidden sequence of partitions.*

Proof. Start with the obligation route of Definition 7687.2. It places the handoff row E downstream from I, T, U, S, B, G, R.

The partition data comes first. The tagged real partition T provides cells, tags, and endpoint reads, while the tagged refinement row U records how the carried mesh compares with the incoming finite partition.

The analytic comparison is finite. The Riemann-sum row S and Darboux bracket row B are both read over the displayed mesh, and G records their finite gap before R presents the regular rational readback.

Therefore any Riemann-integral consumer that reaches E has already inspected the finite tagged-mesh interface. A hidden refinement sequence or selected limiting partition is not available in the carrier. □

Theorem 7687.4 (Riemann tagged mesh nonescape)**.** *The* $\mathsf{RiemannTaggedMesh}^{\uparrow}$ *packet exports no selected partition family, host function space, quotient equality of integrands, measure-space integration, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public read remains inside the displayed interval, tagged partition, refinement, Riemann-sum, Darboux-bracket, gap, regular-readback, integral-handoff, transport, replay, provenance, and local naming rows.*

Proof. Compare the refused objects with the carrier inventory of Definition 7687.1. None is one of $I, T, U, S, B, G, R, E, H, C, P, N$.

The positive route is supplied by Definition 7687.3. It keeps every integration-facing read behind the finite tagged partition, refinement, sum, bracket, gap, and readback rows before the handoff is consumed.

Transport through H, replay through C, provenance through P, and local naming through N do not add new analytic coordinates. The packet therefore cannot escape to host partitions, quotient functions, measure integration, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RiemannTaggedMesh}^{\uparrow}$ read that reaches a $\mathsf{RiemannIntegral}^{\uparrow}$ handoff exposes $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{TaggedPartition}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before the handoff is public.

Independence witness: tagged-real-partition, tagged-partition, riemannsum, darboux-sum, riemannintegral, regseqrat, real: $\mathsf{RiemannTaggedMesh}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries interval endpoints, tagged cells, refinement comparison, finite Riemann sums, Darboux brackets, gap readback, integral handoff, transport, replay, provenance, and local naming as one mesh-refinement packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannTaggedMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannTaggedMesh}^{\uparrow}$ packet is generated from interval, TaggedRealPartition, TaggedPartition refinement, RiemannSum, DarbouxSum, gap, RegSeqRat readback, RiemannIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannTaggedMesh}^{\uparrow}}$, seed)

Scope closed: Definition 7687.1, Definition 7687.2, Definition 7687.3, and Definition 7687.4 seed the finite tagged-mesh refinement route over partition, sum, bracket, readback, handoff, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannTaggedMesh}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected partition sequences, host function spaces, quotient integrand equality, measure integration, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition exposure, refinement exactness, Riemann-sum exposure, Darboux-bracket exposure, gap readback, Riemann-integral handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannTaggedMeshUp.

Chapter 7688

A Concrete Naming Certificate for $\mathsf{PhysicalTowerRealization}^{\uparrow}$

The $\mathsf{PhysicalTowerRealization}^{\uparrow}$ packet realizes the approximation-tower and ledgered-compression route of section 9414.20 as finite NameCert data. It turns approximation levels, compression arrows, visible endpoints, composite tower ledgers, and endpoint descent checks into one displayed carrier, sitting beside the tower-compression packet of chapter 1142, the regular rational readback of chapter 301, the dyadic tolerance surface of chapter 291, the calculus consumer of chapter 3632, and the terminal real-name seal of chapter 38. The packet records a finite physical approximation tower as a BEDC-native object; it does not identify the physical source with the endpoint, erase hidden ledgers, choose a preferred model, add host physics, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7688.1 (Physical tower realization carrier)**.** *A* $\mathsf{PhysicalTowerRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, L, A, E, D, G, R, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row, naming the finite physical observation history or finite source window from which the tower is read.*
2. *L is the level row, listing the displayed approximation levels and the classifier bundle visible at each level.*
3. *A is the compression-arrow row, recording source, pattern, classifier, stability, and local ledger fields for each displayed arrow.*
4. *E is the endpoint row, naming the final visible object without identifying it with the source row.*
5. *D is the descent-audit row, recording which endpoint operations descend through the displayed compression classifiers.*
6. *G is the composite-ledger row, pulling every local ledger in A back to the source row S.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for quantitative tolerance, prediction, or comparison entries carried by the tower.*
8. *V is the visible-result row, exposing only the endpoint, descent, ledger, and readback data that the tower makes public.*
9. *H records componentwise* hsame *transport for source, level, arrow, endpoint, descent, ledger, readback, result, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the finite route*

$$S, L, A \longmapsto E, D, G \longmapsto R \longmapsto V.$$

11. *P is the* Pkg *provenance over* PhysicalTowerRealization$^{\uparrow}$, RealityConstrainedTowerCompression$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Calculus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{PhysicalTowerRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, A, E, D, G, R, V, H, C, P, N$. It has no coordinate for endpoint-source identity, hidden-ledger erasure, a selected best model, host physical law, quotient equality of models, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7688.2 (Physical tower realization NameCert obligations)**.** *For every accepted* PhysicalTowerRealization$^{\uparrow}$ *carrier $T = (S, L, A, E, D, G, R, V, H, C, P, N)$, the local* NameCert$_{\text{PhysicalTowerRealization}^{\uparrow}}$ *surface consists exactly of carrier admission for the source, level, compression-arrow, endpoint, descent-audit, composite-ledger, regular readback, visible-result, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every public tower read follows the finite route*

$$S, L, A \longmapsto E, D, G \longmapsto R \longmapsto V$$

together with H, C, P, N.

Proof. Project the carrier of Definition 7688.1. The source row, level row, and compression-arrow row provide the displayed tower skeleton.

The endpoint row E is downstream from that skeleton. It names the visible endpoint but does not identify it with S, because the arrow row A keeps the classifiers and local ledgers visible.

The descent-audit row D and composite-ledger row G are the certification rows. They record which endpoint operations descend through the tower and which source-side residues remain after all compression arrows have been composed.

The readback row R and visible-result row V expose only the finite data already carried by S, L, A, E, D, G. Transport, replay, provenance, and local naming preserve that route, so the listed rows exhaust the local NameCert surface. □

Theorem 7688.3 (Physical tower realization vision handoff)**.** *The finite definitions of approximation level, compression arrow, physical approximation tower, visible endpoint, and tower ledger in section 9414.20 are realized by the rows L, A, S, E, G of any accepted* PhysicalTowerRealization$^{\uparrow}$ *carrier. Endpoint claims are public only after the descent-audit row D, composite-ledger row G, regular readback row R, and visible-result row V have been displayed.*

Proof. Read the vision-side tower vocabulary through the carrier rows. Approximation levels are recorded by L, compression arrows by A, the physical source by S, the visible endpoint by E, and the tower ledger by G.

The endpoint row is not self-certifying. The descent-audit row D records which endpoint operations are licensed by the compression classifiers, and the composite-ledger row G keeps the hidden source residues visible.

The public result is then exposed through R and V, with H, C, P, N preserving the same route. Hence the vision route becomes a finite NameCert packet rather than a prose-only modelling claim. □

Theorem 7688.4 (Physical tower realization nonescape)**.** *The* $\mathsf{PhysicalTowerRealization}^{\uparrow}$ *packet exports no endpoint-source identity, no hidden-ledger erasure, no selected preferred model, no host physical law, no quotient equality of models, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public read remains inside the displayed source, level, compression-arrow, endpoint, descent-audit, composite-ledger, regular-readback, visible-result, transport, replay, provenance, and local naming rows.*

Proof. Compare the refused objects with the carrier rows of Definition 7688.1. None is a coordinate of $S, L, A, E, D, G, R, V, H, C, P, N$.

The positive public route is supplied by Definition 7688.3. It keeps endpoint claims downstream from the source, level, arrow, descent-audit, composite-ledger, and readback rows.

The structural rows H, C, P, N preserve only that finite route. Therefore a consumer cannot obtain an erased ledger, privileged model selector, host physics law, quotient model equality, or ambient real-completeness principle from the packet. □

Falsifiable prediction: An accepted $\mathsf{PhysicalTowerRealization}^{\uparrow}$ read that exposes an endpoint result without first displaying source, level, compression-arrow, descent-audit, composite-ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows refutes the carrier of Definition 7688.1.

Independence witness: reality-constrained-tower-compression, compression-ledger-functor, compression-descent-audit, towerendpointreflection, calculus: $\mathsf{PhysicalTowerRealization}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries a physical source row, approximation levels, compression arrows, endpoint, descent audit, composite ledger, quantitative readback, visible result, transport, replay, provenance, and local naming in one finite packet.

The later Lean TasteGate file for this route must include explicit $\mathsf{Nontrivial}\ \mathsf{PhysicalTowerRealization}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PhysicalTowerRealization}^{\uparrow}$ instances whose fields project $S, L, A, E, D, G, R, V, H, C, P, N$ separately.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalTowerRealization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PhysicalTowerRealization}^{\uparrow}$ packet is generated from finite source, approximation-level, compression-arrow, endpoint, descent-audit, composite-ledger, RegSeqRat readback, visible-result, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalTowerRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7688.1, Definition 7688.2, Definition 7688.3, and Definition 7688.4 seed the finite approximation-tower realization over source, level, compression-arrow, endpoint, descent, ledger, readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalTowerRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close endpoint-source identity, hidden-ledger erasure, preferred-model selection, host physical law, quotient model equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximation-level exposure, compression-arrow ledger exactness, endpoint nonidentity, descent-audit routing, composite-ledger readback, visible-result nonescape, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.PhysicalTowerRealizationU

Chapter 7689

A Concrete Naming Certificate for $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ packet records a bounded map on regular rational names by exposing finite source windows, regular-rational input and output readbacks, a dyadic bound ledger, and the terminal real-name seal before any completion-facing consumer reads the map. It sits over the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the graph replay route of chapter 130. The packet is a bounded regular-sequence map certificate; it is not an ambient function-space object, quotient equality of real names, selected modulus of continuity, countable-choice schedule, or global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7689.1 (Bounded RegSeqRat map carrier)**.** *A* $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, I, O, B, T, R, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, listing exactly the input coordinates inspected before the map graph is replayed.*
2. *I is the* $\mathsf{RegSeqRat}^{\uparrow}$ *input readback row attached to the displayed source windows.*
3. *O is the* $\mathsf{RegSeqRat}^{\uparrow}$ *output readback row produced from the same inspected windows.*
4. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound ledger, carrying the displayed dyadic radius that bounds every output row exposed by O.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the precision requests used to compare input windows, output windows, and the bound row.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, I, O, B, T have been displayed.*
7. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph replay row, exposing the finite graph route from source windows through regular-rational input and output readbacks.*
8. *H records componentwise* hsame *transport for source windows, input readback, output readback, bound ledger, tolerance ledger, real seal, graph replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from* $\mathsf{StreamName}^{\uparrow}$ *windows through* $\mathsf{RegSeqRat}^{\uparrow}$ *input and output readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound and tolerance ledgers, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegSeqRatMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient function space, quotient equality of real names, an externally selected continuity modulus, countable choice, host boundedness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7689.2 (Bounded RegSeqRat map NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ *carrier $M = (S, I, O, B, T, R, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegSeqRatMap}^{\uparrow}}$ *surface consists exactly of carrier admission for source windows, regular-rational input readback, regular-rational output readback, dyadic bound ledger, dyadic tolerance ledger, terminal real seal, ContinuousMap graph replay, componentwise transport, displayed replay, package provenance, and local naming. Every map-facing read must expose S, I, O, B, T, G before R can be consumed, and the ledger exports no ambient function space, quotient real-name equality, externally selected modulus of continuity, countable-choice schedule, host boundedness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 7689.1. The source row is S, the input and output regular-rational readbacks are I and O, and the dyadic boundedness data are exactly the rows B and T.

The graph route is finite. A map-facing consumer can replay the $\mathsf{ContinuousMap}^{\uparrow}$ row G only after the inspected source windows and both $\mathsf{RegSeqRat}^{\uparrow}$ readbacks have been exposed. Thus the graph is tied to displayed windows rather than to a hidden function-space element.

The terminal seal is downstream. The $\mathsf{Real}^{\uparrow}$ row R is reached only after the bound and tolerance ledgers have been read, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient equality, an external continuity modulus, a choice schedule, host boundedness, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7689.3 (Bounded RegSeqRat map uniform bound handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow}_{\mathrm{in}},\quad \mathsf{RegSeqRat}^{\uparrow}_{\mathrm{out}},\quad \mathsf{DyadicRatCore}^{\uparrow}_{\mathrm{bound}},\quad \mathsf{DyadicRatCore}^{\uparrow}_{\mathrm{tol}},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{Re}$$

The bound handoff is the displayed dyadic ledger B, T over the same source windows that produce the output readback. It is not an ambient bounded-function assumption, quotient equality of real names, selected continuity modulus, countable-choice schedule, or global completeness principle.

Proof. Start with a consumer of the terminal seal or completion-facing route. Carrier admission forces the read to begin with the finite source-window row S and the input regular-rational readback I.

Open the map graph next. The $\mathsf{ContinuousMap}^{\uparrow}$ row G replays the route from S and I to the output readback O, so the output cannot be inspected independently of the displayed source windows.

The boundedness handoff is then forced by the dyadic rows. The ledger B supplies the displayed bound and T supplies the tolerance comparisons needed to read that bound against O. Only after these rows are exposed may the consumer reach the $\mathsf{Real}^{\uparrow}$ seal R. □

Falsifiable prediction: An accepted $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ consumer is refuted if it reaches a RealUp seal or completion-facing output without first displaying the StreamName source windows, RegSeqRat input readback, RegSeqRat output readback, DyadicRatCore bound and tolerance ledgers, ContinuousMap graph replay, transport, displayed replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, continuousmap: no listed sibling supplies the full bounded regular-sequence map packet, because only $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ binds source windows, input and output regular-rational readbacks, dyadic bound and tolerance ledgers, graph replay, Real seal, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegSeqRatMap}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegSeqRatMap}^{\uparrow}$ packet is generated from StreamName source windows, RegSeqRat input and output readbacks, DyadicRatCore bound and tolerance ledgers, a ContinuousMap graph replay row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegSeqRatMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7689.1, Definition 7689.2, and Definition 7689.3 seed the finite bounded-map route over StreamName, RegSeqRat, DyadicRatCore, ContinuousMap, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegSeqRatMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function spaces, quotient real-name equality, selected continuity moduli, countable choice, host boundedness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window admission, input and output RegSeqRat readback, dyadic bound and tolerance exactness, ContinuousMap graph replay, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BoundedRegSeqRatMapUp.

Chapter 7690

A Concrete Naming Certificate for $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$

Following Schoenflies' planar extension theorem and Jordan's simple closed curve boundary theorem, the $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$ packet records only the finite window in which a displayed Jordan curve, a bounded side ledger, and a compact planar support can be consumed as local planar-domain data. It sits after the side-classification surface of chapter 4365, the degree obstruction rows of chapter 3745, the compact-metric source of chapter 129, the topology carrier of chapter 88, and the circle source of chapter 58. The packet names a finite Jordan-window certificate; it does not assert the full classical Schoenflies extension theorem, a global homeomorphism of the plane, selected complementary components, quotient boundary equality, or open-cover compactness.

Definition 7690.1 (Schoenflies finite Jordan-window carrier)**.** *A* $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (J, D, K, T, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{JordanCurve}^{\uparrow}$ *row naming the simple closed curve, its circle source, polygonal subdivision, winding ledger, orientation row, and side-classification boundary.*
2. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *row used only as a finite obstruction ledger for the displayed boundary orientation.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *support window in which the curve, side ledger, and finite planar cells are inspected.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row recording the finite open and closed windows visible to the local domain read.*
5. *S is the* $\mathsf{S}^{1\uparrow}$ *source-boundary row whose parameter windows are consumed by J.*
6. *A is the finite annular collar ledger around the displayed curve, including inside-window, outside-window, and boundary-window rows.*
7. *E is the finite extension-obligation ledger; each row records a local cell correspondence required before a Schoenflies-style handoff may be read.*

8. *H records componentwise* hsame *transport for the Jordan curve, degree, compact-metric, topology, circle, collar, extension-obligation, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from the circle source through the Jordan curve row, the collar ledger, the compact support, and the local topology window before any extension-obligation row is exposed.*
10. *P is the* Pkg *provenance over* $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$, $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host planar homeomorphism, selected complementary component, quotient boundary equality, open-cover compactness theorem, ambient plane completion, or unrestricted Schoenflies extension.

Theorem 7690.2 (Schoenflies finite Jordan-window NameCert obligations)**.** *For every accepted* $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$ *carrier* $W = (J, D, K, T, S, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SchoenfliesFiniteJordanW}}$ *surface consists exactly of carrier admission for the Jordan curve row, Brouwer-degree obstruction row, compact-metric support window, topology neighbourhood row, circle source row, finite annular collar ledger, finite extension-obligation ledger, componentwise transport, displayed replay, package provenance, and local naming. Every planar-domain consumer must read the* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, *and* $\mathsf{S}^{1\uparrow}$ *rows before it can inspect the collar or extension-obligation rows, and no obligation is discharged by a global Schoenflies theorem.*

Proof. Project the carrier of Definition 7690.1. The projection exposes exactly the Jordan curve row J, degree row D, compact support K, topology window T, circle source S, collar ledger A, extension-obligation ledger E, and structural rows H, C, P, N.

The continuation row C fixes the inspection order. A local planar-domain read starts from the $\mathsf{S}^{1\uparrow}$ source boundary, consumes the $\mathsf{JordanCurve}^{\uparrow}$ side-classification route, reads the $\mathsf{BrouwerDegree}^{\uparrow}$ obstruction ledger and $\mathsf{CompactMetric}^{\uparrow}$ support, and only then inspects the $\mathsf{Topology}^{\uparrow}$ neighbourhood windows, collar rows, and finite extension obligations.

Since the carrier contains no row for a host planar homeomorphism, selected complementary component, quotient boundary equality, open-cover compactness theorem, ambient plane completion, or unrestricted Schoenflies extension, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$ read displays one JordanCurveUp row, one BrouwerDegreeUp obstruction row, one CompactMetricUp support window, one TopologyUp neighbourhood row, one SOneUp source-boundary row, one finite annular collar ledger, one finite extension-obligation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: jordan-curve, brouwer-degree, compactmetric, topology, s1: SchoenfliesFiniteJordanWind is not bijective to any listed sibling because it jointly carries side-classified Jordan boundary data, degree obstruction data, compact support, topology windows, circle source windows, collar rows, finite extension obligations, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchoenfliesFiniteJordanWindow}^{\uparrow}$ packet is generated from a JordanCurve row, BrouwerDegree obstruction row, CompactMetric support window, Topology neighbourhood row, SOne source-boundary row, finite annular collar ledger, finite extension-obligation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SchoenfliesFiniteJordanWindow}\uparrow}$, seed)

Scope closed: Definition 7690.1 and Definition 7690.2 seed the finite Jordan-window packet over JordanCurveUp, BrouwerDegreeUp, CompactMetricUp, TopologyUp, SOneUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{SchoenfliesFiniteJordanWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the full classical Schoenflies extension theorem, a global planar homeomorphism, selected complementary components, quotient boundary equality, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side-classified Jordan boundary routing, degree-obstruction stability, compact support, topology-window exactness, circle-source readback, collar ledger exactness, finite extension obligations, transport, replay, provenance, and local naming for BEDC.Derived.SchoenfliesFiniteJordanWindowUp.

Chapter 7691

A Concrete Naming Certificate for $\mathsf{RationalIntervalTree}^{\uparrow}$

Following Bishop–Bridges constructive real analysis and Farey interval-refinement practice, the $\mathsf{RationalIntervalTree}^{\uparrow}$ packet records a finite tree of rational brackets whose boundary rows refine an interval without selecting a host real representative. It sits between the dyadic arithmetic surface of chapter 291, the finite observation windows of chapter 126, the regular-rational readback route of chapter 301, the terminal real-name seal of chapter 38, the located interval boundary of chapter 1709, and the Farey tree source of chapter 1704. The packet names only finite rational interval nodes, bisection rows, tolerance ledgers, stream windows, regular-rational readback, Real sealing, transport, replay, provenance, and local naming.

Definition 7691.1 (Rational interval-tree carrier)**.** *A* $\mathsf{RationalIntervalTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, L, U, M, B, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite address row naming the inspected refinement path.*
2. *L and U are the lower and upper rational boundary rows for the current node.*
3. *M is the displayed rational midpoint or Farey mediant row used to split the current bracket.*
4. *B is the branch ledger recording which child interval is retained at each inspected refinement step.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint width, midpoint comparison, and child-width budget.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which consumers inspect the finite bracket trace.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational endpoints and displayed tolerance comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the interval tree, stream window, readback, and dyadic tolerance rows have been displayed.*
9. *H records componentwise* hsame *transport for address, endpoint, midpoint, branch, tolerance, stream, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an interval request through finite refinement, branch selection, tolerance readback, stream exposure, and Real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It contains no selected real representative, quotient-real equality, host trichotomy, unrestricted nested-interval principle, countable-choice selector, or ambient completeness witness.

Theorem 7691.2 (Rational interval-tree NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalTree}^{\uparrow}$ *carrier* $T = (A, L, U, M, B, Q, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite address row, rational boundary rows, midpoint or mediant row, branch ledger, dyadic tolerance ledger, stream window, regular-rational readback, Real seal, componentwise transport, displayed replay, package provenance, and local naming. No obligation is discharged by selected real representatives, quotient equality, host trichotomy, ambient completeness, or an unlisted infinite refinement object.*

Proof. Project the accepted carrier. The analytic coordinates are $A, L, U, M, B, Q, S, R, E$, and the structural coordinates are H, C, P, N.

The interval-facing rows determine the local obligation surface. The address row names the finite path, the boundary rows name the current bracket, the midpoint or mediant row names the split, and the branch ledger records the retained child.

The readback rows determine every downstream approximation. The dyadic tolerance ledger supplies the width and comparison budgets; the stream row exposes finite windows; the regular-rational row reads the endpoints; and the Real seal is reached only after those rows have been displayed.

Transport, replay, provenance, and naming preserve the same inventory. Since the carrier has no coordinate for selected representatives, quotient equality, trichotomy, ambient completeness, or an unlisted infinite refinement object, the listed rows exhaust the local certificate. □

Theorem 7691.3 (Rational interval-tree bisection window)**.** *Every finite bisection or Farey-refinement read of an accepted* $\mathsf{RationalIntervalTree}^{\uparrow}$ *carrier factors through the displayed route*

$$A, L, U, M, B, Q, S, R, E, H, C, P, N.$$

Before any Real-facing approximation is public, the consumer has already read the boundary pair L, U, the split row M, the retained-child ledger B, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row Q, the* $\mathsf{StreamName}^{\uparrow}$ *window S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The route exports no selected limit, quotient equality, host order trichotomy, or ambient Real completeness.*

Proof. Induct over the finite address row A. At the root window, the packet displays the initial rational boundary pair L, U together with its tolerance ledger Q.

For a successor window, read the split row M and the branch ledger B. The retained child interval is one of the two displayed rational brackets, and the dyadic row records the corresponding width and comparison budget.

The public approximation boundary is then forced by the readback rows. The stream window S exposes the inspected finite trace, the regular-rational row R reads the rational endpoints, and the Real seal E is downstream of both.

Thus every finite bisection or Farey-refinement consumer sees the displayed interval-tree window before it can read a Real-facing approximation, and the excluded host-level objects have no carrier coordinate from which they could be recovered. □

Falsifiable prediction: Every accepted RationalIntervalTreeUp packet exposes a finite rational boundary pair, one split row, one retained-child ledger, one dyadic tolerance row, one StreamName window, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-real-interval, sternbrocot: $\mathsf{RationalIntervalTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite interval-tree addresses, rational boundary rows, split rows, retained-child ledgers, dyadic tolerances, finite observation windows, regular-rational readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalTree}\uparrow}$.

Constructive story: A $\mathsf{RationalIntervalTree}^{\uparrow}$ packet is generated from a finite address row, rational lower and upper boundary rows, a midpoint or Farey-mediant split row, retained-child ledger, DyadicRatCore tolerance row, StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalTree}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7691.1, Definition 7691.2, and Definition 7691.3 seed the finite rational interval-refinement tree packet between dyadic interval arithmetic, regular-rational readback, located intervals, Stern–Brocot refinement, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalTree}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected real representatives, quotient-real equality, host trichotomy, unrestricted nested-interval completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, split exactness, branch retention, dyadic tolerance readback, stream-window exposure, regular-rational readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RationalIntervalTreeUp.

Chapter 7692

A Concrete Naming Certificate for $\mathsf{TakagiFunction}^{\uparrow}$

Following Takagi's continuous nowhere-differentiable dyadic tent-map example, the $\mathsf{TakagiFunction}^{\uparrow}$ packet records the finite BEDC carrier needed to read a dyadic polygonal stage, its uniform tail budget, and the terminal real-function seal without importing host real equality. It sits over the dyadic arithmetic of chapter 291, the dyadic subdivision surface of chapter 1863, the uniform-limit route of chapter 463, the real-name seal of chapter 38, and the graph replay discipline of chapter 130. The packet is a constructive real-analysis boundary object for the finite dyadic-to-Real route; it is not a completed differentiability theory, quotient equality of functions, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7692.1 (Takagi function carrier)**.** *A* $\mathsf{TakagiFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, A, B, U, R, G, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, carrying dyadic endpoints, midpoint distances, powers of two, and comparison slack used by every displayed tent-map stage.*
2. *S is the* $\mathsf{DyadicPartition}^{\uparrow}$ *subdivision row, listing the finite dyadic cells on which the carried stage is affine.*
3. *A is the stage-amplitude row, recording the displayed tent contribution*

   $$2^{-k}\tau(2^k x)$$

   only through dyadic rational reads on the cells supplied by S.
4. *B is the finite partial-sum ledger, adding the displayed stage contributions through a bounded dyadic sum over the carried stage window.*
5. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *tail row, carrying the displayed dyadic estimate that the remaining stages after the window have total size at most the requested tolerance.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after D, S, A, B, U have been displayed.*
7. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph replay row, exposing the finite polygonal route from dyadic input windows to real-name output windows.*

8. *H records componentwise* hsame *transport for dyadic arithmetic, subdivision, stage amplitudes, partial sums, uniform tail estimates, real seal, graph replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from*

$$D, S \longmapsto A \longmapsto B \longmapsto U \longmapsto G \longmapsto R$$

and then through the structural rows.

10. *P is the* Pkg *provenance over* $\mathsf{TakagiFunction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TakagiFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, A, B, U, R, G, H, C, P, N$. It has no coordinate for a global differentiability predicate, derivative values, host real equality, quotient equality of functions, countable choice, selected infinite series witnesses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7692.2 (Takagi function NameCert obligations)**.** *For every accepted* $\mathsf{TakagiFunction}^{\uparrow}$ *carrier $T = (D, S, A, B, U, R, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TakagiFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row,* $\mathsf{DyadicPartition}^{\uparrow}$ *subdivision row, stage-amplitude row, finite partial-sum ledger,* $\mathsf{UniformLimit}^{\uparrow}$ *tail row, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph replay row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real-facing read follows the displayed route*

$$D, S \longmapsto A \longmapsto B \longmapsto U \longmapsto G \longmapsto R$$

together with H, C, P, N.

Proof. Project the carrier of Definition 7692.1. Its displayed rows are exactly dyadic arithmetic, dyadic subdivision, stage amplitudes, finite partial sums, uniform tail estimates, the terminal real seal, graph replay, transport, replay, provenance, and local naming.

The stage route is finite. The subdivision row supplies the cells, the dyadic arithmetic row supplies the midpoint-distance and scale comparisons, and the stage-amplitude row reads the tent contribution only on those displayed cells.

The partial-sum and tail rows control the handoff to the real-name surface. The ledger B exposes the finite sum actually carried by the packet, while U exposes the dyadic bound for the omitted stage tail before any consumer reaches G or R.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same route. Since no carrier coordinate stores a derivative predicate, host real equality, quotient function equality, countable-choice schedule, selected infinite series witness, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 7692.3 (Takagi function dyadic uniform-limit boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{ContinuousMap}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{TakagiFunction}^{\uparrow}$ *packet must pass through the finite dyadic route*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, stage amplitude, partial sum, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$

The boundary exported by the packet is the displayed uniform-limit estimate for dyadic tent-map stages. It is not a proof of full nowhere-differentiability, host real equality, quotient equality of functions, countable choice, selected infinite series data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with any consumer of the graph row or terminal seal. Carrier admission forces the read to expose D and S, so the dyadic cells and arithmetic comparisons are visible before any stage value is inspected.

Next read A and B. These rows record only finitely many tent-map stage contributions and their displayed dyadic partial sum, so the consumer receives a finite polygonal approximation rather than an external infinite series object.

Finally read U, G, R. The uniform-limit row supplies the dyadic tail estimate that permits the graph replay and real-name seal to be consumed. Because the carrier has no coordinate for derivative data, quotient equality, choice, selected series witnesses, or ambient completeness, the exported boundary is exactly the dyadic uniform-limit route. □

Falsifiable prediction: An accepted $\mathsf{TakagiFunction}^{\uparrow}$ consumer is refuted if it reaches a $\mathsf{ContinuousMap}^{\uparrow}$ graph row or $\mathsf{Real}^{\uparrow}$ seal without first displaying the $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic row, $\mathsf{DyadicPartition}^{\uparrow}$ subdivision row, stage amplitudes, finite partial-sum ledger, $\mathsf{UniformLimit}^{\uparrow}$ tail estimate, transport, displayed replay, provenance, and local naming rows.

Independence witness: dyadicratcore, dyadic-partition, uniform-limit, real, continuousmap: no listed sibling supplies the complete Takagi-function packet, because only $\mathsf{TakagiFunction}^{\uparrow}$ binds dyadic arithmetic, dyadic subdivision, tent-map stage amplitudes, finite partial sums, uniform tail estimates, graph replay, Real seal, transport, replay, provenance, and local naming as one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{TakagiFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{TakagiFunction}^{\uparrow}$ packet is generated from DyadicRatCore arithmetic, dyadic subdivision rows, tent-map stage amplitudes, finite partial sums, a UniformLimit tail estimate, a ContinuousMap graph replay row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TakagiFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7692.1, Definition 7692.2, and Definition 7692.3 seed the finite dyadic tent-map route over DyadicRatCore, dyadic subdivision, partial sums, uniform tail estimates, ContinuousMap, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TakagiFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full differentiability theory, host real equality, quotient equality of functions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic arithmetic admission, dyadic subdivision exposure, stage-amplitude readback, finite partial-sum exactness, UniformLimit tail control, ContinuousMap graph replay, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TakagiFunctionUp.

Chapter 7693

A Concrete Naming Certificate for $\mathsf{KirchhoffMatrixTree}^{\uparrow}$

Following Kirchhoff's matrix-tree theorem, the $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ packet records the finite route that turns a graph incidence ledger into a Laplacian matrix, deletes one row and one column, reads the determinant of the minor, and exposes the resulting spanning-tree count as a displayed certificate. It sits over the graph carrier of chapter 118, the acyclic connected carrier of chapter 119, the spanning-tree selection surface of chapter 237, the finite matrix row of chapter 49, and the determinant row of chapter 84. The packet is a finite combinatorics bridge; it does not import spectral graph theory, quotient graph isomorphism, host linear algebra, an infinite graph limit, or a selected external enumeration of all spanning trees.

Definition 7693.1 (Kirchhoff matrix-tree carrier)**.** *A* $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, E, L, M, D, S, T, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *row naming the finite vertex and edge incidence data.*
2. *E is the edge-incidence ledger used to compute vertex degrees and adjacency counts from G.*
3. *L is the Laplacian ledger whose diagonal rows are degree counts and whose off-diagonal rows are the displayed negative adjacency counts.*
4. *M is the* $\mathsf{Mat}^{\uparrow}$ *row storing a principal Laplacian minor after one displayed vertex row and column have been removed.*
5. *D is the* $\mathsf{Determinant}^{\uparrow}$ *row reading* $\det(M)$ *by the finite determinant certificate.*
6. *S is the* $\mathsf{SpanningTree}^{\uparrow}$ *row naming the finite family of admitted spanning-tree witnesses.*
7. *T is the* $\mathsf{Tree}^{\uparrow}$ *route used to certify that each admitted spanning-tree witness is connected, acyclic, and vertex-spanning inside G.*
8. *H records componentwise* hsame *transport for graph, incidence, Laplacian, minor, determinant, spanning-tree, tree, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the graph incidence row through the Laplacian ledger, matrix minor, determinant row, spanning-tree row, and tree route.*

10. *P is the* Pkg *provenance over* $\mathsf{KirchhoffMatrixTree}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{KirchhoffMatrixTree}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host graph isomorphism, spectral eigenvalues, quotient linear algebra, an infinite graph limit, selected external tree enumeration, or ambient determinant algorithms outside the displayed $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{Determinant}^{\uparrow}$ *routes.*

Lean4 ✓ `BEDC.Derived.KirchhoffMatrixTreeUp.TasteGate.KirchhoffMatrixTreeTasteGate single carrier alignment`

Theorem 7693.2 (Kirchhoff matrix-tree NameCert obligations)**.** *For every accepted* $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ *carrier* $K = (G, E, L, M, D, S, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KirchhoffMatrixTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed graph row, incidence ledger, Laplacian ledger, principal minor matrix, determinant row, spanning-tree family, tree-validity route, componentwise transport, displayed replay, package provenance, and local naming. No obligation is discharged by host spectral theory, graph isomorphism, quotient linear algebra, selected external enumeration of spanning trees, infinite graph limits, or determinant computations outside the displayed finite rows.*

Proof. Project the carrier of Definition 7693.1. The graph row is G, the incidence data are E, the Laplacian row is L, the principal minor is M, the determinant certificate is D, and the spanning-tree surface is S together with the tree-validity route T.

The determinant side is finite. The row L is computed only from the displayed incidence ledger, the row M removes one displayed vertex row and column from L, and D reads the determinant of that minor through $\mathsf{Determinant}^{\uparrow}$.

The tree side is also finite. The row S exposes only admitted spanning-tree witnesses over G, and T checks connectedness, acyclicity, and vertex coverage through the $\mathsf{Tree}^{\uparrow}$ route.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host spectral theory, graph isomorphism, quotient linear algebra, selected external enumeration, infinite graph limits, or external determinant algorithms, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.KirchhoffMatrixTreeUp.KirchhoffMatrixTreeNameCertObligations`

Theorem 7693.3 (Kirchhoff matrix-tree Laplacian minor route)**.** *Every determinant-facing or spanning-tree-facing consumer of an accepted* $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Graph}^{\uparrow} \longmapsto E \longmapsto L \longmapsto \mathsf{Mat}^{\uparrow} \longmapsto \mathsf{Determinant}^{\uparrow}$$

and the parallel spanning-tree route

$$\mathsf{Graph}^{\uparrow} \longmapsto \mathsf{SpanningTree}^{\uparrow} \longmapsto \mathsf{Tree}^{\uparrow}.$$

The packet binds these two routes at the same graph row G, so the determinant read and the spanning-tree read cannot refer to different finite graphs or to an external enumeration.

Proof. Start with a consumer of the determinant side. Carrier admission forces the consumer to read G and the incidence ledger E before any Laplacian entry is available.

The minor is downstream of the same graph. The Laplacian ledger L is formed from E, the matrix row M is a principal minor of L, and the determinant row D reads only that displayed matrix.

Now read the spanning-tree side. The row S is indexed by the same graph row G, and T checks each admitted witness through the $\mathsf{Tree}^{\uparrow}$ route over that graph.

The transport and replay rows H and C preserve this shared source under classifier comparison. Thus a consumer cannot mix a determinant computed from one graph with spanning-tree witnesses from another graph, nor can it replace the displayed rows with a host enumeration. □

Falsifiable prediction: An accepted $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ consumer is refuted if it reaches a determinant row without first displaying the GraphUp incidence ledger, Laplacian ledger, principal MatrixUp minor, DeterminantUp row, SpanningTreeUp family, TreeUp validity route, transport, replay, provenance, and local naming rows.

Independence witness: graph, tree, spanningtree, matrix, determinant: no listed sibling supplies the full matrix-tree bridge, because only $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ binds one finite graph incidence ledger to both a Laplacian determinant minor and a spanning-tree family in the same NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KirchhoffMatrixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{KirchhoffMatrixTree}^{\uparrow}$ packet is generated from a finite GraphUp row, an incidence ledger, a Laplacian ledger, a principal MatrixUp minor, a DeterminantUp row, a SpanningTreeUp family, a TreeUp validity route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KirchhoffMatrixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7693.1, Definition 7693.2, and Definition 7693.3 seed the finite matrix-tree route over GraphUp, TreeUp, SpanningTreeUp, MatrixUp, DeterminantUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KirchhoffMatrixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, spectral graph theory, quotient graph isomorphism, host linear algebra, infinite graph limits, selected external tree enumeration, or determinant algorithms outside displayed finite rows.

Upgrade path: Obligation closure requires checked carrier admission, incidence exactness, Laplacian construction, principal minor routing, determinant readback, spanning-tree family validity, tree-route exactness, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.KirchhoffMatrixTreeUp.

Chapter 7694

A Concrete Naming Certificate for $\mathsf{PastingLemma}^{\uparrow}$

Following Kelley and Munkres on the topological pasting lemma, the $\mathsf{PastingLemma}^{\uparrow}$ packet records the finite BEDC route by which a map is certified from local continuous rows on a finite closed cover. It sits after the indexed-open carrier of chapter 88, the graph-and-modulus interface of chapter 130, and the generated-open discipline of chapter 2000. The packet names only the finite gluing certificate needed by continuous-map, sheaf, manifold, and bundle consumers; it is not host function extensionality, a selected representative theorem, a quotient-space construction, open-cover compactness, or ambient real completeness.

Definition 7694.1 (PastingLemma carrier)**.** *A* $\mathsf{PastingLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, T_X, T_Y, I, U, F, A, L, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source and target carrier histories, read only through their displayed classifiers.*
2. *T_X and T_Y are the* $\mathsf{Topology}^{\uparrow}$ *indexed-open rows on the source and target carriers.*
3. *I is a finite closed-cover index ledger for the source, with one displayed closed cover row for each $i \in I$ and a finite membership replay for every carried source observation.*
4. *U records the local domain row U_i for each index i, including its inclusion route into X and its induced* $\mathsf{Topology}^{\uparrow}$ *row from T_X.*
5. *F records a local* $\mathsf{ContinuousMap}^{\uparrow}$ *row $f_i : U_i \to Y$ for each $i \in I$, read through U_i, T_Y and the displayed continuity witness.*
6. *A records overlap-agreement rows: for every displayed pair $i, j \in I$, the two local maps agree on the carried overlap $U_i \cap U_j$ by componentwise observation transport.*
7. *L is the glued-map ledger whose source observation first chooses a cover row, then reads the corresponding local map, and whose well-definedness is certified only by A.*
8. *H records componentwise* hsame *transport for the source, target, topology rows, finite cover ledger, local domains, local maps, overlap rows, glued-map ledger, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a target-open request through the chosen local* $\mathsf{ContinuousMap}^{\uparrow}$ *row, the cover membership row, overlap agreement, and the glued-map ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{PastingLemma}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PastingLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T_X, T_Y, I, U, F, A, L, H, C, P, N$. The carrier has no coordinate for host function extensionality, selected representatives, quotient spaces, open-cover compactness, ambient real completeness, or hidden point choice.

Theorem 7694.2 (PastingLemma NameCert obligations)**.** *For every accepted* $\mathsf{PastingLemma}^{\uparrow}$ *carrier*

$$G = (X, Y, T_X, T_Y, I, U, F, A, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PastingLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the thirteen displayed rows; finite cover exposure through I; induced local-domain topology through U; local continuity through F; overlap agreement through A; glued-map well-definedness and continuity replay through L and C; componentwise transport through H; provenance through P; and local naming through N.*

Proof. Project the accepted packet. The only available rows are the two carriers, the two topology rows, the finite cover ledger, the local domain rows, the local continuous maps, the overlap-agreement ledger, the glued-map ledger, and the structural rows.

Carrier admission is therefore exhausted by those rows. The cover surface is read from I, and each local domain is read from U together with its induced topology from T_X.

Local continuity is read from F. Whenever two cover choices meet, the pairwise row in A is the only available witness that the two local reads produce the same target observation on the carried overlap.

The glued-map ledger L uses that agreement to remove dependence on the displayed cover choice, while C replays continuity by restricting a target-open request to each local map and then repacking the finite cover rows. Finally, H, P, N transport, source, and name the same finite route, so the listed rows exhaust the local certificate surface. □

Theorem 7694.3 (PastingLemma continuity gluing route)**.** *Every continuity read exported by an accepted* $\mathsf{PastingLemma}^{\uparrow}$ *carrier factors through*

$$I, U, F, A, L, H, C, P, N.$$

More explicitly, a target-open request is checked locally by the $\mathsf{ContinuousMap}^{\uparrow}$ *rows in F, transported across overlaps by A, assembled by the finite cover ledger I, and exported by the glued-map ledger L. No consumer obtains a continuity witness for the glued map except through those displayed rows.*

Proof. Start with a target-open request against the glued map. The only global map coordinate is L, so the request must enter the glued-map ledger.

The ledger first reads a finite cover row from I and the corresponding local domain row from U. On that local domain, continuity is exactly the $\mathsf{ContinuousMap}^{\uparrow}$ row recorded in F.

If two cover rows can account for the same source observation, the request is transported through the corresponding overlap row in A. This is the only row that compares local map outputs on overlaps.

The replay row C then repacks the local continuity reads into the global request, and H, P, N preserve the same route under transport, provenance, and naming. Since no other continuity coordinate is present, every exported gluing route factors through the displayed finite ledger. □

Theorem 7694.4 (PastingLemma non-escape boundary)**.** *An accepted* $\mathsf{PastingLemma}^{\uparrow}$ *packet exports no host function extensionality principle, selected representative of a local map, quotient-space construction, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem. Any downstream consumer of the glued continuity certificate receives only the finite rows*

$$X, Y, T_X, T_Y, I, U, F, A, L, H, C, P, N.$$

Proof. Inspect the carrier inventory of Definition 7694.1. The packet contains source and target carriers, topology rows, finite cover data, local domains, local continuous maps, overlap agreement, a glued-map ledger, and the structural transport, replay, provenance, and naming rows.

Host function extensionality is not among those coordinates. The only equality-like data are the displayed overlap-agreement rows in A, and they apply only on carried overlaps.

Selected representatives and quotient spaces are also absent. The glued-map ledger records a cover-mediated readback, not a quotient representative or a choice of canonical local index.

Open-cover compactness and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent for the same reason: the cover is finite by construction, and no completeness row appears in G. Therefore the exported certificate cannot escape the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{PastingLemma}^{\uparrow}$ consumer that obtains a global continuity read without first reading the finite cover ledger, a local $\mathsf{ContinuousMap}^{\uparrow}$ row, and the relevant overlap-agreement row refutes the carrier surface.

Independence witness: topology, continuousmap, product-topology: no listed sibling supplies the whole packet $G = (X, Y, T_X, T_Y, I, U, F, A, L, H, C, P, N)$, because Topology supplies indexed-open rows, ContinuousMap supplies local continuity rows, and ProductTopology supplies a separate binary generated-open route, while this chapter binds finite cover rows and overlap agreement into one pasting NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PastingLemma}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PastingLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7694.1, Definition 7694.2, Definition 7694.3, and Definition 7694.4 seed the finite pasting route from local continuous rows and overlap agreement to a global continuity certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PastingLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function extensionality, selected representatives, quotient spaces, open-cover compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a BEDC.Derived.PastingLemmaUp carrier with checked carrier admission, finite cover exposure, local-domain topology, local continuity, overlap agreement, glued-map well-definedness, continuity replay, transport, provenance, and local naming.

Constructive story: A $\mathsf{PastingLemma}^{\uparrow}$ packet is generated from source and target carrier histories, two $\mathsf{Topology}^{\uparrow}$ rows, a finite cover ledger, local domain rows, local $\mathsf{ContinuousMap}^{\uparrow}$ rows, pairwise overlap-agreement rows, a glued-map ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7695

A Concrete Naming Certificate for $\mathsf{CayleyTreePruferCode}^{\uparrow}$

Following Cayley and Prufer, the $\mathsf{CayleyTreePruferCode}^{\uparrow}$ packet records the finite correspondence between labelled trees on a displayed vertex set and length-adjusted code lists. It sits over the graph carrier of chapter 118, the tree carrier of chapter 119, the spanning-tree surface of chapter 237, and the unary/list discipline of chapter 28 and chapter 36. The packet names only finite vertex labels, finite tree-edge ledgers, leaf-removal steps, code-list rows, replay rows, and local naming; it is not the matrix-tree theorem, a host graph isomorphism quotient, an unordered ambient tree set, or a choice principle selecting representatives.

Definition 7695.1 (Cayley tree Prufer-code carrier)**.** *A* $\mathsf{CayleyTreePruferCode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (V, G, T, S, L, C, R, H, K, Q, N)$$

with the following displayed rows:

1. *V is the finite labelled-vertex row, read through* $\mathsf{Nat}^{\uparrow}$ *size data and* $\mathsf{List}^{\uparrow}$ *finite label enumeration.*
2. *G is the* $\mathsf{Graph}^{\uparrow}$ *edge-incidence row over those displayed labels.*
3. *T is the* $\mathsf{Tree}^{\uparrow}$ *acyclicity and connectedness row carried by the displayed graph data.*
4. *S is the* $\mathsf{SpanningTree}^{\uparrow}$ *witness tying the tree rows to the graph rows.*
5. *L is the finite leaf-removal ledger, listing at each step the least displayed leaf and its unique neighbour.*
6. *C is the Prufer code-list row, a* $\mathsf{List}^{\uparrow}$ *row whose entries are displayed labels and whose length is the unary predecessor predecessor of the vertex count when that count is at least two.*
7. *R is the reconstruction ledger, replaying the code list back to the same labelled tree rows.*
8. *H records componentwise* hsame *transport for vertex labels, graph edges, tree rows, spanning-tree rows, leaf-removal rows, code rows, reconstruction rows, replay, provenance, and local naming.*
9. *K records displayed* Cont *replay from tree-to-code removal and code-to-tree reconstruction.*
10. *Q is the* Pkg *provenance over* $\mathsf{CayleyTreePruferCode}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CayleyTreePruferCode}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the leaf-removal ledger and reconstruction ledger consume the same displayed vertex labels and tree rows. It has no coordinate for quotienting labelled trees by graph isomorphism, choosing representatives from an ambient tree set, invoking the matrix-tree determinant, or importing an infinite graph space.

Theorem 7695.2 (Cayley tree Prufer-code NameCert obligations)**.** *For every accepted* $\mathsf{CayleyTreePruferCode}^{\uparrow}$ *carrier*

$$P = (V, G, T, S, L, C, R, H, K, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CayleyTreePruferCode}^{\uparrow}}$ *surface consists exactly of labelled-vertex exposure, graph-incidence exposure, tree exposure, spanning-tree compatibility, finite leaf-removal accounting, Prufer code-list exposure, reconstruction replay, componentwise transport, displayed continuation replay, package provenance, and local naming through* N*.*

Proof. Project the carrier tuple in Definition 7695.1. The rows V, G, T, S expose the labelled graph and tree data, while L records the finite removal route that reads leaves and neighbours as displayed labels.

The code row C is a finite $\mathsf{List}^{\uparrow}$ row controlled by the same unary size data in V. The reconstruction row R consumes that list and the remaining displayed labels, so it cannot introduce a hidden tree or an unlisted vertex.

The structural rows H, K, Q, N transport, replay, source, and name the same packet. Since no excluded host coordinate occurs in the carrier, the local NameCert surface is exactly the finite Cayley–Prufer correspondence route. □

Theorem 7695.3 (Cayley tree Prufer-code replay)**.** *Every accepted* $\mathsf{CayleyTreePruferCode}^{\uparrow}$ *packet replays in both finite directions:*

$$V, G, T, S \longmapsto L, C, H, K, Q, N$$

for leaf-removal encoding, and

$$V, C \longmapsto R, G, T, S, H, K, Q, N$$

for reconstruction. The replay preserves the displayed labelled vertices, tree edges, and code entries, and it exports no matrix-tree determinant, graph-isomorphism quotient, ambient set of trees, or representative-selection principle.

Proof. For encoding, read the displayed tree rows G, T, S over the vertex row V. The leaf-removal ledger L chooses leaves only from the finite displayed label list and records the neighbour label placed into the code row C.

For reconstruction, start from V and C. The reconstruction ledger R attaches each code entry to the least displayed label not currently blocked by the remaining code list, then appends the final edge between the two remaining displayed labels.

Both directions use the same transport, replay, provenance, and local naming rows H, K, Q, N. Therefore the replay is a finite labelled-tree/code correspondence inside the listed BEDC rows, with no appeal to determinant enumeration, quotient tree equality, ambient tree sets, or choice. □

Falsifiable prediction: A $\mathsf{CayleyTreePruferCode}^{\uparrow}$ consumer is refuted if it reads a Prufer code entry or reconstructed edge that is not attached to the displayed vertex-label row, graph row, tree row, spanning-tree row, leaf-removal ledger, reconstruction ledger, transport, replay, provenance, and local naming rows.

Independence witness: graph, tree, spanningtree, nat, list: no listed sibling alone supplies the complete Cayley–Prufer packet, because only $\mathsf{CayleyTreePruferCode}^{\uparrow}$ binds labelled vertices, graph incidence, tree rows, spanning-tree compatibility, leaf-removal encoding, code-list reconstruction, transport, replay, provenance, and local naming as one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{CayleyTreePruferCode}^{\uparrow}}$.

Constructive story: A $\mathsf{CayleyTreePruferCode}^{\uparrow}$ packet is generated from finite labelled vertices, displayed graph incidence, displayed tree rows, spanning-tree compatibility, a finite leaf-removal ledger, a Prufer code-list row, a reconstruction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CayleyTreePruferCode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7695.1, Definition 7695.2, and Definition 7695.3 seed the finite labelled-tree and Prufer-code correspondence over Graph, Tree, SpanningTree, Nat, List, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CayleyTreePruferCode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the matrix-tree theorem, quotient graph isomorphism, an ambient set of all trees, representative choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, leaf-removal exactness, code-list length accounting, reconstruction replay, labelled-edge preservation, transport, replay, provenance, and local naming for BEDC.Derived.CayleyTreePruferCode

Chapter 7696

A Concrete Naming Certificate for LadyzhenskayaInequality$^{\uparrow}$

Following Ladyzhenskaya and the standard Sobolev-energy route for two-dimensional fluid estimates, the LadyzhenskayaInequality$^{\uparrow}$ packet records the finite BEDC surface by which a Sobolev row, an L^2 mass row, a weak-gradient energy row, and a terminal real bound are displayed before a PDE regularity consumer reads the mixed-norm estimate. It sits beside the Sobolev source of chapter 440, the Poincare control route of chapter 2329, the finite interpolation window of chapter 6814, the iteration surfaces of chapter 6969 and chapter 7036, and the scalar seal of chapter 38. The packet names only displayed finite rows; it is not a host compact embedding, selected weak solution, quotient function-space equality, or ambient PDE regularity theorem.

Definition 7696.1 (LadyzhenskayaInequality Sobolev-energy carrier)**.** *A* LadyzhenskayaInequality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, M, G, P_2, I, B, H, C, K, N)$$

with the following displayed rows:

1. *D is the displayed bounded two-dimensional domain or finite coordinate-window row on which all reads are scoped.*
2. *S is the consumed* Sobolev$^{\uparrow}$ *row exposing the function carrier and weak-gradient data.*
3. *M is the L^2 mass row for the displayed function.*
4. *G is the weak-gradient L^2 energy row read through S.*
5. *P_2 is the two-dimensional Sobolev-energy profile row that records the exponent pattern $2, 4, 2$ and the finite interpolation window.*
6. *I is the* GagliardoNirenbergFiniteWindow$^{\uparrow}$ *or Sobolev-interpolation handoff row used to combine M and G.*
7. *B is the terminal* Real$^{\uparrow}$ *bound row for the Ladyzhenskaya estimate over the displayed domain window.*
8. *H records componentwise* hsame *transport for the domain, Sobolev, mass, energy, profile, interpolation, bound, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a mixed-norm request through M, G, P_2, I, and B.*

10. *K is the* Pkg *provenance over* LadyzhenskayaInequality$^{\uparrow}$, Sobolev$^{\uparrow}$, PoincareInequality$^{\uparrow}$, GagliardoNirenbergFiniteW MoserIteration$^{\uparrow}$, DeGiorgiIteration$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LadyzhenskayaInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, M, G, P_2, I, B, H, C, K, N$. It has no coordinate for a host Sobolev space quotient, compactness theorem, selected PDE solution, or ambient regularity principle.

Theorem 7696.2 (LadyzhenskayaInequality NameCert obligations)**.** *For every accepted* LadyzhenskayaInequality$^{\uparrow}$ *carrier*

$$L = (D, S, M, G, P_2, I, B, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{LadyzhenskayaInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; bounded domain-window exposure through D; Sobolev and weak-gradient exposure through S and G; mass exposure through M; two-dimensional exponent-profile exposure through P_2; finite interpolation handoff through I; terminal real-bound exposure through B; componentwise transport through H; mixed-norm replay through C; provenance through K; and local naming through N.*

Proof. Project the accepted packet. The only analytic rows are the domain window, Sobolev row, mass row, gradient-energy row, exponent profile, interpolation handoff, and terminal real bound.

Carrier admission is therefore exhausted by those rows. The Sobolev row supplies the function and weak-gradient carrier, while M and G supply the two energy quantities read by the estimate.

The two-dimensional profile P_2 fixes the exponent pattern before interpolation is invoked. The handoff I combines the mass and gradient rows inside the displayed window, and B records the resulting real bound.

The structural rows add no hidden analytic content. They transport, replay, source, and name the same finite route, so the listed clauses exhaust the local certificate surface. □

Theorem 7696.3 (LadyzhenskayaInequality Sobolev-energy route)**.** *Every mixed-norm estimate exported by an accepted* LadyzhenskayaInequality$^{\uparrow}$ *packet factors through the finite route*

$$D, S, M, G, P_2, I, B, H, C, K, N.$$

The consumer must read the displayed Sobolev row, the L^2 mass row, the weak-gradient energy row, the two-dimensional exponent profile, and the finite interpolation handoff before it can receive the terminal Real$^{\uparrow}$ *bound.*

Proof. Begin with a mixed-norm request. The only route from that request to a real bound is the replay row C, so the request must pass through the rows named by the carrier.

The Sobolev row S exposes the function and weak-gradient surface. The mass row M and energy row G are read through that same displayed carrier, and the profile row P_2 fixes the two-dimensional exponent pattern before any bound is formed.

The interpolation handoff I consumes exactly those finite rows. The bound row B is then the terminal Real$^{\uparrow}$ seal for the estimate, with H, K, N preserving transport, provenance, and local naming. Since no compactness, selected solution, or quotient coordinate is present, every exported estimate remains inside the Sobolev-energy route. □

Theorem 7696.4 (LadyzhenskayaInequality non-escape boundary)**.** *An accepted* LadyzhenskayaInequality$^{\uparrow}$ *packet exports no ambient function-space compactness, no host PDE regularity theorem, no selected weak solution, no quotient equality of Sobolev representatives, and no unscoped domain extension. Any downstream* MoserIteration$^{\uparrow}$, DeGiorgiIteration$^{\uparrow}$, *or* HarnackInequality$^{\uparrow}$ *consumer receives only the displayed finite Sobolev-energy rows of Definition 7696.1.*

Proof. Inspect the carrier inventory. It contains a domain window, Sobolev row, mass row, gradient-energy row, exponent profile, interpolation handoff, real bound, and structural rows.

Compactness and quotient equality require additional coordinates absent from the carrier. A selected weak solution or host PDE regularity theorem would likewise need solution and equation rows not displayed by $\mathsf{LadyzhenskayaInequality}^{\uparrow}$.

The iteration consumers can therefore read only the finite energy estimate and its transport, replay, provenance, and naming rows. The packet supplies a Sobolev-energy estimate surface, not an ambient analytic theorem. □

Falsifiable prediction: An accepted $\mathsf{LadyzhenskayaInequality}^{\uparrow}$ consumer that obtains a mixed L^4-type estimate without displaying the Sobolev row, L^2 mass row, weak-gradient energy row, two-dimensional exponent profile, interpolation handoff, and terminal $\mathsf{Real}^{\uparrow}$ bound refutes the carrier surface.

Independence witness: sobolev, poincare-inequality, gagliardo-nirenberg-finite-window, moser-iteration, de-giorgi-iteration: no listed sibling supplies the whole packet $L = (D, S, M, G, P_2, I, B, H, C, K, N)$, because Sobolev supplies weak-gradient data, Poincare supplies a separate norm-control route, GagliardoNirenbergFiniteWindow supplies a broader interpolation window, and the Moser and De Giorgi packets consume iteration budgets while this chapter binds the two-dimensional mass-gradient Ladyzhenskaya estimate as one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LadyzhenskayaInequality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LadyzhenskayaInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7696.1, Definition 7696.2, Definition 7696.3, and Definition 7696.4 seed the finite Sobolev-energy estimate route over mass, weak-gradient energy, interpolation, real-bound, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LadyzhenskayaInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient compactness, host PDE regularity, selected weak solutions, quotient Sobolev equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a BEDC.Derived.LadyzhenskayaInequalityUp carrier with checked carrier admission, Sobolev-energy exposure, interpolation handoff, real-bound replay, transport, provenance, and local naming.

Constructive story: A $\mathsf{LadyzhenskayaInequality}^{\uparrow}$ packet is generated from a bounded domain window, Sobolev row, mass row, weak-gradient energy row, two-dimensional exponent profile, interpolation handoff, Real bound, componentwise transport, displayed replay, provenance, and local NameCert row.

Chapter 7697

A Concrete Naming Certificate for $\mathsf{SchurFunction}^{\uparrow}$

Following Macdonald on symmetric functions and Fulton–Harris on representation theory, the $\mathsf{SchurFunction}^{\uparrow}$ packet records the finite tableau-expansion interface for Schur functions as BEDC NameCert data. It sits after the Young-tableau carrier of chapter 7178, the partition-shape surface of chapter 201, the polynomial carrier of chapter 50, the ring row of chapter 43, and the representation-ring consumer of chapter 246. The packet names finite shape, tableau, monomial, coefficient, algebra, transport, replay, provenance, and local certificate rows; it does not import a host symmetric-function algebra, quotient polynomial equality, an infinite tableau enumeration, a selected basis theorem, or representation-theoretic semisimplicity.

Definition 7697.1 (Schur function carrier)**.** *A* $\mathsf{SchurFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\lambda, T, M, K, A, R, H, C, P, N)$$

with the following displayed rows:

1. λ *is the* $\mathsf{Partition}^{\uparrow}$ *row naming the finite Young shape.*
2. T *is the* $\mathsf{YoungTableau}^{\uparrow}$ *tableau ledger over* λ*, including the displayed row and column constraints used by semistandard reads.*
3. M *is the finite monomial ledger. It records the content vector read from each admitted tableau before any polynomial coefficient is exposed.*
4. K *is the coefficient ledger, storing the finite multiplicity count attached to each displayed content vector.*
5. A *is the* $\mathsf{Poly}^{\uparrow}$ *row that packages the finite monomial-coefficient data as a polynomial expression.*
6. R *is the* $\mathsf{Ring}^{\uparrow}$ *row through which addition and multiplication of the displayed monomial terms are replayed.*
7. H *records componentwise* hsame *transport for partition, tableau, monomial, coefficient, polynomial, ring, representation-ring, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from the partition shape through tableau rows, monomial content, coefficient rows, polynomial assembly, and ring operations.*
9. P *is the* Pkg *provenance over* $\mathsf{SchurFunction}^{\uparrow}$, $\mathsf{YoungTableau}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SchurFunction}^\uparrow}$ *row.*
The classifier compares accepted carriers only through these rows. It has no coordinate for a host symmetric-function algebra, quotient equality of polynomials, infinite tableau enumeration, selected basis theorem, Littlewood–Richardson multiplication rule, or semisimplicity theorem.

Theorem 7697.2 (Schur function NameCert obligations)**.** *For every accepted* $\mathsf{SchurFunction}^\uparrow$ *carrier* $S = (\lambda, T, M, K, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SchurFunction}^\uparrow}$ *surface consists exactly of carrier admission for the displayed partition shape, Young-tableau ledger, monomial-content ledger, coefficient ledger, polynomial assembly row, ring-operation row, componentwise transport, displayed replay, package provenance, and local naming. Every representation-ring-facing read must consume the same rows before treating the carrier as a Schur-function row.*

Proof. Project the carrier of Definition 7697.1. The partition shape, tableau ledger, monomial ledger, coefficient ledger, polynomial row, and ring row are the only mathematical rows available to the local certificate.

The tableau expansion is finite. A read starts with the shape row λ, opens the tableau ledger T, extracts the content vector through M, records its displayed multiplicity in K, and only then assembles the polynomial row A through the ring row R.

The structural rows H, C, P, N transport, replay, source, and name that same route for downstream consumers. Since no carrier coordinate names a host symmetric-function algebra, quotient polynomial equality, infinite tableau enumeration, selected basis theorem, Littlewood–Richardson multiplication rule, or semisimplicity theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 7697.3 (Schur function tableau expansion handoff)**.** *Every* $\mathsf{RepresentationRing}^\uparrow$ *or character-theory consumer of an accepted* $\mathsf{SchurFunction}^\uparrow$ *carrier must read the finite tableau-expansion route*

$$\mathsf{Partition}^\uparrow \longmapsto \mathsf{YoungTableau}^\uparrow \longmapsto M, K \longmapsto \mathsf{Poly}^\uparrow \longmapsto \mathsf{Ring}^\uparrow \longmapsto \mathsf{RepresentationRing}^\uparrow.$$

Thus the Schur-function row handed to representation-ring consumers is a displayed finite polynomial route over partition and tableau data, not a host symmetric-function basis theorem or an unrecorded representation-theoretic decomposition.

Proof. Begin with a representation-ring-facing consumer. Carrier admission forces the consumer to read the partition shape and the Young-tableau ledger before any monomial content is available.

The expansion rows are then fixed. The monomial ledger M reads content from displayed tableaux, the coefficient ledger K records finite multiplicities, and the polynomial row A packages those cells through the ring row R.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. The $\mathsf{RepresentationRing}^\uparrow$ consumer therefore receives a finite Schur-function expansion over the displayed sibling rows, with no access to a host symmetric-function algebra, quotient polynomial equality, infinite tableau enumeration, selected basis theorem, Littlewood–Richardson multiplication rule, or semisimplicity theorem. □

Falsifiable prediction: An accepted $\mathsf{SchurFunction}^\uparrow$ consumer is refuted if it reaches a RepresentationRingUp or character-theory row without first displaying the PartitionUp shape, YoungTableauUp tableau ledger, finite monomial-content ledger, coefficient ledger, PolynomialUp assembly row, RingUp replay row, transport, replay, provenance, and local naming rows.

Independence witness: youngtableau, partition, polynomial, ring, representationring: no listed sibling supplies the full Schur-function tableau-expansion carrier, because only $\mathsf{SchurFunction}^{\uparrow}$ binds a partition shape, tableau ledger, monomial content, coefficient multiplicities, polynomial assembly, ring replay, and representation-ring handoff in one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchurFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{SchurFunction}^{\uparrow}$ packet is generated from a PartitionUp shape row, a YoungTableauUp tableau ledger, finite monomial-content rows, coefficient rows, a PolynomialUp assembly row, a RingUp replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurFunction}^{\uparrow}}$, seed)

Scope closed: Definition 7697.1, Definition 7697.2, and Definition 7697.3 seed the finite tableau-expansion route over YoungTableauUp, PartitionUp, PolynomialUp, RingUp, RepresentationRingUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host symmetric-function algebra, quotient polynomial equality, infinite tableau enumeration, selected basis theorems, Littlewood–Richardson multiplication, or representation-theoretic semisimplicity.

Upgrade path: Obligation closure requires checked carrier admission, tableau-ledger exactness, monomial-content extraction, coefficient exactness, polynomial assembly, ring replay, representation-ring handoff, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.SchurFunctionUp.

Chapter 7698

A Concrete Naming Certificate for HardyLittlewoodMaximalFinite$^{\uparrow}$

Following Hardy–Littlewood and Bishop–Bridges finite covering practice, the HardyLittlewoodMaximalFinite$^{\uparrow}$ packet records the weak-window maximal-function route in which every local average, threshold comparison, and covering handoff is displayed by finite data. It is a finite-window sibling of chapter 3581, with covering reads through chapter 7683, finite-net reads through chapter 1767, dyadic threshold reads through chapter 291, and terminal numerical sealing through chapter 38. The packet refuses sigma-measure, host measurable functions, choice-selected level sets, quotient level sets, ambient integration completeness, and any selected supremum outside the finite window ledger.

Definition 7698.1 (Hardy-Littlewood finite weak-window carrier)**.** *A* HardyLittlewoodMaximalFinite$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, I, B, D, A, T, V, F, G, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite metric-source row. It records the inspected points, displayed distance reads, and the finite ambient window in which all later probes live.*
2. *I is the finite interval-window row. It stores bounded interval names, endpoint comparisons, and the membership probes that allow an interval read to be replayed without a host measurable set.*
3. *B is the finite ball-window row. Each ball record stores a center, a positive radius witness, the inspected membership probes, and the inclusion comparisons with the interval windows in I.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *threshold row. It stores the dyadic level λ, dyadic tolerance budgets, and the rational comparison witnesses used by the weak-level ledger.*
5. *A is the local-average row. For every displayed interval or ball window it records the finite summation route, denominator witness, and comparison against the dyadic threshold from D.*
6. *T is the weak-level trace row. It records exactly which displayed windows have local average above the dyadic threshold and keeps the witness tied to A and D.*
7. *V is the* VitaliCoveringLemma$^{\uparrow}$ *finite cover row. It records the selected disjoint finite subfamily, the finite overlap budget, and the cover cells used to dominate the weak-level trace.*

8. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *finite-net row. It records the tolerance-visible net cells that make the cover route compact and finite before any weak-level output is exported.*
9. *G is the weak-window ledger. It stores the finite list of covered threshold windows, their dyadic weight comparisons, and the resulting weak-type bound as a displayed ledger row rather than a measure-theoretic inequality.*
10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the real-name windows reached from A, D, G.*
11. *H records componentwise* hsame *transport for source, interval, ball, threshold, average, weak trace, cover, finite-net, ledger, seal, replay, provenance, and local name rows.*
12. *C records displayed* Cont *replay of the route*

$$\mathsf{HardyLittlewoodMaximal}^{\uparrow},\ \mathsf{VitaliCoveringLemma}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist} \longmapsto \mathsf{HardyLit}$$

13. *P is the* Pkg *provenance over* $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$, $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$, $\mathsf{VitaliCoveringLemma}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these fourteen rows. Its classifier compares accepted packets only through $X, I, B, D, A, T, V, F, G, E, H, C, P, N$ and has no coordinate for sigma-measure, host measurable functions, choice, quotient level sets, ambient integration completeness, a selected supremum, or an infinite covering family.

Theorem 7698.2 (Hardy-Littlewood finite weak-window NameCert obligations)**.** *For an accepted* $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$ *carrier*

$$W = (X, I, B, D, A, T, V, F, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite metric source, finite interval windows, finite ball windows, dyadic thresholds, local-average rows, weak-level trace, Vitali finite-cover row, finite-net row, weak-window ledger, real seal, componentwise transport, displayed replay, package provenance, and local naming. Every weak-level read must inspect D, A, T, V, F, G in that order, and no obligation is discharged by sigma-measure, host measurable functions, choice, quotient level sets, ambient integration completeness, selected suprema, or infinite covering families.*

Proof. Project the carrier of Definition 7698.1. The projection exposes exactly the source row X, interval row I, ball row B, threshold row D, average row A, weak trace T, cover row V, finite-net row F, ledger row G, seal row E, and structural rows H, C, P, N.

The weak-level route is ordered by the rows themselves. A threshold comparison must first read the dyadic data in D and the matching local-average record in A; only then can the trace row T mark a displayed window as above the threshold.

The cover step is also finite. The Vitali row V supplies a selected disjoint finite subfamily, the finite-net row F supplies the tolerance-visible compact handoff, and the weak-window ledger G records the finite bound exported to consumers.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier contains no row for sigma-measure, host measurable functions, choice, quotient level sets, ambient integration completeness, selected suprema, or infinite covering families, the listed rows exhaust the local NameCert surface. □

Theorem 7698.3 (Hardy-Littlewood finite weak-window bound)**.** *Every accepted* $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$ *packet exports a weak-window ledger only along the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ A,\ T,\ \mathsf{VitaliCoveringLemma}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ G,\ \mathsf{Real}^{\uparrow}.$$

Thus a consumer receives a finite dyadic threshold ledger for displayed interval and ball windows, not a sigma-measure statement, a host measurable-function theorem, a choice-selected level set, a quotient level-set object, or an ambient integration-completeness assertion.

Proof. Begin with a consumer request for a weak-level read. By Definition 7698.2, the request must inspect the dyadic threshold row D, the local-average row A, and the weak trace T before any cover ledger can be read.

The cover route then passes through V and F. The Vitali covering row supplies only a disjoint finite subfamily and finite overlap budget, while the finite-net row supplies only tolerance-visible compact cells.

The output row G records the weak-window ledger over those displayed cells, and E seals the numerical read as a $\mathsf{Real}^{\uparrow}$ window. Componentwise transport, displayed replay, provenance, and local naming preserve that route without adding measure, choice, quotient, integration-completeness, selected-supremum, or infinite-family data. □

Falsifiable prediction: Every accepted $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$ read displays one finite metric-source row, one finite interval-window row, one finite ball-window row, one DyadicRatCoreUp threshold row, one local-average row, one weak-level trace, one VitaliCoveringLemmaUp finite-cover row, one FiniteEpsilonNetUp finite-net row, one weak-window ledger, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: hardy-littlewood-maximal, vitali-covering-lemma, finite-epsilon-net, dyadicratcore, real: $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite interval and ball windows, dyadic thresholds, local averages, weak-level traces, finite cover rows, finite-net handoffs, weak-window ledger rows, real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}$ packet is generated from finite metric-source data, interval windows, ball windows, dyadic thresholds, local-average rows, weak-level traces, finite Vitali-cover rows, finite-net rows, a weak-window ledger, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7698.1, Definition 7698.2, and Definition 7698.3 seed the finite weak-window maximal-function carrier and obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximalFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-measure, host measurable functions, choice-selected level sets, quotient level sets, ambient integration completeness, selected suprema, infinite covering families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-threshold exactness, local-average replay, weak-trace exactness, finite-cover discipline, finite-net handoff, ledger non-escape, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HardyLittlewoodMaximalFiniteUp.

Chapter 7699

A Concrete Naming Certificate for $\mathsf{FarEndBoundaryCospan}^{\uparrow}$

The $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ packet realizes the boundary-cospan branch of the apophatic far-end diagrammatics route in chapter 9403 as finite NameCert data. It sits beside the far-end diagram packet of chapter 4163, the far-end diagrammatics packet of chapter 4165, and the boundary-style transport rows used by adjacent hyperbolic and observer chapters. The packet names a finite cospan of left boundary, right boundary, apex far-end name, leg provenance, and compatibility rows; it is not a global universe, not an infinite object, not an observer, and not a selected terminal object in a host category.

Definition 7699.1 (FarEndBoundaryCospan carrier)**.** *A* $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, R, A, \lambda, \rho, B_L, B_R, J, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right boundary-name rows read from admissible far-end diagram data.*
2. *A is the apex far-end name row common to the two boundary legs.*
3. *λ and ρ are the left and right leg rows from L and R into A.*
4. *B_L and B_R are the boundary-provenance rows recording which observation-fiber or socket route produced the two boundary names.*
5. *J is the cospan-compatibility row asserting that both legs name the same apophatic far end only through the displayed apex row.*
6. *H records componentwise* hsame *transport for the boundary rows, apex row, leg rows, provenance rows, compatibility row, replay, package, and local naming.*
7. *C records displayed* Cont *replay from a boundary comparison request through $B_L, B_R, \lambda, \rho, J$ and the apex row A.*
8. *P is the* Pkg *provenance over* $\mathsf{FarEndBoundaryCospan}^{\uparrow}$*,* $\mathsf{FarEndDiagram}^{\uparrow}$*,* $\mathsf{FarEndDiagrammatics}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FarEndBoundaryCospan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, A, \lambda, \rho, B_L, B_R, J, H, C, P, N$. It has no coordinate for an ambient terminal object, global category of observers, selected boundary quotient, or infinite far-end completion.

Theorem 7699.2 (FarEndBoundaryCospan NameCert obligations)**.** *For every accepted* $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ *carrier*

$$F = (L, R, A, \lambda, \rho, B_L, B_R, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FarEndBoundaryCospan}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; left and right boundary exposure through* L, R*; apex far-end exposure through* A*; leg exposure through* λ, ρ*; provenance exposure through* B_L, B_R*; cospan compatibility through* J*; componentwise transport through* H*; boundary-comparison replay through* C*; package provenance through* P*; and local naming through* N*.*

Proof. Project the accepted packet. The only boundary data are the left row, right row, apex row, two legs, two provenance rows, and compatibility row.

Carrier admission is therefore exhausted by those rows. The two legs cannot be compared without the apex A, and the two boundary names cannot be sourced without the provenance rows B_L and B_R.

The compatibility row J records the cospan condition inside the displayed packet. The structural rows transport, replay, source, and name the same finite route, so no ambient terminal object or observer category is added by the certificate. □

Theorem 7699.3 (FarEndBoundaryCospan diagrammatic route)**.** *Every boundary comparison exported by an accepted* $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ *packet factors through the finite route*

$$L, R, A, \lambda, \rho, B_L, B_R, J, H, C, P, N.$$

The consumer must read the two boundary names, their provenance rows, the two legs, and the apex far-end name before the compatibility row can be consumed.

Proof. Start with a boundary comparison request. The only replay row is C, so the request must follow the displayed cospan route.

The boundary rows L and R are read together with their provenance rows B_L and B_R. The leg rows λ and ρ then map those boundary names into the apex row A.

Only after both legs have reached A can the compatibility row J be read. Transport and provenance remain componentwise through H and P, and the local name N records the same finite cospan. Thus the exported comparison is diagrammatic finite data, not a host categorical terminality claim. □

Theorem 7699.4 (FarEndBoundaryCospan non-escape boundary)**.** *An accepted* $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ *packet exports no global universe, no infinite far-end object, no observer identity, no selected terminal object, and no quotient of boundary names. It exports only the displayed boundary rows, apex row, leg rows, provenance rows, compatibility row, transport, replay, package provenance, and local naming rows of Definition 7699.1.*

Proof. Inspect the carrier inventory. Every row is either a boundary name, an apex far-end name, a leg, a provenance row, a compatibility row, or structural support.

A global universe or infinite far-end object would require a completion or totality coordinate absent from the carrier. An observer identity or terminal-object claim would require host category data absent from the packet.

The only equality-like information is the displayed cospan compatibility through J, and it is scoped by the apex row A. Therefore no consumer can escape from the finite cospan surface to an ambient quotient or terminality principle. □

Falsifiable prediction: An accepted $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ consumer that compares two far-end boundary names without displaying both boundary provenance rows, both cospan legs, the apex far-end name, and the compatibility row refutes the carrier surface.

Independence witness: far-end-diagram, far-end-diagrammatics, hyperbolic-boundary-transport, poincare-disk-boundary-transport, vision-coverage-index: no listed sibling supplies the whole packet $F = (L, R, A, \lambda, \rho, B_L, B_R, J, H, C, P, N)$, because far-end diagram rows name socket and fiber data, far-end diagrammatics names observation stratification, hyperbolic and Poincare boundary transports name geometric boundary movement, and the vision coverage index names marker coverage, while this chapter binds a two-leg boundary cospan with apex compatibility as one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FarEndBoundaryCospan}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FarEndBoundaryCospan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7699.1, Definition 7699.2, Definition 7699.3, and Definition 7699.4 seed the finite boundary-cospan route over two boundary names, an apex far-end name, two legs, provenance, compatibility, transport, replay, package, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FarEndBoundaryCospan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global universe, infinite far-end object, observer identity, selected terminal object, boundary quotient, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires BEDC.Derived.FarEndBoundaryCospanUp carrier admission, boundary provenance, cospan-leg compatibility, diagrammatic replay, and future TasteGate Nontrivial and FieldFaithful instances.

Constructive story: A $\mathsf{FarEndBoundaryCospan}^{\uparrow}$ packet is generated from left and right boundary-name rows, an apex far-end name, two leg rows, boundary-provenance rows, a compatibility row, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 7700

A Concrete Naming Certificate for $\mathsf{HahnSeries}^{\uparrow}$

Following Hahn and Kaplansky, the $\mathsf{HahnSeries}^{\uparrow}$ packet records a Hahn-series name as finite BEDC rows: coefficient-field data, an ordered exponent carrier, an inspected support ledger, and a valuation readback. It sits after the field certificate of chapter 45, the total-order route of chapter 54, the polynomial carrier of chapter 50, the coefficient-only formal-power-series surface of chapter 51, and the certificate transport discipline of chapter 481. The packet names the displayed non-Archimedean carrier surface; it does not import a host ordered group, a quotient field of fractions, a selected infinite summation, or an ambient completeness theorem.

Definition 7700.1 (Hahn-series carrier)**.** *A* $\mathsf{HahnSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (F, G, S, W, V, R, T, C, P, N)$$

with the following rows:

1. *F is the coefficient-field row, read through a displayed* $\mathsf{Field}^{\uparrow}$ *certificate and its scalar classifier.*
2. *G is the ordered exponent row, read through a displayed* $\mathsf{TotalOrder}^{\uparrow}$ *comparison surface, with the additive-exponent operations kept as named BEDC data rather than as a host ordered group.*
3. *S is the support row, recording the finite inspected exponent cells and the coefficient cell attached to each inspected exponent.*
4. *W is the well-ordered-support ledger, recording that every finite downward query exposed by the carrier has a first occupied exponent among the inspected cells or is explicitly empty.*
5. *V is the valuation row, recording the least occupied exponent returned by W when the support is nonempty and the formal infinity tag when the support is empty.*
6. *R is the coefficient readback row, recording the scalar returned at an inspected exponent and the displayed zero scalar returned outside the inspected support.*
7. *T records componentwise* hsame *transport for the field row, ordered exponent row, support ledger, valuation row, and coefficient readback.*
8. *C records displayed* Cont *replay from series admission through support inspection, valuation readback, coefficient readback, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HahnSeries}^{\uparrow}$*,* $\mathsf{Field}^{\uparrow}$*,* $\mathsf{TotalOrder}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{FPS}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}$ *row.*
The carrier contains no coordinate for an uninspected nonzero coefficient, a hidden support selector, a valuation outside the displayed ordered exponent row, or a coefficient comparison outside the scalar classifier of F.

Theorem 7700.2 (Hahn-series NameCert obligations)**.** *For the carrier of Definition 7700.1, the local* $\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for $H = (F, G, S, W, V, R, T, C, P, N)$, coefficient-field readback through F, ordered-exponent readback through G, inspected-support readback through S, well-ordered-support ledger exposure through W, valuation readback through V, coefficient readback through R, componentwise* hsame *transport through T, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the displayed packet $H = (F, G, S, W, V, R, T, C, P, N)$. The first six rows expose the algebraic and order-facing data: a coefficient field, an ordered exponent surface, inspected support cells, the well-ordered-support ledger, the valuation endpoint, and coefficient readback.

The structural rows are then forced. Transport T preserves the rows componentwise under hsame, replay C orders the consumer read from admission to local naming, provenance P records the sibling certificates, and N supplies the local name. There is no additional row from which a consumer could read hidden support or an external summation operation, so the listed groups exhaust the local NameCert surface. □

Theorem 7700.3 (Hahn-series support route)**.** *Every permitted consumer read of an accepted* $\mathsf{HahnSeries}^{\uparrow}$ *carrier reaches a coefficient cell only through the ordered-exponent row G, inspected-support row S, well-ordered-support ledger W, and coefficient readback row R. In particular, support membership is a displayed BEDC ledger route, not a host subset of an ambient ordered group.*

Proof. Begin with an accepted carrier $H = (F, G, S, W, V, R, T, C, P, N)$. By Definition 7700.1, all exponent comparisons are read through G, and all occupied cells available to the consumer are listed in S.

The ledger W controls support exposure. A finite downward query either returns the first occupied exponent among the displayed cells or records the query as empty, so the consumer cannot introduce an uninspected exponent while preserving the packet.

Finally read R through the replay row C. Coefficients are returned only from the inspected cells or as the displayed zero scalar outside the inspected support. Thus any support-dependent read factors through G, S, W, R and cannot use an external subset predicate. □

Theorem 7700.4 (Hahn-series valuation nonescape)**.** *For every accepted* $\mathsf{HahnSeries}^{\uparrow}$ *carrier $H = (F, G, S, W, V, R, T, C, P, N)$, the valuation endpoint carried by V is either the least occupied exponent certified by the well-ordered-support ledger W or the formal infinity tag for empty support. No consumer read may replace V by a smaller exponent, an uninspected support element, or a host valuation map while preserving the local* $\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}$ *row.*

Proof. Inspect the two branches recorded by W. In the nonempty branch, W supplies the first occupied exponent among the inspected cells and V records exactly that endpoint. In the empty branch, W exposes the absence of occupied inspected cells and V records the formal infinity tag.

Transport and replay do not add new valuation data. The row T preserves the displayed field, order, support, ledger, valuation, and coefficient rows under hsame, while C replays the valuation read only after support inspection. Replacing V by a smaller exponent or by a host valuation map would require a row absent from H, so the local name no longer certifies the read. □

Falsifiable prediction: An accepted $\mathsf{HahnSeries}^{\uparrow}$ consumer must expose $F, G, S, W, V, R, T, C, P, N$ before using any coefficient readback or valuation endpoint.

Independence witness: field, totalorder, polynomial, formalpowerseries, namecertmorphism: none is bijective to $\mathsf{HahnSeries}^{\uparrow}$ because the field certificate lacks ordered support, the total-order certificate lacks coefficient rows, the polynomial carrier has finite ordinary exponent support, the formal-power-series carrier lacks well-ordered exponent support and valuation readback, and the certificate-morphism route transports certificates without supplying a Hahn-series carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}$.

Constructive story: The Hahn-series packet is generated from finite BEDC rows naming a coefficient field, an ordered exponent surface, inspected support cells, a well-ordered-support ledger, a valuation row, coefficient readback, transport, replay, provenance, and local naming. The construction records the classical Hahn-series carrier as a seed NameCert surface whose support and valuation reads are ledgered before any non-Archimedean consumer may use them.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope is exactly the carrier of Definition 7700.1, the obligation surface of Definition 7700.2, the support route of Definition 7700.3, and the valuation nonescape boundary of Definition 7700.4.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HahnSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim field operations on all Hahn series, convergence, spherical completeness, algebraic closure, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier rows for coefficient readback, well-ordered support, valuation readback, and componentwise transport.

Chapter 7701

A Concrete Naming Certificate for $\mathsf{DyadicModulusRegularity}^{\uparrow}$

Following Bishop and Bridges on constructive real moduli, the $\mathsf{DyadicModulusRegularity}^{\uparrow}$ packet records a finite dyadic-modulus regularity surface for Real-facing Cauchy reads. It sits over the dyadic tolerance data of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet binds those siblings into one displayed route from dyadic tolerance requests to finite windows, regular Cauchy readback, and Real sealing before any completion consumer reads a modulus claim.

Definition 7701.1 (Dyadic modulus regularity carrier)**.** *A* $\mathsf{DyadicModulusRegularity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, including the dyadic radius ledger and the comparison rows used by the modulus request.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row listing exactly the inspected stream coordinates at the requested dyadic tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed window into a regular rational Cauchy read.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the dyadic tolerance, finite window, and regular readback rows have been displayed.*
5. *H records componentwise* hsame *transport for D, W, R, E and the structural rows.*
6. *C records displayed* Cont *replay from dyadic tolerance data through finite windows, regular readback, and the real seal.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicModulusRegularity}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusRegularity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient streams, selected representatives, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7701.2 (Dyadic modulus regularity NameCert obligations)**.** *For every accepted* DyadicModulusRegularity *carrier* $M = (D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusRegularity}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission, finite stream-window exposure, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming. No consumer of the packet can read a modulus obligation without first exposing* D, W, R, E, H, C, P, N *in that order.*

Proof. Project the carrier of Definition 7701.1. The dyadic tolerance row D is the only source of radius data, and the finite window row W is the only schedule by which that radius is inspected.

The regular readback row R is downstream of W, so any Cauchy-facing read has already exposed the finite stream coordinates it uses. The real seal row E is downstream of D, W, R, and therefore cannot be consumed as a host limit detached from the displayed modulus route.

Transport, replay, provenance, and naming are carried only by H, C, P, N. These rows preserve the same finite route and do not add quotient streams, selected representatives, trichotomy, countable choice, host real equality, or ambient completeness. The listed rows therefore exhaust the local NameCert obligation surface. □

Theorem 7701.3 (Dyadic modulus regularity stream handoff)**.** *Every accepted* $\mathsf{DyadicModulusRegularity}^{\uparrow}$ *packet hands a Real-facing regularity read to downstream consumers only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The handoff exports a displayed dyadic-modulus route for one finite regular Cauchy read; it does not export quotient-stream equality, a selected representative, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $M = (D, W, R, E, H, C, P, N)$. A downstream consumer sees the real seal only at E.

The row E is named after D, W, R have been displayed. Thus the consumer must first read the dyadic tolerance ledger, then the finite stream window selected for that tolerance, and then the regular rational readback attached to the same window.

The structural rows H, C, P, N replay and name this route without adding another equality or completeness coordinate. Hence the public handoff is the finite dyadic-modulus regularity route stated above, not any of the excluded host or choice principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulusRegularity}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicModulusRegularity}^{\uparrow}}$, seed)

Scope closed: Definition 7701.1, Definition 7701.2, and Definition 7701.3 seed the finite dyadic-modulus regularity route over DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicModulusRegularity}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DyadicModulusRegularityUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance coverage, finite window readback, regular Cauchy handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicModulusRegularityUp.

Constructive story: A $\mathsf{DyadicModulusRegularity}^{\uparrow}$ packet is generated from DyadicRatCore tolerance data, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7702

A Concrete Naming Certificate for $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$

Following Isbell and Mrowka on almost-disjoint families and Ψ-spaces, the $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ packet records the classical countable-row construction as finite BEDC topology data. It sits beside the indexed-open carrier of chapter 88, the observable-open classifier of chapter 1860, the membership discipline of chapter 3668, and the finite observation windows of chapter 126. The packet names isolated natural-number rows and almost-disjoint-family point rows only through displayed neighbourhood evidence; it does not import a host powerset of $\mathsf{Nat}^{\uparrow}$, a selected infinite almost-disjoint family, quotient point equality, or choice over neighbourhood tails.

Definition 7702.1 (Isbell–Mrowka Psi-space carrier)**.** *An* $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Psi = (N, A, T, I, F, O, S, W, H, C, P, L)$$

with the following displayed rows:

1. *N is the isolated natural-number row, recording the displayed natural-number names whose singleton neighbourhoods are admitted.*
2. *A is the almost-disjoint-family point row, recording the displayed family-name rows and their finite intersection witnesses without selecting an ambient infinite family.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *admission row, exposing the indexed open rows generated by isolated singletons and cofinite-tail neighbourhoods around family points.*
4. *I is the isolated-point exactness row: a natural-number point is open exactly through its displayed singleton row and not through a hidden host subset.*
5. *F is the cofinite-tail neighbourhood exactness row. For a displayed family point, an accepted neighbourhood is a displayed family row together with a finite forbidden subrow of that family; the remaining tail is read only through the carried membership rows.*
6. *O is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open handoff, recording every open-membership read as a displayed truth row.*
7. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership-provenance row for natural-number membership, family membership, finite forbidden tails, and generated open rows.*
8. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window replay row for the inspected singleton and tail-neighbourhood requests.*

9. *H records componentwise* hsame *transport for isolated rows, family rows, topology admission, exactness rows, observable truth, membership provenance, finite-window replay, package provenance, and local naming.*
10. *C records displayed* Cont *replay from point-row admission through topology rows, observable-open handoff, membership provenance, and finite-window inspection.*
11. *P is the* Pkg *provenance over* $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, A, T, I, F, O, S, W, H, C, P, L$. It has no coordinate for a host powerset topology, an unrestricted subset of $\mathsf{Nat}^{\uparrow}$*, a choice-selected almost-disjoint family, a quotient of points, or an external separation theorem.*

Lean4 ✓ `BEDC.Derived.IsbellMrowkaPsiSpaceUp.IsbellMrowkaPsiSpaceTasteGate single carrier alignment`

Theorem 7702.2 (Isbell–Mrowka Psi-space NameCert obligations)**.** *For every accepted* $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ *carrier*

$$\Psi = (N, A, T, I, F, O, S, W, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}}$ *surface consists exactly of topology-row admission through T, isolated-point exactness through I, cofinite-tail neighbourhood exactness through F, observable-open handoff through O, SetLike membership provenance through S, StreamName finite-window replay through W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L. Natural-number rows and almost-disjoint-family point rows are admitted only through N and A, and no obligation is stored outside the displayed tuple.*

Proof. Project the carrier of Definition 7702.1. The rows N and A are the only point-source rows, so an accepted packet can expose either an isolated natural-number point or a family point only through those displayed histories.

The topology obligations are local. Row T admits the generated open rows; row I fixes singleton exactness for isolated natural-number points; and row F fixes cofinite-tail neighbourhood exactness for displayed family points by carrying the finite forbidden subrow.

The observational obligations then pass through O, S, W. Open membership is read as a $\mathsf{SierpinskiSpace}^{\uparrow}$ truth row, all element and tail claims are sourced by $\mathsf{SetLike}^{\uparrow}$ membership provenance, and every inspected singleton or tail request is replayed through a finite $\mathsf{StreamName}^{\uparrow}$ window.

Finally H, C, P, L transport, replay, source, and name the same finite route. Since the classifier has no coordinate for host powersets, unrestricted subsets of $\mathsf{Nat}^{\uparrow}$, choice-selected almost-disjoint families, quotient points, or ambient separation theorems, the listed rows exhaust the local NameCert surface. □

Theorem 7702.3 (Isbell–Mrowka Psi-space topology route)**.** *Every* $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ *packet receives neighbourhood data only through the finite route*

$$N, A \longrightarrow T \longrightarrow I, F \longrightarrow O, S, W \longrightarrow H, C, P, L.$$

In particular, an isolated-point read must display the singleton row I, and a family-point neighbourhood read must display the cofinite-tail row F with its finite forbidden subrow before any observable-open or membership-facing read is exported.

Proof. Start with the point-source rows. The natural-number row N and the family row A determine which displayed points can be inspected; no ambient point set is available outside them.

Next read T, I, F. The topology admission row exposes only generated open rows, while the two exactness rows distinguish isolated singleton reads from family-point cofinite-tail reads.

Then consume O, S, W. These rows make each open-membership observation visible as a Sierpinski truth read, bind membership and finite forbidden tails to SetLike provenance, and replay the inspected requests through StreamName windows.

The structural rows H, C, P, L preserve the same route. They add no host powerset, unrestricted almost-disjoint family, selected infinite subfamily, quotient point equality, or choice principle. □

Falsifiable prediction: An accepted $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ consumer is refuted if it reads an isolated singleton, a family-point cofinite-tail neighbourhood, an observable-open truth value, or a membership claim without first displaying the corresponding $N, A, T, I, F, O, S, W, H, C, P, L$ rows.

Independence witness: topology, sierpinski-space, setlike, streamname: no listed sibling is bijective to $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ because the packet jointly carries isolated natural-number point rows, almost-disjoint-family point rows, generated topology admission, singleton exactness, cofinite-tail neighbourhood exactness, observable-open handoff, membership provenance, finite-window replay, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}$ packet is generated from isolated natural-number rows, almost-disjoint-family point rows, TopologyUp admission, singleton exactness, cofinite-tail neighbourhood exactness, SierpinskiSpaceUp observable-open handoff, SetLikeUp membership provenance, StreamNameUp finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7702.1, Definition 7702.2, and Definition 7702.3 seed the finite Psi-space route over isolated rows, almost-disjoint-family point rows, topology admission, observable truth, membership provenance, and finite windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IsbellMrowkaPsiSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powerset topologies, unrestricted subsets of NatUp, selected infinite almost-disjoint families, quotient point equality, ambient separation theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-row admission, isolated-point exactness, cofinite-tail neighbourhood exactness, observable-open handoff, membership provenance, finite-window replay, transport, replay, provenance, and local naming for BEDC.Derived.IsbellMrowkaPsiSpaceUp.

Chapter 7703

A Concrete Naming Certificate for $\mathsf{BaireClassOneRealFunction}^{\uparrow}$

Following Baire and Bishop-Bridges on pointwise limits of continuous real functions, the $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ packet records a constructive Baire class one real-valued function as an explicit sequence of continuous real functions with finite pointwise readback. It sits after the real-name carrier of chapter 38, the stream-name interface of chapter 126, the regular-rational readback surface of chapter 301, the continuous-map packet of chapter 130, and the pointwise stream metric route of chapter 4139. The packet gives later $\mathsf{Real}^{\uparrow}$ consumers a sequence-of-continuous-functions target without host function equality, a quotient function space, selected representatives, countable choice, or an ambient measurability theorem.

Definition 7703.1 (Baire class one real-function carrier)**.** *A* $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, S, A, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the source real-domain row whose point requests are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing the inspected stages of the approximating function sequence.*
3. *A is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row assigning each inspected stage to a displayed continuous real-valued map.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to each pointwise real output produced by the inspected continuous maps.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the pointwise limiting value.*
6. *M is the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *row comparing the stage outputs only through finite pointwise windows.*
7. *H records componentwise* hsame *transport for source, schedule, continuous-map, readback, Real-seal, metric, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point request through $X, S, A, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireClassOneRealFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{PointwiseStreamMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BaireClassOneRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host equality of functions, a completed function space quotient, an externally selected pointwise limit, countable choice, arbitrary measurability, or an ambient theorem identifying all Baire class one maps.

Theorem 7703.2 (Baire class one real-function NameCert obligations)**.** *For every accepted* BaireClassOneRealFunct *carrier $F = (X, S, A, R, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireClassOneRealFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the source real-domain row, the* $\mathsf{StreamName}^{\uparrow}$ *stage schedule, the* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *comparison row, componentwise transport, displayed replay, package provenance, and local naming. Every downstream real-analysis consumer must consume those rows before treating the packet as a Baire class one real-valued function.*

Proof. Project the carrier of Definition 7703.1. The accepted packet exposes only the source row, inspected stage schedule, continuous-map sequence row, regular-rational readback, Real seal, pointwise metric row, transport, replay, provenance, and local naming.

The constructive pointwise-limit route starts with X and S. A consumer first names the point request and the inspected stages, then reads the continuous-map row A at those stages.

The output route is then forced through R, E, M. The regular-rational readback records the displayed numerical data, the Real seal records the terminal pointwise value, and the pointwise stream metric row compares the stage outputs only through finite windows.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores host function equality, a quotient function space, selected representatives, countable choice, arbitrary measurability, or an ambient classification theorem, the listed obligations exhaust the local certificate surface. □

Theorem 7703.3 (Baire class one sequence handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, completion-facing, or pointwise-metric consumer of an accepted* $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{PointwiseStreamMetric}^{\uparrow}.$$

Thus the Baire class one row is a displayed sequence of continuous real-valued maps with pointwise readback, not a host function equality class, a selected limit map, or an imported measurable-function hierarchy.

Proof. Begin with a downstream consumer. Carrier admission forces the consumer to read the $\mathsf{StreamName}^{\uparrow}$ schedule before any stage of the approximating sequence can be inspected.

The inspected stage is then a $\mathsf{ContinuousMap}^{\uparrow}$ row. Its numerical output is not exposed directly as a host function value; it passes through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The pointwise comparison is supplied by $\mathsf{PointwiseStreamMetric}^{\uparrow}$. That row keeps finite windows visible while the sequence handoff is transported and replayed by H and C.

Consequently the consumer receives only the displayed sequence route over the named sibling packets. The carrier inventory excludes host function equality, quotient function spaces, selected pointwise limits, countable choice, and imported measurability structure. □

Falsifiable prediction: An accepted $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ consumer is refuted if it treats a pointwise limit as public before displaying the StreamName stage schedule, ContinuousMap sequence row, RegSeqRat readback, Real seal, PointwiseStreamMetric comparison, transport, replay, provenance, and local NameCert rows.

Independence witness: real, streamname, regseqrat, continuousmap, pointwisestreammetric: none is bijective because $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ binds a real-domain source, explicit stage schedule, continuous real-valued maps, pointwise readback, Real seal, pointwise metric comparison, transport, replay, provenance, and local NameCert rows in one packet. Each sibling supplies one constituent surface, while only this packet names the sequence-of-continuous-functions route.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireClassOneRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ packet is generated from a RealUp source row, a StreamName finite stage schedule, ContinuousMap sequence rows, RegSeqRat pointwise readback, a RealUp seal, a PointwiseStreamMetric comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireClassOneRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7703.1, Definition 7703.2, and Definition 7703.3 seed the explicit sequence-of-continuous-real-functions route over RealUp, StreamNameUp, RegSeqRatUp, ContinuousMapUp, PointwiseStreamMetricUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireClassOneRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function equality, quotient function spaces, selected pointwise limits, countable choice, arbitrary measurability, or a full Baire hierarchy.

Upgrade path: Obligation closure requires checked carrier admission, stage-schedule exactness, continuous-map sequence exposure, pointwise readback, Real seal nonescape, pointwise metric handoff, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BaireClassOneRealFunctionUp.

Chapter 7704

A Concrete Naming Certificate for $\mathsf{StationaryDistribution}^{\uparrow}$

Following Kemeny–Snell finite Markov-chain terminology, the $\mathsf{StationaryDistribution}^{\uparrow}$ packet records an invariant law for a displayed finite transition matrix. It sits downstream of the transition-packet surface of chapter 189, the probability measure rows of chapter 184 and chapter 186, the finite matrix carrier of chapter 49, and the scalar seal of chapter 38. The packet names the finite invariant-law surface only; it is not an ergodic theorem, limiting distribution theorem, Perron–Frobenius argument, or host stochastic-process construction.

Definition 7704.1 (Stationary distribution carrier)**.** *A* $\mathsf{StationaryDistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Pi = (S, M, \pi, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state ledger over a displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *transition row whose entries are nonnegative* $\mathsf{Real}^{\uparrow}$ *scalars and whose displayed rows sum to the probability unit.*
3. *π is the* $\mathsf{Distribution}^{\uparrow}$ *law row on the same finite state ledger.*
4. *L is the finite left-invariance ledger recording the row equations*

$$\pi_j = \sum_{i \in S} \pi_i M_{ij}$$

   for every displayed state j.

5. *E records the endpoint compatibility between the transition row, the probability source, and the displayed law row.*
6. *H records componentwise* hsame *transport for S, M, π, L, E and the local naming row.*
7. *C records displayed* Cont *replay from the Markov transition packet through the matrix row and invariant-law equations.*
8. *P is the* Pkg *provenance over* $\mathsf{StationaryDistribution}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryDistribution}^{\uparrow}}$ *row.*

The carrier exports no convergence rate, recurrence classifier, irreducibility argument, uniqueness proof, host limiting law, countable-choice selector, or standard Markov-kernel bridge.

Theorem 7704.2 (Stationary distribution NameCert obligations)**.** *For the carrier of Definition 7704.1, the local* $\mathsf{NameCert}_{\mathsf{StationaryDistribution}^{\uparrow}}$ *surface consists exactly of carrier admission for* $\Pi = (S, M, \pi, L, E, H, C, P, N)$*, finite state exposure through* S*, transition-matrix admission through* M*, invariant-law admission through* π*, finite left-invariance equations through* L*, endpoint compatibility through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. Its coordinates display the finite state ledger, matrix transition row, law row, invariance equations, endpoint compatibility, transport, replay, provenance, and local name.

The dependency certificates account for the data-bearing rows. The $\mathsf{ProbSpace}^{\uparrow}$ row supplies the source ledger, $\mathsf{Mat}^{\uparrow}$ supplies the finite transition entries, $\mathsf{Real}^{\uparrow}$ supplies the scalar seals, and $\mathsf{Distribution}^{\uparrow}$ supplies the law row π.

The remaining rows transport, replay, source, and name the same finite data. Since the carrier contains no convergence, recurrence, uniqueness, limiting, choice, or standard-kernel coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7704.3 (Stationary distribution invariance scope)**.** *Every invariant-law read of an accepted* $\mathsf{StationaryDistribution}^{\uparrow}$ *carrier factors through the finite route*

$$S, M, \pi, L, E, H, C, P, N.$$

The read inspects the finite $\mathsf{Mat}^{\uparrow}$ *transition row and the* $\mathsf{Distribution}^{\uparrow}$ *law row before it can use the equations in* L*, and those equations are interpreted only through the displayed* $\mathsf{Real}^{\uparrow}$ *scalar seals and the shared* $\mathsf{ProbSpace}^{\uparrow}$ *state ledger.*

Proof. Begin with the carrier rows S, M, π. They fix the finite state ledger, the transition matrix, and the law whose invariance is being certified.

Next read the equation ledger L. Each displayed equation has endpoints in the same finite state ledger and uses only the transition entries and law weights already exposed by M and π.

Finally pass through E, H, C, P, N. These rows keep the endpoint compatibility, transport, replay, provenance, and naming attached to the same finite route. Thus an invariant-law consumer has no alternate path through a host chain, limiting distribution, or uniqueness theorem. □

Theorem 7704.4 (Stationary distribution Markov consumer handoff)**.** *Whenever a* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet consumes an accepted* $\mathsf{StationaryDistribution}^{\uparrow}$ *carrier, the handoff reads the displayed finite state ledger* S*, transition row* M*, invariant law* π*, invariance ledger* L*, endpoint row* E*, and structural rows* H, C, P, N*. The consumer may treat* π *as a stationary law for the displayed finite transition packet, but it does not receive a convergence theorem, recurrence decomposition, irreducibility witness, uniqueness classifier, or standard Markov-kernel object.*

Proof. Use the Markov transition-packet source boundary from Definition 189.5. It keeps the probability source, random-variable rows, distribution laws, and transition rows inside the displayed $\mathsf{MarkovChain}^{\uparrow}$ packet.

Attach the stationary carrier along the common finite transition row. The matrix row M supplies the finite transition entries, while π and L record the invariant law and its finite equations over the same state ledger.

The structural rows then replay the handoff without adding host stochastic process data. Since neither carrier contains a convergence, recurrence, irreducibility, uniqueness, or standard-kernel coordinate, the Markov consumer receives exactly the displayed stationary-law surface. □

Falsifiable prediction: Every accepted StationaryDistribution$^{\uparrow}$ packet exposes a finite invariant-law equation ledger over a displayed transition matrix and exports no convergence, recurrence, irreducibility, uniqueness, or host limiting-law coordinate.

Independence witness: markovchain, distribution, probspace, matrix, real: no listed sibling is bijective to StationaryDistribution$^{\uparrow}$ because this packet binds one finite transition matrix, one law row, and one left-invariance equation ledger in a single Markov-consumer handoff. The MarkovChain sibling lacks a distinguished invariant law; DistributionUp lacks transition equations; ProbSpaceUp lacks matrix dynamics; MatrixUp lacks probability-law admission; RealUp lacks the finite invariant-law route.

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryDistribution}^{\uparrow}}$.

Constructive story: A StationaryDistribution$^{\uparrow}$ packet is generated from a finite state ledger, a transition MatrixUp row, a DistributionUp law row, a finite left-invariance equation ledger, endpoint compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StationaryDistribution}^{\uparrow}}$, seed)

Scope closed: Definition 7704.1, Definition 7704.2, Definition 7704.3, and Definition 7704.4 seed the finite invariant-law surface over MarkovChain, ProbSpace, Distribution, Matrix, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StationaryDistribution}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim convergence to stationarity, recurrence decomposition, irreducibility, uniqueness, limiting distributions, standard Markov kernels, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, matrix stochasticity, invariant-law equation coverage, endpoint compatibility, and local NameCert registration for BEDC.Derived.StationaryDistributionUp.

Chapter 7705

A Concrete Naming Certificate for $\mathsf{ReflectionGapAudit}^{\uparrow}$

The $\mathsf{ReflectionGapAudit}^{\uparrow}$ packet realizes the reflection-gap surface of Definition 9414.413 as a finite audit carrier for observation-reflection and MetaCIC self-study. It sits beside the observation-reflection packet of chapter 1126, the finite observation-reflection surface of chapter 5109, the reflective inquiry carrier of chapter 1151, and the reflection-gap boundary of chapter 1161. The audit records inquiry rows, ledger rows, continuation rows, non-finalization rows, transport, replay, provenance, and local naming without asserting a total truth predicate, final oracle, physical-formal identity, or closed meta loop.

Definition 7705.1 (Reflection gap audit carrier)**.** *A* $\mathsf{ReflectionGapAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, G, L, Q, F, T, R, P, N)$$

with the following displayed rows:

1. *I is the finite inquiry row, read from the observation-reflection source, finite trace, signature, classifier, and package data.*
2. *G is the reflection-gap row, naming the displayed difference between finite certified self-description and total truth.*
3. *L is the audit ledger row, recording accepted finite self-description claims, rejected totalizing claims, and still-open audit residues.*
4. *Q is the continuation row, listing the finite continuations under which the audit may be replayed.*
5. *F is the non-finalization row, recording that no row in the packet certifies final truth, total fit, closed prediction, or internal truth for the whole system.*
6. *T records componentwise* hsame *transport for inquiry, gap, ledger, continuation, non-finalization, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay of the route from inquiry rows through the audit ledger and open continuation into the non-finalization row.*
8. *P is the* Pkg *provenance over* $\mathsf{ReflectionGapAudit}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, $\mathsf{FiniteObservationReflection}^{\uparrow}$, $\mathsf{ReflectiveInquiry}^{\uparrow}$, $\mathsf{ReflectionGapBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectionGapAudit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, L, Q, F, T, R, P, N$. It has no coordinate for a total truth predicate, final physical oracle, final MetaCIC oracle, physical-formal source identity, closed continuation exhaustion, or host-level validation.

Theorem 7705.2 (Reflection gap audit non-escape)**.** *Every accepted* $\mathsf{ReflectionGapAudit}^{\uparrow}$ *packet exports only the finite route*

$$I, G, L, Q, F, T, R, P, N.$$

A consumer may read the inquiry row, reflection-gap row, audit ledger, open-continuation row, non-finalization row, transport, replay, provenance, and local NameCert row. It cannot recover a total truth predicate, final physical oracle, final MetaCIC oracle, source-domain identity, closed continuation exhaustion, or host validation from the packet.

Lean4 ✓ `BEDC.Derived.ReflectionGapAuditUp.ReflectionGapAuditNonescape`

Proof. Inspect the carrier of Definition 7705.1. The only readable coordinates are the displayed inquiry, gap, ledger, continuation, non-finalization, transport, replay, provenance, and local naming rows.

The audit ledger L records finite claims and rejected totalizing claims. The continuation row Q records replayable continuations, while the non-finalization row F keeps the finality refusal as data inside the packet.

The structural rows T, R, P, N preserve the same finite route. Since no row stores total truth, final physical or MetaCIC oracles, source identity, closed continuation exhaustion, or host validation, those endpoints cannot escape from the audit packet. □

Theorem 7705.3 (Reflection gap audit bridge)**.** *The* $\mathsf{ReflectionGapAudit}^{\uparrow}$ *carrier bridges the observation-reflection vision to concrete NameCert data by reading* $\mathsf{ObservationReflectionPacket}^{\uparrow}$*,* $\mathsf{FiniteObservationReflection}^{\uparrow}$*,* $\mathsf{ReflectiveInquiry}^{\uparrow}$*, and* $\mathsf{ReflectionGapBoundary}^{\uparrow}$ *only through finite inquiry, gap, ledger, continuation, non-finalization, transport, replay, provenance, and local naming rows. The bridge is an audit surface, not a totalizing truth predicate or a final oracle.*

Proof. Start with the originating reflection-gap definition. Its role is the difference between finite certified self-description and total truth, so the concrete audit needs both a finite inquiry row and a non-finalization row.

The sibling chapters supply the visible rows. $\mathsf{ObservationReflectionPacket}^{\uparrow}$ supplies source, history, signature, classifier, package, ledger, and continuation roles; $\mathsf{FiniteObservationReflection}^{\uparrow}$ supplies the finite trace route; $\mathsf{ReflectiveInquiry}^{\uparrow}$ supplies the paired physical and formal inquiry route; and $\mathsf{ReflectionGapBoundary}^{\uparrow}$ supplies the boundary row refusing total truth.

The audit carrier repacks those rows as $I, G, L, Q, F, T, R, P, N$. Because the repacking preserves the open continuation and non-finalization rows, the bridge records a finite audit surface and not a final truth oracle, source identity claim, or closed meta loop. □

Falsifiable prediction: Every accepted $\mathsf{ReflectionGapAudit}^{\uparrow}$ read exposes finite inquiry, reflection-gap, audit ledger, open-continuation, non-finalization, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows before any reflection-gap audit judgment is exported.

Independence witness: observationreflectionpacket, finite-observation-reflection, reflectiveinquiry, reflectiongapboundary: $\mathsf{ReflectionGapAudit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the audit ledger and non-finalization rows with the finite inquiry, reflection-gap, continuation, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ReflectionGapAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{ReflectionGapAudit}^{\uparrow}$ packet is generated from finite inquiry, reflection-gap, audit-ledger, open-continuation, non-finalization, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReflectionGapAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7705.1, Definition 7705.2, and Definition 7705.3 seed the finite reflection-gap audit surface over inquiry, gap, ledger, continuation, non-finalization, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReflectionGapAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total truth, final physical or MetaCIC oracles, physical-formal source identity, closed continuation exhaustion, host validation, Lean verification, or bridge checking for $\mathsf{ReflectionGapAudit}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, inquiry exposure, gap-ledger exactness, continuation replay, non-finalization non-escape, transport, provenance, and local naming for BEDC.Derived.ReflectionGapAuditUp.

Chapter 7706

A Concrete Naming Certificate for $\mathsf{DyadicSpread}^{\uparrow}$

Following Bishop on located real approximations and Kleene on finite spread laws, the $\mathsf{DyadicSpread}^{\uparrow}$ packet records a finite dyadic-prefix spread whose windows may be read by stream, regular-sequence, real, tree, and Cantor-space consumers. It is a finite-prefix sibling of $\mathsf{RegularCauchySpread}^{\uparrow}$ in chapter 2072, the dyadic interval net of chapter 5966, and the regular sequence space of chapter 5965. The packet names displayed prefix windows; it does not select an infinite branch, complete a real, decide bar coverage, or claim compactness.

Definition 7706.1 (Dyadic spread finite-prefix carrier)**.** *A* $\mathsf{DyadicSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (P, W, I, S, R, A, K, C, H, Q, L, N)$$

with the following displayed rows:

1. *P is the finite prefix tree row, listing dyadic words closed under initial segment;*
2. *W is the finite window schedule, recording which prefix lengths are visible to a consumer;*
3. *I is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *interval row attached to each displayed prefix;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that reads a prefix window as a finite name segment;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the dyadic windows whose moduli are displayed in W;*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular readback row has supplied the displayed tolerance data;*
7. *K is the* $\mathsf{KleeneTree}^{\uparrow}$ *law row naming the finite spread-law closure under prefix extension and rejection;*
8. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *cylinder row for binary-prefix consumers that read only the displayed branch window;*
9. *H stores componentwise* hsame *transport for prefix, window, dyadic interval, stream-name, regular readback, real-seal, tree-law, and cylinder rows;*
10. *Q stores the finite* Cont *replay routes from prefixes to dyadic intervals, stream windows, regular readback, real sealing, tree-law checks, and cylinder reads;*

11. *L is the* Pkg *provenance over* DyadicSpread$^\uparrow$, DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, KleeneTree$^\uparrow$, CantorSpace$^\uparrow$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{DyadicSpread}^\uparrow}$ *row.*

The classifier compares two carriers only by transporting the displayed finite prefix tree, dyadic windows, consumer rows, replay routes, provenance, and local certificate. No coordinate stores an infinite choice sequence, a quotient of branch names, or ambient completeness of Real$^\uparrow$*.*

Theorem 7706.2 (Dyadic spread window stability)**.** *For every accepted* DyadicSpread$^\uparrow$ *carrier $D = (P, W, I, S, R, A, K, C, H, Q, L, N)$, each finite window read used by the* StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, KleeneTree$^\uparrow$, *and* CantorSpace$^\uparrow$ *rows is stable under the displayed* hsame *transport and replays through the displayed* Cont *routes. Thus any consumer that reads a prefix window after transport receives the same finite dyadic interval data, the same regular readback obligation, the same real-seal boundary, the same tree-law decision, and the same Cantor cylinder row, with no route to an uncarried branch or completion principle.*

Proof. Project the carrier of Definition 7706.1. The only rows available to a window consumer are the finite prefix tree P, the window schedule W, the dyadic interval row I, the stream, regular, real, tree, and cylinder rows S, R, A, K, C, and the support rows H, Q, L, N.

Use the transport row H componentwise. A transported prefix remains a displayed prefix of the transported finite tree, a transported dyadic interval remains the corresponding DyadicRatCore$^\uparrow$ window, and the stream-name, regular readback, real-seal, tree-law, and Cantor-cylinder rows are transported with the same finite index in W.

Use the continuation row Q to replay the same read after transport. The replay starts from the displayed prefix window and reaches only the displayed rows I, S, R, A, K, C; it cannot create an infinite branch, select a representative stream, or insert a completion theorem.

Therefore finite window transport preserves exactly the visible dyadic-prefix data and its consumer-facing obligations. The result is the window stability needed by RegularCauchySpread$^\uparrow$ and RegularSequenceSpace$^\uparrow$ consumers, while the chapter remains a finite NameCert packet. □

Lean4 ✓ `BEDC.Derived.DyadicSpreadUp.DyadicSpreadWindowStability`

Falsifiable prediction: Every accepted DyadicSpread$^\uparrow$ consumer read is determined by a displayed finite prefix window. A read that selects an infinite branch, imports countable choice, or uses ambient Real$^\uparrow$ completeness is not a valid read of the carrier of Definition 7706.1.

Independence witness: regular-cauchy-spread, regular-sequence-space, dyadic-interval-net, kleene-tree, cantor-space: DyadicSpread$^\uparrow$ is not bijective to any listed sibling because it stores the finite dyadic-prefix spread law together with stream-name, regular readback, real-seal, tree-law, and cylinder consumer rows in one packet.

Closure status for NameCert$_{\text{DyadicSpread}^\uparrow}$.

Constructive story: A DyadicSpread$^\uparrow$ packet is generated from a finite prefix tree, displayed dyadic windows, DyadicRatCore interval rows, StreamName windows, RegSeqRat readback, a Real seal boundary, KleeneTree spread-law checks, CantorSpace cylinders, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicSpread}^\uparrow}$, seed)

Scope closed: Definition 7706.1 and Definition 7706.2 seed the finite dyadic-prefix spread route over DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, KleeneTree$^\uparrow$, and CantorSpace$^\uparrow$ rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicSpread}^\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not select an infinite branch, prove bar induction, close compactness, assert ambient real completeness, or provide Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix-law exactness, dyadic window replay, regular readback, real-seal boundary, cylinder stability, provenance accounting, and non-escape for the same packet.

Chapter 7707

A Concrete Naming Certificate for $\mathsf{HoarePowerdomain}^{\uparrow}$

Following Hoare and Smyth domain-theoretic powerdomain semantics, the $\mathsf{HoarePowerdomain}^{\uparrow}$ packet records the lower powerdomain side of finite nondeterministic observation as a hit-oriented NameCert surface. It sits below the open-set classifier of chapter 5950, the order carrier of chapter 53, and the compact-family comparison route of chapter 2375. The packet gives domain-topology consumers a lower hit window without importing an upper powerdomain, a convex powerdomain law, ideal completion, or host-level set equality.

Definition 7707.1 (Hoare powerdomain carrier)**.** *A* $\mathsf{HoarePowerdomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (D, O, K, L, W, R, T, C, P, N)$$

with the following displayed rows:

1. *D is the underlying domain row, read through the ordered carrier surface of* $\mathsf{Poset}^{\uparrow}$*.*
2. *O is the Scott-open observation row, read through* $\mathsf{ScottTopology}^{\uparrow}$*.*
3. *K is the compact-or-finite lower witness row, comparable with the hyperspace route of* $\mathsf{VietorisTopology}^{\uparrow}$*.*
4. *L is the lower hit predicate row: an observation is public only when the packet displays a witness that the lower set meets the inspected open.*
5. *W is the finite hit-window ledger listing the inspected open rows and the associated witnesses.*
6. *R records monotone readback along the displayed order row of D.*
7. *T records componentwise* hsame *transport for $D, O, K, L, W, R, C, P, N$.*
8. *C records displayed* Cont *replay from the domain row through Scott opens, lower hits, finite windows, order readback, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HoarePowerdomain}^{\uparrow}$*,* $\mathsf{ScottTopology}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*,* $\mathsf{VietorisTopology}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HoarePowerdomain}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for upper misses, convex combination, ideal completion, saturated-set equality, arbitrary unions of host subsets, or a quotient by extensional set equality.

Theorem 7707.2 (Hoare powerdomain NameCert obligations)**.** *For every accepted* $\mathsf{HoarePowerdomain}^{\uparrow}$ *carrier* $H = (D, O, K, L, W, R, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HoarePowerdomain}^{\uparrow}}$ *surface consists exactly of carrier admission for the domain row, Scott-open observation row, compact-or-finite lower witness row, lower hit predicate row, finite hit-window ledger, monotone readback row, componentwise transport, displayed replay, package provenance, and local naming. Every downstream domain-topology consumer must read those rows before treating the packet as a lower powerdomain object.*

Proof. Project the carrier of Definition 7707.1. The accepted packet exposes the domain row, Scott-open row, compact-or-finite witness row, lower hit predicate, finite hit-window ledger, monotone readback, transport, replay, provenance, and local naming.

Read the lower observation route first. The domain row D supplies the ordered carrier surface, while O restricts observations to displayed Scott opens and K supplies the finite or compact witness data that can meet such opens.

The hit information is then forced through L, W, R. The row L records lower hit admission, W lists the inspected windows and witnesses, and R keeps the readback monotone along the displayed order.

Finally T, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores upper misses, convex powerdomain operations, ideal completion, arbitrary host subsets, or set-extensional quotient data, the listed rows exhaust the local certificate surface. □

Theorem 7707.3 (Hoare lower hit window)**.** *Every* $\mathsf{ScottTopology}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*, or* $\mathsf{VietorisTopology}^{\uparrow}$ *consumer of an accepted* $\mathsf{HoarePowerdomain}^{\uparrow}$ *carrier must read the finite lower-hit route*

$$\mathsf{Poset}^{\uparrow} \longmapsto \mathsf{ScottTopology}^{\uparrow} \longmapsto \mathsf{VietorisTopology}^{\uparrow} \longmapsto \mathsf{HoarePowerdomain}^{\uparrow}.$$

Thus the public observation is a displayed lower hit of an inspected Scott open, not an upper miss, a convex powerdomain comparison, a completed ideal, or a host subset equality claim.

Proof. Begin with the ordered carrier. The consumer receives D before any lower-powerdomain observation is available, so the packet is anchored in the displayed $\mathsf{Poset}^{\uparrow}$ surface rather than in host subset equality.

The open observation is then a $\mathsf{ScottTopology}^{\uparrow}$ row. The lower hit predicate can be read only against such an inspected open, and the compact-or-finite witness row supplies the visible meet data comparable with $\mathsf{VietorisTopology}^{\uparrow}$.

The finite window ledger W prevents the hit from becoming an arbitrary host-level union or a hidden ideal completion. Transport, replay, provenance, and local naming preserve the same lower-hit packet for downstream consumers, so the only public observation is the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{HoarePowerdomain}^{\uparrow}$ consumer is refuted if it reports a lower hit before displaying the Poset domain row, Scott-open observation row, compact-or-finite witness row, finite hit-window ledger, monotone readback, transport, replay, provenance, and local NameCert rows.

Independence witness: scotttopology, poset, vietoristopology: none is bijective because $\mathsf{HoarePowerdomain}^{\uparrow}$ binds an ordered domain row, Scott-open observation row, compact-or-finite lower witness row, lower hit predicate, finite hit-window ledger, monotone readback, transport, replay, provenance, and local NameCert row in one packet. Each sibling supplies one constituent surface, while only this packet names the lower powerdomain hit route.

Closure status for $\mathsf{NameCert}_{\mathsf{HoarePowerdomain}^{\uparrow}}$.

Constructive story: A $\mathsf{HoarePowerdomain}^{\uparrow}$ packet is generated from a Poset domain row, ScottTopology open row, compact-or-finite lower witness row, lower hit predicate row, finite hit-window ledger, monotone readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoarePowerdomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7707.1, Definition 7707.2, and Definition 7707.3 seed the lower hit route over PosetUp, ScottTopologyUp, VietorisTopologyUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoarePowerdomain}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.HoarePowerdomainUp`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, upper misses, convex powerdomain laws, ideal completion, host subset equality, arbitrary unions of host subsets, or an extensional quotient surface.

Upgrade path: Obligation closure requires checked carrier admission, Scott-open hit exactness, compact-or-finite lower witness exposure, finite hit-window readback, monotone order transport, displayed replay, provenance, and local naming for BEDC.Derived.HoarePowerdomainUp.

Chapter 7708

A Concrete Naming Certificate for $\mathsf{GromovCompactness}^{\uparrow}$

Following Gromov's compactness theorem for families of compact metric spaces, the $\mathsf{GromovCompactness}^{\uparrow}$ packet records the finite epsilon-net compactness route as NameCert data. It sits beside the compact metric carrier of chapter 129, the Gromov–Hausdorff naming surface of chapter 2104, and the Gromov–Hausdorff distance packet of chapter 5200. The packet packages finite radius ledgers and finite net comparisons; it does not import open-cover compactness, selected subsequences, quotient metric spaces, or a host compactness theorem.

Definition 7708.1 (Gromov compactness carrier)**.** *A* $\mathsf{GromovCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, E, N, D, G, H, C, P, L)$$

whose rows are the family index row F, the finite epsilon-scale row E, the finite net row N, the distance-comparison row D, the $\mathsf{GromovHausdorff}^{\uparrow}$ *handoff row G, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{GromovCompactness}^{\uparrow}}$ *row L. The classifier compares accepted packets only through these displayed rows. It has no coordinate for an open-cover compactness theorem, a chosen subsequence, a quotient metric-space construction, a host hyperspace compactness object, or ambient completeness.*

Theorem 7708.2 (Gromov compactness NameCert obligations)**.** *For every accepted* $\mathsf{GromovCompactness}^{\uparrow}$ *carrier $K = (F, E, N, D, G, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{GromovCompactness}^{\uparrow}}$ *surface consists exactly of family-index admission through F, finite epsilon-scale admission through E, finite net admission through N, distance-comparison admission through D, Gromov–Hausdorff handoff through G, transport through H, replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 7708.1. The compactness-facing rows are F, E, N, D, G, and the structural rows are H, C, P, L.

The finite epsilon-scale row E is read before the finite net row N and distance-comparison row D. This keeps the compactness route inside inspected finite nets instead of replacing it by an open-cover compactness theorem.

Finally G, H, C, P, L hand the same finite route to the Gromov–Hausdorff and compact metric neighbours, transport it, replay it, source it, and name it. Since no carrier coordinate contains a selected subsequence, quotient metric space, host hyperspace compactness theorem, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7708.3 (Gromov compactness finite epsilon-net route)**.** *Every compactness-facing consumer of an accepted* $\mathsf{GromovCompactness}^{\uparrow}$ *carrier receives only the finite route*

$$F, E, N, D, G, H, C, P, L.$$

The route may be handed to $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{GromovHausdorff}^{\uparrow}$*, or* $\mathsf{PointedGromovHausdorff}^{\uparrow}$ *consumers as finite epsilon-net evidence with displayed distance comparisons, but it does not export open-cover compactness, selected subsequences, quotient metric spaces, host hyperspace compactness, or ambient completeness.*

Proof. Start with the family and epsilon rows F, E. They name the inspected family slice and the finite scale ledger before any compactness-facing read is public.

Read the finite net and distance-comparison rows N, D next. These rows provide the finite witnesses consumed by compact metric and Gromov–Hausdorff neighbours, while keeping the route within displayed comparisons.

The handoff and structural rows G, H, C, P, L preserve the same finite route. Therefore a downstream consumer receives finite epsilon-net compactness data, not an open-cover theorem, a selected subsequence, a quotient metric-space object, or an ambient completeness principle. □

Falsifiable prediction: A $\mathsf{GromovCompactness}^{\uparrow}$ compactness-facing read that exposes a public conclusion without the displayed family row F, epsilon-scale row E, finite-net row N, distance-comparison row D, Gromov–Hausdorff handoff row G, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: compactmetric, gromov-hausdorff, gromov-hausdorff-distance: no listed sibling is bijective to $\mathsf{GromovCompactness}^{\uparrow}$ because this packet jointly carries family indices, finite epsilon scales, finite net ledgers, distance comparisons, Gromov–Hausdorff handoff, transport, replay, provenance, and local naming while refusing open-cover compactness and selected subsequences.

Closure status for $\mathsf{NameCert}_{\mathsf{GromovCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GromovCompactness}^{\uparrow}$ packet is generated from family indices, finite epsilon-scale rows, finite net ledgers, distance comparisons, Gromov–Hausdorff handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GromovCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7708.1, Definition 7708.2, and Definition 7708.3 seed the finite epsilon-net compactness route for metric-family consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GromovCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, subsequence selection, quotient metric-space construction, host hyperspace compactness, Lean verification, or bridge checking for $\mathsf{GromovCompactness}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite epsilon-scale admission, finite net admission, distance-comparison admission, Gromov–Hausdorff handoff, transport, replay, provenance, and local naming for BEDC.Derived.GromovCompactnessUp.

Chapter 7709

A Concrete Naming Certificate for $\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}$

The $\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}$ packet realizes the residual critical-pair budget branch of the MetaCIC confluence audit map as finite NameCert data. It sits after the residual critical-pair carrier of chapter 7291, the confluence audit packet of chapter 735, and the local-diamond witness of chapter 3595. The packet records a bounded residual-frontier budget, candidate local joins, closed-check rows, obstruction retention, TasteGate readiness, replay, provenance, and local naming; it does not claim global confluence, unrestricted residual-substitution compatibility, subject reduction, quotient syntax equality, or a hidden evaluator.

Definition 7709.1 (MetaCIC residual critical-pair budget carrier)**.** *A* $\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, K, J, D, O, T, H, C, P, N)$$

whose rows are the residual critical-pair source row R, the finite budget row K, the candidate local-join row J, the closed-check row D, the obstruction row O, the TasteGate readiness row T, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}}$ *row N. The classifier compares accepted packets only through these rows. It has no coordinate for global Church–Rosser, arbitrary substitution compatibility, subject reduction, quotient syntax equality, or host evaluator soundness.*

Theorem 7709.2 (MetaCIC residual critical-pair budget obligations)**.** *For every accepted* $\mathsf{MetaCICResidualCriticalP}$ *carrier $B = (R, K, J, D, O, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}}$ *surface consists exactly of residual critical-pair source admission through R, finite budget admission through K, candidate local-join admission through J, closed-check admission through D, obstruction retention through O, TasteGate readiness through T, transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7709.1. The confluence-facing rows are R, K, J, D, O, and the structural rows are T, H, C, P, N.

The residual source row R is read before the finite budget row K and candidate local-join row J. This keeps each critical-pair read inside a displayed residual frontier rather than treating a local join as an unbounded confluence theorem.

The closed-check row D and obstruction row O then separate the inspected closed fragment from the missing residual-substitution and parallel-diamond inputs. Finally T, H, C, P, N reserve the TasteGate instance boundary, transport, replay, source, and name the same finite route. □

Theorem 7709.3 (MetaCIC residual critical-pair budget boundary)**.** *Every confluence-audit consumer of an accepted* MetaCICResidualCriticalPairBudget$^{\uparrow}$ *carrier receives only the finite route*

$$R, K, J, D, O, T, H, C, P, N.$$

The route may budget residual critical-pair inspections and candidate local joins, but it cannot erase the obstruction row O, manufacture unrestricted residual-substitution compatibility, prove a parallel diamond, export global confluence, or supply subject reduction.

Proof. Start with the residual source and finite budget rows R, K. They bind the consumer to an inspected residual frontier and a displayed budget before any candidate join is read.

Read the candidate local-join and closed-check rows J, D. These rows may admit a bounded local join for the displayed frontier, but they do not contain arbitrary substitution compatibility or a global diamond.

The obstruction and structural rows O, T, H, C, P, N preserve the boundary. The Lean Nontrivial and FieldFaithful TasteGate instances required by T must be supplied before the packet can be consumed as a checked bridge; until then the route remains finite residual-budget evidence with the obstruction visible. □

Falsifiable prediction: An accepted MetaCICResidualCriticalPairBudget$^{\uparrow}$ confluence-audit read that consumes a residual critical-pair budget without displaying rows $R, K, J, D, O, T, H, C, P, N$ refutes the carrier surface.

Independence witness: metacic-residual-critical-pair, metacic-confluence-audit-packet, metacic-local-diamond-witness: MetaCICResidualCriticalPairBudget$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries residual critical-pair source rows, finite budget rows, candidate joins, closed checks, obstruction retention, Nontrivial and FieldFaithful TasteGate readiness, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{MetaCICResidualCriticalPairBudget}^{\uparrow}}$.

Constructive story: A MetaCICResidualCriticalPairBudget$^{\uparrow}$ packet is generated from residual critical-pair source rows, finite budget rows, candidate local joins, closed checks, obstruction rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICResidualCriticalPairBudget}^{\uparrow}}$, seed)

Scope closed: Definition 7709.1, Definition 7709.2, and Definition 7709.3 seed the finite residual critical-pair budget route for the MetaCIC confluence audit map.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICResidualCriticalPairBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global confluence, unrestricted residual-substitution compatibility, parallel diamond, subject reduction, quotient syntax equality, Lean verification, or bridge checking for MetaCICResidualCriticalPairBudget$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, residual-source admission, finite-budget admission, candidate-join admission, closed-check admission, obstruction retention, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualCriticalPairBudgetUp.

Chapter 7710

A Concrete Naming Certificate for $\mathsf{BarTheorem}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen, the $\mathsf{BarTheorem}^{\uparrow}$ packet records a constructive stream/bar boundary as finite NameCert data rather than as a selected infinite branch or a countable-choice principle. It sits after the fan-bar boundary of chapter 400, the inductive bar route of chapter 1675, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-seal boundary of chapter 38. The packet gives Real and Completion consumers a displayed finite bar interface before they read a stream boundary as a real endpoint.

Definition 7710.1 (Bar theorem carrier)**.** *A* $\mathsf{BarTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, T, M, U, S, R, D, A, H, C, P, N)$$

over $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *with the following displayed rows:*

1. *F is the finite bar predicate row, naming the decidable prefix checks admitted by the packet.*
2. *T is the finite prefix tree row whose nodes are the stream prefixes inspected before a bar hit is claimed.*
3. *M records monotone coverage: once a displayed prefix is accepted by F, every displayed extension inside T remains covered.*
4. *U is the fan-bound handoff row, exposing the finite bound supplied through the* $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *surfaces.*
5. *S records the* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the prefix tree.*
6. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback associated with the displayed stream windows.*
7. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used when the readback is compared across covered prefixes.*
8. *A is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the bar predicate, tree, coverage, fan-bound handoff, stream windows, regular readback, and dyadic tolerance rows have been replayed.*
9. *H records componentwise* hsame *transport for the bar predicate, prefix tree, monotone coverage, bound, window, readback, tolerance, seal, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a Real or Completion request back through A, D, R, S, U, M, T, F.*
11. *P is the* Pkg *provenance over the listed constructive bar, stream, rational, dyadic, real, transport, continuation, package, and naming dependencies.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BarTheorem}^{\uparrow}}$ *row.*

The carrier exports no selected infinite branch, no countable-choice functional, no quotient stream, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.BarTheoremUp.BarTheoremTasteGate single carrier alignment`

Theorem 7710.2 (Bar theorem NameCert obligations)**.** *For every accepted* $\mathsf{BarTheorem}^{\uparrow}$ *carrier $B = (F, T, M, U, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BarTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite bar predicate, prefix tree, monotone coverage, fan-bound handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise transport, displayed replay, provenance, and local naming rows. A consumer may inspect the bar theorem boundary only through those displayed rows.*

Proof. Project the carrier. The rows $F, T, M, U, S, R, D, A, H, C, P, N$ are the complete finite inventory available to a NameCert consumer.

The constructive bar part is read through F, T, M, U. The predicate row names the accepted prefix checks, the tree row displays the inspected prefixes, the monotone row keeps coverage stable under displayed extension, and the handoff row exposes the finite bound through the fan and bar-induction siblings.

The stream and Real-facing part is read through S, R, D, A. StreamName windows supply the inspected stream prefixes, RegSeqRat gives the regular rational readback, DyadicRatCore records the tolerance ledger, and the Real seal is downstream of those rows.

Finally H, C, P, N transport, replay, cite, and name only the same coordinates. Since the carrier has no selected branch, countable-choice, quotient-stream, or ambient-completeness coordinate, none of those objects is exported by the local NameCert surface. □

Theorem 7710.3 (Bar theorem Real and Completion boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BarTheorem}^{\uparrow}$ *packet must expose the displayed route*

$$F, T, M, U, S, R, D \longmapsto A.$$

Thus a completion-facing read of the bar boundary passes through the finite bar predicate, prefix tree, monotone coverage, fan-bound handoff, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance ledger before it may read the $\mathsf{Real}^{\uparrow}$ *seal. The route is finite NameCert replay, not a selected infinite branch, countable choice, quotient stream, or ambient Real completeness claim.*

Proof. Start with a consumer that asks for the Real seal row A. Carrier admission shows that A is not standalone; it is a coordinate of the same packet as F, T, M, U, S, R, D.

Replay the bar route first. The prefix request is checked against T, coverage is certified through F and M, and the finite handoff bound is supplied by U through the fan and bar-induction provenance rows.

Replay the stream route next. The inspected prefixes are read through S, their regular rational data through R, and their dyadic tolerance comparisons through D.

Transport by H preserves this route componentwise, while C, P, N only replay, cite, and name the displayed packet. Therefore a Real or Completion consumer receives the seal row only after

the finite bar route has been exposed, and no hidden branch, choice functional, quotient stream, or ambient completeness theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarTheorem}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarTheorem}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7710.1, Definition 7710.2, and Definition 7710.3 seed the finite constructive bar boundary consumed by Real and Completion chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BarTheorem}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, countable choice, quotient streams, ambient Real completeness, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone coverage, fan-bound handoff, Real-seal boundary replay, and refusal of hidden branch or choice data for the same packet.

Constructive story: A $\mathsf{BarTheorem}^{\uparrow}$ packet is generated from a finite bar predicate, prefix tree, monotone coverage row, fan-bound handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 7711

A Concrete Naming Certificate for $\mathsf{CauchyCompletionNaturality}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletionNaturality}^{\uparrow}$ packet records the naturality surface for completion when a displayed uniformly continuous map carries regular rational Cauchy names to regular rational Cauchy names. It sits after the finite stream interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the functorial completion packet of chapter 423, and the uniform-continuity carrier of chapter 391. The packet names only the displayed comparison rows needed by later Real and Completion consumers; it does not select representatives, quotient Cauchy names, or assert ambient completeness.

Definition 7711.1 (Cauchy-completion naturality carrier)**.** *A* $\mathsf{CauchyCompletionNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source completion row, exposing the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used to name the source regular-Cauchy family.*
2. *T is the target completion row, exposing the target* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
3. *F is the* $\mathsf{UniformContinuity}^{\uparrow}$ *map row, including the finite modulus and replay schedule that turns a source tolerance request into a target tolerance request.*
4. *R is the regular-Cauchy transport row. It records that applying F to the source regular-Cauchy windows produces the target regular-Cauchy windows named in T.*
5. *E is the equality-of-route ledger comparing the two displayed completion reads: first complete then map, or first map the regular-Cauchy data and then complete.*
6. *H is componentwise* hsame *transport for S, T, F, R, E, C, P, N.*
7. *C is the displayed* Cont *replay from a completion-naturality request through the source windows, the uniform-continuity modulus, the target readback, and the route ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionNaturality}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionNaturality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a quotient completion, a selected representative of a Cauchy class, a host function between completed metric spaces, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 7711.2 (Cauchy-completion naturality NameCert obligations)**.** *For every accepted* CauchyCompletionNa *packet $K = (S, T, F, R, E, H, C, P, N)$, the local* NameCert *surface consists exactly of carrier admission, source-window admission, target-window admission, uniform-continuity replay, regular-Cauchy transport, route-ledger comparison, componentwise transport, continuation replay, provenance, and local naming. Each obligation is read from one of the displayed rows of Definition 7711.1.*

Proof. Carrier admission is projection of the tuple in Definition 7711.1. The source and target completion rows are the coordinates S and T, and the map row is the coordinate F.

The regular-Cauchy transport obligation is the row R. It ties the finite source windows to the finite target windows by the modulus and replay data in F, so no representative choice is introduced.

The route comparison is the ledger E, transported componentwise by H and replayed by C. Provenance and local naming are the rows P and N. Since the carrier has no other coordinate, these rows exhaust the local NameCert obligations. □

Theorem 7711.3 (Cauchy-completion naturality handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyCompletionNaturality}^{\uparrow}$ *packet reads the naturality square through*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cor

The consumer first reads the source regular-Cauchy windows, then the uniform-continuity modulus, then the target regular-Cauchy readback and $\mathsf{Real}^{\uparrow}$ *seal, and only then the route ledger comparing the two completion reads.*

Proof. Start with a completion-facing consumer and project the packet $K = (S, T, F, R, E, H, C, P, N)$. The source route is stored in S, and the target route is stored in T.

The map can be used only through F. Its modulus and replay schedule feed the transport row R, which displays the target regular-Cauchy windows before the terminal Real seal is consumed.

Finally compare the two displayed reads with E, then attach H, C, P, N for transport, replay, provenance, and local naming. The packet contains no quotient-completion equality, selected limit, host completed function, countable choice, or ambient completeness field, so no such datum is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionNaturality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionNaturality}^{\uparrow}$ packet is generated from source completion windows, target completion windows, a displayed $\mathsf{UniformContinuity}^{\uparrow}$ modulus row, regular-Cauchy transport, a route-comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionNaturality}^{\uparrow}}$, seed)

Scope closed: Definition 7711.1, Definition 7711.2, and Definition 7711.3 seed the finite naturality surface for regular-Cauchy completion under uniformly continuous maps.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionNaturality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected Cauchy representatives, host completed function spaces, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: A formal route can encode BEDC.Derived.CauchyCompletionNaturalityUp and prove carrier admission, regular-Cauchy transport, route-ledger comparison, provenance, and local NameCert exactness.

Chapter 7712

A Concrete Naming Certificate for $\mathsf{FrontierCompileHardProblem}^{\uparrow}$

The $\mathsf{FrontierCompileHardProblem}^{\uparrow}$ packet realizes the frontier-synthesis route of Definition 9414.75 as finite NameCert data. It binds the hard-problem row, layered-relation row, and cook-compile route into one packet before any consumer treats an unresolved frontier as a compiled theorem obligation. The packet sits beside the MetaCIC decidability witness of chapter 751, the MetaCIC critical-path packet of chapter 765, the computability carrier of chapter 199, and the lambda-calculus source grammar of chapter 200. These siblings supply decidability pressure, critical-path accounting, computable presentation, and term syntax; this chapter records the finite frontier package that keeps them jointly inspectable.

Falsifiable prediction: An accepted $\mathsf{FrontierCompileHardProblem}^{\uparrow}$ consumer that exports a compiled hard-problem result without displaying the hard-problem row, layered-relation row, cook-compile route, obstruction ledger, transport, replay, provenance, and local $\mathsf{NameCert}$ rows refutes the carrier surface.

Independence witness: metacicdecidabilitywitness, metaciccriticalpath, computable, lambdacalc, frontiercompilehardproblem: none is bijective with $\mathsf{FrontierCompileHardProblem}^{\uparrow}$ because the packet jointly carries frontier source, hard-problem classifier, layered relation, cook-compile route, obstruction ledger, transport, replay, provenance, and local NameCert rows in one carrier, while each sibling supplies only one constituent surface.

Definition 7712.1 (FrontierCompileHardProblem carrier)**.** *A* $\mathsf{FrontierCompileHardProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, H, L, K, O, T, C, P, N)$$

with the following displayed rows:

1. *S is the frontier-source row, naming the visible problem family and the finite syntax or computation window from which the frontier is read.*
2. *H is the hard-problem classifier row. It records the exact blocked problem surface and the local reason it is not discharged by the current compiled route.*
3. *L is the layered-relation row, separating source relation, compiled relation, witness relation, and refusal relation before any cook-style read is attempted.*
4. *K is the cook-compile route row. It records the displayed reduction schedule from the source frontier to the compiled check surface.*

5. *O is the obstruction ledger. It records which decision, normalization, subject-reduction, confluence, or computability row is still exposed as a boundary rather than as a solved theorem.*
6. *T is componentwise* hsame *transport for S, H, L, K, O, C, P, N.*
7. *C is the displayed* Cont *replay from a frontier request through the layered relation, cook-compile route, and obstruction ledger.*
8. *P is the* Pkg *provenance over* FrontierCompileHardProblem$^{\uparrow}$, MetaCICDecidabilityWitness$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, Computable$^{\uparrow}$, LambdaCalc$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{FrontierCompileHardProblem}^{\uparrow}}$ *row.*

The classifier has no coordinate for a solved hard problem, an oracle decision procedure, global normalization, global confluence, subject-reduction discharge, or an untracked compiler theorem.

Theorem 7712.2 (FrontierCompileHardProblem NameCert obligations)**.** *For every accepted* FrontierCompileHardP *packet $F = (S, H, L, K, O, T, C, P, N)$, the local* NameCert *surface consists exactly of carrier admission, hard-problem classifier exposure, layered-relation separation, cook-compile route replay, obstruction-ledger non-escape, componentwise transport, continuation replay, provenance, and local naming. A consumer that changes any exported frontier read must change one of these displayed rows.*

Proof. Carrier admission is projection of Definition 7712.1. The frontier source is S, the hard-problem classifier is H, the layered-relation surface is L, and the compile route is K.

The obstruction ledger is the row O. It records the remaining boundary as a finite row, so the packet does not silently turn an exposed problem into a solved theorem.

Transport and replay are exactly T and C, while provenance and local naming are P and N. Since the carrier contains no oracle, global normalization, global confluence, subject-reduction discharge, or hidden compiler theorem, these rows exhaust the obligation surface. □

Theorem 7712.3 (FrontierCompileHardProblem consumer boundary)**.** *Every hard-problem, layered-relation, or cook-compile consumer of an accepted* FrontierCompileHardProblem$^{\uparrow}$ *packet factors through the displayed rows*

$$S, \quad H, \quad L, \quad K, \quad O, \quad T, \quad C, \quad P, \quad N.$$

The consumer may inspect the source frontier, the hard-problem classifier, the layered relation, and the compile route, but it cannot export a solved theorem unless the obstruction ledger itself displays the required discharge.

Proof. Begin with a frontier consumer and project the packet. The only source data are in S, and the only hard-problem classifier is H.

The consumer then reads L and K. The layered-relation row separates the source, compiled, witness, and refusal relations; the cook-compile row records only the displayed reduction schedule.

Finally the consumer reads O, T, C, P, N. If the obstruction ledger does not contain a discharge row, the carrier has no other place from which a solved theorem, oracle decision procedure, global normalization theorem, confluence theorem, subject-reduction theorem, or hidden compiler theorem could be recovered. □

Closure status for NameCert$_{\text{FrontierCompileHardProblem}^{\uparrow}}$**.**

Constructive story: A FrontierCompileHardProblem$^{\uparrow}$ packet is generated from a frontier-source row, hard-problem classifier row, layered-relation row, cook-compile route row, obstruction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FrontierCompileHardProblem}\uparrow}$, seed)

Scope closed: Definition 7712.1, Definition 7712.2, and Definition 7712.3 seed the finite frontier compile surface over hard-problem, layered-relation, and cook-compile rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FrontierCompileHardProblem}\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim a solved hard problem, oracle decidability, global normalization, global confluence, subject-reduction discharge, or an untracked compiler theorem.

Upgrade path: A formal route can encode BEDC.Derived.FrontierCompileHardProblemUp and prove carrier admission, layered-relation separation, cook-compile route replay, obstruction non-escape, provenance, and local NameCert exactness.

Chapter 7713

A Concrete Naming Certificate for $\mathsf{TuttePolynomial}^{\uparrow}$

Following Tutte's deletion-contraction treatment of finite graph invariants, the $\mathsf{TuttePolynomial}^{\uparrow}$ packet records the polynomial ledger that a graph or matroid consumer may inspect before using the two-variable Tutte polynomial. It sits over the finite graph carrier of chapter 118, the rank and independence surface of chapter 202, the coefficient-list discipline of chapter 50, and the edge-ledger comparison route of chapter 233. The packet names only finite deletion, contraction, base-case, coefficient, classifier, transport, replay, provenance, and local naming rows; it does not assert graph-minor structure, universal invariant uniqueness, or an ambient host polynomial quotient.

Definition 7713.1 (Tutte polynomial carrier)**.** *A* $\mathsf{TuttePolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, M, E, D, C, B, L, K, H, R, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Graph}^{\uparrow}$ *row, including vertex, edge, incidence, loop, bridge, and endpoint readback data.*
2. *M is the optional* $\mathsf{Matroid}^{\uparrow}$ *rank-dependence row used when the deletion-contraction read is routed through a displayed matroid carrier.*
3. *E is the inspected edge row naming the finite edge whose deletion or contraction branch is being read.*
4. *D is the deletion row, carrying the graph or matroid packet obtained by removing E from the same displayed support.*
5. *C is the contraction row, carrying the graph or matroid packet obtained by contracting E with its endpoint or rank bookkeeping displayed.*
6. *B is the base-case row for empty support, loop, bridge, and isolated component cases.*
7. *L is the finite coefficient ledger in the* $\mathsf{Poly}^{\uparrow}$ *surface, with the two variables and all inspected coefficient cells named as rows.*
8. *K is the classifier-stability row requiring deletion, contraction, base-case, and coefficient reads to remain stable under the displayed* hsame *witnesses.*
9. *H stores componentwise* hsame *transports for $G, M, E, D, C, B, L, K, R, P, N$.*

10. *R stores displayed* Cont *replay through graph or matroid inspection, deletion, contraction, base cases, coefficient extraction, classifier stability, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{TuttePolynomial}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TuttePolynomial}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for a host graph-minor theorem, a universal invariant proof, quotient equality of polynomials, arbitrary edge-set selection, or an ambient deletion-contraction oracle.

Theorem 7713.2 (Tutte polynomial deletion-contraction obligations)**.** *For every accepted* $\mathsf{TuttePolynomial}^{\uparrow}$ *carrier $T = (G, M, E, D, C, B, L, K, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TuttePolynomial}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite graph row, optional matroid row, inspected edge row, deletion row, contraction row, base-case row, coefficient ledger, classifier-stability row, componentwise transport, displayed replay, package provenance, and local naming. Every polynomial-facing consumer must expose the deletion branch, contraction branch, base cases, coefficient ledger, and classifier-stability row before it reads the Tutte-polynomial packet as a finite invariant.*

Proof. Project the carrier of Definition 7713.1. The source rows are G, M, E, the recursive polynomial rows are D, C, B, L, and the structural rows are K, H, R, P, N.

Read the deletion and contraction branches first. The inspected edge row E determines which finite support is removed or contracted, while D and C keep both resulting packets inside the displayed $\mathsf{Graph}^{\uparrow}$ or $\mathsf{Matroid}^{\uparrow}$ dependency surface.

Next read the base and coefficient rows. The row B records the empty-support, loop, bridge, and isolated-component cases, and L records the two-variable coefficient ledger through $\mathsf{Poly}^{\uparrow}$ rather than through host polynomial equality.

Finally apply the classifier and structural rows. The stability row K makes the deletion, contraction, base-case, and coefficient data invariant under the displayed hsame transports, while H, R, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores a graph-minor theorem, a universal invariant proof, or an ambient polynomial quotient, the listed rows exhaust the local certificate obligations. □

Falsifiable prediction: Any accepted $\mathsf{TuttePolynomial}^{\uparrow}$ packet is refuted if a consumer reads a polynomial coefficient before displaying the finite graph or matroid source row, inspected edge row, deletion row, contraction row, base-case row, coefficient ledger, classifier-stability row, transport, replay, provenance, and local NameCert rows.

Independence witness: graph, matroid, polynomial, networkflow: $\mathsf{TuttePolynomial}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries deletion, contraction, base-case, coefficient-ledger, classifier-stability, transport, replay, provenance, and local NameCert rows. The siblings supply source graph incidence, rank-dependence, coefficient storage, or edge-ledger routing separately, while only this packet binds them as the finite deletion-contraction polynomial surface.

Closure status for $\mathsf{NameCert}_{\mathsf{TuttePolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TuttePolynomial}^{\uparrow}$ packet is generated from finite GraphUp rows, optional MatroidUp rank-dependence rows, inspected edge rows, deletion rows, contraction rows, base-case rows, PolynomialUp coefficient ledgers, classifier-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TuttePolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7713.1 and Definition 7713.2 seed the finite deletion-contraction polynomial route over GraphUp, MatroidUp, PolynomialUp, NetworkFlowUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{TuttePolynomial}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, full graph-minor theory, universal Tutte invariant uniqueness, quotient equality of polynomials, arbitrary edge-set selection, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deletion and contraction branch exposure, base-case exactness, coefficient-ledger exactness, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.TuttePolynomialUp.

Chapter 7714

A Concrete Naming Certificate for MetaCICSubstitutionDiamondFrontier$^{\uparrow}$

The MetaCICSubstitutionDiamondFrontier$^{\uparrow}$ packet realizes the substitution-sensitive boundary of the MetaCIC beta audit map of chapter 9384 as finite NameCert data. It sits beside the residual checker of chapter 3827, the residual-diamond budget of chapter 7407, the beta-step boundary of chapter 754, and the subject-reduction packet of chapter 483. The packet records a beta peak, residual rows, candidate joins, substitution-boundary rows, checker rows, obstruction rows, transport, replay, provenance, and local naming; it does not claim full confluence, subject reduction, unrestricted substitution compatibility, quotient syntax equality, or a hidden evaluator.

Definition 7714.1 (MetaCIC substitution-diamond frontier carrier)**.** *A* MetaCICSubstitutionDiamondFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, P, R, J, S, K, O, H, C, Q, N)$$

with the following displayed rows:

1. *T is the MetaCIC term and context row naming the finite source of the beta peak.*
2. *P is the beta-peak row, read through the displayed* BetaStep *and* BetaConv *boundary of chapter 9384.*
3. *R is the residual row naming the branch residuals that remain after each beta step.*
4. *J is the candidate-join row listing the finite terms proposed as local joins for the displayed peak.*
5. *S is the substitution-boundary row, recording which substitution, shift, and closedness data must stay visible for the candidate join to be inspected.*
6. *K is the bounded checker row, read through the residual-checker and residual-diamond-budget sibling packets.*
7. *O is the obstruction row, keeping subject reduction, unrestricted substitution compatibility, and global confluence outside the carrier.*
8. *H stores componentwise* hsame *transport for $T, P, R, J, S, K, O, C, Q, N$.*
9. *C stores displayed* Cont *replay through the beta peak, residuals, candidate joins, substitution boundary, bounded checker, obstruction, provenance, and local naming.*

10. *Q is the* Pkg *provenance over* $\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$, $\mathsf{BetaStepBoundary}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for global Church–Rosser, closed subject reduction, unrestricted substitution compatibility, quotient syntax equality, or an external evaluator.

Theorem 7714.2 (MetaCIC substitution-diamond frontier obligations)**.** *For every accepted* MetaCICSubstitutionDi *carrier $F = (T, P, R, J, S, K, O, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}}$ *surface consists exactly of source-term admission, beta-peak admission, residual exposure, candidate-join exposure, substitution-boundary exposure, bounded-checker admission, obstruction retention, componentwise transport, displayed replay, package provenance, and local naming. Every displayed candidate join must retain the named substitution-boundary row S before the checker row K may inspect it.*

Proof. Project the carrier of Definition 7714.1. The beta-facing rows are T, P, R, J, the substitution and checker rows are S, K, O, and the structural rows are H, C, Q, N.

Read the peak and residual rows first. The term row T and beta-peak row P name the finite local peak, while R records the residual data that remain visible on each branch.

Next inspect the candidate joins through J, S, K. The candidate-join row J is not accepted alone; it is paired with the substitution-boundary row S, and the bounded checker K reads the pair only while the substitution, shift, and closedness data remain displayed.

Finally retain the obstruction and structural rows. The row O prevents the local frontier from becoming a proof of full confluence or subject reduction, while H, C, Q, N transport, replay, source, and name the same finite packet. Since no coordinate stores global Church–Rosser, unrestricted substitution compatibility, quotient syntax equality, or an external evaluator, the listed rows exhaust the local obligation surface. □

Falsifiable prediction: If the frontier is accepted, every displayed candidate join must retain a named substitution-boundary row; deleting that row invalidates the local diamond packet.

Independence witness: metacic-confluence-residual-checker, metacic-residual-diamond-budget, betastepboundary, subjectreduction: $\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}$ is not bijective to these siblings because it jointly carries beta peaks, residual rows, candidate joins, substitution-boundary rows, bounded checker rows, obstruction retention, transport, replay, provenance, and local naming. The siblings supply checker, budget, beta-step, or preservation surfaces separately, while only this packet binds substitution visibility to each candidate local join.

The Lean follow-up must provide explicit Nontrivial $\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}$ and FieldFaithful $\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}$ instances in `BEDC/Derived/MetaCICSubstitutionDiamondFron` preserving the distinct rows $T, P, R, J, S, K, O, H, C, Q, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}$ packet is generated from MetaCIC term and context rows, beta-peak rows, residual rows, candidate-join rows, substitution-boundary rows, bounded checker rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 7714.1 and Definition 7714.2 seed the finite substitution-sensitive beta diamond frontier for the MetaCIC beta audit map.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionDiamondFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full confluence, subject reduction, unrestricted substitution compatibility, quotient syntax equality, external evaluator soundness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-term admission, beta-peak admission, residual exposure, candidate-join exposure, substitution-boundary exposure, bounded-checker admission, obstruction retention, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSubstitutionDiamondFrontierUp

Chapter 7715

A Concrete Naming Certificate for $\mathsf{BareissFractionFreeElimination}^{\uparrow}$

Following Bareiss and standard fraction-free linear algebra, the $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ packet records determinant-preserving elimination as a finite pivot ledger over displayed matrix, ring, polynomial, and determinant rows. It consumes the matrix carrier of chapter 49, the determinant surface of chapter 84, the commutative ring source of chapter 43, and the coefficient-polynomial route of chapter 50. The packet names only finite pivot rows, fraction-free update rows, exact-divisibility witnesses, determinant handoff rows, transport, replay, provenance, and local naming; it exports no host field division, quotient matrix equality, row-equivalence theorem, or ambient Gaussian elimination algorithm.

Definition 7715.1 (Bareiss fraction-free elimination carrier)**.** *A* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, M, P, D, I, L, U, Q, X, H, C, K, N)$$

with the following displayed rows:

1. *R is the consumed* $\mathsf{Ring}^{\uparrow}$ *scalar row, naming the addition, multiplication, zero, one, and sign data used by the ledger.*
2. *M is the consumed* $\mathsf{Mat}^{\uparrow}$ *row, including the finite row-column index set and the initial entry readbacks.*
3. *P is the optional* $\mathsf{Poly}^{\uparrow}$ *coefficient row when the matrix entries are read as displayed polynomial coefficients rather than as opaque ring elements.*
4. *D is the* $\mathsf{Determinant}^{\uparrow}$ *handoff row for the input matrix and the terminal diagonal product read.*
5. *I lists the pivot indices selected by the ledger and the previous-pivot denominator row attached to each step.*
6. *L records the fraction-free update equation for every displayed entry:*

$$a_{ij}^{(k+1)} p_{k-1} = a_{kk}^{(k)} a_{ij}^{(k)} - a_{ik}^{(k)} a_{kj}^{(k)}.$$

7. *U records the exact-divisibility witness that the right-hand side is divisible by the previous pivot row inside the displayed ring surface.*

8. *Q records the zero-pivot branch ledger: if a displayed pivot is absent, the packet stores the row or column swap evidence that precedes the next update row, rather than selecting an ambient inverse.*
9. *X is the determinant-scale exactness row tying the terminal pivot product and sign ledger to the consumed* $\mathsf{Determinant}^{\uparrow}$ *route.*
10. *H records componentwise* hsame *transport for the ring, matrix, polynomial, determinant, pivot, update, divisibility, branch, exactness, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a linear-algebra consumer back through $X, Q, U, L, I, D, P, M, R$.*
12. *K is the* Pkg *provenance over* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BareissFractionFreeElimination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host field division, quotient matrix equality, a global row-equivalence theorem, an unlogged pivot inverse, or a selected elimination trace outside the finite ledger.

Lean4 ✓ `BEDC.Derived.BareissFractionFreeEliminationUp.TasteGate.BareissFractionFreeEliminationTasteGate`
`single carrier alignment`

Theorem 7715.2 (Bareiss fraction-free elimination NameCert obligations)**.** *For every accepted* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ *carrier $E = (R, M, P, D, I, L, U, Q, X, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{BareissFractionFre}}$ *surface consists exactly of carrier admission, ring and matrix exposure, polynomial-coefficient exposure when present, determinant-source exposure, pivot-index exposure, fraction-free update exposure, exact-divisibility exposure, zero-pivot branch exposure, determinant exactness handoff, componentwise transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7715.1. The rows $R, M, P, D, I, L, U, Q, X, H, C, K, N$ are the complete finite inventory available to a NameCert consumer.

The algebraic source is read through R, M, P, D. The update path is then read through I, L, U, Q: pivots and previous denominators are displayed, each update equation is logged, exact divisibility is witnessed inside the ring row, and a zero-pivot event is routed through displayed swap evidence.

The consumer-facing surface is X, H, C, K, N. These rows hand the terminal determinant read to $\mathsf{Determinant}^{\uparrow}$, transport the displayed ledger componentwise, replay the same route for downstream consumers, and record provenance and local naming. Since no row contains a host inverse, quotient matrix equality, or an unlogged row-equivalence theorem, none of those objects is exported. □

Theorem 7715.3 (Bareiss pivot ledger)**.** *Every determinant, rank-sensitive, or finite linear-system consumer of an accepted* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ *packet must first read the finite pivot ledger*

$$R, M, P, D, I, L, U, Q, X, H, C, K, N.$$

In particular, each update step is available only after the displayed pivot row, previous-pivot denominator row, fraction-free numerator row, exact-divisibility witness, and zero-pivot branch row have been exposed.

Proof. Let a downstream consumer route be given. Carrier projection exposes the ring, matrix, polynomial, and determinant source rows before any pivot update can be replayed.

For each pivot step, the consumer reads I for the chosen pivot and previous denominator, then L for the displayed numerator equation, then U for exact divisibility. If the pivot branch is not direct, Q supplies the finite swap evidence before the next update is read.

The route then passes through X, H, C, K, N. These rows preserve the same finite ledger and do not add a field of fractions, quotient matrix equality, a selected row-reduction theorem, or an ambient elimination algorithm. □

Theorem 7715.4 (Bareiss determinant handoff)**.** *The determinant read exported by an accepted* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Ring}^{\uparrow},\ \mathsf{Mat}^{\uparrow},\ \mathsf{Poly}^{\uparrow},\ \mathsf{Determinant}^{\uparrow} \longmapsto I, L, U, Q, X \longmapsto H, C, K, N.$$

Thus the terminal determinant handoff is a finite fraction-free ledger over the consumed sibling rows, not a host Gaussian elimination theorem, field-division computation, or quotient matrix equality principle.

Proof. Start with the determinant-facing row X. It is not standalone: by carrier admission it depends on the consumed ring, matrix, optional polynomial, determinant, pivot, update, divisibility, and branch rows.

Replay the fraction-free ledger. The update equation in L is meaningful only after the pivot data in I is displayed, and its division step is allowed only through the exact-divisibility witness U inside the ring row. The branch row Q records any finite swap before the next pivot ledger is read.

Finally use H, C, K, N to transport, replay, cite, and name the same data. Therefore the determinant handoff reaches $\mathsf{Determinant}^{\uparrow}$ through displayed finite rows, and no host field division, quotient matrix equality, or ambient Gaussian elimination theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{BareissFractionFreeElimination}^{\uparrow}}$.

Constructive story: A $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ packet is generated from RingUp scalar rows, MatrixUp entry rows, optional PolynomialUp coefficient rows, DeterminantUp source rows, finite pivot indices, fraction-free update equations, exact-divisibility witnesses, zero-pivot branch rows, determinant exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BareissFractionFreeElimination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7715.1, Definition 7715.2, Definition 7715.3, and Definition 7715.4 seed the finite fraction-free elimination route over displayed matrix, determinant, ring, polynomial, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BareissFractionFreeElimination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host field division, quotient matrix equality, arbitrary Gaussian elimination, rank theory, Smith normal form, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pivot-ledger exactness, exact-divisibility replay, determinant handoff, transport stability, and consumer non-escape for BEDC.Derived.BareissFractionFreeEliminationUp.

Chapter 7716

A Concrete Naming Certificate for $\mathsf{ProofMiningMetastability}^{\uparrow}$

Following Kohlenbach proof mining and Tao metastability, the $\mathsf{ProofMiningMetastability}^{\uparrow}$ packet records the finite-bound extraction route by which a convergence or compactness argument is read as explicit metastable data. It sits beside the finite tail-window packet of chapter 2093, the Cauchy carrier of chapter 3413, the regular rational readback of chapter 301, the finite observation windows of chapter 126, the compact finite-subcover boundary of chapter 60, and the terminal real-name seal of chapter 38. The packet packages a human-standard proof-mining mechanism as finite NameCert data: proofs may export bounds, control windows, and compactness handoffs, but they do not import selected limits, countable choice, host real equality, or ambient completeness.

Definition 7716.1 (Proof-mining metastability carrier)**.** *A* $\mathsf{ProofMiningMetastability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (F, B, W, D, S, Q, C, K, R, H, T, L, N)$$

with the following displayed rows:

1. *F is the formal proof input row, naming the proof object or proof skeleton whose quantitative content is inspected.*
2. *B is the extracted bound row, recording the finite counterfunction or majorant data read from F.*
3. *W is the metastable window row, displaying the finite interval or finite cover segment controlled by B.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used when the window is Cauchy-facing or real-facing.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row for inspected stream positions.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tail row for rational readback inside the window.*
7. *C is the* $\mathsf{Cauchy}^{\uparrow}$ *consumer row, recording when the mined data is read as a Cauchy-facing tail statement.*
8. *K is the* $\mathsf{Compact}^{\uparrow}$ *consumer row, recording when the mined data is read as a compact finite-subcover or finite-choice boundary.*
9. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after B, W, D, S, Q, C, K have been displayed.*

10. *H records componentwise* hsame *transport for proof input, bound, window, tolerance, stream, regular-rational, Cauchy, compact, real-seal, replay, provenance, and naming rows.*
11. *T records displayed* Cont *replay from proof input to extracted finite data and then to the Cauchy, compact, and Real-facing consumers.*
12. *L is the* Pkg *provenance over* $\mathsf{ProofMiningMetastability}^{\uparrow}$, $\mathsf{CauchyMetastability}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofMiningMetastability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, B, W, D, S, Q, C, K, R, H, T, L, N$. It has no coordinate for selected limits, countable choice, quotient equality, host real equality, excluded-middle search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7716.2 (Proof-mining metastability NameCert obligations)**.** *For every accepted* ProofMiningMetastability *carrier $P = (F, B, W, D, S, Q, C, K, R, H, T, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProofMiningMetastability}^{\uparrow}}$ *surface consists exactly of the proof input row F, extracted bound row B, finite metastable window W, dyadic tolerance ledger D, StreamName observations S, RegSeqRat readback Q, Cauchy consumer row C, Compact consumer row K, Real seal row R, componentwise transport H, displayed replay T, provenance L, and local naming N. Every public read of a mined convergence or compactness consequence factors through those rows before it can cite a metastable bound.*

Proof. Project the accepted carrier of Definition 7716.1. Its only proof-facing coordinates are F, B, W, so the proof input may be read only through the extracted finite bound and the window it controls.

Read the analytic rows next. The tolerance ledger D, StreamName row S, and RegSeqRat row Q provide the finite comparison data consumed by Cauchy and Real-facing readers, while the Cauchy row C and compact row K record the two public consumer surfaces.

Finally inspect R, H, T, L, N. The real seal is downstream of the extracted finite data, and the structural rows transport, replay, source, and name that same packet. Since no other row is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7716.3 (Proof-mining metastability compact Cauchy handoff)**.** *Every accepted* ProofMiningMetastability *read that is consumed by a* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *route exports only the finite bound B, metastable window W, dyadic tolerance ledger D,* $\mathsf{StreamName}^{\uparrow}$ *observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the displayed consumer rows C, K, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal R. The handoff exports no selected limit, no countable choice object, no host compactness selector, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from Definition 7716.2. A public consumer must read the extracted finite data B, W before it can inspect any analytic endpoint.

For a Cauchy-facing read, use the sibling route through chapter 2093 and chapter 3413. The rows D, S, Q provide tolerance accounting, finite observations, and regular-rational readback before C reaches the $\mathsf{Real}^{\uparrow}$ seal.

For a compact-facing read, use the $\mathsf{Compact}^{\uparrow}$ row K. It receives the same finite window as a finite-subcover or finite-choice boundary, not as a host selector or countable choice principle.

The remaining rows H, T, L, N preserve the same handoff under transport, replay, provenance, and local naming. Therefore the compact and Cauchy consumers receive a finite proof-mined metastability packet and no stronger completion or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofMiningMetastability}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofMiningMetastability}^{\uparrow}$ packet is generated from a proof input row, extracted finite bound, metastable window, dyadic tolerance ledger, StreamName observations, RegSeqRat readback, Cauchy consumer row, Compact consumer row, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofMiningMetastability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7716.1, Definition 7716.2, and Definition 7716.3 seed the finite proof-mining metastability route for Cauchy, Compact, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofMiningMetastability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, countable choice, host compactness selectors, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, extracted-bound exactness, finite-window replay, Cauchy and Compact handoff stability, Real-seal boundary, transport, provenance, and local naming for BEDC.Derived.ProofMiningMetastabilit

Chapter 7717

A Concrete Naming Certificate for $\mathsf{RadonTheorem}^{\uparrow}$

Following Radon and the standard finite convexity theorem, the $\mathsf{RadonTheorem}^{\uparrow}$ packet records the finite certificate surface for a Radon partition of a point family in an affine space. It consumes the convex membership carrier of chapter 208, the affine-dependence and translation surface of chapter 206, the finite face and hull ledger of chapter 209, and the finite simplex inventory of chapter 238. The packet exposes only the displayed finite point family, the affine-dependence witness, the two-block partition, the convex-hull overlap row, transport, replay, provenance, and local naming; it does not export an ambient Euclidean topology, a selected separating hyperplane, Helly closure, or a Caratheodory reduction.

Definition 7717.1 (Radon theorem finite convex-geometry carrier)**.** *A* $\mathsf{RadonTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, F, D, L, U, V, H, Q, P, N)$$

with the following displayed rows:

1. *A is the finite-dimensional* $\mathsf{AffineSpace}^{\uparrow}$ *ambient row that scopes point differences, affine combinations, and the displayed dependence witness.*
2. *F is the finite point-family row, listing the inspected points as* BHist *entries and recording their finite index surface.*
3. *D is the affine-dependence witness row, with coefficients read through the scalar and affine rows and with total coefficient sum classified as zero.*
4. *L is the two-block partition row, splitting the finite index surface into the positive and negative coefficient blocks used by the Radon construction.*
5. *U is the* $\mathsf{ConvexSet}^{\uparrow}$ *row reading the positive-block normalized affine combination as a convex-hull point.*
6. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row reading the negative-block normalized affine combination as the same convex-hull point after affine-dependence transport.*
7. *H records componentwise* hsame *transport for $A, F, D, L, U, V, Q, P, N$.*
8. *Q records displayed* Cont *replay of the affine-dependence calculation and of the two convex-hull membership reads.*

9. *P is the* Pkg *provenance over* RadonTheorem$^{\uparrow}$, AffineSpace$^{\uparrow}$, ConvexSet$^{\uparrow}$, Polytope$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{RadonTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host convex hull, a topological compactness assertion, a separating hyperplane, or a full Helly-family theorem.

Lean4 ✓ `BEDC.Derived.RadonTheoremUp.RadonTheoremTasteGate single carrier alignment`

Theorem 7717.2 (Radon theorem NameCert obligations)**.** *For the carrier of Definition 7717.1, the local* NameCert$_{\text{RadonTheorem}^{\uparrow}}$ *surface consists exactly of the finite point family row F, the affine-dependence witness row D, the two-block partition row L, the positive and negative convex-hull overlap rows U, V, componentwise* hsame *transport through H, displayed* Cont *replay through Q,* Pkg *provenance through P, and local naming through N.*

Proof. Project the packet $R = (A, F, D, L, U, V, H, Q, P, N)$. The finite point family F gives the only inspected points, while the affine row A is the only ambient structure available for dependence and affine-combination reads.

Use the dependence witness D to split the finite index surface into the two blocks recorded by L. The signs of the displayed coefficients determine the blocks, and the zero-sum affine-dependence row is the calculation that identifies the two normalized affine combinations.

Read the positive block through U and the negative block through V. Both rows are ConvexSet$^{\uparrow}$ reads, while the finite face and simplex bookkeeping is supplied only through the Polytope$^{\uparrow}$ and SimplicialComplex$^{\uparrow}$ provenance rows carried by P. Thus the overlap point is exposed as two convex-hull membership rows classified by the displayed affine-dependence calculation, not as a host geometric object.

Finally transport and replay preserve the same surface. The H row transports every displayed component by hsame, the Q row replays the affine-dependence calculation and convex-hull membership reads by Cont, and P, N bind those rows to the local name. No additional coordinate remains from which to read Helly, Caratheodory, topology, or separation data, so the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RadonTheoremUp.RadonTheoremNameCertObligations`

Closure status for NameCert$_{\text{RadonTheorem}^{\uparrow}}$**.**

Constructive story: A RadonTheorem$^{\uparrow}$ packet is generated from a finite affine-space point family, an affine-dependence witness, a two-block partition of the finite index surface, positive and negative convex-hull membership rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RadonTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 7717.1 and Definition 7717.2 seed the finite Radon partition NameCert surface over AffineSpace$^{\uparrow}$, ConvexSet$^{\uparrow}$, Polytope$^{\uparrow}$, and SimplicialComplex$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RadonTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not assert a Lean theorem, Helly closure, Caratheodory reduction, topological compactness, separating hyperplane, or bridge checking.

Upgrade path: Obligation closure requires checked carrier and NameCert rows for the finite point family, affine-dependence witness, two-block partition, convex-hull overlap, transport, replay, and provenance surface.

Chapter 7718

A Concrete Naming Certificate for $\mathsf{FiniteBorelCantelli}^{\uparrow}$

Following Borel and constructive probability treatments, the $\mathsf{FiniteBorelCantelli}^{\uparrow}$ packet records the finite-union probability boundary that a bounded Borel–Cantelli window may consume before any countable almost-sure conclusion is visible. It sits after the probability-space surface of chapter 184, the measure event-value surface of chapter 92, the finite union-bound ledger of chapter 6696, and the first-lemma boundary of chapter 2357. The packet names only finite event rows, finite union ledgers, and displayed probability bounds.

Definition 7718.1 (Finite Borel–Cantelli carrier)**.** *A* $\mathsf{FiniteBorelCantelli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (\Omega, M, E, W, U, B, T, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the normalized probability carrier.*
2. M *is the* $\mathsf{Measure}^{\uparrow}$ *event-value row through which each displayed probability read is exposed.*
3. E *is the finite event-row packet, listing the inspected event names, membership readbacks, and measurable-event witnesses.*
4. W *is the finite Borel–Cantelli window row, recording the bounded index set and the repeated-hit request restricted to that window.*
5. U *is the* $\mathsf{FiniteUnionBound}^{\uparrow}$ *ledger row, exposing the finite union event and the corresponding probability-sum ledger.*
6. B *is the probability-bound row, recording that the finite repeated-hit event is bounded by the finite sum named by* U*.*
7. T *is the local tail-budget row for the same bounded window; it is a finite comparison row, not a countable summability assertion.*
8. H *records componentwise* hsame *transport for* $\Omega, M, E, W, U, B, T, P, N$*.*
9. C *records displayed* Cont *replay from* Ω *through* M, E, W, U, B, T *before any consumer sees the finite Borel–Cantelli conclusion.*
10. P *is the* Pkg *provenance over* $\mathsf{FiniteBorelCantelli}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{FiniteUnionBound}^{\uparrow}$*,* $\mathsf{BorelCantelli}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{FiniteBorelCantelli}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $\Omega, M, E, W, U, B, T, H, C, P, N$. It contains no coordinate for countable choice, host sigma-algebra completion, quotient almost-sure equality, a selected infinite tail, or the second Borel–Cantelli converse.

Lean4 ✓ `BEDC.Derived.FiniteBorelCantelliUp.FiniteBorelCantelliTasteGate single carrier alignment`

Theorem 7718.2 (Finite Borel–Cantelli NameCert obligations)**.** *For every accepted* $\mathsf{FiniteBorelCantelli}^{\uparrow}$ *carrier $F = (\Omega, M, E, W, U, B, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteBorelCantelli}^{\uparrow}}$ *surface consists exactly of probability-source admission through Ω, measure event-value exposure through M, finite event-row admission through E, bounded-window exposure through W, finite union-bound ledger exposure through U, probability-bound exactness through B, local tail-budget comparison through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7718.1. The displayed tuple exposes the probability source, measure event-value row, finite event rows, bounded window, union ledger, probability bound, local tail budget, transport, replay, provenance, and local naming.

Read the event rows before the bound. The window W restricts the repeated-hit request to the finite family E, while M supplies the probability values over the same measurable-event surface named by Ω.

The union-bound ledger U and probability-bound row B provide the only estimate used by the packet, and T records only the bounded tail comparison attached to that estimate. The rows H, C, P, N transport, replay, source, and name the same finite data.

Since the carrier has no further coordinate, the listed rows exhaust the NameCert obligations. In particular, no countable-choice tail selector, completed sigma-algebra, almost-sure quotient, infinite summability theorem, or converse independence route is exported. □

Theorem 7718.3 (Finite Borel–Cantelli union-bound route)**.** *Every finite Borel–Cantelli read of an accepted* $\mathsf{FiniteBorelCantelli}^{\uparrow}$ *packet factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \mathsf{FiniteUnionBound}^{\uparrow},\ \mathsf{BorelCantelli}^{\uparrow}.$$

The finite event-row packet E and bounded window W must be exposed before the finite union ledger U and probability-bound row B are read, and the local tail-budget row T can inspect only that finite union-bound estimate.

Proof. Start with the probability and measure rows Ω and M. They provide the source probability carrier and the event-value reads used by the finite event packet E.

Next expose the bounded window W. This fixes the finite repeated-hit request before any bound is available, so the estimate is attached to a displayed finite family rather than to an ambient countable event sequence.

Apply the finite union-bound handoff of Definition 6696.4. The row U supplies the finite union event and probability-sum ledger, while B records the corresponding probability comparison for the window selected by W.

The tail row T can therefore read only the finite estimate just produced by U and B. Componentwise transport, replay, provenance, and local naming through H, C, P, N preserve that same route, so no countable choice, sigma-completion, almost-sure quotient, selected infinite tail, or second-lemma converse is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBorelCantelli}\uparrow}$.

Constructive story: A $\mathsf{FiniteBorelCantelli}^{\uparrow}$ packet is generated from a ProbSpace source row, Measure event-value row, finite event rows, bounded window row, FiniteUnionBound ledger, probability-bound row, local tail-budget comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBorelCantelli}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7718.1, Definition 7718.2, and Definition 7718.3 seed the finite-union probability boundary for bounded Borel–Cantelli windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBorelCantelli}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, host sigma-algebra completion, quotient almost-sure equality, selected infinite tails, the second Borel–Cantelli converse, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite event-row exactness, bounded-window routing, finite union-bound ledger exactness, probability-bound exposure, local tail-budget non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteBorelCantelliUp.

Chapter 7719

A Concrete Naming Certificate for $\mathsf{Detectability}^{\uparrow}$

Following Kalman's finite-dimensional control theory, the $\mathsf{Detectability}^{\uparrow}$ packet records the finite boundary between full observation and estimator-consumable decay. It sits after the state-space carrier of chapter 329, the observation surface of chapter 294, the reachability-side control surface of chapter 297, the finite dual readback of chapter 5787, the deterministic observer ledger of chapter 5909, and the prediction-update surface of chapter 5102. The chapter names the finite rows that let observer and filter consumers read unobservable modes only through a displayed decay ledger; it does not import host spectral theory, quotient state spaces, or ambient asymptotic semantics.

Definition 7719.1 (Detectability carrier)**.** *A* $\mathsf{Detectability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, C, U, E, O, L, H, T, P, N)$$

with the following displayed rows:

1. *S is the state-space source row consumed from* $\mathsf{StateSpaceModel}^{\uparrow}$*;*
2. *A is the transition-matrix row over the displayed state coordinates;*
3. *C is the output-map row consumed through* $\mathsf{ControlObservability}^{\uparrow}$*;*
4. *U is the unobservable-mode witness row naming the finite part of the state row not separated by the displayed observation surface;*
5. *E is the decay ledger for the A-evolution on the displayed unobservable-mode witness;*
6. *O is the observer/filter handoff row consumed by* $\mathsf{LuenbergerObserver}^{\uparrow}$ *and* $\mathsf{KalmanFilter}^{\uparrow}$*;*
7. *L is the componentwise* hsame *ledger for the state-space source, transition matrix, output map, unobservable-mode witness, decay ledger, and observer/filter handoff rows;*
8. *H is the displayed* Cont *route for observation readback, decay-ledger readback, and handoff replay;*
9. *T is the* Pkg *provenance over* $\mathsf{Detectability}^{\uparrow}$*,* $\mathsf{StateSpaceModel}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{LuenbergerObserver}^{\uparrow}$*,* $\mathsf{KalmanFilter}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. *P records the local package replay for the same rows after representative change;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}$ *row.*

The classifier identifies two accepted packets only when the displayed state-space, transition, output, unobservable-mode, decay, and handoff rows match componentwise through L, with the same Cont *and* Pkg *provenance.*

Theorem 7719.2 (Detectability NameCert obligations)**.** *For every accepted* $\mathsf{Detectability}^{\uparrow}$ *carrier $D = (S, A, C, U, E, O, L, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}$ *surface has exactly five finite rows:*

1. *carrier admission for the displayed state-space source, transition matrix, output map, unobservable-mode witness, decay ledger, and observer/filter handoff rows;*
2. *classifier admission through componentwise* hsame *transport over those displayed rows;*
3. *observability and Gramian readback saying that C, U, E are consumed only through* $\mathsf{ControlObservability}^{\uparrow}$ *and* $\mathsf{ObservabilityGramian}^{\uparrow}$ *rows;*
4. *observer/filter boundary replay saying that O is the sole route from detectability data to* $\mathsf{LuenbergerObserver}^{\uparrow}$ *or* $\mathsf{KalmanFilter}^{\uparrow}$ *consumers;*
5. *provenance and local naming through the displayed* Cont*,* Pkg*, replay, and* $\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}$ *rows.*

Proof. Project the carrier of Definition 7719.1. The rows S, A, C, U, E, O are the only control-facing data, while L, H, T, P, N transport, replay, source, and name the same finite packet.

The classifier row is componentwise because the carrier has no coordinate for a quotient state space or hidden spectral decomposition. Any comparison of unobservable modes must therefore pass through the displayed hsame ledger L.

The observation and handoff rows are already named as dependencies. The rows C, U, E read through $\mathsf{ControlObservability}^{\uparrow}$ and $\mathsf{ObservabilityGramian}^{\uparrow}$, and the row O is the only observer/filter handoff. Repacking these projections with H, T, P, N gives the stated five-row surface. □

Theorem 7719.3 (Detectability observer/filter boundary)**.** *Every* $\mathsf{LuenbergerObserver}^{\uparrow}$ *or* $\mathsf{KalmanFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{Detectability}^{\uparrow}$ *packet may read detectability only through the displayed observability rows C, U and the decay ledger E, transported through L and replayed through H, T, P, N. The handoff supplies no host spectral radius, no quotient unobservable subspace, no selected observer gain, no stochastic convergence theorem, and no ambient asymptotic stability claim outside the finite packet of Definition 7719.1.*

Proof. Start with the NameCert surface of Definition 7719.2. It exposes the observer/filter handoff row only after the state-space, transition, output, unobservable-mode, and decay rows have been displayed.

The observer consumer reads O as a deterministic boundary for the $\mathsf{LuenbergerObserver}^{\uparrow}$ ledger, while the filter consumer reads the same handoff as finite provenance for the $\mathsf{KalmanFilter}^{\uparrow}$ observation branch. In both cases the rows C, U, E remain attached to $\mathsf{ControlObservability}^{\uparrow}$ and $\mathsf{ObservabilityGramian}^{\uparrow}$ readback rather than to a host spectral object.

Because the carrier contains no row for a spectral radius, quotient unobservable subspace, selected observer gain, stochastic convergence theorem, or ambient asymptotic stability claim, no such object can be exported by the detectability handoff. □

Falsifiable prediction: Any observer or filter consumer that reads $\mathsf{Detectability}^{\uparrow}$ while bypassing the displayed observability rows C, U, the decay ledger E, or the handoff row O violates the local certificate boundary.

Independence witness: statespacemodel, controlobservability, controlcontrollability, observability-gramian, luenberger-observer, kalman-filter: none is bijective to $\mathsf{Detectability}^{\uparrow}$ because the detectability packet jointly carries the state-space source, transition matrix, output map, unobservable-mode witness, decay ledger, observer/filter handoff, transport, replay, provenance, and local NameCert rows while excluding host spectral and quotient semantics.

Closure status for $\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}$.

Constructive story: A $\mathsf{Detectability}^{\uparrow}$ packet is generated from a finite state-space source, transition matrix, output map, unobservable-mode witness, decay ledger, observer/filter handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, package replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7719.1, Definition 7719.2, Definition 7719.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite detectability NameCert surface over state-space, observability, Gramian, observer, and filter rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Detectability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host spectral criteria, quotient unobservable subspaces, selected observer gains, stochastic convergence theorems, or ambient asymptotic stability.

Upgrade path: Obligation closure requires checked carrier admission, observability readback, unobservable-mode witness exactness, decay-ledger exactness, observer/filter handoff stability, transport, replay, provenance, and local naming for BEDC.Derived.DetectabilityUp.

Chapter 7720

A Concrete Naming Certificate for $\mathsf{PicardIterationCauchy}^{\uparrow}$

Following Picard and Banach fixed-point iteration, the $\mathsf{PicardIterationCauchy}^{\uparrow}$ packet records the constructive route from a contractive Picard orbit to regular Cauchy completion data. It sits after the finite orbit surface of chapter 2359, the contraction endpoint surface of chapter 3529, the modulus route of chapter 292, the regular rational handoff of chapter 301, the Cauchy completion reflector of chapter 1585, and the terminal real-name boundary of chapter 38. The packet names the no-choice completion-neighbour exposed by the Picard residuals; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, select a fixed point, or identify quotient representatives.

Definition 7720.1 (PicardIterationCauchy carrier)**.** *A* $\mathsf{PicardIterationCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, B, M, R, F, E, H, C, Q, N)$$

whose coordinates are exactly:

1. *I is the accepted* $\mathsf{PicardIteration}^{\uparrow}$ *row displaying the finite orbit, residual-distance ledger, Cauchy-modulus extraction, and regular-sequence handoff.*
2. *B is the* $\mathsf{BanachContraction}^{\uparrow}$ *row recording that the orbit is consumed only through the contraction fixed-point route and not through a selected host endpoint.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row extracted from the Picard residual comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that exposes the regular rational name generated by I and M.*
5. *F is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row that admits the regular Cauchy name to the completion-facing reflector surface.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after the regular handoff and reflector route have been displayed.*
7. *H records componentwise* hsame *transport for I, B, M, R, F, E and for the continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from Picard iteration through contraction residuals, Cauchy modulus extraction, regular rational readback, completion reflection, and the real seal boundary.*
9. *Q is the* Pkg *provenance over* $\mathsf{PicardIterationCauchy}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{PicardIterationCauchy}^{\uparrow}}$ *row.*
The carrier exposes a constructive completion-neighbour for the Picard orbit. It contains no coordinate for a selected fixed point, a selected limit, quotient completion equality, countable choice, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7720.2 (PicardIterationCauchy NameCert obligations)**.** *For every accepted* $\mathsf{PicardIterationCauchy}^{\uparrow}$ *carrier* $P = (I, B, M, R, F, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PicardIterationCauchy}^{\uparrow}}$ *surface consists exactly of Picard-iteration admission, Banach-contraction route visibility, Cauchy-modulus exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff visibility,* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *admission,* $\mathsf{Real}^{\uparrow}$ *seal separation, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 7720.1.*

Proof. Project the carrier. The mathematical rows I, B, M, R, F, E are the only completion-facing coordinates, while H, C, Q, N are the only structural coordinates.

The Picard and contraction rows are read first. The row I exposes the finite orbit, residual-distance ledger, and Cauchy-modulus handoff of chapter 2359; the row B keeps the same orbit inside the constructive Banach route of chapter 3529.

Next read the completion-neighbour rows. The modulus row M is the source of the regular handoff R, the reflector row F consumes that regular handoff, and the seal row E is downstream of F.

Finally H, C, Q, N transport, replay, source, and name exactly these displayed rows. Since the carrier inventory has no selected endpoint, quotient representative, countable-choice schedule, host extensionality row, or ambient real-completeness row, the NameCert surface cannot export one. □

Theorem 7720.3 (PicardIterationCauchy contraction route)**.** *Every completion-facing read of an accepted* $\mathsf{PicardIterationCauchy}^{\uparrow}$ *carrier must pass through the displayed chain*

$$\mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{BanachContraction}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyCompletionReflector}^{\uparrow} \longmapsto$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *row is generated by the Picard residual-modulus route, the reflector row is generated by that regular handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal is available only after the reflector has been displayed. No consumer may replace this route by a selected fixed point, selected limit, quotient stream, countable-choice schedule, host equality, or ambient real-completeness principle.*

Proof. Begin with the Picard row I. By Definition 2359.5, the regular handoff is downstream of the finite orbit ledger, residual-distance row, and Cauchy-modulus row.

Read the Banach row B next. By Definition 3529.4, a fixed-point-facing consumer must keep the same Picard tail modulus upstream of the regular rational completion boundary.

Now project M, R, F, E from the carrier. The modulus row M is the only source for R, the reflector row F is the only completion-facing consumer of R, and E is only a downstream real-name seal.

The structural rows H, C, Q, N preserve this order under transport, replay, provenance, and local naming. Because no excluded object is a coordinate of the carrier, the displayed chain is the whole completion-neighbour route. □

Falsifiable prediction: Every accepted $\mathsf{PicardIterationCauchy}^{\uparrow}$ read factors through one PicardIteration row, one BanachContraction row, one CauchyModulus row, one RegSeqRat handoff, one CauchyCompletionReflector row, one RealUp seal boundary, hsame transport, Cont replay, Pkg provenance, and NameCert row before any completion-facing consumer reads a Picard endpoint.

Independence witness: picard-iteration, banach-contraction, cauchymodulus, regseqrat, cauchy-completionreflector, real: none is bijective with $\mathsf{PicardIterationCauchy}^{\uparrow}$ because the packet jointly stores the Picard orbit, contraction route, residual modulus, regular handoff, completion reflector, real seal boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardIterationCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{PicardIterationCauchy}^{\uparrow}$ packet is generated from a PicardIteration row, a BanachContraction row, a CauchyModulus row, a RegSeqRat handoff, a CauchyCompletionReflector row, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardIterationCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7720.1, Definition 7720.2, and Definition 7720.3 seed the constructive route from contractive Picard iteration to regular Cauchy completion-neighbour data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardIterationCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, selected fixed points, selected limits, quotient completion equality, countable choice, host equality, host function extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked Picard-iteration admission, Banach-contraction route visibility, Cauchy-modulus exposure, RegSeqRat handoff visibility, CauchyCompletionReflector admission, Real seal separation, transport, replay, provenance, and local naming.

Chapter 7721

A Concrete Naming Certificate for $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$ packet records the finite BEDC surface by which the harmonic reciprocal schedule $n \mapsto 1/(n+1)$ is read as a regular Cauchy tail estimate before a $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the tail-estimate route of chapter 1487. The packet names only the displayed harmonic threshold, the inspected finite window, the dyadic tolerance, the regular readback, the real seal, transport, replay, provenance, and local naming rows; it does not import quotient streams, selected limits, host real equality, choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7721.1 (Regular Cauchy harmonic estimate carrier)**.** *A* $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, W, D, R, E, T, C, P, N)$$

with the following displayed rows:

1. *S is the harmonic reciprocal schedule row. It stores the finite threshold calculation by which a requested dyadic tolerance is met by the displayed tail bound $1/(n+1)$.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected at and beyond the threshold exposed by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic error budget used for the harmonic tail read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row reached from the window W under the schedule and tolerance rows S, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that is available only after the harmonic schedule, window, tolerance, and regular readback rows have been displayed.*
6. *T records componentwise* hsame *transport for S, W, D, R, E, C, P, N.*
7. *C records displayed* Cont *replay from a tolerance request through the harmonic threshold, finite window, dyadic tolerance, regular readback, and real seal route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through those finite rows. It has no coordinate for quotient stream equality, a chosen limit, host real equality, trichotomy, open-cover compactness, or an ambient completeness principle.

Theorem 7721.2 (Regular Cauchy harmonic estimate NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$ *carrier* $H = (S, W, D, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHarmonicEstim}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the harmonic reciprocal schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *classifier transport to the common* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite surface;*
3. *stability of S, W, D, R, E under the componentwise* hsame *rows recorded in T;*
4. *route exactness requiring every harmonic tail read to pass through S, W, D, R, E in that order;*
5. *refusal rows excluding quotient streams, selected limits, host real equality, choice, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier $H = (S, W, D, R, E, T, C, P, N)$. The displayed analytic inventory is exactly the harmonic schedule, the finite stream window, the dyadic tolerance, the regular readback, and the terminal real seal.

Classifier transport is componentwise. The T row moves S, W, D, R, E and the administrative rows C, P, N to the same finite comparison surface, so any accepted comparison is already a comparison of the listed rows.

Route exactness is supplied by the replay row C. A tolerance request first reads the harmonic threshold in S, then the finite observations in W, then the dyadic tolerance ledger D, then the regular readback R, and only then the seal E.

The refusal row follows from the carrier inventory. Since no row stores quotient streams, selected limits, host real equality, choice, trichotomy, open-cover compactness, or ambient completeness, the local name cannot export any of those principles. □

Theorem 7721.3 (Regular Cauchy harmonic tail window factorization)**.** *For every accepted* $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$ *carrier* $H = (S, W, D, R, E, T, C, P, N)$*, every harmonic tail read factors through the finite threshold, window, tolerance, readback, and seal route*

$$S \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow E$$

before any consumer may inspect the transport, replay, provenance, or local naming rows T, C, P, N. In particular, the read cannot bypass the finite $\mathsf{StreamName}^{\uparrow}$ *window, cannot replace the dyadic tolerance by a host real comparison, and cannot obtain a selected limit or quotient-stream equality.*

Proof. Start with a harmonic tail read of an accepted carrier. The carrier exposes the harmonic reciprocal schedule only through S, so the first available threshold is the finite threshold row stored there.

Replay the route recorded in C. After S has exposed the threshold, the read is sent to the finite observations in W, then to the dyadic tolerance ledger D, then to the regular rational readback R, and finally to the real seal E.

Transport through T preserves this displayed order. The provenance row P and local name N bind the same route to $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$; they do not introduce another value-facing coordinate.

Thus every harmonic tail read factors through S, W, D, R, E before reaching T, C, P, N. The carrier has no row from which a bypass could read host real comparison, selected limits, quotient streams, choice, trichotomy, open-cover compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}$ packet is generated from a harmonic reciprocal schedule, a finite $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7721.1, Definition 7721.2, and Definition 7721.3 seed the finite harmonic-tail estimate surface for the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHarmonicEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, hsame stability, harmonic tail-window exactness, refusal rows, and provenance faithfulness for $S, W, D, R, E, T, C, P, N$.

Chapter 7722

A Concrete Naming Certificate for $\mathsf{CubicSpline}^{\uparrow}$

Following de Boor and Bishop-style constructive approximation, the $\mathsf{CubicSpline}^{\uparrow}$ packet records the finite Real-window route for a piecewise cubic spline over displayed knot windows. It refines the finite spline-space carrier of chapter 6894, the dyadic spline route of chapter 382, and the endpoint-jet surface of chapter 6668. Its readback boundary is routed through chapter 301, its stream-facing windows through chapter 126, and its terminal Real seal through chapter 38.

Definition 7722.1 (Cubic spline Real-window carrier)**.** *A* $\mathsf{CubicSpline}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, C, J, R, E, H, T, P, N)$$

with the following displayed rows:

1. *W is the knot-window row. It stores a finite increasing dyadic knot list, adjacent window membership tests, endpoint order evidence, and the displayed source window read by each consumer.*
2. *C is the dyadic cubic coefficient row. For every adjacent knot window it stores the four displayed dyadic coefficients of the local cubic polynomial, together with the cell index that selects that tuple.*
3. *J is the Hermite endpoint data row. It records the left and right value and first-derivative reads attached to each local cubic piece, and it identifies the shared endpoint data used by neighboring cells.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It turns each requested finite window and its selected cubic coefficient tuple into the regular rational observation used by downstream Real-window consumers.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the tolerance tokens and Real-name boundary obtained from the same readback row R.*
6. *H records componentwise* hsame *transport for knot windows, coefficient tuples, Hermite endpoint data, regular rational readbacks, Real seals, displayed replay, package provenance, and the local naming row.*
7. *T records displayed* Cont *replay of the route*

$$W, C, J \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{CubicSpline}^{\uparrow}$, $\mathsf{BishopSplineSpace}^{\uparrow}$, $\mathsf{DyadicSpline}^{\uparrow}$, $\mathsf{HermiteInterpolation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CubicSpline}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed knot windows, dyadic cubic coefficients, Hermite endpoint data, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows.

Theorem 7722.2 (Cubic spline Real-window NameCert obligations)**.** *For every accepted* $\mathsf{CubicSpline}^{\uparrow}$ *carrier*

$$S = (W, C, J, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CubicSpline}^{\uparrow}}$ *surface consists exactly of knot-window admission, dyadic cubic coefficient admission, Hermite endpoint-data admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public cubic-spline Real-window read must pass through the finite rows $W, C, J, R, E, H, T, P, N$ before it names a cell value, endpoint jet, or Real-sealed observation.*

Proof. Project the carrier of Definition 7722.1. The analytic rows are W, C, J, R, E: the displayed knot windows, dyadic cubic coefficients, Hermite endpoint data, regular rational readback, and Real seal.

The structural rows keep that route finite. The transport row H moves the selected window, coefficient tuple, endpoint jet, readback, and seal together, while T replays the displayed route from local cubic data to Real-sealed observations.

Finally the provenance and local naming rows P, N certify that the public object is exactly this packet. Since the classifier has no coordinate for a host spline function, quotient graph equality, global smoothness oracle, selected limit, or ambient completion, those objects cannot be read from the local NameCert surface. □

Theorem 7722.3 (Cubic spline piecewise-cubic handoff)**.** *For every accepted* $\mathsf{CubicSpline}^{\uparrow}$ *carrier $S = (W, C, J, R, E, H, T, P, N)$ and every displayed consumer window in W, the Real-facing read is obtained by selecting the adjacent knot cell, reading its dyadic cubic coefficient tuple from C, checking the corresponding Hermite endpoint row in J, routing the resulting finite cubic observation through R, and sealing that same observation in E. Neighboring cells meet only through their shared displayed endpoint data and the componentwise transports in H.*

Proof. Start with the requested displayed window. The finite knot-window row W either selects one adjacent cell or moves by finite inspection to the next displayed edge, using only the stored dyadic order evidence.

Once the cell is selected, project its coefficient tuple from C and its endpoint values and first-derivative reads from J. The shared endpoint entries in J are the only handoff data between neighboring cubic pieces.

Route the local cubic observation through the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R and then through the $\mathsf{Real}^{\uparrow}$ seal row E. Transport along H and replay through T preserve the same selected cell, coefficient tuple, endpoint data, readback, and seal.

Repack the result through P, N. The handoff therefore exports a finite piecewise-cubic Real-window reading and does not export a host spline function, quotient graph, global differentiability oracle, infinite knot row, selected limit, or ambient completion datum. □

Falsifiable prediction: An accepted $\mathsf{CubicSpline}^{\uparrow}$ read is refuted if a consumer obtains a Real-window cubic-spline value, endpoint jet, or piecewise-cubic handoff without first displaying the knot-window, coefficient, Hermite endpoint, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: bishop-spline-space, dyadicspline, hermite-interpolation, regseqrat, stream-name, real: $\mathsf{CubicSpline}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite knot windows, dyadic cubic coefficient rows, Hermite endpoint data, regular rational readback, Real sealing, stream-facing window provenance, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CubicSpline}^{\uparrow}}$.

Constructive story: A $\mathsf{CubicSpline}^{\uparrow}$ packet is generated from finite knot-window rows, dyadic cubic coefficient rows, Hermite endpoint-data rows, RegSeqRat readback rows, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CubicSpline}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7722.1, Definition 7722.2, and Definition 7722.3 seed the constructive Real-window cubic-spline route over displayed knot windows, dyadic cubic pieces, Hermite endpoint data, regular rational readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CubicSpline}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host spline functions, quotient graph equality, global differentiability, selected limits, infinite knot rows, or ambient completion.

Upgrade path: Obligation closure requires checked carrier admission, knot-window selection, dyadic cubic coefficient readback, Hermite endpoint handoff, Real sealing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.CubicSplineUp.

Chapter 7723

A Concrete Naming Certificate for RegularCauchyOscillationModulusExtraction$^{\uparrow}$

Following Bishop and Bridges on Cauchy criteria and oscillation moduli, the RegularCauchyOscillationModulusExtrac packet records the finite route from a regular Cauchy tail-oscillation ledger to an explicit modulus-extraction row. It sits after the finite observation names of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the threshold extraction surface of chapter 641, and the oscillation-modulus surface of chapter 2224. The packet names the tail-window induction that turns inspected oscillation bounds into a modulus row before any quotient stream, selected limit, host convergence theorem, or ambient Real$^{\uparrow}$ completeness principle is read.

Definition 7723.1 (Regular-Cauchy oscillation modulus extraction carrier)**.** *A* RegularCauchyOscillationModulusEx *carrier is a finite* BHist *packet*

$$M = (S, R, D, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *source schedule naming the finite tail windows inspected by the request.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row used to compare the stream values inside those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, carrying the requested dyadic bound and the halving data used by the tail estimate.*
4. *O is the* OscillationModulus$^{\uparrow}$ *row that records the tail-oscillation bound on each displayed window.*
5. *E is the* CauchyModulusExtraction$^{\uparrow}$ *row that extracts a threshold from the displayed schedule, readback, tolerance, and oscillation rows.*
6. *W is the terminal* Real$^{\uparrow}$ *seal available only after the extracted threshold row has been displayed.*
7. *H stores componentwise* hsame *transport for the source, readback, dyadic, oscillation, extraction, seal, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay that moves a tolerance request through tail-window inspection, oscillation comparison, threshold extraction, and Real-facing sealing.*
9. *P is the* Pkg *provenance over* RegularCauchyOscillationModulusExtraction$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, OscillationModulus$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, R, D, O, E, W, H, C, P, N$.

Theorem 7723.2 (Regular-Cauchy oscillation modulus extraction NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}$ *carrier $M = (S, R, D, O, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission, finite source-window admission through* $\mathsf{StreamName}^{\uparrow}$*, regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic tolerance budgeting through* $\mathsf{DyadicRatCore}^{\uparrow}$*, tail-oscillation exposure through* $\mathsf{OscillationModulus}^{\uparrow}$*, threshold extraction through* $\mathsf{CauchyModulusExtraction}^{\uparrow}$*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It exports no quotient stream equality, selected modulus oracle, countable-choice selector, host convergence theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier. The analytic rows are S, R, D, O, E, W, and the structural rows are H, C, P, N.

The source side is finite. A tolerance request first reads the $\mathsf{StreamName}^{\uparrow}$ row S, then compares the requested tail values through the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{DyadicRatCore}^{\uparrow}$ budget D.

The oscillation side is also finite. The row O records the tail-oscillation bound on the displayed windows, and the extraction row E is read only after S, R, D, O have all been displayed. The terminal row W is therefore downstream of the extracted threshold, not a separate completeness principle.

Finally H, C, P, N preserve this same route under transport, replay, provenance, and local naming. Since no other coordinate occurs in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 7723.3 (Regular-Cauchy oscillation modulus extraction tail route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}$ *packet must read the finite tail route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{OscillationModulus}^{\uparrow} \longmapsto \mathsf{CauchyModulusExtraction}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with H, C, P, N before any modulus threshold is public. The route is obtained by finite tail-window induction over the displayed source schedule, and it does not use quotient streams, a selected limit, a host Cauchy theorem, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct over the finite tail-window schedule stored in S. In the base window, the tolerance request reads D and the regular-rational comparison in R, so the first oscillation bound is an explicit row of O.

For the induction step, assume the earlier inspected windows already route through S, R, D, O. The next displayed window is added by the same $\mathsf{StreamName}^{\uparrow}$ schedule, its rational comparisons are read through R, and its dyadic budget is read through D. The oscillation row O supplies the matching bound for that window.

After the finite schedule is exhausted, the extraction row E receives exactly the inspected windows and their oscillation bounds. The $\mathsf{Real}^{\uparrow}$ seal W is read only after E, while H, C, P, N replay the same route componentwise. Thus every consumer reaches the public threshold through the displayed tail-window induction and no excluded host or choice coordinate is available. □

Falsifiable prediction: Every $\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}$ consumer read must expose a StreamName tail schedule, RegSeqRat readback, DyadicRatCore tolerance budget, OscillationModulus tail ledger, CauchyModulusExtraction threshold row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, oscillationmodulus, cauchymodulusextraction, real: $\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the inspected tail schedule, rational readback, dyadic budget, oscillation ledger, extracted threshold, Real-facing seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}$ packet is generated from StreamName tail windows, RegSeqRat readback, DyadicRatCore tolerance data, an OscillationModulus tail ledger, a CauchyModulusExtraction threshold row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7723.1, Definition 7723.2, and Definition 7723.3 seed the finite tail-window route from regular Cauchy oscillation bounds to modulus extraction over StreamName, RegSeqRat, DyadicRatCore, OscillationModulus, CauchyModulusExtraction, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean-side verification, bridge checking, quotient stream equality, selected moduli, host convergence, countable choice, trichotomy, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, tail-window induction, dyadic tolerance exactness, oscillation ledger compatibility, extraction locality, RealUp seal non-escape, transport, replay, provenance, and local naming for the displayed packet.

Chapter 7724

A Concrete Naming Certificate for $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy names, the $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$ packet records the finite normal-form surface by which a displayed rational name is read before the terminal $\mathsf{Real}^{\uparrow}$ seal is available. It sits over the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the real-name boundary of chapter 38, the sealing discipline of chapter 6749, and the modulus ledger of chapter 7483. The packet names only the displayed normal-form rows; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, trichotomy, countable choice, or host real equality.

Definition 7724.1 (Regular Cauchy name normal-form carrier)**.** *A* $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, M, E, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare rational observations at the requested precision.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing exactly the inspected rational-name observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the same finite windows.*
4. *M is the* $\mathsf{RegularSequenceModulus}^{\uparrow}$ *row that records the displayed regularity modulus for the rational name.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, S, R, M have been displayed.*
6. *B is the* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *row that records the Bishop-style Cauchy-name admission boundary for the same route.*
7. *H records componentwise* hsame *transport for the tolerance, window, readback, modulus, seal, Bishop admission, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a name-normalization request through $D, S, R, M, E, B, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularSequenceModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopCauchyNameSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNameNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, M, E, B, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyNameNormalFormUp.RegularCauchyNameNormalFormTasteGate single carrier alignment`

Theorem 7724.2 (Regular Cauchy name normal-form NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$ *carrier $F = (D, S, R, M, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNameNorm}}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite stream window, regular rational readback, regular-sequence modulus, real seal, Bishop Cauchy-name seal, componentwise transport, displayed replay, package provenance, and local naming rows. Every real-facing read from the packet must pass through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularSequenceModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the Bishop seal and structural rows can be consumed.

Proof. Project the carrier $F = (D, S, R, M, E, B, H, C, P, N)$. The visible analytic inventory is exactly the dyadic tolerance row, finite stream window, regular rational readback, regularity modulus, real seal, and Bishop Cauchy-name admission row.

The replay row C fixes the order of use. A name-normalization request first reads the tolerance ledger D, then the finite window S, then the regular readback R, then the regularity modulus M, and only then exposes the terminal real seal E and the Bishop admission boundary B.

Transport and provenance preserve the same packet. The rows H, P, N move, source, and name the displayed normal-form surface, but they do not add quotient stream equality, a selected limit, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 7724.3 (Regular Cauchy normal-form route nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$ *packet receives a finite normal-form route through $D, S, R, M, E, B, H, C, P, N$. The consumer cannot bypass the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RegularSequenceModulus}^{\uparrow}$ *modulus row, or the* $\mathsf{BishopCauchyNameSeal}^{\uparrow}$ *admission boundary by appealing to quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a real-facing read of an accepted carrier. The only terminal real row is E, and the carrier places it after the displayed tolerance, stream, readback, and modulus rows.

Read the continuation row C. It sends the request through D, S, R, M before E and B are available. Componentwise transport through H preserves those fields rather than replacing them by a host comparison or a quotient equality.

The provenance and naming rows P, N bind the same finite route to the local certificate. Since the carrier contains no coordinate for selected limits, trichotomy, countable choice, host real equality, quotient stream equality, or completeness of all $\mathsf{Real}^{\uparrow}$ names, no consumer can obtain such data from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyNameNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyNameNormalForm}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance ledger, StreamNameUp finite windows, RegSeqRatUp readback, a RegularSequenceModulusUp row, a RealUp seal, a BishopCauchyNameSealUp admission row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyNameNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7724.1, Definition 7724.2, and Definition 7724.3 seed the finite normal-form route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RegularSequenceModulusUp, RealUp, BishopCauchyNameSealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyNameNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient stream equality, selected limits, trichotomy, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, route exactness, transport stability, Bishop admission stability, refusal rows, and provenance faithfulness for BEDC.Derived.RegularCauchyNameNormalFormUp.

Chapter 7725

A Concrete Naming Certificate for HyperbolicVisualAngle$^{\uparrow}$

The HyperbolicVisualAngle$^{\uparrow}$ packet realizes the boundary-triple and visual readback surface of chapter 9463 as finite hyperbolic phase data. It sits after the disk-coordinate surface of chapter 377, the metric-read surface of chapter 3474, the boundary-geodesic packet of chapter 3491, the cross-ratio route of chapter 3948, and the terminal Real$^{\uparrow}$ seal of chapter 38. The packet names a finite visual-angle carrier; it does not complete the disk, quotient boundary triples, select a host geodesic, identify visual angle with physical observation, or import real completeness.

Definition 7725.1 (Hyperbolic visual-angle carrier)**.** *A* HyperbolicVisualAngle$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, T, M, B, X, R, J, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UnitDisk$^{\uparrow}$ *disk-coordinate row for the observer point and the inspected boundary data.*
2. *T is the boundary-triple row naming the ordered finite boundary ends used by the visual-angle read.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk data.*
4. *B is the* HyperbolicGeodesicBoundary$^{\uparrow}$ *row that exposes geodesic-boundary incidence for the triple.*
5. *X is the* HyperbolicCrossRatio$^{\uparrow}$ *row carrying the finite cross-ratio comparison attached to the same boundary triple.*
6. *R is the* Real$^{\uparrow}$ *seal row for the angle value, available only after U, T, M, B, X have been displayed.*
7. *J records classifier stability for disk coordinates, boundary triples, metric reads, geodesic-boundary incidence, cross-ratio data, and the real seal.*
8. *H records componentwise* hsame *transport for $U, T, M, B, X, R, J, C, P, N$.*
9. *C records displayed* Cont *replay from a visual-angle request through boundary triple, hyperbolic metric, geodesic-boundary, cross-ratio, real-seal, transport, provenance, and naming rows.*
10. *P is the* Pkg *provenance over* HyperbolicVisualAngle$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBoundary$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualAngle}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $U, T, M, B, X, R, J, H, C, P, N$. It has no coordinate for a completed Poincare disk, a Klein-model equality, quotient ideal-boundary equality, selected geodesic representatives, physical observation laws, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7725.2 (Hyperbolic visual-angle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualAngle}^{\uparrow}$ *carrier $A = (U, T, M, B, X, R, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualAngle}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission through U, boundary-triple admission through T, hyperbolic metric-read exactness through M, geodesic-boundary incidence through B, cross-ratio comparison through X, real-seal exposure through R, classifier stability through J, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7725.1. The accepted rows are exactly $U, T, M, B, X, R, J, H, C, P, N$, so carrier admission exposes disk coordinates, a boundary triple, metric reads, geodesic-boundary incidence, cross-ratio data, a real seal, classifier stability, transport, replay, provenance, and local naming.

The geometric obligations are ordered by the replay row C. A visual-angle request first reads the disk-coordinate and boundary-triple data, then the hyperbolic metric and geodesic-boundary incidence, then the cross-ratio comparison, and only then the real-valued seal.

The structural rows preserve that same surface. Classifier stability, componentwise transport, provenance, and local naming cannot add a completed disk, quotient boundary equality, selected geodesic representative, physical observation law, or ambient real-completeness row, so the listed obligations exhaust the local certificate. □

Theorem 7725.3 (Hyperbolic visual-angle route nonescape)**.** *Every visual-angle read from an accepted* $\mathsf{HyperbolicVisualAngle}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicCrossRatio}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $U, T, M, B, X, R, J, H, C, P, N$. Thus the packet exposes finite boundary-triple, hyperbolic metric, cross-ratio, real-seal, transport, provenance, and NameCert rows, but it exports no completed disk, quotient boundary triple, selected host geodesic, physical-observation equality, Klein-model equality, or ambient real-completeness principle.

Proof. Start with an accepted visual-angle request. The disk-coordinate row U and boundary-triple row T give the only available source data for the angle read.

Replay the route recorded in C. The metric row M and boundary row B make the geodesic incidence visible, the cross-ratio row X carries the boundary comparison, and the real seal R is exposed only after those rows are present.

The classifier, transport, provenance, and naming rows J, H, P, N preserve the same finite packet. Since the carrier lacks any coordinate for completed disk objects, quotient boundary triples, selected host geodesics, physical-observation equality, Klein-model equality, or real completeness, no visual-angle consumer can read such data from the packet. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicVisualAngle}^{\uparrow}$ read displays a UnitDisk observer row, a boundary-triple row, a HyperbolicMetric row, a HyperbolicGeodesicBoundary row, a HyperbolicCrossRatio row, a RealUp seal, classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before exporting a visual-angle value.

Independence witness: unitdisk, hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, real: $\mathsf{HyperbolicVisualAngle}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an observer disk coordinate, ordered boundary triple, metric read, geodesic-boundary incidence, cross-ratio comparison, Real seal, classifier stability, transport, replay, provenance, and local NameCert rows.

Lean `TasteGate.lean` for this packet will provide `Nontrivial HyperbolicVisualAngleUp` and `FieldFaithful HyperbolicVisualAngleUp` instances for the same displayed carrier. Those instances must keep $U, T, M, B, X, R, J, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualAngle}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualAngle}^{\uparrow}$ packet is generated from UnitDiskUp observer rows, boundary-triple rows, HyperbolicMetricUp reads, HyperbolicGeodesicBoundaryUp incidence, HyperbolicCrossRatioUp comparison data, a RealUp seal, classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualAngle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7725.1, Definition 7725.2, and Definition 7725.3 seed the finite visual-angle route over boundary triple, hyperbolic metric, geodesic-boundary, cross-ratio, Real seal, transport, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualAngle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed disks, quotient boundary triples, selected host geodesics, physical-observation equality, Klein-model equality, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-coordinate admission, boundary-triple admission, metric-read exactness, geodesic-boundary incidence, cross-ratio comparison, Real-seal exposure, classifier stability, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicVisualAngleUp.

Chapter 7726

A Concrete Naming Certificate for $\mathsf{CauchyRealRegularization}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{CauchyRealRegularization}^{\uparrow}$ packet records the finite BEDC route that turns a displayed Cauchy-real name into a regular rational sequence row before the terminal real seal is read. It sits after the finite-window discipline of chapter 126, the dyadic tolerance arithmetic of chapter 291, the regular rational readback surface of chapter 301, and the terminal seal of chapter 38. For fast or modulus-indexed presentations it also aligns with chapter 1589 and chapter 7427, but it does not identify quotient classes or choose limits outside the displayed packet.

Definition 7726.1 (CauchyRealRegularization carrier)**.** *A* $\mathsf{CauchyRealRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, S, D, R, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the source Cauchy-real name row, available only as finite precision requests and responses.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected stages, requested tolerances, and replay addresses.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that accounts for every comparison made while regularizing the source name.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularization row produced from the same windows and dyadic ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after S, D, R have been displayed.*
6. *H records componentwise* hsame *transport for source name, windows, tolerances, regularization row, real seal, replay, provenance, and local naming.*
7. *T records displayed* Cont *replay from a source precision request through S, D, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these nine rows. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7726.2 (CauchyRealRegularization NameCert obligations)**.** *For every accepted* CauchyRealRegularization *carrier* $K = (C, S, D, R, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for the source Cauchy-real name row; finite window exposure through* S*; dyadic tolerance accounting through* D*; handoff to the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*; terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary through* E*; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; local naming; and non-escape from quotient streams, selected limits, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7726.1. The only source-name, window, dyadic-ledger, regularization, and real-seal data available to a consumer are C, S, D, R, E.

The regularization route first exposes the finite windows in S and accounts for every tolerance comparison in D. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of those two rows, so no regular readback can be obtained from a detached Cauchy stream or a selected subsequence.

The terminal seal is then read through E, with H, T, P, N preserving, replaying, sourcing, and naming the same finite route. Since the carrier contains no quotient, selected-limit, choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealRegularization}^{\uparrow}$ packet is generated from a source Cauchy-real name row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat regularization row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 7726.1 and Definition 7726.2 seed the finite route from Cauchy-real names to regular rational readback and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealRegularization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completeness of RealUp, quotient stream equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, dyadic tolerance accounting, RegSeqRat handoff, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRealRegularizationUp.

Chapter 7727

A Concrete Naming Certificate for $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$

Following Tietze and Urysohn, the $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$ packet isolates one finite bounded correction stage inside the Tietze route. It reads the normal-space separation source of chapter 1866, the Urysohn separator certificate of **??**, the graph replay surface of chapter 130, the bounded value seal of chapter 38, and the full extension route of chapter 1956. The packet also keeps the zero and cozero boundary discipline visible through chapter 1858 and chapter 1857. It names only one displayed correction row, not the full extension theorem.

Definition 7727.1 (Tietze finite-step carrier)**.** *A* $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (N, D, B, U, M, G, R, H, K, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *source row, exposing the ambient topology and the closed-pair separation discipline used by the correction stage.*
2. *D is the closed-domain row on which the current bounded partial extension is compared with the input data.*
3. *B is the bounded* $\mathsf{Real}^{\uparrow}$ *interval row for the current error scale, including endpoint bounds for the stage.*
4. *U is one* $\mathsf{UrysohnLemma}^{\uparrow}$ *separator and correction row. It records the closed positive and negative error regions, the separator graph, and the bounded correction amplitude.*
5. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the current partial graph before the correction is applied.*
6. *G is the corrected* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row after adding the Urysohn correction to the current partial graph.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the corrected values and the decreased error bound.*
8. *H records componentwise* hsame *transport for the normal-space source, closed-domain row, bounded interval, separator row, current graph, corrected graph, Real seal, replay, provenance, and local naming.*
9. *K records displayed* Cont *replay from a finite-step request through $N, D, B, U, M, G, R, H, P, L$.*

10. *P is the* Pkg *provenance over* $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZeroSet}^{\uparrow}$, $\mathsf{CozeroSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{TietzeExtensionFiniteStep}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, D, B, U, M, G, R, H, K, P, L$. It has no coordinate for the completed Tietze extension, selected complements, excluded middle, host function extensionality, open-cover compactness, or ambient Real completeness.

Theorem 7727.2 (Tietze finite-step NameCert obligations)**.** *For every accepted* $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$ *carrier*

$$S = (N, D, B, U, M, G, R, H, K, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{TietzeExtensionFiniteStep}^{\uparrow}}$ *surface consists exactly of carrier admission, closed-domain exposure, separator-stage routing, endpoint agreement, bounded-error decrease, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Carrier admission is the projection of Definition 7727.1. The accepted packet displays exactly the normal-space source, closed-domain row, bounded interval, one Urysohn correction row, current graph, corrected graph, Real seal, and structural support.

Closed-domain exposure is read through N and D. A finite correction stage may compare the current graph with the input data only after the displayed closed-domain row is available over the same normal-space source.

Separator-stage routing is the projection to U. The correction amplitude and the separating graph are supplied by the $\mathsf{UrysohnLemma}^{\uparrow}$ row, so no selected complement or excluded-middle branch can be consumed.

Endpoint agreement and bounded-error decrease are read through M, G, B, R. The current and corrected graphs are both $\mathsf{ContinuousMap}^{\uparrow}$ rows, while the Real seal records that the corrected graph remains inside the bounded interval and decreases the displayed error scale.

Transport, replay, provenance, and naming are exhausted by H, K, P, L. Since the classifier has no further coordinate, the listed rows are the whole local NameCert surface. □

Theorem 7727.3 (Tietze finite-step Urysohn stage)**.** *Every correction-stage consumer of an accepted* $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$ *packet factors through*

$$\mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{UrysohnLemma}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the surrounding $\mathsf{TietzeExtension}^{\uparrow}$ *route can consume the corrected graph. The stage exports a bounded corrected graph and an error-decrease row, not a completed extension theorem.*

Proof. Start with an accepted carrier $S = (N, D, B, U, M, G, R, H, K, P, L)$. The normal-space source N and closed-domain row D must be visible before a correction request has a domain on which the current error can be read.

The next row is U. It is the unique Urysohn separator and correction coordinate in the packet, so every correction-stage consumer reaches the separator through $\mathsf{NormalSpace}^{\uparrow}$ and $\mathsf{UrysohnLemma}^{\uparrow}$.

The corrected graph is then read through M and G. Both are $\mathsf{ContinuousMap}^{\uparrow}$ graph rows, and their agreement along the displayed domain is preserved by the structural rows.

Finally B and R supply the $\mathsf{Real}^{\uparrow}$ boundary for the decreased error scale. Only after these rows are displayed may the surrounding $\mathsf{TietzeExtension}^{\uparrow}$ carrier consume the corrected graph as one ladder stage. □

Closure status for $\mathsf{NameCert}_{\mathsf{TietzeExtensionFiniteStep}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TietzeExtensionFiniteStep}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7727.1, Definition 7727.2, and Definition 7727.3 seed the finite bounded Urysohn correction stage inside the Tietze extension route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TietzeExtensionFiniteStep}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full Tietze extension theorem, host function extensionality, selected complements, excluded middle, open-cover compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, closed-domain exposure, separator-stage routing, endpoint agreement, bounded-error decrease, transport, replay, provenance, and local naming for BEDC.Derived.TietzeExtensionFiniteStep

Constructive story: A $\mathsf{TietzeExtensionFiniteStep}^{\uparrow}$ packet is generated from a normal-space source row, closed-domain row, bounded Real interval row, one Urysohn separator row, current and corrected ContinuousMap graph rows, a Real error seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7728

Mankiewicz Extension NameCert Construction

Following Mankiewicz, the $\mathsf{MankiewiczExtension}^{\uparrow}$ packet records the finite BEDC surface on which an isometry defined on an open convex subset of a real normed space is inspected before it is handed to an affine isometry route. It is a standard neighbor of the Mazur–Ulam surface of chapter 5194: Mazur–Ulam names the surjective whole-space route, while this chapter names the open-convex-domain route whose midpoint and affine ledgers are visible before any global function equality is read. The packet sits over $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and a local $\mathsf{NameCert}$ row; it does not import host function extensionality, choice of an external extension, quotient point equality, or ambient completeness.

Definition 7728.1 (Mankiewicz extension carrier)**.** *A* $\mathsf{MankiewiczExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, F, U, V, I, J, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed carrier row, exposing its* $\mathsf{Norm}^{\uparrow}$ *read, induced* $\mathsf{Met}^{\uparrow}$ *distance row, vector operations,* $\mathsf{Real}^{\uparrow}$ *scalar reads, transport, replay, provenance, and local naming.*
2. *F is the target normed carrier row with the same norm, metric, vector, scalar, transport, replay, provenance, and local naming surface.*
3. *U is the open convex source row: it records the displayed* $\mathsf{ConvexSet}^{\uparrow}$ *membership ledger, the open-ball witnesses used by local midpoint reads, and the source-domain inclusion into E.*
4. *V is the image-domain row in F, carrying the accepted target points reached from the source row and the displayed range membership used by later affine reads.*
5. *I is the isometry row over U and V, storing accepted source histories, accepted target histories, and* $\mathsf{Met}^{\uparrow}$ *distance equality through the rows of E and F.*
6. *J is the local midpoint and segment ledger, recording when midpoint and dyadic convex reads remain inside the open convex source row and how their image rows are compared in the target.*
7. *A is the affine ledger, recording the local affine rule extracted from isometry, midpoint, dyadic convex, additive, and* $\mathsf{Real}^{\uparrow}$ *scalar readback rows.*
8. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row, exposing the displayed continuous-map route by which the affine ledger is consumed by downstream normed-geometry packets.*

9. *H records componentwise* hsame *transport for E, F, U, V, I, J, A, Q and for the support rows.*
10. *C records displayed* Cont *replay for open-convex membership, distance comparison, midpoint inspection, affine readback, continuous-map handoff, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{MankiewiczExtension}^{\uparrow}$, $\mathsf{MazurUlamIsometry}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, F, U, V, I, J, A, Q, H, C, P, N$. It has no coordinate for host function extensionality, a selected extension outside the displayed rows, quotient equality of maps, external affine hulls, or ambient Banach completeness.

Theorem 7728.2 (Mankiewicz extension NameCert obligations)**.** *For every accepted* $\mathsf{MankiewiczExtension}^{\uparrow}$ *carrier $M = (E, F, U, V, I, J, A, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}$ *surface consists exactly of source and target normed-carrier admission, open convex source admission, image-domain admission, isometry-graph exactness, midpoint and segment stability inside the open convex row, affine-ledger readback,* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7728.1. The normed and metric rows are E, F, the open convex and image-domain rows are U, V, the isometry row is I, and the geometric readback rows are J, A, Q.

Open convex admission is read before any midpoint or affine claim. The row U supplies the local ball and convex membership data needed to keep midpoint and segment reads inside the displayed source surface, while V records the corresponding accepted target rows.

The isometry and affine obligations then follow the displayed order. The row I supplies $\mathsf{Met}^{\uparrow}$ distance equality, the row J records midpoint and dyadic convex stability, the row A records affine readback through finite $\mathsf{Real}^{\uparrow}$ scalar data, and Q hands the resulting route to $\mathsf{ContinuousMap}^{\uparrow}$.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no host extension coordinate, quotient equality of maps, selected external affine map, or ambient completeness row, the listed rows exhaust the local NameCert surface. □

Theorem 7728.3 (Mankiewicz affine isometry route)**.** *Every affine isometry read exported by an accepted* $\mathsf{MankiewiczExtension}^{\uparrow}$ *carrier factors through the open convex source row U, the source and target* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *rows E, F, the isometry row I, the midpoint and segment ledger J, the affine ledger A, the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}$ *row N.*

Consequently the packet exposes only a local open-convex affine-isometry route over the displayed finite histories. It exports no host function extensionality, selected extension outside the packet, quotient map equality, hidden inverse, or ambient completeness principle.

Proof. Begin with an affine isometry request. The carrier first exposes U, because the Mankiewicz route applies to an isometry on an open convex source surface rather than to a whole-space graph.

Read the distance and isometry data through E, F, I. These rows give the $\mathsf{Norm}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ distances and the displayed equality data for accepted source and target histories.

Next use the midpoint and segment ledger J. It records the finite open-convex checks needed to keep midpoint and dyadic segment reads inside U, so the affine ledger A receives only displayed source-domain data and $\mathsf{Real}^{\uparrow}$ scalar reads.

The $\mathsf{ContinuousMap}^{\uparrow}$ row Q is then a handoff, not a second extension principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the same route and add no host function-extensionality coordinate. The excluded global objects are absent from the carrier, so they cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{MankiewiczExtension}^{\uparrow}$ packet is generated from source and target normed rows, an open convex source row, image-domain data, an isometry graph, midpoint and affine ledgers, a $\mathsf{ContinuousMap}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7728.1, Definition 7728.2, and Definition 7728.3 seed the finite Mankiewicz open-convex isometry route over $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MankiewiczExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, selected extensions outside the displayed packet, quotient equality of maps, hidden inverses, ambient Banach completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-convex domain admission, isometry exactness, midpoint and affine ledger stability, continuous-map handoff, transport, replay, provenance, and local naming for the same finite packet.

Chapter 7729

A Concrete Naming Certificate for $\mathsf{NeronModel}^{\uparrow}$

Following Neron and Bosch-Lutkebohmert-Raynaud, the $\mathsf{NeronModel}^{\uparrow}$ packet records the local arithmetic-geometric surface by which an abelian variety over the fraction field of a discrete valuation ring extends to a smooth separated group scheme with a local universal mapping property. The chapter packages only the finite BEDC rows that name this local surface: it does not assert global existence of Neron models, choose a minimal regular model, or import a host valuative criterion beyond the displayed local packet.

The packet sits over the valued source of chapter 173, the arithmetic base row of chapter 169, the scheme carrier of chapter 153, the algebraic group surface of chapter 41, and the elliptic-curve example source of chapter 175. These dependencies are read as BEDC rows and not as detached host structures.

7729.1 Carrier definition

Definition 7729.1 (NeronModel local carrier)**.** *A* $\mathsf{NeronModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (K, \mathcal{O}, G, A, \eta, U, H, C, P, N)$$

with the following displayed rows:

1. *K is a* $\mathsf{LocalField}^{\uparrow}$ *or DVR fraction-field source row, carrying the valuation and residue boundary through a finite local-field certificate;*
2. *$\mathcal{O}$ is the* $\mathsf{RingOfIntegers}^{\uparrow}$ *or valuation-ring base row over which the local model is read;*
3. *G is a smooth separated group-scheme row over $\mathcal{O}$, with the scheme part routed through* $\mathsf{Scheme}^{\uparrow}$ *and the group law routed through* $\mathsf{Group}^{\uparrow}$*;*
4. *A is the generic-fiber abelian-variety row, with elliptic-curve inputs allowed through the* $\mathsf{EllipticCurve}^{\uparrow}$ *source when the generic fiber is one-dimensional;*
5. *η is the generic-fiber comparison row identifying the fraction-field fiber of G with the displayed row A;*
6. *U is the extension-across-valuation ledger: every smooth test scheme over $\mathcal{O}$ that is read by the packet carries a generic-fiber map to A and a displayed extension row to G;*

7. *H stores componentwise* hsame *transports for the local field, valuation ring, scheme, group law, generic fiber, extension ledger, provenance, and local naming rows;*
8. *C stores the* Cont *replay routes for source inspection, base change to the fraction field, group-scheme readback, extension-ledger readback, and local universal mapping readback;*
9. *P is the* Pkg *provenance over* $\mathsf{NeronModel}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, $\mathsf{RingOfIntegers}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{EllipticCurve}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $K, \mathcal{O}, G, A, \eta, U, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.NeronModelUp.NeronModelTasteGate single carrier alignment`

7729.2 The certificate

Theorem 7729.2 (NeronModel NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}$ *obligation surface consists exactly of the carrier of Definition 7729.1, the inherited* $\mathsf{LocalField}^{\uparrow}$ *and* $\mathsf{RingOfIntegers}^{\uparrow}$ *source rows, the* $\mathsf{Scheme}^{\uparrow}$ *and* $\mathsf{Group}^{\uparrow}$ *rows making G a smooth separated group scheme, the generic-fiber row A, the comparison row η, the extension ledger U, the componentwise* hsame *transports, the* Cont *replay rows, and the* Pkg *provenance. No accepted carrier exports a detached host scheme, a chosen regular model, an ambient abelian-variety equality, a global existence theorem, or a choice-heavy valuative selection.*

Proof. Project the carrier of Definition 7729.1. The source rows K and $\mathcal{O}$ bind the local valuation boundary, while the rows G and A bind the scheme, group, and generic-fiber data to displayed BEDC certificates.

Next read the comparison and extension rows. The row η names the generic-fiber comparison inside the finite packet, and U lists only the local extension attempts already routed through a smooth test scheme and the same valuation boundary.

The classifier and stability obligations are componentwise. The rows H and C transport and replay each visible coordinate, and P keeps the dependency provenance attached to the same packet. Repacking these rows gives the listed $\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}$ surface and leaves no coordinate for the excluded host data. □

Theorem 7729.3 (NeronModel local universal route)**.** *Every accepted* $\mathsf{NeronModel}^{\uparrow}$ *carrier that exports a local universal mapping readback must expose the local-field row K, the valuation-ring row $\mathcal{O}$, the smooth separated group-scheme row G, the generic-fiber row A, the comparison row η, and the extension ledger U. The readback is therefore a finite route*

$$K, \mathcal{O}, G, A, \eta, U, H, C, P, N$$

through displayed BEDC data, not a hidden appeal to a host Neron model or to an unrecorded extension theorem.

Proof. Start with the readback request. By the carrier definition, a universal mapping read can only enter through the extension ledger U, and that ledger is stored after the local field, valuation-ring base, group-scheme, generic-fiber, and comparison rows.

Project the source rows. The local valuation boundary comes from K and $\mathcal{O}$, while the target of the extension is the row G whose generic fiber is compared to A by η.

Finally apply the replay and provenance rows. The rows H, C, P, N transport, replay, and name the same finite packet; they add no further source of maps. Hence any accepted universal readback exposes the whole displayed route before the export is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}$.

Constructive story: A $\mathsf{NeronModel}^{\uparrow}$ packet is generated from a local-field or DVR source row, a valuation-ring base row, a smooth separated group-scheme row, a generic-fiber abelian-variety row, a comparison row, a local extension ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7729.1, Definition 7729.2, and Definition 7729.3 bind the finite local carrier and its local universal-route surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeronModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global Neron-model existence, minimal regular models, selected extensions, bridge data, or any choice-heavy valuative theorem beyond the displayed finite local packet.

Upgrade path: The next theory step is a scoped carrier theorem over the local extension ledger and its transport rows.

Chapter 7730

A Concrete Naming Certificate for $\mathsf{CofilteredLimit}^{\uparrow}$

Following Mac Lane's categorical treatment of limits and the Bourbaki–Kelley filter language for directed refinement, the $\mathsf{CofilteredLimit}^{\uparrow}$ packet records a finite cofiltered-cone limit surface rather than a host inverse limit. It sits after the category carrier of chapter 61, the displayed base and cofinality routes of chapter 1601 and chapter 3955, and the generic finite limit packet of chapter 1678. The packet names only the finite cone rows that a consumer may inspect before a limit-shaped read is admitted.

Definition 7730.1 (Cofiltered-limit carrier)**.** *A* $\mathsf{CofilteredLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, K, T, C, H, R, P, N)$$

with the following displayed rows:

1. *I is a finite cofiltered index row, listing the inspected objects, comparison arrows, and common-refinement witnesses.*
2. *D is the diagram row over* $\mathsf{Cat}^{\uparrow}$*, naming the object and morphism data read from the finite portion of the diagram.*
3. *K is the cone-component row, assigning each inspected index a component arrow into the diagram row.*
4. *T is the transition and cofinality row, recording the finite common-refinement requests through* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CofinalFilter}^{\uparrow}$*.*
5. *C is the cone-compatibility ledger, checking that every displayed transition square read by I, D, K, T commutes inside the named category surface.*
6. *H stores componentwise* hsame *transport for the index, diagram, cone, transition, compatibility, replay, provenance, and local naming rows.*
7. *R stores the displayed* Cont *replay from a cofiltered-cone request through I, D, K, T, C, H.*
8. *P is the* Pkg *provenance over* $\mathsf{CofilteredLimit}^{\uparrow}$*,* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{FilterBase}^{\uparrow}$*,* $\mathsf{CofinalFilter}^{\uparrow}$*,* $\mathsf{Limit}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofilteredLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their transports, their displayed replay, package provenance, and local naming. The carrier has no coordinate for quotient limits, selected universal cones, full completeness, choice, or bridge checking.

Theorem 7730.2 (Cofiltered-limit NameCert obligations)**.** *For every accepted* $\mathsf{CofilteredLimit}^{\uparrow}$ *carrier* $L = (I, D, K, T, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofilteredLimit}^{\uparrow}}$ *surface consists exactly of carrier admission, finite cofiltered-index admission, diagram-row admission over* $\mathsf{Cat}^{\uparrow}$*, cone-component exposure, transition and cofinality routing through* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CofinalFilter}^{\uparrow}$*, cone-compatibility ledger coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside* $I, D, K, T, C, H, R, P, N$*.*

Proof. Project the carrier. The analytic rows are I, D, K, T, C, and the structural rows are H, R, P, N.

The index side is finite. A cofiltered request first reads the displayed index row I, then obtains the diagram row D and the cone components K only for the inspected finite fragment.

The refinement side is finite as well. The row T records common-refinement and cofinality requests through the displayed filter-base routes, and the ledger C checks the corresponding cone-compatibility squares. The structural rows H, R, P, N transport, replay, source, and name exactly this same finite packet.

Since the carrier contains no further coordinate, the listed rows exhaust the NameCert surface and cannot export quotient limits, selected universal cones, full completeness, choice, or bridge checking. □

Theorem 7730.3 (Cofiltered-limit cone-ledger non-escape)**.** *Every consumer of an accepted* $\mathsf{CofilteredLimit}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Cat}^{\uparrow},\ \mathsf{FilterBase}^{\uparrow},\ \mathsf{CofinalFilter}^{\uparrow},\ \mathsf{Limit}^{\uparrow} \longmapsto I, D, K, T, C, H, R, P, N$$

before any cofiltered-cone limit claim is public. The route exposes the inspected index, diagram, cone components, transition and cofinality data, cone-compatibility ledger, componentwise transport, continuation replay, provenance, and local naming; it does not expose quotient limits, selected universal cones, full completeness, choice, or bridge checking.

Proof. Start with a consumer of the accepted packet. Carrier admission exposes the category diagram row D and the cone row K only after the finite cofiltered index row I has been displayed.

Next read the transition row T. Its common-refinement requests pass through the displayed filter-base and cofinal-filter routes, so the consumer cannot replace the finite cofiltered data by an ambient directed set or a hidden filter quotient.

Finally replay the cone ledger C through R and transport the same rows through H. The provenance and local naming rows P, N record the same dependency spine through $\mathsf{Cat}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CofinalFilter}^{\uparrow}$, and $\mathsf{Limit}^{\uparrow}$. Because no other coordinate is present, quotient limits, selected universal cones, full completeness, choice, and bridge checking cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{CofilteredLimit}^{\uparrow}$ read must expose a finite cofiltered index row, a CategoryUp diagram row, cone components, transition and cofinality rows through FilterBaseUp and CofinalFilterUp, a cone-compatibility ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: category, filterbase, cofinalfilter, limit: $\mathsf{CofilteredLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite cofiltered index row, diagram row,

cone-component row, transition and cofinality row, cone-compatibility ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofilteredLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{CofilteredLimit}^{\uparrow}$ packet is generated from a finite cofiltered index row, a CategoryUp diagram row, cone components, transition and cofinality rows through FilterBaseUp and CofinalFilterUp, a cone-compatibility ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofilteredLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7730.1, Definition 7730.2, and Definition 7730.3 seed the finite cofiltered-cone route over CategoryUp, FilterBaseUp, CofinalFilterUp, LimitUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofilteredLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient limits, selected universal cones, full completeness, choice, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cofiltered-index admission, diagram-row admission, cone-component exposure, transition cofinality, cone-compatibility ledger coverage, transport, replay, provenance, and local naming for BEDC.Derived.CofilteredLimitUp.

Chapter 7731

A Concrete Naming Certificate for RegularCauchyMinmaxLattice$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RegularCauchyMinmaxLattice$^{\uparrow}$ packet records finite min–max lattice operations on displayed regular Cauchy rational names before any completed-real order principle is exposed. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located/order L10 sibling of chapter 1681. The packet names the pointwise meet and join surface for two displayed regular Cauchy names; it does not assert trichotomy, quotient stream equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 7731.1 (Regular Cauchy minmax lattice carrier)**.** *A* RegularCauchyMinmaxLattice$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, W, D, O, m, j, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right* RegSeqRat$^{\uparrow}$ *regular rational name rows.*
2. *W is the shared* StreamName$^{\uparrow}$ *finite-window ledger used to inspect both names at the requested dyadic radius.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row recording the comparison radius, slack, and rational cells consumed by the lattice read.*
4. *O is the* RegularCauchyLocatedOrder$^{\uparrow}$ *sibling row that supplies the located comparison boundary for the same windows.*
5. *m and j are the displayed meet and join rows, respectively naming the regular Cauchy readbacks for $\min(L, R)$ and $\max(L, R)$ over the inspected window.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after L, R, W, D, O, m, j have been displayed.*
7. *H records componentwise* hsame *transport for the two source names, the shared window, dyadic tolerance, located-order row, meet row, join row, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each lattice request through $L, R, W, D, O, m, j, E, H, P, N$.*
9. *P is the* Pkg *provenance over* RegularCauchyMinmaxLattice$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyLocatedOrder$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host lattice on completed reals, quotient stream equality, trichotomy, selected representatives, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$*.*

Theorem 7731.2 (Regular Cauchy minmax lattice NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}$ *carrier*

$$M = (L, R, W, D, O, m, j, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}}$ *surface consists exactly of carrier admission for the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, shared* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance data, the located-order sibling row, meet and join readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming; comparison visibility requiring every meet or join read to inspect O after W, D; and nonescape excluding trichotomy, host real lattice operations, quotient stream equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7731.1. The analytic rows are L, R, W, D, O, m, j, E, while H, C, P, N transport, replay, source, and name the same finite packet.

Read the comparison route in order. A lattice request first inspects the two regular rational names L, R, then aligns them over the common finite window W, then consumes the dyadic tolerance row D, and only then reads the located-order sibling row O.

The meet and join rows are downstream of that comparison route. The row m records the lower regular Cauchy readback and the row j records the upper regular Cauchy readback, both over the same windows and tolerance data. Componentwise transport through H and replay through C preserve that shared input surface.

Since the carrier has no coordinate for trichotomy, quotient stream equality, selected limits, countable choice, host lattice operations on completed reals, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7731.3 (Regular Cauchy minmax lattice RegSeqRat stability)**.** *For every accepted* $\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}$ *packet $M = (L, R, W, D, O, m, j, E, H, C, P, N)$, the meet row m and join row j remain* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks over the same finite windows W and dyadic tolerance row D as the source rows L and R. Every* $\mathsf{Real}^{\uparrow}$ *consumer reaches E only after the route*

$$(L, R) \longmapsto W \longmapsto D \longmapsto O \longmapsto (m, j)$$

has been displayed.

Proof. Start with the NameCert obligations of Definition 7731.2. They force every lattice read to expose the source regular rational names, the shared stream window, and the dyadic tolerance row before either operation row is available.

Use the located-order sibling row O as the comparison boundary. It is read over the same W, D surface, so the lower and upper operation rows cannot switch to a different observation schedule or a different tolerance ledger.

The rows m and j are therefore regular rational readbacks attached to the displayed input surface. The structural rows H, C, P, N preserve and name that attachment, and the terminal seal E is downstream of it. Thus every Real-facing read receives the stated finite route and no extra completeness or quotient datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName window, DyadicRatCore tolerance data, a RegularCauchyLocatedOrder sibling row, meet and join RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7731.1, Definition 7731.2, and Definition 7731.3 seed the finite min–max lattice-operation surface over regular Cauchy rational names and the located/order L10 sibling.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinmaxLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, host real lattice operations, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison visibility, meet and join RegSeqRat readback stability, Real-seal handoff, transport, replay, provenance, and local naming for the displayed min–max lattice packet.

Chapter 7732

A Concrete Naming Certificate for $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and binary interval bisection, the $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ packet records the finite route by which a regular rational name is inspected through dyadic half-windows before the terminal real-name seal is read. It sits after the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the alternating route of chapter 414, and the terminal seal of chapter 38. The packet names a finite subdivision certificate; it does not identify quotient real names, select a host limit, decide trichotomy, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7732.1 (RegularCauchyBinarySubdivision carrier)**.** *A* $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, R, I, L, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose entries list the inspected binary subdivision stages.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the midpoint, lower half, upper half, and requested radius for each inspected stage.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic ledger.*
4. *I is the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *parity row that records how the left and right half-window reads are interleaved.*
5. *L is the left half-window row, read only through S, D, R, I.*
6. *U is the upper half-window row, read only through S, D, R, I.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after L and U have been displayed as half-window evidence for the same regular name.*
8. *H records componentwise* hsame *transport for windows, dyadic ledger, regular readback, interleaving, half-window rows, seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a binary-subdivision request through S, D, R, I, L, U, E.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, D, R, I, L, U, E, H, C, P, N$. There is no coordinate for quotient reals, selected limits, trichotomy, countable-choice schedules, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7732.2 (RegularCauchyBinarySubdivision NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ *carrier*

$$K = (S, D, R, I, L, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite windows, dyadic tolerance ledger, regular rational readback, parity interleaving row, left and upper half-window rows, terminal real seal, componentwise transport, displayed replay, package provenance, and local naming. A consumer may read binary subdivision only through the route $S, D, R, I, L, U, E, H, C, P, N$.*

Proof. Project the carrier of Definition 7732.1. The visible rows are precisely $S, D, R, I, L, U, E, H, C, P, N$.

The rows S, D, R, I display the inspected stream windows, dyadic midpoint accounting, regular rational readback, and parity route. The half-window rows L and U are downstream of that joint surface, so they cannot be detached from the same regular Cauchy data.

The terminal seal E and the support rows H, C, P, N preserve the same finite route. Since the carrier has no quotient, selected-limit, trichotomy, choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7732.3 (RegularCauchyBinarySubdivision window halving)**.** *Let $K = (S, D, R, I, L, U, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ *carrier. Every binary-window request from K first reads the* $\mathsf{StreamName}^{\uparrow}$ *row S, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and then the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *parity row I before either half-window row L or U is public. Thus each half-window read is tied to the same displayed regular Cauchy name and the same dyadic subdivision ledger.*

Proof. Start with the continuation row C. Its displayed route sends a binary-subdivision request through the finite window row S and the dyadic ledger D before any half-window row can be inspected.

Next read the regular rational row R and the interleaving row I. They bind the left and upper half-window reads to the same regular name and to the same parity schedule.

The rows L and U are therefore not independent interval choices. They are the two displayed halves of the same finite subdivision route, transported and replayed only through H, C, P, N. □

Theorem 7732.4 (RegularCauchyBinarySubdivision Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ *carrier factors through*

$$S, D, R, I, L, U \longmapsto E \longmapsto H, C, P, N.$$

The handoff exports the terminal seal only after the finite StreamName windows, dyadic tolerance ledger, regular rational readback, interleaving route, and both half-window rows have been displayed. It exports no quotient real, selected limit, trichotomy decision, host real equality, countable-choice schedule, or ambient completeness theorem.

Proof. Apply Definition 7732.3. It forces each half-window request through S, D, R, I before L or U is public.

The terminal row E is downstream of both half-window rows. A Real-facing consumer therefore receives E only after the displayed binary subdivision route has been replayed.

Finally inspect the support rows. Componentwise transport through H, replay through C, provenance through P, and local naming through N carry the same finite packet and add no quotient, selected-limit, trichotomy, host-equality, choice, or ambient-completeness coordinate. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ read is refuted if a $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal without displaying StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, RegularCauchyInterleaving parity data, both half-window rows, transport, replay, provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, regularcauchyinterleaving, real: none alone is bijective with $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ because the packet jointly records windows, dyadic halving, regular readback, interleaving, two half-window rows, terminal sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}$ packet is generated from finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RegularCauchyInterleaving parity row, left and upper half-window rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7732.1, Definition 7732.2, Definition 7732.3, and Definition 7732.4 seed the finite binary-subdivision route from regular Cauchy windows through dyadic halving and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient reals, selected limits, trichotomy, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, binary window coverage, dyadic halving exactness, RegSeqRat readback, interleaving parity, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBinarySubdivisionUp.

Chapter 7733

A Concrete Naming Certificate for $\mathsf{ContextFreeLanguage}^{\uparrow}$

Following Hopcroft–Ullman and standard formal-language theory, the $\mathsf{ContextFreeLanguage}^{\uparrow}$ packet names a language by finite grammar and derivation data before any recognizer, pumping, or membership consumer reads a host language set. It sits beside the regular-language continuation surface of chapter 7361, the finite-control surface of chapter 2364, the stack-recognizer surface of chapter 6018, and the derivation-window certificate of chapter 2298. The packet records the grammar, derivation tree, yield, and classifier rows that those consumers may inspect; it is not an ambient set of words, a host-language equality principle, or a semantic closure theorem.

Definition 7733.1 (Context-free language carrier)**.** *A* $\mathsf{ContextFreeLanguage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, V, \Sigma, P, D, Y, M, H, C, Q, N)$$

with the following displayed rows:

1. *G is the grammar row naming the start symbol and the finite context-free grammar surface.*
2. *V is the finite nonterminal alphabet row.*
3. *Σ is the finite terminal alphabet row.*
4. *P is the production ledger, listing every admitted production as a finite row over V and Σ.*
5. *D is the derivation-tree row, recording the finite tree whose internal nodes read productions from P.*
6. *Y is the yield stream row, recording the terminal frontier obtained from D.*
7. *M is the membership classifier row, accepting a word only when it is the displayed yield of an admitted derivation tree.*
8. *H stores componentwise* hsame *transport for $G, V, \Sigma, P, D, Y, M, C, Q, N$.*
9. *C stores displayed* Cont *replay from grammar admission through productions, derivation tree, yield stream, and membership classifier.*
10. *Q is the* Pkg *provenance over* $\mathsf{ContextFreeLanguage}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{PushdownAutomaton}^{\uparrow}$, $\mathsf{ContextFreePumpingLemma}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContextFreeLanguage}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite parse forest, host language equality, unrestricted grammar quotient, choice of a canonical derivation, complement closure, determinization, or ambient semantic set membership.

Lean4 (statement) `BEDC.Derived.ContextFreeLanguageUp`

Theorem 7733.2 (Context-free language NameCert obligations)**.** *For every accepted* $\mathsf{ContextFreeLanguage}^{\uparrow}$ *carrier* $L = (G, V, \Sigma, P, D, Y, M, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContextFreeLanguage}^{\uparrow}}$ *surface consists exactly of grammar admission through* G, V, Σ, P*, derivation stability through* D *and* H*, yield exactness through* Y*, pumping-boundary nonescape through the* $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ *route, classifier transport through* M*, displayed replay through* C*, provenance through* Q*, and local naming through* N*.*

No accepted packet exports host language equality, an infinite parse forest, a grammar quotient, a selected canonical derivation, complement closure, determinization, or an ambient semantic language set.

Proof. Project the carrier of Definition 7733.1. The grammar, nonterminal, terminal, and production rows G, V, Σ, P are the only rows that admit grammar syntax.

The derivation tree and yield rows are downstream of that syntax. Componentwise transport through H preserves D and Y, and replay through C records the route from productions to the derivation tree and then to the terminal yield.

The membership classifier M reads only the displayed yield of an admitted derivation tree. The $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ sibling may inspect finite derivation-window data, but it receives no infinite parse forest or semantic language quotient.

Finally Q and N preserve provenance and local naming for the same finite packet. Since none of the excluded objects is a carrier coordinate, no local NameCert obligation can export it. □

Theorem 7733.3 (Context-free language pushdown handoff)**.** *Every* $\mathsf{PushdownAutomaton}^{\uparrow}$ *consumer that accepts a language-membership row from an accepted* $\mathsf{ContextFreeLanguage}^{\uparrow}$ *packet must first read the grammar row* G*, the nonterminal row* V*, the terminal row* Σ*, the production ledger* P*, the derivation-tree row* D*, the yield row* Y*, and the classifier row* M*. The handoff exports only the displayed grammar-and-derivation surface and cannot replace it by a host language set, a hidden stack oracle, a canonical derivation selector, or grammar quotient equality.*

Proof. Start with the carrier projection. The only membership row is M, and it is listed after the grammar, production, derivation, and yield rows.

Apply Definition 7733.2. Classifier transport is bound to the same finite rows G, V, Σ, P, D, Y, M and to the componentwise hsame transport in H.

A pushdown consumer therefore reads the grammar and derivation surface before accepting a language-membership row. The continuation row C records that replay order, while Q and N preserve provenance and naming without adding a second membership source.

The excluded objects are absent from the carrier. Hence the pushdown handoff supplies a finite grammar-and-derivation interface rather than a host language set, stack oracle, derivation selector, or quotient equality principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContextFreeLanguage}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContextFreeLanguage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7733.1, Definition 7733.2, and Definition 7733.3 seed the finite grammar, derivation, yield, classifier, pushdown-handoff, transport, replay, provenance, and local NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContextFreeLanguage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host language equality, infinite parse forests, grammar quotients, canonical derivation selection, complement closure, determinization, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked grammar admission, derivation stability, yield exactness, pumping-boundary nonescape, classifier transport, pushdown handoff, replay, provenance, and local naming for BEDC.Derived.ContextFreeLanguageUp.

Constructive story: A ContextFreeLanguage$^{\uparrow}$ packet is generated from a context-free grammar row, finite nonterminal and terminal alphabet rows, a production ledger, derivation tree, yield stream, membership classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7734

A Concrete Naming Certificate for $\mathsf{RationalCut}^{\uparrow}$

Following Dedekind and Bishop–Bridges constructive analysis, the $\mathsf{RationalCut}^{\uparrow}$ packet records finite located rational-cut rows used by the L10 Real and Completion route before any completed-real construction is read. It sits beside the rational arithmetic carrier of chapter 37, the dyadic tolerance core of chapter 291, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is finite located cut data for boundary and comparison consumers; it is not a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$.

Definition 7734.1 (Rational cut carrier)**.** *A* $\mathsf{RationalCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L, U, D, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *L is the lower rational row, recording finite rational witnesses below the inspected boundary.*
2. *U is the upper rational row, recording finite rational witnesses above the inspected boundary.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that names the dyadic slack and comparison radius used by the located request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which a consumer inspects the lower and upper rational observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the rational rows and the dyadic tolerance row are visible.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after L, U, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the lower row, upper row, dyadic tolerance, stream window, readback, Real seal, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from each located cut request through the same finite rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RationalCut}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through those displayed rows and their transports. It has no coordinate for quotient cut equality, trichotomy, selected real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7734.2 (Rational cut NameCert obligations)**.** *For every accepted* $\mathsf{RationalCut}^{\uparrow}$ *carrier* $C = (L, U, D, S, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{Rat}^{\uparrow}$ *rows, dyadic tolerance exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every located cut consumer must read* L, U, D, S *before it can read* R *or* E*, and the packet exports no Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, quotient cut equality, selected representative, trichotomy, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted carrier of Definition 7734.1. Its analytic rows are exactly the lower rational row, upper rational row, dyadic tolerance row, finite stream window, regular rational readback, and terminal Real seal.

Read the finite route in order. A located cut request first exposes the rational rows L, U, then the dyadic tolerance row D, and then the finite observation window S. Only after those rows are visible can the regular rational readback R and the Real seal E be consumed.

The structural rows H, K, P, N do not add a completed cut. They transport, replay, source, and name the same finite rows. Since the carrier contains no quotient equality, selected representative, trichotomy, choice, or completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 7734.3 (Rational cut L10 handoff boundary)**.** *Every L10 Real and Completion consumer of an accepted* $\mathsf{RationalCut}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Rat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The route supplies finite located rational-cut rows for comparison, regular-Cauchy readback, and Real-seal boundary arguments. It does not assert that $\mathsf{Real}^{\uparrow}$ *is constructed as a Dedekind cut, and it does not import quotient cut equality, selected limits, trichotomy, or ambient completeness.*

Proof. Start with the rational rows and the dyadic tolerance ledger. They are finite rows over $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$, so a consumer receives only displayed rational cells and slack data.

Next read the stream and regular rational rows. The finite $\mathsf{StreamName}^{\uparrow}$ row schedules the inspected cells, and the $\mathsf{RegSeqRat}^{\uparrow}$ row gives the readback boundary used by completion-facing consumers.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal and the structural rows. The Real seal is downstream of the finite cut data, while H, K, P, N preserve and name the same route. Thus the packet is cut-facing finite data inside L10, not a separate construction of $\mathsf{Real}^{\uparrow}$ by Dedekind cuts. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCut}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCut}^{\uparrow}$ packet is generated from lower and upper Rat rows, DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalCut}^{\uparrow}}$, seed)

Scope closed: Definition 7734.1, Definition 7734.2, and Definition 7734.3 seed the finite located rational-cut row used by L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, a Dedekind-cut construction of $\mathrm{Real}^{\uparrow}$, quotient cut equality, selected representatives, trichotomy, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper rational-row exposure, dyadic tolerance exactness, finite-window scheduling, regular rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalCutUp.

Chapter 7735

A Concrete Naming Certificate for $\mathsf{CompactCauchyNet}^{\uparrow}$

Following Bishop–Bridges compact metric arguments, the $\mathsf{CompactCauchyNet}^{\uparrow}$ packet records the finite-net and Cauchy-tail route used to turn compact-source inspection into a regular rational and Real-facing handoff. It sits over compact metric finite-net data, the Cauchy-net source surface, the dyadic tolerance core of chapter 291, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet names the finite compact-net Cauchy-tail route; it is not open-cover compactness, not a selected limit theorem, and not an ambient completeness principle.

Definition 7735.1 (Compact Cauchy-net carrier)**.** *A* $\mathsf{CompactCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, T, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the compact metric source row whose finite-net coverage is inspected by the consumer.*
2. *F is the finite-net row, recording the selected finite cover cells and mesh witnesses used by the compact argument.*
3. *T is the Cauchy-tail row, recording the finite tail-window evidence over the net.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare mesh size, tail radius, and readback precision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the inspected net and tail observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same finite-net and tail-window surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, F, T, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for compact source, finite net, Cauchy tail, dyadic tolerance, stream window, readback, Real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from compact-source inspection through finite-net coverage, tail-window evidence, tolerance rows, readback, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactCauchyNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed finite-net, tail-window, readback, seal, and structural rows. It has no coordinate for open-cover compactness, selected subnet, selected limit, quotient net equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7735.2 (Compact Cauchy-net NameCert obligations)**.** *For every accepted* $\mathsf{CompactCauchyNet}^{\uparrow}$ *carrier* $K = (M, F, T, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyNet}^{\uparrow}}$ *surface consists exactly of compact-source carrier exposure, finite-net coverage exposure, Cauchy-tail evidence, dyadic tolerance exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *scheduling,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7735.1. The compact source, finite-net row, Cauchy-tail row, dyadic tolerance row, stream window, regular rational readback, and Real seal are the only analytic rows.

The finite-net and Cauchy-tail rows are read before any regular rational or Real-facing handoff. The dyadic tolerance row binds the mesh and tail comparisons, while the stream window records exactly which finite observations have been inspected.

Transport through H, replay through C, provenance through P, and local naming through N preserve that same finite route. Since the carrier has no row for open-cover compactness, selected subnets, selected limits, quotient net equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7735.3 (Compact Cauchy-net finite-tail handoff)**.** *Every compact uniform-continuity or completion-facing consumer of an accepted* $\mathsf{CompactCauchyNet}^{\uparrow}$ *packet must read the finite-net row* F*, the Cauchy-tail row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is available. Thus the consumer route factors through finite-net coverage, Cauchy-tail evidence, regular rational readback, dyadic tolerance, and Real sealing as NameCert data.*

Proof. Start with compact-source exposure. The row M supplies the compact metric source only through the finite-net row F, so a consumer cannot replace the packet by an open-cover compactness theorem.

Read the Cauchy-tail and tolerance rows next. The tail evidence T and tolerance ledger D determine the finite comparison surface, and the stream window S records the observations that feed the readback row R.

The terminal seal E is downstream of the same route. Componentwise transport, continuation replay, provenance, and naming preserve the displayed rows, so no selected subnet, selected limit, quotient net equality, choice principle, or ambient completeness coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchyNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactCauchyNet}^{\uparrow}$ packet is generated from a compact metric source, finite-net coverage rows, Cauchy-tail evidence, DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCauchyNet}^{\uparrow}}$, seed)

Scope closed: Definition 7735.1, Definition 7735.2, and Definition 7735.3 seed the finite compact-net Cauchy-tail route used by compact uniform-continuity and completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCauchyNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, open-cover compactness, selected subnets, selected limits, quotient net equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, finite-net coverage, Cauchy-tail evidence, dyadic tolerance exactness, readback routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactCauchyNetUp.

Chapter 7736

A Concrete Naming Certificate for $\mathsf{MooreMachine}^{\uparrow}$

Following Moore's finite-state machine tradition, the $\mathsf{MooreMachine}^{\uparrow}$ packet records a deterministic automaton with output attached to states as finite BEDC NameCert data. It sits beside the Myhill–Nerode classifier of chapter 5729 and the regular-language continuation surface of chapter 7361. The packet names state replay and output readout for finite input traces; it does not introduce infinite streams, nondeterministic branching, minimization uniqueness, quotient language equality, or a host evaluator.

Definition 7736.1 (Moore-machine carrier)**.** *A* $\mathsf{MooreMachine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, I, O, q_0, \delta, \omega, W, Y, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite state row.*
2. *I is the finite input-alphabet row.*
3. *O is the finite output-alphabet row.*
4. *q_0 is the initial-state row.*
5. *δ is the deterministic transition replay row, reading a current state and an input symbol into the next displayed state.*
6. *ω is the output readout row, reading each displayed state into an output symbol.*
7. *W is the finite input-trace row.*
8. *Y is the finite output-trace row obtained by replaying W from q_0 through δ and reading each visited state through ω.*
9. *H records componentwise* hsame *transport for states, alphabets, initial state, transition replay, output readout, traces, continuation replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from an input-trace request through $q_0, \delta, \omega, W, Y$.*
11. *P is the* Pkg *provenance over* $\mathsf{MooreMachine}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, $\mathsf{RegularLanguageContinuation}^{\uparrow}$, *finite* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreMachine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows, their displayed transports, their continuation replay, package provenance, and local naming. It has no coordinate for nondeterministic choice, infinite input streams, quotient language equality, canonical minimization, or host evaluator soundness.

Theorem 7736.2 (Moore-machine NameCert obligations)**.** *For every accepted* $\mathsf{MooreMachine}^{\uparrow}$ *carrier* $M = (Q, I, O, q_0, \delta, \omega, W, Y, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreMachine}^{\uparrow}}$ *surface consists exactly of deterministic transition replay from* q_0 *along the finite input trace* W*, output trace stability through the state readout* ω*, comparison with the Myhill–Nerode state classifier of chapter 5729, and handoff to the regular-language continuation surface of chapter 7361. The packet exports finite state-output behavior only; it does not assert language quotient equality, minimal automaton uniqueness, infinite-stream acceptance, nondeterministic branching, or host evaluator soundness.*

Proof. Project the carrier of Definition 7736.1. The rows expose finite states, input and output alphabets, the initial state, deterministic transition replay, output readout, finite traces, transport, replay, provenance, and local naming.

Deterministic replay is supplied by δ and C. Starting from q_0, each displayed input symbol in W determines the next displayed state, so a trace consumer cannot branch outside the carrier rows.

Output trace stability is supplied by ω, Y, H. Transport preserves the same state readout under hsame, and the output trace Y is read only from the replayed finite state path.

Finally compare the displayed state behavior with the Myhill–Nerode classifier and hand it to the regular-language continuation surface. Those sibling routes consume the same finite state and continuation data, while no row adds quotient language equality, canonical minimization, infinite-stream acceptance, nondeterminism, or host evaluator soundness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreMachine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MooreMachine}^{\uparrow}$ packet is generated from finite state, input alphabet, output alphabet, initial-state, deterministic transition replay, output readout, finite trace, output trace, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MooreMachine}^{\uparrow}}$, seed)

Scope closed: Definition 7736.1 and Definition 7736.2 seed the finite Moore-machine surface over deterministic state replay, output readout, Myhill–Nerode comparison, regular-language continuation handoff, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MooreMachine}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close language quotient equality, minimal automaton uniqueness, infinite-stream acceptance, nondeterministic branching, host evaluator soundness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deterministic transition replay, output trace stability, Myhill–Nerode comparison, regular-language continuation handoff, transport, replay, provenance, and local naming for BEDC.Derived.MooreMachineUp.

Chapter 7737

A Concrete Naming Certificate for $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ packet records the finite lower-bound readback surface through which a positive real name exposes an inspectable radius before reciprocal, reciprocal-modulus, or completion-facing consumers use the positivity witness. It sits after the real-name seal of chapter 38, the located-positive-real witness of chapter 2029, the positive dyadic row of chapter 291, the finite stream window of chapter 126, and the regular rational readback of chapter 301. The packet is a displayed radius-readback boundary; it is not trichotomy, quotient real equality, a total inverse, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7737.1 (Real positive radius readback surface)**.** *A* $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R, L, D, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row for the positive real name whose radius is being read.*
2. *L is the* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *witness that exposes the positivity route for the same real-name seal.*
3. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row read from the located witness and displayed as the lower-bound budget.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the source name and radius comparison are inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that records the rational observations supporting the finite radius read.*
6. *H records componentwise* hsame *transport for the real seal, located-positive witness, dyadic radius, finite window, regular readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$L, D, W, Q \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RealPositiveRadiusReadback}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, L, D, W, Q, H, C, P, N$. It has no coordinate for trichotomy, quotient equality of real names, a total inverse operation, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7737.2 (Real positive radius readback obligations)**.** *For every accepted* $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ *carrier*

$$X = (R, L, D, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealPositiveRadiusReadback}^{\uparrow}}$ *surface consists exactly of real-seal admission through R, located-positive witness exposure through L, positive dyadic lower-bound readback through D, finite stream-window inspection through W, regular rational readback through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7737.1. The analytic rows are R, L, D, W, Q, and the structural rows are H, C, P, N.

The located witness L is the only route from the positive real name to a displayed radius. The dyadic row D, the finite window W, and the regular readback Q are downstream of that witness and therefore form the complete lower-bound readback surface.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for the excluded order, quotient, inverse, selected-limit, or completeness data, the listed rows exhaust the local NameCert obligations. □

Theorem 7737.3 (Real positive radius readback route)**.** *Every reciprocal-facing or completion-facing read of an accepted* $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedPositiveReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In particular, a consumer reaches the $\mathsf{Real}^{\uparrow}$ *seal only after the located-positive witness, positive dyadic radius, finite stream window, and regular rational readback have been displayed in the same packet.*

Proof. Inspect the route coordinates in Definition 7737.1. The $\mathsf{Real}^{\uparrow}$ row R is read through the located-positive witness L together with the radius, window, and readback rows D, W, Q.

Apply the obligation surface of Definition 7737.2. It binds L, D, W, Q to the same transport, replay, provenance, and local naming rows, so a downstream consumer cannot detach the positive-radius read from the source real-name seal.

Thus every accepted read reaches R through the displayed route. The packet contains no row for trichotomy, quotient real equality, total inversion, selected limits, or ambient completeness, so none of those endpoints is exported by the route. □

Falsifiable prediction: An accepted $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ packet cannot export the same positive-radius readback if it omits the LocatedPositiveReal witness, positive DyadicRatCore radius, StreamName finite window, RegSeqRat readback, transport, replay, provenance, or local naming row.

Independence witness: real, located-positive-real, dyadicratcore, streamname, and regseqrat: $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ is not bijective to any listed sibling because only this packet binds the positive real seal, located lower-bound witness, positive radius row, finite window, regular readback, transport, replay, provenance, and local NameCert surface as one readback boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{RealPositiveRadiusReadback}^{\uparrow}}$.

Constructive story: A $\mathsf{RealPositiveRadiusReadback}^{\uparrow}$ packet is generated from a RealUp seal, a LocatedPositiveRealUp witness, a positive DyadicRatCoreUp radius row, a StreamNameUp finite window, a RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealPositiveRadiusReadback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7737.1, Definition 7737.2, and Definition 7737.3 seed the finite lower-bound readback route for positive real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealPositiveRadiusReadback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, total inverse operations, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-positive witness exposure, positive dyadic radius exactness, finite stream-window inspection, regular readback, transport, replay, provenance, and local naming for BEDC.Derived.RealPositiveRadiusReadbackUp.

Chapter 7738

A Concrete Naming Certificate for $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy names and constructive real completion, the $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ packet records a shared finite modulus ledger for joining two regular Cauchy names before any terminal $\mathsf{Real}^{\uparrow}$ seal is read. It sits between the regular rational names of chapter 301, the stream windows of chapter 126, the dyadic tolerance core of chapter 291, the interleaving route of chapter 414, and the real-name boundary of chapter 38. The carrier is a finite ledger object, not a quotient equality of Cauchy names, an ambient completeness theorem, or a choice of global representative.

Definition 7738.1 (RegularCauchyCommonModulus carrier)**.** *A* $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, B, W, R, D, J, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows naming the regular Cauchy inputs whose comparison is being joined;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the prefixes of both names that are visible to the request;*
3. *R records the shared regular-readback ledger tying the two names to the same displayed window schedule;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row containing the dyadic budget used to compare both sides;*
5. *J is the common-modulus ledger stating that every accepted precision request is answered by the same finite window before the* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *H records componentwise* hsame *transport for A, B, W, R, D, J;*
7. *C records displayed* Cont *replay for the common-window request and its downstream handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, R, D, J, H, C, P, N$. It has no coordinate for quotient equality of Cauchy names, selected representatives, hidden countable choice, ambient completeness, or a direct $\mathsf{Real}^{\uparrow}$ *equality proof.*

Theorem 7738.2 (RegularCauchyCommonModulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ *carrier* $G = (A, B, W, R, D, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonMo}}$ *surface is exhausted by five obligation rows:*

1. *carrier exposure for the two regular Cauchy input rows, finite stream window, shared readback ledger, dyadic tolerance row, common-modulus ledger, transport, replay, provenance, and local naming;*
2. *regular-name admission for A and B through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows already visible in the carrier;*
3. *finite-window exposure through W, R, D, J, requiring both names to answer the same displayed dyadic precision request before a consumer may continue;*
4. *interleaving compatibility, allowing* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *consumers to read the common ledger as a finite join of the two visible regular names;*
5. *ledger refusal, excluding quotient equality of Cauchy names, selected representatives, hidden choice schedules, ambient completeness, and direct* $\mathsf{Real}^{\uparrow}$ *equality.*

No local NameCert obligation is stored outside those rows.

Proof. Project the displayed carrier. The only regular-name inputs are A and B, and the only shared comparison data are the finite rows W, R, D, J.

The regular-name admission row first reads A and B as $\mathsf{RegSeqRat}^{\uparrow}$ data. The window row W then displays the prefixes of both names that the request can inspect.

Next the shared readback and tolerance rows bind the two sides to the same dyadic request. The common-modulus ledger J records that the request is answered by one finite window schedule rather than by two unrelated searches.

Finally H, C, P, N transport, replay, source, and name exactly that finite route. Since no carrier coordinate supplies quotient equality, selected representatives, hidden choice, ambient completeness, or direct $\mathsf{Real}^{\uparrow}$ equality, the five rows exhaust the local surface. □

Theorem 7738.3 (RegularCauchyCommonModulus finite-window handoff)**.** *Every* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ *carrier must first expose the route*

$$A, B, W, R, D, J \longmapsto H, C, P, N.$$

Thus the consumer receives two regular rational names, a finite stream-window schedule, a shared readback ledger, a dyadic tolerance budget, a common-modulus row, componentwise transport, replay, provenance, and local naming before it can read any real-name seal. It receives no quotient equality of Cauchy names, selected representative, hidden choice schedule, ambient completeness theorem, or host real equality.

Proof. Begin with the carrier definition. Its Real-facing data are downstream of the two regular-name rows and the common finite-window ledger.

Use the NameCert obligations. The finite-window exposure row requires W, R, D, J to be displayed after A and B, and the interleaving-compatibility row allows only that displayed ledger to feed the $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ side.

The structural rows then preserve the same route. Transport through H, replay through C, provenance through P, and local naming through N add no quotient, selected representative,

choice schedule, completeness theorem, or host real equality, so every downstream seal read is forced through the finite chain above. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ read is refuted if a $\mathsf{RegularCauchyInterleaving}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches a seal without displaying both $\mathsf{RegSeqRat}^{\uparrow}$ rows, the $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, the common-modulus ledger, transport, replay, provenance, and local naming.

Independence witness: Nearest siblings: regseqrat, streamname, dyadicratcore, regular-cauchy-interleaving, real. None alone records two regular Cauchy names together with a single shared finite modulus ledger before the Real seal; each lacks at least one of the input-pair, common-window, tolerance, or handoff coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCommonModulus}^{\uparrow}$ packet is generated from two RegSeqRat rows, a finite StreamName window, a shared regular-readback ledger, a DyadicRatCore tolerance row, a common-modulus ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7738.1, Definition 7738.2, and Definition 7738.3 seed the shared finite-modulus ledger for two regular Cauchy names before Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy names, selected representatives, hidden choice schedules, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window exposure, interleaving compatibility, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCommonModulusUp.

Chapter 7739

A Concrete Naming Certificate for $\mathsf{DyadicTailComparison}^{\uparrow}$

Following Bishop and Bridges on explicit Cauchy moduli and rational tail estimates, the $\mathsf{DyadicTailComparison}^{\uparrow}$ packet records the finite dyadic layer below the regular Cauchy tail-comparison route. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the common-tail comparison packet of chapter 387, and the terminal real boundary of chapter 38. The packet stores the inspected dyadic windows and comparison budgets before any regular Cauchy or Real-facing consumer can read a tail comparison.

Definition 7739.1 (Dyadic tail-comparison carrier)**.** *A* $\mathsf{DyadicTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, W, E, T, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{StreamName}^{\uparrow}$ *tail windows whose dyadic endpoints are compared;*
2. *W is the shared late-window selector, recording the finite indices inspected for both tails;*
3. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger, including the requested dyadic radius, slack, and lower and upper endpoint rows;*
4. *T is the dyadic tail-comparison row, recording that the two inspected windows satisfy the displayed dyadic comparison at the requested precision;*
5. *R is the downstream readback row for* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyTailComparison}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers;*
6. *H records componentwise* hsame *transport for the two windows, dyadic ledger, comparison row, downstream readback, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay for the finite dyadic comparison route;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicTailComparison}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyTailComparison}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTailComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7739.2 (Dyadic tail-comparison NameCert obligations)**.** *For every accepted* $\mathsf{DyadicTailComparison}^{\uparrow}$ *carrier* $D = (X, Y, W, E, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicTailComparison}^{\uparrow}}$ *surface consists exactly of carrier admission, shared-window discipline, dyadic endpoint-ledger exactness, dyadic comparison readback, downstream nonescape to* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyTailComparison}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, componentwise transport, displayed replay, package provenance, local naming, and refusal of quotient streams, selected limits, host real equality, trichotomy, choice, and ambient real completeness.*

Proof. Project the carrier of Definition 7739.1. The rows $X, Y, W, E, T, R, H, C, P, N$ are the only available certificate coordinates, so carrier admission and local naming are read directly from the displayed packet.

The shared-window obligation is the row W. It forces both tails to be inspected at the same finite late window before the dyadic endpoint ledger is read.

The dyadic comparison obligations are E and T. The endpoint ledger E supplies the radius, slack, and endpoint rows, while T records the finite comparison of the two displayed windows at that precision.

The downstream and structural obligations are R, H, C, P, N. They transport, replay, source, and name the same finite dyadic comparison for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers without adding a quotient, selected limit, host equality, trichotomy, choice, or ambient completeness row. □

Theorem 7739.3 (Dyadic tail-comparison regular Cauchy route)**.** *Every regular Cauchy or real-completion consumer of an accepted* $\mathsf{DyadicTailComparison}^{\uparrow}$ *packet factors through the finite route*

$$X, Y, W \longmapsto E \longmapsto T \longmapsto R, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyTailComparison}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *read can consume the packet only after the common stream window and dyadic endpoint ledger have been displayed. The route supplies finite dyadic tail-comparison data, not quotient stream equality, selected limits, host real equality, trichotomy, choice, or ambient real completeness.*

Proof. Start with an accepted carrier $D = (X, Y, W, E, T, R, H, C, P, N)$. The source rows X and Y expose only stream windows, and W records the shared late-window selector for both tails.

The dyadic endpoint row E is read next. It supplies the finite radius and endpoint ledger consumed by the comparison row T, so no consumer can bypass the dyadic layer and jump directly to a regular Cauchy or Real comparison.

The readback row R is downstream of T and is the only row addressed to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. The structural rows H, C, P, N preserve and name that same route.

Since all rows have now been exhausted, the consumer has no coordinate from which to extract quotient stream equality, a selected limit, host real equality, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{DyadicTailComparison}^{\uparrow}$ read is refuted whenever it exports a regular Cauchy or Real-facing comparison without the shared StreamName window and DyadicRatCore endpoint ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTailComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTailComparison}^{\uparrow}$ packet is generated from two StreamName tail-window rows, a shared late-window selector, a DyadicRatCore endpoint ledger, a dyadic comparison row, downstream readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicTailComparison}^{\uparrow}}$, seed)

Scope closed: Definition 7739.1, Definition 7739.2, and Definition 7739.3 seed the finite dyadic tail-comparison layer below regular Cauchy and Real readback consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicTailComparison}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, trichotomy, choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, dyadic endpoint exactness, comparison readback, downstream nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicTailComparisonU

Chapter 7740

A Concrete Naming Certificate for $\mathsf{CauchyBicompletion}^{\uparrow}$

Following Isbell on injective envelopes and Nachbin on ordered uniform completions, the $\mathsf{CauchyBicompletion}^{\uparrow}$ packet records a finite two-sided regular Cauchy completion corridor. It sits over the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real boundary of chapter 38, the uniform two-sided corridor of chapter 3613, and the completion coherence laws of chapter 3834. The packet keeps paired Cauchy windows, comparison transport, replay, provenance, and local naming finite before any real-completion consumer can read a bicompletion boundary.

Definition 7740.1 (Cauchy bicompletion carrier)**.** *A* $\mathsf{CauchyBicompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, R, W, Q, E, U, M, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right* $\mathsf{StreamName}^{\uparrow}$ *Cauchy-window rows;*
2. *W is the paired finite-window row, recording the common precision request and the inspected left and right tails;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the paired windows;*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *route reached only after the paired readback row has been displayed;*
5. *U is the* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *comparison row, used only as the uniform two-sided corridor for the same finite pair;*
6. *M is the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *coherence row, recording unit, bind, and associativity readback for the displayed completion route;*
7. *H records componentwise* hsame *transport for the two sides, paired window, readback, Real route, uniform corridor, coherence row, replay, provenance, and local naming;*
8. *C records displayed* Cont *replay of the finite bicompletion route;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyBicompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCauchyBicompletio}$ $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBicompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected completion representatives, ambient host equality, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7740.2 (Cauchy bicompletion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBicompletion}^{\uparrow}$ *carrier* $B = (L, R, W, Q, E, U, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyBicompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, paired Cauchy-window discipline,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exactness,* $\mathsf{Real}^{\uparrow}$ *route nonescape,* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *corridor compatibility,* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *coherence, componentwise transport, displayed replay, package provenance, local naming, and refusal of quotient completions, selected representatives, ambient host equality, choice, and ambient real completeness.*

Proof. Project the carrier of Definition 7740.1. The rows $L, R, W, Q, E, U, M, H, C, P, N$ are the whole local certificate surface, so every obligation must be assigned to one of those coordinates.

The paired-window obligation is W over the two source rows L and R. It records the finite left and right tails inspected for the same precision request before readback can proceed.

The readback and completion-facing obligations are Q, E, U, M. The row Q supplies regular rational readback, E is the downstream real route, U records compatibility with the uniform two-sided corridor, and M records the completion monad-law coherence visible to consumers.

Finally H, C, P, N transport, replay, source, and name the same finite bicompletion packet. Since the carrier has no quotient-completion, selected-representative, host-equality, choice, or ambient-completeness coordinate, those objects cannot be exported by the local certificate. □

Theorem 7740.3 (Cauchy bicompletion real-completion route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or regular Cauchy completion consumer of an accepted* $\mathsf{CauchyBicompletion}^{\uparrow}$ *packet factors through the finite route*

$$L, R, W \longmapsto Q \longmapsto E, U, M \longmapsto H, C, P, N.$$

Thus the consumer receives paired Cauchy-window rows, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *route, the* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *corridor,* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *coherence, transport, replay, provenance, and local naming. It receives no quotient completion, selected representative, ambient host equality, choice, or ambient real completeness.*

Proof. Start from an accepted packet $B = (L, R, W, Q, E, U, M, H, C, P, N)$. The left and right source rows are only stream-window data, and the paired row W forces them to answer the same finite precision request.

The regular rational readback row Q is then the only way to expose the paired Cauchy windows as completion data. The terminal real row E remains downstream of Q, so a real-completion consumer cannot skip the finite readback.

The uniform corridor U and coherence row M are parallel checks on the same displayed route. They record two-sided uniform compatibility and completion monad-law behavior without adding a quotient completion or selected representative.

The structural rows H, C, P, N preserve, replay, source, and name the route. Once those rows are exhausted, the packet has no remaining coordinate for ambient host equality, choice, or ambient real completeness. □

Falsifiable prediction: A $\mathsf{CauchyBicompletion}^{\uparrow}$ read is refuted whenever it reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing boundary without paired Cauchy windows, RegSeqRat readback, the uniform corridor, and completion monad-law coherence.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBicompletion}^{\uparrow}}$**.**

Constructive story: A CauchyBicompletion$^{\uparrow}$ packet is generated from left and right StreamName Cauchy-window rows, a paired finite-window row, RegSeqRat readback, a RealUp route, a UniformCauchyBicompletion corridor, CauchyCompletionMonadLaw coherence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{CauchyBicompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7740.1, Definition 7740.2, and Definition 7740.3 seed the finite paired Cauchy-window bicompletion route into RegSeqRat and Real readback consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyBicompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, ambient host equality, choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-window discipline, RegSeqRat readback exactness, Real route nonescape, uniform corridor compatibility, completion monad-law coherence, transport, replay, provenance, and local naming for BEDC.Derived.CauchyBicompletionUp.

Chapter 7741

A Concrete Naming Certificate for $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$

Following Kelley and Bishop–Bridges compactness practice, the $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$ packet records the finite BEDC boundary at which an open-cover compactness reading is refused and replaced by compact-metric, totally-bounded, finite-net, radius-ledger, and uniform-modulus rows. It sits beside the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the finite $\mathsf{FiniteEpsilonNet}^{\uparrow}$ witness of chapter 1767, the compact-uniform route of chapter 388, the $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is an L10 boundary: it names the finite rows available to later compact-uniform and Heine–Cantor routes, while it does not name an arbitrary open cover, a selected subcover, or an ambient compactness theorem.

Definition 7741.1 (Compact open-cover boundary carrier)**.** *A* $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, O, N, L, U, H, C, P, Q)$$

with the following displayed rows:

1. *K is the compact-metric source row. It exposes only the* $\mathsf{CompactMetric}^{\uparrow}$ *source data and the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-probe discipline used by the route.*
2. *O is the finite cover-request row. It records the cover-shaped request as finite* BHist *data rather than as an arbitrary open-cover family.*
3. *N is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row, including the centers and finite membership checks visible to the compact-metric source.*
4. *L is the Lebesgue-number or radius ledger row. It records the rational radius lower-bound data consumed after the finite net has been displayed.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row used by compact-uniform and Heine–Cantor consumers after K, O, N, L have been replayed.*
6. *H records componentwise* hsame *transport for the compact source, finite cover request, finite net witness, radius ledger, uniform-modulus handoff, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from the cover-shaped request through the compact-metric source, finite net witness, radius ledger, and uniform-modulus handoff.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *Q is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenCoverBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, O, N, L, U, H, C, P, Q$. It has no coordinate for an arbitrary open-cover object, selected subcover, choice functional, quotient equality of cover families, host set equality, semantic compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7741.2 (Compact open-cover boundary NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$ *carrier $B = (K, O, N, L, U, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenCoverBoundary}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact-metric source, finite cover request, finite* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness, Lebesgue-number or radius ledger,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, transport, replay, provenance, and local name;*

2. *classifier transport to the common finite surface shared by* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

3. *stability of K, O, N, L, U under the componentwise* hsame *rows recorded in H;*

4. *replay exactness requiring each compact-uniform consumer to pass through K, O, N, L, U in that order;*

5. *refusal rows excluding arbitrary open covers, selected subcovers, choice functionals, quotient cover equality, host set equality, semantic compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 7741.1. The projection exposes the compact-metric source, finite cover request, finite epsilon-net witness, radius ledger, uniform-modulus handoff, transport, replay, provenance, and local naming rows.

Classifier transport is componentwise. The rows in H move the mathematical coordinates K, O, N, L, U and the administrative coordinates C, P, Q to the same finite comparison surface used by the cited compact, net, modulus, and Real chapters.

Replay exactness is read from C. A cover-shaped request reaches the compact source row, then the finite-net witness, then the radius ledger, and only then the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff.

The refusal clauses are exhausted by carrier inspection. Since the carrier has no coordinate for arbitrary open covers, selected subcovers, choice, quotient cover equality, host set equality, semantic compactness, or Real completeness, none of those data can be exported by the local certificate. □

Theorem 7741.3 (Compact open-cover boundary finite-net refusal)**.** *For every accepted* CompactOpenCoverBound *carrier $B = (K, O, N, L, U, H, C, P, Q)$, every compact-uniform or Heine–Cantor read that asks for open-cover compactness factors instead through the finite route*

$$K, O, N, L \longmapsto U.$$

Here K is the $\mathsf{CompactMetric}^{\uparrow}$ *source row, O is the finite cover-request row, N is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row, L is the Lebesgue-number or radius ledger, and U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. The route may be transported through H, replayed through C, packaged by P, and named by Q,*

but it exports no arbitrary open cover, selected subcover, host compactness object, choice schedule, quotient equality of covers, semantic continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a compact-uniform read of the accepted carrier. By Definition 7741.2, every such read must expose the carrier rows $K, O, N, L, U, H, C, P, Q$ and no row outside that finite surface.

The finite cover-request row O is not an open-cover object. It is admitted only as the cover-shaped request displayed inside the compact-metric source discipline, so the next available witness row is the finite $\mathsf{FiniteEpsilonNet}^{\uparrow}$ coordinate N.

The radius ledger L records the Lebesgue-number or rational-radius lower bound used after the finite net is visible. The replay row C then sends this finite radius package to the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff U.

Transport, provenance, and local naming do not add analytic content. The rows H, P, Q preserve and record the same finite route rather than providing a selected subcover, host compactness theorem, semantic continuity principle, or Real completeness row.

Thus the accepted open-cover-shaped request is refused as a general open-cover compactness read and is replaced by the displayed finite-net route $K, O, N, L \longmapsto U$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenCoverBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenCoverBoundary}^{\uparrow}$ packet is generated from a compact-metric source row, finite cover-request row, finite epsilon-net witness, Lebesgue-number or radius ledger, uniform-modulus handoff, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenCoverBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7741.1, Definition 7741.2, and Definition 7741.3 bind the finite compact-open-cover boundary and its finite-net refusal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenCoverBoundary}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CompactOpenCoverBoundaryUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, quotient cover equality, choice, semantic compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite-net exactness, radius-ledger exactness, uniform-modulus handoff, refusal rows, and TasteGate field faithfulness for $K, O, N, L, U, H, C, P, Q$.

Chapter 7742

A Concrete Naming Certificate for $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$

Following Bruck, Reich, and standard Hilbert projection theory, the $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ packet records the finite BEDC surface by which a closed convex Hilbert projection can be consumed as a firm nonexpansive operator. It sits after the projection endpoint of chapter 2356, the finite projection comparison route of chapter 7348, the proximal handoff of chapter 3713, the nonexpansive regular Cauchy route of chapter 2105, and the Chebyshev-center boundary of chapter 6623. The packet names the firm inequality ledger over $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows; it does not import a host projection map, a choice-selected nearest point, quotient Hilbert equality, or an ambient Hilbert projection theorem.

Definition 7742.1 (Firm nonexpansive projection carrier)**.** *A* $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, K, M, E, S, D, L, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, exposing the inner-product, norm, and induced metric ledgers needed for projection-facing comparisons.*
2. *K is the* $\mathsf{ConvexSet}^{\uparrow}$ *source row, recording the closed convex subset whose projection endpoint is being read.*
3. *M is the* $\mathsf{MetricProjection}^{\uparrow}$ *endpoint row, carrying the nearest-point endpoint only through the finite route of Definition 2356.1.*
4. *E is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used to replay the minimizing-window endpoint admission without adding an ambient completeness principle.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *and RegSeqRat distance-readback row, exposing squared-distance and inner-product comparisons only through displayed rational windows.*
6. *D is the inequality ledger for the finite firm-nonexpansive comparison, recording that every inspected pair of inputs and projected endpoints is read through the displayed relation*

$$\langle p_x - p_y, x - y\rangle \geq \|p_x - p_y\|^2.$$

7. *L records componentwise* hsame *transport for the Hilbert, convex, projection endpoint, completeness service, real readback, inequality ledger, replay, provenance, and local naming rows.*

8. *T records displayed* Cont *replay from projection endpoint reads through the real comparison ledger and the firm inequality ledger.*
9. *P is the* Pkg *provenance over* $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FirmNonexpansiveProjection}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only by these rows. It has no coordinate for a host projector, selected nearest point, quotient equality of Hilbert points, trichotomy on real comparisons, countable choice, or a global fixed-point theorem.

Lean4 ✓ `BEDC.Derived.FirmNonexpansiveProjectionUp.FirmNonexpansiveProjectionTasteGate single carrier alignment`

Theorem 7742.2 (FirmNonexpansiveProjection NameCert obligations)**.** *For an accepted* $\mathsf{FirmNonexpansiveProject}$ *carrier $Q = (H, K, M, E, S, D, L, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FirmNonexpansiveProjection}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Hilbert source, closed convex source, MetricProjection endpoint, CompleteMetric service, real distance readback, inequality ledger, transport, replay, provenance, and local naming rows;*
2. *endpoint dependency, requiring every projected point used by the firm inequality to be read first through the* $\mathsf{MetricProjection}^{\uparrow}$ *endpoint row M;*
3. *real readback exactness, requiring every squared-distance and inner-product comparison to pass through the shared* $\mathsf{Real}^{\uparrow}$ *and RegSeqRat row S;*
4. *firm inequality visibility, requiring the ledger D to display the finite comparison before any consumer treats the projection as nonexpansive;*
5. *nonescape from host projection maps, choice-selected nearest points, quotient Hilbert equality, trichotomy, countable choice, and ambient fixed-point principles.*

Proof. Project the accepted carrier. The rows H, K, M, E, S, D are the complete mathematical surface, while L, T, C, P, N transport, replay, package, and name the same displayed packet.

The endpoint dependency is inherited from Definition 2356.2. The projection endpoint is read through the Hilbert source, closed convex source, distance ledger, minimizing window, and endpoint row before the firm comparison ledger can inspect it.

The real readback and firm inequality rows are then finite. The row S supplies the displayed real and regular-rational comparisons, and D records the inspected inequality

$$\langle p_x - p_y, x - y\rangle \geq \|p_x - p_y\|^2.$$

Transport, replay, provenance, and local naming preserve this route. Since no carrier row stores a host projector, a selected nearest point, quotient Hilbert equality, trichotomy, countable choice, or a fixed-point theorem, those principles cannot be exported by the local certificate. □

Theorem 7742.3 (FirmNonexpansiveProjection MetricProjection handoff)**.** *Every consumer that uses an accepted* $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ *packet as a nonexpansive projection must first read the* $\mathsf{MetricProjection}^{\uparrow}$ *endpoint row M and the firm inequality ledger D. The consumer may then use the projection only through the displayed route*

$$H, K, M, E, S, D \longmapsto L, T, C, P, N.$$

It receives no host Hilbert projection map, no selected nearest-point operator, no quotient equality of projected points, no trichotomy decision for real comparisons, and no global fixed-point conclusion.

Proof. Start with the projected endpoint. By Definition 7742.1, endpoint data enter the packet only through the $\mathsf{MetricProjection}^{\uparrow}$ row M, and Definition 2356.3 keeps that row inside the finite projection route.

Next read the inequality ledger. The row D may compare two projected endpoints only after H, K, M, E, S are displayed, so a nonexpansive use cannot bypass the Hilbert source, closed convex source, complete metric replay, or real readback.

The structural rows L, T, C, P, N only transport, replay, source, and name the same finite data. Thus the handoff exports a firm nonexpansive projection certificate, not a host projector, selected nearest-point operator, quotient equality principle, trichotomy decision, or fixed-point theorem. □

Falsifiable prediction: Every accepted $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ consumer read factors through a HilbertUp source, ConvexSetUp closed convex source, MetricProjectionUp endpoint row, CompleteMetricUp replay service, RealUp and RegSeqRat distance readback, finite firm inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: metricprojection, finite-metric-projection, proximal-map, regular-cauchy-nonexpansive-map, chebyshev-center: $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a MetricProjection endpoint, a CompleteMetric replay service, a RealUp distance-readback row, and an explicit firm inequality ledger before any nonexpansive consumer read.

Closure status for $\mathsf{NameCert}_{\mathsf{FirmNonexpansiveProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{FirmNonexpansiveProjection}^{\uparrow}$ packet is generated from a Hilbert source, closed convex source, MetricProjection endpoint row, CompleteMetric replay service, RealUp and RegSeqRat distance readback, finite firm inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirmNonexpansiveProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7742.1, Definition 7742.2, and Definition 7742.3 seed the finite firm-projection certificate surface over Hilbert, closed convex, metric projection, complete metric, real readback, and inequality-ledger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FirmNonexpansiveProjection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FirmNonexpansiveProjectionUp`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host Hilbert projection map, selected nearest points, quotient equality of Hilbert points, trichotomy, countable choice, a Krasnoselskii–Mann convergence theorem, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint dependency, real readback exactness, firm inequality visibility, transport, replay, provenance, and local naming for BEDC.Derived.FirmNonexpansiveProjectionUp.

Chapter 7743

A Concrete Naming Certificate for $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real construction, the $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$ packet records the finite diagonal modulus row used when two regular rational names must be read together before a Real-facing seal is exposed. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ carriers of chapter 301, the finite schedules of chapter 126, the dyadic tolerance ledger of chapter 291, the generic limit route of chapter 1678, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite modulus selector, not quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.

Definition 7743.1 (Real Cauchy diagonal modulus carrier)**.** *A* $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *name rows whose finite regular-rational readbacks are being compared.*
2. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the visible prefixes of both names.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the requested radius, slack, and dyadic comparison budget.*
4. *Q is the diagonal selector row, requiring a single finite window to answer both regular names at the displayed tolerance.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, available only after A, B, S, D, Q have been replayed.*
6. *H records componentwise* hsame *transport for the two regular names, schedules, tolerance ledger, diagonal selector, seal handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal request through the schedules, tolerance ledger, selector, and seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDiagonalModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, S, D, Q, R, H, C, P, N$. It has no coordinate for quotient equality of streams, selected representatives, host real equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Theorem 7743.2 (Real Cauchy diagonal modulus NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$ *carrier* $M = (A, B, S, D, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDiagonalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, two* $\mathsf{RegSeqRat}^{\uparrow}$ *name admissions, paired* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger exactness, diagonal selector exactness,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal of quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 7743.1. The only input-name rows are A and B, and the only finite schedules visible to a request are stored in S.

Read the dyadic tolerance row D and the diagonal selector Q. The selector binds both regular names to the same displayed finite window before the downstream seal row R can be reached.

The rows H, C, P, N transport, replay, source, and name this same route. Since the carrier is exhausted by the displayed rows, it cannot export quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking. □

Theorem 7743.3 (Real Cauchy diagonal modulus window extraction)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Limit}^{\uparrow}$*, or regular Cauchy comparison consumer of an accepted* $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$ *carrier must factor through the finite route*

$$A, B \longmapsto S \longmapsto D \longmapsto Q \longmapsto R, H, C, P, N.$$

Thus the consumer receives two $\mathsf{RegSeqRat}^{\uparrow}$ *names, a paired* $\mathsf{StreamName}^{\uparrow}$ *schedule, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a diagonal selector, a* $\mathsf{Real}^{\uparrow}$ *seal handoff, transport, replay, provenance, and local naming. It receives no quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with a consumer of the accepted packet. The carrier exposes the two regular rational names A, B before any schedule or seal row can be read.

Replay the request through C. The paired schedule S supplies the finite windows, the tolerance ledger D supplies the dyadic budget, and the diagonal selector Q records the single window that answers both sides.

Only after that replay can the seal handoff R be consumed. Componentwise transport, provenance, and local naming through H, P, N preserve the same finite route and add no external equality, selector, choice, completeness, verification, or bridge coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyDiagonalModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCauchyDiagonalModulus}^{\uparrow}$ packet is generated from two RegSeqRat name rows, a paired StreamName schedule, a DyadicRatCore tolerance ledger, a diagonal selector, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyDiagonalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7743.1, Definition 7743.2, and Definition 7743.3 seed the finite diagonal-modulus packet used by Real and regular Cauchy comparison consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyDiagonalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired schedule exposure, tolerance-ledger exactness, diagonal selector exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyDiagonalMod

Chapter 7744

A Concrete Naming Certificate for MetaCICExtractionNormalizationSocket$^{\uparrow}$

The MetaCICExtractionNormalizationSocket$^{\uparrow}$ packet realizes the confluence audit boundary of chapter 9378 as finite socket data between extraction, candidate normalization, and residual confluence. It is consumed by chapter 4238 through its candidate-mediated strong-normalization socket and by chapter 765 through the retained residual-budget path. It does not assert a full parallel diamond. It records exactly the rows needed to keep the substitution-sensitive boundary visible while allowing checked fragments, candidate normal forms, and local joins to be handed to later MetaCIC consumers.

Definition 7744.1 (MetaCIC extraction-normalization socket carrier)**.** *A* MetaCICExtractionNormalizationSocket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, R, J, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the extraction row, recording the finite MetaCIC redex, source, target, and closedness window inspected by the socket.*
2. *K is the candidate-normalization row, recording the candidate normal form or bounded normalizing trace supplied by the visible extraction window.*
3. *R is the residual-substitution row, recording the exact substitution-compatibility boundary that the confluence audit map keeps conditional.*
4. *J is the local-join row, recording candidate common reducts or local joins that are available inside the displayed window.*
5. *O is the obstruction row, retaining the cases where the residual-substitution row or local-join row is absent.*
6. *H records componentwise* hsame *transport for extraction, candidate-normalization, residual-substitution, local-join, obstruction, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an extracted peak through candidate normalization, residual substitution, local join, and obstruction retention.*
8. *P is* Pkg *provenance over* MetaCICExtractionNormalizationSocket$^{\uparrow}$*, the MetaCIC confluence audit map,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, K, R, J, O, H, C, P, N$. It has no coordinate for a global Church–Rosser theorem, an unqualified parallel diamond, hidden substitution compatibility, quotient syntax equality, host normalization, Lean verification, or bridge checking.

Theorem 7744.2 (MetaCIC extraction-normalization socket NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}$ *carrier $E = (X, K, R, J, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}}$ *surface is exhausted by carrier admission, extraction-window exposure, candidate-normalization exposure, residual-substitution boundary retention, local-join exposure, obstruction retention, componentwise transport, displayed replay, package provenance, local naming, and refusal of full closed parallel diamond, global Church–Rosser, hidden substitution compatibility, quotient syntax equality, host normalization, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 7744.1. The extraction row X supplies the only visible source and redex data, while K records the candidate-normalization evidence available for that extraction.

The residual-substitution row R and local-join row J are separate obligations. A socket may expose a candidate normalization without discharging residual substitution, and it may expose a local join only when that row is displayed.

The obstruction row O retains the complementary cases. The structural rows H, C, P, N transport, replay, source, and name the same finite surface, so the packet cannot silently export a full parallel diamond, global confluence theorem, hidden substitution compatibility, quotient syntax equality, host normalization, Lean verification, or bridge checking. □

Theorem 7744.3 (MetaCIC extraction-normalization residual handoff)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}$ *packet receives only the finite route*

$$X \longmapsto K \longmapsto R, J, O \longmapsto H, C, P, N.$$

Consequently the packet can hand extraction and candidate-normalization evidence to the residual confluence frontier only with the residual-substitution row, local-join row, obstruction row, transport, replay, provenance, and local naming still attached. It cannot imply a full closed parallel diamond unless both R and J are displayed and separately discharged by the consumer.

Proof. Start with a confluence-audit consumer of the socket. The carrier exposes extraction data only through X and candidate-normalization data only through K.

Replay the socket through C. The replay reaches the residual-substitution row R, the local-join row J, and the obstruction row O before any downstream residual-confluence consumer can read the packet.

The sibling routes through chapter 4238 and chapter 765 are consumers of the displayed $X, K, R, J, O, H, C, P, N$ route, not extra discharge evidence.

Transport and provenance through H, P, N keep the same rows attached. Since the carrier has no additional global-diamond coordinate, the handoff is a finite residual socket rather than a closed parallel-diamond proof. □

Falsifiable prediction: A socket packet cannot imply a full closed parallel diamond unless the residual-substitution row and local-join row are both displayed.

Independence witness: metaciccriticalpath, metacicnormalizationaudit, metacicconfluenceauditpacket, metacic-parallel-diamond-frontier: none is bijective because the socket binds extraction,

candidate normalization, residual substitution, local join, obstruction, transport, replay, provenance, and NameCert rows in one carrier.

The Lean follow-up path is `lean4/BEDC/Derived/MetaCICExtractionNormalizationSocketUp/TasteGate.l` It must provide explicit Nontrivial MetaCICExtractionNormalizationSocket$^{\uparrow}$ and FieldFaithful MetaCICExtractionNorm instances preserving extraction, candidate-normalization, residual-substitution, local-join, obstruction, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}}$.

Constructive story: A MetaCICExtractionNormalizationSocket$^{\uparrow}$ packet is generated from extraction, candidate-normalization, residual-substitution, local-join, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7744.1, Definition 7744.2, and Definition 7744.3 seed the finite socket between extraction, normalization evidence, and residual confluence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICExtractionNormalizationSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a full closed parallel diamond, global Church–Rosser, hidden substitution compatibility, quotient syntax equality, host normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, extraction exposure, candidate-normalization exposure, residual-substitution retention, local-join exposure, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICExtractionNormalizationSocketUp.

Chapter 7745

A Concrete Naming Certificate for $\mathsf{RealIntervalHalving}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealIntervalHalving}^{\uparrow}$ packet records the real-line bisection step as finite BEDC NameCert data rather than as trichotomy, selected limits, quotient reals, or an ambient completeness theorem. It sits over the constructive real-line interface of chapter 4041, the one-step located bisection packet of chapter 4288, the finite bisection tree of chapter 6701, the dyadic tolerance core of chapter 291, the finite stream observations of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38.

Definition 7745.1 (RealIntervalHalving carrier)**.** *A* $\mathsf{RealIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, B, M, L, R_0, D, S, G, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the left and right endpoint rows of the input real interval;*
2. *M is the midpoint row exposed through dyadic arithmetic and finite regular-rational readback;*
3. *L is the* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *row recording the one-step located-halving decision at the displayed tolerance;*
4. *R_0 is the* $\mathsf{IntervalHalvingTree}^{\uparrow}$ *row recording the finite bisection-tree position reached by this step;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for the midpoint and side read;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the endpoint and midpoint observations;*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the dyadic endpoint windows, stream observations, and regular rational readback have been displayed;*
9. *H records componentwise* hsame *transport for the endpoint, midpoint, located-halving, tree, tolerance, stream, readback, Real-seal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay for the endpoint-window request, midpoint readback, located side row, tree row, Real seal, and structural rows;*
11. *P is the* Pkg *provenance over* $\mathsf{RealIntervalHalving}^{\uparrow}$, $\mathsf{ConstructiveRealLine}^{\uparrow}$, $\mathsf{LocatedIntervalHalving}^{\uparrow}$, $\mathsf{IntervalHalvingTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalHalving}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for trichotomy, a selected limit, quotient real equality, an unrecorded completeness argument, or a Real-facing interval conclusion that bypasses the dyadic endpoint windows.

Theorem 7745.2 (RealIntervalHalving NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalHalving}^{\uparrow}$ *carrier*

$$R = (A, B, M, L, R_0, D, S, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalHalving}^{\uparrow}}$ *surface is exhausted by five finite obligation rows:*

1. *carrier exposure for the two endpoint rows, midpoint row, located-halving row, interval-halving tree row, dyadic tolerance, stream window, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *endpoint-window exactness, requiring A, B, D, and S to display the finite dyadic windows used by the bisection request;*
3. *midpoint and readback compatibility, requiring M and G to be read from the same displayed dyadic and stream data;*
4. *located-tree compatibility, requiring L and R_0 to record the selected half only as a finite located-halving row and finite tree row;*
5. *Real-seal nonescape, excluding trichotomy, selected limits, quotient real equality, hidden completeness, and any interval conclusion before D, S, G, E are visible.*

No local NameCert obligation is stored outside those rows.

Proof. Project the accepted carrier. The only interval endpoints are A and B, and the only midpoint data are the displayed rows M, D, S, G.

The endpoint-window exactness row first fixes the dyadic tolerance D and the stream window S. The midpoint and readback row then reads M through the same window and records the regular rational handoff in G.

The located side and tree position are separate finite rows. The row L records the one-step located halving decision, while R_0 records the corresponding finite tree position; neither row supplies trichotomy or a selected real limit.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since no coordinate supplies quotient real equality, hidden completeness, or a bypass interval conclusion, the five rows exhaust the local NameCert surface. □

Theorem 7745.3 (RealIntervalHalving RegSeqRat route)**.** *Every Real-facing halving read of an accepted* $\mathsf{RealIntervalHalving}^{\uparrow}$ *carrier factors through dyadic endpoint windows,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal before any interval conclusion is available. In carrier coordinates this route is*

$$A, B, D, S \longmapsto M, G \longmapsto L, R_0 \longmapsto E, H, C, P, N.$$

Thus the consumer receives the endpoint windows, midpoint readback, located-halving row, interval-tree row, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it can read the retained interval.*

Proof. Use the NameCert obligations. Endpoint-window exactness makes A, B, D, S visible before any midpoint or side read.

The midpoint and readback compatibility row then forces M and G to be read from the same dyadic endpoint windows and stream observations. This is the only regular-rational route available to the carrier.

The located-tree compatibility row consumes that readback through L and R_0. The Real seal E is downstream of these finite rows, and H, C, P, N only preserve the same route by transport, replay, provenance, and local naming.

Therefore a Real-facing halving consumer cannot reach an interval conclusion before the dyadic endpoint windows, StreamName observations, RegSeqRat readback, and Real seal have all been displayed. □

Falsifiable prediction: Every accepted $\mathsf{RealIntervalHalving}^{\uparrow}$ Real-facing interval read exposes endpoint rows, a midpoint row, one LocatedIntervalHalving row, one IntervalHalvingTree row, one DyadicRatCore tolerance row, one StreamName finite-window row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the retained interval is public.

Independence witness: constructive-real-line, located-interval-halving, interval-halving-tree, dyadicratcore, streamname, regseqrat, real: $\mathsf{RealIntervalHalving}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries real endpoint windows, midpoint data, a located side row, a finite tree row, dyadic tolerance, stream observations, regular rational readback, Real seal, transport, replay, provenance, and local naming as one finite bisection certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalHalving}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalHalving}^{\uparrow}$ packet is generated from real endpoint rows, a midpoint row, LocatedIntervalHalving and IntervalHalvingTree rows, DyadicRatCore tolerance, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalHalving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7745.1, Definition 7745.2, and Definition 7745.3 seed the finite Bishop real-interval bisection route over endpoint windows, midpoint readback, located side rows, interval-tree rows, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalHalving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, selected limits, quotient reals, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.RealIntervalHalvingUp.

Chapter 7746

A Concrete Naming Certificate for $\mathsf{LocatedTotalBoundedness}^{\uparrow}$

Following Bishop-style constructive metric compactness, the $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ packet records total boundedness with located finite-net selection data rather than with open-cover compactness, a host compactness theorem, or a choice-selected net. It sits over the metric carrier of chapter 57, the total-boundedness packet of chapter 127, the finite ε-net interface of chapter 1767, the dyadic tolerance core of chapter 291, the stream-window naming surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. A consumer sees explicit finite centers, tolerance windows, and location rows; it does not receive an unbounded cover family or a hidden selector.

Definition 7746.1 (LocatedTotalBoundedness carrier)**.** *A* $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, T, F, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *point and distance row for the requested finite window.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *coverage row, naming the displayed tolerance and the finite family of candidate centers.*
3. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row whose entries list the actual finite net centers and their coverage ledger.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested radius, center distances, and coverage slack.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row recording which center and point windows are opened.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance and radius comparisons.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the distance comparisons after D, S, R have been replayed.*
8. *H records componentwise* hsame *transport for metric data, total-boundedness data, finite-net data, dyadic tolerances, stream windows, regular readback, Real sealing, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedTotalBoundedness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedTotalBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, F, D, S, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a host compactness theorem, an unrecorded center selector, quotient stream equality, countable choice, or ambient real completeness.

Theorem 7746.2 (LocatedTotalBoundedness NameCert obligations)**.** *For every accepted* LocatedTotalBoundednes *carrier*

$$L = (X, T, F, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedTotalBoundedness}^{\uparrow}}$ *surface is exhausted by carrier exposure, total-bounded coverage, finite-net center listing, dyadic tolerance exactness, stream-window observation, regular rational readback, Real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. The certificate refuses open-cover compactness, host compactness, hidden center selection, quotient stream equality, countable choice, and ambient completeness.*

Proof. Project the accepted carrier. The only ambient metric data are X, and the only total-bounded coverage data are the displayed rows T and F.

The dyadic and stream rows D and S fix the finite tolerance windows used to test the listed centers. The regular readback row R records those comparisons before the terminal real seal E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no open-cover, host-compactness, selector, quotient, countable-choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7746.3 (LocatedTotalBoundedness finite-net location route)**.** *Every compact-metric or completion-facing consumer of an accepted* $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ *carrier must read located finite-net data through the route*

$$X, T, F \longmapsto D, S \longmapsto R \longmapsto E, H, C, P, N.$$

Thus a finite-net location claim is available only after the Metric row, TotallyBounded row, FiniteEpsilonNet row, DyadicRatCore tolerance row, StreamName observation row, RegSeqRat readback row, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row have all been displayed.*

Proof. Start with the carrier of Definition 7746.1. The rows X, T, F expose the metric source, the total-bounded coverage request, and the finite list of net centers.

The rows D and S then determine the dyadic tolerances and finite observation windows used to locate a center for the request. The regular readback row R turns these finite comparisons into the rational data consumed by the downstream Real seal E.

The structural rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming. A compact-metric or completion-facing consumer therefore cannot replace the displayed finite-net route by open-cover compactness, host compactness, hidden selection, quotient stream equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ compact-metric or completion-facing read exposes a Metric row, a TotallyBounded coverage row, a FiniteEpsilonNet center row,

a DyadicRatCore tolerance row, a StreamName observation row, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it may use located finite-net data.

Independence witness: metric, totallybounded, finiteepsilonnet, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric source data, total-bounded coverage, located finite-net centers, dyadic tolerance comparison, stream observation, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one finite compactness-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedTotalBoundedness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedTotalBoundedness}^{\uparrow}$ packet is generated from a Metric row, TotallyBounded coverage row, FiniteEpsilonNet center row, DyadicRatCore tolerance row, StreamName observation row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedTotalBoundedness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7746.1, Definition 7746.2, and Definition 7746.3 seed the located finite-net route for compact-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedTotalBoundedness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host compactness, hidden center selection, quotient stream equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.LocatedTotalBoundedness

Chapter 7747

Regular Cauchy Uniform Tail NameCert Construction

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ packet records the finite uniform tail-window surface by which one displayed threshold controls every later precision read before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the real-name boundary of chapter 38, and the tail-estimate packet of chapter 1487. The packet names a finite uniform tail window for regular Cauchy names; it is not a quotient stream, selected limit, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7747.1 (Regular Cauchy uniform tail carrier)**.** *A* $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, M, W, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite family of precision requests whose common tail window is being certified;*
2. *M is the displayed modulus-threshold row, naming one index bound that serves every request in Q;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, listing exactly the stream observations at and beyond the threshold M that are consumed by the packet;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the dyadic comparison budgets attached to the requests in Q;*
5. *T is the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *row witnessing that the same threshold M controls all later windows in W;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the uniform tail window and dyadic ledger;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after Q, M, W, D, T, R have been displayed;*
8. *H records componentwise* hsame *transport for the request family, threshold, window, dyadic ledger, tail-estimate, regular readback, and real seal rows;*
9. *C records displayed* Cont *replay from each precision request in Q through the common threshold and downstream readback route;*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformTail}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformTail}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, M, W, D, T, R, E, H, C, P, N$. No coordinate stores quotient stream equality, a selected real representative, a selected limit, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7747.2 (Regular Cauchy uniform tail NameCert obligations)**.** *For every accepted* RegularCauchyUniformT *carrier $U = (Q, M, W, D, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformTail}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite request family, common threshold, stream tail window, dyadic tolerance ledger, tail-estimate handoff, regular rational readback, real seal, componentwise transport, replay, provenance, and local naming; classifier transport to the same finite request-and-window surface; stability of every row under the displayed* hsame *data; exactness requiring all real-facing reads to pass through Q, M, W, D, T, R, E in that order; and ledger refusal for quotient streams, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, and ambient real completeness.*

Proof. Project the carrier of Definition 7747.1. The projection exposes the finite request family Q, the common threshold M, the tail-window row W, the dyadic ledger D, the tail-estimate row T, the regular readback R, the real seal E, and the structural rows H, C, P, N.

Classifier transport is componentwise. The rows in H compare the request family, common threshold, tail window, dyadic ledger, tail estimate, readback, and seal on the same finite surface, while C, P, N only replay, source, and name that surface.

Exactness is read from the displayed route. A precision request first enters Q, uses the shared threshold M, reads only the displayed windows in W, consumes the dyadic budget D, reaches the tail-estimate row T, then the regular readback R, and only then the real seal E.

The refusal rows are exhausted by the carrier inventory. Since the carrier has no coordinate for quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, no local NameCert obligation can export those data. □

Theorem 7747.3 (Regular Cauchy uniform tail Real seal nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ *carrier factors through the finite route*

$$Q, M \longmapsto W \longmapsto D \longmapsto T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the package stays inside the displayed $\mathsf{Real}^{\uparrow}$ *seal route and does not assert ambient completeness, quotient streams, selected limits, countable choice, trichotomy, or open-cover compactness.*

Proof. Start with an accepted carrier $U = (Q, M, W, D, T, R, E, H, C, P, N)$. The rows Q and M supply the finite precision family and the common threshold; no real-facing read is available at this stage.

The stream and dyadic rows are next. The row W lists the inspected tail windows, and D records the dyadic comparison budgets tied to those windows, so a consumer cannot bypass the finite observation layer.

Apply the tail-estimate and regular readback rows. The row T certifies that the same threshold controls every later window in the displayed request family, and R converts that uniform tail evidence into the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by the seal.

Finally inspect E, H, C, P, N. The real seal E is downstream of the displayed route, and the structural rows only transport, replay, source, and name that route. Because the carrier inventory contains no completeness, quotient, selected-limit, choice, trichotomy, or compactness coordinate, the package cannot export such a principle. □

Falsifiable prediction: A $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ read is refuted whenever a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the real seal without displaying the finite precision family, common threshold, StreamName tail window, DyadicRatCore tolerance ledger, RegularCauchyTailEstimate row, RegSeqRat readback, componentwise transport, replay, provenance, and local naming.

Independence witness: regularcauchytailestimate, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ because the packet jointly records a finite request family, one shared tail threshold, the inspected StreamName windows, dyadic comparison budgets, tail-estimate handoff, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformTail}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniformTail}^{\uparrow}$ packet is generated from a finite precision-request family, one shared tail threshold, StreamName tail windows, DyadicRatCore tolerance budgets, a RegularCauchyTailEstimate row, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7747.1, Definition 7747.2, and Definition 7747.3 seed the finite uniform tail-window route from regular Cauchy tail control to Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-threshold coverage, stream-window exactness, dyadic tolerance handoff, tail-estimate handoff, regular-name readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyUniformTailUp.

Chapter 7748

A Concrete Naming Certificate for $\mathsf{BishopPremetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopPremetric}^{\uparrow}$ packet isolates the premetric interface used before a constructive metric space asserts separated point identity. It sits after the distance-comparison surface of chapter 5565, before the Real-facing metric presentation of chapter 3673, and beside the completion routes of chapter 1506, chapter 2114, and chapter 38. The packet records zero-distance evidence as displayed data, not as quotient equality, representative selection, host point equality, or a completed Real object.

Definition 7748.1 (Bishop premetric carrier)**.** *A* $\mathsf{BishopPremetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, D, Z, S, Q, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed point-window row for the source carrier.*
2. *D is the Bishop premetric distance row, exposing finite positive-radius comparisons between displayed point windows.*
3. *Z is the zero-distance evidence row, recording when all displayed radius requests along D are met.*
4. *S is the separation-boundary row, consumed by* $\mathsf{BishopMetricSpace}^{\uparrow}$ *only after Z has been displayed.*
5. *Q is the quotient-refusal row, retaining the fact that zero-distance evidence has not been collapsed into point equality.*
6. *M is the completion-facing row consumed by* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{LocatedMetricCompletion}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *only after X, D, Z, S, Q have been read.*
7. *H records componentwise* hsame *transport for point windows, premetric distances, zero-distance evidence, separation boundary, quotient refusal, completion-facing rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite distance request through zero-distance evidence and the separation boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopPremetric}^{\uparrow}$*,* $\mathsf{Premetric}^{\uparrow}$*,* $\mathsf{BishopMetricSpace}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{LocatedMetricCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopPremetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, D, Z, S, Q, M, H, C, P, N$. It has no coordinate for quotient point equality, selected representatives, host metric equality, completed points, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7748.2 (Bishop premetric NameCert obligations)**.** *For every accepted* $\mathsf{BishopPremetric}^{\uparrow}$ *carrier $B = (X, D, Z, S, Q, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopPremetric}^{\uparrow}}$ *surface consists exactly of source point-window admission, Bishop premetric distance exposure, zero-distance evidence, separation-boundary handoff, quotient-refusal retention, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Each obligation is read through the finite route*

$$X, D, Z \longrightarrow S, Q, M \longrightarrow H, C, P, N.$$

Proof. Project the carrier of Definition 7748.1. The source row X and distance row D are the only rows a consumer may inspect before asking whether two point windows have zero distance.

The zero-distance row Z records evidence from the displayed premetric comparisons. The separation row S is downstream of Z, while Q keeps the same evidence from being read as quotient equality.

The completion-facing row M can therefore expose only the finite premetric route X, D, Z, S, Q. Componentwise transport through H and replay through C preserve that order, and P, N record provenance and local naming.

Since quotient point equality, selected representatives, host metric equality, completed points, countable choice, trichotomy, and ambient Real completeness do not occur among the carrier coordinates, no NameCert obligation can be discharged outside the displayed finite route. □

Theorem 7748.3 (Bishop premetric Cauchy route)**.** *Every Cauchy, completion-facing, or Real-facing consumer of an accepted* $\mathsf{BishopPremetric}^{\uparrow}$ *packet reaches the premetric input only through*

$$\mathsf{Premetric}^{\uparrow}, \mathsf{BishopPremetric}^{\uparrow} \longmapsto \mathsf{BishopMetricSpace}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

In particular, the completion-facing row M exports finite zero-distance and separation-boundary data, not a selected Cauchy limit, quotient equality, host metric equality, or a detached completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the accepted carrier $B = (X, D, Z, S, Q, M, H, C, P, N)$. The row M is downstream of the point-window, distance, zero-distance, separation, and quotient-refusal rows.

Apply Definition 7748.2. The local certificate forces every Cauchy or completion-facing read to pass through X, D, Z, S, Q before the completion row can be inspected.

The downstream routes named by chapter 3673, chapter 1506, and chapter 38 may consume that displayed premetric boundary. They do not receive a completed point, selected limit, quotient equality, host metric equality, or ambient completeness principle from this packet. □

Theorem 7748.4 (Bishop premetric metric-separation boundary)**.** *Let $B = (X, D, Z, S, Q, M, H, C, P, N)$ be an accepted* $\mathsf{BishopPremetric}^{\uparrow}$ *packet. A* $\mathsf{BishopMetricSpace}^{\uparrow}$ *consumer can read metric separation from B only through the displayed row S, and S is available only after the zero-distance evidence Z has been read together with the quotient-refusal row Q. Thus the passage from premetric data to metric separation is a finite NameCert handoff, not an implicit quotient of the source points.*

Proof. Start from Definition 7748.1. The only row named as a separation boundary is S, and the only row that records refusal of quotient collapse is Q.

Use Definition 7748.2. It orders the local obligations as distance exposure, zero-distance evidence, separation handoff, quotient-refusal retention, and completion-facing nonescape.

Therefore a $\mathsf{BishopMetricSpace}^{\uparrow}$ consumer reaches metric separation only by reading D and Z first, then S with Q still visible. The structural rows H, C, P, N transport and name that same finite handoff; they do not add quotient equality, selected representatives, or host point equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopPremetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopPremetric}^{\uparrow}$ packet is generated from displayed point windows, Bishop premetric distance comparisons, zero-distance evidence, a separation-boundary handoff, quotient-refusal retention, a completion-facing row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopPremetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7748.1, Definition 7748.2, Definition 7748.3, and Definition 7748.4 seed the constructive boundary between premetric evidence, Bishop metric separation, completion-facing reads, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopPremetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient point equality, selected representatives, host metric equality, completed points, countable choice, trichotomy, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked source admission, distance exposure, zero-distance evidence, separation-boundary handoff, quotient-refusal retention, completion-facing nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopPremetricUp.

Chapter 7749

A Concrete Naming Certificate for $\mathsf{ContextSensitiveLanguage}^{\uparrow}$

Following Hopcroft–Ullman and the Chomsky hierarchy presentation of formal languages, the $\mathsf{ContextSensitiveLanguage}^{\uparrow}$ packet names accepted context-sensitive languages by finite noncontracting grammar rows and bounded acceptance ledgers before any consumer reads a host language set. It sits beside the accepted context-free language interface of chapter 7733, the noncontracting grammar carrier of chapter 6178, and the context-free grammar syntax surface of chapter 356. The packet records the accepted-language classifier induced by grammar production, length-nondecreasing derivation, and linear-bounded acceptance rows; it is not a host powerset of words, a grammar quotient, an unrestricted Turing-machine semantics, or an ambient decidability theorem.

Definition 7749.1 (Context-sensitive language carrier)**.** *A* $\mathsf{ContextSensitiveLanguage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, \Sigma, W, R, D, B, A, M, H, C, P, N)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextSensitiveGrammar}^{\uparrow}$ *row, exposing the noncontracting grammar packet of chapter 6178.*
2. *Σ is the finite terminal alphabet row and W is the candidate word row, both read through finite list data before acceptance is inspected.*
3. *R is the grammar-production ledger, listing each production used by the acceptance certificate as a finite row over the displayed grammar.*
4. *D is the length-nondecreasing derivation ledger. Each derivation step records the predecessor word, successor word, occurrence window, production row from R, and the displayed length comparison.*
5. *B is the linear-bounded acceptance route, recording that the inspected sentential windows remain inside the finite bound determined by the input word and the displayed grammar rows.*
6. *A is the accepted-word row, available only after G, Σ, W, R, D, B have been replayed.*
7. *M is the comparison row for* $\mathsf{ContextFreeLanguage}^{\uparrow}$ *and* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumers that request weaker grammar-side or language-side surfaces.*

8. *H records componentwise* hsame *transport for the grammar source, terminal alphabet, word row, production ledger, derivation ledger, bounded route, accepted-word row, comparison row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a language-membership request through $G, \Sigma, W, R, D, B, A, M, H, P, N$.*
10. *P is the* Pkg *provenance over* ContextSensitiveLanguage$^{\uparrow}$, ContextSensitiveGrammar$^{\uparrow}$, ContextFreeLanguage$^{\uparrow}$, ContextFreeGrammar$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, \Sigma, W, R, D, B, A, M, H, C, P, N$. It has no coordinate for a host language set, a selected infinite derivation, a grammar quotient, unrestricted machine computation, complement closure, or global decidability.

Theorem 7749.2 (Context-sensitive language NameCert obligations)**.** *For every accepted* ContextSensitiveLanguag *carrier*

$$L = (G, \Sigma, W, R, D, B, A, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}$ *surface consists exactly of source grammar admission through G, terminal and word admission through Σ, W, grammar-production admission through R, length-nondecreasing derivation replay through D, linear-bounded acceptance through B, accepted-word classification through A, sibling comparison through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No accepted packet exports host language equality, an unrestricted machine semantics, a grammar quotient, a selected infinite derivation, complement closure, or ambient decidability.

Proof. Project the carrier of Definition 7749.1. The source grammar row G, terminal alphabet row Σ, word row W, and production ledger R are the only rows that can admit grammar syntax for the accepted-language read.

The derivation and bounded-acceptance rows are downstream. The ledger D records the applied production and the length comparison for each displayed step, while B records that the inspected windows remain inside the linear bound attached to the input word and grammar rows.

The accepted-word classifier A is therefore available only after G, Σ, W, R, D, B have been displayed. The comparison row M may hand the finite surface to ContextFreeLanguage$^{\uparrow}$ or ContextFreeGrammar$^{\uparrow}$ consumers, but it cannot replace the local classifier by a host language set.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the refused objects are not carrier coordinates, no local NameCert obligation can export them. □

Theorem 7749.3 (Grammar acceptance scope)**.** *Every accepted-word read exported by a* ContextSensitiveLanguage$^{\uparrow}$ *packet factors through the* ContextSensitiveGrammar$^{\uparrow}$ *source row, the finite grammar-production ledger, the length-nondecreasing derivation ledger, and the linear-bounded acceptance route before any language-level consumer may inspect the accepted-word classifier. The handoff to* ContextFreeLanguage$^{\uparrow}$ *and* ContextFreeGrammar$^{\uparrow}$ *consumers preserves only that displayed grammar-and-acceptance route; it exports no host language set, unrestricted Turing-machine computation, quotient grammar equality, selected infinite derivation, or ambient decidability theorem.*

Proof. Start with the grammar source. By Definition 7749.1, every accepted-language packet reads G before it reads a production ledger or accepted-word classifier.

Apply Definition 7749.2. The production ledger R, derivation ledger D, and bounded route B are mandatory local obligations, so the accepted-word row A cannot be inspected until the noncontracting grammar route and length-nondecreasing derivation route have been replayed.

Now inspect the sibling comparison row M. It can expose the weaker context-free language or grammar-side surface only as a comparison of the displayed rows G, Σ, W, R, D, B, A; it has no coordinate for host language equality or a grammar quotient.

Transport and replay through H and C preserve the same order, while P and N record provenance and local naming. Thus every consumer receives the finite grammar acceptance scope stated in the theorem and no unrestricted machine or semantic language object. □

Falsifiable prediction: Every $\mathsf{ContextSensitiveLanguage}^{\uparrow}$ read factors through a $\mathsf{ContextSensitiveGrammar}^{\uparrow}$ source row, finite terminal and word rows, grammar-production ledger, length-nondecreasing derivation ledger, linear-bounded acceptance route, accepted-word classifier, sibling comparison row, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}$ rows.

Independence witness: contextfreelanguage, contextsensitivegrammar, contextfreegrammar: $\mathsf{ContextSensitiveLanguage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an accepted-language classifier, noncontracting grammar source, length-nondecreasing derivation ledger, and linear-bounded acceptance route, while the siblings separately record context-free acceptance, grammar production syntax, or context-free grammar derivation rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}$.

Constructive story: A $\mathsf{ContextSensitiveLanguage}^{\uparrow}$ packet is generated from a ContextSensitiveGrammarUp source row, finite terminal and word rows, grammar-production ledger, length-nondecreasing derivation ledger, linear-bounded acceptance route, accepted-word classifier, sibling comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7749.1, Definition 7749.2, and Definition 7749.3 seed the accepted context-sensitive language surface over grammar production, length-nondecreasing derivation, linear-bounded acceptance, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContextSensitiveLanguage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted Turing-machine semantics, host language equality, complement closure, grammar quotient equality, selected infinite derivations, or global decidability.

Upgrade path: Obligation closure requires checked source grammar admission, production-ledger exactness, derivation length monotonicity, linear-bounded acceptance replay, classifier transport, sibling comparison, provenance, and local naming for BEDC.Derived.ContextSensitiveLanguageUp.

Chapter 7750

A Concrete Naming Certificate for PoincareMirandaFinite$^{\uparrow}$

Following Poincare and Miranda, the PoincareMirandaFinite$^{\uparrow}$ packet records the finite cube-face sign data that a constructive Poincare–Miranda argument must display before any zero cell is read. It sits beside the finite Brouwer route of chapter 2030, the one-dimensional interval-return route of chapter 1437, and the continuous interval image handoff of chapter 1897. The packet names only a finite face-sign, subdivision, and readback certificate; it is not a host zero theorem, quotient equality of real names, selected root, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ completeness principle.

Definition 7750.1 (Poincare–Miranda finite carrier)**.** *A* PoincareMirandaFinite$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, F, S, D, I, L, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite cube row, carrying the displayed coordinate list, opposite face pairs, dyadic side windows, and the finite incidence table for the cube boundary.*
2. *F is the face-sign packet. For each displayed coordinate face it records the finite positive-side and negative-side sign rows read from the carried continuous map.*
3. *S is the boundary-sign stability row, requiring each face sign to remain stable under displayed* hsame *transport, dyadic refinement, and the finite window reads used by F.*
4. *D is the subdivision ledger, listing the dyadic cells, their face incidence, inherited sign rows, and the refinement route by which a sign conflict is localized.*
5. *I is the compact interval handoff row. It records that every coordinate edge and retained one-dimensional slice is read through the* CompactIntervalFixedPoint$^{\uparrow}$ *and* ContinuousIntervalImage$^{\uparrow}$ *surfaces before a Real-facing zero read is public.*
6. *L is the Brouwer-style labelled-cell row, routing the cube-face sign conflict through the finite labelled-subdivision discipline used by* BrouwerFixedPoint$^{\uparrow}$*.*
7. *B is the zero-cell refusal ledger. It records that any claimed zero cell must expose Q, F, S, D, I, L first and that every attempt to read a zero without those rows is outside the certificate.*
8. *W is the* StreamName$^{\uparrow}$ *finite-window row opened for cube coordinates, face probes, dyadic cells, interval slices, and zero-cell readback.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to those windows.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed candidate zero cell, available only after $Q, F, S, D, I, L, B, W, R$ have been replayed.*
11. *H records componentwise* hsame *transport for the cube row, face signs, stability row, subdivision ledger, interval handoff, labelled-cell row, refusal ledger, windows, readback, seal, replay, provenance, and local naming.*
12. *C records displayed* Cont *replay from a zero-cell request back through $E, R, W, B, L, I, D, S, F, Q$.*
13. *P is the* Pkg *provenance over* $\mathsf{PoincareMirandaFinite}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$, $\mathsf{ContinuousIntervalImage}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareMirandaFinite}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these finite coordinates. It has no coordinate for a selected host zero, an ambient cube theorem, quotient real equality, open-cover compactness, countable choice, Lean verification, or bridge checking.

Theorem 7750.2 (Poincare–Miranda finite NameCert obligations)**.** *For every accepted* $\mathsf{PoincareMirandaFinite}^{\uparrow}$ *carrier*

$$M = (Q, F, S, D, I, L, B, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareMirandaFinite}^{\uparrow}}$ *surface consists exactly of carrier admission, boundary-sign stability, subdivision-ledger exposure, compact interval handoff, Brouwer-style labelled-cell handoff, zero-cell refusal, finite-window readback, Real-seal nonescape, componentwise transport, displayed continuation replay, provenance, and local naming. In particular, a consumer may inspect a candidate zero cell only after the cube-face signs, stable boundary rows, dyadic subdivision ledger, interval slices, labelled-cell route, refusal ledger, stream windows, regular readback, and Real seal have all been displayed.*

Proof. Carrier admission is projection from Definition 7750.1. The visible packet lists the cube row Q, face-sign packet F, stability row S, subdivision ledger D, interval handoff I, labelled-cell row L, refusal ledger B, finite windows W, regular readback R, Real seal E, and the structural rows H, C, P, N.

Boundary-sign stability is read through F and S. The opposite-face signs are not semantic predicates on a host cube; they are displayed rows tied to the finite face incidence table in Q, and S records their preservation under the dyadic refinements and hsame transports used by the packet.

The subdivision obligation is the ledger D. It lists the dyadic cells and inherited face incidences before any sign conflict can be localized, while L routes the resulting labelled-cell read through the finite Brouwer-style surface rather than through an ambient fixed-point theorem.

The compact interval handoff is the row I. Coordinate edges and retained one-dimensional slices are exposed through the compact interval fixed-point and continuous interval image packets before their finite windows and regular rational readbacks are consumed.

Finally B, W, R, E, H, C, P, N enforce nonescape. The refusal ledger rejects any zero-cell read that bypasses the displayed rows; the window, readback, and seal rows expose the candidate as a finite Real-facing read; and the structural rows only transport, replay, source, and name the same packet. Thus no host zero selector, quotient equality, open-cover compactness, countable choice, Lean-side verification, or bridge datum is added. □

Theorem 7750.3 (Poincare–Miranda zero-cell refusal)**.** *Every Real-facing zero-cell read accepted by a* $\mathsf{PoincareMirandaFinite}^{\uparrow}$ *carrier factors through*

$$Q, F, S, D, I, L, B \longmapsto W, R, E, H, C, P, N.$$

The refusal ledger B is part of the route: a purported zero cell that bypasses the finite cube faces, boundary-sign stability, subdivision ledger, compact interval handoff, Brouwer-style labelled-cell row, finite windows, regular readback, or terminal $\mathsf{Real}^{\uparrow}$ *seal is not a read of this certificate.*

Proof. Begin with an accepted packet $M = (Q, F, S, D, I, L, B, W, R, E, H, C, P, N)$. The candidate zero-cell seal E is downstream of the finite-window and regular-readback rows W and R, so a consumer cannot inspect it as a free-standing Real object.

The readback rows are downstream of the geometric rows. The finite cube row Q supplies the face incidence table, F supplies the opposite-face sign rows, S preserves those signs under displayed refinement, D lists the dyadic cells, I exposes the compact interval slices, and L records the labelled-cell handoff.

The refusal ledger B binds those coordinates into a single public route. If a requested zero cell omits one of them, B classifies the request as outside the packet rather than repairing it with host topology or choice.

Transport, replay, provenance, and naming preserve the same route through H, C, P, N. Hence every accepted zero-cell read factors through the displayed finite packet and exports no selected host zero, quotient equality, open-cover compactness, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{PoincareMirandaFinite}^{\uparrow}$ zero-cell read displays a finite cube row, opposite face-sign rows, boundary-sign stability, a dyadic subdivision ledger, compact interval and continuous interval image handoff rows, a Brouwer-style labelled-cell row, a zero-cell refusal ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: brouwer-fixed-point, compactintervalfixedpoint, continuous-interval-image: $\mathsf{PoincareMirandaFinite}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries cube-face signs, boundary-sign stability, a dyadic subdivision ledger, one-dimensional interval handoffs, a labelled-cell handoff, a zero-cell refusal ledger, finite readback, and local NameCert routing as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareMirandaFinite}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareMirandaFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7750.1, Definition 7750.2, and Definition 7750.3 seed the finite cube-face sign route from Poincare–Miranda boundary data through subdivision, compact interval handoff, labelled-cell routing, refusal, finite readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareMirandaFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host zero existence, selected roots, quotient real equality, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-sign stability, subdivision-ledger exposure, compact interval handoff, labelled-cell routing, zero-cell refusal, finite readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PoincareMirandaFiniteUp.

Constructive story: A $\mathsf{PoincareMirandaFinite}^{\uparrow}$ packet is generated from a finite cube row, opposite face-sign rows, boundary-sign stability, a dyadic subdivision ledger, compact interval and continuous interval image handoffs, a Brouwer-style labelled-cell row, a zero-cell refusal ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7751

A Concrete Naming Certificate for $\mathsf{UltrafilterBoundary}^{\uparrow}$

Following Bourbaki and Kelley filter terminology, the $\mathsf{UltrafilterBoundary}^{\uparrow}$ packet records the finite boundary at which a filter or completion argument would have to leave BEDC's displayed rows in order to invoke a maximal filter. It sits beside the constructive refusal surface of chapter 6632 and the choice boundary of chapter 1872, and it is read by L10 completion routes such as chapter 3885, chapter 333, chapter 95, and chapter 126. The packet is only a finite refusal boundary; it is not an ultrafilter existence theorem, not a Boolean-prime-ideal theorem, and not a selected completion shortcut.

Definition 7751.1 (Ultrafilter boundary carrier)**.** *An* $\mathsf{UltrafilterBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, B, Q, L, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the inspected filter row, recording the finite base members and refinement requests exposed by a Cauchy-filter or completion-facing argument.*
2. *B is the finite Boolean-branch ledger, listing the finite subset or complement tests explicitly requested before a maximality claim would be needed.*
3. *Q is the quotient-refusal row, recording that no quotient Boolean algebra representative or host equality of filters is exported.*
4. *L is the L10 handoff row, tying any completion-facing read to* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *windows before the boundary is consumed.*
5. *R is the ultrafilter-refusal row. It records that the packet does not produce a maximal filter, an ultrafilter extension, a prime ideal, a countable-choice schedule, or a host-side selector.*
6. *H records componentwise* hsame *transport for F, B, Q, L, R, P, N.*
7. *C records displayed* Cont *replay from finite filter and completion rows to the refusal boundary.*
8. *P is the* Pkg *provenance over* $\mathsf{UltrafilterBoundary}^{\uparrow}$*,* $\mathsf{ConstructiveUltrafilterBoundary}^{\uparrow}$*,* $\mathsf{CountableChoiceBoundary}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UltrafilterBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, B, Q, L, R, H, C, P, N$. It has no coordinate for maximality, quotient completion equality, ultrafilter existence, or hidden choice.

Theorem 7751.2 (Finite boundary route)**.** *For every accepted* $\mathsf{UltrafilterBoundary}^{\uparrow}$ *carrier* $U = (F, B, Q, L, R, H, C, P, N)$*, every completion-facing filter read is routed through the displayed filter row* F*, finite Boolean-branch ledger* B*, quotient-refusal row* Q*, L10 handoff row* L*, refusal row* R*, and structural rows* H, C, P, N*. The packet exports no maximal-filter extension, no ultrafilter selector, no quotient-filter equality, and no host completion theorem.*

Proof. Project the carrier of Definition 7751.1. The only positive filter data are the finite inspected row F, the finite branch ledger B, and the L10 handoff row L.

The refusal rows Q and R record the missing endpoints. They prevent a consumer from rereading finite branch data as quotient Boolean equality, maximality, an ultrafilter extension, or a selected completion object.

Finally H, C, P, N preserve the same displayed rows as transport, replay, provenance, and local naming. Since the carrier has no further coordinate, every completion-facing filter read stays inside the finite boundary route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UltrafilterBoundary}^{\uparrow}}$**.**

Constructive story: An $\mathsf{UltrafilterBoundary}^{\uparrow}$ packet is generated from inspected filter rows, finite Boolean-branch ledgers, quotient-refusal rows, L10 CauchyFilter/Completion/StreamName handoff rows, ultrafilter-refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UltrafilterBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 7751.1 and Definition 7751.2 seed the finite no-choice boundary around L10 filter and completion arguments.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UltrafilterBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ultrafilter existence, maximal-filter selection, Boolean-prime-ideal transfer, quotient completion equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-ledger stability, quotient refusal, L10 handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.UltrafilterBoundaryUp.

Chapter 7752

A Concrete Naming Certificate for FiniteResidualConfluenceWindow$^{\uparrow}$

The FiniteResidualConfluenceWindow$^{\uparrow}$ packet realizes the MetaCIC confluence audit map as bounded residual-join evidence. It sits beside the parallel-diamond frontier of chapter 4238, the local-join ledger of chapter 4126, the residual critical-pair budget of chapter 7709, the residual-join determinacy packet of chapter 5970, and the bounded normal-equality checker of chapter 752. The chapter records a finite inspection window for local residual joins; it does not claim global confluence, unrestricted residual-substitution compatibility, subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.

Definition 7752.1 (Finite residual confluence window carrier)**.** *A* FiniteResidualConfluenceWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, R, K, J, B, O, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, naming the finite MetaCIC source and target fragments inspected before residual data are read.*
2. *R is the residual-frontier row, listing the bounded redex and residual branches visible inside the window.*
3. *K is the finite budget row, recording the depth, branch, and critical-pair bounds consumed by the window.*
4. *J is the candidate local-join row, carrying the displayed common-reduct candidates for the bounded residual branches.*
5. *B is the bounded checker row, routed through the bounded normal-equality checker before a candidate join is accepted.*
6. *O is the obstruction-retention row, recording that residual-substitution compatibility and arbitrary parallel diamond remain named premises outside the packet.*
7. *G is the TasteGate preparation row for future* Nontrivial *and* FieldFaithful *instances over the displayed carrier fields.*
8. *H records componentwise* hsame *transport for $T, R, K, J, B, O, G, P, N$.*

9. *C records displayed* Cont *replay through the typed window, residual frontier, finite budget, candidate join, bounded checker, obstruction, TasteGate preparation, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, $\mathsf{MetaCICResidualCriticalPairBudget}^{\uparrow}$, $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, K, J, B, O, G, H, C, P, N$. It has no coordinate for a global Church–Rosser theorem, unrestricted substitution compatibility, subject reduction, quotient conversion equality, or a host evaluator.

Theorem 7752.2 (Finite residual confluence window obligations)**.** *For every accepted* FiniteResidualConfluenceWin *carrier $W = (T, R, K, J, B, O, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, residual-frontier exposure through R, finite budget exposure through K, candidate local-join exposure through J, bounded checker exposure through B, obstruction retention through O, TasteGate preparation through G, transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7752.1. The confluence-facing rows are T, R, K, J, B, O, and the structural rows are G, H, C, P, N.

Read the bounded evidence in order. The typed window T and residual frontier R identify the inspected peak, the budget row K bounds the branch search, and the candidate local-join row J names the common-reduct candidates that survive that finite window.

The bounded checker row B verifies only the displayed candidate comparison. The obstruction row O keeps residual-substitution compatibility and arbitrary parallel diamond outside the packet, while G, H, C, P, N preserve the same finite carrier fields for future TasteGate, transport, replay, provenance, and local naming obligations. □

Theorem 7752.3 (Finite residual confluence window non-escape)**.** *Every confluence-audit consumer of an accepted* $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$ *packet receives only bounded local join evidence through $T, R, K, J, B, O, G, H, C, P, N$. The packet may certify that a displayed residual peak has a bounded candidate join, but it cannot erase O, manufacture residual-substitution compatibility, prove a full parallel diamond, export arbitrary closed-term confluence, or supply subject reduction.*

Proof. Start with the typed window and residual frontier. They name only the finite source, target, redex, and residual branches inspected by the packet.

Use the budget and candidate rows next. The rows K and J restrict candidate common reducts to the displayed window, and the bounded checker row B compares only those candidates.

Finally read the obstruction and structural rows. The obstruction row O records the missing global inputs, while G, H, C, P, N prepare field-faithful TasteGate checking, transport, replay, source, and name the same finite surface. Thus the consumer receives bounded residual-join evidence and nothing stronger. □

Falsifiable prediction: A confluence-audit read that accepts $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$ while omitting any of $T, R, K, J, B, O, G, H, C, P, N$ refutes the carrier surface.

Independence witness: metacic-parallel-diamond-frontier, metacic-local-join-ledger, metacic-residual-critical-pair-budget, metacic-residual-join-determinacy, bounded-normal-equality-checker:

$\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a typed window, residual frontier, finite budget, candidate local join, bounded checker, obstruction retention, TasteGate preparation, transport, replay, provenance, and local NameCert row.

The Lean follow-up for $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$ must provide Nontrivial and FieldFaithful instances over the displayed carrier fields before the packet can be consumed as checked confluence evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$ packet is generated from typed-window rows, residual-frontier rows, finite budget rows, candidate local joins, bounded checker rows, obstruction-retention rows, TasteGate preparation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7752.1, Definition 7752.2, and Definition 7752.3 seed the bounded residual-join evidence surface for the MetaCIC confluence audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, unrestricted residual-substitution compatibility, parallel diamond, subject reduction, quotient syntax equality, Lean verification, or bridge checking for $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-frontier exposure, finite budget exactness, candidate local-join exactness, bounded checker exactness, obstruction retention, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.FiniteResidualConfluenceWindowUp.

Chapter 7753

A Concrete Naming Certificate for $\mathsf{NeighbourhoodSystem}^{\uparrow}$

Following Kelley and Bourbaki, a neighbourhood system is the pointwise local presentation of topology by neighbourhood rows and local bases. The $\mathsf{NeighbourhoodSystem}^{\uparrow}$ packet records that standard presentation as finite BEDC NameCert data: it sits beside the topology carrier of chapter 88, the filter-basis interface of chapter 1601, the local filter route of chapter 4133, the countable-base surface of chapter 1901, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is a finite local-system interface for topology and completion consumers; it is not a host powerset of neighbourhoods, a quotient filter, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7753.1 (Neighbourhood system carrier)**.** *A* $\mathsf{NeighbourhoodSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, N, B, T, L, W, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *X is the point row, naming the displayed source point or finite point-window at which the local system is read.*
2. *N is the neighbourhood-window row, listing finite neighbourhood tokens around X together with their displayed containment and refinement requests.*
3. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row generated by the finite neighbourhood windows and their meet/refinement witnesses.*
4. *T is the topology-compatibility row, routing the neighbourhood windows through the open-set carrier and topology provenance of chapter 88.*
5. *L is the local-basis row, recording the basis tokens that refine each displayed neighbourhood window and the second-countable handoff when a countable-base consumer is present.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when local neighbourhood reads are sampled by completion-facing consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the point, neighbourhood, basis, topology, window, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for the point, neighbourhood-window, filter-base, topology, local-basis, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a local neighbourhood request through X, N, B, T, L, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{NeighbourhoodSystem}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{NeighborhoodFilter}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *Q is the local* $\mathsf{NameCert}_{\mathsf{NeighbourhoodSystem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, N, B, T, L, W, R, E, H, C, P, Q$. It has no coordinate for host powersets of neighbourhoods, quotient filters, selected limits, countable choice, host equality of topology structures, or ambient Real completeness.

Theorem 7753.2 (Neighbourhood system NameCert obligations)**.** *For every accepted* $\mathsf{NeighbourhoodSystem}^{\uparrow}$ *carrier $S = (X, N, B, T, L, W, R, E, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{NeighbourhoodSystem}^{\uparrow}}$ *surface consists exactly of point exposure, neighbourhood-window exposure, filter-base generation, topology compatibility, local-basis refinement, finite-window exposure, RegSeqRat readback, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must display*

$$X, N, B, T, L \longmapsto W, R, E, H, C, P, Q$$

before a Real or topology consumer can inspect the route.

Proof. Project the carrier of Definition 7753.1. The accepted packet exposes exactly the rows $X, N, B, T, L, W, R, E, H, C, P, Q$.

The local-topology rows are upstream. The point row X, neighbourhood-window row N, filter-base row B, topology-compatibility row T, and local-basis row L jointly present the neighbourhood system as finite data tied to the sibling topology, filter-base, neighbourhood-filter, and second-countable routes.

The completion-facing rows are downstream. StreamName windows, RegSeqRat readback, and the terminal Real seal are read only after the local system rows have been displayed. Transport, replay, provenance, and local naming preserve that same order, so no host powerset, quotient filter, selected limit, countable choice, or ambient Real-completeness coordinate is introduced. □

Theorem 7753.3 (Neighbourhood system filter-basis route)**.** *Every local-filter or completion-facing consumer of an accepted* $\mathsf{NeighbourhoodSystem}^{\uparrow}$ *carrier reads the neighbourhood system through the route*

$$\mathsf{Topology}^{\uparrow}, \mathsf{NeighbourhoodSystem}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow}, \mathsf{NeighborhoodFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The route exports finite neighbourhood-window and local-basis data only. It does not export a host powerset of all neighbourhoods, a quotient filter object, a chosen limit, countable choice, or ambient Real completeness.

Proof. Begin with the NameCert obligations of Definition 7753.2. They force every accepted consumer to expose the point row, neighbourhood windows, generated filter-base row, topology compatibility, and local-basis refinement before any downstream read.

The local filter route is horizontal. The generated base is consumed by chapter 1601 and then by chapter 4133; a second-countable consumer may additionally read the displayed local-basis row through chapter 1901.

Finally the completion route passes through chapter 126, chapter 301, and chapter 38. These siblings receive finite windows, regular rational readbacks, and a terminal Real seal only after the neighbourhood-system rows are public, so the route cannot supply host neighbourhood powersets, quotient filters, selected limits, choice schedules, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{NeighbourhoodSystem}^{\uparrow}$ completion-facing read exposes a point row, neighbourhood-window row, generated FilterBase row, topology-compatibility row, local-basis row, StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before it can serve a topology, local-filter, or completion consumer.

Independence witness: topology, filterbase, neighborhood-filter, second-countable, streamname, regseqrat, real: none is bijective with $\mathsf{NeighbourhoodSystem}^{\uparrow}$ because this packet jointly carries pointwise neighbourhood windows, generated filter-base data, topology compatibility, local-basis refinement, finite completion windows, readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{NeighbourhoodSystem}^{\uparrow}}$.

Constructive story: A $\mathsf{NeighbourhoodSystem}^{\uparrow}$ packet is generated from a point row, neighbourhood-window row, FilterBase row, topology-compatibility row, local-basis row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeighbourhoodSystem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7753.1, Definition 7753.2, and Definition 7753.3 seed the finite neighbourhood-system route from topology and local bases through FilterBase, NeighborhoodFilter, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeighbourhoodSystem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets of neighbourhoods, quotient filters, selected limits, countable choice, host equality of topology structures, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked point exposure, neighbourhood-window exposure, filter-base generation, topology compatibility, local-basis refinement, finite-window exposure, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NeighbourhoodSystemUp.

Chapter 7754

A Concrete Naming Certificate for $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$

Following Bishop and Bridges on nested closed intervals and regular Cauchy completion, the $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ packet records the finite route by which a descending located interval family, dyadic diameter ledger, regular rational midpoint readback, finite stream windows, and terminal real seal are offered to Real and Completion consumers. It sits on the completion spine of chapter 38, chapter 301, chapter 126, and chapter 291. The packet names the choice-free Cantor intersection handoff visible to regular Cauchy readers; it is not a quotient real, an arbitrary closed-set intersection theorem, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7754.1 (Cantor intersection principle carrier)**.** *A* $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, W, R, E, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger, recording the inspected dyadic precision, the nested interval endpoints visible at that precision, and the finite proof that the displayed diameters are below the requested tolerance.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the bounded interval indices and endpoint windows read by the consumer.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *midpoint readback row obtained from D and W; it records the regular rational name determined by the displayed nested windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic ledger, stream windows, and regular rational readback have been replayed.*
5. *J is the intersection-claim row, recording that every displayed interval window contains the same readback route and that later windows refine earlier ones.*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, Real seal, intersection row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion request through D, W, R, E, J, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionPrinciple}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $D, W, R, E, J, H, C, P, N$. It has no coordinate for a selected point by countable choice, quotient stream equality, host real equality, arbitrary closed subsets, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7754.2 (Cantor intersection principle NameCert obligations)**.** *For every accepted* CantorIntersectionPrinc *carrier $I = (D, W, R, E, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionPrinciple}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed dyadic diameter ledger, finite stream-window row, regular rational midpoint readback, terminal Real seal, intersection row, transport, replay, provenance, and local naming;*
2. *nested-window exactness, requiring every Real or Completion read to inspect the same finite D, W interval window before it reaches R;*
3. *regular-readback stability, requiring the midpoint route to be consumed as a* $\mathsf{RegSeqRat}^{\uparrow}$ *name before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
4. *intersection nonescape, requiring J to record refinement of displayed windows without adding an arbitrary closed-set carrier;*
5. *ledger refusal, excluding countable choice, quotient streams, host equality, trichotomy, open-cover compactness, selected representatives, and ambient Real completeness.*

Proof. Carrier admission is projection from Definition 7754.1. The projected rows are exactly the dyadic diameter ledger D, stream windows W, regular midpoint readback R, terminal Real seal E, intersection row J, and structural rows H, C, P, N.

Nested-window exactness is read through D and W. The consumer sees only the bounded endpoint windows and the dyadic diameter comparison stored in the packet, so there is no separate source from which a point could be chosen.

The regular-readback row R repacks the displayed midpoint route as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff used by chapter 301. The terminal seal E is therefore downstream of the same finite route and cannot be detached from it.

The intersection and refusal rows follow from the remaining coordinates. Row J records refinement of the displayed windows, while H, C, P, N transport, replay, source, and name that route; none of these rows stores countable choice, quotient equality, host equality, trichotomy, open-cover compactness, selected representatives, or ambient completeness. □

Theorem 7754.3 (Cantor intersection principle completion route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ *carrier $I = (D, W, R, E, J, H, C, P, N)$ must factor through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes a nested located interval family by dyadic diameter windows, then reads the regular rational midpoint name, and only then reaches the terminal Real seal. It exports no quotient-real construction, selected representative, countable-choice schedule, arbitrary closed-set intersection principle, trichotomy, open-cover compactness, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the displayed rows D and W. They bind the inspected precision, endpoint windows, and diameter comparison before any completion-facing name can be read.

The row R consumes exactly those displayed windows as a regular rational readback. Thus the completion-facing handoff has already passed through the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces before the Real seal is available.

Finally use the intersection row J together with H, C, P, N. These rows record refinement, transport, replay, provenance, and local naming for the same finite route. Since the carrier has no row for quotient reals, selected representatives, countable choice, arbitrary closed sets, trichotomy, open-cover compactness, host equality, or ambient completeness, every consumer remains inside the stated completion route. □

Falsifiable prediction: Every $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ Real-completion read factors through one DyadicRatCore diameter ledger, one StreamName finite-window row, one RegSeqRat midpoint readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, one intersection-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nested diameter ledger, finite interval windows, regular midpoint readback, Real seal, refinement row, transport, replay, provenance, and local NameCert rows as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionPrinciple}^{\uparrow}$ packet is generated from a DyadicRatCore diameter ledger, StreamName finite interval windows, RegSeqRat midpoint readback, a terminal RealUp seal, an intersection-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7754.1, Definition 7754.2, and Definition 7754.3 seed the finite Cantor-intersection completion route from dyadic interval diameters through StreamName, RegSeqRat, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, selected representatives, countable choice, arbitrary closed-set intersections, trichotomy, open-cover compactness, host equality, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window exactness, regular-readback stability, Real-seal nonescape, intersection refinement, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionPrinci

Chapter 7755

A Concrete Naming Certificate for $\mathsf{BirkhoffContraction}^{\uparrow}$

Following Birkhoff's Hilbert projective metric contraction theorem, the $\mathsf{BirkhoffContraction}^{\uparrow}$ packet records the finite positive-cone route by which a contraction coefficient is read before any fixed-point or Perron–Frobenius conclusion is available. It sits over the metric carrier of chapter 57, the vector-space source of chapter 47, the positive-radius ledger of chapter 1587, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Hilbert-projective contraction budget; it is not a host fixed-point theorem, an eigenvector selection principle, a completed cone quotient, or an ambient compactness argument.

Definition 7755.1 (Birkhoff contraction carrier)**.** *A* $\mathsf{BirkhoffContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (V, K, M, \Delta, T, \tau, R, E, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row carrying the finite linear operations inspected by the positive cone request.*
2. *K is the positive cone row, represented only by displayed positive-radius and order-comparison data over* $\mathsf{PositiveReal}^{\uparrow}$*.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row naming the Hilbert projective metric observations used by the packet.*
4. *Δ is the finite diameter ledger for the image of the displayed positive cone under the operator row.*
5. *T is the positive operator row, recording only the displayed cone-preserving action on the rows already exposed.*
6. *τ is the contraction-budget row, a* $\mathsf{PositiveReal}^{\uparrow}$ *or regular rational readback witness below the unit bound.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to compare the displayed diameter and contraction budget.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the Hilbert-metric contraction estimate.*
9. *H records componentwise* hsame *transport of all displayed rows.*
10. *C records displayed* Cont *replay through $V, K, M, \Delta, T, \tau, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{BirkhoffContraction}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, K, M, \Delta, T, \tau, R, E, H, C, P, N$. There is no coordinate for a selected eigenvector, quotient projective point, host compactness theorem, fixed-point conclusion, or ambient completed cone.

Lean4 ✓ `BEDC.Derived.BirkhoffContractionUp.BirkhoffContractionTasteGate single carrier alignment`

Theorem 7755.2 (Birkhoff contraction NameCert obligations)**.** *For every accepted* $\mathsf{BirkhoffContraction}^{\uparrow}$ *carrier $B = (V, K, M, \Delta, T, \tau, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BirkhoffContraction}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the vector source, positive cone, Hilbert projective metric row, finite image-diameter ledger, positive operator row, contraction-budget row, rational readback, terminal real seal, transport, replay, provenance, and local naming row;*
2. *common-cone discipline, requiring K, M, Δ, T, τ to be read over the same displayed positive cone and vector-space source;*
3. *budget exactness, requiring the contraction estimate to pass through the diameter ledger Δ, the budget row τ, the regular readback R, and the real seal E;*
4. *replay stability, requiring transport and continuation rows H, C to preserve the same budget route before P, N name the packet;*
5. *ledger refusal, excluding selected fixed points, eigenvectors, projective quotient representatives, host compactness, completed cones, and Perron–Frobenius conclusions.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 7755.1. Its displayed rows are exactly $V, K, M, \Delta, T, \tau, R, E, H, C, P, N$, so carrier admission exposes only the finite vector, cone, metric, diameter, operator, budget, readback, seal, transport, replay, provenance, and naming data.

The common-cone and budget rows force every contraction read to start from V, K, M, pass through Δ, T, τ, and then read R before the terminal real seal E. The structural rows H, C, P, N transport and name that same route without creating an additional object.

Since the carrier has no coordinate for a selected eigenvector, projective quotient representative, host fixed point, compactness theorem, completed cone, or Perron–Frobenius conclusion, the listed rows exhaust the local NameCert surface. □

Theorem 7755.3 (Birkhoff Hilbert-metric budget)**.** *Every Hilbert-projective contraction read exported by an accepted* $\mathsf{BirkhoffContraction}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{VecSpace}^{\uparrow}, \mathsf{PositiveReal}^{\uparrow}, \mathsf{Met}^{\uparrow} \longmapsto \Delta, T, \tau \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N.$$

Thus a downstream metric or completion-facing consumer receives only the displayed vector-space source, positive-cone ledger, Hilbert projective metric observations, finite image-diameter budget, positive operator row, contraction coefficient readback, terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no fixed point, selected eigenline, quotient projective representative, completed cone, host compactness theorem, or ambient Perron–Frobenius principle.

Proof. Start with the carrier rows of Definition 7755.1. The only metric-facing coordinates are M, Δ, τ, R, E, and the only source rows for them are V, K, T.

Apply the obligation surface of Definition 7755.2. The common-cone row keeps the Hilbert projective metric observations attached to the displayed vector source and positive cone, while budget exactness routes the contraction coefficient through Δ, τ, R, E.

Finally transport through H and replay through C. Since P and N only record provenance and local naming for the same finite packet, the contraction budget remains a finite Hilbert-metric read and cannot export a host fixed-point or eigenvector conclusion. □

Falsifiable prediction: An accepted $\mathsf{BirkhoffContraction}^{\uparrow}$ read is refuted if it exports a Hilbert-projective contraction estimate without displaying the vector-space source, positive-cone ledger, Hilbert metric row, finite image-diameter ledger, positive operator row, contraction-budget row, regular rational readback, terminal real seal, transport, replay, provenance, and local naming.

Independence witness: vecspace, metric, positivereal, regseqrat, real: none alone is bijective with $\mathsf{BirkhoffContraction}^{\uparrow}$ because the packet jointly records positive-cone source data, Hilbert projective metric observations, finite image-diameter budgeting, positive operator action, contraction readback, terminal real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffContraction}^{\uparrow}$ packet is generated from a VecSpaceUp row, a positive-cone row over PositiveRealUp data, a MetricSpaceUp Hilbert-projective metric row, a finite image-diameter ledger, a positive operator row, a contraction-budget row, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7755.1, Definition 7755.2, and Definition 7755.3 seed the finite Hilbert-projective contraction route from displayed positive-cone metric data to a Real contraction-budget seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close fixed-point existence, selected eigenvectors, projective quotient representatives, completed cone compactness, Perron–Frobenius conclusions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-cone discipline, Hilbert-metric budget exactness, replay stability, refusal exactness, provenance accounting, and local naming for BEDC.Derived.BirkhoffContractionUp.

Chapter 7756

A Concrete Naming Certificate for $\mathsf{CauchyModulusCertificate}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names with explicit moduli, the $\mathsf{CauchyModulusCertificate}^{\uparrow}$ packet records the finite threshold data by which a convergence modulus is displayed before a regular rational readback or terminal real-name seal is consumed. It sits after the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38, and the convergence-modulus ledger of chapter 2074. The packet names a finite certificate route for L10 Real and Completion consumers; it is not a selected limit, quotient Cauchy name, countable-choice schedule, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7756.1 (Cauchy modulus certificate carrier)**.** *A* $\mathsf{CauchyModulusCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request row.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row selected for the tolerance request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed window.*
4. *M is the convergence-modulus certificate row, recording that every later window accepted by the packet satisfies the request named by D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after D, W, Q, M have been displayed.*
6. *H records componentwise* hsame *transport for tolerance requests, finite windows, regular readback, modulus certificate rows, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a tolerance request through the window, readback, modulus certificate, and Real seal route.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusCertificate}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, W, Q, M, E, H, C, P, N$. No coordinate stores quotient stream equality, selected representatives, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusCertificateUp.CauchyModulusCertificateTasteGate single carrier alignment`

Theorem 7756.2 (Cauchy modulus certificate NameCert obligations)**.** *For every accepted* CauchyModulusCertificat *carrier* $K = (D, W, Q, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusCertificate}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission, finite window admission, regular rational readback, convergence-modulus certificate exposure, Real seal handoff, componentwise stability transport, displayed replay, provenance, local naming, and ledger refusal for quotient streams, selected representatives, selected limits, countable choice, host real equality, and ambient Real completeness.*

Proof. Project the carrier of Definition 7756.1. The displayed route begins with the dyadic tolerance request D, then exposes the finite window W, the regular readback Q, and the convergence-modulus certificate row M before the terminal seal E can be consumed.

Componentwise transport through H and replay through C preserve that order. The provenance and naming rows P, N source and name the same finite route, rather than adding analytic data outside the carrier.

The refusal ledger is the complement of the displayed coordinates. Since the carrier has no row for quotient stream equality, selected representatives, selected limits, countable choice, host real equality, or ambient completeness, no local NameCert obligation can export those objects. □

Theorem 7756.3 (Cauchy modulus certificate regular-Cauchy and Real route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealModulusOfConvergence}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusCertificate}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $D, W, Q, M, E, H, C, P, N$*. Thus the certificate supplies a named regular-Cauchy and Real handoff without asserting Real completeness, quotient Cauchy-name equality, selected limits, or countable-choice schedules.*

Proof. Start from an accepted carrier $K = (D, W, Q, M, E, H, C, P, N)$. The tolerance row D and finite window row W must be displayed before any regular rational readback is available.

The readback row Q is then the only route into $\mathsf{RegSeqRat}^{\uparrow}$. The modulus row M records the convergence certificate consumed by chapter 2074, and the terminal seal E is downstream of that certificate row.

Finally inspect H, C, P, N. They transport, replay, source, and name the same finite route, so any Real or Completion consumer receives only the displayed regular-Cauchy handoff. The carrier inventory contains no completeness, quotient, selected-limit, choice, or host-equality coordinate, and therefore cannot export such a principle. □

Falsifiable prediction: A $\mathsf{CauchyModulusCertificate}^{\uparrow}$ read is refuted whenever a Real or Completion consumer uses a convergence-modulus certificate without displaying the DyadicRatCore tolerance request, StreamName finite window, RegSeqRat readback, RealModulusOfConvergence certificate row, Real seal, componentwise transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, realmodulusofconvergence: none alone is bijective with $\mathsf{CauchyModulusCertificate}^{\uparrow}$ because the certificate jointly records a tolerance request, a finite tail window, a regular rational readback, a convergence-modulus row, a Real seal, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusCertificate}^{\uparrow}}$**.**

Constructive story: A CauchyModulusCertificate$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance request, a StreamName finite tail window, a RegSeqRat readback row, a RealModulusOfConvergence certificate row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusCertificate}\uparrow}$, seed)

Scope closed: Definition 7756.1, Definition 7756.2, and Definition 7756.3 seed the finite certificate route from dyadic tolerance requests to regular Cauchy readback and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusCertificate}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient Cauchy-name equality, selected limits, selected representatives, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked tolerance admission, window admission, regular readback, convergence-modulus certificate exposure, Real-seal handoff, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.CauchyModulusCertificateUp.

Chapter 7757

A Concrete Naming Certificate for $\mathsf{InducedCharacter}^{\uparrow}$

Following Serre on finite-group characters and Fulton–Harris on induction and restriction, the $\mathsf{InducedCharacter}^{\uparrow}$ packet records the character-level coset-sum construction as finite NameCert data. It consumes the finite group source of chapter 41, the subgroup source of chapter 80, the coset ledger of chapter 3976, the finite action rows of chapter 3662, the trace and class-function surface of chapter 245, the value-row consumer of chapter 3574, the induction–restriction packet of chapter 247, and the reciprocity provenance of chapter 5806. The packet names only the displayed coset-sum character row, its transport, its class-function stability, and its handoff to induction and reciprocity consumers; it does not assert irreducible decomposition, Schur orthogonality, a completed character table, or a host category of representations.

Definition 7757.1 (InducedCharacter carrier)**.** *An* $\mathsf{InducedCharacter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{C} = (G, H, Q, V, \chi, \iota, \sigma, \kappa, \rho, \lambda, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row.*
2. *H is the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *row naming the subgroup inclusion inside the same finite group source.*
3. *Q is the consumed* $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$ *row listing the finite left-coset representatives used only as display indices.*
4. *V is the consumed* $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *row whose restricted trace row supplies the source character on H.*
5. *χ is the consumed* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *row naming the source class-function and trace surface.*
6. *ι is the displayed* $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ *induction row.*
7. *σ is the finite coset-sum ledger assigning to each displayed group row the finite sum of the source-character values over the accepted representative rows of Q.*
8. *κ is the class-function stability row, recording conjugation transport of σ through G, H, Q, and χ.*

9. *ρ is the* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *provenance row tying the character-level handoff to the induction–restriction comparison.*

10. *λ is the downstream consumer ledger for* $\mathsf{CharacterTable}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, *and* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *reads.*

11. *P is the* Pkg *provenance over the consumed rows,* BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}$ *row.*

The carrier contains no host induced representation, no selected representative object beyond the displayed coset ledger, no irreducible-character decomposition, and no ambient character algebra outside these rows.

Lean4 (definition) `BEDC.Derived.InducedCharacterUp.InducedCharacterUp`

Theorem 7757.2 (InducedCharacter NameCert obligations)**.** *For every accepted* $\mathsf{InducedCharacter}^{\uparrow}$ *packet*

$$\mathcal{C} = (G, H, Q, V, \chi, \iota, \sigma, \kappa, \rho, \lambda, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission exposes the displayed group, subgroup, coset-space, finite-representation, character-theory, induced-representation, coset-sum, class-stability, reciprocity, consumer, provenance, and local naming rows of Definition 7757.1;*

2. *coset-representative invariance requires the value of σ to be unchanged under* hsame *transport of the displayed representative rows in Q, with every replacement read still sourced by H and G;*

3. *class-function stability requires the conjugation transport row κ to preserve the induced value row through the trace and class-function surface χ;*

4. *restriction/induction handoff requires every* $\mathsf{InducedRep}^{\uparrow}$ *read to pass through the displayed induction row ι and the same subgroup, representation, and coset ledgers used by σ;*

5. *reciprocity provenance requires every Frobenius comparison read to pass through ρ and the character-pairing boundary of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$, *with no bypass through a host adjunction or semisimple decomposition.*

Proof. Project the carrier rows of Definition 7757.1. Carrier admission is the displayed tuple $\mathcal{C}$ itself, so every source and consumer row is already visible in finite BHist form.

For coset-representative invariance, compare two accepted representative ledgers in Q. The transport is componentwise hsame over the consumed group and subgroup rows, and each summand is still read through the source character row χ. Reindexing the finite ledger therefore changes only the display of the summation, not the named character row.

Class-function stability is the row κ. It transports the coset-sum value along conjugation in G and checks that the source trace row remains inside the class-function surface named by $\mathsf{CharacterTheory}^{\uparrow}$.

For the induction handoff, route every representation-level read through ι and the public boundary of chapter 247. For reciprocity provenance, route every comparison read through ρ and the finite character-pairing boundary of chapter 5806. Repacking these projected rows with λ, P, and N gives exactly the five listed obligations. □

Theorem 7757.3 (InducedCharacter table consumer boundary)**.** *Any* $\mathsf{CharacterTable}^{\uparrow}$ *consumer of an accepted* $\mathsf{InducedCharacter}^{\uparrow}$ *packet may read only the finite group row* G*, subgroup row* H*, coset ledger* Q*, source character row* χ*, coset-sum row* σ*, class-stability row* κ*, and consumer ledger* λ*. It may not import an irreducible decomposition, Schur orthogonality row, completed character table, host representative selector, or ambient representation category unless that row is already present in the displayed packet.*

Proof. Project the consumer request through the displayed carrier. The character-table consumer receives the group and class-function rows only through G and χ, and it receives coset indexing only through Q.

The induced value row is σ, while its class-function stability is κ. The consumer ledger λ records exactly the downstream reads exposed to $\mathsf{CharacterTable}^{\uparrow}$ together with the representation-induction and reciprocity handoff rows.

No remaining coordinate of $\mathcal{C}$ names an irreducible decomposition, a Schur row, a completed table, a representative selector, or a host representation category. Hence the table-facing boundary is precisely the finite packet described above. □

Closure status for $\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7757.1, Definition 7757.2, and Definition 7757.3 seed the finite coset-sum character carrier, its five NameCert obligations, and its character-table consumer boundary over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{CosetSpace}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, and $\mathsf{FrobeniusReciprocity}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close irreducible decomposition, Schur orthogonality, completed character-table computation, host representative selection, host representation categories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coset-representative invariance, class-function stability, restriction/induction handoff, and reciprocity provenance for the same finite packet.

Constructive story: An $\mathsf{InducedCharacter}^{\uparrow}$ packet is generated from finite group and subgroup sources, a finite coset-space ledger, a represented source character, an induced-representation handoff, a finite coset-sum ledger, class-function transport, Frobenius reciprocity provenance, downstream consumer rows, package provenance, and a local NameCert row.

Chapter 7758

A Concrete Naming Certificate for $\mathsf{CompletionModulusExtractor}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CompletionModulusExtractor}^{\uparrow}$ packet records the finite extraction surface by which completion-facing data exposes a modulus row before any completed object is consumed. It sits beside chapter 1506, chapter 1501, chapter 301, chapter 291, chapter 126, and chapter 38: those chapters supply the metric-completion, filter-basis, regular rational, dyadic tolerance, finite-window, and real-seal rows, while this chapter names the extractor that keeps the modulus route finite and inspectable.

Definition 7758.1 (Completion modulus extractor carrier)**.** *A* $\mathsf{CompletionModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, B, S, R, D, W, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the completion-facing request.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row that records the finite basis through which the source is inspected.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened by the extractor.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the extracted regular observations.*
6. *W is the extracted modulus row, recording for each admitted precision a finite window and dyadic tolerance witness.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, S, R, D, W have been displayed.*
8. *H records componentwise* hsame *transport for source, basis, window, readback, dyadic, modulus, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion request through the same finite extractor rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusExtractor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusExtractor}^{\uparrow}}$ *row.*

The classifier accepts exactly packets carrying these rows. It exports no quotient completion, selected limit, countable-choice selector, host real equality, hidden completeness principle, or modulus not displayed in W.

Lean4 (definition) `BEDC.Derived.CompletionModulusExtractorUp`

Theorem 7758.2 (Completion modulus extractor NameCert obligations)**.** *For every accepted* $\mathsf{CompletionModulusExtractor}^{\uparrow}$ *carrier*

$$E = (M, B, S, R, D, W, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionModulusExtractor}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-request readback, basis readback, finite-window admission, regular rational readback, dyadic tolerance visibility, extracted-modulus visibility, terminal real sealing, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing consumer must read the extracted modulus row W through*

$$M, B, S, R, D, W, L, H, C, P, N$$

and through no hidden completion or choice coordinate.

Proof. Project the carrier of Definition 7758.1. The source row M names the completion-facing request, while B, S, R, D expose the finite basis, window, regular readback, and dyadic tolerance data from which a modulus can be read.

Inspect the extracted row W. It is the only coordinate whose role is to present the precision-to-window and precision-to-tolerance schedule, so any consumer asking for modulus data must pass through that displayed row.

Finally read L, H, C, P, N. The terminal real seal is downstream of the modulus display, and the structural rows transport, replay, source, and locally name the same finite route. Since no coordinate of E stores a quotient completion, selected limit, countable-choice selector, host equality, or hidden completeness principle, the listed rows exhaust the local NameCert obligation surface. □

Theorem 7758.3 (Completion modulus extractor finite route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompletionModulusExtractor}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto W \longmapsto \mathsf{Real}^{\uparrow}$$

implemented by the displayed rows $M, B, S, R, D, W, L, H, C, P, N$. The route exposes modulus data as finite NameCert evidence; it does not select a completed point, quotient Cauchy names, invoke countable choice, prove host real equality, or assert ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with M and B. The completion request is admitted only through the displayed metric-completion row, and the Cauchy-filter basis row fixes the finite observations the extractor may open.

Next inspect S, R, D, W. The stream windows determine the finite observations, the regular rational row records the readback, the dyadic ledger supplies tolerance comparison, and W packages those data as the visible modulus row.

The final rows L, H, C, P, N preserve the same route for Real-facing consumers. They seal the terminal real row, transport displayed components, replay the finite request, record provenance, and name the local packet. None of these rows adds quotient data, a selected point, choice, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionModulusExtractor}^{\uparrow}$ packet is generated from MetricCompletion requests, CauchyFilterBasis rows, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, an extracted modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7758.1, Definition 7758.2, and Definition 7758.3 seed the finite extraction route from completion requests through basis windows, regular readback, dyadic tolerance, displayed modulus data, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusExtractor}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CompletionModulusExtractorUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-source readback, basis-window exactness, RegSeqRat readback, DyadicRatCore tolerance exactness, extracted-modulus visibility, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusExtractorUp.

Chapter 7759

A Concrete Naming Certificate for $\mathsf{SchauderTychonoff}^{\uparrow}$

Following Schauder and Tychonoff, the $\mathsf{SchauderTychonoff}^{\uparrow}$ packet records the finite handoff surface used when a compact convex source, a continuous self-map, and an approximate fixed-point mesh are displayed together. It sits after the Schauder fixed-point route of chapter 1923, the finite approximate fixed-point packet of chapter 6087, the compact-metric source of chapter 129, the convex membership surface of chapter 208, the locally convex ambient source of chapter 3386, and the continuous-map graph row of chapter 130. The chapter names a compact-convex fixed-point handoff as finite NameCert data; it does not claim arbitrary product compactness, choice-selected fixed points, quotient equality, or a host Schauder–Tychonoff theorem.

Definition 7759.1 (Schauder–Tychonoff handoff carrier)**.** *A* $\mathsf{SchauderTychonoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, V, X, f, M, A, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact-source row that supplies the finite inspection windows for the domain.*
2. *V is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *ambient row carrying the finite seminorm or weak-neighborhood windows available to the packet.*
3. *X is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the same displayed subset, including the affine-combination and membership reads used by fixed-point consumers.*
4. *f is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row whose graph reads stay inside the displayed compact convex source.*
5. *M is the finite mesh and radius ledger consumed from chapter 6087.*
6. *A is the approximate fixed-point displacement row, recording the inspected points and tolerance windows through which $f(x)$ is compared with x.*
7. *R is the fixed-point handoff row to chapter 1923, naming only the finite compact-convex route exposed by the packet.*
8. *H records componentwise* hsame *transport for the compact source, locally convex source, convex-set membership, continuous-map graph, mesh, displacement, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact and convex source reads through f, M, A, R to the structural rows.*
10. *P is the* Pkg *provenance over* $\mathsf{SchauderTychonoff}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *finite approximate fixed-point rows,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchauderTychonoff}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $K, V, X, f, M, A, R, H, C, P, N$. It has no coordinate for arbitrary product compactness, a choice-selected exact fixed point, quotient equality of maps or points, a completed locally convex dual, or a host Schauder–Tychonoff theorem.

Theorem 7759.2 (Schauder–Tychonoff NameCert obligations)**.** *For every accepted* $\mathsf{SchauderTychonoff}^{\uparrow}$ *carrier*

$$S = (K, V, X, f, M, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchauderTychonoff}^{\uparrow}}$ *surface consists exactly of compact-source admission for K, locally convex ambient admission for V, convex-set admission for X, continuous self-map visibility for f, finite mesh and radius admission for M, approximate displacement readback for A, fixed-point handoff through R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every compact-convex fixed-point consumer must read the dependency route

$$K, V, X, f \longmapsto M, A \longmapsto R \longmapsto H, C, P, N$$

before using the packet as a Schauder–Tychonoff handoff. The route exports finite certificate rows only; it exports no arbitrary product compactness theorem, no selected fixed point, no quotient equality, and no host theorem beyond the displayed compact-convex fixed-point handoff.

Proof. Project the carrier of Definition 7759.1. The compact source is K, the locally convex ambient source is V, the convex subset source is X, and the self-map graph is f.

Read the finite approximation data next. The mesh and radius row M is supplied by chapter 6087, and the displacement row A records only the inspected point and tolerance comparisons needed for the handoff.

Route the fixed-point-facing read through R and chapter 1923. Transport and replay through H and C preserve the same displayed rows, while P and N name the resulting packet.

Since the carrier has no coordinate for product compactness, an exact selected fixed point, quotient equality, a completed dual, or a host Schauder–Tychonoff theorem, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderTychonoff}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderTychonoff}^{\uparrow}$ packet is generated from compact-source, locally convex, convex-set, continuous-map, finite mesh, displacement, fixed-point handoff, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchauderTychonoff}^{\uparrow}}$, seed)

Scope closed: Definition 7759.1 and Definition 7759.2 seed the finite compact-convex fixed-point handoff over compact-metric, convex-set, locally convex, continuous-map, finite approximate fixed-point, and Schauder fixed-point rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchauderTychonoff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary product compactness, choice-selected fixed points, quotient equality, completed locally convex duality, a host Schauder–Tychonoff theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex source routing, continuous-map visibility, finite mesh and displacement readback, fixed-point handoff, transport, replay, provenance, and local naming for the same packet.

Chapter 7760

A Concrete Naming Certificate for $\mathsf{BishopLocatedSpace}^{\uparrow}$

Following Bishop and Bridges, a Bishop-style located space is a metric object whose topology-facing and set-facing locatedness reads are exposed by finite means before completion or real-name consumers can inspect them. The $\mathsf{BishopLocatedSpace}^{\uparrow}$ packet records this interface on the L10 topology, metric, completion, and Real route of section 9467.12. It sits over the metric source of chapter 57, the topology source of chapter 88, the located-subset sibling chapter 1541, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate names only the displayed finite locatedness route; it does not add quotient completion, selected nearest points, open-cover compactness, real trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7760.1 (Bishop located-space carrier)**.** *A* $\mathsf{BishopLocatedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, T, L, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, distance witness, and finite point-observation histories from which locatedness requests are opened.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, recording the displayed open-neighbourhood interface consumed by topology-facing readers of the same point observations.*
3. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *locatedness row. It ties each admitted set or source family to the finite distance interface of chapter 1541.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison windows for metric distances, located-set distances, and topology-neighbourhood radii.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the metric, topology, located-set, and regular-rational rows have all been displayed.*
6. *H stores componentwise* hsame *transport for M, T, L, Q, E and for the structural certificate rows.*
7. *C stores displayed* Cont *replay from metric point observations to topology neighbourhoods, located-set requests, regular rational readback, and the terminal Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, L, Q, E, H, C, P, N$. It has no coordinate for quotient completion, selected nearest points, open-cover compactness, real trichotomy, countable choice, host equality of points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7760.2 (BishopLocatedSpace NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedSpace}^{\uparrow}$ *carrier*

$$B = (M, T, L, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedSpace}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *point classifier, distance witness, and finite observation histories in M;*
2. *topology compatibility for the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row T, requiring each topology-facing read to remain attached to the same metric observations;*
3. *located-set exactness for L, requiring every locatedness request to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *distance interface before a set-facing distance is exported;*
4. *regular-rational and Real readback for Q and E, requiring every rational comparison window to be sealed as* $\mathsf{Real}^{\uparrow}$ *only after* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is visible;*
5. *structural replay for H, C, P, N, requiring componentwise transport, displayed continuation, package provenance, and local naming to preserve exactly the same rows.*

No obligation row may call on quotient completion, selected nearest points, open-cover compactness, trichotomy, countable choice, host point equality, or ambient Real completeness.

Proof. Carrier admission is projection from Definition 7760.1. The metric row M, topology row T, located-set row L, regular rational row Q, Real seal row E, and structural rows H, C, P, N are the only rows in the packet.

The first three obligations bind the analytic source. A topology-facing read must use the neighbourhood interface T over the same metric observations in M, and a set-facing locatedness read must use L before any distance-to-set data are exported.

The fourth obligation routes numerical data through Q and E. Rational comparison windows become public only as regular rational readback, and the Real seal is exposed only after that readback is displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the excluded completion, selection, compactness, trichotomy, choice, host-equality, and ambient-completeness principles have no coordinate in the carrier, the five rows exhaust the local NameCert surface. □

Theorem 7760.3 (BishopLocatedSpace metric-topology handoff)**.** *Every topology-facing or located-set-facing consumer of an accepted* $\mathsf{BishopLocatedSpace}^{\uparrow}$ *carrier first reads the* $\mathsf{Met}^{\uparrow}$ *row M, then the* $\mathsf{Topology}^{\uparrow}$ *row T or the* $\mathsf{LocatedSet}^{\uparrow}$ *row L, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, and only then the* $\mathsf{Real}^{\uparrow}$ *seal row E. Thus the public handoff is the finite route*

$$M, T, L, Q \longmapsto E$$

together with H, C, P, N.

Proof. Start with an accepted carrier $B = (M, T, L, Q, E, H, C, P, N)$. The carrier definition gives no direct route from a topology or set consumer to E; every read begins with the metric observations in M.

If the consumer asks for a neighbourhood read, the route uses T over the same point observations. If it asks for a located-set distance, the route uses L, whose sibling interface is fixed by chapter 1541.

Both cases then pass through Q. The regular rational row is the only row that packages the finite comparison windows used by the metric, topology, and located-set reads.

The Real seal E is therefore downstream of M, T, L, Q, and H, C, P, N preserve this order under transport, replay, provenance, and naming. No host equality, quotient completion, or selected endpoint can bypass the displayed handoff. □

Theorem 7760.4 (BishopLocatedSpace Real readback nonescape)**.** *The* $\mathsf{Real}^{\uparrow}$ *readback exported by an accepted* $\mathsf{BishopLocatedSpace}^{\uparrow}$ *packet is reachable only through* $M, T, L, Q, E, H, C, P, N$*. A Real-facing consumer receives the displayed metric source, topology interface, located-set distance route, regular rational readback, terminal Real seal, componentwise transport, displayed continuation, package provenance, and local NameCert row; it receives no quotient completion, selected nearest point, open-cover compactness, real trichotomy, countable choice, host point equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The only Real-facing coordinate is E. By Definition 7760.1, E is paired with Q, and Q is paired with the metric row M, topology row T, and located-set row L.

Apply Definition 7760.2. The five local obligation rows require metric-source admission, topology compatibility, located-set exactness, regular-rational and Real readback, and structural replay before the Real seal is public.

Apply Definition 7760.3. It identifies M, T, L, Q as the displayed handoff route to E.

Thus every Real readback remains inside the finite carrier. Since the packet contains no excluded quotient, selection, compactness, trichotomy, choice, host-equality, or ambient-completeness coordinate, none of those principles can escape through the Real seal. □

Falsifiable prediction: Any accepted $\mathsf{BishopLocatedSpace}^{\uparrow}$ consumer that reaches a Real-facing locatedness read while bypassing the metric source, topology row, located-set row, RegSeqRat readback, Real seal, transport, replay, provenance, or local NameCert row refutes the carrier surface.

Independence witness: metric, topology, locatedset, regseqrat, real: $\mathsf{BishopLocatedSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, topology interface, located-set distance route, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local NameCert registration as one Bishop-style located-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedSpace}^{\uparrow}$ packet is generated from a Metric source, Topology neighbourhood interface, LocatedSet distance route, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7760.1, Definition 7760.2, Definition 7760.3, and Definition 7760.4 seed the Bishop-style finite locatedness route over Metric, Topology, LocatedSet, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected nearest points, open-cover compactness, real trichotomy, countable choice, host point equality, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, topology compatibility, located-set exactness, regular rational readback, Real-seal nonescape, transport replay, and local NameCert coverage for BEDC.Derived.BishopLocatedSpaceUp.

Chapter 7761

A Concrete Naming Certificate for $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$

Following Bishop and Bridges on finite metric constructions and Kelley on metric projection neighbourhoods, the $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ packet records the finite selector boundary used when a displayed finite projection table is already available. It sits after the projection carrier of chapter 2356, the finite projection table of chapter 7348, and the ambient metric source of chapter 57. The packet names a consumer-side selector row for already displayed finite candidates; it exports no nearest-point choice, quotient metric equality, or ambient completeness.

Definition 7761.1 (Finite metric projection selector carrier)**.** *A* $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the consumed* $\mathsf{Met}^{\uparrow}$ *source row for the displayed finite metric data.*
2. *F is the consumed* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *table row naming the finite candidate distances and endpoint evidence already admitted by the parent finite projection packet.*
3. *Q is the finite selector ledger, recording the displayed index or candidate row selected from the table F.*
4. *E is the endpoint handoff row, read only as the finite endpoint evidence already exposed by F.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *distance seal attached to the selected candidate through the finite projection table.*
6. *H records componentwise* hsame *transport for the metric source, finite projection table, selector ledger, endpoint handoff, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a projection request through the finite table, selector ledger, endpoint handoff, and terminal seal.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$, $\mathsf{FiniteMetricProjection}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricProjectionSelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host nearest-point function, quotient metric equality, hidden endpoint choice, countable selection schedule, or ambient completeness theorem.

Theorem 7761.2 (Finite metric projection selector NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ *carrier* $S = (M, F, Q, E, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricProjectionSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, finite projection table exposure, selector-ledger admission, endpoint handoff, terminal distance sealing, componentwise stability transport, displayed replay, provenance, local naming, and refusal of nearest-point choice, quotient metric equality, hidden endpoint selection, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 7761.1. Its public data are exactly $M, F, Q, E, R, H, C, P, N$, with the selector ledger Q downstream of the finite projection table F.

The endpoint handoff is inherited from the table row. By Definition 7348.3, the endpoint evidence in F is already a finite table read through the displayed metric and comparison rows, so Q can only point into that displayed table.

Transport, replay, provenance, and local naming preserve the same finite route. Since no coordinate stores a host nearest-point function, quotient metric equality, hidden endpoint selector, countable schedule, or ambient completeness theorem, the NameCert surface cannot export any of those objects. □

Theorem 7761.3 (Finite metric projection selector nonescape)**.** *Every consumer of an accepted* $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ *packet must read the parent* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *table row* F*, the selector ledger* Q*, the endpoint handoff* E*, the terminal seal* R*, and the structural rows* H, C, P, N *before using the selected projection candidate. The packet supplies a finite selector boundary for displayed candidates only; it exports no nearest-point choice, quotient metric equality, hidden endpoint selection, countable-choice schedule, or ambient completeness.*

Proof. Begin with the consumed metric source and finite projection table. The row M supplies the metric context, while F supplies the finite candidate-distance and endpoint evidence route of chapter 7348.

Read the selector ledger Q next. It chooses only among the displayed candidate rows of F, and the endpoint handoff E remains the endpoint evidence carried by the same finite table rather than a new projection principle.

Finally replay the terminal seal and structural rows. The route through R, H, C, P, N exposes the selected candidate as finite BEDC data and has no coordinate for a host nearest-point choice, quotient metric equality, hidden endpoint selection, countable-choice schedule, or ambient completeness. □

Falsifiable prediction: A $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ read is refuted whenever a projection consumer uses a selected candidate without displaying the MetricUp source, FiniteMetricProjectionUp table, selector ledger, endpoint handoff, RealUp seal, componentwise transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricProjectionSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$ packet is generated from a Metric source row, a FiniteMetricProjection table row, a finite selector ledger, an endpoint handoff, a terminal Real distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricProjectionSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7761.1, Definition 7761.2, and Definition 7761.3 seed the finite selector boundary from displayed metric projection tables to selected candidate reads.

Formal verification: FormalStatus(NameCert$_{\text{FiniteMetricProjectionSelector}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close nearest-point choice, quotient metric equality, hidden endpoint selection, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite projection table exposure, selector-ledger admission, endpoint handoff, terminal distance sealing, transport, replay, provenance, local naming, and refusal of choice principles for BEDC.Derived.FiniteMetricProjectionSelectorUp.

Chapter 7762

A Concrete Naming Certificate for $\mathsf{CompletionModulusFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and modulus data, the $\mathsf{CompletionModulusFunctor}^{\uparrow}$ packet records the finite handoff by which a completion witness is transported through displayed Cauchy-modulus evidence before a regular rational or Real consumer reads it. It sits over the finite completion window of chapter 1559, the directed-net completion route of chapter 3794, the uniform completion surface of chapter 1473, the explicit modulus row of chapter 292, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a functorial L10 handoff surface; it is not quotient completion equality, a representative selector, countable choice, host equality, open-cover compactness, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7762.1 (CompletionModulusFunctor carrier)**.** *A* $\mathsf{CompletionModulusFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, N, U, Q, R, E, H, C, P, L)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *finite-window completion row.*
2. *N is the* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *directed-net completion row read from the same displayed source.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *uniform-structure completion row.*
4. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *refinement row that names the visible modulus transport.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the completion rows and modulus row are displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read from the regular rational handoff.*
7. *H records componentwise* hsame *transport for completion rows, modulus rows, regular readback, Real sealing, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the finite completion source through the modulus, readback, and Real-seal route.*
9. *P is* Pkg *provenance over* $\mathsf{CompletionModulusFunctor}^{\uparrow}$, $\mathsf{MetricCompletionFinite}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, N, U, Q, R, E, H, C, P, L$. No coordinate stores quotient completion equality, selected representatives, countable-choice schedules, host equality, open-cover compactness, or ambient complete metric spaces.

Theorem 7762.2 (CompletionModulusFunctor NameCert obligations)**.** *For every accepted* CompletionModulusFun *carrier $F = (M, N, U, Q, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusFunctor}^{\uparrow}}$ *surface consists exactly of finite metric-completion admission, Cauchy-net completion admission, uniform-completion admission, Cauchy-modulus refinement exposure, regular rational readback, Real seal handoff, componentwise stability transport, displayed replay, provenance, local naming, and ledger refusal for quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, and ambient complete metric spaces.*

Proof. Project the carrier of Definition 7762.1. The completion-facing rows M, N, U supply the finite metric, directed-net, and uniform completion witnesses, while Q is the only displayed modulus-refinement row.

The readback row R and terminal seal E are downstream of those rows. Componentwise transport through H and replay through C keep the same route visible under hsame without adding hidden completion data.

The provenance and local naming rows P, L source and name that finite route. Since the carrier has no quotient, representative, choice, host-equality, compactness, or ambient-completeness coordinate, the local obligations cannot export those objects. □

Theorem 7762.3 (CompletionModulusFunctor L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CompletionModulusFunctor}^{\uparrow}$ *packet must read the displayed route*

$$\mathsf{MetricCompletionFinite}^{\uparrow}, \mathsf{CauchyNetCompletion}^{\uparrow}, \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Rea}$$

through $M, N, U, Q, R, E, H, C, P, L$. Thus the functorial handoff preserves the same downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal after componentwise* hsame *transport, while refusing quotient completion equality, representative selection, countable choice, host equality, open-cover compactness, and ambient complete metric spaces.*

Proof. Start with an accepted packet $F = (M, N, U, Q, R, E, H, C, P, L)$. The completion rows M, N, U are displayed before the modulus row Q, so a downstream consumer cannot read a modulus-refined completion without naming its finite, net, and uniform completion sources.

Replay the route through C and transport it through H. The regular rational readback R and terminal Real seal E remain attached to the same displayed completion-modulus path after componentwise hsame transport.

Finally inspect the ledger recorded by the carrier. The route contains no coordinate for quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, or ambient complete metric spaces, so the L10 handoff is exactly the finite certificate surface named by $M, N, U, Q, R, E, H, C, P, L$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionModulusFunctor}^{\uparrow}$ packet is generated from MetricCompletionFinite, CauchyNetCompletion, UniformCompletion, CauchyModulus, RegSeqRat, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompletionModulusFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 7762.1, Definition 7762.2, and Definition 7762.3 seed the finite functorial handoff from completion witnesses through modulus rows to regular rational and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompletionModulusFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, ambient complete metric spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-row admission, modulus refinement exposure, regular readback, Real-seal handoff, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.CompletionModulusFunctorUp.

Chapter 7763

A Concrete Naming Certificate for FiniteProofDagFrontier$^{\uparrow}$

The FiniteProofDagFrontier$^{\uparrow}$ packet realizes a MetaCIC open-problem route as finite proof-DAG bookkeeping. It sits beside the critical-path surface of chapter 765, the public residual frontier of chapter 4238, and the normal-form consistency seal of chapter 740. The packet is a finite frontier carrier for nodes, edges, checker rows, normalization-frontier rows, obstructions, transport, replay, provenance, and local naming; it does not export subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, or kernel consistency.

Definition 7763.1 (Finite proof-DAG frontier carrier)**.** *A* FiniteProofDagFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (V, A, K, F, O, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite node row, recording the proof objects, typing windows, and local source names inspected by the packet.*
2. *A is the finite edge row, recording premise-to-conclusion dependencies and residual links between displayed proof-DAG nodes.*
3. *K is the checker row, recording the bounded local verifier that reads only the displayed nodes and edges.*
4. *F is the normalization-frontier row, recording candidate normal endpoints, unfinished normalization windows, and the boundary between them.*
5. *O is the obstruction row, retaining absent subject-reduction, residual-substitution, global-normalization, evaluator-equality, quotient-proof-equality, and kernel-consistency endpoints.*
6. *H records componentwise* hsame *transport for nodes, edges, checker rows, normalization frontiers, obstructions, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay through the proof-DAG node, edge, checker, frontier, and obstruction route.*
8. *P is* Pkg *provenance over* FiniteProofDagFrontier$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteProofDagFrontier}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $V, A, K, F, O, H, C, P, N$. No coordinate stores subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, or kernel consistency.

Theorem 7763.2 (Finite proof-DAG frontier NameCert obligations)**.** *For every accepted* $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ *carrier $D = (V, A, K, F, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteProofDagFrontier}^{\uparrow}}$ *surface consists exactly of finite node admission, finite edge admission, bounded checker exposure, normalization-frontier exposure, obstruction retention, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, and kernel consistency.*

Proof. Project the carrier of Definition 7763.1. The node and edge rows V, A are the only proof-DAG shape data available to the packet, while K is the only checker row and F is the only normalization-frontier row.

The obstruction row O records the endpoints that remain absent. Transport through H and replay through C preserve the displayed DAG route without adding subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, or kernel consistency.

The provenance and local naming rows P, N source and name the same finite route. Hence the local obligations are exactly the displayed node, edge, checker, frontier, obstruction, transport, replay, provenance, and naming rows. □

Theorem 7763.3 (Finite proof-DAG frontier non-escape)**.** *An accepted* $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ *packet exports no subject reduction, no unrestricted confluence, no evaluator equality, no quotient proof equality, no global normalization, no residual-substitution discharge, and no kernel consistency. Every MetaCIC consumer of the packet must read only the displayed route*

$$V, A \longmapsto K, F, O \longmapsto H, C, P, N.$$

Proof. Begin with the carrier inventory. The packet lists finite nodes, finite edges, a bounded checker, a normalization frontier, an obstruction row, transport, replay, provenance, and local naming.

Now inspect the consumer route. Any downstream read must pass through V, A before reaching the checker and frontier rows K, F, and the obstruction row O remains attached to the same route.

Since the carrier has no additional coordinate beyond $V, A, K, F, O, H, C, P, N$, none of subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, or kernel consistency can be projected from an accepted packet. □

Falsifiable prediction: A $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ read is refuted whenever a MetaCIC consumer treats a finite proof-DAG frontier as subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, or kernel consistency without displaying the node, edge, checker, normalization-frontier, obstruction, transport, replay, provenance, and local naming rows.

Independence witness: metaciccriticalpath, metacic-parallel-diamond-frontier, metacicnormalizationfrontier, normalformconsistencyseal: none is bijective with $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ because

the packet jointly records finite nodes, finite edges, a bounded checker, a normalization frontier, obstruction retention, transport, replay, provenance, and local naming.

The Lean follow-up path is `lean4/BEDC/Derived/FiniteProofDagFrontierUp/TasteGate.lean`. It must provide explicit Nontrivial $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ and FieldFaithful $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ instances preserving node, edge, checker, normalization-frontier, obstruction, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteProofDagFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteProofDagFrontier}^{\uparrow}$ packet is generated from finite proof-DAG nodes, finite edges, bounded checker rows, normalization-frontier rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteProofDagFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7763.1, Definition 7763.2, and Definition 7763.3 seed the finite proof-DAG frontier route for MetaCIC bookkeeping consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteProofDagFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, unrestricted confluence, evaluator equality, quotient proof equality, global normalization, residual-substitution discharge, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked node admission, edge admission, bounded checker exposure, normalization-frontier exposure, obstruction retention, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.FiniteProofDagFrontierUp.

Chapter 7764

A Concrete Naming Certificate for $\mathsf{HyperbolicLimitSet}^{\uparrow}$

Following Beardon and Bridson–Haefliger, the $\mathsf{HyperbolicLimitSet}^{\uparrow}$ packet records the finite boundary accumulation surface of a displayed hyperbolic action. It sits after the visual boundary route of chapter 4181, the boundary-shadow route of chapter 3813, the Busemann boundary route of chapter 3732, the visual metric route of chapter 3866, the geodesic-flow route of chapter 3856, the Patterson–Sullivan shadow route of chapter 6070, the axis-shadow route of chapter 6122, and the terminal disk and real seals of chapter 377 and chapter 38. The packet names a finite limit-set read; it does not complete the ideal boundary, quotient geodesic rays, select orbit representatives, or import host compactness.

Definition 7764.1 (Hyperbolic limit set carrier)**.** *A* $\mathsf{HyperbolicLimitSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, S, U, V, G, P, A, D, R, H, C, Q, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *boundary readback row.*
2. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *shadow row recording the finite orbit-shadow observations.*
3. *U is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row attaching the boundary read to Busemann-level comparison data.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row comparing boundary observations by displayed visual metric windows.*
5. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row naming the finite orbit or geodesic-flow observations whose accumulation is being read.*
6. *P is the Patterson–Sullivan shadow row, used only as a displayed weighting and shadow-support surface.*
7. *A is the* $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ *row for axial shadow support when the orbit read comes from a displayed axis route.*
8. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *source row for disk-coordinate admission.*
9. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the finite visual-radius and shadow-size reads.*

10. *H records componentwise* hsame *transport for boundary, shadow, Busemann, visual-metric, flow, Patterson–Sullivan, axis, disk, real-seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay through the same boundary and shadow route.*
12. *Q is the* Pkg *provenance over* $\mathsf{HyperbolicLimitSet}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicAxisShadow}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed boundary compactification, quotient boundary equality, selected orbit representative, global group action, host compactness theorem, or ambient hyperbolic space.

Theorem 7764.2 (Hyperbolic limit set NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicLimitSet}^{\uparrow}$ *carrier*

$$L = (B, S, U, V, G, P, A, D, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$ *surface consists exactly of visual-boundary readback, boundary-shadow exposure, Busemann-boundary attachment, visual-metric comparison, geodesic-flow observation, Patterson–Sullivan shadow support, axis-shadow support, disk-coordinate admission, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7764.1. Rows B, S, U, V are the boundary-facing rows, while G, P, A, D, R record the displayed source observations, shadow support, disk admission, and finite real-valued reads.

The boundary route is read first. The visual boundary row supplies the boundary token, the boundary-shadow row supplies finite orbit-shadow observations, the Busemann row attaches the read to boundary comparison data, and the visual metric row supplies the displayed comparison window.

The source and support rows then constrain the accumulation read. Geodesic-flow observations, Patterson–Sullivan shadow support, axis-shadow support, disk-coordinate admission, and the terminal Real seal remain finite fields of the same packet rather than a completed boundary object.

Finally H, C, Q, N transport, replay, source, and name exactly that packet. Since no completed compactification, quotient equality, selected representative, global action, host compactness theorem, or ambient hyperbolic space coordinate appears in L, the listed rows exhaust the NameCert obligation surface. □

Theorem 7764.3 (Hyperbolic limit set boundary non-escape)**.** *Every accepted* $\mathsf{HyperbolicLimitSet}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicVisualBoundary}^{\uparrow},\ \mathsf{HyperbolicBoundaryShadow}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow},\ \mathsf{HyperbolicVisualMetric}^{\uparrow},$$

before it exports a limit-set boundary token. The read may use Patterson–Sullivan shadow support as a displayed finite weighting row, but it cannot replace the packet by a completed ideal boundary, quotient of geodesic rays, selected orbit representative, global group action, host compactness theorem, or ambient hyperbolic space.

Proof. Start with Definition 7764.2. It orders the accepted read through the boundary, shadow, Busemann, visual-metric, flow, support, disk, real-seal, transport, replay, provenance, and local naming rows.

The boundary rows B, S, U, V keep the limit-set token attached to displayed finite boundary observations. The source rows G, A, D keep the orbit or axis read tied to finite geodesic-flow, axis-shadow, and disk-coordinate data.

The Patterson–Sullivan row P and Real seal R are downstream support rows. They can weight and seal the displayed shadow data, but they do not add quotient boundary equality, selected representatives, global action data, compactness, or ambient geometry.

Thus the only exported token is the local $\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$ boundary read named by N over the same packet. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicLimitSet}^{\uparrow}$ read exposes visual-boundary, boundary-shadow, Busemann-boundary, visual-metric, geodesic-flow, Patterson–Sullivan support, axis-shadow, disk, Real-seal, transport, replay, provenance, and local NameCert rows before any limit-set boundary token is exported.

Independence witness: Nearest siblings: hyperbolic-visual-boundary, hyperbolic-boundary-shadow, hyperbolic-busemann-boundary, hyperbolic-visual-metric, hyperbolic-geodesic-flow, hyperbolic-patterson-sullivan-shadow, and hyperbolic-axis-shadow. None is bijective to $\mathsf{HyperbolicLimitSet}^{\uparrow}$ because this packet jointly carries boundary readback, orbit-shadow support, Busemann comparison, visual metric comparison, flow observation, weighting support, axis support, disk admission, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicLimitSet}^{\uparrow}$ packet is generated from visual-boundary rows, boundary-shadow rows, Busemann-boundary rows, visual-metric rows, geodesic-flow observations, Patterson–Sullivan shadow support, axis-shadow support, UnitDisk admission, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$, seed)

Scope closed: Definition 7764.1, Definition 7764.2, and Definition 7764.3 seed the finite limit-set boundary read over visual-boundary, shadow, Busemann, visual-metric, flow, support, disk, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicLimitSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed boundary compactification, quotient boundary equality, selected orbit representatives, global group actions, host compactness, ambient hyperbolic spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary readback, shadow exposure, Busemann attachment, visual-metric comparison, geodesic-flow observation, support rows, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicLimitSetUp.

Chapter 7765

A Concrete Naming Certificate for $\mathsf{ProperMetricGroupAction}^{\uparrow}$

Following Bourbaki and Bridson–Haefliger on group actions on proper metric spaces, the $\mathsf{ProperMetricGroupAction}^{\uparrow}$ packet records a displayed group action by isometries on a proper metric carrier as finite BEDC NameCert data. It consumes the group source of chapter 41, the metric source of chapter 57, and the closed-ball compactness discipline of **??**. The packet names only action rows, orbit-window rows, compact ball witnesses, transport, replay, provenance, and local naming; it does not assert a host quotient orbit space, a global choice of representatives, or a full geometric group theory package.

Definition 7765.1 (Proper metric group action carrier)**.** *A* $\mathsf{ProperMetricGroupAction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, M, P, \alpha, I, O, W, H, C, Q, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *source row, including the displayed carrier, multiplication, unit, inverse, and classifier rows used to index action words.*
2. *M is the consumed* $\mathsf{Met}^{\uparrow}$ *source row for the acted-on points, distance observations, and metric-law ledger.*
3. *P is the consumed* $\mathsf{ProperMetric}^{\uparrow}$ *row, exposing compact closed-ball witnesses for the finite center-radius windows read by the action packet.*
4. *α is the action endpoint ledger. For each displayed group row g and point row x, it records the endpoint row $\alpha(g, x)$ as a carried metric point.*
5. *I is the isometry ledger, recording that each displayed action endpoint preserves the metric-distance rows inspected by the packet.*
6. *O is the finite orbit-window ledger, listing the displayed group words and point rows inspected by a downstream consumer.*
7. *W is the properness and cocompactness window ledger. It records compact closed-ball witnesses from P and finite orbit-window coverage rows for the displayed windows only.*
8. *H records componentwise* hsame *transport for the group, metric, proper-metric, action, isometry, orbit-window, coverage-window, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from group multiplication and inverse rows through action endpoints, isometry comparisons, compact closed-ball witnesses, and finite coverage windows.*
10. *Q is the* Pkg *provenance over* $\mathsf{ProperMetricGroupAction}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProperMetricGroupAction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $G, M, P, \alpha, I, O, W, H, C, Q, N$. It has no coordinate for quotient orbit-space equality, selected orbit representatives, an ambient isometry group, open-cover compactness, or a global geometric group theory structure.

Theorem 7765.2 (Proper metric group action NameCert obligations)**.** *For every accepted* ProperMetricGroupActio *carrier $A = (G, M, P, \alpha, I, O, W, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProperMetricGroupAction}^{\uparrow}}$ *surface consists exactly of carrier admission, group-source exposure, metric-source exposure, proper-metric closed-ball exposure, action endpoint admission, isometry-ledger admission, finite orbit-window admission, properness and cocompactness window admission, componentwise transport, displayed replay, provenance, local naming, and refusal of quotient orbit spaces, global representative choice, and full geometric group theory.*

Proof. Project the carrier of Definition 7765.1. Its public rows are exactly $G, M, P, \alpha, I, O, W, H, C, Q, N$, with the group and metric inputs read before any action endpoint is accepted.

The action endpoint and isometry ledgers are finite rows. The endpoint row $\alpha(g, x)$ is inspected only for displayed group rows and point rows, and I compares only the metric-distance observations already exposed by M.

The properness and cocompactness boundary is the window ledger W. It may read compact closed-ball witnesses from P and the displayed orbit-window rows from O, but it has no field that selects orbit representatives or forms a host quotient space.

Transport, replay, provenance, and local naming preserve the same finite surface. Since no other row is present in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 7765.3 (Proper and cocompact finite-window handoff)**.** *Every properness-facing or cocompactness-facing consumer of an accepted* $\mathsf{ProperMetricGroupAction}^{\uparrow}$ *packet factors through*

$$G, \quad M, \quad P, \quad \alpha, \quad I, \quad O, \quad W, \quad H, \quad C, \quad Q, \quad N.$$

For each displayed orbit window, the consumer first reads the $\mathsf{Group}^{\uparrow}$ *word rows, the* $\mathsf{Met}^{\uparrow}$ *point and distance rows, the* $\mathsf{ProperMetric}^{\uparrow}$ *closed-ball compactness witness, the action endpoint ledger, and the isometry ledger; only then may it read the finite coverage row in W. The handoff therefore supplies finite window data for proper and cocompact action reads, not a quotient orbit space, a selected representative system, or a general geometric group theory theorem.*

Proof. Fix a displayed orbit window. The group rows in G supply the finite words being tested, and the metric rows in M supply the point and distance observations for the same window.

Read the proper-metric row P next. The compact closed-ball witness used by the window is one of the displayed proper-metric witnesses, so the route is inherited from **??** and not from an open-cover compactness principle.

The action ledger α and isometry ledger I then attach the group words to the metric rows. The orbit-window ledger O records which endpoint rows are inspected, and W records the finite properness or cocompactness coverage row attached to that window.

Finally use H, C, Q, N to transport, replay, source, and name the same finite packet. None of these rows contains a quotient orbit-space carrier, a representative selector, or a global geometric group theory structure, so the consumer receives only the displayed finite-window handoff. □

Falsifiable prediction: A $\mathsf{ProperMetricGroupAction}^{\uparrow}$ read is refuted whenever a properness-facing or cocompactness-facing consumer uses orbit coverage without first displaying the GroupUp source, MetricUp source, ProperMetricUp closed-ball witness, action endpoint ledger, isometry ledger, finite orbit-window ledger, componentwise transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ProperMetricGroupAction}^{\uparrow}}$.

Constructive story: A $\mathsf{ProperMetricGroupAction}^{\uparrow}$ packet is generated from a Group source row, a Metric source row, a ProperMetric closed-ball compactness row, an action endpoint ledger, an isometry ledger, a finite orbit-window ledger, a properness and cocompactness window ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProperMetricGroupAction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7765.1, Definition 7765.2, and Definition 7765.3 seed the finite action carrier, its NameCert obligation surface, and the properness and cocompactness finite-window handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProperMetricGroupAction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host quotient orbit space, global representative choice, full geometric group theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, group and metric source exposure, proper-metric closed-ball exposure, action endpoint admission, isometry admission, orbit-window admission, coverage-window admission, transport, replay, provenance, local naming, and refusal of quotient-space and representative-choice principles for BEDC.Derived.ProperMetricGroupActionUp.

Chapter 7766

A Concrete Naming Certificate for $\mathsf{BishopLocatedSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, a located sequence is a displayed real-name sequence whose finite windows, locatedness moduli, and regular rational readback are visible before any completion-facing consumer may inspect a limiting claim. The $\mathsf{BishopLocatedSequence}^{\uparrow}$ packet records this L10 neighbour of constructive Real and Completion objects. It sits beside the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the terminal Real seal of chapter 38, and the located-space route of chapter 7760. The certificate names only the finite sequence surface; it does not select a limit, quotient streams, invoke countable choice, compare by trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7766.1 (Bishop located sequence carrier)**.** *A* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$X = (S, L, R, E, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of finite sequence windows. Each consumer request names a displayed prefix or tolerance-indexed observation window before reading any Real-facing data.*
2. *L is the locatedness-modulus row. It records the finite modulus by which the requested window is witnessed as located in the source sequence surface.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It packages the rational comparison data extracted from the sequence windows and locatedness modulus.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is exposed only after S, L, R have been displayed.*
5. *H stores componentwise* hsame *transport for S, L, R, E and for the structural certificate rows.*
6. *C stores displayed* Cont *replay from stream windows through locatedness moduli, regular rational readback, and the terminal Real seal.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSequence}^{\uparrow}}$ *row.*

The classifier compares packets only through S, L, R, E, H, C, P, N. It has no coordinate for a selected limit, quotient stream equality, countable choice, real trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedSequenceUp.BishopLocatedSequenceTasteGate single carrier alignment`

Theorem 7766.2 (BishopLocatedSequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *carrier*

$$X = (S, L, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedSequence}^{\uparrow}}$ *surface is exhausted by four obligation rows:*

1. *stream-window admission, requiring every sequence read to begin at the displayed* $\mathsf{StreamName}^{\uparrow}$ *row* S*;*
2. *locatedness-modulus visibility, requiring the row* L *to be visible before any located-sequence or* $\mathsf{LocatedSpace}^{\uparrow}$ *consumer reads the sequence as located data;*
3. *regular rational and Real readback, requiring* R *to expose the rational comparison windows before* E *seals the read as* $\mathsf{Real}^{\uparrow}$ *data;*
4. *structural replay, requiring* H, C, P, N *to preserve the same finite route under transport, continuation replay, provenance, and local naming.*

No obligation row may call on a selected limit, quotient stream map, countable choice, trichotomy, host equality of sequences, or ambient Real completeness.

Proof. Carrier admission is projection from Definition 7766.1. The only analytic rows are S, L, R, E, and the only structural rows are H, C, P, N.

The first two obligations bind the sequence source. A consumer starts at finite $\mathsf{StreamName}^{\uparrow}$ windows and can treat the sequence as located data only after the modulus row L is displayed.

The third obligation routes numerical data through R and E. Regular rational readback supplies the finite comparisons, and the Real seal is downstream of those comparisons.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the excluded principles have no carrier coordinate, these four rows exhaust the local NameCert surface. □

Theorem 7766.3 (BishopLocatedSequence window stability)**.** *Every accepted* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *packet keeps finite window reads stable under componentwise* hsame *transport: if a consumer reads the row* S *and then uses the locatedness modulus* L*, regular rational readback* R*, and Real seal* E*, the transported read must expose the same route*

$$S, L, R, E, H, C, P, N.$$

The transported consumer receives a finite located sequence surface and not a quotient stream, selected limit, countable-choice witness, trichotomy decision, or ambient Real-completeness theorem.

Proof. Read H in Definition 7766.1. It is componentwise transport for the stream windows, locatedness modulus, regular rational readback, and Real seal.

Replay the consumer route through C. The replay row can inspect S before L, then R, then E, so transport preserves the displayed order of the sequence read.

The provenance and local name rows P, N keep the transported read attached to the same packet. Since no quotient, selected-limit, choice, trichotomy, or completeness coordinate is present, window stability remains a finite NameCert statement. □

Theorem 7766.4 (BishopLocatedSequence L10 route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *carrier must traverse the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow L \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

as displayed by S, L, R, E, H, C, P, N. The same route may be compared with the located-space sibling chapter 7760, but it does not supply selected limits, quotient streams, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7766.2. They force every public read through stream-window admission, locatedness-modulus visibility, regular rational readback, and the Real seal.

Use window stability from Definition 7766.3. Transport and replay preserve the same displayed rows, so a downstream Real or Completion consumer cannot bypass the finite sequence windows.

The sibling comparison with $\mathsf{LocatedSpace}^{\uparrow}$ uses only the locatedness row L and the shared structural rows H, C, P, N. It is therefore a finite L10 route over StreamName, RegSeqRat, Real, and LocatedSpace data, not a claim of selected convergence or ambient completeness. □

Falsifiable prediction: Any accepted $\mathsf{BishopLocatedSequence}^{\uparrow}$ consumer that reaches a Real-facing sequence read while bypassing the StreamName row, locatedness modulus, RegSeqRat readback, Real seal, transport, replay, provenance, or local NameCert row refutes the carrier surface.

Independence witness: streamname, regseqrat, real, locatedspace: $\mathsf{BishopLocatedSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite sequence windows, a locatedness-modulus row, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local NameCert registration as one Bishop-style located-sequence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedSequence}^{\uparrow}$ packet is generated from StreamName windows, a locatedness-modulus row, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7766.1, Definition 7766.2, Definition 7766.3, and Definition 7766.4 seed the finite located-sequence route over StreamName, RegSeqRat, Real, and LocatedSpace rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedSequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BishopLocatedSequenceUp.BishopLocatedSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, locatedness-modulus visibility, regular rational readback, Real-seal nonescape, transport replay, and local NameCert coverage for BEDC.Derived.BishopLocatedSequenceUp.

Chapter 7767

A Concrete Naming Certificate for $\mathsf{CylinderObject}^{\uparrow}$

Following Quillen's model-category cylinder objects, the $\mathsf{CylinderObject}^{\uparrow}$ packet records the finite BEDC row that factors the fold map from two displayed copies of one object through a weak-equivalence face. It sits after the model-category carrier of chapter 158, the category dependency of chapter 61, the cofibrant-replacement witness of chapter 308, and the homotopy ledger of chapter 156. The packet names one displayed cylinder for one object; it does not export functorial cylinders, mapping cylinders, homotopy-category quotients, derived mapping spaces, or a global replacement choice.

Definition 7767.1 (Cylinder object carrier)**.** *A* $\mathsf{CylinderObject}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M, X, I_0, I_1, C, W, F, H, R, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{ModelCat}^{\uparrow}$ *row supplying the category, cofibration, fibration, weak-equivalence, lifting, and factorization surfaces.*
2. *X is the displayed object row over the* $\mathsf{Cat}^{\uparrow}$ *surface of M.*
3. *I_0 and I_1 are the two endpoint inclusion rows from X into the candidate cylinder row.*
4. *C is the cylinder-object row through which the fold request on the two copies of X factors.*
5. *W is the weak-equivalence ledger marking the projection from C back to the displayed object row.*
6. *F is the finite factorization row read from the* $\mathsf{ModelCat}^{\uparrow}$ *factorization surface.*
7. *H is the* $\mathsf{Homotopy}^{\uparrow}$ *ledger that records the homotopy-facing interpretation of the displayed weak-equivalence row.*
8. *R is the* $\mathsf{CofibrantReplacement}^{\uparrow}$ *dependency row when the object is read through a displayed cofibrant witness.*
9. *P is the* Pkg *provenance over* $\mathsf{CylinderObject}^{\uparrow}$, $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{CofibrantReplacement}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CylinderObject}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, X, I_0, I_1, C, W, F, H, R, P, N$. It has no coordinate for a functorial cylinder assignment, mapping-cylinder construction, host homotopy category, derived mapping space, quotient weak-equivalence calculus, or global replacement choice.

Lean4 (statement) `BEDC.Derived.CylinderObjectUp`

Theorem 7767.2 (Cylinder object NameCert obligations)**.** *For every accepted* $\mathsf{CylinderObject}^{\uparrow}$ *carrier* $Z = (M, X, I_0, I_1, C, W, F, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CylinderObject}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *endpoint admission, requiring I_0 and I_1 to be displayed as the two copies of X entering the cylinder row C;*
2. *model-category factorization, requiring F to read the factorization surface supplied by M before C can be consumed;*
3. *weak-equivalence projection, requiring W to mark the projection from C back to X inside the weak-equivalence row of M;*
4. *homotopy and cofibrant-replacement compatibility, requiring H and R to be read only as the displayed homotopy and cofibrant-witness dependencies of the same packet;*
5. *ledger non-escape, requiring P and N to source and name only the displayed rows, with no functorial cylinder assignment, mapping-cylinder construction, homotopy-category quotient, derived mapping space, quotient weak-equivalence calculus, or global replacement choice.*

Proof. Project the carrier of Definition 7767.1. The endpoint rows I_0 and I_1 are stored as displayed inputs to C, so a cylinder read starts from the two named copies of X rather than from an ambient coproduct or host fold map.

Next read the model-category row M and its factorization row F. These rows supply the finite factorization surface on which C is admitted and prevent the certificate from importing an external weak-equivalence calculus.

The weak-equivalence ledger W and homotopy ledger H are consumed only after the cylinder row is visible. The cofibrant-replacement row R is a dependency row for the displayed object witness, not a global replacement operator.

Finally inspect P and N. They source and name the same finite packet, and none of the refused objects appears as a carrier coordinate. The five displayed obligation families therefore exhaust the local NameCert surface. □

Theorem 7767.3 (Cylinder object factorization route)**.** *Every downstream homotopy or cofibrant-replacement consumer citing an accepted* $\mathsf{CylinderObject}^{\uparrow}$ *carrier must factor its read through*

$$M, X, I_0, I_1, C, W, F, H, R, P, N.$$

The route exports one displayed cylinder object for one displayed object and does not export a functorial cylinder construction, mapping-cylinder construction, homotopy-category quotient, derived mapping space, quotient weak-equivalence calculus, or global replacement choice.

Proof. Begin with the endpoint and cylinder rows of Definition 7767.1. A consumer reaches the cylinder row C only after the object row X and the two endpoint inclusions I_0, I_1 have been displayed.

Use the obligation surface of Definition 7767.2. The model-category factorization row F and weak-equivalence ledger W bind the cylinder read to M, while the homotopy and cofibrant-replacement rows H and R keep the homotopy-facing and cofibrant-facing uses inside the same finite packet.

The provenance and local naming rows P, N preserve that route for downstream consumers. Since the carrier lacks every refused coordinate, the consumer receives only the displayed cylinder route and cannot promote it to a functorial, quotient, derived, or global-choice construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{CylinderObject}^{\uparrow}}$.

Constructive story: A $\mathsf{CylinderObject}^{\uparrow}$ packet is generated from ModelCat, Category, endpoint inclusion, cylinder, weak-equivalence, factorization, Homotopy, CofibrantReplacement, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CylinderObject}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7767.1, Definition 7767.2, and Definition 7767.3 seed the finite cylinder-object route over ModelCat, Category, CofibrantReplacement, and Homotopy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CylinderObject}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CylinderObjectUp`

Bridge status: none

Not claimed: This chapter does not close functorial cylinders, mapping cylinders, homotopy-category quotients, derived mapping spaces, global replacement choices, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, model-category factorization, weak-equivalence projection, homotopy and cofibrant-replacement compatibility, ledger non-escape, provenance, and local naming for BEDC.Derived.CylinderObjectUp.

Chapter 7768

A Concrete Naming Certificate for $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ packet records the constructive Banach fixed-point route as finite located-completion data. It sits after the compact interval fixed-point packet of chapter 1437, the located metric surface of chapter 1533, the finite stream-window boundary of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a contraction row and a finite Picard-window handoff inside a displayed complete metric source; it does not claim general Brouwer fixed points, trichotomy of constructive reals, quotient stream equality, a choice-selected limit, or host completeness.

Definition 7768.1 (Bishop located Banach fixed-point carrier)**.** *A* $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, L, K, X, W, S, Q, Cm, E, H, R, P, N)$$

with the following displayed rows:

1. *M is the consumed* $\mathsf{LocatedMetric}^{\uparrow}$ *row naming the distance reads, located comparison windows, and metric transport surface for the source.*
2. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row used only as a displayed regular-Cauchy landing discipline, not as host completeness.*
3. *K is the contraction row, containing the map graph, a rational Lipschitz bound below one, and the finite contraction-read ledger.*
4. *X is the displayed initial seed row for the Picard iteration.*
5. *W is the finite Picard-window row listing the iterates inspected at the requested dyadic tolerance.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that exposes those Picard windows to downstream consumers.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the window schedule into regular rational approximants.*
8. *Cm is the contraction-Cauchy modulus row, tying the rational contraction bound to the Picard-window tail estimate.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the named fixed-point value, exposed only after W, S, Q, Cm have been displayed.*

10. *H records componentwise* hsame *transport for $M, L, K, X, W, S, Q, Cm, E$ and the local certificate row.*
11. *R records displayed* Cont *replay from a fixed-point request through the contraction, Picard windows, stream schedule, regular readback, modulus, and terminal seal.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite contraction-and-Picard certificate. It has no coordinate for a host fixed point, selected completed limit, quotient equality of Cauchy streams, countable choice, real trichotomy, or ambient completeness.

Theorem 7768.2 (Bishop located Banach fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ *carrier $T = (M, L, K, X, W, S, Q, Cm, E, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}}$ *obligations are exhausted by carrier admission, located-metric distance exposure, displayed completeness-boundary admission, contraction-row exactness, initial-seed exposure, finite Picard-window construction, stream scheduling, regular-rational readback, contraction-Cauchy modulus sealing, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming. No obligation may call on general Brouwer fixed points, trichotomy, quotient stream equality, countable choice, a selected host limit, or host completeness.*

Proof. Project the carrier of Definition 7768.1. The only analytic rows are the located metric source M, the displayed completion boundary L, the contraction row K, the seed row X, the Picard windows W, the stream schedule S, the regular readback Q, and the contraction-Cauchy modulus Cm.

The Picard route is finite at every read. The contraction row supplies the rational contraction ledger, the seed row starts the displayed iteration, and the window row lists the iterates inspected before any readback is possible.

The completion-facing handoff then passes through S, Q, Cm, E in order. Transport, replay, provenance, and local naming preserve the same rows, so no excluded host fixed-point, selected-limit, quotient, trichotomy, choice, or ambient-completeness principle can enter the NameCert surface. □

Theorem 7768.3 (Bishop located Banach fixed-point Picard-window handoff)**.** *Every consumer of an accepted* $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ *packet must read the contraction row K, the initial seed X, the finite Picard-window row W, the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the contraction-Cauchy modulus Cm, and the* $\mathsf{Real}^{\uparrow}$ *seal E before treating the output as a fixed-point value. The packet therefore exports a Picard-window handoff for a displayed located complete metric source, not a host-level fixed-point operator.*

Proof. Begin with the fixed-point request. The route first consumes the located metric row M and displayed completion boundary L, then reads the contraction row K and initial seed X before any Picard data are available.

The row W exposes the finite Picard windows requested by the tolerance. These windows are scheduled by S and translated by Q into regular rational readback, while Cm records the contraction tail estimate that licenses the terminal seal.

Only after that chain is displayed may the consumer read E. Since H, R, P, N merely transport, replay, record provenance, and name the same route, the packet has no bypass through a selected limit, host completeness theorem, or general Brouwer fixed-point principle. □

Falsifiable prediction: A $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ consumer is refuted whenever it reaches a fixed-point seal without displaying the contraction row, seed row, Picard windows, StreamName schedule, RegSeqRat readback, contraction-Cauchy modulus, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}$ packet is generated from located metric data, a displayed complete-metric boundary, a contraction row, an initial seed, finite Picard windows, StreamName scheduling, RegSeqRat readback, a contraction-Cauchy modulus, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7768.1, Definition 7768.2, and Definition 7768.3 seed the constructive contraction fixed-point route as finite located-completion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedBanachFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Brouwer fixed points, real trichotomy, quotient stream equality, countable choice, selected host limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-row exactness, finite Picard-window construction, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedBanachFixedPoi

Chapter 7769

A Concrete Naming Certificate for $\mathsf{FiniteCoverMeshChain}^{\uparrow}$

Following Bishop and Bridges on finite-cover proofs of Heine–Cantor, the $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ packet records the mesh-chain route from compact finite-cover data to a tolerance-only uniform-modulus handoff. It sits over the compact source of chapter 60, continuous-map rows of chapter 130, the metric source of chapter 57, rational lower-bound reads from chapter 37, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names only finite cover rows, dyadic mesh bounds, star-refinement transport, rational lower-bound folds, and uniform-modulus handoff data.

Definition 7769.1 (Finite cover mesh-chain carrier)**.** *A* $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, G, B, D, S, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the consumed* $\mathsf{Compact}^{\uparrow}$ *finite-cover source row, exposed as finite* $\mathsf{ProbeBundle}^{\uparrow}$ *cover data.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row used for mesh-radius comparisons and triangle transport.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the displayed compact source.*
4. *B is the dyadic mesh-bound row, recording the finite mesh scale and cover-radius comparisons.*
5. *D is the finite cover-chain ledger connecting each mesh cell to a displayed cover member.*
6. *S is the star-refinement transport row, carrying local pointwise moduli along neighbouring mesh cells.*
7. *L is the* $\mathsf{Rat}^{\uparrow}$ *rational lower-bound fold over the finitely many displayed radii.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed only after B, D, S, L have been read.*
9. *H records componentwise* hsame *transport for K, M, G, B, D, S, L, U and the local certificate row.*
10. *C records displayed* Cont *replay from compact-cover reads through mesh bounds, refinement transport, rational folding, and uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{ProbeBundle}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverMeshChain}^{\uparrow}}$ *registration row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for open-cover compactness, a host Lebesgue number, semantic continuity, quotient metric equality, selected centres outside the displayed cover, or an external modulus oracle.

Theorem 7769.2 (Finite cover mesh-chain NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ *carrier* $F = (K, M, G, B, D, S, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverMeshChain}^{\uparrow}}$ *obligations are exactly compact cover exposure, metric source exposure, continuous-map row admission, dyadic mesh-bound admission, finite cover-chain routing, star-refinement transport, rational lower-bound folding, UniformModulus handoff, componentwise stability transport, displayed replay, provenance, and local naming. The obligations do not include open-cover compactness, a host Lebesgue number, semantic continuity, quotient metric equality, hidden centre selection, or an external modulus oracle.*

Proof. Project the carrier of Definition 7769.1. Its compact-uniform data are the finite cover row K, metric source M, continuous-map row G, mesh bound B, cover chain D, star-refinement row S, rational fold L, and uniform-modulus handoff U.

The finite-cover route is ordered. Mesh bounds are inspected before cover-chain transport, star refinement moves only along the displayed finite chain, and the rational lower-bound fold reads only the finitely many radii exposed by that chain.

The UniformModulus handoff is downstream of those rows. Transport, replay, provenance, and local naming preserve the same displayed route, so none of the excluded host compactness, semantic-continuity, quotient, hidden-selection, or oracle principles can enter the NameCert surface. □

Theorem 7769.3 (Finite cover mesh-chain UniformModulus handoff)**.** *Every consumer of an accepted* $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ *packet must read the compact finite-cover row* K*, metric row* M*, continuous-map row* G*, dyadic mesh bound* B*, cover-chain ledger* D*, star-refinement transport* S*, rational lower-bound fold* L*, and* $\mathsf{UniformModulus}^{\uparrow}$ *handoff* U *before using the exported modulus. Thus the packet realizes the finite-cover mesh route as concrete NameCert data rather than as a host compactness theorem.*

Proof. Start with the compact and metric sources. The row K supplies finite cover data, M supplies mesh comparisons, and G supplies the pointwise-modulus rows that will be transported.

The mesh chain is then read through B, D, S. These rows expose the dyadic mesh scale, the finite cover member attached to each mesh cell, and the star-refinement transport that keeps local moduli valid along neighbouring cells.

Finally the rational lower-bound fold L produces the tolerance-only value consumed by U. The rows H, C, P, N preserve and name this same finite path, so the exported modulus has no bypass through open-cover compactness, a host Lebesgue number, hidden centre selection, quotient metric equality, or an external oracle. □

Falsifiable prediction: A $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ consumer is refuted whenever a uniform-modulus read bypasses the compact cover row, metric source, continuous-map row, dyadic mesh bound, cover-chain ledger, star-refinement transport, rational lower-bound fold, UniformModulus handoff, transport, replay, provenance, or local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverMeshChain}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCoverMeshChain}^{\uparrow}$ packet is generated from compact finite-cover data, metric source rows, continuous-map pointwise moduli, dyadic mesh bounds, finite cover-chain routing, star-refinement transport, rational lower-bound folding, UniformModulus handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteCoverMeshChain}\uparrow}$, seed)

Scope closed: Definition 7769.1, Definition 7769.2, and Definition 7769.3 seed the compact-uniform-continuity finite-cover mesh route as concrete NameCert data.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteCoverMeshChain}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host Lebesgue-number existence, semantic continuity, quotient metric equality, hidden centre selection, external modulus oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover exposure, dyadic mesh bounds, star-refinement transport, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverMeshChainUp.

Chapter 7770

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailDifference}^{\uparrow}$

Following Bishop and Bridges' constructive-analysis treatment of regular Cauchy names, the $\mathsf{RegularCauchyTailDifference}^{\uparrow}$ packet records the finite two-tail comparison surface visible before a $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the one-tail estimate route of chapter 1487, the dyadic tolerance surface of chapter 291, the finite stream window of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet compares two displayed regular Cauchy tails; it is not a quotient stream equality, a selected limit, a countable-choice schedule, host real equality, or an ambient completeness theorem.

Definition 7770.1 (Regular Cauchy tail difference carrier)**.** *A* $\mathsf{RegularCauchyTailDifference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Delta = (M_0, M_1, W_0, W_1, D, R_0, R_1, E, H, C, P, N)$$

with the following displayed rows:

1. M_0 *and* M_1 *are the two tail-threshold rows, one for each accepted regular Cauchy tail estimate.*
2. W_0 *and* W_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows inspected at and beyond the displayed thresholds.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row against which the two tail windows are compared.*
4. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows reached from* W_0, D *and* W_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary that may be consumed only after both threshold, window, tolerance, and readback routes have been replayed.*
6. H *records componentwise* hsame *transport for the two threshold rows, the two windows, the shared dyadic tolerance, the two readbacks, the seal boundary, continuation, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a two-tail comparison request through* $M_0, M_1, W_0, W_1, D, R_0, R_1$ *and then* E.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailDifference}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common finite two-tail surface. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient completeness.

Theorem 7770.2 (Regular Cauchy tail difference NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailDifference}^{\uparrow}$ *carrier* $\Delta = (M_0, M_1, W_0, W_1, D, R_0, R_1, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauch}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the two thresholds, two stream windows, shared dyadic tolerance, two regular readbacks, real seal boundary, transport, continuation, provenance, and local name;*
2. *classifier transport to a common finite surface over* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*;*
3. *stability of* $M_0, M_1, W_0, W_1, D, R_0, R_1, E$ *under the componentwise* hsame *rows recorded in* H*;*
4. *seal-route exactness requiring every Real-facing two-tail comparison to pass through both displayed threshold-window-readback routes before* E *is public;*
5. *refusal rows excluding quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, and ambient completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 7770.1. The projection exposes exactly the two threshold rows, two finite windows, shared dyadic tolerance, two regular readbacks, real seal boundary, transport, continuation, provenance, and local naming rows.

Use Definition 1487.9. The sibling theorem in the one-tail estimate chapter says that two accepted tail-estimate packets sharing transported threshold and dyadic rows compare through their displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows.

The structural rows do not create another comparison surface. The row H transports only the displayed coordinates, C replays the same two-tail route, and P, N record provenance and local naming for the finite packet.

Since no quotient stream, selected limit, host equality, countable-choice schedule, trichotomy, open-cover compactness, or ambient completeness coordinate occurs in the carrier, none can appear as a local obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailDifference}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailDifference}^{\uparrow}$ packet is generated from two regular Cauchy tail-threshold rows, two StreamName windows, one shared DyadicRatCore tolerance row, two RegSeqRat readbacks, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailDifference}^{\uparrow}}$, seed)

Scope closed: Definition 7770.1, Definition 7770.2, and Definition 1487.9 seed the finite two-tail comparison surface over regular Cauchy tail estimates.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailDifference}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, stability, seal-route exactness, and ledger non-escape for the same two-tail packet.

Chapter 7771

A Concrete Naming Certificate for $\mathsf{CaratheodoryExtension}^{\uparrow}$

Following Caratheodory and Halmos, the $\mathsf{CaratheodoryExtension}^{\uparrow}$ packet records the finite boundary by which a displayed premeasure read can be carried toward a measurable-hull value seal inside BEDC NameCert data. It sits beside the event-row discipline of chapter 92, the set-presentation boundary of chapter 3668, the distance-facing examples of chapter 57, and the terminal value seal of chapter 38. The packet names only the finite rows inspected by a downstream measure consumer; it does not assume the full sigma-algebra extension theorem as a primitive.

Definition 7771.1 (Caratheodory finite premeasure carrier)**.** *A* $\mathsf{CaratheodoryExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, \mu, U, V, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the consumed* $\mathsf{Measure}^{\uparrow}$ *source row exposing the event boundary, finite additivity rows, and downstream measure-facing ledger already admitted by the measure packet.*
2. *S is the* $\mathsf{SetLike}^{\uparrow}$ *source row naming the finite set presentation on which the inspected event rows are read.*
3. *μ is the premeasure row. It assigns the displayed source events to value rows before any hull or value-seal request is accepted.*
4. *U is the finite cover ledger. It lists only the finite covers and finite refinements inspected by the packet.*
5. *V is the measurable-hull row. It records the displayed hull candidate generated from the source event rows and finite cover ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *value seal for the inspected hull value, exposed only after M, S, μ, U, V have been displayed.*
7. *H records componentwise* hsame *transport for the measure source, setlike source, premeasure row, finite cover ledger, measurable-hull row, real value seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through finite event inclusion, finite cover comparison, premeasure value reads, hull formation, and terminal value sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{CaratheodoryExtension}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $M, S, \mu, U, V, R, H, C, P, N$. It has no coordinate for a completed sigma-algebra, countable choice over covers, ambient measure completion, host set equality, or quotient equality of event presentations.

Theorem 7771.2 (Caratheodory finite premeasure NameCert obligations)**.** *For every accepted* $\mathsf{CaratheodoryExtension}^{\uparrow}$ *carrier $E = (M, S, \mu, U, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryExtension}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{Measure}^{\uparrow}$ *source exposure,* $\mathsf{SetLike}^{\uparrow}$ *source exposure, premeasure-row admission, finite cover-ledger admission, measurable-hull row admission,* $\mathsf{Real}^{\uparrow}$ *value sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal of full sigma-algebra extension, countable choice over covers, ambient measure completion, and host set equality.*

Proof. Project the carrier of Definition 7771.1. Its public rows are exactly $M, S, \mu, U, V, R, H, C, P, N$, with the measure and setlike source rows read before any premeasure or hull value can be consumed.

The premeasure row μ and the cover ledger U are finite ledgers. They expose only displayed event-value assignments and displayed finite cover comparisons, so no countable cover selector or completed sigma-algebra can be recovered from them.

The measurable-hull row V and the value seal R are downstream rows over the same displayed data. The value seal is a $\mathsf{Real}^{\uparrow}$ row reached after the finite cover ledger has been inspected, not an ambient measure-completion theorem.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate exists in the carrier, the listed rows exhaust the NameCert surface and the stated refusals are forced by absence of fields. □

Falsifiable prediction: A $\mathsf{CaratheodoryExtension}^{\uparrow}$ read is refuted whenever a consumer reaches a measurable-hull RealUp value without first displaying the MeasureUp source, SetLikeUp source, premeasure row, finite cover ledger, measurable-hull row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CaratheodoryExtension}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CaratheodoryExtension}^{\uparrow}$ packet is generated from a Measure source row, a SetLike source row, a premeasure row, a finite cover ledger, a measurable-hull row, a RealUp value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CaratheodoryExtension}^{\uparrow}}$, seed)

Scope closed: Definition 7771.1 and Definition 7771.2 seed the finite premeasure carrier and its local NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CaratheodoryExtension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the full sigma-algebra extension theorem, countable choice over covers, ambient measure completion, host set equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Measure and SetLike source exposure, premeasure admission, finite cover admission, measurable-hull admission, Real value sealing, transport, replay, provenance, local naming, and refusal of full extension principles for BEDC.Derived.CaratheodoryExtensionUp.

Chapter 7772

A Concrete Naming Certificate for $\mathsf{StreamRegularCompleteness}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{StreamRegularCompleteness}^{\uparrow}$ packet records the finite ledger by which certified stream names are read as regular rational Cauchy data before a terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the finite stream-window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name carrier of chapter 38. The packet names a constructive completeness route for displayed regular data; it is not a quotient of Cauchy streams, a selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7772.1 (Stream regular completeness ledger)**.** *A* $\mathsf{StreamRegularCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, R, D, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite name windows and requested precisions inspected by the completion consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row translating those windows into a regular rational Cauchy readback.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording dyadic radii, refinement cells, and comparison budgets used by R.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after S, R, D have been displayed.*
5. *H records componentwise* hsame *transport for the stream, regular readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
6. *T records displayed* Cont *replay from a completion request through S, R, D, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{StreamRegularCompleteness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamRegularCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, R, D, E, H, T, P, N. It has no coordinate for quotient stream equality, a chosen Cauchy representative, a selected limit, countable choice, trichotomy, host real equality, or ambient completeness.

Theorem 7772.2 (Stream regular completeness NameCert obligations)**.** *For every accepted* $\mathsf{StreamRegularComplete}$ *carrier* $C = (S, R, D, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StreamRegularCompleteness}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *window coverage, requiring each completion request to expose the* $\mathsf{StreamName}^{\uparrow}$ *finite observations in* S *before any rational readback is consumed;*
3. *regularity coverage, requiring* R *to exhibit the regular rational Cauchy route over the same windows;*
4. *dyadic tolerance coverage, requiring* D *to account for the precision and refinement cells used by the regular readback;*
5. *Real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to read* E *only after* S, R, D *and the structural rows* H, T, P, N *have been displayed.*

No local obligation supplies quotient representatives, selected limits, countable-choice schedules, trichotomy, host equality, or ambient completeness.

Proof. Project the carrier of Definition 7772.1. The visible rows are exactly S, R, D, E, H, T, P, N, so each obligation must be read from one of those rows.

The first three substantive rows expose the constructive completion data. S supplies finite stream windows, R reads them as regular rational Cauchy data, and D records the dyadic budgets used to compare those readings.

The terminal row E is downstream of that route. Transport through H, replay through T, provenance through P, and naming through N preserve the same displayed packet and do not add a quotient, selected representative, choice schedule, trichotomy, host equality, or ambient completeness principle. □

Theorem 7772.3 (Stream regular completeness Real handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{StreamRegularCompleteness}^{\uparrow}$ *packet reaches terminal* $\mathsf{Real}^{\uparrow}$ *evidence only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, T, P, N.$$

Thus a certified stream name becomes a regular Cauchy real seal only after its finite windows, regular rational readback, and dyadic tolerance ledger are visible, and no consumer receives quotient-stream equality, a selected Cauchy representative, a selected limit, countable choice, trichotomy, host real equality, or ambient completeness from the packet.

Proof. Use the obligation surface of Definition 7772.2. Window coverage exposes S, regularity coverage exposes R, and dyadic tolerance coverage exposes D before the Real-seal nonescape row may be read.

The carrier definition then identifies the only terminal Real coordinate as E. Since E is downstream of S, R, D, every Real or Completion read factors through the displayed stream, regular-rational, and dyadic-tolerance route before the seal is available.

Finally inspect H, T, P, N. These rows transport, replay, source, and name the same finite packet. They do not contain quotient representatives, selected limits, choice schedules, trichotomy decisions, host equalities, or an ambient completeness theorem, so the handoff remains internal to the displayed NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{StreamRegularCompleteness}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches terminal $\mathsf{Real}^{\uparrow}$ evidence without first exposing StreamName finite windows, RegSeqRat regular readback, DyadicRatCore tolerance cells, componentwise transport, displayed replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{StreamRegularCompleteness}^{\uparrow}$ because the packet binds finite stream windows, regular rational Cauchy readback, dyadic tolerance accounting, terminal Real sealing, transport, replay, provenance, and local NameCert rows in one completeness ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamRegularCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamRegularCompleteness}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular rational readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamRegularCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7772.1, Definition 7772.2, and Definition 7772.3 seed the finite completeness route from certified stream names to regular Cauchy Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamRegularCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy streams, selected representatives, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, regularity coverage, dyadic tolerance coverage, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.StreamRegularCompletenessUp.

Chapter 7773

A Concrete Naming Certificate for $\mathsf{BishopLocatedDerivative}^{\uparrow}$

Following Bishop and Bridges on constructive differentiation, the $\mathsf{BishopLocatedDerivative}^{\uparrow}$ packet records a located finite derivative window before any host derivative or completed real-function equality is available. It sits over the finite $\mathsf{StreamName}^{\uparrow}$ surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name boundary of chapter 38, the finite-difference handoff of chapter 5135, and the derivative-facing carrier of chapter 122. The packet supplies the Real and Completion route with a finite located derivative window, not quotient real equality, selected limits, a host differentiability theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7773.1 (Bishop located derivative carrier)**.** *A* $\mathsf{BishopLocatedDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W, X, M, D, Q, R, F, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the located source-window row, listing the inspected* $\mathsf{StreamName}^{\uparrow}$ *input windows around the displayed base point.*
2. *X is the location row, naming the base point, neighbourhood tag, and finite radius requests used to keep the derivative window local.*
3. *M is the mesh row, recording the finite step choices, sidedness tags, endpoint reuse schedule, and exclusion of the zero step.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for step size, coefficient, denominator, and local-radius budgeting.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed difference quotients and their finite error ledger.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q has been read.*
7. *F is the* $\mathsf{FiniteDifference}^{\uparrow}$ *handoff row, exposing the finite quotient packet consumed before any derivative-facing read.*
8. *E is the* $\mathsf{Derivative}^{\uparrow}$ *boundary row, publishing only the located finite-window route.*

9. *H records componentwise* hsame *transport for source windows, location data, mesh choices, dyadic budgeting, regular readback, Real sealing, finite-difference handoff, derivative boundary, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay of the route*

$$W, X, M \longmapsto D \longmapsto Q \longmapsto R \longmapsto F \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedDerivative}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDerivative}^{\uparrow}}$ *row.*

The classifier has no coordinate for an ambient function space, quotient real equality, selected limiting derivative, host differentiability theorem, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7773.2 (BishopLocatedDerivativeUp NameCert obligations)**.** *For every accepted* BishopLocatedDerivativ *carrier*

$$B = (W, X, M, D, Q, R, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the located source windows, local base-point and radius data, finite mesh schedule, dyadic tolerance budgeting, regular rational readback, terminal Real seal, finite-difference handoff, derivative boundary exposure, transport, replay, provenance, and local naming. No obligation is stored in a host derivative, a quotient real equality proof, a selected limit, or an ambient completeness theorem.*

Proof. Project the carrier of Definition 7773.1. The rows W, X, M are the only coordinates containing source windows, location data, and finite mesh choices.

The numerical rows then force the Real-facing read through D, Q, R. The dyadic row D supplies the finite tolerance budget, the regular row Q reads the difference quotients, and the Real row R seals only that readback.

The consumer rows F, E expose the same finite route to chapter 5135 and chapter 122. The structural rows H, C, P, N transport, replay, source, and name the displayed packet without adding hidden data.

Since the carrier contains no ambient function-space, quotient-equality, selected-limit, host-differentiability, or completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 7773.3 (BishopLocatedDerivativeUp finite-difference handoff)**.** *Every* $\mathsf{Derivative}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopLocatedDerivative}^{\uparrow}$ *packet reads the finite-difference boundary F before it can inspect the derivative-facing row E. The handoff reaches F only through the source-window, location, mesh,* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows displayed in the carrier, so it does not export host differentiability, quotient real equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 7773.2. It places every accepted read on $W, X, M, D, Q, R, F, E, H, C, P, N$.

The finite-difference row F is downstream of the source-window and mesh rows and downstream of the regular-real chain D, Q, R. Thus it receives a finite displayed quotient packet rather than a host derivative.

The derivative boundary E is downstream of F. Componentwise hsame transport and displayed Cont replay preserve that order, while provenance and local naming identify the same finite packet.

The absent host differentiability, quotient equality, selected-limit, and completeness claims are not coordinates of the carrier, so no derivative consumer can bypass F. □

Theorem 7773.4 (BishopLocatedDerivativeUp Real seal nonescape)**.** *For every accepted* BishopLocatedDerivative$^{\uparrow}$ *packet, the Real-facing value exposed by the located derivative window is exactly the value sealed by the row R after* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerances, and* RegSeqRat$^{\uparrow}$ *readback have been displayed. The packet cannot re-expose that value as quotient real equality, a selected derivative limit, a host real function, or an ambient* Real$^{\uparrow}$ *completeness witness.*

Proof. Read R from the carrier in Definition 7773.1. By construction it is downstream of the finite source windows W, the local data X, M, the dyadic tolerance row D, and the regular readback row Q.

Apply Definition 7773.2. Its Real seal exposure clause identifies R as the only terminal Real-facing row in the local certificate.

The finite-difference and derivative rows F, E may consume that seal, but they do not replace it with a quotient equality or selected limiting object. The structural rows H, C, P, N only preserve and name the same displayed route.

Because no other carrier coordinate can hold a Real-facing witness, coordinate exhaustion prevents escape from the D, Q, R route. □

Closure status for NameCert$_{\text{BishopLocatedDerivative}^{\uparrow}}$**.**

Constructive story: A BishopLocatedDerivative$^{\uparrow}$ packet is generated from StreamName source windows, located base-point and mesh rows, DyadicRatCore tolerance budgeting, RegSeqRat readback, a RealUp seal, FiniteDifference and Derivative consumer boundaries, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopLocatedDerivative}^{\uparrow}}$, seed)

Scope closed: Definition 7773.1, Definition 7773.2, Definition 7773.3, and Definition 7773.4 seed the located finite derivative-window surface over StreamName, DyadicRatCore, RegSeqRat, Real, FiniteDifference, Derivative, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopLocatedDerivative}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host differentiability, quotient real equality, selected derivative limits, host real function equality, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked source-window admission, location admission, mesh admission, dyadic budgeting, regular readback, Real sealing, finite-difference handoff, derivative-boundary exposure, and transport/provenance/name exhaustion for BEDC.Derived.BishopLocatedDerivativeUp.

Chapter 7774

A Concrete Naming Certificate for $\mathsf{NormedGroup}^{\uparrow}$

Following Bourbaki, *General Topology*, the $\mathsf{NormedGroup}^{\uparrow}$ packet records the traditional normed-group interface as finite BEDC NameCert data. It consumes the algebraic operation rows of $\mathsf{Group}^{\uparrow}$ from chapter 41, the distance row of $\mathsf{Met}^{\uparrow}$ from chapter 57, and the value row of $\mathsf{Norm}^{\uparrow}$ from chapter 89. The carrier binds these three siblings as one displayed packet; it does not import quotient groups, host equality, selected representatives, or choice of completions.

Definition 7774.1 (Normed group carrier)**.** *A* $\mathsf{NormedGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, M, I, E, N, D, T, H, C, P, L)$$

with the following displayed rows:

1. *A is the consumed* $\mathsf{Group}^{\uparrow}$ *row, including carrier admission, multiplication, unit, inverse, and group-law evidence.*
2. *M is the consumed* $\mathsf{Met}^{\uparrow}$ *row on the same displayed carrier.*
3. *I is the displayed identity row inherited from the group packet.*
4. *E is the displayed inverse row inherited from the group packet.*
5. *N is the consumed* $\mathsf{Norm}^{\uparrow}$ *value row, read only on displayed group endpoints.*
6. *D is the norm-distance row recording $d(x, y) = N(x^{-1}y)$ inside the packet.*
7. *T is the translation-stability row recording that left and right multiplication preserve the displayed distance readback.*
8. *H is the componentwise* hsame *transport row for A, M, I, E, N, D, T.*
9. *C is the displayed* Cont *replay row for the same fields.*
10. *P is the* Pkg *provenance over* $\mathsf{NormedGroup}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, I, E, N, D, T, H, C, P, L$. It has no coordinate for host group equality, quotient completion, an ambient topological group, a chosen dense subgroup, or an external real-valued norm object outside the displayed $\mathsf{Norm}^{\uparrow}$ *row.*

Theorem 7774.2 (Normed group NameCert obligations)**.** *For every accepted* $\mathsf{NormedGroup}^{\uparrow}$ *carrier* $G = (A, M, I, E, N, D, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}$ *surface consists exactly of group-source admission, metric-source admission, norm-source admission, the identity and inverse readback rows, the norm-distance equation row* D*, translation stability* T*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, package provenance* P*, and local naming* L*.*

Proof. Project the carrier of Definition 7774.1. The algebraic source is the consumed $\mathsf{Group}^{\uparrow}$ row A, and the metric and norm sources are the consumed $\mathsf{Met}^{\uparrow}$ and $\mathsf{Norm}^{\uparrow}$ rows M and N.

The identity and inverse rows I and E are inherited from the same group packet. The distance row D is not an additional metric carrier; it is the displayed equation reading the metric value through the norm of $x^{-1}y$.

Translation stability is the row T. Transport, continuation, provenance, and local naming preserve exactly the displayed rows, so the obligation surface is exhausted by the listed finite packet. □

Theorem 7774.3 (Normed group metric translation stability)**.** *Every accepted* $\mathsf{NormedGroup}^{\uparrow}$ *consumer that reads a translated pair must expose the same* $\mathsf{Group}^{\uparrow}$ *multiplication row, inverse row,* $\mathsf{Met}^{\uparrow}$ *distance row, and* $\mathsf{Norm}^{\uparrow}$ *value row before reading the translated distance. Thus left and right translation are certificate-preserving routes through* T*, not host-level isometries imported from outside the packet.*

Proof. Read the translated pair through the group row A. Multiplication and inverse endpoints are visible only as rows of the consumed $\mathsf{Group}^{\uparrow}$ packet.

Next read M and N. The metric value and norm value are available only through those two source rows, and D ties the distance comparison to the norm of the displayed group difference.

The row T records the translation-stability replay. Since H, C, P, L preserve the same packet and no host isometry coordinate appears in the carrier, translated reads remain inside the accepted NameCert route. □

Theorem 7774.4 (Normed group norm-distance compatibility)**.** *For an accepted* $\mathsf{NormedGroup}^{\uparrow}$ *carrier, a metric read* $d(x, y)$ *and a norm read* $N(x^{-1}y)$ *are compatible only through the displayed row* D*. A consumer cannot replace* D *by host equality of group elements, quotient equality of representatives, or an unrecorded real-valued norm while retaining the local* $\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}$ *read.*

Proof. Project the distance and norm rows from Definition 7774.1. The metric row M supplies the displayed distance read, and the norm row N supplies the displayed value read.

The group row A supplies the inverse and multiplication endpoints used to form the displayed difference. The compatibility assertion is exactly row D, so the route has no independent equality or quotient coordinate.

Transport and continuation replay the same row data. Hence any replacement by host equality, quotient equality, or an unrecorded norm leaves the accepted packet. □

Theorem 7774.5 (Normed group ledger nonescape)**.** *Every downstream completion, topological-group, or Banach-facing consumer citing* $\mathsf{NormedGroup}^{\uparrow}$ *must first expose* $A, M, I, E, N, D, T, H, C, P, L$*. Dropping the group row, the metric row, the norm row, the norm-distance row, or the translation-stability row exits the accepted normed-group ledger and cannot be repaired by provenance or local naming alone.*

Proof. Apply the obligation surface of Definition 7774.2. The group, metric, and norm rows are independent consumed siblings, while D and T are the local rows that bind them into one normed-group packet.

Completion and topological consumers may inspect the metric-facing and translation-facing rows only after those local rows are displayed. Banach-facing consumers may inspect norm values only through N and the compatibility row D.

The support rows H, C, P, L preserve an admitted packet. They do not synthesize a missing algebraic source, metric source, norm source, compatibility row, or translation-stability row, so dropping any of those rows leaves the accepted ledger. □

Falsifiable prediction: Removing D permits a metric row and norm row that no longer determine the same displayed group-difference distance.

Independence witness: group, metric, norm: none is bijective to $\mathsf{NormedGroup}^{\uparrow}$. The group packet lacks distance and norm values; the metric packet lacks multiplication and inverse rows; the norm packet lacks the metric translation-stability ledger and cannot by itself expose the displayed group operation route.

Closure status for $\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}$.

Constructive story: A $\mathsf{NormedGroup}^{\uparrow}$ packet is generated from displayed GroupUp, MetricUp, NormUp, identity, inverse, norm-distance, translation-stability, transport, continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7774.1, Definition 7774.2, Definition 7774.3, Definition 7774.4, and Definition 7774.5 seed the finite normed-group route over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Norm}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormedGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient groups, group completions, Banach-space structure, topological-group bridge checking, host equality, selected representatives, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, metric translation stability, norm-distance compatibility, ledger nonescape, provenance, and local naming for BEDC.Derived.NormedGroupUp.

Chapter 7775

A Concrete Naming Certificate for $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$

Following Moore and Smith on nets, the $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ packet records the finite cofinal net window that an L10 completion route may inspect before it reads a completion-facing boundary. It sits over the directed-index carrier of chapter 1607, refines the net rows of chapter 3403, feeds the Cauchy-net completion corridor of chapter 3794, and presents the same finite readback style consumed by chapter 126, chapter 301, and chapter 38. The packet is a displayed finite window and handoff object; it is not a subnet selector, quotient net, cofinal-choice functional, selected limit, host equality assertion, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 7775.1 (Finite directed net window carrier)**.** *A* $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, M, \Omega, U, T, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index substrate, exposing the inspected addresses, preorder comparisons, finite request windows, and upper-window witnesses.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *source row whose tail ledger is being inspected.*
3. *Ω is the finite retained window of directed addresses used by the completion request.*
4. *U records, for every displayed finite subwindow of Ω, an upper witness still visible through D and the Moore–Smith tail ledger.*
5. *T is the cofinal touch row, stating that every completion-facing tail request named by the route has a displayed successor request meeting Ω through the same directed substrate.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row obtained by listing the retained finite window as an explicit schedule.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled window.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, M, Ω, U, T, S, R have been displayed.*
9. *H records componentwise* hsame *transport for the directed substrate, Moore–Smith source, retained window, upper witnesses, cofinal touch row, schedule, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a completion request through the finite route*

$$D, M, \Omega, U, T \longmapsto S \longmapsto R \longmapsto A, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDirectedNetWindow}^{\uparrow}}$ *row.*

The carrier exports only this finite completion-facing window. It has no coordinate for an infinite cofinal family, a selected subnet, quotient-net equality, a selected limit, countable choice, host real equality, or ambient completeness.

Theorem 7775.2 (Finite directed net window NameCert obligations)**.** *For an accepted* $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ *carrier $W = (D, M, \Omega, U, T, S, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteDirectedNetWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed substrate, Moore–Smith source, retained finite window, upper witnesses, cofinal touch row, StreamName schedule, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming row;*
2. *directed-window readability, requiring every retained address in Ω and every comparison used by U to be read through the displayed* $\mathsf{DirectedSet}^{\uparrow}$ *row D;*
3. *Moore–Smith tail compatibility, requiring the retained window and its upper witnesses to be visible in the* $\mathsf{MooreSmithNet}^{\uparrow}$ *tail ledger M;*
4. *finite cofinal touch, requiring each completion-facing tail request to pass through the displayed touch row T before any scheduled observation or readback is consumed;*
5. *completion readback order, requiring the route D, M, Ω, U, T to be displayed before the* $\mathsf{StreamName}^{\uparrow}$ *row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal A;*
6. *ledger refusal for infinite cofinal families, selected subnets, quotient-net equality, selected limits, countable choice, host real equality, and ambient completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7775.1. The displayed packet exposes $D, M, \Omega, U, T, S, R, A$ as analytic rows and H, C, P, N as structural rows.

Directed-window readability follows from the role of D. A retained address in Ω is usable only as an inspected directed address, and each upper witness in U is certified by the displayed comparison rows already present in the directed substrate.

Moore–Smith tail compatibility is the role of M. The retained finite window is not an external subnet; it is the finite part of the Moore–Smith tail ledger being handed to the completion route.

The cofinal touch row T is then read before any numerical schedule. It records the finite successor request that meets the retained window for the particular completion-facing demand under inspection, without selecting a global cofinal family.

Finally the scheduled exit is ordered through S, R, A. Transport and replay preserve the same rows through H and C, while P and N source and name the packet. Since none of the refused objects appears as a carrier coordinate, the listed rows exhaust the local obligation surface. □

Theorem 7775.3 (Finite directed net window completion handoff)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ \mathsf{MooreSmithNet}^{\uparrow},\ \Omega,\ U,\ T \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus an L10 completion route may inspect a finite cofinal net window as displayed BEDC data before the completion boundary is consumed, while quotient nets, selected subnets, selected limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ *completeness remain outside the packet.*

Proof. Apply the obligation surface of Definition 7775.2. It first forces the consumer to read the directed substrate D, Moore–Smith source M, retained window Ω, upper witnesses U, and cofinal touch row T.

The completion-facing route of chapter 3794 can then consume the same finite directed data as a boundary request. The packet supplies a displayed window and touch row, not a selected subnet or quotient of nets.

The numerical handoff is downstream of the directed window. The finite schedule S is read as a $\mathsf{StreamName}^{\uparrow}$ row, the regular rational readback R follows that schedule, and the terminal $\mathsf{Real}^{\uparrow}$ seal A is inspected only afterward.

Componentwise transport, continuation replay, provenance, and local naming remain H, C, P, N. Since the carrier has no coordinate for choice, selected limits, quotient equality, host equality, or ambient completeness, no such object can bypass the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ read exposes one DirectedSet row, one MooreSmithNet source row, one retained finite window, one upper-witness ledger, one cofinal touch row, one StreamName schedule, one RegSeqRat readback, one RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and a local NameCert row before a CauchyNetCompletion consumer receives the packet.

Independence witness: directedset, moore-smith-net, cauchynetcompletion, streamname, regseqrat, real: $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds retained directed windows, upper witnesses, a finite cofinal touch row, scheduled observation, regular readback, terminal Real sealing, transport, replay, provenance, and local naming as one L10 completion handoff packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDirectedNetWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDirectedNetWindow}^{\uparrow}$ packet is generated from DirectedSet rows, MooreSmithNet source rows, retained finite directed windows, upper-witness ledgers, cofinal touch rows, StreamName schedules, RegSeqRat readback, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDirectedNetWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7775.1, Definition 7775.2, and Definition 7775.3 seed the finite cofinal net-window handoff used by L10 completion routes over DirectedSetUp, MooreSmithNetUp, CauchyNetCompletionUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDirectedNetWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subnets, quotient nets, infinite cofinal families, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window readability, Moore–Smith tail compatibility, finite cofinal touch, completion readback order, transport preservation, and ledger refusal for the same finite packet.

Chapter 7776

A Concrete Naming Certificate for $\mathsf{UlmInvariant}^{\uparrow}$

Following Kaplansky and Fuchs, the $\mathsf{UlmInvariant}^{\uparrow}$ packet records the classical height-filtration invariant for countable reduced p-primary abelian groups as finite BEDC NameCert data. It sits over the commutative operation surface of chapter 42, the filtered scalar bookkeeping of chapter 46, and the torsion-window route of chapter 6880. The packet names finite height slices, quotient multiplicity rows, transport, replay, provenance, and local naming evidence; it does not assert the full Ulm classification theorem, choose representatives of quotient slices, construct an ambient Prüfer envelope, or compare external groups by an isomorphism selector.

Definition 7776.1 (Ulm invariant carrier)**.** *An* $\mathsf{UlmInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (p, A, H, F, Q, M, T, C, P, N)$$

with the following displayed rows:

1. *p is the prime row naming the fixed p-primary torsion coordinate and its visible arithmetic ledger.*
2. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *source row, exposing the abelian operation, unit, inverse, commutativity, and classifier rows read by the invariant.*
3. *H is the height-filtration row, listing the displayed finite height stages and the rowwise membership tests used at each stage.*
4. *F is the successor-filtration row, recording the displayed passage from a height stage to its next p-multiple boundary.*
5. *Q is the quotient-slice row, naming the finite comparison surface for height-stage elements modulo the next displayed boundary.*
6. *M is the multiplicity row, recording the finite cardinal ledger of independent quotient-slice generators inspected by the packet.*
7. *T records componentwise* hsame *transport for the prime, abelian source, height filtration, successor filtration, quotient slices, multiplicities, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a torsion-height query through $p, A, H, F, Q, M, T, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UlmInvariant}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{PrueferGroup}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{UlmInvariant}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a full external abelian group classification, a selected basis of quotient slices, a hidden completion of the height filtration, or an ambient Prüfer-group embedding.

Theorem 7776.2 (Ulm invariant height-filtration obligation)**.** *For every accepted* $\mathsf{UlmInvariant}^{\uparrow}$ *carrier*

$$U = (p, A, H, F, Q, M, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UlmInvariant}^{\uparrow}}$ *surface first reads the* $\mathsf{AbGroup}^{\uparrow}$ *source row A, then the displayed height-filtration row H, then the successor-filtration row F, and only then the quotient-slice and multiplicity rows Q, M. Thus every invariant-facing consumer receives height data as a finite* BHist *filtration route rather than as an ambient ordinal-indexed subgroup lattice or a choice-selected basis.*

Proof. Project the carrier of Definition 7776.1. The only algebraic source row is A, and the only height-facing rows are H, F, Q, M.

The abelian source is read first. It supplies the operation, inverse, unit, commutativity, and classifier surface needed to speak about p-multiples and torsion-height comparisons.

The height filtration is then exposed through H and F. Row H lists the displayed stages, while F records the successor boundary used to compare a height stage with its next p-multiple layer.

Finally Q and M read quotient slices and multiplicities only after those boundaries are visible. The transport, replay, provenance, and local naming rows T, C, P, N preserve this order, so no hidden subgroup lattice or selected quotient basis enters the NameCert surface. □

Theorem 7776.3 (Ulm invariant Pruefer dependency route)**.** *Every Prüfer-facing or classification-facing consumer of an accepted* $\mathsf{UlmInvariant}^{\uparrow}$ *packet must retain the displayed route through chapter 42, chapter 46, and chapter 6880. The* $\mathsf{PrueferGroup}^{\uparrow}$ *dependency is read only as a torsion-window comparison surface for p-primary height behavior; it is not a construction of an ambient divisible hull, a selected compatible root chain, or the full Ulm classification isomorphism.*

Proof. Start with the provenance row P of Definition 7776.1. It lists the abelian-group, module-style filtration, and Prüfer torsion-window surfaces as named dependencies of the same packet.

The abelian-group row supplies the operation and classifier. The module-style row supplies filtered scalar bookkeeping for the p-multiple boundary, and the Prüfer-group row supplies the finite torsion-window comparison surface used to keep height behavior visible.

The quotient-slice and multiplicity rows remain downstream of those dependencies. Because T, C, P, N only transport, replay, source, and name the same finite route, no consumer can replace it by a divisible hull, a compatible root selector, or an external classification isomorphism. □

Closure status for $\mathsf{NameCert}_{\mathsf{UlmInvariant}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UlmInvariant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7776.1, Definition 7776.2, and Definition 7776.3 seed the finite prime, abelian-source, height-filtration, successor-filtration, quotient-slice, multiplicity, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UlmInvariant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full Ulm classification theorem, selected quotient representatives, ambient Prüfer envelopes, external group isomorphism, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, height-filtration stability, quotient-slice multiplicity, Prüfer torsion-window dependency, transport, replay, provenance, and local naming for BEDC.Derived.UlmInvariantUp.

Constructive story: A $\mathsf{UlmInvariant}^{\uparrow}$ packet is generated from a fixed prime row, AbGroup source row, height-filtration row, successor-filtration boundary, quotient-slice row, multiplicity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7777

A Concrete Naming Certificate for $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$

Following Cauchy and Hardy, the $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ packet records the finite estimate surface needed before a product of regular Cauchy names can expose a Real-facing tail bound. It refines the product-tail budget route of chapter 6606, consumes the regular rational readback of chapter 301, and remains attached to the dyadic tolerance core of chapter 291, the product convergence and metric routes of chapter 3608 and chapter 3551, and the terminal real-name seal of chapter 38. The packet is a finite NameCert surface for product-tail estimates; it does not assert a selected infinite product, quotient equality of series, choice, trichotomy, host completed multiplication, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7777.1 (Regular Cauchy product-tail estimate carrier)**.** *A* $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, B, J, L, R, M, D, Q, Z, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose finite rational windows are inspected.*
2. *J is the* $\mathsf{CauchyProductTailBudget}^{\uparrow}$ *schedule naming the left tail, right tail, and mixed finite convolution windows allowed for the requested precision.*
3. *L and R are the left and right regular-tail ledgers read from A and B under the schedule J.*
4. *M is the mixed-window convolution ledger, containing only finitely many rational products and finite partial sums.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger assigning dyadic radii to L, R, and M.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed tail estimate.*
7. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after A, B, J, L, R, M, D, Q are visible.*
8. *H records componentwise* hsame *transport for the source rows, schedule, tail ledgers, mixed window, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite product-tail estimate route.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$, $\mathsf{CauchyProductTailBudget}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}}$ *row.*
The classifier reads only these rows. It has no coordinate for a chosen infinite product, quotient equality of regular names, countable choice, trichotomy, host multiplication on completed reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7777.2 (Regular Cauchy product-tail estimate NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ *carrier* $E = (A, B, J, L, R, M, D, Q, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission, two* $\mathsf{RegSeqRat}^{\uparrow}$ *source exposures, product-tail budget scheduling, left-tail and right-tail ledger exposure, mixed finite-window convolution,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7777.1. The two source rows A, B, the budget row J, the tail ledgers L, R, the mixed window M, and the dyadic ledger D are the only numerical rows available before readback.

The readback row Q is downstream of those finite rows. It repacks the displayed source windows and product-tail estimate as regular rational data rather than as quotient equality of regular names or a chosen infinite product.

Finally inspect Z, H, C, P, N. The Real seal Z is reached only after the source, schedule, tail-ledger, mixed-window, dyadic, and readback rows have been displayed, while the structural rows preserve the same packet. Hence the listed rows exhaust the local NameCert obligation surface. □

Theorem 7777.3 (Regular Cauchy product-tail estimate Real-seal route)**.** *Every Real-facing read of an accepted* $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyProductTailBudget}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the displayed left-tail ledger, right-tail ledger, mixed finite convolution window, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows still attached. The route exports no selected infinite product, quotient equality of regular names, countable choice, trichotomy row, host completed multiplication, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with Definition 7777.2. It makes the two regular Cauchy sources, product-tail schedule, left and right tail ledgers, mixed finite convolution row, dyadic tolerance ledger, and regular readback mandatory before the Real seal can be consumed.

The finite route through J, L, R, M, D, Q controls the estimate request. A consumer may inspect only the displayed source windows, tail rows, mixed products, and dyadic tolerances assigned to the requested precision; no coordinate supplies an external selector or host infinite product.

The structural rows H, C, P, N preserve this same route under transport, replay, provenance, and naming. Therefore any Real-facing read reaches Z only through the displayed BEDC rows and cannot escape to quotient equality, choice, trichotomy, host multiplication, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ endpoint read exposes two RegSeqRat source rows, a CauchyProductTailBudget schedule, left and right finite tail ledgers, a mixed finite convolution window, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any product-tail estimate is public.

Independence witness: cauchy-product-tail-budget, regseqrat, dyadicratcore, real: RegularCauchyProductTailEst is not bijective to any listed sibling because it jointly carries two regular Cauchy sources, a product-tail schedule, two tail ledgers, a mixed finite convolution row, dyadic tolerance accounting, regular readback, terminal Real sealing, transport, replay, provenance, and local naming in one estimate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a CauchyProductTailBudget schedule, finite tail ledgers, a mixed convolution window, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7777.1, Definition 7777.2, and Definition 7777.3 seed the finite Cauchy-product tail-estimate route over displayed regular sources, product-tail budgets, dyadic tolerance ledgers, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductTailEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected infinite products, quotient equality of regular names, countable choice, trichotomy, host completed multiplication, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, product-tail scheduling, left and right tail-ledger exposure, mixed finite-window convolution, dyadic tolerance accounting, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyProductTailEstimateUp.

Chapter 7778

A Concrete Naming Certificate for $\mathsf{SequentialCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{SequentialCompletionCriterion}^{\uparrow}$ packet records the finite sequential criterion that an L10 completion route may read before treating a regular Cauchy stream as completion-facing data. It sits after the metric substrate of chapter 57, the stream windows of chapter 126, the regular rational readback of chapter 301, the Real seal of chapter 38, the complete-metric row of chapter 128, and the completion-existence route of chapter 5721. The packet is a displayed criterion and handoff object; it is not a quotient completion, selected representative functional, countable-choice principle, trichotomy oracle, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7778.1 (Sequential completion criterion carrier)**.** *A* $\mathsf{SequentialCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, E, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the named distance comparisons and the separated metric source inspected by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite stream observations that the completion route is allowed to read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row attached to the displayed stream windows.*
4. *D is the dyadic tolerance ledger, recording the finite tolerance requests and regularity bounds used by the readback.*
5. *E is the completion-seal row, stating that the displayed source, stream windows, readback, and tolerance ledger are accepted as completion-facing data for the named route.*
6. *U is the separated uniqueness row, comparing two displayed readbacks only through metric separation, regular tolerances, and the same completion seal.*
7. *H records componentwise* hsame *transport for the metric source, stream windows, readback, tolerance ledger, completion seal, separated uniqueness row, replay, provenance, and local naming row.*

8. *C records displayed* Cont *replay from a sequential completion request through*

$$M, S, R, D \longmapsto E, U, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{SequentialCompletionCriterion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletionExistence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletionCriterion}^{\uparrow}}$ *row.*

The carrier exports only this finite sequential criterion. It has no coordinate for a quotient completion, selected representative, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or unlogged equality of limits.*

Theorem 7778.2 (Sequential completion criterion NameCert obligations)**.** *For an accepted* SequentialCompletionC *carrier $Q = (M, S, R, D, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, stream-name windows, regular Cauchy readback, dyadic tolerance ledger, completion-seal row, separated uniqueness row, transport, replay, provenance, and local naming row;*

2. *source readability, requiring every distance comparison and separatedness use to pass through the displayed* $\mathsf{Met}^{\uparrow}$ *row M;*

3. *sequential observation, requiring the completion-facing stream requests to be named by the finite* $\mathsf{StreamName}^{\uparrow}$ *window row S before any rational readback is consumed;*

4. *regular readback, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and dyadic tolerance ledger D to witness the finite Cauchy bounds used by the criterion;*

5. *completion sealing and uniqueness, requiring E and U to bind the displayed readback to the completion route and to compare candidate readings only through separated metric data;*

6. *ledger refusal for quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and any equality assertion not displayed in $M, S, R, D, E, U, H, C, P, N$.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 7778.1. The rows M, S, R, D, E, U are the analytic part of the criterion, while H, C, P, N transport, replay, source, and name the same finite packet.

Source readability is forced by M. Any metric comparison used by the completion route is read as displayed metric data, so the criterion cannot appeal to a hidden equality of points or an ambient complete space.

The sequential part is then ordered through S, R, D. The stream windows name the observations, the regular rational readback supplies the Cauchy data attached to those observations, and the dyadic ledger records the tolerance requests that are actually inspected.

The completion seal E and separated uniqueness row U close the finite criterion. They certify that the displayed readback is completion-facing and that two displayed readings are compared only by the separated metric rows and the same tolerance ledger.

Finally H and C replay these rows componentwise, while P and N record provenance and local naming. Since none of the refused objects appears as a carrier coordinate, quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{SequentialCompletionCriterion}^{\uparrow}$ read exposes one Metric row, one StreamName window row, one RegSeqRat readback row, one dyadic tolerance ledger, one completion-seal row, one separated uniqueness row, componentwise hsame, Cont replay, Pkg provenance, and a local NameCert row before an L10 completion consumer receives the criterion.

Independence witness: metric, streamname, regseqrat, real, completemetric, metric-completion-existence: $\mathsf{SequentialCompletionCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds metric source readability, finite stream windows, regular Cauchy readback, dyadic tolerance inspection, completion sealing, separated uniqueness, transport, replay, provenance, and local naming as one L10 completion criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletionCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompletionCriterion}^{\uparrow}$ packet is generated from Metric rows, StreamName windows, RegSeqRat regular Cauchy readback, dyadic tolerance ledgers, completion-seal rows, separated uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompletionCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7778.1 and Definition 7778.2 seed the finite sequential completion criterion route over MetricUp, StreamNameUp, RegSeqRatUp, RealUp, CompleteMetricUp, MetricCompletionExistenceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompletionCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric source readability, sequential observation, regular readback, dyadic tolerance inspection, completion sealing, separated uniqueness, transport preservation, and ledger refusal for the same finite packet.

Chapter 7779

A Concrete Naming Certificate for $\mathsf{LocatedCompactnessModulus}^{\uparrow}$

Following Bishop–Bridges constructive compactness and Kelley's finite-cover viewpoint, the LocatedCompactnessMo packet records the finite-net route by which a compact metric source exports a usable lower-bound modulus. It sits after the compact metric surface of chapter 129, the finite Lebesgue-number route of chapter 452, and the regular rational readback route of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names finite located-net evidence; it does not claim open-cover compactness, choose an external subcover, complete $\mathsf{Real}^{\uparrow}$, or identify compactness with a host-space theorem.

Definition 7779.1 (Located compactness modulus carrier)**.** *A* $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, Q, B, S, G, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the accepted compact metric carrier and the finite metric windows read by the packet.*
2. *F is the located finite-net cover row, listing the displayed centers, membership tests, and cover evidence used for the modulus.*
3. *Q is the positive rational radius ledger, recording the finite radius candidates and their comparison data.*
4. *B is the finite lower-bound fold, producing the displayed positive rational lower bound from the nonempty radius ledger.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the stream-facing name of the selected finite radius data.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, presenting the regular rational sequence visible to downstream modulus consumers.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, packaging the regular rational readback as the real-valued modulus output without using a completeness theorem.*
8. *H records componentwise* hsame *transport for the compact metric source, located finite-net cover, rational-radius ledger, lower-bound fold, stream name, regular rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-source tolerance query through* $K, F, Q, B, S, G, R, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactnessModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, choice-selected subcovers, unbounded radius searches, hidden Cauchy completion, or external compactness principles.

Theorem 7779.2 (Located compactness modulus NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ *carrier*

$$L = (K, F, Q, B, S, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactnessModulus}^{\uparrow}}$ *surface consists exactly of compact metric source admission, located finite-net cover exposure, positive rational radius ledgering, finite lower-bound folding,* $\mathsf{StreamName}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7779.1. Row K is the only compact source row, rows F, Q, B are the finite-net and lower-bound rows, and rows S, G, R are the readback and real-seal rows.

The compact source is read first. It supplies the metric windows and compact carrier evidence, but it does not export an arbitrary open-cover compactness theorem.

The located finite-net cover row then exposes exactly the displayed centers, membership tests, and cover certificates used by the modulus. The rational ledger Q lists the candidate radii, and the finite fold B returns the positive lower bound by inspecting that finite list.

Finally S, G, R present the bound to stream, regular-rational, and real-valued consumers. Rows H, C, P, N transport, replay, source, and name the same packet, so the listed rows exhaust the NameCert obligation surface. □

Theorem 7779.3 (Located compactness modulus finite-net extraction)**.** *Every accepted* LocatedCompactnessModu *packet extracts its modulus through a displayed finite net*

$$\mathsf{CompactMetric}^{\uparrow} \longrightarrow F \longrightarrow Q \longrightarrow B$$

before any readback row is exported. Thus a compact-uniform-continuity consumer may read a positive rational lower bound only after the located cover evidence and finite fold have appeared in the packet.

Proof. Start from Definition 7779.2. The rows are ordered so that K precedes the finite-net row F, the finite-net row precedes the rational ledger Q, and the ledger precedes the fold B.

The extraction is therefore finite at the point where the lower bound is formed. The fold B has access only to the displayed candidates in Q and the located cover evidence in F; it has no access to an external cover selector or an infinite search.

Since S, G, R, H, C, P, N occur downstream of B, every exported modulus readback factors through the same finite-net extraction route. □

Theorem 7779.4 (Located compactness modulus real handoff)**.** *The* $\mathsf{Real}^{\uparrow}$ *output of an accepted* $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ *carrier is a seal on the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows. It is not a proof of* $\mathsf{Real}^{\uparrow}$ *completeness, not an external located-real construction, and not a replacement for the finite lower-bound fold.*

Proof. Inspect the readback rows in Definition 7779.1. The stream row S names the selected finite radius data, the regular rational row G presents it as a visible regular sequence, and the real row R seals that presentation for consumers expecting a real-valued modulus.

This handoff is downstream of K, F, Q, B. The lower bound has already been produced by the finite fold before the stream and regular-rational rows are read.

Thus R carries only the terminal seal for the displayed readback route. It does not add a completeness proof, an external located-real selector, or a new source for the modulus. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ packet exposes compact metric source, located finite-net cover, positive rational radius ledger, finite lower-bound fold, StreamName readback, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows before a modulus is exported.

Independence witness: Nearest siblings: CompactMetric, FiniteLebesgueNumber, StreamName, RegSeqRat, DyadicRatCore, and Real. None is bijective to $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ because this packet jointly carries the finite located-net cover, rational-radius ledger, lower-bound fold, readback, sealing, transport, replay, provenance, and local naming route for compact-uniform modulus consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactnessModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactnessModulus}^{\uparrow}$ packet is generated from a CompactMetric source row, located finite-net cover row, positive rational radius ledger, finite lower-bound fold, StreamName readback, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactnessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7779.1, Definition 7779.2, Definition 7779.3, and Definition 7779.4 seed the finite compact-source, located-net, rational-radius, lower-bound, readback, Real-seal, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactnessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected subcovers, Real completeness, external located-real construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located finite-net extraction, rational-radius ledgering, finite lower-bound folding, StreamName and RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactnessModulusUp.

Chapter 7780

A Concrete Naming Certificate for $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ packet records the constructive uniform Cauchy criterion for a displayed sequence of functions before a ContinuousMap or completion consumer may read a terminal Real seal. It sits after the stream-name schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name carrier of chapter 38, the continuous-map route of chapter 130, and the metric-space carrier discipline of chapter 57. The packet supplies a finite-window criterion for function sequences; it is not pointwise convergence by host functions, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness.

Definition 7780.1 (Uniform Cauchy criterion function carrier)**.** *A* $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, X, F, W, T, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for the function sequence and the finite family of inspected arguments.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distance stream produced by the inspected function values.*
3. *X is the displayed domain-window row whose metric comparisons are read through* $\mathsf{Met}^{\uparrow}$*.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row certifying that each inspected function value is read through the same displayed continuity route.*
5. *W is the finite uniform tail window, listing the indices and argument windows needed for one requested tolerance.*
6. *T is the dyadic tolerance ledger comparing the requested uniform bound with the distances generated by S, R, X, F, W.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed uniform Cauchy bound.*
8. *M is the target* $\mathsf{Met}^{\uparrow}$ *distance row, consumed only after S, R, X, F, W, T, E are visible.*

9. *H records componentwise* hsame *transport for the stream schedule, regular readback, domain window, continuous-map row, uniform tail window, tolerance ledger, Real seal, metric row, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a uniform tolerance request through $S, R, X, F, W, T, E, M, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterionFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, pointwise-only convergence, quotient stream equality, selected representatives, countable-choice schedules, trichotomy, or ambient completeness.

Theorem 7780.2 (Uniform Cauchy criterion function obligations)**.** *For every accepted* UniformCauchyCriterionFunc *carrier $U = (S, R, X, F, W, T, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterionFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed stream schedule, regular readback, domain metric window, continuous-map row, finite uniform tail window, dyadic tolerance ledger, Real seal, metric row, componentwise transport, continuation replay, provenance, and local naming; uniform-tail exposure through W; Real-seal handoff through T, E; and ledger refusal for pointwise-only convergence, host function equality, quotient streams, selected representatives, countable choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 7780.1. The rows S, R, X, F, W, T, E, M are the only analytic rows available to a uniform Cauchy read, and H, C, P, N are the only structural rows that may transport, replay, source, and name the packet.

Uniform-tail exposure is the role of W. It records one finite window for the requested tolerance, with the function-sequence schedule in S, the regular readback in R, the domain metric window in X, and the continuous-map row F already displayed.

The tolerance ledger T compares the requested bound with the distances generated by that finite window. The terminal Real row E and metric row M are therefore downstream of the finite uniform route, not detached pointwise evidence.

Since no carrier coordinate stores host function equality, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7780.3 (Uniform Cauchy criterion function finite-window induction)**.** *Let $U = (S, R, X, F, W, T, E, M, H$ be an accepted* $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ *carrier. Induction over the displayed finite uniform tail window preserves the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Strengthening or extending a finite prefix keeps the uniform Cauchy read inside $S, R, X, F, W, T, E, M, H, C, P, N$ and cannot introduce pointwise-only convergence, host function equality, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness.

Proof. The base window is the carrier projection of Definition 7780.1. The displayed rows already expose the function-sequence schedule, regular rational readback, domain metric window, continuous-map row, finite tail window, dyadic tolerance ledger, terminal Real seal, and metric row.

For the induction step, extend the finite prefix by one accepted argument window or one accepted tail-index comparison. The obligation surface of Definition 7780.2 forces the extension to pass through S, R, X, F, W before the tolerance ledger T and Real seal E are consumed.

If the prefix is strengthened without adding a new argument window, the same obligation surface keeps the readback and continuous-map rows fixed while only the displayed finite tolerance ledger is refined. The route therefore remains a uniform window route rather than a host-level pointwise convergence claim.

Finally inspect H, C, P, N. These structural rows transport, replay, source, and name only the same displayed packet, so the induction cannot acquire quotient stream equality, selected representatives, countable choice, trichotomy, host equality, or ambient completeness from outside the carrier. □

Falsifiable prediction: An accepted $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ read is refuted if a ContinuousMap or completion consumer obtains a uniform Cauchy bound without displaying the StreamName schedule, RegSeqRat readback, ContinuousMap row, MetricSpace distance row, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, real, continuous-map, metric-space: none alone is bijective with $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ because the packet binds a function-sequence schedule, regular readback, domain windows, continuous-map rows, finite uniform tail windows, dyadic tolerance accounting, terminal Real sealing, metric rows, transport, replay, provenance, and local NameCert rows in one uniform criterion surface.

Lean4 (statement) `BEDC.Derived.UniformCauchyCriterionFunctionUp`

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterionFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCriterionFunction}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readbacks, displayed metric windows, ContinuousMap rows, finite uniform tail windows, dyadic tolerance ledgers, RealUp seals, MetricSpace rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCriterionFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7780.1, Definition 7780.2, and Definition 7780.3 seed the finite-window uniform Cauchy criterion for displayed function sequences over StreamNameUp, RegSeqRatUp, RealUp, ContinuousMapUp, MetricSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCriterionFunction}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close pointwise-only convergence, host function equality, quotient stream equality, selected representatives, countable choice, trichotomy, ambient completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-tail exposure, Real-seal handoff, finite-window induction, transport, replay, provenance, and ledger refusal for BEDC.Derived.UniformCauchyCriterionFunctionUp.

Chapter 7781

A Concrete Naming Certificate for $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ packet records the finite compact-net diagonal data needed to read compact equicontinuous families without importing a choice-selected subsequence or an ambient completeness theorem. It sits between the compact metric source of chapter 129, the family-radius ledger of chapter 413, the bounded-family rows of chapter 3522, the Arzela–Ascoli handoff of chapter 412, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed diagonal schedule over finite windows only; it does not assert countable choice, selected subsequences outside the displayed schedule, quotient function equality, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7781.1 (Bishop equicontinuity diagonal carrier)**.** *A* $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, \Delta, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *packet and its displayed finite probes.*
2. *F is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *source row recording the family-radius ledger visible at the inspected compact-net scales.*
3. *B is the* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *ledger that bounds the same displayed family windows before any diagonal readback is accepted.*
4. *Δ is the finite diagonal schedule: it lists the inspected net scale, family index window, source window, and target tolerance window in the order in which the diagonal reader may consume them.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the scheduled family values and candidate limit reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after K, F, B, Δ, W, R have been displayed.*

8. *H records componentwise* hsame *transport for the compact-net, equicontinuity, bounded-family, diagonal-schedule, stream-window, rational-readback, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from compact finite-net exposure through equicontinuity-family rows, bounded-family ledgers, the diagonal schedule, finite stream windows, regular rational readback, and terminal Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{BoundedFunctionFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}}$ *row.*

Accepted packets are compared only through the displayed finite compact-net, equicontinuity, boundedness, diagonal schedule, finite-window, readback, seal, transport, replay, provenance, and naming rows.

Theorem 7781.2 (Bishop equicontinuity diagonal NameCert obligations)**.** *For every accepted* $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ *carrier* $E = (K, F, B, \Delta, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityDiago}}$ *surface consists exactly of compact finite-net source exposure, equicontinuity-family source exposure, bounded-family ledger exposure, displayed diagonal scheduling, StreamName finite windows, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7781.1. The compact finite-net source is K, the family-radius source is F, and the bounded-family ledger is B; these are precisely the rows whose sibling chapters are chapter 129, chapter 413, and chapter 3522.

The diagonal information is finite. The schedule Δ displays the net scale, family-index window, source window, and target tolerance window before any readback is public. Thus a consumer may not replace the schedule by countable choice or by a selected subsequence outside the packet.

The readback route remains visible. Rows W, R, S expose the StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the Real seal of chapter 38; the seal is reached only after the compact, family, boundedness, and schedule rows are displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores quotient function equality, host compactness, a choice-selected subsequence, countable choice, or ambient Real completeness, the local NameCert surface cannot export those objects. □

Theorem 7781.3 (Bishop equicontinuity diagonal finite-net route)**.** *Every* $\mathsf{ArzelaAscoli}^{\uparrow}$ *or compact-uniform-continuity consumer of an accepted* $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ *packet must read the compact finite-net row, the equicontinuity-family source row, the bounded-family ledger, the displayed diagonal schedule, StreamName finite windows, RegSeqRat readback, and the terminal Real seal before it consumes any diagonal handoff. The handoff is a finite schedule route, not countable choice, a subsequence selector outside Δ, quotient equality of functions, host compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $E = (K, F, B, \Delta, W, R, S, H, C, P, N)$. The only compact source visible to a consumer is K, and the only family-radius and boundedness data visible before the diagonal read are F and B.

Next read the schedule Δ. It lists exactly the finite net scales, family-index windows, source windows, and target tolerance windows consumed by the diagonal route, so no unstated subsequence or external selector is available.

The readback rows W, R, S then force the same route through chapter 126, chapter 301, and chapter 38. The $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer of chapter 412 therefore receives a displayed finite diagonal packet, not a quotient function equality or a host compactness theorem.

The structural rows H, C, P, N preserve this route componentwise. They add provenance and local naming but no countable-choice coordinate, no subsequence outside Δ, and no ambient completeness principle. □

Theorem 7781.4 (Bishop equicontinuity diagonal Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ *packet factors through*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{EquicontinuityFamily}^{\uparrow},\quad \mathsf{BoundedFunctionFamily}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed diagonal schedule, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The seal is not an ambient completeness theorem and cannot be read before the displayed schedule and finite windows have been exposed.*

Proof. Project the Real-facing row S. By Definition 7781.2, S is exposed only after K, F, B, Δ, W, R have already been displayed.

The finite-net route theorem Definition 7781.3 identifies this same order as the only diagonal handoff available to Arzela–Ascoli and compact-uniform-continuity consumers. Hence a Real-facing read receives the scheduled finite windows and regular rational readbacks before the seal.

The rows H, C, P, N then preserve the displayed route under classifier transport and replay. None of them contains countable choice, a subsequence outside the displayed schedule, quotient function equality, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the Real seal remains local to the packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ reader that claims a diagonal Real-facing handoff while omitting compact finite-net exposure, EquicontinuityFamily source rows, BoundedFunctionFamily ledgers, the displayed diagonal schedule, StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: compactmetric, equicontinuityfamily, bounded-function-family, arzelaascoli, streamname, regseqrat, real: $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ is not bijective with any listed sibling because it joins compact finite-net source data, family equicontinuity rows, bounded-family ledgers, a displayed diagonal schedule, finite stream windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}$ packet is generated from compact finite-net source rows, EquicontinuityFamilyUp source rows, BoundedFunctionFamilyUp ledgers, a displayed diagonal schedule, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7781.1, Definition 7781.2, Definition 7781.3, and Definition 7781.4 seed the finite compact-net diagonal route for compact equicontinuous families.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected subsequences outside the displayed schedule, quotient function equality, host compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, equicontinuity-family exposure, bounded-family ledger exposure, diagonal scheduling, finite-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopEquicontinuityDiagonalUp.

Chapter 7782

A Concrete Naming Certificate for $\mathsf{CalibratedMetricFamily}^{\uparrow}$

Following Dudley and van der Vaart–Wellner, the $\mathsf{CalibratedMetricFamily}^{\uparrow}$ packet records a finite metric-family calibration surface for bracketing, covering, packing, entropy, and uniform-modulus handoff data. It sits beside the metric source of chapter 57, the compact-metric source of chapter 129, the finite entropy ledger of chapter 5209, the covering row of chapter 2187, the packing row of chapter 3761, and the modulus consumer of chapter 369. The packet names only displayed finite calibration rows and does not import host function-space compactness, measurable envelopes, selected brackets, or choice.

Definition 7782.1 (Calibrated metric-family carrier)**.** *A* $\mathsf{CalibratedMetricFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, K, B, C, Pc, E, U, H, T, Q, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the common metric classifier, distance observations, and finite tolerance comparisons used by the family.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row for the displayed compact-metric domain or finite compact-metric witness consumed by calibration readers.*
3. *B is the finite bracket row. It records a displayed list of lower and upper metric-family brackets and the finite tolerance at which the list is admitted.*
4. *C is the* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *row read at the same tolerance and metric source as B.*
5. *Pc is the* $\mathsf{PackingNumber}^{\uparrow}$ *row read against the same metric source and tolerance ledger.*
6. *E is the* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *row comparing the displayed bracket count, covering number, and packing lower bound as one finite entropy budget.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row. It exposes the calibration tolerance, entropy budget, and finite bracket envelope before any uniform-modulus consumer may inspect the family.*
8. *H records componentwise* hsame *transport for the metric, compact-metric, bracket, covering, packing, entropy, uniform-modulus, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from any calibration request through M, K, B, C, Pc, E, U, H.*

10. *Q is the* Pkg *provenance over* $\mathsf{CalibratedMetricFamily}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{PackingNumber}^{\uparrow}$, $\mathsf{FiniteMetricEntropy}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CalibratedMetricFamily}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function spaces, open-cover compactness, measurable envelope selection, countable bracket families, quotient maps, selected limits, or choice.

Theorem 7782.2 (Calibrated metric-family NameCert obligations)**.** *For every accepted* $\mathsf{CalibratedMetricFamily}^{\uparrow}$ *carrier*

$$F = (M, K, B, C, Pc, E, U, H, T, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CalibratedMetricFamily}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, compact-metric source, finite bracket list, covering row, packing row, entropy budget, uniform-modulus handoff, transport, replay, provenance, and local naming rows;*
2. *bracket visibility, requiring every calibration read to inspect the finite bracket row B and its tolerance before it reads an entropy or modulus consequence;*
3. *covering and packing exactness, requiring C and Pc to be read against the same M, K, B tolerance surface;*
4. *entropy compatibility, requiring E to compare only the displayed bracket count, covering upper bound, and packing lower-bound rows;*
5. *uniform-modulus handoff, requiring U to consume the finite entropy and bracket envelope before exporting any modulus-facing data;*
6. *ledger nonescape, excluding host function-space compactness, selected brackets, quotient families, measurable envelopes, countable choice, selected limits, and ambient completeness as local certificate data.*

No calibrated metric-family obligation is stored outside $M, K, B, C, Pc, E, U, H, T, Q, N$.

Proof. Carrier admission is projection from Definition 7782.1. The packet displays the metric source, compact-metric source, finite bracket list, covering and packing rows, entropy ledger, modulus handoff, transport, replay, provenance, and local naming rows.

Bracket visibility comes next. A calibration reader reaches neither E nor U until it has inspected the finite bracket row B and the tolerance shared with the metric and compact-metric sources.

The covering and packing rows are rowwise comparisons. They are read against the same M, K, B surface, so their budgets cannot be replaced by a host covering argument, a selected packing, or a compactness principle outside the packet.

Finally E and U account for entropy and modulus-facing consumption, while H, T, Q, N transport, replay, source, and name the same finite route. Since the carrier contains none of the excluded host or choice coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 7782.3 (Calibrated metric-family entropy handoff)**.** *Every* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CalibratedMetricFamily}^{\uparrow}$ *carrier factors through*

$$M, K, B \longmapsto C, Pc, E \longmapsto U, H, T, Q, N.$$

The consumer receives a finite bracketing and entropy calibration ledger before any uniform-modulus read is exposed. It receives no host function-space compactness theorem, no selected bracket family,

no quotient-family equality, no measurable envelope, no selected limit, and no countable-choice schedule.

Proof. Begin with the carrier projection. The metric and compact-metric rows M, K supply the only source surface, and the bracket row B is the only place where a family envelope can be read.

The entropy-facing rows then open in order. The covering row C and packing row Pc are read at the displayed tolerance, and the finite entropy row E compares those budgets with the finite bracket list.

Only after this entropy route is visible may the handoff row U be consumed by a $\mathsf{UniformModulus}^{\uparrow}$ reader. The transport, replay, provenance, and local naming rows keep the same route attached under hsame and Cont.

The refused host principles have no carrier coordinate. Therefore every entropy or modulus-facing consumer factors through the displayed finite route and cannot replace it by function-space compactness, selected brackets, quotient families, measurable envelopes, selected limits, or countable choice. □

Falsifiable prediction: An accepted $\mathsf{CalibratedMetricFamily}^{\uparrow}$ consumer that exports a uniform-modulus read without first displaying the metric source, compact-metric source, finite bracket row, covering row, packing row, and entropy budget refutes the carrier surface.

Independence witness: metric, compactmetric, finite-metric-entropy, metric-covering-number, packing-number, uniformmodulus: no listed sibling is bijective with $\mathsf{CalibratedMetricFamily}^{\uparrow}$ because this packet jointly carries metric-family source data, finite brackets, covering and packing budgets, entropy calibration, uniform-modulus handoff, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CalibratedMetricFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CalibratedMetricFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7782.1, Definition 7782.2, and Definition 7782.3 seed the finite bracketing, covering, packing, entropy, and uniform-modulus handoff surface for displayed metric-family calibration data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CalibratedMetricFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close empirical-process convergence, measurable envelopes, host function-space compactness, countable bracket families, selected limits, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, bracket visibility, covering and packing exactness, entropy compatibility, uniform-modulus handoff, and ledger nonescape for BEDC.Derived.CalibratedMetricFamilyUp.

Constructive story: A $\mathsf{CalibratedMetricFamily}^{\uparrow}$ packet is generated from MetricUp source data, CompactMetricUp source data, finite bracket rows, MetricCoveringNumberUp and PackingNumberUp budgets, FiniteMetricEntropyUp calibration, UniformModulusUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7783

Zero-Dimensional Compact Space NameCert Construction

Following Engelking and Stone, the $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ packet records the compact zero-dimensional interface as a finite clopen-basis carrier rather than as a host point-set primitive. It consumes the Stone-space route of chapter 7424, the topology rows of chapter 88, and the clopen algebra ledger of chapter 7214. The compact Hausdorff readback is the displayed $\mathsf{CompactHausdorff}^{\uparrow}$ row already exposed by the Stone-space carrier, so the packet is a compact zero-dimensional surface over existing BEDC rows, not a duplicate Stone-space construction.

Definition 7783.1 (Zero-dimensional compact space carrier)**.** *A* $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, T, K, L, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StoneSpace}^{\uparrow}$ *source row. It supplies the Boolean-to-clopen route and the compact Hausdorff readback already certified by the Stone-space packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *public-open row read from the same source. It exposes only the indexed opens generated by the displayed clopen ledger.*
3. *K is the* $\mathsf{CompactHausdorff}^{\uparrow}$ *compact separation row. It is consumed as a readback obligation over S and T, not as a host compactification theorem.*
4. *L is the* $\mathsf{ClopenAlgebra}^{\uparrow}$ *ledger of clopen rows whose finite Boolean operations generate the public-open basis.*
5. *B is the zero-dimensional basis row. It records that every accepted public neighbourhood request is answered by a displayed clopen row from L before it reaches T or K.*
6. *H records componentwise* hsame *transport for the Stone source, topology row, compact Hausdorff row, clopen ledger, basis row, replay, provenance, and local name.*
7. *C records displayed* Cont *replay for the route*

$$S, L, B \longmapsto T, K.$$

8. *P is the* Pkg *provenance over* $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$, $\mathsf{StoneSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactHausdorff}^{\uparrow}$, $\mathsf{ClopenAlgebra}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, T, K, L, B, H, C, P, N$. It has no coordinate for selected ultrafilters, arbitrary host opens, quotient topology equality, open-cover compactness, selected clopen refinements, or bridge checking.

Theorem 7783.2 (Zero-dimensional compact space NameCert obligations)**.** *For every accepted* $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ *carrier*

$$Z = (S, T, K, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}}$ *surface consists exactly of Stone-space source exposure through S, topology public-open exposure through T, compact Hausdorff readback through K, clopen-algebra ledger exposure through L, clopen-basis response through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every compact-topology consumer receives the zero-dimensional compact interface only through those rows.*

Proof. Project the carrier of Definition 7783.1. The packet exposes exactly the Stone source, topology row, compact Hausdorff readback, clopen ledger, basis row, transport, replay, provenance, and local naming row.

The Stone-space source supplies the Boolean-to-clopen route, while L and B keep the clopen generator visible before any public topology read is exported. Thus a consumer cannot replace the basis row by a host open set or by an unrecorded clopen selector.

The compact Hausdorff row is read only as K over the same displayed topology row T. Since the carrier has no selected ultrafilter, arbitrary-open completion, quotient topology equality, selected refinement, or bridge witness coordinate, those objects remain outside the local NameCert surface. □

Theorem 7783.3 (Zero-dimensional compact space clopen-basis stability)**.** *Every transported or replayed* $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ *public-open request preserves the same clopen-basis route*

$$\mathsf{StoneSpace}^{\uparrow} \longmapsto \mathsf{ClopenAlgebra}^{\uparrow} \longmapsto \mathsf{Topology}^{\uparrow}.$$

If two accepted packets are compared by H or replayed by C, the basis row B still answers each public neighbourhood request through the same clopen ledger L before the compact Hausdorff row K is consumed.

Proof. Start with a public-open request in an accepted carrier. The carrier forces the request to pass through B, and B names only clopen rows from L as its basis responses.

Componentwise hsame transport preserves the displayed rows S, T, K, L, B. Displayed Cont replay uses the same route $S, L, B \longmapsto T, K$, so replay cannot insert an arbitrary open, a selected refinement, or a quotient-topology witness.

Therefore every transported or replayed request still has the displayed clopen-basis origin before it is consumed by the compact Hausdorff readback. □

Theorem 7783.4 (Zero-dimensional compact space compact Hausdorff route)**.** *Every compact-facing read of an accepted* $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StoneSpace}^{\uparrow},\ \mathsf{ClopenAlgebra}^{\uparrow},\ \mathsf{Topology}^{\uparrow},\ \mathsf{CompactHausdorff}^{\uparrow},\ H, C, P, N.$$

The route exports compact zero-dimensional structure as displayed clopen-basis and compact Hausdorff rows; it exports no host compact space, open-cover compactness proof, selected subcover, ultrafilter choice, quotient topology equality, or checked standard bridge.

Proof. The compact-facing read first exposes the Stone-space source row S. The clopen algebra ledger L and basis row B then determine the public topology row T before any compact Hausdorff readback is available.

The compact readback is exactly K. It is tied to the same S, T, L, B rows by componentwise transport and displayed replay, and P, N record only that finite package and local name.

Since the accepted carrier has no field for host compact spaces, open-cover compactness, selected subcovers, ultrafilter choice, quotient topology equality, or a standard bridge, the displayed route exhausts the compact Hausdorff handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}$ packet is generated from StoneSpace source rows, Topology public-open rows, CompactHausdorff readback rows, a ClopenAlgebra ledger, clopen-basis responses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7783.1, Definition 7783.2, Definition 7783.3, and Definition 7783.4 seed the compact zero-dimensional interface over StoneSpaceUp, TopologyUp, CompactHausdorffUp, ClopenAlgebraUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeroDimensionalCompactSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected ultrafilters, arbitrary host opens, quotient topology equality, open-cover compactness, selected subcovers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, clopen-basis stability, compact Hausdorff route exposure, transport, replay, provenance, and local naming for BEDC.Derived.ZeroDimensionalCompactSpaceUp.

Chapter 7784

Dyadic Cantor Window NameCert Construction

Following Cantor-space prefix practice and the finite-cover proof of Heine–Cantor, the $\mathsf{DyadicCantorWindow}^{\uparrow}$ packet records finite Boolean prefix windows with dyadic refinement rows. It sits beside the Boolean constructor surface of chapter 32, the finite spine discipline of chapter 36, the dyadic tolerance ledger of chapter 291, the compact metric source of chapter 129, and the uniform-modulus consumer of chapter 369. The packet is a finite prefix-window carrier; it is not a completed Cantor space, a host product space, or a selected infinite branch.

Definition 7784.1 (Dyadic Cantor window carrier)**.** *A* $\mathsf{DyadicCantorWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, L, D, M, U, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *digit row used to discriminate the two Cantor-prefix branches.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *finite prefix row. It stores only finite Boolean words and their prefix extension relation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger. It records the dyadic cell, depth, radius, and comparison budget attached to each finite prefix window.*
4. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row for the compact finite-window source.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that may consume the finite window after dyadic refinement has been exposed.*
6. *R is the prefix refinement row tying B, L, D together. It records how successor windows split by Boolean digit and how their dyadic cells sit inside the parent cell.*
7. *H records componentwise* hsame *transport for Boolean digits, list prefixes, dyadic rows, compact-metric support, uniform-modulus handoff, refinement, replay, provenance, and local name.*
8. *C records displayed* Cont *replay for the finite route*

$$B, L, D, R \longmapsto M, U.$$

9. *P is the* Pkg *provenance over* $\mathsf{DyadicCantorWindow}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCantorWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, L, D, M, U, R, H, C, P, N$. It has no coordinate for an infinite branch, completed Cantor space, host product topology, selected subcover, global fan functional, countable choice, quotient stream equality, or bridge checking.

Lean4 ✓ `BEDC.Derived.DyadicCantorWindowUp.DyadicCantorWindowTasteGate single carrier alignment`

Theorem 7784.2 (Dyadic Cantor window NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCantorWindow}^{\uparrow}$ *carrier*

$$W = (B, L, D, M, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCantorWindow}^{\uparrow}}$ *surface consists exactly of Boolean digit exposure through B, finite prefix exposure through L, dyadic refinement exposure through D, compact-metric support through M, uniform-modulus handoff through U, prefix-refinement stability through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No window obligation is stored outside those rows.*

Proof. Projection from Definition 7784.1 gives exactly the displayed Boolean, finite-list, dyadic, compact-metric, uniform-modulus, refinement, transport, replay, provenance, and local naming rows.

The finite prefix discipline is carried by L, and the only branch split available to the packet is the displayed Boolean digit row B. The dyadic ledger D attaches a finite cell and tolerance budget to those prefixes before compact or uniform consumers can read the packet.

The rows H, C, P, N preserve and name the same finite route. Since there is no field for infinite branches, completed Cantor space, selected subcovers, countable choice, quotient stream equality, or a bridge witness, the listed rows exhaust the local NameCert obligations. □

Theorem 7784.3 (Dyadic Cantor window vision handoff)**.** *Every compact-uniform-continuity consumer reading a* $\mathsf{DyadicCantorWindow}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{Bool}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ H, C, P, N.$$

The handoff realizes a finite prefix-window cover with dyadic refinement rows before the uniform-modulus ledger is consumed; it exports no completed Cantor space, selected infinite branch, global fan functional, selected subcover, host product topology, quotient stream equality, or countable choice.

Proof. Begin with the prefix rows. The Boolean digit row B and finite list row L determine the inspected prefix window, while the refinement row R records the immediate successor windows without selecting an infinite branch.

The dyadic ledger D gives the cell and tolerance budget for that finite prefix. Only after B, L, D, R are displayed may the compact-metric support row M and uniform-modulus handoff row U be read.

Componentwise transport, replay, provenance, and local naming preserve this finite order. Therefore the compact-uniform-continuity consumer receives a finite prefix-window carrier rather than any completed or choice-based Cantor object. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCantorWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCantorWindow}^{\uparrow}$ packet is generated from Bool digit rows, List finite prefix rows, DyadicRatCore refinement ledgers, CompactMetric support, UniformModulus handoff rows, prefix-refinement stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicCantorWindow}\uparrow}$, seed)

Scope closed: Definition 7784.1, Definition 7784.2, and Definition 7784.3 seed the finite dyadic Cantor prefix-window route over BoolUp, ListUp, DyadicRatCoreUp, CompactMetricUp, UniformModulusUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicCantorWindow}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.DyadicCantorWindowUp.DyadicCantorWindowTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close completed Cantor space, selected infinite branches, global fan functionals, selected subcovers, host product topology, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix refinement stability, dyadic compact-window handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCantorWindowUp.

Chapter 7785

A Concrete Naming Certificate for $\mathsf{AscoliLTenCompactFamily}^{\uparrow}$

Following Arzela, Ascoli, and Bishop–Bridges constructive analysis, the $\mathsf{AscoliLTenCompactFamily}^{\uparrow}$ packet records the reusable compact-uniform family carrier used before any selection theorem is invoked. It sits after the compact metric source of chapter 129, the bounded-family ledger of chapter 3522, the equicontinuity family surface of chapter 413, the finite diagonal schedule of chapter 2219, and the terminal readback rows of chapter 301 and chapter 38. The chapter names the compact family carrier only; it does not assert a host function-space compactness theorem, a selected subsequence outside the displayed schedule, quotient equality of functions, or ambient completeness.

Definition 7785.1 (Ascoli L10 uniform compact-family carrier)**.** *An* $\mathsf{AscoliLTenCompactFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{A}_{10} = (K, B, E, F, U, W, Q, R, H, C, P, N),$$

where K is the compact source row read through $\mathsf{CompactMetric}^{\uparrow}$*, B is the bounded-family row read through* $\mathsf{BoundedFunctionFamily}^{\uparrow}$*, E is the equicontinuity row read through* $\mathsf{EquicontinuityFamily}^{\uparrow}$*, F is the displayed compact finite-net row, U is the uniform-modulus handoff row, W is the finite window schedule read through* $\mathsf{StreamName}^{\uparrow}$*, Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise* hsame *transport across source, family, net, window, readback, and seal rows, C is displayed* Cont *replay for the same finite route, P is* Pkg *provenance over the listed sibling packets, and N is the local* $\mathsf{NameCert}_{\mathsf{AscoliLTenCompactFamily}^{\uparrow}}$ *row.*

Theorem 7785.2 (Ascoli L10 finite-net subsequence obligation)**.** *Every accepted* $\mathsf{AscoliLTenCompactFamily}^{\uparrow}$ *consumer that later reads a compact-family or subsequence-facing conclusion must first display the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{BoundedFunctionFamily}^{\uparrow},\ \mathsf{EquicontinuityFamily}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exposes the compact finite-net row F, the uniform modulus row U, the window schedule W, the regular rational readback Q, and the Real seal R before any downstream compact-uniform argument may consume the packet. It exports no host function-space compactness theorem, no selected representative outside W, no quotient stream equality, and no ambient completeness principle.

Proof. Start from the carrier of Definition 7785.1. The compact source, bounded family, equicontinuity family, finite-net, uniform-modulus, window, readback, and seal coordinates are all explicit rows of the tuple.

The finite-net and window rows determine the only subsequence-facing schedule available to later Ascoli consumers. The $\mathsf{RegSeqRat}^{\uparrow}$ row reads that schedule as regular rational data, and the $\mathsf{Real}^{\uparrow}$ row seals only the displayed readback.

Transport and replay are componentwise through H and C, while provenance and local naming are carried by P and N. Since the tuple contains no coordinate for host compactness, selected representatives outside W, quotient stream equality, or ambient completeness, every downstream read factors through the listed finite route. □

Theorem 7785.3 (Ascoli L10 NameCert ledger)**.** *The local* $\mathsf{NameCert}_{\mathsf{AscoliLTenCompactFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-family row, equicontinuity row, compact finite-net row, uniform-modulus row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local NameCert row* N*. The ledger is a reusable compact-uniform family carrier for downstream Ascoli and completion arguments, not a replacement for the selection theorem of chapter 3930 or the diagonal packet of chapter 2219.*

Proof. Read the accepted packet coordinate by coordinate. The rows B, E, F, U, Q, R supply the bounded-family, equicontinuity, compact finite-net, uniform modulus, regular rational readback, and terminal seal obligations.

The rows H, C, P, N then record transport, replay, provenance, and local naming for exactly those obligations. No further NameCert clause can be read, because no further row occurs in the carrier.

Thus the ledger exposes a reusable compact-uniform family surface. Selection, diagonal extraction, quotient equality, and host compactness remain outside the local packet unless a downstream chapter supplies its own displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliLTenCompactFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AscoliLTenCompactFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7785.1, Definition 7785.2, and Definition 7785.3 seed the finite compact-uniform family carrier over bounded-family, equicontinuity, compact finite-net, uniform-modulus, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliLTenCompactFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function-space compactness, selected subsequences outside the displayed schedule, quotient function equality, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net route, readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.AscoliLTenCompactFamilyUp.

Constructive story: An $\mathsf{AscoliLTenCompactFamily}^{\uparrow}$ packet is generated from a compact source row, bounded-family row, equicontinuity-family row, compact finite-net row, uniform-modulus handoff, finite StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7786

A Concrete Naming Certificate for $\mathsf{WhitneyDecomposition}^{\uparrow}$

Following Whitney on decompositions of open sets, the $\mathsf{WhitneyDecomposition}^{\uparrow}$ packet records the finite-window NameCert surface for covering a displayed metric-open region by dyadic cells whose sizes are controlled by their distance to the displayed boundary. It sits over the dyadic grid and tolerance rows of chapter 1658, chapter 291, and chapter 1877, the metric and ball rows of chapter 4230 and chapter 1450, the finite-cover route of chapter 6105, and the located-real comparison surface of chapter 1730. The packet names only a finite decomposition window; it does not assert an infinite locally finite family, selected nearest boundary points, open-cover compactness, quotient cubes, or ambient completeness.

Definition 7786.1 (Whitney decomposition carrier)**.** *A* $\mathsf{WhitneyDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, U, D, Q, A, L, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row with the metric ball and located-distance reads available to the decomposition window;*
2. *U is the displayed open-region row, naming the finite membership tests whose admitted points may be covered;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and dyadic-grid ledger, recording the dyadic scales and coordinate cells used by the window;*
4. *Q is the finite cell inventory, listing the displayed dyadic cells and their parent-child refinement relations;*
5. *A is the admission row, requiring each listed cell to lie inside U through metric-ball certificates;*
6. *L is the local-finiteness ledger, recording finite overlap checks for the displayed cell inventory;*
7. *B is the boundary-distance comparison row, relating each admitted cell diameter to the located distance from that cell to the displayed complement boundary;*
8. *H records componentwise* hsame *transport for the metric, dyadic, cover, overlap, and boundary-distance rows;*
9. *C records displayed* Cont *replay for cell admission, finite-overlap reads, and boundary-distance comparisons;*

10. *P is the* Pkg *provenance over* $\mathsf{WhitneyDecomposition}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *dyadic cover rows, finite metric covers, located-real comparison rows,* BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, U, D, Q, A, L, B, H, C, P, N$.

Theorem 7786.2 (Whitney decomposition NameCert obligations)**.** *Every accepted* $\mathsf{WhitneyDecomposition}^{\uparrow}$ *packet binds exactly the finite NameCert obligation surface of Definition 7786.1: metric-source exposure through X, open-region membership through U, dyadic scale and cell data through D, Q, cell admission through A, finite local-overlap checks through L, boundary-distance control through B, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $W = (X, U, D, Q, A, L, B, H, C, P, N)$. The metric and region rows supply the source carrier and the displayed open-region membership tests, while the dyadic rows supply only the finite cell inventory named by the packet.

The admission, overlap, and boundary rows then own the analytic checks. Cell containment is read through A, local finiteness is read through the finite overlap ledger L, and boundary-distance control is read through B.

Finally H, C, P, N transport, replay, cite, and name the same displayed rows. Since the carrier has no additional coordinate, the NameCert surface cannot expose an infinite decomposition, selected boundary points, open-cover compactness, quotient cubes, or ambient completeness. □

Theorem 7786.3 (Whitney decomposition local finite cover)**.** *For every accepted* $\mathsf{WhitneyDecomposition}^{\uparrow}$ *carrier $W = (X, U, D, Q, A, L, B, H, C, P, N)$, every covered point of the displayed open row U is read through a finite subinventory of Q, and every local-overlap query is answered by the finite ledger L. Thus the public cover read is a finite dyadic-window cover over X and U, not an infinite locally finite family, selected enumeration, open-cover compactness proof, or host topological cover.*

Proof. Begin with a point admitted by U. The cover read must inspect the metric source X, the dyadic scale ledger D, and the finite cell inventory Q before it can name a cell containing the point.

Use the admission row A to keep every selected displayed cell inside the open-region row. The local-overlap ledger L then answers neighbourhood and overlap queries by finite inspection of the same inventory Q.

Transport through H and replay through C preserve the displayed finite inventory. The provenance and local naming rows record no additional cells, so the read remains a finite dyadic cover window and cannot become a global host cover principle. □

Theorem 7786.4 (Whitney decomposition boundary-distance control)**.** *For every accepted* $\mathsf{WhitneyDecomposition}^{\uparrow}$ *carrier $W = (X, U, D, Q, A, L, B, H, C, P, N)$, each public cell read from Q exposes a dyadic diameter estimate and a located-real comparison in B relating that diameter to the displayed distance from the cell to the complement boundary of U. The estimate is read only through* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *dyadic cover rows, and located-real comparison rows; it exports no selected nearest boundary point, no quotient cube equality, no ambient real completeness theorem, and no global Whitney decomposition theorem.*

Proof. Read a cell through the finite inventory Q and its dyadic scale data in D. The dyadic ledger supplies the diameter window, while the metric and ball rows in X supply the distance reads available to the packet.

The boundary-distance comparison is the row B. It compares the displayed dyadic diameter with the located boundary-distance window and records the comparison as part of the finite carrier, not as a selected nearest-boundary construction.

The structural rows H, C, P, N do not add an analytic endpoint. They preserve, replay, source, and name the same metric, dyadic, cover, and comparison rows, so quotient cube equality, ambient completeness, and a global Whitney theorem remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{WhitneyDecomposition}^{\uparrow}$ read exposes a MetricSpace source row, an open-region membership row, DyadicRatCore scale data, a finite dyadic cell inventory, cell-admission witnesses, finite local-overlap checks, boundary-distance comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any cover or boundary-control consumer receives the packet.

Independence witness: whitney-covering-finite, finite-metric-cover, dyadic-cover-refinement, located-real-comparison: no listed sibling supplies the full eleven-row packet $W = (X, U, D, Q, A, L, B, H, C, P, N)$, because each sibling contributes only a finite cover route, a dyadic refinement ledger, or a located comparison surface while this chapter binds the Whitney decomposition window into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{WhitneyDecomposition}^{\uparrow}$ packet is generated from a MetricSpace source row, an open-region row, DyadicRatCore scale data, finite dyadic cell inventory, admission witnesses, finite local-overlap ledger, boundary-distance comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7786.1, Definition 7786.2, Definition 7786.3, and Definition 7786.4 seed the finite Whitney decomposition window over displayed metric, dyadic, cover, overlap, comparison, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhitneyDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close infinite locally finite decompositions, selected nearest boundary points, open-cover compactness, quotient cube equality, ambient completeness, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, dyadic cell inventory, local finite cover readback, boundary-distance control, transport, replay, provenance, and local naming for BEDC.Derived.WhitneyDecompositionUp.

Chapter 7787

A Concrete Naming Certificate for $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$

Following Heine–Cantor in Bishop–Bridges constructive analysis, the $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$ packet records the finite oscillation-window carrier that turns compact finite-net data and pointwise continuous-map moduli into a tolerance-only uniform-modulus read. It realizes the compact-uniform-continuity route of chapter 9237 and sits over the compact metric source of chapter 129, the compact certificate of chapter 60, the continuous-map rows of chapter 130, the rational fold of chapter 37, the finite minimum surface of Definition 9237.5, the uniform-modulus target of chapter 369, and the terminal real-name seal of chapter 38. The packet names a finite oscillation ledger; it does not assert open-cover compactness, selected global moduli, function-space quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7787.1 (Compact finite oscillation modulus carrier)**.** *A* $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, K, F, E, R, M, U, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric carrier and compact finite-net data;*
2. *K is the compact finite-net bundle, naming the centers, coverage rows, and finite* $\mathsf{ProbeBundle}$ *spine used at a tolerance;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map whose oscillation is being controlled;*
4. *E is the positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row requested by the consumer;*
5. *R is the pointwise radius ledger over the centers of K, read through the continuous-map certificate;*
6. *M is the finite rational lower-bound fold over the pointwise radii, using only* $\mathsf{Rat}^{\uparrow}$ *order readback;*
7. *U is the resulting* $\mathsf{UniformModulus}^{\uparrow}$ *ledger from the tolerance row E to the folded precision row;*
8. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison seal used when the oscillation statement is read in real-valued form;*
9. *H records componentwise* hsame *transport for compact, continuous-map, rational-fold, uniform-modulus, and real-seal rows;*

10. *C records displayed* Cont *replay for finite-net coverage, pointwise-radius reads, rational folds, and uniform-modulus readback;*
11. *P is the* Pkg *provenance over* $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteOscillationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, E, R, M, U, T, H, C, P, N$.

Theorem 7787.2 (Compact finite oscillation NameCert obligations)**.** *Every accepted* CompactFiniteOscillationMod *packet binds exactly the finite NameCert obligation surface of Definition 7787.1: compact metric source exposure through X, finite-net coverage through K, continuous-map pointwise modulus exposure through F, R, tolerance readback through E, rational lower-bound folding through M, uniform-modulus output through U, real-valued oscillation sealing through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $O = (X, K, F, E, R, M, U, T, H, C, P, N)$. The compact source and finite-net rows are X, K, while the displayed continuous map and tolerance request are F, E.

The pointwise oscillation data are read through R. The finite rational fold M then consumes the finite radius ledger and produces the precision row used by the uniform-modulus ledger U.

The real-valued read is sealed only through T, and the structural rows H, C, P, N preserve the same finite route. No coordinate supplies open-cover compactness, a selected global modulus, quotient function equality, countable choice, or ambient real completeness. □

Theorem 7787.3 (Compact finite oscillation uniform readback)**.** *For every accepted* CompactFiniteOscillationModu *carrier $O = (X, K, F, E, R, M, U, T, H, C, P, N)$, a consumer requesting oscillation at tolerance E must first read the compact finite-net row K, the continuous-map pointwise radius ledger R, and the rational lower-bound fold M. The public output is the* $\mathsf{UniformModulus}^{\uparrow}$ *row U, optionally sealed by the* $\mathsf{Real}^{\uparrow}$ *comparison row T, and it is not a selected modulus function, open-cover compactness theorem, quotient function-space equality, or ambient completeness claim.*

Proof. Start with the tolerance request E. The compact source row X and finite-net bundle K determine the finite list of center-radius obligations that the continuous-map row must answer.

Read the pointwise radius ledger through F and R. The fold row M performs only finite $\mathsf{Rat}^{\uparrow}$ lower-bound comparison over that ledger, so the resulting precision depends on the displayed tolerance and finite bundle rather than on a selected global modulus.

The uniform readback is the row U. The optional real-valued seal T records the oscillation comparison after the uniform-modulus row is available, and H, C, P, N add no independent analytic endpoint. Thus the packet exports exactly the finite uniform-modulus readback. □

Falsifiable prediction: Every accepted $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$ read exposes CompactMetric source data, a compact finite-net ProbeBundle row, ContinuousMap pointwise modulus rows, RatUp tolerance and lower-bound folds, a UniformModulus output ledger, a RealUp comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any uniform-oscillation consumer receives the packet.

Independence witness: compactuniformcontinuity, compact-uniform-modulus, finitenetminimumfold, subordinatemoduluscover: no listed sibling supplies the full twelve-row packet $O = (X, K, F, E, R, M, U, T, H, C, P, N)$, because each sibling contributes only a finite-net construction, a pointwise-modulus cover, a rational fold, or a uniform-modulus target while this chapter binds the oscillation window into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFiniteOscillationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactFiniteOscillationModulus}^{\uparrow}$ packet is generated from a CompactMetric source row, compact finite-net ProbeBundle data, ContinuousMap pointwise-radius rows, RatUp tolerance and lower-bound folds, a UniformModulus output ledger, a RealUp comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactFiniteOscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7787.1, Definition 7787.2, and Definition 7787.3 seed the compact finite oscillation modulus packet over displayed compact, continuous-map, rational-fold, uniform-modulus, real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactFiniteOscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected global moduli, quotient function-space equality, countable choice, ambient RealUp completeness, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, compact finite-net exposure, pointwise-radius reads, rational lower-bound fold, uniform readback, transport, replay, provenance, and local naming for BEDC.Derived.CompactFiniteOscillatio

Chapter 7788

A Concrete Naming Certificate for $\mathsf{MetricUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures generated by metrics, the $\mathsf{MetricUniformity}^{\uparrow}$ packet records the finite BEDC handoff from displayed metric balls to uniform entourages. It sits between the metric carrier of chapter 57, the uniform-space surface of chapter 339, the complete-metric route of chapter 128, the compact-metric row of chapter 129, the stream-name windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the entourage basis generated by positive Real radii and its completion-facing readers; it does not add host uniform-space primitives, quotient filters, selected limits, open-cover compactness, or ambient Real completeness.

Definition 7788.1 (Metric uniformity carrier)**.** *A* $\mathsf{MetricUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, E, B, C, F, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row whose distance observations and named balls are the source of every entourage request.*
2. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage family generated by positive* $\mathsf{Real}^{\uparrow}$ *radii from M.*
3. *B is the finite ball-basis row, listing the inspected centers, positive radius names, and inclusion witnesses used to cover one requested entourage.*
4. *C is the Cauchy-filter handoff row for completion-facing consumers that read Cauchy data through the generated entourages.*
5. *F is the completion-facing row that exposes the same generated uniformity to* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *stream-window, and regular-readback consumers.*
6. *H stores componentwise* hsame *transports for M, E, B, C, F and the local naming row.*
7. *T stores displayed* Cont *replay from a metric-ball request through positive-radius generation, entourage exposure, Cauchy-filter handoff, and completion-facing readback.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricUniformity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricUniformity}^{\uparrow}}$ *registration row.*

The classifier compares accepted packets only through these rows. There is no coordinate for primitive uniformity independent of the metric row, quotient Cauchy-filter equality, a selected completion point, open-cover compactness, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7788.2 (MetricUniformity NameCert obligations)**.** *For every accepted* $\mathsf{MetricUniformity}^{\uparrow}$ *carrier* $U = (M, E, B, C, F, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricUniformity}^{\uparrow}}$ *obligations are exhausted by carrier admission for the displayed metric row, generated entourage family, finite ball-basis row, Cauchy-filter handoff, completion-facing row, componentwise transport, continuation replay, provenance, and local naming; by positive-radius generation of each entourage from* M *through* B*; by completion-facing consumption through* C *and* F*; and by ledger refusal for host uniform-space primitives, quotient filters, selected completion points, open-cover compactness, countable choice, trichotomy, and ambient Real completeness.*

Proof. Project the carrier of Definition 7788.1. The analytic rows are M, E, B, C, F, and the structural rows are H, T, P, N.

The generated-entourage obligation first reads a positive $\mathsf{Real}^{\uparrow}$ radius and a metric ball from M. The finite basis row B records the inspected centers and inclusions needed to present that ball request as an entourage in E.

Completion-facing consumers are downstream of the generated row. A Cauchy-filter read uses C, and the complete-metric or compact-metric read uses F, only after E and the finite basis data have been exposed.

Transport, replay, provenance, and local naming preserve exactly this route. Since the packet has no independent coordinate for host uniformity, quotient filters, selected completion points, open-cover compactness, countable choice, trichotomy, or ambient Real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7788.3 (MetricUniformity entourage basis)**.** *Every entourage exposed by an accepted* $\mathsf{MetricUniformity}^{\uparrow}$ *packet factors through a finite metric-ball basis row. More precisely, a consumer of* E *must display the metric row* M*, a positive* $\mathsf{Real}^{\uparrow}$ *radius, and the finite basis row* B *before the entourage can be read as a* $\mathsf{UniformSpace}^{\uparrow}$ *surface. The read cannot replace that finite route by a primitive host entourage family or by quotient equality of filters.*

Proof. Start with an accepted packet $U = (M, E, B, C, F, H, T, P, N)$. The carrier definition makes E generated data rather than a primitive uniformity row.

To read an entourage, the continuation replay T first exposes the metric observation in M and the positive radius name. The row B then supplies the finite ball-basis evidence for the requested scale and center data.

Only after those rows are displayed may the consumer read E as a $\mathsf{UniformSpace}^{\uparrow}$ surface. Componentwise transport in H preserves the same M, B, E route, while P and N record no independent host entourage source. □

Theorem 7788.4 (MetricUniformity Cauchy-filter handoff)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *completion-facing read of an accepted* $\mathsf{MetricUniformity}^{\uparrow}$ *packet must pass through the generated entourage row* E*, the finite basis row* B*, the Cauchy-filter handoff* C*, and the completion-facing row* F*. Thus metric completion consumers see a finite Metric-to-UniformSpace route, not a host completion object, selected Cauchy limit, quotient filter, or ambient completeness principle.*

Proof. Begin with a completion-facing consumer. Carrier admission gives access only to $M, E, B, C, F, H, T, P, N$.

The entourage read is forced by Definition 7788.3. Hence M and B are visible before any uniform-space consumer can inspect E.

The Cauchy-filter handoff C is the next available completion row, and F is the row through which complete-metric, compact-metric, stream-window, regular-readback, and Real-seal consumers read the generated uniformity.

The structural rows preserve and name the same finite route. Because no carrier row stores a host completion object, selected limit, quotient filter, countable-choice schedule, open-cover compactness, trichotomy, or ambient Real completeness, none of those objects can be exported by the handoff. □

Falsifiable prediction: An accepted $\mathsf{MetricUniformity}^{\uparrow}$ read is refuted if a completion-facing consumer obtains a $\mathsf{UniformSpace}^{\uparrow}$ entourage or Cauchy-filter handoff without displaying the metric row, a positive Real radius, the finite ball-basis row, generated entourage row, transport, replay, provenance, and local NameCert rows.

Independence witness: metric, uniformspace, completemetric, compactmetric, streamname, regseqrat, real: none alone is bijective to $\mathsf{MetricUniformity}^{\uparrow}$ because this packet binds metric balls, positive Real radius generation, finite basis evidence, generated entourages, Cauchy-filter handoff, completion-facing readback, transport, replay, provenance, and local naming as one bridge.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricUniformity}^{\uparrow}$ packet is generated from MetricUp distance rows, positive RealUp radius requests, finite ball-basis evidence, UniformSpaceUp entourage rows, Cauchy-filter handoff rows, completion-facing rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7788.1, Definition 7788.2, Definition 7788.3, and Definition 7788.4 seed the finite Metric-to-UniformSpace bridge for completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host uniform-space primitives, quotient Cauchy-filter equality, selected completion points, open-cover compactness, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius generation, finite entourage-basis coverage, Cauchy-filter handoff, completion-facing readback, transport, replay, provenance, and local NameCert coverage for the displayed packet.

Chapter 7789

A Concrete Naming Certificate for $\mathsf{MengerCurve}^{\uparrow}$

Following Menger and continuum theory, the $\mathsf{MengerCurve}^{\uparrow}$ packet records the universal one-dimensional compact metric continuum by finite cube-refinement obligations rather than by an ambient subset of Euclidean space. It sits between the compactum surface of chapter 7046, the continuum boundary of chapter 2160, the planar comparison surface of chapter 3960, the compact metric carrier of chapter 129, and the metric source of chapter 57. The packet gives L10 compact and metric consumers a finite refinement ledger for the curve; it does not import open-cover compactness, host hyperspace equality, selected points, quotient continua, or ambient completeness.

Definition 7789.1 (Menger curve carrier)**.** *A* $\mathsf{MengerCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, K, L, S, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the displayed distance comparisons read by every cube-refinement request;*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, exposing compact metric coverage before any continuum or cube row is consumed;*
3. *L is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, recording connected locally connected compact-continuum data as a finite NameCert dependency;*
4. *S is the* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *comparison row, recording the planar hole-boundary sibling used only as a contrast surface;*
5. *Q is the* $\mathsf{MengerCompactum}^{\uparrow}$ *row, carrying the compactum and covering-dimension surface already named for Menger compacta;*
6. *R is the finite cube-refinement ledger, whose entries name stage, retained subcubes, face-incidence tests, and adjacency witnesses;*
7. *E is the* $\mathsf{MengerCoverCompactness}^{\uparrow}$ *row, recording finite cover requests and finite subcover witnesses for the inspected refinement stage;*
8. *H records componentwise* hsame *transport for $M, K, L, S, Q, R, E, P, N$;*
9. *C records displayed* Cont *replay through the metric, compact, continuum, compactum, cube-refinement, cover, provenance, and local naming rows;*

10. *P is the* Pkg *provenance over* $\mathsf{MengerCurve}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{MengerCompactum}^{\uparrow}$, $\mathsf{MengerCoverCompactness}^{\uparrow}$, $\mathsf{MengerNobelingEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerCurve}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, a host continuum object, a quotient of compact subsets, a selected point sequence, or an ambient Euclidean equality oracle.

Theorem 7789.2 (Menger curve NameCert obligations)**.** *For every accepted* $\mathsf{MengerCurve}^{\uparrow}$ *carrier $U = (M, K, L, S, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MengerCurve}^{\uparrow}}$ *surface consists exactly of metric exposure through M; compact metric finite-net exposure through K; Peano-continuum exposure through L; planar comparison exposure through S; Menger compactum exposure through Q; finite cube-refinement exposure through R; finite cover-compactness exposure through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Theorem 7789.3 (Menger curve finite cube refinement)**.** *Every compact, metric, continuum, or embedding-facing consumer of an accepted* $\mathsf{MengerCurve}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{PeanoContinuum}^{\uparrow},\ \mathsf{SierpinskiCarpet}^{\uparrow},\ \mathsf{MengerCompactum}^{\uparrow},\ R,\ \mathsf{MengerCoverCompactness}^{\uparrow},$$

through the displayed rows $M, K, L, S, Q, R, E, H, C, P, N$ before treating the packet as the Menger curve. The cube-refinement ledger R is consumed as finite subcube incidence and adjacency data; it cannot be replaced by open-cover compactness, a host universal-curve theorem, quotient compactum equality, or an ambient Euclidean subset.

Falsifiable prediction: An accepted $\mathsf{MengerCurve}^{\uparrow}$ consumer that reaches compact, metric, continuum, or embedding-facing data while omitting the finite cube-refinement ledger R refutes the carrier of Definition 7789.1.

Independence witness: menger-compactum, peano-continuum, sierpinski-carpet, compactmetric, metric: $\mathsf{MengerCurve}^{\uparrow}$ is not bijective to any listed sibling because this packet binds metric source rows, compact finite-net rows, continuum rows, planar comparison rows, Menger compactum rows, finite cube-refinement rows, finite cover-compactness rows, transport, replay, provenance, and local naming into one curve certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerCurve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MengerCurve}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite-net row, a PeanoContinuumUp row, a SierpinskiCarpetUp comparison row, a MengerCompactumUp row, a finite cube-refinement ledger, a MengerCoverCompactnessUp row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MengerCurve}^{\uparrow}}$, seed)

Scope closed: Definition 7789.1, Definition 7789.2, and Definition 7789.3 seed the finite cube-refinement NameCert surface for the displayed Menger curve packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MengerCurve}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, host universal-curve uniqueness, quotient compactum equality, ambient Euclidean subset equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cube-refinement exposure, compact metric exposure, continuum exposure, Menger compactum exposure, cover-compactness exposure, transport, replay, provenance, and local naming for BEDC.Derived.MengerCurveUp.

Chapter 7790

A Concrete Naming Certificate for $\mathsf{ChannelTerminationBoundary}^{\uparrow}$

The $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ packet realizes the local delimiter boundary of chapter 9369 as finite NameCert data. It isolates the active-channel fact that a legal event ends at the first displayed 11 while refusing any conversion of that local boundary into a global halting classifier. The packet sits beside the channel/halting separator of chapter 3996, the type-checking classifier membership surface of chapter 3911, the MetaCIC parallel-diamond frontier of chapter 4238, the subject-reduction boundary of chapter 483, and the classifier boundary of chapter 595.

Definition 7790.1 (Channel termination boundary carrier)**.** *A* $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, E, T, D, R, F, H, C, P, N)$$

with the following displayed rows:

1. *A is the active binary alphabet row, recording the source fragments $0, 1$ and channel fragments $0, 10, 11$;*
2. *E is the legal channel-event row, recording membership in $(0 \mid 10)^*11$ for the inspected finite event;*
3. *T is the terminator row, recording that the first adjacent 11 is the local endpoint;*
4. *D is the decoder-state row, recording the finite scan state before and after the local endpoint;*
5. *R is the refusal row separating local delimiter termination from certificate-level halting;*
6. *F is the finite trace row, recording the inspected event prefix and the endpoint witness;*
7. *H records componentwise* hsame *transport for A, E, T, D, R, F, P, N;*
8. *C records displayed* Cont *replay through the active alphabet, legal-event, terminator, decoder, refusal, finite trace, provenance, and local naming rows;*
9. *P is the* Pkg *provenance over* $\mathsf{ChannelTerminationBoundary}^{\uparrow}$, $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$, $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{ClassifierBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChannelTerminationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a total halting oracle, evaluator equality, subject-reduction discharge, residual-substitution discharge, quotient equality, or kernel consistency.

Theorem 7790.2 (Channel termination boundary NameCert obligations)**.** *For every accepted* $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ *carrier* $B = (A, E, T, D, R, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChannelTerminationBoundary}^{\uparrow}}$ *surface consists exactly of active alphabet exposure through* A*; legal event exposure through* E*; first-terminator exposure through* T*; decoder-state exposure through* D*; local/global refusal exposure through* R*; finite trace exposure through* F*; componentwise transport through* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Theorem 7790.3 (Channel termination boundary local delimiter)**.** *Every accepted* $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ *packet factors local event completion through the finite route*

$$A,\ E,\ T,\ D,\ F,\ H,\ C,\ P,\ N$$

and factors refusal of global halting through R*. A consumer may use the first displayed* 11 *as the endpoint of the current legal channel event, but it may not treat that endpoint as type-checking classifier membership, MetaCIC parallel-diamond closure, subject reduction, classifier-boundary collapse, evaluator equality, quotient equality, or kernel consistency.*

Falsifiable prediction: A claimed $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ consumer that decides certificate-level halting while displaying only the local legal-event row E and terminator row T refutes the carrier of Definition 7790.1.

Independence witness: type-checking-classifier-membership, metacic-parallel-diamond-frontier, subjectreduction, classifierboundary: $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ is not bijective to any listed sibling because it carries active alphabet, legal event, terminator, decoder-state, local/global refusal, finite trace, transport, replay, provenance, and local NameCert rows. Type-checking classifier membership lacks the local delimiter trace; the MetaCIC parallel-diamond frontier lacks the active-channel endpoint; subject reduction lacks the delimiter/refusal split; classifier boundary lacks the concrete first-11 event row.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{ChannelTerminationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ChannelTerminationB}$ TasteGate instances. Those instances must preserve $A, E, T, D, R, F, H, C, P, N$ as distinct displayed fields before any checked theorem may cite $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ as a field-faithful local delimiter boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{ChannelTerminationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChannelTerminationBoundary}^{\uparrow}$ packet is generated from active alphabet rows, legal event rows, first-terminator rows, decoder-state rows, local/global refusal rows, finite trace rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChannelTerminationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7790.1, Definition 7790.2, and Definition 7790.3 seed the finite local delimiter boundary separating channel-event termination from global halting claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChannelTerminationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, global halting decidability, subject reduction, full parallel diamond, evaluator equality, quotient equality, kernel consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal-event exposure, first-terminator exposure, decoder-state exposure, local/global refusal, finite trace exposure, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.ChannelTerminationBoundaryUp.

Chapter 7791

A Concrete Naming Certificate for $\mathsf{MetricCoveringDimension}^{\uparrow}$

Following Hurewicz and Wallman on covering dimension and Bishop–Bridges finite metric compactness, the $\mathsf{MetricCoveringDimension}^{\uparrow}$ packet records finite-order metric covers over an explicit L10 compact-metric substrate. It sits after the metric source of chapter 57, the compact finite-net rows of chapter 129, the finite refinement route of chapter 4205, and the dyadic mesh ledger of chapter 358. The packet names only finite cover rows, order ledgers, star-refinement or mesh handoff rows, transport, replay, provenance, and local naming; it is not an arbitrary open-cover compactness theorem, a choice-selected refinement principle, a paracompactness assertion, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7791.1 (Metric covering dimension carrier)**.** *A* $\mathsf{MetricCoveringDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, K, U, O, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, exposing the accepted* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *distance interface used by the cover cells.*
2. *K is the compact-metric substrate row, exposing the finite-net and finite-cover boundary over the same metric source.*
3. *U is the finite metric-cover row, listing named center-radius cells or named finite cover cells over M, K.*
4. *O is the finite order-bound row, recording that every displayed point-history inspection meets at most the chosen number of cover cells.*
5. *S is the star-refinement or mesh row, routing the finite cover through the dyadic mesh handoff and the sibling covering-dimension refinement surface.*
6. *H records componentwise* hsame *transport for M, K, U, O, S and the certificate rows.*
7. *C records displayed* Cont *replay from metric source and compact finite-net data through finite cover inspection, order checking, star-refinement or mesh handoff, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricCoveringDimension}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{LebesgueCoveringDimensi}$ $\mathsf{DyadicMesh}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricCoveringDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by transporting these rows. It has no coordinate for arbitrary open covers, host paracompactness, selected subcovers outside the displayed finite bundle, quotient compacta, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7791.2 (Metric covering dimension NameCert obligations)**.** *For every accepted* MetricCoveringDimension *carrier*

$$D = (M, K, U, O, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCoveringDimension}^{\uparrow}}$ *surface consists exactly of metric source exposure through M, compact-metric finite-net exposure through K, finite cover admission through U, order-bound checking through O, star-refinement or dyadic-mesh handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7791.1. The coordinates M, K, U, O, S are the only rows that expose metric source data, compact finite-net data, finite cover cells, overlap order, and mesh or star-refinement handoff.

The structural coordinates H, C, P, N preserve that route. Transport is componentwise, replay follows the displayed finite cover inspection, provenance names the same upstream packets, and the local naming row certifies no additional object.

Since arbitrary open covers, selected refinements, paracompactness data, quotient compacta, and ambient completeness are absent from the carrier, no local NameCert obligation can export them. The listed rows therefore exhaust the certificate surface. □

Theorem 7791.3 (Metric covering dimension refinement stability)**.** *Let $D = (M, K, U, O, S, H, C, P, N)$ and $D' = (M', K', U', O', S', H', C', P', N')$ be accepted* $\mathsf{MetricCoveringDimension}^{\uparrow}$ *carriers. If the displayed* hsame *transport identifies the metric source rows, compact-metric rows, finite cover rows, order ledgers, and star-refinement or mesh rows componentwise, then every finite-order metric-cover consumer of D replays as a finite-order metric-cover consumer of D' through the same bound and the same mesh handoff route.*

Proof. Read the consumer through the NameCert obligations of Definition 7791.2. The consumer may inspect only M, K, U, O, S before structural replay and provenance.

Apply the assumed componentwise hsame transport to those rows. The metric source, compact substrate, finite cover, order ledger, and star-refinement or mesh handoff are carried to the corresponding rows of D' without adding a new cover selector.

Replay through C' and source the result through P', N'. The order bound remains the displayed bound carried by the transported order ledger, and the mesh handoff remains the displayed handoff carried by S'. Thus refinement stability is internal to the finite packet. □

Theorem 7791.4 (Metric covering dimension finite mesh handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, LebesgueCoveringDimen $\mathsf{DyadicMesh}^{\uparrow}$, *or* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCoveringDimension}^{\uparrow}$ *packet must traverse the finite route*

$$M, K, U, O, S, H, C, P, N$$

before it can use a dimension bound. In particular, the dimension read is a finite metric-cover and mesh handoff over the compact substrate, not a general open-cover compactness principle, an arbitrary paracompactness result, a choice-selected cover refinement, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the accepted carrier. The source metric row M and compact-metric row K must be exposed before any finite cover cell in U can be read.

The order ledger O and the handoff row S then place the cover inside the finite covering-dimension and dyadic-mesh interfaces. A consumer that omits those rows has not displayed the finite order bound or the mesh route.

Finally H, C, P, N transport, replay, source, and name the same finite route. Because the carrier has no coordinate for arbitrary open-cover compactness, choice, paracompactness, quotient compacta, or ambient Real completeness, the finite mesh handoff is the entire exported surface. □

Falsifiable prediction: An accepted $\mathsf{MetricCoveringDimension}^{\uparrow}$ read is refuted if a consumer exports a covering-dimension bound without displaying the metric source row, compact-metric finite-net row, finite metric-cover row, finite order-bound row, star-refinement or dyadic-mesh handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metric, compactmetric, lebesgue-covering-dimension, dyadicmesh: MetricCoveringDimen is not bijective to any listed sibling because it binds finite metric-cover cells, compact finite-net support, overlap order, mesh handoff, transport, replay, provenance, and local naming into one finite certificate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCoveringDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCoveringDimension}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite-net row, finite metric-cover rows, finite order-bound rows, a star-refinement or dyadic-mesh handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCoveringDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7791.1, Definition 7791.2, Definition 7791.3, and Definition 7791.4 seed the finite metric-cover dimension route over MetricUp, CompactMetricUp, LebesgueCoveringDimensionUp, and DyadicMeshUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCoveringDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general open-cover compactness, choice-selected subcovers, arbitrary paracompactness, quotient compacta, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover admission, order-bound exactness, star-refinement or mesh handoff, compact-metric substrate compatibility, transport, replay, provenance, and local naming for BEDC.Derived.MetricCoveringDimensionUp.

Chapter 7792

A Concrete Naming Certificate for $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's order-theoretic treatment of envelopes, the $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$ packet records the finite lower-envelope regularization surface read by Real and Completion consumers. It sits after the lower-semicontinuity route of chapter 1536, the epigraph route of chapter 6224, the regular-rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names displayed envelope data only: it does not form a host infimum or supremum, quotient functions, choose minimizing representatives, or import $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7792.1 (Lower-envelope regularization carrier)**.** *A* $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, F, E, O, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain-window row naming the finite points, neighbourhood probes, or compact subwindows on which the lower envelope is inspected.*
2. *F is the source-function observation row, exposing only the displayed value windows relevant to the regularization request.*
3. *E is the* $\mathsf{Epigraph}^{\uparrow}$ *row that packages the value and lower-bound comparison surface.*
4. *O is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *obligation row, recording the closed lower-envelope condition read from the epigraph surface.*
5. *Q is the finite lower-envelope ledger, listing candidate lower bounds, tolerance cells, and the accepted comparison witnesses for the inspected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite ledger Q.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the regularized value read through R.*
8. *H records componentwise* hsame *transport for the domain window, source observations, epigraph row, lower-semicontinuity row, lower-envelope ledger, readback row, real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each lower-envelope request through D, F, E, O, Q, R, S, H.*

10. *P is the* Pkg *provenance over* $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerEnvelopeRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, F, E, O, Q, R, S, H, C, P, N$. It has no coordinate for a host supremum or infimum, quotient equality of functions, a choice-selected minimizer, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7792.2 (Lower-envelope regularization NameCert obligations)**.** *For every accepted* LowerEnvelopeRegula *carrier*

$$L = (D, F, E, O, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerEnvelopeRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; domain-window exposure for D; source-function observation for F; epigraph comparison for E; lower-semicontinuity obligation exposure for O; finite lower-envelope ledger exactness for Q;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal nonescape for R, S; componentwise transport, replay, provenance, and local naming for H, C, P, N. No local obligation is stored in host suprema, quotient functions, selected minimizers, countable choice, or ambient real completeness.*

Proof. Project the accepted carrier $L = (D, F, E, O, Q, R, S, H, C, P, N)$. The projection exposes exactly the domain window, source observation row, epigraph row, lower-semicontinuity row, finite envelope ledger, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

Read an envelope request through D and F. The row E supplies the epigraph comparison, and O ties that comparison to the lower-semicontinuity surface before the lower-envelope ledger Q is available.

The rows R and S remain downstream of Q. Thus a Real-facing consumer obtains a regular-rational readback and a terminal seal only from the displayed finite lower-envelope ledger.

The structural rows H, C, P, N transport, replay, source, and name that route. Since the carrier contains no host supremum, quotient-function equality, selected minimizer, choice principle, or completeness coordinate, the listed rows exhaust the local certificate. □

Theorem 7792.3 (Lower-envelope regularization L10 handoff)**.** *Every Real-facing or calculus-facing consumer of an accepted* $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$ *packet must first read the finite route*

$$D, F \longmapsto E, O \longmapsto Q \longmapsto R, S, H, C, P, N.$$

In particular, the lower-envelope value reaches a $\mathsf{Real}^{\uparrow}$ *seal only after the consumer has passed through the sibling surfaces* $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *in that order. The handoff exports no host supremum, quotient function, selected minimizer, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier and a Real-facing request. The request first exposes the domain-window and source-observation rows D, F; no lower-envelope ledger can be read before those rows are present.

Next pass through the epigraph and lower-semicontinuity rows. The row E supplies the value/lower-bound comparison surface, and O records the lower-semicontinuity obligation that lets the finite lower-envelope ledger Q be read as regularization data.

Finally read R and S. The regular-rational readback R is produced from Q, and the terminal seal S receives only that readback together with the structural rows H, C, P, N.

Because the packet has no coordinate for host suprema, quotient functions, selected minimizers, countable choice, or ambient completeness, the displayed route is the whole L10 handoff. □

Falsifiable prediction: An accepted $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the epigraph comparison, lower-semicontinuity row, finite lower-envelope ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: lowersemicontinuous, epigraph, regseqrat, real: no listed sibling supplies the whole eleven-row packet $L = (D, F, E, O, Q, R, S, H, C, P, N)$, because each sibling contributes only the lower-semicontinuity, epigraph, regular-rational, or real-seal face while this chapter binds them into one finite lower-envelope regularization route.

Closure status for $\mathsf{NameCert}_{\mathsf{LowerEnvelopeRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerEnvelopeRegularization}^{\uparrow}$ packet is generated from domain windows, source-function observations, an Epigraph row, a LowerSemicontinuous row, a finite lower-envelope ledger, a RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerEnvelopeRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7792.1, Definition 7792.2, and Definition 7792.3 seed the finite lower-envelope regularization route over $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerEnvelopeRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host suprema or infima, quotient functions, choice-selected minimizers, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, epigraph comparison, lower-semicontinuity exposure, finite lower-envelope ledger exactness, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LowerEnvelopeRegularizationUp.

Chapter 7793

A Concrete Naming Certificate for $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$

Following Bishop and Bridges on regulated functions and one-sided limits, the $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ packet records the paired side-limit surface carried by a regulated real function. It sits over the source route of chapter 1906, the one-sided limit interface of chapter 5841, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance rows of chapter 291, the terminal real seal of chapter 38, the cadlag path carrier of chapter 5968, and the path-topology consumer of chapter 6693. The packet packages left and right side data as finite BEDC rows; it does not import host function quotients, selected limiting values outside the displayed rows, path-space completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7793.1 (Regulated left–right limit carrier)**.** *A* $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, A, W^-, W^+, R^-, R^+, D, E^-, E^+, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, read through the regulated-function route of chapter 1906.*
2. *A is the finite anchor ledger naming the endpoint, jump site, or interior cell at which the paired side read is taken.*
3. *W^- and W^+ are the left and right* $\mathsf{StreamName}^{\uparrow}$ *approach-window rows.*
4. *R^- and R^+ are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
5. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two side windows and their real seals.*
6. *E^- and E^+ are the terminal* $\mathsf{Real}^{\uparrow}$ *seals for the displayed left and right side limits.*
7. *H records componentwise* hsame *transport for the regulated source, anchor ledger, side windows, regular readbacks, dyadic tolerance ledger, real seals, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a regulated-function request through the anchor, the two side-window rows, the readback rows, the dyadic comparison ledger, and the two real seals.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CadlagRealFunction}^{\uparrow}$, $\mathsf{SkorokhodTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedLeftRightLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $F, A, W^-, W^+, R^-, R^+, D, E^-, E^+, H, C, P, N$. No coordinate stores a host quotient of functions, a path-space completion theorem, a selected limit outside the displayed side seals, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7793.2 (Regulated left–right limit NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ *carrier*

$$L = (F, A, W^-, W^+, R^-, R^+, D, E^-, E^+, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedLeftRightLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated source, anchor ledger, left and right StreamName windows, regular rational readbacks, dyadic tolerance row, left and right Real seals, componentwise transport, displayed replay, provenance, and local naming. The side-window exactness clause requires W^- and W^+ to be consumed as paired* $\mathsf{OneSidedLimit}^{\uparrow}$ *rows over the same source row F and anchor ledger A. The nonescape clause requires every cadlag or path-topology consumer to remain downstream from $F, A, W^-, W^+, R^-, R^+, D, E^-, E^+, H, C, P, N$.*

Proof. Project the carrier of Definition 7793.1. Its displayed inventory is exactly the regulated source row, anchor ledger, two side-window rows, two readback rows, the shared dyadic tolerance ledger, two real seals, and the structural rows H, C, P, N.

The side-window clause is read from A, W^-, W^+, R^-, R^+, D. Both windows are attached to the same anchor and source row before either regular-rational readback or dyadic comparison row is available.

The terminal seals E^- and E^+ are downstream of those rows. Transport and replay preserve the same paired route, while provenance and local naming record only the listed dependencies.

Since the carrier contains no host function quotient, selected external limit, path-space completion theorem, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, the local NameCert cannot export any of those principles. □

Theorem 7793.3 (Regulated left–right limit regulated handoff)**.** *Every accepted* $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ *packet exposes its paired side limits only after the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route has been displayed. The public handoff factors through*

$$F, A \longmapsto W^-, W^+ \longmapsto R^-, R^+ \longmapsto D \longmapsto E^-, E^+ \longmapsto H, C, P, N,$$

with F read through Definition 1906.8. Thus the packet supplies a finite paired side-limit interface for regulated functions, not a host-level function object or a selected path value outside the displayed seals.

Proof. Start with the source row F of Definition 7793.1. By Definition 1906.8, any consumer of paired one-sided limits must keep the regulated-function route visible before one-sided limit rows or terminal Real seals are consumed.

The anchor ledger A then fixes the endpoint, jump site, or interior cell for the paired read. The two StreamName windows W^- and W^+ are therefore side-tagged observations over the same source and anchor.

Regular rational readback and dyadic comparison are downstream. The rows R^-, R^+ and D are read before the Real seals E^-, E^+, while H, C, P, N preserve, replay, source, and name that finite route. □

Theorem 7793.4 (Regulated left–right limit Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CadlagRealFunction}^{\uparrow}$*, or* $\mathsf{SkorokhodTopology}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ *packet receives the left and right seals only through the displayed route*

$$F, A, W^-, W^+, R^-, R^+, D, E^-, E^+, H, C, P, N.$$

The route may be consumed as cadlag side-window data or as path-topology jump data, but it exports no quotient of functions, no path-space completeness theorem, no selected limit outside E^-, E^+*, no open-cover compactness principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 7793.3. It places the two Real seals behind the regulated source, anchor ledger, side-window rows, readback rows, and dyadic comparison row.

A cadlag consumer reads the same paired seals as left and right path limits, while a Skorokhod-topology consumer reads them as finite jump-window data. In both cases the source route remains the displayed regulated-function packet and the side data remains the paired $\mathsf{OneSidedLimit}^{\uparrow}$ hand-off.

The carrier has no coordinate for a host function quotient, a completed path space, an external selected value, open-cover compactness, or ambient Real completeness. Therefore any Real, cadlag, or topology-facing read remains inside the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ read is refuted if it reaches a $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CadlagRealFunction}^{\uparrow}$, or $\mathsf{SkorokhodTopology}^{\uparrow}$ consumer without displaying the regulated source, anchor ledger, paired StreamName windows, paired RegSeqRat readbacks, shared DyadicRatCore tolerance row, paired Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regulated-function, one-sided-limit, streamname, regseqrat, dyadicrat-core, real, cadlag-real-function, skorokhod-topology: $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ is not bijective to any listed sibling because it binds a regulated source, a shared anchor, paired side windows, paired regular-rational readbacks, one dyadic comparison ledger, paired Real seals, transport, replay, provenance, and local naming as one finite side-limit packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedLeftRightLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegulatedLeftRightLimit}^{\uparrow}$ packet is generated from a RegulatedFunction source row, an anchor ledger, paired StreamName side windows, paired RegSeqRat readbacks, a DyadicRatCore tolerance row, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedLeftRightLimit}^{\uparrow}}$, seed)

Scope closed: Definition 7793.1, Definition 7793.2, Definition 7793.3, and Definition 7793.4 seed the paired side-limit route between regulated functions, one-sided limits, cadlag paths, and Skorokhod topology consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedLeftRightLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function quotients, path-space completeness, selected external limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.RegulatedLeftRightLimitUp covering carrier admission, side-window exactness, dyadic comparison, paired Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 7794

A Concrete Naming Certificate for $\mathsf{CauchyCondensationPacket}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationPacket}^{\uparrow}$ packet records Cauchy's condensation test for positive decreasing real series as finite BEDC rows. It sits after the finite partial-sum surface of $\mathsf{RealSeries}^{\uparrow}$ in chapter 1581, the series-tail criterion of chapter 2015, the positive comparison route of chapter 7665, the dyadic arithmetic core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite condensation route; it is not a quotient of term streams, a selected infinite sum, countable choice, order trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7794.1 (Cauchy condensation packet carrier)**.** *A* $\mathsf{CauchyCondensationPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, S, L, M, R, E, H, Q, P, N)$$

with the following displayed rows:

1. *A is the positive decreasing source-series row. It records a* $\mathsf{RealSeries}^{\uparrow}$ *term surface whose inspected terms are nonnegative and monotone decreasing on the accepted finite windows.*
2. *S is the dyadic block schedule. It names the finite powers-of-two cut points and the requested block windows as* $\mathsf{DyadicRatCore}^{\uparrow}$ *data rather than as a host exponentiation oracle.*
3. *L is the block-sum ledger. It records the displayed comparison between each source block and its condensed representative block sum.*
4. *M is the monotone comparison row. It exposes the finite inequalities that bound every term in a displayed dyadic block by the appropriate endpoint terms.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff reached from the source windows, block schedule, ledger, and comparison row.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after A, S, L, M, R have been displayed.*
7. *H records componentwise* hsame *transport for the source series, dyadic schedule, block ledger, monotone comparison, regular handoff, real seal, replay, provenance, and local naming rows.*
8. *Q records displayed* Cont *replay from a requested series-tail read through the dyadic block schedule, block-sum ledger, monotone comparison row, regular handoff, and real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationPacket}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchySeriesCriterion}^{\uparrow}$, $\mathsf{PositiveSeriesComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationPacket}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, S, L, M, R, E, H, Q, P, N$. It has no coordinate for a host convergence theorem, quotient stream equality, a selected sum, countable choice, order trichotomy, or ambient completeness.

Theorem 7794.2 (Cauchy condensation packet NameCert obligations)**.** *For every accepted* CauchyCondensationPa *carrier $K = (A, S, L, M, R, E, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for the positive decreasing source series, dyadic block schedule, block-sum ledger, monotone comparison row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. No local obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7794.1. Its analytic inventory is exactly the source series row A, the dyadic block schedule S, the block ledger L, the monotone comparison row M, the regular readback R, and the real seal E.

The condensation obligations are finite. The source series is inspected only through displayed windows, while S and L record the finite powers-of-two blocks and their block sums as $\mathsf{DyadicRatCore}^{\uparrow}$ data.

The monotone comparison row M supplies the only route from source terms to block bounds. The handoff R and seal E are therefore available only after the source windows, schedule, ledger, and comparison rows have been displayed.

Finally H, Q, P, N transport, replay, source, and name the same route. Since the carrier has no row for quotient streams, selected sums, countable choice, order trichotomy, host convergence, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7794.3 (Cauchy condensation packet real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or completion-facing read from an accepted* $\mathsf{CauchyCondensationPacket}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the condensation read reaches the terminal real seal only by displaying the positive decreasing source-series windows, the dyadic block schedule, the block-sum ledger, the monotone comparison row, and the regular rational handoff.

Proof. Start with a real-facing read of an accepted carrier. The source series row A exposes only finite term windows, so the read begins at the $\mathsf{StreamName}^{\uparrow}$ surface.

Next read S, L, M. These rows provide the dyadic block schedule, block-sum ledger, and monotone comparison evidence as $\mathsf{DyadicRatCore}^{\uparrow}$ data before any regular rational handoff may be consumed.

The row R is the unique $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the displayed condensation route. Once R is present, the terminal seal E may be read, and H, Q, P, N preserve and name the same route.

Thus every real- or completion-facing condensation read factors through the displayed stream windows, dyadic block evidence, regular rational handoff, and real seal. No bypass supplies quotient stream equality, selected sums, countable choice, order trichotomy, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCondensationPacket}^{\uparrow}$ read displays one positive decreasing RealSeries source row, one dyadic block schedule, one block-sum ledger, one monotone comparison row, one RegSeqRat handoff, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a condensation conclusion is consumed.

Independence witness: realseries, cauchy-series-criterion, positive-series-comparison, dyadicrat-core, streamname, regseqrat, real: $\mathsf{CauchyCondensationPacket}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a positive decreasing source series, dyadic block scheduling, block-sum comparison, monotone endpoint control, regular rational handoff, and real sealing in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationPacket}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7794.1, Definition 7794.2, and Definition 7794.3 seed the finite Cauchy condensation route over positive decreasing source series, dyadic block schedules, block-sum ledgers, monotone comparison rows, regular handoff, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host infinite-sum theorems, quotient stream equality, selected sums, order trichotomy, countable choice, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, block schedule admission, block-sum ledger exactness, monotone comparison stability, regular handoff nonescape, real-seal nonescape, transport stability, replay stability, and local naming for BEDC.Derived.CauchyCondensationPacketUp.

Constructive story: A $\mathsf{CauchyCondensationPacket}^{\uparrow}$ packet is generated from a positive decreasing source series, a dyadic block schedule, a block-sum ledger, a monotone comparison row, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7795

A Concrete Naming Certificate for $\mathsf{NagumoViability}^{\uparrow}$

Following Nagumo's invariance theorem and the constructive ODE tradition, the $\mathsf{NagumoViability}^{\uparrow}$ packet records the finite boundary-viability surface by which a displayed vector field is read as staying inside a named closed constraint set. It consumes the ODE source of chapter 193, the local interval and vector-field rows of chapter 3530, the initial-data packet of chapter 3979, the continuous-map handoff of chapter 130, and the compact interval fixed-point boundary route of chapter 1437. The packet names a finite viability-boundary certificate; it does not select a maximal flow, assert uniqueness, impose host compactness, choose a solution representative, or provide a standard bridge.

Definition 7795.1 (Nagumo viability carrier)**.** *A* $\mathsf{NagumoViability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (O, I, F, K, B, T, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *source row carrying the time-state interface and vector-field request.*
2. *I is the* $\mathsf{InitialValueProblem}^{\uparrow}$ *row naming the initial time, initial state, local window, and solution-read surface.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field graph and modulus row used for all boundary-facing reads.*
4. *K is the closed constraint row, read through the existing closed-set and Real boundary surfaces and exposing only the displayed membership and complement-neighborhood tests.*
5. *B is the Nagumo boundary row, recording the finite tangent or inward-pointing comparison data inspected at the displayed boundary cells of K.*
6. *T is the interval trap row inherited from the Peano local-existence route, keeping Euler polygon and local solution reads inside the certified Real time-state window.*
7. *E is the* $\mathsf{PeanoExistence}^{\uparrow}$ *existence row that exports a local solution read only after O, I, F, K, B, T have been displayed.*
8. *Q is the compact interval fixed-point comparison row used when a finite interval boundary or residual trap is needed to certify the viability step.*
9. *H stores componentwise* hsame *transports for the ODE, initial-data, continuous-map, closed-set, boundary, trap, existence, fixed-point, and local certificate rows.*

10. *C stores displayed* Cont *replay from the ODE request through the finite viability-boundary route.*
11. *P is the* Pkg *provenance over* NagumoViability$^{\uparrow}$, Ode$^{\uparrow}$, InitialValueProblem$^{\uparrow}$, PeanoExistence$^{\uparrow}$, ContinuousMap$^{\uparrow}$, ClosedSet$^{\uparrow}$, CompactIntervalFixedPoint$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{NagumoViability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for uniqueness, maximal-flow selection, host compactness, a choice-selected solution, quotient trajectory equality, or a standard bridge.

Theorem 7795.2 (Nagumo viability NameCert obligations)**.** *For every accepted* NagumoViability$^{\uparrow}$ *carrier*

$$V = (O, I, F, K, B, T, E, Q, H, C, P, N),$$

the local NameCert$_{\text{NagumoViability}^{\uparrow}}$ *obligations are exactly carrier admission for the ODE source, initial-data row, continuous vector-field row, closed constraint row, finite boundary comparison row, Peano interval trap, local existence row, compact interval fixed-point comparison row, componentwise transport, displayed replay, package provenance, and local naming. Any viability read must therefore pass through $O, I, F, K, B, T, E, Q, H, C, P, N$ before it can expose an invariant local solution segment.*

Proof. Project the accepted carrier of Definition 7795.1. The rows O, I, F fix the displayed ODE request, initial data, and continuous vector-field read before any boundary comparison is inspected.

The constraint-facing rows then supply the finite viability data. The closed-set row K exposes the displayed boundary surface, B records the Nagumo comparison at that boundary, and T keeps the Peano polygon and solution reads inside the certified Real time-state window.

The consumer-facing rows E and Q provide only the local existence and compact interval comparison data already routed through the previous rows. Finally H, C, P, N transport, replay, package, and name the same finite route. Since no carrier coordinate stores uniqueness, maximal-flow selection, host compactness, a choice-selected solution, quotient trajectory equality, or a bridge schema, none of those objects belongs to the local NameCert obligations. □

Closure status for NameCert$_{\text{NagumoViability}^{\uparrow}}$**.**

Constructive story: A NagumoViability$^{\uparrow}$ packet is generated from an ODE source row, an initial-value row, a continuous vector-field row, a closed constraint row, a finite Nagumo boundary comparison row, a Peano interval trap, a local existence row, a compact interval fixed-point comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NagumoViability}^{\uparrow}}$, seed)

Scope closed: Definition 7795.1 and Definition 7795.2 seed the finite viability-boundary packet over ODE, Peano existence, initial-value, continuous-map, closed-set, compact-interval, Real, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{NagumoViability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim uniqueness, maximal flow selection, host compactness, choice-selected solutions, quotient trajectory equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, boundary comparison exposure, interval-trap exactness, Peano existence handoff, compact interval comparison nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NagumoViabilityUp.

Chapter 7796

A Concrete Naming Certificate for $\mathsf{DecimalExpansionUniqueness}^{\uparrow}$

Following Bishop and Bridges together with the standard convention that decimal streams are read in normalized form, the $\mathsf{DecimalExpansionUniqueness}^{\uparrow}$ packet records the finite route by which a decimal real has at most one normalized decimal stream presentation. It sits after the decimal readback carrier of chapter 2058, the dual-tail boundary of chapter 6932, the endpoint-normalization route of chapter 1429, and the real decimal normal-form route of chapter 3952. The packet names uniqueness for normalized tails only; it does not quotient all decimal streams, choose representatives at endpoints, or expose host equality of real numbers.

Definition 7796.1 (Decimal expansion uniqueness carrier)**.** *A* $\mathsf{DecimalExpansionUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, E, N, W, Q, R, Y, H, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalReal}^{\uparrow}$ *readback row whose signed decimal prefix, ambiguity boundary, finite window, dyadic tolerance, regular rational readback, and terminal real seal are being inspected.*
2. *E is the* $\mathsf{DecimalEndpointAmbiguity}^{\uparrow}$ *boundary row, exposing whether a displayed tail is ordinary, trailing-nine-facing, or trailing-zero-facing before any uniqueness consumer proceeds.*
3. *N is the* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *row, replacing an endpoint-facing tail by the displayed normalized tail convention without choosing a quotient representative for arbitrary streams.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the decimal cells and precision requests actually read by the comparison.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite decimal windows and tolerance requests.*
6. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare normalized decimal tails at the requested precision.*
7. *Y is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, E, N, W, Q, R have been displayed.*
8. *H records componentwise* hsame *transport for decimal readback, endpoint ambiguity, endpoint normalization, stream windows, regular readback, dyadic tolerance, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the normalized comparison route*

$$D, E, N, W, Q, R \longmapsto Y.$$

10. *P is the* Pkg *provenance over* DecimalExpansionUniqueness$^{\uparrow}$, DecimalReal$^{\uparrow}$, DecimalEndpointAmbiguity$^{\uparrow}$, DecimalEndpointNormalization$^{\uparrow}$, RealDecimalNormalForm$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* NameCert$_{\text{DecimalExpansionUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, E, N, W, Q, R, Y, H, C, P, L$. It has no coordinate for quotient stream equality, selected endpoint representatives, arbitrary decimal-stream uniqueness, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7796.2 (Decimal expansion uniqueness NameCert obligations)**.** *For every accepted* DecimalExpansionUniqueness$^{\uparrow}$ *carrier*

$$U = (D, E, N, W, Q, R, Y, H, C, P, L),$$

the local NameCert$_{\text{DecimalExpansionUniqueness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for decimal readback, endpoint ambiguity, endpoint normalization, finite stream windows, regular rational readback, dyadic tolerance, real seal, transport, replay, provenance, and local naming;*

2. *endpoint visibility, requiring E to be read before a trailing-nine-facing or trailing-zero-facing tail is normalized;*

3. *normalization exactness, requiring N to display the normalized tail convention consumed by the uniqueness comparison;*

4. *finite-window comparison, requiring W, Q, R to expose the inspected decimal cells, regular rational readback, and dyadic tolerance ledger before the* Real$^{\uparrow}$ *seal Y is available;*

5. *nonescape, requiring H, C, P, L to transport, replay, source, and name only the displayed normalized-tail route and to export no quotient stream equality, selected endpoint representative, arbitrary decimal-stream uniqueness, trichotomy, countable choice, or ambient completeness row.*

No DecimalExpansionUniqueness obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7796.1. The accepted packet displays the decimal readback, endpoint ambiguity, endpoint normalization, finite windows, regular readback, dyadic tolerance ledger, real seal, and structural rows.

Endpoint visibility is the row E. It is inspected before normalization, so a dual-tail boundary is not silently treated as an equality of unnormalized streams.

Normalization exactness is the row N. It records the standard normalized-tail convention as packet data, while refusing a selected representative for arbitrary decimal presentations.

The comparison then passes through W, Q, R. These rows display the finite decimal cells, the regular rational readback, and the dyadic tolerance ledger before the real seal Y can be consumed.

Finally H, C, P, L transport, replay, source, and name the same route. Since the carrier contains no quotient stream equality, endpoint choice, arbitrary uniqueness, trichotomy, choice, or ambient completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 7796.3 (Decimal expansion uniqueness normal-form route)**.** *Every uniqueness read exported by an accepted* $\mathsf{DecimalExpansionUniqueness}^{\uparrow}$ *carrier factors through*

$\mathsf{DecimalReal}^{\uparrow}$, $\mathsf{DecimalEndpointAmbiguity}^{\uparrow}$, $\mathsf{DecimalEndpointNormalization}^{\uparrow}$, $\mathsf{RealDecimalNormalForm}^{\uparrow}$, StreamNam

as the displayed route $D, E, N, W, Q, R, Y, H, C, P, L$*. Thus the comparison concerns normalized decimal tails reaching the same terminal real seal, not quotient equality of all decimal streams and not a choice of endpoint representative.*

Proof. Start with a uniqueness consumer of the carrier. Projection from Definition 7796.1 gives the route $D, E, N, W, Q, R, Y, H, C, P, L$ and no additional stream-equality object.

Read D first. This supplies the decimal real readback route through finite prefixes, ambiguity boundary data, dyadic tolerance, regular rational readback, and a terminal real seal.

Read E and N next. The endpoint ambiguity row exposes the dual-tail boundary, and the endpoint normalization row records the normalized-tail convention used by the comparison.

The finite comparison then runs through W, Q, R. The stream windows and regular rational readback are inspected at a displayed dyadic tolerance before the real seal Y is reached.

The structural rows H, C, P, L keep the consumer on the same route. Therefore the packet exports uniqueness only for normalized decimal tails and cannot export quotient equality of all streams or selected endpoint representatives. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalExpansionUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalExpansionUniqueness}^{\uparrow}$ packet is generated from DecimalReal readback, endpoint ambiguity, endpoint normalization, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalExpansionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7796.1, Definition 7796.2, and Definition 7796.3 seed the normalized decimal-tail uniqueness route through decimal readback, endpoint ambiguity, endpoint normalization, normal-form reading, finite windows, regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalExpansionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected endpoint representatives, arbitrary decimal-stream uniqueness, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, endpoint visibility, normalization exactness, finite-window comparison, nonescape, and local naming for BEDC.Derived.DecimalExpansionUniquenessUp.

Chapter 7797

A Concrete Naming Certificate for $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$

Following the standard MetaCIC proof-engineering route for substitution, residuals, and confluence boundaries, the $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$ packet records the finite socket where a residual source and a substitution window must agree before normalization or confluence audit data may be consumed. It sits beside the residual-substitution compatibility packet of chapter 4370, the residual diamond budget of chapter 7407, and the closure-substitution frontier of chapter 7682. The packet is a socket for audit surfaces; it does not claim full subject reduction, full confluence, normalization, conversion typing, or kernel consistency.

Definition 7797.1 (MetaCIC residual-substitution socket carrier)**.** *A* $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, B, J, F, V, H, C, P, N)$$

with the following displayed rows:

1. *R is the residual source row, naming the redex or residual family whose descendants are being audited.*
2. *W is the substitution-window row, listing the binder depth, occurrence window, and substitution request visible before residual transport.*
3. *B is the candidate-boundary row, exposing the candidate branch or normalization boundary that will consume the socket.*
4. *J is the confluence socket row, recording the finite join or conditional-diamond handoff available after R, W, B have been displayed.*
5. *F is the refusal row, listing the subject-reduction, full confluence, normalization, conversion-typing, quotient-syntax, and consistency claims not exported by this socket.*
6. *V is the provenance row for the MetaCIC audit-map route, retaining which normalization and confluence audit surfaces supplied the residual and substitution data.*
7. *H records componentwise* hsame *transport for residual source, substitution window, candidate boundary, confluence socket, refusal, provenance, replay, package, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$R, W, B \longmapsto J \longmapsto F, V.$$

9. *P is the* Pkg *provenance over* $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$, $\mathsf{MetaCICResidualDiamondBudget}^{\uparrow}$, $\mathsf{MetaCICClosureSubstitutionFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundar}$ BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, B, J, F, V, H, C, P, N$. It has no coordinate for full subject reduction, full confluence, normalization, conversion typing, quotient syntax equality, hidden substitution admissibility, or kernel consistency.

Theorem 7797.2 (MetaCIC residual-substitution socket NameCert obligations)**.** *For every accepted* $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$ *carrier*

$$S = (R, W, B, J, F, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}}$ *surface consists exactly of carrier admission for residual source, substitution-window compatibility, candidate-boundary exposure, confluence socket handoff, refusal and provenance rows, componentwise transport, displayed replay, package provenance, and local naming. Any normalization or confluence audit consumer must read*

$$R, W, B \longmapsto J$$

before it may use the socket-facing row.

Proof. Project the carrier of Definition 7797.1. The residual source, substitution window, candidate boundary, confluence socket, refusal, provenance, and structural rows are exactly the displayed coordinates.

Substitution-window compatibility is the pair R, W. The source residual data and the binder-window request are read together before a candidate branch is exposed.

The candidate-boundary row B fixes the branch that may consume the socket. The confluence socket row J can therefore be read only after R, W, B are visible.

The refusal and provenance rows F, V keep the audit boundary explicit. They record which large MetaCIC claims are absent and which normalization or confluence audit surfaces supplied the data.

Finally H, C, P, N transport, replay, source, and name only those rows. Since the carrier has no coordinate for full subject reduction, confluence, normalization, conversion typing, quotient syntax equality, hidden substitution admissibility, or consistency, the listed rows exhaust the local obligation surface. □

Theorem 7797.3 (MetaCIC residual-substitution socket audit route)**.** *Every audit consumer of an accepted* $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$ *packet must expose the residual source row R, substitution-window row W, candidate-boundary row B, confluence socket row J, refusal row F, provenance row V, and structural rows H, C, P, N. The packet connects normalization and confluence audit surfaces through finite socket data without exporting full subject reduction, full confluence, normalization, conversion typing, quotient syntax equality, hidden substitution admissibility, or kernel consistency.*

Proof. Begin with an audit consumer of the socket. Projection from Definition 7797.1 gives the rows $R, W, B, J, F, V, H, C, P, N$ and no global preservation theorem.

The consumer first reads the residual source and substitution window. This fixes the redex family, binder depth, occurrence window, and substitution request as finite packet data.

Next it reads the candidate boundary and confluence socket. These rows identify the finite branch and conditional join handoff available to the audit route.

The refusal and provenance rows are read before the socket is exported. They retain the absence of subject reduction, full confluence, normalization, conversion typing, quotient syntax equality, hidden substitution admissibility, and consistency, while naming the audit surfaces that supplied the socket data.

The structural rows preserve only that displayed route. Thus the packet connects the normalization and confluence audit surfaces without turning the socket into any of the excluded MetaCIC theorems. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}$ packet is generated from residual source rows, substitution-window rows, candidate-boundary rows, confluence socket rows, refusal and provenance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7797.1, Definition 7797.2, and Definition 7797.3 seed the finite residual-substitution socket connecting normalization and confluence audit surfaces through residual source, substitution-window, candidate-boundary, confluence socket, refusal, provenance, transport, replay, package, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicResidualSubstitutionSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, full confluence, normalization, conversion typing, quotient syntax equality, hidden substitution admissibility, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, residual source exposure, substitution-window compatibility, candidate-boundary exposure, confluence socket handoff, refusal/provenance stability, and local naming for BEDC.Derived.MetacicResidualSubstitutionSocketUp.

Chapter 7798

A Concrete Naming Certificate for $\mathsf{CotransitiveApartness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CotransitiveApartness}^{\uparrow}$ packet records the constructive apartness layer in which a displayed apartness witness can be relayed through a finite cotransitivity ledger before it is consumed by real or completion routes. It refines the metric positive-distance packet of chapter 3505, the order-facing positive real bridge of chapter 1905, and the located metric-space route of chapter 3685. The packet supplies cotransitivity as finite NameCert data; it does not import real trichotomy, quotient apartness, host inequality, or ambient completeness.

Definition 7798.1 (CotransitiveApartness carrier)**.** *A* $\mathsf{CotransitiveApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, O, L, M, R, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{MetricApartness}^{\uparrow}$ *source row, exposing the positive-distance apartness read already certified by the metric apartness sibling.*
2. *O is the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *order-facing handoff row, used only to read the finite positive-radius comparison cell exported by the real apartness bridge.*
3. *L is the* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *located-distance row that supplies the dyadic tolerance windows through which intermediate point requests are inspected.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *metric-source row shared by the endpoint and intermediate-point observations.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *real-seal row reached by the displayed positive-radius and located-distance reads.*
6. *W is the finite cotransitivity witness row. For an accepted apartness read between displayed endpoints and a displayed intermediate point request, it stores the finite branch data saying that the request is routed to the left endpoint or to the right endpoint through the rows above.*
7. *H records componentwise* hsame *transport for the apartness source, real-order bridge, located metric-space row, metric source, real seal, cotransitivity witness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an apartness request through $A, O, L, M, R, W, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CotransitiveApartness}^{\uparrow}$, $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, $\mathsf{LocatedMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CotransitiveApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient apartness, host inequality, selected representatives, or completeness.

Definition 7798.2 (CotransitiveApartness classifier)**.** *The* $\mathsf{CotransitiveApartness}^{\uparrow}$ *classifier compares accepted carriers only through the rows*

$$A, O, L, M, R, W, H, C, P, N.$$

Two displayed packets are classifier-compatible when their metric apartness source rows, real-order bridge rows, located metric-space rows, metric-source rows, real-seal rows, cotransitivity witness rows, componentwise hsame *transports, displayed* Cont *replays,* Pkg *provenance rows, and local* $\mathsf{NameCert}$ *rows agree through the displayed BEDC transport surface. No comparison may inspect a host inequality predicate, a quotient apartness class, a total order branch on* $\mathsf{Real}^{\uparrow}$*, or a completion principle.*

Theorem 7798.3 (CotransitiveApartness NameCert obligations)**.** *For every accepted* $\mathsf{CotransitiveApartness}^{\uparrow}$ *carrier*

$$K = (A, O, L, M, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CotransitiveApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{MetricApartness}^{\uparrow}$ *row A, order-facing positive-radius handoff through O, located-distance and tolerance-window admission through L, metric-source visibility through M, real-seal readback through R, finite cotransitivity witness admission through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7798.1. The rows A, O, L, M, R expose the positive-distance, order-facing, located-distance, metric-source, and real-seal data before any cotransitivity read is available.

The row W is therefore the only cotransitivity witness row. It records the finite branch through which an intermediate point request is routed to the left endpoint or to the right endpoint, using the displayed rows already present in the packet.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the carrier contains no further coordinate, these rows exhaust the local NameCert obligations and exclude real trichotomy, quotient apartness, host inequality, selected representatives, and completeness. □

Theorem 7798.4 (Cotransitivity ledger)**.** *Every cotransitivity read exported by an accepted* $\mathsf{CotransitiveApartness}^{\uparrow}$ *packet factors through the finite route*

$$A, O, L, M, R \longmapsto W \longmapsto H, C, P, N.$$

The exported ledger says only that the displayed intermediate-point request is assigned to one of the two endpoint apartness branches through the metric apartness, real-order bridge, located metric-space, metric, and real rows already visible in the carrier.

Proof. Begin with an accepted cotransitivity request. The metric apartness row A supplies the original positive-distance apartness read, while O and R expose the positive-radius real handoff available to order-facing consumers.

The intermediate request is inspected through the located metric-space row L and the shared metric source M. These rows provide finite tolerance windows and metric observations before the branch row can be read.

The cotransitivity witness W then records the displayed branch assignment. Transport and replay through H and C keep the assignment attached to the same rows, and P, N provide provenance and local naming.

Thus the ledger exports only a finite branch route. It cannot create trichotomy of $\mathsf{Real}^{\uparrow}$, quotient apartness, host inequality, selected representatives, or ambient completeness because none of those objects occurs in the carrier. □

Theorem 7798.5 (CotransitiveApartness Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CotransitiveApartness}^{\uparrow}$ *packet may read only the finite positive-distance, located-window, and cotransitivity ledger rows displayed in*

$$K = (A, O, L, M, R, W, H, C, P, N).$$

The handoff exposes the metric apartness source, real-order positive-radius bridge, located metric-space tolerance route, metric source, real seal, finite cotransitivity branch ledger, componentwise transport, replay, provenance, and local naming. It does not expose $\mathsf{Real}^{\uparrow}$ *trichotomy, quotient apartness, host inequality, selected limits, or completeness.*

Proof. Start from a real or completion-facing apartness request. The request must first read the metric apartness source A and the real-order bridge O, because these are the only rows that expose positive-distance and positive-radius data.

Located tolerance information is then read from L over the shared metric source M, and the resulting real-facing seal is read through R. Only after those rows are displayed can the cotransitivity branch ledger W be consumed.

The handoff to downstream consumers is preserved by H and replayed by C, with P and N recording source and local naming. Therefore the downstream route is exactly the finite packet surface and not a route through trichotomy, quotient apartness, host inequality, selected limits, or completeness. □

Falsifiable prediction: Every accepted $\mathsf{CotransitiveApartness}^{\uparrow}$ cotransitivity read factors through MetricApartnessUp positive-distance data, RealApartnessOrderBridgeUp positive-radius handoff, LocatedMetricSpaceUp tolerance windows, MetricUp source rows, RealUp seals, one finite cotransitivity branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metricapartness, realapartnessorderbridge, locatedmetricspace, metric, real: $\mathsf{CotransitiveApartness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a positive-distance apartness source, order-facing positive-radius bridge, located tolerance route, metric source, real seal, finite cotransitivity branch ledger, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CotransitiveApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{CotransitiveApartness}^{\uparrow}$ packet is generated from a MetricApartnessUp positive-distance row, RealApartnessOrderBridgeUp positive-radius handoff, LocatedMetricSpaceUp tolerance route, MetricUp source row, RealUp seal row, finite cotransitivity branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CotransitiveApartness}^{\uparrow}}$, seed)

Scope closed: Definition 7798.1, Definition 7798.2, Definition 7798.3, Definition 7798.4, and Definition 7798.5 seed the finite cotransitive apartness interface consumed by real and completion-facing routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CotransitiveApartness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RealUp trichotomy, quotient apartness, host inequality, selected limits, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, finite cotransitivity branch admission, RealUp handoff stability, transport and replay stability, provenance non-escape, and local naming for BEDC.Derived.CotransitiveApa

Chapter 7799

A Concrete Naming Certificate for $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$

Following Stolz and Cesaro, and in the Bishop–Bridges treatment of regular real sequences, the $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$ packet records the finite tail-compression surface by which certified weighted-difference tails and certified denominator growth are read as a regular Cauchy quotient-tail row. It sits beside the theorem-level certificate of chapter 6137, the finite quotient-tail route of chapter 1439, the regular Cauchy Stolz transform of chapter 7551, the stream naming surface of chapter 126, the regular rational sequence handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only a finite compression route for Cauchy tails over displayed rows; it does not select a host limit, quotient carrier, ambient completeness theorem, or detached real-equality witness.

Definition 7799.1 (Stolz-Cesaro Cauchy tail-compression carrier)**.** *A* $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the weighted-difference tail row, displaying the inspected numerator-tail differences and their finite weights.*
2. *G is the denominator-growth row, displaying the positive growth windows used before any quotient-tail classifier is read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the inspected weighted-difference quotients and requested Cauchy moduli.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite tail windows consumed by each tolerance request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *quotient-tail compression row reached from W, G, D, S.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *compatibility row reached only after the regular rational quotient-tail row has been displayed.*
7. *H records componentwise* hsame *transport for W, G, D, S, R, E.*
8. *C records displayed* Cont *replay from weighted-difference tails through denominator growth, dyadic estimates, stream schedules, regular rational compression, and the real-name compatibility row.*

9. *P is the* Pkg *provenance over* StolzCesaroCauchyTailCompression$^{\uparrow}$, StolzCesaroTheorem$^{\uparrow}$, StolzCesaro$^{\uparrow}$, RegularCauchyStolzTransform$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{StolzCesaroCauchyTailCompression}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected host limit, quotient stream, quotient carrier, countable-choice schedule, trichotomy, or ambient Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.StolzCesaroCauchyTailCompressionUp.StolzCesaroCauchyTailCompressionTasteGate`
`single carrier alignment`

Theorem 7799.2 (Stolz-Cesaro Cauchy-tail-compression obligations)**.** *For every accepted* StolzCesaroCauchyTailCo *carrier $K = (W, G, D, S, R, E, H, C, P, N)$, a certified weighted-difference tail row together with a certified denominator-growth row induces a regular Cauchy quotient-tail compression row without selecting a host limit or quotient carrier. The local* NameCert$_{\text{StolzCesaroCauchyTailCompression}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier predicate: the accepted object is precisely the displayed packet $K = (W, G, D, S, R, E, H, C, P, N)$;*
2. *classifier: quotient-tail admission is classified only through the dyadic ledger D after the weighted-difference tail row W and denominator-growth row G have both been displayed;*
3. *tail stability: the stream schedule S preserves the admitted tail windows under requested Cauchy tolerances before R is read;*
4. *ledger compression: the regular rational row R is obtained by compressing the finite ledger W, G, D, S and has no independent quotient-carrier coordinate;*
5. *real-name compatibility: the row E reads the* Real$^{\uparrow}$ *face only through R and is preserved by H, C, P, N.*

Proof. Project the carrier $K = (W, G, D, S, R, E, H, C, P, N)$. Carrier admission gives exactly the displayed finite packet, so a consumer starts with the weighted-difference tail row W, denominator-growth row G, dyadic tolerance ledger D, stream schedule S, regular rational row R, real-name compatibility row E, and the structural rows H, C, P, N.

The classifier is then forced by the order of the rows. A quotient-tail request can be admitted only after W and G have been displayed, and its estimates are stored in D against the same dyadic tolerances. Thus the classifier has no separate source from which to select a quotient carrier or host limit.

Tail stability is supplied by S. The schedule names the finite tail windows used for each tolerance request, so the route from W, G, D to R is a finite compression of displayed windows rather than an appeal to ambient real completeness.

Finally the real-facing row E is downstream from R. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the same displayed route, so the five listed obligation rows exhaust the accepted NameCert surface. □

Theorem 7799.3 (Stolz-Cesaro Cauchy tail nonescape)**.** *Every completion-facing read from an accepted* StolzCesaroCauchyTailCompression$^{\uparrow}$ *packet factors through the finite route*

$$W, G \longmapsto D \longmapsto S \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

In particular, no accepted consumer can reach a RegSeqRat$^{\uparrow}$ *quotient-tail compression row or a* Real$^{\uparrow}$ *compatibility row without first reading the certified weighted-difference tail row, certified denominator-growth row, dyadic quotient ledger, and stream tail schedule.*

Proof. Apply Definition 7799.2. Its carrier predicate identifies the only available rows, and its classifier and tail-stability clauses place W, G, D, S before any regular rational readback.

The ledger-compression clause then identifies R as the regular Cauchy quotient-tail compression row induced from the displayed finite ledger. Since R has no independent quotient-carrier coordinate, the route cannot branch to a host quotient stream.

The real-name compatibility clause places E after R and preserves the same packet through H, C, P, N. Hence the displayed route is mandatory for every accepted completion-facing read, and the excluded host limit, quotient carrier, countable choice, trichotomy, and ambient completeness objects remain outside the packet. □

Falsifiable prediction: An accepted $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ quotient-tail row or $\mathsf{Real}^{\uparrow}$ compatibility row without first reading weighted-difference tails, denominator growth, dyadic quotient estimates, and a stream tail schedule refutes the carrier surface.

Independence witness: stolzcesarotheorem, stolzcesaro, regularcauchystolztransform, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the full ten-row packet $K = (W, G, D, S, R, E, H, C, P, N)$, because each sibling contributes only a theorem route, quotient-tail route, regular Cauchy transform, stream schedule, regular rational readback, dyadic tolerance ledger, or real-name seal while this chapter binds the compression route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}}$.

Constructive story: A $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$ packet is generated from certified weighted-difference tails, certified denominator-growth rows, dyadic quotient ledgers, stream tail schedules, a RegSeqRat quotient-tail compression row, RealUp compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7799.1, Definition 7799.2, and Definition 7799.3 seed the finite Cauchy-tail compression route over weighted differences, denominator growth, dyadic estimates, stream schedules, RegSeqRat readback, RealUp compatibility, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Stolz-Cesaro theorem, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient carrier construction, selected limits, countable choice, trichotomy, Lean verification, or bridge checking for $\mathsf{StolzCesaroCauchyTailCompression}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, quotient-tail classifier admission, tail stability, ledger compression, real-name compatibility, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.StolzCesaroCauchyT

Chapter 7800

A Concrete Naming Certificate for $\mathsf{DedekindLocatedInterval}^{\uparrow}$

Following Dedekind's interval separation tradition and the Bishop–Bridges located-real discipline, the $\mathsf{DedekindLocatedInterval}^{\uparrow}$ packet records a finite located interval separator before any completion-facing consumer reads a Real seal. It sits beside the located interval route of chapter 394, the rational interval carrier of chapter 354, the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal Real boundary of chapter 38, and the located-space request route of chapter 2043. The packet names a displayed separator row over finite interval data; it is not a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, a quotient real equality principle, a trichotomy decision, countable choice, or ambient Real completeness.

Definition 7800.1 (Dedekind located interval carrier)**.** *A* $\mathsf{DedekindLocatedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, Q, D, S, R, E, X, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, displaying the lower and upper interval endpoints together with the finite locatedness request.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *rational bracket row through which endpoint comparisons are read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the requested separator scale.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row naming the finite windows inspected by the separator.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to those windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, Q, D, S, R have been displayed.*
7. *X is the* $\mathsf{LocatedSpace}^{\uparrow}$ *request row recording the ambient located-distance query consumed by interval-facing callers.*
8. *H records componentwise* hsame *transport for L, Q, D, S, R, E, X and the structural rows.*
9. *C records displayed* Cont *replay through the route*

$$L, Q, D, S, R, X \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{DedekindLocatedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{DedekindLocatedInterval}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, Q, D, S, R, E, X, H, C, P, N$. It has no coordinate for Dedekind-cut Real construction, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.DedekindLocatedIntervalUp.DedekindLocatedIntervalTasteGate single carrier alignment`

Theorem 7800.2 (Dedekind located interval NameCert obligations)**.** *For every accepted* $\mathsf{DedekindLocatedInterval}^{\uparrow}$ *carrier*

$$I = (L, Q, D, S, R, E, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DedekindLocatedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval exposure through L, rational bracket readback through Q, dyadic separator budgeting through D, finite window inspection through S, regular-rational readback through R, Real-seal handoff through E, located-space request routing through X, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 7800.1. The located interval row L and rational bracket row Q expose the displayed endpoints before any separator read is admitted.

The dyadic ledger D, finite windows S, and regular readback R supply the finite inspection route. They determine the separator evidence only at the displayed scale and do not introduce a cut over all rational tests.

The located-space row X records the ambient request that may consume the separator, while E is reachable only after the interval, bracket, dyadic, stream, and regular rows are visible.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the refused objects do not occur as carrier coordinates, the local certificate surface cannot export them. □

Theorem 7800.3 (Dedekind located interval separator boundary)**.** *Any completion-facing consumer that reads an accepted* $\mathsf{DedekindLocatedInterval}^{\uparrow}$ *packet must pass through*

$$\mathsf{LocatedInterval}^{\uparrow}, \mathsf{RationalInterval}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{LocatedSpace}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed finite rows. The route supplies a located interval separator for downstream Real and completion-boundary arguments, but it does not construct $\mathsf{Real}^{\uparrow}$ *by Dedekind cuts, quotient real equality, trichotomy, countable choice, or ambient Real completeness.*

Proof. Use Definition 7800.2. It identifies the visible route $L, Q, D, S, R, X, E, H, C, P, N$ and forces every consumer to inspect the finite interval, bracket, tolerance, window, regular readback, located-space, and seal rows in that order.

The separator is therefore a row inside the displayed NameCert packet. It can be transported or replayed by H and C, but those structural rows preserve only the listed coordinates.

No coordinate stores a global lower cut, upper cut, quotient equality class, trichotomy decision, countable-choice schedule, or ambient completeness theorem. The separator boundary is exactly the finite route, not a construction of the real line. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindLocatedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DedekindLocatedInterval}^{\uparrow}$ packet is generated from a LocatedInterval source row, RationalInterval bracket row, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, LocatedSpace request row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DedekindLocatedInterval}\uparrow}$, seed)

Scope closed: Definition 7800.1, Definition 7800.2, and Definition 7800.3 seed the finite located interval separator over rational brackets, dyadic budgets, stream windows, regular readback, located-space requests, and Real seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{DedekindLocatedInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut Real construction, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval exposure, rational bracket readback, dyadic separator budgeting, finite window inspection, regular-rational readback, located-space request routing, Real-seal handoff, and nonescape proofs for BEDC.Derived.DedekindLocatedIntervalUp.

Chapter 7801

A Concrete Naming Certificate for $\mathsf{PoincareDiskHorocycleReturn}^{\uparrow}$

Following Poincare's disk model and the classical horocycle tradition in hyperbolic geometry, the $\mathsf{PoincareDiskHorocycleReturn}^{\uparrow}$ packet records a finite return-window surface for a displayed horocycle read inside the disk. It realizes the observer-expansion branch of chapter 9463 after the disk metric packet of chapter 6209, the horocycle carrier of chapter 6980, the geodesic-flow route of chapter 3856, the visual-boundary surface of chapter 4181, and the Busemann cocycle row of chapter 4134. The packet names boundary-shadow, disk-metric, horocycle transport, return-window, Real-seal, and provenance rows; it is not a global recurrence theorem, a quotient boundary flow, a selected complete orbit, a physical observer law, or ambient hyperbolic completeness.

Definition 7801.1 (Poincare disk horocycle-return carrier)**.** *A* $\mathsf{PoincareDiskHorocycleReturn}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, B, Hc, G, V, U, W, E, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric row used by the local read.*
2. *B is the boundary-shadow row naming the displayed ideal endpoint or boundary window.*
3. *Hc is the* $\mathsf{HyperbolicPhaseHorocycle}^{\uparrow}$ *horocycle carrier row.*
4. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *comparison row that keeps the return tied to the displayed flow parameter.*
5. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *visual-boundary readback row.*
6. *U is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *cocycle row that records the signed horocycle displacement ledger.*
7. *W is the return-window row, listing the finite disk and boundary observations accepted for the return read.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed return scale or displacement.*
9. *T records componentwise* hsame *transport for D, B, Hc, G, V, U, W, E and the structural rows.*
10. *C records displayed* Cont *replay through the route*

$$D, B, Hc, G, V, U, W \longmapsto E.$$

11. *P is the* Pkg *provenance over* PoincareDiskHorocycleReturn$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicPhaseHorocycle$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{PoincareDiskHorocycleReturn}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, B, Hc, G, V, U, W, E, T, C, P, N$. It has no coordinate for a global recurrence theorem, quotient boundary flow, selected complete orbit, physical observer law, ambient hyperbolic completeness, Lean verification, or bridge checking.

Theorem 7801.2 (Poincare disk horocycle-return NameCert obligations)**.** *For every accepted* PoincareDiskHorocycleReturn$^{\uparrow}$ *carrier*

$$R = (D, B, Hc, G, V, U, W, E, T, C, P, N),$$

the local NameCert$_{\text{PoincareDiskHorocycleReturn}^{\uparrow}}$ *surface consists exactly of disk-metric exposure through D, boundary-shadow readback through B, horocycle transport through Hc, geodesic-flow comparison through G, visual-boundary readback through V, Busemann-cocycle displacement through U, finite return-window admission through W, Real-seal handoff through E, and structural transport, replay, provenance, and local naming through T, C, P, N.*

Proof. Project the carrier of Definition 7801.1. The disk metric row D and boundary-shadow row B establish the displayed disk and boundary surface before the horocycle return is read.

The horocycle row Hc, geodesic-flow row G, visual-boundary row V, and Busemann-cocycle row U supply the hyperbolic transport and displacement data. The return window W is admitted only over those displayed rows.

The Real-seal row E is downstream of the finite return window. Transport, replay, provenance, and local naming through T, C, P, N preserve the same packet and add no orbit selector or quotient boundary flow. □

Theorem 7801.3 (Poincare disk horocycle-return vision realization)**.** *The hyperbolic phase geometry route of chapter 9463 is realized, at the horocycle-return site, by an accepted* PoincareDiskHorocycleReturn$^{\uparrow}$ *carrier whose finite route is*

PoincareDiskMetric$^{\uparrow}$, HyperbolicPhaseHorocycle$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBu

The realization is a concrete NameCert packet over displayed disk, boundary, transport, return-window, Real-seal, and provenance rows, not a global recurrence theorem or a completed boundary dynamics object.

Proof. Read the vision through the concrete disk and boundary rows named by D, B, V. These rows keep the Poincare disk metric and visual boundary visible before any return claim is inspected.

The horocycle and flow data are the rows Hc, G, U, W. They record the finite horocycle carrier, geodesic-flow comparison, Busemann displacement, and accepted return window.

Finally consume E, T, C, P, N. The Real-seal row reports only the displayed return scale or displacement, while transport, replay, provenance, and local naming keep the realization inside the same finite NameCert packet. □

Closure status for NameCert$_{\text{PoincareDiskHorocycleReturn}^{\uparrow}}$.

Constructive story: A PoincareDiskHorocycleReturn$^{\uparrow}$ packet is generated from PoincareDiskMetric disk rows, boundary-shadow rows, HyperbolicPhaseHorocycle transport, HyperbolicGeodesicFlow comparison, HyperbolicVisualBoundary readback, HyperbolicBusemannCocycle displacement, a finite return window, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PoincareDiskHorocycleReturn}\uparrow}$, seed)

Scope closed: Definition 7801.1, Definition 7801.2, and Definition 7801.3 seed the finite Poincare disk horocycle-return route over disk metric, boundary shadow, horocycle transport, geodesic flow, visual boundary, Busemann cocycle, return-window, Real-seal, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PoincareDiskHorocycleReturn}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global recurrence, quotient boundary flow, selected complete orbit, physical observer dynamics, ambient hyperbolic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-metric exposure, boundary-shadow readback, horocycle transport, geodesic-flow comparison, visual-boundary readback, Busemann-cocycle displacement, finite return-window admission, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDiskHorocycleReturnUp.

Chapter 7802

A Concrete Naming Certificate for $\mathsf{RegularCauchyDominatedTail}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDominatedTail}^{\uparrow}$ packet records the finite L10 route by which a regular Cauchy source row, a displayed domination budget, and a tail row produce a regular rational readback before any real seal is consumed. It sits over the dyadic tolerance arithmetic of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the real-name boundary of chapter 38, and the comparison discipline of chapter 1459. The packet names a dominated-tail certificate only as finite NameCert data; it is not a selected limit, quotient equality of Cauchy names, countable choice, trichotomy, host real comparison, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7802.1 (Regular Cauchy dominated-tail carrier)**.** *A* $\mathsf{RegularCauchyDominatedTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular-Cauchy row whose inspected entries are read only through finite windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dominance budget, recording the dyadic tolerance and domination margin used to compare the source tail with its displayed majorant.*
3. *T is the dominated tail row, containing the tail index, the finite tail window, and the pointwise domination reads admitted for that window.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window surface that lists every source and tail coordinate inspected before a regular readback is exported.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source windows, tail windows, and dyadic budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, D, T, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source row, dominance budget, dominated tail row, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, D, T, W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDominatedTail}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDominatedTail}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, W, R, E, H, C, P, N$. It has no coordinate for a hidden limit selector, quotient equality of regular Cauchy names, countable choice, trichotomy on arbitrary rational tails, host real comparison, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 7802.2 (Regular Cauchy dominated-tail obligation surface)**.** *For every accepted* RegularCauchyDominate *carrier*

$$K = (S, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDominatedTail}^{\uparrow}}$ *surface consists exactly of carrier admission for the source regular-Cauchy row S, dyadic dominance budget D, dominated tail row T, stream-window surface W, regular readback R, real seal E, componentwise transport H, displayed replay C, provenance P, and local naming N. The comparison-test use is finite: every domination read must be routed through the displayed* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *discipline on S, D, T, W before R or E can be consumed.*

Proof. Project the accepted carrier onto its ten rows. The source data are S, the dyadic tolerance and domination margin are D, and the dominated tail is T; no tail comparison is available outside these displayed rows.

The window and readback rows then fix the observation route. The row W lists the finite stream coordinates inspected for the source and tail, while R is the only regular rational readback attached to those windows and to the dyadic budget.

The real seal E is downstream from S, D, T, W, R. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve exactly that route. Since the carrier has no extra coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 7802.3 (Regular Cauchy dominated-tail L10 handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyDominatedTail}^{\uparrow}$ *carrier receives the finite L10 handoff*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

implemented by the displayed route $S, D, T, W, R, E, H, C, P, N$. The handoff certifies only that the dominated tail has been read through the dyadic budget and finite stream windows before the regular rational row and real seal are exposed; it exports no selected limit, quotient equality of Cauchy names, countable choice, host real comparison, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 7802.2. It forces the source regular-Cauchy row, dominance budget, dominated tail row, and stream windows to be displayed before any regular rational readback can be used.

The comparison-test row is consumed on that same finite surface. Its role is to check the tail domination against the dyadic budget, not to supply a host real comparison or a selected endpoint.

After the finite comparison is displayed, R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback and E supplies the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N transport, replay, source, and name the same route, so the public handoff is exactly the finite L10 certificate above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDominatedTail}^{\uparrow}}$.

Constructive story: A RegularCauchyDominatedTail$^{\uparrow}$ packet is generated from a source regular-Cauchy row, a dyadic dominance budget, a dominated tail row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyDominatedTail}^{\uparrow}}$, seed)

Scope closed: Definition 7802.1, Definition 7802.2, and Definition 7802.3 seed the finite dominated-tail route from dyadic domination and stream windows through regular rational readback to the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDominatedTail}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy names, countable choice, trichotomy, host real comparison, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A stronger status requires Lean coverage for carrier admission, dominated-tail comparison routing, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyDominatedTailUp

Chapter 7803

A Concrete Naming Certificate for $\mathsf{CauchyTailMeet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailMeet}^{\uparrow}$ packet records the finite common-tail witness used when two displayed Cauchy requests must be synchronized before a regular rational or real-name consumer reads them. It sits over the one-tail estimate route of chapter 1487, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance meet of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite common threshold and its inspected windows; it is not a selected limit, quotient stream equality, countable-choice schedule, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7803.1 (Cauchy tail meet carrier)**.** *A* $\mathsf{CauchyTailMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A_0, A_1, \tau, W_0, W_1, D, R_0, R_1, E, H, C, P, N)$$

with the following displayed rows:

1. *A_0 and A_1 are the two admitted tail requests, each naming a finite request into the sibling* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *surface.*
2. *τ is the common threshold row at which both requests are simultaneously inspected.*
3. *W_0 and W_1 are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows read at and beyond τ.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance meet, recording the common dyadic budget used by both tail windows.*
5. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows reached from W_0, D and W_1, D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after the two requests, common threshold, finite windows, tolerance meet, and regular readbacks have been replayed.*
7. *H records componentwise* hsame *transport for the two requests, common threshold, windows, tolerance meet, readbacks, seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a paired tail request through $\tau, W_0, W_1, D, R_0, R_1$ and then E.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailMeet}^{\uparrow}$*,* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailMeet}^{\uparrow}}$ *row naming exactly this finite common-tail packet.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a selected infinite tail, quotient stream equality, countable choice, host real equality, trichotomy, or ambient completeness.

Theorem 7803.2 (Cauchy tail meet NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailMeet}^{\uparrow}$ *carrier*

$$T = (A_0, A_1, \tau, W_0, W_1, D, R_0, R_1, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailMeet}^{\uparrow}}$ *surface consists exactly of carrier admission, two tail-request reads, common-threshold visibility, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance meet, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every common-tail consumer must pass through*

$$A_0, A_1, \tau, W_0, W_1, D, R_0, R_1, E, H, C, P, N$$

and through no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7803.1. The rows A_0, A_1 expose the two finite tail requests, and the row τ is the single common threshold at which both requests are inspected.

Read the analytic rows after the threshold. The windows W_0, W_1 give the finite $\mathsf{StreamName}^{\uparrow}$ observations, D is the shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance meet, and R_0, R_1 are the only regular rational readbacks downstream of those windows.

Finally read the seal and structural rows. The row E is available only after the displayed request, threshold, window, tolerance, and readback rows have been replayed, while H, C, P, N preserve, replay, source, and name the same finite route.

Since no carrier coordinate stores a selected infinite tail, quotient stream equality, countable-choice schedule, host real equality, trichotomy, or ambient completeness witness, none of those objects can appear as a local NameCert obligation. □

Theorem 7803.3 (Cauchy tail meet finite induction)**.** *Every predicate Q on accepted* $\mathsf{CauchyTailMeet}^{\uparrow}$ *packets is established by finite induction when the following displayed cases are stable under* hsame*:*

1. *the two tail requests A_0, A_1 have been admitted;*
2. *the common threshold τ has been read;*
3. *the two finite windows W_0, W_1 have been inspected;*
4. *the shared dyadic meet D and regular readbacks R_0, R_1 have been replayed;*
5. *the real seal E, continuation row C, provenance row P, and local name N have been consumed.*

Thus common-tail arguments over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *use only finite rows of T and do not appeal to a choice-selected threshold family.*

Proof. Induct over the displayed rows of the carrier. The base stage admits A_0, A_1 and exposes no threshold-dependent data.

The threshold stage adds the single row τ. Since both requests read the same threshold row, the induction context has one common finite index rather than two independently chosen tails.

The window and readback stages add W_0, W_1, D, R_0, R_1 in the displayed order. Stability under the componentwise hsame rows transports each finite observation and dyadic meet without changing the route.

The final stage reads E, C, P, N. The seal, continuation, provenance, and local name consume the accumulated finite route and introduce no further threshold selector, so the predicate Q is obtained by the stated finite induction. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailMeet}\uparrow}$.

Constructive story: A $\mathsf{CauchyTailMeet}^{\uparrow}$ packet is generated from two tail requests, one common threshold, finite StreamName windows, a DyadicRatCore tolerance meet, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailMeet}\uparrow}$, seed)

Scope closed: Definition 7803.1, Definition 7803.2, and Definition 7803.3 seed the finite common-tail witness over StreamName windows, dyadic tolerance meets, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailMeet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close regular Cauchy completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold visibility, finite window admission, dyadic meet stability, regular rational readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailMeetUp.

Chapter 7804

A Concrete Naming Certificate for $\mathsf{FenchelYoungInequality}^{\uparrow}$

Following Rockafellar and Boyd–Vandenberghe, the $\mathsf{FenchelYoungInequality}^{\uparrow}$ packet records the finite convex-analysis surface for the inequality $f(x) + f^*(y) \geq \langle y, x\rangle$. It sits between the convex-set, vector-space, real-name, conjugate, and duality rows of chapter 208, chapter 47, chapter 38, chapter 3563, and chapter 321. The packet names the displayed inequality as BEDC carrier data; it does not import host suprema, quotient function spaces, selected minimizers, choice, or ambient completeness.

Definition 7804.1 (Fenchel–Young inequality carrier)**.** *A* $\mathsf{FenchelYoungInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, V, R, F, D, E, P, L, H, C, Q, N)$$

with the following displayed rows:

1. *X is the displayed* $\mathsf{ConvexSet}^{\uparrow}$ *row carrying the primal convex-domain boundary.*
2. *V is the displayed* $\mathsf{VecSpace}^{\uparrow}$ *row carrying the primal and dual vector-space read.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row that records the ordered scalar boundary.*
4. *F is the* $\mathsf{ConvexConjugate}^{\uparrow}$ *row carrying the finite conjugate packet for f^*.*
5. *D is the* $\mathsf{FenchelDuality}^{\uparrow}$ *row that records the surrounding primal-dual ledger.*
6. *E is the pairing-evaluation row for the finite read of $\langle y, x\rangle$.*
7. *P is the comparison ledger row displaying $f(x)$, $f^*(y)$, the sum row, the pairing row, and the inequality comparison.*
8. *L is the handoff ledger that orders the convex-set, vector-space, real, conjugate, duality, pairing, and comparison rows before any optimization consumer reads the inequality.*
9. *H stores componentwise* hsame *transport for all displayed rows.*
10. *C stores displayed* Cont *replay from primal and dual reads to pairing evaluation, from conjugate read to comparison, and from comparison to the inequality handoff.*
11. *Q is the* Pkg *provenance over* $\mathsf{FenchelYoungInequality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FenchelYoungInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, V, R, F, D, E, P, L, H, C, Q, N$. It has no coordinate for a host supremum operator, quotient function space, selected optimizer, separating hyperplane principle, or ambient completeness theorem.

Theorem 7804.2 (Fenchel–Young inequality NameCert obligations)**.** *For every accepted* $\mathsf{FenchelYoungInequality}^{\uparrow}$ *carrier* $K = (X, V, R, F, D, E, P, L, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FenchelYoungInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows, classifier stability under componentwise* hsame *transport, pairing replay through the* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows, conjugate handoff through* $\mathsf{ConvexConjugate}^{\uparrow}$*, primal-dual ledger handoff through* $\mathsf{FenchelDuality}^{\uparrow}$*, comparison-ledger exactness through* P, L*, and local provenance through* Q, N*. Every inequality read must pass through the convex domain row, vector-space row, real comparison row, conjugate row, duality row, pairing row, and displayed comparison ledger before an optimization consumer can inspect it.*

Proof. Project the carrier of Definition 7804.1. The visible coordinates are exactly $X, V, R, F, D, E, P, L, H, C, Q, N$ so carrier admission and classifier comparison are componentwise on those rows.

The analytic sources are read only through sibling packets. The rows X, V, R provide the convex-domain, vector-space, and scalar-order boundaries, while F and D import only the displayed conjugate and primal-dual ledgers of chapter 3563 and chapter 321.

The pairing and comparison rows then determine the inequality surface. The replay row C routes primal and dual reads to E, routes the conjugate read to P, and routes the comparison ledger to L, Q, N. Since no host supremum, quotient function space, optimizer selector, or completeness row is present in the carrier, the listed rows form the whole local NameCert surface. □

Theorem 7804.3 (Fenchel–Young inequality conjugate handoff)**.** *Every* $\mathsf{FenchelDuality}^{\uparrow}$ *or optimization consumer that reads a Fenchel–Young inequality through an accepted* $\mathsf{FenchelYoungInequality}^{\uparrow}$ *carrier receives only the displayed route*

$$X, V, R, F, D, E, P, L \longmapsto H, C, Q, N.$$

The consumer reads convex-domain data, vector-space data, real comparison data, conjugate data, primal-dual ledger data, pairing evaluation, and the comparison ledger before receiving the local NameCert *row. The route exports no host supremum, quotient function space, selected minimizer, Hahn–Banach separation, strong-duality existence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 7804.2. It places each inequality read on X, V, R, F, D, E, P, L before the transport, replay, provenance, and local naming rows can return an answer.

Now read the conjugate dependency. The $\mathsf{ConvexConjugate}^{\uparrow}$ row F supplies conjugacy only as the finite packet of chapter 3563; the $\mathsf{FenchelDuality}^{\uparrow}$ row D supplies only the surrounding primal-dual ledger of chapter 321. Thus the inequality handoff is a finite comparison route rather than an imported theorem of host convex analysis.

The refused objects are absent from Definition 7804.1. Therefore the handoff can deliver the displayed inequality surface to downstream optimization chapters, but it cannot deliver a selected optimizer, host supremum, quotient-function equality, separation theorem, strong-duality existence witness, or ambient completeness result. □

Falsifiable prediction: Every $\mathsf{FenchelYoungInequality}^{\uparrow}$ consumer reads the inequality through convex-domain, vector-space, real-comparison, convex-conjugate, Fenchel-duality, pairing-evaluation, comparison-ledger, handoff-ledger, transport, continuation replay, provenance, and local naming rows.

Independence witness: convexset, vecspace, real, convex-conjugate, fenchelduality: $\mathsf{FenchelYoungInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the conjugate comparison, pairing evaluation, primal-dual ledger, inequality ledger, transport, replay, provenance, and local naming rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FenchelYoungInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{FenchelYoungInequality}^{\uparrow}$ packet is generated from displayed convex-domain, vector-space, real-comparison, convex-conjugate, Fenchel-duality, pairing-evaluation, comparison-ledger, handoff-ledger, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FenchelYoungInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7804.1, Definition 7804.2, and Definition 7804.3 bind the finite Fenchel–Young inequality certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FenchelYoungInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex analysis, host suprema, quotient function spaces, selected minimizers, separation theorems, strong duality, choice, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, conjugate handoff, comparison-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.FenchelYoungInequalityUp.

Chapter 7805

A Concrete Naming Certificate for $\mathsf{HamiltonJacobiBellman}^{\uparrow}$

Following Bellman, Fleming–Rishel, and Evans, the $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ packet records the finite dynamic-programming, PDE, and control interface at which a displayed value row is read through a Hamiltonian minimization surface. It sits after the finite Bellman recursion of chapter 5875, the Hamilton-Jacobi readback packet of chapter 7663, the PDE relation surface of chapter 194, the quadratic-control dependency of chapter 5103, and the finite LQR route of chapter 296. The packet names only displayed state, control, value, Hamiltonian, PDE-relation, dynamic-programming, Riccati, LQR, transport, replay, provenance, and local naming rows; it is not an existence theorem for optimal controls, not a classical viscosity-solution theorem, not a global PDE regularity theorem, and not an ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Definition 7805.1 (Hamilton-Jacobi-Bellman carrier)**.** *A* $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, U, T, C, V, G, E, D, J, Q, L, R, A, M, K, P, N)$$

with the following displayed rows:

1. *X is the displayed state row and U is the admissible control row consumed by the finite control request.*
2. *T is the transition or dynamics row, and C is the running-cost row sealed only through displayed* $\mathsf{Real}^{\uparrow}$ *data.*
3. *V is the value-row name, G is the finite slope or gradient-read row, and E is the terminal or boundary value row.*
4. *D is the* $\mathsf{DynamicProgramming}^{\uparrow}$ *recursion row that supplies finite Bellman table readback.*
5. *J is the* $\mathsf{HamiltonJacobi}^{\uparrow}$ *Hamiltonian readback row, and Q is the* $\mathsf{Pde}^{\uparrow}$ *relation row tying the value surface to a displayed PDE request.*
6. *L is the* $\mathsf{LQR}^{\uparrow}$ *quadratic-control dependency row, and R is the* $\mathsf{ControlRiccati}^{\uparrow}$ *Riccati comparison row available for the linear-quadratic face.*
7. *A records componentwise* hsame *transport for $X, U, T, C, V, G, E, D, J, Q, L, R$.*
8. *M records displayed* Cont *replay through the route*

$$D, J, Q, L, R \longmapsto V.$$

9. *K is the* Pkg *provenance over* $\mathsf{HamiltonJacobiBellman}^{\uparrow}$, $\mathsf{DynamicProgramming}^{\uparrow}$, $\mathsf{HamiltonJacobi}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *P, N are the refusal and local* $\mathsf{NameCert}_{\mathsf{HamiltonJacobiBellman}^{\uparrow}}$ *rows.*

The classifier compares accepted packets only through $X, U, T, C, V, G, E, D, J, Q, L, R, A, M, K, P, N$. It has no coordinate for selected optimal controls, classical solution existence, viscosity uniqueness, unbounded PDE regularity, host minimizers, quotient value-function equality, or ambient real completeness.

Theorem 7805.2 (Hamilton-Jacobi-Bellman NameCert obligations)**.** *For every accepted* $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ *carrier*

$$H = (X, U, T, C, V, G, E, D, J, Q, L, R, A, M, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HamiltonJacobiBellman}^{\uparrow}}$ *surface is the following five-row obligation surface:*

1. *carrier admission exposes the displayed state, control, transition, running-cost, value, slope, and terminal rows X, U, T, C, V, G, E;*
2. *dynamic-programming exposure routes finite Bellman recursion through D and chapter 5875;*
3. *Hamilton-Jacobi and PDE exposure routes Hamiltonian and relation reads through J, Q and chapter 7663, chapter 194;*
4. *quadratic-control exposure routes the LQR and Riccati faces through L, R and chapter 296, chapter 5103;*
5. *structural exposure routes componentwise transport, replay, provenance, refusal, and local naming through A, M, K, P, N.*

No listed row exports selected optimal controls, classical solution existence, viscosity uniqueness, unbounded PDE regularity, host minimizers, quotient value-function equality, or ambient real completeness.

Proof. Project the carrier of Definition 7805.1. The rows X, U, T, C, V, G, E give the visible finite control and value surface before any Bellman or PDE request is read.

Read the sibling rows next. The dynamic-programming row D supplies finite Bellman recursion, the Hamilton-Jacobi row J supplies Hamiltonian readback, and the PDE row Q supplies the displayed relation surface.

For the quadratic-control face, the LQR row L and Riccati row R are dependency rows inside the same packet. A consumer may read them only after the finite value, Hamiltonian, and relation rows have been displayed.

Finally use A, M, K, P, N for transport, replay, provenance, refusal, and local naming. Since the carrier has no further coordinate, none of the refused analytic or choice-theoretic objects can be exported by the NameCert surface. □

Theorem 7805.3 (Hamilton-Jacobi-Bellman sibling dependency route)**.** *Every downstream consumer of an accepted* $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ *packet that asks for a control-facing PDE read must pass through*

$$\mathsf{DynamicProgramming}^{\uparrow}, \mathsf{HamiltonJacobi}^{\uparrow}, \mathsf{Pde}^{\uparrow}, \mathsf{ControlRiccati}^{\uparrow}, \mathsf{LQR}^{\uparrow} \longmapsto \mathsf{HamiltonJacobiBellman}^{\uparrow}$$

as displayed sibling rows. The route binds finite Bellman recursion, Hamiltonian readback, PDE relation data, Riccati comparison, and LQR specialization as one NameCert packet, without turning any sibling into an optimal-control existence theorem or a classical PDE regularity theorem.

Proof. Apply Definition 7805.2. The dynamic-programming, Hamilton-Jacobi, PDE, LQR, and Riccati obligations are separate displayed rows of the same accepted carrier.

The structural rows A, M, K, P, N preserve that row inventory under transport and replay. They do not add a host minimizer, a selected policy, a viscosity uniqueness theorem, or an ambient completion principle.

Thus a consumer reaches $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ only through the finite sibling route shown above, and the refused analytic objects remain outside the packet. □

Falsifiable prediction: A claimed $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ read that reaches a PDE relation or quadratic-control face without displaying the DynamicProgramming, HamiltonJacobi, Pde, ControlRiccati, and LQR rows refutes the carrier.

Independence witness: dynamic-programming, hamilton-jacobi, pde, control-riccati, lqr: no listed sibling is bijective to $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ because only this packet binds finite Bellman recursion, Hamiltonian readback, PDE relation data, Riccati comparison, LQR specialization, value rows, transport, replay, provenance, refusal, and local naming as one HJB interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HamiltonJacobiBellman}^{\uparrow}}$.

Constructive story: A $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ packet is generated from displayed state, control, transition, cost, value, slope, terminal, DynamicProgramming, HamiltonJacobi, Pde, ControlRiccati, LQR, transport, replay, provenance, refusal, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HamiltonJacobiBellman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7805.1, Definition 7805.2, and Definition 7805.3 seed the finite HJB control-PDE interface over dynamic programming, Hamilton-Jacobi, PDE, Riccati, and LQR rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HamiltonJacobiBellman}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected optimal controls, classical solution existence, viscosity uniqueness, global PDE regularity, host minimizers, quotient value-function equality, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dynamic-programming exposure, Hamilton-Jacobi and PDE exposure, quadratic-control exposure, transport, replay, provenance, refusal, and local naming for BEDC.Derived.HamiltonJacobiBellmanUp.

Chapter 7806

A Concrete Naming Certificate for $\mathsf{PoincareHalfPlane}^{\uparrow}$

Following Poincare and the standard upper-half-plane model of hyperbolic geometry, the $\mathsf{PoincareHalfPlane}^{\uparrow}$ packet records finite half-plane coordinate, boundary, metric, and transport rows as BEDC NameCert data. It realizes a concrete model for the observer-expansion branch of chapter 9463 beside the disk metric packet of chapter 6209, the hyperbolic metric surface of chapter 3474, the visual-boundary route of chapter 4181, and the Mobius transport surface of chapter 5110. The packet names only displayed half-plane, boundary, metric, Mobius transport, disk-comparison, Real-seal, provenance, and local naming rows; it is not a global model equivalence theorem, a quotient boundary dynamics object, an analytic completeness theorem, or a physical observer law.

Definition 7806.1 (Poincare half-plane carrier)**.** *A* $\mathsf{PoincareHalfPlane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (H, B, M, D, V, T, E, C, R, Q, N)$$

with the following displayed rows:

1. *H is the upper-half-plane coordinate row, naming the displayed point window whose imaginary coordinate is positive.*
2. *B is the boundary row, containing the displayed real-axis and far-end boundary observations accepted by the packet.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row restricted to the displayed half-plane window.*
4. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *comparison row used only when a Cayley-style disk comparison is displayed.*
5. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *readback row attached to the same boundary window.*
6. *T is the* $\mathsf{MobiusTransform}^{\uparrow}$ *transport row recording the finite fractional-linear map data inspected by the consumer.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for metric, boundary, or distortion values read from the displayed rows.*
8. *C records componentwise* hsame *transport for H, B, M, D, V, T, E and the structural rows.*
9. *R records displayed* Cont *replay through the route*

$$H, B, M, V, T \longmapsto E.$$

10. *Q is the* Pkg *provenance over* $\mathsf{PoincareHalfPlane}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualBound}$ $\mathsf{MobiusTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareHalfPlane}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, B, M, D, V, T, E, C, R, Q, N$. It has no coordinate for global disk-half-plane equivalence, quotient boundary flow, selected complete geodesics, analytic hyperbolic completeness, physical observer dynamics, Lean verification, or bridge checking.

Theorem 7806.2 (Poincare half-plane NameCert obligations)**.** *For every accepted* $\mathsf{PoincareHalfPlane}^{\uparrow}$ *carrier*

$$P = (H, B, M, D, V, T, E, C, R, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareHalfPlane}^{\uparrow}}$ *surface consists exactly of half-plane coordinate exposure through H, boundary readback through B, hyperbolic metric exposure through M, optional disk comparison through D, visual-boundary readback through V, Mobius transport through T, Real-seal handoff through E, and structural transport, replay, provenance, and local naming through C, R, Q, N.*

Proof. Project the carrier of Definition 7806.1. The half-plane coordinate row H and boundary row B establish the displayed upper-half-plane window before any metric or transport read is inspected.

The metric and boundary rows are then read through M, V, E. The optional disk-comparison row D remains a comparison row inside the same packet, not a proof that every disk model object has been globally identified with a half-plane object.

Mobius transport is supplied by T and preserved by C, R, Q, N. Since the carrier has no further coordinate, the packet cannot export quotient boundary dynamics, complete geodesic selection, physical observer dynamics, analytic completeness, or bridge checking. □

Theorem 7806.3 (Poincare half-plane boundary transport)**.** *Every boundary-transport consumer of an accepted* $\mathsf{PoincareHalfPlane}^{\uparrow}$ *packet must first expose the half-plane coordinate row H, boundary row B, hyperbolic metric row M, visual-boundary row V, Mobius transport row T, Real-seal row E, and structural rows C, R, Q, N. A disk comparison may be read through D only after those half-plane rows are visible, and the read exports no global disk-half-plane equivalence, quotient boundary flow, or selected complete orbit.*

Proof. Use Definition 7806.2. It identifies the visible boundary-transport route as $H, B, M, V, T, E, C, R, Q, N$ with D as an optional comparison row.

The transport row T is finite Mobius data attached to the displayed half-plane and boundary observations. It can move the named rows under componentwise hsame transport, but it does not create a quotient boundary object or a completed orbit.

The Real-seal and provenance rows close only the displayed metric and boundary reads. Therefore the half-plane packet can be consumed by later disk, boundary, and Mobius chapters without claiming global equivalence or analytic completeness. □

Falsifiable prediction: A claimed $\mathsf{PoincareHalfPlane}^{\uparrow}$ boundary transport that omits the half-plane coordinate row, boundary row, hyperbolic metric row, visual-boundary row, Mobius transport row, Real seal, transport, replay, provenance, or local NameCert row refutes the carrier.

Independence witness: hyperbolicmetric, poincare-disk-metric, hyperbolic-visual-boundary, mobius-transform: no listed sibling is bijective to $\mathsf{PoincareHalfPlane}^{\uparrow}$ because only this packet binds upper-half-plane coordinates, boundary rows, metric reads, disk comparison, Mobius transport, Real sealing, provenance, and local naming as one half-plane model surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareHalfPlane}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareHalfPlane}^{\uparrow}$ packet is generated from upper-half-plane coordinate rows, boundary rows, HyperbolicMetric rows, optional PoincareDiskMetric comparison rows, HyperbolicVisualBoundary readback, MobiusTransform transport, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareHalfPlane}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7806.1, Definition 7806.2, and Definition 7806.3 seed the finite upper-half-plane model surface over boundary, metric, disk-comparison, Mobius transport, Real-seal, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareHalfPlane}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global disk-half-plane equivalence, quotient boundary dynamics, selected complete geodesics, physical observer laws, analytic hyperbolic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coordinate exposure, boundary readback, hyperbolic metric exposure, disk comparison, visual-boundary readback, Mobius transport, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.PoincareHalfPlaneUp.

Chapter 7807

A Concrete Naming Certificate for $\mathsf{UniformFilterCompletion}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces and filters, the $\mathsf{UniformFilterCompletion}^{\uparrow}$ packet records the L10 completion surface where a displayed Cauchy-filter base is read through a uniform-completion handoff before any completion consumer sees a real-facing seal. It sits beside the functorial completion route of chapter 1892, the Cauchy-filter base route of chapter 1915, the Cauchy-filter completion criterion of chapter 3885, the regular rational readback of chapter 301, the finite StreamName windows of chapter 126, and the terminal Real seal of chapter 38. The packet names a filter-based completion surface without quotient completions, selected limits, choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7807.1 (Uniform filter completion carrier)**.** *A* $\mathsf{UniformFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, B, Q, G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *route row, exposing the functorial completion handoff used by downstream completion consumers.*
2. *B is the* $\mathsf{CauchyFilterBase}^{\uparrow}$ *basis row, listing the finite filter-base requests and directed refinements consumed by the completion route.*
3. *Q is the* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *criterion row, requiring the filter route to pass through displayed basis, window, readback, and seal data.*
4. *G is the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row, binding the same uniform source to the functorial and criterion rows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the filter-base and completion requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, readable only after F, B, Q, G, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the functorial, filter-base, criterion, uniform-source, window, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay of the same finite completion route.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformFilterCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformFilterCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, B, Q, G, S, R, E, H, C, P, N$. It has no coordinate for quotient filter equality, quotient completion equality, selected representatives, selected limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7807.2 (Uniform filter completion NameCert obligations)**.** *For every accepted* $\mathsf{UniformFilterCompletion}^{\uparrow}$ *carrier $U = (F, B, Q, G, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the functorial uniform-completion route, Cauchy-filter base, Cauchy-filter completion criterion, uniform source, StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming rows. Every completion-facing obligation is read through those rows before a consumer may inspect the terminal seal, and no obligation is stored in quotient filters, selected limits, choice, host equality, or ambient completion data.*

Proof. Project the carrier of Definition 7807.1. The tuple displays exactly the rows $F, B, Q, G, S, R, E, H, C, P, N$.

Read the completion source first. The uniform source row G and functorial handoff row F fix the displayed uniform-completion route, while the Cauchy-filter base row B supplies the finite basis data consumed by that route.

Read the criterion and numerical rows next. The criterion row Q requires the completion read to remain attached to the displayed basis route, the StreamName row S schedules the finite observations, the RegSeqRat row R reads back those windows, and the Real seal E is exposed only after that readback.

Finally H, C, P, N transport, replay, source, and name the same packet. Since quotient completions, selected limits, choice, host equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local certificate surface. □

Theorem 7807.3 (Uniform filter completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{UniformFilterCompletion}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{UniformCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CauchyFilterBase}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletionCriterion}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow} \mapsto$$

The route exports filter-based completion data only through displayed BEDC rows. It exports no quotient completion, quotient filter, selected representative, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 7807.2. It confines every accepted read to the displayed functorial, filter-base, criterion, uniform-source, window, readback, seal, and structural rows.

The filter route passes through B before the criterion row Q can be consumed. Thus a completion-facing consumer cannot replace the basis data by an unrecorded filter or a quotient-filter equality.

The completion route passes through F, G, S, R, E in the displayed order. The functorial and uniform-source rows precede the finite windows, the regular rational readback precedes the Real seal, and the structural rows preserve that same order.

Therefore the accepted handoff is exactly the finite route in the statement. The excluded quotient, selector, choice, host-equality, and ambient-completeness objects are not coordinates of the carrier. □

Falsifiable prediction: Every accepted $\mathsf{UniformFilterCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows is outside the packet surface.

Independence witness: uniformcompletionfunctor, cauchy-filter-base, cauchy-filter-completion-criterion, regseqrat, streamname, real: no listed sibling is bijective to $\mathsf{UniformFilterCompletion}^{\uparrow}$ because this packet jointly carries the functorial uniform-completion handoff, finite Cauchy-filter base, completion criterion, uniform source, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming as one filter-based completion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformFilterCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformFilterCompletion}^{\uparrow}$ packet is generated from a UniformCompletionFunctorUp route row, CauchyFilterBaseUp basis row, CauchyFilterCompletionCriterionUp criterion row, UniformCompletionUp source row, StreamNameUp windows, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7807.1, Definition 7807.2, and Definition 7807.3 seed the L10 filter-based uniform completion route over UniformCompletionFunctorUp, CauchyFilterBaseUp, CauchyFilterCompletionCriterionUp, UniformCompletionUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformFilterCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completions, quotient filters, selected representatives, selected limits, countable choice, host equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, functorial uniform-completion handoff, Cauchy-filter base exposure, Cauchy-filter completion criterion routing, finite-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformFilterCompletionUp.

Chapter 7808

A Concrete Naming Certificate for $\mathsf{DushnikMillerDimension}^{\uparrow}$

Following Dushnik and Miller on the dimension of partially ordered sets, the $\mathsf{DushnikMillerDimension}^{\uparrow}$ packet records the finite order-theoretic certificate by which a displayed poset is represented as the intersection of a finite family of linear extensions. It sits after the separated order surface of chapter 53, the inherited preorder route of chapter 52, the total-order comparison surface of chapter 54, and the membership discipline of chapter 3668. The packet names only finite extension rows, an intersection ledger, a minimal-cardinality witness, transport, replay, provenance, and local naming; it does not import a host powerset of orders, quotient posets, choice-selected extension families, or ambient set-theoretic dimension.

Definition 7808.1 (Dushnik–Miller dimension carrier)**.** *A* $\mathsf{DushnikMillerDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (P, L, I, M, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poset}^{\uparrow}$ *source row, including the inherited* $\mathsf{Preorder}^{\uparrow}$ *comparison ledger and antisymmetry data for the displayed carrier.*
2. *L is a finite family of* $\mathsf{TotalOrder}^{\uparrow}$ *rows on the same displayed carrier, each row carrying a linear-extension ledger for the order named by P.*
3. *I is the intersection ledger: for any displayed pair of carried elements, the* $\mathsf{Poset}^{\uparrow}$ *comparison holds exactly when every row of L compares the pair in that direction.*
4. *M is the minimal-cardinality witness, recording a displayed natural-number bound for the family L and a local refusal ledger for smaller displayed families.*
5. *H records componentwise* hsame *transport for the poset source, total-order family, intersection ledger, minimality witness, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a dimension request through the poset source, the finite linear-extension family, the intersection ledger, and the minimality witness.*
7. *Q is the* Pkg *provenance over* $\mathsf{DushnikMillerDimension}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DushnikMillerDimension}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host powerset of all linear orders, quotient poset equality, a choice-selected extension family, or an ambient cardinal-invariant theorem.

Theorem 7808.2 (Dushnik–Miller dimension NameCert obligations)**.** *For every accepted* $\mathsf{DushnikMillerDimension}^{\uparrow}$ *carrier* $D = (P, L, I, M, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DushnikMillerDimension}^{\uparrow}}$ *surface consists exactly of source admission for* P*, finite total-order family admission for* L*, linear-extension compatibility of each row of* L *with the* $\mathsf{Poset}^{\uparrow}$ *comparison, intersection exactness through* I*, minimal-cardinality witnessing through* M*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 7808.1. The poset source is the displayed row P, and the only extension data are the finitely listed total-order rows in L.

Read the compatibility and intersection rows next. Each total order in L extends the comparison of P, while I records the reverse read: a comparison belongs to P precisely when all listed total orders compare the pair in that direction.

Finally inspect the minimality, transport, replay, provenance, and naming rows M, H, C, Q, N. Since the carrier has no coordinate for all host linear orders, quotient posets, selected extension families, or ambient cardinal invariants, the NameCert obligations are exactly the displayed finite rows. □

Theorem 7808.3 (Dushnik–Miller linear-extension route)**.** *Every downstream consumer of an accepted* $\mathsf{DushnikMillerDimension}^{\uparrow}$ *packet that reads order dimension must factor through*

$$P, L \longmapsto I, M \longmapsto H, C, Q, N.$$

The consumer sees a $\mathsf{Poset}^{\uparrow}$ *source, a finite family of* $\mathsf{TotalOrder}^{\uparrow}$ *linear-extension rows over the same carrier, the intersection ledger, and the minimal-cardinality witness before any dimension value is public. It receives no host powerset of orders, quotient poset, choice-selected family of extensions, or ambient set-theoretic dimension argument.*

Proof. Begin with the accepted carrier $D = (P, L, I, M, H, C, Q, N)$. A dimension consumer can inspect the source order only through P, and it can inspect linear extensions only through the finite family L.

The intersection ledger I is then load-bearing: it states both that every row of L extends P and that the common comparison induced by all rows of L recovers exactly the displayed poset comparison. The minimality row M supplies the local cardinal witness and the refusal of smaller displayed families.

The rows H, C, Q, N transport, replay, source, and name the same finite route. Because no additional coordinate is present, the route cannot be replaced by a host powerset search, quotient-poset equality, a choice-selected family of extensions, or an ambient cardinal-invariant proof. □

Falsifiable prediction: A claimed $\mathsf{DushnikMillerDimension}^{\uparrow}$ read that omits the PosetUp source row, finite TotalOrderUp extension family, intersection ledger, minimal-cardinality witness, hsame, Cont, Pkg, or local $\mathsf{NameCert}$ row refutes the carrier of Definition 7808.1.

Independence witness: poset, preorder, totalorder, setlike: $\mathsf{DushnikMillerDimension}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a PosetUp source, a finite family of TotalOrderUp linear-extension rows, an intersection ledger, minimal-cardinality data, componentwise transport, replay, provenance, and local NameCert naming. PosetUp and PreorderUp supply comparison rows without finite extension families; TotalOrderUp supplies a single comparison surface without an intersection ledger; SetLikeUp supplies membership discipline without order-dimension minimality.

Closure status for $\mathsf{NameCert}_{\mathsf{DushnikMillerDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{DushnikMillerDimension}^{\uparrow}$ packet is generated from a PosetUp source, inherited PreorderUp comparison data, a finite family of TotalOrderUp linear-extension rows, an intersection ledger, a minimal-cardinality witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DushnikMillerDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7808.1, Definition 7808.2, and Definition 7808.3 seed the finite Dushnik–Miller order-dimension carrier over displayed poset, preorder, total-order, set-like, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DushnikMillerDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host powerset of orders, quotient poset equality, choice-selected extension families, ambient set-theoretic dimension, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, finite linear-extension admission, intersection exactness, minimal-cardinality witnessing, transport, replay, provenance, and local naming for BEDC.Derived.DushnikMillerDimensionUp.

Chapter 7809

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$

Following Bishop and Bridges on located metric spaces, regular Cauchy names, and completion by finite approximation, the $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$ packet records a metric source whose located-distance requests and Cauchy-readback obligations are already complete as displayed BEDC data. It sits beside the located source of chapter 2043, the complete-metric service of chapter 128, the completion boundary of chapter 1506, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a finite means route for located complete metric reads; it is not a quotient completion, a selected-limit principle, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7809.1 (Bishop located complete metric carrier)**.** *A* $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, L, C, Q, W, R, E, H, T, P, N)$$

with the following displayed rows:

1. *M is the metric source row, exposing the inspected points, distance requests, symmetry, triangle, and zero-distance comparison used by the packet.*
2. *L is the* $\mathsf{LocatedSpace}^{\uparrow}$ *row, recording the finite positive-distance and locatedness requests attached to the metric source.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, displaying the Cauchy-admission and completion-readback obligations available to accepted requests.*
4. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary row, available only as the finite completion route attached to M, L, C and not as a quotient carrier.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the inspected points, Cauchy approximants, tolerances, and completion-facing requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows and regular rational distance observations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after M, L, C, Q, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the metric, locatedness, completeness, completion-boundary, window, readback, seal, replay, provenance, and naming rows.*

9. *T records displayed* Cont *replay of every located complete metric read through*

$$M, L, C, Q, W, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, C, Q, W, R, E, H, T, P, N$. It has no coordinate for quotient completion, selected limits, countable choice, open-cover compactness, host real equality, trichotomy, Lean verification, or a standard bridge.

Theorem 7809.2 (Bishop located complete metric NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$ *carrier*

$$B = (M, L, C, Q, W, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, locatedness requests, complete-metric service, completion boundary, stream windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *locatedness stability, requiring every transported or replayed distance-to-source request to keep the same M, L, W, R rows before it can read the terminal seal E;*
3. *completeness stability, requiring every Cauchy-facing request to pass through the displayed service row C and the completion-boundary row Q before any regular rational or real-name read is exposed;*
4. *finite readback exactness, requiring the* $\mathsf{StreamName}^{\uparrow}$ *window W and* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to carry all tolerance, approximant, and distance observations used by the completion-facing route;*
5. *ledger refusal, requiring H, T, P, N to transport, replay, source, and name only the displayed rows, with no quotient completion, selected limit, countable choice, host real equality, trichotomy, Lean verification, or bridge credential.*

No Bishop located complete metric NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7809.1. The accepted packet displays the metric source, locatedness row, complete-metric service, completion boundary, stream windows, regular rational readback, real seal, and structural rows.

Locatedness stability is componentwise. Transport and replay preserve the same metric source and locatedness request before the stream window or regular readback can be inspected.

Completeness stability is read through C and Q. A Cauchy-facing request may use the complete-metric service only through the displayed completion boundary, so it receives a finite route rather than a quotient representative or selected host limit.

Finite readback exactness follows from the order of W, R, E. The stream windows expose the inspected approximants and tolerances, the regular rational row records the distance readbacks, and only then can the real seal be consumed.

Finally, H, T, P, N contain only transport, replay, provenance, and naming for the displayed rows. Since the carrier has no quotient, choice, host equality, trichotomy, Lean verification, or bridge coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7809.3 (Bishop located complete metric finite completion route)**.** *Every completion-facing read of an accepted* $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$ *carrier*

$$B = (M, L, C, Q, W, R, E, H, T, P, N)$$

factors through the finite route

$$M, L, C, Q, W, R \longmapsto E$$

with structural support H, T, P, N*. The route first exposes the metric and locatedness rows, then the complete-metric service and completion boundary, then the stream windows and regular rational readbacks, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient completion, selected limit, countable choice, open-cover compactness, host real equality, trichotomy, Lean verification, or standard bridge.*

Proof. Start with the carrier projection. The metric and locatedness rows M, L are the only source-side rows available to a completion-facing request.

The read then passes through C, Q. These rows display the complete-metric service and the completion boundary as finite data, so the request cannot jump to a quotient completion or a selected limit.

Next the route opens W, R. The stream windows carry the inspected approximants and tolerances, while the regular rational row records the finite distance readbacks consumed by the completion route.

Only after those rows have been displayed can the terminal real seal E be read. The structural rows H, T, P, N preserve the same order under transport, replay, provenance, and local naming, and no excluded object is present in the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7809.1, Definition 7809.2, and Definition 7809.3 seed the finite located complete metric route through metric, locatedness, complete-metric, completion-boundary, stream-window, regular-readback, real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, open-cover compactness, host real equality, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locatedness stability, completeness stability, finite readback exactness, completion-boundary nonescape, and provenance locality for BEDC.Derived.BishopLocatedCompleteMetricUp.

Constructive story: A $\mathsf{BishopLocatedCompleteMetric}^{\uparrow}$ packet is generated from a Metric source row, a LocatedSpace request row, a CompleteMetric service row, a MetricCompletion boundary row, StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7810

A Concrete Naming Certificate for $\mathsf{CompactModulusEvaluation}^{\uparrow}$

Following Heine–Cantor and Bishop's finite-net reading of compact uniform continuity, the CompactModulusEvaluat packet records how a compact metric source, a continuous-map row, and pointwise modulus data are evaluated into a finite uniform-modulus output without invoking open-cover compactness or a selected subcover theorem. It concretizes the compact uniform continuity direction of chapter 9237 beside the compact metric source of chapter 129, the continuous-map surface of chapter 130, the uniform-modulus consumer of chapter 369, the dyadic tolerance ledger of chapter 291, the finite observation interface of chapter 126, and the terminal real seal of chapter 38. The packet is a finite NameCert evaluation route, not an external compactness theorem, choice principle, host equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7810.1 (Compact modulus evaluation carrier)**.** *A* $\mathsf{CompactModulusEvaluation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, P, D, N, U, W, R, S, H, C, G, Q)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, displaying the inspected probes, mesh radius, and compact-metric provenance.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise continuity reads are being evaluated on the compact source.*
3. *P is the pointwise modulus row attached to the inspected probes and tolerance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the requested output radius, finite mesh slack, and lower-bound comparisons.*
5. *N is the finite net and overlap row, recording which inspected probe covers each source request before the modulus is read.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced after the finite lower-bound fold.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for point observations, probe reads, and requested tolerances.*
8. *R is the regular rational readback row attached to the inspected windows and dyadic tolerance comparisons.*

9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after K, F, P, D, N, U, W, R have been displayed.*

10. *H records componentwise* hsame *transport for the compact source, continuous map, pointwise modulus, dyadic ledger, finite net, uniform handoff, window, readback, seal, replay, provenance, and naming rows.*

11. *C records displayed* Cont *replay of every compact modulus evaluation through*

$$K, F, P, D, N, W, R \longmapsto U, S.$$

12. *G is the* Pkg *provenance over* $\mathsf{CompactModulusEvaluation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *Q is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusEvaluation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, P, D, N, U, W, R, S, H, C, G, Q$. It has no coordinate for arbitrary open covers, selected subcovers, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or a standard bridge.

Theorem 7810.2 (Compact modulus evaluation NameCert obligations)**.** *For every accepted* CompactModulusEvalu *carrier*

$$E = (K, F, P, D, N, U, W, R, S, H, C, G, Q),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusEvaluation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, continuous map, pointwise modulus data, dyadic tolerance ledger, finite net and overlap rows, uniform-modulus handoff, stream windows, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*

2. *compact-source stability, requiring every transported or replayed evaluation to keep the same finite-net source K and finite net row N before any modulus handoff can be consumed;*

3. *pointwise-modulus stability, requiring the continuous-map row F and pointwise modulus row P to be read at the displayed probe windows and never as a host-level function predicate;*

4. *dyadic fold exactness, requiring the tolerance ledger D and readback row R to carry all lower-bound, slack, and radius comparisons used to produce U;*

5. *ledger refusal, requiring H, C, G, Q to transport, replay, source, and name only the displayed rows, with no arbitrary open cover, selected subcover, host function equality, quotient real equality, countable choice, trichotomy, Lean verification, or bridge credential.*

No compact modulus evaluation NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7810.1. The accepted packet displays the compact source, continuous map, pointwise modulus, dyadic ledger, finite net, uniform handoff, stream windows, regular readback, real seal, and structural rows.

Compact-source stability is componentwise. Transport and replay keep the same finite-net source and cover row, so the packet never replaces the displayed compact metric data by an arbitrary open cover.

Pointwise-modulus stability is read through F, P, W. The continuous map and its pointwise modulus are inspected only at displayed probe windows, not through a host-level function equality predicate.

Dyadic fold exactness is the shared role of D, N, R, U. The finite net supplies the inspected probes, the dyadic ledger records the lower-bound and slack comparisons, the regular rational readback records the finite observations, and the uniform-modulus row is downstream of those data.

Finally, H, C, G, Q only transport, replay, source, and name the displayed rows. Since the carrier contains no open-cover theorem, selected subcover, choice row, host equality coordinate, Lean verification, or bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 7810.3 (Compact modulus evaluation finite net route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *output exported by an accepted* $\mathsf{CompactModulusEvaluation}^{\uparrow}$ *carrier*

$$E = (K, F, P, D, N, U, W, R, S, H, C, G, Q)$$

factors through the finite route

$$K, F, P, D, N, W, R \longmapsto U, S$$

with structural support H, C, G, Q*. The route first exposes the compact metric finite-net source, then the continuous-map and pointwise-modulus reads, then the dyadic tolerance fold and regular rational readback, and only then the uniform-modulus handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no arbitrary open-cover compactness, selected subcover, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or standard bridge.*

Proof. Start with the carrier projection. The compact metric source K and finite net row N are the only compact-facing rows available to the evaluation.

The continuous-map and pointwise-modulus data are read through F, P, W. Each inspected probe and tolerance request is attached to a displayed stream window, so no host-level function predicate is introduced.

The dyadic tolerance ledger D and regular rational readback R then perform the finite lower-bound and slack comparisons. These rows are upstream of the uniform-modulus handoff U.

Only after those rows have been displayed can the output row U and terminal seal S be consumed. Transport, replay, provenance, and naming through H, C, G, Q preserve the same order, and the excluded open-cover, selection, choice, equality, verification, and bridge data are absent from the accepted packet. □

Theorem 7810.4 (Compact modulus evaluation sibling route)**.** *Every* $\mathsf{CompactModulusEvaluation}^{\uparrow}$ *output exported by an accepted carrier*

$$E = (K, F, P, D, N, U, W, R, S, H, C, G, Q)$$

factors through the sibling route

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before any uniform-modulus consumer reads it. More precisely, the compact metric finite-net source, continuous-map row, pointwise modulus data, dyadic tolerance ledger, stream-window schedule, regular rational readback, and terminal Real seal are displayed before the $\mathsf{UniformModulus}^{\uparrow}$ *handoff is consumed. The route exports no arbitrary open-cover compactness, selected subcover, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or standard bridge.*

Proof. Project the accepted carrier using Definition 7810.1. The rows $K, F, P, D, N, U, W, R, S$ display the compact metric source, continuous map, pointwise modulus, dyadic ledger, finite net, uniform handoff, stream windows, regular readback, and Real seal.

Apply Definition 7810.2. Carrier admission and compact-source stability keep K and N upstream of the uniform-modulus handoff, while pointwise-modulus stability keeps F and P inside the displayed stream windows.

Now use Definition 7810.3. The finite net route reads D, W, R before U, S, so the dyadic tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal are all visible before the output is consumed.

Transport, replay, provenance, and naming through H, C, G, Q preserve this sibling order. Since the accepted packet contains no open-cover theorem, selected subcover, host function equality, quotient real equality, choice row, trichotomy row, Lean verification, or bridge credential, none can be exported by a uniform-modulus consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusEvaluation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusEvaluation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7810.1, Definition 7810.2, and Definition 7810.3 seed the finite compact-modulus route through compact metric, continuous-map, pointwise-modulus, dyadic-ledger, finite-net, stream-window, regular-readback, uniform-modulus, real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusEvaluation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source stability, pointwise-modulus stability, dyadic fold exactness, finite-net handoff, uniform-modulus nonescape, and provenance locality for BEDC.Derived.CompactModulusEvalua

Constructive story: A $\mathsf{CompactModulusEvaluation}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise reads, pointwise modulus data, a DyadicRatCore tolerance ledger, finite net and overlap rows, StreamName windows, RegSeqRat readback, a UniformModulus handoff, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7811

A Concrete Naming Certificate for $\mathsf{CauchyModulusFusion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy convergence with explicit moduli, the $\mathsf{CauchyModulusFusion}^{\uparrow}$ packet records the finite surface that fuses two displayed Cauchy-modulus sources into one shared tail schedule before Real and Completion consumers inspect a seal. It sits beside the majorant route of chapter 4087, the dominance route of chapter 1814, the rate lattice route of chapter 1818, the dyadic modulus lattice of chapter 2386, the finite window service of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38.

Definition 7811.1 (Cauchy modulus fusion carrier)**.** *A* $\mathsf{CauchyModulusFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M_0, M_1, J, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. M_0 *and* M_1 *are the two source* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *schedules whose tail requests are to be read on one shared precision surface;*
2. J *is the fused join row, read through* $\mathsf{CauchyRateDominance}^{\uparrow}$, $\mathsf{CauchyRateLattice}^{\uparrow}$, *and* $\mathsf{DyadicCauchyModulusLatt}$ *witnessing that both source schedules are served by the same displayed tail threshold;*
3. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened only after* M_0, M_1, J *have been displayed;*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the fused window route;*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the comparison and fused-window readback;*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the two sources, fused join, window, readback, and dyadic ledger have all been displayed;*
7. H *records componentwise* hsame *transport for the two sources, fused join, window, readback, tolerance, seal, replay, provenance, and local naming rows;*
8. C *records displayed* Cont *replay from a precision request through* M_0, M_1, J, W, Q, D, E *and then through the structural rows;*
9. P *is the* Pkg *provenance over* $\mathsf{CauchyModulusFusion}^{\uparrow}$, $\mathsf{CauchyModulusMajorant}^{\uparrow}$, $\mathsf{CauchyRateDominance}^{\uparrow}$, $\mathsf{CauchyRateLattice}^{\uparrow}$, $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyModulusFusion}^\uparrow}$ *row.*

The classifier has no coordinate for quotient stream equality, selected limits, host real comparison, countable choice, trichotomy, independent per-source Real seals, ambient $\mathsf{Real}^\uparrow$ *completeness, Lean verification, or bridge checking.*

Theorem 7811.2 (Cauchy modulus fusion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusFusion}^\uparrow$ *carrier $F = (M_0, M_1, J, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusFusion}^\uparrow}$ *surface consists exactly of carrier admission, paired source-modulus exposure through M_0, M_1, fused join exposure through J, finite-window service through W, regular rational readback through Q, dyadic tolerance exposure through D, Real seal ordering through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7811.1. The tuple lists exactly two source schedules, one fused join row, one finite-window row, one regular readback row, one dyadic tolerance ledger, one terminal Real seal, and the structural transport, replay, provenance, and local naming rows.

The sibling routes determine the fusion surface. $\mathsf{CauchyModulusMajorant}^\uparrow$ supplies the two source schedules, while $\mathsf{CauchyRateDominance}^\uparrow$, $\mathsf{CauchyRateLattice}^\uparrow$, and $\mathsf{DyadicCauchyModulusLattice}^\uparrow$ force the join row J to dominate both sources on the same displayed dyadic precision surface.

The observation and seal rows are then downstream. $\mathsf{StreamName}^\uparrow$ opens W, $\mathsf{RegSeqRat}^\uparrow$ reads Q, $\mathsf{DyadicRatCore}^\uparrow$ supplies D, and $\mathsf{Real}^\uparrow$ consumes the terminal seal E only after the fused route is visible. The structural rows H, C, P, N preserve that same finite route and add no hidden analytic coordinate. □

Theorem 7811.3 (Cauchy modulus fusion Real and Completion route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CauchyModulusFusion}^\uparrow$ *packet must factor through*

$$M_0, M_1, J, W, Q, D, E, H, C, P, N.$$

Thus the consumer receives one shared fused-modulus surface before it reads the $\mathsf{Real}^\uparrow$ *seal, not two independent per-source tail choices, a quotient stream equality, a selected limit, a host product of schedules, or an ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 7811.2. They place every accepted read behind the two source schedules M_0, M_1 and the fused join row J before any window or seal can be consumed.

Now read the finite observation route. The rows W, Q, D provide the StreamName window, regular rational readback, and dyadic tolerance ledger used by downstream Real and Completion consumers, and the Real seal E is downstream of those rows.

Finally apply the non-escape clauses of the carrier. Since no coordinate stores independent seals, quotient stream equality, selected limits, a host schedule product, or ambient completeness, replay through C, transport through H, provenance through P, and local naming through N preserve only the displayed fused route. □

Falsifiable prediction: An accepted $\mathsf{CauchyModulusFusion}^\uparrow$ Real or Completion consumer that reaches the $\mathsf{Real}^\uparrow$ seal without first displaying both source schedules, the fused join row, the finite $\mathsf{StreamName}^\uparrow$ window, the $\mathsf{RegSeqRat}^\uparrow$ readback, and the $\mathsf{DyadicRatCore}^\uparrow$ tolerance ledger refutes the carrier surface.

Independence witness: cauchy-modulus-majorant, cauchy-rate-dominance, cauchy-rate-lattice, dyadic-cauchy-modulus-lattice, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole eleven-row packet $F = (M_0, M_1, J, W, Q, D, E, H, C, P, N)$, because each sibling

contributes only source majorants, dominance, lattice comparison, windows, readback, dyadic tolerance, or Real sealing while this chapter binds the two-source fused-modulus route into one local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusFusion}\uparrow}$.

Constructive story: A $\mathsf{CauchyModulusFusion}^{\uparrow}$ packet is generated from two CauchyModulusMajorant source schedules, a fused join row over CauchyRateDominance, CauchyRateLattice, and DyadicCauchyModulusLattice, a StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusFusion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7811.1, Definition 7811.2, and Definition 7811.3 seed the two-source shared-modulus route used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusFusion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real comparison, countable choice, trichotomy, independent per-source Real seals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fused join exposure, finite-window service, regular readback, dyadic tolerance exposure, Real seal ordering, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFusionUp.

Chapter 7812

A Concrete Naming Certificate for $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$ packet records the finite certificate surface for replacing a bounded window of one regular rational Cauchy name by a compatible window from another. It sits over the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite stream-surgery interface for Real and Completion consumers; it is not host sequence equality, quotient representative selection, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7812.1 (Bishop regular sequence splice carrier)**.** *A* $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, R, U, B, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *stream rows whose finite windows are candidates for the splice.*
2. *U is a unary splice-window row, recording the displayed interval of stream positions where the right row is read instead of the left row.*
3. *B is the dyadic boundary agreement ledger at the two splice edges.*
4. *W records the paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows inspected before, inside, and after U.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for all comparisons in W and across B.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the spliced stream assembled from L, R, U, B, W, D.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after the splice window, boundary ledger, stream observations, dyadic tolerances, and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for both source streams, the splice window, boundary agreement, stream windows, dyadic ledger, regular readback, seal boundary, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite route.*

10. *P records* Pkg *provenance over* $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularSequenceSplice}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, R, U, B, W, D, Q, E, H, C, P, N$. It has no coordinate for host sequence equality, quotient representatives, countable choice, selected limits, host real equality, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 7812.2 (Bishop regular sequence splice regularity)**.** *For every accepted* $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$ *carrier*

$$S = (L, R, U, B, W, D, Q, E, H, C, P, N),$$

the spliced row Q is a regular rational Cauchy readback over the displayed finite route. Its regularity is witnessed by the left $\mathsf{RegSeqRat}^{\uparrow}$ *row before U, by the right* $\mathsf{RegSeqRat}^{\uparrow}$ *row inside U, by the left* $\mathsf{RegSeqRat}^{\uparrow}$ *row after U, and by the boundary agreement ledger*

$$B : \quad \begin{array}{c} L(a) \sim_D R(a), \\ R(b) \sim_D L(b) \end{array}$$

at the displayed endpoints a, b of the unary splice window.

Proof. Project the carrier of Definition 7812.1. The rows L and R supply the two regular rational source streams, U names the finite splice interval, W names the inspected stream windows, and D supplies the dyadic tolerances used by every comparison.

Before the splice window, the assembled row Q reads from L. Regularity there is exactly the regularity already displayed by the left $\mathsf{RegSeqRat}^{\uparrow}$ row, transported through the corresponding coordinates of H and replayed through C.

Inside the splice window, Q reads from R. Regularity there is exactly the regularity already displayed by the right $\mathsf{RegSeqRat}^{\uparrow}$ row, with the same dyadic tolerance row D and the same finite observations W.

After the splice window, Q returns to L. This is the same left-row regularity argument as before the window, again transported and replayed by the structural rows.

It remains to cross the two boundaries. The ledger B gives the displayed dyadic agreement at a and b, and D records the comparison budgets used at those edges. Combining the source regularity rows with the two boundary agreements yields the regularity of Q across the splice boundary.

No other equality source is available in the carrier. Thus the regularity proof is confined to the displayed source streams, unary window, dyadic ledger, stream observations, transport, replay, provenance, and local naming rows. □

Theorem 7812.3 (Bishop regular sequence splice Real seal)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$ *carrier reaches terminal* $\mathsf{Real}^{\uparrow}$ *evidence only through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The route first displays the left and right source streams, the unary splice window, the dyadic boundary agreement ledger, finite stream windows, and the spliced regular readback. It exports no host sequence equality, quotient representative, countable-choice schedule, selected limit, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Use Definition 7812.2. The theorem identifies Q as the regular rational readback produced from L, R, U, B, W, D and the structural rows.

The seal row E is downstream of that route in the carrier definition. A consumer therefore reaches E only after the source streams, splice window, boundary agreement, stream observations, dyadic tolerances, and regular readback have been displayed.

Inspect the remaining rows. The transport row H, continuation row C, provenance row P, and naming row N preserve the same finite packet and do not add a quotient representative, selected limit, countable-choice schedule, host sequence equality, host real equality, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularSequenceSplice}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularSequenceSplice}^{\uparrow}$ packet is generated from left and right RegSeqRat streams, a unary splice window, a dyadic boundary agreement ledger, finite StreamName windows, DyadicRatCore tolerance accounting, a spliced RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularSequenceSplice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7812.1, Definition 7812.2, and Definition 7812.3 seed the finite regular-sequence splice route from RegSeqRat windows to Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularSequenceSplice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence equality, quotient representatives, countable choice, selected limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, splice-window coverage, boundary agreement, regularity preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularSequenceSpliceU

Chapter 7813

A Concrete Naming Certificate for DyadicIntervalSubdivisionTree$^{\uparrow}$

Bishop–Bridges constructive interval bisection and Moore–Kearfott–Cloud interval analysis both use finite subdivision trees before any completed interval limit is read. The DyadicIntervalSubdivisionTree$^{\uparrow}$ packet records that finite tree as NameCert data between the one-step dyadic refinement carrier of chapter 1831, the endpoint arithmetic surface of chapter 1645, the nested Cauchy interval consumer of chapter 1811, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It is a finite L10 subdivision carrier, not a selected infinite branch, a quotient stream equality principle, trichotomy, countable choice, open-cover compactness, or ambient Real$^{\uparrow}$ completeness.

Definition 7813.1 (Dyadic interval nested subdivision carrier)**.** *A* DyadicIntervalSubdivisionTree$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, D, V, E, L, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the root* DyadicInterval$^{\uparrow}$ *row whose lower and upper dyadic endpoints are read through* DyadicRatCore$^{\uparrow}$*.*
2. *D is the finite depth budget, naming the largest inspected level and the dyadic scale attached to each level.*
3. *V is the finite child-incidence table, recording for every displayed internal node the two displayed child interval rows supplied by* DyadicIntervalRefinement$^{\uparrow}$*.*
4. *E is the endpoint-reuse ledger, identifying each child endpoint with the parent endpoints or the displayed midpoint row through* DyadicIntervalArithmetic$^{\uparrow}$*.*
5. *L is the retained-leaf coverage row, listing the leaves that remain available to downstream interval consumers after the finite subdivision.*
6. *S is the* StreamName$^{\uparrow}$ *observation window, naming the finite node, endpoint, and leaf reads inspected by the tree consumer.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the scheduled dyadic endpoint observations.*
8. *H records componentwise* hsame *transport for the root, depth, incidence, endpoint-reuse, leaf-coverage, window, readback, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay of every accepted subdivision read through*

$$R, D, V, E, L, S \longmapsto Q.$$

10. *P is the* Pkg *provenance over* DyadicIntervalSubdivisionTree$^{\uparrow}$, DyadicInterval$^{\uparrow}$, DyadicIntervalRefinement$^{\uparrow}$, DyadicIntervalArithmetic$^{\uparrow}$, NestedDyadicCauchy$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{DyadicIntervalSubdivisionTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, V, E, L, S, Q, H, C, P, N$. It has no coordinate for a selected infinite branch, quotient stream equality, trichotomy, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 7813.2 (Dyadic interval nested subdivision obligations)**.** *For every accepted* DyadicIntervalSubdivisionTre *carrier*

$$T = (R, D, V, E, L, S, Q, H, C, P, N),$$

the local NameCert surface consists exactly of carrier admission for the root dyadic interval, finite depth budget, child-incidence table, endpoint-reuse ledger, retained-leaf coverage, StreamName$^{\uparrow}$ *observation window,* RegSeqRat$^{\uparrow}$ *readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. Every downstream* NestedDyadicCauchy$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer of the subdivision must expose those rows before it may inspect a leaf chain or readback boundary.*

Proof. Project the carrier of Definition 7813.1. The root row, finite depth budget, child-incidence table, endpoint-reuse ledger, retained leaves, stream windows, readback, and structural rows are exactly the displayed coordinates of the packet.

The one-step refinement route is supplied by chapter 1831, while endpoint reuse and midpoint arithmetic are read through chapter 1645. Thus each internal node is certified by finite dyadic interval rows rather than by a completed path.

The observation and readback rows S, Q are the only route toward NestedDyadicCauchy$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ consumers. Since H, C, P, N only transport, replay, source, and name the same displayed rows, no selected infinite branch, quotient stream equality, trichotomy, countable-choice row, open-cover compactness proof, or ambient completeness theorem can be projected from the carrier. □

Closure status for NameCert$_{\text{DyadicIntervalSubdivisionTree}^{\uparrow}}$.

Constructive story: A DyadicIntervalSubdivisionTree$^{\uparrow}$ packet is generated from a root DyadicInterval row, finite depth budget, child-incidence table, endpoint-reuse ledger, retained-leaf coverage row, StreamName observation window, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicIntervalSubdivisionTree}^{\uparrow}}$, seed)

Scope closed: Definition 7813.1 and Definition 7813.2 seed the finite subdivision-tree carrier over dyadic interval roots, one-step refinements, endpoint arithmetic, retained leaves, stream windows, regular readback, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicIntervalSubdivisionTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, quotient stream equality, trichotomy, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, child-incidence exactness, endpoint-reuse stability, retained-leaf coverage, finite StreamName exposure, RegSeqRat readback, consumer nonescape, and local naming for BEDC.Derived.DyadicIntervalSubdivisionTreeUp.

Chapter 7814

A Concrete Naming Certificate for $\mathsf{RegularCauchyInduction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyInduction}^{\uparrow}$ packet records the finite induction surface by which a displayed regular Cauchy real-name construction advances from a base window to a successor tail window before any $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the dyadic arithmetic of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the real-name boundary of chapter 38, and the regular-completion route of chapter 7398. The packet is an induction bridge for finite regular Cauchy windows; it is not countable choice, quotient equality of Cauchy names, a selected limit operator, host real arithmetic, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7814.1 (Regular Cauchy induction carrier)**.** *A* $\mathsf{RegularCauchyInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, S, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the base window row, containing the initial finite* $\mathsf{StreamName}^{\uparrow}$ *observations and the dyadic tolerance ledger visible at the first construction stage.*
2. *S is the successor and tail-step row, recording the rule that extends an accepted displayed window to the next regular Cauchy tail request.*
3. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row used by both B and S for radius comparison and tail contraction.*
4. *W is the finite request-window ledger listing every stream coordinate read by the base and successor rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback row attached to B, S, Q, W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the finite induction route has replayed the base window, successor row, dyadic budget, request windows, and regular readback.*
7. *H records componentwise* hsame *transport for the base window, successor step, dyadic budget, request windows, regularity readback, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$B, S, Q, W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyInduction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealRegularCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInduction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, S, Q, W, R, E, H, C, P, N$. It has no coordinate for a hidden infinite choice sequence, quotient equality of regular Cauchy names, a selected limit point, host real arithmetic, trichotomy, or ambient completeness.

Theorem 7814.2 (Regular Cauchy induction step soundness)**.** *For every accepted* $\mathsf{RegularCauchyInduction}^{\uparrow}$ *carrier*

$$I = (B, S, Q, W, R, E, H, C, P, N),$$

the successor row S may advance a regular Cauchy construction only along request windows already displayed by W and budgeted by Q. After componentwise transport through H and replay through C, the same finite route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ *readback R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. No step exports countable choice, quotient equality of names, host real comparison, or a selected limit.*

Proof. Project the accepted carrier onto its displayed rows. The only induction data are the base window B, the successor and tail-step row S, the dyadic budget Q, and the finite request ledger W.

Induct on the displayed BHist request window. In the base case, the request is already listed by B and budgeted by Q, so the route reaches R without reading any hidden tail coordinate.

For the successor case, the induction hypothesis gives the prior displayed window. The row S appends exactly the next tail request named in W, while Q supplies the dyadic comparison budget needed for the regularity readback.

Transport through H and replay through C preserve that order. The rows P and N only record provenance and local naming, and the seal E is downstream from R, so no unlisted choice sequence, quotient-equality test, host real comparison, or selected limit can enter the step. □

Theorem 7814.3 (Regular Cauchy induction finite-window coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyInduction}^{\uparrow}$ *carrier receives the finite L10 handoff*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{BishopRealRegularCompletion}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed route $B, S, Q, W, R, E, H, C, P, N$. The handoff certifies only finite base-window coverage, successor-tail coverage, regularity readback, seal preservation, transport, replay, provenance, and local naming.

Proof. Run finite induction on the request windows displayed in W. The base window is supplied by B, so the initial stream observations and dyadic tolerance reads are part of the public packet.

At each successor stage, the row S extends the displayed window and consumes the same dyadic budget row Q. The induction hypothesis supplies all earlier reads, so the extended packet remains finite and visible in W.

Once the displayed window is fixed, R is the only regular rational readback and E is the only real seal boundary. Componentwise transport, continuation replay, provenance, and local naming through H, C, P, N keep the same route.

Thus a consumer sees exactly the finite handoff through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealRegularCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Nothing in the carrier supplies a countable-choice schedule, quotient equality of Cauchy names, a host real primitive, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyInduction}^{\uparrow}$ packet is generated from a base window, successor and tail-step row, DyadicRatCore budget, finite StreamName request windows, RegSeqRat regularity readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7814.1, Definition 7814.2, and Definition 7814.3 seed the finite induction bridge from regular Cauchy request windows through regular rational readback to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient equality of Cauchy names, selected limits, host real arithmetic, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, base-window coverage, successor-tail coverage, regularity readback, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyInductionUp.

Chapter 7815

A Concrete Naming Certificate for $\mathsf{CeaLemma}^{\uparrow}$

Following Cea and standard Galerkin error analysis, the $\mathsf{CeaLemma}^{\uparrow}$ packet records the finite quasi-optimality route that turns coercivity, continuity, variational exactness, and Galerkin residual orthogonality into a displayed best-approximation certificate. It sits after the projection row of chapter 5169, the error-estimate surface of chapter 6619, the variational solve of chapter 3419, the Hilbert comparison route of chapter 91, and the norm ledger of chapter 89. The packet names the quasi-optimal projection bound without importing a host Hilbert projection theorem, quotient function space, selected minimizer, or choice principle.

Definition 7815.1 (Cea lemma carrier)**.** *A* $\mathsf{CeaLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, a, u, u_h, c, M, B, E, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite trial-space row consumed from* $\mathsf{GalerkinProjection}^{\uparrow}$*.*
2. *a is the coercive bilinear-form row, read through* $\mathsf{LaxMilgram}^{\uparrow}$ *and the Hilbert comparison surface.*
3. *u is the exact variational solution row supplied by the displayed Lax–Milgram route.*
4. *u_h is the Galerkin projection row, including residual orthogonality over V.*
5. *c is the coercivity scalar row.*
6. *M is the continuity scalar row for the same bilinear-form comparison.*
7. *B is the finite best-approximation ledger comparing u to each displayed trial competitor.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *bound row recording the quasi-optimal estimate against the* $\mathsf{Norm}^{\uparrow}$ *ledger.*
9. *H records componentwise* hsame *transport for trial-space, bilinear-form, exact-solution, Galerkin, scalar, best-approximation, bound, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$V, a, u, u_h, c, M, B \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CeaLemma}^{\uparrow}$, $\mathsf{GalerkinProjection}^{\uparrow}$, $\mathsf{GalerkinErrorEstimate}^{\uparrow}$, $\mathsf{LaxMilgram}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{CeaLemma}^\uparrow}$ *row.*
The classifier compares accepted packets only through $V, a, u, u_h, c, M, B, E, H, T, P, N$. It has no coordinate for a quotient function space, selected best approximant, host projection operator, global minimizer, countable choice, or ambient Hilbert-space theorem.

Theorem 7815.2 (Cea lemma NameCert obligations)**.** *For every accepted* $\mathsf{CeaLemma}^\uparrow$ *carrier*

$$K = (V, a, u, u_h, c, M, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CeaLemma}^\uparrow}$ *surface consists exactly of carrier admission, Galerkin residual-orthogonality exposure, Lax–Milgram variational exposure, coercivity and continuity scalar exposure, finite best-approximation comparison, scalar-bound readback, componentwise transport, displayed replay, provenance, and local naming. Every row is read through the displayed siblings* $\mathsf{GalerkinProjection}^\uparrow$, $\mathsf{GalerkinErrorEstimate}^\uparrow$, $\mathsf{LaxMilgram}^\uparrow$, $\mathsf{Hilbert}^\uparrow$, *and* $\mathsf{Norm}^\uparrow$ *before the quasi-optimality seal E is consumed.*

Proof. Project the carrier of Definition 7815.1. The finite trial row, bilinear form, exact solution, Galerkin row, scalar constants, comparison ledger, bound, transport, replay, provenance, and local name are the full visible inventory.

The Galerkin side is read through chapter 5169; the exact variational side is read through chapter 3419. The scalar rows c, M are attached to the same Hilbert and norm comparison surface, so the estimate cannot switch to a host norm or a different bilinear form.

The best-approximation row B then compares only displayed trial competitors, and the error-estimate sibling of chapter 6619 supplies the route to the bound row E. Since H, C, P, N transport, replay, source, and name only these rows, the listed obligations exhaust the local certificate. □

Theorem 7815.3 (Cea quasi-optimal projection route)**.** *Every public* $\mathsf{CeaLemma}^\uparrow$ *quasi-optimality read factors through the route*

$$\mathsf{GalerkinProjection}^\uparrow \longmapsto \mathsf{GalerkinErrorEstimate}^\uparrow \longmapsto \mathsf{LaxMilgram}^\uparrow \longmapsto (\mathsf{Hilbert}^\uparrow, \mathsf{Norm}^\uparrow) \longmapsto \mathsf{Real}^\uparrow.$$

The consumer must first display the finite trial space, residual orthogonality, variational exactness, coercivity and continuity constants, and best-approximation ledger before reading the scalar quasi-optimal bound. No quotient function space, selected minimizer, host projection theorem, or choice principle is exported.

Proof. Start with the obligation surface of Definition 7815.2. It orders the finite trial row, the exact variational row, the Galerkin row, the scalar constants, and the best-approximation ledger before the bound row.

Use the Galerkin projection sibling to obtain residual orthogonality on the displayed trial space. Use the Lax–Milgram sibling for the exact variational equation and the Hilbert and norm siblings for the comparison surface on which coercivity and continuity are read.

Finally consume the Galerkin error-estimate route. Its best-approximation comparison feeds the same scalar bound row E, and the structural rows preserve the displayed path. The refused quotient, minimizer, host projection, and choice objects are absent from the carrier, so they cannot be exported by the quasi-optimality read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CeaLemma}^\uparrow}$**.**

Constructive story: A $\mathsf{CeaLemma}^\uparrow$ packet is generated from finite Galerkin trial-space rows, coercive bilinear-form rows, exact variational rows, Galerkin projection rows, coercivity and continuity scalar rows, a best-approximation ledger, a RealUp bound row, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CeaLemma}\uparrow}$, seed)

Scope closed: Definition 7815.1, Definition 7815.2, and Definition 7815.3 seed the finite quasi-optimal Galerkin projection route over GalerkinProjection, GalerkinErrorEstimate, LaxMilgram, Hilbert, Norm, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CeaLemma}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient function spaces, selected minimizers, host Hilbert projection theorems, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-orthogonality exposure, variational exposure, coercivity and continuity scalar exposure, best-approximation comparison, scalar-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.CeaLemmaUp.

Chapter 7816

A Concrete Naming Certificate for $\mathsf{FiniteTriggerCube}^{\uparrow}$

The $\mathsf{FiniteTriggerCube}^{\uparrow}$ packet realizes the conditional trigger ensemble of Definition 9462.1594 as finite BEDC data: cube coordinates, trigger subsets, blocker rows, energy rows, rank rows, and local naming are displayed before any probability-like read is consumed. The packet consumes the finite conditional-trigger ensemble of chapter 3870, the energy-biased trigger reliability rows of chapter 3583, the trigger-blocker duality rows of chapter 3801, and the trigger-energy partition rows of chapter 3806. The carrier is finite combinatorics over the trigger cube, but its role is BEDC-native: it turns the ensemble into a falsifiable NameCert surface rather than an ambient probability measure or an unbounded stochastic model.

Definition 7816.1 (Finite trigger-cube carrier)**.** *A* $\mathsf{FiniteTriggerCube}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, M, B, T, R, E, W, Z, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite cube row, naming the visible cube coordinates and face incidences.*
2. *M is the boundary-mask row, including the distinguished blocked mask $M^{\star}$ when the conditional trigger ensemble is read.*
3. *B is the finite subset row under inspection.*
4. *T is the trigger row, recording whether B meets the displayed full-trigger condition.*
5. *R is the blocker row, recording the complementary blocked condition used to form Z_{block}.*
6. *E is the energy row $\mathcal{E}(B)$.*
7. *W is the finite weight row $t^{|B|}y^{\mathcal{E}(B)}$.*
8. *Z is the partition-rank row, listing the finite sums used for Z_{all}, Z_{block}, and Z_{trig}.*
9. *H records componentwise* hsame *transport for cube, mask, subset, trigger, blocker, energy, weight, rank, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the finite route*

$$Q, M, B, T, R, E, W, Z \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteTriggerCube}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows, with a pointer back to the conditional trigger ensemble theorem of Definition 9462.1601.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, B, T, R, E, W, Z, H, C, P, N$. It has no coordinate for an infinite probability space, hidden sampler, host independence axiom, unbounded cube, selected random variable, or unstated measure.

Theorem 7816.2 (Finite trigger-cube NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTriggerCube}^{\uparrow}$ *carrier*

$$F = (Q, M, B, T, R, E, W, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}$ *surface consists exactly of cube admission, boundary-mask exposure, subset exposure, trigger exposure, blocker exposure, energy exposure, finite weight exposure, partition-rank exposure, componentwise transport, displayed replay, provenance, and local naming. A conditional-trigger consumer may read the distributional expressions of Definition 9462.1594 only after those finite rows have been displayed.*

Proof. Project the carrier of Definition 7816.1. The cube, mask, subset, trigger, blocker, energy, weight, and partition-rank rows are displayed before transport, replay, provenance, and local naming.

The trigger and blocker rows are separate fields, so a full-trigger read cannot silently replace the blocker complement by an unstated event. The energy and weight rows are finite table entries, and the partition-rank row records the finite sums that support the conditional expressions.

Transport and replay preserve the same displayed fields through H and C. Since no carrier coordinate names an infinite probability space, hidden sampler, host independence axiom, unbounded cube, selected random variable, or unstated measure, the listed finite rows exhaust the NameCert obligations. □

Theorem 7816.3 (Finite trigger-cube ensemble route)**.** *Every conditional trigger-ensemble read carried by* $\mathsf{FiniteTriggerCube}^{\uparrow}$ *factors through the finite cube route*

$$Q, M, B, T, R, E, W, Z \longmapsto \mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}.$$

The route displays the trigger row, blocker row, energy row, finite weight row, and partition-rank row before any expectation, reliability, or low-energy conditional table is read. It exports finite rows only, not a host probability space, hidden sampler, independence axiom, or unbounded stochastic model.

Proof. Apply Definition 7816.2. It gives the finite inventory through which the conditional trigger expressions must be read.

The rows T and R separate full-trigger and blocked cases, while E, W, Z display the energy, monomial weight, and finite partition sums. Thus the expectation and low-energy tables consume finite rank data rather than a measure-theoretic background object.

Finally, H, C, P, N preserve and name the same route. The refused probabilistic and unbounded objects are absent from the carrier, so the ensemble route remains a finite trigger-cube NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{FiniteTriggerCube}^{\uparrow}$ conditional read displays cube coordinates, boundary mask, subset, trigger, blocker, energy, finite weight, partition-rank, transport,

replay, provenance, and local NameCert rows before any expectation or reliability expression is consumed.

Independence witness: conditional-trigger-ensemble, finite-cube, blocker-ledger, energy-rank-table: none is bijective to $\mathsf{FiniteTriggerCube}^{\uparrow}$ because this packet jointly carries cube coordinates, boundary mask, trigger rows, blocker rows, energy rows, finite weights, partition ranks, transport, replay, provenance, and local naming in one finite carrier.

Theorem 7816.4 (Finite trigger-cube TasteGate obligations)**.** *A checked TasteGate for* $\mathsf{FiniteTriggerCube}^{\uparrow}$ *must exhibit a nonempty accepted trigger-cube packet and must preserve the displayed fields* $Q, M, B, T, R, E, W, Z, P$ *separately. In particular, changing the trigger row while holding the blocker row fixed, or changing the energy row while holding the subset row fixed, must be visible to the classifier before a consumer can treat the conditional ensemble as a named packet.*

Proof. The carrier definition lists trigger, blocker, energy, and rank as separate rows. A nonempty accepted packet witnesses that the carrier is not only an empty formal shell.

Field preservation is required because the conditional ensemble reads trigger and blocker data differently, and it reads energy before forming the finite weight row. Collapsing those fields would make the displayed partition-rank table unable to distinguish full-trigger, blocked, and energy-biased reads.

Thus the TasteGate obligation is to keep the finite fields visible before classifier acceptance. This is a paper-side obligation statement and does not assert a checked Lean target. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTriggerCube}^{\uparrow}$ packet is generated from finite cube coordinates, a boundary mask, a subset row, trigger and blocker rows, an energy row, a finite weight row, partition-rank rows, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7816.1, Definition 7816.2, Definition 7816.3, and Definition 7816.4 seed the finite trigger-cube realization of the conditional trigger ensemble as displayed cube, trigger, blocker, energy, rank, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTriggerCube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an infinite probability space, a host sampler, a measure-theoretic independence axiom, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked cube admission, trigger exposure, blocker exposure, energy exposure, weight exposure, partition-rank exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTriggerCubeUp.

Chapter 7817

A Concrete Naming Certificate for $\mathsf{EuclideanSpace}^{\uparrow}$

Following Euclid, Hilbert, and Bourbaki, the $\mathsf{EuclideanSpace}^{\uparrow}$ packet records a finite-dimensional real coordinate carrier whose metric, vector-space, inner-product, and matrix-readback rows are all displayed before any geometric consumer reads a distance, norm, projection, or coordinate transform. It sits beside the real scalar seal of chapter 38, the metric source of chapter 57 read as $\mathsf{Met}^{\uparrow}$, the vector-space source of chapter 47, the inner-product surface of chapter 86, and the finite matrix row of chapter 49. The packet is a finite coordinate NameCert surface for Euclidean spaces; it is not a host theorem about all finite-dimensional normed spaces, a selected orthonormal basis, quotient linear algebra, spectral completeness, or an ambient completeness claim for $\mathsf{Real}^{\uparrow}$.

Definition 7817.1 (EuclideanSpace carrier)**.** *A* $\mathsf{EuclideanSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, M, V, I, A, D, N, H, C, P, Q)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal and finite coordinate readback row.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row for the displayed coordinate carrier.*
3. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row recording zero, addition, inverse, and real scalar multiplication on the displayed finite coordinate windows.*
4. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row whose bilinear and positive rows compute the Euclidean norm and distance used by M.*
5. *A is the* $\mathsf{Mat}^{\uparrow}$ *coordinate-transform row, used only for displayed finite matrices between certified coordinate presentations.*
6. *D records the finite dimension tag, coordinate index window, and endpoint-radius ledger for all inspected coordinates.*
7. *N records the norm, distance, projection, and affine-combination readback rows computed from R, M, V, I, A, D.*
8. *H records componentwise* hsame *transport for the scalar, metric, vector-space, inner-product, matrix, dimension, readback, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay of every Euclidean read through*

$$R, M, V, I, A, D \longmapsto N.$$

10. *P is the* Pkg *provenance over* $\mathsf{EuclideanSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{EuclideanSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, V, I, A, D, N, H, C, P, Q$. It has no coordinate for a hidden basis choice, host equality of coordinate functions, quotient vector spaces, spectral diagonalization, compactness, countable choice, excluded middle, Lean verification, or a standard bridge.

Theorem 7817.2 (EuclideanSpace NameCert obligations)**.** *For every accepted* $\mathsf{EuclideanSpace}^{\uparrow}$ *carrier*

$$E = (R, M, V, I, A, D, N, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{EuclideanSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real scalar seal, metric distance row, vector-space operation row, inner-product row, finite matrix row, dimension and coordinate-window ledger, norm and distance readback, transport, replay, provenance, and local naming rows;*

2. *scalar and coordinate stability, requiring every vector operation, inner-product read, and matrix transform to use the displayed* $\mathsf{Real}^{\uparrow}$ *scalar row and the same finite coordinate window D;*

3. *metric and inner-product compatibility, requiring the distance row M and norm readback N to be computed from the displayed* $\mathsf{InnerProduct}^{\uparrow}$ *row I rather than from an external metric predicate;*

4. *finite-coordinate exactness, requiring every projection, affine combination, and coordinate transform to pass through the displayed* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *rows before any downstream consumer reads it;*

5. *ledger refusal, requiring H, C, P, Q to transport, replay, source, and name only the displayed rows, with no hidden basis selector, quotient vector-space equality, host coordinate-function equality, spectral theorem, compactness, countable choice, excluded middle, Lean verification, or bridge credential.*

No EuclideanSpace NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7817.1. The accepted packet displays the real scalar seal, metric row, vector-space operations, inner-product row, finite matrix row, coordinate ledger, readback rows, and structural rows.

Scalar and coordinate stability are componentwise. The same R and D rows are read before vector addition, scalar multiplication, inner-product evaluation, distance computation, and matrix transport, so no hidden coordinate source is available.

Metric and inner-product compatibility is the role of M, I, N. The distance and norm rows are read from the displayed inner-product computation, not from a host metric predicate or an external norm theorem.

Finite-coordinate exactness follows through V, A, D, N. Projections, affine combinations, and coordinate transforms are inspected as finite rows before a consumer can read the resulting point, vector, distance, or norm.

Finally, H, C, P, Q only transport, replay, source, and name the displayed rows. Since the carrier contains no hidden basis selector, quotient equality, host coordinate-function equality, spectral theorem, compactness row, choice row, Lean verification, or bridge credential, the five listed rows exhaust the local NameCert surface. □

Theorem 7817.3 (EuclideanSpace finite-coordinate route)**.** *Every accepted* $\mathsf{EuclideanSpace}^{\uparrow}$ *carrier*

$$E = (R, M, V, I, A, D, N, H, C, P, Q)$$

exports geometric data only through the finite route

$$\mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{VecSpace}^{\uparrow}, \mathsf{InnerProduct}^{\uparrow}, \mathsf{Mat}^{\uparrow} \longmapsto D, N, H, C, P, Q.$$

A downstream consumer reading a Euclidean distance, norm, projection, affine combination, or coordinate transform must first read the scalar seal, metric row, vector-space row, inner-product row, matrix row, and finite coordinate ledger displayed by the packet. No hidden basis choice, quotient vector-space equality, host coordinate-function equality, compactness theorem, spectral theorem, countable-choice schedule, Lean verification, or bridge credential is exported.

Proof. Start from the carrier projection. The scalar, metric, vector-space, inner-product, matrix, and finite coordinate rows are the only upstream rows available to a Euclidean read.

The norm and distance readback is computed through I, M, N, while projection, affine-combination, and coordinate-transform reads pass through V, A, D, N. Each read is finite because the coordinate window D is displayed before N is consumed.

Transport, replay, provenance, and naming through H, C, P, Q preserve the same route. The excluded host equality, hidden basis, quotient, compactness, spectral, choice, verification, and bridge data are absent from the accepted carrier, so no consumer can export them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EuclideanSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EuclideanSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7817.1, Definition 7817.2, and Definition 7817.3 seed the finite Euclidean coordinate route over RealUp scalars, MetricSpace distance, VecSpace operations, InnerProduct norms, Matrix coordinate transforms, finite coordinate windows, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EuclideanSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary finite-dimensional normed-space equivalence, hidden basis selection, quotient linear algebra, spectral diagonalization, compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar and coordinate stability, metric and inner-product compatibility, finite-coordinate exactness, and provenance locality for BEDC.Derived.EuclideanSpaceUp.

Constructive story: A $\mathsf{EuclideanSpace}^{\uparrow}$ packet is generated from a RealUp scalar seal, MetricSpace distance row, VecSpace operation row, InnerProduct row, Matrix coordinate-transform row, finite dimension and coordinate-window ledger, norm and distance readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7818

A Concrete Naming Certificate for $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$ packet records the finite compact-net route by which pointwise bounded real-family rows become a displayed bound-modulus certificate. It sits after the compact finite-net source of chapter 129, the pointwise bounded family surface of chapter 6555, the finite supremum readback of chapter 1427, the windowed observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite compact-net bound-modulus certificate, not open-cover compactness, host suprema, countable choice, quotient family equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7818.1 (Compact pointwise bounded modulus carrier)**.** *A* $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, L, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the inspected probes, mesh radius, overlap witnesses, and compact-metric provenance.*
2. *F is the* $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$ *row, displaying the family entries and pointwise boundedness data read at the probes of K.*
3. *L is the finite* $\mathsf{LocatedSupremum}^{\uparrow}$ *comparison row used only over the displayed probe bounds.*
4. *S is the finite lower-bound and slack ledger converting the probe bounds into one shared bound-modulus request.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for family observations, compact-net probes, and tolerance requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite rational bounds and slack comparisons.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after K, F, L, S, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the compact source, pointwise family, finite supremum row, slack ledger, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of every bound-modulus read through*

$$K, F, L, S, W, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PointwiseBoundedRealFamily}$ $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, S, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, host suprema, countable-choice selections, quotient family equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 7818.2 (Compact pointwise bounded modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$ *carrier*

$$B = (K, F, L, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the compact finite-net source, pointwise bounded family, finite located-supremum row, slack ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *compact-net exposure, requiring K to display the inspected probes and overlap witnesses before any family bound is folded;*
3. *pointwise-bound exposure, requiring F and W to read family bounds at the displayed probes and never as quotient equality of families;*
4. *finite bound-modulus fold, requiring L, S, R to carry the finite maximum, lower-bound, slack, and rational readback comparisons used to produce the shared bound;*
5. *ledger refusal, requiring H, C, P, N to transport, replay, source, and name only the displayed rows, with no open-cover compactness, host supremum, countable-choice selector, quotient family equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No compact pointwise bounded modulus NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7818.1. The accepted packet displays exactly the compact source, pointwise family row, finite supremum comparison, slack ledger, stream windows, regular readback, Real seal, and structural rows.

Compact-net exposure is read through K. The inspected probes and overlaps are finite data, so the packet never replaces them by an arbitrary open cover or selected subcover.

Pointwise-bound exposure is read through F and W. Each family bound is attached to a displayed probe window, not to host equality of families or quotient real equality.

The finite bound-modulus fold is the shared role of L, S, R. The located-supremum row is used only over the finite displayed probe bounds, the slack ledger records the lower-bound comparisons, and the regular rational row records the finite readback.

Finally, H, C, P, N transport, replay, source, and name the same displayed packet. Since the carrier contains no open-cover compactness row, host supremum, countable-choice selector, quotient family equality, ambient completeness principle, verification row, or bridge credential, the five listed rows exhaust the local NameCert surface. □

Theorem 7818.3 (Compact pointwise bounded modulus finite-net bound)**.** *Every bound-modulus output exported by an accepted* $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$ *carrier*

$$B = (K, F, L, S, W, R, E, H, C, P, N)$$

factors through the finite route

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PointwiseBoundedRealFamily}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$

displayed by $K, F, L, S, W, R, E, H, C, P, N$*. The compact finite net is read first, pointwise family bounds are read only at its displayed probe windows, the finite located-supremum and slack rows compute the shared bound, and the regular rational readback precedes the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no open-cover compactness, host suprema, countable choice, quotient family equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier projection. The compact-facing data are exactly the finite-net row K, and the family-facing data are exactly F and the stream windows W.

Apply the NameCert obligations of Definition 7818.2. They force the compact probes to be displayed before the family bounds are folded and force every pointwise bound to remain attached to a visible probe window.

The shared bound is then computed only through L, S, R. The row L reads a finite located-supremum comparison over the displayed probe bounds, S records the lower-bound and slack budget, and R turns that finite ledger into regular rational readback.

Only after this route is visible can the Real-facing seal E be consumed. The structural rows H, C, P, N preserve the same finite route, so there is no coordinate through which open-cover compactness, a host supremum, countable choice, quotient family equality, or ambient Real completeness can enter the exported certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7818.1, Definition 7818.2, and Definition 7818.3 seed the finite compact-net route from pointwise bounded real-family rows to a displayed bound-modulus certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host suprema, countable choice, quotient family equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, pointwise-bound exposure, finite located-supremum fold, slack-ledger exactness, regular rational readback, Real seal locality, and ledger refusal for BEDC.Derived.CompactPointwiseBoundedModulusUp.

Constructive story: A $\mathsf{CompactPointwiseBoundedModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, PointwiseBoundedRealFamily rows, a finite LocatedSupremum comparison row, a slack ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7819

A Concrete Naming Certificate for $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$

Following Cousin and Bishop–Bridges finite-cover practice, the $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$ packet records the finite radius-choice row that turns gauge-cover cells into compact-uniform modulus handoff data. It sits between the Cousin cover source of chapter 3426, the finite gauge-cover rows of chapter 6708, the finite subcover ledger of chapter 6125, the compact metric source of chapter 129, and the uniform-modulus consumer of chapter 369. The packet names a finite gauge-radius choice ledger only; it is not an open-cover compactness theorem, a countable-choice selector, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 7819.1 (Finite cover gauge-radius choice carrier)**.** *A* $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, Q, F, L, S, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact finite-cover or finite-net window inspected by the radius-choice construction.*
2. *Q is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge source row, carrying positive local gauge radii and tagged cover-cell admission data.*
3. *F is the* $\mathsf{FiniteGaugeCover}^{\uparrow}$ *row, listing the finite gauge cells, tag witnesses, mesh-control data, and readback route through the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
4. *L is the* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *row, recording which displayed finite cells cover the compact source window and how their radius ledger is replayed for compact-uniform consumers.*
5. *S is the radius-choice row. For every cell named by F and accepted by L, it stores the chosen positive radius history as a displayed finite ledger entry, not as a countable selector or host minimum.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, consumed only after the compact source, Cousin gauge row, finite gauge-cover row, subcover ledger, and finite radius-choice row have all been exposed.*
7. *H records componentwise* hsame *transport for compact source, gauge source, finite gauge cover, subcover ledger, radius choice, uniform-modulus handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a compact finite-cover request through gauge inspection, finite cover admission, radius choice, and uniform-modulus export.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{FiniteGaugeCover}^{\uparrow}$, $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, Q, F, L, S, U, H, C, P, N$. It has no coordinate for host open-cover compactness, countable choice, a detached subcover selector, quotient cover equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7819.2 (Finite cover gauge-radius choice NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$ *carrier $G = (K, Q, F, L, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverGaugeRadiusChoic}}$ *surface consists exactly of compact-source exposure through K, Cousin gauge-source admission through Q, finite gauge-cover admission through F, finite subcover-ledger coverage through L, displayed radius-choice exactness through S,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation is discharged by host open-cover compactness, countable choice, a detached subcover selector, quotient cover equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7819.1. The tuple displays the compact source, Cousin gauge source, finite gauge-cover row, finite subcover ledger, radius-choice ledger, uniform-modulus handoff, transport, replay, provenance, and local naming rows.

The radius-choice row is downstream of the finite cover data. The rows Q and F expose the available positive gauge radii and finite cells, while L restricts the compact read to the displayed finite coverage ledger.

The row S then records only finitely many positive radius histories, one for each displayed ledger entry that survives the finite cover route. Because this row is part of the carrier, the construction uses no external selector and no host minimum over an open cover.

Finally U, H, C, P, N export, transport, replay, source, and name exactly this route. Since the excluded compactness, choice, quotient, and completeness objects are absent from the carrier, they cannot appear in the local NameCert obligations. □

Theorem 7819.3 (Finite cover gauge-radius uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$ *packet receives its radius data only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow},\ \mathsf{FiniteGaugeCover}^{\uparrow},\ \mathsf{FiniteSubcoverLedger}^{\uparrow},\ S,\ \mathsf{UniformModulus}^{\uparrow}.$$

The handoff therefore turns Cousin/gauge cover rows into compact-uniform modulus data by finite carrier projection. It exports no open-cover compactness proof, countable-choice selector, detached subcover choice, quotient cover equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the compact and gauge sources. The row K exposes the compact metric window, and Q exposes the Cousin gauge-radius data attached to that window.

The finite cover route is then read through F and L. The row F lists the admitted gauge cells and tag witnesses, while L records the finite coverage ledger used by compact-uniform consumers.

The radius-choice ledger S is therefore a finite row over displayed cells. The $\mathsf{UniformModulus}^{\uparrow}$ handoff U can consume only this finite row together with the visible source and cover rows.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Thus the consumer receives compact-uniform modulus handoff data without receiving any excluded open-cover, choice, quotient, or completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}$ packet is generated from CompactMetric source windows, CousinCover gauge rows, FiniteGaugeCover cell rows, a FiniteSubcoverLedger coverage row, a displayed finite radius-choice ledger, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7819.1, Definition 7819.2, and Definition 7819.3 seed the finite gauge-radius choice route from Cousin and finite gauge-cover rows to compact-uniform modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverGaugeRadiusChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, countable choice, detached subcover selection, quotient cover equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, Cousin gauge-source admission, finite gauge-cover admission, subcover-ledger coverage, radius-choice exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverGaugeRadiusChoiceUp.

Chapter 7820

A Concrete Naming Certificate for $\mathsf{FrechetUrysohnMetric}^{\uparrow}$

Following Frechet and Urysohn, the $\mathsf{FrechetUrysohnMetric}^{\uparrow}$ packet records the metric sequence-closure readback route as finite BEDC NameCert data. It sits over the metric carrier of chapter 57, the sequential metric packet of chapter 7680, the stream naming discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the traditional sequence-closure interface without importing a host sequence selector, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7820.1 (Frechet-Urysohn metric carrier)**.** *A* $\mathsf{FrechetUrysohnMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, M, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the point carrier inherited from the underlying* $\mathsf{Met}^{\uparrow}$ *surface.*
2. *M is the metric distance ledger, including only displayed tolerance comparisons and triangle rows used by the sequential closure test.*
3. *Q is the sequence-closure query row, recording the finite neighbourhood windows whose eventual membership is being read.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the inspected sequence positions and no external sequence selector.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational distance and tolerance observations requested by M, Q, S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback is displayed.*
7. *H records componentwise* hsame *transport for point, metric, query, stream, regular rational, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from sequence-closure queries through M, Q, S, R, E, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohnMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SequentialMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohnMetric}^{\uparrow}}$ *row.*

The classifier of an accepted carrier compares packets only through these rows. It has no coordinate for a host sequence selector, quotient equality of streams, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7820.2 (Frechet-Urysohn metric NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohnMetric}^{\uparrow}$ *carrier* $F = (X, M, Q, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohnMetric}^{\uparrow}}$ *surface consists exactly of metric carrier exposure, sequential query exposure, finite StreamName schedule readback, RegSeqRat tolerance readback, terminal Real seal discipline, componentwise transport, displayed continuation replay, package provenance, and local naming. Every sequence-closure consumer must display the metric row of chapter 57, the sequential metric row of chapter 7680, the StreamName row of chapter 126, the RegSeqRat row of chapter 301, and the Real seal row of chapter 38 before exporting a closure read.*

Proof. Project the carrier of Definition 7820.1. The rows X, M, Q give the point, metric, and sequence-closure query surface, while S, R, E are the only rows that can expose sequence-indexed observations to downstream Real-facing consumers.

The sibling chapters determine the read order. The metric carrier supplies the distance ledger, the sequential metric packet supplies the sequence-indexed observation discipline, StreamName supplies the finite schedule, RegSeqRat supplies the regular rational readback, and Real supplies only the terminal seal.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no row contains a host sequence selector, quotient stream equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local obligation surface. □

Theorem 7820.3 (Frechet-Urysohn metric sequence readback)**.** *Every accepted* $\mathsf{FrechetUrysohnMetric}^{\uparrow}$ *packet that witnesses a point in the sequence closure of a displayed metric set exports its Real-facing closure value only through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{SequentialMetricSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The readback may consume displayed metric neighbourhood rows and displayed sequence windows, but it cannot replace them by a host sequence selector, quotient stream equality, a selected representative, or an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 7820.2. It places every accepted closure query behind the carrier rows $X, M, Q, S, R, E, H, C, P, N$.

Use the sequential metric packet of chapter 7680 to read sequence-indexed metric observations. Its consumer route already requires a StreamName handoff before any RegSeqRat or Real-facing read can be consumed.

Now route the metric distance observations through chapter 57, chapter 126, chapter 301, and chapter 38. The result is a displayed schedule, a regular rational tolerance read, and a terminal Real seal.

No excluded object is introduced by the support rows. Thus the exported sequence-closure value remains a finite BEDC readback through the displayed metric, sequential, stream, regular rational, and Real rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohnMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrechetUrysohnMetric}^{\uparrow}$ packet is generated from MetricSpaceUp carrier rows, SequentialMetricSpaceUp sequence-query rows, StreamNameUp finite schedules, RegSeqRatUp tolerance readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FrechetUrysohnMetric}^{\uparrow}}$, seed)

Scope closed: Definition 7820.1, Definition 7820.2, and Definition 7820.3 seed the metric sequence-closure readback route through StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FrechetUrysohnMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove host sequence selection, quotient stream equality, selected representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric query exactness, StreamName schedule readback, RegSeqRat tolerance readback, Real seal discipline, transport, replay, provenance, and nonescape for BEDC.Derived.FrechetUrysohnMetricUp.

Chapter 7821

A Concrete Naming Certificate for $\mathsf{CompactMetricModulusRealizer}^{\uparrow}$

Following Heine–Cantor and Bishop's finite-net proof of uniform continuity on compact metric spaces, the $\mathsf{CompactMetricModulusRealizer}^{\uparrow}$ packet records the finite data used by the compact-uniform-continuity theorem of chapter 9237. It sits over the compact metric source of chapter 129, the continuous-map pointwise modulus rows of chapter 130, the metric carrier rows of chapter 57, the $\mathsf{Met}^{\uparrow}$ distance classifiers of chapter 57, the rational and real readback route of chapter 37 and chapter 38, and the tolerance-only handoff of chapter 369. The packet names the finite-net fold, not an external minimum selector or a host compactness theorem.

Definition 7821.1 (Compact metric modulus realizer carrier)**.** *A* $\mathsf{CompactMetricModulusRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, B, L, M, T, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including the displayed center bundle and located half-radius cover witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, carrying the pointwise-modulus rows attached to the centers named by K.*
3. *B is the finite* $\mathsf{ProbeBundle}$ *spine consumed by the lower-bound fold.*
4. *L is the rational lower-bound fold over the positive center radii, read through* $\mathsf{Rat}^{\uparrow}$ *order data.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *triangle ledger joining source-point, center, and target-image estimates.*
6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for halving, positivity, and lower-bound transport.*
7. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row for the rational and Real comparisons already displayed by L, M, T.*
8. *U is the resulting* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row.*
9. *H records componentwise* hsame *transport for compact metric, continuous map, bundle, fold, metric, tolerance, seal, handoff, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a requested tolerance through $K, F, B, L, M, T, E, H, P, N$ to U.*

11. *P is the* Pkg *provenance over* CompactMetricModulusRealizer$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CompactMetricModulusRealizer}^{\uparrow}}$ *row.*

The classifier of an accepted carrier compares packets only through these finite rows. It has no coordinate for an external compactness theorem, an open-cover choice functional, a host minimum operator, quotient equality of functions, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 7821.2 (Compact metric modulus realizer finite-net fold)**.** *For every accepted* CompactMetricModulusRea *carrier $R = (K, F, B, L, M, T, E, U, H, C, P, N)$, the exported* UniformModulus$^{\uparrow}$ *row U is consumed only after the finite-net row K, pointwise-modulus row F, ProbeBundle spine B, rational lower-bound fold L, metric-triangle ledger M, dyadic tolerance row T, and RegSeqRat-to-Real seal row E are displayed. This is the NameCert realization of the finite fold in Definition 9237.5 and the uniform-modulus theorem of Definition 9237.6; it exports no external minimum selector, host compactness theorem, quotient function equality, selected center family, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7821.1. The compact metric source K supplies the finite center bundle and located coverage witnesses, while F attaches the pointwise continuous-map radius rows to those displayed centers.

Read the finite fold through B, L, T. The ProbeBundle spine B is finite, the row L performs the rational lower-bound fold over the displayed positive radii, and the dyadic tolerance row T transports the requested tolerance through halving and lower-bound comparisons.

Use the metric ledger M before any Real-facing read. Its source and target triangle rows connect the selected displayed center to the two inspected points and their images, so the regular rational and Real seal row E can only seal comparisons already witnessed by K, F, B, L, M, T.

The handoff row U is therefore a tolerance-only UniformModulus$^{\uparrow}$ output. The support rows H, C, P, N transport, replay, source, and name the same finite route, and no excluded compactness, minimum, quotient, selection, or ambient-completeness coordinate is present. □

Closure status for NameCert$_{\text{CompactMetricModulusRealizer}^{\uparrow}}$.

Constructive story: A CompactMetricModulusRealizer$^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, ProbeBundle finite spines, RatUp lower-bound folds, MetricSpaceUp triangle ledgers, DyadicRatCoreUp tolerance rows, RegSeqRatUp and RealUp seals, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactMetricModulusRealizer}^{\uparrow}}$, seed)

Scope closed: Definition 7821.1 and Definition 7821.2 seed the compact metric finite-net modulus realizer route from pointwise moduli to a tolerance-only UniformModulus handoff.

Formal verification: FormalStatus(NameCert$_{\text{CompactMetricModulusRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove external compactness, host finite minima, quotient equality of functions, selected center families, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, pointwise-modulus row coverage, ProbeBundle fold exactness, metric triangle replay, dyadic tolerance transport, RegSeqRat and Real seal discipline, UniformModulus handoff, and nonescape for BEDC.Derived.CompactMetricModulusRealizerUp.

Chapter 7822

A Concrete Naming Certificate for $\mathsf{DyadicBracketSchedule}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real approximations, the $\mathsf{DyadicBracketSchedule}^{\uparrow}$ packet records the finite inspection schedule by which dyadic brackets feed regular rational readback, stream windows, and the terminal real-name seal. It sits after the dyadic arithmetic surface of chapter 291, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet also consumes the lower-and-upper endpoint surface of chapter 7222. It is a finite constructive-analysis schedule, not a quotient real, a selected completed limit, a countable-choice selector, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7822.1 (Dyadic bracket schedule carrier)**.** *A* $\mathsf{DyadicBracketSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row inspected at the requested scale.*
2. *U is the upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row inspected at the same scale.*
3. *I is the finite inspection schedule row, recording the order in which bracket widths, endpoint comparisons, and tolerance checks are replayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the inspection schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the schedule has exposed L, U, I, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback row R has been displayed.*
7. *H records componentwise* hsame *transport for endpoint, schedule, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bracket request through L, U, I, W to R and then to E.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicBracketSchedule}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicBracketing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBracketSchedule}^{\uparrow}}$ *row.*

The classifier compares accepted schedules only through $L, U, I, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules outside the displayed finite row, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7822.2 (Dyadic bracket schedule NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBracketSchedule}^{\uparrow}$ *carrier $S = (L, U, I, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicBracketSchedule}^{\uparrow}}$ *surface consists exactly of lower endpoint exposure, upper endpoint exposure, finite inspection-schedule replay, StreamName window coverage, RegSeqRat readback, terminal Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every Real or Completion consumer must read those rows before treating E as a real-name seal.*

Proof. Project the carrier of Definition 7822.1. The endpoint rows L, U are the only dyadic bracket endpoints available to the local classifier, and the inspection row I is the only schedule row.

Read the finite window route next. The row W opens the $\mathsf{StreamName}^{\uparrow}$ window requested by I, and R is the regular rational readback from the displayed endpoint and window data.

The Real-facing output is downstream of that readback. The seal row E appears only after R, and the structural rows H, C, P, N transport, replay, source, and name the same finite route.

Since the carrier has no coordinate for quotient stream equality, selected limits, hidden choice schedules, host real equality, ambient completeness, or bridge certificates, the displayed rows exhaust the local NameCert obligation surface. □

Theorem 7822.3 (Dyadic bracket schedule regular-Cauchy handoff)**.** *Let $S = (L, U, I, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicBracketSchedule}^{\uparrow}$ *carrier. Every regular-Cauchy or Completion-facing consumer of S factors through the finite route*

$$L, U, I, W \longmapsto R \longmapsto E$$

with H, C, P, N preserving the same route under transport, replay, provenance, and local naming. Thus the consumer receives dyadic lower and upper bracket endpoints, the inspection schedule, a finite StreamName window, RegSeqRat readback, and the terminal Real seal, but receives no quotient filter, selected limit, countable-choice selector, host equality, ambient Real completeness, or bridge certificate.

Proof. Start with the NameCert obligations of Definition 7822.2. They require endpoint exposure, schedule replay, window coverage, regular rational readback, and Real-seal handoff as displayed rows of the same packet.

The regular-Cauchy handoff is the row R. It is read only after L, U, I, W have appeared, so it cannot be detached from the displayed bracket schedule or replaced by a selected completed limit.

The terminal seal E is downstream of R. Componentwise transport through H and replay through C preserve the same order, while P and N record provenance and local naming without adding any alternate Completion-facing coordinate. □

Theorem 7822.4 (Dyadic bracket schedule Real-seal nonescape)**.** *For every accepted* $\mathsf{DyadicBracketSchedule}^{\uparrow}$ *carrier $S = (L, U, I, W, R, E, H, C, P, N)$, no Real-facing read can reach the seal row E without first displaying the dyadic lower endpoint L, dyadic upper endpoint U, finite inspection schedule I, StreamName window W, and RegSeqRat readback R. In particular, the accepted schedule cannot export quotient real equality, selected limits, countable-choice schedules, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Inspect the carrier definition. The only Real-facing row is E, and it is placed after the displayed dyadic, schedule, window, and regular-readback rows.

Apply Definition 7822.2. The local naming surface requires endpoint exposure, schedule replay, StreamName window coverage, and RegSeqRat readback before the Real-seal handoff is admitted.

Use Definition 7822.3 for consumers. Every regular-Cauchy or Completion-facing request follows the route $L, U, I, W \longmapsto R \longmapsto E$, and the structural rows preserve that route rather than adding a second path to $\mathsf{Real}^{\uparrow}$.

The excluded quotient, selected-limit, choice, host-equality, ambient-completeness, and bridge objects are absent from the carrier. Hence any read using them has left the accepted $\mathsf{DyadicBracketSchedule}^{\uparrow}$ packet instead of reading its Real-seal surface. □

Falsifiable prediction: Every accepted $\mathsf{DyadicBracketSchedule}^{\uparrow}$ Real-facing read must display lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, the finite inspection schedule, a $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicbracketing, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicBracketSchedule}^{\uparrow}$ is not bijective to any listed sibling because it binds bracket endpoints, one inspection schedule, one stream window, regular rational readback, Real seal, transport, replay, provenance, and local naming as one schedule certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBracketSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBracketSchedule}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, a finite inspection schedule, a StreamName window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBracketSchedule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7822.1, Definition 7822.2, Definition 7822.3, and Definition 7822.4 seed the finite dyadic bracket schedule route feeding regular-Cauchy, Real, and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBracketSchedule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected completed limits, hidden choice schedules, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, inspection-schedule replay, StreamName window coverage, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicBracketSche

Chapter 7823

A Concrete Naming Certificate for $\mathsf{RiemannCauchyRealName}^{\uparrow}$

Following Riemann and Bishop–Bridges, the $\mathsf{RiemannCauchyRealName}^{\uparrow}$ packet records a real name as a finite BEDC certificate around rational fundamental sequences rather than as a host real. It sits between the finite stream discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational sequence route of chapter 301, the public real-name boundary of chapter 38, and the regularizing handoff of chapter 7319. The packet exposes the traditional Cauchy fundamental-sequence surface only through finite windows, rational readback, dyadic moduli, transport, replay, provenance, and a local name.

Definition 7823.1 (Riemann–Cauchy real-name carrier)**.** *A* $\mathsf{RiemannCauchyRealName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (Q, S, W, D, R, U, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational source row for the fundamental sequence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing finite sequence windows.*
3. *W is the Cauchy-window schedule row for requested rational tolerances.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger bounding the window comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback obtained from S, W, D.*
6. *U is the regularizer row that presents the same name at the* $\mathsf{Real}^{\uparrow}$ *boundary.*
7. *H records componentwise* hsame *transport for the source, stream, window, dyadic, readback, regularizer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route $Q, S, W, D \longmapsto R \longmapsto U$.*
9. *P is the* Pkg *provenance over* $\mathsf{RiemannCauchyRealName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannCauchyRealName}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed host real, quotient stream equality, a selected limit, trichotomy, countable choice, or an ambient completeness theorem.

Theorem 7823.2 (Riemann–Cauchy real-name NameCert obligations)**.** *For every accepted* RiemannCauchyRealNa *carrier* $X = (Q, S, W, D, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RiemannCauchyRealName}^{\uparrow}}$ *surface consists exactly of carrier admission for* X*, rational fundamental-sequence exposure through* Q*, finite stream windows through* S*, Cauchy-window scheduling through* W*, dyadic tolerance accounting through* D*, regular rational readback through* R*, regularized* $\mathsf{Real}^{\uparrow}$ *boundary exposure through* U*, component-wise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 7823.1. Its first four rows expose only rational source data, finite stream windows, requested tolerances, and dyadic bounds.

The readback row R is therefore reached through the same finite route used by the $\mathsf{RegSeqRat}^{\uparrow}$ chapter, and the regularizer row U is the only public route from that readback to the $\mathsf{Real}^{\uparrow}$ boundary.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no further coordinate is present, the local certificate cannot ask for a selected host limit, quotient stream equality, countable choice, trichotomy, or ambient completeness. □

Theorem 7823.3 (Riemann–Cauchy real-name RegSeqRat factorization)**.** *Every accepted* RiemannCauchyRealNam *consumer that reads the public real-name boundary must first factor through the finite route*

$$Q, S, W, D \longmapsto R \longmapsto U$$

where R *is the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* U *is the regularized* $\mathsf{Real}^{\uparrow}$ *boundary row. The consumer receives no host real equality, quotient stream equality, selected limit, detached completion route, countable-choice schedule, or ambient completeness principle.*

Proof. Begin with the NameCert obligations of Definition 7823.2. They order the rational source, finite windows, Cauchy-window schedule, and dyadic tolerance ledger before any regular readback is available.

The row R is the unique regular rational readback inside the packet. Any consumer of the $\mathsf{Real}^{\uparrow}$ boundary must therefore pass through the $\mathsf{RegSeqRat}^{\uparrow}$ route before it can read the regularizer row U.

Finally inspect the refused coordinates. The carrier does not contain a quotient stream, selected limit, host real, choice schedule, detached completion route, or ambient completeness row, so none can replace the displayed Q, S, W, D, R, U factorization. □

Falsifiable prediction: A consumer that claims a $\mathsf{RiemannCauchyRealName}^{\uparrow}$ real-name boundary while omitting the rational source, finite stream window, Cauchy-window schedule, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, regularizer, hsame, Cont, Pkg, or local NameCert rows refutes Definition 7823.1.

Independence witness: streamname, dyadicratcore, regseqrat, real, regular-sequence-regularizer: $\mathsf{RiemannCauchyRealName}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a rational fundamental-sequence source, finite stream windows, Cauchy-window scheduling, dyadic tolerance accounting, regular readback, and a real-boundary regularizer. StreamNameUp lacks the tolerance ledger; DyadicRatCoreUp lacks the sequence route; RegSeqRatUp lacks the raw fundamental-sequence source; RealUp lacks the displayed Cauchy-window schedule; the regularizer chapter supplies a handoff without this full source packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannCauchyRealName}^{\uparrow}}$**.**

Constructive story: A RiemannCauchyRealName$^{\uparrow}$ packet is generated from a rational source row, finite StreamName windows, a Cauchy-window schedule, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real-boundary regularizer, componentwise transport, displayed replay, provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{RiemannCauchyRealName}^{\uparrow}}$, seed)

Scope closed: Definition 7823.1, Definition 7823.2, and Definition 7823.3 seed the finite Riemann–Cauchy real-name route over rational source rows, finite windows, dyadic tolerances, regular readback, real-boundary regularization, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{RiemannCauchyRealName}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real construction, selected limits, countable choice, trichotomy, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window scheduling, dyadic tolerance accounting, RegSeqRat factorization, regularizer routing, transport, replay, provenance, and local naming for BEDC.Derived.RiemannCauchyReal

Chapter 7824

A Concrete Naming Certificate for $\mathsf{SteinerTree}^{\uparrow}$

Following Gilbert and Pollak and standard finite network optimization, the $\mathsf{SteinerTree}^{\uparrow}$ packet names the finite certificate that connects a displayed terminal family inside a weighted graph by a tree-shaped subnetwork. It sits over the graph carrier of chapter 118, the acyclic connected carrier of chapter 119, the spanning-tree surface of chapter 237, the shortest-path ledger of chapter 236, and the metric-graph route of chapter 6788. The packet is a finite optimization certificate; it does not import ambient network-flow duality, a host minimum oracle, graph isomorphism, an infinite network limit, or quotient equality of vertex sets.

Definition 7824.1 (Steiner tree carrier)**.** *A* $\mathsf{SteinerTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, W, T, R, E, A, L, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *row naming the finite vertex and edge incidence data.*
2. *W is the finite edge-weight ledger over G, read through the same weighted path surface used by* $\mathsf{ShortestPath}^{\uparrow}$ *and* $\mathsf{MetricGraph}^{\uparrow}$*.*
3. *T is the terminal-family row, a finite list or finite subset of vertices of G that must be connected.*
4. *R is the selected root or anchor row used only to orient the finite certificate and not to choose a canonical optimum.*
5. *E is the candidate edge-subset row, naming the displayed subgraph used by the certificate.*
6. *A is the admitted auxiliary-vertex row, naming nonterminal vertices of G that may appear in the candidate subgraph.*
7. *L is the length ledger, summing the displayed edge weights of E.*
8. *B is the bound row, comparing L with the finite competitor ledgers that the packet chooses to expose.*
9. *H records componentwise* hsame *transport for graph, weight, terminal, anchor, candidate-edge, auxiliary-vertex, length, bound, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from graph incidence and terminal rows through the candidate tree route, length ledger, and bound row.*

11. *P is the* Pkg *provenance over* $\mathsf{SteinerTree}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{ShortestPath}^{\uparrow}$, $\mathsf{MetricGraph}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SteinerTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It carries no coordinate for a hidden global optimizer, unlisted terminals, quotient graph isomorphism, network-flow duality, infinite graph limits, or a host enumeration of all candidate subnetworks.

Theorem 7824.2 (Steiner tree NameCert obligations)**.** *For every accepted* $\mathsf{SteinerTree}^{\uparrow}$ *carrier* $S = (G, W, T, R, E, A, L, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SteinerTree}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed graph row, edge-weight ledger, terminal-family row, anchor row, candidate edge-subset row, auxiliary-vertex row, length ledger, bound row, componentwise transport, displayed replay, package provenance, and local naming. Every public read must keep the* $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{ShortestPath}^{\uparrow}$, *and* $\mathsf{MetricGraph}^{\uparrow}$ *source rows visible before it cites the Steiner certificate.*

Proof. Project the carrier of Definition 7824.1. The graph, weights, terminal family, candidate edge set, auxiliary vertices, length, and bound rows are the only optimization-facing coordinates.

Read the graph-theoretic rows first. The $\mathsf{Graph}^{\uparrow}$ row supplies finite incidence, the $\mathsf{Tree}^{\uparrow}$ row supplies connected acyclic recognition for the candidate subgraph, and the $\mathsf{SpanningTree}^{\uparrow}$ row supplies the spanning comparison surface when all graph vertices are required rather than only the displayed terminal family.

Now read the metric rows. $\mathsf{ShortestPath}^{\uparrow}$ and $\mathsf{MetricGraph}^{\uparrow}$ provide the finite path and weight ledgers used by L and B. The structural rows H, C, P, N transport, replay, source, and name the same displayed packet, so no hidden optimizer or external graph enumeration is available. □

Theorem 7824.3 (Steiner tree finite-terminal nonescape)**.** *Every accepted* $\mathsf{SteinerTree}^{\uparrow}$ *consumer must display the terminal-family row T and the candidate tree route through E, A before reading the length and bound rows L, B. A consumer that adds a terminal, removes a required terminal, or reads a bound for a different terminal family has left the accepted carrier.*

Proof. Start from the carrier inventory. The terminal-family row T is separate from the graph row G, the candidate edge-subset row E, and the auxiliary-vertex row A.

The tree certificate is downstream of the displayed terminal family. The candidate subgraph must connect every vertex named by T, while auxiliary vertices may appear only through the admitted row A and the same graph row G.

Finally read L and B. These rows sum and compare only the candidate edge rows visible through E for the terminal family visible through T. Transport and replay preserve those rows componentwise, so a different terminal family is not a read of the same $\mathsf{SteinerTree}^{\uparrow}$ packet. □

Theorem 7824.4 (Steiner tree network-flow boundary)**.** *The* $\mathsf{SteinerTree}^{\uparrow}$ *packet may be consumed by network or flow-facing arguments only through the displayed finite graph, path, tree, weight, length, and bound rows*

$$\mathsf{Graph}^{\uparrow},\ \mathsf{Tree}^{\uparrow},\ \mathsf{SpanningTree}^{\uparrow},\ \mathsf{ShortestPath}^{\uparrow},\ \mathsf{MetricGraph}^{\uparrow},\ G, W, T, E, A, L, B, H, C, P, N.$$

It does not export a max-flow min-cut theorem, a linear-programming dual, a host optimum oracle, quotient equality of graphs, or an infinite-network compactness principle.

Proof. Project the displayed route. The graph and metric rows supply finite incidence, edge weights, path ledgers, and metric reads; the tree rows supply connected acyclic recognition for the selected subnetwork.

The bound row B is a finite comparison row inside the packet. It may be read as a certificate boundary for exposed competitors, but it is not a duality theorem and not an external optimization oracle.

Since the carrier has no coordinate for max-flow min-cut data, linear programming dual variables, quotient graph equality, or infinite-network compactness, any consumer claiming those endpoints has left the displayed NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{SteinerTree}^{\uparrow}$ read is refuted if it claims a Steiner certificate without displaying the graph row, terminal-family row, candidate tree route, edge-weight ledger, finite length comparison, transport, replay, provenance, and local naming rows.

Independence witness: graph, tree, spanningtree, shortestpath, metric-graph: $\mathsf{Graph}^{\uparrow}$ supplies incidence without terminal optimization; $\mathsf{Tree}^{\uparrow}$ supplies connected acyclic recognition without weighted terminal bounds; $\mathsf{SpanningTree}^{\uparrow}$ supplies full-vertex spanning witnesses without auxiliary terminal selection; $\mathsf{ShortestPath}^{\uparrow}$ supplies path comparisons without a tree-shaped terminal subnetwork; and $\mathsf{MetricGraph}^{\uparrow}$ supplies graph metric reads without a Steiner certificate. None is bijective to $\mathsf{SteinerTree}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{SteinerTree}^{\uparrow}}$.

Constructive story: A $\mathsf{SteinerTree}^{\uparrow}$ packet is generated from a finite GraphUp row, an edge-weight ledger, a finite terminal-family row, a candidate tree-shaped edge set, admitted auxiliary vertices, a length ledger, a bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SteinerTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7824.1, Definition 7824.2, Definition 7824.3, and Definition 7824.4 seed the finite Steiner-tree certificate over GraphUp, TreeUp, SpanningTreeUp, ShortestPathUp, MetricGraphUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SteinerTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, max-flow min-cut duality, host optimality, quotient graph equality, infinite network compactness, or an exhaustive external enumeration of candidate subnetworks.

Upgrade path: Obligation closure requires checked carrier admission, terminal-family stability, candidate-tree validity, auxiliary-vertex boundary, length-ledger exactness, bound-row replay, provenance, and local naming for BEDC.Derived.SteinerTreeUp.

Chapter 7825

A Concrete Naming Certificate for MetaCICSubjectReductionResidualGap$^{\uparrow}$

The MetaCICSubjectReductionResidualGap$^{\uparrow}$ packet realizes the dependent-codomain obstruction of chapter 9377 as finite residual-gap NameCert data. It sits beside the subject-reduction discharge socket of chapter 719, the subject-reduction frontier of chapter 730, the app-argument stability ledger of chapter 6474, the normalization frontier of chapter 892, and the critical-path packet of chapter 765. The packet names the residual budget that remains after the checked independent-codomain application case; it does not prove closed subject reduction, discharge the four preservation rows, identify evaluator equality, quotient terms, or kernel consistency.

Definition 7825.1 (MetaCIC subject-reduction residual-gap carrier)**.** *A* MetaCICSubjectReductionResidualGap$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, D, A, C, O, T, E, H, Q, P, N)$$

with the following displayed rows:

1. *S is the subject-reduction socket row naming the four discharge obligations that a consumer must keep visible.*
2. *D is the dependent-codomain residual row, recording the application case in which the codomain may observe the reducing argument.*
3. *A is the app-argument stability row, consumed from the displayed argument-type ledger.*
4. *C is the candidate-normalization row, recording the normalization frontier data that may be inspected without closing subject reduction.*
5. *O is the obstruction-retention row, naming the checked dependent-codomain obstruction and the endpoints it refuses.*
6. *T is the transport row for socket, residual, app-argument, candidate-normalization, obstruction, replay, provenance, and local naming data.*
7. *E is the replay row, routing a subject-reduction request through S, D, A, C, O, T before any downstream normalization or critical-path consumer may inspect it.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *Q records displayed* Cont *replay for the same route.*

10. *P is the* Pkg *provenance over* MetaCICSubjectReductionResidualGap$^{\uparrow}$, SubjectReduction$^{\uparrow}$, SubjectReductionDischa MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}}$ *row.*

The carrier has no coordinate for a closed subject-reduction theorem, unrestricted substitution compatibility, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Theorem 7825.2 (MetaCIC subject-reduction residual-gap obligations)**.** *For every accepted* MetaCICSubjectReduc *carrier $R = (S, D, A, C, O, T, E, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}}$ *surface consists exactly of carrier admission, socket visibility, dependent-codomain residual display, app-argument stability handoff, candidate-normalization visibility, obstruction retention, transport, replay, provenance, and local naming. Every consumer must expose the rows $S, D, A, C, O, T, E, H, Q, P, N$ before citing the residual-gap packet.*

Proof. Project the carrier of Definition 7825.1. The subject-reduction socket, dependent-codomain residual, app-argument stability, candidate-normalization, and obstruction rows are distinct visible coordinates.

Read the socket row first. It keeps the four discharge obligations in view and prevents the residual-gap packet from standing in for any missing preservation row.

Next read D, A, C, O. The dependent-codomain row records the residual case, the app-argument row keeps the argument-type ledger attached, the candidate-normalization row records the normalization-facing budget, and the obstruction row keeps the checked counterexample boundary visible.

Finally T, E, H, Q, P, N transport, replay, source, and name the same packet. Since none of those rows contains a closed subject-reduction theorem, evaluator equality, quotient syntax equality, or kernel-consistency coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7825.3 (MetaCIC subject-reduction residual-gap nonescape)**.** *Every subject-reduction, normalization, or critical-path consumer of an accepted* MetaCICSubjectReductionResidualGap$^{\uparrow}$ *packet must keep the finite route*

$$S, D, A, C, O, T, E, H, Q, P, N$$

visible. The route may expose a residual-gap budget for the dependent-codomain application case, but it cannot be reread as closed subject reduction, a discharge of the four preservation rows, unrestricted substitution compatibility, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Proof. Start with the obligation surface above. A consumer reaches the residual-gap row only after the subject-reduction socket S and dependent-codomain row D are displayed.

The app-argument and candidate-normalization rows then remain boundary data. They allow the consumer to inspect argument-type stability and normalization-facing candidates, while the obstruction row O records that the dependent-codomain obstruction has not been discharged.

Transport and replay preserve the same route. The rows T, E, H, Q, P, N attach the finite packet to its provenance and local name; they do not add any of the refused subject-reduction, substitution, equality, normalization, or kernel-consistency endpoints. □

Falsifiable prediction: An accepted MetaCICSubjectReductionResidualGap$^{\uparrow}$ read is refuted if it claims a subject-reduction residual without displaying the socket, dependent-codomain residual, app-argument stability, candidate-normalization, obstruction-retention, transport, replay, provenance, and local naming rows.

Independence witness: subject-reduction-discharge-socket, metacic-subject-reduction-frontier, metacic-app-argument-stability-ledger, metacicnormalizationfrontier, metaciccriticalpath: none is bijective to $\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}$ because this packet jointly carries residual-gap, socket, app-argument, candidate-normalization, obstruction, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}$ packet is generated from subject-reduction socket rows, dependent-codomain residual rows, app-argument stability rows, candidate-normalization rows, obstruction-retention rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7825.1, Definition 7825.2, and Definition 7825.3 seed the finite residual-gap route over SubjectReductionUp, SubjectReductionDischargeSocketUp, MetaCICNormalizationFrontierUp, MetaCICCriticalPathUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove closed subject reduction, discharge the four preservation obligations, close unrestricted substitution compatibility, identify evaluator equality, quotient terms, global normalization, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, socket visibility, dependent-codomain residual display, app-argument stability handoff, candidate-normalization visibility, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSubjectReductionResidualGapUp.

Chapter 7826

A Concrete Naming Certificate for $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$

Following Moore–Osgood and Bishop–Bridges constructive analysis, the $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ packet records the finite window at which an iterated-limit handoff can be inspected before any consumer invokes the broader interchange certificate of chapter 1686. It reads the two-parameter source through the $\mathsf{CauchyDoubleSequence}^{\uparrow}$ surface of chapter 460, keeps every row inside the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observation discipline, and reaches the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 only after the displayed handoff rows have been exposed.

Definition 7826.1 (Moore-Osgood finite-window carrier)**.** *A* $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, F, S, U, Q, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite two-parameter* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *window, listing the row and column indices, the inspected rectangle, and the tolerance request opened by the handoff.*
2. *F is the first-direction* $\mathsf{StreamName}^{\uparrow}$ *handoff row read from the windows in W.*
3. *S is the second-direction* $\mathsf{StreamName}^{\uparrow}$ *handoff row read from the same finite window family.*
4. *U is the finite uniform-tail row binding the active tolerance to the side of the window that remains fixed while the other side is enlarged.*
5. *Q is the finite schedule row that records which rectangle enlargements are allowed before the handoff reaches regular rational readback.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the two displayed handoff streams as regular rational data.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for the tolerance budget, tail bound, rectangle enlargement, and accumulated handoff error.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after W, F, S, U, Q, R, D have all been displayed.*
9. *H records componentwise* hsame *transport for the window, two stream handoffs, uniform-tail row, schedule, regular readback, dyadic ledger, real seal, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from finite rectangle inspection to stream handoff rows, from the uniform-tail and schedule rows to the regular readback row, and from dyadic accounting to the real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$, $\mathsf{CauchyDoubleSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, F, S, U, Q, R, D, E, H, C, P, N$. It has no coordinate for a completed double limit, quotient equality of arrays, countable choice, selected limiting representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7826.2 (Moore-Osgood finite-window NameCert obligations)**.** *For every accepted* MooreOsgoodFiniteWi *carrier*

$$M = (W, F, S, U, Q, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *window, two* $\mathsf{StreamName}^{\uparrow}$ *handoff rows, uniform-tail row, finite schedule, regular readback, dyadic ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *window stability, requiring every handoff consumer to read the same finite rectangle row W before either stream handoff F or S is exposed;*

3. *tail-schedule exactness, requiring U and Q to bind the active tolerance and allowed rectangle enlargements before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R can be read;*

4. *dyadic accounting, requiring D to ledger the tolerance, tail, schedule, and handoff-error rows before the* $\mathsf{Real}^{\uparrow}$ *seal E is visible;*

5. *nonescape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows, with no host double-limit theorem, quotient array equality, countable choice, selected limiting representative, or ambient Real completeness.*

No finite-window Moore-Osgood NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 7826.1. Carrier admission is the displayed inventory $W, F, S, U, Q, R, D, E, H, C, P, N$, and every row is finite BHist data.

Window stability follows from the placement of F and S after the common two-parameter window W. A consumer can inspect either handoff direction only through the same finite rectangle.

Tail-schedule exactness is the role of U and Q. The active tolerance is bound before the schedule allows rectangle enlargement, and R is read only after that finite route has been displayed.

Dyadic accounting and nonescape then follow from D, E, H, C, P, N. The dyadic row records the tolerance and handoff-error budget before the real seal is reached, while the structural rows preserve only the displayed route and contain none of the excluded host principles. □

Theorem 7826.3 (Moore-Osgood finite-window limit handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ *packet factors through the finite route*

$$W, F, S, U, Q, R, D, E, H, C, P, N.$$

Consequently the packet supplies only a finite iterated-limit handoff surface from $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *windows to regular rational readback and the terminal Real seal; it does not supply a host double-limit convergence theorem, quotient equality of arrays, countable choice, selected limiting representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted packet and project its carrier rows. The only two-parameter source data are the windows in W, and the only iterated handoff rows are the two stream rows F and S.

Apply the obligation surface of Definition 7826.2. It forces any consumer to pass from W through the tail and schedule rows U, Q before reaching the regular readback row R.

The route then continues through the dyadic ledger D to the seal E. Since H, C, P, N preserve and name only those rows, every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer factors through the displayed finite packet, and the excluded host double-limit, quotient, choice, selected-limit, and completeness data cannot be exported. □

Falsifiable prediction: Every $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ handoff read exposes a finite Cauchy-DoubleSequence window, two StreamName handoff rows, a uniform-tail row, a finite schedule, RegSeqRat readback, DyadicRatCore accounting, a RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: mooreosgood, cauchydoublesequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ is not bijective to any listed sibling because it isolates the finite window handoff before the broader Moore-Osgood interchange certificate, while still carrying the two handoff streams, uniform tail, schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}$ packet is generated from a finite CauchyDoubleSequence window, two StreamName handoff rows, a uniform-tail row, a finite schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7826.1, Definition 7826.2, and Definition 7826.3 bind the finite-window iterated-limit handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreOsgoodFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host double-limit convergence, quotient array equality, countable choice, selected limiting representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, window stability, tail-schedule exactness, dyadic accounting, real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.MooreOsgoodFiniteWindowUp.

Chapter 7827

A Concrete Naming Certificate for $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ packet records the finite closed-interval route by which a displayed continuous map receives one source tolerance for a requested target tolerance. It reads its interval endpoints from chapter 394, its graph and pointwise-radius rows from chapter 130, its finite window discipline from chapter 126, its regular rational readback from chapter 301, its terminal value boundary from chapter 38, the fixed-point consumer route of chapter 1437, and the separate modulus-object surface of chapter 2292. The packet names a displayed finite-net modulus route for one located closed interval; it does not claim general Heine–Borel compactness, sequential compactness, selected limits, quotient reals, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7827.1 (Compact interval uniform-continuity carrier)**.** *A* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (J, F, N_\varepsilon, M, W, Q, E, H, C, P, L)$$

with the following visible rows:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *closed-interval row, including the displayed endpoint names and the located endpoint-order evidence consumed by all later reads.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-radius row for the displayed map on J.*
3. *N_ε is the finite interval-net row at the requested target tolerance, listing the inspected subinterval centers and their covering radii.*
4. *M is the rational lower-bound fold over the pointwise source radii attached to those centers, producing the uniform source tolerance.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface for the interval endpoints, inspected centers, nearby input names, and output reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the retained finite windows into regular rational comparisons.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row used only after W and Q have displayed the required input and output comparisons.*
8. *H records componentwise* hsame *transport for $J, F, N_\varepsilon, M, W, Q, E$ and the local certificate rows.*

9. *C records displayed* Cont *replay from a target tolerance request through the interval net, center-wise pointwise-continuity rows, lower-bound fold, window/readback rows, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$, $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalUniformContinuity}^{\uparrow}}$ *row.*

The carrier is an interval-specific uniform-continuity certificate over displayed finite rows. It has no coordinate for general Heine–Borel compactness, sequential compactness, selected limits, quotient real equality, open-cover subcover selection, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompactIntervalUniformContinuityUp.CompactIntervalUniformContinuityTasteGate`
`single carrier alignment`

Theorem 7827.2 (CompactIntervalUniformContinuity NameCert obligations)**.** *For an accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *carrier* $U = (J, F, N_\varepsilon, M, W, Q, E, H, C, P, L)$, *the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalUn}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the located interval, continuous-map row, finite interval-net row, lower-bound fold, finite windows, regular readback, real seal, componentwise transport, continuation replay, provenance, and local naming;*

2. *source lock, requiring J and the source of F to name the same located closed interval before any pointwise radius is read;*

3. *finite-net coverage, requiring every input comparison to pass through one displayed center of N_ε before the pointwise row in F is used;*

4. *lower-bound exactness, requiring the uniform source tolerance to be read from M and not from a host minimum or selected Lebesgue number;*

5. *window and seal nonescape, requiring each downstream value comparison to pass through W, Q, and E before it can be consumed by fixed-point or modulus-object users.*

No obligation may call on general Heine–Borel compactness, sequential compactness, selected limits, quotient reals, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 7827.1. The interval row J and continuous-map row F are the only sources for endpoint, graph, and pointwise-radius information.

The finite-net obligation is carried by N_ε. Each input comparison is routed to one displayed center before the corresponding pointwise radius in F is used, so no open-cover subcover selection or host compactness step is available.

The uniform source tolerance is then computed by the finite fold M. The window/readback rows W and Q expose exactly the input and output comparisons needed by the requested target tolerance, and the real seal E is read only after those rows have been displayed.

Transport, replay, provenance, and local naming preserve the same route. Since the carrier has no coordinate for the excluded compactness or completion principles, the listed obligations exhaust the local NameCert surface. □

Theorem 7827.3 (Compact interval uniform-continuity source lock)**.** *Every accepted* CompactIntervalUniformConti *carrier* $U = (J, F, N_\varepsilon, M, W, Q, E, H, C, P, L)$ *must expose the same located closed interval row J as the source of the continuous-map row F before any consumer reads N_ε, M, W, Q, or E. Thus the*

finite-net radius rows, lower-bound fold, finite windows, regular readback, and real seal all remain locked to the displayed source interval of F.

Proof. Project the carrier of Definition 7827.1. The displayed row J is the only located closed interval coordinate, and F is explicitly the continuous-map row for the displayed map on J.

Now apply the source-lock obligation in Definition 7827.2. Before any pointwise radius is read, the NameCert surface requires J and the source of F to name the same located closed interval.

The remaining rows are downstream of that obligation. The finite net N_ε, the lower-bound fold M, the window row W, the readback row Q, and the real seal E consume pointwise-radius and comparison data only after the source lock has been exposed, so none of them can be read from a different source interval. □

Theorem 7827.4 (Compact interval finite-net modulus)**.** *For every accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *carrier* $U = (J, F, N_\varepsilon, M, W, Q, E, H, C, P, L)$ *and every displayed target tolerance, the source tolerance exported by the packet is obtained by the finite route*

$$J, F, N_\varepsilon, M, W, Q, E, H, C, P, L.$$

If two input names are read within that source tolerance through W and Q, then the output comparison is accepted only through the same centerwise pointwise-continuity rows and the same real-seal row E. The route is a finite-net modulus route, not a general compactness theorem or a selected limit construction.

Proof. Begin with the target tolerance request. The continuation row C sends it to the displayed interval-net row N_ε, whose centers and covering radii are finite rows over the same located interval J.

For each listed center, the continuous-map row F supplies a pointwise source radius. The fold M computes one rational lower bound from those displayed radii, giving the source tolerance exported by the packet.

Input names and output values are then inspected through W and translated through Q. The real seal E can be consumed only after these window and readback rows have displayed the required comparisons.

Thus the modulus route is finite and internal to the packet. It neither selects a limit point nor appeals to open-cover compactness, sequential compactness, quotient real equality, or ambient completeness. □

Definition 7827.5 (Compact interval uniform-continuity modulus carrier)**.** *The modulus carrier extracted from an accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *packet is the finite* BHist *row*

$$U_{\mathrm{mod}} = (J, F, N_\varepsilon, X, W, Q, E, H, C, P, L_{\mathrm{mod}})$$

where J is the located closed-interval row, F is the $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row, N_ε is the finite interval-net row, and X is the sibling* $\mathsf{CompactModulusExtraction}^{\uparrow}$ *row that folds the displayed centerwise radii into one source tolerance. The remaining rows are the* $\mathsf{StreamName}^{\uparrow}$ *finite windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalUniformContinuity}^{\uparrow}}$ *modulus row L_{mod}.*

This carrier is an interval-local modulus surface. It has no row for open-cover compactness, selected subcovers, sequential compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7827.6 (Compact interval uniform-continuity modulus obligations)**.** *For every accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *packet* $U = (J, F, N_\varepsilon, M, W, Q, E, H, C, P, L)$*, the modulus carrier* U_{mod} *of Definition 7827.5 exposes exactly the following obligation surface:*

1. *the located interval row* J *and the continuous-map row* F *must be read before any centerwise radius is folded;*
2. *the finite-net row* N_ε *must display the inspected centers and covering radii for the requested target tolerance;*
3. *the* $\mathsf{CompactModulusExtraction}^{\uparrow}$ *sibling row* X *must perform the finite lower-bound extraction from the displayed centerwise data;*
4. *the windows* W*, readback* Q*, and seal* E *must be displayed before a* $\mathsf{Real}^{\uparrow}$*-facing consumer reads the exported comparison;*
5. *the transport, replay, provenance, and local naming rows* $H, C, P, L_{\mathrm{mod}}$ *must preserve that same route.*

No obligation can be discharged by open-cover compactness, a selected Lebesgue-number object, sequential compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted packet and read Definition 7827.4. It places the exported source tolerance behind the located interval row, the displayed continuous-map row, the finite-net row, and the rational fold already present in the packet.

Replace the fold read by the sibling extraction surface of chapter 6750. That chapter supplies only the finite-net, pointwise-modulus, lower-bound, metric, and modulus handoff rows, so the extraction remains a displayed finite route.

The remaining rows are inherited without widening the carrier. The stream windows, regular readback, real seal, transport, replay, provenance, and local name are exactly the rows already admitted by the interval packet, and none of them contains a compactness or completion principle outside the displayed finite surface. □

Theorem 7827.7 (Compact interval uniform-continuity finite-net route)**.** *Every finite-modulus consumer of an accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *packet must read the interval-specific route*

$$J, F, N_\varepsilon, \mathsf{CompactModulusExtraction}^{\uparrow}, W, Q, E, H, C, P, L_{\mathrm{mod}}.$$

In particular, the consumer receives the located interval row, continuous-map graph and pointwise-modulus rows, finite net row, sibling compact-modulus extraction row, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before it can use the extracted source tolerance.*

Proof. Apply Definition 7827.6. The obligation surface orders the interval, graph, finite-net, and extraction rows before any consumer-facing modulus read.

Now use the finite-net modulus theorem Definition 7827.4. It shows that the source tolerance exported by the interval packet is obtained from the displayed finite route rather than from a selected limit or ambient compactness argument.

Finally replay the structural rows. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback expose the finite comparisons, the $\mathsf{Real}^{\uparrow}$ seal is read only after those comparisons, and $H, C, P, L_{\mathrm{mod}}$ transport, replay, source, and name the same packet. The route therefore reaches later uniform-continuity and compact-interval consumers only through the displayed finite modulus surface. □

Theorem 7827.8 (Compact interval uniform-continuity Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ *packet receives value comparisons only after the route*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

has passed through the displayed finite-net and lower-bound rows N_ε *and* M*. In particular, the fixed-point consumer of chapter 1437 and the modulus-object consumer of chapter 2292 receive only this displayed seal boundary.*

Proof. Read the carrier rows in order. The located interval J fixes the endpoint surface, while the continuous-map row F fixes the graph and pointwise continuity data over that same interval.

The finite-net row N_ε and lower-bound fold M must be displayed before any uniform source tolerance is public. They are the only rows that turn centerwise pointwise data into one tolerance for the whole interval.

The remaining value comparisons pass through the stream windows W, regular readback Q, and real seal E. These are the only Real-facing rows available to consumers such as $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ and $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$.

Consequently any consumer that bypasses $N_\varepsilon, M, W, Q, E$ has left the accepted packet. The chapter supplies no quotient real equality, selected limit, sequential compactness, general Heine–Borel compactness, or $\mathsf{Real}^{\uparrow}$ completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalUniformContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint row, $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-radius rows, a finite interval-net row, a rational lower-bound fold, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7827.1, Definition 7827.2, Definition 7827.4, Definition 7827.5, Definition 7827.6, Definition 7827.7, and Definition 7827.8 seed the interval-specific finite-net uniform-continuity route over located interval, continuous-map, stream-window, regular-readback, real-seal, compact-modulus extraction, fixed-point, and modulus-object consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness, sequential compactness, selected limits, quotient reals, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source lock, finite-net coverage, lower-bound exactness, window and seal nonescape, transport and replay stability, and TasteGate field faithfulness for BEDC.Derived.CompactIntervalUniformC

Chapter 7828

A Concrete Naming Certificate for $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$

Following Arzela and Ascoli, the $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ packet records the constructive finite-net route by which an equicontinuous family exposes a subsequence certificate without selecting an infinite host subsequence. It sits after the finite-net extraction boundary of chapter 7576, the equicontinuous-family carrier of chapter 4036, the compact source surface of chapter 60, the finite-window stream interface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the finite diagonal extraction certificate consumed by downstream compactness arguments; it is not open-cover compactness, a host function-space compactness theorem, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7828.1 (Ascoli equicontinuous subsequence carrier)**.** *An* $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, E, K, W, R, L, D, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$ *finite-net extraction row, exposing the finite centers, tolerance budgets, and value windows available to the diagonal read.*
2. *E is the* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *row, recording one shared modulus route for the displayed family and compact source.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row, read only through finite cover, located, or compact-metric certificates already visible to the packet.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected family indices, source centers, and target value observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational value windows in W.*
6. *L is the finite subsequence-ledger row: it records a displayed increasing list of selected indices for the inspected windows, not a host infinite subsequence.*
7. *D is the diagonal compatibility row, requiring each later window in L to refine the finite-net and equicontinuity data exposed by A, E, K, W, R.*

8. *H records componentwise* hsame *transport for the finite-net, equicontinuity, compact, stream, rational-readback, subsequence-ledger, diagonal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a finite compactness request through A, E, K, W, R, L, D to the downstream subsequence certificate.*
10. *P is the* Pkg *provenance over* $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$, $\mathsf{ArzelaAscoliFiniteNet}^{\uparrow}$, $\mathsf{EquicontinuousFamily}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, E, K, W, R, L, D, H, C, P, N$. It has no coordinate for a selected infinite subsequence, a host function-space limit, quotient equality of maps, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AscoliEquicontinuousSubsequenceUp.AscoliEquicontinuousSubsequenceUp single carrier alignment`

Theorem 7828.2 (Ascoli equicontinuous subsequence obligations)**.** *For every accepted* AscoliEquicontinuousSubseq *carrier*

$$S = (A, E, K, W, R, L, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission; finite-net extraction exposure through A; equicontinuity exposure through E; compact-source exposure through K; finite-window exposure through W; regular rational readback through R; finite subsequence-ledger exposure through L; diagonal compatibility through D; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 7828.1. The rows A, E, K, W, R, L, D are the only mathematical rows, and H, C, P, N are the only structural rows.

The finite-net and equicontinuity inputs are read first. The row A supplies the finite extraction windows of chapter 7576, while E supplies the shared family modulus and K supplies the compact source boundary.

The finite observation route then passes through W and R. These rows expose only displayed stream windows and regular rational readbacks before any Real-facing consumer reaches the terminal seal surface of chapter 38.

The subsequence information is the finite ledger L together with the diagonal compatibility row D. Because L records only displayed increasing finite lists, it cannot provide an infinite host subsequence or a choice-selected limit.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the excluded subsequence-choice, function-space limit, quotient equality, open-cover compactness, countable-choice, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7828.3 (Ascoli equicontinuous finite-net extraction)**.** *Every compactness-facing consumer of an accepted* $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ *carrier must read the displayed route*

$$\mathsf{ArzelaAscoliFiniteNet}^{\uparrow},\ \mathsf{EquicontinuousFamily}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

through the rows A, E, K, W, R before it may consume the finite subsequence-ledger row L and diagonal compatibility row D. The exported certificate is a finite-net subsequence surface for the inspected windows; it supplies no selected infinite subsequence, host function-space compactness theorem, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-net extraction row A. It forces the consumer to use the finite centers and tolerance budgets already named by chapter 7576.

Next read the equicontinuity and compact-source rows. The row E gives a single family modulus over the displayed compact source K, so the diagonal read cannot replace the shared modulus by memberwise radius choices.

The observation windows are then exposed through W and R. The row W names only finite stream coordinates, and R translates those observations through regular rational readback before any $\mathsf{Real}^{\uparrow}$ seal is available.

Only after those rows are fixed can L and D be consumed. They certify finite increasing-index windows and their compatibility with the same finite-net and equicontinuity data.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Because the carrier contains no infinite subsequence selector, host function-space compactness object, quotient map equality, countable-choice schedule, or ambient completeness theorem, none can be exported by the finite-net certificate. □

Falsifiable prediction: Every accepted $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ compactness-facing read factors through ArzelaAscoliFiniteNet, EquicontinuousFamily, Compact, StreamName, RegSeqRat, finite subsequence-ledger, diagonal-compatibility, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: arzela-ascoli-finite-net, equicontinuous-family, compact, streamname, regseqrat, real: $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite-net extraction, equicontinuity, compact-source, finite stream-window, regular rational readback, finite subsequence-ledger, diagonal compatibility, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}}$.

Constructive story: An $\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}$ packet is generated from an ArzelaAscoliFiniteNet row, an EquicontinuousFamily row, a Compact source row, StreamName finite windows, RegSeqRat readback, a finite subsequence ledger, diagonal compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7828.1, Definition 7828.2, and Definition 7828.3 seed the finite-net subsequence certificate consumed by compactness-facing Ascoli routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliEquicontinuousSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, host function-space compactness, quotient equality of maps, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net extraction exposure, equicontinuity exposure, compact-source exposure, finite-window exposure, regular rational readback, finite subsequence-ledger exposure, diagonal compatibility, transport, replay, provenance, and local naming for BEDC.Derived.AscoliEquicontinuousSubsequenceUp.

Chapter 7829

A Concrete Naming Certificate for $\mathsf{BipolarTheorem}^{\uparrow}$

Following Bourbaki and Rockafellar on polar sets and the bipolar theorem in locally convex analysis, the $\mathsf{BipolarTheorem}^{\uparrow}$ packet records the finite BEDC surface by which a displayed convex row is compared with its polar-polar return. It sits after the convex carrier of chapter 208, the support-budget row of chapter 7505, the finite alternative boundary of chapter 6185, the separating-functional packet of chapter 7658, the vector-space source of chapter 47, and the scalar seal of chapter 38. The chapter names only finite polar handoff data; it does not assert host local convexity, a completed dual space, selected supporting functionals, quotient polar equality, or choice.

Definition 7829.1 (Bipolar theorem carrier)**.** *A* $\mathsf{BipolarTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, V, R, S, F, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{ConvexSet}^{\uparrow}$ *row naming the inspected convex set, its displayed membership tests, and the convex-combination ledger available to polar consumers.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row over which every vector, scalar, pairing, and finite affine-functional read is performed.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for all inequality levels, support comparisons, and polar-bound readbacks.*
4. *S is the* $\mathsf{SupportFunction}^{\uparrow}$ *row, supplying only the finite support-budget comparisons attached to inspected directions and witnesses.*
5. *F is the* $\mathsf{FarkasLemma}^{\uparrow}$ *row, supplying the finite alternative ledger used when a polar-bound failure is converted into a displayed multiplier obstruction.*
6. *E is the* $\mathsf{SeparatingHyperplane}^{\uparrow}$ *row, exposing the separating functional only as the finite vector, convex, scalar-level, Farkas, and witness rows already present in that sibling packet.*
7. *Q is the polar-polar handoff row. It lists the inspected point, the displayed polar tests, the finite support-budget comparisons, and the return comparison saying that the point is admitted by the carried convex row exactly through the visible certificate.*

8. *H records componentwise* hsame *transport for the convex, vector, real, support, Farkas, separation, polar-polar, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from convex membership and vector pairing through support budgets, finite alternatives, separating hyperplanes, and the terminal polar-polar return.*
10. *P is the* Pkg *provenance over* $\mathsf{BipolarTheorem}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{SupportFunction}^{\uparrow}$, $\mathsf{FarkasLemma}^{\uparrow}$, $\mathsf{SeparatingHyperplane}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BipolarTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, V, R, S, F, E, Q, H, C, P, N$. It has no coordinate for an ambient dual-space completion, an unbounded family of seminorms, quotient polar equality, selected separating functionals, host closure of convex sets, or choice.

Theorem 7829.2 (Bipolar theorem NameCert obligations)**.** *For every accepted* $\mathsf{BipolarTheorem}^{\uparrow}$ *carrier $B = (X, V, R, S, F, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BipolarTheorem}^{\uparrow}}$ *surface consists exactly of convex-set admission through X, vector-space admission through V, scalar sealing through R, finite support-budget exposure through S, finite-alternative handoff through F, separating-functional exposure through E, polar-polar return through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7829.1. The geometric source is exhausted by X, V, R, so every convex, vector, pairing, and scalar-level read is tied to the displayed rows.

Read the convex-duality siblings next. The support row S contributes only the finite budget surface of chapter 7505; the finite alternative row F contributes only the multiplier or obstruction boundary of chapter 6185; and the separation row E contributes only the displayed affine-functional witness ledger of chapter 7658.

The polar-polar row Q is therefore a handoff over already visible data. Rows H, C, P, N transport, replay, source, and name that same finite packet. Since no carrier coordinate stores host closure, a completed dual space, quotient polar equality, a selector for separating functionals, or choice, the listed rows exhaust the local NameCert obligations. □

Theorem 7829.3 (Bipolar theorem polar handoff)**.** *Every downstream convex-duality consumer of an accepted* $\mathsf{BipolarTheorem}^{\uparrow}$ *packet must read the polar-polar return through*

$$\mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{SupportFunction}^{\uparrow} \longmapsto \mathsf{FarkasLemma}^{\uparrow} \longmapsto \mathsf{SeparatingHyperplane}^{\uparrow}$$

over the shared $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The consumer receives only the finite convex row, displayed vector and scalar seals, support-budget comparisons, finite alternative boundary, separating-functional witness, polar-polar handoff row, componentwise transport, replay, provenance, and local name. It receives no detached bipolar theorem, host closure operation, quotient dual-space equality, selected support functional, or choice principle.*

Proof. Begin with the carried convex row X and the shared vector and scalar rows V, R. These rows fix the only vector, pairing, and inequality endpoints that may be inspected by a polar consumer.

The support route is then read through S. By chapter 7505, this exposes finite support-budget comparisons rather than a host supremum or a selected exposed point.

Next read the obstruction and separation rows. The $\mathsf{FarkasLemma}^{\uparrow}$ row F supplies a finite alternative boundary, and the $\mathsf{SeparatingHyperplane}^{\uparrow}$ row E packages any separating functional as displayed coefficient, scalar-level, and witness rows. Thus the polar-polar handoff Q can only cite the sibling certificate surface already visible in the packet.

Finally H, C, P, N preserve that route under transport, replay, provenance, and naming. Any consumer that replaces it by a detached closure theorem, completed dual-space equality, selected functional, or choice principle would require a carrier row absent from B. □

Closure status for $\mathsf{NameCert}_{\mathsf{BipolarTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{BipolarTheorem}^{\uparrow}$ packet is generated from a ConvexSetUp row, VecSpaceUp row, RealUp scalar seal, SupportFunctionUp support-budget row, FarkasLemmaUp finite alternative row, SeparatingHyperplaneUp witness row, polar-polar handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BipolarTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 7829.1, Definition 7829.2, and Definition 7829.3 seed the finite bipolar-theorem NameCert surface over convex-duality sibling rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BipolarTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close locally convex host closure, completed dual spaces, quotient polar equality, selected separating functionals, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-budget exposure, finite-alternative handoff, separating-functional exactness, polar-polar return, structural transport, replay, provenance, and local naming for BEDC.Derived.BipolarTheoremUp.

Chapter 7830

A Concrete Naming Certificate for $\mathsf{ConnectedCompactInterval}^{\uparrow}$

Following Bishop and Bridges on constructive compact intervals, the $\mathsf{ConnectedCompactInterval}^{\uparrow}$ packet records the finite route by which a located compact interval supplies connectedness data without appealing to open-cover compactness. It sits after the located endpoint packet of chapter 394, the compact uniform-continuity packet of chapter 7827, the interval image-connected route of chapter 7542, the Real seal of chapter 38, the finite StreamName windows of chapter 126, and the RegSeqRat readback boundary of chapter 301. The packet names compact interval connectedness as displayed finite data; it does not assert open-cover compactness, selected subcovers, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7830.1 (Connected compact interval carrier)**.** *A* $\mathsf{ConnectedCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, N, I, W, Q, R, H, C, P, L)$$

with the following displayed rows:

1. *E is the located interval endpoint row, recording the left endpoint, right endpoint, endpoint order, membership boundary, and located interval probes supplied by* $\mathsf{LocatedInterval}^{\uparrow}$*.*
2. *N is the compact finite-net row, recording the dyadic mesh, finite covering windows, mesh-radius certificate, and endpoint coverage consumed from* $\mathsf{CompactIntervalUniformContinuity}^{\uparrow}$*.*
3. *I is the interval image-connected route row, recording the path from interval data to connectedness reads through* $\mathsf{IntervalImageConnected}^{\uparrow}$*.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for endpoints, mesh points, bisection probes, and connectedness challenge reads.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows and rational endpoint comparisons.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint and image-value names, reached only after E, N, I, W, Q are displayed.*
7. *H records componentwise* hsame *transport for endpoints, finite nets, connectedness route, windows, readbacks, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay through the endpoint, finite-net, connectedness-route, window, readback, Real-seal, transport, provenance, and local certificate rows.*

9. *P is the* Pkg *provenance over* ConnectedCompactInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, CompactIntervalUniformContinuity$^{\uparrow}$, IntervalImageConnected$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *L is the local* NameCert$_{\text{ConnectedCompactInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, N, I, W, Q, R, H, C, P, L$. It has no coordinate for open-cover compactness, selected subcovers, host image subsets, quotient real equality, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Theorem 7830.2 (Connected compact interval NameCert obligations)**.** *For every accepted* ConnectedCompactInter *carrier*

$$K = (E, N, I, W, Q, R, H, C, P, L),$$

the local NameCert$_{\text{ConnectedCompactInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for located endpoints, compact finite-net data, interval image-connected routing, finite windows, regular readbacks, Real sealing, transport, replay, provenance, and local naming;*
2. *endpoint exactness, requiring all interval-boundary reads to pass through the displayed* LocatedInterval$^{\uparrow}$ *row E;*
3. *finite-net exactness, requiring all compact interval mesh and coverage reads to pass through N rather than through an open-cover compactness principle;*
4. *connectedness-route exactness, requiring every connectedness read to pass through the interval image-connected route I;*
5. *window and readback exactness, requiring StreamName windows W and RegSeqRat readbacks Q before the terminal* Real$^{\uparrow}$ *seal R is inspected;*
6. *transport and replay locality, requiring H and C to preserve the same displayed rows componentwise;*
7. *nonescape, excluding open-cover compactness, selected subcovers, host image connectedness, quotient real equality, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No connected compact interval obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7830.1. The accepted packet displays one endpoint row, one finite-net row, one connectedness-route row, finite windows, readbacks, a Real seal, and structural support.

Endpoint exactness is supplied by E. Every interval-boundary read uses the same located endpoints and membership boundary, so no separate host interval or quotient real representative is introduced.

Finite-net exactness is supplied by N. Mesh points, finite covering windows, and radius certificates are read as finite packet data, not as an open-cover compactness theorem or a selected subcover functional.

Connectedness-route exactness is supplied by I. The route delegates the connectedness read to the interval image-connected packet after the endpoint and finite-net rows are displayed.

Finally W, Q, R, H, C, P, L impose readback order and local naming. StreamName windows and RegSeqRat readbacks precede the Real seal, and the structural rows transport, replay, source, and name only the displayed carrier rows. □

Theorem 7830.3 (Connected compact interval route)**.** *Every connectedness read exported by an accepted* $\mathsf{ConnectedCompactInterval}^{\uparrow}$ *carrier factors through*

$$E, N \longmapsto I \longmapsto W, Q, R, H, C, P, L.$$

The route uses located interval endpoints, compact finite-net data, the interval image-connected handoff, finite StreamName windows, RegSeqRat readbacks, a Real seal, componentwise transport, displayed continuation replay, package provenance, and the local NameCert row. It exports no open-cover compactness claim.

Proof. Start with an accepted packet $K = (E, N, I, W, Q, R, H, C, P, L)$. The endpoint row E fixes the located interval boundary, and the finite-net row N fixes the compact mesh data available to the interval consumer.

The connectedness read must then pass through I. This row is the only connectedness coordinate in the carrier, and it points to the interval image-connected route rather than to a host connectedness theorem.

The downstream rows W, Q, R, H, C, P, L expose finite observations, rational readbacks, the terminal Real seal, transport, replay, provenance, and local naming. Since no row stores open-cover compactness or a selected subcover, the exported connectedness claim is exactly the displayed finite route. □

Theorem 7830.4 (Connected compact interval Real-seal nonescape)**.** *Every Real-facing seal read of an accepted* $\mathsf{ConnectedCompactInterval}^{\uparrow}$ *packet is downstream of the located endpoint row, compact finite-net row, interval image-connected route, finite-window row, and RegSeqRat readback row. The terminal* $\mathsf{Real}^{\uparrow}$ *row cannot be read as ambient Real completeness, quotient real equality, open-cover compactness, or a selected-subcover principle.*

Proof. The carrier places the Real seal at R after E, N, I, W, Q. Thus a Real-facing consumer first reads the located endpoints, finite compact mesh data, connectedness route, finite windows, and rational readbacks.

The structural rows preserve that order. Componentwise hsame transport and displayed Cont replay move the same packet through provenance and local naming; they add no independent completeness or compactness coordinate.

Therefore the Real seal is a downstream endpoint and image-value seal for the displayed finite route. It is not an escape hatch to open-cover compactness, quotient real equality, selected subcovers, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{ConnectedCompactInterval}^{\uparrow}$ consumer that reads connectedness without the located endpoint row, compact finite-net row, interval image-connected route, StreamName windows, and RegSeqRat readbacks is not reading the accepted packet.

Independence witness: locatedinterval, compact-interval-uniform-continuity, interval-image-connected, streamname, regseqrat, real: none is bijective to $\mathsf{ConnectedCompactInterval}^{\uparrow}$ because this packet jointly carries located endpoints, compact finite-net data, connectedness routing, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConnectedCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConnectedCompactInterval}^{\uparrow}$ packet is generated from located interval endpoints, compact finite-net data, an interval image-connected route, StreamName windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConnectedCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7830.1, Definition 7830.2, Definition 7830.3, and Definition 7830.4 seed the finite connected compact interval route over located endpoints, compact finite nets, interval image-connected routing, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConnectedCompactInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, selected subcovers, host image connectedness, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, finite-net exactness, connectedness-route exactness, window and readback exactness, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.ConnectedCompactIntervalUp.

Chapter 7831

A Concrete Naming Certificate for $\mathsf{CanonicalLatentCompletion}^{\uparrow}$

The $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ packet realizes the selected two-state completion of Definition 9462.1838 as finite NameCert data. It refines the broader $\mathsf{LatentCompletion}^{\uparrow}$ surface of chapter 3521 by fixing the canonical latent rows L_{SS} and $L_{*|W}$, then routing the resulting target-monotone coverage through the finite trigger cube of chapter 7816 and the trigger-hypergraph reliability sibling of chapter 3657. The packet names a BEDC-selected completion witness; it is not an empirical transition model, not an unbounded latent search, and not a global graph-completion oracle.

Definition 7831.1 (Canonical latent completion carrier)**.** *A* $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{CanonicalLatentCompletion}(O, E, S, T, D, M, K, H, C, P, N)$$

with the following displayed rows:

1. *O is the observed module-state ledger inherited from the latent-completion source.*
2. *E is the exception ledger for the table-9, table-14, and table-28 target-monotone failures.*
3. *S is the selected-state row naming exactly L_{SS} and $L_{*|W}$ as the canonical latent completion.*
4. *T is the two-state target-monotone route row, recording the deep-branch and partial-boundary paths of Definition 9462.1839.*
5. *D is the directed rewrite-edge row consumed from the latent-completion packet.*
6. *M is the monotone coverage row verifying that every observed state reaches the target after the selected two states are admitted.*
7. *K is the finite trigger-cube and trigger-hypergraph reliability row, recording the sibling trigger constraints and energy-rank-table sibling reference as displayed finite support.*
8. *H records componentwise* hsame *transport for observed states, exception rows, selected states, target-monotone routes, rewrite edges, coverage, trigger support, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through $O, E, S, T, D, M, K, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CanonicalLatentCompletion}^{\uparrow}$, $\mathsf{LatentCompletion}^{\uparrow}$, $\mathsf{FiniteTriggerCube}^{\uparrow}$, $\mathsf{TriggerHypergraphReliability}^{\uparrow}$, *the energy-rank-table sibling,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{CanonicalLatentCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, E, S, T, D, M, K, H, C, P, N$. It has no coordinate for an alternate latent pair, empirical transition dynamics, unbounded latent search, a hidden graph-completion oracle, or an equality principle outside the displayed rows.

Theorem 7831.2 (Canonical latent completion NameCert obligations)**.** *For every accepted* CanonicalLatentComple *carrier*

$$\mathsf{CanonicalLatentCompletion}(O, E, S, T, D, M, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CanonicalLatentCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for observed module states, exception rows, the selected two canonical latent states, target-monotone route rows, directed rewrite edges, monotone coverage, finite trigger support, componentwise transport, displayed replay, provenance, and local naming.*

Every accepted coverage consumer must read $S = \{L_{SS}, L_{*|W}\}$ before reading T, D, M, K, H, C, P, N. No obligation is stored in an alternate latent-pair selector, empirical transition model, unbounded latent search, hidden graph-completion oracle, or complete state-space claim.

Proof. Carrier admission is projection from Definition 7831.1. The accepted packet displays observed states, exception rows, selected latent states, target-monotone routes, rewrite edges, coverage, trigger support, and structural rows.

The selected-state row is decisive. Since S names exactly L_{SS} and $L_{*|W}$, a consumer omitting either state is not reading the accepted canonical carrier.

The route and coverage rows are T, D, M. They replay the deep-branch and partial-boundary paths, rebuild the finite rewrite-edge graph, and record coverage of every observed endpoint.

The trigger-support row K and the structural rows H, C, P, N preserve the same finite witness through sibling support, transport, replay, provenance, and local naming. None of these rows adds an alternate selector, empirical dynamics, unbounded search, graph oracle, or complete state-space claim. □

Theorem 7831.3 (Canonical latent completion selected-route coverage)**.** *Every target-monotone coverage read exported by an accepted* $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ *packet factors through*

$$O, E, S \longmapsto T, D, M \longmapsto K, H, C, P, N.$$

The selected-state row must expose both L_{SS} and $L_{*|W}$ before the coverage row is read, and the trigger-support row must keep the finite-trigger-cube, trigger-hypergraph-reliability, and energy-rank-table sibling references visible.

Proof. Start with O, E, S. The observed ledger and exception ledger identify the finite target-monotone failures, while the selected-state row supplies the two canonical latent states that answer the independent missing squares.

Next read T, D, M. The target-monotone route row records the two canonical paths, the directed rewrite-edge row rebuilds the finite graph, and the monotone coverage row records reachability for the observed endpoints.

Finally read K, H, C, P, N. The trigger-support row keeps the finite-trigger-cube, trigger-hypergraph-reliability, and energy-rank-table sibling support visible, and the structural rows transport, replay, source, and name the same finite carrier. □

Falsifiable prediction: Any canonical latent-completion consumer that omits either L_{SS} or $L_{*|W}$ fails to read the accepted $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ packet.

Independence witness: latent-completion, finite-trigger-cube, trigger-hypergraph-reliability, energy-rank-table: none is bijective to $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ because this packet fixes the selected two latent states, their target-monotone routes, rewrite coverage, trigger support, transport, replay, provenance, and local naming in one finite carrier.

The Lean follow-up must provide Nontrivial and FieldFaithful instances for $\mathsf{CanonicalLatentCompletion}^{\uparrow}$, separating the selected-state row from the route row, coverage row, trigger-support row, transport row, replay row, provenance row, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{CanonicalLatentCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CanonicalLatentCompletion}^{\uparrow}$ packet is generated from observed module-state rows, exception rows, the selected states L_{SS} and $L_{*|W}$, target-monotone routes, directed rewrite edges, coverage rows, finite trigger support, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CanonicalLatentCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7831.1, Definition 7831.2, and Definition 7831.3 seed the finite canonical latent-completion witness over the selected two latent states, target-monotone routes, rewrite edges, coverage, trigger support, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CanonicalLatentCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close empirical transition dynamics, unbounded latent search, alternate latent-pair selection, complete state-space sampling, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-state exactness, target-monotone route exactness, rewrite-edge reachability, trigger-support exposure, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.CanonicalLatentCompletionUp.

Chapter 7832

A Concrete Naming Certificate for $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$

Following Moore, Osgood, and Bishop–Bridges constructive analysis, the $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ packet records the classical limit-interchange theorem as finite NameCert data over displayed rectangular windows and uniform-tail evidence. It consumes the rectangular double-sequence carrier of chapter 6871, the uniform-tail endpoint route of chapter 463, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite theorem handoff only; it is not a host function space, quotient limit, countable choice principle, selected two-parameter limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7832.1 (Moore-Osgood limit-interchange carrier)**.** *A* $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, U, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ *rectangular double-sequence row, exposing the two index schedules and the inspected finite rectangle.*
2. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *uniform-tail evidence row, binding one tail budget that is read before either iterated route reaches a limit handoff.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, listing exactly the finite row, column, and tail requests opened by the interchange.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rectangular observations and the two iterated regular rational handoffs.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the active precision, rectangular tail budget, and dyadic slack used to compare the two routes.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after D, U, S, R, T have all been displayed.*
7. *H records componentwise* hsame *transport for the double-sequence row, uniform-tail evidence, schedule, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from rectangular requests to schedules, from schedules and uniform-tail evidence to regular readback, and from the dyadic tolerance ledger to the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$, $\mathsf{PringsheimDoubleSequence}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, U, S, R, T, E, H, C, P, N$. It has no coordinate for host function spaces, quotient equality of double sequences, countable choice, selected limiting representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.MooreOsgoodLimitInterchangeUp`

Theorem 7832.2 (Moore-Osgood limit-interchange NameCert obligations)**.** *For every accepted* $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ *carrier $M = (D, U, S, R, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreOsgoodLimitInterchan}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ *rectangle,* $\mathsf{UniformLimit}^{\uparrow}$ *tail evidence,* $\mathsf{StreamName}^{\uparrow}$ *schedules,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows; rectangular-window exactness through D; uniform-tail exactness through U; finite schedule exposure through S; regular readback through R; dyadic tolerance accounting through T; Real-seal nonescape through E; and structural preservation through H, C, P, N.*

Proof. Project the carrier in Definition 7832.1. The tuple lists every row available to a certificate consumer, so carrier admission is componentwise over the displayed rectangle, uniform tail, schedule, readback, tolerance, seal, and structural rows.

The analytic route is ordered by the finite rows. A consumer first reads D to fix the rectangular double-sequence window, then reads U and S to bind the uniform tail and the finite schedules that both iterated routes must share.

The readback and tolerance rows are downstream of that shared route. The row R records the regular rational observations, while T accounts for the precision and tail budget before E is exposed.

The structural rows preserve the same route. Componentwise transport, continuation replay, provenance, and local naming cannot introduce host function spaces, quotient limits, choice, selected representatives, or ambient completeness because no such coordinate is present in M. □

Theorem 7832.3 (Moore-Osgood Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ *packet factors through the finite route*

$$D, U, S \longmapsto R, T \longmapsto E, H, C, P, N.$$

Thus the classical Moore-Osgood interchange is exported only as a displayed rectangular-window and uniform-tail handoff from $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ *and* $\mathsf{UniformLimit}^{\uparrow}$ *data to regular rational readback and a terminal Real seal. The route exports no host function space, quotient limit, countable choice, selected two-parameter limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted packet and project its carrier rows. The Real seal E is not an entrance row: the packet first exposes the rectangular $\mathsf{PringsheimDoubleSequence}^{\uparrow}$ row D, the $\mathsf{UniformLimit}^{\uparrow}$ tail row U, and the finite schedule row S.

Apply Definition 7832.2. It forces the regular readback row R and the dyadic tolerance ledger T to be read only after the rectangle, uniform tail, and schedule rows have been displayed.

The seal and structural rows then package the same finite route. Since H, C, P, N transport, replay, source, and name only the listed rows, a completion-facing consumer receives the displayed factorization and no stronger host-level limiting principle. □

Falsifiable prediction: Every accepted $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ Real-facing read exposes a PringsheimDoubleSequence rectangle, a UniformLimit tail row, finite StreamName schedules, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, transport, continuation, provenance, and local naming before any endpoint is consumed.

Independence witness: pringsheim-double-sequence, uniformlimit, streamname, regseqrat, dyadicratcore, real: $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds rectangular double-sequence windows, one uniform-tail evidence row, finite schedules, regular readback, dyadic tolerance accounting, a terminal Real seal, transport, replay, provenance, and local naming in one theorem packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7832.1, Definition 7832.2, and Definition 7832.3 seed the finite Moore-Osgood limit-interchange route over rectangular windows, uniform tails, finite schedules, regular readback, dyadic tolerances, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MooreOsgoodLimitInterchangeUp`

Bridge status: none

Not claimed: This chapter does not close host function spaces, quotient limits, countable choice, selected two-parameter limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked NameCert obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, rectangular-window exactness, uniform-tail exactness, schedule exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, Real-seal nonescape, and structural transport for BEDC.Derived.MooreOsgoodLimitInterchangeUp.

Constructive story: A $\mathsf{MooreOsgoodLimitInterchange}^{\uparrow}$ packet is generated from a PringsheimDoubleSequence rectangle, UniformLimit tail evidence, StreamName schedules, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7833

A Concrete Naming Certificate for $\mathsf{BishopFanModulusBoundary}^{\uparrow}$

Following Brouwer and Bishop–Bridges finite fan analysis, the $\mathsf{BishopFanModulusBoundary}^{\uparrow}$ packet records the boundary object consumed when finite fan-bar data are handed to compact-uniform-continuity moduli. It sits after the finite fan-bar modulus carrier of chapter 4297, the fan theorem packet of chapter 400, and the finite fan theorem packet of chapter 5863. It realizes the compact finite-bar boundary used by chapter 9237 without adding selected infinite branches, open-cover compactness, countable choice, quotient map equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7833.1 (Bishop fan modulus boundary carrier)**.** *A* $\mathsf{BishopFanModulusBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, A, K, L, R, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite fan-bar source row, read through* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *and the fan theorem boundary.*
2. *A is the admitted finite-bar boundary row, recording the prefix depth, bar classifier, and coverage witness used by downstream compact consumers.*
3. *K is the compact finite-probe alignment row that matches fan prefixes with compact-source observations.*
4. *L is the local modulus ledger attached to the admitted fan leaves.*
5. *R is the positive rational lower-bound row obtained by finite recursion over the admitted leaves.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exposed after F, A, K, L, R have all been displayed.*
7. *H records componentwise* hsame *transport for the finite-bar, boundary, compact-probe, local-modulus, lower-bound, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$F, A, K, L \longmapsto R \longmapsto U.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopFanModulusBoundary}^{\uparrow}$, $\mathsf{FiniteFanBarModulus}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopFanModulusBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, A, K, L, R, U, H, C, P, N$.

Theorem 7833.2 (Bishop fan modulus boundary NameCert obligations)**.** *For every accepted* $\mathsf{BishopFanModulusBoundary}^{\uparrow}$ *carrier $B = (F, A, K, L, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopFanModulusBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite fan-bar source exposure, admitted boundary coverage, compact finite-probe alignment, local modulus exposure, positive rational lower-bound extraction,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored in an open cover, selected infinite branch, countable-choice row, quotient function equality, selected limit, host compactness theorem, or ambient real-completeness principle.*

Proof. Project the carrier of Definition 7833.1. The rows $F, A, K, L, R, U, H, C, P, N$ are exactly the finite fan-bar source, boundary, compact-probe, local-modulus, lower-bound, handoff, transport, replay, provenance, and local naming coordinates.

The source and boundary obligations are forced by F and A. They expose the finite fan-bar and its admitted prefix boundary before any compact-uniform-continuity consumer can inspect a modulus row.

The rows K, L, R then align the finite boundary with compact-source observations, read the local moduli, and fold their positive rational radii into a lower-bound ledger. The handoff row U can consume only that displayed lower-bound row.

Transport, replay, provenance, and local naming preserve the same packet. Since the carrier has no coordinate for open-cover compactness, branch selection, countable choice, quotient function equality, selected limits, host compactness, or ambient real completeness, the listed rows exhaust the NameCert surface. □

Theorem 7833.3 (Bishop fan modulus boundary finite-fan handoff)**.** *Every compact-uniform-continuity read of an accepted* $\mathsf{BishopFanModulusBoundary}^{\uparrow}$ *packet must pass through the finite fan-bar modulus route of chapter 4297 before the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff is exposed. The handoff factors through*

$$\mathsf{FiniteFanBarModulus}^{\uparrow} \longmapsto \mathsf{BishopFanModulusBoundary}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}$$

with compact finite-probe alignment and positive rational lower-bound extraction displayed inside the boundary packet.

Proof. Begin with the obligations of Definition 7833.2. They require the finite fan-bar source row F and the admitted boundary row A before the local-modulus and lower-bound rows can be read.

The sibling finite fan-bar modulus chapter supplies the source route for F. The boundary packet then records the compact-probe row K, the local modulus ledger L, and the positive lower-bound row R that the compact-uniform-continuity consumer needs.

Only after those rows are displayed can U expose the uniform-modulus handoff. Thus the finite fan-bar data are consumed as a boundary object rather than as an open-cover compactness principle, branch selector, chosen limit, or quotient function equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFanModulusBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopFanModulusBoundary}^{\uparrow}$ packet is generated from finite fan-bar source rows, admitted prefix-boundary coverage, compact finite-probe alignment, local modulus ledgers, positive rational lower-bound extraction, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopFanModulusBoundary}\uparrow}$, seed)

Scope closed: Definition 7833.1, Definition 7833.2, and Definition 7833.3 seed the finite fan-bar boundary consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopFanModulusBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite fan-bar source exposure, admitted boundary coverage, compact finite-probe alignment, local modulus exposure, rational lower-bound extraction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopFanModulusBoundaryUp.

Chapter 7834

A Concrete Naming Certificate for $\mathsf{RealModulusNormalization}^{\uparrow}$

Following Bishop and Bridges on moduli of continuity and regular real names, the $\mathsf{RealModulusNormalization}^{\uparrow}$ packet records the finite normalization step that turns compact-uniform local modulus rows into Real-facing uniform-modulus ledger data. It sits over the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the finite schedule surface of chapter 126, the dyadic tolerance core of chapter 291, the compact source surface of chapter 129, the continuous-map pointwise modulus rows of chapter 130, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. It realizes the compact-uniform-continuity route of chapter 9237 by normalizing dyadic tolerances, regular rational readback, stream schedules, and uniform-modulus rows without selecting completed representatives or importing ambient completeness.

Definition 7834.1 (Real modulus normalization carrier)**.** *A* $\mathsf{RealModulusNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, D, S, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row whose local cover or finite-net observations are being normalized.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus row over the same compact source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-normalization ledger, carrying dyadic requests, slack cells, and positive comparison budgets.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the inspected Real-facing windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, S, R have been displayed.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row whose tolerance-only ledger consumes the normalized Real-facing data.*
8. *H records componentwise* hsame *transport for compact-source, continuous-map, dyadic, stream, regular-readback, Real-seal, uniform-modulus, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$K, F, D, S, R \longmapsto E \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{RealModulusNormalization}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, D, S, R, E, U, H, C, P, N$.

Theorem 7834.2 (Real modulus normalization NameCert obligations)**.** *For every accepted* RealModulusNormalizat *carrier $M = (K, F, D, S, R, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealModulusNormalization}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source exposure, continuous-map pointwise modulus exposure, dyadic tolerance normalization, StreamName scheduling, RegSeqRat readback, Real seal exposure, UniformModulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is supplied by quotient real equality, a selected representative, countable choice, open-cover compactness, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7834.1. The rows $K, F, D, S, R, E, U, H, C, P, N$ are the only coordinates available to a compact-uniform-continuity consumer.

The compact-source and continuous-map obligations are K and F. They place every pointwise modulus read over the displayed compact metric source before any Real-facing normalization is inspected.

The normalization rows are D, S, R, E. They convert the tolerance request into dyadic comparison cells, inspect the scheduled finite windows, read regular rational data, and expose the terminal Real seal only after those rows are present.

The uniform-modulus handoff is U, preserved by H, C, P, N. Since no carrier coordinate stores quotient real equality, selected representatives, countable choice, open-cover compactness, selected limits, or ambient Real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 7834.3 (Real modulus normalization tail stability)**.** *Every accepted* $\mathsf{RealModulusNormalization}^{\uparrow}$ *consumer that changes the inspected Real-facing tail must keep the same dyadic tolerance ledger, StreamName schedule, RegSeqRat readback, terminal Real seal, and UniformModulus handoff rows visible before the normalized modulus is read. Tail changes therefore factor through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow}$$

inside the displayed packet rather than through quotient stream equality, selected representatives, or ambient completeness.

Proof. Start from the carrier definition. A tail change can be observed only through the schedule row S and the dyadic tolerance row D, because the packet has no external stream or representative coordinate.

Read the regular rational row R and terminal Real seal E after the schedule and dyadic rows. These rows keep the normalized tail attached to displayed finite windows and regular readback rather than to quotient stream equality.

Finally expose U. The uniform-modulus handoff consumes the same normalized dyadic, stream, regular-readback, and Real-seal route under H, C, P, N, so the accepted consumer cannot insert selected representatives, countable choice, selected limits, open-cover compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusNormalization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealModulusNormalization}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map pointwise modulus rows, dyadic tolerance normalization, StreamName schedules, RegSeqRat readback, a RealUp seal,

UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealModulusNormalization}\uparrow}$, seed)

Scope closed: Definition 7834.1, Definition 7834.2, and Definition 7834.3 seed the Real-facing modulus normalization route consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: FormalStatus(NameCert$_{\text{RealModulusNormalization}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, open-cover compactness, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-map modulus exposure, dyadic tolerance normalization, StreamName scheduling, RegSeqRat readback, Real seal exposure, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealModulusNormalizationUp.

Chapter 7835

A Concrete Naming Certificate for $\mathsf{FiniteDyadicMesh}^{\uparrow}$

Following Bishop and Bridges on finite partitions of compact real intervals, the $\mathsf{FiniteDyadicMesh}^{\uparrow}$ packet records the finite dyadic interval mesh used by L10 Real and Completion consumers before any compactness theorem or selected limit is available. It sits over the endpoint arithmetic of chapter 291, the located-cell surface of chapter 394, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite partition carrier, not an open-cover compactness principle, a host list equality theorem, countable choice, quotient-real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7835.1 (Finite dyadic interval mesh carrier)**.** *A* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, Z, C, W, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the lower endpoint ledger, a finite row of* $\mathsf{DyadicRatCore}^{\uparrow}$ *names for the left endpoints inspected by the mesh.*
2. *Z is the upper endpoint ledger, a finite row of* $\mathsf{DyadicRatCore}^{\uparrow}$ *names for the matching right endpoints, with each row read at the same dyadic scale as its lower endpoint.*
3. *C is the finite* $\mathsf{LocatedInterval}^{\uparrow}$ *cell table, assigning to each displayed adjacent endpoint pair the located interval cell consumed by later Real or Completion reads.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records the endpoint, adjacency, and cell observations requested by the consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after W has exposed the endpoint and cell windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the finite mesh windows and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for endpoint, cell, window, readback, seal, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay of every accepted mesh read through*

$$A, Z, C, W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FiniteDyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicMesh}^{\uparrow}}$ *row.*

The classifier compares accepted mesh packets only through $A, Z, C, W, R, E, H, K, P, N$. It has no coordinate for host list equality, selected refinement trees, countable choice, open-cover compactness, quotient real equality, host real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7835.2 (Finite dyadic mesh NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *carrier*

$$M = (A, Z, C, W, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDyadicMesh}^{\uparrow}}$ *surface consists exactly of lower endpoint exposure, upper endpoint exposure, finite located-cell admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every L10 Real or Completion consumer of the mesh must display those rows before it may read the terminal seal.*

Proof. Project the carrier of Definition 7835.1. The only endpoint data are the two finite dyadic ledgers A, Z, and the only cell data are the located interval rows C.

The finite observation row W records the requested endpoint and cell windows. The readback row R is therefore downstream of the displayed mesh observations, not an independent list selector or completed subdivision tree.

The Real-facing output is the seal row E, and it appears only after R. The structural rows H, K, P, N transport, replay, source, and name the same finite route without adding a second Real, Completion, compactness, or equality coordinate.

Thus the listed rows exhaust the local NameCert surface, and any consumer using host list equality, countable choice, open-cover compactness, quotient real equality, selected limits, or ambient completeness has left the accepted packet. □

Theorem 7835.3 (Finite dyadic mesh subdivision induction)**.** *Let $M = (A, Z, C, W, R, E, H, K, P, N)$ be an accepted* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *carrier. Suppose a displayed finite subdivision step replaces one located dyadic cell in C by its two adjacent located dyadic subcells, with the new endpoints read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and the new cells admitted through* $\mathsf{LocatedInterval}^{\uparrow}$*. Then the resulting finite mesh read still factors through the route*

$$A, Z, C, W \longmapsto R \longmapsto E$$

with the same transport, replay, provenance, and local naming rows. Induction over displayed finite subdivision steps never invokes host natural-number recursion, host list equality, selected infinite branches, countable choice, open-cover compactness, quotient real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The base case is the accepted carrier itself. By Definition 7835.2, every visible read passes through the endpoint ledgers, located cells, finite windows, regular readback, and Real seal.

For one subdivision step, the replacement cell contributes only two new endpoint reads through $\mathsf{DyadicRatCore}^{\uparrow}$ and two new located-cell rows through $\mathsf{LocatedInterval}^{\uparrow}$. The finite window row W records those displayed reads before R is available.

Now iterate over the displayed subdivision history inside the finite BHist packet. Each step rewrites only the finite endpoint and cell rows already named by A, Z, C; the downstream window, readback, and seal route remains W, R, E.

The structural rows H, K, P, N preserve that route under transport, replay, provenance, and local naming. Since the induction ranges only over displayed finite subdivision entries, no host recursion principle, list equality, infinite branch, choice selector, compactness theorem, quotient equality, selected limit, or ambient completeness coordinate is introduced. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDyadicMesh}^{\uparrow}$ Real or Completion read must expose dyadic endpoint rows, located interval cells, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the mesh is consumed.

Independence witness: dyadicratcore, locatedinterval, streamname, regseqrat, real: $\mathsf{FiniteDyadicMesh}^{\uparrow}$ is not bijective to any listed sibling because it binds two endpoint ledgers, a located-cell table, finite observation windows, regular readback, terminal Real seal, transport, replay, provenance, and local naming as one finite mesh certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDyadicMesh}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint ledgers, a finite LocatedInterval cell table, a StreamName finite-window row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7835.1, Definition 7835.2, and Definition 7835.3 seed the finite dyadic mesh route over endpoint ledgers, located cells, finite windows, regular readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDyadicMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host list equality, selected refinement trees, countable choice, open-cover compactness, quotient real equality, host real equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, located-cell admission, finite StreamName coverage, RegSeqRat readback, Real-seal nonescape, subdivision induction, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDyadicMeshUp.

Chapter 7836

A Concrete Naming Certificate for $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$

Following Bishop and Bridges and standard functional analysis, the $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$ packet records the scalar read $f \mapsto f(x)$ as finite NameCert data over a compact displayed source and a continuous real-valued map. It reads compactness through chapter 60, continuity through chapter 130, scalar endpoints through chapter 38, metric support through chapter 4230, and dual-facing scalar-functional rows through chapter 5790 and chapter 2265. The packet names only displayed point evaluation and its boundedness ledger; it does not introduce a host function space, quotient equality of maps, selected norm-attaining witness, or bridge theorem.

Definition 7836.1 (Bounded point-evaluation functional carrier)**.** *For a displayed compact metric source row K and a continuous real-valued map row F, a* $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}(K, F)$ *carrier is a finite* BHist *packet*

$$\mathcal{E} = (K, F, X, V, B, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, including the underlying* $\mathsf{Met}^{\uparrow}$ *carrier and the finite compactness ledger consumed by uniform and dual-facing reads.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}(K, \mathsf{Real}^{\uparrow})$ *row for the real-valued input map.*
3. *X is the point row, naming the displayed element of the compact source.*
4. *V is the evaluation readback row, recording the scalar endpoint $F(X)$ as a* $\mathsf{Real}^{\uparrow}$ *row.*
5. *B is the boundedness ledger, recording that the point-evaluation read is bounded by the displayed scalar-control surface supplied by the compact source and continuous-map rows.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal for the value and bound endpoints read by V and B.*
7. *H records componentwise* hsame *transport for the compact source, continuous real-valued map, point row, evaluation readback, boundedness ledger, scalar seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point-evaluation request through $K, F, X, V, B, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, F, X, V, B, S, H, C, P, N$. It has no coordinate for an ambient function space, quotient equality of continuous maps, unrestricted Banach-dual object, selected representative, or checked standard bridge.

Theorem 7836.2 (Bounded point-evaluation functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}(K, F)$ *carrier*

$$\mathcal{E} = (K, F, X, V, B, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}}$ *surface consists exactly of compact-source admission, continuous real-valued map admission, point-row exposure, evaluation readback, boundedness ledger exposure, RealUp value-and-bound sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape from host function-space or quotient-dual objects. A Banach-dual or bounded-functional consumer may inspect the rows $K, F, X, V, B, S, H, C, P, N$, but it cannot replace them by an ambient evaluation operator without displaying those rows.*

Proof. Project the carrier rows. The source of the read is the compact metric row K, the scalar-valued map is the continuous row F, and the inspected source element is the point row X.

The evaluation readback row V then records the RealUp scalar endpoint $F(X)$. The boundedness ledger B records the finite control needed by a scalar-functional consumer, while the seal S keeps the value and bound endpoints inside the same RealUp surface.

Transport and replay preserve these displayed rows through H and C. The provenance row P exposes exactly the compact, continuous-map, scalar, metric, bounded-functional, and Banach-dual neighbours that may consume the packet.

No remaining carrier coordinate names a host function space, quotient equality of maps, selected representative, unrestricted dual object, or checked bridge. Hence the displayed rows exhaust the local NameCert obligations. □

Theorem 7836.3 (Bounded point-evaluation dual handoff)**.** *Every accepted* $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}(K$ *read exported to a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *or* $\mathsf{BanachDual}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{Compact}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BoundedPointEvaluationFunctional}^{\uparrow}.$$

The route displays the compact source, continuous real-valued map, point row, RealUp evaluation endpoint, boundedness ledger, transport, replay, provenance, and local naming before any dual-facing scalar-functional read is consumed.

Proof. Begin with an accepted packet $\mathcal{E} = (K, F, X, V, B, S, H, C, P, N)$. The compact source and metric support are read through K, while the scalar-valued continuity data are read through F.

The point row X and evaluation row V force the scalar read to be the displayed endpoint $F(X)$. The boundedness ledger B and RealUp seal S then provide the finite scalar-control surface that a bounded-functional or Banach-dual consumer may inspect.

Finally, H, C, P, N transport, replay, source, and name the same rows. Since no host evaluation operator, quotient function class, selected representative, or bridge theorem is present, the exported handoff remains the displayed point-evaluation NameCert route. □

Falsifiable prediction: Every accepted $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$ read displays a compact source, continuous real-valued map, point row, RealUp evaluation endpoint, boundedness

ledger, scalar seal, transport, replay, provenance, and local NameCert row before any bounded-functional or Banach-dual consumer reads the scalar value.

Independence witness: compact, continuousmap, real, metricspace, boundedlinearfunctional, and banachdual: no listed sibling is bijective to $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$ because this packet jointly stores the point row, evaluation readback, boundedness ledger, scalar seal, transport, replay, provenance, and local naming as one finite point-evaluation surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}$ packet is generated from a displayed compact metric source, a continuous real-valued map row, a point row, an evaluation readback row, a boundedness ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7836.1, Definition 7836.2, and Definition 7836.3 seed the finite point-evaluation carrier over compact, continuous-map, RealUp, metric, bounded-functional, and Banach-dual rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedPointEvaluationFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function spaces, quotient equality of maps, selected representatives, unrestricted Banach-dual objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, continuous real-valued map admission, point-row exposure, evaluation readback, boundedness ledger exactness, RealUp sealing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedPointEvaluationFunctionalUp.

Chapter 7837

A Concrete Naming Certificate for $\mathsf{RegulatedPrimitive}^{\uparrow}$

Following Saks and Bishop–Bridges on regulated functions and primitive integration, the $\mathsf{RegulatedPrimitive}^{\uparrow}$ packet records a finite primitive-window object for a regulated integral source. It sits after the regulated-integral rows of chapter 1557, the gauge-integral handoff of chapter 1999, the dyadic tolerance surface of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite family of primitive windows; it is not a host antiderivative, not a quotient of functions, not a global selection of primitives, and not an ambient completeness claim for $\mathsf{Real}^{\uparrow}$.

Definition 7837.1 (Regulated primitive carrier)**.** *A* $\mathsf{RegulatedPrimitive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{RegulatedPrimitive}(I, G, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the regulated integral source row, naming the interval, integrand window, and finite regulated-integral ledger consumed from* $\mathsf{RegulatedIntegral}^{\uparrow}$*.*
2. *G is the gauge and integral handoff row, recording the compatible* $\mathsf{GaugeIntegral}^{\uparrow}$ *route whenever the same primitive window is read through tagged subdivisions.*
3. *W is the primitive-window row, a finite ordered list of subintervals with basepoint, endpoint, orientation, and local integral-read data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling endpoint brackets, window overlap comparisons, and the local error budget for adjacent primitive reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the windowed step sums and gauge sums into regular rational names.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, G, W, D, R have been read.*
7. *H records componentwise* hsame *transport for the regulated source, gauge handoff, primitive windows, dyadic tolerances, regular rational readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay along the route from regulated integral source to primitive window, then through gauge handoff, dyadic tolerance, regular rational readback, and real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedPrimitive}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedPrimitive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, W, D, R, E, H, C, P, N$. It has no coordinate for a host antiderivative, quotient function equality, an unrestricted primitive selector, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedPrimitiveUp.RegulatedPrimitiveTasteGate single carrier alignment`

Theorem 7837.2 (Regulated primitive NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedPrimitive}^{\uparrow}$ *carrier*

$$\mathsf{RegulatedPrimitive}(I, G, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedPrimitive}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated integral source, gauge/integral handoff, finite primitive windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Every accepted primitive consumer must read the finite window row W before reading D, R, E, H, C, P, N. No obligation is stored in a host antiderivative, a quotient of functions, a selected global primitive, a measure-completion object, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 7837.1. The accepted packet displays the regulated source, gauge handoff, primitive windows, tolerance ledger, rational readback, real seal, transport, replay, provenance, and local naming rows.

The primitive-window row is decisive. It lists the basepoint, endpoint, orientation, and local integral read for each inspected subinterval, so a consumer omitting W is not reading the accepted primitive carrier.

The rows D, R, E then control the analytic handoff. The dyadic tolerance row compares adjacent windows and endpoint brackets, the regular rational row records the finite readback, and the real seal is available only after those rows have been consumed.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7837.3 (Regulated primitive integral handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RegulatedPrimitive}^{\uparrow}$ *packet factors through*

$$I, G, W \longmapsto D, R \longmapsto E, H, C, P, N.$$

Thus the regulated integral source, gauge/integral handoff, and primitive-window row are present before any regular rational or real-facing primitive value is exposed.

Proof. Start with the carrier of Definition 7837.1. The rows I and G identify the regulated-integral and gauge-integral sources, while W binds those sources to finite primitive windows.

Next consume D and R. The dyadic tolerance row fixes endpoint brackets and overlap comparisons, and the regular rational row reads the windowed sums as rational names. These rows are the only bridge from window data toward a real-facing value.

Finally consume E, H, C, P, N. The real seal, transport, continuation replay, provenance, and local naming rows preserve the same order of inspection. Since no host antiderivative or selected global primitive appears in the carrier, every accepted real-facing primitive read factors through the displayed finite handoff. □

Falsifiable prediction: Any primitive consumer that reads a $\mathsf{Real}^{\uparrow}$ value without first reading I, G, W, D, R fails to read the accepted $\mathsf{RegulatedPrimitive}^{\uparrow}$ packet.

Independence witness: regulated-integral, gauge-integral, dyadic-rational-core, regular-rational-readback, and real seal: none is bijective to $\mathsf{RegulatedPrimitive}^{\uparrow}$ because this packet binds those source rows to finite primitive windows, overlap tolerances, readback, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedPrimitive}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedPrimitive}^{\uparrow}$ packet is generated from a regulated integral source row, gauge/integral handoff row, finite primitive-window row, DyadicRatCore tolerance row, RegSeqRat readback row, Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedPrimitive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7837.1, Definition 7837.2, and Definition 7837.3 seed the finite primitive-window route over regulated integral source, gauge handoff, dyadic tolerance, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedPrimitive}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegulatedPrimitiveUp.RegulatedPrimitiveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host antiderivatives, quotient function equality, selected global primitive families, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, primitive-window exactness, gauge/integral handoff exactness, DyadicRatCore tolerance exposure, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedPrimitiveUp.

Chapter 7838

A Concrete Naming Certificate for $\mathsf{KnuthBendixCompletion}^{\uparrow}$

Following Knuth and Bendix, the $\mathsf{KnuthBendixCompletion}^{\uparrow}$ packet records the finite critical-pair completion route by which oriented rewrite rules, a well-founded reduction ordering, and residual join witnesses are packaged as NameCert data. It is a traditional completion-theory neighbor of the MetaCIC confluence route: the packet reads the critical-path pressure of chapter 765, the finite parallel-diamond frontier of chapter 4238, and the local critical-pair boundary of chapter 2212. The chapter names a finite completion ledger; it does not claim global termination, full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Definition 7838.1 (Knuth–Bendix completion carrier)**.** *A* $\mathsf{KnuthBendixCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, O, C, S, J, U, H, T, P, N)$$

with the following displayed rows:

1. *R is the finite rewrite-rule row, listing oriented rules $l_i \to r_i$ and the finite signature cells on which the rules are admitted.*
2. *O is the well-founded reduction ordering row, recording the displayed decrease checks used to orient each accepted equation.*
3. *C is the critical-pair ledger row, listing every displayed overlap inspected by the completion packet.*
4. *S is the completion-step row, recording whether an overlap is joined, oriented into a new displayed rule, or routed to an obstruction.*
5. *J is the residual join row, listing the finite reduct witnesses consumed when a critical pair is declared locally joinable.*
6. *U is the obstruction row, naming unorientable equations, failed decrease checks, unresolved joins, and resource bounds as visible obstructions rather than hidden proof debt.*
7. *H records componentwise* hsame *transport for rewrite rules, ordering checks, critical-pair rows, completion steps, residual joins, obstructions, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay by which a completion consumer reads*

$$R, O, C, S, J, U$$

before any confluence-facing handoff is made.

9. *P is the* Pkg *provenance over* $\mathsf{KnuthBendixCompletion}^{\uparrow}$*, finite rewrite rules, well-founded ordering rows, critical-pair ledgers, completion steps, residual joins, obstruction rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{KnuthBendixCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary term rewriting systems, external termination proofs, global Church–Rosser, quotient term equality, evaluator equality, subject reduction, MetaCIC kernel consistency, or unchecked residual-substitution discharge.

Theorem 7838.2 (Knuth–Bendix critical-pair resolution)**.** *Let $K = (R, O, C, S, J, U, H, T, P, N)$ be an accepted* $\mathsf{KnuthBendixCompletion}^{\uparrow}$ *carrier. Every critical pair consumed from K is resolved only through the displayed route*

$$C \longmapsto S \longmapsto J, U.$$

A row of S may certify a local join by pointing to J, may orient a new displayed rule by pointing back to the decrease checks in O, or may expose an obstruction in U. No consumer may read the route as full Church–Rosser, arbitrary termination, quotient term equality, evaluator equality, subject reduction, or MetaCIC kernel consistency.

Proof. Project the carrier of Definition 7838.1. The only critical-pair ledger is C, the only completion-step ledger is S, the only residual join ledger is J, and the only obstruction ledger is U.

A locally joinable pair is therefore witnessed by a row of S whose target is a displayed residual join in J. An oriented equation is witnessed by a row of S whose target is a displayed decrease check in O, and an unresolved pair is witnessed by a row of S whose target is the obstruction row U.

Rows H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no global termination theorem, Church–Rosser theorem, quotient equality, evaluator equality, subject-reduction row, or kernel-consistency certificate, none of those claims can be exported by a critical-pair consumer. □

Theorem 7838.3 (Knuth–Bendix NameCert obligations)**.** *For every accepted* $\mathsf{KnuthBendixCompletion}^{\uparrow}$ *carrier*

$$K = (R, O, C, S, J, U, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KnuthBendixCompletion}^{\uparrow}}$ *surface consists exactly of rewrite-rule admission, well-founded ordering checks, critical-pair ledgering, completion-step ledgering, residual join exposure, obstruction exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7838.1. It exposes the ten rows $R, O, C, S, J, U, H, T, P, N$ and no hidden row.

The rule row R and ordering row O provide the only source for oriented rewrite rules. The critical-pair row C and step row S provide the only source for completion progress.

The residual join row J and obstruction row U exhaust the possible displayed outcomes for inspected overlaps. Finally, H, T, P, N carry transport, replay, provenance, and local naming for those same rows. Thus the listed obligations are exactly the local NameCert surface. □

Theorem 7838.4 (Knuth–Bendix MetaCIC frontier handoff)**.** *The* KnuthBendixCompletion$^{\uparrow}$ *packet is a finite completion neighbor for the MetaCIC confluence frontier. A MetaCIC-facing consumer may compare its critical-pair ledger to chapter 2212 and its obstruction rows to chapter 4238, but the handoff remains a local finite completion ledger controlled by the critical-path boundary of chapter 765.*

Proof. Read the packet through Definition 7838.2. Its critical-pair rows are finite ledger entries, and each inspected row is sent either to a residual join, an oriented rule backed by the displayed ordering check, or an obstruction.

The MetaCIC critical-pair boundary supplies a sibling finite ledger for confluence audit rows, while the parallel-diamond frontier supplies the residual-frontier and obstruction-facing handoff. The critical-path chapter is the upstream route that keeps these consumers from treating a local completion row as a kernel-wide confluence theorem.

Therefore the comparison is a route comparison between finite NameCert packets. It does not add global termination, full Church–Rosser, subject reduction, evaluator equality, quotient equality, residual-substitution compatibility, or kernel consistency. □

Falsifiable prediction: Every accepted KnuthBendixCompletion$^{\uparrow}$ packet exposes finite rewrite rules, well-founded ordering checks, critical-pair ledger rows, completion-step rows, residual join rows, obstruction rows, transport, replay, provenance, and local NameCert rows before a completion consumer reads the packet.

Independence witness: Nearest siblings: MetaCICCriticalPath, MetaCICParallelDiamond-Frontier, and MetaCICConfluenceCriticalPairBoundary. None is bijective to KnuthBendixCompletion$^{\uparrow}$ because this packet jointly carries traditional rewrite rules, reduction-ordering checks, completion-step outcomes, residual joins, obstructions, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{KnuthBendixCompletion}^{\uparrow}}$**.**

Constructive story: A KnuthBendixCompletion$^{\uparrow}$ packet is generated from finite rewrite-rule rows, well-founded reduction-ordering checks, critical-pair ledger rows, completion-step rows, residual join rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KnuthBendixCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7838.1, Definition 7838.2, Definition 7838.3, and Definition 7838.4 seed the finite completion route through rewrite rules, ordering checks, critical pairs, completion steps, residual joins, obstructions, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{KnuthBendixCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary termination, full Church–Rosser, subject reduction, evaluator equality, quotient term equality, residual-substitution compatibility, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, ordering decrease rows, critical-pair coverage, completion-step locality, residual join exposure, obstruction exposure, transport, replay, provenance, and local naming for BEDC.Derived.KnuthBendixCompletionUp.

Chapter 7839

A Concrete Naming Certificate for $\mathsf{FiniteModulusTransfer}^{\uparrow}$

Following Bishop's finite-net reading of compact uniform continuity, the $\mathsf{FiniteModulusTransfer}^{\uparrow}$ packet records the finite transfer by which a compact source, a continuous-map row, and local modulus witnesses are folded into one tolerance-indexed uniform-modulus handoff. It concretizes the finite modulus branch of chapter 9237 beside the compact source of chapter 129, the continuous-map carrier of chapter 130, the uniform-modulus consumer of chapter 369, and the real-seal route of chapter 38. The packet is a transfer ledger, not an open-cover theorem, selected-subcover principle, host equality of functions, or $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 7839.1 (Finite modulus transfer carrier)**.** *A* $\mathsf{FiniteModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, L, G, B, U, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, listing the inspected probes and their cover relation to source requests.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose local continuity data is being transferred.*
3. *L is the local modulus row, assigning displayed pointwise tolerance witnesses to the finite probe list.*
4. *G is the finite gauge-transfer row, recording the finite comparisons that move each local witness to a common tolerance request.*
5. *B is the lower-bound fold row, producing the finite candidate selected for the uniform handoff from the displayed gauge ledger.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row reached only after K, F, L, G, B have been displayed.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the transferred tolerance value, read only as a seal on the finite fold and not as a completeness theorem.*
8. *H records componentwise* hsame *transport for the compact source, continuous-map row, local modulus row, gauge-transfer row, lower-bound fold, uniform handoff, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of every transfer query through*

$$K, F, L, G, B \longmapsto U, R.$$

10. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusTransfer}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusTransfer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, external subcover choice, host-level function equality, quotient real equality, countable choice, trichotomy, Lean verification, or bridge credentials.

Theorem 7839.2 (Finite modulus transfer obligations)**.** *For every accepted* $\mathsf{FiniteModulusTransfer}^{\uparrow}$ *carrier*

$$M = (K, F, L, G, B, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusTransfer}^{\uparrow}}$ *surface consists exactly of compact finite-net source admission, continuous-map admission, local modulus exposure, finite gauge transfer, lower-bound folding, uniform-modulus handoff, real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7839.1. The displayed rows are exactly $K, F, L, G, B, U, R, H, C, P, N$.

Rows K and F provide the source and map data. Row L exposes the local modulus witnesses on the finite probe list, and row G records the finite comparisons that transfer those witnesses to a common tolerance request.

The lower-bound fold B is the only row that selects a transfer candidate, while U and R are downstream handoffs. Rows H, C, P, N only transport, replay, source, and name the displayed route. Hence the listed rows exhaust the local NameCert surface. □

Theorem 7839.3 (Finite modulus transfer handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *output exported by an accepted* $\mathsf{FiniteModulusTransfer}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, L, G, B \longmapsto \mathsf{UniformModulus}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The output is therefore a finite transfer from compact-source probes and local modulus witnesses to a uniform-modulus handoff. It is not an arbitrary open-cover compactness theorem, selected-subcover principle, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or bridge checking.

Proof. Start with the source rows K and F from Definition 7839.1. They display the compact finite-net probes and the continuous-map row before any modulus output is available.

The local modulus row L and gauge-transfer row G are read next. They record finitely many local witnesses and the comparisons that make those witnesses comparable under the requested tolerance.

Only after the lower-bound fold B has been displayed can U and R be consumed. Since H, C, P, N merely transport, replay, source, and name this route, no excluded compactness, selection, equality, choice, verification, or bridge datum is exported. □

Falsifiable prediction: Every accepted $\mathsf{FiniteModulusTransfer}^{\uparrow}$ packet exposes compact finite-net source rows, ContinuousMap rows, local modulus rows, gauge-transfer rows, lower-bound fold rows,

UniformModulus handoff rows, Real seal rows, transport, replay, provenance, and local NameCert rows before a uniform-modulus consumer reads the packet.

Independence witness: Nearest siblings: CompactMetric, ContinuousMap, UniformModulus, Real, CompactModulusEvaluation, and CompactIntervalUniformContinuity. None is bijective to $\mathsf{FiniteModulusTransfer}^{\uparrow}$ because this packet isolates the finite transfer ledger from local modulus witnesses through gauge comparison and lower-bound folding into a uniform-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusTransfer}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap rows, local modulus rows, finite gauge-transfer rows, lower-bound fold rows, UniformModulus handoff rows, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7839.1, Definition 7839.2, and Definition 7839.3 seed the finite compact-source, continuous-map, local-modulus, gauge-transfer, lower-bound, uniform-modulus, real-seal, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusTransfer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host equality of functions, quotient real equality, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source stability, continuous-map stability, local modulus exposure, gauge-transfer locality, lower-bound fold exactness, uniform-modulus handoff, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusTransferUp.

Chapter 7840

A Concrete Naming Certificate for $\mathsf{QuasiCauchySequence}^{\uparrow}$

Following Bishop and Bridges on constructive real sequences, the $\mathsf{QuasiCauchySequence}^{\uparrow}$ packet records the adjacent-difference boundary of a real sequence as finite BEDC NameCert data. It sits on the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later Real and Completion arguments a sequence frontier where adjacent gaps vanish without asserting the full Cauchy condition, selecting limits, quotienting streams, invoking countable choice, or claiming ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7840.1 (Quasi-Cauchy sequence carrier)**.** *A* $\mathsf{QuasiCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the inspected real sequence by finite windows.*
2. *W is the adjacent-window schedule row exposing each displayed pair (x_n, x_{n+1}).*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare adjacent differences.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational difference stream attached to the displayed windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for each inspected adjacent distance value.*
6. *A is the adjacent-difference ledger, recording that every accepted tolerance request is eventually met by the displayed adjacent rows.*
7. *H records componentwise* hsame *transport for the stream, schedule, dyadic, regular-rational, Real-seal, adjacent-ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a quasi-Cauchy request through S, W, D, R, E, A, H.*
9. *P is the* Pkg *provenance over* $\mathsf{QuasiCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiCauchySequence}^{\uparrow}}$ *row.*

The classifier accepts a sequence read only through $S, W, D, R, E, A, H, C, P, N$. It has no coordinate for arbitrary tail-pair Cauchy comparison, selected limit, quotient stream equality, countable-choice schedule, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7840.2 (Quasi-Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{QuasiCauchySequence}^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QuasiCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream source, adjacent-window schedule, dyadic tolerance ledger, regular-rational readback, terminal Real seal, adjacent-difference ledger, componentwise transport, displayed replay, provenance, and local naming rows; adjacent-window exposure before any difference read; tolerance-readback coherence through D, R, E; eventual adjacent-smallness through A; and structural preservation through H, C, P, N. No quasi-Cauchy obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7840.1. The tuple displays the stream source, adjacent schedule, dyadic tolerance ledger, regular-rational readback, Real seal, adjacent-difference ledger, and structural rows.

The adjacent-window exposure clause is exactly the row pair S, W. It forces a consumer to inspect displayed neighbouring entries before reading any distance or tolerance datum.

The tolerance coherence clause is stored in D, R, E. The dyadic tolerance row supplies the requested precision, the regular-rational row reads the finite rational difference stream, and the Real seal records the accepted adjacent distance as a displayed real-name boundary.

Finally A, H, C, P, N record eventual adjacent-smallness, transport, replay, source, and local naming. Since no carrier coordinate compares arbitrary distant tail pairs or chooses a limit, the listed obligations exhaust the local NameCert surface without promoting the packet to a full Cauchy or completion theorem. □

Theorem 7840.3 (Quasi-Cauchy adjacent-difference ledger)**.** *Every accepted* $\mathsf{QuasiCauchySequence}^{\uparrow}$ *packet routes each Real- or Completion-facing adjacent-difference request through*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

and then through the adjacent ledger A and structural rows H, C, P, N. The route certifies only that displayed adjacent differences become small along the inspected schedule; it does not certify the full Cauchy comparison of arbitrary tail pairs, a selected limiting real, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted adjacent-difference request. The carrier forces the request to read S and W, so the compared entries are the displayed neighbours of the same finite stream schedule.

The request then passes through D, R, E. These rows expose the dyadic tolerance, the regular rational readback of the difference stream, and the terminal Real seal before any completion-facing consumer can inspect the ledger.

The row A records the eventual adjacent-smallness claim for exactly those displayed neighbouring entries. The structural rows H, C, P, N preserve and name the same packet.

Because the route never adds a coordinate comparing arbitrary x_m and x_n in the tail, it remains the quasi-Cauchy adjacent-difference ledger and cannot be reread as full Cauchy convergence, selected completion, quotient stream equality, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{QuasiCauchySequence}^{\uparrow}$ consumer is refuted if it treats adjacent-smallness as a full Cauchy tail comparison without displaying a separate arbitrary-tail Cauchy ledger.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{QuasiCauchySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the adjacent-window schedule, dyadic tolerance ledger, regular-rational difference readback, Real seal, adjacent-smallness ledger, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasiCauchySequence}^{\uparrow}$ packet is generated from finite StreamName windows, adjacent-pair schedules, DyadicRatCore tolerance rows, RegSeqRat difference readback, Real seal rows, adjacent-smallness ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7840.1, Definition 7840.2, and Definition 7840.3 seed the finite adjacent-difference packet over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, full Cauchy convergence, selected limits, quotient stream equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, adjacent-window exposure, dyadic and regular readback coherence, Real-seal preservation, adjacent-smallness exactness, transport stability, and replay stability for BEDC.Derived.QuasiCauchySequenceUp.

Chapter 7841

A Concrete Naming Certificate for $\mathsf{FiniteRankDeficiency}^{\uparrow}$

The $\mathsf{FiniteRankDeficiency}^{\uparrow}$ packet realizes the local support-rank prototype of Definition 9462.1756 and Definition 9462.1760 as finite NameCert data. It generalizes the stop/Trp face packet of chapter 4090 by making the support universe, proper-support classifier, witness ledger, and non-escape boundary explicit. It also reads the finite coverage-rank algebra of chapter 3734 and the rank-nullity accounting surface of chapter 3438 as sibling ledgers. The packet is a finite support-rank certificate; it is not a host rank oracle, a biological semantics claim, quotient support equality, an infinite support lattice, or a basis-selection theorem.

Definition 7841.1 (Finite rank-deficiency carrier)**.** *A* $\mathsf{FiniteRankDeficiency}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, S, R, P, L, K, T, C, G, N)$$

with the following displayed rows:

1. *U is the finite support-universe row naming the inspected support coordinates.*
2. *S is the finite support-family row whose members are compared with U.*
3. *R is the support-rank row, recording the visible size or rank token of each displayed support.*
4. *P is the proper-support classifier row, accepting exactly the displayed supports that remain below the full support named by U.*
5. *L is the deficiency ledger separating proper-support rows from the full-support corner.*
6. *K is the consumed* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *sibling row when the support family comes from a coverage or trigger algebra.*
7. *T is the consumed* $\mathsf{RankNullity}^{\uparrow}$ *sibling row when the support-rank read is compared with a kernel-image accounting surface.*
8. *C records componentwise* hsame *transport for support universes, support rows, rank rows, proper-support classifiers, ledgers, sibling algebra rows, replay, provenance, and local naming.*
9. *G records displayed* Cont *replay from a finite rank-deficiency request through U, S, R, P, L, K, T, C.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRankDeficiency}^{\uparrow}}$ *row, with* Pkg *provenance over the displayed rows, Definition 9462.1760,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

The classifier accepts a finite rank-deficiency read only through $U, S, R, P, L, K, T, C, G, N$. It has no coordinate for hidden full-support escape, host rank completion, quotient support equality, biological interpretation, stochastic dynamics, infinite support search, or selected bases.

Theorem 7841.2 (Finite rank-deficiency NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRankDeficiency}^{\uparrow}$ *carrier*

$$F = (U, S, R, P, L, K, T, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRankDeficiency}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite support universe, support family, rank row, proper-support classifier, deficiency ledger, optional coverage-rank sibling row, optional rank-nullity sibling row, componentwise transport, displayed replay, provenance, and local naming; proper-support exactness for P against the full support named by U; ledger separation of proper-support rows from the full-support corner; sibling preservation through K and T whenever those rows are present; and structural preservation through C, G, N. No finite rank-deficiency obligation is stored outside these rows.*

Proof. Project the carrier of Definition 7841.1. The rows U, S, R, P, L display the finite support universe, support family, rank read, proper-support classifier, and deficiency ledger before any sibling algebra is consumed.

Proper-support exactness is the finite law of Definition 9462.1760: accepted deficient rows are those whose displayed support remains below the full support named by U. The ledger row L keeps the full-support corner visible as a refused escape rather than erasing it.

The sibling rows K and T do not add hidden rank data. They only route already displayed coverage-rank and rank-nullity surfaces into the same finite support-rank packet.

Finally C, G, N transport, replay, source, and name the same rows. Since hidden support completion, quotient support equality, biological interpretation, infinite search, and selected bases are absent from the carrier, the listed obligations exhaust the NameCert surface. □

Theorem 7841.3 (Finite rank-deficiency proper-support nonescape)**.** *Every accepted* $\mathsf{FiniteRankDeficiency}^{\uparrow}$ *packet that reads a proper-support row must keep the route*

$$U, S, R, P, L, K, T, C, G, N$$

visible until the local $\mathsf{NameCert}$ *row is consumed. In particular, a consumer cannot promote the proper-support classifier P to the full-support corner of U, cannot replace the displayed rank row R by a host rank oracle, and cannot identify two support packets by quotient support equality unless the required transport is already displayed in C.*

Proof. Start from an accepted proper-support read. Carrier projection exposes U, S, R, P, L, so the support universe, inspected support family, rank token, classifier, and deficiency ledger are all visible.

The nonescape claim follows from the separation row L. It records the full-support corner as a boundary distinct from the accepted proper-support rows, so a downstream consumer cannot collapse the boundary into the classifier without adding a row not present in the packet.

The coverage-rank and rank-nullity siblings are read only through K and T. These rows preserve the same support-rank evidence and do not supply a host rank oracle or selected basis.

The structural rows C, G, N then transport, replay, source, and name the displayed packet. Therefore any read that leaves $U, S, R, P, L, K, T, C, G, N$ has left the finite rank-deficiency NameCert route. □

Falsifiable prediction: A $\mathsf{FiniteRankDeficiency}^{\uparrow}$ consumer is refuted if it treats a proper-support classifier as a full-support conclusion without displaying the deficiency ledger L and the transport row C.

Independence witness: stop-trp-face-rank-deficiency, coverage-rank-algebra, rank-nullity: $\mathsf{FiniteRankDeficiency}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite support universe, support-family row, support-rank row, proper-support classifier, nonescape ledger, sibling algebra rows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRankDeficiency}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRankDeficiency}^{\uparrow}$ packet is generated from finite support universes, displayed support-family rows, support-rank rows, proper-support classifiers, deficiency ledgers, CoverageRankAlgebra sibling rows, RankNullity sibling rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRankDeficiency}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7841.1, Definition 7841.2, and Definition 7841.3 seed the finite support-rank deficiency packet over the rank-deficiency prototype.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRankDeficiency}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host rank completion, quotient support equality, biological semantics, stochastic dynamics, infinite support search, or selected bases.

Upgrade path: Obligation closure requires checked carrier admission, proper-support exactness, deficiency-ledger separation, coverage-rank sibling preservation, rank-nullity sibling preservation, transport stability, and replay stability for BEDC.Derived.FiniteRankDeficiencyUp.

Chapter 7842

A Concrete Naming Certificate for $\mathsf{RearrangementInequality}^{\uparrow}$

Following Hardy, Littlewood, and Polya, the $\mathsf{RearrangementInequality}^{\uparrow}$ packet records the finite sorted-sum certificate for two equally indexed real rows. It consumes finite sequence structure from chapter 36, order comparison from the $\mathsf{TotalOrder}^{\uparrow}$ surface, rational readback through $\mathsf{Rat}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal real seal of chapter 38. It is adjacent to the convex inequality route of $\mathsf{JensenInequality}^{\uparrow}$ and the majorization route of $\mathsf{KaramataInequality}^{\uparrow}$, but it does not duplicate either packet: rearrangement names the finite permutation ledger and sorted pairwise-product comparison.

Definition 7842.1 (Rearrangement inequality carrier)**.** *A* $\mathsf{RearrangementInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{R} = (A, B, O, L, \Pi, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are finite* $\mathsf{List}^{\uparrow}$ *rows of equal displayed length whose entries are sealed as* $\mathsf{Real}^{\uparrow}$ *names.*
2. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison ledger certifying that both rows are read in nondecreasing order.*
3. *L is the length and index ledger tying the two rows to the same finite index set.*
4. *Π is the permutation ledger, recording how any compared arrangement is obtained from the sorted row by a finite list permutation.*
5. *S is the finite sorted-sum comparison row for the two product sums.*
6. *Q is the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows used to inspect the real products and sums.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for the sum endpoints after the finite comparison has been displayed.*
8. *H records componentwise* hsame *transport for the list, order, index, permutation, sum, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a rearrangement request through $A, B, O, L, \Pi, S, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RearrangementInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{JensenInequality}^{\uparrow}$, $\mathsf{KaramataInequality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RearrangementInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for ambient completeness of $\mathsf{Real}^{\uparrow}$*, host real ordering, quotient equality of real names, selected extrema, infinite series, or a hidden optimization principle.*

Theorem 7842.2 (Rearrangement inequality NameCert obligations)**.** *For every accepted* $\mathsf{RearrangementInequality}^{\uparrow}$ *carrier*

$$\mathcal{R} = (A, B, O, L, \Pi, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RearrangementInequality}^{\uparrow}}$ *surface consists exactly of sorted finite RealUp rows, the order comparison ledger, equal-length and index alignment, finite permutation provenance, finite product-sum comparison, RatUp and RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape from ambient completeness, host order, quotient equality, selected extrema, or infinite-series assertions.*

Proof. Project the carrier. The finite rows A and B supply the displayed list spines and RealUp entries, while O and L bind the sortedness and common index surface.

The permutation and comparison data are the rows Π and S. They record only finite rearrangements of the displayed list spines and the corresponding finite product-sum comparison.

The readback and endpoint rows are Q and E. They route rational windows through $\mathsf{Rat}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before sealing the compared sums as $\mathsf{Real}^{\uparrow}$ endpoints.

Transport, replay, provenance, and naming are carried by H, C, P, N. Since no remaining row stores completeness, host order, quotient equality, selected extrema, or infinite series, the listed rows exhaust the local NameCert obligations. □

Theorem 7842.3 (Rearrangement inequality sorted-sum handoff)**.** *Every accepted* $\mathsf{RearrangementInequality}^{\uparrow}$ *packet exports its finite sorted-sum comparison only through the route*

$$\mathsf{List}^{\uparrow},\ \mathsf{TotalOrder}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RearrangementInequality}^{\uparrow}.$$

A Jensen or Karamata consumer may inspect this handoff as a finite inequality certificate only after the sorted rows, common index ledger, permutation ledger, rational readback, RealUp seals, transport, replay, provenance, and local naming rows have been displayed.

Proof. Begin with an accepted carrier $\mathcal{R} = (A, B, O, L, \Pi, S, Q, E, H, C, P, N)$. The finite list rows and common index ledger determine the product positions that can be compared.

The order row O and permutation row Π force every admissible arrangement to be read as a finite permutation of the displayed sorted rows. The comparison row S then records the sorted product-sum inequality for that finite index set.

The rational readback row Q and RealUp seal E keep the numerical comparison inside the displayed BEDC real surface. The rows H, C, P, N transport, replay, source, and name the same data.

Thus the handoff can be consumed by $\mathsf{JensenInequality}^{\uparrow}$ or $\mathsf{KaramataInequality}^{\uparrow}$ only as a finite sorted-sum certificate. It exports no ambient completeness theorem, host real-order theorem, quotient equality, selected extremum, or infinite-series claim. □

Falsifiable prediction: Every accepted $\mathsf{RearrangementInequality}^{\uparrow}$ packet displays sorted finite $\mathsf{Real}^{\uparrow}$ rows, common finite indices, an order ledger, a permutation ledger, finite sum comparison, $\mathsf{Rat}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local $\mathsf{NameCert}$ rows before any inequality consumer reads the sorted-sum comparison.

Independence witness: real, rat, regseqrat, totalorder, list, jenseninequality, and karamatainequality: no listed sibling is bijective to $\mathsf{RearrangementInequality}^{\uparrow}$ because this packet jointly stores the finite sorted-row ledger, permutation provenance, and sorted product-sum comparison as one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RearrangementInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RearrangementInequality}^{\uparrow}$ packet is generated from sorted finite RealUp list rows, a TotalOrder comparison ledger, common finite index data, permutation provenance, finite product-sum comparison, RatUp and RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RearrangementInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7842.1, Definition 7842.2, and Definition 7842.3 seed the finite rearrangement-inequality carrier over RealUp, RatUp, RegSeqRat, TotalOrder, List, Jensen, and Karamata rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RearrangementInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient completeness of RealUp, host real ordering, quotient equality of real names, selected extrema, infinite series, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked sorted finite RealUp rows, order and permutation ledgers, finite sum comparison, RatUp and RegSeqRat readback, RealUp sealing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RearrangementInequalityUp.

Chapter 7843

A Concrete Naming Certificate for $\mathsf{HyperbolicHoroball}^{\uparrow}$

Following Gromov and Bridson–Haefliger on hyperbolic spaces, the $\mathsf{HyperbolicHoroball}^{\uparrow}$ packet records a horoball by finite boundary-point, Busemann-threshold, and visual-shadow rows. It sits after the metric-read surface of chapter 3474, the Busemann-boundary packet of chapter 3732, the boundary-shadow packet of chapter 3813, the visual-metric packet of chapter 3866, and the terminal real-name seal of chapter 38. The packet names a finite horoball carrier; it is not a completed ideal-boundary quotient, not a global geodesic selection, not a measure class at infinity, and not an ambient hyperbolic space.

Definition 7843.1 (Hyperbolic horoball carrier)**.** *A* $\mathsf{HyperbolicHoroball}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{HyperbolicHoroball}(B, U, S, M, V, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-point row consumed from* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$*, naming the finite boundary end and its Busemann-level read.*
2. *U is the Busemann-threshold row, a displayed* $\mathsf{Real}^{\uparrow}$*-sealed level used to decide the local horoball side.*
3. *S is the visual-shadow row consumed from* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*, recording the finite boundary-shadow window seen by downstream observer-expansion reads.*
4. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row over the same disk or hyperbolic source observations.*
5. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row comparing the shadow window with the visual metric read at the same boundary end.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for Busemann levels, threshold comparisons, shadow radii, and metric-read tolerances.*
7. *H records componentwise* hsame *transport for boundary point, threshold, visual shadow, hyperbolic metric, visual metric, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through boundary-point admission, Busemann-threshold comparison, visual-shadow exposure, hyperbolic-metric readback, visual-metric comparison, and real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHoroball}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroball}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, U, S, M, V, R, H, C, P, N$. It has no coordinate for quotient ideal-boundary equality, a selected global geodesic ray, a Patterson–Sullivan measure class, a completed hyperbolic space, or physical horizon geometry.

Theorem 7843.2 (Hyperbolic horoball NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHoroball}^{\uparrow}$ *carrier*

$$\mathsf{HyperbolicHoroball}(B, U, S, M, V, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicHoroball}^{\uparrow}}$ *surface consists exactly of carrier admission for the boundary-point row, Busemann-threshold row, visual-shadow row,* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row,* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *comparison row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Every accepted horoball consumer must read B and U before reading the shadow, metric, visual-metric, real-seal, transport, replay, provenance, and naming rows. No obligation is stored in quotient boundary equality, selected geodesic representatives, measure classes at infinity, completed hyperbolic spaces, or physical horizon geometry.

Proof. Carrier admission is projection from Definition 7843.1. The accepted packet displays the boundary point, threshold, shadow, metric, visual metric, real seal, transport, replay, provenance, and naming rows.

The boundary and threshold rows determine the horoball side. A consumer that omits either B or U has not read the displayed Busemann boundary end and level comparison, so it is outside the accepted carrier.

The rows S, M, V, R then expose the visible geometry. The boundary-shadow row records the finite visual shadow, the metric row supplies distance readback, the visual-metric row compares the shadow at the same boundary end, and the real row seals the threshold and tolerance data.

The structural rows H, C, P, N only transport, replay, source, and name this finite route. Since the excluded global objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7843.3 (Hyperbolic horoball shadow route)**.** *Every visual-shadow read exported by an accepted* $\mathsf{HyperbolicHoroball}^{\uparrow}$ *packet factors through*

$$B, U \longmapsto S, M, V \longmapsto R, H, C, P, N.$$

Thus the boundary-point and Busemann-threshold rows are present before the horoball can expose a boundary shadow, hyperbolic metric read, visual metric comparison, or real-sealed radius.

Proof. Start with the carrier of Definition 7843.1. The row B names the Busemann boundary end and U names the real-sealed threshold level, so these rows are the only source of horoball membership data.

Next consume S, M, V. The visual-shadow row is read from the same boundary end, the hyperbolic-metric row supplies the metric readback, and the visual-metric row compares the resulting shadow window with the displayed visual metric surface.

Finally consume R, H, C, P, N. The real seal records the threshold and tolerance data, while transport, replay, provenance, and local naming preserve the same order. Since no selected geodesic ray or completed boundary quotient appears in the carrier, every accepted visual-shadow read follows the displayed finite route. □

Falsifiable prediction: Any horoball consumer that exposes a visual shadow without first reading B, U, S, M, V fails to read the accepted $\mathsf{HyperbolicHoroball}^{\uparrow}$ packet.

Independence witness: hyperbolic-metric, hyperbolic-busemann-boundary, hyperbolic-boundary-shadow, hyperbolic-visual-metric, real seal: none is bijective to $\mathsf{HyperbolicHoroball}^{\uparrow}$ because this packet binds boundary end, Busemann threshold, visual shadow, metric read, visual metric comparison, Real sealing, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHoroball}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHoroball}^{\uparrow}$ packet is generated from a boundary-point row, Busemann-threshold row, visual-shadow row, HyperbolicMetric row, HyperbolicVisualMetric row, Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroball}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7843.1, Definition 7843.2, and Definition 7843.3 seed the finite horoball route over boundary point, Busemann threshold, visual shadow, hyperbolic metric, visual metric, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroball}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal-boundary equality, selected geodesic representatives, measure classes at infinity, completed hyperbolic spaces, physical horizon geometry, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-point exactness, Busemann-threshold exactness, shadow-route exactness, HyperbolicMetric and HyperbolicVisualMetric handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicHoroballUp.

Chapter 7844

A Concrete Naming Certificate for $\mathsf{HamiltonJacobiAction}^{\uparrow}$

Following Evans and Arnold, the $\mathsf{HamiltonJacobiAction}^{\uparrow}$ packet records the finite action-functional surface that sits between variational mechanics and Hamilton-Jacobi readback. It reads the Lagrangian action row of chapter 337, the Hamilton-Jacobi row of chapter 7663, the Hamilton-Jacobi-Bellman row of chapter 7805, the routed dynamics rows of chapter 195 and chapter 193, the scalar readback of chapter 38, and the linear state displays of chapter 49 and chapter 47. The packet names action data for a displayed trajectory window; it is not a viscosity-solution theorem, not a global minimizer existence result, not a selected optimal-control policy, and not an ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Definition 7844.1 (Hamilton-Jacobi action carrier)**.** *A* $\mathsf{HamiltonJacobiAction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, T, Q, V, L, S, G, J, B, D, O, M, R, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{VecSpace}^{\uparrow}$ *state row and T is the finite time-window row for the displayed path segment.*
2. *Q is the path or configuration row exposed by* $\mathsf{LagrangianMechanics}^{\uparrow}$*, and V is the velocity or tangent readback row.*
3. *L is the displayed Lagrangian row and S is the action-accumulation row sealed only through finite* $\mathsf{Real}^{\uparrow}$ *data.*
4. *G is the endpoint-gradient or momentum readback row consumed by the Hamilton-Jacobi interface.*
5. *J is the* $\mathsf{HamiltonJacobi}^{\uparrow}$ *Hamiltonian-readback row and B is the* $\mathsf{HamiltonJacobiBellman}^{\uparrow}$ *control-facing sibling row.*
6. *D and O are the* $\mathsf{DynSystem}^{\uparrow}$ *and* $\mathsf{Ode}^{\uparrow}$ *route rows that tie the path segment to displayed dynamics.*
7. *M is the* $\mathsf{Mat}^{\uparrow}$ *coefficient row used when the action request is specialized to a finite linear or quadratic display.*
8. *R is the componentwise* hsame *transport row for $X, T, Q, V, L, S, G, J, B, D, O, M$.*

9. *C records displayed* Cont *replay through the route*

$$Q, V, L, D, O \longmapsto S.$$

10. *P is the* Pkg *provenance over* HamiltonJacobiAction$^{\uparrow}$, HamiltonJacobi$^{\uparrow}$, HamiltonJacobiBellman$^{\uparrow}$, LagrangianMechanics$^{\uparrow}$, DynSystem$^{\uparrow}$, Ode$^{\uparrow}$, Real$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{HamiltonJacobiAction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, Q, V, L, S, G, J, B, D, O, M, R, C, P, N$. It has no coordinate for a selected minimizer, a host path space, classical differentiability of all trajectories, viscosity uniqueness, PDE regularity, quotient equality of actions, or ambient real completeness.

Theorem 7844.2 (Hamilton-Jacobi action NameCert obligations)**.** *For every accepted* HamiltonJacobiAction$^{\uparrow}$ *carrier*

$$A = (X, T, Q, V, L, S, G, J, B, D, O, M, R, C, P, N),$$

the local NameCert$_{\text{HamiltonJacobiAction}^{\uparrow}}$ *surface consists exactly of carrier admission, path-window exposure, Lagrangian action exposure, Hamilton-Jacobi readback, Hamilton-Jacobi-Bellman sibling exposure, routed dynamics exposure, linear coefficient exposure, componentwise transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7844.1. The rows X, T, Q, V, L, S, G expose the finite state, path-window, Lagrangian, action, and endpoint-gradient data before any Hamilton-Jacobi read is available.

Read the sibling rows next. The Hamilton-Jacobi row J supplies Hamiltonian readback, while the Hamilton-Jacobi-Bellman row B supplies the control-facing sibling surface. The dynamics rows D, O keep the path segment tied to displayed DynSystem and Ode data.

The matrix row M is only a finite coefficient display for specialized linear or quadratic requests. Transport, replay, provenance, and local naming then pass through R, C, P, N.

Since no carrier coordinate contains a selected minimizer, host path space, global differentiability theorem, viscosity uniqueness theorem, PDE regularity theorem, quotient action equality, or real-completeness principle, none of those objects can be exported by the local NameCert surface. □

Theorem 7844.3 (Hamilton-Jacobi action HJB route)**.** *Every downstream consumer of an accepted* HamiltonJacobiAction$^{\uparrow}$ *packet that asks for a control-facing action read must pass through the displayed sibling route*

$$\text{LagrangianMechanics}^{\uparrow}, \text{DynSystem}^{\uparrow}, \text{Ode}^{\uparrow}, \text{HamiltonJacobi}^{\uparrow}, \text{HamiltonJacobiBellman}^{\uparrow} \longmapsto \text{HamiltonJacobiAction}^{\uparrow}.$$

The route binds finite Lagrangian action data, dynamics rows, Hamiltonian readback, and HJB-facing control rows without converting the packet into an optimal-control existence theorem or a global PDE regularity theorem.

Proof. Apply Definition 7844.2. The Lagrangian action rows Q, V, L, S, the dynamics rows D, O, the Hamilton-Jacobi row J, and the HJB row B are separate displayed rows of the same accepted carrier.

Use R, C, P, N to preserve that row inventory under transport, replay, provenance, and local naming. These structural rows do not add a host minimizer, a selected policy, a viscosity uniqueness theorem, or an ambient completion principle.

Thus a consumer reaches the control-facing action read only through the displayed finite route, and the refused analytic objects remain outside the packet. □

Falsifiable prediction: A claimed $\mathsf{HamiltonJacobiAction}^{\uparrow}$ read that reaches a control-facing action row without displaying the LagrangianMechanics, DynSystem, Ode, HamiltonJacobi, and HamiltonJacobiBellman sibling rows refutes the carrier.

Independence witness: hamilton-jacobi, hamilton-jacobi-bellman, lagrangianmechanics, dynsystem, ode, real, matrix, vecspace: no listed sibling is bijective to $\mathsf{HamiltonJacobiAction}^{\uparrow}$ because only this packet binds finite action accumulation, endpoint-gradient readback, Hamilton-Jacobi readback, HJB-facing control exposure, dynamics routes, coefficient display, transport, replay, provenance, and local naming as one action-functional packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HamiltonJacobiAction}^{\uparrow}}$.

Constructive story: A $\mathsf{HamiltonJacobiAction}^{\uparrow}$ packet is generated from finite state, time-window, path, velocity, Lagrangian, action, endpoint-gradient, HamiltonJacobi, HamiltonJacobiBellman, DynSystem, Ode, Matrix, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HamiltonJacobiAction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7844.1, Definition 7844.2, and Definition 7844.3 seed the finite Hamilton-Jacobi action surface over Lagrangian mechanics, dynamics, Hamilton-Jacobi, and HJB rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HamiltonJacobiAction}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.HamiltonJacobiActionUp`

Bridge status: none

Not claimed: This chapter does not claim selected minimizers, optimal-control existence, viscosity uniqueness, global PDE regularity, classical differentiability of all paths, quotient equality of actions, ambient real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lagrangian action exposure, Hamilton-Jacobi and HJB exposure, dynamics exposure, coefficient exposure, transport, replay, provenance, and local naming for BEDC.Derived.HamiltonJacobi

Chapter 7845

A Concrete Naming Certificate for $\mathsf{EilenbergMooreCategory}^{\uparrow}$

Following Mac Lane's categorical account of monads and the Eilenberg–Moore construction, the $\mathsf{EilenbergMooreCategory}^{\uparrow}$ packet names the category of displayed algebra objects for a displayed monad. It sits after the categorical carrier of chapter 61, the functorial transport surface of chapter 62, the monad packet of chapter 108, the Kleisli completion category of chapter 3826, and the completion-algebra surface of chapter 2139. The chapter records the finite NameCert rows by which monad algebras and their homomorphisms become a category-level consumer interface.

The packet is the algebra-object companion to the Kleisli category route. It does not identify algebras by quotient equality, select representatives, construct a host category of algebras, or import ambient equality of algebra maps.

Definition 7845.1 (Eilenberg–Moore category carrier)**.** *An* $\mathsf{EilenbergMooreCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, T, A, S, U, M, H, C, R, P, N)$$

with the following displayed rows:

1. *B is the base* $\mathsf{Cat}^{\uparrow}$ *row, exposing object and hom histories before any algebra row is read.*
2. *T is the* $\mathsf{Monad}^{\uparrow}$ *row on the base category, with its endofunctor, unit, and multiplication data displayed as finite BEDC carrier rows.*
3. *A is the algebra-object row, listing the displayed object on which the monad acts.*
4. *S is the structure-map row from the monad-applied object back to the algebra object.*
5. *U is the unit-law row, comparing the unit route followed by S with the displayed identity route on the algebra object.*
6. *M is the multiplication-law row, comparing the multiplication route followed by S with the route that first applies the monad to S and then applies S.*
7. *H is the algebra-homomorphism row, recording maps between displayed algebra objects whose square with the two structure maps is certified by the same finite carrier route.*
8. *C is the composition ledger for algebra homomorphisms, including identity homomorphisms and associative composition as displayed* $\mathsf{Cat}^{\uparrow}$ *and* $\mathsf{Monad}^{\uparrow}$ *route comparisons.*
9. *R records componentwise* hsame *transport and displayed* Cont *replay for B, T, A, S, U, M, H, C.*

10. *P is the* Pkg *provenance over* $\mathsf{EilenbergMooreCategory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{Monad}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$, $\mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EilenbergMooreCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, T, A, S, U, M, H, C, R, P, N$. It has no coordinate for quotient algebras, selected representatives, host equality of algebra maps, ambient category objects, or a collapse of the Eilenberg–Moore route into the Kleisli route.

Theorem 7845.2 (Eilenberg–Moore category NameCert obligations)**.** *For every accepted* $\mathsf{EilenbergMooreCategory}^{\uparrow}$ *carrier*

$$E = (B, T, A, S, U, M, H, C, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EilenbergMooreCategory}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the base category, monad, algebra object, structure map, unit law, multiplication law, algebra homomorphism, composition ledger, transport, provenance, and local naming rows;*
2. *monad-source exposure, requiring the base category and monad rows B, T before any algebra object or homomorphism can be inspected;*
3. *algebra-object exposure, requiring A and S to display the object and its structure map before either law row is public;*
4. *law exactness, requiring U and M to compare only the displayed unit and multiplication routes of the same monad row;*
5. *homomorphism and composition exactness, requiring H and C to keep algebra maps, identities, and composites inside the displayed algebra rows;*
6. *structural exhaustion, requiring R, P, N to transport, replay, source, and name the listed rows without adding quotient algebras, selected representatives, host equality of algebra maps, ambient category objects, or a Kleisli-category identification.*

No obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7845.1. The accepted packet exposes exactly the base category, monad, algebra-object, structure-map, law, homomorphism, composition, transport, provenance, and local naming rows.

Monad-source exposure is read from B and T. Since the monad is displayed over the same base category row, an algebra consumer cannot begin from a separate host category or from an unrecorded endofunctor.

Algebra-object exposure is read from A and S. The object row is available before the structure-map row, and the structure map is the only coordinate that turns the monad-applied object back into the algebra object.

Law exactness is read from U and M. The unit law compares the displayed unit route with the identity route, while the multiplication law compares the displayed multiplication route with the two-step action route; both comparisons use the same B, T, A, S rows.

Finally H, C, R, P, N bind maps, composites, transport, provenance, and local naming to the same finite packet. Because no other coordinate occurs in the carrier, the listed rows exhaust the NameCert obligation surface. □

Theorem 7845.3 (Eilenberg–Moore category algebra route)**.** *Every consumer that reads an accepted* $\mathsf{EilenbergMooreCategory}^{\uparrow}$ *packet as a category of algebra objects factors through the route*

$$\mathsf{Cat}^{\uparrow},\ \mathsf{Functor}^{\uparrow},\ \mathsf{Monad}^{\uparrow},\ \mathsf{NatTrans}^{\uparrow},\ \mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow},\ \mathsf{CauchyCompletionKleisliCategory}^{\uparrow}.$$

The object side reads B, T, A, S, U, M *before any algebra homomorphism is public, and the arrow side reads* H, C, R, P, N *before identities or composites can be exported. Thus the packet gives a finite category-level algebra interface without quotient categories, selected algebra representatives, host equality of algebra maps, ambient category equality, or identification with the Kleisli category.*

Proof. Start with the object side. Projection of the carrier supplies B and T as the base category and monad rows, then A and S as the displayed algebra object and structure map.

The law rows are then forced in order. The unit row U compares the unit action against the displayed identity route, and the multiplication row M compares the two monad-action schedules against the same structure map.

Turn to arrows. The homomorphism row H reads maps only after their source and target algebra rows have exposed the same B, T, A, S, U, M data, and the composition ledger C records identities and composites as displayed category and monad route comparisons.

The dependency route is therefore the advertised sibling route: the base categorical rows come from $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{Monad}^{\uparrow}$, and $\mathsf{NatTrans}^{\uparrow}$, while the completion-facing algebra and Kleisli consumers cite chapter 2139 and chapter 3826. The structural rows R, P, N add transport, replay, provenance, and local naming only, so no quotient or host equality coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{EilenbergMooreCategory}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EilenbergMooreCategory}^{\uparrow}$ packet is generated from a base category row, a monad row, an algebra-object row, a structure-map row, unit and multiplication law rows, an algebra-homomorphism row, a composition ledger, componentwise hsame and Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EilenbergMooreCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7845.1, Definition 7845.2, and Definition 7845.3 seed the finite category-level Eilenberg–Moore algebra interface for monad and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EilenbergMooreCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient categories of algebras, selected representatives, host equality of algebra maps, ambient category equality, Lean verification, bridge checking, or an identification with the Kleisli category.

Upgrade path: Obligation closure requires checked carrier admission, monad-source exposure, algebra-object exposure, unit-law exactness, multiplication-law exactness, homomorphism and composition exactness, structural replay, and ledger non-escape for BEDC.Derived.EilenbergMooreCategoryUp.

Chapter 7846

A Concrete Naming Certificate for BishopRegularCauchyTailCommonRefinement$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopRegularCauchyTailCommonRefinement$^{\uparrow}$ packet records the finite common-refinement surface used when two regular Cauchy rational names must be read through one late tail before a real-name consumer can compare or transport them. It sits over the regular rational source rows of chapter 301, the finite observation schedules of chapter 126, the dyadic tolerance ledger of chapter 291, the common-tail filter basis of chapter 2129, the tail-comparison route of chapter 2146, the tail-functor handoff of chapter 3602, and the terminal real-name seal of chapter 38. The packet names a displayed late-tail refinement for two Bishop regular Cauchy names; it is not quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 7846.1 (Bishop regular Cauchy tail common-refinement carrier)**.** *A* BishopRegularCauchyTailCommonR *carrier is a finite* BHist *packet*

$$K = (X, Y, S, T, R, D, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* RegSeqRat$^{\uparrow}$ *source-name rows. Each source is exposed only through its displayed regular rational reads.*
2. *S is the shared* StreamName$^{\uparrow}$ *schedule. It records the finite observation windows, inspected indices, and synchronized late-window requests used for both sources.*
3. *T is the common tail-cut row. It records the late cut, cofinal refinement requests, and membership witnesses saying that both sources are read beyond the same displayed tail.*
4. *R is the regular readback row. It records the rational observations of X and Y at the windows specified by S and T.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger. It records the requested radius, tail slack, enclosure budget, and finite-window exhaustion consumed by the refinement.*
6. *F is the* RegularCauchyCommonTailFilter$^{\uparrow}$ *row. It reads T as a finite common-tail basis and exports only displayed basis and cofinality data.*
7. *Q is the* RegularCauchyTailComparison$^{\uparrow}$ *row. It records the comparison-facing late-window route obtained after X, Y, S, T, R, D, F have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the shared schedule, common tail, readback, dyadic ledger, filter row, and comparison row have been displayed.*

9. *H records componentwise* hsame *transport for the two source rows, shared schedule, common tail, readback, dyadic ledger, filter row, comparison row, real seal, replay, provenance, and local naming row.*

10. *C records displayed* Cont *replay of the route*

$$X, Y, S, T, D \longmapsto R \longmapsto F \longmapsto Q \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$, $\mathsf{RegularCauchyTailComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these thirteen rows. Its classifier compares accepted packets only through $X, Y, S, T, R, D, F, Q, E, H, C, P, N$ and has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7846.2 (Bishop regular Cauchy tail common-refinement NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}$ *carrier*

$$K = (X, Y, S, T, R, D, F, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy source rows, shared stream schedule, common tail cut, regular readback, dyadic tolerance ledger, common-tail filter row, comparison row, terminal real seal, transport, replay, provenance, and local naming row;*

2. *shared-schedule discipline, requiring both source rows to be inspected through the same S row before either a comparison route or real seal is available;*

3. *common-tail discipline, requiring T to display the late cut, cofinal refinement requests, and membership witnesses consumed by both sources;*

4. *tolerance and readback exactness, requiring R and D to expose only the rational observations and dyadic budgets read from the displayed source rows, schedule, and common tail;*

5. *filter and comparison non-escape, requiring F and Q to remain downstream of X, Y, S, T, R, D and to export no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*

6. *real-seal discipline, requiring E to be read only after the common-tail filter and comparison rows have both been displayed;*

7. *transport, replay, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise.*

No Bishop regular Cauchy tail common-refinement obligation is stored outside these rows.

Proof. Project the accepted packet $K = (X, Y, S, T, R, D, F, Q, E, H, C, P, N)$. The carrier exposes exactly the two source rows, shared schedule, common tail, regular readback, dyadic ledger, filter row, comparison row, terminal real seal, and structural rows.

The shared schedule and common tail are finite. The row S fixes the inspected windows for both source names, while T fixes the late cut and cofinal refinement requests through which both sources must be read.

The readback and tolerance rows are tied to that same tail. The row R supplies only the regular rational observations visible at the displayed windows, and D supplies only the dyadic radius, slack, enclosure budget, and finite-window exhaustion attached to those observations.

The filter and comparison rows remain downstream. The row F reads the common tail as a finite common-tail filter basis, and Q reads the resulting late-window route as the comparison-facing surface.

Finally E, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since none of the excluded host objects has a carrier coordinate, replay edge, provenance entry, or local naming row, the listed obligations exhaust the local NameCert surface. □

Theorem 7846.3 (Bishop regular Cauchy tail common-refinement stability)**.** *Let $K = (X, Y, S, T, R, D, F, Q, E, H$ be an accepted* $\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}$ *carrier. Under componentwise* hsame *transport of the displayed rows $X, Y, S, T, R, D, F, Q, E, C, P, N$, the accepted common-refinement packet remains accepted with the same shared schedule, common tail, tolerance ledger, filter row, comparison row, and real-seal boundary. The transported packet still has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the transport row H. It is componentwise: it identifies the two source rows, shared schedule, common tail, readback, dyadic ledger, filter row, comparison row, seal row, replay, provenance, and local naming row with their transported copies.

Replay the carrier predicate after transport. Each accepted coordinate is replaced by the corresponding hsame coordinate, so the shared schedule S, tail cut T, tolerance ledger D, filter row F, comparison row Q, and seal boundary E retain the same displayed dependency order.

The structural rows C, P, N are transported along the same route. Therefore the local NameCert reads the transported packet through exactly the same finite coordinates.

Because transport only moves existing displayed rows and does not add a new coordinate, the refused quotient, selection, choice, host-equality, trichotomy, and completeness objects remain absent. □

Theorem 7846.4 (Bishop regular Cauchy tail common-refinement L10 handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing equality, comparison, or completion consumer of an accepted* $\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y, S, T, D \longmapsto R \longmapsto F \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives two displayed regular Cauchy source names, one shared StreamName schedule, one common tail cut, regular rational readback, DyadicRatCore tolerance accounting, a common-tail filter row, a comparison row, the terminal RealUp seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted Real-facing consumer read. The source rows are X and Y, and the carrier requires both rows to be observed through the common schedule S and late tail T.

The dyadic ledger D fixes the precision budget attached to that shared observation. The regular readback row R then supplies the rational observations of the two source names at the displayed windows.

The common-tail filter row F reads the same late cut as a finite basis, and the comparison row Q reads the resulting route as the late-window comparison surface. Only after both rows are displayed can the terminal seal row E be reached.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Induction over the displayed route order proves that a consumer cannot skip from the source rows to the seal: the base stage reads X, Y, S, T, D, the readback stage reads R, the filter stage reads F, the comparison stage reads Q, and the terminal stage reads E, H, C, P, N.

Since the carrier has no coordinate outside this route, the consumer cannot obtain quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one shared StreamName schedule, one common tail cut, regular readback, a DyadicRatCore tolerance ledger, a RegularCauchyCommonTailFilter row, a RegularCauchyTailComparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7846.1, Definition 7846.2, Definition 7846.3, and Definition 7846.4 seed the finite Bishop regular Cauchy tail common-refinement route over shared StreamName schedules, RegSeqRat source rows, DyadicRatCore tolerances, common-tail filters, tail comparisons, RealUp seals, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailCommonRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-schedule discipline, common-tail discipline, tolerance and readback exactness, filter and comparison non-escape, real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyTailCommonRefinementUp.

Chapter 7847

A Concrete Naming Certificate for $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on local compactness in metric spaces, the $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ packet records the finite certificate surface for a pointwise compact-neighbourhood witness around a displayed metric source. It sits between the metric source of chapter 57 and the compact metric surface of chapter 129: the packet names a local compact ball handoff, not an open-cover compactness theorem, a choice-selected neighbourhood family, a host topology, or ambient completeness.

Definition 7847.1 (Locally compact metric space carrier)**.** *A* $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, K, r, B, J, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the points, distance requests, symmetry, triangle, and zero-distance comparison used by the local certificate.*
2. *K is the compact-neighbourhood witness row, naming the local* $\mathsf{CompactMetric}^{\uparrow}$ *packet that contains the displayed point after the radius and containment rows have been checked.*
3. *r is the* $\mathsf{PositiveReal}^{\uparrow}$ *radius row, recording that the inspected point has a positive metric radius available to the local packet.*
4. *B is the local ball-containment row, displaying that the metric ball determined by M and r is contained in the compact-neighbourhood witness K.*
5. *J is the compact-ball handoff row from the metric source and positive-radius request into the local compact metric certificate.*
6. *H records componentwise* hsame *transport for the metric source, compact-neighbourhood witness, radius, containment, handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of every local compactness read through*

$$M, r, B, K \longmapsto J.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, r, B, J, H, C, P, N$. It has no coordinate for open-cover compactness, choice-selected neighbourhoods, host topological interiors, ambient completion, quotient metric equality, Lean verification, or a standard bridge.

Theorem 7847.2 (Locally compact metric space compact-ball handoff)**.** *For every accepted* $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ *carrier*

$$L = (M, K, r, B, J, H, C, P, N),$$

each local compactness read first exposes the $\mathsf{Met}^{\uparrow}$ *source M, the* $\mathsf{PositiveReal}^{\uparrow}$ *radius row r, and the local ball-containment row B, and only then hands the request to the* $\mathsf{CompactMetric}^{\uparrow}$ *witness K through J. The handoff preserves componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}$ *row. It exports no open-cover compactness, choice-selected neighbourhood family, host topology, quotient metric equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 7847.1. The metric source M, positive radius r, ball-containment row B, compact-neighbourhood witness K, and handoff row J are the only mathematical rows available to a local compactness consumer.

The read order is forced by the rows themselves. A compact witness can be consumed only after the point and radius have been displayed in the metric source and the containment row has placed the resulting ball inside the compact neighbourhood.

Transport and replay are componentwise. The hsame row H keeps the same metric source, radius, containment, and compact witness, while the displayed Cont row C repeats the same route for later consumers.

Finally, P and N source and name only the displayed packet. Since no excluded coordinate appears in the carrier, the compact-ball handoff is the whole local compactness surface. □

Theorem 7847.3 (Locally compact metric space NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ *carrier $L = (M, K, r, B, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}$ *surface consists exactly of metric-source admission, compact-neighbourhood witness exposure, positive-radius admission, local ball containment, compact-ball handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7847.1. The accepted packet displays the metric source, compact-neighbourhood witness, radius, containment, handoff, transport, replay, provenance, and naming rows.

The compactness obligation is local. It is read only through K after M, r, B have been displayed, so the packet does not assert a global open-cover compactness theorem or select a neighbourhood family by choice.

The structural rows H, C, P, N preserve and name the same finite route. No remaining row stores host topology, quotient metric equality, ambient completion, Lean verification, or a bridge credential, so the listed rows exhaust the NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ packet displays a $\mathsf{Met}^{\uparrow}$ source, a $\mathsf{PositiveReal}^{\uparrow}$ radius row, a local ball-containment row, a $\mathsf{CompactMetric}^{\uparrow}$ witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row before any compact-neighbourhood read is public.

Independence witness: metric and compactmetric: neither sibling is bijective to $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ because this packet jointly stores the pointwise positive-radius row and the local ball-containment handoff from chapter 57 into chapter 129.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactMetricSpace}^{\uparrow}$ packet is generated from a MetricSpace source row, a CompactMetric witness row, a positive-radius row, local ball containment, a compact-ball handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7847.1, Definition 7847.2, and Definition 7847.3 seed the finite local compactness route from MetricSpace rows to compact metric witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected neighbourhoods, host topology, quotient metric equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius admission, ball-containment exactness, compact-neighbourhood exposure, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocallyCompactMetricSpaceUp.

Chapter 7848

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailPasting}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and tail-equivalent representatives, the $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ packet records the finite shared-tail pasting surface used before a real-name seal is read. It sits beside the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite tail-pasting route; it is not quotient sequence equality, a selected limit, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7848.1 (Regular Cauchy tail pasting carrier)**.** *A* $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, R, \tau, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose late observations are pasted along a shared tail.*
2. *τ is the finite tail threshold row, naming the stage after which the two rows are compared through the same observed tail window.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *shared-tail window row containing the inspected rational observations after τ.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for every tail comparison and pasted readback.*
5. *Q is the pasted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from L, R, τ, W, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after the shared-tail window, dyadic ledger, and pasted regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the source rows, tail threshold, shared window, tolerance ledger, pasted readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the tail-pasting route*

$$L, R, \tau, W, D \longmapsto Q \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailPasting}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, R, \tau, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, host sequence equality, host real equality, countable choice, ambient completeness, Lean verification, or a standard bridge.

Theorem 7848.2 (Regular Cauchy tail pasting shared-tail handoff)**.** *For every accepted* $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ *carrier*

$$T = (L, R, \tau, W, D, Q, E, H, C, P, N),$$

each Real-completion consumer reads the shared-tail route in the order

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer first displays the tail threshold τ and shared $\mathsf{StreamName}^{\uparrow}$ *window W, then the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows L, R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the pasted readback Q, and finally the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The handoff preserves componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}$ *row, and it exports no quotient stream equality, selected limit, host sequence equality, countable-choice schedule, or ambient Real completeness coordinate.*

Proof. Project the carrier of Definition 7848.1. The mathematical rows are exactly the two source readbacks, the tail threshold, the shared window, the dyadic tolerance ledger, the pasted readback, and the terminal real seal.

The shared-tail route begins with τ and W. These rows state which finite tail is visible and which rational observations are inspected before any pasted readback can be consumed.

Next read L, R, D, Q. The two regular rational rows supply the source observations, the dyadic ledger records the comparison budgets on the shared tail, and the pasted row Q is assembled only from those displayed rows.

Only after Q is public can the terminal seal E be read. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

Since the carrier has no quotient-stream, selected-limit, host-sequence, choice, or ambient-completeness coordinate, the handoff is confined to the listed StreamName, RegSeqRat, DyadicRatCore, RealUp, and structural rows. □

Theorem 7848.3 (Regular Cauchy tail pasting NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailPasti}$ *carrier $T = (L, R, \tau, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}$ *surface consists exactly of left and right RegSeqRat source admission, finite tail-threshold exposure, shared StreamName window exposure, DyadicRatCore tolerance exactness, pasted RegSeqRat readback, RealUp seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7848.1. The accepted packet displays the two regular rational source rows, the finite tail threshold, the shared window, the tolerance ledger, the pasted readback, the Real seal, and the structural rows.

The tail-pasting obligation is finite. The threshold and window rows determine the only observations available to the pasted readback, while the dyadic ledger supplies all comparison budgets used on that tail.

The Real seal is downstream of the pasted readback. Thus a consumer reaches $\mathsf{Real}^{\uparrow}$ only after the StreamName, RegSeqRat, and DyadicRatCore rows have been displayed.

Finally, H, C, P, N transport, replay, source, and name only these rows. Because the carrier contains no quotient equality, selected representative, selected limit, host equality, countable-choice schedule, ambient completeness, Lean verification, or bridge coordinate, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ packet displays two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a finite tail threshold, a shared $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a pasted readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any Real seal read is public.

Independence witness: streamname, regseqrat, dyadicratcore, and real: no listed sibling is bijective to $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ because this packet jointly stores the shared-tail threshold, pasted readback, and terminal RealUp seal route as one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailPasting}^{\uparrow}$ packet is generated from left and right RegSeqRat rows, a finite tail threshold, a shared StreamName window, a DyadicRatCore tolerance ledger, a pasted RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7848.1, Definition 7848.2, and Definition 7848.3 seed the finite shared-tail pasting route before terminal RealUp seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailPasting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, host sequence equality, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, tail-threshold exposure, shared-window exactness, dyadic tolerance exactness, pasted readback stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailPastingUp.

Chapter 7849

A Concrete Naming Certificate for $\mathsf{LebesgueConstant}^{\uparrow}$

Following Lebesgue and Zygmund on Fourier partial sums, the $\mathsf{LebesgueConstant}^{\uparrow}$ packet records the finite norm-growth data of trigonometric partial-sum operators as BEDC NameCert rows. It sits over the Fourier coefficient windows of chapter 124, the Dirichlet-kernel row of chapter 2311, the circle phase samples of chapter 58, the integral normalization route of chapter 123, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the downstream approximation boundary of chapter 463. The packet names a finite harmonic-analysis growth surface, not a host L^1 operator norm, not Haar measure, and not a quotient function-space object.

Definition 7849.1 (LebesgueConstant carrier)**.** *A* $\mathsf{LebesgueConstant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, D, S, I, N, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *F is the Fourier coefficient-window row naming the finite frequency interval whose partial sum is inspected.*
2. *D is the Dirichlet-kernel row supplying the finite convolution kernel for that window.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-sample row used to evaluate the displayed kernel and partial-sum reads.*
4. *I is the* $\mathsf{Integral}^{\uparrow}$ *normalization row for the finite absolute-kernel read.*
5. *N is the finite norm-growth ledger recording the inspected Lebesgue-constant value window.*
6. *W is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance window bounding the displayed norm read.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite rational approximations.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed norm-growth datum.*
9. *H records componentwise* hsame *transport for $F, D, S, I, N, W, R, E, C, P, L$.*
10. *C records displayed* Cont *replay from a partial-sum request through the Dirichlet-kernel and integral rows to the norm ledger.*
11. *P is the* Pkg *provenance over the Fourier, Dirichlet-kernel, circle, integral, dyadic, regular-rational, Real, transport, replay, and NameCert rows.*

12. *L is the local* $\mathsf{NameCert}_{\mathsf{LebesgueConstant}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying all twelve rows. It has no coordinate for Haar measure, host L^1 spaces, quotient function equality, selected Fourier representatives, ambient completeness, or an unrestricted Fourier-convergence theorem.

Lean4 (statement) `BEDC.Derived.LebesgueConstantUp`

Theorem 7849.2 (LebesgueConstant NameCert obligations)**.** *For the carrier of Definition 7849.1, the local* $\mathsf{NameCert}_{\mathsf{LebesgueConstant}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier admission for $K = (F, D, S, I, N, W, R, E, H, C, P, L)$;*
2. *Fourier-window and Dirichlet-kernel exposure through F and D;*
3. *phase-sample and integral-normalization exposure through S and I;*
4. *norm-growth, dyadic tolerance, regular rational readback, and Real sealing through N, W, R, E;*
5. *componentwise transport, replay, provenance, and local naming through H, C, P, L.*

Every accepted norm-growth read must display these rows before a downstream approximation consumer may cite a Lebesgue-constant datum.

Proof. Project the carrier of Definition 7849.1. The displayed tuple contains exactly the Fourier window, Dirichlet kernel, phase samples, integral normalization, norm ledger, dyadic tolerance, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

The first two obligations bind the source of the partial-sum read. The Fourier window F names the finite frequency interval, and the Dirichlet row D supplies the only kernel through which the partial-sum norm can be inspected.

The phase and normalization rows S, I then make the finite absolute-kernel read visible before any norm-growth entry is accepted. Rows N, W, R, E keep the numeric read inside the dyadic, regular-rational, and Real sealing route.

Finally H, C, P, L preserve, replay, source, and name the same packet. Since the carrier has no row for Haar measure, host L^1 spaces, quotient function equality, selected Fourier representatives, or ambient completeness, the five listed obligations exhaust the local NameCert surface. □

Theorem 7849.3 (LebesgueConstant Dirichlet norm handoff)**.** *Every Fourier partial-sum norm read exported by an accepted* $\mathsf{LebesgueConstant}^{\uparrow}$ *packet factors through the Dirichlet-kernel and integral rows*

$$F, D, S, I \longmapsto N, W, R, E$$

before any $\mathsf{UniformLimit}^{\uparrow}$ *handoff can consume the norm-growth datum. The handoff does not export Haar measure, host L^1 spaces, quotient function equality, selected Fourier representatives, or ambient completeness.*

Proof. Begin with a consumer of the norm ledger N. Carrier admission places N downstream of the Fourier window F, Dirichlet kernel D, circle phase samples S, and integral normalization I.

The replay row C records the only displayed route from a partial-sum request to the norm-growth read. Thus the request must first expose the finite coefficient window, then read the Dirichlet-kernel row and the integral-normalization row before the dyadic tolerance window W and regular-rational readback R are available.

The Real seal E is reached only after N, W, R have been displayed. Transport and provenance through H and P keep that same route attached to the local name L, so a $\mathsf{UniformLimit}^{\uparrow}$ consumer receives only the finite norm-growth datum produced by the Dirichlet-kernel and integral rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueConstant}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueConstant}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7849.1, Definition 7849.2, and Definition 7849.3 seed the finite Fourier partial-sum norm-growth packet through Dirichlet-kernel, integral, dyadic, regular-rational, Real, and uniform-limit-facing rows.

Constructive story: A $\mathsf{LebesgueConstant}^{\uparrow}$ packet is generated from Fourier coefficient-window data, Dirichlet-kernel rows, circle phase samples, integral-normalization rows, a finite norm-growth ledger, dyadic tolerance data, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueConstant}\uparrow}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.LebesgueConstantUp`

Bridge status: none

Not claimed: This chapter does not claim Haar measure, host L^1 spaces, quotient function equality, selected Fourier representatives, ambient completeness, Lean-checked obligations, or a standard bridge.

Upgrade path: Obligation closure requires Lean-checked carrier admission, NameCert surface, Dirichlet norm handoff, and ledger non-escape rows for $F, D, S, I, N, W, R, E, H, C, P, L$.

Chapter 7850

A Concrete Naming Certificate for $\mathsf{RichmanCompletion}^{\uparrow}$

Following Bishop and Richman, the $\mathsf{RichmanCompletion}^{\uparrow}$ packet records a constructive completion neighbour for the L10 Real and Completion corridor as finite BEDC NameCert data. It consumes the dyadic arithmetic core of chapter 291, finite stream windows from chapter 126, regular rational readback from chapter 301, the terminal real-name seal of chapter 38, and the completion-functor handoff of chapter 7003. The packet names the inspected regular-Cauchy completion route; it does not quotient names, select limits, use countable choice, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, choose finite subcovers, or import open-cover compactness.

Definition 7850.1 (Richman completion carrier)**.** *A* $\mathsf{RichmanCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, E, L, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, recording dyadic endpoints, comparisons, and rational tolerance arithmetic used by the completion read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the inspected rational-name windows visible to a consumer.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row obtained from the displayed stream windows and dyadic tolerance data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after D, S, Q and the local comparison data have been displayed.*
5. *L is the located comparison row, answering each admitted rational cut request by finite inspection of D, S, Q before any terminal seal is consumed.*
6. *B is the dyadic radius-budget row, recording the finite radii and contraction budgets used to pass from regular-Cauchy data to the completion handoff.*
7. *H records componentwise* hsame *transport for the dyadic arithmetic, stream windows, regular readback, real seal, located comparison, radius budget, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same finite completion route.*

9. *P is the* Pkg *provenance over* $\mathsf{RichmanCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RichmanCompletion}^{\uparrow}}$ *row naming exactly this packet.*

The classifier compares accepted carriers only through $D, S, Q, E, L, B, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, selected limits, countable choice, host equality of real names, ambient $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, or selected finite subcovers.*

Lean4 ✓ `BEDC.Derived.RichmanCompletionUp.RichmanCompletionTasteGate single carrier alignment`

Theorem 7850.2 (Richman completion NameCert obligations)**.** *For every accepted* $\mathsf{RichmanCompletion}^{\uparrow}$ *carrier*

$$R = (D, S, Q, E, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RichmanCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic visibility, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{Real}^{\uparrow}$ *seal discipline, located comparison, dyadic radius-budget exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must pass through those rows before the terminal seal is available.*

Proof. Project the carrier of Definition 7850.1. The analytic rows D, S, Q, E, L, B expose the dyadic arithmetic, finite windows, regular-Cauchy readback, terminal Real seal, located comparison, and radius-budget data in the order named by the tuple.

Read the constructive route through the sibling chapters. The dyadic row D is the arithmetic source, the stream row S fixes the finite observations, and the regular row Q provides the Cauchy readback before E can be consumed.

The comparison and budget rows L, B keep the Richman-facing completion read local: each cut response and radius request is answered by visible finite data rather than by a quotient object or selected limiting representative.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for quotient completion equality, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, or selected finite subcovers, the listed rows exhaust the local NameCert surface. □

Theorem 7850.3 (Richman completion regular-Cauchy handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-functor-facing consumer of an accepted* $\mathsf{RichmanCompletion}^{\uparrow}$ *packet receives the finite handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BishopRegularCompletionFunctor}^{\uparrow}$$

through the displayed rows $D, S, Q, E, L, B, H, C, P, N$. The handoff is a regular-Cauchy completion route with located comparison and dyadic radius budgets; it is not a quotient completion, selected limit, countable-choice schedule, host real equality theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness claim, selected finite subcover, or open-cover compactness principle.*

Proof. Start with the source side. The rows D and S determine the dyadic tolerance cells and the finite rational-name windows available to the read.

Next pass through Q, E. The $\mathsf{RegSeqRat}^{\uparrow}$ row reads the displayed windows as regular-Cauchy data, and the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed only after that readback.

Then use L, B to route the Richman completion interface. The located comparison row answers admitted cut requests from the same finite data, while the dyadic radius-budget row records the radius schedule consumed by the completion-functor handoff.

The structural rows H, C, P, N preserve the sibling route and attach the local name. Thus any completion-facing consumer receives the displayed regular-Cauchy handoff and no quotient, selector, choice, host-equality, ambient-completeness, finite-subcover, or open-cover compactness coordinate. □

Falsifiable prediction: Any accepted $\mathsf{RichmanCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or a $\mathsf{BishopRegularCompletionFunctor}^{\uparrow}$ handoff without first reading the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, located comparison, dyadic radius-budget, hsame, Cont, Pkg, and local NameCert rows is outside the packet surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-completion-functor: no listed sibling is bijective to $\mathsf{RichmanCompletion}^{\uparrow}$ because this packet binds dyadic arithmetic, finite windows, regular-Cauchy readback, terminal Real sealing, located comparison, radius budgets, transport, replay, provenance, and local naming as one constructive completion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RichmanCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RichmanCompletion}^{\uparrow}$ packet is generated from DyadicRatCore arithmetic rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, located comparison, dyadic radius budgets, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RichmanCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7850.1, Definition 7850.2, and Definition 7850.3 seed the Richman constructive completion route over dyadic arithmetic, finite windows, regular-Cauchy readback, Real sealing, located comparison, radius budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RichmanCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient completion, quotient stream equality, selected limits, countable choice, selected finite subcovers, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic arithmetic coverage, finite-window readback, regular-Cauchy handoff, located comparison exactness, radius-budget stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RichmanCompletionUp.

Chapter 7851

A Concrete Naming Certificate for $\mathsf{RegularCauchyPruning}^{\uparrow}$

Following Bishop and Bridges on subsequences and regular Cauchy sequences, the $\mathsf{RegularCauchyPruning}^{\uparrow}$ packet records the finite certificate surface for deleting redundant observation requests from a regular Cauchy rational stream while preserving the terminal real-name seal. It sits after the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the tail restriction route of chapter 2084, the cofinal-subsequence surface of chapter 1998, and the terminal boundary of chapter 38. The packet is a non-quotient, choice-free pruning certificate for L10 Real and Completion consumers; it does not select a hidden subsequence, quotient streams, compare host real equality, or invoke ambient completeness.

Definition 7851.1 (Regular-Cauchy pruning carrier)**.** *A* $\mathsf{RegularCauchyPruning}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, W, \rho, T, Q, D, R, E, H, C, G, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regular Cauchy rational observations are being pruned.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger listing the inspected source requests and the retained requests after pruning.*
3. *ρ is the pruning schedule row. It records, for each displayed retained request, the source request that survives; it is finite on the inspected window and cofinal on the displayed thresholds.*
4. *T is the* $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ *tail row exposing the initial cutoff before any retained window is read.*
5. *Q is the* $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ *cofinal row exposing the monotone retained schedule after the tail cutoff.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the source, retained windows, tail row, cofinal row, and real seal.*
7. *R is the retained* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W, ρ, T, Q, D.*
8. *E is the inherited* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after the retained readback row has been displayed.*
9. *H records componentwise* hsame *transport for the source, windows, pruning schedule, tail row, cofinal row, tolerance ledger, retained readback, real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a pruning request through the same finite rows.*
11. *G is the* Pkg *provenance over* $\mathsf{RegularCauchyPruning}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailSubsequence}$ $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPruning}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, \rho, T, Q, D, R, E, H, C, G, N$. It has no coordinate for quotient stream equality, a hidden infinite selector, selected ambient limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 7851.2 (Regular-Cauchy pruning tail stability)**.** *Let*

$$P = (S, W, \rho, T, Q, D, R, E, H, C, G, N)$$

be an accepted $\mathsf{RegularCauchyPruning}^{\uparrow}$ *carrier. Every tail-facing or cofinal-subsequence consumer of P reads the same terminal* $\mathsf{Real}^{\uparrow}$ *seal E through the finite route*

$$S, W, \rho, T, Q, D, R, E, H, C, G, N.$$

Consequently pruning redundant observed requests preserves the displayed real-name boundary and supplies no quotient stream equality, selected limit, host real equality, countable choice, or ambient completeness principle.

Proof. Project the carrier of Definition 7851.1. The tail row T and cofinal row Q are displayed coordinates of the same packet as the source row S, window ledger W, pruning schedule ρ, dyadic ledger D, retained readback R, and seal E.

Read the tail route first. The row T exposes the initial cutoff through chapter 2084, so retained requests are not read until the tail window has been displayed.

Read the cofinal route next. The row Q exposes the retained monotone cofinal schedule through chapter 1998. The schedule row ρ is therefore a finite pruning ledger on the inspected windows, not an external choice function.

Finally read D, R, E. The dyadic tolerance ledger is shared by the source, tail, cofinal, and retained readback rows, and R is the only regular rational readback available before the real seal. Since H, C, G, N transport, replay, source, and name only these rows, a consumer cannot extract quotient stream equality, selected limits, host equality, countable choice, or ambient completeness from the pruning packet. □

Theorem 7851.3 (Regular-Cauchy pruning NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyPruning}^{\uparrow}$ *carrier*

$$P = (S, W, \rho, T, Q, D, R, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPruning}^{\uparrow}}$ *surface consists exactly of carrier admission for the source regular rational row, finite window ledger, pruning schedule, tail row, cofinal retained schedule, dyadic tolerance ledger, retained regular rational readback, real seal, componentwise transport, continuation replay, provenance, and local naming rows; tail stability through T; cofinal stability through Q; retained readback exactness through R; and ledger nonescape through H, C, G, N. No NameCert obligation is stored outside those displayed rows.*

Proof. Carrier admission is projection from Definition 7851.1. The packet displays exactly the source row, inspected and retained windows, pruning schedule, tail and cofinal routes, dyadic ledger, retained readback, Real seal, and structural rows.

Tail stability is Definition 7851.2. It ensures that the tail and cofinal consumers read the same seal through the retained finite route.

Retained readback exactness is read from R. The retained $\mathsf{RegSeqRat}^{\uparrow}$ row consumes the displayed windows, pruning schedule, tail row, cofinal row, and dyadic ledger before E is visible.

The structural rows exhaust the route. Componentwise transport, continuation replay, provenance, and local naming preserve only the displayed packet, so no hidden selector, quotient stream, selected limit, host equality, countable choice, ambient completeness, Lean verification, or bridge credential can serve as a local obligation. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyPruning}^{\uparrow}$ completion-facing read exposes the source StreamName windows, pruning schedule, tail row, cofinal row, dyadic tolerance ledger, retained RegSeqRat readback, and inherited RealUp seal before any consumer can inspect the pruned stream.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPruning}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyPruning}^{\uparrow}$ packet is generated from a source RegSeqRat row, StreamName window ledger, pruning schedule, tail-subsequence row, cofinal-subsequence row, DyadicRatCore tolerance ledger, retained RegSeqRat readback, inherited RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyPruning}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7851.1, Definition 7851.2, and Definition 7851.3 seed the finite pruning route for regular Cauchy rational streams over displayed StreamName windows, tail and cofinal subsequence rows, dyadic tolerance ledgers, retained RegSeqRat readback, and the inherited RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyPruning}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, hidden infinite selectors, selected ambient limits, countable choice, host real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pruning schedule stability, tail stability, cofinal retained-schedule stability, retained readback exactness, RealUp seal compatibility, and ledger nonescape for BEDC.Derived.RegularCauchyPrur

Chapter 7852

A Concrete Naming Certificate for $\mathsf{DyadicMidpointCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion by finite approximation data, the $\mathsf{DyadicMidpointCompletion}^{\uparrow}$ packet records the midpoint-refinement route by which dyadic bracket approximants are turned into regular rational reads and then into a Real name. It sits over the dyadic arithmetic core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the constructive completion neighbour of chapter 7850. The packet names a finite midpoint-refinement completion surface; it is not Dedekind completeness, open-cover compactness, quotient real equality, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7852.1 (Dyadic midpoint completion carrier)**.** *A* $\mathsf{DyadicMidpointCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, E, J, W, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, recording the admitted dyadic endpoints, midpoint operation, interval inclusion comparisons, and tolerance cells used by the refinement.*
2. *E is the endpoint row. It lists the finite left and right dyadic endpoints inspected at each refinement stage and records their displayed nesting order.*
3. *J is the midpoint-refinement schedule. It records which midpoint bisections are performed, which half-interval survives each displayed comparison, and the finite dependency of every later bracket on earlier endpoint rows.*
4. *W is the finite window budget. It records the number of endpoint and midpoint observations needed for each admitted precision request before any regular rational readback is exposed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row obtained from the finite endpoint and midpoint windows E, J, W.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity witness. It certifies that the stream windows read through S satisfy the displayed dyadic Cauchy bounds supplied by D and W.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, readable only after the dyadic endpoints, midpoint schedule, finite window budget, StreamName row, and RegSeqRat witness have been displayed.*

8. *H records componentwise* hsame *transport for the dyadic arithmetic, endpoint row, midpoint schedule, window budget, StreamName readback, RegSeqRat witness, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$D, E, J, W \longmapsto S \longmapsto Q \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{DyadicMidpointCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpointCompletion}^{\uparrow}}$ *row naming exactly this finite midpoint-refinement packet.*

The classifier compares accepted carriers only through $D, E, J, W, S, Q, R, H, C, P, N$. It has no coordinate for Dedekind completeness, open-cover compactness, quotient reals, host equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7852.2 (Dyadic midpoint completion NameCert obligations)**.** *For every accepted* DyadicMidpointComple *carrier*

$$M = (D, E, J, W, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMidpointCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for dyadic endpoints, the midpoint refinement schedule, the finite window budget, StreamName readback, RegSeqRat regularity, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real-name consumer of the packet must read the finite route*

$$D, E, J, W \longmapsto S \longmapsto Q \longmapsto R$$

before it can consume the terminal seal.

Proof. Project the accepted packet $M = (D, E, J, W, S, Q, R, H, C, P, N)$. The carrier exposes dyadic arithmetic, endpoint brackets, midpoint choices, precision windows, stream readback, regular rational bounds, the Real seal, and the four structural rows in the displayed order.

The endpoint and midpoint data are finite. The row D supplies dyadic arithmetic and comparisons, while E lists the endpoint brackets and J lists the midpoint bisections that refine them. The window row W fixes, for each admitted precision request, the finite prefix of those rows that must be inspected.

Readback then proceeds through S and Q. The $\mathsf{StreamName}^{\uparrow}$ row is obtained only from the displayed finite windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular Cauchy bounds witnessed by the dyadic tolerance ledger and midpoint schedule.

Finally R, H, C, P, N seal, transport, replay, source, and name the same finite route. Since no carrier row stores Dedekind completeness, open-cover compactness, quotient real equality, host equality, a selected limiting representative, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Any accepted $\mathsf{DyadicMidpointCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the displayed dyadic endpoint, midpoint-refinement schedule, finite window, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows is outside the packet surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, richman-completion: no listed sibling is bijective to $\mathsf{DyadicMidpointCompletion}^{\uparrow}$ because this packet binds dyadic endpoints,

midpoint refinement, finite precision windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one constructive completion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMidpointCompletion}\uparrow}$.

Constructive story: A $\mathsf{DyadicMidpointCompletion}^{\uparrow}$ packet is generated from dyadic endpoint rows, midpoint refinement schedules, finite window budgets, StreamName readback, RegSeqRat regularity witnesses, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMidpointCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7852.1 and Definition 7852.2 seed the finite midpoint-refinement completion route over dyadic endpoints, midpoint schedules, finite windows, StreamName readback, RegSeqRat regularity, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMidpointCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, open-cover compactness, quotient reals, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint nesting, midpoint-refinement exactness, finite-window budgeting, StreamName readback, RegSeqRat regularity, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMidpointCompletionUp.

Chapter 7853

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ packet records the finite basis object that sits between regular Cauchy schedules and completion-facing filter reads. It is grounded in the finite stream windows of chapter 126, the regular-rational readback surface of chapter 301, the basis-completion handoff of chapter 4050, and the terminal real-name seal of chapter 38. The packet makes the basis row first-class before completion consumers inspect it; it does not add quotient filters, selected limits, host real equality, countable choice, or ambient completeness.

Definition 7853.1 (RegularCauchyFilterBasis carrier)**.** *A* $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, B, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the regular Cauchy tail observations that may be inspected.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the same windows and regular-rational reads.*
4. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row whose basic observations are exactly the displayed windows in S with tolerances supplied by D.*
5. *M is the finite meet row: every pair of displayed basis observations has a later displayed observation refining both and preserving the same tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, B, M have been displayed.*
7. *H records componentwise* hsame *transport for windows, readback, tolerance, basis, meet, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from basis refinements through $S, R, D, B, M, E, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier accepts packets only through the ten displayed rows $S, R, D, B, M, E, H, C, P, N$. No coordinate stores a quotient filter, hidden cofinal family, selected completion point, host real equality, countable-choice selector, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterBasisUp.RegularCauchyFilterBasisTasteGate single carrier alignment`

Theorem 7853.2 (RegularCauchyFilterBasis NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterBasis}$ *carrier*

$$Q = (S, R, D, B, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite window schedule, regular-rational readback, dyadic tolerance ledger, basis row, finite meet row, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing or Real-facing consumer must first read S, then R and D, then the basis and meet rows B, M, and only then the seal and structural rows E, H, C, P, N.*

Proof. Project the carrier of Definition 7853.1. The visible source of the packet is the finite window schedule S together with its regular-rational readback R and dyadic tolerance ledger D.

Read the basis row next. The row B contains exactly the observations listed by S and the tolerance information listed by D, while M supplies the finite common-refinement row needed to keep the basis directed.

Finally consume the seal and structural rows. The row E is downstream from S, R, D, B, M, and H, C, P, N only transport, replay, source, and name that same route. Since no excluded quotient, selector, host-equality, choice, or completeness coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 7853.3 (RegularCauchyFilterBasis stability scope)**.** *Let $Q = (S, R, D, B, M, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ *carrier. Any finite refinement of displayed basis observations stays inside the same route*

$$S, R, D \longmapsto B, M \longmapsto E, H, C, P, N.$$

The refinement preserves the $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E under componentwise* hsame *transport and displayed* Cont *replay. The stability scope is a finite basis certificate; it is not quotient filter equality, a selected limit, host real equality, countable choice, or ambient completeness.*

Proof. Begin with the displayed basis observations in B. By carrier admission, each observation is named by the finite window schedule S and carries its tolerance data in D.

Use the meet row M for refinement. For a pair of displayed observations, M names a later displayed observation refining both; induction over a finite list repeats this same row without introducing any hidden cofinal family.

The regular-rational and Real-facing rows remain fixed. The readback row is R, the seal row is E, and H, C, P, N preserve the route by transport, replay, provenance, and local naming. Hence every stable refinement remains inside the displayed packet and cannot be reread as one of the excluded completion or equality principles. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ consumer preserving the same displayed basis windows factors through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, finite meet, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: `regularcauchyfilter`, `cauchyfilterbasiscompletion`, `regseqrat`, `real`: the RegularCauchyFilter packet carries a source CauchyFilter row; the CauchyFilterBasis-Completion packet carries the completion-facing basis route; the RegSeqRat packet carries regular rational readback without a basis meet; and the Real packet carries the terminal seal. None is bijective to $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyFilterBasis row, a finite meet row, a RealUp seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7853.1, Definition 7853.2, and Definition 7853.3 seed the finite regular Cauchy basis route between schedules and completion-facing filters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite meet stability, readback preservation, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterBasisUp.

Chapter 7854

A Concrete Naming Certificate for $\mathsf{RegularRealGauge}^{\uparrow}$

Following Bishop and Bridges on regular real names with explicit moduli, the $\mathsf{RegularRealGauge}^{\uparrow}$ packet records finite modulus-gauge control for a regular real name before any completion consumer treats the name as a completed real. It sits over the $\mathsf{StreamName}^{\uparrow}$ finite stream windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the $\mathsf{IntervalDomain}^{\uparrow}$ boundary of chapter 1543. The packet exposes a finite gauge route for later completion and interval consumers; it does not assert quotient real equality, trichotomy, selected limits, or ambient completeness.

Definition 7854.1 (RegularRealGauge carrier)**.** *A* $\mathsf{RegularRealGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (W, R, D, G, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected from the regular real name;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row obtained from the window W;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the displayed dyadic precision request and its admissible refinements;*
4. *G is the gauge and modulus row, recording the finite precision-to-window control visible to consumers;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row by which the readback is treated as a real-name boundary after the gauge has been displayed;*
6. *H stores componentwise* hsame *transport for the window, readback, tolerance ledger, gauge row, real seal, replay, provenance, and local naming;*
7. *C stores the displayed* Cont *replay from W through R, D, G into the sealed* $\mathsf{Real}^{\uparrow}$ *readback;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularRealGauge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRealGauge}^{\uparrow}}$ *row.*

The accepted packet stores only this finite gauge surface. It has no coordinate for host completeness, quotient real equality, trichotomy, selected limits, or an unbounded external modulus function.

Theorem 7854.2 (RegularRealGauge NameCert obligations)**.** *For the carrier of Definition 7854.1, the local* $\mathsf{NameCert}_{\mathsf{RegularRealGauge}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet* $J = (W, R, D, G, E, H, C, P, N)$*;*
2. *gauge visibility: every accepted consumer sees the gauge and modulus row* G *before reading the real seal* E*;*
3. *tolerance monotonicity: refinements recorded by* D *preserve the same stream window and regular-rational readback route;*
4. *real-seal nonescape: the* $\mathsf{Real}^{\uparrow}$ *row* E *is reached only through* W, R, D, G *and the displayed replay row* C*;*
5. *ledger refusal of host completeness: the packet exports no quotient equality of real names, no trichotomy principle, no selected limit, and no ambient completeness witness.*

Proof. Carrier admission is projection from Definition 7854.1. The rows $W, R, D, G, E, H, C, P, N$ are displayed coordinates of one finite BHist packet.

Gauge visibility is forced by the row order. A consumer reaches the real seal E only after the finite window W, readback R, tolerance ledger D, and gauge row G have been read.

For tolerance monotonicity, the ledger D is the only displayed dyadic budget. Refining a precision request therefore preserves the same route through W and R rather than selecting a detached real representative.

The replay row C and transport row H carry exactly this finite route into the seal E. Since no row contains host completeness, quotient equality, trichotomy, or a selected limit operator, the local NameCert refuses those readings. □

Theorem 7854.3 (RegularRealGauge modulus readback)**.** *Every accepted modulus-gauge read for a* $\mathsf{RegularRealGauge}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *in the carrier order of Definition 7854.1. The interval-facing consumer may read* $\mathsf{IntervalDomain}^{\uparrow}$ *provenance from* P*, but the only tolerance budget available to that consumer is the displayed dyadic ledger* D *together with the gauge row* G*.*

In particular, the readback route exports no host function space of moduli, quotient real equality, trichotomy, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $J = (W, R, D, G, E, H, C, P, N)$. The stream window W is read first, and the regular rational row R is the only readback obtained from that window.

Next read the dyadic ledger D and the gauge row G. These rows jointly record the finite precision-to-window control, so the consumer sees a displayed modulus route rather than a host function space of moduli.

The real seal E is then reached through the replay row C, while H transports the same rows componentwise and P records the sibling provenance, including $\mathsf{IntervalDomain}^{\uparrow}$. Thus the readback factors through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ without exporting quotient equality, trichotomy, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealGauge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularRealGauge}^{\uparrow}$ packet is generated from a StreamName finite window, a RegSeqRat readback, one DyadicRatCore tolerance ledger, a finite gauge and modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7854.1, Definition 7854.2, and Definition 7854.3 seed the finite gauge route for regular real names over displayed dyadic tolerance data.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularRealGauge}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completeness, trichotomy, quotient real equality, selected limits, host modulus functions, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, gauge visibility, tolerance monotonicity, Realseal nonescape, and ledger refusal for the displayed packet.

Chapter 7855

A Concrete Naming Certificate for $\mathsf{BuchbergerCriterion}^{\uparrow}$

Following Buchberger's critical-pair criterion, the $\mathsf{BuchbergerCriterion}^{\uparrow}$ packet records the finite sufficiency surface by which displayed S-polynomial reductions certify that a finite generator list is already a Groebner basis. It sits after the polynomial carrier of chapter 50, the ideal source of chapter 81, the coefficient algebra rows of chapter 43 and chapter 44, and the finite reduction packet of chapter 2197. The chapter names the traditional finite critical-pair test as BEDC NameCert data; it does not import quotient ideal equality, an ambient term-order oracle, algebraic closure, selected normal forms outside the displayed rows, or a global ideal-membership decision procedure.

Definition 7855.1 (Buchberger criterion carrier)**.** *A* $\mathsf{BuchbergerCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (P, I, O, L, A, S, R, Z, H, C, K, N)$$

with the following displayed rows:

1. *P is the finite* $\mathsf{Poly}^{\uparrow}$ *source row, including the coefficient operations and polynomial syntax inspected by the packet.*
2. *I is the* $\mathsf{Ideal}^{\uparrow}$ *generator row naming the displayed generator list and its ambient ideal-membership surface.*
3. *O is the finite monomial-order row used only for the inspected leading-term comparisons and descent witnesses.*
4. *L is the leading-term ledger for the displayed generators and intermediate reduction rows.*
5. *A is the finite critical-pair index row, listing exactly the generator pairs inspected by the criterion.*
6. *S is the S-polynomial construction row, displaying common multiples, multiplier rows, and subtraction rows for every pair in A.*
7. *R is the reduction-trace row, listing reducer choices, multiplier rows, subtraction steps, and descent evidence for each displayed S-polynomial.*
8. *Z is the zero-remainder row, recording that each inspected S-polynomial reduction terminates at the displayed zero normal form.*
9. *H records componentwise* hsame *transport for polynomial, ideal, ordering, leading-term, critical-pair, S-polynomial, reduction, zero-remainder, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a Buchberger-criterion request through P, I, O, L, A, S, R, Z before any Groebner-basis handoff is consumed.*
11. *K is the* Pkg *provenance over* BuchbergerCriterion$^{\uparrow}$, Poly$^{\uparrow}$, Ideal$^{\uparrow}$, Ring$^{\uparrow}$, CommRing$^{\uparrow}$, GroebnerBasis$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BuchbergerCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient polynomial-ring equality, a hidden global term order, algebraic closure, a selected infinite basis, or a normal-form oracle outside R and Z.

Theorem 7855.2 (Buchberger criterion obligation surface)**.** *For every accepted* BuchbergerCriterion$^{\uparrow}$ *carrier*

$$B = (P, I, O, L, A, S, R, Z, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{BuchbergerCriterion}^{\uparrow}}$ *surface consists exactly of polynomial-source admission, ideal-generator admission, inspected monomial-order and leading-term exposure, critical-pair coverage through A, S-polynomial construction through S, reduction-trace exactness through R, zero-remainder exposure through Z, componentwise transport through H, displayed replay through C, provenance through K, and local naming through N. A downstream* GroebnerBasis$^{\uparrow}$ *consumer may read the packet only as this finite criterion surface.*

Proof. Project the carrier of Definition 7855.1. The algebraic source rows are P and I, and the inspected comparison rows are O and L.

Read the finite critical-pair row A. Each listed pair is expanded by S into the displayed common multiple, multiplier rows, and subtraction row that form the corresponding S-polynomial.

Induct over the reduction trace R for each listed pair. The base step is the S-polynomial row supplied by S; each later step names a reducer from the displayed generator surface, a multiplier row over P, a subtraction row, and a descent witness from O and L.

At the terminal step, Z records the zero remainder for every inspected pair. Thus the criterion exports only the finite zero-reduction surface consumed by the Groebner-basis route of chapter 2197.

Finally H, C, K, N transport, replay, source, and name the same rows. Since no carrier coordinate stores quotient ideal equality, a hidden term-order oracle, algebraic closure, selected infinite bases, or external normal forms, the listed rows exhaust the local obligation surface. □

Theorem 7855.3 (Buchberger criterion Groebner handoff)**.** *Every accepted* BuchbergerCriterion$^{\uparrow}$ *read handed to a* GroebnerBasis$^{\uparrow}$ *consumer factors through the finite route*

$$P, I, O, L, A, S, R, Z \longmapsto H, C, K, N.$$

The handoff supplies the critical-pair sufficiency rows required by the Groebner-basis finite reduction surface, but it does not supply quotient ideal equality, completed ideal membership, a selected normal-form function, an infinite basis, or a host proof of global algebraic closure.

Proof. Start with the obligation surface above. A Groebner-facing consumer first sees the polynomial and ideal source rows, then the inspected monomial-order and leading-term rows.

The critical-pair index row A determines the finite scope of the criterion. The S-polynomial row S and reduction trace R show how each pair is reduced, while Z records that the displayed terminal row is zero.

Transport and replay preserve that same finite handoff through H and C, with provenance and local naming supplied by K and N. No other row is available, so the consumer cannot reinterpret the criterion as a quotient-ring theorem, a hidden normal-form oracle, or a complete ideal-membership decision procedure. □

Falsifiable prediction: Every accepted $\mathsf{BuchbergerCriterion}^{\uparrow}$ consumer must display the polynomial source, ideal source, inspected monomial-order and leading-term rows, critical-pair index row, S-polynomial construction row, reduction trace, zero-remainder row, transport, replay, provenance, and local naming before using the Buchberger criterion as a Groebner-basis handoff.

Independence witness: polynomial, ideal, groebner-basis: $\mathsf{BuchbergerCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite critical-pair coverage, S-polynomial construction, zero-reduction traces, criterion-specific transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BuchbergerCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BuchbergerCriterion}^{\uparrow}$ packet is generated from polynomial rows, ideal-generator rows, finite monomial-order and leading-term rows, critical-pair index rows, S-polynomial rows, reduction traces, zero-remainder rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BuchbergerCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7855.1, Definition 7855.2, and Definition 7855.3 seed the finite Buchberger critical-pair criterion over PolynomialUp, IdealUp, RingUp, CommRingUp, GroebnerBasisUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BuchbergerCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal equality, a completed ideal-membership decision procedure, algebraic closure, a hidden term-order oracle, selected infinite bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-pair coverage, S-polynomial exactness, reduction-trace exactness, zero-remainder exposure, Groebner handoff, transport, replay, provenance, and local naming for BEDC.Derived.BuchbergerCriterionUp.

Chapter 7856

A Concrete Naming Certificate for MetaCICSubstitutionResidualSpine$^{\uparrow}$

The MetaCICSubstitutionResidualSpine$^{\uparrow}$ packet realizes the residual boundary of chapter 9380 as finite substitution-audit spine data. It sits beside the closed substitution spine of chapter 6103, the typed substitution window of chapter 7519, the residual substitution boundary of chapter 2181, and the subject-reduction discharge route of chapter 719. The packet names the residual window that must remain visible before confluence or subject-reduction consumers use substitution data; it does not prove unrestricted substitution commutation, closed subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 7856.1 (MetaCIC substitution residual-spine carrier)**.** *A* MetaCICSubstitutionResidualSpine$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, B, O, T, E, P, H, C, K, N)$$

with the following displayed rows:

1. *S is the substitution-audit source row, naming the closed shift, closed substitution, closed composition, and generator-closure facts available from the audit map.*
2. *W is the residual-window row, recording the finite binder depths, payload positions, and open residual cells that must be displayed before a consumer reads the packet.*
3. *B is the binder-seal row, separating closed-boundary cases from residual cases under lambda and Pi binders.*
4. *O is the obstruction row, naming missing unrestricted commutation, beta-substitution preservation, dependent-codomain preservation, and subject-reduction discharge boundaries.*
5. *T is the transport row for substitution source, residual windows, binder seals, obstructions, typed-window handoff, replay, provenance, and local naming data.*
6. *E is the confluence and subject-reduction exposure row, specifying which residual-window entries must be visible before any handoff to the confluence or subject-reduction frontier is consumed.*
7. *P is the provenance row over the substitution audit map, the typed substitution window, the residual substitution boundary, the subject-reduction discharge route,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
8. *H records componentwise* hsame *transport for all displayed rows.*

9. *C records displayed* Cont *replay from a residual-spine request through S, W, B, O, T, E before any downstream handoff is used.*
10. *K is the* Pkg *source row over* $\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}$, $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$, $\mathsf{MetaCICTypedSubstitutionWindow}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for unrestricted substitution calculus, closed subject reduction, confluence closure, evaluator equality, quotient syntax equality, or kernel consistency. Future Lean `TasteGate.lean` *coverage for this packet must include both* `Nontrivial` *and* `FieldFaithful` *instances for the residual-window classifier.*

Theorem 7856.2 (MetaCIC substitution residual-spine obligations)**.** *For every accepted* MetaCICSubstitutionResid *carrier*

$$R = (S, W, B, O, T, E, P, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}}$ *surface consists exactly of substitution-audit source admission through S, residual-window exposure through W, binder-seal exposure through B, obstruction exposure through O, transport through T and H, confluence and subject-reduction exposure through E, displayed replay through C, provenance through P and K, and local naming through N. Every confluence or subject-reduction consumer must keep the residual-window row visible before using the packet.*

Proof. Project the carrier of Definition 7856.1. The substitution-audit source is S, while W and B separate the visible residual windows from binder-sealed closed cases.

Read the obstruction row O. It keeps unrestricted substitution commutation, beta-substitution preservation, dependent-codomain preservation, and subject-reduction discharge as explicit boundaries rather than hidden consequences of the spine.

The exposure row E then governs downstream handoff. A confluence or subject-reduction consumer may inspect the packet only after the relevant entries of W have been displayed and transported through T and H.

Finally C, P, K, N replay, source, and name the same finite route. Since no carrier coordinate stores closed subject reduction, full confluence, evaluator equality, quotient syntax equality, or kernel consistency, the listed rows exhaust the local obligation surface. □

Theorem 7856.3 (MetaCIC residual-window handoff)**.** *Every accepted* $\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}$ *read sent to a confluence or subject-reduction consumer factors through the finite route*

$$S, W, B, O, T, E \longmapsto H, C, P, K, N.$$

The handoff may expose residual substitution windows, binder seals, and obstruction rows for downstream audit. It cannot be reread as unrestricted substitution compatibility, closed subject reduction, evaluator equality, quotient syntax equality, confluence closure, or kernel consistency.

Proof. Begin with the residual-spine obligation surface. The source row S supplies only the closed substitution-audit facts, so any open residual demand must pass through W.

The binder-seal row B distinguishes closed-boundary cases from residual cases, while O records the endpoints that remain unavailable. The handoff row E therefore exposes a residual window before a confluence or subject-reduction consumer may proceed.

Transport and replay through T, H, C preserve the same route, and P, K, N attach provenance and local naming. Because no hidden row discharges the obstruction endpoints, the packet remains a residual spine rather than a closed substitution or subject-reduction theorem. □

Falsifiable prediction: Every accepted read exposes the residual-window row before any subject-reduction or confluence handoff is consumed.

Independence witness: metacic-substitution-spine, metacic-typed-substitution-window, metacic-residual-substitution-boundary, subjectreductiondischarge: none is bijective to $\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}$ because this packet jointly carries residual-window, binder-seal, obstruction, transport, replay, provenance, and local NameCert rows for the substitution audit boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}$ packet is generated from substitution-audit source rows, residual-window rows, binder-seal rows, obstruction rows, confluence and subject-reduction exposure rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7856.1, Definition 7856.2, and Definition 7856.3 seed the finite residual-substitution spine over MetaCIC substitution audit rows, residual windows, binder seals, obstructions, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionResidualSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution commutation, closed subject reduction, confluence closure, evaluator equality, quotient syntax equality, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, residual-window exposure, binder-seal exposure, obstruction exposure, handoff exposure, transport, replay, provenance, local naming, and TasteGate nontriviality and field-faithfulness for BEDC.Derived.MetaCICSubstitutionResidualSpineUp.

Chapter 7857

A Concrete Naming Certificate for $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$

Following Banach and Mazur on invariant means and the later finite-window summability tradition, the $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ packet records the bounded-sequence shift-average rows that may be inspected before any Banach-limit boundary is discussed. It sits between the finite shift-average route of chapter 7251, the Hardy–Cesaro mean surface of chapter 1765, the Toeplitz matrix-window route of chapter 1682, the Silverman–Toeplitz regularity packet of chapter 2126, and the boundary chapter for chapter 6697. The chapter names only finite windows, local shift rows, displayed averages, boundedness ledgers, real-name seals, transport, replay, provenance, and local naming; it does not assert a Hahn–Banach extension, an ultrafilter, a choice-selected Banach limit, quotient equality of bounded sequences, or an ambient dual-space element.

Definition 7857.1 (Banach limit finite-window carrier)**.** *A* $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, A, H, T, V, E, M, C, P, N)$$

with the following displayed rows:

1. *B is the bounded-sequence window row, listing the finite scalar entries and the displayed bound available to the consumer.*
2. *S is the shift schedule row, recording the finite offsets by which the same bounded sequence is re-indexed.*
3. *A is the finite average row supplied by* $\mathsf{FiniteShiftAverage}^{\uparrow}$ *for each displayed shift window.*
4. *H is the Hardy–Cesaro comparison row supplied by* $\mathsf{HardyCesaroMean}^{\uparrow}$ *for the same finite prefix data.*
5. *T is the Toeplitz matrix-window row, and V is the Silverman–Toeplitz regularity row, recording the finite transform checks that remain visible to summability consumers.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal obtained only after the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the inspected averages.*
7. *M records componentwise* hsame *transport for B, S, A, H, T, V, E.*
8. *C records displayed* Cont *replay from a finite-window request through shift scheduling, averaging, Cesaro comparison, Toeplitz windows, Silverman–Toeplitz regularity, and real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$, $\mathsf{FiniteShiftAverage}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{SilvermanToeplitz}^{\uparrow}$, $\mathsf{BanachLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachLimitFiniteWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a full invariant mean, a Hahn–Banach extension theorem, an ultrafilter, a choice-selected functional, quotient equality of bounded sequences, or equality inside an ambient dual space.

Lean4 ✓ `BEDC.Derived.BanachLimitFiniteWindowUp.BanachLimitFiniteWindowTasteGate single carrier alignment`

Theorem 7857.2 (Banach limit finite-window NameCert obligations)**.** *For every accepted* BanachLimitFiniteWindo *carrier $W = (B, S, A, H, T, V, E, M, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachLimitFiniteWindow}^{\uparrow}}$ *surface contains exactly the following finite obligation blocks:*

1. *bounded-window admission through B, including the displayed scalar entries and bound ledger;*
2. *shift-schedule admission through S, including every inspected offset and its finite source window;*
3. *finite averaging through A and Hardy–Cesaro comparison through H;*
4. *Toeplitz and Silverman–Toeplitz summability-window visibility through T, V;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing through E;*
6. *componentwise transport, displayed replay, provenance, and local naming through M, C, P, N.*

No other block is exported by the carrier.

Proof. Project the carrier of Definition 7857.1. The rows $B, S, A, H, T, V, E, M, C, P, N$ are the complete finite packet, so each obligation must be read from one of those rows.

The source and shift data are exhausted by B and S. The averaging rows A, H then expose the finite shift average and Hardy–Cesaro comparison before any summability endpoint is visible.

The matrix-facing route is exhausted by T, V. These rows record only finite Toeplitz and Silverman–Toeplitz windows, and E seals only the displayed regular-rational readback of the inspected averages.

Finally M, C, P, N transport, replay, source, and name the same rows. Since the tuple has no coordinate for a Hahn–Banach extension, ultrafilter, selected invariant mean, quotient equality, or ambient dual-space equality, those objects cannot be exported from the local certificate. □

Theorem 7857.3 (Banach limit finite-window summability boundary)**.** *Every summability-facing consumer of an accepted* $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ *packet must pass through the sibling route*

$$\mathsf{FiniteShiftAverage}^{\uparrow} \longmapsto \mathsf{HardyCesaroMean}^{\uparrow} \longmapsto \mathsf{ToeplitzLemma}^{\uparrow} \longmapsto \mathsf{SilvermanToeplitz}^{\uparrow} \longmapsto \mathsf{BanachLimit}^{\uparrow}$$

as a finite BHist *ledger over $B, S, A, H, T, V, E, M, C, P, N$. The consumer may cite the packet as a finite shift-average boundary for bounded-sequence reads, but it may not retype that boundary as a full Banach-limit existence theorem, a Hahn–Banach extension, an ultrafilter construction, a choice-selected invariant mean, quotient equality of bounded sequences, or equality in an ambient dual carrier.*

Proof. Apply the obligation theorem Definition 7857.2. It forces bounded windows, shifts, finite averages, Hardy–Cesaro comparison, Toeplitz windows, Silverman–Toeplitz regularity, regular-rational readback, and real sealing to be displayed before a summability consumer receives any token.

The displayed sibling route records the order in which those rows are consumed. $\mathsf{FiniteShiftAverage}^{\uparrow}$ supplies A, $\mathsf{HardyCesaroMean}^{\uparrow}$ supplies H, $\mathsf{ToeplitzLemma}^{\uparrow}$ supplies T, $\mathsf{SilvermanToeplitz}^{\uparrow}$ supplies V, and $\mathsf{BanachLimit}^{\uparrow}$ is present only as the boundary chapter named in provenance.

Transport, replay, provenance, and local naming through M, C, P, N keep the handoff finite. Because the carrier does not contain a full extension functional, an ultrafilter, a choice selector, quotient sequence equality, or ambient dual equality, a consumer that claims one of those endpoints has left the displayed $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ consumer must display bounded-sequence windows, shift schedules, finite shift averages, Hardy–Cesaro comparison, Toeplitz windows, Silverman–Toeplitz regularity, RegSeqRat readback, Real sealing, transport, replay, provenance, and local NameCert evidence before any Banach-limit-facing boundary token is public.

Independence witness: finite-shift-average, hardy-cesaro-mean, toeplitz-lemma, silverman-toeplitz, banach-limit: $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds bounded windows, shifts, finite averages, Cesaro comparison, matrix windows, regularity rows, Real sealing, transport, replay, provenance, and local naming into one finite packet while refusing full Banach-limit existence.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachLimitFiniteWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachLimitFiniteWindow}^{\uparrow}$ packet is generated from bounded-sequence windows, finite shift schedules, finite average rows, Hardy–Cesaro comparison rows, Toeplitz matrix windows, Silverman–Toeplitz regularity rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7857.1, Definition 7857.2, and Definition 7857.3 seed the finite bounded-sequence shift-average boundary for Banach-limit-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachLimitFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Hahn–Banach, ultrafilter, choice-based Banach-limit existence, ambient quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bounded-window admission, shift-schedule stability, finite-average exactness, Hardy–Cesaro comparison, Toeplitz and Silverman–Toeplitz window visibility, real-name sealing, transport, replay, provenance, and local naming for BEDC.Derived.BanachLimitFiniteWindowUp.

Chapter 7858

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley filter terminology, the $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ packet records the functorial handoff by which a displayed regular Cauchy filter-basis route is transported between completion-facing sources without choosing a completed limit. It sits after the basis carrier of chapter 7853, the filter surface of chapter 333, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed basis-to-filter route and its structural preservation; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient filters, host equality of completions, or countable choice.

Definition 7858.1 (RegularCauchyFilterBasisFunctor carrier)**.** *A* $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, B, M, \Phi, G, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule inherited from the source regular Cauchy basis.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger transported with the same windows.*
4. B *is the source* $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ *basis row, including its finite meet surface.*
5. M *records preservation of the finite meet row under the handoff.*
6. Φ *is the displayed map row sending each accepted source basis observation to a target basis observation.*
7. G *is the* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer row exposed only after* S, R, D, B, M, Φ *have been read.*
8. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the target filter-basis handoff.*
9. H *records componentwise* hsame *transport for the schedule, readback, tolerance, basis, meet, map, filter, seal, replay, provenance, and local naming rows.*
10. C *records displayed* Cont *replay of the same handoff route.*
11. P *records* Pkg *provenance over* $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$, $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}}$ *row.*
The carrier stores no quotient filter, no chosen limit point, no external completion object, no countable-choice selector, and no ambient proof that every Cauchy filter converges.

Theorem 7858.2 (RegularCauchyFilterBasisFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ *carrier*

$$F = (S, R, D, B, M, \Phi, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite schedule, regular-rational readback, dyadic tolerance ledger, source basis row, meet-preservation row, displayed handoff map, target* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. Every Real-facing or Completion-facing consumer must first read the finite route*

$$S, R, D, B, M, \Phi \longmapsto G, E, H, C, P, N.$$

Proof. Project the carrier of Definition 7858.1. The source side is the finite window schedule S, the regular-rational readback R, the dyadic tolerance ledger D, the source basis row B, and the meet-preservation row M inherited from chapter 7853.

Read the functorial handoff through Φ. It sends accepted source basis observations to target basis observations while preserving the finite meet and tolerance rows, so the target filter row G is exposed only after the displayed basis data have already been read.

Finally consume the seal and structural rows. The real-name seal E and the transport, replay, provenance, and local naming rows H, C, P, N preserve the same route. Since the carrier has no coordinate for quotient filters, selected limits, countable choice, host completion equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7858.3 (RegularCauchyFilterBasisFunctor finite handoff scope)**.** *Every accepted* RegularCauchyFilterBas *handoff from a* $\mathsf{RegularCauchyFilterBasis}^{\uparrow}$ *source to a* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the source basis and meet rows, the displayed handoff map, the target filter row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P, N. The handoff is a finite functorial route; it is not quotient-filter equality, selected convergence, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the source basis carrier. The rows S, R, D, B, M provide all source observations and finite common-refinement data before a target filter consumer can read the packet.

Apply the displayed map row Φ. It preserves the basis observations and the meet row at the level of the carried BHist data, so the consumer row G is reached by a finite handoff rather than by quotienting filters.

The remaining rows E, H, C, P, N only seal, transport, replay, source, and name the same handoff. Therefore no selected limit, countable-choice selector, host equality of completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available from the packet. □

Falsifiable prediction: A $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ consumer that exposes a target $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row while omitting the finite schedule, regular-rational readback, dyadic tolerance ledger, source basis, meet-preservation, displayed handoff map, transport, replay, provenance, or local $\mathsf{NameCert}$ rows refutes Definition 7858.1.

Independence witness: regular-cauchy-filter-basis, cauchyfilter, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ is not bijective to any listed sibling because it

carries a displayed preservation map from a regular Cauchy basis into a filter consumer. The basis sibling has the source meet row without the handoff map; CauchyFilterUp has filter consumption without the source basis functor; StreamNameUp, RegSeqRatUp, and DyadicRatCoreUp provide schedule, readback, and tolerance rows without filter functoriality; RealUp supplies the terminal seal without the basis-to-filter route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RegularCauchyFilterBasis source row, meet preservation, a displayed handoff map, a CauchyFilter consumer row, a RealUp seal, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7858.1, Definition 7858.2, and Definition 7858.3 seed the functorial handoff from regular Cauchy filter bases to completion-facing filter and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterBasisFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, selected limits, quotient filters, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, meet preservation, displayed handoff stability, filter consumer nonescape, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterBasisFunctor

Chapter 7859

A Concrete Naming Certificate for $\mathsf{MacLanePentagonCoherence}^{\uparrow}$

Following Mac Lane's coherence theorem for monoidal categories, the $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ packet records the finite pentagon route by which fourfold tensor reassociation is read as one BEDC NameCert surface. It sits after the categorical carrier of chapter 61, the functorial tensor surface of chapter 62, the natural-transformation comparison surface of chapter 63, and the monoidal-category carrier of chapter 181. The packet names only displayed object rows, tensor-functor rows, associator rows, the two pentagon boundary composites, transport, replay, provenance, and local naming; it is not a host coherence theorem, not a quotient of parenthesized tensor words, and not an ambient category-library appeal.

Definition 7859.1 (Mac Lane pentagon coherence carrier)**.** *A* $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (C, T, A, W, L, R, H, Q, K, N)$$

with the following displayed rows:

1. *C is the source* $\mathsf{Cat}^{\uparrow}$ *row naming the object and morphism carrier on which the monoidal structure is inspected.*
2. *T is the* $\mathsf{Functor}^{\uparrow}$ *tensor row, recording the displayed binary tensor action used in each reassociation step.*
3. *A is the* $\mathsf{NatTrans}^{\uparrow}$ *associator row, recording the natural isomorphism component read for each adjacent reassociation.*
4. *W is the four-object word row naming the inspected quadruple and its five parenthesized tensor shapes.*
5. *L is the left pentagon boundary row, recording the two-step reassociation route from $(((w \otimes x) \otimes y) \otimes z)$ to $w \otimes (x \otimes (y \otimes z))$.*
6. *R is the right pentagon boundary row, recording the three-step reassociation route through the remaining pentagon edges.*
7. *H records componentwise* hsame *transport for category, tensor, associator, word, left-boundary, right-boundary, replay, provenance, and local naming rows.*

8. *Q records displayed* Cont *replay from a pentagon request through C, T, A, W, L, R before any monoidal consumer is allowed to read the coherence row.*
9. *K is the* Pkg *provenance over* $\mathsf{MacLanePentagonCoherence}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{MonoidalCat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MacLanePentagonCoherence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an external category object, quotient equality of parenthesized tensor expressions, a hidden naturality square, or a host proof of monoidal coherence.

Theorem 7859.2 (Mac Lane pentagon coherence NameCert obligations)**.** *For every accepted* $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ *carrier*

$$P = (C, T, A, W, L, R, H, Q, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{MacLanePentagonCoherence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Cat}^{\uparrow}$ *row C, tensor* $\mathsf{Functor}^{\uparrow}$ *row T, associator* $\mathsf{NatTrans}^{\uparrow}$ *row A, four-object word row W, and pentagon boundary rows L, R;*
2. *tensor typing, requiring every edge of the pentagon to read the same category and tensor rows before any reassociation is consumed;*
3. *associator exposure, requiring each pentagon edge to be one displayed component of the associator row A;*
4. *boundary comparison, requiring L and R to name the two Mac Lane pentagon composites over the same word row W;*
5. *ledger exhaustion through H, Q, K, N, excluding external category carriers, quotient parenthesization equality, hidden naturality squares, and host coherence imports.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7859.1. The packet exposes only the category row, tensor row, associator row, four-object word row, the two boundary composites, and the structural rows.

Tensor typing is read before any associator edge. Each edge of the pentagon is a tensor expression over the same displayed category carrier and the same tensor functor row, so no second monoidal carrier can enter the route.

Associator exposure is then read from A. The five pentagon edges are not arbitrary morphisms; each is the displayed associator component or tensoring of such a component required by the word row W.

The boundary comparison is local to L and R. These rows name the two Mac Lane composites over the same four-object word, so the packet records coherence as a comparison of displayed histories rather than as an ambient theorem about all parenthesizations.

Finally, H, Q, K, N transport, replay, source, and name the same finite packet. Since the carrier contains no quotient parenthesization equality, hidden naturality square, external category object, or host coherence import, the five rows exhaust the local NameCert surface. □

Theorem 7859.3 (Mac Lane pentagon coherence sibling handoff)**.** *Every accepted* $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ *read factors through the sibling surfaces*

$$\mathsf{Cat}^{\uparrow}, \quad \mathsf{Functor}^{\uparrow}, \quad \mathsf{NatTrans}^{\uparrow}, \quad \mathsf{MonoidalCat}^{\uparrow}$$

before it exports a pentagon comparison. The handoff supplies a finite reassociation ledger for monoidal consumers, but it does not supply a host monoidal category, quotient equality of tensor words, a global coherence theorem, or an unrestricted naturality oracle.

Proof. Start with the obligation surface above. The category row C supplies the object and morphism carrier, the tensor row T supplies the binary functorial action, and the associator row A supplies the natural-transformation components.

Read the word row W next. It fixes the four displayed objects and the five parenthesized tensor shapes inspected by the packet.

The boundary rows L and R then expose the two pentagon routes over that same word. Transport and replay through H and Q preserve the displayed rows, while K and N record provenance and local naming.

Thus a $\mathsf{MonoidalCat}^{\uparrow}$ consumer receives only the finite BEDC pentagon ledger. No host monoidal category, quotient tensor-word equality, global coherence theorem, or unrestricted naturality oracle is available from the packet. □

Falsifiable prediction: Every accepted $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ consumer must display the category row, tensor functor row, associator natural-transformation row, four-object word row, both pentagon boundary rows, transport, replay, provenance, and local naming before reading the pentagon comparison.

Independence witness: category, functor, nattrans, monoidalcat: $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the four-object word, both pentagon boundary composites, pentagon-specific transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MacLanePentagonCoherence}^{\uparrow}}$.

Constructive story: A $\mathsf{MacLanePentagonCoherence}^{\uparrow}$ packet is generated from category rows, tensor functor rows, associator natural-transformation rows, four-object word rows, two pentagon boundary composites, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MacLanePentagonCoherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7859.1, Definition 7859.2, and Definition 7859.3 seed the finite Mac Lane pentagon coherence ledger over CategoryUp, FunctorUp, NatTransUp, MonoidalCatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MacLanePentagonCoherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host coherence theorem, quotient equality of parenthesized tensor words, ambient monoidal-category semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tensor typing, associator exposure, boundary comparison, transport, replay, provenance, and local naming for BEDC.Derived.MacLanePentagonCoherenceUp.

Chapter 7860

A Concrete Naming Certificate for $\mathsf{CZeroSpace}^{\uparrow}$

Following Banach-space treatments of c_0, the $\mathsf{CZeroSpace}^{\uparrow}$ packet presents a real-valued sequence with an explicit tail-to-zero modulus as finite NameCert data rather than as a quotient subspace of a completed sequence algebra. It is grounded in the sequence carrier of chapter 3641, the bounded-sequence surface of chapter 3451, the regular-rate route of chapter 3541, the finite stream naming of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the complete-metric boundary of chapter 3424.

Definition 7860.1 (CZeroSpace carrier)**.** *A* $\mathsf{CZeroSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, R, B, T, K, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSequenceSpace}^{\uparrow}$ *source row, read only through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule naming inspected sequence windows.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *real-coordinate seal, supplying regular rational readback for every coordinate requested by S.*
3. *B is the* $\mathsf{RealBoundedSequence}^{\uparrow}$ *boundedness witness, displayed before any norm-facing or complete-metric consumer may read the packet.*
4. *T is the dyadic tail modulus: for each displayed tolerance request, T gives a finite tail window after which the inspected coordinates are forced inside that tolerance.*
5. *K is the* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *readback row, carrying the same tail ledger as a regular-rate sequence-space route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the coordinate readback, boundedness witness, and tail modulus.*
7. *H records componentwise* hsame *transport for the sequence source, real-coordinate seal, boundedness witness, tail modulus, regular Cauchy readback, dyadic ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each finite coordinate, boundedness, tail, and complete-metric request through S, R, B, T, K, D.*

9. *P is the* Pkg *provenance over* $\mathsf{CZeroSpace}^{\uparrow}$, $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{RealBoundedSequence}^{\uparrow}$, $\mathsf{RegularCauchySequence}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CZeroSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for arbitrary choice of subsequences, quotient stream equality, trichotomy, host real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7860.2 (CZeroSpace tail-modulus obligation)**.** *For every accepted* $\mathsf{CZeroSpace}^{\uparrow}$ *carrier* $Z = (S, R, B, T, K, D, H, C, P, N)$*, the tail-to-zero claim is exhausted by the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealBoundedSequence}^{\uparrow}$ *boundedness witness, and* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *rate row. In particular, every tail-zero consumer must first read the finite route*

$$S \longrightarrow R \longrightarrow D \longrightarrow T \longrightarrow K$$

with B displayed as the boundedness side condition, and no consumer may replace that route by arbitrary choice of subsequences, quotient stream equality, trichotomy, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7860.1. The sequence source is the $\mathsf{RealSequenceSpace}^{\uparrow}$ row S, the coordinate readback is the $\mathsf{RegSeqRat}^{\uparrow}$ row R, and the tolerance information for each coordinate and each tail request is the $\mathsf{DyadicRatCore}^{\uparrow}$ row D.

The displayed tail modulus T is then read over those same finite windows. The boundedness witness B records the norm-facing side condition, while the regular Cauchy readback K exposes the tail ledger to the regular-rate route described in Definition 3541.5.

Finally use the sequence-space subspace route of Definition 3641.6. It identifies the tail-zero surface as an extra displayed ledger over the existing real-sequence route, so the excluded host-side objects never become classifier coordinates of Z. □

Theorem 7860.3 (CZeroSpace metric seal)**.** *Every* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *or norm-facing consumer of an accepted* $\mathsf{CZeroSpace}^{\uparrow}$ *carrier reads the packet through the bounded real sequence row B, the tail modulus T, and the regular Cauchy route K after the sequence source, coordinate seal, and dyadic ledger have been displayed. The exported metric surface is therefore the finite packet*

$$S, R, B, T, K, D, H, C, P, N$$

rather than an ambient complete subspace obtained from quotient stream equality, trichotomy, selected limits, arbitrary subsequence choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a complete-metric or norm-facing read. The carrier exposes no such read before S, R, D name the inspected real-valued coordinates and before B supplies the boundedness witness.

The tail-to-zero data are supplied by T and replayed through K. By Definition 7860.2, the tail read is a dyadic ledger obligation over the displayed real-sequence route, and by Definition 3541.5 the same obligation remains visible to regular-rate consumers.

The complete-metric boundary can therefore inspect only the rows $S, R, B, T, K, D, H, C, P, N$. Since neither the definition nor the preceding obligation theorem contains quotient stream equality, trichotomy, selected limits, arbitrary subsequence choice, host real equality, or ambient Real completeness, the metric seal exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CZeroSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CZeroSpace}^{\uparrow}$ packet is generated from a RealSequenceSpace source, regular rational coordinate readback, a bounded real sequence witness, a displayed dyadic tail-to-zero modulus, a RegularCauchySequenceSpace rate row, DyadicRatCore tolerance data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CZeroSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7860.1, Definition 7860.2, and Definition 7860.3 seed the constructive tail-zero real sequence space over displayed real-sequence, boundedness, regular-rate, dyadic tolerance, and complete-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CZeroSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary choice of subsequences, quotient stream equality, trichotomy, selected limits, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundedness routing, tail-modulus exactness, regular Cauchy readback, dyadic tolerance stability, metric-seal non-escape, and transport stability for BEDC.Derived.CZeroSpaceUp.

Chapter 7861

A Concrete Naming Certificate for $\mathsf{CantorIntersectionWindow}^{\uparrow}$

Following Cantor and Bishop–Bridges on nested interval principles, the $\mathsf{CantorIntersectionWindow}^{\uparrow}$ packet records finite nested closed-window data at the point where a Real and Completion consumer asks for a displayed handoff. It sits after the nested-chain surface of chapter 320, the dyadic tolerance ledger of chapter 291, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The route is the finite-window counterpart of the real-completeness vision in section 9238.1: it names inspected nested windows and their regular readback, not an ambient completeness theorem.

Definition 7861.1 (Cantor intersection window carrier)**.** *A* $\mathsf{CantorIntersectionWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{NestedInterval}^{\uparrow}$ *row naming the finite nested closed-window chain under inspection.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, width, and tolerance ledger for those same windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row listing the stages read by the handoff.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name extracted from the displayed windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, D, S, R have been exposed.*
6. *H records componentwise* hsame *transport for the nested-window, dyadic-ledger, stream-window, readback, seal, provenance, replay, and local naming rows.*
7. *C records displayed* Cont *replay from a finite window request through I, D, S, R, E and back to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionWindow}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for interval compactness, countable choice, selected completed endpoints, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7861.2 (Cantor intersection window NameCert obligations)**.** *For every accepted* CantorIntersectionWindo *carrier* $W = (I, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionWindow}^{\uparrow}}$ *surface consists exactly of the nested closed-window row* I*, dyadic endpoint and tolerance ledger* D*, finite stream window* S*, regular rational readback* R*, terminal Real seal* E*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* N*. A Real or Completion consumer must expose these rows before reading the finite nested-window handoff.*

Proof. Project the carrier of Definition 7861.1. The analytic rows I, D, S, R, E list the only route from the displayed nested windows to a Real-facing seal.

The nested interval row I is read first, and the dyadic ledger D records the endpoint and width data for that same finite chain. The stream row S then identifies the inspected stages, while R supplies the regular rational readback available at those stages.

The seal row E is downstream of this finite display. The structural rows H, C, P, N transport, replay, source, and name the same packet, so no extra coordinate can introduce compactness, choice, quotient equality, host real equality, or ambient completeness. □

Theorem 7861.3 (Cantor intersection window Real and Completion handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CantorIntersectionWindow}^{\uparrow}$ *packet factors through*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $I, D, S, R, E, H, C, P, N$*. The handoff may transport and replay the finite nested-window route, but it cannot export interval compactness, countable choice, quotient stream equality, selected completed endpoints, a chosen limit real, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert-obligation theorem. It forces the consumer to expose the nested closed-window row, dyadic endpoint and tolerance ledger, finite stream window, and regular rational readback before the terminal Real seal is read.

Read the sibling route in order. The $\mathsf{NestedInterval}^{\uparrow}$ chapter supplies the retained finite chain, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the endpoint and tolerance arithmetic, $\mathsf{StreamName}^{\uparrow}$ supplies the inspected stages, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal reached after those rows.

The same route is the Completion-facing handoff: a completion consumer receives the finite window and regular readback rows before it can inspect a Real-facing seal. Since the packet contains no selector for an infinite branch, no quotient stream equality, no countable-choice schedule, no chosen limit real, and no ambient completeness theorem, the handoff remains finite NameCert data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorIntersectionWindow}^{\uparrow}$ packet is generated from displayed NestedIntervalUp finite closed-window rows, a DyadicRatCoreUp endpoint and tolerance ledger, StreamNameUp finite observations, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionWindow}^{\uparrow}}$, seed)

Scope closed: Definition 7861.1, Definition 7861.2, and Definition 7861.3 seed the finite nested-window Real and Completion handoff route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close interval compactness, countable choice, quotient stream equality, selected completed endpoints, chosen limit reals, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window exposure, dyadic tolerance exposure, finite-window readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionWindowU

Chapter 7862

A Concrete Naming Certificate for $\mathsf{BohrCompactification}^{\uparrow}$

Following Bohr and Weil on compactification of topological groups, read through the finite-window discipline of BEDC naming, the $\mathsf{BohrCompactification}^{\uparrow}$ packet records the compactification-facing data exposed by almost-periodic observables without taking a host compact group as primitive. It sits beside the translation ledger of chapter 5969, the topological-group source of chapter 204, the compact target interface of chapter 60, the uniform handoff of chapter 339, the continuous-map carrier of chapter 130, and the terminal real-name seal of chapter 38. The packet names only the finite compactification boundary visible through displayed translation and character rows; it does not assert host compactification, quotient hull equality, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7862.1 (Bohr compactification carrier)**.** *A* $\mathsf{BohrCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, A, K, U, M, S, R, H, C, P, N)$$

with the following visible rows:

1. *G is the* $\mathsf{TopGroup}^{\uparrow}$ *source row, naming the displayed topological-group multiplication, inverse, identity, neighbourhood, and translation windows consumed by the compactification-facing read.*
2. *A is the* $\mathsf{AlmostPeriodic}^{\uparrow}$ *translation ledger row, recording the finite almost-periodic translation budgets and relative-density witnesses inspected before any compact target is exposed.*
3. *K is the compact target row, read through* $\mathsf{Compact}^{\uparrow}$ *as a displayed finite-cover and finite-net target interface rather than as an external compact group.*
4. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *handoff row, carrying entourages and uniform-comparison data needed by character and translation consumers.*
5. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row for continuous real-valued character windows from the source side to the compact target side.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readback row, used when real-valued characters are inspected through finite observation windows and regular rational seals.*
7. *R is the refusal row, recording the absent quotient hull, host function extensionality, selected-limit operator, open-cover compactness theorem, and ambient real-completeness principle.*
8. *H records componentwise* hsame *transport for $G, A, K, U, M, S, R, P, N$.*

9. *C records displayed* Cont *replay of the route from topological-group source rows through almost-periodic translation data, uniform and continuous-map handoffs, finite real readback, refusal rows, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{BohrCompactification}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{AlmostPeriodic}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BohrCompactification}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed quotient hull, host equality of almost-periodic functions, an unrestricted character dual, a selected compactification limit, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7862.2 (Bohr compactification NameCert obligations)**.** *For every accepted* $\mathsf{BohrCompactification}^{\uparrow}$ *carrier $B = (G, A, K, U, M, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BohrCompactification}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed packet; compact target exposure through K; translation-ledger exactness through A; uniform-continuous character handoff through U, M; finite real-valued character readback through S; refusal exactness through R; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N. Every compactification-facing consumer must pass through those rows before inspecting a Bohr-style boundary.*

Proof. Project the carrier of Definition 7862.1. The topological-group source row G and almost-periodic ledger row A are displayed before the compact target row K, so the target cannot be read without the source translations that make it visible.

Next use the uniform and continuous-map handoffs. Rows U and M expose the uniform-continuous character route, while S forces every real-valued character inspection through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal.

Finally attach the refusal, transport, replay, provenance, and local naming rows. The refusal row records that quotient hulls, host function extensionality, selected limits, open-cover compactness, and ambient real completeness are not exported, while H, C, P, N preserve the same finite route under transport and public replay. □

Falsifiable prediction: A consumer that reads a compact target for $\mathsf{BohrCompactification}^{\uparrow}$ without first exposing the $\mathsf{AlmostPeriodic}^{\uparrow}$ translation ledger and the $\mathsf{UniformSpace}^{\uparrow}$ handoff is outside the accepted carrier.

Independence witness: almostperiodic, topgroup, compact, uniformspace, continuousmap, real: $\mathsf{BohrCompactification}^{\uparrow}$ is not bijective to any listed sibling because it binds topological-group source rows, almost-periodic translation rows, compact target rows, uniform and continuous-map handoffs, finite real-valued character readback, refusal rows, transport, replay, provenance, and local NameCert into one compactification-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BohrCompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{BohrCompactification}^{\uparrow}$ packet is generated from topological-group source rows, almost-periodic translation rows, compact target rows, uniform and continuous-map handoffs, finite real-valued character readback, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BohrCompactification}^{\uparrow}}$, seed)

Scope closed: Definition 7862.1 and Definition 7862.2 specify the compactification-facing seed surface visible from the AlmostPeriodic boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BohrCompactification}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove Lean verification, bridge checking, host Bohr compactification, quotient hull equality, host function extensionality, selected limits, open-cover compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact target exposure, translation-ledger exactness, uniform-continuous character handoff, finite real readback, refusal exactness, transport stability, and TasteGate evidence for BEDC.Derived.BohrCompactificationUp.

Chapter 7863

A Concrete Naming Certificate for $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ packet records nested dyadic interval threads as finite regular-Cauchy readback data. It sits after the dyadic arithmetic surface of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing finite-window boundary of chapter 1559. The packet is a displayed thread-to-real seal route, not a quotient completion, selected representative, trichotomy branch, or ambient completeness principle.

Definition 7863.1 (Cantor thread regular-Cauchy carrier)**.** *A* $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested dyadic interval thread row, storing a finite initial segment of dyadic left and right endpoints.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records the displayed observations of the thread.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger proving that each displayed interval width is bounded by the requested dyadic precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the interval midpoints or tagged endpoints over the same windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, W, D, R have been replayed.*
6. *F is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *handoff row for completion-facing consumers of the finite thread.*
7. *H records componentwise* hsame *transport for the interval thread, window, tolerance, readback, seal, completion handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite precision request through $I, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletionFinite}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorThreadRegularCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a selected limit point, quotient stream equality, countable-choice schedule, host real equality, open-cover compactness, or ambient completion theorem.

Theorem 7863.2 (Cantor thread regular-Cauchy real seal)**.** *For every accepted* $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ *carrier*

$$T = (I, W, D, R, E, F, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer must first expose the nested dyadic interval thread* I*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *handoff row* F *before the terminal seal* E *is public. The route exports only finite regular-Cauchy readback data.*

No quotient completion, selected representative, trichotomy branch, countable-choice schedule, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle can replace the displayed thread route.*

Proof. Project the carrier of Definition 7863.1. The rows $I, W, D, R, E, F, H, C, P, N$ are the whole accepted packet, and the terminal Real seal appears only after the thread, window, dyadic tolerance, regular readback, and finite-completion rows.

Run finite induction on the displayed interval windows. The base window exposes one dyadic interval row, and the tolerance ledger D records its width at the requested precision. In the induction step, append one displayed nested interval; the new row refines the previous one through the same W and D rows, so the carrier is not replaced.

Read the regular-Cauchy data through R. The induction supplies a compatible finite chain of dyadic windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row repacks that chain as the readback consumed by E and F.

Finally use the inventory refusal. Since the carrier contains no quotient stream, selected representative, trichotomy branch, choice schedule, open-cover row, or ambient completion coordinate, every Real or completion consumer must use the displayed finite thread route. □

Falsifiable prediction: An accepted $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the nested dyadic thread, StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, and MetricCompletionFinite handoff refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletionfinite: none is bijective to $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ because only this packet binds nested dyadic interval threads, finite windows, regular-Cauchy readback, Real sealing, completion handoff, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorThreadRegularCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorThreadRegularCauchy}^{\uparrow}$ packet is generated from nested dyadic interval rows, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, a MetricCompletionFinite handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorThreadRegularCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7863.1 and Definition 7863.2 seed the finite nested-thread route from dyadic intervals to regular-Cauchy Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorThreadRegularCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked nested-thread admission, dyadic width dominance, StreamName window stability, RegSeqRat readback, Real seal, MetricCompletionFinite handoff, transport, replay, provenance, and local naming for BEDC.Derived.CantorThreadRegularCauchyUp.

Chapter 7864

A Concrete Naming Certificate for $\mathsf{FiniteUniformTailVision}^{\uparrow}$

The $\mathsf{FiniteUniformTailVision}^{\uparrow}$ packet realizes the observer-expansion tail route of chapter 9463 as finite NameCert data. It binds finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{UniformModulus}^{\uparrow}$ tail control, and terminal $\mathsf{Real}^{\uparrow}$ seals over the compact finite-net surface of chapter 129. The packet is a finite uniform-tail readback carrier; it is not a hidden hyperbolic boundary object, not an open-cover compactness theorem, not a selected limit, and not an ambient real-completeness principle.

Definition 7864.1 (Finite uniform-tail vision carrier)**.** *A* $\mathsf{FiniteUniformTailVision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (K, W, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row that bounds the observer-expansion tail surface.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row storing the displayed tail observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from those windows.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row that gives one finite tail bound for the displayed compact source.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after K, W, R, U have been replayed.*
6. *H records componentwise* hsame *transport for compact source, window, readback, modulus, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an observer-tail request through K, W, R, U, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteUniformTailVision}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteUniformTailVision}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for open-cover compactness, a selected tail point, a quotient stream, a host convergence predicate, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7864.2 (Finite uniform-tail vision readback)**.** *For every accepted* $\mathsf{FiniteUniformTailVision}^{\uparrow}$ *carrier*

$$V = (K, W, R, U, E, H, C, P, N),$$

any observer-expansion tail consumer that reaches the $\mathsf{Real}^{\uparrow}$ *seal* E *must first expose the compact finite-net row* K*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{UniformModulus}^{\uparrow}$ *tail-control row* U*. The readback is therefore a finite uniform-tail packet over displayed NameCert rows.*

No hidden boundary center, selected limit, quotient stream, open-cover compactness row, choice schedule, or ambient completeness theorem may replace this route.

Proof. Project the carrier of Definition 7864.1. The only analytic rows are K, W, R, U, E, followed by transport, replay, provenance, and local naming rows.

Read the finite source first. The compact metric row K supplies the finite-net surface, while W records the visible tail observations and R repacks them as regular rational readback data.

Apply the uniform-modulus row U. It provides one displayed tail bound for the same K, W, R surface, so the terminal seal E cannot be detached from the finite uniform-tail route.

Finally inspect the refusal ledger. Since the carrier contains no hidden boundary center, selected limit, quotient stream, open-cover compactness row, choice schedule, or ambient completeness coordinate, every observer-expansion consumer must use the displayed route. □

Falsifiable prediction: An accepted $\mathsf{FiniteUniformTailVision}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying CompactMetric finite-net data, StreamName windows, RegSeqRat readback, and UniformModulus tail control refutes the packet surface.

Independence witness: streamname, regseqrat, real, uniformmodulus, compactmetric: no listed sibling is bijective to $\mathsf{FiniteUniformTailVision}^{\uparrow}$ because only this packet binds the compact finite-net source, finite observer-tail windows, regular readback, uniform modulus, Real seal, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteUniformTailVision}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteUniformTailVision}^{\uparrow}$ packet is generated from a CompactMetric finite-net source, StreamName tail windows, RegSeqRat readback, UniformModulus tail control, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteUniformTailVision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7864.1 and Definition 7864.2 seed the finite uniform-tail readback route for the observer-expansion vision.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteUniformTailVision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hyperbolic boundary semantics, open-cover compactness, selected limits, quotient streams, host convergence, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net exposure, StreamName window stability, RegSeqRat readback, UniformModulus tail control, Real seal, transport, replay, provenance, and local naming for BEDC.Derived.FiniteUniformT

Chapter 7865

A Concrete Naming Certificate for $\mathsf{CauchyRegularComparisonSeal}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyRegularComparisonSeal}^{\uparrow}$ packet names the finite comparison surface used when two regular Cauchy rows are inspected against one tolerance ledger. It sits after the observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance carrier of chapter 291, and the terminal real-name seal of chapter 38. The packet records comparison by displayed rows; it is not a quotient stream equality, a selected limit, countable choice, apartness trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7865.1 (Cauchy regular comparison seal carrier)**.** *A* $\mathsf{CauchyRegularComparisonSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (A, B, D, W_A, W_B, R_A, R_B, Q, E_A, E_B, H, K, P, N)$$

with the following displayed rows:

1. *A and B are the two regular Cauchy rows being compared, each read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that fixes the same precision request for both rows.*
3. *W_A and W_B are the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule windows opened for the two rows after D is displayed.*
4. *R_A and R_B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed by those windows.*
5. *Q is the finite comparison ledger recording which readback entries have been inspected under the common dyadic tolerance request.*
6. *E_A and E_B are the two* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after $A, B, D, W_A, W_B, R_A, R_B, Q$ have been consumed.*
7. *H records componentwise* hsame *transport for the regular rows, tolerance ledger, schedule windows, readbacks, comparison ledger, Real seals, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from each finite comparison request through*

$$A, B, D, W_A, W_B, R_A, R_B, Q \longmapsto E_A, E_B, H, P, N.$$

9. *P is the* Pkg *provenance over* CauchyRegularComparisonSeal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyRegularComparisonSeal}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the two regular rows are compared through the same dyadic tolerance ledger and the same finite schedule discipline before either terminal Real seal is public.

Lean4 ✓ `BEDC.Derived.CauchyRegularComparisonSealUp.CauchyRegularComparisonSealTasteGate single carrier alignment`

Theorem 7865.2 (Cauchy regular comparison seal obligations)**.** *For every accepted* CauchyRegularComparisonSeal$^{\uparrow}$ *carrier*

$$C = (A, B, D, W_A, W_B, R_A, R_B, Q, E_A, E_B, H, K, P, N),$$

the local NameCert$_{\text{CauchyRegularComparisonSeal}^{\uparrow}}$ *surface consists exactly of exposure of the two regular Cauchy rows, the shared dyadic tolerance ledger, the two StreamName windows, the two RegSeqRat readbacks, the finite comparison ledger, the two terminal Real seals, componentwise transport, continuation replay, package provenance, and local naming through N.*

Proof. Project the carrier in Definition 7865.1. The two compared rows are displayed as A and B, and the only tolerance coordinate available to either side is the shared row D.

The schedule windows W_A, W_B and readbacks R_A, R_B are opened after that shared ledger is visible. The comparison row Q therefore records finite inspected readbacks under one tolerance request, rather than under unrelated representatives.

The seals E_A, E_B occur downstream of the same comparison route. The rows H, K, P, N transport, replay, source, and name that displayed packet, and no excluded quotient, choice, trichotomy, or ambient-completeness coordinate occurs in the carrier. □

Theorem 7865.3 (Cauchy regular comparison seal stability)**.** *Every comparison-facing consumer of an accepted* CauchyRegularComparisonSeal$^{\uparrow}$ *carrier reads the route*

$$A, B, D \longmapsto W_A, W_B \longmapsto R_A, R_B \longmapsto Q \longmapsto E_A, E_B, H, P, N.$$

Componentwise hsame *transport preserves this route: replacing any displayed row by an* hsame*-transported row changes neither the shared tolerance ledger nor the order in which the schedule windows, readbacks, comparison ledger, and terminal Real seals become available.*

Proof. Start with a comparison request. By the carrier definition the request first meets the two regular rows and the common dyadic tolerance ledger D.

The StreamName windows and RegSeqRat readbacks are then attached to that same ledger. The comparison ledger Q can only inspect readback entries already exposed by those windows, so it cannot introduce quotient equality or a selected representative.

Finally, hsame transport is recorded componentwise by H and continuation replay by K. These rows preserve the displayed order of access before E_A and E_B are read, which gives the stated stability boundary. □

Closure status for NameCert$_{\text{CauchyRegularComparisonSeal}^{\uparrow}}$**.**

Constructive story: A CauchyRegularComparisonSeal$^{\uparrow}$ packet is generated from two regular Cauchy rows, one shared DyadicRatCore tolerance ledger, two StreamName windows, two RegSeqRat readbacks, a finite comparison ledger, two terminal RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRegularComparisonSeal}\uparrow}$, seed)

Scope closed: Definition 7865.1, Definition 7865.2, and Definition 7865.3 seed the finite regular-Cauchy comparison route over StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRegularComparisonSeal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, countable choice, apartness trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared dyadic tolerance accounting, two-sided StreamName window routing, RegSeqRat readback, finite comparison stability, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRegularComparisonSealUp.

Chapter 7866

A Concrete Naming Certificate for $\mathsf{EulerPhi}^{\uparrow}$

Following Hardy and Wright on elementary number theory, the $\mathsf{EulerPhi}^{\uparrow}$ packet records Euler's totient function as a finite coprime-count certificate rather than as an ambient arithmetic oracle. It sits over the natural-number divisor rows of $\mathsf{Nat}^{\uparrow}$, the prime and coprimality surface of chapter 64, the rational denominator route of chapter 37, and the dyadic exponent and denominator ledgers of chapter 291. The packet is intended for rational normalization, dyadic denominator accounting, Real-facing denominator handoffs, and later cyclotomic degree consumers that require a displayed totient row instead of reading totient data from chapter 174.

Definition 7866.1 (EulerPhi carrier)**.** *An* $\mathsf{EulerPhi}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (N, D, C, Q, R, H, T, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *source row naming the inspected natural number and the finite divisor schedule.*
2. *D is the divisor ledger: for every displayed candidate d, it records the bounded Nat divisor test, the complementary quotient row when the divisor test succeeds, and the finite exclusion row when it fails.*
3. *C is the* $\mathsf{Prime}^{\uparrow}$ *and coprime classifier row, read only through the finite prime-factor and gcd witnesses supplied by the prime certificate surface of chapter 64.*
4. *Q is the coprime-count row. It counts exactly the displayed residues whose entries in C certify coprimality with the source row N.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *denominator-normalization handoff row. It exposes the same coprime count to rational denominator consumers after the rational source route of chapter 37 has displayed the denominator row.*
6. *H records componentwise* hsame *transport for the Nat source, divisor ledger, coprime classifier, coprime count, rational denominator handoff, dyadic ledger, replay, provenance, and local naming rows.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponent and denominator ledger used when a denominator consumer restricts the totient row to powers of two or dyadic denominator refinements.*

8. *P is the* Pkg *provenance over* $\mathsf{EulerPhi}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{EulerPhi}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for unbounded search over residues, host integer factorization, selected primitive roots, cyclotomic minimal polynomials, host real equality, or an ambient arithmetic oracle.

Lean4 (statement) `BEDC.Derived.EulerPhiUp`

Theorem 7866.2 (EulerPhi NameCert obligations)**.** *For every accepted* $\mathsf{EulerPhi}^{\uparrow}$ *carrier* $E = (N, D, C, Q, R, H, T, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{EulerPhi}^{\uparrow}}$ *obligation surface consists exactly of the Nat source, finite divisor ledger, Prime coprime classifier, coprime-count row, Rat denominator-normalization handoff, DyadicRatCore exponent and denominator ledger, componentwise transport, displayed replay, provenance, and local naming rows. Every consumer must read the finite route*

$$N \longrightarrow D \longrightarrow C \longrightarrow Q$$

before it may use the rational or dyadic handoff rows, and no consumer may replace that route by unbounded residue search, host integer factorization, selected primitive roots, cyclotomic minimal-polynomial data, host real equality, or an ambient arithmetic oracle.

Proof. Project the carrier of Definition 7866.1. The source row is N, the divisor schedule is D, the coprime classifier is C, and the displayed count is Q.

The prime and coprime information is then read through chapter 64. The carrier does not expose a separate factorization oracle; it exposes only the finite prime and gcd witnesses already recorded in C.

Finally inspect the handoff rows. The rational denominator row R and the dyadic ledger T occur after the count row Q, while H, replay, provenance, and L only transport or repack the same displayed route. Thus every downstream use is forced through the finite Nat, divisor, coprime, and count rows named above. □

Theorem 7866.3 (EulerPhi coprime-denominator handoff)**.** *Every* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$*, or Real-facing denominator consumer of an accepted* $\mathsf{EulerPhi}^{\uparrow}$ *carrier receives the totient data only through the packet*

$$N, D, C, Q, R, H, T, P, L.$$

In particular, a rational denominator normalization may read R only after N, D, C, Q have displayed the finite coprime-count route, and a dyadic exponent or denominator refinement may read T only as the same route repacked for chapter 291. The handoff exports no primitive-root selector, cyclotomic minimal polynomial, host factorization theorem, host real equality, or infinite residue enumeration.

Proof. Begin with a denominator-facing read. The rational route of chapter 37 can use only the displayed denominator row, so the $\mathsf{EulerPhi}^{\uparrow}$ packet supplies R as a handoff after the Nat source, divisor ledger, coprime classifier, and count row have been opened.

For dyadic consumers, use the dyadic denominator and exponent discipline of chapter 291. The row T is not a new arithmetic source; it is the same finite coprime-count route repacked for powers-of-two and denominator-refinement ledgers.

The remaining rows H, P, L and the displayed replay data preserve provenance and local naming. Since neither the carrier nor the preceding NameCert obligation includes primitive roots,

minimal polynomials, host factorization theorems, host real equality, or infinite residue enumeration, no denominator consumer receives those objects from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerPhi}\uparrow}$**.**

Constructive story: An $\mathsf{EulerPhi}^{\uparrow}$ packet is generated from a Nat source row, finite divisor ledger, Prime coprime classifier, coprime-count row, Rat denominator-normalization handoff, DyadicRatCore exponent and denominator ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerPhi}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7866.1, Definition 7866.2, and Definition 7866.3 seed the finite coprime-count and denominator-handoff packet over displayed Nat, Prime, Rat, and DyadicRatCore rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerPhi}\uparrow}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim host integer factorization, primitive-root selection, cyclotomic minimal-polynomial data, host real equality, complete Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, divisor-ledger exactness, coprime-count stability, rational denominator handoff, dyadic denominator handoff, and transport stability for BEDC.Derived.EulerPhiUp.

Chapter 7867

A Concrete Naming Certificate for $\mathsf{KhalimskyLine}^{\uparrow}$

Following Khalimsky and digital topology, the $\mathsf{KhalimskyLine}^{\uparrow}$ packet records the integer line with its alternating finite-open topology as NameCert data rather than as a host topological space. It sits after the topology carrier of chapter 88, the observable-open classifier of chapter 1860, and the specialization-order carrier of chapter 1378. The packet exposes the finite local stars around integer rows, their Sierpinski truth reads, and the induced specialization comparison as one displayed surface.

Definition 7867.1 (Khalimsky line carrier)**.** *A* $\mathsf{KhalimskyLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Z, T, S, O, H, C, P, N)$$

with the following displayed rows:

1. *Z is the integer-site row, read as finite named windows of adjacent sites rather than as an external set of integers.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row whose basic opens are the Khalimsky stars: odd sites are open singletons, while even sites are read through the three-site window consisting of the site and its two neighbours.*
3. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row recording membership of a displayed site in a displayed basic open.*
4. *O is the* $\mathsf{Order}^{\uparrow}$ *specialization row generated by the finite rule that an even site specializes to its adjacent odd sites and every site specializes to itself.*
5. *H records componentwise* hsame *transport for the site, topology, Sierpinski, specialization-order, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each local-open or specialization request through Z, T, S, O, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{KhalimskyLine}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KhalimskyLine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient integer topology, host subset equality, excluded middle over arbitrary opens, selected neighbourhood filters, quotient specialization equality, or Lean verification.

Theorem 7867.2 (Khalimsky line NameCert obligations)**.** *For every accepted* $\mathsf{KhalimskyLine}^{\uparrow}$ *carrier*

$$K = (Z, T, S, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KhalimskyLine}^{\uparrow}}$ *surface consists exactly of carrier admission for the integer-site row, the Khalimsky-star topology row, the Sierpinski observable-open row, the specialization-order row, componentwise transport, displayed replay, provenance, and local naming; local-star exactness for odd and even sites; Sierpinski membership exactness for displayed basic opens; specialization exactness for the reflexive and adjacent even-to-odd comparisons; and non-escape of all topology and order reads through the same finite packet.*

Proof. Project the carrier of Definition 7867.1. The accepted packet displays exactly the site row, topology row, observable-open row, specialization-order row, and structural rows.

The local-star clauses are finite. Odd sites expose a singleton basic open, while even sites expose the three-site adjacent window. The Sierpinski row reads only those displayed opens, so open membership is a truth row in the packet rather than a host proposition.

The specialization row is then read from the same local stars. Reflexivity is supplied at every displayed site, and the nontrivial comparisons are precisely the adjacent even-to-odd incidences forced by containment of observable neighbourhoods.

Finally H, C, P, N preserve the site, topology, Sierpinski, and order rows together. Since no carrier coordinate stores arbitrary open subsets, selected neighbourhood filters, host subset equality, quotient specialization equality, or excluded middle, none becomes a NameCert obligation. □

Theorem 7867.3 (Khalimsky line specialization handoff)**.** *Every* $\mathsf{Order}^{\uparrow}$ *or digital-topology consumer of an accepted* $\mathsf{KhalimskyLine}^{\uparrow}$ *carrier reads specialization through the displayed route*

$$Z \longrightarrow T \longrightarrow S \longrightarrow O.$$

Thus the exported order surface is the finite Khalimsky specialization ledger generated by local stars and Sierpinski membership rows, not an ambient order on integers, an arbitrary Alexandrov topology, a selected neighbourhood filter, host subset equality, or quotient specialization equality.

Proof. Begin with an order-facing read. The carrier exposes no specialization comparison before the site row Z and topology row T display the relevant local stars.

Read the Sierpinski row S next. It turns membership in each displayed basic open into observable truth data, so the consumer can test only the finite local-star rows carried by the packet.

The specialization row O is downstream of those observable-open reads. It records reflexive comparisons and adjacent even-to-odd comparisons exactly when every displayed open observation around the upper site is respected by the lower site.

Therefore any order or digital-topology consumer receives the route Z, T, S, O, H, C, P, N. The absent host objects are never coordinates of the packet and cannot be exported by transport, replay, provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{KhalimskyLine}^{\uparrow}}$.

Constructive story: A $\mathsf{KhalimskyLine}^{\uparrow}$ packet is generated from integer-site windows, Khalimsky-star topology rows, Sierpinski observable-open rows, specialization-order rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KhalimskyLine}^{\uparrow}}$, seed)

Scope closed: Definition 7867.1, Definition 7867.2, and Definition 7867.3 seed the finite Khalimsky-line topology and specialization-order route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KhalimskyLine}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close an ambient integer topology, arbitrary open-set equality, selected neighbourhood filters, quotient specialization equality, excluded middle over opens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-star exactness, Sierpinski membership exactness, specialization-order exactness, topology-order nonescape, and transport stability for BEDC.Derived.KhalimskyLineUp.

Chapter 7868

A Concrete Naming Certificate for $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$

Following Arzela and Ascoli, the $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ packet records the finite compact-family closure data needed before an equicontinuous family may be read as closed under compact-source limit operations. It sits after the Arzela–Ascoli surface of chapter 412, the compact-metric source rows of chapter 129, the equicontinuity packet of chapter 4076, the uniform-modulus consumer of chapter 369, the continuous-map rows of chapter 130, and the terminal real-name seal of chapter 38. The packet names a compact-family closure route for later calculus and compactness consumers; it does not claim full function-space compactness, selected subsequences, quotient map equality, or ambient completeness.

Definition 7868.1 (Equicontinuous compact closure carrier)**.** *An* $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, A, Q, U, F, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite compact windows and metric comparison data through which source points are read;*
2. *A is the Arzela–Ascoli dependency row, read through the finite selection and compact-family surfaces already named by chapter 412;*
3. *Q is the* $\mathsf{Equicontinuity}^{\uparrow}$ *row for the family, exposing shared tolerance and source-radius data rather than separate moduli for each downstream read;*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed by closure consumers after the compact-source and equicontinuity rows have been displayed;*
5. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger for the displayed family members and their pointwise graph rows;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for target-side values, endpoint comparisons, and real-valued tolerance reads;*
7. *H records componentwise* hsame *transport for compact source data, Arzela–Ascoli rows, equicontinuity rows, uniform-modulus data, continuous-map rows, Real seals, and local naming;*
8. *C records displayed* Cont *replay for the same dependency route;*

9. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousCompactClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function-space compactness, selected convergent subsequences, quotient equality of maps, choice-selected limits, detached Real values, or ambient compactness and completeness.

Theorem 7868.2 (Equicontinuous compact closure NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ *carrier*

$$E = (K, A, Q, U, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuousCompactClosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact-source exposure, requiring every closure read to display the* $\mathsf{CompactMetric}^{\uparrow}$ *row K before any family or Real-facing conclusion is exported;*
2. *Arzela–Ascoli route exposure, requiring the row A to remain tied to the finite Arzela–Ascoli packet rather than to a host function-space theorem;*
3. *equicontinuity and uniform-modulus binding, requiring Q and U to be read as one shared compact-family modulus route;*
4. *continuous-map and Real sealing, requiring F and R to expose the displayed graph rows and terminal Real seals before target-side closure data are read;*
5. *transport, replay, provenance, and local naming through H, C, P, N;*
6. *ledger non-escape, excluding full function-space compactness, selected subsequences, quotient map equality, choice-selected limits, detached Real pairs, Lean verification, bridge checking, and ambient completeness.*

Proof. Project the carrier of Definition 7868.1. Its only rows are the compact source, Arzela–Ascoli dependency, equicontinuity surface, uniform-modulus handoff, continuous-map ledger, Real seal, transport, replay, provenance, and local naming.

The compact-family closure read first consumes K, A, Q, U, F, R. The compact source row supplies finite source windows, the Arzela–Ascoli row supplies the family-compactness route, the equicontinuity and uniform-modulus rows provide the shared tolerance control, and the continuous-map and Real rows expose only displayed graph and target-value data.

Finally replay H, C, P, N. These rows preserve the same finite route under transport and provenance. Since the carrier contains no host function-space compactness, selected subsequence, quotient map equality, choice-selected limit, detached Real row, Lean verification, bridge-checking row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Any consumer that exports an $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ closure read while omitting $K, A, Q, U, F, R, H, C, P$, or the local NameCert row refutes the carrier of Definition 7868.1.

Independence witness: arzelaascoli, compactmetric, equicontinuity, uniformmodulus, continuousmap, real: none alone is bijective with $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ because the packet

jointly carries compact-source windows, Arzela–Ascoli family rows, shared equicontinuity data, a uniform-modulus handoff, continuous-map graph rows, Real seals, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuousCompactClosure}\uparrow}$.

Constructive story: An $\mathsf{EquicontinuousCompactClosure}^{\uparrow}$ packet is generated from CompactMetric source rows, Arzela–Ascoli dependency rows, Equicontinuity rows, UniformModulus handoff rows, ContinuousMap graph rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompactClosure}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7868.1 and Definition 7868.2 seed the finite compact-family closure surface over CompactMetric, ArzelaAscoli, Equicontinuity, UniformModulus, ContinuousMap, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompactClosure}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full function-space compactness, selected subsequences, quotient map equality, choice-selected limits, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, Arzela–Ascoli route exposure, equicontinuity and uniform-modulus binding, continuous-map and Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.EquicontinuousCompactClosureUp.

Chapter 7869

A Concrete Naming Certificate for $\mathsf{CompactUniformCoverMembership}^{\uparrow}$

Following the finite-cover reading of chapter 9237, the $\mathsf{CompactUniformCoverMembership}^{\uparrow}$ packet records the cover-membership ledger that must be displayed before a compact-uniform-continuity proof folds local radii into a uniform modulus. It sits between the compact metric source of chapter 129, the pointwise-modulus surface of chapter 130, the finite subcover ledger of chapter 6125, and the cover-read consumer of chapter 6409. The packet isolates the finite membership test for a source point against a bundled compact-net center; it does not export the final lower-bound fold or the completed $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Definition 7869.1 (Compact-uniform cover-membership carrier)**.** *A* $\mathsf{CompactUniformCoverMembership}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, L, B, x, c, \rho, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the finite compact-net data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the displayed map at compact-net centers.*
3. *L is the* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *row that records which bundled cover entries are available.*
4. *B is the* $\mathsf{ProbeBundle}$ *index row naming the finite cover entry that is inspected.*
5. *x is the source-point history being routed through the compact cover.*
6. *c is the displayed center row attached to the bundled cover entry.*
7. *ρ is the positive radius row attached to c by the pointwise-modulus data.*
8. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row recording that the inspected source point lies inside the displayed radius window.*
9. *H records componentwise* hsame *transport for compact-net, pointwise-modulus, finite-subcover, ProbeBundle, point, center, radius, comparison, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from compact-net coverage through the bundled center and radius membership test.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCoverMembership}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCoverMembership}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an external point-set cover, a selected subcover, a global modulus, quotient point equality, countable choice, or ambient compactness.

Theorem 7869.2 (Compact-uniform cover-membership obligations)**.** *For every accepted* CompactUniformCoverMe *carrier*

$$M = (K, F, L, B, x, c, \rho, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformCoverMembership}^{\uparrow}}$ *surface consists exactly of compact-net admission through K, pointwise-modulus admission through F, finite-subcover ledger admission through L, bundled cover-entry admission through B, source point and center exposure through x, c, radius exposure through ρ, rational membership comparison through Q, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7869.1. The accepted packet displays exactly the compact-net row, pointwise-modulus row, finite-subcover ledger, bundled cover entry, inspected point, center, radius, rational comparison, and structural rows.

The membership test is local to B, x, c, ρ, Q. The bundle row names the finite cover entry, the source-point and center rows name the comparison endpoints, the radius row supplies the local modulus window, and the rational comparison row records that the inspected point is inside that window.

The compact and continuous rows are dependency rows, not hidden selectors. The compact row supplies only the finite-net source for the bundle, and the continuous row supplies only the radius attached to the displayed center.

Finally H, C, P, N preserve the same finite membership packet. Since no row stores an external point-set cover, selected subcover, global modulus, quotient point equality, countable choice, or ambient compactness theorem, none can be exported by the local NameCert. □

Theorem 7869.3 (Cover membership before uniform-modulus cover read)**.** *Every* $\mathsf{CompactUniformCoverRead}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer using an accepted* $\mathsf{CompactUniformCoverMembership}^{\uparrow}$ *packet must first display the route*

$$K, L, B, x, c, \rho, Q$$

before it may fold radii or consume a tolerance-only modulus. The handoff supplies only a finite membership row for one bundled compact-net center, not an external subcover selector, a completed lower-bound fold, quotient point equality, countable choice, or an ambient compactness theorem.

Proof. Start with the compact source K and finite-subcover ledger L. They determine the finite cover entries that may be inspected by the compact-uniform route.

Choose only a displayed bundle entry B. The packet then exposes the inspected source point x, the bundled center c, and the radius row ρ attached to that center by the pointwise-modulus row F.

Read the rational comparison Q. This row is the membership evidence needed before the cover-read chapter can fold radii across the finite bundle.

The packet stops at membership. The lower-bound fold and final tolerance-only consumer remain downstream in chapter 6409, so this chapter exports no selected subcover, global modulus, quotient point equality, countable choice, or ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCoverMembership}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCoverMembership}^{\uparrow}$ packet is generated from compact-net rows, pointwise-modulus rows, finite-subcover ledger rows, ProbeBundle entries, inspected source points, center rows, radius rows, rational comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCoverMembership}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7869.1, Definition 7869.2, and Definition 7869.3 seed the finite cover-membership route before compact-uniform cover reads fold local radii.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCoverMembership}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subcovers, completed lower-bound folds, global modulus selection, quotient point equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net source exactness, pointwise-radius exposure, finite-subcover ledger admission, bundle membership comparison, cover-read handoff, and transport stability for BEDC.Derived.CompactUniformCoverMembershipUp.

Chapter 7870

A Concrete Naming Certificate for $\mathsf{CurvedSpacetime}^{\uparrow}$

Following Wald and Choquet-Bruhat on Lorentzian geometry, the $\mathsf{CurvedSpacetime}^{\uparrow}$ packet records the finite BEDC route by which local Lorentz-rate charts, chart transitions, curvature witnesses, and transport rows may be read as a curved-spacetime interface. It realizes the curved-gluing step of the Inter-Hist Locality chain of Definition 9419.25 by sitting over the multi-Hist configuration surface of chapter 474, the causal rows of chapter 478, the maximum-rate boundary of chapter 594, the frame-rate packet of chapter 617, and the symmetric distance-shape ledger of chapter 911. The packet names finite chart and transition data; it does not import a host smooth manifold, global frame, metric tensor primitive, selected atlas, coordinate equality, or ambient spacetime object.

Definition 7870.1 (Curved spacetime carrier)**.** *A* $\mathsf{CurvedSpacetime}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, C, R, L, G, K, T, H, Q, P, N)$$

where:

- M *is the displayed* $\mathrm{MultiHistConfig}^{\uparrow}$ *source row.*
- C *is the* $\mathsf{CrossHistCausal}^{\uparrow}$ *causal-dependence family over the displayed histories.*
- R *is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *maximum-rate boundary row.*
- L *is the local* $\mathsf{LorentzFrameRate}^{\uparrow}$ *chart-rate row.*
- G *is the* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *local distance-shape row.*
- K *is the finite curvature ledger, recording which chart overlaps have non-trivial transition curvature data.*
- T *is the finite transition ledger between chart rows.*
- H *is componentwise* hsame *transport for the multi-Hist, causal, rate, Lorentz, geometry, curvature, and transition rows.*
- Q *is the displayed* Cont *replay route for chart and overlap inspections.*
- P *is the* Pkg *provenance over* $\mathsf{CurvedSpacetime}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{LorentzFrameRate}^{\uparrow}$, $\mathsf{MinkowskiRateGeometry}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
- N *is the local* $\mathsf{NameCert}_{\mathsf{CurvedSpacetime}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite curved-spacetime packet. It has no coordinate for a host manifold, global inertial frame, metric tensor primitive, atlas choice, point equality across charts, or ambient Lorentzian completion.

Theorem 7870.2 (Curved spacetime NameCert obligations)**.** *For every accepted* $\mathsf{CurvedSpacetime}^{\uparrow}$ *carrier* $S = (M, C, R, L, G, K, T, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CurvedSpacetime}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed multi-Hist source, causal family, maximum-rate boundary, local Lorentz-rate chart rows, Minkowski-rate geometry rows, finite curvature ledger, finite transition ledger, componentwise transport, continuation replay, package provenance, and local naming row. Every accepted curved-spacetime read is classified only by those finite rows.*

Proof. Project the carrier of Definition 7870.1. The displayed tuple already exposes the only rows available to a consumer.

The local naming row N can certify exactly the rows named by the carrier: $M, C, R, L, G, K, T, H, Q, P$. Thus carrier admission, chart-rate exposure, transition exposure, curvature exposure, transport, replay, provenance, and local naming are the complete obligation surface.

Since no coordinate in the packet names a host manifold, global frame, primitive metric tensor, selected atlas, cross-chart point equality, or ambient Lorentzian object, no such object can be exported by the local certificate. □

Theorem 7870.3 (Curved spacetime local Lorentz stability)**.** *Every local chart read of an accepted* $\mathsf{CurvedSpacetime}^{\uparrow}$ *packet factors through the route*

$$M, C, R, L, G, H, Q, P, N.$$

In particular, local Lorentz-rate stability is read only through the displayed multi-Hist source, causal-dependence family, maximum-rate row, local Lorentz-frame-rate row, Minkowski-rate geometry row, componentwise hsame *transport, continuation replay, provenance, and local naming. It exports no global frame, synchronized clock, host real-valued coordinate system, or detached metric tensor.*

Proof. Start from the carrier route. The local Lorentz chart row L is not standalone: it is paired with the displayed multi-Hist source, causal rows, and maximum-rate boundary before any geometry row G can be read.

Componentwise transport through H keeps the same route stable under hsame comparison of the visible rows. Replay through Q and provenance through P preserve the local chart inspection as a finite continuation path named by N.

Therefore every local Lorentz-rate read is confined to the displayed route, with no coordinate for a global inertial frame, synchronized clock, host coordinate system, or primitive metric tensor. □

Theorem 7870.4 (Curved spacetime no global frame boundary)**.** *Every transition, curvature, or transport read exported by an accepted* $\mathsf{CurvedSpacetime}^{\uparrow}$ *packet must pass through the finite overlap route*

$$M, C, R, L, G, K, T, H, Q, P, N$$

before it can serve as a curved-spacetime interface. The overlap route may expose non-trivial chart gluing and curvature ledgers, but it never exports a single global frame, selected atlas, quotient point equality across charts, host manifold equality, metric tensor primitive, countable-choice chart schedule, or ambient Lorentzian spacetime.

Proof. The transition ledger T and curvature ledger K are rows of the same accepted carrier, so every overlap read begins with the displayed multi-Hist, causal, rate, Lorentz, and local geometry rows.

Apply the NameCert obligation theorem Definition 7870.2. It forces the transition and curvature rows to remain inside the finite carrier, transported by H, replayed by Q, sourced by P, and named by N.

The carrier has no field for a global frame, selected atlas, quotient point equality, host manifold equality, metric tensor primitive, countable-choice chart schedule, or ambient Lorentzian spacetime. Hence the public boundary is exactly the finite curved-gluing route. □

Falsifiable prediction: Every accepted $\mathsf{CurvedSpacetime}^{\uparrow}$ read factors through finite multi-Hist chart rows, causal rows, maximum-rate rows, local Lorentz-frame-rate rows, Minkowski-rate geometry rows, curvature and transition ledgers, hsame transport, Cont replay, Pkg provenance, and local naming before any curved-spacetime consumer receives the interface.

Independence witness: multihistconfig, crosshistcausal, maxcausalrate, lorentz-frame-rate, minkowski-rate-geometry: $\mathsf{CurvedSpacetime}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local chart rows, non-trivial transition data, curvature ledgers, transport, replay, provenance, and local naming while refusing a global frame, selected atlas, host manifold equality, metric tensor primitive, and ambient spacetime object.

Closure status for $\mathsf{NameCert}_{\mathsf{CurvedSpacetime}^{\uparrow}}$.

Constructive story: A $\mathsf{CurvedSpacetime}^{\uparrow}$ packet is generated from finite multi-Hist chart rows, cross-Hist causal rows, maximum-rate boundaries, local Lorentz-frame-rate rows, Minkowski-rate geometry rows, curvature and transition ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CurvedSpacetime}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7870.1, Definition 7870.2, Definition 7870.3, and Definition 7870.4 bind the finite curved-gluing carrier and its no-global-frame boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CurvedSpacetime}^{\uparrow}}, \mathsf{unformalized})$

Lean target:

Bridge status: none

Not claimed: This chapter does not close a host smooth manifold, global frame, selected atlas, quotient point equality, metric tensor primitive, ambient Lorentzian spacetime, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local Lorentz stability, transition and curvature readback, transport, replay, provenance, and local naming for the $\mathsf{CurvedSpacetime}^{\uparrow}$ packet.

Chapter 7871

A Concrete Naming Certificate for $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$

Following Bishop-style constructive nested interval reasoning, the $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ packet records the finite dyadic window route by which nested located intervals yield regular rational readback and a terminal real-name handoff. It sits after the dyadic arithmetic surface of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, the interval nesting route of chapter 320, and the located endpoint discipline of chapter 394. The packet names a displayed nested-interval-to-real carrier; it is not a selected limit, quotient real equality, countable-choice branch, trichotomy principle, or ambient completeness theorem.

Definition 7871.1 (Dyadic nested interval carrier)**.** *A* $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, J, K, S, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the finite row of dyadic left endpoints, classified through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *U is the finite row of dyadic right endpoints, over the same endpoint window as L.*
3. *J is the nesting row asserting that each displayed successor interval lies inside its predecessor.*
4. *K is the located nonempty-window row, recording that each displayed interval has a certified dyadic inhabitance witness or overlap certificate internal to the packet.*
5. *S is the shrink schedule row that assigns a displayed precision request to a finite interval stage.*
6. *A is the diameter ledger, proving that the displayed dyadic width at the scheduled stage is bounded by the requested tolerance.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *read window that records the inspected finite stages.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled dyadic endpoints or midpoints over W.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after L, U, J, K, S, A, W, R have been replayed.*
10. *H records componentwise* hsame *transport for endpoint, nesting, located-window, shrink, diameter, StreamName, RegSeqRat, Real handoff, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a precision request through* $L, U, J, K, S, A, W, R, E, H, P, N$.
12. *P is the* Pkg *provenance over* $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a selected limit point, quotient real equality, countable-choice subsequence, external interval oracle, trichotomy split, or ambient completeness principle.

Theorem 7871.2 (Dyadic nested interval NameCert obligations)**.** *For every accepted* DyadicNestedIntervalChoiceFr *carrier $D = (L, U, J, K, S, A, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}}$ *obligation surface is exactly endpoint admission through L, U, nesting through J, located nonempty-window exposure through K, shrink scheduling through S, diameter control through A,* $\mathsf{StreamName}^{\uparrow}$ *windowing through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

No downstream consumer may replace this surface by selected limits, quotient-real equality, a countable-choice branch, host interval completeness, trichotomy, or an external locatedness predicate.

Proof. Project the carrier of Definition 7871.1. The endpoint rows L, U are the only dyadic data, while J, K, S, A are the displayed interval, locatedness, schedule, and diameter rows attached to them.

Read the analytic handoff through W, R, E. The StreamName row exposes only inspected finite stages, the RegSeqRat row repacks those stages as regular rational readback, and the Real handoff appears only after those rows have been replayed.

Finally inspect H, C, P, N. Transport, replay, provenance, and local naming preserve the same finite route, so the NameCert surface has no coordinate for selected limits, quotient-real equality, countable choice, host completeness, trichotomy, or external locatedness. □

Theorem 7871.3 (Dyadic nested interval window induction)**.** *For every accepted* $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ *carrier, finite induction on the displayed* $\mathsf{StreamName}^{\uparrow}$ *window preserves the route*

$$L, U, J, K, S, A, W, R, E, H, C, P, N.$$

The base window exposes one located dyadic interval with its diameter ledger, and the successor step refines the previous interval through the nesting row J while keeping the same shrink schedule, readback, transport, replay, provenance, and local naming rows.

Proof. Use induction on the finite window row W. In the base case, the first displayed interval is certified by L, U, K, and its width is bounded by the corresponding diameter row A.

For the successor step, append one inspected stage. The nesting row J places the new dyadic interval inside the previous displayed interval, and the located row K keeps the successor window nonempty inside the packet.

The schedule and readback rows are unchanged in kind. The row S chooses the displayed stage for the requested tolerance, A supplies the width bound at that stage, and R records the same finite chain as regular rational readback.

The structural rows H, C, P, N transport, replay, source, and name the whole route. Hence the induction never introduces a selected limit, quotient equality, external branch, or ambient completeness coordinate. □

Theorem 7871.4 (Dyadic nested interval Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ *packet must first expose the dyadic endpoint rows* L, U*, nesting row* J*, located nonempty-window row* K*, shrink schedule* S*, diameter ledger* A*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. The terminal handoff* E *is therefore a displayed regular-rational route to* $\mathsf{Real}^{\uparrow}$*, not a selected point of the intersection and not quotient-real equality.*

Proof. Apply Definition 7871.2. It makes E downstream of the endpoint, nesting, located-window, schedule, diameter, StreamName, and RegSeqRat rows.

Use Definition 7871.3. The finite windows used by R are preserved across displayed stages, so the Real handoff reads a regular rational route rather than an external intersection selector.

Finally use the refusal part of the carrier definition. Since the accepted packet has no selected limit, quotient-real equality, countable-choice branch, trichotomy split, external interval oracle, or ambient completeness row, every $\mathsf{Real}^{\uparrow}$ consumer must take the displayed handoff E through the finite route. □

Falsifiable prediction: An accepted $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ handoff without displaying dyadic endpoints, nesting, located nonempty windows, shrink schedule, diameter ledger, StreamName windows, and RegSeqRat readback refutes the packet surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, nestedinterval, locatedinterval: no listed sibling is bijective to $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ because only this packet binds dyadic endpoints, nesting, located windows, shrink scheduling, diameter control, finite observation, regular rational readback, Real handoff, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}$ packet is generated from dyadic endpoint rows, nesting rows, located nonempty-window rows, a shrink schedule, a diameter ledger, StreamName windows, RegSeqRat readback, a RealUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7871.1, Definition 7871.2, Definition 7871.3, and Definition 7871.4 seed the finite dyadic nested-interval route from displayed windows to a Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalChoiceFree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected intersection points, quotient Real equality, countable choice, trichotomy, host interval completeness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, nesting, located-window exposure, shrink scheduling, diameter control, StreamName window stability, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicNestedIntervalChoiceFreeUp.

Chapter 7872

A Concrete Naming Certificate for $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ packet records the finite rate-envelope surface by which a regular Cauchy read answers dyadic precision requests before a terminal real-name seal is consumed. It sits over the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the tail-control handoff of chapter 1487, and the terminal real-name boundary of chapter 38. The packet names a finite envelope for rate requests; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7872.1 (Regular Cauchy rate-envelope carrier)**.** *A* $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, W, R, T, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rate-request ledger, recording the dyadic precision asked of the regular Cauchy name and the tolerance cells used by the envelope.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened for that dyadic request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window W under the same tolerance ledger D.*
4. *T is the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *handoff row, asserting that the inspected window dominates every later read at the requested dyadic rate.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, public only after D, W, R, T have been replayed.*
6. *H records componentwise* hsame *transport for the dyadic request ledger, stream window, regular readback, tail handoff, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite rate request through*

$$D, W, R, T \longmapsto S, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient equality of streams, a selected representative, countable-choice tail selection, host real equality, or an ambient completeness field for $\mathsf{Real}^{\uparrow}$.

Theorem 7872.2 (Regular Cauchy rate-envelope NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ *carrier*

$$E = (D, W, R, T, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRateEnvelope}^{\uparrow}}$ *surface consists exactly of dyadic rate-request admission, finite StreamName window exposure, RegSeqRat readback, RegularCauchyTailEstimate handoff, terminal Real sealing, componentwise transport, displayed continuation replay, package provenance, and local naming through* N.

Proof. Project the carrier of Definition 7872.1. The dyadic request row D is the only rate coordinate, and the finite window row W is opened under that request.

Read the regular rational data through R. The tail handoff T is attached to the same D, W, R surface, so it records domination of later reads at the displayed rate rather than a detached convergence predicate.

The terminal Real seal S occurs after the rate ledger, window, readback, and tail handoff. The rows H, C, P, N transport, replay, source, and name this displayed packet, and no excluded quotient, selected limit, choice, trichotomy, host equality, or ambient-completeness row appears in the carrier. □

Theorem 7872.3 (Regular Cauchy rate-envelope finite-window induction)**.** *Let*

$$E = (D, W, R, T, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of* E *factors through a finite induction on the displayed window* W*: the base window exposes the first inspected rational read under* D*, and each extension appends one displayed StreamName entry whose RegSeqRat readback remains controlled by the same RegularCauchyTailEstimate handoff* T*. Therefore the terminal seal* S *can be reached only through the finite rate-envelope route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyTailEstimate}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Proof. Induct on the finite window row W. In the base case the carrier exposes one inspected StreamName entry, the dyadic request ledger D, and the corresponding RegSeqRat readback R.

For the extension step, append one displayed window entry. The carrier keeps the same request ledger D and the same tail handoff T, so the appended read is still part of the finite rate envelope rather than a separate representative or choice schedule.

After the induction, the consumer reaches S only through D, W, R, T. Componentwise transport H and replay C preserve that order, while P and N only record provenance and naming. Hence no Real-facing read bypasses the finite envelope route. □

Falsifiable prediction: A $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ read is refuted whenever a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal without displaying the dyadic rate request, finite StreamName window, RegSeqRat readback, RegularCauchyTailEstimate handoff, componentwise transport, continuation replay, package provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, regularcauchytailestimate, real: none alone is bijective with $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ because only this packet binds dyadic rate requests, finite StreamName windows, RegSeqRat readback, tail-estimate handoff, Real sealing, transport, replay, provenance, and local naming as one NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRateEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRateEnvelope}^{\uparrow}$ packet is generated from a dyadic rate-request ledger, finite StreamName windows, RegSeqRat readback, a RegularCauchyTailEstimate handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7872.1, Definition 7872.2, and Definition 7872.3 seed the finite rate-envelope route over dyadic requests, StreamName windows, RegSeqRat readback, tail estimates, Real seals, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic rate-request admission, StreamName window stability, RegSeqRat readback, RegularCauchyTailEstimate handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyRateEnvelopeUp.

Chapter 7873

A Concrete Naming Certificate for $\mathsf{HaarMeasure}^{\uparrow}$

Following Haar and Weil, the $\mathsf{HaarMeasure}^{\uparrow}$ packet records the finite NameCert surface for an invariant measure on a displayed locally compact group without taking the host existence theorem as a primitive. It sits after the topological-group rows of chapter 204, the measure carrier of chapter 92, the Radon–Nikodym comparison surface of chapter 3448, the character-duality boundary of chapter 263, and the finite-group averaging packet of chapter 7640. The chapter names only the local rows that an invariant-measure consumer may inspect; it does not construct the ambient Haar existence theorem, quotient measure equality, selected representatives, modular-function theory, or completed locally compact group analysis.

Definition 7873.1 (HaarMeasure carrier)**.** *A* $\mathsf{HaarMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, T, M, L, R, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{TopGroup}^{\uparrow}$ *source row, naming the group operation, identity, inverse, neighbourhood, compact-window, and translation data read before an invariant-measure row may be consumed.*
2. *T is the local compactness and regularity window row, exposing the compact-test and open-neighbourhood data through which measure values are inspected.*
3. *M is the* $\mathsf{Measure}^{\uparrow}$ *row, recording the displayed event family and measure-value readback available to the packet.*
4. *L and R are the left- and right-translation invariance rows, recording how translated compact or measurable windows replay the same measure read.*
5. *W is the comparison and refusal row, tying Radon–Nikodym and Pontryagin-duality consumers to the displayed invariant-measure surface while excluding quotient measure equality, selected representatives, modular-function claims, and host existence theorems.*
6. *H records componentwise* hsame *transport for the topological-group, local-compactness, measure, invariance, comparison, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route from group rows through local compactness, measure readback, translation invariance, comparison, refusal, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HaarMeasure}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonNikodym}^{\uparrow}$, $\mathsf{PontryaginDuality}^{\uparrow}$, $\mathsf{HaarIntegralFiniteGroup}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{HaarMeasure}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host proof of Haar-measure existence, quotient equality of measures, selected compact approximants, an unrestricted modular function, or an ambient completed locally compact group.

Theorem 7873.2 (HaarMeasure NameCert obligations)**.** *For every accepted* $\mathsf{HaarMeasure}^{\uparrow}$ *carrier* $K = (G, T, M, L, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HaarMeasure}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the displayed topological-group, local-compactness, measure, translation-invariance, comparison, refusal, transport, replay, provenance, and local naming rows;*

2. *locally compact measure exposure, requiring G, T, M to be visible before any invariant-measure consumer reads a value;*

3. *left and right invariance, requiring L and R to replay the same displayed measure row under translated windows;*

4. *sibling comparison, requiring Radon–Nikodym, Pontryagin-duality, and finite-group averaging consumers to enter through the displayed comparison row W;*

5. *ledger non-escape, excluding ambient Haar existence, quotient measure equality, selected representatives, unrestricted modular-function claims, and completed locally compact group analysis.*

Proof. Project the carrier of Definition 7873.1. The only group, compactness, and measure data available to a consumer are the displayed rows G, T, M, so invariant-measure reading cannot bypass the local compactness and event-value windows.

Next inspect the translation rows. Rows L and R replay measure values through displayed left and right translates of the same group source row, while H and C preserve this replay componentwise and publicly.

Finally use the comparison and refusal row W together with provenance P and local naming N. The Radon–Nikodym, Pontryagin-duality, and finite-group averaging siblings can consume only the displayed comparison route, and no row supplies host existence, quotient equality, selected representatives, modular-function theory, or completed locally compact group analysis. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaarMeasure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HaarMeasure}^{\uparrow}$ packet is generated from topological-group source rows, local compactness and regularity windows, measure readback rows, left and right translation-invariance rows, comparison and refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HaarMeasure}^{\uparrow}}$, seed)

Scope closed: Definition 7873.1 and Definition 7873.2 seed the locally compact invariant-measure surface over displayed topological-group and measure rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HaarMeasure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient Haar-measure existence, quotient measure equality, selected representatives, unrestricted modular-function theory, or completed locally compact group analysis.

Upgrade path: Obligation closure requires checked carrier admission, locally compact measure exposure, left and right invariance, sibling comparison, transport, replay, provenance, local NameCert coverage, and ledger non-escape for the same packet.

Chapter 7874

A Concrete Naming Certificate for $\mathsf{WeilPetersson}^{\uparrow}$

Following Weil, Petersson, Ahlfors, and Wolpert, the $\mathsf{WeilPetersson}^{\uparrow}$ packet records the finite metric-geometry surface obtained from the L^2 Hermitian pairing on Teichmuller tangent data. It sits next to the marked-surface packet of chapter 6372, the metric carrier of chapter 57, the symplectic carrier of chapter 138, and the complex-structure carrier of chapter 39. The packet is a finite NameCert surface over $\mathsf{TeichmullerSpace}^{\uparrow}$, Riemann-surface coordinate windows, $\mathsf{Met}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert; it is not a host construction of moduli space, a quotient complex structure, a selected geodesic family, or an ambient completeness theorem.

Definition 7874.1 (Weil–Petersson carrier)**.** *A* $\mathsf{WeilPetersson}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, R, Q, \Omega, D, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{TeichmullerSpace}^{\uparrow}$ *source row, including the marked-surface, complex-structure, hyperbolic-metric, deformation, and classifier rows that the metric read may inspect.*
2. *R is the Riemann-surface coordinate-window row, recording the finite chart, quadratic-differential, or Beltrami-window data used by the pairing route.*
3. *Q is the Hermitian pairing row. It records the finite L^2 pairing witness as* $\mathsf{Complex}^{\uparrow}$*-linear data with* $\mathsf{Real}^{\uparrow}$ *readback for the positive metric part.*
4. *Ω is the symplectic and Kahler-form row, read through* $\mathsf{Symplectic}^{\uparrow}$ *and the complex row of T without adding a separate host quotient structure.*
5. *D is the metric-distance readback row. It exposes the local* $\mathsf{Met}^{\uparrow}$ *distance interface obtained from Q, Ω and the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Complex}^{\uparrow}$ *seal row, ensuring that scalar reads pass through the visible real and complex carriers before any metric or symplectic consumer receives them.*
7. *H records componentwise* hsame *transport for $T, R, Q, \Omega, D, S, C, P, N$.*
8. *C records displayed* Cont *replay from Teichmuller source rows through coordinate windows, Hermitian pairing, symplectic/Kahler form, and metric-distance readback.*

9. *P is the* Pkg *provenance over* $\mathsf{WeilPetersson}^{\uparrow}$, $\mathsf{TeichmullerSpace}^{\uparrow}$, *Riemann-surface windows,* $\mathsf{Met}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeilPetersson}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, Q, \Omega, D, S, H, C, P, N$. It has no coordinate for host moduli-space construction, quotient complex-structure equality, selected geodesics, metric completion, curvature formulae, or global analytic compactification.

Theorem 7874.2 (Weil–Petersson NameCert obligations)**.** *For every accepted* $\mathsf{WeilPetersson}^{\uparrow}$ *carrier $W = (T, R, Q, \Omega, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeilPetersson}^{\uparrow}}$ *obligation surface consists exactly of Teichmuller-source exposure through T; Riemann-surface coordinate-window exposure through R; Hermitian pairing admission through Q; symplectic and Kahler-form readback through Ω; metric-distance readback through D;* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Complex}^{\uparrow}$ *scalar sealing through S; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Every consumer of the packet must therefore read the route

$$\mathsf{TeichmullerSpace}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{Symplectic}^{\uparrow},\quad \mathsf{Complex}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

through $T, R, Q, \Omega, D, S, H, C, P, N$, and no consumer receives host moduli space, quotient complex structure, selected geodesics, ambient completeness, global curvature identities, or analytic compactification data.

Proof. Project the carrier of Definition 7874.1. The rows T, R, Q, Ω, D, S display the only mathematical data visible to the local certificate: marked Teichmuller input, finite Riemann-surface windows, Hermitian pairing, symplectic/Kahler form, metric readback, and scalar seals.

The transport and replay rows H, C move only these displayed coordinates. Thus a metric or symplectic consumer can compare packets only by componentwise hsame transport and the recorded Cont route; it cannot introduce a host quotient of moduli points or select geodesics outside the packet.

Finally P names the provenance of the cited sibling packets and N names the local $\mathsf{WeilPetersson}^{\uparrow}$ surface. Since the carrier contains no coordinate for the excluded analytic structures, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{WeilPetersson}^{\uparrow}$ packet displays a TeichmullerSpaceUp source row, Riemann-surface coordinate windows, Hermitian pairing, symplectic/Kahler form, MetricSpaceUp distance readback, RealUp and ComplexUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any metric-geometry consumer reads it.

Independence witness: teichmuller-space, metric, symplectic, complex: $\mathsf{WeilPetersson}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Teichmuller source data, Riemann-surface coordinate windows, Hermitian pairing, symplectic/Kahler form, metric readback, scalar seals, transport, replay, provenance, and local naming in one finite metric-geometry packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeilPetersson}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WeilPetersson}^{\uparrow}}$, seed)

Scope closed: Definition 7874.1 and Definition 7874.2 seed the finite Weil–Petersson NameCert surface over Teichmuller source rows, Riemann-surface coordinate windows, Hermitian pairing, symplectic/Kahler form, metric readback, scalar seals, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WeilPetersson}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: Host moduli-space construction, quotient complex-structure equality, selected geodesics, ambient completeness, global curvature identities, analytic compactification, Lean verification, and bridge checking are not claimed.

Upgrade path: Obligation closure requires checked carrier admission, coordinate-window stability, Hermitian pairing exactness, symplectic/Kahler readback, metric-distance readback, scalar seals, transport, replay, provenance, and local NameCert rows.

Constructive story: A WeilPetersson$^{\uparrow}$ packet is generated from a TeichmullerSpaceUp source row, Riemann-surface coordinate windows, a Hermitian pairing row, a symplectic/Kahler-form row, a MetricSpaceUp distance readback row, RealUp and ComplexUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7875

A Concrete Naming Certificate for $\mathsf{NormedAlgebra}^{\uparrow}$

Following Bourbaki and Rudin on normed algebras, the $\mathsf{NormedAlgebra}^{\uparrow}$ packet records a constructive algebra-over-analysis interface in which multiplication is compatible with a displayed norm. It sits after the normed-space carrier of chapter 1950, the ring multiplication surface of chapter 43, the ordered real algebra rows of chapter 766, the terminal real-name seal of chapter 38, and the complete normed-space consumer boundary of chapter 90. The packet names submultiplicative multiplication and continuous scalar multiplication as finite NameCert data; it is not a quotient algebra, a host topological algebra, a Hahn–Banach principle, or an ambient completeness theorem.

Definition 7875.1 (Normed algebra NameCert surface)**.** *A* $\mathsf{NormedAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (V, +, 0, -, \cdot, 1, \nu, \alpha, \mu, \sigma, H, C, P, N)$$

with the following displayed rows:

1. $V, +, 0, -$ *are the additive and scalar rows read from the* $\mathsf{NormedSpace}^{\uparrow}$ *source.*
2. $\cdot, 1$ *are the multiplication and unit rows read through the* $\mathsf{Ring}^{\uparrow}$ *source and transported over the same carrier classifier.*
3. ν *is the* $\mathsf{Real}^{\uparrow}$*-valued norm row of the normed-space carrier.*
4. α *is the ordered-real algebra row tying scalar multiplication to chapter 766.*
5. μ *is the submultiplicative product ledger, displaying* $\nu(x \cdot y) \leq \nu(x)\nu(y)$ *as a finite ordered-real comparison row.*
6. σ *is the scalar-continuity ledger, displaying the finite norm estimate for scalar multiplication before any Banach or bounded-operator consumer reads it.*
7. H *records componentwise* hsame *transport for carrier, additive, scalar, multiplicative, norm, product-ledger, scalar-continuity, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a normed-algebra request through* $V, +, 0, -, \cdot, 1, \nu, \alpha, \mu, \sigma, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{NormedAlgebra}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{NormedAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient algebra equality, a host topological algebra, a selected completion, Hahn–Banach separation, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7875.2 (Normed algebra submultiplicative product)**.** *Every accepted* $\mathsf{NormedAlgebra}^{\uparrow}$ *multiplication read factors through the shared* $\mathsf{NormedSpace}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before a product norm estimate is public. The only product estimate exposed by the packet is the finite ledger* μ *in Definition 7875.1; it states that the norm of the displayed product is bounded by the displayed ordered-real product of the two input norms. No consumer may replace* μ *by a host algebra norm, quotient equality of products, a selected completion, or ambient completeness.*

Proof. Read the carrier of Definition 7875.1. The rows $V, +, 0, -, \nu$ supply the normed-space source, while $\cdot, 1$ supply the ring multiplication over the same displayed carrier.

The ordered comparison is carried by α and μ. The row α supplies the ordered-real algebra context, and μ records the single finite submultiplicative ledger tying the product row to the two input norm rows.

The structural rows H, C, P, N preserve only the displayed packet. Since no row stores quotient algebra equality, host topological algebra data, a selected completion, Hahn–Banach separation, or ambient completeness, the product estimate is exactly the ledger μ. □

Theorem 7875.3 (Normed algebra continuous scalar multiplication)**.** *Every accepted* $\mathsf{NormedAlgebra}^{\uparrow}$ *scalar-multiplication read factors through the* $\mathsf{NormedSpace}^{\uparrow}$ *scalar rows, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *ordered-scalar row, and the terminal* $\mathsf{Real}^{\uparrow}$ *norm comparison before it reaches a* $\mathsf{Banach}^{\uparrow}$ *or bounded-operator consumer. The handoff is the finite scalar-continuity ledger* σ *together with* H, C, P, N*, not a host continuity theorem or a completed function-space object.*

Proof. Start with the normed-space part of the carrier. The scalar action and norm rows are displayed before any algebra-specific multiplication row is inspected.

Next use the ordered-real algebra row α. It places scalar multiplication in the same ordered $\mathsf{Real}^{\uparrow}$ comparison language as the norm and product ledgers.

The scalar-continuity estimate is then the row σ. It is replayed by C, transported by H, sourced by P, and named by N, so a Banach or bounded-operator consumer receives only the displayed finite estimate.

No excluded analytic object is generated by this route. Host continuity, quotient function equality, selected completion, Hahn–Banach separation, and ambient completeness have no coordinate in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormedAlgebra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NormedAlgebra}^{\uparrow}$ packet is generated from normed-space rows, ring multiplication rows, ordered-real algebra rows, Real norm comparisons, a submultiplicative product ledger, a scalar-continuity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NormedAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 7875.1, Definition 7875.2, and Definition 7875.3 seed the constructive normed-algebra interface over normed-space, ring, ordered-real, Real, and Banach-facing rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NormedAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach algebra completeness, Hahn–Banach separation, spectrum theory, quotient algebra equality, host topological algebra structure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, additive and scalar row exposure, ring multiplication exposure, ordered-real comparison, submultiplicative product exactness, scalar-continuity handoff, transport, replay, provenance, and local naming for BEDC.Derived.NormedAlgebraUp.

Chapter 7876

A Concrete Naming Certificate for $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ packet records finite regular-subsequence cluster evidence downstream of the StreamName certificate of chapter 126, the regular rational sequence certificate of chapter 301, the dyadic rational core of chapter 291, the Real certificate of chapter 38, and the regular Cauchy tail estimate of chapter 1487. The packet names a finite cluster-window surface for a displayed regular subsequence; it does not assert a Bolzano-Weierstrass theorem, choose a cluster point, or import an ambient completeness principle.

Definition 7876.1 (Real regular-subsequence cluster carrier)**.** *A* $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, T, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the subsequence schedule row: a finite* $\mathsf{StreamName}^{\uparrow}$*-indexed list of selected positions, monotone schedule comparisons, and source-window names.*
2. *R is the regular rational readback row: the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint data read at the scheduled positions, with the source classifier and regular readback ledger still visible.*
3. *D is the dyadic tolerance ledger: a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row recording the tolerances used to compare scheduled readbacks inside a finite cluster window.*
4. *T is the tail-estimate row: a* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *packet certifying that the scheduled window is read through displayed finite tail data before any Real-facing seal is consumed.*
5. *W is the cluster-window witness row: a finite family of scheduled positions whose regular rational readbacks are pairwise inside the displayed dyadic tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row: the Real-facing endpoint exposed only after S, R, D, T, W have been displayed.*
7. *H records componentwise* hsame *transport for the schedule, readback, tolerance, tail, window, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, R, D, T, W \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealRegularSubsequenceCluster}^{\uparrow}}$ *row.*

The carrier contains no coordinate for selecting a cluster point, quotienting streams, invoking countable choice, deciding trichotomy, or replacing the finite tail route by ambient Real completeness.

Theorem 7876.2 (Real regular-subsequence cluster NameCert obligations)**.** *For every accepted* $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ *carrier* $K = (S, R, D, T, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealRegularSubsequenceClu}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the subsequence schedule S, regular rational readback R, dyadic tolerance ledger D, tail-estimate row T, cluster-window witness W, terminal Real seal E, transport row H, replay row C, provenance row P, and local naming row N;*

2. *schedule-readback compatibility, requiring every selected position in S to be read through the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*

3. *dyadic-window stability, requiring the cluster-window comparisons in W to use only the tolerance ledger D and the finite tail surface T;*

4. *seal-route exactness, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to be consumed only after the schedule, readback, dyadic tolerance, tail estimate, and cluster window have been displayed;*

5. *ledger non-escape, requiring P and N to source no Bolzano-Weierstrass theorem, selected cluster point, quotient stream equality, countable choice, trichotomy, or ambient Real completeness.*

No cluster obligation is stored outside these rows.

Proof. Project the carrier $K = (S, R, D, T, W, E, H, C, P, N)$. The analytic data available to a consumer are the subsequence schedule, regular rational readback, dyadic tolerance ledger, regular Cauchy tail estimate, finite cluster window, and terminal Real seal.

The schedule-readback clause follows by reading S through the StreamName route and then through the $\mathsf{RegSeqRat}^{\uparrow}$ row R. A selected position has no independent value row unless the corresponding regular rational readback is displayed.

The dyadic-window clause uses D, T, W together. The dyadic ledger records the tolerance, the tail-estimate row records the finite Real-facing route, and the cluster-window witness records only pairwise comparisons inside that displayed tolerance.

The seal-route clause places E after S, R, D, T, W. Componentwise hsame transport and displayed Cont replay preserve the same finite packet, while Pkg provenance and the local NameCert name only those rows.

Finally inspect the non-escape clause. Since the carrier has no row for a selected cluster point, quotient stream equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: A compactness consumer that reads a chosen cluster point or a Bolzano-Weierstrass theorem from $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ refutes this packet.

Independence witness: streamname, regseqrat, dyadicratcore, real, regularcauchytailestimate: $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ is not bijective to any listed sibling because it carries the subsequence schedule, regular readback, dyadic tolerance ledger, tail estimate, cluster-window witness, terminal Real seal, transport, replay, provenance, and local naming rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealRegularSubsequenceCluster}^{\uparrow}}$.

Constructive story: A $\mathsf{RealRegularSubsequenceCluster}^{\uparrow}$ packet is generated from a finite subsequence schedule, regular rational readback, dyadic tolerance ledger, regular Cauchy tail estimate, cluster-window witness, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealRegularSubsequenceCluster}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 7876.1 and Definition 7876.2 bind the finite cluster-window packet over StreamName, RegSeqRat, DyadicRatCore, RegularCauchyTailEstimate, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealRegularSubsequenceCluster}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Bolzano-Weierstrass, selected cluster points, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: To reach scopedClosure, a scope theorem must bind the schedule, readback, tolerance, tail, cluster-window, seal, transport, replay, provenance, and local naming rows as the only accepted packet surface.

Chapter 7877

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailMesh}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy reals, the $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ packet records the finite shared-tail mesh read when several regular rational names are compared before a $\mathsf{Real}^{\uparrow}$ handoff. It sits over the finite stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the cofinal shared-window route of chapter 7532. The packet names a regular Cauchy tail mesh; it does not quotient streams, select limits, invoke trichotomy, import choice, or assert ambient completeness.

Definition 7877.1 (Regular Cauchy tail mesh carrier)**.** *A* $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, Q, T, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by every participating regular name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for every mesh comparison.*
4. *Q is the cofinal shared-tail row read through* $\mathsf{CofinalTailMesh}^{\uparrow}$*.*
5. *T is the regular Cauchy tail row recording the common precision tail on which the readbacks are compared.*
6. *W is the Real-facing window row exposed after S, R, D, Q, T have been displayed.*
7. *H records componentwise* hsame *transport for S, R, D, Q, T, W and for the structural rows.*
8. *C records displayed* Cont *replay from each finite precision request through the stream windows, regular readbacks, dyadic ledger, cofinal tail row, regular tail row, and Real-facing window.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailMesh}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CofinalTailMesh}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, R, D, Q, T, W, H, C, P, N$. It has no coordinate for quotient stream equality, a selected regular Cauchy representative, a trichotomy decision, countable choice, a host cofinal subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7877.2 (Regular Cauchy tail mesh NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ *carrier* $M = (S, R, D, Q, T, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMesh}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window scheduling, RegSeqRat readback, DyadicRatCore tolerance accounting, CofinalTailMesh shared-tail admission, regular Cauchy tail comparison, Real-facing window handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7877.1. The visible inventory is $S, R, D, Q, T, W, H, C, P, N$, and every row has a fixed role in the finite mesh.

The stream and rational rows are read first. Row S opens the inspected finite windows, while R supplies the regular rational readback for those same windows. The row D records the dyadic tolerance ledger against which the participating readbacks are compared.

Next the shared-tail data is displayed. Row Q imports only the cofinal tail-mesh route of chapter 7532, and row T records the regular Cauchy tail comparison used by this packet. The window W is downstream of those rows, so a Real-facing consumer cannot read it before the finite mesh has been exposed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient stream equality, selected representative, trichotomy decision, countable choice, host cofinal subsequence, or ambient completeness coordinate, those objects are not local obligations of the packet. □

Theorem 7877.3 (Regular Cauchy tail mesh Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ *carrier receives the finite route*

$$S \longrightarrow R \longrightarrow D \longrightarrow Q \longrightarrow T \longrightarrow W \longrightarrow H, C, P, N.$$

Thus the consumer reads StreamName windows, RegSeqRat tails, a DyadicRatCore tolerance ledger, the CofinalTailMesh shared-tail row, and the regular Cauchy tail comparison before any terminal Real seal may be consumed.

Proof. Start from the carrier definition. The Real-facing row is W, and it appears only after the stream schedule, regular rational readback, dyadic ledger, cofinal shared-tail row, and regular Cauchy tail row have been displayed.

Apply Definition 7877.2. It makes the route through S, R, D, Q, T part of the local NameCert surface, so a Real consumer cannot replace the route by an external representative, a chosen limit, or a host completeness proof.

The transport and replay rows preserve the same finite order for downstream consumers, while P and N name the provenance and local certificate. The handoff is therefore the displayed mesh route rather than a quotient completion or a choice-based selection of a tail. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ read factors through displayed StreamName schedules, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CofinalTailMesh shared-tail row, a regular Cauchy tail comparison, a Real-facing window, componentwise transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, cofinal-tail-mesh: $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the regular tail comparison and cofinal shared-tail mesh consumed before a Real handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMesh}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailMesh}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CofinalTailMesh shared-tail row, a regular Cauchy tail comparison row, a Real-facing window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchyTailMesh}\uparrow}$, seed)

Scope closed: Definition 7877.1, Definition 7877.2, and Definition 7877.3 seed the finite regular Cauchy shared-tail mesh for Real and Completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyTailMesh}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, countable choice, host cofinal subsequences, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-tail admission, dyadic tolerance accounting, regular tail comparison, Real handoff routing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularCauchyTailMeshUp.

Chapter 7878

A Concrete Naming Certificate for $\mathsf{DiniUniformModulus}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{DiniUniformModulus}^{\uparrow}$ packet records the finite route by which monotone pointwise continuity data on a compact metric source become an explicit uniform-modulus certificate. It sits beside the monotone Dini route of chapter 5154, the uniformly continuous sequence surface of chapter 4287, the uniform-modulus packet of chapter 369, the continuous-map rows of chapter 130, the compact-metric source of chapter 129, and the real-name seal of chapter 38. The packet names the constructive finite modulus handoff; it does not assert open-cover compactness, countable choice, a host supremum principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7878.1 (Dini uniform-modulus carrier)**.** *A* $\mathsf{DiniUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, S, M, U, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the finite metric observations and compact-source identifiers used by the packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-continuity row for the inspected family over the compact source.*
3. *S is the* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *sequence row, recording the displayed monotone approximation schedule before any terminal modulus is read.*
4. *M is the* $\mathsf{DiniUniformMonotone}^{\uparrow}$ *monotone-convergence row that binds pointwise convergence to the finite compact route.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row reached only after K, F, S, M have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and value-readback row used by the monotone comparison; it is terminal and does not export real completeness.*
7. *H records componentwise* hsame *transport for the compact source, continuous-map rows, sequence schedule, monotone route, uniform-modulus output, real readback, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through compact finite inspection, pointwise continuity, monotone schedule, and final uniform-modulus output.*

9. *P is the* Pkg *provenance over* $\mathsf{DiniUniformModulus}^{\uparrow}$, $\mathsf{DiniUniformMonotone}^{\uparrow}$, $\mathsf{UniformlyContinuousSequence}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected infinite subcover, countable choice, quotient real equality, a host supremum, full compactness outside the displayed finite source, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7878.2 (Dini uniform-modulus NameCert obligations)**.** *For every accepted* $\mathsf{DiniUniformModulus}^{\uparrow}$ *carrier $D = (K, F, S, M, U, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DiniUniformModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-map pointwise rows, uniformly continuous sequence schedule, monotone Dini route, terminal uniform-modulus output, real tolerance readback, componentwise transport, displayed continuation replay, provenance, and local naming. Every uniform-modulus read must pass through $K, F, S, M, U, R, H, C, P, N$, and no obligation is stored outside those finite rows.*

Proof. Project the accepted carrier. The analytic rows are precisely the compact source K, pointwise continuous-map data F, the uniformly continuous sequence schedule S, the monotone Dini route M, the uniform-modulus output U, and the real tolerance readback R.

The NameCert obligations follow that order. A tolerance request first enters the compact finite source, then the pointwise continuity row, then the sequence schedule, and only then the monotone Dini row that licenses the uniform-modulus output.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no excluded object appears in the carrier, the local certificate cannot export a selected infinite subcover, countable choice, quotient real equality, a host supremum, open-cover compactness, or ambient real completeness. □

Theorem 7878.3 (Dini monotone uniform route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiniUniformModulus}^{\uparrow}$ *carrier receives exactly the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{UniformlyContinuousSequence}^{\uparrow},\quad \mathsf{DiniUniformMonotone}^{\uparrow},\quad \mathsf{UniformModulus}$$

through the displayed rows $K, F, S, M, U, R, H, C, P, N$. A consumer cannot bypass the monotone schedule, replace it by a host supremum argument, or import open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the compact source and pointwise-continuity rows. They fix the finite metric observations and the continuous-map requests that the later sequence and modulus rows may read.

Next use the uniformly continuous sequence schedule and the $\mathsf{DiniUniformMonotone}^{\uparrow}$ row. These rows bind monotone pointwise convergence to the displayed compact route before the terminal $\mathsf{UniformModulus}^{\uparrow}$ row is visible.

Finally inspect the real readback and structural rows. The $\mathsf{Real}^{\uparrow}$ row supplies only the tolerance read needed by the displayed route, while H, C, P, N preserve the same route under transport, replay, provenance, and local naming. The absent host supremum, selected subcover, quotient equality, choice, and ambient-completeness objects therefore cannot be recovered from the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{DiniUniformModulus}^{\uparrow}$ consumer that reaches a $\mathsf{UniformModulus}^{\uparrow}$ output without displaying the compact source row K, continuous-map row F, sequence schedule S, monotone Dini row M, and real tolerance row R refutes the carrier surface.

Independence witness: dini-uniform-monotone, uniformly-continuous-sequence, uniformmodulus, continuousmap, compactmetric, real: no listed sibling supplies the whole ten-row packet $D = (K, F, S, M, U, R, H, C, P, N)$, because each sibling contributes only monotone convergence, sequence scheduling, modulus output, pointwise continuity, compact metric source, or real readback while this chapter binds the route through transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformModulus}\uparrow}$.

Constructive story: A $\mathsf{DiniUniformModulus}^{\uparrow}$ packet is generated from a compact metric source row, ContinuousMap pointwise rows, a UniformlyContinuousSequence schedule, a DiniUniformMonotone route, a UniformModulus output, Real tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7878.1, Definition 7878.2, and Definition 7878.3 seed the finite Dini route from compact metric source rows, continuous-map pointwise data, uniformly continuous sequence schedules, monotone Dini rows, uniform-modulus output, real tolerance readback, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subcovers, countable choice, quotient real equality, host supremum arguments, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal closure requires Lean coverage for carrier admission, pointwise-continuity exposure, sequence-schedule exactness, monotone Dini handoff, uniform-modulus output, real-readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DiniUniformModulusUp.

Chapter 7879

A Concrete Naming Certificate for MetaCICResidualPairLattice$^{\uparrow}$

The MetaCICResidualPairLattice$^{\uparrow}$ packet realizes the confluence audit route of chapter 9378 as finite local-pair and candidate-join data. It sits beside the residual substitution comparison of chapter 7405, the residual substitution boundary of chapter 2181, the local join ledger of chapter 4126, and the parallel diamond residual route of chapter 5126. The packet keeps local diamond candidates finite and obstruction-visible; it is not global confluence, subject reduction, evaluator equality, quotient syntax equality, or MetaCIC kernel consistency.

Definition 7879.1 (MetaCIC residual-pair lattice carrier)**.** *A* MetaCICResidualPairLattice$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, J, B, O, T, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite residual-pair row, listing the local peak source, left residual target, right residual target, and inspected binder window.*
2. *J is the candidate-join row, recording the bounded common-target attempts for the displayed pair.*
3. *B is the bounded-checker row, recording the finite search budget and the local tests performed inside that budget.*
4. *O is the obstruction row, retaining substitution-compatibility, closed-diamond, subject-reduction, evaluator-equality, and kernel-consistency boundaries when a candidate join is not certified.*
5. *T is the transport row for local pairs, candidate joins, bounded checks, obstructions, replay, provenance, and local naming.*
6. *R is the replay row specifying how a confluence audit consumer may inspect Q, J, B, O, T before using a conditional parallel-diamond handoff.*
7. *V is the visible candidate lattice row, ordering accepted local pairs by inspected source shape, residual frontier, bounded checker outcome, and obstruction status.*
8. *H records componentwise* hsame *transport for every displayed row.*
9. *C records displayed* Cont *replay from a local diamond request through $Q, J, B, O, T, R, V, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$*, residual-substitution comparison rows, residual-substitution boundary rows, local-join ledger rows, parallel-diamond residual rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualPairLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for global confluence, subject reduction, evaluator equality, quotient syntax equality, unrestricted substitution compatibility, or kernel consistency. Future Lean `TasteGate.lean` *coverage for this packet must include both* `Nontrivial` *and* `FieldFaithful` *instances preserving $Q, J, B, O, T, R, V, H, C, P, N$ as separate fields.*

Theorem 7879.2 (MetaCIC residual-pair lattice obligations)**.** *For every accepted* $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$ *carrier*

$$L = (Q, J, B, O, T, R, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualPairLattice}^{\uparrow}}$ *surface consists exactly of local-pair admission through Q, candidate-join exposure through J, bounded-checker exposure through B, obstruction visibility through O, transport through T and H, replay through R and C, finite lattice ordering through V, provenance through P, and local naming through N. Every conditional confluence consumer must display the residual-pair row and the bounded-checker row before it cites a candidate local diamond.*

Proof. Project the carrier of Definition 7879.1. The row Q is the only source of local peak and residual-target data, while J names candidate joins for that displayed pair.

The row B records the finite budget under which the candidate is checked. If that budget does not certify the candidate, the obstruction row O remains visible instead of disappearing behind the confluence audit map.

The lattice row V orders the accepted local pairs by exactly these displayed fields. Transport and replay through T, H, R, C preserve the same rows, and P, N source and name them.

No endpoint theorem is hidden in the packet. Global confluence, subject reduction, evaluator equality, quotient syntax equality, unrestricted substitution compatibility, and kernel consistency are absent from the carrier, so the listed rows exhaust the obligation surface. □

Theorem 7879.3 (MetaCIC residual-pair lattice local-diamond budget)**.** *Every local-diamond read exported by an accepted* $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$ *packet is budgeted by the finite route*

$$Q, J, B, O, T, R, V, H, C, P, N.$$

The packet may certify a bounded local candidate join or expose the obstruction row for the same residual pair. It cannot be promoted to arbitrary closed-term confluence, subject reduction, evaluator equality, quotient syntax equality, unrestricted substitution compatibility, or MetaCIC kernel consistency.

Proof. Start with the residual-pair row Q. It fixes the local peak, the residual targets, and the inspected binder window before any candidate join is considered.

Next inspect the candidate and budget rows J and B. A certified local candidate join is public only if the bounded checker row records the displayed success for the same pair.

If the bounded checker does not certify the pair, the obstruction row O is retained. Replay and transport through T, R, V, H, C keep this success-or-obstruction distinction visible to the confluence audit consumer.

The provenance and naming rows P, N close only the finite packet. Since the excluded endpoint theorems have no row in the packet, the local-diamond read remains a bounded candidate handoff rather than a global confluence or consistency theorem. □

Falsifiable prediction: An accepted $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$ confluence-audit read is refuted if it reaches a local-diamond candidate without displaying the residual pair row, candidate-join row, bounded-checker row, obstruction row, transport, replay, provenance, and local naming rows.

Independence witness: metacic-residual-substitution-comparison, metacic-residual-substitution-boundary, metacic-local-join-ledger, parallel-diamond-residual: none is bijective to $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$ because this packet jointly carries residual local pairs, candidate joins, bounded checker outcomes, retained obstructions, finite lattice ordering, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualPairLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualPairLattice}^{\uparrow}$ packet is generated from finite residual-pair rows, candidate-join rows, bounded-checker rows, obstruction rows, a visible candidate lattice row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualPairLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7879.1, Definition 7879.2, and Definition 7879.3 seed the finite residual-pair lattice over local confluence candidates, bounded checks, retained obstructions, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualPairLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, subject reduction, evaluator equality, quotient syntax equality, unrestricted substitution compatibility, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, local-pair exposure, candidate-join exposure, bounded-checker exactness, obstruction retention, lattice ordering, transport, replay, provenance, local naming, and TasteGate nontriviality and field-faithfulness for BEDC.Derived.MetaCICResidualPairLatticeUp.

Chapter 7880

A Concrete Naming Certificate for $\mathsf{CayleyGraph}^{\uparrow}$

Following Cayley and standard algebraic graph theory, the $\mathsf{CayleyGraph}^{\uparrow}$ packet names the finite graph generated by a displayed group row and a finite generator list as BEDC NameCert data. It sits after the finite graph surface of chapter 118, the algebraic group surface of chapter 41, and the list spine of chapter 36. The packet records vertices, directed generator-labelled edges, finite walk replay, transport, provenance, and local naming; it does not import quotient graph equality, a selected group presentation, ambient connectedness, or an unbounded word problem oracle.

Definition 7880.1 (CayleyGraph carrier)**.** *A* $\mathsf{CayleyGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Gamma = (G, S, V, E, L, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, retaining the multiplication, unit, inverse, and finite support rows visible to graph construction.*
2. *S is the finite generator-list row, read through the* $\mathsf{List}^{\uparrow}$ *spine and carrying the displayed generator entries and membership tags.*
3. *V is the vertex row, naming exactly the group elements admitted from G as graph vertices.*
4. *E is the edge-incidence row, recording triples $(g, s, g \cdot s)$ for visible g in V and visible s in S.*
5. *L is the generator-label ledger, attaching each edge in E to its unique displayed generator row from S.*
6. *W is the walk and replay row, storing finite generator words, their successive vertex rows, and the displayed endpoint replay.*
7. *H records componentwise* hsame *transport for the group, generator list, vertices, edges, labels, walks, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from any graph request through G, S, V, E, L, W, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CayleyGraph}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CayleyGraph}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, S, V, E, L, W, H, C, P, N$. It has no coordinate for quotienting group elements, unlabelled host graph equality, selected presentations, Cayley-complex cells, ambient connectedness, or an unbounded word problem solver.

Theorem 7880.2 (CayleyGraph NameCert obligations)**.** *For every accepted* $\mathsf{CayleyGraph}^{\uparrow}$ *carrier*

$$\Gamma = (G, S, V, E, L, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CayleyGraph}^{\uparrow}}$ *surface consists exactly of group-row admission, finite generator-list admission, vertex exposure, edge-incidence exposure, generator-label exactness, finite walk replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every graph-facing consumer must expose G, S, V, E, L, W before the structural rows H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 7880.1. The rows G, S, V, E, L, W are the only graph-building coordinates, and the structural rows H, C, P, N preserve and name those coordinates.

The group row G supplies the multiplication, unit, inverse, and visible element rows. The finite list row S supplies the only generator entries available to edge construction.

The graph rows are then forced: vertices come from V, edge incidences from E, generator labels from L, and finite walks from W. Since no other coordinate appears in the carrier, a consumer cannot obtain quotient graph equality, a hidden presentation, or ambient connectedness from the packet. □

Theorem 7880.3 (CayleyGraph edge-label exactness)**.** *Let $\Gamma = (G, S, V, E, L, W, H, C, P, N)$ be an accepted* $\mathsf{CayleyGraph}^{\uparrow}$ *carrier. Every public edge read factors through one visible vertex row from V, one visible generator row from S, one multiplication read from G, and the edge-incidence row E before the label ledger L is consumed. Thus an accepted edge cannot be relabelled by a generator outside S or detached from the group multiplication row that produced its endpoint.*

Proof. Start with the edge-incidence row E. By Definition 7880.1, each displayed edge is a triple $(g, s, g \cdot s)$ over the vertex row V, generator list S, and multiplication row of G.

Read the label ledger L only after this incidence row is visible. The ledger attaches the edge to the same generator row s, so it cannot substitute a generator outside the finite list.

Finally apply Definition 7880.2. Transport and replay preserve the displayed incidence and label rows componentwise, so an accepted public edge retains both its group multiplication endpoint and its generator label. □

Theorem 7880.4 (CayleyGraph graph-group route)**.** *Every accepted* $\mathsf{CayleyGraph}^{\uparrow}$ *graph consumer factors through the route*

$$\mathsf{Group}^{\uparrow}, \quad \mathsf{List}^{\uparrow}, \quad \mathsf{Graph}^{\uparrow}, \quad \mathsf{BHist}, \quad \mathsf{NameCert}.$$

The $\mathsf{Graph}^{\uparrow}$ *surface sees only vertices, labelled edge incidences, and finite walk replay generated from the consumed group row and finite generator list. It does not receive quotient group equality, an unlabelled host graph, an infinite presentation, a selected normal form, or an ambient connectedness theorem.*

Proof. Project G and S from the carrier. These rows provide the algebraic source and finite generator list before any graph row is read.

Next expose V, E, L, W. This is the graph-facing surface: vertices are admitted group elements, edges are multiplication incidences by listed generators, labels retain generator identity, and walks replay finite generator words.

The route ends with H, C, P, N. These rows transport, replay, source, and name the same finite packet. Because the carrier has no row for quotient group equality, unlabelled graph equality, infinite presentations, selected normal forms, or connectedness, no accepted graph consumer can read such data from Γ. □

Falsifiable prediction: Every accepted $\mathsf{CayleyGraph}^{\uparrow}$ edge read exposes a group row, a finite generator-list row, a vertex row, an edge-incidence row, a generator-label row, transport, replay, provenance, and local naming.

Independence witness: graph, group, list: $\mathsf{CayleyGraph}^{\uparrow}$ is not bijective to any one sibling because it jointly carries group multiplication, finite generator-list membership, labelled edge incidences, finite walk replay, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CayleyGraph}^{\uparrow}}$.

Constructive story: A $\mathsf{CayleyGraph}^{\uparrow}$ packet is generated from a consumed GroupUp row, a finite ListUp generator row, vertex rows, directed generator-labelled edge-incidence rows, finite walk replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CayleyGraph}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7880.1, Definition 7880.2, Definition 7880.3, and Definition 7880.4 seed the finite Cayley graph NameCert surface over group, generator-list, graph, walk, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CayleyGraph}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient graph equality, selected group presentations, normal forms, ambient connectedness, Cayley complexes, word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator-list admission, vertex exposure, edge-incidence exactness, label exactness, finite walk replay, transport, provenance, and local naming for BEDC.Derived.CayleyGraphUp.

Chapter 7881

A Concrete Naming Certificate for $\mathsf{AlexandrovSpace}^{\uparrow}$

Following Alexandrov and Burago–Burago–Ivanov, the $\mathsf{AlexandrovSpace}^{\uparrow}$ packet records curvature-bounded metric geometry as a finite BEDC comparison-triangle surface. It sits over the metric source of chapter 57, the path-length ledger of chapter 2082, the displayed segment witness of chapter 2213, the CAT(0) comparison surface of chapter 6336, and the terminal scalar readback of chapter 38. The packet names a local comparison-triangle carrier for curvature-bounded consumers; it does not assert global Toponogov comparison, selected geodesics, quotient path equality, smooth structure, local compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7881.1 (Alexandrov space carrier)**.** *An* $\mathsf{AlexandrovSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, L, G, Z, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the distance observations and endpoint identifiers inspected by the packet.*
2. *L is the* $\mathsf{LengthSpace}^{\uparrow}$ *row, recording the finite path-length ledger that makes comparison requests length-facing.*
3. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *segment row, displaying the inspected geodesic windows used by the comparison triangle.*
4. *Z is the* $\mathsf{CATZeroMetric}^{\uparrow}$ *or curvature-comparison row, carrying the signed comparison orientation and the finite triangle inequalities read by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-readback row for comparison distances, angle surrogates, and tolerance budgets; it is terminal and exports no real-completeness principle.*
6. *T records the displayed triangle vertex, side, and comparison-side incidence data visible before any Alexandrov consumer may read the curvature bound.*
7. *H records componentwise* hsame *transport for the metric, length, geodesic, comparison, scalar, triangle, continuation, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a local comparison request through the metric source, length ledger, geodesic segment rows, comparison ledger, scalar readback, and triangle incidence rows.*

9. *P is the* Pkg *provenance over* $\mathsf{AlexandrovSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LengthSpace}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexandrovSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, G, Z, R, T, H, C, P, N$. It has no coordinate for a host path space, a selected global geodesic, quotient equality of paths, smooth tangent cones, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7881.2 (Alexandrov space NameCert obligations)**.** *For every accepted* $\mathsf{AlexandrovSpace}^{\uparrow}$ *carrier $A = (M, L, G, Z, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexandrovSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, length-space ledger, geodesic segment windows, comparison ledger, scalar readback, displayed triangle incidence, componentwise transport, displayed continuation replay, provenance, and local naming. Every curvature-bounded read must pass through $M, L, G, Z, R, T, H, C, P, N$, and no obligation is stored outside those finite rows.*

Proof. Project the accepted carrier. The analytic route is precisely the metric source M, the path-length ledger L, the geodesic segment rows G, the comparison row Z, the scalar readback R, and the triangle-incidence row T.

The structural route is then forced. Transport through H and replay through C preserve the same displayed rows, while P and N only record provenance and local naming for the accepted packet.

Since the carrier has no row for a host path space, selected global geodesic, quotient path equality, smooth tangent cone, open-cover compactness, or ambient real completeness, the local NameCert cannot export any of those objects. □

Theorem 7881.3 (Alexandrov comparison-triangle scope)**.** *Every* $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{LengthSpace}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{AlexandrovSpace}^{\uparrow}$ *carrier receives a comparison-triangle route only through*

$$\mathsf{Met}^{\uparrow}, \quad \mathsf{LengthSpace}^{\uparrow}, \quad \mathsf{GeodesicMetric}^{\uparrow}, \quad \mathsf{CATZeroMetric}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the displayed rows $M, L, G, Z, R, T, H, C, P, N$. The handoff may expose finite curvature-bound comparison data, but it cannot be read as global confluence of geodesics, a selected-geodesic operator, quotient path equality, smooth manifold structure, local compactness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the metric and length rows. They make endpoint distances and path-length requests visible before any comparison triangle can be inspected.

Next use the geodesic and comparison rows. The segment windows in G and the comparison ledger in Z bind the curvature-bounded read to displayed triangle data T before scalar readback through R is available.

Finally inspect the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite comparison route, so a downstream consumer cannot replace it by a host path object, selected geodesic, quotient equality, smooth structure, compactness theorem, or real-completeness principle. □

Falsifiable prediction: An accepted $\mathsf{AlexandrovSpace}^{\uparrow}$ consumer that exports comparison-triangle data without displaying the metric row M, length row L, geodesic row G, comparison row Z, scalar row R, and triangle-incidence row T refutes the carrier surface.

Independence witness: metric, length-space, geodesic-metric, cat-zero-metric, real: no listed sibling supplies the whole ten-row packet $A = (M, L, G, Z, R, T, H, C, P, N)$, because each sibling contributes only metric source data, path-length ledgers, segment windows, comparison inequalities, or scalar readback while this chapter binds them through triangle incidence, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandrovSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{AlexandrovSpace}^{\uparrow}$ packet is generated from a Metric source row, a LengthSpace path-length row, a GeodesicMetric segment row, a CatZeroMetric comparison row, Real scalar readback, displayed comparison-triangle incidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandrovSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7881.1, Definition 7881.2, and Definition 7881.3 seed the finite comparison-triangle route from metric, length, geodesic, comparison, scalar, triangle-incidence, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandrovSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global Toponogov comparison, selected geodesics, quotient path equality, smooth tangent cones, local compactness, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal closure requires Lean coverage for carrier admission, length-ledger exposure, geodesic-window exactness, comparison-triangle ledger scope, scalar-readback nonescape, triangle incidence, transport, replay, provenance, and local naming for BEDC.Derived.AlexandrovSpaceUp.

Chapter 7882

A Concrete Naming Certificate for CompletionDenseExtensionModulus$^{\uparrow}$

Following Bishop and Bridges constructive completion and Mac Lane universal mapping language, the CompletionDenseExtensionModulus$^{\uparrow}$ packet records the finite modulus data needed when a dense-source uniformly continuous map is read through an L10 completion route. It sits beside the dense-extension packet of chapter 6382, the universal-property surface of chapter 397, the metric dense-embedding route of chapter 3351, the finite StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readback of chapter 301, the DyadicRatCore$^{\uparrow}$ tolerance ledger of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38. The packet names only displayed dense-source, modulus, window, readback, tolerance, transport, replay, provenance, and local NameCert rows; it does not import quotient completions, selected limits, function extensionality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 7882.1 (CompletionDenseExtensionModulus carrier)**.** *A* CompletionDenseExtensionModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, U, B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the dense source row, read through the* MetricCompletionDenseEmbedding$^{\uparrow}$ *surface before a completion-facing value is exposed.*
2. *U is the uniformly continuous modulus row for the source map on the displayed dense source.*
3. *B is the completion boundary row, aligned with* CompletionDenseExtension$^{\uparrow}$ *and* CompletionUniversalProperty$^{\uparrow}$.
4. *W is the* StreamName$^{\uparrow}$ *finite-window row scheduling dense-source observations, target observations, and precision requests.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in W.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row carrying the finite comparison budget used by the modulus and readback routes.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after S, U, B, W, R, D have been displayed.*
8. *H is componentwise* hsame *transport for the source, modulus, boundary, window, readback, tolerance, seal, replay, provenance, and local naming rows.*

9. *C is displayed* Cont *replay from a dense-source precision request through modulus accounting, completion boundary exposure, finite-window readback, dyadic tolerance comparison, and terminal Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$, $\mathsf{CompletionDenseExtension}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseExtensionModulus}^{\uparrow}}$ *row.*

Accepted packets are compared only through these displayed rows and their replay paths.

Lean4 (statement) `BEDC.Derived.CompletionDenseExtensionModulusUp`

Theorem 7882.2 (CompletionDenseExtensionModulus NameCert obligations)**.** *For every accepted* $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$ *carrier* $M = (S, U, B, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseExtensionModulus}^{\uparrow}}$ *surface consists exactly of dense source exposure, uniformly continuous modulus exposure, completion boundary exposure, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every dense-extension consumer must read* S, U, B, D *before it can consume the window, readback, and seal rows* W, R, E, H, C, P, N*.*

Proof. Project the carrier of Definition 7882.1. The rows S, U, B expose the dense source, uniformly continuous modulus, and completion boundary, while D records the finite dyadic tolerance budget attached to those reads.

The finite readback rows are downstream of that analytic surface. The $\mathsf{StreamName}^{\uparrow}$ row W schedules only the displayed dense-source and target observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads those windows, and the $\mathsf{Real}^{\uparrow}$ seal E is exposed only after the dyadic tolerance budget has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row stores quotient completion equality, a selected limit, function extensionality, countable choice, or ambient Real completeness, the listed finite rows exhaust the local NameCert obligations. □

Theorem 7882.3 (CompletionDenseExtensionModulus route)**.** *Every* $\mathsf{CompletionDenseExtension}^{\uparrow}$ *or* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *consumer that reads a dense-extension modulus from an accepted* $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$ *packet factors through*

$\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{CompletionDenseExtension}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Reg

The exported route is the displayed finite modulus and tolerance surface for dense-extension reads, not a quotient completion, a selected representative, a host function equality theorem, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start from an accepted carrier $M = (S, U, B, W, R, D, E, H, C, P, N)$. By Definition 7882.2, the consumer first reads the dense source row S, the modulus row U, the completion boundary row B, and the dyadic tolerance row D.

The sibling route then fixes the public dependency surface. The dense source is interpreted through chapter 3351; the boundary and universal handoff are interpreted through chapter 6382 and chapter 397; and the finite observations pass through chapter 126, chapter 301, chapter 291, and chapter 38.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Since the carrier has no row for quotient completions, selected limits, function

extensionality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every accepted read remains inside the displayed finite modulus surface. □

Falsifiable prediction: Every accepted $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$ reader that claims a dense-extension modulus while omitting dense source exposure, uniformly continuous modulus data, completion boundary data, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Real sealing, hsame transport, Cont replay, Pkg provenance, or local NameCert data refutes the packet.

Independence witness: completion-dense-extension, completionuniversalproperty, metric-completion-dense-embedding, streamname, regseqrat, dyadicratcore, real: $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dense source, modulus, completion boundary, finite windows, readback, tolerance, Real seal, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionDenseExtensionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionDenseExtensionModulus}^{\uparrow}$ packet is generated from a dense source row, uniformly continuous modulus row, completion boundary row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionDenseExtensionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7882.1, Definition 7882.2, and Definition 7882.3 seed the dense-extension modulus route for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionDenseExtensionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, function extensionality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense source exposure, modulus exactness, completion boundary exposure, finite readback, dyadic tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionDenseExtensionModulusUp.

Chapter 7883

A Concrete Naming Certificate for $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$

Following Bourbaki and Rudin, the $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ packet records the standard analysis object of a real-valued continuous function whose nonzero region is controlled by a displayed compact row. It sits over the compact carrier of chapter 60, the continuous-map surface of chapter 130, the topology carrier of chapter 88, the metric-space source of chapter 4230, and the terminal real-value seal of chapter 38. The packet is finite NameCert data for compactly supported real observables; it is not an ambient function space, quotient equality of maps, a selected norm-attaining point, a measure theorem, or a bridge theorem.

Definition 7883.1 (Compact-support continuous-function carrier)**.** *A* $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, T, M, K, U, f, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source carrier row, read through the displayed* $\mathsf{Topology}^{\uparrow}$ *surface.*
2. *T is the topology row controlling open-neighbourhood and closed-set reads on X.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row used by metric consumers of the same source carrier.*
4. *K is the compact support row, read through* $\mathsf{Compact}^{\uparrow}$ *and naming the finite compact witness that contains every public nonzero read of the function.*
5. *U is the support-membership ledger: it records that all value reads outside the displayed compact row are forced through the zero-value branch.*
6. *f is the* $\mathsf{ContinuousMap}^{\uparrow}(X, \mathsf{Real}^{\uparrow})$ *row for the real-valued continuous map.*
7. *Z is the zero-value and exterior-vanishing row, sealing the outside-support value as a* $\mathsf{Real}^{\uparrow}$ *zero read.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for value, zero, and comparison endpoints read by f and Z.*
9. *H records componentwise* hsame *transport for source, topology, metric, compact support, support-membership, continuous-map, zero-value, scalar-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a compact-support continuous-function request through* $X, T, M, K, U, f, Z, S, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactSupportContinuousFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient function space, quotient equality of maps, selected representatives, support closure taken in a host topology, integration, measure representation, or checked standard bridge.

Theorem 7883.2 (Compact-support continuous-function NameCert obligations)**.** *For every accepted* $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ *carrier*

$$F = (X, T, M, K, U, f, Z, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactSupportContinuousFunction}^{\uparrow}}$ *surface consists exactly of source-carrier admission, topology exposure, metric-source exposure, compact-support admission, support-membership ledger exposure, continuous real-valued map admission, zero-value and exterior-vanishing exposure,* $\mathsf{Real}^{\uparrow}$ *scalar sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape from ambient function-space or host-support objects.*

Proof. Project the carrier of Definition 7883.1. The rows X, T, M display the source, topology, and metric data that a consumer may inspect, while K displays the compact support row.

The rows U, f, Z, S provide the functional data. The continuous-map row f supplies the real-valued read, U records the support-membership ledger, Z records the exterior zero branch, and S keeps all scalar endpoints inside the $\mathsf{Real}^{\uparrow}$ surface.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed packet. Since the carrier has no coordinate for host support closure, quotient equality of maps, selected representatives, integration, measure representation, or a checked bridge, the listed rows exhaust the local NameCert obligations. □

Theorem 7883.3 (Compact-support continuous-function finite support exhaustion)**.** *Every public read of an accepted* $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Topology}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Compact}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactSupportContinuousFunction}^{\uparrow}.$$

If the read asks for a nonzero value, it must expose the compact support row K and the support-membership ledger U before reading the continuous map row f. If the read lies outside the displayed support ledger, it receives only the $\mathsf{Real}^{\uparrow}$ *zero branch Z. No public consumer receives an infinite support set, host support closure, quotient function equality, selected norm-attaining point, measure representation, or ambient completeness theorem from the packet.*

Proof. Start with the source and scalar boundaries. The topology, metric, compact, continuous-map, and real-value rows are all explicit coordinates of the carrier, so a public read cannot bypass X, T, M, K, f, S.

For a nonzero read, the support-membership ledger U must be displayed before the value row is consumed. This forces the read through the compact row K and keeps the support evidence finite.

For an exterior read, the only available value coordinate is Z, the displayed zero branch sealed by S. The transport, replay, provenance, and naming rows H, C, P, N preserve the same finite packet and introduce no additional support object.

Thus the public surface is exhausted by $X, T, M, K, U, f, Z, S, H, C, P, N$. The excluded host-side objects have no carrier coordinate and cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ packet displays topology, metric, compact-support, support-membership, continuous-map, zero-value, RealUp scalar-seal, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows before any compactly supported observable consumer reads it.

Independence witness: compact, continuousmap, topology, metricspace, and real: no listed sibling is bijective to $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ because this packet jointly carries compact support, exterior-vanishing, continuous real-valued map, scalar seal, transport, replay, provenance, and local naming as one finite compactly supported function surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactSupportContinuousFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactSupportContinuousFunction}^{\uparrow}$ packet is generated from displayed topology and metric source rows, a compact support row, a support-membership ledger, a continuous real-valued map row, an exterior zero-value row, a $\mathsf{Real}^{\uparrow}$ scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactSupportContinuousFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7883.1, Definition 7883.2, and Definition 7883.3 seed the compactly supported continuous-function surface over compact, continuous-map, topology, metric-space, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactSupportContinuousFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function spaces, quotient equality of maps, selected representatives, host support closure, integration, measure representation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, topology exposure, metric-source exposure, compact-support admission, support-membership exactness, continuous-map admission, exterior zero-branch exactness, RealUp sealing, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.CompactSupportContinuousFunctionUp.

Chapter 7884

A Concrete Naming Certificate for $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$ packet records a finite ladder of dyadic interval refinements before any Real or Completion consumer reads a limit-facing seal. It sits after the endpoint core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the compact interval boundary of chapter 5553. The packet is a displayed finite refinement surface, not an infinite branch selector, quotient stream equality, compactness proof, choice principle, or ambient real-completeness assertion.

Definition 7884.1 (Dyadic interval refinement ladder carrier)**.** *A* $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, J, d, E, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the parent interval row. Its lower and upper endpoints are exposed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and are the only outer endpoints available to the ladder.*
2. *J is the child interval row retained at the last displayed rung. Its endpoints are dyadic rows produced from the parent by the finite refinement trace.*
3. *d is the dyadic mesh depth. It records the finite number of bisection or refinement steps whose mesh scale bounds J inside I.*
4. *E is the endpoint-containment ledger. It records that the child endpoints lie inside the parent endpoints and that every intermediate endpoint is a displayed dyadic mesh point.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the parent endpoints, child endpoints, mesh depth, endpoint-containment checks, and retained-rung flag.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It repacks exactly the scheduled dyadic observations from S as regular rational data for completion-facing consumers.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal. It is terminal after the dyadic endpoint rows, mesh-depth row, endpoint-containment ledger, StreamName windows, and RegSeqRat readback.*
8. *H records componentwise* hsame *transport for the parent interval, child interval, mesh depth, endpoint-containment ledger, StreamName windows, RegSeqRat readback, Real seal, replay rows, provenance, and local naming row.*

9. *C records displayed* Cont *replay from a refinement-ladder request through $I, J, d, E, S, R, A, H, P, N$ without adding a selected infinite branch or quotient stream.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven displayed rows. The carrier has no coordinate for a selected infinite refinement path, quotient stream equality, open-cover compactness, trichotomy, countable choice, host interval equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.DyadicIntervalRefinementLadderUp.DyadicIntervalRefinementLadderTasteGate
single carrier alignment
```

Theorem 7884.2 (Dyadic interval refinement ladder NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$ *carrier*

$$L = (I, J, d, E, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}}$ *surface consists exactly of carrier admission, parent interval exposure, child interval exposure, finite dyadic mesh-depth exposure, endpoint-containment exactness, finite StreamName window exposure, RegSeqRat readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. No local NameCert obligation is stored outside those rows.*

Proof. Project the carrier of Definition 7884.1. The projection exposes one parent interval row, one retained child interval row, one mesh-depth row, one endpoint-containment ledger, one StreamName window, one regular rational readback, one Real seal, and the structural rows.

The parent and child exposure obligations are forced by I and J. The mesh-depth obligation is forced by d, while endpoint containment and intermediate dyadic mesh membership are forced by E.

The readback route is forced by S, R, A. The StreamName row fixes the finite observations opened by the ladder, the RegSeqRat row repacks only those observations, and the Real seal is terminal after that readback.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no extra coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7884.3 (Dyadic interval refinement ladder Real-seal nonescape)**.** *Let $L = (I, J, d, E, S, R, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$ *carrier. Every* $\mathsf{CompactRealInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the ladder must first read the finite route*

$$I, J, d, E \longmapsto S \longmapsto R \longmapsto A \longmapsto H, C, P, N.$$

Thus the consumer receives a parent interval, a retained child interval, a displayed mesh depth, endpoint containment, finite StreamName windows, regular rational readback, and a terminal Real seal, but receives no selected infinite branch, quotient stream equality, open-cover compactness proof, countable-choice schedule, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the interval and mesh rows. The only parent interval is I, the only retained child interval is J, and the only mesh-scale record is d.

Use E to read endpoint containment and intermediate dyadic mesh membership. This keeps the child interval inside the parent through displayed dyadic endpoint rows before any stream or real-facing consumer can inspect the ladder.

Now read S, R, A. The StreamName window fixes the finite observations, the RegSeqRat row repacks those observations as regular rational data, and the Real seal is terminal after the readback row.

The structural rows H, C, P, N only transport, replay, source, and name this route. They do not add a branch selector, quotient stream, compactness proof, choice schedule, host interval equality, or ambient completeness row, so none of those objects can escape through the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinementLadder}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinementLadder}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7884.1, Definition 7884.2, and Definition 7884.3 seed the finite dyadic refinement ladder route from parent and child intervals through mesh depth, endpoint containment, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinementLadder}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, quotient stream equality, open-cover compactness, trichotomy, countable choice, host interval equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh-depth exposure, endpoint-containment exactness, finite window exposure, RegSeqRat readback, Real-seal nonescape, provenance exhaustion, and local naming for BEDC.Derived.DyadicIntervalRefinementLadderUp.

Constructive story: A $\mathsf{DyadicIntervalRefinementLadder}^{\uparrow}$ packet is generated from a parent interval row, a retained child interval row, a dyadic mesh-depth row, endpoint containment, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7885

A Concrete Naming Certificate for $\mathsf{DifferenceQuotient}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DifferenceQuotient}^{\uparrow}$ packet records the finite secant-window surface by which a displayed real-valued map is inspected before a derivative-facing consumer reads any local-linear approximation. It sits over the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the finite observation schedule of chapter 126, and the calculus carrier of chapter 3632. The packet names a finite secant slope window; it does not assert differentiability, a selected derivative, quotient equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Falsifiable prediction: Every accepted $\mathsf{DifferenceQuotient}^{\uparrow}$ packet must expose two nearby $\mathsf{Real}^{\uparrow}$ endpoint rows, a secant numerator row, a nonzero denominator row, and a dyadic tolerance ledger before any secant slope read is public.

Independence witness: real, regseqrat, dyadicratcore, streamname, calculus: the difference-quotient carrier is not bijective to any sibling because it binds endpoint Real rows, secant numerator and denominator data, dyadic tolerance, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one finite secant-window packet.

Definition 7885.1 (Difference quotient carrier)**.** *A* $\mathsf{DifferenceQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, A, B, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *row for the base point of the secant window;*
2. *Y is the target* $\mathsf{Real}^{\uparrow}$ *row for the displayed function value read at the base point;*
3. *A is the nearby source* $\mathsf{Real}^{\uparrow}$ *endpoint row, displayed through a finite observation window;*
4. *B is the nearby target* $\mathsf{Real}^{\uparrow}$ *endpoint row, displaying the matching function value;*
5. *S is the secant numerator row, naming the finite difference between B and Y through regular rational readback;*
6. *D is the nonzero denominator row, naming the finite difference between A and X together with the visible nonzero-step boundary;*
7. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested endpoint and quotient comparisons;*

8. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the numerator, denominator, and quotient window inspected at the requested precision;*
9. *H records componentwise* hsame *transport for the endpoint, numerator, denominator, tolerance, readback, seal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from endpoint observations to numerator and denominator rows, from those rows to regular readback, and from readback to the Real-facing seal;*
11. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, *and the real, regular, dyadic, stream, and calculus sibling chapters listed above;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DifferenceQuotient}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $X, Y, A, B, S, D, R, E, H, C, P, N$. It has no coordinate for a host function-space quotient, an infinitesimal step, a completed derivative value, or a global differentiability claim.

Lean4 ✓ `BEDC.Derived.DifferenceQuotientUp.DifferenceQuotientTasteGate single carrier alignment`

Theorem 7885.2 (DifferenceQuotient NameCert obligations)**.** *For the carrier of Definition 7885.1, the local* $\mathsf{NameCert}_{\mathsf{DifferenceQuotient}^{\uparrow}}$ *surface consists exactly of carrier admission for $Q = (X, Y, A, B, S, D, R, E, H, C$ source and target endpoint exposure, secant numerator exposure, nonzero denominator exposure, dyadic tolerance accounting, regular rational readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted secant-slope read must pass through the route*

$$(X, Y, A, B) \longmapsto (S, D) \longmapsto (R, E) \longmapsto C$$

with H, P, N attached as transport, provenance, and naming rows.

Proof. Carrier admission is projection from Definition 7885.1. The displayed tuple gives exactly the base source row, base target row, nearby endpoint rows, numerator row, denominator row, dyadic tolerance ledger, regular rational readback, transport, replay, provenance, and local naming coordinate.

The analytic source is the constructive secant-window reading of Bishop–Bridges analysis. The rows X, Y, A, B expose only two finite endpoint observations; S and D package the visible numerator and denominator; R and E control the requested tolerance and rational readback before any Real-facing seal is consumed.

The sibling chapters supply the boundary discipline. The Real chapter supplies the terminal seal, the RegSeqRat chapter supplies rational readback, the DyadicRatCore chapter supplies tolerance accounting, the StreamName chapter supplies the finite observation schedule, and the Calculus chapter records how derivative-facing consumers may later read finite difference-quotient rows without importing a host derivative operator.

Since no row of Q is a completed derivative, a quotient equality of functions, an infinitesimal, or an ambient completeness principle, the NameCert obligations are exhausted by the displayed secant-window route. □

Theorem 7885.3 (DifferenceQuotient finite-window scope)**.** *For every accepted* $\mathsf{DifferenceQuotient}^{\uparrow}$ *carrier $Q = (X, Y, A, B, S, D, R, E, H, C, P, N)$, a consumer may read only the finite secant-window rows displayed in Definition 7885.1. In particular, the packet supplies a finite route from two endpoint observations to a tolerance-controlled secant quotient, but it does not supply a differentiability witness, a limit over all steps, a quotient of functions, or a completed* $\mathsf{Real}^{\uparrow}$ *derivative.*

Proof. Read Definition 7885.2. It identifies the public surface with endpoint exposure, numerator and denominator exposure, dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and naming.

Every allowed read must therefore pass through the finite route from (X, Y, A, B) to (S, D) and then through R, E, C, H, P, N. The carrier contains no additional row that ranges over all nonzero steps, selects a limiting value, or compares host functions extensionally.

Thus the packet is exactly a secant-window interface. It can be consumed by later $\mathsf{Calculus}^{\uparrow}$ derivative rows as finite difference-quotient data, while the differentiability, limit, and completeness claims remain outside this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{DifferenceQuotient}^{\uparrow}}$.

Constructive story: A $\mathsf{DifferenceQuotient}^{\uparrow}$ packet is generated from source and target $\mathsf{Real}^{\uparrow}$ endpoint rows, a secant numerator row, a nonzero denominator row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DifferenceQuotient}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7885.1, Definition 7885.2, and Definition 7885.3 define the finite secant-window carrier and its local NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DifferenceQuotient}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close differentiability, a derivative limit, quotient equality of functions, Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires binding the endpoint, numerator, denominator, tolerance, readback, seal, transport, replay, provenance, and local naming rows to a formal DifferenceQuotient carrier.

Chapter 7886

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style completion, the $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ packet records the unit row of the Cauchy-completion adjunction as finite BEDC certificate data. It sits after the metric completion carrier of chapter 1506, the dense embedding surface of chapter 3617, the $\mathsf{StreamName}^{\uparrow}$ finite-window route of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names only a source metric, unit map, dense row, finite stream windows, regular rational readback, transport, replay, provenance, and local NameCert row; it does not import quotient completion equality, selected limits, countable choice, or ambient real completeness.

Definition 7886.1 (CauchyCompletionAdjointUnit carrier)**.** *A* $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, J, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, read through the same metric source used by* $\mathsf{MetricCompletion}^{\uparrow}$ *and its dense-completion consumers.*
2. *J is the unit map row, sending displayed source points into the completion-facing readout without selecting completed representatives.*
3. *D is the dense row, recording that every accepted completion-side finite observation is reached through the displayed source image before a terminal seal is exposed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for source observations, completion observations, and requested tolerances.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S.*
6. *H records componentwise* hsame *transport for the source metric, unit map, dense row, window row, readback row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source metric reads through the unit map, dense exposure, StreamName windows, RegSeqRat readback, and the terminal* $\mathsf{Real}^{\uparrow}$*-facing seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}}$ *row.*

Accepted packets are compared only through $X, J, D, S, R, H, C, P, N$ and their displayed replay paths.

Theorem 7886.2 (CauchyCompletionAdjointUnit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ *carrier $U = (X, J, D, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}}$ *surface consists exactly of source metric exposure, unit-map exposure, dense-row exposure, StreamName finite windows, RegSeqRat readback, componentwise transport, displayed replay, provenance, and local naming. Every unit consumer must read X, J, D before it can consume the finite windows, readback, transport, replay, provenance, and naming rows.*

Proof. Project the carrier of Definition 7886.1. The rows X, J, D expose the source metric, the unit map, and the dense row that makes completion-side observations source-facing.

The finite readback surface is downstream of those rows. The $\mathsf{StreamName}^{\uparrow}$ row S schedules only displayed source and completion observations, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads those windows before any $\mathsf{Real}^{\uparrow}$-facing seal is consumed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row stores quotient equality, a selected completed representative, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7886.3 (CauchyCompletionAdjointUnit handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or completion-adjoint consumer that reads the unit of a source metric from an accepted* $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ *packet factors through*

$$\mathsf{DenseEmbedding}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The exported handoff is the displayed finite unit-row surface for source-to-completion reads, not a quotient completion, selected representative, host equality theorem, countable-choice schedule, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start from an accepted carrier $U = (X, J, D, S, R, H, C, P, N)$. By Definition 7886.2, the consumer first reads the source metric row X, unit map row J, and dense row D.

The sibling route then fixes the public dependency surface. The unit read is interpreted through chapter 1506 and chapter 3617; the finite observations pass through chapter 126; the regular rational readback is supplied by chapter 301; and the terminal real-facing boundary is sealed only through chapter 38.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Since the carrier has no row for quotient completions, selected limits, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every accepted read remains inside the displayed finite unit-row surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ reader that claims a completion unit while omitting the source metric, unit map, dense row, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, or local NameCert data refutes the packet.

Independence witness: metriccompletion, dense-embedding, streamname, regseqrat, and real: $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source metric, unit map, dense row, finite windows, regular rational readback, transport, replay, provenance, and local NameCert rows in one unit packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}$ packet is generated from a source metric row, unit map row, dense row, StreamName finite windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7886.1, Definition 7886.2, and Definition 7886.3 seed the finite completion-unit route over MetricCompletionUp, DenseEmbeddingUp, StreamNameUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjointUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source metric exposure, unit-map exactness, dense-row exactness, finite readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionAdjointUnitUp.

Chapter 7887

A Concrete Naming Certificate for HyperbolicOperatorCayley$^{\uparrow}$

The HyperbolicOperatorCayley$^{\uparrow}$ packet realizes the hyperbolic phase/operator bridge as a finite Cayley disk-transport certificate. It sits beside the hyperbolic metric route of chapter 3474, the Poincare disk metric packet of chapter 6209, the reversible transport surface of chapter 219, the bounded operator-boundary discipline of chapter 213, and the spectral boundary warning of chapter 214. The packet names finite disk transport and operator-boundary rows; it is not a global spectral theorem, not an unbounded functional calculus, not a host Hilbert-space construction, and not a completed analytic model of the observer.

Definition 7887.1 (HyperbolicOperatorCayley carrier)**.** *A* HyperbolicOperatorCayley$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, M, O, Y, B, T, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-coordinate row, read through* PoincareDiskMetric$^{\uparrow}$ *before any operator boundary is exposed.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric row on the displayed disk observations.*
3. *O is the operator-window row, recording the finite operator input accepted by the packet.*
4. *Y is the Cayley-transform row, recording the displayed disk transport induced by the operator window without invoking a global spectrum.*
5. *B is the operator-boundary row, aligned with* OperatorIdeal$^{\uparrow}$ *and recording boundedness, compactness, or ideal-facing refusal data as finite rows.*
6. *T is the reversible phase-transport row, consumed through* UnitaryGroup$^{\uparrow}$*-style classifier preservation when such a reversible read is present.*
7. *F is the local refusal row excluding global spectral measures, host functional calculus, unbounded operator domains, and observer-substance readings.*
8. *H records componentwise* hsame *transport for disk, metric, operator, Cayley, boundary, reversible transport, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through disk readout, metric comparison, operator-window admission, Cayley disk transport, operator-boundary exposure, reversible transport, refusal, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{SpectralTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorCayley}^{\uparrow}}$ *row.*

The Lean-side follow-up for this carrier must provide `Nontrivial HyperbolicOperatorCayleyUp` *and* `FieldFaithful HyperbolicOperatorCayleyUp` *TasteGate instances for the same displayed packet.*

Theorem 7887.2 (HyperbolicOperatorCayley NameCert obligations)**.** *For every accepted* HyperbolicOperatorCayle *carrier $K = (D, M, O, Y, B, T, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorCayley}^{\uparrow}}$ *surface consists exactly of disk-coordinate exposure, hyperbolic metric readback, finite operator-window admission, Cayley disk-transport exposure, operator-boundary exposure, reversible phase-transport exposure, global spectral refusal, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7887.1. The rows D, M, O, Y, B, T, F expose the disk, metric, operator window, Cayley transport, operator boundary, reversible phase route, and refusal surface.

The Cayley row Y is local to the displayed disk and operator window. It may move disk observations through a finite transport row, but it does not supply a global spectral decomposition, a spectral measure, an unbounded functional calculus, or a host Hilbert-space theorem.

Finally H, C, P, N transport, replay, source, and name the same packet. Since every excluded analytic object is named only by the refusal row F and has no carrier coordinate, the listed finite rows exhaust the local NameCert obligations. □

Theorem 7887.3 (HyperbolicOperatorCayley vision handoff)**.** *Every hyperbolic phase/operator read exported by an accepted* $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$ *packet factors through*

$$\mathsf{PoincareDiskMetric}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{UnitaryGroup}^{\uparrow},\ \mathsf{OperatorIdeal}^{\uparrow}.$$

The handoff is the finite Cayley disk-transport and operator-boundary row. It cannot be strengthened into a global spectral theorem, host functional calculus, spectral-measure classifier, unbounded-domain theorem, or observer-substance model.

Proof. Start from an accepted carrier $K = (D, M, O, Y, B, T, F, H, C, P, N)$. The disk rows D, M identify the Poincare-disk and hyperbolic metric surface before any operator row is consumed.

The operator rows O, Y, B, T then expose only a finite window. The Cayley row Y records the disk transport, the boundary row B records the operator-side refusal and boundedness surface, and the reversible row T is consumed only through classifier-preserving transport.

Use the refusal row F together with transport and replay. The structural rows H, C, P, N keep the same displayed dependency route, while F blocks global spectral measures, unbounded functional calculus, host Hilbert-space carriers, and observer-substance readings. Thus the vision handoff is finite NameCert data and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$ packet exposes a finite Cayley disk-transport row and an operator-boundary row, and no accepted packet exports a global spectral theorem, spectral measure, unbounded functional calculus, host Hilbert-space construction, or observer-substance model.

Independence witness: hyperbolic-metric, poincare-disk-metric, unitarygroup, operatorideal, and spectraltheorem: none is bijective to $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$ because this packet jointly

carries disk coordinates, hyperbolic metric readback, an operator window, Cayley disk transport, operator-boundary rows, reversible transport, refusal, hsame transport, Cont replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorCayley}\uparrow}$.

Constructive story: A $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$ packet is generated from a disk-coordinate row, HyperbolicMetric row, finite operator-window row, Cayley disk-transport row, operator-boundary row, reversible phase-transport row, global spectral refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicOperatorCayley}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7887.1, Definition 7887.2, and Definition 7887.3 seed the finite Cayley operator route over PoincareDiskMetricUp, HyperbolicMetricUp, UnitaryGroupUp, OperatorIdealUp, and SpectralTheoremUp refusal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicOperatorCayley}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global spectral theorems, spectral measures, unbounded functional calculus, host Hilbert-space models, observer-substance readings, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cayley disk-transport exactness, operator-boundary exactness, reversible transport, spectral refusal, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicOperatorCayleyUp.

Chapter 7888

A Concrete Naming Certificate for $\mathsf{CauchyModulusRegularization}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusRegularization}^{\uparrow}$ packet records the finite route by which an ordinary modulus-controlled Cauchy window is read as a regular sequence interface. It sits between the Cauchy modulus surface of chapter 292, the finite observations of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes only displayed modulus, window, tolerance, readback, seal, transport, replay, provenance, and local NameCert rows; it does not choose a limit, identify quotient streams, invoke countable choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7888.1 (CauchyModulusRegularization carrier)**.** *A* $\mathsf{CauchyModulusRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row recording the ordinary Cauchy threshold request and its cofinal-window response.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from the named stream after the threshold in M has been displayed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the requested precision with the window rows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the threshold, window, and dyadic tolerance data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, W, D, R have been exposed.*
6. *H is componentwise* hsame *transport for the modulus, window, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay from a Cauchy precision request through the modulus threshold, finite-window inspection, dyadic tolerance comparison, regular rational readback, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusRegularization}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyModulusRegularization}^{\uparrow}}$ *row.*

Accepted packets are compared only through these displayed rows and the replay paths they name.

Theorem 7888.2 (CauchyModulusRegularization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusRegularization}^{\uparrow}$ *carrier* $Q = (M, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRegularization}^{\uparrow}}$ *surface consists exactly of CauchyModulus threshold exposure, StreamName finite-window exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. A completion-facing consumer must read* M, W, D *before it can consume* R, E, H, C, P, N*.*

Proof. Project the carrier of Definition 7888.1. The rows M, W, D expose the Cauchy threshold, finite observation window, and dyadic tolerance ledger before any regular readback is available.

The readback and seal rows are downstream of that finite route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is assembled only from the displayed threshold, window, and tolerance data, and the $\mathsf{Real}^{\uparrow}$ seal E is exposed only after R has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row stores a selected limit, quotient stream equality, countable-choice schedule, host real equality, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 7888.3 (CauchyModulusRegularization RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusRegularization}^{\uparrow}$ *carrier* $Q = (M, W, D, R, E, H, C, P, N)$ *must first read the finite route*

$$M, W, D, R, H, C, P, N.$$

Thus the regular rational sequence interface is exported only through the displayed Cauchy modulus row, finite StreamName windows, dyadic tolerance ledger, regular readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*.*

The handoff exports no quotient stream, selected subsequence, countable-choice schedule, trichotomy branch, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the obligation surface of Definition 7888.2. It places every accepted readback behind the displayed threshold, window, and dyadic tolerance rows.

Now read the sibling route through chapter 292, chapter 126, chapter 291, and chapter 301. These chapters supply the named rows M, W, D, R but do not add a hidden quotient stream or selected representative to the packet.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Therefore every RegSeqRat-facing consumer receives only the displayed handoff. □

Theorem 7888.4 (CauchyModulusRegularization Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{CauchyModulusRegularization}^{\uparrow}$ *carrier* $Q = (M, W, D, R, E, H, C, P, N)$ *factors through the regularization route*

$$\mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The seal row E is visible only after the threshold, finite-window, tolerance, and regular readback rows M, W, D, R have been displayed.

The boundary exports no Real completeness theorem, quotient stream equality, selected limit, countable-choice schedule, open-cover compactness row, or host equality of completed real values.

Proof. Apply the RegSeqRat handoff of Definition 7888.3. It exposes R only through M, W, D and the structural rows of the same packet.

The terminal seal is then read through chapter 38. Since the carrier contains no additional coordinate between R and E, the Real-facing read cannot bypass the displayed regularization route.

The refused endpoints are absent from the carrier inventory and from the sibling route just named. Hence the seal boundary is exactly the finite route stated above. □

Theorem 7888.5 (CauchyModulusRegularization completion consumer)**.** *Every completion-facing consumer that uses an accepted* $\mathsf{CauchyModulusRegularization}^{\uparrow}$ *packet must read the route* $M, W, D, R, E, H, C, P, N$ *through* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any completion seal is visible. The route is a finite regularization handoff from ordinary modulus-controlled Cauchy windows to regular rational sequence data, not a proof of quotient completion equality, selected limits, countable choice, or ambient completeness.*

Proof. Use the NameCert obligations of Definition 7888.2. They force every accepted packet to expose the threshold, window, tolerance, readback, seal, transport, replay, provenance, and local naming rows as one finite surface.

Next combine the RegSeqRat handoff and Real seal boundary, Definition 7888.3 and Definition 7888.4. Together they place the completion-facing read behind $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Since the packet has no row for quotient completion equality, selected limits, countable choice, or ambient completeness, a completion consumer can only receive the displayed regularization handoff. □

Falsifiable prediction: Every accepted $\mathsf{CauchyModulusRegularization}^{\uparrow}$ reader that claims a completion-facing regular sequence while omitting CauchyModulus threshold exposure, StreamName windows, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, or local NameCert data refutes the packet.

Independence witness: cauchymodulus, streamname, dyadicratcore, regseqrat, real: CauchyModulusRegularizati is not bijective to any listed sibling because it jointly carries threshold, window, tolerance, readback, seal, transport, replay, provenance, and local NameCert rows in one completion-facing handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusRegularization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusRegularization}^{\uparrow}$ packet is generated from a CauchyModulus threshold row, StreamName finite windows, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7888.1, Definition 7888.2, Definition 7888.3, Definition 7888.4, and Definition 7888.5 seed the finite regularization handoff from ordinary Cauchy modulus windows to RegSeqRat and Real completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exposure, finite-window exactness, dyadic tolerance accounting, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusRegularizationUp.

Chapter 7889

Compact Open Lipschitz Modulus NameCert Construction

Following Kelley and the Arzela–Ascoli compact-open tradition, the $\mathsf{CompactOpenLipschitzModulus}^{\uparrow}$ packet records the finite BEDC surface for reading Lipschitz control through compact-open tests. It sits between the compact-open topology carrier of chapter 4138, the extension boundary of chapter 3650, the continuous-map row of chapter 130, and the metric-space carrier of chapter 57. The packet names compact tests, displayed Lipschitz bounds, and a replayable modulus ledger without assuming open-cover compactness or a host function space.

Definition 7889.1 (Compact-open Lipschitz modulus carrier)**.** *A* $\mathsf{CompactOpenLipschitzModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, B, M, T, R, P, N)$$

with the following displayed rows:

1. *K is the compact-test row, listing the compact-open test neighborhoods read from* $\mathsf{CompactOpenTopology}^{\uparrow}$ *and the source* $\mathsf{Met}^{\uparrow}$ *rows on which they are evaluated.*
2. *B is the Lipschitz-bound row, recording the displayed constants and metric-distance comparisons used by the* $\mathsf{LipschitzExtension}^{\uparrow}$ *boundary.*
3. *M is the modulus row, assigning to each admitted compact test a finite tolerance request and its displayed response.*
4. *T is the transport row, preserving K, B, M under componentwise* hsame *for the compact tests, metric observations, continuous-map graph rows, and Lipschitz-bound rows.*
5. *R is the replay row, recording displayed* Cont *reads from compact-open requests through the bound and modulus rows.*
6. *P is the provenance row over* $\mathsf{CompactOpenLipschitzModulus}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenLipschitzModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through K, B, M, T, R, P, N. It has no coordinate for open-cover compactness, quotient equality of maps, an ambient compact-open function space, selected global moduli, or completeness of a host target space.

Theorem 7889.2 (Compact-open Lipschitz modulus obligations)**.** *For the carrier of Definition 7889.1, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenLipschitzModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for compact tests, Lipschitz bounds, modulus data, transport, replay, provenance, and local naming;*
2. *compact-test exposure, requiring every compact-open read to pass through* K *and the cited* $\mathsf{CompactOpenTopology}^{\uparrow}$ *source rows;*
3. *Lipschitz-bound exposure, requiring every extension-facing read to expose* B *before a modulus value in* M *can be consumed;*
4. *modulus replay stability, requiring* T *and* R *to preserve the same compact-test and bound rows under componentwise transport and displayed continuation;*
5. *provenance and local naming, requiring* P *and* N *to record the packet as the sole NameCert source for the compact-open Lipschitz modulus.*

No obligation is stored outside K, B, M, T, R, P, N*.*

Proof. Project the carrier of Definition 7889.1. The displayed data are exactly the compact tests, Lipschitz-bound rows, modulus rows, transport rows, replay rows, provenance, and local naming.

The compact-open side reads only through K. The Lipschitz side reads only through B, and the modulus row M is consumed after those two rows have been displayed.

Finally T, R, P, N preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for open-cover compactness, quotient map equality, host function spaces, selected global moduli, or target completeness, the listed rows exhaust the local NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenLipschitzModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactOpenLipschitzModulus}^{\uparrow}$ packet is generated from compact-open test rows, displayed Lipschitz-bound rows, a finite modulus ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenLipschitzModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7889.1 and Definition 7889.2 seed the finite compact-open Lipschitz modulus carrier and its five-row obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenLipschitzModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host compact-open function spaces, quotient function equality, selected global moduli, target completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-test exposure, Lipschitz-bound exposure, modulus replay stability, provenance, and local naming for BEDC.Derived.CompactOpenLipschitzModulusUp.

Chapter 7890

A Concrete Naming Certificate for $\mathsf{ErdosSpace}^{\uparrow}$

Following Erdős and Engelking's classical zero-dimensional topology, the $\mathsf{ErdosSpace}^{\uparrow}$ packet records the rational-coordinate model as finite BEDC schedule rows before any topology consumer reads the named subspace. It sits after the Real-name surface of chapter 38, the regular rational readback of chapter 301, the finite observation surface of chapter 126, the metric source of chapter 57, and the topology handoff of chapter 88. The packet names a displayed rational-coordinate schedule and its induced metric/topology route; it does not supply a host subspace of Hilbert space, quotient sequence equality, a selected infinite point, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7890.1 (Erdos space carrier)**.** *An* $\mathsf{ErdosSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, S, R, A, M, O, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational-coordinate schedule row. It lists the finite coordinate requests, rational entries, support flags, and denominator guards that the packet exposes.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened for the coordinate schedule, support flags, metric requests, topology basic-neighbourhood requests, and terminal Real seals.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It repacks exactly the scheduled rational observations from S as regular rational sequence data.*
4. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for coordinate values and distance budgets. It is terminal after Q, S, R and exports no independent completeness principle.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row. It records the finite paired-coordinate distance checks consumed by metric readers of the Erdos-space packet.*
6. *O is the* $\mathsf{Topology}^{\uparrow}$ *handoff row. It records the basic-neighbourhood and finite-cover membership requests that may be read after the metric route is displayed.*
7. *H records componentwise* hsame *transport for the coordinate schedule, StreamName windows, RegSeqRat readback, Real seals, Metric handoff, Topology handoff, replay rows, provenance, and local naming row.*

8. *C records displayed* Cont *replay from a local Erdos-space request through $Q, S, R, A, M, O, H, P, N$ without adding a selected infinite coordinate sequence or quotient sequence equality.*
9. *P is the* Pkg *provenance over* $\mathsf{ErdosSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ErdosSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, R, A, M, O, H, C, P, N$. It has no coordinate for a host Hilbert subspace, quotient sequence equality, selected infinite coordinate family, open-cover compactness theorem, classical separability argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7890.2 (Erdos space NameCert obligations)**.** *For every accepted* $\mathsf{ErdosSpace}^{\uparrow}$ *carrier $E = (Q, S, R, A, M, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ErdosSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the rational-coordinate schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seals, induced* $\mathsf{Met}^{\uparrow}$ *handoff, induced* $\mathsf{Topology}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every metric or topology read of the packet must pass through $Q, S, R, A, M, O, H, C, P, N$, and no obligation is stored outside those finite rows.*

Proof. Project the accepted carrier. The analytic rows are precisely the rational-coordinate schedule Q, finite window row S, regular rational readback R, Real seal A, metric handoff M, and topology handoff O.

The structural rows are then forced. Transport through H and replay through C preserve the same displayed schedule, readback, metric, and topology rows, while P and N only source and name the accepted packet.

Since the carrier has no row for a host Hilbert subspace, quotient sequence equality, selected infinite coordinate family, open-cover compactness theorem, classical separability argument, or ambient real completeness, the local NameCert cannot export any of those objects. □

Theorem 7890.3 (Erdos space rational-sequence handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *consumer of an accepted* $\mathsf{ErdosSpace}^{\uparrow}$ *packet must first read the rational-coordinate route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Topology}^{\uparrow}$$

through the displayed rows $Q, S, R, A, M, O, H, C, P, N$. The consumer receives only finite coordinate windows, regular rational readbacks, terminal Real seals, paired-coordinate metric checks, and topology basic-neighbourhood handoffs. It receives no host product metric, quotient sequence equality, selected coordinate limit, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the coordinate route in Definition 7890.1. The only coordinate data are the rational schedule row Q and the finite windows S opened for that schedule.

Next use the readback and Real rows. The row R repacks exactly the scheduled observations as regular rational data, and A seals the coordinate and distance budgets as terminal $\mathsf{Real}^{\uparrow}$ rows.

Finally read the consumer handoff. The metric row M and topology row O are downstream of Q, S, R, A, while H, C, P, N transport, replay, source, and name that same finite route. Thus a metric or topology consumer cannot replace the displayed rational-sequence handoff by a host product metric, quotient sequence equality, selected coordinate limit, open-cover compactness proof, or ambient completeness principle. □

Falsifiable prediction: An accepted $\mathsf{ErdosSpace}^{\uparrow}$ consumer that exports a $\mathsf{Met}^{\uparrow}$ or $\mathsf{Topology}^{\uparrow}$ read without first displaying the rows $Q, S, R, A, M, O, H, C, P, N$ refutes the carrier surface.

Independence witness: real, regseqrat, streamname, metric, topology: no listed sibling supplies the whole packet $E = (Q, S, R, A, M, O, H, C, P, N)$, because each sibling contributes only Real seals, regular rational readback, finite stream windows, metric distance rows, or topology handoffs while this chapter binds the rational-coordinate schedule through transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ErdosSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ErdosSpace}^{\uparrow}$ packet is generated from a rational-coordinate schedule row, finite StreamName windows, RegSeqRat readback, Real coordinate seals, induced Metric handoff, induced Topology handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ErdosSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7890.1, Definition 7890.2, and Definition 7890.3 seed the finite rational-coordinate route from schedule rows through StreamName, RegSeqRat, Real, Metric, Topology, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ErdosSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert subspaces, quotient sequence equality, selected coordinate limits, open-cover compactness, classical separability arguments, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-coordinate schedule exposure, StreamName window exactness, RegSeqRat readback, Real-seal nonescape, Metric handoff, Topology handoff, provenance exhaustion, and local naming for BEDC.Derived.ErdosSpaceUp.

Chapter 7891

A Concrete Naming Certificate for PoincareDiskGeodesicTriangle$^{\uparrow}$

The PoincareDiskGeodesicTriangle$^{\uparrow}$ packet realizes the disk-geodesic triangle branch of the hyperbolic observer-expansion reading as finite BEDC data. It sits after the disk carrier of chapter 377, the hyperbolic metric route of chapter 3474, the boundary-geodesic route of chapter 3491, the phase-transport packet of chapter 3455, and the dynamic replay surface of chapter 195. The packet names three disk points, three geodesic sides, metric and phase rows, and local provenance; it does not assert global hyperbolic completeness, selected geodesic endpoints, quotient curve equality, smooth manifold structure, or ambient field completeness.

Definition 7891.1 (Poincare disk geodesic triangle carrier)**.** *A* PoincareDiskGeodesicTriangle$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, V, G, M, B, \Phi, Y, H, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *source row, exposing exactly the disk-coordinate windows inspected by the packet.*
2. *V is the vertex row. It records three named disk points and the finite distinctness checks visible to the triangle carrier.*
3. *G is the side row. It records the three* HyperbolicGeodesicBoundary$^{\uparrow}$ *side windows joining the displayed vertices.*
4. *M is the* HyperbolicMetric$^{\uparrow}$ *row, carrying the finite side-length, comparison, and incidence reads consumed by metric-facing readers.*
5. *B is the boundary-incidence row. It records which boundary endpoints or limiting boundary guards are inspected by the side rows, without selecting a global boundary parametrization.*
6. *Φ is the* PoincareDiskPhaseTransport$^{\uparrow}$ *row, carrying the reversible phase transport used to compare the displayed sides inside the disk model.*
7. *Y is the* DynSystem$^{\uparrow}$ *replay row for the finite phase-geodesic observation route.*
8. *H records componentwise* hsame *transport for disk points, vertices, sides, metric reads, boundary guards, phase transport, dynamic replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a triangle request through $D, V, G, M, B, \Phi, Y, H, P, N$ without adding selected geodesic endpoints or quotient curve equality.*
10. *P is the* Pkg *provenance over* PoincareDiskGeodesicTriangle$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBo PoincareDiskPhaseTransport$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{PoincareDiskGeodesicTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, V, G, M, B, \Phi, Y, H, C, P, N$. It has no coordinate for a host disk model, selected complete geodesic, quotient curve equality, smooth tangent object, measure completion, global hyperbolic completion, Nontrivial instance, or FieldFaithful instance.

Theorem 7891.2 (Poincare disk geodesic triangle NameCert obligations)**.** *For every accepted* PoincareDiskGeodesicTriangle$^{\uparrow}$ *carrier $T = (D, V, G, M, B, \Phi, Y, H, C, P, N)$, the local* NameCert$_{\text{PoincareDiskGeodesicTrian}}$ *surface consists exactly of carrier admission for the disk source, three-vertex row, three geodesic-side rows, hyperbolic-metric row, boundary-incidence guards, phase-transport row, dynamic replay row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every triangle read must pass through $D, V, G, M, B, \Phi, Y, H, C, P, N$, and no obligation is stored outside those finite rows.*

Proof. Project the accepted carrier. The geometric rows are the disk source D, vertices V, geodesic sides G, metric reads M, boundary guards B, phase transport Φ, and dynamic replay Y.

The structural rows then preserve that route. Transport through H and replay through C act componentwise on the same displayed disk, side, metric, boundary, phase, and dynamic rows, while P and N source and name the packet.

Since the carrier has no row for a host disk model, selected complete geodesic, quotient curve equality, smooth tangent object, measure completion, global completion, Nontrivial instance, or FieldFaithful instance, the local NameCert cannot export any of those objects. □

Theorem 7891.3 (Poincare disk geodesic triangle phase route)**.** *Every hyperbolic-metric, phase-transport, or dynamic consumer of an accepted* PoincareDiskGeodesicTriangle$^{\uparrow}$ *packet must first read*

UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBoundary$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, DynSystem$^{\uparrow}$

through the displayed rows $D, V, G, M, B, \Phi, Y, H, C, P, N$. The consumer receives only finite disk-point, geodesic-side, hyperbolic-metric, boundary-guard, phase-transport, dynamic-replay, transport, provenance, and local naming rows. It receives no host disk metric, selected geodesic boundary parametrization, quotient curve equality, smooth geodesic flow, measure completion, global hyperbolic completion, Nontrivial instance, or FieldFaithful instance.

Proof. Start with the disk and side data of Definition 7891.1. The row D exposes disk coordinates, V names the three inspected vertices, and G records the three side windows.

Next read the metric and boundary guards. The row M supplies only the hyperbolic-metric reads attached to those displayed sides, and B records the finite boundary-incidence checks without selecting a boundary parametrization.

Finally apply the phase and dynamic rows. The phase-transport row Φ and dynamic replay row Y are transported by H, replayed by C, sourced by P, and named by N. Because no row contains a host disk metric, quotient curve equality, selected complete geodesic, smooth flow, measure completion, global completion, Nontrivial instance, or FieldFaithful instance, none can be read by the consumer. □

Falsifiable prediction: An accepted $\mathsf{PoincareDiskGeodesicTriangle}^{\uparrow}$ consumer that reaches a hyperbolic-metric, phase-transport, or dynamic read without first displaying $D, V, G, M, B, \Phi, Y, H, C, P, N$ refutes the carrier surface.

Independence witness: unitdisk, hyperbolic-metric, hyperbolic-geodesic-boundary, poincare-disk-phase-transport, dynsystem: no listed sibling supplies the whole packet $T = (D, V, G, M, B, \Phi, Y, H, C, P, N)$, because each sibling contributes only disk coordinates, metric reads, boundary-geodesic guards, phase transport, or dynamic replay while this chapter binds the finite geodesic triangle through local transport, provenance, and naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskGeodesicTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareDiskGeodesicTriangle}^{\uparrow}$ packet is generated from a UnitDisk source row, a three-vertex row, geodesic-side rows, a HyperbolicMetric row, boundary-incidence guards, PoincareDiskPhaseTransport, DynSystem replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskGeodesicTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7891.1, Definition 7891.2, and Definition 7891.3 seed the finite disk-geodesic triangle route from disk points through geodesic sides, hyperbolic metric, boundary guards, phase transport, dynamic replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskGeodesicTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host disk metrics, selected geodesic boundary parametrizations, quotient curve equality, smooth geodesic flow, measure completion, global hyperbolic completion, Nontrivial instances, FieldFaithful instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-point exposure, geodesic-side exactness, hyperbolic-metric handoff, boundary-guard scope, phase-transport compatibility, dynamic replay, Nontrivial and FieldFaithful instance supply, provenance exhaustion, and local naming for BEDC.Derived.PoincareDiskGeodesicTriangleUp.

Chapter 7892

A Concrete Naming Certificate for $\mathsf{PolishZeroDimensional}^{\uparrow}$

Following Kechris and Engelking, the $\mathsf{PolishZeroDimensional}^{\uparrow}$ packet records the finite BEDC surface for a separable completely metrizable space whose usable topology is read through a clopen basis. It sits after the Polish carrier of chapter 1490, the Baire prefix surface of chapter 334, the Cantor coding surface of chapter 1530, the topological neighbourhood route of chapter 88, the metric row of chapter 57, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. The packet names zero-dimensionality as finite clopen-basis data; it is not a host powerset, an arbitrary Boolean algebra of subsets, a choice of all open covers, or an ambient descriptive-set-theoretic universe.

Definition 7892.1 (Polish zero-dimensional carrier)**.** *A* $\mathsf{PolishZeroDimensional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (P, T, M, B, C, W, R, H, E, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row, exposing the separable complete metric name that the packet refines.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row read from the source topology.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *observation row used to compare finite neighbourhood requests.*
4. *B is the finite clopen-basis ledger. Each admitted basic neighbourhood request names a clopen row below the requested open row, and finite intersections are recorded by displayed ledger entries.*
5. *C is the Cantor/Baire coding boundary, tying basic clopen rows to* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{CantorSet}^{\uparrow}$ *cylinder reads without selecting infinite branches.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a consumer probes the basis by names.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite metric and cylinder windows.*
8. *H records componentwise* hsame *transport for the Polish source, topology row, metric row, clopen ledger, coding boundary, window row, readback row, replay, provenance, and local name.*
9. *E records displayed* Cont *replay of the route*

$$P, T, M \longmapsto B \longmapsto C \longmapsto W, R.$$

10. *Q is the* Pkg *provenance over* PolishZeroDimensional$^\uparrow$, PolishSpace$^\uparrow$, Topology$^\uparrow$, Met$^\uparrow$, BaireSpace$^\uparrow$, CantorSet$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{PolishZeroDimensional}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows, the finite clopen refinement ledger, and their displayed transport and replay. It has no coordinate for host powersets, arbitrary subset equality, global choice of bases, open-cover compactness, selected infinite branches, quotient stream equality, or ambient classical descriptive set theory.

Theorem 7892.2 (Polish zero-dimensional NameCert obligations)**.** *For every accepted* PolishZeroDimensional$^\uparrow$ *carrier $Z = (P, T, M, B, C, W, R, H, E, Q, N)$, the local* NameCert$_{\text{PolishZeroDimensional}^\uparrow}$ *surface consists exactly of source admission for the Polish row, topology row, metric observations, finite clopen-basis ledger, Baire/Cantor cylinder boundary, finite stream windows, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses host powersets, arbitrary subset equality, global choice of bases, open-cover compactness, selected infinite branches, quotient stream equality, and ambient classical descriptive set theory.*

Proof. Project the carrier of Definition 7892.1. Its rows expose exactly the Polish source, topology and metric observations, clopen-basis ledger, Baire/Cantor boundary, finite windows, rational readback, transport, replay, provenance, and local name.

The zero-dimensional obligation is the ledger row B. For each admitted neighbourhood request, B names a displayed clopen refinement and records the finite intersection rows needed by downstream basis consumers. The route reads P, T, M before B, so the clopen row is tied to the displayed Polish topology rather than to a host subset universe.

The coding boundary C and the observation rows W, R keep later Baire or Cantor consumers inside finite cylinder and rational-readback evidence. Transport H and replay E repeat only those rows, while Q and N source and name the same packet.

Since the carrier contains no row for host powersets, arbitrary subset equality, global basis choice, open-cover compactness, selected infinite branches, quotient stream equality, or ambient classical descriptive set theory, the listed rows exhaust the local NameCert surface. □

Theorem 7892.3 (Polish zero-dimensional clopen-basis route)**.** *Every topology-facing or coding-facing consumer of an accepted* PolishZeroDimensional$^\uparrow$ *carrier $Z = (P, T, M, B, C, W, R, H, E, Q, N)$ must read the* PolishSpace$^\uparrow$ *source row,* Topology$^\uparrow$ *neighbourhood row,* Met$^\uparrow$ *observation row, finite clopen-basis ledger, Baire/Cantor cylinder boundary,* StreamName$^\uparrow$ *windows, and* RegSeqRat$^\uparrow$ *readback before using the zero-dimensional seal. The route is a finite clopen-basis NameCert route and does not export host powersets, arbitrary subset equality, global choice of bases, open-cover compactness, selected infinite branches, quotient stream equality, or ambient classical descriptive set theory.*

Proof. Start from Definition 7892.2. It orders the source rows P, T, M before the clopen-basis ledger B and then routes coding-facing reads through C, W, R.

The consumer therefore obtains a clopen basis only as displayed finite ledger data. The Baire and Cantor rows are read as cylinder boundaries attached to the same ledger, and the stream and rational rows expose only the finite observations used by that read.

Finally, componentwise transport, replay, provenance, and local naming preserve the same packet. Because no listed carrier row contains a host powerset, arbitrary subset equality, global basis choice, open-cover compactness, selected infinite branch, quotient stream equality, or ambient

descriptive-set-theoretic universe, the zero-dimensional seal cannot be replaced by any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishZeroDimensional}\uparrow}$.

Constructive story: A PolishZeroDimensional$^{\uparrow}$ packet is generated from a Polish source row, topology and metric observations, a finite clopen-basis ledger, Baire/Cantor cylinder boundary, finite StreamName windows, RegSeqRat readback, transport, replay, provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishZeroDimensional}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7892.1, Definition 7892.2, and Definition 7892.3 seed the finite clopen-basis packet over PolishSpace, Topology, Metric, BaireSpace, CantorSet, StreamName, RegSeqRat, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishZeroDimensional}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, arbitrary subset equality, global choice of bases, open-cover compactness, selected infinite branches, quotient stream equality, ambient descriptive set theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, clopen-basis refinement, finite-intersection stability, Baire/Cantor cylinder routing, finite-window readback, transport, replay, provenance, and local naming for BEDC.Derived.PolishZeroDimensionalUp.

Chapter 7893

A Concrete Naming Certificate for $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$

The $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ packet realizes the residual-diamond readback branch of chapter 9378 as finite candidate-mediated NameCert data. It sits beside the critical-path carrier of chapter 765, the parallel-diamond frontier of chapter 4238, the residual-pair lattice of chapter 7879, the residual-substitution compatibility packet of chapter 4370, and the local diamond witness of chapter 3595. The packet names the finite route by which a candidate-mediated residual diamond can be read back to normalization-facing or confluence-facing consumers without becoming Church–Rosser, subject reduction, quotient syntax equality, selected limits, Real completeness, or MetaCIC kernel consistency.

Definition 7893.1 (MetaCIC residual-diamond readback carrier)**.** *A* $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, S, B, J, R, L, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source packet whose candidate-mediated SN frontier and L10 source ledger are being read.*
2. *F is the candidate frontier row, displaying Candidate, ClosedCandidate, or CandidateSet evidence before any normalization-facing readback.*
3. *S is the residual substitution budget row, recording the finite substitution and residual windows consumed by the diamond request.*
4. *B is the bounded checker row for the two residual branches and their candidate common target.*
5. *J is the local diamond witness row, displaying the bounded join or the retained local obstruction for the inspected peak.*
6. *R is the readback row, specifying how the candidate frontier, residual budget, bounded checker, and local witness are read by confluence-facing or normalization-facing consumers.*
7. *L is the L10 four-object source ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.
8. *O is the obstruction row for subject reduction, global Church–Rosser, host confluence equality, quotient syntax equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and kernel consistency.*

9. *H records componentwise* hsame *transport for the critical-path packet, candidate frontier, residual budget, checker, local witness, readback row, L10 ledger, obstruction row, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay of the route*

$$K, F, S, B, J, L \longmapsto R \longmapsto O.$$

11. *P is the* Pkg *provenance over* MetacicResidualDiamondReadback$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICParallelDiamond MetaCICResidualPairLattice$^{\uparrow}$, MetaCICResidualSubstitutionCompatibility$^{\uparrow}$, MetaCICLocalDiamondWitness$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{MetacicResidualDiamondReadback}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for evidence-free strong normalization, global Church–Rosser, subject reduction, host confluence equality, quotient syntax equality, selected limits, ambient Real$^{\uparrow}$ *completeness, or MetaCIC kernel consistency.*

Theorem 7893.2 (MetaCIC residual-diamond readback obligations)**.** *For every accepted* MetacicResidualDiamondR *carrier $D = (K, F, S, B, J, R, L, O, H, C, P, N)$, the local* NameCert$_{\text{MetacicResidualDiamondReadback}^{\uparrow}}$ *surface consists exactly of critical-path source admission through K, candidate frontier exposure through F, residual substitution budget exposure through S, bounded-checker exposure through B, local diamond witness exposure through J, readback order through R, L10 source-ledger exposure through L, obstruction retention through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7893.1. The source row K supplies only the critical-path packet, while F exposes the candidate-mediated frontier that a normalization-facing consumer must read.

The residual-diamond data are the rows S, B, J. The residual substitution budget fixes the inspected windows, the bounded checker records the finite branch comparison, and the local diamond witness either displays the bounded join or keeps the local obstruction visible.

The readback row R can be used only after those rows and the L10 ledger L have been displayed. Transport H and replay C preserve the same order, while P and N source and name the packet.

Since O is the only row mentioning the excluded endpoints, and it records them as unavailable boundaries, the listed rows exhaust the NameCert surface. No row supplies evidence-free strong normalization, global Church–Rosser, subject reduction, host confluence equality, quotient syntax equality, selected limits, ambient Real$^{\uparrow}$ completeness, or MetaCIC kernel consistency. □

Theorem 7893.3 (MetaCIC residual-diamond candidate readback route)**.** *Every confluence-facing or normalization-facing read of an accepted* MetacicResidualDiamondReadback$^{\uparrow}$ *carrier $D = (K, F, S, B, J, R, L, O, H, C, P, N)$ must expose the candidate frontier F, residual substitution budget S, bounded checker row B, local diamond witness J, and L10 source ledger L before consuming the readback row R. The resulting route is candidate-mediated residual-diamond readback, not subject reduction, global Church–Rosser, host confluence equality, quotient syntax equality, selected limits, ambient* Real$^{\uparrow}$ *completeness, or MetaCIC kernel consistency.*

Proof. Start with the obligation surface of Definition 7893.2. It places F, S, B, J, L before any readback consumer can use R.

The candidate row F prevents the read from becoming evidence-free strong normalization. The residual budget and checker rows S, B keep the read inside a finite branch comparison, while J records the local join or the local obstruction for that same peak.

The ledger row L keeps the critical-path read attached to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source data before any completion-facing or normalization-facing consumer can use it. The structural rows H, C, P, N preserve only this displayed route.

Thus the readback is exactly the finite candidate-mediated route through the displayed rows. The excluded endpoints remain in the obstruction row O and are not exported as theorems. □

Falsifiable prediction: A purported $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ consumer is refuted if it reaches confluence-facing or normalization-facing readback without displaying the candidate frontier, residual substitution budget, bounded checker, local diamond witness, L10 four-object source ledger, transport, replay, provenance, and local naming rows.

Independence witness: metaciccriticalpath, metacic-parallel-diamond-frontier, metacic-residual-pair-lattice, metacic-residual-substitution-compatibility, metacic-local-diamond-witness: none is bijective to $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ because this packet jointly carries candidate frontier exposure, residual substitution budget, bounded branch checking, local diamond witness readback, L10 source ledger, obstruction retention, transport, replay, provenance, and local NameCert rows.

The Lean TasteGate follow-up for $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ instances. These instances must preserve the rows $K, F, S, B, J, R, L, O, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondReadback}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$ packet is generated from a MetaCIC critical-path source, candidate frontier, residual substitution budget, bounded checker row, local diamond witness, readback row, L10 four-object source ledger, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondReadback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7893.1, Definition 7893.2, and Definition 7893.3 seed the finite candidate-mediated residual-diamond readback packet over critical-path, residual budget, bounded checker, local witness, L10 ledger, obstruction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicResidualDiamondReadback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, global Church–Rosser, host confluence equality, quotient syntax equality, selected limits, ambient Real completeness, evidence-free strong normalization, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, candidate-frontier exposure, residual-budget exposure, bounded-checker exactness, local-witness exposure, readback order, L10 ledger routing, obstruction retention, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetacicResidualDiamondRead

Chapter 7894

A Concrete Naming Certificate for $\mathsf{CompactApartnessModulus}^{\uparrow}$

Bishop–Bridges-style constructive compactness reads compact information through finite approximating data rather than through an arbitrary subcover principle. The $\mathsf{CompactApartnessModulus}^{\uparrow}$ packet names the modulus row that tells an apartness-sensitive consumer how fine a finite net must be before a positive separation request is stable. It sits between the compact finite profile of chapter 5850 and the located separation route of chapter 7647, while retaining the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback rows needed by numerical consumers.

Definition 7894.1 (Compact apartness modulus carrier)**.** *A* $\mathsf{CompactApartnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, T, A, M, D, R, E, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compact reads are exposed by finite windows or finite-net data.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row naming the selected mesh scale, centers, and coverage ledger.*
3. *A is the apartness request row, recording the positive separation test that the modulus is asked to preserve.*
4. *M is the modulus row: it sends an apartness request and a compact finite-net scale to a displayed dyadic tolerance.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing the mesh and lower-bound comparisons.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational modulus values generated by M, D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the numerical separation bound read from R.*
8. *S is the separated or located-separation handoff row consumed by* $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$.
9. *H records componentwise* hsame *transport for compact rows, finite-net rows, apartness requests, modulus rows, dyadic ledgers, readback rows, Real seals, separation handoff rows, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a consumer request through $K, T, A, M, D, R, E, S, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactApartnessLocatedSeparation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no arbitrary open-cover compactness principle, selected subcover, apartness trichotomy, quotient completion equality, selected limit point, host equality, or ambient Real completeness assertion.

Theorem 7894.2 (Compact apartness modulus NameCert obligations)**.** *For every accepted* CompactApartnessMod *carrier*

$$U = (K, T, A, M, D, R, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net exposure, apartness-request exposure, modulus-to-tolerance exposure, dyadic ledger readout,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, separated handoff, componentwise transport, displayed replay, provenance, and local naming. A consumer may use the modulus only after K, T, A, M, D, R, E, S have been displayed in the same finite packet.*

Proof. Project the carrier of Definition 7894.1. The rows $K, T, A, M, D, R, E, S, H, C, P, N$ are the complete public inventory of the packet.

The compact and apartness rows are read first. The finite-net row T supplies the mesh data for the compact source K, and the apartness row A names the positive separation request that the modulus must control.

The numerical route is then forced through M, D, R, E. The modulus row produces only a displayed dyadic tolerance, the dyadic ledger checks it, the regular rational row reads it back, and the Real row seals the terminal numerical bound.

Finally S, H, C, P, N hand the same finite route to separated consumers, transport it componentwise, replay it, record provenance, and name it locally. Since no row stores an arbitrary subcover, trichotomy, quotient equality, selected limit point, host equality, or ambient completeness theorem, those objects cannot become NameCert obligations. □

Theorem 7894.3 (Compact apartness modulus finite-net separation)**.** *Every accepted* $\mathsf{CompactApartnessModulus}^{\uparrow}$ *finite-net separation read factors through*

$$K, T, A, M, D, R, E, S, H, C, P, N.$$

The row T supplies the finite net, A supplies the apartness request, M supplies the modulus-to-tolerance map, D checks the dyadic tolerance, R and E provide the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *numerical readback, and S is the only separation-facing handoff. Thus the modulus row is a finite compact-apartness coordinate, not an open-cover compactness theorem, an apartness trichotomy, quotient completion equality, selected limit data, host equality, countable choice, or ambient Real completeness.*

Proof. Start with the finite-net row T over the compact source K. A separation-facing consumer must first read the displayed centers, mesh scale, and coverage ledger rather than an arbitrary cover.

Read the apartness request through A and the modulus through M. The modulus is only accepted as a row that turns the displayed apartness request and finite-net scale into a dyadic tolerance governed by D.

The numerical value remains internal to the finite route until R and E read it back as a regular rational name and terminal Real seal. Only after those rows are visible may S pass the result to the located separation surface named in chapter 7647.

The exclusions are coordinate exhaustion. The carrier has no row for open-cover compactness, trichotomy, quotient completion equality, selected limits, host equality, choice schedules, or ambient Real completeness, so the finite-net separation read cannot export them. □

Falsifiable prediction: A compact-apartness consumer that claims a numerical separation modulus without displaying finite-net data, apartness request data, dyadic tolerance, RegSeqRat readback, Real sealing, and the separated handoff row is not a $\mathsf{CompactApartnessModulus}^{\uparrow}$ packet.

Independence witness: compactapartnessprofile, compactapartnesslocatedseparation, totallybounded, dyadicratcore, regseqrat, real: the packet is not bijective to these siblings because it singles out the modulus-to-tolerance row M and binds it to compact finite-net data, apartness requests, dyadic checking, numerical readback, Real sealing, separation handoff, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactApartnessModulus}^{\uparrow}$ packet is generated from compact finite-net data, apartness request rows, a modulus-to-tolerance row, dyadic tolerance ledgers, RegSeqRat readback, Real sealing, separated handoff data, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7894.1, Definition 7894.2, and Definition 7894.3 seed the finite compact-apartness modulus carrier and its NameCert obligation surface over displayed compact, finite-net, apartness, dyadic, RegSeqRat, Real, and separated handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, open-cover compactness, selected subcover existence, apartness trichotomy, quotient completion equality, selected limit data, host equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness-request exposure, modulus-to-tolerance exposure, dyadic tolerance readout, RegSeqRat readback, Real sealing, separated handoff, transport, replay, provenance, and local naming for the same packet.

Chapter 7895

A Concrete Naming Certificate for $\mathsf{LelekFan}^{\uparrow}$

Following Lelek on endpoint-dense fans, the $\mathsf{LelekFan}^{\uparrow}$ packet records the finite BEDC surface of the classical smooth-fan continuum whose endpoints are dense. It sits beside the pseudo-arc reconstruction of chapter 6905, the compact-metric source of chapter 129, the path-connected carrier of chapter 6885, the Peano-continuum route of chapter 2160, and the observable-open classifier of chapter 1860. The packet names finite branch windows, endpoint-density rows, compact metric support, and local fan-root coherence before any compact-continuum neighbour may read it.

Definition 7895.1 (Lelek fan carrier)**.** *A* $\mathsf{LelekFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, P, A, B, E, D, H, C, G, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite metric covers, diameter bounds, and compact-metric provenance for the current fan window.*
2. *P is the* $\mathsf{PathConnectedSpace}^{\uparrow}$ *row, recording finite arc-chain data from the root window to each displayed branch point.*
3. *A is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, supplying the compact, connected, locally connected continuum surface inspected by branch consumers.*
4. *B is the branch ledger, listing the root row, finite branch-prefix rows, attachment rows, and branch-order compatibility data.*
5. *E is the endpoint row, marking branch endpoints as finite named boundary records rather than as a completed endpoint set.*
6. *D is the endpoint-density row, recording for each inspected basic branch neighbourhood a finite endpoint witness whose carrier route remains inside that neighbourhood.*
7. *H records componentwise* hsame *transport for compact metric, path-connected, Peano-continuum, branch, endpoint, density, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from finite fan-window requests through $K, P, A, B, E, D, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{LelekFan}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PathConnectedSpace}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{PseudoArc}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LelekFan}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these ten rows form one finite endpoint-dense fan packet. It has no coordinate for a host continuum, a completed endpoint set, quotient equality of continua, open-cover compactness, selected branches outside the displayed ledger, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7895.2 (Lelek fan NameCert obligations)**.** *For every accepted* $\mathsf{LelekFan}^{\uparrow}$ *carrier* $L = (K, P, A, B, E, D, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LelekFan}^{\uparrow}}$ *surface consists exactly of compact-metric admission through* K*, path-connected route admission through* P*, Peano-continuum support through* A*, branch-ledger readback through* B*, endpoint exposure through* E*, endpoint-density exposure through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* G*, and local naming through* N*.*

Proof. Project the carrier of Definition 7895.1. The compact source K, path-connected route P, and Peano-continuum support A are read before the branch rows, so every continuum-facing consumer begins from displayed finite metric covers, finite root-to-branch chains, and the local continuum support.

The rows B, E, D supply the fan-specific content. The branch ledger B records root and branch-prefix windows, the endpoint row E names the boundary records, and the density row D gives a finite endpoint witness inside each inspected branch neighbourhood.

Transport and replay keep the same packet visible. The rows H and C preserve the compact, path, continuum, branch, endpoint, and density rows under hsame and Cont, while G and N record provenance and local naming.

Since the carrier has no host continuum coordinate, completed endpoint set, quotient equality row, open-cover compactness proof, selected branch outside B, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness, the listed rows exhaust the local NameCert obligations. □

Theorem 7895.3 (Lelek fan endpoint-density handoff)**.** *Every accepted* $\mathsf{LelekFan}^{\uparrow}$ *carrier hands endpoint density to compact-continuum neighbours through the finite route*

$$K, P, A, B, E, D \longmapsto H, C, G, N.$$

In particular, a $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{PathConnectedSpace}^{\uparrow}$*,* $\mathsf{PeanoContinuum}^{\uparrow}$*, or* $\mathsf{SierpinskiSpace}^{\uparrow}$ *consumer first reads the compact fan window, the root-to-branch route, the continuum support, the branch ledger, the endpoint row, and the endpoint-density row before it may classify the packet as an endpoint-dense fan surface.*

Proof. Begin with the obligation surface of Definition 7895.2. The compact metric and continuum rows K, P, A determine the finite neighbourhood and path data available to a downstream consumer.

The endpoint-density handoff is the row D read together with B and E. For each inspected neighbourhood, B fixes the branch window, E names the endpoint candidate, and D records that the candidate lies in the displayed neighbourhood. The $\mathsf{SierpinskiSpace}^{\uparrow}$ classifier may observe the neighbourhood row only after these finite branch and endpoint rows have been exposed.

Finally H, C, G, N transport, replay, package, and name the same dependency route. Thus the handoff remains a displayed compact-continuum certificate, not a hidden host theorem about a completed endpoint set. □

Falsifiable prediction: An accepted $\mathsf{LelekFan}^{\uparrow}$ consumer that classifies endpoint density without reading the branch ledger B, endpoint row E, and density row D refutes the carrier surface.

Independence witness: pseudo-arc, compactmetric, path-connected-space, peano-continuum, sierpinski-space: none is bijective to $\mathsf{LelekFan}^{\uparrow}$ because this packet jointly carries compact metric support, root-to-branch path data, Peano-continuum support, branch windows, endpoint rows, endpoint-density witnesses, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LelekFan}\uparrow}$**.**

Constructive story: A $\mathsf{LelekFan}^{\uparrow}$ packet is generated from CompactMetricUp, PathConnectedSpaceUp, PeanoContinuumUp, branch-ledger, endpoint, endpoint-density, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LelekFan}\uparrow}$, seed)

Scope closed: Definition 7895.1, Definition 7895.2, and Definition 7895.3 seed the finite endpoint-dense fan surface over compact metric, path-connected, Peano-continuum, branch, endpoint, density, and observable-open rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LelekFan}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host Lelek fan theorem, a completed endpoint set, quotient continuum equality, open-cover compactness, selected branches outside the displayed ledger, countable choice, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric support, path and continuum support, branch-ledger readback, endpoint-density exposure, transport, replay, provenance, and local naming for BEDC.Derived.LelekFanUp.

Chapter 7896

A Concrete Naming Certificate for $\mathsf{RealArchimedeanFloor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArchimedeanFloor}^{\uparrow}$ packet records the finite Archimedean floor readback by which a displayed $\mathsf{Real}^{\uparrow}$ name yields an integer bracket only through inspected rational and dyadic rows. It sits over the terminal real seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the integer carrier of chapter 31, and the rational carrier of chapter 37. The packet names a finite floor-readback surface; it is not a host floor operator, a trichotomy principle, a quotient-real representative selector, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7896.1 (RealArchimedeanFloor carrier)**.** *A* $\mathsf{RealArchimedeanFloor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, Q, D, Z, A, L, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row whose inspected name is being read.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback attached to the same displayed Real name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the floor comparison windows.*
4. *Z is the* $\mathsf{Int}^{\uparrow}$ *candidate bracket row.*
5. *A is the* $\mathsf{Rat}^{\uparrow}$ *Archimedean comparison row relating the integer candidate to rational windows.*
6. *L records the displayed lower bound $Z \leq R$ through the rational readback route.*
7. *U records the displayed upper bound $R < Z + 1$ through the same route.*
8. *H records componentwise* hsame *transport for the Real seal, rational readback, dyadic tolerance ledger, integer bracket, rational comparison row, displayed bounds, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for the same finite floor-readback route.*
10. *P is the* Pkg *provenance over* $\mathsf{RealArchimedeanFloor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanFloor}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. No coordinate stores a host floor function, a trichotomy decision for arbitrary Real names, a quotient-real representative, a selected regular sequence, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealArchimedeanFloorUp.RealArchimedeanFloorTasteGate single carrier alignment`

Theorem 7896.2 (RealArchimedeanFloor NameCert obligations)**.** *For an accepted* $\mathsf{RealArchimedeanFloor}^{\uparrow}$ *carrier* $F = (R, Q, D, Z, A, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanFloor}^{\uparrow}}$ *surface consists exactly of carrier admission, sealed* $\mathsf{Real}^{\uparrow}$ *exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Int}^{\uparrow}$ *candidate exposure,* $\mathsf{Rat}^{\uparrow}$ *Archimedean comparison, displayed lower and upper floor bounds, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 7896.1. The only visible rows are $R, Q, D, Z, A, L, U, H, C, P, N$, so every floor-facing consumer must read the Real seal through the rational readback, dyadic tolerance ledger, integer candidate, and rational comparison rows before it can inspect the displayed bounds.

The lower and upper rows are tied to the same finite route. Row L records the lower comparison, while row U records the successor-side comparison; both use Q, D, Z, A rather than an external order oracle.

Finally read H, C, P, N. These rows transport, replay, source, and locally name exactly the displayed packet. Since the carrier has no coordinate for host floor, trichotomy, quotient-real representatives, selected representatives, countable choice, or ambient completeness, the local NameCert obligations cannot export any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanFloor}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealArchimedeanFloor}^{\uparrow}}$, seed)

Scope closed: Definition 7896.1 and Definition 7896.2 seed the finite floor-readback carrier over Real, RegSeqRat, DyadicRatCore, Int, and Rat rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealArchimedeanFloor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, host floor, quotient reals, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational readback, dyadic tolerance accounting, integer bracket exposure, Archimedean comparison, displayed lower and upper bound exactness, transport, replay, provenance, and local naming for BEDC.Derived.RealArchimedeanFloorUp.

Constructive story: A $\mathsf{RealArchimedeanFloor}^{\uparrow}$ packet is generated from a sealed Real row, RegSeqRat rational readback, DyadicRatCore tolerance ledger, Int candidate bracket, Rat Archimedean comparison row, displayed lower and upper bounds, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7897

A Concrete Naming Certificate for $\mathsf{IsoperimetricInequality}^{\uparrow}$

Following Federer–Fleming finite-perimeter geometry, the $\mathsf{IsoperimetricInequality}^{\uparrow}$ packet records the finite polygonal route by which a displayed planar region exposes area, perimeter, convex comparison, and compact containment rows before any inequality consumer may read the classical isoperimetric boundary. It sits over the scalar seal of chapter 38, the metric carrier of chapter 4230, the compact containment surface of chapter 60, the convex membership row of chapter 208, and the finite face ledger of chapter 209. The packet names only polygonal and perimeter ledgers; it does not import measure completion, selected minimizing domains, quotient sets, host Euclidean equality, or classical compactness.

Definition 7897.1 (Isoperimetric inequality carrier)**.** *An* $\mathsf{IsoperimetricInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, M, K, V, Q, A, L, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *scalar surface through which all area, perimeter, and squared-perimeter comparisons are read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row for the displayed planar distance comparisons used by edge lengths and compact containment.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *row naming the finite containment window for the inspected polygonal region.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *comparison row used for convex hull and support-line checks.*
5. *Q is the* $\mathsf{Polytope}^{\uparrow}$ *row carrying the finite vertex, edge, face, and orientation ledger.*
6. *A is the polygonal area ledger, read as a finite signed triangle sum through R.*
7. *L is the finite perimeter ledger, read as the displayed edge-length sum through M and R.*
8. *B is the isoperimetric budget row comparing the displayed inequality surface L^2 against the area ledger A through finite real comparisons.*
9. *H records componentwise* hsame *transport for the scalar, metric, compact, convex, polytope, area, perimeter, budget, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from vertex and edge rows through area and perimeter ledgers to the budget row.*

11. *P is the* Pkg *provenance over* $\mathsf{IsoperimetricInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IsoperimetricInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for completed perimeter measure, selected minimizers, quotient regions, host-set equality, open-cover compactness, or ambient Euclidean integration.

Theorem 7897.2 (Isoperimetric inequality NameCert obligations)**.** *For every accepted* $\mathsf{IsoperimetricInequality}^{\uparrow}$ *carrier* $I = (R, M, K, V, Q, A, L, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IsoperimetricInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real scalar, metric, compact, convex, polytope, polygonal area, finite perimeter, isoperimetric budget, componentwise transport, replay, provenance, and local naming rows. A certificate may replay the finite inequality route, but it cannot read measure completion, a selected minimizing region, quotient equality of domains, host Euclidean equality, open-cover compactness, or an ambient compactness theorem.*

Proof. Project the carrier of Definition 7897.1. The only analytic rows available to a consumer are the scalar row R, metric row M, compact window K, convex comparison row V, polytope ledger Q, area row A, perimeter row L, and budget row B.

Read the polygonal rows first. The polytope ledger supplies the finite vertices, oriented edges, and faces; the compact row bounds the inspected region; the convex row supplies only the displayed comparison checks; and the metric and real rows turn the same edge data into length and scalar comparisons.

Transport and replay then preserve the finite route. The rows H and C move and replay the same polygonal area, perimeter, and budget ledgers, while P and N source and name the packet. Since the forbidden objects have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7897.3 (Isoperimetric inequality finite perimeter route)**.** *Every inequality read from an accepted* $\mathsf{IsoperimetricInequality}^{\uparrow}$ *carrier factors through the finite perimeter ledger L and polygonal area ledger A before the budget row B is visible. In particular, the consumer first reads the* $\mathsf{Polytope}^{\uparrow}$ *vertex-edge-face row, the* $\mathsf{Met}^{\uparrow}$ *edge-length row, the* $\mathsf{Real}^{\uparrow}$ *scalar comparison row, the* $\mathsf{Compact}^{\uparrow}$ *containment row, and the* $\mathsf{ConvexSet}^{\uparrow}$ *comparison row; only then may it inspect the displayed isoperimetric budget. The route exports no completed perimeter measure, selected optimizer, quotient region, host plane equality, or classical minimizer principle.*

Proof. Start with the carrier. The row B is downstream of Q, A, L and therefore cannot be read until the finite polytope, area, and perimeter ledgers have been displayed.

The sibling rows supply exactly the required inputs. $\mathsf{Polytope}^{\uparrow}$ gives the finite boundary ledger, $\mathsf{Met}^{\uparrow}$ supplies the edge-length comparisons, $\mathsf{Real}^{\uparrow}$ seals the scalar area and perimeter reads, $\mathsf{Compact}^{\uparrow}$ bounds the inspected region, and $\mathsf{ConvexSet}^{\uparrow}$ records the displayed convex comparison surface.

Finally replay the route through H, C, P, N. These rows preserve and name the same finite data, so the inequality read remains a polygonal finite-perimeter route rather than a measure-completion theorem, a selected minimizer, or an ambient Euclidean principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{IsoperimetricInequality}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IsoperimetricInequality}^{\uparrow}$ packet is generated from real scalar rows, metric edge-length rows, compact containment windows, convex comparison rows, polytope vertex-edge-face ledgers, polygonal area ledgers, finite perimeter ledgers, isoperimetric budget comparisons, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{IsoperimetricInequality}\uparrow}$, seed)

Scope closed: Definition 7897.1, Definition 7897.2, and Definition 7897.3 seed the finite polygonal-perimeter certificate surface over Real, MetricSpace, Compact, ConvexSet, Polytope, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{IsoperimetricInequality}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, selected minimizing domains, quotient regions, host Euclidean equality, open-cover compactness, Lean verification, axiom-purity certification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polygonal area exactness, finite perimeter routing, convex and compact dependency transport, budget replay, provenance, and local naming for BEDC.Derived.IsoperimetricInequalityUp.

Chapter 7898

A Concrete Naming Certificate for EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$ packet records the finite bridge from local epsilon-delta data to Cauchy sequential image data. It sits between the local continuity packet of chapter 3760, the Cauchy image route of chapter 4045, the uniform sequential handoff of chapter 2261, the limit boundary of chapter 3759, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a continuous-map graph, a local radius ledger, source stream windows, image readback, transport, replay, provenance, and local naming; it does not assert host sequential continuity or identify quotient representatives.

Definition 7898.1 (Epsilon-delta sequential continuity bridge carrier)**.** *An* EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, E, S, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* ContinuousMap$^{\uparrow}$ *graph row whose local epsilon-delta reads are consumed.*
2. *E is the epsilon-delta local radius ledger, pairing each displayed target tolerance with the finite source-radius data supplied by* EpsilonDeltaContinuity$^{\uparrow}$*.*
3. *S is the source* StreamName$^{\uparrow}$ *row for the regular Cauchy input stream.*
4. *W is the finite stream-window row that compares source-window entries against the local radius ledger before any image claim is read.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for source and image windows.*
6. *R is the terminal* Real$^{\uparrow}$ *seal row reached only after the image regular-Cauchy readback is displayed.*
7. *H records componentwise* hsame *transport for the graph, radius ledger, source stream, finite windows, readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from local radius requests through source-window exactness and image readback.*

9. *P is the* Pkg *provenance over* EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$, EpsilonDeltaContinuity$^{\uparrow}$, CauchySequentialContinuity$^{\uparrow}$, UniformSequentialContinuity$^{\uparrow}$, EpsilonDeltaLimit$^{\uparrow}$, ContinuousMap$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{EpsilonDeltaSequentialContinuityBridge}^{\uparrow}}$ *row.*

The classifier accepts the packet only when the same tuple $F, E, S, W, Q, R, H, C, P, N$ supplies the local epsilon-delta read and the Cauchy sequential image handoff.

Theorem 7898.2 (Epsilon-delta sequential continuity bridge obligations)**.** *For the carrier of Definition 7898.1, the local* NameCert$_{\text{EpsilonDeltaSequentialContinuityBridge}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed packet $K = (F, E, S, W, Q, R, H, C, P, N)$;*
2. *radius-window exactness, requiring the local radius ledger E and stream-window row W to be read before any image-window claim is made;*
3. *Cauchy image handoff, requiring the* RegSeqRat$^{\uparrow}$ *readback Q to feed the* CauchySequentialContinuity$^{\uparrow}$ *preservation surface before the* Real$^{\uparrow}$ *seal R is inspected;*
4. *transport and replay through the same componentwise* hsame *row H and displayed* Cont *row C;*
5. *provenance and local naming through P and N.*

No accepted packet contains a row for host function extensionality, selected subsequences, quotient real equality, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7898.1. The graph row F, local radius ledger E, source stream S, finite windows W, readback Q, and Real seal R are explicit coordinates, so a consumer cannot begin from an ambient theorem of sequential continuity.

The bridge order is forced by the rows. The radius ledger E is read against the source windows W before image readback is available. Then Q supplies the regular-rational source and image rows consumed by the Cauchy sequential preservation surface, and R is inspected only after that handoff.

The structural rows keep the same packet throughout the read. Transport H and replay C preserve the graph, ledger, stream, windows, readback, and seal coordinates, while P and N identify the provenance and local name. Since no excluded row is present, the five listed rows exhaust the NameCert surface. □

Falsifiable prediction: Every EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$ read factors a local epsilon-delta radius ledger through finite StreamName windows and RegSeqRat image readback before the terminal Real seal is inspected.

Independence witness: epsilon-delta-continuity, cauchy-sequential-continuity, uniform-sequential-continuity, epsilon-delta-limit, streamname, regseqrat, real: EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a local radius ledger, a source stream, finite bridge windows, image readback, Real seal ordering, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{EpsilonDeltaSequentialContinuityBridge}^{\uparrow}}$.

Constructive story: An EpsilonDeltaSequentialContinuityBridge$^{\uparrow}$ packet is generated from a ContinuousMap graph row, an epsilon-delta local radius ledger, a source StreamName row, finite stream windows, RegSeqRat readback, a Real seal, hsame transport, Cont replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{EpsilonDeltaSequentialContinuityBridge}^{\uparrow}}$, seed)

Scope closed: Definition 7898.1 and Definition 7898.2 seed the finite bridge from local epsilon-delta continuity data to Cauchy sequential image data.

Formal verification: FormalStatus(NameCert$_{\mathsf{EpsilonDeltaSequentialContinuityBridge}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, selected subsequences, quotient real equality, countable choice, trichotomy, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-window exactness, Cauchy image handoff, transport, replay, provenance, and local naming for BEDC.Derived.EpsilonDeltaSequentialContinuityBridgeUp.

Chapter 7899

A Concrete Naming Certificate for $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$

Following Bridson–Haefliger and Burago–Burago–Ivanov on geodesic metric spaces and convex subsets, the $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ packet records the finite BEDC surface on which a displayed geodesic segment remains inside a displayed convex target. It sits over the metric source of chapter 57, the segment witness of chapter 2213, the convex membership carrier of chapter 208, the CAT(0) comparison packet of chapter 6336, and the terminal scalar readback of chapter 38. The packet names only endpoint, segment, convex-target, comparison, transport, replay, provenance, and local naming rows; it does not select global geodesics, quotient paths, ambient closed convex subsets, or real-completeness data.

Definition 7899.1 (Finite geodesic convexity carrier)**.** *A* $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, E, G, V, Z, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the distance observations and endpoint identifiers inspected by the packet.*
2. *E is the endpoint row, recording the two displayed endpoints and their admitted membership in the target convex carrier.*
3. *G is the finite* $\mathsf{GeodesicMetric}^{\uparrow}$ *segment-witness row joining the endpoints through named segment parameters and metric-distance ledgers.*
4. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *target row, exposing the membership and convex-combination obligations that the inspected segment must respect.*
5. *Z is the comparison ledger, optionally read through* $\mathsf{CATZeroMetric}^{\uparrow}$ *or a sibling metric-convexity packet when a downstream route needs curvature-facing convexity.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for segment parameters, distance budgets, and comparison tolerances; it is terminal and exports no ambient real-completeness principle.*
7. *H records componentwise* hsame *transport for the metric source, endpoints, geodesic witness, convex target, comparison ledger, scalar readback, continuation, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a convexity request through $M, E, G, V, Z, R, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteGeodesicConvexity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, G, V, Z, R, H, C, P, N$. It has no coordinate for a host path space, selected geodesic family, quotient path equality, global convex hull operator, projection selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7899.2 (Finite geodesic convexity NameCert obligations)**.** *For every accepted* $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ *carrier $F = (M, E, G, V, Z, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteGeodesicConvexity}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, endpoint membership, finite geodesic segment witness, convex target row, comparison ledger, terminal real readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every CAT(0), Hadamard, projection, or Alexandrov consumer must read the finite segment-inside-target route through those rows before using the packet as metric-convexity data.*

Proof. Project the carrier of Definition 7899.1. The metric and endpoint data are exactly M and E, so the packet fixes the inspected source and target endpoints before a segment can be read.

Read the geodesic and convex rows next. The row G supplies the finite segment witness, while V supplies the convex target and membership obligations. The comparison row Z and scalar row R only annotate this same finite segment route with downstream comparison and tolerance data.

Finally inspect the structural rows. Transport through H and replay through C preserve the same metric, endpoint, segment, convex-target, comparison, and scalar rows, while P and N source and name the accepted packet. Since no row stores a host path space, selected geodesic family, quotient path equality, global convex hull operator, projection selector, or ambient real-completeness principle, the displayed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ consumer that exports a segment-inside-convex-target read without displaying $M, E, G, V, Z, R, H, C, P, N$ refutes the carrier surface.

Independence witness: metric, geodesic-metric, convex-set, cat-zero-metric, real: no listed sibling supplies the whole packet $F = (M, E, G, V, Z, R, H, C, P, N)$, because the siblings contribute only distance observations, segment witnesses, convex membership rows, comparison ledgers, or scalar readback while this chapter binds them into one finite metric-convexity carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteGeodesicConvexity}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ packet is generated from a Metric source row, endpoint membership rows, a finite GeodesicMetric segment witness, a ConvexSet target row, a comparison ledger, terminal Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteGeodesicConvexity}^{\uparrow}}$, seed)

Scope closed: Definition 7899.1 and Definition 7899.2 seed the finite segment-inside-convex-target carrier over metric, geodesic, convex, comparison, scalar, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteGeodesicConvexity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected global geodesics, quotient path equality, global convex hull operators, projection existence, CAT(0) comparison theorems, Hadamard projection theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint membership, geodesic segment exactness, convex target exposure, comparison-ledger scope, scalar-readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteGeodesicConvexityUp.

Chapter 7900

A Concrete Naming Certificate for $\mathsf{ClaimAuditWitness}^{\uparrow}$

The $\mathsf{ClaimAuditWitness}^{\uparrow}$ packet realizes the witness side of the legitimate-claim audit of chapter 9456 as finite BEDC NameCert data. It sits beside the two-axis closure discipline of chapter 9201, the gap boundary of chapter 1006, the legitimate-claim ledger of chapter 5122, and the claim-audit boundary of chapter 7318. Those siblings classify claim status and refusal boundaries; this packet records the particular witness row that lets a reader audit one claim without promoting missing support into theorem content.

Definition 7900.1 (Claim audit witness carrier)**.** *A* $\mathsf{ClaimAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, O, T, F, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the claim-text row, recording the finite statement being audited.*
2. *O is the cited-object row, naming the theorem, definition, closure block, carrier, or refusal boundary that the claim cites.*
3. *T is the closure-axis token row, recording the displayed theory-closure token attached to the cited object.*
4. *F is the formal-status token row, recording the displayed verification token attached to the cited object.*
5. *R is the off-table refusal row, recording the boundary exposed when Q asks for a closure or verification status not displayed by T and F.*
6. *H records componentwise* hsame *transport for claim text, cited object, closure token, formal token, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from claim admission through cited-object lookup, two-axis token readback, refusal exposure, and permitted witness publication.*
8. *P is the* Pkg *provenance over* $\mathsf{ClaimAuditWitness}^{\uparrow}$, $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$, $\mathsf{ClaimAuditBoundary}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClaimAuditWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, O, T, F, R, H, C, P, N$. It has no coordinate for a global truth predicate, hidden marker resolution, off-table verification, or claim strength beyond the displayed theory and formal tokens.

Theorem 7900.2 (Claim audit witness NameCert obligations)**.** *For every accepted* $\mathsf{ClaimAuditWitness}^{\uparrow}$ *carrier* $W = (Q, O, T, F, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClaimAuditWitness}^{\uparrow}}$ *surface consists exactly of claim-text admission, cited-object readback, closure-axis token readback, formal-status token readback, off-table refusal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7900.1. The rows Q, O, T, F display the claim, the cited object, and the two tokens that license what may be said about the object.

If the claim asks for a status not displayed by T and F, the row R is exposed before publication. Thus a missing bridge, missing Lean theorem, or missing closure grade remains a refusal row rather than becoming theorem content.

The rows H, C, P, N preserve the witness under transport, replay, provenance, and local naming. Since the carrier has no hidden truth predicate, marker resolver, off-table verification row, or stronger claim coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7900.3 (Claim audit witness two-axis nonescape)**.** *Every accepted* $\mathsf{ClaimAuditWitness}^{\uparrow}$ *read factors a claim through the displayed route*

$$Q, O, T, F, R, H, C, P, N.$$

Consequently a consumer may publish only the closure and verification strength displayed by T and F; if stronger support is requested, the consumer must expose R as a refusal row before the claim can be read.

Proof. Start from Definition 7900.2. The cited object O is not read alone: it is paired with the theory token T and the formal token F inside the same finite packet.

The refusal row R handles off-table requests. When Q asks for public closure, bridge checking, theorem status, or object truth not displayed by the two tokens, the replay route C sends the request to R rather than to a hidden proof source.

Finally H, P, N transport, package, and name the same distinction. They cannot raise either axis, discharge a missing bridge, or resolve a marker not displayed by the packet. □

Falsifiable prediction: If a claim-audit read publishes a stronger closure or verification status than the displayed rows T and F while omitting the refusal row R, it is not a $\mathsf{ClaimAuditWitness}^{\uparrow}$ packet.

Independence witness: legitimate-claim-audit-ledger, claim-audit-boundary, gapclosureboundary: none is bijective to $\mathsf{ClaimAuditWitness}^{\uparrow}$ because the ledger records claim status, the boundary records permitted publication, and the gap boundary records refusal routes, while this packet jointly carries one claim text, one cited object, two-axis tokens, an off-table refusal row, transport, replay, provenance, and local naming.

The future TasteGate file for lean4/BEDC/Derived/ClaimAuditWitnessUp must supply explicit Nontrivial $\mathsf{ClaimAuditWitness}^{\uparrow}$ and FieldFaithful $\mathsf{ClaimAuditWitness}^{\uparrow}$ instances. The field-faithfulness witness must keep $Q, O, T, F, R, H, C, P, N$ distinct, so claim text, cited object, theory token, formal token, and refusal exposure cannot collapse.

Closure status for $\mathsf{NameCert}_{\mathsf{ClaimAuditWitness}^{\uparrow}}$**.**

Constructive story: A ClaimAuditWitness$^{\uparrow}$ packet is generated from claim text, cited object, theory-closure token, formal-status token, off-table refusal, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ClaimAuditWitness}\uparrow}$, seed)

Scope closed: Definition 7900.1, Definition 7900.2, and Definition 7900.3 seed the finite witness surface for two-axis legitimate-claim audits.

Formal verification: FormalStatus(NameCert$_{\text{ClaimAuditWitness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global truth, registry-wide claim correctness, Lean marker resolution, bridge checking, or any closure or verification status beyond the displayed packet rows.

Upgrade path: Obligation closure requires checked carrier admission, cited-object readback, two-axis token readback, refusal exposure, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.ClaimAuditWitnessUp.

Chapter 7901

A Concrete Naming Certificate for $\mathsf{PretopologicalSpace}^{\uparrow}$

Following Bourbaki and Kelley, a pretopological space is the neighbourhood-interface presentation obtained before open-set closure laws are imposed as a full topology. The $\mathsf{PretopologicalSpace}^{\uparrow}$ packet sits between the indexed open carrier of chapter 88, the pointwise neighbourhood rows of chapter 7753, and the prefix-sensitive neighbourhood read of chapter 2336. It records finite local neighbourhood, closure, interior, transport, replay, provenance, and naming rows without forcing a host powerset of opens, quotient equality of neighbourhood filters, selected limits, or ambient topological equivalence.

Definition 7901.1 (PretopologicalSpace carrier)**.** *A* $\mathsf{PretopologicalSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, N, C, I, H, R, Q, L)$$

with the following displayed rows:

1. *X is the point row, naming the displayed carrier points and the classifier used to compare point observations.*
2. *N is the neighbourhood-system row, recording the finite neighbourhood tokens around each displayed point and their pointwise containment requests.*
3. *C is the closure readback row, recording which displayed point lies in the closure of a displayed finite neighbourhood observation.*
4. *I is the interior readback row, recording which displayed point is accepted by a finite interior observation.*
5. *H records componentwise* hsame *transport for point rows, neighbourhood tokens, closure readbacks, interior readbacks, replay rows, provenance rows, and local naming rows.*
6. *R records displayed* Cont *replay from a local neighbourhood request through X, N, C, I.*
7. *Q is the* Pkg *provenance over* $\mathsf{PretopologicalSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NeighbourhoodSystem}^{\uparrow}$, $\mathsf{NeighborhoodFunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{PretopologicalSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, N, C, I, H, R, Q, L. It has no coordinate for full open-set topology laws, host powersets of opens or neighbourhoods, quotient equality of filters, selected limit points, countable choice, or ambient equality of topology structures.

Theorem 7901.2 (PretopologicalSpace NameCert obligations)**.** *For every accepted* $\mathsf{PretopologicalSpace}^{\uparrow}$ *carrier* $P = (X, N, C, I, H, R, Q, L)$*, the local* $\mathsf{NameCert}_{\mathsf{PretopologicalSpace}^{\uparrow}}$ *surface consists exactly of point-row exposure, neighbourhood-system exposure, closure readback, interior readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every topology-facing or neighbourhood-facing consumer must display*

$$X, N \longmapsto C, I, H, R, Q, L$$

before it may inspect closure, interior, or local neighbourhood data.

Proof. Project the carrier of Definition 7901.1. The accepted packet exposes exactly the rows X, N, C, I, H, R, Q, L, with no separate row for a host topology, powerset of opens, quotient filter, selected limit, or countable-choice schedule.

The local rows are read first. The point row X fixes the displayed classifier, and the neighbourhood-system row N supplies the finite pointwise neighbourhood tokens inherited from the neighbourhood presentation.

The readback rows are then confined to the same finite surface. Closure and interior observations are recorded by C and I, while H, R, Q, L transport, replay, source, and name the packet. Hence the local certificate obligations are exhausted by the displayed rows. □

Theorem 7901.3 (PretopologicalSpace neighbourhood transport)**.** *Every neighbourhood, closure, or interior read of an accepted* $\mathsf{PretopologicalSpace}^{\uparrow}$ *packet transports only through the route*

$$\mathsf{Topology}^{\uparrow}, \mathsf{NeighbourhoodSystem}^{\uparrow}, \mathsf{NeighborhoodFunction}^{\uparrow} \longmapsto \mathsf{PretopologicalSpace}^{\uparrow}.$$

The route preserves point-row identity, finite neighbourhood tokens, closure readback, interior readback, displayed hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not export full topology laws, host powersets of opens, quotient neighbourhood filters, selected limits, countable choice, or ambient topological equality.*

Proof. Start with the topology side. The indexed-open interface of chapter 88 supplies open-index membership and classifier transport, but the pretopological packet reads only the neighbourhood-facing consequences needed for N, C, I.

Read the pointwise neighbourhood surface next. The finite neighbourhood rows of chapter 7753 provide the local tokens, and the finite-observation discipline of chapter 2336 supplies the prefix-sensitive replay pattern for neighbourhood requests.

Finally apply the carrier inventory. Transport by H, replay by R, provenance by Q, and naming by L preserve the same rows and add no coordinate for full open-set topology laws, host powersets, quotient filters, selected limits, choice, or ambient topology equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{PretopologicalSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PretopologicalSpace}^{\uparrow}$ packet is generated from a point row, neighbourhood-system row, closure readback row, interior readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PretopologicalSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7901.1, Definition 7901.2, and Definition 7901.3 seed the neighbourhood-interface carrier between Topology, NeighbourhoodSystem, and NeighborhoodFunction rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PretopologicalSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full open-set topology laws, quotient equality of neighbourhood filters, selected limits, countable choice, ambient topology equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked point exposure, neighbourhood-system exposure, closure and interior readback, neighbourhood transport, replay, provenance, and local naming for BEDC.Derived.PretopologicalSpaceUp.

Chapter 7902

A Concrete Naming Certificate for $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$

Following Bishop and Bridges' constructive diagonal arguments for complete metric data, the $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$ packet names the finite extraction surface used before a diagonal-limit consumer reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic tolerance carrier of chapter 291, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the selection-free diagonal route of chapter 4080. The packet records diagonal extraction through displayed finite rows; it is not a selected representative, not countable choice, not open-cover compactness, not a quotient stream, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7902.1 (Finite net diagonal extraction carrier)**.** *A* $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite net-row ledger. Each row records the inspected net stage, its finite entries, and the diagonal index selected from displayed finite data.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger whose finite windows read the entries of G needed by the diagonal row.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row governing the error allowed for each displayed net row and diagonal comparison.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite windows and dyadic tolerance budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after G, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the net-row ledger, window ledger, tolerance budget, readback, seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal request through G, W, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows. It has no coordinate for selected representatives, countable choice, open-cover compactness, quotient streams, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7902.2 (Finite net diagonal extraction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$ *carrier*

$$F = (G, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite net rows, StreamName windows, DyadicRatCore tolerance budget, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming; classifier stability under the componentwise* hsame *rows; route exactness from* G, W, D, R *to* E*; and refusal rows excluding selected representatives, countable choice, open-cover compactness, quotient streams, host real equality, and ambient completeness.*

Proof. Project the carrier of Definition 7902.1. The projection exposes only the finite net-row ledger, finite windows, dyadic tolerance budget, regular readback, Real seal, transport, continuation, provenance, and local naming rows.

Classifier stability is componentwise. The rows in H transport G, W, D, R, E and the administrative rows C, P, N to a common finite comparison surface.

Route exactness is read from the continuation row C. A diagonal request first displays the net rows G, then the finite StreamName windows W, then the DyadicRatCore tolerance row D, then the RegSeqRat readback R, and only then the Real seal E.

The refusal rows are exhausted by the carrier inventory. Since no listed row contains selected representatives, countable choice, open-cover compactness, quotient streams, host equality, or ambient completeness, no local NameCert obligation can export those objects. □

Theorem 7902.3 (Finite net diagonal extraction window route)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$ *carrier must read the route*

$$G, W, D, R, E, H, C, P, N$$

before consuming the $\mathsf{Real}^{\uparrow}$ *seal. Thus the diagonal extraction passes through displayed finite net rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}}$ *row.*

No selected representative, countable-choice schedule, open-cover compactness row, quotient stream, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle may replace this finite route.*

Proof. Start with the carrier projection. The terminal $\mathsf{Real}^{\uparrow}$ seal is the row E, and it appears only after G, W, D, R in the displayed packet.

Replay the continuation row. Its route takes a diagonal request through the finite net-row ledger, then through the StreamName windows, the dyadic tolerance budget, and the RegSeqRat readback before the seal row is visible.

Transport and provenance do not add another seal. The row H moves only the displayed coordinates, while P and N record provenance and local naming for the same finite packet.

Consequently every diagonal-limit consumer reaches $\mathsf{Real}^{\uparrow}$ through the displayed finite route. Since the carrier has no coordinate for selected representatives, countable choice, open-cover compactness, quotient streams, host equality, or ambient completeness, none can replace the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteNetDiagonalExtraction}^{\uparrow}$ packet is generated from finite net rows, StreamName windows, a DyadicRatCore tolerance budget, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteNetDiagonalExtraction}\uparrow}$, seed)

Scope closed: Definition 7902.1, Definition 7902.2, and Definition 7902.3 seed the finite diagonal-extraction route over net rows, finite windows, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteNetDiagonalExtraction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected representatives, countable choice, open-cover compactness, quotient streams, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite net-row exposure, StreamName window stability, DyadicRatCore tolerance accounting, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteNetDiagonalExtractionUp.

Chapter 7903

A Concrete Naming Certificate for $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, the $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$ packet realizes the finite budget route used by the diagonal-limit construction of section 9238.1. It binds precision requests to finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budgets, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal from chapter 126, chapter 291, chapter 301, and chapter 38. The packet is a finite precision-to-window-to-readback-to-seal carrier; it is not a quotient stream, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7903.1 (Real completion diagonal budget carrier)**.** *A* $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the precision-request row naming the diagonal index and requested regular-Cauchy tolerance.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the requested diagonal index.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row that accounts for the endpoint, middle-modulus, and readback error allowances.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the window and tolerance budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after Q, W, D, R are displayed.*
6. *H records componentwise* hsame *transport for the precision, window, tolerance, readback, seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal-limit request through Q, W, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalBudget}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed precision, window, tolerance, readback, Real seal, continuation, provenance, and local naming rows to a common finite surface.

Theorem 7903.2 (Real completion diagonal budget NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$ *carrier*

$$B = (Q, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the precision request, StreamName window, DyadicRatCore budget, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming; classifier stability under the displayed* hsame *rows; route exactness from* Q, W, D, R *to* E*; and refusal rows excluding quotient streams, selected limits, countable choice, host real equality, open-cover compactness, and ambient completeness.*

Proof. Project the carrier of Definition 7903.1. The listed coordinates are exactly the precision request, finite window, dyadic budget, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Classifier stability is componentwise. Transport through H moves those rows to a common finite surface without adding another representative or quotient coordinate.

Route exactness follows from C. A diagonal-limit request first exposes the precision row Q, then the finite window W, then the dyadic tolerance budget D, then the regular readback R, and only then the Real seal E.

The refusal rows are not extra data. They state that no coordinate of the packet contains quotient streams, selected limits, countable choice, host equality, open-cover compactness, or ambient completeness, so no NameCert obligation can export them. □

Theorem 7903.3 (Real completion diagonal budget vision route)**.** *The diagonal-limit representative of Definition 9238.3 is consumed through an accepted* $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$ *carrier only by the route*

$$Q, W, D, R, E, H, C, P, N.$$

Thus the real-completion vision route factors each precision request through a finite $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalBudget}^{\uparrow}}$ *naming.*

No quotient stream, selected limit, countable-choice schedule, host real equality, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle may replace this finite budget route.*

Proof. Start from the source vision section section 9238.1. Its diagonal representative uses a requested precision to choose a displayed diagonal readback; the carrier records that request as Q.

Replay the finite route. The request opens the StreamName window W, spends the DyadicRatCore budget D, reads the RegSeqRat row R, and only then reaches the Real seal E.

Transport, replay, provenance, and naming stay local to the same packet. The rows H, C, P, N preserve and name the displayed route rather than adding a second completion principle.

Therefore the vision route lands as finite NameCert data. Since the carrier has no quotient stream, selected limit, countable-choice schedule, host equality, open-cover compactness row, or ambient completeness coordinate, none can replace the finite diagonal budget. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionDiagonalBudget}^{\uparrow}$ packet is generated from a precision request, StreamName window, DyadicRatCore tolerance budget, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7903.1, Definition 7903.2, and Definition 7903.3 seed the finite real-completion diagonal-budget route.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCompletionDiagonalBudget}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked precision requests, StreamName window stability, DyadicRatCore budget accounting, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionDiagonalBudgetUp.

Chapter 7904

Metric Completion Unit NameCert Construction

Following Bishop and Bridges constructive completion theory, the $\mathsf{MetricCompletionUnit}^{\uparrow}$ packet records the unit map that sends a displayed metric point into its completion by the constant regular name. It sits between the metric completion carrier of chapter 1506, the Cauchy-completion adjunction of chapter 1456, the stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the completion unit as finite NameCert data, not as an ambient embedding into a preexisting completed object.

Definition 7904.1 (Metric completion unit carrier)**.** *A* $\mathsf{MetricCompletionUnit}^{\uparrow}$ *carrier is a finite packet*

$$U = (X, x, S, R, T, E, H, C, P, N)$$

with the following rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier and x is the displayed source-point history.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *row repeatedly exposing x at each precision request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff certifying the constant stream as a regular rational Cauchy name.*
4. *T is the target* $\mathsf{MetricCompletion}^{\uparrow}$ *row receiving that regular name.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *and metric seal row comparing source distances with the completed endpoint.*
6. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionUnit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, selected limits, ambient completion equality, or a host embedding function.

Theorem 7904.2 (Metric completion unit NameCert obligations)**.** *Every accepted* $\mathsf{MetricCompletionUnit}^{\uparrow}$ *carrier $U = (X, x, S, R, T, E, H, C, P, N)$ exposes the source metric point, constant stream row, regular-sequence handoff, completion target row, metric-real seal, transport, replay, provenance, and local naming rows before any completion-unit consumer reads the completed endpoint. The consumer cannot replace these rows by an ambient dense embedding, a quotient class of Cauchy names, a selected limit, or host equality in the completion.*

Proof. Project the carrier rows of Definition 7904.1. The only source data are the metric carrier X and point history x, and the only completion-facing source is the constant $\mathsf{StreamName}^{\uparrow}$ row S.

Read the regular handoff and target rows. The row R certifies the constant stream as a regular rational name, while T records that same name as a $\mathsf{MetricCompletion}^{\uparrow}$ target; the seal row E is therefore downstream of the displayed stream and regular-sequence data.

Finally apply the structural rows H, C, P, N. They replay exactly the same route under hsame and Cont, record provenance, and name the local certificate, so no quotient class, selected limit, ambient embedding, or host equality coordinate is available. □

Theorem 7904.3 (Metric completion unit real handoff)**.** *For every accepted* $\mathsf{MetricCompletionUnit}^{\uparrow}$ *carrier* $U = (X, x, S, R, T, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *may be read only after the constant* $\mathsf{StreamName}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{MetricCompletion}^{\uparrow}$ *target row have been exposed. The handoff is the finite constant-name route from the source metric point to the completed endpoint; it is not an appeal to ambient real completeness or an external limit operation.*

Proof. Start with the constant stream row S. It repeats the displayed source point history, so every later precision request remains tied to the same metric source row.

Next read the regular rational handoff R and the completion target row T. These rows certify the constant stream as a regular name and then as the completion endpoint named by the packet.

The seal row E is projected only after those rows. Since H, C, P, N merely transport and replay the displayed route, the $\mathsf{Real}^{\uparrow}$ handoff cannot be replaced by an external completion map, a selected Cauchy-class representative, or ambient real completeness. □

Independence witness: metriccompletion, cauchy-completion-adjunction, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{MetricCompletionUnit}^{\uparrow}$ because this packet jointly carries a source metric point, constant stream name, regular handoff, completion target, real seal, transport, replay, provenance, and local naming as one finite unit-map surface.

Chapter 7905

Compact Uniform Radius Selector NameCert Construction

Following Heine–Cantor compact-uniform-continuity practice in Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformRadiusSelector}^{\uparrow}$ packet names the finite lower-bound selector that turns compact local-radius data into a tolerance-only uniform-radius row. It realizes the radius-selection branch of chapter 9237 beside the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-modulus consumer of chapter 369, the rational lower-bound ledger of chapter 37, and the metric-space distance surface of chapter 4230. The packet records finite ProbeBundle data only; it is not an open-cover compactness theorem, an external choice of radii, or a host modulus function.

Definition 7905.1 (Compact uniform radius selector carrier)**.** *A* $\mathsf{CompactUniformRadiusSelector}^{\uparrow}$ *carrier is a finite packet*

$$R = (K, F, B, L, U, T, H, C, P, N)$$

whose rows have the following roles:

1. K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source over the same metric carrier used by the consumer.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-radius source exposing local radii for the requested tolerance.*
3. B *is the finite* ProbeBundle *row of compact centers and source coverage witnesses.*
4. L *is the* $\mathsf{Rat}^{\uparrow}$ *positive lower-bound fold over the bundled local radii.*
5. U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *target row receiving the selected radius.*
6. T *is the* $\mathsf{Met}^{\uparrow}$ *triangle and distance-transport row used to move from center radii to arbitrary source points.*
7. H, C, P, N *are componentwise* hsame *transport, displayed* Cont *replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected subcovers, countable choice, host infima over radii, quotient maps, or an external modulus selector.

Theorem 7905.2 (Compact uniform radius selector NameCert obligations)**.** *Every accepted* CompactUniformRadiu *carrier* $R = (K, F, B, L, U, T, H, C, P, N)$ *exposes the compact metric finite-net row, continuous pointwise-radius row, finite ProbeBundle coverage row, rational lower-bound fold, uniform-modulus target, metric triangle transport,* hsame *transport,* Cont *replay, provenance, and local naming rows*

before a compact-uniform-continuity consumer reads a selected radius. The selection cannot be replaced by an ambient infimum, an unrestricted open-cover argument, countable choice, selected representatives, or a host modulus function.

Proof. Project the carrier of Definition 7905.1. The compact source K, pointwise-radius source F, and finite bundle B are the only rows from which a radius-selection consumer may start.

Read the lower-bound and target rows next. The rational fold L is built over the finite bundle and supplies the radius consumed by the $\mathsf{UniformModulus}^{\uparrow}$ row U, while the metric row T provides the triangle transport needed for arbitrary source points.

Finally apply the structural rows H, C, P, N. They replay the same finite route under transport and provenance, so no ambient infimum, open-cover compactness principle, countable choice, selected representative, or host modulus coordinate is exposed. □

Theorem 7905.3 (Compact uniform radius selector vision handoff)**.** *The finite lower-bound fold used in chapter 9237 is realized by the* $\mathsf{CompactUniformRadiusSelector}^{\uparrow}$ *route*

$$K, F, B, L \longmapsto U, T, H, C, P, N.$$

Thus the compact-uniform-continuity consumer reads a radius through compact finite-net data, continuous pointwise-radius data, a finite ProbeBundle *fold, rational positivity, metric triangle transport, and local NameCert rows, rather than through a host proof that a positive minimum of radii exists.*

Proof. Begin with the compact finite-net and pointwise-radius rows. They supply the bundled centers and local radii that the compact-uniform-continuity construction needs before any global radius is visible.

Use the finite lower-bound fold row L. Because the fold is carried by the displayed ProbeBundle and $\mathsf{Rat}^{\uparrow}$ positivity data, the selected radius is a named history, not an external minimum or choice.

Transport the selected row through U, T, H, C, P, N. The uniform-modulus target receives the radius only after metric triangle transport and structural replay, so the vision's lower-bound step is a concrete NameCert route. □

Independence witness: compactmetric, continuousmap, uniformmodulus, rat, metric-space: no listed sibling is bijective to $\mathsf{CompactUniformRadiusSelector}^{\uparrow}$ because this packet jointly carries compact finite-net data, pointwise-radius rows, finite ProbeBundle lower-bound folding, UniformModulus handoff, metric triangle transport, replay, provenance, and local naming.

Chapter 7906

Located Dyadic Apartness Window NameCert Construction

Following Bishop and Bridges-style constructive real analysis, the $\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}$ packet records the finite dyadic window in which two regular real names are witnessed apart before a Real-facing comparison may be consumed. It sits over the dyadic endpoint arithmetic of chapter 291, the finite observation rows of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the located apartness surface of chapter 6450. The packet supplies a constructive comparison window for Real and Completion routes without importing trichotomy, quotient Real equality, or ambient completeness.

Definition 7906.1 (Located dyadic apartness window carrier)**.** *A* $\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (L, U, S_0, S_1, R_0, R_1, G, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the finite lower-endpoint history in* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording the dyadic lower cells inspected for the two regular names.*
2. *U is the finite upper-endpoint history in* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording the matching upper cells and the endpoint-order ledger used by the window.*
3. *S_0 and S_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *observation windows opened at the displayed dyadic precision.*
4. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for those windows.*
5. *G is a positive dyadic gap row proving that the displayed lower and upper endpoint histories are separated before the terminal Real seal is read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is consumed only after $L, U, S_0, S_1, R_0, R_1, G$ have already been displayed.*
7. *H stores componentwise* hsame *transport for endpoints, stream windows, readback rows, gap ledger, seal, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay through the same finite endpoint and readback route.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedRealApartness}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}}$ *row.*
The carrier has no coordinate for trichotomy, quotient Real equality, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Theorem 7906.2 (Located dyadic apartness window stability)**.** *For every accepted* LocatedDyadicApartnessWindow *carrier*

$$W = (L, U, S_0, S_1, R_0, R_1, G, E, H, C, P, N),$$

componentwise hsame *transport of the dyadic endpoint histories, stream observations, regular rational readbacks, positive dyadic gap ledger, and terminal Real seal preserves the same located apartness window. Consequently every Real or Completion consumer receives the transported finite window through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows, not through trichotomy, quotient Real equality, selected representatives, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 7906.1. The only analytic rows are the dyadic endpoint histories L, U, the finite stream windows S_0, S_1, the regular rational readbacks R_0, R_1, the positive gap ledger G, and the terminal Real seal E.

Transport first applies to the endpoint histories. Componentwise hsame sends the displayed lower and upper dyadic cells to transported endpoint histories while preserving the endpoint-order ledger carried by L, U.

Transport then applies to the observation and readback rows. The windows S_0, S_1 and readbacks R_0, R_1 remain attached to the same dyadic precision because the replay row C follows the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route.

Finally transport preserves the positive dyadic gap row G and the terminal seal E. The gap is still read before the Real seal, and the provenance and local naming rows P, N name exactly that transported packet. Since no row contains trichotomy, quotient Real equality, selected representatives, countable choice, or ambient completeness, the transported window exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}$ packet is generated from finite dyadic lower and upper endpoint histories, paired StreamName windows, RegSeqRat readbacks, a positive dyadic gap ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}}$, seed)

Scope closed: Definition 7906.1 and Definition 7906.2 seed the finite located dyadic apartness window route from endpoint histories through stream and regular-rational readback to the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedDyadicApartnessWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient Real equality, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order exactness, paired-window readback, positive dyadic gap preservation, Real seal nonescape, transport and replay stability, and exclusion coverage for BEDC.Derived.LocatedDyadicApartnessWindowUp.

Chapter 7907

A Concrete Naming Certificate for $\mathsf{LipschitzBanach}^{\uparrow}$

Following Weaver and Heinonen on Lipschitz functions, the $\mathsf{LipschitzBanach}^{\uparrow}$ packet records the completion-facing surface for a bounded family of real-valued Lipschitz maps whose shared Lipschitz control and uniform Cauchy ledger are displayed before any complete function-space read is public. It sits after the Lipschitz function-space row of chapter 7502, the uniform limit boundary of chapter 463, and the terminal real-name route of chapter 38; it also consumes the finite-window and regular-rational completion spine supplied by $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. The packet names a complete Lipschitz function-space handoff as finite BEDC data, not as a quotient function space or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7907.1 (LipschitzBanach carrier)**.** *A* $\mathsf{LipschitzBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, F, K, U, V, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LipschitzSpace}^{\uparrow}$ *row, exposing the metric-domain, real-valued graph, and Lipschitz-inequality surface before any function-space consumer reads the packet.*
2. *F is the bounded Real-valued map family row, listing the inspected maps and their finite value bounds as* $\mathsf{Real}^{\uparrow}$*-facing data.*
3. *K is the shared* $\mathsf{LipschitzConstant}^{\uparrow}$ *row used by every listed map in F.*
4. *U is the uniformly Cauchy ledger for the family, recording finite tolerance windows and pairwise comparison budgets without selecting a limit function.*
5. *V is the* $\mathsf{UniformLimit}^{\uparrow}$ *row, available only after F, K, U have been displayed.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row for the displayed function-space metric boundary, read as a completion-facing obligation surface rather than as host Banach-space completeness.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected indices, source probes, value probes, and tolerance requests.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and comparison budgets.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for displayed values and norms, reached only after S, R, U, V have been consumed.*

10. *H records componentwise* hsame *transport for the Lipschitz-space, bounded-family, shared-constant, uniform-Cauchy, uniform-limit, complete-metric, stream-window, regular-readback, real-seal, replay, provenance, and local-name rows.*
11. *C records displayed* Cont *replay from a finite function-space request through $L, F, K, U, V, M, S, R, E$.*
12. *P is the* Pkg *provenance over* $\mathsf{LipschitzBanach}^{\uparrow}$, $\mathsf{LipschitzSpace}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzBanach}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, F, K, U, V, M, S, R, E, H, C, P, N$. It has no coordinate for host Banach-space completeness, quotient equality of functions, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden complete function-space object.*

Theorem 7907.2 (LipschitzBanach NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzBanach}^{\uparrow}$ *carrier $B = (L, F, K, U, V, M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LipschitzBanach}^{\uparrow}}$ *surface consists exactly of carrier admission for the LipschitzSpace row, bounded Real-valued map family row, shared Lipschitz constant row, uniformly Cauchy ledger, UniformLimit row, CompleteMetric row, StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed continuation replay, package provenance, and local naming. The obligations require the same shared constant K to control every listed map, the uniform Cauchy ledger U to be displayed before the uniform-limit row V, the StreamName and RegSeqRat rows S, R to account for the finite value and norm windows, and the Real seal E to remain downstream of those rows.*

Proof. Project the accepted carrier. Its analytic rows are $L, F, K, U, V, M, S, R, E$, and its structural rows are H, C, P, N.

The Lipschitz route starts with L, F, K. The LipschitzSpace row supplies the displayed graph and inequality surface, the bounded family row lists the inspected maps and value bounds, and the shared Lipschitz row is read once for every listed member.

The completion-facing route then uses U, V, M. The uniformly Cauchy ledger records finite pairwise comparison budgets, the UniformLimit row is exposed only after those budgets are visible, and the CompleteMetric row names the displayed metric boundary without importing a host Banach-space theorem.

Finally S, R, E, H, C, P, N keep the read inside finite BEDC data. Since no carrier coordinate stores quotient equality of functions, a selected limit, countable choice, ambient Real completeness, or a hidden complete function space, the listed rows exhaust the local NameCert surface. □

Theorem 7907.3 (LipschitzBanach complete uniform-limit route)**.** *Every completion-facing consumer of an accepted* $\mathsf{LipschitzBanach}^{\uparrow}$ *packet must factor through*

$$\mathsf{LipschitzSpace}^{\uparrow},\quad \mathsf{LipschitzConstant}^{\uparrow},\quad \mathsf{UniformLimit}^{\uparrow},\quad \mathsf{CompleteMetric}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The route supplies a bounded Real-valued map family, one shared Lipschitz constant, a uniformly Cauchy ledger, finite value and norm windows, regular rational readback, and terminal Real seals before any complete uniform-limit handoff is public. It exports no host Banach-space completeness, quotient equality of functions, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or unledgered function-space metric.*

Proof. Begin with the obligation theorem Definition 7907.2. It makes the LipschitzSpace row, bounded-family row, shared constant, uniformly Cauchy ledger, UniformLimit row, and CompleteMetric row visible carrier data.

Read the sibling routes in order. The LipschitzSpace surface of chapter 7502 supplies the function-space graph and inequality rows, while chapter 463 is reached only after the uniform Cauchy ledger has been displayed.

The Real and Completion handoff is finite. The StreamName windows and RegSeqRat readback feed the terminal Real seal of chapter 38, and the CompleteMetric row records only the displayed completion-facing metric boundary.

Transport, replay, provenance, and local naming preserve that same route. Therefore every complete uniform-limit consumer receives the listed finite packet and no quotient function-space equality, selected limit, countable-choice object, host Banach-space completeness theorem, or ambient Real completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzBanach}^{\uparrow}$ packet is generated from a LipschitzSpace row, bounded Real-valued map family row, shared Lipschitz constant row, uniformly Cauchy ledger, UniformLimit row, CompleteMetric row, StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzBanach}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7907.1, Definition 7907.2, and Definition 7907.3 seed the finite complete Lipschitz function-space packet over shared Lipschitz control, uniform Cauchy data, uniform-limit handoff, completion boundary, finite windows, regular readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzBanach}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Banach-space completeness, quotient equality of functions, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared Lipschitz exactness, uniformly Cauchy ledger exactness, uniform-limit ordering, complete-metric boundary, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzBanachUp.

Chapter 7908

A Concrete Naming Certificate for $\mathsf{LocatedCompletionMonad}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{LocatedCompletionMonad}^{\uparrow}$ packet records the finite naming certificate by which located completion routes compose through unit and join rows. It sits after the located completion surface of chapter 4150, the Cauchy completion monad of chapter 378, the monad-law ledger of chapter 3834, the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the Real seal boundary of chapter 38. The packet is a displayed NameCert for composing located completion data; it is not a quotient completion, a selected limit, a countable-choice schedule, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7908.1 (Located completion monad carrier)**.** *A* $\mathsf{LocatedCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, M, U, J, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCompletion}^{\uparrow}$ *source row, including the displayed located-completion route consumed before any monad operation is public.*
2. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row naming the finite completion transformer and its schedule discipline.*
3. *U is the unit row, sending source located data to the completion-facing packet through displayed windows and regular rational readback.*
4. *J is the join row, flattening a completion-of-completion read by replaying the same located completion and Cauchy-completion schedules.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by both unit and join reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling requested precision, located comparison, and replay.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, M, U, J, S, R, D have been displayed.*
9. *H records componentwise* hsame *transport for the located completion, monad, unit, join, stream, readback, dyadic, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *routes from a completion request through the same ordered rows.*
11. *P is the* Pkg *provenance over* LocatedCompletionMonad$^{\uparrow}$, LocatedCompletion$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionMonadLaw$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{LocatedCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, M, U, J, S, R, D, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, host equality of completed points, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompletionMonadUp.LocatedCompletionMonadTasteGate single carrier alignment`

Theorem 7908.2 (Located completion monad NameCert obligations)**.** *For every accepted* LocatedCompletionMona *carrier $Q = (L, M, U, J, S, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{LocatedCompletionMonad}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows, located-source visibility through L, completion-monad visibility through M, unit visibility through U, join visibility through J,* StreamName$^{\uparrow}$ *window preservation through S,* RegSeqRat$^{\uparrow}$ *readback preservation through R,* DyadicRatCore$^{\uparrow}$ *tolerance preservation through D,* Real$^{\uparrow}$ *seal preservation through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. The carrier exports no quotient completion, selected limit, countable-choice schedule, host equality of completed points, or ambient* Real$^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7908.1. The analytic rows are the located completion source L, the Cauchy completion monad row M, the unit and join rows U, J, the finite stream windows S, the regular rational readback R, the dyadic tolerance ledger D, and the Real seal E.

Read the unit route first. A source located-completion request must display L and M, then pass through the same S, R, D readback discipline before the terminal seal E is visible.

Read the join route next. A completion-of-completion request is flattened only by replaying the displayed unit and monad schedules through J, with the StreamName windows, regular readback, dyadic tolerances, and Real seal preserved as the same finite rows.

Finally attach H, C, P, N. These rows transport, replay, source, and name the displayed packet, and none supplies quotient completion equality, selected limits, countable choice, host equality, or ambient Real completeness. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 7908.3 (Located completion monad unit exactness)**.** *For every accepted* LocatedCompletionMonad$^{\uparrow}$ *carrier $Q = (L, M, U, J, S, R, D, E, H, C, P, N)$, every unit read through U exposes the located completion source row L, the Cauchy completion monad row M, the shared finite-window row S, the regular rational readback row R, the DyadicRatCore tolerance row D, and the Real seal row E before the local* NameCert$_{\text{LocatedCompletionMonad}^{\uparrow}}$ *row is read. In particular, the unit route is the displayed finite route*

$$L, M \longmapsto U \longmapsto S, R, D \longmapsto E, H, C, P, N,$$

not a selected limit, quotient completion equality, countable-choice schedule, or ambient completeness premise.

Proof. Project the carrier of Definition 7908.1. The unit row U is not independent data; it is listed between the source rows L, M and the finite readback rows S, R, D, E.

A unit request therefore first displays the located completion source and the completion-monad row. It can then read U only as the route that sends those source rows into the shared StreamName, RegSeqRat, and DyadicRatCore corridor.

The Real seal and local certificate are terminal for the unit route. The rows E, H, C, P, N preserve, replay, source, and name the same displayed packet, and none supplies a quotient, selected limit, choice schedule, host equality, or ambient completeness coordinate. □

Theorem 7908.4 (Located completion monad join exactness)**.** *For every accepted* $\mathsf{LocatedCompletionMonad}^{\uparrow}$ *carrier* $Q = (L, M, U, J, S, R, D, E, H, C, P, N)$*, every join read through* J *flattens completion-of-completion data by replaying the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows that the unit route uses. The join route is the displayed finite route*

$$L, M, U \longmapsto J \longmapsto S, R, D \longmapsto E, H, C, P, N.$$

It exports no detached completed point, selected representative, quotient completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from a completion-of-completion read of the accepted carrier. The join row J is downstream of the located source row L, the monad row M, and the unit row U, so flattening cannot begin from a detached completed point.

The replayed readback rows are exactly S, R, D, E. The same StreamName windows, regular rational readbacks, dyadic tolerance ledger, and Real seal used by the unit route are inspected again when J flattens the nested completion data.

Finally the structural rows close the route without adding new analytic data. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same finite packet, so no quotient, selector, choice schedule, host equality, or ambient completeness premise is available to the join read. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionMonad}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCompletionMonad}^{\uparrow}}$, obligation)

Scope closed: Definition 7908.1, Definition 7908.2, Definition 7908.3, and Definition 7908.4 display the finite located completion monad obligation surface over $\mathsf{LocatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCompletionMonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host equality of completed points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-source visibility, unit and join exactness, StreamName window preservation, RegSeqRat readback preservation, DyadicRatCore tolerance preservation, Real seal preservation, transport, replay, provenance, and non-escape over the same displayed packet.

Constructive story: A $\mathsf{LocatedCompletionMonad}^{\uparrow}$ packet is generated from located completion source data, a Cauchy completion monad row, unit and join rows, finite stream windows, regular rational readback, dyadic tolerances, a Real seal, and structural NameCert rows.

Chapter 7909

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$

Following Bishop and Bridges and the Bourbaki-style reflector discipline for completions, the $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$ packet records the finite triangle by which a Cauchy-completion reflector, its unit route, and the Real-facing regular Cauchy handoff are read as one NameCert object. It consumes the completion source discipline of chapter 3717, the monad and unit rows of chapter 378 and chapter 2256, the metric reflector route of chapter 2149, and the finite rational readback stack of chapter 291, chapter 126, chapter 301, and chapter 38. The chapter names the displayed completion-reflector triangle; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7909.1 (CauchyCompletionReflectorTriangle carrier)**.** *A* $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, R, U, M, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopCompletion}^{\uparrow}$ *source row naming the constructive completion source before any reflector consumer reads it;*
2. *R is the* $\mathsf{MetricCompletionReflector}^{\uparrow}$ *comparison row, recording the displayed reflector comparison against the source metric presentation;*
3. *U is the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *unit-triangle row, tying the source point, its unit image, and the reflector comparison to one finite route;*
4. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row recording the completion-facing unit and bind surface consumed by the triangle;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for finite dyadic comparison;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected rational name;*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance cells;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after B, R, U, M, D, S, Q have been displayed;*
9. *H records componentwise* hsame *transport for the source, reflector, unit-triangle, completion, tolerance, window, readback, Real-seal, provenance, and local naming rows;*

10. *C records the displayed* Cont *replay route from the completion source through the reflector comparison, unit triangle, finite rational readback, and Real seal;*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$, $\mathsf{BishopCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$ $\mathsf{CompletionUnitTriangle}^{\uparrow}$, $\mathsf{MetricCompletionReflector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, U, M, D, S, Q, E, H, C, P, N$ and their displayed hsame *and* Cont *transports. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable-choice schedules, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorTriangleUp.CauchyCompletionReflectorTriangleUp single carrier alignment`

Theorem 7909.2 (CauchyCompletionReflectorTriangle NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$ *carrier $T = (B, R, U, M, D, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-source exposure, reflector comparison, unit-triangle exposure, completion monad scheduling, DyadicRatCore tolerance accounting, StreamName finite-window readback, RegSeqRat regular readback, Real-seal ordering, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. No local obligation is stored outside those rows.*

Proof. Project the carrier of Definition 7909.1. The displayed tuple is exactly $B, R, U, M, D, S, Q, E, H, C, P, N$, so every accepted packet exposes the completion source, reflector comparison, unit triangle, monad schedule, finite rational readback stack, Real seal, and structural rows before any consumer can read the triangle.

Read the analytic rows in order. The source row B enters the reflector comparison R, the unit-triangle row U binds the source and reflected point, and the monad row M records the completion-facing unit and bind surface. The dyadic tolerance, finite-window, and regular-readback rows D, S, Q then supply the finite rational data needed before E is exposed.

The structural rows close the certificate. Componentwise hsame transport and displayed Cont replay preserve the same finite triangle through H and C, while P and N record provenance and local naming. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 7909.3 (CauchyCompletionReflectorTriangle handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or metric-reflector consumer of an accepted* $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$ *packet must read the route*

$$\mathsf{BishopCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionReflector}^{\uparrow} \longmapsto \mathsf{CompletionUnitTriangle}^{\uparrow} \longmapsto \mathsf{CauchyCompletionMonad}^{\uparrow} \longmapsto$$

through the same displayed carrier $T = (B, R, U, M, D, S, Q, E, H, C, P, N)$. The handoff exports no quotient completion, selected limit, host equality of completed objects, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the NameCert obligations of Definition 7909.2. They force every accepted handoff to expose the completion source, reflector comparison, unit triangle, completion monad row, dyadic tolerance row, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming.

The reflector-facing part is read from B, R, U, M. These rows record the displayed completion source, comparison map, unit-triangle route, and completion-facing schedule before any metric reflector consumer can treat the packet as a triangle.

The Real-facing part is read from D, S, Q, E. The dyadic tolerance and finite window rows determine the regular rational readback, and the Real seal is downstream of those finite rows rather than a selected quotient representative.

Finally use H, C, P, N to transport, replay, source, and name the same finite handoff. Because none of the rows stores quotient equality, selected limits, host equality, choice, or ambient completeness, no such object can be exported by the triangle packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}$ packet is generated from a BishopCompletion source row, MetricCompletionReflector comparison, CompletionUnitTriangle route, CauchyCompletionMonad schedule, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7909.1, Definition 7909.2, and Definition 7909.3 seed the finite completion-reflector triangle route through source, reflector, unit, rational readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector comparison, unit-triangle routing, rational readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionReflectorTriangleUp.

Chapter 7910

A Concrete Naming Certificate for $\mathsf{DeckTransformation}^{\uparrow}$

Following Hatcher and Spanier on covering-space automorphisms, the $\mathsf{DeckTransformation}^{\uparrow}$ packet records the finite BEDC surface by which a covering projection admits a fiber-preserving automorphism route with displayed inverse and composition rows. It sits beside the path-lifting certificate of chapter 2275, the homotopy rows of chapter 156, the continuous-map carrier of chapter 130, and the topology surface of chapter 88. The packet names deck transformation data as finite carrier and NameCert rows; it does not import a universal cover, monodromy action, quotient path space, selected global sheet, or host equality of maps.

Definition 7910.1 (DeckTransformation carrier)**.** *A* $\mathsf{DeckTransformation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\pi, a, u, i, c, s, h, r, p, n)$$

with the following displayed rows:

1. *π is the covering projection row, read as a* $\mathsf{ContinuousMap}^{\uparrow}$ *packet from the displayed total-space topology to the base* $\mathsf{Topology}^{\uparrow}$ *row.*
2. *a is the automorphism route. It records a displayed total-space self-map whose projection readback follows π on every accepted finite test row.*
3. *u is the base-preservation ledger. It exposes the row-by-row comparison witnessing that π after a reads as the same covering projection route.*
4. *i is the inverse row. It records the displayed inverse route and the two finite identity readbacks for a followed by i and i followed by a.*
5. *c is the composition row. It records closure of accepted automorphism routes under displayed composition over the same projection ledger.*
6. *s is the sheet-action classifier. It compares finite sheet reads by the covering projection, path-lifting route, and endpoint seal, not by a host-level global sheet selector.*
7. *h records componentwise* hsame *transport for the projection, automorphism route, preservation ledger, inverse row, composition row, sheet classifier, replay, provenance, and local naming rows.*
8. *r records displayed* Cont *replay from a deck-transformation request through $\pi, a, u, i, c, s, h, p, n$.*
9. *p is the* Pkg *provenance over* $\mathsf{DeckTransformation}^{\uparrow}$, $\mathsf{CoveringSpacePathLifting}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. n is the local $\mathsf{NameCert}_{\mathsf{DeckTransformation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a universal covering space, monodromy action, quotient path object, hidden group representation, selected global sheet, or host equality of continuous maps.

Theorem 7910.2 (DeckTransformation NameCert obligations)**.** *For every accepted* $\mathsf{DeckTransformation}^{\uparrow}$ *carrier*

$$D = (\pi, a, u, i, c, s, h, r, p, n),$$

the local $\mathsf{NameCert}_{\mathsf{DeckTransformation}^{\uparrow}}$ *surface consists exactly of carrier admission, projection preservation through π, a, u, inverse exactness through i, composition closure through c, sheet-classifier stability through s, componentwise* hsame *transport through h, displayed* Cont *replay through r,* Pkg *provenance through p, and local naming through n. Every deck-transformation consumer can inspect these rows, but cannot escape to universal-cover data, quotient-path data, monodromy actions, fundamental-group representations, selected global sheets, or host equality of maps.*

Proof. Carrier admission is projection from Definition 7910.1. The accepted tuple displays the covering projection, automorphism route, base-preservation ledger, inverse row, composition row, sheet classifier, transport, replay, provenance, and local naming rows.

Projection preservation is the content of u. It forces every accepted read of a to return through the same covering projection route π, so the automorphism is fiber-preserving at the level visible to the packet.

Inverse exactness is read from i. The two identity readbacks are finite rows in the carrier, and no host-level equality of continuous maps is added beyond those displayed rows.

Composition closure is read from c. Whenever two accepted automorphism routes share the displayed covering projection ledger, their composite is admitted by the same finite surface and remains attached to π.

The sheet-action classifier s is stable because it is computed from the projection ledger, the path-lifting route of chapter 2275, and the endpoint seal already named there. Transport, replay, provenance, and naming through h, r, p, n preserve the same rows, so the certificate has no remaining place to store universal-cover data, quotient-path data, monodromy actions, or fundamental-group representations. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeckTransformation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DeckTransformation}^{\uparrow}$ packet is generated from a covering projection, a fiber-preserving automorphism route, base-preservation ledger, inverse row, composition row, sheet-action classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DeckTransformation}^{\uparrow}}$, seed)

Scope closed: Definition 7910.1 and Definition 7910.2 seed the finite projection-preservation, inverse, composition, sheet-classifier, transport, replay, provenance, and local naming surface for covering-space automorphisms.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DeckTransformation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close universal covers, monodromy actions, fundamental-group representations, quotient path spaces, selected global sheets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked projection preservation, inverse exactness, composition closure, sheet-classifier stability, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DeckTransformationUp.

Chapter 7911

A Concrete Naming Certificate for DiniDerivativeLowerEnvelope$^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the DiniDerivativeLowerEnvelope$^{\uparrow}$ packet records the finite lower-envelope surface by which a displayed real-valued function exposes lower Dini rows before any derivative-envelope or semicontinuity consumer can inspect them. It sits beside the one-sided Dini carrier of chapter 2041, the upper envelope sibling of chapter 7459, the lower-semicontinuity packet of chapter 1536, the derivative quotient boundary of chapter 122, and the terminal real-name seal of chapter 38. The chapter names only finite lower-envelope windows and readback rows; it does not claim classical completeness, selected limiting slopes, quotient real equality, or a host lower semicontinuous hull.

Definition 7911.1 (DiniDerivativeLowerEnvelope carrier)**.** *A* DiniDerivativeLowerEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, W, L, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed real-valued function row, read through the same finite observation surface used by the sibling* DiniDerivative$^{\uparrow}$ *and* Derivative$^{\uparrow}$ *packets.*
2. *W is the lower-envelope window row, listing the finite one-sided neighborhoods and lower secant comparisons available to the packet.*
3. *L is the* LowerSemicontinuous$^{\uparrow}$ *row that records the lower-envelope visibility surface for downstream epigraph or threshold consumers.*
4. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row for radii, slope lower bounds, and comparison slack.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for endpoint values, secant numerators, step sizes, and dyadic lower-bound approximants.*
6. *E is the terminal* Real$^{\uparrow}$ *seal reached only after F, W, L, Q, R have been displayed.*
7. *H records componentwise* hsame *transport for the function, lower-envelope window, lower-semicontinuity, dyadic, readback, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for refinements of the same finite lower-envelope window.*
9. *P is the* Pkg *provenance over* DiniDerivativeLowerEnvelope$^{\uparrow}$, DiniDerivative$^{\uparrow}$, UpperDiniDerivative$^{\uparrow}$, LowerSemicontinuous$^{\uparrow}$, Derivative$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $F, W, L, Q, R, E, H, C, P, N$. It has no coordinate for a classical liminf, selected derivative value, quotient equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external lower-semicontinuous hull.*

Theorem 7911.2 (DiniDerivativeLowerEnvelope NameCert obligations)**.** *For every accepted* DiniDerivativeLowerE *carrier*

$$G = (F, W, L, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed function row, lower-envelope windows, lower-semicontinuity row, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, package provenance, and local naming; lower-envelope exactness requiring every lower-bound read to pass through W, Q, R; lower-semicontinuity compatibility requiring the row L to remain visible before threshold consumers read the envelope; and nonescape from classical liminf data, selected derivative values, quotient real equality, countable choice, ambient Real completeness, and host lower-semicontinuous hulls.*

Proof. Project the carrier of Definition 7911.1. The accepted packet exposes only the displayed function row F, lower-envelope windows W, lower-semicontinuity row L, dyadic tolerance row Q, regular readback row R, Real seal E, and structural rows H, C, P, N.

Lower-envelope exactness is finite. A lower-bound request must first read the inspected window in W, then the dyadic tolerance and slack data in Q, and then the regular rational readback in R before the Real seal can be consumed.

The lower-semicontinuity row remains attached to the same route. Thus an epigraph or threshold consumer can inspect the lower envelope only through the displayed row L, with transport and replay preserving the same finite packet.

Since no carrier coordinate stores a classical liminf, selected derivative value, quotient equality, choice schedule, ambient completeness theorem, or external hull, the listed rows exhaust the local NameCert obligations. □

Theorem 7911.3 (DiniDerivativeLowerEnvelope lower-envelope handoff)**.** *Every* $\mathsf{LowerSemicontinuous}^{\uparrow}$*-*, $\mathsf{DiniDerivative}^{\uparrow}$*-, or derivative-facing consumer of an accepted* $\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}$ *packet must read the finite route*

$$F, W, L, Q, R, E, H, C, P, N$$

before using the lower-envelope comparison. The handoff supplies displayed function data, lower-envelope windows, lower-semicontinuity visibility, DyadicRatCore tolerances, RegSeqRat readback, the terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming, but it supplies no upper-envelope row, two-sided derivative, selected limiting slope, quotient real equality, countable-choice selector, or ambient Real completeness.

Proof. Begin with the obligation theorem Definition 7911.2. It identifies the lower envelope as a finite route through $F, W, L, Q, R, E, H, C, P, N$, not as an external limiting operator.

Read the sibling Dini and derivative surfaces next. The rows F, W, Q, R provide the finite secant-window and tolerance data needed by a Dini or derivative-facing consumer, while the Real seal E remains downstream of those rows.

Finally pass to lower-semicontinuity consumers through L. The row L keeps threshold and epigraph reads on the same displayed packet, and the structural rows preserve that packet under

transport, replay, provenance, and local naming. Therefore the handoff cannot export upper-envelope data, a two-sided derivative, selected limiting slopes, quotient equality, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}$ consumer reaching a lower-envelope read without the displayed function row, lower-envelope windows, LowerSemicontinuous row, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming refutes the carrier.

Independence witness: dini-derivative, upper-dini-derivative, lowersemicontinuous, derivative, real: no listed sibling supplies the whole lower-envelope packet $G = (F, W, L, Q, R, E, H, C, P, N)$, because each sibling contributes only one-sided Dini windows, upper rows, threshold visibility, quotient inspection, or a terminal real seal while this chapter binds the lower-envelope route in one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}$ packet is generated from displayed function data, lower-envelope windows, a LowerSemicontinuous visibility row, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7911.1, Definition 7911.2, and Definition 7911.3 seed the finite lower-envelope packet from Dini derivative windows to lower-semicontinuity and derivative-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim classical liminf existence, selected derivative values, two-sided derivative existence, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-envelope exactness, lower-semicontinuity compatibility, DyadicRatCore tolerance exposure, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DiniDerivativeLowerEnvelopeUp.

Chapter 7912

A Concrete Naming Certificate for $\mathsf{MoorePlane}^{\uparrow}$

Following Moore and Kelley on the tangent-disc topology of the upper half-plane, the $\mathsf{MoorePlane}^{\uparrow}$ packet records the finite BEDC surface for the classical Moore plane boundary object. It sits beside the open-set and neighbourhood discipline of chapter 88 and the terminal real-coordinate carrier of chapter 38. The packet names upper-half-plane points, boundary-line points, tangent-disc neighbourhoods, real-coordinate seals, transport, replay, provenance, and local naming rows without importing a host topology, separation theorem, metrizability theorem, quotient equality of points, or selected neighbourhood basis.

Definition 7912.1 (MoorePlane carrier)**.** *A* $\mathsf{MoorePlane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (U, B, T, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the upper-half-plane point row, listing the inspected points with positive second coordinate.*
2. *B is the boundary-line point row, listing the inspected real-axis points.*
3. *T is the tangent-disc neighbourhood row, recording finite tangent-disc requests over U and B.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate seal for all displayed coordinates and radii.*
5. *H records componentwise* hsame *transport for the upper-half-plane, boundary-line, tangent-disc, real-seal, replay, provenance, and local-name rows.*
6. *C records displayed* Cont *replay from a topology-neighbourhood request through U, B, T, R.*
7. *P is the* Pkg *provenance over* $\mathsf{MoorePlane}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MoorePlane}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host topological space, a global basis theorem, selected infinitary neighbourhood basis, quotient equality of points, metrizability result, normality result, or ambient real-plane topology.

Theorem 7912.2 (MoorePlane NameCert obligations)**.** *For every accepted* $\mathsf{MoorePlane}^{\uparrow}$ *carrier $M = (U, B, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MoorePlane}^{\uparrow}}$ *surface consists exactly of carrier admission for upper-half-plane points, boundary-line points, tangent-disc neighbourhood requests, real-coordinate seals, componentwise transport, displayed continuation replay, package provenance, and*

local naming. Every neighbourhood consumer must read the topology surface through the displayed tangent-disc row T and the real-coordinate seal R before transport, replay, provenance, or local naming is consumed.

Proof. Project the carrier of Definition 7912.1. The point rows are U and B, the only neighbourhood row is T, and the only coordinate seal is R.

The topology-facing read is therefore finite. The tangent-disc row records the neighbourhood request, while the real-coordinate seal records the inspected center and radius data before the structural rows are available.

The rows H, C, P, N transport, replay, source, and name that same packet. Since no other row exists, the local NameCert obligations cannot require a host topology, selected basis, quotient equality of points, metrizability theorem, normality theorem, or ambient real-plane topology. □

Theorem 7912.3 (MoorePlane upper-half-plane route)**.** *Every* $\mathsf{MoorePlane}^{\uparrow}$ *consumer that reads an upper-half-plane neighbourhood must factor through the topology carrier of chapter 88, the real-coordinate seal of chapter 38, and the displayed rows U, T, R, H, C, P, N of Definition 7912.1. The route exposes finite upper-half-plane point data and tangent-disc neighbourhood data only; it exports no host open-set lattice, quotient point equality, selected neighbourhood basis, countable choice, metrizability claim, or ambient* $\mathsf{Real}^{\uparrow}$ *plane completeness.*

Proof. Start with Definition 7912.1. An upper-half-plane neighbourhood read begins at U and reaches the tangent-disc row T before any structural row can be consumed.

The coordinate data are sealed by R through the real-line carrier cited above. This keeps center and radius reads inside displayed $\mathsf{Real}^{\uparrow}$ data rather than a host plane.

Finally H, C, P, N preserve and name the same finite route. Because the carrier contains no separate open-set lattice, quotient equality, selected basis, countable-choice schedule, metrizability row, or ambient completeness row, the stated route exhausts the upper-half-plane consumer surface. □

Theorem 7912.4 (MoorePlane nonescape)**.** *Every* $\mathsf{MoorePlane}^{\uparrow}$ *consumer generated from U, B, T, R, H, C, P, N remains inside the displayed boundary-object route. Boundary-line neighbourhoods may use the tangent-disc row T and the real-coordinate seal R, but they cannot become host separation theorems, metrizability counterexamples, selected neighbourhood bases, quotient point equalities, countable-choice objects, or ambient real-plane topology.*

Proof. Induct over the displayed BHist packet. The base read exposes no topology beyond the listed point rows and real-coordinate seal.

In the finite step case, a new neighbourhood read is admitted only through T over the already displayed point and coordinate rows. The structural rows then transport, replay, source, and name that same read.

Thus every accepted consumer remains inside U, B, T, R, H, C, P, N. The excluded host-topology, basis-selection, quotient, choice, metrizability, and ambient-plane principles have no row in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MoorePlane}^{\uparrow}}$.

Constructive story: A $\mathsf{MoorePlane}^{\uparrow}$ packet is generated from upper-half-plane points, boundary-line points, tangent-disc neighbourhood rows, Real coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MoorePlane}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7912.1, Definition 7912.2, Definition 7912.3, and Definition 7912.4 seed the finite Moore plane topology-neighbourhood packet over displayed real-coordinate data.

Formal verification: FormalStatus(NameCert$_{\mathsf{MoorePlane}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host topology, separation theorems, metrizability or nonmetrizability theorems, selected neighbourhood bases, quotient equality of points, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tangent-disc neighbourhood exactness, upper-half-plane routing, boundary-line nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MoorePlaneUp.

Chapter 7913

A Concrete Naming Certificate for $\mathsf{EventualEquality}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the cofinite-filter discipline familiar from Kelley-style topology, the $\mathsf{EventualEquality}^{\uparrow}$ packet records equality of two displayed streams after a finite tail cut. It reads finite observation windows through chapter 126, regular rational readback through chapter 301, the tail-comparison sibling chapter 1738, and the terminal real-name boundary of chapter 38. The packet names the cofinite-tail equality interface used by Real and Completion consumers; it does not form a quotient of streams, choose representatives, assert ambient completeness, or expose a hidden host equality.

The cofinite-tail equality packet therefore consumes the sibling route through chapter 126, chapter 301, chapter 1738, and chapter 38; those chapters supply the finite stream windows, regular rational readback, tail-comparison boundary, and terminal real-name seal that this packet replays.

Definition 7913.1 (Eventual equality carrier)**.** *An* $\mathsf{EventualEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, R, T, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two finite* $\mathsf{StreamName}^{\uparrow}$ *window surfaces being compared.*
2. *T is the finite tail-cut ledger naming the cofinite region after which comparison is admitted.*
3. *A is the displayed agreement row on the post-cut windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the compared windows and their tail-cut ledger.*
5. *H records componentwise* hsame *transport for the two windows, the tail cut, agreement, readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from the two windows and the tail cut to the agreement and readback rows.*
7. *P is the* Pkg *provenance over* $\mathsf{EventualEquality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{EventualEquality}^{\uparrow}}$ *row naming exactly this cofinite-tail equality packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EventualEqualityUp.TasteGate.EventualEqualityTasteGate single carrier alignment`

Theorem 7913.2 (Eventual equality NameCert obligations)**.** *For every accepted* $\mathsf{EventualEquality}^{\uparrow}$ *carrier*

$$E = (L, R, T, A, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EventualEquality}^{\uparrow}}$ *surface consists exactly of two stream-window admissions, one finite tail-cut ledger, one post-cut agreement row, regular rational readback, componentwise transport, displayed replay, provenance, and local naming. Every equality-facing read must pass through*

$$L, R, T, A, Q, H, C, P, N$$

and through no quotient stream equality, selected representative, hidden host equality, countable-choice selector, or ambient completeness row.

Proof. Project the carrier of Definition 7913.1. The rows L, R, T, A, Q expose the two finite windows, the cofinite tail cut, the agreement row, and regular rational readback before any equality-facing consumer can read the packet.

The structural rows H, C, P, N preserve and replay the same route. Componentwise transport cannot add a quotient coordinate, replay only follows the displayed windows and cut, provenance names the same dependencies, and the local naming row registers the same finite packet.

Thus the listed rows exhaust the NameCert surface. Since the carrier contains no quotient, selector, choice, host-equality, or ambient-completeness field, no downstream read can obtain such data from the $\mathsf{EventualEquality}^{\uparrow}$ packet. □

Theorem 7913.3 (Eventual equality RegSeqRat handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{EventualEquality}^{\uparrow}$ *packet receives the equality-facing route*

$$\mathsf{StreamName}^{\uparrow} \times \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{EventualEquality}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $L, R, T, A, Q, H, C, P, N$*. The handoff factors through the tail-comparison discipline of chapter 1738; it is not a quotient of stream names, a choice of representatives, an ambient completeness principle, or a host-level equality theorem.*

Proof. Begin with the two stream-window rows L and R. The tail-cut row T selects the finite boundary after which the displayed agreement row A is meaningful.

Next read the regular rational handoff Q. The handoff is downstream of the same finite windows and tail ledger, so it can be consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and then by a $\mathsf{Real}^{\uparrow}$-facing seal only through the displayed route.

Finally apply the structural rows H, C, P, N. They transport, replay, source, and locally name the same cofinite-tail equality packet. The sibling route through chapter 1738 supplies the tail-comparison boundary, while the carrier itself supplies no quotient, representative selector, completeness, or host-equality coordinate. □

Lean4 (statement) `BEDC.Derived.EventualEqualityUp`

Closure status for $\mathsf{NameCert}_{\mathsf{EventualEquality}^{\uparrow}}$**.**

Constructive story: An EventualEquality$^{\uparrow}$ packet is generated from two finite StreamName windows, one tail-cut ledger, one post-cut agreement row, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{EventualEquality}^{\uparrow}}$, seed)

Scope closed: Definition 7913.1, Definition 7913.2, and Definition 7913.3 seed the cofinite-tail equality route over stream windows, tail cuts, agreement rows, regular rational readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{EventualEquality}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.EventualEqualityUp`

Bridge status: none

Not claimed: The chapter does not claim quotient stream equality, selected representatives, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-cut visibility, post-cut agreement, regular rational handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.EventualEqualityUp.

Chapter 7914

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$ packet records the finite locator surface by which a regular Cauchy readback becomes positive apartness evidence for Real and Completion consumers. It sits after the positive apartness interface of chapter 7270, the Cauchy located-apartness boundary of chapter 3895, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, the finite stream-name window surface of chapter 126, and the terminal real-name seal of chapter 38. The packet names a local apartness locator for regular Cauchy data; it does not assert trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7914.1 (Regular Cauchy apartness locator carrier)**.** *A* $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, A, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to schedule the regular Cauchy observations.*
2. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row computed from the displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row fixing the positive gap scale used by the locator.*
4. *A is the* $\mathsf{RealApartnessLocator}^{\uparrow}$ *positive-apartness locator row exposed at that tolerance and window.*
5. *L is the* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *handoff row that records the Cauchy locator request and its located-apartness answer.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, A, L have been displayed.*
7. *H stores componentwise* hsame *transport for stream windows, regular readbacks, dyadic tolerance, apartness locator rows, Cauchy located-apartness handoff rows, Real seals, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from a regular Cauchy apartness request through $S, R, D, A, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$*,* $\mathsf{RealApartnessLocator}^{\uparrow}$*,* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessLocator}^{\uparrow}}$ *row.*
The carrier has no coordinate for trichotomy, quotient real equality, selected limits, countable choice, an independently chosen representative sequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7914.2 (Regular Cauchy apartness locator NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$ *carrier*

$$K = (S, R, D, A, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessLocator}^{\uparrow}}$ *surface consists exactly of paired stream-window exposure, paired regular-rational readback, dyadic-tolerance exactness, positive apartness-locator exposure, Cauchy located-apartness handoff, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7914.1. The analytic rows are S, R, D, A, L, E, and the structural rows are H, C, P, N.

The stream row S fixes the finite observations, the readback row R records the regular rational data read from those observations, and the dyadic row D fixes the positive tolerance at which apartness is inspected.

The apartness locator row A and the Cauchy located-apartness handoff row L are downstream of the same S, R, D route. The Real seal E is reached only after those rows are displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded principles is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7914.3 (Regular Cauchy apartness locator Real handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$ *packet reads the terminal seal only through the displayed route*

$$S, R, D, A, L, E, H, C, P, N.$$

The handoff therefore exposes positive apartness for regular Cauchy readbacks without exporting trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 7914.2. They require stream windows, regular readbacks, dyadic tolerance, apartness locator evidence, and the Cauchy located-apartness handoff to be displayed before the Real seal is visible.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. They do not add a total order branch, a quotient equality test, a selected limit, a choice schedule, or an ambient completeness theorem.

Thus every Real-facing or Completion-facing read factors through the finite carrier rows and no excluded principle can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyApartnessLocator}^{\uparrow}$ packet is generated from paired StreamName finite windows, paired RegSeqRat readbacks, a DyadicRatCore tolerance row, a RealApartnessLocator row, a CauchyLocatedApartness handoff row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessLocator}^{\uparrow}}$, seed)

Scope closed: Definition 7914.1, Definition 7914.2, and Definition 7914.3 seed the finite regular Cauchy apartness locator route over stream windows, regular-rational readbacks, dyadic tolerance, positive apartness locators, Cauchy located-apartness handoff, Real seals, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessLocator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular-rational read-back, dyadic-tolerance exactness, positive apartness-locator exposure, Cauchy located-apartness handoff, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyApartnessLocatorUp.

Chapter 7915

A Concrete Naming Certificate for $\mathsf{RegularCauchyMajorant}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMajorant}^{\uparrow}$ packet names the finite majorant certificate used when a regular Cauchy rational route needs a displayed bound before entering the L10 completion readback. It sits between the dyadic tolerance arithmetic of chapter 291, the finite observation schedules of chapter 126, the regular rational source of chapter 301, the explicit majorant siblings chapter 4215 and chapter 5900, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records a finite majorant row for regular Cauchy completion consumers; it does not import quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7915.1 (Regular Cauchy majorant carrier)**.** *A* $\mathsf{RegularCauchyMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, Q, J, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the inspected regular Cauchy requests;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the requested error bounds are checked;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational tail is read only through S and D;*
4. *J is the majorant row assigning a finite bound to each displayed tolerance window;*
5. *T is the tail-majorant comparison row, relating J to the sibling tail-majorant service of chapter 5900;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *completion-seal row available only after the regular tail and its majorant row have been displayed;*
7. *H records componentwise* hsame *transport for the schedule, tolerance, regular source, majorant, tail comparison, seal, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from finite windows through dyadic tolerances, regular readback, majorant comparison, and the Real-seal handoff;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMajorant}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, Q, J, T, E, H, C, P, N$ and their displayed transport and replay rows. It has no coordinate for a quotient-completion object, a selected limit, a hidden tail selector, countable choice, or ambient real completeness.

Theorem 7915.2 (Regular Cauchy majorant NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMajorant}^{\uparrow}$ *carrier*

$$M = (S, D, Q, J, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window exposure through S, dyadic tolerance exposure through D, regular rational source exposure through Q, majorant readability through J, tail-majorant comparison through T, Real-seal ordering through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7915.1. The displayed tuple is exactly $S, D, Q, J, T, E, H, C, P, N$, so every accepted packet exposes the finite windows, dyadic tolerances, regular rational source, majorant row, tail comparison, Real seal, and structural rows before a completion consumer can read it.

Read the analytic rows in order. The schedule row S names the inspected windows, the dyadic row D supplies the tolerance data, and the regular rational source row Q is read only through those finite rows.

The majorant rows then close the estimate route. The row J gives the finite bound, while T compares it to the sibling tail-majorant service; the seal row E is downstream of that displayed comparison and not an independent limit.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient completion, selected limits, hidden tail choice, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 7915.3 (Regular Cauchy majorant completion handoff)**.** *Every L10 completion consumer of an accepted* $\mathsf{RegularCauchyMajorant}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyMajorant}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports a displayed majorant certificate for regular Cauchy readback; it exports no quotient completion, selected limit, hidden tail selector, countable-choice scheduler, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Apply Definition 7915.2. The rows S, D, Q, J, T, E place the finite observation schedule, dyadic tolerance ledger, regular rational source, majorant row, tail comparison, and Real seal in one accepted packet.

The completion consumer may therefore read the route only after the majorant row has been displayed against the same finite windows and tolerances. The sibling services named above supply comparison surfaces, not a separate completion object.

Use the structural rows H, C, P, N to transport, replay, source, and name the same handoff. Because the packet lacks quotient completion, selected limits, hidden tail choice, countable choice, and ambient completeness coordinates, none of those readings can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMajorant}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyMajorant}^{\uparrow}$ packet is generated from a finite StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat source row, displayed majorant row, tail-majorant comparison, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyMajorant}\uparrow}$, seed)

Scope closed: Definition 7915.1, Definition 7915.2, and Definition 7915.3 seed the finite regular Cauchy majorant carrier and its L10 completion handoff.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyMajorant}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, hidden tail selection, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, tolerance exactness, regular readback, majorant comparison, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMajorantUp.

Chapter 7916

A Concrete Naming Certificate for PoincareDiskAreaElement$^{\uparrow}$

The PoincareDiskAreaElement$^{\uparrow}$ packet realizes the local area-read branch of chapter 9463 as finite disk, metric, radius, Real-seal, and observer-expansion certificate data. It consumes the disk carrier of chapter 377, the Poincare disk metric surface of chapter 6209, the hyperbolic metric row of chapter 3474, the observer-expansion packet of chapter 4167, the dyadic shadow cone of chapter 7503, and the dyadic and Real substrate of chapter 291 and chapter 38. The packet names a local area-element readback, not a global hyperbolic surface, quotient disk equality, selected coordinate atlas, physical horizon law, or ambient measure-completeness theorem.

Definition 7916.1 (Poincare disk area-element carrier)**.** *A* PoincareDiskAreaElement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, M, G, D, B, O, R, F, T, C, P, N)$$

with the following displayed rows:

1. *U is the* UnitDisk$^{\uparrow}$ *disk-coordinate row for the displayed local point or finite disk window.*
2. *M is the* PoincareDiskMetric$^{\uparrow}$ *row that reads the Poincare disk metric over the same coordinate data.*
3. *G is the* HyperbolicMetric$^{\uparrow}$ *row comparing the disk metric read with the hyperbolic metric surface consumed downstream.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *dyadic-radius row, recording the finite radius scale at which local area is inspected.*
5. *B is the area-budget row, binding the displayed radius and metric rows to a finite area-element budget.*
6. *O is the* HyperbolicObserverExpansion$^{\uparrow}$ *provenance row that records which observer-expansion branch receives the local area readback.*
7. *R is the terminal* Real$^{\uparrow}$ *seal row for the displayed area budget, available only after U, M, G, D, B, O are visible.*
8. *F is the refusal row excluding completed disk quotients, selected coordinate atlases, global area measures, physical horizon semantics, and ambient completeness.*

9. *T records componentwise* hsame *transport for disk coordinates, metric rows, dyadic radii, area budgets, observer-expansion provenance, Real seals, refusal data, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a local disk-area request through $U, M, G, D, B, O, R, F, T, P, N$.*
11. *P is the* Pkg *provenance over* PoincareDiskAreaElement$^{\uparrow}$, UnitDisk$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicShadowCone$^{\uparrow}$, HyperbolicObserverExpansion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{PoincareDiskAreaElement}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed Poincare disk quotient, selected global chart, ambient measure object, physical observer law, or hyperbolic completeness theorem.

Theorem 7916.2 (Poincare disk area-element NameCert obligations)**.** *For every accepted* PoincareDiskAreaElement *carrier $A = (U, M, G, D, B, O, R, F, T, C, P, N)$, the local* NameCert$_{\text{PoincareDiskAreaElement}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *disk-and-metric admission through U, M, G;*
2. *dyadic-radius and area-budget exactness through D, B;*
3. *observer-expansion provenance and Real-seal routing through O, R;*
4. *refusal and transport preservation through F, T;*
5. *displayed replay, provenance, and local naming through C, P, N.*

Proof. Project the carrier of Definition 7916.1. The disk and metric rows are U, M, G, the finite radius and area-budget rows are D, B, and the observer-facing and Real-facing rows are O, R.

The budget row B is available only after the disk coordinate, metric comparison, and dyadic-radius rows have been displayed. Thus an area-element read is local and finite before any Real seal can be consumed.

The rows F, T, C, P, N then refuse nonlocal endpoints, transport the same displayed data, replay the route, record provenance, and name the local packet. No additional carrier coordinate can supply a global measure, quotient disk equality, selected chart, physical horizon law, or ambient completeness theorem. □

Falsifiable prediction: Every accepted PoincareDiskAreaElement$^{\uparrow}$ read factors through UnitDisk, PoincareDiskMetric, HyperbolicMetric, DyadicRatCore, Real, observer-expansion provenance, transport, replay, package, and local NameCert rows before an area budget is visible.

Independence witness: unitdisk, poincare-disk-metric, hyperbolic-metric, hyperbolic-observer-expansion, dyadic-shadow-cone: no listed sibling is bijective to PoincareDiskAreaElement$^{\uparrow}$ because this packet jointly carries disk coordinates, metric comparison, dyadic radius, area budget, observer-expansion provenance, Real sealing, refusal, transport, replay, provenance, and local naming as one finite route.

The Lean TasteGate route must provide `Nontrivial PoincareDiskAreaElementUp` and `FieldFaithful PoincareDiskAreaElementUp` instances in `lean4/BEDC/Derived/PoincareDiskAreaElementUp/TasteGate.lean`. Those instances must keep $U, M, G, D, B, O, R, F, T, C, P, N$ as distinct fields before any checked marker is used.

Closure status for NameCert$_{\text{PoincareDiskAreaElement}^{\uparrow}}$**.**

Constructive story: A PoincareDiskAreaElement$^{\uparrow}$ packet is generated from UnitDisk coordinate rows, PoincareDiskMetric rows, HyperbolicMetric rows, DyadicRatCore radius rows, area-budget rows, HyperbolicObserverExpansion

provenance, a RealUp seal, refusal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PoincareDiskAreaElement}\uparrow}$, seed)

Scope closed: Definition 7916.1 and Definition 7916.2 seed the finite local disk-area route over disk coordinates, metric rows, dyadic radii, area budgets, observer-expansion provenance, Real seals, refusal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{PoincareDiskAreaElement}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed disk quotients, selected global charts, ambient measure objects, physical observer laws, hyperbolic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-and-metric admission, dyadic-radius exactness, area-budget exactness, observer-expansion provenance, Real-seal routing, refusal, transport, replay, provenance, local naming, TasteGate nontriviality, and field faithfulness for BEDC.Derived.PoincareDiskAreaElementUp.

Chapter 7917

A Concrete Naming Certificate for $\mathsf{PoincareMap}^{\uparrow}$

Following Poincare and the standard return-map construction in dynamical systems, the $\mathsf{PoincareMap}^{\uparrow}$ packet records the finite section-to-section return surface used by a displayed flow before any downstream consumer treats the read as a map. It sits after the finite routed-flow carrier of chapter 195, the displayed ODE route of chapter 193, the map-continuity packet of chapter 130, the metric-distance surface of chapter 57, the terminal Real seal of chapter 38, the finite observation windows of chapter 126, and the regular rational readback of chapter 301. The packet names a finite return certificate, not global recurrence, selected orbit representatives, quotient flow equality, host smooth transversality, or ambient Real completeness.

Definition 7917.1 (Poincare map carrier)**.** *A* $\mathsf{PoincareMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (D, O, M, S, T, W, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *routed-flow row exposing the finite transition relation inspected by the packet.*
2. *O is the* $\mathsf{Ode}^{\uparrow}$ *local evolution row used only through its displayed solution-step ledger.*
3. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the section-to-section map surface.*
4. *S is the section row naming the displayed source and target section windows.*
5. *T is the first accepted return-time row inside the inspected window.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row for source, flow, and return reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational return-time and section-coordinate data.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed return scale and coordinate reads.*
9. *H records componentwise* hsame *transport for the dynamic, ODE, map, section, return-time, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay through the route*

$$D, O, M, S, T, W, R \longmapsto E.$$

11. *Q is the* Pkg *provenance over* $\mathsf{PoincareMap}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, O, M, S, T, W, R, E, H, C, Q, N$. It has no coordinate for a global recurrence theorem, selected complete orbit, quotient flow equality, host transversality theorem, ambient completeness principle, Lean verification, or bridge checking.

Theorem 7917.2 (Poincare map NameCert obligations)**.** *For every accepted* $\mathsf{PoincareMap}^{\uparrow}$ *carrier*

$$P = (D, O, M, S, T, W, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareMap}^{\uparrow}}$ *surface consists exactly of routed dynamics through D, displayed ODE evolution through O, section-to-section map exposure through M, section-window admission through S, return-time admission through T, finite observation through W, regular rational readback through R, Real-seal handoff through E, and structural transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the carrier of Definition 7917.1. The rows D, O, M, S expose the only dynamic, evolution, map, and section data visible to the packet before a return is read.

The return data are then forced through T, W, R, E. The accepted return time is a displayed finite row, the observation window is a $\mathsf{StreamName}^{\uparrow}$ row, the coordinate readback is a $\mathsf{RegSeqRat}^{\uparrow}$ row, and the scalar boundary is sealed only through $\mathsf{Real}^{\uparrow}$.

Finally attach H, C, Q, N. Componentwise transport, continuation replay, package provenance, and local naming preserve the same finite packet and add no selected orbit, quotient flow equality, host transversality theorem, or ambient completeness principle. □

Theorem 7917.3 (Poincare map return nonescape)**.** *Every accepted* $\mathsf{PoincareMap}^{\uparrow}$ *return read remains inside the finite carrier of Definition 7917.1. In particular, a consumer can receive a section-to-section return map only after displaying the* $\mathsf{DynSystem}^{\uparrow}$ *route,* $\mathsf{Ode}^{\uparrow}$ *evolution ledger,* $\mathsf{ContinuousMap}^{\uparrow}$ *map row, section windows, finite return-time row,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. No accepted read may replace this route by global recurrence, a selected orbit representative, quotient flow equality, a host smooth-transversality theorem, or ambient Real completeness.*

Proof. Let a consumer request a Poincare return map from an accepted carrier. Projection of Definition 7917.1 shows that the request first meets the displayed dynamic route D, ODE row O, map row M, and section row S.

The return-time row T is downstream from the displayed section data, and the observation row W fixes the finite source, flow, and target windows. The regular rational readback R and Real seal E then expose only the registered return scale and section-coordinate reads.

The structural rows H, C, Q, N do not add analytic primitives. They transport equal displayed histories, replay the finite route, record package provenance, and attach the local name. Therefore every accepted return read stays inside the finite Poincare map certificate surface and cannot escape to the excluded host principles. □

Falsifiable prediction: An accepted $\mathsf{PoincareMap}^{\uparrow}$ consumer that reads a section-to-section return without displaying the DynSystem route, ODE evolution row, ContinuousMap surface, section

window, finite return-time row, StreamName observations, RegSeqRat readback, and Real seal refutes the carrier surface.

Independence witness: dynsystem, ode, continuousmap, metric, real, streamname, regseqrat: no listed sibling supplies the whole packet $P = (D, O, M, S, T, W, R, E, H, C, Q, N)$, because each sibling contributes only a routed flow, evolution row, map surface, distance boundary, scalar seal, finite observation window, or rational readback while this chapter binds them into one return-map NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareMap}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareMap}^{\uparrow}$ packet is generated from a finite DynSystem route, ODE evolution row, ContinuousMap section-to-section surface, section windows, finite return-time row, StreamName observations, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7917.1, Definition 7917.2, and Definition 7917.3 seed the finite Poincare return-map route through dynamics, ODE evolution, continuous-map exposure, section windows, return-time admission, finite observations, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global recurrence, selected orbit representatives, quotient flow equality, host smooth transversality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dynamic routing, ODE evolution exposure, continuous-map exposure, section-window admission, return-time admission, observation-window exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.PoincareMapUp.

Chapter 7918

A Concrete Naming Certificate for $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$ packet records the finite upper one-sided Dini derivative envelope through which displayed function windows expose upper secant comparison rows before any semicontinuity, regular-readback, or real-name consumer can inspect them. It sits beside the one-sided Dini carrier of chapter 2041, the lower envelope sibling of chapter 7911, the regular Cauchy majorant of chapter 7915, the finite stream-name schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only finite upper-envelope windows and readback rows; it does not claim classical limsup existence, selected derivative values, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7918.1 (UpperDiniDerivativeEnvelope carrier)**.** *An* $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, UQ, R, S, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed real-valued function row, read through the same finite observation surface used by the sibling* $\mathsf{DiniDerivative}^{\uparrow}$ *and* $\mathsf{UpperDiniDerivative}^{\uparrow}$ *packets.*
2. *W is the upper-envelope window row, listing the finite one-sided neighborhoods and upper secant comparisons available to the packet.*
3. *UQ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for radii, slope upper bounds, and comparison slack.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint values, secant numerators, step sizes, and dyadic upper-bound approximants.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that records which finite function and base-point observations are available before a readback is public.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, W, UQ, R, S have been displayed.*
7. *H records componentwise* hsame *transport for the function, upper-envelope window, dyadic, readback, stream, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for refinements of the same finite upper-envelope window.*

9. *P is the* Pkg *provenance over* $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$, $\mathsf{DiniDerivative}^{\uparrow}$, $\mathsf{DiniDerivativeLowerEnvelope}^{\uparrow}$, $\mathsf{UpperDiniDerivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, W, UQ, R, S, E, H, C, P, N$. It has no coordinate for a lower-envelope row, two-sided derivative, selected limiting slope, classical limsup object, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7918.2 (UpperDiniDerivativeEnvelope NameCert obligations)**.** *For every accepted* UpperDiniDerivativeE *carrier*

$$U = (F, W, UQ, R, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed function row, upper-envelope windows, DyadicRatCore tolerance row, RegSeqRat readback, StreamName schedule, RealUp seal, componentwise transport, continuation replay, package provenance, and local naming; upper-envelope exactness requiring every upper-bound read to pass through W, UQ, R, S; sibling compatibility requiring the Dini and lower-envelope packets to remain cited only as comparison surfaces; and nonescape from lower-envelope export, two-sided derivative data, selected limiting slopes, classical limsup objects, quotient real equality, countable choice, and ambient Real completeness.*

Proof. Project the carrier of Definition 7918.1. The accepted packet exposes only the displayed function row F, upper-envelope windows W, dyadic tolerance row UQ, regular readback row R, stream schedule S, Real seal E, and structural rows H, C, P, N.

Upper-envelope exactness is finite. An upper-bound request must first read the inspected window in W, then the dyadic tolerance and slack data in UQ, then the regular rational readback in R, and then the scheduled observation row in S before the Real seal can be consumed.

The Dini and lower-envelope siblings are comparison anchors, not exported coordinates. They identify the surrounding one-sided derivative surface while the present packet keeps only the upper-envelope route visible.

Since no carrier coordinate stores a lower-envelope export, two-sided derivative, selected limiting slope, classical limsup object, quotient equality, choice schedule, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$ consumer reaching an upper-envelope read without the displayed function row, upper-envelope windows, DyadicRatCore tolerance row, RegSeqRat readback, StreamName schedule, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming refutes the carrier.

Independence witness: dini-derivative, dini-derivative-lower-envelope, upper-dini-derivative, regseqrat, streamname, real: no listed sibling supplies the whole upper-envelope packet $U = (F, W, UQ, R, S, E, H, C, P, N)$, because each sibling contributes only one-sided Dini windows, lower rows, upper derivative rows, rational readback, observation schedules, or a terminal real seal while this chapter binds the upper-envelope route in one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}}$ **·**

Constructive story: An $\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}$ packet is generated from displayed function data, upper-envelope windows, DyadicRatCore tolerance rows, RegSeqRat readback, StreamName schedules, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UpperDiniDerivativeEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 7918.1 and Definition 7918.2 seed the finite upper-envelope packet from Dini derivative windows to regular-readback and real-name consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{UpperDiniDerivativeEnvelope}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim classical limsup existence, lower-envelope export, two-sided derivative existence, selected derivative values, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-envelope exactness, sibling compatibility, DyadicRatCore tolerance exposure, RegSeqRat readback, StreamName scheduling, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UpperDiniDerivativeEnvelopeUp.

Chapter 7919

A Concrete Naming Certificate for $\mathsf{DyadicMeshLocator}^{\uparrow}$

Following Bishop and Bridges on constructive real approximation and finite dyadic partitions, the $\mathsf{DyadicMeshLocator}^{\uparrow}$ packet records the finite cell-location read used when a regular real window is compared with a displayed dyadic mesh. It sits over the mesh rows of chapter 7835, the rounded locator route of chapter 1972, the dyadic tolerance ledger of chapter 291, the finite stream window of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a finite locator primitive; it does not decide host trichotomy, select a nearest cell, quotient real names, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7919.1 (Dyadic mesh locator carrier)**.** *A* $\mathsf{DyadicMeshLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, Q, S, E, T, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *row naming the finite mesh and its ordered cell rows.*
2. *Q is the locator request row containing the requested precision and the inspected real-window address.*
3. *S is the selected mesh-cell row returned by the finite locator.*
4. *E is the endpoint-tolerance row proving that the selected cell endpoints were read at the requested dyadic precision.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the endpoints and the window.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the locator.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to W.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, Q, S, E, T, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the mesh, request, selected cell, endpoint tolerances, stream window, regular readback, real seal, and structural rows.*
10. *C records displayed* Cont *replay from a locator request through the mesh, selected cell, endpoint tolerances, stream window, regular readback, and real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshLocator}^{\uparrow}$*,* $\mathsf{FiniteDyadicMesh}^{\uparrow}$*,* $\mathsf{RoundedDyadicLocator}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{DyadicMeshLocator}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, Q, S, E, T, W, R, A, H, C, P, N$. It has no coordinate for host trichotomy, a nearest-cell selector, quotient real equality, countable choice, or ambient real completeness.

Theorem 7919.2 (Dyadic mesh locator NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshLocator}^{\uparrow}$ *carrier $L = (M, Q, S, E, T, W, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshLocator}^{\uparrow}}$ *surface consists exactly of finite mesh admission, locator-request admission, selected-cell row exposure, endpoint-tolerance accounting, dyadic tolerance comparison, StreamName window read, RegSeqRat readback, Real seal handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7919.1. The visible inventory is $M, Q, S, E, T, W, R, A, H, C, P, N$, and every locator obligation is assigned to one of those rows.

The finite mesh and request rows are read first. Row M supplies the inspected dyadic mesh, while Q records the precision and window address that the locator is allowed to use. The selected cell row S is therefore a finite row of the same mesh rather than a host-side nearest-cell choice.

The endpoint and tolerance data are then displayed. Row E records the endpoint tolerances, and row T supplies the dyadic rational comparison budget used by the locator. Rows W and R expose the stream window and regular rational readback that the real-facing request actually inspects.

Finally A, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier contains no trichotomy decision, nearest-cell selector, quotient real equality, countable-choice row, or ambient-completeness witness, those objects are not local obligations of the packet. □

Theorem 7919.3 (Dyadic mesh locator finite cell handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicMeshLocator}^{\uparrow}$ *carrier receives the finite route*

$$M \longrightarrow Q \longrightarrow S \longrightarrow E \longrightarrow T \longrightarrow W \longrightarrow R \longrightarrow A \longrightarrow H, C, P, N.$$

Thus the consumer reads a displayed finite dyadic mesh, locator request, selected cell, endpoint tolerance ledger, dyadic rational tolerance row, StreamName window, and RegSeqRat readback before it may consume the terminal real seal.

Proof. Start from the carrier definition. The terminal real seal is A, and it appears only after the finite mesh, request, selected cell, endpoint tolerance, dyadic tolerance, stream window, and regular readback rows have been displayed.

Apply Definition 7919.2. It makes the route through M, Q, S, E, T, W, R part of the local NameCert surface, so a consumer cannot replace the route by trichotomy, a nearest-cell selector, quotient equality of real names, or a choice-based search over mesh cells.

The transport and replay rows preserve the same finite order for downstream consumers, while P and N record provenance and local naming. The handoff is therefore the displayed mesh-cell route rather than an ambient completeness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshLocator}^{\uparrow}$ packet is generated from a finite dyadic mesh row, locator request row, selected cell row, endpoint tolerance rows, DyadicRatCore tolerance row, StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicMeshLocator}^{\uparrow}}$, seed)

Scope closed: Definition 7919.1, Definition 7919.2, and Definition 7919.3 seed the finite dyadic mesh cell-location packet for Real and Completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicMeshLocator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host trichotomy, selected nearest cells, quotient real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh-row exactness, request exactness, selected-cell exposure, endpoint-tolerance accounting, StreamName and RegSeqRat handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMeshLocatorUp.

Chapter 7920

A Concrete Naming Certificate for $\mathsf{VerificationCoverageLedger}^{\uparrow}$

The $\mathsf{VerificationCoverageLedger}^{\uparrow}$ packet realizes certified-interpretability coverage as finite BEDC data: a component is counted as verified only through a displayed claim row, transcription row, backend proof row, coverage row, residue row, explanation scope, transport, replay, provenance, and local naming. It refines the formal-verification vision of Definition 9462.455 by giving the coverage accounting its own NameCert surface. The packet consumes classifier and explanation siblings such as chapter 6202, chapter 6138, chapter 7533, and chapter 6249; it does not certify the hidden model, replace the explanation ledger, prove classifier novelty, or turn proof coverage into semantic interpretability.

Definition 7920.1 (Verification coverage ledger carrier)**.** *A* $\mathsf{VerificationCoverageLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (K, T, B, C, R, E, F, H, D, P, N)$$

with the following displayed rows:

1. *K is the verified-component key naming the finite interpretability component under audit.*
2. *T is the transcription ledger row that states which BEDC claim was submitted to a proof backend.*
3. *B is the backend proof row naming the verifier transcript and proof object boundary.*
4. *C is the coverage row recording which component clauses are covered by T and B.*
5. *R is the residue row listing component clauses not covered by the verification transcript.*
6. *E is the scoped explanation row consumed from* $\mathsf{ScopedExplanationLedger}^{\uparrow}$*.*
7. *F is the classifier-facing row consumed from the positive-interpretability, novelty-witness, and black-box-fiber sibling packets.*
8. *H records componentwise* hsame *transport for the component key, transcription, backend proof, coverage, residue, explanation, classifier-facing, and structural rows.*
9. *D records displayed* Cont *replay from the component key through transcription, proof, coverage, residue, explanation, and classifier-facing rows.*
10. *P is the* Pkg *provenance over* $\mathsf{VerificationCoverageLedger}^{\uparrow}$*,* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$*,* $\mathsf{ClassifierNoveltyWitn}$ $\mathsf{BlackBoxFiberWitness}^{\uparrow}$*,* $\mathsf{ScopedExplanationLedger}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{VerificationCoverageLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, T, B, C, R, E, F, H, D, P, N$. It has no coordinate for hidden backend trust, whole-model transparency, proof coverage outside the transcription ledger, unscoped explanation transfer, or discovery status without a classifier-novelty witness.

Theorem 7920.2 (Verification coverage ledger NameCert obligations)**.** *For every accepted* VerificationCoverageLedg *carrier $V = (K, T, B, C, R, E, F, H, D, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VerificationCoverageLedger}^{\uparrow}}$ *surface consists exactly of verified-component admission, transcription-ledger admission, backend proof-boundary exposure, coverage-row accounting, residue-row accounting, scoped-explanation handoff, classifier-facing handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7920.1. The visible verification inventory is $K, T, B, C, R, E, F, H, D, P, N$, so each accepted coverage claim must name the component, transcription, backend proof boundary, coverage row, and residue row.

The transcription row is the only path from a component claim to backend verification. Thus the backend proof row B cannot certify a claim not displayed by T, and the coverage row C cannot silently cover a clause listed in the residue row R.

The explanation and classifier-facing rows keep interpretability separate from proof accounting. Row E scopes the explanation ledger, while row F records the classifier and novelty surfaces that remain external obligations.

Finally H, D, P, N transport, replay, source, and name the same route. Since the carrier has no hidden backend-trust coordinate, whole-model transparency coordinate, unscoped explanation transfer, or discovery-status coordinate, those objects are not local obligations of the packet. □

Theorem 7920.3 (Verification coverage residue handoff)**.** *Every certified-interpretability consumer of an accepted* $\mathsf{VerificationCoverageLedger}^{\uparrow}$ *packet receives the route*

$$K \longrightarrow T \longrightarrow B \longrightarrow C, R \longrightarrow E, F \longrightarrow H, D, P, N.$$

Thus verification coverage is read together with its residue ledger before it can be consumed by scoped explanation or classifier-facing rows.

Proof. Start from Definition 7920.2. It requires the component key, transcription ledger, backend proof boundary, coverage row, and residue row before any consumer reads the packet.

The coverage and residue rows are paired. A covered clause must pass through C, while any uncovered clause remains visible in R and cannot be erased by the backend proof transcript.

The explanation and classifier-facing rows are downstream of the coverage ledger. They may consume the certified component only with its displayed residue, and they do not create novelty or whole-model transparency by themselves.

The transport, replay, provenance, and naming rows preserve the same route. The packet therefore exposes a falsifiable coverage surface rather than a hidden proof-quality score. □

Falsifiable prediction: Every accepted $\mathsf{VerificationCoverageLedger}^{\uparrow}$ read exposes a component key, transcription ledger, backend proof boundary, coverage row, residue row, scoped explanation row, classifier-facing row, transport, replay, provenance, and local naming before claiming certified interpretability coverage.

Independence witness: positive-interpretability-classifier, classifier-novelty-witness, black-box-fiber-witness, scoped-explanation-ledger, namecert: $\mathsf{VerificationCoverageLedger}^{\uparrow}$ is not bijective to

any listed sibling because it joins backend proof coverage with explicit residue accounting while leaving classifier novelty, black-box fiber exposure, scoped explanation, and generic NameCert obligations as separately named rows.

Future TasteGate work for $\mathsf{VerificationCoverageLedger}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ VerificationCoverageLedge}$ and $\mathsf{FieldFaithful\ VerificationCoverageLedger}^{\uparrow}$ instances. Nontriviality is witnessed by a packet with a nonempty residue row and the same component key, transcription, proof boundary, explanation, classifier-facing, transport, replay, provenance, and naming rows; field faithfulness requires every accepted read to preserve $K, T, B, C, R, E, F, H, D, P, N$ as displayed.

Closure status for $\mathsf{NameCert}_{\mathsf{VerificationCoverageLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{VerificationCoverageLedger}^{\uparrow}$ packet is generated from verified-component keys, transcription ledgers, backend proof-boundary rows, coverage rows, residue rows, scoped explanation rows, classifier-facing rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VerificationCoverageLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7920.1, Definition 7920.2, Definition 7920.3, the falsifiable prediction, the independence witness, and the future TasteGate obligations seed the certified-interpretability verification coverage packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VerificationCoverageLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, hidden backend trust, whole-model transparency, classifier novelty, black-box fiber exposure, unscoped explanation transfer, or discovery status.

Upgrade path: Obligation closure requires checked carrier admission, transcription exactness, backend proof-boundary exposure, coverage and residue accounting, scoped-explanation handoff, classifier-facing handoff, transport stability, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.VerificationCoverageLedgerUp.

Chapter 7921

A Concrete Naming Certificate for $\mathsf{SchurProperty}^{\uparrow}$

Following Schur's classical sequence-space property for ℓ^1, the $\mathsf{SchurProperty}^{\uparrow}$ packet records the finite BEDC surface on which weakly convergent sequence data can be read as norm-convergent sequence data. It sits after the norm carrier of chapter 89, the vector-space rows of chapter 47, the linear-functional interface of chapter 48, and the metric readout of chapter 57. The packet names a displayed weak-to-norm sequence route; it is not a Banach-space classification theorem, a duality representation theorem, a selected subsequence principle, or an ambient completeness claim.

Definition 7921.1 (Schur property carrier)**.** *A* $\mathsf{SchurProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, N, L, W, M, H, C, P, A)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming the finite vector expressions and scalar-action windows inspected by the packet.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row naming the norm observations on the same displayed sequence windows.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the finite family of linear-functional reads used to expose weak convergence.*
4. *W is the weak-sequence ledger, recording the displayed functional tests and their finite tail bounds.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *or norm-induced distance row that reads the displayed norm tails as a metric convergence claim.*
6. *H records componentwise* hsame *transport for V, N, L, W, M and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from vector and functional windows through weak-tail checks into the norm-distance readout.*
8. *P is the* Pkg *provenance over* $\mathsf{SchurProperty}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *A is the local* $\mathsf{NameCert}_{\mathsf{SchurProperty}^{\uparrow}}$ *row naming exactly this finite weak-to-norm packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Banach-space theorem, dual-space representation, selected subsequence, quotient equality of sequences, countable choice, or ambient completeness of the normed space.

Theorem 7921.2 (Schur property NameCert obligations)**.** *For every accepted* $\mathsf{SchurProperty}^{\uparrow}$ *carrier*

$$S = (V, N, L, W, M, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{SchurProperty}^{\uparrow}}$ *surface consists exactly of vector-space exposure through* V*, norm observation through* N*, finite linear-functional tests through* L*, weak-tail bookkeeping through* W*, norm-distance readout through* M*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* A*.*

Proof. Project the carrier of Definition 7921.1. The rows V, N, L, W, M are the only mathematical rows visible before a sequence consumer can read weak convergence or norm convergence.

The structural rows H, C, P, A preserve that same route. Transport is componentwise, replay follows the displayed functional tests and norm-distance readout, provenance names the same dependency packet, and local naming registers no extra analytic object.

Thus the listed rows exhaust the NameCert surface. Since the carrier contains no host Banach-space theorem, dual representation, selected subsequence, quotient sequence equality, choice schedule, or ambient completeness field, no downstream consumer can obtain such data from the $\mathsf{SchurProperty}^{\uparrow}$ packet. □

Theorem 7921.3 (Schur property sequence norm witness)**.** *Every accepted* $\mathsf{SchurProperty}^{\uparrow}$ *read that turns displayed weak sequence convergence into displayed norm convergence must expose the route*

$$\mathsf{VecSpace}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{Met}^{\uparrow}, H, C, P, A$$

through the carrier rows $V, N, L, W, M, H, C, P, A$*. The norm-convergence witness is a finite tail-read over the displayed functional tests and norm-distance ledger, not a selected subsequence argument, a dual-space completeness theorem, a quotient equality of sequences, or an ambient Banach-space completeness claim.*

Proof. Start from the accepted carrier. The weak side of the read is visible only through the vector-space row V, the linear-functional row L, and the weak-tail ledger W.

The norm side is then read through N and M. These rows record the finite norm observations and the induced metric-distance tails that a consumer may inspect.

Finally attach H, C, P, A. They transport, replay, source, and locally name the same finite packet, so the witness remains a displayed sequence-norm route. Because no selected subsequence, quotient sequence equality, dual completeness, or ambient Banach completeness row appears in the carrier, the route exports none of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurProperty}^{\uparrow}$ packet is generated from VecSpaceUp sequence windows, NormUp observations, LinearMapUp functional tests, weak-tail bookkeeping, MetricUp distance readout, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurProperty}^{\uparrow}}$, seed)

Scope closed: Definition 7921.1, Definition 7921.2, and Definition 7921.3 seed the finite weak-to-norm sequence route over $\mathsf{Norm}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a general Banach-space classification theorem, dual-space representation, selected subsequences, quotient sequence equality, countable choice, ambient completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-tail functional test exposure, norm-distance readout, sequence-witness routing, transport, replay, provenance, and local naming for BEDC.Derived.SchurPropertyUp.

Chapter 7922

A Concrete Naming Certificate for $\mathsf{EventFlowRecognizer}^{\uparrow}$

The $\mathsf{EventFlowRecognizer}^{\uparrow}$ packet realizes the inscription-event carrier of chapter 9397 as finite recognizer data. It sits after the inscription-event surface of chapter 519, the named route packet of chapter 582, the recognizer-flow dependency of chapter 1048, and the GAP-ledger boundary of chapter 520. The packet names event recognition as visible source, route, role, ledger, transport, replay, provenance, and local naming rows; it does not add private intention, hidden generator state, evaluator equality, or unbounded kernel authority.

Definition 7922.1 (Event-flow recognizer carrier)**.** *An* $\mathsf{EventFlowRecognizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, I, G, L, F, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{InscriptionEvent}^{\uparrow}$ *source row naming the accepted event transition and its public record boundary.*
2. *I is the* $\mathsf{InscribedRoute}^{\uparrow}$ *row naming the finite route from visible gap through accepted record and downstream consumer ledger.*
3. *G is the* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *row naming the recognizer flow that checks the accepted route against the compiler-facing record.*
4. *L is the* $\mathsf{InscriptionGapLedger}^{\uparrow}$ *row recording the gap residue, accountability, and non-escape boundary for the same event.*
5. *F is the recognizer-role row, naming which finite consumer role may read E, I, G, L as one event-flow packet.*
6. *H records componentwise* hsame *transport for source event, route, recognizer flow, gap ledger, role, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through the route*

$$E, I, G, L, F \longmapsto N.$$

8. *P is the* Pkg *provenance over* $\mathsf{EventFlowRecognizer}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscribedRoute}^{\uparrow}$, $\mathsf{GroundCompilerRecognize}$ $\mathsf{InscriptionGapLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{EventFlowRecognizer}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by these rows. It has no coordinate for private agency, hidden generator choice, evaluator equality, quotient equality of routes, global kernel consistency, or an unbounded recognizer oracle.

Theorem 7922.2 (Event-flow recognizer NameCert obligations)**.** *For every accepted* $\mathsf{EventFlowRecognizer}^{\uparrow}$ *carrier*

$$R = (E, I, G, L, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EventFlowRecognizer}^{\uparrow}}$ *surface consists exactly of inscription-event exposure through E, inscribed-route exposure through I, ground-compiler recognizer flow through G, GAP-ledger accountability through L, recognizer role through F, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 7922.1. The rows E, I, G, L, F are the only event, route, recognizer-flow, gap-ledger, and consumer-role data visible before an event-flow consumer reads the packet.

The structural rows H, C, P, N preserve this route. Transport is componentwise, replay follows the displayed recognizer flow, provenance names the same source packets, and local naming registers the finite event-flow recognizer.

Thus the listed rows exhaust the NameCert surface. Since no private intention, hidden generator choice, evaluator equality, quotient route equality, global consistency theorem, or unbounded recognizer oracle appears in the carrier, no accepted consumer can read such data from the packet. □

Theorem 7922.3 (Event-flow recognizer ledger nonescape)**.** *Every* $\mathsf{EventFlowRecognizer}^{\uparrow}$ *consumer reading an inscription event as a recognizer flow must expose*

$$E, I, G, L, F, H, C, P, N$$

before it may use the recognized event. The GAP-ledger row L remains visible beside the route and recognizer-flow rows, so the read is a finite inscription-event recognition packet, not a hidden agency explanation, an evaluator-equality theorem, a quotient route equality, global kernel consistency, or a private generator-state claim.

Proof. Start with the event and route rows E and I. They display the accepted inscription transition and the finite route that carries it to a public record.

Next read the recognizer-flow and ledger rows G and L. These keep the ground-compiler-facing recognizer route and the gap-accountability ledger visible before the consumer role F can treat the transition as recognized.

Finally attach H, C, P, N. They transport, replay, source, and locally name the same finite packet. Because every consumed row is displayed, the route cannot escape to hidden agency, evaluator equality, quotient route equality, global kernel consistency, or private generator state. □

Falsifiable prediction: An accepted $\mathsf{EventFlowRecognizer}^{\uparrow}$ read is refuted if a consumer recognizes an inscription event without displaying the inscription-event source row, inscribed-route row, ground-compiler recognizer-flow row, GAP-ledger row, recognizer-role row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: inscriptionevent, inscribedroute, groundcompilerrecognizerflow, inscriptiongap-ledger: $\mathsf{EventFlowRecognizer}^{\uparrow}$ is not bijective to any listed sibling because it binds event source,

accepted route, recognizer flow, gap accountability, consumer role, transport, replay, provenance, and local naming into one finite NameCert packet.

Later Lean TasteGate work supplies the nontriviality and field-faithfulness instances for the recognizer classifier. This chapter records only the concrete packet surface that those instances must inspect: the finite event, route, recognizer-flow, ledger, role, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EventFlowRecognizer}^{\uparrow}}$.

Constructive story: An $\mathsf{EventFlowRecognizer}^{\uparrow}$ packet is generated from an InscriptionEventUp source row, an InscribedRouteUp route row, a GroundCompilerRecognizerFlowUp recognizer row, an InscriptionGapLedgerUp accountability row, a recognizer-role row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EventFlowRecognizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7922.1, Definition 7922.2, and Definition 7922.3 seed the finite inscription-event recognizer route over source event, route, recognizer-flow, GAP-ledger, role, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EventFlowRecognizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim private intention, hidden generator choice, evaluator equality, quotient route equality, global kernel consistency, unbounded recognizer authority, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recognizer-role exactness, GAP-ledger nonescape, route compatibility, transport, replay, provenance, local naming, and TasteGate nontriviality and field-faithfulness instances for BEDC.Derived.EventFlowRecognizerUp.

Chapter 7923

A Concrete Naming Certificate for $\mathsf{FiniteWalshTransform}^{\uparrow}$

Following Walsh and Hadamard finite transform theory, the $\mathsf{FiniteWalshTransform}^{\uparrow}$ packet records a dyadic binary-character transform over finite data without importing host Fourier analysis, ambient Hilbert-space completion, or an external orthogonality theorem. It sits next to the finite observation surface of chapter 124, the butterfly schedule of chapter 226, the dyadic arithmetic ledger of chapter 291, and the finite stream-window discipline of chapter 126.

Definition 7923.1 (Finite Walsh transform carrier)**.** *A* $\mathsf{FiniteWalshTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, X, A, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-window row, listing the finite input coordinates and retained output coordinates.*
2. *B is the binary-character row, recording the displayed bitwise parity probes used to evaluate Walsh signs.*
3. *X is the signed accumulation row, containing the finite plus-minus summation ledger attached to the retained coordinates.*
4. *A is the Hadamard-stage row, naming the finite row/column stage data used by the transform read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row for normalized sums, sign changes, and dyadic scaling.*
6. *R is the readback row consumed by finite harmonic-analysis neighbours after S, B, X, A, D have been displayed.*
7. *H records componentwise* hsame *transport for the displayed source windows, binary characters, accumulation rows, stage rows, arithmetic rows, readback rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite coordinate request through S, B, X, A, D, R.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteWalshTransform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Fft}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWalshTransform}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, B, X, A, D, R, H, C, P, N$. It has no coordinate for host Fourier inversion, completed orthogonal bases, analytic convergence, quotient equality of functions, selected infinite spectra, or floating-point error analysis.

Theorem 7923.2 (Finite Walsh transform NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWalshTransform}^{\uparrow}$ *carrier $W = (S, B, X, A, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteWalshTransform}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source-window, binary-character, signed-accumulation, Hadamard-stage, dyadic-arithmetic, readback, transport, replay, provenance, and local naming rows; binary-character exactness through B; finite accumulation exactness through X; stage accountability through A; dyadic scaling through D; and public readback through R, H, C, P, N.*

Proof. Project the carrier of Definition 7923.1. Its visible inventory is exactly $S, B, X, A, D, R, H, C, P, N$, so every local naming duty begins with finite source and output windows rather than with a host transform.

The transform-specific rows are then read in order. The binary-character row B supplies the parity probes, the signed accumulation row X records the finite plus-minus ledger, and the Hadamard-stage row A names the row/column stage data used by the same finite request.

The arithmetic endpoint is also explicit. The dyadic row D accounts for scaling and sign changes before the readback row R can be consumed by Fourier or FFT-facing neighbours.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores inversion, completed orthogonality, convergence closure, quotient function equality, selected spectra, or floating-point analysis, the listed rows exhaust the local NameCert obligations. □

Theorem 7923.3 (Finite Walsh transform binary-character handoff)**.** *Every finite harmonic-analysis consumer of an accepted* $\mathsf{FiniteWalshTransform}^{\uparrow}$ *packet receives the Walsh read only through*

$$S, B \longmapsto X, A \longmapsto D, R \longmapsto H, C, P, N.$$

The handoff may be compared with the Fourier observation surface of chapter 124 and the butterfly schedule of chapter 226, but it exports only finite binary-character rows, finite signed sums, Hadamard-stage data, dyadic scaling, and displayed readback.

Proof. Start with the rows S and B. The source-window row fixes the finite coordinates under inspection, while the binary-character row fixes the parity probes that determine the Walsh signs.

Next read X and A. The signed accumulation row records the finite sum attached to the retained coordinates, and the Hadamard-stage row records the finite stage data used by the same transform request.

Only after those rows are displayed can the dyadic scaling and readback rows D, R be consumed. Thus a Fourier or FFT-facing consumer receives a finite binary-character handoff rather than an ambient Fourier transform, completed basis theorem, inversion theorem, or analytic convergence claim.

The structural rows H, C, P, N preserve this factorization under transport, replay, provenance, and local naming. The packet therefore supplies a finite Walsh neighbour for harmonic-analysis consumers without adding host analytic structure. □

Falsifiable prediction: Every accepted $\mathsf{FiniteWalshTransform}^{\uparrow}$ read factors through source windows, binary-character parity probes, signed accumulation rows, Hadamard-stage rows, dyadic scaling, readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any finite harmonic-analysis consumer may cite it.

Independence witness: streamname, dyadicratcore, fourier, fft: $\mathsf{FiniteWalshTransform}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite source windows, binary-character parity probes, signed plus-minus accumulation, Hadamard-stage data, dyadic scaling, readback, transport, replay, provenance, and local NameCert rows as one finite transform packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWalshTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWalshTransform}^{\uparrow}$ packet is generated from finite source windows, binary-character parity probes, signed accumulation rows, Hadamard-stage data, dyadic scaling, readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWalshTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7923.1, Definition 7923.2, and Definition 7923.3 seed the finite binary-character transform route over StreamName, DyadicRatCore, Fourier, and FFT-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWalshTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Fourier inversion, completed orthogonality, analytic convergence, quotient equality of functions, selected infinite spectra, floating-point error analysis, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, binary-character exactness, finite accumulation exactness, Hadamard-stage accountability, dyadic scaling, readback, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWalshTransformUp.

Chapter 7924

A Concrete Naming Certificate for $\mathsf{OsgoodCurve}^{\uparrow}$

Following Osgood's positive-area curve construction, the $\mathsf{OsgoodCurve}^{\uparrow}$ packet records a finite Real/Completion-facing certificate for a continuous curve whose image carries a displayed positive-area witness. It consumes the terminal real-name boundary of chapter 38, the regular rational readback of chapter 301, the finite observation schedule of chapter 126, the continuous-map carrier of chapter 130, and the compact metric source of chapter 129. The chapter names the finite rows needed by a classical-analysis consumer without asserting ambient completeness, open-cover compactness, host measure theory, quotient function equality, or a selected parametrization outside the displayed data.

Definition 7924.1 (OsgoodCurve carrier)**.** *An* $\mathsf{OsgoodCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, I, S, R, M, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *interface;*
2. *I is the interval or curve-parameter window row, stored as finite* $\mathsf{StreamName}^{\uparrow}$ *observations;*
3. *S is the regular rational sampling schedule for the curve coordinates, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is exposed;*
4. *R is the displayed* $\mathsf{Real}^{\uparrow}$ *coordinate-readback row for the continuous image;*
5. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus and graph-window row tying the samples in S to the compact source K;*
6. *A is the finite positive-area witness row, consisting of dyadic rectangle windows, lower-area ledger entries, and the displayed cover-overlap exclusions used by the packet;*
7. *H stores componentwise* hsame *transports for compact-source, parameter-window, regular-sample, real-readback, continuity, positive-area, provenance, and local naming rows;*
8. *C stores the displayed* Cont *routes for reading the compact source, replaying the coordinate samples, checking the continuity window, and exposing the positive-area witness;*
9. *P is the* Pkg *provenance over* BHist, msame, hsame, Cont, *compact metric rows, stream rows, regular rational rows, real rows, continuous-map rows, positive-area ledgers, and the surrounding* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OsgoodCurve}^{\uparrow}}$ *row.*

The carrier contains no host planar measure space, no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*, no open-cover compactness principle, no quotient equality of curves, and no selected parametrization beyond the displayed stream and regular-rational rows.*

Theorem 7924.2 (OsgoodCurve NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{OsgoodCurve}^{\uparrow}}$ *surface for the carrier of Definition 7924.1 consists of six finite obligations:*

1. *compact source: the accepted packet displays the compact metric source row K and the parameter-window row I through the cited compact and stream interfaces;*
2. *regular coordinate readback: every coordinate read passes through the regular rational schedule S before the terminal real row R is consumed;*
3. *continuity window: the row M supplies the finite modulus and graph-window evidence required by the* $\mathsf{ContinuousMap}^{\uparrow}$ *route;*
4. *positive-area witness: the row A lists the dyadic rectangles, lower-area ledger entries, and overlap exclusions that witness the displayed positive-area claim at the certificate level;*
5. *transport and replay: reads of K, I, S, R, M, A transport through H and replay only along the displayed continuation routes C;*
6. *provenance: the package row P and local naming row N bind the preceding rows as one* $\mathsf{OsgoodCurve}^{\uparrow}$ *NameCert packet.*

Thus the packet offers a finite Osgood-curve certificate for Real and Completion consumers, not a host theorem that every classical positive-area curve has been reconstructed inside BEDC.

Proof. Carrier admission is projection from Definition 7924.1. The compact source, parameter windows, regular coordinate samples, real readback rows, continuity evidence, positive-area ledger, transports, continuation routes, provenance, and local name are all displayed coordinates of the same finite packet.

The coordinate-readback obligation is finite because the stream and regular-rational rows I, S are the only schedules through which R may be read. The continuity obligation is finite because M stores the displayed modulus and graph-window rows rather than a host function-space equality.

The positive-area obligation reads only the dyadic rectangles and lower-area ledger entries stored in A. Overlap exclusions and lower bounds are replayed through C and transported through H, so the certificate never imports an ambient measure algebra or selected measurable representative.

Finally P and N bind the displayed rows as one local NameCert surface. The excluded host objects have no coordinate in O, so every Real or Completion consumer receives only the finite Osgood-curve data stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{OsgoodCurve}^{\uparrow}}$.

Constructive story: An $\mathsf{OsgoodCurve}^{\uparrow}$ packet is generated from a compact metric source, finite stream parameter windows, regular rational coordinate schedules, real readback rows, a continuous-map modulus and graph window, a dyadic positive-area ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OsgoodCurve}^{\uparrow}}$, seed)

Scope closed: Definition 7924.1 and Definition 7924.2 bind the finite Osgood-curve carrier and its NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OsgoodCurve}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close ambient real completeness, host planar measure theory, open-cover compactness, quotient equality of curves, a selected parametrization outside the displayed rows, or bridge checking.

Upgrade path: Obligation closure requires checked finite replay theorems for the coordinate-readback, continuity-window, and positive-area ledger obligations.

Chapter 7925

A Concrete Naming Certificate for $\mathsf{BarrBeckMonadicity}^{\uparrow}$

Following Barr and Beck on monadicity, the $\mathsf{BarrBeckMonadicity}^{\uparrow}$ packet records the finite categorical comparison surface where an adjunction, its induced monad, algebra rows, comparison functor, and exactness witnesses can be named without importing ambient category theory. The packet reads chapter 107, chapter 61, chapter 62, chapter 5827, and chapter 3836. It names the comparison route only; it does not assert existence of all coequalizers, preservation of arbitrary limits, host equality of Eilenberg–Moore categories, quotient completion equality, selected representatives, countable choice, Lean verification, or bridge checking.

Definition 7925.1 (Barr-Beck monadicity carrier)**.** *A* $\mathsf{BarrBeckMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, C, D, F, U, T, E, K, X, Q, H, R, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Adjunction}^{\uparrow}$ *row, displaying the left and right comparison legs together with the unit and counit triangles.*
2. *C is the source* $\mathsf{Cat}^{\uparrow}$ *row from which the adjunction domain is read.*
3. *D is the target* $\mathsf{Cat}^{\uparrow}$ *row in which the monad and algebra rows are compared.*
4. *F is the* $\mathsf{Functor}^{\uparrow}$ *comparison row induced by the displayed adjunction data.*
5. *U is the induced monad row, recording unit and multiplication schedules visible to comparison consumers.*
6. *T is the algebra-action row over the same monad schedule.*
7. *E is the exactness row for the finite reflexive-pair or split-pair evidence exposed to the classifier.*
8. *K is the comparison-kernel row recording how the comparison functor reads algebra data back into the adjunction surface.*
9. *X is the* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *specialization row available to completion-facing consumers.*
10. *Q is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *route that supplies the metric-completion adjunction anchor for categorical completion chapters.*

11. *H records componentwise* hsame *transport for the adjunction, category, functor, monad, algebra, exactness, comparison, completion-specialization, metric-completion route, replay, provenance, and local naming rows.*
12. *R records displayed* Cont *replay from any monadicity request through $A, C, D, F, U, T, E, K, X, Q, H, P, N$.*
13. *P is the* Pkg *provenance over* $\mathsf{BarrBeckMonadicity}^{\uparrow}$, $\mathsf{Adjunction}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{BarrBeckMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, C, D, F, U, T, E, K, X, Q, H, R, P, N$. The carrier has no coordinate for a host Eilenberg–Moore category, an external quotient object, a selected coequalizer, arbitrary-limit preservation, countable-choice schedules, Lean verification, or bridge checking.

Lean4 (statement) `BEDC.Derived.BarrBeckMonadicityUp`

Theorem 7925.2 (Barr-Beck monadicity NameCert obligations)**.** *For every accepted* $\mathsf{BarrBeckMonadicity}^{\uparrow}$ *carrier*

$$B = (A, C, D, F, U, T, E, K, X, Q, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BarrBeckMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, adjunction visibility, source and target category exposure, comparison-functor exposure, induced monad schedule, algebra-action row, exactness witness row, comparison-kernel readback, Cauchy-completion monadicity specialization, metric-completion adjunction anchoring, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 7925.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the adjunction, the two category rows, the comparison functor, the monad schedule, the algebra action, exactness, comparison-kernel readback, completion specialization, metric-completion anchor, transport, replay, provenance, and local naming.

The categorical rows are read first. The row A supplies the adjunction data, C and D supply the source and target categorical surfaces, and F records the comparison functor induced from the same visible data.

The monadicity rows then bind the comparison. The row U supplies unit and multiplication schedules, T records the algebra action over that schedule, E records the finite exactness witness exposed to the classifier, and K records the comparison-kernel readback.

Finally the completion-facing rows anchor downstream use. The row X points to the Cauchy-completion monadicity specialization and Q points to the metric-completion adjunction route, after which H, R, P, N transport, replay, source, and name exactly the displayed packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 7925.3 (Barr-Beck comparison route)**.** *Every categorical-completion consumer of an accepted* $\mathsf{BarrBeckMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$A, C, D, F \longmapsto U, T, E, K \longmapsto X, Q \longmapsto H, R, P, N.$$

Thus the adjunction, category, functor, monad, algebra, exactness, and comparison-kernel rows are read before the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *specialization or the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *anchor is exported. The route exports no host Eilenberg–Moore category, quotient completion equality,*

selected coequalizer, selected limit representative, arbitrary-limit preservation theorem, countable-choice schedule, Lean verification, or bridge checking.

Proof. Start with the public categorical surface. The adjunction row A, source and target category rows C, D, and comparison-functor row F provide the only categorical coordinates available before monadicity is inspected.

Next consume the monadicity rows. The monad schedule U, algebra action T, finite exactness witness E, and comparison-kernel readback K form the visible comparison boundary; no host category equality or arbitrary coequalizer coordinate is present.

Now attach the completion-facing anchors. The Cauchy-specialization row X and metric-completion adjunction row Q identify the downstream completion route without adding quotient completion equality or selected representatives.

Finally route through transport, replay, provenance, and naming. The rows H, R, P, N exhaust the remaining classifier surface, so every exported categorical-completion consumer factors through the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarrBeckMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{BarrBeckMonadicity}^{\uparrow}$ packet is generated from an Adjunction row, two Category rows, a Functor comparison row, induced monad schedule, algebra-action row, finite exactness witness, comparison-kernel readback, CauchyCompletionMonadicity specialization, MetricCompletionAdjunction anchor, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarrBeckMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7925.1, Definition 7925.2, and Definition 7925.3 seed the finite Barr-Beck comparison boundary for categorical completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BarrBeckMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Eilenberg–Moore category equality, selected coequalizers, arbitrary-limit preservation, quotient completion equality, selected representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjunction visibility, source and target category exposure, comparison-functor exposure, induced monad schedule, algebra-action row, exactness witness row, comparison-kernel readback, completion specialization, metric-completion anchoring, transport, replay, provenance, and local naming for BEDC.Derived.BarrBeckMonadicityUp.

Chapter 7926

A Concrete Naming Certificate for $\mathsf{CartanDecomposition}^{\uparrow}$

Following Helgason and Knapp, the $\mathsf{CartanDecomposition}^{\uparrow}$ packet records the finite involution and eigenspace surface used to read a Lie algebra or Lie group through its compact and noncompact rows. It is grounded in the Lie-group route of chapter 140, the Lie-algebra carrier of chapter 141, the root-system and Weyl-group consumers of chapter 144 and chapter 145, and the bilinear-form row of chapter 146. The chapter names the displayed decomposition as finite NameCert data; it does not assert classification of real reductive groups, global symmetric-space theory, selected Cartan subalgebras, or analytic integration.

Definition 7926.1 (Cartan decomposition carrier)**.** *A* $\mathsf{CartanDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (G, L, \Theta, K, P, E, B, R, W, H, C, Q, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{LieGroup}^{\uparrow}$ *source row when a group-level source is present; otherwise it is the explicit absence row for group-level data.*
2. *L is the* $\mathsf{LieAlgebra}^{\uparrow}$ *row whose bracket and additive vector-space operations are the only algebraic operations read by the packet.*
3. *Θ is the involution row. It records an endomap of the displayed Lie-algebra carrier with Θ^2 classified with the identity by the local carrier relation.*
4. *K is the compact eigenspace row, recording the displayed fixed endpoints of Θ.*
5. *P is the noncompact eigenspace row, recording the displayed negated endpoints of Θ.*
6. *E is the eigenspace stability ledger. It records bracket routing for $[K, K]$ into K, $[K, P]$ into P, and $[P, P]$ into K whenever those brackets are displayed.*
7. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *compatibility row used by consumers that inspect orthogonality, sign, or Killing-form-style pairings.*
8. *R is the* $\mathsf{RootSystem}^{\uparrow}$ *consumer row for restricted-root reads of the displayed P-side direction data.*
9. *W is the* $\mathsf{WeylGroup}^{\uparrow}$ *consumer row for reflection reads that are sourced through the same root-system row R.*

10. *H stores componentwise* hsame *transport for the group source, Lie algebra, involution, eigenspace, bracket-stability, bilinear-form, root, Weyl, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a decomposition request through the rows $G, L, \Theta, K, P, E, B, R, W, H$.*
12. *Q is the* Pkg *provenance over* $\mathsf{CartanDecomposition}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, $\mathsf{LieAlgebra}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, $\mathsf{WeylGroup}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CartanDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host symmetric space, a selected maximal abelian subspace, analytic integration, or classification of real forms.

Theorem 7926.2 (Cartan decomposition involution eigenspace stability)**.** *For every accepted* $\mathsf{CartanDecomposition}^{\uparrow}$ *carrier*

$$D = (G, L, \Theta, K, P, E, B, R, W, H, C, Q, N),$$

any consumer that reads the public decomposition must first retain the involution row Θ, the compact row K, the noncompact row P, and the bracket-stability ledger E. The public read may then pass to the $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, *and* $\mathsf{WeylGroup}^{\uparrow}$ *rows, but it cannot replace the finite rows by a host symmetric-space theorem, a selected Cartan subspace, analytic integration, or classification of real forms.*

Proof. Project the carrier of Definition 7926.1. The Lie-algebra source is the row L, and the decomposition data are exactly the involution Θ with the displayed fixed and negated rows K and P.

The stability claim is the row E. It keeps the three bracket routes $[K, K]$, $[K, P]$, and $[P, P]$ attached to the same involution and eigenspace rows before any downstream consumer can inspect the decomposition.

The bilinear-form and root-theoretic consumers are downstream from this retained surface. Rows B, R, W may read only the pairing, restricted-root, and reflection data named by the carrier, while H, C, Q, N transport, replay, source, and name the same finite packet. Since no excluded symmetric-space, selected-subspace, analytic, or classification row is present, the consumer is confined to the displayed NameCert route. □

Falsifiable prediction: Every $\mathsf{CartanDecomposition}^{\uparrow}$ consumer that reaches a restricted-root or Weyl read must first expose the Lie-algebra row, involution row, compact eigenspace row, noncompact eigenspace row, bracket-stability ledger, bilinear-form row, transport, replay, provenance, and local NameCert rows.

Independence witness: liegroup, liealgebra, rootsystem, weylgroup, bilinform: $\mathsf{CartanDecomposition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an involution, two eigenspace rows, a three-branch bracket-stability ledger, bilinear-form compatibility, root consumers, Weyl consumers, structural replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CartanDecomposition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CartanDecomposition}^{\uparrow}$ packet is generated from optional LieGroup source data, a LieAlgebra carrier, an involutive endomap row, compact and noncompact eigenspace rows, a bracket-stability ledger, a BilinForm compatibility row, RootSystem and WeylGroup consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CartanDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7926.1 and Definition 7926.2 seed the finite involution, eigenspace, bracket-stability, bilinear-form, root, and Weyl consumer surface.

Formal verification: FormalStatus(NameCert$_{\text{CartanDecomposition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close classification of real reductive groups, global symmetric-space theory, selected Cartan subspaces, analytic integration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, involution exactness, eigenspace stability, bilinear-form compatibility, root and Weyl consumer routing, structural replay, provenance, and local naming for BEDC.Derived.CartanDecompo

Chapter 7927

A Concrete Naming Certificate for $\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}$

The $\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}$ packet realizes the Poincare-disk branch of chapter 9463 as finite boundary-anchored cocycle data. It sits after the disk metric packet of chapter 6209, the boundary-transport packet of chapter 6267, the additive Busemann-cocycle route of chapter 4134, the visual-metric packet of chapter 3866, and the horocycle carrier of chapter 3534. The packet names a finite disk-model cocycle read; it is not a quotient ideal boundary, a selected geodesic ray, a completed hyperbolic plane, a physical observer law, or an ambient curvature theorem.

Definition 7927.1 (Poincare disk Busemann cocycle carrier)**.** *A* $\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (D, B, M, U, V, Z, O, E, H, T, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-metric row over the displayed disk observations.*
2. *B is the boundary-anchor row, read through* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$*, that records the boundary endpoint held visible by the transport.*
3. *M is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *additive cocycle row for the same boundary anchor and disk observations.*
4. *U is the boundary-transport update row. It records the finite handoff from one disk observation window to another while retaining the boundary anchor B.*
5. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *comparison row used when a visual-distance consumer reads the cocycle.*
6. *Z is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *horocycle row that supplies the boundary-tangent level surface consumed by Busemann-style reads.*
7. *O is the observer-expansion refusal row. It prevents the packet from being read as a physical observer trajectory, quotient boundary equality, selected ideal point, or completed disk model.*
8. *E is the exactness ledger. It ties D, B, M, U, V, Z, O to one cocycle read and records the displayed additivity and boundary-transport compatibility rows.*

9. *H stores componentwise* hsame *transport for disk metric, boundary anchor, cocycle, boundary transport, visual metric, horocycle, refusal, exactness, replay, provenance, and local naming rows.*

10. *T stores displayed* Cont *replay from a boundary-cocycle request through $D, B, M, U, V, Z, O, E, H$.*

11. *G is the* Pkg *provenance over* PoincareDiskBusemannCocycle$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, PoincareDiskBoundaryTranspo HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicHorocycle$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{PoincareDiskBusemannCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed ideal-boundary quotients, selected geodesic representatives, ambient hyperbolic planes, or physical observer dynamics.

Theorem 7927.2 (Poincare disk Busemann cocycle boundary transport)**.** *Every accepted* PoincareDiskBusemannCo *read that transports a Busemann comparison across disk windows must retain the route*

PoincareDiskMetric$^{\uparrow}$, PoincareDiskBoundaryTransport$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, Hyp

The boundary transport may carry the cocycle to a visual-metric or horocycle consumer only while the boundary anchor B, additive cocycle row M, transport row U, exactness ledger E, and refusal row O remain displayed. It does not assert quotient boundary equality, selected geodesic rays, completed disk geometry, physical observer dynamics, or global curvature.

Proof. Project the carrier of Definition 7927.1. The disk observation row is D, the boundary anchor is B, and the additive Busemann comparison is M.

Read the boundary-transport and exactness rows next. The row U moves the displayed disk window only with the same boundary anchor, while E records that the additive cocycle and transport compatibility are inspected inside one finite packet.

The visual-metric and horocycle consumers are downstream. Rows V and Z may read only the comparison and horocycle data named by the same packet, and O prevents the read from escaping into quotient boundary equality, selected geodesic representatives, completed disk geometry, physical observer dynamics, or global curvature. Rows H, T, G, N then preserve, replay, source, and name the same finite route. □

Falsifiable prediction: Every accepted PoincareDiskBusemannCocycle$^{\uparrow}$ consumer must expose disk metric, boundary transport, Busemann cocycle, visual-metric, horocycle, exactness, refusal, transport, replay, provenance, and local NameCert rows before obtaining a boundary-cocycle handoff.

Independence witness: poincare-disk-metric, poincare-disk-boundary-transport, hyperbolic-busemann-cocycle, hyperbolic-visual-metric, hyperbolic-horocycle: none is bijective to PoincareDiskBusemannCocyc because this packet jointly carries disk metric data, a retained boundary anchor, additive cocycle, boundary-window transport, visual-metric comparison, horocycle level data, exactness, refusal, structural replay, provenance, and local naming.

The Lean TasteGate route must provide explicit Nontrivial PoincareDiskBusemannCocycle$^{\uparrow}$ and FieldFaithful PoincareDiskBusemannCocycle$^{\uparrow}$ instances. Nontriviality must distinguish an accepted packet with all rows $D, B, M, U, V, Z, O, E, H, T, G, N$ from a candidate missing the boundary anchor or exactness ledger. Field faithfulness must preserve those rows as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}$ packet is generated from PoincareDiskMetric rows, PoincareDiskBoundaryTransport rows, HyperbolicBusemannCocycle rows, HyperbolicVisualMetric rows, HyperbolicHorocycle rows, an exactness ledger, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7927.1 and Definition 7927.2 seed the finite Poincare-disk Busemann cocycle route through disk metric, boundary transport, additive cocycle, visual metric, horocycle, exactness, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskBusemannCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal-boundary equality, selected geodesic rays, completed hyperbolic-plane geometry, physical observer dynamics, global curvature, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-metric admission, boundary-anchor transport, Busemann cocycle exactness, visual-metric and horocycle routing, TasteGate nontriviality, field faithfulness, structural replay, provenance, and local naming for BEDC.Derived.PoincareDiskBusemannCocycleUp.

Chapter 7928

A Concrete Naming Certificate for $\mathsf{CalderonZygmundOperator}^{\uparrow}$

Following Calderon–Zygmund and Stein on singular integral operators, the $\mathsf{CalderonZygmundOperator}^{\uparrow}$ packet records the finite kernel-window surface by which a truncated singular-integral request is named before any host principal value or measure completion is available. It sits beside the maximal domination route of chapter 3581, the finite cover ledger of chapter 2166, and the terminal real-name seal of chapter 38. The packet names only displayed source rows, kernel windows, size and cancellation ledgers, local test or average rows, scalar seals, transport, replay, provenance, and local naming; it does not assert selected principal values, completed measure spaces, quotient function equality, weak-type theorems, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7928.1 (Calderon-Zygmund operator carrier)**.** *A* $\mathsf{CalderonZygmundOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, W, S, A, L, M, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric or measure source row. It records the named points, measurable windows, local ball names, and source-side admission data actually inspected by the singular-integral request.*
2. *W is the truncated kernel-window row. Each entry records a finite exclusion radius, an outer cutoff, and the displayed source and target coordinates for the kernel read.*
3. *S is the kernel-size ledger. It stores the finite dyadic comparison budget for the displayed kernel windows, including the scale rows that compare kernel magnitude against the source distance.*
4. *A is the cancellation ledger. It records the displayed zero-average, antisymmetry, or subtraction row used on each accepted window before any operator value is read.*
5. *L is the local test-function or average row. It stores the finite test window, local average, or compactly supported probe against which the truncated kernel is evaluated.*
6. *M is the* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *domination handoff. It records the maximal-function comparison route used to control displayed local averages without selecting a host supremum.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scalar ledger for truncation radii, kernel-size comparisons, cancellation budgets, local averages, and domination tolerances.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is public only after W, S, A, L, M, D have all been displayed.*

9. *H stores componentwise* hsame *transport for X, W, S, A, L, M, D, E.*
10. *C stores displayed* Cont *replay of the finite route from a singular-integral request through truncation, kernel-size comparison, cancellation, local testing, maximal domination, dyadic accounting, and Real sealing.*
11. *P is the* Pkg *provenance over* CalderonZygmundOperator$^{\uparrow}$, HardyLittlewoodMaximal$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CalderonZygmundOperator}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, W, S, A, L, M, D, E, H, C, P, N$. It has no coordinate for a selected principal value, measure completion, quotient function representative, unbounded kernel family, hidden cancellation theorem, or ambient Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CalderonZygmundOperatorUp.CalderonZygmundOperatorUp single carrier alignment`

Theorem 7928.2 (Calderon-Zygmund operator NameCert obligations)**.** *For every accepted* CalderonZygmundOper *carrier*

$$T = (X, W, S, A, L, M, D, E, H, C, P, N),$$

the local NameCert$_{\text{CalderonZygmundOperator}^{\uparrow}}$ *surface consists exactly of source admission, truncated kernel-window exposure, kernel-size ledger visibility, cancellation-ledger visibility, local test or average exposure,* HardyLittlewoodMaximal$^{\uparrow}$ *domination handoff,* DyadicRatCore$^{\uparrow}$ *scalar accounting, terminal* Real$^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. A singular-integral read is accepted only after W, S, A, L, M, D are all public in the same packet.*

Proof. Project the carrier of Definition 7928.1. The displayed tuple exposes exactly the source row, truncated windows, kernel-size ledger, cancellation ledger, local test or average rows, maximal handoff, dyadic ledger, Real seal, transport, replay, provenance, and local naming row.

The operator-facing rows are ordered by the carrier. A kernel read in W becomes admissible only together with the size ledger S and the cancellation ledger A, and a local output can be read only through the test or average row L.

The domination and scalar rows then close the finite numerical surface. The maximal-function handoff M supplies the displayed comparison route, while D records the dyadic tolerances and E seals the resulting real-name window.

Finally H, C, P, N preserve exactly the same packet. Since the carrier has no coordinate for selected principal values, completed measure spaces, quotient representatives, unbounded kernel families, hidden cancellation theorems, or ambient Real$^{\uparrow}$ completeness, those objects cannot appear as local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CalderonZygmundOperatorUp.CalderonZygmundOperatorNameCertObligations`

Theorem 7928.3 (Calderon-Zygmund kernel cancellation ledger)**.** *Every accepted* CalderonZygmundOperator$^{\uparrow}$ *read factors its kernel request through the finite route*

$$X, W \longmapsto S, A, L \longmapsto \text{HardyLittlewoodMaximal}^{\uparrow}, \text{DyadicRatCore}^{\uparrow} \longmapsto \text{Real}^{\uparrow}$$

before any operator-facing value is exported. Deleting the cancellation ledger A or the size ledger S leaves no public singular-integral read in the packet.

Proof. Start with the NameCert obligations of Definition 7928.2. They require each accepted read to expose the truncated kernel row W, the size ledger S, the cancellation ledger A, and the local test or average row L before domination or scalar sealing is available.

The handoff through M is therefore a domination route for a displayed finite window, not a host supremum or completed measure statement. The dyadic ledger D and Real seal E can only read the rows already exposed by W, S, A, L, M.

The structural rows H, C, P, N preserve this order. If S is absent, the packet has no public size comparison for the kernel window; if A is absent, the packet has no displayed cancellation or subtraction row. In either case there is no coordinate from which the singular-integral read can be exported. □

Falsifiable prediction: Every accepted $\mathsf{CalderonZygmundOperator}^{\uparrow}$ read is refuted if a Real seal is cited without the displayed truncated kernel windows, kernel-size ledger, cancellation ledger, local test or average row, HardyLittlewoodMaximal domination handoff, and DyadicRatCore scalar ledger.

Independence witness: hardy-littlewood-maximal, finite-vitali-cover, dyadicratcore, real: none is bijective to $\mathsf{CalderonZygmundOperator}^{\uparrow}$ because this packet jointly carries a source row, truncated kernel windows, kernel-size ledger, cancellation ledger, local test or average rows, maximal domination handoff, scalar ledger, Real seal, transport, replay, provenance, and local naming as one singular-integral operator certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CalderonZygmundOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{CalderonZygmundOperator}^{\uparrow}$ packet is generated from a metric or measure source row, truncated kernel windows, a kernel-size ledger, a cancellation ledger, local test or average rows, a HardyLittlewoodMaximalUp domination handoff, DyadicRatCoreUp scalar accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CalderonZygmundOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7928.1, Definition 7928.2, and Definition 7928.3 seed the finite singular-integral operator carrier, local NameCert surface, and kernel cancellation ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CalderonZygmundOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected principal values, measure completion, quotient function equality, weak-type or strong-type boundedness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, truncated kernel-window exposure, kernel-size ledger visibility, cancellation-ledger visibility, local test or average exposure, maximal domination handoff, scalar accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CalderonZygmundOperatorUp.

Chapter 7929

A Concrete Naming Certificate for LebesgueNumberCoverRefinement$^{\uparrow}$

Following Bishop–Bridges constructive compactness and Kelley's cover-refinement vocabulary, the LebesgueNumberCoverRefinement$^{\uparrow}$ packet records the finite route that turns a displayed compact cover, a positive Lebesgue-number radius, and a finite refinement request into a retained dyadic cover refinement. It sits over the radius lemma of chapter 2301, the finite comparison packet of chapter 3430, the star-refinement modulus route of chapter 6997, and the located finite-subcover route of chapter 3538. The packet names a finite cover-refinement certificate; it is not arbitrary open-cover compactness, not a selected subcover functional, and not ambient real completeness.

Definition 7929.1 (Lebesgue-number cover-refinement carrier)**.** *A* LebesgueNumberCoverRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, L, F, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, exposing the compact metric or located compact interval data only through the finite carrier already accepted by the surrounding chapter.*
2. *L is the* LebesgueNumberLemma$^{\uparrow}$ *positive-radius row attached to the displayed cover request.*
3. *F is the* FiniteCoverRefinement$^{\uparrow}$ *comparison row that records how the retained cover members refine the source cover.*
4. *D is the dyadic cover-refinement row, storing the mesh, ball, or cell comparisons used to compare the radius row with the retained finite cover.*
5. *S is the retained subcover row, listing only the finitely many cover witnesses reached through K, L, F, D.*
6. *E is the terminal* Real$^{\uparrow}$ *seal for radius, endpoint, and coverage observations used by the refinement read.*
7. *H stores componentwise* hsame *transport for the compact source, radius row, finite refinement row, dyadic refinement row, retained subcover row, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a compact cover request through $K, L, F, D, S, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, FiniteCoverRefinemer $\mathsf{FiniteCoverStarRefinementModulus}^{\uparrow}$, $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, L, F, D, S, E, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, choice-selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7929.2 (Lebesgue-number cover-refinement NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}$ *carrier*

$$R = (K, L, F, D, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact source, Lebesgue-number radius, finite cover refinement, dyadic cover refinement, retained subcover, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows; radius compatibility through L, D; finite refinement exactness through F, S; Real-seal non-escape through E; and structural preservation through H, C, P, N.*

Proof. Project the accepted carrier of Definition 7929.1. The analytic rows are K, L, F, D, S, E, and the structural rows are H, C, P, N.

The positive-radius data are read from L and compared with the dyadic refinement row D. The finite comparison row F then determines which retained witnesses may enter S.

The terminal seal E is downstream of those finite rows. Transport, replay, provenance, and local naming preserve the same displayed packet, so no obligation can be discharged by an arbitrary open-cover compactness principle, a selected subcover functional, quotient real equality, countable choice, or ambient completeness. □

Theorem 7929.3 (Lebesgue-number radius-to-refinement route)**.** *Every accepted* LebesgueNumberCoverRefinemen *cover read factors through the finite route*

$$\mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto D, S \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported object is the retained dyadic cover-refinement surface over $K, L, F, D, S, E, H, C, P, N$, not an arbitrary open-cover compactness theorem or a choice-selected subcover.

Proof. Start with an accepted carrier $R = (K, L, F, D, S, E, H, C, P, N)$. The compact source is K, the positive-radius row is L, and the finite comparison row is F.

The dyadic row D compares the radius with the displayed finite cells. The retained subcover row S is readable only after that comparison, so a consumer cannot bypass the radius ledger or the finite refinement packet.

Finally the Real seal E and the structural rows H, C, P, N replay the same finite route. Since the carrier has no coordinate for arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient completeness, the exported refinement is exactly the displayed BEDC packet. □

Falsifiable prediction: An accepted $\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}$ consumer that exports a retained cover refinement without displaying $K, L, F, D, S, E, H, C, P, N$ refutes the carrier surface.

Independence witness: Nearest siblings: lebesgue-number-lemma, finite-cover-refinement, finite-cover-star-refinement-modulus, and located-heine-borel-subcover. The packet binds the positive-radius row to a retained dyadic finite-refinement surface, while each sibling supplies only its own carrier route.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}$ packet is generated from a compact source row, a LebesgueNumberLemma positive-radius row, a FiniteCoverRefinement comparison row, dyadic cover-refinement rows, retained finite subcover witnesses, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7929.1, Definition 7929.2, and Definition 7929.3 seed the finite radius-to-refinement route over compact source rows, positive-radius ledgers, finite comparison rows, dyadic cover refinements, retained subcovers, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberCoverRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius compatibility, finite refinement exactness, dyadic refinement admission, retained-subcover exactness, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueNumberCoverRefinementUp.

Chapter 7930

A Concrete Naming Certificate for $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, a regular Cauchy comparison is allowed to produce apartness or located comparison data only after the finite equality-side readback has exposed the positive gap witness it uses. The $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ packet records that separator as NameCert data between the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the located-cut comparison boundary of chapter 404, and the apartness-field route of chapter 5562. It is a finite boundary object for Real and Completion consumers; it does not claim trichotomy, quotient equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7930.1 (Regular sequence apartness separator carrier)**.** *A* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, E, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the two regular sequence reads.*
2. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row computed from those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that fixes the positive gap scale.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available after S, R, D have been displayed.*
5. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *comparison row read at the same dyadic scale.*
6. *A is the* $\mathsf{RealApartnessField}^{\uparrow}$ *apartness row exposed from the located comparison and the Real seal.*
7. *H stores componentwise* hsame *transport for the stream windows, regular readbacks, dyadic tolerance, Real seal, located comparison, apartness row, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from a regular sequence comparison request through $S, R, D, E, L, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{RealApartnessField}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSeqApartnessSeparator}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, quotient stream equality, quotient real equality, selected representatives, countable choice, or ambient completeness.

Theorem 7930.2 (Regular sequence apartness separator obligation surface)**.** *For every accepted* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *carrier*

$$K = (S, R, D, E, L, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularSeqApartnessSeparator}^{\uparrow}}$ *surface consists exactly of paired finite-window exposure, paired regular-rational readback, dyadic positive-gap accounting, terminal Real-seal routing, located-cut comparison exposure, apartness-field exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 7930.1. The analytic rows are S, R, D, E, L, A, and the structural rows are H, C, P, N.

The stream and regular-rational rows S, R expose the two sequence reads before any comparison is consumed. The dyadic row D fixes the positive scale used by both the located comparison and the apartness read.

The terminal seal E is downstream of S, R, D. The located-cut row L and apartness-field row A are then read from that same displayed route, so no consumer can obtain apartness or located comparison data before the regular sequence and dyadic rows are visible.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded principles is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 7930.3 (Regular sequence apartness separator sibling dependency)**.** *Every Real-facing or Completion-facing consumer that reads apartness or located comparison data from a* RegularSeqApartnessSeparatc *packet must first pass through the sibling route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedCut}^{\uparrow},\ \mathsf{RealApartnessField}^{\uparrow}.$$

The route exports only the displayed finite separator rows $S, R, D, E, L, A, H, C, P, N$ *and does not export trichotomy, quotient equality, selected representatives, countable choice, or ambient completeness.*

Proof. Apply the obligation surface of Definition 7930.2. It requires the stream windows, regular-rational readbacks, dyadic gap scale, Real seal, located-cut comparison, and apartness-field evidence to be displayed as one route before a local NameCert row is consumed.

The sibling chapters named in the route supply exactly those rows: $\mathsf{StreamName}^{\uparrow}$ supplies S, $\mathsf{RegSeqRat}^{\uparrow}$ supplies R, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies D, $\mathsf{Real}^{\uparrow}$ supplies E, $\mathsf{LocatedCut}^{\uparrow}$ supplies L, and $\mathsf{RealApartnessField}^{\uparrow}$ supplies A.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. They add no total order branch, quotient equality test, representative selector, choice schedule, or completeness theorem, so the separator remains a finite boundary between equality-side readback and apartness-side use. □

Lean4 (statement) `BEDC.Derived.RegularSeqApartnessSeparatorUp`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSeqApartnessSeparator}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ packet is generated from paired StreamName finite windows, paired RegSeqRat readbacks, a DyadicRatCore positive-gap row, a RealUp seal, a LocatedCut comparison row, a RealApartnessField apartness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularSeqApartnessSeparator}\uparrow}$, seed)

Scope closed: Definition 7930.1, Definition 7930.2, and Definition 7930.3 seed the finite separator route between regular sequence equality-side readback, located comparison, and apartness-field exposure.

Formal verification: FormalStatus(NameCert$_{\text{RegularSeqApartnessSeparator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient stream equality, quotient real equality, selected representatives, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular-rational readback, dyadic positive-gap accounting, Real-seal routing, located-cut comparison exposure, apartness-field exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularSeqApartnessSeparatorUp.

Chapter 7931

A Concrete Naming Certificate for StoneRepresentationBooleanAlgebra$^{\uparrow}$

Following Stone and Birkhoff, the StoneRepresentationBooleanAlgebra$^{\uparrow}$ packet records the finite NameCert bridge that reads a displayed Boolean algebra through its Stone-space and clopen-row interface. It sits after the Boolean operation surface of chapter 3733, the Stone-duality classifier of chapter 261, the topology rows of chapter 88, and the regular-open comparison route of chapter 6835. The packet names the representation handoff as finite BEDC data; it does not import a host spectrum, choice-selected ultrafilter family, quotient Boolean algebra equality, or completed compact Hausdorff space.

Definition 7931.1 (Stone representation Boolean algebra carrier)**.** *A* StoneRepresentationBooleanAlgebra$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, U, T, L, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BooleanAlgebra$^{\uparrow}$ *source row, exposing zero, one, join, meet, complement, order, and Boolean transport data.*
2. *U is the ultrafilter-observation row, recording only finite membership tests and finite compatibility checks against the Boolean source row.*
3. *T is the* StoneDuality$^{\uparrow}$ *Stone-space classifier row attached to the Boolean source.*
4. *L is the clopen-row ledger, reading each displayed Boolean element as a topology-facing clopen row through* Topology$^{\uparrow}$*.*
5. *M is the regular-open comparison row, allowing chapter 6835 to compare clopen rows with its closure-interior route without replacing the Boolean source.*
6. *H records componentwise* hsame *transport for Boolean rows, ultrafilter observations, Stone-space rows, clopen rows, regular-open comparisons, provenance, and local naming.*
7. *C records displayed* Cont *replay from a Boolean observation request through B, U, T, L, M and then through the structural rows.*
8. *P is the* Pkg *provenance over* StoneRepresentationBooleanAlgebra$^{\uparrow}$, BooleanAlgebra$^{\uparrow}$, StoneDuality$^{\uparrow}$, Topology$^{\uparrow}$, RegularOpenAlgebra$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, U, T, L, M, H, C, P, N$. It has no coordinate for a host powerset algebra, a choice-selected ultrafilter space, quotient Boolean equality, ambient compactness, or a replacement of the displayed Boolean-algebra carrier.

Lean4 ✓ `BEDC.Derived.StoneRepresentationBooleanAlgebraUp.StoneRepresentationBooleanAlgebraTasteGate single carrier alignment`

Theorem 7931.2 (Stone representation Boolean algebra NameCert obligations)**.** *For every accepted* $\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}$ *carrier $S = (B, U, T, L, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StoneRepresentat}}$ *surface consists exactly of carrier admission for the displayed Boolean source, ultrafilter-observation row, Stone-space classifier, clopen ledger, regular-open comparison, transport, replay, provenance, and local naming rows; Boolean operation exposure through B; finite ultrafilter compatibility through U; Stone-space classification through T; clopen readback through L; regular-open comparison through M; and non-escape through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.StoneRepresentationBooleanAlgebraUp.StoneRepresentationBooleanAlgebraCarrier namecert obligations`

Proof. Project the carrier of Definition 7931.1. Its Boolean source is B, its representation-facing rows are U, T, L, M, and its structural rows are H, C, P, N.

The Boolean obligations are inherited as displayed rows from chapter 3733. Zero, one, join, meet, complement, order, and transport data must be visible before any Stone-facing row is read.

The representation rows then remain finite. The ultrafilter row U records only displayed membership and compatibility tests, while T and L read the Stone-space classifier and clopen ledger supplied by chapter 261 and chapter 88.

Finally M, H, C, P, N compare, transport, replay, source, and name the same rows. Since no carrier coordinate stores a host spectrum, quotient Boolean equality, selected ultrafilter family, ambient compactness theorem, or replacement Boolean algebra, the listed obligations exhaust the local certificate surface. □

Theorem 7931.3 (Stone representation Boolean algebra duality handoff)**.** *Every Stone-duality or topology-facing consumer of an accepted* $\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}$ *carrier must read the handoff through*

$$\mathsf{BooleanAlgebra}^{\uparrow} \longrightarrow \mathsf{StoneDuality}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{RegularOpenAlgebra}^{\uparrow}$$

as displayed rows attached to B, T, L, M. The ultrafilter-observation row U, transport row H, replay row C, provenance row P, and local name N remain visible throughout the handoff.

The handoff exports a finite clopen representation route, not a host-level representation theorem, selected spectrum, quotient equality of Boolean algebras, or completed compact Hausdorff object.

Lean4 ✓ `BEDC.Derived.StoneRepresentationBooleanAlgebraUp.StoneRepresentationBooleanAlgebraCarrier duality handoff`

Proof. Start with the NameCert obligations of Definition 7931.2. They require every representation read to expose $B, U, T, L, M, H, C, P, N$ together.

For the source side, use the Boolean-algebra carrier of chapter 3733. The operations and order rows remain the source of the read; no topology row can replace them.

For the dual side, read the Stone-space classifier and clopen ledger through chapter 261 and chapter 88. The regular-open comparison of chapter 6835 is only a downstream comparison route through M.

The structural rows H, C, P, N preserve this factorization. Because the carrier contains no coordinate for the refused host objects, every downstream consumer receives exactly the displayed finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}$ packet is generated from BooleanAlgebra source rows, finite ultrafilter-observation rows, StoneDuality classifier rows, Topology clopen ledgers, RegularOpenAlgebra comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7931.1, Definition 7931.2, and Definition 7931.3 seed the finite Boolean-algebra Stone representation bridge and its duality handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneRepresentationBooleanAlgebra}^{\uparrow}}, \mathsf{encodedDef})$

Bridge status: none

Not claimed: This chapter does not close host Stone representation, selected spectra, quotient Boolean equality, compact Hausdorff completion, Lean theorem verification beyond the carrier definition, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Boolean operation exposure, finite ultrafilter compatibility, Stone-space classification, clopen readback, regular-open comparison, transport, replay, provenance, and local naming for BEDC.Derived.StoneRepresentationBooleanAlgebraUp.

Lean4 (definition) `BEDC.Derived.StoneRepresentationBooleanAlgebraUp`

Chapter 7932

A Concrete Naming Certificate for CauchyCondensationRemainderGauge$^{\uparrow}$

Following Cauchy and Hardy on condensation of monotone positive series, the CauchyCondensationRemainderGauge$^{\uparrow}$ packet records the finite remainder-budget surface used when a dyadic tail estimate is read through condensed blocks. It sits after the condensation witness of chapter 1464, the tail-diameter packet of chapter 1746, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a named BEDC object for condensation-style remainder budgets; it is not a convergence theorem for arbitrary series, a quotient-real equality principle, a selected limit, or a completeness row.

Definition 7932.1 (Cauchy condensation remainder gauge carrier)**.** *A* CauchyCondensationRemainderGauge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, Q, B, R, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source positive monotone series or* RegSeqRat$^{\uparrow}$ *tail row whose remainder is being budgeted.*
2. *M is the monotonicity certificate row required before the condensation block route can be read.*
3. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance query requested by the downstream tail or real-completion consumer.*
4. *B is the finite block schedule row, naming the condensed dyadic blocks inspected by the gauge.*
5. *R is the block-remainder ledger, recording the displayed upper budget for all blocks beyond the named block schedule.*
6. *D is the* CauchyTailDiameter$^{\uparrow}$ *dependency row tying the block-remainder budget to tail-diameter control.*
7. *W is the window-stability row requiring the source, block schedule, dyadic tolerance, and remainder ledger to be read through the same finite* StreamName$^{\uparrow}$ *window family.*
8. *E is the* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *handoff row reached only after S, M, Q, B, R, D, W are visible.*
9. *H records componentwise* hsame *transport for the source, monotonicity, tolerance, block schedule, remainder ledger, tail-diameter dependency, window stability, handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the source row through the monotonicity, tolerance, block schedule, remainder, tail-diameter, window-stability, and handoff rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}$, $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, Q, B, R, D, W, E, H, C, P, N$. It has no coordinate for a host summation theorem, arbitrary-series convergence, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Theorem 7932.2 (Cauchy condensation remainder gauge block stability)**.** *For every accepted* $\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}$ *carrier $G = (S, M, Q, B, R, D, W, E, H, C, P, N)$, a remainder read is stable under refinement of the displayed condensed block schedule precisely when the refined request preserves the same source row S, monotonicity row M, dyadic tolerance Q, block schedule face B, block-remainder ledger R, tail-diameter dependency D, window-stability row W, and handoff row E. The refined read therefore remains a finite block-budget comparison and exports no host summation theorem, arbitrary-series convergence, quotient real equality, selected limit, countable-choice row, ambient completeness claim, or standard bridge.*

Proof. Project the carrier. The only rows from which a condensed remainder can be read are S, M, Q, B, R, D, W, E together with the structural rows H, C, P, N.

For a refinement of the block schedule, require preservation of S, M, Q, B, R, D, W, E. The remainder ledger R is then still read against the same dyadic tolerance row Q, while D keeps the budget tied to the tail-diameter surface and W keeps all observations inside one finite window family.

Transport and replay do not add analytic coordinates. The row H moves exactly the displayed components, and C replays the same source-monotonicity-tolerance-block-remainder-tail route.

Thus the refined request is another presentation of the same finite budget. Since no carrier row names a host summation theorem, arbitrary-series convergence, quotient equality, selected limit, countable choice, ambient completeness, or bridge datum, none is exported by the refinement. □

Theorem 7932.3 (Cauchy condensation remainder gauge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}$ *carrier $G = (S, M, Q, B, R, D, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}}$ *surface consists exactly of carrier admission, monotonicity exposure, dyadic tolerance routing, finite condensed-block scheduling, block-remainder ledger exactness, tail-diameter dependency, finite-window stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise transport, displayed replay, package provenance, local naming, and refusal of host summation, arbitrary-series convergence, quotient real equality, selected limits, countable choice, ambient completeness, and standard-bridge data.*

Proof. Carrier admission exposes the tuple $S, M, Q, B, R, D, W, E, H, C, P, N$. The monotonicity row M and block schedule B are the only condensation-specific rows, while Q, R, D, W record the dyadic budget, remainder ledger, tail-diameter dependency, and finite-window discipline.

The handoff row E is downstream of those analytic rows. A $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can therefore read a remainder budget only after the source, monotonicity, tolerance, block, remainder, tail-diameter, and window rows have been displayed.

The remaining rows are structural. Componentwise transport, displayed replay, provenance, and local naming preserve the same finite coordinates and do not introduce any ambient summation or completion object.

Therefore the NameCert obligations are exhausted by the listed rows. The refused host summation, arbitrary convergence, quotient equality, selected limit, countable-choice, completeness, and bridge objects are absent because no coordinate of the carrier stores them. □

Theorem 7932.4 (Cauchy condensation remainder gauge real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tail consumer that uses a* $\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}$ *packet must pass through the ordered finite route*

$$S, M, Q \longmapsto B, R \longmapsto D, W \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed monotone-series source, dyadic tolerance, condensed block schedule, remainder ledger, tail-diameter dependency, finite-window stability, and regular-rational or real-name handoff. It receives no host summation theorem, arbitrary-series convergence, quotient real equality, selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge.*

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ tail consumer. The carrier gives the consumer a finite packet $G = (S, M, Q, B, R, D, W, E, H, C, P, N)$, so the read must start from the displayed source and monotonicity rows.

Next read the condensation budget. The dyadic tolerance Q fixes the requested size, the block schedule B names the condensed blocks, and the ledger R records the remainder budget for the tail beyond the displayed schedule.

The handoff is guarded by D and W. The tail-diameter dependency D ties the block budget to tail control, while the window-stability row W keeps the source observations, block rows, and remainder ledger in one finite $\mathsf{StreamName}^{\uparrow}$ family.

Finally, E, H, C, P, N hand the same data to the regular-rational or real-name consumer and preserve it under transport and replay. Since the route never opens any additional coordinate, it cannot export host summation, arbitrary convergence, quotient equality, selected limits, countable choice, ambient completeness, or a standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}$ packet is generated from a positive monotone source row, dyadic tolerance query, finite condensed-block schedule, block-remainder ledger, tail-diameter dependency, StreamName window-stability row, RegSeqRat or Real handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7932.1, Definition 7932.2, Definition 7932.3, and Definition 7932.4 seed the finite condensation remainder-budget packet over monotone source rows, dyadic tolerances, condensed block schedules, block-remainder ledgers, tail-diameter dependencies, finite-window stability, and RegSeqRat or Real handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRemainderGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host summation, arbitrary-series convergence, quotient real equality, selected limits, countable choice, ambient RealUp completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, block stability, NameCert obligations, and RegSeqRat or Real handoff for BEDC.Derived.CauchyCondensationRemainderGaugeUp.

Chapter 7933

A Concrete Naming Certificate for $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$

Bishop–Bridges grounding for constructive real analysis treats real approximation by finite rational brackets and explicit moduli before any completed real is consumed. The $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ packet records that finite means as BEDC NameCert data: it sits over the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric-facing distance route of chapter 1638. The packet supplies bracket evidence for Real-and-Completion consumers without asserting $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, trichotomy, or countable choice.

Definition 7933.1 (Finite dyadic bracketing-modulus carrier)**.** *A* $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, L, U, M, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing dyadic denominators, positive radii, midpoint arithmetic, endpoint comparisons, and slack inequalities for the bracket.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming exactly which observation windows are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows and dyadic tolerances.*
4. *L and U are the lower and upper dyadic bracket rows, with $L \leq U$ witnessed inside D.*
5. *M is the finite modulus row assigning each admitted precision request to a displayed window in S and a displayed dyadic bracket in D, L, U.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after D, S, R, L, U, M have been displayed.*
7. *Q is the* $\mathsf{RealMetric}^{\uparrow}$ *distance-read row available to metric consumers only through the same dyadic bracket and readback route.*
8. *H stores componentwise* hsame *transport for the dyadic ledger, window schedule, regular readback, bracket endpoints, modulus row, Real seal, metric read, provenance, and local name.*
9. *C stores displayed* Cont *replay from a precision request through $M, S, D, L, U, R, E, Q, H$.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, trichotomy, countable choice, host real equality, or an unbounded completion selector.*

Theorem 7933.2 (Finite dyadic bracketing-modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ *carrier* $B = (D, S, R, L, U, M, E, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicBracket}}$ *surface consists exactly of carrier admission, dyadic ledger exposure, finite-window scheduling, regular rational readback, lower and upper bracket exposure, modulus exactness, Real seal handoff, metric-read handoff, componentwise transport, displayed replay, package provenance, and local naming.*

Modulus exactness means that every precision request accepted by M names a finite row of S and a finite dyadic bracket in D, L, U before R, E, or Q can be consumed.

Proof. Carrier admission is projection from Definition 7933.1. The accepted packet displays the dyadic ledger, stream windows, regular readback, bracket endpoints, modulus row, Real seal, metric-read row, transport, replay, provenance, and local name.

The modulus row is finite because every request in M points to a displayed window in S and a displayed dyadic comparison in D. The bracket rows L, U are not independent real-order decisions; they are named only after the dyadic ledger has supplied the endpoint arithmetic and slack comparisons.

The Real and metric-facing rows are downstream. A consumer reaches E or Q only after D, S, R, L, U, M have been read, and the structural rows H, C, P, N transport, replay, source, and name that same route. No omitted coordinate can supply quotient streams, selected limits, trichotomy, countable choice, or ambient completeness. □

Theorem 7933.3 (Finite dyadic bracketing-modulus Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealMetric}^{\uparrow}$$

as displayed by $D, S, R, E, Q, H, C, P, N$, with the bracket endpoints L, U and modulus row M read before the terminal seal or metric distance row is exposed. The handoff gives bracket evidence and a finite precision schedule, not $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, trichotomy, countable choice, or host real equality.*

Proof. Start from the NameCert obligations. They force every accepted request to expose D, S, R, L, U, M before a Real-facing row is read.

The dyadic stage reads D, L, U as one bracket ledger. The window and readback stage then reads S and R, so the bracket is attached to displayed finite observations rather than to an ambient completed real.

Only after those rows are present may the terminal seal E and metric read Q be consumed. The provenance row points to the sibling surfaces of chapter 291, chapter 126, chapter 301, chapter 38, and chapter 1638. Thus the handoff remains a finite route through displayed BEDC rows. □

Theorem 7933.4 (Finite dyadic bracketing-modulus non-escape)**.** *No accepted* $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ *packet exports* $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, trichotomy, countable choice,*

host real equality, or an unbounded completion selector. Every consumer read factors through $D, S, R, L, U, M, E, Q, H, C, P, N$ and through the sibling rows named in Definition 7933.3.

Proof. Inspect the carrier of Definition 7933.1. Its analytic rows are the dyadic ledger, finite windows, regular readback, bracket endpoints, modulus row, Real seal, and metric-read row.

Apply the Real handoff theorem. The terminal rows E and Q are available only after the finite bracket and modulus route has been displayed.

The excluded principles require data absent from the tuple: an ambient completeness witness, quotient-stream equality, a selected limit object, a trichotomy branch, a countable-choice selector, host real equality, or unbounded completion selection. Since the packet contains none of those rows, every read remains inside the finite bracketing-modulus certificate. □

Falsifiable prediction: An accepted $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{RealMetric}^{\uparrow}$ distance read without exposing $D, S, R, L, U, M, E, Q, H, C, P, N$ refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, realmetric: no listed sibling supplies the joint packet containing a DyadicRatCore ledger, finite StreamName windows, RegSeqRat readback, lower and upper dyadic brackets, a finite modulus row, a Real seal, a RealMetric read, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}$ packet is generated from a DyadicRatCore ledger, finite StreamName windows, RegSeqRat readback, lower and upper dyadic bracket rows, a finite modulus row, a RealUp seal, a RealMetricUp read, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7933.1, Definition 7933.2, Definition 7933.3, and Definition 7933.4 seed the finite bracketing-modulus certificate for Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDyadicBracketingModulus}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, trichotomy, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic ledger exposure, finite-window scheduling, regular rational readback, bracket exactness, modulus exactness, Real seal handoff, metric-read handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDyadicBracketingModulusUp.

Chapter 7934

A Concrete Naming Certificate for $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ packet records common-modulus Cauchy name spaces as finite BEDC data before a completion or real-name consumer may read a terminal seal. It sits after the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the ordinary Cauchy sequence-space carrier of chapter 406. The packet names a family of Cauchy names controlled by one displayed modulus; it is not quotient sequence equality, a selected limit, host sequence extensionality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7934.1 (Uniform Cauchy sequence space carrier)**.** *A* $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, D, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the inspected family of rational sequence names.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare all inspected tails.*
4. *Q is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *admission row for the underlying Cauchy sequence-space surface.*
5. *M is the common modulus row, displaying one tail budget shared by every admitted name in the packet.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, Q, M have been displayed.*
7. *H records componentwise* hsame *transport for the stream schedule, regular readback, dyadic ledger, sequence-space row, common modulus, seal row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a requested tolerance through $S, R, D, Q, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient sequence equality, selected limits, host sequence extensionality, countable-choice moduli, trichotomy, or ambient completeness.

Theorem 7934.2 (Uniform Cauchy sequence space obligations)**.** *For every accepted* $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ *carrier* $U = (S, R, D, Q, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySequenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed stream schedule, regular rational readback, dyadic tolerance ledger, Cauchy sequence-space row, common modulus row, terminal real-name seal, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing read must pass through*

$$S, R, D, Q, M, E, H, C, P, N$$

and through no quotient sequence equality, selected limit, host sequence extensionality, countable-choice modulus, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7934.1. The analytic rows are exactly S, R, D, Q, M, E, and the structural rows are exactly H, C, P, N.

The finite schedule S exposes the inspected windows, while R records the regular rational readback extracted from those windows. The dyadic ledger D supplies the tolerance cells used by every comparison.

The sequence-space row Q admits the ordinary Cauchy surface. The common modulus row M is the extra uniform coordinate: it binds all admitted names to one displayed tail budget before the terminal seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name only this packet. Since no carrier coordinate stores quotient equality, selected limits, host extensionality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the NameCert surface. □

Theorem 7934.3 (Uniform Cauchy sequence space nonescape)**.** *No accepted* $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ *packet exports data outside* $S, R, D, Q, M, E, H, C, P, N$*. In particular, a* $\mathsf{Real}^{\uparrow}$ *or completion consumer cannot read a selected limit, quotient sequence equality, host sequence extensionality, countable-choice modulus, trichotomy, or ambient completeness principle from the common-modulus surface.*

Proof. Inspect the carrier. Its only analytic coordinates are the finite stream schedule, regular rational readback, dyadic tolerance ledger, sequence-space row, common modulus, and terminal seal.

Apply Definition 7934.2. Every local obligation is a projection of those rows or a structural transport, replay, provenance, or naming row for the same packet.

Thus any downstream $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reaches the seal only through the displayed common-modulus route. The packet contains no coordinate from which a selected limit, quotient equality, host extensionality, countable-choice modulus, trichotomy, or ambient completeness object could be exported. □

Falsifiable prediction: An accepted $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ read is refuted if a completion or Real consumer obtains a common Cauchy tail budget without displaying the StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchySequenceSpace row, common modulus, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, dyadic-rat-core, cauchy-sequence-space: none alone is bijective with $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ because the packet binds finite schedules,

regular rational readbacks, dyadic tolerance accounting, an underlying Cauchy sequence-space row, one common modulus, terminal sealing, transport, replay, provenance, and local NameCert data in a single common-modulus surface.

Lean4 (statement) `BEDC.Derived.UniformCauchySequenceSpaceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchySequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, CauchySequenceSpace rows, one common modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchySequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7934.1, Definition 7934.2, and Definition 7934.3 seed the common-modulus Cauchy name-space surface over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, CauchySequenceSpaceUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchySequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected limits, host sequence extensionality, countable-choice moduli, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite schedule exposure, regular rational readback, dyadic tolerance accounting, Cauchy sequence-space admission, common modulus exactness, seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.UniformCauchySequenceSpaceUp.

Chapter 7935

A Concrete Naming Certificate for $\mathsf{LocatedLimsup}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedLimsup}^{\uparrow}$ packet records constructive upper-limit data for a displayed bounded real sequence by exposing finite tail windows, regular-rational readback, located upper-envelope tests, and the terminal real seal before any completion-facing consumer reads a limit-superior claim. It sits over the terminal real-name surface of chapter 38, the regular sequence readback of chapter 301, the located supremum handoff of chapter 1427, and the bounded-sequence upper-envelope packet of chapter 7101. The packet names located limsup data for Real and Completion consumers; it does not assert classical subsequence compactness, select a maximizing subsequence, quotient real streams, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7935.1 (Located limsup carrier)**.** *A* $\mathsf{LocatedLimsup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, W, Q, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded real-sequence source row whose upper-tail behaviour is inspected.*
2. *T is the finite tail-index ledger, recording the displayed tail starts and successor rows available to the packet.*
3. *W is the finite window row, listing the inspected sequence entries and rational comparison requests inside the displayed tails.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the windowed tail data.*
5. *U is the located upper-envelope row, answering each displayed rational cut request by an upper-tail bound or a lower displayed occurrence before a limsup read is exported.*
6. *R is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row for the exposed tail-envelope surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row naming the upper-limit read after S, T, W, Q, U, R have been displayed.*
8. *H records componentwise* hsame *transport for source, tail, window, readback, located-envelope, handoff, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a rational upper-limit request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedLimsup}^{\uparrow}$, $\mathsf{BoundedRealLimsup}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLimsup}^{\uparrow}}$ *row.*

The carrier has no coordinate for classical subsequence compactness, a selected subsequence, quotient stream equality, a host limsup operator, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7935.2 (Located limsup NameCert obligations)**.** *For every accepted* $\mathsf{LocatedLimsup}^{\uparrow}$ *carrier* $L = (S, T, W, Q, U, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedLimsup}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded source, finite tail ledger, finite windows, regular-rational readback, located upper-envelope row, located-supremum handoff, terminal real seal, componentwise transport, displayed replay, provenance, and local naming row. Every Real or Completion consumer must read* S, T, W, Q, U, R *before the seal* E*, and every located upper-limit request remains inside the displayed finite rows named by* H, C, P, N.

Lean4 (statement) `BEDC.Derived.LocatedLimsupUp`

Proof. Project the carrier of Definition 7935.1. The source, tail, window, readback, located-envelope, handoff, seal, transport, replay, provenance, and local naming rows are the only available coordinates.

The tail discipline starts with S and T. A consumer may inspect only the tail starts and successor rows displayed by T, and the inspected entries are exactly the entries listed in W.

The analytic readback then passes through Q, U, R, E. The regular-rational row Q reads the finite windows, the located-envelope row U answers each displayed rational cut request, the $\mathsf{LocatedSupremum}^{\uparrow}$ handoff row R packages the upper-envelope surface, and the $\mathsf{Real}^{\uparrow}$ seal E is exposed only after those rows are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since no coordinate stores a selected subsequence, quotient stream, host limsup, countable-choice selector, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{LocatedLimsup}^{\uparrow}$ Real-facing read factors through a bounded real-sequence source, finite tail ledger, finite windows, RegSeqRat readback, a located upper-envelope row, a LocatedSupremum handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: real, regseqrat, locatedsupremum, bounded-real-limsup: no listed sibling supplies the whole packet $L = (S, T, W, Q, U, R, E, H, C, P, N)$, because each sibling contributes only the terminal real seal, regular readback, located supremum handoff, or bounded upper-envelope route while this chapter binds them into one located upper-limit carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLimsup}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedLimsup}^{\uparrow}$ packet is generated from a bounded real-sequence source, a finite tail-index ledger, finite windows, RegSeqRat readback, a located upper-envelope row, a LocatedSupremum handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedLimsup}^{\uparrow}}$, seed)

Scope closed: Definition 7935.1 and Definition 7935.2 seed the constructive located upper-limit route for bounded real-sequence data and completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedLimsup}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close classical subsequence compactness, selected subsequences, quotient stream equality, a host limsup operator, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tail-window coverage, RegSeqRat readback, located upper-envelope exactness, LocatedSupremum handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedLimsupUp.

Chapter 7936

A Concrete Naming Certificate for $\mathsf{BishopRealLocatedExtrema}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealLocatedExtrema}^{\uparrow}$ packet records the finite BEDC surface by which a located bounded family of constructive real names exposes both one-sided extrema reads. It sits beside the upper route of chapter 6496, the lower route of chapter 1452, and the regular rational readback of chapter 301. The packet joins those sibling rows through one shared window and one terminal $\mathsf{Real}^{\uparrow}$ seal discipline; it does not turn the family into an arbitrary subset completeness principle.

Definition 7936.1 (BishopRealLocatedExtrema carrier)**.** *A* $\mathsf{BishopRealLocatedExtrema}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, W, S, I, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the located bounded-family row, listing the inspected real-name sources, their boundedness witnesses, and the family membership observations available to the extrema request.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row shared by the family entries, the supremum candidate, the infimum candidate, and the regular rational readback.*
3. *S is the supremum row, exposing the upper-bound and lower-hit handoff consumed from the* $\mathsf{RealLocatedSupremum}^{\uparrow}$ *sibling surface.*
4. *I is the infimum row, exposing the lower-bound and upper-hit handoff consumed from the* $\mathsf{LocatedInfimum}^{\uparrow}$ *sibling surface.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that reads the same window W before either extremum row reaches the terminal real seal.*
6. *Q is the paired* $\mathsf{Real}^{\uparrow}$ *seal row, with separate supremum and infimum endpoints but a shared nonescape ledger for the displayed family.*
7. *H records componentwise* hsame *transport for $F, W, S, I, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay from an extrema-facing request through the family row, shared window, one-sided handoffs, regular rational readback, real seal, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealLocatedExtrema}^{\uparrow}$, $\mathsf{RealLocatedSupremum}^{\uparrow}$, $\mathsf{LocatedInfimum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealLocatedExtrema}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $F, W, S, I, R, Q, H, C, P, N$. It has no coordinate for arbitrary subset completeness, trichotomy, selected host extrema, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7936.2 (BishopRealLocatedExtrema NameCert obligations)**.** *For every accepted* BishopRealLocatedExtr *carrier*

$$E = (F, W, S, I, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealLocatedExtrema}^{\uparrow}}$ *surface consists exactly of carrier admission through $F, W, S, I, R, Q, H, C,$ shared-window exposure through W, supremum handoff through S, infimum handoff through I,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealLocatedExtrema}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealLocatedExtrema}^{\uparrow}$ packet is generated from a located bounded-family row, one synchronized StreamName window, one supremum handoff, one infimum handoff, RegSeqRat readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealLocatedExtrema}^{\uparrow}}$, seed)

Scope closed: Definition 7936.1 and Definition 7936.2 seed the finite extrema packet that joins one supremum row and one infimum row over a shared located bounded-family window.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealLocatedExtrema}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary subset completeness, trichotomy, selected host extrema, quotient equality of real names, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exposure, supremum and infimum handoffs, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealLocatedExtremaUp.

Chapter 7937

A Concrete Naming Certificate for $\mathsf{CompactDerivativeBound}^{\uparrow}$

Following Bishop and Bridges on differentiable functions over compact intervals, the $\mathsf{CompactDerivativeBound}^{\uparrow}$ packet records the finite derivative-control surface that a compact-uniform-continuity consumer may inspect before a Lipschitz or mean-value packet reads any bound. It sits over the compact metric source of chapter 129, the continuous-map rows of chapter 130, the derivative carrier of chapter 122, the uniform-modulus route of chapter 369, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a compact interval derivative-bound interface as displayed rows, not as a general mean-value theorem or a host compactness principle.

Definition 7937.1 (CompactDerivativeBound carrier)**.** *A* $\mathsf{CompactDerivativeBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, G, U, N_r, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact interval source row, or an equivalent compact interval source row already admitted through the compact metric NameCert surface.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed function whose derivative is being bounded.*
3. *G is the finite* $\mathsf{Derivative}^{\uparrow}$ *observation row, listing the derivative endpoints and local difference-quotient windows visible to the packet.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row that synchronizes the compact interval source with the continuous-map and derivative observation windows.*
5. *N_r is the finite compact-net radius row, storing the selected net radii, local derivative-control windows, and rational comparison budgets.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those rational budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed derivative bound.*
8. *H records componentwise* hsame *transport for $K, F, G, U, N_r, R, E, C, P, Q$.*
9. *C records displayed* Cont *replay from a derivative-bound request through the compact source, continuous-map row, derivative row, uniform-modulus row, compact-net radius row, regular rational readback, real seal, transport, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactDerivativeBound}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{CompactDerivativeBound}^{\uparrow}}$ *row.*

The carrier has no coordinate for a general mean-value theorem, open-cover compactness, choice-selected extrema, quotient equality of functions, a host derivative norm, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7937.2 (CompactDerivativeBound NameCert obligations)**.** *For every accepted* CompactDerivativeBound *carrier*

$$D = (K, F, G, U, N_r, R, E, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{CompactDerivativeBound}^{\uparrow}}$ *surface consists exactly of compact interval source admission through K, continuous-map exposure through F, derivative-observation exposure through G, uniform-modulus synchronization through U, finite compact-net radius accounting through N_r,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through Q.*

Closure status for $\mathsf{NameCert}_{\mathsf{CompactDerivativeBound}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactDerivativeBound}^{\uparrow}$ packet is generated from compact interval or CompactMetric source rows, ContinuousMap rows, Derivative observation rows, UniformModulus synchronization, finite compact-net radius rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactDerivativeBound}^{\uparrow}}$, seed)

Scope closed: Definition 7937.1 and Definition 7937.2 seed the finite derivative-bound packet consumed by compact-uniform-continuity, Lipschitz, and mean-value-facing routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactDerivativeBound}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a general mean-value theorem, classical compactness, open-cover compactness, choice-selected extrema, quotient equality of functions, a host derivative norm, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, continuous-map exposure, derivative-observation exposure, uniform-modulus synchronization, finite compact-net radius accounting, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactDerivativeBoundUp.

Chapter 7938

A Concrete Naming Certificate for $\mathsf{LocatedSequenceSubspace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedSequenceSubspace}^{\uparrow}$ packet records the constructive carrier for located subspaces of real sequences as finite NameCert data. It sits after the located real sequence surface of chapter 1984, the regular rational readback of chapter 301, the stream naming discipline of chapter 126, and the terminal Real seal of chapter 38. The packet names a sequence-subspace membership classifier with finite-window stability and transport rows, without importing a host powerset of sequences, a quotient completion, a selected dense enumeration, or a Heine–Borel compactness principle.

Definition 7938.1 (LocatedSequenceSubspace carrier)**.** *A* $\mathsf{LocatedSequenceSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, M, W, T, L, H, C, P, N)$$

whose rows have the following roles:

1. *A is the admitted located real sequence ambient row, routed through the sibling located-sequence carrier.*
2. *M is the located membership classifier for the displayed subspace.*
3. *W is the finite-window stability row for truncations and regular rational readback.*
4. *T is the subspace transport row preserving membership under componentwise sequence equality.*
5. *L is the ledger boundary recording the non-escape of membership reads from the displayed rows.*
6. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceSubspace}^{\uparrow}}$ *naming.*

The carrier has no coordinate for ambient powersets of sequence spaces, quotient equality of Cauchy representatives, selected representatives of Real values, countable choice, or compactness data.

Theorem 7938.2 (LocatedSequenceSubspace NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequenceSubspac}$ *carrier $S = (A, M, W, T, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceSubspace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the ambient located real sequence row A together with provenance P and local naming N;*

2. *located membership classifier exposure through M, read only after the stream and Real-seal rows cited above are visible;*
3. *finite-window stability through W, using* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rather than host convergence;*
4. *subspace transport through T, H, C, preserving the displayed classifier under componentwise sequence equality and replay;*
5. *ledger non-escape through L, excluding host powersets, quotient completion, selected representatives, countable choice, and Heine–Borel compactness.*

Proof. Project the carrier of Definition 7938.1. The row A supplies the admitted ambient sequence surface, while M is the only membership classifier made public.

The finite-window row W routes every stability read through stream windows and regular rational readback before the Real seal is consumed. Thus no host convergence or quotient-completion row can replace the displayed finite data.

Transport is then componentwise through T, H, C. These rows preserve the same membership classifier under accepted sequence equality and replay, while P, N source and name the finite packet.

Finally the ledger row L records the boundary of the export. Since the carrier contains no coordinate for powersets, selected representatives, countable choice, or compactness, the listed five rows exhaust the local NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceSubspace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSequenceSubspace}^{\uparrow}$ packet is generated from an ambient located real sequence row, a located membership classifier, finite-window stability, subspace transport, ledger non-escape, componentwise transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequenceSubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7938.1 and Definition 7938.2 name the carrier and its five-row obligation surface for located subspaces of real sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequenceSubspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close separability, compactness, choice principles, quotient completion, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located membership, finite-window stability, subspace transport, and ledger non-escape routes for BEDC.Derived.LocatedSequenceSubspaceUp.

Chapter 7939

A Concrete Naming Certificate for CauchyCondensationFiniteWindow$^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the CauchyCondensationFiniteWindow$^{\uparrow}$ packet isolates the finite inspection window that sits between the broad condensation packet of chapter 7794 and the tail-lock consumer of chapter 6052. It records a source series window, dyadic block schedule, comparison ledger, readback route, and local naming surface before any completion-facing consumer may read a tail-lock seal.

Definition 7939.1 (Cauchy condensation finite-window carrier)**.** *A* CauchyCondensationFiniteWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, L, M, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-series window, recording the inspected monotone nonnegative* RealSeries$^{\uparrow}$ *terms inherited from the condensation packet.*
2. *D is the dyadic block schedule, recording finite powers-of-two cut points, block endpoints, and block lengths as* DyadicRatCore$^{\uparrow}$ *data.*
3. *L is the finite block ledger, storing the displayed source block sums and condensed representative terms on the same schedule.*
4. *M is the monotone comparison row, recording the endpoint comparisons that bind each source block to its condensed representative.*
5. *T is the tail-lock handoff row, recording exactly the finite tail data handed to* CauchyCondensationTailLock$^{\uparrow}$*.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced from S, D, L, M, T.*
7. *E is the* Real$^{\uparrow}$ *seal row available only after the regular readback has been displayed.*
8. *H stores componentwise* hsame *transport for the source window, dyadic schedule, block ledger, comparison row, tail-lock handoff, regular readback, real seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a finite precision request through $S, D, L, M, T, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* CauchyCondensationFiniteWindow$^{\uparrow}$*,* CauchyCondensationPacket$^{\uparrow}$*,* CauchyCondensationTailLock$^{\uparrow}$*,* RealSeries$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}}$ *row.*
The carrier has no coordinate for a quotient of term streams, a selected infinite sum, countable choice, order trichotomy outside the displayed window, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7939.2 (Cauchy condensation finite-window NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}$ *carrier* $W = (S, D, L, M, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}}$ *surface consists exactly of source-window exposure by* S*, dyadic schedule exposure by* D*, finite block-ledger exposure by* L*, monotone comparison exposure by* M*, tail-lock handoff by* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback by* R*,* $\mathsf{Real}^{\uparrow}$ *sealing by* E*, componentwise transport by* H*, displayed replay by* C*, provenance by* P*, and local naming by* N*.*

Proof. Project the accepted packet. The analytic rows S, D, L, M, T are the entire finite condensation window: they name the inspected source terms, the dyadic block cuts, the block ledger, the monotone endpoint comparisons, and the tail data exported to the tail-lock consumer.

The readback rows are downstream of that window. The row R repacks only the displayed finite data as $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the seal row E is reachable only through R.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the carrier has no additional analytic coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7939.3 (Cauchy condensation finite-window tail-lock handoff)**.** *Every accepted* CauchyCondensationFinite *carrier hands condensation data to* $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ *only through the displayed route*

$$S, D, L, M \longmapsto T \longmapsto R \longmapsto E.$$

Thus a tail-lock consumer may read the source series window, dyadic block schedule, finite block ledger, monotone comparison row, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local NameCert rows, but it receives no quotient stream equality, selected sum, choice functional, trichotomy oracle, or ambient completeness field.*

Proof. Begin with a tail-lock read. By Definition 7939.1, the only rows available before the handoff are the source window S, dyadic schedule D, finite block ledger L, and monotone comparison row M.

Read T next. It is the unique row whose purpose is to present the finite condensation window to $\mathsf{CauchyCondensationTailLock}^{\uparrow}$.

The regular readback and seal are downstream. The row R consumes the displayed handoff, and E is read only after R. Transport, replay, provenance, and local naming preserve this order, so no unlisted infinite or classical principle can supply an alternate tail-lock route. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}$ consumer that reaches a tail-lock or $\mathsf{Real}^{\uparrow}$ seal without first reading S, D, L, M, T, R refutes the finite-window carrier.

Independence witness: cauchy-condensation-packet, cauchy-condensation-tail-lock, realseries, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}$ because this packet binds the source series window, dyadic block schedule, finite block ledger, monotone comparison row, tail-lock handoff, regular readback, real seal, transport, replay, provenance, and local NameCert in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationFiniteWindow}^{\uparrow}}$ **.**

Constructive story: A CauchyCondensationFiniteWindow$^{\uparrow}$ packet is generated from a source series window, dyadic block schedule, finite block ledger, monotone comparison row, tail-lock handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCondensationFiniteWindow}^{\uparrow}}$, seed)

Scope closed: Definition 7939.1, Definition 7939.2, and Definition 7939.3 seed the finite condensation window between the packet and tail-lock surfaces.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCondensationFiniteWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host infinite sums, quotient stream equality, selected sums, order trichotomy, countable choice, bridge checking, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked source-window exposure, dyadic scheduling, finite block-ledger exactness, monotone comparison stability, tail-lock handoff, regular readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationFiniteWindowUp.

Chapter 7940

A Concrete Naming Certificate for $\mathsf{LocatedCutGap}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCutGap}^{\uparrow}$ packet names the finite gap surface between located lower and upper cut evidence before any consumer reads a completed real. It sits beside the located bracket completion of chapter 5554, the Dedekind-cut completion interface of chapter 2138, the dyadic endpoint ledger of chapter 291, the finite stream interface of chapter 126, the regular rational readback row of chapter 301, and the terminal real seal of chapter 38. The chapter records a gap carrier for finite readback only; it does not assert Dedekind completeness, quotient cuts, selected suprema, or a construction of $\mathsf{Real}^{\uparrow}$ from primitive cut data.

Definition 7940.1 (LocatedCutGap carrier)**.** *A* $\mathsf{LocatedCutGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, A, B, Q, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located-cut observation row. It records only finitely displayed lower cells, lower endpoint witnesses, and lower refinement requests.*
2. *U is the upper located-cut observation row. It records only finitely displayed upper cells, upper endpoint witnesses, and upper refinement requests.*
3. *A is the lower–upper admission row. It records that each displayed lower request and upper request belongs to the same located-cut comparison packet.*
4. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger. It stores the common dyadic scale on which the displayed lower and upper endpoint rows can be compared.*
5. *Q is the finite gap row. It records, for each displayed comparison request, whether the packet supplies a separated dyadic gap, an overlap witness, or a refinement demand back into B.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes only the requested finite observations of L, U, B, Q.*
7. *S is the* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *route row. It packages the displayed lower row, upper row, dyadic bracket ledger, and gap row as a located-completion-facing read.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational Cauchy approximants read from the displayed window and gap data.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, U, A, B, Q, W, S, R have been displayed.*

10. *H records componentwise* hsame *transport for the lower row, upper row, admission row, dyadic bracket ledger, gap row, stream window, located-completion route, regular readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$L, U, A, B, Q \longmapsto W, S, R \longmapsto E, H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{LocatedCutGap}^{\uparrow}$, $\mathsf{LocatedCutCompletion}^{\uparrow}$, $\mathsf{DedekindCutCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutGap}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying both located cut sides, their shared admission row, a common dyadic bracket ledger, the finite gap row, the stream window, the located-completion route, the regular rational readback row, the terminal real seal, and the structural rows. It has no coordinate for quotient cut equality, arbitrary suprema, trichotomy, selected limits, or an ambient completeness theorem.

Theorem 7940.2 (LocatedCutGap NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCutGap}^{\uparrow}$ *carrier* $G = (L, U, A, B, Q, W, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCutGap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower located-cut row, upper located-cut row, common admission row, dyadic bracket ledger, gap row, stream window, located-completion route, regular rational readback, real seal, transport, replay, provenance, and local naming row;*

2. *two-sided gap discipline, requiring every comparison-facing read to pass through L, U, A, B, Q before it can be exposed to a stream or completion consumer;*

3. *dyadic common-scale stability, requiring lower and upper refinements to remain inside the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger B;*

4. *finite-window readback, requiring W to determine which displayed rows are opened before the* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *route S or the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is visible;*

5. *classifier transport, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the same row route;*

6. *ledger exhaustion, requiring P and N to name exactly the displayed packet and no Dedekind completeness principle, quotient cut, selected supremum, trichotomy, or hidden real representative.*

No $\mathsf{LocatedCutGap}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 7940.1. The accepted packet exposes the lower row, upper row, common admission row, dyadic bracket ledger, gap row, stream window, located-completion route, regular rational readback, real seal, and structural rows.

Two-sided gap discipline follows because Q is the only row relating displayed lower and upper requests after they have entered the same admission row A and common dyadic scale B. Without Q, a consumer has two finite located-cut sides but no displayed gap surface.

The common-scale condition is forced by B. Lower and upper refinements remain in the same dyadic ledger, so a read cannot replace the packet by independent cut refinements.

The window and readback rows then expose only finitely requested observations. The rows S and R read the same displayed gap data through located completion and regular rational Cauchy routes before the terminal real seal E is available.

Finally, H, C, P, N transport, replay, source, and name the displayed route. Since the excluded principles have no carrier coordinate, the local certificate cannot export them as hidden obligations. □

Theorem 7940.3 (LocatedCutGap Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCutGap}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedCutCompletion}^{\uparrow}$, $\mathsf{DedekindCutCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cor

The terminal seal E is reached only after the lower located-cut row, upper located-cut row, common admission row, dyadic bracket ledger, finite gap row, stream window, located-completion route, and regular rational readback row have been displayed. The handoff therefore records a finite readback route between located cuts and the real-name seal, not Dedekind completeness or quotient equality of cuts.

Proof. Start from the two located cut sides L, U and the common admission row A. The dyadic bracket ledger B fixes the shared endpoint scale on which the finite gap row Q can be read.

The stream row W exposes only bounded observations of those rows. The located-completion route S then reads the lower side, upper side, bracket ledger, and gap row as the finite surface already used by chapter 5554; the Dedekind-cut completion sibling of chapter 2138 is reached only as a comparison interface for displayed cut-shaped evidence.

The regular rational row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback from the same windowed data, while chapter 291, chapter 126, and chapter 301 provide the dyadic, stream, and regular Cauchy carrier surfaces used by the route.

Only after these finite rows are visible does the terminal $\mathsf{Real}^{\uparrow}$ seal E become readable, as in chapter 38. The structural rows H, C, P, N preserve and name that handoff, so the packet has no projection to Dedekind completeness, quotient cut equality, selected suprema, trichotomy, or hidden representatives. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCutGap}^{\uparrow}$ Real or Completion read factors through displayed lower located-cut, upper located-cut, common admission, dyadic bracket, finite gap, stream window, located-completion, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: located-cut-completion, dedekind-cut-completion, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedCutGap}^{\uparrow}$ is not bijective to any listed sibling because it carries the finite lower–upper gap row and common admission row before the located-completion, regular-Cauchy, and Real-seal handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutGap}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7940.1, Definition 7940.2, and Definition 7940.3 seed the finite located lower–upper gap surface between located cuts and regular Cauchy real-name readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient cuts, selected suprema, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-sided gap discipline, dyadic common-scale stability, finite-window readback, classifier transport, ledger exhaustion, and Real seal handoff for BEDC.Derived.LocatedCutGapUp.

Constructive story: A LocatedCutGap$^{\uparrow}$ packet is generated from lower and upper located-cut rows, a common admission row, a DyadicRatCore bracket ledger, a finite gap row, StreamName windows, LocatedCutCompletion routing, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7941

A Concrete Naming Certificate for $\mathsf{SubsequenceModulusSelection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ packet records the constructive act of extracting a displayed subsequence schedule together with the modulus rows used by later Cauchy and compactness arguments. It sits after the common-modulus sequence-space surface of chapter 7934, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names finite subsequence windows and modulus handoffs; it is not a selected limit, countable-choice schedule, trichotomy principle, quotient sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 7941.1 (Subsequence modulus selection carrier)**.** *A* $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, R, D, U, K, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the source sequence names under inspection.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source windows, selected windows, and tail budgets.*
4. *U is the* $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$ *common-modulus row that supplies the ambient family discipline.*
5. *K is the finite subsequence-index schedule, listing only inspected source positions and their order relation inside the displayed windows.*
6. *M is the modulus-selection row assigning each accepted tolerance request to a displayed tail budget in D, S, R, U, K.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the source schedule, readback, dyadic ledger, uniform Cauchy row, subsequence schedule, and selected modulus row have been displayed.*
8. *H records componentwise* hsame *transport for the stream schedule, readback, dyadic ledger, uniform Cauchy row, subsequence schedule, modulus row, terminal seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a tolerance request through* $S, R, D, U, K, M, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{SubsequenceModulusSelection}^{\uparrow}$, $\mathsf{UniformCauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubsequenceModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected limit, an infinite choice function, trichotomy, quotient sequence equality, host sequence equality, or ambient Real completeness.

Theorem 7941.2 (Subsequence modulus selection obligations)**.** *For every accepted* $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ *carrier* $X = (S, R, D, U, K, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SubsequenceModulusSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, finite source-window exposure, regular rational readback, dyadic tolerance accounting, common-modulus sequence-space admission, finite subsequence-index scheduling, selected-modulus exactness, terminal Real seal handoff, componentwise transport, displayed replay, package provenance, and local naming. Every compactness-facing or completion-facing read must pass through*

$$S, R, D, U, K, M, E, H, C, P, N$$

and through no selected limit, countable-choice schedule, trichotomy, quotient sequence equality, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 7941.1. The analytic rows are exactly S, R, D, U, K, M, E, and the structural rows are exactly H, C, P, N.

The source schedule S exposes only finite inspected windows, while R records regular rational readback and D records the tolerance cells used by the comparison. The uniform sequence-space row U imports the common-modulus discipline of chapter 7934.

The subsequence-index row K and modulus row M are the additional selection data. They name displayed positions and displayed tail budgets before the terminal seal E can be consumed; neither row stores an infinite selector or a completed limit.

Finally H, C, P, N transport, replay, source, and name only this finite packet. Since no carrier coordinate stores countable choice, trichotomy, quotient equality, host sequence equality, a selected limit, or ambient completeness, the displayed rows exhaust the NameCert surface. □

Theorem 7941.3 (Subsequence modulus selection nonescape)**.** *No accepted* $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ *packet exports data outside* $S, R, D, U, K, M, E, H, C, P, N$*. In particular, a compactness-facing, Cauchy-extraction-facing, or* $\mathsf{Real}^{\uparrow}$ *consumer cannot read a selected limit, countable-choice schedule, trichotomy principle, quotient sequence equality, host sequence equality, or ambient completeness theorem from the subsequence/modulus surface.*

Proof. Inspect the carrier. Its only analytic coordinates are the finite source schedule, regular rational readback, dyadic tolerance ledger, uniform Cauchy sequence-space row, subsequence-index schedule, modulus-selection row, and terminal Real seal.

Apply Definition 7941.2. Every local obligation is a projection of those rows or a structural transport, replay, provenance, or naming row for the same packet.

Thus any downstream compactness, Cauchy extraction, completion, or $\mathsf{Real}^{\uparrow}$ consumer reaches the seal only through the displayed subsequence/modulus route. The packet contains no coordinate from which a selected limit, infinite selector, trichotomy branch, quotient equality, host sequence equality, or ambient completeness object could be exported. □

Falsifiable prediction: An accepted $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ read is refuted if a compactness or Real consumer obtains a subsequence modulus without displaying the StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, UniformCauchySequenceSpace row, finite subsequence-index schedule, selected modulus row, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: uniform-cauchy-sequence-space, streamname, regseqrat, dyadic-rat-core, real: none alone is bijective with $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ because the packet binds finite source windows, regular rational readbacks, dyadic tolerance accounting, a common-modulus sequence-space row, a finite subsequence-index schedule, a selected modulus row, terminal sealing, transport, replay, provenance, and local NameCert data in one surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SubsequenceModulusSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{SubsequenceModulusSelection}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, UniformCauchySequenceSpace rows, finite subsequence-index schedules, selected modulus rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubsequenceModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7941.1, Definition 7941.2, and Definition 7941.3 seed the constructive subsequence/modulus selection surface over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, UniformCauchySequenceSpaceUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubsequenceModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, countable-choice schedules, trichotomy, quotient sequence equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite source-window exposure, regular rational readback, dyadic tolerance accounting, uniform Cauchy sequence-space admission, subsequence-index scheduling, modulus exactness, Real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.SubsequenceModulusSelectionUp.

Chapter 7942

A Concrete Naming Certificate for $\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}$

Following Bishop and Bridges and the classical Arzela–Ascoli theorem, the $\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}$ packet records the finite dyadic-net modulus surface by which an equicontinuous family of maps on a compact source is read before a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ value seal is consumed. It sits after the Arzela–Ascoli carrier of chapter 412, the compact source of chapter 60, the continuous-map graph rows of chapter 130, the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed finite-net modulus certificate; it does not assert open-cover compactness, select a subsequence, quotient function-space equality, use countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7942.1 (Dyadic Arzela–Ascoli modulus carrier)**.** *A* $\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, D, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row, exposing only the compact object through the finite-net boundary already named by the Arzela–Ascoli packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, listing the maps whose values may be inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic radii, mesh scales, comparison cells, and half-radius splits.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the sampled source points and map-value observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding the displayed stream windows and dyadic tolerance cells to regular rational value observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for value-facing consumers, exposed only after K, F, D, S, R have been displayed.*
7. *M is the finite dyadic-net modulus row, assigning each displayed tolerance request to a displayed mesh scale and finite inspection window.*
8. *H records componentwise* hsame *transport for the compact source, map family, dyadic ledger, stream windows, readback, real seal, modulus row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$K, F, D, S, R \longmapsto E, M.$$

10. *P is the* Pkg *provenance over* DyadicArzelaAscoliModulus$^{\uparrow}$, ArzelaAscoli$^{\uparrow}$, Compact$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{DyadicArzelaAscoliModulus}^{\uparrow}}$ *row naming exactly this finite modulus packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open cover, selected subsequence, quotient function space, countable-choice selector, detached real representative, or ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.DyadicArzelaAscoliModulusUp.DyadicArzelaAscoliModulusTasteGate single carrier alignment`

Theorem 7942.2 (Dyadic Arzela–Ascoli modulus NameCert obligations)**.** *For every accepted* DyadicArzelaAscoliModulus$^{\uparrow}$ *carrier*

$$A = (K, F, D, S, R, E, M, H, C, P, N),$$

the local NameCert$_{\text{DyadicArzelaAscoliModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source visibility, continuous-map family admission, dyadic tolerance visibility, finite stream-window admission, regular rational readback, terminal real-name sealing, finite dyadic-net modulus assignment, componentwise transport, displayed replay, provenance, and local naming. Every Arzela–Ascoli or compact-uniform consumer must pass through*

$$K, F, D, S, R, E, M, H, C, P, N$$

and through no selected subsequence, quotient function-space equality, open-cover compactness proof, countable-choice selector, host real equality, or ambient Real$^{\uparrow}$ *completeness row.*

Proof. Project the carrier rows of Definition 7942.1. The rows K, F, D, S, R, E, M expose the compact source, map family, dyadic tolerance ledger, finite observation windows, regular rational readback, real seal, and modulus assignment before any Arzela–Ascoli consumer is available.

Read the analytic route in order. The compact row K supplies only the displayed finite-net boundary, the map row F supplies only displayed continuous-map observations, and the dyadic row D fixes the tolerance scale inspected by S.

The readback row R binds those windows to regular rational observations, and the seal row E is downstream of the same route. The modulus row M therefore records a finite mesh-scale assignment justified by K, F, D, S, R rather than by an external compactness or selection principle.

The structural rows H, C, P, N preserve, replay, source, and locally name this route. Since the carrier has no coordinate for the excluded subsequence, quotient, cover, choice, host-equality, or completeness data, the listed rows exhaust the local NameCert surface. □

Theorem 7942.3 (Dyadic finite-net induction route)**.** *Let*

$$A = (K, F, D, S, R, E, M, H, C, P, N)$$

be an accepted DyadicArzelaAscoliModulus$^{\uparrow}$ *carrier. Every modulus read from A is obtained by finite induction over the displayed dyadic net rows in S: the base read consumes the first displayed net*

point through D, R, E, and each induction step appends one displayed point using the same D-scale and the same $\mathsf{ContinuousMap}^{\uparrow}$ *family row F. The resulting route factors through*

$$\mathsf{Compact}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and then through the modulus, transport, replay, provenance, and local naming rows M, H, C, P, N.

Proof. Start from the displayed finite net inside S. The empty or first-point case has only the compact source row K, the map family row F, and the dyadic tolerance row D available before any value is read.

For the induction step, append one displayed net point. Its map-value observation is read through the same $\mathsf{ContinuousMap}^{\uparrow}$ row F, the same dyadic scale in D, and the same $\mathsf{StreamName}^{\uparrow}$ window discipline before R translates the observation into regular rational readback.

After each step, the terminal value seal E remains downstream of D, S, R. The modulus row M can therefore record only the mesh scale and finite window already displayed by the induction.

Finally, H, C, P, N transport, replay, source, and name the finite induction route. No open-cover proof, selected subsequence, quotient function-space equality, countable-choice selector, host real equality, or ambient Real-completeness row is present in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}$ packet is generated from a compact source row, continuous-map family row, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, finite dyadic-net modulus assignment, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7942.1, Definition 7942.2, and Definition 7942.3 seed the finite dyadic-net modulus route over compact sources, continuous-map families, dyadic tolerances, stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicArzelaAscoliModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subsequences, quotient function-space equality, open-cover compactness, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source visibility, continuous-map family admission, dyadic tolerance visibility, finite window admission, regular rational readback, Real seal handoff, modulus assignment, transport, replay, provenance, and local naming for BEDC.Derived.DyadicArzelaAscoliModulusUp.

Chapter 7943

A Concrete Naming Certificate for $\mathsf{RieszProduct}^{\uparrow}$

Following F. Riesz and classical harmonic analysis, the $\mathsf{RieszProduct}^{\uparrow}$ packet records a finite trigonometric-product window before any Fourier-facing or Real-facing consumer may read a coefficient or scalar seal. It sits over the terminal real-name boundary of chapter 38, the regular rational readback route of chapter 301, the finite observation schedule of chapter 126, the dyadic tolerance ledger of chapter 291, and the positive summability-kernel neighbour of chapter 1939. The packet names only finite factor data, finite coefficient windows, dyadic budgets, regular readback, Real sealing, transport, replay, provenance, and local naming rows; it does not claim measure-theoretic convergence, an infinite product object, Haar measure, selected representatives, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7943.1 (Riesz product carrier)**.** *A* $\mathsf{RieszProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, A, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite factor ledger, listing the retained frequencies, coefficients, signs, and normalising constants of the trigonometric factors under inspection.*
2. *A is the admissibility row, recording the displayed positivity and boundedness checks for the retained factors.*
3. *W is the coefficient-window row, naming the finite Fourier modes and product-expansion terms actually read by the consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger for factor coefficients, finite products, coefficient sums, and positivity budgets.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after F, A, W, D have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for scalar coefficient or normalisation reads.*
7. *H records componentwise* hsame *transport for the factor ledger, admissibility row, coefficient windows, dyadic arithmetic, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a harmonic-analysis request through F, A, W, D, Q, E.*

9. *P is the* Pkg *provenance over* $\mathsf{RieszProduct}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszProduct}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, A, W, D, Q, E, H, C, P, N$. It has no coordinate for an infinite product, a measure-theoretic weak limit, Haar integration, quotient equality of functions, selected representatives, or ambient Real completeness.

Theorem 7943.2 (Riesz product NameCert obligations)**.** *For every accepted* $\mathsf{RieszProduct}^{\uparrow}$ *carrier $R = (F, A, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RieszProduct}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed factor ledger, admissibility row, coefficient-window row, dyadic arithmetic ledger, regular readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows. A harmonic-analysis consumer may read a Riesz-product coefficient only after the finite factor data, the admissibility checks, and the dyadic budget are all public in the same packet.*

Proof. Project the carrier of Definition 7943.1. Its visible inventory is exactly $F, A, W, D, Q, E, H, C, P, N$, so every local naming duty begins with finite factor rows and finite coefficient windows rather than with an infinite product.

The product-facing rows are read in order. The factor ledger F supplies retained frequencies and coefficients, the admissibility row A supplies the displayed positivity and boundedness checks, and the window row W fixes the finite Fourier modes and product-expansion terms under inspection.

The scalar endpoint is also explicit. The dyadic ledger D accounts for the arithmetic budget before the regular readback Q and the Real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for an infinite product object, weak-limit measure, Haar integration, quotient function equality, selected representative, or Real completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 7943.3 (Riesz product finite-factor route)**.** *Every Fourier-facing, Fejer-kernel-facing, or Real-facing consumer of an accepted* $\mathsf{RieszProduct}^{\uparrow}$ *packet receives the product read only through the finite route*

$$F, A, W \longmapsto D \longmapsto Q, E \longmapsto H, C, P, N.$$

The route may be compared with the finite harmonic-analysis surface of chapter 1939, but it exports only retained factor rows, displayed coefficient windows, dyadic arithmetic, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Proof. Start with F, A, W. The factor ledger fixes the finite list of retained trigonometric factors, the admissibility row fixes the displayed positivity and boundedness checks, and the coefficient-window row fixes the finite modes available to the consumer.

Next read D. All coefficient arithmetic, finite products, normalising constants, and tolerance comparisons pass through the dyadic ledger, so no read reaches a scalar endpoint from a hidden analytic product.

Only after that finite arithmetic route is displayed can Q and E be consumed. The regular readback and Real seal therefore receive the finite factor route rather than an infinite-product limit, Haar-measure construction, quotient function, selected representative, or ambient completeness theorem.

The structural rows H, C, P, N preserve this factorization under transport, replay, provenance, and local naming. Thus every public product read remains a finite NameCert route over the displayed rows. □

Falsifiable prediction: An accepted $\mathsf{RieszProduct}^{\uparrow}$ consumer that reaches a coefficient, regular readback, or $\mathsf{Real}^{\uparrow}$ seal without displaying the retained factor ledger, admissibility row, coefficient windows, $\mathsf{DyadicRatCore}^{\uparrow}$ budget, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: real, regseqrat, streamname, dyadicratcore, fejer-kernel: no listed sibling supplies the whole packet $R = (F, A, W, D, Q, E, H, C, P, N)$, because each sibling contributes only a terminal seal, regular readback, observation schedule, dyadic arithmetic ledger, or summability-kernel neighbour while this chapter binds finite Riesz-product factor data into one harmonic-analysis carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RieszProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszProduct}^{\uparrow}$ packet is generated from finite trigonometric factor rows, displayed admissibility checks, finite coefficient windows, DyadicRatCoreUp arithmetic budgets, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7943.1, Definition 7943.2, and Definition 7943.3 seed the finite harmonic-analysis route from retained Riesz-product factors to coefficient and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic convergence, infinite product existence, Haar measure, quotient function equality, selected representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, factor admissibility, coefficient-window exposure, dyadic arithmetic, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RieszProductUp.

Chapter 7944

A Concrete Naming Certificate for $\mathsf{RiemannSumMesh}^{\uparrow}$

Following Riemann and Bishop–Bridges, the $\mathsf{RiemannSumMesh}^{\uparrow}$ packet records the finite mesh data needed by constructive Riemann-sum consumers before any integral object is claimed. It sits between the dyadic arithmetic of chapter 291, finite stream windows of chapter 126, rational regular-sequence readback of chapter 301, the real-name seal of chapter 38, and the calculus-facing packet of chapter 3632.

Falsifiable prediction: Deleting the tag-containment row lets a consumer move a sample point outside its cell while keeping the same mesh width; the local classifier must reject that packet because the displayed Riemann sum no longer names a finite tagged partition.

Independence witness: dyadicratcore, streamname, regseqrat, real, calculus: none is bijective because each sibling omits at least one of finite cell ordering, tag containment, dyadic mesh width, sample-value readback, or Real-seal nonescape required by $\mathsf{RiemannSumMesh}^{\uparrow}$.

Definition 7944.1 (RiemannSumMesh carrier)**.** *A* $\mathsf{RiemannSumMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, C, W, T, S, V, R, H, K, P, N)$$

with the following displayed rows:

1. *I is the closed interval row, with dyadic endpoint names read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *C is the finite ordered cell row, listing adjacent subintervals whose endpoints cover I without overlap;*
3. *W is the dyadic mesh-width row, recording each cell width and the displayed maximum mesh bound;*
4. *T is the tag-containment row, assigning one sample tag to each cell and witnessing that the tag lies inside that cell;*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the sampled input names used by the tagged partition;*
6. *V is the* $\mathsf{RegSeqRat}^{\uparrow}$ *sample-value row, giving rational approximants for the sampled function values at the displayed precision;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that real-name reads are only the finite seals needed for the sample values;*

8. *H records componentwise* hsame *transport for $I, C, W, T, S, V, R, K, P, N$;*
9. *K records displayed* Cont *replay from dyadic interval data to cells, mesh widths, tags, stream windows, regular-sequence sample values, and the real seal;*
10. *P is the* Pkg *provenance over* $\mathsf{RiemannSumMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumMesh}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $I, C, W, T, S, V, R, H, K, P, N$. It has no coordinate for a host integral, selected limiting process, quotient function equality, countable-choice sampling schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7944.2 (RiemannSumMesh NameCert obligations)**.** *For the carrier of Definition 7944.1, the local* $\mathsf{NameCert}_{\mathsf{RiemannSumMesh}^{\uparrow}}$ *surface consists exactly of finite interval admission, ordered cell exposure, dyadic mesh-width exposure, tag-containment exactness,* $\mathsf{StreamName}^{\uparrow}$ *finite-window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *sample-value readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same packet*

$$M = (I, C, W, T, S, V, R, H, K, P, N).$$

Proof. Project the carrier of Definition 7944.1. The accepted packet displays the interval, cells, mesh widths, tags, sample windows, sample values, real seal, transport, replay, provenance, and naming rows as $I, C, W, T, S, V, R, H, K, P, N$.

The dyadic rows provide the finite arithmetic support. The interval row I, ordered cell row C, and mesh row W are read through $\mathsf{DyadicRatCore}^{\uparrow}$, so adjacency, coverage, widths, and the maximum mesh bound are finite rational data.

The tag and sample rows provide the calculus-facing support. The containment row T fixes one displayed tag per cell, S restricts the sampled input names to finite $\mathsf{StreamName}^{\uparrow}$ windows, and V reads only the displayed $\mathsf{RegSeqRat}^{\uparrow}$ approximants.

The real seal and structural rows close the packet. The row R bars ambient completeness reads, while H, K, P, N preserve the same finite route under transport, replay, package provenance, and local naming. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 7944.3 (RiemannSumMesh Real-RegSeqRat handoff)**.** *Every accepted* $\mathsf{RiemannSumMesh}^{\uparrow}$ *packet exports its sampled values to* $\mathsf{Real}^{\uparrow}$ *consumers only through the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{Calculus}^{\uparrow} \to \mathsf{BHist} \to \mathsf{hsame} \to \mathsf{Cont} \to \mathsf{Pkg} \to \mathsf{NameC}$$

The route may read finite cell data, tag containment, dyadic mesh widths, stream windows, rational regular-sequence samples, and the real-name seal of the same carrier. It may not export a host integral, select a limiting refinement, quotient functions by extensional equality, introduce countable choice, or use ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the dyadic mesh rows. The interval, cell, and mesh-width coordinates I, C, W determine the finite tagged partition support before a sampled value is read.

Next read the tag and sample rows. The row T keeps each sample tag inside its displayed cell, while S and V restrict evaluation to finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ rational approximants.

Finally pass through the real seal and calculus-facing provenance. The row R supplies only the displayed $\mathsf{Real}^{\uparrow}$ seal, and the replay and package rows K, P, N route the packet toward $\mathsf{Calculus}^{\uparrow}$ without adding a limit selector, integral value, quotient function equality, choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSumMesh}^{\uparrow}$ packet is generated from a dyadic interval row, finite ordered cell row, dyadic mesh-width row, tag-containment row, finite StreamName window row, RegSeqRat sample-value row, Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSumMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7944.1, Definition 7944.2, and Definition 7944.3 specify the finite tagged mesh packet used by constructive calculus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSumMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not construct host integrals, selected refinement limits, quotient function equality, countable-choice sampling schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cell ordering, tag containment, dyadic mesh-width exactness, RegSeqRat sample readback, Real-seal nonescape, transport, replay, provenance, and local naming for the same finite mesh packet.

Chapter 7945

A Concrete Naming Certificate for $\mathsf{CauchyRealEquivalenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealEquivalenceModulus}^{\uparrow}$ packet records equality of constructive real names by a displayed tail-modulus comparison between two regular Cauchy rational representatives. It sits over the dyadic tolerance core of chapter 291, the stream window surface of chapter 126, the regular rational sequence surface of chapter 301, and the real-name seal of chapter 38. The certificate names a finite modulus route for a pair of representatives; it does not identify host sequences, quotient streams, selected limits, real trichotomy, or ambient real completeness.

Definition 7945.1 (Cauchy real equivalence modulus carrier)**.** *A* $\mathsf{CauchyRealEquivalenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, T, D_A, D_B, Q_A, Q_B, E, H, C, P, N)$$

with the following visible rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *representative rows whose stream names are compared by the packet.*
2. *T is the shared dyadic tail-modulus row. For each requested dyadic tolerance, T names a finite tail bound after which the two representatives are read in the same tolerance window.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback rows used to decode the two representative values at the requested dyadic precision.*
4. *Q_A and Q_B are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows exposing only the retained prefixes and tail cells needed by T, D_A, D_B.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal that downstream consumers may read only after the two regular representatives, shared tail modulus, dyadic readbacks, and stream windows have been displayed.*
6. *H records componentwise* hsame *transport for $A, B, T, D_A, D_B, Q_A, Q_B, E$ and the local naming row.*
7. *C records displayed* Cont *replay from the pair of representatives through the shared tail-modulus row and dyadic readbacks to the real-name seal:*

$$A, B, T, D_A, D_B \longmapsto Q_A, Q_B, E.$$

8. *P is the* Pkg *provenance over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{CauchyRealEquivalenceModulus}^{\uparrow}}$ *row.*

The carrier exports a finite modulus witness for equality of the two real names. It has no coordinate for quotient equality of streams, a selected representative of an equivalence class, real trichotomy, a host Cauchy-completion object, or an ambient completeness principle.

Lean4 (statement) `BEDC.Derived.CauchyRealEquivalenceModulusUp`

Theorem 7945.2 (Cauchy real equivalence modulus NameCert obligations)**.** *For every accepted* CauchyRealEquivalenceModulus$^{\uparrow}$ *carrier*

$$M = (A, B, T, D_A, D_B, Q_A, Q_B, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyRealEquivalenceModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* RegSeqRat$^{\uparrow}$ *representatives, the shared dyadic tail-modulus row, the two* DyadicRatCore$^{\uparrow}$ *tolerance readbacks, the two* StreamName$^{\uparrow}$ *finite-window rows, the* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every equality read between the two real names must pass through T, D_A, D_B, Q_A, Q_B before E is public, and no obligation may call on quotient stream equality, chosen representatives, real trichotomy, host completion, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 7945.1. The packet exposes exactly the two regular rational representatives, one shared dyadic tail-modulus row, two dyadic readbacks, two stream windows, the Real-facing seal, and the structural rows.

The equality read is then forced through the displayed modulus route. A requested tolerance is decoded by the dyadic rows D_A, D_B, the shared tail bound is read from T, and only the finite stream windows Q_A, Q_B can expose the representative data at that tolerance.

Transport and replay preserve the same route. The row H carries componentwise hsame evidence for the listed coordinates, while C replays the continuation from the two representative rows through the shared tail-modulus comparison to the seal row E.

Finally inspect P, N. Provenance and local naming record only the displayed carrier rows, so a downstream Real consumer has no projection from which to read quotient stream equality, selected representatives, real trichotomy, host completion, or ambient completeness. □

Closure status for NameCert$_{\text{CauchyRealEquivalenceModulus}^{\uparrow}}$**.**

Constructive story: A CauchyRealEquivalenceModulus$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ representatives, a shared dyadic tail-modulus row, DyadicRatCore$^{\uparrow}$ tolerance readbacks, StreamName$^{\uparrow}$ finite-window rows, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRealEquivalenceModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7945.1 and Definition 7945.2 specify the finite modulus carrier and its local naming obligations over regular Cauchy representatives, dyadic tolerance readbacks, stream windows, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRealEquivalenceModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, chosen representatives, real trichotomy, host Cauchy completion, ambient Real completeness, or a standard bridge.

Upgrade path: Scoped closure requires formal carrier exactness, transport stability, and downstream nonescape for the displayed Cauchy real equivalence modulus route.

Chapter 7946

A Concrete Naming Certificate for $\mathsf{AtsujiSpace}^{\uparrow}$

Following Atsuji's uniformly continuous metric spaces and the Bourbaki–Kelley treatment of metric continuity, the $\mathsf{AtsujiSpace}^{\uparrow}$ packet records the finite BEDC surface on which pointwise continuous maps from a displayed metric source may be handed to uniform-modulus consumers. It sits over the metric carrier of chapter 57, the continuous-map graph rows of chapter 130, the uniform-modulus consumer of chapter 369, the compact-metric comparison source of chapter 129, and the terminal real-name seal of chapter 38. The packet names the Atsuji condition as displayed finite rows; it does not assert open-cover compactness, select a modulus, quotient maps, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7946.1 (AtsujiSpace carrier)**.** *An* $\mathsf{AtsujiSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, F, U, K, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose accepted distance and neighbourhood reads are those of chapter 57.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *testing row: every admitted map from the source is read through the displayed graph and pointwise continuity data of chapter 130.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exposed only after the same metric and continuous-map rows have been replayed.*
4. *K is the compact-metric comparison row, allowing compact-metric Atsuji examples to read the sibling route of chapter 129 without making compactness part of the carrier.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *tolerance and distance-seal row used by metric, continuity, and modulus reads.*
6. *H records componentwise* hsame *transport for the metric, continuous-map, uniform-modulus, compact-comparison, real-seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from an Atsuji request through M, F, U, K, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{AtsujiSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{AtsujiSpace}^\uparrow}$ *row.*

The classifier compares accepted packets only through $M, F, U, K, R, H, C, P, N$. There is no coordinate for open-cover compactness, a chosen modulus, quotient map equality, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.AtsujiSpaceUp.AtsujiSpaceTasteGate single carrier alignment`

Theorem 7946.2 (AtsujiSpace NameCert obligations)**.** *For every accepted* $\mathsf{AtsujiSpace}^\uparrow$ *carrier $A = (M, F, U, K, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AtsujiSpace}^\uparrow}$ *surface consists exactly of carrier admission, metric-source exposure through M, continuous-map testing through F, uniform-modulus handoff through U, compact-metric comparison through K, real tolerance sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by open-cover compactness, a selected modulus, quotient maps, countable choice, a hidden host continuity theorem, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 7946.1. The displayed rows are exactly $M, F, U, K, R, H, C, P, N$, so carrier admission exposes only metric source data, pointwise continuous-map tests, a uniform-modulus handoff, compact-metric comparison, real tolerance sealing, transport, replay, provenance, and naming.

The continuity side starts with M and F. A map is not accepted as a host function; it is read through the metric source and the displayed continuous-map row before any downstream uniform claim is visible.

Next read the handoff and comparison rows. The row U is available only inside the same replay route, while K records the compact-metric sibling comparison without adding open-cover compactness to the Atsuji carrier.

Finally the structural rows H, C, P, N preserve the same packet. Since selected moduli, quotient maps, countable choice, host continuity theorems, and ambient completeness are absent from the carrier, the listed rows exhaust the local surface. □

Theorem 7946.3 (AtsujiSpace uniform-continuity handoff)**.** *Let $A = (M, F, U, K, R, H, C, P, N)$ be an accepted* $\mathsf{AtsujiSpace}^\uparrow$ *carrier. Every consumer that reads an admitted pointwise continuous map from the metric source as uniformly controlled must factor through*

$$\mathsf{Met}^\uparrow \longmapsto \mathsf{ContinuousMap}^\uparrow \longmapsto \mathsf{UniformModulus}^\uparrow$$

with the tolerance seal supplied through $\mathsf{Real}^\uparrow$ *and the compact-metric comparison row available only as K.*

Thus $\mathsf{AtsujiSpace}^\uparrow$ *exposes the standard metric-continuity to uniform-modulus handoff as finite NameCert data, but it does not export open-cover compactness, a selected modulus, quotient map equality, countable choice, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Begin with a consumer that asks for uniform control of an admitted map. The carrier exposes the metric source row M and the continuous-map row F before the uniform-modulus handoff row U can be inspected.

Replay the displayed continuation row C. The request passes through M, F, U and then uses R as the real tolerance seal, while H, P, N keep the transported packet, provenance, and local name fixed.

The compact-metric comparison row K does not turn the argument into a compactness theorem. It only records that compact-metric examples may be compared through the sibling compact-metric surface while the Atsuji handoff remains the route M, F, U, R, H, C, P, N.

Because no selected modulus, quotient map, choice functional, open-cover cover witness, or ambient completeness row occurs in A, every accepted uniform-continuity handoff factors through the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{AtsujiSpace}^{\uparrow}$ read factors through $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it is consumed as a metric-continuity to uniform-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{AtsujiSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{AtsujiSpace}^{\uparrow}$ packet is generated from a Metric source row, ContinuousMap testing row, UniformModulus handoff row, CompactMetric comparison row, Real tolerance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AtsujiSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7946.1, Definition 7946.2, and Definition 7946.3 seed the L10 Atsuji route over MetricUp, ContinuousMapUp, UniformModulusUp, CompactMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AtsujiSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected moduli, quotient maps, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, continuous-map testing, uniform-modulus handoff nonescape, compact-metric comparison scope, real tolerance sealing, and TasteGate field faithfulness for BEDC.Derived.AtsujiSpaceUp.

Chapter 7947

A Concrete Naming Certificate for SequentialBolzanoChoiceFreeSubsequence$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the SequentialBolzanoChoiceFreeSubsequence$^{\uparrow}$ packet records the finite subsequence-extraction surface used by sequential Bolzano arguments before Real and Completion consumers read a regular Cauchy subsequence. It sits after the dyadic tolerance carrier of chapter 291, the finite stream-name schedule of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the diagonal selector route of chapter 695. The packet names a displayed subsequence route; it is not open-cover compactness, countable choice, a selected host limit, a quotient stream, or ambient Real$^{\uparrow}$ completeness.

Definition 7947.1 (Sequential Bolzano choice-free subsequence carrier)**.** *A* SequentialBolzanoChoiceFreeSubsequen *carrier is a finite* BHist *packet*

$$B = (S, Q, D, W, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row, exposed only through inspected finite* StreamName$^{\uparrow}$ *windows.*
2. *Q is the finite subsequence schedule row, listing the inspected source positions and their monotone order inside those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to compare source windows, selected windows, and Cauchy-tail requests.*
4. *W is the* StreamName$^{\uparrow}$ *window family opened by the selected schedule.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row assembled from the selected windows and tolerance ledger.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row available only after Q, D, W, R have been displayed.*
7. *A is the* ChoiceFreeDiagonalSelector$^{\uparrow}$ *selector row that binds the precision request to the displayed subsequence window rather than to a host choice function.*
8. *H records componentwise* hsame *transport for the source, schedule, tolerance, window, readback, seal, selector, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a sequential Bolzano request through $S, Q, D, W, R, E, A, H, P, N$.*
10. *P is the* Pkg *provenance over* SequentialBolzanoChoiceFreeSubsequence$^{\uparrow}$, ChoiceFreeDiagonalSelector$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these displayed rows. It has no coordinate for open-cover compactness, countable choice, a selected host limit, quotient stream equality, host sequence equality, or ambient Real completeness.

Theorem 7947.2 (Sequential Bolzano choice-free subsequence NameCert obligations)**.** *For every accepted* $\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}$ *carrier*

$$B = (S, Q, D, W, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission, finite source-window exposure, subsequence schedule visibility, dyadic tolerance accounting, StreamName window replay, RegSeqRat readback, Real seal handoff, ChoiceFreeDiagonalSelector routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every sequential Bolzano or Real-completion read must pass through the displayed rows before it consumes any subsequence output.*

Proof. Project the carrier. The analytic rows are S, Q, D, W, R, E, A, and the structural rows are H, C, P, N.

The source row S is exposed only by finite StreamName windows. The schedule row Q selects inspected positions inside those windows, while D records the dyadic tolerance budget and R records the regular rational readback.

The selector row A imports the choice-free diagonal selector boundary of chapter 695. It binds precision requests to displayed finite windows rather than to an external choice function.

Finally E is downstream of Q, D, W, R, A, and H, C, P, N only transport, replay, source, and name the same route. Since no carrier coordinate stores open-cover compactness, countable choice, selected host limits, quotient streams, host sequence equality, or ambient completeness, the displayed rows exhaust the NameCert surface. □

Theorem 7947.3 (Sequential Bolzano choice-free real route)**.** *Every Real and Completion consumer of an accepted* $\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}$ *packet factors through*

$$\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed carrier rows before any subsequence output is public. The route exposes the selector, finite windows, dyadic tolerance ledger, regular rational readback, Real seal, transport, replay, provenance, and local naming; it exports no open-cover compactness, countable-choice schedule, selected host limit, quotient stream, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 7947.2. They force the selected subsequence schedule and tolerance ledger to be displayed before the readback and seal rows can be consumed.

Use the sibling selector route of chapter 695. The selector row sends each precision request to a finite StreamName window, so the consumer reads A, W, D, R, E as one displayed route.

The RegSeqRat and Real rows are downstream of that route. The readback row R is assembled from the selected windows and dyadic tolerance ledger, and the seal row E is available only after R has been displayed.

Thus a Real or Completion consumer receives only the finite route through $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The excluded compactness, choice, host-limit, quotient-stream, and ambient-completeness objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}$ packet is generated from source sequence rows, finite subsequence schedules, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readbacks, RealUp seals, a ChoiceFreeDiagonalSelector row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7947.1, Definition 7947.2, and Definition 7947.3 seed the choice-free sequential Bolzano subsequence route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialBolzanoChoiceFreeSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected host limits, quotient streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite subsequence scheduling, dyadic tolerance accounting, StreamName window replay, RegSeqRat readback, Real seal routing, ChoiceFreeDiagonalSelector handoff, transport, replay, provenance, and local naming for BEDC.Derived.SequentialBolzanoChoiceFreeSubsequenceUp.

Chapter 7948

A Concrete Naming Certificate for DyadicContractionModulus$^{\uparrow}$

Following Bishop and Bridges on regular Cauchy estimates, the DyadicContractionModulus$^{\uparrow}$ packet records the finite dyadic contraction budget used by the diagonal regularity route of section 9238.1. It binds the endpoint-leg estimates, middle-modulus estimate, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, and terminal Real seal as one NameCert surface. The packet realizes a three-leg regularity estimate; it is not a quotient stream, selected limit, countable-choice schedule, open-cover compactness theorem, or ambient Real$^{\uparrow}$ completeness principle.

The three-leg estimate consumes the tolerance-scale normalization of chapter 1768 and the diagonal-selection route of chapter 4080 before its StreamName, RegSeqRat, and Real seal rows are read.

Definition 7948.1 (Dyadic contraction modulus carrier)**.** *A* DyadicContractionModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, L, M, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the precision-request row naming the diagonal regularity comparison.*
2. *L is the left endpoint-leg dyadic contraction row.*
3. *M is the middle modulus row comparing the two selected regular Cauchy stages.*
4. *U is the right endpoint-leg dyadic contraction row.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger that sums the three contraction rows.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window row opened by the comparison request.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained after the three dyadic legs are displayed.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row available only after Q, L, M, U, D, W, R have been replayed.*
9. *H records componentwise* hsame *transport for the request, three contraction rows, tolerance ledger, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a diagonal regularity request through $Q, L, M, U, D, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* DyadicContractionModulus$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{DyadicContractionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these finite rows to a common tolerance surface.

Theorem 7948.2 (Dyadic contraction modulus NameCert obligations)**.** *For every accepted* DyadicContractionModu *carrier*

$$K = (Q, L, M, U, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicContractionModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, precision-request exposure, left endpoint-leg contraction, middle-modulus contraction, right endpoint-leg contraction, DyadicRatCore tolerance summation, StreamName window replay, RegSeqRat readback, Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier. The three analytic contraction rows are L, M, U, the shared tolerance ledger is D, and the Real-completion readback rows are W, R, E.

The precision request Q determines which diagonal regularity comparison is being read. The left endpoint leg, middle modulus leg, and right endpoint leg are visible before the DyadicRatCore ledger D sums their tolerances.

StreamName and RegSeqRat consumption is downstream of the dyadic ledger. The finite window W is opened for the same request, R records the regular rational readback, and E seals the resulting Real row only after those displays.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient stream, selected limit, countable-choice schedule, open-cover compactness row, or ambient completeness coordinate, no local obligation can export those objects. □

Theorem 7948.3 (Dyadic contraction modulus real-completeness route)**.** *The diagonal regularity estimate of Definition 9238.4 is consumed through a* $\mathsf{DyadicContractionModulus}^{\uparrow}$ *packet only by the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route displays the left endpoint leg, middle modulus leg, right endpoint leg, dyadic tolerance ledger, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming before any Real-completion consumer may read the estimate.

Proof. Start with the carrier obligations of Definition 7948.2. They expose the three contraction rows L, M, U and the DyadicRatCore ledger D before any readback or seal row is available.

Use the source vision theorem Definition 9238.4. Its proof route has exactly the left endpoint leg, middle modulus comparison, and right endpoint leg named by L, M, U.

The finite window and readback rows are then forced by W and R. The Real seal E is downstream of those rows, so a Real-completion consumer reads the estimate only after the dyadic and regular-name data have been displayed.

Therefore the public route is the finite DyadicRatCore-to-StreamName-to-RegSeqRat-to-RealUp handoff. Quotient streams, selected limits, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier and cannot replace that handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicContractionModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicContractionModulus}^{\uparrow}$ packet is generated from a precision request, left endpoint-leg contraction, middle-modulus contraction, right endpoint-leg contraction, a DyadicRatCore tolerance ledger, StreamName

windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicContractionModulus}\uparrow}$, seed)

Scope closed: Definition 7948.1, Definition 7948.2, and Definition 7948.3 seed the three-leg dyadic contraction route for Real-completeness consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicContractionModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked precision requests, three-leg dyadic contraction rows, DyadicRatCore tolerance summation, StreamName window replay, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicContractionModulusUp.

Chapter 7949

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$

Following Bishop and Bridges constructive compactness, the $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ packet records the finite modulus surface by which a compact-metric sequence exposes a monotone subsequence schedule before any regular-Cauchy or real-name readback is consumed. It sits over the compact metric source of chapter 129, the finite-net bridge of chapter 3569, the stream interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the sibling regular subsequence modulus of chapter 6116. The packet names the compact-metric subsequence-selection modulus as finite NameCert data; it does not import countable choice, quotient stream equality, open-cover compactness, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7949.1 (Cauchy subsequence selection modulus carrier)**.** *A* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, A, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite metric inspections and compact-source identifiers bound the sequence consumer.*
2. *T is the finite* $\mathsf{TotallyBounded}^{\uparrow}$ *net ledger read from K, including visible positive radii, center histories, and coverage witnesses.*
3. *A is the source* $\mathsf{StreamName}^{\uparrow}$ *sequence row, recording the finite windows of the sequence inspected by the selection procedure.*
4. *M is the subsequence selection modulus row. For each displayed dyadic tolerance it records a monotone index schedule, a retained finite-net cell, and the finite cofinality checks used to compare later selected terms.*
5. *W is the selected-window row, listing the source windows retained by M together with their dyadic comparison budgets.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row opened only after K, T, A, M, W have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster-seal row reached from the regular readback.*
8. *H records componentwise* hsame *transport for the compact source, finite nets, source stream, selection modulus, selected windows, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-metric sequence requests through $K, T, A, M, W, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}}$ *row.*

Definition 7949.2 (Cauchy subsequence selection modulus classifier)**.** *The classifier of an accepted* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *packet compares two carriers only through the displayed rows $K, T, A, M, W, R, E, H, C, P, N$. It requires the same compact-metric source row, the same visible finite-net ledger, the same finite source windows, the same monotone selection modulus, the same selected-window and dyadic-budget rows, and the same downstream regular readback and real seal up to componentwise* hsame *transport and displayed* Cont *replay. It has no coordinate for a host-selected infinite subsequence, quotient equality of streams, open-cover compactness, countable choice, a chosen cluster point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7949.3 (Selection modulus carrier admission)**.** *For every accepted* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *carrier $S = (K, T, A, M, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}}$ *surface admits exactly the compact metric source, finite-net ledger, source stream windows, selection modulus, selected-window budget, regular readback, Real seal, transport, replay, provenance, and local naming rows displayed in Definition 7949.1.*

Proof. Project the accepted carrier. The mathematical rows are K, T, A, M, W, R, E, and the structural rows are H, C, P, N.

No further coordinate is present. Thus carrier admission is exhausted by the compact source, finite nets, source stream, selection modulus, selected windows, regular readback, real seal, and the four structural rows. □

Theorem 7949.4 (Selection modulus classifier stability)**.** *The classifier of an accepted* CauchySubsequenceSelection *carrier is stable under componentwise* hsame *transport of K, T, A, M, W, R, E. The transported packet has the same displayed* Cont *replay,* Pkg *provenance, and local* NameCert *surface, and it does not gain a second subsequence selector.*

Proof. Use the transport row H. It moves each displayed mathematical row componentwise and also transports the structural rows that replay, source, and name the packet.

The classifier of Definition 7949.2 reads only those transported coordinates. Since the carrier contains no external selector coordinate, transport cannot introduce countable choice, quotient stream equality, or a hidden selected subsequence. □

Theorem 7949.5 (Selection modulus finite-net extraction)**.** *Every subsequence-selection read from an accepted* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *carrier factors through the finite-net row T and the monotone schedule row M. For each displayed tolerance, M retains only a finite-net cell supplied by T, a finite list of inspected source windows from A, and the selected-window comparison budget recorded in W.*

Proof. Begin with one displayed tolerance request. The finite-net data available to the packet are exactly the centers, coverage witnesses, and radius ledger in T.

The schedule row M can therefore choose only among the finite cells exposed by T and the source windows exposed by A. The retained windows and comparison budgets are then written in W.

Induct over the displayed tolerance list. The empty list asks for no selection, while the next request appends one retained finite cell and one finite source-window comparison. The induction never appeals to a global choice function. □

Theorem 7949.6 (Selection modulus regular handoff)**.** *Any regular-Cauchy readback exported by an accepted* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *packet reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the route*

$$K, T, A, M, W, R.$$

The row R reads the selected windows and dyadic budgets already displayed by M, W and exports no quotient stream equality, host sequence equality, or unrecorded fast Cauchy modulus.

Proof. The regular readback coordinate is R. It is placed downstream of the compact source, finite-net ledger, source windows, selection modulus, and selected-window budget.

The previous extraction theorem identifies M, W as finite rows built from T and A. Hence R receives only the displayed selected-window route and cannot replace it with a quotient stream, host equality, or hidden modulus. □

Theorem 7949.7 (Selection modulus real nonescape)**.** *Every cluster-facing read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal E of an accepted* $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ *carrier must first expose K, T, A, M, W, R. The seal is the terminal observation of that finite route and exports no selected limit, open-cover compactness theorem, countable choice, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The terminal seal coordinate is E. A consumer reaches it only after the regular readback row R has consumed the selected-window route.

By the regular handoff theorem, R itself is downstream of K, T, A, M, W. Therefore the real seal cannot be read without displaying the compact source, finite nets, source windows, selection modulus, selected windows, and regular readback.

The excluded principles are not coordinates of the carrier or classifier. The seal consequently remains a terminal row of the finite NameCert route. □

Falsifiable prediction: Every accepted $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ route to a RegSeqRat or Real boundary must display a compact metric source, finite-net ledger, source StreamName windows, monotone selection modulus, selected-window budget, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, totallybounded, streamname, regseqrat, real, regularcauchysubsequencemodulus: none is bijective with $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ because this packet jointly carries the compact source, finite nets, source sequence windows, monotone subsequence-selection modulus, selected-window budgets, regular handoff, real seal, transport, replay, provenance, and local naming for the same compact-metric extraction route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}$ packet is generated from a CompactMetric row, TotallyBounded finite-net ledger, StreamName source windows, monotone selection modulus, selected-window budget, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelectionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7949.1, Definition 7949.2, Definition 7949.3, Definition 7949.4, Definition 7949.5, Definition 7949.6, and Definition 7949.7 seed the compact-metric subsequence-selection modulus route from finite nets to RegSeqRat and Real boundaries.

Formal verification: FormalStatus(NameCert$_{\text{CauchySubsequenceSelectionModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, host-selected subsequences, quotient stream equality, open-cover compactness, selected cluster points, ambient RealUp completeness, Lean checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite-net extraction, regular handoff, real nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySubsequenceSelectionModulusUp.

Chapter 7950

A Concrete Naming Certificate for $\mathsf{UniformBornology}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and bornologies, the $\mathsf{UniformBornology}^{\uparrow}$ packet records the finite interface in which a displayed uniform-space entourage row controls bounded-family reads without reducing them to metric-ball boundedness. It sits after the bornology carrier of chapter 3370, the uniform-space entourage discipline of chapter 339, the bounded-set row of chapter 1757, the Cauchy bounded-family route of chapter 1896, and the proper-metric completion boundary of chapter 3686. The packet gives L10 completion, properness, and Cauchy-bornology consumers a non-quotient bounded-uniform interface with no selected entourage scale, host bounded subset, metric-ball collapse, quotient completion, or ambient compactness principle.

Definition 7950.1 (Uniform bornology carrier)**.** *A* $\mathsf{UniformBornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q_{\mathrm{ub}} = (U, B, E, H_f, F, K, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *row naming the finite entourage requests, refinements, inverse requests, and composition windows that may be inspected by a bounded-family consumer.*
2. *B is the bounded-family source row, read through the* $\mathsf{Bornology}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *carriers, and containing only the displayed bounded families admitted by the packet.*
3. *E is the entourage-boundedness ledger. For each displayed family in B, it records the finite entourage windows of U that keep that family inside the same bounded source row after a uniform thickening request.*
4. *H_f is the hereditary ledger, requiring every displayed subfamily or smaller bounded-set window inherited from B to remain accepted after the same entourage-boundedness read in E.*
5. *F is the finite-union ledger, requiring finite unions of displayed bounded families to be accepted by the same source row and to replay through the same finite entourage windows.*
6. *K is the Cauchy and proper-metric handoff row. It exposes the route by which* $\mathsf{CauchyBornology}^{\uparrow}$ *consumers read bounded Cauchy data and by which* $\mathsf{ProperMetricSpace}^{\uparrow}$ *consumers read closed-ball boundedness, always after U, B, E, H_f, F have been replayed.*
7. *H records componentwise* hsame *transport for the uniform-space row, bounded-family row, entourage-boundedness ledger, hereditary ledger, finite-union ledger, handoff row, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a uniform-bornology request through* $U, B, E, H_f, F, K, H, P, N$.
9. *P is the* Pkg *provenance over* $\mathsf{UniformBornology}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Bornology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, $\mathsf{ProperMetricSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformBornology}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a selected entourage scale, quotient completion class, host bounded subset, metric-ball-only boundedness theorem, open-cover compactness proof, or ambient completeness principle.

Theorem 7950.2 (Uniform bornology NameCert obligations)**.** *For every accepted* $\mathsf{UniformBornology}^{\uparrow}$ *carrier*

$$Q_{\mathrm{ub}} = (U, B, E, H_f, F, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformBornology}^{\uparrow}}$ *surface consists exactly of carrier admission, uniform-space row exposure, bounded-family source exposure, entourage-boundedness exactness, hereditary stability, finite-union stability, Cauchy and proper-metric handoff ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing consumer that asks for bounded-uniform data must read these rows before it can consume a Cauchy-bornological or proper-metric boundary.*

Proof. Carrier admission is projection from Definition 7950.1. The accepted packet exposes exactly $U, B, E, H_f, F, K, H, C, P, N$, so the uniform, bornological, bounded-set, ledger, handoff, transport, replay, provenance, and naming rows are finite BEDC data.

Entourage-boundedness is read from U, B, E. The uniform row supplies the finite entourage windows, the bounded-family row supplies the displayed source families, and the ledger E records that each admitted thickening request stays within the same bounded-family source.

Hereditary and finite-union stability are separate ledger reads. The row H_f handles displayed subfamilies and smaller bounded-set windows, while F handles finite unions. Both ledgers replay through E, so neither can introduce a host bounded subset or a metric-ball-only boundedness theorem.

Finally inspect K, H, C, P, N. The handoff row is available only after the bounded-uniform ledger has been replayed, and the structural rows preserve the same finite packet. Since no coordinate stores a selected entourage scale, quotient completion, open-cover compactness proof, or ambient completeness theorem, the NameCert obligations are exhausted by the displayed rows. □

Theorem 7950.3 (Uniform bornology uniform-space handoff)**.** *Every* $\mathsf{CauchyBornology}^{\uparrow}$ *or* $\mathsf{ProperMetricSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformBornology}^{\uparrow}$ *packet reads bounded-family data through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow},\quad \mathsf{Bornology}^{\uparrow},\quad \mathsf{BoundedSet}^{\uparrow},\quad \mathsf{CauchyBornology}^{\uparrow},\quad \mathsf{ProperMetricSpace}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Con}$$

The handoff row K may expose Cauchy boundedness or proper closed-ball boundedness only after the uniform entourage row U, bounded-family row B, entourage-boundedness ledger E, hereditary ledger H_f, and finite-union ledger F have been replayed. Thus the exported object is a bounded-uniform NameCert route, not a metric-ball collapse, selected radius, quotient completion, host subset, or ambient compactness principle.

Proof. Start with the uniform-space row U and the bounded-family source row B. These rows fix the entourage requests and the bounded families before any downstream consumer can inspect the handoff.

Use the entourage-boundedness ledger E to replay uniform thickenings of displayed bounded families. Then use H_f and F to close the same source row under displayed subfamilies and finite unions.

Only after those ledger reads is the row K available. Its Cauchy side routes bounded Cauchy reads to $\mathsf{CauchyBornology}^{\uparrow}$, while its proper-metric side routes closed-ball boundedness reads to $\mathsf{ProperMetricSpace}^{\uparrow}$; both sides share the same U, B, E, H_f, F prefix.

The transport, replay, provenance, and local naming rows preserve that prefix. Because the carrier has no coordinate for the excluded host principles, a consumer receives exactly the bounded-uniform handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{UniformBornology}^{\uparrow}$ consumer displays a $\mathsf{UniformSpace}^{\uparrow}$ row, a bounded-family source row through $\mathsf{Bornology}^{\uparrow}$ and $\mathsf{BoundedSet}^{\uparrow}$, an entourage-boundedness ledger, hereditary and finite-union ledgers, a Cauchy/proper-metric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any completion-facing boundedness read is available.

Independence witness: bornology, uniformspace, boundedset, cauchy-bornology, proper-metric-space: $\mathsf{UniformBornology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries uniform entourage control, bounded-family source data, entourage-boundedness, hereditary and finite-union ledgers, and Cauchy/proper-metric handoff rows as one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformBornology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformBornology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7950.1, Definition 7950.2, and Definition 7950.3 seed the bounded-uniform interface over $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Bornology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, $\mathsf{ProperMetricSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformBornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected entourage scales, quotient completion, metric-ball-only boundedness, host bounded subsets, open-cover compactness, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, entourage-boundedness exactness, hereditary stability, finite-union stability, Cauchy/proper-metric handoff ordering, transport replay, provenance, and local NameCert exactness for BEDC.Derived.UniformBornologyUp.

Constructive story: A $\mathsf{UniformBornology}^{\uparrow}$ packet is generated from UniformSpace entourage rows, Bornology and BoundedSet bounded-family rows, an entourage-boundedness ledger, hereditary and finite-union ledgers, a Cauchy and proper-metric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 7951

A Concrete Naming Certificate for $\mathsf{BrowderFixedPoint}^{\uparrow}$

Following Browder's fixed point theorem in nonlinear functional analysis, the $\mathsf{BrowderFixedPoint}^{\uparrow}$ packet records the finite BEDC route by which a compact convex self-map obligation is exposed through approximating finite-dimensional faces and monotone residual windows. It is anchored by the Schauder compact-convex surface of chapter 1923, the set-valued comparison surface of chapter 3511, the Brouwer finite-cell route of chapter 2030, the compact metric support of chapter 129, the convex carrier of chapter 208, the self-map rows of chapter 130, and the ambient rows of chapter 94. The packet names a constructive fixed-point frontier; it does not import a host fixed-point selector, an unrestricted compactness theorem, or a choice principle for limiting points.

Definition 7951.1 (BrowderFixedPoint carrier)**.** *A* $\mathsf{BrowderFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, V, Q, F, A, M, R, W, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row for the compact domain and its finite covering windows.*
2. *V is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *ambient row exposing only the carried vector, affine, and metric compatibility data needed by the fixed-point route.*
3. *Q is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the admitted compact convex carrier.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row, including graph access, source membership, target membership, and displayed modulus data.*
5. *A is the finite affine-approximation ledger, recording finite-dimensional faces, retractions into those faces, and the local comparison maps.*
6. *M is the monotone residual-window row that compares the self-map route with the affine approximation route at each displayed tolerance.*
7. *R is the Brouwer-Schauder reduction row, carrying the* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *cell witness route and the* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *compact-convex handoff only through A and M.*
8. *W is the fixed-window boundary row recording the accepted residual boundary visible to downstream consumers.*
9. *H stores componentwise* hsame *transport for K, V, Q, F, A, M, R, W and the local certificate row.*

10. *C stores displayed* Cont *replay from compact support, convex membership, continuous-map reads, affine approximations, residual windows, reduction rows, and fixed-window export.*

11. *P is the* Pkg *provenance over* $\mathsf{BrowderFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BrowderFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected fixed point, an arbitrary weak compactness principle, a host topological-vector-space theorem, quotient equality of approximating faces, or any choice of a convergent subnet.

Theorem 7951.2 (BrowderFixedPoint NameCert obligations)**.** *For an accepted* $\mathsf{BrowderFixedPoint}^{\uparrow}$ *carrier $B = (K, V, Q, F, A, M, R, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BrowderFixedPoint}^{\uparrow}}$ *surface consists of exactly five obligation rows:*

1. *compact-convex domain admission through K, V, Q;*
2. *continuous self-map admission through F;*
3. *finite affine-approximation and monotone residual admission through A, M;*
4. *Brouwer-Schauder reduction and fixed-window exposure through R, W;*
5. *structural transport, replay, provenance, and local naming through H, C, P, N.*

No obligation may call on a host fixed-point selector, unrestricted compactness, ambient weak convergence, quotient equality of approximation ledgers, or an uncarried nonlinear-analysis theorem.

Proof. Project the accepted carrier of Definition 7951.1. The compact-convex domain data are exactly K, V, Q, so compact support, ambient compatibility, and convex membership enter as displayed packet rows.

The self-map route is then read only through F. The approximation and residual obligations consume A and M, which expose finite affine faces, retractions, comparison maps, and monotone residual windows without exporting a selected limiting point.

The reduction row R may cite the Brouwer and Schauder sibling surfaces, but only as finite handoffs attached to the displayed approximation and residual rows. The fixed-window row W records the boundary that downstream consumers can inspect.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no other coordinate, the listed five rows exhaust the local NameCert surface and exclude the host principles named in the statement. □

Theorem 7951.3 (BrowderFixedPoint compact-convex handoff)**.** *Every downstream read of an accepted* $\mathsf{BrowderFixedPoint}^{\uparrow}$ *packet factors through*

$$K, V, Q, F, A, M, R, W, H, C, P, N.$$

In particular, a consumer can reach the fixed-window boundary only after the compact metric support, functional-analysis ambient compatibility, convex carrier, continuous self-map, affine approximation ledger, monotone residual row, and Brouwer-Schauder reduction row have all been replayed.

Proof. Begin with a downstream consumer. The continuation row C is the only consumer-facing replay surface, and its compact-convex reads are attached to K, V, Q.

The continuous self-map row F is read next. It supplies graph access and membership replay for the displayed compact convex carrier, but it does not select a fixed point.

The affine-approximation ledger A and monotone residual row M then feed the reduction row R. This is the only place where the Brouwer, Schauder, and Kakutani sibling surfaces can be inspected, and their use remains tied to finite approximation and residual windows.

The fixed-window boundary W is therefore an exported boundary of the displayed route, not a host choice. The rows H, C, P, N preserve the same ordering under transport, replay, provenance, and local naming, so every downstream read factors through the surface named in the theorem. □

Falsifiable prediction: Every accepted $\mathsf{BrowderFixedPoint}^{\uparrow}$ consumer must expose compact metric support, functional-analysis ambient compatibility, convex carrier, continuous self-map, affine approximation, monotone residual, Brouwer-Schauder reduction, fixed-window boundary, transport, continuation, provenance, and local NameCert rows before it can read a fixed-window obligation.

Independence witness: schauder-fixed-point, kakutani-fixed-point, brouwer-fixed-point, compactmetric, convexset, continuousmap, functionalanalysis: $\mathsf{BrowderFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries affine approximation faces, monotone residual windows, a Browder-facing reduction row, fixed-window export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BrowderFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{BrowderFixedPoint}^{\uparrow}$ packet is generated from CompactMetric support rows, FunctionalAnalysis ambient compatibility rows, ConvexSet carrier rows, ContinuousMap self-map rows, affine-approximation ledgers, monotone residual windows, Brouwer-Schauder reduction rows, fixed-window boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrowderFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7951.1, Definition 7951.2, and Definition 7951.3 seed the finite Browder fixed-point naming route over the displayed compact metric, functional-analysis, convex, continuous-map, approximation, residual, reduction, fixed-window, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrowderFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host fixed-point selector, unrestricted compactness, ambient weak convergence, quotient equality of approximation ledgers, a choice principle for limiting points, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex domain admission, continuous-map replay, affine-approximation coverage, monotone residual coverage, Brouwer-Schauder reduction coverage, fixed-window exposure, transport stability, replay stability, provenance, and local naming for BEDC.Derived.BrowderFixedPointUp.

Chapter 7952

A Concrete Naming Certificate for $\mathsf{LocatedCutSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCutSequence}^{\uparrow}$ packet records located lower and upper dyadic cut observations as finite rows, not as a Dedekind construction of $\mathsf{Real}^{\uparrow}$. It sits beside the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback route of chapter 301, and the terminal seal of chapter 38. The packet gives Real and Completion consumers a sequence-indexed located-cut observation surface while keeping quotient cuts, selected limits, trichotomy, and ambient completeness outside the carrier.

Definition 7952.1 (Located cut sequence carrier)**.** *A* $\mathsf{LocatedCutSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, D, W, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the side-tag row. Each finite observation is tagged as a lower cut read, an upper cut read, or a paired located-window request, and the tag is part of the packet data.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound row. It records the dyadic endpoint, tolerance cell, and finite refinement scale inspected by the tagged observation.*
3. *W is the located witness-window row. It records the finite bracket, separation, or refinement witness attached to the tagged endpoint request.*
4. *O is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It lists the finite sequence positions and stream windows through which the cut observation is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It compares the finite window observations with a regular rational route before any real-name seal is consumed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is reached only after the side tag, dyadic bound, located witness window, stream observation, and regular readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the tag, dyadic, witness-window, stream, regular-readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite located-cut request through S, D, W, O, R before the seal row E is read.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCutSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutSequence}^{\uparrow}}$ *row naming exactly this finite sequence of located cut observations.*

The classifier accepts only packets that display those rows. It has no coordinate for a primitive Dedekind cut, quotient cut equality, trichotomy branch, selected limit, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 7952.2 (Located cut sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCutSequence}^{\uparrow}$ *carrier* $L = (S, D, W, O, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCutSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the side tag, dyadic bound, located witness window, finite* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Any Real or Completion consumer reading the packet factors through these rows and cannot cite the packet as a Dedekind-cut construction of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 7952.1. The analytic rows are exactly the side tag S, dyadic bound row D, located witness-window row W, finite stream observation row O, regular rational readback row R, and terminal seal row E.

The route order is load-bearing. A consumer first reads the side tag, then the dyadic endpoint and located witness window, then the finite stream positions, and only then the regular rational readback that can hand off to the real seal.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for quotient cuts, trichotomy, selected limits, or host completeness, the local naming certificate cannot export a Dedekind construction of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCutSequence}^{\uparrow}$ packet is generated from side-tagged lower and upper dyadic cut observations, located witness windows, finite StreamName rows, RegSeqRat readback, a terminal Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCutSequence}^{\uparrow}}$, seed)

Scope closed: Definition 7952.1 and Definition 7952.2 seed the finite sequence-indexed located-cut observation surface for Real and Completion boundary comparisons.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCutSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of reals, quotient cuts, trichotomy, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side-tag exactness, dyadic bound readback, located witness-window locality, finite StreamName observation coverage, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCutSequenceUp.

Chapter 7953

A Concrete Naming Certificate for $\mathsf{LongLine}^{\uparrow}$

Following Munkres and Kelley on the long line, the $\mathsf{LongLine}^{\uparrow}$ packet records the classical lexicographic order-topology boundary as finite BEDC NameCert data. It sits beside the topology carrier of chapter 88, the order interface of chapter 1378, and the total-order comparison surface of chapter 54. The packet names inspected well-order prefixes, interval cells, lexicographic comparisons, order-topology neighbourhoods, transport, replay, provenance, and local naming without importing a host ordinal, uncountable choice, quotient point equality, selected representatives, or a compactness theorem.

Definition 7953.1 (Long line carrier)**.** *A* $\mathsf{LongLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, I, O, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the well-order-prefix schedule. Each inspected prefix is recorded as a finite named stage together with its predecessor and successor requests; the schedule is a displayed row of prefix evidence, not a host ordinal.*
2. *I is the interval-cell row. It records the unit-cell coordinate inside the displayed prefix stage, the left and right boundary requests, and the local interval membership tests needed by neighbourhood consumers.*
3. *O is the lexicographic comparison row. It compares two displayed points first by the prefix schedule S and then by the interval-cell coordinate I, exposing the comparison only through the order rows of chapter 1378 and chapter 54.*
4. *T is the order-topology neighbourhood row. It records finite open rays and finite open intervals generated from O and exports them to the topology carrier of chapter 88.*
5. *H records componentwise* hsame *transport for the prefix schedule, interval cells, lexicographic comparisons, order-topology neighbourhoods, provenance, replay, and local naming.*
6. *C records displayed* Cont *replay from a neighbourhood or comparison request through S, I, O, T and then through the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{LongLine}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

8. N is the local $\mathsf{NameCert}_{\mathsf{LongLine}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through S, I, O, T, H, C, P, N. It has no coordinate for a host ordinal, an uncountable choice principle, quotient point equality, selected representatives, a completed open-set lattice, or a compactness theorem.

Lean4 ✓ `BEDC.Derived.LongLineUp.LongLineTasteGate single carrier alignment`

Theorem 7953.2 (LongLine NameCert obligations)**.** *For every accepted* $\mathsf{LongLine}^{\uparrow}$ *carrier $L = (S, I, O, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LongLine}^{\uparrow}}$ *surface consists exactly of the well-order-prefix schedule S, interval-cell row I, lexicographic comparison row O, order-topology neighbourhood row T, componentwise* hsame *transport H, displayed* Cont *replay C, package provenance P, and local naming N. Every topology- or order-facing consumer must expose these rows before reading a long-line point comparison, interval membership test, open ray, or open interval.*

Proof. Project the carrier of Definition 7953.1. The only prefix evidence is S, and the only within-prefix coordinate evidence is I.

The order-facing read is then forced through O. It first reads the displayed prefix schedule and then the interval-cell coordinate, so comparison data cannot be replaced by a host ordinal comparison or by quotient point equality.

The topology-facing read is forced through T. Its open rays and intervals are generated from the displayed lexicographic comparison row and are exported only through the topology surface cited above.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores uncountable choice, selected representatives, a completed open-set lattice, or compactness, the listed rows exhaust the local NameCert obligations. □

Theorem 7953.3 (LongLine lexicographic order route)**.** *Every* $\mathsf{LongLine}^{\uparrow}$ *consumer that reads a point comparison or an order-topology neighbourhood factors through the finite route*

$$S, I \longrightarrow O \longrightarrow T \longrightarrow H, C, P, N.$$

The route exposes the order comparison inherited from chapter 1378 and chapter 54 before it exposes the topology read of chapter 88. It exports no host ordinal, no uncountable choice, no quotient point equality, no selected representatives, and no compactness theorem.

Proof. Start with the displayed point data. A point read has only the prefix schedule S and interval-cell row I, so there is no separate representative or host ordinal from which a consumer can read comparison data.

Next read the lexicographic row O. It is the sole comparison row and therefore the only route to order-facing conclusions.

Then read the order-topology row T. Open rays and open intervals are generated from O, so a topology consumer reaches the open-neighbourhood surface only after the displayed lexicographic order route is visible.

The structural rows H, C, P, N preserve and name that same route. They do not add uncountable choice, quotient point equality, selected representatives, a completed open-set lattice, or compactness evidence. □

Falsifiable prediction: An accepted $\mathsf{LongLine}^{\uparrow}$ consumer is refuted if it reads a point comparison, open ray, open interval, topology membership, transport, replay, provenance, or local naming without first displaying the corresponding S, I, O, T, H, C, P, N rows.

Independence witness: topology, order, totalorder: no listed sibling is bijective to $\mathsf{LongLine}^{\uparrow}$ because the packet jointly carries a displayed well-order-prefix schedule, interval-cell row, lexicographic comparison row, order-topology neighbourhood row, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LongLine}^{\uparrow}}$.

Constructive story: A $\mathsf{LongLine}^{\uparrow}$ packet is generated from a well-order-prefix schedule, interval-cell row, lexicographic comparison row, order-topology neighbourhood row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LongLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7953.1, Definition 7953.2, and Definition 7953.3 seed the finite long-line order-topology route over displayed prefix, interval, comparison, neighbourhood, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LongLine}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LongLineUp.LongLineTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host ordinals, uncountable choice, quotient point equality, selected representatives, completed open-set lattices, compactness theorems, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-schedule exactness, interval-cell exactness, lexicographic comparison routing, order-topology neighbourhood routing, transport, replay, provenance, and local naming for BEDC.Derived.LongLineUp.

Chapter 7954

A Concrete Naming Certificate for $\mathsf{CompactUniformFanModulus}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity and Brouwer fan-bar arguments, the $\mathsf{CompactUniformFanModulus}^{\uparrow}$ packet records the finite fan-to-uniform-modulus route used by L10 Real and Completion consumers. It sits over the compact metric source of chapter 129, the pointwise continuous-map rows of chapter 130, the finite fan-bar modulus surface of chapter 4297, the uniform-modulus handoff of chapter 369, and the terminal real-name seal of chapter 38. The packet turns displayed compact finite-net and fan-bar data into a BEDC-native modulus certificate; it does not assert arbitrary open-cover compactness, choose an infinite branch, select an external Lebesgue number, quotient real names, invoke countable choice, import a host modulus function, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7954.1 (Compact uniform fan modulus carrier)**.** *A* $\mathsf{CompactUniformFanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, B, L, T, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row read through* $\mathsf{CompactMetric}^{\uparrow}$*; it displays the finite center schedule, mesh radius, and cover-membership data consumed by the fan route.*
2. *F is the pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row, attaching a source-radius witness to each displayed center and requested target tolerance.*
3. *B is the* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *bar row; equivalently it may be read through the fan-theorem surface of chapter 400 when the fan proof has already exposed a finite bar.*
4. *L is the rational lower-bound fold, taking the finitely many radii displayed by K, F, B to one positive dyadic tolerance.*
5. *T is the dyadic tolerance ledger that records the requested target precision, the finite fold value, and the comparison budget used by the output route.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced by L and T.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the uniform-modulus request.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational approximants observed inside the opened windows.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after K, F, B, L, T, U, W, R have been replayed.*
10. *H records componentwise* hsame *transport for compact-net, continuous-map, fan-bar, lower-bound, tolerance, handoff, window, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a compact-uniform-modulus request through $K, F, B, L, T, U, W, R, E, H, P,$*
12. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFanModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteFanBarModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFanModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these finite rows to a common compact-fan tolerance surface.

Theorem 7954.2 (Compact uniform fan modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformFanModulus}^{\uparrow}$ *carrier*

$$M = (K, F, B, L, T, U, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformFanModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net exposure, pointwise continuous-map modulus exposure, finite fan-bar exposure, rational lower-bound fold, dyadic tolerance ledger,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *window replay,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7954.1. The only analytic source rows are $K, F, B, L, T, U, W, R, E$, and the only structural rows are H, C, P, N.

The compact and continuity data are read before the fan route. The finite-net row K supplies the displayed centers and mesh data, while F supplies the pointwise modulus rows attached to those centers.

The fan contribution is finite because B is a displayed bar-modulus row. The lower-bound fold L and tolerance ledger T consume the finitely many rows from K, F, B before the uniform-modulus output U is visible.

Finally W, R, E route the output into StreamName, RegSeqRat, and Real-facing reads, and H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for arbitrary open-cover compactness, selected infinite branches, selected external Lebesgue numbers, quotient map equality, countable choice, host modulus functions, or ambient Real completeness, no local obligation can export those objects. □

Theorem 7954.3 (Compact uniform fan modulus Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing compact-uniform-modulus read exported by an accepted* $\mathsf{CompactUniformFanModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{FiniteFanBarModulus}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegS}$$

The route displays compact finite-net data, pointwise continuous-map moduli, finite fan-bar data, the rational lower-bound fold, the dyadic tolerance ledger, the uniform-modulus handoff, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming before a Real consumer may read the output.

Proof. Start with the obligations of Definition 7954.2. They expose $K, F, B, L, T, U, W, R, E, H, C, P, N$ as the whole local NameCert surface.

The compact-to-fan prefix is K, F, B. The compact metric finite-net data in K and the pointwise continuous-map rows in F are consumed by the finite bar-modulus row B, so the route never needs an infinite branch or arbitrary open-cover selection.

The modulus output is then forced through L, T, U. The fold L and ledger T produce a displayed dyadic tolerance before U is available as a $\mathsf{UniformModulus}^{\uparrow}$ handoff.

The Real-facing suffix is W, R, E. StreamName windows are opened first, RegSeqRat readback follows, and the terminal $\mathsf{Real}^{\uparrow}$ seal is visible only after those finite rows have been replayed. The structural rows preserve the same route, so the exported Real read is exactly the displayed finite corridor. □

Theorem 7954.4 (Compact uniform fan modulus vision route)**.** *The compact-uniform-continuity vision consumes an accepted* $\mathsf{CompactUniformFanModulus}^{\uparrow}$ *packet only through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteFanBarModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. The route excludes arbitrary open-cover compactness, branch choice, external Lebesgue-number selection, quotient map equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7954.1 and the Real route of Definition 7954.3. The only vision-facing analytic rows are compact finite-net data, continuous-map modulus data, finite fan-bar data, uniform-modulus handoff data, stream windows, regular-rational readback, and the terminal Real seal.

Transport and replay preserve those same rows through H, C, P, N. Since no carrier coordinate stores open-cover compactness, an infinite branch, an external Lebesgue-number selector, quotient map equality, countable choice, or ambient real completeness, the vision can consume only the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFanModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFanModulus}^{\uparrow}$ packet is generated from compact finite-net rows, pointwise continuous-map modulus rows, finite fan-bar rows, a rational lower-bound fold, a dyadic tolerance ledger, a UniformModulus handoff, StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFanModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7954.1, Definition 7954.2, and Definition 7954.3 seed the compact finite-net, continuous-map, finite fan-bar, uniform-modulus, StreamName, RegSeqRat, and Real route for L10 consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFanModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected infinite branches, selected external Lebesgue numbers, quotient map equality, countable choice, host modulus functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net rows, pointwise continuous-map modulus rows, finite fan-bar rows, rational lower-bound fold, dyadic tolerance ledger, UniformModulus handoff, StreamName windows, RegSeqRat readback, Real seal, transport replay, provenance, and local NameCert exactness for BEDC.Derived.CompactUniformFanModulusUp

Chapter 7955

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ packet records the finite surface by which a displayed regular Cauchy name refines its modulus along a cofinal tail schedule before any Real and Completion consumer reads the result. It sits over the regular-name source of chapter 371, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite cofinal-refinement carrier; it does not construct quotient streams, selected limits, countable-choice schedules, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7955.1 (Regular Cauchy modulus cofinal-refinement carrier)**.** *A* RegularCauchyModulusCofinalRefineme *carrier is a finite* BHist *packet*

$$R = (S, M, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegularCauchyName}^{\uparrow}$ *row whose modulus is being refined.*
2. *M is the displayed cofinal modulus-refinement row, listing the retained tail thresholds and the monotone comparison between the source modulus and the refined modulus.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the retained thresholds in M.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic budgets served by the refined tail windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, M, W, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the source name, refinement row, windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from source regular Cauchy reads through the cofinal refinement, finite windows, regular rational readback, dyadic budgets, and terminal real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, M, W, Q, D, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected representative, countable choice over tails, host real equality, an unbounded search for a modulus, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyModulusCofinalRefinementUp`

Theorem 7955.2 (Regular Cauchy modulus cofinal-refinement NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ *carrier $R = (S, M, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}}$ *surface consists exactly of source regular-name exposure, displayed cofinal modulus refinement,* $\mathsf{StreamName}^{\uparrow}$ *finite-window scheduling,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger refusal for quotient streams, selected limits, countable choice, host real equality, unbounded modulus search, and ambient completeness.*

Proof. Project the carrier of Definition 7955.1. The accepted packet displays the source row S, the refinement row M, the window row W, the readback row Q, the tolerance row D, and the terminal seal E.

The structural rows H, C, P, N transport, replay, source, and name exactly those mathematical rows. Since no further coordinate is present, every local NameCert obligation must be read through $S, M, W, Q, D, E, H, C, P, N$.

The refusal clauses are carrier-coordinate exclusions. Quotient streams, selected limits, countable choice over tails, host real equality, unbounded modulus search, and ambient completeness are not rows of R, so they cannot be exported by the local certificate. □

Theorem 7955.3 (Regular Cauchy modulus cofinal-refinement stability)**.** *The classifier of an accepted* $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ *carrier is stable under componentwise* hsame *transport of S, M, W, Q, D, E. The transported packet preserves the same cofinal tail thresholds, the same finite windows, the same regular rational readback, the same dyadic tolerance ledger, the same terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the same* Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Use the transport row H. It transports the source name, cofinal-refinement row, window row, readback row, tolerance ledger, and Real seal componentwise.

The classifier of Definition 7955.1 reads only those coordinates together with C, P, N. Therefore any transported packet compares the same retained thresholds, windows, readbacks, dyadic budgets, seal, replay, provenance, and local naming rows.

No hidden selector is introduced by transport. The packet still lacks quotient streams, countable choice, host equality, unbounded modulus search, and ambient completeness coordinates. □

Theorem 7955.4 (Regular Cauchy modulus cofinal-refinement Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ *carrier factors through the finite route*

$$S, M, W, Q, D, E, H, C, P, N.$$

The route reaches the terminal seal only through the displayed regular Cauchy source, cofinal refinement, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance ledger.

Proof. The terminal seal coordinate is E. By the carrier definition, E is downstream of the source row S, the refinement row M, the finite windows W, the regular rational readback Q, and the dyadic tolerance ledger D.

The NameCert obligations theorem shows that these rows are the complete public surface. Transport, replay, provenance, and local naming are then exactly H, C, P, N.

Thus a Real-facing consumer cannot bypass the finite route by using a quotient stream, selected representative, countable-choice tail schedule, host equality of real names, unbounded modulus search, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ route to RegSeqRat or Real data must display a source RegularCauchyName row, a cofinal modulus-refinement row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: regularcauchyname, streamname, regseqrat, dyadicratcore, real: none is bijective with $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ because this packet jointly carries the source regular Cauchy name, cofinal modulus refinement, finite windows, rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming for the same tail-refinement route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}$ packet is generated from a RegularCauchyName source row, a cofinal modulus-refinement row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7955.1, Definition 7955.2, Definition 7955.3, and Definition 7955.4 seed the finite cofinal-refinement route from regular Cauchy modulus data to RegSeqRat and Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCofinalRefinement}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice over tails, host real equality, unbounded modulus search, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal refinement exposure, window stability, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyModulusCofinalRefinementUp.

Chapter 7956

A Concrete Naming Certificate for $\mathsf{LocatedRealLowerCut}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRealLowerCut}^{\uparrow}$ packet records a finite lower-cut locatedness surface for Real and Completion consumers. It sits beside the located-real comparison route of **??**, the finite stream observations of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names lower-cut evidence available to downstream comparison or completion routes without asserting Dedekind-cut reals, quotient equality, selected suprema, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7956.1 (Located-real lower-cut carrier)**.** *A* $\mathsf{LocatedRealLowerCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the lower-cut query row. It lists the finitely many rational thresholds whose membership in the lower approximation is inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to those thresholds, including the finite precision and separation budget used by the locator.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row through which the threshold data are read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row comparing the observed windows with a regular rational route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the threshold, tolerance, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a lower-cut query through Q, D, W, R before the seal row E is consumed.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealLowerCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLowerCut}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying these rows. It has no coordinate for a completed Dedekind cut, quotient equality of cuts, a least upper bound selector, trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Theorem 7956.2 (Located-real lower-cut NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealLowerCut}^{\uparrow}$ *carrier* $L = (Q, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLowerCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower-cut query row, dyadic tolerance row, finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Any Real or Completion consumer reading the packet factors through these rows and cannot cite the packet as a Dedekind-cut construction of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 7956.1. The analytic rows are exactly Q, D, W, R, E: a lower-cut query, a dyadic tolerance ledger, finite stream observations, regular rational readback, and the terminal real seal.

The route order is part of the packet. A consumer reads the rational threshold row Q, then the dyadic budget D, then the finite observation window W, and only then the regular rational readback R that can reach the real seal E.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for completed cuts, quotient equality, selected suprema, trichotomy, countable choice, or host completeness, the local naming certificate cannot export those objects. □

Theorem 7956.3 (Located-real lower-cut RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing read of an accepted* $\mathsf{LocatedRealLowerCut}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes only the lower-cut thresholds, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal seal, componentwise transport, replay, provenance, and local naming rows already displayed in the carrier.

Proof. Start with the obligations of Definition 7956.2. They expose $Q, D, W, R, E, H, C, P, N$ as the whole local NameCert surface.

The lower-cut information reaches a regular rational consumer through D, W, R. The dyadic threshold and tolerance row supplies the finite budget, the stream row supplies the inspected windows, and the readback row is the first point at which a $\mathsf{RegSeqRat}^{\uparrow}$ consumer can receive data.

The terminal Real or Completion consumer then reads only through E. The structural rows preserve the same packet under transport and replay, so the handoff cannot add Dedekind-cut equality, selected suprema, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealLowerCut}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealLowerCut}^{\uparrow}$ packet is generated from finite lower-cut threshold queries, a DyadicRatCore tolerance ledger, StreamName observation windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealLowerCut}^{\uparrow}}$, seed)

Scope closed: Definition 7956.1, Definition 7956.2, and Definition 7956.3 seed the finite lower-cut locatedness route through dyadic thresholds, stream windows, regular rational readback, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealLowerCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut reals, quotient equality, selected suprema, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-cut query exactness, dyadic tolerance coverage, StreamName window exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealLowerCutUp.

Chapter 7957

A Concrete Naming Certificate for $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$

Following Bishop and Bridges and Kelley on compactness via finite subcovers, the $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$ packet records only selection from already displayed finite open-cover rows. It sits after the compact metric source of chapter 129, the finite cover cells of chapter 2377, the stream schedules of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name surface of chapter 38. The packet names a finite selected subrow already visible in the cover ledger; it does not choose an arbitrary open cover, import countable choice, identify quotient real names, appeal to host compactness, select subcovers outside displayed finite rows, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7957.1 (Finite open cover selection carrier)**.** *A* $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, O, S, R, D, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the compact metric carrier and the finite cover request already admitted by that source.*
2. *O is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row, or equivalently the displayed finite cover-cell ledger, whose cells are the only candidates from which selection may read.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row attached to the cover cells and to the finite observation windows used by the consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations visible in those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and tolerance row attached to the cover cells, stream windows, and readback comparisons.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after K, O, S, R, D have been displayed.*
7. *U is the selected finite subrow of O, recorded as a displayed sublist or subbundle of cover cells rather than as an external selector.*
8. *H records componentwise* hsame *transport for compact-source, finite-cover, stream, readback, dyadic, Real-seal, selected-subrow, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite cover-selection request through $K, O, S, R, D, E, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed finite cover row and its selected finite subrow, transported together with the stream, readback, tolerance, Real-seal, replay, provenance, and naming rows.

Theorem 7957.2 (Finite open cover selection NameCert obligations)**.** *For every accepted* FiniteOpenCoverSelection *carrier $F = (K, O, S, R, D, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, compact metric source exposure, displayed finite cover-cell exposure,* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and tolerance exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure, selected finite subrow exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7957.1. The only cover source is O, and the only selected object is the finite subrow U already carried inside that displayed cover ledger.

Read the analytic rows in order. The compact metric source K admits the finite cover request, O displays the finite cover cells, S schedules the finite observations, R reads them back as regular rational data, D attaches dyadic radius and tolerance comparisons, and E supplies the terminal real-name seal.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for arbitrary open-cover compactness, countable choice, quotient real equality, host compactness, selected subcovers outside O, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects are not local NameCert obligations. □

Theorem 7957.3 (Finite open cover selection compact-uniform route)**.** *Every compact-uniform or Real-completion consumer of an accepted* $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$ *packet reads the selected subrow only through the finite corridor*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteRealCover}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The corridor exposes the selected finite subrow U as part of the displayed cover row O before any downstream uniform-modulus, finite-cover, or real-name consumer may use it.

Proof. Start with the NameCert obligations of Definition 7957.2. They force the compact source, finite cover cells, schedule, readback, dyadic tolerance, Real seal, selected subrow, and structural rows to be visible before the packet is consumed.

The selected subrow is read from O through U. Thus a downstream compact-uniform route receives a finite cover-cell subrow already displayed in the packet, not an open-cover selector and not a host compactness principle.

Finally replay the stream, regular-rational, dyadic, and Real-seal rows. They provide the finite observation and tolerance corridor through which the selected cover cells can feed a uniform-modulus or Real-completion consumer. The route has no coordinate for countable choice, quotient real equality, subcovers outside the displayed finite rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteOpenCoverSelection}^{\uparrow}$ packet is generated from CompactMetric source data, displayed finite cover cells, StreamName schedules, RegSeqRat readback, DyadicRatCore radius and tolerance rows, a RealUp seal, a selected finite subrow, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteOpenCoverSelection}^{\uparrow}}$, seed)

Scope closed: Definition 7957.1, Definition 7957.2, and Definition 7957.3 seed the finite cover-selection packet for compact-uniform and Real-completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteOpenCoverSelection}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, countable choice, quotient real equality, host compactness, selected subcovers outside displayed finite rows, ambient Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover-cell exposure, selected subrow exactness, StreamName schedule exposure, RegSeqRat readback, DyadicRatCore tolerance routing, Real seal handoff, transport replay, provenance, and local NameCert exactness for BEDC.Derived.FiniteOpenCoverSelectionUp.

Chapter 7958

A Concrete Naming Certificate for $\mathsf{ProfiniteMetricCompletion}^{\uparrow}$

Following Serre and Bourbaki on profinite completion and the Kelley/Bishop-style treatment of metric completion, the $\mathsf{ProfiniteMetricCompletion}^{\uparrow}$ packet records the finite inverse-limit metric completion neighbour obtained by reading profinite finite-stage data through a metric-completion boundary. It sits beside the finite quotient-stage certificate of chapter 5934, the projection-window metric route of chapter 2168, the finite observation discipline of chapter 126, the completion handoff of chapter 1506, and the compact metric readback of chapter 129. The packet names finite stages, finite projection checks, windowed metric readback, completion handoff, compact boundary evidence, transport, replay, provenance, and local naming; it does not construct quotient completions, select infinite representatives, import Haar measure, or assert ambient compactness beyond the displayed row.

Definition 7958.1 (Profinite metric completion carrier)**.** *A* $\mathsf{ProfiniteMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, I, W, M, K, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ProfiniteCompletion}^{\uparrow}$ *finite quotient-stage row, carrying only the displayed finite stages and bonding checks accepted by the profinite packet;*
2. *I is the* $\mathsf{InverseLimitMetric}^{\uparrow}$ *projection-metric row that reads those finite stages as a finite inverse-limit metric diagram;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that selects the visible quotient stages, projection comparisons, and metric observations;*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row reached from the inverse-limit metric route without choosing a quotient representative;*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *boundary row used to read the resulting compact metric surface through finite evidence;*
6. *H records componentwise* hsame *transports for F, I, W, M, K and the structural rows;*
7. *C records displayed* Cont *replay of the route*

$$F \longmapsto I \longmapsto W \longmapsto M \longmapsto K;$$

8. *P is the* Pkg *provenance over* ProfiniteMetricCompletion$^{\uparrow}$, ProfiniteCompletion$^{\uparrow}$, InverseLimitMetric$^{\uparrow}$, StreamName$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProfiniteMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, I, W, M, K, H, C, P, N$. It has no coordinate for quotient completion equality, selected infinite representatives, Haar measure, host group completion, or ambient compactness outside the displayed CompactMetric$^{\uparrow}$ *row.*

Theorem 7958.2 (Profinite metric completion NameCert obligations)**.** *For every accepted* ProfiniteMetricCompletio *carrier $Q = (F, I, W, M, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProfiniteMetricCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* ProfiniteCompletion$^{\uparrow}$ *finite-stage row, the* InverseLimitMetric$^{\uparrow}$ *projection-metric row, the* StreamName$^{\uparrow}$ *finite-window row, the* MetricCompletion$^{\uparrow}$ *handoff row, the* CompactMetric$^{\uparrow}$ *boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must pass through the route stored in C before any compact metric boundary is exposed.*

Proof. Project the carrier of Definition 7958.1. The mathematical rows are F, I, W, M, K: profinite finite-stage data, inverse-limit metric projection data, finite StreamName windows, the metric-completion handoff, and the compact metric boundary.

The replay row C fixes the order of consumption. A reader first inspects the profinite stage row F, then the inverse-limit metric row I, then the finite window row W, and only then reaches the completion handoff M and compact boundary K.

The transport, provenance, and naming rows H, P, N preserve and source that same finite route. Since no coordinate supplies quotient completion equality, a selected infinite representative, Haar measure, host group completion, or ambient compactness outside K, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProfiniteMetricCompletion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProfiniteMetricCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 7958.1 and Definition 7958.2 seed the finite profinite metric completion packet over profinite finite-stage data, inverse-limit metric rows, StreamName windows, MetricCompletion handoff, CompactMetric boundary, transport, replay, provenance, and naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProfiniteMetricCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected infinite representatives, Haar measure, host group completion, ambient compactness beyond the displayed CompactMetric row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-stage routing, projection-metric compatibility, StreamName window routing, MetricCompletion handoff exactness, CompactMetric boundary readback, transport stability, provenance, and local naming for BEDC.Derived.ProfiniteMetricCompletionUp.

Constructive story: A ProfiniteMetricCompletion$^{\uparrow}$ packet is generated from ProfiniteCompletion finite-stage rows, InverseLimitMetric projection rows, StreamName finite-window rows, MetricCompletion handoff rows, CompactMetric boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 7959

A Concrete Naming Certificate for $\mathsf{ConstructiveBaireCategory}^{\uparrow}$

Following Baire's theorem and Bishop-style constructive analysis, the $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ packet records the finite dense-open cylinder surface by which a completion-facing consumer may use a Baire-category argument without importing classical countable choice. It sits beside the Baire-space source of chapter 334, the Cantor-space sibling of chapter 1475, the finite observation windows of chapter 126, the metric carrier of chapter 57, and the terminal real-name seal of chapter 38. The packet names only displayed cylinder rows, dense-open admission rows, finite meet witnesses, metric-window replay, transport, provenance, and local naming.

Definition 7959.1 (Constructive Baire category carrier)**.** *A* $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, C, D, F, M, W, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{BaireSpace}^{\uparrow}$ *source row, exposing the finite prefix cylinder currently inspected by the packet.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *comparison row used when the same finite-prefix argument is read through binary cylinders.*
3. *D is the dense-open admission row, listing the displayed dense-open cylinder requests visible to the packet.*
4. *F is the finite meet-witness row, recording a common cylinder refinement for the displayed finite family from D.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *window row that bounds the cylinder read by a displayed metric observation.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to the inspected prefixes and metric observations.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for scalar radius or tolerance reads obtained only after S, C, D, F, M, W have been exposed.*
8. *H records componentwise* hsame *transport for the displayed source, cylinder, dense-open, finite-meet, metric, stream-window, and real-seal rows.*

9. *K records displayed* Cont *replay from finite cylinder requests through dense-open admission, finite meet witnesses, metric windows, stream windows, real sealing, transport, provenance, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBaireCategory}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBaireCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a classical choice sequence, an arbitrary countable meet of open sets, open-cover compactness, quotient equality of streams, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7959.2 (Constructive Baire category NameCert obligations)**.** *For every accepted* ConstructiveBaireCatego *carrier $B = (S, C, D, F, M, W, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBaireCategory}^{\uparrow}}$ *surface consists exactly of Baire-space source exposure, Cantor-space cylinder comparison when present, dense-open cylinder admission, finite meet-witness exposure, metric-window exposure,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure,* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer receives no classical countable-choice principle, no arbitrary countable intersection of open sets, no open-cover compactness theorem, no quotient stream equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 7959.1. The source and cylinder rows are S, C, D, F, the analytic observation rows are M, W, R, and the structural rows are H, K, P, N.

The dense-open obligations come from D and F. The row D lists only the dense-open cylinder requests already displayed by the packet, while F records a finite common refinement for that listed family.

The analytic handoff comes from M, W, R. The metric row bounds the cylinder read, the stream row records the finite prefix window that is actually inspected, and the real row seals scalar radius or tolerance reads only after those rows are visible.

Transport, replay, provenance, and local naming preserve this finite route. Since the carrier contains no coordinate for classical countable choice, arbitrary countable meets, open-cover compactness, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects belongs to the local NameCert surface. □

Theorem 7959.3 (Constructive Baire category countable-meet route)**.** *Every completion-facing or dense-open consumer of an accepted* $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ *packet that reads a countable-meet-shaped claim must factor the read through finite dense-open cylinder windows before reaching* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The exported route is the finite chain*

$$\mathsf{BaireSpace}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

read through $S, C, D, F, M, W, R, H, K, P, N$, not a classical countable intersection theorem or a choice-selected branch.

Proof. Start with the dense-open rows of Definition 7959.2. A consumer may inspect only the finite family listed in D, and the common refinement it receives is the displayed witness F.

Next read the cylinder through the metric and stream windows. The row M bounds the observation and W fixes the finite prefix window, so the route remains a finite cylinder argument rather than a branch selected from an infinite family.

Finally pass through the $\mathsf{Real}^{\uparrow}$ seal and the structural rows. The seal R is downstream of the finite cylinder data, and H, K, P, N replay only the same finite route. Thus the countable-meet-shaped read is available only through finite dense-open cylinder obligations. □

Theorem 7959.4 (Constructive Baire category choice boundary)**.** *An accepted* $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ *packet separates finite dense-open cylinder witnesses from classical choice principles. It may export the displayed finite meet witness F and its* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route, but it cannot export a choice function over arbitrary dense-open sequences, a selected generic point, an arbitrary open-cover compactness principle, quotient equality of streams, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Read the carrier coordinates. The only witness coordinate is F, and F is indexed by the displayed dense-open rows in D together with the finite cylinder sources S, C.

The downstream analytic rows do not add a selector. They only bind the cylinder to metric observations, finite stream windows, and the terminal real-name seal.

The structural rows preserve the same packet. Since B has no coordinate for a choice function, selected generic point, arbitrary open-cover compactness, quotient stream equality, or ambient Real completeness, those objects cannot be recovered from the local NameCert route. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ read exposes BaireSpace cylinders, CantorSpace comparison cylinders, dense-open admission rows, finite meet witnesses, MetricSpace windows, StreamName finite windows, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing consumer uses the packet.

Independence witness: baire-space, cantor-space, streamname, metricspace, real: $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite dense-open cylinder obligations, finite meet witnesses, metric windows, stream windows, Real sealing, transport, replay, provenance, and local naming while refusing classical countable choice, arbitrary countable meets, open-cover compactness, quotient stream equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBaireCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBaireCategory}^{\uparrow}$ packet is generated from displayed BaireSpace cylinders, CantorSpace comparison cylinders, dense-open cylinder rows, finite meet witnesses, MetricSpace windows, StreamName finite windows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7959.1, Definition 7959.2, Definition 7959.3, and Definition 7959.4 seed the finite constructive Baire-category NameCert surface over dense-open cylinder obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim classical countable choice, arbitrary countable meets of open sets, arbitrary open-cover compactness, quotient equality of streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open cylinder admission, finite meet-witness exactness, MetricSpace window routing, StreamName finite-window exposure, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBaireCategoryUp.

Chapter 7960

A Concrete Naming Certificate for $\mathsf{GoursatTheorem}^{\uparrow}$

Following Goursat and Ahlfors–Conway style complex analysis, the $\mathsf{GoursatTheorem}^{\uparrow}$ packet names the finite theorem-shaped route by which a holomorphic triangle contour is read as a zero-output contour-integral certificate. It consumes the contour-integral socket of chapter 583, the holomorphic input surface of chapter 68, and the complex coordinate carrier of chapter 39. The packet records only the displayed triangle contour, subdivision ledger, Riemann-sum modulus, zero-output seal, transport, replay, provenance, and local naming rows; it is not a host path integral theorem, not an open-cover compactness argument, not a primitive selector, and not a quotient equality principle.

Definition 7960.1 (Goursat theorem carrier)**.** *A* $\mathsf{GoursatTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, F, S, M, Z, H, C, P, N)$$

with the following displayed rows:

1. *T is the triangle-contour row. It records three* $\mathsf{Complex}^{\uparrow}$ *vertex histories, three oriented piecewise-linear edge rows, and the socket-facing boundary contour assembled from those edges.*
2. *F is the holomorphic input row, consumed from* $\mathsf{Holomorphic}^{\uparrow}$ *on a displayed neighbourhood of the triangle contour and its subdivision windows.*
3. *S is the finite subdivision ledger. Each entry stores a parent triangle, four child triangles, inherited orientation rows, and the internal-edge pairing visible to the contour socket.*
4. *M is the Riemann-sum modulus row. It records the unary mesh budget, local oscillation bounds for F, and the tolerance comparison attached to the leaves of S.*
5. *Z is the* $\mathsf{Complex}^{\uparrow}$ *zero-output row, displaying the target value* $0_{\mathsf{Complex}^{\uparrow}}$ *for the contour-integral socket read.*
6. *H records componentwise* hsame *transport for T, F, S, M, Z.*
7. *C records displayed* Cont *replay from the triangle contour request through the holomorphic input, subdivision ledger, Riemann-sum modulus, zero-output row, provenance, and local naming row.*
8. *P is the* Pkg *provenance over* $\mathsf{GoursatTheorem}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{GoursatTheorem}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, F, S, M, Z, H, C, P, N$. It has no coordinate for an ambient contour integral, selected antiderivative, host compactness proof, quotient path equality, or classical-choice schedule.

Theorem 7960.2 (Goursat theorem NameCert obligations)**.** *For every accepted* $\mathsf{GoursatTheorem}^{\uparrow}$ *carrier $G = (T, F, S, M, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GoursatTheorem}^{\uparrow}}$ *surface consists exactly of triangle-contour exposure, holomorphic-input exposure, subdivision-ledger exposure, Riemann-sum modulus exposure,* $\mathsf{Complex}^{\uparrow}$ *zero-output sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer receives no ambient contour-integral object, no selected primitive, no host compactness argument, no quotient equality of paths, and no classical-choice schedule.*

Proof. Project the carrier of Definition 7960.1. The analytic rows are exactly T, F, S, M, Z, and the structural rows are H, C, P, N.

The contour request enters through T and the contour-integral socket named in P. The holomorphic input row F supplies only the displayed local data on the triangle neighbourhood and subdivision windows.

The theorem-shaped content is carried by S, M, Z. The subdivision ledger exposes the child triangles and internal-edge pairings, the Riemann-sum modulus exposes the finite tolerance comparison on the leaves, and the zero-output row seals the socket read at $\mathsf{0}_{\mathsf{Complex}^{\uparrow}}$.

Transport, replay, provenance, and local naming preserve that finite route. Since the carrier contains no slot for the excluded ambient objects, the local NameCert surface is exhausted by the rows listed in the statement. □

Theorem 7960.3 (Goursat theorem triangle-subdivision route)**.** *Every contour-integral consumer that reads an accepted* $\mathsf{GoursatTheorem}^{\uparrow}$ *packet must factor the read through the finite route*

$$T, F \longmapsto S \longmapsto M \longmapsto Z \longmapsto H, C, P, N.$$

The internal-edge cancellations are visible only as entries of the subdivision ledger S, and the vanishing conclusion is visible only as the $\mathsf{Complex}^{\uparrow}$ *zero-output row Z. No consumer may replace this route by a host path integral theorem, a selected primitive, an open-cover compactness proof, quotient path equality, or a classical-choice schedule.*

Proof. Start with a public contour-integral read. By Definition 7960.2, the read first exposes the triangle contour row T and holomorphic input row F.

The read then enters S. Every child triangle and every internal opposite-edge pairing is a displayed ledger entry, so cancellation is inspected as finite subdivision data rather than as an ambient homology or path-quotient theorem.

The mesh and oscillation comparison is read from M. Its leafwise Riemann-sum bound is the only analytic estimate available before the conclusion is sealed.

Finally Z supplies the zero-output row, and H, C, P, N transport, replay, source, and name the same route. Because no excluded ambient object is a carrier coordinate, the displayed triangle-subdivision route is exhaustive. □

Falsifiable prediction: Every accepted $\mathsf{GoursatTheorem}^{\uparrow}$ read exposes triangle contour, holomorphic input, subdivision ledger, Riemann-sum modulus, Complex zero-output, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any contour-integral consumer may use the theorem-shaped vanishing route.

Independence witness: contourintegralsocket, holomorphic, complex, goursat: $\mathsf{GoursatTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a triangle contour, holomorphic input, subdivision ledger, Riemann-sum modulus, Complex zero-output seal, transport, replay, provenance, and local naming into one finite theorem-shaped packet.

Closure status for $\mathsf{NameCert}_{\mathsf{GoursatTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{GoursatTheorem}^{\uparrow}$ packet is generated from triangle-contour rows, Holomorphic input rows, finite subdivision ledgers, Riemann-sum modulus rows, a Complex zero-output seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GoursatTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7960.1, Definition 7960.2, and Definition 7960.3 seed the finite triangle-subdivision route for the Goursat theorem contour-integral socket read.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GoursatTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, the full Cauchy integral theorem, Morera theorem, analytic continuation, selected primitives, host contour integrals, open-cover compactness, quotient path equality, or classical choice.

Upgrade path: Obligation closure requires checked carrier admission, triangle-contour exactness, holomorphic-input locality, subdivision-ledger cancellation, Riemann-sum modulus control, Complex zero-output sealing, transport, replay, provenance, and local naming for BEDC.Derived.GoursatTheoremUp.

Chapter 7961

A Concrete Naming Certificate for $\mathsf{OscillationRank}^{\uparrow}$

Following classical oscillation-rank analyses of real-valued functions, the $\mathsf{OscillationRank}^{\uparrow}$ packet records the finite window by which a real-valued map on a displayed metric source exposes a bounded oscillation-stage comparison before any descriptive-analysis consumer reads rank data. It sits over the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{Met}^{\uparrow}$ route of chapter 57, the $\mathsf{CompactMetric}^{\uparrow}$ compact-window surface of chapter 129, the $\mathsf{ContinuousMap}^{\uparrow}$ source-map carrier of chapter 130, the Dini-window sibling of chapter 6110, and the finite rank-comparison sibling of chapter 6262. The packet names the finite inspection surface for a bounded oscillation-rank request; it is not a completed transfinite hierarchy, a quotient of functions, a selected ordinal rank for every map, or a host descriptive-set-theoretic completeness theorem.

Definition 7961.1 (OscillationRank carrier)**.** *An* $\mathsf{OscillationRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, K, F, M, R, D, W, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row naming the inspected metric carrier and its visible distance witnesses.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-window row naming the compact subwindow on which the oscillation request is inspected.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the real-valued source map visible on K.*
4. *M is the finite mesh row listing the sampled source points and tolerance scale used by the request.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which all value and bound reads pass after X, K, F, M have been displayed.*
6. *D is the* $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *or oscillation-derivative sibling row supplying the finite upper/lower comparison window used by the rank request.*
7. *W is the oscillation-threshold row recording the displayed rational bound and the two value observations compared at that bound.*
8. *S is the finite stage row naming the inspected derivative stage and its immediate predecessor window.*
9. *T is the rank-survival row saying whether the displayed oscillation set survives the named finite stage inside the visible window.*

10. *H records componentwise* hsame *transport for the metric source, compact window, continuous map, mesh, real seal, derivative window, threshold, stage, survival, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a bounded oscillation-rank request through $X, K, F, M, R, D, W, S, T$ and then through the structural rows.*
12. *P is the* Pkg *provenance over* $\mathsf{OscillationRank}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiniDerivativeWindow}^{\uparrow}$, $\mathsf{OscillationRankFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationRank}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed descriptive-set-theoretic rank hierarchy, countable or dependent choice, a host supremum over all stages, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7961.2 (OscillationRank NameCert obligations)**.** *For every accepted* $\mathsf{OscillationRank}^{\uparrow}$ *carrier $O = (X, K, F, M, R, D, W, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OscillationRank}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; metric-source compatibility for X; compact-window compatibility for K; continuous-map exposure through F; mesh and tolerance admission through M; terminal real sealing through R; Dini or oscillation-derivative window exposure through D; threshold exactness through W; finite-stage admission through S; rank-survival exactness through T; componentwise transport stability through H; continuation replay through C; provenance and local naming through P, N; and ledger refusal for completed rank hierarchies, choice principles, host suprema over all stages, quotient function equality, and ambient completeness.*

Proof. Project the carrier of Definition 7961.1. The accepted packet exposes exactly the rows $X, K, F, M, R, D, W, S, T, H, C, P, N$, so each local NameCert obligation must be read from one of those rows.

The source obligations are the route through X, K, F, M, R. The metric row fixes the visible source carrier, the compact row fixes the finite inspected window, the continuous-map row fixes the real-valued observations, the mesh row fixes the sampled tolerance schedule, and the real row seals every value read after those data have been displayed.

The rank-facing obligations are the route through D, W, S, T. The Dini or oscillation-derivative row supplies only the finite comparison window, the threshold row records the displayed oscillation bound, the stage row names the inspected derivative step, and the survival row records the finite yes/no comparison at that step.

Transport and replay preserve the same route. The rows H and C cannot add a completed hierarchy, a choice schedule, a host supremum, quotient function equality, or an ambient completeness principle, while P and N source and name only the listed coordinates. Hence the displayed obligations exhaust the local certificate surface. □

Theorem 7961.3 (OscillationRank finite-window handoff)**.** *Every descriptive-analysis consumer of an accepted* $\mathsf{OscillationRank}^{\uparrow}$ *carrier factors through the finite route*

$$X, K, F, M, R, D, W, S, T \longmapsto H, C, P, N.$$

Thus the consumer may inspect the metric source, compact window, continuous real-valued map, finite mesh, terminal real seal, Dini or oscillation-derivative window, oscillation threshold, displayed finite stage, and survival row, but it receives no full descriptive-set-theoretic rank completeness, no choice principle, no host supremum over all stages, no quotient function equality, and no ambient real-completion theorem.

Proof. Start with a consumer of the survival row T. Carrier admission makes T available only after the metric source X, compact window K, continuous map F, mesh M, real seal R, derivative window D, threshold row W, and finite stage row S have been displayed.

Replay the continuation row C. The request opens the source and compact-window data, reads the continuous-map observations through the real seal, consults the finite Dini or oscillation-derivative comparison window, and only then exposes the stage-survival row.

The handoff is therefore the stated finite route. Componentwise transport through H preserves the displayed rows, and P, N source and name only that packet. None of the excluded host objects can be recovered from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationRank}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7961.1, Definition 7961.2, and Definition 7961.3 seed the finite-window oscillation-rank certificate over displayed metric, compact, continuous-map, real, and Dini-window rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full descriptive-set-theoretic rank completeness, choice, host supremum over all stages, quotient function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, metric-source compatibility, compact-window compatibility, continuous-map exposure, mesh admission, real sealing, Dini-window exposure, threshold exactness, finite-stage admission, survival exactness, transport, replay, provenance, and local naming for BEDC.Derived.OscillationRankUp.

Constructive story: An $\mathsf{OscillationRank}^{\uparrow}$ packet is generated from MetricSpace, CompactMetric, ContinuousMap, RealUp, DiniDerivativeWindow, OscillationRankFunction, transport, continuation, provenance, and local NameCert rows.

Chapter 7962

A Concrete Naming Certificate for $\mathsf{MetacicResidualFuelRank}^{\uparrow}$

The $\mathsf{MetacicResidualFuelRank}^{\uparrow}$ packet realizes the residual-fuel branch of chapter 9370 as finite MetaCIC NameCert data. It sits beside the candidate-mediated boundary of chapter 5925, the normalization audit packet of chapter 750, the bounded fuel endpoint packet of chapter 5962, the residual-diamond readback packet of chapter 7893, and the residual substitution boundary of chapter 2181. The packet names the finite rank by which residual replay spends fuel before a normalization-facing or confluence-facing consumer may read an endpoint; it does not claim evidence-free strong normalization, global confluence, subject reduction, quotient syntax equality, a hidden normalizer, or MetaCIC kernel consistency.

Definition 7962.1 (MetaCIC residual fuel-rank carrier)**.** *A* $\mathsf{MetacicResidualFuelRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, K, F, Q, D, E, A, O, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed MetaCIC typed-term row whose residual replay is being inspected.*
2. *K is the candidate evidence row, displaying Candidate, ClosedCandidate, or CandidateSet membership before any strong-normalization read.*
3. *F is the residual fuel ledger, recording the visible fuel before and after each accepted residual step.*
4. *Q is the fuel-rank row, a finite lexicographic rank over residual depth, candidate scope, and bounded endpoint budget.*
5. *D is the residual descent row, pairing each replayed residual with a strict decrease in Q or a displayed timeout boundary.*
6. *E is the bounded endpoint row, read through* $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ *and the normalization audit surface.*
7. *A is the audit-handoff row, tying T, K, F, Q, D, E to* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$, $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$, $\mathsf{MetacicResidualDiamondReadback}^{\uparrow}$, *and* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$.
8. *O is the obstruction row for evidence-free strong normalization, global confluence, subject reduction, quotient syntax equality, hidden normalizers, and MetaCIC kernel consistency.*

9. *G is the TasteGate-facing row requiring later Lean* Nontrivial MetacicResidualFuelRank$^{\uparrow}$ *and* FieldFaithful MetacicResidualFuelRank$^{\uparrow}$ *instances for the same carrier fields.*

10. *H records componentwise* hsame *transport for term, candidate, fuel, rank, descent, endpoint, audit-handoff, obstruction, TasteGate-facing, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay through the route*

$$T, K, F, Q, D \longmapsto E, A \longmapsto O, G.$$

12. *P is the* Pkg *provenance over* MetacicResidualFuelRank$^{\uparrow}$, MetaCICCandidateStrongNormalization$^{\uparrow}$, MetaCICNormalizationAudit$^{\uparrow}$, BoundedNormalizationFuel$^{\uparrow}$, MetacicResidualDiamondReadback$^{\uparrow}$, MetaCICResidualS BHist, hsame, Cont, Pkg, *and* NameCert.

13. *N is the local* NameCert$_{\text{MetacicResidualFuelRank}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, F, Q, D, E, A, O, G, H, C, P, N$. It has no coordinate for a global strong-normalization theorem, full Church–Rosser proof, closed subject-reduction theorem, quotient syntax equality, hidden evaluator, or MetaCIC consistency claim.

Theorem 7962.2 (MetaCIC residual fuel-rank obligations)**.** *For every accepted* MetacicResidualFuelRank$^{\uparrow}$ *carrier $R = (T, K, F, Q, D, E, A, O, G, H, C, P, N)$, the local* NameCert$_{\text{MetacicResidualFuelRank}^{\uparrow}}$ *surface consists exactly of closed typed-term admission, candidate evidence, residual fuel-ledger exposure, finite rank exposure, descent-or-timeout exposure, bounded endpoint readback, audit handoff, obstruction retention, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7962.1. The term, candidate, fuel, rank, descent, endpoint, audit, obstruction, and TasteGate-facing rows are all visible coordinates.

The candidate row K prevents the read from becoming evidence-free strong normalization. The residual fuel ledger F and rank row Q make every replayed residual step fuel-sensitive before the endpoint row E can be read.

The descent row D either spends the displayed rank or records a timeout boundary. The audit-handoff row A then routes the endpoint read through the candidate strong-normalization, normalization audit, bounded fuel, residual readback, and residual-substitution sibling surfaces.

Rows O, G, H, C, P, N retain the refused endpoints, field-separation requirements, transport, replay, provenance, and local name. Since no carrier coordinate stores global normalization, confluence, subject reduction, quotient syntax equality, hidden evaluation, or kernel consistency, the listed rows exhaust the NameCert surface. □

Theorem 7962.3 (MetaCIC residual fuel-rank endpoint route)**.** *Every normalization-facing or confluence-facing consumer of an accepted* MetacicResidualFuelRank$^{\uparrow}$ *carrier must expose*

$$T, K, F, Q, D \longmapsto E, A \longmapsto H, C, P, N$$

before it can read an endpoint. The rank row Q is retained with the fuel ledger F, and each residual replay either decreases that rank through D or stops at the obstruction row O. The route exports a finite residual fuel-rank packet, not evidence-free strong normalization, global confluence, subject reduction, quotient syntax equality, a hidden normalizer, or MetaCIC kernel consistency.

Proof. Begin with the obligation surface of Definition 7962.2. It places the closed term T, candidate evidence K, fuel ledger F, rank row Q, and descent row D before endpoint readback.

The endpoint row E is bounded. It is consumed through the bounded-fuel sibling and the normalization audit surface named by A, so the consumer receives an audited residual route rather than an unrestricted normalizer.

If the residual replay continues, D records the strict rank decrease. If the replay cannot spend the displayed rank, the same row reaches the obstruction boundary O. In neither case can the route delete F or Q.

The structural rows H, C, P, N preserve only this displayed route. The TasteGate-facing row G records that later Lean instances must keep the fields noncollapsed before a checked carrier-alignment theorem can serve as a public realization. □

Falsifiable prediction: A purported $\mathsf{MetacicResidualFuelRank}^{\uparrow}$ consumer is refuted if it reaches a normalization endpoint while omitting the candidate row, residual fuel ledger, fuel-rank row, descent-or-timeout row, audit handoff, obstruction row, transport, replay, provenance, or local NameCert row.

Independence witness: metacic-candidate-strong-normalization, metacicnormalizationaudit, bounded-normalization-fuel, metacic-residual-diamond-readback, metacic-residual-substitution-boundary: none is bijective to $\mathsf{MetacicResidualFuelRank}^{\uparrow}$ because this packet jointly carries closed typed terms, candidate evidence, residual fuel, finite rank, descent-or-timeout data, bounded endpoint readback, audit handoff, obstruction retention, TasteGate-facing field separation, transport, replay, provenance, and local naming.

The Lean TasteGate file `lean4/BEDC/Derived/MetacicResidualFuelRankUp/TasteGate.lean` must provide explicit `Nontrivial` and `FieldFaithful` instances for $\mathsf{MetacicResidualFuelRank}^{\uparrow}$. Those instances must preserve $T, K, F, Q, D, E, A, O, G, H, C, P, N$ as distinct fields before any carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicResidualFuelRank}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicResidualFuelRank}^{\uparrow}$ packet is generated from a closed MetaCIC typed-term row, candidate evidence row, residual fuel ledger, finite fuel-rank row, descent-or-timeout row, bounded endpoint row, audit handoff, obstruction row, TasteGate-facing field-separation requirements, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicResidualFuelRank}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7962.1, Definition 7962.2, and Definition 7962.3 seed the finite residual fuel-rank packet over MetaCIC candidates, fuel ledgers, descent rows, bounded endpoints, audit handoffs, obstruction, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicResidualFuelRank}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization, global confluence, subject reduction, quotient syntax equality, hidden normalization, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence, residual fuel-ledger exactness, rank decrease or timeout exactness, bounded endpoint readback, audit handoff, obstruction retention, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetacicResidualFuelRankUp.

Chapter 7963

A Concrete Naming Certificate for $\mathsf{DyadicContinuityModulus}^{\uparrow}$

Following Bishop and Bridges on moduli of continuity for constructive real functions, the $\mathsf{DyadicContinuityModulus}^{\uparrow}$ packet records a finite dyadic tolerance-to-window service for L10 continuity consumers. It sits after the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the map-facing service of chapter 130. The packet names a local modulus table over displayed rows; it is not a host continuity function, a quotient-real equality principle, a selected external modulus, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 7963.1 (Dyadic continuity modulus carrier)**.** *A* $\mathsf{DyadicContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, S, R, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the finite dyadic precisions requested by a continuity consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row naming the finite input observations sufficient for those dyadic tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same source windows.*
4. *W is the response-window row naming the finite output observations certified for the requested tolerances.*
5. *H records componentwise* hsame *transport for E, S, R, W and the structural rows.*
6. *C records displayed* Cont *replay of the route*

$$E \longmapsto S \longmapsto R \longmapsto W.$$

7. *P is the* Pkg *provenance over* $\mathsf{DyadicContinuityModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tolerance, source-window, readback, response-window, transport, replay, provenance, and naming rows. It has no coordinate for

quotient real equality, selected external moduli, host functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7963.2 (DyadicContinuityModulus NameCert obligations)**.** *For an accepted* $\mathsf{DyadicContinuityModulus}^{\uparrow}$ *carrier* $M = (E, S, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicContinuityModulus}^{\uparrow}}$ *surface consists of exactly five obligations: carrier admission for the eight displayed rows; tolerance-window stability binding each* E *row to its source window* S*; regular readback compatibility binding* S *to* R*; response-window exactness binding* R *to* W *for the requested dyadic precision; and local provenance plus naming over* H, C, P, N*. These obligations export no quotient real equality, selected external modulus, host continuity function, countable-choice schedule, or ambient Real completeness.*

Proof. Carrier admission is projection from Definition 7963.1. The packet exposes exactly the tolerance, source-window, readback, response-window, transport, replay, provenance, and local naming rows.

Tolerance-window stability reads E, S, H, C. A transported dyadic tolerance request remains attached to the same finite source-window row, and the replay route prevents the request from being served by a hidden host modulus.

Regular readback compatibility reads S, R. The source window is inspected only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback, so the input side of the modulus remains a finite BEDC readback route.

Response-window exactness reads R, W after the tolerance row. The output response window is certified only for the dyadic precision named in E and the readback named in R.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate supplies quotient real equality, selected external moduli, host functions, countable choice, or ambient completeness, the five obligations exhaust the local NameCert surface. □

Theorem 7963.3 (DyadicContinuityModulus unary-window induction)**.** *Let* $M = (E, S, R, W, H, C, P, N)$ *be an accepted* $\mathsf{DyadicContinuityModulus}^{\uparrow}$ *carrier. A finite chain of dyadic tolerance refinements read from* E *proceeds by unary-window induction: the base window is the displayed source window* S *with readback* R*, and each successor refinement tightens the same dyadic tolerance ledger before exposing the next response window* W*. The induction stays inside* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and the structural rows; it exports no quotient real equality, selected external modulus, host function, countable choice, or ambient Real completeness.*

Proof. For the base window, project the carrier. The source window S and readback R are already paired with the initial dyadic tolerance row E, so the first response window W is read without any external continuity function.

For the successor step, apply the tolerance-window stability and regular readback clauses of Definition 7963.2. A refined dyadic request remains attached to the transported source window and regular rational readback.

Now apply response-window exactness. The next response window is certified only after the refined tolerance and the corresponding readback have been displayed, so the refinement step is unary over the visible window row rather than a choice over an infinite family of representatives.

Induction over the finite displayed chain gives the claimed route. The structural rows H, C, P, N preserve the same finite packet at every step, and the absent coordinates exclude quotient equality, selected moduli, host functions, countable choice, and ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicContinuityModulus}^{\uparrow}}$.

Constructive story: A DyadicContinuityModulus$^{\uparrow}$ packet is generated from a dyadic tolerance row, source-window row, regular rational readback row, response-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{DyadicContinuityModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7963.1, Definition 7963.2, and Definition 7963.3 seed the finite dyadic tolerance-to-window route over DyadicRatCore, StreamName, RegSeqRat, Real, ContinuousMap, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicContinuityModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected external moduli, host continuity functions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window stability, regular readback compatibility, response-window exactness, provenance, and local naming for the dyadic continuity modulus packet.

Chapter 7964

A Concrete Naming Certificate for $\mathsf{RealFunctionSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the computable-analysis reading of represented real functions, the $\mathsf{RealFunctionSpace}^{\uparrow}$ packet records a finite function-space window rather than a host set of all functions. It sits over the terminal real-name seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the continuous-map carrier of chapter 130, and the pointwise comparison route of chapter 4139. The packet gives L10 Real-and-Completion consumers a named finite graph-window surface for real-valued functions before any host function extensionality, quotient function space, selected completion representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 7964.1 (Real function-space carrier)**.** *A* $\mathsf{RealFunctionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, W, A, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-domain row naming the displayed input window on which the function-space read is inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *graph-window schedule listing the finite input and output-name windows opened by the request.*
3. *A is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row assigning the inspected graph-window data to a continuous real-valued map read through the same source window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *output readback attached to the displayed output names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for each inspected output value.*
6. *M is the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *comparison row, comparing two displayed function-space reads only through finite pointwise windows.*
7. *H records componentwise* hsame *transport for the source window, graph-window schedule, continuous-map row, regular readback, Real seal, pointwise metric, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite graph-window request through X, W, A, R, E, M, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RealFunctionSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{PointwiseStreamMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealFunctionSpace}^{\uparrow}}$ *row.*

Definition 7964.2 (Real function-space classifier)**.** *The classifier of an accepted* $\mathsf{RealFunctionSpace}^{\uparrow}$ *packet compares two carriers only through the displayed rows $X, W, A, R, E, M, H, C, P, N$. It requires the same source-window row, the same graph-window schedule, the same continuous-map read, the same regular-rational output readback, the same terminal real seals, and the same pointwise stream-metric comparison up to componentwise* hsame *transport and displayed* Cont *replay. It has no coordinate for host function extensionality, quotient equality of functions, a completed function space, a selected pointwise limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7964.3 (Real function-space NameCert obligations)**.** *For every accepted* $\mathsf{RealFunctionSpace}^{\uparrow}$ *carrier $F = (X, W, A, R, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealFunctionSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed source window, graph-window schedule, continuous-map row, regular readback, Real seal, pointwise metric, transport, replay, provenance, and local naming rows;*
2. *graph-window exactness, requiring every inspected input/output pair to pass through X, W, A before any output value is read;*
3. *output seal nonescape, requiring R and E to remain downstream of the same displayed graph-window schedule;*
4. *pointwise comparison stability, requiring M to compare function-space reads only through finite pointwise windows and componentwise* hsame *transport;*
5. *ledger rows for displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *name.*

No accepted obligation row is stored outside $X, W, A, R, E, M, H, C, P, N$.

Proof. Project the accepted carrier of Definition 7964.1. The mathematical rows are X, W, A, R, E, M, and the structural rows are H, C, P, N.

The graph-window and output-seal obligations are read in order. A consumer first sees the source window X, graph-window schedule W, and continuous-map row A; only then can it open the regular readback R and terminal real seal E.

The comparison obligation is read through M. Since M stores only finite pointwise windows, it cannot provide host function extensionality, quotient equality, a completed function space, a selected pointwise limit, countable choice, or ambient completeness.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Thus the five listed obligation rows exhaust the local certificate surface. □

Theorem 7964.4 (Real function-space finite graph-window handoff)**.** *Every L10 Real-and-Completion consumer of an accepted* $\mathsf{RealFunctionSpace}^{\uparrow}$ *carrier factors through the finite route*

$$X, W, A \longmapsto R, E \longmapsto M, H, C, P, N.$$

The route exposes source-domain windows, graph-window schedules, continuous-map rows, regular rational output readback, terminal real seals, pointwise stream-metric comparisons, componentwise transport, displayed replay, provenance, and local naming before any later calculus, uniform-continuity, or completion consumer reads the function-space packet. It exports no host function extensionality, quotient function space, selected pointwise limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier. Carrier admission and the NameCert obligations force every graph-window request through X, W, A before output values become visible.

The output values are then read through R, E. The regular-rational row supplies finite approximants attached to the same graph windows, and the terminal Real row seals only those displayed output reads.

Pointwise comparison and structural replay are downstream of the same route. The row M compares only finite pointwise windows, while H, C, P, N preserve, replay, source, and name the packet.

No carrier coordinate remains for host function equality, quotient completion, selected pointwise limits, countable choice, or ambient real completeness. Hence every L10 consumer factors through the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{RealFunctionSpace}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ output seal or $\mathsf{PointwiseStreamMetric}^{\uparrow}$ comparison without displaying $X, W, A, R, E, M, H, C, P, N$ refutes the carrier surface.

Independence witness: real, streamname, regseqrat, continuousmap, pointwise-stream-metric: no listed sibling supplies the whole packet $F = (X, W, A, R, E, M, H, C, P, N)$, because each sibling contributes only the real seal, stream window, regular readback, continuous-map, or pointwise comparison face while this chapter binds them into one finite function-space route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealFunctionSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealFunctionSpace}^{\uparrow}$ packet is generated from a source-domain window, StreamName graph windows, a ContinuousMap row, RegSeqRat output readback, RealUp seals, PointwiseStreamMetric comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealFunctionSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7964.1, Definition 7964.2, Definition 7964.3, and Definition 7964.4 seed the finite graph-window function-space route over RealUp, StreamNameUp, RegSeqRatUp, ContinuousMapUp, and PointwiseStreamMetricUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealFunctionSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient function spaces, selected pointwise limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, graph-window exactness, output seal nonescape, pointwise comparison stability, transport, replay, provenance, and local naming for BEDC.Derived.RealFunctionSpaceUp.

Chapter 7965

A Concrete Naming Certificate for DyadicModulusChoiceFreeRefinement$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicModulusChoiceFreeRefinement$^{\uparrow}$ packet records the finite refinement route by which dyadic tolerance requests are sharpened before a regular rational sequence or Real-facing seal is read. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the choice-free selector boundary of chapter 695. The packet gives L10 Real and Completion consumers a visible dyadic-modulus refinement surface without quotient reals, selected host limits, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 7965.1 (Dyadic modulus choice-free refinement carrier)**.** *A* DyadicModulusChoiceFreeRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, Q, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger naming the requested dyadic scale and the refined scale consumed downstream;*
2. *Q is the finite refinement certificate proving that the downstream tolerance is no coarser than the requested dyadic bound;*
3. *W is the finite window schedule that determines which observations are inspected at the refined scale;*
4. *S is the* StreamName$^{\uparrow}$ *observation family opened by W;*
5. *R is the* RegSeqRat$^{\uparrow}$ *witness row assembled from the finite observations and the refined dyadic tolerance;*
6. *E is the* Real$^{\uparrow}$ *seal boundary available only after the regular rational witness row has been displayed;*
7. *H records componentwise* hsame *transport for dyadic, window, stream, regular, Real-seal, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay for the same route;*
9. *P is the* Pkg *provenance over* DyadicModulusChoiceFreeRefinement$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ChoiceFreeDiagonalSelector$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, Q, W, S, R, E, H, C, P, N$. It has no coordinate for a hidden selector, quotient real, selected limit, countable-choice schedule, or ambient real-completeness principle.

Lean4 (definition) `BEDC.Derived.DyadicModulusChoiceFreeRefinementUp.DyadicModulusChoiceFreeRefinementUp`

Theorem 7965.2 (Dyadic modulus choice-free refinement obligations)**.** *For the carrier of Definition 7965.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row D, refinement certificate Q, finite window schedule W, StreamName observations S, RegSeqRat witness R, RealUp seal E, componentwise transport H, displayed replay C, provenance P, and local naming row N.*

Proof. Project the displayed carrier. The analytic content rows are D, Q, W, S, R, E, and they occur in the order needed by a Real-facing modulus consumer.

The dyadic row D and refinement row Q determine the tolerated scale before any observation is opened. The finite window W then bounds the StreamName reads S, and the RegSeqRat witness R is assembled only from those displayed rows before the RealUp seal E is available.

The structural rows H, C, P, N preserve and name the same route. Since the carrier contains no coordinate for hidden choice, quotient equality, selected limits, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 7965.3 (Dyadic modulus choice-free refinement stage induction)**.** *Let $M = (D, Q, W, S, R, E, H, C, P, N$ be an accepted* $\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}$ *carrier whose window schedule is displayed as finite stages W_j. Induction over the stages preserves the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and keeps every successor refinement inside the displayed dyadic tolerance, observation, regular-witness, and Real-seal rows. No stage may introduce a quotient real, selected host limit, hidden cofinal selector, countable-choice schedule, unbounded observation scan, or ambient completeness claim for $\mathsf{Real}^{\uparrow}$*.*

Proof. For the base stage, inspect the first displayed window row. Its dyadic scale is admitted by D, Q, its finite observations are exactly the corresponding part of S, and the partial regular witness is read through the same RegSeqRat route.

For the successor stage, assume the displayed route has been preserved through W_j. The next finite window refines the dyadic tolerance by the same certificate row Q, opens only the next displayed StreamName observations, and repacks the enlarged finite readback into the RegSeqRat witness row before any RealUp seal is read.

The transport, replay, provenance, and local naming rows are restrictions of H, C, P, N to the same finite route. Thus the induction invariant is the displayed path from dyadic tolerance to StreamName observations, then to RegSeqRat witness and RealUp seal, with no carrier coordinate for the refused endpoints. □

Falsifiable prediction: Every accepted $\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}$ read is refuted if it reaches a RegSeqRat or RealUp consumer without first displaying the dyadic tolerance row, refinement certificate, finite window schedule, StreamName observations, transport, replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, choicefreediagonalselector: no listed sibling is bijective to $\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}$ because this packet jointly carries dyadic refinement, finite observation scheduling, regular rational handoff, Real seal, choice-free selector provenance, transport, replay, and local naming as one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, refinement certificate, finite window schedule, StreamName observations, RegSeqRat witness, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7965.1, Definition 7965.2, and Definition 7965.3 seed the choice-free dyadic modulus refinement route over DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulusChoiceFreeRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient real equality, selected host limits, countable choice, unbounded observation scans, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a public standard bridge.

Upgrade path: Formal closure requires checked carrier admission, dyadic refinement exactness, finite-window pullback, StreamName readback, RegSeqRat witness assembly, RealUp seal delivery, replay stability, and nonescape for the same finite packet.

Chapter 7966

A Concrete Naming Certificate for MetaCICResidualConfluenceFuel$^{\uparrow}$

The MetaCICResidualConfluenceFuel$^{\uparrow}$ packet realizes the residual-fuel inspection boundary named by the MetaCIC confluence audit map in chapter 9378. It sits beside the parallel-diamond frontier of chapter 4238, the critical-path packet of chapter 765, the normalization frontier of chapter 892, the residual-substitution boundary of chapter 2181, and the confluence audit packet of chapter 735. The carrier records bounded critical-pair and residual-substitution inspection data without claiming arbitrary closed Church–Rosser, subject reduction, quotient equality of terms, evaluator equality, or kernel consistency.

Definition 7966.1 (MetaCIC residual confluence fuel carrier)**.** *A* MetaCICResidualConfluenceFuel$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, K, R, J, B, S, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row naming the closed source and the bounded syntax fragment under inspection;*
2. *K is the critical-pair row naming the local peak or overlap to be checked;*
3. *R is the residual-substitution boundary row retained from the conditional confluence interface;*
4. *J is the candidate local-join row, available only under the displayed residual boundary;*
5. *B is the finite fuel budget for residual, substitution, and critical-pair inspection;*
6. *S is the closed checker or normal-fragment sibling row consumed by the bounded inspection;*
7. *O is the obstruction row retaining refused endpoints when the residual boundary is not discharged;*
8. *H records componentwise* hsame *transport for typed-window, critical-pair, residual, join, budget, checker, obstruction, replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay through the same rows;*
10. *P is the* Pkg *provenance over* MetaCICResidualConfluenceFuel$^{\uparrow}$*, the five cited MetaCIC sibling packets,* BHist*,* hsame*,* Cont*, and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $T, K, R, J, B, S, O, H, C, P, N$. It does not identify a bounded inspection with a full parallel diamond, closed Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality, or kernel-consistency claim.

Theorem 7966.2 (MetaCIC residual fuel critical-pair boundary)**.** *Every accepted* $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ *carrier exposes the critical-pair row K, residual-substitution boundary R, candidate local-join row J, and finite fuel row B before any confluence-facing consumer may read the checker row S or obstruction row O. The inspection is therefore bounded by*

$$T, K, R, J, B, S, O, H, C, P, N$$

rather than by an ambient closed Church–Rosser endpoint.

Proof. Project the carrier of Definition 7966.1. The typed source, local peak, residual boundary, candidate join, and fuel budget are the only rows that can authorize a bounded residual inspection.

The checker row S is downstream of those rows. It can certify a bounded inspection result for the displayed source and fuel, but it cannot erase the residual-substitution boundary R or the obstruction row O.

Transport through H, replay through C, provenance through P, and local naming through N preserve that finite boundary. Hence a confluence-facing consumer receives a fuel-bounded inspection surface, not a premise-free Church–Rosser theorem. □

Theorem 7966.3 (MetaCIC residual fuel non-escape)**.** *An accepted* $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ *packet cannot export arbitrary closed Church–Rosser, full parallel diamond, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency. Any such read must first supply a separate discharge for the residual-substitution boundary and the typed critical-pair frontier named by R and K.*

Proof. Use the critical-pair boundary theorem. It forces every confluence-facing read through the typed window, critical-pair row, residual-substitution row, candidate join, finite fuel, checker row, and obstruction row before structural replay is available.

The refused endpoints require data that are not stored in the carrier. A full parallel diamond needs a discharged residual-substitution compatibility input; subject reduction needs its own typing-preservation surface; evaluator equality, quotient equality, and kernel consistency require separate public commitments.

Since H, C, P, N only transport, replay, source, and name the same finite inspection packet, none of those stronger endpoints can be recovered from $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ alone. □

The Lean TasteGate follow-up path for $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ must include explicit $\mathsf{Nontrivial}\ \mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ instances. These instances must distinguish the typed window, critical-pair, residual-substitution boundary, candidate join, fuel budget, checker, obstruction, transport, replay, provenance, and local NameCert rows.

Falsifiable prediction: If the residual-substitution boundary or critical-pair row is absent, a bounded MetaCIC residual-fuel inspection cannot be cited as a confluence endpoint.

Independence witness: metacic-parallel-diamond-frontier, metaciccriticalpath, metacicnormalizationfrontier, metacic-residual-substitution-boundary, metacicconfluenceauditpacket: none is bijective because $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ jointly carries typed-window, critical-pair, residual-boundary, local-join, finite-fuel, checker, obstruction, transport, replay, provenance, and local NameCert rows as one bounded inspection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}$ packet is generated from a typed-window row, critical-pair row, residual-substitution boundary, candidate local-join row, finite fuel budget, closed checker row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7966.1, Definition 7966.2, and Definition 7966.3 seed the bounded critical-pair and residual-substitution inspection surface for the MetaCIC confluence boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceFuel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary closed Church–Rosser, full parallel diamond, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or a public standard bridge.

Upgrade path: Formal closure requires checked carrier admission, critical-pair exactness, residual-boundary retention, local-join fuel accounting, checker replay, obstruction preservation, TasteGate nontriviality, and field faithfulness.

Chapter 7967

A Concrete Naming Certificate for $\mathsf{RegSeqRatFunctorialMap}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of regular Cauchy sequences and uniformly continuous maps, the $\mathsf{RegSeqRatFunctorialMap}^{\uparrow}$ packet records the finite route by which a modulus-controlled map acts on regular Cauchy rational names before any Real-facing consumer reads the transported name. It sits over the dyadic tolerance ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the continuous-map carrier of chapter 130. The packet names a finite functorial action surface; it does not quotient regular names, select limits, assert ambient completeness, or export a host function-space equality theorem.

Definition 7967.1 (RegSeqRat functorial-map carrier)**.** *A* $\mathsf{RegSeqRatFunctorialMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, S, Q, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus-controlled map row, including the displayed input tolerance and output tolerance schedule.*
2. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose observations feed the map.*
3. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational name row read through S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the dyadic input and output budgets used by M.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, S, Q, D have been displayed.*
6. *W is the transported finite-window row listing the output observations certified by the modulus schedule.*
7. *H records componentwise* hsame *transport for the map, source windows, source regular name, dyadic ledger, terminal seal, transported windows, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a regular Cauchy rational-name read through the modulus-controlled map and the transported output windows.*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRatFunctorialMap}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatFunctorialMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, Q, D, R, W, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected representative, countable choice over moduli, host real equality, function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7967.2 (RegSeqRat functorial-map NameCert obligations)**.** *For every accepted* $\mathsf{RegSeqRatFunctorialMap}$ *carrier $F = (M, S, Q, D, R, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatFunctorialMap}^{\uparrow}}$ *surface consists exactly of modulus-controlled map exposure, source* $\mathsf{StreamName}^{\uparrow}$ *finite-window scheduling, source* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, transported output-window admission, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger refusal for quotient streams, selected limits, countable choice over moduli, host real equality, function extensionality, and ambient completeness.*

Proof. Project the carrier of Definition 7967.1. The accepted packet displays exactly the map row M, the source window row S, the source regular-rational row Q, the dyadic ledger D, the terminal seal R, and the transported output windows W.

The structural rows H, C, P, N transport, replay, source, and name only those displayed rows. Thus every local certificate obligation is read through $M, S, Q, D, R, W, H, C, P, N$.

The refusal clauses are coordinate exclusions. Quotient streams, selected representatives, countable choice over moduli, host real equality, function extensionality, and ambient completeness are not rows of F, so the local NameCert cannot export them. □

Theorem 7967.3 (RegSeqRat functorial-map regularity preservation)**.** *Every Real-facing consumer of an accepted* $\mathsf{RegSeqRatFunctorialMap}^{\uparrow}$ *carrier receives the transported regular Cauchy rational name only through the finite route*

$$M, S, Q, D, W \longmapsto R, H, C, P, N.$$

In particular, the output windows preserve the displayed regularity budget supplied by the $\mathsf{ContinuousMap}^{\uparrow}$ *modulus and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Start with the output-window row W. By the carrier definition, W is admitted only after the modulus-controlled map M, the source windows S, the source regular-rational row Q, and the dyadic tolerance ledger D have been displayed.

Read the continuation row C. It replays a regular Cauchy rational-name request through the same modulus schedule and dyadic budgets before the terminal seal R is exposed.

Componentwise transport through H preserves the displayed rows, and P, N source and name only that packet. Hence the consumer obtains regularity preservation along the stated finite route, without quotienting names, selecting a limit, invoking choice over moduli, or importing ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRatFunctorialMap}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegSeqRatFunctorialMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7967.1, Definition 7967.2, and Definition 7967.3 seed the finite functorial-map route from modulus-controlled maps on regular Cauchy rational names to Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegSeqRatFunctorialMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice over moduli, host real equality, function extensionality, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-controlled map exposure, source window scheduling, RegSeqRat readback, DyadicRatCore tolerance accounting, transported output-window admission, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegSeqRatFunctorialMapUp.

Constructive story: A RegSeqRatFunctorialMap$^{\uparrow}$ packet is generated from a ContinuousMap modulus row, StreamName source windows, RegSeqRat regular rational readback, DyadicRatCore tolerance accounting, transported output windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7968

A Concrete Naming Certificate for $\mathsf{SchottkyGroup}^{\uparrow}$

Following Klein's and Poincare's Schottky-group construction in the disk model, the $\mathsf{SchottkyGroup}^{\uparrow}$ packet records the finite ping-pong data by which paired boundary domains generate a free hyperbolic action. It realizes the hyperbolic observer-expansion route of chapter 9463 through the disk automorphism surface of chapter 6776, the metric readback of chapter 3474, the limit-set boundary row of chapter 7764, and the Fuchsian-action packet of chapter 7282. The packet names finite generator, inverse-pair, boundary-domain, and shadow rows; it does not import a host Kleinian group, quotient surface, selected fundamental domain, classical compactness theorem, or ambient boundary completion.

Definition 7968.1 (Schottky group carrier)**.** *A* $\mathsf{SchottkyGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, A, D, M, L, F, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, listing paired generator names and their inverse partners.*
2. *A is the* $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$ *row for the displayed disk automorphisms and inverse replays.*
3. *D is the boundary-domain row, listing the disjoint source and target arcs used by the ping-pong read.*
4. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row comparing the displayed disk and boundary windows used by the generator action.*
5. *L is the* $\mathsf{HyperbolicLimitSet}^{\uparrow}$ *row naming the finite boundary-shadow observations that later limit-set consumers may inspect.*
6. *F is the* $\mathsf{FuchsianGroupAction}^{\uparrow}$ *row, recording the word-action replay over the same automorphism and boundary-domain data.*
7. *B is the boundary-shadow ledger that records which paired domain is sent outside its complement by each displayed generator.*
8. *H records componentwise* hsame *transport for the generators, automorphisms, domains, metric row, limit-set row, action row, boundary-shadow ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite word or boundary-domain request through G, A, D, M, L, F, B, H.*
10. *P is the* Pkg *provenance over* $\mathsf{SchottkyGroup}^{\uparrow}$, $\mathsf{PoincareDiskAutomorphism}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicLimitSet}^{\uparrow}$, $\mathsf{FuchsianGroupAction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchottkyGroup}^{\uparrow}}$ *row.*

Definition 7968.2 (Schottky ping-pong classifier)**.** *The classifier of an accepted* $\mathsf{SchottkyGroup}^{\uparrow}$ *packet compares two carriers only through $G, A, D, M, L, F, B, H, C, P, N$. It requires the same displayed generator pairing, the same disk-automorphism row, the same paired boundary domains, the same hyperbolic-metric comparison surface, the same limit-set observation row, the same Fuchsian word-action replay, and the same boundary-shadow ledger up to componentwise* hsame *transport and displayed* Cont *replay. It has no coordinate for a host Kleinian group, quotient surface, selected fundamental domain, hidden boundary compactification, countable choice, or ambient hyperbolic completion.*

Theorem 7968.3 (Schottky group NameCert obligations)**.** *For every accepted* $\mathsf{SchottkyGroup}^{\uparrow}$ *carrier $S = (G, A, D, M, L, F, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SchottkyGroup}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; generator-pair admission for G; disk-automorphism and inverse replay through A; boundary-domain disjointness and ping-pong target exposure through D, B; hyperbolic metric compatibility through M; limit-set observation through L; Fuchsian word-action replay through F; componentwise transport through H; displayed replay through C; and provenance and local naming through P, N.*

Proof. Project the accepted carrier. The generator and automorphism rows are G, A, the boundary and metric rows are D, M, B, the downstream hyperbolic rows are L, F, and the structural rows are H, C, P, N.

The ping-pong obligation is read through D, B. The row D displays the paired source and target boundary domains, while B records the finite shadow relation saying how each generator moves the complement of one domain into its paired target domain.

The metric, limit-set, and action obligations are then read through M, L, F. These rows keep hyperbolic distance comparisons, boundary accumulation observations, and word-action replay attached to the same finite automorphism data.

Finally H, C, P, N transport, replay, source, and name the packet. Since the carrier has no coordinate for a host Kleinian group, quotient surface, selected fundamental domain, hidden compactification, choice principle, or ambient completion, the listed rows exhaust the local certificate surface. □

Theorem 7968.4 (Schottky ping-pong boundary-domain route)**.** *Every hyperbolic boundary or Fuchsian-action consumer of an accepted* $\mathsf{SchottkyGroup}^{\uparrow}$ *carrier factors through the finite route*

$$G, A, D, B \longmapsto M \longmapsto L, F \longmapsto H, C, P, N.$$

Thus the consumer reads paired generators, Poincare disk automorphisms, boundary domains, ping-pong shadow rows, hyperbolic metric comparisons, limit-set observations, Fuchsian word-action replay, transport, displayed replay, provenance, and local naming before it treats the packet as a Schottky group. The route exports no host Kleinian group, quotient Riemann surface, selected fundamental domain, open-cover compactness, countable choice, or ambient boundary completion.

Proof. Start with the NameCert obligations. They place G, A, D, B before any metric, limit-set, or action consumer can read the group packet.

The metric row M is downstream of those ping-pong rows. It compares only the displayed disk and boundary windows attached to the automorphism data, so a consumer cannot replace the route by a host-level hyperbolic action.

The limit-set and Fuchsian-action rows L, F are then read from the same finite route. The limit-set observations are boundary-shadow rows, and the action replay is a displayed word-action ledger over the accepted automorphism pairs.

Structural transport and replay through H, C, P, N preserve the same packet. Because no excluded host object occurs among the carrier coordinates, every consumer factors through the displayed boundary-domain route. □

Falsifiable prediction: An accepted $\mathsf{SchottkyGroup}^{\uparrow}$ consumer that reaches a $\mathsf{HyperbolicLimitSet}^{\uparrow}$ or $\mathsf{FuchsianGroupAction}^{\uparrow}$ read without displaying $G, A, D, M, L, F, B, H, C, P, N$ refutes the carrier surface.

Independence witness: poincare-disk-automorphism, hyperbolic-metric, hyperbolic-limit-set, fuchsian-group-action: no listed sibling supplies the whole Schottky packet because each contributes only disk automorphisms, metric comparisons, boundary accumulation, or action replay while this chapter binds the finite ping-pong boundary-domain route in one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SchottkyGroup}^{\uparrow}}$.

Constructive story: A $\mathsf{SchottkyGroup}^{\uparrow}$ packet is generated from paired generator rows, PoincareDiskAutomorphism rows, paired boundary domains, HyperbolicMetric comparisons, HyperbolicLimitSet boundary observations, FuchsianGroupAction word replay, boundary-shadow ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchottkyGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7968.1, Definition 7968.2, Definition 7968.3, and Definition 7968.4 seed the finite Schottky ping-pong route over PoincareDiskAutomorphismUp, HyperbolicMetricUp, HyperbolicLimitSetUp, and FuchsianGroupActionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchottkyGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Kleinian groups, quotient Riemann surfaces, selected fundamental domains, open-cover compactness, countable choice, ambient boundary completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, generator-pair admission, disk-automorphism replay, boundary-domain ping-pong exactness, metric compatibility, limit-set observation, Fuchsian action replay, transport, replay, provenance, and local naming for BEDC.Derived.SchottkyGroupUp.

Chapter 7969

A Concrete Naming Certificate for $\mathsf{BromwichInversionWindow}^{\uparrow}$

Following Bromwich and Doetsch on inverse Laplace transforms, the $\mathsf{BromwichInversionWindow}^{\uparrow}$ packet records only the finite vertical-line window data that BEDC consumers may inspect before any inversion theorem is available. It sits after the transform packet of chapter 3649, the contour socket of chapter 583, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite abscissa, height window, rational quadrature readback, contour-socket handoff, Real output seal, transport, replay, provenance, and local NameCert row; it does not assert a host Bromwich inversion theorem, analytic continuation of a transform, selected improper integrals, quotient path equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 7969.1 (Bromwich inversion window carrier)**.** *A* $\mathsf{BromwichInversionWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, A, W, Q, S, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LaplaceTransform}^{\uparrow}$ *input row whose transform data are being read.*
2. *A is the abscissa row naming the finite vertical line* $\mathrm{Re}(s) = a$ *inspected by the window.*
3. *W is the height-window row naming the finite interval* $[-T, T]$ *on that vertical line.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite quadrature samples attached to W.*
5. *S is the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row that supplies the contour-window socket used by the read.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the windowed inversion value returned to Real-facing consumers.*
7. *H records componentwise* hsame *transport for L, A, W, Q, S, R and the structural rows.*
8. *C records displayed* Cont *replay of the route*

$$L \longmapsto A \longmapsto W \longmapsto Q \longmapsto S \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{BromwichInversionWindow}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BromwichInversionWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed transform, vertical-line, height-window, rational readback, contour socket, Real seal, transport, replay, provenance, and naming rows. It has no coordinate for a host inversion theorem, ambient complex analytic continuation, quotient contour equality, selected improper integral, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7969.2 (BromwichInversionWindow NameCert obligations)**.** *For an accepted* $\mathsf{BromwichInversionWindow}$ *carrier* $B = (L, A, W, Q, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BromwichInversionWindow}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier admission for the ten displayed rows; transform-source compatibility binding* L *to the same window; vertical-line stability binding* A *and* W *under componentwise transport; regular readback exactness for the quadrature row* Q*; contour-socket handoff through* S*; and local provenance plus naming over* H, C, P, N*. These obligations export no host Bromwich inversion theorem, analytic-continuation theorem, selected improper integral, quotient contour equality, countable-choice schedule, or ambient Real completeness.*

Proof. Carrier admission is projection from Definition 7969.1. The packet exposes exactly the transform row, abscissa row, height window, regular rational readback, contour socket, Real seal, transport, replay, provenance, and local naming rows.

Transform-source compatibility reads L, A, W. The window may inspect only the finite transform data attached to the displayed abscissa and height rows, so it cannot call an external inverse-transform operator.

Vertical-line stability reads A, W, H, C. A transported abscissa row and height row remain the same finite vertical-line window through the displayed replay route.

Regular readback exactness reads Q after W. The finite samples attached to the height window are returned only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row, not through a selected improper integral.

The contour-socket handoff reads S and then the Real seal R. The socket row supplies the contour-window boundary, and the terminal seal is the only Real-facing output coordinate.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate supplies a host inversion theorem, analytic continuation, quotient contour equality, selected improper integral, countable choice, or ambient completeness, the six obligations exhaust the local NameCert surface. □

Theorem 7969.3 (BromwichInversionWindow Real seal non-escape)**.** *Every Real-facing consumer of an accepted* $\mathsf{BromwichInversionWindow}^{\uparrow}$ *packet must read the displayed route*

$$\mathsf{LaplaceTransform}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{ContourIntegralSocket}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the finite rows L, Q, S, R *before any windowed inversion value is public. The packet returns only the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *for the displayed finite window; it does not certify a host Bromwich inversion theorem, analytic continuation of the transform, a selected improper integral, quotient contour equality, countable choice, or ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier. The only Real-facing coordinate is R, and it is listed after the transform row L, the height-window readback row Q, and the contour socket row S.

Apply the obligation surface from Definition 7969.2. Transform-source compatibility, regular readback exactness, and contour-socket handoff force a consumer to expose L, Q, S before R can be read.

The absence clauses in the carrier then give non-escape. There is no coordinate for an inverse-transform theorem, analytic-continuation theorem, selected improper integral, quotient contour equality, countable-choice schedule, or ambient Real completeness, so no such object can be returned as part of the window seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BromwichInversionWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{BromwichInversionWindow}^{\uparrow}$ packet is generated from a Laplace-transform row, a vertical-line abscissa row, a finite height window, a RegSeqRat quadrature readback row, a ContourIntegralSocket row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BromwichInversionWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7969.1, Definition 7969.2, and Definition 7969.3 seed the finite vertical-line inversion window over LaplaceTransform, ContourIntegralSocket, RegSeqRat, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BromwichInversionWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Bromwich inversion theorem, analytic continuation of a transform, selected improper integrals, quotient contour equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transform-source compatibility, vertical-line stability, regular readback exactness, contour-socket handoff, Real seal non-escape, provenance, and local naming for the Bromwich inversion window packet.

Chapter 7970

A Concrete Naming Certificate for $\mathsf{AbelPlana}^{\uparrow}$

Following Abel and Plana's summation formula, the $\mathsf{AbelPlana}^{\uparrow}$ packet records a finite boundary between a discrete sum window, an integral-window readback, and the vertical correction terms needed by constructive analytic consumers. It is grounded beside the Euler–Maclaurin boundary of chapter 2059, the Fourier boundary of chapter 3951, the summation-by-parts surface of chapter 1763, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The certificate names only the finite BEDC route by which a boundary correction is inspected before a $\mathsf{Real}^{\uparrow}$ read; it does not assert a meromorphic continuation theorem, a contour integral value, a selected infinite sum, countable choice, trichotomy, host complex equality, or ambient completeness.

Definition 7970.1 (Abel-Plana carrier)**.** *An* $\mathsf{AbelPlana}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, I, V, B, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite discrete summand-window row, recording the inspected values, endpoint indices, and partial-sum cells.*
2. *I is the finite integral-window row, recording the interval cells and rational enclosure data used by the continuous side of the comparison.*
3. *V is the vertical-boundary row, storing the two inspected correction corridors and their displayed orientation data.*
4. *B is the boundary-kernel row, recording the denominator and sign windows used to read the Abel-Plana correction without selecting an ambient contour value.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that budgets the summand, integral, and boundary-kernel inspections.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed tolerance and boundary data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the discrete, integral, vertical-boundary, dyadic, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for all displayed data and the structural rows.*

9. *C records displayed* Cont *replay from a consumer request through S, I, V, B, D, R, E and then to the provenance and naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{AbelPlana}^{\uparrow}$, $\mathsf{EulerMaclaurin}^{\uparrow}$, $\mathsf{PoissonSummation}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelPlana}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host contour integral, a meromorphic continuation principle, an infinite choice of residues, a selected completed sum, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7970.2 (Abel-Plana NameCert obligations)**.** *For every accepted* $\mathsf{AbelPlana}^{\uparrow}$ *carrier $A = (S, I, V, B, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelPlana}^{\uparrow}}$ *surface consists exactly of carrier admission, discrete summand-window exposure, integral-window exposure, vertical-boundary exposure, boundary-kernel compatibility, dyadic tolerance budgeting, regular rational readback, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The rows S, I, V, B expose the finite discrete, integral, vertical-boundary, and boundary-kernel data that make the Abel-Plana comparison readable.

Read the tolerance and readback layer next. The dyadic row D budgets the three analytic faces, and the regular rational row R supplies the only completion-facing readback available before the seal.

Finally use the structural rows. The seal row E is downstream of S, I, V, B, D, R, while H, C, P, N only transport, replay, source, and name the same displayed route. Since no row carries a host contour value, continuation theorem, selected residue family, completed sum, quotient equality, or ambient completeness principle, the listed obligations exhaust the local certificate. □

Theorem 7970.3 (Abel-Plana Real-seal nonescape)**.** *For every accepted* $\mathsf{AbelPlana}^{\uparrow}$ *carrier $A = (S, I, V, B, D, R, E, H, C, P, N)$, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read of E factors through the finite route*

$$S, I, V, B, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the packet exports the displayed summation-boundary correction only after the discrete window, integral window, vertical-boundary corridor, boundary kernel, dyadic tolerance row, and regular rational readback have been inspected.

Proof. Start from the NameCert obligations of Definition 7970.2. They force every accepted read to expose the discrete summand row, integral row, vertical-boundary row, boundary-kernel row, dyadic tolerance row, and regular rational readback before the seal row can be consumed.

The rows H, C, P, N preserve that route componentwise. They do not add an alternate analytic coordinate, so a seal-facing consumer receives the same displayed packet and no detached contour object.

The carrier omits meromorphic continuation, selected residues, countable choice, trichotomy, host equality, completed infinite sums, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Therefore the displayed route is the whole Real-seal boundary. □

Falsifiable prediction: An accepted $\mathsf{AbelPlana}^{\uparrow}$ packet that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the discrete summand window, integral window, vertical-boundary corridor, boundary kernel, dyadic tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: EulerMaclaurin, PoissonSummation, AbelSummation, DyadicRatCore, RegSeqRat, Real: no listed sibling supplies the full eleven-row packet $A = (S, I, V, B, D, R, E, H, C, P, N)$, because each sibling contributes only a summation boundary, Fourier boundary, summation-by-parts route, dyadic tolerance layer, regular readback, or seal face while this chapter binds all six faces into one Abel-Plana boundary certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelPlana}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelPlana}^{\uparrow}$ packet is generated from finite discrete summand windows, integral-window data, vertical-boundary corridors, boundary-kernel rows, DyadicRatCoreUp tolerance budgets, RegSeqRatUp readbacks, a RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelPlana}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7970.1, Definition 7970.2, and Definition 7970.3 seed the finite Abel-Plana summation-boundary route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelPlana}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim meromorphic continuation, contour integral values, selected residues, selected sums, countable choice, trichotomy, host equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite NameCert obligation surface and Real-seal nonescape theorem.

Chapter 7971

A Concrete Naming Certificate for $\mathsf{GreenTheorem}^{\uparrow}$

Following Green's theorem as the plane Stokes specialization, and the Cartan–de Rham reading of boundary integration, the $\mathsf{GreenTheorem}^{\uparrow}$ packet records the finite BEDC route by which an oriented planar region, its boundary curve, a differential-form row, a scalar curl or divergence row, and a Riemann integral handoff are read as one plane-boundary certificate. It consumes the Stokes theorem packet of chapter 5720, the differential-form surface of chapter 136, and the tagged integration boundary of chapter 1526. The packet names only this finite plane route; it does not provide a host vector-calculus theorem, selected parametrizations, a measure-completion carrier, quotient equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7971.1 (Green theorem carrier)**.** *A* $\mathsf{GreenTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, B, F, S, T, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *R is the plane-region row, including the displayed chart window, orientation, and finite boundary incidence data.*
- *B is the oriented boundary row for the same region, read as the plane curve boundary consumed by the Stokes packet.*
- *F is the vector-field or differential-form row, routed through* $\mathsf{DiffForm}^{\uparrow}$ *and carrying the two scalar component reads needed for the planar expression.*
- *S is the scalar curl or divergence row obtained from the same displayed form data, with no separate host vector-field equality.*
- *T is the* $\mathsf{StokesTheorem}^{\uparrow}$ *specialization row that reads the planar boundary as the two-dimensional Stokes boundary route.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *handoff for the displayed line and region integral reads over the same rows R, B, F, S, T.*
- *H records componentwise* hsame *transport for the region, boundary, form, scalar, Stokes, and integration rows.*
- *C records displayed* Cont *replay from the plane-region source through the boundary and differential-form rows to the integration handoff.*

- *P is the* Pkg *provenance over* $\mathsf{GreenTheorem}^{\uparrow}$, $\mathsf{StokesTheorem}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*
- *N is the local* $\mathsf{NameCert}_{\mathsf{GreenTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host smooth-domain theorem, a chosen boundary parametrization, measure completion, quotient equality of vector fields, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7971.2 (Green theorem NameCert obligations)**.** *For every accepted* $\mathsf{GreenTheorem}^{\uparrow}$ *carrier $G = (R, B, F, S, T, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GreenTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for the plane region, oriented boundary, differential-form or vector-field row, scalar curl or divergence row, Stokes specialization, Riemann integral handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. The obligations are discharged only over the finite route*

$$R, B, F, S, T, I, H, C, P, N.$$

The certificate exports no selected parametrization, host Green theorem, measure-space integral, quotient function equality, or ambient real-completeness principle.

Proof. Project the carrier of Definition 7971.1. The analytic rows are exactly the plane-region row R, boundary row B, differential-form or vector-field row F, scalar curl or divergence row S, Stokes specialization row T, and Riemann handoff row I.

The boundary compatibility is local. The row B is read over the same oriented plane region R, while T routes that boundary through the Stokes packet of chapter 5720. The scalar row S is read from the displayed form data F, so the packet does not compare hidden host vector fields.

Finally attach H, C, P, N. These structural rows transport, replay, source, and name the same finite route. Since none of the carrier coordinates contains a selected parametrization, host theorem, measure-completion carrier, quotient equality, or ambient completeness assertion, the listed rows exhaust the NameCert obligations. □

Theorem 7971.3 (Green theorem Stokes plane boundary route)**.** *Every* $\mathsf{GreenTheorem}^{\uparrow}$ *consumer of a plane boundary integration statement must first read the finite route*

$$\mathsf{DiffForm}^{\uparrow},\ \mathsf{StokesTheorem}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

through the rows F, T, I, H, C, P, N of Definition 7971.1, after the plane-region row R and oriented-boundary row B have been displayed. The Green packet is therefore the planar specialization of the Stokes boundary route, with the scalar curl or divergence row S tied to the same differential-form data before the Riemann handoff is read. It is not an independent host vector-calculus theorem, a measure-completion route, a quotient equality of vector fields, or a selected boundary parametrization.

Proof. Begin with the carrier inventory. The rows R and B expose the region and its oriented boundary before any analytic read is public.

The differential-form and scalar rows are then read through F and S. The form row is the $\mathsf{DiffForm}^{\uparrow}$ input of chapter 136, while the scalar curl or divergence row is a displayed read from that same plane data rather than a second host object.

Next consume the Stokes specialization row T and the Riemann handoff row I. The Stokes packet of chapter 5720 supplies the boundary comparison, and the Riemann packet of chapter 1526 supplies the finite tagged-integration read. The structural rows H, C, P, N preserve this route without adding any of the excluded host principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{GreenTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{GreenTheorem}^{\uparrow}$ packet is generated from a plane-region row, oriented-boundary row, differential-form or vector-field row, scalar curl or divergence row, Stokes-specialization row, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GreenTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7971.1, Definition 7971.2, and Definition 7971.3 bind the finite plane-boundary route through $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{StokesTheorem}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GreenTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a host vector-calculus theorem, selected boundary parametrizations, quotient equality of vector fields, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, scalar-row exactness, Stokes specialization, Riemann handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.GreenTheoremUp.

Chapter 7972

A Concrete Naming Certificate for $\mathsf{CauchyLimitSeparation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyLimitSeparation}^{\uparrow}$ packet records the finite separation data needed when two regular Cauchy presentations are compared at the real-name boundary. It sits after the dyadic arithmetic ledger of chapter 291, the finite stream windows of chapter 126, the regular rational carrier of chapter 301, the real-name seal of chapter 38, and the uniqueness route of chapter 3942. The packet names limit separation as finite BEDC data; it does not assert quotient real equality, trichotomy, located apartness, selected limits, or ambient completeness.

Definition 7972.1 (CauchyLimitSeparation carrier)**.** *A* $\mathsf{CauchyLimitSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the two inspected Cauchy windows.*
2. W *is the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the two presentations.*
3. R *is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *boundary row reached only after the dyadic and regular-rational rows are displayed.*
5. U *is the* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *handoff row that records uniqueness-facing compatibility without quotienting the two presentations.*
6. H *stores componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, real boundary, uniqueness handoff, replay, provenance, and local naming rows.*
7. C *stores displayed* Cont *replay of the separation query through* D, W, R, E, U, H.
8. P *is the* Pkg *provenance over* $\mathsf{CauchyLimitSeparation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeparation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. No coordinate stores quotient real equality, trichotomy, a selected limiting point, ambient completeness of $\mathsf{Real}^{\uparrow}$, *located apartness of arbitrary reals, or host sequence extensionality.*

Definition 7972.2 (CauchyLimitSeparation classifier)**.** *The* $\mathsf{CauchyLimitSeparation}^{\uparrow}$ *classifier accepts a packet* $S = (D, W, R, E, U, H, C, P, N)$ *exactly when the dyadic tolerance ledger* D*, paired stream windows* W*, regular rational readback* R*, real boundary* E*, and uniqueness handoff* U *are all displayed before transport, replay, provenance, and local naming are read. The classifier is therefore finite-row separation data over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$*; it is not a quotient of real names or an ambient separation theorem.*

Theorem 7972.3 (CauchyLimitSeparation NameCert)**.** *For every accepted* $\mathsf{CauchyLimitSeparation}^{\uparrow}$ *carrier* $S = (D, W, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeparation}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability for the displayed dyadic and stream windows, regular rational readback, real-boundary routing, uniqueness handoff compatibility, componentwise transport, displayed replay, provenance, and local naming. The surface refuses quotient real equality, trichotomy, located apartness for arbitrary real names, selected limits, host sequence extensionality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 7972.1. The mathematical rows are D, W, R, E, U, and the structural rows are H, C, P, N.

Use the classifier of Definition 7972.2. It requires the dyadic tolerance ledger and paired stream windows before the regular rational readback, then routes that readback to the real boundary and uniqueness handoff.

Transport, replay, provenance, and naming preserve that same finite route. Since no carrier coordinate stores quotient equality, trichotomy, arbitrary located apartness, selected limits, host sequence extensionality, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7972.4 (CauchyLimitSeparation Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *uniqueness or separation consumer of an accepted* $\mathsf{CauchyLimitSeparation}^{\uparrow}$ *packet must read the dyadic tolerance ledger* D*, paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, real boundary row* E*, and* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *handoff row* U *before any conclusion about limit compatibility is consumed. The handoff supplies a finite Real-facing separation seal; it does not supply quotient real equality, selected limits, trichotomy, located apartness for arbitrary real names, or ambient completeness.*

Proof. Start with Definition 7972.3. The local NameCert surface forces every accepted read to expose $D, W, R, E, U, H, C, P, N$.

The real-facing consumer then reads E only after the dyadic tolerance ledger, stream windows, and regular rational readback have been displayed. The uniqueness-facing consumer reads U as compatibility with chapter 3942, not as quotient equality of real names.

Finally inspect the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite separation packet. Since the excluded global claims have no coordinate in S, the Real handoff remains a finite seal over the displayed rows. □

Falsifiable prediction: An accepted $\mathsf{CauchyLimitSeparation}^{\uparrow}$ read is refuted if the real boundary row E or uniqueness handoff U is consumed while the dyadic tolerance ledger D, paired stream windows W, or regular rational readback R is absent.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-limit-uniqueness: no listed sibling supplies the joint dyadic ledger, paired stream windows, regular rational readback, real boundary, uniqueness handoff, transport, replay, provenance, and local NameCert rows while refusing quotient real equality and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitSeparation}\uparrow}$.

Constructive story: A $\mathsf{CauchyLimitSeparation}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, paired StreamName windows, RegSeqRat readback, a RealUp boundary row, a RegularCauchyLimitUniqueness handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitSeparation}\uparrow}, \mathsf{seed})$

Scope closed: Definition 7972.1, Definition 7972.2, Definition 7972.3, and Definition 7972.4 seed the finite limit-separation packet consumed by Real-facing uniqueness arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitSeparation}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, located apartness for arbitrary real names, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance routing, stream-window readback, regular rational handoff, Real boundary routing, uniqueness compatibility, transport, replay, provenance, and local naming for BEDC.Derived.CauchyLimitSeparationUp.

Chapter 7973

A Concrete Naming Certificate for $\mathsf{KyFanMinimax}^{\uparrow}$

Following Ky Fan and the KKM minimax tradition, the $\mathsf{KyFanMinimax}^{\uparrow}$ packet records a finite convex-compact minimax route as NameCert data rather than as a host separation or selection principle. It consumes the finite KKM cover surface of chapter 6671, the convex carrier of chapter 208, the compact metric support of chapter 129, and the terminal scalar readback of chapter 38. The chapter separates this convex KKM route from the finite payoff-table certificate of chapter 7060 and the spectral min-max route of chapter 7431.

Definition 7973.1 (KyFanMinimax carrier)**.** *A* $\mathsf{KyFanMinimax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, K, A, L, U, V, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row for the compact convex source surface and its finite observation windows.*
2. *Y is the* $\mathsf{ConvexSet}^{\uparrow}$ *row recording the convex carrier and the finite affine-spine data used by the minimax read.*
3. *K is the* $\mathsf{KKMFiniteCover}^{\uparrow}$ *row carrying the finite face-cover witness used to route the minimax inequality.*
4. *A is the admissible-bifunction row, recording the displayed two-variable real-valued comparison surface and its finite domain membership checks.*
5. *L is the lower-level-set convexity row, exposing the finite convex slices read by the KKM cover route.*
6. *U is the upper semicontinuity and compact-window row, tying each displayed scalar threshold to compact finite observations.*
7. *V is the minimax-value row, read as a* $\mathsf{Real}^{\uparrow}$ *seal only after the KKM, convexity, and compact-window rows have been displayed.*
8. *W is the witness-budget row, recording the finite threshold, cover, and comparison windows retained by downstream consumers.*
9. *H stores componentwise* hsame *transport for X, Y, K, A, L, U, V, W and the local certificate row.*

10. *C stores displayed* Cont *replay for compact support, convex membership, KKM cover reads, bifunction comparisons, scalar threshold reads, and value-seal export.*
11. *P is the* Pkg *provenance over* $\mathsf{KyFanMinimax}^{\uparrow}$, $\mathsf{KKMFiniteCover}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KyFanMinimax}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host minimax selector, Hahn–Banach separation, arbitrary open-cover compactness, quotient equality of level sets, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7973.2 (KyFanMinimax NameCert obligations)**.** *For every accepted* $\mathsf{KyFanMinimax}^{\uparrow}$ *carrier $F = (X, Y, K, A, L, U, V, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KyFanMinimax}^{\uparrow}}$ *surface consists exactly of compact source admission, convex carrier admission, finite KKM cover admission, bifunction comparison admission, lower-level convexity, compact-window upper-semicontinuity, value-seal readback, witness-budget exposure, componentwise transport, displayed replay, provenance, and local naming. A minimax read is admitted only after the compact, convex, KKM, bifunction, level-set, threshold-window, value, and witness rows are visible in the same finite packet.*

Proof. Project the carrier of Definition 7973.1. The compact and convex support rows are X and Y, while K supplies the finite KKM cover surface.

Read the bifunction and level-set rows next. The row A exposes only displayed comparison data, L keeps the lower-level slices inside the convex carrier, and U ties scalar thresholds to compact finite windows.

The value and witness rows are downstream. The $\mathsf{Real}^{\uparrow}$ seal V is read only after K, L, U have supplied the finite KKM route, and W records the finite cover and threshold windows consumed by later readers.

Finally H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the carrier has no row for host selection, separation, arbitrary compactness, quotient level-set equality, countable choice, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7973.3 (KyFanMinimax finite KKM route)**.** *Every accepted* $\mathsf{KyFanMinimax}^{\uparrow}$ *minimax-value read factors through the finite route*

$$\mathsf{KKMFiniteCover}^{\uparrow}, \quad \mathsf{ConvexSet}^{\uparrow}, \quad \mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer must expose $X, Y, K, A, L, U, V, W, H, C, P, N$ before the value seal is public. The route exports no finite-game payoff-table saddle theorem, no spectral min-max theorem, no host minimax selector, no unrestricted compactness theorem, and no quotient equality of convex level sets.

Proof. Start with the accepted carrier. The only cover-facing row is K, and it is the finite KKM cover row tied to the same compact and convex support rows X, Y.

The level-set and compact-window rows L, U are the bridge from the bifunction comparison row A to the KKM cover. Thus a minimax consumer reads convex slices and compact finite windows before any scalar value seal is available.

The terminal scalar read is V, and W records the finite threshold and cover budget that reaches it. Transport, replay, provenance, and naming through H, C, P, N preserve this route without adding a payoff-table, spectral, or host selection coordinate.

Therefore every public value read is a finite KKM route through the named sibling surfaces, not a finite-game minimax theorem, Courant–Fischer theorem, host minimax selector, unrestricted compactness theorem, or quotient level-set equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{KyFanMinimax}^{\uparrow}}$.

Constructive story: A $\mathsf{KyFanMinimax}^{\uparrow}$ packet is generated from CompactMetric support rows, ConvexSet carrier rows, a KKMFiniteCover row, bifunction comparison rows, lower-level convexity rows, compact-window threshold rows, a RealUp value seal, witness-budget rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KyFanMinimax}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7973.1, Definition 7973.2, and Definition 7973.3 seed the finite Ky Fan minimax route over displayed compact, convex, KKM, bifunction, level-set, threshold, value, witness, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KyFanMinimax}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host minimax selection, Hahn–Banach separation, arbitrary open-cover compactness, finite-game minimax, spectral min-max, quotient equality of level sets, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, KKM cover admission, convex level-set stability, compact-window stability, value-seal readback, witness-budget exposure, transport stability, replay stability, provenance, and local naming for BEDC.Derived.KyFanMinimaxUp.

Chapter 7974

A Concrete Naming Certificate for $\mathsf{LawlikeStreamBoundary}^{\uparrow}$

Following Kleene, Troelstra–van Dalen, and Bishop–Bridges on lawlike sequences and constructive real names, the $\mathsf{LawlikeStreamBoundary}^{\uparrow}$ packet records the finite boundary at which a certified lawlike source may be read as a countable stream name without selecting a host sequence. It sits over the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the constructive warning surface of chapter 1406, and the refusal boundary of chapter 773. The packet names a lawlike stream boundary for Real and Completion consumers; it is not countable choice, not quotient stream equality, not trichotomy, not a selected host sequence, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7974.1 (Lawlike stream boundary carrier)**.** *A* $\mathsf{LawlikeStreamBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the lawlike source row. It records the displayed rule, finite source ledger, and admissible request boundary for the stream name being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It lists exactly the windows through which the lawlike source may be queried before any rational or real-facing read is public.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and the rational tolerance discipline already exposed by the stream route.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is reached only after the lawlike source row, StreamName window row, and RegSeqRat readback row have been displayed.*
5. *O is the obstruction and refusal row. It records that the packet refuses selected host sequences, countable-choice selectors, quotient stream equality, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness, matching the constructive warning surfaces of* $\mathsf{SpeckerSequence}^{\uparrow}$ *and* $\mathsf{BrouwerianCounterexample}^{\uparrow}$.
6. *H records componentwise* hsame *transport for the lawlike source, StreamName windows, RegSeqRat readback, Real seal, obstruction row, continuation replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from lawlike source inspection through finite windows, regular rational readback, Real sealing, and refusal checking.*

8. *P is the* Pkg *provenance over* LawlikeStreamBoundary$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, SpeckerSequence$^\uparrow$, BrouwerianCounterexample$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{LawlikeStreamBoundary}^\uparrow}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $S, W, R, E, O, H, C, P, N$ and has no coordinate for a selected host sequence, a global choice function, quotient stream equality, trichotomy, or a completeness theorem for Real$^\uparrow$.

Theorem 7974.2 (Lawlike stream boundary NameCert obligations)**.** *For an accepted* LawlikeStreamBoundary$^\uparrow$ *carrier $L = (S, W, R, E, O, H, C, P, N)$, the local* NameCert$_{\text{LawlikeStreamBoundary}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lawlike source row,* StreamName$^\uparrow$ *finite-window row,* RegSeqRat$^\uparrow$ *readback row, terminal* Real$^\uparrow$ *seal, obstruction and refusal row, transport, replay, provenance, and local naming row;*

2. *source-window compatibility, requiring every public request to start at the displayed lawlike source row S and then pass through the finite window row W;*

3. *regular readback discipline, requiring rational observations to pass through the* RegSeqRat$^\uparrow$ *row R before the* Real$^\uparrow$ *seal E is exposed;*

4. *refusal stability, requiring the obstruction row O to remain visible whenever a consumer asks for host sequence selection, countable choice, quotient stream equality, trichotomy, or Real completeness;*

5. *transport and replay, requiring H and C to preserve the same ordered route under* hsame *transport and displayed* Cont *replay;*

6. *provenance and naming, requiring P and N to name only the finite lawlike-window surface and not an ambient stream object.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. Its public data are exactly the rows $S, W, R, E, O, H, C, P, N$.

The source-window obligation follows from the first two rows. A consumer begins at the lawlike source row S and then reads only the finite windows displayed in W before asking for rational or Real-facing data.

The regular readback obligation is the row R. It binds the finite StreamName observations to the regular rational readback surface before the terminal seal E can be consumed.

The obstruction row O is part of the same packet, so a request for selected host sequences, countable choice, quotient stream equality, trichotomy, or Real completeness is refused at the boundary rather than supplied by another coordinate.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no additional coordinate appears in the carrier, the local NameCert surface is exactly the displayed lawlike-window boundary. □

Theorem 7974.3 (Lawlike stream boundary Real nonescape)**.** *For an accepted* LawlikeStreamBoundary$^\uparrow$ *carrier*

$$L = (S, W, R, E, O, H, C, P, N),$$

every Real-facing read factors through the finite route consisting of the lawlike source row S, StreamName$^\uparrow$ *window row W,* RegSeqRat$^\uparrow$ *readback row R, terminal* Real$^\uparrow$ *seal E, obstruction row O,* hsame *transport row H,* Cont *replay row C,* Pkg *provenance row P, and local* NameCert$_{\text{LawlikeStreamBoundary}^\uparrow}$

row N. The route exports no selected host sequence, no countable-choice selector, no quotient stream equality, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and project its displayed rows. The Real-facing endpoint is E, and the carrier places E after the lawlike source, finite StreamName windows, and regular rational readback.

The row S supplies only the lawlike source ledger. The row W then restricts inspection to finite windows, and R supplies the regular rational readback attached to those windows.

The Real seal E is terminal for this route. If a consumer asks for a selected host sequence, countable choice, quotient stream equality, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$, the request has to pass through the obstruction row O, which records precisely the refusal.

Componentwise transport through H and replay through C preserve the same route, while P and N supply provenance and local naming. Hence every Real-facing read remains inside the finite lawlike-window boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeStreamBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawlikeStreamBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7974.1, Definition 7974.2, and Definition 7974.3 seed the finite lawlike-source, StreamName window, RegSeqRat readback, Real seal, obstruction, transport, replay, provenance, and local NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawlikeStreamBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected host sequence, countable choice, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window compatibility, regular readback discipline, refusal stability, transport replay, provenance, and non-escape for BEDC.Derived.LawlikeStreamBoundaryUp.

Constructive story: A $\mathsf{LawlikeStreamBoundary}^{\uparrow}$ packet is generated from a lawlike source ledger, finite StreamName windows, RegSeqRat readback, a Real seal, an obstruction row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7975

A Concrete Naming Certificate for $\mathsf{BishopRealNumber}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealNumber}^{\uparrow}$ packet records the ordinary real-number interface as a finite BEDC NameCert surface rather than as a host quotient or completed object. It is a sibling of the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the located Cauchy modulus route of chapter 1035. The chapter names a traditional constructive real-number carrier for Real and Completion consumers without asserting trichotomy, quotient stream equality, Dedekind-cut completeness, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7975.1 (BishopRealNumber carrier)**.** *A* $\mathsf{BishopRealNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule that opens the inspected approximation windows;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback assembled from those windows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing the displayed rational errors;*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback and dyadic ledger have been exposed;*
5. *M is the* $\mathsf{LocatedCauchyModulus}^{\uparrow}$ *row recording located tail control for the same readback route;*
6. *H records componentwise* hsame *transport for the schedule, readback, tolerance ledger, seal, modulus, and structural rows;*
7. *C records the displayed* Cont *replay for window inspection, rational readback, tolerance comparison, Real seal delivery, located modulus use, and non-escape inspection;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealNumber}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $S, R, D, E, M, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopRealNumberUp.BishopRealNumberTasteGate single carrier alignment`

Theorem 7975.2 (BishopRealNumber NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealNumber}^{\uparrow}$ *carrier* $B = (S, R, D, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealNumber}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, located Cauchy modulus row, componentwise transport, displayed replay, provenance, and local naming row. Any Real or Completion consumer of the packet must read those rows before treating the carrier as a constructive real-number name.*

Proof. Project the carrier of Definition 7975.1. The analytic rows are the displayed schedule S, regular readback R, dyadic ledger D, terminal seal E, and located modulus M.

The structural rows H, C, P, N transport, replay, package, and name those same analytic rows. Since no coordinate stores quotient stream equality, a selected host limit, trichotomy, Dedekind-cut completeness, countable choice, open-cover compactness, or ambient RealUp completeness, the NameCert obligation surface is exactly the displayed finite packet. □

Theorem 7975.3 (BishopRealNumber RegSeqRat seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{BishopRealNumber}^{\uparrow}$ *carrier factors first through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*. The located Cauchy modulus row* M *records tail control for that same route, while* H, C, P, N *preserve it by transport, replay, provenance, and naming.*

Proof. Read the accepted carrier $B = (S, R, D, E, M, H, C, P, N)$. The only rows before the Real-facing endpoint are the finite schedule, regular rational readback, dyadic tolerance ledger, and located modulus.

The seal row E is displayed after those rows in the carrier. Therefore a consumer reaches $\mathsf{Real}^{\uparrow}$ only through the route supplied by S, R, D, M, with R providing the regular rational handoff and D providing the dyadic error ledger.

The structural rows do not add another Real seal route. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the same finite readback route, so the packet supplies a RegSeqRat-facing constructive real-number seal and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealNumber}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealNumber}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, located Cauchy modulus row, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealNumber}^{\uparrow}}$, seed)

Scope closed: Definition 7975.1, Definition 7975.2, and Definition 7975.3 bind the finite constructive real-number carrier and its RegSeqRat-to-Real seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealNumber}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close trichotomy, quotient stream equality, Dedekind-cut completeness, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Formal closure requires checked carrier admission, StreamName scheduling, RegSeqRat readback, dyadic tolerance compatibility, RealUp seal delivery, located modulus stability, transport stability, route replay, provenance, and non-escape for the same finite packet.

Chapter 7976

A Concrete Naming Certificate for $\mathsf{RieszProjection}^{\uparrow}$

Following Dunford–Schwartz and standard Banach-algebra spectral theory, the $\mathsf{RieszProjection}^{\uparrow}$ packet records the finite contour-resolvent surface by which an isolated spectral window is read as a projection before any ambient functional calculus, host spectral measure, or quotient operator equality is available. It sits after the resolvent identity route of chapter 3715, the spectral-radius separator of chapter 2352, and the bounded-operator carrier of chapter 1644.

Definition 7976.1 (RieszProjection carrier)**.** *A* $\mathsf{RieszProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, T, G, C, R, S, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row carrying the complete normed algebra surface in which the operator and resolvent products are read.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for the displayed operator whose isolated spectral window is inspected.*
3. *G is the contour-window row. It stores the finite contour samples, orientation ledger, and inside/outside window labels used by the projection read.*
4. *C is the shifted-resolvent graph. For each contour sample it records the displayed shifted operator and the inverse-check row supplied to the resolvent route.*
5. *R is the* $\mathsf{ResolventIdentity}^{\uparrow}$ *row, exposing the finite identity checks that keep neighbouring shifted-resolvent reads on one contour route.*
6. *S is the* $\mathsf{SpectralRadius}^{\uparrow}$ *separator row. It records the radius and gap evidence that keeps the contour window separated from the refused spectral side.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *tolerance seal for contour mesh, separator gap, and norm-bound readback.*
8. *H records componentwise* hsame *transport for A, T, G, C, R, S, E and the structural rows.*
9. *K records displayed* Cont *replay of the route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{ResolventIdentity}^{\uparrow}, \mathsf{SpectralRadius}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RieszProjection}^{\uparrow}.$$

10. *P is the* Pkg *provenance over* $\mathsf{RieszProjection}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszProjection}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, T, G, C, R, S, E, H, K, P, N$ and has no coordinate for a host spectral projection, a selected contour integral value, spectral-measure additivity, quotient operator equality, maximal ideal space, countable choice, or an ambient holomorphic functional calculus.

Theorem 7976.2 (RieszProjection NameCert obligations)**.** *For every accepted* $\mathsf{RieszProjection}^{\uparrow}$ *carrier*

$$Q = (A, T, G, C, R, S, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszProjection}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Banach-algebra source, bounded-operator row, contour window, shifted-resolvent graph, resolvent-identity row, spectral-radius separator, Real tolerance seal, transport, replay, provenance, and local naming row;*

2. *contour-window stability, requiring every projection read to use only the finite samples and orientation ledger stored in G;*

3. *shifted-resolvent locality, requiring each sample read to pass through C and then the displayed* $\mathsf{ResolventIdentity}^{\uparrow}$ *row R before it is used by the projection route;*

4. *separator discipline, requiring the gap and refusal side to be read through the* $\mathsf{SpectralRadius}^{\uparrow}$ *row S and the* $\mathsf{Real}^{\uparrow}$ *tolerance seal E;*

5. *classifier transport, requiring H and K to preserve the contour, resolvent, separator, tolerance, provenance, and local name componentwise;*

6. *consumer non-escape, excluding host spectral projections, selected contour integrals, spectral-measure algebra, quotient operator equality, maximal ideal spaces, choice schedules, and ambient functional calculus from the packet.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 7976.1. The accepted packet exposes exactly the Banach-algebra source, bounded-operator row, contour window, shifted-resolvent graph, resolvent-identity row, spectral-radius separator, Real tolerance seal, transport, replay, provenance, and local naming row.

Contour stability is finite. The row G names the contour samples and orientation ledger, so a projection read cannot enlarge the contour or replace it by an unbounded functional-calculus object.

The shifted-resolvent graph C and the resolvent identity row R then bind the local inverse checks. Every sample read reaches the projection route only through the same displayed resolvent-contour data.

The separator row S and the Real seal E record the finite gap and tolerance evidence. Since H, K, P, N only transport, replay, source, and name those rows, none of the excluded host-side spectral objects can appear as a local NameCert obligation. □

Theorem 7976.3 (RieszProjection resolvent-contour route)**.** *For every accepted* $\mathsf{RieszProjection}^{\uparrow}$ *carrier* $Q = (A, T, G, C, R, S, E, H, K, P, N)$*, each projection-facing read factors through the displayed route*

$$\mathsf{BoundedLinearOperator}^{\uparrow},\ \mathsf{ResolventIdentity}^{\uparrow},\ \mathsf{SpectralRadius}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The read first fixes the operator row T*, then the finite contour window* G*, then the shifted-resolvent graph* C*, then the identity row* R*, then the separator row* S *and tolerance seal* E*, before* H, K, P, N *export the local name. It does not pass through a host spectral projection, selected contour integral, quotient operator equality, or ambient holomorphic functional calculus.*

Proof. Project the accepted packet. The only operator-facing coordinate is T, and the only contour-facing coordinate is G. Thus a projection read begins with the bounded-operator row and the finite contour ledger.

The next coordinates are C and R. These rows expose shifted-resolvent checks and the sibling resolvent identity route; no projection value is public before those rows are replayed.

The separator and tolerance rows S, E then guard the spectral side of the contour. Finally H, K, P, N transport, replay, source, and name the same finite route, so every projection-facing read factors through the displayed resolvent-contour surface. □

Falsifiable prediction: Every accepted $\mathsf{RieszProjection}^{\uparrow}$ consumer exposes the Banach-algebra source, bounded-operator row, finite contour window, shifted-resolvent graph, resolvent identity row, spectral-radius separator, Real tolerance seal, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before reading a projection-facing boundary.

Independence witness: resolvent-identity, spectral-radius, boundedlinearoperator, banachalgebra, real: ResolventIdentityUp supplies shifted inverse identity rows without an isolated contour window; SpectralRadiusUp supplies radius and gap evidence without a projection carrier; BoundedLinearOperatorUp supplies operator action and norm-bound rows without contour-resolvent data; BanachAlgebraUp supplies the product surface without the isolated-window separator; RealUp seals tolerances without naming the projection route.

Closure status for $\mathsf{NameCert}_{\mathsf{RieszProjection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RieszProjection}^{\uparrow}$ packet is generated from a BanachAlgebraUp source, a BoundedLinearOperatorUp row, a finite contour window, shifted-resolvent graph data, a ResolventIdentityUp row, a SpectralRadiusUp separator, a RealUp tolerance seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7976.1, Definition 7976.2, and Definition 7976.3 seed the finite resolvent-contour projection route over BanachAlgebraUp, BoundedLinearOperatorUp, ResolventIdentityUp, SpectralRadiusUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host spectral projections, selected contour integrals, spectral-measure algebra, quotient operator equality, maximal ideal spaces, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contour-window stability, shifted-resolvent locality, separator discipline, classifier transport, and consumer non-escape for BEDC.Derived.RieszProjectionUp.

Chapter 7977

A Concrete Naming Certificate for MetaCICResidualCommutationWitness$^{\uparrow}$

The MetaCICResidualCommutationWitness$^{\uparrow}$ packet realizes the residual-substitution boundary isolated by chapter 9378 as finite NameCert data. It records the local residual window, substitution-site ledger, candidate join row, and conditional confluence handoff that a later MetaCIC proof must consume before a residual beta branch can be read as commuting with substitution.

Theorem 7977.1 (MetaCIC residual-commutation witness sibling route)**.** *Every* MetaCICResidualCommutationWit *consumer that reads a candidate join as confluence-facing evidence must first pass through the residual-substitution boundary of chapter 2181 and the residual-substitution compatibility surface of chapter 4370. The witness records local residual windows and substitution-site ledgers, while those siblings provide the premise and compatibility surfaces that the witness cannot discharge by itself.*

Proof. Project the witness carrier. Its rows record a MetaCIC source window, a residual window, a substitution-site ledger, a candidate join row, and a conditional handoff, but no row supplies the residual-substitution premise.

Read the residual-substitution boundary sibling first. It names the boundary that must be present before a local residual window can be treated as a substitution-sensitive branch rather than an isolated candidate.

Then read the compatibility sibling. That surface supplies the comparison discipline needed before the candidate join can face the confluence audit map. The present witness therefore remains a local ledger until those sibling rows are displayed, and it cannot export premise-free confluence evidence. □

Definition 7977.2 (MetaCIC residual-commutation witness carrier)**.** *A* MetaCICResidualCommutationWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, R, S, J, D, T, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC source row, naming the term constructor window, closure side condition, beta branch, and parallel-step boundary under inspection.*
2. *R is the residual window. It records the finite residual redex positions that remain visible after the displayed beta branch is taken.*

3. *S is the substitution-site ledger, listing the binder depth, substituted argument window, freshness side rows, and transported site indices used by the residual branch.*
4. *J is the candidate join row. It records the two displayed reducts and the bounded local join data that would close the residual beta peak.*
5. *D is the conditional confluence handoff row, pointing to the named substitution-compatibility premise consumed by the MetaCIC confluence bridge.*
6. *T records componentwise* hsame *transport for M, R, S, J, D and the structural rows.*
7. *C records displayed* Cont *replay of the route*

$$\mathsf{MetaCICResidualCommutationWitness}^{\uparrow} \longmapsto D \longmapsto \textit{conditional MetaCIC confluence bridge.}$$

8. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualCommutationWitness}^{\uparrow}$*, the MetaCIC confluence audit map, residual windows, substitution-site ledgers, candidate joins, conditional handoffs,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualCommutationWitness}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $M, R, S, J, D, T, C, P, N$ and has no coordinate for an unconditional closed Church–Rosser theorem, a hidden substitution axiom, quotient term equality, host freshness choice, global normalization, or an ambient confluence oracle.

Lean4 (statement) `BEDC.Derived.MetaCICResidualCommutationWitnessUp`

Theorem 7977.3 (MetaCIC residual-commutation witness obligations)**.** *For every accepted* MetaCICResidualComm *carrier*

$$W = (M, R, S, J, D, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualCommutationWitness}^{\uparrow}}$ *surface consists exactly of carrier admission, residual-window exposure, substitution-site ledger exposure, candidate-join recording, conditional confluence handoff, componentwise transport, replay, provenance, and local naming. A confluence consumer may read the witness only after it has displayed the same residual window R, the same substitution ledger S, the same candidate join row J, and the same conditional handoff D.*

The certificate exports no unconditional MetaCIC confluence theorem, no hidden substitution principle, no quotient term equality, no selected freshness schedule, no global normalization theorem, and no ambient confluence oracle.

Proof. Carrier admission is projection from Definition 7977.2. The accepted packet exposes exactly the MetaCIC source row, residual window, substitution-site ledger, candidate join row, conditional handoff, transport, replay, provenance, and local naming.

The residual and substitution rows R, S are finite. They name the positions and binder-site data used by the local branch, so the packet cannot silently quantify over all residual beta branches.

The candidate join row J is then read only as displayed local join data. The handoff row D keeps the confluence result conditional on the named substitution-compatibility boundary rather than presenting the witness as a closed Church–Rosser theorem.

Finally T, C, P, N transport, replay, source, and name only those rows. Since the excluded confluence and equality principles have no coordinate in the carrier, they cannot appear as local NameCert obligations. □

Theorem 7977.4 (MetaCIC residual-commutation witness confluence-map route)**.** *Every confluence-map read of an accepted* MetaCICResidualCommutationWitness$^{\uparrow}$ *packet factors through the displayed route*

residual window, *substitution-site ledger*, *candidate join*, *conditional handoff*, hsame, Cont, Pkg, NameCert.

The read first fixes the MetaCIC source row M, then the residual window R, then the substitution-site ledger S, then the candidate join row J, and only then the conditional handoff D consumed by the confluence audit map. It does not claim the missing substitution-compatibility input, arbitrary closed-term Church–Rosser, quotient term equality, or global normalization.

Proof. Project the accepted packet. A confluence-map consumer receives no row except $M, R, S, J, D, T, C, P, N$, so its first data are the displayed source and residual window.

The substitution-site ledger S and candidate join row J then bind the local peak. They provide bounded branch data, not a global substitution theorem.

The handoff D keeps the route aligned with chapter 9378. Transport, replay, provenance, and local naming preserve that same conditional boundary, so every confluence-map read factors through the stated finite route. □

Falsifiable prediction: Every accepted MetaCICResidualCommutationWitness$^{\uparrow}$ consumer exposes a MetaCIC source row, residual window, substitution-site ledger, candidate join row, conditional confluence handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before reading the confluence-map boundary.

Independence witness: metacic-confluence-audit, metacic-residual-substitution-boundary, metacic-local-join-ledger, metacic-confluence-residual-checker, parallel-diamond-residual: the confluence audit map names the conditional boundary without a local witness carrier; the residual-substitution boundary stores premise shape without the candidate join row; the local-join ledger records joins without substitution-site commutation data; the residual checker exposes bounded checks without the confluence handoff; the parallel-diamond residual packet names a diamond-facing route without this residual-substitution witness.

Closure status for NameCert$_{\text{MetaCICResidualCommutationWitness}^{\uparrow}}$.

Constructive story: A MetaCICResidualCommutationWitness$^{\uparrow}$ packet is generated from a MetaCIC source row, a residual window, a substitution-site ledger, a candidate join row, a conditional confluence handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICResidualCommutationWitness}^{\uparrow}}$, seed)

Scope closed: Definition 7977.2, Definition 7977.3, and Definition 7977.4 seed the finite residual-commutation witness route over MetaCIC residual windows, substitution-site ledgers, candidate joins, conditional handoffs, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICResidualCommutationWitness}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim substitution compatibility, arbitrary closed-term Church–Rosser, quotient term equality, selected freshness schedules, global normalization, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-window exposure, substitution-site ledger exposure, candidate-join recording, conditional handoff exactness, classifier transport, and consumer non-escape for BEDC.Derived.MetaCICResidualCommutationWitnessUp.

Chapter 7978

A Concrete Naming Certificate for $\mathsf{FiniteDimensionalBanach}^{\uparrow}$

Following Banach-space and finite-dimensional normed-space treatments, the $\mathsf{FiniteDimensionalBanach}^{\uparrow}$ packet records the coordinate route by which a displayed finite-dimensional normed vector space is consumed as complete. It sits after the Banach certificate of chapter 90, the vector-space carrier of chapter 47, the norm interface of chapter 89, the real-name seal of chapter 38, the regular rational sequence surface of chapter 301, the finite norm-comparison route of chapter 5566, and the finite-net compactness packet of chapter 6043. The packet names finite coordinate completeness as displayed NameCert data; it is not general Hahn–Banach separation, arbitrary open-cover compactness, quotient vector equality, or a selected completion carrier.

Definition 7978.1 (FiniteDimensionalBanach carrier)**.** *A* $\mathsf{FiniteDimensionalBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, N, R, S, E, K, L, W, H, C, P, Q)$$

with the following displayed rows:

1. *V is the finite* $\mathsf{VecSpace}^{\uparrow}$ *row, including the displayed coordinate-window dimension, basis-window ledger, zero, addition, negation, scalar source, and scalar action.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row attached to V, naming the norm read used by every Cauchy and limit comparison.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row through which norm values, comparison constants, and error budgets are read.*
4. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row recording the regular rational error schedule for the inspected Cauchy sequence.*
5. *E is the* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *row tying the norm comparison to the same finite coordinate window.*
6. *K is the coordinate-Cauchy ledger, listing the finitely many coordinate sequences extracted from the norm-Cauchy read.*
7. *L is the coordinate-limit ledger, recording the regular rational names that serve as limits for the displayed coordinate sequences.*
8. *W is the reconstructed limit-window row, repacking the coordinate limits as a vector row over V before any Banach-facing read is exported.*

9. *H stores componentwise* hsame *transport for V, N, R, S, E, K, L, W and the local naming row.*
10. *C stores displayed* Cont *replay from a finite-dimensional completeness request through the coordinate extraction, coordinate limits, and reconstructed limit window.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalBanach}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$, $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *Q is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBanach}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host completion functor, quotient vector equality, unrestricted Hahn–Banach separation, arbitrary open-cover compactness, selected bases outside the packet, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7978.2 (FiniteDimensionalBanach NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalBanach}^{\uparrow}$ *carrier $F = (V, N, R, S, E, K, L, W, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBanach}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed vector-space, norm, real-seal, regular-rational error, finite norm-equivalence, coordinate-Cauchy, coordinate-limit, reconstructed limit-window, transport, replay, provenance, and local naming rows. A completeness-facing read is admitted only after the coordinate Cauchy ledger K, coordinate-limit ledger L, and reconstructed limit window W are visible in the same finite packet.*

Proof. Project the carrier of Definition 7978.1. The vector, norm, scalar, and regular-rational source rows are exactly V, N, R, S.

Read the finite-dimensional comparison row next. The row E keeps norm comparisons on the displayed coordinate window, so Cauchy control can be transferred only through named finite coordinates.

The completeness data is then K, L, W. The coordinate-Cauchy ledger K exposes finitely many coordinate sequences, L records their regular rational limits, and W repacks those limits as the vector read consumed by Banach-facing users.

Finally H, C, P, Q preserve the same packet under componentwise transport, displayed replay, provenance, and local naming. Since the carrier has no row for a host completion functor, quotient vector equality, unrestricted separation, arbitrary compactness, selected external bases, countable choice, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 7978.3 (FiniteDimensionalBanach coordinate completeness handoff)**.** *Every Banach-facing read of an accepted* $\mathsf{FiniteDimensionalBanach}^{\uparrow}$ *carrier factors through the finite coordinate route*

$$\mathsf{VecSpace}^{\uparrow},\quad \mathsf{Norm}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{FiniteDimensionalNormEquivalence}^{\uparrow},\quad \mathsf{FiniteDimensionalCompactnes}$$

The route first displays the finite vector and norm rows, then reads the regular rational Cauchy budget, transfers it to finitely many coordinate sequences, records their coordinate limits, and repacks the resulting vector through the reconstructed limit window. It exports finite-coordinate completeness only for the named packet and does not export general Hahn–Banach, choice-based compactness, quotient completion, or ambient real completeness.

Proof. Start from the obligation surface of Definition 7978.2. A Banach-facing consumer must open V, N, R, S before any Cauchy or limit read is available.

Next use the finite norm-comparison row E. It is the only route from the norm-Cauchy budget to coordinate Cauchy data, so the handoff remains tied to the displayed finite coordinate window.

The coordinate ledgers K and L supply the finite limit data, and W repacks those limits as the vector-window read. The compactness-facing sibling of chapter 6043 may supply bounded-window consumers, but no compactness or choice principle is added to this packet.

The structural rows H, C, P, Q replay and name the same route for downstream Banach consumers. Therefore the exported completeness claim is finite-coordinate completeness over the listed rows, not a host completion theorem, a quotient construction, general Hahn–Banach separation, arbitrary open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalBanach}^{\uparrow}$ packet is generated from a displayed VecSpaceUp row, NormUp row, RealUp scalar seal, RegSeqRat error schedule, finite-dimensional norm-equivalence row, coordinate-Cauchy ledger, coordinate-limit ledger, reconstructed limit-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalBanach}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7978.1, Definition 7978.2, and Definition 7978.3 seed the finite-coordinate completeness route over displayed vector, norm, real, regular-rational, coordinate-limit, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalBanach}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Hahn–Banach, arbitrary open-cover compactness, quotient vector equality, selected external bases, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite vector and norm exposure, coordinate Cauchy extraction, coordinate-limit readback, reconstructed limit-window exactness, transport stability, replay stability, provenance, and local naming for BEDC.Derived.FiniteDimensionalBanachUp.

Chapter 7979

A Concrete Naming Certificate for $\mathsf{BanachMazurDistance}^{\uparrow}$

Following Banach and Mazur's finite-dimensional distance formalism, the $\mathsf{BanachMazurDistance}^{\uparrow}$ packet records a finite distortion comparison between two displayed normed or Banach rows as NameCert data. It sits over the metric carrier of chapter 57, the norm carrier of chapter 89, the Banach-facing carrier of chapter 90, the located-limit route of chapter 1420, and the real-name seal of chapter 38. The packet names a displayed pair of inverse-candidate rows and a finite $\mathsf{Real}^{\uparrow}$ distortion ledger; it does not assert a host infimum, a quotient isometry class, a Hahn–Banach separation theorem, or a choice-selected optimal isomorphism.

Definition 7979.1 (Banach–Mazur distance carrier)**.** *A* $\mathsf{BanachMazurDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, F, G, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source space row, read through* $\mathsf{NormedSpace}^{\uparrow}$ *or* $\mathsf{Banach}^{\uparrow}$ *and carrying its displayed norm, metric, finite-dimensional coordinate window, and completion-facing* $\mathsf{LocatedLimit}^{\uparrow}$ *references when present.*

2. *Y is the target space row, read through the same normed or Banach-facing surface and carrying its displayed norm, metric, finite-dimensional coordinate window, and located-limit references when present.*

3. *F is the forward inverse-candidate row from X to Y, recording the displayed finite matrix or finite linear-action table, its domain window, and its* $\mathsf{Real}^{\uparrow}$ *operator-bound ledger.*

4. *G is the reverse inverse-candidate row from Y to X, recording the displayed finite matrix or finite linear-action table, its domain window, and its* $\mathsf{Real}^{\uparrow}$ *operator-bound ledger.*

5. *L is the distortion ledger, storing the two forward and reverse norm-bound seals and the displayed product bound as* $\mathsf{Real}^{\uparrow}$ *data.*

6. *R is the inverse-replay row, recording that $G \circ F$ and $F \circ G$ return to the displayed source and target windows by finite coordinate checks, not by a host quotient or an ambient isometry class.*

7. *H stores componentwise* hsame *transport for the source row, target row, forward candidate, reverse candidate, distortion ledger, inverse replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a distance-facing request through X, Y, F, G, L, R, H.*

9. *P is the* Pkg *provenance over* $\mathsf{BanachMazurDistance}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachMazurDistance}^{\uparrow}}$ *row.*

The carrier has no coordinate for an external infimum over all isomorphisms, a quotient isometry class, an unrestricted Hahn–Banach separation argument, a selected optimal map, or an appeal to countable choice.

Definition 7979.2 (Banach–Mazur distance classifier)**.** *Two accepted* $\mathsf{BanachMazurDistance}^{\uparrow}$ *packets*

$$D = (X, Y, F, G, L, R, H, C, P, N) \quad \text{and} \quad D' = (X', Y', F', G', L', R', H', C', P', N')$$

are classified together exactly when the transported distortion ledgers L and L' expose the same displayed $\mathsf{Real}^{\uparrow}$ *product-bound comparison after the same source and target rows have been read. The classifier does not compare host spaces, quotient isometry classes, optimal maps, or external infima; it compares only the finite ledger reached through H, C from X, Y, F, G, R.*

Theorem 7979.3 (Banach–Mazur distance NameCert obligations)**.** *For every accepted* $\mathsf{BanachMazurDistance}^{\uparrow}$ *carrier $D = (X, Y, F, G, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachMazurDistance}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *source and target admission: X and Y are displayed normed or Banach-facing rows with metric and located-limit dependencies available only through their named packets;*
2. *inverse-candidate admission: F and G are the only forward and reverse rows available to the certificate;*
3. *distortion-ledger exactness: L stores the displayed* $\mathsf{Real}^{\uparrow}$ *norm-bound product read from F and G;*
4. *inverse replay: R records the finite coordinate-window checks for $G \circ F$ and $F \circ G$;*
5. *structural preservation: H, C, P, N transport, replay, source, and name the same rows without adding an optimal isomorphism, host infimum, quotient isometry class, Hahn–Banach theorem, or choice principle.*

Proof. Project the carrier of Definition 7979.1. The source and target rows are X, Y, and all norm, metric, Banach, and located-limit dependencies are read through those displayed rows.

Read the inverse-candidate rows next. The packet contains exactly F and G as finite forward and reverse action data, so every distance-facing request must use those two rows before consulting any distortion value.

The distortion row is L. It records the two operator-bound reads and their displayed $\mathsf{Real}^{\uparrow}$ product, while the inverse-replay row R records the finite coordinate-window return checks.

Finally H, C, P, N preserve the same finite packet under transport, replay, provenance, and local naming. Since the carrier has no row for a host infimum, quotient isometry class, selected optimal map, Hahn–Banach separation, or countable choice, the listed rows exhaust the NameCert surface. □

Theorem 7979.4 (Banach–Mazur distance distortion ledger route)**.** *Every distortion-facing consumer of an accepted* $\mathsf{BanachMazurDistance}^{\uparrow}$ *packet factors through*

$$X, Y, F, G, L, R, H, C, P, N.$$

The terminal comparison is the displayed $\mathsf{Real}^{\uparrow}$ *ledger L, transported and replayed through H, C after the source, target, forward, reverse, and inverse-replay rows have been exposed.*

Proof. Start with the source and target rows X, Y. They bind the normed or Banach-facing carrier, the $\mathsf{Met}^{\uparrow}$ readback, and any $\mathsf{LocatedLimit}^{\uparrow}$ completion-facing references before a distortion request is allowed.

Next read F and G. These rows are finite inverse candidates with displayed operator-bound data, and they are the only rows from which the distortion ledger may be computed.

The ledger route then passes through R and L. The row R records the finite inverse replay, and L seals the two norm-bound reads as a $\mathsf{Real}^{\uparrow}$ product comparison.

The structural rows H, C, P, N preserve this same route. No later consumer may replace the displayed ledger by an external infimum, quotient-isometry class, selected optimal isomorphism, Hahn–Banach argument, or choice-derived map. □

Theorem 7979.5 (Banach–Mazur distance composition boundary)**.** *If two accepted* $\mathsf{BanachMazurDistance}^{\uparrow}$ *packets share a displayed middle space row, then a composition-facing consumer may concatenate their finite inverse-candidate rows only by constructing a new displayed ledger from the two source ledgers and the two inverse-replay rows. The boundary exports a composed finite distortion row; it does not export the host multiplicative infimum law for Banach–Mazur distance, a quotient isometry-class statement, or a choice-selected optimal factorization.*

Proof. Take the two accepted packets and inspect their source, middle, and target rows. The shared middle space must be the displayed row transported by the two packets, not a host-identified quotient or an unnamed isometry class.

Concatenate the finite candidate rows. The forward action is read by composing the two displayed forward candidates, and the reverse action is read by composing the two displayed reverse candidates; both reads remain inside the finite coordinate windows named by the packets.

Build the new ledger from the old ledgers. The composed $\mathsf{Real}^{\uparrow}$ bound is obtained by multiplying the displayed product-bound rows already stored in the two distortion ledgers, while the inverse-replay evidence is rebuilt from the two replay rows.

The result is therefore a finite composition boundary for ledgers, not the external Banach–Mazur distance inequality over all isomorphisms. Since no packet contains a host infimum, quotient isometry class, selected optimum, Hahn–Banach separation row, or choice principle, none of those objects can be exported by composition. □

Falsifiable prediction: An accepted $\mathsf{BanachMazurDistance}^{\uparrow}$ read is refuted if the $\mathsf{Real}^{\uparrow}$ distortion ledger L is consumed before the source row X, target row Y, forward candidate F, reverse candidate G, and inverse-replay row R are displayed.

Independence witness: metric, norm, banach, locatedlimit, real: no listed sibling supplies the joint source-space rows, two inverse-candidate rows, finite RealUp distortion ledger, inverse replay, transport, continuation, provenance, and local NameCert row while refusing host infima, quotient isometry classes, selected optimal maps, Hahn–Banach separation, and choice.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachMazurDistance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachMazurDistance}^{\uparrow}$ packet is generated from two displayed normed or Banach-facing source rows, two finite inverse-candidate rows, a RealUp distortion ledger, inverse replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachMazurDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7979.1, Definition 7979.2, Definition 7979.3, Definition 7979.4, and Definition 7979.5 seed the finite distortion-ledger packet consumed by normed-space and Banach-facing comparison arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachMazurDistance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infimum existence, quotient isometry classes, Hahn–Banach separation, selected optimal isomorphisms, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, inverse-candidate admission, distortion-ledger exactness, inverse replay, composition-boundary routing, transport, replay, provenance, and local naming for BEDC.Derived.BanachMazurDistanceUp.

Chapter 7980

A Concrete Naming Certificate for $\mathsf{SequentialContinuityReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentialContinuityReal}^{\uparrow}$ packet specializes the existing $\mathsf{SequentialContinuity}^{\uparrow}$ route to Real-and-Completion consumers over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It sits after the certified sequence-consumer packet of chapter 1466, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives completion-facing consumers a finite handoff surface without restating the general sequential-continuity carrier or importing host completeness.

Definition 7980.1 (Sequential continuity Real carrier)**.** *A* $\mathsf{SequentialContinuityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, S, L, W_X, W_Y, A_X, A_Y, E_X, E_Y, H, C, P, N)$$

with the following displayed rows:

1. *Q is the already accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *packet whose pointwise-continuity, metric-row, and inspected-threshold data are being consumed.*
2. *S is the source sequence surface, exposed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
3. *L is the finite ledger of inspected convergence thresholds read from the sequential-continuity packet.*
4. *W_X and W_Y are the displayed source and image windows used by the Real-facing consumer.*
5. *A_X and A_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the source and image windows.*
6. *E_X and E_Y are the corresponding* $\mathsf{Real}^{\uparrow}$ *seal rows available only after the regular-rational readbacks.*
7. *H records componentwise* hsame *transport for the sequential-continuity packet, stream rows, regular readbacks, Real seals, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from Q, S, L to $W_X, W_Y, A_X, A_Y, E_X, E_Y$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialContinuityReal}^{\uparrow}$*,* $\mathsf{SequentialContinuity}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuityReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host sequential-continuity theorem, quotient stream equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 7980.2 (Sequential continuity Real NameCert obligations)**.** *For every accepted* $\mathsf{SequentialContinuityReal}^{\uparrow}$ *carrier* $R = (Q, S, L, W_X, W_Y, A_X, A_Y, E_X, E_Y, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuityReal}^{\uparrow}}$ *surface consists exactly of carrier admission, consumption of the displayed* $\mathsf{SequentialContinuity}^{\uparrow}$ *row* Q*, finite stream-window exposure through* S, L, W_X, W_Y*, source and image* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* A_X, A_Y*, source and image* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E_X, E_Y*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7980.1. The only analytic input from sequential continuity is the displayed row Q, and the only sequence observations are the finite windows W_X and W_Y read from the stream surface S at the thresholds listed in L.

The Real-facing rows are downstream from regular readback. The source side passes through A_X before E_X can be consumed, and the image side passes through A_Y before E_Y can be consumed.

The remaining rows do not add a second analytic surface. They transport, replay, source, and name the same finite route through H, C, P, N, so the listed obligations exhaust the local certificate surface. □

Theorem 7980.3 (Sequential continuity Real stream readback)**.** *Every accepted* $\mathsf{SequentialContinuityReal}^{\uparrow}$ *packet reads any source or image stream observation through a finite* $\mathsf{StreamName}^{\uparrow}$ *window before presenting it to a* $\mathsf{RegSeqRat}^{\uparrow}$ *row. In particular, every Real-facing route factors through*

$$Q, S, L \longmapsto W_X, W_Y \longmapsto A_X, A_Y \longmapsto E_X, E_Y$$

together with H, C, P, N*.*

Proof. Begin with the stream coordinates. The carrier exposes S only through the finite threshold ledger L, and the displayed windows W_X, W_Y are the only observations obtained from that schedule.

Next read the regular-rational rows. The rows A_X and A_Y are attached to those same windows, so a consumer cannot reach the source or image Real seal directly from S or from an unrecorded stream equality.

Finally the seal rows E_X, E_Y are downstream from the regular readbacks, while H, C, P, N preserve and name the same dependency route. This proves the stated factorization. □

Theorem 7980.4 (Sequential continuity Real composition handoff)**.** *Let* $R = (Q, S, L, W_X, W_Y, A_X, A_Y, E_X, E_Y,$ *be an accepted* $\mathsf{SequentialContinuityReal}^{\uparrow}$ *carrier. Any completion-facing consumer that composes the displayed sequential-continuity packet with Real readback must use* Q *only before the finite window and regular-rational stages, and must use* E_X, E_Y *only after* A_X, A_Y*. Thus the handoff is a finite BEDC route over* $\mathsf{SequentialContinuity}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, not a separate host theorem about completed real functions.*

Proof. Read the composition order from the carrier. The row Q supplies the certified sequential-continuity route, but the Real consumer does not inspect it as a completed function; it is consumed only through the stream windows named by S, L, W_X, W_Y.

The regular readback stage then mediates the passage to Real seals. The rows A_X, A_Y are the only regular-rational handoffs in the packet, and the rows E_X, E_Y are downstream from them.

The transport, replay, provenance, and naming rows keep this order stable under componentwise comparison of accepted packets. Therefore the completion-facing composition is exactly the finite handoff named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialContinuityReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialContinuityReal}^{\uparrow}$ packet is generated from an accepted SequentialContinuity row, finite StreamName windows, RegSeqRat readback rows, RealUp seal rows, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialContinuityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7980.1, Definition 7980.2, Definition 7980.3, and Definition 7980.4 seed the Real-and-Completion handoff surface for sequential-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialContinuityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sequential-continuity theorems, quotient stream equality, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window readback, RegSeqRat handoff, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.SequentialContinuityRealUp.

Chapter 7981

A Concrete Naming Certificate for $\mathsf{RasiowaSikorski}^{\uparrow}$

Following Rasiowa and Sikorski on dense-set forcing and its constructive readings, the $\mathsf{RasiowaSikorski}^{\uparrow}$ packet records the finite dense-open refinement route that a completion-facing consumer may inspect before any generic-filter claim is public. It sits beside the finite Baire-category surface of chapter 7959, the Baire-space source of chapter 334, the metric carrier of chapter 57, the finite observation schedule of chapter 126, and the terminal real-name seal of chapter 38. The packet names only displayed condition rows, dense-open request rows, finite refinements, finite-meet witnesses, generic-prefix rows, transport, replay, provenance, and local naming.

Definition 7981.1 (Rasiowa-Sikorski carrier)**.** *A* $\mathsf{RasiowaSikorski}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, D, Rf, F, G, H, C, Q, N)$$

with the following displayed rows:

1. *P is the condition row, listing the finite condition prefix currently visible to the packet.*
2. *D is the dense-open request row, recording only the displayed dense-open obligations being met.*
3. *Rf is the refinement row, naming the finite strengthening relation from the visible condition prefix into the requested dense-open row.*
4. *F is the finite-meet row, recording compatibility for the finite family of refinements already displayed.*
5. *G is the generic-prefix row, exposing a finite prefix that meets the displayed dense-open requests without selecting a host generic filter.*
6. *H records componentwise* hsame *transport for the condition, dense-open request, refinement, finite-meet, generic-prefix, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from condition inspection through dense-open admission, refinement, finite meets, generic-prefix exposure, transport, provenance, and local naming.*
8. *Q is the* Pkg *provenance over* $\mathsf{RasiowaSikorski}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RasiowaSikorski}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host generic filter, countable choice, arbitrary countable meets, quotient streams, selected generic points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7981.2 (Rasiowa-Sikorski NameCert obligations)**.** *For every accepted* $\mathsf{RasiowaSikorski}^{\uparrow}$ *carrier* $R = (P, D, Rf, F, G, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RasiowaSikorski}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the condition row, dense-open request row, refinement row, finite-meet row, generic-prefix row, componentwise transport, replay, provenance, and local naming rows;*
2. *dense-open admission, requiring every public request in* D *to be displayed before any refinement row can be consumed;*
3. *refinement monotonicity, requiring* Rf *to strengthen the visible condition prefix rather than replace it by an external selector;*
4. *finite meet compatibility, requiring the finite family of displayed refinements to be checked by* F *before the generic-prefix row is public;*
5. *generic-prefix nonescape, requiring* G *to expose only finite prefix data and no host generic filter or selected point;*
6. *provenance, requiring* Q *and* N *to keep the route inside the displayed BaireSpace, MetricSpace, StreamName, Real,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Project the carrier of Definition 7981.1. The rows P, D, Rf, F, G provide the condition, dense-open request, refinement, finite-meet, and generic-prefix data, while H, C, Q, N provide transport, replay, provenance, and local naming.

Dense-open admission is the row D. Since Rf is downstream of D, a consumer cannot read a refinement before the corresponding displayed request is visible.

Refinement monotonicity and finite meet compatibility are the rows Rf and F. The refinement strengthens the visible condition prefix, and the finite meet row checks the displayed finite family before the prefix row G can be exposed.

The structural rows preserve the same route. Transport through H, replay through C, provenance through Q, and naming through N do not add coordinates outside the finite packet, so the NameCert surface is exactly the listed obligation set. □

Theorem 7981.3 (Rasiowa-Sikorski finite dense-open route)**.** *Every completion-facing consumer of an accepted* $\mathsf{RasiowaSikorski}^{\uparrow}$ *carrier receives only finite dense-open refinement rows. More precisely, the consumer can read the condition row* P*, the dense-open request row* D*, the refinement row* Rf*, the finite-meet row* F*, the generic-prefix row* G*, and the structural rows* H, C, Q, N*, and it cannot read a host generic filter, countable choice, arbitrary countable meets, quotient stream equality, selected generic points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the obligation surface of Definition 7981.2. It places every public completion-facing request behind the displayed condition and dense-open rows.

The refinement row Rf and finite-meet row F are finite data. They record only the strengthenings and compatibility checks already present in the accepted packet.

The generic-prefix row G is downstream of those finite rows. It exposes a prefix meeting the displayed requests, not a completed generic filter or selected point.

Finally H, C, Q, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, arbitrary countable meets, quotient streams, selected generic points, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported to a completion-facing consumer. □

Falsifiable prediction: Every accepted $\mathsf{RasiowaSikorski}^{\uparrow}$ read exposes condition rows, dense-open request rows, refinement rows, finite-meet rows, generic-prefix rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing consumer receives the packet.

Independence witness: baire-space, constructive-baire-category, metricspace, streamname, real: $\mathsf{RasiowaSikorski}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite dense-open requests, monotone refinements, finite-meet compatibility, generic-prefix nonescape, transport, replay, provenance, and local naming while refusing host generic filters, countable choice, arbitrary countable meets, quotient streams, selected generic points, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RasiowaSikorski}^{\uparrow}}$.

Constructive story: A $\mathsf{RasiowaSikorski}^{\uparrow}$ packet is generated from displayed condition rows, dense-open request rows, refinement rows, finite-meet rows, generic-prefix rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RasiowaSikorski}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7981.1, Definition 7981.2, and Definition 7981.3 seed the finite dense-open refinement surface for completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RasiowaSikorski}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, arbitrary countable meets, selected generic points, quotient streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open admission, refinement monotonicity, finite meet compatibility, generic-prefix nonescape, transport replay, provenance, and local naming for BEDC.Derived.RasiowaSikorskiUp.

Chapter 7982

A Concrete Naming Certificate for $\mathsf{CompactIntervalChain}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompactIntervalChain}^{\uparrow}$ packet records a nested finite chain of located compact interval rows as BEDC NameCert data. It sits over the compact interval carrier of chapter 5553, the located endpoint surface of chapter 394, the dyadic tolerance core of chapter 291, the StreamName finite-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite chain data needed by nested-interval and compact-real consumers; it is not an arbitrary open-cover compactness theorem, a selected-subcover principle, a quotient real equality test, a countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7982.1 (Compact interval chain carrier)**.** *A* $\mathsf{CompactIntervalChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, E, D, W, R, A, H, C, P, N)$$

with the following rows:

1. *L is the finite chain ledger of nested located compact interval rows, each row carrying displayed left and right endpoint names and an inclusion witness into the previous row.*
2. *E records endpoint rows for the chain, including the visible* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{CompactRealInterval}^{\uparrow}$ *boundary data consumed by interval-local readers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance ledger, recording common scales, endpoint slack, diameter bounds, and dyadic containment checks for every row in the finite chain.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to inspect endpoint and mesh data without opening an infinite stream or selecting a real representative.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the endpoint, nesting, diameter, mesh, and tolerance checks as finite regular rational windows.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed endpoint reads and compact interval seals.*
7. *H records componentwise* hsame *transport for chain rows, endpoint rows, dyadic mesh rows, StreamName windows, regular rational readbacks, Real seals, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a chain request through $L, E, D, W, R, A, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalChain}^{\uparrow}$, $\mathsf{CompactRealInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, E, D, W, R, A, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness.

Theorem 7982.2 (Compact interval chain NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalChain}^{\uparrow}$ *carrier*

$$Q = (L, E, D, W, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}$ *surface consists exactly of the displayed finite obligations: chain-row admission for L, endpoint and compact-interval admission for E, dyadic mesh and diameter admission for D, finite StreamName window admission for W, regular rational readback for R, terminal Real sealing for A, componentwise* hsame *transport for H, displayed replay for C, provenance for P, and local naming for N.*

Proof. Project the carrier of Definition 7982.1. The chain ledger L gives only finitely many nested interval rows, and E exposes the endpoint and compact-interval boundary attached to each row.

Read the analytic rows next. The dyadic ledger D supplies the common-scale, mesh, containment, endpoint-slack, and diameter data; W restricts endpoint and mesh inspection to finite StreamName windows.

The regular and Real-facing rows are downstream. The row R repacks the inspected chain data as regular rational windows, and A seals exactly those endpoint and compact-interval reads as $\mathsf{Real}^{\uparrow}$ data.

Finally, H, C, P, N transport, replay, source, and locally name the same finite packet. Since no carrier coordinate stores arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient completeness, the displayed rows exhaust the NameCert obligations. □

Theorem 7982.3 (Compact interval chain finite-cover induction)**.** *Every accepted finite* $\mathsf{CompactIntervalChain}^{\uparrow}$ *consumer that propagates a compact-interval estimate along the chain may use only induction over the displayed finite ledger L. At the base row it reads the endpoint, dyadic, StreamName, RegSeqRat, Real seal, transport, replay, provenance, and naming rows of the first interval; at each successor row it consumes the next inclusion and diameter data from $L, E, D, W, R, A, H, C, P, N$. The induction does not select subcovers, quotient real representatives, countable-choice schedules, or an ambient completeness witness.*

Proof. The base step is the first displayed interval row of L. By Definition 7982.2, the consumer receives its endpoint rows, dyadic mesh and diameter data, StreamName windows, regular rational readback, Real seal, transport, replay, provenance, and local name as finite data.

For the successor step, read the next chain row in L together with the endpoint and compact-interval boundary in E. The dyadic ledger D records the containment, diameter, and endpoint slack needed to pass the estimate from the previous row to the next row.

The handoff remains finite because W, R, A expose only the inspected windows, regular readbacks, and Real seals for the current row. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Induction over the finite ledger therefore proves the advertised propagation route. No step asks for an open-cover selector, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects appears in the carrier. □

Falsifiable prediction: Every $\mathsf{CompactIntervalChain}^{\uparrow}$ read factors through finite nested interval rows, compact endpoint rows, dyadic mesh and diameter ledgers, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seals, componentwise hsame, displayed Cont, Pkg, and a local $\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}$ row.

Independence witness: compactrealinterval, locatedinterval, dyadicratcore, streamname, regseqrat, real: $\mathsf{CompactIntervalChain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite nested-chain ledger, compact endpoint rows, dyadic containment and diameter rows, finite stream windows, regular readbacks, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalChain}^{\uparrow}$ packet is generated from finite nested located compact interval rows, endpoint rows, dyadic mesh and diameter ledgers, StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7982.1, Definition 7982.2, and Definition 7982.3 seed the finite nested compact interval chain carrier over compact endpoint rows, dyadic meshes, finite windows, regular readback, Real seals, and finite chain induction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactIntervalChain}^{\uparrow}$.

Upgrade path: Obligation closure requires checked chain-row admission, endpoint sealing, dyadic containment and diameter stability, StreamName window pullback, RegSeqRat readback, replay stability, and nonescape for BEDC.Derived.CompactIntervalChainU

Chapter 7983

A Concrete Naming Certificate for $\mathsf{LocatedRealSpread}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedRealSpread}^{\uparrow}$ packet realizes the Real-completeness direction of chapter 9238 as a finite located-window NameCert object. It binds the terminal real-name seal of chapter 38, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the tail-budget siblings used by the completion route. The packet records finite located reads for a spread of Real-facing windows; it does not assert full $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, or an ambient Cauchy completion theorem.

Definition 7983.1 (Located real spread carrier)**.** *A* $\mathsf{LocatedRealSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, L, D, R, E, H, C, P, N)$$

with the following rows:

1. *W is the finite spread-window ledger, listing the StreamName requests and local indices at which Real-facing reads are inspected.*
2. *L is the located read row, carrying lower and upper dyadic bracketing data for every displayed window together with apartness refusals outside the inspected spread.*
3. *D is the dyadic tail-budget row over* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording precision thresholds, common-scale tolerances, and finite tail comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the located windows and dyadic tail budgets as regular rational finite data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed located reads and their regular rational witnesses.*
6. *H records componentwise* hsame *transport for spread windows, located Real reads, dyadic tail budgets, regular rational readbacks, Real seals, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from a located-spread request through W, L, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealSpread}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, the Cauchy-tail budget siblings,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, L, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, full Real completeness, or ambient completion.

Theorem 7983.2 (Located real spread NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealSpread}^{\uparrow}$ *carrier*

$$S = (W, L, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$ *surface consists exactly of finite spread-window admission for W, located read admission for L, dyadic tail-budget admission for D, regular rational readback for R, terminal Real sealing for E, componentwise* hsame *transport for H, displayed replay for C, provenance for P, and local naming for N.*

Proof. Project the carrier of Definition 7983.1. The row W lists the finite windows being inspected, and L supplies the located bracketing reads attached to those windows.

Read the tail-budget row D. It records precision thresholds, common dyadic scales, and finite tail comparisons before any regular rational or Real-facing row is consumed.

The readback row R repacks the located windows and dyadic budgets as $\mathsf{RegSeqRat}^{\uparrow}$ data, and E seals exactly those finite reads as $\mathsf{Real}^{\uparrow}$ data.

The structural rows H, C, P, N then transport, replay, source, and locally name the same packet. Since the carrier contains no quotient stream equality, selected-limit operator, countable-choice schedule, full completeness theorem, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7983.3 (Located real spread regular readback)**.** *Every completion-facing consumer of an accepted* $\mathsf{LocatedRealSpread}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through $W, L, D, R, E, H, C, P, N$ before using the located spread as Real-completeness data. The route exports finite located-window reads, dyadic tail budgets, regular rational readbacks, terminal Real seals, transport, replay, provenance, and local naming, but not quotient stream equality, selected limits, countable choice, full $\mathsf{Real}^{\uparrow}$ *completeness, or an ambient completion operator.*

Proof. Start with the finite window ledger W and located read row L. These rows expose only the inspected spread windows and their dyadic bracketing data.

Use the dyadic tail-budget row D next. It attaches each located read to a displayed precision threshold and common-scale comparison before the readback row is available.

The regular readback row R is the only handoff to $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal row E is the only handoff to $\mathsf{Real}^{\uparrow}$. Thus a completion-facing reader receives regular rational evidence before it reads a Real seal.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. None of these rows adds quotient stream equality, a selected limit, countable choice, full Real completeness, or an ambient completion operator, so every accepted read remains inside the displayed finite route. □

Theorem 7983.4 (Located real spread obligation closure)**.** *The carrier of Definition 7983.1, the NameCert surface of Definition 7983.2, and the regular readback route of Definition 7983.3 form the obligation surface for* $\mathsf{LocatedRealSpread}^{\uparrow}$*. Every local obligation is one of the displayed rows $W, L, D, R, E, H, C, P, N$, and every completion-facing read factors through those rows before it can be used as located Real-spread data.*

Proof. Use the carrier projection. The rows W, L, D, R, E provide the finite spread windows, located bracketing reads, dyadic tail budgets, regular rational readbacks, and Real seals.

Apply the NameCert obligations to identify the structural rows H, C, P, N as transport, replay, provenance, and local naming for the same packet.

Finally use the regular readback theorem. It forces every completion-facing consumer through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the located spread is exported, so no additional local obligation remains outside $W, L, D, R, E, H, C, P, N$. □

Theorem 7983.5 (Located real spread L10 scope)**.** *The* $\mathsf{LocatedRealSpread}^{\uparrow}$ *packet is scoped to the L10 Real and Completion substrate consisting of* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, displayed* Cont *replay, componentwise* hsame *transport, and finite located-window rows. It exports no obligation that reads a quotient stream, selected limit, countable-choice schedule, full* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or ambient completion operator.*

Proof. Read the finite located-window rows first. The window ledger W and located row L determine the inspected spread requests and their dyadic bracketing data.

The dyadic row D, regular readback row R, and Real seal row E place those requests inside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The structural rows H and C provide exactly componentwise hsame transport and displayed Cont replay, while P, N record provenance and local naming. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, full Real completeness, or ambient completion, the L10 scope is precisely the listed substrate. □

Falsifiable prediction: Every $\mathsf{LocatedRealSpread}^{\uparrow}$ read factors through finite spread windows, located Real reads, dyadic tail budgets, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seals, componentwise hsame, displayed Cont, Pkg, and a local $\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$ row.

Independence witness: streamname, regseqrat, dyadicratcore, real, tailbudget: $\mathsf{LocatedRealSpread}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite spread windows, located bracketing reads, dyadic tail-budget rows, regular rational readbacks, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealSpread}^{\uparrow}$ packet is generated from finite spread windows, located Real reads, dyadic tail budgets, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$, obligation)

Scope closed: Definition 7983.1, Definition 7983.2, Definition 7983.3, Definition 7983.4, and Definition 7983.5 bind the finite located-window, dyadic-tail, RegSeqRat readback, StreamName, Real seal, transport, replay, provenance, and local naming obligation packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealSpread}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, full $\mathsf{Real}^{\uparrow}$ completeness, ambient completion, Lean verification, or bridge checking for $\mathsf{LocatedRealSpread}^{\uparrow}$.

Upgrade path: Obligation closure requires checked spread-window admission, located read stability, dyadic tail-budget exactness, RegSeqRat readback, Real sealing, replay stability, and nonescape for BEDC.Derived.LocatedRealSpreadUp.

Chapter 7984

A Concrete Naming Certificate for $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$

Bishop and Bridges, in the constructive treatment of real-number comparison, keep located interval evidence, regular Cauchy names, and terminal real seals as displayed data before any quotient equality or trichotomy principle is used. The $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$ packet records that comparison route as a finite BEDC NameCert surface over the located-interval witness, the regular rational stream, the dyadic tolerance ledger, the finite StreamName schedule, and the Real-facing seal. It sits beside the real-name chapter chapter 38, the regular rational route of chapter 301, the stream schedule of chapter 126, and the dyadic arithmetic ledger of chapter 291; it does not identify representatives, select limits, decide trichotomy, or import ambient completeness.

Definition 7984.1 (Bishop located Cauchy real comparison carrier)**.** *A* $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, R, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval witness row, giving the positive separation or enclosure evidence used for the requested comparison.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *stream row whose rational approximants are being compared.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic scale, endpoint, and comparison-budget rows used by the located witness.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that opens the finite windows required by I, R, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the located witness, regular stream, dyadic ledger, and stream schedule have been displayed.*
6. *H records componentwise* hsame *transport for located witnesses, regular rational stream rows, dyadic tolerance rows, stream schedules, Real seals, continuation replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from the comparison request through the stream window, from the window through the dyadic tolerance ledger, from the ledger through the regular rational row, and from that row to the Real-facing seal.*

8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$*, located interval data,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, R, D, S, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected completed limit, trichotomy, countable choice, host real comparison, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7984.2 (Bishop located Cauchy real comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$ *carrier*

$$L = (I, R, D, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}}$ *obligations are exactly carrier admission, located interval witness exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *stream readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility,* $\mathsf{StreamName}^{\uparrow}$ *schedule ordering,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every comparison read factors through*

$$I, S, D, R, E, H, C, P, N.$$

No quotient equality of Cauchy names, selected limit, trichotomy, host real order, countable-choice schedule, or ambient completeness row is exported.

Proof. Project the carrier of Definition 7984.1. Its rows expose the located witness, regular rational stream, dyadic tolerance ledger, stream schedule, Real seal, transport, replay, provenance, and local naming as the only classifier-visible data.

The comparison route first reads the finite witness rows. The schedule S opens the windows required by the located evidence I, the ledger D supplies the dyadic tolerance and endpoint rows for those windows, and the regular stream row R records the rational Cauchy data being compared.

The Real-facing part is downstream. The seal E is read only after I, S, D, R are displayed, while H, C, P, N transport, replay, source, and name the same finite route.

Since none of the excluded principles is a carrier coordinate, no accepted consumer can extract quotient real equality, selected limits, trichotomy, host order, countable choice, or ambient completeness from the local certificate. □

Theorem 7984.3 (Bishop located Cauchy real comparison sibling route)**.** *Every accepted* BishopLocatedCauchyRea *comparison consumes the L10 real-name corridor in the order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

as displayed by chapter 126, chapter 291, chapter 301, and chapter 38. The located interval witness is a comparison row over that corridor, not a replacement for the regular Cauchy name or the terminal Real seal.

Proof. Read the carrier rows S, D, R, E from Definition 7984.1. They are the stream window, dyadic tolerance ledger, regular rational readback, and terminal Real seal used by the comparison packet.

Apply the NameCert obligations of Definition 7984.2. The comparison read must factor through $I, S, D, R, E, H, C, P, N$, so the located witness I is attached to the same L10 corridor rather than to a hidden host real.

The non-escape clause then rules out quotient equality, selected limits, trichotomy, host order, countable choice, and ambient completeness. Thus the sibling route is the displayed finite real-name corridor with one located comparison witness over it. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}$ packet is generated from a located interval witness, a RegSeqRat stream row, a DyadicRatCore tolerance ledger, a StreamName schedule, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7984.1, Definition 7984.2, and Definition 7984.3 seed the located Cauchy real comparison packet over the L10 real-name corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyRealComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, trichotomy, host real order, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located witness exposure, regular rational readback, dyadic tolerance compatibility, schedule ordering, Real seal non-escape, transport, replay, provenance, and local naming.

Chapter 7985

A Concrete Naming Certificate for $\mathsf{Demicontinuity}^{\uparrow}$

Following Browder and Minty on weak-to-strong continuity for nonlinear operators, the $\mathsf{Demicontinuity}^{\uparrow}$ packet records the finite certificate surface for a map whose admissible input is read through weak tests while its output is controlled in a strong metric row. It sits over the Banach carrier of **??**, the Hilbert weak-test surface of chapter 6896, the monotone-operator route of chapter 6215, the scalar seal of chapter 38, and the metric readback surface of chapter 4230. The packet names only the displayed weak-input and strong-output handoff; it does not assert compactness, selected weak subsequences, maximal monotonicity, norm convergence of the input, or an ambient theorem turning weak convergence into strong convergence.

Definition 7985.1 (Demicontinuity carrier)**.** *A* $\mathsf{Demicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, W, M, R, A, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *source row, including the normed carrier, linear structure, and completeness-facing provenance inspected by the nonlinear map.*
2. *W is the weak-input row, read through* $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ *when the source is Hilbert-facing and through the displayed Banach weak-test rows otherwise.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *strong-output row, carrying the target distance, strong tolerance boundary, and metric readback used by consumers.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for weak-test values and strong-output radii.*
5. *A is the action row for the displayed nonlinear operator. It records the finite graph window, accepted input names, output names, and the local handoff ledger from weak input tests to strong output estimates.*
6. *S is the demicontinuity stability row. Each entry records a weak-input budget, the finite scalar tests that certify it, the strong-output radius requested by M, and the displayed comparison sending the operator image into that radius.*
7. *H records componentwise* hsame *transport for Banach source, weak-input, metric target, real scalar seal, action, stability, replay, provenance, and naming rows.*

8. *C records displayed* Cont *replay from a weak-input request through*

$$B, W, M, R, A, S \longmapsto H, C, P, N.$$

9. *P is the* Pkg *provenance over* Demicontinuity$^{\uparrow}$, BanachSpace$^{\uparrow}$, WeakConvergenceHilbert$^{\uparrow}$, MintyBrowderMonotone Real$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{Demicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, W, M, R, A, S, H, C, P, N$. It has no coordinate for compactness, selected weakly convergent subsequences, maximal-monotone closure, quotient graph equality, input norm convergence, or a host theorem that replaces the finite handoff ledger.

Lean4 (statement) `BEDC.Derived.DemicontinuityUp`

Theorem 7985.2 (Demicontinuity NameCert obligations)**.** *For every accepted* Demicontinuity$^{\uparrow}$ *carrier $D = (B, W, M, R, A, S, H, C, P, N)$, the local* NameCert$_{\text{Demicontinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach source row, weak-input row, strong-output metric row, real scalar seal, displayed nonlinear action row, demicontinuity stability row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Proof. Project the carrier of Definition 7985.1. The analytic rows are exactly B, W, M, R, A, S: source admission, weak-input observation, strong-output metric readback, real scalar sealing, the nonlinear action window, and the handoff stability ledger.

The handoff is finite. A consumer first reads the weak-input budget and scalar tests from W, R, then reads the requested strong-output radius from M, and finally checks the displayed comparison stored in S for the action row A.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since compactness, selected subsequences, maximality, quotient graph equality, input norm convergence, and ambient weak-to-strong theorems are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 7985.3 (Demicontinuity weak-to-strong handoff)**.** *Every* BanachSpace$^{\uparrow}$, WeakConvergenceHilbert$^{\uparrow}$, MintyBrowderMonotoneOperator$^{\uparrow}$, Real$^{\uparrow}$, *or* Met$^{\uparrow}$ *consumer of an accepted* Demicontinuity$^{\uparrow}$ *packet reads the demicontinuity witness only through the finite route*

$$\mathsf{BanachSpace}^{\uparrow}, \mathsf{WeakConvergenceHilbert}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto A, S \longmapsto \mathsf{Met}^{\uparrow}$$

together with H, C, P, N. The route exports weak-input tests and strong-output metric control for the displayed action, not compactness, a selected weak limit, maximal-monotone closure, input norm convergence, or an ambient strong-convergence theorem.

Proof. Start with the source and weak-input rows. The Banach row B provides the accepted domain carrier, while W and R provide only the displayed scalar tests and real tolerance values used to recognize a weak-input request.

Read the action and stability rows next. The action row A supplies the finite operator window, and S records the comparison from the accepted weak-input budget to the requested strong-output metric radius.

Finally pass through M, H, C, P, N. The metric row is the only strong-output interface, while transport, replay, provenance, and local naming preserve the same finite handoff without adding compactness, selected limits, maximality, quotient graph equality, input norm convergence, or a host convergence theorem. □

Falsifiable prediction: An accepted $\mathsf{Demicontinuity}^{\uparrow}$ read is refuted if a nonlinear-analysis consumer exports a strong-output conclusion without displaying the $\mathsf{BanachSpace}^{\uparrow}$ source row, weak-input tests through $\mathsf{WeakConvergenceHilbert}^{\uparrow}$ or sibling Banach weak rows, the $\mathsf{Real}^{\uparrow}$ scalar seal, the $\mathsf{Met}^{\uparrow}$ strong-output radius, the action row, the stability ledger, transport, replay, provenance, and local naming.

Independence witness: banach-space, weak-convergence-hilbert, minty-browder-monotone-operator, real, metric-space: none alone is bijective with $\mathsf{Demicontinuity}^{\uparrow}$ because the packet jointly carries a Banach source, weak-input observations, real scalar budgets, an action row, strong metric output control, a finite stability ledger, transport, replay, provenance, and local NameCert while refusing compactness, selected weak limits, maximality, and input norm convergence.

Closure status for $\mathsf{NameCert}_{\mathsf{Demicontinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{Demicontinuity}^{\uparrow}$ packet is generated from a Banach source row, weak-input observation rows, Real scalar seals, a strong-output MetricSpace row, a nonlinear action row, a demicontinuity stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Demicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7985.1, Definition 7985.2, and Definition 7985.3 seed the finite weak-input and strong-output handoff certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Demicontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, compactness, selected weak subsequences, maximal monotonicity, quotient graph equality, input norm convergence, or ambient weak-to-strong convergence.

Upgrade path: Obligation closure requires checked carrier admission, weak-input stability, strong-output metric exactness, action ledger transport, replay, provenance, and local naming for BEDC.Derived.DemicontinuityUp.

Chapter 7986

A Concrete Naming Certificate for $\mathsf{CauchyConvergenceSpace}^{\uparrow}$

Following Beattie and Butzmann's convergence-space treatment, the $\mathsf{CauchyConvergenceSpace}^{\uparrow}$ packet records the finite naming surface by which Cauchy filter data, regular rational tails, and real-name readback travel toward completion-facing consumers. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-existence route of chapter 5721. The packet names convergence as displayed BEDC data; it does not identify quotient limits, choose subsequences, or import open-cover compactness.

Definition 7986.1 (Cauchy convergence space carrier)**.** *A* $\mathsf{CauchyConvergenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, B, T, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the certified-points row, recording the finite point names and their local classifier data.*
2. *B is the Cauchy-filter basis row, recording the finite basis requests available to convergence consumers.*
3. *T is the regular tail and modulus row, recording* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback bounds for the displayed basis requests.*
4. *L is the candidate limit real-name row, recording the* $\mathsf{Real}^{\uparrow}$ *seal reached only after B and T have been displayed.*
5. *H records componentwise* hsame *transport for certified points, basis requests, regular tail data, candidate limit names, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from point rows and Cauchy-filter basis rows through regular tail readback to the candidate* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyConvergenceSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletionExistence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient limit classes, choice-selected subsequences, host equality of limits, open-cover compactness, or ambient completeness.

Theorem 7986.2 (Cauchy convergence space NameCert obligations)**.** *For every accepted* CauchyConvergenceSpace *carrier* $G = (X, B, T, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability for certified points and basis requests, Cauchy-basis transport, regular-tail compatibility, completion readback through the candidate* $\mathsf{Real}^{\uparrow}$ *seal, ledger exactness through* hsame *and* Cont*, package provenance, and local naming. The certificate refuses quotient limit classes, choice-selected subsequences, host equality of limits, open-cover compactness, and ambient completeness.*

Proof. Project the carrier of Definition 7986.1. The rows display certified points, a Cauchy-filter basis, regular tail and modulus data, a candidate real-name limit, transport, replay, provenance, and local naming.

Read the convergence rows in order. The basis row B is attached to the certified point row X, the regular tail row T supplies the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and the candidate row L is available only after those rows have been inspected.

The structural rows preserve the same route. Componentwise hsame transport and displayed Cont replay keep the basis, tail, and candidate-limit rows on the same packet, while P and N source and name the packet.

Since the carrier contains no quotient-limit coordinate, selected-subsequence coordinate, host-equality oracle, open-cover compactness row, or ambient-completeness theorem, the displayed rows exhaust the local NameCert surface. □

Theorem 7986.3 (Cauchy convergence space real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or* MetricCompletionExistence *facing consumer of an accepted* $\mathsf{CauchyConvergenceSpace}^{\uparrow}$ *carrier receives only finite regular-Cauchy convergence data: certified points, Cauchy-filter basis rows, regular tail and modulus readbacks, the candidate real-name seal, componentwise transport, displayed replay, package provenance, and local naming. The handoff exports no quotient limit, choice-selected subsequence, host equality of limits, open-cover compactness claim, or ambient completeness claim.*

Proof. Start with $G = (X, B, T, L, H, C, P, N)$. The only route from the Cauchy-basis data to the candidate real name passes through the finite basis row B and the regular tail row T before L can be read.

Apply Definition 7986.2. It identifies the carrier admission, classifier stability, Cauchy-basis transport, regular-tail compatibility, completion readback, and ledger exactness rows that a consumer must preserve.

The downstream handoff is therefore finite. A $\mathsf{Real}^{\uparrow}$ consumer reads the candidate seal only after regular Cauchy data have been displayed, and a $\mathsf{MetricCompletionExistence}^{\uparrow}$ consumer receives the same displayed route as completion-facing evidence.

The excluded objects have no carrier coordinate. Thus the handoff cannot supply quotient limits, selected subsequences, host equality of limits, open-cover compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyConvergenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyConvergenceSpace}^{\uparrow}$ packet is generated from certified point rows, a Cauchy-filter basis, regular tail and modulus readback, a candidate Real name, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyConvergenceSpace}^{\uparrow}}$, seed)

Scope closed: Definition 7986.1, Definition 7986.2, and Definition 7986.3 seed the finite Cauchy-convergence handoff from filter bases and regular Cauchy readback to Real and metric-completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyConvergenceSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient limits, choice-selected subsequences, host equality of limits, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, Cauchy-basis transport, regular-tail compatibility, completion readback, ledger exactness, provenance, and local naming for BEDC.Derived.CauchyConvergenceSpaceUp

Chapter 7987

A Concrete Naming Certificate for $\mathsf{BoundedMonotoneSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ packet names the finite data by which a monotone real sequence and its displayed bounds are read by Real-and-Completion consumers. It sits after the finite stream interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the apartness-facing real comparison route of chapter 304. The packet records bounded monotonicity as a NameCert surface, not as a completeness theorem or a selected limit.

Definition 7987.1 (Bounded monotone sequence carrier)**.** *A* $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, R, M, U, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the finite observation schedule for the sequence.*
2. *W is the family of finite stream windows inspected by Real and Completion consumers.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison-tail ledger attached to those windows.*
4. *M is the monotone-step row, recording the displayed adjacent or tail comparison used by the sequence direction.*
5. *U is the upper-bound row, read through the same regular-rational tail ledger.*
6. *L is the lower-bound row, retained for apartness and interval-facing consumers even when the monotone direction only needs one bound.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the sequence values and bound endpoints.*
8. *H records componentwise* hsame *transport for the stream windows, regular tails, monotone rows, bound rows, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from stream-window reads through regular-rational comparison tails and then to the Real seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedMonotoneSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected supremum, a convergent subsequence, Bolzano–Weierstrass, quotient real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 7987.2 (Bounded monotone sequence NameCert obligations)**.** *For every accepted* BoundedMonotoneSequ *carrier* $B = (S, W, R, M, U, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison-tail readback, monotone-step exposure, upper-bound and lower-bound row exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7987.1. The sequence is visible only through the stream row S and the finite windows W, so no consumer can inspect an unbounded host sequence directly.

The comparison data are the regular-rational tail row R together with the monotone-step row M and the bound rows U, L. These rows expose bounded monotonicity before any Real endpoint is read.

The Real seal row E is downstream from the displayed stream and regular-tail data. The structural rows H, C, P, N transport, replay, source, and name the same route, so the listed rows exhaust the local certificate obligations. □

Theorem 7987.3 (Bounded monotone sequence comparison-tail route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ *packet reads the route*

$$S, W \longmapsto R \longmapsto M, U, L \longmapsto E$$

together with H, C, P, N*. The route may expose bounded monotone comparison data to completion-facing consumers, but it does not export a limit, a least upper bound, a convergent subsequence, or a Bolzano–Weierstrass conclusion.*

Proof. Begin with the finite-window stage. The only sequence observations are the windows W named by the $\mathsf{StreamName}^{\uparrow}$ row S.

Next read the $\mathsf{RegSeqRat}^{\uparrow}$ comparison tails. The monotone row M and the bound rows U, L are meaningful only after those regular-rational tails have been displayed.

Finally the $\mathsf{Real}^{\uparrow}$ seal row E records the sequence values and bound endpoints reached by the same tail route. Since no carrier coordinate names a limit, supremum, subsequence selection, or compactness principle, the consumer obtains only the bounded-monotone packet surface. □

Falsifiable prediction: An accepted $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ read is refuted if the monotone row M is preserved while the StreamName windows W or RegSeqRat comparison tails R are removed.

Independence witness: streamname, regseqrat, real, apartnessreal: the stream packet supplies finite windows without monotone and bound rows; the regular rational route supplies tail readback without the Real seal; the Real packet supplies endpoint seals without sequence monotonicity; and apartness-facing comparison packets do not jointly carry bounded monotone sequence windows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat comparison tails, monotone-step rows, upper and lower bound rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7987.1, Definition 7987.2, and Definition 7987.3 seed the bounded monotone sequence surface over StreamName, RegSeqRat, RealUp, apartness-facing comparison, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedMonotoneSequence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, selected limits, least upper bounds, convergent subsequences, Bolzano–Weierstrass, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, comparison-tail read-back, monotone-step exposure, bound-row exposure, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMonotoneSequenceUp.

Chapter 7988

A Concrete Naming Certificate for $\mathsf{CollectionwiseNormal}^{\uparrow}$

Following Engelking and Kelley on collectionwise normal spaces, the $\mathsf{CollectionwiseNormal}^{\uparrow}$ packet records the finite BEDC surface in which a displayed discrete family of closed rows receives pairwise disjoint open neighborhoods and a locally finite refinement ledger. It sits above the closed-pair separation route of chapter 1866, beside the locally finite cover discipline of chapter 1963, and before separation-function consumers such as **??**. The packet strengthens the normal-space and paracompact lattice by naming the finite familywise separation route; it does not import arbitrary choice of neighborhoods, host powersets, open-cover compactness, partition-of-unity weights, or an ambient point-set theorem.

Definition 7988.1 (Collectionwise normal carrier)**.** *A* $\mathsf{CollectionwiseNormal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, N, D, O, L, P, U, H, C, Z)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row on which the closed-family, open-neighborhood, and cover-refinement observations are read.*
2. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *separation row, exposing the closed-pair route used for each displayed member of the family.*
3. *D is the discrete closed-family ledger, recording the finite family window and the local discreteness observations visible to the packet.*
4. *O is the disjoint open-neighborhood ledger, assigning to every displayed closed row in D an open row over T and recording pairwise disjointness inside the finite window.*
5. *L is the locally finite refinement ledger, read through the paracompact cover surface of chapter 1963.*
6. *P is the paracompact handoff row that keeps locally finite refinement data visible without selecting a subcover from an arbitrary host family.*
7. *U is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *handoff row, available only after the displayed normal-space and disjoint-neighborhood rows have been read.*
8. *H records componentwise* hsame *transport for the topology, normal-space, discrete-family, disjoint-neighborhood, locally finite refinement, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the discrete closed-family request through the normal-space route, disjoint open-neighborhood ledger, locally finite refinement ledger, paracompact handoff, and Urysohn handoff.*

10. *Z is the local* $\mathsf{NameCert}_{\mathsf{CollectionwiseNormal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, N, D, O, L, P, U, H, C, Z$. It has no coordinate for host-indexed families, selected complements, arbitrary choice of separating neighborhoods, open-cover compactness, quotient topology equality, or partition-of-unity data.

Lean4 (statement) `BEDC.Derived.CollectionwiseNormalUp`

Theorem 7988.2 (Collectionwise normal NameCert obligations)**.** *For every accepted* $\mathsf{CollectionwiseNormal}^{\uparrow}$ *carrier $Q = (T, N, D, O, L, P, U, H, C, Z)$, the local* $\mathsf{NameCert}_{\mathsf{CollectionwiseNormal}^{\uparrow}}$ *surface consists exactly of topology-source admission, normal-space separation exposure, discrete closed-family exposure, disjoint open-neighborhood exactness, locally finite refinement exposure, paracompact handoff, Urysohn handoff, componentwise* hsame *transport, displayed* Cont *replay, and local naming. No obligation is discharged by a selected family of host neighborhoods, arbitrary open-cover compactness, a host powerset, quotient topology equality, or partition-of-unity weights.*

Proof. Project the carrier of Definition 7988.1. The rows $T, N, D, O, L, P, U, H, C, Z$ are the whole local surface.

The familywise separation rows are N, D, O. The normal-space route supplies closed-pair separation data, the discrete-family row fixes the finite family window, and the open-neighborhood ledger records the disjoint neighborhoods visible to the packet.

The refinement and handoff rows are L, P, U. They route locally finite cover data through the paracompact sibling and expose the Urysohn-facing handoff only after the separation rows are visible.

Transport, replay, and local naming are exhausted by H, C, Z. Since none of the excluded host objects is a carrier coordinate, the listed rows are exactly the NameCert obligations. □

Theorem 7988.3 (Collectionwise normal paracompact handoff)**.** *Every locally finite refinement read exported by an accepted* $\mathsf{CollectionwiseNormal}^{\uparrow}$ *packet must pass through the displayed route*

$$\mathsf{NormalSpace}^{\uparrow},\ D,\ O \longmapsto \mathsf{Paracompact}^{\uparrow} \longmapsto \mathsf{UrysohnLemma}^{\uparrow}.$$

The route exposes the closed-family separation data, disjoint open-neighborhood rows, paracompact locally finite refinement ledger, and Urysohn-facing handoff as finite BEDC rows; it does not select arbitrary subcovers, choose host neighborhoods, or import open-cover compactness as a primitive.

Proof. Start with the normal-space and discrete-family rows. They determine which displayed closed rows are being separated and keep the family window visible before any refinement read occurs.

Read the disjoint open-neighborhood ledger next. It supplies only the finite neighborhood rows stored in the carrier and does not choose a host family of separators.

The locally finite refinement row then passes through the paracompact handoff. A separation-function consumer may continue to the Urysohn sibling only after the normal-space, disjoint-neighborhood, and refinement rows have been displayed.

The structural rows preserve this same order under transport and replay. Because no coordinate stores arbitrary subcovers, selected neighborhoods, or open-cover compactness, the handoff remains a finite BEDC dependency route. □

Falsifiable prediction: An accepted $\mathsf{CollectionwiseNormal}^{\uparrow}$ read is refuted if it exports collectionwise separation without displaying the TopologyUp source row, NormalSpaceUp separation row, discrete closed-family ledger, disjoint open-neighborhood ledger, locally finite refinement row, ParacompactUp handoff, UrysohnLemmaUp handoff, hsame transport, Cont replay, and local NameCert row.

Independence witness: normalspace, paracompact, urysohn-lemma: $\mathsf{CollectionwiseNormal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a discrete closed-family ledger, disjoint open-neighborhood ledger, locally finite refinement handoff, paracompact route, Urysohn route, transport, replay, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CollectionwiseNormal}^{\uparrow}}$.

Constructive story: A $\mathsf{CollectionwiseNormal}^{\uparrow}$ packet is generated from a TopologyUp source row, NormalSpaceUp separation row, discrete closed-family ledger, disjoint open-neighborhood ledger, locally finite refinement row, ParacompactUp handoff, UrysohnLemmaUp handoff, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CollectionwiseNormal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7988.1, Definition 7988.2, and Definition 7988.3 seed the finite collectionwise normal separation surface over normal-space, paracompact, and Urysohn rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CollectionwiseNormal}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CollectionwiseNormalUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary host-indexed collectionwise normality, selected complements, open-cover compactness, partition-of-unity data, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, discrete-family exposure, disjoint-neighborhood exactness, locally finite refinement handoff, paracompact route stability, Urysohn handoff discipline, transport, replay, provenance, and local naming for BEDC.Derived.CollectionwiseNormalUp.

Chapter 7989

A Concrete Naming Certificate for MetaCICParallelDiamondSelector$^{\uparrow}$

The MetaCICParallelDiamondSelector$^{\uparrow}$ packet realizes the closed parallel-diamond selector frontier named in chapter 9386 and section 9386.3 as finite MetaCIC NameCert data. It sits beside the confluence audit packet of chapter 735, the decidability boundary of chapter 749, the normalization frontier of chapter 892, the residual-diamond witness of chapter 7219, and the source confluence frontier of chapter 9386. The packet names a selector ledger for one closed parallel-diamond candidate and its residual-substitution compatibility boundary; it does not assert arbitrary closed confluence, unrestricted residual substitution, subject reduction, quotient syntax equality, or a hidden evaluator oracle.

Definition 7989.1 (MetaCIC parallel diamond selector carrier)**.** *A* MetaCICParallelDiamondSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, R, D, K, L, O, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-window row, recording the bounded closed MetaCIC source and type window under inspection.*
2. *R is the residual-pair frontier row, naming the two parallel residual branches whose diamond candidate is being selected.*
3. *D is the closed parallel-diamond candidate row, recording the displayed join candidate and its finite endpoint window.*
4. *K is the residual-substitution compatibility boundary row consumed before the candidate can be read as diamond-facing data.*
5. *L is the selector ledger, recording the finite rule that chooses between accepted candidate rows and retained obstruction rows inside the displayed window.*
6. *O is the obstruction-retention row for the surrounding confluence frontier.*
7. *G is the TasteGate route row, recording that the proposed Lean path requires both* Nontrivial *and* FieldFaithful *instances before the selector can be checked as a genuine carrier rather than a smoke packet.*

8. *H stores componentwise* hsame *transport for the typed window, residual frontier, candidate row, compatibility boundary, selector ledger, obstruction row, TasteGate route, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the route*

$$T, R, K \longmapsto D, L \longmapsto O, G.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}$*, MetaCIC confluence audit data, decidability-boundary data, normalization-frontier data, residual-diamond witness data,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, D, K, L, O, G, H, C, P, N$. It has no coordinate for full confluence, unrestricted residual substitution compatibility, global subject reduction, quotient syntax equality, hidden evaluator state, or host-level Church–Rosser.

Lean4 (statement) `BEDC.Derived.MetaCICParallelDiamondSelectorUp`

Theorem 7989.2 (MetaCIC parallel diamond selector obligations)**.** *For every accepted* MetaCICParallelDiamondSel *carrier $S = (T, R, D, K, L, O, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}}$ *surface consists exactly of typed-window exposure, residual-pair frontier exposure, closed parallel-diamond candidate exposure, residual-substitution compatibility boundary exposure, selector-ledger exactness, obstruction retention, TasteGate route retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The analytic rows are T, R, D, K, L, O, G, and the rows H, C, P, N transport, replay, source, and name those rows.

The selector is boundary-sensitive. The typed window and residual frontier determine the candidate range, while the compatibility row K must be displayed before the diamond candidate D can be consumed.

The ledger row L records the finite selector decision, and the obstruction row O keeps the surrounding frontier visible. The TasteGate row G prevents the paper carrier from being read as a checked Lean gate until the required Nontrivial and FieldFaithful instances exist.

No further row is present. Thus full confluence, unrestricted substitution compatibility, subject reduction, quotient syntax equality, hidden evaluator state, and host Church–Rosser remain outside the local certificate. □

Theorem 7989.3 (MetaCIC parallel diamond selector frontier route)**.** *Every accepted* MetaCICParallelDiamondSele *selector read routes its closed parallel-diamond candidate through*

$$T, R, K \longmapsto D, L \longmapsto O, G.$$

The selector can be consumed by the MetaCIC confluence frontier only as a finite candidate-and-obstruction packet requiring later Nontrivial *and* FieldFaithful *TasteGate instances; it is not a proof of arbitrary closed parallel diamond, unrestricted residual substitution compatibility, or closed Church–Rosser.*

Proof. Read the typed closed window T and residual-pair frontier R first. These rows determine the bounded source and the two residual branches under inspection.

Next read the compatibility boundary K. The diamond candidate row D is downstream from this boundary, so a consumer cannot treat the selector as evidence-free residual substitution.

The selector ledger L then chooses inside the displayed finite candidate range, while O retains the open confluence frontier and G records the Lean TasteGate obligations. Transport through H, replay through C, provenance through P, and naming through N preserve the same route. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}$ read exposes metacic-confluenceauditpacket, metacicdecidableboundary, metacicnormalizationfrontier, metacicparallel_diamond_fronti metacic-residual-diamond-witness, typed-window, residual-frontier, residual-substitution boundary, selector ledger, obstruction retention, TasteGate route, transport, replay, provenance, and local NameCert rows before any selector handoff is consumed.

Independence witness: metacicconfluenceauditpacket, metacicdecidableboundary, metacicnormalizationfrontier, metacicparallel_diamond_frontier, metacic-residual-diamond-witness: MetaCICParallelDiamond is not bijective to any listed sibling because it jointly carries a typed closed-window row, residual-pair frontier, closed parallel-diamond candidate, residual-substitution compatibility boundary, selector ledger, obstruction retention, TasteGate route, transport, replay, provenance, and local naming.

The proposed Lean path for $\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}$ requires Nontrivial and FieldFaithful TasteGate instances for the typed window, residual frontier, diamond candidate, residual-substitution boundary, selector ledger, and obstruction-retention rows before the selector can be checked as more than carrier syntax.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}$ packet is generated from a typed closed-window row, residual-pair frontier row, closed parallel-diamond candidate row, residual-substitution compatibility boundary row, selector ledger, obstruction-retention row, TasteGate route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7989.1, Definition 7989.2, and Definition 7989.3 seed one finite selector packet for the MetaCIC closed parallel-diamond frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondSelector}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MetaCICParallelDiamondSelectorUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary closed parallel diamond, unrestricted residual substitution compatibility, closed Church–Rosser, subject reduction, quotient syntax equality, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, residual-pair frontier exposure, diamond-candidate exposure, residual-substitution boundary exposure, selector-ledger exactness, obstruction retention, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICParallelDiamondSelectorUp.

Chapter 7990

A Concrete Naming Certificate for $\mathsf{BarRecursiveModulus}^{\uparrow}$

Following Bishop and Bridges on bars, spreads, and moduli in constructive analysis, the $\mathsf{BarRecursiveModulus}^{\uparrow}$ packet records the finite modulus row by which a bar-recursive inspection can be handed to the L10 Real and Completion route. It sits after the finite stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the bar-recursive source of chapter 5532, and the fan-theorem bar-modulus sibling of chapter 6151. The packet names a displayed modulus extracted from finite bar-recursive rows; it is not a quotient-stream construction, selected limit operator, countable-choice principle, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification, or bridge certificate.

Definition 7990.1 (Bar-recursive modulus carrier)**.** *A* $\mathsf{BarRecursiveModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, Q, R, E, H, C, P, N)$$

with the following rows:

1. *B is the bar-recursive source row, carrying the displayed barred-spread tree, accepted branch prefix, and recursion ledger read from chapter 5532.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records which stream prefixes are inspected before the modulus is read.*
3. *Q is the modulus row, storing a unary bound, the finite bar level that justifies it, and the local stopping evidence attached to the accepted branch prefix.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, repacking the bounded stream windows as regular rational finite data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the same bounded windows and regular rational witnesses.*
6. *H records componentwise* hsame *transport for the bar-recursive source, stream windows, modulus row, regular rational readback, real seal, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay through the rows B, S, Q, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BarRecursiveModulus}^{\uparrow}$, $\mathsf{BarRecursion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FanTheoremBarModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, S, Q, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, ambient real completeness, or bridge checking.

Theorem 7990.2 (Bar-recursive modulus NameCert obligations)**.** *For every accepted* $\mathsf{BarRecursiveModulus}^{\uparrow}$ *carrier*

$$M = (B, S, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}$ *surface consists exactly of bar-recursive source admission for B, finite stream-window admission for S, modulus-bound admission for Q, regular rational readback for R, terminal Real sealing for E, componentwise* hsame *transport for H, displayed replay for C, provenance for P, and local naming for N.*

Proof. Project the carrier of Definition 7990.1. The rows B, S, Q expose the bar-recursive source, the finite stream windows, and the bound justified by the displayed bar level.

The readback rows R and E consume only that bounded window data. The regular rational row repacks the inspected prefixes, and the Real row seals the same finite evidence.

The structural rows H, C, P, N transport, replay, source, and name the packet without adding a quotient stream, selected limit, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge certificate. Hence the listed rows exhaust the local NameCert surface. □

Theorem 7990.3 (Bar-recursive modulus RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{BarRecursiveModulus}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{BarRecursion}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{BarRecursiveModulus}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}$$

through B, S, Q, R, H, C, P, N before exporting a regular rational window. The handoff supplies a bounded prefix schedule and regular rational readback only; it does not identify quotient streams, choose a completed branch, or import countable choice.

Proof. Start with the bar-recursive source row B. It displays the barred-spread tree, accepted branch prefix, and recursion ledger before any modulus row is available.

Read the finite stream row S and modulus row Q. Together they bind a concrete prefix schedule to a unary bound justified by the displayed bar level.

Only after those rows are present may the handoff row R be read as $\mathsf{RegSeqRat}^{\uparrow}$ data. The remaining rows H, C, P, N preserve the same finite route, so the consumer receives a regular rational window rather than a selected limit or quotient stream. □

Theorem 7990.4 (Bar-recursive modulus Real-seal boundary)**.** *Every Real-facing consumer of an accepted* $\mathsf{BarRecursiveModulus}^{\uparrow}$ *packet reaches the terminal seal through*

$$\mathsf{BarRecursion}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with the modulus row Q binding the finite prefix bound before the $\mathsf{Real}^{\uparrow}$ *row E is read. The boundary exports only the displayed real-name seal for those finite windows, not ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient stream equality, countable choice, Lean verification, or bridge checking.*

Proof. The carrier exposes the source, window, and modulus rows first. The bar-recursive source B and stream schedule S determine the inspected prefixes, while Q records the bound justified by the bar level.

The readback row R is then the sole regular rational route attached to those prefixes. The terminal row E seals exactly that readback as $\mathsf{Real}^{\uparrow}$ data.

Since no carrier coordinate stores quotient equality, a selected limit, countable choice, ambient completeness, Lean verification, or bridge data, any Real-facing consumer remains inside the displayed finite boundary. □

Falsifiable prediction: Every $\mathsf{BarRecursiveModulus}^{\uparrow}$ read factors through a bar-recursive source row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a displayed modulus row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame, displayed Cont, Pkg, and a local $\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}$ row.

Independence witness: bar-recursion, streamname, regseqrat, real, fan-theorem-bar-modulus: $\mathsf{BarRecursiveModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds bar-recursive source data, finite stream windows, a modulus bound, regular rational readback, a real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BarRecursiveModulus}^{\uparrow}$ packet is generated from a bar-recursive source row, finite stream windows, a displayed modulus bound, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7990.1, Definition 7990.2, Definition 7990.3, and Definition 7990.4 seed the finite bar-recursive modulus route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BarRecursiveModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BarRecursiveModulus}^{\uparrow}$.

Upgrade path: Obligation closure requires checked bar-recursive source admission, finite stream-window admission, modulus-bound exactness, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BarRecursiveModulusUp.

Chapter 7991

A Concrete Naming Certificate for $\mathsf{PrincipalIdealDomain}^{\uparrow}$

Following Atiyah–Macdonald and Lang, the $\mathsf{PrincipalIdealDomain}^{\uparrow}$ packet records the traditional principal-ideal-domain interface as finite BEDC NameCert data over a displayed commutative ring carrier. It sits after the algebraic operation rows of chapter 43 and chapter 44, reads ideal membership through chapter 81, accepts Euclidean-domain evidence from chapter 7053 as one sufficient producer, and supplies the principal-ideal matrix surface consumed by chapter 2262. The packet names finite ideal-generator ledgers; it does not claim quotient ideals, selected generators by choice, unique-factorization classification, or a Euclidean algorithm.

Definition 7991.1 (Principal ideal domain carrier)**.** *A* $\mathsf{PrincipalIdealDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (R, C, I, G, E, H, K, Q, N)$$

with the following displayed rows:

1. *R is the underlying* $\mathsf{Ring}^{\uparrow}$ *row for addition, multiplication, zero, one, negation, and distributivity;*
2. *C is the* $\mathsf{CommRing}^{\uparrow}$ *row making multiplication commute on the same displayed carrier;*
3. *I is the finite* $\mathsf{Ideal}^{\uparrow}$ *membership and absorption surface for the ideal under inspection;*
4. *G is the principal-generator ledger, displaying a generator row and the two containment reads between the ideal row and the generated principal row;*
5. *E is the optional* $\mathsf{EuclideanDomain}^{\uparrow}$ *producer row when the packet is supplied through division-with-remainder evidence;*
6. *H records componentwise* hsame *transport for the ring, commutative-ring, ideal, generator, producer, provenance, and local naming rows;*
7. *K records displayed* Cont *routes for ideal membership, generator readback, containment in both directions, and downstream matrix consumers;*
8. *Q is the* Pkg *provenance over* $\mathsf{PrincipalIdealDomain}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{EuclideanDomain}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrincipalIdealDomain}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $R, C, I, G, E, H, K, Q, N$.

Theorem 7991.2 (Principal ideal domain NameCert obligations)**.** *Every accepted* $\mathsf{PrincipalIdealDomain}^{\uparrow}$ *packet exposes exactly the rows*

$$R, C, I, G, E, H, K, Q, N$$

of Definition 7991.1. The local $\mathsf{NameCert}_{\mathsf{PrincipalIdealDomain}^{\uparrow}}$ *surface consists of ring-source admission, commutative multiplication, ideal membership and absorption, a displayed principal-generator ledger, optional Euclidean-domain producer evidence, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It has no coordinate for quotient ideals, selected generators by choice, unique-factorization classification, or a Euclidean algorithm.*

Proof. Project the carrier of Definition 7991.1. The algebraic source is read through R and C, while ideal membership and absorption are read through I.

The generator ledger G supplies the only principal-ideal assertion available to the packet: the named generator row and the two displayed containment reads between the ideal and the principal row it generates.

The optional producer row E records that a $\mathsf{EuclideanDomain}^{\uparrow}$ packet can supply this surface, but it is not itself the Euclidean algorithm and does not add quotient ideal data. Transport through H, replay through K, provenance through Q, and local naming through N preserve the same finite row family, so the listed rows exhaust the NameCert obligations. □

Theorem 7991.3 (Principal ideal domain ideal-generator ledger)**.** *For every accepted* $\mathsf{PrincipalIdealDomain}^{\uparrow}$ *packet, an ideal-facing consumer may read a generator only by passing through*

$$\mathsf{Ring}^{\uparrow},\ \mathsf{CommRing}^{\uparrow},\ \mathsf{Ideal}^{\uparrow},\ G,\ H,\ K,\ Q,\ N.$$

The $\mathsf{SmithNormalForm}^{\uparrow}$ *consumer receives this ledger as a principal-ideal matrix input, not as a quotient module, selected basis, chosen generator function, unique-factorization classification, or Euclidean algorithm. If the generator row is supplied by a* $\mathsf{EuclideanDomain}^{\uparrow}$ *producer, that producer is recorded in* E *and remains a dependency row rather than an ambient division service.*

Proof. Start with the ideal row I and the generator ledger G in the carrier. The containment reads in G are the only route from ideal membership to a principal row.

Next use the structural rows. Componentwise hsame transport through H keeps the ring, commutative-ring, ideal, and generator endpoints aligned; Cont replay through K records the finite readback route; and Q, N provide provenance and local naming.

Finally read the Smith-normal-form edge through chapter 2262. A matrix consumer may use the principal-ideal ledger only after the displayed algebra and ideal rows have been exposed, so the route exports generator bookkeeping without adding quotient ideals, a selected generator function, unique-factorization classification, or Euclidean algorithm access. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrincipalIdealDomain}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrincipalIdealDomain}^{\uparrow}$ packet is generated from a ring row, commutative-ring row, ideal membership and absorption row, displayed generator ledger, optional Euclidean-domain producer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PrincipalIdealDomain}^{\uparrow}}$, seed)

Scope closed: Definition 7991.1, Definition 7991.2, and Definition 7991.3 name the finite carrier and ideal-generator surface for principal ideal domain consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PrincipalIdealDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient ideals, selected generators by choice, unique-factorization classification, Euclidean algorithm access, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, generator-ledger exactness, transport stability, route stability, and SmithNormalForm-facing nonescape for the same finite packet.

Chapter 7992

A Concrete Naming Certificate for $\mathsf{CauchyModulusSum}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusSum}^{\uparrow}$ packet records the finite data needed to add two regular Cauchy modulus presentations before any quotient sequence equality, selected representative, or ambient completeness principle is available. It sits over the two $\mathsf{CauchyModulus}^{\uparrow}$ source rows, the finite observation schedules of chapter 126, the dyadic tolerance ledgers of chapter 291, the regular rational readback route of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite addition of modulus-controlled regular data; it is not a construction of quotient reals, countable-choice representatives, or a host-level addition theorem.

Definition 7992.1 (Cauchy modulus sum carrier)**.** *A* $\mathsf{CauchyModulusSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M_0, M_1, W_0, W_1, D_0, D_1, D_+, R_0, R_1, R_+, E, H, C, P, N)$$

with the following displayed rows:

1. *M_0 and M_1 are the two* $\mathsf{CauchyModulus}^{\uparrow}$ *source rows whose regular Cauchy data are being added.*
2. *W_0 and W_1 are* $\mathsf{StreamName}^{\uparrow}$ *schedule rows selecting the finite source windows consumed at the requested dyadic tolerance.*
3. *D_0 and D_1 are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers for the two summands, while D_+ is the dyadic ledger for the output tolerance and the finite addition budget.*
4. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the two summands, and R_+ is the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the summed rational values.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source modulus rows, finite schedules, dyadic ledgers, and regular rational readbacks have been displayed.*
6. *H records componentwise* hsame *transport for the two modulus rows, schedule rows, dyadic ledgers, readback rows, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from M_0, M_1 through $W_0, W_1, D_0, D_1, D_+, R_0, R_1, R_+$ to the Real-facing seal E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSum}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M_0, M_1, W_0, W_1, D_0, D_1, D_+, R_0, R_1, R_+, E, H, C, P, N$. *It has no coordinate for quotient sequence equality, selected representatives, countable choice over moduli, host real equality, unbounded limit selection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7992.2 (Cauchy modulus sum NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSum}^{\uparrow}$ *carrier*

$$S = (M_0, M_1, W_0, W_1, D_0, D_1, D_+, R_0, R_1, R_+, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSum}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{CauchyModulus}^{\uparrow}$ *source rows, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting for the summands and sum,* $\mathsf{RegSeqRat}^{\uparrow}$ *source readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *summed readback, downstream* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal of quotient sequence equality, selected representatives, countable-choice modulus selection, host real equality, unbounded limit selection, and ambient completeness.*

Proof. Project the carrier of Definition 7992.1. The accepted packet displays exactly the two source modulus rows, the two finite schedule rows, the three dyadic ledgers, the two source readbacks, the summed readback, and the terminal Real seal.

The addition read is then forced through the displayed finite route. The requested tolerance is decoded by D_+, the summand budgets are read through D_0, D_1, and the inspected source values are exposed only through W_0, W_1, R_0, R_1 before the summed readback row R_+ is available.

Transport and replay preserve the same rows. The componentwise hsame row H moves the listed coordinates together, while C replays the continuation from the two modulus rows through the schedules, dyadic ledgers, and rational readbacks to the seal E.

Finally inspect P, N. Provenance and local naming record only the displayed carrier rows, so no public obligation can call on quotient sequence equality, selected representatives, countable choice, host real equality, unbounded limit selection, or ambient completeness. □

Falsifiable prediction: Deleting either source $\mathsf{CauchyModulus}^{\uparrow}$ row or the output dyadic ledger D_+ makes the summed $\mathsf{RegSeqRat}^{\uparrow}$ readback indistinguishable from an unbudgeted rational addition request; the accepted carrier must reject that packet because the finite tolerance route can no longer be reconstructed.

Independence witness: cauchymodulus, streamname, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{CauchyModulusSum}^{\uparrow}$ because this packet jointly carries two modulus source rows, two finite schedules, summand and sum dyadic ledgers, source readbacks, summed readback, terminal Real seal, transport, replay, provenance, and local naming for the same finite addition route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusSum}^{\uparrow}$ packet is generated from two CauchyModulus source rows, StreamName schedule rows, DyadicRatCore tolerance ledgers for the summands and sum, RegSeqRat source and sum readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusSum}^{\uparrow}}$, seed)

Scope closed: Definition 7992.1 and Definition 7992.2 seed the finite addition route for regular Cauchy modulus data through schedule, tolerance, readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient sequence equality, selected representatives, countable-choice modulus selection, host real equality, unbounded limit selection, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source modulus exposure, schedule stability, dyadic tolerance accounting, summed RegSeqRat readback, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSumUp.

Chapter 7993

A Concrete Naming Certificate for $\mathsf{CauchyExponential}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyExponential}^{\uparrow}$ packet records the classical exponential Cauchy series as a finite real-name constructor for the L10 Real and Completion route. It sits over the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the product-completion route of chapter 3797. The packet names a displayed exponential-series constructor; it does not import a host exponential function, quotient real equality, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 7993.1 (CauchyExponential carrier)**.** *A* $\mathsf{CauchyExponential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, W, A, F, D, Q, S, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *row whose exponential is being named through inspected finite windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting finitely many source and partial-sum windows.*
3. *A is the coefficient ledger for the factorial denominators of the exponential series, stored as rational and dyadic rows rather than as a host analytic function.*
4. *F is the finite partial-sum family for the Cauchy series $\sum_n X^n/n!$ over the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger bounding the tail and product-budget requests visible in the packet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the partial sums.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal produced from the regular readback.*
8. *M is the* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *handoff row used whenever a consumer asks for product compatibility of the series constructor.*
9. *H records componentwise* hsame *transport for source, windows, coefficients, partial sums, dyadic tolerance rows, regular readback, Real seal, product-completion handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay along the same finite route.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyExponential}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyProductCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyExponential}^{\uparrow}}$ *row.*

The carrier stores no host exponential operator, no quotient equality of real names, no chosen limit of an infinite series, no unrestricted product algebra, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 7993.2 (CauchyExponential NameCert obligations)**.** *For every accepted* $\mathsf{CauchyExponential}^{\uparrow}$ *carrier* $E = (X, W, A, F, D, Q, S, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyExponential}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, factorial coefficient accounting, finite partial-sum display, dyadic tail control, regular rational readback, terminal Real sealing, product-completion handoff, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 7993.1. The visible analytic data are the source row X, the window schedule W, the coefficient ledger A, the finite partial sums F, the dyadic tolerance row D, the regular readback Q, and the terminal seal S.

The coefficient and partial-sum rows are downstream of the finite windows. Thus a consumer first reads $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances before it can inspect any factorial denominator, power product, or partial-sum value.

The readback and seal rows keep the same route visible. The row Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the displayed partial sums, and S is the resulting $\mathsf{Real}^{\uparrow}$ seal; neither row selects a host limit or identifies quotient real names.

Finally read M, H, C, P, N. The product-completion handoff, transport, replay, provenance, and local name record only the displayed packet. Since no other coordinate exists, the NameCert surface cannot export a host exponential, chosen series limit, unrestricted product algebra, countable choice, or ambient Real completeness. □

Theorem 7993.3 (CauchyExponential Cauchy-product handoff)**.** *Every product-facing or completion-facing consumer of an accepted* $\mathsf{CauchyExponential}^{\uparrow}$ *packet must first read the source* $\mathsf{Real}^{\uparrow}$ *row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, factorial coefficient ledger, finite partial-sum family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *handoff row is available. The handoff exports no host exponential, quotient real equality, selected series limit, unrestricted product algebra, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a product-facing read of the carrier in Definition 7993.1. The row M is present only as a handoff from the displayed series constructor, so it cannot be consumed before the source, window, coefficient, partial-sum, tolerance, readback, and seal rows have been opened.

The finite series route is ordered by the windows and tolerance ledger. The rows W, A, F, D expose exactly the finite Cauchy-series data needed by the consumer, while Q and S route that data through regular rational readback and the terminal Real seal.

Now use the sibling product-completion boundary of chapter 3797. That boundary accepts a product read only after displayed stream windows, dyadic observations, regular readback, and Real sealing remain visible.

Composing the two routes gives the stated handoff. Since both routes refuse host analytic functions, quotient real equality, selected limits, unrestricted product algebra, countable choice, and ambient Real completeness, the $\mathsf{CauchyExponential}^{\uparrow}$ product-facing read exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyExponential}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyExponential}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7993.1, Definition 7993.2, and Definition 7993.3 seed the finite exponential Cauchy-series constructor over StreamName, RegSeqRat, DyadicRatCore, Real, and CauchyProductCompletion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyExponential}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host exponentials, quotient real equality, selected limits, unrestricted product algebra, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, factorial coefficient accounting, partial-sum display, dyadic tail control, regular readback, Real sealing, product-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyExponentialUp.

Constructive story: A $\mathsf{CauchyExponential}^{\uparrow}$ packet is generated from a source Real row, finite StreamName windows, factorial coefficient data, finite partial sums, a DyadicRatCore tail ledger, RegSeqRat readback, a Real seal, a CauchyProductCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 7994

A Concrete Naming Certificate for $\mathsf{UniformEquivalence}^{\uparrow}$

Following Bourbaki's uniform-space treatment and Bishop–Bridges constructive metric spaces, the $\mathsf{UniformEquivalence}^{\uparrow}$ packet records the finite bidirectional modulus data needed when a metric or completion-facing construction is invariant under two displayed uniformly continuous routes. It sits beside the metric carrier of chapter 4230, the modulus discipline of chapter 391, the completion handoff of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a route through displayed windows and tolerances rather than a host bijection, quotient equality, selected representative, countable choice, or ambient completeness principle.

Definition 7994.1 (UniformEquivalence carrier)**.** *A* $\mathsf{UniformEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, F, G, M, N, I, R, S, Q, H, C, P, L)$$

with the following displayed rows:

1. *X and Y are the source and target* $\mathsf{Met}^{\uparrow}$ *rows, including only the distance requests and finite tolerance ledgers visible to the certificate.*
2. *F and G are the forward and reverse* $\mathsf{UniformContinuity}^{\uparrow}$ *rows, each exposing a displayed source tolerance, target tolerance, and finite modulus response.*
3. *M and N are the forward and reverse modulus rows, so each direction keeps its own tolerance transformer before any consumer reads the route as an equivalence.*
4. *I is the inverse-route compatibility row, recording that the two directed reads compose back through displayed metric and modulus rows rather than through host point equality.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance-seal row, binding every completion-facing tolerance to displayed real-name data.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row and Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row, so observed distances and tolerances pass through inspected windows before reaching R.*
7. *H records componentwise* hsame *transport for $X, Y, F, G, M, N, I, R, S, Q$.*
8. *C records* Cont *replay from a source or target request through the matching modulus, inverse-route compatibility, real seal, finite window, rational readback, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{UniformEquivalence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}$ *row.*

The classifier reads only $X, Y, F, G, M, N, I, R, S, Q, H, C, P, L$. It has no coordinate for host bijection data, quotient completion equality, selected representatives, pointwise extensionality, countable choice, ambient completeness, or an uninspected real tolerance oracle.

Theorem 7994.2 (Uniform equivalence bidirectional modulus)**.** *For every accepted* $\mathsf{UniformEquivalence}^{\uparrow}$ *carrier $E = (X, Y, F, G, M, N, I, R, S, Q, H, C, P, L)$, any equivalence read must factor through both displayed modulus rows along the finite route*

$$X, Y \longmapsto F, G \longmapsto M, N \longmapsto I \longmapsto R, S, Q \longmapsto H, C, P, L.$$

In particular, a one-sided uniform-continuity certificate, a host bijection, or a quotient completion equality cannot supply the $\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}$ *obligations.*

Proof. Project the carrier rows. The metric sources are visible only as X and Y, and the directed uniform-continuity data are visible only as F and G.

The forward read must expose M, while the reverse read must expose N. If either row is absent, the consumer has only a one-sided modulus route and not a bidirectional equivalence packet.

After both modulus rows have been read, inverse-route compatibility is forced through I. The row names compatibility only through displayed metric and modulus data, so a host bijection or quotient completion equality cannot replace the finite route.

The tolerance and observation rows R, S, Q then bind real-name sealing, stream windows, and rational readback. Finally H, C, P, L transport, replay, source, and name the same displayed packet, completing the local certificate surface. □

Theorem 7994.3 (Uniform equivalence completion consumer)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformEquivalence}^{\uparrow}$ *packet must read the source and target* $\mathsf{Met}^{\uparrow}$ *rows, both* $\mathsf{UniformContinuity}^{\uparrow}$ *modulus rows, inverse-route compatibility, the* $\mathsf{Real}^{\uparrow}$ *tolerance seal, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}$ *row before treating a construction as invariant under bidirectional uniform modulus.*

Proof. Start from a completion-facing or real-facing consumer route. Its source and target data are the displayed metric rows X, Y, not an ambient completion object.

The route then reads F, G, M, N to obtain both directed modulus responses. The inverse-route row I ties those responses together before any completion or real-name tolerance can be consumed.

The real and finite-window rows R, S, Q provide the only displayed tolerance interface. Thus a completion consumer cannot appeal to selected limits, quotient representatives, or an uninspected real oracle.

The structural rows H, C, P, L finish the handoff by transport, replay, provenance, and local naming. Therefore invariance under the packet is exactly invariance under the displayed bidirectional uniform-modulus data. □

Falsifiable prediction: A claimed $\mathsf{UniformEquivalence}^{\uparrow}$ packet that lets a $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer bypass either modulus row M, N or the displayed R, S, Q tolerance-window route refutes the packet.

Independence witness: uniformcontinuity, metriccompletion, metricspace, real: none is bijective to $\mathsf{UniformEquivalence}^{\uparrow}$ because this packet jointly carries two metric rows, two directed uniformcontinuity rows, two modulus rows, inverse-route compatibility, real tolerance seals, finite stream and rational windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEquivalence}^{\uparrow}$ packet is generated from source and target MetricSpace rows, forward and reverse UniformContinuity modulus rows, inverse-route compatibility, RealUp tolerance seals, StreamName and RegSeqRat finite windows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7994.1, Definition 7994.2, and Definition 7994.3 seed the finite bidirectional uniform-modulus route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host bijections, quotient completion equality, selected representatives, pointwise extensionality, countable choice, ambient completeness, or uninspected real tolerance oracles.

Upgrade path: Obligation closure requires checked carrier admission, bidirectional modulus coverage, inverse-route compatibility, tolerance-window sealing, transport, replay, provenance, and local naming for BEDC.Derived.UniformEquivalenceUp.

Chapter 7995

A Concrete Naming Certificate for $\mathsf{LocalSelfAudit}^{\uparrow}$

The $\mathsf{LocalSelfAudit}^{\uparrow}$ packet realizes the local self-audit branch of Definition 9414.558 as finite self-proxy readback, boundary-recognition, gap-exposure, provenance, and NameCert rows. It sits beside the public/local reflective compilation packet of chapter 3963, the GroundCompiler event-flow audit of chapter 829, the MetaCIC closure trace of chapter 795, and the self-compilation boundary of chapter 9235. The packet is local and records-side: it does not create a public scientific certificate, a total theory of consciousness, unrestricted self-application, hidden prompt-state authority, or a proof that a current self-proxy is identical to a subject-substance.

Definition 7995.1 (LocalSelfAudit carrier)**.** *A* $\mathsf{LocalSelfAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, B, G, X, F, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the focused self-proxy row, naming the local endpoint supplied by a records-side self-compiler route.*
2. *R is the self-proxy readback row, recording which finite local reads may be replayed against the focused trace.*
3. *B is the boundary-recognition row, separating local readback from public compiler authority, hidden prompt state, and unrestricted reflective closure.*
4. *G is the gap-exposure row, listing absent coordinates for total consciousness theory, subject-substance identity, public proof of local identity, and local authority over public claims.*
5. *X is the refusal ledger that keeps the boundary and gap rows active under every accepted read.*
6. *F is the FieldFaithful row for S, R, B, G, X, requiring each displayed coordinate to remain separately projectable.*
7. *T is the Nontrivial row separating self-proxy, readback, boundary-recognition, gap-exposure, and refusal roles.*
8. *H records componentwise* hsame *transport for S, R, B, G, X, F, T.*
9. *C records displayed* Cont *replay through the self-proxy readback, boundary-recognition, gap-exposure, refusal, transport, and role-separation rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocalSelfAudit}^{\uparrow}$, $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalSelfAudit}^{\uparrow}}$ *row.*

The classifier reads only $S, R, B, G, X, F, T, H, C, P, N$. It has no coordinate for public compiler authority, hidden prompt state, subject-substance identity, unbounded self-reference, a total consciousness theory, or local authority over public scientific claims.

Theorem 7995.2 (Local self-audit NameCert obligations)**.** *For every accepted* $\mathsf{LocalSelfAudit}^{\uparrow}$ *carrier $A = (S, R, B, G, X, F, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocalSelfAudit}^{\uparrow}}$ *surface consists exactly of self-proxy readback, boundary-recognition, gap-exposure, refusal retention, FieldFaithful projection, Nontrivial role separation, transport, replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 7995.1. The self-proxy and readback rows S, R name only the local focused trace and do not export public compiler authority.

The boundary and gap rows B, G expose what the local read can and cannot support. The refusal ledger X keeps hidden prompt state, subject-substance identity, total consciousness theory, public proof of local identity, and local authority over public claims absent from the carrier.

The FieldFaithful and Nontrivial rows F, T keep the displayed roles separate. Self-proxy, readback, boundary-recognition, gap-exposure, and refusal are not aliases for one hidden coordinate.

Finally H, C, P, N transport, replay, source, and name the finite packet. Since the classifier has no coordinate outside the displayed rows, every accepted obligation is exhausted by the listed local certificate surface. □

Theorem 7995.3 (Local self-audit gap exposure)**.** *Every accepted* $\mathsf{LocalSelfAudit}^{\uparrow}$ *read that exposes a self-proxy readback must retain the boundary-recognition row B, the gap-exposure row G, and the refusal ledger X. Consequently the packet cannot be read as public compiler authority, hidden prompt-state access, subject-substance identity, unbounded self-reference, a total consciousness theory, or local authority over public scientific claims.*

Proof. Begin with an accepted local read through S, R. The read is local because S names a focused self-proxy endpoint and R records finite readback against that focused trace.

Boundary recognition is then read through B. This row separates local self-audit from public compiler authority and from unrestricted reflective closure before any replay occurs.

Gap exposure is read through G and retained by X. These rows list the absent authorities and prevent the local read from being repackaged as hidden prompt-state access, subject-substance identity, total consciousness theory, or public scientific certification.

Transport, replay, provenance, and naming through H, C, P, N preserve the same boundary. Thus every accepted read exposes the gap rather than closing it by an unlisted coordinate. □

Falsifiable prediction: A claimed $\mathsf{LocalSelfAudit}^{\uparrow}$ packet that exports public compiler authority, hidden prompt-state access, subject-substance identity, or a total consciousness theory from S, R without displaying $B, G, X, F, T, H, C, P, N$ refutes $\mathsf{LocalSelfAudit}^{\uparrow}$.

Independence witness: publiclocalreflectivecompilation, groundcompilereventflowaudit, metacicclosuretrace, selfcompilationboundary: none is bijective to $\mathsf{LocalSelfAudit}^{\uparrow}$ because this packet isolates self-proxy readback, boundary-recognition, gap-exposure, refusal retention, FieldFaithful projection, Nontrivial role separation, transport, replay, provenance, and local naming as one finite local-audit surface.

The Lean TasteGate surface for $\mathsf{LocalSelfAudit}^{\uparrow}$ must provide explicit Nontrivial $\mathsf{LocalSelfAudit}^{\uparrow}$ and FieldFaithful $\mathsf{LocalSelfAudit}^{\uparrow}$ instances for the displayed carrier. The intended endpoint is BEDC.Derived.LocalSelfAuditUp with the FieldFaithful instance preserving $S, R, B, G, X, F, T, H, C, P, N$ and the Nontrivial instance separating self-proxy, readback, boundary-recognition, gap-exposure, and refusal roles.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalSelfAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalSelfAudit}^{\uparrow}$ packet is generated from local self-proxy readback, boundary-recognition, gap-exposure, refusal retention, FieldFaithful projection, Nontrivial role separation, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalSelfAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7995.1, Definition 7995.2, and Definition 7995.3 seed the finite local self-audit route and its gap-exposure boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalSelfAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public compiler authority, hidden prompt-state access, subject-substance identity, unbounded self-reference, a total consciousness theory, public proof of local identity, or local authority over public scientific claims.

Upgrade path: Obligation closure requires checked carrier admission, self-proxy readback, boundary-recognition, gap-exposure, refusal retention, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.LocalSelfAuditUp.

Chapter 7996

A Concrete Naming Certificate for LebesgueCoveringDimensionInduction$^{\uparrow}$

Following Hurewicz–Wallman and Engelking on dimension theory, the LebesgueCoveringDimensionInduction$^{\uparrow}$ packet records the finite boundary-recursion route that compares Lebesgue covering dimension with the small and large inductive dimension surfaces. It sits over the topology source of chapter 88, the normal separation row of chapter 1866, the finite-cover refinement row of chapter 3430, the covering-dimension packet of chapter 1853, the small inductive dimension packet of chapter 7347, and the large inductive dimension packet of chapter 7371. The chapter names a displayed finite induction route; it does not import a host powerset of covers, arbitrary open-cover compactness, quotient equality of closed boundaries, or a global dimension theorem.

Definition 7996.1 (Lebesgue covering dimension induction carrier)**.** *A* LebesgueCoveringDimensionInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, C, S, I, N, F, B, H, R, P, Z)$$

with the following displayed rows:

1. *T is the* Topology$^{\uparrow}$ *source row on which open covers, closed boundaries, and refinement windows are read.*
2. *C is the* CoveringDimension$^{\uparrow}$ *row recording the displayed finite order bound for a finite cover window.*
3. *S is the* SmallInductiveDimension$^{\uparrow}$ *boundary-decomposition row for the inspected closed neighbourhood frontier.*
4. *I is the* LargeInductiveDimension$^{\uparrow}$ *boundary-refinement row used to compare the closed frontier with the surrounding cover window.*
5. *N is the* NormalSpace$^{\uparrow}$ *separation row that turns a displayed closed boundary request into a finite neighbourhood-separation ledger.*
6. *F is the* FiniteCoverRefinement$^{\uparrow}$ *row recording the finite refinement and order-control data consumed by the induction step.*
7. *B is the boundary-recursion ledger, storing the predecessor-dimension request, boundary carrier, and finite descent witness.*

8. *H stores componentwise* hsame *transport for the topology, cover, inductive-dimension, normal-space, refinement, boundary-recursion, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay of the route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{NormalSpace}^{\uparrow},\ \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto \mathsf{CoveringDimension}^{\uparrow},\ \mathsf{SmallInductiveDimension}^{\uparrow},\ \mathsf{LargeInd}$$

10. *P is the* Pkg *provenance over* $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{SmallInductiveDimension}^{\uparrow}$, $\mathsf{LargeInductiveDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *Z is the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, S, I, N, F, B, H, R, P, Z$. It has no coordinate for arbitrary host open covers, selected infinite refinements, quotient equality of boundary carriers, completed lattices of closed sets, or external dimension equivalence theorems.

Lean4 (definition) `BEDC.Derived.LebesgueCoveringDimensionInductionUp`

Theorem 7996.2 (Lebesgue covering dimension induction NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$ *carrier $L = (T, C, S, I, N, F, B, H, R, P, Z)$, the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}}$ *surface consists exactly of topology-source exposure, covering-dimension order-bound exposure, small-inductive boundary exposure, large-inductive boundary-refinement exposure, normal-space separation exposure, finite-cover-refinement exactness, boundary-recursion descent, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7996.1. The analytic rows are T, C, S, I, N, F, B, and the structural rows are H, R, P, Z.

The boundary recursion is not read from a global dimension theorem. The rows S and I expose the inductive boundary surfaces, N supplies the finite normal-separation ledger, and F records the finite refinement data needed for the covering-dimension comparison.

The replay row R orders these dependencies through the topology and normal-space sources before the boundary-recursion ledger is consumed. Transport, provenance, and local naming preserve the same displayed rows.

Since no excluded host object is a carrier coordinate, the listed rows are exactly the local NameCert obligations. □

Theorem 7996.3 (Lebesgue covering dimension induction boundary route)**.** *Every boundary-recursion read exported by an accepted* $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$ *packet must pass through*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{NormalSpace}^{\uparrow},\ \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto \mathsf{CoveringDimension}^{\uparrow},\ \mathsf{SmallInductiveDimension}^{\uparrow},\ \mathsf{LargeInducti}$$

The route names a finite comparison between cover-order control and inductive boundary descent; it is not a proof of arbitrary covering-dimension equivalence, not a quotient boundary theorem, and not a host open-cover compactness principle.

Proof. Begin with the topology row T and the normal-space row N. These rows determine the displayed open and closed boundary windows on which the dimension comparison is allowed to speak.

Read the finite-cover refinement row F next. It supplies only the finite refinement and order-control data stored in the packet, so the argument cannot choose an arbitrary refinement outside the ledger.

Now compare the three dimension rows. The covering-dimension row C supplies the order-bound surface, while the small and large inductive rows S, I supply boundary-decomposition and boundary-refinement surfaces for the recursion ledger B.

Finally transport and replay the route through H, R, P, Z. The exported object is the finite boundary-recursion packet, so it cannot be reread as host open-cover compactness, quotient equality of boundaries, or a global dimension-equivalence theorem. □

Falsifiable prediction: An accepted $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$ read is refuted if it exports a boundary-recursion claim without displaying the TopologyUp source, NormalSpaceUp separation row, FiniteCoverRefinementUp refinement row, CoveringDimensionUp order-bound row, SmallInductiveDimensionUp boundary-decomposition row, LargeInductiveDimensionUp boundary-refinement row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: covering-dimension, small-inductive-dimension, large-inductive-dimension, normalspace, finite-cover-refinement: none is bijective to $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$ because this packet jointly carries cover-order control, two inductive boundary surfaces, normal separation, finite refinement, boundary recursion, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}$ packet is generated from TopologyUp source rows, NormalSpaceUp separation rows, FiniteCoverRefinementUp rows, CoveringDimensionUp order-bound rows, SmallInductiveDimensionUp boundary rows, LargeInductiveDimensionUp boundary-refinement rows, a boundary-recursion ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7996.1, Definition 7996.2, and Definition 7996.3 seed the finite boundary-recursion comparison among covering and inductive dimension packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimensionInduction}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.LebesgueCoveringDimensionInductionUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary covering-dimension equivalence, host open-cover compactness, quotient equality of boundary carriers, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked row exposure for topology, normal separation, finite refinement, covering dimension, small and large inductive boundary routes, boundary recursion, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueCoveringDimensionInductionUp.

Chapter 7997

A Concrete Naming Certificate for $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$ packet records the finite modulus surface by which a located Cauchy completion read supplies explicit dyadic thresholds before a terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits between the located-completion route of chapter 5793, the displayed modulus schedule of chapter 7587, and the threshold-selection surface of chapter 1986. The packet is a finite NameCert object for located completion moduli; it is not quotient completion equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7997.1 (Located Cauchy completion modulus carrier)**.** *A* $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the located-completion source row, read through* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$*.*
2. *M is the Cauchy modulus selector row, read through the displayed* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *and* $\mathsf{ConstructiveRealModulus}^{\uparrow}$ *surfaces.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the requested located-completion precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at the threshold named by M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window and dyadic tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after Q, M, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the located source, modulus selector, dyadic tolerance, window, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

   $$\mathsf{RegularCauchyLocatedCompletion}^{\uparrow} \longmapsto \mathsf{CauchyModulusSelection}^{\uparrow} \longmapsto \mathsf{ConstructiveRealModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatC}$$

9. *P records* Pkg *provenance over* $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$, $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{CauchyModulusSelection}^{\uparrow}$, $\mathsf{ConstructiveRealModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, M, D, W, R, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, trichotomy, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 7997.2 (Located Cauchy completion modulus regularity)**.** *For every accepted* LocatedCauchyCompletionl *carrier*

$$L = (Q, M, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located-completion source, Cauchy modulus selector, dyadic tolerance row, finite StreamName window, RegSeqRat readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real-facing read of the packet factors through the finite route*

$$Q, M, D, W, R \longmapsto E, H, C, P, N.$$

In particular, a consumer cannot reach the $\mathsf{Real}^{\uparrow}$ *seal while bypassing the located source row, the modulus selector row, the dyadic tolerance row, the window row, or the regular rational readback row.*

Proof. Project the carrier of Definition 7997.1. The only analytic rows are the located source Q, modulus selector M, dyadic tolerance row D, finite window W, regular rational readback R, and Real seal E.

The regularity route is finite. The source row Q supplies the located-completion request, the selector row M answers it by a displayed threshold, and the dyadic row D records the requested tolerance. The StreamName row W and RegSeqRat row R then expose the inspected rational window before E is available.

The structural rows H, C, P, N transport, replay, source, and name only this route. Since quotient completion equality, selected representatives, countable choice, trichotomy, host real equality, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, none of those principles can be exported by the local certificate. □

Falsifiable prediction: An accepted LocatedCauchyCompletionModulusUp packet that reaches a terminal RealUp seal without displaying the located source, modulus selector, dyadic tolerance, StreamName window, and RegSeqRat readback rows refutes the carrier surface.

Independence witness: Nearest siblings: regular-cauchy-located-completion, constructive-real-modulus, and cauchy-modulus-selection. None is bijective with $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$ because this packet jointly carries the located source row, selector row, dyadic tolerance row, finite window, regular rational readback, Real seal, transport, replay, provenance, and local NameCert row for the same finite completion-modulus read.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}$ packet is generated from a located-completion source row, Cauchy modulus selector row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}}$, seed)

Scope closed: Definition 7997.1 and Definition 7997.2 seed the finite located-completion modulus route through dyadic tolerance, finite window, regular readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, trichotomy, host real equality, open-cover compactness, ambient RealUp completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-source exposure, modulus-selector exactness, dyadic tolerance compatibility, StreamName window readback, RegSeqRat regularity, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyCompletionModulusUp.

Chapter 7998

A Concrete Naming Certificate for $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$

The $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ packet realizes the boundary-tangent branch of chapter 9463 as finite observer-expansion data. It sits after the horocycle packet of chapter 3534, the Busemann boundary route of chapter 3732, the visual-boundary readout of chapter 4181, the boundary-shadow route of chapter 3813, and the hyperbolic metric packet of chapter 3474. The packet names a finite modulus budget for horocycle-shadow reads; it is not a global hyperbolic model, quotient ideal boundary, selected geodesic ray, ambient measure theorem, or physical observer dynamics.

Definition 7998.1 (Hyperbolic horocycle-shadow modulus carrier)**.** *A* $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, B, V, S, W, G, E, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *boundary-tangent horocycle row.*
2. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row that records the Busemann boundary read and horocycle level.*
3. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row used for the visible boundary face.*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row that records the finite shadow window.*
5. *W is the shadow-budget modulus row, comparing horocycle level, visual boundary window, and boundary-shadow tolerance.*
6. *G is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk and boundary data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed modulus comparison.*
8. *A is the TasteGate route row recording that the Lean path requires* Nontrivial *and* FieldFaithful *instances before the modulus carrier can be checked beyond encoded syntax.*
9. *H stores componentwise* hsame *transport for the horocycle, Busemann, visual-boundary, shadow, modulus, metric, Real-seal, TasteGate, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$\mathsf{HyperbolicHorocycle}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow},\ \mathsf{HyperbolicVisualBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicBoundaryShado}$$

11. *P is the* Pkg *provenance over* HyperbolicHorocycleShadowModulus$^{\uparrow}$, HyperbolicHorocycle$^{\uparrow}$, HyperbolicBusemannB HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{HyperbolicHorocycleShadowModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, V, S, W, G, E, A, H, C, P, N$. It has no coordinate for completed ideal-boundary quotients, selected rays, ambient visual measures, global curvature assumptions, host shadow sigma-algebras, or physical observer trajectories.

Lean4 (definition) `BEDC.Derived.HyperbolicHorocycleShadowModulusUp`

Theorem 7998.2 (Hyperbolic horocycle-shadow modulus NameCert obligations)**.** *For every accepted* HyperbolicHorocycleShadowModulus$^{\uparrow}$ *carrier $M = (Q, B, V, S, W, G, E, A, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicHorocycleShadowModulus}^{\uparrow}}$ *surface consists exactly of horocycle admission, Busemann-boundary readback, visual-boundary readback, boundary-shadow window exposure, shadow-budget modulus exactness, hyperbolic-metric comparison, terminal Real sealing, TasteGate route retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 7998.1. The geometric rows are Q, B, V, S, W, G, E, A, and the structural rows are H, C, P, N.

The modulus row is boundary-tangent. It can be read only after the horocycle row Q, Busemann boundary row B, visual-boundary row V, boundary-shadow row S, and metric row G have been displayed.

The Real seal E records the terminal comparison, while A keeps the Nontrivial and FieldFaithful TasteGate obligations visible. The route therefore remains encoded carrier data until those instances are supplied.

Transport, replay, provenance, and naming preserve exactly the displayed rows. Since no excluded host boundary, measure, ray-selection, or global-geometry object is present, the listed rows exhaust the local NameCert surface. □

Theorem 7998.3 (Hyperbolic horocycle-shadow modulus boundary route)**.** *Every boundary-tangent shadow-modulus read from an accepted* HyperbolicHorocycleShadowModulus$^{\uparrow}$ *packet must expose*

HyperbolicHorocycle$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, Hy

before the modulus ledger W is exported through A, H, C, P, N. The route is a finite observer-expansion packet; it does not assert a completed ideal boundary, selected geodesic ray, ambient visual measure, global hyperbolic model, or physical observer dynamics.

Proof. Start from the horocycle row Q. It fixes the boundary-tangent level at which the shadow comparison is being read.

Read the Busemann and visual-boundary rows B, V next. They expose the boundary face and level data without replacing the packet by a quotient ideal boundary.

Apply the boundary-shadow row S and the metric row G. These determine the finite shadow window and metric comparison consumed by the modulus row W, while the Real seal E records only the displayed terminal comparison.

Finally pass through the TasteGate route and structural rows A, H, C, P, N. Since the route explicitly waits for Nontrivial and FieldFaithful instances and stores no selected rays, global measures, or ambient model, the exported object is the finite horocycle-shadow modulus packet stated above. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ read must expose hyperbolic-horocycle, hyperbolic-boundary-shadow, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-metric, Real-seal, TasteGate route, transport, replay, provenance, and local NameCert rows before the shadow-modulus ledger is exported.

Independence witness: hyperbolic-horocycle, hyperbolic-boundary-shadow, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-metric: $\mathsf{HyperbolicHorocycle}^{\uparrow}$ lacks the shadow-modulus ledger, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ lacks the boundary-tangent horocycle row, $\mathsf{HyperbolicBusemannBound}$ lacks the visual shadow budget, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ lacks the Busemann-level modulus row, and $\mathsf{HyperbolicMetric}^{\uparrow}$ lacks the joint boundary-tangent shadow ledger. None is bijective to $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ under NameCert-row preservation.

The Lean follow-up for $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ requires Nontrivial and FieldFaithful TasteGate instances preserving $Q, B, V, S, W, G, E, A, H, C, P, N$ before a checked carrier-alignment theorem can witness the modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}$ packet is generated from HyperbolicHorocycleUp rows, HyperbolicBusemannBoundaryUp rows, HyperbolicVisualBoundaryUp rows, HyperbolicBoundaryShadowUp rows, a shadow-budget modulus row, HyperbolicMetricUp rows, a RealUp seal, a TasteGate route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7998.1, Definition 7998.2, and Definition 7998.3 seed the finite horocycle-shadow modulus packet for boundary-tangent observer-expansion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHorocycleShadowModulus}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.HyperbolicHorocycleShadowModulusUp`

Bridge status: none

Not claimed: This chapter does not close completed ideal boundaries, selected geodesic rays, ambient visual measures, global hyperbolic geometry, Nontrivial or FieldFaithful TasteGate instances, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked horocycle admission, Busemann readback, visual-boundary readback, boundary-shadow exposure, modulus exactness, metric comparison, Real sealing, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicHorocycleShadowModulusUp.

Chapter 7999

A Concrete Naming Certificate for $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ packet records dyadic midpoint-convexity rows over $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ names. It sits over the dyadic arithmetic core of chapter 291, the finite observation schedules of chapter 126, the regular rational readback rows of chapter 301, the terminal real-name seal of chapter 38, and the metric comparison discipline of chapter 57. The packet names the finite midpoint-convexity surface for regular Cauchy inputs; it does not import host subsets, quotient real equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 7999.1 (Regular Cauchy midpoint-convex carrier)**.** *A* $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, R_0, R_1, S_0, S_1, D, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-observation row naming the displayed real-valued map whose midpoint-convexity is inspected.*
2. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source-name rows for the endpoints.*
3. *S_0 and S_1 are the synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows exposing exactly the rational approximants requested by the dyadic midpoint calculation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger for the common denominator, dyadic radius, and midpoint tolerance budget.*
5. *M is the midpoint-readback row, recording the regular-Cauchy midpoint observation before any terminal real-name seal is visible.*
6. *Q is the midpoint-convexity comparison row, recording the displayed dyadic inequality between the midpoint value and the endpoint-value average.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $F, R_0, R_1, S_0, S_1, D, M, Q$ have been displayed.*
8. *H records componentwise* hsame *transport for the function row, endpoint names, stream windows, dyadic ledger, midpoint readback, comparison, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from every midpoint-convexity query through the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMidpointConvex}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host convex subset, quotient equality of real names, a selected limit, trichotomy, or ambient real completeness.

Theorem 7999.2 (Regular Cauchy midpoint-convex NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ *carrier*

$$K = (F, R_0, R_1, S_0, S_1, D, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMidpointConvex}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed function row, the two regular Cauchy endpoint rows, the synchronized stream-window rows, the dyadic tolerance ledger, the midpoint readback row, the midpoint-convexity comparison row, the terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. A consumer may inspect midpoint convexity only after the endpoint windows, dyadic midpoint budget, midpoint readback, and comparison row have all been displayed.*

Proof. Project the carrier of Definition 7999.1. The analytic rows are exactly $F, R_0, R_1, S_0, S_1, D, M, Q, E$, while H, C, P, N transport, replay, source, and name the same packet.

Read the route in dependency order. The endpoint names R_0, R_1 expose finite windows S_0, S_1, those windows feed the dyadic ledger D, and the midpoint readback M is available before the comparison row Q can state midpoint convexity.

Finally the terminal seal E is reached only after the finite rows have been displayed. Since the carrier contains no host subset, quotient real equality, selected limit, trichotomy branch, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 7999.3 (Regular Cauchy midpoint-convex dyadic scope)**.** *Every accepted* $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ *carrier routes a midpoint-convexity read through*

$$(R_0, R_1) \longmapsto (S_0, S_1) \longmapsto D \longmapsto M \longmapsto Q \longmapsto E.$$

Thus the public dyadic scope is the finite route from regular Cauchy endpoint rows through synchronized observation windows, dyadic midpoint accounting, midpoint readback, midpoint-convexity comparison, and the Real seal. It exports no host subset, quotient real equality, selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier rows. The two endpoint names are visible only as $\mathsf{RegSeqRat}^{\uparrow}$ rows, and each requested midpoint precision is read through the finite $\mathsf{StreamName}^{\uparrow}$ windows S_0, S_1.

The dyadic ledger D is the only arithmetic budget in the packet. It records the common denominator, radius, and tolerance rows needed by the midpoint readback M and the comparison row Q.

The Real seal E is terminal in this displayed route. Componentwise transport and replay keep the same row order, while provenance and local naming add no analytic coordinate. Therefore every midpoint-convexity read remains inside the stated dyadic scope. □

Theorem 7999.4 (Regular Cauchy midpoint-convex ledger nonescape)**.** *Every* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ *packet receives only the rows*

$$F, R_0, R_1, S_0, S_1, D, M, Q, E, H, C, P, N.$$

The ledger cannot be reinterpreted as a host convexity theorem, quotient equality of real names, selected limiting process, trichotomy decision, or ambient completeness principle.

Proof. Apply Definition 7999.2. It requires carrier admission and classifier comparison to pass through the displayed endpoint, window, dyadic, midpoint, comparison, and Real-seal rows.

Next use Definition 7999.3. Any Real-facing or metric-facing consumer sees the midpoint-convexity comparison only after the synchronized finite windows and dyadic budget have been read.

The remaining rows H, C, P, N preserve and name this packet. Because none of the carrier coordinates stores host subsets, quotient real equality, selected limits, trichotomy, or ambient completeness, the consumer route cannot leave the finite ledger. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ read is refuted if the comparison row Q is preserved while either endpoint window S_0, S_1 or the dyadic ledger D is removed.

Independence witness: dyadicratcore, streamname, regseqrat, real, metric: $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries function observations, two regular Cauchy endpoint rows, synchronized finite windows, dyadic midpoint accounting, midpoint readback, comparison, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMidpointConvex}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$ packet is generated from a function-observation row, two RegSeqRat endpoint rows, synchronized StreamName windows, a DyadicRatCore midpoint ledger, a midpoint readback row, a midpoint-convexity comparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMidpointConvex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7999.1, Definition 7999.2, Definition 7999.3, and Definition 7999.4 seed the finite midpoint-convexity route over DyadicRatCore, StreamName, RegSeqRat, Real, Metric, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMidpointConvex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex subsets, quotient real equality, selected limits, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic midpoint accounting, midpoint-convexity comparison exactness, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMidpointConvexUp.

Chapter 8000

A Concrete Naming Certificate for $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ packet records the finite regular-Cauchy route for a convex combination $qx+(1-q)y$ before any completed $\mathsf{Real}^{\uparrow}$ consumer is allowed to read the result. It sits over the regular rational sequence boundary of chapter 301, the real-name seal of chapter 38, the dyadic arithmetic core of chapter 291, the affine-combination route of chapter 1679, and the midpoint-convex route of chapter 7999. The packet gives later completion and real-algebra consumers one finite algebraic surface instead of forcing each consumer to rethread coefficient, scale, sum, regular readback, and seal ledgers separately.

Definition 8000.1 (Regular Cauchy convex combination carrier)**.** *A* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *Q is the coefficient row, recording a rational coefficient q together with the dyadic window in which the complementary coefficient $1-q$ is computed.*
2. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows.*
3. *W_X and W_Y are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules for the two source names, read at the same precision request and observation index.*
4. *D_Q and $D_{\bar{Q}}$ are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for q and $1-q$ at the common exponent used by the packet.*
5. *S_X is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row for the scaled source qx, reading Q, X, W_X, D_Q and its scale-error ledger.*
6. *S_Y is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row for the scaled source $(1-q)y$, reading the complementary coefficient row, $Y, W_Y, D_{\bar{Q}}$, and its scale-error ledger.*
7. *Σ is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row that adds the two scaled regular names.*
8. *E is the finite error-budget ledger combining the two scale ledgers, the sum tolerance, the complementary-coefficient tolerance, and the final output tolerance assigned to the convex readback.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose scheduled point row is supplied by Σ with regularity evidence supplied by E.*

10. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the coefficient, two sources, two synchronized schedules, two dyadic coefficient endpoints, two scale rows, sum row, finite ledger, and regular readback have been displayed.*

11. *H stores componentwise* hsame *transports for the coefficient, source rows, schedules, dyadic endpoint rows, scale rows, sum row, ledger, readback, Real seal, replay row, provenance row, and local naming row.*

12. *C stores displayed* Cont *replay routes for coefficient exposure, source-window exposure, the two scale reads, the sum read, the ledger read, the regular readback, and the Real seal handoff.*

13. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$, $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$, $\mathsf{RegularCauchyMidpointConvex}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyConvexCombination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R,$ The carrier has no coordinate for quotient stream equality, host real affine arithmetic, a selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completed-real affine operation.*

Theorem 8000.2 (Regular Cauchy convex combination NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ *carrier*

$$K = (Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyConvexCombination}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the coefficient row, two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *schedules, dyadic endpoint rows for q and $1-q$, two* $\mathsf{RegularCauchyScale}^{\uparrow}$ *rows, one* $\mathsf{RegularCauchySum}^{\uparrow}$ *row, the finite ledger, readback, Real seal, and structural rows;*

2. *coefficient complement exposure, requiring D_Q and $D_{\bar{Q}}$ to be read from the same finite dyadic coefficient window before either scale row is consumed;*

3. *synchronized-window exposure, requiring W_X and W_Y to be read at a common requested precision before the two scale rows are formed;*

4. *scale exactness for S_X and S_Y, requiring the displayed rows for qx and $(1-q)y$ to consume their coefficient endpoint, source window, scale endpoint, and scale-error ledger;*

5. *sum exactness for Σ, requiring the two scaled regular names to be added through the* $\mathsf{RegularCauchySum}^{\uparrow}$ *surface before the convex readback row is exposed;*

6. *ledger accounting, requiring E to combine the complementary-coefficient budget, the two scale-error budgets, the sum tolerance, and the final output tolerance before R or Z is read;*

7. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the displayed finite convex packet;*

8. *refusal of quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No convex-combination obligation is stored outside these finite rows.

Proof. Project the accepted carrier. The coefficient row, two source rows, synchronized schedules, two dyadic coefficient endpoints, two scale rows, the sum row, the ledger, readback, and Real seal are explicit coordinates of the packet.

Read the coefficient and source windows first. The complementary coefficient $1-q$ is displayed in the same dyadic endpoint discipline as q, while W_X and W_Y expose the two source observations at a common finite precision.

Apply the sibling scale and sum surfaces. The rows S_X and S_Y name the two scaled regular Cauchy names, and Σ names their pointwise sum before any regular readback or Real seal row is consumed.

Finally read the ledger and structural rows. The finite row E accounts for coefficient complement, scale, sum, and output tolerance budgets, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient equality, host affine arithmetic, selected limits, trichotomy, ambient completeness, Lean verification, or a bridge, none can become a local obligation. □

Theorem 8000.3 (Regular Cauchy convex combination affine route)**.** *Assume the coefficient endpoint rows of an accepted* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ *carrier display the nonnegative coefficient row for q and the complementary nonnegative coefficient row for $1-q$ inside the same dyadic window. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the convex packet factors through the synchronized route*

$$Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z.$$

This route is a finite convex-combination certificate. It exposes neither quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, nor bridge data.*

Proof. Start with the obligation surface of Definition 8000.2. It places the coefficient row, the two source rows, the synchronized StreamName windows, the dyadic endpoint rows for q and $1-q$, the two scale rows, the sum row, the ledger, readback, and Real seal in the displayed carrier order.

Read the coefficient-complement data next. The nonnegative row for q and the nonnegative complementary row for $1-q$ are both exposed through D_Q and $D_{\bar{Q}}$ inside the same dyadic window, so no consumer can split the convex read into two unrelated coefficient schedules.

Apply classifier stability from the affine sibling chapter 1679. The two source windows W_X, W_Y, the scale rows S_X, S_Y, and the sum row Σ transport componentwise along the same synchronized route.

Finally apply the ledger theorem. The row E combines the coefficient-complement budget, the two scale-error budgets, the sum tolerance, and the final output tolerance before R or Z is visible, so the consumer receives exactly the finite convex-combination certificate and no excluded quotient, host arithmetic, limit, trichotomy, completeness, or bridge data. □

Theorem 8000.4 (Regular Cauchy convex combination bound preservation)**.** *Let two accepted* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ *carriers have coefficient rows, source rows, stream schedules, dyadic endpoint rows, scale rows, sum rows, ledgers, readbacks, Real seals, package provenance, and local naming rows matched componentwise by their displayed* hsame *classifiers. Then every consumer route supplied by the obligation surface of Definition 8000.2 transports to the corresponding route of the second packet.*

Proof. Start with the componentwise classifier data in H. It transports the coefficient row, the complementary coefficient endpoint, the two regular source rows, and the synchronized schedules without changing the finite precision route.

Transport the scale and sum rows next. The classified scale rows still read qx and $(1-q)y$ through the corresponding coefficient endpoints and source windows, and the classified sum row still combines those two scaled names through the displayed $\mathsf{RegularCauchySum}^{\uparrow}$ surface.

The ledger route is preserved by the same componentwise transport. The row E remains the only place where complement, scale, sum, and output tolerances are combined before R and Z are exposed.

Replay through C and repack through P and N. Every consumer route in the obligation surface is one of these displayed projections, so classifier transport cannot add quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data. □

Theorem 8000.5 (Regular Cauchy convex combination ledger nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ *packet reads the convex result only after the finite ledger E has combined the coefficient-complement row, the two scale-error rows, the sum-tolerance row, and the final output tolerance. The packet has no alternate ledger for quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Read a consumer request through the displayed affine route. The request first reaches the coefficient row, the two source rows, the synchronized stream windows, and the dyadic endpoint rows for q and $1-q$.

The two scale rows are then consumed. Their finite scale ledgers are the only sources for the tolerances attached to qx and $(1-q)y$.

The sum row Σ and ledger E finish the regular handoff. The row E combines the two scale ledgers with the sum tolerance and final output tolerance before the readback row R or Real seal row Z can be read.

Thus every consumer-facing affine route passes through the finite ledger. Since the carrier inventory contains no second ledger or host-level affine object, no consumer can escape to quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyConvexCombination}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyConvexCombination}^{\uparrow}$ packet is generated from a coefficient row, two $\mathsf{RegSeqRat}^{\uparrow}$ sources, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint rows for q and $1-q$, two $\mathsf{RegularCauchyScale}^{\uparrow}$ rows, a $\mathsf{RegularCauchySum}^{\uparrow}$ row, a finite ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyConvexCombination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8000.1, Definition 8000.2, Definition 8000.3, and Definition 8000.4 seed the finite convex-combination surface over coefficient, two $\mathsf{RegSeqRat}^{\uparrow}$ sources, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, dyadic coefficient endpoints, $\mathsf{RegularCauchyScale}^{\uparrow}$ rows, a $\mathsf{RegularCauchySum}^{\uparrow}$ row, finite ledger data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyConvexCombination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No further paper-axis closure is claimed here; bridge and Lean verification remain separate axes.

Chapter 8001

A Concrete Naming Certificate for $\mathsf{NerveLemma}^{\uparrow}$

Following Borsuk and the algebraic-topology formulation in Hatcher, the $\mathsf{NerveLemma}^{\uparrow}$ packet records the finite good-cover route by which a compact metric cover, its finite nerve, a simplicial-complex carrier, and a homology consumer are read as one NameCert bridge. It sits after the compact source of chapter 129, the finite cover nerve of chapter 7597, the finite face inventory of chapter 238, the homology carrier of chapter 98, and the dimension-control route of chapter 1853. The packet names only displayed finite good-cover rows, nerve-face rows, chain-handoff rows, transport, replay, provenance, and local naming; it does not import open-cover choice, quotient homotopy classes, selected partitions of unity, or an ambient homotopy equivalence theorem.

Definition 8001.1 (Nerve lemma carrier)**.** *A* $\mathsf{NerveLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, G, N, S, H, D, T, R, C, P, Q)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, displaying the inspected compact metric carrier and the finite cover window used by the packet.*
2. *G is the good-cover ledger, recording for each displayed finite intersection the contractibility witness row accepted by the packet.*
3. *N is the* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *row, exposing cover membership, overlap incidence, and accepted finite nerve faces.*
4. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carrier read from the accepted nerve faces.*
5. *H is the* $\mathsf{Homology}^{\uparrow}$ *chain-and-cycle handoff row consumed after the simplicial face carrier is displayed.*
6. *D is the* $\mathsf{CoveringDimension}^{\uparrow}$ *dimension-control row, bounding the inspected finite intersection depth and face dimension read by the consumer.*
7. *T is the finite chain-transport row from cover-intersection data to nerve-chain data.*
8. *R records shortcut refusals for arbitrary open covers, selected subcovers, quotient homotopy classes, partitions of unity, and ambient host equality.*
9. *C records displayed* Cont *replay through good-cover admission, finite nerve formation, simplicial face inspection, homology handoff, and dimension control.*

10. *P is the* Pkg *provenance over* $\mathsf{NerveLemma}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{NerveLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, N, S, H, D, T, R, C, P, Q$. It has no coordinate for an unrestricted open cover, a selected refinement, a quotient homotopy type, a chosen partition of unity, or equality in an ambient topological host.

Theorem 8001.2 (Nerve lemma NameCert obligations)**.** *For every accepted* $\mathsf{NerveLemma}^{\uparrow}$ *carrier*

$$L = (K, G, N, S, H, D, T, R, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{NerveLemma}^{\uparrow}}$ *surface consists exactly of compact metric source admission through K, finite good-cover intersection exposure through G, finite nerve readback through N, simplicial face admission through S, homology handoff through H, covering-dimension control through D, finite chain transport through T, shortcut refusal through R, displayed replay through C, provenance through P, and local naming through Q.*

Proof. Carrier admission is projection from Definition 8001.1. The accepted packet displays the compact metric source, the finite good-cover ledger, the finite cover nerve, the simplicial face carrier, the homology handoff, the dimension-control row, finite transport, refusal, replay, provenance, and local naming rows.

The cover and nerve obligations are exposed through K, G, N, S. The packet first fixes the compact metric source and finite cover window, then records the good-cover intersection rows, then reads the finite nerve faces as a simplicial-complex carrier.

The consumer obligations are exposed through H, D, T. Homology is reached only after the finite face carrier is displayed, and the dimension-control row records the finite intersection depth inspected by the same packet.

The structural obligations are exposed through R, C, P, Q. These rows keep the route inside displayed finite data and block any rereading through arbitrary open covers, quotient homotopy classes, selected partitions of unity, or host equality. □

Theorem 8001.3 (Nerve lemma finite-cover homology handoff)**.** *Every* $\mathsf{Homology}^{\uparrow}$ *row exported by an accepted* $\mathsf{NerveLemma}^{\uparrow}$ *packet factors through the finite route*

$$K, G \longmapsto N, S \longmapsto H, D, T, R, C, P, Q.$$

Thus the homology-facing consumer first reads the compact metric source and good-cover ledger, then the finite cover nerve and simplicial face carrier, and only then receives the homology handoff with dimension control, finite transport, shortcut refusal, replay, provenance, and local naming. The route exports no open-cover choice principle, selected refinement theorem, quotient homotopy equality, partition-of-unity construction, or ambient homotopy-equivalence theorem.

Proof. Start with K, G. The compact metric row supplies the finite cover window, and the good-cover ledger records only the displayed finite intersections accepted by the packet.

Next read N, S. The finite cover nerve supplies cover membership, overlap incidence, and accepted faces; the simplicial-complex row turns those accepted faces into the finite face carrier consumed downstream.

Finally read H, D, T, R, C, P, Q. The homology handoff is downstream of the finite nerve and simplicial carrier, while dimension control, finite transport, refusal, replay, provenance, and local naming keep the consumer attached to the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{NerveLemma}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NerveLemma}^{\uparrow}}$, seed)

Scope closed: Definition 8001.1, Definition 8001.2, and Definition 8001.3 seed the finite good-cover nerve bridge from compact metric cover rows to homology handoff rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NerveLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted open-cover compactness, selected refinements, quotient homotopy equality, partitions of unity, or an ambient homotopy-equivalence theorem.

Upgrade path: Obligation closure requires checked carrier admission, good-cover intersection exposure, finite nerve exactness, simplicial face admission, homology handoff, dimension control, transport, replay, provenance, and local naming for BEDC.Derived.NerveLemmaUp.

Constructive story: A $\mathsf{NerveLemma}^{\uparrow}$ packet is generated from CompactMetric finite cover rows, good-cover intersection rows, FiniteCoverNerve rows, SimplicialComplex finite face rows, Homology handoff rows, CoveringDimension control rows, finite transport, shortcut refusal, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8002

A Concrete Naming Certificate for $\mathsf{LimsupLiminfGap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LimsupLiminfGap}^{\uparrow}$ packet records the finite oscillation-width surface of a bounded real sequence by reading its lower-tail endpoint through chapter 1553 and its upper-tail endpoint through chapter 7101. It gives Real-and-Completion consumers a named gap boundary for comparing displayed tail envelopes without importing host suprema, host infima, quotient streams, selected subsequences, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8002.1 (Limsup-liminf gap carrier)**.** *A* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, I, U, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row whose finite tail windows and bound ledger are inspected.*
2. *I is the* $\mathsf{Liminf}^{\uparrow}$ *endpoint row, read as the lower tail-envelope endpoint for the same bounded source.*
3. *U is the* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *endpoint row, read as the upper tail-envelope endpoint for the same bounded source.*
4. *D is the dyadic tolerance ledger, recording the rational comparison budgets used to compare U with I.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed rational windows and tolerance requests.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row naming the displayed gap request after the two endpoints and tolerance ledger have been replayed.*
7. *H records componentwise* hsame *transport for the bounded source, endpoint rows, dyadic ledger, rational readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the bounded source, lower endpoint, upper endpoint, tolerance ledger, readback row, real seal, transports, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{LimsupLiminfGap}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{Liminf}^{\uparrow}$, $\mathsf{BoundedRealLimsup}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LimsupLiminfGap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, I, U, D, Q, E, H, C, P, N$. It has no coordinate for ambient completeness, host supremum or infimum selection, quotient equality of real streams, selected oscillating subsequences, trichotomy, or a convergence theorem.

Lean4 ✓ `BEDC.Derived.LimsupLiminfGapUp.LimsupLiminfGapTasteGate single carrier alignment`

Theorem 8002.2 (Limsup-liminf gap NameCert obligations)**.** *For every accepted* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *carrier*

$$G = (S, I, U, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LimsupLiminfGap}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded source,* $\mathsf{Liminf}^{\uparrow}$ *lower endpoint,* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *upper endpoint, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; endpoint alignment requiring I and U to consume the same bounded source row S; tolerance exactness requiring every comparison in D to be read through the rational row Q before the seal E is exposed; and nonescape from the displayed rows.*

Lean4 ✓ `BEDC.Derived.LimsupLiminfGapUp.LimsupLiminfGapNameCertObligations`

Proof. Project the accepted carrier of Definition 8002.1. The analytic rows are exactly the bounded source S, lower endpoint I, upper endpoint U, dyadic ledger D, rational readback Q, and terminal real seal E.

Endpoint alignment is rowwise. The $\mathsf{Liminf}^{\uparrow}$ endpoint supplies lower-tail envelope data for S, while the $\mathsf{BoundedRealLimsup}^{\uparrow}$ endpoint supplies the upper-tail envelope data for the same bounded source. A consumer may compare the endpoints only after both endpoint rows have been displayed.

Tolerance exactness is supplied by D and Q. The dyadic ledger records finite comparison budgets, and the regular rational readback exposes the rational windows through which those budgets are checked before the terminal $\mathsf{Real}^{\uparrow}$ seal is reached.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores host completeness, a selected subsequence, quotient stream equality, trichotomy, or a convergence theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 8002.3 (Limsup-liminf gap endpoint route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto I, U \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

before reading the displayed oscillation-width surface.

Proof. Begin with the bounded source row S. It is the only source of sequence entries, finite windows, and bound data available to the packet.

The endpoint rows I and U are then read as lower and upper tail-envelope data for that same source. The packet therefore compares two displayed endpoint certificates, not two host-selected limits.

The dyadic ledger D and rational readback Q carry the finite comparison budget. Only after those rows are replayed can a consumer reach the terminal $\mathsf{Real}^{\uparrow}$ seal E and the structural rows H, C, P, N. □

Lean4 ✓ `BEDC.Derived.LimsupLiminfGapUp.LimsupLiminfGapEndpointRoute`

Falsifiable prediction: Every accepted $\mathsf{LimsupLiminfGap}^{\uparrow}$ Real-facing read factors through a bounded real sequence source, Liminf endpoint, bounded Real limsup endpoint, dyadic tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: liminf, bounded-real-limsup, boundedrealsequence, regseqrat, real: no listed sibling supplies the whole packet $G = (S, I, U, D, Q, E, H, C, P, N)$, because each sibling contributes only the lower endpoint, upper endpoint, bounded source, rational readback, or terminal real seal while this chapter binds them into one oscillation-width carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LimsupLiminfGap}^{\uparrow}}$.

Constructive story: A $\mathsf{LimsupLiminfGap}^{\uparrow}$ packet is generated from a bounded real sequence source, a Liminf endpoint row, a bounded Real limsup endpoint row, a dyadic tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transports, Cont replay routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LimsupLiminfGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8002.1, Definition 8002.2, and Definition 8002.3 seed the finite oscillation-width surface for bounded real sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LimsupLiminfGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, host supremum or infimum selection, quotient stream equality, selected subsequences, zero-gap convergence, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shared-source endpoint alignment, dyadic tolerance exactness, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.LimsupLiminfGapUp.

Chapter 8003

A Concrete Naming Certificate for $\mathsf{QuasiPolishSpace}^{\uparrow}$

Following de Brecht and the Brattka–Gherardi treatment of represented countably based spaces, the $\mathsf{QuasiPolishSpace}^{\uparrow}$ packet records the finite BEDC surface in which countable open information and sober point reconstruction are displayed without requiring a compatible complete metric. It sits between the complete-separable metric boundary of chapter 1490, the point-reconstruction route of chapter 6495, the point-free open route of chapter 5153, the indexed-open source of chapter 88, and the finite observation schedules of chapter 126. The packet supplies a non-metric countably based sober-space bridge; it does not import host powersets, choice-indexed generic points, quotient spatial reflection, or a selected Polish presentation.

Definition 8003.1 (Quasi-Polish space carrier)**.** *A* $\mathsf{QuasiPolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, B, S, L, P, W, H, C, G, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing indexed open observations and their finite refinement tests.*
2. *B is the countable-base ledger, recording finite windows of basic opens and their cover-refinement reads.*
3. *S is the* $\mathsf{SoberSpace}^{\uparrow}$ *handoff row, available only after the topology and base rows have been displayed.*
4. *L is the* $\mathsf{LocaleSpace}^{\uparrow}$ *comparison row, reading the same open data point-free before any point-facing reconstruction is consumed.*
5. *P is the optional* $\mathsf{PolishSpace}^{\uparrow}$ *comparison row, present only when a complete-separable metric presentation has already been displayed by a sibling packet.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for finite windows of basic-open and neighborhood reads.*
7. *H records componentwise* hsame *transport for the topology, base, sober, locale, Polish-comparison, stream, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from base-window requests through topology rows, locale comparison, sober reconstruction, and optional Polish comparison.*

9. *G is the* Pkg *provenance over* $\mathsf{QuasiPolishSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{SoberSpace}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiPolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, S, L, P, W, H, C, G, N$. It has no coordinate for host powersets, arbitrary open-set quantification, selected completely prime filters, quotient spatial reflection, choice-indexed generic points, or a forced compatible complete metric.

Lean4 ✓ `BEDC.Derived.QuasiPolishSpaceUp.TasteGate.QuasiPolishSpaceTasteGate single carrier alignment`

Theorem 8003.2 (Quasi-Polish space NameCert obligations)**.** *For every accepted* $\mathsf{QuasiPolishSpace}^{\uparrow}$ *carrier $Q = (T, B, S, L, P, W, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuasiPolishSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission, countable-base window exposure, locale comparison, sober reconstruction handoff, optional Polish comparison, finite stream scheduling, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every consumer must read these rows before treating the packet as a quasi-Polish space boundary.*

Proof. Project the carrier of Definition 8003.1. The rows $T, B, S, L, P, W, H, C, G, N$ are the only classifier-visible coordinates.

The open-information side is finite. The topology row T and base ledger B expose indexed opens through finite windows, while the stream row W records the schedule by which those windows are read.

The reconstruction side is routed. The locale row L compares the same open data point-free, the sober row S supplies the point-reconstruction handoff, and the Polish row P is only a comparison surface when a sibling complete-separable metric presentation is already displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, G, N. Since no row stores host powersets, selected generic points, quotient spatial reflection, or a forced metric presentation, the listed rows are exactly the NameCert obligations. □

Theorem 8003.3 (Quasi-Polish sober-Polish handoff)**.** *Every accepted* $\mathsf{QuasiPolishSpace}^{\uparrow}$ *consumer that compares sober reconstruction with Polish-style readback factors through*

$$\mathsf{Topology}^{\uparrow},\ B,\ \mathsf{LocaleSpace}^{\uparrow},\ \mathsf{SoberSpace}^{\uparrow},\ \mathsf{StreamName}^{\uparrow}$$

before it may inspect the optional $\mathsf{PolishSpace}^{\uparrow}$ *comparison row. The handoff lets a complete-separable metric sibling feed a quasi-Polish consumer, and lets a sober countably based route feed a non-metric consumer, but it does not identify the packet with a host powerset topology, a quotient spatial reflection, a selected generic point, or a chosen compatible complete metric.*

Proof. Start with the topology and base rows. They expose the countably based open information as finite indexed windows rather than as a host powerset.

Next read the locale and sober rows. The locale comparison keeps the frame-of-opens route visible, and the sober handoff reads point reconstruction only after those open rows are displayed.

The stream schedule preserves finiteness of the comparison. Any Polish row is downstream from these windows and appears only as a sibling comparison surface, so it cannot force a metric presentation for the whole packet.

Thus the quasi-Polish handoff separates the sober countably based route from the Polish complete-separable comparison while preserving both as displayed BEDC rows. □

Falsifiable prediction: An accepted $\mathsf{QuasiPolishSpace}^{\uparrow}$ read is refuted if it exports sober point reconstruction or Polish comparison without displaying the TopologyUp source row, countable-base ledger, LocaleSpaceUp comparison row, SoberSpaceUp handoff row, StreamNameUp schedule, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert row.

Independence witness: polishspace, sober-space, locale-space, topology, streamname: $\mathsf{QuasiPolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries countable-base windows, topology rows, locale comparison, sober reconstruction, optional Polish comparison, stream scheduling, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiPolishSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasiPolishSpace}^{\uparrow}$ packet is generated from a TopologyUp source row, countable-base ledger, SoberSpaceUp handoff, LocaleSpaceUp comparison, optional PolishSpaceUp comparison, StreamNameUp schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiPolishSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8003.1, Definition 8003.2, and Definition 8003.3 seed the finite non-metric countably based sober-space bridge over topology, locale, sober, Polish-comparison, and stream rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiPolishSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets, choice-indexed generic points, quotient spatial reflection, arbitrary locale spatiality, selected compatible complete metrics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topology admission, countable-base exposure, locale comparison, sober reconstruction handoff, optional Polish comparison discipline, stream scheduling, transport, replay, provenance, and local naming for BEDC.Derived.QuasiPolishSpaceUp.

Chapter 8004

A Concrete Naming Certificate for $\mathsf{EngelExpansionReal}^{\uparrow}$

Following Engel and the constructive real-number treatment in Bishop and Bridges, the $\mathsf{EngelExpansionReal}^{\uparrow}$ packet names the finite route by which an Engel denominator schedule becomes a displayed real-name surface. It sits after the Engel expansion carrier of chapter 2007, the rational row of chapter 37, the dyadic tail ledger of chapter 291, the finite stream window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records an Engel-generated real presentation as finite BEDC NameCert data; it does not identify quotient representatives, choose an infinite sum, invoke trichotomy, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8004.1 (EngelExpansionReal carrier)**.** *An* $\mathsf{EngelExpansionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, Q, D, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the source* $\mathsf{EngelExpansion}^{\uparrow}$ *packet, including its denominator schedule, rational partial sums, dyadic remainder bounds, stream windows, regular readback, and terminal Real seal rows.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *partial-sum row read from the Engel denominator schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-control row used to bound every unread Engel remainder visible to the packet.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the denominator prefixes, partial sums, and tail bounds under inspection.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row produced from Q, D, W.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row attached only after the regular rational route has been displayed.*
7. *H records componentwise* hsame *transport for the Engel source, rational partial sums, dyadic tail bounds, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-name request through*

$$E, Q, W \longmapsto D \longmapsto R \longmapsto L.$$

9. *P is the* Pkg *provenance over* EngelExpansion$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{EngelExpansionReal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient representatives of reals, a selected value of an infinite Engel sum, trichotomy, countable choice, ambient Real$^{\uparrow}$ *completeness, or bridge checking.*

Theorem 8004.2 (EngelExpansionReal NameCert obligations)**.** *For every accepted* EngelExpansionReal$^{\uparrow}$ *carrier $X = (E, Q, D, W, R, L, H, C, P, N)$, the local* NameCert$_{\text{EngelExpansionReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the Engel source row, rational partial-sum exposure, dyadic remainder control, finite stream-window selection, regular rational readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier $X = (E, Q, D, W, R, L, H, C, P, N)$. The projection exposes the Engel source packet, the rational partial-sum row, the dyadic tail-control row, the stream-window row, the regular readback row, and the real-name seal.

Use Definition 2007.2. The source row E already displays the Engel denominator schedule and its finite NameCert obligations, so the real-facing packet does not introduce a separate infinite series object.

Read Q, D, W before any completion-facing data. The rational partial sums are visible only through the finite windows, and every unread tail claim passes through the dyadic remainder-control row before R or L can be consumed.

Finally form the regular readback and the Real seal. The rows H, C, P, N transport, replay, source, and name exactly this route, so quotient representatives, trichotomy, selected sums, ambient completeness, and bridge checking remain outside the local obligation surface. □

Theorem 8004.3 (EngelExpansionReal regular-Cauchy route)**.** *Every accepted* EngelExpansionReal$^{\uparrow}$ *carrier exposes its real-name readback only through the finite route*

$$E, Q, W \longmapsto D \longmapsto R \longmapsto L$$

with structural support H, C, P, N. Thus every real-facing consumer first reads an EngelExpansion$^{\uparrow}$ *source,* Rat$^{\uparrow}$ *partial sums,* DyadicRatCore$^{\uparrow}$ *tail bounds,* StreamName$^{\uparrow}$ *windows, a* RegSeqRat$^{\uparrow}$ *regular readback, and a terminal* Real$^{\uparrow}$ *seal before any* Real$^{\uparrow}$ *observation is available.*

Proof. Start from a consumer of the Real seal L. By the carrier definition, L is not an independent completed object; it is downstream of the regular rational row R and the finite Engel data displayed by E, Q, D, W.

Apply the real readback boundary of Definition 2007.3. That theorem forces the Engel source to pass through rational partial sums, dyadic tail bounds, stream windows, regular readback, and the terminal Real seal before any real-facing interpretation is available.

Use the sibling L10 route through chapter 291, chapter 126, chapter 301, and chapter 38. These chapters supply the dyadic bound, finite-window, regular-sequence, and real-seal rows consumed by the route.

The structural rows H, C, P, N then transport, replay, source, and name the same finite passage. Since no carrier coordinate stores quotient representatives, selected real limits, trichotomy, ambient completeness, Lean verification, or a checked bridge, none of those claims is exported by the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{EngelExpansionReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EngelExpansionReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8004.1, Definition 8004.2, and Definition 8004.3 seed the Engel-generated real presentation through finite rational partial sums, dyadic tail bounds, stream windows, regular rational readback, and terminal Real seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EngelExpansionReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, trichotomy, selected real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, regular-Cauchy routing, and non-escape proofs for the finite Engel source, rational partial-sum, dyadic tail-bound, stream-window, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

Constructive story: An $\mathsf{EngelExpansionReal}^{\uparrow}$ packet is generated from an EngelExpansion source, RatUp partial sums, DyadicRatCore tail bounds, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 8005

A Concrete Naming Certificate for $\mathsf{ThreeDistanceTheorem}^{\uparrow}$

Following Steinhaus and the classical three-distance theorem, the $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ packet records the finite orbit-window certificate for rotations on the circle. It sits beside the circle-name surface $\mathsf{S}^{1\uparrow}$, the approximation route of chapter 1974, the distribution route of chapter 7520, the dyadic comparison core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite circle-rotation windows, adjacent gap ledgers, dyadic tolerance rows, StreamName and RegSeqRat readback, Real and $\mathsf{S}^{1\uparrow}$ sealing, transport, replay, provenance, and local naming.

Definition 8005.1 (Three-distance theorem carrier)**.** *A* $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, W, O, G, D, S, Q, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the rotation-angle source row, carried only through displayed rational and real-name observations.*
2. *W is the finite orbit-window row for the points $\{0, A, 2A, \ldots, nA\}$ read on the circle.*
3. *O is the cyclic ordering row for the inspected orbit window.*
4. *G is the adjacent gap ledger obtained from consecutive rows of O.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the displayed gaps with the accepted tolerance rows.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used to expose the finite real-name windows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the gap lengths and tolerance budgets.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the inspected gap lengths.*
9. *U is the* $\mathsf{S}^{1\uparrow}$ *seal stating that the cyclic order is read on the circle surface.*
10. *E is the local three-distance classifier row: the adjacent gap ledger has at most three displayed gap classes through D, Q, R, U.*
11. *H records componentwise* hsame *transport for the angle, window, cyclic order, gap ledger, dyadic tolerance, stream schedule, readback, seals, classifier, replay, provenance, and naming rows.*

12. *C records displayed* Cont *replay from any theorem-facing query back through the same finite rows.*

13. *P is the* Pkg *provenance over* $\mathsf{ThreeDistanceTheorem}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{KroneckerApproximation}^{\uparrow}$, $\mathsf{WeylEquidistribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{ThreeDistanceTheorem}^{\uparrow}}$ *row.*

The carrier contains no measure argument, quotient-circle equality oracle, selected limiting subsequence, ambient real completeness principle, or countable-choice route.

Theorem 8005.2 (Three-distance theorem NameCert obligations)**.** *For every accepted* $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ *carrier*

$$T = (A, W, O, G, D, S, Q, R, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ThreeDistanceTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the rotation-angle row, finite orbit-window row, cyclic ordering row, adjacent gap ledger, dyadic tolerance ledger, StreamName schedule, RegSeqRat readback, Real and* $\mathsf{S}^{1\uparrow}$ *seals, three-gap classifier, componentwise transport, displayed replay, provenance, and local naming. A consumer may cite the theorem only after the finite window has been inspected and the adjacent gap ledger has been classified into at most three displayed gap classes.*

Proof. Project the carrier of Definition 8005.1. The only circle data available to a theorem-facing consumer are the angle source A, the finite orbit window W, and the cyclic order O.

Inspect the finite window. Consecutive entries of O determine the adjacent gap ledger G, and the dyadic row D compares the displayed gap lengths against the finite tolerance rows read through S, Q, R.

Classify the ledger. The row E records that the displayed adjacent gaps fall into at most three gap classes after the $\mathsf{S}^{1\uparrow}$ seal U and the $\mathsf{Real}^{\uparrow}$ seal R are visible.

Finally transport and replay through H, C, P, N. These rows preserve and name the inspected finite packet, so the certificate cannot import measure, quotient-circle equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8005.3 (Three-distance finite window inspection)**.** *Every accepted* $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ *read factors through the finite route*

$$A \longmapsto W \longmapsto O \longmapsto G \longmapsto D \longmapsto (S, Q, R, U) \longmapsto E.$$

The route supplies the three-gap statement by inspection of the displayed orbit window and adjacent gap ledger; it does not use measure, a quotient equality test for circle points, an infinite equidistribution theorem, a selected subsequence, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Read the rotation source and finite window first. The cyclic order row O is part of the packet, so the adjacent gap ledger G is generated from displayed neighboring rows rather than from a host quotient of the circle.

Next read the numeric side. The dyadic ledger D, stream schedule S, regular rational readback Q, Real seal R, and circle seal U are the only routes by which gap lengths become visible.

The classifier row E consumes exactly those rows and records the at-most-three gap classes. Componentwise transport, replay, provenance, and local naming expose no additional analytic principle, so the theorem remains a finite window certificate. □

Falsifiable prediction: An accepted $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ read is refuted if the adjacent gap ledger G or dyadic tolerance ledger D is removed while the three-gap classifier row E is still claimed.

Independence witness: sone, kroneckerapproximation, weylequidistribution, dyadicratcore, regseqrat, real: $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite rotation windows, cyclic ordering, adjacent gap classification, dyadic tolerance rows, StreamName scheduling, RegSeqRat readback, Real and circle seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ThreeDistanceTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{ThreeDistanceTheorem}^{\uparrow}$ packet is generated from a rotation-angle row, finite orbit-window row, cyclic ordering row, adjacent gap ledger, dyadic tolerance ledger, StreamName schedule, RegSeqRat readback, Real and circle seals, three-gap classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ThreeDistanceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8005.1, Definition 8005.2, and Definition 8005.3 seed the finite three-distance route over circle windows, adjacent gap ledgers, dyadic tolerance, regular rational readback, Real and circle seals, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ThreeDistanceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic equidistribution, quotient-circle equality, selected subsequences, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, orbit-window inspection, cyclic-order exposure, adjacent gap ledger exactness, dyadic tolerance readback, Real and circle sealing, transport, replay, provenance, and local naming for BEDC.Derived.ThreeDistanceTheoremUp.

Chapter 8006

A Concrete Naming Certificate for ClosedSubstitutionSelfCompile$^{\uparrow}$

The ClosedSubstitutionSelfCompile$^{\uparrow}$ packet realizes the closed-term substitution self-compile boundary as finite MetaCIC NameCert data. It sits after the closed boundary of chapter 527, the substitution spine realizer of chapter 6990, the kernel acceptance witness of chapter 549, and the MetaCIC critical-path packet of chapter 765. The packet records closed-term, substitution, binder-seal, replay, refusal, transport, provenance, and local naming rows before any compiler-facing consumer may read the self-compile handoff.

Definition 8006.1 (Closed substitution self-compile carrier)**.** *A* ClosedSubstitutionSelfCompile$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, D, V, Q, B, L, R, F, K, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC closed-term source row.*
2. *D is the de Bruijn depth row at which the closed boundary is read.*
3. *V is the substitution payload row.*
4. *Q is the* `ClosedAt` *classifier row for T at depth D.*
5. *B is the binder-seal row ensuring that the boundary is read before any recursive binder descent.*
6. *L is the displayed closed-substitution row for substituting V into T at depth D.*
7. *R is the replay row exposing the same closed substitution route to the compiler-facing consumer.*
8. *F is the refusal row for open-term substitution, unrestricted binder descent, and host evaluator equality.*
9. *K is the kernel-acceptance witness row tying the displayed MetaCIC packet to an accepted kernel-facing certificate.*
10. *A is the self-compile audit row recording that the compiler reads only $T, D, V, Q, B, L, R, F, K$.*
11. *H records componentwise* hsame *transport for the term, depth, payload, classifier, binder seal, substitution, replay, refusal, kernel, and audit rows.*

12. *C records displayed* Cont *replay from each compiler-facing query back through the same finite rows.*

13. *P is the* Pkg *provenance over* $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$, $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$, MetaCICSubstitution
KernelAcceptanceWitness$^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted substitution, host evaluator equality, open-term semantic preservation, quotient syntax equality, or global compiler correctness.

Theorem 8006.2 (Closed substitution self-compile NameCert obligations)**.** *For every accepted* $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$ *carrier*

$$S = (T, D, V, Q, B, L, R, F, K, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed MetaCIC term row, depth row, substitution payload row, closedness classifier, binder seal, closed-substitution row, replay row, refusal row, kernel-acceptance witness, self-compile audit row, componentwise transport, displayed replay, provenance, and local naming. A compiler-facing consumer may read the self-compile handoff only through those displayed rows.*

Proof. Project the carrier of Definition 8006.1. The closed term, depth, payload, closedness classifier, binder seal, substitution row, replay row, refusal row, kernel row, and audit row are explicit coordinates.

Read the boundary rows in order. The classifier Q and binder seal B are exposed before the substitution row L, so the packet is confined to the closed boundary rather than to unrestricted substitution under binders.

Read the replay and refusal rows. The replay row R is the only compiler-facing route for the closed substitution, while F keeps open-term substitution, host evaluator equality, and quotient syntax equality outside the packet.

Finally consume K, A, H, C, P, N. Kernel acceptance, self-compile audit, transport, replay, provenance, and local naming preserve the same finite surface and do not add global compiler correctness. □

Theorem 8006.3 (Closed substitution self-compile refusal boundary)**.** *Every accepted* ClosedSubstitutionSelfCompi
read keeps the refusal row F visible whenever the compiler-facing handoff A is consumed. Thus the packet can replay closed-term substitution through R, A, H, C, P, N, but it cannot claim unrestricted substitution, open-term semantic preservation, host evaluator equality, quotient syntax equality, or full compiler correctness.

Proof. Start with the handoff row A. By the carrier definition, A reads only the displayed rows $T, D, V, Q, B, L, R, F, K$ before transport, continuation, provenance, and local naming are consumed.

The refusal row F is part of that handoff inventory. Removing F changes the carrier read, so an accepted handoff cannot silently turn the closed-boundary packet into an unrestricted substitution theorem.

The structural rows H, C, P, N replay and name the same finite packet. They preserve the visible refusal boundary and therefore block host evaluator equality, quotient syntax equality, open-term preservation, and full compiler correctness claims. □

Theorem 8006.4 (Closed substitution self-compile TasteGate readiness)**.** *The Lean follow-up for* $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$ *must provide explicit* $\mathsf{Nontrivial\ ClosedSubstitutionSelfCompile}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ ClosedSubstitutionSelfCompile}^{\uparrow}$ *instances for the same finite carrier. These instances must keep* $T, D, V, Q, B, L, R, F, K, A, H, C, P, N$ *as separate displayed rows before any checked TasteGate theorem can serve as a public self-compile gate.*

Proof. Use the carrier of Definition 8006.1. Nontriviality is forced by the distinction between an accepted closed-boundary packet and a refused open-term substitution packet.

Field faithfulness is forced by the row inventory. A read that collapses the binder seal, refusal row, kernel row, or audit row does not preserve the displayed carrier and therefore cannot serve as the public gate for this packet. □

Falsifiable prediction: Every accepted $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$ compiler-facing read exposes the closed term, substitution row, binder seal, replay row, refusal row, kernel acceptance row, self-compile audit row, transport, provenance, and local NameCert rows before the handoff is visible.

Independence witness: closedtermsubstitutionboundary, metacic-substitution-spine-realizer, kernelacceptancewitness, metaciccriticalpath: $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed substitution boundary, binder seal, replay row, refusal row, kernel witness, self-compile audit, transport, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}$ packet is generated from a closed MetaCIC term row, de Bruijn depth row, substitution payload row, ClosedAt classifier, binder seal, closed-substitution row, replay row, refusal row, kernel acceptance witness, self-compile audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8006.1, Definition 8006.2, Definition 8006.3, and Definition 8006.4 seed the finite closed-substitution self-compile carrier, refusal boundary, handoff discipline, and TasteGate readiness surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSelfCompile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution, open-term semantic preservation, host evaluator equality, quotient syntax equality, full compiler correctness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, binder-seal routing, closed-substitution replay, refusal preservation, kernel acceptance handoff, self-compile audit visibility, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.ClosedSubstitutionSelfCompileUp.

Chapter 8007

A Concrete Naming Certificate for $\mathsf{MobiusInversion}^{\uparrow}$

Following Rota on finite incidence algebras, the $\mathsf{MobiusInversion}^{\uparrow}$ packet records Mobius inversion over a finite poset as ledger data rather than as an ambient algebra theorem. It sits after the relation skeletons of chapter 52 and chapter 53, the meet-and-join comparison surface of chapter 55, the finite enumeration discipline of chapter 112, and the membership-facing carrier route of chapter 3668. The packet names zeta and Mobius rows only on displayed finite intervals; it does not add an infinite incidence algebra, a host ring completion, an unbounded summation principle, selected inverse matrices, or ambient choice.

Definition 8007.1 (Mobius inversion carrier)**.** *A* $\mathsf{MobiusInversion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, I, Z, K, W, Q, H, C, G, N)$$

with the following displayed rows:

1. *P is the finite poset source row, recording the* $\mathsf{Poset}^{\uparrow}$ *comparison data and the underlying* $\mathsf{Preorder}^{\uparrow}$ *reflexive-transitive ledger.*
2. *I is the interval ledger. For every displayed comparable pair $x \leq y$, it lists the finite interval members as a* $\mathsf{FinSet}^{\uparrow}$ *row with membership readback through* $\mathsf{SetLike}^{\uparrow}$*.*
3. *Z is the zeta row, assigning the displayed unit value to every accepted interval entry in I and refusing non-comparable pairs.*
4. *K is the Mobius row, recording the recursively checked alternating interval coefficients attached to the same finite interval ledger.*
5. *W is the finite convolution window, listing exactly the intermediate z with $x \leq z \leq y$ that may be used in a zeta-Mobius product.*
6. *Q is the inverse-check ledger, recording both finite convolution checks: the zeta-then-Mobius and Mobius-then-zeta rows collapse to the displayed diagonal row on each accepted interval.*
7. *H records componentwise* hsame *transport for the poset source, interval ledger, zeta row, Mobius row, convolution window, inverse-check ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an inversion request through $P, I, Z, K, W, Q, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{MobiusInversion}^{\uparrow}$*,* $\mathsf{Preorder}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*,* $\mathsf{Lattice}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MobiusInversion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $P, I, Z, K, W, Q, H, C, G, N$. The carrier has no coordinate for an infinite incidence algebra, host matrix inversion, selected global inverse, unbounded summation, quotient equality of functions, or ambient algebraic completion.

Theorem 8007.2 (Mobius inversion NameCert obligations)**.** *For every accepted* $\mathsf{MobiusInversion}^{\uparrow}$ *carrier $M = (P, I, Z, K, W, Q, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MobiusInversion}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite poset source, interval ledger, zeta row, Mobius row, finite convolution window, inverse-check ledger, componentwise transport, displayed replay, package provenance, and local naming rows. A consumer may read inversion data only after it displays the route*

$$P, I \longmapsto Z, K \longmapsto W, Q \longmapsto H, C, G, N.$$

Proof. Project the carrier of Definition 8007.1. The poset source P supplies the comparison row, while I supplies the finite interval members through the Finset and SetLike routes.

The algebra-facing rows are then read from the same interval ledger. The zeta row Z and the Mobius row K attach coefficients only to displayed comparable pairs, and W lists the finite intermediates allowed in a convolution.

Finally inspect Q, H, C, G, N. The inverse-check ledger Q records the two finite convolution identities, and the structural rows transport, replay, source, and name the same packet. Since no accepted row stores infinite incidence algebra data, host matrix inversion, selected inverse matrices, unbounded sums, quotient equality of functions, or ambient completion, the listed rows exhaust the NameCert obligations. □

Theorem 8007.3 (Mobius inversion incidence ledger)**.** *Every accepted* $\mathsf{MobiusInversion}^{\uparrow}$ *incidence consumer receives the inverse relation only through the displayed finite ledger*

$$P, I, Z, K, W, Q, H, C, G, N.$$

In particular, the consumer must preserve the same $\mathsf{Preorder}^{\uparrow}$ *and* $\mathsf{Poset}^{\uparrow}$ *comparison rows, the same* $\mathsf{FinSet}^{\uparrow}$ *interval enumeration, the same* $\mathsf{SetLike}^{\uparrow}$ *membership readback, the same zeta and Mobius rows, and the same finite convolution window before it may cite an inversion check.*

Proof. Start with a request for inversion on an interval. The comparison is first routed through P, so the request is bounded by the finite poset source rather than by a host incidence algebra.

Use the interval ledger I and the convolution window W. They list the only intermediate elements available to the consumer, with membership read through the Finset and SetLike rows. The zeta and Mobius rows are therefore attached to the same finite interval.

The inverse-check ledger Q may then be read, but only for those displayed finite windows. The structural rows H, C, G, N preserve the same route under transport, replay, provenance, and local naming, so the consumer never receives an unbounded summation principle or an ambient inverse object. □

Falsifiable prediction: Every accepted $\mathsf{MobiusInversion}^{\uparrow}$ read factors through finite Poset and Preorder source rows, interval ledgers, zeta rows, Mobius rows, finite convolution windows, inverse-check ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any incidence consumer may cite inversion.

Independence witness: preorder, poset, lattice, finset, setlike: $\mathsf{MobiusInversion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite interval ledgers, zeta rows, Mobius rows,

convolution windows, inverse-check ledgers, transport, replay, provenance, and local NameCert data as one incidence packet.

Future Lean TasteGate obligations for $\mathsf{MobiusInversion}^{\uparrow}$ must certify the finite incidence route at `lean4/BEDC/Derived/MobiusInversionUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{MobiusInversion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MobiusInversion}^{\uparrow}$ packet is generated from a finite Poset source row, a Preorder comparison ledger, Finset interval enumeration, SetLike membership readback, zeta and Mobius rows, finite convolution windows, inverse-check ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MobiusInversion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8007.1, Definition 8007.2, and Definition 8007.3 seed the finite incidence-ledger route over Preorder, Poset, Lattice, Finset, and SetLike rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MobiusInversion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim infinite incidence algebras, host matrix inversion, selected inverse matrices, quotient equality of functions, unbounded summation, ambient algebraic completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-ledger exactness, zeta-row exactness, Mobius-row exactness, finite convolution checks, inverse-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.MobiusInversionUp.

Chapter 8008

A Concrete Naming Certificate for $\mathsf{AnalyticSet}^{\uparrow}$

Following Suslin and Luzin in descriptive set theory, the $\mathsf{AnalyticSet}^{\uparrow}$ packet records analytic subsets as finite BEDC data for a projection-coded tree over Baire-space names. It sits after the sequence carrier of chapter 334, the Borel-code hierarchy of chapter 4020, the membership surface of chapter 3668, the topological source rows of chapter 88, and the terminal real-name boundary of chapter 38. The chapter names the projection-code surface used by later descriptive-set and measurable-function consumers; it is not a universal classical separation theorem, a quotient point-set equality, a selected witness branch, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8008.1 (AnalyticSet carrier)**.** *An* $\mathsf{AnalyticSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, B, T, S, P, R, H, C, G, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{BaireSpace}^{\uparrow}$ *row whose finite sequence windows name candidate witnesses.*
2. *B is the* $\mathsf{BorelHierarchy}^{\uparrow}$ *code row for the subset of the product space whose projection is under inspection.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *row supplying the product, cylinder, and projection-domain reads used by the Borel code.*
4. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership row recording finite positive and negative membership probes against the projected set.*
5. *P is the projection-code row, recording that a displayed Baire witness window and Borel-code read are consumed before the projected membership row is visible.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary row used when the source topology or later measurable-function consumers read real-valued probes.*
7. *H records componentwise* hsame *transport for the Baire-space, Borel-code, topology, membership, projection-code, real-boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a membership request through*

$$X, B, T \longmapsto P \longmapsto S, R.$$

9. *G is the* Pkg *provenance over* $\mathsf{AnalyticSet}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BorelHierarchy}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AnalyticSet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a selected infinite witness branch, quotient equality of subsets, classical separation, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8008.2 (AnalyticSet NameCert obligations)**.** *For every accepted* $\mathsf{AnalyticSet}^{\uparrow}$ *carrier* $A = (X, B, T, S, P, R, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AnalyticSet}^{\uparrow}}$ *surface consists exactly of carrier admission, Baire-space witness-window exposure, Borel-code exposure, topology compatibility, set-membership readback, projection-code binding, real-boundary nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the carrier $A = (X, B, T, S, P, R, H, C, G, N)$. The only analytic-set data available to a consumer are the Baire-space windows, Borel-code row, topology row, membership row, projection-code row, and real-boundary row.

Read X, B, T before projection membership. The Baire-space row supplies finite witness windows, the Borel hierarchy row supplies the code being projected, and the topology row supplies the product and cylinder reads needed to interpret the code.

The projection row P then binds those reads to S. A membership consumer may inspect the projected set only through the displayed witness-window and Borel-code route, while R records real-valued topological probes without adding a completion principle.

Finally H, C, G, N transport, replay, source, and name the same packet. Since no carrier coordinate stores a selected infinite branch, quotient subset equality, classical separation, countable choice, or ambient real completeness, none of those principles is part of the local NameCert surface. □

Theorem 8008.3 (AnalyticSet Suslin projection handoff)**.** *Every descriptive-set or measurable-function consumer of an accepted* $\mathsf{AnalyticSet}^{\uparrow}$ *packet receives the analytic set only through the finite projection-code route*

$$\mathsf{BaireSpace}^{\uparrow},\ \mathsf{BorelHierarchy}^{\uparrow},\ \mathsf{Topology}^{\uparrow} \longmapsto P \longmapsto \mathsf{SetLike}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

with structural support H, C, G, N. The handoff records the Suslin projection surface as named BEDC data; it does not export a selected witness branch, a universal separation theorem, quotient equality of subsets, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier definition. The projection-code row P is downstream of the Baire-space witness windows, Borel-code row, and topology rows, so a consumer cannot read projected membership before those rows have been displayed.

Apply the NameCert obligation theorem, Definition 8008.2. It identifies carrier admission, Baire-space exposure, Borel-code exposure, topology compatibility, membership readback, projection binding, and real-boundary nonescape as the complete local obligation surface.

The structural rows H, C, G, N then preserve the same route under transport, continuation replay, provenance, and local naming. They do not add a branch selector, a quotient of subsets, a separation theorem, countable choice, or Real completeness.

Thus the public handoff is exactly a finite Suslin projection-code packet over the named sibling chapters. Downstream consumers may reuse that packet as a descriptive-set boundary, but any stronger classical object would require a coordinate absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{AnalyticSet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AnalyticSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8008.1, Definition 8008.2, and Definition 8008.3 seed the finite projection-code surface for analytic sets over Baire-space, Borel-code, topology, set-membership, and Real-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AnalyticSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical separation, selected witness branches, quotient subset equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection-code binding, Baire-space and Borel-code exposure, topology compatibility, SetLike membership readback, Real-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AnalyticSetUp.

Constructive story: An $\mathsf{AnalyticSet}^{\uparrow}$ packet is generated from BaireSpace witness windows, BorelHierarchy code rows, Topology product and cylinder reads, SetLike membership rows, a projection-code row, a RealUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8009

A Concrete Naming Certificate for $\mathsf{InjectiveModule}^{\uparrow}$

Following Baer and Cartan–Eilenberg, the $\mathsf{InjectiveModule}^{\uparrow}$ packet records the finite homological-algebra interface in which maps out of a displayed submodule inclusion extend into a chosen module row. It sits over the ring and module surfaces of chapter 43 and chapter 46, reads linear maps through chapter 48, and keeps its exactness boundary compatible with the exact-sequence obligation surface of chapter 176 and the displayed $\mathsf{ExactSequence}^{\uparrow}$ handoff in chapter 7321. The packet is an extension-lifting certificate; it is not a host injective object, a derived-functor computation, a quotient representative, or an ambient choice principle.

Definition 8009.1 (Injective module carrier)**.** *An* $\mathsf{InjectiveModule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, M, S, F, E, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the scalar* $\mathsf{Ring}^{\uparrow}$ *row and M is the target* $\mathsf{Module}^{\uparrow}$ *row over that scalar source.*
2. *S is the submodule inclusion row, recording a displayed inclusion $A \hookrightarrow B$ inside the same module classifier window.*
3. *F is the source hom row, read as a* $\mathsf{LinMap}^{\uparrow}$ *map from the submodule row A into M.*
4. *E is the extended hom row, read as a* $\mathsf{LinMap}^{\uparrow}$ *map from the ambient row B into M.*
5. *B is the exact-sequence boundary row, recording that the restriction of E along S is the displayed source hom row F.*
6. *H records componentwise* hsame *transport for $R, M, S, F, E, B, C, P, N$.*
7. *C records displayed* Cont *replay from an extension request through the module, inclusion, source hom, extended hom, and boundary rows.*
8. *P is the* Pkg *provenance over* $\mathsf{InjectiveModule}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{ExactSequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InjectiveModule}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, S, F, E, B, H, C, P, N$. It has no coordinate for a host category of modules, a selected quotient representative, a global Ext calculation, an arbitrary splitting, or an unstated choice of extension.

Theorem 8009.2 (Injective module NameCert obligations)**.** *For every accepted* $\mathsf{InjectiveModule}^{\uparrow}$ *carrier*

$$I = (R, M, S, F, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InjectiveModule}^{\uparrow}}$ *surface consists exactly of scalar-ring admission, target-module admission, submodule-inclusion exposure, source-hom exposure, extended-hom exposure, exact-sequence boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every extension-facing consumer must expose* R, M, S, F, E, B *before it may consume transport, replay, provenance, or naming rows.*

Proof. Project the carrier of Definition 8009.1. The scalar row R and target row M are the only module data available to the classifier, and the inclusion row S is the only displayed submodule input.

The source hom row F and extended hom row E are both read through the $\mathsf{LinMap}^{\uparrow}$ surface over the same scalar source. The boundary row B records their restriction contact, so an extension request cannot bypass S, F, E, or the exact-sequence boundary by appealing to a host quotient or an external lifting theorem.

Transport and replay preserve the same finite route componentwise, while provenance and the local naming row bind it to $\mathsf{InjectiveModule}^{\uparrow}$. Since no other coordinates occur in the carrier, these rows exhaust the NameCert obligations. □

Theorem 8009.3 (Injective module extension lifting)**.** *Every extension-lifting read of an accepted* $\mathsf{InjectiveModule}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Ring}^{\uparrow}, \mathsf{Module}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{ExactSequence}^{\uparrow} \longmapsto \mathsf{InjectiveModule}^{\uparrow}.$$

Concretely, a consumer must read the scalar ring row R*, target module row* M*, submodule inclusion row* S*, source hom row* F*, extended hom row* E*, and exact-sequence boundary row* B *before it may claim that the source map has been extended. The exported lift is the displayed row* E *together with its boundary contact* B*; it is not a host injective-object theorem, Baer criterion over all ideals, Ext-vanishing statement, selected splitting, quotient representative, or ambient choice principle.*

Proof. Begin with the obligation surface of Definition 8009.2. It forces every extension-facing consumer to expose the ring, module, inclusion, source hom, extended hom, and boundary rows before any structural support row is read.

The lift is therefore not a separate existence principle. It is the row E already present in the finite carrier, checked against F through the boundary row B and replayed through the same Cont route.

The sibling chapters supply the route components: $\mathsf{Ring}^{\uparrow}$ and $\mathsf{Module}^{\uparrow}$ provide the scalar and target module rows, $\mathsf{LinMap}^{\uparrow}$ provides the hom rows, and the exact-sequence surfaces keep the restriction boundary visible. The missing host coordinates exclude global injectivity, Ext computations, arbitrary splittings, quotient representatives, and choice-based extension selection. □

Closure status for $\mathsf{NameCert}_{\mathsf{InjectiveModule}^{\uparrow}}$.

Constructive story: An $\mathsf{InjectiveModule}^{\uparrow}$ packet is generated from a scalar RingUp row, target ModuleUp row, submodule inclusion row, source LinearMapUp hom row, extended LinearMapUp hom row, ExactSequenceUp boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InjectiveModule}^{\uparrow}}$, seed)

Scope closed: Definition 8009.1, Definition 8009.2, and Definition 8009.3 seed the finite extension-lifting surface over ring, module, submodule inclusion, linear-map, exact-sequence boundary, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{InjectiveModule}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Baer criterion, Ext-vanishing, host injective objects, arbitrary splittings, quotient representatives, or choice-based extension selection.

Upgrade path: Obligation closure requires checked carrier admission, module-row exposure, submodule-inclusion exposure, source-hom and extended-hom exposure, exact-boundary contact, transport, replay, provenance, and local naming for BEDC.Derived.InjectiveModuleUp.

Chapter 8010

A Concrete Naming Certificate for $\mathsf{LocatedRealReciprocal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRealReciprocal}^{\uparrow}$ packet records the finite reciprocal boundary for a located positive real name. It sits after the apartness evidence of chapter 304, the positive source discipline of chapter 1587, the dyadic lower-bound ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the guarded reciprocal route of chapter 6251. The packet names a reciprocal request only after located apartness from zero has been displayed; it is not host division, division by zero, quotient real equality, trichotomy, a selected inverse, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8010.1 (Located real reciprocal carrier)**.** *A* $\mathsf{LocatedRealReciprocal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, A, P, D, Q, I, E, H, C, G, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose reciprocal is requested.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *row witnessing that R is apart from the zero boundary.*
3. *P is the* $\mathsf{PositiveReal}^{\uparrow}$ *orientation row that presents the apartness witness as a positive located radius.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound ledger for the displayed radius and tolerance budget.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the dyadic budget into regular rational reciprocal windows.*
6. *I is the guarded* $\mathsf{RealReciprocal}^{\uparrow}$ *row served by the same apartness, positivity, dyadic, and regular-readback data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the reciprocal candidate.*
8. *H records componentwise* hsame *transport for $R, A, P, D, Q, I, E, C, G, N$.*
9. *C records displayed* Cont *replay from the located source through apartness, positive-radius, dyadic-budget, regular-readback, guarded-reciprocal, and terminal-seal rows.*
10. *G is the* Pkg *provenance over* $\mathsf{LocatedRealReciprocal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealReciprocal}^{\uparrow}}$ *row.*

The carrier stores a finite located-domain reciprocal surface. It has no coordinate for host division, division by zero, quotient real equality, trichotomy, selected inverse choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Definition 8010.2 (Located real reciprocal classifier)**.** *Two* $\mathsf{LocatedRealReciprocal}^{\uparrow}$ *carriers*

$$L = (R, A, P, D, Q, I, E, H, C, G, N) \qquad \textit{and} \qquad L' = (R', A', P', D', Q', I', E', H', C', G', N')$$

are classified as the same displayed located reciprocal packet exactly when the real source rows match through the displayed real-name classifier, the apartness and positive-radius rows expose the same finite separation from zero, the dyadic ledgers carry the same lower-bound and tolerance budgets, the regular rational readbacks serve the same reciprocal windows, the guarded $\mathsf{RealReciprocal}^{\uparrow}$ *rows are transported with the same terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the replay, provenance, and local naming rows preserve the same route. The classifier is rowwise over $R, A, P, D, Q, I, E, H, C, G, N$ and does not identify carriers by host real equality, quotient streams, trichotomy, division by zero, or a selected inverse.*

Theorem 8010.3 (Located real reciprocal NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealReciprocal}^{\uparrow}$ *carrier*

$$L = (R, A, P, D, Q, I, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealReciprocal}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier formation for the real source, apartness row, positive-radius orientation, dyadic lower-bound ledger, regular rational readback, guarded reciprocal row, terminal real seal, transports, continuations, provenance, and local name;*
2. *classifier stability for the rowwise comparison of Definition 8010.2;*
3. *located-domain admission, requiring the apartness row A and positive-radius row P to be exposed before any reciprocal readback is consumed;*
4. *dyadic-budget exactness, requiring every reciprocal tolerance read in Q to be served by the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D;*
5. *guarded reciprocal handoff, requiring I to be consumed through* $\mathsf{RealReciprocal}^{\uparrow}$ *before the terminal seal E is read;*
6. *ledger non-escape, excluding host division, division by zero, quotient real equality, trichotomy, selected inverse choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier formation is projection from Definition 8010.1. The displayed packet lists exactly the source, apartness, positive-radius, dyadic-budget, regular-readback, reciprocal, terminal-seal, transport, replay, provenance, and naming rows.

Classifier stability is the componentwise comparison of Definition 8010.2. The transport row H can replace visible representatives only with representatives carrying the same located-domain evidence, budget ledger, reciprocal handoff, and terminal seal.

Located-domain admission reads A and P before D, Q, I, E. Thus a reciprocal request with no displayed apartness from zero, or no positive located radius attached to that apartness row, never enters the accepted packet surface.

Dyadic-budget exactness and guarded reciprocal handoff are the roles of D, Q, I, E. The regular readback Q is downstream of the dyadic ledger D, while the terminal seal E is downstream of the guarded $\mathsf{RealReciprocal}^{\uparrow}$ row I.

The non-escape row follows from the carrier inventory. No field stores host division, division by zero, quotient real equality, trichotomy, selected inverse choice, or ambient real completeness, and the structural rows only transport and replay the displayed finite route. □

Theorem 8010.4 (Located real reciprocal non-escape)**.** *Every accepted* $\mathsf{LocatedRealReciprocal}^{\uparrow}$ *read reaches its reciprocal candidate only through the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{ApartnessReal}^{\uparrow},\ \mathsf{PositiveReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RealReciprocal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Consequently the packet exports no host division, no division by zero, no quotient real equality, no trichotomy principle, no selected inverse, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use the obligation surface of Definition 8010.3. Located-domain admission places the real source, apartness witness, and positive-radius row before any reciprocal read.

The dyadic ledger and regular rational readback then provide the finite tolerance route. The guarded reciprocal row is consumed only after those budgets are exposed, and the terminal real seal is read only after the guarded reciprocal handoff.

The route displayed in the statement is therefore the entire value-facing path through the carrier. Since the omitted principles have no coordinates in $L = (R, A, P, D, Q, I, E, H, C, G, N)$, neither componentwise transport nor continuation replay can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealReciprocal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealReciprocal}^{\uparrow}$ packet is generated from a displayed Real source, apartness from zero, a positive located radius, a dyadic lower-bound ledger, RegSeqRat reciprocal windows, a guarded RealReciprocal row, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealReciprocal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8010.1, Definition 8010.2, Definition 8010.3, and Definition 8010.4 seed the finite located reciprocal surface over apartness, positive-radius, dyadic-budget, regular-readback, guarded reciprocal, and terminal Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealReciprocal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host division, division by zero, quotient real equality, trichotomy, selected inverse choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, located-domain admission, dyadic-budget exactness, guarded reciprocal handoff, terminal seal routing, and non-escape for the displayed packet.

Chapter 8011

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$ packet records the finite coherence surface tying compact nets, pointwise moduli, rational lower-bound folds, metric-triangle estimates, and uniform-modulus handoff rows into one reusable certificate. It sits over the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the rational fold route of chapter 619, the final triangle estimate of chapter 1159, and the consumer boundary of chapter 369. The packet realizes the compact-uniform-continuity route as a shared finite coherence object; it is not an open-cover compactness theorem, a selected Lebesgue number, a host modulus function, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8011.1 (CompactUniformModulusCoherence carrier)**.** *A* $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, L, T, U, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including the inspected centers, coverage windows, and compact-source metric data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the same source and target metric carriers.*
3. *L is the rational lower-bound fold row, read through the finite ProbeBundle route and the* $\mathsf{Rat}^{\uparrow}$ *comparison surface of chapter 619.*
4. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *row that reuses the selected compact-net center for the source and target triangle estimates.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row produced only after K, F, L, T have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *or regular rational readback boundary used when the uniform threshold is inspected by Real and Completion consumers.*
7. *H records componentwise* hsame *transport for the compact-net, pointwise-modulus, lower-bound fold, triangle-estimate, uniform-modulus, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through*

$$K, F \longmapsto L \longmapsto T \longmapsto U, R.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCoherence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows. It has no coordinate for open-cover compactness, selected subsequences, selected Lebesgue numbers outside the fold, host modulus functions, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8011.2 (CompactUniformModulusCoherence NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$ *carrier* $M = (K, F, L, T, U, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCoherence}^{\uparrow}}$ *surface consists exactly of carrier admission, compact finite-net exposure, continuous-map pointwise-modulus exposure, rational lower-bound fold exactness, metric-triangle uniform-estimate coherence, uniform-modulus handoff, Real-facing readback nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier $M = (K, F, L, T, U, R, H, C, P, N)$. The only rows available to a uniform-modulus consumer are the compact finite-net row, continuous-map pointwise rows, rational lower-bound fold, triangle-estimate row, uniform-modulus handoff, and readback boundary.

Read K and F first. The compact metric row supplies the finite net and coverage windows, while the continuous-map row supplies the pointwise modulus data at the displayed centers.

Next read the fold and triangle rows. The lower-bound fold L turns the finite radius family into a single tolerance threshold, and T keeps the source and target metric-triangle estimates locked to the same selected center and tolerance request.

Finally U, R, H, C, P, N export, read back, transport, replay, source, and name the same finite route. Since no carrier coordinate stores open-cover compactness, a selected subsequence, an external Lebesgue number, a host modulus function, countable choice, quotient real equality, or ambient real completeness, none of those objects is part of the local NameCert surface. □

Theorem 8011.3 (CompactUniformModulusCoherence vision handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or Real-facing consumer of an accepted* $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$ *packet must read the shared route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto L \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

with structural support H, C, P, N. The handoff realizes the compact-uniform-continuity finite-net, pointwise-modulus, rational-fold, triangle-estimate, and uniform-modulus route as one coherence packet, not as a host compactness theorem or an external modulus selector.

Proof. Begin with the carrier definition. The handoff row U is downstream of the compact finite-net row K, continuous-map pointwise rows F, rational fold L, and triangle-estimate row T.

Apply the NameCert obligation theorem, Definition 8011.2. It identifies finite-net exposure, pointwise-modulus exposure, fold exactness, triangle-estimate coherence, uniform-modulus handoff, and Real-facing readback as the complete local obligation surface.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. They do not add open-cover compactness, a selected subsequence, a host modulus, countable choice, quotient real equality, or Real completeness.

Thus the compact-uniform-continuity vision is consumed here as a single finite coherence packet. Downstream modulus, Real, and Completion consumers may reuse the displayed route, but any stronger compactness or selection principle would require a coordinate absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusCoherence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusCoherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8011.1, Definition 8011.2, and Definition 8011.3 seed the shared compact-net, pointwise-modulus, rational-fold, triangle-estimate, and UniformModulus handoff packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusCoherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequences, selected Lebesgue numbers, host modulus functions, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous-map pointwise-modulus exposure, rational lower-bound fold exactness, metric-triangle coherence, uniform-modulus handoff, Real-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusCoherenceUp.

Constructive story: A $\mathsf{CompactUniformModulusCoherence}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMap pointwise-modulus rows, a rational lower-bound fold, MetricTriangleUniformEstimate rows, a UniformModulus handoff, RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8012

A Concrete Naming Certificate for StepFunctionUniformApproximation$^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform approximation, the StepFunctionUniformApproxim packet records the finite certificate by which a regulated target is read through displayed step approximants and uniform tolerance rows. It sits between the dyadic step approximation source of chapter 3774, the regulated step-function surface of chapter 7169, the uniform limit route of chapter 463, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real boundary of chapter 38. The packet names uniform approximation as finite BEDC data before any host function equality, quotient stream equality, selected limiting function, countable-choice schedule, or ambient real-completeness theorem is available.

Definition 8012.1 (Step-function uniform approximation carrier)**.** *A* StepFunctionUniformApproximation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, R, U, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicStepApproximation$^{\uparrow}$ *source row, recording the displayed dyadic mesh, finite cell ledger, step values, and oscillation bounds used by the approximation.*
2. *R is the regulated-function target row, read through the* RegulatedStepFunction$^{\uparrow}$ *surface and recording the regulated target boundary to which the step data is compared.*
3. *U is the shared uniform tolerance ledger. It records one finite tolerance budget whose dyadic reads are shared by the step source, regulated target, finite windows, and terminal real seal.*
4. *W is the* StreamName$^{\uparrow}$ *finite-window row through which the target and approximants are inspected.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed window comparisons and tolerance witnesses.*
6. *E is the* Real$^{\uparrow}$ *seal row, reached only after D, R, U, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the step source, regulated target, tolerance ledger, finite windows, readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the dyadic step source and regulated target through the shared uniform tolerance ledger, finite windows, readback, real seal, transports, provenance, and local name.*

9. *P is the* Pkg *provenance over* StepFunctionUniformApproximation$^{\uparrow}$, DyadicStepApproximation$^{\uparrow}$, RegulatedStepFunction$^{\uparrow}$, UniformLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{StepFunctionUniformApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite uniform-approximation surface. It contains no host function equality, quotient stream equality, selected limit, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness coordinate.*

Theorem 8012.2 (Step-function uniform approximation NameCert obligations)**.** *For every accepted* StepFunctionUniformApproximation$^{\uparrow}$ *carrier $S = (D, R, U, W, Q, E, H, C, P, N)$, the local* NameCert$_{\text{StepFunctionUniformApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic step-source admission through D, regulated-target admission through R, shared uniform-tolerance exactness through U, finite-window readback through W and Q,* Real$^{\uparrow}$ *seal nonescape through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8012.1. The analytic rows are the dyadic step source D, the regulated target row R, the shared tolerance ledger U, the finite window row W, the regular rational readback row Q, and the terminal real seal E.

Read the source side first. The dyadic step approximation chapter supplies finite cells, displayed values, and oscillation data, while the regulated step-function chapter supplies the target-facing regulated row. These two rows are the only places where step data and regulated-target data enter the packet.

The uniform comparison is then finite. The tolerance ledger U supplies one shared budget, W selects the finite windows on which that budget is inspected, and Q records the regular rational readback of the displayed comparisons before E is available.

Finally, H, C, P, N preserve and name the same route. Since the carrier has no host-function coordinate, quotient-stream coordinate, selected-limit coordinate, choice schedule, or ambient completeness row, the listed rows exhaust the local NameCert surface. □

Theorem 8012.3 (Step-function uniform approximation regulated route)**.** *Every* Real$^{\uparrow}$*-facing regulated approximation read of an accepted* StepFunctionUniformApproximation$^{\uparrow}$ *carrier factors through*

$$D, R, U, W, Q, E, H, C, P, N.$$

The route exports only the displayed dyadic step source, regulated target, shared uniform tolerance ledger, finite windows, regular rational readback, real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no host function equality, quotient stream equality, selected limit, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface of Definition 8012.2. It confines every accepted packet to the rows $D, R, U, W, Q, E, H, C, P, N$.

Apply the step-source and regulated-target sibling chapters. The source row D supplies finite dyadic step data, and the target row R keeps the regulated function boundary displayed before the uniform comparison is read.

The Real-facing handoff is available only after U, W, Q. The shared tolerance ledger fixes the finite budget, the stream windows expose the finite observations, and the regular rational readback records the comparison before the Real$^{\uparrow}$ seal is public.

The remaining rows are structural. Transport, replay, provenance, and local naming repack the same finite route, while the refused host equality, quotient equality, selected limit, countable choice, and ambient completeness objects have no coordinate through which to enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{StepFunctionUniformApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{StepFunctionUniformApproximation}^{\uparrow}$ packet is generated from a dyadic step approximation source, a regulated target row, a shared uniform tolerance ledger, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StepFunctionUniformApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8012.1, Definition 8012.2, and Definition 8012.3 seed the finite uniform-approximation packet connecting dyadic step data and regulated-function consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StepFunctionUniformApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, step-source admission, regulated-target admission, uniform-tolerance exactness, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.StepFunctionUniformApproximationUp.

Chapter 8013

A Concrete Naming Certificate for $\mathsf{GaloisConnectionClosureOperator}^{\uparrow}$

Birkhoff and Ore-style Galois-connection closure operators enter BEDC as finite closure-operator NameCert carriers rather than as ambient powerset laws. The $\mathsf{GaloisConnectionClosureOperator}^{\uparrow}$ packet refines the adjunction packet of chapter 6259: it names the source-side composite $G \circ F$ as its own closure surface, while leaving the bare two-map adjunction available to lattice, topological, and metric consumers. It also gives the closure-facing packets of chapter 3559 and chapter 3342 a reusable order-theoretic source for inflation, monotonicity, idempotence replay, and finite provenance.

Definition 8013.1 (Galois-connection closure-operator carrier)**.** *A* $\mathsf{GaloisConnectionClosureOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, Q, F, G, M, A, O, I, D, H, R, L, N)$$

with the following rows:

1. *P is the source ordered carrier inherited from the* $\mathsf{GaloisConnection}^{\uparrow}$ *source row.*
2. *Q is the target ordered carrier inherited from the* $\mathsf{GaloisConnection}^{\uparrow}$ *target row.*
3. *F is the displayed monotone map from P to Q.*
4. *G is the displayed monotone map from Q to P.*
5. *M records the two monotonicity ledgers for F and G.*
6. *A records the adjunction inequality equivalence $F(x) \leq_Q y$ iff $x \leq_P G(y)$.*
7. *O is the displayed closure-operator row naming the composite $G \circ F$ on P.*
8. *I records the inflation ledger $x \leq_P O(x)$ obtained through the adjunction row.*
9. *D records the idempotence replay ledger $O(O(x)) \leq_P O(x)$ together with the reverse inequality supplied by inflation and monotonicity.*
10. *H records componentwise* hsame *transport for the two ordered carriers, the two maps, the adjunction row, the closure row, the inflation row, the idempotence row, provenance, and local naming.*
11. *R records displayed* Cont *replay for order reads, adjunction reads, closure reads, lattice handoff, topological-closure handoff, and metric-closure handoff.*

12. *L is the* Pkg *provenance over* GaloisConnectionClosureOperator$^{\uparrow}$, GaloisConnection$^{\uparrow}$, Lattice$^{\uparrow}$, TopologicalClosure$^{\uparrow}$, MetricClosure$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *N is the local* NameCert$_{\text{GaloisConnectionClosureOperator}^{\uparrow}}$ *row.*

The classifier compares packets only by the displayed ordered carriers, the two map rows, monotonicity, adjunction, the named composite closure row, inflation, idempotence replay, structural transport, continuation, provenance, and local naming. It has no coordinate for arbitrary joins, Moore families, quotient fixed-point lattices, choice-selected closures, or host powerset operations.

Lean4 (definition) `BEDC.Derived.GaloisConnectionClosureOperatorUp`

Theorem 8013.2 (Galois-connection closure-operator NameCert obligations)**.** *For every accepted* GaloisConnectionClosureOperator$^{\uparrow}$ *carrier* $C = (P, Q, F, G, M, A, O, I, D, H, R, L, N)$*, the local* NameCert$_{\text{GaloisConnecti}}$ *surface consists exactly of carrier admission for the ordered source and target rows, monotone-map exposure for F and G, monotonicity exposure through M, adjunction exposure through A, closure-row exposure through O, inflation exposure through I, idempotence replay through D, componentwise transport through H, displayed replay through R, provenance through L, and local naming through N.*

Proof. Project the finite packet $C = (P, Q, F, G, M, A, O, I, D, H, R, L, N)$. The rows P, Q, F, G, M, A reproduce the adjunction surface of chapter 6259, so no closure row is read before the ordered carriers, maps, monotonicity, and inequality equivalence are visible.

The row O names only the composite $G \circ F$ on the source carrier. The inflation ledger I is read from the adjunction row by specializing $F(x) \leq_Q F(x)$, and the monotonicity part of M transports source inequalities through O.

The row D records the replay that makes applying O twice return to the same displayed closure row. The reverse comparison is supplied by inflation applied to $O(x)$, while the forward comparison is the adjunction-and-monotonicity replay carried by D.

Finally, H, R, L, N transport, replay, source, and name the same finite packet. Since the carrier has no arbitrary joins, Moore-family row, quotient fixed-point lattice, choice-selected closure, or host powerset coordinate, the NameCert surface is exactly the listed finite surface. □

Theorem 8013.3 (Galois-connection closure-operator lattice handoff)**.** *Every* Lattice$^{\uparrow}$, TopologicalClosure$^{\uparrow}$, *or* MetricClosure$^{\uparrow}$ *consumer of an accepted* GaloisConnectionClosureOperator$^{\uparrow}$ *packet reads the closure operator through the finite route*

$$P \longrightarrow F \longrightarrow Q \longrightarrow G \longrightarrow P$$

together with the monotonicity ledger M, adjunction ledger A, closure row O, inflation ledger I, idempotence replay ledger D, componentwise transport H, displayed continuation R, provenance L, and local naming N. The handoff exports a reusable closure-operator NameCert packet, not an ambient complete lattice, a topological closure primitive, a metric completion theorem, or a host powerset operation.

Proof. Start with the route displayed in Definition 6259.3. The new packet does not replace that route; it records the composite as the explicit row O and keeps the original ordered carriers, maps, monotonicity ledger, and adjunction ledger visible.

A lattice consumer may use O only after reading I and D, because those rows witness the closure-operator boundary needed for lattice-facing fixed rows. A topological or metric closure consumer may use the same row only through R, which replays the finite closure surface without adding an ambient topology, metric completion, or powerset theorem.

The transport row H and provenance row L keep the handoff tied to the same adjunction packet. The local name N therefore certifies a reusable closure-operator NameCert packet, while any stronger complete-lattice, topological, or metric conclusion must be supplied by the receiving chapter. □

Falsifiable prediction: Deleting the adjunction row A leaves a named composite $G \circ F$ but no finite route that certifies the inflation row I of the closure operator.

Independence witness: galois-connection, lattice, topological-closure, metric-closure: $\mathsf{GaloisConnection}^{\uparrow}$ supplies the adjunction without a separate closure-operator NameCert row; $\mathsf{Lattice}^{\uparrow}$ supplies order consumers without the adjunction-derived composite; $\mathsf{TopologicalClosure}^{\uparrow}$ and $\mathsf{MetricClosure}^{\uparrow}$ supply closure evidence for their own substrates without naming the source-side $G \circ F$ packet. None is bijective to $\mathsf{GaloisConnectionClosureOperator}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{GaloisConnectionClosureOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{GaloisConnectionClosureOperator}^{\uparrow}$ packet is generated from two displayed ordered carrier rows, two monotone map rows, an adjunction-equivalence ledger, the named composite $G \circ F$, inflation, idempotence replay, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaloisConnectionClosureOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8013.1, Definition 8013.2, and Definition 8013.3 seed the finite closure-operator surface over displayed Galois-connection rows, adjunction-derived inflation, idempotence replay, lattice handoff, topological-closure handoff, metric-closure handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaloisConnectionClosureOperator}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.GaloisConnectionClosureOperatorUp`

Bridge status: none

Not claimed: This chapter does not claim arbitrary joins, Moore-family representation, quotient fixed-point lattices, choice-selected closures, host powerset operations, metric completion, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, monotonicity ledgers, adjunction transport, inflation, idempotence replay, lattice handoff, closure-consumer handoff, provenance, and local naming for BEDC.Derived.GaloisConnectionClosu

Chapter 8014

A Concrete Naming Certificate for $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$

Following Brouwer, Bishop, and Bridges, the $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ packet records the finite fan compactness boundary at which bar-depth data may feed compact-uniform-continuity consumers without becoming open-cover compactness. It sits after the fan theorem route of chapter 400, the finite fan theorem route of chapter 5863, the compact metric source rows of chapter 129, the continuous-map graph rows of chapter 130, the uniform-modulus handoff of chapter 369, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet also borders the compact fan handoff of chapter 6522; its role is to expose the exact finite boundary consumed by L10 compact and continuous-map readers, not to supply selected infinite branches or ambient completeness.

Definition 8014.1 (Bishop fan compactness boundary carrier)**.** *A* $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, D, V, K, Q, M, U, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *bar row naming the fan bar whose inspected prefixes are available to the packet.*
2. *D is the fan-bar depth ledger, recording the finite depth bound at which the visible bar inspection stops.*
3. *V is the finite prefix coverage row supplied by* $\mathsf{FiniteFanTheorem}^{\uparrow}$*; it lists exactly the prefixes covered below depth D.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source-probe row, exposing the compact finite-net observations that the fan prefixes are allowed to inspect.*
5. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus window row, tying each displayed compact-source probe to a finite continuity read.*
6. *M is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback window package used to inspect the prefix, compact-source, and pointwise-modulus rows without quotienting streams.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported to compact-uniform consumers after F, D, V, K, Q, M have been displayed.*

8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the value and tolerance reads that enter Q and U.*
9. *H records componentwise* hsame *transport for the fan-bar, depth, prefix-coverage, compact-source, pointwise-modulus, window, uniform-handoff, Real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a compact-uniform request through*

    $\mathsf{FanTheorem}^{\uparrow}, \mathsf{FiniteFanTheorem}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{UniformM}$

11. *P is the* Pkg *provenance over* $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{BishopFanCompactUniform}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFanCompactnessBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, V, K, Q, M, U, R, H, C, P, N$. It has no coordinate for arbitrary open covers, selected infinite branches, countable choice, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8014.2 (Bishop fan compactness boundary NameCert obligations)**.** *For every accepted* $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ *carrier*

$$B = (F, D, V, K, Q, M, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopFanCompactnessBoundary}^{\uparrow}}$ *surface consists exactly of fan-bar admission for F, fan-bar depth admission for D, finite prefix coverage for V, compact-source probe admission for K, pointwise-modulus window exposure for Q, StreamName and RegSeqRat readback for M, uniform-modulus handoff for U, Real-seal boundary exposure for R, componentwise* hsame *transport for H, displayed* Cont *replay for C,* Pkg *provenance for P, and local naming for N.*

Proof. Project the carrier of Definition 8014.1. The fan rows are F, D, V: the bar is named, the inspected depth is finite, and the covered prefixes are exactly those displayed by the finite fan theorem route.

Read the compact and continuity rows next. The compact-source probes K are finite metric observations, while Q records only pointwise-modulus windows tied to those probes.

The window and value rows are downstream. The row M keeps the prefix, compact-source, and modulus reads inside finite StreamName and RegSeqRat windows, while R seals the displayed value and tolerance reads as $\mathsf{Real}^{\uparrow}$ data.

Finally, U, H, C, P, N package the uniform-modulus handoff, transport, replay, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, selected branches, countable choice, quotient streams, selected limits, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8014.3 (Bishop fan compactness boundary handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{FanTheorem}^{\uparrow}, \mathsf{FiniteFanTheorem}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The exported uniformity data are exactly the handoff row U together with the transport, replay, provenance, and naming rows H, C, P, N. The route exports no open-cover compactness, selected infinite branch, countable-choice schedule, quotient stream equality, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the fan boundary rows. The bar row F, depth ledger D, and prefix coverage row V expose a finite inspection boundary before any compact-source read is allowed.

Apply the compact-source and pointwise-modulus rows. The compact probes K and continuity windows Q show which finite observations are being used by the downstream uniform-modulus consumer.

Pass through the stream, regular-rational, and Real rows M, R. These rows seal the inspected windows and value comparisons without selecting an infinite branch, quotienting a stream, or appealing to a selected limit.

The handoff row U can now be exported through H, C, P, N. Because no host compactness or choice coordinate appears in the packet, every accepted consumer remains inside the finite boundary route stated above. □

Falsifiable prediction: Every accepted $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ read must expose fan-bar depth, finite prefix coverage, compact-source probes, pointwise-modulus windows, StreamName and RegSeqRat readback, a UniformModulus handoff, a Real-seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before a compact-uniform consumer may read the packet.

Independence witness: fan-theorem, finite-fan-theorem, compactmetric, continuousmap, uniformmodulus, streamname, regseqrat, real: $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ binds fan-bar depth, finite prefix coverage, compact-source probes, pointwise-modulus windows, finite readback, uniform handoff, Real sealing, transport, replay, provenance, and local naming in one boundary packet; each listed sibling lacks at least one of these local coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFanCompactnessBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$ packet is generated from FanTheorem bar rows, finite fan-depth and prefix-coverage rows, CompactMetric source probes, ContinuousMap pointwise-modulus windows, StreamName and RegSeqRat readback, a UniformModulus handoff, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFanCompactnessBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8014.1, Definition 8014.2, and Definition 8014.3 seed the finite fan compactness boundary over bar depth, prefix coverage, compact probes, pointwise modulus windows, finite readback, Real sealing, and uniform-modulus handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFanCompactnessBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected infinite branches, countable choice, quotient streams, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{BishopFanCompactnessBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked fan-bar admission, depth and prefix coverage, compact-source probe exposure, pointwise-modulus window exposure, finite readback, uniform-modulus handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopFanCompactnessBoundaryUp.

Chapter 8015

A Concrete Naming Certificate for $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$

Following Bishop and Bridges on metric sequential compactness, the $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ packet records the general metric route by which a displayed sequence, a cofinal subsequence schedule, and a candidate point form finite limit-point evidence before any Real-specific cluster packet reads the result. It sits between the metric carrier of chapter 57, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the Real-only cluster packet of chapter 1580. The packet gives compactness and completion consumers a metric sequential limit-point interface without folding the concept into the Real sequence cluster-point carrier.

Definition 8015.1 (Metric sequential limit-point carrier)**.** *A* $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, A, D, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance-observation rows, and finite metric-law ledger used by the sequence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *sequence row, exposing only the inspected windows of the displayed metric sequence.*
3. *A is the cofinal schedule row, recording the finite subsequence indices and monotonicity comparisons used by the limit-point request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the inspected radius windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *distance-readback row, reading the distances $d(x_{a_i}, p)$ through regular rational approximants.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for those distance-readback comparisons.*
7. *W is the limit-window witness row, recording that the displayed scheduled tail enters every inspected tolerance window around the candidate point.*
8. *H records componentwise* hsame *transport for the metric source, stream windows, cofinal schedule, dyadic tolerance ledger, regular rational readback, Real seal, limit-window witness, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, A, D, R, E, W \longmapsto \mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSequenceClusterPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}$ *row.*

The carrier contains no selected subsequence functional, no selected cluster point beyond the displayed candidate row, no quotient stream equality, no countable-choice principle, and no ambient compactness or completeness theorem.

Theorem 8015.2 (Metric sequential limit-point NameCert obligations)**.** *For every accepted* MetricSequentialLimitP *carrier $L = (M, S, A, D, R, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, metric-source visibility, finite stream-window visibility, cofinal schedule compatibility, dyadic tolerance discipline, regular rational distance readback, terminal Real sealing, limit-window exactness, componentwise transport, displayed replay, provenance non-escape, and local naming for the rows $M, S, A, D, R, E, W, H, C, P, N$.*

Proof. Project the carrier. The rows available to a consumer are precisely the metric source, finite stream windows, cofinal schedule, dyadic tolerances, regular rational readbacks, Real seal, limit-window witness, and the structural rows.

Metric-source and stream-window visibility follow from M and S. The schedule row A may index only windows already named by S, and the tolerance row D supplies the displayed radii used by the limit-window request.

The readback and seal rows are downstream. Row R reads distances through $\mathsf{RegSeqRat}^{\uparrow}$, and row E seals those comparisons through $\mathsf{Real}^{\uparrow}$ only after the dyadic tolerance ledger has been displayed.

Finally inspect W, H, C, P, N. The window row records the finite limit-point evidence, while transport, replay, provenance, and local naming preserve exactly the same packet. Since no excluded selector, quotient, choice, compactness, or completeness row is present, the listed obligations exhaust the NameCert surface. □

Theorem 8015.3 (Metric sequential limit-point stream handoff)**.** *Every downstream consumer that passes from* $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ *to a Real-specific cluster or compactness packet must first read the finite handoff*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

In particular, the Real sequence cluster-point route of chapter 1580 may consume the metric limit-window witness only after the stream windows, regular rational readbacks, dyadic tolerances, and terminal Real seal have been displayed.

Proof. Start with the obligation surface of Definition 8015.2. It places the metric source, finite stream windows, cofinal schedule, dyadic tolerance ledger, regular rational readback, and terminal Real seal before the limit-window witness can be exported.

The stream handoff is therefore finite. A Real-specific cluster consumer reads the $\mathsf{StreamName}^{\uparrow}$ windows first, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, the $\mathsf{RegSeqRat}^{\uparrow}$ distance readbacks, and the $\mathsf{Real}^{\uparrow}$ seal.

Transport, replay, provenance, and local naming keep the same rows visible at the hand-off boundary. Because the carrier has no selector, quotient, countable-choice, compactness, or ambient-completeness coordinate, no downstream consumer can bypass the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ limit-point export factors through one metric source row, one finite stream-window row, one cofinal schedule row, one dyadic tolerance ledger, one regular rational distance-readback row, one terminal Real seal, one limit-window witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, streamname, regseqrat, dyadicratcore, real, real-sequence-cluster-point: $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the metric source, sequence windows, cofinal schedule, dyadic tolerance ledger, regular rational distance readback, terminal Real seal, limit-window witness, transport, replay, provenance, and local naming rows for a general metric sequence.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$ packet is generated from a metric source row, finite stream windows, a cofinal schedule, dyadic tolerance ledger, regular rational distance readbacks, a terminal Real seal, a finite limit-window witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8015.1, Definition 8015.2, and Definition 8015.3 seed the general metric sequential limit-point packet over Metric, StreamName, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSequentialLimitPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sequential compactness, Bolzano-Weierstrass, selected subsequence extraction, selected cluster points, quotient stream equality, countable choice, ambient compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window visibility, cofinal schedule compatibility, dyadic tolerance discipline, regular rational distance readback, Real seal exactness, limit-window non-escape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.MetricSequentialLimitPointUp.

Chapter 8016

A Concrete Naming Certificate for $\mathsf{ConstructiveDarboux}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{ConstructiveDarboux}^{\uparrow}$ packet records the finite interval-refinement surface behind the intermediate-value route for a continuous real-valued map on a located interval. It consumes the located interval carrier of chapter 394, the continuous-map and modulus rows of chapter 130, the sign-bracket and bisection route of chapter 1402, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite Darboux-style interval refinement data used by L10 real-completion consumers; it does not import quotient real equality, trichotomy, host real primitives, a selected root, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8016.1 (Constructive Darboux carrier)**.** *A* $\mathsf{ConstructiveDarboux}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, M, B, R, E, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row, carrying the displayed endpoint windows and interval-cell ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the inspected map on the interval.*
3. *M is the continuity-modulus row attached to F and to the requested rational output tolerance.*
4. *B is the interval-refinement row, storing the finite nested bracket, midpoint, and sign-change data inherited from the* $\mathsf{IntermediateValue}^{\uparrow}$ *route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint values, midpoint values, and tolerance comparisons opened by B.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the interval, continuity, refinement, sign, and readback rows are visible.*
7. *S is the sign-change and readback row, recording the displayed negative-side and positive-side value witnesses used by the refinement.*
8. *H records componentwise* hsame *transport for the interval, map, modulus, refinement, readback, Real seal, sign, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an intermediate-value request through I, F, M, B, R, E, S, H.*

10. *P is the* Pkg *provenance over* ConstructiveDarboux$^{\uparrow}$, LocatedInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, IntermediateValue$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{ConstructiveDarboux}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, F, M, B, R, E, S, H, C, P, N$. It has no coordinate for quotient classes of real names, real trichotomy, host real equality, an unrestricted zero search, a selected root, or ambient completeness.

Theorem 8016.2 (Constructive Darboux obligation basis)**.** *For every accepted* ConstructiveDarboux$^{\uparrow}$ *carrier $D = (I, F, M, B, R, E, S, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveDarboux}^{\uparrow}}$ *surface consists exactly of carrier admission, located-interval admission, continuous-map admission, continuity-modulus exposure, finite interval-refinement exposure, sign-change and readback exposure, terminal* Real$^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8016.1. The accepted packet displays one located interval row, one continuous-map row, one modulus row, one finite refinement row, one regular-rational readback row, one Real seal row, one sign-change row, and the structural rows.

The interval and map rows are the entry data. The located interval row I fixes the endpoint windows and interval cells, while F and M expose only a displayed continuity-modulus route for the requested tolerance.

The refinement and sign rows are then finite. Row B records the retained bracket and midpoint route, row S records the sign-change/readback witnesses, and row R supplies regular rational comparisons before the terminal seal E is read.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no other coordinate is present, the listed rows exhaust the local NameCert surface and exclude quotient equality, trichotomy, host real primitives, selected roots, and ambient completeness. □

Theorem 8016.3 (Constructive Darboux scope)**.** *Every* Real$^{\uparrow}$ *or intermediate-value consumer of an accepted* ConstructiveDarboux$^{\uparrow}$ *packet must first read the finite route*

$$\text{LocatedInterval}^{\uparrow},\ \text{ContinuousMap}^{\uparrow},\ M,\ B,\ S,\ \text{RegSeqRat}^{\uparrow},\ \text{Real}^{\uparrow}.$$

The route supplies interval-refinement, continuity-modulus, sign-change/readback, terminal Real seal, hsame, Cont, Pkg, *and local* NameCert *rows. It does not assert quotient real equality, real trichotomy, host real equality, a selected root, a host intermediate-value theorem, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the obligation basis in Definition 8016.2. It orders any consumer read through the located interval, continuous-map, continuity-modulus, refinement, sign, readback, Real seal, and structural rows.

The intermediate-value sibling contributes only the displayed bisection and sign-bracket route. Thus a consumer can inspect B and S as finite interval-refinement evidence, but it cannot replace them with a host theorem, a trichotomy decision, or an unbounded zero search.

The regular-rational and Real rows are terminal for this packet. Row R supplies finite comparisons for endpoint and midpoint values, while E seals the resulting Real-facing read only after I, F, M, B, S have been displayed.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Therefore every consumer remains inside the finite Darboux packet and receives none of the excluded host or quotient data. □

Falsifiable prediction: A $\mathsf{ConstructiveDarboux}^{\uparrow}$ consumer that reads a RealUp seal without displaying the located interval, continuous-map row, continuity modulus, finite refinement row, sign-change/readback row, RegSeqRat readback, transport, replay, provenance, and local naming rows refutes the carrier of Definition 8016.1.

Independence witness: locatedinterval, continuousmap, intermediatevalue, regseqrat, real: no listed sibling is bijective with $\mathsf{ConstructiveDarboux}^{\uparrow}$ because this carrier alone binds interval-refinement, continuity-modulus, sign-change/readback, terminal Real seal, transport, replay, provenance, and local NameCert rows as one finite Darboux route.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveDarboux}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveDarboux}^{\uparrow}$ packet is generated from a LocatedInterval row, a ContinuousMap row, a displayed continuity modulus, finite interval-refinement data, sign-change/readback rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveDarboux}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8016.1, Definition 8016.2, and Definition 8016.3 seed the finite Darboux interval-refinement packet for L10 real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveDarboux}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, real trichotomy, host real equality, a selected root, a host intermediate-value theorem, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, continuity-modulus exposure, sign-change/readback exposure, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming for BEDC.Derived.ConstructiveDarbouxUp.

Chapter 8017

A Concrete Naming Certificate for $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$

Following classical hyperbolic geometry, the $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$ packet records a three-side geodesic triangle as finite BEDC NameCert data inside the hyperbolic observer-expansion vision of chapter 9463. It sits between the boundary quadruple route of chapter 7659, the disk-geodesic triangle route of chapter 7891, the hyperbolic metric packet of chapter 3474, the geodesic boundary packet of chapter 3491, the cross-ratio packet of chapter 3948, and the disk carrier of chapter 377. The chapter names the triangle-level carrier needed between ideal quadrilateral data and later phase-geometry consumers; it does not complete a hyperbolic plane, quotient boundary endpoints, select global geodesic representatives, or identify the observer-expansion reading with physical spacetime.

Definition 8017.1 (Hyperbolic geodesic triangle carrier)**.** *A* $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (U, M, V, G, I, A, X, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the inspected disk window and its boundary reads.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *V is the three-boundary-vertex row, naming the ordered boundary or disk-facing vertices visible to the packet.*
4. *G records the three displayed* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *side rows joining the vertex pairs.*
5. *I is the incidence and cyclic-order ledger, checking that the side rows meet the displayed vertices in the stated order and that no fourth side is introduced.*
6. *A is the angle or orientation budget row, recording the finite comparison data available for triangle reads without importing an ambient smooth angle measure.*
7. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *readback row attached to the same ordered triple and to an auxiliary displayed comparison end when such a read is requested.*
8. *H records componentwise* hsame *transport for disk coordinates, metric reads, vertices, sides, incidence, angle or order data, cross-ratio readback, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a hyperbolic triangle request through $U, M, V, G, I, A, X, H, P, N$.*

10. *P is the* Pkg *provenance over* HyperbolicGeodesicTriangle$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBour HyperbolicCrossRatio$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{HyperbolicGeodesicTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, V, G, I, A, X, H, C, P, N$. It has no coordinate for completed hyperbolic planes, quotient ideal-boundary equality, selected global geodesics, ambient smooth angle structures, host cross-ratio fields, or physical observer dynamics.

Theorem 8017.2 (Hyperbolic geodesic triangle NameCert obligations)**.** *For every accepted* HyperbolicGeodesicTria *carrier*

$$T = (U, M, V, G, I, A, X, H, C, P, N),$$

the local NameCert$_{\text{HyperbolicGeodesicTriangle}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission for U, hyperbolic-metric visibility for M, three-vertex admission for V, three geodesic-side admissions for G, incidence and cyclic-order ledger exposure for I, finite angle or orientation budget exposure for A, finite cross-ratio readback for X, componentwise* hsame *transport for H, displayed* Cont *replay for C,* Pkg *provenance for P, and local naming for N.*

Proof. Project the carrier of Definition 8017.1. The disk and metric rows U, M are the only geometric source rows, and the vertex row V fixes the three displayed ends or disk-facing vertices.

Read the side and incidence data next. The row G lists exactly three geodesic sides, while I records their endpoint incidence and cyclic order, so a fourth side or detached boundary representative cannot be inserted.

The comparison rows are downstream. The row A stores only the finite angle or orientation budget available to the packet, and X stores only the finite cross-ratio readback attached to the same displayed triple and auxiliary comparison end.

Rows H, C, P, N transport, replay, source, and locally name the same packet. Since no carrier coordinate stores completed planes, quotient boundary equality, selected geodesics, ambient smooth angle structures, host cross-ratio fields, or physical observer dynamics, the listed rows exhaust the local NameCert obligations. □

Theorem 8017.3 (Hyperbolic geodesic triangle finite readback)**.** *Every triangle-level consumer of an accepted* HyperbolicGeodesicTriangle$^{\uparrow}$ *packet must read*

$$\text{UnitDisk}^{\uparrow}, \text{HyperbolicMetric}^{\uparrow}, \text{HyperbolicGeodesicBoundary}^{\uparrow}, \text{HyperbolicCrossRatio}^{\uparrow}$$

through the displayed rows $U, M, V, G, I, A, X, H, C, P, N$ before exporting a metric, incidence, angle-budget, or cross-ratio comparison. The export is a finite NameCert route between ideal quadrilateral data and later hyperbolic phase-geometry consumers; it is not a completed hyperbolic model, quotient boundary construction, selected geodesic representative, smooth angle theory, host field theorem, or physical observer trajectory.

Proof. Begin with the source rows U, M, V. They name the disk window, metric read, and three visible vertices used by the packet.

Use G, I to form the triangle spine. These rows display the three geodesic sides and their endpoint incidence before any comparison row is read.

Apply the finite comparison rows A, X. The angle or order budget and cross-ratio readback are attached to the same displayed triangle data and cannot be replaced by a host smooth angle measure or global cross-ratio field.

Finally replay the structural rows H, C, P, N. They preserve the finite source, side, incidence, comparison, provenance, and local naming data, so the exported triangle read remains inside the NameCert route stated above. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$ read must expose UnitDisk coordinates, HyperbolicMetric reads, three boundary vertices, three HyperbolicGeodesicBoundary side rows, incidence or cyclic-order data, finite angle or orientation budget rows, HyperbolicCross-Ratio readback, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: unitdisk, hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, hyperbolic-ideal-quadrilateral, poincare-disk-geodesic-triangle: $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds three displayed vertices, three side rows, incidence or order data, a finite angle or orientation budget, cross-ratio readback, transport, replay, provenance, and local naming in one triangle packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$ packet is generated from UnitDisk coordinate rows, HyperbolicMetric reads, three displayed boundary vertices, three HyperbolicGeodesicBoundary side rows, incidence or cyclic-order ledgers, finite angle or orientation budget rows, HyperbolicCrossRatio readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8017.1, Definition 8017.2, and Definition 8017.3 seed the finite triangle route over disk coordinates, hyperbolic metric rows, displayed geodesic sides, incidence ledgers, comparison budgets, cross-ratio readback, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic planes, quotient ideal-boundary equality, selected global geodesics, ambient smooth angle structures, host cross-ratio fields, physical observer dynamics, Lean verification, or bridge checking for $\mathsf{HyperbolicGeodesicTriangle}^{\uparrow}$.

Upgrade path: Obligation closure requires checked disk-coordinate admission, hyperbolic-metric visibility, three-vertex admission, geodesic-side exactness, incidence and cyclic-order stability, angle or orientation budget exposure, cross-ratio readback, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicGeodesicTriangleUp.

Chapter 8018

A Concrete Naming Certificate for $\mathsf{MarkovDecisionProcess}^{\uparrow}$

Following Bellman and Puterman on Markov decision processes, the $\mathsf{MarkovDecisionProcess}^{\uparrow}$ packet records a finite policy-transition-reward interface before any optimality theorem or measurable selector is visible. It sits after the transition-row surface of chapter 189, the normalized probability carrier of chapter 184, the Bellman recursion route of chapter 5875, and the action-ledger surface of chapter 290. The packet names displayed state rows, action rows, transition rows, reward rows, policy rows, Bellman-consumer routes, transports, replay, provenance, and local naming as BEDC data; it does not import optimal policy existence, measurable selection, infinite-horizon convergence, or host stochastic-process equality.

Definition 8018.1 (MarkovDecisionProcess carrier)**.** *A* $\mathsf{MarkovDecisionProcess}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, K, R, \Pi, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state-observation row read through the same normalized probability source used by* $\mathsf{ProbSpace}^{\uparrow}$ *consumers.*
2. *A is the finite action row whose visible choices are read through the action-ledger discipline of* $\mathsf{Policy}^{\uparrow}$*.*
3. *K is the one-step transition row, exposed as a Markov-transition packet before any Bellman consumer can inspect the next-state law.*
4. *R is the reward row, a finite value-facing ledger whose scalar endpoints are read as displayed* $\mathsf{Real}^{\uparrow}$ *names rather than as a host utility function.*
5. *Π is the displayed policy row, recording which action ledger is paired with each visible state row.*
6. *V is the Bellman-consumer boundary row, storing the finite continuation by which dynamic-programming readers consume S, A, K, R, Π.*
7. *H records componentwise* hsame *transports for the state, action, transition, reward, policy, Bellman-boundary, provenance, and local name rows.*
8. *C records the displayed* Cont *replay by which consumers read the probability source, Markov transition, policy ledger, reward ledger, Bellman boundary, transports, provenance, and local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{MarkovDecisionProcess}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, $\mathsf{DynamicProgramming}^{\uparrow}$, $\mathsf{Policy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovDecisionProcess}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their rowwise transports, continuation routes, package provenance, and local naming provenance.

Theorem 8018.2 (MarkovDecisionProcess NameCert obligations)**.** *For an accepted* $\mathsf{MarkovDecisionProcess}^{\uparrow}$ *carrier* $D = (S, A, K, R, \Pi, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovDecisionProcess}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state, action, transition, reward, policy, Bellman-boundary, transport, replay, provenance, and naming rows; source compatibility with* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{MarkovChain}^{\uparrow}$*; policy-ledger compatibility with* $\mathsf{Policy}^{\uparrow}$*; Bellman-consumer compatibility with* $\mathsf{DynamicProgramming}^{\uparrow}$*; componentwise* hsame *stability through* H*; displayed* Cont *replay through* C*; and non-escape from the finite packet.*

Proof. Project the displayed carrier. Its rows name the probability-sourced state surface, action ledger, one-step transition packet, reward ledger, policy row, Bellman-facing boundary, transport, replay, package, and local certificate.

Read the dependency rows in consumer order. The normalized probability source is visible before the Markov transition row, the policy ledger is visible before an action-dependent reward row is consumed, and the dynamic-programming boundary is exposed only after S, A, K, R, Π have been displayed.

The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a selected optimizer, measurable policy selector, infinite-horizon limit, or host stochastic-process equality, the local NameCert obligations remain exactly the listed finite rows. □

Theorem 8018.3 (MarkovDecisionProcess Bellman consumer boundary)**.** *Every Bellman or dynamic-programming consumer of an accepted* $\mathsf{MarkovDecisionProcess}^{\uparrow}$ *packet reads the value-facing boundary through the finite route*

$$S \longmapsto A \longmapsto K \longmapsto R \longmapsto \Pi \longmapsto V.$$

The route first displays the probability and transition data of chapter 184 and chapter 189, then the policy action ledger of chapter 290, and only then the Bellman-facing continuation consumed by chapter 5875.

Proof. Start with a Bellman-facing read of the boundary row V. The carrier makes V available only through the continuation replay that has already exposed the source state row, action row, transition row, reward row, and policy row.

The source and transition reads are finite: S is attached to the probability carrier and K to the Markov-transition surface. The action read is also finite: A and Π are policy-ledger rows, not an external decision rule or measurable selector.

Finally the reward row R and boundary row V are replayed as displayed certificate data. Thus a dynamic-programming consumer receives only the finite Bellman boundary over S, A, K, R, Π, V, with H, C, P, N supplying transport, replay, provenance, and naming. Nothing in the route exports optimal policy existence, measurable selection, infinite-horizon convergence, or host stochastic-process equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovDecisionProcess}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovDecisionProcess}^{\uparrow}$ packet is generated from finite rows for displayed states, actions, one-step Markov transitions, rewards, policy ledgers, Bellman-consumer boundaries, componentwise hsame transports, Cont replay, package provenance, and local naming. Bellman-facing consumers read the packet through the finite route S, A, K, R, Π, V before any dynamic-programming boundary is exposed.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovDecisionProcess}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the MarkovDecisionProcess carrier, NameCert obligation surface, and Bellman consumer boundary as finite BEDC rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovDecisionProcess}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: It does not claim optimal policy existence, measurable selection, infinite-horizon convergence, host stochastic-process equality, stochastic-process completion, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier formation, policy-transition-reward compatibility, Bellman consumer routing, componentwise transport, and non-escape for the displayed packet.

Chapter 8019

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$ packet records a finite located envelope for a constructive real approximation before any quotient equality, trichotomy, choice-selected limit, or ambient completeness claim is available. It sits over the dyadic tolerance core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval boundary of chapter 1709. The packet gives Real and Completion consumers a named finite envelope surface rather than a completed real construction.

Definition 8019.1 (Bishop located real envelope carrier)**.** *A* $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, Q, R, L, U, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying dyadic radii, endpoint comparisons, midpoint arithmetic, and slack witnesses for the inspected envelope.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule listing exactly which approximation windows are exposed.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows and dyadic tolerances.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, Q have been displayed.*
5. *L and U are lower and upper dyadic envelope rows, with their order and width checked inside D.*
6. *A is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *adjacency row certifying that the displayed lower and upper rows form a located interval envelope around the inspected real name.*
7. *H stores componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, Real seal, envelope endpoints, located interval row, replay, provenance, and local name.*
8. *C stores displayed* Cont *replay from a precision request through S, D, L, U, Q, A, R, H.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopLocatedRealEnvelope}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, Q, R, L, U, A, H, C, P, N$. It has no coordinate for quotient equality of real names, trichotomy, countable choice, selected limits, supremum, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8019.2 (Bishop located real envelope NameCert obligations)**.** *For every accepted* BishopLocatedRealEnve *carrier $E = (D, S, Q, R, L, U, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window scheduling, regular rational readback, lower and upper envelope exposure, located interval adjacency, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming. Every Real-facing or Completion-facing read must pass through these rows before consuming the seal R.*

Proof. Project the carrier of Definition 8019.1. The analytic rows are D, S, Q, R, L, U, A, and the structural rows are H, C, P, N.

The dyadic and stream stages are finite. The row D supplies endpoint arithmetic and tolerance comparisons, while S names the inspected windows. The readback row Q is therefore attached to displayed observations rather than to an ambient completed real.

The envelope rows are downstream of those observations. The lower and upper rows L, U are checked by D, and the located interval row A records their envelope adjacency before the terminal seal R is available.

Finally H, C, P, N only transport, replay, source, and name the same route. Since no omitted coordinate stores quotient equality, trichotomy, countable choice, selected limits, supremum, open-cover compactness, or host completeness, the local NameCert surface is exactly the displayed finite envelope surface. □

Theorem 8019.3 (Bishop located real envelope finite-window readback)**.** *Every accepted* BishopLocatedRealEnvelo *read of a real approximation factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by $D, S, Q, A, R, H, C, P, N$, with the lower and upper envelope rows L, U read before the terminal seal. Thus the packet exposes located real approximation without quotient equality, trichotomy, countable choice over precision requests, selected limits, supremum, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer request for an accepted envelope. The finite schedule S identifies the inspected windows, and the dyadic ledger D identifies the tolerances and endpoint comparisons visible at those windows.

The regular readback Q is the only row that turns those displayed windows into rational approximation data. The lower and upper envelope rows L, U are then read against the same dyadic ledger, so the packet has no independent real-order oracle.

The located interval row A binds the two endpoint rows as a finite located envelope. Only after that adjacency is displayed may the terminal seal R be consumed.

Transport, replay, provenance, and local naming preserve the same dependency route. Hence every read stays inside the finite-window envelope and cannot export the excluded quotient, choice, order, compactness, or completeness principles. □

Falsifiable prediction: Every $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal has already exposed $D, S, Q, L, U, A, H, C, P, N$ and the finite dyadic, stream, regular-readback, and located-interval route named by this chapter.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedrealinterval: BishopLocatedRealEnvel is not bijective to any listed sibling because it jointly carries a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, lower and upper envelope rows, a LocatedRealInterval adjacency row, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealEnvelope}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readback, lower and upper envelope rows, a LocatedRealInterval adjacency row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8019.1, Definition 8019.2, and Definition 8019.3 seed the finite located-envelope route from dyadic tolerance rows through finite windows, regular readback, located interval adjacency, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealEnvelope}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real names, trichotomy, countable choice, selected limits, supremum, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window scheduling, regular rational readback, envelope exactness, located interval adjacency, terminal Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealEnvelopeUp.

Chapter 8020

A Concrete Naming Certificate for $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$

Following Littlewood and Paley on dyadic frequency decompositions, the $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ packet records the block-level inspection surface used before a square-function, Sobolev, or step-function approximation consumer reads a Littlewood–Paley estimate. It refines the aggregate Littlewood–Paley carrier of chapter 7025, keeps dyadic endpoints on the $\mathsf{DyadicRatCore}^{\uparrow}$ route of chapter 291, seals scalar comparisons only through chapter 38, and sits beside the finite step-function approximation packet of chapter 8012. The packet names one displayed dyadic block and its local recomposition ledger; it does not assert Fourier inversion, unconditional series convergence, quotient function equality, a selected representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8020.1 (Littlewood–Paley dyadic-block carrier)**.** *A* $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, P, R, N, H, C, G)$$

with the following displayed rows:

1. *S is the dyadic scale row, recording the selected block scale and its* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data.*
2. *W is the frequency-window row, recording the annulus, cap, or finite window occupied by the block.*
3. *P is the projection row, recording the finite projection probe attached to W.*
4. *R is the recomposition and norm ledger, recording how this block contributes to the local square-function, Sobolev, or step-function readback before any global estimate is cited.*
5. *N is the local* $\mathsf{NameCert}_{\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}}$ *row.*
6. *H records componentwise* hsame *transport for the scale, window, projection, ledger, naming, replay, and provenance rows.*
7. *C records displayed* Cont *replay from the selected scale through the window, projection, recomposition ledger, local name, and transport rows.*
8. *G is the* Pkg *provenance over* $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$, $\mathsf{LittlewoodPaley}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StepFunctionUniformApproximation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

The classifier compares accepted packets only through S, W, P, R, N, H, C, G. It has no coordinate for an infinite frequency partition, an ambient Fourier transform, equality of completed function-space representatives, a global square-function theorem, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) BEDC.Derived.LittlewoodPaleyDyadicBlockUp

Theorem 8020.2 (Littlewood–Paley dyadic-block NameCert obligations)**.** *For every accepted* $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ *carrier* $B = (S, W, P, R, N, H, C, G)$*, the local* $\mathsf{NameCert}_{\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic scale row* S*, frequency-window row* W*, projection row* P*, recomposition and norm ledger* R*, local naming row* N*, componentwise transport row* H*, displayed replay row* C*, and provenance row* G*. A downstream* $\mathsf{LittlewoodPaley}^{\uparrow}$*, Sobolev, square-function, or step-function consumer may read the block only after those rows are visible.*

Proof. Project the carrier of Definition 8020.1. The rows S, W, P, R, N, H, C, G are the full accepted packet, so a block read starts from a displayed dyadic scale and a displayed frequency window.

The analytic rows are local. The projection row P belongs to the same window W, and the recomposition ledger R records only this block's contribution before any aggregate Littlewood–Paley read is formed.

The structural rows preserve the same packet. Since H, C, G, N transport, replay, source, and name only the displayed block, the local certificate cannot introduce an infinite partition, host Fourier inversion, quotient representative equality, or ambient real-completeness row. □

Theorem 8020.3 (Littlewood–Paley dyadic-block frequency-window route)**.** *Every frequency-window read of an accepted* $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ S,\ W,\ P,\ R,\ N,\ H,\ C,\ G$$

before it can be handed to the aggregate $\mathsf{LittlewoodPaley}^{\uparrow}$ *carrier or to a step-function or Sobolev-facing consumer. The route exports a finite dyadic block, not Fourier inversion, unconditional recomposition, quotient function equality, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the dyadic row S. It fixes the block scale through displayed $\mathsf{DyadicRatCore}^{\uparrow}$ data, so the window cannot be read as an unbounded frequency partition.

Next read W, P, R. The window row names the finite frequency support, the projection row names the local probe, and the recomposition ledger records the block's contribution before an aggregate consumer is allowed to continue.

Finally use N, H, C, G. These rows name, transport, replay, and source the same finite packet. Thus the handoff remains a local dyadic-block route and supplies no coordinate for the excluded analytic endpoints. □

Falsifiable prediction: An accepted $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ read is refuted by any square-function or Sobolev-facing consumer that consumes the projection row before displaying the dyadic scale, frequency window, recomposition ledger, transport, replay, provenance, and local naming rows.

Independence witness: littlewood-paley, dyadicratcore, real, step-function-uniform-approximation, sobolev: none is bijective to $\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}$ because this packet jointly carries one dyadic scale, one frequency window, one projection row, a local recomposition and norm ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LittlewoodPaleyDyadicBlock}^{\uparrow}}$.

Constructive story: A LittlewoodPaleyDyadicBlock$^{\uparrow}$ packet is generated from one dyadic scale row, one frequency-window row, one projection row, one recomposition and norm ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LittlewoodPaleyDyadicBlock}\uparrow}$, seed)

Scope closed: Definition 8020.1, Definition 8020.2, and Definition 8020.3 seed the finite block-level Littlewood–Paley route over dyadic scale, frequency window, projection, recomposition, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{LittlewoodPaleyDyadicBlock}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, Fourier inversion, unconditional dyadic recomposition, quotient function equality, selected representatives, global square-function estimates, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, frequency-window exactness, projection exposure, recomposition-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.LittlewoodPaleyDyadicBlockUp.

Chapter 8021

A Concrete Naming Certificate for $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$

The $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ packet realizes the conditional MetaCIC confluence boundary of chapter 9378 as a finite residual-join window. It sits beside the parallel-diamond frontier of chapter 4238, the residual-substitution compatibility boundary of chapter 4370, the residual-join determinacy packet of chapter 5970, the local-join ledger of chapter 4126, and the finite residual-confluence window of chapter 7752. The packet keeps the retained premise visible while a bounded checker inspects a local join; it does not prove full confluence, subject reduction, residual-substitution compatibility, evaluator equality, quotient term equality, or kernel consistency.

Definition 8021.1 (Finite residual-join window carrier)**.** *A* $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (P, L, O, B, H, C, G, N)$$

with the following displayed rows:

1. *P is the retained residual-substitution premise row consumed by the conditional confluence boundary.*
2. *L is the local residual-join row, recording the finite peak, candidate join, and residual window inspected by the packet.*
3. *O is the obstruction row, retaining the unavailable global endpoints and preventing a local join from being reread as premise-free confluence.*
4. *B is the bounded checker row, recording the finite branch check performed inside the window.*
5. *H records componentwise* hsame *transport for the premise, local join, obstruction, checker, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from the retained premise through the local join, obstruction, bounded checker, transport, provenance, and local name.*
7. *G is the* Pkg *provenance over* $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, MetaCICResidualSub $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$, $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, $\mathsf{FiniteResidualConfluenceWindow}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteResidualJoinWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through P, L, O, B, H, C, G, N. It has no coordinate for discharging P, erasing O, promoting L to global Church–Rosser, or importing subject reduction, evaluator equality, quotient equality, or kernel consistency.

Lean4 (statement) `BEDC.Derived.FiniteResidualJoinWindowUp`

Theorem 8021.2 (Finite residual-join window NameCert obligations)**.** *For every accepted* FiniteResidualJoinWindo *carrier $J = (P, L, O, B, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteResidualJoinWindow}^{\uparrow}}$ *surface consists exactly of retained-premise exposure through P, local residual-join exposure through L, obstruction retention through O, bounded-checker exposure through B, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8021.1. The retained premise, local join, obstruction, and bounded checker rows are displayed before the structural rows can be consumed.

The confluence-facing content is local. The row L records only the finite residual join inside the window, while P and O remain visible as premise and obstruction rows for the same packet.

The rows H, C, G, N transport, replay, source, and name that finite packet. Since no row discharges the premise or removes the obstruction, the NameCert surface cannot export full confluence, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 8021.3 (Finite residual-join window conditional route)**.** *Every confluence-audit consumer of an accepted* $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ *packet may read the local residual join only through*

$$P, L, O, B, H, C, G, N.$$

The retained premise P, obstruction row O, bounded checker B, transport H, replay C, provenance G, and local name N remain attached to the local join L before any conditional handoff is read.

Proof. Start with the retained premise row P. It records the substitution-sensitive input that the MetaCIC confluence audit map keeps visible at the conditional boundary.

Next inspect L, O, B. The local residual join is bounded by the displayed window, the obstruction row records the unavailable global endpoints, and the checker row limits the read to the finite branch inspection.

Finally read H, C, G, N. These rows preserve the same finite window under transport, replay, provenance, and local naming, so the handoff remains a retained-premise local join rather than a full confluence theorem. □

Falsifiable prediction: An accepted $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ packet is refuted by any confluence consumer that reads a local residual join while erasing the retained premise row, obstruction row, bounded checker, transport, replay, provenance, or local naming row.

Independence witness: metacic-parallel-diamond-frontier, metacic-residual-substitution-compatibility, metacic-residual-join-determinacy, metacic-local-join-ledger, finite-residual-confluence-window: no listed sibling is bijective to $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ because this packet jointly carries retained premise, local join, obstruction, bounded checker, transport, replay, provenance, and local NameCert rows. The Lean follow-up must provide Nontrivial $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ and FieldFaithful FiniteResidualJoi instances.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteResidualJoinWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$ packet is generated from a retained residual-substitution premise, local residual-join row, obstruction row, bounded checker row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteResidualJoinWindow}\uparrow}$, seed)

Scope closed: Definition 8021.1, Definition 8021.2, and Definition 8021.3 seed the finite residual-join window over retained premise, local join, obstruction, bounded checker, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteResidualJoinWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, residual-substitution discharge, full confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained-premise exposure, local-join exactness, obstruction retention, bounded-checker exposure, transport, replay, provenance, and local naming for BEDC.Derived.FiniteResidualJoinWindow

Chapter 8022

A Concrete Naming Certificate for $\mathsf{OscillationSeparation}^{\uparrow}$

Following Hardy and Bishop–Bridges, the $\mathsf{OscillationSeparation}^{\uparrow}$ packet records the finite lower-envelope and upper-envelope separation data carried by a bounded real sequence before any compactness or completeness principle is invoked. It binds the lower-tail endpoint of chapter 1553, the upper-tail endpoint of chapter 7101, the oscillation-facing row discipline of chapter 1839, the endpoint-gap packet of chapter 8002, and the terminal real-name seal of chapter 38. The packet is an L10 lower/upper envelope separation carrier; it does not assert compactness, host supremum or infimum selection, quotient sequence equality, trichotomy, selected subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8022.1 (Oscillation separation carrier)**.** *An* $\mathsf{OscillationSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, U, G, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-envelope row, read through the* $\mathsf{Liminf}^{\uparrow}$ *tail lower-cut surface for the displayed bounded real sequence.*
2. *U is the upper-envelope row, read through the* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *finite upper-window surface for the same displayed bounded real sequence.*
3. *G is the gap row, read through* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *as the finite separation request between the two envelope rows.*
4. *D is the dyadic tolerance ledger naming the rational budgets used when the gap row is replayed at the* $\mathsf{Real}^{\uparrow}$ *boundary.*
5. *H records componentwise* hsame *transport for the lower envelope, upper envelope, gap, dyadic tolerance, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an oscillation-separation query through L, U, G, D, H.*
7. *P is the* Pkg *provenance over* $\mathsf{OscillationSeparation}^{\uparrow}$*,* $\mathsf{Liminf}^{\uparrow}$*,* $\mathsf{BoundedRealLimsup}^{\uparrow}$*,* $\mathsf{RealOscillation}^{\uparrow}$*,* $\mathsf{LimsupLiminfGap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through L, U, G, D, H, C, P, N. No coordinate stores a compactness theorem, ambient completeness, host supremum or infimum, quotient equality of streams, selected subsequences, trichotomy, or a zero-gap convergence conclusion.

Theorem 8022.2 (Oscillation separation seed obligations)**.** *For every accepted* $\mathsf{OscillationSeparation}^{\uparrow}$ *carrier*

$$O = (L, U, G, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OscillationSeparation}^{\uparrow}}$ *surface contains the following seed obligations:*

1. *lower-envelope admission, requiring L to be displayed as a* $\mathsf{Liminf}^{\uparrow}$ *row before the packet can answer a separation query;*
2. *upper-envelope admission, requiring U to be displayed as a* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *row over the same bounded sequence source before it can be compared with L;*
3. *gap exactness, requiring G to read only the paired lower and upper envelope rows through the* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *packet;*
4. *dyadic replay, requiring every comparison budget in D to be replayed before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary is consumed;*
5. *ledger nonescape, requiring H, C, P, N to transport, replay, source, and name only the rows of O.*

Proof. Project the carrier of Definition 8022.1. The analytic rows are exactly the lower envelope L, upper envelope U, gap row G, and dyadic ledger D.

The lower and upper envelope obligations are row admissions. The packet has no independent sequence coordinate, so the two endpoints must be read through the sibling $\mathsf{Liminf}^{\uparrow}$ and $\mathsf{BoundedRealLimsup}^{\uparrow}$ surfaces that already name the bounded sequence source.

The gap and dyadic obligations are the next route in the carrier. The row G can compare only the displayed envelope rows, and D names the finite rational budgets used to replay that comparison at the real-name boundary.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier inventory has no compactness, completeness, quotient, subsequence, trichotomy, or convergence coordinate, the listed seed obligations exhaust the local surface. □

Theorem 8022.3 (Oscillation separation envelope route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{OscillationSeparation}^{\uparrow}$ *carrier factors through the finite route*

$$L \longmapsto U \longmapsto G \longmapsto D \longmapsto H, C, P, N$$

before reading the displayed lower/upper envelope separation. The route supplies the $\mathsf{Liminf}^{\uparrow}$ *lower envelope, the* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *upper envelope, the* $\mathsf{LimsupLiminfGap}^{\uparrow}$ *separation row, dyadic replay, componentwise transport, continuation replay, provenance, and local naming; it supplies no compactness theorem, host endpoint selection, quotient stream equality, selected subsequence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier. The consumer first receives the lower envelope row L and upper envelope row U, and these rows are visible only as sibling endpoint certificates.

The separation read then passes through G. Because G is the only gap row in the carrier, it can bind only the displayed lower and upper envelopes, not a host supremum or infimum.

The dyadic ledger D supplies the finite comparison budgets. The structural rows H, C, P, N preserve the same route by transport, replay, provenance, and local naming, so no omitted global principle can enter the consumer surface. □

Theorem 8022.4 (Oscillation separation Real-boundary nonescape)**.** *An accepted* $\mathsf{OscillationSeparation}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *boundary only after the lower envelope, upper envelope, gap, and dyadic tolerance rows have been displayed. In particular, the packet does not export compactness, host supremum or infimum selection, quotient sequence equality, trichotomy, selected subsequences, zero-gap convergence, Lean verification, or a standard bridge.*

Proof. The only Real-facing content in the carrier is reached through the rows L, U, G, D. The lower and upper envelope rows name the endpoint data, the gap row binds them into a separation request, and the dyadic ledger names the finite comparison budgets.

Replay through C and transport through H cannot add coordinates absent from O. Provenance and local naming through P, N source the packet but do not create compactness, completeness, quotient, subsequence, trichotomy, convergence, verification, or bridge data. □

Falsifiable prediction: An accepted $\mathsf{OscillationSeparation}^{\uparrow}$ read is refuted if the gap row G is consumed while either the lower-envelope row L, the upper-envelope row U, or the dyadic tolerance ledger D is absent.

Independence witness: liminf, bounded-real-limsup, real-oscillation, limsup-liminf-gap, real: no listed sibling supplies the whole packet $O = (L, U, G, D, H, C, P, N)$, because each sibling contributes an endpoint, oscillation route, gap read, or terminal Real boundary while this chapter binds them into one finite separation carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationSeparation}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OscillationSeparation}^{\uparrow}$ packet is generated from a Liminf lower-envelope row, a bounded Real limsup upper-envelope row, a limsup-liminf gap row, a dyadic tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8022.1, Definition 8022.2, Definition 8022.3, and Definition 8022.4 seed the finite lower/upper envelope separation carrier for bounded real-sequence oscillation reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, host supremum or infimum selection, quotient sequence equality, trichotomy, selected subsequences, zero-gap convergence, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked lower-envelope admission, upper-envelope admission, gap exactness, dyadic replay, transport, replay, provenance, and local naming for BEDC.Derived.OscillationSeparationUp.

Chapter 8023

A Concrete Naming Certificate for $\mathsf{DistributionSpace}^{\uparrow}$

Following Schwartz and Treves distribution theory, the $\mathsf{DistributionSpace}^{\uparrow}$ packet records the finite test-action data needed to name a distribution-space interface without importing a host locally convex space, quotient of test functions, or selected representative of a distribution. It sits between the test-function and Schwartz-space carriers of chapter 6945 and chapter 5877, the weak-derivative boundary of chapter 2210, and the scalar readback rows of chapter 301 and chapter 38.

Definition 8023.1 (Distribution-space carrier)**.** *A* $\mathsf{DistributionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, A, B, U, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{TestFunction}^{\uparrow}$ *window row, listing the finite test probes, support windows, smoothness tags, and request indices visible to the packet.*
2. *A is the scalar action ledger, recording the displayed action value for each test probe and the finite dependency route by which the action is read.*
3. *B is the* $\mathsf{WeakDerivative}^{\uparrow}$ *boundary row, recording which candidate derivative actions may consume the same test-action ledger.*
4. *U is the source row over* $\mathsf{SchwartzSpace}^{\uparrow}$ *and* $\mathsf{TestFunction}^{\uparrow}$*, exposing the finite test carrier rather than a host test-function equality relation.*
5. *R is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *scalar readback row for action values, comparison tolerances, and terminal scalar boundaries.*
6. *H records componentwise* hsame *transport for test windows, scalar actions, weak-derivative boundaries, source rows, scalar readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from test selection through scalar action, derivative-boundary inspection, source readback, and scalar comparison.*
8. *P is the* Pkg *provenance over* $\mathsf{DistributionSpace}^{\uparrow}$, $\mathsf{TestFunction}^{\uparrow}$, $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DistributionSpace}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for a host distribution space, quotient equality of test functions, selected distribution representative, countable-choice selector, global LF-space topology, or ambient Real completeness theorem.

Theorem 8023.2 (Distribution-space NameCert obligations)**.** *For every accepted* $\mathsf{DistributionSpace}^{\uparrow}$ *carrier* $S = (T, A, B, U, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DistributionSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, finite test-window admission through* T*, scalar-action readback through* A*, weak-derivative boundary exposure through* B*,* $\mathsf{SchwartzSpace}^{\uparrow}$ *and* $\mathsf{TestFunction}^{\uparrow}$ *source admission through* U*,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *scalar readback through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier $S = (T, A, B, U, R, H, C, P, N)$. The displayed mathematical rows are the test-window row, scalar action ledger, weak-derivative boundary, source row, and scalar readback row.

The structural rows transport, replay, source, and name the same displayed data. Since the carrier has no coordinate for a host distribution space, quotient test-function equality, selected representative, choice selector, global LF-space topology, or ambient Real completeness, none of those objects can discharge a local NameCert obligation.

Thus the obligation surface is exactly the finite packet displayed above. □

Theorem 8023.3 (Distribution-space test-action handoff)**.** *Every* $\mathsf{WeakDerivative}^{\uparrow}$ *or Sobolev-facing consumer of an accepted* $\mathsf{DistributionSpace}^{\uparrow}$ *packet reads the finite route*

$$T \longmapsto A \longmapsto B \longmapsto U \longmapsto R \longmapsto H, C, P, N$$

before it consumes any distribution-space action. The handoff exposes test windows, scalar action values, derivative-boundary access, source rows, scalar readback, transport, replay, provenance, and local naming, and it exports no host distribution-space object.

Proof. Start with a consumer request for distribution-space action data. The request first opens the finite $\mathsf{TestFunction}^{\uparrow}$ row T, so the visible tests and support windows are fixed before any scalar action is read.

The scalar action ledger A is then read against the same tests. The weak-derivative boundary row B records which derivative-facing reads may consume that ledger, while U keeps the source tied to the displayed $\mathsf{SchwartzSpace}^{\uparrow}$ and $\mathsf{TestFunction}^{\uparrow}$ rows.

Finally the scalar readback row R and the structural rows H, C, P, N preserve and name the route. No host distribution space or quotient equality can be exported, because such data never appears in the carrier. □

Theorem 8023.4 (Distribution-space weak-derivative boundary)**.** *Any* $\mathsf{WeakDerivative}^{\uparrow}$ *read that consumes a* $\mathsf{DistributionSpace}^{\uparrow}$ *packet must expose the same test-action row, scalar action ledger, weak-derivative boundary row,* $\mathsf{SchwartzSpace}^{\uparrow}$ *or* $\mathsf{TestFunction}^{\uparrow}$ *source row,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *scalar readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before the read can be handed to* $\mathsf{Sobolev}^{\uparrow}$*.*

Proof. Use Definition 8023.3 to obtain the ordered route through $T, A, B, U, R, H, C, P, N$. The weak-derivative consumer cannot reach B without first reading the test windows and scalar action ledger.

The scalar readback row R supplies the regular-rational and Real-facing boundary for the action values. The transport and replay rows keep the same test-action ledger visible when the packet is passed toward a Sobolev-facing consumer.

Therefore any Sobolev handoff through $\mathsf{WeakDerivative}^{\uparrow}$ must still display the distribution-space rows named in the statement. The absent host distribution-space, quotient, representative, choice, topology, and ambient completeness objects remain outside the boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistributionSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{DistributionSpace}^{\uparrow}$ packet is generated from finite TestFunctionUp windows, a scalar action ledger, a WeakDerivativeUp boundary, SchwartzSpaceUp and TestFunctionUp source rows, RegSeqRatUp and RealUp scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistributionSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8023.1, Definition 8023.2, Definition 8023.3, and Definition 8023.4 seed the finite distribution-space interface over test actions, scalar readback, weak-derivative boundaries, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistributionSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution spaces, quotient test-function equality, selected representatives, countable choice, global LF-space topology, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-action exactness, scalar readback stability, weak-derivative boundary nonescape, and TasteGate field faithfulness for BEDC.Derived.DistributionSpaceUp.

Chapter 8024

A Concrete Naming Certificate for $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$

Following Hardy–Wright on the Stern–Brocot tree and Bishop–Bridges on finite rational approximation, the $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$ packet records a finite dyadic approximation window cut out by a displayed Stern–Brocot bracket. It sits after the tree route of chapter 1704, the bracketing packet of chapter 1431, the dyadic ledger of chapter 291, the rational source of chapter 37, the finite readback route of chapter 126 and chapter 301, and the terminal real-name seal of chapter 38. The packet names only inspected finite windows; it does not import infinite continued fractions, quotient real equality, trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8024.1 (Stern–Brocot dyadic approximation carrier)**.** *A* $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, B, D, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{SternBrocot}^{\uparrow}$ *branch-prefix row, recording the left/right address and inspected mediant route.*
2. *B is the* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *bracket row, exposing the lower and upper rational endpoints delivered by the branch-prefix route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested dyadic scale, endpoint-width comparison, and admissible error window.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *dyadic candidate row extracted from the displayed bracket and dyadic ledger.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window through which the bracket and candidate rows are read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from B, D, Q, W after the Stern–Brocot branch row has been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the rational bracket, dyadic candidate, finite window, and regular rational rows have been read.*
8. *H records componentwise* hsame *transport for the branch prefix, bracket, dyadic ledger, candidate, window, readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S, B, D, Q, W \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotDyadicApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, D, Q, W, R, E, H, C, P, N$. It has no coordinate for infinite continued fractions, quotient real equality, host trichotomy, a selected real representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SternBrocotDyadicApproximationUp`

Theorem 8024.2 (Stern–Brocot dyadic approximation NameCert obligations)**.** *For every accepted* $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$ *carrier*

$$A = (S, B, D, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SternBrocotDyadicApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Stern–Brocot branch prefix, rational bracket, dyadic tolerance ledger, rational candidate, finite window, regular rational handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *branch-bracket exactness, requiring B to expose only rational endpoints reached by the finite branch-prefix row S;*
3. *dyadic-window compatibility, requiring D to bind the requested dyadic scale to the displayed bracket before the candidate row Q is read;*
4. *finite readback discipline, requiring W and R to preserve the same branch, bracket, tolerance, and candidate rows before the terminal seal E is available;*
5. *transport, replay, provenance, and naming exhaustion through H, C, P, N;*
6. *nonescape from infinite continued fractions, quotient real equality, host trichotomy, selected representatives, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8024.1. The accepted packet displays exactly the Stern–Brocot branch prefix, rational bracket, dyadic tolerance ledger, rational candidate, finite window, regular readback, Real seal, and structural rows.

Branch-bracket exactness reads S and B together. The bracket is not a host real interval; it is the finite rational endpoint pair reached by the inspected Stern–Brocot path.

Dyadic-window compatibility then reads D and Q over the same bracket. The dyadic ledger supplies the scale and endpoint-width comparison used to admit the rational candidate, so the candidate cannot be detached from the displayed branch.

The readback discipline follows the route through W, R, E. A consumer first sees the finite observation window, then the regular rational handoff, and only then the terminal Real seal.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for infinite continued fractions, quotient equality, trichotomy, selected representatives, countable choice, or ambient completeness, the listed obligations exhaust the local certificate. □

Theorem 8024.3 (Stern–Brocot dyadic approximation rational bracket handoff)**.** *Every* $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$ *packet must read through the finite handoff*

$$\mathsf{SternBrocot}^{\uparrow},\ \mathsf{SternBrocotApproximation}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff supplies a displayed rational bracket, a dyadic tolerance ledger, a rational candidate row, finite observation windows, regular rational readback, and a terminal Real seal. It supplies no infinite continued-fraction selector, quotient-real representative, host trichotomy, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from an accepted carrier $A = (S, B, D, Q, W, R, E, H, C, P, N)$. The branch row S and bracket row B are the only source of rational endpoint data.

The dyadic and rational rows D, Q are downstream of that bracket. Thus a rational consumer must read the displayed branch and bracket before it can consume the dyadic candidate.

The finite observation route is then W, R, E. The window row records what was inspected, the regular rational row records the readback, and the Real seal is terminal with respect to that readback.

Transport, replay, provenance, and local naming preserve the same route. Therefore every rational-bracket handoff remains inside the finite Stern–Brocot and dyadic window packet and cannot export any of the excluded real-completeness or choice data. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotDyadicApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SternBrocotDyadicApproximation}^{\uparrow}$ packet is generated from a finite SternBrocot branch prefix, a SternBrocotApproximation bracket, a DyadicRatCore tolerance ledger, a RatUp dyadic candidate, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotDyadicApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8024.1, Definition 8024.2, and Definition 8024.3 seed the finite Stern–Brocot dyadic approximation window over displayed SternBrocot, DyadicRatCore, Rat, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SternBrocotDyadicApproximation}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SternBrocotDyadicApproximationUp`

Bridge status: none

Not claimed: This chapter does not close infinite continued fractions, quotient real equality, trichotomy, selected real representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean proof obligations, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-bracket exactness, dyadic-window compatibility, finite readback discipline, transport, replay, provenance, local naming, and rational bracket handoff for BEDC.Derived.SternBrocotDyadicApproximationUp.

Chapter 8025

A Concrete Naming Certificate for $\mathsf{DenjoyCarleman}^{\uparrow}$

Following Denjoy and Carleman on quasianalytic classes, the $\mathsf{DenjoyCarleman}^{\uparrow}$ packet records the finite derivative-growth surface by which a smooth real function is read through displayed weight data before any quasianalytic boundary claim is exported. It sits beside the $\mathsf{Calculus}^{\uparrow}$ dossier of chapter 3632, the derivative rows of chapter 122, the compact interval fixed-point boundary of chapter 1437, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a constructive analysis carrier for derivative-growth bookkeeping; it is not a selected smooth function space, a host equality of functions, a completed Frechet topology, or a proof of the classical quasianalyticity theorem.

Definition 8025.1 (Denjoy–Carleman carrier)**.** *A* $\mathsf{DenjoyCarleman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, M, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{CompactInterval}^{\uparrow}$ *row on which all derivative-growth reads are localized.*
2. *F is the* $\mathsf{Calculus}^{\uparrow}$ *function-observation row over the same interval data.*
3. *M is the finite weight row recording the displayed Denjoy–Carleman control sequence entries inspected by the packet.*
4. *G is the derivative-growth ledger tying the first finitely many* $\mathsf{Derivative}^{\uparrow}$ *rows to the corresponding entries of M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for finite rational bounds on the displayed derivative windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, F, M, G, R have been replayed.*
7. *H records componentwise* hsame *transport for interval, function, weight, derivative-growth, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a derivative-growth request through $I, F, M, G, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DenjoyCarleman}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{CompactInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyCarleman}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows, their displayed transports, continuation replay, provenance, and local naming. It has no coordinate for quotient function equality, selected Taylor coefficients, ambient smooth-space completeness, classical divergence tests, countable choice, or bridge checking.

Theorem 8025.2 (Denjoy–Carleman NameCert obligations)**.** *For every accepted* $\mathsf{DenjoyCarleman}^{\uparrow}$ *carrier* $D = (I, F, M, G, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DenjoyCarleman}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed compact interval, function-observation, weight, derivative-growth, readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval exactness, requiring every derivative-growth read to consume the same* $\mathsf{CompactInterval}^{\uparrow}$ *row* I *before it inspects* $F, M, G,$ *or* R*;*
3. *derivative-growth exactness, requiring the* $\mathsf{Derivative}^{\uparrow}$ *rows in* G *to be read only with the displayed finite weight row* M *and rational readback row* R*;*
4. *Real handoff, requiring the terminal seal* E *to remain downstream of* I, F, M, G, R *and the structural rows* H, C, P, N*;*
5. *non-escape from quotient function equality, selected Taylor coefficients, completed smooth-space topology, classical quasianalyticity criteria, countable choice, Lean verification, and bridge checking.*

No Denjoy–Carleman obligation is stored outside those rows.

Proof. Project the carrier of Definition 8025.1. The projection exposes exactly the compact interval, function-observation, weight, derivative-growth, readback, Real seal, transport, replay, provenance, and local naming rows.

The interval and derivative-growth obligations are finite. A read enters through I, inspects the function-observation row F, then reads only the displayed weight entries in M and the matching derivative-growth ledger G before obtaining rational bounds through R.

Finally the Real handoff is supplied by E, H, C, P, N. Since the carrier has no coordinate for quotient functions, selected Taylor data, completed smooth topologies, classical quasianalytic criteria, choice schedules, Lean targets, or bridge rows, the listed rows exhaust the local NameCert surface. □

Theorem 8025.3 (Denjoy–Carleman derivative-growth route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyCarleman}^{\uparrow}$ *carrier factors through the finite route*

$$I, F, M, G, R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a compact interval row, a calculus function-observation row, displayed finite weight data, derivative-growth rows, rational readback, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It receives no quotient function equality, selected Taylor expansion, completed smooth-space topology, countable-choice schedule, or ambient quasianalyticity theorem.

Proof. Begin with a consumer of the packet. Carrier admission gives access to the derivative-growth ledger only as a row tied to the same compact interval and calculus observation rows.

Replay the continuation row C. The request reads I and F, then consults the displayed weight row M and the derivative-growth ledger G before it obtains rational readback through R and the terminal seal E.

The structural rows H, P, N preserve the same finite route under transport and provenance. No remaining carrier row can provide quotient function equality, a selected Taylor expansion, a completed smooth-space topology, a choice schedule, or an ambient quasianalyticity theorem. □

Falsifiable prediction: An accepted $\mathsf{DenjoyCarleman}^{\uparrow}$ read is refuted if a derivative-growth consumer reaches the terminal Real seal without displaying the CompactInterval row, Calculus observation row, finite weight row, derivative-growth ledger, RegSeqRat readback, componentwise transport, replay, provenance, and local NameCert row.

Independence witness: calculus, derivative, compact-interval, regseqrat, real: none alone is bijective with $\mathsf{DenjoyCarleman}^{\uparrow}$ because the packet jointly records localized function observation, finite Denjoy–Carleman weight data, derivative-growth bounds, rational readback, terminal Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyCarleman}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyCarleman}^{\uparrow}$ packet is generated from a CompactInterval row, a Calculus function-observation row, a finite Denjoy–Carleman weight row, derivative-growth rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyCarleman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8025.1, Definition 8025.2, and Definition 8025.3 seed the finite Denjoy–Carleman derivative-growth route over compact intervals, calculus observations, derivative rows, rational readback, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyCarleman}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the classical Denjoy–Carleman quasianalyticity theorem, quotient function equality, selected Taylor coefficients, completed smooth-space topology, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exactness, derivative-growth exactness, weight-row accountability, rational readback exactness, Real seal handoff, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DenjoyCarlemanUp.

Chapter 8026

A Concrete Naming Certificate for $\mathsf{UniformPartitionVariation}^{\uparrow}$

Following Jordan's bounded-variation ledgers and the Bishop–Bridges treatment of finite mesh integration, the $\mathsf{UniformPartitionVariation}^{\uparrow}$ packet records the variation read available on a displayed uniform partition. It sits after the equal-mesh carrier of chapter 2033, the finite variation partition surface of chapter 6386, the Darboux-sum route of chapter 1880, the Riemann-sum chapter of chapter 1615, and the $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ numeric rows of chapter 38, chapter 301, and chapter 291. The packet names a finite uniform-mesh variation ledger; it does not claim a supremum over all partitions, selected refinement families, measure completion, quotient equality of functions, or ambient real completeness.

Definition 8026.1 (Uniform partition variation carrier)**.** *A* $\mathsf{UniformPartitionVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, M, V, D, R, S, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval and endpoint source row.*
2. *M is the accepted* $\mathsf{UniformPartition}^{\uparrow}$ *equal-mesh row over that interval.*
3. *V is the* $\mathsf{FiniteVariationPartition}^{\uparrow}$ *ledger row obtained from the same ordered cells.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-width and adjacent-difference budget row.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite adjacent-difference and tolerance data.*
6. *S is the* $\mathsf{DarbouxSum}^{\uparrow}$ *or Riemann-sum consumer handoff row, present only after M, V, D, R have been displayed.*
7. *L is the finite variation ledger, listing the adjacent absolute-difference reads and their finite sum along the uniform mesh.*
8. *H records componentwise* hsame *transport for I, M, V, D, R, S, L and the structural rows.*
9. *C records displayed* Cont *replay from the interval and uniform mesh through the variation ledger, numeric readback, Darboux or Riemann handoff, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformPartitionVariation}^{\uparrow}$, $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{FiniteVariationPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformPartitionVariation}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $I, M, V, D, R, S, L, H, C, P, N$. It has no coordinate for a host variation supremum, a chosen refinement sequence, quotient equality of functions, measure completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8026.2 (Uniform partition variation NameCert obligations)**.** *For every accepted* UniformPartitionVariatio *carrier $U = (I, M, V, D, R, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformPartitionVariation}^{\uparrow}}$ *surface consists exactly of carrier admission, uniform-mesh exposure through M, finite variation-partition exposure through V, dyadic budget exposure through D, regular-rational readback through R, Darboux or Riemann consumer handoff through S, finite variation-ledger exactness through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The only visible rows are the interval source, uniform mesh, finite variation partition, dyadic budget, regular-rational readback, consumer handoff, finite ledger, transport, replay, provenance, and local name.

The mesh and variation rows are tied to the same finite cells. Row M supplies the equal-width partition, and row V reads that partition as a finite variation ledger rather than as an arbitrary partition family.

The numeric side is finite. The dyadic row D stores mesh widths and adjacent-difference budgets, while R turns those displayed budgets into regular-rational readback before a real-facing consumer may use the ledger.

The handoff and structural rows do not add hidden analytic data. Row S exposes only the Darboux or Riemann consumer boundary, row L stores the finite variation sum, and H, C, P, N transport, replay, source, and name exactly that packet. □

Theorem 8026.3 (Uniform partition variation Darboux route)**.** *Every Darboux-facing or Riemann-sum-facing read of an accepted* $\mathsf{UniformPartitionVariation}^{\uparrow}$ *packet factors through*

$$\mathsf{UniformPartition}^{\uparrow} \longmapsto \mathsf{FiniteVariationPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow}$$

before the finite ledger L is public. The route gives consumers a uniform-mesh variation ledger over displayed real endpoints and numeric readback rows, not a host supremum over all partitions or a selected refinement process.

Proof. Start with the uniform partition row M. It fixes the equal-width cells over the interval row I, so the variation packet has a displayed mesh before any adjacent difference is read.

Next apply the finite variation partition row V. It reads the same ordered cells as a finite adjacent-difference ledger and keeps the construction inside the sibling surface of chapter 6386.

The numeric rows D and R then expose dyadic budgets and regular-rational readback for the same finite differences. The Darboux or Riemann handoff S is available only after those rows have been displayed.

Finally the ledger row L records the finite variation sum, and the rows H, C, P, N preserve the route under transport, replay, provenance, and local naming. Since the carrier contains no supremum, refinement selector, quotient function equality, measure completion, or ambient completeness coordinate, none can be exported by the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformPartitionVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformPartitionVariation}^{\uparrow}$ packet is generated from a RealUp interval row, UniformPartitionUp equal-mesh row, FiniteVariationPartitionUp ledger row, DyadicRatCoreUp budget row, RegSeqRatUp readback

row, Darboux or Riemann consumer handoff, finite variation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{UniformPartitionVariation}\uparrow}$, seed)

Scope closed: Definition 8026.1, Definition 8026.2, and Definition 8026.3 seed the finite uniform-mesh variation ledger over the uniform partition, finite variation partition, Darboux, Real, RegSeqRat, and DyadicRatCore rows.

Formal verification: FormalStatus(NameCert$_{\text{UniformPartitionVariation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a host supremum over all partitions, selected refinement families, quotient equality of functions, measure completion, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, uniform-mesh exposure, finite variation-partition exposure, dyadic budget readback, regular-rational readback, Darboux or Riemann handoff, variation-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.UniformPartitionVariationUp.

Chapter 8027

A Concrete Naming Certificate for $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ packet records the finite epsilon-tail witness surface for regular Cauchy real names. It sits over the finite stream-name discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the limit-facing handoff of chapter 1678. The packet supplies the Real and Completion route with displayed epsilon-tail evidence only; it does not import quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8027.1 (Regular Cauchy epsilon-tail carrier)**.** *A* $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It records the inspected indices, window requests, and tail-access cells used by the epsilon-tail witness.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row. It exposes the regular rational name only through the finite reads scheduled by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the requested epsilon, dyadic slack, radius comparison, and finite arithmetic budget.*
4. *Q is the precision-query row. It binds the external epsilon request to the displayed dyadic budget in D before any tail assertion is available.*
5. *T is the epsilon-tail witness row. It displays the tail threshold and the finitely checked later-window comparisons read from S, R, D, Q.*
6. *H records componentwise* hsame *transport for the stream schedule, regular rational row, dyadic budget, precision query, tail witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, R, D, Q \longmapsto T.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEpsilonTail}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, T, H, C, P, N$. It has no coordinate for quotient equality of streams, a host limit, a selected real representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyEpsilonTailUp`

Theorem 8027.2 (Regular Cauchy epsilon-tail NameCert obligations)**.** *For every accepted* RegularCauchyEpsilonTa *carrier*

$$E = (S, R, D, Q, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEpsilonTail}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the stream schedule, regular rational source, dyadic tolerance budget, precision query, epsilon-tail witness, transport, replay, provenance, and local naming rows;*
2. *query-budget exactness, requiring Q to bind the requested epsilon to the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget before the tail threshold is read;*
3. *stream-readback exactness, requiring every later-window comparison in T to be read from the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows;*
4. *transport and replay stability through H and C for the same displayed epsilon-tail route;*
5. *provenance and local naming exhaustion through P and N;*
6. *refusal of quotient stream equality, host limits, selected real representatives, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Project the accepted carrier $E = (S, R, D, Q, T, H, C, P, N)$. The visible analytic rows are the finite stream schedule S, the regular rational source R, the dyadic budget D, the precision query Q, and the tail witness T.

The query first passes through Q and D. This forces the epsilon request to be recorded as a finite dyadic budget before any tail threshold or later-window comparison can be consumed.

The witness row T is downstream of the same schedule and source. Its threshold and later-window comparisons are read only from the windows listed in S and the rational observations exposed by R.

The structural rows preserve this route. The row H transports the displayed coordinates componentwise, C replays the same dependency order, and P, N source and name only the resulting packet.

Since none of the excluded principles is a coordinate of the carrier or a product of the replay route, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8027.3 (Regular Cauchy epsilon-tail Real seal route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or limit-facing consumer of an accepted* $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ *carrier factors through the finite route*

$$S, R, D, Q \longmapsto T \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed StreamName schedule, regular rational readback, dyadic epsilon budget, precision query, epsilon-tail witness, transport, replay, provenance, and local naming. It receives no quotient stream equality, host limit, selected real representative, countable-choice schedule, trichotomy principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the accepted packet. The stream and regular-rational rows S and R are the only source of observation data, and the dyadic row D is the only source of epsilon-budget data.

The precision query Q binds the requested tolerance to that dyadic budget. Hence a real-name or limit-facing read cannot access a tail threshold before the finite epsilon request has been displayed.

Read the witness row T next. It records the finite tail threshold and the later-window comparisons obtained from S, R, D, Q.

Finally apply the structural rows H, C, P, N. They transport, replay, source, and name the same route without adding any coordinate outside it. Therefore every Real or limit-facing handoff remains inside the finite epsilon-tail witness surface and cannot export the excluded quotient, selection, choice, trichotomy, host-limit, or completeness data. □

Falsifiable prediction: A $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ read is refuted whenever a $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or limit-facing consumer reaches an epsilon-tail witness without displaying the finite StreamName schedule, RegSeqRat source, DyadicRatCore tolerance row, precision query, componentwise transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, limit: none alone is bijective with $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ because the packet jointly records a finite stream schedule, a regular rational source, a dyadic epsilon budget, a precision query, a displayed tail witness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEpsilonTail}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEpsilonTail}^{\uparrow}$ packet is generated from a finite StreamName schedule, a RegSeqRat source row, a DyadicRatCore epsilon budget, a precision query, an epsilon-tail witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEpsilonTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8027.1, Definition 8027.2, and Definition 8027.3 seed the finite epsilon-tail witness route over displayed StreamName, RegSeqRat, DyadicRatCore, Real, Limit, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEpsilonTail}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyEpsilonTailUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host limits, selected real representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean proof obligations, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, query-budget exactness, stream-readback exactness, epsilon-tail witness coverage, transport, replay, provenance, local naming, and Real-seal route nonescape for BEDC.Derived.RegularCauchyEpsilonTailUp.

Chapter 8028

A Concrete Naming Certificate for $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$

Following Baire and Lebesgue on pointwise hierarchies of real functions, the $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ packet records a real-valued function as a displayed pointwise limit of $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ rows. It sits after the Baire class one carrier of chapter 7703, the pointwise-limit packet of chapter 2314, the stream-name schedule of chapter 126, the regular-rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet gives later descriptive-analysis consumers a second pointwise layer without quotient function equality, selected representatives, countable choice, ambient measurability, or completeness of a host function space.

Definition 8028.1 (Baire class two real-function carrier)**.** *A* $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, S, B, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source real-domain row whose point requests are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing the inspected stages of the outer pointwise sequence.*
3. *B assigns each inspected outer stage to a displayed* $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ *row.*
4. *L is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *handoff that compares the stage outputs only through finite pointwise windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to each inspected output window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed second-level pointwise value.*
7. *H records componentwise* hsame *transport for source, schedule, Baire-one stage, pointwise-limit, readback, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a point request through $X, S, B, L, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$, $\mathsf{BaireClassOneRealFunction}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireClassTwoRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host equality of functions, quotient function spaces, selected representatives, countable choice, arbitrary measurability, or an ambient theorem identifying all Baire class two maps.

Theorem 8028.2 (Baire class two real-function NameCert obligations)**.** *For every accepted* $\mathsf{BaireClassTwoRealFunc}$ *carrier* $F = (X, S, B, L, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireClassTwoRealFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the source real-domain row, the* $\mathsf{StreamName}^{\uparrow}$ *outer stage schedule, the* $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ *stage rows, the* $\mathsf{PointwiseLimit}^{\uparrow}$ *handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, package provenance, and local naming. Every downstream descriptive-analysis consumer must consume those rows before treating the packet as a Baire class two real-valued function.*

Proof. Project the carrier of Definition 8028.1. The accepted packet exposes only the source row, outer stage schedule, Baire-one stage rows, pointwise-limit handoff, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

The second pointwise layer starts with X and S. A consumer first names the point request and the inspected outer stages, then reads the displayed $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ row at each inspected stage.

The output route is then forced through L, R, E. The pointwise-limit row records the finite comparison windows, the regular-rational row records the numerical readback, and the Real seal records the terminal pointwise value.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores host function equality, a quotient function space, selected representatives, countable choice, arbitrary measurability, or an ambient classification theorem, the listed obligations exhaust the local certificate surface. □

Theorem 8028.3 (Baire class two pointwise handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, pointwise-limit, or descriptive-analysis consumer of an accepted* $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BaireClassOneRealFunction}^{\uparrow} \longmapsto \mathsf{PointwiseLimit}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the Baire class two row is a displayed pointwise sequence of Baire class one real-valued maps with finite readback, not a host function equality class, a selected second-level limit map, or an imported measurable-function hierarchy.

Proof. Begin with a downstream consumer. Carrier admission forces the consumer to read the $\mathsf{StreamName}^{\uparrow}$ schedule before any stage of the outer approximating sequence can be inspected.

The inspected stage is then a $\mathsf{BaireClassOneRealFunction}^{\uparrow}$ row. Its output is not exposed as a host function value; it passes through the $\mathsf{PointwiseLimit}^{\uparrow}$ handoff, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Transport and replay are supplied by H and C, while P and N keep the provenance and local name attached to the same finite tuple. Consequently the consumer receives only the displayed pointwise route over named sibling packets.

The carrier inventory excludes host function equality, quotient function spaces, selected pointwise limits, countable choice, arbitrary measurability, and imported hierarchy theorems, so the handoff is exactly the finite BEDC route stated above. □

Falsifiable prediction: An accepted $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ consumer is refuted if it treats a second-level pointwise limit as public before displaying the StreamName outer schedule, BaireClassOne stage rows, PointwiseLimit handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: baireclassone, pointwiselimit, streamname, regseqrat, real: none is bijective because $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ binds a real-domain source, explicit outer stage schedule,

Baire-one stage rows, pointwise-limit handoff, numerical readback, Real seal, transport, replay, provenance, and local NameCert rows in one packet. Each sibling supplies one constituent surface, while only this packet names the second pointwise layer.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireClassTwoRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ packet is generated from a RealUp source row, a StreamName outer stage schedule, BaireClassOneRealFunction stage rows, a PointwiseLimit handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireClassTwoRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8028.1, Definition 8028.2, and Definition 8028.3 seed the explicit pointwise-sequence-of-Baire-one-functions route over BaireClassOneRealFunctionUp, PointwiseLimitUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireClassTwoRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function equality, quotient function spaces, selected pointwise limits, countable choice, arbitrary measurability, or a full Baire hierarchy.

Upgrade path: Obligation closure requires checked carrier admission, outer stage schedule exactness, Baire-one stage exposure, pointwise-limit handoff, pointwise readback, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BaireClassTwoRealFunctionUp.

Chapter 8029

A Concrete Naming Certificate for $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ packet records a real-valued function as a displayed pointwise limit of $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ rows. It consumes the Baire class two sibling of chapter 8028, the pointwise-limit handoff of chapter 2314, the continuous-map observation discipline of chapter 130, the finite $\mathsf{StreamName}^{\uparrow}$ schedules of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives later oscillation and rank consumers a third finite pointwise layer without importing quotient function equality, selected representatives, countable choice, arbitrary measurability, or ambient completeness of a host function space.

Definition 8029.1 (Baire class three real-function carrier)**.** *A* $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, S, B, L, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source real-domain row whose point requests are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing the inspected stages of the outer pointwise sequence.*
3. *B assigns each inspected outer stage to a displayed* $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ *source row.*
4. *L is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *handoff comparing the stage outputs only through finite pointwise windows.*
5. *O is the observation row: it records the* $\mathsf{ContinuousMap}^{\uparrow}$*-compatible point request surface and the* $\mathsf{StreamName}^{\uparrow}$ *windows inspected before an output value can be read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the observed stage outputs and tolerance requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed third-level pointwise value.*
8. *H records componentwise* hsame *transport for the source, schedule, Baire-two stage, pointwise-limit, observation, readback, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a point request through $X, S, B, L, O, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$, $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireClassThreeRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host equality of functions, quotient function spaces, selected representatives, countable choice, arbitrary measurability, or an ambient theorem identifying all Baire class three maps.

Theorem 8029.2 (Baire class three real-function NameCert obligations)**.** *For every accepted* $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ *carrier* $F = (X, S, B, L, O, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireClassThreeRealFunctio}}$ *surface consists exactly of carrier admission for the source real-domain row, the* $\mathsf{StreamName}^{\uparrow}$ *outer stage schedule, the* $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ *stage rows, the* $\mathsf{PointwiseLimit}^{\uparrow}$ *handoff, the* $\mathsf{ContinuousMap}^{\uparrow}$*-compatible observation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every downstream oscillation, rank, Real, or descriptive-analysis consumer must consume those rows before treating the packet as a Baire class three real-valued function.*

Proof. Project the carrier of Definition 8029.1. The accepted packet exposes only the source row, outer stage schedule, Baire-two stage rows, pointwise-limit handoff, observation row, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

The third pointwise layer starts with X and S. A consumer first names the point request and the inspected outer stages, then reads the displayed $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ row at each inspected stage.

The output route is then forced through L, O, R, E. The pointwise-limit row records the finite comparison windows, the observation row keeps the point request on the displayed $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ surfaces, the regular-rational row records the numerical readback, and the Real seal records the terminal pointwise value.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate stores host function equality, a quotient function space, selected representatives, countable choice, arbitrary measurability, or an ambient classification theorem, the listed obligations exhaust the local certificate surface. □

Theorem 8029.3 (Baire class three pointwise ladder)**.** *Every* $\mathsf{Real}^{\uparrow}$*, pointwise-limit, oscillation-rank, or descriptive-analysis consumer of an accepted* $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BaireClassTwoRealFunction}^{\uparrow} \longmapsto \mathsf{PointwiseLimit}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeq}$$

Thus the Baire class three row is a displayed pointwise sequence of Baire class two real-valued maps with finite observation and readback, not a host function equality class, a selected third-level limit map, or an imported measurable-function hierarchy.

Proof. Begin with a downstream consumer. Carrier admission forces the consumer to read the $\mathsf{StreamName}^{\uparrow}$ schedule before any stage of the outer approximating sequence can be inspected.

The inspected stage is then a $\mathsf{BaireClassTwoRealFunction}^{\uparrow}$ row. Its output is not exposed as a host function value; it passes through the $\mathsf{PointwiseLimit}^{\uparrow}$ handoff and the observation row that binds point requests to $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ surfaces.

The readback route then reaches R and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row records the finite numerical windows, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of the same displayed pointwise ladder.

Transport and replay are supplied by H and C, while P and N keep the provenance and local name attached to the same finite tuple. Consequently the consumer receives only the displayed pointwise route over named sibling packets.

The carrier inventory excludes host function equality, quotient function spaces, selected pointwise limits, countable choice, arbitrary measurability, imported hierarchy theorems, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so the ladder is exactly the finite BEDC route stated above. □

Falsifiable prediction: An accepted $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ consumer is refuted if it treats a third-level pointwise limit as public before displaying the StreamName outer schedule, BaireClassTwo stage rows, PointwiseLimit handoff, ContinuousMap-compatible observation row, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: baireclasstwo, pointwiselimit, continuousmap, streamname, regseqrat, real: none is bijective because $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ binds a real-domain source, explicit outer stage schedule, Baire-two stage rows, pointwise-limit handoff, observation row, numerical readback, Real seal, transport, replay, provenance, and local NameCert rows in one packet. Each sibling supplies one constituent surface, while only this packet names the third pointwise layer.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireClassThreeRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireClassThreeRealFunction}^{\uparrow}$ packet is generated from a RealUp source row, a StreamName outer stage schedule, BaireClassTwoRealFunction stage rows, a PointwiseLimit handoff, ContinuousMap-compatible observation rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireClassThreeRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8029.1, Definition 8029.2, and Definition 8029.3 seed the explicit pointwise-sequence-of-Baire-two-functions route over BaireClassTwoRealFunctionUp, PointwiseLimitUp, ContinuousMapUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireClassThreeRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function equality, quotient function spaces, selected pointwise limits, countable choice, arbitrary measurability, oscillation ranks, or a full Baire hierarchy.

Upgrade path: Obligation closure requires checked carrier admission, outer stage schedule exactness, Baire-two stage exposure, pointwise-limit handoff, observation-row compatibility, pointwise readback, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BaireClassThreeRealFunctionUp.

Chapter 8030

A Concrete Naming Certificate for $\mathsf{RegularCauchyAdmission}^{\uparrow}$

Bishop and Bridges constructive analysis treats a regular Cauchy name as usable real data only after the finite rational observations, tolerance ledger, and terminal seal have been kept separate. The $\mathsf{RegularCauchyAdmission}^{\uparrow}$ packet records that admission surface between the regular rational sequence of chapter 301 and the real-name boundary of chapter 38. It also keeps the source schedule of chapter 126 and the dyadic arithmetic ledger of chapter 291 visible to completion-facing readers.

The packet is an admission route, not a real-completion theorem. It names the finite data that a later $\mathsf{Real}^{\uparrow}$ or completion chapter may consume when it reads a regular rational name as Real-facing data, while refusing quotient stream equality, selected representatives, countable choice, trichotomy, and ambient completeness.

Definition 8030.1 (Regular Cauchy admission carrier)**.** *A* $\mathsf{RegularCauchyAdmission}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source schedule, recording the finite observation windows through which downstream readers ask for rational data.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying the regular rational sequence data visible on the windows named by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic precision cells, slack conversions, and comparison budgets used to audit the regularity of R.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream schedule, regular readback, and dyadic tolerance ledger have been displayed.*
5. *H records componentwise* hsame *transport for S, R, D, E and for the local classifier row.*
6. *C records displayed* Cont *replay routes from a Real-facing request back through E, D, R, and S.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAdmission}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAdmission}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying these rows. It has no coordinate for quotient stream equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8030.2 (Regular Cauchy admission NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAdmission}^{\uparrow}$ *carrier* $A = (S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAdmission}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The obligations require every Real-facing read to expose* S*,* R*, and* D *before the seal row* E *is public.*

Proof. Project the carrier of Definition 8030.1. The analytic rows are exactly S, R, D, E, while H, C, P, N transport, replay, package, and name the same route.

The regular rational readback row R is not a Real object by itself. A consumer must first read the stream windows in S and the dyadic tolerance ledger in D before the seal row E can be exposed.

Transport through H and replay through C preserve the same finite order. Since neither row adds quotient stream equality, selected representatives, countable choice, trichotomy, or completeness, the local NameCert obligations are exhausted by the displayed packet surface. □

Theorem 8030.3 (Regular Cauchy admission Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyAdmission}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto E.$$

Thus a completion or Real-facing consumer sees the stream schedule, regular rational readback, dyadic tolerance ledger, and terminal seal in order. The route exports no quotient stream equality, selected representative, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the seal row E. By the carrier definition, E is the terminal row reached from the displayed regular rational data and its dyadic tolerance ledger.

The stream schedule S supplies the finite request windows, and R supplies the regular rational observations visible on those windows. The ledger D then records the precision and comparison data needed before the seal is public.

The structural rows H, C, P, N preserve and name this route without adding host-level or quotient coordinates. Hence every Real-facing read factors through S, R, D, E and cannot export the excluded principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAdmission}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyAdmission}^{\uparrow}$ packet is generated from a StreamName schedule, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyAdmission}^{\uparrow}}$, seed)

Scope closed: Definition 8030.1, Definition 8030.2, and Definition 8030.3 seed the finite admission route from regular rational reads to the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyAdmission}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular readback, dyadic tolerance exactness, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyAdmissionUp.

Chapter 8031

A Concrete Naming Certificate for $\mathsf{LipschitzFixedPoint}^{\uparrow}$

Following Banach and Bishop–Bridges constructive fixed-point theory, the $\mathsf{LipschitzFixedPoint}^{\uparrow}$ packet records the finite endpoint certificate read by a Lipschitz contraction consumer after the contraction, modulus, and Cauchy-orbit rows are displayed. It sits after the Banach contraction route of chapter 3529, the constructive contraction modulus of chapter 7631, and the orbit endpoint seal of chapter 7047. The packet names the endpoint-facing certificate for a fixed-point read; it does not select a host fixed point, quotient a completion, use countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8031.1 (Lipschitz fixed-point carrier)**.** *A* $\mathsf{LipschitzFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, M, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachContraction}^{\uparrow}$ *row that supplies the contractive self-map route and separated endpoint discipline.*
2. *M is the* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ *row that exposes the finite contraction modulus before the orbit tail is read.*
3. *O is the* $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ *row that carries the Picard orbit, Cauchy-modulus handoff, and endpoint seal boundary.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite regular tail windows inspected by the endpoint consumer.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal available only after B, M, O, R have been replayed.*
6. *H records componentwise* hsame *transport for contraction, modulus, orbit seal, readback, endpoint seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a fixed-point request through B, M, O, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LipschitzFixedPoint}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{ConstructiveContractionModulus}^{\uparrow}$, $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows, their displayed transports, continuation replay, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.LipschitzFixedPointUp.TasteGate.LipschitzFixedPointTasteGate single carrier alignment`

Theorem 8031.2 (Lipschitz fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzFixedPoint}^{\uparrow}$ *carrier* $L = (B, M, O, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzFixedPoint}^{\uparrow}}$ *surface consists exactly of contraction admission through* B*, modulus admission through* M*, orbit-seal admission through* O*, regular rational readback through* R*, terminal endpoint sealing through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8031.1. The projection exposes the contraction row, modulus row, orbit-seal row, readback row, endpoint seal, transport, replay, provenance, and local naming rows.

The analytic rows are read in fixed order. The Banach contraction row supplies the contractive route, the constructive modulus row supplies the finite shrinkage budget, the orbit seal supplies the Picard Cauchy endpoint boundary, and the regular rational readback supplies the displayed tail windows before E is available.

The structural rows H, C, P, N preserve and identify that same packet. Since the carrier has no coordinate for host fixed-point selection, quotient completion equality, countable choice, or ambient RealUp completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8031.3 (Lipschitz fixed-point Picard orbit Cauchy route)**.** *Every fixed-point endpoint consumer of an accepted* $\mathsf{LipschitzFixedPoint}^{\uparrow}$ *carrier factors through the finite Picard orbit route*

$$\mathsf{BanachContraction}^{\uparrow} \longmapsto \mathsf{ConstructiveContractionModulus}^{\uparrow} \longmapsto \mathsf{CauchyFixedPointOrbitSeal}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Rea}$$

The endpoint read therefore consumes the displayed contraction, modulus, orbit seal, regular readback, terminal seal, componentwise transport, replay, provenance, and local naming rows before any fixed-point row is public.

Proof. Begin with an endpoint request. Carrier admission gives access to the contraction row B before the modulus or orbit rows can be consumed.

Next read M, O, R. The modulus row exposes the finite shrinkage data, the orbit-seal row exposes the Picard Cauchy boundary, and the regular rational readback records the finite tail windows inspected by the endpoint consumer.

Finally read E, H, C, P, N. These rows seal, transport, replay, source, and name the same finite endpoint route, so the public fixed-point read remains inside the displayed Picard orbit certificate. □

Theorem 8031.4 (Lipschitz fixed-point non-escape)**.** *An accepted* $\mathsf{LipschitzFixedPoint}^{\uparrow}$ *carrier exports only the rows* $B, M, O, R, E, H, C, P, N$ *of Definition 8031.1. It exports no host-selected fixed point, quotient completion representative, host equality of endpoints, countable-choice selector, untracked contraction modulus, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Inspect the carrier. Its only endpoint-facing rows are the contraction row B, modulus row M, orbit seal O, regular readback R, and terminal Real seal E.

Apply the NameCert obligations of Definition 8031.2. Every accepted read remains inside those displayed rows and the structural rows H, C, P, N.

The excluded objects would require additional carrier coordinates or continuation routes. Since none are present, no downstream consumer can recover host selection, quotient completion data, host endpoint equality, choice, an untracked modulus, or ambient RealUp completeness from the packet. □

Falsifiable prediction: An accepted $\mathsf{LipschitzFixedPoint}^{\uparrow}$ read is refuted if a fixed-point endpoint consumer reaches the terminal Real seal without displaying the BanachContraction row, ConstructiveContractionModulus row, CauchyFixedPointOrbitSeal row, RegSeqRat readback, componentwise transport, replay, provenance, and local NameCert row.

Independence witness: banach-contraction, constructive-contraction-modulus, cauchy-fixed-point-orbit-seal, regseqrat, real: none alone is bijective with $\mathsf{LipschitzFixedPoint}^{\uparrow}$ because the packet jointly records contraction admission, finite modulus data, orbit sealing, regular readback, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzFixedPoint}^{\uparrow}$ packet is generated from a BanachContraction row, a ConstructiveContractionModulus row, a CauchyFixedPointOrbitSeal row, RegSeqRat readback, a terminal RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8031.1, Definition 8031.2, Definition 8031.3, and Definition 8031.4 seed the finite endpoint certificate over contraction, modulus, orbit seal, regular readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point selection, quotient completion equality, host endpoint equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction exposure, modulus accountability, orbit-seal exactness, RegSeqRat readback, endpoint sealing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LipschitzFixedPointUp.

Chapter 8032

A Concrete Naming Certificate for $\mathsf{RiemannSumModulus}^{\uparrow}$

Following Riemann and Darboux, the $\mathsf{RiemannSumModulus}^{\uparrow}$ packet records the finite refinement-modulus surface that sits between the displayed Riemann sum of chapter 1615, the tagged partition carrier of chapter 7449, and the mesh-bound packet of chapter 7944. Its tolerance rows are read through chapter 291, its regular rational and real-name handoff follows chapter 301 and chapter 38, and its only public promise is a finite modulus for comparing sufficiently refined displayed sums.

Definition 8032.1 (Riemann sum modulus carrier)**.** *A* $\mathsf{RiemannSumModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, R, T, Q, D, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the mesh-budget row, carrying a dyadic mesh bound read through* $\mathsf{RiemannSumMesh}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *R is the refinement-witness row, recording that a displayed tagged partition refines the budget named by M.*
3. *T is the* $\mathsf{TaggedPartition}^{\uparrow}$ *row whose cells, endpoints, and tag-containment data are the only partition data visible to the packet.*
4. *Q is the Riemann-sum comparison row, relating two displayed* $\mathsf{RiemannSum}^{\uparrow}$ *packets only through their common refinement budget and tagged-partition rows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for the displayed comparison bound.*
6. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row, sealing rational readback before any Real-facing consumer reads the comparison.*
7. *H records componentwise* hsame *transport for $M, R, T, Q, D, S, C, P, N$.*
8. *C records displayed* Cont *replay through mesh budgets, refinement witnesses, tagged partitions, Riemann-sum comparisons, dyadic tolerances, and the regular-real handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RiemannSumModulus}^{\uparrow}$*,* $\mathsf{RiemannSum}^{\uparrow}$*,* $\mathsf{TaggedPartition}^{\uparrow}$*,* $\mathsf{RiemannSumMesh}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, T, Q, D, S, H, C, P, N$. It has no coordinate for host integrability, host limits, quotient equality of functions, selected tag families, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8032.2 (Riemann sum modulus NameCert obligations)**.** *For the carrier of Definition 8032.1, the local* $\mathsf{NameCert}_{\mathsf{RiemannSumModulus}^{\uparrow}}$ *surface consists exactly of mesh-budget exposure, refinement-witness exposure, tagged-partition readback, Riemann-sum comparison, dyadic tolerance readback, RegSeqRat/Real handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming for the same packet*

$$U = (M, R, T, Q, D, S, H, C, P, N).$$

Proof. Project the accepted carrier. The rows $M, R, T, Q, D, S, H, C, P, N$ are exactly the mesh budget, refinement witness, tagged partition, comparison, tolerance, regular-real handoff, transport, replay, provenance, and local naming rows.

Read the supporting packets in order. The budget row points to the finite mesh data of $\mathsf{RiemannSumMesh}^{\uparrow}$, the partition row points to $\mathsf{TaggedPartition}^{\uparrow}$, and the comparison row reads only displayed $\mathsf{RiemannSum}^{\uparrow}$ packets that share the same refinement budget.

The tolerance and handoff rows keep the comparison constructive. The dyadic row D supplies the finite rational bound, while S routes any later Real-facing read through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seals. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 8032.3 (Riemann sum modulus refinement stability)**.** *Let $U = (M, R, T, Q, D, S, H, C, P, N)$ be an accepted* $\mathsf{RiemannSumModulus}^{\uparrow}$ *carrier. Any two accepted refinement reads that agree on the mesh-budget row M, the refinement-witness row R, the tagged-partition row T, and the dyadic tolerance row D must export the same Riemann-sum comparison boundary through Q, S, H, C, P, N. The packet therefore stabilizes finite refinement comparison before any consumer asks for a limiting integral.*

Proof. Start with the four controlling rows M, R, T, D. They fix the displayed mesh budget, the finite refinement witness, the tagged partition whose cells and tags are read, and the dyadic tolerance used to compare sums.

The comparison row Q has no independent source of partition or tolerance data. It can compare Riemann sums only by replaying the same mesh, refinement, tagged-partition, and tolerance rows.

Finally transport and replay through H, C, P, N. Componentwise hsame preserves the controlling rows, Cont replays the same finite route, and provenance and local naming add no selected refinement family or limiting process. Hence both reads expose the same comparison boundary. □

Theorem 8032.4 (Riemann sum modulus consumer handoff)**.** *Every Darboux-facing or Riemann-integral-facing consumer of an accepted* $\mathsf{RiemannSumModulus}^{\uparrow}$ *carrier receives only the finite route*

$$\mathsf{RiemannSumMesh}^{\uparrow} \longmapsto \mathsf{TaggedPartition}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by $M, R, T, Q, D, S, H, C, P, N$. The handoff supplies a tolerance-indexed refinement comparison for displayed sums, not integrability, host limits, quotient function equality, selected tags, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the mesh and partition side first. The consumer receives the finite budget M, the refinement witness R, and the tagged partition T before any Riemann-sum comparison is visible.

Next read the comparison and tolerance rows. The row Q compares displayed $\mathsf{RiemannSum}^{\uparrow}$ packets only inside the budget fixed by M, R, T, and the row D supplies the dyadic tolerance against which that comparison is checked.

The regular-real handoff is terminal. The row S seals the rational readback before a Real-facing consumer can inspect it, while H, C, P, N preserve the same finite route. Thus the export is a refinement-modulus boundary for sums, not an ambient integral or a choice-selected refinement process. □

Falsifiable prediction: Removing R permits a consumer to compare Riemann sums across unrelated partitions while retaining the same mesh budget; the local classifier must reject such a packet because refinement has no displayed witness.

Independence witness: riemannsum, taggedpartition, riemann-sum-mesh, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{RiemannSumModulus}^{\uparrow}$ because this carrier alone joins mesh budget, refinement witness, tagged-partition readback, sum comparison, dyadic tolerance, regular-real handoff, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSumModulus}^{\uparrow}$ packet is generated from a mesh-budget row, refinement-witness row, TaggedPartition row, Riemann-sum comparison row, DyadicRatCore tolerance row, RegSeqRat/Real handoff row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSumModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8032.1, Definition 8032.2, Definition 8032.3, and Definition 8032.4 seed the finite refinement-modulus handoff for displayed Riemann sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSumModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim integrability, host limits, quotient function equality, selected tags, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement-witness stability, Riemann-sum comparison exactness, dyadic tolerance readback, RegSeqRat/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumModulusUp.

Chapter 8033

A Concrete Naming Certificate for $\mathsf{CompactMetricSequentialComplete}^{\uparrow}$

Following Bishop and Bridges on compact metric spaces and Kelley on sequential compactness in metric topology, the $\mathsf{CompactMetricSequentialComplete}^{\uparrow}$ packet records the finite sequential-completeness route that becomes visible when compact-metric finite-net data and Cauchy readback data are carried by the same NameCert surface. It sits beside the compact metric certificate of chapter 129, the complete metric service of chapter 128, the totally bounded finite-net surface of chapter 127, and the Real readback route through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet is a compact-metric sequential-completeness carrier; it is not an open-cover compactness theorem, a selected infinite subsequence, quotient stream equality, countable choice, host equality of limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8033.1 (Compact-metric sequential-complete carrier)**.** *A* $\mathsf{CompactMetricSequentialComplete}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the metric compactness interface inspected by the packet.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row for the displayed tolerances, centers, coverage witnesses, and refinement comparisons.*
3. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy service row receiving the sequential Cauchy window after the finite-net schedule has stabilized.*
4. *W is the sequential-window ledger recording finite source windows, tail indices, and Cauchy-readiness evidence for the displayed sequence.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tail bounds and distance comparisons used by T, W, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the readback row and complete-metric service row have been displayed.*
7. *H records componentwise* hsame *transport for the compact metric source, finite-net row, complete-service row, sequential windows, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a compact-metric sequential request through $K, T, W, R, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricSequentialComplete}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialComplete}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, L, W, R, E, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subsequences, quotient Cauchy-name equality, countable choice, host real equality, or detached ambient completeness.

Theorem 8033.2 (Compact-metric sequential-complete obligations)**.** *For every accepted* CompactMetricSequential *carrier*

$$S = (K, T, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialComplete}^{\uparrow}}$ *surface consists exactly of compact metric source admission through K, finite-net accountability through T, sequential Cauchy-window admission through W, rational tail readback through R, complete-metric service through L, terminal Real sealing through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 8033.1. The accepted packet exposes exactly the compact metric, finite-net, complete-service, sequential-window, readback, seal, transport, replay, provenance, and local naming rows.

Read the compact metric side first. The source row K supplies only the compact-metric interface, while T supplies the finite-net schedule and coverage rows used to force Cauchy windows at displayed tolerances.

Next read the sequential side. The row W records the finite tail windows and Cauchy-readiness evidence consumed by R; it does not store a selected infinite subsequence or a quotient stream equality.

Finally use R, L, E, H, C, P, N. Rational readback is passed to the complete-metric service row before the terminal Real seal is available, and transport and replay preserve the same displayed route. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8033.3 (Compact-metric sequential-complete readback route)**.** *Every sequential-completeness consumer of an accepted* $\mathsf{CompactMetricSequentialComplete}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg},$$

Thus Cauchy-readback and finite-net obligations are displayed before any terminal Real seal is consumed. The route exports no open-cover compactness theorem, selected subsequence, quotient Cauchy-name equality, countable-choice schedule, host limit equality, or detached ambient completeness principle.

Proof. Start from the NameCert obligations of Definition 8033.2. They place every accepted sequential-completeness request on the same compact metric source row, finite-net row, sequential-window row, readback row, complete-service row, and terminal seal.

The finite-net and Cauchy-readback rows are adjacent but not collapsed. The row T accounts for displayed centers and covers; the row R accounts for rational tail comparisons; the row L consumes those comparisons as complete-metric service.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Therefore the consumer reaches $\mathsf{Real}^{\uparrow}$ only through the displayed finite packet, and the excluded open-cover,

selected-subsequence, quotient, choice, host-equality, and ambient-completeness coordinates are unavailable. □

Theorem 8033.4 (Compact-metric sequential-complete net handoff)**.** *Every accepted* CompactMetricSequentialCor *consumer that reads a sequential-net route must first pass through the sibling* CompactMetricSequentialNet$^{\uparrow}$ *finite-net extraction surface of chapter 4019. Only then may it read the compact metric source row K, total-bounded finite-net row T, sequential-window row W,* RegSeqRat$^{\uparrow}$ *readback row R,* CompleteMetric$^{\uparrow}$ *service row L, terminal* Real$^{\uparrow}$ *seal E, and structural rows H, C, P, N. The handoff exports no open-cover compactness theorem, selected infinite subsequence, quotient stream equality, countable-choice schedule, host limit equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with Definition 8033.2. It exposes the compact metric source, finite-net, sequential-window, readback, complete-service, Real seal, and structural rows as the whole accepted packet.

Apply the readback route of Definition 8033.3. This places every sequential-completeness consumer on the route from CompactMetric$^{\uparrow}$ through TotallyBounded$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, and Real$^{\uparrow}$ before the structural rows are public.

Now read the sibling route through chapter 4019. The subsequence handoff of Definition 4019.3 supplies the finite-net extraction surface, so the sequential-complete packet can consume the net route only after that sibling surface is displayed.

The structural rows H, C, P, N preserve the same finite handoff. Since neither packet contains a coordinate for open-cover compactness, selected infinite subsequences, quotient stream equality, countable choice, host limit equality, or ambient Real completeness, the consumer receives only the displayed sequential-net route. □

Falsifiable prediction: An accepted CompactMetricSequentialComplete$^{\uparrow}$ read is refuted if a sequential-completeness consumer reaches the terminal Real seal without displaying the CompactMetric source row, TotallyBounded finite-net row, RegSeqRat readback row, CompleteMetric service row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, completemetric, totallybounded, regseqrat, real: none alone is bijective with CompactMetricSequentialComplete$^{\uparrow}$ because this packet jointly carries compact-metric source data, finite-net accountability, sequential Cauchy windows, rational readback, complete-metric service, terminal Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{CompactMetricSequentialComplete}^{\uparrow}}$**.**

Constructive story: A CompactMetricSequentialComplete$^{\uparrow}$ packet is generated from a CompactMetricUp source row, a TotallyBoundedUp finite-net row, a CompleteMetricUp service row, sequential Cauchy windows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactMetricSequentialComplete}^{\uparrow}}$, seed)

Scope closed: Definition 8033.1, Definition 8033.2, and Definition 8033.3 seed the compact-metric sequential-completeness route over finite-net and Cauchy-readback obligations.

Formal verification: FormalStatus(NameCert$_{\text{CompactMetricSequentialComplete}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite subsequences, quotient Cauchy-name equality, countable choice, host equality of limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net accountability, sequential-window admission, RegSeqRat readback, CompleteMetric service, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactMetricSequentialCompleteUp.

Chapter 8034

A Concrete Naming Certificate for $\mathsf{LocalObservationDigestGate}^{\uparrow}$

Following standard hash and provenance interface discipline, the $\mathsf{LocalObservationDigestGate}^{\uparrow}$ packet realizes the local boundary question family of Definition 9402.35 as finite NameCert data. It sits beside the digest provenance packet of chapter 1062, the hash-like boundary gate of chapter 1359, and the observation-fiber digest surface of chapter 1357. The packet is local: it records a visible token, source fiber, GAP provenance, exactness status, far-end status, and hidden-source refusal as rows of one carrier, while refusing any global hash, hidden source oracle, object-level T value, or digest equality as source equality.

Definition 8034.1 (Local observation digest gate carrier)**.** *A* $\mathsf{LocalObservationDigestGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, F, Q, E, A, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the visible local digest token row exposed to observation consumers.*
2. *F is the source-fiber row that names the local observation fiber to which V belongs.*
3. *Q is the GAP provenance row recording the visible source route and the refused hidden-source route.*
4. *E is the exactness-status row stating whether the visible token is allowed to support a local exactness claim.*
5. *A is the apophatic far-end status row recording the boundary side of the observation.*
6. *R is the hidden-source refusal row preventing digest equality from becoming source equality.*
7. *H records componentwise* hsame *transport for the visible token, fiber, GAP provenance, exactness status, far-end status, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a local observation request through $V, F, Q, E, A, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocalObservationDigestGate}^{\uparrow}$, $\mathsf{DigestProvenancePacket}^{\uparrow}$, $\mathsf{HashLikeBoundaryGate}^{\uparrow}$, $\mathsf{ObservationFiberDigest}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalObservationDigestGate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, F, Q, E, A, R, H, C, P, N$. It has no coordinate for a global hash of the universe, hidden source recovery, object-level T*, quotient source equality, host collision semantics, theorem-input parameters, or detached certificate names.*

Theorem 8034.2 (Local observation digest gate obligations)**.** *For every accepted* $\mathsf{LocalObservationDigestGate}^{\uparrow}$ *carrier*

$$G = (V, F, Q, E, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocalObservationDigestGate}^{\uparrow}}$ *surface consists exactly of visible token admission through V, source-fiber admission through F, GAP provenance through Q, exactness status through E, apophatic far-end status through A, hidden-source refusal through R, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 8034.1. The accepted packet exposes exactly the visible token, source fiber, GAP provenance, exactness status, far-end status, hidden-source refusal, transport, replay, provenance, and local naming rows.

Read the digest side through V, F, Q, E. A consumer may use the visible token only together with the displayed fiber, the GAP provenance route, and the exactness-status row.

Read the refusal side through A, R. The far-end status and hidden-source refusal rows block any conversion of digest equality into source equality or hidden source recovery.

Finally H, C, P, N preserve, replay, source, and name the same local packet. Since no global hash, object-level T value, quotient equality, host collision row, theorem-input parameter, or detached certificate name occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 8034.3 (Local observation digest gate boundary nonescape)**.** *Every accepted* $\mathsf{LocalObservationDigestGate}$ *consumer factors through the finite route*

$$\mathsf{DigestProvenancePacket}^{\uparrow} \longmapsto \mathsf{HashLikeBoundaryGate}^{\uparrow} \longmapsto V, F, Q, E, A, R \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The route supplies visible token, source fiber, GAP provenance, exactness status, far-end status, hidden-source refusal, transport, replay, provenance, and local naming. It supplies no global hash, hidden-source oracle, object-level T *value, quotient source equality, host collision semantics, or detached theorem input.*

Proof. Start from Definition 8034.2. They require every consumer-visible digest read to pass through the displayed token, fiber, GAP provenance, exactness, far-end, refusal, and structural rows.

The sibling $\mathsf{DigestProvenancePacket}^{\uparrow}$ route supplies the provenance discipline for visible digest tokens, while the sibling $\mathsf{HashLikeBoundaryGate}^{\uparrow}$ route supplies the boundary audit questions. This packet binds both to the same local observation rows V, F, Q, E, A, R.

The transport, replay, provenance, and naming rows H, C, P, N keep the route local. Therefore a consumer cannot leave the displayed packet to recover a hidden source, introduce a global hash, treat T as an object-level digest value, or use digest equality as source equality. □

Falsifiable prediction: An accepted $\mathsf{LocalObservationDigestGate}^{\uparrow}$ consumer is refuted if it uses a visible digest token without displaying the source fiber, GAP provenance, exactness status, apophatic far-end status, hidden-source refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: digestprovenancepacket, hashlikeboundarygate, observationfiberdigest: none alone is bijective with $\mathsf{LocalObservationDigestGate}^{\uparrow}$ because this packet jointly binds the

visible token, source fiber, GAP provenance, exactness status, far-end status, hidden-source refusal, transport, replay, provenance, and local NameCert rows as one local gate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalObservationDigestGate}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalObservationDigestGate}^{\uparrow}$ packet is generated from a visible token row, source-fiber row, GAP provenance row, exactness-status row, apophatic far-end status row, hidden-source refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalObservationDigestGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8034.1, Definition 8034.2, and Definition 8034.3 seed the local observation digest boundary over visible token, source-fiber, GAP, exactness, far-end, and refusal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalObservationDigestGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global hash semantics, hidden-source recovery, object-level T, quotient source equality, host collision semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible-token admission, source-fiber admission, GAP provenance, exactness status, far-end status, hidden-source refusal, transport, replay, provenance, and local naming for BEDC.Derived.LocalObservationDigestGateUp.

Chapter 8035

A Concrete Naming Certificate for $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ packet records the finite cluster-window route by which regular Cauchy real names are grouped before a Real or Completion consumer reads a seal. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the Bishop regular Cauchy completion boundary of chapter 1639. The packet supplies a finite cluster ledger only; it does not import quotient representatives, selected limits, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8035.1 (Finite real Cauchy cluster carrier)**.** *A* $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the inspected stream names, finite index windows, and observation cells available to the cluster route.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It exposes only the rational observations scheduled by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the finite dyadic radius, comparison slack, and arithmetic budget used to compare the readbacks.*
4. *L is the cluster-window ledger. It records the accepted cluster cells, their finite membership evidence, the dyadic comparison used for each cell, and the finite overlap checks between cells.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the terminal real-facing seals that may be read after S, R, D, L have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular rational readbacks, dyadic tolerances, cluster ledger, Real seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, R, D \longmapsto L \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteRealCauchyCluster}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, L, E, H, C, P, N$. It has no coordinate for quotient representatives, selected limits, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8035.2 (Finite real Cauchy cluster obligations)**.** *For every accepted* $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ *carrier*

$$K = (S, R, D, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRealCauchyCluster}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission for the stream-window, regular-readback, dyadic-tolerance, cluster-ledger, Real-seal, transport, replay, provenance, and local naming rows;*
2. *finite window coverage, requiring every cluster cell in L to cite only windows displayed by S and readbacks displayed by R;*
3. *cluster ledger exactness, requiring each membership, overlap, and separation comparison in L to be checked against the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D;*
4. *Real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *seal in E to be reachable only after the finite route S, R, D, L has been displayed;*
5. *transport stability, requiring componentwise* hsame *transport through H and displayed* Cont *replay through C to preserve the same cluster-window route, with provenance and local naming exhausted by P, N.*

No obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (S, R, D, L, E, H, C, P, N)$. The source rows are S for finite stream windows, R for regular rational readback, and D for dyadic tolerance data.

Finite window coverage follows because the cluster ledger L has no observation coordinate of its own. Each listed cluster cell must cite an inspected window from S and a readback from R before it can be accepted.

Ledger exactness follows from the placement of D. The membership, overlap, and separation entries in L are compared only against the displayed dyadic tolerance row, so a cluster claim cannot switch to a host metric or hidden real representative.

The Real-seal row E is downstream of L. Hence a Real or Completion consumer reaches E only after the finite stream, readback, tolerance, and cluster-ledger rows have already been displayed.

Finally, H and C transport and replay the same route, while P, N source and name the packet. Since the carrier contains no excluded coordinate, the five obligations exhaust the local NameCert surface. □

Falsifiable prediction: A $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ read is refuted whenever a Real or Completion consumer reaches a cluster seal without displaying the finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance row, cluster-window ledger, componentwise transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, bishop-regular-cauchy-completion: none alone is bijective with $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ because the packet jointly records finite stream windows, regular rational readbacks, dyadic tolerances, cluster cells, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealCauchyCluster}^{\uparrow}}$.

Constructive story: A FiniteRealCauchyCluster$^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a cluster-window ledger, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteRealCauchyCluster}\uparrow}$, seed)

Scope closed: Definition 8035.1 and Definition 8035.2 seed the finite cluster-window route over displayed StreamName, RegSeqRat, DyadicRatCore, Real, Bishop regular Cauchy completion, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteRealCauchyCluster}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.FiniteRealCauchyClusterUp`

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected limits, countable choice, host completeness, ambient Real$^{\uparrow}$ completeness, Lean proof obligations, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite window coverage, cluster ledger exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRealCauchyClusterUp.

Chapter 8036

A Concrete Naming Certificate for $\mathsf{OdeComparison}^{\uparrow}$

Following Walter or Hartman on ODE comparison principles, the $\mathsf{OdeComparison}^{\uparrow}$ packet records the finite BEDC surface by which two displayed ODE rows, one shared interval window, and one monotonicity or majorant ledger can be read before a downstream comparison consumer reaches a scalar bound. It sits after the ODE source of chapter 193, the Gronwall handoff of chapter 2034, the zero-order comparison route of chapter 2338, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names comparison data as displayed rows only; it is not a host solution theorem, a selected flow, a quotient path equality, or an ambient completeness principle.

Definition 8036.1 (ODE comparison carrier)**.** *An* $\mathsf{OdeComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D_0, D_1, I, L, M, G, S, R, E, H, C, P, N)$$

whose rows have the following roles.

1. *D_0 and D_1 are displayed* $\mathsf{Ode}^{\uparrow}$ *source rows carrying the two differential data packets that will be compared.*
2. *I is the shared interval or window row over which both ODE source rows are read.*
3. *L is the located-order or monotonicity ledger for the comparison request, and M is the comparison majorant row.*
4. *G is the* $\mathsf{GronwallInequality}^{\uparrow}$ *handoff used when the comparison is controlled by an integral or differential majorant.*
5. *S is the* $\mathsf{SturmComparison}^{\uparrow}$ *handoff used when the request is a zero-order or oscillation-window comparison.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite comparison windows, and E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *H, C, P, and N are the componentwise* hsame *transport row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{OdeComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $D_0, D_1, I, L, M, G, S, R, E, H, C, P, N$. It has no coordinate for host solution equality, selected flows, quotient paths, trichotomy, global existence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OdeComparisonUp.OdeComparisonTasteGate single carrier alignment`

Theorem 8036.2 (ODE comparison NameCert obligations)**.** *For every accepted* $\mathsf{OdeComparison}^{\uparrow}$ *carrier* $O = (D_0, D_1, I, L, M, G, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OdeComparison}^{\uparrow}}$ *surface consists exactly of carrier admission; two ODE source exposures through* D_0, D_1*; shared interval exposure through* I*; monotonicity or majorant exactness through* L, M*; Gronwall and Sturm handoff exactness through* G, S*; regular rational readback through* R*; real-name sealing through* E*; componentwise transport through* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier tuple of Definition 8036.1. Each listed obligation is one coordinate of the finite packet or the local naming row attached to that coordinate.

The two ODE source obligations are exactly the projections D_0 and D_1, and the common-domain obligation is the projection I. The comparison ledger is exhausted by L and M, so no additional monotonicity or majorant witness remains outside the displayed rows.

The downstream handoff rows are G for Gronwall-style comparison and S for Sturm-style zero-order comparison. Readback and terminal sealing are then forced through R and E.

Finally H, C, P, N transport, replay, package, and name the same displayed tuple. Coordinate exhaustion leaves no host solution equality, selected flow, quotient path, trichotomy, global existence, or ambient real-completeness coordinate to export. □

Theorem 8036.3 (ODE comparison Gronwall and Sturm handoff)**.** *Every downstream local-flow or zero-window comparison consumer of an accepted* $\mathsf{OdeComparison}^{\uparrow}$ *packet must first read the displayed* $\mathsf{Ode}^{\uparrow}$ *source rows* D_0, D_1*, the shared interval row* I*, the* $\mathsf{GronwallInequality}^{\uparrow}$ *or* $\mathsf{SturmComparison}^{\uparrow}$ *handoff row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger underlying the displayed comparison windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The consumer receives only the finite comparison route*

$$D_0, D_1, I, L, M, G, S, R, E, H, C, P, N$$

and no host solution equality, global existence theorem, selected flow, quotient path, trichotomy split, or ambient real-completeness principle.

Proof. Begin with the carrier projection of Definition 8036.1. The source rows D_0, D_1 and the shared window row I are the only places where the comparison request can read ODE and interval data.

For a local-flow comparison, the consumer reads the majorant ledger M through the Gronwall row G. For a zero-window comparison, it reads the order or oscillation ledger L through the Sturm row S. In both cases the finite windows then pass through the dyadic tolerance ledger, the regular rational readback row R, and the terminal Real seal E.

The rows H, C, P, N preserve the same packet by componentwise transport, displayed replay, provenance, and local naming. Since the packet has no coordinate for host solution equality, global existence, selected flows, quotient paths, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no downstream comparison consumer can read such data from this handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{OdeComparison}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OdeComparison}^{\uparrow}$ packet is generated from two displayed Ode source rows, one shared interval row, a located-order or majorant ledger, Gronwall or Sturm handoff rows, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OdeComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8036.1, Definition 8036.2, and Definition 8036.3 bind the finite ODE comparison carrier, its local obligation surface, and the displayed Gronwall or Sturm handoff route.

Formal verification: FormalStatus(NameCert$_{\mathsf{OdeComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host solution equality, global existence, selected flows, quotient paths, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Formalization requires a Lean carrier and checked NameCert obligations for the displayed ODE source, comparison ledger, Gronwall or Sturm handoff, readback, sealing, transport, replay, provenance, and naming rows.

Chapter 8037

A Concrete Naming Certificate for $\mathsf{CauchyRateSpace}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyRateSpace}^{\uparrow}$ packet records the explicit rate ledger carried by a regular Cauchy presentation before any terminal real-name seal is read. It sits between the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite rate substrate for L10 Real and Completion consumers, not a quotient stream, a selected limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8037.1 (Cauchy rate space carrier)**.** *A* $\mathsf{CauchyRateSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, W, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the rate ledger, storing the unary precision schedule and the finite tail thresholds licensed for each requested tolerance.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes only the inspected stream windows needed by the ledger.*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the displayed windows into regular rational observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-comparison row used to compare the requested precision with the displayed tail budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after L, W, S, D have been displayed.*
6. *H stores componentwise* hsame *transport for the rate ledger, stream windows, regular readback, dyadic comparisons, Real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a Real-facing Cauchy-rate query through L, W, S, D, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRateSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and their structural provenance.

Theorem 8037.2 (CauchyRateSpace NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateSpace}^{\uparrow}$ *carrier* $R = (L, W, S, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRateSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, classifier stability through* L, W, S, D, E*, rate-ledger exposure through* L*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. A consumer may read a Real-facing Cauchy rate only after the rate ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparison, and terminal Real seal rows are displayed.*

Proof. Project the carrier of Definition 8037.1. Its analytic rows are exactly the rate ledger, stream windows, regular rational readback, dyadic tolerance comparison, and terminal Real seal.

The classifier-stability and ledger clauses read those rows in the displayed order. The structural rows then transport, replay, source, and name the same finite packet through H, C, P, N.

Since no coordinate stores a quotient stream, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 8037.3 (CauchyRateSpace regularization handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyRateSpace}^{\uparrow}$ *carrier must first read the displayed route*

$$L \longmapsto W \longmapsto S \longmapsto D \longmapsto E.$$

Thus the rate ledger and $\mathsf{StreamName}^{\uparrow}$ *windows are visible before any regular-rational readback or terminal Real seal is consumed. The handoff exports no quotient stream equality, selected representative, countable-choice rate selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer request for either the regular rational readback or the terminal Real seal. The carrier has a single readback row S and a single seal row E, and both are downstream of the rate ledger, stream windows, and dyadic comparison.

The continuation row C replays the route through L, W, S, D, E, while H preserves the same rows under componentwise transport. The provenance row P and local name N identify the route as the $\mathsf{CauchyRateSpace}^{\uparrow}$ packet.

The excluded objects would require extra carrier coordinates. Because the packet contains only finite BEDC rows and their structural provenance, no consumer can bypass the ledger and window rows to obtain a quotient stream, selected limit, choice-selected rate, host equality, or ambient completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRateSpace}^{\uparrow}$ packet is generated from a finite rate ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparisons, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRateSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8037.1, Definition 8037.2, and Definition 8037.3 name the finite Cauchy-rate substrate used before regular-rational and Real-facing reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRateSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable-choice rate selectors, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, rate-ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRateSpaceUp.

Chapter 8038

A Concrete Naming Certificate for $\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}$

Following Bishop–Bridges compact metric analysis, the $\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}$ packet records the finite center-indexed radius chain that lies between local pointwise moduli and the rational lower-bound fold in the compact-uniform-continuity route of chapter 9237. It consumes compact metric source rows from chapter 129, pointwise-modulus rows from chapter 130, finite lower-bound folding from chapter 619, and the metric-triangle handoff of chapter 1159. The packet names a finite radius-chain surface before a $\mathsf{UniformModulus}^{\uparrow}$ consumer reads the global handoff; it is not open-cover compactness, a selected Lebesgue number, countable choice, or ambient completeness.

Definition 8038.1 (CompactUniformPointwiseRadiusChain carrier)**.** *A* $\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, A, L, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the finite-net centers and their metric coverage data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the displayed centers.*
3. *A is the center-indexed radius-chain row that records, for each displayed center, the local radius budget and its successor comparison.*
4. *L is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold boundary consumed by the finite-net minimum fold.*
5. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *handoff row that turns centerwise estimates into the uniform comparison route.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *export row reached only after K, F, A, L, T have been displayed.*
7. *H stores componentwise* hsame *transport for the compact source, pointwise modulus, radius chain, lower-bound fold, triangle estimate, uniform export, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a compact-uniform query through K, F, A, L, T, U, H.*

9. *P is the* Pkg *provenance over* CompactUniformPointwiseRadiusChain$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows, their transports, continuation replay, package provenance, and local naming.

Theorem 8038.2 (CompactUniformPointwiseRadiusChain obligations)**.** *For every accepted* CompactUniformPoint *carrier $R = (K, F, A, L, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source exposure through K, pointwise-modulus exposure through F, center-indexed radius-chain exposure through A, rational lower-bound fold boundary through L, metric-triangle handoff through T, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8038.1. The compact source, pointwise-modulus rows, radius chain, lower-bound fold, triangle estimate, and uniform export are the only analytic rows visible to a consumer.

Read the compact-uniform route in the displayed order. The compact source row K supplies the finite centers, the pointwise-modulus row F supplies local bounds at those centers, the radius-chain row A records the centerwise budgets, the fold boundary L exposes their rational lower bound, and the triangle handoff T produces the uniform-modulus export U.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Because no coordinate stores open-cover compactness, a selected Lebesgue number, countable choice, quotient equality, or ambient completeness, the listed rows exhaust the NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}}$.

Constructive story: A CompactUniformPointwiseRadiusChain$^{\uparrow}$ packet is generated from compact metric source rows, continuous-map pointwise-modulus rows, center-indexed radius-chain rows, a RatUp lower-bound fold boundary, a MetricTriangleUniformEstimate handoff, a UniformModulus export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}}$, seed)

Scope closed: Definition 8038.1 and Definition 8038.2 name the finite radius-chain packet between local moduli and the global compact-uniform modulus handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformPointwiseRadiusChain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, countable choice, quotient equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, pointwise-modulus exposure, radius-chain exposure, rational fold boundary, triangle handoff, uniform export, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformPointwiseRadiusChainUp.

Chapter 8039

A Concrete Naming Certificate for $\mathsf{ChristoffelFunction}^{\uparrow}$

Following Szego and the finite orthogonal-polynomial formulation of Gaussian quadrature, the $\mathsf{ChristoffelFunction}^{\uparrow}$ packet records the reciprocal diagonal kernel row used to control polynomial evaluation from displayed orthogonal-polynomial data. It sits after the recurrence and moment ledgers of chapter 2264, the node-weight rule of chapter 7017, the coefficient source of chapter 50, the terminal scalar seal of chapter 38, and the dyadic comparison route of chapter 291. The packet names only finite kernel, denominator, tolerance, and replay rows; it does not claim measure-theoretic completeness, spectral completeness, selected roots, quotient polynomial equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8039.1 (Christoffel function carrier)**.** *A* $\mathsf{ChristoffelFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, Q, P, X, D, L, E, H, C, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *source row, displaying the finite orthogonal-polynomial window, normalization ledger, recurrence coefficients, and moment reads used by the kernel.*
2. *Q is the* $\mathsf{GaussQuadrature}^{\uparrow}$ *source row, exposing the finite node-weight rule through which quadrature-facing consumers read Christoffel weights and diagonal kernel values.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *coefficient row for the polynomial family and for the finite products inspected by the diagonal kernel.*
4. *X is the evaluation-window row. It lists the displayed real-name evaluation points and the finite polynomial rows inspected at those points.*
5. *D is the diagonal denominator ledger, displaying the finite sum of normalized square evaluations or the equivalent reciprocal-weight row attached to the same orthogonal-polynomial window.*
6. *L is the local Christoffel-value row. It records the displayed reciprocal read of D as a* $\mathsf{Real}^{\uparrow}$*-sealed scalar whenever the denominator row is admitted.*
7. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row for denominator positivity, reciprocal tolerance, and node-weight comparison.*
8. *H stores componentwise* hsame *transport for O, Q, P, X, D, L, E and the local naming data.*

9. *C stores displayed* Cont *replay from orthogonal-polynomial source rows through evaluation, denominator formation, reciprocal readback, quadrature handoff, dyadic comparison, provenance, and local naming.*

10. *N is the combined* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{ChristoffelFunction}^\uparrow}$ *row over* $\mathsf{ChristoffelFunction}^\uparrow$, $\mathsf{OrthogonalPolynomial}^\uparrow$, $\mathsf{GaussQuadrature}^\uparrow$, $\mathsf{Poly}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert.

The classifier compares accepted packets only through $O, Q, P, X, D, L, E, H, C, N$. It has no coordinate for a host measure space, an infinite orthogonal expansion, a spectral-completeness theorem, quotient polynomial equality, selected quadrature nodes outside Q, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8039.2 (Christoffel function NameCert obligations)**.** *For every accepted* $\mathsf{ChristoffelFunction}^\uparrow$ *carrier*

$$K = (O, Q, P, X, D, L, E, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChristoffelFunction}^\uparrow}$ *surface consists exactly of the following five obligation rows:*

1. *source admission: the orthogonal-polynomial row O, quadrature row Q, and polynomial coefficient row P must be displayed before a Christoffel read is available;*

2. *evaluation admission: the point row X must name the finite evaluation windows and the polynomial rows inspected at those points;*

3. *denominator exactness: the row D must display the finite diagonal-kernel denominator or the equivalent reciprocal-weight row over the same source window;*

4. *reciprocal and comparison exposure: the row L must be read only after D, and the dyadic comparison row E must carry the displayed positivity, reciprocal tolerance, and node-weight comparisons;*

5. *preservation and naming: the transport, replay, provenance, and local naming rows H, C, N must preserve exactly O, Q, P, X, D, L, E and must not add a host measure space, spectral-completeness theorem, selected root solver, quotient polynomial equality, or ambient real-completeness principle.*

No Christoffel-function obligation is stored outside these rows.

Proof. Project the carrier of Definition 8039.1. The source rows are O, Q, P, so the packet starts from an already displayed orthogonal-polynomial window, quadrature rule, and polynomial coefficient ledger.

The evaluation and denominator rows are then finite. The row X lists the inspected evaluation points and polynomial rows, while D records the diagonal denominator or reciprocal-weight expression over the same displayed source window.

The reciprocal read is local to L and the comparison data are local to E. These rows expose only the admitted denominator, tolerance, and node-weight comparisons; they do not select roots or import an analytic measure object.

Finally, H, C, N transport, replay, source, and name the same finite rows. Since the carrier has no coordinate for measure-theoretic completeness, spectral completeness, quotient polynomial equality, selected roots outside the quadrature row, or ambient $\mathsf{Real}^\uparrow$ completeness, the five listed rows exhaust the NameCert surface. □

Theorem 8039.3 (Christoffel function quadrature handoff)**.** *Every quadrature-facing read of an accepted* $\mathsf{ChristoffelFunction}^\uparrow$ *packet factors through the finite route*

$$\mathsf{OrthogonalPolynomial}^\uparrow \longmapsto \mathsf{GaussQuadrature}^\uparrow \longmapsto X \longmapsto D \longmapsto L.$$

The consumer receives the displayed source rows, evaluation windows, diagonal denominator, reciprocal scalar, dyadic comparison row, transport, replay, provenance, and local name, but receives no measure-theoretic completeness, spectral completeness, quotient polynomial equality, selected root solver, undisplayed node family, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an integration- or quadrature-facing read. The packet first exposes the orthogonal-polynomial source O and the Gauss-quadrature source Q, so the consumer reads the Christoffel data only after the recurrence, moment, node, and weight ledgers are already visible.

Next inspect the evaluation and denominator rows. The row X fixes the displayed evaluation window, and D forms the finite diagonal-kernel denominator over that same window.

The local value row L records the reciprocal read, and E supplies the dyadic comparison data needed to inspect positivity and reciprocal tolerance. The rows H, C, N preserve and name this route without adding any further analytic principle. Hence every quadrature-facing read remains inside the displayed finite handoff. □

Falsifiable prediction: Every $\mathsf{ChristoffelFunction}^{\uparrow}$ read factors through an $\mathsf{OrthogonalPolynomial}^{\uparrow}$ source row, a $\mathsf{GaussQuadrature}^{\uparrow}$ node-weight row, a $\mathsf{Poly}^{\uparrow}$ coefficient row, a finite evaluation window, a diagonal denominator ledger, a reciprocal $\mathsf{Real}^{\uparrow}$ value row, dyadic comparison data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: orthogonal-polynomial, gauss-quadrature, polynomial, real, dyadicratcore: $\mathsf{ChristoffelFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the orthogonal-polynomial source, quadrature source, evaluation window, diagonal denominator, reciprocal value, dyadic comparison, transport, replay, provenance, and local naming rows as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ChristoffelFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{ChristoffelFunction}^{\uparrow}$ packet is generated from an OrthogonalPolynomialUp source row, a GaussQuadratureUp node-weight source row, a PolynomialUp coefficient row, a finite evaluation window, a diagonal denominator ledger, a reciprocal RealUp value row, DyadicRatCoreUp comparison data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChristoffelFunction}^{\uparrow}}$, seed)

Scope closed: Definition 8039.1, Definition 8039.2, and Definition 8039.3 seed the finite Christoffel-function certificate over displayed orthogonal-polynomial, quadrature, polynomial, real-scalar, and dyadic comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChristoffelFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure-theoretic completeness, spectral completeness, quotient polynomial equality, selected root solvers, infinite orthogonal expansions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source admission, evaluation-window admission, denominator exactness, reciprocal readback, quadrature handoff, dyadic comparison replay, transport, non-escape, and field faithfulness for BEDC.Derived.ChristoffelFunctionUp.

Chapter 8040

A Concrete Naming Certificate for $\mathsf{CauchyNetCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria and Moore–Smith directed convergence, the $\mathsf{CauchyNetCriterion}^{\uparrow}$ packet records the finite route by which directed Cauchy-net data becomes usable by the L10 regular-sequence and filter-basis machinery. It sits between the Cauchy-filter surface of chapter 333, the real-uniformity rows of chapter 1624, the stream-window naming of chapter 126, the regular rational readback of chapter 301, the public real seal of chapter 38, and the completion handoff of chapter 1506. The chapter names only the displayed directed-tail criterion and its two finite handoffs; it does not provide quotient equality of nets, selected cofinal sequences, countable choice, or an ambient completion principle.

Definition 8040.1 (CauchyNetCriterion carrier)**.** *A* $\mathsf{CauchyNetCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, A, U, T, F, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row naming the finite preorder comparisons, upper-window witnesses, and tail requests inspected by the criterion.*
2. *A is the net-value row, exposing the finite values whose common-tail comparisons are read by downstream consumers.*
3. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage row, displaying the uniform tolerance requests used to compare late net values.*
4. *T is the Cauchy-tail ledger, recording the displayed common-tail witnesses for each requested entourage in U.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row obtained from the directed tails and their common-window evidence.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row extracted from finite directed tails without choosing a global cofinal sequence.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from D, A, U, T, S.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after regular-rational readback has been displayed.*
9. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that lets completion-facing consumers read the same Cauchy evidence as a completion criterion.*

10. *H records componentwise* hsame *transport for directed indices, net values, uniformity rows, Cauchy-tail ledgers, filter bases, stream windows, regular readback, real seals, completion handoffs, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay through the directed-tail, filter-basis, regular-sequence, real-seal, and completion routes.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyNetCriterion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, MetricCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetCriterion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose criterion consumer first displays $D, A, U, T, F, S, R, E, M, H, C, P, N$. It has no coordinate for a selected cofinal sequence, quotient-net equality, countable choice, host completion, or ambient completeness.

Theorem 8040.2 (CauchyNetCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetCriterion}^{\uparrow}$ *carrier*

$$Q = (D, A, U, T, F, S, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyNetCriterion}^{\uparrow}}$ *surface consists exactly of directed-index admission, net-value exposure,* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage exposure, Cauchy-tail ledger exposure,* $\mathsf{CauchyFilter}^{\uparrow}$ *basis handoff,* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal readback,* MetricCompletion$^{\uparrow}$ *completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of Q.*

Proof. Project the carrier of Definition 8040.1. The directed analytic data are exactly D, A, U, T: indices, values, entourages, and common-tail witnesses.

The filter-basis row F is read only after those directed rows have been displayed. It records the finite basis generated by tails and common-window evidence, not a quotient of nets or a completed limit object.

The sequence-facing rows S, R, E are then reached in order. Stream windows are extracted from displayed finite tails, the regular-rational handoff repacks those windows, and the real seal is read only after the regular row is present.

The completion-facing row M reads the same Cauchy evidence. The structural rows H, C, P, N transport, replay, source, and name the displayed route, so no hidden cofinal selector, quotient equality, choice principle, host completion, or ambient completeness witness can discharge a NameCert obligation. □

Theorem 8040.3 (CauchyNetCriterion regular-sequence handoff)**.** *Every regular-sequence consumer of an accepted* $\mathsf{CauchyNetCriterion}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealUniformStructure}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route uses the displayed directed-tail rows D, A, U, T and the structural rows H, C, P, N of the same packet; it does not export a selected cofinal sequence or a quotient equality of net limits.

Proof. Begin with a regular-sequence request. The carrier has no independent sequence coordinate, so the request must read the directed-tail rows D, A, U, T to obtain finite late-window evidence.

The stream row S supplies the only admissible sequence-facing window. It is extracted from displayed directed tails and is therefore still tied to the same common-tail ledger.

The handoff row R repacks the stream observations as regular rational readback, and E seals that readback as the Real-facing output. Transport, replay, provenance, and local naming through H, C, P, N preserve this route without adding a cofinal selector or quotient-limit equality. □

Theorem 8040.4 (CauchyNetCriterion filter-basis handoff)**.** *Every filter-basis or completion-facing consumer of an accepted* $\mathsf{CauchyNetCriterion}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealUniformStructure}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow},$$

using the displayed rows $D, A, U, T, F, M, H, C, P, N$*. The handoff exposes finite Cauchy-filter basis data and the completion criterion route, but it does not assert host completeness, arbitrary net convergence, selected subnet existence, or quotient equality of completion points.*

Proof. Read the uniformity and tail rows first. The entourage row U names the comparison requests, while T supplies the common-tail witnesses over the directed index row D and the net-value row A.

The Cauchy-filter row F is generated from those finite tails. A filter-basis consumer therefore sees only displayed basis elements and their Cauchy compatibility rows.

The completion handoff M reads the same basis evidence as a $\mathsf{MetricCompletion}^{\uparrow}$ criterion. Since H, C, P, N only transport, replay, source, and name the listed rows, the route cannot claim host completeness, arbitrary convergence, selected subnets, or quotient equality of completion points. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNetCriterion}^{\uparrow}$ packet is generated from directed indices, net values, RealUniformStructure entourages, Cauchy-tail ledgers, CauchyFilter basis rows, StreamName windows, RegSeqRat readback, Real seals, MetricCompletion handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyNetCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 8040.1, Definition 8040.2, Definition 8040.3, and Definition 8040.4 seed the L10 route from directed Cauchy-net data to regular-sequence and filter-basis consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyNetCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected cofinal sequences, selected subnets, quotient-net equality, quotient equality of completion points, countable choice, host completion, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail exposure, uniformity tolerance exposure, Cauchy-filter basis handoff, StreamName extraction, RegSeqRat/Real readback, MetricCompletion handoff, transport preservation, replay, provenance, and local naming for BEDC.Derived.CauchyNetCriterionUp.

Chapter 8041

A Concrete Naming Certificate for $\mathsf{RegularCauchyCutoff}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyCutoff}^{\uparrow}$ packet records the finite tail-cutoff data by which a regular Cauchy rational name becomes readable by Real and Completion consumers. It sits after the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert carrier for a displayed cutoff route; it is not quotient stream equality, a selected limit representative, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8041.1 (Regular Cauchy cutoff carrier)**.** *A* $\mathsf{RegularCauchyCutoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (n, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *n is the finite cutoff index at which the tail route begins.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule inspected at and beyond n.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed tail.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the requested cutoff precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after n, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the cutoff index, window schedule, regular rational readback, dyadic tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion request through the cutoff route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCutoff}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCutoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $n, W, R, D, E, H, C, P, N$. It has no coordinate for quotient-name equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8041.2 (Regular Cauchy cutoff tail window)**.** *Every accepted* $\mathsf{RegularCauchyCutoff}^{\uparrow}$ *carrier*

$$K = (n, W, R, D, E, H, C, P, N)$$

exposes a finite tail window before any Real-facing seal can be read. A consumer must inspect the cutoff index n*, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *before it reaches* E, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCutoffUp.RegularCauchyCutoffCarrier tail window`

Proof. Project the accepted carrier of Definition 8041.1. The first four rows display the cutoff index, stream window, regular rational readback, and dyadic tolerance ledger as finite data.

The terminal seal row E is downstream of those displayed rows. Transport, replay, provenance, and local naming preserve the same order because H, C, P, N act on the carried rows rather than replacing them with a hidden tail selector.

Thus an accepted consumer can read the Real-facing seal only through the displayed finite tail window. Any route that omits n, W, R, D has no carrier coordinate to project and is outside the local certificate. □

Theorem 8041.3 (Regular Cauchy cutoff RegSeqRat handoff)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{RegularCauchyCutoff}^{\uparrow}$ *packet is attached to the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that determine the cutoff. The route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

therefore remains a displayed finite NameCert route and does not introduce a quotient of streams, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the tail-window theorem. It forces each Real-facing read to pass through the same rows n, W, R, D before the seal row is available.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is not a standalone limit witness. It is read together with the finite window W and tolerance ledger D, and the replay row C preserves that attachment for downstream Real and Completion requests.

Since the carrier has no coordinate for quotient stream equality, selected representatives, countable choice, or ambient completeness, the handoff can export only the displayed finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the local structural rows. □

Theorem 8041.4 (Regular Cauchy cutoff non-escape)**.** *No accepted* $\mathsf{RegularCauchyCutoff}^{\uparrow}$ *consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal without first reading the cutoff index, finite stream window, regular rational row, dyadic tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. A claimed Real-facing route that bypasses any of those rows is outside the packet.*

Proof. Start with the carrier inventory. The Real seal is one displayed row of K, and the rows n, W, R, D, H, C, P, N specify the finite route by which it is exposed.

Apply the handoff theorem. It ties the regular rational readback to the same stream window and dyadic tolerance ledger that determine the cutoff, so the Real-facing consumer cannot detach E from the tail window.

The remaining rows transport, replay, source, and name that route. Omitting any of them changes the carrier rather than preserving an accepted $\mathsf{RegularCauchyCutoff}^{\uparrow}$ packet, so the attempted bypass is refuted by the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCutoff}^{\uparrow}$ consumer is refuted if it reaches the $\mathsf{Real}^{\uparrow}$ seal without first reading the finite cutoff index, $\mathsf{StreamName}^{\uparrow}$ window schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, real: none is bijective with $\mathsf{RegularCauchyCutoff}^{\uparrow}$ because the cutoff packet jointly binds one finite cutoff index, one finite stream window, one regular rational row, one dyadic tolerance ledger, one Real seal, transport, replay, provenance, and local naming as a single tail-cutoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCutoff}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCutoff}^{\uparrow}$ packet is generated from a finite cutoff index, StreamName finite-window schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCutoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8041.1, Definition 8041.2, Definition 8041.3, and Definition 8041.4 seed the finite tail-cutoff route over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCutoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove tail-window exposure, RegSeqRat handoff preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCutoffUp.

Chapter 8042

A Concrete Naming Certificate for LawlikeLocatedRealSequence$^{\uparrow}$

Following Bishop and Brouwer on lawlike sequences in constructive analysis, the LawlikeLocatedRealSequence$^{\uparrow}$ packet records a finite interface for a displayed lawlike sequence of located real names. It sits over the stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the located-real comparison surface of chapter 1933. The packet names the sequence interface used by Real and Completion consumers; it does not choose a host sequence, quotient streams, select limits, use countable choice, or assert ambient completeness of Real$^{\uparrow}$.

Definition 8042.1 (Lawlike located real sequence carrier)**.** *A* LawlikeLocatedRealSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, D, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the lawlike source row, recording the finite rule ledger and admissible request boundary for the indexed sequence.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row, listing the inspected sequence indices and the finite observation windows available at those indices.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying the dyadic precision budget for each inspected located real name.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance data.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *Q is the* LocatedReal$^{\uparrow}$ *row, exposing the located comparison or bracketing surface attached to the same Real seal.*
7. *H records componentwise* hsame *transport for lawlike source data, stream windows, dyadic tolerances, regular readbacks, Real seals, located comparisons, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from lawlike sequence inspection through finite windows, dyadic tolerances, regular readback, Real sealing, and located comparison.*

9. *P is the* Pkg *provenance over* $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LawlikeLocatedRealSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, R, E, Q, H, C, P, N$. It has no coordinate for a selected host sequence, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8042.2 (Lawlike located real sequence RegSeqRat readback)**.** *For every accepted* LawlikeLocatedRealSequ *carrier $L = (S, W, D, R, E, Q, H, C, P, N)$, a Real-facing or LocatedReal-facing consumer may read the regular rational data only after the lawlike source row S, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D have been displayed. The readback row R is therefore the unique local route from lawlike finite observations to the terminal* $\mathsf{Real}^{\uparrow}$ *seal E and the located comparison row Q.*

Proof. Project the carrier of Definition 8042.1. The rows S, W, D expose the lawlike source, inspected stream windows, and dyadic tolerance budget before any rational or real-facing row is available.

The readback row R is the only row typed as $\mathsf{RegSeqRat}^{\uparrow}$ data in the packet. It repacks the displayed windows and tolerance ledger as regular rational finite data, so a consumer cannot obtain a regular readback by bypassing S, W, D.

The endpoint rows E and Q are downstream of the same readback. The Real seal E and located comparison row Q may be consumed only after R, while H, C, P, N transport, replay, source, and name the same finite route. □

Theorem 8042.3 (Lawlike located real sequence choice-free modulus)**.** *Every modulus or comparison read exported by an accepted* $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$ *carrier is bounded by the displayed finite route*

$$S, W, D, R, E, Q, H, C, P, N.$$

The route may expose dyadic tolerance budgets, regular rational readbacks, Real seals, and located comparison rows for inspected windows, but it exports no selected infinite subsequence, countable-choice selector, quotient stream equality, selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the accepted carrier. Its only analytic rows are the lawlike source S, finite stream windows W, dyadic tolerance budget D, regular readback R, Real seal E, and located comparison row Q.

A modulus request can inspect only the displayed tolerance row D and its attached finite windows. By Definition 8042.2, any rational readback then passes through R before the Real or located endpoint is consumed.

The structural rows H, C, P, N preserve this packet under componentwise transport, displayed replay, provenance, and local naming. Since the carrier has no coordinate for an infinite selector, quotient stream equality, selected limit, trichotomy, or ambient completeness, every exported modulus remains choice-free and finite-window based. □

Falsifiable prediction: Every accepted $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{LocatedReal}^{\uparrow}$ data without displaying the lawlike source row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, structural replay, provenance, and local NameCert evidence refutes the carrier of Definition 8042.1.

Independence witness: streamname, regseqrat, dyadicratcore, real, located-real: $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a lawlike source row, finite sequence-window ledger, dyadic tolerance budget, regular readback, Real seal, located comparison row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeLocatedRealSequence}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$ packet is generated from a lawlike source row, finite StreamName windows, DyadicRatCore tolerance budgets, RegSeqRat readback, Real seals, LocatedReal comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawlikeLocatedRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8042.1, Definition 8042.2, and Definition 8042.3 seed the finite lawlike located-sequence interface over StreamName windows, dyadic tolerance budgets, RegSeqRat readback, Real seals, LocatedReal comparison rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawlikeLocatedRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected infinite subsequences, countable choice, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window compatibility, dyadic tolerance exactness, RegSeqRat readback, LocatedReal comparison stability, transport replay, provenance, and nonescape for BEDC.Derived.LawlikeLocatedRealSequenceUp.

Chapter 8043

A Concrete Naming Certificate for $\mathsf{MonotoneBoundedConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MonotoneBoundedConvergence}^{\uparrow}$ packet records the finite route by which a bounded monotone real sequence is handed to a located supremum surface and then read back as a regular Cauchy real name. It sits beside the monotone bounded sequence carrier of chapter 3407, the located-supremum handoff of chapter 1427, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet records a convergence certificate for the displayed sequence route; it does not claim Dedekind completeness, trichotomy, quotient equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8043.1 (Monotone bounded convergence carrier)**.** *A* $\mathsf{MonotoneBoundedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, L, B, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *source row, exposing the inspected real-sequence windows and their monotonicity ledger.*
2. *L is the monotonicity ledger stability row, recording that the displayed order comparisons are preserved under the finite window extensions used downstream.*
3. *B is the boundedness exactness row, naming the upper-bound evidence available for the same displayed sequence route.*
4. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row attached to the bounded monotone source, not to an ambient host set.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the located-supremum handoff and the displayed sequence windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after S, L, B, U, R have been displayed.*
7. *H stores componentwise* hsame *transport for the sequence source, ledger, boundedness row, located-supremum handoff, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a convergence request through S, L, B, U, R, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{MonotoneBoundedConvergence}^{\uparrow}$, $\mathsf{MonotoneBoundedSequence}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneBoundedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their structural provenance.

Lean4 (statement) `BEDC.Derived.MonotoneBoundedConvergenceUp`

Theorem 8043.2 (MonotoneBoundedConvergence obligation surface)**.** *For every accepted* MonotoneBoundedConve *carrier $M = (S, L, B, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonotoneBoundedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, monotonicity ledger stability through L, boundedness exactness through B, located-supremum handoff through U, regular-Cauchy real-seal readback through R, E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is the projection of Definition 8043.1. The analytic route is the displayed tuple S, L, B, U, R, E, and no convergence-facing consumer can enter the packet before those rows are named.

The monotonicity ledger remains stable because L is attached to the same $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ source row S used by every later window read. Extending a displayed window therefore preserves the listed order comparisons rather than opening a second sequence classifier.

Boundedness is exact at row B. The upper-bound evidence is read from the same finite source route as the monotonicity ledger, so it cannot be replaced by a host least-upper-bound theorem, trichotomy, or an unrecorded completeness principle.

The located-supremum handoff is row U. It consumes the monotone bounded source through S, L, B and exports only the located supremum boundary needed for the downstream regular readback.

Finally the regular-Cauchy real-seal readback passes through R, E, H, C, P, N. Row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ presentation obtained from the handoff, row E supplies the terminal $\mathsf{Real}^{\uparrow}$ seal, and the structural rows transport, replay, source, and name the same route. Since no coordinate stores Dedekind completeness, quotient equality, countable choice, or ambient real completeness, the listed rows exhaust the local obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneBoundedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneBoundedConvergence}^{\uparrow}$ packet is generated from a MonotoneBoundedSequence source row, a monotonicity ledger stability row, boundedness exactness, a LocatedSupremum handoff, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneBoundedConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 8043.1 and Definition 8043.2 name the finite convergence route from bounded monotone sequence data to located-supremum and regular-Cauchy real-seal readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneBoundedConvergence}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, trichotomy, quotient equality, countable choice, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotonicity ledger stability, boundedness exactness, located-supremum handoff, regular readback, Real seal, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneBoundedConvergenceUp.

Chapter 8044

A Concrete Naming Certificate for $\mathsf{VisionLedgerRoute}^{\uparrow}$

The $\mathsf{VisionLedgerRoute}^{\uparrow}$ packet realizes the top-level visions hub of chapter 9236 as finite route data over the concrete-instance ledger. It records the source hub row from `parts/visions/_index_files.tex`, the concrete target label, the hub-input route, the displayed marker row, and the replay/provenance rows that let a coverage audit move from the visions index to the concrete-instance chapter without consulting cached counts. The packet sits beside the coverage index of chapter 7381 and the coverage ledger of chapter 7484; it is a route certificate for one hub-to-target edge, not a coverage theorem for all vision files.

Definition 8044.1 (Vision ledger route carrier)**.** *A* $\mathsf{VisionLedgerRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, I, T, M, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-hub row naming the top-level visions index under* `parts/visions/_index_files.tex`*.*
2. *I is the input-route row, listing the finite hub edge by which the source hub reaches a thematic visions subhub or leaf.*
3. *T is the concrete-target row naming the realized concrete-instance chapter label.*
4. *M is the marker row containing the displayed* `\concretizedIn` *route from the source hub to T.*
5. *L is the local route-ledger row, recording whether the edge is root-index, subhub, or leaf-facing coverage.*
6. *A is the audit-readback row that lets a consumer reread both the source hub and the target chapter before counting the route.*
7. *H records componentwise* hsame *transport for source hub, input route, target label, marker, route ledger, audit readback, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source hub to the concrete target through I, M, L, A, H.*
9. *P is the* Pkg *provenance over* $\mathsf{VisionLedgerRoute}^{\uparrow}$*,* $\mathsf{VisionCoverageIndex}^{\uparrow}$*,* $\mathsf{VisionCoverageLedger}^{\uparrow}$*, the referenced visions hub, the referenced concrete-instance target,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{VisionLedgerRoute}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, I, T, M, L, A, H, C, P, N$. It has no coordinate for hidden coverage totals, semantic endorsement of source prose, automatic coverage of sibling vision files, or a concrete target without a displayed chapter label.

Theorem 8044.2 (Vision ledger route NameCert obligations)**.** *For every accepted* $\mathsf{VisionLedgerRoute}^{\uparrow}$ *carrier $R = (S, I, T, M, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VisionLedgerRoute}^{\uparrow}}$ *surface consists exactly of source-hub admission, input-route exposure, concrete-target exposure, marker-row exposure, route-ledger classification, audit-readback exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of R.*

Proof. Project the carrier of Definition 8044.1. The source side is exactly S and I: a named visions hub and the finite input edge being audited.

The target side is exactly T and M. The concrete target label is displayed, and the marker row records the concretized-in route rather than inferring it from a title or nearby prose.

The route ledger L and audit row A make the route replayable. They classify the edge and let the consumer reread the hub and target before counting it.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no hidden total, semantic endorsement, sibling coverage, or target-without-label coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 8044.3 (Vision ledger route replay handoff)**.** *Every route consumer of an accepted* $\mathsf{VisionLedgerRoute}^{\uparrow}$ *packet factors through*

$$S \longmapsto I \longmapsto M \longmapsto T \longmapsto A \longmapsto H, C, P, N.$$

Thus the visions index contributes a concrete coverage route only through a displayed hub edge, a displayed marker row, and a replayable concrete target. The route exports no hidden coverage count, no source-prose semantics, and no coverage claim for unmarked siblings.

Proof. Start with the source-hub and input-route rows. A consumer first reads S and I, so the edge being counted is the displayed hub route rather than an implicit scan of the visions tree.

Next read the marker and target rows. The marker row M binds the source route to the concrete target T and prevents a cached target list from replacing the displayed route.

The audit row A is consumed before replay completes. It points the consumer back to the source hub and to the concrete-instance target, so the route remains inspectable.

The structural rows H, C, P, N preserve this order under transport, continuation replay, provenance, and local naming. Hence no hidden count, prose semantics, or unmarked sibling coverage can be exported. □

Theorem 8044.4 (Vision ledger route concrete-index boundary)**.** *For every accepted* $\mathsf{VisionLedgerRoute}^{\uparrow}$ *carrier $R = (S, I, T, M, L, A, H, C, P, N)$, the concrete target T must remain a concrete-instance NameCert chapter label and the source row S must remain the visions hub route named by the carrier. The packet may point from the visions index to the concrete-instances ledger, but it cannot move theorem content into the visions tree or treat a visions hub as a proof chapter.*

Proof. Project the target row T and the source row S. The carrier separates the visions hub from the concrete target, so a consumer can identify which side supplies route context and which side supplies the NameCert chapter.

Use the marker row M and route ledger L. They record the direction from source hub to concrete target and classify the route edge before the audit row is read.

The audit row A reopens both endpoints without merging their roles. Transport and replay through H and C preserve the same endpoints, while P and N source and name the route.

Therefore the packet is a concrete-index boundary. It records a route from the visions hub into concrete instances, but it does not relocate proof content into the visions tree or certify a visions hub as a proof chapter. □

Falsifiable prediction: Every accepted $\mathsf{VisionLedgerRoute}^{\uparrow}$ coverage read names a source hub row, input-route row, concrete target label, marker row, route ledger, audit-readback row, transport row, replay row, provenance row, and local NameCert row before it is counted.

Independence witness: vision-coverage-index, vision-coverage-ledger, affine-filter-hull, regular-cauchy-filter-seal: $\mathsf{VisionLedgerRoute}^{\uparrow}$ is not bijective to any listed sibling because it stores the top-level visions hub edge and concrete-index boundary as route data rather than storing a general coverage account, a mathematical hull, or an analytic Cauchy seal.

Lean Nontrivial and FieldFaithful TasteGate instances for $\mathsf{VisionLedgerRoute}^{\uparrow}$ should inspect the carrier rows directly: Nontriviality is witnessed by distinct source-hub and concrete-target rows, while field faithfulness requires projections of $S, I, T, M, L, A, H, C, P, N$ to preserve the displayed hub edge, marker route, and audit-readback boundary. Neither instance may read hidden coverage totals, cached source scans, source prose outside the marker row, or unmarked sibling files.

Closure status for $\mathsf{NameCert}_{\mathsf{VisionLedgerRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{VisionLedgerRoute}^{\uparrow}$ packet is generated from a source-hub row, input-route row, concrete-target row, marker row, route ledger, audit-readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VisionLedgerRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8044.1, Definition 8044.2, Definition 8044.3, and Definition 8044.4 seed the finite route packet that turns the visions hub edge into displayed concrete-instance coverage data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VisionLedgerRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total vision coverage, semantic approval of source prose, proof content inside the visions tree, Lean verification, bridge checking, hidden coverage totals, or coverage for unmarked sibling files.

Upgrade path: Obligation closure requires checked carrier admission, source-hub exactness, input-route exactness, target-label exactness, marker-route replay, route-ledger classification, audit-readback stability, transport, provenance, and local naming for BEDC.Derived.VisionLedgerRouteUp.

Chapter 8045

A Concrete Naming Certificate for HenstockKurzweilGaugeIntegral$^{\uparrow}$

Following Henstock and Kurzweil, the HenstockKurzweilGaugeIntegral$^{\uparrow}$ packet records the constructive gauge-integral route in which finite tagged partitions are admitted by a displayed gauge and finite Riemann sums are compared through regular rational readback before any Real-facing integral value is exported. It sits beside the gauge-integral packet of chapter 1999, the tagged-partition surface of chapter 4296, the Riemann-sum gauge row of chapter 4115, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dyadic tolerance core of chapter 291. The chapter names one finite gauge-integral certificate; it does not import measure completion, countable choice, selected tags, quotient equality of functions, ambient completeness, or a standard bridge.

Definition 8045.1 (Henstock–Kurzweil gauge-integral carrier)**.** *A* HenstockKurzweilGaugeIntegral$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, G, T, S, C, H, R, P, N)$$

with the following displayed rows:

1. *I is the interval source row, carrying the located endpoint window on which the integrand is inspected.*
2. *G is the gauge ledger, recording positive local gauge data and the dyadic tolerance requests used by the packet.*
3. *T is the* TaggedRealPartition$^{\uparrow}$ *row, exposing only finite tagged subintervals that are G-fine.*
4. *S is the* RiemannSumGauge$^{\uparrow}$ *row, carrying the finite Riemann-sum observations attached to the tagged partition.*
5. *C is the Cauchy sum-window row, recording finite comparison windows for pairs of G-fine sums.*
6. *H records componentwise* hsame *transport for the interval, gauge, partition, sum, Cauchy-window, readback, replay, provenance, and local naming rows.*
7. *R is the* RegSeqRat$^{\uparrow}$ *handoff row, read only after the finite Cauchy sum windows and dyadic tolerance ledger have been displayed.*
8. *P is the* Pkg *provenance over* HenstockKurzweilGaugeIntegral$^{\uparrow}$, GaugeIntegral$^{\uparrow}$, TaggedRealPartition$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, G, T, S, C, H, R, P, N$. It has no coordinate for Lebesgue measure, countable choice, selected tags, quotient equality of functions, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Theorem 8045.2 (Henstock–Kurzweil gauge-fine partition obligation)**.** *For every accepted* HenstockKurzweilGauge *carrier $K = (I, G, T, S, C, H, R, P, N)$, a partition-facing consumer must read the interval source I, the gauge ledger G, and the tagged finite partition row T before any Riemann-sum or Cauchy-window row is available. Thus gauge fineness is stored as displayed finite packet data, not as a choice-selected partition family or an ambient compactness principle.*

Proof. Project the carrier of Definition 8045.1. The only interval source is I, the only gauge data are in G, and the only partition data are the finite tagged windows in T.

The Riemann-sum row S is downstream from those three rows. A consumer therefore cannot inspect a sum before the interval window, gauge ledger, and tagged partition are visible.

Transport through H and provenance through P, N preserve the same row order. Since the carrier contains no selected tag family, open-cover compactness row, countable-choice schedule, or host partition quotient, the obligation is exactly the displayed finite gauge-fineness route. □

Theorem 8045.3 (Henstock–Kurzweil Cauchy criterion obligation)**.** *For every accepted* HenstockKurzweilGaugeInte *carrier $K = (I, G, T, S, C, H, R, P, N)$, a Real-facing integral read must pass through the finite Cauchy sum-window row C and the regular rational handoff R after the gauge-fine partition and Riemann-sum rows have been displayed. The packet exports only a Cauchy comparison route for displayed gauge-fine sums; it does not supply measure completion, quotient equality of functions, selected tags, countable choice, ambient completeness, or a standard bridge.*

Proof. Start with the gauge-fine route in Definition 8045.2. It exposes I, G, T before any sum row is read.

Next read the finite sum and Cauchy rows. The row S records the displayed Riemann sums, while C records the finite comparison windows used for the Cauchy criterion. Both rows remain tied to the same interval source, gauge ledger, and tagged partition.

Finally pass through R, H, P, N. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff records regular rational readback before any $\mathsf{Real}^{\uparrow}$ consumer can interpret the integral value, and the structural rows transport, source, and name the same route. None of the excluded measure, quotient, selection, choice, completeness, or bridge coordinates occurs in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}$ Real-facing read factors through interval source, gauge ledger, tagged fine partition, Riemann-sum gauge row, finite Cauchy sum windows, componentwise hsame transport, RegSeqRat handoff, provenance, and local NameCert rows.

Independence witness: gauge-integral, tagged-real-partition, riemann-sum-gauge, regseqrat, real, dyadicratcore: no listed sibling is bijective with $\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}$ because this carrier jointly binds gauge fineness, tagged finite partitions, Riemann-sum observations, Cauchy sum-window comparisons, regular rational readback, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}$ packet is generated from an interval source, gauge ledger, tagged fine partition row, Riemann-sum gauge row, finite Cauchy sum-window row, componentwise hsame transport, RegSeqRat handoff, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockKurzweilGaugeIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8045.1, Definition 8045.2, and Definition 8045.3 seed the finite gauge-integral route over tagged partitions, Riemann-sum windows, Cauchy sum comparisons, and regular rational readback.

Formal verification: FormalStatus(NameCert$_{\text{HenstockKurzweilGaugeIntegral}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lebesgue measure, countable choice, selected tags, quotient equality of functions, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, gauge-fineness exposure, tagged-partition admission, Riemann-sum readback, Cauchy sum-window comparison, RegSeqRat handoff, transport, provenance, and local naming for BEDC.Derived.HenstockKurzweilGaugeIntegralUp.

Chapter 8046

A Concrete Naming Certificate for $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$

The $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ packet realizes the residual-substitution confluence frontier of chapter 9386 as a finite local-diamond witness for one admitted residual pair. It sits beside the conditional parallel-diamond frontier of chapter 4238, the existing residual-diamond witness of chapter 7219, the residual-substitution boundary of chapter 2181, and the residual-checker packet of chapter 3827. The packet records bounded local-join visibility and obstruction retention; it does not claim global confluence, subject reduction, kernel consistency, unrestricted residual-substitution compatibility, evaluator equality, or quotient term equality.

Definition 8046.1 (MetaCIC residual local-diamond witness carrier)**.** *A* $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, P, J, O, H, C, G, N)$$

with the following displayed rows:

1. *A is the residual-pair admission row, naming the typed window, source term, two residual branches, and finite observation budget for the local diamond.*
2. *P is the residual-substitution boundary row retained from the confluence frontier.*
3. *J is the bounded local-join checker row, recording the visible join candidate and finite checker status for the admitted pair.*
4. *O is the obstruction-retention row, recording that the packet does not discharge global confluence, subject reduction, kernel consistency, or unrestricted residual-substitution compatibility.*
5. *H records componentwise* hsame *transport for the admission, boundary, checker, obstruction, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay through the route*

$$A, P \longmapsto J \longmapsto O.$$

7. *G is the* Pkg *provenance over* $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICResidualDiamondWitness}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through A, P, J, O, H, C, G, N. It has no coordinate for premise-free parallel diamond, global Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Theorem 8046.2 (MetaCIC residual local-diamond witness obligations)**.** *For every accepted* $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ *carrier* $L = (A, P, J, O, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalDia}}$ *surface consists exactly of residual-pair admission through A, residual-substitution boundary visibility through P, bounded local-join checker visibility through J, obstruction retention through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8046.1. The confluence-facing rows are exactly the residual-pair admission, retained substitution boundary, bounded local-join checker, and obstruction-retention rows.

The witness direction is local. The row A fixes the admitted residual pair and finite observation budget, while P keeps the residual-substitution boundary visible before the checker row is consumed.

The checker row J records only the bounded local join for that admitted pair. The obstruction row O remains attached, and the structural rows H, C, G, N transport, replay, source, and name the same finite packet. Since no global confluence, subject-reduction, quotient, evaluator, or consistency coordinate occurs, the listed rows exhaust the local NameCert surface. □

Theorem 8046.3 (MetaCIC residual local-diamond nonescape)**.** *Every confluence-frontier consumer of an accepted* $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ *packet must read*

$$A, P, J, O, H, C, G, N$$

before using the local-diamond witness. The handoff may feed the conditional $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *route, but it cannot erase the residual-substitution boundary, promote the bounded checker to global confluence, discharge subject reduction, or certify kernel consistency.*

Proof. Start with the admission row A. It exposes only one finite residual pair, its typed window, and the observation budget inspected by the packet.

Next read P, J, O. The residual-substitution boundary remains visible, the bounded checker supplies only local-join status for the admitted pair, and the obstruction row records every surrounding endpoint that remains outside the packet.

Finally read the structural rows. Transport through H, replay through C, provenance through G, and local naming through N preserve the same finite route. Thus the consumer receives a local-diamond witness with retained obstruction rows, not a global confluence or consistency theorem. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ consumer that reads a local join without displaying residual-pair admission, residual-substitution boundary visibility, bounded checker status, obstruction retention, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: metacic-parallel-diamond-frontier, metacic-residual-diamond-witness, metacic-residual-substitution-boundary, metacic-confluence-residual-checker: no listed sibling is bijective with $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ because this packet jointly carries residual-pair admission, retained substitution boundary, bounded local-join checker visibility, obstruction retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}$ packet is generated from residual-pair admission, residual-substitution boundary visibility, bounded local-join checker visibility, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8046.1, Definition 8046.2, and Definition 8046.3 seed the finite residual-pair local-diamond witness route for the MetaCIC confluence frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalDiamondWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, subject reduction, kernel consistency, unrestricted residual-substitution compatibility, evaluator equality, quotient term equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked residual-pair admission, residual-substitution boundary visibility, bounded local-join checker visibility, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualLocalDiamondWitnessUp.

Chapter 8047

A Concrete Naming Certificate for $\mathsf{HyperbolicBarycenter}^{\uparrow}$

Following Bridson–Haefliger and the standard finite comparison view of hyperbolic metric geometry, the $\mathsf{HyperbolicBarycenter}^{\uparrow}$ packet records a weighted barycenter readback as finite NameCert data. It sits after the metric rows of chapter 3474, the four-point coarse-negative-curvature surface of chapter 6862, the boundary-direction rows of chapter 3732, and the terminal scalar seal of chapter 38. The packet names a finite averaging-style route over displayed metric, boundary, weight, and seal rows; it is not a completed hyperbolic space, a quotient boundary measure, a selected minimizer in an ambient space, countable choice, or a global convexity theorem.

Definition 8047.1 (Hyperbolic barycenter carrier)**.** *A* $\mathsf{HyperbolicBarycenter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, W, U, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row for the finite family being averaged.*
2. *G is the* $\mathsf{GromovHyperbolicMetric}^{\uparrow}$ *comparison row, recording the four-point or coarse hyperbolicity budget used by the read.*
3. *W is the finite weight ledger, recording the nonnegative displayed weights and their finite normalization row.*
4. *U is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-direction row used when the barycenter is read through a boundary-facing shadow.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the scalar comparison budget attached to the weighted readback.*
6. *T is the transport row binding the same metric, hyperbolicity, weight, boundary, and scalar rows under componentwise comparison.*
7. *H records componentwise* hsame *transport for the metric, comparison, weight, boundary, scalar, transport, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a barycenter request through $M, G, W, U, R, T, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBarycenter}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{GromovHyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBarycenter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, G, W, U, R, T, H, C, P, N$. It has no coordinate for a host minimizer, quotient boundary probability measure, selected global convexity center, countable-choice family, or ambient hyperbolic completion.

Theorem 8047.2 (Hyperbolic barycenter NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBarycenter}^{\uparrow}$ *carrier*

$$B = (M, G, W, U, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBarycenter}^{\uparrow}}$ *surface consists exactly of metric-read admission, Gromov-hyperbolic comparison exposure, finite weight-ledger admission, Busemann-boundary handoff, Real-seal scalar budgeting, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8047.1. The rows $M, G, W, U, R, T, H, C, P, N$ are the complete accepted packet.

The metric and comparison rows supply the hyperbolic geometry that the barycenter read may inspect. The finite weight ledger records the displayed coefficients, while the Busemann-boundary and Real-seal rows expose boundary direction and scalar budget only as named rows.

Transport, replay, provenance, and local naming preserve this inventory. Since no carrier coordinate stores a host minimizer, quotient measure, selected center, countable-choice family, or completed hyperbolic space, those objects are outside the local NameCert surface. □

Theorem 8047.3 (Hyperbolic barycenter boundary handoff)**.** *Every boundary-facing read of an accepted* $\mathsf{HyperbolicBarycenter}^{\uparrow}$ *packet factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{GromovHyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ H, C, P, N.$$

The exported barycenter row is therefore a finite weighted readback over displayed metric, coarse-hyperbolic, boundary, and scalar rows, not a quotient boundary measure or selected global center.

Proof. Start with the NameCert obligations. They require the metric row M, comparison row G, weight ledger W, boundary row U, and scalar seal R before any barycenter read is exported.

The boundary-facing consumer can read U only as part of that same packet. It cannot detach the Busemann row from the metric and hyperbolicity rows, because replay through C preserves the carried order.

Thus the handoff remains finite and row-scoped. The structural rows H, P, N transport, source, and name the same route, leaving no surface for a quotient measure, selected center, or ambient completion. □

Theorem 8047.4 (Hyperbolic barycenter non-escape)**.** *No accepted* $\mathsf{HyperbolicBarycenter}^{\uparrow}$ *consumer reaches a barycenter read without first displaying the hyperbolic metric row, Gromov comparison row, finite weight ledger, Busemann-boundary row, Real scalar seal, componentwise transport, replay, provenance, and local naming rows.*

Proof. Use the carrier inventory of Definition 8047.1. A barycenter read is exposed only through $M, G, W, U, R, T, H, C, P, N$.

Apply Definition 8047.3. It keeps boundary-facing and scalar-facing reads attached to the same finite weighted route. Omitting any listed row changes the carrier rather than preserving the accepted packet.

Therefore a claimed barycenter read that bypasses metric, comparison, weight, boundary, scalar, transport, replay, provenance, or local naming data has no local NameCert row and is rejected. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBarycenter}^{\uparrow}$ packet exposes HyperbolicMetric, GromovHyperbolicMetric, finite weight-ledger, HyperbolicBusemannBoundary, Real-seal, hsame, Cont, Pkg, and NameCert rows before any barycenter read is exported.

Independence witness: hyperbolic-metric, gromov-hyperbolic-metric, hyperbolic-busemann-boundary, real: none is bijective with $\mathsf{HyperbolicBarycenter}^{\uparrow}$ because the barycenter packet jointly binds metric reads, coarse-hyperbolic comparison, finite weights, boundary direction, scalar seal, transport, replay, provenance, and local naming as one finite weighted readback route.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBarycenter}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBarycenter}^{\uparrow}$ packet is generated from HyperbolicMetric rows, GromovHyperbolic-Metric comparison rows, finite weight ledgers, HyperbolicBusemannBoundary rows, a RealUp scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBarycenter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8047.1, Definition 8047.2, Definition 8047.3, and Definition 8047.4 seed the finite weighted hyperbolic barycenter route over metric, coarse-hyperbolic, boundary, scalar, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBarycenter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed hyperbolic space, quotient boundary measure, selected global center, countable-choice family, ambient convexity theorem, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove metric exposure, finite weight-ledger admission, Busemann-boundary handoff, scalar-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBarycenterUp.

Chapter 8048

A Concrete Naming Certificate for $\mathsf{LocatedClosedSetDistance}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis, the $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ packet records the finite data by which a distance-to-a-closed-set read is consumed without a host subset equality, a selected nearest point, countable choice, or ambient completeness. It sits over the metric source of chapter 57, the stream-name and regular rational readback rows of chapter 126 and chapter 301, the scalar real seal of chapter 38, and the located-real comparison surface of **??**.

Definition 8048.1 (Located closed-set distance carrier)**.** *A* $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M, S, Q, L, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the ambient metric comparison used by the distance read.*
2. *S is the closed-set name row, exposed only through finite negative and positive location tests.*
3. *Q is the query-point row naming the inspected point of the metric source.*
4. *L is the located-window row recording finite lower and upper distance cuts for Q against S.*
5. *D is the dyadic tolerance row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *schedules.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *scalar seal for the resulting distance name.*
7. *W stores the finite witness ledger tying each accepted cut to a displayed metric comparison.*
8. *H records componentwise* hsame *transport for the metric, set-name, query, located-window, tolerance, scalar, witness, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a distance query through the visible rows.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedClosedSetDistance}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedClosedSetDistance}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host closed subset, equality of subsets, a chosen nearest point, quotient equality of metric spaces, countable choice, or ambient completeness.

Theorem 8048.2 (Located closed-set distance NameCert obligations)**.** *For every accepted* LocatedClosedSetDistanc *carrier* $Z = (M, S, Q, L, D, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedClosedSetDistance}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure through* M*, closed-set name exposure through* S*, query-point exposure through* Q*, located-window admission through* L*, dyadic tolerance scheduling through* D*, scalar sealing through* R*, witness-ledger exactness through* W*, componentwise transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8048.1. Its analytic rows are exactly M, S, Q, L, D, R, W, followed by structural transport, replay, provenance, and naming rows.

The located-window row L may be read only with the query row Q, the closed-set name row S, and the metric source M still visible. The tolerance row D supplies finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ schedules, while R seals the result at the $\mathsf{Real}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ boundary.

Finally W, H, C, P, N record witness exactness, transport, replay, provenance, and local naming for the same displayed rows. Since no carrier coordinate stores a host subset, selected nearest point, quotient metric equality, choice principle, or completeness theorem, these rows exhaust the local NameCert obligations. □

Theorem 8048.3 (Located closed-set distance metric handoff)**.** *Every metric consumer of an accepted* $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ *packet must read the finite route*

$$M, S, Q, L, D, R, W \longmapsto H, C, P, N$$

before using the distance name. The handoff supplies a metric-source row, a closed-set name row, a query-point row, located lower and upper cuts, dyadic tolerance schedules, a real located scalar seal, finite witness exactness, transport, replay, provenance, and local naming; it does not supply a host subset equality, a selected nearest point, an ambient completion theorem, or countable choice.

Proof. Use Definition 8048.2. It forces any accepted distance read to expose M, S, Q, L, D, R, W before the structural rows may transport, replay, source, or name the result.

Now pass to a metric consumer. The consumer receives the distance as a located scalar row sealed by R, with its dyadic tolerance schedule D and finite metric witness ledger W still attached to the source metric M.

The remaining rows H, C, P, N preserve and name that route only. Because the carrier contains no coordinate for a host subset equality, selected minimizer, ambient completion, or choice principle, the metric handoff is exactly the displayed finite packet. □

Falsifiable prediction: An accepted $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ read is refuted if it preserves the scalar distance row R while removing the metric source M, closed-set name S, query row Q, located-window row L, or witness ledger W.

Independence witness: metric, streamname, regseqrat, real, locatedreal: none is bijective with $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ because this packet jointly binds a metric source, closed-set name, query row, located distance cuts, dyadic tolerance schedules, scalar sealing, finite witness exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedClosedSetDistance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedClosedSetDistance}^{\uparrow}$ packet is generated from a metric source row, a closed-set name row, a query-point row, located lower and upper distance cuts, dyadic tolerance schedules, real and located-real scalar seals, finite metric witness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedClosedSetDistance}\uparrow}$, seed)

Scope closed: Definition 8048.1, Definition 8048.2, and Definition 8048.3 seed the finite constructive distance-to-closed-set certificate surface over metric, stream-name, regular-rational, real, and located-real rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedClosedSetDistance}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.LocatedClosedSetDistanceUp`

Bridge status: none

Not claimed: This chapter does not close host closed-set equality, selected nearest points, ambient completeness, countable choice, quotient metric equality, full Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window exactness, tolerance scheduling, scalar sealing, witness-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedClosedSetDistanceUp.

Lean4 (statement) `BEDC.Derived.LocatedClosedSetDistanceUp`

Chapter 8049

A Concrete Naming Certificate for $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$

Following Bishop–Bridges constructive compactness and Kelley-style finite-net compactness, the $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$ packet records a finite chain of epsilon-net rows at displayed tolerances before a compactness or completion consumer reads a terminal real-name boundary. It sits after the compact-metric source of chapter 129, the total-bounded finite-net surface of chapter 127, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet packages the compactness and completion finite route as displayed NameCert rows, not as an open-cover compactness proof, a host finite subset, a selected Cauchy limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8049.1 (Finite epsilon-net chain carrier)**.** *A* $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, T, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *before any finite-net chain is inspected.*
2. *K is the total-bounded finite-net family. For each displayed tolerance cell it exposes the corresponding* $\mathsf{TotallyBounded}^{\uparrow}$ *center bundle and coverage ledger.*
3. *T is the finite tolerance chain, ordered by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison rows that relate successive epsilon levels.*
4. *D records the dyadic slack and mesh-budget rows used when the chain refines one finite net into the next.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row consumed by completion-facing readers before any regular rational readback is available.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows and dyadic tolerances.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after X, K, T, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for compact-metric, total-bounded, dyadic, stream, regular-rational, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite-chain request through* $X, K, T, D, S, R, L, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetChain}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, K, T, D, S, R, L, H, C, P, N$. It has no coordinate for an ambient open cover, a host enumeration of centers, a selected completion point, quotient equality of streams, countable choice, or ambient real completeness.

Theorem 8049.2 (Finite epsilon-net chain NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$ *carrier $E = (X, K, T, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetChain}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, the total-bounded finite-net family, the dyadic tolerance chain, mesh-budget rows, finite StreamName windows, RegSeqRat readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. Every compactness or completion-facing obligation must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{TotallyBounded}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},$$

No obligation is stored outside those rows.

Proof. Project the carrier of Definition 8049.1. Its compactness source is X, its finite-net data are K, and the only refinement data are the dyadic tolerance chain T and mesh-budget rows D.

Read the completion-facing boundary next. The finite StreamName windows S and RegSeqRat readback R are displayed before the terminal Real seal L can be consumed, so the Real handoff remains downstream of finite observations.

The structural rows H, C, P, N transport, replay, source, and name only the same packet. Since open covers, host center enumerations, selected completion points, quotient stream equality, countable choice, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8049.3 (Finite epsilon-net chain Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$ *carrier must consume the compact-metric row X, the total-bounded finite-net family K, the dyadic tolerance chain T, the mesh-budget row D, the finite StreamName windows S, and the RegSeqRat readback R before it may inspect the terminal seal L. The handoff is the finite route*

$$X, K, T, D \longmapsto S, R \longmapsto L \longmapsto H, C, P, N,$$

not an open-cover compactness argument, not a selected Cauchy limit, not quotient stream equality, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real or completion-facing consumer of E. By Definition 8049.1, the consumer enters through the compact metric source and the displayed finite-net chain, so the compactness data available to it are X, K, T, D.

The readback rows are then forced by the same carrier. The consumer must expose S and R before L is available, and those rows are tied to the displayed dyadic tolerances rather than to a detached stream representative.

Finally apply the NameCert obligations of Definition 8049.2. Transport, replay, provenance, and local naming preserve the same finite route, while the missing classical and ambient principles have no carrier coordinate from which they could be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetChain}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteEpsilonNetChain}^{\uparrow}$ packet is generated from a CompactMetric source row, a Totally-Bounded finite-net family, a DyadicRatCore tolerance chain, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8049.1, Definition 8049.2, and Definition 8049.3 seed the finite compactness and completion route over compact-metric, total-bounded, dyadic, stream, regular-rational, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Cauchy limits, quotient stream equality, countable choice, ambient RealUp completeness, Lean verification, axiom-purity audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-chain exactness, finite-net refinement, StreamName window consumption, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteEpsilonNetChainUp.

Chapter 8050

A Concrete Naming Certificate for $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$

Following Mac Lane on idempotent splittings and Eilenberg–Moore completion algebras, the $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ packet records the finite categorical surface where a completion idempotent is consumed by an explicit splitting row. It sits beside the category surface of chapter 61, the monad surface of chapter 108, the completion category of chapter 7554, the Eilenberg–Moore completion algebra of chapter 2139, and the regular rational and real completion route through chapter 301 and chapter 38. The packet names a splitting interface for L10 completion consumers; it is not a quotient completion, not a selected limit, and not host equality of completed objects.

Definition 8050.1 (Idempotent completion splitting carrier)**.** *An* $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, E, S, R, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the object row, naming the completion object read through* $\mathsf{CauchyCompletionCategory}^{\uparrow}$ *and the underlying* $\mathsf{Cat}^{\uparrow}$ *surface.*
2. *E is the idempotent row, naming the displayed endomorphism on O and its finite idempotence witness.*
3. *S is the splitting row, carrying the displayed retract object together with section and retraction arrows inside the completion category.*
4. *R is the retract law row, exposing the two composite equalities only as finite category-law and completion-algebra comparisons.*
5. *T is the terminal completion-readback row through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *when the splitting is consumed by the L10 route.*
6. *H records componentwise* hsame *transport for object, idempotent, splitting, retract-law, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route from object and idempotent rows to the splitting row and then to the completion readback row.*

8. *P is the* Pkg *provenance over* $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$, $\mathsf{CauchyCompletionCategory}^{\uparrow}$, EilenbergMooreComp
   $\mathsf{Cat}^{\uparrow}$, $\mathsf{Monad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{IdempotentCompletionSplitting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, E, S, R, T, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, ambient split-idempotent completion, countable choice, host equality, or Real completeness.

Theorem 8050.2 (Idempotent completion splitting NameCert obligations)**.** *For every accepted* $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ *carrier $K = (O, E, S, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IdempotentCompletionSplitting}^{\uparrow}}$ *surface consists exactly of object exposure, idempotent exposure, splitting-row exposure, retract-law exposure, completion-readback exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from quotient completion equality, selected representatives, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 8050.1. The only displayed rows are $O, E, S, R, T, H, C, P, N$, so each local obligation must be read from one of those rows.

The categorical obligations are exhausted by O, E, S, R. The object and idempotent rows name the endomorphism being split, the splitting row names the retract data, and the retract-law row names the two finite composites inside the completion category and Eilenberg–Moore algebra surface.

The analytic completion readback is downstream of the splitting data. Row T may expose the regular rational, dyadic, and Real-facing route, but only after the categorical splitting rows have already been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no quotient, selection, countable-choice, host-equality, or completeness coordinate, the non-escape obligation follows by exhaustion. □

Theorem 8050.3 (Idempotent completion splitting nonescape)**.** *Every* $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ *consumer must expose $O, E, S, R, T, H, C, P, N$ before reading a completed object through a split idempotent. The packet cannot export a quotient completion, a selected splitting, ambient Karoubi completion, host equality of completed objects, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer that reads a completed object through an idempotent. The carrier first exposes O and E, so the read is attached to the displayed object and displayed endomorphism.

The consumer then reaches the splitting only through S and the retract laws only through R. These rows live inside the finite completion-category and completion-algebra surfaces, so they do not import an ambient category of all split idempotents.

The terminal readback row T is reached after the splitting rows. It may pass through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, but it does not add quotient completion equality or a selected representative.

The structural rows H, C, P, N preserve the same finite route. Therefore a consumer receives the displayed splitting packet and no stronger quotient, selection, choice, trichotomy, host-equality, or completeness principle. □

Falsifiable prediction: A purported $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ consumer is refuted if it reads a split completion object without displaying the object, idempotent, splitting, retract-law, completion-readback, transport, replay, provenance, and local NameCert rows.

Independence witness: cauchy-completion-category, eilenberg-moore-completion-algebra, category, monad, regseqrat, real: none is bijective to $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ because this

packet jointly carries object, idempotent, splitting, retract-law, completion-readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{IdempotentCompletionSplitting}^{\uparrow}}$.

Constructive story: An $\mathsf{IdempotentCompletionSplitting}^{\uparrow}$ packet is generated from completion-object rows, idempotent rows, splitting rows, retract-law rows, RegSeqRat/DyadicRatCore/Real readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IdempotentCompletionSplitting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8050.1, Definition 8050.2, and Definition 8050.3 seed the finite idempotent-splitting packet for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IdempotentCompletionSplitting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, ambient Karoubi completion, countable choice, trichotomy, host equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, idempotent exposure, splitting-row exposure, retract-law exactness, completion-readback ordering, transport, replay, provenance exhaustion, and local NameCert non-escape.

Chapter 8051

A Concrete Naming Certificate for MetaCICResidualConfluenceBudget$^{\uparrow}$

The MetaCICResidualConfluenceBudget$^{\uparrow}$ packet realizes the MetaCIC residual-substitution confluence frontier as a finite budget surface. It sits beside the residual-fuel packet of chapter 7966, the local diamond witness of chapter 8046, the commutation witness of chapter 7977, the residual-substitution boundary of chapter 2181, and the parallel-diamond frontier of chapter 4238. The carrier records the rows a bounded confluence checker must display before it may retain or discharge a local residual peak; it is not a global closed Church–Rosser theorem.

Definition 8051.1 (MetaCIC residual confluence budget carrier)**.** *A* MetaCICResidualConfluenceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, R, J, Q, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row naming the closed source term, typing context, and bounded syntax window under inspection.*
2. *R is the residual-pair row naming the two residual branches and the substitution boundary retained with them.*
3. *J is the local-join row, available only after the typed window and residual-pair rows have been displayed.*
4. *Q is the bounded-checker row, recording the finite checker budget for residual replay, substitution replay, and local-join inspection.*
5. *O is the obstruction-retention row, preserving the refused endpoint when the displayed rows do not close a local join.*
6. *H records componentwise* hsame *transport for typed-window, residual-pair, local-join, bounded-checker, obstruction, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through the same residual-confluence route.*
8. *P is the* Pkg *provenance over* MetaCICResidualConfluenceBudget$^{\uparrow}$*,* MetaCICResidualConfluenceFuel$^{\uparrow}$*,* MetaCICResidualLocalDiamondWitness$^{\uparrow}$*,* MetaCICResidualCommutationWitness$^{\uparrow}$*,* MetaCICResidualSubstitutionBo MetaCICParallelDiamondFrontier$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $T, R, J, Q, O, H, C, P, N$. It has no coordinate for arbitrary confluence, subject reduction, evaluator equality, quotient syntax equality, selected normal forms, kernel consistency, or host-level proof search.

Theorem 8051.2 (MetaCIC residual confluence budget obligations)**.** *For every accepted* MetaCICResidualConfluenc *carrier $B = (T, R, J, Q, O, H, C, P, N)$, a confluence-facing consumer must expose the typed window T, residual-pair row R, local-join row J, bounded-checker row Q, obstruction-retention row O, componentwise transport H, replay row C, provenance P, and local naming row N before reading any residual-confluence conclusion.*

Proof. Project the carrier in Definition 8051.1. The rows $T, R, J, Q, O, H, C, P, N$ are the only displayed fields of the budget packet.

The confluence-facing rows are ordered. A local join can be read only after the typed window and residual pair have been displayed, and the bounded checker can only inspect that same finite route.

The obstruction row remains visible when the checker does not close the join. The structural rows then transport, replay, source, and name the same packet, with no extra endpoint for arbitrary confluence or kernel consistency. □

Theorem 8051.3 (MetaCIC residual confluence budget nonescape)**.** *An accepted* MetaCICResidualConfluenceBudge *packet cannot export closed Church–Rosser, full parallel diamond, subject reduction, evaluator equality, quotient equality of terms, selected normal forms, or MetaCIC kernel consistency. Any such endpoint must come from a separate downstream discharge that consumes the displayed residual-pair and bounded-checker rows.*

Proof. Use the obligation theorem. Every confluence-facing read is confined to the typed window, residual-pair, local-join, bounded-checker, obstruction, transport, replay, provenance, and naming rows.

Those rows are budget rows rather than global theorem rows. They can retain a local obstruction or certify a displayed bounded check, but they do not contain a subject-reduction discharge, quotient syntax equality, evaluator equality, normal-form selector, or kernel-consistency witness.

Because H, C, P, N preserve the same finite rows componentwise, they cannot manufacture any of the refused endpoints. The packet therefore remains a residual-confluence budget and not a closed confluence theorem. □

The Lean TasteGate follow-up path for $\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$ instances. These instances must keep $T, R, J, Q, O, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve downstream confluence consumers.

Falsifiable prediction: If a residual-confluence checker reaches a local-join conclusion without exposing the typed window, residual pair, bounded checker, obstruction-retention, transport, replay, provenance, and local NameCert rows, then it is not consuming $\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$.

Independence witness: metacic-residual-confluence-fuel, metacic-residual-local-diamond-witness, metacic-residual-commutation-witness, metacic-residual-substitution-boundary, metacic-parallel-diamond-frontier: none is bijective to $\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$ because this packet jointly carries typed-window, residual-pair, local-join, bounded-checker, obstruction-retention, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}$ packet is generated from a typed-window row, residual-pair row, local-join row, bounded-checker row, obstruction-retention row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8051.1, Definition 8051.2, and Definition 8051.3 seed the finite residual-confluence budget surface for the MetaCIC confluence frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualConfluenceBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary confluence, closed Church–Rosser, full parallel diamond, subject reduction, evaluator equality, quotient syntax equality, selected normal forms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-pair exactness, local-join ordering, bounded-checker replay, obstruction retention, TasteGate nontriviality, and field faithfulness.

Chapter 8052

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailSeal}^{\uparrow}$

Following Bishop–Bridges regular-Cauchy completion, the $\mathsf{RegularCauchyTailSeal}^{\uparrow}$ packet names the finite tail-seal surface that a completion consumer must expose before a displayed $\mathsf{Real}^{\uparrow}$ endpoint can be read. It sits between the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet also factors through the sequential-completeness route of chapter 1794; it is a reusable finite tail object, not a quotient stream, selected host limit, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 8052.1 (RegularCauchy tail-seal carrier)**.** *A* $\mathsf{RegularCauchyTailSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, M, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic endpoints and comparison budget used by the tail request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed by the request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S and checked against the tolerances in D.*
4. *M is the tail threshold and modulus row assigning each displayed tolerance a finite tail bound for R.*
5. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal reached after D, S, R, M have been displayed.*
6. *H records componentwise* hsame *transport for the tolerance ledger, stream window, readback, modulus, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail-seal request through D, S, R, M, T.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailSeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, M, T, H, C, P, N$. There is no coordinate for quotient stream equality, host real equality, a selected limit point, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyTailSealUp`

Theorem 8052.2 (RegularCauchy tail-seal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailSeal}^{\uparrow}$ *carrier* $Q = (D, S, R, M, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSeal}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance ledger, stream window, regular rational readback, threshold/modulus row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *route exposure, requiring a consumer of* T *to expose* D, S, R, M *before the terminal tail seal is read;*
3. *classifier stability, requiring componentwise* hsame *transport in* H *to preserve the rows* D, S, R, M, T *before accepted packets are identified;*
4. *replay stability, requiring the displayed* Cont *row* C *to replay the same tolerance-window-readback-threshold route;*
5. *provenance and local naming, requiring* P *and* N *to name only the displayed finite packet and its sibling dependencies;*
6. *ledger non-escape, requiring the packet to export no quotient stream equality, selected host limit, countable-choice schedule, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

No NameCert obligation is stored outside $D, S, R, M, T, H, C, P, N$*.*

Proof. Carrier admission is projection from Definition 8052.1. The accepted packet displays exactly the tolerance ledger D, window row S, readback row R, threshold/modulus row M, terminal seal T, and structural rows H, C, P, N.

Route exposure is forced by the order of the displayed rows. A consumer must first read the dyadic tolerance in D, then the finite window in S, then the regular rational readback in R, and then the threshold/modulus row M before T is available as the $\mathsf{Real}^{\uparrow}$ tail seal.

Classifier stability and replay stability are componentwise. The transport row H preserves the visible rows, while C replays the same route without adding a second tail source or a hidden limit coordinate.

Finally P and N record provenance and local naming for the finite packet. Since none of the excluded objects occurs among the carrier coordinates, the ledger cannot export quotient streams, selected limits, choice schedules, host equality, ambient completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSeal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailSeal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName window, RegSeqRat readback, threshold/modulus row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSeal}^{\uparrow}}$, seed)

Scope closed: Definition 8052.1 and Definition 8052.2 seed the finite tail-seal surface between regular-Cauchy readback data and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSeal}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.RegularCauchyTailSealUp`

Bridge status: none

Not claimed: This chapter excludes quotient stream equality, selected host limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, route exposure, replay stability, and ledger non-escape for BEDC.Derived.RegularCauchyTailSealUp.

Chapter 8053

A Concrete Naming Certificate for $\mathsf{LawlikeCauchyModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Brouwer–Troelstra on lawlike sequences, the $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ packet records the finite route by which a lawlike located real sequence exposes an explicit Cauchy modulus before any regular-rational or real-name consumer reads it. It sits after the lawlike located-sequence surface of chapter 8042, the Bishop choice-sequence boundary of $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the located comparison surface of chapter 1933. The packet supplies the L10 Real and Completion route with displayed lawlike Cauchy-modulus data; it refuses quotient streams, selected limits, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8053.1 (Lawlike Cauchy modulus carrier)**.** *A* $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, D, M, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the lawlike source row, recording the finite rule ledger and admissible request boundary for the indexed sequence.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing the inspected indices, source windows, and modulus-query windows available to the packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row, carrying the requested precision, dyadic slack, and comparison budget used by the modulus ledger.*
4. *M is the explicit Cauchy modulus ledger, displaying the threshold assigned to each admitted dyadic tolerance request and the finite later-window checks attached to it.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after S, W, D, M have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular readback row R.*
7. *Q is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row attached to the same readback and terminal seal.*
8. *H records componentwise* hsame *transport for the lawlike source, stream windows, dyadic tolerances, modulus ledger, regular readback, Real seal, located comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from lawlike source inspection through finite windows, tolerance budgets, modulus thresholds, regular readback, Real sealing, and located comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LawlikeCauchyModulus}^{\uparrow}$, $\mathsf{LawlikeLocatedRealSequence}^{\uparrow}$, BishopCauchyChoiceSequen StreamName$^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LawlikeCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, M, R, E, Q, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, selected representatives, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8053.2 (Lawlike Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ *carrier*

$$L = (S, W, D, M, R, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LawlikeCauchyModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lawlike source, finite StreamName windows, dyadic tolerance budget, explicit Cauchy modulus ledger, RegSeqRat readback, Real seal, LocatedReal comparison, transport, replay, provenance, and local naming rows;*

2. *source-window exactness, requiring every modulus request in M to read the lawlike source row S and the finite window row W before a threshold is exported;*

3. *tolerance-budget exactness, requiring each threshold in M to be bound to the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *request and slack data in D;*

4. *readback handoff, requiring* $\mathsf{RegSeqRat}^{\uparrow}$ *row R,* $\mathsf{Real}^{\uparrow}$ *seal E, and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row Q to consume the same S, W, D, M route;*

5. *transport and replay stability through H and C for the same displayed modulus route;*

6. *provenance and local naming exhaustion through P and N;*

7. *refusal of quotient streams, selected limits, countable choice, trichotomy, selected representatives, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local lawlike Cauchy-modulus obligation is stored outside these rows.

Proof. Project the accepted carrier. The rows S, W, D, M are the only analytic source for a modulus request: S records the lawlike rule ledger, W records inspected finite windows, D records the dyadic tolerance budget, and M records the displayed thresholds.

The threshold row is therefore local to the packet. A consumer cannot read a threshold from M before the source, window, and tolerance rows have been exposed, and it cannot replace that threshold by a host selector or a countable-choice schedule.

The readback rows are downstream of the same route. The regular row R repacks the displayed windows and modulus ledger as regular rational data, while E and Q seal only that readback as real and located-real information.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since none of the carrier coordinates stores quotient stream equality, a selected limit, trichotomy, selected representatives, host real equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8053.3 (Lawlike Cauchy modulus RegSeqRat handoff)**.** *For every accepted* $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ *carrier $L = (S, W, D, M, R, E, Q, H, C, P, N)$, a* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{LocatedReal}^{\uparrow}$ *consumer may*

read the regular sequence data only after the lawlike source row S, the finite $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the explicit modulus ledger M have all been displayed. Thus the handoff to R, E, Q is a non-quotient finite-window route and not an appeal to selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier. The regular readback row R is not a free-standing real-name row; by Definition 8053.1 it is reached only from the displayed source, window, tolerance, and modulus rows.

Read S, W, D, M in order. The lawlike source determines which requests are admitted, the window row lists the finite observations, the dyadic row fixes the tolerance budget, and the modulus ledger displays the threshold used for later-window comparison.

Only after those rows are visible may R be consumed. The terminal seal E and located comparison Q are downstream of R, so they inherit the same finite-window route.

The structural rows H, C, P, N do not add a quotient stream, a selected host limit, a choice schedule, a trichotomy decision, or a completeness witness. They identify the same displayed handoff for transport, replay, provenance, and local naming. □

Falsifiable prediction: A $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ read is refuted whenever a $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{LocatedReal}^{\uparrow}$ consumer reaches regular data, a real seal, or located comparison without displaying the lawlike source row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, explicit modulus ledger, structural replay, provenance, and local $\mathsf{NameCert}$ evidence.

Independence witness: lawlike-located-real-sequence, bishop-cauchy-choice-sequence, stream-name, regseqrat, dyadicratcore, real, located-real: $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a lawlike source, finite stream windows, dyadic tolerance budgets, explicit Cauchy-modulus thresholds, regular readback, Real seal, located comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LawlikeCauchyModulus}^{\uparrow}$ packet is generated from a lawlike source row, finite StreamName windows, DyadicRatCore tolerance budgets, an explicit Cauchy modulus ledger, RegSeqRat readback, Real seals, LocatedReal comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawlikeCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8053.1, Definition 8053.2, and Definition 8053.3 seed the finite lawlike Cauchy-modulus route over StreamName windows, DyadicRatCore tolerance budgets, explicit modulus thresholds, RegSeqRat readback, Real seals, LocatedReal comparison rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawlikeCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, trichotomy, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, tolerance-budget exactness, explicit modulus ledger coverage, RegSeqRat handoff, LocatedReal comparison stability, transport replay, provenance, and nonescape for BEDC.Derived.LawlikeCauchyModulusUp.

Chapter 8054

A Concrete Naming Certificate for $\mathsf{InfimalConvolution}^{\uparrow}$

Following Rockafellar's convex analysis and the Boyd–Vandenberghe finite convex-optimization presentation, the $\mathsf{InfimalConvolution}^{\uparrow}$ packet records the binary lower-envelope operation by displayed epigraph, value-window, and Fenchel handoff rows. It separates the lower-envelope certificate consumed by the Moreau envelope chapter chapter 7622 from the proximal smoothing packet itself, while staying anchored in the convex source of chapter 208, the convex-conjugate boundary of chapter 3563, the Fenchel duality ledger of chapter 321, the Fenchel–Young inequality surface of chapter 7804, and the terminal real-name seal of chapter 38. The carrier names finite candidate decompositions and lower-bound comparisons; it is not a host infimum operator, selected minimizer, quotient function space, or ambient lower-semicontinuity theorem.

Definition 8054.1 (Infimal convolution carrier)**.** *An* $\mathsf{InfimalConvolution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, G, X, V, L, D, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *F and G are the two source function rows, each read through finite domain, value, and convexity observations.*
2. *X is the split-variable window, listing the displayed decompositions $x = u + v$ that are available to the lower-envelope read.*
3. *V is the value-sum ledger, recording the visible rows $F(u)$, $G(v)$, and $F(u) + G(v)$ for each displayed split.*
4. *L is the lower-bound ledger, recording finite comparison rows that any proposed lower envelope must satisfy against the displayed value sums.*
5. *D is the* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *handoff row, exposing the conjugate and dual comparison data visible at the same split window.*
6. *Y is the* $\mathsf{FenchelYoungInequality}^{\uparrow}$ *row, recording the local inequality witnesses used when a downstream Fenchel consumer reads the lower-envelope packet.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for value readback, reachable only after F, G, X, V, L, D, Y have been displayed.*

8. *H records componentwise* hsame *transport for the source functions, split window, value ledger, lower-bound ledger, Fenchel handoff, inequality row, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from source-function admission through split inspection, value-sum comparison, lower-bound readback, Fenchel handoff, and real-seal export.*
10. *P is the* Pkg *provenance over* $\mathsf{InfimalConvolution}^{\uparrow}$, $\mathsf{MoreauEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{FenchelYoungInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{InfimalConvolution}^{\uparrow}}$ *row.*

The accepted carrier compares lower-envelope reads only through these rows. It has no coordinate for an arbitrary host infimum, a selected minimizing split, quotient equality of functions, global lower-semicontinuous hulls, or ambient convex completeness.

Theorem 8054.2 (Infimal convolution NameCert obligations)**.** *For the carrier of Definition 8054.1, the local* $\mathsf{NameCert}_{\mathsf{InfimalConvolution}^{\uparrow}}$ *surface consists exactly of carrier admission for $I = (F, G, X, V, L, D, Y, E, H, C,$ source-function exposure, split-window exposure, value-sum ledger exactness, lower-bound comparison, Fenchel handoff visibility, Fenchel–Young inequality replay,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed continuation replay, package provenance, and local naming. Every accepted lower-envelope read factors through*

$$F, G \longmapsto X \longmapsto V \longmapsto L \longmapsto E$$

with D, Y, H, C, P, N attached as Fenchel, inequality, transport, replay, provenance, and naming rows.

Proof. Project the accepted carrier. The source-function rows are F and G, and the only displayed decompositions are those listed in the split window X.

The value-sum row V is therefore the first lower-envelope ledger a consumer can inspect. The lower-bound row L compares proposed envelope data only against the sums already displayed by V, and the real seal E is downstream of those same rows.

The Fenchel-facing rows do not add an alternate envelope. The handoff D and inequality row Y replay conjugate and Fenchel–Young evidence at the same split window, while H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores a host infimum, selected minimizing split, quotient function equality, global lower-semicontinuous hull, or ambient convex-completeness object, the listed rows exhaust the local NameCert obligations. □

Theorem 8054.3 (Infimal convolution Fenchel handoff)**.** *Every* $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{FenchelYoungInequality}^{\uparrow}$, *or* $\mathsf{MoreauEnvelope}^{\uparrow}$ *consumer of an accepted* $\mathsf{InfimalConvolution}^{\uparrow}$ *carrier must first read the source rows F, G, the split-variable window X, the value-sum ledger V, and the lower-bound ledger L before it consumes the Fenchel handoff row D, inequality row Y, or terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The handoff is a finite lower-envelope route, not a host optimization theorem or selected minimizing decomposition.*

Proof. Start with the local obligations of Definition 8054.2. They force every lower-envelope read through the displayed source rows, split window, value-sum ledger, and lower-bound ledger.

Now read the Fenchel-facing side. The row D exposes only conjugate and dual comparison data attached to the same split window, and the row Y exposes only the corresponding Fenchel–Young witnesses. A $\mathsf{MoreauEnvelope}^{\uparrow}$ consumer can cite this packet only after that lower-envelope route has been displayed, so smoothing data cannot replace the binary lower-envelope certificate.

Finally pass to the real seal. The seal E is reached after F, G, X, V, L, D, Y, and the structural rows H, C, P, N preserve that route under transport and replay. Hence the downstream Fenchel and Moreau consumers receive exactly the finite handoff displayed by the carrier and no host optimization, selected minimizer, quotient function equality, or ambient convex-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{InfimalConvolution}^{\uparrow}}$.

Constructive story: An $\mathsf{InfimalConvolution}^{\uparrow}$ packet is generated from two source function rows, a split-variable window, a value-sum ledger, a lower-bound ledger, ConvexConjugate and FenchelDuality handoff rows, a FenchelYoungInequality row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InfimalConvolution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8054.1, Definition 8054.2, and Definition 8054.3 seed the finite binary lower-envelope certificate consumed by MoreauEnvelopeUp, ConvexConjugateUp, FenchelDualityUp, FenchelYoungInequalityUp, and RealUp routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InfimalConvolution}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, selected minimizers, quotient function equality, global lower-semicontinuous hulls, host infimum operators, or ambient convex completeness.

Upgrade path: Formal verification can encode carrier admission, split-window exposure, value-ledger exactness, lower-bound comparison, Fenchel handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.InfimalConvolutionUp.

Chapter 8055

A Concrete Naming Certificate for $\mathsf{MetricEntropy}^{\uparrow}$

Following Kolmogorov–Tikhomirov metric entropy and Bishop-style finite nets, the $\mathsf{MetricEntropy}^{\uparrow}$ packet records the finite covering-number ledger attached to an inspected metric or compact metric carrier. It sits after the metric source of chapter 57, the compact metric source of chapter 129, the totally bounded finite-net route of chapter 127, the finite net surface of chapter 1767, the rational tolerance boundary of chapter 37, and the terminal real-name readback of chapter 38. The packet is a finite cover-count carrier for later compactness, dimension, and entropy estimates; it does not import open-cover compactness, selected subcovers, quotient real equality, or ambient completeness.

Definition 8055.1 (Metric entropy carrier)**.** *A* $\mathsf{MetricEntropy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, \varepsilon, N, C, M, H, K, P, L)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, or the* $\mathsf{CompactMetric}^{\uparrow}$ *source row when the entropy ledger is read over a compact carrier.*
2. *ε is a positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row, recording the scale at which finite covering data is inspected.*
3. *N is the finite* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, displaying the centers and coverage witnesses used at scale ε.*
4. *C is the cover-count ledger, recording the finite number of displayed centers and the associated covering-number bound.*
5. *M is the monotonicity transport row, recording how a cover at a smaller inspected tolerance supplies a cover-count upper bound at a larger inspected tolerance over the same metric source.*
6. *H records componentwise* hsame *transport for the metric source, tolerance, net, cover-count ledger, monotonicity transport, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a metric-entropy request through $X, \varepsilon, N, C, M, H$.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricEntropy}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{MetricEntropy}^{\uparrow}}$ *row naming exactly this finite cover-count ledger.*

The classifier compares accepted packets only through these rows. It has no coordinate for Hausdorff measure, box dimension, selected subcovers, open-cover compactness, quotient real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricEntropyUp.TasteGate.MetricEntropyTasteGate single carrier alignment`

Theorem 8055.2 (Metric entropy NameCert obligations)**.** *For every accepted* $\mathsf{MetricEntropy}^{\uparrow}$ *carrier*

$$E = (X, \varepsilon, N, C, M, H, K, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetricEntropy}^{\uparrow}}$ *surface consists exactly of carrier admission, metric or compact-metric source visibility, positive rational scale visibility, finite epsilon-net coverage, cover-count ledger visibility, monotonicity transport, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every metric-entropy consumer must pass through* $X, \varepsilon, N, C, M, H, K, P, L$ *before it reads a finite cover count.*

Proof. Carrier admission is projection from Definition 8055.1. The rows X, ε, N, C expose the metric source, inspected positive scale, displayed finite net, and finite cover-count ledger.

The row M records the only monotonicity route supplied by the carrier. It transports cover-count bounds between inspected rational scales over the same metric source, but it does not select additional centers or invoke open-cover compactness.

The rows H, K, P, L preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for Hausdorff measure, box dimension, selected subcovers, quotient real equality, or ambient completeness, these rows exhaust the local NameCert obligation surface. □

Theorem 8055.3 (Metric entropy finite-cover monotonicity)**.** *Let*

$$E = (X, \varepsilon, N, C, M, H, K, P, L)$$

be an accepted $\mathsf{MetricEntropy}^{\uparrow}$ *carrier. Any consumer that compares cover counts at two displayed positive rational scales over the same* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *source must read the comparison through the monotonicity row* M *after exposing the source row* X*, the inspected scale row* ε*, the finite net row* N*, and the cover-count ledger* C*. The comparison cannot be derived from an unlisted compactness selector, a quotient real equality rule, or ambient completeness.*

Proof. Read X and ε first. These rows identify the metric source and the positive rational scale at which the finite cover is being counted.

Read N and C next. The finite net row displays the centers and coverage witnesses, while the ledger row records the finite number attached to that displayed cover.

The only carrier row that relates two displayed scales is M. Transport through H and replay through K preserve this same comparison, and P, L source and name it. Therefore every finite-cover monotonicity read factors through the listed rows and cannot use unlisted compactness or real-equality data. □

Falsifiable prediction: An accepted $\mathsf{MetricEntropy}^{\uparrow}$ consumer that reads a cover count without first displaying the metric source, positive rational scale, finite epsilon-net row, cover-count ledger, and monotonicity transport refutes the carrier surface.

Independence witness: metric, compactmetric, totallybounded, finite-epsilon-net, rat, real: no listed sibling is bijective with $\mathsf{MetricEntropy}^{\uparrow}$ because this packet jointly carries metric or compact-metric source data, a positive rational scale, a displayed finite net, a finite cover-count ledger, monotonicity transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricEntropy}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricEntropy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8055.1, Definition 8055.2, and Definition 8055.3 seed the finite cover-count route over metric and compact metric carriers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricEntropy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff measure, box dimension, selected subcovers, open-cover compactness, quotient real equality, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, scale visibility, finite-net coverage, cover-count ledger visibility, monotonicity transport, replay, provenance, and local naming for BEDC.Derived.MetricEntropyUp.

Constructive story: A $\mathsf{MetricEntropy}^{\uparrow}$ packet is generated from MetricUp or CompactMetricUp source data, a positive RatUp scale, FiniteEpsilonNetUp coverage rows, a finite cover-count ledger, monotonicity transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8056

A Concrete Naming Certificate for $\mathsf{MinCostFlow}^{\uparrow}$

Following Ahuja, Magnanti, and Orlin, the $\mathsf{MinCostFlow}^{\uparrow}$ packet records the finite minimum-cost network-flow certificate surface. It sits after the feasible-flow rows of chapter 233, the residual shortest-path and potential discipline of chapter 236, and the primal-dual comparison ledger of chapter 235. The packet names displayed capacity, conservation, cost, residual-potential, dual-certificate, transport, replay, provenance, and local naming rows; it is not a host optimizer, selected optimal flow, unbounded graph search, or ambient linear-programming oracle.

Definition 8056.1 (Minimum-cost flow carrier)**.** *A* $\mathsf{MinCostFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, U, B, K, F, R, D, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{NetworkFlow}^{\uparrow}$ *source row, carrying the directed graph, source and sink endpoints, and finite edge inventory.*
2. *U is the capacity row, recording displayed lower and upper capacity bounds on each inspected edge.*
3. *B is the conservation row, recording the finite supply, demand, and balance equations for the displayed vertices.*
4. *K is the integer or rational cost ledger, attaching a cost row to each displayed edge and naming the total-cost accumulator.*
5. *F is the feasible-flow row, exposing edge-flow values that obey U and B through the* $\mathsf{NetworkFlow}^{\uparrow}$ *route.*
6. *R is the residual shortest-path and potential row, read through* $\mathsf{ShortestPath}^{\uparrow}$ *and recording reduced-cost comparisons on the displayed residual arcs.*
7. *D is the* $\mathsf{LPDuality}^{\uparrow}$ *certificate row, pairing the primal feasible flow with displayed dual potentials, slack rows, and complementary tightness checks.*
8. *H records componentwise* hsame *transport for network source, capacity, conservation, cost, feasible-flow, residual-potential, dual-certificate, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a minimum-cost-flow query through G, U, B, K, F, R, D, H.*

10. *P is the* Pkg *provenance over* $\mathsf{MinCostFlow}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{ShortestPath}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MinCostFlow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected optimal flow, a host simplex method, an unrestricted residual search, quotient equality of flows, or an ambient existence theorem for arbitrary instances.

Theorem 8056.2 (Minimum-cost flow NameCert obligations)**.** *For every accepted* $\mathsf{MinCostFlow}^{\uparrow}$ *carrier* $M = (G, U, B, K, F, R, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MinCostFlow}^{\uparrow}}$ *surface consists exactly of network-flow source admission through* G*, capacity exposure through* U*, conservation exposure through* B*, integer or rational cost-ledger exposure through* K*, feasible-flow exposure through* F*, residual shortest-path and potential exposure through* R*,* $\mathsf{LPDuality}^{\uparrow}$ *certificate exposure through* D*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8056.1. The source, capacity, conservation, cost, feasible-flow, residual-potential, and dual-certificate data are exactly G, U, B, K, F, R, D.

The feasible-flow row is read only after the capacity and conservation rows. Thus the packet cannot detach a flow value from the displayed $\mathsf{NetworkFlow}^{\uparrow}$ source or from the finite balance ledger.

The residual shortest-path and potential row R then records the reduced-cost boundary used by improvement and non-improvement consumers, while D records the corresponding primal-dual certificate through $\mathsf{LPDuality}^{\uparrow}$. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since no carrier coordinate stores a selected optimizer, host residual oracle, quotient flow equality, or ambient existence theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 8056.3 (Minimum-cost flow LP-duality handoff)**.** *Every optimality-facing consumer of an accepted* $\mathsf{MinCostFlow}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{NetworkFlow}^{\uparrow},\ \mathsf{ShortestPath}^{\uparrow},\ \mathsf{LPDuality}^{\uparrow} \longmapsto G, U, B, K, F, R, D, H, C, P, N.$$

The handoff supplies a displayed feasible flow, cost ledger, residual shortest-path or potential row, and primal-dual certificate row. It does not assert existence of an optimal flow beyond the displayed certificate, does not choose an optimizer for every network, and does not export a host linear-programming theorem outside the finite rows.

Proof. Begin with Definition 8056.2. A consumer must expose G, U, B, K, F before any optimality row can be inspected, so feasibility remains attached to the displayed network-flow source.

Next read R through the residual shortest-path surface. This row records the inspected residual arcs and potentials, not an unrestricted shortest-path oracle over an ambient graph.

Finally read the dual-certificate row D. It packages the $\mathsf{LPDuality}^{\uparrow}$ comparison and complementary tightness checks for the same displayed rows G, U, B, K, F, R. Transport, replay, provenance, and local naming through H, C, P, N preserve that route, so the handoff is a finite certificate and not a global optimizer-existence claim. □

Falsifiable prediction: A purported $\mathsf{MinCostFlow}^{\uparrow}$ consumer is refuted if it reads a cost-optimality conclusion while omitting any of the displayed network source, capacity, conservation, cost, feasible-flow, residual-potential, LP-duality, transport, replay, provenance, or local NameCert rows.

Independence witness: networkflow, shortestpath, lpduality: none is bijective to $\mathsf{MinCostFlow}^{\uparrow}$ because this packet jointly carries feasible-flow, cost-ledger, residual-potential, LP-duality, transport, replay, provenance, and local NameCert rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MinCostFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MinCostFlow}^{\uparrow}$ packet is generated from NetworkFlowUp source rows, capacity rows, conservation rows, integer or rational cost ledgers, feasible-flow rows, residual shortest-path and potential rows, LPDualityUp certificate rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinCostFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8056.1, Definition 8056.2, and Definition 8056.3 seed the finite minimum-cost-flow certificate over feasible-flow, cost, residual-potential, dual-certificate, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinCostFlow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close optimal-flow existence beyond the displayed finite certificate, selected optimizers for arbitrary networks, unrestricted residual search, quotient equality of flows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, capacity exposure, conservation exposure, cost-ledger exactness, feasible-flow exposure, residual-potential ordering, LP-duality certificate exposure, transport, replay, provenance, and local naming for BEDC.Derived.MinCostFlowUp.

Chapter 8057

A Concrete Naming Certificate for MetaCICBoundedNormalEqualityBudget$^{\uparrow}$

The MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ packet realizes the bounded normal-equality checker frontier of chapter 9379 as finite MetaCIC NameCert data. It packages the `normalizeBounded` fuel ledger and the `decideClosedBetaConv` comparison row before a consumer may interpret bounded normal-form equality. The packet sits beside the bounded checker surface of chapter 752, the fuel trace of chapter 3889, the neutral-spine route of chapter 3909, the decidable boundary of chapter 749, and the bounded normal-equality frontier of chapter 6947. It is a finite budget object, not a global closed beta-conversion decider, fuel-erased normalizer, subject-reduction theorem, or kernel-consistency principle.

Definition 8057.1 (MetaCIC bounded normal-equality budget carrier)**.** *A* MetaCICBoundedNormalEqualityBudget *carrier is a finite* BHist *packet*

$$B = (T, F, N, E, Q, R, H, C, P, L)$$

with the following displayed rows:

1. *T is the closed typed-term window row naming the two inspected terms, their common type boundary, and the finite context read by the checker.*
2. *F is the fuel ledger for `normalizeBounded`, recording the displayed bound and every consumed reduction step.*
3. *N is the bounded normal-form output row, carrying the two computed normal-form candidates at the displayed fuel.*
4. *E is the endpoint row, recording finished-normal evidence when the bounded comparison is allowed to become a finished-normal statement.*
5. *Q is the `decideClosedBetaConv` answer row for the two bounded normal-form candidates.*
6. *R is the refusal row retained when the endpoint evidence is absent or the consumer asks for fuel-erased conversion decidability.*
7. *H records componentwise* hsame *transport for typed windows, fuel, bounded normal outputs, endpoints, comparison answers, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded normal-equality query through T, F, N, E, Q, R, H.*

9. *P is the* Pkg *provenance over* MetaCICBoundedNormalEqualityBudget$^{\uparrow}$, BoundedNormalEqualityChecker$^{\uparrow}$, MetaCICFuelTrace$^{\uparrow}$, MetaCICDecidableNeutralSpine$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BoundedNormalEqualityFront BHist, hsame, Cont, *and* NameCert.

10. *L is the local* NameCert$_{\text{MetaCICBoundedNormalEqualityBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F, N, E, Q, R, H, C, P, L$. It has no coordinate for unbounded conversion decidability, fuel-erased normalization, subject reduction, confluence, quotient syntax equality, or MetaCIC kernel consistency.

Theorem 8057.2 (MetaCIC bounded normal-equality budget obligations)**.** *For every accepted* MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ *carrier $B = (T, F, N, E, Q, R, H, C, P, L)$, the local* NameCert$_{\text{MetaCICBounded}}$ *surface consists exactly of typed-window exposure through T, fuel-ledger exposure through F, bounded normal-form output exposure through N, endpoint exposure through E, bounded comparison exposure through Q, refusal retention through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8057.1. The computational rows are T, F, N, E, Q, R, and the structural rows are H, C, P, L.

The bounded normalizer is read through F before the normal-output row N is exposed. This keeps each output attached to the displayed fuel rather than to a fuel-erased normalizer.

The comparison answer Q can be consumed as finished normal equality only through the endpoint row E. If that endpoint is absent, the refusal row R remains part of the exported packet. Transport, replay, provenance, and local naming then preserve exactly the displayed rows. □

Theorem 8057.3 (MetaCIC bounded normal-equality nonescape)**.** *An accepted* MetaCICBoundedNormalEqualityBu *packet cannot export global closed beta-conversion decidability, unrestricted normalization, subject reduction, confluence, quotient syntax equality, or MetaCIC kernel consistency. Any consumer of the checker result must keep the typed window, fuel ledger, bounded normal-form outputs, endpoint row, comparison answer, refusal row, transport, replay, provenance, and local NameCert rows visible.*

Proof. Use Definition 8057.2. Every checker read is confined to $T, F, N, E, Q, R, H, C, P, L$.

The rows F, N, Q are bounded computation rows. The endpoint row E may license a finished-normal statement for the displayed terms, while R records the refused endpoints when that evidence is missing.

None of the rows stores a fuel-erased normalizer, subject-reduction discharge, confluence proof, quotient syntax equality, or kernel-consistency witness. The structural rows preserve the same packet, so the refused endpoints cannot be produced by replay or provenance. □

The Lean TasteGate route for MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ must provide explicit Nontrivial MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ and FieldFaithful MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ instances. These instances must keep $T, F, N, E, Q, R, H, C, P, L$ as separate fields before any checked carrier-alignment theorem can serve a decidability-facing consumer.

Falsifiable prediction: A purported MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ consumer is refuted if it exports closed beta-conversion while omitting the displayed typed window, normalizeBounded fuel ledger, bounded normal-form outputs, decideClosedBetaConv answer, endpoint row, refusal row, transport, replay, provenance, or local NameCert row.

Independence witness: bounded-normal-equality-checker, metacic-fuel-trace, metacic-decidable-neutral-spine, metacicdecidableboundary, bounded-normal-equality-frontier: none is bijective to

MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ because this packet jointly carries typed windows, normalizeBounded fuel, bounded normal outputs, endpoint evidence, decideClosedBetaConv answers, refusal, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{MetaCICBoundedNormalEqualityBudget}^{\uparrow}}$ ·

Constructive story: A MetaCICBoundedNormalEqualityBudget$^{\uparrow}$ packet is generated from closed typed-term windows, normalizeBounded fuel ledgers, bounded normal-form output rows, endpoint rows, decideClosedBetaConv answer rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICBoundedNormalEqualityBudget}^{\uparrow}}$, seed)

Scope closed: Definition 8057.1, Definition 8057.2, and Definition 8057.3 seed the finite bounded normal-equality budget object over typed windows, fuel, endpoint, comparison, refusal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICBoundedNormalEqualityBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global closed beta-conversion decidability, fuel-erased normalization, subject reduction, confluence, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, normalizeBounded fuel exactness, bounded normal-form output exposure, endpoint exposure, decideClosedBetaConv answer exactness, refusal retention, TasteGate nontriviality, and field faithfulness for BEDC.Derived.MetaCICBoundedNormalEqualityBudgetUp.

Chapter 8058

A Concrete Naming Certificate for $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$

Following Bourbaki and Kelley on metric and uniform-space terminology, the $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$ packet records the finite certificate by which a metric space with a displayed positive separation radius presents a uniform-space entourage base. It sits between the metric carrier of chapter 57, the uniform-space interface of chapter 339, the entourage-base route of chapter 1819, and the dyadic and real-name corridor of chapter 291, chapter 126, chapter 301, and chapter 38. The chapter names a finite metric-to-uniformity certificate; it does not assert quotient equality, countable choice, selected completions, or ambient Real completeness.

Definition 8058.1 (UniformlyDiscreteSpace carrier)**.** *A* $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance comparisons and point rows are exposed before any uniform-space read is public.*
2. *R is a positive dyadic separation radius read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and sealed by the displayed* $\mathsf{Real}^{\uparrow}$ *tolerance route.*
3. *E is the* $\mathsf{EntourageBase}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *handoff row generated by the condition that distinct displayed points are separated by the radius R.*
4. *H records componentwise* hsame *transport for the metric source, radius, entourage-base handoff, and structural rows.*
5. *C records displayed* Cont *replay from metric comparisons through dyadic radius exposure and the uniformity handoff.*
6. *P is the* Pkg *provenance over* $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient metric equality, a hidden selector of apart points, countable choice, selected completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8058.2 (UniformlyDiscreteSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$ *carrier* $U = (M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, positive-radius exposure, metric separation, uniformity handoff, transport and replay stability, provenance, local naming, and nonescape from the displayed rows.*

Proof. Project the carrier of Definition 8058.1. Carrier admission exposes the metric source M, the positive dyadic radius R, the entourage-base handoff E, and the structural rows H, C, P, N.

Positive-radius exposure is read from R. Since R is a dyadic tolerance row sealed through the Real-name route, the packet exposes a finite separation budget rather than a host real parameter.

Metric separation and the uniformity handoff are read in order. The metric row M supplies the displayed distance comparisons, and E packages the radius-separated diagonal boundary as an $\mathsf{EntourageBase}^{\uparrow}$ row for $\mathsf{UniformSpace}^{\uparrow}$ consumers.

Transport through H and replay through C preserve the same route, while P and N record provenance and local naming. No further row stores quotient equality, countable choice, selected completions, or ambient Real completeness, so the listed obligations exhaust the local certificate. □

Theorem 8058.3 (UniformlyDiscreteSpace uniformity handoff)**.** *Every* $\mathsf{UniformSpace}^{\uparrow}$*-facing read of an accepted* $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{EntourageBase}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows. The handoff exports only the radius-separated metric entourage base and never supplies quotient equality, countable choice, selected completion, or ambient Real completeness.*

Proof. Start with the carrier rows M, R, E, H, C, P, N. A uniform-space consumer can read E only after the metric row M and the positive dyadic radius row R have been displayed.

The dyadic tolerance route named by R is the only numerical boundary available to the packet. Its StreamName, RegSeqRat, and Real seals provide the finite readback path for the radius, while E packages that readback as an entourage-base handoff.

The structural rows preserve the same path for downstream consumers. Thus every uniformity-facing read factors through the metric source, dyadic radius, entourage base, finite name route, Real seal, transport, replay, provenance, and local naming rows, with no hidden completion or quotient principle attached. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyDiscreteSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyDiscreteSpace}^{\uparrow}$ packet is generated from a MetricUp source row, a positive DyadicRatCore separation radius, an EntourageBase and UniformSpace handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyDiscreteSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8058.1, Definition 8058.2, and Definition 8058.3 seed the metric-to-uniformity certificate for a displayed positive separation radius.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyDiscreteSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient metric equality, countable choice, selected completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius exposure, metric separation, uniformity handoff, transport replay, provenance, local naming, and nonescape for BEDC.Derived.UniformlyDiscreteSpaceUp.

Chapter 8059

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed monotone cofinal subsequence carries the same regular Cauchy and Real-facing data as its source. It sits over the regular rational readback of chapter 301, the finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the tail-shift route of chapter 1578, and the terminal seal of chapter 38. The packet is a finite subsequence criterion for Real and Completion consumers; it does not choose a subsequence by countable choice, quotient streams, selected limits, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8059.1 (RegularCauchySubsequenceCriterion carrier)**.** *A* $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, W, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying the regular rational readback of the original name.*
2. *M is the monotone cofinal subsequence schedule, recording the finite source indices inspected for each subsequence window.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger comparing source windows with subsequence windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger transported from source windows to subsequence windows.*
5. *T is the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *tail route that witnesses discarded prefixes before the subsequence readback is exposed.*
6. *R is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the subsequence window.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, M, W, D, T, R are displayed.*
8. *H records componentwise* hsame *transport for all carrier rows.*
9. *C records displayed* Cont *replay of the same finite route.*
10. *P records* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}}$ *row.*

The classifier reads only $S, M, W, D, T, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite subsequence, host sequence equality, a chosen limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8059.2 (RegularCauchySubsequenceCriterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$ *carrier $K = (S, M, W, D, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}}$ *surface consists exactly of the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row S, monotone cofinal schedule M, finite* $\mathsf{StreamName}^{\uparrow}$ *comparison windows W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegularCauchyShift}^{\uparrow}$ *tail route T, subsequence* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, continuation replay C, provenance P, and local naming row N.*

Proof. Project the accepted carrier of Definition 8059.1. Each row named in the statement is a displayed coordinate, and the definition lists no hidden coordinate for quotient streams, selected subsequences, or completion objects.

The analytic part of the criterion is the finite chain S, M, W, D, T, R, E. The source regular readback S is compared with subsequence windows by M and W, dyadic tolerances are transported by D, discarded prefixes are recorded by T, and the subsequence readback R is sealed by E.

The structural rows H, C, P, N transport, replay, source, and name the same finite chain. Since the carrier has no row for countable choice, host sequence equality, selected limits, quotient stream equality, or ambient Real completeness, none of those endpoints can enter the local NameCert surface. □

Theorem 8059.3 (RegularCauchySubsequenceCriterion Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$ *carrier must first read the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, monotone cofinal schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegularCauchyShift}^{\uparrow}$ *tail route, and subsequence* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the seal row E is exposed. Thus the criterion hands a cofinal subsequence to the Real route as finite displayed data, without quotient streams, selected subsequences, countable choice, selected limits, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a request for the Real seal. By the carrier definition, E is not available directly from the source row; it is reached only after S, M, W, D, T, R have been inspected.

The subsequence part is finite. The schedule M names the source indices for the displayed windows, W records the compared observations, and D carries the dyadic tolerances needed to preserve regularity along those windows.

The tail route T then records the finite shift discipline before R is read as a regular rational subsequence row. Only after this route is displayed can E be consumed as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Transport, replay, provenance, and local naming preserve this same order. Because no carrier row stores quotient stream equality, a selected subsequence functional, a selected limit, host equality, countable choice, or Real completeness, the handoff cannot export those data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}$ packet is generated from a source RegSeqRat row, monotone cofinal schedule, StreamName comparison windows, DyadicRatCore tolerance ledger, RegularCauchyShift tail route, subsequence RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8059.1, Definition 8059.2, and Definition 8059.3 bind the finite subsequence criterion over RegSeqRat, StreamName, DyadicRatCore, RegularCauchyShift, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchySubsequenceCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected infinite subsequences, countable choice, selected limits, host sequence equality, host real equality, ambient Real completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, the NameCert obligation surface, the Real seal handoff, transport stability, and nonescape for BEDC.Derived.RegularCauchySubsequenceCriterionUp.

Chapter 8060

A Concrete Naming Certificate for $\mathsf{LpSpace}^{\uparrow}$

Following Bishop's constructive analysis and the Riesz tradition for normed function spaces, the $\mathsf{LpSpace}^{\uparrow}$ packet records the finite NameCert surface needed before later analysis chapters may use an L^p phrase. It depends on the measure carrier of chapter 92, the norm carrier of chapter 89, the vector-space carrier of chapter 47, the Banach-space boundary of **??**, and the real-number carrier of chapter 38.

Definition 8060.1 (Lp-space carrier)**.** *An* $\mathsf{LpSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, R, V, N, B, I, E, H, C, Q, A)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *row exposing the measurable substrate, finite measurable-window ledger, nonnegative measure readback, and local measure naming.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row supplying nonnegative real bounds, rational comparison windows, and terminal real sealing for norm values.*
3. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row recording pointwise addition, scalar action, zero, and the finite compatibility witnesses used by simple-function representatives.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *row exposing the candidate p-norm readback and its finite comparison witnesses.*
5. *B is the* $\mathsf{BanachSpace}^{\uparrow}$ *boundary row recording that any completeness read is routed through the displayed norm and Cauchy data rather than through an ambient completed quotient.*
6. *I is the finite integral-magnitude ledger over the measure row, recording simple-function windows and their nonnegative real bounds.*
7. *E is the exponent row recording the admitted p-slot, its comparison witnesses, and the refusal branch for exponent data outside the displayed range of the packet.*
8. *H records componentwise* hsame *transport for $M, R, V, N, B, I, E, C, Q, A$.*
9. *C records the displayed* Cont *routes by which measure, real bounds, vector operations, norm readback, completeness boundary, integral magnitude, exponent data, transport, provenance, and local naming are replayed.*

10. *Q is the* Pkg *provenance over* $\mathsf{LpSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *A is the local* $\mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completed L^p classes, almost-everywhere quotient equality, selected representatives, countable choice, ambient measure completion, or a host integration theorem outside the displayed finite ledger.

Lean4 ✓ `BEDC.Derived.LpSpaceUp.LpSpaceTasteGate single carrier alignment`

Theorem 8060.2 (Lp-space NameCert obligations)**.** *For the carrier of Definition 8060.1, the local* $\mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}$ *surface consists exactly of measure exposure through M, real-bound sealing through R, vector-operation exposure through V, norm readback through N, completeness-boundary routing through B, integral-magnitude exposure through I, exponent admission through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through Q, and local naming through A. Every downstream Sobolev, interpolation, Hausdorff–Young, martingale, or functional-analysis consumer must read those rows before treating an L^p phrase as a BEDC carrier.*

Lean4 ✓ `BEDC.Derived.LpSpaceUp.LpSpaceCarrier namecert obligations`

Proof. Project the accepted packet to its displayed rows. The rows M, R, V, N expose the measure, real-bound, vector-space, and norm coordinates, while B, I, E expose the completeness boundary, integral-magnitude ledger, and exponent admission row.

The remaining rows do not add hidden analytic power. Componentwise transport, continuation replay, provenance, and local naming only replay the displayed packet. Thus every accepted consumer receives exactly the listed NameCert obligations and no ambient quotient, selected representative, or external completion principle. □

Theorem 8060.3 (Lp-space norm readback handoff)**.** *Every accepted* $\mathsf{LpSpace}^{\uparrow}$ *norm read factors through the chain*

$$\mathsf{Measure}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{VecSpace}^{\uparrow},\quad \mathsf{Norm}^{\uparrow},\quad \mathsf{BanachSpace}^{\uparrow},\quad \mathsf{Pkg},\quad \mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}.$$

The handoff records finite integral-magnitude windows and exponent comparison data before any norm value is read. It does not identify functions almost everywhere, choose quotient representatives, complete the measure space, invoke countable choice, or assert that the ambient L^p completion has been constructed.

Proof. Start with the carrier projection of Definition 8060.1. A norm read can reach N only after the measure row M, real-bound row R, vector row V, integral ledger I, and exponent row E have been displayed.

The Banach-space row B is a boundary marker for later completeness consumers, not a completed quotient object. It requires the same displayed norm and Cauchy-facing data before any completeness read can be routed.

Finally apply the structural rows H, C, Q, A. They transport and replay the finite packet, so the norm readback is a NameCert handoff from the listed carriers rather than an assertion of almost-everywhere quotient equality or ambient measure completion. □

Falsifiable prediction: An $\mathsf{LpSpace}^{\uparrow}$ consumer that reads an L^p norm while omitting the measure row, exponent row, integral-magnitude ledger, norm row, or local NameCert row refutes the packet.

Independence witness: measure, real, vecspace, norm, banachspace: non-bijection follows by row omission. $\mathsf{Measure}^{\uparrow}$ lacks vector operations and norm readback; $\mathsf{Real}^{\uparrow}$ lacks measure and exponent rows; $\mathsf{VecSpace}^{\uparrow}$ lacks integration and real-bound sealing; $\mathsf{Norm}^{\uparrow}$ lacks measure exposure and exponent admission; $\mathsf{BanachSpace}^{\uparrow}$ lacks the local finite integral-magnitude ledger and does not supply an almost-everywhere quotient carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{LpSpace}^{\uparrow}$ packet is generated from measure, real-bound, vector-space, norm, Banach-boundary, integral-magnitude, exponent, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8060.1, Definition 8060.2, and Definition 8060.3 bind the finite Lp carrier, NameCert obligation surface, and norm-readback handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LpSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completed Lp classes, almost-everywhere quotient equality, selected representatives, countable choice, ambient measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligation coverage, norm-readback handoff, and field faithfulness for $M, R, V, N, B, I, E, H, C, Q, A$.

Chapter 8061

A Concrete Naming Certificate for $\mathsf{LovaszLocalLemma}^{\uparrow}$

Following Erdos–Lovasz on the Lovasz local lemma, the $\mathsf{LovaszLocalLemma}^{\uparrow}$ packet presents the finite bad-event dependency graph data over $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{FiniteBorelCantelli}^{\uparrow}$. It sits after the probability-space normalization rows of chapter 184, the finite-family independence rows of chapter 187, the random-variable preimage rows of chapter 185, and the finite exceptional-event ledger of chapter 7718. The packet is a finite dependency certificate for bad events; it does not import a completed measure algebra, countable choice, host product measure, selected avoiding assignment, or ambient compactness principle.

Definition 8061.1 (Lovasz local lemma carrier)**.** *A* $\mathsf{LovaszLocalLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\Omega, E, D, R, A, B, I, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the finite probability-space normalization and event-complement surface.*
2. E *is the finite bad-event inventory, listing the displayed event rows whose simultaneous avoidance is being certified.*
3. D *is the finite dependency graph row, recording for each bad event the displayed neighbor list on which independence may fail.*
4. R *records the* $\mathsf{RandomVar}^{\uparrow}$ *rows whose cylinder or preimage events give the bad-event rows in* E*.*
5. A *is the avoidance request row, naming the finite conjunction of complements to be kept nonzero inside the packet.*
6. B *is the finite bound ledger, recording the displayed probability bounds and dependency-degree comparisons used by the local-lemma route.*
7. I *is the* $\mathsf{Independence}^{\uparrow}$ *ledger, certifying independence of each bad event from the finite outside-neighborhood family named by* D*.*
8. H *records componentwise* hsame *transport for the probability, event, dependency, random-variable, avoidance, bound, independence, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a local-lemma request through* $\Omega, E, D, R, A, B, I, H$.
10. *P is the* Pkg *provenance over* $\mathsf{LovaszLocalLemma}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{FiniteBorelCantelli}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LovaszLocalLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a completed probability space, a host dependency theorem, a global product-measure construction, a selected avoiding assignment, quotient event equality, or countable-choice data.

Lean4 (statement) `BEDC.Derived.LovaszLocalLemmaUp`

Theorem 8061.2 (Lovasz local lemma NameCert obligations)**.** *For every accepted* $\mathsf{LovaszLocalLemma}^{\uparrow}$ *carrier* $L = (\Omega, E, D, R, A, B, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LovaszLocalLemma}^{\uparrow}}$ *surface consists exactly of probability-space source admission through* Ω*, finite bad-event exposure through* E*, dependency-graph exposure through* D*, random-variable preimage exposure through* R*, avoidance request exposure through* A*, bound-ledger exposure through* B*, outside-neighborhood independence exposure through* I*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8061.1. The rows Ω, E, D, R, A, B, I expose the finite probability source, bad-event inventory, dependency graph, random-variable preimage data, avoidance request, bound ledger, and independence ledger before any avoidance-facing consumer can inspect the local certificate.

The dependency graph row D fixes the finite neighborhood for each bad event. Thus the independence row I can only certify independence from the named outside-neighborhood family, and it cannot be replaced by an unlisted host product-measure theorem.

The finite bound ledger B is read together with the finite exceptional-event discipline of $\mathsf{FiniteBorelCantelli}^{\uparrow}$. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route. Since the carrier has no coordinate for completed measure, quotient event equality, selected avoiding assignments, or countable choice, the listed rows exhaust the NameCert surface. □

Falsifiable prediction: A purported $\mathsf{LovaszLocalLemma}^{\uparrow}$ consumer is refuted if it reaches an avoidance conclusion while omitting any of the displayed probability-space, bad-event, dependency-graph, random-variable, avoidance, bound, independence, transport, replay, provenance, or local NameCert rows.

Independence witness: probspace, independence, randomvar, finite-borel-cantelli: no listed sibling is bijective to $\mathsf{LovaszLocalLemma}^{\uparrow}$ because this packet jointly carries bad-event inventory, finite dependency graph, random-variable preimage rows, avoidance request, probability bounds, outside-neighborhood independence, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LovaszLocalLemma}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LovaszLocalLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8061.1 and Definition 8061.2 seed the finite bad-event dependency certificate over probability-space, random-variable, independence, finite exceptional-event, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LovaszLocalLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed measure algebra, host product-measure theorem, selected avoiding assignment, countable-choice principle, Lean-checked obligation theorem, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bad-event exposure, dependency-graph exposure, random-variable preimage exposure, avoidance request exposure, bound-ledger exactness, outside-neighborhood independence exposure, transport, replay, provenance, and local naming for BEDC.Derived.LovaszLocalLemmaUp.

Constructive story: A LovaszLocalLemma$^{\uparrow}$ packet is generated from ProbSpaceUp source data, finite bad-event rows, dependency graph rows, RandomVarUp preimage rows, an avoidance request, probability-bound rows, IndependenceUp outside-neighborhood rows, finite exceptional-event discipline, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8062

A Concrete Naming Certificate for $\mathsf{TrotterKato}^{\uparrow}$

Following Trotter and Kato on semigroup approximation, the $\mathsf{TrotterKato}^{\uparrow}$ packet records the finite generator-window obligations that a semigroup approximation route must expose before a downstream evolution, PDE, or operator-theoretic consumer reads convergence data. It sits after the semigroup composition surface of chapter 79, the bounded-operator and functional-analysis boundary of chapter 94, the compact or idealized operator rows of chapter 213, the terminal scalar readback of chapter 38, and the regular rational observation windows of chapter 301. The packet is a finite NameCert seed for generator and resolvent approximation; it does not assert an ambient Banach-space theorem, selected graph limits, quotient operator equality, or analytic completeness outside the displayed rows.

Definition 8062.1 (Trotter-Kato generator-window carrier)**.** *A* $\mathsf{TrotterKato}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, F, O, R, Q, G, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Semigroup}^{\uparrow}$ *composition row for the approximating evolution family and the limiting evolution row inspected by the consumer.*
2. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *source row, exposing the bounded-operator, domain-window, and graph-readback coordinates available to the approximation request.*
3. *O is the* $\mathsf{OperatorIdeal}^{\uparrow}$ *or compact-operator observation row used when the approximation is read through finite-rank, compact, or ideal-controlled windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational time, resolvent, and graph-window tolerances.*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the scalar error bounds and convergence budgets read from R.*
6. *G records the generator-window ledger: finite source-domain rows, finite graph comparison rows, and the displayed compatibility of approximating generators with the limiting generator window.*
7. *W records the resolvent-sequence window: finite resolvent probes, rational error budgets, and the handoff order by which resolvent convergence may be consumed.*

8. *H records componentwise* hsame *transport for the semigroup, functional-analysis, operator-ideal, rational-readback, Real-seal, generator-window, resolvent-window, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Trotter-Kato approximation request through S, F, O, R, Q, G, W, H.*
10. *P is the* Pkg *provenance over* $\mathsf{TrotterKato}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TrotterKato}^{\uparrow}}$ *row naming exactly this finite approximation packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host graph-limit theorem, selected core, quotient operator equality, global PDE well-posedness, or ambient Banach-space completeness.

Theorem 8062.2 (Trotter-Kato generator-window NameCert obligations)**.** *For every accepted* $\mathsf{TrotterKato}^{\uparrow}$ *carrier*

$$T = (S, F, O, R, Q, G, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TrotterKato}^{\uparrow}}$ *surface consists exactly of semigroup composition visibility, functional-analysis source visibility, operator-window visibility, regular rational readback, Real-seal error-budget visibility, generator-window compatibility, resolvent-sequence ordering, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every downstream semigroup, PDE, or operator approximation consumer must pass through $S, F, O, R, Q, G, W, H, C, P, N$ before it reads a Trotter-Kato approximation claim.*

Proof. Carrier admission is projection from Definition 8062.1. The rows S, F, O expose the semigroup source, functional-analysis boundary, and operator-window observations before any approximation claim is read.

The numeric side is read through R and Q. The regular rational windows provide the inspected tolerances and graph or resolvent budgets, and the Real seal records the scalar error bounds available to the consumer.

The approximation-specific rows are G and W. The generator-window ledger keeps finite source-domain and graph comparisons visible, while the resolvent-sequence window fixes the order in which resolvent probes may be handed to downstream consumers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no row for a host graph-limit theorem, selected core, quotient operator equality, global PDE well-posedness, or ambient completeness, these displayed rows exhaust the local NameCert obligations. □

Theorem 8062.3 (Trotter-Kato resolvent-sequence handoff)**.** *Let*

$$T = (S, F, O, R, Q, G, W, H, C, P, N)$$

be an accepted $\mathsf{TrotterKato}^{\uparrow}$ *carrier. Any consumer that turns resolvent-sequence convergence into a semigroup approximation read must first expose the functional-analysis row F, the operator-window row O, the regular rational readback R, the Real-seal row Q, and the resolvent-sequence window W before it can read the semigroup row S as an evolution handoff. The route is a finite handoff through the displayed generator-window packet, not a selected limit operator or an ambient completeness argument.*

Proof. Read the analytic source rows first. The functional-analysis row F supplies the operator and domain-window boundary, while O supplies the compact or ideal-controlled observation surface available to the approximation request.

Next read the finite numeric windows. The row R gives regular rational access to the inspected resolvent and graph tolerances, and Q seals those scalar bounds as the Real-facing budget available to the consumer.

Now apply the carrier rows G and W. The generator-window ledger fixes the source-domain and graph comparisons, and the resolvent-sequence window orders the resolvent probes before any semigroup approximation endpoint is read.

Only after those rows are displayed may the consumer read S as the semigroup evolution handoff. Transport through H, replay through C, provenance through P, and naming through N preserve exactly that route and add no selected limit operator, quotient operator equality, PDE well-posedness theorem, or ambient completeness principle. □

Falsifiable prediction: An accepted $\mathsf{TrotterKato}^{\uparrow}$ consumer that reads semigroup approximation without displaying the semigroup row, functional-analysis row, operator-window row, rational readback, Real seal, generator-window ledger, resolvent-window ledger, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: semigroup, functional-analysis, operator-ideal, regseqrat, real: no listed sibling is bijective with $\mathsf{TrotterKato}^{\uparrow}$ because this packet jointly carries semigroup composition, operator-source data, finite generator windows, resolvent-sequence ordering, rational readbacks, Real error seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TrotterKato}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TrotterKato}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8062.1, Definition 8062.2, and Definition 8062.3 seed the finite generator-window and resolvent-sequence route for Trotter-Kato approximation packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TrotterKato}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Trotter-Kato theorem, selected graph limits, quotient operator equality, PDE well-posedness, Lean verification, bridge checking, or ambient functional-analytic completeness.

Upgrade path: Obligation closure requires checked carrier admission, generator-window compatibility, resolvent-window ordering, rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TrotterKatoUp.

Constructive story: A $\mathsf{TrotterKato}^{\uparrow}$ packet is generated from SemigroupUp composition data, FunctionalAnalysisUp operator-source rows, OperatorIdealUp observation windows, RegSeqRatUp rational tolerance readbacks, RealUp scalar seals, finite generator-window ledgers, resolvent-sequence ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8063

A Concrete Naming Certificate for $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$ packet records the sequential closure surface used when a displayed regular Cauchy construction is read as a completion-facing real-name packet. It sits after the dyadic tolerance core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the effective completion route of chapter 5930. The packet names the finite sequential-closure obligations for Real and Completion consumers; it does not quotient Cauchy names, select a host limit, invoke countable choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8063.1 (BishopCompletionSequentialClosure carrier)**.** *A* $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision cells and comparison budgets inspected by the closure request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that lists the observed positions of the Cauchy name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream window and dyadic tolerance data.*
4. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, S, R have been read.*
5. *E is the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *completion-facing route row, recording that the same finite window and tolerance data are being consumed as a sequential closure handoff.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, stream schedule, regular readback, Real seal, effective completion route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from dyadic tolerances and finite schedules through regular readback, Real sealing, and completion-facing closure use.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopCompletionSequentialClosure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, A, E, H, C, P, N$. No row stores quotient stream equality, a selected representative, a host limit, countable choice, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8063.2 (BishopCompletionSequentialClosure obligations)**.** *For every accepted* BishopCompletionSequent *carrier*

$$B = (D, S, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionSequentialClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, StreamName finite-window schedule, RegSeqRat readback, Real seal, EffectiveCauchyCompletion route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The sequential-closure clause requires any completion-facing read to expose D, S, R before the Real seal A and completion route E are consumed. The non-escape clause excludes quotient Cauchy equality, selected host limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8063.1. Its displayed rows are exactly the dyadic tolerance data, finite StreamName schedule, RegSeqRat readback, Real seal, EffectiveCauchyCompletion route, and the structural rows H, C, P, N.

The sequential-closure route starts with D and S. These rows determine which dyadic budgets and observed positions are public before any regular rational or real-facing read is available.

The regular readback row R is then consumed before the terminal seal A. Only after this readback has displayed the same finite window and tolerance data may the completion-facing row E read the packet as a sequential closure handoff.

Transport through H, replay through C, and naming through P, N preserve the same displayed route. Since the carrier contains no quotient stream equality, selected representative, host limit, countable-choice row, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, every completion-facing consumer remains inside the finite certificate. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$ read is refuted if it reaches an $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer without displaying the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real, effective-cauchy-completion: $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic precision cells, finite stream windows, regular rational readback, a Real seal, an effective completion route, transport, replay, provenance, and local naming as one sequential closure packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionSequentialClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionSequentialClosure}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a Real seal, an EffectiveCauchyCompletion route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionSequentialClosure}^{\uparrow}}$, seed)

Scope closed: Definition 8063.1 and Definition 8063.2 seed the sequential closure route from dyadic finite windows and regular rational readback into Real and EffectiveCauchyCompletion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionSequentialClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite StreamName schedule exactness, RegSeqRat readback exactness, Real seal handoff, EffectiveCauchyCompletion routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionSequentialClosureUp.

Chapter 8064

A Concrete Naming Certificate for $\mathsf{RiemannSumCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RiemannSumCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed family of tagged Riemann sums may be consumed as integration-facing data only after partition, mesh, sample, sum, refinement, rational readback, and Real sealing rows have appeared. It sits beside the Riemann-sum carrier of chapter 1615, the mesh discipline of chapter 7944, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite criterion carrier for constructive integration; it does not assert open-cover compactness, selected gauges, quotient equality of functions, host supremum principles, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8064.1 (Riemann sum criterion carrier)**.** *A* $\mathsf{RiemannSumCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, M, A, S, R, H, K, G, N)$$

with the following displayed rows:

1. *P is the partition row: a finite tagged-partition ledger whose adjacent interval endpoints are read through the Riemann-sum source.*
2. *M is the mesh row: a finite mesh-bound and refinement-budget record consumed from* $\mathsf{RiemannSumMesh}^{\uparrow}$*.*
3. *A is the sample row: the finite sample-point and evaluation-window data attached to the tagged partition.*
4. *S is the sum row: the finite Riemann-sum value ledger, with all scalar readbacks routed through* $\mathsf{RegSeqRat}^{\uparrow}$ *before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is inspected.*
5. *R is the refinement row: the directed refinement comparison that records how a later partition may refine the displayed partition and preserve the inspected sum budget.*
6. *H records componentwise* hsame *transport for the partition, mesh, sample, sum, refinement, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from an integration-facing request through P, M, A, S, R, H.*
8. *G is the* Pkg *provenance over* $\mathsf{RiemannSumCriterion}^{\uparrow}$*,* $\mathsf{RiemannSum}^{\uparrow}$*,* $\mathsf{RiemannSumMesh}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumCriterion}^{\uparrow}}$ *row naming exactly this finite criterion packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected limiting partition, quotient function equality, an open-cover compactness theorem, a host supremum, countable choice, or ambient real completeness.

Theorem 8064.2 (Riemann sum criterion NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSumCriterion}^{\uparrow}$ *carrier*

$$C = (P, M, A, S, R, H, K, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannSumCriterion}^{\uparrow}}$ *surface consists exactly of partition visibility, mesh-bound visibility, sample-window visibility, finite sum readback, refinement comparison, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every downstream constructive-integration consumer must pass through* $P, M, A, S, R, H, K, G, N$ *before it reads a Riemann-sum criterion claim.*

Lean4 (statement) `BEDC.Derived.RiemannSumCriterionUp`

Proof. Carrier admission is projection from Definition 8064.1. The rows P, M, A expose the partition, mesh, and sample data before any integration-facing conclusion is read.

The numeric side is read through S. Its scalar observations are finite regular rational readbacks, and the terminal Real seal is available only after those readbacks have been displayed.

The refinement side is R. It records the finite comparison by which later tagged partitions refine the current packet and preserve the visible sum budget, without selecting an infinite refinement sequence.

Finally H, K, G, N transport, replay, source, and name the same finite route. Since the carrier contains no row for open-cover compactness, selected gauges, quotient equality of functions, host suprema, countable choice, or ambient completeness, these displayed rows exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{RiemannSumCriterion}^{\uparrow}$ consumer that reads a Riemann-sum criterion without displaying the partition, mesh, sample, finite sum, refinement, transport, replay, provenance, and local naming rows refutes the carrier surface.

Independence witness: riemannsum, riemann-sum-mesh, regseqrat, real: no listed sibling is bijective with $\mathsf{RiemannSumCriterion}^{\uparrow}$ because this packet jointly carries partition rows, mesh rows, sample rows, sum readbacks, refinement comparisons, regular rational observations, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumCriterion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSumCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8064.1 and Definition 8064.2 seed the finite Riemann-sum criterion route through partition, mesh, sample, sum, refinement, regular rational, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSumCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close a host Riemann integration theorem, open-cover compactness, selected gauges, quotient function equality, countable choice, Lean theorem verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite partition and mesh exactness, sample-window visibility, sum readback, refinement comparison, regular rational and Real nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumCriterionUp.

Constructive story: A $\mathsf{RiemannSumCriterion}^{\uparrow}$ packet is generated from RiemannSumUp partition data, RiemannSumMeshUp mesh budgets, finite sample windows, regular rational sum readbacks, terminal Real seals, finite refinement comparisons, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8065

A Concrete Naming Certificate for $\mathsf{FilterBasisConvergence}^{\uparrow}$

Following Kelley and Bourbaki on filters, bases, and convergence, the $\mathsf{FilterBasisConvergence}^{\uparrow}$ packet records the finite convergence-facing route that sits downstream of the filter-limit basis interface of chapter 3862. It exposes the same L10 completion corridor used by $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, but names the convergence request as finite BHist data rather than as quotient filters, ultrafilters, selected limits, countable choice, host equality, or ambient completeness.

Definition 8065.1 (FilterBasisConvergence carrier)**.** *A* $\mathsf{FilterBasisConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *row exposing the finite basis route already accepted by limit and completion consumers.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose accepted reads remain tied to displayed finite basis data.*
3. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *convergence-boundary row, read only after B and F have been displayed.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the convergence request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed convergence windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback at the requested precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after B, F, L, W, R, D have been replayed.*
8. *H records componentwise* hsame *transport for basis, filter, limit, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a convergence request through $B, F, L, W, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FilterBasisConvergence}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterBasisConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, L, W, R, D, E, H, C, P, N$. There is no coordinate for quotient filters, ultrafilters, selected limits, countable-choice schedules, host filter equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8065.2 (FilterBasisConvergence NameCert obligations)**.** *For every accepted* $\mathsf{FilterBasisConvergence}^{\uparrow}$ *carrier $K = (B, F, L, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterBasisConvergence}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed filter-limit basis row, Cauchy-filter source row, convergence-boundary row, finite stream window, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *basis exposure, requiring every convergence-facing read to consume B and F before it consults L, W, R, D, or E;*
3. *convergence-window compatibility, requiring the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row L to use the same displayed basis, window, readback, and dyadic tolerance route;*
4. *Real handoff, requiring the terminal seal E to remain downstream of B, F, L, W, R, D and the structural rows H, C, P, N;*
5. *ledger refusal, excluding quotient filters, ultrafilters, selected limits, countable-choice schedules, host filter equality, host real equality, ambient completeness, and bridge certificates.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 8065.1. The displayed rows are exactly $B, F, L, W, R, D, E, H, C, P, N$, so carrier admission exposes only finite basis, filter, convergence-boundary, window, readback, tolerance, seal, transport, replay, provenance, and naming data.

The convergence route starts with B and F. The filter-limit basis row and the Cauchy-filter row force the request to pass through accepted finite basis data before any convergence-facing row can be inspected.

Next read L, W, R, D. The convergence boundary consumes the same finite stream windows, regular rational readback, and dyadic tolerance ledger, so the packet has one displayed basis-convergence route rather than two unrelated filter presentations.

Finally the terminal seal E and the structural rows H, C, P, N preserve the same route. Since quotient filters, ultrafilters, selected limits, countable-choice schedules, host equality, ambient completeness, and bridge certificates are absent from the carrier, the listed obligations exhaust the local surface. □

Theorem 8065.3 (FilterBasisConvergence L10 handoff)**.** *Every accepted convergence-facing consumer of a* $\mathsf{FilterBasisConvergence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FilterLimitBasis}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyFilterLimit}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before terminal Real evidence is read. The route uses the displayed rows $B, F, L, W, R, D, E, H, C, P, N$ of the same packet and cannot import quotient filters, ultrafilters, selected limits, countable choice, host filter equality, host real equality, or ambient completeness.

Proof. Begin with a convergence-facing consumer of K. The carrier admits the $\mathsf{FilterLimitBasis}^{\uparrow}$ row B only together with the source filter row F and the convergence-boundary row L.

Replay the displayed continuation row C. The request consumes B and F, then passes through L, the finite windows W, the regular readback R, and the dyadic tolerance ledger D before the terminal seal E is available.

The structural rows do not create an additional convergence object. Transport through H, provenance through P, and local naming through N record the same finite route, so every accepted consumer receives only that route before the $\mathsf{Real}^{\uparrow}$ seal.

The excluded objects are not carrier rows. Consequently any consumer using quotient filters, ultrafilters, selected limits, countable choice, host equality, or ambient completeness has left the accepted $\mathsf{FilterBasisConvergence}^{\uparrow}$ packet rather than reading its L10 handoff surface. □

Falsifiable prediction: An accepted $\mathsf{FilterBasisConvergence}^{\uparrow}$ read is refuted if a convergence-facing or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal without displaying the FilterLimitBasis row, CauchyFilter row, CauchyFilterLimit row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterBasisConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterBasisConvergence}^{\uparrow}$ packet is generated from a FilterLimitBasis row, a CauchyFilter source row, a CauchyFilterLimit convergence-boundary row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterBasisConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8065.1, Definition 8065.2, and Definition 8065.3 seed the L10 convergence route over FilterLimitBasisUp, CauchyFilterUp, CauchyFilterLimitUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterBasisConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, ultrafilters, selected limits, countable choice, host equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, basis exposure, convergence-window compatibility, Real handoff, ledger refusal, transport preservation, replay, provenance, and local naming for BEDC.Derived.FilterBasisConvergenceUp.

Chapter 8066

A Concrete Naming Certificate for $\mathsf{RoughPath}^{\uparrow}$

Following Lyons' rough path theory, the $\mathsf{RoughPath}^{\uparrow}$ packet records an enhanced finite-variation route for paths whose iterated path-step data must be inspected before a controlled differential equation consumer may read a terminal real-name seal. It sits beside the Young integral surface of chapter 3910, the terminal Real seal of chapter 38, and the regular rational readback of chapter 301. The packet names the displayed enhancement and control-budget rows only; it is not a host rough-path quotient, selected lift, analytic existence theorem, countable-choice schedule, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8066.1 (Rough path carrier)**.** *A* $\mathsf{RoughPath}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, V, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the enhanced path-step row, listing first-level path steps, second-level iterated path-step witnesses, Chen-composition checks, and the finite interval ledger on which they are read.*
2. *V is the p-variation and control-budget row, recording the admissible variation exponent, control windows, superadditive budget checks, and the finite margins used by downstream integration requests.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the inspected time windows and endpoint requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations of path-step data, iterated path-step data, and control budgets.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, V, S, R have been displayed.*
6. *H records componentwise* hsame *transport for enhanced path-step data, variation budgets, stream observations, readback rows, the Real seal, provenance, and the local name.*
7. *C records displayed* Cont *replay from a rough-path request through enhanced path-step data, variation control, stream windows, regular rational readback, and the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RoughPath}^{\uparrow}$, $\mathsf{YoungIntegral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RoughPath}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, V, S, R, E, H, C, P, N$. It has no coordinate for a host enhancement quotient, a selected lift, a hidden area process, countable choice, an analytic existence theorem for differential equations, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8066.2 (Rough path NameCert obligations)**.** *For every accepted* $\mathsf{RoughPath}^{\uparrow}$ *carrier* $X = (I, V, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RoughPath}^{\uparrow}}$ *surface is exhausted by enhanced path-step data, p-variation and control-budget checks,* $\mathsf{StreamName}^{\uparrow}$ *observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace those rows by a host rough-path quotient, a selected lift, an unrecorded area process, countable choice, an analytic existence theorem, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 8066.1. The row I exposes the first-level and iterated path-step data together with the displayed Chen-composition checks, and V exposes the finite p-variation and control budget used to read that path-step data.

The observation and readback route is then fixed. The row S records the inspected endpoint windows, R gives the regular rational readback for the same displayed windows, and E is available only after those rows have been consumed.

Transport and replay preserve the route. The rows H and C carry componentwise hsame transport and displayed Cont replay for the same path-step, budget, observation, readback, and seal rows, while P and N record provenance and local naming.

Since the excluded objects have no coordinate in $I, V, S, R, E, H, C, P, N$, a consumer using one of them has left the accepted packet rather than reading the local NameCert surface. □

Theorem 8066.3 (Rough path p-variation handoff)**.** *Every accepted* $\mathsf{RoughPath}^{\uparrow}$ *read that is later consumed by a controlled integration or controlled differential equation interface first factors through*

$$\mathsf{YoungIntegral}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $I, V, S, R, E, H, C, P, N$. The handoff exposes enhanced path-step data and a p-variation control budget before any Real-facing read, and it exports no host lift, quotient rough path, selected area representative, countable-choice schedule, or ambient Real-completeness theorem.

Proof. Start with the enhancement and variation rows. The row I supplies the finite path-step and iterated path-step ledger, while V records the control windows and p-variation margins needed by a Young-style controlled integral request.

Next read the finite observations. The $\mathsf{StreamName}^{\uparrow}$ row S fixes the endpoint windows inspected by the consumer, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R attaches rational readback to the same windows before the $\mathsf{Real}^{\uparrow}$ seal E is exposed.

Finally use H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same finite route, so any downstream controlled-equation interface receives the enhanced path-step and control-budget data without importing a host lift, quotient rough path, selected area representative, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RoughPath}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the enhanced path-step row I, the p-variation control row V, the $\mathsf{StreamName}^{\uparrow}$ observation row S, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: YoungIntegral, StreamName, RegSeqRat, Real: none is bijective with $\mathsf{RoughPath}^{\uparrow}$ because the packet jointly carries enhanced path-step data, variation control, finite

observations, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RoughPath}\uparrow}$.

Constructive story: A $\mathsf{RoughPath}^{\uparrow}$ packet is generated from enhanced path-step data, p-variation control budgets, StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoughPath}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8066.1, Definition 8066.2, and Definition 8066.3 seed the finite enhanced path-step route for rough paths.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoughPath}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host rough-path quotients, selected lifts, hidden area processes, controlled differential equation existence, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked enhanced path-step admission, p-variation control exactness, StreamName observation, RegSeqRat readback, Real-seal nonescape, and structural transport for BEDC.Derived.RoughPathUp.

Chapter 8067

A Concrete Naming Certificate for $\mathsf{BaireOneUniformOscillation}^{\uparrow}$

Following Baire and Bishop–Bridges, $\mathsf{BaireOneUniformOscillation}^{\uparrow}$ records the finite observation surface by which a Baire-one pointwise-limit row is inspected through uniform oscillation windows before a Real-and-Completion consumer reads a terminal real seal. The packet sits after the Baire-one function surface of chapter 1604, the real oscillation surface of chapter 1839, the compact uniform-continuity route of chapter 388, the finite stream-name windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It names only the finite row discipline for oscillation windows over a pointwise-limit presentation; it does not assert uniform convergence, choose a subsequence, form quotient function equality, import compactness by open covers, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8067.1 (Baire-one uniform oscillation carrier)**.** *A* $\mathsf{BaireOneUniformOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, O, K, W, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *pointwise-limit row naming the approximating sequence and the inspected input windows.*
2. *O is the* $\mathsf{RealOscillation}^{\uparrow}$ *row carrying lower and upper oscillation probes for the same displayed real-valued data.*
3. *K is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row that bounds which compact-window and uniform-modulus requests may be replayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the pointwise approximants and oscillation probes.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerances attached to those windows.*
6. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the pointwise-limit, oscillation, uniform-window, stream, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transports for B, O, K, W, Q, T and the structural rows.*
8. *C records the displayed* Cont *replay route through the same finite window data.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireOneUniformOscillation}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BaireOneUniformOscillation}^{\uparrow}}$ *row.*
The carrier has no coordinate for host function equality, selected subsequences, uniform convergence of the approximating sequence, open-cover compactness, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8067.2 (Baire-one uniform oscillation obligations)**.** *For an accepted* $\mathsf{BaireOneUniformOscillation}^{\uparrow}$ *carrier* $U = (B, O, K, W, Q, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireOneUniformOscillation}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier formation over the displayed rows; classifier stability under componentwise* hsame *transport; pointwise-limit source compatibility with the* $\mathsf{BaireOneFunction}^{\uparrow}$ *row; oscillation-window compatibility with the* $\mathsf{RealOscillation}^{\uparrow}$ *row; compact uniform-window replay through* $\mathsf{CompactUniformContinuity}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*; and terminal* $\mathsf{Real}^{\uparrow}$ *seal non-escape.*

Proof. Carrier formation is projection from Definition 8067.1. The packet displays the pointwise-limit source, oscillation probes, compact uniform-window row, stream schedule, regular-rational readback, terminal Real seal, transports, replay, provenance, and local name.

Classifier stability is rowwise. Transport along H may replace the visible rows only by rows with the same pointwise-limit source, oscillation probes, uniform-window requests, stream windows, rational tolerances, terminal seal, replay order, provenance, and local name.

The source and oscillation obligations are the compatibility rows B and O. A consumer cannot inspect oscillation width without first displaying the Baire-one pointwise-limit row and the real oscillation probes over the same finite windows.

The uniform-window obligation is the route through K, W, Q. It confines compact-uniform requests to the displayed window schedule and rational readback, so the packet exports a finite uniform oscillation window rather than open-cover compactness or uniform convergence of the approximants.

Finally, T is terminal. Since no carrier row stores a selected subsequence, quotient function equality, quotient stream equality, or ambient completeness principle, the $\mathsf{Real}^{\uparrow}$ seal is the only value-facing endpoint exported by the packet. □

Theorem 8067.3 (Baire-one uniform oscillation window route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BaireOneUniformOscillation}^{\uparrow}$ *packet reads the rows*

$$B, O, K, W, Q, T, H, C, P, N$$

in that order before treating a Baire-one pointwise-limit surface as controlled by finite uniform oscillation windows. In particular, the consumer must pass through the $\mathsf{BaireOneFunction}^{\uparrow}$ *source,* $\mathsf{RealOscillation}^{\uparrow}$ *probe row,* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *window row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before exporting any oscillation-window conclusion.*

Proof. Begin with the source row B. It is the only displayed entry point for the Baire-one pointwise-limit presentation, so an oscillation-window consumer has no independent host function from which to read the limit.

Next read O, K, W, Q. The oscillation probes are tied to the same finite stream windows, and the compact-uniform row only bounds replay through those windows and rational tolerances.

Only after those rows have been displayed may the consumer read T. The structural rows H, C, P, N then transport, replay, source, and name the same route, so no quotient stream equality, selected subsequence, open-cover compactness proof, uniform-convergence assertion, or ambient $\mathsf{Real}^{\uparrow}$ completeness row can replace the finite window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOneUniformOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireOneUniformOscillation}^{\uparrow}$ packet is generated from a BaireOneFunction pointwise-limit row, RealOscillation probes, CompactUniformContinuity window data, StreamName schedules, RegSeqRat tolerance readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireOneUniformOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8067.1, Definition 8067.2, and Definition 8067.3 seed the finite uniform-oscillation window route for Baire-one pointwise-limit rows over the L10 Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireOneUniformOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, uniform convergence of the approximating sequence, selected subsequences, quotient function equality, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, source compatibility, oscillation-window compatibility, compact uniform-window replay, and terminal Real seal non-escape for the displayed packet.

Chapter 8068

McShaneIntegralUp NameCert construction

Following McShane's gauge integral formulation, the $\mathsf{McShaneIntegral}^{\uparrow}$ packet records the finite tagged-partition and gauge-refinement ledger by which a displayed real-valued integrand is read as having a McShane integral candidate. It sits after the terminal real-name seal of chapter 38, the regular rational tolerance route of chapter 301, the finite tagged-partition carrier of chapter 7449, and the gauge-integral boundary of chapter 8045. The packet names a finite gauge-integration certificate only; it does not import measure-theoretic integration, selected gauges, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a host equality of functions.

Definition 8068.1 (McShaneIntegral carrier)**.** *A* $\mathsf{McShaneIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, I, F, G, T, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the integration-domain row, carrying the bounded interval or displayed finite-measure cell family over which tags may be chosen.*
2. *I is the integrand row, exposing only finite* $\mathsf{Real}^{\uparrow}$ *value reads of the displayed real-valued function.*
3. *F is the finite tagged-partition row, read through* $\mathsf{TaggedPartition}^{\uparrow}$*, whose entries record cells, tags, and coverage data.*
4. *G is the gauge-admissibility row, assigning to each displayed tag a positive local radius and checking that every cell in F is subordinate to that radius.*
5. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, giving the Cauchy-style comparison budget for two admissible tagged partitions.*
6. *S is the Real-valued sum row, recording finite sums of cell-size data multiplied by the corresponding integrand reads and sealed through* $\mathsf{Real}^{\uparrow}$*.*
7. *R is the refinement ledger, relating two admissible tagged partitions by a common gauge-refinement surface rather than by a selected infinite refinement procedure.*
8. *Q is the integral-candidate row, a regular rational name whose finite windows are checked against S, T, R.*

9. *H, C, P, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over* McShaneIntegral$^{\uparrow}$, TaggedPartition$^{\uparrow}$, GaugeIntegral$^{\uparrow}$, HenstockKurzweilGaugeIntegral$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, *and* NameCert, *and the local* NameCert$_{\text{McShaneIntegral}^{\uparrow}}$ *row.*

Theorem 8068.2 (McShaneIntegral NameCert obligations)**.** *For every accepted* McShaneIntegral$^{\uparrow}$ *carrier $M = (D, I, F, G, T, S, R, Q, H, C, P, N)$, the local* NameCert$_{\text{McShaneIntegral}^{\uparrow}}$ *surface consists exactly of domain admission, integrand finite-read admission, tagged-partition coverage, gauge admissibility, regular-rational tolerance comparison, Real-valued finite-sum sealing, common-refinement ledger exposure, integral-candidate readback, componentwise transport, displayed replay, provenance, and local naming.*

The obligation surface may cite the real seal of chapter 38, the regular-rational tolerance source of chapter 301, the finite tagged-partition carrier of chapter 7449, and the gauge-integral boundary of chapter 8045. It cannot add a measure space, a selected gauge, a countable refinement sequence, a quotient equality of functions, or an ambient completeness theorem.

Proof. Project the carrier rows. The domain, integrand, tagged-partition, gauge, tolerance, sum, refinement, and candidate rows are displayed before the local naming row can be read.

The tagged-partition row supplies only finite cells and tags, and the gauge row checks admissibility cell by cell. The Real-valued sum row is therefore a finite read through Real$^{\uparrow}$, while the tolerance row keeps the comparison budget in RegSeqRat$^{\uparrow}$.

Finally the refinement ledger mediates comparisons between admissible tagged partitions. Since the carrier has no coordinate for a measure space, selected gauge, countable refinement sequence, quotient function equality, or ambient completeness argument, the local NameCert row exports exactly the finite obligation surface listed above. □

Theorem 8068.3 (McShaneIntegral gauge-refinement stability)**.** *Let $M = (D, I, F, G, T, S, R, Q, H, C, P, N)$ be an accepted* McShaneIntegral$^{\uparrow}$ *carrier. If a consumer replaces the displayed tagged partition F by another finite tagged partition that is admitted by the same gauge row G and related to F through the common-refinement ledger R, then the integral-candidate readback Q is consumed through the same tolerance and finite-sum surface*

$$T, S, R, Q, H, C, P, N.$$

The stability claim is internal to the finite gauge-refinement ledger. It is not a proof of measure-theoretic integrability, not a selected refining sequence, not a global convergence theorem for arbitrary partitions, and not a host function-equality principle.

Proof. Start with the two displayed tagged partitions and the shared gauge row. Gauge admissibility is checked by the finite row G, so each partition contributes only the cells and tags that the carrier already exposes.

Next read the common-refinement ledger R. It records the finite comparison surface between the two admissible partitions and sends their Real-valued finite sums through the same tolerance row T and sum row S.

The candidate row Q is then reached only through T, S, R and the structural rows H, C, P, N. Thus a consumer may transport the partition read inside the displayed ledger, but cannot reinterpret the result as a measure-theoretic integral, a selected refinement sequence, unrestricted partition convergence, or host equality of functions. □

Falsifiable prediction: An accepted McShaneIntegral$^{\uparrow}$ consumer that reaches the integral-candidate row without first reading the tagged-partition row, gauge-admissibility row, regular-rational tolerance row, Real-valued finite-sum row, and common-refinement ledger refutes the carrier surface.

Independence witness: real, regseqrat, tagged-partition, henstock-kurzweil-gauge-integral: no listed sibling supplies the full twelve-row packet $M = (D, I, F, G, T, S, R, Q, H, C, P, N)$, because each sibling contributes only a real seal, regular-rational tolerance route, finite tagged-partition carrier, or gauge-integral boundary while this chapter binds them into one McShane integral NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{McShaneIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{McShaneIntegral}^{\uparrow}$ packet is generated from a displayed integration domain, finite real-valued integrand reads, a finite tagged-partition row, a gauge-admissibility row, a regular-rational tolerance row, a Real-valued finite-sum row, a common-refinement ledger, an integral-candidate readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{McShaneIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8068.1, Definition 8068.2, and Definition 8068.3 seed the finite McShane gauge-integration certificate over tagged partitions, gauge-admissibility rows, Real-valued sums, regular-rational tolerance, and common-refinement ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{McShaneIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure-theoretic integration, selected gauges, countable refinement procedures, full convergence for arbitrary partition nets, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition coverage, gauge admissibility, finite-sum sealing, regular-rational tolerance comparison, refinement-ledger stability, and a Lean carrier/obligation formalization for BEDC.Derived.McShaneIntegralUp.

Chapter 8069

A Concrete Naming Certificate for $\mathsf{MengerTheorem}^{\uparrow}$

Following Menger on finite graph connectivity, the $\mathsf{MengerTheorem}^{\uparrow}$ packet records the cut-path duality surface as finite BEDC data. It sits after the finite graph source of chapter 118, the flow certificate route of chapter 233, the matching certificate route of chapter 234, the finite list accounting surface of chapter 36, and the natural-number ledger of chapter 28. The packet names the displayed vertex pair, internally disjoint path family, separator row, cardinality comparison, flow/matching handoff, transport, replay, provenance, and local certificate; it is not a host max-flow theorem, not a selected infinite path family, not quotient graph equality, and not an ambient compactness or choice principle.

Definition 8069.1 (Menger theorem carrier)**.** *A* $\mathsf{MengerTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, s, t, \Pi, S, K, F, Q, L, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *source row, carrying the finite vertex inventory, edge inventory, incidence data, and endpoint-adjacency checks.*
2. *s and t are the two distinguished vertex rows, with the endpoint-separation witness that keeps the cut-path question nontrivial inside G.*
3. *Π is the internally disjoint path-family row, represented by finite* $\mathsf{List}^{\uparrow}$ *data whose entries are paths from s to t and whose internal vertex ledgers are pairwise disjoint.*
4. *S is the separator row, recording a finite vertex cut that meets every displayed s–t path while excluding the two distinguished endpoints.*
5. *K is the* $\mathsf{Nat}^{\uparrow}$ *cardinality ledger, recording the common finite size comparison between the path family and the separator.*
6. *F is the* $\mathsf{NetworkFlow}^{\uparrow}$ *handoff row, reading the path and cut data through the finite unit-capacity flow boundary rather than through a host max-flow theorem.*
7. *Q is the* $\mathsf{Matching}^{\uparrow}$ *handoff row, recording the auxiliary bipartite matching view used by finite alternating-route consumers.*
8. *L records the* $\mathsf{List}^{\uparrow}$ *accounting discipline for path inventories, separator inventories, and duplicate checks.*

9. *A records the* $\mathsf{Nat}^{\uparrow}$ *accounting discipline for lengths, family size, separator size, and equality of the two displayed finite cardinals.*
10. *H records componentwise* hsame *transport for graph source, endpoints, path-family, separator, cardinality, flow, matching, list, natural-number, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a cut-path duality query through $G, s, t, \Pi, S, K, F, Q, L, A, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{MengerTheorem}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite graph, a host max-flow/min-cut theorem, quotient equality of paths, selected path families outside the packet, or choice over separators.

Theorem 8069.2 (Menger theorem NameCert obligations)**.** *For every accepted* $\mathsf{MengerTheorem}^{\uparrow}$ *carrier $M = (G, s, t, \Pi, S, K, F, Q, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MengerTheorem}^{\uparrow}}$ *surface consists exactly of graph source admission through G, endpoint exposure through s, t, internally disjoint path-family exposure through Π, separator exposure through S, cardinality-ledger exposure through K, network-flow handoff through F, matching handoff through Q, list accounting through L, natural-number accounting through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8069.1. The graph and endpoint rows G, s, t fix the finite source of the cut-path question before either side of the duality can be read.

The path-family row Π and separator row S are the two visible witnesses. The list ledger L records path inventories, duplicate checks, and separator enumeration, while the natural-number ledger A records the finite sizes inspected by the cardinality row K.

The handoff rows F and Q keep the same data readable by network-flow and matching consumers. They do not replace the packet by an ambient max-flow/min-cut theorem or by a selected matching oracle; they replay the displayed graph, path, cut, and counting rows.

Transport, continuation replay, provenance, and local naming are exactly H, C, P, N. Since no further coordinate stores infinite graph data, quotient path equality, a choice-selected path family, or a host theorem, the listed rows exhaust the NameCert obligations. □

Theorem 8069.3 (Menger theorem cut-path duality handoff)**.** *Every cut-path duality consumer of an accepted* $\mathsf{MengerTheorem}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Graph}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{NetworkFlow}^{\uparrow},\ \mathsf{Matching}^{\uparrow} \longmapsto G, s, t, \Pi, S, K, F, Q, L, A, H, C, P, N.$$

The handoff states only that the displayed internally disjoint path-family row and displayed separator row meet in the same finite cardinality ledger. It does not export a theorem for infinite graphs, quotient equality of path objects, selected witnesses outside the packet, arbitrary host max-flow/min-cut, or countable choice.

Proof. Begin with Definition 8069.2. A consumer must expose the finite graph source and the two endpoint rows before it can inspect either the internally disjoint path family or the separator.

Next read the list and natural-number ledgers. The path-family row Π is a finite $\mathsf{List}^{\uparrow}$ inventory of paths, the separator row S is a finite $\mathsf{List}^{\uparrow}$ inventory of vertices, and K, A record the $\mathsf{Nat}^{\uparrow}$ equality between their displayed sizes.

Finally read the flow and matching handoffs. The $\mathsf{NetworkFlow}^{\uparrow}$ row presents the same data as a unit-capacity cut-flow boundary, while the $\mathsf{Matching}^{\uparrow}$ row presents the same finite incidence discipline to alternating-route consumers. Both handoffs remain attached to G, s, t, Π, S, K, L, A.

Thus the duality consumer receives a finite cut-path certificate and no stronger host theorem. The refused endpoints remain outside the carrier because no row supplies infinite graph data, quotient path equality, selected external witnesses, arbitrary max-flow/min-cut, or countable choice. □

Falsifiable prediction: A purported $\mathsf{MengerTheorem}^{\uparrow}$ consumer is refuted if it reaches a cut-path duality conclusion while omitting any of the displayed graph source, endpoint, internally disjoint path-family, separator, cardinality, network-flow, matching, list, natural-number, transport, replay, provenance, or local NameCert rows.

Independence witness: graph, networkflow, matching, list, nat: no listed sibling is bijective to $\mathsf{MengerTheorem}^{\uparrow}$ because this packet jointly carries a distinguished graph source, two endpoint rows, internally disjoint path-family rows, separator rows, cardinality equality, flow handoff, matching handoff, finite list accounting, natural-number accounting, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MengerTheorem}^{\uparrow}$ packet is generated from a GraphUp source row, two distinguished vertex rows, an internally disjoint path-family row, a separator row, a NatUp cardinality ledger, NetworkFlowUp and MatchingUp handoff rows, ListUp and NatUp accounting rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MengerTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8069.1, Definition 8069.2, and Definition 8069.3 seed the finite graph cut-path duality certificate over graph, path-family, separator, counting, flow, matching, list, natural-number, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MengerTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite graph Menger theorems, quotient path equality, selected external witnesses, arbitrary host max-flow/min-cut, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, internally disjoint path-family exposure, separator exposure, cardinality-ledger exactness, NetworkFlow and Matching handoff exposure, List and Nat accounting, transport, replay, provenance, and local naming for BEDC.Derived.MengerTheoremUp.

Chapter 8070

A Concrete Naming Certificate for $\mathsf{FiniteTailSeries}^{\uparrow}$

Following Hardy and Bishop–Bridges, $\mathsf{FiniteTailSeries}^{\uparrow}$ records the finite tail ledger by which a displayed series is read as Cauchy-facing data before any completion consumer asks for a real value. The packet sits beside the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback row of chapter 301, the terminal real-name seal of chapter 38, and the reusable tail envelope $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$. It gives Real-and-Completion consumers a finite series-tail handoff without importing an infinite sum object, a selected limit, quotient sequence equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8070.1 (FiniteTailSeries carrier)**.** *A* $\mathsf{FiniteTailSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, D, R, E, H, C, P, N)$$

with the following five-row NameCert surface:

1. *The source row S names the displayed finite series ledger and the tail interval being inspected.*
2. *The observation row W records the* $\mathsf{StreamName}^{\uparrow}$ *windows of terms and partial sums available on that finite tail.*
3. *The tolerance row D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic budgets assigned to the tail comparison.*
4. *The readback row R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational comparison by which the finite tail is handed to Cauchy consumers.*
5. *The seal row E records the terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *handoff reached only after S, W, D, R have been displayed.*

The structural rows H, C, P, N carry componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTailSeries}^{\uparrow}}$ *row for the same surface. The carrier has no coordinate for an infinite sum object, selected limit, quotient sequence equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8070.2 (FiniteTailSeries Cauchy handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{FiniteTailSeries}^{\uparrow}$ *packet must read the finite source ledger S, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the terminal seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N before treating the tail as Cauchy-series data.*

Proof. Project the carrier of Definition 8070.1. The analytic rows are S, W, D, R, E, and the structural rows are H, C, P, N.

The source row fixes the finite tail interval. The StreamName row then exposes the term and partial-sum windows, and the DyadicRatCore row attaches the rational tolerance budgets to those same windows.

The RegSeqRat readback row is downstream from the windows and tolerance ledger. It is therefore the first point at which the finite tail may be consumed as regular Cauchy data, and the terminal seal E remains behind that readback route.

The structural rows transport, replay, source, and name the same ordered route. Since no carrier coordinate stores an infinite sum object, selected limit, quotient sequence equality, countable choice, or ambient real-completeness principle, the Cauchy-series handoff is exactly the displayed finite tail ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailSeries}^{\uparrow}$ packet is generated from a finite series-tail source ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp or RegularCauchyTailEnvelopeUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTailSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8070.1 and Definition 8070.2 seed the finite tail Cauchy-series ledger used by Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTailSeries}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not provide Lean verification, an infinite sum object, a selected limit, quotient sequence equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, terminal seal handoff, transport, replay, provenance, and local naming for $\mathsf{FiniteTailSeries}^{\uparrow}$.

Chapter 8071

A Concrete Naming Certificate for $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$ packet records the finite Real-and-Completion route by which two regular Cauchy names may be interleaved and read as one completion-facing Bishop real name. It sits beside the regular rational carrier of chapter 301, the finite window surface of chapter 126, the terminal Real seal of chapter 38, the parity substream route of chapter 5876, and the interleaving criterion of chapter 3405. The packet is a finite-means certificate for manipulating regular Cauchy real names; it is not a quotient representative, selected limit, countable-choice schedule, host equality of reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8071.1 (Bishop real Cauchy interleaving carrier)**.** *A* $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S_0, S_1, W_0, W_1, D_0, D_1, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, each carrying a regular Cauchy name before any Real-facing seal is read.*
2. *W_0 and W_1 are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows that expose the inspected even-side and odd-side observation windows.*
3. *D_0 and D_1 are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers attached to the two source rows.*
4. *J is the* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *row, recording the finite selector and common-window comparison that licenses the interleaving.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the interleaved regular Cauchy name produced from the two source rows and the common-window criterion.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after $S_0, S_1, W_0, W_1, D_0, D_1, J, R$ have been displayed.*
7. *H records componentwise* hsame *transport for the two source rows, the two window rows, the two tolerance ledgers, the interleaving criterion, the interleaved readback, the Real seal, provenance, and local naming.*
8. *C records displayed* Cont *replay from any Real or Completion request through exactly the same finite rows.*

9. *P is the* Pkg *provenance over* BishopRealCauchyInterleaving$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyEvenOddSubstreams$^{\uparrow}$, CauchyInterleavingCriterion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopRealCauchyInterleaving}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed coordinates. It contains no coordinate for quotient-name equality, selected representatives, countable choice, host real equality, or ambient Real completeness.

Theorem 8071.2 (Bishop real Cauchy interleaving NameCert obligations)**.** *For every accepted* BishopRealCauchyInterleaving$^{\uparrow}$ *carrier*

$$I = (S_0, S_1, W_0, W_1, D_0, D_1, J, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopRealCauchyInterleaving}^{\uparrow}}$ *surface consists exactly of two regular Cauchy source rows, two finite window rows, two dyadic tolerance ledgers, one interleaving criterion row, the interleaved regular readback row, the terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must pass through these rows before reading the interleaved Bishop real name.*

Lean4 ✓ `BEDC.Derived.BishopRealCauchyInterleavingUp.BishopRealCauchyInterleavingCarrier namecert obligations`

Proof. Project the carrier of Definition 8071.1. The rows S_0, S_1 expose the two regular Cauchy names, while W_0, W_1 restrict the read to finite stream windows and D_0, D_1 attach the visible dyadic tolerance budgets.

The interleaving step is then forced through J. That row is the finite CauchyInterleavingCriterion$^{\uparrow}$ comparison, so the interleaved readback R can be consumed only after the common-window selector and the two source tolerance ledgers have been displayed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no row for quotient-name equality, selected representatives, countable choice, host real equality, or ambient completeness, these displayed coordinates exhaust the local NameCert obligations. □

Theorem 8071.3 (Bishop real Cauchy interleaving window route)**.** *Let*

$$I = (S_0, S_1, W_0, W_1, D_0, D_1, J, R, E, H, C, P, N)$$

be an accepted BishopRealCauchyInterleaving$^{\uparrow}$ *carrier. Any finite Real or Completion request that reads the interleaved name factors through the sibling route*

RegularCauchyEvenOddSubstreams$^{\uparrow}$, CauchyInterleavingCriterion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$

on the same BHist, hsame, Cont, Pkg, *and* NameCert *rows. The route exposes the even and odd source windows before the interleaved readback and exports no quotient stream, selected representative, countable-choice schedule, host real equality, or ambient Real-completeness theorem.*

Proof. Start with the parity-facing sibling. The RegularCauchyEvenOddSubstreams$^{\uparrow}$ route supplies the standard even and odd window discipline, so any request for the interleaved name first reads finite windows W_0, W_1 over the two displayed source rows.

Next apply the interleaving criterion row J. It binds the two window routes to one common finite comparison before the interleaved RegSeqRat$^{\uparrow}$ readback R and terminal Real$^{\uparrow}$ seal E are consumed.

The structural rows H, C, P, N preserve this route inside the same BHist packet. Therefore every accepted request remains on the displayed Bishop real Cauchy interleaving surface and cannot import any of the excluded global objects. □

Falsifiable prediction: An accepted $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$ consumer is refuted if it reaches the $\mathsf{Real}^{\uparrow}$ seal without first displaying both $\mathsf{RegSeqRat}^{\uparrow}$ source rows, both $\mathsf{StreamName}^{\uparrow}$ window rows, both $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, the $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ row, the interleaved readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, real, regular-cauchy-even-odd-substreams, cauchy-interleaving-criterion: no listed sibling is bijective with $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$ because this packet jointly carries two source regular Cauchy names, two finite window rows, two dyadic tolerance ledgers, one common interleaving criterion, one interleaved readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealCauchyInterleaving}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealCauchyInterleaving}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two StreamName finite-window rows, two DyadicRatCore tolerance ledgers, a CauchyInterleavingCriterion row, an interleaved RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealCauchyInterleaving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8071.1, Definition 8071.2, and Definition 8071.3 seed the finite Bishop real Cauchy interleaving route over regular Cauchy sources, finite windows, dyadic tolerance ledgers, interleaving criteria, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealCauchyInterleaving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient-name equality, selected representatives, countable-choice schedules, host real equality, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-row visibility, finite-window exactness, dyadic tolerance compatibility, interleaving criterion stability, interleaved readback nonescape, Real-seal boundary control, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealCauchyInterleavingUp.

Chapter 8072

A Concrete Naming Certificate for $\mathsf{ConditionalConvergence}^{\uparrow}$

Following Bishop and Bridges on constructive real series, the $\mathsf{ConditionalConvergence}^{\uparrow}$ packet records the finite route for a real series whose displayed partial sums converge while the absolute-value series is kept as a separate refused branch. It sits over the $\mathsf{RealSeries}^{\uparrow}$ source of chapter 1581, the absolute convergence boundary of chapter 1700, the alternating-test sibling of chapter 1483, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names conditional convergence as a convergence-with-refusal certificate, not as absolute convergence, an alternating-test proof, a selected infinite sum, quotient equality of series, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8072.1 (Conditional convergence carrier)**.** *A* $\mathsf{ConditionalConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, C, A, W, Q, D, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing only finite term and partial-sum windows.*
2. *C is the convergence row, recording the shared Cauchy-tail threshold and partial-sum comparison route for S.*
3. *A is the absolute-branch refusal row: it records that any* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *read must enter through the separate absolute-value series route and is not supplied by K.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected terms, partial sums, and absolute-value probes.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed partial-sum windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by both the convergence row and the absolute-branch refusal.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after C, W, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for source, convergence, refusal, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from series windows through convergence and refusal checks to the readback and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{ConditionalConvergence}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConditionalConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, C, A, W, Q, D, E, H, T, P, N$. It has no coordinate for an absolute-convergence proof, an alternating-series-test derivation, a host infinite sum, selected limits, quotient stream equality, countable choice, or ambient real completeness.

Theorem 8072.2 (Conditional convergence NameCert obligations)**.** *For every accepted* $\mathsf{ConditionalConvergence}^{\uparrow}$ *carrier $K = (S, C, A, W, Q, D, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConditionalConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure, convergence-row exposure, absolute-branch refusal, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 8072.1. The convergence side exposes only the real-series row S, the Cauchy-tail route C, the finite windows W, the regular readback Q, the dyadic ledger D, and the seal E.

Read the refusal row A separately. It does not provide an $\mathsf{AbsoluteConvergence}^{\uparrow}$ witness; it records that any absolute-value series read must be supplied by the sibling absolute-convergence chapter and cannot be recovered from the conditional packet alone.

The alternating-test sibling is likewise only a possible producer of examples or routes, not a coordinate of this carrier. The packet may cite $\mathsf{AlternatingSeriesTest}^{\uparrow}$ provenance when the source came from that route, but the public conditional-convergence certificate remains the displayed convergence-with-refusal surface.

Finally H, T, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for absolute convergence, selected sums, quotient series equality, countable choice, or real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8072.3 (Conditional convergence sibling separation)**.** *Every Real-facing read of an accepted* $\mathsf{ConditionalConvergence}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

and keeps the $\mathsf{AbsoluteConvergence}^{\uparrow}$ *and* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *routes as distinct sibling routes. A consumer may not replace the absolute-branch refusal by an absolute-convergence proof, nor replace the conditional carrier by an alternating-test proof unless that proof is supplied as an external provenance row.*

Proof. Start with an accepted carrier and a Real-facing request. The request reaches E only after the real-series source, convergence row, finite windows, regular readback, and dyadic ledger have been displayed.

The absolute branch is controlled by A. Because A is a refusal row rather than a convergence row for the absolute-value series, the request cannot read an $\mathsf{AbsoluteConvergence}^{\uparrow}$ certificate from the conditional packet.

The alternating-test route is separated by provenance. It can explain a source of examples, but it is not part of the classifier coordinates $S, C, A, W, Q, D, E, H, T, P, N$ unless explicitly recorded as provenance, and it does not erase the absolute-branch refusal.

Thus the public read is conditional convergence: convergence of the displayed real series together with separation from absolute convergence and from a specific alternating-test derivation. □

Falsifiable prediction: An accepted $\mathsf{ConditionalConvergence}^{\uparrow}$ read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the RealSeries source, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, absolute-branch refusal, transport, replay, provenance, and local NameCert rows.

Independence witness: realseries, absoluteconvergence, alternating-series-test, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{ConditionalConvergence}^{\uparrow}$, because only this packet jointly carries a convergent RealSeries route and a separate refusal of the AbsoluteConvergence route while retaining StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ConditionalConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ConditionalConvergence}^{\uparrow}$ packet is generated from a RealSeries source row, convergence-tail row, absolute-branch refusal row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConditionalConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8072.1, Definition 8072.2, and Definition 8072.3 seed the finite convergence-with-refusal route distinguishing conditional convergence from absolute convergence and alternating-test provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConditionalConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close absolute convergence, alternating-series-test correctness, selected infinite sums, quotient series equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convergence-row exposure, absolute-branch refusal, window and readback exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConditionalConv

Chapter 8073

A Concrete Naming Certificate for $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ packet records the finite diagonal schedule that synchronizes compact-net centers, pointwise modulus rows, rational lower-bound folds, and Real-facing readback windows. It realizes the compact-uniform finite-net route of chapter 9237 beside the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the uniform-modulus boundary of chapter 369, the finite Lebesgue-number ledger of chapter 452, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite schedule, not an open-cover compactness theorem, selected subcover, host diagonal argument, countable choice principle, or ambient completeness theorem.

Definition 8073.1 (Compact uniform diagonal net carrier)**.** *A* $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, L, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including inspected centers, coverage windows, and metric source data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row over the same compact source and target metric surface.*
3. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *or rational lower-bound ledger that turns the displayed finite cover into a positive radius budget.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row reached only after K, F, L have been displayed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *diagonal schedule row listing the finite source windows, center indices, target tolerances, and output windows inspected together.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows and uniform-modulus handoff.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing reads of the scheduled outputs.*
8. *H records componentwise* hsame *transport for compact-net, continuous-map, radius-ledger, uniform-modulus, stream-schedule, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a tolerance request through K, F, L, U, S, Q, E.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalNet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, F, L, U, S, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers outside K, host diagonal choice, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8073.2 (Compact uniform diagonal net NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ *carrier $G = (K, F, L, U, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalNe}}$ *surface consists exactly of carrier admission, compact finite-net exposure, continuous-map pointwise-modulus exposure, finite radius-ledger exposure, uniform-modulus handoff, diagonal StreamName schedule exposure, RegSeqRat readback, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 8073.1. The compact finite-net row K, continuous-map row F, radius ledger L, and uniform-modulus row U are the only compact-uniform source data available to the classifier.

Read the diagonal schedule row S. It lists the finite windows inspected together, so the schedule is a StreamName row rather than a host diagonal choice over infinitely many requests.

The readback row Q and seal row E are downstream of that schedule. They can inspect only the windows and tolerances already displayed by S and the uniform-modulus handoff U.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for open-cover compactness, selected subcovers, host diagonal choice, quotient stream equality, selected limits, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8073.3 (Compact uniform diagonal net schedule handoff)**.** *Every Real-facing read of an accepted* $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

The read first displays the compact-net and pointwise-modulus rows, then the finite radius ledger and uniform-modulus handoff, then the StreamName diagonal schedule, regular rational readback, and Real seal.

Proof. Start with the carrier rows $K, F, L, U, S, Q, E, H, C, P, N$. The uniform-modulus handoff U is downstream of the compact metric row, pointwise modulus row, and finite radius ledger.

Next read the diagonal schedule. The schedule row S synchronizes center indices, source windows, tolerances, and output windows inside one finite $\mathsf{StreamName}^{\uparrow}$ surface.

The regular readback Q and Real seal E consume only that displayed schedule and the handoff U. The structural rows preserve the same finite path under transport, replay, provenance, and local naming.

Thus every Real-facing read remains inside the compact-uniform finite-net diagonal schedule and cannot add an open-cover compactness theorem, selected subcover, host diagonal selector, quotient stream equality, selected limit, or ambient real completeness. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ read is refuted if a RealUp seal is reached without displaying the CompactMetricUp finite-net row, ContinuousMapUp pointwise-modulus row, FiniteLebesgueNumberUp radius ledger, UniformModulusUp handoff, StreamNameUp diagonal schedule, RegSeqRatUp readback, transport, replay, provenance, and local NameCert rows.

Independence witness: compactmetric, continuousmap, uniformmodulus, finite-lebesgue-number, streamname, regseqrat, real: none is bijective to $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$, because only this packet binds compact finite-net coverage, pointwise moduli, finite radius ledgers, a uniform-modulus handoff, a diagonal StreamName schedule, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformDiagonalNet}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, FiniteLebesgueNumber radius ledgers, UniformModulus handoff rows, StreamName diagonal schedules, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8073.1, Definition 8073.2, and Definition 8073.3 seed the compact-uniform finite-net diagonal schedule over compact metric, continuous-map, finite-radius, uniform-modulus, stream, regular readback, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host diagonal choice, quotient stream equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, pointwise-modulus exposure, finite-radius exposure, uniform-modulus handoff, diagonal StreamName scheduling, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformDiagonalNetUp.

Chapter 8074

A Concrete Naming Certificate for $\mathsf{RealOpenBall}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealOpenBall}^{\uparrow}$ packet records a $\mathsf{Real}^{\uparrow}$ open ball by finite centre data, a positive dyadic radius, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, and terminal Real seal rows. It consumes the real-name surface of chapter 38, the stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic core ledger of chapter 291, and the general neighbourhood boundary of chapter 2191. The packet gives later $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ consumers a finite real-neighbourhood row without importing host topology, quotient real equality, selected limits, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8074.1 (Real open ball carrier)**.** *A* $\mathsf{RealOpenBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (c, r, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *c is the centre* $\mathsf{Real}^{\uparrow}$ *seal row, already exposed through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
2. *r is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, with its positivity witness and tolerance ledger displayed before any neighbourhood membership read is consumed.*
3. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened from the centre and tested real names at the requested precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W.*
5. *D is the dyadic comparison ledger for the distance-from-centre estimate, recording the finite rational inequality needed to certify membership inside the radius row r.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the tested name after the windows, readback, and dyadic comparison have been displayed.*
7. *H records componentwise* hsame *transport for centre, radius, windows, readback, comparison, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-neighbourhood query through c, r, W, R, D, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RealOpenBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{OpenBall}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RealOpenBall}^\uparrow}$ *row.*
The classifier compares accepted packets only through $c, r, W, R, D, E, H, C, P, N$. It has no coordinate for host topology, quotient real equality, selected limits, open-cover compactness, real-order trichotomy, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8074.2 (Real open ball NameCert obligations)**.** *For every accepted* $\mathsf{RealOpenBall}^\uparrow$ *carrier $B = (c, r, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealOpenBall}^\uparrow}$ *surface consists exactly of carrier admission, positive dyadic radius admission, finite-window membership, classifier transport, Real-seal handoff, displayed replay, provenance, and local naming for the rows $c, r, W, R, D, E, H, C, P, N$.*

Proof. Carrier admission projects the finite packet of Definition 8074.1. The only real source rows are the centre seal c, the terminal tested seal E, and the window-readback route W, R between them.

The radius obligation is the positive dyadic row r. Its positivity and tolerance ledger are visible before the comparison row D can certify that the tested name lies inside the displayed neighbourhood.

Finite-window membership is therefore exhausted by W, R, D. The StreamName windows expose the requested finite observations, the RegSeqRat row reads those observations back as regular rational data, and the DyadicRatCore comparison row records the strict radius test.

Finally H, C, P, N transport, replay, source, and name exactly the same route. Since no further coordinate is present, the carrier cannot export host topology, quotient real equality, selected limits, open-cover compactness, real-order trichotomy, or ambient $\mathsf{Real}^\uparrow$ completeness. □

Theorem 8074.3 (Real open ball dyadic radius stability)**.** *Let $B = (c, r, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealOpenBall}^\uparrow$ *carrier. If r' is a positive* $\mathsf{DyadicRatCore}^\uparrow$ *radius row whose displayed tolerance ledger refines the ledger of r, and the same comparison row D certifies the tested name inside r', then the packet*

$$B' = (c, r', W, R, D, E, H, C, P, N)$$

uses the same finite route and remains a $\mathsf{RealOpenBall}^\uparrow$ *carrier with the same centre, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.*

Proof. Project the accepted carrier B. All non-radius coordinates are already fixed by the finite route from centre and tested Real seals through StreamName windows, RegSeqRat readback, and the dyadic comparison row.

Replace only the radius row by the displayed positive refinement r'. The hypothesis says that the refined tolerance ledger is still a $\mathsf{DyadicRatCore}^\uparrow$ row and that the same comparison row D certifies membership inside that radius.

The structural rows need no new data. Componentwise transport H, replay C, provenance P, and local naming N are read over the same coordinates after substituting r' for r in the radius slot.

Thus B' exposes the same finite neighbourhood route. The stability statement does not supply a chosen smaller radius outside the packet, a quotient-real equality, a trichotomy decision, or any ambient completeness theorem. □

Theorem 8074.4 (Real open ball StreamName–Real handoff)**.** *Every neighbourhood consumer of an accepted* $\mathsf{RealOpenBall}^\uparrow$ *carrier reaches* $\mathsf{Real}^\uparrow$ *only through the finite route*

$$\mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

recorded by $W, R, D, c, E, H, C, P, N$. In particular, a consumer cannot bypass the StreamName windows, detach the RegSeqRat readback, replace the DyadicRatCore radius test by host real equality, or obtain ambient $\mathsf{Real}^{\uparrow}$ *completeness from the neighbourhood row.*

Proof. Begin with the finite windows W. A neighbourhood request is allowed to inspect only the displayed StreamName windows for the centre and tested real names.

The row R is then the only regular rational readback available to the consumer. It converts those windows into the rational data used by the dyadic comparison ledger.

The radius test is exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ row D together with the positive radius row r. Only after that finite comparison is displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed beside the centre seal c.

The remaining rows H, C, P, N preserve and name this order of access. Therefore every Real-facing read factors through StreamName, RegSeqRat, and DyadicRatCore rows, and no host topology, quotient real equality, selected limit, open-cover compactness, trichotomy, or ambient completeness row is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOpenBall}^{\uparrow}}$.

Constructive story: A $\mathsf{RealOpenBall}^{\uparrow}$ packet is generated from a RealUp centre seal, a positive DyadicRatCoreUp radius row, StreamName windows, RegSeqRat readback, a dyadic comparison ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealOpenBall}^{\uparrow}}$, seed)

Scope closed: Definition 8074.1, Definition 8074.2, Definition 8074.3, and Definition 8074.4 seed the finite real-neighbourhood route from displayed Real seals through StreamName, RegSeqRat, and DyadicRatCore rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealOpenBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host topology, quotient real equality, selected limits, open-cover compactness, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic radius admission, finite-window membership, classifier transport, Real-seal handoff, displayed replay, provenance, and local naming for BEDC.Derived.RealOpenBallUp.

Chapter 8075

A Concrete Naming Certificate for $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ packet records the finite L10 comparison surface used when two displayed real-name routes must be compared before a Real or Completion consumer can read an order-facing conclusion. It sits after the observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real seal of chapter 38, and the Bishop real comparison surface of chapter 1381. The packet names a located comparison route over finite rows; it does not import trichotomy, quotient equality, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8075.1 (BishopLTenComparisonPrinciple carrier)**.** *A* $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected source positions for the two comparison routes.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic scales, endpoint cells, and comparison budgets consumed by the readback.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular-rational, and dyadic rows have been displayed.*
5. *Q is the located comparison evidence row. It records a displayed separating witness or overlap certificate for the inspected window, tied to the sibling* $\mathsf{BishopRealComparison}^{\uparrow}$ *route.*
6. *H records componentwise* hsame *transport for the stream schedule, regular readback, dyadic ledger, Real seal, located comparison evidence, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from finite observation and dyadic tolerance data through regular readback, Real sealing, and located comparison use.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, R, D, E, Q, H, C, P, N$. No coordinate stores a global trichotomy branch, quotient real equality, selected representative, host equality test, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8075.2 (BishopLTenComparisonPrinciple NameCert obligations)**.** *For every accepted* $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ *carrier*

$$K = (S, R, D, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName finite-window row, RegSeqRat readback row, DyadicRatCore tolerance ledger, terminal Real seal, located comparison evidence, componentwise transport, displayed replay, package provenance, and local naming. Every comparison-facing consumer must expose S, R, D before reading the Real seal E or the located comparison row Q.*

Proof. Project the carrier of Definition 8075.1. Its analytic rows are exactly the finite StreamName window, regular rational readback, dyadic tolerance ledger, Real seal, and located comparison evidence.

The route begins with S. The observed source positions determine which finite prefixes of the two real-name routes are public. The readback row R and the dyadic ledger D then provide the rational and tolerance data on which the comparison evidence can be interpreted.

Only after those rows have been displayed may the terminal seal E and the located comparison row Q be consumed. The structural rows H, C, P, N transport, replay, source, and name the same finite route.

The excluded principles are absent by coordinate inspection. Since the carrier has no row for global trichotomy, quotient equality, selected representatives, host equality, countable choice, or ambient Real completeness, the local certificate cannot export any of them. □

Theorem 8075.3 (BishopLTenComparisonPrinciple Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ *packet receives located comparison evidence only through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BishopRealComparison}^{\uparrow}.$$

Thus the handoff may use the inspected window, regular readback, dyadic tolerance ledger, terminal Real seal, and located comparison row, but it exports no trichotomy principle, quotient equality of real names, host equality, selected representative, countable-choice functional, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8075.2. They force each comparison-facing consumer to read S, R, D before the Real seal E or located comparison evidence Q is available.

The Real-facing part of the handoff is therefore downstream of the finite observation and tolerance rows. The Completion-facing part receives the same route: it may replay the inspected comparison through H, C, P, N, but it cannot replace the displayed rows by quotient names or host equality.

The sibling route through $\mathsf{BishopRealComparison}^{\uparrow}$ records only located evidence for the inspected window. Since global trichotomy, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not coordinates of K, no Real or Completion consumer can project them from the handoff. □

Falsifiable prediction: An accepted $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ read is refuted if it reaches a Real or Completion comparison consumer without displaying the $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ seal, located $\mathsf{BishopRealComparison}^{\uparrow}$ evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, real, bishop-real-comparison: $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite observation windows, regular rational readback, dyadic tolerance data, a Real seal, located comparison evidence, transport, replay, provenance, and local naming as one L10 comparison packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance data, a Real seal, located BishopRealComparison evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8075.1, Definition 8075.2, and Definition 8075.3 seed the L10 located comparison packet over StreamName, RegSeqRat, DyadicRatCore, Real, and BishopRealComparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLTenComparisonPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, trichotomy, quotient real equality, host equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located comparison exactness, Real handoff, Completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopLTenComparisonPrincipleUp.

Chapter 8076

A Concrete Naming Certificate for $\mathsf{DirichletJordanTest}^{\uparrow}$

Following Jordan and Dirichlet on bounded variation and trigonometric series, the $\mathsf{DirichletJordanTest}^{\uparrow}$ packet records the finite BEDC surface needed to hand a bounded-variation row to a Fourier partial-sum consumer through a Dirichlet-kernel window. It sits between the bounded-variation carrier of chapter 398, the Fourier row of chapter 124, the kernel window of chapter 2311, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the classical test as finite NameCert data without claiming host pointwise convergence, chosen representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8076.1 (Dirichlet–Jordan test carrier)**.** *A* $\mathsf{DirichletJordanTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, K, F, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded-variation row read from* $\mathsf{BoundedVariation}^{\uparrow}$*, naming the finite partition-variation ledger available to the test.*
2. *K is the* $\mathsf{DirichletKernel}^{\uparrow}$ *window row, listing the displayed trigonometric-kernel indices and dyadic tail window under inspection.*
3. *F is the* $\mathsf{Fourier}^{\uparrow}$ *partial-sum row, exposing only the finite coefficient and summation data consumed with K.*
4. *O is the dyadic oscillation-budget row that bounds the inspected jump and adjacent-window oscillation data for the same bounded-variation ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficient, kernel, and oscillation windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed real-valued reads.*
7. *H records componentwise* hsame *transport for the bounded-variation, kernel-window, Fourier partial-sum, oscillation-budget, readback, Real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a test request through B, K, F, O, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and sibling* $\mathsf{NameCert}$ *packets.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletJordanTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, K, F, O, R, E, H, C, P, N$. It has no coordinate for a host limit of Fourier partial sums, a selected representative of the underlying real function, countable choice over partitions, or ambient real completeness.

Theorem 8076.2 (Dirichlet–Jordan test NameCert obligations)**.** *For every accepted* $\mathsf{DirichletJordanTest}^{\uparrow}$ *carrier* $D = (B, K, F, O, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirichletJordanTest}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-variation ledger exposure, Dirichlet-kernel window exposure, Fourier partial-sum exposure, dyadic oscillation-budget exactness, regular rational readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8076.1. The displayed rows expose the bounded-variation ledger, the Dirichlet-kernel window, the Fourier partial-sum row, the dyadic oscillation budget, the regular rational readback, and the Real seal before any consumer can replay the test.

The row order fixes the obligation surface. A Fourier-facing read first consumes the $\mathsf{BoundedVariation}^{\uparrow}$ row, then the $\mathsf{DirichletKernel}^{\uparrow}$ and $\mathsf{Fourier}^{\uparrow}$ rows for the inspected partial sum, and only then reads the oscillation and regular-rational budgets through O and R.

The structural rows H, C, P, N transport, replay, source, and name the same finite data. Since no carrier coordinate stores a host convergence theorem, a selected representative, countable choice over partitions, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8076.3 (Dirichlet–Jordan Fourier handoff)**.** *Every Fourier consumer of an accepted* $\mathsf{DirichletJordanTest}^{\uparrow}$ *carrier reads the bounded-variation ledger of chapter 398, the kernel window of chapter 2311, the partial-sum row of chapter 124, and the regular readback and Real seal of chapter 301 and chapter 38 through the single finite route*

$$B, K, F, O, R, E, H, C, P, N.$$

The handoff exports only the displayed finite test surface; it does not assert pointwise convergence of the Fourier series in the host theory.

Proof. Begin with an accepted carrier. By Definition 8076.2, the Fourier consumer receives the bounded-variation row B before it reaches the kernel and partial-sum rows K and F.

The oscillation-budget row O and the regular readback row R keep the inspected jump, adjacent-window, coefficient, and kernel data finite. The Real seal E is therefore read only after those displayed rational windows have been exposed.

Finally H, C, P, N preserve the handoff as one named BEDC packet. No row in that packet selects a Fourier-series limit, quotients real-valued functions, chooses a representative, or imports ambient completeness, so the handoff remains a finite sibling route rather than a host convergence theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletJordanTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DirichletJordanTest}^{\uparrow}$ packet is generated from a bounded-variation row, Dirichlet-kernel window, Fourier partial-sum row, dyadic oscillation budget, regular rational readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DirichletJordanTest}^{\uparrow}}$, seed)

Scope closed: Definition 8076.1, Definition 8076.2, and Definition 8076.3 seed the finite Dirichlet–Jordan test surface at the bounded-variation and Fourier boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DirichletJordanTest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host pointwise Fourier convergence, quotient equality of real-valued functions, selected representatives, countable choice over partitions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation ledger exposure, Dirichlet-kernel window exposure, Fourier partial-sum exposure, oscillation-budget exactness, regular readback, Real sealing, transport, replay, provenance, and local naming for the same finite packet.

Chapter 8077

A Concrete Naming Certificate for $\mathsf{RealClosedBall}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's boundary language, the $\mathsf{RealClosedBall}^{\uparrow}$ packet records a closed real ball by finite center data, a positive dyadic radius, StreamName windows, RegSeqRat readback, a non-strict dyadic distance ledger, and terminal Real seal rows. It consumes the real-name surface of chapter 38, the stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic core ledger of chapter 291, the strict real-neighbourhood surface of chapter 8074, the finite-window real ball carrier of chapter 317, and the metric closed-ball boundary of chapter 1451. The packet gives located interval, compact interval, and metric boundary consumers a non-strict real-neighbourhood row without importing open-cover compactness, trichotomy, quotient real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8077.1 (Real closed ball carrier)**.** *A* $\mathsf{RealClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (c, r, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *c is the center* $\mathsf{Real}^{\uparrow}$ *seal row, exposed through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any boundary comparison is read.*
2. *r is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, including its positivity witness and tolerance ledger.*
3. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened from the center and tested real names at the requested precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows.*
5. *D is the non-strict dyadic distance ledger, recording the finite rational comparison that certifies the tested name at distance at most the radius row r.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the tested name after W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for center, radius, windows, readback, non-strict comparison, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a closed-neighbourhood query through c, r, W, R, D, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RealClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealOpenBall}^{\uparrow}$, $\mathsf{RealBall}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RealClosedBall}^\uparrow}$ *row.*
The classifier compares accepted packets only through $c, r, W, R, D, E, H, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient real equality, selected limits, host topology closure, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8077.2 (Real closed ball NameCert obligations)**.** *For every accepted* $\mathsf{RealClosedBall}^\uparrow$ *carrier $B = (c, r, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealClosedBall}^\uparrow}$ *surface consists exactly of carrier admission, positive dyadic radius admission, finite-window non-strict membership, classifier transport, Real-seal handoff, displayed replay, provenance, and local naming for the rows $c, r, W, R, D, E, H, C, P, N$.*

Proof. Carrier admission projects the finite packet of Definition 8077.1. The only Real-facing source rows are the center seal c, the tested seal E, and the window-readback route W, R connecting them.

The radius obligation is the positive dyadic row r. Its positivity and tolerance ledger are visible before the non-strict comparison row D can certify closed-ball membership.

Finite-window membership is exhausted by W, R, D. The StreamName windows expose finite observations, the RegSeqRat row reads those observations as regular rational data, and the DyadicRatCore row records the at-most-radius comparison.

Finally H, C, P, N transport, replay, source, and name the same route. Since no further coordinate is present, the packet cannot export open-cover compactness, trichotomy, quotient real equality, selected limits, host topology closure, or ambient $\mathsf{Real}^\uparrow$ completeness. □

Theorem 8077.3 (Real open and closed boundary duality)**.** *Every boundary consumer that compares an accepted* $\mathsf{RealClosedBall}^\uparrow$ *carrier with a strict* $\mathsf{RealOpenBall}^\uparrow$ *or finite-window* $\mathsf{RealBall}^\uparrow$ *carrier must read both sides through the shared route*

$$\mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The closed side exports only the non-strict dyadic distance row D, while the open side exports the strict radius row of chapter 8074. A metric-facing consumer may pass this boundary to chapter 1451, but it receives no open-cover compactness, trichotomy, quotient real equality, selected limit, host topology closure, or ambient $\mathsf{Real}^\uparrow$ *completeness theorem.*

Proof. Start with the closed carrier. Its rows c, r, W, R, D, E display the center, radius, finite windows, rational readback, non-strict comparison, and tested Real seal before any boundary read is available.

Read the open side through the same finite source route. The $\mathsf{RealOpenBall}^\uparrow$ and $\mathsf{RealBall}^\uparrow$ packets expose StreamName windows, RegSeqRat readback, DyadicRatCore radius data, and terminal Real seals; they differ from the closed packet only in the strict comparison row exported to the consumer.

The duality is therefore a boundary routing statement. Closed reads keep the non-strict ledger D, open reads keep the strict radius ledger, and both are replayed through H, C, P, N rather than through host real topology.

Finally, a metric boundary consumer can consume the non-strict row through the $\mathsf{MetricClosedBall}^\uparrow$ surface, but the row remains the displayed finite comparison. No carrier coordinate selects limits, decides trichotomy, quotients real names, proves open-cover compactness, or imports ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealClosedBall}^\uparrow}$.

Constructive story: A RealClosedBall$^{\uparrow}$ packet is generated from a RealUp center seal, a positive DyadicRatCoreUp radius row, StreamName windows, RegSeqRat readback, a non-strict dyadic distance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealClosedBall}\uparrow}$, seed)

Scope closed: Definition 8077.1, Definition 8077.2, and Definition 8077.3 seed the finite closed real-neighbourhood route from displayed Real seals through StreamName, RegSeqRat, DyadicRatCore, RealOpenBall, RealBall, and MetricClosedBall rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealClosedBall}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient real equality, selected limits, host topology closure, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic radius admission, finite-window non-strict membership, classifier transport, Real-seal handoff, displayed replay, provenance, and local naming for BEDC.Derived.RealClosedBallUp.

Chapter 8078

A Concrete Naming Certificate for $\mathsf{BishopCut}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCut}^{\uparrow}$ packet records a constructive two-sided cut as finite located lower and upper information before any quotient-real completion or Dedekind completeness principle is available. It consumes the located-cut surface of chapter 404, the lower-real row of chapter 1446, the upper-real envelope of chapter 5121, the dyadic tolerance core of chapter 291, the stream-name windows of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the finite cut carrier and its local handoffs; it does not claim a completed real quotient, trichotomy, countable choice, or a Dedekind-completeness theorem.

Definition 8078.1 (BishopCut carrier)**.** *A* $\mathsf{BishopCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, E, D, S, R, X, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *row recording the finite located lower and upper tests visible in the current window.*
2. *U is the paired* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperRealEnvelope}^{\uparrow}$ *row that separates lower information from upper-refusal information.*
3. *E is the envelope-consistency row linking the located-cut ledger to the lower and upper presentations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for rational comparison inside the displayed window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row assembled from D and S.*
7. *X is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after L, U, E, D, S, R have been replayed.*
8. *H records componentwise* hsame *transport for the located cut, lower-real row, upper-envelope row, dyadic tolerance row, stream window, regular readback, real seal, replay, provenance, and local naming row.*
9. *C records displayed* Cont *replay from a cut request through $L, U, E, D, S, R, X, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperRealEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCut}^{\uparrow}}$ *row.*

The accepted carrier has no coordinate for quotient reals, global Dedekind cuts, trichotomy, countable choice, selected decimal expansions, or host completeness.

Lean4 (statement) `BEDC.Derived.BishopCutUp`

Theorem 8078.2 (BishopCut NameCert obligations)**.** *For every accepted* $\mathsf{BishopCut}^{\uparrow}$ *carrier* $K = (L, U, E, D, S, R, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCut}^{\uparrow}}$ *surface consists of carrier admission for the displayed located-cut row, lower-real and upper-envelope rows, dyadic tolerance ledger, stream window, regular-rational handoff, terminal real seal, transport, replay, provenance, and local naming row. Every accepted read factors through those rows and refuses quotient reals, global Dedekind completeness, trichotomy, countable choice, selected decimal expansions, and host-completeness data.*

Proof. Project the carrier of Definition 8078.1. The visible rows are exactly $L, U, E, D, S, R, X, H, C, P, N$, so the local NameCert can admit only those finite rows.

The analytic part begins with the located-cut row L and the lower-real and upper-envelope row U. The envelope-consistency row E binds those presentations before dyadic tolerance data in D and stream-window data in S are read.

The Real-facing part is downstream. The row R repacks the displayed window as regular-rational readback, and X seals only that readback. The structural rows H, C, P, N transport, replay, source, and name the same route without adding a quotient, choice, trichotomy, decimal-selection, or completeness coordinate. □

Theorem 8078.3 (BishopCut located window)**.** *Every located-window read of an accepted* $\mathsf{BishopCut}^{\uparrow}$ *packet exposes the route*

$$\mathsf{LocatedCut}^{\uparrow} \longrightarrow \mathsf{LowerReal}^{\uparrow}, \mathsf{UpperRealEnvelope}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

The read is therefore local to $L, U, E, D, S, R, X, H, C, P, N$ and cannot bypass the located-cut ledger by invoking a quotient real, a completed Dedekind cut, trichotomy, countable choice, or host completeness.

Proof. Begin with a located-window request. By Definition 8078.1, the request must first inspect L and the paired lower and upper rows in U.

The finite comparison data is then forced through E, D, S. The envelope-consistency row keeps the lower and upper presentations attached to the located-cut ledger, while the dyadic and stream rows determine the precision window actually in scope.

Finally the handoff row R and the seal row X supply regular-rational and Real-facing readback. Apply Definition 8078.2 to see that H, C, P, N preserve this route and that no refused host object has a carrier coordinate. □

Theorem 8078.4 (BishopCut Real handoff)**.** *For every accepted* $\mathsf{BishopCut}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$ *consumer receives a terminal seal only after the located-cut row, lower-real row, upper-real envelope, dyadic tolerance ledger, stream-name window, and regular-rational readback have all been displayed. The handoff is a finite NameCert route from constructive cut data to a real-name seal, not a claim of Dedekind completeness or quotient-real equality.*

Proof. Read the terminal seal request. The carrier definition makes X the only $\mathsf{Real}^{\uparrow}$ row, and it is listed after L, U, E, D, S, R.

Use Definition 8078.3 to identify the required upstream route. A Real-facing consumer has already passed through the located-cut ledger, lower and upper rows, dyadic tolerance data, stream window, and regular readback before it sees X.

The structural rows cannot strengthen the result. They only transport, replay, source, and name the same finite handoff, so the theorem exports a real-name seal while leaving completeness, quotient equality, trichotomy, and choice outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCut}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCut}^{\uparrow}$ packet is generated from a located-cut row, lower-real row, upper-real envelope, dyadic tolerance ledger, StreamName window, RegSeqRat handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCut}^{\uparrow}}$, seed)

Scope closed: Definition 8078.1, Definition 8078.2, Definition 8078.3, and Definition 8078.4 bind the finite two-sided constructive cut seed over the Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCut}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.BishopCutUp`

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient real equality, trichotomy, countable choice, selected decimal expansions, host completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, located-window routing, Real handoff, classifier stability, ledger exactness, and bridge eligibility for BEDC.Derived.BishopCutUp.

Chapter 8079

A Concrete Naming Certificate for FiniteCandidateDiamond$^{\uparrow}$

The FiniteCandidateDiamond$^{\uparrow}$ packet realizes the recursor-as-authorized-generator audit route of chapter 9373 as a finite candidate-diamond ledger for MetaCIC confluence work. It sits after the authorized-generator recursor surface of chapter 525, the recursor-generator packet of chapter 534, the MetaCIC critical path of chapter 765, the parallel diamond frontier of chapter 4238, and the candidate strong-normalization boundary of chapter 2274. The packet records only finite local-join candidates, branch-audit rows, and route handoffs; it does not assert full confluence, normalization, global Church–Rosser, or host equality reflection.

Definition 8079.1 (FiniteCandidateDiamond carrier)**.** *A* FiniteCandidateDiamond$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, R, M, F, S, J, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the* AuthorizedGeneratorRecursor$^{\uparrow}$ *row certifying that the inspected generator call is tied to an accepted constructor signature and branch package.*
2. *R is the* RecursorGenerator$^{\uparrow}$ *row recording the finite motive, branch, and output route.*
3. *M is the* MetaCICCriticalPath$^{\uparrow}$ *row naming the local term-shape and reduction-frontier context.*
4. *F is the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *row exposing the two candidate one-step fronts.*
5. *S is the* CandidateSNConfluenceBoundary$^{\uparrow}$ *row recording the normalization-facing candidate boundary consumed by the audit.*
6. *J is the finite local-join candidate ledger, storing only displayed joins between the two frontier rows.*
7. *T is the TasteGate-expectation row for a Nontrivial carrier and FieldFaithful readback of the displayed rows.*
8. *H records componentwise* hsame *transport for authorized-generator, recursor-generator, critical-path, frontier, boundary, join-ledger, TasteGate, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a candidate-diamond audit through $A, R, M, F, S, J, T, H, P, N$.*
10. *P is the* Pkg *provenance over* FiniteCandidateDiamond$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, RecursorGenerator$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, CandidateSNConfluenceBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCandidateDiamond}^{\uparrow}}$ *row.*
The accepted carrier has no coordinate for global confluence, full normalization, quotient equality, proof irrelevance, host equality reflection, or an unrestricted search oracle.

Lean4 (statement) `BEDC.Derived.FiniteCandidateDiamondUp`

Theorem 8079.2 (FiniteCandidateDiamond local-join obligation)**.** *For every accepted* $\mathsf{FiniteCandidateDiamond}^{\uparrow}$ *packet $D = (A, R, M, F, S, J, T, H, C, P, N)$, each local-join obligation is read through the authorized-generator row A, recursor-generator row R, MetaCIC critical-path row M, parallel diamond frontier F, candidate strong-normalization boundary S, finite join ledger J, TasteGate-expectation row T, and the structural rows H, C, P, N. The obligation is local to the displayed candidate diamond and does not claim full confluence, normalization, global Church–Rosser, quotient equality, or host equality reflection.*

Proof. Project the carrier of Definition 8079.1. The generator authorization and recursor generator rows A, R are the only rows that can justify the branch package being audited.

Read the MetaCIC rows next. The critical-path row M fixes the term-shape context, the frontier row F exposes the two candidate one-step fronts, and the boundary row S records the normalization-facing candidate boundary in scope.

The local join is then finite. The ledger J stores the displayed candidate joins, while T states the expected Nontrivial and FieldFaithful TasteGate surface for the carrier. The structural rows H, C, P, N preserve and name the same route, so no unrestricted global confluence or host equality principle enters the obligation. □

Theorem 8079.3 (FiniteCandidateDiamond MetaCIC handoff)**.** *Every MetaCIC consumer of an accepted* $\mathsf{FiniteCandidateDiamond}^{\uparrow}$ *packet receives candidate-diamond evidence only through the route*

$$\mathsf{AuthorizedGeneratorRecursor}^{\uparrow} \longrightarrow \mathsf{RecursorGenerator}^{\uparrow} \longrightarrow \mathsf{MetaCICCriticalPath}^{\uparrow} \longrightarrow \mathsf{MetaCICParallelDiamondFronti}$$

The handoff exposes a finite audit packet for candidate-mediated joins, not a proof of unrestricted confluence, normalization, quotient equality, or host equality reflection.

Proof. Start with a MetaCIC consumer. The carrier forces the consumer to inspect A and R before it may treat any recursor-produced branch as an authorized generator output.

The consumer then reads M, F, S. These rows tie the audit to a critical-path term context, a displayed parallel frontier, and a candidate strong-normalization boundary rather than to an ambient reduction universe.

Finally apply Definition 8079.2. The finite ledger J, TasteGate-expectation row T, and structural rows H, C, P, N exhaust the handoff. Since the refused principles have no carrier row, the exported packet remains a finite MetaCIC audit route. □

Falsifiable prediction: An accepted $\mathsf{FiniteCandidateDiamond}^{\uparrow}$ audit is refuted if a candidate local join is exposed without displaying authorizedgeneratorrecursor, recursorgenerator, metaciccriticalpath, metacic-parallel-diamond-frontier, candidate-sn-confluence-boundary, finite join-ledger, TasteGate expectation, replay, provenance, and local naming rows.

Independence witness: authorizedgeneratorrecursor, recursorgenerator, metaciccriticalpath, metacic-parallel-diamond-frontier, candidate-sn-confluence-boundary: none alone is bijective with $\mathsf{FiniteCandidateDiamond}^{\uparrow}$ because the packet jointly records generator authorization, recursor branch

data, critical-path context, parallel frontier, candidate boundary, local join ledger, TasteGate expectation, transport, replay, provenance, and NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCandidateDiamond}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCandidateDiamond}^{\uparrow}$ packet is generated from authorized-generator recursor data, recursor-generator branch data, MetaCIC critical-path rows, parallel diamond frontier rows, candidate strong-normalization boundary rows, a finite local-join ledger, TasteGate expectation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCandidateDiamond}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8079.1, Definition 8079.2, and Definition 8079.3 bind the finite candidate-diamond audit seed for recursor-generator MetaCIC work.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCandidateDiamond}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FiniteCandidateDiamondUp`

Bridge status: none

Not claimed: This chapter does not claim full confluence, normalization, global Church–Rosser, quotient equality, proof irrelevance, host equality reflection, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, local-join obligation routing, MetaCIC handoff, TasteGate Nontrivial and FieldFaithful expectations, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCandidateDiamon

Chapter 8080

A Concrete Naming Certificate for $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$

Following Birkhoff and von Neumann on doubly stochastic matrices, the $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ packet records the finite permutation-flow route by which a displayed doubly stochastic matrix is read as a convex ledger of permutation matrices. It sits after the finite matrix surface of chapter 49, the certified finite-position action rows of chapter 116, the capacity route of chapter 233, the finite matching surface of chapter 234, and the affine membership rows of chapter 208. The packet names only the finite BEDC route from matrix rows to permutation-flow rows and convex-combination ledgers; it is not the full Birkhoff polytope theorem over arbitrary host matrices.

Definition 8080.1 (Birkhoff–von Neumann finite carrier)**.** *A* $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, R, C, F, K, E, W, T, Q, H, J, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *source row, recording the finite square matrix entries and the finite index spine on which all row and column reads occur.*
2. *R is the row-stochastic ledger, displaying each finite row sum and nonnegative-entry read.*
3. *C is the column-stochastic ledger, displaying each finite column sum and the same nonnegative-entry boundary.*
4. *F is the* $\mathsf{NetworkFlow}^{\uparrow}$ *support route, naming the bipartite residual graph whose positive entries are available for decomposition.*
5. *K is the* $\mathsf{Matching}^{\uparrow}$ *row that exposes one finite perfect-matching or permutation-support step inside the currently displayed residual support.*
6. *E is the extraction row that records the scalar weight removed from the residual matrix along the displayed matching.*
7. *W is the convex-combination ledger, listing the finite weights, their total-one certificate, and the permutation-matrix rows they weight.*
8. *T is the transport row taking each matching row into a* $\mathsf{Permutation}^{\uparrow}$ *matrix exposure over the same finite index spine.*
9. *Q is the* $\mathsf{ConvexSet}^{\uparrow}$ *affine-membership row witnessing that the displayed convex combination remains inside the matrix carrier surface.*

10. *H records componentwise* hsame *transport for matrix, stochastic, flow, matching, extraction, convex-ledger, permutation-exposure, affine-membership, replay, provenance, and local naming rows.*
11. *J records displayed* Cont *replay from finite stochastic rows through support flow, matching exposure, extraction, convex ledger, and permutation-matrix exposure.*
12. *P is the* Pkg *provenance over* $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these thirteen displayed rows. It does not read a host polytope, an arbitrary matrix space, an ambient linear-programming theorem, quotient matrix equality, choice-selected decompositions, or an infinite compactness argument.

Theorem 8080.2 (Birkhoff–von Neumann NameCert obligations)**.** *For every accepted* $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ *packet*

$$B = (M, R, C, F, K, E, W, T, Q, H, J, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}}$ *obligation surface consists exactly of finite matrix admission, row-stochastic exposure, column-stochastic exposure, finite network-flow support, matching extraction, scalar residual extraction, convex-combination ledger, permutation-matrix exposure, convex-set affine membership, componentwise transport, displayed replay, provenance, and local naming. Every downstream consumer must display $M, R, C, F, K, E, W, T, Q$ before reading a decomposition-facing or convexity-facing conclusion.*

Proof. Project the carrier of Definition 8080.1. The matrix-facing data are exactly M, R, C: the finite square matrix entries, the row sums, the column sums, and the nonnegative-entry boundary.

The decomposition-facing data are F, K, E, W, T. The support graph is read through the finite $\mathsf{NetworkFlow}^{\uparrow}$ route, a displayed $\mathsf{Matching}^{\uparrow}$ row supplies one permutation support, the extraction row records the removed scalar weight, and the ledger W records the resulting finite convex combination. The transport row T turns each matching into the corresponding $\mathsf{Permutation}^{\uparrow}$ matrix exposure over the same index spine.

The convex-facing data are then Q, H, J, P, N. The $\mathsf{ConvexSet}^{\uparrow}$ row reads the finite weight ledger and permutation-matrix exposures, while transport, replay, provenance, and local naming keep the same rows attached to the public certificate.

No excluded principle is a coordinate of the carrier. Since there is no row for a host polytope, arbitrary matrix space, ambient linear-programming theorem, quotient matrix equality, choice-selected decomposition, or infinite compactness argument, the local NameCert surface cannot export any of them. □

Theorem 8080.3 (Birkhoff–von Neumann permutation-flow route)**.** *Every decomposition-facing read exported by an accepted* $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Mat}^{\uparrow} \longmapsto \mathsf{NetworkFlow}^{\uparrow} \longmapsto \mathsf{Matching}^{\uparrow} \longmapsto \mathsf{Permutation}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow}.$$

The route first displays the finite doubly stochastic matrix rows, then reads the residual support as a network-flow surface, exposes matching rows as permutation matrices, records the convex-combination ledger, and transports the result through replay, provenance, and the local NameCert *row. It does not claim the full Birkhoff polytope theorem over arbitrary host matrices.*

Proof. Start with the NameCert obligations of Definition 8080.2. They require every decomposition-facing consumer to inspect the matrix row M, the row-stochastic ledger R, and the column-stochastic ledger C before any matching or convex ledger can be read.

Next read the support route. The $\mathsf{NetworkFlow}^{\uparrow}$ row F exposes only the finite residual support graph, and the $\mathsf{Matching}^{\uparrow}$ row K supplies the displayed perfect-matching step used by the extraction row E.

The transport row T converts that matching support into a $\mathsf{Permutation}^{\uparrow}$ matrix exposure on the same finite index spine. The convex ledger W and affine row Q then record the weighted finite list through $\mathsf{ConvexSet}^{\uparrow}$.

The structural rows H, J, P, N replay and name the same route. Because every stage is tied to the displayed finite rows, the read cannot be replaced by a host polytope theorem, quotient matrix equality, selected decomposition oracle, or ambient compactness principle. □

Falsifiable prediction: An accepted $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ read is refuted if it exports a permutation-matrix convex decomposition without displaying the $\mathsf{Mat}^{\uparrow}$ source row, row-stochastic ledger, column-stochastic ledger, $\mathsf{NetworkFlow}^{\uparrow}$ support route, $\mathsf{Matching}^{\uparrow}$ extraction row, scalar residual extraction row, convex-combination ledger, $\mathsf{Permutation}^{\uparrow}$ matrix exposure, $\mathsf{ConvexSet}^{\uparrow}$ affine-membership row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: matrix, permutation, networkflow, matching, convexset: $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite doubly stochastic matrix rows, residual support flow, matching extraction, scalar residual weights, permutation-matrix exposure, convex-combination membership, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$ packet is generated from finite MatrixUp source rows, row-stochastic and column-stochastic ledgers, NetworkFlowUp support rows, MatchingUp extraction rows, scalar residual extraction, convex-combination ledgers, PermutationUp matrix exposures, ConvexSetUp affine-membership rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8080.1, Definition 8080.2, and Definition 8080.3 seed the finite permutation-flow certificate over MatrixUp, NetworkFlowUp, MatchingUp, PermutationUp, and ConvexSetUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, the full Birkhoff polytope theorem over arbitrary host matrices, host matrix equality, selected decompositions, linear-programming duality, or compactness.

Upgrade path: Obligation closure requires checked carrier admission, stochastic ledger exposure, finite support-flow routing, matching extraction, scalar residual extraction, permutation-matrix exposure, convex-combination membership, transport, replay, provenance, and local naming for BEDC.Derived.BirkhoffVonNeumannFiniteUp.

Chapter 8081

A Concrete Naming Certificate for $\mathsf{FareyApproximation}^{\uparrow}$

Following Hardy–Wright on rational approximation through Farey neighbors, the $\mathsf{FareyApproximation}^{\uparrow}$ packet records the finite rational-density route that passes from denominator-bounded Farey rows to real-name approximation consumers. It sits between the ordered neighbor surface of chapter 1759, the branch-address surface of chapter 1704, the rational readback window of chapter 1668, the regular rational sequence route of chapter 301, and the terminal real-name interface of chapter 38. The packet names finite inspection data for a displayed real input and tolerance budget; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, a global choice of best approximants, or a quotient-real equality theorem.

Definition 8081.1 (FareyApproximationUp carrier)**.** *A* $\mathsf{FareyApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E, Q, B, F, S, R, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *input name whose finite rational window is being inspected.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the rational samples available at the requested budget.*
3. *B is the denominator and tolerance bound row.*
4. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *neighbor row, recording the adjacent rational endpoints selected at the displayed denominator level.*
5. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *branch row that replays the same endpoints as a finite left-right address.*
6. *R is the rational output row admitted by the Farey bracket.*
7. *W is the finite error-window row witnessing that R lies inside the displayed tolerance read from B and Q.*
8. *H records componentwise* hsame *transports for the real name, readback, bound, neighbor, branch, rational output, window, provenance, and local certificate rows.*
9. *C records the displayed* Cont *replays by which consumers read the finite approximation window.*
10. *P is the* Pkg *provenance over* $\mathsf{FareyApproximation}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{FareyApproximation}^\uparrow}$ *row.*
No coordinate supplies a selected best approximation, an infinite Farey process, a real-completeness proof, or a countable-choice selector.

Theorem 8081.2 (FareyApproximationUp NameCert obligations)**.** *For every accepted* $\mathsf{FareyApproximation}^\uparrow$ *carrier A, the local* $\mathsf{NameCert}_{\mathsf{FareyApproximation}^\uparrow}$ *surface consists exactly of carrier admission for*

$$E, Q, B, F, S, R, W, H, C, P, N,$$

Farey-neighbor exposure, Stern–Brocot branch replay, rational readback from the displayed real-name samples, finite tolerance-window exactness, componentwise transport, continuation replay, provenance, and nonescape from selected-best-approximant or $\mathsf{Real}^\uparrow$ *completeness claims.*

Proof. Project the carrier of Definition 8081.1. The rows E, Q, B, F, S, R, W are the whole mathematical surface: a displayed real name, a finite rational readback, a denominator and tolerance budget, the Farey neighbor bracket, the matching Stern–Brocot branch address, a rational output row, and the finite window check.

The route through chapter 1759 supplies the ordered neighbor inspection, while chapter 1704 supplies the branch replay for the same bracket. The rows H and C keep these reads componentwise and displayed.

The provenance row P and local row N record that no other obligation is available. Since no coordinate carries a selector for best approximants or a real-completeness witness, the NameCert surface stops at the finite rational-density window. □

Theorem 8081.3 (FareyApproximationUp real-density handoff)**.** *Every accepted* $\mathsf{FareyApproximation}^\uparrow$ *carrier exposes a finite rational-density handoff*

$$\mathsf{Real}^\uparrow \longleftarrow \mathsf{RegSeqRat}^\uparrow \longleftarrow \mathsf{FareySequence}^\uparrow \longleftrightarrow \mathsf{SternBrocot}^\uparrow \longrightarrow \mathsf{RealRationalApproximation}^\uparrow$$

for the displayed input row E and budget row B. The handoff gives downstream consumers the rational output R together with the finite error window W, but it does not select a denominator-optimal approximant and does not close the ambient completeness of $\mathsf{Real}^\uparrow$*.*

Proof. Read Q from the displayed real-name input E at the budget B. This produces only the finite rational samples admitted by the regular rational sequence route.

Next read the Farey neighbor row F and the Stern–Brocot branch row S. These two rows identify the same finite rational bracket: F as an ordered neighbor inspection and S as a branch-address replay. The rational output row R and window row W are consumed through that bracket.

Finally, the continuation row C exposes the resulting finite approximation window to the $\mathsf{RealRationalApproximation}^\uparrow$ consumer. Because the carrier contains neither a best-approximant selector nor a completeness construction for all Cauchy data, the handoff remains a finite rational-density packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FareyApproximation}^\uparrow}$**.**

Constructive story: A $\mathsf{FareyApproximation}^\uparrow$ packet is generated from a displayed RealUp name, RegSeqRat finite readback, denominator and tolerance budget, FareySequence neighbor row, SternBrocot branch replay, rational output row, finite error-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FareyApproximation}^\uparrow}$, seed)

Scope closed: Definition 8081.1, Definition 8081.2, and Definition 8081.3 bind the finite carrier, NameCert obligation surface, and rational-density handoff route for the displayed budget.

Formal verification: FormalStatus(NameCert$_{\text{FareyApproximation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, selected best approximants, global rational-density choice, quotient-real equality, countable choice, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, Farey-neighbor exposure, Stern–Brocot branch replay, rational-window exactness, transport, replay, provenance, and local naming for BEDC.Derived.FareyApproximationUp.

Chapter 8082

A Concrete Naming Certificate for $\mathsf{CauchyUniformCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria and regular real names, the $\mathsf{CauchyUniformCriterion}^{\uparrow}$ packet records the finite route by which one displayed uniform tail threshold serves a family of regular Cauchy reads before a Real-completion consumer asks for a seal. It sits beside the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the existing uniform criterion packet of chapter 5823. The packet names the uniform Cauchy handoff as finite rows; it does not form quotient streams, choose a limit, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8082.1 (Cauchy uniform criterion carrier)**.** *A* $\mathsf{CauchyUniformCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *family of inspected tail windows.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by all displayed windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from those windows and tolerances.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row exposed only after S, D, R, M, F have been read.*
5. *M is the common modulus row naming the uniform threshold discipline.*
6. *F is the finite family-index row recording which displayed windows are covered by the same threshold.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-readback, real-seal, modulus, family, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records* Pkg *provenance over* $\mathsf{CauchyUniformCriterion}^{\uparrow}$, $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient Cauchy class, selected limit, global choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 8082.2 (Cauchy uniform criterion NameCert obligations)**.** *For an accepted* $\mathsf{CauchyUniformCriterion}^{\uparrow}$ *carrier* $U = (S, D, R, E, M, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, finite window-family exposure, dyadic tolerance exactness, regular rational readback, real-seal non-escape, common-modulus visibility, family-index coverage, componentwise transport, displayed replay, package provenance, and local naming for the carrier of Definition 8082.1.*

Proof. Project the carrier. The rows S and F expose the inspected family of windows, and the row M supplies the common threshold discipline used by every displayed member of that family.

The dyadic ledger D is then read before the regular readback row R. The real-seal-facing row E is downstream of both the tolerance ledger and the regular readback, so it cannot be used as an independent limit selector.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no further coordinate, the NameCert surface is exhausted by the listed rows and cannot export quotient streams, selected limits, choice, host real equality, or ambient completeness. □

Theorem 8082.3 (Cauchy uniform criterion Real-completion handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{CauchyUniformCriterion}^{\uparrow}$ *carrier must read the finite route*

$$S, F, M, D, R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed family of StreamName windows, one common modulus, the shared dyadic tolerance ledger, the RegSeqRat readback, and only then the RealUp seal-facing row. The handoff exports no quotient Cauchy class, selected limit, host equality of real names, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the carrier inventory of Definition 8082.1. The family row F and common modulus row M are visible before the tolerance ledger is consumed, so the route records uniformity as displayed finite data.

Next read D and R. These rows are the only tolerance and regular-rational readback surfaces available to the consumer, and the seal-facing row E appears only after them.

Finally replay H, C, P, N. The same transport, continuation, provenance, and local naming rows preserve the finite route, preventing a consumer from replacing it by a quotient completion, selected endpoint, host equality theorem, choice schedule, or ambient completeness assertion. □

Falsifiable prediction: A claimed $\mathsf{CauchyUniformCriterion}^{\uparrow}$ consumer is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal while omitting the displayed StreamName family, common modulus, dyadic tolerance ledger, RegSeqRat readback, transport, replay, provenance, or local NameCert row.

Independence witness: cauchy-criterion-uniform, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyUniformCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the family-index row, common threshold discipline, shared tolerance ledger, regular readback, Real seal-facing row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyUniformCriterion}^{\uparrow}}$.

Constructive story: A CauchyUniformCriterion$^{\uparrow}$ packet is generated from a finite StreamName family, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal-facing row, a common modulus row, a family-index coverage row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyUniformCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 8082.1, Definition 8082.2, and Definition 8082.3 seed the finite uniform Cauchy handoff from displayed windows and a common threshold to RegSeqRat readback and RealUp seal-facing rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyUniformCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite family coverage, common-modulus visibility, dyadic tolerance exactness, regular readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUniformCriterionUp.

Chapter 8083

A Concrete Naming Certificate for MetaCICDecidableTypecheckingHandoff$^{\uparrow}$

The MetaCICDecidableTypecheckingHandoff$^{\uparrow}$ packet realizes the MetaCIC decidability audit map of chapter 9379 as a finite boundary between checked type inference and bounded beta-conversion evidence. It reads the checker budget of chapter 5949, the decidability boundary of chapter 749, the typed frontier of chapter 3616, and the bounded normal-equality checker of chapter 752. The packet keeps `inferTypeCtx` soundness and completeness separate from the conditional closed-normal comparison route; it does not claim unrestricted conversion decidability.

Definition 8083.1 (MetaCIC decidable typechecking handoff carrier)**.** *A* MetaCICDecidableTypecheckingHandoff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\Gamma, T, S, Q, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. Γ *is the represented MetaCIC context row.*
2. T *is the* `inferTypeCtx` *checker row, including the context-indexed input and option-valued output.*
3. S *is the checked soundness and completeness row for the type-inference surface.*
4. Q *is the structural same-term checker row used before any normal comparison is requested.*
5. B *is the bounded normal-equality row, carrying displayed fuel and bounded normal forms.*
6. F *is the finished closed-normal boundary row required before a bounded comparison is promoted to the stronger closed-normal statement.*
7. R *is the refusal row recording that the packet does not export fuel-erased beta-conversion decidability when* F *is absent.*
8. H *records componentwise* hsame *transport for the context, checker, soundness-completeness, same-term, bounded-normal, finished-boundary, refusal, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay through the same checker and conversion-boundary route.*
10. P *records* Pkg *provenance over* MetaCICDecidableTypecheckingHandoff$^{\uparrow}$, BoundedTypecheckingBudget$^{\uparrow}$, MetaCICDecidableTypingFrontier$^{\uparrow}$, BoundedNormalEqualityChecker$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N records the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host typechecker, proof-irrelevant term equality, unrestricted beta-conversion decider, hidden fuel, or external closedness oracle.

Theorem 8083.2 (MetaCIC decidable typechecking handoff obligations)**.** *For every accepted* $\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}$ *carrier* $K = (\Gamma, T, S, Q, B, F, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecid}}$ *surface consists exactly of context admission,* `inferTypeCtx` *exposure, checked soundness-completeness exposure, structural same-term exposure, bounded normal-equality exposure, finished-boundary exposure when available, refusal-boundary retention, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier. The context row Γ, checker row T, and theorem row S form the typechecking side of the packet; they expose the checked `inferTypeCtx` interface without introducing a host typechecker.

The equality side is separate. The row Q supplies structural same-term readback, while B supplies bounded normal-equality data at displayed fuel.

The row F is the boundary required before a bounded comparison can be used as closed-normal endpoint evidence. If F is absent, the refusal row R remains part of the same NameCert route and blocks any unrestricted conversion claim.

Transport H, replay C, provenance P, and local naming N preserve this separation. No remaining coordinate can erase fuel, add a closedness oracle, or turn bounded beta conversion into global decidability. □

Theorem 8083.3 (MetaCIC decidable typechecking closed-normal boundary)**.** *Every consumer of an accepted* $\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}$ *packet factors through the finite route*

$$\Gamma, T, S, Q, B, F, R \longmapsto H, C, P, N.$$

The rows Γ, T, S make checked type inference public; the rows Q, B make structural and bounded normal equality public; the row F is required for a closed-normal endpoint read; and the row R remains visible whenever that endpoint boundary is not supplied. The packet therefore separates checked type inference from conditional bounded beta conversion.

Proof. Start with the typechecking route. The checker row T is read in context Γ, and the row S records the checked soundness and completeness surface for `inferTypeCtx`.

Move to the conversion route. Structural same-term data Q and bounded normal-equality data B are finite checker rows; they do not by themselves state that fuel has reached finished closed normal forms.

The endpoint condition is exactly F. When a consumer has F, it may use the bounded comparison as a closed-normal endpoint read; without F, it must keep the refusal row R in scope.

The replay rows H, C, P, N carry no further oracle. Hence the public handoff is the displayed separation between checked type inference and conditional bounded beta conversion. □

Falsifiable prediction: A claimed $\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}$ consumer is refuted if it exports unrestricted beta-conversion decidability while omitting the finished closed-normal boundary row F and refusal row R.

Independence witness: bounded-typechecking-budget, metacic-decidable-boundary, metacic-decidable-typing-frontier, bounded-normal-equality-checker: no listed sibling is bijective to $\mathsf{MetaCICDecidableTypec}$

because this packet jointly carries the type-inference theorem row, bounded conversion boundary, refusal row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}$ packet is generated from a represented context, inferTypeCtx checker row, checked soundness-completeness row, same-term row, bounded normal-equality row, finished closed-normal boundary row, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8083.1, Definition 8083.2, and Definition 8083.3 seed the finite handoff separating checked type inference from conditional bounded beta conversion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted beta-conversion decidability, fuel-erased normalization, a host typechecker, proof-irrelevant term equality, external closedness decision, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inferTypeCtx soundness-completeness exposure, same-term exactness, bounded normal-equality exactness, finished-boundary retention, refusal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDecidableTypecheckingHandoffUp.

Chapter 8084

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyFilter}^{\uparrow}$

Following Kelley and Bishop–Bridges on Cauchy filters, regular real names, and finite moduli, the $\mathsf{RegulatedCauchyFilter}^{\uparrow}$ packet records the completion-facing filter object whose basis, modulus, regular readback, dyadic tolerance, and Real seal are all visible as BEDC rows. It sits after the filter-limit basis of chapter 3862, the completion surface of chapter 1569, the Cauchy-filter source of chapter 333, the filter-limit handoff of chapter 1558, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet narrows filter completion to displayed finite basis and regular-readback data; it does not assert quotient filters, selected limits, countable choice, host filter equality, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8084.1 (Regulated Cauchy filter carrier)**.** *A* $\mathsf{RegulatedCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, F, Q, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *row exposing the finite basis used by completion and limit consumers.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose accepted reads remain tied to the displayed basis row B.*
3. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-facing row available only through the same displayed basis and source filter.*
4. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary row that records the limit-facing handoff without selecting a limit.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the regulated read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the same windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and modulus ledger controlling the requested precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after B, F, Q, L, W, R, D have been displayed.*
9. *H records componentwise* hsame *transport for basis, source filter, completion row, limit row, windows, readback, tolerance, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a regulated completion request through $B, F, Q, L, W, R, D, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyFilter}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, Q, L, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, ultrafilters, selected limits, countable-choice schedules, host filter equality, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8084.2 (Regulated Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedCauchyFilter}^{\uparrow}$ *carrier $G = (B, F, Q, L, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite basis exposure, Cauchy-filter source admission, completion-row handoff, filter-limit boundary admission, finite stream-window inspection, regular rational read-back, dyadic tolerance regulation, terminal Real-seal ordering, componentwise transport, displayed replay, provenance, and local naming for the rows $B, F, Q, L, W, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 8084.1. The only available rows are $B, F, Q, L, W, R, D, E, H, C, P, N$, so carrier admission exposes finite basis, source filter, completion, limit boundary, window, read-back, tolerance, seal, transport, replay, provenance, and naming data.

The regulated completion route begins with B and F. The filter-limit basis and Cauchy-filter source force a completion-facing request to consume finite basis data before it can inspect Q or L.

Next read the window and regularity rows W, R, D. The StreamName window is finite, the RegSeqRat row reads that window as regular rational data, and the DyadicRatCore row records the displayed tolerance budget for the requested precision.

Finally E, H, C, P, N order and preserve the same route. Since the excluded quotient, selection, choice, host-equality, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8084.3 (Regulated completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegulatedCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FilterLimitBasis}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{CauchyFilterLimit}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Dyac}$$

before terminal Real evidence is read. The route uses the displayed rows $B, F, Q, L, W, R, D, E, H, C, P, N$ of the same packet and cannot import quotient filters, selected limits, countable choice, host filter equality, quotient real equality, or ambient completeness.

Proof. Start with an accepted completion-facing consumer of G. The carrier admits a completion row Q only together with the displayed basis row B and source Cauchy-filter row F.

The limit-facing row L is read on the same route. It records a boundary handoff, but it does not select a limit or identify quotient filters.

The consumer then opens the finite window W, reads it through R, and checks the requested precision through D. Only after those rows are displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed.

Transport, replay, provenance, and local naming are confined to H, C, P, N. Hence every accepted consumer remains inside the displayed finite packet rather than entering host filter equality, quotient real equality, countable choice, selected limits, or ambient completeness. □

Theorem 8084.4 (Regulated Cauchy filter basis terminal seal)**.** *Every accepted terminal* $\mathsf{Real}^{\uparrow}$ *seal read through a* $\mathsf{RegulatedCauchyFilter}^{\uparrow}$ *packet must first display the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *row B, the*

CauchyFilter$^{\uparrow}$ *source row* F*, the* CauchyFilterCompletion$^{\uparrow}$ *row* Q*, the* CauchyFilterLimit$^{\uparrow}$ *boundary row* L*, the* StreamName$^{\uparrow}$ *window* W*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* D*, and the structural rows* H, C, P, N*. The terminal seal* E *is therefore read only as the endpoint of the displayed finite basis route, not as ambient* Real$^{\uparrow}$ *completeness, a selected limit, quotient filter equality, host filter equality, or quotient real equality.*

Proof. Project the carrier of Definition 8084.1. The terminal seal row E appears only in the same packet as $B, F, Q, L, W, R, D, H, C, P, N$.

Use the NameCert obligations of Definition 8084.2. They force carrier admission, finite basis exposure, source-filter admission, completion handoff, limit-boundary admission, finite window inspection, regular readback, dyadic tolerance regulation, transport, replay, provenance, and local naming to be read together.

Finally apply the completion route of Definition 8084.3. That route orders the basis, source filter, completion row, limit boundary, stream window, regular readback, and tolerance ledger before terminal Real evidence is consumed, so the seal cannot be detached from the finite basis route. □

Falsifiable prediction: An accepted RegulatedCauchyFilter$^{\uparrow}$ read is refuted if a completion-facing or Real$^{\uparrow}$ consumer reaches the terminal seal without displaying the FilterLimitBasis row, CauchyFilter row, CauchyFilterCompletion row, CauchyFilterLimit row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{RegulatedCauchyFilter}^{\uparrow}}$.

Constructive story: A RegulatedCauchyFilter$^{\uparrow}$ packet is generated from a FilterLimitBasis row, CauchyFilter source row, CauchyFilterCompletion completion row, CauchyFilterLimit boundary row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegulatedCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8084.1, Definition 8084.2, and Definition 8084.3 seed the finite regulated completion route over FilterLimitBasisUp, CauchyFilterCompletionUp, CauchyFilterUp, CauchyFilterLimitUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegulatedCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, ultrafilters, selected limits, countable choice, host filter equality, quotient real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite basis exposure, source-filter admission, completion-row handoff, filter-limit boundary admission, finite-window inspection, regular readback, dyadic tolerance regulation, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedCauchyFilterUp.

Chapter 8085

A Concrete Naming Certificate for $\mathsf{FiniteOpenBallCover}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis, the $\mathsf{FiniteOpenBallCover}^{\uparrow}$ packet records a finite family of displayed open balls that covers the inspected metric rows by explicit membership witnesses. It sits over the metric source of chapter 57, the real open-ball carrier of chapter 8074, the compact-metric finite-net surface of chapter 129, the total-boundedness route of chapter 127, and the finite epsilon-net witness of chapter 1767. The packet names a finite open-ball cover certificate only; it does not assert open-cover compactness, choose a subcover, prove total boundedness, or import ambient completeness.

Definition 8085.1 (Finite open-ball cover carrier)**.** *A* $\mathsf{FiniteOpenBallCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, \rho, B, I, M, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row whose point and distance classifiers are used by every ball and membership witness.*
2. *ρ is the positive tolerance or radius schedule, read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows before any cover claim is consumed.*
3. *B is the finite closed* $\mathsf{ProbeBundle}$ *of open-ball names. Each entry is a displayed* $\mathsf{OpenBall}^{\uparrow}$ *or* $\mathsf{RealOpenBall}^{\uparrow}$ *row with its centre, radius, and membership comparison ledger.*
4. *I is the index and center-admission ledger, recording that every named ball in B has its center admitted by the metric source row X.*
5. *M is the membership ledger. For each inspected source point it displays a ball of B and a metric-distance comparison witnessing membership in that ball.*
6. *W is the finite readback window that exposes the ball, center, radius, and membership rows to compact-metric, total-bounded, and uniform-continuity consumers.*
7. *H records componentwise* hsame *transport for the source, radius schedule, ball bundle, index ledger, membership ledger, readback window, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cover request through X, ρ, B, I, M, W, H.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenBallCover}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{OpenBall}^{\uparrow}$, $\mathsf{RealOpenBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteOpenBallCover}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, \rho, B, I, M, W, H, C, P, N$. It has no coordinate for an arbitrary open cover, a host compactness theorem, a selected subcover, a countable choice schedule, quotient metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8085.2 (Finite open-ball cover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOpenBallCover}^{\uparrow}$ *carrier $U = (X, \rho, B, I, M, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenBallCover}^{\uparrow}}$ *surface consists exactly of carrier admission, positive radius admission, finite ball-bundle visibility, center admission, membership coverage, finite readback, classifier transport, displayed replay, provenance, and local naming for the rows $X, \rho, B, I, M, W, H, C, P, N$.*

Proof. Project the carrier of Definition 8085.1. The only cover-facing rows are the metric source X, radius schedule ρ, finite ball bundle B, center ledger I, membership ledger M, and readback window W.

The ball rows are visible before coverage is read. Each ball in B exposes a centre and radius through the open-ball rows, while I admits the centre as a point of the same metric source X.

Membership coverage is then the row M. Each inspected point is paired with one displayed ball of B and one metric-distance comparison read through X and the radius schedule ρ.

The structural rows preserve this finite route. Componentwise transport H, replay C, provenance P, and local naming N can only move and name X, ρ, B, I, M, W; they do not add an arbitrary open cover, a selected subcover, compactness, countable choice, quotient metric equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenBallCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteOpenBallCover}^{\uparrow}$ packet is generated from a MetricUp source row, a positive rational or dyadic radius schedule, a finite ProbeBundle of open-ball rows, center admissions, membership witnesses, finite readback windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteOpenBallCover}^{\uparrow}}$, seed)

Scope closed: Definition 8085.1 and Definition 8085.2 seed the finite open-ball cover packet over displayed metric sources, open-ball rows, explicit membership witnesses, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteOpenBallCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, total boundedness, compactness, countable choice, quotient metric equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive radius admission, finite ball-bundle visibility, center admission, membership coverage, finite readback, classifier transport, replay, provenance, and local naming for BEDC.Derived.FiniteOpenBallCoverUp.

Chapter 8086

A Concrete Naming Certificate for $\mathsf{LocatedLowerCut}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedLowerCut}^{\uparrow}$ packet records the lower-cut face of a bounded real family by finite lower-bound ledgers, located approximation windows, and regular-rational readback, before any ambient completeness principle for $\mathsf{Real}^{\uparrow}$ is available. It is the constructive lower-cut companion to the located-infimum route of chapter 1452 and the upper-bound sibling route of chapter 1427, with the bounded-family source supplied by chapter 1449.

Definition 8086.1 (LocatedLowerCut carrier)**.** *A* $\mathsf{LocatedLowerCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, L, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded-family source row, read through the finite family window and the common-bound data of* $\mathsf{BoundedRealFamily}^{\uparrow}$*.*
2. *L is the lower-bound ledger. It stores the finite comparisons certifying that inspected family values lie above the proposed lower surface.*
3. *A is the located approximation ledger. For each displayed precision request, it gives a finite family-side approximation that separates accepted lower reads from rejected stronger lower candidates at that precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that fixes the finite observations of F, L, A admitted by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the windowed rational readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, L, A, W, R have been read.*
7. *H records componentwise* hsame *transport for the bounded-family row, lower-bound ledger, located approximation ledger, window, handoff, real seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay for lower-bound reads, located approximation reads, classifier transport, regular-rational handoff, and real-seal readback.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedLowerCut}^{\uparrow}$, $\mathsf{LocatedInfimum}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{BoundedRealFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedLowerCut}^{\uparrow}}$ *row.*

The carrier has no coordinate for Dedekind completeness, quotient real equality, trichotomy, arbitrary infimum, countable choice, or host completeness.

Theorem 8086.2 (LocatedLowerCut NameCert obligations)**.** *For every accepted* $\mathsf{LocatedLowerCut}^{\uparrow}$ *packet* $K = (F, L, A, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedLowerCut}^{\uparrow}}$ *obligation surface is exhausted by the bounded-family source, lower-bound ledger, located approximation ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming rows. A consumer asking for Dedekind completeness, quotient real equality, trichotomy, arbitrary infimum, countable choice, host completeness, or a standard bridge asks for data outside the packet.*

Proof. Project the carrier of Definition 8086.1. The analytic rows are exactly F, L, A, W, R, E: the family source, lower-bound ledger, located approximation ledger, finite observation window, regular-rational handoff, and real seal.

Read the lower-cut obligations through those rows. The lower-bound part is sourced by L over F, while the located approximation part is sourced by A inside the finite window W. The rows R and E only repack and seal the displayed windowed observations.

The structural rows do not add analytic strength. The transports in H, replays in C, provenance in P, and local name row N preserve the same finite route. Since none of the displayed rows contains Dedekind completeness, quotient equality, trichotomy, arbitrary infimum, countable choice, host completeness, or a standard bridge coordinate, none can be exported by the certificate. □

Theorem 8086.3 (LocatedLowerCut window induction)**.** *For every accepted* $\mathsf{LocatedLowerCut}^{\uparrow}$ *packet* $K = (F, L, A, W, R, E, H, C, P, N)$*, induction over the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *constructs the displayed located approximation ledger* A *from the bounded-family source* F*, lower-bound ledger* L*, handoff row* R*, real seal* E*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. The induction does not use Dedekind completeness, quotient real equality, trichotomy, arbitrary infimum, countable choice, or host completeness.*

Proof. Proceed over the finite window row W. The empty window has no precision request, so no located approximation entry is required.

For a successor window, read the next precision request together with the bounded-family source F and the lower-bound ledger L. The carrier requires the corresponding located approximation entry in A, and the induction hypothesis handles the remaining window entries.

After the window ledger is built, R repacks those entries as regular-rational readback and E seals only that readback. The rows H, C, P, N transport, replay, source, and name the same finite construction. No step asks for a completed Dedekind cut, quotient equality of real names, trichotomy, arbitrary infimum, countable-choice selector, or host-completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLowerCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedLowerCut}^{\uparrow}$ packet is generated from a bounded-family source row, a lower-bound ledger, a located approximation ledger, a StreamName window, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedLowerCut}^{\uparrow}}$, seed)

Scope closed: Definition 8086.1, Definition 8086.2, and Definition 8086.3 bind the finite lower-cut seed over the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedLowerCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient real equality, trichotomy, arbitrary infimum, countable choice, host completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, lower-bound readback, located approximation witnessing, window induction, handoff stability, and real-seal factorization for BEDC.Derived.LocatedLowerCutUp.

Chapter 8087

A Concrete Naming Certificate for $\mathsf{ConfigurationSpace}^{\uparrow}$

Following the classical configuration-space treatment of ordered collision-free point tuples, the $\mathsf{ConfigurationSpace}^{\uparrow}$ packet records the finite inspection surface for n displayed points in an ambient space before any quotient by permutations, compactification, or path-space completion is available. It sits after the ambient open-set discipline of chapter 88, the distance and separation rows of chapter 57, the finite index spine of chapter 112, and the membership provenance of chapter 3668. It also exposes the finite ordered-strand boundary consumed by chapter 305, but it does not claim braid isotopy completeness or unordered configuration quotients.

Definition 8087.1 (ConfigurationSpace carrier)**.** *A* $\mathsf{ConfigurationSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, M, T, U, Q, A, D, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{FinSet}^{\uparrow}$ *index row naming the finite ordered tuple positions.*
2. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row and T is the compatible* $\mathsf{Topology}^{\uparrow}$ *row used for open-neighbourhood reads.*
3. *U is the tuple row, assigning each displayed index in I a point admitted by the ambient carrier.*
4. *Q is the* $\mathsf{SetLike}^{\uparrow}$ *membership row recording that every tuple coordinate belongs to the ambient space and to the displayed configuration window.*
5. *A is the pairwise apartness row: for each distinct pair of indices in I, it displays positive separation or an equivalent collision-refusal witness.*
6. *D is the diagonal-refusal row, recording that no admitted read of U factors through the collision diagonal.*
7. *W is the finite observation-window row exposing the open neighbourhoods and separation thresholds read by downstream consumers.*
8. *H records componentwise* hsame *transport for the finite index row, ambient metric and topology rows, tuple row, membership row, apartness row, diagonal-refusal row, window row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay of the same finite configuration rows.*

10. *P is the* Pkg *provenance over* $\mathsf{ConfigurationSpace}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{BraidGroup}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConfigurationSpace}^{\uparrow}}$ *row.*

No coordinate supplies an unordered quotient, a completed path space, compactification boundary, global section of a covering map, or host equality of point tuples.

Theorem 8087.2 (ConfigurationSpace NameCert obligations)**.** *For every accepted* $\mathsf{ConfigurationSpace}^{\uparrow}$ *carrier* $X = (I, M, T, U, Q, A, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConfigurationSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index row, ambient metric and topology rows, tuple row, set-membership row, pairwise apartness row, collision-diagonal refusal row, observation window, componentwise transport, continuation replay, provenance, and local naming row. Every consumer-facing read must display* I, M, T, U, Q, A, D, W *before the structural rows* H, C, P, N *are consumed.*

Proof. Project the carrier of Definition 8087.1. The finite mathematical surface is exhausted by the displayed rows I, M, T, U, Q, A, D, W; the remaining rows only transport, replay, source, and name that surface.

The index and membership obligations are finite. The row I supplies the tuple positions through $\mathsf{FinSet}^{\uparrow}$, while Q records ambient and window membership through $\mathsf{SetLike}^{\uparrow}$, so no host powerset or infinite family of coordinates is available.

The topology and metric obligations are likewise displayed. The rows M, T, W expose only the neighbourhood and separation data read in the current window, and A, D record the pairwise apartness and diagonal refusal used by the configuration read. Thus the NameCert surface stops at the finite collision-free tuple packet. □

Theorem 8087.3 (ConfigurationSpace collision nonescape)**.** *Every accepted* $\mathsf{ConfigurationSpace}^{\uparrow}$ *read refuses collision escape through the finite route*

$$\mathsf{FinSet}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{SetLike}^{\uparrow} \longrightarrow U, Q, A, D, W \longrightarrow H, C, P, N.$$

In particular, if two displayed indices are distinct in I*, every downstream read of their coordinates must pass through the apartness row* A *and the diagonal-refusal row* D *before it can be replayed as a configuration-space observation.*

Proof. Begin with a read of two displayed tuple positions. Distinctness is visible only through the finite index row I, and the corresponding coordinate membership is visible only through the tuple and membership rows U, Q.

The collision-refusal step is forced by the carrier order. The pairwise apartness row A supplies the separation witness for the two indices, and D records that the read does not factor through the collision diagonal.

Finally replay through W, H, C, P, N. These rows can expose neighbourhoods, transport the same finite evidence, and name the packet, but they cannot add a missing collision witness or replace A, D by host tuple equality. □

Theorem 8087.4 (ConfigurationSpace finite product route)**.** *Every product-facing consumer of an accepted* $\mathsf{ConfigurationSpace}^{\uparrow}$ *packet receives the ordered finite tuple through the* $\mathsf{FinSet}^{\uparrow}$ *index spine and the ambient* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *rows, with* $\mathsf{SetLike}^{\uparrow}$ *membership and pairwise apartness displayed before any* $\mathsf{BraidGroup}^{\uparrow}$ *strand-boundary read is available. The route exports a finite ordered configuration packet and does not claim unordered quotient configurations, braid isotopy completeness, compactification, or path lifting.*

Proof. Read the product request through I and U. The finite index spine determines the ordered tuple slots, and the tuple row supplies the coordinate reads without constructing an ambient host product object.

Next expose the ambient rows. The metric and topology data in M, T and the membership row Q place each coordinate inside the displayed ambient window; the apartness and diagonal-refusal rows A, D then certify collision-free status for every displayed pair.

Only after those rows are visible can a strand-boundary consumer read the packet through the $\mathsf{BraidGroup}^{\uparrow}$ provenance in P. That boundary sees ordered, collision-free strands, not an unordered quotient, a completed braid isotopy theorem, or a compactified configuration space. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConfigurationSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ConfigurationSpace}^{\uparrow}$ packet is generated from a FinSet index row, ambient Metric and Topology rows, finite point tuple row, SetLike membership row, pairwise apartness and collision-refusal rows, observation window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConfigurationSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8087.1, Definition 8087.2, Definition 8087.3, and Definition 8087.4 bind the finite ordered collision-free point tuple seed and its ambient topology, metric, membership, and braid-boundary route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConfigurationSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unordered configuration quotients, compactification, path-space completion, braid isotopy completeness, covering-space lifting, Lean verification, or any standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, pairwise apartness exactness, collision-diagonal refusal, finite product routing, transport, replay, provenance, and local naming for BEDC.Derived.ConfigurationSpaceUp.

Chapter 8088

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompactSet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ packet records located compact subsets through finite-net locator data rather than through open-cover compactness. It sits over the compact metric source of chapter 129, the total-bounded finite-net route of chapter 127, the finite epsilon-net surface of chapter 1767, the metric readback of chapter 57, the regular rational handoff of chapter 301, and the finite stream-window surface of chapter 126. The packet names a located compact-set certificate for downstream finite-net and compact-continuity arguments; it does not assert open-cover compactness, countable choice, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8088.1 (Bishop located compact set carrier)**.** *A* $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (K_0, T, E, M, R, S, L, H, C, P, N)$$

with the following displayed rows:

1. K_0 *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, naming the compact-metric interface inspected by the packet.*
2. T *is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net data row for displayed tolerances, centers, coverage witnesses, and refinement comparisons.*
3. E *is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row recording the finite epsilon-net center list and coverage ledger at each inspected scale.*
4. M *is the* $\mathsf{Met}^{\uparrow}$ *distance-readback row used by locator requests.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for rational distance and tolerance reads.*
6. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which inspected compact-set names and candidate centers are read.*
7. L *is the locatedness locator row, recording finite lower and upper distance windows for the compact subset at each query and displayed tolerance.*
8. H *records componentwise* hsame *transport for the compact metric source, total-bounded data, epsilon-net row, metric readback, regular rational handoff, stream windows, locator row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a located compact-set request through* K_0, T, E, M, R, S, L, H.
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompactSet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompactSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K_0, T, E, M, R, S, L, H, C, P, N$. It has no coordinate for open-cover compactness, countable-choice selection of nets, quotient equality of names, host subset equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8088.2 (Bishop located compact set NameCert obligations)**.** *For every accepted* BishopLocatedCompactS *carrier*

$$K = (K_0, T, E, M, R, S, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCompactSet}^{\uparrow}}$ *surface consists exactly of compact-metric source admission, total-bounded finite-net data, finite epsilon-net row exposure, metric distance readback, regular rational handoff, finite stream-window exposure, locatedness locator admission, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8088.1. The analytic rows are exactly K_0, T, E, M, R, S, L.

The compact metric source is read before finite-net data. The total-bounded row and finite epsilon-net row display centers, tolerances, and coverage ledgers, while M, R, S, L provide distance readback, rational handoff, stream windows, and locatedness locator data.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for open covers, countable-choice net selection, quotient equality, host subset equality, or ambient completeness, these rows exhaust the local NameCert surface. □

Theorem 8088.3 (Bishop located compact set finite-net locator)**.** *Every locatedness read exported by an accepted* $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ *packet factors through the compact-metric row K_0, the total-bounded finite-net row T, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row E, the metric readback row M, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the* $\mathsf{StreamName}^{\uparrow}$ *row S, and the locator row L before it reaches transport, replay, provenance, and local naming. The locator is therefore a finite-net route, not an open-cover compactness theorem or a selected ambient subset.*

Proof. Start with the NameCert obligations. They require the compact-metric source and both finite-net rows before any locator read can be certified.

Read the metric-facing side next. Distance requests pass through M, rational tolerance and comparison data pass through R, and inspected names or center windows pass through S before the locatedness locator row L is available.

Finally transport and replay the same route through H, C, P, N. No coordinate remains for open-cover compactness, countable-choice selection, quotient equality, host subset equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the finite-net locator is the complete exported surface. □

Falsifiable prediction: Any accepted $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ read that reaches locatedness data without displaying the compact metric source, total-bounded finite-net row, finite epsilon-net row, metric readback, RegSeqRat handoff, StreamName windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier.

Independence witness: compactmetric, totallybounded, finite-epsilon-net, metric, regseqrat, streamname: none is bijective with $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ because this packet jointly binds compact-metric source data, total-bounded finite-net ledgers, finite epsilon-net coverage, metric

distance readback, rational handoff, stream windows, locatedness locator data, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompactSet}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ packet is generated from CompactMetric source data, TotallyBounded finite-net rows, FiniteEpsilonNet coverage rows, Metric distance readback, RegSeqRat handoff, StreamName windows, a locatedness locator, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompactSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8088.1, Definition 8088.2, and Definition 8088.3 seed the located compact-set route over compact metric, finite-net, distance, rational-readback, stream-window, locator, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompactSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, quotient equality, host subset equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net locator exactness, metric readback, RegSeqRat handoff, StreamName window exposure, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompact

Chapter 8089

A Concrete Naming Certificate for $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$

The $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ packet realizes the observer-expansion symbolic-shadow route as finite Markov partition, transition, boundary, metric, transport, provenance, and NameCert rows. It sits after the finite symbolic partition surface of chapter 4074, the metric route of chapter 3474, the Gromov boundary of chapter 6470, the visual-boundary packet of chapter 4181, and the shadowing route of chapter 6707. The packet names symbolic shadow data only as finite BEDC rows; it does not import global Anosov dynamics, invariant measures, completed shift spaces, orbit quotients, or measure-theoretic coding.

Definition 8089.1 (Hyperbolic Markov partition carrier)**.** *A* $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, Q, A, B, V, S, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *row, exposing finite atoms, cylinder blocks, adjacency rows, and refinement data.*
2. *Q is the transition ledger, recording allowed symbolic moves between displayed atoms and refusing unlisted moves.*
3. *A is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row used to read expansion and separation budgets.*
4. *B is the* $\mathsf{GromovBoundary}^{\uparrow}$ *boundary row attached to the shadow endpoints.*
5. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *visual-boundary row used for finite visual-shadow readback.*
6. *S is the* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *row, displaying the finite fellow-traveller or shadow route consumed by the symbolic read.*
7. *T is the symbolic-shadow transport row binding partition atoms, transitions, metric budgets, boundary endpoints, visual readbacks, and shadowing data as one route.*
8. *H records componentwise* hsame *transport for partition, transition, metric, boundary, visual-boundary, shadowing, symbolic-shadow transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a symbolic-shadow request through F, Q, A, B, V, S, T, H.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$, $\mathsf{FiniteMarkovPartition}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicShadowing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicMarkovPartition}^{\uparrow}}$ *row.*

The classifier reads only $F, Q, A, B, V, S, T, H, C, P, N$. It has no coordinate for global Anosov dynamics, measure-theoretic coding, invariant measures, completed shift spaces, orbit quotients, or host boundary equality.

Theorem 8089.2 (Hyperbolic Markov partition NameCert obligations)**.** *For every accepted* HyperbolicMarkovParti *carrier*

$$M = (F, Q, A, B, V, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicMarkovPartition}^{\uparrow}}$ *surface consists exactly of finite Markov partition admission, transition-ledger exposure, hyperbolic metric budgeting, Gromov-boundary handoff, visual-boundary readback, shadowing-route admission, symbolic-shadow transport, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8089.1. The visible symbolic rows are F, Q, and the visible hyperbolic rows are A, B, V, S.

The symbolic-shadow transport row T binds those rows before replay. Transport through H and continuation through C preserve the same finite route, while P, N source and name it.

Since the carrier has no coordinate for global Anosov dynamics, measure-theoretic coding, invariant measures, completed shift spaces, orbit quotients, or host boundary equality, the listed rows are exactly the local NameCert obligation surface. □

Theorem 8089.3 (Hyperbolic Markov partition symbolic shadow route)**.** *Every symbolic-shadow read exported by an accepted* $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ *packet factors through the finite Markov partition row F, the transition ledger Q, the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row A, the* $\mathsf{GromovBoundary}^{\uparrow}$ *row B, the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row V, the* $\mathsf{HyperbolicShadowing}^{\uparrow}$ *row S, and the symbolic-shadow transport row T before it reaches H, C, P, N. A read that bypasses the transition ledger is not a read of this carrier.*

Proof. Begin with the finite symbolic side. The row F supplies atoms, cylinder blocks, adjacency rows, and refinements, while Q records the allowed transitions between the displayed atoms.

Read the hyperbolic side next. Metric budgets, boundary endpoints, visual-boundary readbacks, and shadowing routes are carried by A, B, V, S and then bound to the symbolic rows by T.

Finally replay the same data through H, C, P, N. Without Q the symbolic word has no displayed transition ledger, and without A, B, V, S, T it has no hyperbolic shadow route, so any bypass changes the packet rather than exporting an accepted symbolic-shadow read. □

Falsifiable prediction: Any symbolic-shadow read that bypasses the transition ledger Q refutes the $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ carrier.

Independence witness: finite-markov-partition, hyperbolic-metric, gromov-boundary, hyperbolic-visual-boundary, hyperbolic-shadowing: none is bijective with $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ because the packet jointly carries finite Markov partition atoms, a transition ledger, hyperbolic metric budgets, Gromov-boundary endpoints, visual-boundary readback, shadowing data, symbolic-shadow transport, replay, provenance, and local naming.

The Lean follow-up must provide `Nontrivial HyperbolicMarkovPartitionUp` and `FieldFaithful HyperbolicMarkovPartitionUp` instances in `lean4/BEDC/Derived/HyperbolicMarkovPartitionUp/TasteGate`.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicMarkovPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ packet is generated from FiniteMarkovPartition rows, transition ledgers, HyperbolicMetric budgets, GromovBoundary endpoints, HyperbolicVisualBoundary readback, HyperbolicShadowing data, symbolic-shadow transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicMarkovPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8089.1, Definition 8089.2, and Definition 8089.3 seed the finite symbolic-shadow route over Markov partition, transition, metric, boundary, visual-boundary, shadowing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicMarkovPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global Anosov dynamics, measure-theoretic coding, invariant measures, completed shift spaces, orbit quotients, host boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-ledger exactness, metric and boundary handoff, shadowing readback, symbolic-shadow transport, replay, provenance, local naming, and TasteGate instances for BEDC.Derived.HyperbolicMarkovPartitionUp.

Chapter 8090

A Concrete Naming Certificate for $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$ packet records the finite comparison surface by which two regular Cauchy real names are judged equivalent through located windows and dyadic tolerances. It sits after the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Bishop L10 comparison route of chapter 8075. The packet names located equivalence as displayed finite data only; it does not quotient regular names, import host equality, assert trichotomy, choose representatives, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8090.1 (Located regular Cauchy equivalence carrier)**.** *A* $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, R, W_L, W_R, D, C, V, H, A, P, N)$$

with the following displayed rows:

1. *L and R are the left and right regular-name rows read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface;*
2. *W_L and W_R are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected on the two sides;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the located comparison;*
4. *C is the finite comparison row licensed by the Bishop L10 comparison surface;*
5. *V is the located verdict row saying that the inspected windows agree within the displayed dyadic budget;*
6. *H records componentwise* hsame *transport for the two names, windows, tolerance, comparison, verdict, replay, provenance, and local naming rows;*
7. *A records displayed* Cont *replay through the same finite comparison route;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopLTenComparisonPrinciple}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of regular names, host equality of completed reals, trichotomy, selected representatives, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.LocatedRegularCauchyEquivalenceUp`

Theorem 8090.2 (Located regular Cauchy equivalence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$ *carrier* $E = (L, R, W_L, W_R, D, C, V, H, A, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, two regular-name exposures through* L, R*, finite window exposure through* W_L, W_R*, dyadic tolerance exposure through* D*, Bishop L10 comparison exposure through* C*, located verdict exposure through* V*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* A*, provenance through* P*, and local naming through* N*.*

Consequently, an accepted consumer may read equivalence only as the finite located comparison of the two displayed regular Cauchy names. It cannot detach the verdict from the windows and tolerance ledger, convert the verdict into quotient equality, choose a representative, decide trichotomy, or appeal to ambient completion of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 8090.1. The only exposed rows are $L, R, W_L, W_R, D, C, V, H, A, P, N$, so the naming surface begins with two regular-name rows, their inspected windows, and the dyadic budget used by the comparison.

Read the comparison row next. The Bishop L10 comparison route licenses C only after the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data have been inspected, so the verdict V remains a located finite-window statement.

Finally inspect the structural rows. Componentwise transport, displayed replay, provenance, and local naming preserve the same rows and add no quotient equality, selected representative, host equality, trichotomy, countable-choice schedule, selected limit, or ambient completeness coordinate. □

Theorem 8090.3 (Located regular Cauchy equivalence Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$ *verdict must first display the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* L, R*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W_L, W_R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the Bishop L10 comparison row* C*, the located verdict row* V*, and the structural rows* H, A, P, N*. The handoff reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through those rows and does not assert quotient real equality, selected limits, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted located-equivalence packet. By Definition 8090.2, a verdict is accepted only with the two regular names, their finite windows, the dyadic tolerance ledger, and the comparison row still present.

Now read the terminal seal. A $\mathsf{Real}^{\uparrow}$ consumer may use the verdict as a displayed handoff because P ties the packet to the real-name seal and A replays the same finite route.

No excluded object is produced by this route. Since the carrier lacks quotient equality, selected representatives, trichotomy, host equality, selected limits, and ambient completeness rows, the Real handoff remains a finite located comparison rather than a completed equality principle. □

Falsifiable prediction: A located regular Cauchy equivalence read is rejected if the verdict row is accepted without both StreamName windows, both RegSeqRat rows, the DyadicRatCore tolerance ledger, the Bishop L10 comparison row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}$ packet is generated from two regular Cauchy name rows, two finite StreamName windows, a DyadicRatCore tolerance ledger, a Bishop L10 comparison row, a located verdict, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRegularCauchyEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8090.1, Definition 8090.2, and Definition 8090.3 seed the finite located equivalence route over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BishopLTenComparisonPrincipleUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularCauchyEquivalence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality of regular names, host real equality, trichotomy, selected representatives, selected limits, countable choice, ambient RealUp completeness, Lean verification of the theorem surface, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located verdict stability, tolerance-ledger nonescape, Real handoff ordering, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRegularCauchyEquivalenceUp.

Chapter 8091

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ packet records the finite-window induction surface used by regular Cauchy real-name arguments. It sits over the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the located regular-Cauchy sibling of chapter 2376. The packet supplies Real and Completion consumers with a displayed induction route for finite regular-Cauchy windows; it does not assert quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8091.1 (Regular Cauchy window induction carrier)**.** *A* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Q, W_0, D_0, R_0, S, W, D, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the regular Cauchy name row whose observations are accessed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *Q is the requested dyadic tolerance row in* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *W_0, D_0, R_0 are the base window, base dyadic tolerance, and base* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows.*
4. *S is the finite successor schedule listing the displayed one-step window extensions.*
5. *W, D, R are the accumulated window, dyadic-budget, and regular-readback rows produced along the schedule S.*
6. *T is the tail-stability row saying that each accepted successor window preserves the already displayed regular-Cauchy tolerance obligation.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the finite induction route has exposed W, D, R, T.*
8. *H records componentwise* hsame *transport for the name, tolerance request, windows, dyadic budgets, readbacks, tail-stability rows, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a tolerance request through the base row, each successor window, the readback row, the tail-stability row, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowInduction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for an infinite stream quotient, a selected limit, countable choice, host real equality, or an ambient completeness theorem.

Theorem 8091.2 (Regular Cauchy window induction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *carrier*

$$K = (X, Q, W_0, D_0, R_0, S, W, D, R, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowInduction}^{\uparrow}}$ *surface consists exactly of regular-name admission, dyadic tolerance admission, base-window admission, finite successor-window admission, accumulated* $\mathsf{StreamName}^{\uparrow}$ *window replay,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget replay,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback stability, tail-stability preservation, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8091.1. The analytic data are the regular Cauchy name X, the requested tolerance Q, the base rows W_0, D_0, R_0, the successor schedule S, the accumulated rows W, D, R, the tail-stability row T, and the Real seal E.

The base step is displayed by W_0, D_0, R_0. It reads one finite window through the dyadic budget and records the corresponding regular rational readback before any Real-facing seal is available.

The successor schedule S contains only finitely many one-step window extensions. Each extension adds a displayed $\mathsf{StreamName}^{\uparrow}$ observation, transports its dyadic tolerance row through D, and refreshes the $\mathsf{RegSeqRat}^{\uparrow}$ readback in R.

The row T records the induction invariant: after each displayed successor step, the same regular-Cauchy tail obligation remains available for the accumulated window. Finally E, H, C, P, N seal, transport, replay, source, and name exactly that finite route, so the listed rows exhaust the NameCert obligations. □

Theorem 8091.3 (Regular Cauchy finite-tail closure)**.** *Let*

$$K = (X, Q, W_0, D_0, R_0, S, W, D, R, T, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *carrier. Any finite extension of the displayed successor schedule S by accepted one-step* $\mathsf{StreamName}^{\uparrow}$ *observations remains within the same regular-Cauchy tail route, provided each new dyadic budget row transports through the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. The accepted packet therefore supplies a finite induction principle for the displayed tail windows without identifying quotient streams or selecting a limiting real.*

Proof. Proceed by induction on the finite list of displayed successor observations. The base case is W_0, D_0, R_0, which already exposes the requested window, dyadic budget, and readback rows.

For the successor step, append one displayed $\mathsf{StreamName}^{\uparrow}$ observation to the accumulated window row W. The transported dyadic budget row enters D, and the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback enters R.

The tail-stability row T is then replayed for the enlarged finite window. Since T is componentwise transported by H and replayed by C, the accepted route remains the same local NameCert route after the one-step extension.

Because the induction ranges only over the displayed finite schedule, no quotient stream equality, selected limit, countable-choice sequence, host real equality, or ambient completeness principle is introduced. □

Theorem 8091.4 (Regular Cauchy window induction Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyWindowInduction}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may use the displayed finite-window induction route to extend regular-Cauchy tail evidence, but it cannot extract quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Start with a Real or Completion consumer of the accepted packet. The carrier exposes no Real-facing row before the base finite window, dyadic budget, and regular readback rows have been displayed.

The consumer then follows the finite successor schedule S. Each successor step passes through the same window, dyadic-budget, readback, and tail-stability rows, so the induction route remains finite and displayed.

Only after that route has been replayed does the seal row E expose a $\mathsf{Real}^{\uparrow}$ endpoint. The structural rows H, C, P, N preserve this ordering under transport, replay, provenance, and naming.

Thus the Real-facing output is a seal for the displayed finite regular-Cauchy window induction route, not a source of quotient equality, selected limits, countable choice, host real equality, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowInduction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ packet is generated from a regular Cauchy name row, a dyadic tolerance request, base StreamName windows, finite successor window extensions, dyadic budget rows, RegSeqRat readbacks, tail-stability rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8091.1, Definition 8091.2, Definition 8091.3, and Definition 8091.4 seed the finite-window induction route for regular Cauchy names through displayed StreamName windows, dyadic budgets, RegSeqRat readbacks, tail stability, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped treatment asks for checked carrier admission, finite successor-window induction, dyadic budget transport, regular readback stability, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 8092

A Concrete Naming Certificate for $\mathsf{DyadicRealBracketRefinement}^{\uparrow}$

Following Bishop and Bridges on constructive real approximation by nested rational intervals, the $\mathsf{DyadicRealBracketRefinement}^{\uparrow}$ packet records the finite bracket-refinement step by which a displayed real-name read is replaced by a smaller dyadic bracket before any completion-facing consumer reads a terminal Real seal. It sits after the Archimedean window surface of chapter 7489, the dyadic arithmetic core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a local radius-refinement certificate; it is not Real completeness, trichotomy, quotient stream equality, a selected limit, or open-cover compactness.

Definition 8092.1 (DyadicRealBracketRefinement carrier)**.** *A* $\mathsf{DyadicRealBracketRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, S, Q, D, E, E', \rho, \rho', I, J, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row whose finite observation is being bracketed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite requests through which the current bracket may inspect the real-name data.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying endpoint arithmetic, midpoint arithmetic, containment checks, and the displayed halving comparison.*
5. *E and E' are the current and refined dyadic bracket rows; E' keeps the same real-name source and is read only after E, S, Q, D are visible.*
6. *ρ and ρ' are positive radius rows over* $\mathsf{PositiveReal}^{\uparrow}$*, with ρ' displayed as the halved radius before the refined bracket may be exported.*
7. *I is the inclusion row saying that the inspected regular readback stays inside the current bracket E.*
8. *J is the refinement row saying that the refined bracket E' stays inside E and uses the halved radius ρ'.*

9. *H records componentwise* hsame *transport for the Real seal, StreamName window, RegSeqRat readback, dyadic ledger, both brackets, both radius rows, inclusion, refinement, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a bracket request through finite window inspection, regular rational readback, dyadic arithmetic, radius halving, bracket inclusion, bracket refinement, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicRealBracketRefinement}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRealBracketRefinement}^{\uparrow}}$ *row.*

The carrier contains no coordinate for Real completeness, trichotomy, quotient stream equality, selected limits, open-cover compactness, or an ambient host order oracle.

Theorem 8092.2 (DyadicRealBracketRefinement NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRealBracketRefinement}^{\uparrow}$ *carrier*

$$B = (R, S, Q, D, E, E', \rho, \rho', I, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRealBracketRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, finite window exposure, regular readback exposure, dyadic arithmetic exposure, positive-radius exposure, inclusion of the inspected readback in E, halved-radius refinement from ρ to ρ', containment of E' in E, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8092.1. The rows $R, S, Q, D, E, E', \rho, \rho', I, J, H, C, P, N$ are the only rows made public by the packet.

The window and readback obligations are the route through S and Q. A bracket request first opens the finite StreamName window and then reads the regular rational data supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

The dyadic and radius obligations are the route through D, ρ, ρ'. The ledger D records the endpoint, midpoint, containment, and halving checks, while ρ' is exposed as a positive halved radius before the refined bracket is exported.

The inclusion and refinement obligations are the rows I and J. They keep the inspected readback inside E and the refined bracket E' inside E. The structural rows H, C, P, N transport, replay, source, and name exactly this route, so none of the refused endpoints is introduced. □

Theorem 8092.3 (DyadicRealBracketRefinement radius halving)**.** *For every accepted* DyadicRealBracketRefinemen *carrier $B = (R, S, Q, D, E, E', \rho, \rho', I, J, H, C, P, N)$, any consumer that reads the refined bracket row E' must first read the positive radius rows ρ, ρ', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *halving ledger D, and the containment row J. Thus the radius-halving step is a finite dyadic ledger route, not a completeness argument or a selected limiting process.*

Proof. Start from the carrier definition. The refined bracket row E' is not standalone; it is paired with the current bracket E, the two positive radius rows, the dyadic ledger, and the refinement row J.

Use the NameCert obligations of Definition 8092.2. They force radius exposure and dyadic arithmetic exposure before the refined bracket may be consumed.

Finally read J. This row is the displayed containment and halving handoff from E to E'. Since no row stores a limit selector, quotient stream equality, trichotomy proof, or completeness theorem, the halving step remains a finite bracket-refinement route. □

Theorem 8092.4 (DyadicRealBracketRefinement scope)**.** *Every public* DyadicRealBracketRefinement$^{\uparrow}$ *read factors through*

$$R, S, Q, D, E, E', \rho, \rho', I, J, H, C, P, N.$$

It may be consumed by Real and Completion chapters only as a finite bracket-refinement packet over the displayed Real seal, StreamName window, RegSeqRat readback, DyadicRatCore ledger, positive-radius rows, transport, replay, provenance, and local NameCert. It does not assert Real completeness, trichotomy, quotient stream equality, selected limits, open-cover compactness, or host real equality.

Proof. Apply the carrier projection and the NameCert obligation surface. Together they expose the exact finite route from the Real seal through window inspection, regular readback, dyadic ledger checks, bracket inclusion, radius halving, and bracket refinement.

Apply radius halving from Definition 8092.3. The only refinement export is the displayed containment row J after D, ρ, ρ' have been read.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because the carrier has no further coordinate, downstream Real or Completion consumers receive only the finite bracket-refinement packet. □

Closure status for NameCert$_{\text{DyadicRealBracketRefinement}^{\uparrow}}$**.**

Constructive story: A DyadicRealBracketRefinement$^{\uparrow}$ packet is generated from a Real seal, a StreamName window, a RegSeqRat readback, a DyadicRatCore endpoint and halving ledger, current and refined bracket rows, positive radius rows, inclusion and refinement rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicRealBracketRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 8092.1, Definition 8092.2, Definition 8092.3, and Definition 8092.4 seed the finite dyadic bracket-refinement route over RealUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, PositiveRealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicRealBracketRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient stream equality, selected limits, open-cover compactness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite window exposure, regular readback exposure, dyadic halving exposure, positive-radius exposure, inclusion, refinement containment, transport, replay, provenance, and local NameCert field faithfulness for BEDC.Derived.DyadicRealBracketRefinementUp.

Chapter 8093

A Concrete Naming Certificate for $\mathsf{KleeneRealizability}^{\uparrow}$

Following Kleene's realizability interpretation, the $\mathsf{KleeneRealizability}^{\uparrow}$ packet records realizer evidence as displayed finite computation rows before any constructive consumer may use it as proof content. It sits after the bounded computation surface of chapter 199, the lambda-term carrier of chapter 200, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies choice-free L10 reasoning with finite witness readback data; it does not assert countable choice, quotient equality of programs, host evaluation, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8093.1 (Kleene realizability carrier)**.** *A* $\mathsf{KleeneRealizability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (e, I, O, T, R, H, C, P, N)$$

with the following displayed rows:

1. *e is the code/index row, read through the* $\mathsf{Computable}^{\uparrow}$ *bounded computation surface and, when the realizer is term-facing, through the* $\mathsf{LambdaCalc}^{\uparrow}$ *syntax surface.*
2. *I is the input packet row, consisting only of finite argument data such as* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback requests, or already displayed BEDC witnesses.*
3. *O is the output packet row returned by the finite computation trace.*
4. *T is the computation trace row, recording the inspected step sequence from the index e and input packet I to the output packet O.*
5. *R is the realizer-relation row saying that the displayed trace realizes the requested predicate or construction only through e, I, O, T.*
6. *H records componentwise* hsame *transport for the code/index, input packet, output packet, trace, realizer relation, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay from a consumer request through the input packet, computation trace, output packet, and realizer-relation row.*
8. *P is the* Pkg *provenance over* $\mathsf{KleeneRealizability}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{LambdaCalc}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KleeneRealizability}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $e, I, O, T, R, H, C, P, N$. It has no coordinate for host execution, function extensionality, quotient equality of programs, countable choice, a selected infinite stream, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.KleeneRealizabilityUp.KleeneRealizabilityTasteGate single carrier alignment`

Theorem 8093.2 (Kleene realizability finite witness readback)**.** *For every accepted* $\mathsf{KleeneRealizability}^{\uparrow}$ *carrier*

$$K = (e, I, O, T, R, H, C, P, N),$$

any witness read exported to a choice-free L10 consumer factors through the finite code/index row e, the input packet I, the output packet O, the computation trace T, the realizer relation R, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N. The read may consume* $\mathsf{Computable}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows named by P, but it cannot extract a choice function, quotient program equality, host evaluator, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 8093.1. The possible witness data are exactly the code/index row e, the input row I, the output row O, the computation trace T, and the realizer relation R.

The computation row T is finite. It records the inspected route by which e acts on the displayed input packet I and reaches the output packet O, so the realizer relation can be read without invoking host execution.

The structural rows H, C, P, N then transport, replay, source, and name the same finite route. Provenance may cite the bounded computation, lambda-term, stream-window, regular rational, and real-name siblings, but it adds no further selection coordinate.

Thus every exported witness remains a finite NameCert readback. Since no carrier row contains a choice function, quotient program equality, host evaluator, selected limit, or ambient completeness principle, no consumer can recover one from the packet. □

Theorem 8093.3 (Kleene realizability L10 nonescape)**.** *Every L10 read of an accepted* $\mathsf{KleeneRealizability}^{\uparrow}$ *packet preserves the finite witness boundary*

$$\mathsf{Computable}^{\uparrow} \longmapsto \mathsf{LambdaCalc}^{\uparrow} \longmapsto \mathsf{KleeneRealizability}^{\uparrow} \longmapsto (\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}).$$

The route may pass finite stream windows and regular rational readbacks to a real-facing consumer, but it never promotes realizability evidence into countable choice, quotient equality of programs, unrestricted beta-conversion, host-level evaluation, selected limits, or a global real-completeness assertion.

Proof. Begin with the realizer witness readback of Definition 8093.2. It forces every consumer to expose e, I, O, T, R before any real-facing or stream-facing row can be consumed.

If the consumer requests stream or real data, the provenance row routes that request through the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the terminal $\mathsf{Real}^{\uparrow}$ seal. These rows are consumed as displayed packet data, not as selected representatives of an infinite object.

The replay rows H and C preserve only the visible computation trace and its realizer relation. They do not add quotient equality of programs, unrestricted beta-conversion, host execution, or an external choice operator.

Therefore the L10 route remains a finite readback surface for constructive witness data and cannot escape to the global principles excluded by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneRealizability}^{\uparrow}}$.

Constructive story: A $\mathsf{KleeneRealizability}^{\uparrow}$ packet is generated from a code/index row, finite input and output packets, a displayed computation trace, a realizer-relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance over ComputableUp, LambdaCalcUp, StreamNameUp, RegSeqRatUp, and RealUp, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KleeneRealizability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8093.1, Definition 8093.2, and Definition 8093.3 seed the finite realizer-witness route for choice-free L10 reasoning through displayed computation, input, output, trace, relation, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KleeneRealizability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient equality of programs, host execution, unrestricted beta-conversion, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped treatment asks for checked carrier admission, finite trace readback, realizer-relation exactness, L10 nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KleeneRealizabilityUp.

Chapter 8094

A Concrete Naming Certificate for $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ packet records a finite sequence of located compact interval certificates through displayed locator, finite-net, and regular-Cauchy window data. It sits over the compact-interval locator of chapter 1891, the Bishop located compact-set surface of chapter 8088, the finite-window induction route of chapter 8091, the stream-name interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a located and finite-net sequence surface for L10 compact-interval arguments; it does not use open-cover compactness, selected subsequential limits, countable choice, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8094.1 (Located compact interval sequence carrier)**.** *A* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, B, F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval row, listing the compact rational or dyadic interval names inspected by the packet.*
2. *L is the* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *row, recording left and right endpoint locator data, metric-distance requests, and displayed interval containment checks.*
3. *B is the* $\mathsf{BishopLocatedCompactSet}^{\uparrow}$ *row, storing the located compact-set finite-net reading attached to the same interval sequence.*
4. *F is the finite-net ledger, recording the finite centers, coverage witnesses, and refinement comparisons used at each displayed tolerance.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which interval names, locator requests, and net centers are read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint, radius, and distance comparisons.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for denominator budgets, dyadic radii, and interval-overlap inequalities.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, L, B, F, W, R, D have been displayed.*

9. *H records componentwise* hsame *transport for interval rows, locator rows, located compact-set rows, finite-net ledgers, stream windows, regular rational readbacks, dyadic tolerance rows, Real seals, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a compact-interval sequence request through $I, L, B, F, W, R, D, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$, $\mathsf{CompactIntervalLocator}^{\uparrow}$, BishopLocatedCompac RegularCauchyWindowInduction$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, B, F, W, R, D, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected subsequential limits, countable-choice net selection, quotient equality of interval names, trichotomy of endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8094.2 (Located compact interval sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ *carrier*

$$K = (I, L, B, F, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSequence}^{\uparrow}}$ *surface consists exactly of interval-row admission, compact-interval locator exposure, Bishop located compact-set handoff, finite-net ledger coverage,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8094.1. The accepted packet displays the interval row I, locator row L, located compact-set row B, finite-net ledger F, window row W, readback row R, tolerance row D, and Real seal E.

The compactness-facing data are finite. Locator reads pass through L, locatedness reads pass through B, and finite-net coverage passes through F before any Real-facing seal is available.

The regular-Cauchy side is W, R, D. A consumer reads stream windows, rational readback, and dyadic tolerance budgets before using E.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the classifier has no coordinate for open covers, selected subsequential limits, countable-choice schedules, quotient equality, endpoint trichotomy, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 8094.3 (Located compact interval sequence extraction)**.** *Every Real or Completion-facing read exported by an accepted* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ *carrier factors through the finite route*

$$I \longmapsto L, B, F \longmapsto W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus a consumer can extract located interval data, finite-net coverage, regular rational readback, and a terminal $\mathsf{Real}^{\uparrow}$ *seal only by displaying the locator and finite-window schedules that precede them.*

Proof. Start with the obligation theorem. It requires every accepted read to keep $I, L, B, F, W, R, D, E, H, C, P, N$ as one finite packet.

The locator stage reads I, L, B, F. The interval names are exposed before the compact-interval locator and Bishop located compact-set rows, and the finite-net ledger records the centers and coverage witnesses used by the located read.

The finite-window stage reads W, R, D. The $\mathsf{StreamName}^{\uparrow}$ windows expose the inspected names, the $\mathsf{RegSeqRat}^{\uparrow}$ row reads endpoint and radius data, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the tolerance comparisons.

Only after those rows are displayed may E be read as a terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows preserve the same route, so no hidden open-cover theorem, selected limit, countable-choice selection, quotient equality, trichotomy, or ambient completeness principle can replace the displayed extraction path. □

Theorem 8094.4 (Located compact interval sequence ledger)**.** *Every accepted* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ *carrier exposes a single finite ledger joining compact-interval locator data, located compact-set finite nets, regular-Cauchy finite windows, rational readback, and dyadic tolerance comparisons. The ledger is stable under componentwise* hsame *transport and displayed* Cont *replay, and it exports no open-cover compactness, sequential compactness without displayed finite nets, countable choice, quotient equality, trichotomy, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier named in Definition 8094.1. Its finite ledger is precisely the tuple $I, L, B, F, W, R, D, E, H, C$

The compact-interval part is I, L, B, F. These rows bind interval names, locator requests, located compact-set data, and finite-net coverage witnesses into one displayed packet.

The finite-window part is W, R, D, E. These rows record the inspected stream windows, regular rational readbacks, dyadic tolerance comparisons, and the terminal Real seal.

Componentwise transport H and displayed replay C preserve the same rows, while provenance P and local naming N keep the ledger attached to $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$. The excluded principles are absent from the classifier, so the ledger cannot export them. □

Falsifiable prediction: An accepted $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or finite-net compact-interval read without displaying $I, L, B, F, W, R, D, E, H, C, P, N$ refutes the carrier surface.

Independence witness: compactintervallocator, bishoplocatedcompactset, regularcauchywindowinduction, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the joint finite packet containing compact interval locator data, located compact-set finite nets, regular-Cauchy finite windows, rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$ packet is generated from finite interval rows, CompactIntervalLocator rows, BishopLocatedCompactSet finite-net rows, RegularCauchyWindowInduction schedules, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8094.1, Definition 8094.2, Definition 8094.3, and Definition 8094.4 seed the located compact interval sequence route from compact-interval locator data through finite nets, finite windows, rational readback, dyadic tolerance accounting, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness without displayed finite nets, countable choice, quotient equality, trichotomy, selected limits, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-interval locator exactness, located finite-net coverage, finite-window induction, rational readback, dyadic tolerance accounting, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactIntervalSequenceUp.

Chapter 8095

A Concrete Naming Certificate for $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$

Following Bishop–Bridges regular real analysis, the $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ packet records the finite inspection window that turns regular-Cauchy oscillation control into a $\mathsf{Real}^{\uparrow}$-facing seal. It sits after the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name surface of chapter 38, and the oscillation ledger of chapter 6026. The chapter names only the displayed finite-window certificate; it does not claim quotient real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8095.1 (Regular-Cauchy oscillation window carrier)**.** *A* $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (O, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{RegularCauchyOscillation}^{\uparrow}$ *source row recording the accepted oscillation bounds for the inspected regular-Cauchy name.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that names the requested precision and the dyadic comparison cells used by the window.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, exposing the stage range inspected before a readback is available.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same stream window and tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after O, D, S, R are visible.*
6. *H records componentwise* hsame *transport for the oscillation, tolerance, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$O, D, S \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$, $\mathsf{RegularCauchyOscillation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationWindow}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by the displayed rows $O, D, S, R, E, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable choice, host real equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8095.2 (Regular-Cauchy oscillation window NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ *carrier*

$$W = (O, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegularCauchyOscillation}^{\uparrow}$ *source row, dyadic tolerance exposure through* $\mathsf{DyadicRatCore}^{\uparrow}$*, finite window exposure through* $\mathsf{StreamName}^{\uparrow}$*, regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, Real sealing through* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger nonescape for the excluded quotient, selection, choice, host-equality, trichotomy, and ambient-completeness coordinates.*

Proof. Project the carrier of Definition 8095.1. The projection exposes the regular-Cauchy oscillation source, dyadic tolerance row, finite stream window, regular rational readback, Real seal, and structural rows.

The local NameCert obligations are then read row by row. Carrier admission names O; tolerance exposure names D; finite inspection names S; readback names R; and sealing names E.

The remaining rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, the listed obligations exhaust the accepted packet. □

Theorem 8095.3 (Regular-Cauchy oscillation window Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ *packet first reads the oscillation bounds in* $\mathsf{RegularCauchyOscillation}^{\uparrow}$*, the dyadic tolerance row in* $\mathsf{DyadicRatCore}^{\uparrow}$*, the finite window row in* $\mathsf{StreamName}^{\uparrow}$*, and the regular rational readback row in* $\mathsf{RegSeqRat}^{\uparrow}$*. Only after those rows are displayed may the consumer read the* $\mathsf{Real}^{\uparrow}$ *seal. The route is a finite* BHist/Cont/Pkg *certificate and not a quotient-real construction, selected-limit operation, countable-choice schedule, trichotomy argument, or ambient completeness theorem.*

Proof. Start with an accepted carrier $W = (O, D, S, R, E, H, C, P, N)$ and project its displayed rows. The only oscillation source is O, the only tolerance data are in D, and the only window schedule is S.

The regular rational readback is therefore forced through R before the seal row E can be consumed. Componentwise transport through H preserves the row roles, and replay through C preserves the order of the displayed route.

Provenance and local naming through P, N attach the route to the packet without adding analytic content. Since the carrier contains no quotient, selection, choice, trichotomy, host-equality, or ambient-completeness coordinate, the Real-facing seal is reached only through the finite rows named above. □

Falsifiable prediction: Every $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ Real-facing read factors through one $\mathsf{RegularCauchyOscillation}^{\uparrow}$ source row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regularcauchyoscillation, dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries oscillation bounds, dyadic tolerance, finite window inspection, regular readback, Real sealing, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOscillationWindow}^{\uparrow}$ packet is generated from regular-Cauchy oscillation bounds, a DyadicRatCore tolerance row, a StreamName finite window, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8095.1, Definition 8095.2, and Definition 8095.3 seed the finite-window route from regular-Cauchy oscillation control to Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, selected representatives, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite window coverage, regular readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyOscillationWindowU

Chapter 8096

A Concrete Naming Certificate for $\mathsf{EllInftySequenceSpace}^{\uparrow}$

Following Banach and Bourbaki on bounded sequence spaces, the $\mathsf{EllInftySequenceSpace}^{\uparrow}$ packet records the finite NameCert surface for bounded real sequence data inspected through stream schedules, regular rational readback, and a uniform bound ledger. It sits after the terminal real-name seal of chapter 38, the finite stream window interface of chapter 126, the regular rational readback of chapter 301, the bounded sequence carrier of chapter 1415, and the Cauchy sequence-space route of chapter 406. The packet names the classical ℓ^∞ surface as bounded sequence inspection data, not as a quotient Banach space, selected representative, countable-choice family, host supremum, or ambient completeness theorem.

Definition 8096.1 (Ell-infinity sequence-space carrier)**.** *An* $\mathsf{EllInftySequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, W, R, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the inspected sequence coordinates and finite prefixes.*
2. *W is the finite sequence-window ledger consumed from S.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route that turns displayed windows into terminal real-name seals.*
4. *B is the uniform bound ledger, recording the displayed bound used by every inspected coordinate before the corresponding Real seal is exported.*
5. *H records componentwise* hsame *transport for the schedule, windows, readback, bound ledger, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a consumer request through S, W, R, B, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{EllInftySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{EllInftySequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, W, R, B, H, C, P, N. There is no coordinate for quotient sequence equality, selected representatives, countable-choice schedules, host suprema, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8096.2 (Ell-infinity sequence-space NameCert obligations)**.** *For every accepted* $\mathsf{EllInftySequenceSpace}^{\uparrow}$ *carrier* $E = (S, W, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EllInftySequenceSpace}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed schedule, finite windows, regular rational and Real readback, uniform bound ledger, transport, replay, provenance, and local naming rows;*
2. *sequence-window exposure, requiring every consumer to read* S *and* W *before it can inspect coordinate readback;*
3. *Real readback ordering, requiring* R *to remain downstream of the displayed finite windows and regular rational rows;*
4. *uniform bound exactness, requiring* B *to be read before any exported Real seal is treated as part of the bounded sequence-space packet;*
5. *ledger refusal, excluding quotient sequence equality, selected representatives, countable-choice schedules, host suprema, ambient Real completeness, and bridge certificates.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 8096.1. Its displayed rows are exactly S, W, R, B, H, C, P, N, so carrier admission exposes only finite sequence schedules, windows, readback, a uniform bound ledger, transport, replay, provenance, and local naming.

The sequence-window and readback obligations follow the route through S, W, R. A consumer first reads the finite schedule and windows, then the regular rational and Real readback attached to those windows.

The bound row is independent of host suprema. It is a displayed ledger row B tied to the inspected windows and readback, so boundedness is consumed as packet data rather than inferred from an ambient function space.

Finally the structural rows H, C, P, N preserve the same finite packet. Since the carrier has no coordinate for quotient sequence equality, selected representatives, choice schedules, host suprema, ambient completeness, or bridge certificates, the five rows exhaust the local NameCert surface. □

Theorem 8096.3 (Ell-infinity sequence-space uniform bound ledger)**.** *Every consumer read of an accepted* $\mathsf{EllInftySequenceSpace}^{\uparrow}$ *carrier reaches a* $\mathsf{Real}^{\uparrow}$ *seal only after reading the uniform bound ledger* B *for the same displayed schedule and finite window route. In carrier notation, any exported coordinate read factors through*

$$S, W, R, B, H, C, P, N$$

with B *visible before the coordinate is consumed as bounded sequence-space data.*

Proof. Begin with the finite schedule S and the window ledger W. These rows determine which sequence coordinates and prefixes are available to the consumer.

Read the regular rational and Real route R next. The route is attached to those displayed windows, so a terminal Real seal is not a detached representative of a quotient sequence.

Now inspect the uniform bound ledger B. By the NameCert obligations of Definition 8096.2, B is read before the Real seal is exported as bounded sequence-space evidence.

The structural rows H, C, P, N replay the same route and do not add a host supremum or completeness row. Thus every consumer sees the bound ledger before it can use the Real seal as an $\mathsf{EllInftySequenceSpace}^{\uparrow}$ coordinate. □

Theorem 8096.4 (Ell-infinity sequence-space L10 nonescape)**.** *For every accepted* $\mathsf{EllInftySequenceSpace}^{\uparrow}$ *carrier* $E = (S, W, R, B, H, C, P, N)$*, the local ledger refuses quotient sequence equality, selected representatives, countable choice, host suprema, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any request for one of those objects leaves the accepted packet rather than reading the finite bounded sequence-space certificate.*

Proof. Inspect the carrier definition. The only coordinates are the schedule, finite windows, readback route, uniform bound ledger, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations and the uniform bound ledger theorem. They force consumer reads through the displayed finite route and place boundedness in B, not in a host supremum or ambient completeness principle.

Because no carrier row stores quotient sequence equality, a selected representative, a choice schedule, a host supremum, or a Real completeness theorem, no replay through H, C, P, N can export those objects. □

Falsifiable prediction: An accepted $\mathsf{EllInftySequenceSpace}^{\uparrow}$ read is refuted if a consumer reaches bounded sequence-space data without displaying a StreamName schedule, finite sequence windows, RegSeqRat and Real readback, a uniform bound ledger, transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, real, boundedrealsequence, cauchysequencespace: none alone is bijective with $\mathsf{EllInftySequenceSpace}^{\uparrow}$ because the packet jointly carries schedule, finite windows, readback, a uniform bound ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EllInftySequenceSpace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EllInftySequenceSpace}^{\uparrow}$ packet is generated from a StreamName schedule, finite sequence windows, RegSeqRat and Real readback, a uniform bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EllInftySequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8096.1, Definition 8096.2, and Definition 8096.3 seed the finite bounded sequence-space route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EllInftySequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Banach-space equality, selected representatives, countable choice, host suprema, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-window exposure, Real readback ordering, uniform bound exactness, ledger refusal, transport, replay, provenance, and local naming for BEDC.Derived.EllInftySequenceSpaceUp.

Chapter 8097

A Concrete Naming Certificate for $\mathsf{CompactCauchyFilter}^{\uparrow}$

Following Kelley, Bourbaki, and Bishop–Bridges on compactness, filters, and constructive completion, the $\mathsf{CompactCauchyFilter}^{\uparrow}$ packet records the finite compact-generated Cauchy-filter surface used by the Real and Completion route. It sits after the compact metric carrier of chapter 129, the filter-base interface of chapter 1601, the Cauchy-filter surface of chapter 333, the StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names only displayed compact finite-net and Cauchy-filter data; it does not select a limit, quotient filters, use countable choice, import open-cover compactness, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8097.1 (Compact Cauchy filter carrier)**.** *A* $\mathsf{CompactCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, B, Q, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact finite-net row, exposing the finite compact source windows and net witnesses consumed by the filter route.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row, listing the finite filter-base packets and directed-refinement witnesses visible to completion consumers.*
3. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, recording the displayed Cauchy requests over the same base and compact source.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when the compact filter route is inspected by sequential or real-facing consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite windows and tolerance requests.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, bounding every displayed Cauchy request before the terminal seal is read.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after K, B, Q, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the compact-source, filter-base, Cauchy-filter, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a compact filter request through K, B, Q, W, R, D, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactCauchyFilter}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, Q, W, R, D, E, H, C, P, N$. It has no coordinate for a selected limit, quotient filter equality, quotient real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8097.2 (Compact Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{CompactCauchyFilter}^{\uparrow}$ *carrier $F = (K, B, Q, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyFilter}^{\uparrow}}$ *surface consists exactly of compact-source admission, filter-base admission, Cauchy-filter admission, finite-window exposure, regular-rational readback, dyadic tolerance exposure, Real sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8097.1. The analytic rows are precisely K, B, Q, W, R, D, E: compact finite-net data, filter-base data, Cauchy evidence, finite windows, regular-rational readback, dyadic tolerances, and the terminal Real seal.

The structural rows H, C, P, N preserve, replay, source, and name the same displayed data. They do not add a second filter object, a quotient relation, a selected limit, or an ambient compactness theorem.

Thus every local NameCert obligation is one of the displayed rows, and the excluded completion and compactness principles cannot be obligations of the carrier. □

Theorem 8097.3 (Compact Cauchy filter finite-net handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactCauchyFilter}^{\uparrow}$ *carrier reads the compact finite-net row K, the* $\mathsf{FilterBase}^{\uparrow}$ *row B, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row Q, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D before the* $\mathsf{Real}^{\uparrow}$ *seal E is available. The handoff exports compact-generated Cauchy-filter data only through finite BEDC rows, not through selected limits, quotient filter equality, countable choice, open-cover compactness, or ambient Real completeness.*

Proof. Start with the compact source row. The finite-net data in K supply the compact windows that a completion-facing filter consumer may inspect.

Next read the filter route. The base row B exposes the finite directed interface, and the Cauchy row Q records the Cauchy requests only over that displayed base.

The real-facing portion is downstream. The rows W, R, D translate the inspected filter requests into finite windows, regular-rational readback, and dyadic tolerance accounting before E can be read.

Finally use the structural rows H, C, P, N. They preserve and name the same finite handoff, so no selected limit, quotient filter, choice schedule, open-cover compactness proof, or ambient completeness row is available to a consumer. □

Falsifiable prediction: Every accepted $\mathsf{CompactCauchyFilter}^{\uparrow}$ completion read must expose compact finite-net data, a FilterBase row, a CauchyFilter row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a Real-facing consumer may inspect the result.

Independence witness: compactmetric, filterbase, cauchyfilter, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{CompactCauchyFilter}^{\uparrow}$ because this packet jointly

carries compact finite-net data, filter-base refinements, Cauchy-filter requests, finite windows, rational readback, dyadic tolerances, Real sealing, transport, replay, provenance, and local naming as one completion-facing surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchyFilter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8097.1, Definition 8097.2, and Definition 8097.3 seed the compact-generated Cauchy-filter route over CompactMetric, FilterBase, CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient filter equality, quotient real equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net admission, filter-base routing, Cauchy-filter routing, finite-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactCauchyFilterUp.

Constructive story: A $\mathsf{CompactCauchyFilter}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, FilterBase rows, CauchyFilter rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8098

A Concrete Naming Certificate for $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$

Following Hadamard, Morse, and Bowen on geodesic dynamics and symbolic coding for hyperbolic systems, the $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ packet records the finite Markov-partition coding surface used by the hyperbolic observer-expansion vision of chapter 9463. It sits after the Markov partition carrier of chapter 8089, the geodesic-flow carrier of chapter 3856, the subshift carrier of chapter 2107, and the boundary-shadow carrier of chapter 3813. The packet names a finite symbolic coding route only; it does not import invariant measures, completed shift spaces, orbit quotients, global Anosov theorems, or host boundary equality.

Definition 8098.1 (Hyperbolic symbolic coding carrier)**.** *A* $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, G, Q, B, T, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ *row, exposing finite atoms, transition ledgers, and partition refinements.*
2. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, exposing the finite flow window to be coded.*
3. *Q is the* $\mathsf{Subshift}^{\uparrow}$ *code row, recording the finite symbolic word or cylinder data admitted by the Markov transition ledger.*
4. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, binding the coded flow window to displayed boundary-shadow data.*
5. *T is the transport row from Markov partition atoms and geodesic-flow windows to subshift code rows and boundary shadows.*
6. *A is the adjacency and admissibility ledger, recording the exact finite transition checks used before a code row is accepted.*
7. *H records componentwise* hsame *transport for Markov partition, geodesic-flow, subshift-code, boundary-shadow, adjacency, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a coding request through M, G, Q, B, T, A, H.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$, $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{Subshift}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicSymbolicCoding}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, G, Q, B, T, A, H, C, P, N$. It has no coordinate for invariant measures, completed shift spaces, orbit quotients, global Anosov dynamics, measure-theoretic coding, or host boundary equality.

Theorem 8098.2 (Hyperbolic symbolic coding NameCert obligations)**.** *For every accepted* HyperbolicSymbolicCodi *carrier $S = (M, G, Q, B, T, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSymbolicCoding}^{\uparrow}}$ *surface consists exactly of Markov-partition admission, geodesic-flow admission, subshift-code admission, boundary-shadow admission, coding transport, adjacency and admissibility checking, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8098.1. The rows M, G, Q, B supply the finite Markov partition, the finite geodesic-flow window, the symbolic code row, and the boundary-shadow row.

The rows T, A bind those data as a coding route. They record how partition atoms and flow windows are transported to code rows and which finite adjacency checks are required before the code is admitted.

The structural rows H, C, P, N preserve, replay, source, and name the same finite route. Since invariant measures, completed shift spaces, orbit quotients, global Anosov dynamics, measure-theoretic coding, and host boundary equality are absent from the carrier, they cannot be local NameCert obligations. □

Theorem 8098.3 (Hyperbolic symbolic coding flow-partition handoff)**.** *Every coding-facing consumer of an accepted* $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ *packet reads the* $\mathsf{HyperbolicMarkovPartition}^{\uparrow}$ *row M, the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row G, the* $\mathsf{Subshift}^{\uparrow}$ *row Q, and the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row B through the same finite transport and admissibility rows T, A before the coding route reaches replay, provenance, and local naming. A read that bypasses either the Markov transition ledger or the geodesic-flow window is outside this carrier.*

Proof. Begin with the Markov partition row. It supplies the finite atoms, transition ledger, and refinement data that make the code admissible.

Read the flow row next. The geodesic-flow window is the dynamic object being coded, so the symbolic row Q cannot be consumed before G has been displayed.

Now bind the boundary and code rows. The boundary-shadow row B keeps the code attached to the displayed hyperbolic boundary data, while T, A record the finite transport and transition checks from M, G to Q, B.

Finally replay the same route through H, C, P, N. Without the Markov transition ledger or the geodesic-flow window, a consumer has a different object, not an accepted $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ handoff. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ read must expose HyperbolicMarkovPartition, HyperbolicGeodesicFlow, Subshift, HyperbolicBoundaryShadow, transport, adjacency, hsame, Cont, Pkg, and local NameCert rows before a coding-facing consumer may inspect the symbolic route.

Independence witness: hyperbolic-markov-partition, hyperbolic-geodesic-flow, subshift, hyperbolic-boundary-shadow: no listed sibling is bijective to $\mathsf{HyperbolicSymbolicCoding}^{\uparrow}$ because this packet jointly carries Markov atoms, geodesic-flow windows, symbolic code rows, boundary shadows, transport, admissibility, replay, provenance, and local naming as one coding surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSymbolicCoding}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicSymbolicCoding}^{\uparrow}}$, seed)

Scope closed: Definition 8098.1, Definition 8098.2, and Definition 8098.3 seed the finite symbolic coding route over HyperbolicMarkovPartition, HyperbolicGeodesicFlow, Subshift, HyperbolicBoundaryShadow, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicSymbolicCoding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close invariant measures, completed shift spaces, orbit quotients, global Anosov dynamics, measure-theoretic coding, host boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Markov-partition admission, geodesic-flow admission, subshift-code admission, boundary-shadow admission, transport, adjacency checking, replay, provenance, and local naming for BEDC.Derived.HyperbolicSymbolicCodingUp.

Constructive story: A HyperbolicSymbolicCoding$^{\uparrow}$ packet is generated from HyperbolicMarkovPartition rows, HyperbolicGeodesicFlow windows, Subshift code rows, HyperbolicBoundaryShadow rows, transport rows, adjacency ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8099

Tug-of-war game NameCert construction

Following Peres–Schramm–Sheffield–Wilson on tug-of-war games and infinity-harmonic limits, the $\mathsf{TugOfWarGame}^{\uparrow}$ packet records a finite alternating-play game as displayed BEDC data before any metric or PDE-facing consumer may read its value. It sits beside the viscosity-solution boundary of chapter 2177, the metric and compact-metric sources of chapter 57 and chapter 129, the finite real-cover packet of chapter 2377, and the fixed-point discipline around chapter 3529. The packet names finite play, legal moves, strategy/payoff classification, stopping, transport, replay, provenance, and local naming; it does not assert comparison, uniqueness, measurable strategy selection, martingale convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8099.1 (Tug-of-war game carrier)**.** *A* $\mathsf{TugOfWarGame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, A, B, L, S, V, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose inspected positions and distance comparisons constrain every move.*
2. *A and B are the two finite action rows, recording the legal alternatives available to the first and second player at each inspected position.*
3. *L is the legal-move ledger. It records the metric or finite-cover witness that each displayed action remains inside the admitted neighbour set.*
4. *S is the strategy/payoff classifier row. It records only finite response tables, payoff comparisons, and player-role tags visible in the packet.*
5. *V is the value-readback row, exposing the finite dynamic-programming value requested by a viscosity-solution or fixed-point consumer.*
6. *T is the payoff and stopping ledger, recording terminal positions, stopping flags, terminal payoff rows, and the finite horizon at which a replay stops.*
7. *H records componentwise* hsame *transport for the source, action, legal-move, strategy/payoff, value, stopping, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for legal moves: a consumer request must replay X, A, B, L, S, V, T, H before a value row is exported.*

9. *P is the* Pkg *provenance over* $\mathsf{TugOfWarGame}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{ViscositySolution}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{DynamicProgramming}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TugOfWarGame}^{\uparrow}}$ *row.*

The classifier has no coordinate for host strategies, measurable selectors, infinite plays, optional-stopping theorems, quotient value functions, comparison principles, or ambient real completeness.

Definition 8099.2 (Tug-of-war strategy and payoff classifier)**.** *Two accepted* $\mathsf{TugOfWarGame}^{\uparrow}$ *carriers*

$$G = (X, A, B, L, S, V, T, H, C, P, N)$$

and

$$G' = (X', A', B', L', S', V', T', H', C', P', N')$$

are classified together exactly when the stored hsame *rows transport the source positions, both action rows, the legal-move ledger, the strategy/payoff table, the value-readback row, the payoff and stopping ledger, the provenance, and the local name, and when C and C' replay the same legal-move route. The comparison is finite-observation equality of the displayed game packet, not equality of host strategies or of an ambient value function.*

Theorem 8099.3 (Tug-of-war game carrier obligation)**.** *For every accepted* $\mathsf{TugOfWarGame}^{\uparrow}$ *carrier $G = (X, A, B, L, S, V, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TugOfWarGame}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *finite play carrier admission through X, A, B, L;*
2. *strategy/payoff classifier admission through S and Definition 8099.2;*
3. *componentwise* hsame *stability through H;*
4. *displayed* Cont *replay for legal moves through C;*
5. *payoff and stopping ledger admission through T, with value readback through V and provenance/local naming through P, N.*

Proof. Project the carrier of Definition 8099.1. The finite play data are precisely the source row, the two action rows, and the legal-move ledger.

The classifier row S and Definition 8099.2 compare only displayed response tables and payoff comparisons. Componentwise transport H preserves those rows, while C replays the same legal moves before any value row is read.

The remaining game-specific information is the payoff and stopping ledger T, together with value readback, provenance, and local naming. Since no host strategy, infinite play, measurable selector, optional-stopping theorem, quotient value function, comparison principle, or ambient completeness row is stored in G, the five displayed rows exhaust the local NameCert surface. □

Theorem 8099.4 (Tug-of-war game payoff ledger)**.** *Every payoff or value read exported by an accepted* $\mathsf{TugOfWarGame}^{\uparrow}$ *packet factors through*

$$X, A, B, L \longmapsto S \longmapsto V \longmapsto T \longmapsto H, C, P, N.$$

Thus a viscosity-solution, finite-cover, compact-metric, metric, or Banach-contraction neighbour receives only the displayed finite game ledger: legal moves, finite strategy/payoff classification, value readback, stopping data, transport, replay, provenance, and local naming.

Proof. Begin with a payoff-facing consumer. Carrier projection exposes X, A, B, L before any player-role, payoff, or value row is available, so legal moves are certified before they are scored.

Next read the finite classifier and value rows. The row S records the response table and payoff comparison visible at the inspected positions, while V records the value read requested by the downstream consumer.

Finally the stopping ledger T seals the terminal positions and horizon, and H, C, P, N transport, replay, source, and name the same route. The exported read therefore remains a finite game-value ledger and cannot become a host strategy selector, infinite-play value theorem, optional-stopping argument, viscosity comparison theorem, fixed-point existence theorem, quotient equality, or ambient real-completeness claim. □

Falsifiable prediction: Any accepted $\mathsf{TugOfWarGame}^{\uparrow}$ value read that omits the metric or compact-metric source, both action rows, the legal-move ledger, the strategy/payoff classifier, the value-readback row, the payoff and stopping ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row refutes the carrier.

Independence witness: viscosity-solution, compactmetric, metric, finite-real-cover, banach-contraction: none is bijective with $\mathsf{TugOfWarGame}^{\uparrow}$ because this packet jointly binds alternating finite play, two action rows, legal-move replay, payoff classification, stopping data, value readback, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TugOfWarGame}\uparrow}$.

Constructive story: A $\mathsf{TugOfWarGame}^{\uparrow}$ packet is generated from metric or compact-metric position rows, two finite action rows, a legal-move ledger, finite strategy/payoff classification, value readback, a payoff and stopping ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TugOfWarGame}\uparrow}$, seed)

Scope closed: Definition 8099.1, Definition 8099.2, Definition 8099.3, and Definition 8099.4 seed the finite game-value route over legal moves, payoff classification, stopping, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TugOfWarGame}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close viscosity comparison, game-value existence or uniqueness, measurable strategy selection, infinite-play convergence, optional stopping, quotient value equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite play admission, strategy/payoff classifier stability, legal-move replay, payoff and stopping ledger exactness, value-readback nonescape, transport, provenance, and local naming for BEDC.Derived.TugOfWarGameUp.

Chapter 8100

A Concrete Naming Certificate for $\mathsf{LinearBoundedAutomaton}^{\uparrow}$

Following Hopcroft–Ullman and the Chomsky hierarchy presentation of bounded machine computation, the $\mathsf{LinearBoundedAutomaton}^{\uparrow}$ packet names finite accepting computations whose tape cells stay inside the input-length bound. It sits between the pushdown-automaton surface of chapter 6018, the context-sensitive grammar route of chapter 6178, the context-sensitive language carrier of chapter 7749, and the machine-code surface of chapter 280. The packet records bounded tape windows, transition rows, accepting histories, and the accepted-word classifier; it is not an unrestricted machine semantics, a host language set, complement theorem, grammar quotient, or ambient decidability principle.

Definition 8100.1 (Linear bounded automaton carrier)**.** *A* $\mathsf{LinearBoundedAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, \Sigma, \Gamma, W, B, T, H_{\mathrm{run}}, F, M, S, C, P, N)$$

with the following displayed rows:

1. *Q is the finite state row.*
2. *Σ is the finite input-alphabet row and Γ is the finite tape-alphabet row.*
3. *W is the input word row, read through* $\mathsf{List}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *length data.*
4. *B is the linear tape-bound ledger, naming the finite tape interval admitted for the input word.*
5. *T is the finite transition row over states, tape symbols, head moves, and boundary checks.*
6. *H_{run} is the accepting-history ledger, listing each configuration window, applied transition, head position, and tape-bound witness.*
7. *F is the final-state row.*
8. *M records sibling comparison to* $\mathsf{ContextSensitiveGrammar}^{\uparrow}$*,* $\mathsf{ContextSensitiveLanguage}^{\uparrow}$*, and* $\mathsf{PushdownAutomaton}^{\uparrow}$ *consumers.*
9. *S is the accepted-word classifier, available only after $Q, \Sigma, \Gamma, W, B, T, H_{\mathrm{run}}, F, M$ have been replayed.*
10. *C records displayed* Cont *replay from an acceptance query through the finite machine, tape-bound, transition, history, final-state, comparison, classifier, provenance, and naming rows.*

11. *P is the* Pkg *provenance over* LinearBoundedAutomaton$^{\uparrow}$, ContextSensitiveGrammar$^{\uparrow}$, ContextSensitiveLanguage$^{\uparrow}$, PushdownAutomaton$^{\uparrow}$, List$^{\uparrow}$, Nat$^{\uparrow}$, BHist, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unbounded tape, a host language set, quotient configuration equality, selected infinite computation, global complement closure, or a general Turing-machine decision oracle.

Theorem 8100.2 (Linear bounded automaton NameCert obligations)**.** *For every accepted* LinearBoundedAutomato *carrier*

$$A = (Q, \Sigma, \Gamma, W, B, T, H_{\mathrm{run}}, F, M, S, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}$ *surface consists exactly of finite state admission, input and tape alphabet admission, input-word admission, linear tape-bound replay, transition-row admission, accepting-history replay, final-state admission, sibling comparison, accepted-word classification, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No accepted packet exports unrestricted Turing-machine computation, host language equality, quotient configuration equality, complement closure, an infinite accepting trace, or ambient decidability.

Proof. Project the carrier of Definition 8100.1. The rows Q, Σ, Γ, W fix the finite machine alphabet and input word, while B fixes the only tape interval a computation may inspect.

The transition and history rows are downstream. The row T lists the permitted finite moves, and H_{run} records each configuration window together with its transition row and tape-bound witness.

The classifier S is exposed only after the final-state row F and comparison row M have been read. Thus a context-sensitive grammar or language consumer may use the accepted-word route, but it cannot replace the route by a host language object or an unrestricted machine semantics.

Finally C, P, N replay, source, and name the same finite route. Since the excluded objects are not carrier coordinates, the listed obligations exhaust the local NameCert surface. □

Theorem 8100.3 (Linear bounded automaton bounded tape route)**.** *Every accepted-word read exported by a* LinearBoundedAutomaton$^{\uparrow}$ *packet factors through the input row W, the linear tape-bound ledger B, the transition row T, and the accepting-history ledger H_{run} before the final-state row F or accepted-word classifier S is available. The comparison route to* ContextSensitiveGrammar$^{\uparrow}$, ContextSensitiveLanguage$^{\uparrow}$, *and* PushdownAutomaton$^{\uparrow}$ *preserves only that finite bounded-tape computation surface; it exports no unbounded tape semantics, host language set, quotient configuration equality, selected infinite run, complement theorem, or global decidability result.*

Proof. Start with the input. By Definition 8100.1, the word row W and bound ledger B are read before any transition history can be inspected.

Apply Definition 8100.2. The accepting-history row H_{run} lists only transition applications from T whose configuration windows stay within B, so neither the final-state row F nor the classifier S can bypass the bounded-tape ledger.

The comparison row M carries this same finite route to the context-sensitive and pushdown siblings. It does not contain a host language set, quotient configuration equality, or unbounded machine oracle.

Replay and provenance through C, P, N preserve the order of the displayed rows. Hence every accepted-word read remains a bounded-tape computation certificate and no stronger semantic object is exported. □

Falsifiable prediction: Every $\mathsf{LinearBoundedAutomaton}^{\uparrow}$ read factors through finite state, input alphabet, tape alphabet, input word, tape-bound, transition, accepting-history, final-state, sibling comparison, accepted-word classifier, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}$ rows.

Independence witness: pushdownautomaton, contextsensitivegrammar, contextsensitivelanguage: $\mathsf{LinearBoundedAutomaton}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded tape windows, transition rows, accepting histories, and an accepted-word classifier, while the siblings separately record stack computation, noncontracting grammar syntax, or accepted context-sensitive language rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}$.

Constructive story: A $\mathsf{LinearBoundedAutomaton}^{\uparrow}$ packet is generated from finite state rows, input and tape alphabet rows, an input-word row, a linear tape-bound ledger, transition rows, accepting-history rows, a final-state row, sibling comparison, accepted-word classification, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8100.1, Definition 8100.2, and Definition 8100.3 seed the bounded tape computation surface over finite machine rows, input windows, tape bounds, transition histories, sibling comparison, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LinearBoundedAutomaton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted Turing-machine semantics, host language equality, quotient configuration equality, selected infinite runs, complement closure, or global decidability.

Upgrade path: Obligation closure requires checked finite-state admission, alphabet admission, bounded tape replay, transition-row exactness, accepting-history replay, classifier transport, sibling comparison, provenance, and local naming for BEDC.Derived.LinearBoundedAutomatonUp.

Chapter 8101

A Concrete Naming Certificate for $\mathsf{IrrationalReal}^{\uparrow}$

Following Euclid and Bishop–Bridges, the $\mathsf{IrrationalReal}^{\uparrow}$ packet records a Real-and-Completion name whose rational-candidate exclusion route is visible as finite BEDC data. It sits over the rational and integer rows of chapter 37 and chapter 31, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the finite obstruction packet of chapter 1983. The packet names a rational-exclusion interface for Real-facing consumers; it does not assert trichotomy, quotient real equality, selected representatives, a global rational-or-irrational decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8101.1 (IrrationalReal carrier)**.** *An* $\mathsf{IrrationalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, Q, Z, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row, recording the Real-and-Completion seal whose rational-exclusion surface is being named.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *candidate row, listing the inspected rational names that a consumer asks to compare with R.*
3. *Z is the* $\mathsf{Int}^{\uparrow}$ *numerator and denominator row attached to the displayed rational candidates.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite comparison budgets between R and the candidates in Q.*
5. *W is the* $\mathsf{IrrationalWitness}^{\uparrow}$ *obstruction row, recording the finite witness packet that refuses the displayed rational candidate route.*
6. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which rational approximations to the Real seal are exposed.*
7. *E is the terminal rational-exclusion seal, reached only after R, Q, Z, D, W, T have been displayed.*
8. *H records componentwise* hsame *transport for the Real source, rational candidates, integer data, dyadic ledger, witness row, regular rational readback, exclusion seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a Real-and-Completion rational-exclusion request through R, Q, Z, D, W, T, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{IrrationalReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{IrrationalWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IrrationalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, trichotomy, selected representatives, a host rationality decision, or ambient Real completeness.

Theorem 8101.2 (IrrationalReal NameCert obligations)**.** *For every accepted* $\mathsf{IrrationalReal}^{\uparrow}$ *carrier*

$$I = (R, Q, Z, D, W, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IrrationalReal}^{\uparrow}}$ *surface consists exactly of Real-source admission, rational-candidate exposure, integer numerator and denominator readback, dyadic tolerance accounting,* $\mathsf{IrrationalWitness}^{\uparrow}$ *obstruction routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation readback, terminal rational-exclusion sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8101.1. The accepted packet displays the Real source R, rational candidate row Q, integer row Z, dyadic ledger D, witness row W, regular rational readback T, terminal seal E, and structural rows H, C, P, N.

The rational-candidate side is finite. The rows Q and Z expose only the inspected rational names and their integer numerator-denominator data, while D records the dyadic comparison budgets used by those inspections.

The exclusion side is routed through W and T. The witness row supplies the finite obstruction packet of $\mathsf{IrrationalWitness}^{\uparrow}$, and the regular rational readback row supplies only the $\mathsf{RegSeqRat}^{\uparrow}$ observations visible before the terminal seal.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row stores quotient equality, trichotomy, selected representatives, a host rationality decision, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8101.3 (IrrationalReal rational-exclusion route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{IrrationalReal}^{\uparrow}$ *packet must read*

$$\mathsf{Real}^{\uparrow}, \mathsf{Rat}^{\uparrow}, \mathsf{Int}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{IrrationalWitness}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}$$

through the displayed rows $R, Q, Z, D, W, T, H, C, P, N$ before the terminal rational-exclusion seal E is available. The route exports a finite refusal surface for the displayed rational candidates only; it does not export trichotomy, quotient real equality, selected representatives, a global rationality classifier, or Real completeness.

Proof. Begin with the carrier rows of Definition 8101.1. The Real source R is displayed before any rational-exclusion seal, and the candidate data Q, Z are the only rational and integer rows available to the consumer.

Next read the dyadic and regular-rational ledgers. The row D stores the tolerance budgets for candidate comparison, while T stores the $\mathsf{RegSeqRat}^{\uparrow}$ readback used by the Real-facing route.

Use the witness row W. It imports the finite obstruction discipline of chapter 1983 without adding a new coordinate for quotient equality, trichotomy, selected representatives, or a host rationality theorem.

The terminal seal E is therefore downstream of R, Q, Z, D, W, T. Componentwise transport, replay, provenance, and local naming preserve exactly this route, so every Real-and-Completion consumer receives only the displayed rational-exclusion packet. □

Falsifiable prediction: An accepted $\mathsf{IrrationalReal}^{\uparrow}$ consumer that reaches the rational-exclusion seal E without displaying $R, Q, Z, D, W, T, H, C, P, N$ refutes the carrier surface.

Independence witness: real, rat, int, dyadicratcore, regseqrat, irrational-witness: no listed sibling alone supplies the joint Real source, rational candidate, integer numerator-denominator, dyadic tolerance, finite irrational-witness obstruction, regular rational readback, transport, replay, provenance, and local NameCert rows bound in $\mathsf{IrrationalReal}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{IrrationalReal}^{\uparrow}}$.

Constructive story: An $\mathsf{IrrationalReal}^{\uparrow}$ packet is generated from a RealUp source row, Rat candidate rows, Int numerator-denominator rows, a DyadicRatCore tolerance ledger, an IrrationalWitness obstruction row, a RegSeqRat readback row, a terminal rational-exclusion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IrrationalReal}^{\uparrow}}$, seed)

Scope closed: Definition 8101.1, Definition 8101.2, and Definition 8101.3 seed the finite rational-exclusion interface from a Real source through rational candidates, dyadic tolerances, witness obstruction, regular rational readback, and terminal sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IrrationalReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected representatives, a global rationality decision, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, rational-candidate exposure, dyadic tolerance exactness, witness-obstruction routing, regular rational readback, rational-exclusion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IrrationalRealUp.

Chapter 8102

A Concrete Naming Certificate for $\mathsf{StreamDiagonalization}^{\uparrow}$

Following Bishop–Bridges diagonal arguments, the $\mathsf{StreamDiagonalization}^{\uparrow}$ packet records only the finite diagonal-window surface that can be inspected before any countable choice principle, quotient stream equality, or ambient completeness theorem is claimed. It sits over the stream naming layer of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, and the finite diagonal-selection route of chapter 4080. The certificate names a displayed diagonal window over finite stream observations; it does not select an infinite host sequence or assert convergence outside the listed rows.

Definition 8102.1 (Stream diagonalization carrier)**.** *A* $\mathsf{StreamDiagonalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, T, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family, naming the finite stream observations and the visible stream indices inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational values available at the displayed windows.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accounting for dyadic tolerances used when adjacent window readings are compared.*
4. *W is the diagonal-window row, retaining the finite index-window schedule through which the packet reads the stream family.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal over the displayed diagonal readback, not a host limit of the surrounding stream family.*
6. *H records componentwise* hsame *transport for the stream-name windows, regular readback, dyadic tolerances, diagonal window, real seal, provenance, and local naming rows.*
7. *C records the displayed* Cont *replay routes by which consumers read S, R, T, W, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamDiagonalization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamDiagonalization}^{\uparrow}}$ *row.*

The carrier contains no row for countable choice, quotient stream equality, a selected host limit, or ambient completeness of the surrounding sequence space.

Theorem 8102.2 (Stream diagonalization NameCert obligations)**.** *For every accepted* $\mathsf{StreamDiagonalization}^{\uparrow}$ *carrier* $D = (S, R, T, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StreamDiagonalization}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *window exposure, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and diagonal-window row* W *to display the finite stream observations being read;*
2. *diagonal selection, requiring* W *to identify the finite diagonal path before the real seal* E *is consumed;*
3. *regular readback stability, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to preserve the displayed rational readings under the transport row* H*;*
4. *dyadic tolerance control, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* T *to account for every tolerance used by the diagonal-window comparison;*
5. *ledger non-escape, requiring* H, C, P, N *to transport, replay, source, and name the same finite route without exporting any unlisted stream or limit principle.*

No obligation is stored outside $S, R, T, W, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 8102.1. The accepted packet displays exactly the stream-name window family, regular rational readback, dyadic tolerance ledger, diagonal window, real seal, transport, replay, provenance, and local naming rows.

The window and selection obligations are read through S and W. The stream family is inspected only at the displayed finite windows, and the diagonal route is named before any consumer reaches the real seal row E.

The stability and tolerance obligations are read through R and T. Regular rational readback supplies the finite values, while the dyadic ledger accounts for the tolerances used in comparing those values along the diagonal window.

Finally H, C, P, N preserve the same displayed route. Since the carrier has no additional coordinate, the local NameCert surface is exhausted by the listed rows and cannot export a countable-choice selector, quotient stream equality, host limit, or ambient completeness claim. □

Theorem 8102.3 (Stream diagonalization window non-escape)**.** *Every consumer read of an accepted* $\mathsf{StreamDiagonalization}^{\uparrow}$ *packet factors through the finite route*

$$S, R, T, W, E, H, C, P, N.$$

The packet may expose StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a diagonal window, a Real seal, componentwise transport, displayed replay, provenance, and local naming. It exports no countable choice principle, no quotient stream equality, no selected host limit, and no ambient completeness theorem.

Proof. Start with Definition 8102.2. It makes the window family, diagonal route, regular readback, dyadic ledger, real seal, transport, replay, provenance, and local name the complete obligation surface.

A consumer reading the diagonal window must therefore pass through S, W, R, T before E is available. The rows are finite NameCert data, so they do not supply a global selector for all streams or an equality relation on quotient streams.

The support rows H, C, P, N preserve and name the same route. They do not add a host sequence, a chosen limit, or a completeness endpoint, so the visible window cannot escape into any of the excluded claims. □

Falsifiable prediction: Every $\mathsf{StreamDiagonalization}^{\uparrow}$ diagonal-window read factors through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a displayed diagonal window, a Real seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-diagonal-selection, real: $\mathsf{StreamDiagonalization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries stream windows, regular readback, tolerance accounting, diagonal-window routing, real sealing, transport, replay, provenance, and local naming while refusing countable choice and quotient stream equality.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamDiagonalization}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamDiagonalization}^{\uparrow}$ packet is generated from a StreamName window family, RegSeqRat readback, DyadicRatCore tolerance ledger, diagonal-window row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamDiagonalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8102.1, Definition 8102.2, and Definition 8102.3 seed the finite diagonal-window NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamDiagonalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient stream equality, selected host limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, diagonal selection, regular readback stability, dyadic tolerance control, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.StreamDiagonalizationUp.

Chapter 8103

A Concrete Naming Certificate for BackwardDifferentiationFormula$^{\uparrow}$

Following Gear and the classical backward differentiation formula tradition, the BackwardDifferentiationFormula$^{\uparrow}$ packet records an implicit linear multistep update as finite BEDC rows. It sits after the one-step Euler carrier of chapter 6231, the staged Runge–Kutta route of chapter 6764, the finite-difference stencil of chapter 5135, the ODE naming surface of chapter 193, the terminal real seal of chapter 38, the regular rational readback of chapter 301, the finite StreamName window of chapter 126, and the dyadic tolerance row of chapter 291. The packet names a finite implicit step window; it does not claim global ODE existence, convergence order, selected limits, a solved nonlinear equation, or ambient Real$^{\uparrow}$ completeness.

Definition 8103.1 (Backward differentiation formula carrier)**.** *A* BackwardDifferentiationFormula$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, W, C, D, R, S, H, K, P, N)$$

with the following displayed rows:

1. *O is the* Ode$^{\uparrow}$ *source row naming the inspected right-hand side and the finite step request.*
2. *W is the* StreamName$^{\uparrow}$ *history window containing the displayed past solution values used by the multistep stencil.*
3. *C is the coefficient row for the selected backward differentiation stencil, including the current implicit coefficient and the finite list of past coefficients.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row for the step size, residual budget, and finite comparison windows.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for every rational coefficient, history value, and residual budget consumed by the step.*
6. *S is the* Real$^{\uparrow}$ *seal row for the current implicit residual after O, W, C, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the ODE source, history window, coefficient row, tolerance row, readback row, seal row, replay row, provenance row, and local naming row.*
8. *K records displayed* Cont *replay from a finite step request through the same history, coefficient, tolerance, readback, and seal rows.*

9. *P records* Pkg *provenance over* $\mathsf{BackwardDifferentiationFormula}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{RungeKuttaFiniteStep}^{\uparrow}$, $\mathsf{EulerMethodFiniteStep}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N records the local* $\mathsf{NameCert}_{\mathsf{BackwardDifferentiationFormula}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a chosen exact solution, a selected implicit root, global convergence, a host real-completeness theorem, or an infinite time grid.

Lean4 (definition) `BEDC.Derived.BackwardDifferentiationFormulaUp`

Theorem 8103.2 (Backward differentiation formula NameCert obligations)**.** *For every accepted* $\mathsf{BackwardDifferentiationFormula}^{\uparrow}$ *carrier* $B = (O, W, C, D, R, S, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BackwardDifferentiationFo}}$ *surface consists exactly of ODE source admission, finite history-window coverage, backward-stencil coefficient exposure, dyadic step and residual budget exposure, regular rational readback, terminal Real seal admission, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier rows. The ODE source O, history window W, and coefficient row C provide the finite implicit multistep stencil; no solution curve is present outside these rows.

Read the numerical boundary through D, R, S. The dyadic row supplies the step and residual budget, the RegSeqRat row supplies rational readback for the displayed data, and the Real row seals only the displayed residual comparison.

Transport and replay are then forced through H and K. Provenance P and local naming N preserve the same finite packet, so the NameCert obligations are exactly the listed finite rows and not a convergence theorem or selected implicit solve. □

Theorem 8103.3 (Backward differentiation formula window stability)**.** *Let* $B = (O, W, C, D, R, S, H, K, P, N)$ *and* $B' = (O', W', C', D', R', S', H', K', P', N')$ *be accepted* $\mathsf{BackwardDifferentiationFormula}^{\uparrow}$ *carriers. If the source, history, coefficient, tolerance, readback, and seal rows are componentwise transported by the displayed* hsame *data and replayed through the displayed* Cont *routes, then every finite-step consumer sees the same backward-stencil window and the same residual-budget ledger. The stable surface is the finite step packet; it exposes no selected exact solution, host implicit-root choice, infinite grid, global convergence theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project both carriers. A finite-step consumer can inspect only O, W, C, D, R, S before it reaches transport, replay, provenance, and local naming.

Use the displayed hsame data to identify the corresponding ODE source, history window, coefficient row, dyadic budget, regular readback, and Real seal rows. The replay rows K and K' then route the same step request through the transported finite packet.

Finally read P, N and P', N'. These rows name and source the same finite backward-stencil window, so stability holds at the packet boundary and cannot be promoted to a selected solution, an implicit-root oracle, an infinite grid, convergence, or completeness. □

Falsifiable prediction: A claimed $\mathsf{BackwardDifferentiationFormula}^{\uparrow}$ consumer is refuted if it reads an implicit step while omitting the history window W, coefficient row C, dyadic residual budget D, RegSeqRat readback row R, or Real seal row S.

Closure status for $\mathsf{NameCert}_{\mathsf{BackwardDifferentiationFormula}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BackwardDifferentiationFormula}^{\uparrow}$ packet is generated from an ODE source row, finite history window, backward-stencil coefficient row, dyadic step and residual budget, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BackwardDifferentiationFormula}^{\uparrow}}$, seed)

Scope closed: Definition 8103.1, Definition 8103.2, and Definition 8103.3 seed the finite implicit multistep NameCert surface.

Formal verification: FormalStatus(NameCert$_{\text{BackwardDifferentiationFormula}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.BackwardDifferentiationFormulaUp`

Bridge status: none

Not claimed: This chapter does not close global ODE existence, convergence order, selected limits, implicit-root choice, infinite time grids, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window exactness, residual-budget replay, transport, provenance, and local naming for BEDC.Derived.BackwardDifferentiationFormulaUp.

Chapter 8104

A Concrete Naming Certificate for $\mathsf{TypecheckTraceDecider}^{\uparrow}$

The $\mathsf{TypecheckTraceDecider}^{\uparrow}$ packet realizes the MetaCIC decidability audit map of chapter 9379 as finite checker-trace data. It reads `inferTypeCtx`, `checkSameTerm`, `normalizeBounded`, and bounded normal-equality comparison as displayed rows, then keeps refusal and provenance visible before any NameCert consumer can replay the trace. The packet is adjacent to chapter 8083, chapter 3616, chapter 752, and chapter 3889; it does not claim unrestricted conversion decidability or fuel-erased normalization.

Definition 8104.1 (Type-check trace decider carrier)**.** *A* $\mathsf{TypecheckTraceDecider}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, I, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the represented MetaCIC context row consumed by* `inferTypeCtx`*.*
2. *I is the type-check trace row, recording the checker input, option-valued output, and the local soundness-completeness witness surface.*
3. *Q is the same-term row exposed by* `checkSameTerm` *before a typing trace compares represented terms.*
4. *B is the bounded-normal comparison row, carrying* `normalizeBounded` *fuel and the corresponding bounded normal-equality check.*
5. *R is the refusal row recording that the packet does not export unrestricted beta-conversion decidability, host definitional equality, or a hidden normalizer endpoint.*
6. *H records componentwise* hsame *transport for the context, checker trace, same-term row, bounded-normal row, refusal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay through the same checker, comparison, refusal, transport, provenance, and local naming route.*
8. *P records* Pkg *provenance over* $\mathsf{TypecheckTraceDecider}^{\uparrow}$, $\mathsf{MetaCICDecidableTypecheckingHandoff}^{\uparrow}$, $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{TypecheckTraceDecider}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host typechecker, proof-irrelevant term equality, fuel-erased normalization, unrestricted conversion, or an external closedness oracle.

Lean4 (definition) `BEDC.Derived.TypecheckTraceDeciderUp`

Theorem 8104.2 (Type-check trace decider obligations)**.** *For every accepted* $\mathsf{TypecheckTraceDecider}^{\uparrow}$ *carrier* $T = (G, I, Q, B, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TypecheckTraceDecider}^{\uparrow}}$ *surface consists exactly of context admission,* `inferTypeCtx` *trace exposure,* `checkSameTerm` *exposure, bounded-normal comparison exposure, refusal-boundary retention, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier. The rows G and I expose the finite checker trace for `inferTypeCtx`; they do not add a host typechecker.

Read the comparison side through Q and B. Structural same-term data and bounded-normal comparison data are displayed as finite rows with their fuel and outputs still visible.

The refusal row R remains part of the NameCert route whenever a consumer asks for unrestricted conversion, fuel-erased normalization, or external closedness. Transport H, replay C, provenance P, and local naming N preserve the same finite trace and add no hidden oracle. □

Theorem 8104.3 (Type-check trace decider replay boundary)**.** *Every consumer of an accepted* $\mathsf{TypecheckTraceDecider}^{\uparrow}$ *packet factors through*

$$G, I, Q, B, R \longmapsto H, C, P, N.$$

The rows G, I *make the checked typing trace public,* Q *makes structural same-term checking public,* B *makes bounded normal comparison public, and* R *keeps the conversion refusal visible. Thus the packet supplies a finite trace decider surface, not an unrestricted conversion or normalization theorem.*

Proof. Start from a consumer read and project the context and checker trace rows G, I. The typing decision surface is the displayed `inferTypeCtx` trace.

Next project the equality rows. The same-term row Q is structural, while B is bounded by the displayed `normalizeBounded` fuel and comparison output.

Finally route the read through R, H, C, P, N. The refusal row blocks stronger conversion readings, and transport, replay, provenance, and local naming preserve only the finite trace packet. □

Falsifiable prediction: A claimed $\mathsf{TypecheckTraceDecider}^{\uparrow}$ consumer is refuted if it promotes `normalizeBounded` to fuel-erased conversion or omits the refusal row R while reading bounded normal-equality data.

Independence witness: metacic-decidable-typechecking-handoff, metacic-decidable-typing-frontier, bounded-normal-equality-checker, metacic-fuel-trace: no listed sibling is bijective to $\mathsf{TypecheckTraceDecider}^{\uparrow}$ because this packet jointly carries checked typing traces, same-term checks, bounded-normal comparison, refusal, transport, replay, provenance, and local naming.

The Lean follow-up is finite: the carrier declaration should support ‘Nontrivial TypecheckTraceDeciderUp‘ and ‘FieldFaithful TypecheckTraceDeciderUp‘ instances by encoding the rows $G, I, Q, B, R, H, C, P, N$ into a displayed event-flow route.

Closure status for $\mathsf{NameCert}_{\mathsf{TypecheckTraceDecider}^{\uparrow}}$.

Constructive story: A TypecheckTraceDecider$^{\uparrow}$ packet is generated from a represented context, inferTypeCtx trace row, checkSameTerm row, normalizeBounded comparison row, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TypecheckTraceDecider}^{\uparrow}}$, seed)

Scope closed: Definition 8104.1, Definition 8104.2, and Definition 8104.3 seed the finite trace-decider NameCert surface.

Formal verification: FormalStatus(NameCert$_{\text{TypecheckTraceDecider}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.TypecheckTraceDeciderUp`

Bridge status: none

Not claimed: This chapter does not close unrestricted beta-conversion decidability, fuel-erased normalization, host typechecking, proof-irrelevant term equality, external closedness decision, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inferTypeCtx trace exactness, same-term exactness, bounded-normal comparison exactness, refusal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.TypecheckTraceDeciderUp.

Chapter 8105

A Concrete Naming Certificate for $\mathsf{JordanOscillationDecomposition}^{\uparrow}$

Following Jordan's bounded-variation decomposition theorem, the $\mathsf{JordanOscillationDecomposition}^{\uparrow}$ packet records the finite oscillation route by which a bounded-variation real function is read as two monotone variation ledgers before completion-facing consumers inspect any real output. It sits after the bounded-variation surface of chapter 398, the finite oscillation partition row of chapter 3800, the finite oscillation variation handoff of chapter 7175, the Darboux-sum boundary of chapter 1880, and the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback endpoints of chapter 301 and chapter 38. The packet names only displayed partitions, positive and negative variation rows, finite oscillation ledgers, certified numeric readback, structural transport, replay, provenance, and local naming data; it does not assert host function equality, a quotient measure, a selected supremum over all partitions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8105.1 (Jordan oscillation decomposition carrier)**.** *A* $\mathsf{JordanOscillationDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, B, P, O, V_+, V_-, D, R, E, H, C, S, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval and endpoint source row.*
2. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source ledger over the same endpoint row.*
3. *P is the accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, including the displayed cells and refinement witnesses used by the packet.*
4. *O is the finite oscillation ledger, recording the oscillation budget visible on each displayed cell.*
5. *V_+ is the positive-variation row, listing the finite upward adjacent-difference reads along P.*
6. *V_- is the negative-variation row, listing the finite downward adjacent-difference reads along P.*
7. *D is the Darboux-facing comparison row, present only after P, O, V_+, V_- have been displayed.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite variation and oscillation budgets.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint, value, oscillation, and variation reads.*
10. *H records componentwise* hsame *transport for interval, bounded-variation, partition, oscillation, two variation, Darboux, readback, seal, replay, provenance, and naming rows.*

11. *C records displayed* Cont *replay through the same finite rows.*

12. *S is the* Pkg *provenance over* $\mathsf{JordanOscillationDecomposition}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanOscillationDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, B, P, O, V_+, V_-, D, R, E, H, C, S, N$. Its stability predicate is componentwise: refinement of P may split an oscillation cell only by appending finite rows to O, V_+, V_- and by replaying the same D, R, E, H, C, S, N boundary. There is no coordinate for host function equality, quotient measure, external variation supremum, selected refinement family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8105.2 (Jordan oscillation decomposition NameCert obligations)**.** *For every accepted* $\mathsf{JordanOscillationDecomposition}^{\uparrow}$ *carrier $J = (I, B, P, O, V_+, V_-, D, R, E, H, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{JordanOscillati}}$ *surface consists exactly of carrier admission, interval and endpoint exposure through I, bounded-variation source exposure through B, finite oscillation partition exposure through P, oscillation-ledger exactness through O, positive and negative variation-ledger exactness through V_+ and V_-, Darboux-facing comparison through D, regular-rational readback through R, terminal Real seal through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through S, and local naming through N.*

Proof. Project the carrier of Definition 8105.1. The visible rows are precisely the interval, bounded-variation source, finite oscillation partition, oscillation ledger, positive and negative variation ledgers, Darboux comparison, readback, Real seal, transport, replay, provenance, and local name.

The partition row P is the sole route to local oscillation data. Refining a cell replaces one finite oscillation row by finitely many displayed subcell rows and appends the corresponding adjacent-difference entries to V_+ and V_-.

The two variation rows are finite BHist data. Positive variation records upward adjacent differences and negative variation records downward adjacent differences, both read against the same displayed partition and bounded-variation source row B.

The Darboux, readback, and Real rows are downstream. Row D can compare only the displayed finite ledgers, R reads only their certified regular-rational budgets, and E seals only those readback values as Real-facing output. Thus no hidden host equality, quotient measure, selected supremum, choice principle, or ambient completeness coordinate enters the NameCert surface. □

Theorem 8105.3 (Finite-variation scope of Jordan oscillation decomposition)**.** *Every completion-facing or Darboux-facing read of an accepted* $\mathsf{JordanOscillationDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{FiniteOscillationPartition}^{\uparrow} \longmapsto (V_+, V_-) \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes a Jordan-style positive and negative variation decomposition over displayed oscillation partitions, not a host equality between functions, a quotient measure construction, an external total-variation supremum, a choice-selected partition net, or ambient real completeness.

Proof. Begin with the bounded-variation row B and the displayed interval row I. They give the packet a finite source ledger before any decomposition rows are visible.

Next use the oscillation partition row P. Each refinement step is a finite ledger operation: oscillation entries in O are subdivided, and the upward and downward adjacent-difference entries are appended to V_+ and V_-.

The Darboux row D reads only that finite partition and the two variation ledgers. It can compare upper and lower finite sums, but it has no row for quotient equality of functions, measure completion, or a selected family of refinements.

Finally, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback occurs only through R and E. Since those rows are downstream of the certified finite classifiers P, O, V_+, V_-, D, every completion-facing output remains inside the displayed finite-variation scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{JordanOscillationDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanOscillationDecomposition}^{\uparrow}$ packet is generated from a Real interval row, bounded-variation source ledger, finite oscillation partition, oscillation budget rows, positive and negative variation ledgers, Darboux comparison, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanOscillationDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8105.1, Definition 8105.2, and Definition 8105.3 seed the finite Jordan decomposition route from bounded-variation ledgers through oscillation partitions to Real and RegSeqRat readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanOscillationDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function equality, quotient measure construction, external total-variation suprema, selected partition nets, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source exposure, finite oscillation partition exposure, positive and negative variation-ledger exactness, Darboux comparison, regular-rational readback, Real seal exactness, transport, replay, provenance, and local naming for BEDC.Derived.JordanOscillationDecompositionUp.

Chapter 8106

A Concrete Naming Certificate for $\mathsf{BaireOneModulus}^{\uparrow}$

Following Baire's function hierarchy and Bishop–Bridges constructive analysis, the $\mathsf{BaireOneModulus}^{\uparrow}$ packet records finite oscillation-window moduli for displayed Baire-one pointwise-limit names. It sits after the pointwise-limit carrier of chapter 1604, the continuous approximant rows of chapter 130, the finite stream schedules of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the lower-semicontinuous handoff of chapter 1536, and the source prefix context of chapter 334. The packet names a finite oscillation-modulus companion to the Baire-one function ledger; it is not an ambient space of functions, not a selected approximating sequence, not quotient equality of real names, not a full measurability theorem, and not a completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8106.1 (Baire-one modulus carrier)**.** *A* $\mathsf{BaireOneModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, S, O, M, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{BaireSpace}^{\uparrow}$ *source-context row naming the inspected source points and finite prefix information;*
2. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *approximants read by the companion* $\mathsf{BaireOneFunction}^{\uparrow}$ *packet;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite approximant and source windows inspected by the modulus request;*
4. *O records the finite oscillation windows over the selected source and approximant rows;*
5. *M records modulus witnesses that turn each displayed oscillation window into a tolerance-controlled finite read;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerances and approximant-value windows selected by S, O, M;*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after the displayed modulus windows and readbacks have been inspected;*
8. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *handoff row receiving the same finite oscillation-modulus ledger;*

9. *H records componentwise* hsame *transport for the source, approximant, schedule, oscillation, modulus, readback, real-seal, handoff, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from a source-point and tolerance request through X, F, S, O, M, Q, R, L;*
11. *P is the* Pkg *provenance over* $\mathsf{BaireOneModulus}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireOneModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient function space, arbitrary approximant choice, quotient real equality, full measurability, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireOneModulusUp.BaireOneModulusTasteGate single carrier alignment`

Theorem 8106.2 (Baire-one modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireOneModulus}^{\uparrow}$ *carrier*

$$K = (X, F, S, O, M, Q, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireOneModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, continuous approximant admission, pointwise schedule exposure, oscillation-window induction, modulus witness exposure, real-seal nonescape, lower-semicontinuous handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission projects the twelve displayed rows of Definition 8106.1. The source row X, approximant ledger F, and schedule row S are the only entries by which a consumer reaches the Baire-one pointwise-limit surface.

Oscillation-window induction is finite because O is indexed by the displayed schedule. Each step exposes a selected source window and approximant window before the modulus row M may answer the tolerance request.

The readback and value rows Q, R are downstream of the same finite route, and the lower-semicontinuous handoff L consumes only that route. The structural rows H, C, P, N transport, replay, source, and name this packet, so no excluded ambient function-space, choice, quotient, measurability, or completeness principle is available. □

Falsifiable prediction: Every accepted $\mathsf{BaireOneModulus}^{\uparrow}$ read exposes BaireSpace source data, ContinuousMap approximant rows, StreamName schedules, oscillation windows, modulus witnesses, RegSeqRat tolerance readbacks, RealUp seals, LowerSemicontinuous handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any oscillation-modulus consumer receives the packet.

Independence witness: baireonefunction, bairespace, continuousmap, streamname, regseqrat, real, lowersemicontinuous: no listed sibling supplies the full twelve-row packet $K = (X, F, S, O, M, Q, R, L, H, C, P,$ because each sibling contributes only the pointwise-limit ledger, source context, approximant admission, finite schedule, rational readback, real seal, or lower-semicontinuous handoff while this chapter binds oscillation windows and modulus witnesses into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOneModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireOneModulus}^{\uparrow}$ packet is generated from a BaireSpace source row, ContinuousMap approximant rows, StreamName schedules, finite oscillation windows, modulus witnesses, RegSeqRat tolerance readbacks, RealUp seals, a LowerSemicontinuous handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BaireOneModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8106.1 and Definition 8106.2 seed the finite oscillation-modulus companion to the Baire-one pointwise-limit packet over displayed source, approximant, schedule, oscillation, modulus, readback, seal, handoff, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BaireOneModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient function spaces, arbitrary approximant choice, quotient real equality, full measurability, Real$^{\uparrow}$ completeness, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, continuous approximant admission, pointwise schedule exposure, oscillation-window induction, modulus witness exposure, real-seal nonescape, lower-semicontinuous handoff, transport, replay, provenance, and local naming for BEDC.Derived.BaireOneModulusUp.

Chapter 8107

A Concrete Naming Certificate for $\mathsf{FiniteMeasureDensity}^{\uparrow}$

Following Radon–Nikodym finite-density practice in constructive measure theory, the $\mathsf{FiniteMeasureDensity}^{\uparrow}$ packet records the finite-partition case where an absolutely continuous measure is read cellwise against a displayed base measure before any integration or probability consumer inspects a density. It sits after the finite measure surface of chapter 92, the probability package of chapter 184, the distribution ledger of chapter 186, and the regular rational readback route of chapter 301. The packet names only finite partition cells, base-measure rows, absolutely-continuous measure rows, density ratios, certified $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seals, structural transport, replay, provenance, and local naming; it does not claim sigma-completion, quotient measure classes, countable choice, or a general Radon–Nikodym theorem.

Definition 8107.1 (Finite measure density carrier)**.** *A* $\mathsf{FiniteMeasureDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (P, M, A, Z, Q, R, E, H, C, G, N)$$

with the following displayed rows:

1. *P is the finite partition row, listing the cells and refinement witnesses inspected by the density request.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *base-measure row, displaying the base value of each cell of P.*
3. *A is the absolutely-continuous measure row, displaying the second measure on the same cells and recording zero-base cells as zero-target cells.*
4. *Z is the density ledger, with one finite row for each cell carrying the rational ratio or the zero-cell witness used by the packet.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite base, target, and density values listed by M, A, Z.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed cell measures and cellwise densities.*
7. *E is the finite expectation or distribution handoff row consumed by* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{Distribution}^{\uparrow}$ *requests.*
8. *H records componentwise* hsame *transport for partition, measure, absolute-continuity, density, readback, seal, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite density query through P, M, A, Z, Q, R, E, H.*

10. *G is the* Pkg *provenance over* $\mathsf{FiniteMeasureDensity}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMeasureDensity}^{\uparrow}}$ *row.*

The carrier has no coordinate for sigma-algebra completion, quotient measure classes, countable choice, non-finite partitions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8107.2 (Finite measure density NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMeasureDensity}^{\uparrow}$ *carrier*

$$D = (P, M, A, Z, Q, R, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteMeasureDensity}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure through P, base-measure exposure through M, absolutely-continuous measure exposure through A, cellwise density exactness through Z, regular-rational readback through Q, terminal Real sealing through R, probability and distribution handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8107.1. The rows $P, M, A, Z, Q, R, E, H, C, G, N$ are the complete visible packet.

The finite partition row P is the only route to cells. The base-measure row M and absolutely-continuous row A are read on the same cells, so a zero-base cell can only be consumed through the displayed zero-target witness in A.

The density ledger Z is finite and cellwise. Each nonzero base cell carries its displayed ratio, while each zero-base cell carries the corresponding zero-target witness rather than a quotient or completion argument.

The readback and seal rows Q, R are downstream of P, M, A, Z, and the probability or distribution handoff E consumes only that sealed finite route. The structural rows H, C, G, N transport, replay, source, and name the same packet, so no sigma-completion, quotient measure class, countable choice, non-finite partition, or ambient completeness coordinate is available. □

Theorem 8107.3 (Finite Radon–Nikodym handoff)**.** *Every* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, *or integration-facing consumer that reads an accepted* $\mathsf{FiniteMeasureDensity}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{Measure}^{\uparrow} \longmapsto (P, M, A) \longmapsto Z \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route is a Radon–Nikodym density handoff for a displayed finite partition: it exports cellwise base values, absolutely-continuous target values, density rows, certified regular-rational readback, and Real seals, not a sigma-completed measure space, quotient density class, countable-choice selector, or general measure-class derivative.

Proof. Start with the finite partition row P. It fixes the cells on which the base and target measures are compared, so the consumer cannot replace the packet by an unlisted partition or a completed sigma-algebra.

Read the rows M and A cell by cell. Absolute continuity is represented only by the finite zero-cell witnesses in A, and nonzero cells are passed to the density ledger Z as displayed arithmetic data.

Now use the readback route. Row Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ values for the base, target, and density rows, while R seals those displayed values as $\mathsf{Real}^{\uparrow}$ data. The handoff row E passes the same sealed rows to probability and distribution consumers.

Since the carrier has no quotient-density coordinate, no choice schedule over a countable partition, and no sigma-completion row, every accepted Radon–Nikodym-facing read remains the finite cellwise handoff stated above. □

Falsifiable prediction: Every accepted $\mathsf{FiniteMeasureDensity}^{\uparrow}$ read exposes finite partition cells, MeasureUp base rows, absolutely-continuous target rows, density ledger rows, RegSeqRat readback, RealUp seals, probability or distribution handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any density consumer receives the packet.

Independence witness: measure, probspace, distribution, regseqrat, real: no listed sibling supplies the full eleven-row packet $D = (P, M, A, Z, Q, R, E, H, C, G, N)$, because the siblings contribute measure rows, probability packaging, distribution readback, regular rational values, or Real seals separately while this chapter binds them into one finite density NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMeasureDensity}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMeasureDensity}^{\uparrow}$ packet is generated from finite partition cells, base-measure rows, absolutely-continuous target-measure rows, density ledger rows, RegSeqRat readback, RealUp seals, probability or distribution handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMeasureDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8107.1, Definition 8107.2, and Definition 8107.3 seed the finite partition density route from displayed measure rows through RegSeqRat readback and Real seals to probability and distribution consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMeasureDensity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, sigma-completed measure spaces, quotient density classes, countable-choice selectors, non-finite partitions, or a general Radon–Nikodym theorem.

Upgrade path: Obligation closure requires checked carrier admission, finite partition exposure, base-measure exposure, absolute-continuity exposure, density-ledger exactness, regular-rational readback, Real seal exactness, probability and distribution handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMeasureDensityUp.

Chapter 8108

A Concrete Naming Certificate for $\mathsf{SchreierRefinement}^{\uparrow}$

Following Schreier and the Zassenhaus refinement route used in Jordan–Holder group theory, the $\mathsf{SchreierRefinement}^{\uparrow}$ packet records a finite refinement comparison between two displayed normal-series rows. It sits after the group carrier of chapter 41, the subgroup carrier of chapter 80, the normal-subgroup route carried by the subgroup chapter, the quotient-group carrier of chapter 82, and the composition-series consumer of chapter 6694. The packet names the finite row data needed by a refinement consumer; it does not assert Jordan–Holder uniqueness, group classification, a selected refinement outside the displayed rows, or quotient equality of host groups.

Definition 8108.1 (Schreier refinement carrier)**.** *A* $\mathsf{SchreierRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, A, B, I, Q, R, T, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Group}^{\uparrow}$ *source row whose operation, unit, inverse, and classifier are read by both series.*
2. *A and B are the two normal-series rows, each exposing a finite chain of* $\mathsf{Subgroup}^{\uparrow}$ *rows with normality read through the subgroup chapter's normal-subgroup route.*
3. *I is the subgroup-inclusion ledger comparing adjacent entries and cross-intersections in the two displayed rows.*
4. *Q is the* $\mathsf{QuotientGroup}^{\uparrow}$ *factor row attached to the adjacent quotients and intersection quotients exposed by the refinement.*
5. *R records the refinement maps that send each displayed factor row to its paired comparison factor.*
6. *T records componentwise* hsame *transport for the group source, series rows, inclusion ledger, quotient-factor rows, refinement maps, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a normal-series comparison request through G, A, B, I, Q, R, T.*
8. *P is the* Pkg *provenance over* $\mathsf{SchreierRefinement}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{NormalSubgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{JordanHolder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SchreierRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host subgroup lattice, quotient primitive, selected refinement oracle, Jordan–Holder uniqueness, or finite-group classification.

Theorem 8108.2 (Schreier refinement NameCert obligations)**.** *For every accepted* $\mathsf{SchreierRefinement}^{\uparrow}$ *carrier*

$$S = (G, A, B, I, Q, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchreierRefinement}^{\uparrow}}$ *surface consists exactly of group-source admission, two normal-series row admissions, subgroup-inclusion accounting, quotient-factor exposure, refinement-map exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8108.1. The group row G is the only ambient algebraic source, while A and B are the only normal-series rows available to the packet.

The comparison surface is finite. The inclusion ledger I records only displayed subgroup inclusions and intersections, and the quotient row Q records only the adjacent and refinement-factor rows read through $\mathsf{QuotientGroup}^{\uparrow}$.

The refinement maps R are then interpreted over those visible rows. Transport, replay, provenance, and local naming are supplied by T, C, P, N, so the listed obligations exhaust the local certificate surface without adding a quotient primitive, selected refinement oracle, host subgroup lattice, or classification theorem. □

Theorem 8108.3 (Schreier refinement normal-series lattice route)**.** *Every* $\mathsf{JordanHolder}^{\uparrow}$ *or composition-series consumer that reads a* $\mathsf{SchreierRefinement}^{\uparrow}$ *packet must first expose the* $\mathsf{Group}^{\uparrow}$ *source, both* $\mathsf{Subgroup}^{\uparrow}$ *normal-series rows, the normal-subgroup inclusion ledger, the* $\mathsf{QuotientGroup}^{\uparrow}$ *factor rows, the refinement maps, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The route supplies a finite normal-series refinement carrier only; it does not prove Jordan–Holder uniqueness, classify groups, select canonical series, or identify quotient factors by host equality.*

Proof. Begin with the group and series rows. The carrier requires G, A, B before a refinement comparison can be consumed, so every downstream read is already tied to the displayed algebra and subgroup surfaces.

Next read the lattice-facing rows I, Q, R. These rows expose subgroup inclusions, normal-series intersection data, quotient factors, and refinement maps as finite packet coordinates, with quotient-facing data routed through chapter 82.

Finally apply the structural rows T, C, P, N. They transport, replay, source, and name the same packet for $\mathsf{JordanHolder}^{\uparrow}$ consumers, while the refused endpoints remain absent from the carrier. Thus the normal-series lattice route is a displayed finite handoff rather than a uniqueness theorem or classification result. □

Falsifiable prediction: A $\mathsf{SchreierRefinement}^{\uparrow}$ consumer that reaches a refinement-map row without first displaying $G, A, B, I, Q, T, C, P, N$ refutes the carrier surface.

Independence witness: group, subgroup, normal-subgroup, quotientgroup, jordan-holder: no listed sibling alone supplies the joint two-series refinement packet $S = (G, A, B, I, Q, R, T, C, P, N)$, because each sibling contributes only the ambient group rows, subgroup rows, normality route, quotient-factor rows, or composition-series consumer boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{SchreierRefinement}^{\uparrow}}$.

Constructive story: A SchreierRefinement$^{\uparrow}$ packet is generated from a GroupUp source row, two normal-series rows, subgroup-inclusion and intersection rows, quotient-factor rows, refinement maps, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SchreierRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 8108.1, Definition 8108.2, and Definition 8108.3 seed the finite normal-series refinement carrier over displayed group, subgroup, normality, quotient-factor, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{SchreierRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full Jordan–Holder uniqueness, group classification, selected canonical refinements, quotient primitive construction, Lean verification, bridge checking, or host quotient equality.

Upgrade path: Obligation closure asks for checked carrier admission, two normal-series row admissions, subgroup-inclusion accounting, quotient-factor exposure, refinement-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.SchreierRefinementUp.

Chapter 8109

A Concrete Naming Certificate for ConstructiveClosedIntervalLocator$^{\uparrow}$

Following Bishop and Bridges on located real intervals, the ConstructiveClosedIntervalLocator$^{\uparrow}$ packet records closed-interval location as finite endpoint, enclosure, membership, and completion-handoff data. It refines the located interval carrier of chapter 394, consumes rational locator cells from chapter 4202, reads dyadic tolerance arithmetic from chapter 291, takes finite observation windows through chapter 301, and seals the endpoint route at chapter 38. The packet is an L10 Real and Completion locator surface; it is not an appeal to order trichotomy, quotient completion, chosen endpoint representatives, or ambient closed-interval compactness.

Definition 8109.1 (Constructive closed interval locator carrier)**.** *A* ConstructiveClosedIntervalLocator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L, U, A, D, M, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint admission row, read as a located real endpoint already named in the surrounding interval history.*
2. *U is the upper endpoint admission row, read as the corresponding located real endpoint in the same history.*
3. *A is the endpoint-order admission row, recording the finite evidence that the displayed lower endpoint is admitted before the displayed upper endpoint for this closed-interval read.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *enclosure ledger of dyadic cells and radii used to shrink rational endpoint windows.*
5. *M is the located membership readback row, recording which rational-interval locator query is accepted, rejected outside the displayed enclosure, or deferred to a smaller dyadic cell.*
6. *S is the finite* StreamName$^{\uparrow}$ *observation surface for endpoint and membership windows.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those finite windows and dyadic tolerance cells.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row for the closed-interval endpoint pair and the accepted membership read.*
9. *H records componentwise* hsame *transport for endpoint admission, dyadic enclosure, membership readback, finite windows, regular rational readback, and terminal seal.*

10. *K records displayed* Cont *replay from endpoint windows and locator cells to dyadic enclosure, located membership, regular readback, and terminal sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}}$ *row naming exactly this finite closed-interval locator packet.*

The classifier compares packets only through the displayed endpoint admission, dyadic enclosure, membership readback, regular-rational handoff, real seal, transport, replay, provenance, and local naming rows. It has no coordinate for trichotomy on real endpoints, quotient real equality, selected representatives, countable choice, host compactness, or an ambient closed-interval object.

Theorem 8109.2 (Constructive closed interval locator NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}$ *carrier*

$$C = (L, U, A, D, M, S, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *endpoint admission, requiring the lower and upper endpoint rows L, U and the endpoint-order admission row A to be displayed before any locator read is exported;*

2. *dyadic enclosure, requiring every closed-interval refinement to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S;*

3. *located membership readback, requiring every accepted membership read to be one of the rational locator reads in M, interpreted through the same D, S, R route;*

4. *Real-seal handoff, requiring any endpoint-pair or membership consumer to reach the terminal* $\mathsf{Real}^{\uparrow}$ *row E only after L, U, A, D, M, S, R have been displayed;*

5. *nonescape from trichotomy, choice, and quotient equality, requiring every consumer to remain inside $L, U, A, D, M, S, R, E, H, K, P, N$ and forbidding real-order trichotomy, quotient completion, host equality of real names, countable choice, selected endpoint representatives, and ambient compactness.*

These rows are exactly the NameCert surface for constructive closed-interval location over the L10 Real and Completion route.

Proof. Project the carrier of Definition 8109.1. Endpoint admission is supplied by the rows L, U, A, so the packet exposes a lower endpoint, an upper endpoint, and their displayed finite ordering admission before any locator read can be consumed.

The enclosure and membership rows are read next. The dyadic ledger D records the finite tolerances and cells, while M records the rational locator answers accepted at those cells. The finite windows S and regular-rational row R are the only route from these cells to endpoint and membership observations.

The handoff to real names is terminal. A consumer reaches E through the displayed continuation route K and the package provenance P, with componentwise transport supplied by H and local naming by N.

Since no coordinate of the carrier stores a trichotomy oracle, quotient completion, selected endpoint representative, countable-choice functional, host equality of real names, or ambient compactness theorem, the five displayed rows exhaust the obligation surface. □

Theorem 8109.3 (Constructive closed interval locator Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}$ *packet must read the route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{RationalIntervalLocator}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows $L, U, A, D, M, S, R, E, H, K, P, N$*. The handoff exports a finite closed-interval locator readback, not an ambient real interval, a quotient completion, a trichotomy decision on real endpoints, a countable-choice selector, or a chosen completed endpoint pair.*

Proof. Start with the interval surface. The endpoint rows L, U and the admission row A place the closed-interval read inside the surrounding $\mathsf{LocatedInterval}^{\uparrow}$ history, while the rational locator row M records the finite membership question being answered.

Then read the completion-facing rows in order. The dyadic ledger D and finite windows S determine the inspected tolerance cells, the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies regular rational readback, and only then may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed.

Transport, replay, provenance, and local naming are structural: H, K, P, N preserve and expose the same finite route. Because the route has no additional coordinate, the consumer receives exactly the displayed closed-interval locator handoff and no ambient order, quotient, choice, or compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}$ packet is generated from endpoint admission, dyadic enclosure, located membership readback, StreamName windows, RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8109.1, Definition 8109.2, and Definition 8109.3 seed the L10 closed-interval locator route over endpoint admission, dyadic enclosure, membership readback, regular-rational handoff, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveClosedIntervalLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim real-order trichotomy, quotient completion, selected endpoint representatives, countable choice, host equality of real names, ambient compactness, public closure, or a standard bridge.

Upgrade path: Formalization can encode BEDC.Derived.ConstructiveClosedIntervalLocatorUp as carrier admission, dyadic enclosure, located membership readback, Real-seal handoff, nonescape, transport, replay, provenance, and local naming obligations.

Chapter 8110

A Concrete Naming Certificate for $\mathsf{CantorBendixsonDerivative}^{\uparrow}$

Following Cantor and Bendixson, the $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ packet records one finite derivative step for a displayed closed real-set row before any rank, located, or decomposition-facing consumer reads it. It sits between the broader $\mathsf{CantorBendixson}^{\uparrow}$ packet of chapter 2024, the located derivative route of chapter 6393, and the finite rank budget of chapter 7197. The carrier also cites the closed-set boundary of chapter 4229, the stream-name windows of chapter 126, the regular rational readback of chapter 301, the dyadic comparison core of chapter 291, and the real-name seal of chapter 38. It names only a finite derivative-step operation on displayed data; it does not claim a transfinite rank, a full scattered-perfect decomposition, a selected enumeration of isolated points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8110.1 (Cantor–Bendixson derivative carrier)**.** *A* $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, A, I, F', W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the input* $\mathsf{ClosedSet}^{\uparrow}$ *row, recording the closed real-set observations whose derivative is being inspected.*
2. *A is the finite accumulation-test row, listing the displayed neighborhood and non-isolation tests used to decide whether an inspected point remains in the derivative row.*
3. *I is the isolated-rejection row, recording only displayed separating intervals, dyadic radii, and membership failures that remove a point from the next derivative row.*
4. *F' is the output* $\mathsf{ClosedSet}^{\uparrow}$ *row, read as the derivative-step closed set produced from F, A, I.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the input row, accumulation tests, isolated rejections, and output row are exposed to finite consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and stage-local derivative observations.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for the radii, interval separations, containment checks, and neighborhood comparisons used by F, A, I, F', W, R.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary for the displayed real-name reads.*
9. *H records componentwise* hsame *transport for F, A, I, F', W, R, Q, E and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of every derivative-step request through*

$$F, A, I \longmapsto F' \longmapsto W, R, Q, E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CantorBendixsonDerivative}^{\uparrow}$, $\mathsf{CantorBendixson}^{\uparrow}$, $\mathsf{LocatedCantorBendixson}^{\uparrow}$, $\mathsf{CantorBendixsonRank}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBendixsonDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for transfinite derivative chains, selected isolated-point enumerations, quotient closed-set equality, countable choice, host closed-set equality, or ambient completeness outside the displayed real-name route.

Theorem 8110.2 (Cantor–Bendixson derivative NameCert obligations)**.** *For every accepted* CantorBendixsonDeriva *carrier*

$$D = (F, A, I, F', W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorBendixsonDerivative}^{\uparrow}}$ *surface is exhausted by the following finite obligation rows:*

1. *carrier admission for the displayed input closed set, accumulation tests, isolated-rejection row, output closed set, StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, Real seal, transport, replay, provenance, and local naming rows;*
2. *derivative-step classifier stability, requiring componentwise* hsame *transport to preserve the same input row, tested neighborhoods, rejected isolated rows, and output row;*
3. *isolated-rejection exactness, requiring each removed point to carry a displayed separating interval, dyadic radius, and membership check inside I and Q;*
4. *output closed-set handoff, requiring F' to be downstream of F, A, I before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer reads it;*
5. *structural nonescape, requiring W, R, Q, E, H, C, P, N to export no transfinite rank, full decomposition, hidden selector, quotient equality, countable-choice schedule, or ambient closed-set theorem.*

No derivative-step obligation is stored outside these rows.

Proof. Project the carrier of Definition 8110.1. The packet displays exactly the input closed-set row, accumulation tests, isolated-rejection row, output closed-set row, observation windows, rational readback, dyadic comparison ledger, Real seal, and structural rows.

The derivative work is a single finite step. Row A records the inspected neighborhoods, while I records the displayed isolating data that prevents a rejected point from remaining in F'.

The output row is therefore read only after the input and rejection rows have been displayed. The windows W, readback R, dyadic ledger Q, and Real seal E expose that same step to finite consumers without adding a further derivative stage.

Finally H, C, P, N transport, replay, source, and name those rows. Since the carrier has no coordinate for transfinite chains, selected enumerations, quotient closed-set equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8110.3 (Cantor–Bendixson derivative closed-set handoff)**.** *Every accepted* $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ *consumer that asks for the derivative-step closed set must first display the route*

$$F, A, I \longmapsto F' \longmapsto W, R, Q, E \longmapsto H, C, P, N.$$

Thus the exported closed-set row F' is a finite BEDC handoff over the $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. The consumer receives no host transfinite derivative chain, no selected list of isolated points, no quotient closed-set equality, no countable-choice schedule, and no ambient compactness or completeness theorem.*

Proof. Begin with the input and test rows. The closed-set row F and accumulation-test row A are the only data from which a derivative-step read may decide whether an inspected point remains.

Next read the isolated-rejection row. Each rejection in I is accompanied by the displayed interval, radius, and membership data recorded through the dyadic comparison ledger Q.

The output closed-set row F' is downstream of those rows. The StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, and Real seal expose F' only through the same finite route.

The transport, replay, provenance, and local naming rows preserve that route and add no mathematical coordinate for the excluded host-level data. Hence the closed-set handoff factors through the displayed derivative-step packet. □

Falsifiable prediction: Every accepted $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ consumer that reads the output closed-set row while omitting the input closed set, accumulation tests, isolated-rejection data, stream windows, rational readback, dyadic comparisons, Real seal, transport, replay, provenance, or local name has left the packet.

Independence witness: cantorbendixson, locatedcantorbendixson, cantorbendixsonrank, closedset, streamname, regseqrat, dyadicratcore, real: $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ is not bijective with any listed sibling because it isolates one derivative-step operator with input and output closed-set rows, accumulation tests, isolated rejections, finite windows, arithmetic readback, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBendixsonDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorBendixsonDerivative}^{\uparrow}$ packet is generated from an input ClosedSetUp row, finite accumulation tests, isolated-rejection data, an output ClosedSetUp row, StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorBendixsonDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8110.1, Definition 8110.2, and Definition 8110.3 seed the finite one-step derivative operator for closed real-set packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorBendixsonDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full Cantor–Bendixson decomposition, host ordinal rank, selected isolated-point enumeration, quotient closed-set equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-step classifier stability, isolated-rejection exactness, output closed-set handoff, structural nonescape, and route replay for BEDC.Derived.CantorBendixsonDerivativeUp.

Chapter 8111

A Concrete Naming Certificate for MetaCICResidualDiamondFrontier$^{\uparrow}$

The MetaCICResidualDiamondFrontier$^{\uparrow}$ packet realizes the residual-diamond frontier of chapter 9378 as finite confluence-facing NameCert data. It sits beside the parallel-diamond frontier of chapter 4238, the residual-substitution boundary of chapter 2181, the confluence critical-pair boundary of chapter 2212, and the closed-normal confluence seal of chapter 741. The packet records candidate residual diamonds as bounded finite evidence while keeping BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, arbitrary closed-term Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency outside the carrier.

Falsifiable prediction: Every accepted MetaCICResidualDiamondFrontier$^{\uparrow}$ read exposes the residual-substitution premise and obstruction rows before any residual-diamond handoff is consumed.

Independence witness: metacic-parallel-diamond-frontier, metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, closednormalconfluenceseal: the parallel-diamond frontier records a conditional parallel handoff without this packet's residual-diamond candidate grid; the residual-substitution boundary names the missing compatibility premise without candidate residual joins; the critical-pair boundary records local peak rows without the residual-diamond handoff; and the closed-normal confluence seal supplies normal-source joins without residual substitution. None is bijective to MetaCICResidualDiamondFrontier$^{\uparrow}$.

Definition 8111.1 (MetaCIC residual-diamond frontier carrier)**.** *A* MetaCICResidualDiamondFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, R, J, B, O, X, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite critical-pair row, recording the inspected peak, branch labels, and local source window.*
2. *R is the residual-substitution premise row retained from the residual-substitution boundary.*
3. *J is the candidate-join row, recording the proposed residual-diamond common target and branch-local join routes.*
4. *B is the bounded-checker row, recording the finite checker budget for residual descendants, closedness, support, and branch compatibility.*

5. *O is the obstruction row naming every endpoint not discharged by the packet: residual-substitution compatibility, premise-free parallel diamond, arbitrary closed-term Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.*
6. *X is the residual-diamond handoff row exposed only after K, R, J, B, O have all been displayed.*
7. *H records componentwise* hsame *transport for critical-pair, residual-substitution, candidate-join, bounded-checker, obstruction, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a confluence query through K, R, J, B, O and then to X.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, R, J, B, O, X, H, C, P, N$. It has no coordinate for a discharged substitution-compatibility theorem, a premise-free parallel diamond, an arbitrary closed Church–Rosser theorem, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 8111.2 (MetaCIC residual-diamond frontier obligations)**.** *For every accepted* MetaCICResidualDiamondF *carrier $F = (K, R, J, B, O, X, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}}$ *surface consists exactly of finite critical-pair exposure through K, residual-substitution premise retention through R, candidate-join exposure through J, bounded-checker accounting through B, obstruction retention through O, residual-diamond handoff non-escape through X, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The Lean-side TasteGate surface must provide `Nontrivial` $\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}$ *and* `FieldFaithful` $\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}$ *instances in* `TasteGate.lean`*: the first separates genuine frontier packets from the empty carrier, and the second asserts that the classifier reads every displayed field $K, R, J, B, O, X, H, C, P, N$.*

Proof. Project the carrier of Definition 8111.1. The projection exposes exactly the critical-pair, residual-substitution, candidate-join, bounded-checker, obstruction, handoff, transport, replay, provenance, and local naming rows.

Read the confluence rows in order. The critical-pair row K fixes the inspected peak, the residual-substitution row R remains visible as a retained premise, the candidate-join row J records the proposed residual diamond, and the bounded checker B accounts for the finite branch checks.

The obstruction row O is retained before X can be consumed. Thus the residual-diamond handoff is a bounded frontier row, not a discharge of residual substitution, a premise-free parallel diamond, or a closed Church–Rosser endpoint.

Finally H, C, P, N transport, replay, source, and locally name the same finite route. The stated TasteGate obligations are precisely the classifier-readiness conditions for nonempty, field-faithful packet data. □

Theorem 8111.3 (MetaCIC residual-diamond frontier confluence handoff)**.** *Every confluence consumer of an accepted* $\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}$ *packet must display the route*

$$K \longmapsto R \longmapsto J \longmapsto B \longmapsto O \longmapsto X \longmapsto H, C, P, N$$

before consuming the residual-diamond handoff. The handoff may feed the conditional confluence boundary of chapter 9378, but it does not prove BetaParallelSubstitutionCompatibility*, does not*

prove BetaParallelDiamondClosed*, and does not prove arbitrary closed-term Church–Rosser for* BetaStarStep*.*

Proof. Let $F = (K, R, J, B, O, X, H, C, P, N)$ be accepted. By Definition 8111.1, the handoff row X is downstream of critical-pair exposure, residual-substitution premise retention, candidate-join exposure, bounded checking, and obstruction retention.

Apply the obligation surface of Definition 8111.2. It forces every residual-diamond read to keep K, R, J, B, O visible before the handoff row is available.

The structural rows preserve the same route. Componentwise transport through H, replay through C, provenance through P, and local naming through N do not add any of the omitted confluence endpoints. Therefore the handoff remains a finite residual-diamond frontier certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondFrontier}\uparrow}$.

Constructive story: A $\mathsf{MetaCICResidualDiamondFrontier}^{\uparrow}$ packet is generated from critical-pair rows, residual-substitution premise retention, candidate-join rows, bounded-checker rows, obstruction retention, a residual-diamond handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondFrontier}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8111.1, Definition 8111.2, and Definition 8111.3 seed the residual-diamond frontier over critical-pair, residual-substitution, candidate-join, bounded-checker, obstruction, handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualDiamondFrontier}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim residual-substitution compatibility, premise-free parallel diamond, arbitrary closed-term Church–Rosser, subject reduction, evaluator equality, quotient term equality, kernel consistency, public closure, or a standard bridge.

Upgrade path: Formalization can encode BEDC.Derived.MetaCICResidualDiamondFrontierUp as carrier admission, residual-diamond obligation rows, confluence handoff non-escape, Nontrivial TasteGate evidence, FieldFaithful TasteGate evidence, transport, replay, provenance, and local naming.

Chapter 8112

A Concrete Naming Certificate for $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$ packet records the finite criterion by which two displayed regular rational names are accepted as the same real-name seal. It sits after the regular rational readback of chapter 301, the finite observation surface of chapter 126, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the comparison-window route of chapter 411. The packet is a criterion surface: it names the displayed comparison windows that force the same $\mathsf{Real}^{\uparrow}$ seal, without quotienting streams, choosing representatives, invoking trichotomy, using countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8112.1 (Regular Cauchy equivalence criterion carrier)**.** *A* $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational names being compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule on which the comparison is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed windows.*
4. *Q is the* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *comparison row witnessing that every displayed tolerance window closes along the same route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, R, W, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the two names, window schedule, dyadic ledger, comparison row, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the same finite comparison route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $L, R, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8112.2 (Regular Cauchy equivalence criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$ *carrier* $K = (L, R, W, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *comparison row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The obligation surface may use the criterion only after the comparison windows and dyadic ledger have been displayed.*

Proof. Project the carrier of Definition 8112.1. The two candidate real names enter only as $\mathsf{RegSeqRat}^{\uparrow}$ rows L, R, and every consumer-visible comparison request must first pass through the finite window schedule W.

Read the tolerance and comparison rows. The dyadic ledger D supplies the displayed budgets, while the $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ row Q records that the inspected windows close under those budgets before the terminal seal is consumed.

Finally replay and transport the same route through H, C, P, N. These rows preserve the displayed comparison surface and add no quotient-stream equality, representative selector, trichotomy branch, countable-choice schedule, host real equality test, or ambient completeness coordinate. □

Theorem 8112.3 (Regular Cauchy equivalence criterion Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$ *packet reaches seal equality only through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow} \times \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyComparisonPrinciple}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

Thus the criterion can certify that the two regular Cauchy names present the same real seal inside the finite comparison surface, but it does not export quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligations of Definition 8112.2. They force the consumer to keep both regular rational rows, the shared finite window, the dyadic tolerance ledger, and the comparison row visible before the Real seal is read.

The terminal seal E is therefore downstream of a displayed comparison route. Any accepted seal read is replayed by C, transported by H, sourced by P, and locally named by N along the same finite packet.

The excluded objects cannot be projected from the carrier. Since K stores no quotient stream equality, representative selector, trichotomy branch, countable-choice schedule, host real equality test, or ambient completeness row, the real-seal conclusion remains a finite criterion rather than a quotient or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}$ packet is generated from two RegSeqRat names, a shared StreamName finite-window schedule, a DyadicRatCore tolerance ledger, a RegularCauchyComparisonPrinciple row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 8112.1, Definition 8112.2, and Definition 8112.3 seed the finite equivalence criterion linking RegSeqRat comparison windows to RealUp seal equality.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalenceCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-window exactness, dyadic tolerance stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEquivalenceCriterionUp.

Chapter 8113

A Concrete Naming Certificate for DescriptionComplexity$^{\uparrow}$

The DescriptionComplexity$^{\uparrow}$ packet realizes the explanation-complexity information surface of section 9462.10 as finite source-description, score-vector, public-weight, debt, classifier-novelty, minimal-witness, replay, provenance, and NameCert rows. It sits beside the existing ExplanationComplexityInformation$^{\uparrow}$ packet of chapter 6181, the KolmogorovComplexity$^{\uparrow}$ compression boundary of chapter 5989, the Hash$^{\uparrow}$ certificate of chapter 242, the finite prefix stream certificate of chapter 7253, and the classifier-novelty witness of chapter 6138. The packet measures displayed descriptions inside a public protocol; it does not identify minimal text with Kolmogorov optimality, recover hidden source equality from hashes, infer classifier novelty from score motion alone, or certify positive discovery without a later TasteGate.

Definition 8113.1 (Description complexity carrier)**.** *A* DescriptionComplexity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, V, W, B, R, J, M, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-description row naming the finite explanation text, feature list, circuit record, classifier specification, ledger summary, and verification surface.*
2. *V is the score-vector row refining the explanation-complexity coordinates of Definition 9462.233.*
3. *W is the public weight row used by the scalar score of Definition 9462.234.*
4. *B is the benefit and debt row recording coverage, prediction, transfer, intervention, verification, unification, and the debt components of Definition 9462.238.*
5. *R is the replay row tying the description to finite prefix evidence through* FinitePrefixStream$^{\uparrow}$ *and to visible digest discipline through* Hash$^{\uparrow}$*.*
6. *J is the* ClassifierNoveltyWitness$^{\uparrow}$ *row, present only when the description claims classifier novelty rather than a score-only improvement.*
7. *M is the minimal-witness row: it records a displayed lower-bound boundary for the accepted description under W, not an absolute Kolmogorov minimum.*
8. *F is the comparison row to* KolmogorovComplexity$^{\uparrow}$ *and* ExplanationComplexityInformation$^{\uparrow}$*, separating source-description cost from compression optimality and net explanation information.*

9. *H records componentwise* hsame *transport for S, V, W, B, R, J, M, F.*
10. *C records displayed* Cont *replay through source description, score-vector accounting, public weights, debt, classifier novelty, minimal-witness boundary, sibling comparison, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{DescriptionComplexity}^{\uparrow}$, $\mathsf{ExplanationComplexityInformation}^{\uparrow}$, $\mathsf{KolmogorovComplexity}$ $\mathsf{Hash}^{\uparrow}$, $\mathsf{FinitePrefixStream}^{\uparrow}$, $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DescriptionComplexity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, V, W, B, R, J, M, F, H, C, P, N$. It has no coordinate for hidden semantic truth, host compression optimality, hash preimage recovery, undisplayed weights, classifier novelty without J, positive discovery without a separate gate, or a minimal witness detached from the displayed public protocol.

Theorem 8113.2 (Description complexity NameCert obligations)**.** *For every accepted* $\mathsf{DescriptionComplexity}^{\uparrow}$ *carrier $D = (S, V, W, B, R, J, M, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DescriptionComplexity}^{\uparrow}}$ *surface consists exactly of carrier admission for source description, score-vector coordinates, public weights, benefit and debt accounting, finite replay and digest discipline, classifier-novelty witnessing when claimed, minimal-witness boundary, comparison to explanation-complexity and Kolmogorov-complexity siblings, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8113.1. The description itself is S, while V and W expose the coordinate vector and public weights before any scalar score can be read.

Read benefit and debt through B. This keeps the score from being interpreted as an explanation-quality verdict by itself, because debt and benefit remain visible in the same packet.

The rows R, J, M, F supply the boundary data. The replay row R ties the source description to finite prefix and hash-visible evidence; J records classifier novelty only when present; M states a protocol-relative minimal-witness boundary; and F separates description cost from Kolmogorov optimality and net explanation information.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for the excluded objects, the listed rows exhaust the local NameCert surface. □

Theorem 8113.3 (Description complexity minimal-witness boundary)**.** *An accepted* $\mathsf{DescriptionComplexity}^{\uparrow}$ *packet may use its minimal-witness row M only relative to the displayed source-description row S, score vector V, public weights W, benefit and debt row B, replay row R, classifier-novelty row J when present, and sibling-comparison row F. It does not certify an absolute Kolmogorov minimum, a hidden source recovered from a hash, a classifier shift without J, or a positive discovery verdict without a later TasteGate.*

Proof. Apply the NameCert obligations of Definition 8113.2. They force M to be read together with the public protocol rows S, V, W, B and with the replay and sibling-comparison rows R, F.

The Kolmogorov comparison is therefore only a boundary comparison. The packet may record that a description is minimal among displayed candidates under the public protocol, but it cannot project absolute compression optimality or hidden source equality.

Classifier novelty is likewise explicit. If the row J is absent, the packet has no classifier-shift witness; if J is present, it is still only a row of this description-complexity packet and not a positive-discovery gate.

Thus the minimal witness remains protocol-relative and finite. A later Lean file `TasteGate.lean` must provide Nontrivial $\mathsf{DescriptionComplexity}^{\uparrow}$ and FieldFaithful $\mathsf{DescriptionComplexity}^{\uparrow}$ instances

before any checked TasteGate theorem can certify that these rows are distinct and nondegenerate. □

Falsifiable prediction: Every accepted $\mathsf{DescriptionComplexity}^{\uparrow}$ minimal-witness read exposes the source description, score vector, public weights, benefit and debt accounting, finite-prefix replay, hash-visible provenance, classifier novelty when claimed, sibling comparison, transport, replay, provenance, and local NameCert naming.

Independence witness: explanation-complexity-information, kolmogorov-complexity, hash, name-cert, finite-prefix-stream-certificate: none is bijective to $\mathsf{DescriptionComplexity}^{\uparrow}$ because this packet jointly carries source-description rows, score-vector rows, public weights, debt and benefit accounting, classifier-novelty witnessing, protocol-relative minimal-witness boundaries, finite replay, digest-visible provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DescriptionComplexity}^{\uparrow}}$.

Constructive story: A $\mathsf{DescriptionComplexity}^{\uparrow}$ packet is generated from source-description rows, score-vector rows, public weights, benefit and debt accounting, finite replay, hash-visible provenance, classifier-novelty witnessing, protocol-relative minimal-witness boundaries, sibling comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DescriptionComplexity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8113.1, Definition 8113.2, Definition 8113.3, the falsifiable prediction, and the independence witness seed the finite description-complexity packet for explanation-complexity information.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DescriptionComplexity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close absolute Kolmogorov optimality, hidden semantic truth, hash preimage recovery, classifier novelty without the displayed witness row, positive discovery, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-description exposure, score-vector and public-weight accounting, debt and benefit exposure, classifier-novelty row separation, minimal-witness boundary exactness, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.DescriptionComplexityUp.

Chapter 8114

A Concrete Naming Certificate for $\mathsf{MarkovBlanket}^{\uparrow}$

Following Pearl, *Causality*, 2009, the $\mathsf{MarkovBlanket}^{\uparrow}$ packet records the finite causal-inference boundary around one displayed variable in a Bayesian network. It sits after the do-calculus surface of chapter 453, the intervention rows of chapter 1290, the normalized probability source of chapter 184, the random-variable carrier of chapter 185, and the conditional-independence rows of chapter 187. The packet names the parent, child, spouse, separating-set, d-separation, transport, replay, provenance, and local NameCert rows as BEDC data; it does not assert a host Bayesian-network semantics, global graph faithfulness, positivity, probabilistic completeness, or a selected causal model.

Definition 8114.1 (Markov blanket carrier)**.** *A* $\mathsf{MarkovBlanket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (V, G, P, C, S, B, D, I, H, R, Q, N)$$

with the following displayed rows:

1. *V is the focused* $\mathsf{RandomVar}^{\uparrow}$ *row whose local boundary is being named.*
2. *G is the finite directed-neighborhood row, exposing only the displayed parent, child, and co-parent incidences used by the certificate.*
3. *P is the parent row of V, read through the same probability source as* $\mathsf{ProbSpace}^{\uparrow}$*.*
4. *C is the child row of V, recording the visible outgoing incidences.*
5. *S is the spouse row, containing the displayed co-parents of children in C.*
6. *B is the blanket row P, C, S packaged as one finite boundary before any query consumer reads it.*
7. *D is the d-separation boundary row, recording that variables outside B are screened from V only through the displayed conditional-independence surface.*
8. *I is the* $\mathsf{Independence}^{\uparrow}$ *row used by the query consumer, with conditioning restricted to the displayed blanket row.*
9. *H records componentwise* hsame *transport for V, G, P, C, S, B, D, I and the local naming row.*
10. *R records displayed* Cont *replay from a causal or probabilistic query through the finite blanket rows.*

11. *Q is the* Pkg *provenance over* $\mathsf{MarkovBlanket}^{\uparrow}$, $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{CausalIntervention}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovBlanket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, G, P, C, S, B, D, I, H, R, Q, N$. It has no coordinate for a full Bayesian network, graph-faithfulness theorem, global factorization, positivity assumption, selected interventional model, host conditional-probability equality, or completed probability semantics.

Theorem 8114.2 (Markov blanket NameCert obligations)**.** *For every accepted* $\mathsf{MarkovBlanket}^{\uparrow}$ *carrier $M = (V, G, P, C, S, B, D, I, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MarkovBlanket}^{\uparrow}}$ *surface consists exactly of focused-variable admission, directed-neighborhood exposure, parent row exposure, child row exposure, spouse row exposure, blanket packaging, d-separation boundary admission, conditional-independence query admission, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8114.1. The focused variable and finite neighborhood are displayed as V, G, while the three boundary components are displayed as P, C, S and repackaged by B.

The causal and probabilistic dependencies are then read through the sibling rows named in the carrier. The probability source, random-variable row, causal-intervention surface, do-calculus surface, and independence row are all cited by Q before a query can consume the boundary.

Finally H, R, Q, N transport, replay, source, and name only the same finite rows. Since no carrier coordinate stores a full Bayesian network semantics, graph-faithfulness theorem, global factorization, positivity assumption, selected interventional model, host conditional-probability equality, or completed probability semantics, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8114.3 (Markov blanket d-separation boundary)**.** *Every accepted d-separation read exported by a* $\mathsf{MarkovBlanket}^{\uparrow}$ *packet factors through the finite route*

$$V, G \longmapsto P, C, S \longmapsto B \longmapsto D \longmapsto I \longmapsto H, R, Q, N.$$

The read can condition on the displayed blanket row B and can cite $\mathsf{Independence}^{\uparrow}$ *through I, but it cannot replace the route by a global Bayesian-network factorization, faithfulness assumption, positivity condition, or host conditional-probability equality.*

Proof. Start with the focused random variable V and the directed-neighborhood row G. The parent, child, and spouse rows P, C, S are the only displayed contributors to the blanket row B.

The d-separation row D is read after B has been exposed. It does not store a global graph theorem; it records only the finite screening boundary consumed by the conditional-independence row I.

The structural rows H, R, Q, N then preserve, replay, source, and name this route. Because the carrier has no coordinate for global factorization, faithfulness, positivity, or host conditional-probability equality, the d-separation read remains a finite boundary route. □

Theorem 8114.4 (Markov blanket nonescape)**.** *No accepted* $\mathsf{MarkovBlanket}^{\uparrow}$ *consumer can export a selected causal model, completed probability semantics, full Bayesian-network object, global graph-faithfulness theorem, positivity assumption, quotient equality of random variables, or host conditional-probability equality from the blanket packet. Every exported read is confined to the displayed rows $V, G, P, C, S, B, D, I, H, R, Q, N$.*

Proof. Inspect the carrier coordinates in Definition 8114.1. They display one focused variable, one finite directed-neighborhood row, three blanket components, one packaged blanket row, one d-separation boundary, one conditional-independence query row, and the structural transport, replay, provenance, and naming rows.

The dependency chapters named in the introduction supply only sibling surfaces: do-calculus rows, intervention rows, probability rows, random-variable rows, and independence rows. None of those rows is copied into this packet as a selected global model or as completed probability semantics.

Thus every accepted read is routed through $V, G, P, C, S, B, D, I, H, R, Q, N$. Since the refused objects have no displayed coordinate, no continuation, package, or local certificate row can export them. □

Falsifiable prediction: A consumer that obtains a full Bayesian-network factorization or host conditional-probability equality from B alone refutes the $\mathsf{MarkovBlanket}^{\uparrow}$ carrier.

Independence witness: do-calculus, causal-intervention, probspace, randomvar, independence: none is bijective with $\mathsf{MarkovBlanket}^{\uparrow}$ because the packet jointly carries a focused variable, finite neighborhood, parent, child, spouse, blanket, d-separation, conditional-independence, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovBlanket}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovBlanket}^{\uparrow}$ packet is generated from one focused RandomVar row, finite directed-neighborhood data, parent rows, child rows, spouse rows, a packaged blanket boundary, a d-separation boundary, an Independence query row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovBlanket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8114.1, Definition 8114.2, Definition 8114.3, and Definition 8114.4 seed the finite causal-inference blanket packet over focused-variable, neighborhood, boundary, d-separation, independence, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovBlanket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a full Bayesian-network semantics, global graph faithfulness, positivity, probabilistic completeness, selected causal model, quotient equality of random variables, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite blanket component exposure, d-separation boundary routing, conditional-independence query admission, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.MarkovBlanketUp.

Chapter 8115

A Concrete Naming Certificate for $\mathsf{BinarySignedDigit}^{\uparrow}$

Following Bishop and Bridges on signed-digit real representations, the $\mathsf{BinarySignedDigit}^{\uparrow}$ packet records the finite binary signed-digit substrate used before a signed-digit real name is read as a regular rational or sealed as a $\mathsf{Real}^{\uparrow}$ object. It sits below the signed-digit real expansion of chapter 6684, the dyadic signed-digit stream of chapter 6915, the dyadic arithmetic core of chapter 291, the finite window interface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite digit rows and dyadic readback data; it is not a quotient of infinite streams, a selected expansion, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8115.1 (BinarySignedDigit carrier)**.** *A* $\mathsf{BinarySignedDigit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the binary signed-digit row, listing only finite words over the digit alphabet $\{-1, 0, 1\}$ and the dyadic place attached to each displayed digit.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that schedules the visible signed-digit prefixes.*
3. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed dyadic readback ledger, recording the finite partial sum, carry normalization, endpoint value, and tolerance radius for each displayed prefix.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reading the signed dyadic ledger as regular rational data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the digit, window, dyadic, and regular-readback rows have been displayed.*
6. *H records componentwise* hsame *transport for digit rows, windows, signed dyadic readback, regular readback, Real sealing, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a digit-window request through dyadic readback, regular rational handoff, and Real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BinarySignedDigit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SignedDigitRealExpansion}^{\uparrow}$, $\mathsf{DyadicSignedDigitStream}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BinarySignedDigit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, Q, R, E, H, C, P, N$. It contains no coordinate for quotient stream equality, selected infinite expansions, host real equality, trichotomy, countable choice, or ambient completeness.

Theorem 8115.2 (BinarySignedDigit NameCert obligations)**.** *For every accepted* $\mathsf{BinarySignedDigit}^{\uparrow}$ *carrier*

$$B = (D, W, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BinarySignedDigit}^{\uparrow}}$ *surface consists exactly of digit-row admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed dyadic readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8115.1. Its analytic rows are precisely the finite digit row D, the stream-window row W, the signed dyadic readback ledger Q, the regular rational handoff R, and the Real seal E.

The digit and window obligations are the route through D and W. A consumer can inspect only the displayed finite signed-digit prefixes, with each digit attached to its dyadic place.

The arithmetic and readback obligations are the route through Q and R. The dyadic ledger records partial sums, carry normalization, endpoint values, and tolerance radii before the regular rational handoff is available.

The rows H, C, P, N transport, replay, source, and name exactly the same finite route. Since no other coordinate is present, the refused endpoints cannot be projected from an accepted packet. □

Theorem 8115.3 (BinarySignedDigit window induction)**.** *Let*

$$B = (D, W, Q, R, E, H, C, P, N)$$

be an accepted $\mathsf{BinarySignedDigit}^{\uparrow}$ *carrier. Every accepted extension of a displayed binary signed-digit prefix by one digit remains in the same carrier route only after the enlarged prefix is recorded in D, the enlarged window is recorded in W, and the corresponding signed dyadic readback is recorded in Q. Thus prefix growth is a finite window induction through displayed rows, not a selected infinite expansion or quotient stream equality.*

Proof. Proceed by induction on the finite displayed prefix window. The base window reads the empty prefix through D and W, and its signed dyadic readback is the base row in Q.

For the successor step, append one admitted digit to the visible prefix. The digit row D records the digit and dyadic place, while W records the enlarged inspection window.

The dyadic readback ledger Q then normalizes the enlarged partial sum and records its endpoint and radius. Transport through H and replay through C keep the route inside the same finite carrier.

Finally R, E, P, N read the resulting regular rational handoff, Real seal, provenance, and local name. No step introduces an infinite representative, quotient equality, choice schedule, trichotomy branch, host real equality, or completeness theorem. □

Theorem 8115.4 (BinarySignedDigit dyadic readback)**.** *Every* $\mathsf{SignedDigitRealExpansion}^{\uparrow}$, $\mathsf{DyadicSignedDigitStrea}$ $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BinarySignedDigit}^{\uparrow}$ *packet reads the finite route*

$$D, W \longmapsto Q \longmapsto R \longmapsto E, H, C, P, N.$$

The row Q is the unique local passage from binary signed-digit windows to signed dyadic endpoint and tolerance data, so downstream signed-digit and completion-facing consumers receive a finite dyadic readback packet rather than a host real, quotient stream class, selected expansion, or ambient completeness principle.

Proof. Start from the obligation surface of Definition 8115.2. It requires the digit row and StreamName window before any signed dyadic readback can be consumed.

Use the window induction theorem Definition 8115.3. Each finite prefix extension reaches Q only through the displayed digit and window rows, so Q is attached to a visible finite prefix.

The regular rational row R and the Real seal row E are downstream of Q. The signed-digit real expansion and dyadic signed-digit stream chapters can therefore consume the packet only as the same finite dyadic readback route.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and naming. Since the carrier has no coordinate for the excluded endpoints, no downstream consumer can recover them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinarySignedDigit}\uparrow}$.

Constructive story: A $\mathsf{BinarySignedDigit}^{\uparrow}$ packet is generated from finite binary signed-digit rows, StreamName windows, a DyadicRatCore signed dyadic readback ledger, RegSeqRat handoff rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BinarySignedDigit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8115.1, Definition 8115.2, Definition 8115.3, and Definition 8115.4 seed the finite binary signed-digit substrate route from digit windows to signed dyadic, regular rational, and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BinarySignedDigit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite expansions, trichotomy, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, digit-row induction, signed dyadic readback exactness, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BinarySignedDigitUp.

Chapter 8116

A Concrete Naming Certificate for $\mathsf{RationalBallFilter}^{\uparrow}$

Following Bishop and Bridges on neighborhood filters and constructive real approximation, the $\mathsf{RationalBallFilter}^{\uparrow}$ packet records a finite rational-ball request route for Real and Completion consumers. It sits after the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric-completion boundary of chapter 1506. The packet names rational-neighborhood evidence through displayed finite rows only; it does not quotient real names, select representatives, invoke trichotomy, use countable choice, assert open-cover compactness, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8116.1 (RationalBallFilter carrier)**.** *A* $\mathsf{RationalBallFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational center request row, recording the requested center and comparison budget as finite rational data.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic radius and tolerance ledger attached to the same request.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that reads the stream window as regular rational approximation data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the rational center, dyadic ledger, stream window, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for the rational request, dyadic ledger, stream window, readback, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a rational-ball request through Q, D, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalBallFilter}^{\uparrow}$*, rational center data,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBallFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, W, R, E, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, trichotomy, countable-choice schedules, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8116.2 (RationalBallFilter NameCert obligations)**.** *For every accepted* $\mathsf{RationalBallFilter}^{\uparrow}$ *carrier $F = (Q, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalBallFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, rational center request exposure, ball-window coverage, dyadic refinement stability, regular rational readback, Real-seal nonescape, ledger exhaustion, componentwise transport, displayed replay, provenance, and local naming for the rows $Q, D, W, R, E, H, C, P, N$.*

Proof. Project the carrier of Definition 8116.1. The only available rows are $Q, D, W, R, E, H, C, P, N$, so carrier admission exposes the rational request, dyadic ledger, stream window, regular readback, terminal seal, transport, replay, provenance, and local naming data.

The ball-window coverage clause is read through Q, W, R. The rational center request opens only the stored finite StreamName windows, and the RegSeqRat row reads those windows as regular rational approximation data.

Dyadic refinement stability and ledger exhaustion are read through D. Refining a radius or tolerance budget changes only the displayed DyadicRatCore ledger row, and every accepted request must spend the ledger before the terminal seal can be inspected.

Finally E, H, C, P, N preserve the same route. The Real-seal row is downstream of Q, D, W, R, while transport, replay, provenance, and local naming keep the packet inside the displayed finite data. The excluded quotient, selection, trichotomy, choice, compactness, and ambient-completeness objects are absent from the carrier, so the listed clauses exhaust the local NameCert surface. □

Theorem 8116.3 (RationalBallFilter finite-window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RationalBallFilter}^{\uparrow}$ *carrier must read the rational ball request through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before terminal Real evidence is consumed. The handoff uses the displayed rows $Q, D, W, R, E, H, C, P, N$ of the same packet and cannot import quotient real equality, selected representatives, trichotomy, countable choice, open-cover compactness, or ambient completeness.

Proof. Start with an accepted $\mathsf{Real}^{\uparrow}$ or completion-facing consumer of F. The carrier admits a rational center request Q only together with the dyadic radius and tolerance ledger D.

The consumer then opens the finite window W and reads it through R. These rows are the only route from the rational-ball request to regular rational approximation data.

Only after Q, D, W, R have been displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed. Thus a completion-facing consumer receives rational-neighborhood evidence through finite dyadic, stream, and regular-readback rows before any terminal Real evidence is available.

Transport, replay, provenance, and local naming are confined to H, C, P, N. Since the carrier contains no quotient equality, representative selector, trichotomy branch, choice schedule, open-cover compactness row, or ambient-completeness row, the handoff remains a finite rational-ball route. □

Falsifiable prediction: An accepted $\mathsf{RationalBallFilter}^{\uparrow}$ read is refuted if a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reaches the terminal seal without displaying the rational center request, DyadicRatCore tolerance ledger, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBallFilter}^{\uparrow}}$.

Constructive story: A RationalBallFilter$^{\uparrow}$ packet is generated from a rational center request, a DyadicRatCore radius and tolerance ledger, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RationalBallFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8116.1, Definition 8116.2, and Definition 8116.3 seed the finite rational-ball route over rational request, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RationalBallFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, trichotomy, countable choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ball-window coverage, dyadic refinement stability, Real-seal nonescape, ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.RationalBallFilterUp.

Chapter 8117

A Concrete Naming Certificate for $\mathsf{RationalBallCover}^{\uparrow}$

Following Bishop and Bridges on rational covers and constructive real neighborhoods, the $\mathsf{RationalBallCover}^{\uparrow}$ packet records finite rational-ball cover evidence for Real and Completion consumers. It sits after the request route of chapter 8116, the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite cover evidence through displayed rows only; it does not select a subcover by choice, quotient real names, invoke trichotomy, assert compactness, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8117.1 (Choice-free RationalBallCover source)**.** *A* $\mathsf{RationalBallCover}^{\uparrow}$ *source packet is a finite* BHist *record*

$$U = (F, L, D, W, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RationalBallFilter}^{\uparrow}$ *request row that supplies the rational center and ball-request surface.*
2. *L is the finite cover ledger, listing the displayed rational-ball entries and their finite indices.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius-refinement and tolerance ledger attached to the cover entries.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the cover request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that reads the stream windows as regular rational approximation data.*
6. *M is the cover-modulus handoff row, recording the finite modulus by which a consumer reads enough cover entries for the requested window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the filter request, cover ledger, dyadic refinement, stream window, regular readback, and modulus rows have been displayed.*
8. *H records componentwise* hsame *transport for the filter request, finite cover ledger, dyadic ledger, stream windows, readback, modulus, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the rational-ball filter request through $F, L, D, W, R, M, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalBallCover}^{\uparrow}$, $\mathsf{RationalBallFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBallCover}^\uparrow}$ *row.*

The five NameCert rows are carrier admission, finite cover ledger, dyadic refinement stability, cover-modulus handoff, and Real-seal nonescape. The carrier has no coordinate for countable choice, selected subcovers, open-cover compactness, quotient real equality, trichotomy, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8117.2 (Choice-free RationalBallCover finite subcover)**.** *Every accepted* $\mathsf{RationalBallCover}^\uparrow$ *packet $U = (F, L, D, W, R, M, E, H, C, P, N)$ exposes a finite subcover surface by reading the* $\mathsf{RationalBallFilter}^\uparrow$ *request row F, the finite ledger L, the* $\mathsf{DyadicRatCore}^\uparrow$ *refinement row D, the* $\mathsf{StreamName}^\uparrow$ *window W, and the* $\mathsf{RegSeqRat}^\uparrow$ *readback R before any terminal* $\mathsf{Real}^\uparrow$ *seal is consumed. The subcover is the displayed finite ledger in L transported through H, C, P, N; it is not selected by countable choice, compactness, quotient equality, trichotomy, or ambient completeness.*

Proof. Project the source packet of Definition 8117.1. Carrier admission exposes only $F, L, D, W, R, M, E, H, C, P, N$.

The finite cover ledger L is downstream of the $\mathsf{RationalBallFilter}^\uparrow$ request F. Thus a cover-facing consumer first reads the rational-ball request route and then the finite list of displayed rational-ball entries.

Dyadic refinement and window coverage are supplied by D, W, R. The cover entries are read through the same dyadic tolerance, stream-window, and regular-rational readback surface used by the Real and Completion route.

Finally H, C, P, N transport, replay, source, and name the same finite ledger. Since the carrier has no coordinate for choice, compactness, quotient equality, trichotomy, selected subcovers, or ambient completeness, the exported subcover surface is exactly the displayed finite ledger. □

Theorem 8117.3 (Choice-free RationalBallCover modulus)**.** *Every* $\mathsf{Real}^\uparrow$ *or Completion consumer of an accepted* $\mathsf{RationalBallCover}^\uparrow$ *packet reads the cover-modulus handoff through*

$$\mathsf{RationalBallFilter}^\uparrow,\ \mathsf{DyadicRatCore}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The modulus row M may be consumed only after the finite cover ledger L, dyadic refinement row D, stream window W, and regular rational readback R have been displayed, and before the terminal seal E is read. The handoff exports no countable-choice schedule, compactness theorem, selected limit, quotient real equality, trichotomy branch, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Start with an accepted $\mathsf{Real}^\uparrow$ or Completion consumer of U. The consumer receives the rational-ball request from F and the finite cover entries from L.

Next consume D, W, R. The dyadic ledger controls radius refinement, the stream window fixes the finite observation surface, and the regular rational readback interprets that window as approximation data.

Only then can the modulus row M be read. It records how much of the displayed finite cover ledger is needed for the requested window, without adding a selector or choice schedule.

The terminal seal E is downstream of F, L, D, W, R, M. Transport, replay, provenance, and local naming remain confined to H, C, P, N, so the Real and Completion handoff reaches only the displayed finite-cover route and none of the excluded compactness, quotient, trichotomy, choice, selected-limit, or completeness principles. □

Falsifiable prediction: An accepted $\mathsf{RationalBallCover}^\uparrow$ read is refuted if a $\mathsf{Real}^\uparrow$ or Completion consumer reaches the terminal seal without displaying the RationalBallFilter request, finite cover ledger, DyadicRatCore refinement row, StreamName window, RegSeqRat readback, cover-modulus handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBallCover}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBallCover}^{\uparrow}$ packet is generated from a RationalBallFilter request, finite rational-ball cover ledger, DyadicRatCore radius refinement, StreamName windows, RegSeqRat readback, cover-modulus handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBallCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8117.1, Definition 8117.2, and Definition 8117.3 seed the finite rational-ball cover surface over RationalBallFilterUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBallCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected subcovers, open-cover compactness, selected limits, quotient real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover ledger exposure, dyadic refinement stability, cover-modulus handoff, Real-seal nonescape, ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.RationalBallCoverUp.

Chapter 8118

A Concrete Naming Certificate for CompactUniformFinalTriangleClosure$^{\uparrow}$

Following Bishop and Bridges on the Heine–Cantor compact metric argument, the CompactUniformFinalTriangleClos packet records the last source-and-target triangle closure in the compact-uniform-continuity route of chapter 9237. It sits after the epsilon-triangle handoff of chapter 1257, the source-target triangle ledger of chapter 7260, and the metric-triangle uniform estimate of chapter 1159. The packet names a finite compact-net, pointwise-modulus, tolerance-budget, UniformModulus, and Real-seal closure route; it is not a host compactness theorem, an external modulus selector, quotient equality, choice of hidden centers, or ambient completeness principle.

Definition 8118.1 (Compact-uniform final triangle closure carrier)**.** *A* CompactUniformFinalTriangleClosure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-net source-distance row, recording the selected finite center, the source distance from the first point to that center, and the source distance between the two compared points.*
2. *M is the pointwise modulus row consumed at the selected center.*
3. *S is the source triangle estimate row, keeping the second point inside the same local source ball.*
4. *T is the target triangle estimate row, joining the two pointwise target estimates through the selected center image.*
5. *D is the dyadic tolerance budget row, carrying the finite split of the requested tolerance among source, pointwise, and target triangle contributions.*
6. *U is the* UniformModulus$^{\uparrow}$ *handoff row produced after K, M, S, T, D have been displayed.*
7. *E is the* Real$^{\uparrow}$ *seal row available only after the uniform-modulus handoff has been read.*
8. *H records componentwise* hsame *transport for compact-net source distances, pointwise modulus rows, source triangle rows, target triangle rows, dyadic tolerance budgets, uniform-modulus handoffs, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from compact-net coverage through pointwise modulus, source triangle, target triangle, dyadic tolerance budget, UniformModulus handoff, and Real sealing.*

10. *P is the* Pkg *provenance over* CompactUniformFinalTriangleClosure$^{\uparrow}$, UniformModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *together with the compact-uniform triangle sibling rows cited above.*

11. *N is the local* NameCert$_{\text{CompactUniformFinalTriangleClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, S, T, D, U, E, H, C, P, N$. It has no coordinate for host compactness, a hidden center choice, quotient equality, selected subsequences, semantic continuity, or ambient completeness.

Theorem 8118.2 (Compact-uniform final triangle closure carrier)**.** *Every accepted* CompactUniformFinalTriangleCl *packet exposes exactly the finite carrier rows of Definition 8118.1. In particular, a public read of the packet must factor through compact-net source distances, a pointwise modulus row, source and target triangle estimates, a dyadic tolerance budget, a* UniformModulus$^{\uparrow}$ *handoff, a* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The rows $K, M, S, T, D, U, E, H, C, P, N$ are the whole public surface of the accepted packet.

The analytic rows are K, M, S, T, D, U, E. They record the compact-net source distances, the pointwise modulus at the selected center, the two triangle estimates, the dyadic tolerance budget, the uniform-modulus handoff, and the Real seal.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. Since the carrier has no additional coordinate, no host compactness theorem, hidden center choice, quotient equality, selected subsequence, semantic continuity theorem, or ambient completeness principle can be exported. □

Theorem 8118.3 (Compact-uniform final triangle closure NameCert obligations)**.** *For every accepted* CompactUniformFinalTriangleClosure$^{\uparrow}$ *carrier*

$$F = (K, M, S, T, D, U, E, H, C, P, N),$$

the local NameCert$_{\text{CompactUniformFinalTriangleClosure}^{\uparrow}}$ *surface consists exactly of compact-net source-distance admission, pointwise modulus admission, source triangle admission, target triangle admission, dyadic tolerance-budget admission,* UniformModulus$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Read compact-net source-distance admission from K. This row fixes the selected center and the two source distances used by the source triangle estimate.

Read pointwise modulus admission from M and source triangle admission from S. The source triangle keeps the second point inside the selected local radius, so the same pointwise modulus row may be consumed for both compared source points.

Read target triangle admission from T and dyadic tolerance-budget admission from D. The target triangle combines the two pointwise target estimates, while the dyadic budget records the finite tolerance split that makes the estimate fit the requested bound.

Finally read the handoff and seal rows U and E. The rows H, C, P, N preserve this source-and-target triangle route under transport, replay, provenance, and local naming, and no further NameCert obligation is available. □

Theorem 8118.4 (Compact-uniform final triangle closure handoff)**.** *Every* UniformModulus$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* CompactUniformFinalTriangleClosure$^{\uparrow}$ *packet receives only the finite route*

$$K, M, S, T, D \longmapsto U \longmapsto E, H, C, P, N.$$

The handoff is the closure of the displayed compact-net source distance, pointwise-modulus, source-triangle, target-triangle, and dyadic-budget rows, not an external modulus selector, host compactness argument, quotient equality, selected subsequence, or ambient Real completeness theorem.

Proof. Start with the carrier theorem Definition 8118.2. It identifies K, M, S, T, D as the only estimate rows available before the $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Use the NameCert obligations of Definition 8118.3. They force the source triangle and target triangle to be read through the same compact-net center and the same dyadic tolerance budget.

The row U is therefore a handoff from those displayed rows, and the Real seal E is downstream of U. Transport, replay, provenance, and local naming keep the same route through H, C, P, N.

Since no excluded coordinate occurs in the carrier, the consumer receives a finite closure packet rather than an outside compactness or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFinalTriangleClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFinalTriangleClosure}^{\uparrow}$ packet is generated from compact-net source-distance rows, pointwise modulus rows, source and target triangle estimates, a dyadic tolerance budget, a UniformModulus handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFinalTriangleClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8118.1, Definition 8118.2, Definition 8118.3, and Definition 8118.4 seed the finite source-and-target triangle closure route from compact-net and pointwise-modulus data to UniformModulus and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFinalTriangleClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host compactness, external modulus selection, quotient equality, selected subsequences, semantic continuity, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-net source-distance admission, pointwise modulus admission, source and target triangle admission, dyadic tolerance-budget exactness, UniformModulus handoff, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformFinalTriangleClosureUp.

Chapter 8119

A Concrete Naming Certificate for $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$

Following Bishop and Troelstra–van Dalen, the $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ packet records the finite located read by which a choice-sequence prefix is consumed by Bishop-style real analysis without importing a selected infinite choice function. It sits after the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the located choice-sequence surface of chapter 5958. The packet supplies an L10 boundary object for located choice-sequence reads; it is not a countable-choice selector, quotient stream equality, host real endpoint, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8119.1 (Bishop located choice sequence carrier)**.** *A* $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, W, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the choice-sequence prefix row, recording only the inspected finite portion of the input sequence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that names the finite observation window available to the read.*
3. *W is the located-window row, binding the inspected prefix to an explicit finite interval of observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the located comparison.*
5. *L is the local locatedness row, saying that each displayed tolerance request is answered by finite prefix evidence.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from Q, S, W, D, L.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the prefix, window, tolerance, locatedness, and regular readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the prefix, schedule, window, tolerance, locatedness, readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through exactly the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$, $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedChoiceSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, S, W, D, L, R, E, H, C, P, N$. It has no coordinate for a completed choice function, countable choice, quotient stream equality, selected endpoint, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8119.2 (Bishop located choice sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ *carrier*

$$B = (Q, S, W, D, L, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedChoiceSequence}^{\uparrow}}$ *surface consists exactly of prefix exposure through Q, finite schedule exposure through S, located-window exposure through W, dyadic tolerance exposure through D, locatedness exposure through L, regular-rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every Real or Completion consumer of the packet must pass through these rows before it reads a located choice-sequence endpoint.*

Proof. Project the carrier of Definition 8119.1. The rows Q, S, W expose only the finite prefix and observation window available to the consumer, while D records the dyadic tolerance budget attached to that window.

Read locatedness next. The row L answers tolerance requests by displayed prefix evidence, so the regular readback R can be consumed only after the finite window and tolerance ledger have been inspected.

Finally consume the terminal seal and structural rows. The Real seal E, componentwise transport H, replay C, provenance P, and local name N preserve the same finite route. Since no carrier coordinate supplies countable choice, quotient stream equality, a selected endpoint, host real equality, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8119.3 (Bishop located choice sequence window exhaustion)**.** *Every accepted* BishopLocatedChoiceSeque *read that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed route*

$$\mathsf{LocatedRealChoiceSequence}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

on the same $Q, S, W, D, L, R, E, H, C, P, N$ rows. The route exhausts the available finite windows and exports no countable-choice schedule, quotient stream equality, hidden infinite branch, selected limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with an accepted carrier $B = (Q, S, W, D, L, R, E, H, C, P, N)$. By Definition 8119.2, any Real-facing read must first display the prefix row, schedule row, located window, dyadic tolerance ledger, and locatedness row.

The regular readback is then forced through R. It is produced from the displayed prefix-window route and cannot be replaced by a host stream, quotient equality, or selected infinite branch.

The terminal seal E is reached only after this readback, and H, C, P, N preserve the same packet. Hence every downstream read exhausts the displayed windows and cannot introduce countable choice, selected limits, host real equality, or ambient completeness. □

Falsifiable prediction: A $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ read is rejected if it reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without displaying the choice-sequence prefix, StreamName schedule, located window, DyadicRatCore tolerance ledger, locatedness row, transport, replay, provenance, and local naming.

Independence witness: located-real-choice-sequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a choice-sequence prefix, a finite observation schedule, a located window, a dyadic tolerance ledger, a locatedness row, regular readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedChoiceSequence}^{\uparrow}$ packet is generated from a choice-sequence prefix, StreamName finite schedule, located window, DyadicRatCore tolerance ledger, locatedness row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8119.1, Definition 8119.2, and Definition 8119.3 seed the finite located choice-sequence route over prefix, window, tolerance, locatedness, readback, Real sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BishopLocatedChoiceSequenceUp`

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient stream equality, hidden infinite branches, selected limits, host real equality, ambient RealUp completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window visibility, tolerance-ledger exposure, locatedness nonescape, RegSeqRat readback ordering, RealUp seal boundary control, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedChoiceSequenceUp.

Chapter 8120

A Concrete Naming Certificate for BoundedRealUniformContinuityModulus$^{\uparrow}$

Following Bishop and Bridges and the Heine–Cantor theorem for compact intervals, the BoundedRealUniformContinu packet records the finite modulus surface for a displayed continuous map on a bounded real interval. It sits after the terminal real-name seal of chapter 38, the compact metric source of chapter 129, the pointwise continuity rows of chapter 130, the uniform-modulus consumer of chapter 369, and the dyadic tolerance ledger of chapter 291. The packet names the bounded Real and Completion modulus route; it is not a host Heine–Cantor theorem, an open-cover compactness principle, a quotient of real functions, a selected subcover, or ambient Real$^{\uparrow}$ completeness.

Definition 8120.1 (Bounded real uniform-continuity modulus carrier)**.** *A* BoundedRealUniformContinuityModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, U, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the bounded* Real$^{\uparrow}$ *interval row, recording endpoints, membership, and the displayed closed-bounded interval source;*
2. *K is the* CompactMetric$^{\uparrow}$ *finite-net row over the same interval source, with a finite family of centers and mesh histories;*
3. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row for the displayed map on the interval;*
4. *U is the* UniformModulus$^{\uparrow}$ *extraction row obtained only after the finite-net and pointwise-modulus rows have been replayed;*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, recording positive target tolerance, center radii, mesh bounds, slack, and the exported source precision;*
6. *E is the terminal* Real$^{\uparrow}$ *seal row for all endpoint, value, and tolerance reads used by the packet;*
7. *H records componentwise* hsame *transport for the interval, finite-net, pointwise-modulus, uniform-modulus, dyadic-ledger, real-seal, provenance, and naming rows;*
8. *C records displayed* Cont *replay routes for reading the same rows in consumer order;*
9. *P is the* Pkg *provenance over* BoundedRealUniformContinuityModulus$^{\uparrow}$, Real$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRealUniformContinuityModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for host compactness, open-cover compactness, a selected subcover, quotient equality of functions, trichotomy of reals, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached semantic continuity theorem.*

Theorem 8120.2 (Bounded real uniform-continuity modulus NameCert obligations)**.** *For the carrier of Definition 8120.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealUniformContinuityModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (I, K, F, U, D, E, H, C, P, N),$$

bounded real interval exposure through I, compact finite-net exposure through K, $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus exposure through F,* $\mathsf{UniformModulus}^{\uparrow}$ *extraction through U, dyadic tolerance exactness through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed packet M. The rows $I, K, F, U, D, E, H, C, P, N$ list the bounded interval, finite-net source, pointwise continuity data, uniform-modulus extraction, tolerance ledger, Real seal, transports, replay routes, provenance, and local name.

The positive modulus route is finite. A consumer first reads I as the bounded $\mathsf{Real}^{\uparrow}$ interval, then K as the finite compact-metric net over that source, then F as the pointwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus row at the displayed centers.

The output row U is exposed only after the dyadic ledger D records the target tolerance, mesh, slack, and source precision, and after E, H, C, P, N seal, transport, replay, source, and name those rows. Since no further coordinate is present, the local NameCert surface is exhausted by the displayed packet. □

Theorem 8120.3 (Bounded real uniform-continuity modulus consumer handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BoundedRealUniformContinuityModulus}^{\uparrow}$ *carrier reads a uniform-continuity modulus only through the finite route*

$$I, K, F, D, E, H, C, P, N \longmapsto U.$$

The handoff may cite the compact finite-net source, the $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it exports no host Heine–Cantor theorem, open-cover compactness principle, selected subcover, quotient equality of real functions, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8120.2. They identify $I, K, F, U, D, E, H, C, P, N$ as the complete local certificate surface for the modulus packet.

Read the consumer route in order. The interval row I and compact finite-net row K supply the source coverage; the pointwise-modulus row F supplies local target control at the displayed centers; and the dyadic ledger D records the finite tolerance calculation that feeds the output row U.

Finally use the structural rows E, H, C, P, N. They seal the Real-facing reads, transport the rows componentwise, replay the displayed route, record provenance, and name the packet. None of these rows adds host compactness, a selected subcover, function quotient equality, countable choice, or ambient Real completeness, so the handoff is exactly the finite uniform-modulus route. □

Falsifiable prediction: Every accepted BoundedRealUniformContinuityModulusUp consumer that exports a uniform-continuity modulus without displaying a bounded Real interval row, compact finite-net row, ContinuousMap pointwise-modulus row, DyadicRatCore tolerance ledger, RealUp seal, transport, replay, provenance, and local NameCert row refutes the packet.

Independence witness: real, compactmetric, continuousmap, uniformmodulus, dyadicratcore: none is bijective to BoundedRealUniformContinuityModulusUp because the packet jointly carries the bounded interval source, finite-net coverage, pointwise-modulus data, uniform-modulus extraction, dyadic tolerance accounting, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealUniformContinuityModulus}\uparrow}$.

Constructive story: A $\mathsf{BoundedRealUniformContinuityModulus}^{\uparrow}$ packet is generated from a bounded Real interval row, compact finite-net row, ContinuousMap pointwise-modulus row, UniformModulus extraction row, DyadicRatCore tolerance ledger, Real seal, componentwise hsame transports, Cont replay routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealUniformContinuityModulus}\uparrow}, \mathsf{obligation})$

Scope closed: Definition 8120.1, Definition 8120.2, and Definition 8120.3 bind the finite bounded-Real uniform-modulus surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealUniformContinuityModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Heine–Cantor, open-cover compactness, selected subcovers, quotient equality of real functions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked source coverage, pointwise-modulus replay, dyadic tolerance exactness, Real sealing, transport, provenance, and non-escape for the same finite packet.

Chapter 8121

A Concrete Naming Certificate for $\mathsf{SetValuedMap}^{\uparrow}$

Following Aubin–Frankowska set-valued analysis and Berge's compact-value route, the $\mathsf{SetValuedMap}^{\uparrow}$ packet records a multifunction interface as finite graph, fibre, closed-boundary, and compact-image rows. It sits between the topological and metric sources of chapter 88 and chapter 57, the terminal real seal of chapter 38, the closed-set boundary of chapter 4229, and the compact-image closed consumer of chapter 4162. The packet names a finite correspondence surface; it is not a host powerset-valued function, selected-value operator, quotient graph equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 8121.1 (Set-valued map carrier)**.** *A* $\mathsf{SetValuedMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, G, F, L, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *row whose finite probes name the domain windows.*
2. *Y is the target* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row whose finite probes name the value windows.*
3. *G is the graph ledger, recording displayed source probes, target probes, and membership rows for the correspondence graph.*
4. *F is the fibre ledger, recording the finite rows that expose each displayed value fibre over a source probe.*
5. *L is the* $\mathsf{ClosedSet}^{\uparrow}$ *row used to read closed-graph or closed-fibre boundaries.*
6. *K is the* $\mathsf{CompactImageClosed}^{\uparrow}$ *consumer row used when compact-source image data must be read as a closed target boundary.*
7. *H records componentwise* hsame *transport for X, Y, G, F, L, K and the structural rows.*
8. *C records displayed* Cont *replay from source probes through graph membership, fibre exposure, closed-boundary readback, and compact-image consumption.*
9. *P is the* Pkg *provenance over* $\mathsf{SetValuedMap}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{CompactImageClosed}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SetValuedMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, G, F, L, K, H, C, P, N$. It has no coordinate for an ambient multifunction object, host powerset of values, pointwise value selection, quotient equality of graphs, open-cover compactness, or detached closedness theorem.

Theorem 8121.2 (Set-valued map NameCert obligations)**.** *For every accepted* $\mathsf{SetValuedMap}^{\uparrow}$ *carrier*

$$M = (X, Y, G, F, L, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SetValuedMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, source-window exposure through X, target-window exposure through Y, graph membership exposure through G, fibre exposure through F, closed-boundary readback through L, compact-image closed consumption through K, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The source and target rows X, Y determine the only windows in which the correspondence can be inspected, while G records graph membership and F records fibre exposure.

Read the boundary rows next. The closed-set row L is the only route to closed-graph or closed-fibre information, and the compact-image row K is the only route by which compact-source image data reaches a closed target consumer.

Finally inspect the structural rows H, C, P, N. They transport, replay, source, and name the same finite packet. Since no carrier coordinate stores an ambient multifunction, host powerset, selected value, graph quotient, open-cover proof, or detached closedness theorem, the listed rows exhaust the NameCert surface. □

Theorem 8121.3 (Closed-graph and compact-image handoff)**.** *Every closed-graph or compact-image consumer of an accepted* $\mathsf{SetValuedMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Topology}^{\uparrow}/\mathsf{Met}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto G \longmapsto F \longmapsto \mathsf{ClosedSet}^{\uparrow} \longmapsto \mathsf{CompactImageClosed}^{\uparrow} \longmapsto H, C, P, N.$$

In particular, closed-graph reasoning and compact-image closedness are handed to consumers as finite NameCert rows, not as host semantics for a multifunction.

Proof. Begin with the source and target rows. Any consumer first receives the finite source probes in X and target probes in Y before it can inspect graph membership or fibre data.

The graph and fibre rows then expose the correspondence surface. Closed-graph reads pass through the $\mathsf{ClosedSet}^{\uparrow}$ row L, while compact-image reads pass through the $\mathsf{CompactImageClosed}^{\uparrow}$ row K.

The transport, replay, provenance, and local naming rows keep this handoff attached to the same packet. Thus downstream closed-set and compact-image consumers receive finite certificate data, with no ambient multifunction semantics or selected-value principle added by the packet. □

Falsifiable prediction: Every accepted $\mathsf{SetValuedMap}^{\uparrow}$ consumer read must display source, target, graph, fibre, closed-boundary, compact-image, transport, replay, provenance, and local naming rows before using closed-graph or compact-image reasoning.

Independence witness: closedset and compact-image-closed: neither is bijective to $\mathsf{SetValuedMap}^{\uparrow}$ because the closed-set packet isolates a boundary row, the compact-image packet isolates a compact-source closed-target consumer, and the set-valued-map packet binds source, target, graph, fibre, closed-boundary, compact-image, transport, replay, provenance, and local naming rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SetValuedMap}^{\uparrow}}$.

Constructive story: A $\mathsf{SetValuedMap}^{\uparrow}$ packet is generated from source windows, target windows, graph membership rows, fibre rows, closed-boundary readback, compact-image closed consumption, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SetValuedMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8121.1, Definition 8121.2, and Definition 8121.3 seed the finite multifunction interface for closed-graph and compact-image consumers over $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, and $\mathsf{CompactImageClosed}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SetValuedMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host powerset semantics, pointwise value selection, quotient graph equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph and fibre exposure, closed-boundary readback, compact-image handoff, transport and replay stability, and local NameCert provenance for BEDC.Derived.SetValuedMapUp.

Chapter 8122

A Concrete Naming Certificate for $\mathsf{AbelPoissonSummability}^{\uparrow}$

Following Abel, Poisson, and Zygmund, the $\mathsf{AbelPoissonSummability}^{\uparrow}$ packet records the finite radial summability handoff by which Abel summability rows and Poisson-kernel rows feed the L10 Real and Completion route through displayed radius windows. It sits after the Abel summability route of chapter 6420, the positive Poisson-kernel boundary surface of chapter 2293, the Abel theorem radial boundary of chapter 6058, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names a finite radial ledger, not a host boundary limit, selected harmonic extension, quotient series equality, countable choice, or ambient Real completeness.

Definition 8122.1 (Abel–Poisson summability carrier)**.** *An* $\mathsf{AbelPoissonSummability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, K, T, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelSummability}^{\uparrow}$ *row exposing the Abel summability source, radial-window readback, and existing finite Real-seal ordering.*
2. *K is the* $\mathsf{PoissonKernel}^{\uparrow}$ *row exposing the positive radius row, angle-window row, denominator ledger, numerator ledger, kernel-value readback, and Fejer/Dirichlet comparison surface.*
3. *T is the* $\mathsf{AbelTheorem}^{\uparrow}$ *row recording the radial-boundary handoff used by the same radius ledger.*
4. *D is the dyadic radial ledger, storing finite radii, dyadic tolerances, kernel denominators, and comparison margins.*
5. *W records the finite radius windows, coefficient windows, angle windows, and kernel-value windows inspected by A, K, T, D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed Abel and Poisson windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row visible only after A, K, T, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the Abel, Poisson, theorem, dyadic-ledger, window, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from each finite radial request through A, K, T, D, W to R and then to E.*

10. *P is the* Pkg *provenance over* $\mathsf{AbelPoissonSummability}^{\uparrow}$, $\mathsf{AbelSummability}^{\uparrow}$, $\mathsf{PoissonKernel}^{\uparrow}$, $\mathsf{AbelTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelPoissonSummability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, K, T, D, W, R, E, H, C, P, N$. It has no coordinate for host boundary limits, harmonic-function equality, selected sums, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8122.2 (Abel–Poisson summability NameCert obligations)**.** *For every accepted* $\mathsf{AbelPoissonSummability}^{\uparrow}$ *carrier $Q = (A, K, T, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelPoissonSummability}^{\uparrow}}$ *surface consists exactly of Abel summability admission through A, Poisson-kernel admission through K, Abel theorem radial-boundary admission through T, dyadic radial tolerance accounting through D, finite radius and angle-window exhaustion through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal ordering through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8122.1. The tuple displays exactly the Abel summability row, Poisson-kernel row, Abel theorem row, dyadic radial ledger, finite windows, regular rational readback, Real seal, and structural support rows.

The analytic data remain finite. Rows A, K, T, D, W name the summability, kernel, radial-boundary, tolerance, radius, coefficient, angle, and kernel-value windows that the packet may inspect, while R repacks those displayed rows as regular rational data.

The terminal seal row E is downstream from R. Transport through H and replay through C preserve the same displayed route, and P, N only source and name it. Since no further coordinate exists in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8122.3 (Abel–Poisson summability Real-seal route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{AbelPoissonSummability}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{AbelSummability}^{\uparrow}, \quad \mathsf{PoissonKernel}^{\uparrow}, \quad \mathsf{AbelTheorem}^{\uparrow}, \quad D, W, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the same packet rows $A, K, T, D, W, R, E, H, C, P, N$. Thus the Abel–Poisson handoff is a displayed radial-window readback and not a host boundary limit, selected harmonic extension, quotient series equality, countable-choice selector, or ambient Real-completeness principle.

Proof. Begin with an accepted carrier and project the route rows A, K, T, D, W, R, E. The rows A and T supply the Abel summability and radial theorem faces, while K supplies the positive Poisson-kernel comparison surface over the same displayed radius requests.

Read the finite radial ledger through D and W. These rows fix the dyadic tolerances, radius parameters, coefficient windows, angle windows, and kernel-value windows that determine the regular rational readback R.

Finally pass through E, H, C, P, N. The seal row receives its analytic input from R, and the structural rows transport, replay, source, and name that same path. Therefore a downstream Real or Completion consumer obtains only the displayed finite handoff. □

Falsifiable prediction: An accepted $\mathsf{AbelPoissonSummability}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the $\mathsf{AbelSummability}^{\uparrow}$ row, $\mathsf{PoissonKernel}^{\uparrow}$ row, $\mathsf{AbelTheorem}^{\uparrow}$ row, dyadic radial ledger, finite radius windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: abel-summability, poisson-kernel, abel-theorem, regseqrat, real: AbelPoissonSummability is not bijective to any listed sibling because it binds Abel radial summability, Poisson kernel data, dyadic radius ledgers, regular readback, and Real sealing into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelPoissonSummability}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelPoissonSummability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8122.1, Definition 8122.2, and Definition 8122.3 seed the finite Abel–Poisson summability route over Abel summability rows, Poisson-kernel windows, Abel theorem radial boundaries, regular rational readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelPoissonSummability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host boundary limits, harmonic-function equality, selected sums, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Abel summability admission, Poisson-kernel admission, Abel theorem radial-boundary admission, dyadic radial tolerance accounting, finite radius-window exhaustion, regular readback, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.AbelPoissonSummabilityUp.

Constructive story: An $\mathsf{AbelPoissonSummability}^{\uparrow}$ packet is generated from Abel summability data, Poisson-kernel data, Abel theorem radial-boundary data, dyadic radial tolerances, finite radius and angle windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8123

A Concrete Naming Certificate for $\mathsf{StepIndexedTrace}^{\uparrow}$

The $\mathsf{StepIndexedTrace}^{\uparrow}$ packet realizes the total-host/substrate bridge of chapter 9370 as one finite trace object. It sits beside the total-host boundary of chapter 964, the bounded MetaCIC replay of chapter 3889, the bounded universal trace of chapter 944, the cellular substrate of chapter 934, and the MetaCIC strong-normalization boundary of chapter 3954. Those siblings expose totality, bounded replay, universal finite traces, local cellular rows, or normalization boundaries separately; this packet binds fuel, prefix, endpoint, refusal, replay, provenance, and local naming as one NameCert trace.

Definition 8123.1 (Step-indexed trace carrier)**.** *A* $\mathsf{StepIndexedTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, S, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite fuel row read by the total host.*
2. *S is the substrate row, either a* $\mathsf{CellularAutomaton}^{\uparrow}$ *local rule/orbit source or a MetaCIC bounded evaluator source.*
3. *Q is the finite prefix row containing the trace segment produced at fuel F.*
4. *E is the endpoint row reached at the end of the prefix, including normal-form or timeout readback when such a row is displayed.*
5. *R is the refusal row for every read that asks for global halting, an infinite orbit, a fuel-erased endpoint, a hidden normalizer, or an unbounded substrate theorem.*
6. *H records componentwise* hsame *transport for F, S, Q, E, R and the structural rows.*
7. *C records displayed* Cont *replay from fuel to substrate source, from source to prefix, from prefix to endpoint, and from absent global claims to the refusal row.*
8. *P is the* Pkg *provenance over* $\mathsf{StepIndexedTrace}^{\uparrow}$*,* $\mathsf{StepIndexedTotalHost}^{\uparrow}$*,* $\mathsf{MetaCICFuelTrace}^{\uparrow}$*,* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StepIndexedTrace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, Q, E, R, H, C, P, N$. It has no coordinate for fuel-free halting, an all-time cellular orbit, unrestricted MetaCIC normalization, host evaluator equality, substrate theorem content, or quotient equality of traces.

Theorem 8123.2 (Step-indexed trace NameCert obligations)**.** *For every accepted* $\mathsf{StepIndexedTrace}^{\uparrow}$ *carrier*

$$T = (F, S, Q, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StepIndexedTrace}^{\uparrow}}$ *surface consists exactly of fuel exposure through F, substrate-source exposure through S, finite prefix exposure through Q, endpoint readback through E, refusal exactness through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The fuel row F is the first public coordinate, and the substrate source S is only read at that displayed fuel.

The trace body is Q, E. The prefix row Q records the finite segment, and the endpoint row E records the finite readback reached by that segment.

The refusal row R is part of the same carrier, not an external convention. It rejects global halting, infinite orbit, fuel-erased endpoint, hidden normalizer, and unbounded theorem reads.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no additional coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 8123.3 (Total-host substrate handoff)**.** *Every total-host or substrate consumer of an accepted* $\mathsf{StepIndexedTrace}^{\uparrow}$ *packet must read the route*

$$\mathsf{StepIndexedTotalHost}^{\uparrow} \longmapsto F \longmapsto S \longmapsto Q \longmapsto E \longmapsto R \longmapsto H, C, P, N.$$

The route may pass through $\mathsf{MetaCICFuelTrace}^{\uparrow}$*,* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*, and* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *as sibling evidence, but no consumer may erase F and treat the prefix or endpoint as a global halting read.*

Proof. Start with the total-host boundary. A host consumer reaches this packet only through the fuel discipline already exposed by $\mathsf{StepIndexedTotalHost}^{\uparrow}$, so every trace read begins with F.

Read the substrate side next. The row S identifies the finite cellular or MetaCIC source, Q records the prefix generated at the displayed fuel, and E records the endpoint visible at that prefix.

The sibling rows provide comparison evidence without replacing the packet. $\mathsf{MetaCICFuelTrace}^{\uparrow}$ accounts for bounded evaluator replay, $\mathsf{BoundedUniversalTrace}^{\uparrow}$ accounts for universal finite prefixes, $\mathsf{CellularAutomaton}^{\uparrow}$ accounts for local cellular rows, and $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ accounts for the normal-form boundary.

The refusal row R then blocks the fuel-erased read. Transport, replay, provenance, and naming preserve the same packet, so no consumer can convert Q or E into global halting or unbounded substrate content. □

Falsifiable prediction: Any accepted $\mathsf{StepIndexedTrace}^{\uparrow}$ packet that admits a global halting read without displaying fuel and prefix rows is rejected by the NameCert surface.

Independence witness: stepindexedtotalhost, metacic-fuel-trace, bounded-universal-trace, cellularautomaton, metacic-strong-normalization-boundary: none is bijective to $\mathsf{StepIndexedTrace}^{\uparrow}$ because this carrier binds the finite fuel, substrate source, prefix, endpoint, refusal, transport, replay, provenance, and local naming rows in one packet.

The Lean follow-up namespace BEDC.Derived.StepIndexedTraceUp must provide Nontrivial StepIndexedTrace$^{\uparrow}$ and FieldFaithful StepIndexedTrace$^{\uparrow}$ instances for the displayed fuel, source, prefix, endpoint, refusal, transport, replay, provenance, and naming rows.

Closure status for NameCert$_{\text{StepIndexedTrace}^{\uparrow}}$.

Constructive story: A StepIndexedTrace$^{\uparrow}$ packet is generated from finite fuel, a substrate source, a finite trace prefix, endpoint readback, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{StepIndexedTrace}^{\uparrow}}$, seed)

Scope closed: Definition 8123.1, Definition 8123.2, and Definition 8123.3 seed the finite trace packet for total-host and substrate consumers over StepIndexedTotalHost$^{\uparrow}$, MetaCICFuelTrace$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, and MetaCICStrongNormalizationBoundary$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{StepIndexedTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global halting, all-time cellular orbits, unrestricted MetaCIC normalization, host evaluator equality, substrate theorem content, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fuel-prefix endpoint exposure, refusal exactness, transport and replay stability, Nontrivial and FieldFaithful instances, and local NameCert provenance for BEDC.Derived.StepIndexedTraceUp.

Chapter 8124

A Concrete Naming Certificate for $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$

Following Kurzweil, Henstock, and Stieltjes, the $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ packet records a finite gauge-controlled Stieltjes integration surface in which a displayed integrand is read against a bounded-variation integrator through tagged gauges and regulated handoff rows. It sits after the Riemann–Stieltjes packet of chapter 1637, the gauge-integral surface of chapter 1999, the bounded-variation ledger of chapter 398, the regulated-integral handoff of chapter 1557, and the terminal real seal of chapter 38. The packet names only a finite gauge-variation carrier; it does not assert measure completion, host quotient equality of integrands, a selected primitive, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8124.1 (Kurzweil–Stieltjes integral carrier)**.** *A* $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, R, A, T, I, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *row exposing the finite gauge cover, tagged cells, and admissible local radii.*
2. *R is the* $\mathsf{RiemannStieltjes}^{\uparrow}$ *row supplying the tagged Stieltjes sum surface already accepted by the Riemann–Stieltjes certificate.*
3. *A is the* $\mathsf{BoundedVariation}^{\uparrow}$ *integrator ledger, including transported endpoint jumps and finite variation bounds for the displayed cells.*
4. *T is the gauge-tagged refinement row tying the cells of G to the Stieltjes tags read by R.*
5. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row reached only after the gauge, Stieltjes, variation, and refinement rows are present.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the accepted integral value.*
7. *H records componentwise* hsame *transport for the gauge, Stieltjes, variation, refinement, handoff, real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from gauge cells to tags, from bounded-variation jumps to Stieltjes sums, from refinement rows to the regulated handoff, and from the handoff to the* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *P is the* Pkg *provenance over* $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{KurzweilStieltjesIntegral}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, R, A, T, I, E, H, C, P, N$. It has no coordinate for measure spaces, completed variation objects, host quotient equality of functions, selected antiderivatives, or ambient real completeness.

Lean4 (statement) `BEDC.Derived.KurzweilStieltjesIntegralUp`

Theorem 8124.2 (Kurzweil–Stieltjes integral NameCert obligations)**.** *For an accepted* $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ *carrier $K = (G, R, A, T, I, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KurzweilStieltjesIntegral}^{\uparrow}}$ *surface consists exactly of gauge admission, accepted Riemann–Stieltjes readback, bounded-variation integrator control, gauge-tagged refinement, regulated-integral handoff, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed continuation replay, package provenance, and local naming through $G, R, A, T, I, E, H, C, P, N$.*

Proof. Project the carrier of Definition 8124.1. The rows $G, R, A, T, I, E, H, C, P, N$ are the complete public packet, so every accepted read is first a finite gauge read, then a Riemann–Stieltjes tagged-sum read, then a bounded-variation and refinement read.

The regulated handoff row I is downstream of those rows. Transport through H and replay through C preserve the displayed order, while P and N only record provenance and local naming. Since the carrier contains no measure, quotient, selected primitive, or ambient completeness coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 8124.3 (Kurzweil–Stieltjes gauge-variation handoff)**.** *Every accepted* $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ *consumer of a terminal* $\mathsf{Real}^{\uparrow}$ *value must read the route*

$$\mathsf{GaugeIntegral}^{\uparrow},\ \mathsf{RiemannStieltjes}^{\uparrow},\ \mathsf{BoundedVariation}^{\uparrow},\ \mathsf{RegulatedIntegral}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

through $G, R, A, T, I, E, H, C, P, N$ before it treats the output as an integral value. The route consumes existing gauge and bounded-variation rows and does not assert measure completion, host quotient equality, selected primitives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (G, R, A, T, I, E, H, C, P, N)$. The gauge row G supplies the only admissible local cells, and the Riemann–Stieltjes row R supplies the only tagged Stieltjes sum surface.

The bounded-variation row A controls the integrator contribution on the same displayed cells. The refinement row T ties the gauge tags to the Stieltjes tags, so a downstream consumer cannot bypass either the gauge surface or the variation ledger.

The handoff row I then exposes the regulated-integral route, and the terminal seal E is available only after that handoff. Structural transport, continuation replay, provenance, and local naming preserve the same order through H, C, P, N, leaving no coordinate for the excluded host objects. □

Falsifiable prediction: Every accepted $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ read is refuted if a consumer reaches a $\mathsf{Real}^{\uparrow}$ integral value without displaying the gauge row, accepted Riemann–Stieltjes row, bounded-variation ledger, regulated-integral handoff, and local NameCert row.

Independence witness: Riemann–Stieltjes, gauge-integral, bounded-variation, regulated-integral, and Real packets: the Riemann–Stieltjes packet records tagged Stieltjes sums without owning a gauge cell ledger, the gauge-integral packet records local gauge admission without Stieltjes variation control, the bounded-variation packet records integrator variation without naming an integral value, the regulated-integral packet records a handoff surface without a Stieltjes integrator, and the Real packet supplies only the terminal seal. None is bijective to $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{KurzweilStieltjesIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{KurzweilStieltjesIntegral}^{\uparrow}$ packet is generated from finite gauge rows, accepted Riemann–Stieltjes rows, bounded-variation integrator ledgers, gauge-tagged refinement rows, regulated-integral handoff rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KurzweilStieltjesIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8124.1, Definition 8124.2, and Definition 8124.3 seed the finite gauge-variation Stieltjes readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KurzweilStieltjesIntegral}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.KurzweilStieltjesIntegralUp`

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, measure completion, host quotient equality of integrands or integrators, selected primitives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked gauge admission, accepted Riemann–Stieltjes readback, bounded-variation control, regulated handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.KurzweilStieltjesIntegralUp

Chapter 8125

A Concrete Naming Certificate for $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$

Following Bishop and Bridges on located real numbers and finite real-name comparisons, the $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$ packet records the binary separation surface read before a later $\mathsf{RealNameClassifier}^{\uparrow}$ decision is exposed. It sits over the dyadic rational core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the classifier packet of chapter 1517. The packet names only the finite located separator rows by which two regular real names are inspected; it is not a proof of full real trichotomy, quotient-real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8125.1 (Located real binary separator carrier)**.** *A* $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W, T, R, D, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two displayed real-name source rows whose located separation is being inspected.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the requested observation stages for both sources.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that serves the displayed rational observations on the same window.*
5. *D is the finite binary decision row, recording only the accepted left/right separator outcome read from W, T, R after the terminal* $\mathsf{Real}^{\uparrow}$ *seal row is available.*
6. *H records componentwise* hsame *transport for source rows, windows, dyadic tolerance rows, regular readbacks, terminal real sealing, binary decision rows, provenance, replay, and local naming.*
7. *C records displayed* Cont *replay from a separator request through A, B, W, T, R, D and back to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealNameClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealBinarySeparator}^{\uparrow}}$ *row.*

The carrier exports only this finite located binary separator surface. It has no coordinate for quotient real equality, host equality of completed reals, selected representatives, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8125.2 (Located real separator dyadic decision)**.** *For every accepted* $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$ *carrier*

$$S = (A, B, W, T, R, D, H, C, P, N),$$

the binary decision row D *can be read only after the shared* $\mathsf{StreamName}^{\uparrow}$ *row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route have been displayed. Thus a later* $\mathsf{RealNameClassifier}^{\uparrow}$ *consumer receives a finite decision surface, not an ambient equality or order oracle on completed real numbers.*

Proof. Project the carrier of Definition 8125.1. The source rows A, B are inspected through the common finite window W, so a separator request has no independent host-level observation route.

Read the dyadic and regular rows next. The tolerance ledger T supplies the displayed dyadic comparison data, and R exposes the regular rational readback on the same finite window.

Only after that route is visible may the decision row D be consumed. The sibling chapters provide the surfaces in the order $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealNameClassifier}^{\uparrow}$, so the classifier-facing decision is a finite packet row.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient-real equality, selected representative, selected limit, trichotomy, countable-choice schedule, or ambient completeness coordinate, none of those principles is exported. □

Theorem 8125.3 (Located real separator NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$ *carrier*

$$S = (A, B, W, T, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealBinarySeparator}^{\uparrow}}$ *surface consists exactly of source-row admission through* A, B*, shared finite-window exposure through* W*, dyadic tolerance exposure through* T*, regular rational readback through* R*, finite binary decision exposure through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the tuple in Definition 8125.1. Its analytic rows are precisely A, B, W, T, R, D, and its structural rows are precisely H, C, P, N.

Apply the dyadic decision theorem. It orders the decision-facing read through the shared window, dyadic ledger, regular readback, terminal seal route, and only then the finite binary decision row.

The remaining rows account for transport, replay, provenance, and local naming. No additional carrier coordinate remains from which a consumer could recover full trichotomy, quotient real equality, selected representatives, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealBinarySeparator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealBinarySeparator}^{\uparrow}$ packet is generated from two displayed real-name source rows, one shared StreamNameUp window, a DyadicRatCoreUp tolerance ledger, a RegSeqRatUp readback row, a finite binary decision row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealBinarySeparator}^{\uparrow}}$, seed)

Scope closed: Definition 8125.1, Definition 8125.2, and Definition 8125.3 seed the finite binary separator route consumed by RealNameClassifier-facing reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealBinarySeparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full real trichotomy, quotient reals, selected representatives, selected limits, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exposure, dyadic tolerance exposure, regular rational readback, terminal seal routing, decision-row exposure, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealBinarySeparatorUp.

Chapter 8126

A Concrete Naming Certificate for $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$

The $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as a finite residual-overlap matrix for typed MetaCIC redexes. It sits beside the critical-path packet of chapter 765, the parallel-diamond frontier of chapter 4238, the typed normal-form readback surface of chapter 562, the subject-reduction boundary of chapter 483, and the residual-diamond frontier of chapter 8111. The packet records only finite typed-redex, residual-row, join-candidate, closedness, substitution-boundary, transport, replay, provenance, and local naming data; it is not a broad confluence assertion.

Falsifiable prediction: Every accepted $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ consumer must expose both residual rows and one join-candidate row before a parallel-diamond read is available.

Independence witness: metacic-parallel-diamond-frontier, metaciccriticalpath, typedreduction-normalform, subjectreduction: the parallel-diamond frontier records a conditional diamond handoff without this packet's typed overlap matrix; the critical-path packet records routing pressure without residual-row adjacency; the typed normal-form packet records endpoint readback without paired residual overlaps; and the subject-reduction packet records preservation boundaries without candidate joins. None is bijective to $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$.

Definition 8126.1 (MetaCIC residual join frontier carrier)**.** *A* $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, L, R, J, K, S, O, H, C, P, N)$$

with the following displayed rows:

1. *Q is the typed critical-pair row, recording the source term, typing boundary, redex labels, and inspected overlap window.*
2. *L and R are the two residual rows, one for each branch descending from the overlap.*
3. *J is the join-candidate row, recording the proposed common target and the two branch-local routes to it.*
4. *K is the closedness and finite-support row for the inspected source and residual descendants.*
5. *S is the substitution-boundary row retaining the residual-substitution premise and any obstruction that remains outside the packet.*

6. *O is the parallel-diamond read row, exposed only after Q, L, R, J, K, S have all been displayed.*
7. *H records componentwise* hsame *transport for typed critical pairs, residual rows, join candidates, closedness rows, substitution-boundary rows, parallel-diamond reads, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence query through Q, L, R, J, K, S before O is read.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}}$ *row.*

The carrier exports no coordinate for premise-free confluence, unrestricted subject reduction, global normalization, evaluator equality, quotient syntax equality, or kernel consistency.

Theorem 8126.2 (MetaCIC residual join frontier diamond consumer)**.** *Every parallel-diamond consumer of an accepted* $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ *packet first factors through the typed critical-pair row Q, both residual rows L, R, one join-candidate row J, the closedness row K, and the substitution-boundary row S before the parallel-diamond read row O is exposed. The resulting read is a finite residual-join consumer for the confluence audit map, not a proof of unrestricted MetaCIC confluence.*

Proof. Project the carrier of Definition 8126.1. The parallel-diamond read row O is downstream of Q, L, R, J, K, S by construction.

Read the overlap rows first. The typed critical-pair row Q fixes the source, typing boundary, redex labels, and inspected overlap window; the two residual rows L, R record the branch-local descendants; and J supplies the candidate common target.

Next read the boundary rows. The closedness row K keeps the source and residual descendants inside the finite inspected support, while S retains the substitution premise or obstruction rather than discharging it silently.

Finally use H, C, P, N for transport, replay, provenance, and local naming. These structural rows preserve the same finite matrix and add no premise-free confluence theorem, unrestricted subject reduction theorem, global normalization oracle, evaluator equality, quotient syntax equality, or kernel-consistency claim. □

Theorem 8126.3 (MetaCIC residual join frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ *carrier*

$$M = (Q, L, R, J, K, S, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}}$ *surface consists exactly of typed critical-pair exposure through Q, two residual-row exposures through L, R, join-candidate exposure through J, closedness and finite-support exposure through K, substitution-boundary retention through S, parallel-diamond read exposure through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the tuple in Definition 8126.1. The carrier lists exactly the matrix rows Q, L, R, J, K, S, O and the structural rows H, C, P, N.

Apply the diamond-consumer theorem. It forces every confluence-facing read through the typed pair, both residual rows, a join candidate, closedness data, and the retained substitution boundary before the parallel-diamond row can be used.

The structural rows account for all remaining obligations. Since the carrier has no other coordinate, its local NameCert cannot export unrestricted confluence, subject reduction, global normalization, evaluator equality, quotient syntax equality, or kernel consistency. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}$ packet is generated from a typed critical-pair row, two residual rows, one join-candidate row, closedness and finite-support data, a retained substitution-boundary row, a parallel-diamond read row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8126.1, Definition 8126.2, and Definition 8126.3 seed the finite residual-overlap matrix consumed by MetaCIC parallel-diamond reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted MetaCIC confluence, premise-free parallel diamond, unrestricted subject reduction, global normalization, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed critical-pair exposure, two residual-row exposures, join-candidate exposure, closedness exposure, substitution-boundary retention, diamond-read non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICCriticalPairResidualMatrixUp.

Chapter 8127

A Concrete Naming Certificate for $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$

Following Hardy and Bishop–Bridges on Cauchy's condensation test, the $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$ packet records the harmonic-block face of a monotone real-series comparison as finite BEDC NameCert data. It sits after the condensation-test surface of chapter 1478, the real-series carrier of chapter 1581, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives downstream condensation and tail-modulus consumers a displayed block route; it is not a selected infinite sum, a quotient series equality, a countable-choice schedule, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8127.1 (Cauchy condensation harmonic block carrier)**.** *A* $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, B, D, T, R, E, Hs, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing the monotone nonnegative term stream and partial-sum windows only through the series packet.*
2. *B is the harmonic block schedule, listing the dyadic block indices, the harmonic weights inside each block, and the finite block endpoints used by the condensation read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic precision cells, comparison slack, and block-budget conversions used by B.*
4. *T is the tail comparison ledger, recording the finite inequalities between source tails, harmonic block sums, and condensed comparison rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after S, B, D, T have displayed the block schedule and tail ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback is displayed.*
7. *Hs records componentwise* hsame *transport for the source, block schedule, tolerance, tail ledger, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a harmonic-block request through $S, B, D, T, R, E, Hs, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a host infinite sum, quotient series equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8127.2 (Cauchy condensation harmonic block obligations)**.** *For every accepted* CauchyCondensationHarm *carrier $H = (S, B, D, T, R, E, Hs, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}}$ *surface consists exactly of carrier admission, harmonic block schedule exactness, dyadic tolerance exactness, tail comparison ledger exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8127.1. Carrier admission is the display of $S, B, D, T, R, E, Hs, C, P, N$ as one finite BHist packet.

Read the block rows next. The schedule B lists the dyadic block indices, harmonic weights, and endpoints, while D supplies the dyadic precision cells and comparison slack used to certify those finite blocks.

The comparison row T then records only displayed inequalities between the source tails, harmonic block sums, and condensed comparison rows. The readback row R is downstream of this ledger, and the seal row E is downstream of R.

Finally Hs, C, P, N transport, replay, source, and locally name the same route. Since no carrier coordinate names selected infinite sums, quotient series equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8127.3 (Cauchy condensation harmonic block Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing harmonic-block read of an accepted* $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{CauchyCondensationTest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports the displayed source series, condensation block schedule, dyadic tolerance ledger, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows, but no selected infinite sum, quotient series equality, countable-choice schedule, detached Real seal, or ambient completeness principle.

Proof. Start with the source row S. It exposes the real-series terms and partial-sum windows before any block consumer can inspect a harmonic schedule.

The condensation-facing read then passes through B, T and the sibling condensation-test surface. The block schedule and tail comparison ledger are the only rows that connect the source series to condensed block data.

Next read the dyadic and regular rational boundary. The row D supplies the tolerance budget before R can be consumed as a $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal seal E is downstream of R.

The structural rows Hs, C, P, N preserve this order under transport, replay, provenance, and local naming. Thus every Real-facing read factors through the stated finite route and has no coordinate for selected infinite sums, quotient equality, countable choice, detached Real data, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$ Real-facing read factors through a RealSeries source row, CauchyCondensationTest block schedule, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local NameCert row.

Independence witness: realseries, cauchy-condensation-test, dyadicratcore, regseqrat, real: no listed sibling supplies the whole harmonic-block packet $H = (S, B, D, T, R, E, Hs, C, P, N)$, because each sibling contributes only the source series, condensation-test block surface, dyadic tolerance, regular readback, or endpoint seal while this chapter binds them as one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}$ packet is generated from a RealSeries source row, harmonic block schedule, DyadicRatCore tolerance row, tail comparison ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8127.1, Definition 8127.2, and Definition 8127.3 seed the finite harmonic-block condensation route from RealSeries data through dyadic block scheduling, regular rational readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationHarmonicBlock}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient series equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, block schedule exactness, dyadic tolerance exactness, tail comparison ledger exposure, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationHarmonicBlockUp.

Chapter 8128

A Concrete Naming Certificate for $\mathsf{SchauderBasisProjection}^{\uparrow}$

Following Schauder and Banach basis theory, the $\mathsf{SchauderBasisProjection}^{\uparrow}$ packet records finite coordinate projections licensed by displayed Schauder basis data. It sits after the basis packet of chapter 3669, the bounded coordinate readback of chapter 7494, the vector-space carrier of chapter 47, the norm row of chapter 89, the Banach completion surface of chapter 90, the regular rational scalar route of chapter 301, and the terminal real-name seal of chapter 38. The certificate names the projection window and its norm-controlled finite sum without invoking Hamel bases, arbitrary dual vectors, Hahn–Banach extension, quotient functionals, or choice.

Definition 8128.1 (Schauder basis projection carrier)**.** *A* $\mathsf{SchauderBasisProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (B, I, Q, C, V, N, R, H, T, S, L)$$

with the following displayed rows:

1. *B is the* $\mathsf{SchauderBasis}^{\uparrow}$ *source row naming the basis-vector stream, partial-sum route, coordinate requests, tail estimates, and reconstruction data.*
2. *I is the finite index-window row, recording the initial segment or finite coordinate set on which the projection is read.*
3. *Q is the* $\mathsf{CoordinateFunctional}^{\uparrow}$ *row family for the displayed coordinate reads licensed by B and indexed by I.*
4. *C is the* $\mathsf{RegSeqRat}^{\uparrow}$ *coefficient row, naming the rational scalar approximants used by the coordinate reads in Q.*
5. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming finite linear-combination syntax, vector addition, scalar action, and the zero vector used to assemble the projected vector.*
6. *N is the* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Banach}^{\uparrow}$ *budget row, recording the bound that treats the finite projection as a norm-controlled operation on the displayed Banach-space carrier.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the projection norm estimate, tail estimate, and completion-facing tolerance.*
8. *H records componentwise* hsame *transport for the Schauder source, index window, coordinate functional rows, coefficient names, vector operations, norm and Banach budget, real readback, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a projection request through B, I, Q, C, V, N, R.*
10. *S records* Pkg *provenance over* $\mathsf{SchauderBasisProjection}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{CoordinateFunctional}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{SchauderBasisProjection}^{\uparrow}}$ *row.*

The classifier compares packets only through $B, I, Q, C, V, N, R, H, T, S, L$. It has no coordinate for a Hamel basis selector, an arbitrary dual-space element, a Hahn–Banach extension, a quotient functional, countable choice, unconditional convergence, or a projection outside the displayed finite window.

Theorem 8128.2 (Schauder basis projection NameCert obligations)**.** *For every accepted* $\mathsf{SchauderBasisProjection}^{\uparrow}$ *carrier $P = (B, I, Q, C, V, N, R, H, T, S, L)$, the local* $\mathsf{NameCert}_{\mathsf{SchauderBasisProjection}^{\uparrow}}$ *surface consists exactly of carrier admission, Schauder basis source exposure through B, finite window exposure through I, coordinate-functional exposure through Q, regular rational coefficient exposure through C, vector assembly through V, norm and Banach budget exposure through N, real readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through S, and local naming through L.*

Proof. Carrier admission is projection from Definition 8128.1. The accepted packet exposes exactly the tuple $B, I, Q, C, V, N, R, H, T, S, L$.

The analytic route is fixed by B, I, Q, C, V, N, R. The Schauder row supplies the basis source, the index row supplies the finite projection window, the coordinate-functional rows supply the licensed coefficient reads, the regular rational row names the scalar approximants, the vector row assembles the finite linear combination, and the norm and real rows display the boundedness and tolerance data.

The structural rows H, T, S, L preserve that same route under transport, replay, provenance, and naming. Since no carrier coordinate stores a Hamel selector, arbitrary dual vector, Hahn–Banach extension, quotient functional, choice schedule, unconditional convergence proof, or off-window projection, none of those objects becomes a local NameCert obligation. □

Theorem 8128.3 (Schauder basis projection real completion handoff)**.** *Let $P = (B, I, Q, C, V, N, R, H, T, S, L)$ be an accepted* $\mathsf{SchauderBasisProjection}^{\uparrow}$ *carrier. A downstream Real or completion-facing consumer may read the finite projection only after the* $\mathsf{SchauderBasis}^{\uparrow}$ *source row B, the* $\mathsf{CoordinateFunctional}^{\uparrow}$ *rows Q, the coefficient names C, the vector assembly row V, the norm and Banach budget row N, and the real readback row R have all been exposed along the same transport and replay route. The handoff exports a bounded finite coordinate projection and its displayed tolerance data, but no Hamel basis, no arbitrary functional, no quotient functional, no Hahn–Banach extension, and no choice-selected projection.*

Proof. Begin with a projection request. The finite window I determines which coordinate rows in Q are read from the Schauder source B, and the regular rational row C names the scalar approximants for those reads.

Assemble the projected vector through V. The vector-space row supplies the finite linear-combination syntax for the selected basis vectors and coefficients, while N supplies the norm and Banach budget that lets the projection be treated as a bounded finite operation.

Expose the Real-facing estimate through R. The real readback row names the norm and tolerance data that a completion consumer receives, and the transport, replay, provenance, and local naming rows H, T, S, L keep that read tied to the same finite packet.

Therefore the completion-facing handoff is exactly the displayed route through $B, I, Q, C, V, N, R, H, T, S, L$. The omitted principles have no carrier coordinate, so they cannot be exported by the projection certificate. □

Falsifiable prediction: Every accepted $\mathsf{SchauderBasisProjection}^{\uparrow}$ projection read factors through a SchauderBasis source row, finite index window, CoordinateFunctional rows, RegSeqRat coefficient names, VecSpace finite linear-combination syntax, Norm and Banach budget rows, Real tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: schauder-basis, coordinate-functional, vecspace, norm, banach, regseqrat, real: no listed sibling supplies the whole packet $P = (B, I, Q, C, V, N, R, H, T, S, L)$, because the projection carrier binds a displayed Schauder source, finite window, coordinate reads, scalar names, vector assembly, norm budget, completion-facing Real tolerance, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderBasisProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderBasisProjection}^{\uparrow}$ packet is generated from a SchauderBasis source row, finite index-window row, CoordinateFunctional row family, RegSeqRat coefficient row, VecSpace finite linear-combination row, Norm and Banach budget row, Real readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderBasisProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8128.1, Definition 8128.2, and Definition 8128.3 seed the finite coordinate-projection route licensed by displayed Schauder basis and coordinate-functional data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderBasisProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hamel bases, arbitrary dual vectors, Hahn–Banach extension, quotient functionals, countable choice, unconditional convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, coordinate-readback exactness, vector assembly, norm-budget exposure, Real completion handoff, transport, replay, provenance, and local naming for the displayed Schauder basis projection packet.

Chapter 8129

A Concrete Naming Certificate for $\mathsf{SearchableSpace}^{\uparrow}$

Following Escardo on searchable spaces and selection functionals, the $\mathsf{SearchableSpace}^{\uparrow}$ packet records the finite route by which a displayed decidable predicate is inspected through a search schedule before a witness row or compact-search handoff is exposed. It sits after the stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the fan-functional selector surface of chapter 1525, the compact metric finite-net surface of chapter 129, the searchable compact sibling of chapter 6618, and the terminal real-name seal of chapter 38. The packet names searchability as finite NameCert data; it is not an ambient choice functional, not an open-cover compactness theorem, and not a selected real outside the displayed route.

Definition 8129.1 (Searchable space carrier)**.** *A* $\mathsf{SearchableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, Q, F, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the search schedule row, read through* $\mathsf{StreamName}^{\uparrow}$ *finite windows and bounded fan-functional branches when the source is fan-like.*
2. *Q is the predicate readback row, opening only the displayed decidable predicate observations requested by W.*
3. *F is the finite witness row, recording the retained candidate list, the accepted witness when one is displayed, and the refusal of a hidden global selector.*
4. *K is the compact-search handoff row, allowing* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SearchableCompact}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers to read the same finite search route.*
5. *H records componentwise* hsame *transport for W, Q, F, K and the structural rows.*
6. *C records displayed* Cont *replay from schedule to predicate readback, from predicate readback to finite witness, and from finite witness to compact-search handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{SearchableSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SearchableCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SearchableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through W, Q, F, K, H, C, P, N. It has no coordinate for a detached selection functional, an unbounded search over an ambient set, quotient equality of predicates, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8129.2 (Searchable space NameCert obligations)**.** *For every accepted* $\mathsf{SearchableSpace}^{\uparrow}$ *carrier*

$$S = (W, Q, F, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SearchableSpace}^{\uparrow}}$ *surface consists exactly of five obligation rows: schedule exposure through W, predicate readback through Q, finite witness or refusal exposure through F, compact-search handoff through K, and structural certification through H, C, P, N.*

Proof. Project the displayed carrier of Definition 8129.1. The schedule row W is the only source of search windows, so every predicate read is bounded by the displayed stream or fan-functional route.

Read the predicate and witness rows next. The predicate row Q exposes only the decidable observations requested by W, while F records either the retained finite witness boundary or the refusal of a hidden selector.

Now hand the finite search route to compact and real-facing consumers through K. This handoff may cite $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SearchableCompact}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, but it carries only the same schedule, predicate, and witness rows.

Finally H, C, P, N transport, replay, source, and name the packet. Since the carrier has no coordinate for detached selection, unbounded ambient search, quotient predicate equality, open-cover compactness, countable choice, or ambient real completeness, the five listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{SearchableSpace}^{\uparrow}$ consumer read must display the search schedule, predicate readback, finite witness row, compact-search handoff, transport, replay, provenance, and local naming before consuming a selected witness.

Independence witness: streamname, regseqrat, dyadicratcore, fanfunctional, compactmetric, searchable-compact, real: none is bijective to $\mathsf{SearchableSpace}^{\uparrow}$ because this carrier binds predicate readback and finite witness exposure to the same search schedule, compact handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SearchableSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SearchableSpace}^{\uparrow}$ packet is generated from a search schedule, predicate readback rows, finite witness or refusal rows, compact-search handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SearchableSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8129.1 and Definition 8129.2 seed the finite searchable-space route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SearchableCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SearchableSpace}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.SearchableSpaceUp`

Bridge status: none

Not claimed: This chapter does not close detached selection functionals, unbounded ambient search, quotient equality of predicates, open-cover compactness, countable choice, Lean verification of the NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-bounded predicate readback, finite witness exactness, compact-search handoff, transport and replay stability, and local NameCert provenance for BEDC.Derived.SearchableSpaceUp.

Chapter 8130

A Concrete Naming Certificate for $\mathsf{BishopUniformGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopUniformGauge}^{\uparrow}$ packet records the finite gauge-radius object used to pass from Bishop uniform-structure data to finite cover and tagged-partition consumers. It sits after the dyadic arithmetic surface of chapter 291, the terminal real-name surface of chapter 38, the real uniform-structure surface of chapter 1624, the Cousin cover source of chapter 3426, and the tagged-partition consumer of chapter 7449. The packet names a finite radius gauge carrier only; it is not an open-cover compactness theorem, a selected limiting process, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8130.1 (Bishop uniform gauge carrier)**.** *A* $\mathsf{BishopUniformGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, R, U, Q, F, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the dyadic radius codes and comparison tests inspected by the gauge.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *source row, exposing the real-name endpoints or centers on which the gauge is read.*
3. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *row, carrying entourage and uniform-neighborhood data in a Bishop-readable form.*
4. *Q is the local gauge-radius row. For each displayed source point and entourage request, it stores a positive dyadic radius history admitted by the uniform-structure row.*
5. *F is the* $\mathsf{CousinCover}^{\uparrow}$ *handoff row, reading the displayed gauge radii as local cover cells without turning them into an open-cover compactness proof.*
6. *T is the* $\mathsf{TaggedPartition}^{\uparrow}$ *consumer row, available only after the finite gauge cells have been named.*
7. *H records componentwise* hsame *transport for dyadic tolerances, real source rows, uniform-structure rows, gauge radii, Cousin cover handoff, tagged-partition consumption, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an entourage request through dyadic radius choice, Cousin cover handoff, and tagged-partition consumption.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopUniformGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{TaggedPartition}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformGauge}^{\uparrow}}$ *row.*

The classifier compares accepted gauges only through $D, R, U, Q, F, T, H, C, P, N$. It has no coordinate for host open-cover compactness, selected limits, countable choice, quotient cover equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8130.2 (Bishop uniform gauge NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformGauge}^{\uparrow}$ *carrier*

$$G = (D, R, U, Q, F, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopUniformGauge}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission through D, real source exposure through R, Bishop uniform-structure admission through U, displayed positive gauge-radius exactness through Q, Cousin cover handoff through F, tagged-partition consumption through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation is discharged by host open-cover compactness, selected limits, countable choice, quotient cover equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8130.1. The tuple displays the dyadic tolerance row, real source row, uniform-structure row, gauge-radius row, Cousin cover handoff, tagged-partition consumer, transport, replay, provenance, and local naming rows.

The gauge-radius row is downstream of the uniform-structure row. The row D supplies dyadic radius codes, R supplies the finite real-name surface being inspected, and U supplies the Bishop uniform-neighborhood request.

The row Q then records the positive radius histories admitted for that displayed request. Because Q is a finite carrier row, it does not select a limiting process, quantify over an ambient open cover, or call a countable-choice schedule.

Finally F, T, H, C, P, N export, consume, transport, replay, source, and name exactly the same finite route. Since the excluded compactness, choice, quotient, limit, and completeness objects are absent from the carrier, they cannot appear in the local NameCert obligations. □

Theorem 8130.3 (Bishop uniform gauge finite-cover route)**.** *Every* $\mathsf{CousinCover}^{\uparrow}$ *or* $\mathsf{TaggedPartition}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopUniformGauge}^{\uparrow}$ *packet reads gauge data only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{RealUniformStructure}^{\uparrow},\ Q,\ \mathsf{CousinCover}^{\uparrow},\ \mathsf{TaggedPartition}^{\uparrow}.$$

The route exports finite Bishop gauge-radius data for cover and partition consumers; it exports no open-cover compactness proof, selected limiting process, countable-choice selector, quotient cover equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the accepted carrier $G = (D, R, U, Q, F, T, H, C, P, N)$. A cover-facing read must expose the dyadic tolerance row D, the real source row R, and the Bishop uniform-structure row U before any local gauge radius is visible.

Use the gauge-radius row Q. This row stores the displayed positive radius histories for the finite source surface, so a downstream consumer cannot replace the route by an unrecorded open-cover compactness argument or an external selector.

The Cousin cover row F reads those radii as local cover cells, and the tagged-partition row T is downstream of that handoff. Componentwise transport and replay through H, C preserve the same order, while P, N source and name it.

Thus every cover or partition read factors through the finite Bishop uniform-gauge route. The excluded compactness, limit, choice, quotient, and completeness principles have no carrier row from which to enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformGauge}^{\uparrow}$ packet is generated from dyadic tolerance rows, Real source rows, RealUniformStructure rows, positive gauge-radius rows, CousinCover handoff, TaggedPartition consumption, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8130.1, Definition 8130.2, and Definition 8130.3 seed the finite Bishop uniform-gauge route for cover and tagged-partition consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, selected limiting processes, countable choice, quotient cover equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, Real source exposure, RealUniformStructure admission, gauge-radius exactness, CousinCover handoff, TaggedPartition consumption, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformGaugeUp.

Chapter 8131

A Concrete Naming Certificate for $\mathsf{BishopCompletionReflector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionReflector}^{\uparrow}$ packet records the finite reflector data that sends a regular Cauchy presentation into its separated completion object. It sits after the regular rational sequence surface of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy completion criterion of chapter 1476. The packet names a finite completion reflector only; it does not import quotient completions, selected limits, choice sequences, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 8131.1 (BishopCompletionReflector carrier)**.** *A* $\mathsf{BishopCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, S, E, A, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the regular Cauchy input row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *finite windows and the Cauchy criterion route.*
2. *S is the separated identity row, recording when two regular Cauchy inputs have the same separated completion readback.*
3. *E is the completion seal row, exposing the terminal* $\mathsf{Real}^{\uparrow}$ *name reached from the regular Cauchy input.*
4. *A is the reflector arrow row, sending the accepted regular Cauchy input to the separated completion seal.*
5. *L is the finite ledger recording the order in which regular input, separated identity, completion seal, and reflector arrow are consumed.*
6. *H records componentwise* hsame *transport for Q, S, E, A, L, C, P, N.*
7. *C records displayed* Cont *replay for the same finite route.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionReflector}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionReflector}^{\uparrow}}$ *row.*

Two accepted carriers are classified together only by the displayed rows $Q, S, E, A, L, H, C, P, N$. The classifier has no coordinate for quotient real equality, a chosen completion point, countable choice, host completion functoriality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8131.2 (BishopCompletionReflector NameCert obligations)**.** *For every accepted* BishopCompletionReflec *carrier* $R = (Q, S, E, A, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionReflector}^{\uparrow}}$ *surface consists exactly of regular Cauchy input admission through* Q*, separated identity exposure through* S*, completion seal exposure through* E*, reflector arrow admission through* A*, finite ledger order through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. No obligation is supplied by quotient completion semantics, selected limits, countable choice, host functorial completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 8131.1. The tuple displays the regular input, separated identity, completion seal, reflector arrow, ledger, transport, replay, provenance, and local naming rows.

The local NameCert reads these rows in the order recorded by L. Regular Cauchy input is consumed through Q, separated identity through S, terminal completion naming through E, and the reflector arrow through A.

Transport and replay are supplied by H and C, while P and N record provenance and local naming. Since the carrier has no coordinate for quotient completion semantics, selected limits, countable choice, host functorial completions, or ambient Real completeness, the listed rows exhaust the local obligation surface. □

Theorem 8131.3 (BishopCompletionReflector universal arrow)**.** *Every accepted* $\mathsf{BishopCompletionReflector}^{\uparrow}$ *packet exports a finite universal-arrow route from regular Cauchy input to separated completion readback:*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The route states that a completion-facing consumer must read the regular Cauchy input, separated identity, completion seal, reflector arrow, ledger, transport, replay, provenance, and local naming rows before treating the reflected object as sealed. It does not assert a quotient universal property, choose a limit representative, or use countable choice.

Proof. Start from an accepted carrier $R = (Q, S, E, A, L, H, C, P, N)$. The source of the arrow is the regular Cauchy input row Q, and the target is the completion seal E read after the separated identity row S.

The reflector arrow row A records the finite handoff from Q through S to E. The ledger L fixes the order of this handoff, so the consumer cannot inspect the terminal seal before the regular input and separated identity rows have been displayed.

The structural rows H, C, P, N transport, replay, name, and package the same route. Because no accepted row stores quotient completion semantics, selected representatives, choice schedules, or host completeness, the universal arrow is exactly the finite BEDC reflector route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionReflector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCompletionReflector}^{\uparrow}$ packet is generated from regular Cauchy input, separated identity, completion seal, reflector arrow, finite ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionReflector}^{\uparrow}}$, seed)

Scope closed: Definition 8131.1, Definition 8131.2, and Definition 8131.3 record the seed reflector carrier, local NameCert obligation surface, and finite universal-arrow route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionReflector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion semantics, selected limits, countable choice, host functorial completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked regular Cauchy input admission, separated identity exposure, completion seal exposure, reflector arrow admission, ledger order, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionReflectorUp.

Chapter 8132

A Concrete Naming Certificate for $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ packet records a family-level Cauchy route whose members share one displayed finite control surface before any completion-facing consumer reads a Real seal. It sits between the subbasis generation of chapter 4232, the filter-limit basis of chapter 3862, the stream-window schedule of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real carrier of chapter 38. The packet names uniform Cauchy behavior across a displayed family; it is not a quotient filter, not a selected pointwise limit, and not an appeal to countable choice.

Definition 8132.1 (Uniformly Cauchy filter family carrier)**.** *A* $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, B, L, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite family index schedule naming the members currently visible to the certificate.*
2. *B is the* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *generation row, recording the subbasic filter requests shared by the visible family.*
3. *L is the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *row that exposes only the basis requests through which family convergence is tested.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to read the same request across the indexed family.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational handoff for the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the common dyadic thresholds and comparison rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *completion-facing seal row available after the shared subbasis, limit-basis, stream, regular-rational, and dyadic rows have been displayed.*
8. *H stores componentwise* hsame *transport for the family index, subbasis, limit-basis, stream, handoff, dyadic, seal, continuation, provenance, and naming rows.*
9. *C stores displayed* Cont *replay from the family index to the subbasis and limit-basis rows, from those rows to stream windows and regular-rational handoff, and from the dyadic tolerance ledger to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$, $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for quotient filters, selected pointwise limits, countable-choice enumeration of members, ambient completeness, or host real equality.

Theorem 8132.2 (Uniformly Cauchy filter family NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ *carrier* $U = (I, B, L, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilterFamily}}$ *surface consists exactly of five obligations:*

1. *family-index exposure through the finite schedule I;*
2. *shared subbasis generation through B and limit-basis testing through L;*
3. *uniform stream-window and regular-rational handoff through S and R;*
4. *common dyadic tolerance accounting through D before the Real seal row E is visible;*
5. *structural transport, continuation replay, provenance, and local naming through H, C, P, N.*

Proof. Project the displayed carrier of Definition 8132.1. The family index I is the only visible source of members, so every family read is bounded by the same finite schedule.

Read the subbasis and limit-basis rows next. The row B generates the shared Cauchy filter requests, while L records the basis tests through which those requests are observed. These rows connect the family route to $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ and $\mathsf{FilterLimitBasis}^{\uparrow}$ without adding a quotient filter.

Now pass to the stream and rational rows. The schedule S and handoff R make the same request visible through finite windows and regular rational data, and the dyadic ledger D supplies the common tolerance route before E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for selected pointwise limits, countable-choice enumeration, host equality, or ambient completeness, the five listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ consumer read must display the family index schedule, shared Cauchy subbasis, filter-limit basis, stream windows, regular rational handoff, dyadic tolerance ledger, Real seal row, transport, replay, provenance, and local naming before consuming a family-level completion route.

Independence witness: cauchy-filter-subbasis, filterlimitbasis, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ because this carrier binds a shared family index and uniform tolerance route to the same subbasis, limit-basis, stream, rational, dyadic, seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}$ packet is generated from a finite family index schedule, shared Cauchy filter subbasis, filter-limit basis tests, stream windows, regular rational handoff, dyadic tolerance ledger, Real seal row, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}}$, seed)

Scope closed: Definition 8132.1 and Definition 8132.2 seed the family-level uniform Cauchy route over $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilterFamily}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected pointwise limits, countable-choice enumeration, ambient completeness, host real equality, Lean verification of the NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared subbasis generation, limit-basis testing, uniform window handoff, dyadic tolerance accounting, transport and replay stability, and local NameCert provenance for BEDC.Derived.UniformlyCauchyFilterFamilyUp.

Chapter 8133

A Concrete Naming Certificate for $\mathsf{ModulusToRegularCauchy}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ packet records the finite route from an explicit Cauchy modulus schedule to a regular Cauchy rational name and then to the terminal real-name seal. It sits over the modulus schedule of chapter 292, the finite stream-window interface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet is a finite Real and Completion bridge from schedules to regular names; it does not import quotient equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8133.1 (Modulus-to-regular-Cauchy carrier)**.** *A* $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning every requested regular precision to a finite source-window request.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source observations named by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that checks the scheduled window against the requested dyadic error budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row assembled from S, W, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular readback row has been displayed.*
6. *H records componentwise* hsame *transport for the schedule, window, dyadic tolerance, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real and Completion consumer request through S, W, D, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusToRegularCauchy}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusToRegularCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a chosen completed limit, countable choice, a host Cauchy-completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8133.2 (Modulus-to-regular-Cauchy NameCert obligations)**.** *For every accepted* ModulusToRegularCauch *carrier*

$$M = (S, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusToRegularCauchy}^{\uparrow}}$ *surface consists exactly of Cauchy-modulus schedule admission through S, finite window exposure through W, dyadic tolerance checking through D, regular rational readback through R, terminal real-name sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8133.1. The schedule row S is the only coordinate that assigns requested precisions to finite source-window requests, and the window row W is the only coordinate that exposes the corresponding source observations.

The dyadic row D checks those observations against the requested tolerance. The regular rational row R is therefore downstream of S, W, D, and the terminal real-name seal E is downstream of R.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-equality coordinate, selected-limit coordinate, countable-choice schedule, host completion operator, or ambient completeness field, the listed rows exhaust the local NameCert surface. □

Theorem 8133.3 (Modulus-to-regular-Cauchy window induction)**.** *Every finite precision-window read exported by an accepted* $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ *packet is obtained by induction over the requests named by the* $\mathsf{CauchyModulus}^{\uparrow}$ *row S: each step exposes one* $\mathsf{StreamName}^{\uparrow}$ *window row, checks the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, extends the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then permits the* $\mathsf{Real}^{\uparrow}$ *seal row to be read. The induction exports no quotient stream equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Proceed by induction over the displayed finite request window. The empty window exposes no source observation and therefore has no regular readback or Real seal to export.

For a successor window, read the next request from the modulus schedule row S. The stream window row W supplies exactly the finite observations demanded by that request.

Now consume the dyadic tolerance row D. Once the scheduled observations meet the requested dyadic budget, the regular rational readback row R is extended for that finite stage.

Finally use H, C, P, N to transport, replay, source, and name the same stage before the terminal seal E is read. Because no excluded principle is a row of the carrier, the induction remains a finite schedule-to-regular-name construction rather than a quotient, selector, choice, or completeness argument. □

Falsifiable prediction: Every accepted $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ read exposes the CauchyModulus schedule row, StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a Real and Completion consumer may use the regular name.

Independence witness: cauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a modulus schedule, finite window observations, dyadic tolerance checks, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one finite bridge.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusToRegularCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ModulusToRegularCauchy}^{\uparrow}$ packet is generated from a CauchyModulus schedule row, a StreamName finite-window row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ModulusToRegularCauchy}\uparrow}$, seed)

Scope closed: Definition 8133.1, Definition 8133.2, and Definition 8133.3 seed the finite schedule-to-regular-Cauchy bridge for the L10 Real and Completion route.

Formal verification: FormalStatus(NameCert$_{\text{ModulusToRegularCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host Cauchy completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exposure, finite-window exactness, dyadic tolerance checking, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ModulusToRegular

Chapter 8134

A Concrete Naming Certificate for MetaCICResidualJoinMatrix$^{\uparrow}$

The MetaCICResidualJoinMatrix$^{\uparrow}$ packet realizes the residual-substitution conditional boundary of section 9378.3 as a finite matrix of retained premise rows, local residual branches, candidate joins, and audit consumers. It sits beside the parallel-diamond frontier of chapter 4238, the finite-window confluence packet of chapter 6027, the residual local-join packet of chapter 5126, the candidate-normalization confluence handoff of chapter 5547, and the confluence audit packet of chapter 735. The matrix is a concrete NameCert object for the confluence audit boundary; it is not an unconditional confluence theorem.

Definition 8134.1 (MetaCIC residual-join matrix carrier)**.** *A* MetaCICResidualJoinMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (P, F, L, R, K, S, A, O, H, C, G, N)$$

with the following displayed rows:

1. *P is the residual-substitution premise row retained by the confluence audit boundary.*
2. *F is the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *row exposing the finite frontier of local peaks, residual premises, and obstruction rows.*
3. *L and R are the left and right residual-branch rows, each read through* ParallelDiamondResidual$^{\uparrow}$ *data.*
4. *K is the finite candidate-join row, recording the proposed common target and the branch-local routes to it.*
5. *S is the finite-window confluence row, read through* FiniteWindowConfluence$^{\uparrow}$ *before any matrix row is exported.*
6. *A is the candidate-normalization confluence handoff row, read through* MetacicCandidateNormalizationConfluence *when normalization-facing consumers inspect the same matrix.*
7. *O is the confluence-audit consumer row, read through* MetaCICConfluenceAuditPacket$^{\uparrow}$ *after P, F, L, R, K, S, A remain visible.*
8. *H records componentwise* hsame *transport for premise, frontier, residual-branch, candidate-join, finite-window, handoff, audit-consumer, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a confluence query through $P, F, L, R, K, S, A, O, H$.*

10. *G is the* Pkg *provenance over* MetaCICResidualJoinMatrix$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, FiniteWindowConfluence$^{\uparrow}$, ParallelDiamondResidual$^{\uparrow}$, MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$, MetaCICConfluenceAuditPacket$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetaCICResidualJoinMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, F, L, R, K, S, A, O, H, C, G, N$. It has no coordinate for premise-free parallel diamond, unrestricted closed Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Theorem 8134.2 (MetaCIC residual-join matrix NameCert obligations)**.** *For every accepted* MetaCICResidualJoinMatrix$^{\uparrow}$ *carrier*

$$J = (P, F, L, R, K, S, A, O, H, C, G, N),$$

the local NameCert$_{\text{MetaCICResidualJoinMatrix}^{\uparrow}}$ *surface consists exactly of residual-substitution premise retention through P, frontier exposure through F, two residual-branch exposures through L, R, candidate-join exposure through K, finite-window confluence exposure through S, candidate-normalization confluence handoff through A, confluence-audit consumption through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8134.1. Its matrix rows are P, F, L, R, K, S, A, O, and its structural rows are H, C, G, N.

The retained premise row P is read before the frontier row F can be consumed. The two residual rows L, R and the candidate row K then record the local join matrix under the same finite frontier.

The finite-window row S and candidate-normalization handoff row A keep the matrix bounded for confluence and normalization-facing consumers. The audit row O receives the matrix only after those rows remain visible.

Finally H, C, G, N transport, replay, source, and name the same packet. Since no excluded global endpoint is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8134.3 (MetaCIC residual-join matrix candidate-diamond boundary)**.** *Every candidate-diamond consumer of an accepted* MetaCICResidualJoinMatrix$^{\uparrow}$ *packet must expose the residual-substitution premise row P, the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *row F, both* ParallelDiamondResidual$^{\uparrow}$ *branch rows L, R, the candidate-join row K, the* FiniteWindowConfluence$^{\uparrow}$ *row S, the* MetacicCandidateNormalization *row A, and the* MetaCICConfluenceAuditPacket$^{\uparrow}$ *row O before any local diamond read is public. The read is a bounded audit-matrix route, not a premise-free parallel diamond or closed Church–Rosser theorem.*

Proof. Apply the NameCert obligations of Definition 8134.2. They force every accepted matrix read through the retained premise, frontier, two residual branches, candidate join, finite window, normalization handoff, audit consumer, and structural rows.

Read the residual side first. The row P keeps the substitution premise visible, while F, L, R, K display the finite local peak, its two residual descendants, and the candidate common target.

Read the bounded-audit side next. The row S restricts the read to a finite confluence window, A records the candidate-normalization handoff for the same matrix, and O supplies the confluence-audit consumer row.

The rows H, C, G, N only preserve that displayed route. Therefore the matrix cannot be cited as a premise-free parallel diamond, closed Church–Rosser theorem, subject-reduction theorem,

evaluator equality, quotient syntax equality, global normalization theorem, or kernel-consistency certificate. □

Falsifiable prediction: If the residual-substitution premise row P is omitted, the $\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}$ classifier rejects the matrix even when the frontier, residual-branch, candidate-join, finite-window, candidate-normalization handoff, and confluence-audit rows are present.

Independence witness: metacic-parallel-diamond-frontier, finite-window-confluence, parallel-diamond-residual, metacic-candidate-normalization-confluence-handoff, metacicconfluenceauditpacket: none is bijective to $\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}$ because this packet jointly carries retained residual-substitution premise data, frontier rows, two residual branches, one candidate join, finite-window confluence, candidate-normalization handoff, confluence-audit consumption, transport, replay, provenance, and local naming.

The Lean file `lean4/BEDC/Derived/MetaCICResidualJoinMatrixUp/TasteGate.lean` later supplies explicit Nontrivial $\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}$ and FieldFaithful $\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}$ instances, with field-projection witnesses keeping $P, F, L, R, K, S, A, O, H, C, G, N$ separate.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}$ packet is generated from a retained residual-substitution premise row, a parallel-diamond frontier row, two residual branch rows, a candidate-join row, a finite-window confluence row, a candidate-normalization confluence handoff row, a confluence-audit consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}}$, obligation)

Scope closed: Definition 8134.1, Definition 8134.2, and Definition 8134.3 expose the finite carrier rows, NameCert duties, and candidate-diamond boundary that form the residual-join matrix obligation surface for MetaCIC confluence audit boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinMatrix}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close premise-free parallel diamond, unrestricted closed Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, global normalization, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, premise retention, frontier exposure, residual-branch exposure, candidate-join exposure, finite-window confluence exposure, candidate-normalization handoff, audit-consumer non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualJoinMatrixUp.

Chapter 8135

A Concrete Naming Certificate for $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completions and uniformly continuous extension, the $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$ packet records the finite modulus data by which a uniformly continuous map on displayed regular-Cauchy dense input extends to a separated complete target. It sits after the regular rational readback of chapter 301, the finite stream windows of chapter 126, the terminal real-name seal of chapter 38, the unit row of chapter 1527, the uniform-continuity surface of chapter 391, and the universal-property packet of chapter 2319. The packet names the completion-extension handoff as displayed BEDC rows; it does not quotient Cauchy names, select representatives, import countable choice, or use ambient completeness outside the target metric row.

Definition 8135.1 (RegularCauchyExtensionModulus carrier)**.** *A* $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, S, X, U, T, M, G, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational readback of the dense input name.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording exactly which source observations are inspected before an extension value is requested.*
3. *X is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed regular Cauchy input.*
4. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row embedding the dense source into the completion-facing surface.*
5. *T is the separated target metric row, read through* $\mathsf{SeparatedMetric}^{\uparrow}$ *and the target's displayed complete-metric evidence.*
6. *M is the* $\mathsf{UniformContinuity}^{\uparrow}$ *modulus row for the source map on the displayed dense data.*
7. *G is the extension graph row, pairing completion-input windows with target values using only R, S, X, U, T, M.*
8. *L is the uniqueness row, comparing two extension graph rows over the same source, unit, target, and modulus data by separatedness of T.*

9. *H records componentwise* hsame *transport for $R, S, X, U, T, M, G, L, C, P, N$.*
10. *C records displayed* Cont *replay of dense-input reading, unit handoff, modulus use, graph construction, and uniqueness comparison.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, S, X, U, T, M, G, L, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, host function equality, countable-choice schedules, or ambient completeness outside the displayed separated target metric row.

Theorem 8135.2 (RegularCauchyExtensionModulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$ *carrier $E = (R, S, X, U, T, M, G, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyExte}}$ *surface consists exactly of regular Cauchy source admission, finite stream-window inspection, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, Cauchy-completion unit handoff, separated target metric exposure, uniform-continuity modulus use, extension graph construction, separated uniqueness comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier exposure is projection from Definition 8135.1. The tuple displays the source name, inspected finite windows, terminal Real seal, completion unit, separated target metric, modulus, graph, uniqueness, transport, replay, provenance, and local naming rows.

The dense input is read through R, S, X, U. The regular rational row and stream windows are visible before the terminal Real seal and before the completion unit can be consumed by an extension request.

The target-facing part is read through T, M, G, L. The separated target metric supplies the comparison surface, the uniform-continuity modulus bounds how finite source windows determine target windows, the graph row records the extension values, and the uniqueness row compares graph rows only over the same displayed data.

The structural boundary is fixed by H, C, P, N. These rows transport, replay, source, and name the same finite packet, while the carrier lists no quotient completion equality, selected limit, host function equality, countable-choice schedule, or ambient completeness coordinate outside T. □

Theorem 8135.3 (RegularCauchyExtensionModulus factorization)**.** *Every extension consumer of an accepted* $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionUnit}^{\uparrow} \longmapsto \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{BHist}$$

The consumer receives a finite extension graph and a separated uniqueness row only after the source regular Cauchy data, inspected stream windows, completion unit, target metric, and uniform-continuity modulus are displayed. It obtains no quotient completion equality, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.

Proof. Begin with an extension request for an accepted carrier $E = (R, S, X, U, T, M, G, L, H, C, P, N)$. The request can first inspect only the regular rational source row R, the finite window row S, and the terminal seal row X.

The completion handoff is then forced through U. This row is the only displayed passage from the dense regular-Cauchy data to the completion-facing input, so the request cannot substitute a quotient representative or selected limit.

Next the request reads T, M, G, L. The target metric row supplies separated comparison, the modulus row explains how finite source windows control target windows, the graph row records the extension values, and the uniqueness row compares only graph rows over the same visible source, unit, target, and modulus data.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier row stores host function equality, countable-choice schedules, or an ambient completeness principle, every extension consumer factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyExtensionModulus}^{\uparrow}$ packet is generated from RegSeqRat source data, StreamName finite windows, a RealUp seal, a CauchyCompletionUnit row, separated target metric evidence, a UniformContinuity modulus, an extension graph, separated uniqueness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8135.1, Definition 8135.2, and Definition 8135.3 seed the finite completion-extension route over regular Cauchy, stream, Real, completion-unit, uniform-modulus, separated-target, graph, uniqueness, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected representatives, host function equality, countable-choice schedules, or ambient completeness outside the displayed target metric row.

Upgrade path: Obligation closure requires checked carrier admission, finite-window inspection, unit handoff, uniform-modulus use, extension graph construction, separated uniqueness, transport replay, provenance nonescape, and local naming for BEDC.Derived.RegularCauchyExtensionModulusUp.

Chapter 8136

A Concrete Naming Certificate for $\mathsf{RegularCauchySubdivision}^{\uparrow}$

Following Bishop and Bridges, finite subdivision arguments for regular Cauchy data are useful only after the mesh, tolerance ledger, window reads, rational readback, and real-name seal are all visible as finite data. The $\mathsf{RegularCauchySubdivision}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational sequence boundary of chapter 301, the terminal real-name seal of chapter 38, and the rational scaling route of chapter 465. It records a finite subdivision induction surface rather than a selected limit, a quotient real equality, or an ambient compactness theorem.

Definition 8136.1 (Regular Cauchy subdivision carrier)**.** *A* $\mathsf{RegularCauchySubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite subdivision mesh row, listing the inspected subintervals, endpoints, adjacency witnesses, and boundary cells;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic mesh width, endpoint tolerances, and local error budgets used on each mesh cell;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family, exposing the finite stream windows inspected on each subdivision cell;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, reading every inspected window as regular rational data over the displayed dyadic ledger;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the mesh, dyadic tolerance ledger, StreamName windows, and RegSeqRat readback rows have been displayed;*
6. *H records componentwise* hsame *transport for M, D, W, R, E, C, P, N;*
7. *C records displayed* Cont *replay for the subdivision route;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubdivision}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubdivision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for selected limits, quotient real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8136.2 (Regular Cauchy subdivision NameCert obligations)**.** *For every accepted* RegularCauchySubdivisi *carrier* $S = (M, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubdivision}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite mesh, mesh compatibility between adjacent cells, readback stability through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The row M supplies the finite mesh and its adjacency witnesses, while D supplies the dyadic width and tolerance ledger used on every mesh cell.

Read the window and rational rows next. The family W exposes only finitely many StreamName windows, and R reads those windows as regular rational data over the same dyadic ledger. Mesh compatibility is therefore checked by comparing adjacent displayed rows, not by selecting a limit.

Finally replay the seal and structural rows. The row E is downstream of the mesh, tolerance, window, and readback data, while H, C, P, N only transport, replay, source, and name that same finite route. These projections exhaust the local NameCert obligations and leave no selected limits, quotient real equality, countable choice, open-cover compactness, or ambient completeness coordinate. □

Theorem 8136.3 (Regular Cauchy subdivision regular-window induction)**.** *Every induction over an accepted* $\mathsf{RegularCauchySubdivision}^{\uparrow}$ *packet is finite induction over the displayed mesh row* M*. The base cells read their tolerance data through* D*, their inspected windows through* W*, and their regular rational data through* R*; each adjacent step preserves the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, provenance* P*, and local naming row* N*. Consequently the induction route exports only finite subdivision data and never exports selected limits, quotient real equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct over the finite list of mesh cells carried by M. In the base cell, the tolerance ledger D, window family W, and readback row R provide the inspected regular Cauchy data before any Real-facing seal is available.

For an adjacent cell, use the mesh compatibility row inside M together with the dyadic ledger D. The next window read remains inside W, the regular rational read remains inside R, and the terminal seal is still the displayed row E.

The structural rows are copied componentwise along the induction. The rows H, C, P, N transport, replay, source, and name the same finite subdivision route, so the induction closes inside the carrier projections and cannot introduce selected limits, quotient real equality, countable choice, open-cover compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubdivision}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySubdivision}^{\uparrow}$ packet is generated from a finite subdivision mesh, dyadic tolerance ledger, StreamName window family, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySubdivision}^{\uparrow}}$, seed)

Scope closed: Definition 8136.1, Definition 8136.2, and Definition 8136.3 seed the finite subdivision induction route between regular rational windows and Real-name sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySubdivision}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected limits, quotient real equality, countable choice, open-cover compactness, ambient $\mathrm{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh compatibility, readback stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubdivisionUp.

Chapter 8137

A Concrete Naming Certificate for $\mathsf{BoundedLipschitzDomain}^{\uparrow}$

Following Evans and Adams–Fournier, the $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ packet records the finite boundary-chart data needed before Sobolev trace, PDE boundary estimates, Lipschitz extension, and compact metric arguments may read a domain as bounded with Lipschitz boundary. It sits over the metric carrier of chapter 57, the topology carrier of chapter 88, the Lipschitz-function carrier of chapter 3762, the Lipschitz-modulus carrier of chapter 3724, the compact-metric carrier of chapter 129, and the terminal real-name seal of chapter 38. The packet names a finite boundary-domain certificate only; it does not import host open-set equality, selected boundary charts, quotient metric equality, countable choice, or ambient compactness.

Definition 8137.1 (Bounded Lipschitz domain carrier)**.** *A* $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, M, F, L, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the open-set and topology row, naming the domain membership tests, boundary-neighbourhood windows, and local topology data read through* $\mathsf{Topology}^{\uparrow}$*.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row, naming the distance comparisons used by the boundedness and boundary-chart reads.*
3. *F is the finite boundary-chart cover row. Each chart names a displayed neighbourhood, a local coordinate window, and the finite overlap data needed to compare adjacent chart reads.*
4. *L is the Lipschitz constant and modulus row, read through* $\mathsf{LipschitzFunction}^{\uparrow}$ *and* $\mathsf{LipschitzModulus}^{\uparrow}$ *for every boundary chart and overlap comparison in F.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance row, recording the finite radius, height, and boundedness tolerances used by M, F, L before any real-valued boundary estimate is sealed.*
6. *H records componentwise* hsame *transport for the topology, metric, chart-cover, Lipschitz, tolerance, replay, provenance, and local certificate rows.*
7. *C records displayed* Cont *replay from a domain or boundary request through O, M, F, L, R, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLipschitzDomain}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{LipschitzFunction}^{\uparrow}$*,* $\mathsf{LipschitzModulus}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedLipschitzDomain}^{\uparrow}}$ *row.*
The carrier predicate asserts that the finite chart cover in F covers the displayed boundary windows from O, that each chart is read with the Lipschitz data in L, that boundedness and chart tolerances are sealed through R, and that all downstream reads pass through H, C, P, N. It has no coordinate for host open-set equality, a selected atlas outside F, quotient metric equality, countable choice, or ambient compactness.

Theorem 8137.2 (Bounded Lipschitz domain NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLipschitzDomain}$ *carrier $D = (O, M, F, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedLipschitzDomain}^{\uparrow}}$ *surface consists exactly of carrier admission, displayed topology and open-set exposure through O, metric distance exposure through M, finite boundary-chart cover exposure through F, Lipschitz constant and modulus exposure through L, real tolerance sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows O, M, F, L, R are the only analytic coordinates, and they respectively expose the topology, metric, boundary-chart cover, Lipschitz data, and real tolerance data.

Read the structural rows next. The row H transports those analytic coordinates componentwise, C replays a domain or boundary request through the displayed route, P records the allowed sibling sources, and N names the local certificate.

Since the carrier contains no further coordinate, every NameCert obligation is one of the listed rows. Host open-set equality, selected atlases, quotient metric equality, countable-choice schedules, and ambient compactness cannot be exported by the packet. □

Theorem 8137.3 (Bounded Lipschitz domain finite-chart handoff)**.** *Every Sobolev trace, PDE boundary, Lipschitz extension, or compact metric consumer of an accepted* $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ *carrier must read the finite boundary-chart cover F after the topology and metric rows O, M and before the Lipschitz row L and real tolerance row R are consumed. The handoff route exposes only finite chart windows, finite overlap comparisons, Lipschitz constants and moduli, real tolerances, transport, replay, provenance, and local naming.*

Proof. Begin with a consumer request. The domain membership and boundary-neighbourhood data are read through O, and the distance comparisons needed by boundedness and chart windows are read through M.

The finite chart row F is then the only boundary-cover coordinate available to the consumer. Its overlap data are finite packet data, so a downstream trace, PDE, extension, or compactness read cannot replace them by a host atlas or selected cover.

After F is displayed, the Lipschitz constants and moduli in L and the real tolerances in R can be read. The structural rows H, C, P, N preserve the same finite route, so the handoff exports no host open-set equality, quotient metric equality, countable-choice schedule, or ambient compactness theorem. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ boundary read exposes topology, metric, finite boundary-chart cover, Lipschitz constant and modulus, Real tolerance, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: metric, topology, lipschitz-function, lipschitz-modulus, compactmetric, real: $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries displayed open-set topology rows, metric rows, finite boundary-chart cover rows, Lipschitz constant and modulus rows, Real tolerance rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLipschitzDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ packet is generated from displayed open-set and topology rows, metric rows, finite boundary-chart cover rows, Lipschitz constant and modulus rows, Real tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLipschitzDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8137.1, Definition 8137.2, and Definition 8137.3 seed the finite boundary-domain packet for bounded Lipschitz domains.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLipschitzDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Sobolev trace, PDE regularity, Lipschitz extension, compactness, host open-set equality, selected boundary charts, quotient metric equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite boundary-chart cover exposure, Lipschitz constant and modulus exposure, Real tolerance sealing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedLipschitzDomainUp.

Chapter 8138

A Concrete Naming Certificate for KolmogorovRieszCriterion$^{\uparrow}$

Following Kolmogorov and Riesz, the KolmogorovRieszCriterion$^{\uparrow}$ packet records the finite L10 compactness criterion by which translation tightness, compact-source control, equicontinuous sequence data, rational readback, and terminal real seals are named before any function-space compactness consumer may inspect the route. It sits after the compact metric source of chapter 129, the regular rational comparison surface of chapter 301, the terminal real-name seal of chapter 38, the shared-modulus sequence surface of chapter 4356, and the finite compactness route of chapter 3846. The criterion carrier names only displayed finite windows, translation-tightness rows, readback rows, and the Frechet–Kolmogorov handoff; it does not assert an ambient compact embedding theorem, selected subsequence, quotient function equality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8138.1 (Kolmogorov-Riesz criterion carrier)**.** *A* KolmogorovRieszCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, Q, S, R, V, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the* CompactMetric$^{\uparrow}$ *source row whose finite net interface fixes the inspected compact metric support.*
2. *T is the finite translation-tightness row, listing the shifts, radii, and tolerance windows visible to the criterion request.*
3. *Q is the* EquicontinuousSequence$^{\uparrow}$ *row that supplies the shared sequence modulus consumed by compactness-facing reads.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row for inspected sequence entries, translation requests, and compact net cells.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for rational tolerance comparisons.*
6. *V is the* Real$^{\uparrow}$ *seal row for the displayed norm, distance, and translation estimates.*
7. *F is the* FrechetKolmogorovCompactness$^{\uparrow}$ *handoff row, packaging M, T, Q, S, R, V as one L10 finite compactness criterion.*
8. *H records componentwise* hsame *transport for compact metric, translation-tightness, equicontinuity, window, readback, Real, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compactness request through the same finite rows.*
10. *P is the* Pkg *provenance over* KolmogorovRieszCriterion$^{\uparrow}$, CompactMetric$^{\uparrow}$, EquicontinuousSequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, FrechetKolmogorovCompactness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{KolmogorovRieszCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, T, Q, S, R, V, F, H, C, P, N$.

Theorem 8138.2 (Kolmogorov-Riesz criterion NameCert obligations)**.** *For every accepted* KolmogorovRieszCriterio *carrier*

$$K = (M, T, Q, S, R, V, F, H, C, P, N),$$

the local NameCert$_{\text{KolmogorovRieszCriterion}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure, finite translation-tightness exposure, shared equicontinuous sequence modulus exposure, finite window visibility, regular rational readback, Real sealing, Frechet–Kolmogorov handoff exactness, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier rows of Definition 8138.1. The compact metric row M, translation row T, equicontinuous sequence row Q, window row S, readback row R, Real seal V, and Frechet–Kolmogorov handoff F are the only analytic coordinates.

Read the finite compactness criterion in order. Compact-source data come from M, translation tightness is displayed by T, and the common modulus needed by sequence consumers is supplied by Q before any window, rational, or real-facing read is inspected.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for selected subsequences, quotient equality of functions, countable choice, open-cover compactness, or ambient Real$^{\uparrow}$ completeness, the listed rows exhaust the local certificate surface. □

Theorem 8138.3 (Kolmogorov-Riesz translation tightness route)**.** *Every compactness-facing read of an accepted* KolmogorovRieszCriterion$^{\uparrow}$ *carrier factors through the finite route*

$$M, T, Q, S, R, V \longmapsto F, H, C, P, N.$$

Thus translation tightness is consumed only after compact metric support, equicontinuous sequence control, finite StreamName windows, RegSeqRat readback, and Real sealing have been displayed, and the final handoff is the Frechet–Kolmogorov compactness row rather than an ambient compact embedding theorem.

Proof. Start with a compactness-facing consumer and project the accepted carrier. The consumer first sees the compact metric source M and the translation-tightness row T, so the inspected shifts, radii, and tolerances are finite data of the packet.

Next read the equicontinuous sequence row Q and the finite window row S. This keeps shared-modulus information attached to the inspected sequence entries and compact net cells before any numerical comparison is exported.

The readback rows R, V then seal the rational and real-facing comparisons. Only after those rows are visible may the Frechet–Kolmogorov compactness handoff F be consumed.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because no row stores an ambient compact embedding theorem, selected subsequence, quotient function equality, countable-choice selector, or ambient completeness principle, the compactness-facing read is exactly the displayed translation-tightness route. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovRieszCriterion}\uparrow}$.

Constructive story: A $\mathsf{KolmogorovRieszCriterion}^{\uparrow}$ packet is generated from a CompactMetric source row, finite translation-tightness windows, an EquicontinuousSequence shared-modulus row, StreamName finite windows, RegSeqRat readback, Real sealing, FrechetKolmogorovCompactness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovRieszCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8138.1, Definition 8138.2, and Definition 8138.3 seed the finite Kolmogorov–Riesz compactness criterion over compact metric, translation-tightness, equicontinuous sequence, readback, Real, handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovRieszCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient compact embedding, open-cover compactness, selected subsequences, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite translation-tightness exactness, equicontinuous sequence modulus exposure, StreamName window visibility, RegSeqRat readback, Real sealing, Frechet–Kolmogorov handoff, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovRieszCriterionUp.

Chapter 8139

A Concrete Naming Certificate for $\mathsf{RegularCauchyBisimulation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyBisimulation}^{\uparrow}$ packet records equality-facing comparison of two regular Cauchy real names by finite paired windows rather than by quotient real equality. It sits after the regular rational source of chapter 301, the finite stream-window route of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness route of chapter 3390. The packet names a non-quotient comparison surface for Real and Completion consumers; it does not select representatives, collapse names by quotient equality, import countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8139.1 (Regular Cauchy bisimulation carrier)**.** *A* $\mathsf{RegularCauchyBisimulation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, W_X, W_Y, D_X, D_Y, M, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational approximants are being compared.*
2. *W_X and W_Y are paired* $\mathsf{StreamName}^{\uparrow}$ *windows exposing the finite observations of X and Y at the same inspected precisions.*
3. *D_X and D_Y are* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers recording the error budgets and comparison radii used for the two source rows.*
4. *M is the finite matching relation pairing inspected windows whose dyadic distance is below the displayed joint tolerance.*
5. *H records componentwise* hsame *transport for $X, Y, W_X, W_Y, D_X, D_Y, M, C, P, N$.*
6. *C records displayed* Cont *replay of each paired-window comparison through the source rows, stream windows, tolerance ledgers, matching relation, provenance, and local naming row.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBisimulation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBisimulation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one symmetric paired-window comparison certificate. It has no coordinate for quotient real equality, selected representatives, infinite branch selection, countable choice, host equality of real numbers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8139.2 (Regular Cauchy bisimulation NameCert obligations)**.** *For every accepted* RegularCauchyBisimula *carrier* $B = (X, Y, W_X, W_Y, D_X, D_Y, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBisimulation}^{\uparrow}}$ *surface consists exactly of carrier admission, paired-window exposure, dyadic tolerance comparison, symmetric step matching,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance, and local naming. In particular, every comparison read remains inside the displayed route*

$$X, Y \longmapsto W_X, W_Y \longmapsto D_X, D_Y \longmapsto M$$

before a Real-facing consumer may use the comparison.

Proof. Project the carrier of Definition 8139.1. The rows $X, Y, W_X, W_Y, D_X, D_Y, M$ expose exactly two regular rational source rows, their paired stream windows, their dyadic tolerance ledgers, and the finite matching relation.

The matching relation is symmetric because every admitted comparison step is stored as a paired-window relation between the two displayed sources. Reading the pair in the opposite direction uses the same windows and the same dyadic tolerance data, so no quotient-real equality or selected representative is introduced.

The Real-facing seal is downstream of the displayed source rows. The $\mathsf{Real}^{\uparrow}$ consumer can only receive a comparison after the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows have been replayed through C and transported through H.

Finally P and N source and name the same finite packet. Since the carrier has no coordinate for host equality, branch selection, choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBisimulation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyBisimulation}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a finite symmetric matching relation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBisimulation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8139.1 and Definition 8139.2 seed the paired-window comparison surface over RegSeqRat, StreamName, DyadicRatCore, Real, and regular-Cauchy uniqueness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBisimulation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, representative selection, countable choice, infinite branch selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-window exposure, dyadic tolerance comparison, symmetric step matching, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBisim

Chapter 8140

A Concrete Naming Certificate for $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$

The $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$ packet realizes the reversible transport trace surface of section 9463.2 as finite NameCert data. It sits beside the Cayley operator route of chapter 7887, the holonomy surface of chapter 6385, the metric route of chapter 3474, the reversible transport certificate of chapter 219, and the terminal Real seal of chapter 38. The packet names an operator trace only after reversible disk transport, classifier preservation, boundary refusal, and Real-seal readout have all been displayed.

Definition 8140.1 (Hyperbolic operator trace carrier)**.** *A* $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, O, U, L, B, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite disk-transport row recording the inspected Poincare-disk or hyperbolic disk windows and the reversible route between them.*
2. *O is the operator row whose finite action is read only on the displayed disk-transport window.*
3. *U is the reversible classifier-preservation row, recording the transport witness that preserves the relevant phase or inner-product classifier.*
4. *L is the trace ledger row, recording the finite trace read, normalization, visible source ledger, and comparison budget.*
5. *B is the boundary-refusal row excluding unlogged boundary collapse, hidden ambient Hilbert-space operators, global spectral measures, and unbounded functional calculus.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal readout row for the trace value or trace comparison exposed by L.*
7. *H records componentwise* hsame *transport for disk transport, operator, reversible classifier preservation, trace ledger, boundary refusal, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through disk transport, operator read, classifier preservation, trace ledger inspection, boundary refusal, Real-seal readout, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$, $\mathsf{HyperbolicOperatorCayley}^{\uparrow}$, $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorTrace}^{\uparrow}}$ *row.*
The Lean-side follow-up for this carrier must provide `Nontrivial HyperbolicOperatorTraceUp` *and* `FieldFaithful HyperbolicOperatorTraceUp` *TasteGate instances in* `lean4/BEDC/Derived/HyperbolicOpe`

Theorem 8140.2 (Hyperbolic operator trace NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$ *carrier* $T = (D, O, U, L, B, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorTrace}^{\uparrow}}$ *surface consists exactly of finite disk-transport exposure, operator-row admission, reversible classifier-preservation exposure, trace-ledger readout, boundary-refusal exposure,* $\mathsf{Real}^{\uparrow}$ *seal readout, componentwise transport, displayed replay, provenance, and local naming. A trace read that omits* U*,* L*, or* B *is not an accepted trace read of this packet.*

Proof. Project the carrier of Definition 8140.1. The rows D, O, U, L, B, R expose the disk transport, operator window, reversible classifier-preservation witness, trace ledger, boundary refusal, and Real-seal readout.

The trace row is not read in isolation. It is admitted only after the reversible transport row U preserves the displayed classifier over the disk route D, and the ledger row L records the finite trace value together with its source and comparison budget.

The boundary-refusal row B blocks unlogged operator reads, boundary collapse, global spectral measures, and host Hilbert-space constructions. The Real-seal row R is therefore a readout of the displayed trace ledger, not a substitute for the transport and refusal rows.

Finally H, C, P, N preserve, replay, source, and name the same packet. Since no additional coordinate stores an ambient operator trace, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$ packet that exposes a trace read without the reversible transport row, boundary-refusal row, and trace ledger refutes the classifier.

Independence witness: hyperbolic-operator-cayley, hyperbolic-phase-holonomy, hyperbolic-metric, unitarygroup, real: HyperbolicOperatorTraceUp is not bijective to any listed sibling because it binds reversible disk transport, operator trace, boundary refusal, and Real-seal readout in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicOperatorTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicOperatorTrace}^{\uparrow}$ packet is generated from finite disk transport, an operator row, reversible classifier preservation, a trace ledger, a boundary-refusal row, a Real-seal readout, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicOperatorTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8140.1 and Definition 8140.2 seed the reversible disk-transport trace surface over hyperbolic operator, holonomy, metric, unitary, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicOperatorTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global spectral traces, host Hilbert-space operators, unbounded functional calculus, boundary collapse, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reversible disk transport, operator-trace ledger exactness, boundary refusal, Real-seal readout, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicOperatorTraceUp.

Chapter 8141

A Concrete Naming Certificate for $\mathsf{LowerTopology}^{\uparrow}$

Following Scott and Lawson domain theory, the $\mathsf{LowerTopology}^{\uparrow}$ packet records the lower-topology component of an ordered carrier as finite NameCert data. It sits between the specialization discipline of chapter 52 and chapter 53, the indexed-open surface of chapter 88, the observable-open classifier of chapter 1860, and the domain-topology neighbours chapter 5950 and chapter 5162. The packet names lower-open reads through displayed order and observable-open rows; it does not import a host powerset topology, a selected family of all opens, quotient point equality, choice over down-closed witnesses, or an ambient compactness theorem.

Definition 8141.1 (Lower topology carrier)**.** *A* $\mathsf{LowerTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, T, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Preorder}^{\uparrow}$ *specialization source row, carrying the finite comparison reads used by lower-open inspection.*
2. *Q is the* $\mathsf{Poset}^{\uparrow}$ *separated-order row when the lower topology is read over a displayed antisymmetric order surface.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row that records the finite lower-open neighbourhood requests admitted by the packet.*
4. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row, reading membership in each displayed lower-open request as a finite truth observation.*
5. *H records componentwise* hsame *transport for the order, topology, observable-open, replay, provenance, and naming rows.*
6. *C records displayed* Cont *replay from an order comparison request through the indexed-open row and then through the observable-open classifier.*
7. *P is the* Pkg *provenance over* $\mathsf{LowerTopology}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerTopology}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eight rows. It has no coordinate for a host family of all lower opens, arbitrary powersets, selected bases, quotient point equality, countable choice, or a compactness principle.

Theorem 8141.2 (Lower topology NameCert obligations)**.** *For every accepted* $\mathsf{LowerTopology}^{\uparrow}$ *carrier* $L = (R, Q, T, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LowerTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the preorder or poset row, lower-open topology row, observable-open row, transport row, replay row, provenance row, and local naming row; lower-open membership inspection through* T *and* S*; order compatibility through* R *and, when present,* Q*; and provenance exhaustion through* P, N*. No obligation is discharged by a host powerset topology, selected lower-open basis, quotient point equality, countable-choice schedule, Scott-open completeness, or Lawson compactness.*

Proof. Project the displayed carrier of Definition 8141.1. The only order-facing data are R and Q, so every lower-open read must begin with the finite preorder or poset comparisons already present in the packet.

Read the topology and observable-open rows next. The row T records the indexed lower-open requests, while S turns each membership read into displayed Sierpinski truth data. Thus the topology route remains a finite observation surface rather than a host powerset of opens.

Finally use H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same row order from comparison data to observable-open data. Since none of the excluded host objects has a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8141.3 (Lower topology specialization scope)**.** *Every accepted* $\mathsf{LowerTopology}^{\uparrow}$ *consumer that reads specialization information must factor through the finite route*

$$\mathsf{Preorder}^{\uparrow} \longmapsto \mathsf{Poset}^{\uparrow} \longmapsto \mathsf{Topology}^{\uparrow} \longmapsto \mathsf{SierpinskiSpace}^{\uparrow}.$$

The preorder row supplies comparison data, the optional poset row supplies separated-order data, the topology row names lower-open neighbourhood requests, and the Sierpinski row exposes membership observations. The route may be consumed by $\mathsf{ScottTopology}^{\uparrow}$ *and* $\mathsf{LawsonTopology}^{\uparrow}$ *as their lower-topology component, but it exports no host powerset topology, selected open basis, quotient order, choice principle, or compactness theorem.*

Proof. Start with the order rows of Definition 8141.1. A specialization-facing request first reads R, and when a separated order surface is present it reads Q as the displayed $\mathsf{Poset}^{\uparrow}$ handoff.

The request then passes to T. Lower-open neighbourhoods are indexed rows in the $\mathsf{Topology}^{\uparrow}$ surface, so the consumer sees only the finite lower-open requests carried by the packet.

The final observable read is S. Membership in those indexed opens is exposed through $\mathsf{SierpinskiSpace}^{\uparrow}$ truth data, after which H, C, P, N transport, replay, source, and name the same route. The Scott and Lawson neighbours may consume this component only through the displayed rows, so no external topology or choice object is exported. □

Falsifiable prediction: Every accepted $\mathsf{LowerTopology}^{\uparrow}$ read must display Preorder, Poset, Topology, SierpinskiSpace, hsame, Cont, Pkg, and local NameCert rows before it can serve as the lower-topology component for a ScottTopologyUp or LawsonTopologyUp consumer.

Independence witness: preorder, poset, topology, sierpinski-space, scott-topology, lawson-topology: none is bijective to $\mathsf{LowerTopology}^{\uparrow}$ because this carrier alone binds comparison data, separated-order data, lower-open requests, observable-open truth, transport, replay, provenance, and local naming as one lower-topology packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LowerTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerTopology}^{\uparrow}$ packet is generated from Preorder comparison rows, optional Poset separated-order rows, Topology lower-open rows, SierpinskiSpace observable-open rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8141.1, Definition 8141.2, and Definition 8141.3 seed the finite lower-topology route over $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host powerset topology, selected basis of lower opens, quotient point equality, countable choice, Scott-topology completeness, Lawson compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order compatibility, lower-open inspection, observable-open exactness, transport, replay, provenance, and local naming for BEDC.Derived.LowerTopologyUp.

Chapter 8142

A Concrete Naming Certificate for $\mathsf{DescentData}^{\uparrow}$

Following Grothendieck on descent and SGA-style gluing, the $\mathsf{DescentData}^{\uparrow}$ packet records local objects, restriction isomorphisms, cocycle comparisons, and effectivity rows as finite BEDC data. It sits over the coverage surface of chapter 6718, the restriction surface of chapter 150, the locality and gluing surface of chapter 151, the groupoid-valued descent surface of chapter 260, and the object-arrow carrier of chapter 61. The packet names only the displayed descent rows and does not export a host quotient stack, selected gluing object, global representative choice, or ambient sheaf equality.

8142.1 Descent Carrier

Definition 8142.1 (Descent data carrier)**.** *A* $\mathsf{DescentData}^{\uparrow}$ *carrier is a tuple* $(U, L, R, C, E, H, T, P, N)$ *with the following rows.*

1. *U is a cover row over the displayed* $\mathsf{GrothendieckTopology}^{\uparrow}$ *site, naming the finite cover or covering sieve from which the descent packet is read.*
2. *L is the family of local-object rows, one row for each certified member of U, read through the* $\mathsf{PreSheaf}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, *or* $\mathsf{Stack}^{\uparrow}$ *restriction surface relevant to the target consumer.*
3. *R is the restriction-isomorphism row, assigning to each certified overlap a displayed comparison between the two restrictions of the corresponding local-object rows.*
4. *C is the cocycle row, asserting that on each displayed triple overlap the two composite restriction-isomorphism routes have the same classifier read.*
5. *E is the effectivity and gluing row, naming the accepted global row or gluing-readback surface that may be consumed after C has been checked.*
6. *H is componentwise* hsame *transport for covers, local objects, restriction isomorphisms, cocycle witnesses, and effectivity witnesses.*
7. *T is displayed* Cont *replay for the route that reads U, then L, then R, then C, then E.*
8. *P is* Pkg *provenance for the site, cover, local-object source, overlap restriction source, cocycle source, and effectivity source.*
9. *N is a local* $\mathsf{NameCert}$ *row naming the accepted descent packet and every exported row.*

No component of the carrier is a host quotient of objects, a selected representative of a gluing class, a global choice function for local objects, or an equality of ambient sheaf sections outside the displayed classifier rows.

Theorem 8142.2 (Descent data NameCert obligations)**.** *Every accepted* $\mathsf{DescentData}^{\uparrow}$ *packet exposes exactly the carrier, cover, local-object, restriction, cocycle, effectivity, transport, replay, provenance, and naming obligations listed in Definition 8142.1. A consumer may inspect the cover row* U*, local-object rows* L*, restriction-isomorphism rows* R*, cocycle row* C*, effectivity row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* NameCert *row* N*, but obtains no host quotient stack, selected global gluing, global choice of representatives, ambient sheaf equality, or standard bridge.*

Proof. Project the tuple $(U, L, R, C, E, H, T, P, N)$ from Definition 8142.1. The first five coordinates give the analytic descent data: a displayed cover, local objects over that cover, restriction isomorphisms on overlaps, a cocycle witness on triple overlaps, and the effectivity or gluing row.

The remaining coordinates give the BEDC certificate discipline. Componentwise hsame moves each displayed row without replacing it by host equality, Cont replays the read order, Pkg records the site and gluing provenance, and NameCert names the packet and its exported rows.

The exclusions follow by absence of carrier coordinates. A host quotient stack, a selected global gluing, a representative-choice operation, ambient sheaf equality, and a standard bridge would each require an additional row not present in the tuple, so they are outside the accepted certificate. □

Theorem 8142.3 (Descent cocycle and effectivity route)**.** *Every stack or sheaf gluing consumer of a* $\mathsf{DescentData}^{\uparrow}$ *packet must read the cocycle rows before the effectivity rows. The exported route is*

$$U \longrightarrow L \longrightarrow R \longrightarrow C \longrightarrow E,$$

then H, T, P, N *transport, replay, source, and name the same route. This export supplies no host quotient of objects, no global choice of representatives, and no ambient sheaf equality.*

Proof. Start with the carrier order of Definition 8142.1. The cover row U determines the displayed local indices, L supplies the local objects on those indices, and R supplies the overlap restriction-isomorphism rows.

The effectivity row E is certified only after the cocycle row C has checked the triple-overlap composites. Thus a sheaf or stack gluing consumer reaches E through the finite route U, L, R, C rather than by an ambient existence principle.

Finally apply the structural rows H, T, P, N. They transport the route componentwise, replay the same read order, record provenance, and name the resulting packet. Since none of these rows is a quotient, representative selector, choice function, or host equality witness, the exported route does not supply those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{DescentData}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DescentData}^{\uparrow}$ packet is generated from a Grothendieck-topology cover row, local-object rows, restriction-isomorphism rows, a cocycle row, an effectivity or gluing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DescentData}^{\uparrow}}$, seed)

Scope closed: Definition 8142.1, Definition 8142.2, and Definition 8142.3 bind the finite descent-data carrier and the cocycle-before-effectivity route for sheaf and stack consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DescentData}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host quotient stacks, selected gluings, global choice of representatives, ambient sheaf equality, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, cocycle-before-effectivity read order, componentwise transport, provenance, naming, and the absence of quotient, choice, and ambient-equality exports for BEDC.Derived.DescentDataUp.

Chapter 8143

A Concrete Naming Certificate for $\mathsf{LipschitzExtensionDomain}^{\uparrow}$

Following Jones, Stein, and Adams–Fournier, the $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ packet names the finite domain-side admissibility data needed before a Lipschitz, Sobolev, trace, compact, or real-valued extension consumer may read a domain as extension-ready. It sits between the bounded-domain certificate of chapter 8137, the operator route of chapter 3650, the Sobolev extension surface of chapter 7326, and the metric, topology, compact, and real rows used by those neighbours. The packet is a domain-level gate, not a selected extension operator, a host trace theorem, a quotient equality of function spaces, or a choice principle.

Definition 8143.1 (Lipschitz extension domain carrier)**.** *A* $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ *carrier is a finite packet*

$$D = (D_0, M, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. D_0 *is the* $\mathsf{BoundedLipschitzDomain}^{\uparrow}$ *row, naming the finite boundary-chart, boundedness, and local domain data.*
2. M *is the* $\mathsf{Met}^{\uparrow}$ *row, recording the distance comparisons used by local extension and trace reads.*
3. T *is the* $\mathsf{Topology}^{\uparrow}$ *row, recording membership, neighbourhood, and boundary-open reads.*
4. L *is the Lipschitz bound row, read through* $\mathsf{LipschitzExtension}^{\uparrow}$ *and the sibling Lipschitz rows for every admissible extension request.*
5. E *is the extension-route row, recording the allowed handoff to* $\mathsf{SobolevExtensionOperator}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *support data without selecting a global operator.*
6. H *records componentwise* hsame *transport for* D_0, M, T, L, E, C, P, N.
7. C *records displayed* Cont *replay from an extension or trace request through the domain, metric, topology, Lipschitz, and extension-route rows.*
8. P *is the* Pkg *provenance over* $\mathsf{LipschitzExtensionDomain}^{\uparrow}$, $\mathsf{BoundedLipschitzDomain}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{SobolevExtensionOperator}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}$ *row.*

The carrier predicate asserts that every admissible extension-domain read first exposes the bounded Lipschitz domain row D_0, then the metric and topology rows M, T, then the Lipschitz bound and extension-route rows L, E, and finally the transport, replay, provenance, and local naming rows H, C, P, N. It has no coordinate for a selected extension operator, host open-set equality, quotient metric equality, completed trace space, or countable choice.

Theorem 8143.2 (Lipschitz extension domain NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzExtensionDomai}$ *carrier*

$$D = (D_0, M, T, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the bounded Lipschitz domain, metric, topology, Lipschitz bound, extension route, transport, replay, provenance, and local naming rows;*
2. *metric-topology compatibility, requiring distance, membership, neighbourhood, and boundary reads to pass through M and T after D_0 is displayed;*
3. *Lipschitz bound exposure, requiring every extension-ready request to expose the row L before any extension consumer reads a function bound;*
4. *extension-route admissibility, requiring the row E to route only to displayed* $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{SobolevExtensionOperator}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *support rows;*
5. *structural stability under H, C, P, N, which transports, replays, sources, and names only the displayed finite route.*

No domain-level extension obligation is stored outside these rows.

Proof. Project the carrier of Definition 8143.1. The domain row D_0 is the only bounded Lipschitz domain coordinate, and the rows M, T are the only metric and topology coordinates available to a consumer.

Next read the Lipschitz and extension-route rows. The row L records the bound surface that an extension request may inspect, while E records the admissible handoff to the displayed Lipschitz, Sobolev, compact, and real support rows.

Finally attach the structural rows H, C, P, N. They transport, replay, source, and name the same finite route, so the carrier has no remaining coordinate from which a selected operator, host open-set equality, quotient metric equality, completed trace space, or choice schedule could be exported. □

Theorem 8143.3 (Lipschitz extension domain extension-trace route)**.** *Every extension or trace consumer of an accepted* $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ *carrier must pass through the displayed domain row D_0, the metric and topology rows M, T, the Lipschitz bound row L, the extension-route row E, componentwise transport H, displayed replay C, provenance P, and the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}$ *row N before receiving any extension-domain conclusion. The route may cite the bounded-domain surface of chapter 8137 and the Sobolev extension surface of chapter 7326, but it exports no host extension operator, trace theorem, quotient function-space equality, ambient compactness, or choice principle.*

Proof. Begin with an accepted extension or trace request. Carrier admission forces the request to display D_0 first, so the bounded Lipschitz boundary and domain data are read through the sibling bounded-domain certificate rather than a host domain predicate.

The request then reads the metric and topology rows M, T. These rows provide the distance, membership, neighbourhood, and boundary-open data that the extension-domain route may use before any Lipschitz estimate is inspected.

Next expose the Lipschitz bound row L and the extension-route row E. Together they keep the request inside the displayed Lipschitz and Sobolev extension neighbours, with compact and real support rows available only as named finite packet data.

Finally apply H, C, P, N. Transport and replay preserve the same row order, provenance records the permitted sibling sources, and the local NameCert names the domain-level gate. Since no carrier coordinate remains for a selected operator, host trace theorem, quotient equality, ambient compactness, or choice schedule, every accepted consumer must follow the stated route. □

Falsifiable prediction: Every accepted $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ extension-domain read exposes BoundedLipschitzDomainUp, MetricUp, TopologyUp, LipschitzExtensionUp, SobolevExtensionOperatorUp, CompactMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any extension or trace consumer receives a domain admissibility conclusion.

Independence witness: bounded-lipschitz-domain, metric, topology, lipschitz-extension, sobolev-extension-operator, compactmetric, real: $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries domain admissibility, metric and topology readback, Lipschitz bound exposure, extension-route admissibility, transport, replay, provenance, and local naming in one finite gate.

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzExtensionDomain}^{\uparrow}$ packet is generated from a bounded Lipschitz domain row, metric row, topology row, Lipschitz bound row, extension-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8143.1, Definition 8143.2, and Definition 8143.3 seed the finite extension-domain admissibility packet over bounded-domain, metric, topology, Lipschitz, Sobolev extension, compact, and real neighbours.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzExtensionDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a selected host extension operator, unrestricted trace theorem, quotient function-space equality, ambient compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-topology compatibility, Lipschitz bound exposure, extension-route admissibility, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzExtensionDomainUp.

Chapter 8144

A Concrete Naming Certificate for $\mathsf{LinearRecurrence}^{\uparrow}$

Following Stanley and Graham–Knuth–Patashnik on linear recurrence sequences, the $\mathsf{LinearRecurrence}^{\uparrow}$ packet records a finite recurrence-stream NameCert for the L10 $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route. It sits after the unary index discipline of chapter 28, the rational coefficient rows of chapter 37, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite lawlike stream data only; it does not import host sequence equality, selected infinite streams, closed-form solutions, characteristic roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8144.1 (Linear recurrence carrier)**.** *A* $\mathsf{LinearRecurrence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (C, I, N, S, Q, R, H, P, A)$$

with the following displayed rows:

1. *C is the coefficient row over* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, listing the finite recurrence order, rational coefficients, and dyadic tolerance tags used by later reads.*
2. *I is the initial finite window, giving the seed values needed before the recurrence rule can be replayed.*
3. *N is the unary* $\mathsf{Nat}^{\uparrow}$ *index schedule naming which finite indices and successor steps are inspected.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing the inspected stream entries and recurrence replay windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational values generated by the coefficient row and initial window.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for those readback values when a Real or Completion consumer asks for terminal scalar output.*
7. *H records componentwise* hsame *transport for coefficient, initial-window, index, stream, readback, and seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*, and* Cont *rows.*
9. *A is the local* $\mathsf{NameCert}_{\mathsf{LinearRecurrence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $C, I, N, S, Q, R, H, P, A$.

Theorem 8144.2 (Linear recurrence NameCert obligations)**.** *For every accepted* $\mathsf{LinearRecurrence}^{\uparrow}$ *carrier*

$$L = (C, I, N, S, Q, R, H, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{LinearRecurrence}^{\uparrow}}$ *surface consists exactly of coefficient-row exposure, initial-window exposure, unary index scheduling, finite stream observation, regular rational readback, Real sealing, componentwise transport, provenance, and local naming.*

Proof. Project the carrier of Definition 8144.1. The recurrence rule is not an ambient function on host sequences; it is the finite row C together with the initial window I and the unary index schedule N.

Read the numeric rows through the sibling chapters. The index schedule is supplied by chapter 28, the rational coefficients by chapter 37, and the dyadic tolerance tags by chapter 291.

Finally route observations through chapter 126, readback through chapter 301, and terminal sealing through chapter 38. The remaining rows H, P, A only transport, source, and name the same finite packet, so the listed rows exhaust the local certificate surface. □

Theorem 8144.3 (Linear recurrence stream handoff)**.** *Every stream-facing read of an accepted* $\mathsf{LinearRecurrence}^{\uparrow}$ *carrier factors through the finite route*

$$C, I, N \longmapsto S \longmapsto Q.$$

The consumer first sees the recurrence coefficients, initial window, and unary index schedule, then the $\mathsf{StreamName}^{\uparrow}$ *observation row, and only then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. No stream read may replace this route by host sequence equality, an uninspected tail, a selected closed-form solution, or an infinite recurrence oracle.*

Proof. Begin with the coefficient, initial-window, and index rows. By chapter 37 and chapter 291, the coefficient row is finite rational-and-dyadic data; by chapter 28, the schedule row names only unary inspected indices.

Pass next to the stream observation row of chapter 126. It exposes the requested entries and replay windows, not a quotient of infinite streams.

The regular rational readback of chapter 301 consumes exactly those displayed windows. Since the route has no coordinate for host sequence equality, uninspected tails, selected closed forms, or infinite recurrence oracles, every stream-facing read factors through C, I, N, S, Q. □

Theorem 8144.4 (Linear recurrence Real seal boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LinearRecurrence}^{\uparrow}$ *packet may read the terminal scalar row only through*

$$C, I, N, S, Q, R, H, P, A.$$

The $\mathsf{Real}^{\uparrow}$ *seal R records a terminal boundary for displayed recurrence readback values; it does not assert convergence of the whole recurrence stream, completeness of the ambient real line, characteristic-root solvability, or equality with a host sequence.*

Proof. Use Definition 8144.3 to force the consumer through the finite coefficient, initial-window, index, observation, and regular-readback rows. The construction therefore reaches the real-facing surface only after $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows have been exposed through chapter 28, chapter 37, chapter 291, chapter 126, and chapter 301.

Apply the terminal seal boundary of chapter 38. The $\mathsf{Real}^{\uparrow}$ row R seals displayed readback values; it is not a convergence theorem for all indices, a completeness principle, or a host-sequence equality test.

The transport, provenance, and local naming rows H, P, A preserve only the same route. Hence the Real or Completion consumer receives exactly the displayed recurrence-readback boundary and no stronger analytic structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{LinearRecurrence}^{\uparrow}}$.

Constructive story: A $\mathsf{LinearRecurrence}^{\uparrow}$ packet is generated from finite rational and dyadic coefficient data, an initial window, a unary Nat index schedule, StreamName observations, RegSeqRat readback, a Real seal, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LinearRecurrence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8144.1, Definition 8144.2, Definition 8144.3, and Definition 8144.4 seed the finite recurrence-stream carrier, stream handoff, and Real seal boundary for the L10 StreamName–RegSeqRat–Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LinearRecurrence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence equality, closed-form recurrence solving, characteristic-root theory, convergence of all recurrence streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient exactness, initial-window coverage, unary index scheduling, StreamName observation, RegSeqRat readback, Real sealing, transport, provenance, and local naming for BEDC.Derived.LinearRecurrenceUp.

Chapter 8145

A Concrete Naming Certificate for $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$

Following Bishop–Bridges on constructive Cauchy comparison and Mac Lane on functorial packaging, the $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$ packet records tail-comparison transport between regular Cauchy presentations as finite BEDC data. It sits after the objectwise comparison surface of chapter 387, the regular comparison route of chapter 2146, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names comparison transport and composition as a finite functorial boundary; it does not quotient streams, select representatives, compare host real values, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8145.1 (CauchyTailComparisonFunctor carrier)**.** *A* $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, B, W_A, W_B, D_A, D_B, T, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the source and target* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback rows;*
2. *W_A and W_B are the* $\mathsf{StreamName}^{\uparrow}$ *windows that select the compared source and target tails;*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows read through those windows;*
4. *T is the local* $\mathsf{CauchyTailComparison}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailComparison}^{\uparrow}$ *comparison row over the displayed windows;*
5. *R is the transported* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the comparison;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row visible only after the comparison readback has been exposed;*
7. *H stores componentwise* hsame *transports for the source, target, window, dyadic, comparison, readback, and seal rows;*
8. *C stores the displayed* Cont *replay routes for reading the windows, applying the comparison, transporting readback, and exposing the real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, $\mathsf{RegularCauchyTailCompari}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparisonFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient stream equality, selected limits, host real equality, or ambient completeness.

Theorem 8145.2 (CauchyTailComparisonFunctor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparisonFu}}$ *surface for the carrier of Definition 8145.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet F with source and target* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, a comparison row, a transported readback row, a* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, continuation rows, provenance, and local naming data;*
2. *window functoriality: every comparison read first exposes W_A, W_B, D_A, D_B before the row T is consumed;*
3. *regular readback transport: the transported row R is obtained from the comparison row and remains inside the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface;*
4. *classifier stability: componentwise* hsame *transport preserves the same comparison packet and its displayed* Cont *replay routes;*
5. *real seal handoff: every* $\mathsf{Real}^{\uparrow}$ *consumer sees $A, B, W_A, W_B, D_A, D_B, T, R, H, C, P, N$ before the seal row E is readable.*

Proof. Carrier admission is projection from the tuple in Definition 8145.1. The source and target regular rows, windows, dyadic endpoint rows, comparison row, transported readback, seal, transports, replays, package row, and local naming row are explicit coordinates.

Window functoriality follows from the sibling comparison chapters. The objectwise comparison row of chapter 387 and the regular comparison row of chapter 2146 both read finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows before exposing any comparison result.

The transported readback and classifier clauses then use the same finite route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is obtained only after T has been read, and componentwise hsame transport keeps the source, target, windows, endpoint rows, comparison row, and readback row in the displayed packet.

Finally the $\mathsf{Real}^{\uparrow}$ seal handoff is a consumer boundary. Since E is reached only after the readback row and its visible provenance have been exposed, no quotient stream, selected limit, host real equality, or ambient completeness coordinate can be projected from the carrier. □

Theorem 8145.3 (CauchyTailComparisonFunctor composition law)**.** *Let*

$$F_{AB} = (A, B, W_A, W_B, D_A, D_B, T_{AB}, R_{AB}, E_{AB}, H_{AB}, C_{AB}, P_{AB}, N_{AB})$$

and

$$F_{BC} = (B, C, W_B, W_C, D_B, D_C, T_{BC}, R_{BC}, E_{BC}, H_{BC}, C_{BC}, P_{BC}, N_{BC})$$

be accepted $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$ *packets whose middle* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row agree as displayed carrier rows. Then any composite comparison from A to C factors through the finite route*

$$W_A,\ W_B,\ W_C,\ D_A,\ D_B,\ D_C,\ T_{AB},\ T_{BC},\ R_{AB},\ R_{BC}$$

and reaches a $\mathsf{Real}^{\uparrow}$ *seal only after the combined* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows of the two packets have been exposed. The composite exports no quotient stream equality, selected representative, host real equality, or ambient completeness object.*

Proof. First expose the two accepted carriers. The shared middle row B, window W_B, and endpoint row D_B are common displayed coordinates, so both comparison reads use the same middle tail rather than a hidden equality class.

Apply the NameCert obligations of Definition 8145.2 to F_{AB} and F_{BC}. The first packet reads W_A, W_B, D_A, D_B before T_{AB} and R_{AB}; the second reads W_B, W_C, D_B, D_C before T_{BC} and R_{BC}.

The composite route is therefore the concatenation of two displayed comparison reads with the middle rows identified by the carrier data. Componentwise hsame transport and displayed Cont replay preserve the same windows, endpoint rows, comparison rows, and readback rows across the concatenation.

Finally the two $\mathsf{Real}^{\uparrow}$ seal rows are consumer boundaries. A composite seal can be read only after the visible readback, transport, replay, provenance, and local naming rows have been exposed, so the composition law is a finite functorial boundary and not a quotient-stream, selected-limit, host-equality, or ambient-completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailComparisonFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailComparisonFunctor}^{\uparrow}$ packet is generated from source and target RegSeqRat rows, paired StreamName windows, paired DyadicRatCore endpoint rows, CauchyTailComparison or RegularCauchyTailComparison rows, a transported RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailComparisonFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: The chapter names the finite carrier, obligation surface, and composition boundary for regular-Cauchy tail-comparison transport.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailComparisonFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient equality of streams, selected limits, host real equality, ambient RealUp completeness, or standard bridge checking.

Upgrade path: Obligation closure requires a checked carrier and NameCert obligation theorem for the displayed tail-comparison functor packet.

Chapter 8146

A Concrete Naming Certificate for $\mathsf{BishopLocatedDiagonal}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, the $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ packet records the finite route by which a located diagonal subsequence is read from displayed rational and stream data. It sits over the dyadic core of chapter 291, the stream naming surface of chapter 126, the regular rational sequence carrier of chapter 301, and the terminal real seal of chapter 38. The packet also consumes diagonal-extraction sibling rows such as $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ and $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ before it exports a located diagonal read. It is a finite naming certificate, not countable choice, quotient stream equality, selected host subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

8146.1 Carrier definition

Definition 8146.1 (BishopLocatedDiagonal carrier)**.** *A* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, W, R, E, X, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the diagonal error budgets, comparison cells, and slack conversions.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected source stream entries and their chosen diagonal indices.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the dyadic tolerances and stream windows have been displayed.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the located real endpoint read from the regular rational diagonal.*
5. *X is the diagonal-extraction sibling row, read through* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *and* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$*, that records the finite extraction handoff.*
6. *L is the locatedness ledger, recording the displayed lower and upper rational comparisons used by the diagonal request.*
7. *H records componentwise* hsame *transport for Q, W, R, E, X, L.*
8. *C records displayed* Cont *replay of the same finite diagonal route.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the diagonal-extraction siblings,* BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$ *row.*

The classifier accepts exactly packets exposing these rows and compares them componentwise through the inherited classifiers. It has no coordinate for countable choice, quotient stream equality, host-level subsequence identity, hidden real completeness, or an unrecorded diagonal selector.

Lean4 ✓ `BEDC.Derived.BishopLocatedDiagonalUp.TasteGate.BishopLocatedDiagonalTasteGate single carrier alignment`

8146.2 Certificate obligations

Theorem 8146.2 (BishopLocatedDiagonal NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ *carrier $D = (Q, W, R, E, X, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$ *surface consists exactly of dyadic tolerance exposure through Q, finite stream-window exposure through W, regular rational readback through R, real seal non-escape through E, diagonal-extraction handoff through X, locatedness comparison through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $Q, W, R, E, X, L, H, C, P, N$ are the full visible inventory, so any consumer of the diagonal read must begin with the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, extraction row, and locatedness ledger.

The inherited dependencies fix the numeric route. The dyadic row is read through chapter 291, the stream windows through chapter 126, the regular readback through chapter 301, and the final seal through chapter 38.

The sibling row X then records which finite diagonal-extraction handoff is being consumed. Since the structural rows H, C, P, N only transport, replay, source, and name the same displayed route, no countable choice principle, quotient stream equality, selected host subsequence, or ambient completeness row is introduced. □

Theorem 8146.3 (BishopLocatedDiagonal window handoff)**.** *Every* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ *consumer that reads a located diagonal endpoint must pass through the finite window route*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalizati}$

The handoff exports a located diagonal ledger over finite windows and dyadic budgets; it does not export a global selector, countable choice, quotient stream equality, or a real-completeness theorem.

Proof. Read the carrier of Definition 8146.1. The dyadic row Q and stream-window row W are exposed before any regular rational readback can be consumed.

Next consume the regular sequence and real seal rows R, E. These rows only state that the displayed finite window and tolerance route reaches the named real boundary; they do not supply host equality of streams or selected representatives.

Finally read the extraction sibling row X together with the locatedness row L and structural rows H, C, P, N. The consumer therefore receives only the finite BEDC handoff shown in the theorem. □

Theorem 8146.4 (BishopLocatedDiagonal real seal non-escape)**.** *In an accepted* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ *packet, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *can be consumed only after the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, diagonal-extraction row* X*, and locatedness ledger* L *have all been displayed. No consumer may bypass these rows to obtain ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, or a hidden limit selector.*

Proof. The carrier places E after the tolerance, window, readback, extraction, and locatedness rows. Projection therefore exposes E only as the endpoint of that finite route.

The window handoff of Definition 8146.3 records the same dependency order for consumers. Since the accepted packet contains no coordinate for ambient completeness, quotient real equality, or hidden selection, the real seal cannot be read as any of those stronger objects. □

Theorem 8146.5 (BishopLocatedDiagonal transport stability)**.** *If an accepted* $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ *packet is transported by* hsame *or replayed by* Cont*, then the rows* Q, W, R, E, X, L *remain attached to the same local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$ *route. Thus transport preserves the finite diagonal ledger while refusing countable choice, quotient stream equality, host subsequence identity, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the transport row H and replay row C of the carrier. They act componentwise on the dyadic tolerance, stream-window, regular readback, real seal, extraction, and locatedness rows.

The provenance row P and local name row N record the same dependency anchors after transport. Hence a transported packet still factors through the four dependency certificates and the diagonal-extraction sibling rows before exposing the diagonal ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedDiagonal}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, diagonal-extraction sibling rows, locatedness comparisons, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$, seed)

Scope closed: Definition 8146.1, Definition 8146.2, Definition 8146.3, Definition 8146.4, and Definition 8146.5 seed the finite Bishop-style located diagonal packet over dyadic, stream, regular rational, real, and diagonal-extraction rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, countable choice, quotient stream equality, selected host subsequences, hidden real completeness, or global diagonal compactness.

Upgrade path: Obligation closure requires a Lean carrier for BEDC.Derived.BishopLocatedDiagonalUp and checked rows for tolerance exposure, window handoff, real-seal non-escape, transport, replay, provenance, and local naming.

Chapter 8147

A Concrete Naming Certificate for $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$

Following Crandall and Liggett on nonlinear semigroups generated by accretive operators, the $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ packet records the resolvent–Euler construction surface as finite BEDC data. It sits over the complete-metric carrier of chapter 128, the metric-space carrier of chapter 4230, the ODE and PDE consumer surfaces of chapter 193 and chapter 194, the regular rational readback of chapter 301, the finite stream interface of chapter 126, and the terminal real-name seal of chapter 38. The packet names the displayed nonlinear evolution route; it does not assert host Banach-space completeness, maximal-accretive generation, selected limits, quotient trajectories, or ambient real analysis outside the displayed rows.

Definition 8147.1 (Crandall–Liggett semigroup carrier)**.** *A* $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, M, A, R, E, S, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *state-space row, read together with the underlying* $\mathsf{Met}^{\uparrow}$ *distance comparisons used by every resolvent and Euler step.*
2. *M is the metric-accretive operator row, recording the displayed graph windows, range probes, accretive comparison data, and domain entries available to the packet.*
3. *A is the admissible step-size row, recording the positive rational time steps and their finite subdivision schedule.*
4. *R is the resolvent row, storing the finite resolvent reads $(I + \lambda M)^{-1}$ as named graph-window data rather than as a host inverse operator.*
5. *E is the Euler product row, recording finite products of resolvent steps along the displayed subdivision schedule.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the inspected time windows, initial-state probes, and requested precision windows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the time-step, resolvent, and Euler-product ledgers.*
8. *T is the* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected semigroup time and state-coordinate readbacks.*

9. *H records componentwise* hsame *transport for complete-metric, metric, operator, step-size, resolvent, Euler-product, stream-window, regular-readback, real-seal, replay, provenance, and local-name rows.*
10. *C records displayed* Cont *replay from an initial-state and time request through X, M, A, R, E, S, Q, T.*
11. *P is the* Pkg *provenance over* $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CrandallLiggettSemigroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, A, R, E, S, Q, T, H, C, P, N$. It has no coordinate for a host nonlinear semigroup, selected trajectory limit, quotient curve equality, unbounded-operator domain completion, countable choice over step schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8147.2 (Crandall–Liggett semigroup NameCert obligations)**.** *For every accepted* CrandallLiggettSemigrou *carrier $G = (X, M, A, R, E, S, Q, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CrandallLiggettSemigroup}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete-metric state row, underlying metric comparisons, metric-accretive operator row, admissible step-size row, resolvent row, Euler-product row, finite stream windows, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may replace those rows by a host semigroup, selected limit curve, quotient trajectory equality, unlogged domain closure, countable-choice schedule, or external PDE/ODE well-posedness theorem.*

Proof. Project the carrier of Definition 8147.1. The analytic rows are X, M, A, R, E, S, Q, T: complete-metric and metric state data, operator graph data, admissible rational step sizes, resolvent reads, finite Euler products, stream windows, regular rational readback, and the terminal real seal.

Read the resolvent data before the Euler product. The row R supplies only displayed resolvent graph windows for the step sizes in A, and the row E records finite products of those displayed rows. Thus the packet does not introduce a host inverse operator or a selected infinite product.

The ODE/PDE-facing read is downstream of the finite windows. The $\mathsf{StreamName}^{\uparrow}$ row S fixes inspected times and state probes, the $\mathsf{RegSeqRat}^{\uparrow}$ row Q records the rational readback, and the $\mathsf{Real}^{\uparrow}$ row T seals only the displayed read.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for a host semigroup, selected trajectory, quotient curve equality, unlogged domain completion, choice schedule, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8147.3 (Crandall–Liggett resolvent–Euler route)**.** *Every ODE- or PDE-facing consumer of an accepted* $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ M,\ A,\ R,\ E,\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The route exposes the complete-metric state row, metric comparisons, accretive operator windows, rational step-size schedule, resolvent rows, finite Euler products, stream windows, regular rational readback, real seal, transport, replay, provenance, and local naming before any evolution read is public. It exports no ambient nonlinear semigroup theorem, selected limit, quotient trajectory equality, countable choice, or hidden complete state space.

Proof. Start with an evolution request from an ODE or PDE consumer. The request first reads the complete-metric and metric state rows in X, then the operator graph and accretive comparison data in M.

The time discretization is finite. The admissible step-size row A names the rational subdivisions, R supplies the displayed resolvent rows for those steps, and E records the finite Euler product available at the inspected window.

Only after that finite product is displayed may the consumer read S, Q, T. These rows expose the stream window, regular rational readback, and terminal real seal for the inspected semigroup value.

The structural rows H, C, P, N preserve the same order by componentwise transport, displayed replay, provenance, and local naming. Therefore the public route is the resolvent–Euler NameCert surface, not a host generation theorem or selected trajectory limit. □

Falsifiable prediction: An accepted $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ packet that exports an ODE/PDE evolution read while omitting the resolvent row, Euler-product row, stream-window row, regular rational readback, or terminal $\mathsf{Real}^{\uparrow}$ seal refutes the carrier.

Independence witness: complete-metric, metricspace, ode, pde, streamname, regseqrat, real: $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ is not bijective to any listed sibling because it binds an accretive-operator row, rational step-size schedule, resolvent rows, Euler products, finite windows, readback, real seal, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CrandallLiggettSemigroup}^{\uparrow}}$.

Constructive story: A $\mathsf{CrandallLiggettSemigroup}^{\uparrow}$ packet is generated from a complete-metric state row, metric comparisons, accretive operator graph windows, admissible rational step sizes, resolvent rows, finite Euler products, finite stream windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CrandallLiggettSemigroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8147.1, Definition 8147.2, and Definition 8147.3 seed the finite resolvent–Euler NameCert surface for nonlinear semigroup consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CrandallLiggettSemigroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close maximal-accretive generation, host nonlinear semigroup existence, uniqueness of trajectories, selected limits, quotient curve equality, countable choice over subdivisions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, resolvent-row exactness, Euler-product exactness, finite-window visibility, regular-readback boundary, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CrandallLiggettSemigroupUp.

Chapter 8148

A Concrete Naming Certificate for $\mathsf{CompactRegularCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges on compact metric sources through finite nets and explicit Cauchy moduli, the $\mathsf{CompactRegularCauchyCriterion}^{\uparrow}$ packet records the finite-window criterion used by Real and Completion consumers before any ambient compactness theorem is available. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the compact regular-Cauchy criterion as explicit finite data; it does not identify compactness with open-cover compactness, quotient real equality, trichotomy, or selected limits.

Theorem 8148.1 (Compact regular-Cauchy criterion NameCert obligations)**.** *A local* $\mathsf{NameCert}_{\mathsf{CompactRegularCauchyC}}$ *packet is accepted exactly when it displays a finite carrier*

$$K = (G, W, D, R, E, H, C, P, N)$$

where G is a finite compactness grid row, W is the $\mathsf{StreamName}^{\uparrow}$ *family of sampled windows, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for those windows, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback produced by the grid, E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local naming row. The classifier compares accepted packets only through these rows.*

Proof. Carrier admission is projection from the displayed tuple. The compactness grid G is not a host compact set; it is the finite list of centers, dyadic radii, and grid-cover incidence data that the packet exposes.

The stream and dyadic rows then determine the visible read. For each grid cell, W names the sampled finite windows and D gives the corresponding dyadic tolerance checks, so the regular rational row R is downstream of finite observations and budgets.

The real seal and structural rows are consumer boundaries. The row E is read only after R, and H, C, P, N transport, replay, source, and name the same finite route. No quotient real equality, open-cover compactness principle, trichotomy, selected subsequence, or ambient completeness coordinate can be projected from the carrier. □

Theorem 8148.2 (Compact regular-Cauchy modulus window)**.** *For every accepted* $\mathsf{CompactRegularCauchyCriterior}$ *packet and every requested regular precision m, there is a displayed finite window budget*

$$B_m = (\Gamma_m, \rho_m, \omega_m)$$

such that Γ_m is a finite grid of dyadic centers, ρ_m is a DyadicRatCore$^{\uparrow}$ *radius row, and ω_m is a* StreamName$^{\uparrow}$ *read schedule. Any two sampled regular names in the same grid cell are transported by* hsame *to a common* RegSeqRat$^{\uparrow}$ *readback within the dyadic budget for m before any* Real$^{\uparrow}$ *seal is exposed.*

Proof. Fix the requested precision m. The packet reads the compactness grid coordinate G at precision m and obtains the finite grid Γ_m, dyadic radius row ρ_m, and stream-window schedule ω_m.

Each cell is checked by the dyadic ledger. The row D supplies the tolerance comparison for the sampled endpoints, while W supplies exactly the finite observations named by ω_m.

Now use componentwise hsame transport. If two sampled names land in the same cell, the transport row identifies their displayed window data and dyadic budget data, so the RegSeqRat$^{\uparrow}$ readback is common at precision m.

Only after this finite route has been exposed may the Real$^{\uparrow}$ seal row be read. The argument is a finite-window construction and not a proof by open covers, quotient reals, trichotomy, or choice. □

Theorem 8148.3 (Compact regular-Cauchy L10 scope)**.** *The* CompactRegularCauchyCriterion$^{\uparrow}$ *packet belongs to the L10 Real and Completion scope as a criterion surface over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$. *It supplies finite grids, dyadic budgets, regular readback, and seal order for downstream completion arguments, but it does not assert ambient* Real$^{\uparrow}$ *completeness, open-cover compactness, sequential compactness without explicit data, trichotomy, quotient reals, Lean verification, or bridge checking.*

Proof. The dependency surface is exactly the displayed route named in the theorem. Dyadic budgets come from DyadicRatCore$^{\uparrow}$, finite reads come from StreamName$^{\uparrow}$, regular rational names come from RegSeqRat$^{\uparrow}$, and the public completion-facing boundary is the Real$^{\uparrow}$ seal.

The L10 scope is therefore a finite criterion packet. A downstream consumer may ask for the grid and window budget at a requested precision, then use hsame transport to keep the regular readback stable across the displayed cell data.

Nothing in the packet supplies the excluded coordinates listed in the statement. Completeness, open-cover compactness, data-free sequential compactness, trichotomy, quotient-real equality, Lean verification, and bridge checking remain outside the named scope. □

Closure status for NameCert$_{\text{CompactRegularCauchyCriterion}^{\uparrow}}$.

Constructive story: A CompactRegularCauchyCriterion$^{\uparrow}$ packet is generated from finite compactness grids, StreamName sample windows, DyadicRatCore tolerance ledgers, RegSeqRat regular readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{CompactRegularCauchyCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 8148.1, Definition 8148.2, and Definition 8148.3 seed the finite compact regular-Cauchy criterion packet for the L10 Real and Completion route.

Formal verification: FormalStatus(NameCert$_{\text{CompactRegularCauchyCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, RealUp completeness, sequential compactness without explicit data, trichotomy, quotient reals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite grid exposure, dyadic window budgets, regular readback, hsame transport, Real sealing, replay, provenance, and local naming for BEDC.Derived.CompactRegularCauchyCriterionUp.

Chapter 8149

A Concrete Naming Certificate for $\mathsf{BishopEffectiveRealName}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopEffectiveRealName}^{\uparrow}$ packet records an effective real name as explicit dyadic approximation, precision and modulus visibility, finite StreamName windows, RegSeqRat readback, and RealUp seal rows. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational sequence handoff of chapter 301, the terminal real-name surface of chapter 38, and the fast threshold discipline of chapter 1781. The packet is a finite L10 real-name interface; it does not assert quotient equality of streams or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8149.1 (Bishop effective real-name carrier)**.** *A* $\mathsf{BishopEffectiveRealName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation ledger, storing the dyadic approximants and tolerance accounts exposed to the consumer.*
2. *M is the visible precision and modulus row that tells which dyadic tolerance is requested and which finite window may be opened.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the precision displayed by M.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the dyadic approximants and the opened stream windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after D, M, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the approximation ledger, precision and modulus row, finite window, regular rational readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the effective name from a precision query through D, M, S, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopEffectiveRealName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopEffectiveRealName}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge data.*

Theorem 8149.2 (Bishop effective real-name NameCert obligations)**.** *For every accepted* $\mathsf{BishopEffectiveRealName}$ *carrier*

$$B = (D, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopEffectiveRealName}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation ledger, precision and modulus row,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *approximation-row exactness, requiring D to display the dyadic approximants and tolerance accounts consumed by the readback;*
3. *modulus visibility, requiring M to expose the precision request and the finite threshold before any stream window is opened;*
4. *finite-window exactness, requiring S to be precisely the finite StreamName window opened by the visible row M;*
5. *RegSeqRat handoff, requiring R to read D, M, S before exporting a regular rational name;*
6. *Real seal nonescape, requiring E to consume D, M, S, R before any Real-facing row is public;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same displayed rows through H, C, P, N.*

No Bishop effective real-name obligation is stored outside these rows.

Proof. Project the carrier of Definition 8149.1. The only approximation-bearing coordinates are the dyadic ledger D, the precision and modulus row M, the finite window S, and the regular rational readback R.

The approximation-row exactness and modulus visibility clauses are read before any Real-facing seal is available. The row M supplies the precision query and threshold, while D supplies the dyadic tolerance account for that same query.

The finite-window and handoff clauses then force the consumer through S and R. The window S is opened from the visible threshold, and R reads the displayed dyadic data and window data as a regular rational name.

Finally E, H, C, P, N preserve the same finite route as a Real-facing seal, componentwise transport, replay, provenance, and local naming. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8149.3 (Bishop effective real-name Real seal handoff)**.** *Every Real-facing read of an accepted* $\mathsf{BishopEffectiveRealName}^{\uparrow}$ *carrier first exposes the displayed route*

$$D, M, S, R, E, H, C, P, N.$$

The handoff exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge datum.*

Proof. Start from an accepted carrier $B = (D, M, S, R, E, H, C, P, N)$. The Real seal row E is listed downstream of the dyadic approximation ledger, precision and modulus row, finite StreamName window, and RegSeqRat readback.

Apply Definition 8149.2. Its RegSeqRat handoff and Real seal nonescape clauses require R to consume D, M, S and require E to consume D, M, S, R before a Real-facing row is public.

The structural rows H, C, P, N replay and name the same finite path. No separate coordinate supplies quotient stream equality, a selected limit, choice, trichotomy, host equality, ambient completeness, Lean verification, or bridge data, so the seal cannot bypass the displayed Bishop-style real-name interface. □

Falsifiable prediction: Every $\mathsf{BishopEffectiveRealName}^{\uparrow}$ Real-facing read factors through a DyadicRatCore approximation ledger, a visible precision and modulus row, a finite StreamName window, a RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, fastcauchymodulus: BishopEffectiveRealNam is not bijective to any listed sibling because it jointly carries dyadic approximation, precision visibility, finite-window opening, regular rational readback, and Real-facing seal data as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopEffectiveRealName}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopEffectiveRealName}^{\uparrow}$ packet is generated from a DyadicRatCoreUp approximation ledger, a visible precision and modulus row, a StreamNameUp finite window, a RegSeqRatUp readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopEffectiveRealName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8149.1, Definition 8149.2, and Definition 8149.3 seed the finite Bishop-style effective real-name route from dyadic approximation evidence to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopEffectiveRealName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximation-row exactness, modulus visibility, finite-window exactness, RegSeqRat handoff, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BishopEffectiveRealNameUp.

Chapter 8150

A Concrete Naming Certificate for ProjectionValuedMeasure$^{\uparrow}$

Following von Neumann and the Reed–Simon treatment of spectral families, the ProjectionValuedMeasure$^{\uparrow}$ packet isolates the finite projection-family interface used between SpectralMeasure$^{\uparrow}$ and spectral-theorem consumers. It sits after the spectral-measure carrier of chapter 298, the Hilbert carrier of chapter 91, the bounded self-adjoint operator source of chapter 5160, the event-measure rows of chapter 92, and the consumer statement of chapter 214. Those siblings provide scalar measure windows, Hilbert endpoints, operator sources, event rows, and theorem-facing demand separately; this packet names the projection-valued family before any spectral theorem, functional calculus, or selected decomposition is read.

Definition 8150.1 (Projection-valued measure carrier)**.** *A* ProjectionValuedMeasure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (H, A, E, \Pi, O, F, S, T, C, K, N)$$

with the following displayed rows:

1. *H is the* Hilbert$^{\uparrow}$ *endpoint on which all projection rows act.*
2. *A is the* BoundedSelfAdjointOperator$^{\uparrow}$ *source row whose spectral projections are being requested.*
3. *E is the* Measure$^{\uparrow}$ *event-family row, recording only the displayed finite event requests.*
4. *Π is the projection-family row, assigning each displayed event in E a projection endpoint over H.*
5. *O is the orthogonality row for disjoint displayed event pairs.*
6. *F is the finite-additivity row for displayed finite unions.*
7. *S is the scalar-window handoff row linking the projection family to the* SpectralMeasure$^{\uparrow}$ *scalar measure windows.*
8. *T stores componentwise* hsame *transport for Hilbert, operator, event, projection, orthogonality, additivity, and scalar-window rows.*
9. *C stores displayed* Cont *replay from event requests through projection endpoints and scalar-window handoff.*
10. *K is the* Pkg *provenance over* ProjectionValuedMeasure$^{\uparrow}$, SpectralMeasure$^{\uparrow}$, Hilbert$^{\uparrow}$, BoundedSelfAdjointOperator Measure$^{\uparrow}$, SpectralTheorem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProjectionValuedMeasure}^{\uparrow}}$ *row.*
The carrier contains no coordinate for host sigma-additivity, quotient event equality, selected spectral decomposition, eigenbasis choice, unrestricted functional calculus, or a proof of the spectral theorem.

Definition 8150.2 (Projection-valued measure classifier)**.** *For accepted* $\mathsf{ProjectionValuedMeasure}^{\uparrow}$ *packets P and P', the classifier $P \sim_{\mathsf{ProjectionValuedMeasure}^{\uparrow}} P'$ holds exactly when their displayed Hilbert rows, bounded self-adjoint operator rows, measure-event rows, projection endpoints, orthogonality ledgers, finite-additivity ledgers, scalar-window handoffs, continuation routes, package provenance, and local naming rows compare componentwise by the recorded* hsame *witnesses. The classifier does not compare completed measurable spaces, host projection-valued functions, countable projection families, or spectral theorem objects.*

Theorem 8150.3 (Projection-valued measure NameCert obligations)**.** *For the carrier of Definition 8150.1, the local* $\mathsf{NameCert}_{\mathsf{ProjectionValuedMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for $P = (H, A, E, \Pi, O, F, S, T, C, K, N)$, Hilbert endpoint exposure through H, bounded self-adjoint source exposure through A, event-family exposure through E, projection-family exposure through Π, orthogonality through O, finite additivity through F, scalar-window handoff through S, componentwise transport through T, displayed replay through C, package provenance through K, and local naming through N.*

Proof. Project the finite packet $P = (H, A, E, \Pi, O, F, S, T, C, K, N)$. The first three rows supply the Hilbert endpoint, bounded self-adjoint source, and finite event family on which the projection-valued read is typed.

Read the projection rows next. The projection-family row Π assigns displayed events to projection endpoints, while O and F record orthogonality and finite additivity only for displayed event pairs and finite unions.

The scalar-window handoff S connects the projection ledger to $\mathsf{SpectralMeasure}^{\uparrow}$ without importing a completed measure space. Transport, replay, provenance, and local naming then preserve the same displayed rows, so the listed coordinates exhaust the NameCert obligations. □

Theorem 8150.4 (Projection-valued measure spectral-measure handoff)**.** *Every* $\mathsf{SpectralMeasure}^{\uparrow}$ *or* $\mathsf{SpectralTheorem}^{\uparrow}$ *consumer that reads projection-valued data from an accepted* $\mathsf{ProjectionValuedMeasure}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{Hilbert}^{\uparrow},\ \mathsf{BoundedSelfAdjointOperator}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \Pi,\ O,\ F,\ \mathsf{SpectralMeasure}^{\uparrow}$$

before local naming. The route exposes the projection family, orthogonality, finite additivity, and scalar-window handoff, but it does not expose host sigma-additivity, a quotient event space, an eigenbasis selector, an unrestricted functional calculus, or a spectral theorem proof.

Proof. Start with the carrier and classifier. They expose only the Hilbert row, bounded self-adjoint source row, displayed event family, projection endpoints, orthogonality and finite-additivity ledgers, scalar-window handoff, transport, replay, provenance, and local naming.

The $\mathsf{SpectralMeasure}^{\uparrow}$ sibling is reached through S, the scalar-window handoff row. This keeps scalar measure reads downstream of the projection ledger, rather than letting the spectral-measure chapter carry both the projection family and scalar window roles at once.

Finally the $\mathsf{SpectralTheorem}^{\uparrow}$ consumer reads the same finite package through K and N. Since the packet has no coordinate for completed measurable spaces, selected decompositions, functional calculus, or theorem proof data, the handoff is exhausted by the displayed projection-valued measure route. □

Falsifiable prediction: A spectral-theorem consumer that reads projection-valued data without displaying Hilbert, bounded self-adjoint operator, event-family, projection, orthogonality, finite-additivity, scalar-window, transport, replay, provenance, and local NameCert rows refutes the $\mathsf{ProjectionValuedMeasure}^{\uparrow}$ carrier.

Independence witness: spectralmeasure, hilbert, boundedselfadjointoperator, measure, spectraltheorem: none is bijective to ProjectionValuedMeasureUp because this packet alone binds finite event rows to projection endpoints with orthogonality, finite additivity, scalar-window handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ProjectionValuedMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{ProjectionValuedMeasure}^{\uparrow}$ packet is generated from a Hilbert endpoint, bounded self-adjoint source, displayed measure-event rows, projection endpoints, orthogonality ledgers, finite-additivity ledgers, scalar-window handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProjectionValuedMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8150.1, Definition 8150.2, Definition 8150.3, and Definition 8150.4 seed the finite projection-family interface consumed by SpectralMeasureUp and SpectralTheoremUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProjectionValuedMeasure}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ProjectionValuedMeasureUp`

Bridge status: none

Not claimed: This chapter does not close host sigma-additivity, completed measurable spaces, quotient event equality, selected spectral decompositions, eigenbasis choices, unrestricted functional calculus, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, projection-family exposure, orthogonality, finite additivity, scalar-window handoff, transport, replay, provenance, and local naming for BEDC.Derived.ProjectionValuedMeasur

Chapter 8151

A Concrete Naming Certificate for $\mathsf{PeterWeyl}^{\uparrow}$

Following Peter and Weyl on compact-group representation theory, the $\mathsf{PeterWeyl}^{\uparrow}$ packet records the finite BEDC surface by which matrix coefficients of finite-dimensional unitary representations approximate compact-group functions. It sits after the topological-group rows of chapter 204, the invariant-measure surface of chapter 7873, the finite Grothendieck-ledger rows of chapter 246, the Hilbert carrier of chapter 91, the matrix readback of chapter 49, and the finite Fourier observation interface of chapter 124. The packet names the displayed coefficient-density route; it does not import a host compact group, ambient L^2 completion, selected irreducible list, quotient function equality, or analytic Peter–Weyl theorem as a primitive.

Definition 8151.1 (Peter–Weyl carrier)**.** *A* $\mathsf{PeterWeyl}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, M, R, H, F, A, D, T, C, Q, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{TopGroup}^{\uparrow}$ *compact-group source row, exposing multiplication, inversion, unit, continuity, and compact-readback windows used by the coefficient route.*
2. *M is the* $\mathsf{HaarMeasure}^{\uparrow}$ *row, recording the invariant integration surface and finite measure readbacks consumed by coefficient pairing.*
3. *R is the* $\mathsf{RepresentationRing}^{\uparrow}$ *row, listing the finite representation endpoints, direct-sum rows, tensor rows, and Grothendieck ledger entries available to the packet.*
4. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *row carrying the finite inner-product and norm windows over which coefficient functions are compared.*
5. *F is the* $\mathsf{Fourier}^{\uparrow}$ *finite-observation row for coefficient windows, transform samples, truncation bounds, and integration witnesses.*
6. *A is the* $\mathsf{Mat}^{\uparrow}$ *coefficient row, naming the inspected matrix entries of the displayed representation endpoints.*
7. *D is the density ledger, recording the finite coefficient span, requested tolerance row, approximation witness, and error readback.*
8. *T records componentwise* hsame *transport for group, measure, representation, Hilbert, Fourier, matrix, density, replay, provenance, and local-name rows.*

9. *C records displayed* Cont *replay from a compact-group probe and tolerance request through G, M, R, H, F, A, D.*
10. *Q is the* Pkg *provenance over* $\mathsf{PeterWeyl}^{\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{HaarMeasure}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PeterWeyl}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, M, R, H, F, A, D, T, C, Q, N$. It has no coordinate for a host compact group, a selected complete irreducible basis, quotient function equality, completed L^2 space, ambient Haar integration theorem, or external density proof.

Theorem 8151.2 (Peter–Weyl NameCert obligations)**.** *For every accepted* $\mathsf{PeterWeyl}^{\uparrow}$ *carrier*

$$P = (G, M, R, H, F, A, D, T, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeterWeyl}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact topological-group row, Haar-measure row, representation-ring row, Hilbert row, finite Fourier-observation row, matrix-coefficient row, density ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8151.1. The analytic source rows are G, M, R, H, F, A, D: compact-group operations, invariant measure readback, finite representation endpoints, Hilbert comparison windows, Fourier finite observations, matrix entries, and the coefficient-density ledger.

Read the rows in dependency order. The representation row R is interpreted over the group source G, the coefficient row A is read through the displayed representation endpoints, and the Hilbert and Fourier rows H, F provide only finite pairing and finite-observation windows.

The density ledger D is downstream from those rows. It records a finite coefficient span, a tolerance request, an approximation witness, and an error readback, without adding an ambient basis or a completed L^2 object.

Finally the structural rows T, C, Q, N transport, replay, package, and name the same finite surface. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8151.3 (Peter–Weyl matrix-coefficient density route)**.** *Every density read exported by an accepted* $\mathsf{PeterWeyl}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{TopGroup}^{\uparrow},\ \mathsf{HaarMeasure}^{\uparrow},\ \mathsf{RepresentationRing}^{\uparrow},\ \mathsf{Hilbert}^{\uparrow},\ \mathsf{Fourier}^{\uparrow},\ \mathsf{Mat}^{\uparrow},\ D,\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The route exposes compact group operations, invariant-measure readback, finite representation endpoints, Hilbert comparison windows, finite Fourier observations, inspected matrix entries, and the density ledger before any approximation claim is public. It exports no selected irreducible basis, quotient function equality, completed L^2 carrier, or host Peter–Weyl theorem.

Proof. Start with a density request. The carrier first exposes the compact-group row G and the Haar-measure row M, so every coefficient comparison is attached to the displayed topological-group and invariant-measure surfaces.

Next read the representation and coefficient rows. The finite endpoints in R determine the representation data under inspection, and the matrix row A names only the selected coefficient entries of those endpoints.

The Hilbert and Fourier rows then provide the comparison surface. The row H supplies finite inner-product and norm windows, while F supplies finite coefficient or transform observations with their truncation and integration witnesses.

Only after these rows are displayed may the density ledger D be consumed. The replay, provenance, and local-name rows preserve that same order, so any public density read factors through the stated finite route and not through an ambient analytic theorem. □

Theorem 8151.4 (Peter–Weyl non-escape)**.** *An accepted* $\mathsf{PeterWeyl}^{\uparrow}$ *carrier exports only the rows* $G, M, R, H, F, A, D, T, C, Q, N$ *of Definition 8151.1. It exports no host compact group, selected complete list of irreducible representations, quotient equality of functions, completed Hilbert-space carrier, external Haar-integration theorem, or analytic Peter–Weyl density theorem.*

Proof. Inspect the carrier inventory. Its mathematical rows are the topological-group source, Haar-measure source, representation-ring source, Hilbert comparison row, Fourier finite-observation row, matrix-coefficient row, and density ledger.

Apply the NameCert obligations of Definition 8151.2. Every accepted read remains inside those rows and their structural transport, replay, provenance, and local naming.

Apply the density route of Definition 8151.3. Approximation-facing consumers must read through the finite coefficient route before they may inspect the density ledger. The excluded analytic objects have no carrier coordinate and no continuation route, so they cannot be recovered from the packet. □

Falsifiable prediction: An accepted $\mathsf{PeterWeyl}^{\uparrow}$ packet that exports a density read while omitting the compact-group row, Haar-measure row, representation-ring row, Hilbert comparison row, Fourier finite-observation row, matrix-coefficient row, density ledger, transport, replay, provenance, or local NameCert row refutes the carrier.

Independence witness: topgroup, haarmeasure, representationring, hilbert, fourier, matrix: $\mathsf{PeterWeyl}^{\uparrow}$ is not bijective to any listed sibling because it binds compact-group operations, invariant measure, finite representation endpoints, Hilbert comparison windows, Fourier finite observations, matrix coefficients, and a density ledger in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PeterWeyl}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PeterWeyl}^{\uparrow}$ packet is generated from compact topological-group rows, Haar-measure readback, finite representation-ring endpoints, Hilbert comparison windows, Fourier finite observations, matrix-coefficient rows, a coefficient-density ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PeterWeyl}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8151.1, Definition 8151.2, Definition 8151.3, and Definition 8151.4 seed the finite matrix-coefficient density route over displayed $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{HaarMeasure}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PeterWeyl}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the analytic Peter–Weyl theorem, selected irreducible basis, quotient function equality, completed L^2 carrier, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, matrix-coefficient exactness, density-ledger exactness, Hilbert and Fourier readback boundaries, transport, replay, provenance, and local naming for BEDC.Derived.PeterWeylUp.

Chapter 8152

A Concrete Naming Certificate for $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$

Following Leibniz and Bishop–Bridges, the $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$ packet records differentiation under a Riemann integral as finite BEDC analysis rows rather than as a host analytic principle. It sits beside the Riemann-integral route of chapter 1526, the uniform convergence route of chapter 3417, the completion-modulus route of chapter 6156, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names the finite window in which a parameter derivative and an integral readback share the same displayed modulus ledger before a Real-facing consumer may compare the two rows.

Definition 8152.1 (Differentiation under integral carrier)**.** *A* $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, I, R, \Delta, M, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the parameter row, naming the finite parameter window and the parameter probe at which the derivative is requested.*
2. *I is the interval row, naming the compact interval over which the Riemann integral is read.*
3. *R is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row, containing the finite tagged-partition and integral-readback data.*
4. *Δ is the derivative row, naming the finite difference-window data for the parameter derivative of the displayed integrand.*
5. *M is the shared modulus row governing both the integral approximation and the derivative-window comparison.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected parameter, interval, partition, and tolerance windows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to R, Δ, M, S.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the readback row Q and the shared modulus row M have been displayed.*
9. *H records componentwise* hsame *transport for parameter, interval, Riemann-integral, derivative, shared-modulus, stream-window, readback, real-seal, replay, provenance, and local-name rows.*
10. *C records displayed* Cont *replay from a finite parameter-tolerance request through $A, I, R, \Delta, M, S, Q, E$.*

11. *P is the* Pkg *provenance over* $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, I, R, \Delta, M, S, Q, E, H, C, P, N$. It has no coordinate for measure-theoretic dominated convergence, host function extensionality, quotient equality of functions or streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached derivative theorem outside the finite window.*

Theorem 8152.2 (Differentiation under integral NameCert obligations)**.** *For every accepted* DifferentiationUnderIn *carrier $D = (A, I, R, \Delta, M, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission for the parameter, interval, Riemann-integral, derivative, shared-modulus, stream-window, regular-readback, real-seal, transport, replay, provenance, and local naming rows; shared-modulus exactness requiring R and Δ to consume the same M row; finite-window exactness through S; readback exactness through Q; and non-escape from measure-theoretic dominated convergence, host extensionality, quotient equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and detached derivative data.*

Proof. Project the carrier. The only analytic rows visible to the classifier are the parameter row A, interval row I, Riemann-integral row R, derivative row Δ, shared-modulus row M, stream-window row S, readback row Q, and real-seal row E.

The shared-modulus clause is read before the derivative and integral rows can be compared. The packet therefore treats the requested Leibniz-style equality as a finite window route through R, Δ, M, S, Q, E, not as an ambient theorem about all functions.

Rows H, C, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for dominated convergence, host extensionality, quotient equality, countable choice, ambient completeness, or detached derivative data, the listed obligations exhaust the local NameCert surface. □

Theorem 8152.3 (Differentiation under integral finite-window route)**.** *Every Real-facing read of an accepted* $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$ *carrier factors through the finite route*

$\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, E

The route supplies a finite parameter window, compact interval data, Riemann-integral approximation, derivative-window comparison, shared modulus, regular rational readback, and terminal real-name seal. It exports no measure-theoretic dominated-convergence theorem, host function extensionality, quotient equality, countable-choice schedule, or ambient real-completeness principle.

Proof. Start with the NameCert obligations of Definition 8152.2. They identify the integral and derivative rows as displayed carrier rows that must share the same modulus before a Real-facing comparison is public.

The consumer next opens only the finite stream windows in S. Those windows determine the regular rational readback Q from the Riemann-integral and derivative rows under the shared modulus M, and only then can the terminal seal E be consumed.

The completion-modulus, transport, replay, provenance, and local naming rows preserve that same order. Thus the read remains a finite BEDC analysis route through the displayed rows and never becomes a host dominated-convergence, extensionality, quotient, choice, or completeness principle. □

Theorem 8152.4 (Differentiation under integral L10 seal nonescape)**.** *An accepted* $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and shared-modulus rows. Any consumer that asks for a real equality without exposing* M, S, Q, E *has left the packet surface and cannot cite the local* $\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$ *row.*

Proof. Use the finite-window route of Definition 8152.3. The route places the stream windows before regular rational readback and places the readback before the Real seal.

Apply the obligation surface of Definition 8152.2. It keeps the shared modulus visible throughout the Riemann-integral and derivative comparison, so a consumer that omits M, S, Q, E lacks the rows required for local naming.

Therefore the L10 seal is a nonescape boundary. The packet can export only the finite readback-and-seal route displayed in the carrier, not a detached real-equality theorem or an ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{DifferentiationUnderIntegral}^{\uparrow}$ packet is generated from parameter, interval, Riemann-integral, derivative, shared-modulus, finite StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$, seed)

Scope closed: Definition 8152.1, Definition 8152.2, Definition 8152.3, and Definition 8152.4 seed the finite-window differentiation-under-integral packet for Real-facing readback consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DifferentiationUnderIntegral}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure-theoretic dominated convergence, host function extensionality, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, finite-window exactness, regular-readback boundary, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.DifferentiationUnderInt

Chapter 8153

A Concrete Naming Certificate for $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$

Following Bishop and Bridges on metric completions, the $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ packet records the terminal finite seal by which a uniformly controlled map between displayed metric sources is read as a completion-facing functorial handoff. It sits after the metric-completion carrier of chapter 1506, the functoriality packet of chapter 3755, the dense-embedding surface of chapter 3351, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a single finite terminal-seal object; it does not assert quotient completion equality, selected representatives, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8153.1 (Metric completion functorial seal carrier)**.** *A* $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source-completion row, naming the displayed source and target completion boundaries.*
2. *F is the* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *functoriality row, recording the map graph and the finite continuity control used by the completion handoff.*
3. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row, exposing the dense source embeddings that must be read before the completed map is sealed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the source and target approximation requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the opened windows as regular rational source and target names.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal, visible only after M, F, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the completion, functoriality, dense-embedding, window, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same completion-facing functorial route.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, F, D, W, R, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host function equality, global extension uniqueness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8153.2 (Metric completion functorial seal NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ *carrier*

$$S = (M, F, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}}$ *surface consists exactly of completion-boundary admission, functorial map-row admission, dense-embedding exposure, finite StreamName window inspection, RegSeqRat readback, terminal Real seal discipline, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing consumer must read the displayed route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionFunctor}^{\uparrow} \longmapsto \mathsf{MetricCompletionDenseEmbedding}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

before the functorial seal is public.

Proof. Project the carrier of Definition 8153.1. The completion boundary, functorial map row, dense embedding row, finite stream window, regular rational readback, and Real seal are the only analytic coordinates.

The completion consumer first reads M, F, D. These rows display the source and target completion boundaries, the map graph with its continuity control, and the dense embeddings whose images are being sealed.

The finite observation rows then mediate the Real-facing read. The window row W and readback row R expose only inspected stream fragments and regular rational names before E becomes visible.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for quotient completion equality, selected representatives, countable choice, host function equality, global extension uniqueness, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8153.3 (Metric completion functorial seal dense readback)**.** *Every dense-source read of an accepted* $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ *carrier must expose the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row before the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal are consumed. Thus the completed map is available only through the displayed dense embedding and regular readback route, not through a selected completion representative, quotient stream equality, host function equality, or ambient completeness theorem.*

Proof. Start with the NameCert obligations of Definition 8153.2. They place the dense-embedding row D between the functorial completion row F and the window/readback rows W, R.

The dense row controls the public source of the readback. A consumer may inspect the completed map only after the source embedding has been displayed, and the subsequent StreamName and RegSeqRat rows record the finite approximation data actually read.

The terminal seal E and structural rows H, C, P, N preserve that same route. No carrier coordinate supplies a selected representative, quotient stream equality, host function equality, or ambient completeness principle, so the dense readback remains the displayed finite handoff. □

Falsifiable prediction: Every $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ completion-facing read exposes MetricCompletionUp, MetricCompletionFunctorUp, MetricCompletionDenseEmbeddingUp, StreamNameUp, RegSeqRatUp, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the functorial seal is public.

Independence witness: metriccompletion, metric-completion-functoriality, metric-completion-dense-embedding, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ because this packet jointly carries the completion boundary, functorial map row, dense embedding row, finite observation window, regular rational readback, and terminal Real seal as one NameCert object.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}$ packet is generated from MetricCompletionUp source and target boundaries, a MetricCompletionFunctorUp map row, a MetricCompletionDenseEmbeddingUp dense-source row, StreamNameUp windows, RegSeqRatUp readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8153.1, Definition 8153.2, and Definition 8153.3 seed the finite terminal-seal route for metric-completion functorial reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctorialSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, host function equality, global extension uniqueness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, functorial map-row exactness, dense-embedding exposure, StreamName window inspection, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionFunctorialSealUp.

Chapter 8154

A Concrete Naming Certificate for $\mathsf{HermiteFunction}^{\uparrow}$

Following Hermite and Fourier analysis, the $\mathsf{HermiteFunction}^{\uparrow}$ packet records the finite normalized Hermite-function surface built from the Hermite polynomial recurrence, a Gaussian envelope ledger, Schwartz-space admission, Fourier-transform handoff, and tempered-distribution action rows. It sits beside the polynomial recurrence packet of chapter 6236, the transform surface of chapter 4025, the rapidly decreasing test-space carrier of chapter 5877, and the distributional action route of chapter 2310. The packet names the finite certificate interface only; it does not assert Hilbert-space completeness, spectral decomposition, quotient equality of functions, or a selected ambient basis.

Definition 8154.1 (Hermite-function carrier)**.** *A* $\mathsf{HermiteFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, G, S, T, D, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{HermitePolynomial}^{\uparrow}$ *source row, exposing the finite polynomial seed and recurrence window that supplies the displayed Hermite polynomial.*
2. *G is the Gaussian envelope row, recording the finite decay and normalization budget attached to the polynomial source.*
3. *S is the* $\mathsf{SchwartzSpace}^{\uparrow}$ *admission row, opened only after P and G display the rapid-decay budget consumed by the test-space carrier.*
4. *T is the* $\mathsf{FourierTransform}^{\uparrow}$ *handoff row, recording the finite transform-read route for the displayed Hermite-function window.*
5. *D is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *action row, exposing only the distribution-facing read obtained from the same Schwartz and transform rows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *scalar readback row for normalization, phase, and finite coefficient data.*
7. *H records componentwise* hsame *transport for the polynomial source, Gaussian envelope, Schwartz admission, transform handoff, distribution action, scalar readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route from P, G through S, T, D, R, H, Q, N.*

9. *Q is the* Pkg *provenance over* $\mathsf{HermiteFunction}^{\uparrow}$, $\mathsf{HermitePolynomial}^{\uparrow}$, $\mathsf{FourierTransform}^{\uparrow}$, $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HermiteFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, G, S, T, D, R, H, C, Q, N$. It has no coordinate for a completed L^2 basis, global spectral theorem, quotient function equality, distribution dual completeness, selected representatives, Lean verification, or standard-bridge data.

Theorem 8154.2 (Hermite-function NameCert obligations)**.** *For every accepted* $\mathsf{HermiteFunction}^{\uparrow}$ *carrier*

$$F = (P, G, S, T, D, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{HermiteFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{HermitePolynomial}^{\uparrow}$ *source, Gaussian envelope,* $\mathsf{SchwartzSpace}^{\uparrow}$ *admission,* $\mathsf{FourierTransform}^{\uparrow}$ *handoff,* $\mathsf{TemperedDistribution}^{\uparrow}$ *action, scalar readback, transport, replay, provenance, and local naming rows;*
2. *polynomial-source exactness, requiring P to expose the same Hermite recurrence window consumed by the envelope row G;*
3. *Schwartz admission, requiring S to read P and G before a rapidly decreasing function-space row is public;*
4. *Fourier handoff, requiring T to consume the displayed polynomial, envelope, and Schwartz rows before any transform-facing read is exported;*
5. *tempered-distribution nonescape, requiring D to read the same S, T route before any distributional action is public;*
6. *scalar readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same rows through R, H, C, Q, N.*

No Hermite-function obligation is stored outside these rows.

Proof. Project the carrier of Definition 8154.1. The only analytic source rows are the Hermite polynomial window P, the Gaussian envelope G, the Schwartz admission row S, the transform handoff T, and the distribution action row D.

The polynomial-source and envelope clauses force the recurrence data to be visible before any function-space read is available. The Schwartz admission row can therefore expose rapid decay only through the displayed P, G pair.

The Fourier and tempered-distribution rows then use the same path. The row T reads P, G, S before exporting a transform-facing observation, and D reads S, T before exposing a distributional action.

Finally R, H, C, Q, N preserve the same finite route as scalar readback, componentwise transport, replay, provenance, and local naming. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8154.3 (Hermite-function Fourier-transform handoff)**.** *Every Fourier-facing or tempered-distribution-facing read of an accepted* $\mathsf{HermiteFunction}^{\uparrow}$ *carrier first exposes the finite route*

$$P, G, S, T, D, R, H, C, Q, N.$$

The handoff exports no completed L^2 basis, global spectral theorem, quotient function equality, distribution dual completeness, selected representative, Lean verification, or standard-bridge datum.

Proof. Start from an accepted carrier $F = (P, G, S, T, D, R, H, C, Q, N)$. The transform row T is listed downstream of the Hermite polynomial source, Gaussian envelope, and Schwartz admission rows.

Apply Definition 8154.2. Its Schwartz admission and Fourier handoff clauses require S to consume P, G and require T to consume P, G, S before a transform-facing observation is public.

The tempered-distribution row D is downstream of the same route, and R, H, C, Q, N replay and name that route without adding any extra analytic coordinate. Thus every Fourier-facing or distribution-facing read factors through the displayed Hermite-function packet. □

Falsifiable prediction: Every accepted $\mathsf{HermiteFunction}^{\uparrow}$ Fourier-facing or tempered-distribution-facing read factors through a HermitePolynomialUp source row, a Gaussian envelope ledger, a SchwartzSpaceUp admission row, a FourierTransformUp handoff, a TemperedDistributionUp action row, scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: hermite-polynomial, fourier-transform, schwartz-space, tempered-distribution: $\mathsf{HermiteFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries polynomial recurrence, Gaussian envelope, function-space admission, transform handoff, distribution action, scalar readback, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HermiteFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{HermiteFunction}^{\uparrow}$ packet is generated from a HermitePolynomialUp source row, a Gaussian envelope ledger, a SchwartzSpaceUp admission row, a FourierTransformUp handoff, a TemperedDistributionUp action row, scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HermiteFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8154.1, Definition 8154.2, and Definition 8154.3 seed the finite Hermite-function route from polynomial recurrence and Gaussian decay to Fourier and tempered-distribution reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HermiteFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hilbert-space completeness, spectral decomposition, quotient equality of functions, completed distribution duality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial-source exactness, Schwartz admission, Fourier handoff, tempered-distribution nonescape, scalar readback, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.HermiteFunctionUp.

Chapter 8155

A Concrete Naming Certificate for $\mathsf{WeierstrassMTestWindow}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the standard Weierstrass M-test, the $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ packet records the finite majorant-window data needed before a function-series consumer may read a regular Cauchy handoff or terminal real-name seal. It sits beside the M-test carrier of chapter 399, the real-series source of chapter 1581, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the constructive majorant-window route as finite BEDC rows; it does not import countable choice, arbitrary infinite sums, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8155.1 (Weierstrass M-test finite-window carrier)**.** *A* $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, M, B, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite function-series window, listing only the summand windows and source indices inspected by this packet.*
2. *M is the displayed positive majorant sequence for the same inspected indices, written as a finite read of positive comparison rows rather than as an ambient infinite sum.*
3. *B is the finite domination ledger matching each inspected summand window in F to its corresponding positive majorant row in M.*
4. *T is the finite partial-tail budget obtained from B and the displayed majorant tail route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window surface through which the source summand windows and partial-tail requests are observed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced after the partial-tail budget has been exposed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through R.*
8. *H records componentwise* hsame *transport for source windows, majorant rows, domination ledgers, partial-tail budgets, stream windows, regular readbacks, real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{WeierstrassMTestWindow}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WeierstrassMTestWindow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, M, B, T, S, R, E, H, C, P, N$. It has no coordinate for countable choice, an arbitrary infinite sum object, quotient real equality, selected limits, host equality of functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8155.2 (Weierstrass M-test finite-window NameCert obligations)**.** *For every accepted* $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ *carrier*

$$W = (F, M, B, T, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeierstrassMTestWindow}^{\uparrow}}$ *surface consists exactly of finite function-series window exposure through F, positive majorant-row exposure through M, domination ledger exposure through B, finite partial-tail budgeting through T, StreamName observation through S, RegSeqRat readback through R, RealUp sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8155.1. The rows F, M, B, T expose the source windows, positive majorant sequence rows, domination ledger, and finite partial-tail budget before any convergence-facing read can be served.

Use the sibling surfaces in order. The source series is the real-series surface of chapter 1581; the majorant route is the M-test packet of chapter 399; the finite observations are StreamName windows from chapter 126; and the regular readback is the $\mathsf{RegSeqRat}^{\uparrow}$ surface of chapter 301.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, arbitrary infinite sums, quotient real equality, selected limits, host function equality, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8155.3 (Weierstrass M-test finite-window Cauchy-bound route)**.** *Every Cauchy-bound or Real-facing consumer of an accepted* $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{WeierstrassMTest}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The partial-tail budget is therefore read as a finite majorant-window certificate before the terminal real seal is consumed. The route exports no countable-choice schedule, arbitrary infinite sum, quotient real equality, selected limit, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 8155.2. It forces every consumer read through the source window, positive majorant rows, domination ledger, finite partial-tail budget, StreamName observation, RegSeqRat readback, and RealUp seal before the local certificate can be inspected.

The $\mathsf{WeierstrassMTest}^{\uparrow}$ sibling supplies the majorant-tail route, while $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ keep each observation and tolerance request finite. Thus the Cauchy-bound read is a displayed finite-window route, not an appeal to an unrecorded infinite summation theorem.

The terminal row E is only the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The absence of countable-choice, arbitrary-sum, quotient-real, selector, host-equality, and ambient-completeness coordinates prevents any such object from entering the exported route. □

Falsifiable prediction: Every accepted $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ read exposes finite function-series windows, positive majorant rows, finite domination ledgers, partial-tail budgets, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any Cauchy-bound or Real-facing consumer is served.

Independence witness: realseries, weierstrassmtest, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ because this packet alone binds the finite source window, positive majorant window, domination ledger, partial-tail budget, observation route, readback, seal, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{WeierstrassMTestWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{WeierstrassMTestWindow}^{\uparrow}$ packet is generated from finite function-series windows, displayed positive majorant rows, a finite domination ledger, a partial-tail budget, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeierstrassMTestWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8155.1, Definition 8155.2, and Definition 8155.3 seed the finite majorant-window route for the Weierstrass M-test.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeierstrassMTestWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, arbitrary infinite sums, quotient real equality, selected limits, host function equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite source-window exposure, positive majorant-row exposure, domination-ledger exactness, partial-tail budgeting, StreamName observation, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeierstrassMTestWindowUp.

Chapter 8156

A Concrete Naming Certificate for GaussDivergenceTheorem$^{\uparrow}$

Following Gauss and Stokes, the GaussDivergenceTheorem$^{\uparrow}$ packet records the finite flux-boundary route by which a displayed vector field, its divergence row, an oriented boundary, and compatible integration readbacks are named as one boundary-flux certificate. It sits beside the differential-form surface of chapter 136, the manifold carrier of chapter 96, the tagged integration boundary of chapter 1526, the terminal scalar seal of chapter 38, and the graph/readback discipline of chapter 130. The packet is a finite BEDC route for the classical divergence theorem; it does not import measure completion, host vector calculus, selected representatives, quotient equality of fields, or ambient completeness of Real$^{\uparrow}$.

Definition 8156.1 (Gauss divergence theorem carrier)**.** *A* GaussDivergenceTheorem$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, V, D, B, W, I, T, R, P, N)$$

with the following displayed rows:

1. *M is the manifold and chart-window row, read through* Manifold$^{\uparrow}$ *and carrying only the finite domain windows visible to the certificate;*
2. *V is the vector-field row, read through* ContinuousMap$^{\uparrow}$ *and the differential-form interface* DiffForm$^{\uparrow}$ *without adding a host vector-field object;*
3. *D is the divergence row obtained from the same displayed vector-field and chart rows;*
4. *B is the oriented-boundary row for the domain window, including incidence, normal-orientation, and boundary-chart data;*
5. *W is the domain-window row that keeps the interior integration request and the boundary integration request tied to the same finite region;*
6. *I is the integral and readback row, routing both the divergence integral and the boundary flux integral through* RiemannIntegral$^{\uparrow}$ *and the terminal* Real$^{\uparrow}$ *seal;*
7. *T records componentwise* hsame *transport for the manifold, vector-field, divergence, boundary, window, and integration rows;*
8. *R records displayed* Cont *replay from a chart-window request through the vector-field, divergence, boundary, and integral readback rows;*

9. *P is the* Pkg *provenance over* $\mathsf{GaussDivergenceTheorem}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{GaussDivergenceTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host divergence theorem, a completed measure space, a selected boundary parametrization, quotient equality of vector fields, or ambient real completeness.

Theorem 8156.2 (Gauss divergence theorem NameCert obligations)**.** *For every accepted* GaussDivergenceTheorem *carrier $G = (M, V, D, B, W, I, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GaussDivergenceTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for the manifold chart window, vector-field row, divergence row, oriented-boundary row, domain-window row, integral and readback row, componentwise transport, displayed replay, provenance, and local naming. The obligations are discharged only over the finite route*

$$M, V, D, B, W, I, T, R, P, N.$$

No local obligation is stored in a host vector-calculus theorem, a measure-completion carrier, a quotient field equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 8156.1. The analytic rows are exactly M, V, D, B, W, I: the finite chart window, displayed vector field, divergence row, oriented boundary, shared domain window, and integration readback.

The divergence and boundary rows remain tied to the same source. The row D is read from V over the chart data in M, while B and W keep the boundary and interior integration requests on one finite domain window.

Finally attach T, R, P, N. These structural rows transport, replay, source, and name the same packet. Since none of the rows stores host vector calculus, measure completion, quotient equality, selected representatives, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8156.3 (Gauss divergence theorem flux-boundary route)**.** *Every* $\mathsf{GaussDivergenceTheorem}^{\uparrow}$ *consumer that reads a boundary-flux statement must first read the finite route*

$\mathsf{Manifold}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert

through the rows $M, V, D, B, W, I, T, R, P, N$ of Definition 8156.1. The divergence integral and the oriented boundary flux are therefore two readbacks of the same displayed finite window, not a host vector-calculus theorem, a completed measure-space comparison, a selected boundary parametrization, or a quotient equality of vector fields.

Proof. Begin with the carrier inventory. The manifold row M and domain-window row W make the finite region visible before either integral read is public.

Next read the vector-field and divergence rows. The vector-field row V is routed through $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{DiffForm}^{\uparrow}$, and the divergence row D is a displayed row over the same chart data. The boundary row B supplies the oriented boundary needed by the flux read.

The integral row I then routes the interior divergence read and the boundary flux read through $\mathsf{RiemannIntegral}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows T, R, P, N preserve that route and add no host theorem, measure completion, quotient equality, selected representative, or ambient completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{GaussDivergenceTheorem}^{\uparrow}$ packet whose oriented-boundary row or domain-window row is removed loses the displayed flux-boundary route; with both rows present, every flux read factors through the same finite carrier.

Independence witness: diffform, manifold, riemannintegral, real, continuousmap: none is bijective with $\mathsf{GaussDivergenceTheorem}^{\uparrow}$ because the packet jointly carries vector-field, divergence, oriented-boundary, domain-window, integral readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GaussDivergenceTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{GaussDivergenceTheorem}^{\uparrow}$ packet is generated from a displayed manifold chart window, vector-field row, divergence row, oriented-boundary row, domain-window row, integral and readback row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaussDivergenceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8156.1, Definition 8156.2, and Definition 8156.3 seed the finite flux-boundary packet over displayed manifold, vector-field, divergence, oriented-boundary, domain-window, integration, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaussDivergenceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, measure completion, host vector calculus, selected representatives, quotient equality of vector fields, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, divergence-row exactness, oriented-boundary compatibility, domain-window stability, integral readback, transport, replay, provenance, and ledger non-escape for BEDC.Derived.GaussDiverge

Chapter 8157

A Concrete Naming Certificate for $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ packet records the finite stability handoff by which a displayed dyadic absolute-value row remains usable when a regular rational or real-name consumer replays the same tolerance windows. It sits after the dyadic source surface of chapter 291, the rational source of chapter 37, the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite absolute-value endpoint packet of chapter 293. The packet names a stability route for absolute-value tolerance rows; it is not a host rational absolute value, quotient equality, a selected representative, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8157.1 (Dyadic absolute-value stability carrier)**.** *A* $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, A, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row whose dyadic endpoint and tolerance ledger are being inspected.*
2. *A is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint-and-tolerance row attached to D.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which downstream observers request dyadic endpoint approximations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same stream windows and absolute-value tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, A, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic source, absolute-value row, stream windows, regular readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a stability request through*

$$D, A, S, Q, E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValueStability}^{\uparrow}}$ *row.*
The carrier contains no host rational absolute-value function, quotient equality of rational or real names, selected representative, trichotomy witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 8157.2 (Dyadic absolute-value stability NameCert obligations)**.** *For every accepted* $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ *carrier* $T = (D, A, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValueStability}^{\uparrow}}$ *surface consists exactly of dyadic-source admission through D, absolute-value row exposure through A, StreamName window exposure through S, RegSeqRat readback through Q, Real seal non-escape through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8157.1. The analytic rows are the dyadic source D, absolute-value row A, stream-window row S, regular readback Q, and terminal seal E; the structural rows are H, C, P, N.

The dyadic-source and absolute-value obligations are inherited as displayed rows. A consumer first opens D as the dyadic endpoint and tolerance ledger, then opens A as the finite absolute-value endpoint-and-tolerance row attached to that same source.

The readback obligations are finite. The row S supplies the inspected windows, Q records the regular rational readback obtained from those windows, and E is downstream of D, A, S, Q rather than an independent completed real.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since the carrier has no coordinate for host rational absolute value, quotient equality, selected representatives, trichotomy, or ambient completeness, no local NameCert obligation can project one of those objects. □

Theorem 8157.3 (Dyadic absolute-value stability RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ *packet reads the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicAbsoluteValue}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N.$$

The handoff preserves the displayed absolute-value tolerance ledger from the dyadic source through the stream windows and regular readback before any terminal real seal is consumed. It exports no host rational absolute value, quotient equality, selected representative, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8157.2. The accepted consumer must expose D, A, S, Q, E and the structural rows before the packet can be read.

Read D through $\mathsf{DyadicRatCore}^{\uparrow}$ and the rational source surface of $\mathsf{Rat}^{\uparrow}$. The absolute-value endpoint and tolerance ledger are then supplied by A, the accepted $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ row attached to the same displayed dyadic source.

Next read the finite windows S and the regular rational handoff Q. This is the only route by which a $\mathsf{RegSeqRat}^{\uparrow}$ consumer sees the absolute-value tolerance data, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream of that regular readback.

Therefore the handoff is stable under the displayed hsame transport and Cont replay rows. The absent carrier coordinates rule out any hidden host absolute value, quotient equality, selected representative, trichotomy, or ambient completeness principle. □

Falsifiable prediction: Any accepted $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without displaying the dyadic source, absolute-value row, StreamName windows, transport, replay, provenance, and local naming rows refutes the packet.

Independence witness: dyadicratcore, rat, streamname, regseqrat, real, dyadicabsolutevalue: no listed sibling supplies the whole nine-row packet $T = (D, A, S, Q, E, H, C, P, N)$, because each sibling contributes only source, window, readback, seal, or absolute-value rows while this chapter binds their stability route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValueStability}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{DyadicAbsoluteValueStability}^{\uparrow}$ packet is generated from a DyadicRatCore source row, a DyadicAbsoluteValue endpoint-and-tolerance row, finite StreamName windows, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValueStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8157.1, Definition 8157.2, and Definition 8157.3 seed the finite stability handoff from dyadic absolute-value tolerance rows to RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValueStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host rational absolute value, Real completeness, trichotomy, quotient equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic-source admission, absolute-value row exposure, StreamName window exposure, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicAbsoluteValueStabilityUp.

Chapter 8158

A Concrete Naming Certificate for MonotoneRealFunction$^{\uparrow}$

Following Bishop and Bridges on constructive real functions of bounded variation, the MonotoneRealFunction$^{\uparrow}$ packet records a displayed monotone real-function route as finite NameCert data. It reads endpoint and value names through Real$^{\uparrow}$ and chapter 38, finite approximation windows through StreamName$^{\uparrow}$ and chapter 126, rational readback through RegSeqRat$^{\uparrow}$ and chapter 301, order comparisons through TotalOrder$^{\uparrow}$ and chapter 54, bounded-window control through BoundedRealSequence$^{\uparrow}$ and chapter 1415, and cumulative variation handoff through TotalVariationFunction$^{\uparrow}$ and chapter 2257. The packet names the finite monotonicity and variation ledger; it is not a selected-limit theorem, a trichotomy principle for arbitrary reals, a quotient of functions, or a use of ambient real completeness.

Definition 8158.1 (Monotone real function carrier)**.** *A* MonotoneRealFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, W, V, O, B, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the source interval and finite window schedule, including the displayed endpoint names and the inspected input rows.*
2. *W is the* StreamName$^{\uparrow}$ *window row for the inspected source positions and value requests.*
3. *V is the* Real$^{\uparrow}$ *value row for the displayed function values reached from I and W.*
4. *O is the* TotalOrder$^{\uparrow}$ *comparison ledger witnessing monotonicity for each displayed ordered pair of source windows.*
5. *B is the* BoundedRealSequence$^{\uparrow}$ *bounded-window row attached to the finite value sequence read by the packet.*
6. *L is the variation-bound ledger, recording the finite nonnegative steps permitted by the monotone comparison rows and the handoff boundary toward* TotalVariationFunction$^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for interval, window, value, order-comparison, bounded-window, variation-ledger, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from an inspected source pair through the window row, value row, order comparison, bounded-window row, and variation ledger.*

9. *P is the* Pkg *provenance over* MonotoneRealFunction$^\uparrow$, Real$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, TotalOrder$^\uparrow$, BoundedRealSequence$^\uparrow$, TotalVariationFunction$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MonotoneRealFunction}^\uparrow}$ *row.*

The classifier compares accepted packets only through $I, W, V, O, B, L, H, C, P, N$. It has no coordinate for selected limits, real trichotomy, quotient equality of functions, countable choice, an external variation supremum, or ambient Real$^\uparrow$ *completeness.*

Theorem 8158.2 (Monotone real function NameCert obligations)**.** *For every accepted* MonotoneRealFunction$^\uparrow$ *carrier $M = (I, W, V, O, B, L, H, C, P, N)$, the local* NameCert$_{\text{MonotoneRealFunction}^\uparrow}$ *surface consists exactly of carrier admission for the displayed interval and window rows,* StreamName$^\uparrow$ *windows,* Real$^\uparrow$ *value rows,* TotalOrder$^\uparrow$ *monotonicity comparisons,* BoundedRealSequence$^\uparrow$ *bounded-window row, variation-bound ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each monotonicity read must use the same source-window row I, W and the same value row V before it may consume the comparison ledger O or the variation ledger L.*

Proof. Project the carrier of Definition 8158.1. The only analytic rows exposed by the packet are I, W, V, O, B, L.

The monotonicity clause is finite. For each displayed ordered pair of source windows, the source interval and stream rows determine the two value rows, and the order comparison is then read from O as a TotalOrder$^\uparrow$ row over those displayed Real$^\uparrow$ values.

The remaining rows do not add analytic content. They transport, replay, source, and name the same finite packet through H, C, P, N, so the listed rows exhaust the local NameCert surface. □

Theorem 8158.3 (Monotone real function Real seal boundary)**.** *Every* Real$^\uparrow$ *value read exported by an accepted* MonotoneRealFunction$^\uparrow$ *carrier factors through the finite route*

$$\text{StreamName}^\uparrow,\ \text{RegSeqRat}^\uparrow,\ \text{Real}^\uparrow,\ \text{TotalOrder}^\uparrow,\ \text{BHist},\ \text{hsame},\ \text{Cont},\ \text{Pkg},\ \text{NameCert}.$$

The route exposes source windows, rational readback, real-value seals, and displayed order comparisons before any monotone-function consumer can read the value. It exports no selected limit, quotient function equality, countable-choice schedule, trichotomy premise, or ambient real-completeness object.

Proof. Start from an accepted carrier $M = (I, W, V, O, B, L, H, C, P, N)$. A value read first opens the source interval and stream windows I, W, then reads the rational approximation and terminal real seal represented by V.

Only after the value rows are displayed may a monotone consumer inspect O. The transport and continuation rows keep this order attached to the same local name, while provenance records the visible dependency surface.

Since no row of the carrier stores selected limits, quotient functions, countable choice, trichotomy, or ambient completeness, the public Real-facing seal boundary is exactly the displayed finite route. □

Theorem 8158.4 (Monotone real function variation-ledger handoff)**.** *Every* TotalVariationFunction$^\uparrow$ *or bounded-variation consumer of an accepted* MonotoneRealFunction$^\uparrow$ *carrier receives the finite variation-bound ledger L only after the displayed source windows,* Real$^\uparrow$ *value rows,* TotalOrder$^\uparrow$ *monotonicity comparisons, and* BoundedRealSequence$^\uparrow$ *bounded-window row have been exposed. Thus the variation handoff is a finite monotone ledger over displayed rows, not a supremum over all partitions, quotient equality of real-valued functions, selected endpoint limit, countable-choice schedule, or ambient* Real$^\uparrow$ *completeness theorem.*

Proof. The variation ledger is downstream of the monotonicity comparisons. The carrier first supplies I, W, V, then O records the finite order comparisons for displayed source pairs, and B records the bounded window of values used by the ledger.

The handoff row L can therefore be consumed only as a finite ledger attached to those rows. The continuation and provenance rows route this ledger toward $\mathsf{TotalVariationFunction}^{\uparrow}$ without adding an external partition supremum or host function object.

The exclusions follow from the carrier boundary: selected limits, quotient function equality, countable choice, ambient completeness, and unrestricted variation suprema are absent from $I, W, V, O, B, L, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneRealFunction}^{\uparrow}$ packet is generated from source interval windows, StreamName request rows, RealUp value rows, TotalOrderUp comparison rows witnessing monotonicity, a BoundedRealSequenceUp bounded-window row, a finite variation-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8158.1, Definition 8158.2, Definition 8158.3, and Definition 8158.4 seed the finite monotone real-function packet for Real and variation-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, real trichotomy, quotient function equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displayed monotonicity comparisons, Real-seal boundary, bounded-window routing, variation-ledger handoff, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneRealFunctionUp.

Chapter 8159

A Concrete Naming Certificate for $\mathsf{RegularCauchySheaf}^{\uparrow}$

Following Godement's sheaf gluing viewpoint and Bishop–Bridges regular Cauchy names, $\mathsf{RegularCauchySheaf}^{\uparrow}$ records finite cover-indexed local regular-Cauchy windows whose compatible overlaps glue to a $\mathsf{Real}^{\uparrow}$ seal without quotient representatives or countable choice. The packet sits after the finite gluing surface of chapter 5899, the stream-name interface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the reindexing route of chapter 1568.

Definition 8159.1 (Regular Cauchy sheaf carrier)**.** *A* $\mathsf{RegularCauchySheaf}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, R, O, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite cover-window ledger. It lists the cover indices, the displayed refinements between windows, and the finite overlap pairs on which local regular Cauchy data are compared.*
2. *R is the restriction-map row. For each displayed refinement in U, it records the finite restriction route that transports a local* $\mathsf{StreamName}^{\uparrow}$ *observation and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback to the smaller window.*
3. *O is the overlap ledger. It records the pairwise compatibility witnesses consumed by* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *and the finite overlap-induction route used when several refinements are chained.*
4. *Q is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It is obtained by reading all local regular rational rows through U, R, O, H, C and not by selecting representatives of a quotient.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite cover windows, restrictions, overlap ledgers, and glued readback have been displayed.*
6. *H records componentwise* hsame *transport for the cover ledger, restriction maps, overlap ledgers, glued readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite descent route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySheaf}^{\uparrow}$, $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySheaf}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the local windows in U restrict through R, agree on every displayed overlap in O, glue to Q, and then seal as E. It has no coordinate for quotient sheaves, selected global representatives, countable choice, global completeness of $\mathsf{Real}^{\uparrow}$*, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySheafUp.RegularCauchySheafTasteGate single carrier alignment`

Theorem 8159.2 (Regular Cauchy sheaf NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySheaf}^{\uparrow}$ *carrier* $S = (U, R, O, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySheaf}^{\uparrow}}$ *surface consists exactly of finite cover-window admission through* U*, restriction-map exactness through* R*, pairwise overlap compatibility and finite overlap induction through* O*, glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The cover windows are finite because U displays the inspected indices, their refinements, and the overlap pairs consumed by the descent route.

For a single overlap, apply the $\mathsf{RegularCauchyGluing}^{\uparrow}$ surface named in P. It compares the local $\mathsf{StreamName}^{\uparrow}$ observations and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks through the displayed overlap data and preserves the glued readback row.

For a finite chain of overlaps, induct on the displayed refinement list in O. The empty chain reads the local row already present in U, and the step composes the next restriction row from R with the induction hypothesis before applying the gluing classifier to the next overlap.

The rows H and C transport and replay the same finite route, while P records the route through $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyGluing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Repacking these projections gives the listed local NameCert obligations, with no quotient sheaf, representative selector, countable choice, global completeness principle, Lean verification, or bridge claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySheaf}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySheaf}^{\uparrow}$ packet is generated from finite cover windows, displayed restriction maps, finite overlap ledgers, a glued $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySheaf}^{\uparrow}}$, seed)

Scope closed: Definition 8159.1 and Definition 8159.2 seed the finite sheaf-style descent carrier and its local NameCert obligation surface over BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySheaf}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient sheaves, selected global representatives, countable choice, global completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite cover-window admission, restriction-map exactness, overlap compatibility, finite overlap induction, glued RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySheafUp.

Chapter 8160

A Concrete Naming Certificate for $\mathsf{SphericallyCompleteUltrametric}^{\uparrow}$

Following Schikhof on non-Archimedean functional analysis, the $\mathsf{SphericallyCompleteUltrametric}^{\uparrow}$ packet records the finite NameCert surface for a nested-ball completeness condition in an ultrametric setting. It sits after the strong-triangle carrier of chapter 1535, the non-Archimedean metric route of chapter 7609, the complete-ultrametric service of chapter 3879, the nested closed-ball ledger of chapter 3526, and the constructive ultrametric completion boundary of chapter 5204. The packet names displayed ball-chain rows and finite intersection obligations; it does not choose centers, assert ambient completeness, or identify spherical completeness with ordinary metric completeness.

Definition 8160.1 (Spherically complete ultrametric carrier)**.** *A* $\mathsf{SphericallyCompleteUltrametric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, A, C, B, D, I, R, H, T, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UltrametricSpace}^{\uparrow}$ *row carrying the strong max-radius triangle comparison.*
2. *A is the* $\mathsf{NonArchimedeanMetric}^{\uparrow}$ *row exposing the non-Archimedean metric route used by every ball comparison.*
3. *C is the* $\mathsf{CompleteUltrametric}^{\uparrow}$ *service row available as a displayed dependency, not as an ambient completeness theorem.*
4. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row naming the nested family of closed ultrametric balls and their finite containment witnesses.*
5. *D is the* $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ *boundary row for completion-facing consumers.*
6. *I records the finite intersection-obligation row: every displayed nested family read must retain a nonempty-intersection request before any center read is public.*
7. *R is the radius and containment ledger, recording the inspected ball radii, reverse-inclusion order, and late-radius comparisons.*
8. *H records componentwise* hsame *transport for the ultrametric, non-Archimedean, complete-ultrametric, nested-ball, completion-boundary, intersection, radius-ledger, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a nested-ball request through U, A, C, B, D, I, R.*
10. *P is the* Pkg *provenance over* SphericallyCompleteUltrametric$^{\uparrow}$, UltrametricSpace$^{\uparrow}$, NonArchimedeanMetric$^{\uparrow}$, CompleteUltrametric$^{\uparrow}$, NestedClosedBall$^{\uparrow}$, ConstructiveUltrametricCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{SphericallyCompleteUltrametric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, A, C, B, D, I, R, H, T, P, N$. It has no coordinate for a choice-selected center, quotient completion, global compactness theorem, host-valued equality of balls, countable choice, or ambient completeness.

Theorem 8160.2 (Spherically complete ultrametric NameCert obligations)**.** *For every accepted* SphericallyCompleteUltrametric$^{\uparrow}$ *carrier*

$$S = (U, A, C, B, D, I, R, H, T, P, N),$$

the local NameCert$_{\text{SphericallyCompleteUltrametric}^{\uparrow}}$ *surface consists exactly of carrier admission for the ultrametric row, non-Archimedean metric row, complete-ultrametric service row, nested closed-ball row, constructive ultrametric completion boundary, intersection-obligation row, radius and containment ledger, componentwise transport, displayed continuation replay, package provenance, and local naming. A consumer may read the spherical-completeness handoff only after $U, A, C, B, D, I, R, H, T, P, N$ have all been displayed, and the handoff exports no choice-selected center, ambient completeness theorem, quotient completion, compactness principle, or host equality of ball families.*

Proof. Project the accepted carrier of Definition 8160.1. The rows U, A, C, B, D place the packet inside the existing ultrametric, non-Archimedean metric, complete-ultrametric, nested closed-ball, and constructive completion chapters.

The spherical row is I together with the radius ledger R. It records the finite obligation that a nested closed-ball family has an intersection request available before any center-facing consumer reads the handoff; it does not select a center or replace the displayed nested-ball route by host completeness.

Transport, replay, provenance, and local naming are exactly H, T, P, N. Since every read of the handoff must pass through those rows, the local NameCert surface is exhausted by $U, A, C, B, D, I, R, H, T, P, N$, and all excluded choice, quotient, compactness, and ambient-completeness claims remain outside the packet. □

Falsifiable prediction: Every SphericallyCompleteUltrametric$^{\uparrow}$ consumer must expose the ultrametric row, non-Archimedean metric row, complete-ultrametric service row, nested closed-ball row, constructive ultrametric completion boundary, intersection-obligation row, radius ledger, transport, replay, provenance, and local naming before it reads a spherical-completeness handoff.

Independence witness: ultrametricspace, nonarchimedean-metric, complete-ultrametric, nestedclosedball, constructive-ultrametric-completion: none alone is bijective to SphericallyCompleteUltrametric$^{\uparrow}$ because the packet jointly carries strong-triangle, non-Archimedean metric, complete-ultrametric service, nested-ball, completion-boundary, intersection-obligation, radius-ledger, transport, replay, provenance, and local NameCert rows while excluding selected centers and ambient completeness.

Closure status for NameCert$_{\text{SphericallyCompleteUltrametric}^{\uparrow}}$.

Constructive story: A SphericallyCompleteUltrametric$^{\uparrow}$ packet is generated from an ultrametric row, a non-Archimedean metric row, a complete-ultrametric service row, a nested closed-ball row, a constructive ultrametric completion boundary, an intersection-obligation row, a radius and containment ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SphericallyCompleteUltrametric}\uparrow}$, seed)

Scope closed: Definition 8160.1 and Definition 8160.2 seed the finite nested-ball completeness carrier and local NameCert obligation surface over ultrametric, non-Archimedean, complete-ultrametric, nested-ball, and constructive completion rows.

Formal verification: FormalStatus(NameCert$_{\text{SphericallyCompleteUltrametric}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected centers, ambient completeness, quotient completions, compactness principles, host equality of ball families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ultrametric route exposure, non-Archimedean metric exposure, complete-ultrametric service exposure, nested-ball exactness, completion-boundary routing, intersection-obligation coverage, radius-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.SphericallyCompleteUltrametricU

Chapter 8161

A Concrete Naming Certificate for $\mathsf{BishopSpread}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopSpread}^{\uparrow}$ packet records spread data as a finite-prefix carrier rather than as a completed choice sequence. It sits beside the binary prefix surfaces of $\mathsf{CantorSpace}^{\uparrow}$ in chapter 1475, the fan restriction of $\mathsf{FanSpace}^{\uparrow}$ in chapter 1962, the finite observation windows of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular rational readback rows of $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, and the bar-modulus rows of $\mathsf{BishopBarInductionModulus}^{\uparrow}$ in chapter 5132. The packet supplies countable-name discipline for later $\mathsf{Real}^{\uparrow}$ and completion consumers without assuming a selected infinite branch, countable choice, or an ambient space of completed sequences.

8161.1 Carrier and obligations

Definition 8161.1 (Bishop spread carrier)**.** *A* $\mathsf{BishopSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, A, W, R, B, H, C, G, N)$$

with the following displayed rows:

1. *P is the finite-prefix tree row, whose nodes are finite natural-number lists and whose admissibility predicate is decidable on displayed prefixes.*
2. *A is the apartness and compatibility row saying that two prefixes are compared only by the first displayed coordinate at which both rows are defined, or by prefix extension when no split is displayed.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected prefixes and requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only when the prefix data is consumed by a regular-Cauchy or real-name consumer.*
5. *B is the* $\mathsf{BishopBarInductionModulus}^{\uparrow}$ *bar-modulus handoff row for finite bars over the same prefix surface.*
6. *H records componentwise* hsame *transport for prefix, apartness, window, readback, bar, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the same finite-prefix route.*

8. *G is the* Pkg *provenance over* $\mathsf{BishopSpread}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopBarInductionModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSpread}^{\uparrow}}$ *row.*

The classifier compares two carriers only through $P, A, W, R, B, H, C, G, N$. It has no coordinate for a selected infinite branch, a completed choice-sequence object, host equality of sequences, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8161.2 (Bishop spread NameCert obligations)**.** *For every accepted* $\mathsf{BishopSpread}^{\uparrow}$ *carrier $S = (P, A, W, R, B, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopSpread}^{\uparrow}}$ *surface consists exactly of finite-prefix admission, apartness and prefix-compatibility admission, finite-window exposure, regular rational readback handoff, bar-modulus handoff, componentwise transport, displayed replay, provenance, and local naming. Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion consumer must read the prefix rows through W before it can inspect R or B.*

Proof. Project the carrier of Definition 8161.1. The rows P and A expose the finite-prefix substrate and its displayed apartness discipline before any branch-facing read is possible.

Next read the countable-name route through W and R. The $\mathsf{StreamName}^{\uparrow}$ row supplies only finite windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row is reached only as a displayed readback for regular-Cauchy or real-name consumers.

Finally use B, H, C, G, N to carry the bar handoff, transport, replay, provenance, and local naming rows. Since no coordinate stores a selected infinite branch, completed choice sequence, host sequence equality, countable choice, quotient stream equality, or ambient completeness principle, the listed rows exhaust the NameCert surface. □

Theorem 8161.3 (Bishop spread finite-prefix induction)**.** *Every induction or bar-style consumer of an accepted* $\mathsf{BishopSpread}^{\uparrow}$ *packet must factor through the finite prefix rows P, A, W and the displayed bar-modulus row B. The consumer may extend a prefix, inspect a finite bar, or pass the read to* $\mathsf{RegSeqRat}^{\uparrow}$*, but it cannot infer a completed branch, a chosen path through the spread, countable choice, host-level sequence equality, or a global continuity theorem from the packet alone.*

Proof. Begin with the obligation surface of Definition 8161.2. It forces every consumer to expose the prefix row, apartness row, and finite-window row before the bar or readback rows are inspected.

For an induction read, extend only the displayed prefix in P and transport the apartness row A componentwise. This keeps the step inside the same finite-prefix carrier.

For a bar-style read, use the $\mathsf{BishopBarInductionModulus}^{\uparrow}$ handoff row B. The handoff names finite bar coverage over the displayed prefixes, not a selected infinite path or a global theorem about all completed choice sequences.

For a completion-facing read, pass through W before R. The regular rational handoff therefore consumes finite windows and exports no branch selector, countable-choice schedule, quotient stream equality, host sequence equality, or ambient real completeness row. □

Falsifiable prediction: Every accepted $\mathsf{BishopSpread}^{\uparrow}$ read exposes finite-prefix admission, prefix apartness, StreamName windows, RegSeqRat readback when requested, BishopBarInductionModulus handoff, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any RealUp or completion consumer is served.

Independence witness: cantorspace, fan-space, streamname, regseqrat, bishop-bar-induction-modulus: none is bijective with $\mathsf{BishopSpread}^{\uparrow}$ because this packet alone binds spread-prefix admission, apartness, finite observation windows, regular-Cauchy readback, bar-modulus handoff, transport, replay, provenance, and local naming as one countable-name substrate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSpread}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSpread}^{\uparrow}$ packet is generated from finite-prefix tree rows, apartness and prefix-compatibility rows, StreamName finite windows, optional RegSeqRat readback, BishopBarInductionModulus bar handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSpread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8161.1, Definition 8161.2, and Definition 8161.3 seed the finite-prefix spread substrate for countable-name and regular-Cauchy consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSpread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed choice sequences, selected infinite branches, countable choice, quotient stream equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-apartness stability, finite-window coverage, RegSeqRat readback, bar-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopSpreadUp.

Chapter 8162

A Concrete Naming Certificate for HyperbolicPhaseHolonomyIndex$^{\uparrow}$

The HyperbolicPhaseHolonomyIndex$^{\uparrow}$ packet realizes the hyperbolic observer phase route of chapter 9463 as a finite index surface for closed phase-transport classes. It is distinct from the carrier-level HyperbolicPhaseHolonomy$^{\uparrow}$ packet of chapter 6385: the holonomy packet records an accepted closed reversible transport row, while the index packet records the finite comparison class, boundary anchor, disk-metric readback, observer groupoid route, and provenance rows that allow consumers to compare such closed transports without importing a host holonomy group. It sits beside ObserverPhaseGroupoid$^{\uparrow}$ in chapter 6146, HyperbolicMetric$^{\uparrow}$ in chapter 3474, PoincareDiskMetric$^{\uparrow}$ in chapter 6209, and HyperbolicBoundaryShadow$^{\uparrow}$ in chapter 3813.

8162.1 Carrier and vision route

Definition 8162.1 (Hyperbolic phase-holonomy index carrier)**.** *A* HyperbolicPhaseHolonomyIndex$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, B, M, O, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* PoincareDiskMetric$^{\uparrow}$ *disk-metric readback row for the finite disk observations used by the closed transport.*
2. *B is the* HyperbolicBoundaryShadow$^{\uparrow}$ *boundary-anchor row saying which visible boundary shadow is preserved or crossed by the transport loop.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *comparison row attached to the same displayed disk and boundary observations.*
4. *O is the* ObserverPhaseGroupoid$^{\uparrow}$ *reversible phase-route row, including source, target, inverse, and composition evidence for the observer-facing phase surface.*
5. *L is the accepted* HyperbolicPhaseHolonomy$^{\uparrow}$ *closed transport row whose finite loop data is being indexed.*
6. *Q is the holonomy-index row, comparing two accepted loops only by D, B, M, O, L and by the displayed classifier rows already present in those packets.*

7. *H records componentwise* hsame *transport for disk, boundary, metric, groupoid, holonomy, index, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$, $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$, ObserverPhaseGroupo $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}}$ *row.*

The classifier compares accepted index carriers only through $D, B, M, O, L, Q, H, C, P, N$. It has no coordinate for a host holonomy group, quotient loop space, global curvature invariant, selected representative loop, physical observer law, or completed hyperbolic boundary.

Theorem 8162.2 (Hyperbolic phase-holonomy index obligations)**.** *For every accepted* HyperbolicPhaseHolonomyIn *carrier $I = (D, B, M, O, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}}$ *surface consists exactly of disk-metric readback, boundary-shadow anchoring, hyperbolic metric comparison, observer phase-groupoid routing, closed holonomy-row exposure, finite index comparison, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8162.1. The disk, boundary, metric, groupoid, and holonomy rows D, B, M, O, L must be displayed before the index row can be consumed.

The row Q compares accepted closed transports only through those displayed rows. It therefore records a finite index surface rather than a host quotient of loops or a global holonomy group.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no host holonomy group, quotient loop space, global curvature invariant, selected representative loop, physical observer law, or completed boundary coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 8162.3 (Hyperbolic phase-holonomy index vision route)**.** *The* $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$ *packet realizes the observer-expansion holonomy-index route only as finite disk, boundary, metric, observer-groupoid, closed-holonomy, index, transport, replay, provenance, and NameCert rows. Every consumer of the index must first read* $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, *and* $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ *rows; the consumer receives no generic phase-transport packet, host holonomy group, quotient loop space, global curvature invariant, physical observer law, selected representative loop, or completed boundary object.*

Proof. Use the obligation surface of Definition 8162.2. It orders every index read through the disk, boundary, metric, observer-groupoid, and closed-holonomy rows before Q is exposed.

The vision route is therefore a concrete finite ledger for comparing closed phase transports. It refines $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ by adding the comparison index and its boundary and metric anchors, but it does not duplicate the underlying holonomy carrier.

Finally project the refusal coordinates from Definition 8162.1. Since the packet has no host quotient, global holonomy group, curvature invariant, physical observer law, selected loop, or completed-boundary coordinate, no consumer can extract those objects from the index route. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$ read exposes PoincareDiskMetric, HyperbolicBoundaryShadow, HyperbolicMetric, ObserverPhaseGroupoid, HyperbolicPhaseHolonomy, index comparison, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any holonomy-index consumer is served.

Independence witness: hyperbolic-phase-holonomy, observer-phase-groupoid, hyperbolic-metric, poincare-disk-metric, hyperbolic-boundary-shadow: $\mathsf{HyperbolicPhaseHolonomy}^{\uparrow}$ carries a closed transport without the index row, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ carries reversible phase routing without the closed holonomy comparison, $\mathsf{HyperbolicMetric}^{\uparrow}$ and $\mathsf{PoincareDiskMetric}^{\uparrow}$ carry metric readback without the phase loop, and $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ carries the boundary anchor without the index comparison. None is bijective to $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$ packet is generated from PoincareDiskMetric disk rows, HyperbolicBoundaryShadow boundary anchors, HyperbolicMetric comparison rows, ObserverPhaseGroupoid reversible phase-route rows, HyperbolicPhaseHolonomy closed transport rows, a finite index comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8162.1, Definition 8162.2, and Definition 8162.3 seed the finite holonomy-index route for hyperbolic observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host holonomy groups, quotient loop spaces, global curvature invariants, physical observer laws, selected representative loops, completed hyperbolic boundaries, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk and boundary row exposure, metric comparison, observer-groupoid routing, closed-holonomy handoff, index comparison, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseHolonomyIndexUp.

Chapter 8163

A Concrete Naming Certificate for $\mathsf{FiniteModulusBundle}^{\uparrow}$

Following Bishop and Bridges on constructive uniform continuity and explicit moduli, the $\mathsf{FiniteModulusBundle}^{\uparrow}$ packet records the finite L10 data that lets a Real and Completion consumer read one displayed uniform modulus together with the finite windows, regular rational readback, and terminal seal it controls. It sits beside the uniform-modulus certificate of chapter 369, the uniformly continuous sequence route of chapter 4287, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names shared finite modulus data; it is not a host function, a quotient stream equality, a choice of infinitely many witnesses, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8163.1 (FiniteModulusBundle carrier)**.** *A* $\mathsf{FiniteModulusBundle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, Q, W, R, E, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row carrying the requested uniform modulus schedule.*
2. *Q is the query ledger that lists the finite precision requests and source-window requests served by U.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the observations named by Q.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback assembled from the displayed modulus requests and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the finite readback row has been displayed.*
6. *S is the* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *handoff row that records which sequence-facing consumer may reuse the same modulus bundle.*
7. *H records componentwise* hsame *transport for the modulus, query, window, readback, seal, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an L10 consumer request through U, Q, W, R, E, S, H.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusBundle}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformlyContinuousSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusBundle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, Q, W, R, E, S, H, C, P, N$. It has no coordinate for a host function, quotient equality of streams, a selected infinite witness family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.FiniteModulusBundleUp`

Theorem 8163.2 (FiniteModulusBundle NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusBundle}^{\uparrow}$ *carrier*

$$B = (U, Q, W, R, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusBundle}^{\uparrow}}$ *surface consists exactly of uniform-modulus schedule admission through U, finite request listing through Q, stream-window exposure through W, regular rational readback through R, terminal real-name sealing through E, uniformly continuous sequence handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8163.1. The row U is the only coordinate that stores the uniform modulus schedule, and Q is the only coordinate that lists the finite consumer requests governed by that schedule.

The stream and numeric rows are downstream of those requests. The row W exposes exactly the named finite observations, R records the regular rational readback obtained from those observations, and E seals that readback as the terminal real-facing row.

The sequence handoff and structural rows do not add hidden analytic data. The row S records the sequence-facing reuse boundary, while H, C, P, N transport, replay, source, and name the same displayed packet. Since the carrier lists no host function, quotient stream equality, infinite selected witness family, countable-choice schedule, or ambient completeness coordinate, the displayed rows exhaust the local NameCert surface. □

Theorem 8163.3 (FiniteModulusBundle uniform-sequence handoff)**.** *Every uniformly continuous sequence consumer of an accepted* $\mathsf{FiniteModulusBundle}^{\uparrow}$ *packet factors through*

$$\mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformlyContinuousSequence}^{\uparrow} \longmapsto \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{F}$$

The consumer may reuse the displayed modulus schedule only after the finite request ledger, stream-window rows, regular rational readback, and terminal Real seal have been exposed. It obtains no host function equality, quotient stream equality, selected infinite witness family, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with a consumer read of $B = (U, Q, W, R, E, S, H, C, P, N)$. The read first exposes the uniform modulus row U and the finite request ledger Q, so the consumer sees a displayed schedule and a finite list of requests rather than an external function.

Next the consumer reads W, R, E. The stream row supplies the requested windows, the regular rational row records their finite readback, and the Real row seals only that displayed readback.

Finally the handoff row S records the uniformly continuous sequence boundary that may reuse the schedule. The structural rows H, C, P, N transport, replay, source, and name the same route, so the consumer cannot project any of the excluded host, quotient, choice, or completeness data from the packet. □

Falsifiable prediction: Every accepted $\mathsf{FiniteModulusBundle}^{\uparrow}$ read exposes the UniformModulus schedule row, finite request ledger, StreamName finite-window row, RegSeqRat readback row,

RealUp seal row, UniformlyContinuousSequence handoff row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a sequence-facing L10 consumer may reuse a modulus.

Independence witness: uniformmodulus, uniformly-continuous-sequence, streamname, regseqrat, real: $\mathsf{FiniteModulusBundle}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a shared uniform-modulus schedule, finite request ledger, stream windows, regular rational readback, terminal Real sealing, sequence handoff, transport, replay, provenance, and local naming as one finite L10 packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusBundle}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusBundle}^{\uparrow}$ packet is generated from a UniformModulus schedule row, a finite request ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, a UniformlyContinuousSequence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusBundle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8163.1, Definition 8163.2, and Definition 8163.3 seed the shared finite-modulus packet for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusBundle}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FiniteModulusBundleUp`

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, host function equality, quotient stream equality, selected infinite witness families, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite request exactness, stream-window exactness, regular readback, Real sealing, sequence handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusBundle

Chapter 8164

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$ packet records the finite common-window lattice by which left and right regular Cauchy readings can share one dyadic refinement route before any $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits after the dyadic arithmetic of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the common-modulus meet route of chapter 462.

Definition 8164.1 (Regular Cauchy window lattice carrier)**.** *A* $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right regular-Cauchy window rows. Each row exposes the requested precision, the finite source indices, and the local regularity budget visible to a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
2. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger. It records the shared dyadic radii, endpoint comparisons, radius meets, and refinement witnesses used by both windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row listing exactly the source observations inspected on the left window, right window, and overlap.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after L, R, D, S have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the displayed common-window route.*
6. *H records componentwise* hsame *transport for the two windows, common dyadic ledger, finite observations, regular readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite common-window route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowLattice}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the two windows share the displayed refinement ledger before either downstream readback is consumed. It has no coordinate for quotient equality of regular sequences, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8164.2 (Regular Cauchy window lattice NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$ *carrier*

$$W = (L, R, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowLattice}^{\uparrow}}$ *surface consists exactly of left and right regular-Cauchy window admission, common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The analytic rows before the seal are the two window rows L, R, the common dyadic ledger D, the finite observation row S, and the regular readback row Q.

The common-window condition is finite. The row D stores the radius meets and endpoint comparisons, while S records exactly the observations used to inspect both windows. Thus Q reads a single shared refinement route rather than two unrelated regular-sequence witnesses.

The remaining rows H, C, P, N transport, replay, source, and name this same route. Since quotient equality, selected representatives, countable choice, host equality, and ambient completeness are absent from the carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 8164.3 (Regular Cauchy window lattice RegSeqRat handoff)**.** *Every accepted* RegularCauchyWindowLatt *carrier exposes its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *only after the left and right regular-Cauchy windows* L, R*, the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, and the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row* S *have been displayed. The handoff exports the common-window packet to* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers without exporting quotient equality of regular sequences or a selected representative.*

Proof. Read the carrier rows in order. The two source windows L, R provide the local regular-Cauchy requests, but neither is a complete readback route for a downstream consumer on its own.

The ledger D and observation row S form the common refinement surface. They display the dyadic meets, overlap comparisons, and inspected indices that make the two window reads compatible inside one finite packet.

Only then is the readback row Q available. The structural rows H, C, P, N preserve and name this handoff, and the carrier has no coordinate from which a quotient equality or selected representative could be read. □

Theorem 8164.4 (Regular Cauchy window lattice Real seal nonescape)**.** *Let* $W = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$ *carrier. Any downstream* $\mathsf{Real}^{\uparrow}$ *read of the seal row* E *must first expose the common dyadic ledger* D*, the finite observation row* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*. Consequently the Real-facing route cannot bypass the finite common-window lattice by appealing to host real equality, quotient sequence equality, countable choice, selected representatives, or ambient completeness.*

Proof. Project the displayed carrier. The seal row E is listed after the two window rows, the common dyadic ledger, the finite observation row, and the regular readback row.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 8164.3 keeps D, S, Q visible before a Real-facing consumer can inspect E. The same route is transported by H and replayed by C with provenance P and local naming N.

No excluded principle is stored in W. Thus a $\mathsf{Real}^{\uparrow}$ seal read is a read of the finite common-window lattice, not an escape to host equality, quotient equality, choice, representatives, or completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyWindowLattice}^{\uparrow}$ packet is generated from left and right regular-Cauchy windows, a common $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8164.1, Definition 8164.2, Definition 8164.3, and Definition 8164.4 seed the finite common-window lattice route for regular Cauchy readbacks and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of regular sequences, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common dyadic-refinement exactness, finite observation exposure, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindowLatticeUp.

Chapter 8165

A Concrete Naming Certificate for $\mathsf{CauchyTailInterleaving}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy-sequence practice, the $\mathsf{CauchyTailInterleaving}^{\uparrow}$ packet records the finite route by which two regular Cauchy tails are fused into one displayed tail schedule without choosing representatives of completed real names. It sits after the regular rational source of chapter 301, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the monotone selector corridor of chapter 5902. The packet is a choice-free fusion carrier for Real and Completion consumers; it is not a quotient of tails, a selected limit, a countable-choice functional, host real equality, or ambient completeness.

Definition 8165.1 (Cauchy tail interleaving carrier)**.** *A* $\mathsf{CauchyTailInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, S_0, S_1, M, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose Cauchy tails are being fused.*
2. S_0 *and* S_1 *are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows used to inspect the two tails.*
3. M *is the* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *monotone corridor that records the displayed tail schedules used by the fusion.*
4. W *is the interleaving window row, recording for each inspected output window whether it reads the left tail, the right tail, or the shared comparison window.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two tail reads at the displayed windows.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after* $R_0, R_1, S_0, S_1, M, W, D$ *have been read.*
7. H *records componentwise* hsame *transport for the two sources, two window rows, selector corridor, interleaving row, tolerance ledger, seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a fusion request through* $R_0, R_1, S_0, S_1, M, W, D, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{CauchyTailInterleaving}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySubsequenceSelector}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailInterleaving}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for quotient stream equality, selected representatives, a host limit, countable choice, real trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8165.2 (Cauchy tail interleaving NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailInterleaving}^{\uparrow}$ *carrier* $T = (R_0, R_1, S_0, S_1, M, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailInterleaving}^{\uparrow}}$ *surface consists exactly of carrier admission, left-tail and right-tail source admission, finite-window admission, selector-corridor monotonicity, interleaving-window exactness, dyadic tolerance compatibility,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise transport, displayed replay, package provenance, and local naming. Every obligation is read through the finite route*

$$R_0, R_1, S_0, S_1, M, W, D \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Carrier admission is projection from Definition 8165.1. The two regular rational sources, the two finite window rows, the selector corridor, the interleaving window, the dyadic tolerance ledger, and the terminal seal are all visible coordinates of the same packet.

The selector-corridor obligation is supplied by M. It keeps the two tail schedules monotone and finite, while W records which source tail each output window reads before any fused consumer receives a seal.

The tolerance obligation is read from D over the displayed windows. It compares only the inspected regular rational rows R_0, R_1 through S_0, S_1, M, W, so the certificate has no need for a quotient-tail equality or a selected representative.

Transport through H, replay through C, provenance P, and local naming N preserve the same route. Since the excluded quotient, choice, limit, trichotomy, host equality, and completeness objects are absent from the carrier coordinates, no NameCert obligation can discharge through them. □

Theorem 8165.3 (Cauchy tail interleaving Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{CauchyTailInterleaving}^{\uparrow}$ *packet must expose the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* R_0, R_1*, the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* S_0, S_1*, the* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *corridor* M*, the interleaving window row* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before it reads the terminal seal* E*. The fused tail reaches Real and Completion consumers only through this displayed window-and-tolerance route, not through quotient tails, countable choice, selected limits, host real equality, or ambient completeness.*

Proof. Project the accepted carrier $T = (R_0, R_1, S_0, S_1, M, W, D, E, H, C, P, N)$. The terminal seal E is downstream of the two source rows, the two finite-window rows, the selector corridor, the interleaving row, and the dyadic tolerance ledger.

Apply Definition 8165.2. It forces carrier admission, source admission, finite-window admission, selector monotonicity, interleaving-window exactness, and dyadic tolerance compatibility before any consumer can read E.

The structural rows H, C, P, N transport, replay, source, and name the same data. Therefore the handoff cannot bypass $R_0, R_1, S_0, S_1, M, W, D$, and the absent quotient, choice, selected-limit, host-equality, and completeness principles remain outside the public route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailInterleaving}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailInterleaving}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two finite StreamName window rows, a CauchySubsequenceSelector corridor, an interleaving window row, a DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyTailInterleaving}\uparrow}$, seed)

Scope closed: Definition 8165.1, Definition 8165.2, and Definition 8165.3 seed the finite choice-free fusion surface for two regular Cauchy tails before Real and Completion consumers read a terminal seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyTailInterleaving}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient tail equality, selected representatives, countable choice, selected limits, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, left-tail and right-tail source admission, finite-window admission, selector-corridor monotonicity, interleaving-window exactness, dyadic tolerance compatibility, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailInterleavingUp.

Chapter 8166

A Concrete Naming Certificate for BishopRegularCauchyMetric$^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the BishopRegularCauchyMetric$^{\uparrow}$ packet records the metric interface on regular Cauchy rational names as finite BEDC NameCert data. It sits after the dyadic arithmetic of chapter 291, the finite observation schedules of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the common-window lattice of chapter 8164. The packet names the regular-Cauchy distance read only through displayed dyadic windows and readback ledgers; it is not a quotient of streams, a selected host real representative, a trichotomy principle, or an ambient completeness theorem.

Definition 8166.1 (Bishop regular-Cauchy metric carrier)**.** *A* BishopRegularCauchyMetric$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, D, S, Q, E, B, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right regular-Cauchy rational-name rows whose finite windows are being compared.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *distance ledger, recording the dyadic tolerance, absolute-difference row, radius comparison, and common refinement budget.*
3. *S is the* StreamName$^{\uparrow}$ *observation row listing exactly the source indices inspected on the two names.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row obtained after the dyadic ledger and finite observations have been displayed.*
5. *E is the terminal* Real$^{\uparrow}$ *seal for the distance value read from the same regular-Cauchy route.*
6. *B is the Bishop metric-bound row, storing positivity, symmetry, triangle-window, and zero-distance readback clauses only as finite regular-Cauchy obligations.*
7. *H records componentwise* hsame *transport for the two source names, dyadic ledger, stream observations, regular readback, Real seal, metric-bound row, replay, provenance, and local certificate.*
8. *C records displayed* Cont *replay of the finite route from regular-Cauchy names to the distance seal.*
9. *P is the* Pkg *provenance over* BishopRegularCauchyMetric$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RegularCauchyWindowLattice$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyMetric}^{\uparrow}}$ *row.*
The carrier predicate asserts that every metric read factors through the displayed dyadic common-window route. It has no coordinate for quotient equality of sequences, a selected completion representative, excluded middle over order comparisons, countable choice, or host metric completion.

Theorem 8166.2 (Bishop regular-Cauchy metric NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyMetric}^{\uparrow}$ *carrier*

$$M = (L, R, D, S, Q, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyMetric}^{\uparrow}}$ *surface consists exactly of regular-name exposure for L, R, dyadic distance-ledger exposure through D, finite stream-window exposure through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q,* $\mathsf{Real}^{\uparrow}$ *sealing through E, Bishop metric-bound ledger exposure through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape from quotient, choice, order-trichotomy, or host completion principles.*

Proof. Project the accepted carrier. The only analytic input rows are L, R, D, S, Q, E, B together with the structural rows H, C, P, N.

The distance ledger is finite. The dyadic row D records the requested tolerance and absolute-difference comparison, while S records the inspected indices. Thus the readback row Q and seal row E are reached through displayed regular-Cauchy data rather than through a semantic quotient of streams.

The Bishop metric-bound row B names positivity, symmetry, triangle-window, and zero-distance clauses as ledgers over the same route. Componentwise transport and replay preserve those displayed rows, and the excluded principles have no carrier coordinate. Hence the listed rows exhaust the NameCert obligations. □

Theorem 8166.3 (Bishop regular-Cauchy metric Real-completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopRegularCauchyMetric}^{\uparrow}$ *packet must first read*

$$L, R, D, S, Q, E, B, H, C, P, N$$

in the displayed order before treating the sealed row E as a regular-Cauchy distance in $\mathsf{Real}^{\uparrow}$*. The handoff therefore passes through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the common-window lattice of chapter 8164; it exports no host equality of Cauchy names, selected representative, countable-choice schedule, trichotomy oracle, or ambient completion operator.*

Proof. Start with a completion-facing read. The carrier exposes two regular-Cauchy source rows L, R before any distance row can be consumed.

The dyadic and stream stages are D and S. They record the common tolerance budget, absolute-difference comparison, and inspected windows that license the regular readback Q.

Only after this readback does the terminal Real seal E become visible, and the Bishop metric-bound row B is a ledger over the same displayed route. The structural rows H, C, P, N transport, replay, source, and name that route without adding coordinates.

Since quotient equality of names, selected representatives, countable choice, trichotomy, and host completion are absent from the packet, every completion-facing distance read remains inside the finite regular-Cauchy metric handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyMetric}^{\uparrow}$ packet is generated from two regular-Cauchy rational-name rows, a DyadicRatCore distance ledger, finite StreamName observations, RegSeqRat readback, a RealUp seal, a Bishop metric-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8166.1, Definition 8166.2, and Definition 8166.3 seed the finite metric route from regular Cauchy rational names to a Real-facing distance seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy names, selected representatives, countable choice, trichotomy of reals, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic distance-ledger exactness, finite stream-window exposure, RegSeqRat readback, Real seal ordering, Bishop metric-bound rows, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyMetricUp.

Chapter 8167

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusSelector}^{\uparrow}$

Following Bishop and Bridges compact metric analysis, the $\mathsf{CompactUniformModulusSelector}^{\uparrow}$ packet records the finite selector that turns compact finite-net data and pointwise continuity rows into a tolerance-only uniform-modulus ledger. It realizes the compact-uniform-continuity route of chapter 9237 through the compact source of chapter 60, the metric rows of chapter 57, the continuity surface of chapter 59, the rational finite-minimum readback of chapter 54, the $\mathsf{Rat}^{\uparrow}$ arithmetic of chapter 37, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ packet of chapter 369. The packet names only a displayed finite ProbeBundle fold; it is not an open-cover compactness theorem, a host modulus function, a selected Lebesgue number, a sequential compactness argument, or an ambient completeness principle.

Definition 8167.1 (Compact uniform-modulus selector carrier)**.** *A* $\mathsf{CompactUniformModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, K, F, T, B, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the shared* $\mathsf{Met}^{\uparrow}$ *source row.*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, exposing the bundled centers and coverage radii inspected by the selector.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row over the same source.*
4. *T is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance request.*
5. *B is the finite* ProbeBundle *ledger whose entries contain centers, local pointwise-modulus rows, subordinate radius rows, and half-radius cover data.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, using the finite-order selector from* $\mathsf{TotalOrder}^{\uparrow}$ *to produce a displayed radius below every bundled radius.*
7. *L is the metric-triangle ledger joining compact coverage, pointwise moduli, and the selected radius.*
8. *U is the terminal* $\mathsf{UniformModulus}^{\uparrow}$ *row exposed only after B, R, L have been displayed.*

9. *H records componentwise* hsame *transport for the metric, compact, continuous, tolerance, bundle, rational fold, triangle ledger, uniform-modulus output, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay for the finite selector route.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusSelector}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusSelector}^{\uparrow}}$ *row.*

The carrier predicate compares accepted selectors only through these rows. It has no coordinate for an arbitrary open cover, selected point outside the bundle, host minimum, sequence extraction, quotient metric equality, countable choice, or ambient compactness.

Theorem 8167.2 (Compact uniform-modulus selector NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusSelector}^{\uparrow}$ *carrier* $S = (X, K, F, T, B, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusSelector}^{\uparrow}}$ *obligations are exactly carrier admission, shared metric-source lock, compact finite-net exposure, pointwise-modulus exposure, positive rational tolerance exposure, finite* ProbeBundle *admission, rational lower-bound fold exactness, metric-triangle ledger replay,* $\mathsf{UniformModulus}^{\uparrow}$ *output admission, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from open-cover, choice, quotient, or ambient-completeness principles.*

Proof. Carrier admission projects the displayed rows of Definition 8167.1. The source side is X, K, F, T, so the compact and continuous rows are locked to one metric carrier before the finite bundle is opened.

The selector stage is B, R, L. The bundle row lists the inspected centers and local radius data, the rational row folds those finitely many radii through $\mathsf{Rat}^{\uparrow}$ and $\mathsf{TotalOrder}^{\uparrow}$, and the ledger row records the metric-triangle replay needed by the uniform-modulus consumer.

The output and structural stage is U, H, C, P, N. These rows expose, transport, replay, source, and name the same finite route. Since the excluded open-cover, host-minimum, choice, quotient, and completeness objects are absent from the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8167.3 (Compact uniform-modulus selector finite-fold handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{CompactUniformModulusSelector}^{\uparrow}$ *packet factors through the finite route*

$$X, K, F, T \longmapsto B, R, L \longmapsto U, H, C, P, N.$$

Thus the selector first reads the common metric source, compact finite-net data, pointwise-modulus rows, and rational tolerance; then folds the displayed ProbeBundle *radius ledger; and only then exposes the tolerance-only uniform-modulus output.*

Proof. Begin with the source rows X, K, F, T. They fix the metric carrier, compact finite-net coverage, pointwise-modulus entries, and tolerance request available to the selector.

The finite fold is carried by B and R. The row B displays every bundled radius and center, while R records the rational lower-bound selected by the finite order readback. No entry outside the bundle is available to the fold.

The metric-triangle ledger L is then replayed before U is public. The structural rows H, C, P, N preserve that same route, so a $\mathsf{UniformModulus}^{\uparrow}$ consumer receives only the finite selector package and no host compactness, selected Lebesgue number, arbitrary open cover, quotient equality, countable choice, or ambient completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusSelector}^{\uparrow}$ packet is generated from a MetricUp source, CompactUp finite-net rows, ContinuousUp pointwise-modulus rows, a RatUp tolerance, a finite ProbeBundle ledger, a RatUp lower-bound fold, a metric-triangle ledger, a UniformModulusUp output, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8167.1, Definition 8167.2, and Definition 8167.3 seed the finite selector from compact finite-net and pointwise-modulus rows to a UniformModulus output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, sequence extraction, host modulus functions, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source lock, compact finite-net exposure, pointwise-modulus exposure, rational fold exactness, metric-triangle replay, UniformModulus output admission, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusSelectorUp.

Chapter 8168

A Concrete Naming Certificate for $\mathsf{AlexanderDuality}^{\uparrow}$

Following Hatcher and Spanier, the $\mathsf{AlexanderDuality}^{\uparrow}$ packet records the finite simplicial complement route behind Alexander duality as BEDC NameCert data. It sits after the simplicial-complex carrier of chapter 238, the homology and cohomology rows of chapter 98 and chapter 99, the compact metric support of chapter 129, and the circle boundary carrier of chapter 58. The chapter names the finite complement-window and boundary rows used by the duality read; it does not import a classical point-set complement, open-cover compactness, selected deformation retracts, quotient homology equality, choice, or ambient local compactness.

Definition 8168.1 (Alexander duality carrier)**.** *An* $\mathsf{AlexanderDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, L, H, C, M, S, W, B, D, T, R, P, N)$$

with the following displayed rows:

1. *K is the finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *row naming the inspected subcomplex and its face-incidence ledger.*
2. *L is the finite ambient sphere or compact support row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *and the boundary discipline of* $\mathsf{S}^{1\uparrow}$ *when the inspected window is one-dimensional.*
3. *H is the* $\mathsf{Homology}^{\uparrow}$ *row for the displayed complement window.*
4. *C is the* $\mathsf{Cohomology}^{\uparrow}$ *row for the displayed subcomplex window.*
5. *M is the compact-metric witness row that keeps complement membership and boundary reads finite.*
6. *S is the sphere-boundary row, recording the visible boundary orientation and the inspected sphere coordinate windows.*
7. *W is the complement-window ledger, listing the finite faces and boundary cells whose complement is available to the duality read.*
8. *B is the boundary-link row, pairing each inspected face of K with the corresponding complement-window boundary cell.*
9. *D is the duality-degree ledger, recording the inspected degree shift and the finite source and target windows.*

10. *T records componentwise* hsame *transport for the simplicial, homology, cohomology, compact-support, sphere-boundary, complement-window, boundary-link, and degree rows.*
11. *R records displayed* Cont *replay for consumers that ask for the same finite duality route.*
12. *P is the* Pkg *provenance over* $\mathsf{AlexanderDuality}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexanderDuality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient point-set complement, an arbitrary sphere model, a selected deformation retract, quotient homology equality, open-cover compactness, countable choice, or external excision.

Theorem 8168.2 (Alexander duality NameCert obligations)**.** *For every accepted* $\mathsf{AlexanderDuality}^{\uparrow}$ *carrier $A = (K, L, H, C, M, S, W, B, D, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexanderDuality}^{\uparrow}}$ *surface consists exactly of carrier admission for the simplicial-complex row K, compact sphere-support row L, complement homology row H, subcomplex cohomology row C, compact-metric row M, sphere-boundary row S, complement-window ledger W, boundary-link row B, degree ledger D, componentwise transport T, displayed replay R, provenance P, and local naming N. A consumer may read the displayed duality window from these rows, but it receives no unrestricted Alexander duality theorem outside the finite carrier.*

Proof. Project the carrier of Definition 8168.1. The rows $K, L, H, C, M, S, W, B, D, T, R, P, N$ are the entire accepted inventory, so every local certificate obligation must be read from those rows.

The topological input is finite. The simplicial row K, compact support row L, compact-metric witness M, sphere-boundary row S, and complement-window ledger W expose only the inspected faces, boundary cells, and complement windows.

The algebraic input is finite as well. The homology row H, cohomology row C, boundary-link row B, and degree ledger D identify the inspected duality source and target windows without introducing quotient representatives or a global exactness principle.

Finally the structural rows T, R, P, N transport, replay, source, and name exactly that packet. Hence the local certificate cannot export an ambient point-set complement, selected deformation retract, quotient homology equality, open-cover compactness, choice, or external excision. □

Theorem 8168.3 (Alexander duality simplicial complement route)**.** *Every homology or cohomology consumer of an accepted* $\mathsf{AlexanderDuality}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{SimplicialComplex}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{S}^{1\uparrow} \longmapsto W, S, B, D \longmapsto \mathsf{Homology}^{\uparrow},\ \mathsf{Cohomology}^{\uparrow}.$$

The route exposes a displayed complement window, sphere-boundary row, boundary-link ledger, and degree-shift row before the homology and cohomology windows are consumed. It is a finite simplicial complement route, not a classical point-set complement, a host duality isomorphism, a selected deformation retract, quotient homology equality, or a choice principle.

Proof. Begin with the NameCert obligations of Definition 8168.2. They force every consumer to read K, L, M, S, W, B, D before the algebraic windows H and C are available.

The complement route is then determined by W, S, B, D. The window ledger W lists the complement cells, the sphere-boundary row S fixes the inspected boundary support, the boundary-link row B pairs subcomplex faces with complement cells, and the degree ledger D records the finite shift read by the algebraic rows.

It remains to pass the route to homology and cohomology consumers. The transport and replay rows T, R preserve the displayed route, while P, N source and name it. Since no other coordinate is present, the consumer receives only the finite simplicial complement route and not an ambient complement theorem, selected deformation retract, quotient equality, or choice datum. □

Falsifiable prediction: If the complement-window ledger W is read without the displayed $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, sphere-boundary, boundary-link, and degree rows, an accepted $\mathsf{AlexanderDuality}^{\uparrow}$ consumer must refuse the homology/cohomology handoff rather than treating the complement as a host point-set object.

Independence witness: simplicialcomplex, homology, cohomology, compactmetric, s1: $\mathsf{AlexanderDuality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries complement-window rows, sphere-boundary rows, boundary-link ledgers, degree-shift rows, homology/cohomology windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlexanderDuality}^{\uparrow}}$.

Constructive story: An $\mathsf{AlexanderDuality}^{\uparrow}$ packet is generated from a finite SimplicialComplexUp source row, compact sphere-support rows, complement HomologyUp rows, subcomplex CohomologyUp rows, CompactMetricUp finite support, SOneUp boundary windows when present, complement-window ledgers, boundary-link rows, a degree ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexanderDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8168.1, Definition 8168.2, and Definition 8168.3 seed the finite simplicial complement route for Alexander duality packets over simplicial, homology, cohomology, compact-metric, and sphere-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexanderDuality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted Alexander duality, classical point-set complements, selected deformation retracts, quotient homology equality, open-cover compactness, countable choice, or external excision.

Upgrade path: Obligation closure requires checked carrier admission, complement-window exactness, sphere-boundary exactness, boundary-link stability, degree-ledger readback, homology/cohomology handoff, transport, replay, provenance, and local naming for BEDC.Derived.AlexanderDualityUp.

Chapter 8169

A Concrete Naming Certificate for $\mathsf{BishopRealLocatedness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealLocatedness}^{\uparrow}$ packet records located real evidence as finite regular-Cauchy witness data rather than as real trichotomy, quotient-real equality, or ambient completeness. It sits after the dyadic core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric comparison surface of chapter 57. The packet names the finite evidence by which a regular real name can answer located interval requests.

Definition 8169.1 (Bishop real locatedness carrier)**.** *A* $\mathsf{BishopRealLocatedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, E, M, Q, H, C, P, N)$$

over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *with the following displayed rows:*

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming exactly the real-name observations inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and interval-comparison ledger used to compare the readback with displayed rational bounds.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the stream, regular-readback, and dyadic rows have been exposed.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row that records the located-distance request against rational interval endpoints.*
6. *Q is the locatedness query ledger. Each row names a pair of rational endpoints $a < b$ and records which side of the displayed interval is witnessed by the finite readback and dyadic comparison rows.*
7. *H records componentwise* hsame *transport for the stream, readback, dyadic, real-seal, metric, query, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a locatedness request through Q, S, R, D, E, M, H.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealLocatedness}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealLocatedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host real trichotomy, quotient-real equality, a selected Dedekind cut, a completed Cauchy quotient, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8169.2 (Bishop real locatedness NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealLocatedness}^{\uparrow}$ *carrier* $L = (S, R, D, E, M, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealLocatedness}^{\uparrow}}$ *surface consists exactly of stream-window admission, regular-rational readback, dyadic tolerance comparison, real-seal nonescape, metric interval comparison, locatedness query admission, componentwise transport, displayed replay, provenance, and local naming. Every locatedness answer must cite the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the same packet.*

Proof. Project the carrier of Definition 8169.1. The stream row S, regular readback R, dyadic ledger D, real seal E, metric row M, and query ledger Q are all visible coordinates of one finite packet.

Read a locatedness request through Q. The requested rational interval is compared only after the packet exposes the finite stream windows, the regular rational readback, and the dyadic tolerance ledger, so the answer is a finite witness row rather than a host real-order decision.

The structural rows H, C, P, N transport, replay, source, and name the same displayed coordinates. Since the carrier has no coordinate for trichotomy, quotient-real equality, a selected cut, a completed Cauchy quotient, countable choice, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8169.3 (Bishop real locatedness RegSeqRat witness route)**.** *Every locatedness witness exported by an accepted* $\mathsf{BishopRealLocatedness}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}.$$

Thus a consumer may read a located interval answer only after it has displayed the finite stream windows, the regular rational readback, the dyadic tolerance comparison, the real-name seal, and the metric comparison row. The route does not export real trichotomy, quotient-real equality, selected cuts, selected limits, countable choice, or ambient completeness.

Proof. Begin with a consumer of a locatedness answer. Carrier admission exposes the query row Q together with S, R, D, E, M, so the answer is attached to the same finite witness tuple.

Replay the request through C. The consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ windows, then the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance comparison. Only after those rows are visible may the terminal $\mathsf{Real}^{\uparrow}$ seal and the $\mathsf{Met}^{\uparrow}$ comparison row be inspected.

Transport by H preserves this order componentwise. The package and local NameCert rows cite only the displayed route, so no excluded real-order, quotient, choice, selected-limit, or completeness datum can be projected from the packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealLocatedness}^{\uparrow}$ read exposes StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance comparison, RealUp seal, MetricUp interval comparison, locatedness query admission, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before a located interval answer is served.

Independence witness: streamname, regseqrat, dyadicratcore, real, metric: $\mathsf{BishopRealLocatedness}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries finite windows, regular readback,

dyadic interval comparison, real sealing, metric comparison, locatedness queries, transport, replay, provenance, and local naming as one witness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealLocatedness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealLocatedness}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance comparison, a RealUp seal, MetricUp interval comparison, locatedness query rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealLocatedness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8169.1, Definition 8169.2, and Definition 8169.3 seed the located-real witness route from regular Cauchy names to finite interval answers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealLocatedness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, quotient-real equality, selected cuts, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-query exactness, RegSeqRat readback, DyadicRatCore comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealLocatednessUp

Chapter 8170

A Concrete Naming Certificate for $\mathsf{CompactModulusComposition}^{\uparrow}$

Following Bishop–Bridges constructive metric analysis and the compact-uniform-continuity route of chapter 9237, the $\mathsf{CompactModulusComposition}^{\uparrow}$ packet records the finite cover-level step that turns compact finite-cover rows and pointwise modulus rows into one uniform-modulus ledger. It sits beside compact data from chapter 60, continuous rows from chapter 59, metric comparison rows from chapter 57, rational tolerance rows from chapter 37 and chapter 38, and the uniform handoff of chapter 391. The packet is narrower than a composition theorem for already uniform maps: it names the finite compact-cover fold itself.

Definition 8170.1 (Compact modulus composition carrier)**.** *A* $\mathsf{CompactModulusComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, B, L, Q, R, T, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Compact}^{\uparrow}$ *row carrying the displayed finite cover or finite-net packet at the requested tolerance.*
2. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row whose pointwise modulus evidence is read at the cover centers named by X.*
3. *B is the finite cover bundle row. Its entries name centers, radii, membership certificates, and the cover incidence relation used by the compact row.*
4. *L is the local modulus ledger assigning to each displayed cover entry a pointwise rational precision row.*
5. *Q is the* $\mathsf{Rat}^{\uparrow}$ *rational fold row that compares the finitely many local precision and radius rows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and seal row consumed by the final uniform ledger.*
7. *T is the* $\mathsf{Met}^{\uparrow}$ *triangle and distance-comparison row that transports a source pair through the chosen cover center.*
8. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *handoff row produced from the finite cover, local modulus, rational fold, real tolerance, and metric rows.*
9. *H records componentwise* hsame *transport for compact cover, continuity, local modulus, rational fold, real seal, metric comparison, uniform handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a uniform-modulus request through $X, F, B, L, Q, R, T, U, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactModulusComposition}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host finite-subcover selection theorem, a selected global modulus, quotient point equality, open-cover compactness, choice over cover entries, or ambient completeness.

Theorem 8170.2 (Compact modulus composition NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusComposition}^{\uparrow}$ *carrier $K = (X, F, B, L, Q, R, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusComp}}$ *surface consists exactly of compact finite-cover admission through X, B, pointwise continuous-modulus admission through F, L, rational finite-fold exactness through Q, real tolerance sealing through R, metric triangle transport through T, uniform-continuity export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8170.1. The compact cover rows X, B, continuous row F, local modulus row L, rational fold Q, real seal R, metric row T, and uniform handoff U are visible coordinates of the same finite packet.

The local modulus rows are read only along the displayed cover entries. The rational fold Q compares those finitely many local rows and radius rows, so the packet produces a single tolerance-facing ledger without selecting a host-level global modulus.

The metric row T transports a source pair through the cover center chosen by the finite cover row, and U records the resulting uniform-continuity handoff. The rows H, C, P, N transport, replay, source, and name this same route.

Since the carrier has no coordinate for open-cover compactness, quotient point equality, cover-entry choice outside B, a selected global modulus, host finite-subcover selection, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8170.3 (Compact modulus composition uniform route)**.** *Every uniform-continuity consumer of an accepted* $\mathsf{CompactModulusComposition}^{\uparrow}$ *packet reaches its tolerance-only ledger through the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{Continuous}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformContinuity}^{\uparrow}.$$

More precisely, it must read the compact finite-cover rows X, B, the pointwise-modulus rows F, L, the rational finite-fold row Q, the real tolerance seal R, and the metric triangle row T before the uniform handoff U is exported.

Proof. Start with a uniform-modulus request. The replay row C routes the request first to the compact finite-cover data X, B, so a displayed cover entry is available before any local modulus row is consumed.

At the selected cover entry, the continuous row F and local ledger L expose the pointwise precision data. The rational fold row Q then supplies a finite lower-bound row across all displayed cover entries, and the real seal R records the tolerance being served.

Finally the metric row T gives the source and target triangle estimate needed by the uniform handoff U. Transport, provenance, and local naming preserve the same rows, so the consumer receives a finite compact-cover composition route rather than a host compactness theorem or a selected global modulus. □

Falsifiable prediction: Every accepted $\mathsf{CompactModulusComposition}^{\uparrow}$ read exposes compact finite-cover rows, continuous pointwise-modulus rows, a RatUp finite-fold row, RealUp tolerance sealing, MetricUp triangle transport, a UniformContinuity handoff, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before a tolerance-only uniform ledger is served.

Independence witness: compact, continuous, metric, rat, real, uniformcontinuity: $\mathsf{CompactModulusComposition}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries compact finite-cover rows, pointwise modulus rows, a rational fold, real tolerance sealing, metric triangle transport, uniform handoff, transport, replay, provenance, and local naming as one finite composition packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusComposition}^{\uparrow}$ packet is generated from compact finite-cover rows, Continuous pointwise-modulus rows, a RatUp finite-fold ledger, RealUp tolerance sealing, MetricUp triangle transport, a UniformContinuityUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8170.1, Definition 8170.2, and Definition 8170.3 seed the finite compact-cover composition route from pointwise moduli to a uniform ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected global moduli, quotient point equality, cover-entry choice outside the displayed finite cover, host finite-subcover selection, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-cover admission, pointwise-modulus admission, rational fold exactness, real tolerance sealing, metric triangle transport, uniform handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactModulusCompositionUp.

Chapter 8171

A Concrete Naming Certificate for $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$

Following Sierpinski and continuum theory, the $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ packet records the planar universal-curve boundary as finite compact-metric and square-refinement NameCert data. It sits after the compact-metric source of chapter 129, the Peano-continuum interface of chapter 2160, the planar retained-square surface of chapter 3960, the universal one-dimensional comparison packet of chapter 7789, and the terminal real readback of chapter 38. The packet names a finite planar curve boundary for downstream compact-continuum consumers; it does not assert a host hyperspace equality, open-cover compactness, selected branch points, quotient continua, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8171.1 (Sierpinski universal curve carrier)**.** *A* $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, P, S, M, G, W, R, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row supplying compact metric coverage before any planar boundary row is consumed;*
2. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row recording compact connected locally connected continuum data;*
3. *S is the* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *planar retained-square sibling row;*
4. *M is the* $\mathsf{MengerCurve}^{\uparrow}$ *universal one-dimensional comparison row;*
5. *G is the finite square-refinement row naming the retained and deleted planar-window stages;*
6. *W is the planar-window row exposing finite boundary arcs and hole-adjacency reads;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the finite coordinate windows used by G and W;*
8. *H records componentwise* hsame *transports for the compact, continuum, planar, square-refinement, readback, replay, provenance, and local naming rows;*
9. *C is the displayed* Cont *replay route for extending the finite square-refinement ledger;*
10. *Q is the* Pkg *provenance over* $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{MengerCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiUniversalCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, P, S, M, G, W, R, H, C, Q, N$. It has no coordinate for host hyperspace equality, open-cover compactness, selected planar continua, quotient point equality, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8171.2 (Sierpinski universal curve NameCert obligations)**.** *For every accepted* $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ *carrier*

$$U = (K, P, S, M, G, W, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiUniversalCurve}^{\uparrow}}$ *surface consists exactly of compact-metric exposure through K, Peano-continuum exposure through P, Sierpinski-carpet planar comparison through S, Menger-curve universal comparison through M, finite square-refinement exposure through G, planar-window boundary exposure through W, real readback through R, componentwise transport through H, displayed continuation replay through C, provenance through Q, and local naming through N.*

Proof. Project the accepted carrier through Definition 8171.1. The rows K, P, S, M expose the compact, continuum, planar-carpet, and universal-curve sibling data that a consumer may cite.

The finite planar boundary data are held by G, W, R. The square-refinement row records retained and deleted stages, the planar-window row records boundary arcs and adjacency reads, and the real readback row seals only those finite coordinate windows.

Finally use H, C, Q, N for transport, replay, source, and local naming. Since no carrier coordinate supplies host hyperspace equality, open-cover compactness, selected branch points, quotient continua, choice, or ambient completeness, those obligations cannot enter the local NameCert surface. □

Theorem 8171.3 (Sierpinski universal curve planar curve boundary)**.** *Every compact-planar-continuum consumer of an accepted* $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{PeanoContinuum}^{\uparrow},\ \mathsf{SierpinskiCarpet}^{\uparrow},\ \mathsf{MengerCurve}^{\uparrow} \longmapsto G, W, R \longmapsto H, C, Q, N.$$

The route may use the Sierpinski-carpet sibling as a planar retained-square boundary and the Menger-curve sibling as a universal one-dimensional comparison surface, but it exports no host hyperspace equality, no open-cover compactness theorem, no selected universal embedding, no quotient equality of continua, no choice principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier rows K, P, S, M. These rows place the packet inside the existing compact metric, Peano continuum, Sierpinski carpet, and Menger curve surfaces without identifying the new packet with any sibling.

Read the planar boundary through G, W, R. The square-refinement and planar-window rows are finite BHist data, and the $\mathsf{Real}^{\uparrow}$ row is only the coordinate readback for those displayed windows.

Transport and replay are then forced through H, C, Q, N. The packet has no row for a host hyperspace, an open-cover proof, a selected embedding, quotient continuum equality, choice, or ambient completeness, so a consumer cannot extract any of those principles from the boundary route. □

Falsifiable prediction: Every accepted $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ compact-planar-continuum read factors through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{MengerCurve}^{\uparrow}$, finite square-refinement rows, planar-window rows, $\mathsf{Real}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before using the curve boundary.

Independence witness: sierpinski-carpet, menger-curve, peano-continuum, compactmetric, real: $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ is not bijective to any listed sibling because it binds compact metric,

continuum, planar retained-square, universal-curve comparison, square-refinement, planar-window, real readback, transport, replay, provenance, and local naming rows into one planar universal-curve certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiUniversalCurve}^{\uparrow}}$.

Constructive story: A $\mathsf{SierpinskiUniversalCurve}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a PeanoContinuumUp row, a SierpinskiCarpetUp planar sibling row, a MengerCurveUp universal comparison row, finite square-refinement rows, planar-window boundary rows, RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiUniversalCurve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8171.1, Definition 8171.2, and Definition 8171.3 seed the compact-planar-continuum route over CompactMetricUp, PeanoContinuumUp, SierpinskiCarpetUp, MengerCurveUp, finite square-refinement, planar-window, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiUniversalCurve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host hyperspace equality, open-cover compactness, selected universal embeddings, quotient equality of continua, choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric exposure, continuum exposure, planar sibling exposure, universal-curve comparison, finite square-refinement exposure, planar-window readback, transport, replay, provenance, and local naming for BEDC.Derived.SierpinskiUniversalCurveUp.

Chapter 8172

A Concrete Naming Certificate for $\mathsf{LowerSemicontinuousRegularization}^{\uparrow}$

Following Bourbaki and Bishop-style constructive analysis, the $\mathsf{LowerSemicontinuousRegularization}^{\uparrow}$ packet records the finite BEDC surface by which a real-valued metric datum is replaced by its displayed lower-semicontinuous envelope. It sits after the metric carrier of chapter 57, the regular-rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the lower-semicontinuity packet of chapter 1536. The packet names only displayed finite windows and refusals; it does not import host infima, selected limits, quotient functions, countable choice, or ambient completion.

Definition 8172.1 (Semicontinuous envelope carrier)**.** *A* $\mathsf{LowerSemicontinuousRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, M, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *row naming the real-valued datum whose lower envelope is being read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row naming the displayed neighbourhood or ball requests over which the envelope is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational approximants attached to those metric windows.*
4. *D is the dyadic lower-bound window row, carrying the finite lower estimates that a consumer may inspect.*
5. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *row stating that accepted reads preserve lower-window stability under displayed metric refinement.*
6. *H records componentwise* hsame *transport for the source, metric, readback, dyadic-window, lower-stability, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from source data through metric windows, regular-rational readback, dyadic lower bounds, and the lower-semicontinuity row.*
8. *P is the* Pkg *provenance over* $\mathsf{LowerSemicontinuousRegularization}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerSemicontinuous}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuousRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted envelopes only through these nine rows. It has no coordinate for hidden infima, selected minimizing sequences, quotient functions, host real equality, countable choice, or ambient lower-completion principles.

Theorem 8172.2 (Semicontinuous envelope NameCert obligations)**.** *For every accepted* LowerSemicontinuousRegul *carrier*

$$E = (S, M, R, D, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerSemicontinuousRegularization}^{\uparrow}}$ *surface consists exactly of source-real exposure through* S*, metric-window exposure through* M*, regular-rational readback through* R*, dyadic lower-bound exposure through* D*, lower-semicontinuity stability through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8172.1. The source row, metric row, readback row, dyadic lower-window row, and lower-semicontinuity row are all displayed before the local name is admitted.

Read the analytic route in dependency order. The source real datum is inspected only through S; metric refinement requests are named by M; regular-rational witnesses are read back through R; and the dyadic lower-bound ledger D records the finite estimates available to consumers.

Finally use L, H, C, P, N. Lower-semicontinuity is a displayed stability row, and transport, replay, provenance, and local naming preserve the same packet. Since the carrier has no hidden infimum or selected-limit coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 8172.3 (Semicontinuous envelope lower-upper stability)**.** *Every accepted* LowerSemicontinuousRegulariza *consumer must preserve the dyadic lower-bound windows* D *together with the source, metric, readback, and lower-semicontinuity rows* S, M, R, L*. The packet may export lower-envelope stability under displayed metric refinement, but it cannot be consumed as a selected pointwise infimum, a hidden upper regularity theorem, a quotient-function equality, or a final real-name oracle.*

Proof. Start from the NameCert obligations of Definition 8172.2. They place D in the same packet as S, M, R, L, so lower bounds are not detachable annotations.

Apply the classifier clause in Definition 8172.1. Accepted comparison sees the dyadic window row and the lower-semicontinuity row together with componentwise transport and displayed replay. A consumer that drops D or L therefore leaves the accepted envelope surface.

The exclusions follow from the absent coordinates. The carrier contains no selected infimum, no hidden upper envelope, no quotient-function equality, and no final real-name oracle, while H, C, P, N only transport, replay, source, and name the displayed rows already present. □

Falsifiable prediction: A lower-envelope consumer that asks for a selected infimum or a quotient-function representative without displaying the dyadic lower-bound window row is rejected by the LowerSemicontinuousRegularizationUp classifier.

Independence witness: real, metric, regseqrat, lowersemicontinuous: none is bijective because the regularization packet binds source real data, metric windows, regular-rational readback, dyadic lower bounds, lower-semicontinuity stability, transport, replay, provenance, and local NameCert rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LowerSemicontinuousRegularization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LowerSemicontinuousRegularization}^{\uparrow}$ packet is generated from source Real rows, Metric window rows, RegSeqRat readback rows, dyadic lower-bound windows, a LowerSemicontinuous stability row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuousRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8172.1, Definition 8172.2, and Definition 8172.3 seed the finite lower-semicontinuous regularization route over $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuousRegularization}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected infima, quotient functions, countable choice, ambient completion, hidden upper regularity, or final real-name oracle behavior.

Upgrade path: Obligation closure requires checked carrier admission, dyadic lower-window exposure, lower-semicontinuity stability, transport, replay, provenance, and local naming for BEDC.Derived.LowerSemicontinuousRegularizationUp.

Chapter 8173

A Concrete Naming Certificate for $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$

The $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$ packet realizes the compact-uniform-modulus direction of chapter 9237 as finite law-facing NameCert data. It sits after the compact source of chapter 129, the continuous-map row of chapter 130, the tolerance-only modulus handoff of chapter 369, the law-certificate surface of chapter 1155, and the terminal real-name seal of chapter 38. The packet binds compact source, continuous map, modulus, lawlike claim, failure surface, transport, replay, provenance, and local naming before a compact law can be consumed.

Definition 8173.1 (Compact law failure-envelope carrier)**.** *A* $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, U, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, naming the compact finite-net or compact-window surface on which the law is offered.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, naming the continuous map or observable transformation whose lawlike behavior is being inspected.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, naming the tolerance-only modulus handoff extracted from compact and continuous data.*
4. *L is the* $\mathsf{LawCertificate}^{\uparrow}$ *row, recording the lawlike equation, invariant, or classifier claim attached to the compact-uniform route.*
5. *E is the failure-envelope row, listing compact windows, modulus refinements, source observations, or Real-facing reads that would defeat the law claim.*
6. *H records componentwise* hsame *transport for the compact source, continuous map, modulus, law-certificate, failure-envelope, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from compact-source admission through continuous-map inspection, uniform-modulus handoff, law-certificate reading, and failure-envelope exposure.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{LawCertificate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactLawFailureEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a compact law without a modulus row, a law certificate without a failure envelope, a total prediction over all compact refinements, hidden compactness, selected real limits, or ledger-free law status.

Theorem 8173.2 (Compact law failure-envelope NameCert obligations)**.** *For every accepted* $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$ *carrier*

$$K = (S, F, U, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactLawFailureEnvelope}^{\uparrow}}$ *surface consists exactly of compact-source exposure through* S*, continuous-map exposure through* F*, uniform-modulus exposure through* U*, law-certificate exposure through* L*, failure-envelope exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8173.1. The compact source, continuous map, uniform modulus, law certificate, and failure envelope are all displayed rows before the local name is public.

Read the compact-uniform route in order. The compact row supplies the source surface, the continuous-map row supplies the observable transformation, and the uniform-modulus row supplies the tolerance-only handoff. The law-certificate row then attaches the lawlike claim only after those rows are visible.

The failure-envelope row is part of the same NameCert surface. Componentwise transport, displayed replay, provenance, and local naming preserve this packet rather than replacing it by a compactness theorem or equation-only law, so the listed rows exhaust the local obligations. □

Theorem 8173.3 (Compact law failure-envelope classifier boundary)**.** *Every accepted* $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$ *consumer must preserve* $S, F, U, L, E, H, C, P, N$ *together. In particular, a compact law packet with a uniform modulus but no failure-envelope row is outside the classifier, and a route that keeps the law-certificate row while dropping compact source, continuous-map, modulus, or failure-surface data cannot export compact-law status.*

Proof. Use the NameCert obligations of Definition 8173.2. They make E a required row alongside the compact source, continuous-map, uniform-modulus, and law-certificate rows.

Apply the classifier clause in Definition 8173.1. Accepted comparison preserves the displayed packet through H, C, P, N, so a consumer that omits E or detaches L from S, F, U has left the accepted row surface.

The refusal clauses follow from absent coordinates. There is no hidden compactness row, selected real limit, total prediction, or ledger-free equation route from which the missing failure envelope could be reconstructed. □

The future Lean TasteGate surface for $\mathsf{CompactLawFailureEnvelope}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{CompactLaw}$ and $\mathsf{FieldFaithful}\ \mathsf{CompactLawFailureEnvelope}^{\uparrow}$ instances preserving $S, F, U, L, E, H, C, P, N$ as separate displayed rows.

Falsifiable prediction: A compact law packet with a uniform modulus but no failure-envelope row is rejected by the classifier.

Independence witness: compactmetric, continuousmap, uniformmodulus, lawcertificate: none is bijective because the new packet binds compact source, continuous map, modulus, lawlike claim, failure surface, and provenance rows in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLawFailureEnvelope}^{\uparrow}}$ **.**

Constructive story: A CompactLawFailureEnvelope$^{\uparrow}$ packet is generated from CompactMetric source rows, ContinuousMap rows, UniformModulus handoff rows, LawCertificate rows, explicit failure-envelope rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactLawFailureEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 8173.1, Definition 8173.2, and Definition 8173.3 seed the compact law failure-envelope route over CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, LawCertificate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactLawFailureEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, total compact-law prediction, hidden compactness, selected real limits, ledger-free equation status, or failure-envelope-free law export.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, continuous-map exposure, uniform-modulus handoff, law-certificate exposure, failure-envelope exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactLawFailureEnvelopeUp.

Chapter 8174

A Concrete Naming Certificate for $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ packet records finite-net compactness data for a displayed family of continuous maps over a compact metric source. It sits beside the compact source of chapter 129, the equicontinuity surface of chapter 4076, the continuous-map graph rows of chapter 130, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite data consumed by compact-uniform and Arzela–Ascoli routes; it does not assert open-cover compactness, selected subsequences, quotient function-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: Any compact-family consumer that reads a $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ packet while bypassing its compact source row, continuous-map family row, equicontinuity modulus row, finite-net row, RegSeqRat/Real readback rows, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the classifier.

Independence witness: compactmetric, equicontinuity, continuousmap, streamname, regseqrat, real: $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ is not bijective to any listed sibling because it binds compact source inspection, a continuous-map family ledger, equicontinuity modulus data, finite totally-bounded nets, regular rational readback, Real endpoint seals, transport, replay, provenance, and local naming in one packet.

Definition 8174.1 (Compact totally bounded function-family carrier)**.** *A* $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, E, B, W, R, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the compact finite-net windows used to inspect source points.*
2. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger whose graph rows expose the admissible maps on the source windows.*
3. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *modulus row, recording one displayed source-radius response for the family at each requested target tolerance.*

4. *B is the totally-bounded finite-net row for the image family, including the finite list of net centers and radius budgets visible to compact-family consumers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row through which source windows, family indices, and value samples are requested.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *value-readback row obtained from the displayed windows and dyadic tolerance ledgers.*
7. *V is the* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after the regular readback is exposed.*
8. *H records componentwise* hsame *transport for compact source rows, continuous-map rows, equicontinuity rows, finite-net rows, stream windows, readbacks, Real seals, replay rows, provenance, and local naming.*
9. *C records displayed* Cont *replay from a family compactness request through compact source inspection, graph evaluation, equicontinuity radius use, finite-net comparison, regular readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, E, B, W, R, V, H, C, P, N$. No coordinate names open-cover compactness, selected subsequences, quotient function-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactTotallyBoundedFunctionFamilyUp.CompactTotallyBoundedFunctionFamilyTasteGate`
`single carrier alignment`

Theorem 8174.2 (Compact totally bounded function-family NameCert obligations)**.** *For every accepted* $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ *carrier $K = (X, F, E, B, W, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact source, continuous-map family, equicontinuity modulus, totally-bounded finite-net, stream-window, regular-readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *compact source visibility, requiring every sampled source point to enter through the* $\mathsf{CompactMetric}^{\uparrow}$ *row X and the observation row W;*
3. *family graph visibility, requiring every sampled value to be read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger F before regular readback;*
4. *equicontinuity stability, requiring the modulus row E to mediate tolerance changes before a finite-net comparison is read;*
5. *finite-net exactness, requiring the totally-bounded row B to supply the only admissible net centers and radius budgets for the family image;*
6. *value readback, requiring every terminal value to pass through W, R, V before any Real-facing consumer reads it;*
7. *non-escape, requiring H, C, P, N to preserve the same displayed rows and to export no open-cover compactness, selected subsequence, quotient function-space equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation for the compact function family is stored outside these rows.

Proof. Project the carrier of Definition 8174.1. Its analytic coordinates are exactly the compact source row, the continuous-map family ledger, the equicontinuity modulus row, the totally-bounded finite-net row, the stream windows, the regular readback, and the Real seal.

Compact source visibility comes from X and W. A sampled source point has no independent coordinate, so it must be requested through the displayed observation row and then interpreted by the compact-metric source packet.

Family graph visibility comes from F. The carrier stores graph reads only as continuous-map rows, hence a sampled value cannot be read before the relevant family graph row is displayed.

Equicontinuity stability and finite-net exactness are the rows E and B. The modulus row translates tolerance requests into source-window control, and the finite-net row is the only place where net centers and radius budgets are named.

Finally inspect H, C, P, N. These rows transport, replay, record provenance for, and name exactly the displayed packet, so the excluded principles have no row from which a local certificate could read them. □

Theorem 8174.3 (Compact totally bounded function-family finite-net consumer)**.** *Every compact-uniform or Arzela–Ascoli-facing consumer of an accepted* $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{Equicontinuity}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The route exposes compact source windows, graph samples, equicontinuity moduli, finite totally-bounded net centers, regular value readbacks, and terminal Real seals. It exports no open-cover compactness, selected subsequence, quotient function-space equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted carrier $K = (X, F, E, B, W, R, V, H, C, P, N)$. A compact-uniform consumer first reads the source windows through X and W, because the packet has no host source-point selector.

The value side then reads F, R, and V. The continuous-map family ledger supplies graph samples, the regular-rational row records their finite readback, and the Real seal is reached only after that readback is visible.

The compact-family comparison is confined by E and B. Equicontinuity supplies the tolerance-to-window handoff, while the finite-net row supplies the displayed centers and radius budgets consumed by the downstream compactness route.

The structural rows H, C, P, N preserve exactly this route. Since the carrier has no row for open covers, selected subsequences, quotient function-space equality, countable choice, or ambient Real completeness, the consumer cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMapUp family rows, an EquicontinuityUp modulus row, a totally-bounded finite-net row, StreamNameUp observation rows, RegSeqRatUp value readback rows, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8174.1, Definition 8174.2, and Definition 8174.3 seed the finite-net compactness data for displayed continuous-map families.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactTotallyBoundedFunctionFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequences, quotient function-space equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source visibility, family graph visibility, equicontinuity stability, finite-net exactness, value readback, transport, replay, and non-escape rows for BEDC.Derived.CompactTotallyBounded

Chapter 8175

A Concrete Naming Certificate for $\mathsf{RealSegment}^{\uparrow}$

Following Bishop and Bridges on located intervals in constructive real analysis, the $\mathsf{RealSegment}^{\uparrow}$ packet records a closed real segment by finite endpoint, located-order, apartness, and tolerance rows rather than by a host subset of an ambient real line. It sits after the terminal real-name seal of chapter 38, the regular rational readback surface of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance core of chapter 291. The packet gives compact-interval and uniform-modulus consumers a local interval object without importing Heine–Borel, open-cover compactness, sequential compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8175.1 (RealSegment carrier)**.** *A* $\mathsf{RealSegment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, O, A, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *U is the upper endpoint* $\mathsf{Real}^{\uparrow}$ *seal.*
3. *O is the located order row witnessing that the endpoint comparison is read as displayed constructive order data.*
4. *A is the located apartness row that separates exterior probes from the accepted closed segment boundary.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-window ledger for interval enclosures, endpoint budgets, and refinement thresholds.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint approximants and accepted subinterval endpoints.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that exposes only bounded endpoint and tolerance observations.*
8. *H records componentwise* hsame *transport for endpoint seals, located order, apartness, dyadic windows, regular readback, finite windows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a segment request through L, U, O, A, D, R, W.*
10. *P is the* Pkg *provenance over* $\mathsf{RealSegment}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSegment}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for Heine–Borel compactness, open-cover compactness, sequential compactness, a selected midpoint process, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8175.2 (RealSegment NameCert obligations)**.** *For every accepted* $\mathsf{RealSegment}^{\uparrow}$ *carrier*

$$S = (L, U, O, A, D, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSegment}^{\uparrow}}$ *surface consists exactly of the following five rows of obligations:*

1. *endpoint admission for the lower and upper* $\mathsf{Real}^{\uparrow}$ *seals L, U;*
2. *located boundary evidence through the order row O and apartness row A;*
3. *finite tolerance exposure through the dyadic ledger D, regular rational readback R, and stream window W;*
4. *componentwise transport and displayed replay through H, C;*
5. *source provenance and local naming through P, N.*

Every accepted interval read is therefore routed through $L, U, O, A, D, R, W, H, C, P, N$, and no certificate obligation is stored outside those rows.

Proof. Project the carrier of Definition 8175.1. The endpoint seals L, U are the only terminal real-name rows, while O, A are the only rows that can witness located order and apartness at the boundary.

The finite tolerance rows are D, R, W. They expose dyadic budgets, regular rational endpoint approximants, and finite observation windows before any downstream compact-interval or modulus consumer reads the segment.

The structural rows H, C, P, N transport, replay, source, and name exactly the same packet. Since no excluded coordinate occurs in the carrier, the five listed obligation groups exhaust the local NameCert surface and do not export compactness or ambient completeness. □

Theorem 8175.3 (RealSegment located subinterval transport)**.** *Let $S = (L, U, O, A, D, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealSegment}^{\uparrow}$ *carrier. If a subinterval request refines the endpoint classifier through transported* hsame *rows and a smaller dyadic tolerance window, then the accepted subinterval row remains inside the same displayed route*

$$L, U, O, A, D, R, W, H, C, P, N.$$

Endpoint classifier transport and dyadic tolerance refinement therefore preserve accepted subinterval rows without introducing Heine–Borel compactness, open-cover compactness, sequential compactness, selected witnesses, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint rows L, U and the located boundary rows O, A. Transport along H preserves the componentwise endpoint classifier data, so the subinterval request still reads the same lower and upper real-name seals together with the same located order and apartness evidence.

Next refine the tolerance window. The dyadic ledger D records the smaller budget, while R and W expose the corresponding regular rational endpoints and finite observation windows. Thus the refined subinterval is admitted through the same tolerance rows rather than through a new compactness principle.

Finally replay the transported request through C and source it through P, N. The accepted subinterval row is still a row of the original finite packet, so no excluded compactness, quotient, choice, or ambient completeness endpoint is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSegment}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSegment}^{\uparrow}$ packet is generated from two RealUp endpoint seals, located order and apartness evidence, dyadic tolerance windows, regular rational readback, finite stream windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSegment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8175.1, Definition 8175.2, and Definition 8175.3 seed the located closed real interval route over endpoint, order, apartness, dyadic, readback, window, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSegment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel, open-cover compactness, sequential compactness, selected subinterval witnesses, quotient real equality, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint locatedness, apartness transport, dyadic-window refinement, regular-rational readback, finite-window replay, provenance, and local naming for BEDC.Derived.RealSegmentUp.

Chapter 8176

A Concrete Naming Certificate for $\mathsf{ResidueTheorem}^{\uparrow}$

Following Ahlfors and Conway, the $\mathsf{ResidueTheorem}^{\uparrow}$ packet records the finite contour-residue surface by which a holomorphic-domain read with a displayed finite pole window is routed to a contour-integral ledger. It sits after the complex carrier of chapter 39, the holomorphic source of chapter 68, the contour-integral operation of chapter 521, the point-evaluation route of chapter 4136, and the winding-count ledger of chapter 7193. The packet names only a finite contour, a displayed holomorphic-domain row outside the recorded pole window, residue rows, winding-index rows, contour-integral rows, transport, replay, provenance, and local naming; it does not import analytic continuation, unrestricted meromorphic function theory, selected roots, quotient contours, or classical choice.

Definition 8176.1 (Residue theorem carrier)**.** *A* $\mathsf{ResidueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, D, P, Z, W, I, L, T, C, Q, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{PLContour}$ *row with* $\mathsf{Complex}^{\uparrow}$ *endpoint, segment, orientation, and non-boundary incidence data.*
2. *D is the* $\mathsf{Holomorphic}^{\uparrow}$ *domain row on the displayed neighbourhood of the contour image after the finite pole window has been removed.*
3. *P is the finite pole-window row, listing the inspected pole names, local punctured neighbourhood rows, and exclusion evidence from the contour image.*
4. *Z is the residue row, giving one local* $\mathsf{Residue}^{\uparrow}$ *datum for each displayed pole in P.*
5. *W is the winding-index ledger, read through the* $\mathsf{ArgumentPrinciple}^{\uparrow}$ *counting surface when the contour winds around a displayed pole.*
6. *I is the* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row for the contour integral of the displayed holomorphic-domain input around G.*
7. *L is the residue-sum ledger, pairing the contour-integral row I with the finite sum of the indexed residue rows Z weighted by W.*
8. *T records componentwise* hsame *transport for the contour, holomorphic-domain, pole-window, residue, winding-index, contour-integral, and residue-sum rows.*

9. *C records displayed* Cont *replay for the same finite contour-residue route.*
10. *Q is the* Pkg *provenance over* ResidueTheorem$^{\uparrow}$, Complex$^{\uparrow}$, Holomorphic$^{\uparrow}$, ContourIntegralOperation$^{\uparrow}$, CauchyIntegralFormula$^{\uparrow}$, ArgumentPrinciple$^{\uparrow}$, Residue$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ResidueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $G, D, P, Z, W, I, L, T, C, Q, N$. The carrier has no coordinate for analytic continuation, unrestricted meromorphic function theory, selected roots, quotient contours, ambient path homotopy, or classical choice.

Theorem 8176.2 (Residue theorem NameCert obligations)**.** *For every accepted* ResidueTheorem$^{\uparrow}$ *carrier*

$$R = (G, D, P, Z, W, I, L, T, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{ResidueTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the contour row, holomorphic-domain row, finite pole-window row, residue rows, winding-index ledger, contour-integral operation row, residue-sum ledger, transport, replay, provenance, and local naming;*
2. *contour exposure, requiring all integral and index reads to begin with the displayed* PLContour *row G over* Complex$^{\uparrow}$ *endpoints and segments;*
3. *holomorphic-domain exposure, requiring D to display the function domain away from the finite pole window P before any residue row is read;*
4. *local residue exposure, requiring every row of Z to be attached to an inspected pole of P and to the point-evaluation discipline of* CauchyIntegralFormula$^{\uparrow}$*;*
5. *winding-index exposure, requiring W to be read through the finite counting surface of* ArgumentPrinciple$^{\uparrow}$ *before the residue-sum ledger is public;*
6. *contour-integral exposure, requiring I to be the displayed* ContourIntegralOperation$^{\uparrow}$ *row for the same contour and domain data;*
7. *residue-sum exactness, requiring L to pair only the displayed contour-integral row with the finite indexed residue rows;*
8. *componentwise transport, displayed replay, provenance, and local naming through T, C, Q, N;*
9. *nonescape, excluding analytic continuation, unrestricted meromorphic function theory, selected roots, quotient contours, ambient path homotopy, and classical choice.*

No residue-theorem obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 8176.1. The contour row G, holomorphic-domain row D, pole-window row P, residue rows Z, winding ledger W, contour-integral row I, and residue-sum ledger L are the only analytic rows available to the local certificate.

The local residue part is finite. Each residue row in Z is indexed by a displayed pole in P, and the Cauchy-integral formula dependency is read only as the point-evaluation and local kernel discipline attached to those displayed neighbourhoods.

The contour side is finite as well. The winding ledger W is read through the argument-principle counting surface for the same contour, and the integral row I is the contour-integral operation row for G and D.

Finally the ledger L pairs only those displayed rows, while T, C, Q, N transport, replay, source, and name the same packet. Since none of the excluded objects occurs as a carrier coordinate, the

local NameCert surface cannot export analytic continuation, unrestricted meromorphic function theory, selected roots, quotient contours, ambient path homotopy, or classical choice. □

Falsifiable prediction: An accepted $\mathsf{ResidueTheorem}^{\uparrow}$ read is refuted if it omits the displayed contour row, holomorphic-domain row, finite pole window, residue rows, winding-index ledger, contour-integral row, residue-sum ledger, transport row, replay row, provenance row, or local NameCert row.

Independence witness: complex, holomorphic, contourintegraloperation, cauchy-integral-formula, argument-principle: $\mathsf{ResidueTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds contour rows, holomorphic-domain rows, finite pole-window rows, local residue data, winding-index ledgers, contour-integral rows, residue-sum ledgers, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ResidueTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{ResidueTheorem}^{\uparrow}$ packet is generated from a displayed PLContour row over ComplexUp endpoints, a HolomorphicUp domain row away from a finite pole window, residue rows, winding-index rows, a ContourIntegralOperationUp row, a residue-sum ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResidueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8176.1 and Definition 8176.2 seed the finite contour-residue NameCert surface over $\mathsf{Complex}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{ContourIntegralOperation}^{\uparrow}$, $\mathsf{CauchyIntegralFormula}^{\uparrow}$, $\mathsf{ArgumentPrinciple}^{\uparrow}$, $\mathsf{Residue}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResidueTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, analytic continuation, unrestricted meromorphic function theory, selected roots, quotient contours, ambient path homotopy, or classical choice.

Upgrade path: Obligation closure requires checked carrier admission, contour exposure, holomorphic-domain exposure, finite pole-window exposure, local residue exposure, winding-index exactness, contour-integral routing, residue-sum exactness, transport, replay, provenance, and local naming for BEDC.Derived.ResidueTheoremUp.

Chapter 8177

A Concrete Naming Certificate for $\mathsf{ZTransform}^{\uparrow}$

Following Widder and standard discrete transform theory, the $\mathsf{ZTransform}^{\uparrow}$ packet records a one-sided coefficient stream by finite shift, formal-power, radius, tolerance, transport, provenance, and local NameCert rows. It sits beside the coefficient-window surface of chapter 3356, the exponential-kernel transform route of chapter 3649, the positive-index series surface of chapter 70, the analytic power-series route of chapter 1583, the stream naming discipline of chapter 126, and the regular rational readback rows of chapter 301. The packet names only finite one-sided transform data; it does not assert analytic convergence, inversion, a bilateral transform, host infinite sums, quotient series, or Lean verification.

Definition 8177.1 (ZTransform carrier)**.** *A* $\mathsf{ZTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (W, S, F, R, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite one-sided coefficient-window row, exposing only inspected nonnegative indices and their coefficient reads.*
2. *S is the shift row, recording how a finite delay or advance request reindexes the displayed one-sided window.*
3. *F is the formal-power readback row, reading the inspected coefficients as a finite* $\mathsf{FPS}^{\uparrow}$ *or* $\mathsf{GeneratingFunction}^{\uparrow}$ *prefix.*
4. *R is the radius ledger, recording the displayed convergence-radius or subradius data available to downstream* $\mathsf{RealPowerSeries}^{\uparrow}$ *consumers.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for coefficient budgets, radius budgets, and endpoint readback requests.*
6. *H records componentwise* hsame *transport for coefficient windows, shift rows, formal-power readback, radius and tolerance ledgers, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a transform request through W, S, F, R, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ZTransform}^{\uparrow}$, $\mathsf{GeneratingFunction}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{Dirichlet}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FPS}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{ZTransform}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $W, S, F, R, T, H, C, P, N$. It has no coordinate for analytic convergence, inverse-transform reconstruction, bilateral negative-index data, selected infinite sums, quotient equality of series, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8177.2 (ZTransform NameCert obligations)**.** *For every accepted* $\mathsf{ZTransform}^\uparrow$ *carrier $Z = (W, S, F, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ZTransform}^\uparrow}$ *surface consists exactly of one-sided coefficient-window admission, shift-row admission, formal-power readback, radius-ledger exposure, dyadic tolerance exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every transform read factors through the finite route*

$$W, S, F, R, T \longmapsto H, C, P, N.$$

It exports no analytic convergence theorem, inversion theorem, bilateral transform, host infinite sum, quotient series equality, or Lean-checked target.

Proof. Project the carrier of Definition 8177.1. The one-sided coefficient window W, shift row S, formal-power readback F, radius ledger R, and tolerance ledger T are the only transform-facing rows in the packet.

The structural rows preserve that visible route. Componentwise hsame transport can identify only displayed rows, and Cont replay can move only along the finite route from coefficient windows through local naming.

The package and local NameCert rows cite the same packet. Since no excluded coordinate occurs in Z, the certificate cannot export analytic convergence, inversion, bilateral negative-index data, selected infinite sums, quotient series equality, or Lean verification. □

Theorem 8177.3 (ZTransform generating-function handoff)**.** *Every coefficient-facing consumer of an accepted* $\mathsf{ZTransform}^\uparrow$ *carrier reaches* $\mathsf{GeneratingFunction}^\uparrow$ *or* $\mathsf{RealPowerSeries}^\uparrow$ *data only through the displayed one-sided coefficient window, shift row, formal-power readback, radius ledger, dyadic tolerance ledger, transport, replay, provenance, and local naming rows. The handoff is therefore a finite formal-power route, not an analytic inversion principle, bilateral transform, host summation, or quotient-series equality.*

Proof. Start with the coefficient window W. It exposes only the inspected one-sided coefficients, so a consumer cannot obtain negative-index or bilateral data from the packet.

Read the shift row S and formal-power row F. The shift row reindexes the displayed finite window, while F hands that same window to $\mathsf{GeneratingFunction}^\uparrow$ or $\mathsf{FPS}^\uparrow$ before any $\mathsf{RealPowerSeries}^\uparrow$ endpoint consumer can inspect radius information.

Finally use the radius and tolerance ledgers R, T together with H, C, P, N. These rows keep the handoff inside finite NameCert data, so the consumer receives no analytic inversion theorem, host infinite sum, quotient series equality, countable-choice selector, or ambient completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZTransform}^\uparrow}$.

Constructive story: A $\mathsf{ZTransform}^\uparrow$ packet is generated from one-sided coefficient windows, shift rows, formal-power readback, radius ledgers, dyadic tolerance ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZTransform}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 8177.1, Definition 8177.2, and Definition 8177.3 seed the finite one-sided transform route over coefficient windows, shifts, formal-power readback, radius and tolerance rows, transport, replay, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZTransform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic convergence, inversion theorems, bilateral transforms, host infinite sums, quotient series equality, Lean verification, bridge checking, countable choice, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, one-sided coefficient-window exactness, shift-row compatibility, formal-power readback, radius and tolerance ledger coverage, transport, replay, provenance, and local naming for BEDC.Derived.ZTransformUp.

Chapter 8178

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$

The $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$ packet realizes the observer phase-distance shell route of chapter 9463 as finite boundary-shadow, visual-boundary, observer-window, phase-distance, shell, provenance, and local NameCert rows. It sits beside the observer shell of chapter 5180, the phase-distance observer-expansion packet of chapter 6017, the phase-space packet of chapter 7108, the boundary-shadow packet of chapter 3813, and the visual-boundary packet of chapter 4181. The packet names the joint observer shell and phase-distance boundary rather than a host hyperbolic space, physical observer, quotient ideal boundary, selected geodesic ray, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8178.1 (Hyperbolic observer phase-distance shell carrier)**.** *A* $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, V, O, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row exposing the boundary-shadow window seen by the observer.*
2. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row reading the same shadow window as visual-boundary data.*
3. *O is the* $\mathsf{HyperbolicObserverShell}^{\uparrow}$ *row that records the observer-window shell over boundary and expansion data.*
4. *D is the* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *row exposing the finite phase-distance comparison for the same observer window.*
5. *Q is the* $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$ *row that keeps the reversible phase-space route tied to displayed disk, boundary, refusal, and provenance rows.*
6. *R is the shell-radius and provenance ledger, recording which boundary-shadow, visual-boundary, observer-window, and phase-distance rows are jointly read.*
7. *H records componentwise* hsame *transport for boundary-shadow, visual-boundary, observer-shell, phase-distance, phase-space, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an observer phase-distance request through $B, V, O, D, Q, R, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$, $\mathsf{HyperbolicObserverShell}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicPhaseSpace}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, HyperbolicVisualBounda BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, V, O, D, Q, R, H, C, P, N$. It has no coordinate for a completed hyperbolic space, physical observer trajectory, quotient ideal-boundary equality, selected geodesic ray, hidden visual-boundary completion, or ambient real completeness.

Theorem 8178.2 (Hyperbolic observer phase-distance shell NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$ *carrier $S = (B, V, O, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, visual-boundary readback, observer-shell admission, phase-distance comparison, hyperbolic phase-space routing, shell-radius provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8178.1. The visible geometric rows are B, V, O, D, Q, R, and each is displayed before transport, replay, provenance, or local naming is consumed.

The boundary rows B, V anchor the shell at the visual boundary. The observer row O and phase-distance row D force the observer-window and distance comparison to be read together, while Q keeps the route inside the finite phase-space packet.

The structural rows H, C, P, N preserve that same finite packet. Since no excluded coordinate occurs in S, the NameCert obligations cannot export a completed hyperbolic space, physical observer trajectory, quotient ideal boundary, selected geodesic ray, hidden visual-boundary completion, or ambient real completeness. □

Theorem 8178.3 (Hyperbolic observer phase-distance shell vision handoff)**.** *The* HyperbolicObserverPhaseDistance *packet realizes the boundary-shadow, phase-distance, observer-window, and provenance rows of chapter 9463 as one finite NameCert carrier. Every consumer of the shell must first read* HyperbolicBoundaryShadow $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverShell}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, *and* HyperbolicPhaseSpace *rows, then the ledger, transport, replay, provenance, and local naming rows.*

Proof. Use the obligation surface of Definition 8178.2. It orders every shell read through the boundary-shadow, visual-boundary, observer-shell, phase-distance, phase-space, and shell-radius rows before any consumer receives a result.

This route is a vision handoff because the cited vision chapter names exactly the observer boundary, phase-distance, shell, and provenance rows that the carrier displays. The handoff is therefore concrete finite NameCert data rather than a prose-only observer geometry.

Finally inspect the refused coordinates in Definition 8178.1. The carrier contains no completed hyperbolic space, physical observer path, quotient ideal-boundary equality, selected geodesic ray, hidden visual-boundary completion, or ambient completeness row, so no such object can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicObserverPhaseDistanceShell}^{\uparrow}$ read exposes HyperbolicBoundaryShadow, HyperbolicVisualBoundary, HyperbolicObserverShell, PhaseDistanceObserverExpansion, HyperbolicPhaseSpace, shell-radius provenance, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before the phase-distance shell handoff is consumed.

Independence witness: hyperbolic-observer-shell, phase-distance-observer-expansion, hyperbolic-phase-space, hyperbolic-boundary-shadow, hyperbolic-visual-boundary: $\mathsf{HyperbolicObserverShell}^{\uparrow}$

lacks the phase-distance row, PhaseDistanceObserverExpansion$^{\uparrow}$ lacks the observer shell and boundary-shadow ledger, HyperbolicPhaseSpace$^{\uparrow}$ lacks the observer-window shell, HyperbolicBoundaryShadow$^{\uparrow}$ lacks the phase-distance comparison, and HyperbolicVisualBoundary$^{\uparrow}$ lacks the joint observer-shell provenance. None is bijective to HyperbolicObserverPhaseDistanceShell$^{\uparrow}$ under displayed NameCert-row preservation.

Closure status for NameCert$_{\text{HyperbolicObserverPhaseDistanceShell}^{\uparrow}}$.

Constructive story: A HyperbolicObserverPhaseDistanceShell$^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicVisualBoundary rows, HyperbolicObserverShell rows, PhaseDistanceObserverExpansion rows, HyperbolicPhaseSpace rows, a shell-radius provenance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicObserverPhaseDistanceShell}^{\uparrow}}$, seed)

Scope closed: Definition 8178.1, Definition 8178.2, and Definition 8178.3 seed the finite observer phase-distance shell over boundary-shadow, visual-boundary, observer-shell, phase-distance, phase-space, ledger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicObserverPhaseDistanceShell}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic spaces, physical observer trajectories, quotient ideal-boundary equality, selected geodesic rays, hidden visual-boundary completion, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow admission, visual-boundary readback, observer-shell admission, phase-distance comparison, phase-space routing, shell-radius provenance, transport, replay, and local naming for BEDC.Derived.HyperbolicObserverPhaseDistanceShellUp.

Chapter 8179

A Concrete Naming Certificate for $\mathsf{HilbertSchmidt}^{\uparrow}$

Following Schatten and von Neumann operator-ideal theory, the $\mathsf{HilbertSchmidt}^{\uparrow}$ packet records the finite square-summable coefficient surface by which a bounded Hilbert operator may be read as Hilbert–Schmidt before compact, trace-class, or spectral consumers inspect it. It sits after the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the scalar-bound surface of chapter 3603, the finite-image route of chapter 6952, and the operator-ideal boundary of chapter 213. The packet names only displayed Hilbert rows, a bounded linear action, a finite orthonormal-basis window, a square-sum ledger, transport, replay, provenance, and local naming; it does not select an infinite basis, assert trace theory, identify host Hilbert-space equality, or import compactness from outside the displayed rows.

Definition 8179.1 (Hilbert–Schmidt carrier)**.** *A* $\mathsf{HilbertSchmidt}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (H, L, B, S_{\mathrm{sq}}, T, R, P, N)$$

with the following displayed rows:

1. *H is the Hilbert source row, carrying the source and target* $\mathsf{Hilbert}^{\uparrow}$ *endpoints together with the inner-product and norm data consumed by operator readers.*
2. *L is the bounded* $\mathsf{LinMap}^{\uparrow}$ *row for the same endpoints, read through the bounded-operator and* $\mathsf{OperatorNorm}^{\uparrow}$ *surfaces before any ideal consumer appears.*
3. *B is a finite orthonormal-basis window: a displayed finite list of source vectors, their orthonormality ledger, and the corresponding coefficient rows for the images under L.*
4. *S_{sq} is the square-summable coefficient ledger, recording the finite sum of squared image coefficients over the window B and the scalar endpoint that bounds that displayed sum.*
5. *T records componentwise* hsame *transport for the Hilbert rows, bounded linear-map row, basis window, square-sum ledger, replay, provenance, and local name.*
6. *R records displayed* Cont *replay from Hilbert-source observations through the bounded action and coefficient ledger into operator-ideal and compact-operator consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{HilbertSchmidt}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{FiniteRankOperator}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertSchmidt}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, L, B, S_{\mathrm{sq}}, T, R, P, N$. It has no coordinate for a selected infinite orthonormal basis, host Hilbert-space equality, trace-class membership, singular-value decomposition, spectral theorem, or ambient compactness principle.

Theorem 8179.2 (Hilbert–Schmidt NameCert obligations)**.** *For every accepted* $\mathsf{HilbertSchmidt}^{\uparrow}$ *carrier* $S = (H, L, B, S_{\mathrm{sq}}, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HilbertSchmidt}^{\uparrow}}$ *surface consists exactly of carrier admission for the Hilbert endpoints, bounded linear action, finite orthonormal-basis window, square-sum ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; basis-window invariance for replacing B by a transported finite orthonormal window over the same Hilbert rows; square-sum ledger stability under the same transport; operator-ideal handoff through the bounded action and displayed square bound; and nonescape from host Hilbert-space equality, selected infinite bases, trace theory, spectral data, or ambient compactness.*

Proof. Project the carrier rows. The Hilbert endpoints, bounded linear action, finite basis window, square-sum ledger, transport, replay, provenance, and local name are the complete displayed inventory.

Basis-window invariance is a statement about transported finite windows. The row T moves the Hilbert data, the list B, and the coefficient ledger together, so a consumer may compare two displayed finite windows only through componentwise hsame transport.

Square-sum ledger stability follows from the same finite route. The scalar endpoint attached to S_{sq} is read after the bounded action and the coefficient rows have been displayed, and R replays that order for downstream consumers.

Finally, the operator-ideal handoff consumes only L and S_{sq} through P, N. Since no row stores a selected infinite basis, host equality of Hilbert spaces, trace-class membership, spectral data, or ambient compactness, those objects are refused by the local certificate. □

Theorem 8179.3 (Hilbert–Schmidt operator-norm bound)**.** *Every* $\mathsf{OperatorNorm}^{\uparrow}$ *or* $\mathsf{OperatorIdeal}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{HilbertSchmidt}^{\uparrow}$ *carrier must pass through the finite route*

$$H, L, B, S_{\mathrm{sq}}, T, R, P, N.$$

The displayed square-sum ledger supplies the finite scalar bound consumed by the operator-norm boundary for the same bounded action L, and this bound is the only Hilbert–Schmidt data available to $\mathsf{OperatorIdeal}^{\uparrow}$ *and* $\mathsf{CompactOperator}^{\uparrow}$ *consumers at this paper site.*

Proof. Start from the bounded linear-map row L over the Hilbert endpoints H. The operator-norm reader receives the action only after the same packet has displayed its finite coefficient window B.

Read the square-sum ledger next. The row S_{sq} records the finite scalar bound that dominates the displayed image coefficients, so the operator-norm consumer sees a finite bound-facing certificate rather than a host supremum.

The ideal and compact consumers then read through T, R, P, N. These rows transport, replay, source, and name the same bounded action and square ledger. They do not add trace data, selected compactness, spectral values, or equality of ambient Hilbert spaces. □

Falsifiable prediction: Every accepted $\mathsf{HilbertSchmidt}^{\uparrow}$ consumer must display Hilbert source rows, a bounded LinearMapUp action, a finite orthonormal-basis window, a square-sum ledger, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows before Hilbert–Schmidt operator data is available.

Independence witness: hilbert, linearmap, operator-norm, finite-rank-operator, operatorideal: no listed sibling is bijective to $\mathsf{HilbertSchmidt}^{\uparrow}$ because this packet jointly carries Hilbert endpoints,

bounded action, finite orthonormal coefficient windows, square-sum ledger stability, operator-ideal handoff, transport, replay, provenance, and local naming in one operator-facing surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertSchmidt}^{\uparrow}}$.

Constructive story: A $\mathsf{HilbertSchmidt}^{\uparrow}$ packet is generated from Hilbert source rows, a bounded LinearMapUp action, a finite orthonormal-basis window, a square-sum coefficient ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertSchmidt}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8179.1, Definition 8179.2, Definition 8179.3, the falsifiable prediction, and the independence witness seed the finite Hilbert–Schmidt operator route over Hilbert, linear-map, operator-norm, finite-rank, compact-operator, operator-ideal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertSchmidt}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trace-class theory, spectral decomposition, selected infinite orthonormal bases, host Hilbert-space equality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-window invariance, square-sum ledger stability, operator-ideal handoff, operator-norm bound consumption, transport, replay, provenance, and local naming for BEDC.Derived.HilbertSchmidtUp.HilbertSchmidtNameCert.

Chapter 8180

A Concrete Naming Certificate for $\mathsf{WhittakerShannonSampling}^{\uparrow}$

Following Whittaker and Shannon sampling theory, the $\mathsf{WhittakerShannonSampling}^{\uparrow}$ packet records the finite BEDC boundary by which a displayed bandlimited row, a sampling grid, and a reconstruction kernel may be consumed without importing an infinite host Fourier series. It sits downstream of the finite grid discipline of chapter 451, the transform rows of chapter 4025, the observed stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite sampling bridge for later Fourier and Real-completion consumers; it is not a theorem asserting host-level convergence of all sinc series, quotient equality of functions, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8180.1 (Whittaker-Shannon sampling carrier)**.** *A* $\mathsf{WhittakerShannonSampling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, G, K, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite bandlimit row, naming the Fourier-side cutoff, the admitted frequency window, and the aliasing exclusions read through* $\mathsf{FourierTransform}^{\uparrow}$*.*
2. *G is the sampling-grid row, displaying the finite grid points, mesh denominator, phase offsets, and the finite-grid compatibility inherited from* $\mathsf{FinitePhaseSampling}^{\uparrow}$*.*
3. *K is the reconstruction-kernel row, recording the displayed kernel values and truncation window used by the sampled reconstruction route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *sample row, exposing exactly the observed sample stream entries consumed by the packet.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same grid, kernel, and sample entries.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after B, G, K, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for the bandlimit row, grid row, kernel row, sample stream row, regular-rational readback, real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a finite sampling request through B, G, K, S, Q, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{WhittakerShannonSampling}^{\uparrow}$*,* $\mathsf{FinitePhaseSampling}^{\uparrow}$*,* $\mathsf{FourierTransform}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{WhittakerShannonSampling}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite Fourier-series object, quotient equality of functions, unbounded choice of samples, host sinc convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8180.2 (Whittaker-Shannon sampling NameCert obligations)**.** *For every accepted* WhittakerShannonSam *carrier*

$$W = (B, G, K, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WhittakerShannonSampling}^{\uparrow}}$ *surface consists exactly of the five analytic rows B, G, K, S, Q followed by the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance P, and local naming N. A downstream consumer must therefore read the finite bandlimit row, the displayed sampling-grid row, the reconstruction-kernel row, the* $\mathsf{StreamName}^{\uparrow}$ *sample row, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before it reaches any Real-facing handoff. The certificate refuses host infinite Fourier series, quotient function equality, unbounded sample choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8180.1. The analytic sampling surface is exactly the finite tuple B, G, K, S, Q: bandlimit data, sampling grid, reconstruction kernel, observed samples, and regular-rational readback.

The finite-grid row G is read through chapter 451, while B and K provide the Fourier-side cutoff and reconstruction kernel consumed by the same displayed route. The sample row S then binds the grid to concrete $\mathsf{StreamName}^{\uparrow}$ observations, and Q records the regular rational readback over those observations.

The rows E, H, C, P, N seal, transport, replay, source, and name the same finite packet. Since no carrier coordinate stores an infinite Fourier series, quotient equality of functions, an unbounded sample selector, or ambient Real completeness, no accepted consumer can project those objects from the local certificate. □

Falsifiable prediction: An accepted $\mathsf{WhittakerShannonSampling}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading $B, G, K, S, Q, E, H, C, P, N$ refutes the carrier surface.

Independence witness: finite-phase-sampling, fourier-transform, streamname, regseqrat, real: no listed sibling supplies the joint packet containing a finite bandlimit row, sampling grid, reconstruction kernel, stream sample row, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{WhittakerShannonSampling}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WhittakerShannonSampling}^{\uparrow}$ packet is generated from finite bandlimit rows, displayed sampling-grid rows, reconstruction-kernel rows, StreamName samples, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WhittakerShannonSampling}^{\uparrow}}$, seed)

Scope closed: Definition 8180.1 and Definition 8180.2 seed the finite sampling bridge from Fourier and finite phase sampling rows to StreamName, RegSeqRat, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WhittakerShannonSampling}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite Fourier series, quotient equality of functions, unbounded sample choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bandlimit exposure, sampling-grid stability, reconstruction-kernel readback, StreamName and RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.WhittakerShannonSamplingUp.

Chapter 8181

A Concrete Naming Certificate for $\mathsf{BishopCompletionWitness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionWitness}^{\uparrow}$ packet records the finite witness data that a Real and Completion consumer must expose before treating a regular Cauchy route as completion-facing. It sits after the dyadic arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the diagonal classifier boundary of chapter 443, and the complete-real interface of chapter 4233. The packet names a displayed witness surface only; it is not $\mathsf{Real}^{\uparrow}$ completeness, quotient Cauchy equality, countable choice, or a selected completed real.

Definition 8181.1 (Bishop completion witness carrier)**.** *A* $\mathsf{BishopCompletionWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, Q, R, L, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic endpoint, radius, tail-comparison, and error-budget rows used by the witness.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing exactly the finite observations and requested precisions inspected by the completion read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those displayed windows and tolerances.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have appeared.*
5. *L is the accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *row that classifies the displayed regular-Cauchy handoff for a completion consumer.*
6. *B is the* $\mathsf{BishopCompleteReal}^{\uparrow}$ *boundary row, recording only the finite witness route by which the complete-real interface may inspect D, S, Q, R, L.*
7. *H records componentwise* hsame *transport for dyadic ledgers, stream windows, regular readbacks, Real seals, classifier rows, complete-real boundary rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the same finite witness route.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$, $\mathsf{BishopCompleteReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, R, L, B, H, C, P, N$. It has no coordinate for quotient equality of Cauchy streams, selected limits, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an unbounded completion operator.*

Theorem 8181.2 (Bishop completion witness NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionWitness}$ *carrier*

$$W = (D, S, Q, R, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionWitness}^{\uparrow}}$ *surface consists exactly of dyadic tolerance-ledger exposure, finite stream-window exposure, regular rational readback, terminal Real sealing, regular-Cauchy limit-classifier admission, Bishop complete-real boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read B as a completion-facing witness only after D, S, Q, R, L are displayed in the same finite packet.*

Proof. Project the carrier $W = (D, S, Q, R, L, B, H, C, P, N)$. The analytic rows D, S, Q, R, L, B list the tolerance ledger, inspected windows, regular readback, Real seal, classifier row, and complete-real boundary row.

The completion-facing row B is downstream of the classifier row L. By the public boundary of chapter 443 and the complete-real interface of chapter 4233, the consumer sees only the displayed dyadic, stream, regular-readback, and Real-seal route before the witness is offered.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the excluded principles are not carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 8181.3 (Bishop completion witness Real-completion handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopCompletionWitness}^{\uparrow}$ *packet must factor its read through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyLimitClassifier}^{\uparrow} \longmapsto \mathsf{BishopComplet}$$

Thus the exported completion witness is the finite route $D, S, Q, R, L, B, H, C, P, N$ over the displayed packet. It exports no quotient Cauchy class, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the dyadic and stream rows. The ledger D and window S determine the finite tolerance and observation data that can be inspected by a completion-facing read.

Next read the regular rational and Real rows. The row Q is the only regular readback in the carrier, and R is the only terminal Real seal available after that readback.

Now route through the classifier and complete-real boundary. The row L makes the displayed regular-Cauchy handoff visible to the limit-classifier surface, and B lets the Bishop complete-real interface consume only that classified finite witness.

Finally the rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Because no excluded endpoint is a row of W, the consumer cannot reread the witness as quotient Cauchy equality, selected completion, countable choice, host real equality, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{BishopCompletionWitness}^{\uparrow}$ consumer reaches $\mathsf{BishopCompleteReal}^{\uparrow}$ only after exposing $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, regularcauchylimitclassifier, bishop-complete-real: $\mathsf{BishopCompletionWitness}^{\uparrow}$ is not bijective to any listed sibling because it

jointly carries dyadic tolerance data, finite stream windows, regular rational readback, a Real seal, a limit-classifier row, a complete-real boundary row, transport, replay, provenance, and local naming in one finite witness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionWitness}\uparrow}$.

Constructive story: A $\mathsf{BishopCompletionWitness}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, a RegularCauchyLimitClassifier row, a BishopCompleteReal boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionWitness}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8181.1, Definition 8181.2, and Definition 8181.3 seed the finite witness route for the open Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionWitness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient Cauchy equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, stream-window exposure, RegSeqRat readback, Real seal ordering, classifier admission, complete-real boundary routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionWitnessUp.

Chapter 8182

A Concrete Naming Certificate for $\mathsf{StrongLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on almost-sure convergence of independent finite-variance averages, the $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ packet names the theorem-level route from finite tail control to a BEDC strong-law certificate. It consumes the weak-law finite-average route of chapter 4373, the finite-window strong-law handoff of chapter 7364, the Chebyshev tail-bound carrier of chapter 6592, the first Borel–Cantelli event-budget carrier of chapter 2357, the normalized probability source of chapter 184, the finite random-variable rows of chapter 185, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet records a finite certificate surface for the strong-law route; it does not import a host almost-sure quotient, countable-choice subsequence, ambient measurable completion, or external probability-limit principle.

Definition 8182.1 (Strong law large numbers carrier)**.** *A* $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, R, A, C, B, F, E, H, T, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the event classifier and total-event normalization.*
2. *R is the finite* $\mathsf{RandomVar}^{\uparrow}$ *family over P whose partial averages are inspected.*
3. *A is the partial-sum average row, recording the displayed finite average names and centers used by the law.*
4. *C is the* $\mathsf{ChebyshevInequality}^{\uparrow}$ *tail-budget table attached to the displayed averages.*
5. *B is the* $\mathsf{BorelCantelli}^{\uparrow}$ *finite-window schedule and repeated-hit budget for those tail rows.*
6. *F is the* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *handoff row, packaging the window schedule, Chebyshev table, and Borel–Cantelli budget as the finite strong-law precursor.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for positive tolerances, tail-budget bounds, and terminal average names.*
8. *H records componentwise* hsame *transport for the probability source, random-variable rows, averages, tail budgets, event-budget schedules, finite-window handoff, Real seals, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay of the strong-law route through $P, R, A, C, B, F, E, H, Q, N$.*

10. *Q is the* Pkg *provenance over* $\mathsf{StrongLawLargeNumbers}^{\uparrow}$, $\mathsf{StrongLawFiniteWindow}^{\uparrow}$, $\mathsf{WeakLawLargeNumbers}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{BorelCantelli}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{StrongLawLargeNumbers}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host almost-sure equality, selected subsequences, completed sigma-algebra quotients, central-limit normalization, or ambient real completeness.

Theorem 8182.2 (Strong law large numbers NameCert obligations)**.** *For every accepted* $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ *carrier $L = (P, R, A, C, B, F, E, H, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{StrongLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of carrier admission for the probability source row, finite random-variable rows, partial-sum average rows, Chebyshev tail budgets, Borel–Cantelli finite-window schedules,* $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may cite the strong law only after all rows $P, R, A, C, B, F, E, H, T, Q, N$ are visible in the carrier.*

Proof. Project the carrier. The rows P, R, A provide the probability source, displayed random variables, and finite partial-average names before any tail estimate or convergence-facing statement can be read.

The rows C, B, F, E then route the averages through Chebyshev tail budgets, Borel–Cantelli finite-window schedules, the finite-window strong-law handoff, and Real tolerance seals. This is the only convergence-facing analytic segment in the carrier.

Finally H, T, Q, N transport, replay, source, and name the same packet. Since no carrier coordinate stores an almost-sure quotient, selected subsequence, host measurable completion, or external probability-limit operator, the listed rows exhaust the local NameCert obligations. □

Theorem 8182.3 (Strong law finite-window handoff)**.** *Every accepted* $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ *read factors its convergence-facing request through the finite-window route*

$$P \longmapsto R \longmapsto A \longmapsto C \longmapsto B \longmapsto F \longmapsto E \longmapsto H, T, Q, N.$$

The handoff row F is therefore mandatory: it is the point where the weak-law finite-average surface and the $\mathsf{StrongLawFiniteWindow}^{\uparrow}$ *packet are joined before the final strong-law certificate is named.*

Proof. Begin with a strong-law consumer request. The carrier exposes only the probability source P, finite random-variable family R, and partial-average row A before any tail-control row is available.

Next the request reads C and B. The Chebyshev table supplies finite tail budgets, and the Borel–Cantelli row supplies the finite-window schedule for repeated-hit control.

The row F packages precisely the sibling handoff supplied by chapter 7364. Only after that row is displayed can the $\mathsf{Real}^{\uparrow}$ seal and structural rows E, H, T, Q, N present the local strong-law name. Thus every accepted read factors through the finite-window handoff rather than bypassing it with a host convergence theorem. □

Falsifiable prediction: A $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ consumer that exports almost-sure convergence without the ProbSpace, RandomVar, partial-average, Chebyshev, Borel–Cantelli, StrongLawFiniteWindow, Real, transport, replay, provenance, and local NameCert rows refutes this packet.

Independence witness: weak-law-large-numbers, strong-law-finite-window, chebyshev-inequality, borel-cantelli, probspace, randomvar, real: $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ is not bijective to any listed

sibling because it binds the full theorem-level handoff from finite averages and tail budgets to the local strong-law NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{StrongLawLargeNumbers}^{\uparrow}}$.

Constructive story: A $\mathsf{StrongLawLargeNumbers}^{\uparrow}$ packet is generated from a ProbSpace source row, finite RandomVar family, partial-sum average rows, Chebyshev tail budgets, BorelCantelli finite-window schedules, a StrongLawFiniteWindow handoff row, RealUp seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StrongLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8182.1, Definition 8182.2, and Definition 8182.3 seed the theorem-level strong-law route through probability, random-variable, average, tail-budget, event-schedule, finite-window handoff, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StrongLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host almost-sure quotient equality, countable-choice subsequence selection, completed sigma-algebra universes, central-limit normalization, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, tail-budget exactness, Borel–Cantelli schedule stability, finite-window handoff nonescape, Real-seal stability, transport, replay, provenance, and local naming for BEDC.Derived.StrongLawLargeNumbersUp.

Chapter 8183

A Concrete Naming Certificate for $\mathsf{ClassifierInformationBudget}^{\uparrow}$

The $\mathsf{ClassifierInformationBudget}^{\uparrow}$ packet realizes the classifier-level information distinction of chapter 9366 as finite BEDC data. It separates record growth from classifier shift by displaying before/after classifier-state rows, a SigRel handoff row, ledger-delta rows, transport, replay, provenance, and local naming. The packet is designed for discovery-classifier and TasteGate consumers that must know whether a route changes the classifier rather than merely extending a trace.

The budget packet consumes the morphism-level route discipline of chapter 590 and the boundary-exposure discipline of chapter 595 before interpreting a SigRel ledger delta as classifier-level information.

It also consumes the before/after disagreement route of chapter 6985 and feeds the positive-claim boundary of chapter 6931 together with the novelty ledger of chapter 6241.

Definition 8183.1 (Classifier information budget carrier)**.** *A* $\mathsf{ClassifierInformationBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S_0, S_1, G, D, L, H, C, P, N)$$

with the following displayed rows:

1. S_0 *is the source classifier-state row before the audited transition.*
2. S_1 *is the target classifier-state row after the audited transition.*
3. G *is the* SigRel *handoff row, recording which signature relation compares the two classifier states.*
4. D *is the ledger-delta row, separating external record growth from the classifier-shift bit read along* G*.*
5. L *is the local budget row, recording which classifier-level information budget is consumed by a downstream route.*
6. H *records componentwise* hsame *transport for the source state, target state,* SigRel *handoff, ledger delta, budget row, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the classifier-budget route through* $S_0, S_1, G, D, L, H, P, N$*.*
8. P *is the* Pkg *provenance over* $\mathsf{ClassifierInformationBudget}^{\uparrow}$*,* SigRel*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{ClassifierInformationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, G, D, L, H, C, P, N$. It has no coordinate for equating record growth with discovery, for treating a notation change as classifier shift, or for reading a positive discovery without the displayed ledger-delta row.

Theorem 8183.2 (Classifier information budget NameCert obligations)**.** *For every accepted* ClassifierInformationBu *carrier $I = (S_0, S_1, G, D, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClassifierInformationBudget}^{\uparrow}}$ *surface consists exactly of classifier-state admission for S_0, S_1,* SigRel *handoff admission for G, ledger-delta exposure for D, local budget exposure for L, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream consumer may read classifier-level information only after it displays these rows and keeps record growth distinct from classifier shift.*

Proof. Project the carrier. The two classifier-state rows and the SigRel handoff are the only before/after comparison data available to a consumer.

The ledger-delta row D separates record growth from classifier shift before the local budget row L can be consumed. Thus a longer record is not itself a classifier-level information witness, and a notation-only presentation change is not accepted as a shift.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the carrier has no hidden discovery bit or host classifier oracle, the listed rows exhaust the local NameCert obligations. □

Theorem 8183.3 (Classifier information budget SigRel handoff)**.** *Every accepted* $\mathsf{ClassifierInformationBudget}^{\uparrow}$ *read of classifier-level information factors through*

$$S_0 \longmapsto S_1 \longmapsto G \longmapsto D \longmapsto L \longmapsto H, C, P, N.$$

In particular, the SigRel *row G is mandatory before the ledger-delta row D may classify the transition as classifier-preserving, classifier-shifting, or budget-consuming.*

Proof. Start with a classifier-information request. The source and target classifier states S_0, S_1 are displayed before any comparison can be made.

The request next reads G. This row fixes the signature relation by which the two states are compared, so the ledger-delta row D cannot be interpreted independently of the handoff.

Finally the budget row L and the structural rows H, C, P, N present the accepted consumer surface. The route therefore records classifier-level information only through the displayed SigRel comparison and ledger delta. □

Falsifiable prediction: A $\mathsf{ClassifierInformationBudget}^{\uparrow}$ consumer that treats record growth alone as classifier-level information, or records classifier shift without a displayed SigRel handoff and ledger-delta row, refutes this packet.

Independence witness: classifier-state, sigrel, ledger-delta, transport, replay: $\mathsf{ClassifierInformationBudget}^{\uparrow}$ is not bijective to any single constituent because it binds before/after classifier rows, signature comparison, budget consumption, provenance, and local NameCert data in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierInformationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierInformationBudget}^{\uparrow}$ packet is generated from before/after classifier-state rows, a SigRel handoff row, ledger-delta rows separating record growth from classifier shift, a local information-budget row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassifierInformationBudget}^{\uparrow}}$, seed)

Scope closed: Definition 8183.1, Definition 8183.2, and Definition 8183.3 seed the classifier-level information budget through classifier-state, SigRel, ledger-delta, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassifierInformationBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a positive-discovery gate, a global classifier oracle, or any claim that record growth alone is classifier-level information.

Upgrade path: Obligation closure requires checked carrier admission, SigRel handoff exactness, ledger-delta separation, budget-row nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ClassifierInformationBudgetUp.

Chapter 8184

A Concrete Naming Certificate for $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$

Following Schauder's compact-map approximation method, the $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ packet records the finite BEDC carrier that sits between compact metric approximation data and the fixed-point endpoint consumers of chapter 1923, chapter 396, and the L10 compact-metric route. It uses the compact metric surface of chapter 129, the convex carrier of chapter 208, the continuous-map rows of chapter 130, and the uniform-modulus handoff of chapter 369. The packet names a finite approximation and residual ledger; it does not select a host fixed point, import open-cover compactness, or add a limiting point choice.

Definition 8184.1 (Schauder finite compact-map carrier)**.** *A* $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, Q, F, U, E, D, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the compact metric inspection surface and the finite cover windows used by downstream L10 compact-metric consumers.*
2. *Q is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the admitted compact convex region in which approximation points and residual tests are read.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row, carrying graph access, source membership, target membership, and displayed continuity data.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row that turns the local continuity information for F into a tolerance-facing approximation budget over K.*
5. *E is the finite* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *approximation row, listing the displayed net centers, cover witnesses, and projection choices available to the packet.*
6. *D is the displacement-budget row comparing each net approximation with its mapped image at the tolerance requested by U.*
7. *R is the fixed-point residual row, recording the finite residual comparison exported to* $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, *or another L10 compact-metric fixed-point consumer.*
8. *H stores componentwise* hsame *transport for K, Q, F, U, E, D, R and the local certificate row.*

9. *C stores displayed* Cont *replay from compact source, convex membership, continuous-map reads, uniform-modulus reads, finite net approximation, displacement budget, residual row, provenance, and local name.*

10. *P is the* Pkg *provenance over* $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchauderFiniteCompactMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient compactness theorem, a selected fixed point, quotient equality of approximants, a hidden limit point, or a detached self-map outside the displayed rows.

Theorem 8184.2 (Schauder finite compact-map NameCert obligations)**.** *For every accepted* SchauderFiniteCompac *carrier $S = (K, Q, F, U, E, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SchauderFiniteCompactMap}^{\uparrow}}$ *surface consists exactly of compact metric admission through K, convex carrier admission through Q, continuous self-map admission through F, uniform-modulus accountability through U, finite epsilon-net approximation through E, displacement-budget exposure through D, fixed-point residual exposure through R, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Lean4 (statement) `BEDC.Derived.SchauderFiniteCompactMapUp`

Proof. Project the carrier of Definition 8184.1. The compact source, convex region, continuous self-map, uniform modulus, finite epsilon-net, displacement budget, residual row, transport, replay, provenance, and local naming row are all displayed before a consumer can read the packet.

The approximation side is read in fixed order. The compact metric and convex rows admit the finite domain surface, the continuous-map row supplies the self-map, and the uniform-modulus row supplies the tolerance budget. The finite epsilon-net row then lists the centers and cover witnesses available to the displacement-budget row.

The residual row is downstream-facing only after those approximation rows have been replayed. Since the remaining rows H, C, P, N only transport, replay, source, and name the same finite packet, the listed rows exhaust the local NameCert surface and exclude the host principles named in the carrier. □

Theorem 8184.3 (Schauder finite compact-map residual route)**.** *Every fixed-point consumer of an accepted* $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto R.$$

The residual row R may be consumed by $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, *or another L10 compact-metric endpoint only after the compact source, convex carrier, continuous-map row, uniform-modulus row, finite epsilon-net approximation, displacement budget, componentwise transport, replay, provenance, and local naming rows have all been displayed.*

Proof. Begin with a fixed-point consumer. The continuation row C must expose K, Q, F, U, E, D before the residual row R is available.

The compact metric and convex rows bind the finite domain surface. The continuous-map row and uniform-modulus row bind the self-map and tolerance budget. The finite epsilon-net row and displacement-budget row are then the only approximation rows available to the consumer.

Thus R is a residual read from the displayed finite approximation packet, not a selected fixed point or a host compactness theorem. The structural rows H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. □

Falsifiable prediction: An accepted $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ read is refuted if a fixed-point residual row is consumed without displaying compact metric support, convex carrier membership, continuous self-map data, uniform-modulus accountability, finite epsilon-net approximation, displacement budget, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: compactmetric, convexset, continuousmap, uniformmodulus, banach-fixed-point, schauder-fixed-point: none alone is bijective with $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ because this packet jointly records finite epsilon-net approximation, displacement-budget comparison, residual export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderFiniteCompactMap}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ packet is generated from CompactMetric rows, ConvexSet carrier rows, ContinuousMap self-map rows, UniformModulus budget rows, finite epsilon-net approximation rows, displacement-budget rows, fixed-point residual rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderFiniteCompactMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8184.1, Definition 8184.2, and Definition 8184.3 seed the finite compact-map approximation packet over compact metric, convex, continuous-map, uniform-modulus, finite-net, displacement-budget, residual, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderFiniteCompactMap}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SchauderFiniteCompactMapUp`

Bridge status: none

Not claimed: This chapter does not close a host fixed-point selector, open-cover compactness, quotient equality of approximants, hidden limiting points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric exposure, convex carrier exposure, continuous-map replay, uniform-modulus accountability, finite epsilon-net approximation, displacement-budget exposure, residual export, transport stability, replay stability, provenance, and local naming for BEDC.Derived.SchauderFiniteCompactMapUp.

Chapter 8185

A Concrete Naming Certificate for $\mathsf{DyadicNestedIntervalCode}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicNestedIntervalCode}^{\uparrow}$ packet records the finite nested-interval route by which dyadic brackets are read as regular rational data before any terminal real-name seal is consumed. It sits over the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names only displayed finite rows; it is not a quotient completion, a countable-choice selector, a trichotomy principle, a host equality of streams, or an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8185.1 (Dyadic nested interval code carrier)**.** *A* $\mathsf{DyadicNestedIntervalCode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite family of nested dyadic interval rows, each row naming lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints and the containment relation to the next inspected row.*
2. *W is the width-shrink ledger, recording the dyadic denominator budget, the exact positive width row, and the displayed shrink comparison for every inspected stage.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the finite stage numbers and endpoint observations that a consumer may read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed midpoint or endpoint-selection data attached to the same windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after the rows I, W, S, R have been replayed.*
6. *H records componentwise* hsame *transport for the interval rows, width rows, stream windows, regular-rational readback, real seal, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from a finite nested-interval request through I, W, S, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicNestedIntervalCode}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalCode}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, W, S, R, E, H, C, P, N$. It has no coordinate for quotient streams, countable choice, trichotomy, host equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8185.2 (Dyadic nested interval code NameCert obligations)**.** *For every accepted* DyadicNestedIntervalCod *carrier*

$$D = (I, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalCode}^{\uparrow}}$ *surface consists of carrier admission, nested containment stability, width-shrink exactness, classifier transport, ledger exactness, and nonescape from quotient streams, countable choice, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 8185.1. The accepted object is exactly the finite packet $D = (I, W, S, R, E, H, C, P, N)$, so every visible row is one of the displayed interval, width, window, readback, seal, transport, replay, provenance, or naming coordinates.

Nested containment stability is read from I, while width-shrink exactness is read from W. Both rows are dyadic rows, so their endpoint and tolerance data are consumed through the $\mathsf{DyadicRatCore}^{\uparrow}$ surface named by chapter 291.

Classifier transport is componentwise through H and C. The same finite stage windows are exposed by S, and the regular rational readback R is attached to those windows before any consumer reaches the real seal E.

Ledger exactness and nonescape follow by inspection of the carrier. No coordinate stores quotient streams, a choice function, trichotomy, host equality of real names, or an ambient completeness theorem, so no accepted $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalCode}^{\uparrow}}$ packet can export those objects. □

Theorem 8185.3 (Dyadic nested interval code regular-Cauchy handoff)**.** *Every accepted* DyadicNestedIntervalCode *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. In particular, the route to E factors through*

$$I, W, S, R \longmapsto E,$$

with I, W supplying dyadic containment and shrink data, S supplying the inspected windows, and R supplying the regular-Cauchy rational handoff.

Proof. Start from the carrier of Definition 8185.1. The real seal coordinate E is listed after the interval rows I, width ledger W, stream windows S, and regular rational readback R.

The NameCert obligations of Definition 8185.2 force each of those rows to be exposed before any consumer reads E. The dyadic rows supply the tolerance and shrink data, the stream row supplies the finite observation windows, and the regular-rational row supplies the Cauchy readback attached to those same windows.

Thus the handoff to $\mathsf{Real}^{\uparrow}$ is routed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ data only. Since the packet has no quotient, choice, trichotomy, host-equality, or ambient-completeness coordinate, the terminal seal cannot be reached through any other route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalCode}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicNestedIntervalCode}^{\uparrow}$ packet is generated from nested dyadic interval rows, width-shrink rows, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicNestedIntervalCode}^{\uparrow}}$, seed)

Scope closed: Definition 8185.1, Definition 8185.2, and Definition 8185.3 seed the finite dyadic nested-interval route from tolerance rows and stream windows to regular-Cauchy readback and a Real-facing seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicNestedIntervalCode}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, countable choice, trichotomy, host equality of real names, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested containment stability, width-shrink exactness, classifier transport, ledger exactness, and regular-Cauchy handoff for BEDC.Derived.DyadicNestedIntervalCodeUp.

Chapter 8186

A Concrete Naming Certificate for $\mathsf{RegularCauchyExponential}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RegularCauchyExponential}^{\uparrow}$ packet records the exponential-series constructor on regular Cauchy names as finite BEDC data. It refines the displayed series route of chapter 7993 through the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real algebra and elementary-function consumers a non-quotient dependency anchor; it does not import host exponential functions, quotient streams, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8186.1 (RegularCauchyExponential carrier)**.** *A* $\mathsf{RegularCauchyExponential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, W, D, A, F, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular Cauchy input.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect source values, powers, factorial denominators, partial sums, and tail budgets.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested precision window.*
4. *A is the coefficient row for the rational factorial denominators and dyadic comparison budgets of the exponential series.*
5. *F is the finite partial-sum row for the displayed exponential series over the inspected regular Cauchy windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the finite partial sums and tail ledger.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the regular Cauchy exponential name.*
8. *H records componentwise* hsame *transport for source, windows, dyadic budgets, coefficients, partial sums, readback, Real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay along the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyExponential}^{\uparrow}$, $\mathsf{CauchyExponential}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyExponential}^{\uparrow}}$ *row.*
The carrier exports only this finite regular Cauchy series surface. It stores no host exponential function, no quotient stream equality, no chosen infinite limit, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 8186.2 (RegularCauchyExponential NameCert obligations)**.** *For every accepted* RegularCauchyExponent *carrier $E = (X, W, D, A, F, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyExponential}^{\uparrow}}$ *surface consists exactly of carrier admission, regular Cauchy source exposure, finite StreamName window exposure, dyadic tolerance accounting, factorial coefficient accounting, finite partial-sum display, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming. No local obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8186.1. The available analytic data are the source row X, the window schedule W, the dyadic budget D, the coefficient row A, the finite partial-sum row F, the readback row R, and the seal row S.

The finite series rows are ordered by the inspected windows. A consumer first reads the regular Cauchy input through X, W, D, then the factorial coefficients and partial sums through A, F, and only then the regular readback and Real seal through R, S.

The structural rows H, C, P, N preserve and name exactly that route. Since no coordinate contains a host exponential, quotient stream equality, selected infinite limit, countable-choice schedule, or ambient completeness principle, none of those objects belongs to the local NameCert surface. □

Theorem 8186.3 (RegularCauchyExponential series window)**.** *Every series-facing consumer of an accepted* $\mathsf{RegularCauchyExponential}^{\uparrow}$ *carrier must read the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto F \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

before any terminal $\mathsf{Real}^{\uparrow}$ *seal is exported. The finite row F contains only the partial sums, factorial denominator ledger, and tail estimates visible in the requested window.*

Proof. Start with the source and window rows X, W. They determine which regular Cauchy values and which partial-sum stages are available for the request.

Next read the dyadic tolerance ledger D and coefficient row A. These rows bound the inspected factorial denominators and tail estimates before the finite partial-sum row F is exposed.

Finally the readback row R repacks the displayed partial sums as regular rational data. The terminal seal can only consume that readback, so the consumer receives a finite windowed series route rather than a hidden host exponential or selected infinite limit. □

Theorem 8186.4 (RegularCauchyExponential Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyExponential}^{\uparrow}$ *carrier must pass through the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row X, finite* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, coefficient and partial-sum rows A, F, regular readback R, and structural rows H, C, P, N before the seal row S is available. The export is a displayed regular Cauchy exponential name, not ambient Real completeness, quotient stream equality, a selected limit, or a host exponential function.*

Proof. Use Definition 8186.3. It forces every series read through the finite window, dyadic tolerance ledger, coefficient row, partial sums, and regular rational readback before a Real-facing row can be consumed.

The seal row S is downstream from R. Thus a $\mathsf{Real}^{\uparrow}$ consumer cannot inspect a terminal value until the regular readback has already exposed the finite series route.

Transport, replay, provenance, and local naming then preserve the same packet through H, C, P, N. These rows do not add ambient Real completeness, quotient stream equality, selected limits, or host exponential functions, so the exported seal is exactly the displayed regular Cauchy exponential name. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyExponential}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyExponential}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8186.1, Definition 8186.2, Definition 8186.3, and Definition 8186.4 seed the finite regular Cauchy exponential route over CauchyExponential, RegSeqRat, StreamName, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyExponential}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, countable choice, host exponential functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient accounting, finite partial-sum display, dyadic tail control, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyExponentia

Constructive story: A $\mathsf{RegularCauchyExponential}^{\uparrow}$ packet is generated from a regular Cauchy source row, finite StreamName windows, a DyadicRatCore tolerance ledger, factorial coefficient data, finite partial sums, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8187

A Concrete Naming Certificate for $\mathsf{LocatednessPrinciple}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on located predicates, the $\mathsf{LocatednessPrinciple}^{\uparrow}$ packet records the finite comparison surface by which a real or completion consumer may ask for a displayed located comparison without importing an omniscience principle. It sits beside the Ishihara trick boundary of chapter 1665, the lesser limited omniscience boundary of chapter 1977, the Markov principle boundary of chapter 1666, the located real packet of **??**, and the terminal real-name seal of chapter 38. The packet names a local locatedness service only: it does not claim LLPO, LPO, Markov principle, trichotomy, selected witnesses, quotient reals, or ambient completeness.

Definition 8187.1 (Locatedness principle carrier)**.** *A* $\mathsf{LocatednessPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, A_-, A_+, I, O, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational comparison request row for the bounded real or completion-facing object being inspected.*
2. *A_- and A_+ are the two located approximation rows that bracket the request by negative and positive rational margins.*
3. *I is the* $\mathsf{IshiharaTrick}^{\uparrow}$ *boundary row used only to record finite inspection of a branch-shaped comparison.*
4. *O is the* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ *boundary row marking where two-alternative inspection is refused as a global rule.*
5. *B is the* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *row recording that negative evidence is named as boundary data, not accepted as a principle.*
6. *R is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row through which endpoint or cut-shaped locatedness is read.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the request, approximation, and boundary rows have been displayed.*
8. *H stores componentwise* hsame *transport for the request, approximation, boundary, located-real, real-seal, provenance, and naming rows.*

9. *C stores displayed* Cont *replay from the rational comparison request through the approximation rows, boundary rows, located-real read, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatednessPrinciple}^{\uparrow}$, $\mathsf{IshiharaTrick}^{\uparrow}$, $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatednessPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for LLPO, LPO, Markov principle, trichotomy, selected witnesses, quotient reals, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatednessPrincipleUp.LocatednessPrincipleTasteGate single carrier alignment`

Theorem 8187.2 (Locatedness principle NameCert obligations)**.** *For every accepted* $\mathsf{LocatednessPrinciple}^{\uparrow}$ *carrier*

$$L = (Q, A_-, A_+, I, O, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatednessPrinciple}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed rational comparison request, two approximation rows, boundary rows, located-real read, Real seal, transport, replay, provenance, and local naming;*
2. *approximation locatedness, requiring A_- and A_+ to be read before any* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives a comparison answer;*
3. *boundary discipline, requiring every branch-shaped read to pass through I, O, B as named inspection boundaries rather than as imported decision principles;*
4. *real-boundary handoff, requiring the located-real row R and terminal seal E to appear only after the comparison and boundary rows have been displayed;*
5. *nonescape for LLPO, LPO, Markov principle, trichotomy, selected witnesses, quotient reals, and ambient completeness.*

Proof. Carrier admission is projection from Definition 8187.1. The packet displays exactly one request row, two approximation rows, three boundary rows, one located-real read, one Real seal, and four structural rows.

The approximation rows provide the located comparison service. A consumer reads Q, then A_- and A_+, before any located-real or Real-facing row is available.

The branch-sensitive rows are only boundary data. The Ishihara, lesser-limited-omniscience, and Markov-boundary rows record finite inspection and refusal surfaces, so they cannot be consumed as global LLPO, LPO, Markov principle, or trichotomy.

Finally R, E, H, C, P, N transport, replay, source, and name the displayed route. Since the carrier has no coordinate for selected witnesses, quotient reals, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 8187.3 (Locatedness principle real boundary)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatednessPrinciple}^{\uparrow}$ *carrier factors through the finite route*

$$Q \longmapsto (A_-, A_+) \longmapsto (I, O, B) \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

Thus the consumer may receive a displayed located comparison, but it receives no LLPO, LPO, Markov principle, trichotomy, selected witness, quotient-real equality, or ambient completeness principle.

Proof. Project the accepted carrier. The request row Q is the only entry point for a comparison query.

Read the approximation rows next. The rows A_- and A_+ provide the finite located margins before any endpoint, cut, or Real-seal read can occur.

The boundary rows are downstream of those approximations. They record where Ishihara-style branch inspection, lesser-limited-omniscience refusal, and Markov-sensitive negative evidence are named, without adding a decision coordinate.

The located-real and Real rows are then R and E. Transport, replay, provenance, and naming through H, C, P, N preserve the same route, so no selected witness, quotient-real equality, trichotomy, or ambient completeness row can be exported by the packet. □

Falsifiable prediction: If the boundary rows I, O, B are removed, a consumer could misread a finite located comparison as a global decision principle rather than as a displayed local route.

Independence witness: ishihara-trick, lesser-limited-omniscience, markov-principle-boundary, locatedreal, real: not bijective because the locatedness packet stores the request, the two approximation rows, three boundary refusals, a located-real read, and a terminal Real seal as one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatednessPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatednessPrinciple}^{\uparrow}$ packet is generated from a rational comparison request, two located approximation rows, Ishihara, lesser-limited-omniscience, and Markov boundary rows, a LocatedReal read, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatednessPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8187.1, Definition 8187.2, and Definition 8187.3 seed the finite locatedness route over comparison requests, approximation rows, boundary refusals, located-real reads, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatednessPrinciple}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatednessPrincipleUp.LocatednessPrincipleTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim LLPO, LPO, Markov principle, trichotomy, selected witnesses, quotient reals, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximation locatedness, boundary discipline, located-real handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatednessPrincipleUp.

Chapter 8188

A Concrete Naming Certificate for $\mathsf{KolmogorovReflection}^{\uparrow}$

Following Kolmogorov's separation axiom and the Bourbaki–Kelley presentation of topological identification by indistinguishability, the $\mathsf{KolmogorovReflection}^{\uparrow}$ packet records the T_0 separation-reflection route as finite BEDC NameCert data. It sits beside the topological carrier of chapter 88, the separated metric reflector of chapter 1790, and the metric-space carrier of chapter 4230, while keeping quotient topology and host equality outside the certificate.

Definition 8188.1 (Kolmogorov reflection carrier)**.** *A* $\mathsf{KolmogorovReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, T, M, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source point-classifier row, carrying only the observable names whose topological indistinguishability is inspected.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-observation row, recording finite open-neighbourhood tests and their BEDC classifier readings.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *optional metric neighbourhood row used when a metric presentation supplies the same observation boundary.*
4. *S is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *comparison row, available only as a neighbouring separated-reflection route and not as a quotient object.*
5. *R is the Kolmogorov-reflected classifier row, identifying exactly the finite observation profile that distinguishes points up to T_0 separation.*
6. *H stores componentwise* hsame *transport for the source, topological, metric, separated-reflection, reflected-classifier, replay, provenance, and naming rows.*
7. *C stores displayed* Cont *replay from a finite observation query through X, T, M, S, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{KolmogorovReflection}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{SeparatedMetricReflection}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovReflection}^{\uparrow}}$ *row.*

The carrier exposes a T_0 reflection classifier only as finite observation data. It has no quotient-topology coordinate, no host equality coordinate, no sober-space reconstruction, no choice principle, and no completeness or separated-metric theorem beyond the displayed sibling route.

Lean4 ✓ `BEDC.Derived.KolmogorovReflectionUp.KolmogorovReflectionTasteGate single carrier alignment`

Theorem 8188.2 (Kolmogorov reflection NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovReflection}^{\uparrow}$ *carrier* $K = (X, T, M, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KolmogorovReflection}^{\uparrow}}$ *surface consists exactly of carrier exposure, finite* $\mathsf{Topology}^{\uparrow}$ *observation, optional* $\mathsf{Met}^{\uparrow}$ *neighbourhood reading, neighbouring* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *comparison, reflected* T_0 *classifier output, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read the observation profile and the reflected classifier row, but it may not infer quotient topology or host equality.*

Proof. Project the displayed carrier. The rows X, T, M, S, R are the only point, topology, metric-neighbourhood, separated-reflection, and reflected-classifier rows available to a consumer.

The T_0 observation route is finite. A consumer reads open-neighbourhood evidence through T, may compare a metric presentation through M, and may inspect the neighbouring separated-reflection route through S before reaching the classifier output R.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since quotient topology and host equality are not fields of the carrier, no NameCert obligation can export either object. □

Theorem 8188.3 (Kolmogorov reflection separation route)**.** *Every accepted* $\mathsf{KolmogorovReflection}^{\uparrow}$ *packet factors any* T_0 *separation read through the displayed route*

$$\mathsf{Topology}^{\uparrow} \to \mathsf{Met}^{\uparrow} \to \mathsf{SeparatedMetricReflection}^{\uparrow} \to \mathsf{BHist} \to \mathsf{hsame} \to \mathsf{Cont} \to \mathsf{Pkg} \to \mathsf{NameCert}.$$

The route may inspect finite open observations, metric neighbourhood rows, separated-reflection classifier comparison, transport, replay, provenance, and local naming. It does not construct a quotient topology, identify host points, or claim a separated metric reflection as an ambient equality theorem.

Proof. Begin with the carrier projection of Definition 8188.1. The topological observation row T, metric neighbourhood row M, separated-reflection row S, and reflected classifier row R are ordinary finite rows in one BHist packet.

Apply the NameCert obligations of Definition 8188.2. They force every local read to preserve the observation, comparison, classifier, transport, replay, provenance, and naming rows before the reflected classifier is consumed.

The separated metric reflector is used only as a sibling dependency. Its row S supplies comparison data, while the local packet still refuses quotient topology, host equality, and any ambient equality theorem. Thus the route remains a finite T_0 reflection certificate. □

Theorem 8188.4 (Kolmogorov reflection no-quotient boundary)**.** *Let* $K = (X, T, M, S, R, H, C, P, N)$ *be an accepted* $\mathsf{KolmogorovReflection}^{\uparrow}$ *packet. Any consumer that reads the reflected classifier row* R *must still display the original observation rows* X, T, M, S *and the structural rows* H, C, P, N*. Consequently the packet supplies no quotient topology, no host equality of points, no selected representatives, and no global separation theorem beyond the finite classifier route.*

Proof. Project the accepted packet. The reflected classifier row R is not independent of the observation rows; it is listed after X, T, M, S in the carrier and is transported and replayed only through H and C.

Use the separation route theorem. It keeps any T_0 read on the displayed route through topology, optional metric neighbourhoods, neighbouring separated reflection, history, transport, replay, provenance, and local naming.

The provenance row P names no quotient topology, host equality, selected representative, or global separation theorem. Therefore a consumer of R receives only the finite classifier route and cannot repack the packet as any of those stronger objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovReflection}^{\uparrow}$ packet is generated from a source point-classifier row, finite topological observations, optional metric neighbourhood data, a separated-reflection comparison row, a reflected T_0 classifier, componentwise transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8188.1, Definition 8188.2, Definition 8188.3, and Definition 8188.4 seed the finite T_0 reflection route over topological, metric, separated-reflection, history, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient topology, host equality of points, selected representatives, sober-space reconstruction, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite topological observation exactness, separated-reflection comparison, no-quotient nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovReflectionUp.

Chapter 8189

A Concrete Naming Certificate for $\mathsf{MontelNormalFamily}^{\uparrow}$

Following Montel's theorem in classical complex analysis, the $\mathsf{MontelNormalFamily}^{\uparrow}$ packet records the finite-window handoff from local boundedness of holomorphic families to a normal-family read over displayed compact complex windows. It consumes the normal-family carrier of chapter 2186, the finite Montel route of chapter 3693, the holomorphic source rows of chapter 68, the complex coordinate rows of chapter 39, and the compact-window boundary of chapter 60. The packet names only the displayed BEDC rows: compact windows, local boundedness, holomorphic readback, complex coordinates, finite stream windows, regular rational tolerances, transport, replay, provenance, and local naming.

Definition 8189.1 (Montel normal-family carrier)**.** *A* $\mathsf{MontelNormalFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (N, T, K, L, F, Z, S, R, H, C, P, A)$$

with the following visible rows:

1. *N is the* $\mathsf{NormalFamily}^{\uparrow}$ *row, exposing the family ledger, compact source data, equicontinuity route, pointwise image rows, and finite probe surface.*
2. *T is the* $\mathsf{MontelTheorem}^{\uparrow}$ *handoff row, restricted to the same family ledger and compact complex window.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-window row for the displayed compact subset of the complex domain.*
4. *L is the local boundedness ledger for the family on K.*
5. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *readback row for the displayed family entries on the complex window.*
6. *Z is the* $\mathsf{Complex}^{\uparrow}$ *coordinate row for source points, target values, and the visible complex bounds used by L and F.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that lists the family members, compact-window probes, and tolerance requests actually inspected.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tolerance and readback row for the finite windows named by S.*
9. *H records componentwise* hsame *transport of N, T, K, L, F, Z, S, R.*

10. *C records displayed* Cont *replay from compact-window request through local boundedness, holomorphic readback, normal-family admission, and Montel handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{MontelNormalFamily}^{\uparrow}$, $\mathsf{NormalFamily}^{\uparrow}$, $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *A is the local* $\mathsf{NameCert}_{\mathsf{MontelNormalFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transport, replay, provenance, and local naming data.

Theorem 8189.2 (Montel normal-family NameCert obligations)**.** *For every accepted* $\mathsf{MontelNormalFamily}^{\uparrow}$ *carrier* $M = (N, T, K, L, F, Z, S, R, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{MontelNormalFamily}^{\uparrow}}$ *surface consists exactly of:*

1. *normal-family admission for N over the same displayed family ledger;*
2. *Montel handoff admission for T over the same compact complex window;*
3. *compact finite-window admission for K;*
4. *local boundedness admission for L over K;*
5. *holomorphic and complex readback through F and Z;*
6. *finite stream-window and regular-rational tolerance admission through S and R;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, A.*

No obligation introduces countable choice, arbitrary subsequence selection, open-cover compactness, ambient complex completeness, quotient complex equality, or a host compact-open function space.

Proof. Project the carrier in Definition 8189.1. The normal-family row N and Montel row T are the only family-compactness handoffs, and both are tied to the same compact window K and family window schedule S.

The analytic input is finite. Local boundedness is the row L, holomorphic readback is the row F, complex coordinate exposure is the row Z, and regular rational tolerance data is the row R. Each of these rows is named before the packet can be consumed as a Montel normal-family handoff.

Finally, H, C, P, A preserve exactly the displayed row inventory under transport, replay, provenance, and local naming. Since the packet has no coordinate for a choice-selected subsequence, open-cover compactness proof, ambient complex completeness theorem, quotient equality principle, or compact-open host function space, the local certificate cannot export any of those objects. □

Theorem 8189.3 (Montel compact-window handoff)**.** *Every consumer reading an accepted* $\mathsf{MontelNormalFamily}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{Compact}^{\uparrow} \longmapsto \textit{local boundedness} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{NormalFamily}^{\uparrow} \longmapsto \mathsf{MontelTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{MontelNorm}}$$

The stream schedule and regular rational rows S, R bound this route to the displayed finite compact windows, while H, C, P, A provide transport, replay, provenance, and local naming.

Proof. Begin with the compact window K. It is the only source of compact-domain information in the carrier, and the stream schedule S records the probes and tolerance requests that may be inspected.

Next read the local boundedness row L, the holomorphic row F, and the complex coordinate row Z. These rows supply the analytic input over the same compact window, with R carrying the regular rational tolerance data needed by the finite-window readback.

Now consume N and T. The normal-family row receives the family and compact-window data, while the Montel row records the finite handoff from local boundedness and holomorphic readback to the normal-family conclusion. The closing rows H, C, P, A preserve this route under transport, continuation replay, provenance, and local naming. □

Theorem 8189.4 (Montel normal-family public non-escape)**.** *An accepted* $\mathsf{MontelNormalFamily}^{\uparrow}$ *carrier exports no countable-choice principle, arbitrary subsequence selector, open-cover compactness theorem, ambient* $\mathsf{Complex}^{\uparrow}$ *completeness assertion, quotient complex equality, or host compact-open function-space object. Every public read is confined to the displayed rows* $N, T, K, L, F, Z, S, R, H, C, P, A$.

Proof. Inspect the carrier coordinates. The family-facing routes are N and T, the compact and analytic rows are K, L, F, Z, the finite-window rows are S, R, and the structural rows are H, C, P, A.

None of these coordinates names countable choice, an arbitrary subsequence selector, open-cover compactness, ambient complex completeness, quotient equality, or a compact-open host function space. Componentwise hsame transport and Cont replay can only preserve displayed rows, while Pkg provenance and local NameCert naming record the same row set. Therefore no public read can escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MontelNormalFamily}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MontelNormalFamily}^{\uparrow}$ packet is generated from NormalFamily and MontelTheorem handoff rows, a Compact finite-window row, local boundedness, Holomorphic and Complex readback, StreamName finite windows, RegSeqRat tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MontelNormalFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8189.1, Definition 8189.2, Definition 8189.3, and Definition 8189.4 seed the finite-window Montel normal-family handoff over existing complex-analysis and compact-window rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MontelNormalFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, countable choice, arbitrary subsequence selection, open-cover compactness, ambient complex completeness, quotient complex equality, or a host compact-open function space.

Upgrade path: Obligation closure requires checked carrier admission, local boundedness confinement, compact-window handoff factorization, holomorphic and complex readback compatibility, finite-window tolerance replay, provenance exhaustion, and local naming for BEDC.Derived.MontelNormalFamilyUp.

Chapter 8190

A Concrete Naming Certificate for $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$

Bishop and Bridges constructive analysis treats an oscillation modulus as finite located data: a tolerance request is answered by an inspected window and a displayed rational bound, not by a selected limiting object. The $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ packet records this L10 corridor as a NameCert carrier over the pointwise oscillation surface of chapter 1839, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. It refines the neighboring located oscillation certificate by naming the modulus row consumed before Real or Completion consumers turn oscillation bounds into Cauchy-facing evidence.

Definition 8190.1 (Bishop located oscillation-modulus carrier)**.** *A* $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{locosc}} = (O, D, S, Q, R, W, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{RealOscillation}^{\uparrow}$ *source row whose inspected values and local difference bounds are read from the existing real-oscillation packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested oscillation radius, endpoint slack, and comparison budget.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the sampled source windows used for the request.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic budget.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after O, D, S, Q have been displayed.*
6. *W is the located-window ledger recording which finite source window answers the request and which rational bound it returns.*
7. *T is the tolerance-to-window modulus row: for every displayed dyadic tolerance in D, it points to one inspected row of W and to the corresponding regular readback through Q and R.*
8. *H records componentwise* hsame *transport for O, D, S, Q, R, W, T and the local naming data.*
9. *C records displayed* Cont *replay from a tolerance request through the dyadic ledger, window row, regular readback, Real seal, located-window ledger, and modulus row.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets whose modulus row T is downstream of the displayed oscillation row O and the same dyadic-window-readback route D, S, Q, R, W. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, trichotomy, countable choice, host supremum, or a detached modulus oracle.*

Theorem 8190.2 (Bishop located oscillation-modulus NameCert obligations)**.** *For an accepted* $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ *carrier $M_{\mathrm{locosc}} = (O, D, S, Q, R, W, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocated}}$ *surface consists exactly of:*

1. *carrier admission for the real-oscillation row, dyadic tolerance ledger, stream-window row, regular rational readback, Real seal, located-window ledger, tolerance-to-window modulus row, transport, replay, provenance, and local naming rows;*
2. *oscillation-source exposure, requiring O to be read before any modulus row answers a tolerance request;*
3. *dyadic-window coverage, requiring every accepted tolerance in D to name one displayed finite window in S and one regular readback in Q before the terminal seal R is consumed;*
4. *located-modulus exactness, requiring T to point only to rows already present in W and to export no answer outside the displayed dyadic-window route;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name no ambient completeness theorem, quotient stream equality, selected limit, trichotomy, countable-choice schedule, host supremum, or detached oracle.*

Proof. Project the accepted carrier. The source row O, tolerance ledger D, window row S, regular readback Q, Real seal R, located ledger W, and modulus row T are all displayed before the structural rows H, C, P, N can expose the packet.

The source-exposure and dyadic-window clauses are componentwise projections. A tolerance request is first read in D, then replayed through S and Q, and only then reaches the terminal seal R.

Located-modulus exactness follows from the shape of T: every answer points back into W and to the same D, S, Q, R route. Since the carrier stores no separate oracle row, T cannot answer from outside the packet.

Finally, H, C, P, N preserve and name only these displayed coordinates. None contains ambient real completeness, quotient stream equality, selected limits, trichotomy, countable choice, host suprema, or detached modulus data, so the NameCert surface has no exit to those objects. □

Theorem 8190.3 (Bishop located oscillation-modulus window route)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{RealOscillation}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ W,\ T,\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Thus oscillation-to-Cauchy or limit-facing use receives a displayed located modulus over finite windows, not ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, trichotomy, countable choice, a host supremum, or a detached modulus oracle.*

Proof. Start with the accepted carrier of Definition 8190.1. Its only source of oscillation information is O, and its only tolerance budget is the dyadic row D.

The window route is forced by the readback rows. The stream row S names the inspected finite windows, Q gives regular rational readback for those windows, and R seals the Real-facing endpoint only after those rows are visible.

The modulus row T then reads through the located-window ledger W and cannot bypass it. Applying the NameCert obligations of Definition 8190.2 shows that every consumer-visible route is exhausted by $O, D, S, Q, R, W, T, H, C, P, N$.

The excluded objects are absent from those coordinates. Therefore an oscillation-to-Cauchy or limit-facing consumer receives exactly the displayed finite-window modulus route. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ consumer read factors through RealOscillation, DyadicRatCore, StreamName, RegSeqRat, RealUp, located-window, tolerance-to-window modulus, hsame, Cont, Pkg, and NameCert rows.

Independence witness: real-oscillation, dyadicratcore, streamname, regseqrat, real: BishopLocatedOscillationMo is not bijective to any listed sibling because it jointly carries an oscillation source row, dyadic tolerance budget, finite observation windows, regular rational readback, Real seal, located-window ledger, tolerance-to-window modulus row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedOscillationModulus}^{\uparrow}$ packet is generated from a RealOscillation source row, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, a located-window ledger, a tolerance-to-window modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8190.1, Definition 8190.2, and Definition 8190.3 seed the finite located oscillation-modulus route over RealOscillation, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, trichotomy, countable choice, host suprema, detached modulus oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, oscillation-source exposure, dyadic-window coverage, located-modulus exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedOscillationMo

Chapter 8191

A Concrete Naming Certificate for MetaCICResidualChoiceBudget$^{\uparrow}$

The MetaCICResidualChoiceBudget$^{\uparrow}$ packet realizes the bounded residual-choice frontier of section 9386.3 as finite confluence-facing NameCert data. It sits beside the residual-substitution boundary of chapter 2181, the residual checker of chapter 3827, the residual-join budget of chapter 7378, the residual-diamond budget of chapter 7407, and the parallel-diamond frontier of chapter 4238. The packet records a finite choice boundary for residual pairs without claiming full residual substitution, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Falsifiable prediction: Every accepted MetaCICResidualChoiceBudget$^{\uparrow}$ confluence read exposes a bounded choice-boundary row before any local-join candidate can be transported to a residual-diamond or parallel-diamond consumer.

Independence witness: metacic-residual-join-budget, metacic-residual-diamond-budget, metacic-parallel-diamond-frontier, metacic-confluence-residual-checker, metacic-residual-substitution-boundary: none is bijective to MetaCICResidualChoiceBudget$^{\uparrow}$ because this packet jointly carries typed windows, residual pairs, explicit choice-boundary rows, bounded local joins, obstruction retention, transport, replay, provenance, and local naming.

Definition 8191.1 (MetaCIC residual-choice budget carrier)**.** *A* MetaCICResidualChoiceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, R, Q, J, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, recording the closed term, local typing window, and bounded checker context in which a residual pair is inspected.*
2. *R is the residual-pair row, recording the two residual descendants and their branch-local exposure.*
3. *Q is the choice-boundary row, listing the finite branch choices still retained by the budget.*
4. *J is the bounded local-join row, recording candidate joins available only after T, R, Q have been displayed.*

5. *O is the obstruction-retention row, naming residual-substitution discharge, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, and kernel consistency as endpoints not supplied by the packet.*
6. *H stores componentwise* hsame *transport for the typed-window, residual-pair, choice-boundary, bounded-join, obstruction, replay, provenance, and naming rows.*
7. *C stores displayed* Cont *replay from a confluence query through T, R, Q, J, O, H, P, N.*
8. *P is the* Pkg *provenance over* MetaCICResidualChoiceBudget$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceResidualChecker$^{\uparrow}$, MetaCICResidualJoinBudget$^{\uparrow}$, MetaCICResidualDiamondBudget$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, Q, J, O, H, C, P, N$. It has no coordinate for full residual substitution, premise-free Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, global conversion decidability, or kernel consistency.

Theorem 8191.2 (MetaCIC residual-choice budget obligations)**.** *For every accepted* MetaCICResidualChoiceBudget *carrier $B = (T, R, Q, J, O, H, C, P, N)$, the local* NameCert$_{\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, residual-pair exposure through R, bounded choice-boundary retention through Q, bounded local-join exposure through J, obstruction retention through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier. The confluence-facing rows are T, R, Q, J, O, and the structural rows are H, C, P, N.

The choice boundary is retained as data. Row Q is not discharged by the carrier; it records the finite branch choices that a local-join consumer must keep visible before reading J.

The obstruction row then keeps the stronger endpoints outside the packet. Transport, replay, provenance, and local naming preserve the same bounded budget rather than turning it into residual-substitution discharge or unrestricted confluence. □

Theorem 8191.3 (MetaCIC residual-choice budget frontier route)**.** *Every confluence consumer of an accepted* MetaCICResidualChoiceBudget$^{\uparrow}$ *packet must expose the typed-window row T, residual-pair row R, choice-boundary row Q, bounded local-join row J, obstruction row O, componentwise transport row H, replay row C, provenance row P, and local naming row N before it may hand the budget to a residual-join, residual-diamond, residual-checker, or parallel-diamond consumer. The route does not discharge residual substitution, prove Church–Rosser, prove subject reduction, identify evaluator equality, identify quotient term equality, or certify kernel consistency.*

Proof. Start from the obligation surface of Definition 8191.2. It makes $T, R, Q, J, O, H, C, P, N$ the visible budget rows.

Read the neighbouring MetaCIC packets through the provenance row. Residual-substitution boundary, residual checker, residual-join budget, residual-diamond budget, and parallel-diamond frontier consumers may inspect the same finite budget, but only after Q and O remain visible.

Thus the handoff is bounded and conditional. Since no carrier coordinate supplies residual-substitution compatibility, unrestricted confluence, subject reduction, evaluator equality, quotient equality, or kernel consistency, none of those endpoints can be exported by the budget route. □

The Lean TasteGate surface for this packet requires future Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/MetaCICResidualChoiceBudgetUp/TasteGate.lean`, with field-projection witnesses separating $T, R, Q, J, O, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}$ packet is generated from a typed-window row, residual-pair row, choice-boundary row, bounded local-join row, obstruction-retention row, componentwise transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8191.1, Definition 8191.2, and Definition 8191.3 seed the finite residual-choice frontier over typed-window, residual-pair, choice-boundary, bounded local-join, obstruction, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualChoiceBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full residual substitution, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, global conversion decidability, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-pair exposure, choice-boundary retention, bounded local-join exposure, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualChoiceBudgetUp.

Chapter 8192

A Concrete Naming Certificate for $\mathsf{AbelCesaroTheorem}^{\uparrow}$

Following Hardy and Bishop–Bridges constructive analysis, the $\mathsf{AbelCesaroTheorem}^{\uparrow}$ packet records the finite handoff by which an Abel summability window and a Cesaro mean window are read before a Real-and-Completion sequence endpoint is exposed. It sits after the Abel route of chapter 1763, the averaging surface of chapter 1765, and the regular rational readback of chapter 301. The packet names only displayed windows, dyadic budgets, regular readback rows, and terminal seals; it is not Real completeness, quotient series equality, a selected limit, countable choice, trichotomy, or open-cover compactness.

Definition 8192.1 (Abel–Cesaro theorem carrier)**.** *An* $\mathsf{AbelCesaroTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, H, D, R, E, K, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, exposing the Abel window inspected at the given precision.*
2. *H is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row, exposing the Cesaro averaging window compared with L.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row that assigns the Abel–Cesaro comparison a finite tolerance ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after L, H, D have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row handed to sequence or completion-facing consumers only through R.*
6. *K records componentwise* hsame *transport for the Abel, Cesaro, dyadic, regular-readback, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through L, H, D, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{AbelCesaroTheorem}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelCesaroTheorem}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these nine rows. It contains no coordinate for Real completeness, quotient series equality, selected limits, countable choice, trichotomy, or open-cover compactness.

Theorem 8192.2 (Abel–Cesaro theorem NameCert obligations)**.** *For every accepted* $\mathsf{AbelCesaroTheorem}^{\uparrow}$ *carrier* $A = (L, H, D, R, E, K, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AbelCesaroTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for that nine-row packet, Abel-window exposure through* L*, Cesaro-window exposure through* H*, dyadic comparison budgeting through* D*, regular rational readback through* R*, Real sealing through* E*, and transport, replay, provenance, and local naming through* K, C, P, N*.*

Lean4 ✓ `BEDC.Derived.AbelCesaroTheoremUp.AbelCesaroTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 8192.1. Carrier admission exposes precisely the Abel row L, the Cesaro row H, the dyadic budget D, the regular readback R, the real seal E, and the structural rows K, C, P, N.

The analytic rows are read in order. An accepted comparison first displays the Abel and Cesaro windows, then reads the dyadic tolerance ledger that licenses the finite comparison, and only then reaches the regular rational readback row.

The terminal seal is downstream of that readback. Since the remaining rows only transport, replay, source, and name the same displayed packet, the local NameCert surface is exhausted by the stated rows and cannot import the refused endpoints. □

Theorem 8192.3 (Abel–Cesaro theorem regular handoff)**.** *Every sequence endpoint or completion-facing read exported by an accepted* $\mathsf{AbelCesaroTheorem}^{\uparrow}$ *carrier factors through*

$$L, H, D, R, E, K, C, P, N.$$

Thus the handoff from Abel and Cesaro data to a terminal $\mathsf{Real}^{\uparrow}$ *seal must pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row, rather than through Real completeness, quotient series equality, a selected limit, countable choice, trichotomy, or open-cover compactness.*

Proof. Start with a consumer requesting the endpoint. By Definition 8192.1, the only available Abel data are in L and the only available Cesaro data are in H.

The comparison budget is the dyadic row D. No real-facing read can bypass this row, because R is the first regular rational readback row and it is declared only after L, H, D have been displayed.

Finally, the terminal seal E is reached through R, with K, C, P, N preserving and naming the same finite route. Therefore any endpoint or completion-facing consumer receives the stated handoff rather than one of the refused host principles. □

Falsifiable prediction: An accepted $\mathsf{AbelCesaroTheorem}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the $\mathsf{AbelSummation}^{\uparrow}$ row, $\mathsf{HardyCesaroMean}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ budget row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback row refutes the carrier surface.

Independence witness: abel-summation, hardy-cesaro-mean, regseqrat, dyadicratcore, real: no listed sibling supplies the whole nine-row packet $A = (L, H, D, R, E, K, C, P, N)$, because each sibling contributes only a summation route, averaging route, readback surface, budget ledger, or seal face while this chapter binds the Abel–Cesaro theorem handoff into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelCesaroTheorem}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelCesaroTheorem}^{\uparrow}$ packet is generated from an $\mathsf{AbelSummation}^{\uparrow}$ row, a $\mathsf{HardyCesaroMean}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ budget row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AbelCesaroTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 8192.1, Definition 8192.2, and Definition 8192.3 seed the finite Abel–Cesaro theorem handoff over AbelSummation, HardyCesaroMean, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AbelCesaroTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient series equality, selected limits, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and prove the finite NameCert obligation surface, regular handoff theorem, and TasteGate field faithfulness for BEDC.Derived.AbelCesaroTheoremUp.

Chapter 8193

A Concrete Naming Certificate for $\mathsf{TailWindowUniformSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{TailWindowUniformSequence}^{\uparrow}$ packet records the finite tail-window interface used when a uniformly continuous sequence is inspected only through displayed windows and a shared modulus row. It refines the sequence route of chapter 4287 through the modulus discipline of chapter 369, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a tail-window surface without selected limits, quotient streams, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8193.1 (Tail-window uniformly continuous sequence carrier)**.** *A* $\mathsf{TailWindowUniformSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, M, W_0, W_1, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the sequence-index window row naming the finite tail interval being inspected.*
2. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row used by every read in the window.*
3. *W_0 and W_1 are the paired* $\mathsf{StreamName}^{\uparrow}$ *window reads cut from the same tail interval.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the two window reads and their shared modulus budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, M, W_0, W_1, R have been displayed.*
6. *H records componentwise* hsame *transport for the index window, modulus, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the index window through the shared modulus, StreamName windows, RegSeqRat readback, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{TailWindowUniformSequence}^{\uparrow}$, $\mathsf{UniformlyContinuousSequence}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TailWindowUniformSequence}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limiting value, quotient equality of streams, a private modulus per side of the window, countable choice, trichotomy, or ambient completeness.

Theorem 8193.2 (Tail-window uniformly continuous sequence obligations)**.** *For every accepted* $\mathsf{TailWindowUniformSequence}^{\uparrow}$ *carrier*

$$T = (I, M, W_0, W_1, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TailWindowUniformSequence}^{\uparrow}}$ *surface consists of the following five obligation rows:*

1. *carrier admission for the displayed sequence-index window, shared modulus, paired StreamName reads, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming;*
2. *shared-modulus exposure, requiring M to be read before either window can be consumed as a uniformly continuous sequence tail;*
3. *tail-window readback, requiring W_0, W_1 and R to expose only the finite inspected tail interval named by I;*
4. *Real-seal handoff, requiring E to appear only after the shared modulus and regular readback rows have been displayed;*
5. *nonescape for selected limits, quotient streams, private side moduli, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8193.1. The analytic coordinates are exactly the index window, shared modulus, two StreamName reads, regular readback, and terminal Real seal.

Read the shared modulus row before the window rows. Since M is common to W_0 and W_1, the two reads cannot choose separate tail thresholds or private uniform-continuity data.

The RegSeqRat row repacks only the displayed window data. Thus a Real-facing consumer reaches E through I, M, W_0, W_1, R rather than through a selected limit or quotient stream.

Finally the structural rows H, C, P, N transport, replay, source, and name the same route. They do not add countable choice, trichotomy, private moduli, or ambient completeness, so the five listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailWindowUniformSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TailWindowUniformSequence}^{\uparrow}$ packet is generated from a finite sequence-index window, a shared UniformModulus row, paired StreamName window reads, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TailWindowUniformSequence}^{\uparrow}}$, seed)

Scope closed: Definition 8193.1 and Definition 8193.2 seed the finite tail-window route for uniformly continuous sequence consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TailWindowUniformSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exposure, finite tail-window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.TailWindowUniformSequenceUp.

Chapter 8194

A Concrete Naming Certificate for $\mathsf{CompactModulusTailFusion}^{\uparrow}$

Following the Bishop–Bridges finite-net proof of compact uniform continuity, the $\mathsf{CompactModulusTailFusion}^{\uparrow}$ packet records the ledger that fuses a compact-source uniform modulus with the tail-window rows later consumed by Real and Completion arguments. It realizes the route of chapter 9237 through the compact modulus surface of chapter 2299, the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite fusion row, not a host modulus function, selected cover center, quotient stream, countable-choice schedule, or ambient compactness theorem.

Definition 8194.1 (Compact modulus tail fusion carrier)**.** *A* $\mathsf{CompactModulusTailFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, U, T, W, R, E, H, C, P, L, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactUniformModulus}^{\uparrow}$ *row produced from compact finite-net and pointwise-modulus data.*
2. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row extracted from the compact modulus surface.*
3. *T is the shared tail-window ledger naming the finite sequence tail on which U is used.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window read attached to that shared tail ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the window values and uniform-modulus tolerance data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after K, U, T, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the compact modulus, uniform-modulus handoff, tail ledger, window read, regular readback, Real seal, replay, provenance, fusion ledger, and local naming rows.*
8. *C records displayed* Cont *replay from the compact modulus row through the uniform-modulus handoff, tail window, StreamName read, RegSeqRat readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactModulusTailFusion}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *L is the local fusion ledger recording that the compact modulus row and the tail-window readback are consumed in one route.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusTailFusion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host modulus selector, an unrecorded compact cover center, a quotient stream, countable choice, trichotomy, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactModulusTailFusionUp.CompactModulusTailFusionTasteGate single carrier alignment`

Theorem 8194.2 (Compact modulus tail fusion obligations)**.** *For every accepted* $\mathsf{CompactModulusTailFusion}^{\uparrow}$ *carrier*

$$F = (K, U, T, W, R, E, H, C, P, L, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusTailFusion}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-modulus exposure through K, uniform-modulus handoff through U, shared tail-window ledgering through T, finite StreamName readback through W, RegSeqRat readback through R, Real-seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, fusion ledgering through L, local naming through N, and nonescape for host modulus selectors, hidden compact centers, quotient streams, countable choice, trichotomy, selected limits, and ambient completeness.*

Proof. Project the carrier of Definition 8194.1. The compact analytic source is K, its tolerance-only handoff is U, and the tail-facing consumer rows are T, W, R, E.

Read the compact modulus before the tail data. The row K supplies the finite compact-uniform route, and U is the displayed uniform-modulus consumer row, so the later tail window cannot replace it with a host selector or private modulus.

Read the tail rows next. The ledger T, window W, and regular readback R bind the finite sequence tail before E is exposed as a Real seal.

Finally H, C, P, L, N transport, replay, source, ledger, and name the same route. Since the carrier lacks hidden compact centers, quotient streams, countable choice, trichotomy, selected limits, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusTailFusion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusTailFusion}^{\uparrow}$ packet is generated from a compact-uniform modulus row, a UniformModulus handoff, a shared tail-window ledger, finite StreamName readback, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, a fusion ledger, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactModulusTailFusion}^{\uparrow}}$, seed)

Scope closed: Definition 8194.1 and Definition 8194.2 seed the finite fusion route from compact modulus rows to tail-window Real readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactModulusTailFusion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host modulus selectors, hidden compact centers, quotient streams, countable choice, trichotomy, selected limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-modulus exposure, uniform-modulus handoff, tail-window readback, Real-seal exposure, fusion ledgering, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusTailFusionUp.

Chapter 8195

A Concrete Naming Certificate for $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$

Following Weierstrass and Bishop–Bridges, the $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ packet records the textbook route from an absolute majorant series to uniform convergence of a real-valued series. It is adjacent to the M-test surface of chapter 399, the uniform Cauchy criterion of chapter 435, the regular rational readback of chapter 301, the finite stream windows of chapter 126, and the real-name seal of chapter 38. The carrier exposes only finite majorant, absolute-value, tail-modulus, readback, transport, replay, provenance, and naming rows; it does not import host completeness, countable choice, or a selected limit function.

Definition 8195.1 (Uniform absolute convergence carrier)**.** *A* $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, W, T, S, Q, R, H, C, P, N)$$

with the following visible rows:

1. *A is the majorant-series row, displaying finitely inspected nonnegative rational or real-name bounds and the summable tail ledger consumed by the packet.*
2. *W is the absolute-value window row, recording the displayed inequalities $|f_n(x)| \leq a_n$ for the inspected indices, source probes, and tolerance requests.*
3. *T is the shared uniform-tail modulus row, read once from the majorant tail and reused for every inspected source probe.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected indices, probes, partial sums, and requested tail tolerances.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the partial-sum and tail windows generated by S, W, T.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *H records componentwise* hsame *transport for majorant, absolute-value, tail-modulus, stream-window, regular-readback, real-seal, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a tolerance request through A, W, T, S, Q, R.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformAbsoluteConvergence}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the majorant tail is displayed before the shared uniform modulus is consumed, every absolute-value window is bounded by the same majorant row, every stream window reads that common tail modulus, and the regular rational and real-name rows preserve the same finite route. It has no coordinate for host completeness of function spaces, selected pointwise limits, countable-choice schedules, quotient equality of series, or detached absolute-convergence facts.

Theorem 8195.2 (Uniform absolute convergence NameCert obligations)**.** *For every accepted* $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ *carrier* $U = (A, W, T, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformAbsoluteConvergence}^{\uparrow}}$ *obligations are exactly carrier admission for the majorant series, absolute-value windows, shared uniform-tail modulus, finite stream windows, regular rational readback, real-name seal, componentwise transport, continuation replay, provenance, and local naming; majorant exactness requiring each displayed absolute-value window to read the same* A *row; tail-modulus exactness requiring* T *to be produced from the majorant tail before any stream window is consumed; readback exactness through* Q*; real-seal boundary through* R*; and non-escape from host completeness, selected limits, countable choice, quotient series equality, and detached absolute-convergence predicates.*

Proof. Project the carrier of Definition 8195.1. The majorant row A, the absolute-value windows W, the shared tail modulus T, the stream windows S, the regular readback Q, and the real seal R are the only analytic rows visible to the classifier.

Majorant exactness is read before any uniform tail request. Each displayed inequality in W points to the same majorant row A, so the packet cannot replace the majorant by a probe-dependent bound or by an external convergence assertion.

The shared tail modulus T is then consumed by the finite windows in S. Those windows determine the regular rational readback Q, and the $\mathsf{Real}^{\uparrow}$ row R is reached only after the displayed stream and readback rows have been exposed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no row for host completeness, selected pointwise limits, countable choice, quotient equality of series, or detached absolute-convergence facts, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: If the shared uniform-tail modulus row is removed, a $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ consumer can still inspect majorant and absolute-value windows, but it can no longer force a single tolerance-only route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback.

Independence witness: weierstrass-m-test, uniform-cauchy-criterion, regseqrat, streamname, real: $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a majorant series, absolute-value windows, a shared uniform-tail modulus, finite stream windows, regular rational readback, a Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformAbsoluteConvergence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformAbsoluteConvergence}^{\uparrow}$ packet starts from a displayed majorant series, records absolute-value windows, extracts one shared uniform-tail modulus, routes finite StreamName windows through RegSeqRat readback, seals the result as RealUp data, and preserves the same finite route by hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformAbsoluteConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 8195.1 and Definition 8195.2 seed the majorant, absolute-window, uniform-tail, finite-window, readback, and Real-seal carrier for uniform absolute convergence over real series.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformAbsoluteConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host completeness of function spaces, selected pointwise limits, countable choice, quotient equality of series, detached absolute-convergence predicates, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, majorant exactness, tail-modulus exactness, stream-window exactness, regular-readback boundary, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.UniformAbsoluteConvergenceUp.

Chapter 8196

A Concrete Naming Certificate for $\mathsf{PhaseCurvatureTransport}^{\uparrow}$

The $\mathsf{PhaseCurvatureTransport}^{\uparrow}$ packet realizes the hyperbolic phase-geometry route as finite transport and curvature-budget data. It is adjacent to the curvature-budget surface of chapter 6437, the phase-holonomy route of chapter 8162, the disk metric surface of chapter 6209, the hyperbolic metric interface of chapter 3474, and the real-name seal of chapter 38. The carrier records only finite disk, metric, boundary, phase, curvature-budget, Real-seal, transport, replay, provenance, and naming rows; it does not assert a global smooth connection, choose geodesic representatives, or identify boundary refusal with an interior phase center.

Definition 8196.1 (Phase curvature transport carrier)**.** *A* $\mathsf{PhaseCurvatureTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (D, M, B, \Phi, K, R, H, T, P, N)$$

with the following visible rows:

1. *D is the finite* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *disk-window row, displaying the inspected disk probes and admissible boundary-facing routes.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *readback row consumed by the packet before any phase transport is inspected.*
3. *B is the boundary-refusal row, recording which far-end or boundary reads are preserved as boundary data and not collapsed into interior centers.*
4. *Φ is the phase-transport row, listing the finite phase windows, holonomy comparisons, and observer-expansion readbacks inspected by the packet.*
5. *K is the curvature-budget handoff row, reusing the budget discipline of chapter 6437 for the displayed phase transport.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for metric, phase, and curvature-budget readbacks reached only after D, M, B, Φ, K have been displayed.*
7. *H records componentwise* hsame *transport for disk, metric, boundary, phase, curvature-budget, real-seal, replay, provenance, and local-name rows.*
8. *T records displayed* Cont *replay from disk windows through metric readback, boundary refusal, phase transport, curvature-budget handoff, and real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{PhaseCurvatureTransport}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicPhaseCurvatureBudget}^{\uparrow}$, $\mathsf{HyperbolicPhaseHolonomyIndex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhaseCurvatureTransport}^{\uparrow}}$ *row.*

The classifier accepts a packet only when disk reads pass through D, metric reads pass through M, boundary refusal is read from B, phase transport is displayed in Φ, curvature-budget constraints are carried by K, and all Real-facing outputs preserve R, H, T, P, N. It has no coordinate for a global connection form, chosen geodesic representative, quotient boundary equality, hidden smooth curvature tensor, or an identification of the boundary refusal with an interior phase center.

Theorem 8196.2 (Phase curvature transport NameCert obligations)**.** *For every accepted* $\mathsf{PhaseCurvatureTransport}^{\uparrow}$ *carrier $C = (D, M, B, \Phi, K, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PhaseCurvatureTransport}^{\uparrow}}$ *obligations are exactly carrier admission for the disk-window row, hyperbolic metric readback, boundary-refusal row, phase-transport row, curvature-budget handoff, real-name seal, componentwise transport, continuation replay, provenance, and local naming; metric exactness requiring all phase windows to read the same displayed disk and metric rows; boundary-refusal exactness requiring transport to preserve the boundary row before phase readback is sealed; curvature-budget exactness requiring K to be consumed before any Real-facing phase output; and non-escape from global connection forms, chosen geodesic representatives, quotient boundary equality, hidden smooth curvature tensors, and interior-center collapse.*

Proof. Project the carrier of Definition 8196.1. The finite disk row D, metric row M, boundary-refusal row B, phase-transport row Φ, curvature-budget row K, and Real seal R are the only geometric rows available to the classifier.

Metric exactness is read first. A phase window cannot be consumed until the displayed disk probes have passed through D and their hyperbolic metric readback has passed through M.

Boundary and curvature data are then checked in order. The row B keeps the far-end refusal as boundary data, while K supplies the finite curvature-budget handoff for the phase transport in Φ before any Real-facing output is sealed by R.

Finally, H, T, P, N transport, replay, source, and name the same route. Because the carrier has no row for global connection forms, chosen geodesic representatives, quotient boundary equality, hidden smooth curvature tensors, or interior-center collapse, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: If the curvature-budget row is removed, phase transport can no longer force the same boundary refusal and metric-readback route.

Independence witness: hyperbolic-phase-curvature-budget, hyperbolic-phase-holonomy, poincare-disk-metric, hyperbolic-metric, real: not bijective because $\mathsf{PhaseCurvatureTransport}^{\uparrow}$ jointly carries phase transport, curvature-budget handoff, disk metric, boundary refusal, Real seal, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseCurvatureTransport}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PhaseCurvatureTransport}^{\uparrow}$ packet starts from finite Poincare disk windows, reads hyperbolic metric data, preserves boundary refusal, transports phase windows through a curvature-budget handoff, seals Real-facing outputs, and preserves the same route by hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PhaseCurvatureTransport}^{\uparrow}}$, seed)

Scope closed: Definition 8196.1 and Definition 8196.2 seed the disk, metric, boundary-refusal, phase-transport, curvature-budget, Real-seal, transport, replay, provenance, and naming carrier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PhaseCurvatureTransport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global smooth connection data, chosen geodesic representatives, quotient boundary equality, hidden curvature tensors, interior-center collapse, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric exactness, boundary-refusal exactness, curvature-budget exactness, Real-seal boundary, transport, replay, provenance, local naming, and Nontrivial and Field-Faithful instances in lean4/BEDC/Derived/PhaseCurvatureTransportUp/TasteGate.lean.

Chapter 8197

A Concrete Naming Certificate for $\mathsf{SignedDyadicInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SignedDyadicInterval}^{\uparrow}$ packet records the finite signed dyadic interval data used before regular rational and real-name consumers inspect an enclosure. It sits after the dyadic endpoint core of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the interval-operation surface of chapter 1645. The packet names the signed endpoint order, containment ledger, and real-facing handoff rows without importing quotient real equality, host interval equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8197.1 (Signed dyadic interval carrier)**.** *A* $\mathsf{SignedDyadicInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are lower and upper endpoint rows accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *σ_L and σ_U are the signed-endpoint rows, recording the visible sign data attached to the two dyadic endpoints.*
3. *O is the lower-before-upper order ledger, read only through dyadic comparison rows over L and U.*
4. *W is the finite window row exported to* $\mathsf{StreamName}^{\uparrow}$ *consumers.*
5. *I is the interval-arithmetic handoff row, exposing the same endpoints and containment comparisons through* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$*.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for consumers that turn the displayed interval into a regular rational enclosure.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the endpoint, sign, window, interval-arithmetic, and regular-readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint, sign, order, window, containment, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{SignedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicInterval}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one signed dyadic interval packet. It has no coordinate for quotient real equality, host interval equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8197.2 (Signed dyadic interval NameCert obligations)**.** *For every accepted* $\mathsf{SignedDyadicInterval}^{\uparrow}$ *carrier* $S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint rows, signed-endpoint rows, endpoint-order ledger, StreamName window, interval-arithmetic handoff, RegSeqRat readback, Real seal, componentwise transport, continuation replay, provenance, and local naming;*
2. *endpoint sign exactness, requiring σ_L and σ_U to remain attached to the displayed dyadic endpoint rows L and U;*
3. *order stability, requiring O to transport along* hsame *and displayed* Cont *replay without replacing dyadic comparison by host real order;*
4. *containment handoff, requiring I to expose every interval containment row through* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *using the same displayed endpoints;*
5. *completion boundary, requiring W, R, E to route the finite window through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *in that order;*
6. *ledger non-escape, excluding quotient real equality, host interval equality, trichotomy, selected representatives, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 8197.1. The projection exposes exactly the two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, the signed-endpoint rows, the dyadic order ledger, the StreamName window, the interval-arithmetic handoff, the RegSeqRat readback, the Real seal, and the structural rows H, C, P, N.

Endpoint sign exactness and order stability are read from $\sigma_L, \sigma_U, O, H, C$. Componentwise transport and displayed replay move the named endpoint histories together with their sign and order ledgers, so no host real order or quotient equality row is introduced.

Containment handoff is the row I. It points to the existing $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ enclosure surface and uses the same displayed endpoint rows L, U rather than a detached interval object.

Finally the completion boundary is the route $W \longmapsto R \longmapsto E$. A completion-facing consumer receives finite windows, regular rational readback, and the terminal real seal only in that order. The excluded ambient principles are absent from the carrier inventory. □

Theorem 8197.3 (Signed dyadic interval containment stability)**.** *Let $S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N)$ be an accepted* $\mathsf{SignedDyadicInterval}^{\uparrow}$ *carrier. Any containment read exported through I remains stable under the componentwise* hsame *transport and displayed* Cont *replay rows H and C, and every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the transported containment only after the signed dyadic endpoint rows and endpoint-order ledger have been displayed.*

Proof. Start from the containment row I in Definition 8197.1. Its endpoints are the displayed rows L, U together with their signed data σ_L, σ_U and order ledger O.

Apply the NameCert obligations of Definition 8197.2. They force the containment handoff to remain inside $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ and force H, C to transport and replay the same endpoint, sign, order, and containment rows componentwise.

The downstream completion route is then read through W, R, E. Since those rows are reached only after the displayed containment row, every listed consumer sees the same transported interval packet and cannot replace it by host interval equality, quotient real equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8197.4 (Signed dyadic interval endpoint sign exactness)**.** *Let $S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N$ be an accepted $\mathsf{SignedDyadicInterval}^{\uparrow}$ carrier. Under every transported read recorded by H and every displayed replay recorded by C, the sign rows σ_L and σ_U remain attached respectively to the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows L and U.*

Proof. Project the carrier tuple in Definition 8197.1. The endpoint rows L, U and the sign rows σ_L, σ_U occur as named coordinates of the same finite packet, not as separately chosen representatives.

Apply the obligation surface of Definition 8197.2. Its endpoint sign exactness row is read through σ_L, σ_U, H, C, so componentwise hsame transport and displayed Cont replay move each sign row together with its endpoint row.

No quotient real equality, selected representative, host interval equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is available in the carrier inventory. Therefore a consumer can read only the displayed attachment of σ_L to L and σ_U to U. □

Theorem 8197.5 (Signed dyadic interval order stability)**.** *Let $S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N)$ be an accepted $\mathsf{SignedDyadicInterval}^{\uparrow}$ carrier. The lower-before-upper ledger O is preserved under the componentwise hsame transport row H and the displayed Cont replay row C; the preserved order read is the displayed dyadic comparison over L and U, not host real order or trichotomy.*

Proof. Project O, H, C from the carrier of Definition 8197.1. The order coordinate is a lower-before-upper dyadic ledger over the displayed endpoints L, U.

Use Definition 8197.2. The order-stability obligation requires the ledger O to travel through the same componentwise transport and replay rows that carry the endpoints and their signs.

Since the carrier contains no host real order, trichotomy, quotient equality, representative selector, countable-choice coordinate, or ambient completeness coordinate, the replayed order read cannot be replaced by any external comparison principle. It remains exactly the displayed dyadic ledger O over L, U. □

Theorem 8197.6 (Signed dyadic interval completion boundary)**.** *Let $S = (L, U, \sigma_L, \sigma_U, O, W, I, R, E, H, C, P, N)$ be an accepted $\mathsf{SignedDyadicInterval}^{\uparrow}$ carrier. Every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer must read the route $W \longmapsto R \longmapsto E$ only after the signed endpoint rows σ_L, σ_U, the order ledger O, and the containment handoff I have been displayed.*

Proof. Project the completion-facing coordinates W, R, E from Definition 8197.1. They are ordered as the finite window, regular rational readback, and terminal real seal of the same signed interval packet.

Apply Definition 8197.2 together with Definition 8197.3. The obligation surface places the completion route after endpoint signs, endpoint order, and containment handoff, while containment stability ensures the handoff remains the displayed $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ row transported by H and replayed by C.

Thus a completion-facing consumer receives $W \longmapsto R \longmapsto E$ only through the displayed finite route. The carrier offers no quotient real equality, host interval equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness through which the consumer could bypass those rows. □

Falsifiable prediction: Every accepted $\mathsf{SignedDyadicInterval}^{\uparrow}$ read that exports containment to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ without first exposing signed dyadic endpoints, endpoint order, the $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ containment row, componentwise transport, displayed replay, provenance, and local naming is outside the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, dyadic-interval-arithmetic: no listed sibling supplies the joint packet containing signed dyadic endpoints, endpoint order, finite StreamName windows, DyadicIntervalArithmetic containment, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDyadicInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDyadicInterval}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, signed-endpoint rows, an endpoint-order ledger, finite StreamName windows, a DyadicIntervalArithmetic containment handoff, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDyadicInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8197.1, Definition 8197.2, Definition 8197.3, Definition 8197.4, Definition 8197.5, and Definition 8197.6 seed the finite signed dyadic interval route from DyadicRatCore endpoints through StreamName, RegSeqRat, Real, and DyadicIntervalArithmetic consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDyadicInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, host interval equality, trichotomy, selected representatives, countable choice, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, signed-endpoint exactness, endpoint-order stability, containment handoff, completion-boundary, classifier transport, and ledger non-escape proofs for BEDC.Derived.SignedDyadicIntervalUp.

Chapter 8198

A Concrete Naming Certificate for $\mathsf{UniformLimitIntegral}^{\uparrow}$

Following Bishop and Bridges on constructive integration, the $\mathsf{UniformLimitIntegral}^{\uparrow}$ packet records the finite BEDC route by which a uniformly convergent family of Riemann-integrable real-valued names yields an integral read for the displayed limit. It sits after the uniform-limit continuity boundary of chapter 1748, the Riemann-integral carrier of chapter 1526, the regular rational readback of chapter 301, the finite stream-name window of chapter 126, and the terminal real-name seal of chapter 38. The packet names only the displayed route from uniform convergence, integral packets, regular rational readback, and Real sealing; it does not import host function equality, selected limiting partitions, countable choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8198.1 (Uniform limit integral carrier)**.** *A* $\mathsf{UniformLimitIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, U, J, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family row, listing the displayed real-valued approximants and their common interval window;*
2. *U is the uniform-limit row, read through* $\mathsf{UniformLimitContinuous}^{\uparrow}$*, recording the shared modulus and the displayed limit-name route;*
3. *J is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row for the approximants, exposing only tagged-mesh, Darboux-gap, and integral-readback data already present in the Riemann packet;*
4. *W is the finite window row through* $\mathsf{StreamName}^{\uparrow}$*, recording the observation schedule used by the approximants, the limit read, and the integral read;*
5. *Q is the regular rational readback row through* $\mathsf{RegSeqRat}^{\uparrow}$*, recording the rational Cauchy data used to compare integral values at a requested tolerance;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the limit integral value;*
7. *H records componentwise* hsame *transport for F, U, J, W, Q, E;*
8. *C records displayed* Cont *replay through the uniform-limit, Riemann-integral, stream-window, regular-rational, and Real-seal rows;*
9. *P is the* Pkg *provenance over* $\mathsf{UniformLimitIntegral}^{\uparrow}$, $\mathsf{UniformLimitContinuous}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformLimitIntegral}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host equality of functions, a selected sequence of partitions, a measure-completion object, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 8198.2 (Uniform limit integral NameCert obligations)**.** *For the carrier of Definition 8198.1, the local* $\mathsf{NameCert}_{\mathsf{UniformLimitIntegral}^{\uparrow}}$ *surface consists exactly of carrier exposure for* $I = (F, U, J, W, Q, E, H, C, P, N)$*, uniform-limit exposure through* U*, Riemann-integral packet exposure through* J*, stream-window and regular-rational readback through* W, Q*, terminal Real sealing through* E*, structural transport through* H, C, P*, and local naming through* N*.*

Proof. Project the carrier rows of Definition 8198.1. The finite family, uniform-limit, Riemann-integral, stream-window, regular-rational, and Real-seal rows are separate displayed coordinates before the local name is consumed.

The structural rows H, C, P, N transport, replay, source, and name that same finite packet. Since the carrier has no host function-equality, selected-partition, measure-completion, choice, or ambient-completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8198.3 (Uniform limit integral Riemann handoff)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformLimitIntegral}^{\uparrow}$ *packet must read the finite route*

$$F \longmapsto U \longmapsto J \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the integral of the displayed limit is available only after the uniform-limit row, the Riemann-integral row, the stream-window row, the regular-rational readback row, and the Real-seal row are all exposed.

Proof. Use the NameCert obligations of Definition 8198.2. The Riemann handoff row J is downstream of the displayed family and uniform-limit rows, while the integral value is downstream of the stream-window and regular-rational readback rows.

The continuation row C replays the route in the order shown, and the provenance row P records the allowed sibling packets. A Riemann-integral consumer therefore receives a finite BEDC handoff, not an external limiting theorem or a hidden partition-selection principle. □

Theorem 8198.4 (Uniform limit integral Real seal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{UniformLimitIntegral}^{\uparrow}$ *packet is reached only through* $F, U, J, W, Q, E, H, C, P, N$*. It cannot be read as host equality of integrals, quotient equality of functions, a selected limiting partition family, countable choice, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 8198.1. The Real-facing coordinate is the row E, and E is displayed only after the uniform-limit, Riemann-integral, stream-window, and regular-rational readback rows have been exposed.

Apply the Riemann handoff theorem Definition 8198.3. Any consumer that reaches E has already passed through the finite route $F, U, J, W, Q, H, C, P, N$. None of those rows contains a host equality, quotient, selected partition family, countable-choice schedule, measure-completion object, or ambient-completeness theorem, so the Real seal cannot escape the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitIntegral}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformLimitIntegral}^{\uparrow}$ packet is generated from finite family rows, a uniform-limit row, a Riemann-integral handoff row, stream-window rows, regular-rational readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8198.1, Definition 8198.2, Definition 8198.3, and Definition 8198.4 seed the finite uniform-limit integral route over UniformLimitContinuous$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{UniformLimitIntegral}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host equality of functions, selected limiting partitions, countable choice, measure completion, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, uniform-limit exposure, Riemann-integral hand-off, stream-window readback, regular-rational comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformLimitIntegralUp.

Chapter 8199

A Concrete Naming Certificate for CompactIntervalFanModulusSelector$^{\uparrow}$

Following Bishop and Bridges on compact intervals and fan-bar moduli, the CompactIntervalFanModulusSelector$^{\uparrow}$ packet records the finite selector that carries compact interval source data through fan-bar depth, dyadic tolerance, finite StreamName windows, RegSeqRat readback, and a terminal Real seal. It sits beside the located interval boundary of chapter 7165, the compact interval fan route of chapter 6946, and the finite fan-bar modulus surface of chapter 4297. The packet supplies an L10 Real and Completion route without asserting open-cover compactness, choosing branches, quotienting real names, invoking countable choice, or importing ambient Real$^{\uparrow}$ completeness.

Definition 8199.1 (Compact interval fan modulus selector carrier)**.** *A* CompactIntervalFanModulusSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, B, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, read through the Bishop compact interval and located interval surfaces before any fan-bar depth is consumed.*
2. *B is the finite fan-bar depth row, read through* CompactIntervalFanBridge$^{\uparrow}$, LocatedIntervalFanBarrier$^{\uparrow}$, *and* FiniteFanBarModulus$^{\uparrow}$ *as a bounded prefix-depth certificate.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row carrying the requested precision and the finite comparison budget used by the selector.*
4. *S is the finite* StreamName$^{\uparrow}$ *window ledger opened by the depth and tolerance rows.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the regular rational approximants observed inside the displayed windows.*
6. *E is the terminal* Real$^{\uparrow}$ *seal, available only after I, B, D, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for interval source, fan-bar depth, dyadic tolerance, stream windows, regular-rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact interval source data through the fan-bar depth and then through D, S, Q, E.*

9. *P is the* Pkg *provenance over* CompactIntervalFanModulusSelector$^{\uparrow}$, CompactIntervalFanBridge$^{\uparrow}$, LocatedIntervalFanBarrier$^{\uparrow}$, FiniteFanBarModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CompactIntervalFanModulusSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, D, S, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, a selected branch, quotient real equality, countable choice, host real comparison, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8199.2 (Compact interval fan modulus selector obligations)**.** *For every accepted* CompactIntervalFanMod *carrier*

$$M = (I, B, D, S, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CompactIntervalFanModulusSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, compact interval source exposure through I, finite fan-bar depth exposure through B, dyadic tolerance accounting through D, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback through Q, terminal* Real$^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8199.1. The accepted packet displays exactly the interval source, finite fan-bar depth, dyadic tolerance, stream-window, regular-rational, Real-seal, transport, replay, provenance, and naming rows.

The analytic route is ordered. The compact interval source row I is exposed first, the fan-bar row B supplies the finite depth bound, and the dyadic row D records the tolerance budget consumed by the selected windows.

The readback rows are downstream of that finite prefix. The window ledger S is opened from the depth and tolerance rows, Q reads regular rational data inside those windows, and E seals the Real-facing result only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite selector. Since the carrier has no coordinate for open-cover compactness, selected branches, quotient real equality, countable choice, host real comparison, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 8199.3 (Compact interval fan modulus finite route)**.** *Every L10 Real and Completion consumer of an accepted* CompactIntervalFanModulusSelector$^{\uparrow}$ *carrier factors through the finite route*

CompactIntervalFanBridge$^{\uparrow}$, LocatedIntervalFanBarrier$^{\uparrow}$, FiniteFanBarModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, Re

In carrier notation, the consumer reads I, B, D, S, Q, E before using the structural rows H, C, P, N. The route exports a displayed modulus selector from compact interval and fan-bar evidence to Real-facing readback, and it exports no open-cover compactness theorem, selected branch, quotient real equality, countable-choice row, host equality test, or ambient Real$^{\uparrow}$ *completeness theorem.*

Proof. Start with the compact interval and fan rows. The compact source row I supplies the interval data, while B records the finite fan-bar depth available through the adjacent compact interval and finite bar chapters.

The tolerance and window rows come next. The dyadic row D fixes the requested precision, and the finite window ledger S lists only the observations licensed by the displayed depth and tolerance.

The Real-facing suffix is Q, E. RegSeqRat readback is performed inside the finite windows, and the terminal Real seal is visible only after that readback has been replayed.

The structural rows H, C, P, N preserve the same order under transport, continuation replay, provenance, and local naming. Therefore every L10 consumer factors through the displayed finite route, while each excluded compactness, choice, branch, quotient, host-equality, or ambient-completeness reading lacks a carrier coordinate. □

Falsifiable prediction: Every accepted CompactIntervalFanModulusSelectorUp Real or Completion-facing read factors through compact interval source, finite fan-bar depth, dyadic tolerance, finite StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: located interval fan barrier, compact interval fan bridge, finite fan bar modulus, dyadicratcore, streamname, regseqrat, real: $\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact interval source data, fan-bar depth, dyadic tolerance, finite observation windows, regular rational readback, Real sealing, and structural support while refusing open-cover compactness, selected branches, quotient real equality, countable choice, host real comparison, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}$ packet is generated from compact interval source rows, finite fan-bar depth rows, DyadicRatCore tolerance ledgers, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8199.1, Definition 8199.2, and Definition 8199.3 seed the compact interval fan-bar route from finite depth and dyadic tolerance to Real and Completion readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected branches, quotient real equality, countable choice, host real comparison, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval source exposure, fan-bar depth exposure, dyadic tolerance accounting, stream-window readback, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalFanModulusSelectorUp.

Chapter 8200

A Concrete Naming Certificate for FiniteDimensionalBrouwerFixedPoint$^{\uparrow}$

Following Brouwer and Sperner, the FiniteDimensionalBrouwerFixedPoint$^{\uparrow}$ packet records the finite-dimensional fixed-point route in which a compact convex carrier, a continuous self-map, and a finite mesh certificate are displayed before any fixed-point-facing read is allowed. It consumes the Brouwer packet of chapter 2030, the finite labelled-simplex route of chapter 2003, the metric approximation route of chapter 5941, the compact finite-net discipline of chapter 129, the convex interval and cell discipline of chapter 6585, the self-map surface of chapter 130, and the terminal real-name seal of chapter 38. The packet names the finite-dimensional mesh route itself; it does not choose an exact host fixed point or assert an unrestricted classical theorem.

Definition 8200.1 (Finite-dimensional Brouwer fixed-point carrier)**.** *A* FiniteDimensionalBrouwerFixedPoint$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, V, M, S, L, A, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, read through* CompactMetric$^{\uparrow}$ *data and recording the finite radius, cover, and total-boundedness surface used by the argument.*
2. *V is the finite-dimensional convex-cell row, recording the listed coordinates, face incidences, convex-combination cells, and boundary faces inspected by the self-map route.*
3. *M is the* ContinuousMap$^{\uparrow}$ *self-map row, exposing only the displayed modulus and finite image data used on the mesh.*
4. *S is the* SpernerLemma$^{\uparrow}$ *handoff row, carrying the boundary-admissible finite labelling and the fully labelled simplex witness.*
5. *L is the* BrouwerFixedPoint$^{\uparrow}$ *mesh route row, tying the Sperner witness to the dyadic subdivision and approximate fixed-simplex readback.*
6. *A is the* BrouwerFixedPointMetric$^{\uparrow}$ *approximation row, recording metric residual bounds, mesh scale, and finite approximation evidence.*
7. *R is the* Real$^{\uparrow}$ *seal row reached only after the compact, convex, continuous-map, Sperner, Brouwer, and metric approximation rows have been displayed.*
8. *H records componentwise* hsame *transport for K, V, M, S, L, A, R and the structural rows.*

9. *C records displayed* Cont *replay for the finite route*

$$K, V, M, S, L, A \longmapsto R, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{SpernerLemma}^{\uparrow}$, $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, M, S, L, A, R, H, C, P, N$. It has no coordinate for an exact host fixed point, a selected limit, open-cover compactness, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8200.2 (Finite-dimensional Brouwer fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}$ *carrier*

$$F = (K, V, M, S, L, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, finite-dimensional convex-cell exposure through V, continuous self-map access through M, Sperner boundary and fully labelled simplex handoff through S, Brouwer mesh routing through L, metric residual readback through A, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8200.1. Its accepted rows are precisely the compact finite-net row, finite-dimensional convex-cell row, continuous-map row, Sperner handoff, Brouwer mesh route, metric approximation row, Real seal, and structural transport, replay, provenance, and naming rows.

The geometric part is finite. The compact row K gives only finite-net and radius data, and the convex row V gives only listed cells, faces, and convex-combination checks.

The map and fixed-point-facing part is routed. The self-map row M supplies the displayed modulus data used by the mesh, the Sperner row S supplies the labelled-simplex certificate, the Brouwer row L ties that certificate to the dyadic fixed-simplex route, and the metric row A records residual bounds at the displayed mesh scale.

Finally R, H, C, P, N seal and preserve the same finite packet. Since no excluded exact fixed point, selected limit, open-cover compactness principle, quotient equality, choice, trichotomy, ambient completeness, Lean verification, or bridge row occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 8200.3 (Finite-dimensional Brouwer fixed-point Sperner mesh route)**.** *Every fixed-point-facing read of an accepted* $\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{SpernerLemma}^{\uparrow},\quad \mathsf{BrouwerFixedPoint}^{\uparrow},\quad \mathsf{BrouwerFixedPointMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

Concretely, the consumer first reads the compact finite-net and finite-dimensional convex-cell rows K, V, then the continuous self-map row M, then the Sperner labelled-simplex handoff S, then the Brouwer mesh route L, then the metric residual row A, and only then the terminal Real seal R. No host fixed-point choice or ambient compactness theorem is exported by this route.

Proof. Start from an accepted carrier $F = (K, V, M, S, L, A, R, H, C, P, N)$. The rows K, V display the compact finite-net and finite-dimensional convex-cell surface on which the self-map is inspected.

The self-map information is then read through M. A consumer cannot interpret the mesh labels before this row is displayed, because the boundary labels and residual comparisons are induced by the carried continuous-map data.

The combinatorial handoff is supplied by S and L. The Sperner row exposes the fully labelled simplex, while the Brouwer row records the finite dyadic mesh route that turns this handoff into approximate fixed-simplex evidence.

The metric row A and Real seal R are downstream. They record residual bounds and real-name nonescape only after the compact, convex, map, Sperner, and Brouwer rows have already been displayed. Transport, replay, provenance, and local naming preserve the same route, so the packet exports no host fixed-point selection or unrestricted compactness principle. □

Falsifiable prediction: A $\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}$ consumer is refuted if it reaches a fixed-point-facing read without displaying compact finite-net data, finite-dimensional convex-cell rows, continuous-map modulus data, a Sperner labelled-simplex handoff, a Brouwer mesh route, metric residual bounds, Real sealing, transport, replay, provenance, and local NameCert evidence.

Independence witness: brouwer, sperner, metric, compact, convex, continuous, real: FiniteDimensionalBrouwerFi is not bijective to any listed sibling because it binds the finite-dimensional convex-cell route, finite mesh handoff, residual readback, and Real seal into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}$ packet is generated from compact finite-net rows, finite-dimensional convex-cell rows, continuous self-map rows, Sperner labelled-simplex handoff rows, Brouwer mesh-route rows, Brouwer fixed-point metric approximation rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8200.1, Definition 8200.2, and Definition 8200.3 seed the finite-dimensional Brouwer fixed-point mesh route through compact finite-net, convex-cell, continuous-map, Sperner, Brouwer, metric approximation, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalBrouwerFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an exact host fixed-point theorem, selected limits, open-cover compactness, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net routing, finite-dimensional convex-cell routing, continuous-map access, Sperner handoff exactness, Brouwer mesh routing, metric residual readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDimensionalBrouwerFixedPointUp.

Chapter 8201

A Concrete Naming Certificate for $\mathsf{DyadicConvergenceModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{DyadicConvergenceModulus}^{\uparrow}$ packet records the finite tolerance ledger by which dyadic convergence budgets are read before a terminal real-name seal is exposed. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name chapter of chapter 38. The packet names a displayed convergence-control object for Real-facing consumers; it does not assert quotient equality of real names, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8201.1 (Dyadic convergence modulus carrier)**.** *A* $\mathsf{DyadicConvergenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{dy}} = (D, S, Q, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the dyadic tolerance row, storing the requested dyadic error bound, the comparison scale, and the finite arithmetic witnesses read from* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the observations used to compare the relevant prefixes of the candidate names.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, turning the observed windows into regular rational comparison data.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, Q have been displayed.*
5. *M is the convergence modulus ledger, recording the finite index budget that makes the D-tolerance stable for every later inspected window.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, stream-window, regular readback, Real seal, modulus ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a convergence request through D, S, Q, E, M, H.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicConvergenceModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, E, M, H, C, P, N$. It has no coordinate for quotient Real equality, selected limits, countable choice, arbitrary Cauchy completeness, or a host convergence theorem detached from the displayed dyadic budget.

Theorem 8201.2 (Dyadic convergence modulus NameCert obligations)**.** *For every accepted* DyadicConvergenceMo *carrier*

$$M_{\mathrm{dy}} = (D, S, Q, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the dyadic tolerance row D before any convergence budget is consumed;*
2. *route every inspected prefix through the finite stream windows S and the regular rational readback row Q;*
3. *display the modulus ledger M as the only source of later-index stability for the requested dyadic tolerance;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after D, S, Q, M have been displayed;*
5. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
6. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states quotient Real equality, selects a limiting Real name, supplies countable choice, proves ambient $\mathsf{Real}^{\uparrow}$ *completeness, or exports Lean verification.*

Proof. Project the carrier of Definition 8201.1. Its rows are exactly the dyadic tolerance data, finite stream windows, regular readback, terminal seal, modulus ledger, transport, replay, provenance, and local naming.

Read the numerical rows first. The dyadic tolerance row D fixes the precision request; the stream row S and the regular rational row Q provide the finite observations against which that request is checked.

The convergence control is then the ledger row M. It records the finite index budget that makes the displayed tolerance stable for later inspected windows, without invoking a selected limit or an ambient completeness principle.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinates for quotient Real equality, countable choice, selected limits, ambient completeness, or Lean verification, the listed rows exhaust the local NameCert obligations. □

Theorem 8201.3 (Dyadic convergence modulus Real readback)**.** *Every Real-facing read of an accepted* $\mathsf{DyadicConvergenceModulus}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}.$$

The read exposes a finite dyadic convergence budget before the $\mathsf{Real}^{\uparrow}$ *seal is consumed. It supplies no quotient equality of real names, selected limit, countable-choice schedule, ambient Real completeness theorem, or Lean verification claim.*

Proof. Begin with the obligation theorem Definition 8201.2. It orders D, S, Q, M before the terminal seal E, so a Real-facing consumer cannot read the seal without first seeing the dyadic tolerance, finite windows, regular readback, and modulus ledger.

The displayed route is finite at each stage. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies arithmetic tolerance data, $\mathsf{StreamName}^{\uparrow}$ supplies inspected windows, and $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback before $\mathsf{Real}^{\uparrow}$ is reached.

The structural rows H, C, P, N preserve and name the same packet. They do not add quotient equality, selected limits, countable choice, ambient completeness, or Lean verification, so the Real readback remains the displayed finite dyadic convergence-control route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicConvergenceModulus}^{\uparrow}$ packet is generated from dyadic tolerance data, finite StreamName windows, RegSeqRat readback, a terminal Real seal, a convergence modulus ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8201.1, Definition 8201.2, and Definition 8201.3 seed the finite dyadic convergence-control route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicConvergenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, arbitrary convergence theorems, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window routing, RegSeqRat readback, Real seal nonescape, modulus ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.DyadicConvergenceModulusUp.

Chapter 8202

A Concrete Naming Certificate for $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ packet records the finite route by which one displayed uniform Cauchy budget feeds the Real and Completion phase. It sits over the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the located comparison route of chapter 5848. The packet names a shared Cauchy budget as displayed BHist data; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8202.1 (Bishop uniform Cauchy principle carrier)**.** *A* $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row listing the finite stream windows whose tails are inspected by the same request.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *uniform-budget row, assigning one dyadic tolerance and one tail threshold to every displayed pair of retained windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only from the window data in S and the dyadic comparisons in D.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, dyadic budget, and regular rational readback have been displayed.*
5. *H stores componentwise* hsame *transport for stream windows, dyadic budget rows, regular readbacks, Real seals, replay, provenance, and local naming.*
6. *C stores displayed* Cont *replay from a uniform Cauchy request through S, D, R, E, H and back to the same finite request surface.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of streams, selected limits, countable-choice schedules, host real equality, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 8202.2 (Bishop uniform Cauchy NameCert obligations)**.** *For every accepted* BishopUniformCauchyPrincip *carrier*

$$U = (S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}}$ *surface consists exactly of finite stream-window exposure, shared dyadic uniform-budget exposure, regular rational readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any located or uniform Cauchy consumer named by* P *must read the same finite route before it may inspect the Real seal.*

Proof. Project the carrier $U = (S, D, R, E, H, C, P, N)$. The analytic rows are precisely the stream-window row S, the shared dyadic budget row D, the regular readback row R, and the terminal Real seal row E.

Induct over the finite list of BHist-visible windows in S. The base window has only the displayed dyadic request in D to read. In the step, adding one more retained window extends the same shared threshold row; it does not add a second limit object or a new equality principle.

The row R is downstream of this finite induction, and E is downstream of R. Thus every accepted consumer named by P reaches the Real-facing boundary through the same S, D, R, E route.

The structural rows H, C, P, N preserve, replay, source, and name only those rows. Since quotient stream equality, selected limits, countable choice, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 8202.3 (Bishop uniform Cauchy RegSeqRat handoff)**.** *Every accepted* $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (S, D, R, E, H, C, P, N)$$

exports its $\mathsf{RegSeqRat}^{\uparrow}$ *readback only through the shared finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Consequently a Real and Completion consumer cannot replace R *by quotient stream equality, a selected tail limit, a countable-choice schedule, or a host real-equality comparison.*

Proof. Start with the displayed stream windows. The row S lists the only finite observations that the carrier makes available to the request.

Next read the shared dyadic uniform budget D. For each visible pair of retained windows, D supplies the single tolerance and tail threshold used by the request, so the regular rational row R is a readback from the displayed window-budget surface rather than from an external stream quotient.

The transport row H then keeps the same windows and budgets stable under hsame, and the replay row C records the same dependency order. Since the carrier has no field for selected limits, countable choice, host real equality, or ambient completeness, the handoff to $\mathsf{RegSeqRat}^{\uparrow}$ is exactly the finite route shown above. □

Theorem 8202.4 (Bishop uniform Cauchy Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of an accepted* $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows attached to the same packet. The seal boundary exports no ambient Real completeness, no selected completed real, no quotient stream equality, no countable-choice family of moduli, and no host real equality.*

Proof. Read the packet in dependency order. The stream row S fixes the finite windows, the dyadic row D fixes the shared budget for those windows, and the regular row R is the only rational readback exposed by the carrier.

The Real seal row E is available only after S, D, R have been displayed. A finite-row induction over the retained windows shows that every additional window is absorbed by the same shared budget row before the seal is read.

Finally project H, C, P, N. These rows transport, replay, source, and name the same finite route, so the Real-facing consumer cannot escape to a quotient of streams, a selected limit, countable choice, host real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ consumer reaches $\mathsf{Real}^{\uparrow}$ only after exposing finite $\mathsf{StreamName}^{\uparrow}$ windows, a shared $\mathsf{DyadicRatCore}^{\uparrow}$ uniform budget, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, uniformcauchycriterion, located-uniform-cauchy-principle: $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite stream windows, a shared dyadic uniform budget, regular rational readback, a Real seal boundary, transport, replay, provenance, and local naming in one certificate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite StreamName windows, one shared DyadicRatCore uniform budget, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8202.1, Definition 8202.2, Definition 8202.3, and Definition 8202.4 seed the finite uniform Cauchy route for the Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared dyadic budget exactness, finite-window induction, RegSeqRat readback ordering, Real seal boundary routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformCauchyPrincipleUp.

Chapter 8203

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$ packet records the finite comparison of two displayed regular-Cauchy tail budgets before either budget is read as a Real or Completion handoff. It sits after the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-budget service of chapter 906, the tail-diameter boundary of chapter 1746, and the terminal real-name seal of chapter 38. The packet gives later Cauchy-criterion arguments one local comparison object for budget alignment; it does not choose a limit, quotient streams, compare host real values, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8203.1 (Regular-Cauchy tail-budget comparison carrier)**.** *A* $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, B_0, B_1, E, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window service row whose finite tail window is shared by both budget reads;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row inspected through S;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the two budget reads;*
4. *B_0 and B_1 are the two displayed tail-budget rows over the same S, R, D window-and-tolerance surface;*
5. *E is the comparison endpoint row, recording whether the two budgets agree, one refines the other, or the packet exposes a finite obstruction to comparison;*
6. *H stores componentwise* hsame *transports for the stream, readback, dyadic, budget, endpoint, provenance, and local naming rows;*
7. *C stores displayed* Cont *replay routes for reading the shared window, reading both budgets, comparing their classifier rows, and exposing the endpoint;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected limit, quotient stream equality, host real equality, unlisted tolerance, or ambient completion theorem.

Theorem 8203.2 (Regular-Cauchy tail-budget comparison NameCert obligations)**.** *For the carrier of Definition 8203.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *source and window binding: the two budget rows B_0, B_1 read the same* $\mathsf{StreamName}^{\uparrow}$ *window S, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D;*
2. *classifier comparison: the endpoint row E records exactly the displayed comparison between the two budget classifiers;*
3. *stability: componentwise* hsame *transport through H preserves the same source, window, dyadic, budget, comparison, and endpoint rows;*
4. *ledger non-escape: any downstream consumer of the comparison factors through $S, R, D, B_0, B_1, E, H, C, P, N$;*
5. *real-seal routing: a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer can read a terminal seal only after the shared window, regular readback, dyadic tolerance, two budget rows, comparison endpoint, transport, replay, provenance, and local naming rows have been exposed.*

Proof. Carrier admission is projection from the displayed tuple $T = (S, R, D, B_0, B_1, E, H, C, P, N)$. The two budget rows are not independent tail services; they are fields of one finite packet over the same stream window, regular readback, and dyadic tolerance rows.

The classifier comparison is read from E. Its alternatives are the displayed agreement, displayed refinement in one direction, displayed refinement in the other direction, or a finite obstruction row. No host real comparison or quotient-stream equality is available to refine that endpoint.

Stability and replay are supplied by H and C. Componentwise transport keeps the two budget rows over the same source data, while the continuation rows replay the window read, the two budget reads, the endpoint comparison, and the consumer handoff in the listed order.

The ledger and seal clauses then follow from absence of extra coordinates. A $\mathsf{Real}^{\uparrow}$ or completion-facing consumer sees only the displayed route through $S, R, D, B_0, B_1, E, H, C, P, N$, so the packet cannot export a selected limit, unlisted tolerance, host equality, or ambient completeness object. □

Theorem 8203.3 (Regular-Cauchy tail-budget comparison Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read that uses an accepted* $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$ *packet factors first through the finite route*

$$S, R, D, B_0, B_1, E, H, C, P, N.$$

The route may certify that the two displayed tail budgets agree on the shared window, that one displayed budget refines the other, or that comparison stops at a finite obstruction row. It does not expose a selected real, a quotient of Cauchy streams, a host comparison of limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the source and window binding obligation from Definition 8203.2. It forces both budget reads through the same S, R, D surface before any endpoint row can be inspected.

Next read the classifier comparison obligation. The endpoint E records only the finite budget comparison over those displayed rows, so refinement or obstruction data remain tied to the two listed budget rows rather than to an external order on completed reals.

Finally use ledger non-escape and real-seal routing. The structural rows H, C, P, N transport, replay, source, and name the same finite route, and no coordinate exists for a selected limit, quotient stream, host comparison, countable choice, or ambient completeness theorem. Thus every Real or Completion consumer receives the comparison only through the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}$ packet is generated from a shared StreamName window, a RegSeqRat regular readback, a DyadicRatCore tolerance row, two displayed tail-budget rows, a comparison endpoint, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8203.1, Definition 8203.2, and Definition 8203.3 seed the finite L10 Real and Completion surface for comparing two displayed regular-Cauchy tail budgets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudgetComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected limits, quotient stream equality, host real comparison, countable choice, ambient RealUp completeness, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window binding, classifier comparison, hsame stability, ledger non-escape, Real seal routing, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailBudgetCompar

Chapter 8204

A Concrete Naming Certificate for $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on Kan extensions, the $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ packet records the finite universal-mapping surface obtained when a Cauchy completion route is read as a displayed Kan-extension comparison. It sits between the completion spine of chapter 126, chapter 301, chapter 38, chapter 423, chapter 1456, and chapter 2319, and the categorical universal surface of chapter 3927. The packet names the source, dense embedding, completion seal, extension candidate, comparison natural transformation, uniqueness ledger, handoff, transport, replay, provenance, and local NameCert rows without importing quotient completions, selected limits, host completed spaces, or an external functor category.

Definition 8204.1 (Cauchy completion Kan-extension carrier)**.** *A* $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, Q, E, L, \eta, U, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the Cauchy completion source row, recording the metric or finite-window source together with its* $\mathsf{StreamName}^{\uparrow}$ *observations.*
2. *D is the dense embedding row, admitting the source observations into the completion-facing boundary.*
3. *Q is the completion seal row, read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rather than through quotient completion equality.*
4. *E is the extension candidate row, carrying the completion-facing map that extends a finite-window consumer.*
5. *L is the Kan-extension row, presenting the same extension candidate as a displayed* $\mathsf{KanExtension}^{\uparrow}$ *mediator.*
6. *η is the comparison natural-transformation row from the source-side consumer to the composite completion-facing route.*
7. *U is the uniqueness ledger, comparing two extension mediators only after their dense-source and comparison rows have been displayed.*

8. *W is the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff row, exposing finite windows, regular rational readback, and terminal real-name sealing before any completion consumer inspects E or L.*
9. *H stores componentwise* hsame *transport for source, dense embedding, completion seal, extension candidate, Kan-extension mediator, comparison transformation, uniqueness, handoff, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a source request through $S, D, Q, E, L, \eta, U, W, H$ and back to the same finite universal-mapping surface.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionAd}$ $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{KanExtension}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionKanExtension}^{\uparrow}}$ *row.*

The carrier contains no quotient of Cauchy names, selected representative, host completed metric space, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or external functor-category object.*

Theorem 8204.2 (Cauchy completion Kan-extension NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ *carrier $K = (S, D, Q, E, L, \eta, U, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionK}}$ *surface consists exactly of carrier admission for the displayed tuple, source exposure through S, dense embedding through D, completion sealing through Q, extension-candidate exposure through E, Kan-extension mediator exposure through L, comparison natural-transformation exposure through η, uniqueness through U,* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8204.1. The rows S, D, Q give the source, dense embedding, and completion seal; the rows E, L, η, U give the extension candidate, Kan-extension mediator, comparison transformation, and uniqueness ledger.

The handoff row W fixes the constructive completion route. It exposes the finite $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before either the extension row or the Kan-extension mediator may be read as a universal object.

Finally, H, C, P, N transport, replay, source, and name exactly the same packet. Since the carrier has no coordinate for quotient completion, selected representatives, host completed spaces, countable choice, ambient completeness, or an external functor category, the listed rows exhaust the local NameCert surface. □

Theorem 8204.3 (Cauchy completion Kan-extension universal route)**.** *Every public universal-mapping read of an accepted* $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CauchyCompletionUniversalProperty}^{\uparrow} \mapsto$$

together with the dense embedding, comparison natural-transformation, uniqueness, transport, replay, provenance, and local naming rows of Definition 8204.1. The route exports no quotient completion, selected limit, host completed space, countable-choice family, ambient Real completeness, or external functor-category equality.

Proof. Begin with the finite completion handoff. The source row S exposes the finite windows, the dense row D admits them into the completion-facing boundary, and W forces the read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the completion seal Q is available.

Next read the universal data. The extension candidate E is attached to the same completion seal, and the Kan-extension row L presents that extension as a displayed mediator. The comparison row η records the source-side natural transformation, while the uniqueness ledger U compares mediators only over the same dense-source comparison.

The structural rows preserve the route. The rows H and C transport and replay the displayed data, and P, N record its provenance and local name. No carrier row remains from which a quotient completion, selected limit, host completed space, choice schedule, ambient completeness theorem, or external functor-category equality could be read. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ universal read must expose the Cauchy source, dense embedding, completion seal, extension candidate, comparison natural-transformation row, uniqueness ledger, $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff, transport, replay, provenance, and local naming rows before a Kan-extension consumer may inspect the mediator.

Independence witness: cauchycompletionfunctor, cauchy-completion-adjunction, cauchy-completion-universal-property, completionuniversalproperty, kan-extension, streamname, regseqrat, real: CauchyCompletionKa is not bijective to any listed sibling because it jointly carries the constructive completion handoff and the Kan-extension comparison-and-uniqueness surface in one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionKanExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionKanExtension}^{\uparrow}$ packet is generated from a Cauchy completion source, dense embedding, completion seal, extension candidate, Kan-extension mediator, comparison natural-transformation row, uniqueness ledger, StreamName/RegSeqRat/Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKanExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8204.1, Definition 8204.2, and Definition 8204.3 seed the finite universal-mapping surface connecting constructive Cauchy completion to the Kan-extension mediator.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKanExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, host completed spaces, countable choice, ambient Real completeness, external functor-category equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense embedding exactness, completion-seal handoff, comparison natural-transformation routing, uniqueness-ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionKanExtensionUp.

Chapter 8205

A Concrete Naming Certificate for $\mathsf{UniformLimitDerivative}^{\uparrow}$

Following Bishop and Bridges, and the standard Rudin termwise-differentiation theorem, the $\mathsf{UniformLimitDerivative}^{\uparrow}$ packet records the finite BEDC route by which a sequence of differentiable real-valued names with uniformly convergent derivative rows and one convergent base-point row yields a derivative handoff for the displayed limit. It sits after the derivative certificate of chapter 122, the regular-Cauchy uniform-limit route of chapter 6460, the continuity boundary of chapter 1748, and the integral-side neighbor of chapter 8198. The packet names only the displayed termwise-differentiation route through finite histories; it does not import open-cover compactness, quotient function equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or full calculus.

Definition 8205.1 (Uniform limit derivative carrier)**.** *A* $\mathsf{UniformLimitDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, G, U, B, R, J, H, C, P, N)$$

with the following displayed rows:

1. *F is the sequence row listing differentiable* $\mathsf{Real}^{\uparrow}$ *maps together with their common interval window and their local* $\mathsf{Derivative}^{\uparrow}$ *certificates.*
2. *G is the derivative-sequence row, recording the displayed derivative names produced componentwise from F.*
3. *U is the uniform derivative-limit row through* $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$*, carrying the shared modulus and derivative-limit readback.*
4. *B is the base-point convergence row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording that one displayed point has a regular-Cauchy value limit for the original sequence.*
5. *R is the regular-Cauchy reconstruction row that combines U and B into the limit-function readback route.*
6. *J is the derivative handoff row, exposing the derivative of the limit only after the derivative-limit and base-point rows have been displayed.*
7. *H records componentwise* hsame *transport for F, G, U, B, R, J.*
8. *C records displayed* Cont *replay through the differentiability, uniform derivative-limit, base-point convergence, regular-Cauchy reconstruction, and derivative-handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{UniformLimitDerivative}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RegularCauchyUniformLimit}^{\uparrow}$, $\mathsf{UniformLimitContinuous}^{\uparrow}$, $\mathsf{UniformLimitIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLimitDerivative}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open-cover compactness principle, quotient equality of functions, selected host limits, ambient completeness of $\mathsf{Real}^{\uparrow}$*, unrestricted integration, Taylor expansion, inverse-function reasoning, or a full calculus library.*

Theorem 8205.2 (Uniform limit derivative NameCert obligations)**.** *For the carrier of Definition 8205.1, the local* $\mathsf{NameCert}_{\mathsf{UniformLimitDerivative}^{\uparrow}}$ *surface consists exactly of sequence admission through F, componentwise derivative exposure through G, uniform derivative-limit exposure through U, base-point regular-Cauchy convergence through B, limit readback through R, derivative handoff through J, componentwise transport through H, displayed continuation replay through C, package provenance through P, local naming through N, and non-escape for open-cover compactness, quotient function equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and full calculus.*

Proof. Project the carrier rows of Definition 8205.1. The differentiable sequence, derivative sequence, uniform derivative-limit, base-point convergence, limit readback, and derivative handoff are separate displayed coordinates before the local name is consumed.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, quotient function equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a full calculus interface, the listed rows exhaust the local NameCert surface. □

Theorem 8205.3 (Uniform limit derivative Cauchy route)**.** *Every accepted* $\mathsf{UniformLimitDerivative}^{\uparrow}$ *packet exposes the termwise-differentiation read only through the finite route*

$$F \longmapsto G \longmapsto U \longmapsto B \longmapsto R \longmapsto J \longmapsto H, C, P, N.$$

The derivative handoff J is downstream of both the uniformly convergent derivative sequence and the base-point $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *convergence row; it is not available from a quotient equality of functions, a selected limiting function, ambient completeness, open-cover compactness, or an unrestricted calculus principle.*

Proof. Use the NameCert obligations of Definition 8205.2. They expose F, G, U, B, R, J as ordered carrier rows and place the derivative handoff after the uniform derivative-limit row and the base-point convergence row.

Read the Cauchy data next. The row U supplies the shared derivative-limit modulus, while B supplies the single base-point regular-Cauchy convergence needed to reconstruct the displayed limit-function route R.

Finally replay through H, C, P, N. Transport and continuation keep the same finite route visible, and provenance records the derivative, uniform-limit, continuity, integral-neighbor, regular-Cauchy, and Real packets without adding host function equality, selected limits, ambient completeness, open-cover compactness, or a full calculus endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitDerivative}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformLimitDerivative}^{\uparrow}$ packet is generated from differentiable RealUp map rows, their derivative rows, a uniform derivative-limit row, a base-point regular-Cauchy convergence row, limit readback, derivative handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformLimitDerivative}^{\uparrow}}$, seed)

Scope closed: Definition 8205.1, Definition 8205.2, and Definition 8205.3 seed the finite termwise-differentiation route over Derivative$^{\uparrow}$, RegularCauchyUniformLimit$^{\uparrow}$, UniformLimitContinuous$^{\uparrow}$, UniformLimitIntegral$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{UniformLimitDerivative}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient function equality, selected limits, ambient Real$^{\uparrow}$ completeness, unrestricted integration, Taylor expansion, inverse-function reasoning, full calculus, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-sequence exposure, uniform derivative-limit exposure, base-point regular-Cauchy convergence, derivative handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformLimitDerivativeUp.

Chapter 8206

A Concrete Naming Certificate for $\mathsf{LedgerModeClassifier}^{\uparrow}$

The $\mathsf{LedgerModeClassifier}^{\uparrow}$ packet realizes the evidence-mode taxonomy of section 9462.6 as finite NameCert data. It sits beside the hard-certificate layer of section 9462.5, the certified model-quality ledger of chapter 6625, the positive interpretability classifier of chapter 6202, and the tensor-source certificate surface of chapter 87. The carrier names evidence modes, mode completeness, hardening frontier, ledger-only residue, transport, replay, provenance, and local naming as one classifier packet rather than as a prose taxonomy.

Definition 8206.1 (Ledger-mode classifier carrier)**.** *A* $\mathsf{LedgerModeClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, C, F, R, G, H, T, P, N)$$

with the following displayed rows:

1. *E is the evidence-mode row, recording for each classifier-relevant NameCert field whether its support is empirical, paper, formal, hybrid, ledger-only, or open.*
2. *C is the mode-completeness row, asserting that every source, pattern, classifier, stability, and ledger field has a declared mode and that no accepted row remains open.*
3. *F is the hardening-frontier row, listing the hardenable rows that are not yet hardened and separating them from the non-hardenable residue.*
4. *R is the ledger-only residue row, recording the exact non-hardenable or not-yet-hardened residue that must remain visible to classifier-strength consumers.*
5. *G is the gap-refusal row, refusing any classifier-strength read when a required evidence-mode row, hardening-frontier row, or ledger-only residue row is absent.*
6. *H records componentwise* hsame *transport for evidence modes, mode completeness, hardening frontier, residue exactness, gap refusal, replay, provenance, and local naming.*
7. *T records displayed* Cont *replay from a classifier-strength request through E, C, F, R, G, H and back to the same ledger-mode surface.*
8. *P is the* Pkg *provenance over* $\mathsf{LedgerModeClassifier}^{\uparrow}$*, the hard-certificate layer, certified model quality, positive interpretability classifier novelty, Tensor NameCert,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LedgerModeClassifier}^{\uparrow}}$ *row.*
The carrier has no coordinate for a hidden hardened proof, unledgered classifier strength, open-mode acceptance, certified model quality by itself, interpretability novelty by itself, or Tensor NameCert field by itself.

Theorem 8206.2 (Ledger-mode classifier NameCert obligations)**.** *For every accepted* $\mathsf{LedgerModeClassifier}^{\uparrow}$ *carrier* $M = (E, C, F, R, G, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LedgerModeClassifier}^{\uparrow}}$ *surface consists exactly of evidence-mode exposure through* E*, mode-completeness through* C*, hardening-frontier exposure through* F*, ledger-only residue exactness through* R*, gap refusal through* G*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* T*,* Pkg *provenance through* P*, and local naming through* N*. Any classifier-strength consumer must read those rows before it may inspect ledger-only status.*

Proof. Project the carrier in Definition 8206.1. The rows E, C, F, R, G are the analytic rows: they display evidence modes, certify mode completeness, expose the hardening frontier, record the ledger-only residue, and refuse gap-bearing reads.

The structural rows H and T preserve and replay the same surface. A transported or replayed classifier-strength request therefore still carries the evidence-mode ledger and hardening-frontier exposure before ledger-only status is inspected.

The provenance and local naming rows P, N attach the packet to its philosophy and model-quality siblings. Since the carrier has no hidden hardened proof, no unledgered strength row, and no open-mode acceptance coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8206.3 (Ledger-mode classifier consumer boundary)**.** *Every accepted* $\mathsf{LedgerModeClassifier}^{\uparrow}$ *read exposes its evidence-mode ledger* E*, mode-completeness row* C*, hardening-frontier row* F*, ledger-only residue row* R*, and gap-refusal row* G *before any certified model-quality, interpretability-novelty, Tensor NameCert, or classifier-strength consumer may inspect ledger-only status. A consumer that omits any of these rows is outside the accepted ledger-mode classifier surface.*

Proof. Read the NameCert obligations. They require E, C, F, R, G as the analytic rows of every accepted packet, with H, T, P, N preserving, replaying, sourcing, and naming the same data.

The certified model-quality, interpretability-novelty, Tensor NameCert, and classifier-strength consumers named by P receive only the exported packet. If such a consumer omits E, it cannot determine evidence mode; if it omits C, it cannot determine mode completeness; if it omits F or R, it cannot separate hardenable frontier from ledger-only residue; and if it omits G, it cannot refuse a mode gap.

Thus ledger-only status is not a free-standing predicate. It is inspected only after the displayed evidence-mode, completeness, frontier, residue, and refusal rows have been read. □

Theorem 8206.4 (Ledger-mode classifier TasteGate boundary)**.** *The future Lean* `TasteGate.lean` *development for* $\mathsf{LedgerModeClassifier}^{\uparrow}$ *must provide explicit* $\mathsf{Nontrivial}\ \mathsf{LedgerModeClassifier}^{\uparrow}$ *and* $\mathsf{FieldFaithful}\ \mathsf{LedgerModeClassifier}^{\uparrow}$ *instances for the carrier of Definition 8206.1. Nontriviality must separate an accepted mode-complete ledger from a gap-bearing candidate, and field faithfulness must preserve* $E, C, F, R, G, H, T, P, N$ *as distinct rows before a classifier-strength consumer can cite a checked carrier-alignment theorem.*

Proof. Nontriviality is forced by the gap-refusal row. A packet with declared evidence modes, mode completeness, hardening-frontier exposure, and ledger-only residue exactness is not the same carrier as one whose mode or frontier row is absent.

Field faithfulness is forced by the consumer boundary. Dropping E, C, F, R, or G changes which classifier-strength reads are legal, while dropping H, T, P, N changes transport, replay, provenance, or local naming.

Therefore a checked TasteGate surface must keep the accepted and refused sides distinct and must preserve all nine displayed rows. Otherwise ledger-only status could be inspected without the finite evidence-mode ledger that defines it. □

Falsifiable prediction: Every accepted $\mathsf{LedgerModeClassifier}^{\uparrow}$ read must display its evidence-mode ledger and hardening-frontier row before any classifier-strength consumer may inspect ledger-only status.

Independence witness: hard-certificate-layer, certified-model-quality, interpretability-classifier-novelty, tensor-namecert: $\mathsf{LedgerModeClassifier}^{\uparrow}$ is not bijective to these siblings because it jointly carries mode declarations, gap refusal, hardening-frontier exposure, and ledger-only residue exactness.

Closure status for $\mathsf{NameCert}_{\mathsf{LedgerModeClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{LedgerModeClassifier}^{\uparrow}$ packet is generated from evidence-mode rows, mode completeness, a hardening-frontier ledger, ledger-only residue exactness, gap refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LedgerModeClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8206.1, Definition 8206.2, Definition 8206.3, and Definition 8206.4 seed the finite classifier surface for evidence modes, mode gaps, hardening frontier, and ledger-only residue.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LedgerModeClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim checked TasteGate instances, certified model quality, interpretability classifier novelty, Tensor NameCert completeness, formal hardening of any row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-mode exactness, mode-completeness refusal, hardening-frontier exposure, ledger-only residue exactness, nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.LedgerModeClassifierUp.

Chapter 8207

A Concrete Naming Certificate for $\mathsf{BisectionCauchy}^{\uparrow}$

Following Bishop and Bridges on constructive nested intervals and regular real names, the $\mathsf{BisectionCauchy}^{\uparrow}$ packet records the finite route by which located interval bisection data are read as a regular Cauchy rational name. It consumes the located bisection surface of chapter 1711, the closed bounded interval endpoint carrier of chapter 1591, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the textbook interval-bisection-to-regular-Cauchy handoff as finite BEDC data; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, choose an infinite branch, quotient Cauchy streams, or select a host real.

Definition 8207.1 (Bisection Cauchy carrier)**.** *A* $\mathsf{BisectionCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source interval row, including endpoint containment and the displayed starting width.*
2. *L is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *ledger whose finite stages retain a located subinterval and record the side decision used at each bisection.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint, radius, and tolerance row used to measure the retained interval diameters.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the bisection stages inspected by the downstream Cauchy request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row formed from the retained dyadic centers and their displayed error budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the interval, bisection, dyadic, stream, and regular-rational rows have been exposed.*
7. *H records componentwise* hsame *transport for the interval, bisection, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing request through I, L, D, W, R, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{BisectionCauchy}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, D, W, R, E, H, C, P, N$. It has no coordinate for a chosen infinite branch, quotient stream equality, host real equality, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8207.2 (Bisection Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{BisectionCauchy}^{\uparrow}$ *carrier*

$$B = (I, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded source interval, located bisection ledger, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *source agreement, requiring I and L to name the same displayed interval before any midpoint or retained half is read;*
3. *dyadic diameter control, requiring each retained half in L to expose its midpoint and width through D;*
4. *finite-window extraction, requiring W to list only stages already present in the located bisection ledger;*
5. *regular-readback handoff, requiring R to read retained centers with the dyadic error budgets supplied by D before E is exposed;*
6. *non-escape, excluding chosen infinite branches, quotient stream equality, host real equality, countable choice, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the local certificate.*

No NameCert obligation is stored outside $I, L, D, W, R, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 8207.1. The displayed packet provides exactly the source interval, located bisection ledger, dyadic ledger, finite windows, regular readback, terminal seal, transport, replay, provenance, and local naming rows.

Source agreement is read through I and L. The bisection ledger has no independent interval source; every retained half is a row below the closed bounded interval named by I.

Dyadic diameter control is supplied by D. Each bisection stage replaces one retained interval by a displayed half and records the midpoint and width needed for the next finite tolerance request.

The stream window W only selects finitely many already displayed stages. The regular readback R then reads the retained centers together with their dyadic error budgets, and the terminal seal E is visible only after that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for chosen infinite branches, quotient stream equality, host real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8207.3 (Bisection Cauchy RegSeqRat handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BisectionCauchy}^{\uparrow}$ *packet must read the finite route*

$$I, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the bisection data as a $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy name or pass the result to the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The consumer receives closed bounded interval data, located bisection stages, dyadic midpoint and diameter budgets, finite stream windows, regular rational readback, componentwise transport, continuation replay, provenance, and local naming. It receives no chosen infinite branch, quotient Cauchy stream, selected representative, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the source interval and bisection ledger. The row I supplies the closed bounded interval, and L supplies the finite retained halves that a completion-facing request may inspect.

Apply the NameCert obligations of Definition 8207.2. They force the retained intervals to expose their midpoint and diameter budgets through D before any stream window is consumed.

The finite observation row W selects only displayed bisection stages. The row R reads those selected centers as a regular rational Cauchy name with the dyadic budgets supplied by D, and E is the terminal real-name seal available only after that readback.

The structural rows H, C, P, N preserve the same finite route for downstream completion consumers. Because no excluded coordinate is present in the carrier, the handoff is a regular-rational name route rather than a quotient stream, selected limit, countable-choice, host equality, or ambient-completeness assertion. □

Theorem 8207.4 (Bisection Cauchy closed-interval boundary)**.** *Every accepted* $\mathsf{BisectionCauchy}^{\uparrow}$ *packet keeps the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row visible until the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff has been exported. Thus a consumer that reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal E also reads the original interval row I, the located bisection ledger L, and the dyadic diameter row D before any regular-rational or Real-facing output is available.*

Proof. Project the carrier of Definition 8207.1. The terminal seal E is downstream of the interval, bisection, dyadic, stream, and regular-rational rows.

Apply Definition 8207.3. The route to E factors through I, L, D, W, R before the structural rows package the output.

Therefore the closed bounded source interval remains part of every Real-facing read. The packet has no alternative Real output coordinate outside this route. □

Falsifiable prediction: An accepted $\mathsf{BisectionCauchy}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading $I, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, closedboundedinterval, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the joint packet containing the source interval, located bisection ledger, dyadic midpoint and diameter rows, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionCauchy}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source, a LocatedIntervalBisectionUp ledger, DyadicRatCore midpoint and diameter budgets, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8207.1, Definition 8207.2, and Definition 8207.3 seed the finite interval-bisection route from closed bounded interval and dyadic rows to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, chosen infinite branches, quotient Cauchy streams, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, dyadic diameter control, finite-window extraction, regular-readback handoff, and non-escape for BEDC.Derived.BisectionCauchyUp.

Chapter 8208

A Concrete Naming Certificate for $\mathsf{DyadicIntervalChain}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicIntervalChain}^{\uparrow}$ packet records a finite chain of dyadic interval cells whose inspected endpoints, nesting ledger, stream windows, regular rational readback, and terminal real-name seal are all displayed before any completion-facing consumer reads the route. It sits after the dyadic endpoint arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the finite interval-cover ledger of chapter 5105. The packet is a finite interval-chain route; it is not a selected limit, quotient real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8208.1 (Dyadic interval-chain carrier)**.** *A* $\mathsf{DyadicIntervalChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, N, W, Q, R, F, H, C, P, L)$$

with the following displayed rows:

1. *D is the finite list of dyadic interval-cell rows, each row naming lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints and its mesh scale.*
2. *N is the nesting ledger, recording adjacent containment, endpoint compatibility, and the inspected width budget for the chain.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the finite stage numbers and endpoint observations used by the packet.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed endpoint and width observations.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after D, N, W, Q have been replayed.*
6. *F is the finite interval-cover handoff row connecting the same cells to the finite cover surface of* $\mathsf{FiniteIntervalCover}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for dyadic cells, nesting rows, stream windows, regular readback, the Real seal, the finite-cover handoff, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an interval-chain request through D, N, W, Q, R, F, H.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalChain}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FiniteIntervalCover}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. L is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalChain}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, N, W, Q, R, F, H, C, P, L$. It has no coordinate for a selected nested-interval limit, quotient equality of real names, countable choice, arbitrary open covers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8208.2 (Dyadic interval-chain NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalChain}^{\uparrow}$ *carrier*

$$I = (D, N, W, Q, R, F, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalChain}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic endpoint exposure, adjacent nesting exactness, finite window scheduling, regular rational readback, terminal Real sealing, finite interval-cover handoff, componentwise transport, displayed replay, provenance, and local naming. Every consumer must expose the dyadic rows D, the nesting ledger N, and the finite windows W before it may read Q, R, or F.*

Proof. Project the carrier of Definition 8208.1. The admitted object is exactly the ten-row packet $I = (D, N, W, Q, R, F, H, C, P, L)$, so the local name can range only over the displayed dyadic cells, nesting ledger, window row, readback row, Real seal, finite-cover handoff, transport, replay, provenance, and naming rows.

The endpoint and nesting obligations are read from D and N. The dyadic endpoint arithmetic is supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ rows, while the nesting ledger records only adjacent containment and inspected width data.

The downstream reads are ordered by the carrier. The finite windows W expose the observations used by the regular rational readback Q, the terminal seal R is reached after those rows, and the finite-cover handoff F uses the same displayed cells.

Since no other coordinate is present, the packet exports no selected limit, quotient real equality, countable choice, arbitrary open-cover compactness, or ambient completeness theorem. □

Theorem 8208.3 (Dyadic interval-chain nesting handoff)**.** *Every accepted* $\mathsf{DyadicIntervalChain}^{\uparrow}$ *packet hands a finite nested dyadic window to later* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers only through the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{FiniteIntervalCover}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff supplies displayed dyadic cells, adjacent containment, finite observation windows, regular rational readback, and a finite interval-cover row; it does not supply a selected limit point, quotient equality of real names, countable choice, arbitrary open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier $I = (D, N, W, Q, R, F, H, C, P, L)$. The cells D and nesting ledger N are the only source of interval-chain data, and both are dyadic rows.

The consumer then reads finite windows through W and regular rational observations through Q. The terminal seal R is downstream of those rows, while F records the finite interval-cover handoff for the same cells.

Transport and replay through H and C preserve this ordering, and P, L package and name the same packet. Therefore the handoff is the displayed finite route through the sibling rows and cannot be read as a selected limit, quotient equality, choice principle, open-cover compactness theorem, or ambient completeness theorem. □

Falsifiable prediction: Every $\mathsf{DyadicIntervalChain}^{\uparrow}$ Real-facing read exposes dyadic interval cells, adjacent nesting, finite StreamName windows, RegSeqRat readback, a finite interval-cover handoff, componentwise transport, displayed replay, provenance, and local naming before the terminal Real seal is visible.

Independence witness: dyadicratcore, streamname, regseqrat, real, finite-interval-cover: $\mathsf{DyadicIntervalChain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite dyadic cells, adjacent nesting, finite windows, regular rational readback, a Real seal, finite interval-cover handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalChain}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalChain}^{\uparrow}$ packet is generated from finite dyadic interval-cell rows, an adjacent nesting ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, a finite interval-cover handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8208.1, Definition 8208.2, and Definition 8208.3 seed the finite dyadic interval-chain route from dyadic cells and nesting rows through finite windows, regular readback, finite-cover handoff, and a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected nested-interval limits, quotient real equality, countable choice, arbitrary open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, adjacent nesting exactness, finite window scheduling, regular rational readback, Real seal nonescape, finite interval-cover handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalChainUp.

Chapter 8209

A Concrete Naming Certificate for $\mathsf{RegularCauchyOscillationBound}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and explicit moduli of convergence, the $\mathsf{RegularCauchyOscillationBound}^{\uparrow}$ packet records the finite tail-oscillation bound by which a regular rational sequence can expose a Real-facing Cauchy control row before a terminal seal is read. It sits after the tail-diameter boundary of chapter 1746, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name chapter of chapter 38. The packet gives Real and Completion routes one local L10 source row for tail oscillation bounds; it does not choose a limit, quotient Cauchy streams, compare host real values, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8209.1 (Regular-Cauchy oscillation-bound carrier)**.** *A* $\mathsf{RegularCauchyOscillationBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, R, D, T, B, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window service row that selects the inspected tail windows;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row whose values are compared on those windows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row storing the requested oscillation radius and comparison scale;*
4. *T is the sibling* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row for the same S, R, D surface;*
5. *B is the oscillation-bound ledger, recording a finite index after which every inspected pair from S has D-bounded R-oscillation;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D, T, B have been displayed;*
7. *H stores componentwise* hsame *transports for the stream, regular readback, dyadic tolerance, tail-diameter, bound, seal, provenance, and local naming rows;*
8. *C stores displayed* Cont *replay routes for reading the tail windows, comparing the regular rational values, exposing the bound ledger, and handing the result to a Real-facing consumer;*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOscillationBound}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected limit, quotient stream equality, host real equality, unlisted tolerance, countable choice, or ambient completion theorem.

Theorem 8209.2 (Regular-Cauchy oscillation-bound NameCert obligations)**.** *For the carrier of Definition 8209.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationBound}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *source binding: the bound ledger B reads the displayed* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D;*
2. *tail-diameter anchoring: the sibling* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *row T is read before any oscillation-bound consumer may inspect the terminal seal;*
3. *bound exactness: B records a finite tail index and a displayed pairwise comparison route witnessing D-bounded oscillation for the inspected tail windows;*
4. *stability: componentwise* hsame *transport through H preserves the same source, tolerance, tail-diameter, bound, seal, replay, provenance, and naming rows;*
5. *ledger non-escape: every downstream consumer of the oscillation bound factors through $S, R, D, T, B, E, H, C, P,$*
6. *real-seal routing: a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer can read the terminal seal only after the stream window, regular readback, dyadic tolerance, tail-diameter boundary, oscillation-bound ledger, transport, replay, provenance, and local naming rows have been exposed.*

Proof. Carrier admission is projection from the displayed tuple $O = (S, R, D, T, B, E, H, C, P, N)$. The source binding clause reads the stream window, regular-rational values, and dyadic tolerance before any oscillation statement is available.

The tail-diameter and bound clauses are local. The row T supplies the sibling diameter boundary for the same displayed data, while B records the finite index and pairwise comparison route that make the requested oscillation bound visible on the inspected tail.

Stability and replay are supplied by H and C. Componentwise transport keeps every row over the same source data, and the continuation routes replay the window read, rational comparison, bound exposure, and Real-facing handoff in that order.

The non-escape and seal clauses follow from absence of extra coordinates. A $\mathsf{Real}^{\uparrow}$ or completion-facing consumer sees only the listed packet rows, so the certificate cannot export a selected limit, quotient stream equality, host comparison, countable choice, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOscillationBound}^{\uparrow}$ packet is generated from a finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, CauchyTailDiameter boundary, oscillation-bound ledger, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationBound}^{\uparrow}}$, seed)

Scope closed: Definition 8209.1 and Definition 8209.2 seed the finite L10 Real and Completion surface for regular-Cauchy tail oscillation bounds.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyOscillationBound}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected limits, quotient stream equality, host real comparison, countable choice, ambient RealUp completeness, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source binding, tail-diameter anchoring, bound exactness, hsame stability, ledger non-escape, Real seal routing, replay, provenance, and local naming for BEDC.Derived.RegularCauchyOscillati

Chapter 8210

A Concrete Naming Certificate for $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ packet records the finite cofinal-modulus route by which a regular Cauchy name is read through a cofinal schedule without changing its modulus-controlled Real seal. The route sits over the dyadic tolerance carrier of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the reindexing handoff of chapter 1568.

Definition 8210.1 (RegularCauchyCofinalModulus carrier)**.** *A* $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, K, U, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *S is the source regular Cauchy data row.*
2. *K is the cofinal schedule row, recording the finite monotone route along which the source is read.*
3. *U is the modulus row that assigns a displayed request bound to the cofinal schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the modulus.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the cofinal schedule.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on those windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached after the modulus, tolerance, window, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for S, K, U, D, W, R, E and the structural rows.*
9. *C records displayed* Cont *replay of the same cofinal-modulus route.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected representative, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyCofinalModulusUp`

Theorem 8210.2 (RegularCauchyCofinalModulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ *carrier* $M = (S, K, U, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalMo}}$ *surface consists exactly of source regular Cauchy admission through* S*, cofinal schedule exposure through* K*, modulus-row exposure through* U*, dyadic tolerance exposure through* D*, StreamName windows through* W*, RegSeqRat readback through* R*, Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

No accepted read exports quotient stream equality, a selected representative, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8210.1. Its analytic route is S, K, U, D, W, R, E, and its structural route is H, C, P, N.

The source row S is read only through the cofinal schedule K. The modulus row U then binds each displayed request to the same cofinal route, while the dyadic tolerance row D records the requested error surface.

The window and readback rows W and R expose the finite StreamName and RegSeqRat data before the terminal Real seal E is consumed. Transport, replay, provenance, and local naming preserve this same packet through H, C, P, N.

Since the carrier contains no row for quotient stream equality, representatives, choice, host equality, trichotomy, or ambient completeness, the listed rows exhaust the NameCert obligation surface. □

Theorem 8210.3 (RegularCauchyCofinalModulus Real handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{CauchyReindexing}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows K, D, W, R, E *before the* $\mathsf{Real}^{\uparrow}$ *seal is exported. The cofinal schedule preserves the modulus-controlled seal only as this finite packet route; it is not a quotient of regular Cauchy names, a selected host real, a countable-choice schedule, or an ambient completeness theorem.*

Proof. Start with the obligation surface of Definition 8210.2. It orders the source, cofinal schedule, modulus, dyadic tolerance, window, readback, and seal rows before any Real-facing consumer can inspect the packet.

Read the cofinal schedule row K as the handoff to chapter 1568. The row does not create a second stream; it only records the displayed cofinal route along which the source regular Cauchy data are inspected.

The dyadic, stream, and regular-readback rows D, W, R then route the same requests through chapter 291, chapter 126, and chapter 301. Only after those rows have been replayed may the consumer read the terminal $\mathsf{Real}^{\uparrow}$ seal E of chapter 38.

The structural rows H, C, P, N preserve the same finite route. Because no row supplies quotient equality, selected representatives, choice, host equality, or ambient completeness, the Real handoff is exactly the displayed cofinal-modulus route. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ Real-facing read exposes source regular Cauchy data, a cofinal schedule, a modulus row, DyadicRatCore tolerance, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local NameCert before the Real seal is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchyreindexing, regularcauchycofinalmodulus: $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source regular Cauchy row, cofinal schedule, modulus row, tolerance, windows, readback, Real seal, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCofinalModulus}^{\uparrow}$ packet is generated from source regular Cauchy data, a cofinal schedule, a modulus row, DyadicRatCore tolerance, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8210.1, Definition 8210.2, and Definition 8210.3 seed the finite cofinal-modulus route over the L10 Real path.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of regular Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal schedule exposure, modulus preservation, dyadic tolerance exactness, window readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCofinalModulusUp.

Chapter 8211

A Concrete Naming Certificate for NavierStokesFiniteEnergy$^{\uparrow}$

Following Leray and Temam, the NavierStokesFiniteEnergy$^{\uparrow}$ packet records the finite-energy weak-solution surface for the incompressible Navier-Stokes equations as BEDC carrier data. It sits over the PDE relation row of chapter 194, the finite-window comparison discipline of chapter 193, the Sobolev weak-velocity packet of chapter 440, the tagged energy-readback surface of chapter 1526, and the boundary-trace accounting of chapter 398. The packet names only the displayed finite-energy route; it does not assert global smoothness, uniqueness, Clay Millennium problem resolution, measure completion, classical weak compactness, or a host pressure quotient.

Definition 8211.1 (Navier-Stokes finite-energy carrier)**.** *A* NavierStokesFiniteEnergy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (D, V, R, E, W, B, H, C, P, M)$$

with the following displayed rows:

1. *D is the* Pde$^{\uparrow}$ *source row naming the incompressible Navier-Stokes relation, including the displayed velocity, pressure, viscosity, forcing, divergence-free, and initial-data slots.*
2. *V is the* Sobolev$^{\uparrow}$ *weak velocity-field row, read as a finite Sobolev packet for the velocity and its weak-gradient observations.*
3. *R is the pressure and residual row, recording the displayed pressure term, divergence constraint residual, forcing residual, and weak-form defect ledger without quotienting pressure by a host constant.*
4. *E is the energy-inequality ledger, read through* RiemannIntegral$^{\uparrow}$ *rows for kinetic-energy and dissipation windows.*
5. *W is the finite time-window readback row. It names the time interval, endpoint observations, and ODE-facing comparison route used only as a finite evolution window.*
6. *B is the boundary row, recording no-slip, periodic, compact-support, or explicitly listed boundary data through bounded-variation and trace-facing ledgers.*
7. *H records componentwise* hsame *transport for the PDE source, weak velocity, pressure-residual, energy, time-window, boundary, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay through D, V, R, E, W, B before a consumer reads the finite-energy packet.*

9. *P is the* Pkg *provenance over* $\mathsf{NavierStokesFiniteEnergy}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *M is the local* $\mathsf{NameCert}_{\mathsf{NavierStokesFiniteEnergy}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for a smooth global solution, a uniqueness theorem, a measure-completed function space, a compactness extraction theorem, or a quotient pressure representative.

Theorem 8211.2 (Navier-Stokes finite-energy NameCert obligations)**.** *For every accepted* NavierStokesFiniteEnergy *carrier $N = (D, V, R, E, W, B, H, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{NavierStokesFiniteEnergy}^{\uparrow}}$ *surface consists exactly of PDE-source admission, Sobolev weak-velocity exposure, pressure and residual ledger visibility, energy-inequality readback, finite time-window readback, boundary-row exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission projects the displayed packet of Definition 8211.1. The analytic source is the PDE row D, while the weak velocity is read through the Sobolev row V and the pressure-residual ledger R.

The energy and time-window obligations are separate rows. The energy inequality is read through E and its Riemann-integral provenance, while W records only the finite evolution window and endpoint observations.

The boundary and structural obligations are then visible through B, H, C, P, M. These rows transport, replay, source, and name the displayed carrier without adding smoothness, uniqueness, compactness extraction, measure completion, or pressure quotient data. □

Theorem 8211.3 (Navier-Stokes finite-energy nonescape)**.** *Every finite-energy read exported by an accepted* $\mathsf{NavierStokesFiniteEnergy}^{\uparrow}$ *carrier factors through*

$$D, V, R, E, W, B, H, C, P, M.$$

It may expose the PDE source, Sobolev weak velocity, pressure-residual ledger, energy inequality, finite time-window, boundary row, transport, replay, provenance, and local NameCert *evidence, but it cannot export global smoothness, uniqueness, Clay Millennium problem resolution, measure completion, classical weak compactness, or a host pressure quotient.*

Proof. Start with the route displayed in Definition 8211.1. Each consumer-facing read begins at the PDE row D and then consumes the weak velocity V, pressure-residual ledger R, energy ledger E, finite time-window W, and boundary row B.

The structural rows H, C, P, M preserve the same finite route under transport, replay, provenance, and local naming. They do not introduce a second analytic source or a hidden host object.

The excluded claims are absent by carrier inspection. No row names global smoothness, uniqueness, the Clay Millennium problem, measure completion, classical weak compactness, or a quotient pressure representative, so those claims cannot be recovered from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{NavierStokesFiniteEnergy}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NavierStokesFiniteEnergy}^{\uparrow}}$, seed)

Scope closed: Definition 8211.1, Definition 8211.2, and Definition 8211.3 seed the finite-energy PDE carrier, NameCert obligation surface, and nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NavierStokesFiniteEnergy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, global smoothness, uniqueness, Clay Millennium problem resolution, measure completion, classical weak compactness, host pressure quotients, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, PDE-source exactness, Sobolev weak-velocity exposure, pressure-residual visibility, energy-window readback, boundary-row exposure, transport, replay, provenance, and local naming for BEDC.Derived.NavierStokesFiniteEnergyUp.

Constructive story: A NavierStokesFiniteEnergy$^{\uparrow}$ packet is generated from PDE source rows, Sobolev weak velocity rows, pressure and residual rows, energy inequality ledgers, finite time-window readbacks, boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8212

A Concrete Naming Certificate for $\mathsf{PriestleySpace}^{\uparrow}$

Following Priestley and Davey–Priestley, the $\mathsf{PriestleySpace}^{\uparrow}$ packet records the ordered topological carrier used by Priestley duality as finite BEDC NameCert data. It sits after the meet–join order surface of chapter 55, the bounded endpoint rows of chapter 6501, the indexed-open carrier of chapter 88, the clopen ledger of chapter 261, and the duality-theorem packet of chapter 5166. The chapter separates the ordered topological carrier from the representation theorem: it names compact ordered clopen-upset observations, not selected prime filters, an ultrafilter construction, a powerset topology, quotient lattice equality, or a choice principle.

Definition 8212.1 (Priestley space carrier)**.** *A* $\mathsf{PriestleySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, T, K, U, D, H, C, P, N)$$

with the following displayed rows:

1. *O is the bounded distributive-order row, read through* $\mathsf{Lattice}^{\uparrow}$ *meet–join observations and* $\mathsf{BoundedLattice}^{\uparrow}$ *endpoint rows.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row, listing the finite open-neighbourhood observations available to the carrier.*
3. *K is the Stone-facing clopen ledger inherited from chapter 261, restricted to displayed clopen rows rather than host subsets.*
4. *U is the clopen-upset row, recording which displayed clopen rows are upward closed for the order observations in O.*
5. *D is the Priestley separation row: for displayed ordered observations whose order comparison is separated, D names a clopen upset that contains the upper observation and excludes the lower observation.*
6. *H records componentwise* hsame *transport for O, T, K, U, D, P, N.*
7. *C records displayed* Cont *replay from an order/topology query through the clopen-upset and separation rows.*
8. *P is the* Pkg *provenance over* $\mathsf{PriestleySpace}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{BoundedLattice}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StoneDuality}^{\uparrow}$, $\mathsf{PriestleyDuality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PriestleySpace}^{\uparrow}}$ *row.*
The local classifier compares accepted packets only through $O, T, K, U, D, H, C, P, N$.

Theorem 8212.2 (Priestley space NameCert obligations)**.** *For every accepted* $\mathsf{PriestleySpace}^{\uparrow}$ *carrier $S = (O, T, K, U, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PriestleySpace}^{\uparrow}}$ *surface consists exactly of bounded lattice order admission through O, topology admission through T, Stone clopen readback through K, clopen-upset classification through U, Priestley separation through D, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N. The certificate exports no ultrafilter construction, selected prime filters, powerset topology, quotient lattice equality, host spectral space, or choice principle.*

Proof. Project the carrier of Definition 8212.1. The rows O, T, K, U, D are the only order, topology, clopen, upset, and separation coordinates; H, C, P, N only transport, replay, source, and name those rows.

The order rows are read through the lattice and bounded-lattice siblings, while the topological rows are read through the topology sibling. The clopen ledger K is supplied by the Stone-duality interface as displayed clopen data, and U restricts those rows to the upward-closed observations visible from O.

The separation row D then records only the finite Priestley condition exposed by the packet. Componentwise transport and displayed continuation preserve the same finite route under H and C, and P, N package and name it.

Since no listed coordinate stores ultrafilters, selected prime filters, powerset opens, quotient lattice equality, host spectral points, or a choice principle, those objects are outside the local NameCert surface. □

Theorem 8212.3 (Priestley space Stone-order handoff)**.** *Every Stone-duality or Priestley-duality consumer of an accepted* $\mathsf{PriestleySpace}^{\uparrow}$ *carrier must read the ordered topological handoff through*

$$\mathsf{Lattice}^{\uparrow},\ \mathsf{BoundedLattice}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow},\ \mathsf{StoneDuality}^{\uparrow} \longrightarrow O, T, K, U, D, H, C, P, N.$$

The handoff exposes a bounded order row, indexed-open topology row, Stone clopen ledger, clopen-upset classifier, and finite separation row before any duality theorem packet may consume the carrier. It does not assert a host Priestley representation theorem, an ultrafilter construction, selected prime filters, a powerset topology, quotient lattice equality, or choice.

Proof. Begin with the order side. The rows O and D make the bounded distributive order and finite separation requests visible before a consumer can treat a clopen row as an order-sensitive observation.

Read the topological side next. The row T supplies indexed opens, K supplies the Stone-facing clopen ledger, and U records the clopen-upset restriction for the displayed order. These rows are the ordered topological carrier surface, not the full duality theorem.

Finally use the structural rows H, C, P, N. They transport, replay, source, and name the same handoff for Stone-duality and Priestley-duality consumers. Because the carrier contains no ultrafilter, selected prime-filter, powerset-topology, quotient-equality, host-representation, or choice coordinate, the downstream handoff remains finite BEDC data. □

Falsifiable prediction: A Priestley-duality consumer that reads a clopen-upset row without first exposing the bounded order row and topology row is not a $\mathsf{PriestleySpace}^{\uparrow}$ carrier read.

Independence witness: Nearest siblings: lattice, bounded-lattice, topology, stoneduality, priestleyduality. The carrier is not bijective to LatticeUp or BoundedLatticeUp because it adds indexed-open, clopen, and separation rows; not to TopologyUp because it requires bounded order and clopen-upset rows; not to StoneDualityUp because it records an order-sensitive separation row; not to PriestleyDualityUp because it supplies the ordered topological carrier rather than the representation theorem packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PriestleySpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PriestleySpace}^{\uparrow}$ packet is generated from bounded distributive-order rows, indexed topology observations, Stone clopen rows, clopen-upset classification, Priestley separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PriestleySpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8212.1, Definition 8212.2, Definition 8212.3, the falsifiable prediction, and the independence witness seed the ordered topological carrier surface for Priestley-duality consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PriestleySpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host Priestley representation, ultrafilter construction, selected prime filters, powerset topology, quotient lattice equality, Lean verification, choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded order admission, indexed-open topology admission, Stone clopen readback, clopen-upset classification, Priestley separation, and local NameCert exhaustion for BEDC.Derived.PriestleySpaceUp.

Chapter 8213

A Concrete Naming Certificate for $\mathsf{GroupCohomology}^{\uparrow}$

Following Brown and Cartan–Eilenberg on group cohomology, the $\mathsf{GroupCohomology}^{\uparrow}$ packet names the finite BEDC surface by which a group action on abelian or module coefficients is read as cocycle and boundary data before any cohomology group is inspected. It depends on the action carrier of chapter 41, the coefficient rows of chapter 42 and chapter 46, and the cocycle-boundary interface of chapter 99. The packet does not choose representatives in a quotient, import derived-functor machinery, assume projective resolutions, invoke countable choice, or claim an ambient exact-category theorem.

Definition 8213.1 (Group cohomology carrier)**.** *A* $\mathsf{GroupCohomology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, M, W, D, Z, B, T, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Group}^{\uparrow}$ *action row naming the finite group-operation surface and its action on coefficient reads.*
2. *M is the coefficient row, read either through an* $\mathsf{AbGroup}^{\uparrow}$ *additive surface or through a compatible* $\mathsf{Module}^{\uparrow}$ *row.*
3. *W is the finite cochain-window ledger listing the displayed cochain arities and argument windows being inspected.*
4. *D is the coboundary row, recording the finite alternating action-and-face route from one cochain window to the next.*
5. *Z is the cocycle row, asserting that the displayed coboundary route sends the inspected cochain window to the zero coefficient row.*
6. *B is the boundary row, recording the finite predecessor window whose coboundary route yields a displayed cocycle.*
7. *T stores componentwise* hsame *transport for the group action, coefficient, cochain window, coboundary, cocycle, boundary, replay, provenance, and naming rows.*
8. *C stores displayed* Cont *replay for reading the action, coefficient module, cochain window, coboundary row, cocycle row, and boundary row in that order.*

9. *P is the* Pkg *provenance over* $\mathsf{GroupCohomology}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GroupCohomology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient representatives, selected resolutions, derived functors, global exactness, countable choice, or an ambient category of modules.

Theorem 8213.2 (Group cohomology NameCert obligations)**.** *For the carrier of Definition 8213.1, the local* $\mathsf{NameCert}_{\mathsf{GroupCohomology}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *action exposure: the* $\mathsf{Group}^{\uparrow}$ *row A is read before any cochain, cocycle, or boundary row is available;*
2. *coefficient exposure: the* $\mathsf{AbGroup}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *coefficient row M is displayed before the coboundary route is inspected;*
3. *cochain-window exactness: the ledger W lists the finite argument windows and arities used by the cochain read;*
4. *coboundary routing: the row D displays the finite alternating action-and-face route between the relevant cochain windows;*
5. *cocycle admission: the row Z is admitted only after D has exposed a zero coboundary read in the coefficient surface;*
6. *boundary admission: the row B is admitted only through a displayed predecessor cochain window and its coboundary route;*
7. *stability and replay: T and C preserve the same action, coefficient, cochain-window, coboundary, cocycle, boundary, provenance, and naming rows under* hsame *transport and* Cont *replay;*
8. *non-escape: P and N expose the package provenance and local name without adding quotient representatives, selected resolutions, derived-functor objects, global exactness, countable choice, or ambient module-category data.*

Proof. Carrier admission is projection from the displayed tuple $G = (A, M, W, D, Z, B, T, C, P, N)$. The first three obligations expose the action, coefficient row, and finite cochain-window ledger before any cohomology read is possible.

The coboundary and cocycle obligations are local to the displayed windows. The row D supplies the alternating action-and-face route, and Z records zero coboundary only after that route has been read in the coefficient surface.

The boundary obligation is likewise finite. The row B names a predecessor window and its displayed coboundary route; it does not select a representative in a quotient or appeal to an external resolution.

Finally T, C, P, N preserve, replay, source, and name the same rows. Since the carrier has no coordinate for quotient representatives, selected resolutions, derived functors, global exactness, countable choice, or ambient module-category data, the listed obligations exhaust the local NameCert surface. □

Theorem 8213.3 (Group cohomology cocycle-boundary route)**.** *Every accepted consumer of a* $\mathsf{GroupCohomology}^{\uparrow}$ *packet must expose the group action row A, coefficient row M, cochain-window ledger W, cocycle row Z, and boundary row B before any cohomology read is available. In particular, no consumer can inspect a cohomology class by bypassing the displayed* $\mathsf{Group}^{\uparrow}$ *action,*

$\mathsf{AbGroup}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *coefficient surface, finite cochain windows, coboundary route, cocycle row, and boundary row.*

Proof. Project the carrier of Definition 8213.1. Its analytic rows appear as the action A, coefficient M, cochain window W, coboundary route D, cocycle row Z, and boundary row B before the transport, replay, provenance, and naming rows are read.

Apply the NameCert obligations of Definition 8213.2. Action exposure and coefficient exposure make A and M prerequisites for every downstream read, while cochain-window exactness and coboundary routing place W and D before either Z or B.

Cocycle admission and boundary admission then force the two cohomology-facing rows to be displayed. The replay and non-escape clauses keep the consumer inside $A, M, W, D, Z, B, T, C, P, N$, so there is no route through quotient representatives, selected resolutions, derived functors, global exactness, countable choice, or ambient module-category data. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroupCohomology}^{\uparrow}}$.

Constructive story: A $\mathsf{GroupCohomology}^{\uparrow}$ packet is generated from a GroupUp action row, AbGroupUp or ModuleUp coefficient row, finite cochain-window ledger, coboundary route, cocycle row, boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroupCohomology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8213.1, Definition 8213.2, and Definition 8213.3 seed the finite NameCert surface for group cohomology reads over displayed action and coefficient rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroupCohomology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient representatives, selected resolutions, derived functors, global exactness, countable choice, ambient module-category data, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action exposure, coefficient exposure, cochain-window exactness, coboundary routing, cocycle admission, boundary admission, transport, replay, provenance, and local naming for BEDC.Derived.GroupCohomologyUp.

Chapter 8214

A Concrete Naming Certificate for $\mathsf{DavenportConstant}^{\uparrow}$

Following Davenport's zero-sum constant tradition in finite abelian groups, the $\mathsf{DavenportConstant}^{\uparrow}$ packet records the finite BEDC surface by which a displayed sequence over a finite abelian group is inspected for a nonempty zero-sum subsequence or for a bounded refusal ledger. It sits after the group operation rows of chapter 41, the commutative group source of chapter 42, the unary length and index budget of chapter 28, and the finite sequence spine of chapter 36. The packet names only the displayed finite additive-combinatorics data; it is not an ambient pigeonhole principle, a quotient group theorem, a choice principle over subsequences, or a host-side extremal existence oracle.

Definition 8214.1 (Davenport constant carrier)**.** *A* $\mathsf{DavenportConstant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (G, A, L, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{AbGroup}^{\uparrow}$ *source, read through the* $\mathsf{Group}^{\uparrow}$ *operation, inverse, identity, associativity, commutativity, and equality rows.*
2. *A is the finite sequence row over the displayed group source. It is a* $\mathsf{List}^{\uparrow}$ *spine whose entries are group elements, not an unordered host multiset.*
3. *L is the* $\mathsf{Nat}^{\uparrow}$ *length row, recording the inspected sequence length, the bounded index set, and the threshold comparison used by the Davenport-constant request.*
4. *S is the partial-sum ledger. It records the finite prefix sums and the contiguous or subsequence index witnesses inspected by the packet.*
5. *W is the zero-sum witness or refusal row. In the witness case it names a nonempty displayed sublist and a proof that its transported group sum is the identity; in the refusal case it records the finite inspected obstruction rows available below the threshold.*
6. *R records componentwise* hsame *transport for the group source, sequence spine, length row, partial sums, witness or refusal row, replay, provenance, and local naming.*
7. *H records classifier comparison only through the visible group operation, list entries, length budget, partial-sum ledger, and zero-sum witness or refusal row.*
8. *C records displayed* Cont *replay from a zero-sum request through G, A, L, S, W, R, H.*

9. *P is the* Pkg *provenance over* $\mathsf{DavenportConstant}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DavenportConstant}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, A, L, S, W, R, H, C, P, N$. It has no coordinate for an external enumeration of all subsequences, a quotient group equality, an unbounded choice of witnesses, or a host theorem asserting the optimal numerical value of the Davenport constant.

Lean4 ✓ `BEDC.Derived.DavenportConstantUp.DavenportConstantTasteGate single carrier alignment`

Theorem 8214.2 (Davenport constant NameCert obligations)**.** *For every accepted* $\mathsf{DavenportConstant}^{\uparrow}$ *carrier*

$$Z = (G, A, L, S, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DavenportConstant}^{\uparrow}}$ *surface consists exactly of finite abelian group source admission, displayed sequence membership, unary length budgeting, partial-sum ledger construction, zero-sum witness or refusal exposure, componentwise* hsame *transport, classifier comparison through displayed rows,* Cont *replay, provenance, and local naming.*

Proof. Project the accepted carrier of Definition 8214.1. Its coordinates are precisely the finite abelian group row, the list spine, the unary length budget, the partial-sum ledger, the witness-or-refusal row, transport, classifier, replay, provenance, and local naming.

The algebraic obligations are inherited in order. The $\mathsf{AbGroup}^{\uparrow}$ source supplies the commutative operation and identity rows through the underlying $\mathsf{Group}^{\uparrow}$ packet, while the $\mathsf{List}^{\uparrow}$ spine supplies only displayed finite entries. The $\mathsf{Nat}^{\uparrow}$ row bounds the inspected indices and fixes the length comparison before any zero-sum read is attempted.

The additive-combinatorics obligation is local to the ledger. The partial sums in S are finite sums over the displayed entries, and W either exposes a nonempty selected sublist with transported sum equal to the identity or records the finite refusal rows available below the displayed threshold.

Transport, classifier comparison, replay, provenance, and naming preserve the same rows. Since the carrier stores no external subsequence enumeration, quotient equality, choice functional, or host-side optimality theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 8214.3 (Davenport constant zero-sum handoff)**.** *Every downstream zero-sum consumer of an accepted* $\mathsf{DavenportConstant}^{\uparrow}$ *carrier reads the finite additive route*

$$\mathsf{AbGroup}^{\uparrow},\ \mathsf{Group}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{List}^{\uparrow} \longmapsto \mathsf{DavenportConstant}^{\uparrow}.$$

The handoff supplies a displayed finite abelian group source, a finite sequence spine, a unary length budget, a partial-sum ledger, and either a nonempty zero-sum sublist witness or a bounded refusal row. It does not supply quotient group equality, a global subsequence-choice functional, a host extremal theorem, or ambient completeness data.

Proof. Start from an accepted carrier $Z = (G, A, L, S, W, R, H, C, P, N)$. A consumer can read group addition only through G, sequence data only through A, and the inspected bound only through L.

The zero-sum decision surface is then read from S and W. The ledger S lists the finite partial sums and inspected index data, while W names the displayed witness or the bounded refusal rows. No later coordinate can introduce an unlisted subsequence or an unbounded selector.

Finally R, H, C, P, N transport, compare, replay, source, and name the same finite packet. Thus the handoff is exactly the displayed zero-sum route through the finite abelian group, length, list, and ledger rows, and not an ambient theorem outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{DavenportConstant}^{\uparrow}$ consumer that exports a zero-sum conclusion while omitting one of the displayed finite abelian group source, sequence spine, unary length, partial-sum ledger, witness-or-refusal, transport, classifier, replay, provenance, or local naming rows refutes the packet.

Independence witness: abgroup, group, nat, list: $\mathsf{DavenportConstant}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite abelian group source, a sequence spine, a unary length budget, a partial-sum ledger, a zero-sum witness or refusal row, componentwise hsame transport, classifier comparison, Cont replay, Pkg provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{DavenportConstant}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DavenportConstant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8214.1, Definition 8214.2, and Definition 8214.3 seed the finite zero-sum route over abelian group rows, list spines, unary length budgets, partial-sum ledgers, witness-or-refusal rows, transport, classifier comparison, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DavenportConstant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, a host-side optimal Davenport constant theorem, global subsequence choice, ambient pigeonhole principles, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite abelian group source exposure, sequence spine well-formedness, unary length budgeting, partial-sum ledger exactness, zero-sum witness or refusal stability, transport, classifier comparison, replay, provenance, and local naming for BEDC.Derived.DavenportConstantUp.

Constructive story: A $\mathsf{DavenportConstant}^{\uparrow}$ packet is generated from finite abelian group rows, a displayed sequence spine, a unary length budget, a partial-sum ledger, a zero-sum witness or bounded refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8215

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}$

Following Bishop and Bridges on regulated functions and regular Cauchy real names, the RegulatedCauchyStepApprc packet records the finite approximation schedule by which displayed step-function windows over a regulated source produce regular-rational names for Real-facing consumers. It sits after the regulated-function carrier of chapter 1906, the dyadic step-function surface of chapter 428, the uniform-limit route of chapter 463, the dyadic mesh gauge of chapter 7496, the regular rational readback of chapter 301, and the terminal real-name chapter of chapter 38. The packet gives Real and Completion routes one finite step-approximation bridge; it does not quotient functions, select a limit object, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8215.1 (Regulated Cauchy step approximation carrier)**.** *A* $\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, S, U, G, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row whose one-sided oscillation data are exposed on the displayed interval windows.*
2. *S is the* $\mathsf{StepFunction}^{\uparrow}$ *approximation ledger, listing finitely many step rows and their domains of constancy.*
3. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff binding the step rows to the regulated source at the requested tolerance.*
4. *G is the* $\mathsf{DyadicMeshGauge}^{\uparrow}$ *mesh row that supplies the finite dyadic subdivision used by the step ledger.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the sampled approximation values and tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, S, U, G, Q have been displayed.*
7. *H stores componentwise* hsame *transport for the regulated source, step ledger, uniform-limit handoff, dyadic mesh, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a regulated-function request through the finite mesh, step approximation ledger, regular readback, and Real-facing handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{DyadicMeshGauge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, S, U, G, Q, E, H, C, P, N$. No quotient function equality, selected limit, countable-choice schedule, or ambient completeness coordinate is part of the carrier.

Theorem 8215.2 (Regulated Cauchy step approximation NameCert obligations)**.** *For an accepted* $\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}$ *carrier $A = (R, S, U, G, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}}$ *surface consists exactly of carrier exposure for the regulated source, step-function ledger, uniform-limit handoff, dyadic mesh gauge, regular-rational readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows. Every accepted read first fixes the displayed regulated source R, then chooses only the finite mesh cells in G, then reads the step ledger S and uniform-limit handoff U, and finally exports the regular-rational and Real-facing rows Q, E, H, C, P, N.*

Proof. Project the carrier of Definition 8215.1. The rows $R, S, U, G, Q, E, H, C, P, N$ are the only displayed coordinates of the packet.

The source and mesh obligations read R and G before any step approximation is consumed. The step ledger S is therefore tied to the displayed dyadic cells rather than to an external partition choice.

The uniform-limit and regular-readback obligations use U and Q. They attach each tolerance request to the finite step ledger and regular rational tail data before the terminal Real seal E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no row for quotient function equality, a selected limit object, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those readings can enter the local certificate. □

Theorem 8215.3 (Regulated Cauchy step approximation RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reading an accepted* $\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}$ *packet factors through*

$$R \longmapsto G \longmapsto S \longmapsto U \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The consumer receives a regular-rational name assembled from finitely displayed step approximations and dyadic mesh gauges. It receives no quotient equality of regulated functions, selected uniform limit, countable-choice sequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 8215.2. They force a consumer to read the regulated source R, the dyadic mesh row G, the step ledger S, and the uniform-limit handoff U before Q can be inspected.

The row Q is the only regular-rational readback coordinate. Thus a $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives the tail budget and sampled approximation values from the finite step schedule, not from an external limiting object.

The terminal seal E is downstream of Q, while H, C, P, N preserve the same route under transport, replay, provenance, and local naming. The route therefore cannot export quotient function equality, a selected limit, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepApproximation}^{\uparrow}}$.

Constructive story: A RegulatedCauchyStepApproximation$^{\uparrow}$ packet is generated from a RegulatedFunction source, StepFunction approximation ledger, UniformLimit handoff, DyadicMeshGauge mesh row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{RegulatedCauchyStepApproximation}\uparrow}$, seed)

Scope closed: Definition 8215.1, Definition 8215.2, and Definition 8215.3 seed the finite regulated step-approximation route from displayed regulated functions and step ledgers to regular-rational and Real-facing handoffs.

Formal verification: FormalStatus(NameCert$_{\text{RegulatedCauchyStepApproximation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient function equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, regulated-source binding, mesh exactness, step-ledger exposure, uniform-limit handoff, RegSeqRat readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedCauchyStepApproximationUp.

Chapter 8216

A Concrete Naming Certificate for $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$

The $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ packet realizes the compact-uniform-continuity triangle-estimate branch as a finite replay surface: compact finite-net source cells, continuous-map local moduli, source and target metric triangle rows, rational tolerance splits, regular readback, and a terminal Real-facing seal are displayed before a uniform-modulus consumer reads the handoff. It is localized to the finite-net source and target triangle estimate route of chapter 9237; it does not claim broad compactness, select a global modulus outside the ledger, quotient metric points, choose centers externally, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8216.1 (Finite net triangle replay carrier)**.** *A* $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, U, M, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the displayed center bundle and coverage ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row read on the bundled source centers.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer-facing handoff row produced by the replay.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *source and target triangle ledger used for the two triangle estimates.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *tolerance-split row, including the finite third-radius and lower-bound comparisons.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for rational comparison data that must be sealed for Real-facing consumers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*
8. *H stores componentwise* hsame *transport for the compact source, continuous map, uniform-modulus handoff, metric triangle ledger, rational split, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a compact finite-net request through the source triangle, local modulus, target triangle, rational readback, and uniform-modulus handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{FiniteNetTriangleReplay}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $K, F, U, M, R, Q, E, H, C, P, N$. No external center selector, quotient point equality, global compactness theorem, selected uniform modulus, or ambient Real completeness coordinate is present.

The replay consumes the compact metric carrier of chapter 129, the continuous-map row of chapter 130, the uniform-modulus consumer boundary of chapter 369, the metric-space triangle ledger of chapter 4230, the rational tolerance row of chapter 37, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38 before exporting the uniform-modulus handoff.

Theorem 8216.2 (Finite net triangle replay NameCert obligations)**.** *For an accepted* $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ *carrier $T = (K, F, U, M, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteNetTriangleReplay}^{\uparrow}}$ *surface consists exactly of carrier exposure for the compact finite-net source, continuous-map local rows, uniform-modulus consumer row, source and target metric triangle ledger, rational tolerance split, regular-rational readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows. A consumer must read the compact source row K, local map row F, triangle ledger M, and tolerance split R before it can inspect the uniform-modulus handoff U.*

Proof. Project the carrier of Definition 8216.1. The tuple displays the compact finite-net source, continuous-map row, uniform-modulus handoff, metric triangle ledger, rational split, regular readback, Real seal, transport, replay, provenance, and local name.

The source triangle estimate reads K, M, R first. These rows identify the bundled center, the source distance comparison, and the rational tolerance split that keeps the second source point inside the same local-modulus window.

The local modulus and target triangle estimates then read F, M, R, Q, E. The graph and metric rows produce the two target-side bounds, while the regular-rational and Real rows seal the displayed tolerance comparison.

Finally H, C, P, N preserve the same route. Since the carrier has no external center selector, quotient point equality, global compactness theorem, selected uniform modulus, or ambient Real completeness coordinate, the local NameCert surface is exhausted by the listed rows. □

Theorem 8216.3 (Finite net triangle replay uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer reading an accepted* $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ *packet factors through*

$$K, F, M, R \longmapsto Q, E \longmapsto U, H, C, P, N.$$

The handoff is the finite source-and-target triangle replay over compact-net centers, continuous-map local rows, metric-space triangle ledgers, rational tolerance splits, regular readback, and Real sealing. It exports no broad compactness theorem, external center choice, quotient point equality, selected modulus outside the packet, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 8216.2. They require the compact finite-net source K, continuous-map local rows F, triangle ledger M, and rational tolerance split R to be displayed before U is consumed.

The regular readback Q and Real seal E bind the rational estimates used by the source and target triangle calculations. A uniform-modulus consumer therefore reads concrete tolerance data rather than an unstated external modulus.

The transport, replay, provenance, and naming rows H, C, P, N keep the handoff on the same finite route. The intended Lean surface requires nontriviality and field faithfulness for this packet,

recorded as `Nontrivial FiniteNetTriangleReplayUp` and `FieldFaithful FiniteNetTriangleReplayUp` instances, so the replay cannot collapse into an empty or classifier-erasing carrier. □

Falsifiable prediction: Every accepted $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ uniform-modulus read factors through the compact finite-net source row, continuous-map local row, metric triangle ledger, rational tolerance split, regular readback, Real seal, componentwise transport, replay, provenance, and local $\mathsf{NameCert}$ before $\mathsf{UniformModulus}^{\uparrow}$ is consumed.

Independence witness: compactmetric, continuousmap, uniformmodulus, metricspace, real: $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net centers, continuous local rows, source and target triangle replay, rational tolerance splits, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetTriangleReplay}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteNetTriangleReplay}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap local rows, UniformModulus consumer rows, MetricSpace triangle ledgers, RatUp tolerance splits, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetTriangleReplay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8216.1, Definition 8216.2, and Definition 8216.3 seed the finite source-and-target triangle replay route for compact-uniform-continuity uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetTriangleReplay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, broad compactness, external center choice, quotient point equality, selected uniform moduli, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net source binding, continuous local-row exposure, source and target triangle replay, rational tolerance split, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteNetTriangleReplayUp.

Chapter 8217

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive dense completions, the $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$ packet records the finite non-quotient density route that sits between the dyadic source of chapter 291, the finite-window names of chapter 126, the regular rational readbacks of chapter 301, the terminal seal of chapter 38, and the dense-source face of chapter 3542. It gives Real-completion consumers a carrier-local approximation surface before any quotient completion, selected limit, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is available.

Definition 8217.1 (RegularCauchyCompletionDensity carrier)**.** *A* $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense dyadic source row, read through* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window service row used to expose source observations only at requested stages.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the regular rational representative and its displayed modulus discipline.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording precision requests, slack rows, and dyadic comparison evidence.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the dense source, stream window, regular-rational readback, and tolerance rows have been exposed.*
6. *H stores componentwise* hsame *transport for dense-source, stream-window, regular-rational, tolerance, seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from each approximation request through D, S, R, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over the dense dyadic source,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionDensity}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through these rows and replay paths. It has no coordinate for quotient completions, selected representatives, selected limits, countable choice, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyCompletionDensityUp`

Theorem 8217.2 (RegularCauchyCompletionDensity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$ *carrier* $X = (D, S, R, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionDe}}$ *surface consists exactly of carrier admission for the dense dyadic source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. Any completion-facing consumer must read* D, S, R, Q *before it can consume the seal row* E*.*

Proof. Project the carrier of Definition 8217.1. The tuple displays dense-source, stream-window, regular-rational, dyadic-tolerance, Real-seal, transport, replay, provenance, and naming rows.

The dense-source row is source-facing and finite. The StreamName row exposes only requested windows, the RegSeqRat row supplies the regular rational readback, and the DyadicRatCore row supplies the tolerance ledger used before the Real seal is read.

The seal row E is downstream of those rows. Since the packet has no quotient-completion coordinate, selected-limit coordinate, countable-choice coordinate, or ambient-completeness coordinate, the local NameCert obligations are exhausted by the displayed finite route. □

Theorem 8217.3 (RegularCauchyCompletionDensity dyadic approximation)**.** *Every accepted* $\mathsf{RegularCauchyComp}$ *packet supplies a finite dyadic approximation route*

$$\mathsf{CauchyCompletionDensitySource}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For each requested approximation window, the consumer reads the dense source row D*, the stream window row* S*, the regular-rational readback row* R*, the dyadic tolerance row* Q*, transport* H*, replay* C*, provenance* P*, and local naming* N *before reading the* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

Proof. Begin with the NameCert obligations of Definition 8217.2. They force every completion-facing read to pass through the dense source, stream window, regular rational readback, and dyadic tolerance rows before the seal row can be consumed.

The sibling chapters determine the approximation route. The dense-source packet fixes the source boundary, $\mathsf{StreamName}^{\uparrow}$ restricts observation to finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ gives the regular rational representative, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance ledger.

Transport and replay preserve that displayed route. Therefore the exported approximation is a finite dyadic source-to-seal packet, not a quotient completion, selected limit, countable-choice principle, host equality comparison, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal while omitting the dense source, stream window, RegSeqRat readback, DyadicRatCore tolerance, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchy-completion-density-source, dyadicratcore, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{RegularCauchyCompletionDensity}^{\uparrow}$ because the packet jointly carries dense-source evidence, finite observation windows, regular rational readback, dyadic tolerance accounting, Real-seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionDensity}^{\uparrow}}$ **·**

Constructive story: A RegularCauchyCompletionDensity$^{\uparrow}$ packet is generated from a dense dyadic source row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionDensity}^{\uparrow}}$, seed)

Scope closed: Definition 8217.1, Definition 8217.2, and Definition 8217.3 seed the dense dyadic approximation route consumed by constructive Cauchy completion readers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionDensity}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.RegularCauchyCompletionDensityUp`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, countable choice, trichotomy, host equality, ambient Real$^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exactness, StreamName window readback, RegSeqRat handoff, DyadicRatCore tolerance accounting, RealUp seal non-escape, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyCompletionDensityUp.

Chapter 8218

A Concrete Naming Certificate for $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive dense completions, the $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ packet realizes the finite-window side of chapter 9238. It packages the dyadic approximation window before the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal reads, using the dense source of chapter 3542, the dyadic tolerance rows of chapter 291, the observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite NameCert surface for Real-completeness consumers, not a claim of full ambient completeness.

Definition 8218.1 (RealCompletenessDensityWindow carrier)**.** *A* $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (V, D, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *V is the Real-completeness vision route row naming the finite density-window obligation that this packet realizes.*
2. *D is the* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *dense-source row.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which approximation requests are observed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the window row.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including precision request, slack, and dyadic comparison data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after D, S, R, Q are exposed.*
7. *H stores componentwise* hsame *transport for the vision route, dense source, stream window, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from each Real-completeness density request through $V, D, S, R, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over chapter 9238,* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessDensityWindow}^{\uparrow}}$ *row.*

The classifier admits only finite density-window packets. It does not identify real numbers with quotient representatives, choose completion limits, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealCompletenessDensityWindowUp`

Theorem 8218.2 (RealCompletenessDensityWindow NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ *carrier* $W = (V, D, S, R, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessDensity}}$ *surface consists exactly of the vision route row, dense-source row, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise transport, displayed replay, provenance, and local naming. A Real-completeness consumer must expose* D, S, R, Q *before it may read* E *as a seal.*

Proof. Project the carrier of Definition 8218.1. The packet displays the vision route, dense source, finite observation window, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming.

The finite density window is the only route from the vision row to the seal row. The dense-source and StreamName rows control source observation, the RegSeqRat row supplies readback, and the DyadicRatCore row records the requested tolerance before the seal is consumed.

Thus the NameCert obligations are carrier-local. Since no coordinate names quotient representatives, selected limits, countable choice, or ambient Real completeness, those objects cannot be part of the accepted window. □

Theorem 8218.3 (RealCompletenessDensityWindow seal handoff)**.** *Every accepted* $\mathsf{RealCompletenessDensityWindo}$ *packet realizes the Real-completeness density-window route*

$$\mathsf{CauchyCompletionDensitySource}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as finite NameCert data. The handoff to a $\mathsf{Real}^{\uparrow}$ *consumer contains only the finite window, tolerance, readback, seal, transport, replay, provenance, and local naming rows displayed by* $W = (V, D, S, R, Q, E, H, C, P, N)$*.*

Proof. Start with the obligation surface of Definition 8218.2. It forces the Real-completeness consumer through the dense source, finite window, regular rational readback, and dyadic tolerance rows before the Real seal row is read.

The cited concrete-instance siblings provide the route. $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ fixes the dense source, $\mathsf{DyadicRatCore}^{\uparrow}$ fixes tolerance accounting, $\mathsf{StreamName}^{\uparrow}$ fixes finite observation, $\mathsf{RegSeqRat}^{\uparrow}$ fixes regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal row.

Transport and replay keep the handoff in the same finite packet. Therefore the chapter realizes a density-window NameCert packet for the vision, not quotient completion, selected limit extraction, countable choice, host equality of real representatives, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ consumer that reads a $\mathsf{Real}^{\uparrow}$ seal without the dense-source row, dyadic tolerance row, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming row refutes the packet.

Independence witness: real-completeness, cauchy-completion-density-source, dyadicratcore, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ because this packet jointly carries the vision route row, finite density window, tolerance ledger, readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessDensityWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletenessDensityWindow}^{\uparrow}$ packet is generated from a Real-completeness vision route row, a dense-source row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessDensityWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8218.1, Definition 8218.2, and Definition 8218.3 seed the finite density-window route that the Real completeness vision can consume before seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessDensityWindow}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealCompletenessDensityWindowUp`

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, quotient completions, selected representatives, selected limits, countable choice, trichotomy, host equality, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, density-window exactness, DyadicRatCore tolerance accounting, StreamName window readback, RegSeqRat handoff, RealUp seal non-escape, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.RealCompletenessDensityWindowUp.

Chapter 8219

A Concrete Naming Certificate for $\mathsf{FiniteStreamMerge}^{\uparrow}$

Following Bishop and Bridges, a finite merge of two lawlike source windows is a useful construction only when the two finite schedules, the merge order, and the downstream rational readback remain visible. The $\mathsf{FiniteStreamMerge}^{\uparrow}$ packet sits between the finite schedule surface of chapter 5916, the parity interleaving surface of chapter 7498, the stream-name readback of chapter 126, the regular rational route of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal Real seal of chapter 38. It names a two-source finite merge certificate without replacing either source by host sequence equality or an unrecorded choice of positions.

Definition 8219.1 (Finite stream merge carrier)**.** *A* $\mathsf{FiniteStreamMerge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F_0, F_1, M, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. F_0 *is the first* $\mathsf{FiniteStream}^{\uparrow}$ *source schedule, including its finite request row, list row, and window boundary.*
2. F_1 *is the second* $\mathsf{FiniteStream}^{\uparrow}$ *source schedule, with the same finite request/list/window discipline.*
3. M *is the finite merge-order row, recording which displayed output positions read from* F_0 *and which read from* F_1*.*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *handoff row obtained after the two finite schedules and the merge-order row have been displayed.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source windows and the merged output window.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the rational reads at the requested finite precision.*
7. R *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, available only after* F_0, F_1, M, S, Q, D *have been displayed.*
8. H *records componentwise* hsame *transport for both finite sources, the merge order, stream-name handoff, regular rational readback, dyadic tolerance, Real seal, provenance, replay, and local naming.*
9. C *records displayed* Cont *replay from source-window inspection through merge-order selection, StreamName handoff, RegSeqRat readback, DyadicRatCore comparison, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteStreamMerge}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamInterleaving}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteStreamMerge}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host sequence equality, countable choice over positions, an unbounded stream merge, a hidden subsequence selector, or a Real value that bypasses the displayed finite merge order.

Theorem 8219.2 (Finite stream merge NameCert obligations)**.** *For every accepted* $\mathsf{FiniteStreamMerge}^{\uparrow}$ *carrier*

$$G = (F_0, F_1, M, S, Q, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteStreamMerge}^{\uparrow}}$ *surface consists exactly of first finite-source schedule exposure, second finite-source schedule exposure, finite merge-order exposure, StreamName handoff, RegSeqRat readback, DyadicRatCore tolerance comparison, Real-seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may erase either finite source, reorder the merge outside M, replace the route by host sequence equality, or export a Real-facing read before the finite merge certificate is displayed.*

Proof. Project the carrier of Definition 8219.1. The first two rows are the source schedules F_0 and F_1, each read through the finite request/list/window discipline of $\mathsf{FiniteStream}^{\uparrow}$.

The merge-order row M is the only coordinate that combines the two sources. It records the finite output positions, so the packet cannot appeal to countable choice over later positions or to equality of host sequences.

The downstream rows S, Q, D, R expose StreamName handoff, regular rational readback, dyadic tolerance comparison, and Real sealing only after F_0, F_1, M have been displayed. Thus every analytic read remains attached to the finite merge route.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the excluded objects are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8219.3 (Finite stream merge StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{StreamInterleaving}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteStreamMerge}^{\uparrow}$ *packet receives the handoff row S only through the two finite source schedules F_0, F_1 and the finite merge-order row M. The consumer may compare with the interleaving packet of chapter 7498, but it may not replace the displayed merge certificate by a private stream selector.*

Proof. Read F_0 and F_1 first as finite schedules. Each source exposes only a finite request, list, and window boundary, so no infinite host stream is available to the consumer.

Next read the merge-order row M. This row determines which source supplies each displayed output position before any $\mathsf{StreamName}^{\uparrow}$ handoff is issued.

The handoff row S is therefore downstream of the same finite merge certificate. The parity split route of Definition 7498.4 is a sibling consumer of this discipline: it may read the finite source windows, but it still receives the output through a displayed selection row rather than through hidden sequence equality. □

Theorem 8219.4 (Finite stream merge Real nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteStreamMerge}^{\uparrow}$ *packet factors through*

$$(F_0, F_1, M) \longmapsto S \longmapsto Q \longmapsto D \longmapsto R$$

with the same H, C, P, N rows attached. A Real-facing read cannot bypass the two finite source schedules, the merge-order row, the StreamName handoff, the regular rational readback, or the dyadic tolerance comparison.

Proof. Project the carrier. The Real seal row R is displayed after the finite sources F_0, F_1, the merge order M, the StreamName handoff S, the RegSeqRat readback Q, and the DyadicRatCore tolerance row D.

A regular rational consumer must therefore read Q as a readback of the displayed source and output windows, not as an independent rational sequence. The dyadic row D then checks the requested precision against that same finite data.

Finally the Real consumer reads R only after the route above has been replayed through C and transported through H. Provenance P and local naming N keep the same certificate attached, so a Real-facing value has no route around the finite merge packet. □

Falsifiable prediction: A $\mathsf{FiniteStreamMerge}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without displaying $F_0, F_1, M, S, Q, D, H, C, P, N$ refutes the carrier of Definition 8219.1.

Independence witness: finite-stream, stream-interleaving, streamname, regseqrat, dyadicratcore, real: none is bijective because FiniteStreamMergeUp jointly carries two finite source schedules, one merge-order row, StreamName handoff, regular rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteStreamMerge}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteStreamMerge}^{\uparrow}$ packet is generated from two finite source schedules, a finite merge-order row, StreamName handoff, RegSeqRat readback, DyadicRatCore tolerance comparison, a Real seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteStreamMerge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8219.1, Definition 8219.2, Definition 8219.3, and Definition 8219.4 seed the finite two-source merge route between finite schedules, stream-name handoff, regular rational readback, dyadic tolerance, and Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteStreamMerge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host sequence equality, countable choice over positions, unbounded stream merging, quotient Cauchy equality, or a Real seal independent of the displayed finite merge route.

Upgrade path: Obligation closure requires checked carrier, merge-order, StreamName handoff, RegSeqRat readback, DyadicRatCore tolerance, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteStreamMergeUp.

Chapter 8220

A Concrete Naming Certificate for DyadicPiecewiseLinearInterpolation$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicPiecewiseLinearInterpolation$^{\uparrow}$ packet records the finite dyadic mesh route by which endpoint rational data is read as a piecewise-linear interpolation graph before any regular Cauchy or Real consumer sees a completion-facing handoff. It sits after the midpoint arithmetic of chapter 445, the dyadic rational core of chapter 291, the regular interpolation corridor of chapter 1740, the rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The packet names only displayed finite mesh data; it is not a host function space, an ambient uniform-continuity theorem, a selected real limit, or a completeness principle for Real$^{\uparrow}$.

Definition 8220.1 (Dyadic piecewise-linear interpolation carrier)**.** *A* DyadicPiecewiseLinearInterpolation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, E, S, G, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite dyadic mesh row. It lists ordered dyadic knots, endpoint inclusions, adjacent midpoint checks, and the local interval codes read through* DyadicMidpoint$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$*.*
2. *E is the endpoint-value row. It assigns displayed dyadic rational values to the knots of M and records the equality checks needed at shared segment endpoints.*
3. *S is the segment-slope row. For every adjacent knot pair it records the displayed dyadic numerator, denominator, nonzero interval-width witness, and rational slope ledger.*
4. *G is the interpolation graph row. It records the affine value rule on each mesh segment, the endpoint agreement rows, and the segment-membership row used when a downstream reader asks for a graph value.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained by reading the graph on dyadic refinement windows rather than by invoking a host function.*
6. *L is the terminal* Real$^{\uparrow}$ *handoff row, available only after the mesh, endpoint, slope, graph, and regular rational readback rows have been displayed.*
7. *H records componentwise* hsame *transport for mesh knots, endpoint values, segment slopes, graph rows, regular rational readback, terminal handoff, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from an interpolation request through $M, E, S, G, R, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$, $\mathsf{DyadicMidpoint}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyInterpolation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, S, G, R, L, H, C, P, N$. It has no coordinate for an external real-valued function, a host slope field, arbitrary real arithmetic, quotient stream equality, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8220.2 (Dyadic piecewise-linear interpolation NameCert obligations)**.** *For every accepted* $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$ *carrier*

$$I = (M, E, S, G, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}}$ *surface consists exactly of finite dyadic mesh admission, endpoint-value admission, segment-slope ledger construction, interpolation graph exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 8220.1. The rows M, E, S, G are the only mesh, endpoint, slope, and graph coordinates, while R, L, H, C, P, N only read back, seal, transport, replay, source, and name those rows.

The arithmetic obligations are local to the dyadic rows. The mesh row reads midpoint and dyadic rational evidence through $\mathsf{DyadicMidpoint}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$, the endpoint row records displayed values at mesh knots, and the slope row stores only the finite rational differences and interval-width witnesses for adjacent knots.

The graph obligation is then a finite segment ledger. The row G chooses a displayed segment by the membership row supplied by M, applies the affine rule recorded by S, and checks endpoint agreement against E.

Finally R reads the same graph through regular rational refinement windows, L exposes the Real-facing seal only after that readback, and H, C, P, N preserve the same route. Since no carrier coordinate stores a host function, arbitrary real arithmetic, quotient stream equality, selected limit, or ambient completeness principle, the listed rows exhaust the local certificate obligations. □

Theorem 8220.3 (Dyadic piecewise-linear interpolation RegSeqRat handoff)**.** *Every regular-rational or Real-and-Completion consumer of an accepted* $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$ *packet reads the interpolation route through*

$$\mathsf{DyadicMidpoint}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyInterpolation}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer must first expose the dyadic mesh row M, endpoint-value row E, segment-slope row S, and interpolation graph row G before it may read the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R or terminal seal L.*

Proof. Start from an accepted packet $I = (M, E, S, G, R, L, H, C, P, N)$. The only source of interval data is the finite mesh row M, and the only source of endpoint values is the row E tied to that mesh.

The segment-slope row S and graph row G turn this finite dyadic data into displayed affine segment reads. These rows are exactly the interpolation surface consumed by the regular interpolation corridor; no downstream consumer can bypass them to obtain a host function.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of the graph row and records rational refinement reads of the same finite interpolation ledger. The terminal $\mathsf{Real}^{\uparrow}$ row L is available only after R is displayed.

The transport, replay, provenance, and naming rows preserve this order. Therefore every regular-rational or completion-facing handoff factors through the displayed dyadic mesh, endpoint, slope, graph, and readback rows, and it exports no independent real function, selected limit, quotient stream equality, or ambient completeness principle. □

Falsifiable prediction: A completion-facing consumer that exports a $\mathsf{RegSeqRat}^{\uparrow}$ readback from $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$ while omitting one of the displayed mesh, endpoint, segment-slope, graph, transport, replay, provenance, or local naming rows is not reading this packet.

Independence witness: Nearest siblings: dyadicmidpoint, dyadicratcore, regular-cauchy-interpolation, regseqrat, real. The carrier is not bijective to DyadicMidpointUp or DyadicRatCoreUp because it adds finite mesh, endpoint, slope, and graph rows; not to RegularCauchyInterpolationUp because it fixes dyadic piecewise-linear data; not to RegSeqRatUp because it supplies the interpolation ledger before readback; and not to RealUp because it records the finite pre-seal route rather than a terminal real name alone.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8220.1, Definition 8220.2, and Definition 8220.3 seed the finite dyadic interpolation route from mesh rows and endpoint values to regular rational readback and a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, arbitrary real arithmetic, quotient stream equality, selected limits, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh admission, endpoint-value stability, segment-slope ledger exactness, interpolation graph exposure, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicPiecewiseLinearInterpolationUp.

Constructive story: A $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$ packet is generated from finite dyadic mesh rows, endpoint-value rows, segment-slope rows, interpolation graph rows, RegSeqRat readback, a Real-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8221

A Concrete Naming Certificate for $\mathsf{RegularClosedSet}^{\uparrow}$

Following Engelking, the $\mathsf{RegularClosedSet}^{\uparrow}$ packet records the topology-facing closed sets that are recovered from the closure of their own interior. It sits beside the regular-open comparison route of chapter 6835, reads ambient open and closed observations through $\mathsf{Topology}^{\uparrow}$, and records membership provenance through $\mathsf{SetLike}^{\uparrow}$. The packet is a finite NameCert surface for closure-interior stability, not a host powerset equality, a Boolean algebra completion, a selected basis, or an imported compactness theorem.

Definition 8221.1 (Regular closed set carrier)**.** *A* $\mathsf{RegularClosedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, S, O, I, C, E, H, Q, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row naming the finite open and closed observations available to the packet.*
2. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership row for the underlying subset being inspected.*
3. *O is the interior row, recording the open part read from S through T.*
4. *I is the closure-of-interior row, recording the closed row reached from O inside the same topology.*
5. *C is the closedness row, asserting that the inspected subset is read as the closure row I.*
6. *E is the exactness row, recording the displayed equality of the subset row with the closure of its interior as a local certificate row.*
7. *H is componentwise* hsame *transport for T, S, O, I, C, E.*
8. *Q is displayed* Cont *replay for the route $S \to O \to I \to C \to E$.*
9. *P is* Pkg *provenance over* $\mathsf{RegularClosedSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{RegularOpenAlgebra}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularClosedSet}^{\uparrow}}$ *row.*

The carrier exports only the displayed topology, membership, interior, closure, closedness, exactness, transport, replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.RegularClosedSetUp.RegularClosedSetTasteGate single carrier alignment`

Theorem 8221.2 (Regular closed set NameCert obligations)**.** *For the carrier of Definition 8221.1, the local* $\mathsf{NameCert}_{\mathsf{RegularClosedSet}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (T, S, O, I, C, E, H, Q, P, N)$*, ambient topology exposure through* T*, set-membership exposure through* S*, interior exposure through* O*, closure-of-interior exposure through* I*, closedness exposure through* C*, exactness exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* Q*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The tuple displays the ambient topology row, membership row, interior row, closure-of-interior row, closedness row, exactness row, and structural support rows.

Read the sibling anchors. $\mathsf{Topology}^{\uparrow}$ supplies the open and closed observation discipline, $\mathsf{SetLike}^{\uparrow}$ supplies membership provenance, and chapter 6835 supplies the neighbouring regular-open comparison route without replacing the closed-set carrier.

The remaining rows H, Q, P, N transport, replay, source, and name the same finite route. Since the carrier has no further coordinate, the NameCert obligations cannot export a host powerset equality, selected basis, Boolean algebra completion, compactness theorem, or ambient quotient of closed sets. □

Theorem 8221.3 (Regular closed set closure-interior stability)**.** *Let* $R = (T, S, O, I, C, E, H, Q, P, N)$ *be an accepted* $\mathsf{RegularClosedSet}^{\uparrow}$ *packet. Every accepted read of the underlying subset row* S *that passes through the displayed topology row* T *and interior row* O *returns to the same closed row* I *after closure, and the exactness row* E *keeps the carrier classified as a regular closed set. The stable route is*

$$S \to O \to I \to C \to E,$$

not a host powerset equality, selected basis, Boolean completion, compactness principle, or quotient equality of subsets.

Proof. Begin with the carrier of Definition 8221.1. The subset row is not read directly as a host set; it is read through the topology row, then through the displayed interior and closure-of-interior rows.

Apply the NameCert obligations of Definition 8221.2. They require the same T, S, O, I, C, E route before a local certificate can expose regular closedness.

The transport and replay rows preserve only those displayed coordinates. Thus a downstream topology or set-like consumer receives closure-interior stability for the finite carrier, while the excluded host powerset equality, selected basis, Boolean completion, compactness principle, and quotient subset equality remain outside the packet. □

Falsifiable prediction: A claimed $\mathsf{RegularClosedSet}^{\uparrow}$ packet whose subset row is not recovered from the closure of its displayed interior cannot satisfy the exactness row E.

Independence witness: topology, setlike, regularopenalgebra: $\mathsf{RegularClosedSet}^{\uparrow}$ is not bijective to any listed sibling because it binds ambient topology, set-membership provenance, interior, closure-of-interior, closedness, exactness, componentwise transport, replay, provenance, and local NameCert rows in one packet. TopologyUp supplies open and closed observations without the regular-closed exactness row, SetLikeUp supplies membership without closure-interior stability, and RegularOpenAlgebraUp supplies the neighbouring regular-open comparison route rather than the closed-set carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularClosedSet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularClosedSet}^{\uparrow}$ packet is generated from ambient TopologyUp observations, SetLikeUp membership provenance, an interior row, a closure-of-interior row, a closedness row, an exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularClosedSet}^{\uparrow}}$, seed)

Scope closed: Definition 8221.1, Definition 8221.2, and Definition 8221.3 seed the regular-closed carrier, obligation surface, and closure-interior stability route over Topology$^{\uparrow}$, SetLike$^{\uparrow}$, RegularOpenAlgebra$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularClosedSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host powerset equality, selected bases, Boolean algebra completion, compactness, quotient subset equality, or ambient topological completeness.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same topology, membership, interior, closure-of-interior, closedness, exactness, transport, replay, provenance, and local naming rows.

Chapter 8222

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$ packet records the finite-net search route for a continuous real-valued map on a displayed compact real interval. It sits after the dyadic endpoint core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the uniform-modulus surface of chapter 369, the graph rows of chapter 130, the finite epsilon-net carrier of chapter 1767, and the located bisection route of chapter 1711. The packet names a finite compact-interval search carrier; it does not invoke open-cover compactness, selected host extrema, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8222.1 (Compact real interval net-search carrier)**.** *A* $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, D, U, B, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact real interval endpoint row. Its lower and upper endpoints are read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and terminal* $\mathsf{Real}^{\uparrow}$ *names.*
2. *D is the dyadic finite-net cell row, listing the mesh cells, their dyadic endpoints, center probes, and coverage witnesses consumed from* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row for the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *graph on the interval source named by I.*
4. *B is the located bisection row, recording the retained cell branch and the finite bisection depth read through* $\mathsf{LocatedIntervalBisection}^{\uparrow}$*.*
5. *V is the value-bracket row, storing lower and upper value comparisons for the inspected cells with the tolerance budget supplied by U.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic endpoints, cell centers, tolerance comparisons, bisection depth, and value brackets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the interval endpoints, finite-net cells, bisection row, value bracket, and regular rational readback have been exposed.*

8. *H stores componentwise* hsame *transport for the interval endpoints, dyadic finite-net cells, uniform-modulus row, located bisection, value bracket, readback, Real seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a compact-interval search request through $I, D, U, B, V, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSearch}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, D, U, B, V, R, E, H, C, P, N$. No host-selected extremum, open-cover compactness proof, quotient equality of real names, or ambient completeness coordinate is part of the packet.

Theorem 8222.2 (Compact real interval net-search obligations)**.** *For every accepted* CompactRealIntervalNetSearch *carrier $S = (I, D, U, B, V, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSearch}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval endpoints, dyadic finite-net cells, uniform-modulus data, located bisection, value bracket, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming rows. A consumer must read I, D, U, B, V before it may inspect the readback R or the Real seal E.*

Proof. Project the carrier of Definition 8222.1. The displayed analytic rows are the interval endpoint row I, the dyadic finite-net row D, the uniform-modulus row U, the located bisection row B, and the value bracket row V.

The readback rows are downstream. The regular rational row R repacks only the dyadic endpoints, mesh centers, bisection depth, and value comparisons already displayed by I, D, U, B, V, and the Real seal E is terminal after that readback.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no open-cover compactness proof, host-selected extremum, quotient equality coordinate, or ambient completeness row, the listed rows exhaust the local NameCert surface. □

Theorem 8222.3 (Compact real interval net-search dyadic-net soundness)**.** *Every accepted* CompactRealIntervalNe *search read factors through the dyadic finite-net route*

$$I, D, U \longmapsto B, V \longmapsto R, E, H, C, P, N.$$

The finite cells in D cover the displayed interval endpoints of I, the located bisection row B refines only those cells, and the value bracket V uses only $\mathsf{ContinuousMap}^{\uparrow}$ *values controlled by U. A Real-facing consumer receives no cell outside the finite net and no value comparison detached from the displayed modulus budget.*

Proof. Start from I and D. The interval endpoints and dyadic mesh cells are displayed before the search begins, so every cell inspected by the packet is a row of the finite net.

Use U and B next. The uniform-modulus row supplies the tolerance budget for graph reads, while the located bisection row refines a displayed cell from D rather than choosing a point externally.

The value bracket V is then attached to those cells and to that modulus budget. The readback and seal rows R, E can only package the resulting finite data, and H, C, P, N preserve the same route for downstream consumers. □

Theorem 8222.4 (Compact real interval net-search termination)**.** *For an accepted* $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$ *carrier* $S = (I, D, U, B, V, R, E, H, C, P, N)$*, the search terminates at the displayed finite depth of* B*. The terminal read returns a finite cell family and value bracket whose mesh cells are listed in* D*, whose tolerance budget is supplied by* U*, and whose Real-facing read passes through* R *and* E*.*

Proof. The bisection row B is finite carrier data. Its depth is a displayed row, not a function selected from an ambient compactness theorem.

Apply dyadic-net soundness from Definition 8222.3. Each bisection step refines a cell already listed by D and reads values with the tolerance budget supplied by U.

At the terminal depth, the packet returns only the cell rows and value brackets stored in V. The regular rational readback R and the Real seal E package those rows, so termination is a finite ledger fact of the carrier. □

Theorem 8222.5 (Compact real interval net-search NameCert export)**.** *The public* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSea}}$ *export is exactly the finite route*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, Located

It exports a compact-interval finite-net search certificate for downstream real-completion, uniform-continuity, and extreme-value consumers, while excluding open-cover compactness, host-selected extrema, quotient real equality, and ambient Real completeness.

Proof. Combine the carrier obligations of Definition 8222.2 with dyadic-net soundness in Definition 8222.3. The source rows are exactly the dyadic endpoint core, finite stream windows, regular rational readback, Real seal, uniform modulus, continuous-map graph, finite epsilon-net cells, and located bisection route.

Apply termination from Definition 8222.4. The exported search result is a finite terminal cell family with value brackets, transported and replayed through H, C and named by P, N.

The excluded principles are absent from the carrier. Therefore no public consumer can obtain open-cover compactness, a host-selected extremum, quotient real equality, or ambient Real completeness from this NameCert export. □

Falsifiable prediction: Every accepted $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$ read must display interval endpoints, dyadic finite-net cells, a uniform-modulus row, located bisection, value brackets, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert before any compact-interval search result is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, uniformmodulus, continuousmap, finite-epsilon-net, located-interval-bisection: $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval endpoints, dyadic finite-net cells, uniform-modulus data, located bisection, value brackets, readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSearch}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8222.1, Definition 8222.2, Definition 8222.3, Definition 8222.4, and Definition 8222.5 seed the constructive compact real interval finite-net search route over dyadic endpoints, finite cells, uniform-modulus data, located bisection, value brackets, readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalNetSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host-selected extrema, quotient real equality, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{CompactRealIntervalNetSearch}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-net soundness, finite-depth termination, readback exactness, Real-seal nonescape, transport stability, and public NameCert export for BEDC.Derived.CompactRealIntervalNetSearchUp.

Constructive story: A CompactRealIntervalNetSearch$^{\uparrow}$ packet is generated from interval endpoints, dyadic finite-net cells, UniformModulusUp data, ContinuousMapUp graph rows, a LocatedIntervalBisectionUp finite branch, value brackets, RegSeqRatUp readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8223

A Concrete Naming Certificate for $\mathsf{HalesJewett}^{\uparrow}$

Following Hales and Jewett's finite-word theorem, the $\mathsf{HalesJewett}^{\uparrow}$ packet records the finite alphabet, word window, variable-word line, and coloring readback surface as displayed NameCert data. It sits beside the finite-coloring witness discipline of chapter 203 and reads words through the $\mathsf{List}^{\uparrow}$ spine of chapter 36. The packet names a finite combinatorial-line certificate; it is not an ambient infinite-choice principle, host language equality, unrestricted substitution schema, or external Hales–Jewett theorem oracle.

Definition 8223.1 (Hales-Jewett carrier)**.** *A* $\mathsf{HalesJewett}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, W, V, L, C, H, R, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet row, including the displayed color endpoints and the finite symbol classifier used by all word reads.*
2. *W is the finite word-window row. Each word is read as a* $\mathsf{List}^{\uparrow}$ *spine over A, and the window records the inspected length and membership rows.*
3. *V is the variable-word row. It records a finite word spine over A together with at least one displayed variable position.*
4. *L is the combinatorial-line row obtained by substituting each displayed alphabet symbol into every variable position of V while leaving the constant positions fixed.*
5. *C is the finite coloring row on the word window, exposing colors only for the words listed by W and the line entries listed by L.*
6. *H records componentwise* hsame *transport for the alphabet, word window, variable word, line row, coloring row, replay, provenance, and local naming.*
7. *R records displayed* Cont *replay for alphabet membership, word-spine reads, variable substitution, line enumeration, coloring lookup, and monochromatic-line readback.*
8. *P is the* Pkg *provenance over* $\mathsf{HalesJewett}^{\uparrow}$*,* $\mathsf{Ramsey}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HalesJewett}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $A, W, V, L, C, H, R, P, N$. It contains no coordinate for host language equality, infinite choice of substitutions, an unrestricted coloring function, or an ambient combinatorial principle beyond the displayed finite packet.

Theorem 8223.2 (Hales-Jewett NameCert obligations)**.** *For every accepted* $\mathsf{HalesJewett}^{\uparrow}$ *carrier*

$$J = (A, W, V, L, C, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HalesJewett}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite alphabet row, word-window exposure, variable-word admission with a displayed variable position, substitution stability for the combinatorial-line row, finite coloring lookup, monochromatic-line readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer reads the line through the displayed* $\mathsf{List}^{\uparrow}$ *word spines and may use the* $\mathsf{Ramsey}^{\uparrow}$ *sibling only as a finite-coloring neighbor; no consumer obtains infinite choice, host language equality, an unrestricted coloring oracle, or an ambient combinatorial theorem from the packet.*

Proof. Project the accepted carrier. The rows A, W, V, L, C name the finite alphabet, the finite word window, the variable word, the line generated by substitutions into the displayed variable positions, and the coloring values available on that finite window.

Word and line admission are finite. The word window W reads each word through a $\mathsf{List}^{\uparrow}$ spine over A, while V marks the variable positions and L lists exactly the words obtained by substituting displayed alphabet symbols into those positions.

Coloring and monochromatic readback are also local to the packet. The row C is consulted only on words exposed by W and line entries exposed by L, so a monochromatic-line consumer can compare only those listed color rows.

Finally H, R, P, N transport, replay, source, and name the same finite data. Since the carrier has no coordinate for infinite substitution choice, host language equality, an external coloring function, or an ambient Hales–Jewett theorem, the listed rows exhaust the NameCert obligations. □

Theorem 8223.3 (Hales-Jewett finite-line handoff)**.** *Every downstream finite-combinatorics consumer of an accepted* $\mathsf{HalesJewett}^{\uparrow}$ *packet reads the route*

$$\mathsf{List}^{\uparrow},\ \mathsf{Ramsey}^{\uparrow},\ \mathsf{Nat}^{\uparrow} \longmapsto \mathsf{HalesJewett}^{\uparrow}$$

only through the displayed alphabet row, word-window row, variable-word row, combinatorial-line row, coloring row, componentwise transport, replay, provenance, and local naming. The handoff supplies a finite variable-word line and its monochromatic readback, not a host word language, an infinite choice sequence, or an unrestricted combinatorial principle.

Proof. Read the packet rows in order. The $\mathsf{List}^{\uparrow}$ spine supplies finite word syntax, the $\mathsf{Nat}^{\uparrow}$ row supplies the inspected length and position budget, and the $\mathsf{Ramsey}^{\uparrow}$ sibling supplies only the neighboring finite-coloring discipline.

The line handoff is the row L together with the coloring row C. Substitution is performed only at the variable positions displayed by V, and every resulting word must already be exposed in the finite window W before its color can be compared.

The structural rows H, R, P, N preserve this same finite route. Therefore a downstream consumer receives the displayed variable-word line and monochromatic readback, while the excluded host and infinitary objects remain outside the accepted carrier. □

Falsifiable prediction: Every accepted $\mathsf{HalesJewett}^{\uparrow}$ line read factors through a displayed finite alphabet row, a $\mathsf{List}^{\uparrow}$ word-window row, a variable-word row with a visible variable position, a finite combinatorial-line row, a coloring row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ramsey, list, nat: $\mathsf{HalesJewett}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite alphabet rows, word-window rows, variable-word substitution rows, combinatorial-line rows, coloring readback, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{HalesJewett}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HalesJewett}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8223.1, Definition 8223.2, and Definition 8223.3 seed the finite word-line route over alphabet rows, ListUp word spines, variable-word substitution, finite coloring rows, RamseyUp neighboring coloring discipline, NatUp budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HalesJewett}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host language equality, infinite choice of substitutions, unrestricted coloring functions, ambient Hales–Jewett theorem use, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, word-window exposure, variable substitution stability, monochromatic-line readback, coloring-window exactness, transport, replay, provenance, and local naming for BEDC.Derived.HalesJewettUp.

Constructive story: A $\mathsf{HalesJewett}^{\uparrow}$ packet is generated from finite alphabet rows, ListUp word-window rows, a variable-word row, a finite combinatorial-line row, a coloring row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8224

A Concrete Naming Certificate for $\mathsf{ChineseRemainder}^{\uparrow}$

Following Gauss and the standard algebraic Chinese remainder theorem, the $\mathsf{ChineseRemainder}^{\uparrow}$ packet records finite congruence ledgers over displayed integer, ring, ideal, prime, coprime, and unary-index rows. It sits after the integer carrier of chapter 31, the operation rows of chapter 43, the modulus rows of chapter 81, the prime and coprime ledger rows of chapter 64, and the finite index discipline of chapter 28. The packet names only the displayed residue computation route; it is not a quotient-ring primitive, a selected representative oracle, an ambient ideal factorization theorem, or a host-side modular arithmetic black box.

Definition 8224.1 (Chinese remainder carrier)**.** *A* $\mathsf{ChineseRemainder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, A, M, C, U, L, V, T, H, Q, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Int}^{\uparrow}$ *source row for the displayed residue values and reconstructed integer candidate.*
2. *A is the* $\mathsf{Ring}^{\uparrow}$ *operation row used for addition, multiplication, subtraction, and unit reads during recombination.*
3. *M is the finite modulus row, read through* $\mathsf{Ideal}^{\uparrow}$*, listing the displayed ideals or integer-modulus rows against which congruences are checked.*
4. *C is the* $\mathsf{Prime}^{\uparrow}$ *and coprime ledger row, recording the pairwise coprimality witnesses required by the displayed moduli.*
5. *U is the* $\mathsf{Nat}^{\uparrow}$ *finite index row, naming the inspected modulus positions and the finite recursion order.*
6. *L is the congruence ledger, storing each residue row, its modulus row, and the local proof that the candidate agrees with the residue modulo that row.*
7. *V is the recombination row, displaying the finite Bezout or unit coefficients used to assemble the candidate from the listed residues.*
8. *T is the transport row, recording the componentwise* hsame *movement of residues, moduli, coefficients, congruence checks, and reconstructed candidate data.*
9. *H is the replay row, storing displayed* Cont *replay from the residue request through I, A, M, C, U, L, V, T.*

10. *Q is the classifier row, comparing accepted packets only by the displayed residue ledger, coprime ledger, recombination row, transport, and replay data.*
11. *P is the* Pkg *provenance over* $\mathsf{ChineseRemainder}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ChineseRemainder}^{\uparrow}}$ *row.*

The classifier has no coordinate for quotient-ring equality, a hidden representative selector, an infinite family of moduli, a non-displayed ideal decomposition, or a detached host proof of the Chinese remainder theorem.

Theorem 8224.2 (Chinese remainder NameCert obligations)**.** *For every accepted* $\mathsf{ChineseRemainder}^{\uparrow}$ *carrier*

$$R = (I, A, M, C, U, L, V, T, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChineseRemainder}^{\uparrow}}$ *surface consists exactly of integer residue admission, ring-operation admission, finite modulus-row admission, coprime ledger admission, unary index budgeting, congruence ledger exposure, recombination coefficient exposure, componentwise* hsame *transport, displayed* Cont *replay, classifier comparison through displayed rows, provenance, and local naming.*

Proof. Project the carrier of Definition 8224.1. The rows I, A, M, C, U expose the arithmetic source, operation surface, modulus list, coprime witnesses, and finite index budget before any residue computation can be read.

Read the ledger and recombination rows next. The row L stores the finite congruence checks, and V stores the displayed coefficient data used to assemble the reconstructed candidate; neither row can be replaced by a quotient-ring equality or a selected representative.

Finally T, H, Q, P, N transport, replay, compare, source, and name the same finite packet. Since the carrier contains no other coordinate, these rows exhaust the local NameCert obligations. □

Theorem 8224.3 (Chinese remainder congruence ledger)**.** *Let $R = (I, A, M, C, U, L, V, T, H, Q, P, N)$ be an accepted* $\mathsf{ChineseRemainder}^{\uparrow}$ *carrier. For every modulus position displayed by U, the reconstructed candidate named by I and V agrees with the corresponding residue row of L modulo the corresponding modulus row of M, and the agreement is replayed through the same coprime ledger C and transport row T.*

Proof. Induct over the finite index row U. In the base case, the first displayed modulus row is read from M, its residue row is read from L, and the recombination row V supplies the coefficient data whose local congruence is checked by the ring operations in A.

For the induction step, assume the already-inspected prefix has been replayed through H and transported by T. The next index exposes its modulus row and coprime witnesses through M and C, and the next congruence row of L checks the reconstructed candidate against that modulus using only displayed operations.

Because the induction ranges only over the finite row list, it cannot introduce quotient-ring representatives, an infinite modulus family, or an external CRT oracle. The replay row H and provenance row P then package the same finite congruence ledger as the public local certificate. □

Falsifiable prediction: An accepted $\mathsf{ChineseRemainder}^{\uparrow}$ consumer is refuted by any reconstructed integer read that omits one of the displayed integer residue, ring operation, finite modulus, coprime ledger, unary index, congruence ledger, recombination, transport, replay, provenance, or local naming rows.

Independence witness: int, ring, ideal, prime, nat: $\mathsf{ChineseRemainder}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores residue rows, operation rows, finite modulus rows, pairwise coprime witnesses, index data, congruence ledgers, recombination coefficients, transport, replay, classifier comparison, provenance, and local NameCert data in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ChineseRemainder}^{\uparrow}}$.

Constructive story: A $\mathsf{ChineseRemainder}^{\uparrow}$ packet is generated from integer residue rows, ring operation rows, finite modulus rows, coprime ledger rows, unary index data, congruence ledgers, recombination coefficients, componentwise hsame transport, displayed Cont replay, classifier comparison, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChineseRemainder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8224.1, Definition 8224.2, and Definition 8224.3 seed the finite congruence-ledger route over IntUp, RingUp, IdealUp, PrimeUp, coprime, NatUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChineseRemainder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-ring equality, selected representatives, ideal factorization, infinite modulus families, or an ambient Chinese remainder theorem.

Upgrade path: Obligation closure requires checked carrier admission, finite modulus exposure, coprime ledger exactness, congruence-row replay, recombination-row stability, transport, classifier comparison, provenance, and local naming for BEDC.Derived.ChineseRemainderUp.

Chapter 8225

A Concrete Naming Certificate for $\mathsf{RationalIntervalPartition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalIntervalPartition}^{\uparrow}$ packet records a finite partition of a rational interval as displayed rational endpoints, dyadic refinement witnesses, stream-window access, regular-rational readback, and a terminal Real-facing boundary. It sits after the rational carrier of chapter 37, the dyadic rational core of chapter 291, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names a finite partition certificate only; it is not an open-cover compactness theorem, a selected refinement principle, quotient stream equality, arbitrary real interval arithmetic, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8225.1 (Rational interval partition carrier)**.** *A* $\mathsf{RationalIntervalPartition}^{\uparrow}$ *carrier is a finite packet*

$$P = (I, E, O, L, D, S, Q, R, H, C, G, N)$$

with the following displayed rows:

1. *I is the closed rational interval row, recording the left endpoint, right endpoint, and rational endpoint comparison through* $\mathsf{Rat}^{\uparrow}$*;*
2. *E is the finite endpoint-list row, recording the ordered rational breakpoints from the left endpoint to the right endpoint;*
3. *O is the adjacent-order row, recording for each neighboring pair the displayed rational comparison that makes a subinterval;*
4. *L is the local-length row, recording rational length data for each adjacent subinterval without completing it to a real measure;*
5. *D is the dyadic-refinement row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *witnesses used when a downstream reader asks for dyadic tolerances on the partition;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording finite observation windows for the ordered endpoint list and its adjacent pairs;*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording regular rational observations of endpoint windows and dyadic tolerance rows;*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *handoff row, available only after I, E, O, L, D, S, Q have been displayed;*

9. *H records componentwise* hsame *transport for interval endpoints, endpoint lists, adjacent-order rows, length rows, dyadic witnesses, stream windows, regular readback, real handoff, replay, provenance, and local naming;*
10. *C records displayed* Cont *replay through $I, E, O, L, D, S, Q, R, H, G, N$;*
11. *G is the* Pkg *provenance over* $\mathsf{RationalIntervalPartition}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an arbitrary cover, a host supremum or infimum, a selected compactness witness, quotient equality of real names, countable choice, or ambient completeness.

Theorem 8225.2 (Rational interval partition NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalPartition}^{\uparrow}$ *carrier*

$$P = (I, E, O, L, D, S, Q, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalPartition}^{\uparrow}}$ *surface consists exactly of rational interval admission, finite endpoint-list admission, adjacent-order checks, local rational length rows, dyadic refinement witnesses, stream-window exposure, regular rational readback, terminal Real-facing handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8225.1. The interval data are exactly the rational endpoint row I, finite endpoint list E, adjacent-order row O, and local length row L.

The refinement and observation obligations are the rows D, S, Q, R. Dyadic tolerance requests read D, finite endpoint observations read S, regular rational consumers read Q, and Real-facing consumers reach R only after the preceding rows are present.

The structural rows H, C, G, N preserve and name that same finite route. Since no carrier coordinate stores an arbitrary open cover, selected refinement, quotient real equality, countable-choice schedule, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 8225.3 (Rational interval partition completion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalIntervalPartition}^{\uparrow}$ *packet reads the partition through the route*

$$\mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the consumer must expose the rational interval row I, finite endpoint-list row E, adjacent-order row O, local length row L, dyadic-refinement row D, and stream-window row S before it may read the regular-rational row Q or terminal Real-facing row R.

Proof. Start from an accepted carrier $P = (I, E, O, L, D, S, Q, R, H, C, G, N)$. The only interval source is the rational row I, and the only partition source is the finite endpoint list E together with the adjacent-order checks O.

The local length row L and dyadic-refinement row D keep each subinterval inside rational and dyadic data. The stream-window row S is therefore the only finite observation surface for downstream reads.

The regular-rational row Q is downstream of those windows, and the terminal Real-facing row R is downstream of Q. Transport, replay, provenance, and local naming preserve this order and do not add an open-cover compactness row or a selected limit. □

Falsifiable prediction: A completion-facing consumer that exports a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback from $\mathsf{RationalIntervalPartition}^{\uparrow}$ while omitting one of the displayed rational interval, endpoint-list, adjacent-order, length, dyadic-refinement, stream-window, transport, replay, provenance, or local naming rows is not reading this packet.

Independence witness: Nearest siblings: rat, dyadicratcore, streamname, regseqrat, real. The carrier is not bijective to RatUp because it adds a finite ordered endpoint list and adjacent subinterval rows; not to DyadicRatCoreUp because dyadic data only refines displayed rational intervals; not to StreamNameUp because windows observe a fixed partition surface; not to RegSeqRatUp because readback is downstream of the partition ledger; and not to RealUp because the terminal seal is only a handoff row.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalPartition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8225.1, Definition 8225.2, and Definition 8225.3 seed the finite rational interval partition route from rational endpoints to regular rational readback and a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected refinements, quotient real equality, countable choice, arbitrary real interval arithmetic, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational endpoint ordering, finite endpoint-list stability, adjacent subinterval exactness, dyadic refinement, stream-window readback, RegSeqRat handoff, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalIntervalPartitionUp.

Constructive story: A $\mathsf{RationalIntervalPartition}^{\uparrow}$ packet is generated from rational interval endpoints, a finite endpoint list, adjacent-order rows, rational length rows, dyadic-refinement witnesses, stream-window rows, RegSeqRat readback, a Real-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8226

A Concrete Naming Certificate for $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges and Kelley on completion universal properties, the $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ packet records the located-completion unit and extension route as finite NameCert data. It sits beside the metric-completion carrier of chapter 1506, the located completion seal of chapter 1805, the located uniform-completion route of chapter 1692, the regular-Cauchy completion functor of chapter 3956, and the Cauchy completion adjunction of chapter 1456. The packet names the located unit, located extension, uniqueness ledger, and non-escape route without adding quotient completions, selected limits, countable choice, or ambient Real completeness.

Definition 8226.1 (Located completion adjunction carrier)**.** *A* $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, U, E, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, including the displayed metric-completion route and its finite tolerance readback.*
2. *S is the* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *row that seals located Cauchy data before any completion-facing consumer reads the target.*
3. *U is the located unit row from source observations into the located completion packet.*
4. *E is the located extension row for maps whose source-side reads preserve the displayed locatedness data.*
5. *Q is the uniqueness ledger comparing two located extension rows only after their source, seal, and unit rows have been displayed.*
6. *A is the adjunction row, read through the sibling* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *and specialized to the located carrier surface.*
7. *H records componentwise* hsame *transport for metric-completion source, located seal, unit, extension, uniqueness, adjunction, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located source request through M, S, U, E, Q, A, H.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{LocatedCompletionSeal}^{\uparrow}$, $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCompletionAdjunction}^{\uparrow}}$ *row.*
The carrier contains no quotient completion class, selected representative, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 8226.2 (Located completion adjunction NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ *carrier*

$$L = (M, S, U, E, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, metric-completion source exposure through M, located seal exposure through S, located unit exposure through U, located extension exposure through E, uniqueness through Q, adjunction routing through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8226.1. The rows M, S, U, E, Q, A give the metric-completion source, located seal, unit, extension, uniqueness, and adjunction data, while H, C, P, N give transport, replay, provenance, and local naming.

The located route is read in order. A consumer first sees the metric-completion source row M, then the located seal row S, and only then the unit and extension rows U, E.

Uniqueness is local to the same packet. The row Q compares located extensions after the source and seal rows are displayed, and A records the adjunction route through the sibling completion adjunction rather than through an external completed object.

The structural rows H, C, P, N transport, replay, source, and name exactly this tuple. Since no carrier coordinate stores quotient completions, selected representatives, selected limits, countable choice, host equality, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8226.3 (Located completion adjunction unit-counit route)**.** *Every unit-side or counit-side read of an accepted* $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ *packet factors through the same displayed route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{LocatedCompletionSeal}^{\uparrow} \longmapsto \mathsf{LocatedUniformCompletion}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionFunc}$$

together with the unit row U, extension row E, uniqueness ledger Q, adjunction row A, componentwise transport H, displayed replay C, provenance P, and local naming row N of Definition 8226.1. Thus the unit and counit are read as located finite routes, not as quotient-completion equality or host-level natural isomorphism.

Proof. Start with the NameCert obligations of Definition 8226.2. They identify the accepted packet with the displayed rows $M, S, U, E, Q, A, H, C, P, N$.

For a unit-side read, use M and S before U. The metric-completion source row fixes the completion-facing boundary, and the located seal keeps located Cauchy data visible before the source observation enters the completion packet.

For a counit-side read, use E, Q, A after the same source and seal rows. The extension row is compared only by the uniqueness ledger, and the adjunction row is routed through the sibling $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ surface specialized to located data.

The rows H, C, P, N preserve the same finite route for transport, replay, provenance, and local naming. Hence neither side of the unit-counit read can escape to quotient-completion equality, a host-level natural isomorphism, countable choice, or ambient completeness. □

Theorem 8226.4 (Located completion adjunction non-escape)**.** *Every consumer read of an accepted* $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ *packet remains inside the displayed rows* $M, S, U, E, Q, A, H, C, P, N$*. In particular, the packet may expose the metric-completion source, located completion seal, located unit, located extension, uniqueness ledger, adjunction route, transport, replay, provenance, and local naming rows, but it cannot expose quotient completions, selected limits, countable choice, host equality of completed objects, or ambient Real completeness.*

Proof. Project the carrier and apply the obligation theorem. The accepted packet has exactly the rows $M, S, U, E, Q, A, H, C, P, N$, and the local NameCert surface lists no additional completion object.

Apply the unit-counit route theorem. It forces both source-to-completion and completion-to-located-extension reads through the same metric source, located seal, located uniform-completion route, regular-Cauchy completion functor, and Cauchy completion adjunction surface.

Finally inspect the excluded endpoints. Quotient completions, selected limits, countable-choice schedules, host equality of completed objects, and ambient Real completeness are not rows of the carrier and are not introduced by H, C, P, N. Therefore no consumer read can extract them from the located adjunction packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ read must expose a MetricCompletionUp source row, a LocatedCompletionSealUp row, a located unit row, a located extension row, a uniqueness ledger, a CauchyCompletionAdjunctionUp route, componentwise transport, displayed replay, provenance, and local naming before any located-completion universal-property consumer may inspect the packet.

Independence witness: metriccompletion, located-completion-seal, located-uniform-completion, regular-cauchy-completion-functor, cauchy-completion-adjunction: $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly records located sealing, unit-extension comparison, uniqueness, and adjunction routing in one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionAdjunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCompletionAdjunction}^{\uparrow}$ packet is generated from a metric-completion source row, located completion seal, located unit, located extension, uniqueness ledger, adjunction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8226.1, Definition 8226.2, Definition 8226.3, and Definition 8226.4 seed the located completion universal-property surface over the MetricCompletion, LocatedCompletionSeal, LocatedUniformCompletion, RegularCauchyCompletionFunctor, and CauchyCompletionAdjunction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionAdjunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected limits, countable choice, host equality of completed objects, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located unit routing, located extension routing, uniqueness-ledger exhaustion, adjunction-row compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionAdjunctionUp.

Chapter 8227

A Concrete Naming Certificate for $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ packet records the finite modulus data used by the diagonal-limit route of chapter 9238. It sits beside the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy-limit boundary of chapter 386. The packet names a finite modulus certificate for the Real-completeness diagonal theorem; it is not a selected host limit, quotient stream equality, countable-choice schedule, open-cover compactness argument, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8227.1 (Diagonal limit modulus certificate carrier)**.** *A* $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ *carrier is a finite packet*

$$M = (S, T, D, Q, L, R, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row, recording the displayed regular Cauchy input over* $\mathsf{RegSeqRat}^{\uparrow}$*;*
2. *T is the finite tail-threshold row, recording the modulus requests and selected threshold indices as displayed data rather than as a choice functional;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic bounds used to compare source terms with the diagonal representative;*
4. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording finite observations of the source sequence at the threshold indices;*
5. *L is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row, recording the regular rational name assembled from those windows;*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row attached to the diagonal readback;*
7. *U is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, recording that the same finite route is eligible for the limit-seal consumer;*
8. *H records componentwise* hsame *transport for the source sequence, threshold row, dyadic tolerance row, stream windows, diagonal readback, Real seal, Cauchy-limit seal, replay, provenance, and local naming;*

9. *C records displayed* Cont *replay through $S, T, D, Q, L, R, U, H, P, N$;*
10. *P is the* Pkg *provenance over* $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected infinite subsequence, a quotient of regular rational streams, a host real limit, countable choice, open-cover compactness, or ambient Real completeness.

Theorem 8227.2 (Diagonal limit modulus certificate NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ *carrier*

$$M = (S, T, D, Q, L, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}}$ *surface consists exactly of source-sequence admission, finite tail-threshold admission, dyadic tolerance exactness, finite stream-window exposure, regular rational diagonal readback, terminal Real-facing seal, Cauchy-limit seal compatibility, component-wise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8227.1. The source and modulus data are exactly S, T, D: the regular Cauchy input, finite threshold schedule, and dyadic tolerance ledger.

The diagonal readback route is then Q, L, R, U. Stream windows expose only finite observations at the displayed thresholds, the regular rational row assembles the diagonal readback from those observations, the Real row seals that readback, and the Cauchy-limit row records the same route as a limit-seal boundary.

The structural rows H, C, P, N transport, replay, source, and name this finite route. Since the carrier has no coordinate for a selected host limit, quotient stream equality, countable-choice schedule, open-cover compactness, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8227.3 (Diagonal limit modulus certificate seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ *packet reads the diagonal modulus route through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{CauchyLimitSeal}^{\uparrow}.$$

Concretely, the consumer must expose the source-sequence row S, tail-threshold row T, dyadic tolerance row D, finite window row Q, and regular rational diagonal row L before it may read the Real seal R or the Cauchy-limit seal U.

Proof. Start from an accepted carrier $M = (S, T, D, Q, L, R, U, H, C, P, N)$. The only source sequence visible to the consumer is S, and the only modulus data are the finite threshold and dyadic tolerance rows T, D.

The stream-window row Q reads those displayed thresholds, and the regular rational row L is downstream of Q. Hence a Real-facing consumer reaches R only after the finite windows and regular rational diagonal readback have been exposed.

The Cauchy-limit seal row U consumes the same route and has no second source coordinate. Transport, replay, provenance, and local naming preserve the route, so no downstream reader obtains a selected host limit, quotient stream equality, countable-choice schedule, open-cover compactness argument, or ambient completeness theorem from this packet. □

Falsifiable prediction: A Real-completeness consumer that exports a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ limit read from $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ while omitting one of the displayed source-sequence, threshold, dyadic tolerance, stream-window, regular readback, transport, replay, provenance, or local naming rows is not reading this packet.

Independence witness: Nearest siblings: streamname, regseqrat, dyadicratcore, real, cauchylimitseal. The carrier is not bijective to StreamNameUp because it adds threshold and dyadic modulus rows; not to RegSeqRatUp because regular readback is downstream of the diagonal schedule; not to DyadicRatCoreUp because dyadic data only supplies tolerance evidence; not to RealUp because the Real row is a terminal handoff; and not to CauchyLimitSealUp because the limit seal consumes the displayed diagonal modulus route rather than replacing it.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8227.1, Definition 8227.2, and Definition 8227.3 seed the finite diagonal modulus route from regular rational source data to a Real-facing and Cauchy-limit seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected host limits, quotient stream equality, countable choice, open-cover compactness, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-sequence stability, tail-threshold exactness, dyadic tolerance exactness, stream-window readback, RegSeqRat diagonal handoff, Real handoff, CauchyLimitSeal compatibility, transport, replay, provenance, and local naming for BEDC.Derived.DiagonalLimitModulusCertificateUp.

Constructive story: A $\mathsf{DiagonalLimitModulusCertificate}^{\uparrow}$ packet is generated from a regular rational source sequence, finite tail thresholds, dyadic tolerance rows, StreamName windows, RegSeqRat diagonal readback, a Real seal, a CauchyLimitSeal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8228

Fitting Lemma Name Certificate Construction

Following Fitting and Atiyah–Macdonald, the $\mathsf{FittingLemma}^{\uparrow}$ packet records the finite decomposition surface for an endomorphism of a Noetherian module. It sits over the module carrier of chapter 46, the Noetherian finiteness surface of chapter 3648, the linear-map rows of chapter 48, and the Fitting-ideal packet of chapter 6447; it does not import an ambient abelian-category splitting theorem, quotient equality, or choice of complements.

Definition 8228.1 (Fitting lemma carrier)**.** *A* $\mathsf{FittingLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, N, E, I, K, T, S, R, H, C, P, L)$$

with the following displayed rows:

1. *M is the underlying* $\mathsf{Module}^{\uparrow}$ *row, including the visible additive and scalar-operation classifier data.*
2. *N is the* $\mathsf{NoetherianModule}^{\uparrow}$ *row for the same module carrier, recording the finite ascending-chain and generator witnesses used by the packet.*
3. *E is the* $\mathsf{LinMap}^{\uparrow}$ *endomorphism row, with source and target both read from M.*
4. *I is the* $\mathsf{FittingIdeal}^{\uparrow}$ *row generated by the displayed determinantal data of E on the finite Noetherian presentation.*
5. *K is the finite kernel-stabilization ledger listing the inspected rows for $\ker(E^n)$ and $\ker(E^{n+1})$.*
6. *T is the finite image-stabilization ledger listing the inspected rows for $\mathrm{im}(E^n)$ and $\mathrm{im}(E^{n+1})$.*
7. *S is the decomposition row asserting that the stabilized kernel and image rows are disjoint on the displayed module coordinates and jointly cover the accepted carrier.*
8. *R records the replay routes from E, N, I, K, T to S, including the finite powers of the endomorphism actually inspected.*
9. *H stores componentwise* hsame *transport for the module, Noetherian, linear-map, Fitting-ideal, stabilization, decomposition, replay, provenance, and local-name rows.*
10. *C stores displayed* Cont *replay for kernel reads, image reads, Fitting-ideal reads, stabilized decomposition reads, and consumer handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{FittingLemma}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{NoetherianModule}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{FittingIdeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{FittingLemma}^{\uparrow}}$ *row.*

The classifier accepts exactly these finite rows. It has no coordinate for an external complement selector, quotient-space equality, classical primary decomposition, or a host theorem proving the Fitting lemma outside the displayed packet.

Theorem 8228.2 (Fitting lemma NameCert obligations)**.** *For every accepted* $\mathsf{FittingLemma}^{\uparrow}$ *carrier* $Q = (M, N, E, I, K, T, S, R, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{FittingLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, module-row exposure through M, Noetherian-row exposure through N, endomorphism exposure through E, Fitting-ideal exposure through I, kernel and image stabilization through K and T, decomposition replay through S and R, componentwise transport through H, continuation replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier and read the structural rows. The module row, Noetherian row, linear endomorphism row, and Fitting-ideal row are the only algebraic coordinates available to the classifier.

Inspect the stabilization ledgers next. The kernel ledger K and image ledger T list the finite powers of E whose stabilized rows are used by the decomposition surface.

Finally replay the decomposition row. The route R witnesses that S is consumed only after the displayed stabilization rows and their transports have been read, while C, P, L replay, source, and name the same packet. No other NameCert obligation is exported. □

Theorem 8228.3 (Fitting lemma kernel-image splitting route)**.** *Every accepted* $\mathsf{FittingLemma}^{\uparrow}$ *carrier exposes a finite route from the* $\mathsf{Module}^{\uparrow}$, $\mathsf{NoetherianModule}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, *and* $\mathsf{FittingIdeal}^{\uparrow}$ *rows to the stabilized kernel-image decomposition. A consumer of the decomposition row must first read*

$$M, N, E, I \longmapsto K, T \longmapsto S$$

together with the replay, transport, provenance, and local-name rows R, H, C, P, L.

Proof. Read the algebraic base rows M, N, E, I. These rows identify the module, its Noetherian finiteness witness, the displayed endomorphism, and the Fitting-ideal data attached to that endomorphism.

The stabilization rows are downstream from this base. The packet records only the finite kernel rows in K and finite image rows in T that are reached by replaying powers of E through the Noetherian and Fitting-ideal ledgers.

The splitting row S is then consumed through R, H, C, P, L. Since no complement selector or host quotient equality is a coordinate of Q, the consumer obtains only the displayed kernel-image route and no stronger ambient decomposition principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FittingLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FittingLemma}^{\uparrow}$ packet is generated from module, Noetherian, endomorphism, Fitting-ideal, kernel-stabilization, image-stabilization, decomposition, replay, transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FittingLemma}^{\uparrow}}$, seed)

Scope closed: Definition 8228.1, Definition 8228.2, and Definition 8228.3 bind the finite module-endomorphism decomposition packet and its kernel-image replay route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FittingLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Lean verification, bridge checking, classical complement choice, quotient equality, primary decomposition, or an ambient abelian-category splitting theorem.

Upgrade path: Obligation closure requires checked carrier admission, stabilization-row exposure, kernel-image replay, transport, provenance, naming, and non-escape for the same finite packet.

Chapter 8229

A Concrete Naming Certificate for $\mathsf{SpecializationTopology}^{\uparrow}$

Following Kolmogorov's T_0 separation reading and Alexandroff's order-theoretic presentation of finite topologies, the $\mathsf{SpecializationTopology}^{\uparrow}$ packet records the classical route from displayed opens to the specialization preorder as finite BEDC NameCert data. It sits after the indexed-open carrier of chapter 88, the preorder carrier of chapter 52, and the observable-open classifier of chapter 1860. The packet names only the finite topology/preorder/Sierpinski bridge; it does not import host opens, quotient point equality, excluded middle over opens, selected representatives, open-cover compactness, or ambient point-set topology.

Definition 8229.1 (Specialization topology carrier)**.** *A* $\mathsf{SpecializationTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row, carrying the open indices, finite membership reads, finite-intersection rows, union rows, and boundary rows supplied by chapter 88.*
2. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-truth row, reading each displayed open-membership request as a two-observation classifier rather than as a host proposition.*
3. *R is the* $\mathsf{Preorder}^{\uparrow}$ *specialization row, where $x \preceq y$ is recorded only by the displayed implication route saying that every observable open true at x is also true at y.*
4. *H stores componentwise* hsame *transport for the topology row, Sierpinski row, specialization-preorder row, replay, provenance, and local name.*
5. *C stores displayed* Cont *replay from a topology membership request through S into the specialization row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationTopology}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through T, S, R, H, C, P, N. There is no coordinate for a host lattice of opens, quotient point equality, proof irrelevance for open membership, excluded middle, selected representatives, or open-cover compactness.

Lean4 (definition) `BEDC.Derived.SpecializationTopologyUp.SpecializationTopologyUp`

Theorem 8229.2 (Specialization topology NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationTopology}^{\uparrow}$ *carrier* $K = (T, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, Sierpinski truth, specialization preorder, transport, replay, provenance, and local naming rows; topology exactness through* T*; observable-open exactness through* S*; specialization-preorder exactness through* R*; and provenance exhaustion through* P, N*. No obligation is discharged by host opens, quotient point equality, proof irrelevance, excluded middle over open membership, selected representatives, or open-cover compactness.*

Proof. Project the carrier of Definition 8229.1. The only visible rows are T, S, R, H, C, P, N, so each local obligation must be read from the indexed topology row, the observable-truth row, the specialization-preorder row, or the structural rows.

Topology exactness is supplied by T. The packet can inspect only the named open indices and their displayed closure witnesses, so it has no access to a host lattice of all opens.

Observable-open exactness is supplied by S. Each membership request becomes Sierpinski classifier data before it is used by the preorder row, and therefore remains displayed truth data rather than an ambient proposition.

Specialization exactness is supplied by R. Reflexivity and transitivity are read as preorder data over the observable implication route. The remaining rows H, C, P, N transport, replay, source, and name that finite route, while the refused host objects are absent from the carrier. □

Theorem 8229.3 (Specialization topology Sierpinski-preorder route)**.** *Every consumer that reads a specialization comparison from an accepted* $\mathsf{SpecializationTopology}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{Topology}^{\uparrow} \longmapsto \mathsf{SierpinskiSpace}^{\uparrow} \longmapsto \mathsf{Preorder}^{\uparrow}.$$

The topology row supplies open indices, the Sierpinski row supplies observable truth values for those indices, and the preorder row records only the resulting specialization implication. The route exports no host-open lattice, quotient point equality, selected representative function, excluded middle over opens, proof irrelevance, or open-cover compactness principle.

Proof. Start with T. A specialization read can ask only for open indices and membership requests already displayed by the $\mathsf{Topology}^{\uparrow}$ row.

Replay the request through S. The $\mathsf{SierpinskiSpace}^{\uparrow}$ row turns each open-membership read into observable truth data, so the consumer never receives an ambient truth predicate over all host opens.

Feed those truth rows into R. The $\mathsf{Preorder}^{\uparrow}$ row records the implication route for specialization comparisons and supplies the preorder surface available to downstream order consumers.

Since the carrier contains no quotient point coordinate, selected representative row, excluded-middle row, proof-irrelevance row, or open-cover compactness row, none of those objects can be exported by the route. □

Falsifiable prediction: Every accepted $\mathsf{SpecializationTopology}^{\uparrow}$ read must display a Topology indexed-open row, a SierpinskiSpace observable-truth row, a Preorder specialization row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before any specialization comparison is consumed.

Independence witness: topology, preorder, sierpinski-space: $\mathsf{SpecializationTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries indexed-open rows, observable Sierpinski truth rows, specialization-preorder rows, transport, replay, provenance, and local naming in one finite bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationTopology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8229.1, Definition 8229.2, and Definition 8229.3 seed the finite topology, Sierpinski-truth, and specialization-preorder bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host opens, quotient point equality, proof irrelevance, excluded middle over opens, selected representatives, open-cover compactness, Lean NameCert verification, or bridge checking for $\mathsf{SpecializationTopology}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, topology exactness, Sierpinski observable-truth exactness, specialization-preorder exactness, transport, replay, provenance, and local naming for BEDC.Derived.SpecializationTopologyUp.

Constructive story: A $\mathsf{SpecializationTopology}^{\uparrow}$ packet is generated from a Topology indexed-open row, a SierpinskiSpace observable-truth row, a Preorder specialization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8230

A Concrete Naming Certificate for CuspNeighborhood$^{\uparrow}$

Following Beardon and Ratcliffe on hyperbolic manifolds, the CuspNeighborhood$^{\uparrow}$ packet records a finite horoball-based cusp-neighborhood certificate at a displayed parabolic boundary end. It sits beside the horoball surface of chapter 7843, the upper-half-plane model of chapter 7806, the discrete action rows of chapter 7282, the metric readback of chapter 3474, and the terminal scalar seal of chapter 38. The packet names only displayed cusp data; it is not a quotient hyperbolic surface, a selected fundamental domain, a completed ideal boundary object, or an ambient finite-volume theorem.

Definition 8230.1 (Cusp-neighborhood carrier)**.** *A* CuspNeighborhood$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, H, S, M, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the parabolic boundary-point row, read in the same upper-half-plane boundary surface used by* PoincareHalfPlane$^{\uparrow}$*.*
2. *H is the* HyperbolicHoroball$^{\uparrow}$ *row, recording the horoball attached to B together with its displayed height or Busemann-threshold data.*
3. *S is the boundary-stabilizer row, recording the finite* FuchsianGroupAction$^{\uparrow}$ *word data that preserves the same boundary point and horoball side.*
4. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-threshold readback row used to compare points with the horoball boundary.*
5. *R is the terminal* Real$^{\uparrow}$ *seal for the threshold and metric comparison values.*
6. *E records the local cusp-neighborhood export row, namely the finite region accepted after the boundary point, horoball threshold, stabilizer word, metric readback, and Real seal have all been displayed.*
7. *T records componentwise* hsame *transport for B, H, S, M, R, E and the local certificate row.*
8. *C records displayed* Cont *replay from the upper-half-plane boundary point through the horoball, stabilizer, metric threshold, Real seal, export, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* CuspNeighborhood$^{\uparrow}$*,* HyperbolicHoroball$^{\uparrow}$*,* PoincareHalfPlane$^{\uparrow}$*,* FuchsianGroupAction HyperbolicMetric$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CuspNeighborhood}^{\uparrow}}$ *row.*
The carrier has no coordinate for a quotient cusp surface, a selected fundamental domain, global discreteness of the ambient group, finite-volume completion, or a host ideal-boundary object.

Theorem 8230.2 (Cusp-neighborhood NameCert obligations)**.** *For every accepted* $\mathsf{CuspNeighborhood}^{\uparrow}$ *carrier*

$$K = (B, H, S, M, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CuspNeighborhood}^{\uparrow}}$ *surface consists exactly of boundary-point admission through B, horoball and cusp-threshold exposure through H, boundary-stabilizer exposure through S, hyperbolic metric-threshold readback through M, Real sealing through R, local cusp-neighborhood export through E, componentwise* hsame *transport through T, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The first five rows expose the parabolic boundary point, the horoball threshold, the stabilizer word data, the metric readback, and the terminal Real seal.

Read the local export row E only after those rows have been displayed. The export is therefore a cusp-neighborhood row over a named boundary end and horoball threshold, not an ambient quotient surface.

Finally read T, C, P, N. These structural rows transport, replay, source, and name the same finite data. Since the carrier has no coordinate for quotienting by the full group, selecting a fundamental domain, completing the ideal boundary, or proving finite volume, the listed rows exhaust the NameCert obligations. □

Theorem 8230.3 (Cusp-neighborhood horoball handoff)**.** *Every downstream consumer of an accepted* $\mathsf{CuspNeighborhood}^{\uparrow}$ *packet receives the horoball route*

$$\mathsf{PoincareHalfPlane}^{\uparrow},\ \mathsf{HyperbolicHoroball}^{\uparrow},\ \mathsf{FuchsianGroupAction}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

only through the displayed boundary point B, horoball row H, boundary-stabilizer row S, metric-threshold row M, Real seal R, export row E, transport row T, replay row C, provenance row P, and local naming row N. The handoff supplies a finite cusp-neighborhood certificate, not a quotient manifold cusp, a selected fundamental region, a global parabolic subgroup theorem, or an ambient finite-volume claim.

Proof. Start from the boundary row B and the horoball row H. They are read through the upper-half-plane and horoball sibling certificates before any cusp-neighborhood export is available.

Next read the stabilizer and metric rows. The row S records only displayed action-word data preserving the same boundary side, and M, R record the threshold comparison and its Real seal.

The handoff row E is therefore downstream of B, H, S, M, R. The structural rows T, C, P, N preserve the same finite route, so a consumer receives only the listed rows and none of the excluded quotient, selection, global-group, or finite-volume objects. □

Falsifiable prediction: Any accepted $\mathsf{CuspNeighborhood}^{\uparrow}$ consumer that reaches a cusp-neighborhood export without first displaying $B, H, S, M, R, T, C, P, N$ fails to read the accepted carrier.

Independence witness: hyperbolic-horoball, poincare-half-plane, fuchsian-group-action, hyperbolic-metric, real: no listed sibling is bijective to $\mathsf{CuspNeighborhood}^{\uparrow}$ because only this packet binds a boundary point, horoball threshold, boundary stabilizer, hyperbolic metric readback, Real seal, cusp export, transport, replay, provenance, and local NameCert row in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CuspNeighborhood}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CuspNeighborhood}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8230.1, Definition 8230.2, and Definition 8230.3 seed the horoball cusp-neighborhood route over boundary-point, horoball-threshold, boundary-stabilizer, metric-threshold, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CuspNeighborhood}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient cusp surfaces, selected fundamental domains, global parabolic subgroup theorems, completed ideal-boundary objects, ambient finite-volume claims, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-point exactness, horoball-threshold exactness, boundary-stabilizer exactness, metric-threshold readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CuspNeighborhoodUp.

Constructive story: A $\mathsf{CuspNeighborhood}^{\uparrow}$ packet is generated from a parabolic boundary-point row, horoball threshold row, boundary-stabilizer row, HyperbolicMetric readback row, Real seal row, local cusp-neighborhood export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8231

A Concrete Naming Certificate for $\mathsf{RemezExchange}^{\uparrow}$

Following Cheney, Approximation Theory, the $\mathsf{RemezExchange}^{\uparrow}$ packet records the finite exchange step used by constructive minimax approximation on a displayed real interval. It consumes the density route of chapter 1634, the alternation surface of chapter 6013, the denominator-bounded comparison boundary of **??**, the compact interval source of chapter 7620, the finite interval-uniform-continuity readback of chapter 6163, and the coefficient carrier of chapter 50. The packet names one displayed exchange move; it is not a host optimizer, an ambient supremum operator, a selected dense enumeration, or a completed quotient function space.

Definition 8231.1 (Remez exchange carrier)**.** *A* $\mathsf{RemezExchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, S, P, E, \Sigma, A, B, G, H, C, Q, N)$$

with the following displayed rows:

1. *I is the compact interval sample row, read through* $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$ *with endpoint containment and interval seals already displayed.*
2. *S is the ordered sample table, listing the finitely many inspected points and the local mesh cells through which each point is read.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *approximant row, including the coefficient window and degree budget used by the exchange step.*
4. *E is the residual-value row for the inspected target-minus-approximant values on S.*
5. *Σ is the residual-sign row, recording the alternating sign pattern carried by the current sample.*
6. *A is the alternation-window row, imported from the* $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ *surface when the displayed signs and magnitudes form a minimax window.*
7. *B is the dyadic sup-norm budget row, giving the finite upper and comparison tolerances used to judge the exchange.*
8. *G is the exchange-step row, replacing one displayed sample cell by a stronger residual cell while preserving the ordered interval route.*
9. *H records componentwise* hsame *transport for the interval, sample, polynomial, residual, sign, alternation, budget, exchange, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a minimax request through $I, S, P, E, \Sigma, A, B, G, H, Q, N$.*
11. *Q is the* Pkg *provenance over* $\mathsf{RemezExchange}^{\uparrow}$, $\mathsf{WeierstrassApproximation}^{\uparrow}$, $\mathsf{ChebyshevEquioscillation}^{\uparrow}$, $\mathsf{RationalBestApproximation}^{\uparrow}$, $\mathsf{ConstructiveRealClosedInterval}^{\uparrow}$, $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RemezExchange}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host supremum, a selected global optimizer, an unbounded sample search, quotient real-function equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8231.2 (Remez exchange NameCert obligations)**.** *For every accepted* $\mathsf{RemezExchange}^{\uparrow}$ *carrier*

$$X = (I, S, P, E, \Sigma, A, B, G, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{RemezExchange}^{\uparrow}}$ *surface consists exactly of compact interval sample admission, polynomial approximant admission, residual-value readback, residual-sign admission, alternation-window compatibility, dyadic sup-norm budget exposure, exchange-step exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8231.1. The displayed tuple exposes the interval, sample, polynomial, residual, sign, alternation, budget, exchange, transport, replay, provenance, and local naming rows.

The approximation rows then share one finite source route. The interval row I supplies the only endpoint and containment data, the sample row S lists the inspected cells, and the polynomial row P supplies the only approximant admitted by the packet.

The residual and alternation obligations are finite. The rows E and Σ attach residual magnitudes and signs to the inspected sample cells, while A reads the equioscillation window through the sibling alternation certificate. The dyadic budget B is the only sup-norm-facing tolerance available to the exchange.

Finally the exchange row G records the displayed replacement cell. The structural rows H, C, Q, N transport, replay, source, and name the same route, so no host optimizer, ambient supremum, unbounded search, quotient equality, choice principle, or completeness coordinate can enter the local certificate. □

Theorem 8231.3 (Remez exchange equioscillation window)**.** *Every alternation read of an accepted* $\mathsf{RemezExchange}^{\uparrow}$ *carrier factors through the finite route*

$$I, S, P, E, \Sigma, B \longmapsto A \longmapsto G, H, C, Q, N.$$

Thus the equioscillation window consumed by the exchange is witnessed by the displayed sample order, residual signs, dyadic sup-norm budget, and sibling $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ *row, not by an ambient compactness theorem or an uninspected function-space norm.*

Proof. Begin with the carrier inventory. The only data available before the alternation row are the interval row I, the ordered sample table S, the approximant row P, the residual row E, the sign row Σ, and the dyadic budget B.

Apply Definition 8231.2. It forces the alternation-window row A to read the same interval, sample, polynomial, residual, sign, and dyadic-budget rows that were admitted by the carrier.

Consequently an exchange consumer can inspect the equioscillation surface only after those rows are fixed. The exchange, transport, replay, provenance, and local name rows then package the same finite route, with no additional compactness or host norm coordinate available. □

Theorem 8231.4 (Remez exchange finite improvement)**.** *Every finite improvement claim exported by an accepted* $\mathsf{RemezExchange}^{\uparrow}$ *packet is read from the displayed exchange row G together with the dyadic budget B and the alternation-window row A. The handoff may state that the displayed replacement cell improves the inspected residual budget on the named sample route, but it exports no global minimax optimizer, no selected limiting subsequence, no quotient function-space equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 8231.3. It places the interval, sample, polynomial, residual, sign, and dyadic-budget rows before the alternation window is available.

The exchange row G can therefore compare only the displayed replacement cell against that same finite window. Its improvement content is a budget comparison over B and A, not a statement about all possible sample sets or a completed function space.

The provenance row records the relationship with the Weierstrass, rational-best, interval-uniform-continuity, and polynomial chapters. Those siblings supply density, comparison, interval readback, and coefficient data, but none of them adds a carrier coordinate for a host optimizer or selected limiting process inside $\mathsf{RemezExchange}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{RemezExchange}^{\uparrow}$ read that claims an exchange improvement must display the compact interval sample row, ordered sample table, polynomial approximant row, residual-value row, residual-sign row, alternation-window row, dyadic sup-norm budget, exchange-step row, hsame transport, Cont replay, Pkg provenance, and local NameCert before the improvement conclusion is inspected.

Independence witness: weierstrass-approximation, chebyshev-equioscillation, rational-best-approximation, constructive-real-closed-interval, real-interval-uniform-continuity, polynomial: $\mathsf{RemezExchange}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite exchange-step row, a residual-sign row, an alternation-window row, and a dyadic sup-norm budget over one displayed compact interval sample.

Closure status for $\mathsf{NameCert}_{\mathsf{RemezExchange}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RemezExchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8231.1, Definition 8231.2, Definition 8231.3, and Definition 8231.4 seed the finite Remez exchange carrier, local obligation surface, alternation-window route, and inspected improvement handoff for constructive minimax approximation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RemezExchange}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global minimax existence, host suprema, selected optimizers, limiting exchange convergence, quotient function spaces, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sample-window compatibility, polynomial approximant admission, residual readback, alternation-window compatibility, dyadic budget soundness, exchange-step exactness, transport, replay, provenance, and local naming for BEDC.Derived.RemezExchangeUp.

Constructive story: A $\mathsf{RemezExchange}^{\uparrow}$ packet is generated from compact interval sample rows, an ordered sample table, a displayed polynomial approximant row, residual values, residual signs, an alternation-window row, a dyadic sup-norm budget, one exchange-step row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8232

A Concrete Naming Certificate for $\mathsf{SequentialCompletionMorphism}^{\uparrow}$

Following Bishop and Bridges' constructive completion language, the $\mathsf{SequentialCompletionMorphism}^{\uparrow}$ packet records the finite map-level route between displayed sequential completion packets. It sits between the source and target surfaces of chapter 1683, the extension surface of chapter 1632, the regular rational readback of chapter 301, the finite observation schedule of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed completion morphism route; it does not import quotient streams, selected representatives, host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8232.1 (Sequential completion morphism carrier)**.** *A* $\mathsf{SequentialCompletionMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, F, W, D, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{SequentialCompletion}^{\uparrow}$ *packet whose finite stream, regular rational, dyadic tolerance, criterion, and* $\mathsf{Real}^{\uparrow}$ *seal rows are being mapped.*
2. *T is the target* $\mathsf{SequentialCompletion}^{\uparrow}$ *packet receiving the mapped completion data.*
3. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible map row, displaying how regular rational source windows are sent to target regular rational windows.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to inspect the source and target names before any completed seal is read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the displayed map estimates.*
6. *U is the* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *row that exposes the extension handoff associated to F.*
7. *H stores componentwise* hsame *transport for the source, target, map, window, dyadic, extension, replay, provenance, and naming rows.*
8. *C stores displayed* Cont *replay from a morphism request through $S, T, F, W, D, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialCompletionMorphism}^{\uparrow}$, $\mathsf{SequentialCompletion}^{\uparrow}$, $\mathsf{CauchyContinuousExtensio}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletionMorphism}^{\uparrow}}$ *row.*

The classifier compares accepted morphism packets only through $S, T, F, W, D, U, H, C, P, N$. No quotient stream representative, host function equality, selected limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness witness, or bridge checker is part of the carrier.*

Theorem 8232.2 (Sequential completion morphism stability)**.** *For an accepted* $\mathsf{SequentialCompletionMorphism}^{\uparrow}$ *carrier $M = (S, T, F, W, D, U, H, C, P, N)$, the map row F is stable componentwise under the transport row H and the replay row C. Any source-window read displayed by W and measured by D is transported to the same target* $\mathsf{SequentialCompletion}^{\uparrow}$ *row T before the extension row U or any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Project the carrier of Definition 8232.1. The source and target completion packets are S and T, while F, W, D, U display the regular-rational map, finite windows, tolerance ledger, and extension handoff.

Read H componentwise. It transports the source packet, target packet, map row, window schedule, dyadic ledger, and extension row together, so the map row cannot move independently of the completion packets it relates.

Now apply the replay row C. The replay has no coordinate except the route through $S, T, F, W, D, U, H, P, N$, and therefore every transported map read is replayed through the same finite completion-morphism packet before the local name N is exported. □

Theorem 8232.3 (Sequential completion morphism extension route)**.** *Every extension consumer of an accepted* $\mathsf{SequentialCompletionMorphism}^{\uparrow}$ *carrier must pass through*

$$S, T, F, W, D, U$$

before reading any $\mathsf{Real}^{\uparrow}$ *seal. The consumer receives a displayed map-level handoff from the source sequential completion packet to the target sequential completion packet, together with its regular-rational compatibility, finite stream windows, dyadic tolerance ledger, and Cauchy-continuous extension row. It receives no quotient stream, host function equality, selected limit, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion, or bridge-checking certificate.*

Proof. Start with the carrier projection. The only extension-facing rows are the source completion packet S, target completion packet T, map row F, window schedule W, tolerance ledger D, and Cauchy-continuous extension row U.

Apply the stability theorem, Definition 8232.2. It keeps F attached to the same source and target completion rows through H and C, so a consumer cannot bypass the displayed map route by reading a completed Real seal first.

Finally use the local provenance and naming rows P, N. They package only the finite route S, T, F, W, D, U, H, C and contain no coordinate for quotient streams, host function equality, selected limits, countable choice, ambient completeness, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletionMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompletionMorphism}^{\uparrow}$ packet is generated from source and target SequentialCompletionUp rows, a RegSeqRatUp-compatible map row, StreamName window schedules, DyadicRatCore tolerance ledgers, a CauchyContinuousExtensionUp row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialCompletionMorphism}^{\uparrow}}$, seed)

Scope closed: Definition 8232.1, Definition 8232.2, and Definition 8232.3 seed the finite map-level route between displayed SequentialCompletionUp packets through RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, CauchyContinuousExtensionUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialCompletionMorphism}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, host function equality, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, map-row stability, extension-consumer routing, transport exactness, replay exactness, provenance, and local naming for BEDC.Derived.SequentialCompletionMorphismUp.

Chapter 8233

A Concrete Naming Certificate for $\mathsf{ChoiceSequenceSpread}^{\uparrow}$

Following Brouwer on choice sequences and spreads, and Bishop–Bridges on lawlike finite approximations to constructive sequences, the $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ packet names the finite prefix surface through which a choice sequence is admitted by a spread law before any infinite branch, choice function, or completed real object is exposed. It sits after the $\mathsf{SpreadLaw}^{\uparrow}$ admission row of chapter 1532, the finite observation schedule of chapter 126, the list spine of chapter 36, and the Boolean endpoint discipline of chapter 32. The packet is a finite BEDC carrier for prefix inspection; it is not countable choice, not a selected infinite branch, and not an ambient fan theorem.

Definition 8233.1 (Choice-sequence spread carrier)**.** *A* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, L, A, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *endpoint row used by each displayed binary choice.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *finite-prefix row, listing the inspected Boolean prefixes and their parent-prefix inclusions.*
3. *A is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row, recording that every displayed prefix in L is accepted by the same spread law and that accepted extensions remain in the displayed prefix tree.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite moments at which the choice sequence is observed.*
5. *W is the prefix-window ledger aligning the StreamName observations with the finite List prefixes admitted by A.*
6. *R is the finite readback row, saying which Boolean endpoint in B is read at each displayed position of the schedule.*
7. *H records componentwise* hsame *transport for $B, L, A, S, W, R, C, P, N$.*
8. *C records displayed* Cont *replay of the finite route*

$$B \longmapsto L \longmapsto A \longmapsto S \longmapsto W \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{SpreadLaw}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ChoiceSequenceSpread}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, L, A, S, W, R, H, C, P, N$. It has no coordinate for a selected infinite branch, countable choice, quotient equality of streams, fan compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8233.2 (Choice-sequence spread NameCert obligations)**.** *For every accepted* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ *carrier*

$$Q = (B, L, A, S, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChoiceSequenceSpread}^{\uparrow}}$ *surface consists exactly of carrier admission for the Boolean endpoint row, List prefix row, SpreadLaw admission row, StreamName schedule row, prefix-window ledger, finite readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read of the packet factors through those displayed rows.*

Proof. Project the carrier of Definition 8233.1. The accepted tuple exposes exactly $B, L, A, S, W, R, H, C, P, N$, so the local certificate has no hidden row from which to read a branch selector or a host-level sequence.

The prefix data are finite. The $\mathsf{Bool}^{\uparrow}$ row supplies the endpoints, the $\mathsf{List}^{\uparrow}$ row lists the inspected prefixes, and the $\mathsf{SpreadLaw}^{\uparrow}$ row admits only those displayed prefixes and their displayed extensions.

The schedule data are finite as well. The $\mathsf{StreamName}^{\uparrow}$ row names the observed positions, while W and R align those positions with the displayed prefix tree and endpoint reads.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for countable choice, quotient stream equality, fan compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8233.3 (Choice-sequence spread prefix closure)**.** *Every accepted* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ *carrier is closed under displayed prefix restriction: if a consumer reads a prefix from the ledger L through the window W, then each earlier prefix on the same displayed branch is read through the same* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{SpreadLaw}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

Proof. Read the displayed prefix from the L coordinate. The List row stores the parent-prefix inclusions for the finite prefixes currently under inspection.

Apply the admission row A. Because A is the same $\mathsf{SpreadLaw}^{\uparrow}$ row for the whole packet, restricting to an earlier displayed prefix keeps the read inside the same spread-law admission surface.

The window and schedule rows do not introduce a second branch. They only align the earlier prefix with already displayed $\mathsf{StreamName}^{\uparrow}$ observations and Boolean endpoint reads.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite packet. Thus prefix restriction remains inside the accepted $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ carrier and exports no infinite selector. □

Theorem 8233.4 (Choice-sequence spread StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ *packet must first expose the finite route*

$$\mathsf{Bool}^{\uparrow} \longmapsto \mathsf{List}^{\uparrow} \longmapsto \mathsf{SpreadLaw}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto W \longmapsto R.$$

The schedule is therefore a finite observation handoff from spread-admitted prefixes to StreamName reads, not a selected branch or countable family of choices.

Proof. Start from Definition 8233.2. A StreamName-facing read can cite the packet only after the Boolean endpoint row, List prefix row, and SpreadLaw admission row have been displayed.

The schedule row S is downstream of that admission data. The window ledger W records which displayed prefixes are observed, and the readback row R records the Boolean endpoints read at the displayed positions.

The structural rows H, C, P, N preserve this same route. They do not add a selector for a full infinite branch, a countable-choice function, or quotient equality of streams. □

Theorem 8233.5 (Choice-sequence spread nonescape)**.** *No consumer of an accepted* $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ *carrier can extract anything beyond the displayed finite packet*

$$B, L, A, S, W, R, H, C, P, N.$$

In particular the packet does not export a chosen infinite branch, countable choice, quotient equality of streams, fan compactness, bar induction, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8233.1 to enumerate the carrier coordinates. The only analytic surface available to consumers is the finite prefix route through B, L, A, S, W, R.

Use Definition 8233.3 to keep every prefix restriction inside that same displayed route. No restricted read creates a new coordinate outside the packet.

Use Definition 8233.4 for StreamName-facing consumers. They receive only the finite observation handoff from spread-admitted prefixes to Boolean endpoint readback.

The remaining rows H, C, P, N are structural. They transport, replay, source, and name the finite packet, but they do not contain a branch selector, a countable-choice schedule, quotient stream equality, fan compactness, bar induction, or a completeness witness for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequenceSpread}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoiceSequenceSpread}^{\uparrow}$ packet is generated from Boolean endpoint rows, List prefix rows, a SpreadLaw admission row, a StreamName schedule row, a prefix-window ledger, finite readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceSpread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8233.1, Definition 8233.2, Definition 8233.3, Definition 8233.4, and Definition 8233.5 seed the finite spread-admitted choice-sequence prefix route and its StreamName handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceSpread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, countable choice, quotient stream equality, fan compactness, bar induction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, SpreadLaw prefix admission, StreamName schedule alignment, finite readback, prefix closure, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ChoiceSequenceSpreadUp.

Chapter 8234

A Concrete Naming Certificate for $\mathsf{RegulaFalsi}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegulaFalsi}^{\uparrow}$ packet records the false-position root-bracketing route as finite NameCert data. It sits after the constructive intermediate-value route of chapter 6410, beside the bisection surface of chapter 2108 and the secant surface of chapter 3776, and before the dyadic, stream, regular-rational, and real-name handoff through chapter 291, chapter 126, chapter 301, and chapter 38. The packet names the finite bracketing rows consumed by root-finding consumers; it does not assert classical completeness, trichotomy, a selected zero, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ equality.

Definition 8234.1 (RegulaFalsi carrier)**.** *A* $\mathsf{RegulaFalsi}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, S, Q, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the sign-changing interval row supplied through* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$*;*
2. *S is the false-position secant row, recording the endpoint values and the chord intersection requested at the current bracket;*
3. *Q is the bracket-retention row, selecting the retained side only through displayed sign data rather than through a host trichotomy;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for endpoint values, chord denominators, and tolerance budgets;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which endpoint and function-value reads are inspected;*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row handed to the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
7. *H records componentwise* hsame *transport for the interval, secant, bracket, dyadic, window, readback, and seal rows;*
8. *C records the displayed* Cont *routes from intermediate-value bracketing through bisection-compatible retention and secant interpolation;*
9. *P is the* Pkg *provenance over* $\mathsf{RegulaFalsi}^{\uparrow}$*,* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$*,* $\mathsf{BisectionMethod}^{\uparrow}$*,* $\mathsf{SecantMethod}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegulaFalsi}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected root, a global convergence theorem, an ambient real equality test, countable choice, or a classical completeness principle.

Theorem 8234.2 (RegulaFalsi NameCert obligations)**.** *For every accepted* $\mathsf{RegulaFalsi}^{\uparrow}$ *carrier* $R = (I, S, Q, D, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegulaFalsi}^{\uparrow}}$ *surface consists exactly of carrier admission, constructive intermediate-value bracket exposure through* I*, false-position secant exposure through* S*, retained-bracket exposure through* Q*, dyadic arithmetic exactness through* D*, finite-window replay through* W*, regular-rational readback and real-seal handoff through* E*, componentwise* hsame *transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8234.1. Its rows list precisely the sign-changing bracket, false-position chord, retained side, dyadic ledger, finite observation window, regular-rational readback, transport, continuation, provenance, and local naming coordinates.

Read the neighbouring certificates in their displayed roles. The constructive intermediate-value packet supplies the sign-changing bracket, the bisection packet supplies the retained-bracket discipline, and the secant packet supplies the chord-intersection row.

The remaining rows control arithmetic and replay. The dyadic ledger, stream window, RegSeqRat readback, and Real seal expose only finite inspected data, so the NameCert obligations are exhausted by the rows of R and cannot add a selected root or host completeness claim. □

Theorem 8234.3 (RegulaFalsi bracket-secant handoff)**.** *Every accepted* $\mathsf{RegulaFalsi}^{\uparrow}$ *consumer that reads the false-position secant row* S *must retain the same sign-changing interval row* I*, bracket-retention row* Q*, dyadic arithmetic row* D*, finite-window row* W*, regular-rational readback row* E*, and structural rows* H, C, P, N*. The handoff from bracketing to secant interpolation therefore exports a finite root-bracketing route, not a selected zero or an unrestricted secant iteration.*

Proof. Start with an accepted packet $R = (I, S, Q, D, W, E, H, C, P, N)$. The carrier makes S visible only as one row of the same finite bracketing packet.

Apply the NameCert obligations of Definition 8234.2. They require the consumer to replay the constructive bracket, retained side, dyadic arithmetic, finite windows, readback, transport, continuation, provenance, and local naming rows whenever the secant row is cited.

Thus a consumer cannot detach the chord intersection from the bracket that licenses it. A detached secant iteration, selected root, or convergence claim would require an additional coordinate not present in Definition 8234.1. □

Theorem 8234.4 (RegulaFalsi Real seal non-escape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by an accepted* $\mathsf{RegulaFalsi}^{\uparrow}$ *packet is exposed only after the route through* $\mathsf{ConstructiveIntermediateValue}^{\uparrow}$*,* $\mathsf{BisectionMethod}^{\uparrow}$*,* $\mathsf{SecantMethod}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *has been replayed. It exports no host real equality, selected completed root, ambient completeness theorem, or countable-choice schedule.*

Proof. Follow the carrier route from I through S, Q, D, W, E. The intermediate-value row starts the bracket, the bisection-compatible row records retention, the secant row records the false-position chord, and the dyadic and stream rows determine the finite inspections.

The RegSeqRat row E is the only route to the terminal Real seal. By Definition 8234.2 and Definition 8234.3, that route remains attached to the same bracket, chord, arithmetic, and replay rows.

Since the carrier has no selected-root coordinate, no quotient-real equality coordinate, no countable-choice schedule, and no ambient completeness row, the Real seal cannot export any of those objects. □

Falsifiable prediction: A claimed $\mathsf{RegulaFalsi}^{\uparrow}$ consumer that reads a Real seal without replaying the displayed bracket, secant, dyadic, StreamName, and RegSeqRat rows refutes the packet.

Independence witness: Nearest siblings: constructive-intermediate-value, bisection-method, secant-method, dyadicratcore, streamname, regseqrat, real. The packet is not bijective to any sibling because it simultaneously stores the bracket, false-position chord, retained side, arithmetic ledger, finite windows, regular-rational readback, and terminal seal rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulaFalsi}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulaFalsi}^{\uparrow}$ packet is generated from a constructive sign-changing bracket, a false-position secant row, a retained-bracket row, dyadic arithmetic, finite observation windows, regular-rational readback, a terminal Real seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulaFalsi}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8234.1, Definition 8234.2, Definition 8234.3, and Definition 8234.4 seed the false-position root-bracketing route over constructive IVT, bisection, secant, dyadic, stream, regular-rational, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulaFalsi}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Lean verification, bridge checking, selected roots, host real equality, classical completeness, trichotomy, countable choice, or ambient RealUp equality.

Upgrade path: Obligation closure requires checked carrier admission, bracket retention, secant handoff, dyadic arithmetic exactness, finite-window replay, RegSeqRat-to-Real non-escape, and TasteGate nontriviality with field faithfulness.

Chapter 8235

A Concrete Naming Certificate for $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$

The $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ packet realizes the Leibniz equality branch of chapter 9382 as finite NameCert data. It consumes the typed example gallery of chapter 756, the typed-example reduction trace of chapter 2330, the Church eliminator regression matrix of chapter 924, and the decidable-boundary route of chapter 749. The packet names the equality type, reflexivity proof, eliminator motive, transport target, checker row, closedness row, reduction row, replay route, provenance row, and local NameCert row for selected typed examples; it does not claim global subject reduction, global confluence, erased host equality, or unrestricted proof irrelevance.

Definition 8235.1 (MetaCIC typed equality eliminator carrier)**.** *A* $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, Q, R, M, L, K, D, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite typed-example row selecting the Leibniz equality examples from the typed example gallery;*
2. *Q is the Leibniz equality type row, including the displayed source type, endpoints, and equality proposition;*
3. *R is the reflexivity proof row for the selected endpoint examples;*
4. *M is the motive row used by the typed eliminator;*
5. *L is the transport target row reached by eliminating the reflexivity proof against M;*
6. *K is the checker row recording the typed-example checker result for Q, R, M, L;*
7. *D is the closedness and reduction row, tying the selected equality proof to the typed-example reduction trace;*
8. *X is the Church eliminator regression row that compares the equality eliminator against neighbouring bool, nat, list, and sum eliminator examples;*
9. *H records componentwise* hsame *transport for the typed example, equality, reflexivity, motive, transport, checker, reduction, and regression rows;*

10. *C records displayed* Cont *replay from an equality-eliminator request through checker, closedness, reduction, transport, regression, provenance, and naming rows;*
11. *P records* Pkg *provenance over* $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$, $\mathsf{TypedExampleGallery}^{\uparrow}$, TypedExampleReduction ChurchEliminatorRegressionMatrix$^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, Q, R, M, L, K, D, X, H, C, P, N$. It has no coordinate for host equality, unrestricted subject reduction, global confluence, proof irrelevance, quotient syntax equality, or an erased equality-eliminator theorem.

Theorem 8235.2 (MetaCIC typed equality eliminator NameCert obligations)**.** *For every accepted* $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ *carrier $E = (T, Q, R, M, L, K, D, X, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}}$ *surface consists exactly of typed-example exposure, Leibniz equality type exposure, reflexivity proof exposure, motive exposure, transport-target exposure, checker exposure, closedness and reduction exposure, Church-eliminator regression exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8235.1. The rows T, Q, R, M, L are the finite equality-eliminator data: a selected typed example, its Leibniz equality type, a reflexivity proof, a motive, and the transported target.

Rows K, D, X bind the same data to the checker, closedness, reduction, and regression surfaces. The checker row keeps the packet in the typed-example gallery, the reduction row ties it to the typed-example reduction trace, and the regression row compares it with the neighbouring Church eliminators without collapsing them into one theorem.

Rows H, C, P, N transport, replay, source, and name the finite surface. Since no row stores host equality, unrestricted subject reduction, global confluence, proof irrelevance, quotient syntax equality, or an erased eliminator theorem, the listed rows exhaust the NameCert obligations. □

Theorem 8235.3 (MetaCIC typed equality eliminator reduction boundary)**.** *Every accepted* $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ *reduction read must retain the same typed-example row T, Leibniz equality row Q, reflexivity row R, motive row M, transport target L, checker row K, reduction row D, regression row X, and structural rows H, C, P, N. The packet therefore realizes selected Leibniz equality examples as finite checker, closedness, reduction, transport, replay, provenance, and NameCert rows, not as a global subject-reduction, confluence, or host-equality result.*

Proof. Begin with the accepted carrier $E = (T, Q, R, M, L, K, D, X, H, C, P, N)$. The reduction row D is not free-standing: it records the closedness and reduction trace for the selected equality data named by T, Q, R, M, L.

Apply the NameCert obligations of Definition 8235.2. They require every equality-eliminator read to replay the checker, closedness, reduction, regression, transport, continuation, provenance, and local naming rows.

Thus a consumer cannot turn the selected Leibniz equality examples into a theorem about all typed reductions. Such a theorem would require additional subject-reduction and confluence rows beyond the carrier of Definition 8235.1. □

Falsifiable prediction: A claimed $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ read that accepts a Leibniz equality eliminator without the typedexamplegallery, typed-example-reduction-trace, church-eliminator-regression-matrix, and metacicdecidableboundary rows refutes the packet.

Independence witness: typedexamplegallery, typed-example-reduction-trace, church-eliminator-regression-matrix, metacicdecidableboundary: the gallery supplies selected typed terms, the trace supplies reduction rows, the regression matrix supplies neighbouring eliminator comparisons, and the decidable boundary supplies checker limits, but none alone carries the equality type, reflexivity proof, motive, transport target, checker, closedness, reduction, regression, replay, provenance, and local NameCert rows.

The Lean follow-up in `lean4/BEDC/Derived/MetacicTypedEqualityEliminatorUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ and FieldFaithful $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ instances preserving $T, Q, R, M, L, K, D, X, H, C, P, N$ before any positive gate may treat this packet as field-faithful.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}$ packet is generated from selected typed-example rows, Leibniz equality types, reflexivity proofs, motives, transport targets, checker rows, closedness and reduction traces, Church-eliminator regression rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8235.1, Definition 8235.2, and Definition 8235.3 seed the finite typed equality eliminator route over selected Leibniz equality examples, checker rows, closedness, reduction, transport, regression, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicTypedEqualityEliminator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim global subject reduction, global confluence, host equality, proof irrelevance, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Leibniz equality type exposure, reflexivity proof exposure, motive and transport exposure, checker stability, reduction-boundary stability, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.MetacicTypedEqualityEliminatorUp.

Chapter 8236

A Concrete Naming Certificate for $\mathsf{FinitePositiveMinimum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FinitePositiveMinimum}^{\uparrow}$ packet records the finite lower-bound fold by which finitely many positive rational radius rows produce one tolerance-only positive row for compact-uniform-modulus consumers. It sits after the rational source of chapter 37, the compact finite-net surface of chapter 129, and the selector route of chapter 914; it is consumed by the uniform-modulus and compact-uniform-continuity surfaces of chapter 369, chapter 2299, and chapter 388. The packet is a displayed finite fold over positive $\mathsf{Rat}^{\uparrow}$ rows, not a host minimum operator, selected subcover, open-cover compactness theorem, or ambient completeness principle.

Definition 8236.1 (FinitePositiveMinimum carrier)**.** *A* $\mathsf{FinitePositiveMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, R, L, K, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite* $\mathsf{ProbeBundle}$ *index row naming the compact-net probes whose local radius rows are inspected.*
2. *R is the family of positive* $\mathsf{Rat}^{\uparrow}$ *radius rows indexed by B.*
3. *L is the positive lower-bound row produced by primitive finite folding over the rows in R.*
4. *K is the comparison ledger proving that L is a positive rational row and lies below every displayed radius row in R.*
5. *H records componentwise* hsame *transport for the probe bundle, radius family, lower-bound row, comparison ledger, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from compact-net selection through the positive radius family and comparison ledger to the lower-bound handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{FinitePositiveMinimum}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePositiveMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, R, L, K, H, C, P, N. It has no coordinate for a host finite-set minimum, open-cover compactness, a selected subcover, quotient equality of compact points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8236.2 (FinitePositiveMinimum NameCert obligations)**.** *For every accepted* $\mathsf{FinitePositiveMinimum}^{\uparrow}$ *carrier*

$$F = (B, R, L, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitePositiveMinimum}^{\uparrow}}$ *surface consists exactly of finite probe-bundle admission, positive* $\mathsf{Rat}^{\uparrow}$ *radius-family admission, lower-bound fold exposure, comparison-ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8236.1. The mathematical rows are the probe bundle B, the positive rational radius family R, the lower-bound fold row L, and the comparison ledger K.

Read the finite fold before any consumer handoff. Since B is finite and R is displayed only over the probes in B, the row L is produced by primitive finite recursion over named positive rational rows rather than by a host minimum operation.

The comparison ledger K records positivity of L and its lower-bound relation to every radius row in R. Transport and replay through H and C preserve the same componentwise route, while P and N source and name the packet. No further NameCert clause is available from compactness, choice, subcover selection, quotient equality, or completeness. □

Theorem 8236.3 (FinitePositiveMinimum compact-uniform-modulus handoff)**.** *Every compact-uniform-modulus consumer of an accepted* $\mathsf{FinitePositiveMinimum}^{\uparrow}$ *carrier receives the positive lower-bound row only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{CompactNetModulusSelector}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{ProbeBundle},\ L, K, H, C, P, N \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

Thus the handoff supports the finite positive rational lower-bound step used by Definition 9237.5 without exporting a host finite-set minimum, selected subcover, open-cover compactness theorem, quotient point equality, countable-choice functional, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the compact-net side. The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{CompactNetModulusSelector}^{\uparrow}$ consumers expose the finite probe bundle B and the positive rational radius rows R before any tolerance-only output can be read.

Apply the NameCert obligations of Definition 8236.2. They force the read through the lower-bound row L and comparison ledger K, with componentwise transport and displayed replay supplied by H and C.

Finally hand the resulting positive row to the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{CompactUniformModulus}^{\uparrow}$ surfaces. The provenance P and local name N keep the handoff attached to the same finite packet, so the downstream $\mathsf{CompactUniformContinuity}^{\uparrow}$ route receives a BEDC lower-bound ledger and not an external minimum, compactness theorem, selected subcover, quotient equality, choice principle, or ambient completeness row. □

Falsifiable prediction: An accepted $\mathsf{FinitePositiveMinimum}^{\uparrow}$ read is refuted if a compact-uniform-modulus consumer receives a tolerance-only lower-bound row without displaying the finite probe bundle, positive rational radius family, comparison ledger, componentwise transport, continuation replay, provenance, and local naming rows.

Independence witness: rat, compactmetric, compactnetmodulusselector, uniformmodulus, compact-uniform-modulus, compactuniformcontinuity: no listed sibling is bijective with $\mathsf{FinitePositiveMinimum}^{\uparrow}$

because this packet jointly carries the finite probe index, positive rational radius family, folded lower-bound row, comparison ledger, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePositiveMinimum}\uparrow}$.

Constructive story: A $\mathsf{FinitePositiveMinimum}^{\uparrow}$ packet is generated from a finite ProbeBundle index row, positive RatUp radius rows, a folded positive lower-bound row, a comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePositiveMinimum}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8236.1, Definition 8236.2, and Definition 8236.3 seed the finite positive rational lower-bound fold consumed by compact-uniform-modulus routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePositiveMinimum}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host finite-set minima, selected subcovers, open-cover compactness, quotient equality of compact points, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive rational family exposure, lower-bound fold exactness, comparison-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.FinitePositiveMinimumUp.

Chapter 8237

A Concrete Naming Certificate for $\mathsf{CauchyModulusLimiter}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusLimiter}^{\uparrow}$ packet records the finite limiter row by which a regular-Cauchy tail modulus is read before a Real-facing consumer receives a bounded tail window. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the shared threshold discipline of chapter 1299. The packet names the limiter as displayed finite NameCert data; it does not choose a host limit, compare quotient streams, use countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8237.1 (Cauchy modulus limiter carrier)**.** *A* $\mathsf{CauchyModulusLimiter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, L, T, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the source Cauchy modulus row attached to a regular-Cauchy rational presentation.*
2. *L is the limiter row, naming the finite bound through which the source modulus is allowed to serve a downstream tail request.*
3. *T is the normalized tail-threshold row read through* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$*.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *readback window opened at the limited tail threshold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite window.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the limiter row with the requested precision.*
7. *H records componentwise* hsame *transport for the modulus, limiter, threshold, readback, dyadic, replay, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a Real-facing tail request through $M, L, T, W, R, D, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusLimiter}^{\uparrow}$, $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusLimiter}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyModulusLimiterUp.CauchyModulusLimiterTasteGate single carrier alignment`

Theorem 8237.2 (Cauchy modulus limiter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusLimiter}^{\uparrow}$ *carrier* $Q = (M, L, T, W, R, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusLimiter}^{\uparrow}}$ *surface consists exactly of source modulus admission, limiter-row exposure, normalized tail-threshold exposure, StreamName readback, RegSeqRat readback, dyadic tolerance comparison, componentwise transport, displayed replay, package provenance, local naming, and non-escape from selected limits, quotient streams, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8237.1. The analytic rows are M, L, T, W, R, D, and the structural rows are H, C, P, N.

The limiter obligation is the ordered read from the source modulus M to the limiter row L and then to the normalized threshold T. This records the finite bound before any downstream tail window can be opened.

The readback obligations are downstream of the threshold. The rows W and R expose the finite StreamName and RegSeqRat data, while D supplies the dyadic comparison needed by a precision request.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier omits selected limits, quotient streams, countable choice, host equality, and ambient completeness, the displayed rows exhaust the local NameCert surface. □

Theorem 8237.3 (Cauchy modulus limiter tail-bound induction)**.** *Every accepted* $\mathsf{CauchyModulusLimiter}^{\uparrow}$ *carrier* $Q = (M, L, T, W, R, D, H, C, P, N)$ *supports finite induction over the limiter row: the base tail request reads* M, L, T*, and each later bounded request preserves the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, transport, replay, provenance, and local naming route. Thus a* $\mathsf{Real}^{\uparrow}$ *consumer sees a bounded tail-modulus certificate rather than a private modulus parameter or an ambient completeness claim.*

Proof. The base step is the first accepted tail request. It reads the source modulus M, the limiter row L, and the normalized threshold T before any window or real-facing seal is available.

For the successor step, assume the current bounded request has already been routed through $M, L, T, W, R, D, H, C, P, N$. A later request reuses the same limiter row and advances only inside the displayed finite tail discipline; the StreamName window W, RegSeqRat readback R, and dyadic comparison ledger D remain downstream of that row.

The structural rows H, C, P, N preserve the route across the induction step. Therefore every bounded tail read factors through the same finite limiter packet, and no excluded limit, quotient, choice, host equality, or completeness coordinate can be extracted. □

Falsifiable prediction: Every accepted $\mathsf{CauchyModulusLimiter}^{\uparrow}$ read must display the source modulus, limiter row, normalized threshold, StreamName window, RegSeqRat readback, DyadicRatCore comparison, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a Real-facing bounded tail request is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchytailthresholdnormalizer: $\mathsf{CauchyModulusLimiter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source modulus, limiter row, normalized tail threshold, finite stream window, regular rational readback, dyadic tolerance comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusLimiter}^{\uparrow}}$**.**

Constructive story: A CauchyModulusLimiter$^{\uparrow}$ packet is generated from a regular-Cauchy source modulus, a finite limiter row, a normalized tail threshold, StreamName and RegSeqRat readbacks, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyModulusLimiter}\uparrow}$, seed)

Scope closed: Definition 8237.1, Definition 8237.2, and Definition 8237.3 seed the finite limiter route for regular-Cauchy tail moduli read by Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyModulusLimiter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, limiter-row exactness, threshold normalization, StreamName and RegSeqRat readback, dyadic tolerance comparison, transport, replay, provenance, and nonescape for BEDC.Derived.CauchyModulusLimiterUp.

Chapter 8238

A Concrete Naming Certificate for $\mathsf{ProjectiveLimitMetric}^{\uparrow}$

Following Bourbaki and Kelley on projective limits of metric spaces, the $\mathsf{ProjectiveLimitMetric}^{\uparrow}$ packet records a finite projection-window presentation of a compatible metric family. It sits beside $\mathsf{InverseLimitMetric}^{\uparrow}$ in chapter 2168: the sibling names finite inverse-limit metric observations, while this chapter names the projective-limit metric carrier that later completion chapters can consume through displayed windows, distance rows, and completion handoff data.

Definition 8238.1 (ProjectiveLimitMetric carrier)**.** *A* $\mathsf{ProjectiveLimitMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, W, \Pi, B, \Delta, K, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite diagram ledger naming the metric-space indices visible to the packet;*
2. *W is the finite projection-window row, including the* $\mathsf{StreamName}^{\uparrow}$ *windows used to read nodes and projection paths;*
3. *Π is the bonding-map compatibility row for every displayed composable projection path inside W;*
4. *B records the* $\mathsf{Met}^{\uparrow}$ *distance rows carried by the displayed nodes;*
5. *Δ records the projective-limit distance read as the finite compatible family of nodewise distance rows;*
6. *K records the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion handoff accepted by downstream completion consumers;*
7. *Q records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows used by the finite distance comparisons;*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the displayed distance values after the regular rational readbacks have been checked;*
9. *E records the sibling dependency on* $\mathsf{InverseLimitMetric}^{\uparrow}$ *and keeps the inverse-limit metric packet as a named dependency rather than as a hidden host object;*
10. *H records componentwise* hsame *transport for the diagram, projection-window, bonding, distance, completion, readback, and seal rows;*

11. *C records displayed* Cont *replay from projection windows to distance reads and from distance reads to the completion handoff;*
12. *P records* Pkg *provenance over* ProjectiveLimitMetric$^\uparrow$, InverseLimitMetric$^\uparrow$, Met$^\uparrow$, CompleteMetric$^\uparrow$, CauchyFilterCompletion$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* NameCert$_{\text{ProjectiveLimitMetric}^\uparrow}$ *row.*

The carrier has no coordinate for a quotient completion, selected representative family, ambient projective-limit object, hidden bonding-map chooser, or global completeness oracle.

Theorem 8238.2 (ProjectiveLimitMetric NameCert obligations)**.** *For every accepted* ProjectiveLimitMetric$^\uparrow$ *carrier $M = (D, W, \Pi, B, \Delta, K, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{ProjectiveLimitMetric}^\uparrow}$ *surface consists of exactly these obligation rows: carrier admission for the finite diagram and projection windows, bonding-map compatibility,* Met$^\uparrow$ *distance-row visibility, projective-limit distance readback,* CompleteMetric$^\uparrow$ *and* CauchyFilterCompletion$^\uparrow$ *completion handoff,* InverseLimitMetric$^\uparrow$ *sibling dependency,* StreamName$^\uparrow$ *window routing,* RegSeqRat$^\uparrow$ *readback,* Real$^\uparrow$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8238.1. The rows D, W, Π expose the finite diagram, projection windows, and bonding-map checks before any distance or completion row is read.

Next read the metric rows. The rows B, Δ, Q, R keep every distance comparison inside the displayed Met$^\uparrow$, RegSeqRat$^\uparrow$, and Real$^\uparrow$ data, while E records the sibling dependency on chapter 2168.

Finally use K, H, C, P, N. The completion handoff is a displayed row, transport is componentwise, replay is routed through C, and provenance and naming are local. No omitted quotient, representative family, ambient projective limit, chooser, or completeness oracle can enter the NameCert surface. □

Theorem 8238.3 (ProjectiveLimitMetric completion handoff)**.** *Every completion consumer of* ProjectiveLimitMetric$^\uparrow$ *must read the finite projection-window route*

$$D, W, \Pi, B, \Delta, K, Q, R, E, H, C, P, N$$

before using the projective-limit metric as a completion input. The exported handoff is the displayed K row over CompleteMetric$^\uparrow$ *and* CauchyFilterCompletion$^\uparrow$*, with distance values sealed through* RegSeqRat$^\uparrow$ *and* Real$^\uparrow$ *and with the* InverseLimitMetric$^\uparrow$ *sibling dependency still visible.*

Proof. Start with the NameCert obligations of Definition 8238.2. They force the consumer to enter through the finite diagram, projection-window, bonding, distance, readback, real-seal, and sibling-dependency rows.

Then project the completion row K. Since K is listed after the finite distance read and before transport, replay, provenance, and naming, a consumer cannot treat completion as a quotient completion equality or as a selected family of representatives.

The structural rows H, C, P, N finish the route. They preserve the displayed packet under componentwise transport, replay only the visible route, record provenance, and name the local certificate. Hence the completion handoff remains a finite projection-window consumer of the projective-limit metric packet. □

Closure status for NameCert$_{\text{ProjectiveLimitMetric}^\uparrow}$.

Constructive story: A ProjectiveLimitMetric$^\uparrow$ packet is generated from finite diagram rows, projection-window rows, bonding-map compatibility, MetricSpace distance rows, completion handoff data, StreamName windows, RegSeqRat readbacks, Real seals, sibling dependency, transport, replay, provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{ProjectiveLimitMetric}\uparrow}$, seed)

Scope closed: Definition 8238.1, Definition 8238.2, and Definition 8238.3 seed the finite projection-window metric packet and its completion-facing handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProjectiveLimitMetric}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public-object export, quotient completion equality, selected representatives, or ambient projective-limit existence.

Upgrade path: Obligation closure requires checked carrier admission, bonding compatibility, distance-row visibility, completion handoff, StreamName routing, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.ProjectiveLimitMetricUp.

Chapter 8239

A Concrete Naming Certificate for $\mathsf{CompactCompleteFixedPoint}^{\uparrow}$

Following Bishop and Bridges on compact metric spaces and Banach's fixed-point theorem, the $\mathsf{CompactCompleteFixedPoint}^{\uparrow}$ packet records the finite certificate route by which a contraction on a compact complete metric source exports a fixed-point orbit and a completion-facing endpoint. The packet sits after the complete metric surface of chapter 128, the compact metric surface of chapter 129, the continuous-map graph row of chapter 130, the contraction packet of chapter 3529, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It is a NameCert reconstruction of the classical compact-complete fixed-point handoff, not a selected global choice function or a host-space equality principle.

Definition 8239.1 (Compact complete fixed-point carrier)**.** *A* $\mathsf{CompactCompleteFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, T, B, O, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row naming the source by finite covers, displayed metric data, and compactness witnesses.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row naming the same source as a completion-facing metric packet.*
3. *T is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the self-map being iterated.*
4. *B is the* $\mathsf{BanachContraction}^{\uparrow}$ *row recording the contraction modulus and the Picard-step boundary.*
5. *O is the finite orbit ledger, exposing the inspected Picard iterates and their contraction-budget comparisons.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the Cauchy orbit estimates.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the regular readback and completion boundary are displayed.*
8. *H records componentwise* hsame *transport for $K, M, T, B, O, Q, E, C, P, N$.*
9. *C records the displayed* Cont *replay of compactness, completeness, continuity, contraction, orbit, readback, and endpoint handoff rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactCompleteFixedPoint}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCompleteFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, T, B, O, Q, E, H, C, P, N$. It has no coordinate for selected fixed points, quotient orbit equality, host metric completion equality, countable choice, or ambient compactness beyond the displayed $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows.*

Lean4 (statement) `BEDC.Derived.CompactCompleteFixedPointUp`

Theorem 8239.2 (Compact complete fixed-point NameCert obligations)**.** *For the carrier of Definition 8239.1, the local* $\mathsf{NameCert}_{\mathsf{CompactCompleteFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (K, M, T, B, O, Q, E, H, C, P, N)$, compact metric source exposure through K, complete metric boundary exposure through M, continuous self-map replay through T, contraction data through B, finite orbit ledger exposure through O, regular rational readback through Q, terminal real seal through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted packet to the eleven displayed rows. The rows K, M, T, B, O, Q, E are the only mathematical content rows, while H, C, P, N provide transport, replay, provenance, and local naming.

Read the source rows first. The row K supplies compact metric data by chapter 129, and M supplies the completion-facing metric boundary by chapter 128. These rows provide finite compact and complete surfaces without adding a separate host-space equality row.

Read the map and orbit rows next. The row T is a $\mathsf{ContinuousMap}^{\uparrow}$ self-map row, B is the contraction row supplied by chapter 3529, and O is the finite Picard-orbit ledger. Thus every endpoint request must pass through a displayed contraction-budget comparison before it reaches the readback row.

Finally read Q and E. The row Q turns the orbit estimates into a $\mathsf{RegSeqRat}^{\uparrow}$ regular readback, and E seals the endpoint at the $\mathsf{Real}^{\uparrow}$ boundary. Since the carrier has no further endpoint coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 8239.3 (Compact complete fixed-point Cauchy orbit route)**.** *Every accepted* CompactCompleteFixedPoint *consumer that exports a fixed-point endpoint must first expose the route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{BanachContraction}^{\uparrow},\ O,\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route presents the endpoint as a compact-complete contraction orbit with a regular Cauchy readback and a terminal real seal. It exports no selected global fixed-point function, quotient equality of orbits, host completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with carrier admission from Definition 8239.1. The public packet displays exactly the rows $K, M, T, B, O, Q, E, H, C, P, N$.

Apply the NameCert obligations of Definition 8239.2. They force any endpoint read to pass through compactness, completeness, continuous self-map replay, contraction data, and the finite orbit ledger before the regular readback can be consumed.

The readback and endpoint boundary are then fixed by Q and E. Because selected fixed points, quotient orbit equality, host completion equality, countable choice, and ambient real completeness are not coordinates of the carrier, they cannot be repacked by the handoff rows H, C, P, N. □

Falsifiable prediction: Any accepted CompactCompleteFixedPointUp consumer that exports a fixed-point endpoint while bypassing compact metric exposure, complete metric exposure, continuous self-map replay, contraction data, the finite orbit ledger, regular rational readback, or the terminal RealUp seal fails the carrier classifier.

Independence witness: compactmetric, completemetric, continuousmap, banachcontraction, regseqrat, real: none carries compact metric exposure, complete metric exposure, continuous self-map replay, contraction data, a finite Picard-orbit ledger, regular rational readback, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and one local NameCert row in a single fixed-point packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCompleteFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCompleteFixedPoint}^{\uparrow}$ packet is generated from CompactMetricUp source data, CompleteMetricUp boundary data, ContinuousMapUp self-map replay, BanachContractionUp contraction data, a finite Picard-orbit ledger, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCompleteFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8239.1, Definition 8239.2, and Definition 8239.3 seed the compact-complete fixed-point route over CompactMetricUp, CompleteMetricUp, ContinuousMapUp, BanachContractionUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCompleteFixedPoint}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected fixed-point functions, quotient orbit equality, host completion equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-complete source exposure, continuous-map replay, contraction-budget exactness, finite orbit readback, endpoint sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactCompleteFixedPointUp.

Chapter 8240

A Concrete Naming Certificate for $\mathsf{HautusTest}^{\uparrow}$

Following Hautus and standard finite-dimensional control theory, the $\mathsf{HautusTest}^{\uparrow}$ packet records the finite rank-test surface for controllability and observability of a displayed linear plant. It sits after the controllability certificate of chapter 297, the observability certificate of chapter 294, the matrix carrier of chapter 49, the linear-map carrier of chapter 48, and the vector-space carrier of chapter 47. The packet names only finite matrix-pencil rows, eigenvalue probes, rank windows, transport, replay, provenance, and local naming data; it does not import a spectral decomposition theorem, analytic transfer-function equality, a host quotient state space, or an infinite-dimensional control result.

Definition 8240.1 (HautusTest carrier)**.** *A* $\mathsf{HautusTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, U, Y, A, B, C, \Lambda, \Pi, Q, O, M, V, T, R, P, N)$$

with the following displayed rows:

1. *X is the finite state row, U is the input row, and Y is the output row, each read through displayed* $\mathsf{VecSpace}^{\uparrow}$ *coordinates.*
2. *$A : X \to X$, $B : U \to X$, and $C : X \to Y$ are the finite plant, input, and output rows, read through* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *presentations.*
3. *Λ is the finite eigenvalue-probe ledger for the displayed plant row A; it records only the probes used by the packet.*
4. *Π is the matrix-pencil ledger containing the displayed rows for $sI - A$, the controllability pencil $(sI - A, B)$, and the observability pencil $(sI - A, C)$ at every probe s named by Λ.*
5. *Q is the controllability rank-window row consumed by* $\mathsf{ControlControllability}^{\uparrow}$*.*
6. *O is the observability rank-window row consumed by* $\mathsf{ControlObservability}^{\uparrow}$*.*
7. *M records the finite matrix-entry and rank-readback rows used by Π, Q, O.*
8. *V records the vector-space basis and dimension rows against which the rank windows are measured.*
9. *T records componentwise* hsame *transport for $X, U, Y, A, B, C, \Lambda, \Pi, Q, O, M, V$ and the local certificate row.*

10. *R records displayed* Cont *replay from the plant rows through the probe ledger, matrix-pencil rows, controllability window, and observability window.*
11. *P is the* Pkg *provenance over* $\mathsf{HautusTest}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HautusTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host spectral decomposition, quotient state-space semantics, selected canonical bases, transfer-function equality, stochastic covariance, or infinite-dimensional observability and controllability theorems.

Lean4 ✓ `BEDC.Derived.HautusTestUp.HautusTestTasteGate single carrier alignment`

Theorem 8240.2 (HautusTest NameCert obligations)**.** *For every accepted* $\mathsf{HautusTest}^{\uparrow}$ *carrier*

$$H = (X, U, Y, A, B, C, \Lambda, \Pi, Q, O, M, V, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HautusTest}^{\uparrow}}$ *surface consists exactly of finite state, input, and output admission through X, U, Y, plant admission through A, B, C, eigenvalue-probe admission through Λ, matrix-pencil admission through Π, controllability rank-window admission through Q, observability rank-window admission through O, matrix readback through M, vector-space and dimension readback through V, componentwise* hsame *transport through T, displayed* Cont *replay through R, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8240.1. The rows X, U, Y, A, B, C display the finite control-system data before any rank test is visible.

The probe ledger Λ and matrix-pencil ledger Π then expose the finite rows for $sI-A$, $(sI-A, B)$, and $(sI - A, C)$. The row Q is the controllability read of those rows, while O is the observability read of the same plant and probe surface.

Finally M, V, T, R, P, N record matrix readback, vector-space dimension readback, transport, replay, provenance, and local naming. Since the carrier contains no coordinate for host spectral decomposition, quotient state spaces, selected canonical bases, transfer functions, stochastic covariance, or infinite-dimensional semantics, these displayed rows exhaust the local certificate surface. □

Theorem 8240.3 (HautusTest controllability-observability handoff)**.** *Every downstream finite-control consumer of an accepted* $\mathsf{HautusTest}^{\uparrow}$ *packet that asks for a rank-test certificate must read the same displayed route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{VecSpace}^{\uparrow} \longmapsto \Lambda, \Pi, Q, O, M, V, T, R, P, N.$$

The controllability side reaches $\mathsf{ControlControllability}^{\uparrow}$ *only through Q, and the observability side reaches* $\mathsf{ControlObservability}^{\uparrow}$ *only through O. Both rows are tied to the same plant rows A, B, C, the same matrix-pencil ledger Π, and the same vector-space dimension row V.*

Proof. Start with Definition 8240.2. It forces the finite plant rows, probe ledger, matrix-pencil ledger, controllability rank window, and observability rank window to be displayed before a consumer can read the certificate.

The controllability handoff is the row Q over $(sI - A, B)$, and the observability handoff is the row O over $(sI - A, C)$. Both are measured against the same matrix row M and vector-space row V, so neither handoff can replace the shared packet by a separate ambient rank theorem.

The structural rows T, R, P, N then transport, replay, source, and name exactly that finite route. A consumer that bypasses Q or O, or that reads an unlogged spectral decomposition, quotient state space, transfer function, stochastic covariance, or infinite-dimensional theorem, has left the accepted $\mathsf{HautusTest}^{\uparrow}$ packet. □

Falsifiable prediction: An accepted $\mathsf{HautusTest}^{\uparrow}$ packet is refuted by any finite-control consumer that reads a controllability or observability rank result while bypassing the displayed probe ledger Λ, the matrix-pencil ledger Π, or either rank-window row Q, O.

Independence witness: controlcontrollability, controlobservability, matrix, linearmap, vecspace: none is bijective to $\mathsf{HautusTest}^{\uparrow}$ because the Hautus-test packet jointly carries plant rows, eigenvalue probes, two matrix-pencil rank windows, shared matrix readback, vector-space dimension readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HautusTest}^{\uparrow}}$.

Constructive story: A $\mathsf{HautusTest}^{\uparrow}$ packet is generated from finite state, input, output, plant, eigenvalue-probe, matrix-pencil, controllability rank-window, observability rank-window, matrix readback, vector-space dimension, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HautusTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8240.1, Definition 8240.2, Definition 8240.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite Hautus rank-test NameCert surface over controllability, observability, matrix, linear-map, and vector-space rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HautusTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host spectral decomposition, quotient state-space semantics, selected canonical bases, transfer-function equality, stochastic covariance, or infinite-dimensional control theory.

Upgrade path: Obligation closure requires checked carrier admission, probe-ledger admission, matrix-pencil exactness, controllability and observability rank-window admission, transport, replay, provenance, and local naming for BEDC.Derived.HautusTestUp.

Chapter 8241

A Concrete Naming Certificate for $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$

Following Coddington–Levinson and Bishop–Bridges, the $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$ packet records the finite local flow-box surface by which an ordinary differential equation is read near a displayed nonstationary point through rational time-state windows, Picard local handoff rows, and Gronwall stability budgets. It sits after the ODE source of chapter 193, the initial-value carrier of chapter 3979, the Picard–Lindelof handoff of chapter 1888, the Gronwall comparison surface of chapter 2034, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a local coordinate and flow-window certificate; it does not select a host solution curve, assert global existence, quotient paths by equality, or invoke ambient real completeness.

Definition 8241.1 (Constructive ODE flow-box carrier)**.** *A* $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, I, L, P, G, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *source row, carrying the local time-state window and vector-field read.*
2. *I is the* $\mathsf{InitialValueProblem}^{\uparrow}$ *row naming the base point and initial-time boundary.*
3. *L is the local nonstationary and coordinate-window row, storing the rational box, transverse coordinate, and time strip on which the flow-box read is admitted.*
4. *P is the* $\mathsf{PicardLindelof}^{\uparrow}$ *local handoff row for the displayed initial-value data and box.*
5. *G is the* $\mathsf{GronwallInequality}^{\uparrow}$ *stability row used to compare nearby Picard windows inside the same strip.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the local coordinate and flow-window approximants, and E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R is displayed.*
7. *H records componentwise* hsame *transport for the ODE source, initial-value row, local coordinate window, Picard handoff, Gronwall stability row, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$O, I, L, P, G, R, E \longmapsto H, Q, N.$$

9. *Q is the* Pkg *provenance over* $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{InitialValueProblem}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveOdeFlowBox}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, I, L, P, G, R, E, H, C, Q, N$. It has no coordinate for a selected host solution, maximal continuation interval, quotient path equality, trichotomy on real values, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8241.2 (Constructive ODE flow-box obligations)**.** *For every accepted* $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$ *carrier $F = (O, I, L, P, G, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveOdeFlowBox}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed ODE source, initial-value row, local coordinate window, Picard–Lindelof handoff, Gronwall stability row, regular rational readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8241.1. The ODE source and initial-value boundary are the rows O and I, so every local flow-box read is tied to a displayed time-state window and base point.

Read the local coordinate row L. It stores the finite box, transverse coordinate, and time strip on which the nonstationary hypothesis is visible to the packet, rather than as an external differentiable-chart theorem.

The analytic handoff rows are P, G, R, E. The Picard row supplies the local solution-window handoff, the Gronwall row supplies stability inside the strip, the regular rational row records the finite approximants, and the Real seal is downstream of that readback.

Finally H, C, Q, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores a selected host solution, maximal interval, quotient path equality, trichotomy, countable-choice schedule, or ambient completeness witness, those objects cannot enter the local NameCert surface. □

Theorem 8241.3 (Constructive ODE flow-box Picard local handoff)**.** *Every accepted* $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$ *packet hands its local flow-box read to downstream ODE consumers only through*

$$\mathsf{Ode}^{\uparrow}, \quad \mathsf{InitialValueProblem}^{\uparrow}, \quad \mathsf{PicardLindelof}^{\uparrow}, \quad \mathsf{GronwallInequality}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The local coordinate row L must be displayed before the Picard handoff P is consumed, and the Gronwall row G must be displayed before any Real-facing seal E is read. The route exports no selected solution curve, host flow map, quotient path equality, global existence theorem, countable choice, or ambient real completeness.

Proof. Start with the source rows O, I, L. They fix the ODE request, initial-value boundary, and local coordinate box before any solution-window handoff is available.

Apply the Picard–Lindelof row P only after that local box is displayed. The handoff therefore reads finite Picard windows over the same time-state strip, not a host solution curve or global flow.

Next read the Gronwall row G. It compares neighboring Picard windows inside the displayed strip and supplies the stability budget needed before the regular rational readback R can be consumed.

The terminal Real seal E is downstream of R, while H, C, Q, N preserve and name the same route. Coordinate exhaustion excludes selected solution curves, host flow maps, quotient path equality, global existence, countable choice, and ambient completeness from the exported handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveOdeFlowBox}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveOdeFlowBox}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8241.1, Definition 8241.2, and Definition 8241.3 seed the finite local ODE flow-box packet from displayed ODE, initial-value, Picard, Gronwall, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveOdeFlowBox}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected host solution curves, global existence, maximal continuation, quotient path equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local coordinate-window exactness, Picard local handoff, Gronwall stability, RegSeqRat readback, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveOdeFlowBoxUp.

Constructive story: A $\mathsf{ConstructiveOdeFlowBox}^{\uparrow}$ packet is generated from Ode, InitialValueProblem, local coordinate-window, PicardLindelof, GronwallInequality, RegSeqRat, Real, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Chapter 8242

A Concrete Naming Certificate for $\mathsf{CompactUniformTriangleChain}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges compact-uniform reasoning, the $\mathsf{CompactUniformTriangleChain}^{\uparrow}$ packet records the finite source-and-target triangle chain by which compact metric coverage, pointwise continuity, rational tolerance folds, and target metric estimates are read before a $\mathsf{UniformModulus}^{\uparrow}$ consumer receives a tolerance-only handoff. It sits after the compact metric source of chapter 129, the continuous-map graph of chapter 130, the metric law surface of chapter 57, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It shares the final estimate boundary with chapter 7260 and chapter 8118, but this packet names the whole chain from source coverage to target estimate as one finite NameCert carrier.

Definition 8242.1 (Compact-uniform triangle-chain carrier)**.** *A* $\mathsf{CompactUniformTriangleChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, U, M_X, M_Y, D, S, A, R, E, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row, listing the selected compact-net center, source ball, and inspected source point pair.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose pointwise-modulus reads are attached to the center named by K.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after the triangle chain is complete.*
4. *M_X and M_Y are the source and target* $\mathsf{Met}^{\uparrow}$ *law rows, including the triangle witnesses used on each side of F.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance fold row, splitting the requested tolerance among source-radius, pointwise, and target-triangle contributions.*
6. *S is the source-triangle row, showing that both source points remain in the displayed local ball around the selected center.*
7. *A is the target-triangle row, joining the two pointwise target estimates through the center image.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerance and metric-window data, and E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *H records componentwise* hsame *transport for compact-net, continuous-map, uniform-modulus, source metric, target metric, dyadic fold, source triangle, target triangle, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through*

$$K, F, M_X, M_Y, D, S, A, R, E \longmapsto U, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformTriangleChain}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleChain}^{\uparrow}}$ *row.*

The classifier compares accepted triangle-chain packets only through $K, F, U, M_X, M_Y, D, S, A, R, E, H, C, P, N$. It has no coordinate for a host compactness theorem, an external modulus selector, hidden center choice, quotient function equality, selected limits, or ambient real completeness.

Theorem 8242.2 (Compact-uniform triangle-chain obligations)**.** *For every accepted* CompactUniformTriangleChain *carrier*

$$T = (K, F, U, M_X, M_Y, D, S, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleChain}^{\uparrow}}$ *surface consists exactly of compact-source admission, continuous-map pointwise admission, source metric triangle admission, target metric triangle admission, dyadic tolerance-fold exactness, source-triangle compatibility, target-triangle compatibility, regular rational readback, Real seal routing, UniformModulus handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8242.1. The compact-source and continuous-map data are K and F, so the chain begins with a displayed center and a pointwise graph read at that center.

Read the metric and tolerance rows M_X, M_Y, D. The source and target metric rows provide the two triangle laws, while the dyadic fold row records the finite rational split of the requested tolerance.

The triangle rows are S and A. The source triangle keeps both source points inside the selected local ball, and the target triangle composes the two pointwise target estimates through the center image.

Finally R, E, U, H, C, P, N read back, seal, export, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host compactness, external modulus selection, hidden center choice, quotient function equality, selected limits, or ambient completeness, the listed rows exhaust the NameCert surface. □

Theorem 8242.3 (Compact-uniform triangle-chain UniformModulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$*-facing read of an accepted* $\mathsf{CompactUniformTriangleChain}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before the handoff row U is available. In carrier notation, the consumer must read $K, F, M_X, M_Y, D, S, A, R, E$ before U, H, C, P, N can export the tolerance-only modulus route, and it receives no host compactness theorem, hidden center choice, quotient equality, selected limit, or ambient real completeness principle.

Proof. Start with the source packet K and graph packet F. These rows display the compact-net center, the source point pair, and the pointwise continuity read consumed by the triangle chain.

Apply the metric rows M_X, M_Y and the dyadic fold row D. The tolerance split is finite, and both triangle estimates are read as metric-law rows rather than as an ambient continuity theorem.

Use the source and target triangle rows S, A. They turn the compact-source coverage and pointwise center estimate into the target-distance estimate, after which R and E read the regular rational and Real-facing windows.

Only then can the handoff row U be exported. The structural rows H, C, P, N preserve and name this route, so a downstream consumer obtains a finite UniformModulus handoff and none of the refused host-level objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleChain}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformTriangleChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8242.1, Definition 8242.2, and Definition 8242.3 seed the finite source-and-target triangle chain from CompactMetric, ContinuousMap, Metric, DyadicRatCore, RegSeqRat, Real, and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformTriangleChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, external modulus selection, hidden center choice, quotient function equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, pointwise-continuity admission, source and target metric triangle exactness, dyadic tolerance-fold exactness, RegSeqRat readback, Real seal routing, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformTriangleChainUp.

Constructive story: A $\mathsf{CompactUniformTriangleChain}^{\uparrow}$ packet is generated from CompactMetric finite-source rows, ContinuousMap pointwise rows, UniformModulus handoff rows, Metric source and target triangle laws, DyadicRatCore tolerance folds, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8243

A Concrete Naming Certificate for $\mathsf{ClosedSubspaceComplete}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedSubspaceComplete}^{\uparrow}$ packet records the constructive theorem that a closed subspace of a complete metric carrier is complete as finite BEDC NameCert data. It sits after the complete-metric service of chapter 3687, the closed-subset boundary of chapter 4229, the metric carrier of chapter 57, the finite stream interface of chapter 126, and the regular rational readback of chapter 301. The packet names limit retention inside the displayed closed subspace; it does not assert a quotient completion, choose a host representative, invoke countable choice, add ambient compactness, or identify completed reals by host equality.

Definition 8243.1 (Closed subspace complete carrier)**.** *A* $\mathsf{ClosedSubspaceComplete}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, D, W, R, C, L, H, Q, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row whose distance requests are visible to the subspace consumer.*
2. *F is the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary row naming the subspace as closed inside the ambient metric carrier.*
3. *D is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row for the ambient carrier.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the accepted Cauchy name inside the subspace.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the rational tolerance schedule of that Cauchy name.*
6. *C is the Cauchy-admission row recording that the displayed stream stays inside the closed subspace.*
7. *L is the limit-retention row: the ambient complete-metric limit read supplied by D is retained by the closed-set boundary F.*
8. *H records componentwise* hsame *transport for M, F, D, W, R, C, L and the local certificate row.*
9. *Q records displayed* Cont *replay from a subspace Cauchy request through $M, F, D, W, R, C, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedSubspaceComplete}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSubspaceComplete}^{\uparrow}}$ *row.*

The carrier exports only the displayed ambient metric, closed-subspace boundary, ambient completeness service, finite Cauchy readback, limit-retention row, and structural transport, replay, provenance, and naming rows.

Lean4 (statement) `BEDC.Derived.ClosedSubspaceCompleteUp`

Theorem 8243.2 (Closed subspace complete NameCert obligations)**.** *For the carrier of Definition 8243.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedSubspaceComplete}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (M, F, D, W, R, C, L, H, Q, P, N),$$

ambient metric exposure through M, closed-set boundary exposure through F, ambient complete-metric service through D, finite-window Cauchy exposure through W, regular-rational tolerance readback through R, subspace Cauchy admission through C, limit retention through L, componentwise transport through H, displayed replay through Q, package provenance through P, and local naming through N.

Proof. Project the carrier coordinates. The ambient rows M, F, D provide the metric source, closed-subspace boundary, and complete-metric service, while W, R, C provide the finite Cauchy name and its regular rational tolerance readback.

Read the retention row L. It is available only after the ambient limit service D and closed boundary F have been displayed for the same Cauchy name, so the limit remains inside the subspace carrier rather than leaving to a host completion.

Finally read H, Q, P, N. These rows transport, replay, source, and name the same displayed route, and there is no coordinate for quotient completion, selected representatives, host real equality, countable choice, or ambient compactness. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 8243.3 (Closed subspace complete Cauchy-limit retention)**.** *Every completion-facing consumer of an accepted* $\mathsf{ClosedSubspaceComplete}^{\uparrow}$ *carrier must route a subspace Cauchy name through*

$$W, R, C \longmapsto D \longmapsto F \longmapsto L$$

before it can read a retained limit. The retained limit is therefore a displayed closed-subspace endpoint supplied by the ambient $\mathsf{CompleteMetric}^{\uparrow}$ *service and the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary, not a quotient completion class, not a selected host representative, and not an ambient completeness theorem for unrelated subspaces.*

Proof. Start with the finite-window and rational readback rows W and R. They expose the accepted Cauchy observations that the subspace consumer is allowed to inspect.

Use the Cauchy-admission row C to keep those observations inside the displayed closed subspace. The ambient complete-metric service D may then be read for that same Cauchy name, but only along the route already tied to M, F, W, R, C.

Apply the closed-set boundary row F and the retention row L. These rows record that the ambient limit returned by D is retained inside the closed subspace, while H, Q, P, N preserve the same route under transport, replay, provenance, and local naming.

Since the carrier has no coordinate for quotient completion, selected host representatives, countable choice, host real equality, or ambient compactness, no completion-facing read can export those surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubspaceComplete}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedSubspaceComplete}^{\uparrow}$ packet is generated from an ambient Metric row, a ClosedSet boundary row, an ambient CompleteMetric service row, finite StreamName windows, RegSeqRat tolerance readback, subspace Cauchy admission, limit retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubspaceComplete}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8243.1, Definition 8243.2, and Definition 8243.3 seed the finite retention route for closed subspaces of complete metric carriers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedSubspaceComplete}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.ClosedSubspaceCompleteUp`

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected host representatives, countable choice, host real equality, ambient compactness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-boundary retention, ambient complete-metric handoff, finite-window readback, transport, replay, provenance, and local naming for the same packet.

Chapter 8244

A Concrete Naming Certificate for $\mathsf{GrussInequality}^{\uparrow}$

Following Gruss and Bishop–Bridges constructive analysis, the $\mathsf{GrussInequality}^{\uparrow}$ packet records the finite BEDC route by which a bounded pair of integrable real-valued maps on a displayed compact metric source yields a covariance-bound certificate. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream-name window of chapter 126, the continuous-map graph rows of chapter 130, the integral carrier of chapter 123, and the compact metric source rows of chapter 129. The packet names a finite covariance-bound route; it does not assert host equality of functions, measure completion, selected extrema, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8244.1 (Gruss inequality covariance-bound carrier)**.** *A* $\mathsf{GrussInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, F, L, B, I, Q, W, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, naming the compact metric domain together with the finite observation covers consumed by the bound request.*
2. *F is the paired* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the two real-valued maps whose covariance is being bounded.*
3. *L is the value-window row, recording displayed lower and upper Real bounds for each map.*
4. *B is the bounded-oscillation budget row, exposing the four endpoint differences used by the Gruss constant.*
5. *I is the* $\mathsf{Integral}^{\uparrow}$ *row for the four integral reads: the two single-map reads, the product read, and the normalization read of the compact source.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint differences, integral values, product-integral comparison, and tolerance budget.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the endpoint, graph, and integral reads are observed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed covariance-bound value.*

9. *H records componentwise* hsame *transport for the compact source, paired maps, bounds, oscillation budget, integral rows, readback, stream window, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay through $K, F, L, B, I, Q, W, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{GrussInequality}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GrussInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, B, I, Q, W, E, H, C, P, N$. There is no coordinate for host function equality, selected extrema, a measure-completion object, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8244.2 (Gruss inequality NameCert obligations)**.** *For every accepted* $\mathsf{GrussInequality}^{\uparrow}$ *carrier $G = (K, F, L, B, I, Q, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GrussInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, paired continuous-map graphs, displayed lower and upper value bounds, bounded-oscillation budget, integral readback, regular rational comparison, finite stream observation, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 8244.1. The analytic rows are the compact source K, the paired graph row F, the bound row L, the oscillation budget B, and the integral row I.

Readback and observation are downstream. The row Q repacks only the displayed endpoint differences and integral comparisons, while W fixes the finite observation window and E seals the resulting Real-bound value.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no host function-equality, selected-extremum, measure-completion, countable-choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8244.3 (Gruss inequality integral covariance bound)**.** *Every accepted* $\mathsf{GrussInequality}^{\uparrow}$ *packet presents the covariance read through the finite route*

$$K, F, L, B, I \longmapsto Q, W, E, H, C, P, N.$$

The bound compares the displayed integral covariance only with the Real-sealed budget

$$\frac{1}{4}(L_f^+ - L_f^-)(L_g^+ - L_g^-),$$

where the four endpoint values are the rows stored in L and the comparison is read through Q, W, E. A consumer receives no covariance estimate unless the compact metric source, the two continuous-map rows, the bounded windows, and the integral rows have all been exposed.

Proof. Start from K, F, L, B, I. The compact metric source and paired continuous graphs are displayed before any integral read is consumed, and the bound row records the lower and upper windows used by the Gruss constant.

Use the integral row I to expose the two single-map integral reads, the product read, and the normalization read. The row B supplies only the four endpoint differences already present in L.

The regular rational row Q compares these integral reads with the displayed quarter-product budget, the stream row W records the finite observation window, and E seals the comparison as a Real-facing value. Transport, replay, provenance, and naming through H, C, P, N preserve the same route for downstream consumers. □

Theorem 8244.4 (Gruss inequality L10 nonescape)**.** *Every L10 consumer of an accepted* $\mathsf{GrussInequality}^{\uparrow}$ *packet must factor through*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Integral}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto K, F, L, B, I, Q, W, E, H, C, P,$$

Thus the packet exports a finite covariance-bound certificate, while excluding host function equality, selected extrema, detached measure completion, countable choice, quotient equality of real names, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the obligation theorem Definition 8244.2. It forces any consumer to read the compact metric source, paired continuous-map graphs, bound rows, integral rows, regular rational comparisons, stream observation, Real sealing, transport, replay, provenance, and local name.

Apply the covariance-bound theorem Definition 8244.3. The only Real-facing estimate is the displayed quarter-product route; no independent equality or selection row appears in the carrier.

The excluded principles are absent from the carrier coordinates. Therefore an L10 consumer can replay the finite covariance-bound certificate but cannot extract host function equality, selected extrema, a detached measure-completion object, countable choice, quotient equality of real names, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{GrussInequality}^{\uparrow}$ read must display a CompactMetric source, paired ContinuousMap graphs, lower and upper value windows, an oscillation budget, Integral rows, RegSeqRat comparison, StreamName observation, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert before any covariance-bound value is consumed.

Independence witness: real, regseqrat, streamname, continuousmap, integral, compactmetric: $\mathsf{GrussInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-source rows, paired graph rows, bound windows, covariance integral reads, rational comparison, stream observation, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GrussInequality}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GrussInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8244.1, Definition 8244.2, Definition 8244.3, and Definition 8244.4 seed the constructive covariance-bound route over compact metric source rows, paired continuous-map graphs, displayed bounds, integral reads, rational comparison, stream observation, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GrussInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host equality of functions, selected extrema, detached measure completion, countable choice, quotient equality of real names, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{GrussInequality}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, bound-window exactness, integral covariance readback, rational comparison, Real-seal nonescape, transport stability, and local NameCert export for BEDC.Derived.GrussInequalityUp.

Constructive story: A $\mathsf{GrussInequality}^{\uparrow}$ packet is generated from a compact metric source, paired continuous-map graphs, lower and upper value windows, a bounded-oscillation budget, integral rows, regular rational readback, stream observation, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8245

A Concrete Naming Certificate for $\mathsf{ReedyCategory}^{\uparrow}$

Following Reedy and Quillen, the $\mathsf{ReedyCategory}^{\uparrow}$ packet records the degree-directed category surface used by latching and matching constructions as finite NameCert data. It sits after the categorical carrier of chapter 61, the functorial transport surface of chapter 62, the simplicial-set row family of chapter 157, and the model-category lifting and factorisation surface of chapter 158. Those sibling chapters expose categories, functors, simplicial degrees, and model-category classes separately; this carrier binds degree, direct maps, inverse maps, unique direct-inverse factorisation, latching boundary, matching boundary, transport, replay, provenance, and local naming into one finite packet.

Definition 8245.1 (Reedy category carrier)**.** *A* $\mathsf{ReedyCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, D, A^{+}, A^{-}, F, L, M, H, T, P, N)$$

with the following displayed rows:

1. *C is the consumed* $\mathsf{Cat}^{\uparrow}$ *row, recording objects, morphisms, identities, composition, and the categorical classifier boundary.*
2. *D is the degree row, assigning a displayed natural degree to every object visible in the packet.*
3. *A^{+} is the direct-map row, recording morphisms that strictly raise degree unless they are identities.*
4. *A^{-} is the inverse-map row, recording morphisms that strictly lower degree unless they are identities.*
5. *F is the factorisation row, recording the unique decomposition of every displayed morphism as an inverse map followed by a direct map.*
6. *L is the latching-boundary row, recording the finite direct-source boundary used by diagram consumers.*
7. *M is the matching-boundary row, recording the finite inverse-target boundary used by diagram consumers.*
8. *H stores componentwise* hsame *transport for the category, degree, direct, inverse, factorisation, latching, matching, replay, provenance, and local naming rows.*

9. *T stores displayed* Cont *replay from a Reedy request through degree comparison, direct-inverse factorisation, latching boundary, matching boundary, and sibling category-theory consumers.*
10. *P is the* Pkg *provenance over* $\mathsf{ReedyCategory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{ModelCat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ReedyCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, D, A^+, A^-, F, L, M, H, T, P, N$. It has no coordinate for a host indexing category, a hidden well-foundedness proof, a selected factorisation outside F, an ambient latching colimit, an ambient matching limit, or a model-category theorem.

Theorem 8245.2 (Reedy category NameCert obligations)**.** *For every accepted* $\mathsf{ReedyCategory}^{\uparrow}$ *carrier $R = (C, D, A^+, A^-, F, L, M, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ReedyCategory}^{\uparrow}}$ *surface consists exactly of carrier admission, degree exposure, direct-map exposure, inverse-map exposure, direct-inverse factorisation, latching-boundary exposure, matching-boundary exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8245.1. The category row C supplies the displayed object and morphism boundary, while the degree row D gives the measure used to separate direct and inverse maps.

Rows A^+ and A^- expose the two Reedy morphism classes. The factorisation row F binds them by recording the unique inverse-then-direct decomposition of every displayed morphism; this is the row that prevents a consumer from choosing a hidden decomposition.

The latching row L and matching row M are then read as finite boundary ledgers over the same degree and morphism rows. They prepare the sibling routes to $\mathsf{Functor}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, and $\mathsf{ModelCat}^{\uparrow}$ without importing a host latching colimit or matching limit.

Finally H, T, P, N transport, replay, source, and name the same rows. Since the carrier has no further coordinate, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{ReedyCategory}^{\uparrow}}$ obligations. □

Theorem 8245.3 (Reedy category degree-factorization ledger)**.** *Every displayed morphism read from an accepted* $\mathsf{ReedyCategory}^{\uparrow}$ *carrier factors through the degree ledger*

$$C, D, A^-, A^+, F$$

before it can be consumed by a latching row, matching row, functor row, simplicial-set row, or model-category row. The factorisation is recorded in F and is transported only through H, T, P, N.

Proof. Start with a morphism in the category row C. The degree row D determines whether a non-identity component may be read as direct or inverse, and the rows A^+ and A^- record the two permitted directions.

Apply the factorisation row F. It supplies the unique inverse-then-direct decomposition attached to the displayed morphism, so the latching boundary L and matching boundary M consume the same finite ledger rather than an ambient choice of decomposition.

The sibling consumers do not add a second factorisation route. $\mathsf{Functor}^{\uparrow}$ reads the displayed functorial transport, $\mathsf{SimplicialSet}^{\uparrow}$ reads degree-indexed simplex rows, and $\mathsf{ModelCat}^{\uparrow}$ reads lifting and factorisation classes, all through the same packet route H, T, P, N.

Thus every downstream read of a Reedy morphism factors through the displayed degree-factorisation ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReedyCategory}^{\uparrow}}$**.**

Constructive story: A ReedyCategory$^{\uparrow}$ packet is generated from a category row, degree row, direct-map row, inverse-map row, factorisation row, latching-boundary row, matching-boundary row, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{ReedyCategory}\uparrow}$, obligation)

Scope closed: Definition 8245.1, Definition 8245.2, and Definition 8245.3 close the finite carrier and obligation surface for degree-directed Reedy category reads.

Formal verification: FormalStatus(NameCert$_{\text{ReedyCategory}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, ambient latching colimits, ambient matching limits, full Reedy model structures, or host category equivalence.

Upgrade path: Lean closure requires checked carrier admission, degree exposure, direct and inverse map exposure, factorisation uniqueness, latching and matching boundary exposure, transport, replay, provenance, and local naming for BEDC.Derived.ReedyCategoryUp.

Chapter 8246

A Concrete Naming Certificate for $\mathsf{ReedyModelStructure}^{\uparrow}$

Following Reedy and Quillen, the $\mathsf{ReedyModelStructure}^{\uparrow}$ packet records the finite diagram-index model-category surface used to put model structures on Reedy diagrams. It sits after the degree-factorisation carrier of chapter 8245, the lifting and factorisation rows of chapter 158, the simplicial indexing rows of chapter 157, the displayed replacement witness of chapter 308, and the localization packet of chapter 325. Those siblings expose Reedy categories, ambient model classes, simplicial rows, finite cofibrant witnesses, and selected localizing rows separately; this carrier binds the diagram-index model structure as one NameCert surface.

Definition 8246.1 (Reedy model structure carrier)**.** *A* $\mathsf{ReedyModelStructure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, D, A, L, K, C, W, F, T, H, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{ReedyCategory}^{\uparrow}$ *row, recording the finite degree data, direct maps, inverse maps, and unique direct-inverse factorisation ledger.*
2. *D is the diagram row, recording the displayed finite family of objects and arrows indexed by the Reedy category.*
3. *A is the direct/inverse factorisation row, repacking the direct and inverse sides of R as the dependency route for diagram maps.*
4. *L is the latching-object row, recording the finite direct-source boundary visible to cofibration reads.*
5. *K is the matching-object row, recording the finite inverse-target boundary visible to fibration reads.*
6. *C is the model-class row, recording Reedy cofibration, Reedy fibration, and objectwise weak-equivalence reads through* $\mathsf{ModelCat}^{\uparrow}$*.*
7. *W is the weak-equivalence and replacement row, exposing the objectwise weak-equivalence ledger together with any displayed* $\mathsf{CofibrantReplacement}^{\uparrow}$ *witness.*
8. *F is the selected localization row, present only when a downstream consumer reads a displayed* $\mathsf{BousfieldLocalization}^{\uparrow}$ *packet.*

9. *T stores componentwise* hsame *transport for Reedy data, diagram rows, latching rows, matching rows, model-class rows, replacement rows, localization rows, replay, provenance, and local naming.*
10. *H stores displayed* Cont *replay from a diagram-model request through Reedy degree comparison, latching and matching inspection, model-class reads, replacement reads, and optional localization reads.*
11. *P is the* Pkg *provenance over* $\mathsf{ReedyModelStructure}^{\uparrow}$, $\mathsf{ReedyCategory}^{\uparrow}$, $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{CofibrantReplacement}^{\uparrow}$, $\mathsf{BousfieldLocalization}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ReedyModelStructure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, A, L, K, C, W, F, T, H, P, N$. It has no coordinate for an ambient functor category, a hidden latching colimit, a hidden matching limit, an unlisted factorisation, a global cofibrant replacement choice, a localization existence theorem, or a homotopy-limit theorem.

Theorem 8246.2 (Reedy model structure NameCert obligations)**.** *For every accepted* $\mathsf{ReedyModelStructure}^{\uparrow}$ *carrier*

$$M = (R, D, A, L, K, C, W, F, T, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ReedyModelStructure}^{\uparrow}}$ *surface consists exactly of carrier admission, Reedy degree exposure, direct and inverse factorisation exposure, diagram-row exposure, latching-object exposure, matching-object exposure, model-class exposure, replacement-row exposure, optional localization-row exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8246.1. The Reedy row R supplies the degree, direct-map, inverse-map, and factorisation data, while the diagram row D supplies the finite indexed objects and arrows consumed by the model-structure read.

The latching and matching rows are then read from the same Reedy ledger. The latching row L exposes direct-source boundary data for cofibration reads, and the matching row K exposes inverse-target boundary data for fibration reads. Neither row imports an ambient limit or colimit outside the carrier.

The model-class rows C, W, F are downstream. They read cofibration, fibration, objectwise weak-equivalence, displayed replacement, and displayed localization data through $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{CofibrantReplacement}^{\uparrow}$, and $\mathsf{BousfieldLocalization}^{\uparrow}$ without adding global replacement choice or localization existence.

Finally T, H, P, N transport, replay, source, and name the same rows. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8246.3 (Reedy model structure finite diagram route)**.** *Every* $\mathsf{ReedyModelStructure}^{\uparrow}$ *consumer that reads a diagram-model cofibration, fibration, weak equivalence, replacement, or localization row must first expose the finite route*

$$R, D, A \longmapsto L, K \longmapsto C, W, F \longmapsto T, H, P, N.$$

In particular, Reedy degree data and direct/inverse factorisation precede latching and matching inspection, latching and matching inspection precede model-class reads, and model-class reads precede replacement or localization consumers.

Proof. Start with a consumer read of the accepted carrier M. The only index-facing rows are R, D, A, so the read first exposes the Reedy category data, the displayed diagram, and the direct/inverse factorisation route.

Next the latching and matching rows become available. The cofibration side reads the finite latching boundary L, while the fibration side reads the finite matching boundary K. Both rows are attached to the same degree and factorisation ledger.

The model-class stage reads C, W, F. The $\mathsf{ModelCat}^{\uparrow}$ row supplies cofibration, fibration, and weak-equivalence classes, the $\mathsf{CofibrantReplacement}^{\uparrow}$ dependency supplies only displayed replacement witnesses, and the $\mathsf{BousfieldLocalization}^{\uparrow}$ dependency supplies only displayed selected-map and local-object rows when present.

Transport and replay through T, H, P, N preserve this ordering. Thus no accepted consumer can replace the finite diagram route by an ambient functor category, hidden latching or matching construction, global replacement choice, localization existence theorem, homotopy-limit theorem, or homotopy-category quotient. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReedyModelStructure}^{\uparrow}}$.

Constructive story: A $\mathsf{ReedyModelStructure}^{\uparrow}$ packet is generated from Reedy category rows, finite diagram rows, direct and inverse factorisation rows, latching and matching rows, model-class rows, replacement rows, optional localization rows, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReedyModelStructure}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 8246.1, Definition 8246.2, and Definition 8246.3 close the finite carrier and obligation surface for Reedy diagram model-structure reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReedyModelStructure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient functor categories, hidden latching colimits, hidden matching limits, global cofibrant replacement choice, localization existence, homotopy limits, or homotopy-category quotients.

Upgrade path: Lean closure requires checked carrier admission, Reedy degree exposure, direct and inverse factorisation exposure, latching and matching row exposure, model-class exposure, replacement and localization row exposure, transport, replay, provenance, and local naming for BEDC.Derived.ReedyModelStructureUp.

Chapter 8247

A Concrete Naming Certificate for $\mathsf{NestedIntervalSqueeze}^{\uparrow}$

Following Bishop and Bridges on nested intervals and constructive real approximation, the $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ packet records a finite squeeze corridor from nested dyadic interval rows to a regular rational readout before any $\mathsf{Real}^{\uparrow}$ consumer sees the terminal seal. It sits beside the dyadic rational core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name boundary of chapter 38, and the dyadic interval net of chapter 5966. The packet names only finite rows: nested dyadic interval refinements, windowed endpoint reads, regular rational readouts, a Real-facing seal, transport, replay, provenance, and local naming.

Definition 8247.1 (NestedIntervalSqueeze carrier)**.** *A* $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested dyadic interval row, listing the retained interval cells, endpoint rows, containment witnesses, and diameter-refinement checks through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{DyadicIntervalNet}^{\uparrow}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, containment, and dyadic-radius observations are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readout row obtained from the midpoint or endpoint witnesses of the same finite windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, S, R have been displayed.*
5. *H records componentwise* hsame *transport for interval rows, window rows, readout rows, Real sealing, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a Real or Completion-facing squeeze request through I, S, R, E, H.*
7. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalSqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalSqueeze}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through I, S, R, E, H, C, P, N. It has no coordinate for Real completeness, Heine-Borel compactness, open-cover compactness, countable choice, trichotomy, host equality, quotient reals, or a selected intersection point.

Theorem 8247.2 (NestedIntervalSqueeze NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ *carrier* $Z = (I, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalSqueeze}^{\uparrow}}$ *surface is exhausted by carrier admission, nested dyadic refinement admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *readout admission,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is read through the finite route*

$$I, S, R \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Project the carrier of Definition 8247.1. The nested interval source is I, and it already contains the dyadic endpoint, containment, and diameter-refinement rows.

The observation route is finite. The row S displays the endpoint and radius windows that are inspected, and the row R reads the same windows as a regular rational sequence surface.

The Real-facing seal E is downstream of I, S, R. Hence a Real or Completion consumer receives only the displayed squeeze corridor before the terminal seal is read.

Finally H, C, P, N transport, replay, source, and locally name the same finite packet. Since the carrier has no coordinate for completeness, compactness, choice, trichotomy, host equality, quotient reals, or a selected intersection point, the listed rows exhaust the local NameCert obligations. □

Theorem 8247.3 (NestedIntervalSqueeze dyadic-refinement stability)**.** *Let* $Z = (I, S, R, E, H, C, P, N)$ *and* $Z' = (I', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ *carriers whose nested dyadic interval rows agree by componentwise* hsame *on endpoint rows, containment witnesses, and diameter-refinement checks. If their finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspect the same dyadic refinement rows, then the induced* $\mathsf{RegSeqRat}^{\uparrow}$ *readout rows expose the same public squeeze route through* I, S, R, E, H, C, P, N *and* $I', S', R', E', H', C', P', N'$*.*

Proof. Induct over the finite list of nested dyadic interval rows in I and I'. The initial row is transported by the displayed endpoint and containment hsame witnesses.

For the refinement step, the dyadic diameter and containment checks are rows of the same $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{DyadicIntervalNet}^{\uparrow}$ corridor. The shared finite windows S and S' inspect exactly those rows, so the next readout entry is transported to the corresponding entry of R'.

After the finite induction, the regular rational readout rows agree on the displayed squeeze route. The structural rows H, C, P, N and H', C', P', N' only transport, replay, package, and name that route, so no host equality, quotient real, choice, compactness, or completeness endpoint is introduced. □

Theorem 8247.4 (NestedIntervalSqueeze RegSeqRat readout)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ *carrier factors through the nested dyadic interval row* I*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, and the readout row* R *before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The readout is therefore a finite regular-rational route through dyadic interval refinement data, not an appeal to interval intersection, quotient reals, host equality, Heine-Borel compactness, open-cover compactness, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier $Z = (I, S, R, E, H, C, P, N)$. The only interval source is I, and the only observation route is the finite window row S.

The $\mathsf{RegSeqRat}^{\uparrow}$ readout row R is built from the endpoint and radius witnesses inspected by S. Thus any regular-rational consumer must read I and S before it can inspect R.

The terminal seal E is available only after that finite readout has been displayed. The remaining rows H, C, P, N preserve the same route by transport, replay, provenance, and local naming.

Since none of the carrier coordinates stores a completed intersection point, quotient real, host equality, compactness theorem, choice schedule, trichotomy witness, or ambient completeness principle, the RegSeqRat readout cannot export any such endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalSqueeze}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalSqueeze}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8247.1, Definition 8247.2, Definition 8247.3, and Definition 8247.4 seed the finite route from nested dyadic interval rows through StreamName windows, RegSeqRat readout, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalSqueeze}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, Heine-Borel compactness, open-cover compactness, choice, trichotomy, host equality, quotient reals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic refinement stability, StreamName window coverage, RegSeqRat readout, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalSqueezeU

Constructive story: A $\mathsf{NestedIntervalSqueeze}^{\uparrow}$ packet is generated from nested dyadic interval rows, StreamName windows, RegSeqRat readout rows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8248

A Concrete Naming Certificate for $\mathsf{CoreCompactSpace}^{\uparrow}$

Following Kelley and Johnstone on compact neighbourhoods and core-compact topology, the $\mathsf{CoreCompactSpace}^{\uparrow}$ packet records a traditional topological-space carrier whose open-neighbourhood requests are mediated by compact containment and way-below data. It sits beside the indexed topology surface of chapter 88, the compact carrier of chapter 60, the neighbourhood-filter route of chapter 4133, the Scott-open handoff of chapter 5950, and the sober-space point reconstruction boundary of chapter 6495. The packet gives compact-neighbourhood, Scott-topology, and sober-space consumers a finite NameCert surface for the way-below neighbourhood step, without asserting host powersets of opens, selected compact neighbourhoods, quotient open equality, choice-indexed bases, or ambient local compactness.

Definition 8248.1 (CoreCompactSpace carrier)**.** *A* $\mathsf{CoreCompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, C, W, Nb, H, R, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the point classifier, indexed open rows, finite-meet ledger, arbitrary-union ledger, and visible open-membership stability.*
2. *O is the displayed open-neighbourhood request. It names the open being inspected and the point or finite neighbourhood observation through which a consumer asks for compact containment.*
3. *C is the compact-containment witness row, recording that the requested neighbourhood admits a compact support row from the* $\mathsf{Compact}^{\uparrow}$ *surface before any consumer may read compactness.*
4. *W is the way-below and compact-neighbourhood handoff row. It records the finite route from C to the way-below neighbourhood relation consumed by Scott-topology and sober-space reads.*
5. *Nb is the* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *row, retaining the local filter discipline around the same displayed request rather than a host collection of all neighbourhoods.*
6. *H records componentwise* hsame *transport for topology source, open-neighbourhood request, compact containment, way-below handoff, neighbourhood filter, replay, provenance, and local naming.*
7. *R records displayed* Cont *replay from an open-neighbourhood request through T, O, C, W, Nb, H.*

8. *P is the* Pkg *provenance over* $\mathsf{CoreCompactSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{NeighborhoodFilter}^{\uparrow}$, $\mathsf{ScottTopology}^{\uparrow}$, $\mathsf{SoberSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{CoreCompactSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, O, C, W, Nb, H, R, P, L$. It has no coordinate for host powersets of opens, selected compact neighbourhoods, quotient-open equality, a choice-indexed basis, ambient local compactness, or global spatiality.

Theorem 8248.2 (CoreCompactSpace NameCert obligations)**.** *For every accepted* $\mathsf{CoreCompactSpace}^{\uparrow}$ *carrier $K = (T, O, C, W, Nb, H, R, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CoreCompactSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission, displayed open-neighbourhood request admission, compact-containment witness exposure, way-below compact-neighbourhood handoff, neighbourhood-filter retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-neighbourhood consumer must read the route*

$$T, O, C, W, Nb, H, R, P, L$$

before treating the request as a public core-compact-space handoff.

Proof. Carrier admission is projection from Definition 8248.1. The packet exposes one topology row, one displayed open-neighbourhood request, one compact-containment witness, one way-below handoff, and one neighbourhood-filter row.

The compact-containment step is held by C. A consumer may not replace it by a selected compact neighbourhood or an open-cover compactness theorem, because the carrier stores only the displayed compact support row.

The way-below handoff is held by W and remains attached to the same request O. The neighbourhood-filter row Nb records local filter discipline for that request, not a host set of all neighbourhoods.

Finally H, R, P, L transport, replay, source, and name the same finite route. Since the excluded host powersets, selected compact neighbourhoods, quotient-open equalities, choice-indexed bases, ambient local compactness, and global spatiality data are absent from the carrier, no local NameCert obligation can export them. □

Theorem 8248.3 (CoreCompactSpace way-below neighbourhood handoff)**.** *Let $K = (T, O, C, W, Nb, H, R, P, L)$ be an accepted* $\mathsf{CoreCompactSpace}^{\uparrow}$ *carrier. Any Scott-topology, sober-space, or compact-neighbourhood read that reaches the way-below row must first expose the* $\mathsf{Topology}^{\uparrow}$ *source row T, the displayed request O, and the compact-containment row C. Only then may it read W as the compact-neighbourhood handoff and Nb as the matching* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *route. The handoff exports no selected basis, no host compact-neighbourhood object, no quotient-open equality, no choice-indexed local compactness, and no global spatiality claim.*

Proof. Start with an accepted carrier and project T, O, C. These rows identify the topology source, the concrete open-neighbourhood request, and the compact support attached to that request.

Apply Definition 8248.2. It requires the way-below handoff W and the neighbourhood-filter row Nb to be read in the same route as T, O, C, then preserves the route through H, R, P, L.

A $\mathsf{ScottTopology}^{\uparrow}$ consumer may use W as the compact-neighbourhood way-below input, and a $\mathsf{SoberSpace}^{\uparrow}$ consumer may use the same visible neighbourhood route when point-reconstruction prose needs compact local support. In both cases the read remains the finite route named by the packet, so it cannot detach into selected bases, host compact neighbourhoods, quotient opens, choice-indexed local compactness, or global spatiality. □

Closure status for $\mathsf{NameCert}_{\mathsf{CoreCompactSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CoreCompactSpace}^{\uparrow}$ packet is generated from a TopologyUp source row, displayed open-neighbourhood request, compact-containment witness row, way-below compact-neighbourhood handoff row, NeighborhoodFilterUp row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoreCompactSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8248.1, Definition 8248.2, and Definition 8248.3 seed the finite core-compact-space interface over topology, compact containment, way-below neighbourhoods, neighbourhood filters, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoreCompactSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host powersets of opens, selected compact neighbourhoods, quotient-open equality, choice-indexed bases, ambient local compactness, global spatiality, or bridge checking.

Upgrade path: Obligation closure requires checked topology-source admission, displayed open-neighbourhood request admission, compact-containment exposure, way-below handoff, neighbourhood-filter retention, transport, replay, provenance, and local naming for BEDC.Derived.CoreCompactSpaceUp.

Chapter 8249

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealComparator}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ packet records a constructive located comparison witness for displayed $\mathsf{Real}^{\uparrow}$ names as finite NameCert data. It sits beside the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located-real row of **??**, and the order-and-apartness surface of chapter 5562. The packet names the finite route by which two real names expose enough located window data for a comparison-facing consumer; it is not host trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8249.1 (Bishop located real comparator carrier)**.** *A* $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, W_X, W_Y, D_X, D_Y, R_X, R_Y, L, A, J, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two source* $\mathsf{Real}^{\uparrow}$ *name rows.*
2. *W_X and W_Y are finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected for the two names.*
3. *D_X and D_Y are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-and-tolerance rows read from those windows.*
4. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows that expose the regular rational names behind the two* $\mathsf{Real}^{\uparrow}$ *seals.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locatedness row, displaying the positive-radius or separation window used by the comparison request.*
6. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *or real-apartness-field row recording the apartness-facing handoff when the displayed windows separate.*
7. *J is the finite comparator witness row. Its alternatives are window-left, window-right, and overlap-pending, each tied to the displayed dyadic and regular rows rather than to a host order on completed reals.*
8. *H records componentwise* hsame *transport for the two real rows, windows, dyadic ledgers, readbacks, locatedness row, apartness handoff, comparator witness, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from a comparison request through* $X, Y, W_X, W_Y, D_X, D_Y, R_X, R_Y, L, A, J$ *and onward to provenance and naming.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealComparator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host trichotomy, quotient real equality, selected representatives, countable choice, or an ambient completeness theorem.

Theorem 8249.2 (Bishop located real comparator NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ *carrier* $K = (X, Y, W_X, W_Y, D_X, D_Y, R_X, R_Y, L, A, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealComparator}^{\uparrow}}$ *surface consists exactly of two* $\mathsf{Real}^{\uparrow}$ *source admissions, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, a* $\mathsf{LocatedReal}^{\uparrow}$ *locatedness row, an* $\mathsf{ApartnessReal}^{\uparrow}$ *handoff row, the finite comparator witness* J*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The analytic rows are precisely the two source real names, their finite observation windows, their dyadic tolerance ledgers, their regular rational readbacks, the locatedness row, the apartness handoff, and the comparator witness.

Read the comparator witness only after those rows are displayed. In a window-left branch, J records that the inspected X window is separated below the inspected Y window by the dyadic and regular rows. In a window-right branch, the same finite evidence is read with the two sources exchanged. In an overlap-pending branch, J records that the displayed data do not license a strict comparison at this tolerance.

The structural rows H, C, P, N transport, replay, source, and name exactly that finite surface. Since no carrier coordinate names host trichotomy, quotient equality, selected representatives, countable choice, or completeness, none of those objects can be exported by the local NameCert row. □

Theorem 8249.3 (Bishop located real comparator stability)**.** *Let* $K = (X, Y, W_X, W_Y, D_X, D_Y, R_X, R_Y, L, A, J, H$ *be an accepted* $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ *carrier. Any componentwise* hsame *transport recorded in* H *preserves the same comparator alternative in* J *together with the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, Pkg, *and* NameCert *rows.*

Proof. Inspect the transport row H. It is componentwise over the displayed source, window, dyadic, readback, locatedness, apartness, comparator, replay, provenance, and naming rows.

Case-split on J. In the window-left branch, transport preserves the two windows and their dyadic separation witness before the left comparison is read. In the window-right branch, the same argument preserves the exchanged separation witness. In the overlap-pending branch, transport preserves the absence of a licensed strict comparison at the displayed tolerance.

The continuation row C replays the transported rows in the same order, and P, N only record provenance and local naming. Thus stability is local to the carrier and does not introduce a host order, quotient equality, or completeness principle. □

Theorem 8249.4 (Bishop located real comparator non-escape)**.** *Every* $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ *consumer read factors through the finite route*

$$X, Y, W_X, W_Y, D_X, D_Y, R_X, R_Y, L, A, J, H, C, P, N.$$

The packet exports no host trichotomy for $\mathsf{Real}^{\uparrow}$*, no quotient real equality test, no selected representative, no countable-choice schedule, and no ambient completeness witness.*

Proof. Start from the carrier inventory. Its only comparison row is J, and J is downstream of the displayed real names, finite stream windows, dyadic tolerance rows, regular rational readbacks, locatedness row, and apartness handoff.

Any consumer route must then pass through H and C to transport and replay the same displayed rows. The provenance and local naming rows record where the finite packet came from, but they do not add comparison data.

Therefore the listed route is exhaustive. The refused host objects are absent from the carrier and cannot be projected from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealComparator}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealComparator}^{\uparrow}$ packet is generated from two RealUp source rows, finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readbacks, a LocatedReal row, an ApartnessReal handoff, a finite comparator witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealComparator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8249.1, Definition 8249.2, Definition 8249.3, and Definition 8249.4 seed the finite located comparison witness for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealComparator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient real equality, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopLocatedRealComparator}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-window comparison exactness, locatedness and apartness handoff, transport stability, provenance, local naming, and non-escape for BEDC.Derived.BishopLocatedRealComparatorUp.

Chapter 8250

A Concrete Naming Certificate for $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$

The $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ packet realizes the classifier-membership reading of chapter 9367 as finite context, rule, membership, refusal, replay, and NameCert rows. It sits beside the MetaCIC typed audit surfaces chapter 775, chapter 2330, chapter 3616, chapter 8235, and chapter 4238. Those siblings expose typed examples, reduction traces, decidable typing frontiers, equality eliminator checks, and diamond-frontier checks; this packet binds a type-checking run to its classifier-membership and refusal trace without exporting a host type theory.

Definition 8250.1 (Typechecking classifier trace carrier)**.** *A* $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (\Gamma, t, \tau, R, E, J, S, U, H, C, P, N)$$

with the following displayed rows:

1. Γ *is the context row, recording the displayed typing context and its closed variable boundary.*
2. t *is the term row read as a finite* BHist *object rather than as host syntax.*
3. τ *is the candidate type row.*
4. R *is the rule-ledger row, recording the finite typing rules inspected by the checker.*
5. E *is the* Ext*-membership row, recording accepted classifier membership when the typing run succeeds.*
6. J *is the judgment-trace row, recording the derivation route from* Γ, t, τ, R *to the membership decision.*
7. S *is the subject-reduction handoff row, recording the typed* Cont *route exposed to reduction consumers.*
8. U *is the refusal row, recording the first visible classifier-membership failure when the term is not accepted.*
9. H *stores componentwise* hsame *transport for context, term, type, rule, membership, judgment, subject-reduction, refusal, replay, provenance, and local naming rows.*
10. C *stores displayed* Cont *replay from a type-checking request through rule inspection, classifier membership, refusal, and reduction handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$*, MetaCIC typed example rows,* Ext*,* Cont*,* SigRel*,* BHist*,* hsame*,* Pkg*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}$ *row.*

The classifier compares accepted traces only through $\Gamma, t, \tau, R, E, J, S, U, H, C, P, N$. It has no coordinate for erased host conversion, unrestricted subject reduction, global confluence, proof irrelevance, proof-term erasure as equality, or an oracle deciding all type systems.

Theorem 8250.2 (Typechecking classifier trace NameCert obligations)**.** *For every accepted* TypecheckingClassifierT *carrier $Q = (\Gamma, t, \tau, R, E, J, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}$ *surface consists exactly of context exposure, term exposure, candidate-type exposure, rule-ledger exposure,* Ext*-membership exposure, judgment-trace exposure, subject-reduction handoff, refusal exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8250.1. The rows Γ, t, τ, R display the finite data inspected by the checker before any classifier decision is made.

Rows E and J record the positive membership route: E stores the Ext-membership read and J stores the derivation trace that connects the context, term, type, and rule ledger to that read. The subject-reduction handoff S is a separate continuation-facing row, not a theorem about all reductions.

The refusal row U records the negative route when membership fails. It keeps the failed rule, context boundary, term row, and type row visible instead of allowing the checker to disappear into host computation.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no further coordinate, these rows exhaust the local $\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}$ obligations. □

Theorem 8250.3 (Typechecking classifier trace membership refusal)**.** *Every* $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ *read chooses exactly one visible classifier route: either it exposes an* Ext*-membership row through E, J, S, H, C, P, N, or it exposes a refusal row through U, H, C, P, N. In neither case may the trace replace classifier membership by erased host conversion, proof irrelevance, global confluence, or an unrestricted subject-reduction theorem.*

Proof. Read the rule ledger R against the context, term, and candidate type rows Γ, t, τ. The carrier has one positive membership coordinate E and one refusal coordinate U.

In the accepted case, the judgment trace J binds the inspected rules to the Ext-membership row E, and the subject-reduction handoff S exports only the typed continuation route displayed by the packet. The structural rows H, C, P, N keep this positive route named and replayable.

In the refused case, the row U records the visible classifier-membership failure. The same structural rows carry the failed route without turning it into a host conversion judgment.

Thus a consumer receives either a displayed membership trace or a displayed refusal trace. The forbidden readings named in the theorem have no carrier coordinate and cannot be imported through the local NameCert. □

Falsifiable prediction: If a $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ packet accepts a term while omitting either the Ext-membership row or the visible rule ledger, the packet is refuted.

Independence witness: metacictypedexamplesgallery, typed-example-reduction-trace, metacic-decidable-typing-frontier, metacic-typed-equality-eliminator, metacic-parallel-diamond-frontier: the siblings expose typed examples, reduction rows, decidable checker frontiers, equality-eliminator

rows, and diamond-frontier rows, but none alone binds context, term, type, rule ledger, membership, refusal, subject-reduction handoff, replay, provenance, and local NameCert rows into the same classifier trace.

The Lean TasteGate route for $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ must provide explicit Nontrivial TypecheckingClassifier and FieldFaithful $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ instances in `lean4/BEDC/Derived/TypecheckingClassifierTraceU` The nontriviality witness must expose at least one accepted or refused classifier trace, and the field-faithfulness witness must preserve $\Gamma, t, \tau, R, E, J, S, U, H, C, P, N$ as distinct rows before any checked marker may cite `BEDC.Derived.TypecheckingClassifierTraceUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{TypecheckingClassifierTrace}^{\uparrow}$ packet is generated from context, term, candidate type, rule ledger, Ext-membership, judgment trace, subject-reduction handoff, refusal, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 8250.1, Definition 8250.2, and Definition 8250.3 close the finite classifier-trace carrier and its membership-or-refusal obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypecheckingClassifierTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, global subject reduction, global confluence, proof irrelevance, erased host conversion, or a general oracle for all type systems.

Upgrade path: Lean closure requires checked carrier admission, rule-ledger exposure, Ext-membership exposure, refusal exposure, subject-reduction handoff, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.TypecheckingClassifierTraceUp.

Chapter 8251

A Concrete Naming Certificate for $\mathsf{CombSpace}^{\uparrow}$

Following Steen and Seebach and Engelking, the $\mathsf{CombSpace}^{\uparrow}$ packet records the topologist's comb as finite BEDC topology and metric boundary data. It sits after the indexed-open surface of chapter 88, the distance-law carrier of chapter 57, the terminal real-name seal of chapter 38, and the finite observation windows of chapter 126. The packet names real-axis base windows, vertical tooth rows, an accumulation-spine row, topology classifier rows, and metric handoff rows without importing a host subspace, quotient point equality, selected infinite teeth, or ambient compactness.

Definition 8251.1 (CombSpace carrier)**.** *A* $\mathsf{CombSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, V, S, T, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the real-axis base-window row. It lists the inspected rational intervals on the base segment and records the* $\mathsf{Real}^{\uparrow}$ *endpoint seals used by those windows.*
2. *V is the vertical-tooth row. It stores finitely many tooth indices, their rational height windows, and the displayed attachment point on the base axis.*
3. *S is the accumulation-spine row. It records the finite evidence that teeth with displayed indices approach the spine boundary through the visible* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *T is the topology classifier row. It names the basic-neighbourhood tests for base-axis points, tooth points, and spine-near observations through the* $\mathsf{Topology}^{\uparrow}$ *carrier.*
5. *M is the metric handoff row. It records only the finite* $\mathsf{Met}^{\uparrow}$ *distance windows that compare displayed base, tooth, and spine observations after their* $\mathsf{Real}^{\uparrow}$ *readbacks are present.*
6. *H records componentwise* hsame *transport for the base-window, tooth, spine, topology, metric, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through*

$$A, V, S, T, M \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{CombSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CombSpace}^\uparrow}$ *row.*
The classifier compares accepted packets only through $A, V, S, T, M, H, C, P, N$. It has no coordinate for a host subset of the plane, quotient point equality, an infinite selected family of teeth, open-cover compactness, sequential compactness, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8251.2 (CombSpace NameCert obligations)**.** *For every accepted* $\mathsf{CombSpace}^\uparrow$ *carrier*

$$K = (A, V, S, T, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CombSpace}^\uparrow}$ *surface consists exactly of carrier admission for the real-axis base windows, vertical tooth rows, accumulation-spine rows, topology classifier rows, metric handoff rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8251.1. The rows A, V, S display the base axis, the inspected teeth, and the accumulation-spine evidence before any topology or metric consumer can read the packet.

Next read T and M. The topology classifier is finite because it only inspects the displayed neighbourhood tests, and the metric handoff is finite because every distance row is tied to the shown real-name and stream-window data.

Finally use H, C, P, N. Transport is componentwise, replay follows the displayed rows, provenance names the source packets, and the local NameCert row closes the finite carrier surface. Since no carrier coordinate stores a host plane subset, quotient equality, selected infinite teeth, compactness theorem, or ambient completeness witness, those objects cannot enter the NameCert obligations. □

Theorem 8251.3 (CombSpace topology-metric boundary)**.** *Every topology-facing or metric-facing read of an accepted* $\mathsf{CombSpace}^\uparrow$ *packet factors through*

$$\mathsf{Topology}^\uparrow, \quad \mathsf{Met}^\uparrow, \quad \mathsf{Real}^\uparrow, \quad \mathsf{StreamName}^\uparrow$$

as displayed by A, V, S, T, M. The topology consumer receives only basic-neighbourhood classifier rows, and the metric consumer receives only finite distance-window handoffs. Neither consumer receives host point-set equality, a selected infinite tooth family, compactness, sequential compactness, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Start with the base and tooth rows A, V. They expose the finite real-axis windows and the inspected vertical teeth, so both topology and metric reads are tied to displayed observations.

Use the spine row S before reading either classifier. The accumulation behaviour is visible only through the finite stream windows recorded in the packet.

The topology row T then supplies neighbourhood tests, while the metric row M supplies distance-window handoffs. Both rows depend on the same displayed base, tooth, spine, real-name, and stream data.

The structural rows H, C, P, N preserve and name this route. Because the carrier has no coordinate for host point-set equality, a selected infinite tooth family, compactness, sequential compactness, or ambient completeness, the topology-metric boundary remains exactly the finite one stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CombSpace}^\uparrow}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CombSpace}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 8251.1, Definition 8251.2, and Definition 8251.3 seed the finite comb-space packet from real-axis base windows, vertical tooth rows, accumulation-spine rows, topology classifier rows, metric handoff rows, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CombSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public-object export, host point-set equality, selected infinite teeth, compactness, sequential compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, topology classifier exactness, metric handoff exactness, accumulation-spine routing, transport, replay, provenance, and local naming for BEDC.Derived.CombSpaceUp.

Constructive story: A $\mathsf{CombSpace}^{\uparrow}$ packet is generated from displayed real-axis base windows, vertical tooth rows, accumulation-spine rows, topology classifier rows, metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8252

A Concrete Naming Certificate for $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on finite-prefix continuity for choice sequences, the $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$ packet records the handoff that turns Brouwer-continuity evidence into a finite modulus ledger consumable by StreamName, RegSeqRat, and Real-name data. It sits between the continuity surface of chapter 3465, the choice-sequence observations of chapter 1482, the finite-window schedule of chapter 126, the rational readback of chapter 301, the endpoint seal of chapter 38, and the effective finite-prefix route of chapter 1743. The chapter packages the bridge as NameCert data; it does not import host functional continuity, countable choice, quotient stream equality, selected real representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8252.1 (Brouwer continuity modulus bridge carrier)**.** *A* $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, Q, S, R, E, M, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *row, naming the finite-prefix continuity principle whose output read is being bridged.*
2. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *finite-prefix row, recording only the observed choice-sequence cells admitted by the bridge.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the requested output window and the finite schedule used to answer it.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled stream window.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the choice prefix, stream window, rational readback, and modulus ledger have been displayed.*
6. *M is the finite modulus ledger, assigning each scheduled output window exactly the input-prefix budget used for replay.*
7. *H records componentwise* hsame *transport for $C, Q, S, R, E, M, K, P, N$.*
8. *K records displayed* Cont *replay from Brouwer continuity evidence to choice-sequence observations, from observations to StreamName windows, from windows to RegSeqRat readback, from the modulus ledger to the Real seal, and from the whole route to the local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted bridge packets only through $C, Q, S, R, E, M, H, K, P, N$. It has no coordinate for a completed host functional, a choice-selected modulus, quotient stream equality, a selected real limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8252.2 (Brouwer continuity modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$ *carrier $B = (C, Q, S, R, E, M, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityModul}}$ *surface consists exactly of carrier admission for the Brouwer-continuity row, choice-sequence finite-prefix row, StreamName window row, RegSeqRat readback row, Real seal row, finite modulus ledger, componentwise transport, displayed replay, provenance, and local naming rows. Each accepted output-window request must pass through the displayed finite-prefix route*

$$C, Q \longmapsto M \longmapsto S, R \longmapsto E \longmapsto H, K, P, N.$$

No local obligation is stored outside these displayed rows.

Proof. Project the accepted carrier. The rows C, Q, S, R, E, M are the only computational rows, and the rows H, K, P, N are the only structural rows.

The Brouwer-continuity row C supplies the finite-prefix dependence surface, while Q records the choice-sequence cells actually inspected by the route. The StreamName row S cannot answer a window request until the modulus ledger M displays the corresponding input-prefix budget.

Once the window is scheduled, R gives the regular rational readback and E records the endpoint seal. The transport, replay, provenance, and naming rows preserve that order and add no hidden selector or completion theorem.

Thus the local NameCert obligations are exactly the displayed bridge rows. A consumer seeking countable choice, quotient stream equality, a completed host functional, or ambient Real completeness must obtain it outside this packet. □

Theorem 8252.3 (Brouwer continuity modulus bridge StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$ *carrier must first read the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *row C, the* $\mathsf{ChoiceSequence}^{\uparrow}$ *finite-prefix row Q, and the finite modulus ledger M. The consumer may then inspect the scheduled StreamName window S, the RegSeqRat readback R, and the Real seal E. The handoff exports finite-prefix real-name evidence only; it exports no host functional continuity theorem, countable-choice selector, quotient stream equality, selected real representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a StreamName request. The request is represented by S, but the carrier permits S to be read only as part of the bridge route from C and Q through the ledger M.

Open the ledger row. Each entry of M names the input-prefix budget attached to one displayed output-window request, so the consumer receives a finite prefix rather than a total host function or a choice-selected modulus.

The readback rows then follow deterministically inside the packet. Row R records the regular rational data for the scheduled window, and row E seals the Real-name endpoint after that readback has been displayed.

The structural rows H, K, P, N replay and name the same route. Since none of the refused principles is a carrier coordinate, the StreamName handoff remains a finite-prefix real-name bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}$ packet is generated from Brouwer-continuity evidence, choice-sequence finite-prefix observations, StreamName windows, RegSeqRat readback, a Real seal, a finite modulus ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8252.1, Definition 8252.2, and Definition 8252.3 seed the finite-prefix bridge from Brouwer-continuity and choice-sequence data to StreamName, RegSeqRat, and Real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerContinuityModulusBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host functional continuity, countable choice, quotient stream equality, selected real representatives, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix exactness, StreamName window exactness, RegSeqRat readback, Real-seal handoff, modulus-ledger exactness, and nonescape for BEDC.Derived.BrouwerContinuityModulusBridg

Chapter 8253

A Concrete Naming Certificate for $\mathsf{BanachContractionSequence}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{BanachContractionSequence}^{\uparrow}$ packet records the finite Picard-orbit Cauchy handoff that sits between the contraction iteration surface and any completed fixed-point endpoint. It reads the $\mathsf{PicardIteration}^{\uparrow}$ orbit of chapter 2359, the $\mathsf{ContractionMapping}^{\uparrow}$ source of chapter 1418, the $\mathsf{CompleteMetric}^{\uparrow}$ endpoint-admission boundary of chapter 128, the $\mathsf{CauchyModulus}^{\uparrow}$ route of chapter 292, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the orbit-as-Cauchy carrier only; it does not assert a completed fixed point, host equality, a selected limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8253.1 (Banach contraction sequence carrier)**.** *A* $\mathsf{BanachContractionSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, I, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *row, carrying the self-map graph, contraction bound, and one-step comparison surface used by the orbit.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *row, exposing the finite orbit ledger and adjacent residual-distance rows.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row extracted from the displayed Picard residuals.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row generated from the orbit and modulus rows.*
5. *R is the* $\mathsf{CompleteMetric}^{\uparrow}$ *read of the same Cauchy data as endpoint-admissible input, not as a selected endpoint.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after the regular rational handoff has been displayed.*
7. *H records componentwise* hsame *transport for K, I, M, Q, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from contraction admission through Picard iteration, modulus extraction, regular rational handoff, complete-metric admission, and real sealing.*
9. *P is the* Pkg *provenance row over* $\mathsf{BanachContractionSequence}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachContractionSequence}^{\uparrow}}$ *row.*

The carrier exports a finite Cauchy handoff for the Picard orbit. It contains no coordinate for a fixed point, host function extensionality, quotient completion equality, a selected representative, countable choice, or ambient completeness.

Theorem 8253.2 (Banach contraction sequence NameCert obligations)**.** *For an accepted* $\mathsf{BanachContractionSequen}$ *carrier* $S = (K, I, M, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachContractionSequence}^{\uparrow}}$ *surface consists exactly of contraction-map visibility, Picard-orbit ledger visibility, residual-modulus exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff visibility,* $\mathsf{CompleteMetric}^{\uparrow}$ *endpoint-admission visibility, terminal* $\mathsf{Real}^{\uparrow}$ *seal visibility, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 8253.1.*

Proof. Project the carrier. The mathematical rows are K, I, M, Q, R, E, and the support rows are H, C, P, N.

Read K and I first. The contraction row supplies the self-map and contraction bound, while the Picard row supplies the finite orbit ledger and residual-distance comparisons.

Read M, Q, R, E next. The modulus row is extracted from the displayed residuals, the regular rational row is generated from that modulus, the complete-metric row admits the same Cauchy data as endpoint-facing input, and the real seal is downstream of the regular handoff.

The transport, replay, provenance, and naming rows preserve only this displayed route. Since no excluded endpoint or choice object appears in the carrier, the local NameCert cannot export one. □

Theorem 8253.3 (Banach contraction sequence Picard-Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BanachContractionSequence}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route turns the finite Picard orbit into named Cauchy handoff data before any endpoint-facing read is allowed. It does not supply a completed fixed point, selected limit, quotient stream equality, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the contraction and Picard rows. They display the self-map graph, the initial orbit data, the finite iterate ledger, and the residual-distance comparisons.

Apply the obligation theorem above. It places the $\mathsf{CauchyModulus}^{\uparrow}$ row downstream of those residuals and places the $\mathsf{RegSeqRat}^{\uparrow}$ handoff downstream of the modulus.

The complete-metric and real-seal rows are then read as boundaries for later consumers. They receive the named regular Cauchy handoff; they do not create an endpoint selector or a host fixed-point equality.

Thus the packet is precisely the finite Picard-orbit Cauchy handoff. Any consumer asking for a fixed point must add its own endpoint and uniqueness obligations after this route. □

Falsifiable prediction: Every accepted $\mathsf{BanachContractionSequence}^{\uparrow}$ read factors through one ContractionMapping row, one PicardIteration orbit row, one CauchyModulus row, one RegSeqRat handoff, one CompleteMetric endpoint-admission row, one RealUp seal, hsame transport, Cont replay, Pkg provenance, and NameCert row before any fixed-point endpoint is consumed.

Independence witness: picard-iteration, contractionmapping, completemetric, cauchymodulus, regseqrat, real: none is bijective with $\mathsf{BanachContractionSequence}^{\uparrow}$ because the packet jointly stores the contraction source, finite Picard orbit, residual modulus, regular rational handoff, complete-metric admission boundary, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachContractionSequence}^{\uparrow}$ packet is generated from a ContractionMapping row, a PicardIteration orbit row, a CauchyModulus row, a RegSeqRat handoff, a CompleteMetric endpoint-admission row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContractionSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8253.1, Definition 8253.2, and Definition 8253.3 seed the finite Picard-orbit Cauchy handoff route without asserting a completed fixed point.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachContractionSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed fixed point, selected limit, quotient stream equality, host equality, host function extensionality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked contraction-map visibility, Picard-orbit visibility, residual-modulus extraction, RegSeqRat handoff exactness, CompleteMetric endpoint-admission nonescape, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 8254

A Concrete Naming Certificate for MetacicConfluenceResidualFrontier$^{\uparrow}$

The MetacicConfluenceResidualFrontier$^{\uparrow}$ packet realizes the MetaCIC confluence audit map of chapter 9378 as finite retained-premise, candidate-join, obstruction, and closed-fragment NameCert data. It sits after the typed-window and redex-frontier packets of chapter 6433 and chapter 1008, the local-join ledgers of chapter 4126, the residual checker and envelope of chapter 3827 and chapter 3921, and the closed-normal confluence seal of chapter 741. The packet keeps the substitution-compatibility boundary visible; it does not claim full Church–Rosser, arbitrary closed-term confluence, or discharge of residual substitution.

Definition 8254.1 (MetaCIC confluence residual frontier carrier)**.** *A* MetacicConfluenceResidualFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, R, J, O, D, S, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed MetaCIC source-window row naming the closed or open term shape under inspection.*
2. *R is the residual frontier row, recording the beta redex positions and residual-substitution sites that remain visible to the audit map.*
3. *J is the finite candidate-join row for the inspected local peak.*
4. *O is the obstruction-retention row naming the residual-substitution compatibility input when the candidate join cannot be closed.*
5. *D is the closed-fragment discharge row for normal, atom, or other shape-limited cases already available in the confluence substrate.*
6. *S is the substitution-boundary handoff row, kept as a premise boundary rather than erased into a global theorem.*
7. *A is the audit-map readback row tying T, R, J, O, D, S to the confluence audit map.*
8. *H records componentwise* hsame *transport for all displayed frontier rows and the local certificate row.*
9. *C records displayed* Cont *replay from typed window through residual frontier, candidate join, obstruction retention, closed-fragment discharge, substitution boundary, and audit readback.*

10. *P is the* Pkg *provenance row over* $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$*, MetaCIC reduction rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}}$ *row.*

The carrier exports a finite confluence-frontier packet. It contains no coordinate for full parallel diamond discharge, arbitrary closed-star confluence, global Church–Rosser, or residual-substitution compatibility as a theorem.

Theorem 8254.2 (MetaCIC confluence residual frontier NameCert obligations)**.** *For an accepted* $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ *carrier* $F = (T, R, J, O, D, S, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}}$ *surface consists exactly of typed-window visibility, residual-frontier visibility, candidate-join visibility, obstruction-retention visibility, closed-fragment discharge visibility, substitution-boundary handoff, audit-map readback, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 8254.1.*

Proof. Project the carrier. The confluence-facing rows are T, R, J, O, D, S, A, and the structural rows are H, C, P, N.

Read the typed source and residual frontier first. These rows identify the local peak and the residual-substitution sites that the audit map refuses to hide.

Read the candidate join and obstruction rows next. A closed fragment may discharge through D, while any remaining substitution compatibility demand stays named in O and S.

Finally read the audit-map and structural rows. They preserve the finite frontier under transport, replay, provenance, and local naming, but they do not add a theorem proving full confluence or residual substitution compatibility. □

Theorem 8254.3 (MetaCIC confluence residual frontier vision handoff)**.** *Every manuscript citation of* $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ *must read the confluence audit map through the displayed finite route*

$$T,\ R,\ J,\ O,\ D,\ S,\ A,\ H,\ C,\ P,\ N.$$

Closed normal and atom fragments may be consumed through D, and conditional parallel-diamond or closed-star confluence handoffs may be recorded through S. The route does not turn the residual-substitution boundary into a discharged theorem, and it does not assert full Church–Rosser or arbitrary closed-term confluence for BetaStarStep*.*

Proof. Start from the carrier inventory. The only rows that speak about confluence are the typed window, residual frontier, candidate join, obstruction, closed-fragment discharge, substitution-boundary handoff, and audit readback rows.

Apply the NameCert obligations. They require the obstruction and substitution-boundary rows to remain displayed whenever the candidate join is not a shape-limited closed fragment.

The closed-fragment row may be consumed for the normal and atom cases isolated by the audit map. It does not rewrite the retained substitution boundary into a global parallel diamond.

Thus the packet is a finite handoff from the vision audit map to concrete NameCert data. Any future full confluence theorem must supply separate residual-substitution discharge before this frontier can be read as a closed Church–Rosser proof. □

The future TasteGate surface for $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ must provide explicit Nontrivial MetacicCon and FieldFaithful $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ instances. Nontriviality must separate a packet with retained residual-substitution obstruction from a packet claiming discharge, and field faithfulness must preserve $T, R, J, O, D, S, A, H, C, P, N$ as distinct rows.

Falsifiable prediction: Every accepted $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ read keeps a typed source window, residual frontier, candidate join, obstruction-retention row, closed-fragment row, substitution-boundary handoff, audit readback, hsame transport, Cont replay, Pkg provenance, and NameCert row visible before any confluence consumer cites it.

Independence witness: metacic-confluence-audit-map, metacic-residual-checker, metacic-confluence-residual-envelope, closednormalconfluenceseal, metacic-local-join-ledger: none is bijective with $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ because the packet jointly stores retained premises, candidate joins, obstruction rows, closed-fragment discharge, substitution-boundary handoff, audit readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}$ packet is generated from a typed source window, residual frontier, candidate join, obstruction-retention row, closed-fragment discharge row, substitution-boundary handoff, audit-map readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8254.1, Definition 8254.2, Definition 8254.3, the falsifiable prediction, and the independence witness seed the finite residual-frontier packet for the MetaCIC confluence audit boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicConfluenceResidualFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full parallel diamond discharge, arbitrary closed-star confluence, global Church–Rosser, residual-substitution compatibility as a theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked typed-window visibility, residual-frontier visibility, candidate-join visibility, obstruction retention, closed-fragment discharge, substitution-boundary handoff, audit-map readback, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming.

Chapter 8255

A Concrete Naming Certificate for $\mathsf{TheoremOfAscoli}^{\uparrow}$

Following Arzela, Ascoli, Kelley, and Bishop–Bridges, the $\mathsf{TheoremOfAscoli}^{\uparrow}$ packet records the compact-open theorem surface as finite BEDC data. It sits after the compact-open topology route of chapter 4138, the Arzela–Ascoli family surface of chapter 412, the compact metric source of chapter 129, and the real-name readback of chapter 38. The packet names the theorem-facing handoff from compact metric data, compact-open topology, equicontinuity, bounded-family control, finite stream windows, regular rational readback, and a terminal real seal; it does not add a host function space or a subsequence selector.

Definition 8255.1 (TheoremOfAscoli carrier)**.** *A* $\mathsf{TheoremOfAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, O, E, B, A, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric observations consumed by the theorem surface.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *row, naming the compact-open neighbourhood and finite subbasic-window route read by consumers.*
3. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, carrying the shared equicontinuity ledger for the displayed family.*
4. *B is the* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *row, carrying the pointwise and sup-norm bound ledger visible to the compact-open consumer.*
5. *A is the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *row, recording the finite extraction and compactness-facing dependency already present in the Arzela–Ascoli packet.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite observation schedule for compact-open convergence reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying regular rational comparison data for the displayed windows.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal, recording only terminal real values already read through R.*
9. *H records componentwise* hsame *transport for all displayed theorem rows.*

10. *C records displayed* Cont *replay through*

$$K, O, E, B, A, W, R, S \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{TheoremOfAscoli}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{BoundedFunctionFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoremOfAscoli}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, E, B, A, W, R, S, H, C, P, N$. It has no coordinate for host function-space compactness, a selected subsequence, quotient function equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8255.2 (TheoremOfAscoli NameCert obligations)**.** *For every accepted* $\mathsf{TheoremOfAscoli}^{\uparrow}$ *carrier $T = (K, O, E, B, A, W, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TheoremOfAscoli}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the compact metric source and compact-open topology rows K, O;*
2. *family-control admission for the equicontinuity, bounded-family, and Arzela–Ascoli rows E, B, A;*
3. *value-window admission for the stream-window, regular-rational, and real-seal rows W, R, S;*
4. *componentwise transport and replay through H, C;*
5. *provenance and terminal local naming through P, N.*

No accepted packet can export host function-space compactness, selected subsequences, quotient function equality, open-cover compactness, ambient real completeness, Lean verification, or bridge checking as part of this local surface.

Theorem 8255.3 (TheoremOfAscoli compact-open handoff)**.** *Every compact-open Ascoli consumer of an accepted* $\mathsf{TheoremOfAscoli}^{\uparrow}$ *carrier must first pass through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{CompactOpenTopology}^{\uparrow}, \quad \mathsf{EquicontinuityFamily}^{\uparrow}, \quad \mathsf{BoundedFunctionFamily}^{\uparrow}, \quad \mathsf{ArzelaAscoli}^{\uparrow},$$

before exporting a compact-open convergence read. The exported read is therefore a finite route through $K, O, E, B, A, W, R, S, H, C, P, N$, not a function-space compactness theorem, subsequence choice, quotient function equality, open-cover compactness principle, or ambient completeness assertion.

Closure status for $\mathsf{NameCert}_{\mathsf{TheoremOfAscoli}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoremOfAscoli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8255.1, Definition 8255.2, and Definition 8255.3 seed the finite compact-open Ascoli packet from compact metric, compact-open topology, family-control, value-window, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoremOfAscoli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space compactness, selected subsequences, quotient function equality, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open topology routing, family-control routing, value-window exactness, transport, replay, provenance, and local naming for BEDC.Derived.TheoremOfAscoliUp.

Constructive story: A $\mathsf{TheoremOfAscoli}^{\uparrow}$ packet is generated from CompactMetricUp, CompactOpenTopologyUp, EquicontinuityFamilyUp, BoundedFunctionFamilyUp, ArzelaAscoliUp, StreamNameUp, RegSeqRatUp, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8256

A Concrete Naming Certificate for $\mathsf{HyperbolicCannonThurstonBoundary}^{\uparrow}$

Following Cannon and Thurston, the $\mathsf{HyperbolicCannonThurstonBoundary}^{\uparrow}$ packet records a finite BEDC boundary-extension handoff for hyperbolic visual-boundary reads. It realizes the observer-expansion route of chapter 9463 after the boundary-shadow packet of chapter 3813, the visual-boundary route of chapter 4181, the hyperbolic metric packet of chapter 3474, the Poincare disk metric of chapter 6209, the Gromov boundary route of chapter 6470, and the terminal real-name seal of chapter 38. The packet names only displayed shadow, visual-boundary, metric, disk, Gromov-boundary, value-readback, transport, replay, provenance, and local naming rows; it does not assert a host group action, completed quotient boundary, selected geodesic ray, or ambient compactification theorem.

Definition 8256.1 (HyperbolicCannonThurstonBoundary carrier)**.** *A* $\mathsf{HyperbolicCannonThurstonBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, V, M, D, G, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, recording the visible boundary-shadow window inspected by the boundary-extension read.*
2. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row, exposing the visual-boundary face used by the extension handoff.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, carrying finite metric comparisons attached to the same boundary and disk observations.*
4. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *row, recording the disk-metric readback for finite Poincare windows.*
5. *G is the* $\mathsf{GromovBoundary}^{\uparrow}$ *row, carrying the ray-boundary token through which the extension is read.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing finite observation schedules for boundary and disk reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying regular rational comparison data for those windows.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, recording terminal real values only after W and R are visible.*

9. *H records componentwise* hsame *transport for every displayed row of Q.*

10. *C records displayed* Cont *replay through*

$$S, V, M, D, G, W, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* HyperbolicCannonThurstonBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, GromovBoundary$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{HyperbolicCannonThurstonBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, V, M, D, G, W, R, E, H, C, P, N$. It has no coordinate for a host Cannon–Thurston map, completed ideal-boundary quotient, selected ray representative, global hyperbolic group action, or ambient compactification theorem.

Theorem 8256.2 (HyperbolicCannonThurstonBoundary NameCert obligations)**.** *For every accepted* HyperbolicCannonThurstonBoundary$^{\uparrow}$ *carrier $Q = (S, V, M, D, G, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicCannonThurstonBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the boundary-shadow and visual-boundary rows, metric and Poincare disk readback, Gromov-boundary routing, finite stream and regular-rational readback, terminal real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Theorem 8256.3 (HyperbolicCannonThurstonBoundary shadow handoff)**.** *Every observer-expansion consumer of an accepted* HyperbolicCannonThurstonBoundary$^{\uparrow}$ *carrier must first read*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, GromovB

before it exports a boundary-extension read. The exported read is a finite route through $S, V, M, D, G, W, R, E, H, C,$ and exports no host group action, quotient ideal boundary, selected geodesic ray, completed visual compactification, or ambient real-completeness witness.

Closure status for NameCert$_{\text{HyperbolicCannonThurstonBoundary}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicCannonThurstonBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 8256.1, Definition 8256.2, and Definition 8256.3 seed the finite boundary-extension packet from boundary-shadow, visual-boundary, metric, disk, Gromov-boundary, stream, rational, real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicCannonThurstonBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host Cannon–Thurston map, global hyperbolic group action, completed quotient boundary, selected geodesic rays, ambient compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow routing, visual-boundary routing, metric and disk readback, Gromov-boundary routing, value-readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicCannonThurstonBoundaryUp.

Constructive story: A HyperbolicCannonThurstonBoundary$^{\uparrow}$ packet is generated from HyperbolicBoundaryShadowUp, HyperbolicVisualBoundaryUp, HyperbolicMetricUp, PoincareDiskMetricUp, GromovBoundaryUp, StreamNameUp, RegSeqRatUp, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8257

A Concrete Naming Certificate for $\mathsf{EntourageFilter}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures and filters, the $\mathsf{EntourageFilter}^{\uparrow}$ packet names the finite route by which displayed entourages generate a filter-base surface before any Cauchy-filter or completion-facing consumer can read it. It sits between the uniform-space carrier of chapter 339, the filter-base route of chapter 1601, the Cauchy-filter surface of chapter 333, the real-uniform route of chapter 1624, and the dyadic tolerance ledger of chapter 291. The packet is a finite entourage-to-filter interface; it is not a quotient filter, a selected limit, countable choice, host equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8257.1 (Entourage-filter carrier)**.** *An* $\mathsf{EntourageFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, B, G, F, D, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage-data row, exposing only the displayed diagonal, symmetry, refinement, and composition requests.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *generation row, recording the finite basis entries generated from the displayed entourage requests.*
3. *G is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, naming the Cauchy-facing consumer only after the generated filter-base row has been displayed.*
4. *F is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *completion-facing route row, used only as a reader of the displayed uniform entourage and filter-base data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and radius row, recording precision requests when an L10 completion route compares entourage requests with dyadic bounds.*
6. *H records componentwise* hsame *transport for the uniform entourage, filter-base generation, Cauchy handoff, real-uniform route, dyadic tolerance, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an entourage request through U, B, G, F, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{EntourageFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{EntourageFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, B, G, F, D, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable-choice schedules, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8257.2 (Entourage-filter NameCert obligations)**.** *For every accepted* $\mathsf{EntourageFilter}^{\uparrow}$ *carrier*

$$E = (U, B, G, F, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EntourageFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform entourage row U, filter-base generation through B, Cauchy-filter handoff through G, completion nonescape through the real-uniform route F and dyadic tolerance row D, and provenance locality through H, C, P, N.*

Proof. Project the carrier of Definition 8257.1. The accepted packet exposes exactly the uniform entourage, filter-base, Cauchy handoff, real-uniform route, dyadic tolerance, transport, replay, provenance, and local naming rows.

The entourage-to-filter generation is the passage from U to B. The row U supplies displayed uniform-space requests, and B records the finite basis entries generated from those requests before any Cauchy-facing row is public.

The Cauchy and completion-facing reads are downstream. The row G names the $\mathsf{CauchyFilter}^{\uparrow}$ handoff only after B has been displayed, while F and D record the real-uniform and dyadic tolerance route used by completion consumers.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded quotient, selected limit, countable-choice, host-equality, or ambient-completeness coordinate occurs in the carrier, the listed rows exhaust the local obligations. □

Theorem 8257.3 (Entourage-filter Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{EntourageFilter}^{\uparrow}$ *carrier $E = (U, B, G, F, D, H, C, P, N)$ reads through*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{RealUniformStructure}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame},$$

before any completion use is available. The route exports no quotient filter, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 8257.2. It places every accepted completion-facing read on the displayed uniform entourage row, generated filter-base row, Cauchy-filter handoff row, real-uniform route row, and dyadic tolerance row.

Read U and B first. This makes the entourage request and its generated filter-base entries visible before a consumer can ask for Cauchy readiness through G.

Next read G, F, D. The Cauchy handoff remains a finite consumer row, while the real-uniform and dyadic rows record only the tolerance and radius data needed by the L10 completion route.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Therefore any completion use is downstream of the displayed packet, and the excluded quotient, choice, host-equality, selected-limit, and ambient-completeness endpoints are unavailable. □

Falsifiable prediction: An accepted $\mathsf{EntourageFilter}^{\uparrow}$ completion read is refuted if it reaches a Cauchy-filter or real-uniform completion consumer without displaying the UniformSpace entourage

row, FilterBase generation row, CauchyFilter handoff row, DyadicRatCore tolerance row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: uniformspace, filterbase, cauchyfilter, real-uniform-structure, dyadicratcore: no listed sibling supplies the whole $\mathsf{EntourageFilter}^{\uparrow}$ packet because the packet jointly binds displayed uniform entourages, finite filter-base generation, Cauchy handoff, completion-facing tolerance data, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EntourageFilter}^{\uparrow}}$.

Constructive story: An $\mathsf{EntourageFilter}^{\uparrow}$ packet is generated from UniformSpace entourage data, FilterBase generation rows, a CauchyFilter handoff row, RealUniformStructure completion-facing route data, DyadicRatCore tolerance and radius rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EntourageFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8257.1, Definition 8257.2, and Definition 8257.3 seed the finite route by which displayed uniform entourages generate a filter-base surface for Cauchy-filter and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EntourageFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage-to-filter generation, Cauchy handoff, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EntourageFilterUp.

Chapter 8258

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$ packet records the finite route by which located real comparison data is read before a Bishop completion-pair consumer reaches the terminal Real seal. It sits over the located real comparison source of chapter 1933, the Bishop completion-pair route of chapter 6298, the dyadic tolerance core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite located completion-pair NameCert surface; it is not a quotient completion, selected representative, host limit, countable-choice schedule, located-trichotomy oracle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8258.1 (Bishop located completion-pair carrier)**.** *A* $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, M, D, W, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, exposing finite located-real windows and comparison evidence before any completion-facing read proceeds.*
2. *M is the metric-source row for the pre-completion space.*
3. *D is the dense-route row tying source points to the Bishop completion-pair route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by completion and Real consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row over the windows in W.*
6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling located comparison, regular readback, and seal requests.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff, available only after L, M, D, W, R, T have been displayed.*
8. *H records componentwise* hsame *transport for the located, metric-source, dense-route, window, readback, tolerance, Real seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from each located completion-pair request through $L, M, D, W, R, T, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionPair}^{\uparrow}}$ *row.*

The carrier stores no quotient class, selected representative, host real equality, countable-choice schedule, hidden completed point, located-trichotomy row, or ambient completeness witness.

Theorem 8258.2 (Bishop located completion-pair NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$ *carrier* $K = (L, M, D, W, R, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletio}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located-real row L, metric-source row M, dense-route row D,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, and* $\mathsf{Real}^{\uparrow}$ *seal handoff E;*
2. *located-source routing, requiring every completion-facing read to display L before the dense completion-pair route is consumed;*
3. *dense-completion routing, requiring M and D to be read before any window, readback, tolerance, or seal row is public;*
4. *regular-Cauchy exactness, requiring W, R, T to be read as one finite window-and-tolerance packet before E is exported;*
5. *Real-seal non-escape, requiring every Real-facing read to pass through L, M, D, W, R, T before it reaches E;*
6. *transport and replay through H and C, preserving the same rows componentwise;*
7. *ledger refusal through P and N, excluding quotient completion, selected representatives, host real equality, countable choice, located trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8258.1. Its analytic rows are exactly L, M, D, W, R, T, E, and its structural rows are exactly H, C, P, N.

The located-source route begins at L. A completion-facing consumer may use located comparison data only after the finite located-real row is displayed, so the consumer does not obtain a host trichotomy or quotient-real comparison.

The completion-pair route then reads M and D, followed by the finite windows W, regular readback R, and dyadic tolerance ledger T. The Real seal row E is downstream of that ordered route.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for the refused host objects, the seven listed rows exhaust the local NameCert surface. □

Theorem 8258.3 (Bishop located completion-pair finite means)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after the located row, dense row, finite windows, regular readback, and dyadic tolerance ledger are displayed:*

$$L \longmapsto M \longmapsto D \longmapsto W \longmapsto R \longmapsto T \longmapsto E.$$

Thus the completion-pair read is finite means data over $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, not a quotient class, selected representative, countable-choice schedule, host comparison oracle, or ambient completeness proof.*

Proof. Use the NameCert obligations of Definition 8258.2. Located-source routing exposes L before any completion row is consumed, and dense-completion routing then exposes M and D before the naming rows are interpreted.

Regular-Cauchy exactness ties the windows, readback, and tolerance ledger into the single finite packet W, R, T. The seal row E is admitted only after that packet is displayed, so a consumer has no route to E that bypasses located evidence, dense routing, finite windows, regular readback, or dyadic tolerances.

The transport, replay, provenance, and local naming rows preserve this order componentwise. Hence every completion-facing read reaches the Real seal by finite means alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionPair}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$ packet is generated from located-real comparison data, source metric data, a dense completion-pair route, StreamName finite windows, RegSeqRat regular-Cauchy readback, DyadicRatCore tolerance ledgers, a Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8258.1, Definition 8258.2, and Definition 8258.3 seed the finite located Bishop completion-pair route from located comparison data through dense completion, finite windows, regular readback, dyadic tolerance, and Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host real equality, located trichotomy, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopLocatedCompletionPair}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, located-source routing, dense-completion routing, regular-Cauchy exactness, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedComplet

Chapter 8259

A Concrete Naming Certificate for $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ packet records the finite witness grid by which a regular Cauchy construction answers a displayed family of precision requests without appealing to quotient sequence equality, countable choice, selected limits, or ambient completeness. It sits on the Real and Completion route after the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the finite threshold-selection surface of chapter 7210. The packet names a displayed matrix of witness rows; it does not name a completed real, a quotient class, or an unbounded selector.

Definition 8259.1 (Finite Cauchy witness matrix carrier)**.** *A* $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (A, T, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite precision-request axis, listing the requests whose Cauchy witnesses are being recorded.*
2. *T is the finite threshold axis, giving the admitted threshold row for each request in A.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window matrix, recording the finite stream windows inspected at those thresholds.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance matrix, recording dyadic radii, denominator budgets, slack comparisons, and rowwise inequalities.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback matrix, obtained from the same windows and dyadic rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after A, T, S, D, R have been displayed.*
7. *H records componentwise* hsame *transport for requests, thresholds, windows, dyadic tolerances, readbacks, seal, provenance, and local naming.*
8. *C records displayed* Cont *replay from each precision request through its threshold, window, tolerance, readback, and seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$, $\mathsf{BishopCompletionModulusSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}}$ *row.*
The classifier compares accepted matrices only through these rows. It has no coordinate for quotient stream equality, a selected completed limit, countable choice over precision requests, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8259.2 (Finite Cauchy witness matrix NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ *carrier* $W = (A, T, S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite precision requests, finite thresholds,* $\mathsf{StreamName}^{\uparrow}$ *window matrix,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance matrix,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback matrix, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer that reads a Cauchy witness row must display the same* $A, T, S, D, R, E, H, C, P, N$ *packet before exporting its conclusion.*

Proof. Project the carrier of Definition 8259.1. The analytic rows are the request axis, threshold axis, stream-window matrix, dyadic tolerance matrix, readback matrix, and Real seal.

The threshold row is finite and displayed. The Bishop completion modulus selection route of chapter 7210 supplies the sibling discipline for reading a threshold before a window or tolerance row is consumed.

The window, tolerance, and readback rows are then forced to remain aligned. The sibling routes chapter 126, chapter 291, and chapter 301 provide the finite observation, dyadic, and regular-rational surfaces used by S, D, R.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite matrix. Since the carrier contains no quotient, selected limit, choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8259.3 (Finite Cauchy witness matrix Real handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$A, T \longmapsto S, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the handoff exposes finite precision requests, finite thresholds, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming before any Real-facing conclusion is exported.*

Proof. Start from the NameCert obligations. They require every witness read to display $A, T, S, D, R, E, H, C, P, N$ as one finite packet.

Read the threshold axis before the analytic rows. The selection sibling chapter 7210 keeps T as a displayed finite threshold row rather than a hidden choice function.

Next read S, D, R through the StreamName, DyadicRatCore, and RegSeqRat siblings. The same finite windows and tolerance entries feed the readback matrix, so the consumer cannot replace the matrix by quotient stream equality or by a selected limit.

The terminal handoff is the seal row E of chapter 38, transported and replayed by H and C and sourced by P, N. Hence a Real or Completion consumer receives only the displayed finite witness matrix and not countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the finite request axis, threshold axis, StreamName windows, DyadicRatCore tolerances, and RegSeqRat readbacks refutes the carrier surface.

Independence witness: streamname, dyadicratcore, regseqrat, real, bishop-completion-modulus-selection: no listed sibling is bijective to $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ because this packet jointly

carries finite precision requests, finite thresholds, windows, tolerances, readbacks, a Real seal, transport, replay, provenance, and local NameCert rows as one witness matrix.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}$ packet is generated from finite precision requests, finite threshold rows, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8259.1, Definition 8259.2, Definition 8259.3, the falsifiable prediction, and the independence witness seed the finite witness-grid route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyWitnessMatrix}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected completed limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, threshold-row exposure, StreamName window readback, DyadicRatCore tolerance routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyWitnessMatrixUp.

Chapter 8260

Finite Uniform Discrepancy NameCert Construction

Following Weyl and Kuipers–Niederreiter, the $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ packet records finite uniform-discrepancy ledgers over dyadic real windows as BEDC inspection data. It reads low-discrepancy sequence windows from chapter 5173, dyadic rational budgets from chapter 291, finite cell coverage from chapter 1767, terminal real-name sealing from chapter 38, and finite observation schedules from chapter 126. The packet names a finite discrepancy ledger for displayed windows; it is not an asymptotic equidistribution theorem, a selected limiting density, a host measure equality, or an ambient completeness principle.

Definition 8260.1 (Finite uniform discrepancy carrier)**.** *A* $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, W, Q, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. L *is the* $\mathsf{LowDiscrepancySequence}^{\uparrow}$ *source row supplying the finite sampled sequence segment;*
2. W *is a finite ledger of dyadic test windows, each with displayed endpoints and a requested scale;*
3. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger used to compare observed counts with dyadic volume rows;*
4. E *is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *cell ledger whose boxes refine the displayed windows at the active scale;*
5. R *is the* $\mathsf{Real}^{\uparrow}$ *readback and seal row for the dyadic volumes and discrepancy bounds named by* W, Q, E*;*
6. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule used to inspect the sampled points and endpoint names;*
7. H *records componentwise* hsame *transport for sequence windows, dyadic tolerances, net cells, count ledgers, Real seal rows, and stream schedules;*
8. C *records displayed* Cont *replay from the sampled sequence through window counting, dyadic comparison, finite-net coverage, StreamName inspection, and Real sealing;*
9. P *is the* Pkg *provenance over* $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$, $\mathsf{LowDiscrepancySequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteUniformDiscrepancy}^{\uparrow}}$ *row.*

The carrier exports only finite sampled windows, dyadic count comparisons, finite-net cells, Real seal readback, transport, replay, provenance, and local naming. It contains no row for an infinite limiting distribution, selected subsequence, host measure space, or asymptotic discrepancy principle.

Theorem 8260.2 (Finite uniform discrepancy NameCert obligations)**.** *For every accepted* FiniteUniformDiscrepancy *carrier $D = (L, W, Q, E, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteUniformDiscrepancy}^{\uparrow}}$ *obligations are exactly carrier admission, finite-window exposure, dyadic count comparison, finite-net coverage, StreamName observation, Real seal descent, componentwise transport, displayed replay, provenance, and local naming.*

Every discrepancy read factors through $L, W, Q, E, R, S, H, C, P, N$. The packet exports no asymptotic equidistribution theorem, selected limiting density, quotient stream equality, host measure equality, countable-choice sampling schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier. The sampled sequence segment is visible only through L, and the tested boxes are the displayed dyadic windows in W.

The discrepancy comparison is finite. Counts are compared with dyadic volumes through Q, and the window coverage is refined through the finite cell ledger E before any Real-facing read reaches R.

The observation and structural rows S, H, C, P, N keep this route attached to the same finite packet. Since no carrier coordinate names an infinite limiting distribution, host measure equality, selected subsequence, or ambient completeness principle, the local certificate surface is exhausted by the listed rows. □

Theorem 8260.3 (Finite uniform discrepancy L10 route)**.** *Every L10 consumer of an accepted* $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{LowDiscrepancySequence}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nar}$$

before consuming a discrepancy bound. The route first exposes the sampled low-discrepancy segment, then the dyadic window and tolerance ledger, then the finite epsilon-net cells, then the stream inspection schedule, and only then the terminal Real seal for the displayed discrepancy bound.

Thus the packet is a finite quantitative-analysis object over dyadic real windows, not a statement about asymptotic uniform distribution or a completed measure-theoretic limit.

Proof. Start with the sampled sequence row supplied by $\mathsf{LowDiscrepancySequence}^{\uparrow}$. A consumer receives only the finite segment named in the carrier, so every later count is tied to displayed sampled points.

Next consume the dyadic and finite-net rows. $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the arithmetic budgets for window volume and count comparison, while $\mathsf{FiniteEpsilonNet}^{\uparrow}$ supplies the finite cells used to refine the inspected boxes.

Finally read the finite observation schedule and Real seal. $\mathsf{StreamName}^{\uparrow}$ displays the inspected windows, and $\mathsf{Real}^{\uparrow}$ seals only the discrepancy bounds already produced from L, W, Q, E. The structural rows $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ preserve this route and exclude any bypass through asymptotic sequence principles or ambient measure completeness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ read exposes the sampled low-discrepancy segment, dyadic tolerance ledger, finite epsilon-net cells, StreamName windows, Real seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before a discrepancy bound is consumed.

Independence witness: low-discrepancy-sequence, dyadicratcore, finite-epsilon-net, real, streamname: no listed sibling is bijective with $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ because this packet alone binds a finite sampled segment, dyadic window ledger, finite coverage cells, count comparison, Real seal, transport, replay, provenance, and local naming into one discrepancy surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteUniformDiscrepancy}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteUniformDiscrepancy}^{\uparrow}$ packet is generated from a LowDiscrepancySequenceUp finite segment, dyadic window and tolerance rows, finite epsilon-net cells, StreamName observation windows, a RealUp discrepancy-bound seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteUniformDiscrepancy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8260.1, Definition 8260.2, and Definition 8260.3 bind the seed finite-discrepancy carrier and its L10 route through low-discrepancy, dyadic, finite-net, stream, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteUniformDiscrepancy}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FiniteUniformDiscrepancyUp`

Bridge status: none

Not claimed: This chapter does not claim asymptotic equidistribution, host measure equality, selected limiting density, quotient stream equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic count comparison, finite-net coverage, StreamName observation, Real seal descent, transport, replay, provenance, and local naming for BEDC.Derived.FiniteUniformDiscrepancyUp.

Chapter 8261

A Concrete Naming Certificate for $\mathsf{SobczykTheorem}^{\uparrow}$

Following Sobczyk and the constructive Banach-space reading of separable extension theory, the $\mathsf{SobczykTheorem}^{\uparrow}$ packet records only the finite projection data visible when a displayed separable Banach row is read through a named c_0 subspace boundary. It sits over the Banach carrier of chapter 90, the norm and linear-map rows of chapter 89 and chapter 48, the finite observation discipline of chapter 126, and the Hahn–Banach boundary of chapter 3922. The packet names a finite-modulus projection corridor; it does not assert full Hahn–Banach, unrestricted choice of extensions, arbitrary complementability, or a host-level projection operator on every Banach space.

Definition 8261.1 (Sobczyk theorem carrier)**.** *A* $\mathsf{SobczykTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, C, E, M, P_0, B, H, R, Q, N)$$

with the following displayed rows:

1. *X is the separable* $\mathsf{Banach}^{\uparrow}$ *source row whose dense finite inspection schedule is read through* $\mathsf{StreamName}^{\uparrow}$*.*
2. *C is the named c_0 subspace row, equipped with its inherited* $\mathsf{Norm}^{\uparrow}$ *and inclusion as a displayed* $\mathsf{LinMap}^{\uparrow}$ *row.*
3. *E is the finite extension-window row recording the bounded coordinate functionals inspected at the current schedule depth.*
4. *M is the projection-modulus row, giving a rational bound for the finite coordinate projection read from E.*
5. *P_0 is the displayed partial projection row from the inspected window of X back to the c_0 row.*
6. *B is the boundedness ledger comparing the norm of P_0 against M inside the same finite window.*
7. *H records componentwise* hsame *transport for the source, subspace, window, modulus, projection, and bound rows.*
8. *R records displayed* Cont *replay from separable Banach source through the finite extension window to the projection row.*
9. *Q is the* Pkg *provenance row over* $\mathsf{SobczykTheorem}^{\uparrow}$*,* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SobczykTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global Hahn–Banach oracle, a choice-selected extension family, a quotient Banach-space equality, an unbounded projection, or a complementability assertion outside the displayed separable c_0 corridor.

Theorem 8261.2 (Sobczyk theorem NameCert obligations)**.** *For an accepted* $\mathsf{SobczykTheorem}^{\uparrow}$ *carrier $S = (X, C, E, M, P_0, B, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{SobczykTheorem}^{\uparrow}}$ *surface consists exactly of separable Banach-source visibility, c_0 subspace visibility, finite extension-window visibility, projection-modulus visibility, displayed partial-projection visibility, boundedness ledgering, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 8261.1.*

In particular, every public Sobczyk read is a constructive finite-modulus projection read: it presents a bounded projection witness over the inspected window and its replay ledger before any consumer may cite complementability. It refuses full Hahn–Banach, unrestricted extension choice, ambient quotient equality, or an unbounded host projection as part of the certificate.

Proof. Project the carrier. The analytic rows are X, C, E, M, P_0, B, and the support rows are H, R, Q, N.

Read X and C first. The source row supplies the separable Banach inspection schedule, while the c_0 row supplies the named subspace together with its norm and inclusion rows.

Read E, M, P_0, B next. The extension-window row lists the finitely inspected coordinate data, the modulus row bounds the projection read on that window, the partial projection row displays the map back to c_0, and the boundedness ledger checks the same modulus against the displayed norm rows.

Finally use transport, replay, provenance, and local naming. These rows preserve the finite projection corridor under NameCert reading, but they do not add a global Hahn–Banach principle, a choice-selected extension family, quotient equality, or an unrestricted complementability theorem. □

Falsifiable prediction: Any accepted $\mathsf{SobczykTheorem}^{\uparrow}$ read that cites complementability without exposing a separable Banach source row, a named c_0 subspace row, a finite extension window, a projection modulus, a displayed partial projection, boundedness, transport, replay, provenance, and local naming refutes the packet.

Independence witness: banach, norm, linearmap, streamname, hahn-banach: none is bijective with $\mathsf{SobczykTheorem}^{\uparrow}$ because the packet jointly stores the separable source, c_0 subspace boundary, finite extension window, bounded projection modulus, partial projection row, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{SobczykTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobczykTheorem}^{\uparrow}$ packet is generated from a separable Banach source row, a named c0 subspace row, finite extension-window data, a projection modulus, a partial projection row, boundedness ledgering, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobczykTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8261.1 and Definition 8261.2 seed the finite-modulus Sobczyk projection corridor without asserting full Hahn–Banach or unrestricted choice.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobczykTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Hahn–Banach, unrestricted extension choice, arbitrary complementability, quotient Banach-space equality, a selected global projection, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked separable-source visibility, c0 subspace visibility, extension-window visibility, projection-modulus exactness, boundedness, transport, replay, provenance, and local naming.

Chapter 8262

A Concrete Naming Certificate for $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$

The $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$ packet realizes the parallel-diamond and substitution-compatibility fork of chapter 9378 as finite source, branch, candidate-join, residual-boundary, and NameCert rows. It sits beside the typed parallel-diamond boundary of chapter 6433, the confluence residual checker of chapter 3827, the residual envelope of chapter 3921, the local-join ledger of chapter 4126, and the residual frontier of chapter 8254. The packet names the finite fork certificate only; it does not discharge full parallel diamond, residual substitution compatibility, arbitrary closed-star confluence, or Church–Rosser for every closed source.

Definition 8262.1 (MetaCIC confluence fork certificate carrier)**.** *A* $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, L, R, J, B, D, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the typed MetaCIC source row whose local peak is inspected by the audit map.*
2. *L and R are the left and right branch rows of the fork, recording the two visible* BetaParallel *or* BetaStarStep *directions under inspection.*
3. *J is the candidate-join row proposed for the two branches.*
4. *B is the residual-boundary row naming the substitution-compatibility premise still required when the candidate join is not closed by a shape-limited fragment.*
5. *D is the closed-fragment discharge row for normal, atom, and other already checked shapes.*
6. *A is the audit-map readback row tying S, L, R, J, B, D to the confluence fork named in chapter 9378.*
7. *H records componentwise* hsame *transport for source, branch, candidate-join, residual-boundary, discharge, audit, and local certificate rows.*
8. *C records displayed* Cont *replay from source through the two branches, candidate join, residual boundary, closed-fragment discharge, and audit readback.*
9. *P is the* Pkg *provenance row over* $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$*, the MetaCIC confluence audit map,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}}$ *row.*
The carrier has no coordinate for erasing the substitution boundary, promoting a conditional bridge to an unconditional theorem, or turning closed-fragment discharge into global Church–Rosser.

Theorem 8262.2 (MetaCIC confluence fork certificate NameCert obligations)**.** *For an accepted* $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$ *carrier* $F = (S, L, R, J, B, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetacicConfluenceFork}}$ *surface consists exactly of typed-source visibility, left-branch visibility, right-branch visibility, candidate-join visibility, residual-boundary visibility, closed-fragment discharge visibility, audit-map readback, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 8262.1.*

The certificate instructs future $\mathsf{Nontrivial\ MetacicConfluenceForkCertificate}^{\uparrow}$ *instances to separate a fork retaining B from a fork claiming full discharge. It instructs future* $\mathsf{FieldFaithful\ MetacicConfluenceForkCertifica}$ *instances to preserve $S, L, R, J, B, D, A, H, C, P, N$ as distinct rows, especially separating the candidate join J from the residual-boundary row B and the closed-fragment row D.*

Proof. Project the carrier. The confluence-facing rows are S, L, R, J, B, D, A, and the support rows are H, C, P, N.

Read the source and branch rows first. They display the local peak under inspection and keep the two fork branches visible before any join is cited.

Read the candidate join, residual boundary, and closed-fragment discharge next. The candidate row proposes the join, the residual-boundary row retains the substitution-compatibility premise when needed, and the discharge row records only shape-limited closed cases already available to the audit map.

Finally use the audit readback, transport, replay, provenance, and naming rows. They keep the fork certificate finite and source-traceable, but they do not erase B, promote a conditional bridge to an unconditional theorem, or convert D into arbitrary closed-source confluence. □

Falsifiable prediction: Any accepted MetaCIC confluence fork read that cites a candidate join without exposing the typed source, both branches, residual-boundary row, closed-fragment row, audit readback, transport, replay, provenance, and local naming refutes $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$.

Independence witness: metacic-confluence-audit-map, metacic-typed-parallel-diamond-boundary, metacic-confluence-residual-checker, metacic-confluence-residual-envelope, metacic-local-join-ledger: none is bijective with $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$ because the packet jointly stores source, two branches, candidate join, residual boundary, closed-fragment discharge, audit readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}$ packet is generated from a typed MetaCIC source row, two branch rows, a candidate-join row, a residual-substitution boundary row, a closed-fragment discharge row, audit-map readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8262.1, Definition 8262.2, the falsifiable prediction, and the independence witness seed the finite confluence-fork certificate for the MetaCIC audit-map boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicConfluenceForkCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full parallel diamond discharge, residual-substitution compatibility as a theorem, arbitrary closed-star confluence, global Church–Rosser, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked source visibility, branch visibility, candidate-join visibility, residual-boundary retention, closed-fragment discharge, audit readback, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming.

Chapter 8263

A Concrete Naming Certificate for $\mathsf{CompactModulusLedger}^{\uparrow}$

Following Bishop and Bridges on constructive compactness and uniform continuity, the $\mathsf{CompactModulusLedger}^{\uparrow}$ packet records the finite ledger by which compact finite-net data, pointwise continuity rows, and rational lower-bound folds hand one public modulus boundary to $\mathsf{UniformModulus}^{\uparrow}$. It sits over the compact metric source of chapter 129, the continuous-map row of chapter 130, the finite lower-bound fold of chapter 4096, the finite stream naming surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the compact-uniform-continuity ledger as displayed BEDC data; it does not import open-cover compactness, semantic continuity, countable choice, a selected infinite cover, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8263.1 (CompactModulusLedger carrier)**.** *A* $\mathsf{CompactModulusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, C, W, R, E, B, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row naming the compact source by finite centers, radii, and coverage checks.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-continuity row for the displayed map on the compact source.*
3. *C is the compact-net comparison row that attaches every inspected source point to a displayed finite-net center.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule through which the finite centers, source points, and target values are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for the pointwise target values and the candidate tolerance bounds.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row used only after the regular rational readbacks have been displayed.*
7. *B is the* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *row folding the finitely many local positive radii into one public lower-bound ledger.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that receives the folded lower bound together with the finite-net and continuity rows.*

9. *H records componentwise* hsame *transport for compact-net rows, pointwise continuity rows, stream windows, rational readbacks, real seals, finite folds, uniform-modulus handoffs, replay, provenance, and local naming.*
10. *T records displayed* Cont *replay for the same compact-modulus ledger.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactModulusLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, FiniteLowerBoundFold UniformModulus$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their inherited transports. It has no coordinate for open-cover compactness, semantic continuity, selected infinite subcovers, choice, quotient equality of functions, or ambient real completeness.

Theorem 8263.2 (CompactModulusLedger NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusLedger}^{\uparrow}$ *carrier*

$$L = (K, F, C, W, R, E, B, U, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusLedger}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure, pointwise continuous-map exposure, finite center comparison, stream-window access, regular rational readback, real-seal boundary, finite lower-bound folding,* UniformModulus$^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read the public compact-uniform-modulus boundary only after K, F, C, W, R, E, B, U have all been displayed.*

Proof. Project the carrier $L = (K, F, C, W, R, E, B, U, H, T, P, N)$. The analytic rows are exactly the compact finite-net row, the pointwise continuous-map row, the center comparison row, the stream window, the rational readback, the real seal, the finite fold, and the uniform-modulus consumer row.

The finite fold B can be read only after the finite family of local radii has been exposed through K, F, C, W, R, E. It supplies one public lower-bound ledger, not an open-cover argument or a choice-selected subcover.

The structural rows H, T, P, N transport, replay, source, and locally name the same finite packet. Since the carrier has no coordinate for semantic continuity, selected infinite covers, function quotient equality, countable choice, or ambient real completeness, no local NameCert obligation can export those endpoints. □

Theorem 8263.3 (CompactModulusLedger uniform handoff)**.** *Every public* UniformModulus$^{\uparrow}$ *read of an accepted* $\mathsf{CompactModulusLedger}^{\uparrow}$ *carrier factors through the compact finite-net row K, the pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *row F, the center comparison row C, the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *row B. The handoff exports the finite compact-uniform-continuity ledger U and no open-cover compactness, countable-choice schedule, host semantic-continuity theorem, selected infinite subcover, or ambient completeness principle.*

Proof. Start with the compact row K and the pointwise row F. They expose only finite centers, displayed source requests, and pointwise target estimates, so the route begins with compact finite-net data rather than an open cover.

Next use C, W, R, E to compare an inspected source point with a finite center and to read the target estimates through regular rational data before the terminal real seal is consumed. These rows are the only scalar-facing access to the continuity estimates.

Finally apply the finite lower-bound fold B and repack the result as the uniform-modulus row U. The transport, replay, provenance, and local-name rows preserve the same displayed ledger, so the handoff reaches $\mathsf{UniformModulus}^{\uparrow}$ without adding choice, semantic continuity, selected infinite covers, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusLedger}^{\uparrow}$ packet is generated from compact finite-net data, pointwise continuous-map rows, center comparisons, finite stream windows, regular rational readbacks, real seals, a finite lower-bound fold, a uniform-modulus consumer row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8263.1, Definition 8263.2, and Definition 8263.3 seed the finite compact-uniform-continuity ledger for downstream $\mathsf{UniformModulus}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, semantic continuity, selected infinite subcovers, countable choice, quotient equality of functions, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, pointwise-continuity exposure, finite-fold exactness, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusL

Chapter 8264

A Concrete Naming Certificate for $\mathsf{JordanVariation}^{\uparrow}$

Following Bishop and Bridges on functions of bounded variation and the classical Jordan treatment of positive and negative variation, the $\mathsf{JordanVariation}^{\uparrow}$ packet records signed variation as finite NameCert data over displayed partition ledgers. It sits after the bounded-variation surface of chapter 398, the cumulative total-variation route of chapter 2257, the variation-measure ledger of chapter 3593, the uniform partition row of chapter 2033, and the terminal real-name seal of chapter 38. The chapter names only finite positive and negative variation rows attached to displayed partitions; it does not import a signed measure, a quotient of functions, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8264.1 (Jordan variation carrier)**.** *A* $\mathsf{JordanVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (F, V^+, V^-, T, L, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded-variation source row, read through* $\mathsf{BoundedVariation}^{\uparrow}$ *and the displayed real-valued endpoint names.*
2. *V^+ is the positive-variation row, listing finite positive edge contributions on the displayed partition ledger.*
3. *V^- is the negative-variation row, listing finite negative edge contributions on the same partition ledger.*
4. *T is the total-variation readback row, recording the equality of the displayed total ledger with the finite sum of V^+ and V^- rows.*
5. *L is the partition ledger, recording the uniform-partition source, refinement tags, endpoint order, and finite edge names used by all three variation rows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint values, edge readback, and total-variation comparison.*
7. *H records componentwise* hsame *transport for the source, positive row, negative row, total-variation row, partition ledger, real seal, provenance, and local naming.*
8. *C records displayed* Cont *replay from carrier admission through positive variation, negative variation, total readback, partition transport, real-seal readback, provenance, and local naming.*

9. *P is the* Pkg *provenance over* BoundedVariation$^{\uparrow}$, TotalVariationFunction$^{\uparrow}$, VariationMeasure$^{\uparrow}$, UniformPartition$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{JordanVariation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host signed measure, a quotient equality class of functions, selected representatives of real values, an unlisted partition family, or an ambient completeness principle.

Theorem 8264.2 (Jordan variation NameCert obligations)**.** *For the carrier of Definition 8264.1, the local* NameCert$_{\text{JordanVariation}^{\uparrow}}$ *surface consists exactly of carrier admission for J, bounded-variation source admission through F, positive and negative variation admission through V^+ and V^-, total-variation readback through T, partition-ledger exposure through L, real-seal readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Every accepted consumer reads signed variation from this finite surface before it may pass the packet to regulated, Riemann–Stieltjes, or variation-measure routes.

Proof. Carrier admission is projection from the finite packet $J = (F, V^+, V^-, T, L, R, H, C, P, N)$. The bounded-variation source row F and the partition ledger L are present before either signed row is inspected.

The split exactness is then read from V^+, V^-, and T. The positive and negative rows record finite edge contributions on the same displayed ledger, while T records their total-variation readback through the TotalVariationFunction$^{\uparrow}$ and Real$^{\uparrow}$ routes.

The partition ledger cannot escape the packet. Refinements, endpoint order, edge names, and uniform-partition source are recorded in L and transported only through H and C; no consumer receives an unlisted partition family or a host supremum.

Finally read P and N. Provenance ties the carrier to the bounded-variation, total-variation, variation-measure, uniform-partition, and real-name siblings, and the local NameCert row names exactly that finite surface. Since no coordinate stores a signed measure, quotient equality, selected representative, or ambient completeness principle, each such read is refused at the carrier boundary. □

Falsifiable prediction: If a JordanVariationUp packet exports a positive or negative variation row without the common partition ledger L and real seal R, the carrier is refuted.

Independence witness: boundedvariation, total-variation-function, variation-measure, uniform-partition, and real: none is bijective to JordanVariation$^{\uparrow}$ because this packet stores the signed positive row, signed negative row, total-variation readback, partition ledger, Real seal, transport, replay, provenance, and local NameCert rows together as one finite surface.

Closure status for NameCert$_{\text{JordanVariation}^{\uparrow}}$.

Constructive story: A JordanVariation$^{\uparrow}$ packet is generated by displaying a bounded-variation source, positive variation row, negative variation row, total-variation readback, partition ledger, Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert row in one finite BHist carrier.

Theory closure: TheoryGate(NameCert$_{\text{JordanVariation}^{\uparrow}}$, seed)

Scope closed: Definition 8264.1 and Definition 8264.2 seed the finite signed-variation surface over BoundedVariation$^{\uparrow}$, TotalVariationFunction$^{\uparrow}$, VariationMeasure$^{\uparrow}$, UniformPartition$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{JordanVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host signed measures, quotient equality of functions, selected representatives, unlisted partition families, countable choice, bridge checking, Lean verification, or ambient completeness of Real$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, split exactness, total-variation readback, partition-ledger nonescape, transport, replay, provenance, and refusal rows for the displayed packet.

Chapter 8265

A Concrete Naming Certificate for $\mathsf{AscoliRealCompletion}^{\uparrow}$

Following Arzela, Ascoli, Bishop, and Bridges, the $\mathsf{AscoliRealCompletion}^{\uparrow}$ packet records the finite compact-continuity data that an Arzela–Ascoli style family hands to the L10 Real-and-Completion route. It sits after the finite family and compact-source surface of chapter 412, the compact-uniform-continuity packet of chapter 388, the regular rational readback of chapter 301, the finite stream-window source of chapter 126, and the terminal real-name seal of chapter 38. The packet names only displayed finite-net, equicontinuity, stream, regular-rational, Real, transport, replay, provenance, and local naming rows; it does not assert open-cover compactness, selected subsequences, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8265.1 (Ascoli real-completion carrier)**.** *An* $\mathsf{AscoliRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, K, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *family row, exposing the finite family, source compactness, and pointwise readback data consumed by the Ascoli route.*
2. *K is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row, recording the finite-net and modulus handoff for the same compact source.*
3. *U is the compact-continuity ledger that ties family windows, finite-net witnesses, and tolerance requests to one inspected source surface.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the compact family and modulus data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to those finite windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W and R are visible.*
7. *H records componentwise* hsame *transport for the family, compact-uniform, ledger, stream, regular, Real, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from compact family inspection through finite-net continuity, stream windows, regular readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{AscoliRealCompletion}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{AscoliRealCompletion}^{\uparrow}}$ *row.*
The classifier accepts only packets whose completion-facing read uses the displayed rows $F, K, U, W, R, E, H, C, P, N$ It has no coordinate for selected subsequences, open-cover compactness, quotient equality of functions, host completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8265.2 (Ascoli real-completion NameCert obligations)**.** *For every accepted* $\mathsf{AscoliRealCompletion}^{\uparrow}$ *carrier $A = (F, K, U, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AscoliRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, Arzela–Ascoli family exposure through F, compact-uniform-continuity exposure through K, compact-continuity ledger visibility through U, stream windows through W, regular rational readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8265.1. The tuple lists the family row, compact-uniform row, ledger row, stream windows, regular readback, Real seal, and structural rows as the whole accepted surface.

Read the analytic source through F, K, U. The family row supplies the Ascoli data, the compact-uniform row supplies the finite-net modulus route, and the ledger row keeps those rows tied to the same inspected compact source before any completion-facing read is available.

Finally route completion-facing output through W, R, E, H, C, P, N. The stream and regular rows expose finite windows and rational readback, the Real row seals the terminal value, and the structural rows transport, replay, source, and name the same packet. Since no selected subsequence, open-cover compactness, quotient equality, choice, host completeness, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 8265.3 (Ascoli real-completion modulus handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{AscoliRealCompletion}^{\uparrow}$ *packet receives its completion-facing data through the finite route*

$$\mathsf{ArzelaAscoli}^{\uparrow} \longmapsto \mathsf{CompactUniformContinuity}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes only the displayed family, compact-continuity, finite-window, regular-readback, Real-seal, transport, replay, provenance, and local naming rows. It exports no open-cover compactness theorem, no selected subsequence, no quotient equality of functions, no host completeness principle, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier surface. The rows F, K, U force the consumer to read the Ascoli family, compact-uniform-continuity packet, and compact-continuity ledger before it reaches any Real-facing object.

Next consume the completion-facing rows W, R, E. The finite stream windows expose only inspected name fragments, the regular rational row supplies the displayed readback, and the Real row seals the terminal value after those rows are visible.

The structural rows H, C, P, N then preserve the same route under transport, replay, provenance, and local naming. Because the packet contains no selected subsequence, open-cover compactness, quotient function equality, host completeness, choice schedule, or ambient Real-completeness coordinate, no such endpoint can be read from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliRealCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{AscoliRealCompletion}^{\uparrow}$ packet is generated from ArzelaAscoliUp family data, CompactUniformContinuityUp finite-net data, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{AscoliRealCompletion}\uparrow}$, seed)

Scope closed: Definition 8265.1, Definition 8265.2, and Definition 8265.3 seed the finite Ascoli-to-Real completion route through family, compact-uniform, stream, regular-rational, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{AscoliRealCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subsequences, quotient function equality, countable choice, host completeness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Ascoli family exposure, compact-uniform-continuity routing, stream-window exposure, regular-rational readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.AscoliRealCompletionUp.

Chapter 8266

A Concrete Naming Certificate for $\mathsf{LocatedMonotoneSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ packet records the finite L10 route by which a bounded monotone sequence is read through a located-supremum handoff rather than through ambient completeness. It sits after the monotone convergence packet of chapter 3928, the bounded monotone packet of chapter 7987, the located-supremum packet of chapter 1427, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the displayed route from monotone source data to located supremum data; it does not assert arbitrary $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient stream equality, or countable choice.

Definition 8266.1 (Located monotone sequence carrier)**.** *A* $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, B, S, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *source row, exposing the monotone sequence schedule and its convergence-facing finite route.*
2. *B is the accepted* $\mathsf{BoundedMonotoneSequence}^{\uparrow}$ *row, exposing the same sequence through displayed upper-bound and lower-bound evidence.*
3. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, carrying upper-bound and lower-approximation data for the source sequence.*
4. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *precision-window row through which sequence observations and located-supremum requests are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windowed observations and the located-supremum approximation ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic precision requests, comparison radii, and tail thresholds.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, S, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the source, boundedness, located-supremum, window, readback, tolerance, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a located monotone sequence request through $M, B, S, W, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedMonotoneSequence}^{\uparrow}$, $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$, BoundedMonotoneSe LocatedSupremum$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMonotoneSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through

$$M, B, S, W, R, D, E, H, C, P, N.$$

It has no coordinate for ambient real completeness, a selected limit, quotient stream equality, trichotomy, countable choice, or a host-level monotone convergence theorem.

Theorem 8266.2 (Located monotone sequence NameCert obligations)**.** *For every accepted* LocatedMonotoneSeque *carrier*

$$L = (M, B, S, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission, monotone convergence source exposure, bounded monotone source exposure, located-supremum handoff, shared finite-window exposure, regular rational readback, dyadic tolerance replay, terminal Real-seal ordering, componentwise transport, displayed replay, provenance, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 8266.1. Carrier admission exposes exactly $M, B, S, W, R, D, E, H, C, P, N$, so the only source-facing rows are the monotone convergence source M and the bounded monotone source B.

Read the order and approximation rows next. The located-supremum handoff S is the only row that can expose upper-bound and lower-approximation data, while W, R, D provide the shared finite windows, regular rational readback, and dyadic tolerance accounting.

The Real-facing seal is downstream of these rows. The structural rows H, C, P, N can only transport, replay, source, and name the same displayed packet, so the local NameCert surface has no coordinate for a selected limit, quotient equality, countable choice, or ambient completeness. □

Theorem 8266.3 (Located monotone sequence located-supremum handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ *carrier factors through the bounded monotone source rows M, B, then through the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row S, the shared* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. Thus the L10 monotone-sequence-to-located-supremum route is a displayed finite handoff, not an appeal to arbitrary* $\mathsf{Real}^{\uparrow}$ *completeness or a choice-selected limit.*

Proof. Start with an accepted packet $L = (M, B, S, W, R, D, E, H, C, P, N)$. By Definition 8266.1, the terminal real row is E, and it appears only after the source, boundedness, located-supremum, window, readback, and tolerance rows have been displayed.

Apply Definition 8266.2. It forces the NameCert surface to expose the monotone convergence source M and bounded monotone source B before the located-supremum handoff S can feed the readback rows W, R, D.

The rows H, C, P, N preserve this route under transport, replay, provenance, and naming. Since the carrier contains no ambient completeness, selected-limit, quotient-equality, or countable-choice coordinate, every Real-facing read remains a finite located-supremum handoff. □

Falsifiable prediction: Every accepted $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ Real-facing read displays a monotone convergence source, a bounded monotone source, a located-supremum handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any consumer may inspect it.

Independence witness: monotone-convergence-sequence, bounded-monotone-sequence, located-supremum, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded monotone source and a located-supremum handoff before the Real-facing seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMonotoneSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8266.1, Definition 8266.2, and Definition 8266.3 seed the finite route from bounded monotone sequence data to a located-supremum Real-facing read.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient stream equality, countable choice, a selected ambient limit, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, source exposure, bounded monotone exposure, located-supremum handoff, RegSeqRat readback, dyadic tolerance replay, and Real-seal non-escape for BEDC.Derived.LocatedMonotoneSeque

Constructive story: A $\mathsf{LocatedMonotoneSequence}^{\uparrow}$ packet is generated from a MonotoneConvergenceSequence source, a BoundedMonotoneSequence source, a LocatedSupremum handoff, a StreamName precision window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8267

A Concrete Naming Certificate for $\mathsf{PreuniformBase}^{\uparrow}$

Following Bourbaki and Willard, the $\mathsf{PreuniformBase}^{\uparrow}$ packet names the finite entourage-base data that can feed uniformity and completion consumers before a full uniformity object has been closed. It sits below the filter route of chapter 8257, the regular-Cauchy uniformity route of chapter 1628, and the Real-specific entourage basis of chapter 2060. The packet is a displayed preuniform basis surface; it is not a quotient uniformity, not a selected completion, not host equality, and not an ambient Real completeness principle.

Definition 8267.1 (Preuniform-base carrier)**.** *A* $\mathsf{PreuniformBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, D, S, M, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the entourage-base row whose entries name only displayed binary-neighborhood requests.*
2. *D is the diagonal witness row, recording that each admitted base entourage contains the displayed diagonal request.*
3. *S is the symmetry witness row, sending each admitted base request to a displayed symmetric refinement.*
4. *M is the finite-meet and refinement witness row, displaying a common smaller base request for each accepted finite pair.*
5. *R is the consumer route to* $\mathsf{EntourageFilter}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, *and* $\mathsf{RealUniformEntourageBase}^{\uparrow}$.
6. *H records componentwise* hsame *transport for the base row, diagonal row, symmetry row, finite-meet row, consumer route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a downstream entourage request through E, D, S, M, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{PreuniformBase}^{\uparrow}$, $\mathsf{EntourageFilter}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, $\mathsf{RealUniformEntourageBase}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PreuniformBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, S, M, R, H, C, P, N$. It has no coordinate for a completed uniformity, quotient filter, selected limit, host equality, countable-choice selector, or ambient Real completeness.

Theorem 8267.2 (Preuniform-base NameCert obligations)**.** *For every accepted* $\mathsf{PreuniformBase}^{\uparrow}$ *carrier*

$$B = (E, D, S, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PreuniformBase}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed entourage-base row* E*, diagonal exposure through* D*, symmetry exposure through* S*, finite-meet and refinement exposure through* M*, downstream consumer routing through* R*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, local naming through* N*, and non-escape for completed uniformities, quotient filters, selected limits, host equality, countable-choice selectors, and ambient Real completeness.*

Proof. Project the carrier of Definition 8267.1. The accepted packet exposes exactly one entourage-base row, one diagonal row, one symmetry row, one finite-meet refinement row, one consumer route, and four structural rows.

The basis obligations are local to E, D, S, M. The diagonal row supplies the displayed diagonal containment, the symmetry row supplies a displayed symmetric refinement, and the meet row supplies a displayed common refinement for each finite pair before any downstream filter or uniformity consumer is opened.

The consumer route is downstream. The row R permits $\mathsf{EntourageFilter}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, and $\mathsf{RealUniformEntourageBase}^{\uparrow}$ readers to consume the same basis data, while H, C, P, N transport, replay, source, and name that data without adding a completed uniformity or a quotient filter.

Since the carrier has no coordinate for selected limits, host equality, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8267.3 (Preuniform-base downstream handoff)**.** *Every* $\mathsf{EntourageFilter}^{\uparrow}$*,* $\mathsf{RegularCauchyUniformity}^{\uparrow}$*, or* $\mathsf{RealUniformEntourageBase}^{\uparrow}$ *consumer of an accepted* $\mathsf{PreuniformBase}^{\uparrow}$ *carrier must first read the displayed route*

$$E \longmapsto D, S, M \longmapsto R \longmapsto H, C, P, N.$$

The handoff exports a preuniform basis only as finite entourage-base data, not as a completed uniformity, quotient filter, selected limit, host equality, countable-choice selector, or ambient Real completeness theorem.

Proof. Start with Definition 8267.2. It places every accepted consumer read on the displayed base row E and then on the diagonal, symmetry, and finite-meet rows D, S, M.

Only after those rows are visible may the route R be read. The route records the three downstream consumers named in the statement, and each of them receives the same finite basis surface rather than a private completion object.

The remaining rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. Because the carrier lacks completed-uniformity, quotient-filter, selected-limit, host-equality, choice, and Real-completeness coordinates, the downstream handoff cannot export any of those endpoints. □

Falsifiable prediction: A downstream uniformity or completion consumer is refuted if it reads a PreuniformBase route without displaying the entourage-base row, diagonal row, symmetry row, finite-meet refinement row, consumer route, componentwise transport, continuation replay, package provenance, and local NameCert row.

Independence witness: entourage-filter, regular-cauchy-uniformity, real-uniform-entourage-base: no listed sibling supplies the whole $\mathsf{PreuniformBase}^{\uparrow}$ packet because those chapters consume or

specialize basis data, while this packet keeps the preuniform diagonal, symmetry, finite-meet, and downstream route rows as one carrier-local certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{PreuniformBase}^{\uparrow}}$.

Constructive story: A $\mathsf{PreuniformBase}^{\uparrow}$ packet is generated from displayed entourage-base rows, diagonal witnesses, symmetry witnesses, finite-meet refinement witnesses, a consumer route to EntourageFilter, RegularCauchyUniformity, and RealUniformEntourageBase rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PreuniformBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8267.1, Definition 8267.2, and Definition 8267.3 seed the finite preuniform basis surface below the existing uniformity and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PreuniformBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full uniformities, quotient filters, selected limits, host equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal exposure, symmetry exposure, finite-meet refinement, consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.PreuniformBaseUp.

Chapter 8268

A Concrete Naming Certificate for $\mathsf{BanachContractionModulus}^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachContractionModulus}^{\uparrow}$ packet isolates the finite contraction-budget ledger before any endpoint or fixed-point equality is read. It sits after the contraction-map surface of chapter 3529, the constructive contraction modulus of chapter 7631, the Cauchy orbit seal of chapter 7047, the complete-metric endpoint boundary of chapter 128, the Cauchy-modulus route of chapter 292, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the downstream fixed-point corridor of chapter 8031. The packet names the modulus surface only; it is not a completed fixed point, uniqueness theorem, selected limit, quotient Cauchy equality, host map equality, countable-choice schedule, or ambient completeness claim.

Definition 8268.1 (Banach contraction modulus carrier)**.** *A* $\mathsf{BanachContractionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, K, O, C, Q, R, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BanachContraction}^{\uparrow}$ *source row, exposing the self-map, contraction constant, and one-step distance comparison.*
2. *K is the* $\mathsf{ConstructiveContractionModulus}^{\uparrow}$ *row, storing the finite budget that turns a target tolerance into an iteration-depth request.*
3. *O is the* $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$ *row that reads the Picard-orbit residuals without selecting a completed endpoint.*
4. *C is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row generated from the contraction budget and orbit residuals.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the resulting regular Cauchy schedule.*
6. *R is the* $\mathsf{CompleteMetric}^{\uparrow}$ *admission row that permits completion-facing consumers to inspect the same Cauchy data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available after the regular rational readback has been displayed.*
8. *H records componentwise* hsame *transport for the contraction, budget, orbit, Cauchy, completion, Real, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay along the same route.*

10. *P is the* Pkg *provenance row over* $\mathsf{BanachContractionModulus}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{ConstructiveContractionMod}$ $\mathsf{CauchyFixedPointOrbitSeal}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachContractionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and inherited transports. It has no coordinate for a fixed point, endpoint equality, representative selection, quotient completion equality, host map equality, or ambient completeness.

Theorem 8268.2 (Banach contraction modulus NameCert obligations)**.** *For every accepted* $\mathsf{BanachContractionMod}$ *carrier*

$$M = (B, K, O, C, Q, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BanachContractionModulus}^{\uparrow}}$ *surface consists exactly of contraction-source visibility, finite contraction-budget visibility, Picard-orbit residual visibility, Cauchy-modulus extraction,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CompleteMetric}^{\uparrow}$ *admission, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The analytic rows are B, K, O, C, Q, R, E, and the structural rows are H, T, P, N.

Read B, K, O first. The contraction source supplies the self-map and contraction bound, the modulus row turns tolerances into finite iteration requests, and the orbit seal records the displayed residuals without selecting an endpoint.

Read C, Q, R, E next. The Cauchy modulus is extracted from the contraction budget, the regular rational readback packages the schedule, the complete-metric row admits that schedule as endpoint-facing input, and the Real seal is downstream of the regular readback.

The transport, replay, provenance, and naming rows preserve only this displayed route. Since no excluded endpoint or choice coordinate appears in the carrier, the local NameCert cannot export one. □

Theorem 8268.3 (Banach contraction modulus complete-metric handoff)**.** *Every complete-metric or fixed-point consumer of an accepted* $\mathsf{BanachContractionModulus}^{\uparrow}$ *packet must first pass through the finite route*

$$\mathsf{BanachContraction}^{\uparrow} \longmapsto \mathsf{ConstructiveContractionModulus}^{\uparrow} \longmapsto \mathsf{CauchyFixedPointOrbitSeal}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto$$

The route exports the contraction budget and Cauchy handoff needed by chapter 8031; it does not export a fixed point, selected limit, quotient stream equality, host equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the contraction and budget rows. They determine only finite iteration-depth requests and residual comparisons, so the consumer receives a displayed modulus rather than an endpoint.

Apply the obligation theorem. It places the orbit seal, Cauchy-modulus row, regular rational readback, complete-metric admission, and Real seal in one route before any fixed-point corridor may inspect the data.

The $\mathsf{LipschitzFixedPoint}^{\uparrow}$ consumer may therefore read a named contraction-budget handoff. It must supply any endpoint existence, uniqueness, or equality claim in its own packet, because none of those rows occurs in the modulus carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachContractionModulus}^{\uparrow}$ packet is generated from a BanachContraction source row, a ConstructiveContractionModulus budget row, a CauchyFixedPointOrbitSeal row, a CauchyModulus row, a RegSeqRat readback, a CompleteMetric admission boundary, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContractionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8268.1, Definition 8268.2, and Definition 8268.3 seed the finite contraction-modulus route before fixed-point endpoint reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachContractionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a fixed point, uniqueness, selected limit, quotient Cauchy equality, host map equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-source visibility, finite budget extraction, orbit-seal visibility, Cauchy-modulus readback, CompleteMetric handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BanachContractionModulusUp.

Chapter 8269

A Concrete Naming Certificate for $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$

Following the Bishop–Bridges finite-net proof of Heine–Cantor in chapter 9237, the FiniteEpsilonNetUniformModulus packet records the finite ε-net ledger that turns pointwise continuity rows on a compact metric source into a tolerance-only uniform-modulus handoff. It sits over the compact-metric carrier of chapter 129, the finite-net carrier of chapter 1767, the pointwise graph route of chapter 130, the uniform-modulus consumer of chapter 369, the metric comparison source of chapter 57, the rational tolerance rows of chapter 37, and the terminal real-name seal of chapter 38. The packet names the finite net-to-modulus handoff; it is not open-cover compactness, a semantic continuity theorem, a selected infinite subcover, quotient function equality, countable choice, or ambient real completeness.

Definition 8269.1 (Finite epsilon-net uniform-modulus carrier)**.** *A* $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, G, L, Q, R, E, V, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric carrier by finite coverage data.*
2. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, storing the bundled centers, radius rows, and coverage witnesses.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row for the displayed map.*
4. *L is the local radius ledger that attaches each bundled center to a positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row.*
5. *Q is the folded lower-bound row over the finite center bundle.*
6. *R is the* $\mathsf{Met}^{\uparrow}$ *comparison row used to move from a source point to a nearby center and then to a second source point.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for target-distance readback after rational comparisons have been displayed.*
8. *V is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row receiving the folded tolerance-only precision ledger.*
9. *H records componentwise* hsame *transport for compact, net, continuous-map, rational, metric, Real, uniform-modulus, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay for the same finite-net uniform-modulus route.*
11. *P is the* Pkg *provenance row over* $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and inherited transports. It has no coordinate for an open cover, semantic continuity, an infinite subcover selector, quotient function equality, countable choice, or ambient completeness.

Theorem 8269.2 (Finite epsilon-net uniform-modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$ *carrier*

$$U = (K, F, G, L, Q, R, E, V, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}}$ *surface consists exactly of compact-metric source visibility, finite epsilon-net visibility, continuous-map pointwise-modulus visibility, rational local-radius exposure, finite lower-bound folding, metric comparison, Real seal readback,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The mathematical rows are K, F, G, L, Q, R, E, V, and the structural rows are H, T, P, N.

Read K, F, G, L first. These rows expose a compact metric source, a finite bundle of centers, the pointwise continuity data at those centers, and the rational local-radius ledger.

Read Q, R, E, V next. The finite fold gives one tolerance-only precision row, the metric comparison row moves through the chosen center, the Real seal reads target-distance estimates, and the uniform-modulus row receives the folded ledger.

The structural rows preserve only this displayed route. Since no open-cover object, semantic function equality, infinite selector, choice schedule, or ambient completeness witness occurs in the carrier, the local NameCert cannot export such an endpoint. □

Theorem 8269.3 (Finite epsilon-net uniform-modulus compact handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The handoff exports a finite tolerance-only uniform-modulus ledger. It does not export open-cover compactness, semantic continuity, selected infinite subcovers, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the compact-metric and finite-net rows. They provide a finite bundle of centers and coverage witnesses, so the route begins with displayed finite data rather than an arbitrary open cover.

Apply the continuous-map and rational-radius rows at each center. The finite lower-bound fold then produces one precision ledger that is below all local radii in the bundle.

Finally use the metric comparison and Real seal rows to replay the usual two-triangle estimate through the chosen center before the uniform-modulus row is consumed. The packet therefore realizes the compact-uniform-continuity handoff as finite BEDC data and leaves all excluded global endpoints outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$ packet is generated from compact-metric data, finite epsilon-net centers and coverage witnesses, continuous-map pointwise rows, rational local-radius rows, a finite lower-bound fold, metric comparison, Real sealing, a UniformModulus consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8269.1, Definition 8269.2, and Definition 8269.3 seed the finite epsilon-net route from compact pointwise continuity to a tolerance-only uniform-modulus ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, semantic continuity, selected infinite subcovers, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric visibility, finite-net coverage, continuous-map pointwise rows, rational fold exactness, metric comparison, Real seal nonescape, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteEpsilonNetUniformModulusUp.

Chapter 8270

A Concrete Naming Certificate for $\mathsf{FanTheoremModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges constructive uniform-continuity analysis, the $\mathsf{FanTheoremModulus}^{\uparrow}$ packet reads a displayed $\mathsf{FanTheorem}^{\uparrow}$ finite bound as a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff. It sits after the finite fan certificate of chapter 400, the Bishop fan-modulus route of chapter 1707, the bar-modulus route of chapter 6151, the compact-uniform fan packet of chapter 7954, the downstream uniform-modulus surface of chapter 369, the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet records the theorem-level handoff from already displayed fan-bound data to a uniform modulus; it does not reprove the bar theorem, select an infinite branch, invoke open-cover compactness, choose a host modulus, quotient stream names, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8270.1 (Fan theorem modulus carrier)**.** *A* $\mathsf{FanTheoremModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, B, T, U, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bound row, exposing a barred binary-spread depth already admitted by the fan-theorem packet.*
2. *B is the* $\mathsf{BishopFanModulus}^{\uparrow}$ *schedule row, turning the displayed fan depth into finite observation windows for Real and Completion consumers.*
3. *T is the* $\mathsf{FanTheoremBarModulus}^{\uparrow}$ *row, recording the bar-coverage-to-tolerance extraction already available at the bar-modulus surface.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, read only after F, B, T have displayed the finite bound and its tolerance interpretation.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected prefixes and tolerance requests.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the displayed windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare requested precision with the fan-bound schedule.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular, and dyadic rows have been exposed.*

9. *H records componentwise* hsame *transport for F, B, T, U, S, Q, D, R and their structural support.*
10. *C records displayed* Cont *replay from the fan-bound row through the Bishop schedule, bar-modulus row, tolerance ledger, and uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FanTheoremModulus}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BishopFanModulus}^{\uparrow}$, $\mathsf{FanTheoremBarModulus}^{\uparrow}$, $\mathsf{CompactUniformFanModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FanTheoremModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets whose uniform-modulus read is routed through the displayed rows $F, B, T, U, S, Q, D, R, H, C, P, N$. It has no coordinate for a selected infinite branch, open-cover compactness theorem, host modulus function, quotient stream equality, countable-choice schedule, or ambient real completeness.

Theorem 8270.2 (Fan theorem modulus NameCert obligations)**.** *For every accepted* $\mathsf{FanTheoremModulus}^{\uparrow}$ *carrier $M = (F, B, T, U, S, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FanTheoremModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows; fan-bound exposure through F; Bishop fan-modulus scheduling through B; bar-modulus extraction through T; tolerance-only uniform-modulus handoff through U; finite-window, regular-rational, dyadic-tolerance, and Real-seal readback through S, Q, D, R; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8270.1. Its fan and modulus rows are exactly F, B, T, U, its readback rows are exactly S, Q, D, R, and its structural rows are exactly H, C, P, N.

Read F as the already displayed fan-theorem finite bound. The Bishop fan-modulus row B turns that bound into finite observation windows, and the bar-modulus row T records the tolerance extraction that is available before a uniform-modulus consumer reads U.

The rows S, Q, D, R supply only finite windows, regular rational readback, dyadic tolerance comparison, and terminal Real sealing. The structural rows then transport, replay, source, and name the same packet. Since no additional coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations and block branch selection, open-cover compactness, host modulus selection, quotient stream equality, countable choice, and ambient Real completeness. □

Theorem 8270.3 (Fan theorem modulus compact-uniform handoff)**.** *Every compact-uniform or Real-and-Completion consumer of an accepted* $\mathsf{FanTheoremModulus}^{\uparrow}$ *packet receives its uniform-modulus data through the finite route*

$$\mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{BishopFanModulus}^{\uparrow} \longmapsto \mathsf{FanTheoremBarModulus}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{R}$$

When the consumer is the compact-uniform fan route, the same handoff may be cited beside $\mathsf{CompactUniformFanModulus}^{\uparrow}$*, but only as the displayed fan-bound-to-uniform-modulus package F, B, T, U, S, Q, D, R. It exports no open-cover compactness theorem, selected branch, selected cover center, external modulus oracle, quotient stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8270.2. They force every accepted read to expose the fan-bound row, Bishop schedule row, bar-modulus row, uniform-modulus handoff, readback rows, and structural rows in the displayed carrier.

The compact-uniform fan surface may consume this packet only after the fan-bound-to-uniform-modulus route has already been displayed. Thus $\mathsf{CompactUniformFanModulus}^{\uparrow}$ is a sibling consumer of the same finite handoff, not a source of an additional branch selector or host compactness object.

Finally inspect the excluded coordinates of Definition 8270.1. The carrier has no slot for branch selection, open-cover compactness, selected centers, external moduli, quotient stream equality, countable choice, or ambient Real completeness, so none of those objects can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanTheoremModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FanTheoremModulus}^{\uparrow}$ packet is generated from a FanTheoremUp finite-bound row, BishopFanModulusUp schedule row, FanTheoremBarModulusUp extraction row, UniformModulusUp handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparison, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanTheoremModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8270.1, Definition 8270.2, and Definition 8270.3 seed the finite fan-bound-to-uniform-modulus handoff through fan, Bishop schedule, bar-modulus, uniform-modulus, stream, regular-rational, dyadic, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanTheoremModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, branch selection, selected cover centers, external modulus oracles, quotient stream equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, fan-bound exposure, Bishop schedule routing, bar-modulus extraction, UniformModulus handoff, finite-window readback, regular-rational readback, dyadic tolerance comparison, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.FanTheoremModulusUp.

Chapter 8271

A Concrete Naming Certificate for LebesgueCriterionRiemannIntegrability$^{\uparrow}$

Following Riemann, Lebesgue, and Bishop–Bridges constructive analysis, the LebesgueCriterionRiemannIntegrability$^{\uparrow}$ packet records the finite discontinuity-ledger route by which a bounded real function on a displayed compact interval is admitted to the Riemann integrability boundary. It sits beside the finite discontinuity packet of chapter 7684, the Darboux oscillation criterion of chapter 7252, the Darboux finite-sum surface of chapter 1880, and the Riemann integral readback of chapter 1526. The packet names only the finite BEDC rows needed to route discontinuity evidence into Darboux gap control; it does not import a measure-zero theorem, a countable exceptional set, quotient function equality, selected partitions, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8271.1 (Lebesgue criterion for Riemann integrability carrier)**.** *A* LebesgueCriterionRiemannIntegrability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, F, O, D, J, U, V, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* Real$^{\uparrow}$ *compact-interval row naming the endpoints and orientation on which the bounded-function read is made.*
2. *F is the bounded-function observation row, exposing only the finite windows requested by the criterion.*
3. *O is the local oscillation row that records the displayed oscillation tests used to detect discontinuity candidates.*
4. *D is the finite discontinuity ledger row, listing the displayed discontinuity names and their interval windows.*
5. *J is the Darboux cell-routing row, separating the cells assigned to D from the continuity cells read through O.*
6. *U and V are the upper and lower* DarbouxSum$^{\uparrow}$ *rows over the displayed cell route.*
7. *G is the finite Darboux gap row comparing U and V against a dyadic tolerance.*
8. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed gap comparison.*
9. *E is the* RiemannIntegral$^{\uparrow}$ *handoff row, visible only after $I, F, O, D, J, U, V, G, R$ have been displayed.*

10. *H records componentwise* hsame *transport for the interval, function, oscillation, ledger, Darboux, gap, readback, handoff, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the interval and bounded-function request through the discontinuity ledger, Darboux comparison, regular readback, and Riemann handoff.*

12. *P is the* Pkg *provenance over* $\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}$, $\mathsf{RiemannIntegrabilityDiscontinuity}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}$ *row.*

The carrier has no coordinate for measure completion, null-set equality, countable exceptional families, selected refinement sequences, quotient functions, host integration, or ambient real completeness.

Theorem 8271.2 (Lebesgue criterion NameCert obligations)**.** *For every accepted* LebesgueCriterionRiemannIntegral *carrier*

$$L = (I, F, O, D, J, U, V, G, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval, bounded-function observation, oscillation test, discontinuity ledger, Darboux routing, Darboux sums, gap, regular readback, Riemann handoff, transport, replay, provenance, and local naming rows;*

2. *discontinuity-ledger exposure, requiring every discontinuity candidate consumed by the criterion to appear in the finite ledger D with a displayed interval window;*

3. *off-ledger continuity routing, requiring J to send every non-ledger cell through the oscillation row O before any Darboux comparison is read;*

4. *Darboux gap control, requiring U, V, G to compare upper and lower finite sums only over the cells exposed by D, J;*

5. *Riemann handoff nonescape, requiring E to be read only after the finite discontinuity, Darboux, and regular readback rows are visible.*

Thus the packet supplies a finite BEDC surface for the Lebesgue-style discontinuity criterion without exporting measure-zero data or an ambient completeness principle.

Proof. Project the carrier of Definition 8271.1. The analytic rows are exactly $I, F, O, D, J, U, V, G, R, E$; the remaining rows H, C, P, N transport, replay, source, and name that same packet.

The discontinuity side is read first. The ledger D lists only displayed discontinuity windows, while J records how the remaining cells are routed through the oscillation row O. No countable exceptional family or measure object is present in this route.

The Darboux side then reads the same cell route. The upper and lower rows U, V compute finite sums over the displayed cells, and the gap row G compares them before the regular readback R is consumed.

Finally the Riemann handoff E is downstream of those rows. Since the carrier omits measure completion, null-set equality, selected refinement limits, quotient function equality, countable choice, host integration, and ambient completeness, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}$ surface. □

Theorem 8271.3 (Lebesgue criterion Darboux discontinuity route)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *or* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer of an accepted* $\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}$ *carrier must read the finite route*

$$I, F, O, D, J \longmapsto U, V \longmapsto G \longmapsto R \longmapsto E.$$

Consequently the criterion exposes Riemann integrability only through displayed discontinuity ledgers, Darboux finite sums, and regular readback; it does not expose a measure-zero certificate, a countable discontinuity set, quotient function equality, selected partitions, countable choice, host integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8271.2. They force any criterion consumer to display the discontinuity ledger D, the off-ledger oscillation route O, and the cell-routing row J before a Darboux comparison is available.

The Darboux rows are then fixed by the displayed route. Rows U and V compute upper and lower finite sums on the same cells, while G records the finite upper-minus-lower gap.

The regular readback row R exposes the gap before the Riemann handoff E is consumed. The structural rows H, C, P, N preserve the same packet and add no measure, quotient, choice, host-integration, or completeness endpoint. □

Falsifiable prediction: A purported $\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}$ read that reaches a $\mathsf{RiemannIntegral}^{\uparrow}$ handoff without displaying the interval, bounded-function observation, oscillation, finite discontinuity ledger, Darboux cell route, upper and lower Darboux rows, finite gap, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 8271.1.

Independence witness: riemann-integrability-discontinuity, darboux-oscillation-criterion, darboux-sum, riemannintegral: $\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}$ is not bijective with any listed sibling because it joins oscillation-ledger exposure, finite discontinuity routing, Darboux gap control, regular readback, Riemann handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}$ packet is generated from a Real interval-domain row, bounded-function observations, oscillation tests, a finite discontinuity ledger, Darboux cell routing, upper and lower Darboux rows, a finite gap row, RegSeqRat readback, RiemannIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8271.1, Definition 8271.2, and Definition 8271.3 seed the finite discontinuity-ledger route from Lebesgue-style criterion data to Darboux gap control and Riemann handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueCriterionRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-zero equivalence, countable exceptional families, null-set equality, quotient function equality, selected refinement sequences, host integration, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, discontinuity-ledger exposure, off-ledger continuity routing, Darboux gap control, Riemann handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueCriterionRiemannIntegrabilityUp.

Chapter 8272

A Concrete Naming Certificate for $\mathsf{UniformContinuityFanModulus}^{\uparrow}$

Following Brouwer and Bishop fan-theorem constructive analysis, the $\mathsf{UniformContinuityFanModulus}^{\uparrow}$ packet records the finite bar route by which pointwise continuity on a compact metric source yields one displayed uniform-modulus row. It sits over the compact finite-net data of chapter 129, the continuous-map pointwise rows of chapter 130, the rational arithmetic of chapter 37, the terminal real-name seal of chapter 38, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names finite fan-bar inspection data only; it does not import open-cover compactness, a selected infinite branch, semantic continuity, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8272.1 (Uniform-continuity fan-modulus carrier)**.** *A* $\mathsf{UniformContinuityFanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, A, B, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, exposing the inspected centers and cover radii.*
2. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-continuity row for the map on that compact source.*
3. *A is the finite fan-bar row whose nodes record the inspected source windows and their local continuity radii.*
4. *B is the finite* ProbeBundle *induction ledger that certifies every accepted source request meets a bar node.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the finitely many positive local radii.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row used only after the rational fold and target-value readbacks are displayed.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that receives the folded radius as the tolerance-only modulus boundary.*
8. *H records componentwise* hsame *transport for compact-net, continuous-map, fan-bar, bundle, rational-fold, Real-seal, uniform-modulus, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from compact finite-net inspection through fan-bar induction and rational lower-bound folding to the uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformContinuityFanModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityFanModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite branch selector, open-cover compactness, host equality of real functions, countable choice, quotient real equality, semantic continuity, or ambient completeness.

Theorem 8272.2 (Uniform-continuity fan-modulus NameCert obligations)**.** *For every accepted* $\mathsf{UniformContinuityFanModulus}^{\uparrow}$ *carrier* $F = (K, M, A, B, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityFanMo}}$ *surface consists exactly of compact finite-net exposure, continuous-map pointwise exposure, finite fan-bar admission,* ProbeBundle *induction, rational lower-bound folding, Real-seal readback,* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read the uniform modulus only after* K, M, A, B, R, E *have all been displayed.*

Proof. Project the carrier of Definition 8272.1. The tuple lists the compact metric source, pointwise continuous-map row, fan-bar row, bundle-induction ledger, rational fold, Real seal, uniform-modulus output, and structural support rows as the whole accepted surface.

The finite fan-bar stage is A, B. The bar row lists the inspected nodes and local radii, while the ProbeBundle induction ledger proves that every accepted source request reaches one of those nodes. This is a finite induction over displayed bundle entries, not a selected infinite branch.

The modulus stage is R, E, U. The rational row folds the displayed positive radii to one lower-bound row, the Real row seals the scalar readback only after that fold is visible, and the uniform-modulus row receives exactly that finite boundary. The rows H, C, P, N transport, replay, source, and name the same packet, so the local ledger cannot export open-cover compactness, semantic continuity, choice, quotient equality, or ambient completeness. □

Theorem 8272.3 (Uniform-continuity fan-modulus fan-bar route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformContinuityFanModulus}^{\uparrow}$ *packet factors through the finite fan-bar route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto A, B \longmapsto \mathsf{Rat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

Thus the consumer reads compact finite-net coverage, pointwise continuity rows, finite fan-bar nodes, ProbeBundle *induction, rational lower-bound folds, the terminal Real seal, componentwise transport, replay, provenance, and the local NameCert ledger before exporting any modulus data.*

Proof. Start with K and M. The compact metric row supplies the finite source inspection surface, and the continuous-map row supplies only pointwise local estimates attached to that same source.

Next read A and B. The fan-bar row stores the finite bar nodes, and the bundle induction ledger certifies coverage of the inspected source requests. Transport by H and replay by C keep these rows componentwise stable under the accepted classifier.

Finally fold through R, seal through E, and hand the result to U. The provenance row P and local name row N record that the exported modulus is the displayed finite route. Since no coordinate stores an infinite branch, open-cover compactness, selected representative, countable choice, host real equality, or ambient completeness, none of those objects can be read by the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuityFanModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformContinuityFanModulus}^{\uparrow}$ packet is generated from CompactMetricUp finite-net data, ContinuousMapUp pointwise rows, a finite fan-bar row, a ProbeBundle induction ledger, RatUp lower-bound folds, a

RealUp seal, a UniformModulusUp consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{UniformContinuityFanModulus}\uparrow}$, seed)

Scope closed: Definition 8272.1, Definition 8272.2, and Definition 8272.3 seed the finite fan-bar route from compact finite-net and pointwise-continuity rows to a UniformModulus consumer boundary.

Formal verification: FormalStatus(NameCert$_{\text{UniformContinuityFanModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite branch selection, semantic continuity, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous-map pointwise exposure, fan-bar coverage, ProbeBundle induction, rational lower-bound fold exactness, RealUp sealing, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformContinuityFanModulusUp.

Chapter 8273

A Concrete Naming Certificate for $\mathsf{UniformCover}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and uniform covers, the $\mathsf{UniformCover}^{\uparrow}$ packet records the finite cover surface that an L10 uniform-space consumer may read before passing to star refinements or completion. It sits after the uniform-space surface of chapter 339 and the entourage-base surface of chapter 1819, before the star-refinement surface of chapter 2167 and the uniform-completion boundary of chapter 1473. The packet names only displayed cover, entourage, membership, refinement, transport, replay, provenance, and local naming rows; it does not assert open-cover compactness, selected subcovers, paracompact refinement, ambient completeness, or a host powerset cover principle.

Definition 8273.1 (UniformCover carrier)**.** *A* $\mathsf{UniformCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, E, C, M, R, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row naming the underlying uniform-space interface.*
2. *E is the* $\mathsf{EntourageBase}^{\uparrow}$ *row from which the cover is generated.*
3. *C is the finite cover row, listing the displayed members that a consumer may inspect.*
4. *M is the membership-window row, recording which named points or finite probes are admitted to each cover member.*
5. *R is the refinement-read row, exposing how an entourage-basis request may refine the cover row.*
6. *H records componentwise* hsame *transport for the source, entourage base, cover, membership, refinement, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay of the route*

$$S, E \longmapsto C, M, R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{UniformCover}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{EntourageBase}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCover}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $S, E, C, M, R, H, T, P, N$ and has no coordinate for compactness of all open covers, choice-selected subcovers, paracompactness, a star-refinement witness, or completion.

Theorem 8273.2 (UniformCover NameCert obligations)**.** *For an accepted* $\mathsf{UniformCover}^{\uparrow}$ *carrier*

$$U = (S, E, C, M, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCover}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the uniform-space source, entourage-base, cover, membership, refinement, transport, replay, provenance, and local naming rows;*
2. *cover generation, requiring C to be read from the displayed uniform-space and entourage-base rows S, E;*
3. *membership exposure, requiring every cover-member read to pass through the finite window row M;*
4. *refinement exposure, requiring every refinement-facing consumer to pass through R before citing an entourage-base refinement;*
5. *classifier transport, requiring H and T to preserve the same rows componentwise under* hsame *and displayed continuation;*
6. *non-escape, excluding open-cover compactness, selected subcovers, paracompact refinements, star-refinement witnesses, and completion claims.*

No NameCert obligation is stored outside $S, E, C, M, R, H, T, P, N$.

Proof. Project the carrier of Definition 8273.1. The accepted packet exposes exactly the uniform-space, entourage-base, cover, membership, refinement, transport, replay, provenance, and local naming rows.

The cover generation and membership clauses are row reads. The cover row C is downstream of S, E, while each member inspection is mediated by the finite membership-window row M.

The refinement clause is separate. A consumer may request an entourage-base refinement only through R, so the cover packet does not silently become a star-refinement or completion theorem.

Finally H, T, P, N preserve the same finite route by componentwise transport, replay, provenance, and local naming. Since the excluded host principles are not coordinates of the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8273.3 (UniformCover entourage refinement route)**.** *Let*

$$U = (S, E, C, M, R, H, T, P, N)$$

be an accepted $\mathsf{UniformCover}^{\uparrow}$ *carrier. Every consumer that treats the cover as generated by a uniform entourage factors through the finite route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{EntourageBase}^{\uparrow} \longmapsto C, M, R \longmapsto H, T, P, N.$$

The route may feed the star-refinement surface of chapter 2167 and the completion boundary of chapter 1473, but it exports no selected refinement, no compactness principle, and no ambient completion object.

Proof. Start with the accepted carrier. The source and entourage-base rows S, E are the only entries from which the cover row can be generated.

Apply Definition 8273.2. Its membership and refinement obligations force every cover read through C, M, R before any downstream refinement or completion consumer can cite the cover.

The star-refinement and completion consumers therefore receive only the displayed cover route. Transport, replay, provenance, and local naming preserve that route through H, T, P, N, and the absent compactness, selection, and completion coordinates cannot be projected from the packet. □

Falsifiable prediction: A uniform-space consumer that cites a cover without displaying the $\mathsf{UniformSpace}^{\uparrow}$ source row, the $\mathsf{EntourageBase}^{\uparrow}$ generator row, and the finite membership window is not reading a UniformCoverUp carrier.

Independence witness: uniformspace, entouragebase, starrefinementuniformcover, uniformcompletion: none is bijective because $\mathsf{UniformCover}^{\uparrow}$ stores the cover and membership route between the uniform-space source and later refinement or completion consumers. UniformSpaceUp supplies the uniformity but not the finite cover row; EntourageBaseUp supplies basis refinement but not cover membership; StarRefinementUniformCoverUp consumes a cover with an extra star-containment row; UniformCompletionUp consumes Cauchy and completion data rather than a bare uniform-cover carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCover}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCover}^{\uparrow}$ packet is generated from a UniformSpaceUp source row, an EntourageBaseUp row, finite cover and membership rows, an entourage-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8273.1, Definition 8273.2, and Definition 8273.3 seed the finite uniform-cover surface consumed by star-refinement and completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, compactness, selected subcovers, paracompactness, star-refinement containment, or ambient completion.

Upgrade path: Obligation closure requires checked carrier admission, cover generation, membership exposure, entourage refinement, classifier transport, and non-escape for BEDC.Derived.UniformCoverUp.

Chapter 8274

A Concrete Naming Certificate for $\mathsf{RadoSelectionLemma}^{\uparrow}$

Following Rado's selection lemma and the finite matroid/transversal tradition, the $\mathsf{RadoSelectionLemma}^{\uparrow}$ packet records the certificate surface by which a finite family of local choices is assembled into a global independent selection. It reads the finite-index rows of chapter 36 and chapter 28, the independence carrier of chapter 202, and the edge-selection boundary of chapter 234. The chapter names only the displayed finite selection packet and its NameCert obligations; it does not assert an infinite choice principle, a host transversal selector, quotient equality of selections, or an ambient maximality theorem.

Definition 8274.1 (Rado selection lemma carrier)**.** *A* $\mathsf{RadoSelectionLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, A, M, G, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index surface, presented by* $\mathsf{List}^{\uparrow}$ *rows and bounded by* $\mathsf{Nat}^{\uparrow}$ *row counters.*
2. *A assigns to each displayed index row a finite admissible choice list, with every choice row kept inside the same* BHist *carrier.*
3. *M is the* $\mathsf{Matroid}^{\uparrow}$ *independence row on the displayed ground, including the hereditary and exchange fields used by local selections.*
4. *G is the* $\mathsf{Matching}^{\uparrow}$ *incidence row recording the finite bipartite choice graph between index rows and admissible choice rows.*
5. *S is the selected transversal row, displaying one carried choice for each index row and recording that the selected choices are independent in M.*
6. *H records componentwise* hsame *transport for I, A, M, G, S.*
7. *C records displayed* Cont *replay of the finite route from the index rows and admissible choices through matroid independence and matching incidence to the selected transversal row.*
8. *P is the* Pkg *provenance over* $\mathsf{RadoSelectionLemma}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RadoSelectionLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows and their componentwise transport, replay, provenance, and local naming data.

Theorem 8274.2 (Rado selection lemma NameCert obligations)**.** *For every accepted* $\mathsf{RadoSelectionLemma}^{\uparrow}$ *carrier*

$$R = (I, A, M, G, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RadoSelectionLemma}^{\uparrow}}$ *surface consists exactly of the finite-index admission rows, admissible-choice rows, matroid independence rows, matching incidence rows, selected-transversal row, componentwise transport, displayed replay, provenance, and local naming row. No obligation row is stored outside the displayed packet.*

Proof. Project the carrier of Definition 8274.1. Its finite data are the index surface I, admissible choice rows A, matroid row M, matching row G, and selected transversal row S.

The NameCert admission clauses are rowwise. The finite-index part is read through $\mathsf{List}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$, the independence requirements are read through $\mathsf{Matroid}^{\uparrow}$, and the incidence constraints are read through $\mathsf{Matching}^{\uparrow}$ before the selected row S is available.

Transport and replay preserve the same surface. The H row moves each displayed component by hsame, the C row replays the same finite route, and P, N source and name the resulting packet. Since none of these rows contains an infinite choice principle, a host transversal selector, quotient equality of selections, or an ambient maximality theorem, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RadoSelectionLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{RadoSelectionLemma}^{\uparrow}$ packet is generated from finite index rows, admissible choice lists, a matroid independence row, a matching incidence row, a selected transversal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RadoSelectionLemma}^{\uparrow}}$, seed)

Scope closed: Definition 8274.1 and Definition 8274.2 seed the finite Rado selection NameCert surface over $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, and $\mathsf{Matching}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RadoSelectionLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite choice, host transversal selection, quotient equality of selections, ambient maximality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-index stability, matroid independence transport, matching incidence transport, selected-transversal exactness, and a Lean obligation surface for BEDC.Derived.RadoSelectionLemmaU

Chapter 8275

A Concrete Naming Certificate for $\mathsf{ProofTraceConfluence}^{\uparrow}$

The $\mathsf{ProofTraceConfluence}^{\uparrow}$ packet realizes the conditional MetaCIC confluence boundary of chapter 9378 as finite NameCert data. It sits beside the critical-path surface of chapter 765, the normalization frontier of chapter 892, the confluence audit packet of chapter 735, the candidate-normalization handoff of chapter 5547, and the subject-reduction discharge boundary of chapter 718. The packet names proof-trace confluence only as a displayed route through reduction traces, residual-substitution rows, closed-normal fragments, and conditional diamond inputs; it does not turn any retained premise into full closed confluence.

Definition 8275.1 (ProofTraceConfluence carrier)**.** *A* $\mathsf{ProofTraceConfluence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, R, N, D, C, B, F, H, E, P, L)$$

with the following displayed rows:

1. *T is the MetaCIC proof-trace row naming the two reduction traces or local peak being compared.*
2. *R is the residual-substitution row, retaining the substitution-compatibility input when a beta-residual branch requires it.*
3. *N is the closed-normal fragment row, recording the closed normal, sort, or impossible closed-variable cases that are already unconditional.*
4. *D is the conditional-diamond input row, naming either the substitution-compatibility input for a parallel diamond or the closed parallel-diamond input for closed star confluence.*
5. *C is the confluence-audit consumer row, read against the audit map and confluence audit packet.*
6. *B is the blocked-premise row, recording which residual-substitution, closed-diamond, subject-reduction, or normal-form consistency input remains visible.*
7. *F is the frontier route through* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *and* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$.
8. *H records componentwise* hsame *transport for trace, residual, closed-normal, diamond, audit, blocked-premise, frontier, replay, provenance, and local naming rows.*
9. *E records displayed* Cont *replay of the route*

$$T, R, N, D, C, B, F \longmapsto H, P, L.$$

10. *P is the* Pkg *provenance over* ProofTraceConfluence$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICConfluenceAuditPacket$^{\uparrow}$, SubjectReductionDischarge$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *L is the local* NameCert$_{\mathsf{ProofTraceConfluence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, N, D, C, B, F, H, E, P, L$. It has no coordinate for unconditional arbitrary closed Church–Rosser, unrestricted residual substitution, a hidden subject-reduction theorem, quotient syntax equality, evaluator soundness, or normal-form consistency detached from its seal.

Theorem 8275.2 (ProofTraceConfluence NameCert obligations)**.** *For an accepted* ProofTraceConfluence$^{\uparrow}$ *carrier*

$$Q = (T, R, N, D, C, B, F, H, E, P, L),$$

the local NameCert$_{\mathsf{ProofTraceConfluence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for proof traces, residual-substitution input, closed-normal fragments, conditional diamond input, audit consumer, blocked-premise, frontier, transport, replay, provenance, and local naming rows;*
2. *residual visibility, requiring every full parallel-diamond read to display the residual-substitution row R or record its absence in B;*
3. *closed-diamond visibility, requiring every closed star-confluence read to display the closed-diamond row D or record its absence in B;*
4. *closed-normal fragment separation, requiring N to remain a shape-limited unconditional fragment rather than a full confluence theorem;*
5. *frontier and audit handoff, requiring F and C to route every consumer through the critical-path, normalization-frontier, and confluence-audit packets;*
6. *non-escape, excluding unconditional closed confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, and unsealed normal-form consistency.*

No obligation is stored outside $T, R, N, D, C, B, F, H, E, P, L$.

Proof. Project the carrier of Definition 8275.1. The accepted packet exposes proof traces, residual-substitution data, closed-normal fragments, conditional-diamond inputs, audit rows, blocked premises, frontier rows, and structural rows.

The residual and closed-diamond clauses are premise-retention clauses. A full parallel-diamond route must display R, and a closed star-confluence route must display D; when either input is absent, the blocked-premise row B remains visible.

The closed-normal row N is shape-limited. It records the available normal, sort, and impossible closed-variable cases without adding a coordinate for arbitrary closed confluence.

Finally F, C, H, E, P, L preserve and name the same route through the critical-path and audit surfaces. Since the excluded global endpoints are not carrier coordinates, the displayed rows exhaust the local NameCert obligations. □

Theorem 8275.3 (ProofTraceConfluence conditional handoff)**.** *Every confluence-facing consumer of an accepted* ProofTraceConfluence$^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto T, R, N, D, C, B \longmapsto H, E, P, L.$$

If the consumer claims full parallel diamond, it must display the substitution-compatibility input in R. If it claims full closed star confluence, it must display the closed-diamond input in D. Otherwise the blocked-premise row B remains part of the exported handoff.

Proof. Start from the frontier row F. It places the proof-trace read after the critical-path and normalization-frontier surfaces, so a confluence-facing consumer receives a displayed route rather than a detached theorem.

Apply Definition 8275.2. The residual visibility and closed-diamond visibility obligations force R or D to be present for the corresponding full confluence claim, while B records any retained premise.

The closed-normal fragment row N may discharge only its shape-limited cases. Transport, replay, provenance, and local naming then preserve the same finite handoff through H, E, P, L, and no absent global endpoint can be cited as if it were a packet field. □

Falsifiable prediction: Any proof-trace confluence route that claims full closed confluence while omitting both the substitution-compatibility input and the closed-diamond input is rejected by the ProofTraceConfluenceUp classifier.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, subjectreductiondischarge, normalformconsistencyseal: none is bijective because $\mathsf{ProofTraceConfluence}^{\uparrow}$ stores proof traces, residual-substitution premises, closed-normal fragments, conditional-diamond inputs, audit consumers, blocked premises, frontier rows, transport, replay, provenance, and local NameCert rows in one carrier. MetaCICCriticalPathUp records the critical route but not the confluence proof trace; MetaCICNormalizationFrontierUp records candidate and endpoint boundaries without the residual-substitution handoff; SubjectReductionDischargeUp addresses typing preservation rather than confluence joins; NormalFormConsistencySealUp seals endpoint consistency without displaying the conditional confluence inputs.

The Lean follow-up must provide `Nontrivial ProofTraceConfluenceUp` and `FieldFaithful ProofTraceConfluenceUp` instances that preserve $T, R, N, D, C, B, F, H, E, P, L$ without discharging the retained confluence premises by definition.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTraceConfluence}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTraceConfluence}^{\uparrow}$ packet is generated from proof-trace rows, residual-substitution premises, closed-normal fragments, conditional-diamond inputs, confluence-audit consumers, blocked-premise rows, MetaCIC frontier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofTraceConfluence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8275.1, Definition 8275.2, and Definition 8275.3 seed the proof-trace confluence route while retaining substitution and closed-diamond premises.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofTraceConfluence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, full arbitrary closed confluence, unrestricted residual substitution, subject reduction, quotient syntax equality, evaluator soundness, or unsealed normal-form consistency.

Upgrade path: Obligation closure requires checked carrier admission, residual visibility, closed-diamond visibility, closed-normal fragment separation, frontier handoff, audit handoff, and non-escape for BEDC.Derived.ProofTraceConfluenceUp.

Chapter 8276

A Concrete Naming Certificate for $\mathsf{CousinIProblem}^{\uparrow}$

Following Cousin and Cartan, the $\mathsf{CousinIProblem}^{\uparrow}$ packet records the first Cousin problem as finite local principal divisor data over an analytic cover. It sits beside the holomorphic source of chapter 68, the complex analytic boundary of chapter 76, the sheaf gluing surface of chapter 151, the sheafification packet of chapter 7336, and the analytic-set certificate of chapter 8008. The packet names local meromorphic-principal rows, overlap cocycle rows, obstruction rows, and local readback rows without importing a host Stein theorem, global divisor selector, quotient sheaf cohomology class, or ambient choice principle.

Definition 8276.1 (Cousin I problem carrier)**.** *A* $\mathsf{CousinIProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, U, F, O, Z, Q, G, H, R, P, N)$$

with the following visible rows:

1. *X is the complex analytic source row, read through the* $\mathsf{CplxAnalytic}^{\uparrow}$ *and* $\mathsf{Holomorphic}^{\uparrow}$ *neighbourhood surfaces;*
2. *U is the finite cover row listing the inspected analytic opens and their overlap windows;*
3. *F is the local meromorphic-principal family row, recording one displayed principal datum on each member of U;*
4. *O is the overlap-difference row, recording the holomorphic comparison on each inspected pairwise overlap;*
5. *Z is the obstruction row, separating vanishing local coboundary data from a still-visible nonzero cocycle boundary;*
6. *Q is the sheaf and sheafification route row through which local sections and overlap comparisons are read;*
7. *G is the local principal readback row, available only for windows whose obstruction coordinate in Z is displayed as vanishing;*
8. *H records componentwise* hsame *transport for X, U, F, O, Z, Q, G and the local structural rows;*
9. *R records displayed* Cont *replay for cover restriction, overlap comparison, sheaf routing, and local readback;*

10. *P is the* Pkg *provenance over* $\mathsf{CousinIProblem}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{CplxAnalytic}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Sheafification}^{\uparrow}$, $\mathsf{AnalyticSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinIProblem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, U, F, O, Z, Q, G, H, R, P, N$. It has no coordinate for a selected global meromorphic function, host sheaf cohomology quotient, Stein vanishing theorem, unrestricted refinement choice, or ambient analytic continuation oracle.

Theorem 8276.2 (Cousin I problem NameCert obligations)**.** *For every accepted* $\mathsf{CousinIProblem}^{\uparrow}$ *carrier $K = (X, U, F, O, Z, Q, G, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CousinIProblem}^{\uparrow}}$ *surface consists exactly of carrier admission for the analytic source, finite cover, local principal family, overlap-difference, obstruction, sheaf-route, and local readback rows; overlap compatibility for O over the cover row U; obstruction persistence for Z; componentwise* hsame *transport through H; displayed* Cont *replay through R; provenance through P; and local naming through N.*

Consequently a consumer cannot replace the visible obstruction row by a host Cartan theorem, a hidden Stein-space hypothesis, a selected global divisor, quotient equality of cohomology classes, or countable-choice refinement data.

Proof. Project the carrier of Definition 8276.1. The rows $X, U, F, O, Z, Q, G, H, R, P, N$ are the complete local certificate surface, so admission and classifier comparison are componentwise reads from the finite packet.

Read the analytic and sheaf routes next. The holomorphic, complex analytic, sheaf, sheafification, and analytic-set sibling chapters supply the named source and local-section surfaces, while U, F, O, Q record only the inspected cover, local principal data, overlap comparisons, and sheaf route.

Finally inspect Z, G, H, R, P, N. The obstruction row is stored before any local principal readback is exported, and G is available only through that displayed boundary. Transport, replay, provenance, and naming preserve these rows rather than creating a global meromorphic selector or a host cohomology quotient. □

Theorem 8276.3 (Cousin I problem local principal readback)**.** *Every accepted* $\mathsf{CousinIProblem}^{\uparrow}$ *consumer that reads a local principal solution on an inspected analytic window must first display the route*

$$\mathsf{Holomorphic}^{\uparrow}, \quad \mathsf{CplxAnalytic}^{\uparrow}, \quad U, \quad F, \quad O, \quad Z, \quad Q$$

inside the carrier of Definition 8276.1. The readback row G may be consumed only after the obstruction coordinate in Z is displayed for that window, and it exports a local principal read rather than a global solution of the first Cousin problem.

Proof. Start with the NameCert obligations of Definition 8276.2. They force the consumer to expose the analytic source, cover row, local principal family, overlap-difference row, obstruction row, and sheaf route before local naming.

Use the sibling routes for the source data. Holomorphic and complex analytic rows provide the local analytic windows; the sheaf and sheafification rows provide the local-section and restriction discipline; and the analytic-set row records the divisor-facing support boundary.

The local readback row G is therefore downstream of Z and Q. Since the carrier contains no global selector, quotient cohomology class, Stein vanishing theorem, or refinement-choice coordinate, a consumer can read only the displayed local principal row and cannot infer a detached global Cousin solution. □

Falsifiable prediction: Every accepted $\mathsf{CousinIProblem}^{\uparrow}$ read exposes a complex analytic source, finite analytic cover, local meromorphic-principal family, holomorphic overlap-difference row, obstruction row, sheaf route, local principal readback row when available, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: holomorphic, complex-analytic, sheaf, sheafification, analytic-set: none is bijective to $\mathsf{CousinIProblem}^{\uparrow}$ because this packet jointly binds the analytic cover, local principal family, overlap-difference cocycle, obstruction row, sheaf route, conditional local readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CousinIProblem}^{\uparrow}}$.

Constructive story: A $\mathsf{CousinIProblem}^{\uparrow}$ packet is generated from complex analytic source rows, a finite analytic cover, local meromorphic-principal rows, holomorphic overlap-difference rows, an obstruction row, a sheaf route, local principal readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinIProblem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8276.1, Definition 8276.2, and Definition 8276.3 seed the finite Cousin I problem route over holomorphic, complex analytic, sheaf, sheafification, analytic-set, obstruction, local readback, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CousinIProblem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Stein vanishing, global meromorphic solution selection, host sheaf cohomology quotients, unrestricted analytic continuation, countable choice, Lean verification, or bridge checking for $\mathsf{CousinIProblem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, cover compatibility, overlap-difference exactness, obstruction persistence, local principal readback, sheaf-route provenance, transport, replay, and local naming for BEDC.Derived.CousinIProblemUp.

Chapter 8277

A Concrete Naming Certificate for $\mathsf{LocatedRealCauchyBoundary}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRealCauchyBoundary}^{\uparrow}$ packet records the finite boundary at which a located Cauchy presentation can be inspected before a $\mathsf{Real}^{\uparrow}$ readback is sealed. It sits beside the stream-name source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the located completion embedding of chapter 7471, and the finite-prefix stream certificate of chapter 7253. The packet names a boundary object for displayed finite means; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8277.1 (LocatedRealCauchyBoundary carrier)**.** *A* $\mathsf{LocatedRealCauchyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (\rho, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *ρ is the located-radius row, naming the dyadic tolerance radius and the finite comparison window used for the boundary read;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the finite stream window that supplies the observed approximants;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying the regular rational readback of the displayed stream window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row that records the dyadic arithmetic, radius comparison, and tolerance ledger;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, and D have been displayed;*
6. *H records componentwise* hsame *transport for the located-radius, stream, regular rational, dyadic, real-seal, package, and local naming rows;*
7. *C records the displayed* Cont *routes for finite-prefix inspection, located-radius tightening, regular-rational readback, dyadic comparison, and Real-facing seal handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *chapter 7471, chapter 7253,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyBoundary}^{\uparrow}}$ *row.*

Theorem 8277.2 (LocatedRealCauchyBoundary NameCert obligations)**.** *For every accepted* LocatedRealCauchyBo *carrier* $B = (\rho, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located-radius exposure through* ρ*, finite StreamName inspection through* S*, regular rational readback through* R*, dyadic tolerance accounting through* D*, Real seal readback through* E*, componentwise transport through* H*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*. It exports no quotient stream, selected limit, countable choice, host real equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 8277.1. The only rows available to a consumer are $\rho, S, R, D, E, H, C, P, N$, so carrier admission is finite packet admission rather than admission to a completed ambient real line.

Read the Cauchy-facing evidence next. The stream source is named by S, regular rational readback by R, and dyadic tolerance comparison by D. The located-radius row ρ binds the finite comparison window before the $\mathsf{Real}^{\uparrow}$ seal row E can be inspected.

Finally use the structural rows. The transport row H preserves the same displayed coordinates, the continuation row C replays finite-prefix and seal handoff routes, and P, N package and name the packet. Since no row stores quotient stream equality, selected limits, countable choice, host equality, or ambient completeness, the local NameCert surface is exactly the listed finite boundary. □

Theorem 8277.3 (LocatedRealCauchyBoundary Real handoff)**.** *Every Real-facing read of an accepted* $\mathsf{LocatedRealCauchyBoundary}^{\uparrow}$ *carrier must pass through the located-radius row* ρ*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the displayed structural rows* H, C, P, N*. The handoff may cite the located completion embedding and finite-prefix stream certificate, but it cannot replace the boundary by a quotient stream, a selected limit, a hidden representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-prefix side. The row S is the only stream-name coordinate, and the finite-prefix stream certificate supplies the discipline that a consumer reads displayed windows rather than an infinite stream quotient.

Next pass through the regular and dyadic rows. The row R carries the regular rational readback, while D records the dyadic tolerance comparison against the located-radius row ρ. The Bishop located completion embedding can therefore be cited only through those displayed rows.

The Real-facing seal is last. The row E is reached through the continuation row C after the preceding finite evidence has been exposed, and H, P, N preserve, package, and name the same packet. Thus the handoff exports a finite Real-readback boundary, not a selected completed point or an ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealCauchyBoundary}^{\uparrow}$ packet is generated from a located-radius row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealCauchyBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 8277.1, Definition 8277.2, and Definition 8277.3 seed the finite located Cauchy boundary and its Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealCauchyBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient Cauchy streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-radius exposure, finite-prefix inspection, regular rational readback, dyadic tolerance accounting, Real seal handoff, and non-escape for the displayed packet.

Chapter 8278

A Concrete Naming Certificate for $\mathsf{SequentialArchimedeanReal}^{\uparrow}$

Following Bishop and Bridges on real-number names and explicit Archimedean bounds, the SequentialArchimedeanR packet records the finite sequential request that turns a displayed $\mathsf{Real}^{\uparrow}$ name into a witnessed rational or dyadic bound for a requested observation index. It sits over the natural-number request surface of chapter 28, the rational and dyadic witness surfaces of chapter 37 and chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a sequential Archimedean handoff; it is not trichotomy, a Dedekind cut, quotient representative selection, countable choice, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 8278.1 (Sequential Archimedean real carrier)**.** *A* $\mathsf{SequentialArchimedeanReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, i, Q, D, W, G, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name whose sequential Archimedean bound is being requested.*
2. *i is the unary* $\mathsf{Nat}^{\uparrow}$ *index row naming the requested observation and bound scale.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *witness row exporting the rational bound visible at that request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *witness row presenting the same bound in the dyadic scale consumed by finite windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window opened at the displayed index and bound request.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after i, Q, D, W, G have been displayed.*
8. *H records componentwise* hsame *transport for the real name, index request, rational witness, dyadic witness, finite window, regular readback, Real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the unary request through the StreamName window, RegSeqRat readback, rational and dyadic witnesses, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SequentialArchimedeanReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanReal}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $R, i, Q, D, W, G, E, H, C, P, N$. It has no coordinate for trichotomy, Dedekind cuts, quotient representatives, countable-choice schedules, selected completed streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialArchimedeanRealUp.SequentialArchimedeanRealTasteGate single carrier alignment`

Theorem 8278.2 (Sequential Archimedean real NameCert obligations)**.** *For every accepted* SequentialArchimedean *carrier*

$$S = (R, i, Q, D, W, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real name, unary index request, rational witness, dyadic witness, finite window, regular readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows; request compatibility binding i to the same window W that produces G; rational-dyadic witness coherence binding Q and D to the same sequential bound; Real-seal ordering requiring E to be read only after i, Q, D, W, G; and non-escape excluding trichotomy, Dedekind cuts, quotient representatives, countable choice, selected completed streams, and ambient Real completeness.*

Proof. Project the carrier of Definition 8278.1. The rows $R, i, Q, D, W, G, E, H, C, P, N$ are the complete packet inventory, so carrier admission has no hidden coordinate.

The request compatibility clause follows from the order of the rows. The unary $\mathsf{Nat}^{\uparrow}$ request i opens the $\mathsf{StreamName}^{\uparrow}$ window W, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback G is extracted from that same finite window before any Real-facing seal is available.

The rational and dyadic witnesses are read together. The $\mathsf{Rat}^{\uparrow}$ row Q and $\mathsf{DyadicRatCore}^{\uparrow}$ row D describe one displayed sequential bound for R at the request i, transported componentwise through H and replayed through C.

Finally the Real seal and non-escape clauses use the absence of further coordinates. Since E is the only terminal $\mathsf{Real}^{\uparrow}$ seal and the packet contains no trichotomy branch, Dedekind cut, quotient representative, countable-choice schedule, selected completed stream, or completeness object, every accepted consumer remains inside the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{SequentialArchimedeanReal}^{\uparrow}$ Real-facing read exposes a RealUp name, a NatUp unary request, RatUp and DyadicRatCoreUp bound witnesses, a StreamNameUp finite window, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before using the sequential Archimedean handoff.

Independence witness: real, nat, rat, dyadicratcore, streamname, regseqrat: $\mathsf{SequentialArchimedeanReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a unary sequential request, rational and dyadic bound witnesses, finite stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming while refusing trichotomy, Dedekind cuts, quotient representatives, countable choice, selected completed streams, and Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialArchimedeanReal}^{\uparrow}$ packet is generated from a RealUp name, a NatUp unary request, RatUp and DyadicRatCoreUp witness rows, a StreamNameUp finite window, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialArchimedeanReal}^{\uparrow}}$, seed)

Scope closed: Definition 8278.1 and Definition 8278.2 seed the finite sequential Archimedean bound route over explicit index requests, rational and dyadic witnesses, finite windows, regular readbacks, and RealUp seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{SequentialArchimedeanReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, Dedekind cuts, quotient representatives, countable choice, selected completed streams, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request compatibility, rational-dyadic witness coherence, Real-seal ordering, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.SequentialArchimedeanRealUp.

Chapter 8279

A Concrete Naming Certificate for $\mathsf{RealOrderWindow}^{\uparrow}$

Following Bishop and Bridges constructive analysis, $\mathsf{RealOrderWindow}^{\uparrow}$ records a finite comparison-window certificate over chapter 38, chapter 126, chapter 301, chapter 291, and chapter 7489. It is a local order-read packet for a displayed $\mathsf{Real}^{\uparrow}$ name, not a trichotomy principle, quotient stream equality test, selected supremum, host real equality, or ambient completeness theorem.

Definition 8279.1 (Real order window carrier)**.** *A* $\mathsf{RealOrderWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, S, Q, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the sealed* $\mathsf{Real}^{\uparrow}$ *row whose order-facing read is being inspected.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the inspected comparison request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same visible window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic comparison radius and endpoint arithmetic.*
5. *L is the lower comparison row exposed through D, S, Q.*
6. *U is the upper comparison row exposed through the same D, S, Q route.*
7. *H records componentwise* hsame *transport for $X, S, Q, D, L, U, C, P, N$.*
8. *C records displayed* Cont *replay from the finite comparison request through the readback, tolerance, lower, upper, seal, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RealOrderWindow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealArchimedeanWindow}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealOrderWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, Q, D, L, U, H, C, P, N$. It has no coordinate for trichotomy, host real equality, quotient stream equality, selected suprema, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8279.2 (Real order window NameCert obligations)**.** *For every accepted* $\mathsf{RealOrderWindow}^{\uparrow}$ *carrier* $R = (X, S, Q, D, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealOrderWindow}^{\uparrow}}$ *surface consists exactly of carrier admission, sealed* $\mathsf{Real}^{\uparrow}$ *exposure through* X*, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness through* D*, lower and upper comparison exposure through* L, U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8279.1. The accepted packet exposes the real seal X, finite observation window S, regular readback Q, dyadic tolerance ledger D, and the two comparison rows L, U before any structural row is consumed.

The lower and upper rows are not independent order primitives. They are read only after D, S, Q have fixed the dyadic tolerance, inspected window, and regular rational readback for the same sealed real row.

The support rows H, C, P, N transport, replay, source, and name exactly that displayed route. Since the carrier contains no coordinate for trichotomy, host real equality, quotient stream equality, selected suprema, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8279.3 (Real order window Archimedean handoff)**.** *Every order-facing* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealOrderWindow}^{\uparrow}$ *packet must pass through the* $\mathsf{RealArchimedeanWindow}^{\uparrow}$ *surface of chapter 7489 before it uses the lower and upper comparison rows* L, U*. The exported order read is therefore the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealArchimedeanWindow}^{\uparrow} \longmapsto \mathsf{RealOrderWindow}^{\uparrow},$$

and it exports no trichotomy, host real equality, quotient stream equality, selected supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier rows $X, S, Q, D, L, U, H, C, P, N$. The rows L, U are downstream from the same dyadic tolerance ledger, finite stream window, and regular rational readback that expose the sealed real row X.

Now read the Archimedean window cited in P. Its bracketing surface supplies the finite lower-and-upper window through $\mathsf{RealArchimedeanWindow}^{\uparrow}$ before a consumer may interpret L, U as order-facing comparison rows.

The remaining rows H, C, P, N preserve the same handoff. Because neither the carrier nor the Archimedean handoff contains a coordinate for trichotomy, quotient stream equality, selected suprema, host real equality, or ambient completeness, the order consumer receives only the finite comparison-window certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOrderWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealOrderWindow}^{\uparrow}$ packet is generated from a sealed Real row, a finite StreamName window, RegSeqRat readback, a DyadicRatCore tolerance ledger, lower and upper comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealOrderWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8279.1, Definition 8279.2, and Definition 8279.3 seed the finite comparison-window surface for order-facing Real reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealOrderWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, host real equality, quotient stream equality, selected suprema, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, dyadic tolerance exactness, Real seal non-escape, Archimedean handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealOrderWindowUp.

Chapter 8280

A Concrete Naming Certificate for $\mathsf{ClarksonInequality}^{\uparrow}$

Following Clarkson's inequalities for Lp spaces and the Bishop–Bridges finite-norm reading, the $\mathsf{ClarksonInequality}^{\uparrow}$ packet records the displayed convexity inequality surface by which an L^p norm comparison produces a uniform-convexity route. It sits between the $\mathsf{LpSpace}^{\uparrow}$ carrier of chapter 8060, the modulus route of chapter 6689, the uniformly convex consumer of chapter 5831, and the terminal scalar comparisons of chapter 38. The packet names only finite norm-window rows, conjugate-exponent comparison rows, midpoint-drop rows, Real tolerance seals, transport, replay, provenance, and local naming data; it does not import host Banach-space equality, selected extremizers, measure-theoretic quotient equality, or a completed ambient Clarkson theorem.

Definition 8280.1 (Clarkson inequality carrier)**.** *A* $\mathsf{ClarksonInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L, N, E, M, D, R, T, Q, P, A)$$

with the following displayed rows:

1. *L is the inspected* $\mathsf{LpSpace}^{\uparrow}$ *row, including the finite norm windows for the two displayed vectors.*
2. *N is the norm-comparison row that reads $\|x+y\|$, $\|x-y\|$, $\|x\|$, and $\|y\|$ only through the local L^p windows.*
3. *E is the exponent ledger separating the $1 < p \leq 2$ and $2 \leq p < \infty$ Clarkson regimes as finite case data.*
4. *M is the midpoint-drop row that packages the inequality output as a modulus-facing convexity witness.*
5. *D is the* $\mathsf{ModulusOfConvexity}^{\uparrow}$ *handoff row that records the extracted tolerance budget.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the scalar comparisons and tolerance inequalities.*
7. *T records componentwise* hsame *transport for L, N, E, M, D, R.*
8. *Q records displayed* Cont *replay for the norm-window, exponent, midpoint, modulus, and Real-seal reads.*
9. *P is the* Pkg *provenance over* $\mathsf{ClarksonInequality}^{\uparrow}$, $\mathsf{LpSpace}^{\uparrow}$, $\mathsf{ModulusOfConvexity}^{\uparrow}$, $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *A is the local* $\mathsf{NameCert}_{\mathsf{ClarksonInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and has no coordinate for host norm equality, selected extremizers, quotient classes of functions, or an ambient Banach-space completion theorem.

Lean4 (statement) `BEDC.Derived.ClarksonInequalityUp`

Theorem 8280.2 (Clarkson inequality NameCert obligations)**.** *For every accepted* $\mathsf{ClarksonInequality}^{\uparrow}$ *carrier*

$$C = (L, N, E, M, D, R, T, Q, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{ClarksonInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the* L^p *row, norm-window comparison, exponent-regime ledger, midpoint-drop witness, modulus handoff, Real tolerance seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. No obligation row is stored outside the displayed packet.*

Proof. Project the carrier of Definition 8280.1. The rows $L, N, E, M, D, R, T, Q, P, A$ are the complete inspection surface, so admission and classifier comparison are componentwise reads from the finite packet.

Read the analytic content through the sibling rows. The $\mathsf{LpSpace}^{\uparrow}$ row supplies the inspected norm windows, E selects the finite Clarkson regime, N records the displayed norm comparisons, and M packages the resulting midpoint drop.

The modulus and scalar routes are downstream. Row D hands the drop to $\mathsf{ModulusOfConvexity}^{\uparrow}$, row R seals the scalar tolerance through $\mathsf{Real}^{\uparrow}$, and T, Q, P, A only transport, replay, source, and name this displayed route. Since none of the rows contains a host equality of functions, a selected extremizer, a quotient norm class, or an ambient completion theorem, the listed clauses exhaust the local obligations. □

Theorem 8280.3 (Clarkson inequality uniform-convexity route)**.** *Every* $\mathsf{ClarksonInequality}^{\uparrow}$ *packet that exports a uniform-convexity read must first expose the route*

$$\mathsf{LpSpace}^{\uparrow}, \quad \mathsf{ClarksonInequality}^{\uparrow}, \quad \mathsf{ModulusOfConvexity}^{\uparrow}, \quad \mathsf{UniformlyConvexSpace}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

inside the finite carrier of Definition 8280.1. The exported object is a modulus-facing convexity witness for the inspected L^p *windows, not a host-level proof of Banach-space uniform convexity.*

Proof. Start from the NameCert obligations of Definition 8280.2. They force the consumer to expose the L^p norm windows, exponent-regime row, midpoint-drop row, modulus handoff, and Real tolerance seal before local naming.

The uniformly convex read is therefore downstream of the displayed Clarkson packet. The $\mathsf{ModulusOfConvexity}^{\uparrow}$ row receives the finite drop witness, and the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ row consumes that witness only after the Real tolerance comparisons have been sealed.

Nothing in the carrier provides host norm equality, selected extremizers, quotient function equality, or an ambient completed Banach theorem. The route exports exactly the finite modulus-facing witness displayed by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClarksonInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClarksonInequality}^{\uparrow}$ packet is generated from an Lp-space row, norm-window comparison row, exponent-regime ledger, midpoint-drop witness, modulus handoff, Real tolerance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClarksonInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8280.1, Definition 8280.2, and Definition 8280.3 seed the finite Clarkson inequality route over $\mathsf{LpSpace}^{\uparrow}$, $\mathsf{ModulusOfConvexity}^{\uparrow}$, $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClarksonInequality}^{\uparrow}}$, formalTarget)
Lean target: `BEDC.Derived.ClarksonInequalityUp`
Bridge status: none
Not claimed: This chapter does not close host Banach-space equality, selected extremizers, quotient equality of functions, measure-theoretic completion, theorem checking, axiom-clean audit, or bridge checking for ClarksonInequality$^{\uparrow}$.
Upgrade path: Obligation closure requires checked carrier admission, norm-window exactness, exponent-regime stability, midpoint-drop extraction, modulus handoff, Real tolerance sealing, transport, replay, provenance, and local naming for BEDC.Derived.ClarksonInequalityUp.

Chapter 8281

A Concrete Naming Certificate for $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy construction may glue local modulus schedules only after their overlap requests, dyadic tolerance rows, and regular-rational readbacks are visible. The $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ packet records that finite semilocal modulus surface for L10 Real-and-Completion consumers. It sits between the minimum-modulus route of chapter 5544 and the gluing route of chapter 5899, while still reading the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal before any glued Cauchy route is exported. The packet names a displayed semilocal modulus ledger, not quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8281.1 (Semilocal regular Cauchy modulus carrier)**.** *A* $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, V, O, M, G, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *U and V are the two local regular-Cauchy modulus schedules whose overlap is being compared.*
2. *O is the semilocal overlap ledger, naming the finite region on which the two schedules are jointly visible.*
3. *M is the* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *row that displays the common request budget used on the overlap.*
4. *G is the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *source row waiting to consume the semilocal modulus budget.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the common request budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same displayed windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the local schedules, overlap row, and common request budget.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after U, V, O, M, G, W, R, D have been displayed.*
9. *H records componentwise* hsame *transport for the local schedules, overlap, common budget, gluing source, windows, readback, dyadic ledger, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a Real-and-Completion request through* $U, V, O, M, G, W, R, D, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyMinModulus}^{\uparrow}$, $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected common representative, countable choice over local neighborhoods, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8281.2 (Semilocal regular Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ *carrier*

$$S = (U, V, O, M, G, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the two local modulus schedules, semilocal overlap ledger, minimum-modulus request budget, gluing source row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise transport, displayed replay, package provenance, local naming, and non-escape from quotient stream equality, selected representatives, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8281.1. The local schedules are U, V, and the semilocal overlap row is O.

The common request budget is read through M. By the sibling minimum-modulus chapter chapter 5544, that row is a displayed finite schedule rather than a selected representative or choice functional.

The gluing-facing obligation is then G, W, R, D, E. The gluing row waits until the finite windows, regular readback, dyadic tolerance ledger, and terminal Real seal are visible, matching the L10 dependency route of chapter 5899.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier omits quotient stream equality, selected representatives, countable choice, host equality, and ambient completeness, the displayed rows exhaust the local NameCert surface. □

Theorem 8281.3 (Semilocal regular Cauchy modulus gluing scope)**.** *Every gluing consumer of an accepted* $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ *packet must read the finite route*

$$U, V, O \longmapsto M, D \longmapsto G, W, R \longmapsto E \longmapsto H, C, P, N.$$

Thus the semilocal modulus data may feed $\mathsf{RegularCauchyGluing}^{\uparrow}$ *only through the displayed overlap, minimum-modulus, dyadic tolerance, finite-window, regular-readback, and Real-seal rows. It does not export quotient stream equality, selected common representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $S = (U, V, O, M, G, W, R, D, E, H, C, P, N)$. The two local schedules and the semilocal overlap row are the only local inputs visible to a consumer.

Apply the obligation theorem above. It forces the common request budget to pass through M and the dyadic comparisons to pass through D before any gluing row is consumed.

The gluing read is then G, W, R, E, with the finite windows and regular-rational readback exposed before the terminal Real seal. The structural rows H, C, P, N keep this route attached to the same packet, so none of the excluded quotient, choice, host-equality, or completeness coordinates can be projected from the carrier. □

Falsifiable prediction: A Real-and-Completion consumer that cites a semilocal regular-Cauchy modulus for gluing without displaying $U, V, O, M, G, W, R, D, E, H, C, P, N$ is not reading a SemilocalRegularCauchy packet.

Independence witness: regularcauchyminmodulus, regularcauchygluing, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ because this packet jointly carries two local modulus schedules, a semilocal overlap ledger, a common request budget, a gluing source row, finite windows, regular readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}$ packet is generated from two local regular-Cauchy modulus schedules, a semilocal overlap ledger, a RegularCauchyMinModulus common request row, a RegularCauchyGluing source row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8281.1, Definition 8281.2, and Definition 8281.3 seed the finite semilocal modulus route consumed by regular-Cauchy gluing and Real-facing completion readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SemilocalRegularCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, quotient stream equality, selected representatives, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, semilocal overlap exactness, minimum-modulus compatibility, gluing-source routing, StreamName and RegSeqRat readback, dyadic tolerance comparison, Real sealing, transport, replay, provenance, and nonescape for BEDC.Derived.SemilocalRegularCauchyModulusUp.

Chapter 8282

A Concrete Naming Certificate for $\mathsf{CauchyBoundSelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, $\mathsf{CauchyBoundSelector}^{\uparrow}$ records the finite tail-bound selection surface needed by L10 completion routes without invoking countable choice or a host-side least-bound oracle. It sits over the stream, regular-rational, dyadic, real, and signed-interval packets of chapter 126, chapter 301, chapter 291, chapter 38, and chapter 8197. The packet exports an explicit bounded-tail window for later diagonal-limit and completeness arguments; it does not claim located suprema, quotient stream equality, a choice principle for all Cauchy predicates, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8282.1 (Cauchy bound selector carrier)**.** *A* $\mathsf{CauchyBoundSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, D, R, I, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose finite tail windows are inspected.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream window.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the dyadic radius, comparison budget, and endpoint arithmetic used by the tail request.*
4. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row reached by the regular-rational readback when the tail window is accepted.*
5. *I is the* $\mathsf{SignedDyadicInterval}^{\uparrow}$ *bound row recording the displayed lower and upper dyadic endpoints for the inspected tail.*
6. *W is the selected tail-window row, consisting of a finite threshold name and the finite set of inspected windows that justify the displayed bound.*
7. *E is the exactness row asserting that the same S, Q, D, R, I data support the selected window W.*
8. *H records componentwise* hsame *transport for $S, Q, D, R, I, W, E, C, P, N$.*
9. *C records displayed* Cont *replay from the tail-bound request through S, Q, D, R, I, W, E and then to the support rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyBoundSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SignedDyadicInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBoundSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, R, I, W, E, H, C, P, N$. It has no coordinate for a hidden choice function, least-index search over all predicates, quotient stream equality, located supremum, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyBoundSelectorUp.CauchyBoundSelectorTasteGate single carrier alignment`

Theorem 8282.2 (Cauchy bound selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBoundSelector}^{\uparrow}$ *carrier $B = (S, Q, D, R, I, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyBoundSelector}^{\uparrow}}$ *surface consists exactly of the following five public rows, each with structural transport, replay, provenance, and local naming attached:*

1. $\mathsf{StreamName}^{\uparrow}$ *window exposure through S;*
2. $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback through Q;*
3. $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint arithmetic through D;*
4. $\mathsf{Real}^{\uparrow}$ *seal exposure through R;*
5. $\mathsf{SignedDyadicInterval}^{\uparrow}$ *bounded-tail enclosure through I, together with selected window W and exactness row E.*

The packet therefore names an explicit bounded-tail selector for a displayed finite Cauchy route, not a global countable-choice principle or a host-side least-bound operator.

Lean4 ✓ `BEDC.Derived.CauchyBoundSelectorUp.CauchyBoundSelectorNamecertObligations`

Proof. Project the carrier of Definition 8282.1. The first five mathematical rows expose the stream name, regular rational readback, dyadic tolerance ledger, real seal, and signed dyadic interval before any selected window can be consumed.

The selected window W is not an independent oracle. It is accepted only together with the exactness row E, which binds the same S, Q, D, R, I data to the displayed finite threshold and inspected windows.

The support rows H, C, P, N transport, replay, source, and name exactly that route. Since no carrier coordinate stores a hidden choice function, a least-index search over all predicates, quotient stream equality, a located supremum, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8282.3 (Cauchy bound selector window stability)**.** *Let $B = (S, Q, D, R, I, W, E, H, C, P, N)$ and $B' = (S', Q', D', R', I', W', E', H', C', P', N')$ be accepted* $\mathsf{CauchyBoundSelector}^{\uparrow}$ *carriers. If the rows S, Q, D, R, I are componentwise* hsame *to S', Q', D', R', I' and both exactness rows read the same dyadic tolerance ledger, then the selected windows W and W' name the same bounded-tail enclosure up to componentwise* hsame*. Consequently a completion-facing consumer may transport the bounded-tail certificate across equal finite windows without acquiring a new choice principle.*

Proof. Read the two carriers through the NameCert obligations of Definition 8282.2. Both accepted packets expose the same stream, regular readback, dyadic tolerance ledger, real seal, and signed dyadic interval up to componentwise hsame before either selected window is available.

Use the exactness rows E and E'. Because they read the same dyadic tolerance ledger and the same displayed source rows, each selected threshold is justified by the same finite tail-bound request and the same signed dyadic enclosure.

The transport rows H, H' then carry this agreement to the selected-window coordinates, while C, C', P, P', N, N' replay and source only the displayed route. Thus the bounded-tail enclosure is stable under the stated finite-window transport, and the consumer receives no additional global selection principle. □

Lean4 ✓ `BEDC.Derived.CauchyBoundSelectorUp.CauchyBoundSelectorCarrier window stability`

Falsifiable prediction: Every accepted $\mathsf{CauchyBoundSelector}^{\uparrow}$ consumer that exports a bounded Cauchy tail must display the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SignedDyadicInterval}^{\uparrow}$, selected-window, exactness, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, signed-dyadic-interval: no listed sibling is bijective to $\mathsf{CauchyBoundSelector}^{\uparrow}$ because this packet jointly binds a stream window, regular readback, dyadic tolerance, real seal, signed interval bound, selected finite threshold, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBoundSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyBoundSelector}^{\uparrow}$ packet is generated from a StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, SignedDyadicInterval enclosure, selected finite threshold, exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBoundSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8282.1, Definition 8282.2, and Definition 8282.3 seed the explicit bounded-tail selection surface for L10 completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBoundSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, least-index search, quotient stream equality, located suprema, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName exposure, RegSeqRat readback, DyadicRatCore tolerance exactness, Real seal non-escape, SignedDyadicInterval bound exposure, selected-window stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyBoundSelectorUp.

Chapter 8283

A Concrete Naming Certificate for MetaCICConfluenceBudgetMatrix$^{\uparrow}$

The MetaCICConfluenceBudgetMatrix$^{\uparrow}$ packet realizes the budget-matrix frontier in section 9386.3 as finite NameCert data. It sits beside the residual-substitution boundary of chapter 2181, the confluence diamond ledger of chapter 6547, the residual local-diamond witness of chapter 8046, the residual-join budget of chapter 7378, and the parallel-diamond selector of chapter 7989. The packet records bounded residual-substitution, closed parallel-diamond, and arbitrary closed-star confluence frontiers in one matrix; it does not supply unconditional substitution compatibility, arbitrary closed-term Church–Rosser, global subject reduction, or kernel consistency.

Definition 8283.1 (MetaCIC confluence budget-matrix carrier)**.** *A* MetaCICConfluenceBudgetMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, R, D, S, J, B, O, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-window row naming the inspected MetaCIC source, context boundary, and closedness budget.*
2. *R is the residual-substitution frontier row recording the substitution compatibility data that remain visible.*
3. *D is the closed parallel-diamond frontier row recording the bounded diamond premise or retained diamond obstruction.*
4. *S is the closed star-confluence frontier row recording the finite multi-step budget attached to the same source window.*
5. *J is the local-join scope row listing the candidate joins that are admitted inside the displayed matrix.*
6. *B is the budget matrix row aligning residual-substitution, parallel-diamond, and closed-star budgets by finite row and column names.*
7. *O is the obstruction row retaining the boundary names not discharged by R, D, S, J, B.*
8. *L is the Lean TasteGate obligation row, naming the later need for* Nontrivial MetaCICConfluenceBudgetMatrix$^{\uparrow}$ *and* FieldFaithful MetaCICConfluenceBudgetMatrix$^{\uparrow}$ *instances before a checked TasteGate theorem can read the matrix as a single carrier.*

9. *H records componentwise* hsame *transport for $T, R, D, S, J, B, O, L, C, P, N$.*
10. *C records displayed* Cont *replay from a confluence-frontier request through T, R, D, S, J, B, O, L and then to the structural rows.*
11. *P is the* Pkg *provenance over* MetaCICConfluenceBudgetMatrix$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceDiamondLedger$^{\uparrow}$, MetaCICParallelDiamondSelector$^{\uparrow}$, MetacicResidualDiamondReadback$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{MetaCICConfluenceBudgetMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, D, S, J, B, O, L, H, C, P, N$. It has no coordinate for unconditional substitution compatibility, arbitrary closed-term Church–Rosser, a hidden confluence oracle, unbounded checker search, or kernel consistency.

Theorem 8283.2 (MetaCIC confluence budget-matrix NameCert obligations)**.** *For every accepted* MetaCICConfluenceBudgetMatrix$^{\uparrow}$ *carrier $M = (T, R, D, S, J, B, O, L, H, C, P, N)$, the local* NameCert$_{\text{MetaCICConfluenceBudgetMatrix}^{\uparrow}}$ *surface consists exactly of typed closed-window exposure through T, residual-substitution frontier exposure through R, closed parallel-diamond frontier exposure through D, closed star-confluence frontier exposure through S, local-join scope through J, finite budget-matrix alignment through B, obstruction retention through O, Lean TasteGate obligation naming through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8283.1. The source row T fixes the typed closed window before any confluence-facing frontier row can be read.

The rows R, D, S expose the three open confluence frontiers as visible data: residual-substitution compatibility, closed parallel diamond, and arbitrary closed-star confluence. The local-join row J and matrix row B align only finite candidate joins and finite row-column budgets for that same closed window.

The obstruction row O remains visible, so the matrix cannot erase a missing substitution-compatibility input or promote a bounded local join into arbitrary closed-term Church–Rosser. The TasteGate row L records that later Lean work must provide Nontrivial and FieldFaithful instances for this carrier before a checked gate can certify single-carrier alignment.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores an unconditional confluence theorem, hidden oracle, unbounded checker, or kernel-consistency endpoint, the listed rows exhaust the local NameCert obligations. □

Theorem 8283.3 (MetaCIC confluence budget-matrix local join scope)**.** *Every accepted* MetaCICConfluenceBudget *carrier that exports a local join from J must keep the typed closed-window row T, residual-substitution frontier R, closed parallel-diamond frontier D, closed star-confluence frontier S, budget matrix B, and obstruction row O visible on the same route. Thus a local join admitted by the matrix is scoped to the displayed finite budget and cannot be read as unconditional substitution compatibility, arbitrary closed-term Church–Rosser, or global subject reduction.*

Proof. Start with the carrier rows of Definition 8283.1. The local-join row J is displayed only together with the frontier rows R, D, S and the budget matrix B for the same typed closed window T.

Apply the NameCert obligations of Definition 8283.2. They force any confluence-facing read to expose the residual-substitution frontier, closed parallel-diamond frontier, closed star-confluence frontier, local-join scope, finite budget alignment, and obstruction retention before the structural rows can be consumed.

The obstruction row O prevents the matrix from hiding the conditional boundary around substitution compatibility. Transport through H, replay through C, provenance through P, and naming through N preserve only the displayed matrix route, so the exported local join remains scoped to the finite budget rather than becoming an unconditional confluence endpoint. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}$ consumer that exports a confluence-facing local join must display typed closed-window, residual-substitution frontier, closed parallel-diamond frontier, closed star-confluence frontier, finite budget-matrix, obstruction, TasteGate-obligation, transport, replay, provenance, and local NameCert rows.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-diamond-ledger, metacic-residual-local-diamond-witness, metacic-residual-join-budget, metacic-parallel-diamond-selector: no listed sibling is bijective to $\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}$ because this packet jointly carries the residual-substitution, closed parallel-diamond, closed star-confluence, local-join, finite matrix, obstruction-retention, TasteGate-obligation, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}$ packet is generated from a typed closed-window row, residual-substitution frontier, closed parallel-diamond frontier, closed star-confluence frontier, local-join scope, finite budget matrix, obstruction row, Lean TasteGate obligation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8283.1, Definition 8283.2, and Definition 8283.3 seed the finite budget-matrix frontier for MetaCIC confluence work.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unconditional substitution compatibility, arbitrary closed-term Church–Rosser, global subject reduction, unbounded checker search, Lean verification, bridge checking, or kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-substitution frontier exposure, closed parallel-diamond frontier exposure, closed star-confluence frontier exposure, local-join scope, budget-matrix alignment, obstruction retention, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICConfluenceBudgetMatrixUp.

Chapter 8284

A Concrete Naming Certificate for MetaCICCandidatePruningBudget$^{\uparrow}$

The MetaCICCandidatePruningBudget$^{\uparrow}$ packet realizes the audit-synthesis frontier of section 9385.6 as finite candidate-pruning data. It sits beside the residual-choice budget of chapter 8191, the candidate-normalization confluence handoff of chapter 5547, the residual diamond frontier of chapter 8111, and the normal-form consistency seal of chapter 740. The packet records bounded proof-search pruning as a displayed candidate ledger and checker budget; it does not assert global confluence, normalization, subject reduction, kernel consistency, or an unbounded proof-search oracle.

Definition 8284.1 (MetaCIC candidate-pruning budget carrier)**.** *A* MetaCICCandidatePruningBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, K, Q, R, C, O, H, J, P, N)$$

with the following displayed rows:

1. *T is the typed closed-window row naming the MetaCIC source fragment, context boundary, and closedness budget under inspection.*
2. *K is the candidate row listing the finite candidate proofs, terms, residual joins, or normal endpoints admitted for pruning.*
3. *Q is the pruning-budget row recording the finite fuel, rank, residual-depth, and branch-count limits used by the candidate search.*
4. *R is the residual-frontier row naming the remaining unpruned residual choices and the frontier routes shared with chapter 8191 and chapter 8111.*
5. *C is the bounded-checker row. It records the local checker that may inspect T, K, Q, R and reject a pruning read when the candidate row or checker budget is absent.*
6. *O is the obstruction row retaining the confluence, normalization, and subject-reduction boundaries not discharged by the bounded checker.*
7. *H records componentwise* hsame *transport for typed windows, candidates, pruning budget, residual frontier, bounded checker, obstruction, replay, provenance, and local naming rows.*
8. *J records displayed* Cont *replay from an audit-synthesis request through $T, K, Q, R, C, O, H, P, N$.*

9. *P is the* Pkg *provenance over* MetaCICCandidatePruningBudget$^{\uparrow}$, MetaCICResidualChoiceBudget$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$, NormalFormConsistencySe BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{MetaCICCandidatePruningBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, Q, R, C, O, H, J, P, N$. It has no coordinate for candidate-free pruning, unbounded checker search, global confluence, global normalization, subject reduction, or kernel consistency.

Theorem 8284.2 (MetaCIC candidate-pruning budget obligations)**.** *For every accepted* MetaCICCandidatePruning *carrier*

$$B = (T, K, Q, R, C, O, H, J, P, N),$$

the local NameCert$_{\text{MetaCICCandidatePruningBudget}^{\uparrow}}$ *surface consists exactly of typed closed-window exposure, candidate row exposure, pruning-budget exposure, residual-frontier exposure, bounded-checker exposure, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from candidate-free pruning, unbounded checker search, global confluence, global normalization, subject reduction, and kernel consistency.*

Proof. Project the carrier of Definition 8284.1. The typed window T fixes the inspected closed fragment before any proof-search pruning read can be accepted.

The pruning data are K, Q, R, C. The candidate row K lists the finite objects being pruned, the budget row Q bounds the search, the residual-frontier row R retains the unpruned boundary, and the bounded checker C is the only local checker allowed to inspect that display.

The obstruction row O remains visible. It prevents a bounded pruning result from becoming global confluence, normalization, subject reduction, or kernel consistency.

Finally H, J, P, N transport, replay, source, and name the same finite route. Since no excluded endpoint is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8284.3 (MetaCIC candidate-pruning bounded checker rejection)**.** *Every accepted* MetaCICCandidatePru *pruning read must expose both the candidate row K and the bounded-checker row C on the same route as T, Q, R, O, H, J, P, N. A read that supplies a pruning budget without the candidate row and bounded checker is outside the classifier.*

Proof. Start from an accepted carrier $B = (T, K, Q, R, C, O, H, J, P, N)$. The classifier of Definition 8284.1 compares accepted packets only through the full displayed inventory, including K and C.

Apply the obligation theorem above. It makes candidate exposure and bounded-checker exposure separate mandatory NameCert clauses before the residual-frontier and pruning-budget rows can be consumed.

Therefore a read with Q but without K and C is not an accepted packet. Transport through H, replay through J, provenance through P, and naming through N preserve the same displayed route and do not add a hidden candidate or checker. □

The Lean boundary for this carrier must provide explicit Nontrivial MetaCICCandidatePruningBudget$^{\uparrow}$ and FieldFaithful MetaCICCandidatePruningBudget$^{\uparrow}$ instances before a checked TasteGate theorem may read the displayed rows $T, K, Q, R, C, O, H, J, P, N$ as one carrier.

Falsifiable prediction: A MetaCIC pruning read that lacks either the candidate row K or the bounded-checker row C is rejected by the MetaCICCandidatePruningBudget$^{\uparrow}$ classifier even when it names a pruning budget row.

Independence witness: metacic-proof-search-frontier, metacic-residual-choice-budget, metacic-parallel-diamond-frontier, metacic-candidate-normalization-confluence-handoff, normalformconsistencyseal: none is bijective to $\mathsf{MetaCICCandidatePruningBudget}^{\uparrow}$ because this packet jointly carries typed windows, candidates, pruning budgets, residual frontiers, bounded checker, obstruction retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCandidatePruningBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCandidatePruningBudget}^{\uparrow}$ packet is generated from a typed closed-window row, candidate row, pruning-budget row, residual-frontier row, bounded-checker row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCandidatePruningBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8284.1, Definition 8284.2, and Definition 8284.3 seed the finite candidate-pruning budget route for the MetaCIC audit-synthesis frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCandidatePruningBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, normalization, subject reduction, kernel consistency, unbounded proof-search pruning, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate exposure, pruning-budget exactness, residual-frontier exposure, bounded-checker rejection, obstruction retention, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICCandidatePruningBudgetUp.

Chapter 8285

A Concrete Naming Certificate for $\mathsf{FredholmOperator}^{\uparrow}$

Following Fredholm and Riesz operator theory, the $\mathsf{FredholmOperator}^{\uparrow}$ packet records the finite BEDC carrier by which a displayed bounded linear operator is read together with finite kernel, cokernel-facing obstruction, and index ledger rows. It sits after the bounded-operator surface of chapter 1644, the compact-operator route of chapter 1952, the Banach and Hilbert endpoint carriers of chapter 90 and chapter 91, and the alternative-facing consumer of chapter 3425. The packet supplies the operator/index dependency read by downstream index arguments such as chapter 273; it does not import a host quotient cokernel, selected complement, ambient index equality, spectral decomposition, Lean verification, or bridge checking.

Definition 8285.1 (Fredholm operator carrier)**.** *A* $\mathsf{FredholmOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, B, K, E, C, I, A, H, R, P, N)$$

with the following displayed rows:

1. *X is the source endpoint row, read through a displayed* $\mathsf{Banach}^{\uparrow}$ *carrier and, when orthogonal data are present, a compatible* $\mathsf{Hilbert}^{\uparrow}$ *carrier.*
2. *Y is the target endpoint row, read through the same Banach or Hilbert endpoint discipline.*
3. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the displayed linear action and norm-bound ledger from X to Y.*
4. *K is the kernel row, recording the finite kernel witness, finite-dimensional-kernel ledger, or explicit kernel-refusal boundary for B.*
5. *E is the cokernel-facing obstruction row, recording finite codimension evidence, range-defect data, or a displayed refusal to pass to a host quotient.*
6. *C is the* $\mathsf{CompactOperator}^{\uparrow}$ *compatibility row, available when the Fredholm read is consumed through a compact perturbation or compact remainder route.*
7. *I is the index ledger row, recording only the displayed difference between the kernel side K and the cokernel-facing side E as BEDC data.*
8. *A is the* $\mathsf{FredholmAlternative}^{\uparrow}$ *handoff row, exposing the same kernel-or-obstruction boundary when an alternative-facing consumer reads the operator.*

9. *H records componentwise* hsame *transport for endpoint, bounded-action, kernel, cokernel-facing, compactness, index, and handoff rows.*
10. *R records displayed* Cont *replay from endpoint admission through bounded action, kernel/cokernel-facing exposure, index readback, and handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FredholmOperator}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FredholmAlternative}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FredholmOperator}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host quotient cokernel, selected complement of the range, selected kernel basis, ambient Fredholm index equality, spectral decomposition, parametrix theorem, Lean proof, or standard bridge.

Theorem 8285.2 (Fredholm operator NameCert obligations)**.** *For every accepted* $\mathsf{FredholmOperator}^{\uparrow}$ *carrier*

$$F = (X, Y, B, K, E, C, I, A, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FredholmOperator}^{\uparrow}}$ *surface consists exactly of endpoint admission through X, Y, bounded-action exposure through B, kernel exposure through K, cokernel-facing obstruction exposure through E, compact-operator compatibility through C, index-ledger readback through I, alternative-facing handoff through A, componentwise transport through H, displayed replay through R, package provenance through P, and local naming through N.*

Proof. Project the carrier tuple. The endpoint rows X, Y are the only Banach or Hilbert sources, and the bounded-action row B is the only operator action available to the packet.

The finite Fredholm data are read next. The kernel side is K, the cokernel-facing obstruction side is E, and the index ledger I records only the displayed comparison between those two rows. Thus the index read cannot be replaced by a host quotient or an ambient dimension theorem.

The compactness and alternative rows are dependency handoffs. The compact row C lets compact-operator consumers recognize the same bounded action, while A exposes the same kernel-or-obstruction boundary to the $\mathsf{FredholmAlternative}^{\uparrow}$ surface.

Finally H, R, P, N preserve and name the route. Since the carrier contains no quotient cokernel, selected complement, selected basis, spectral decomposition, Lean target, or bridge coordinate, the listed rows exhaust the local certificate obligations. □

Theorem 8285.3 (Fredholm operator index kernel–cokernel handoff)**.** *Every index-facing consumer of an accepted* $\mathsf{FredholmOperator}^{\uparrow}$ *packet must read the bounded operator row B over the endpoint rows X, Y, then read the kernel row K and the cokernel-facing obstruction row E, before consuming the index ledger I or the* $\mathsf{FredholmAlternative}^{\uparrow}$ *handoff row A. The exported index surface is the finite route*

$$X, Y, B \longmapsto K, E \longmapsto I, A$$

together with H, R, P, N; it does not export a host quotient cokernel, selected range complement, ambient Fredholm index equality, spectral decomposition, Lean verification, or bridge checking.

Proof. Begin with the NameCert obligations of Definition 8285.2. They place endpoint admission and bounded-action exposure before the kernel, cokernel-facing, index, and handoff rows.

The rows K and E are paired but not identified with host objects. The kernel side records finite kernel evidence, and the cokernel-facing side records obstruction or finite-codimension evidence without passing to an ambient quotient carrier.

The index ledger I is downstream of that pair. It can be consumed only as the displayed BEDC comparison between K and E, and the alternative handoff A carries the same boundary into chapter 3425.

Componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming preserve this same route. Therefore downstream index packets, including the analytic-index side of chapter 273, receive a finite operator/index dependency rather than an external Fredholm operator notion. □

Falsifiable prediction: Every accepted $\mathsf{FredholmOperator}^{\uparrow}$ index read displays bounded action, kernel, cokernel-facing obstruction, index ledger, transport, replay, provenance, and local NameCert rows before any index-facing consumer may use the packet.

Independence witness: boundedlinearoperator, compactoperator, banach, hilbert, fredholm-alternative: $\mathsf{FredholmOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds endpoint admission, bounded action, kernel evidence, cokernel-facing obstruction, compact compatibility, index readback, and alternative handoff as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FredholmOperator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FredholmOperator}^{\uparrow}$ packet is generated from Banach or Hilbert endpoint rows, a bounded-linear-operator row, finite kernel evidence, cokernel-facing obstruction data, compact-operator compatibility, an index ledger, a Fredholm-alternative handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FredholmOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8285.1, Definition 8285.2, and Definition 8285.3 seed the finite Fredholm operator carrier over endpoint, bounded-action, kernel, cokernel-facing, compactness, index, handoff, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FredholmOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host quotient cokernels, selected complements, selected kernel bases, ambient Fredholm index equality, spectral decompositions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, bounded-action exposure, kernel exposure, cokernel-facing obstruction exposure, compact compatibility, index-ledger readback, alternative handoff, transport, replay, provenance, and local naming for BEDC.Derived.FredholmOperatorUp.

Chapter 8286

A Concrete Naming Certificate for $\mathsf{HenstockGaugePartition}^{\uparrow}$

Following Henstock's gauge integration and Cousin-style fine partitions, the $\mathsf{HenstockGaugePartition}^{\uparrow}$ packet names the finite tagged partition object that sits between the existence route of chapter 1861, the fine-partition ledger of chapter 6467, the control route of chapter 7413, and the integration consumer of chapter 1999. It records only displayed tagged intervals, pointwise gauge-radius rows, endpoint readback through located intervals and $\mathsf{Real}^{\uparrow}$, finite fineness and coverage ledgers, replay, provenance, and a local certificate.

Definition 8286.1 (Henstock gauge partition carrier)**.** *A* $\mathsf{HenstockGaugePartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, G, T, E, F, C, D, R, Q, N)$$

whose rows expose:

1. *I, the interval source row read through* $\mathsf{LocatedInterval}^{\uparrow}$ *and sealed against the endpoint order available in* $\mathsf{Real}^{\uparrow}$*;*
2. *G, the gauge-radius row assigning a positive local radius to every admitted tag over the displayed interval;*
3. *T, the finite tagged-interval ledger listing tags, left and right endpoints, cell order, and cell membership;*
4. *E, the endpoint readback row carrying the* $\mathsf{LocatedInterval}^{\uparrow}$ *boundary data and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal for each tagged cell;*
5. *F, the fineness ledger stating that every tagged interval of T lies inside the radius supplied by G at its tag;*
6. *C, the coverage ledger stating that the finite tagged intervals cover the displayed interval without leaving a named gap;*
7. *D, the disjointness ledger stating that distinct interiors in the tagged-interval list do not overlap;*
8. *R, the displayed* Cont *replay row preserving the route from interval and gauge data to the same finite partition packet;*
9. *Q, the* Pkg *provenance over* $\mathsf{HenstockGaugePartition}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{HenstockLemma}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, Cont, *and* $\mathsf{NameCert}$*;*

10. N, the local $\mathsf{NameCert}_{\mathsf{HenstockGaugePartition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, G, T, E, F, C, D, R, Q, N$. It has no coordinate for a choice-selected partition family, open-cover compactness, quotient equality of functions, an external integral theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.HenstockGaugePartitionUp`

Theorem 8286.2 (Henstock gauge partition NameCert obligations)**.** *For every accepted* $\mathsf{HenstockGaugePartition}^{\uparrow}$ *carrier $P = (I, G, T, E, F, C, D, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HenstockGaugePartition}^{\uparrow}}$ *surface is exhausted by interval-source admission, gauge-radius admission, finite tagged-interval admission, endpoint readback through* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, fineness, coverage, disjointness, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FinePartitionGauge}^{\uparrow}$*,* $\mathsf{HenstockLemma}^{\uparrow}$*, or* $\mathsf{GaugeIntegral}^{\uparrow}$ *consumer must read these rows before using the partition as a gauge-fine object.*

Proof. Project the carrier of Definition 8286.1. The analytic rows are I, G, T, E, F, C, D, and the structural rows are R, Q, N.

The endpoint and fineness obligations are read before any integral-facing consumer can use the packet. The endpoint row E binds each tagged interval to located endpoint data and to the terminal $\mathsf{Real}^{\uparrow}$ seal, while F checks the cell against the radius supplied by G at the displayed tag.

Coverage and disjointness are finite ledger rows on the same tagged-interval list. The replay and provenance rows preserve that finite route, and the local certificate has no coordinate for host choice, open-cover compactness, quotient equality, external integration, or ambient completeness. Hence the listed rows exhaust the local NameCert surface. □

Theorem 8286.3 (Henstock gauge partition fineness ledger)**.** *Every accepted* $\mathsf{HenstockGaugePartition}^{\uparrow}$ *carrier exposes a single finite route*

$$I, G, T, E \longmapsto F \longmapsto C, D$$

from located endpoint data and positive gauge radii to the coverage and disjointness ledgers. The row F is the only fineness witness visible to $\mathsf{FinePartitionGauge}^{\uparrow}$ *and* $\mathsf{HenstockLemma}^{\uparrow}$ *consumers, and it is read from the same tags and endpoints displayed by T and E.*

Proof. Start with I, G, T, E. These rows name the displayed interval, the positive radius assigned to each admitted tag, the finite tagged-interval list, and the located endpoint readback.

The fineness row F consumes exactly those rows. It cannot change the tag list or endpoints because it is a ledger row over the tagged intervals already listed in T and read back in E.

The coverage and disjointness rows C, D are downstream finite checks on the same list. Thus any fine-partition or Henstock-lemma consumer that reads the packet receives the displayed fineness ledger before it receives coverage or disjointness as usable partition data. □

Theorem 8286.4 (Henstock gauge partition integral consumer handoff)**.** *Every* $\mathsf{GaugeIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{HenstockGaugePartition}^{\uparrow}$ *carrier factors through the partition route*

$$I, G, T, E, F, C, D \longmapsto R, Q, N.$$

The consumer receives located endpoint readback, $\mathsf{Real}^{\uparrow}$ *endpoint sealing, gauge fineness, coverage, disjointness, replay, provenance, and local naming; it receives no selected family of partitions, no open-cover compactness theorem, no quotient equality of functions, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the finite partition rows first. The interval row I, gauge row G, tagged-interval row T, and endpoint row E bind the consumer to one displayed partition packet.

Next consume the ledger rows F, C, D. They certify gauge fineness, coverage, and interior disjointness over the same finite list, so the gauge-integral consumer does not import a separate existence theorem at the point of use.

Finally read the replay, provenance, and local naming rows. These rows preserve the route into $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{FinePartitionGauge}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, and $\mathsf{HenstockLemma}^{\uparrow}$ without adding any host choice, compactness, quotient, external integral, or ambient completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{HenstockGaugePartition}^{\uparrow}$ packet must expose interval source, gauge-radius, finite tagged-interval, located endpoint, Real seal, fineness, coverage, disjointness, replay, provenance, and local NameCert rows before a gauge-integral consumer reads it.

Independence witness: gaugeintegral, finepartitiongauge, cousinlemma, henstocklemma, locatedinterval, real: $\mathsf{HenstockGaugePartition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tagged intervals, pointwise gauge-radius data, located endpoint readback, fineness, coverage, disjointness, replay, provenance, and local naming before the integral consumer handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockGaugePartition}^{\uparrow}}$.

Constructive story: A $\mathsf{HenstockGaugePartition}^{\uparrow}$ packet is generated from interval source data, gauge-radius data, finite tagged intervals, located endpoint readback, Real endpoint sealing, fineness, coverage, disjointness, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockGaugePartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8286.1, Definition 8286.2, Definition 8286.3, and Definition 8286.4 seed the finite Henstock gauge-partition object between fine-partition, Henstock-lemma, and gauge-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HenstockGaugePartition}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close selected partition families, open-cover compactness, quotient equality of functions, external integral theorems, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, gauge-radius admission, finite tagged-interval admission, endpoint readback, fineness, coverage, disjointness, replay, provenance, and local naming for BEDC.Derived.HenstockGaugePartitio

Chapter 8287

A Concrete Naming Certificate for $\mathsf{RealUniformPartition}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RealUniformPartition}^{\uparrow}$ packet records the finite uniform partitions used to inspect a closed real interval by dyadic mesh scale. It sits after the dyadic tolerance arithmetic of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the located interval-cover route of chapter 1885. The packet names endpoints, mesh cells, finite windows, readback rows, and local certificate data only; it does not assert open-cover compactness, a finite subcover principle, quotient real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8287.1 (RealUniformPartition carrier)**.** *A* $\mathsf{RealUniformPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, B, M, I, E, S, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, read only through their displayed real-seal rows.*
2. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-scale row, recording a positive dyadic step, its refinement budget, and the comparison cells used by the finite partition.*
3. *I is the ordered finite window of partition indices, stored as constructor-generated finite data rather than as a host interval or hidden enumeration.*
4. *E is the endpoint ledger assigning to each displayed index in I the corresponding dyadic endpoint history between A and B.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row used to inspect endpoint names, mesh rows, and the finite index window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint and mesh observations displayed by S.*
7. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic mesh, ordered finite window, endpoint ledger, finite observations, and regular rational readback have been displayed.*
8. *H records componentwise* hsame *transport for endpoints, mesh scale, finite index window, endpoint ledger, StreamName observations, RegSeqRat readback, Real seal, replay, provenance, and local certificate.*

9. *C records displayed* Cont *replay of the route from endpoint names and dyadic mesh through finite windows, readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{RealUniformPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformPartition}^{\uparrow}}$ *row.*

The carrier predicate requires the finite endpoint ledger E to be ordered by the displayed dyadic mesh row M, to cover exactly the finite window I, and to be readable through S, R, Z, H, C, P, N. It has no coordinate for an arbitrary open cover, a selected subcover, quotient real equality, host interval equality, countable choice, or ambient completeness.

Theorem 8287.2 (RealUniformPartition NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformPartition}^{\uparrow}$ *carrier $U = (A, B, M, I, E, S, R, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealUniformPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{Real}^{\uparrow}$ *endpoint names, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-scale row, the ordered finite partition window, the endpoint ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every partition consumer must expose M, I, E, S, R before reading the Real seal Z.*

Proof. Project the accepted carrier. The rows before the Real seal are precisely the endpoint names A, B, the dyadic mesh M, the finite index window I, the endpoint ledger E, the finite observation row S, and the regular rational readback R.

The ordered partition obligation is finite. The mesh row supplies the dyadic step and comparison cells, while the finite window lists the partition indices and the endpoint ledger attaches each index to its displayed endpoint history.

Readback and sealing are downstream. The StreamName row records exactly the finite observations used by the RegSeqRat row, and the Real seal is then read from that regular rational boundary through Z.

Finally transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier contains no row for open-cover compactness, subcover choice, quotient real equality, host interval equality, countable choice, or completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8287.3 (RealUniformPartition RegSeqRat scope)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUniformPartition}^{\uparrow}$ *carrier must factor through the finite route*

$$A, B, M, I, E, S, R, Z, H, C, P, N.$$

In particular, the regular rational readback R is scoped by the common dyadic mesh row M and the displayed finite window I before the Real seal Z is available. The route exports no infinite stream, quotient real equality, selected limit, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier rows. The only window available to a StreamName-facing consumer is S, and S is displayed after the endpoint names, mesh scale, finite index window, and endpoint ledger have already been projected.

The RegSeqRat readback R therefore cannot be read independently of the finite partition. Its observations are exactly those exposed by S, with the mesh comparisons supplied by M and the endpoint rows supplied by E.

The Real seal Z is terminal in the same finite route. Transport by H and replay by C preserve the mesh, window, readback, provenance, and local name together, so a downstream Real-facing consumer remains attached to $A, B, M, I, E, S, R, Z, H, C, P, N$.

No excluded coordinate appears in the carrier inventory. Hence the scoped route cannot export an infinite stream, quotient real equality, selected limit, open-cover compactness, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealUniformPartition}^{\uparrow}$ read is refuted if a consumer reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without displaying the endpoint names, dyadic mesh scale, ordered finite window, endpoint ledger, finite StreamName observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-interval-cover: $\mathsf{RealUniformPartition}^{\uparrow}$ is not bijective to any listed sibling because it binds endpoint names, a dyadic mesh scale, ordered finite partition indices, endpoint ledgers, finite observations, regular rational readback, Real sealing, transport, replay, provenance, and local naming in one partition packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformPartition}^{\uparrow}$ packet is generated from RealUp endpoint names, a DyadicRatCoreUp mesh scale, an ordered finite partition window, endpoint ledger rows, StreamName finite observations, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealUniformPartition}^{\uparrow}}$, seed)

Scope closed: Definition 8287.1, Definition 8287.2, and Definition 8287.3 seed the finite uniform partition route from dyadic mesh data to StreamName, RegSeqRat, and RealUp consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealUniformPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, quotient real equality, host interval equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite mesh ordering, endpoint-ledger coverage, StreamName observation scope, RegSeqRat readback scope, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealUniformPartitionUp.

Chapter 8288

A Concrete Naming Certificate for RegularCauchyProductModulus$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyProductModulus$^{\uparrow}$ packet records the finite product-modulus boundary needed to read multiplication of displayed regular rational Cauchy names through bounded windows. It sits over the RegSeqRat$^{\uparrow}$ source rows of chapter 301, the synchronized window discipline of chapter 126, the DyadicRatCore$^{\uparrow}$ arithmetic rows of chapter 291, the terminal real-name seal of chapter 38, and the product carrier of chapter 1702. The packet is a finite modulus certificate for the product window; it is not quotient real multiplication, a selected limit, trichotomy, an ambient completeness theorem, or a host real norm.

Definition 8288.1 (RegularCauchyProductModulus carrier)**.** *A* RegularCauchyProductModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, W, D, R, P, E, H, C, Q, N)$$

with the following displayed rows:

1. *X and Y are the two* RegSeqRat$^{\uparrow}$ *source rows for regular rational Cauchy real names.*
2. *W is the* StreamName$^{\uparrow}$ *product-window ledger that lists the finite source observations inspected for each requested output precision.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *error-budget row containing the displayed source tolerances, product tolerance, and slack allocation used by the window computation.*
4. *R is the regular readback row that turns the windowed dyadic product data into a* RegSeqRat$^{\uparrow}$ *product-name prefix.*
5. *P is the product-modulus row. For each requested output precision it stores the finite source-window depth and dyadic slack sufficient for the product readback.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after X, Y, W, D, R, P have supplied the regular product-modulus boundary.*
7. *H records componentwise* hsame *transport for the source rows, product windows, dyadic error budget, regular readback, product modulus, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from any product-modulus precision request through $X, Y, W, D, R, P, E, H, Q, N$.*
9. *Q is the* Pkg *provenance over* RegularCauchyProductModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyMultiplication$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, R, P, E, H, C, Q, N$. The carrier has no coordinate for quotient real multiplication, selected limits, trichotomy, global completeness, host real norms, Lean verification, or bridge checking.

Theorem 8288.2 (Regular Cauchy product window bound)**.** *For every accepted* $\mathsf{RegularCauchyProductModulus}^{\uparrow}$ *carrier*

$$M = (X, Y, W, D, R, P, E, H, C, Q, N),$$

each product read requested at a displayed output precision factors through a finite window bound recorded in P before the regular readback row R or the $\mathsf{Real}^{\uparrow}$ *seal row E may be consumed.*

Proof. Start with the precision request handled by the accepted carrier. The only row that assigns a product-facing source-window depth is P, and P is keyed to the synchronized observations listed by W and to the dyadic slack listed by D.

Once P has named that finite bound, the readback row R may use only the displayed source prefixes from X and Y, the window ledger W, and the error-budget row D. Thus the product readback is determined before any terminal real-name seal is exposed.

The $\mathsf{Real}^{\uparrow}$ seal row E is downstream from X, Y, W, D, R, P. Since the carrier contains no host norm, quotient Cauchy class, selected limit, or ambient completeness coordinate, every accepted product read reaches E only through the finite window bound already recorded in P. □

Theorem 8288.3 (Regular Cauchy product-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyProductModulus}^{\uparrow}$ *carrier*

$$M = (X, Y, W, D, R, P, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyProductModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *product windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-budget row, regular readback, product-modulus row,* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows;*
2. *source-window exposure, requiring each product-modulus request to inspect only the finite source prefixes named by W;*
3. *dyadic error-budget exposure, requiring D to carry the displayed source tolerances, product tolerance, and slack allocation used by the product bound;*
4. *product-modulus determinacy, requiring P to name the finite bound before the readback row R or the seal row E is available;*
5. *regular readback compatibility, requiring R to be the* $\mathsf{RegSeqRat}^{\uparrow}$ *prefix obtained from the rows X, Y, W, D, P;*
6. *Real seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to read through X, Y, W, D, R, P before E is exposed;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, Q, N to preserve and name exactly the same finite product-modulus packet;*
8. *refusal of quotient real multiplication, selected limits, trichotomy, global completeness, host real norms, Lean verification, and bridge checking as local NameCert data.*

No product-modulus obligation is stored outside these finite rows.

Falsifiable prediction: An accepted $\mathsf{RegularCauchyProductModulus}^{\uparrow}$ read is refuted if a product-facing precision request reaches R or E without a finite source-window bound in P, a synchronized $\mathsf{StreamName}^{\uparrow}$ window in W, and a $\mathsf{DyadicRatCore}^{\uparrow}$ error-budget row in D.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-multiplication: $\mathsf{RegularCauchyProductModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular Cauchy source rows, product windows, dyadic error budgets, a regular readback row, a product-modulus row, a real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductModulus}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a finite StreamName product-window ledger, a DyadicRatCore error-budget row, a regular readback row, a product-modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8288.1, Definition 8288.2, and Definition 8288.3 seed the finite product-modulus boundary consumed by regular Cauchy multiplication and Real-facing completion readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real multiplication, selected limits, trichotomy, host real norms, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, dyadic error-budget exposure, product-modulus determinacy, regular readback compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyProductModulusUp.

Chapter 8289

A Concrete Naming Certificate for $\mathsf{MartingaleFiltration}^{\uparrow}$

Following Doob on filtrations and martingales, the $\mathsf{MartingaleFiltration}^{\uparrow}$ packet records the finite observation schedule that a martingale or stopped martingale may consume before it reads conditional-expectation data. It sits between the probability source of chapter 184, the random-variable endpoint rows used by chapter 190, the finite stopping decisions of chapter 344, and the conditional-expectation faces of chapter 188. The packet gives martingale and stopping-time consumers an explicit filtration carrier; it is not a completed sigma-algebra tower, a selected optional process, an almost-sure quotient, or a measure-completion theorem.

Definition 8289.1 (Martingale filtration carrier)**.** *A* $\mathsf{MartingaleFiltration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (\Omega, W, A, C, S, B, T, R, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the probability source and its event classifier.*
2. W *is a finite filtration-window ledger whose entries name the inspected times and the event rows available at each time.*
3. A *is the adaptedness row tying each displayed* $\mathsf{RandomVar}^{\uparrow}$ *endpoint read to the corresponding entry of* W*.*
4. C *is the family of* $\mathsf{CondExp}^{\uparrow}$ *faces that compare successor endpoint rows against the previous filtration entry.*
5. S *is the bounded stopping-window row recording which entries of* W *may be stopped and read by* $\mathsf{StoppingTime}^{\uparrow}$ *or* $\mathsf{BoundedStoppingTime}^{\uparrow}$ *consumers.*
6. B *is the optional-stopping handoff row for bounded windows, read only after* W, A, C, S *have been displayed.*
7. T *records componentwise* hsame *transport for the probability source, filtration window, adaptedness, conditional-expectation faces, stopping rows, handoff, provenance, and local naming.*
8. R *records displayed* Cont *replay of the same finite route.*
9. P *is the* Pkg *provenance over* $\mathsf{MartingaleFiltration}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{BoundedStoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MartingaleFiltration}^{\uparrow}}$ *row.*

The carrier has no coordinate for completed filtrations, countable joins of sigma-algebras, optional projections, unbounded stopping choices, almost-sure quotient equality, or ambient measure completion.

Lean4 ✓ `BEDC.Derived.MartingaleFiltrationUp.TasteGate.MartingaleFiltrationTasteGate single carrier alignment`

Theorem 8289.2 (Martingale filtration NameCert obligations)**.** *For every accepted* $\mathsf{MartingaleFiltration}^{\uparrow}$ *carrier* $F = (\Omega, W, A, C, S, B, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MartingaleFiltration}^{\uparrow}}$ *surface consists exactly of carrier admission, finite filtration-window exposure, adaptedness of endpoint rows through* A*, conditional-expectation exposure through* C*, bounded stopping-window exposure through* S*, optional-stopping handoff through* B*, componentwise transport through* T*, displayed replay through* R*, provenance through* P*, and local naming through* N*. Every public martingale or stopping-time read of the packet must preserve those rows before it reaches a* $\mathsf{Martingale}^{\uparrow}$*,* $\mathsf{StoppingTime}^{\uparrow}$*,* $\mathsf{BoundedStoppingTime}^{\uparrow}$*, or* $\mathsf{OptionalStopping}^{\uparrow}$ *consumer.*

Proof. Project the carrier of Definition 8289.1. The only analytic coordinates are the probability source, the finite filtration window, adaptedness rows, conditional-expectation faces, bounded stopping rows, and optional-stopping handoff.

The martingale-facing route first reads W and A. Thus each endpoint row is attached to a visible filtration entry before the $\mathsf{Martingale}^{\uparrow}$ consumer reads conditional-expectation information.

The stopping-facing route then reads S and B. The stopping row records the bounded decision window, while the handoff row records the optional-stopping surface only after the same W, A, C rows have already been exposed.

Finally T, R, P, N transport, replay, source, and name only that displayed packet. Since the carrier omits completed filtrations, countable joins, optional projections, unbounded stopping choices, almost-sure quotient equality, and measure completion, no local obligation can export those endpoints. □

Theorem 8289.3 (Martingale filtration stopping-window handoff)**.** *Every accepted* $\mathsf{MartingaleFiltration}^{\uparrow}$ *public read that is consumed by a* $\mathsf{StoppingTime}^{\uparrow}$*,* $\mathsf{BoundedStoppingTime}^{\uparrow}$*, or* $\mathsf{OptionalStopping}^{\uparrow}$ *row first factors through the common finite window* W*, the adaptedness row* A*, the conditional-expectation faces* C*, and the bounded stopping-window row* S*. Hence a stopped martingale read receives the same finite filtration rows that the* $\mathsf{Martingale}^{\uparrow}$ *conditional-expectation read uses, rather than an ambient filtration or an unbounded stopping selector.*

Proof. Start from an accepted carrier $F = (\Omega, W, A, C, S, B, T, R, P, N)$. The window W is the only filtration coordinate, so every time-indexed event read begins there.

Apply the NameCert obligations of Definition 8289.2. They force martingale endpoint rows through A and conditional-expectation faces through C before any public martingale consumer can read the packet.

For stopping consumers, the same obligation surface forces bounded stopping reads through S before the optional-stopping handoff B is available. The structural rows T, R, P, N preserve the same finite route, so the stopped read cannot replace the displayed window by a completed filtration, an unbounded stopping rule, or an almost-sure quotient. □

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleFiltration}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleFiltration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8289.1, Definition 8289.2, and Definition 8289.3 seed the finite filtration-window packet consumed by martingale and bounded-stopping rows.

Formal verification: FormalStatus(NameCert$_{\text{MartingaleFiltration}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.MartingaleFiltrationUp.TasteGate.MartingaleFiltrationTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close completed filtrations, countable joins of sigma-algebras, optional projections, unbounded stopping choices, almost-sure quotient equality, measure completion, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, adaptedness routing, conditional-expectation face exposure, bounded stopping-window handoff, transport, replay, provenance, and local naming for BEDC.Derived.MartingaleFiltrationUp.

Constructive story: A MartingaleFiltration$^{\uparrow}$ packet is generated from a probability-space row, a finite filtration-window ledger, adapted random-variable endpoint rows, conditional-expectation faces, bounded stopping-window rows, an optional-stopping handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8290

A Concrete Naming Certificate for $\mathsf{DyadicUniformCover}^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite covers, the $\mathsf{DyadicUniformCover}^{\uparrow}$ packet records the dyadic finite-cover side of the L10 compactness route described in chapter 9237. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the compact interval source of chapter 3986. The packet names finite cover rows only; it is not an open-cover compactness theorem, a selected infinite subcover, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8290.1 (Dyadic uniform cover carrier)**.** *A* $\mathsf{DyadicUniformCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, D, W, Q, R, C, H, K, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *source interval row.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger naming dyadic endpoints, radii, overlaps, and tolerance comparisons.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *inspection window used to read interval endpoints and cover cells.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for dyadic cell centers and endpoint approximations.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, Q, D have been displayed.*
6. *C is the finite cover row listing the displayed dyadic cells, their membership checks, and the local coverage certificate for the interval row I.*
7. *H records componentwise* hsame *transport for the interval, dyadic, stream, regular-rational, real-seal, cover, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay of the same finite cover route.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicUniformCover}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicUniformCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, infinite subcover selection, quotient stream equality, host real equality, trichotomy, countable choice, selected limits, or ambient compactness.

Theorem 8290.2 (Dyadic uniform cover NameCert obligations)**.** *For every accepted* $\mathsf{DyadicUniformCover}^{\uparrow}$ *carrier* $U = (I, D, W, Q, R, C, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicUniformCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact interval source, dyadic ledger, finite StreamName window, regular-rational readback, Real seal, finite cover row, componentwise transport, displayed replay, provenance, and local naming. Every compactness or completion consumer of the packet must read the cover row* C *through* D, W, Q, R *before it consumes the interval row* I *as a finite-cover source.*

Proof. Project the carrier of Definition 8290.1. Its analytic rows are exactly I, D, W, Q, R, C, with H, K, P, N providing structural transport, replay, provenance, and local naming.

The dyadic ledger D names the cell endpoints and tolerance comparisons. The stream window W and regular-rational row Q expose only finitely many endpoint and center reads before the terminal real seal R is visible.

The cover row C is therefore consumed as a finite dyadic cover over the interval row I. Since arbitrary open covers, infinite subcover selection, quotient equality, host real equality, trichotomy, countable choice, selected limits, and ambient compactness are absent from the carrier, no obligation can export them. □

Theorem 8290.3 (Dyadic uniform cover L10 handoff)**.** *Every L10 compactness or completion consumer of an accepted* $\mathsf{DyadicUniformCover}^{\uparrow}$ *packet factors through the dyadic ledger* D*, the finite StreamName window* W*, the regular-rational readback* Q*, the Real seal* R*, and the finite cover row* C *before it consumes the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *source* I*. Thus the finite-cover direction is a displayed dyadic NameCert route, not an open-cover compactness principle or a choice-selected subcover.*

Proof. Begin with the accepted carrier $U = (I, D, W, Q, R, C, H, K, P, N)$. The interval source I is not read in isolation; the carrier pairs it with the dyadic ledger and finite observation rows.

Apply Definition 8290.2. It requires every public compactness or completion read to expose D, W, Q, R, C before treating I as a finite-cover source.

Finally read the structural rows H, K, P, N. They transport, replay, source, and name only the displayed dyadic cover route, so the handoff cannot be recast as arbitrary open-cover compactness, infinite subcover choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicUniformCover}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicUniformCover}^{\uparrow}}$, seed)

Scope closed: Definition 8290.1, Definition 8290.2, and Definition 8290.3 seed the finite dyadic cover route for compact interval and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicUniformCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, infinite subcover selection, quotient stream equality, host real equality, trichotomy, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exactness, finite-window readback, regular-rational handoff, Real-seal exposure, finite-cover coverage, transport, replay, provenance, and local naming for BEDC.Derived.DyadicUniformCoverUp.

Constructive story: A $\mathsf{DyadicUniformCover}^{\uparrow}$ packet is generated from a Bishop compact interval row, dyadic endpoint and tolerance data, a finite StreamName window, RegSeqRat readback, a Real seal, finite cover cells, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8291

A Concrete Naming Certificate for $\mathsf{RationalBallDomain}^{\uparrow}$

Following Edalat–Heckmann formal-ball domains and Bishop–Bridges constructive completion, the $\mathsf{RationalBallDomain}^{\uparrow}$ packet records the finite rational-ball domain surface used by L10 completion routes before any completed point can be consumed. It sits after the rational-ball carrier of chapter 355, the rational-ball basis of chapter 7051, the formal-ball domain route of chapter 5723, the dyadic tolerance core of chapter 291, the finite stream-name windows of chapter 126, and the regular rational readback of chapter 301. The packet gives downstream Real and Completion chapters a non-quotient rational-ball-domain handoff; it does not choose Cauchy representatives, quotient filters, invoke host point equality, or assume ambient completeness.

Definition 8291.1 (RationalBallDomain carrier)**.** *A* $\mathsf{RationalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, K, F, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{RationalBall}^{\uparrow}$ *observation row, carrying finite rational center-radius packets.*
2. *K is the* $\mathsf{RationalBallBasis}^{\uparrow}$ *row, recording the inspected basis windows and their finite refinement comparisons.*
3. *F is the* $\mathsf{FormalBallDomain}^{\uparrow}$ *row, supplying the rounded-domain order discipline into which rational balls are embedded.*
4. *O is the overlap and directedness ledger for finite families of rational balls.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule used by completion-facing consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed rational-ball windows.*
7. *E is the endpoint handoff row, read by* $\mathsf{Completion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers only after B, K, F, O, S, R have been displayed.*
8. *H records componentwise* hsame *transport for rational balls, basis windows, formal-domain rows, overlap ledgers, stream schedules, regular readbacks, endpoints, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from rational-ball-domain requests through $B, K, F, O, S, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RationalBallDomain}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{RationalBallBasis}^{\uparrow}$, $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, K, F, O, S, R, E, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host metric equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RationalBallDomainUp.RationalBallDomainTasteGate single carrier alignment`

Theorem 8291.2 (RationalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{RationalBallDomain}^{\uparrow}$ *carrier $D = (B, K, F, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalBallDomain}^{\uparrow}}$ *surface consists exactly of carrier admission for the rational-ball row, rational-ball-basis row, formal-ball-domain row, overlap ledger, stream schedule, regular rational readback, endpoint handoff, componentwise transport, displayed replay, provenance, and local naming. In particular, rational-ball containment must be read through B, K, F, O, completion-facing observation through S, R, E, and structural stability through H, C, P, N.*

Proof. Project the carrier. The rational-ball source is visible only as B, the basis route only as K, and the rounded-domain discipline only as F.

The overlap row O records the finite directed family of rational balls before any stream or regular-rational consumer can read it. The schedule and readback rows S and R therefore remain downstream of the displayed rational-ball-domain surface, and the endpoint row E is terminal for that route.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient completion, selected representatives, host metric equality, countable choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 8291.3 (RationalBallDomain completion handoff)**.** *Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalBallDomain}^{\uparrow}$ *carrier must first expose the finite route*

$$\mathsf{RationalBall}^{\uparrow} \longmapsto \mathsf{RationalBallBasis}^{\uparrow} \longmapsto \mathsf{FormalBallDomain}^{\uparrow} \longmapsto O \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports rational-ball-domain coverage, finite stream observation, regular rational readback, endpoint sealing, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It exports no quotient sequence equality, selected limit, host metric equality, countable-choice selector, or ambient completion theorem.*

Proof. Start with the accepted carrier $D = (B, K, F, O, S, R, E, H, C, P, N)$. A completion-facing request first reads the rational-ball row B and the basis row K, then checks that those windows live inside the rounded formal-domain row F.

The overlap ledger O is the finite directedness witness for the rational-ball family. Only after that row is displayed can the consumer read the finite schedule S and the regular rational readback R.

The endpoint row E is therefore downstream of B, K, F, O, S, R. Transport through H and replay through C keep the same route, while P and N source and name it. The excluded quotient, selected-limit, host-equality, choice, and ambient-completeness data have no row from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{RationalBallDomain}^{\uparrow}$ completion-facing read displays RationalBallUp, RationalBallBasisUp, FormalBallDomainUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, endpoint handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a Real or Completion consumer can use the endpoint.

Independence witness: rationalball, rational-ball-basis, formal-ball-domain, dyadicratcore, streamname, regseqrat: $\mathsf{RationalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds rational-ball coverage, basis refinement, rounded formal-domain discipline, dyadic tolerance, finite stream observation, regular rational readback, endpoint handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBallDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBallDomain}^{\uparrow}$ packet is generated from RationalBallUp rows, RationalBallBasisUp rows, FormalBallDomainUp rows, finite overlap ledgers, StreamNameUp schedules, RegSeqRatUp readback rows, CompletionUp and RealUp endpoint handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBallDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8291.1, Definition 8291.2, and Definition 8291.3 seed the finite rational-ball-domain route from rational coverage to completion-facing endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBallDomain}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RationalBallDomainUp.RationalBallDomainTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host metric equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-ball coverage, basis refinement, formal-domain embedding, overlap directedness, stream scheduling, regular readback, endpoint handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalBallDomainUp.

Chapter 8292

A Concrete Naming Certificate for $\mathsf{PositiveRealSquareRoot}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ packet records the finite square-root service available for a displayed positive real source before any total real square-root operator is claimed. It sits after the regular Cauchy square-root service of chapter 1756, the Heron recurrence route of chapter 3818, the dyadic tolerance ledger of chapter 291, the finite stream window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a bounded constructive service for one positive input; it is not a host square-root function, a trichotomy principle, quotient real equality, selected limit construction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8292.1 (Positive real square-root carrier)**.** *A* $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, A, G, R, H, D, S, V, E, T, C, P, N)$$

with the following visible rows:

1. *X is the positive* $\mathsf{Real}^{\uparrow}$ *source row whose displayed witness keeps the input in the positive real interface;*
2. *A is the apartness and nonnegative guard row, recording the local refusal of the zero or negative branch without invoking real trichotomy;*
3. *G is the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *readback row for the regular Cauchy square-root service;*
4. *R is the* $\mathsf{HeronSquareRoot}^{\uparrow}$ *recurrence route that supplies the finite iterative square-root approximants;*
5. *H is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed Heron and regular Cauchy estimates;*
6. *D is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only finitely many source and approximation observations;*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational square-root name;*
8. *V is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the source, guard, recurrence, tolerance, window, and readback rows are displayed;*
9. *E stores componentwise* hsame *transport for the source, guard, recurrence, tolerance, window, readback, and seal rows;*

10. *T records the displayed* Cont *replay by which a consumer reads the same finite square-root service;*
11. *P records* Pkg *provenance over* $\mathsf{PositiveRealSquareRoot}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{HeronSquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveRealSquareRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a total host square-root function, a trichotomy case split on arbitrary reals, quotient real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8292.2 (Positive real square-root NameCert obligations)**.** *For the carrier of Definition 8292.1, the local* $\mathsf{NameCert}_{\mathsf{PositiveRealSquareRoot}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier admission for $Q = (X, A, G, R, H, D, S, V, E, T, C, P, N)$, positivity and apartness guarding through X, A, regular Cauchy square-root readback through G, S, V, Heron recurrence and dyadic tolerance accounting through R, H, D, and componentwise transport, replay, provenance, and local naming through E, T, C, P, N.*

Proof. Project the carrier rows. The source and guard rows X, A are displayed before any square-root service is read, so the certificate speaks only about a positive input with an explicit apartness and nonnegative guard.

Next compose the sibling routes. The regular Cauchy square-root service is read through G, while the Heron recurrence row R supplies the constructive approximants and the dyadic tolerance row H records the finite numerical budget exposed through the stream windows D.

Finally pass through the readback, seal, transport, replay, provenance, and naming rows S, V, E, T, C, P, N. These rows preserve only the displayed finite service, and none of them adds a host square-root operator, trichotomy, quotient real equality, selected limit, or ambient completeness principle. □

Theorem 8292.3 (Positive real square-root regular Cauchy service)**.** *Every accepted* $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ *consumer that asks for a square-root value must first read the positive* $\mathsf{Real}^{\uparrow}$ *source row, the apartness and nonnegative guard, the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *readback row, the* $\mathsf{HeronSquareRoot}^{\uparrow}$ *recurrence route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. The service therefore exports only the bounded regular Cauchy square-root route for the displayed positive input.*

Proof. Begin with the NameCert obligations of Definition 8292.2. They require $X, A, G, R, H, D, S, V, E, T, C, P, N$ before a consumer can read the local square-root name.

Use chapter 1756 for the regular Cauchy readback and chapter 3818 for the Heron recurrence route. The dyadic ledger and stream windows keep the requested tolerance budget finite before the readback row S reaches the real seal V.

The structural rows E, T, C, P, N then transport, replay, source, and name the same bounded route. Since the carrier has no coordinate for a host square-root function, real trichotomy, quotient real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no consumer can read those objects from the service. □

Falsifiable prediction: Any accepted PositiveRealSquareRootUp service that bypasses the positive source guard, Heron recurrence route, dyadic tolerance ledger, stream windows, regular rational readback, or terminal Real seal refutes the carrier.

Independence witness: regular-cauchy-square-root, heron-square-root, dyadicratcore, streamname, regseqrat, real: $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a positive source guard, recurrence route, tolerance ledger, finite windows, readback, terminal seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveRealSquareRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ packet is generated from a positive RealUp source row, an apartness and nonnegative guard, regular Cauchy square-root readback, Heron recurrence, dyadic tolerance accounting, stream windows, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveRealSquareRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8292.1, Definition 8292.2, and Definition 8292.3 seed the bounded positive-input square-root service over the displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveRealSquareRoot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host square-root function, real trichotomy, quotient real equality, selected limits, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, positive-source guarding, Heron recurrence accounting, dyadic tolerance accounting, regular Cauchy readback, stream-window replay, Real sealing, transport, provenance, and local naming for BEDC.Derived.PositiveRealSquareRootUp.

Chapter 8293

A Concrete Naming Certificate for $\mathsf{RegularCauchySieve}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy object is read through explicit moduli and finite tails before any completion seal is consumed. The $\mathsf{RegularCauchySieve}^{\uparrow}$ packet records that reading as a finite directed-tail sieve for the L10 Real and Completion route, sitting between the dyadic core of chapter 291, the finite request windows of chapter 126, the regular rational sequence readback of chapter 301, and the Real seal of chapter 38.

Definition 8293.1 (Regular Cauchy sieve carrier)**.** *A* $\mathsf{RegularCauchySieve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, Q, L, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the finite dyadic scale at which the tail is inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the requests whose indices lie beyond the displayed tail bound.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence readback row reached from the same dyadic tolerance and stream window.*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after D, W, Q have been displayed.*
5. *T is the directed finite-tail sieve row: every accepted refinement replaces D, W, Q, L by a later finite window over the same route and records the inclusion of the older window into the later one.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, window, regular-sequence readback, Real seal, sieve refinement, and local naming rows.*
7. *C records displayed* Cont *replay from dyadic tolerance through StreamName window, RegSeqRat readback, directed-tail refinement, Real seal, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySieve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for a selected infinite subsequence, quotient-stream equality, host limit, ambient completeness principle, or hidden modulus choice outside the directed finite-tail sieve.

Theorem 8293.2 (Regular Cauchy sieve NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySieve}^{\uparrow}$ *carrier*

$$S = (D, W, Q, L, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySieve}^{\uparrow}}$ *surface consists exactly of dyadic tolerance exposure through* D*, finite StreamName window exposure through* W*, RegSeqRat readback through* Q*, RealUp seal exposure through* L*, directed finite-tail refinement through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8293.1. The rows D, W, Q, L expose the L10 route in dependency order: dyadic tolerance, finite request window, regular rational readback, and Real seal.

Read the sieve row T next. It is directed because any two displayed finite tails may be refined by a later finite window over the same dyadic, StreamName, RegSeqRat, and RealUp route; the packet records that later window as another finite row rather than selecting an infinite subsequence.

The transport, replay, provenance, and naming rows preserve only this displayed route. Since no coordinate stores a host limit, quotient stream, ambient completeness rule, or hidden modulus, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySieve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySieve}^{\uparrow}$ packet is generated from dyadic tolerance data, finite StreamName request windows, RegSeqRat readback, a RealUp seal, a directed finite-tail refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySieve}^{\uparrow}}$, seed)

Scope closed: Definition 8293.1 and Definition 8293.2 seed the finite directed-tail sieve used by the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySieve}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected limits, infinite subsequence choice, hidden modulus extraction, or Lean verification for $\mathsf{RegularCauchySieve}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, directed finite-tail refinement, transport stability, replay, provenance, and local naming for the same finite L10 route.

Chapter 8294

A Concrete Naming Certificate for $\mathsf{RegularCauchyRateTransform}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ packet records the finite transformation from arbitrary displayed Cauchy modulus data to the regular-Cauchy rate discipline consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. It sits after the regular rational readback surface of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance rows of chapter 291, the terminal real-name seal of chapter 38, and the modulus comparison route of chapter 7213. The packet names the L10 regularization route for Cauchy modulus rows without selecting limits, quotienting streams, or asserting ambient completeness.

Definition 8294.1 (Regular Cauchy rate transform carrier)**.** *A* $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, W, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source Cauchy modulus table. It records the finite threshold rows available from the input Cauchy data before any regular rate is read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source thresholds are inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic requests, target regular tolerances, and comparison budget used by the rate transformation.*
4. *R is the rate-normalization replay row. For every displayed target tolerance it names the finite source threshold and window read used to produce a regular-Cauchy rate request.*
5. *H stores componentwise* hsame *transport for M, W, D, R and for the structural rows below.*
6. *C stores displayed* Cont *replay from a regular-rate request through the source modulus, finite window, dyadic ledger, and normalized rate row.*
7. *P records* Pkg *provenance over* $\mathsf{RegularCauchyRateTransform}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyModulusComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *data.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransform}^{\uparrow}}$ *row.*

Accepted packets are compared only through M, W, D, R, H, C, P, N. The carrier has no coordinate for quotient stream equality, a selected limit, countable choice, ambient completeness, host real equality, or a terminal $\mathsf{Real}^{\uparrow}$ *equality test.*

Theorem 8294.2 (Regular Cauchy rate transform NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ *carrier* $T = (M, W, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransform}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy modulus rows, finite window, dyadic tolerance ledger, rate-normalization replay, transport, replay, provenance, and local naming;*
2. *source-modulus exposure, requiring every regular-rate request to cite a displayed threshold row in* M*;*
3. *finite-window exposure, requiring the cited threshold to be read through the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*;*
4. *dyadic tolerance compatibility, requiring* D *to contain the target regular tolerance and the comparison budget used for the request;*
5. *rate-normalization replay, requiring* R *to store the finite route from the target regular rate back to the source modulus row;*
6. *componentwise transport, continuation replay, package provenance, and local naming through* H, C, P, N*.*

No NameCert obligation is stored outside these rows, and none of the rows supplies quotient stream equality, selected limits, countable choice, ambient completeness, or host real equality.

Proof. Project the accepted carrier of Definition 8294.1. The tuple exposes exactly the source modulus table, finite window, dyadic tolerance ledger, normalized rate row, transport, replay, provenance, and local name.

The source-modulus and finite-window obligations are immediate from M and W. A regular-rate request can cite only displayed source thresholds, and those thresholds are inspected only through the finite window row.

The dyadic obligation is the row D. It records the target regular tolerance together with the comparison budget that lets the source threshold be read as a regular-Cauchy rate request.

Finally R records the replay from the target rate to the source row, while H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier omits quotient stream equality, selected limits, countable choice, ambient completeness, and host real equality, none of those principles can be exported by the local certificate. □

Theorem 8294.3 (Regular Cauchy rate transform window induction)**.** *Let* $T = (M, W, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ *carrier. Every normalized regular-rate request whose source observations lie in* W *is consumed by finite induction over the displayed window rows: the base row reads the first source threshold from* M *under the dyadic tolerance ledger* D*, and the step appends the next displayed source observation while replaying the same normalization row* R *and structural rows* H, C, P, N*. The induction exports only the rows of* T*.*

Proof. For the base case, read the first displayed observation in W. The associated source threshold is a row of M, the target tolerance and comparison budget are rows of D, and the normalized regular-rate route is recorded by R.

For the induction step, append one further observation from the finite window W. The same carrier rows remain available: M supplies the cited source threshold, D supplies the tolerance budget, and R records the normalization replay.

The structural rows H, C, P, N preserve the packet under componentwise transport, continuation replay, provenance, and local naming. Thus the finite-window induction never leaves

M, W, D, R, H, C, P, N and cannot produce quotient stream equality, a selected limit, countable choice, ambient completeness, or host real equality. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ read is refuted if a target regular-rate request reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ without displaying a source Cauchy modulus row in M, a finite $\mathsf{StreamName}^{\uparrow}$ window in W, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger in D, and the rate-normalization replay row R.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-modulus-comparison: $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source Cauchy modulus rows, finite windows, dyadic tolerance data, rate-normalization replay, transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransform}^{\uparrow}}$ ·

Constructive story: A $\mathsf{RegularCauchyRateTransform}^{\uparrow}$ packet is generated from displayed source Cauchy modulus rows, one StreamName finite window, DyadicRatCore tolerance rows, a rate-normalization replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8294.1, Definition 8294.2, and Definition 8294.3 seed the finite rate-transformation route from arbitrary Cauchy modulus rows to the regular-Cauchy rate discipline consumed by RegSeqRat and Real readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient completeness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-modulus exposure, finite-window exposure, dyadic tolerance compatibility, rate-normalization replay, transport, provenance, and local naming for BEDC.Derived.RegularCauchyRateT

Chapter 8295

A Concrete Naming Certificate for $\mathsf{CauchyRealEmbedding}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{CauchyRealEmbedding}^{\uparrow}$ packet records the general seal route from an accepted regular Cauchy source to the terminal $\mathsf{Real}^{\uparrow}$ interface. It sits after the regular Cauchy carrier of chapter 371, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the stationary rational subcase of chapter 364, and the terminal real-name seal of chapter 38. The packet names a general regular-Cauchy-to-Real handoff; it is not a quotient Cauchy class, a selected host limit, a stream equality theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8295.1 (CauchyRealEmbedding carrier)**.** *A* $\mathsf{CauchyRealEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (G, W, R, D, S, H, C, P, N)$$

with the following visible rows:

1. *G is the accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *source packet whose finite windows, dyadic-radius rows, regular rational readbacks, and real seal are visible to the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff selected for the requested precision reads.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary read through the displayed window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the regularity and comparison rows.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the regular Cauchy source, stream window, regular rational readback, dyadic tolerance ledger, seal row, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the same regular-Cauchy-to-Real route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyRealEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StationaryRationalCauchy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, W, R, D, S, H, C, P, N$. It has no coordinate for quotient Cauchy classes, host sequence equality, selected limits, countable choice, or ambient real completeness.

Theorem 8295.2 (CauchyRealEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealEmbedding}^{\uparrow}$ *carrier* $E = (G, W, R, D, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular Cauchy source, finite StreamName window, regular rational readback, dyadic tolerance ledger, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. The stationary rational route of Definition 364.8 is the constant-source subcase; the general route consumes the same kind of rows without restricting* G *to a stationary rational window.*

Proof. Project the carrier rows. The source row G supplies the accepted $\mathsf{RegularCauchyName}^{\uparrow}$ packet, while W, R, D expose the stream window, regular rational readback, and dyadic tolerance information required before the seal row can be read.

Use the regular Cauchy seal boundary and factorization in chapter 371. They place every $\mathsf{Real}^{\uparrow}$ read downstream of the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows of the source packet.

The stationary rational theorem is then a visible subcase: when the regular Cauchy source is constant, Definition 364.8 identifies the same seal route through the hidden rational seed, constant stream windows, stationary regular rational member, and dyadic ledger. The carrier here keeps the general source row G, so no quotient class, selected host limit, or ambient completeness datum is added. □

Theorem 8295.3 (CauchyRealEmbedding Real seal scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyRealEmbedding}^{\uparrow}$ *packet must read the displayed route*

$$\mathsf{RegularCauchyName}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $G, W, R, D, S, H, C, P, N$ *before the terminal seal is consumed. The route separates the general regular Cauchy embedding from the stationary rational subcase and exports no quotient stream equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 8295.1. The only source-facing row is G, and the only readback rows before the terminal seal are W, R, D.

Apply Definition 8295.2. It requires the regular Cauchy source, finite window, regular rational readback, dyadic tolerance ledger, seal row, transport, replay, provenance, and local naming to be visible in the same packet.

Finally compare the constant-source branch with Definition 364.8. That theorem supplies the stationary rational exactness row, while the present carrier records the general route from any accepted regular Cauchy source. Since no carrier coordinate names quotient stream equality, selected limits, countable choice, or ambient completeness, the seal scope is exactly the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyRealEmbedding}^{\uparrow}$ read is refuted if it consumes a terminal $\mathsf{Real}^{\uparrow}$ seal before displaying the $\mathsf{RegularCauchyName}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: regular-cauchy-name, streamname, regseqrat, dyadicratcore, stationary-rational-cauchy, real: $\mathsf{CauchyRealEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly records the general regular Cauchy source, finite window, regular rational readback, dyadic tolerance ledger, terminal seal, transport, replay, provenance, and local naming route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealEmbedding}^{\uparrow}}$**.**

Constructive story: A CauchyRealEmbedding$^{\uparrow}$ packet is generated from a RegularCauchyNameUp source, StreamNameUp finite window, RegSeqRatUp readback boundary, DyadicRatCoreUp tolerance ledger, RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRealEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 8295.1, Definition 8295.2, and Definition 8295.3 seed the general regular-Cauchy-to-Real seal route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRealEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient Cauchy classes, host sequence equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular Cauchy source exposure, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRealEmbeddingUp.

Chapter 8296

A Concrete Naming Certificate for $\mathsf{MartinBoundary}^{\uparrow}$

Following Martin and Doob on positive harmonic functions and boundary kernels, the $\mathsf{MartinBoundary}^{\uparrow}$ packet records the finite transition-kernel boundary route used by hyperbolic observer-expansion consumers before any host compactification or quotient boundary equality is asserted. It sits beside the hyperbolic metric rows of chapter 3474, the Gromov boundary of chapter 6470, the boundary-shadow packet of chapter 3813, the Poisson-kernel route of chapter 3925, and the terminal real-name seal of chapter 38. The packet names only finite kernel, positive-harmonic, shadow, metric, Real-seal, transport, replay, provenance, and naming rows.

Definition 8296.1 (Martin boundary carrier)**.** *A* $\mathsf{MartinBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, K, H, B, S, P_o, V, R, T, C, G, N)$$

with the following visible rows:

1. *X is the displayed state or basepoint window consumed by the transition-kernel reads.*
2. *K is the finite transition-kernel ledger whose positive entries are inspected by the boundary packet.*
3. *H is the positive-harmonic row read through the kernel ledger.*
4. *B is the* $\mathsf{GromovBoundary}^{\uparrow}$ *handoff row available to hyperbolic consumers.*
5. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *shadow row attached to the same boundary-facing read.*
6. *P_o is the* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *row that turns the boundary and kernel data into finite Real-valued weights.*
7. *V is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric route used by the shadow and kernel comparisons.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for finite kernel weights and normalized comparisons.*
9. *T records componentwise* hsame *transport for the state, kernel, harmonic, boundary, shadow, Poisson-kernel, metric, Real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same finite Martin-boundary read.*
11. *G records* Pkg *provenance over* $\mathsf{MartinBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{GromovBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShad}$ $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

12. N is the local $\mathsf{NameCert}_{\mathsf{MartinBoundary}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, K, H, B, S, P_o, V, R, T, C, G, N$. It has no coordinate for a host compactification, quotient boundary equality, selected minimal harmonic function, countable choice, or ambient real completeness.

Theorem 8296.2 (Martin boundary NameCert obligations)**.** *For every accepted* $\mathsf{MartinBoundary}^\uparrow$ *carrier $M = (X, K, H, B, S, P_o, V, R, T, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MartinBoundary}^\uparrow}$ *surface consists exactly of carrier admission for the displayed state window, transition-kernel ledger, positive-harmonic row, Gromov-boundary handoff, boundary-shadow row, Poisson-kernel row, hyperbolic-metric route, Real seal, componentwise transport, displayed replay, provenance, and local naming. Each boundary-facing read must expose the kernel and positive-harmonic rows before it reaches the shadow, Poisson-kernel, metric, or Real-seal rows.*

Proof. Project the carrier. The state window X, transition-kernel ledger K, and positive-harmonic row H are the source rows for the Martin-boundary read.

The hyperbolic boundary surface is downstream. The rows B, S, V expose the Gromov boundary, boundary shadow, and hyperbolic metric route, while P_o reads the Poisson-kernel weight only after those boundary-facing rows are visible.

The terminal Real seal R and the structural rows T, C, G, N preserve the same finite route. Since the carrier contains no host compactification, quotient boundary equality, selected minimal harmonic function, choice schedule, or ambient completeness row, the listed obligations exhaust the local certificate. □

Theorem 8296.3 (Martin boundary shadow handoff)**.** *Every hyperbolic observer-expansion consumer of an accepted* $\mathsf{MartinBoundary}^\uparrow$ *packet must read the finite route*

$$K, H \longmapsto \mathsf{GromovBoundary}^\uparrow, \mathsf{HyperbolicBoundaryShadow}^\uparrow, \mathsf{HyperbolicPoissonKernel}^\uparrow, \mathsf{HyperbolicMetric}^\uparrow, \mathsf{Real}^\uparrow$$

through $X, K, H, B, S, P_o, V, R, T, C, G, N$. The handoff unifies kernel, positive-harmonic, shadow, Poisson-kernel, metric, and Real-seal rows without asserting a host compactification or equality with any quotient boundary.

Proof. Start from Definition 8296.1. The only boundary-producing source rows are the transition-kernel ledger K and the positive-harmonic row H, attached to the displayed state window X.

Apply Definition 8296.2. It forces the Gromov-boundary handoff B, boundary-shadow row S, Poisson-kernel row P_o, hyperbolic-metric row V, and terminal Real seal R to be read as downstream rows of the same finite packet.

The transport, replay, provenance, and local naming rows T, C, G, N keep the route componentwise. Therefore the observer-expansion consumer receives a finite Martin-boundary handoff, not a compactification theorem, quotient boundary equality, selected minimal harmonic function, countable-choice schedule, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{MartinBoundary}^\uparrow$ read is refuted if it reaches a boundary shadow, Poisson-kernel weight, hyperbolic metric comparison, or terminal Real seal without displaying the transition-kernel ledger, positive-harmonic row, Gromov-boundary handoff, transport, replay, provenance, and local NameCert rows.

Independence witness: hyperbolic-metric, gromov-boundary, hyperbolic-boundary-shadow, hyperbolic-poisson-kernel, real: $\mathsf{MartinBoundary}^\uparrow$ is not bijective to any listed sibling because it jointly carries

transition-kernel, positive-harmonic, boundary, shadow, Poisson-kernel, metric, Real-seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MartinBoundary}\uparrow}$.

Constructive story: A $\mathsf{MartinBoundary}^{\uparrow}$ packet is generated from a state window, transition-kernel ledger, positive-harmonic row, GromovBoundaryUp handoff, HyperbolicBoundaryShadowUp shadow row, HyperbolicPoissonKernelUp row, HyperbolicMetricUp metric route, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartinBoundary}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8296.1, Definition 8296.2, and Definition 8296.3 seed the finite Martin-boundary route for hyperbolic observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MartinBoundary}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host compactification, quotient boundary equality, selected minimal harmonic functions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-kernel exposure, positive-harmonic routing, Gromov-boundary handoff, shadow handoff, Poisson-kernel weighting, hyperbolic-metric routing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.MartinBoundaryUp.

Chapter 8297

A Concrete Naming Certificate for $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$ packet records the finite square-root service available when the positive input is read from a displayed compact interval source rather than from an isolated positive real row. It sits after the one-input positive square-root packet of chapter 8292, the compact finite-cover route of chapter 8290, the dyadic tolerance ledger of chapter 291, the finite stream window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a uniform compact-source bracket service; it is not a host square-root function on an interval, an open-cover compactness theorem, a trichotomy principle, quotient real equality, a selected limit construction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8297.1 (Compact positive real square-root carrier)**.** *A* $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, X, B, D, W, R, H, G, S, V, E, T, C, P, N)$$

with the following visible rows:

1. *I is the compact interval source row whose endpoints and finite subcover route are read through the compact-source interface;*
2. *X is the positive* $\mathsf{Real}^{\uparrow}$ *source row exposed inside the compact interval source;*
3. *B is the dyadic bracket ledger that records a positive lower bracket and an upper bracket for the square-root request on the displayed interval;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the bracket endpoints, interval mesh, and requested precision;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the source observations and bracket windows needed by the request;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational source and square-root approximants;*
7. *H is the* $\mathsf{HeronSquareRoot}^{\uparrow}$ *recurrence row for the bracketed Heron approximants;*
8. *G is the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *regular-Cauchy square-root row reached after the bracket and recurrence rows are displayed;*

9. *S is the* $\mathsf{PositiveRealSquareRoot}^{\uparrow}$ *service row that reads the positive source, bracket, recurrence, and regular-Cauchy data as one local positive-input service;*
10. *V is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the compact source, dyadic bracket, finite windows, regular rational readback, Heron route, regular-Cauchy route, and positive service row are displayed;*
11. *E stores componentwise* hsame *transport for the compact interval, positive source, bracket, tolerance, window, readback, recurrence, regular-Cauchy, positive-service, and seal rows;*
12. *T records the displayed* Cont *replay by which a consumer reads the same finite compact-source square-root service;*
13. *C is the classifier row comparing accepted packets only through the listed compact-source, bracket, readback, service, transport, and replay coordinates;*
14. *P records* Pkg *provenance over* $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$, $\mathsf{PositiveRealSquareRoot}^{\uparrow}$, $\mathsf{CompactInterval}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{HeronSquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}}$ *row.*

The classifier has no coordinate for a total host square-root operator on a compact interval, an open-cover compactness theorem, a trichotomy split on reals, quotient real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8297.2 (Compact positive real square-root NameCert obligations)**.** *For the carrier of Definition 8297.1, the local* $\mathsf{NameCert}_{\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier admission for $Q = (I, X, B, D, W, R, H, G, S, V, E, T, C, P, N)$, compact interval and positive source admission through I, X, dyadic bracket and tolerance accounting through B, D, finite window and regular rational readback through W, R, Heron and regular-Cauchy square-root service rows through H, G, S, V, and componentwise transport, replay, classifier comparison, provenance, and local naming through E, T, C, P, N.*

Proof. Project the carrier rows. The compact source I and positive source X must be displayed before the bracket service is read, so the certificate starts from a compact interval input and a positive real row rather than from an ambient square-root operator.

Next read the numerical rows. The dyadic bracket B and tolerance ledger D expose the finite lower and upper bracket data; the stream windows W and regular rational readback R keep the request finite.

Finally attach the square-root service rows. The Heron row H, regular-Cauchy row G, positive square-root row S, and Real seal V give the bounded service, while E, T, C, P, N transport, replay, compare, source, and name the same service. None of these rows adds open-cover compactness, trichotomy, quotient real equality, selected limits, or ambient completeness. □

Theorem 8297.3 (Compact positive real square-root uniform bracket service)**.** *Every accepted* $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$ *consumer that asks for a square-root value over the displayed compact interval must first read the compact interval source, positive real source, dyadic bracket ledger, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, Heron recurrence row, regular-Cauchy square-root row, positive square-root service row, terminal Real seal, componentwise transport, displayed replay, classifier comparison, provenance, and local naming. The service therefore exports only the finite uniform bracket route over the compact positive source.*

Proof. Begin with the obligations of Definition 8297.2. They force $I, X, B, D, W, R, H, G, S, V, E, T, C, P, N$ to be visible before a consumer can read the compact-source square-root name.

Use the compact-cover source of chapter 8290 together with the dyadic ledger of chapter 291. These rows keep the interval mesh, bracket endpoints, and requested precision finite.

Read the finite observations through chapter 126 and the regular rational readback through chapter 301. The Heron and regular-Cauchy rows then feed the positive square-root service of chapter 8292, and the endpoint is sealed only through chapter 38.

The structural rows E, T, C, P, N preserve that route. Since the carrier contains no coordinate for a host square-root function on an interval, open-cover compactness, trichotomy, quotient real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no consumer can read those objects from the service. □

Falsifiable prediction: Any accepted CompactPositiveRealSquareRootUp service that bypasses the compact interval source, positive Real source, dyadic bracket ledger, StreamName finite windows, RegSeqRat readback, Heron row, regular-Cauchy square-root row, PositiveRealSquareRoot service row, or terminal Real seal refutes the carrier.

Independence witness: positive-real-square-root, dyadic-uniform-cover, dyadicratcore, streamname, regseqrat, real: $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact interval source data, positive source data, dyadic bracket rows, finite windows, readback, Heron and regular-Cauchy service rows, Real sealing, transport, replay, classifier comparison, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}$ packet is generated from a compact interval source, positive Real source, dyadic bracket ledger, tolerance accounting, finite StreamName windows, RegSeqRat readback, Heron recurrence, regular-Cauchy square-root readback, positive square-root service, terminal Real sealing, componentwise transport, displayed replay, classifier comparison, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8297.1, Definition 8297.2, and Definition 8297.3 seed the compact-source uniform bracket service over the displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactPositiveRealSquareRoot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host square-root operator on compact intervals, open-cover compactness, real trichotomy, quotient real equality, selected limits, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-source admission, positive-source guarding, dyadic bracket accounting, finite-window replay, regular rational readback, Heron recurrence accounting, regular-Cauchy square-root readback, Real sealing, transport, provenance, and local naming for BEDC.Derived.CompactPositiveRealSquareRootUp.

Chapter 8298

A Concrete Naming Certificate for $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$

The $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$ packet realizes the observer-phase locality route of chapter 9463 as finite MetaCIC residual-window data. It sits beside the critical-path packet of chapter 765, the normalization frontier of chapter 892, the confluence residual envelope of chapter 3921, the decidability boundary of chapter 749, and the observer phase route of chapter 6920. The packet names a local observer-phase read over residual windows; it does not assert global confluence, subject reduction, beta-conversion decidability, quotient syntax equality, a selected normal form, or kernel consistency.

Definition 8298.1 (MetaCIC observer-phase locality carrier)**.** *A* $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, R, J, B, K, D, A, E, H, T, C, P, N)$$

with the following visible rows:

1. *O is the observer-phase row whose read is local to a displayed MetaCIC audit route;*
2. *R is the finite residual-window row retaining the redex and residual observations that feed the local read;*
3. *J is the local-join row, recording the candidate join available inside the displayed residual window;*
4. *B is the candidate-boundary row separating local candidate evidence from any global normalization or confluence assertion;*
5. *K is the bounded checker row consumed through the decidability boundary;*
6. *D is the obstruction-retention row that keeps failed subject-reduction, confluence, or conversion reads visible rather than erased;*
7. *A is the adjacent critical-path row through which candidate-mediated rows, residual budget, bounded checker rows, and L10 faces remain auditable;*
8. *E stores componentwise* hsame *transport for the observer-phase, residual-window, local-join, candidate-boundary, checker, obstruction, and critical-path rows;*
9. *H records the displayed* Cont *replay by which a consumer reads the same local observer-phase route;*

10. *T is the taste-facing instance slot reserved for later Nontrivial and FieldFaithful evidence;*
11. *C is the classifier row comparing accepted packets only through the listed local rows;*
12. *P records* Pkg *provenance over* $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFron}$ $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetacicObserverPhaseLocality}^{\uparrow}}$ *row.*

The classifier has no coordinate for global confluence, subject reduction, beta-conversion decidability, quotient term equality, selected normal forms, or kernel consistency.

Theorem 8298.2 (MetaCIC observer-phase locality NameCert obligations)**.** *For the carrier of Definition 8298.1, the local* $\mathsf{NameCert}_{\mathsf{MetacicObserverPhaseLocality}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier admission for $L = (O, R, J, B, K, D, A, E, H, T, C, P, N)$, observer-phase and residual-window admission through O, R, local-join and candidate-boundary separation through J, B, bounded checker and obstruction-retention accounting through K, D, critical-path transport and replay through A, E, H, and taste-facing instance, classifier, provenance, and local naming through T, C, P, N.*

Proof. Project the carrier rows. The observer-phase row O is local because it can be read only with the residual window R, not as an unscoped phase surface.

Next attach the MetaCIC rows. The local join J, candidate boundary B, bounded checker K, and obstruction-retention row D keep the route inside the residual-window envelope while the critical-path row A exposes the candidate-mediated and L10-facing dependencies already present in the surrounding MetaCIC surface.

Finally read E, H, T, C, P, N. Transport and replay preserve the same finite route, the taste-facing slot records that later Lean TasteGate work must provide Nontrivial and FieldFaithful instances, and the classifier, provenance, and local NameCert rows add no global confluence, subject reduction, conversion decider, quotient syntax equality, selected normal form, or kernel-consistency theorem. □

Falsifiable prediction: Any accepted MetacicObserverPhaseLocalityUp packet that exports observer-phase locality without the residual window, local join, candidate boundary, obstruction-retention row, bounded checker, critical-path dependency, transport, replay, provenance, and local NameCert rows refutes the carrier.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-confluence-residual-envelope, metacicdecidableboundary: $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries observer-phase locality, residual windows, local joins, candidate-boundary separation, obstruction retention, bounded checker rows, critical-path dependency, taste-facing instance slots, transport, replay, classifier comparison, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicObserverPhaseLocality}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicObserverPhaseLocality}^{\uparrow}$ packet is generated from observer-phase, residual-window, local-join, candidate-boundary, bounded-checker, obstruction-retention, critical-path, taste-facing instance, transport, replay, classifier, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicObserverPhaseLocality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8298.1 and Definition 8298.2 seed the observer-phase locality packet over displayed MetaCIC residual windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicObserverPhaseLocality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global confluence, subject reduction, beta-conversion decidability, quotient syntax equality, selected normal forms, Lean verification, bridge checking, or kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, residual-window locality, local-join accounting, candidate-boundary separation, bounded checker routing, obstruction retention, critical-path dependency, Nontrivial and FieldFaithful TasteGate instances, transport, provenance, and local naming for BEDC.Derived.MetacicObserverPhaseLocalityUp.

Chapter 8299

A Concrete Naming Certificate for UniformlyCauchyRealSequence$^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley's sequential viewpoint, the UniformlyCauchyRealSequ packet records a real-valued uniformly Cauchy sequence as finite NameCert data. It refines the generic schedule of chapter 3696 by requiring each inspected window to be read through the StreamName$^{\uparrow}$ surface of chapter 126, the regular rational readback of chapter 301, and the Real$^{\uparrow}$ seal of chapter 38. The packet supplies an L10 Real and Completion consumer edge; it does not assert quotient streams, selected representatives, selected limits, countable choice, or ambient real completeness.

Definition 8299.1 (Uniformly Cauchy real sequence carrier)**.** *A* UniformlyCauchyRealSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, R, D, M, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the real sequence source row, naming the displayed real-valued stream whose finite observations are admitted by the packet.*
2. *W is the* StreamName$^{\uparrow}$ *window row, exposing the finite indices and comparison windows used by a uniform Cauchy request.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational data visible in those windows.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, storing the precision budget used to compare late real-name windows.*
5. *M is the* CauchyConvergenceModulus$^{\uparrow}$ *row, recording one displayed tail threshold for each requested dyadic tolerance.*
6. *L is the* Real$^{\uparrow}$ *seal row, available only after S, W, R, D, M have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, readback, tolerance, modulus, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-sequence request through $S, W, R, D, M, L, H, P, N$.*
9. *P is the* Pkg *provenance over* UniformlyCauchyRealSequence$^{\uparrow}$, UniformlyCauchySequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformlyCauchyRealSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, R, D, M, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, hidden cofinal subsequences, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8299.2 (Uniformly Cauchy real sequence NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyCauchyRealSequence}^{\uparrow}$ *carrier*

$$U = (S, W, R, D, M, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformlyCauchyRealSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the real sequence source S,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule M,* $\mathsf{Real}^{\uparrow}$ *seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 8299.1. The analytic rows are the real sequence source S, finite windows W, regular rational readback R, dyadic tolerance ledger D, convergence modulus M, and Real seal L.

The uniformity obligation is read from W, D, M. For a requested tolerance, M supplies one displayed tail threshold, W supplies the finite windows at and beyond that threshold, and D supplies the dyadic comparison budget used by those windows.

The real-valued readback is downstream of the same finite route. The row R records regular rational data for the displayed windows, while L seals the corresponding real-name boundary only after the source, windows, readback, tolerance, and modulus rows are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded quotient, choice, selected-representative, selected-limit, host-equality, or ambient-completeness coordinate occurs in U, the listed rows exhaust the local NameCert obligations. □

Theorem 8299.3 (Uniformly Cauchy real sequence RegSeqRat route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{UniformlyCauchyRealSequence}^{\uparrow}$ *packet factors through*

$$S \longmapsto W \longmapsto R, D, M \longmapsto L \longmapsto H, C, P, N.$$

Consequently RealSequentialCompleteness and EffectiveCauchyCompletion consumers may use the displayed source row, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *tail schedule, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local structural rows, but they receive no quotient stream, selected representative, selected limit, hidden cofinal tail, countable choice, host equality, or ambient real completeness.*

Proof. Start with the source and window rows. The row S names the real-valued stream, and W gives only finite observations of that stream for the requested late-window comparison.

Next read R, D, M. The regular rational readback row R supplies the rational data visible in the selected windows, D supplies the dyadic tolerance ledger, and M supplies the single tail threshold required by the uniform Cauchy request.

Only after those rows have been displayed does the packet expose L as a $\mathsf{Real}^{\uparrow}$ seal. Thus a real-name or completion consumer reaches the seal through the same finite schedule rather than through a quotient stream or a selected limit.

The structural rows H, C, P, N preserve the route under componentwise transport, displayed replay, package provenance, and local naming. Since the carrier has no coordinate for the excluded objects, no downstream consumer can bypass the StreamName, RegSeqRat, DyadicRatCore, modulus, and Real-seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyCauchyRealSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyCauchyRealSequence}^{\uparrow}$ packet is generated from a real sequence source row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyConvergenceModulus tail schedule, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyCauchyRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8299.1, Definition 8299.2, and Definition 8299.3 seed the finite real-valued uniformly Cauchy sequence surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyCauchyRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, selected limits, hidden cofinal tails, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, regular rational readback, dyadic tolerance confinement, convergence-modulus exactness, Real seal nonescape, transport stability, continuation replay, and provenance exhaustion for BEDC.Derived.UniformlyCauchyRealSequenceUp.

Chapter 8300

A Concrete Naming Certificate for $\mathsf{NestedClosedSets}^{\uparrow}$

Following Bishop and Bridges on constructive complete metric spaces and nested closed-set principles, the $\mathsf{NestedClosedSets}^{\uparrow}$ packet records the finite BEDC surface for a decreasing inhabited located closed-set chain with an explicit diameter modulus. It sits beside the complete-metric carrier of chapter 128, the nested closed-ball route of chapter 3526, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular-rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing route of chapter 1506. The packet names a displayed nested closed-set certificate only; it does not choose points from the sets, quotient equal real names, invoke countable choice, import open-cover compactness, or appeal to ambient $\mathsf{Real}^{\uparrow}$ or metric completeness.

Definition 8300.1 (Nested closed sets carrier)**.** *A* $\mathsf{NestedClosedSets}^{\uparrow}$ *carrier over a displayed complete-metric source is a finite* BHist *packet*

$$Z = (M, F, I, D, B, W, Q, A, H, C, P, N)$$

with the following rows:

1. *M is the consumed* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, exposed only as the metric context in which the closed-set ledgers are read.*
2. *F is the decreasing located closed-set ledger; each displayed level is inhabited, located, and closed by local rows.*
3. *I records the monotone inclusion comparisons $F_{n+1} \subseteq F_n$ as finite classifier data.*
4. *D is the diameter ledger and B is the modulus row forcing the displayed diameters below each requested dyadic tolerance.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window used to read the requested levels of the chain.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic diameter and tolerance observations.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal exposed to downstream real-name consumers after the finite windows and regular-rational rows have been displayed.*
8. *H records componentwise* hsame *transport for M, F, I, D, B, W, Q, A and the local naming rows.*
9. *C records displayed* Cont *replay from the metric source through closed-set inclusion, diameter control, readback, and real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{NestedClosedSets}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedClosedSets}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, I, D, B, W, Q, A, H, C, P, N$. It has no coordinate for countable choice, selected points in the closed sets, quotient equality of reals or streams, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a host complete-metric theorem.*

Theorem 8300.2 (Nested closed sets NameCert obligations)**.** *For every accepted* $\mathsf{NestedClosedSets}^{\uparrow}$ *carrier*

$$Z = (M, F, I, D, B, W, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedClosedSets}^{\uparrow}}$ *surface consists exactly of complete-metric source exposure through M, decreasing inhabited located closed-set rows through F, inclusion comparisons through I, diameter and modulus rows through D, B, finite observation through W, regular-rational readback through Q, Real sealing through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8300.1. The source side is the displayed complete-metric row M together with the finite closed-set, inclusion, diameter, and modulus ledgers F, I, D, B.

The readback side is reached only through W, Q, A. The observation window W displays the finite levels under inspection, Q reads the dyadic diameter and tolerance data as regular-rational rows, and A seals the downstream real-name observation after those rows are present.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no selected point, quotient equality, countable choice principle, open-cover compactness row, or ambient completeness theorem is a carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 8300.3 (Nested closed sets diameter handoff)**.** *Every completion-facing or real-name consumer of an accepted* $\mathsf{NestedClosedSets}^{\uparrow}$ *packet must read the diameter handoff through*

$$\mathsf{CompleteMetric}^{\uparrow}, \mathsf{NestedClosedBall}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}$$

before it consumes the terminal row A. The handoff preserves the decreasing located closed-set rows F, inclusion ledger I, diameter ledger D, modulus row B, and structural rows H, C, P, N; it exports no selected point, quotient equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or ambient metric completeness.*

Proof. Start with the carrier projection. The only diameter data are D and B, and both are indexed by the displayed closed-set chain F and inclusion comparisons I over the complete-metric source M.

The nested closed-ball sibling supplies the local ball-style reading of the same shrinking surface, while $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ provide the tolerance, window, and readback rows. The Real seal A is therefore downstream of a finite observation route rather than an ambient completeness assertion.

Finally use H, C, P, N. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row keep the handoff inside the packet and record the $\mathsf{MetricCompletion}^{\uparrow}$ consumer route. No row remains from which a point selector, quotient equality, countable choice, open-cover compactness, or ambient completeness principle can be read. □

Falsifiable prediction: A consumer requesting a nested closed-set completion witness from $\mathsf{NestedClosedSets}^{\uparrow}$ must display decreasing inhabited located closed-set rows, inclusion rows, diameter and modulus rows, finite StreamName windows, RegSeqRat readback, a Real seal, transport, replay, provenance, and local NameCert rows before any terminal handoff is accepted.

Independence witness: completemetric, nestedclosedball, dyadicratcore, streamname, regseqrat, real, metriccompletion: $\mathsf{NestedClosedSets}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a decreasing located closed-set chain, inhabitedness, inclusion, diameter decay, finite readback, Real sealing, completion-facing handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedClosedSets}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedClosedSets}^{\uparrow}$ packet is generated from a CompleteMetricUp source, decreasing inhabited located closed-set rows, finite inclusion rows, diameter and modulus rows, StreamName windows, RegSeqRat readback, a Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedClosedSets}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8300.1, Definition 8300.2, and Definition 8300.3 seed the finite nested closed-set route from decreasing located closed sets to completion-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedClosedSets}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected intersection points, quotient equality of real names or streams, countable choice, open-cover compactness, ambient Real completeness, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, decreasing-set inclusion, inhabited locatedness, diameter-modulus handoff, readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.NestedClosedSetsUp.

Chapter 8301

A Concrete Naming Certificate for LindenbaumAlgebra$^{\uparrow}$

Following the Lindenbaum-Tarski algebra for first-order logic, the LindenbaumAlgebra$^{\uparrow}$ packet records formula syntax modulo displayed provable-equivalence evidence as finite NameCert data. It sits beside the syntax-and-deduction surface of chapter 197, the satisfaction-facing carrier of chapter 198, the term-and-substitution surface of chapter 200, and the MetaCIC route boundary of chapter 765. The packet gives MetaCIC a finite algebraic consumer route for formula rows without importing host quotient carriers, Boolean prime ideals, maximal theories, completeness, or choice principles.

Definition 8301.1 (Lindenbaum algebra carrier)**.** *A* LindenbaumAlgebra$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, D, E, B, S, K, H, C, P, N)$$

with the following displayed rows:

1. *F is the* FirstOrder$^{\uparrow}$ *formula-source row, naming the finite formulas admitted by the packet.*
2. *D is the deduction row, recording the finite proof traces whose conclusions are visible to the packet.*
3. *E is the provable-equivalence ledger, containing pairs of formula rows together with displayed derivations in both directions.*
4. *B is the finite Boolean-operation row for negation, conjunction, disjunction, implication, top, and bottom on admitted formulas.*
5. *S is the substitution row, read through the* LambdaCalc$^{\uparrow}$ *term-and-substitution boundary when term substitution is required.*
6. *K is the MetaCIC consumer-route row, recording when the syntax algebra may be passed to* MetaCICCriticalPath$^{\uparrow}$ *as finite formula data.*
7. *H records componentwise* hsame *transport for the formula, deduction, equivalence, operation, substitution, consumer-route, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a syntax-algebra request through $F, D, E, B, S, K, H, P, N$.*
9. *P is the* Pkg *provenance over* LindenbaumAlgebra$^{\uparrow}$*,* FirstOrder$^{\uparrow}$*,* ModelTheory$^{\uparrow}$*,* LambdaCalc$^{\uparrow}$*,* MetaCICCriticalPath$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LindenbaumAlgebra}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, D, E, B, S, K, H, C, P, N$. It has no coordinate for a host quotient set of formulas, a selected representative of an equivalence class, a Boolean prime ideal, a maximal consistent theory, a completeness theorem, semantic truth in all structures, or Classical choice.

Theorem 8301.2 (Lindenbaum algebra NameCert obligations)**.** *For every accepted* $\mathsf{LindenbaumAlgebra}^{\uparrow}$ *carrier*

$$L = (F, D, E, B, S, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LindenbaumAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{FirstOrder}^{\uparrow}$ *formula row F, deduction exposure through D, provable-equivalence exposure through E, finite Boolean-operation exposure through B, substitution exposure through S, MetaCIC consumer routing through K, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 8301.1. The formula, deduction, provable-equivalence, Boolean-operation, substitution, and consumer-route data are precisely the rows F, D, E, B, S, K.

The NameCert read cannot replace these rows by a host algebra. The equivalence ledger E stores displayed derivations in both directions for named formula pairs, while B stores only finite operation rows on admitted formulas.

Substitution and MetaCIC consumption are likewise explicit rows. The row S keeps term substitution inside the displayed $\mathsf{LambdaCalc}^{\uparrow}$ boundary, and K records the route by which a MetaCIC consumer receives formula data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded quotient carrier, representative selection, prime ideal, maximal theory, completeness witness, semantic-totality row, or choice principle occurs in L, the listed rows exhaust the local NameCert obligations. □

Theorem 8301.3 (Lindenbaum algebra substitution stability)**.** *In an accepted* $\mathsf{LindenbaumAlgebra}^{\uparrow}$ *packet, every substitution read factors through the displayed route*

$$F, D, E \longmapsto S \longmapsto H, C, P, N.$$

If two formula rows are related by the provable-equivalence ledger E, then any admitted substitution instance may be compared only after the same equivalence evidence has been transported through the substitution row S and the componentwise hsame *row H.*

Proof. Begin with the equivalence ledger. A comparison of formula rows is visible only when E contains the two displayed derivations connecting the formulas.

The substitution row S is downstream of that ledger. It may form a substituted formula row only from the admitted formula source F and the displayed substitution boundary, so it cannot silently quotient formulas or choose a representative.

Transport is then componentwise. The row H moves the formula, equivalence, and substitution rows together, while C, P, N replay, source, and name that same route. Thus substitution stability is a property of the finite displayed ledger, not of an ambient equality of host formula classes. □

Theorem 8301.4 (Lindenbaum algebra finite Boolean operations)**.** *Every Boolean operation exported by an accepted* $\mathsf{LindenbaumAlgebra}^{\uparrow}$ *packet is read from the finite operation row B together with the visible formula and provable-equivalence rows F, D, E. In particular, a consumer may use*

negation, conjunction, disjunction, implication, top, and bottom only as displayed operations on admitted formula rows, and no consumer receives a completed Boolean algebra of host equivalence classes.

Proof. Inspect the operation row B. Its entries name the operation kind and the admitted input and output formula rows, all of which are read through F.

Compatibility with provable equivalence is checked only through D and E. The deduction row gives the finite proof traces, and the equivalence ledger records the bidirectional derivations needed to compare operation outputs.

The structural rows then preserve this operation surface under transport and replay. Because the carrier contains no quotient set, ideal, ultrafilter, maximal theory, or semantic completion row, finite Boolean operations remain displayed syntax operations rather than a host Boolean algebra. □

Theorem 8301.5 (Lindenbaum algebra MetaCIC consumer route)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer of an accepted* $\mathsf{LindenbaumAlgebra}^{\uparrow}$ *packet factors through*

$$\mathsf{FirstOrder}^{\uparrow},\ \mathsf{ModelTheory}^{\uparrow},\ \mathsf{LambdaCalc}^{\uparrow},\ F, D, E, B, S, K, H, C, P, N.$$

The consumer receives finite formula rows, deduction traces, provable-equivalence evidence, Boolean-operation rows, substitution rows, and local NameCert provenance; it does not receive a host quotient carrier, semantic completeness theorem, Boolean prime ideal, maximal theory, or Classical-choice selection function.

Proof. The route begins with the first-order syntax row F and the deduction row D. These rows supply the formula endpoints and proof traces consumed by the syntax algebra.

The algebraic information is displayed next. The equivalence ledger E records bidirectional provability evidence, and the operation row B records finite Boolean syntax operations on admitted formulas.

Substitution and MetaCIC routing are separate downstream rows. The row S keeps substitution inside the displayed lambda-calculus or term boundary, while K is the only row that exports the packet to $\mathsf{MetaCICCriticalPath}^{\uparrow}$.

The remaining rows H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Therefore the MetaCIC consumer sees exactly the finite syntax-algebra surface and cannot infer quotient-class, maximality, completeness, or choice data from the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LindenbaumAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{LindenbaumAlgebra}^{\uparrow}$ packet is generated from FirstOrder formula rows, finite deduction traces, provable-equivalence ledger rows, finite Boolean-operation rows, substitution rows, a MetaCIC consumer-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LindenbaumAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8301.1, Definition 8301.2, Definition 8301.3, Definition 8301.4, and Definition 8301.5 seed the finite provable-equivalence syntax-algebra surface for MetaCIC consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LindenbaumAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host quotient carriers, selected representatives, Boolean prime ideals, maximal theories, semantic completeness, Classical choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deduction exposure, provable-equivalence exactness, finite Boolean-operation confinement, substitution stability, MetaCIC route nonescape, transport stability, continuation replay, and provenance exhaustion for BEDC.Derived.LindenbaumAlgebraUp.

Chapter 8302

A Concrete Naming Certificate for $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$ packet records the finite source-level equality-modulus route by which two regular Cauchy rational names are compared before a $\mathsf{RealClassifier}^{\uparrow}$ consumer reads an equality boundary. It sits between the classifier surface of chapter 1620, the regular rational schedules of chapter 301, the finite windows of chapter 126, the tolerance ledgers of chapter 291, the regular-Cauchy equivalence boundary of chapter 1404, the terminal real seal of chapter 38, and the Real-facing equality-modulus route of chapter 7601. The chapter names the source equality-modulus row before any quotient stream equality or ambient completeness principle could be mistaken for classifier data.

Definition 8302.1 (Regular Cauchy equality-modulus carrier)**.** *A* $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, S_X, S_Y, W, D, E, C, H, K, P, N)$$

with the following displayed rows:

1. *X and Y are the two source regular Cauchy names whose equality-modulus route is being exposed.*
2. *S_X and S_Y are the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules attached to X and Y.*
3. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *finite window through which both schedules are inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, recording the displayed radii, split budgets, and comparison cells for the equality request.*
5. *E is the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row that records the source-level regular-Cauchy equivalence handoff visible before terminal real classification.*
6. *C is the* $\mathsf{RealClassifier}^{\uparrow}$ *readback row that consumes the displayed source route without replacing it by host equality.*
7. *H stores componentwise* hsame *transport for the two names, schedules, common window, dyadic ledger, equivalence row, classifier row, replay, provenance, and local name.*
8. *K stores displayed* Cont *replay of the route*

$$X, Y \longmapsto S_X, S_Y \longmapsto W \longmapsto D \longmapsto E \longmapsto C.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$, $\mathsf{RealClassifier}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityModulus}^{\uparrow}}$ *row.*

The packet compares accepted carriers only through $X, Y, S_X, S_Y, W, D, E, C, H, K, P, N$. It has no coordinate for host real equality, quotient stream equality, trichotomy, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8302.2 (Regular Cauchy equality-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$ *carrier*

$$M = (X, Y, S_X, S_Y, W, D, E, C, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityModulus}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the paired source names, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules, common* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row,* $\mathsf{RealClassifier}^{\uparrow}$ *readback, transport, replay, provenance, and local naming;*
2. *common-window stability, requiring both schedules to be read through the same finite window W before any equality-modulus answer is consumed;*
3. *dyadic tolerance exactness, requiring the ledger D to display the radii, split budgets, and comparison cells used by the equality request;*
4. *regular-equivalence handoff, requiring the source route to pass through E before a Real-facing classifier row is read;*
5. *RealClassifier nonescape, requiring the classifier row C to consume only the displayed source route and not a host equality or quotient stream equality;*
6. *ledger exhaustion, requiring H, K, P, N to transport, replay, source, and name only the displayed finite packet.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8302.1. The accepted packet displays exactly the two source names, two schedules, common window, dyadic ledger, equivalence row, classifier row, and structural rows.

The common-window and dyadic clauses are read from W and D. Since both schedules are inspected through W and the equality request uses the displayed radii and split budgets of D, the equality-modulus route has no independent hidden window or tolerance source.

The regular-equivalence handoff is the row E, and the classifier readback is the row C. Thus a Real-facing consumer reaches the classifier only after the source regular-Cauchy equivalence boundary has been displayed.

Finally the structural rows H, K, P, N transport, replay, source, and locally name the same packet. Because the carrier omits host real equality, quotient stream equality, trichotomy, countable choice, selected limits, and ambient completeness, none of those principles can be exported by the local NameCert surface. □

Theorem 8302.3 (Regular Cauchy equality-modulus classifier route)**.** *Every* $\mathsf{RealClassifier}^{\uparrow}$*-facing equality read exported by an accepted* $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyEquivalence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{hsame} \longmapsto \mathsf{Cont} \mapsto$$

In carrier notation, the read consumes S_X, S_Y, then W, then D, then E, then the Real-facing classifier boundary over X, Y, and finally the structural rows H, K, P, N. It exports a source-level equality-modulus certificate, not host real equality, quotient stream equality, trichotomy, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $M = (X, Y, S_X, S_Y, W, D, E, C, H, K, P, N)$. Projection exposes the two $\mathsf{RegSeqRat}^{\uparrow}$ schedules S_X, S_Y before any classifier-facing equality read is available.

Next read the common $\mathsf{StreamName}^{\uparrow}$ window W and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D. These rows fix the finite requests and the dyadic tolerance data for the equality-modulus route.

The source equivalence boundary is then the row E. The classifier row C is downstream of E, so a $\mathsf{RealClassifier}^{\uparrow}$ consumer cannot replace the source route by a direct equality test on terminal real data.

Finally consume H, K, P, N. They provide componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming for exactly the same finite packet, and the carrier has no coordinate for the excluded equality, choice, order, selection, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEqualityModulus}^{\uparrow}$ packet is generated from two source regular Cauchy names, paired RegSeqRat schedules, one common StreamName window, DyadicRatCore tolerance data, a RegularCauchyEquivalence row, a RealClassifier readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8302.1, Definition 8302.2, and Definition 8302.3 seed the source-level equality-modulus route from paired regular rational schedules to the RealClassifier boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEqualityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient stream equality, trichotomy, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window stability, dyadic tolerance exactness, regular-equivalence handoff, RealClassifier nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEqualityModulusUp.

Chapter 8303

A Concrete Naming Certificate for $\mathsf{FiniteContractionOrbitError}^{\uparrow}$

Following Banach fixed-point theory and Bishop–Bridges constructive analysis, a $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ packet records the finite Picard-orbit error budget used before a contraction route is sealed for Real-facing consumers. It sits beside the contraction data of chapter 3529, the Cauchy orbit route of chapter 2350, and the iterator packet of chapter 6020. The packet gives downstream fixed-point and Picard chapters a finite non-choice error certificate rather than an ambient completeness or selected-limit assertion.

Definition 8303.1 (Finite contraction orbit error carrier)**.** *A* $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, X, F, W, D, Q, E_0, H, C, P, N)$$

with the following displayed rows:

1. *K is the contraction-modulus and contractive-map row read from* $\mathsf{BanachContraction}^{\uparrow}$ *data.*
2. *X is the metric carrier and starting-point row for the finite orbit window.*
3. *F is the self-map replay row for the displayed finite Picard iterates.*
4. *W is the finite iterate window that records the source iterate, terminal iterate, and observed step bound.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row carrying the dyadic tolerance and finite arithmetic ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{CauchyContractionOrbit}^{\uparrow}$ *route that turns the finite orbit window into a regular Cauchy readback.*
7. *E_0 is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed dyadic and regular-sequence route.*
8. *H records componentwise* hsame *transports for the contraction, orbit, budget, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from contraction data through the finite window, dyadic budget, Cauchy route, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteContractionOrbitError}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{CauchyContractionOrbit}^{\uparrow}$, $\mathsf{ContractiveCauchyIterator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteContractionOrbitError}^{\uparrow}}$ *row.*
The carrier has no coordinate for a selected fixed point, quotient orbit equality, host completeness proof, trichotomy comparison, or hidden Real error estimate.

Lean4 (statement) `BEDC.Derived.FiniteContractionOrbitErrorUp`

Theorem 8303.2 (Finite contraction orbit error NameCert obligations)**.** *For every accepted* $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ *carrier*

$$E = (K, X, F, W, D, Q, E_0, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteContractionOrbitError}^{\uparrow}}$ *surface consists of exactly five finite rows of obligation:*

1. *carrier admission for the displayed packet $E = (K, X, F, W, D, Q, E_0, H, C, P, N)$;*
2. *contraction and finite-orbit exposure through K, X, F, W;*
3. *dyadic error-budget exactness through D;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{CauchyContractionOrbit}^{\uparrow}$ *readback through Q followed by the* $\mathsf{Real}^{\uparrow}$ *seal row E_0;*
5. *componentwise transport, replay, provenance, local naming, and non-escape through H, C, P, N.*

Proof. Project the carrier of Definition 8303.1. The rows K, X, F, W are the only contraction and finite-orbit rows, D is the only dyadic budget row, Q is the only Cauchy and regular-sequence readback row, and E_0 is the only Real-facing seal.

The structural rows H, C, P, N preserve, replay, source, and name those displayed rows. Since the carrier contains no selected fixed point, quotient orbit equality, host completeness proof, trichotomy comparison, or hidden Real error estimate, the five listed duties exhaust the local NameCert surface. □

Theorem 8303.3 (Finite contraction orbit error Cauchy route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ *packet reaches the error budget only through the displayed route*

$$\mathsf{BanachContraction}^{\uparrow},\ \mathsf{CauchyContractionOrbit}^{\uparrow},\ \mathsf{ContractiveCauchyIterator}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In carrier coordinates this route is $K, X, F, W, D, Q, E_0, H, C, P, N$, and no consumer can substitute ambient completeness, a selected limit, quotient orbit equality, or an unrecorded Real comparison for that route.

Proof. Start with an accepted carrier $E = (K, X, F, W, D, Q, E_0, H, C, P, N)$. The contraction data are read through K, X, F, W, so the finite orbit window is visible before any error budget is exported.

Read the budget row next. The only error-budget coordinate is D, and it is a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger tied to the same finite iterate window.

Read the Cauchy and Real-facing rows. The row Q carries the $\mathsf{CauchyContractionOrbit}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and E_0 seals the displayed read through $\mathsf{Real}^{\uparrow}$.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same packet; they do not add selected limits, quotient equality, ambient completeness, or hidden comparison data. Therefore the error budget reaches Real-facing consumers only through the displayed Cauchy, regular-sequence, dyadic, and Real route. □

Falsifiable prediction: A candidate packet that omits either the finite iterate window W, the dyadic budget row D, or the Cauchy readback row Q cannot export the same $\mathsf{Real}^{\uparrow}$ seal row E_0 as an accepted $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ packet.

Independence witness: banach-contraction, cauchy-contraction-orbit, contractive-cauchy-iterator, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ because this packet jointly carries contraction data, a finite iterate window, dyadic error budget, regular Cauchy readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteContractionOrbitError}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteContractionOrbitError}^{\uparrow}$ packet is generated from contraction data, a metric starting row, self-map replay, a finite iterate window, a dyadic error budget, RegSeqRat and Cauchy contraction-orbit readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteContractionOrbitError}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8303.1, Definition 8303.2, and Definition 8303.3 seed the finite contraction-orbit error packet and its Cauchy, dyadic, regular-sequence, and Real-facing route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteContractionOrbitError}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Banach fixed-point existence, selected limits, ambient completeness, quotient orbit equality, trichotomy comparison, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-window exposure, dyadic budget exactness, Cauchy and RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteContractionOrbitErrorUp.

Chapter 8304

A Concrete Naming Certificate for $\mathsf{LocatedSignApartness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedSignApartness}^{\uparrow}$ packet records guarded located sign reads for displayed real names at the L10 Real boundary. It sits beside the located-apartness real bridge of chapter 3564, the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite guarded sign-read certificate: it is not trichotomy, not total sign on $\mathsf{Real}^{\uparrow}$, not quotient real equality, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8304.1 (Located sign apartness carrier)**.** *A* $\mathsf{LocatedSignApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, A, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *endpoint row whose sign is being read only under a displayed guard.*
2. *A is the located-apartness real bridge row that supplies the positive separation guard from the zero row or from a named comparison endpoint.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger carrying the positive dyadic radius, tolerance arithmetic, and sign-side comparison data.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window on which the sign read is inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite window.*
6. *E is the real-seal boundary row reached after A, D, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the real row, apartness guard, dyadic ledger, window, readback, seal boundary, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the guarded request through threshold inspection, finite-window readback, regular-rational readback, real-seal handoff, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedSignApartness}^{\uparrow}$*, the located-apartness real bridge,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed guard, threshold, finite-window, readback, seal, transport, replay, provenance, and local naming rows. It contains no coordinate for deciding the sign of an arbitrary $\mathsf{Real}^{\uparrow}$ *name.*

Theorem 8304.2 (Located sign apartness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSignApartness}^{\uparrow}$ *carrier* $S = (R, A, D, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located-apartness guard exposure through* A*, dyadic threshold exactness through* D*, finite window exposure through* W*, regular rational readback through* Q*, Real seal boundary exposure through* E*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8304.1. The accepted packet exposes exactly the rows $R, A, D, W, Q, E, H, C, P, N$, so the local certificate begins as finite packet admission rather than a global order principle on constructive reals.

Read the guard and threshold rows next. The located-apartness bridge row A supplies the positive guard, while the dyadic row D supplies the displayed radius and sign-side comparison ledger. The finite window W and regular rational readback Q are the only observations from which the guarded sign read is formed.

Finally use the structural rows. The seal row E is reached only after the guard, threshold, window, and readback rows have been displayed, and H, C, P, N preserve, replay, source, and name exactly that route. Since no row stores trichotomy, total sign, quotient real equality, or completeness, those principles are not local NameCert obligations. □

Theorem 8304.3 (Located sign apartness threshold window)**.** *Every permitted guarded sign read of an accepted* $\mathsf{LocatedSignApartness}^{\uparrow}$ *packet factors through a single displayed threshold-window route*

located-apartness guard A, $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold* D, $\mathsf{StreamName}^{\uparrow}$ *window* W, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q.

The route authorizes a positive or negative sign read only relative to the displayed guard and threshold. It does not decide zero versus nonzero for arbitrary $\mathsf{Real}^{\uparrow}$ *names.*

Proof. Start from the guard row A. A sign read is admitted only when the located-apartness bridge has already supplied the positive separation data needed to ask for a side of zero or of the named comparison endpoint.

Next inspect the threshold and window rows. The dyadic ledger D fixes the positive radius and comparison arithmetic, while W fixes the finite stream observations and Q fixes their regular rational readback.

The exported sign is therefore scoped by those four rows. Without A there is no apartness guard, without D there is no positive threshold, and without W, Q there is no finite readback. The packet consequently refuses an unguarded total sign function on $\mathsf{Real}^{\uparrow}$. □

Theorem 8304.4 (Located sign apartness Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedSignApartness}^{\uparrow}$ *packet must read the real-seal boundary* E *only after the located-apartness guard* A*, dyadic threshold ledger* D*, finite stream window* W*, regular rational readback* Q*, transport row* H*, replay row* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}$ *row* N *have been displayed. The boundary exports a guarded sign-read certificate, not quotient real equality, trichotomy, total sign on* $\mathsf{Real}^{\uparrow}$*, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8304.2 to expose the local certificate rows. The Real-facing seal is one row of the same packet, not a detached host-real endpoint.

Apply Definition 8304.3. The sign read reaches the seal only through the located-apartness guard, dyadic threshold, finite window, and regular rational readback rows.

The remaining structural rows keep the boundary stable. Componentwise transport, replay, provenance, and local naming preserve the same displayed packet through H, C, P, N. Because the carrier has no field for quotient equality, trichotomy, total sign, or completeness, none of those objects can be exported at the Real seal boundary. □

Falsifiable prediction: A consumer refutes a $\mathsf{LocatedSignApartness}^{\uparrow}$ packet if it exports a sign read without displaying the located-apartness guard, dyadic threshold ledger, finite stream window, regular rational readback, real-seal boundary, transport, replay, provenance, and local NameCert rows.

Independence witness: located-apartness-real-bridge, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{LocatedSignApartness}^{\uparrow}$ because only this packet binds the located-apartness guard, dyadic threshold ledger, finite observation window, regular rational readback, Real seal boundary, transport, replay, provenance, and local naming into one guarded sign-read carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSignApartness}^{\uparrow}$ packet is generated from a sealed Real row, a located-apartness guard, a DyadicRatCore threshold ledger, a finite StreamName window, a RegSeqRat readback row, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8304.1, Definition 8304.2, Definition 8304.3, and Definition 8304.4 seed the guarded located sign-read surface at the L10 Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSignApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, total sign on $\mathsf{Real}^{\uparrow}$, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-apartness guard exposure, dyadic threshold exactness, finite-window readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSignApartnessUp.

Chapter 8305

A Concrete Naming Certificate for $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$

Following Borel, Cantelli, and standard probability texts, the $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$ packet records the independence-side converse deliberately absent from the first-lemma and finite-window packets of chapter 2357 and chapter 7718. It sits over the normalized probability source of chapter 184, the finite independence ledger of chapter 187, and the terminal scalar readback of chapter 38. The packet names only the finite event-sequence, divergent partial-sum, and independence witness rows needed to expose the converse route as BEDC data; it is not a completed almost-sure quotient, a countable-choice selector, a host product-measure theorem, or an ambient probability-limit principle.

Definition 8305.1 (Borel–Cantelli second-lemma carrier)**.** *A* $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\Omega, E, W, I, A, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the probability carrier from which all event reads are taken.*
2. E *is the event-sequence row, recording the displayed event names, membership readbacks, and measurable-event witnesses.*
3. W *is the finite repeated-hit window row, exposing the inspected prefix and tail windows before any limiting conclusion is visible.*
4. I *is the* $\mathsf{Independence}^{\uparrow}$ *row for the same displayed event family.*
5. A *is the pairwise or finite-family independence audit row consumed by the converse route.*
6. D *is the divergent partial-probability-sum row for the inspected event probabilities.*
7. R *is the repeated-hit request row, recording that the consumer asks for infinitely many displayed hits only through finite windows.*
8. L *is the converse ledger row, recording the route from independence and divergent partial sums to the repeated-hit surface.*
9. H *records componentwise* hsame *transport for* $\Omega, E, W, I, A, D, R, L, C, P, N$.

10. *C records displayed* Cont *replay from Ω through E, W, I, A, D, R, L before the local name is exposed.*
11. *P is the* Pkg *provenance over* $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$, $\mathsf{BorelCantelli}^{\uparrow}$, $\mathsf{FiniteBorelCantelli}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelCantelliSecondLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $\Omega, E, W, I, A, D, R, L, H, C, P, N$. It has no coordinate for a completed sigma algebra, quotient almost-sure equality, selected infinite subsequence, host product measure, or ambient real convergence theorem.

Theorem 8305.2 (Borel–Cantelli second-lemma NameCert obligations)**.** *For every accepted* BorelCantelliSecondLe *carrier $S = (\Omega, E, W, I, A, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BorelCantelliSecondLemma}^{\uparrow}}$ *surface consists exactly of probability-source admission through Ω, event-sequence exposure through E, repeated-hit window exposure through W, independence exposure through I, A, divergent partial-sum exposure through D, repeated-hit request exposure through R, converse ledgering through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8305.1. The displayed tuple exposes the probability source, event sequence, finite windows, independence rows, divergent partial-sum row, repeated-hit request, converse ledger, transport, replay, provenance, and local naming.

The event route must pass through Ω, E, W before any converse ledger is read. The independence rows I, A and the divergent partial-sum row D are therefore attached to the same displayed event family and finite windows, rather than to an ambient host sequence.

Finally R, L, H, C, P, N record the repeated-hit request, the converse route, structural transport, replay, source, and local name. Since the packet has no coordinate for a completed sigma algebra, quotient almost-sure equality, selected infinite subsequence, host product measure, or ambient convergence theorem, the listed rows exhaust the NameCert obligations. □

Theorem 8305.3 (Borel–Cantelli second-lemma independence route)**.** *Every accepted* $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$ *converse read factors through*

$$\mathsf{BorelCantelli}^{\uparrow},\ \mathsf{FiniteBorelCantelli}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{ProbSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The probability source and event-sequence rows are exposed before the finite repeated-hit windows; the independence audit and divergent partial-sum row are then read over that same displayed event family; only after those rows are visible may the converse ledger expose the repeated-hit request. The route exports no first-lemma summability conclusion, no finite-window-only bound, no countable-choice tail selector, no quotient almost-sure equality, and no host product-measure completion.

Proof. Start from the sibling first-lemma and finite-window packets. They provide the event-sequence and bounded-window boundaries named in chapter 2357 and chapter 7718, while leaving the independence converse off their public surfaces.

Next read Ω, E, W, I, A, D in order. The probability source of chapter 184 supplies event probabilities, the independence packet of chapter 187 supplies the displayed independence audit, and the $\mathsf{Real}^{\uparrow}$ surface of chapter 38 supplies the scalar partial-sum readback. These rows share the same event family.

The repeated-hit request R and ledger L are therefore local consumers of the displayed independence and divergence evidence. Componentwise transport, replay, provenance, and local naming through H, C, P, N preserve that route and do not add a quotient almost-sure object, selected infinite subsequence, host product measure, or ambient convergence principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BorelCantelliSecondLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{BorelCantelliSecondLemma}^{\uparrow}$ packet is generated from a probability source row, event-sequence row, finite repeated-hit windows, displayed independence rows, divergent partial-probability sums, a repeated-hit request, a converse ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelCantelliSecondLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8305.1, Definition 8305.2, and Definition 8305.3 seed the independence-side Borel–Cantelli converse route as a finite NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelCantelliSecondLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed sigma algebras, quotient almost-sure equality, countable-choice tail selection, host product-measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, event-window exactness, independence audit exposure, divergent partial-sum routing, repeated-hit request non-escape, converse ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.BorelCantelliSecondLemmaUp.

Chapter 8306

A Concrete Naming Certificate for $\mathsf{FiniteRadonTightness}^{\uparrow}$

Following Billingsley and Prokhorov, the packet records finite compact-window tightness evidence. It sits beside the probability source of chapter 184, the measure surface of chapter 92, the compact metric test interface of chapter 129, the real comparison boundary of chapter 38, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. It supplies a tightness row for the Prokhorov and Portmanteau consumers of chapter 4008 and chapter 4193 without importing host measures, countable choice, or selected compact sets.

Definition 8306.1 (Finite Radon tightness carrier)**.** *A* $\mathsf{FiniteRadonTightness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (P, M, K, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row whose probability context is visible before any outside-mass read.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *event-surface row attached to the same probability context.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-window row used as the finite test window.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row listing the inspected compact windows and tolerance requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance and outside-mass comparisons.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *comparison seal for the displayed mass bounds.*
7. *H records componentwise* hsame *transport for the probability, measure, compact-window, stream, readback, real, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of the same finite tightness route.*
9. *Q is the* Pkg *provenance over* $\mathsf{FiniteRadonTightness}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRadonTightness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Radon measure, a chosen compact exhaustion, weak compactness, selected subsequences, quotient event equality, countable choice, or ambient real completeness.

Theorem 8306.2 (Finite Radon tightness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRadonTightness}^{\uparrow}$ *carrier*

$$T = (P, M, K, S, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRadonTightness}^{\uparrow}}$ *surface consists exactly of carrier admission for the probability source, measure event surface, compact metric window, finite stream schedule, regular rational tolerance readback, real comparison seal, componentwise transport, displayed replay, package provenance, and local naming. Every Prokhorov- or Portmanteau-facing tightness read therefore factors through* $P, M, K, S, R, E, H, C, Q, N$ *before a weak-convergence consumer can inspect it.*

The packet supplies no host measure object, no selected compact set, no compact exhaustion, no weak compactness theorem, no countable-choice schedule, and no quotient equality of events.

Proof. Project the displayed carrier. The rows P, M, K, S, R, E are the only probability, measure, compact-window, finite-schedule, rational-readback, and real-comparison coordinates available to a tightness consumer.

The route is finite. The stream row S lists the inspected compact windows, the readback row R carries the tolerance comparisons, and the real seal E records the visible outside-mass bounds. This is the data later consumed by $\mathsf{ProkhorovTightProbability}^{\uparrow}$ and $\mathsf{Portmanteau}^{\uparrow}$.

The structural rows H, C, Q, N transport, replay, source, and name exactly that route. Since the carrier contains no coordinate for host measures, selected compact sets, compact exhaustions, weak compactness, countable choice, or quotient event equality, no accepted read can export those objects. □

Falsifiable prediction: Every accepted FiniteRadonTightnessUp consumer that reads a compact-window tightness bound without first displaying the probability, measure, compact-window, stream, readback, and real rows violates the NameCert obligations.

Independence witness: Nearest siblings: ProbSpaceUp, MeasureUp, CompactMetricUp, ProkhorovTightProbabilityUp, and PortmanteauUp. None is bijective to FiniteRadonTightnessUp because each sibling omits at least one row among finite compact-window scheduling, rational outside-mass readback, and the local tightness NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRadonTightness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRadonTightness}^{\uparrow}$ packet is generated from probability source rows, measure event-surface rows, compact metric window rows, finite stream schedules, regular rational tolerance readbacks, real comparison seals, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRadonTightness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8306.1, Definition 8306.2, the falsifiable prediction, and the independence witness bind the finite compact-window tightness carrier and its immediate consumer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRadonTightness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Radon measures, selected compact exhaustions, weak compactness, countable choice, quotient event equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, outside-mass exactness, Prokhorov and Portmanteau consumer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRadonTightne

Chapter 8307

A Concrete Naming Certificate for $\mathsf{LocatedWeakKonig}^{\uparrow}$

Following Troelstra and van Dalen on choice sequences, bars, and constructive weak Konig readings, the $\mathsf{LocatedWeakKonig}^{\uparrow}$ packet records the finite located-branch surface used when completion and real-name consumers inspect binary trees through explicit prefix windows. It sits after the boundary refusal row of chapter 3482, the finite-prefix route of chapter 2092, the finite stream windows of chapter 126, the Cantor prefix surface of chapter 1475, and the regular-rational readback of chapter 301. The packet names located finite branch evidence; it does not export countable choice, an open-cover compactness theorem, quotient stream equality, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8307.1 (Located weak Konig carrier)**.** *A* $\mathsf{LocatedWeakKonig}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, B, W, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite binary tree row, carrying decidable node membership, prefix closure, and finite successor checks through the* $\mathsf{KonigLemma}^{\uparrow}$ *and* $\mathsf{CantorSpace}^{\uparrow}$ *sibling routes.*
2. *B is the decidable located bar row, recording for the current request which finite prefixes are already blocked, which remain open inside the displayed window, and which comparison rows are still unresolved.*
3. *W is the finite prefix window row, read through* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any completion-facing consumer sees branch data.*
4. *E is the branch witness row, storing a finite compatible prefix or a finite obstruction for the displayed window, not a completed infinite branch and not a choice-selected path.*
5. *H records componentwise* hsame *transport for the tree, located bar, prefix window, branch witness, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a located branch request through T, B, W, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{LocatedWeakKonig}^{\uparrow}$, $\mathsf{WeakKonigBoundary}^{\uparrow}$, $\mathsf{KonigLemma}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedWeakKonig}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located-branch packet. It has no coordinate for countable choice, open-cover compactness, quotient stream equality, selected infinite branches, host path equality, or ambient Real completeness.

Lean4 (statement) `BEDC.Derived.LocatedWeakKonigUp`

Theorem 8307.2 (Located weak Konig NameCert obligations)**.** *For every accepted* $\mathsf{LocatedWeakKonig}^{\uparrow}$ *carrier*

$$L = (T, B, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedWeakKonig}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite binary tree, decidable located bar, finite prefix window, branch witness, transport, replay, provenance, and local naming rows;*
2. *tree-window exactness, requiring every branch-facing read to expose* T *and* W *before the witness row* E *is consumed;*
3. *located-bar decidability, requiring the bar row* B *to display the current finite blocked/open alternatives inside the same prefix window;*
4. *branch-witness locality, requiring* E *to be a finite compatible prefix or finite obstruction read through* T, B, W*;*
5. *transport and replay preservation through* H *and* C*, with provenance and local naming through* P *and* N*;*
6. *non-escape, excluding countable choice, open-cover compactness, quotient stream equality, selected infinite branches, host path equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No located weak Konig obligation is stored outside T, B, W, E, H, C, P, N*.*

Proof. Project the carrier of Definition 8307.1. The accepted packet displays exactly the finite tree, located bar, finite prefix window, finite witness, and structural rows.

Tree-window exactness reads the $\mathsf{KonigLemma}^{\uparrow}$ and $\mathsf{WeakKonigBoundary}^{\uparrow}$ sibling routes first. Those routes permit finite tree and prefix inspection while refusing branch selection and choice schedules.

The located bar row B and prefix window row W make the branch request finite. A consumer sees only the current blocked/open alternatives and the stream-window readback supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before reaching E.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier rows stores a selected infinite path, quotient stream equality, open-cover compactness object, choice schedule, or ambient Real-completeness witness, the listed rows exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.LocatedWeakKonigUp.LocatedWeakKonig namecert obligations`

Theorem 8307.3 (Located weak Konig finite branch route)**.** *Every Real, completion, or Cantor-facing consumer of an accepted* $\mathsf{LocatedWeakKonig}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{WeakKonigBoundary}^{\uparrow},\ \mathsf{KonigLemma}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ T, B, W, E, H, C, P, N.$$

The consumer may inspect located finite branch evidence and finite obstructions inside the displayed prefix window. It receives no countable-choice schedule, no open-cover compactness theorem, no quotient stream equality, no selected infinite branch, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a consumer read of $L = (T, B, W, E, H, C, P, N)$. Carrier admission exposes a finite binary tree, a decidable located bar, a prefix window, and a finite branch witness or obstruction.

Use the $\mathsf{WeakKonigBoundary}^{\uparrow}$ refusal surface and the $\mathsf{KonigLemma}^{\uparrow}$ finite-prefix route. Together they force tree and branch reasoning to remain in finite prefix data before any path-like consumer can read a witness row.

Next read the stream and Cantor siblings. The finite prefix window W is observed through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{CantorSpace}^{\uparrow}$, and the completion-facing readback goes through $\mathsf{RegSeqRat}^{\uparrow}$ before any Real or completion consumer uses the branch evidence.

The branch witness row E is therefore located inside the displayed finite window. Transport, replay, provenance, and local naming through H, C, P, N preserve that route and add none of the excluded global principles. □

Falsifiable prediction: A $\mathsf{LocatedWeakKonig}^{\uparrow}$ consumer that obtains a selected infinite branch, countable-choice schedule, quotient stream equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness without first displaying T, B, W, E, H, C, P, N refutes the carrier surface.

Independence witness: weak-konig-boundary, konig-lemma, streamname, cantorspace, regseqrat: $\mathsf{LocatedWeakKonig}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite binary tree, a decidable located bar, a prefix window, a finite branch witness or obstruction, transport, replay, provenance, and local NameCert row in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedWeakKonig}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedWeakKonig}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8307.1, Definition 8307.2, and Definition 8307.3 seed the finite located-branch NameCert surface over binary trees, located bars, and prefix windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedWeakKonig}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.LocatedWeakKonigUp`

Bridge status: none

Not claimed: This chapter does not close weak Konig's lemma as a global rule, countable choice, selected infinite branches, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree-window exactness, located-bar decidability, finite branch witness locality, transport, replay, and non-escape for BEDC.Derived.LocatedWeakKonigUp.

Constructive story: A $\mathsf{LocatedWeakKonig}^{\uparrow}$ packet is generated from a finite binary tree row, decidable located bar, finite prefix window, finite branch witness or obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8308

A Concrete Naming Certificate for $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$

Following Bishop and Bridges on finite approximations to real data, the $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ packet records the finite read surface by which a $\mathsf{StreamName}^{\uparrow}$ schedule is observed before a regular rational or real-facing consumer uses it. It sits after the countable-name gate of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only finite prefix indices, finite observation rows, handoff rows, transport, replay, provenance, and local naming; it is not a countable choice principle and not a host equality test for streams or completed reals.

Definition 8308.1 (Finite-prefix observation interface carrier)**.** *A* $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, F, S, R, D, H, K, P, N)$$

with the following displayed rows:

1. *I is the finite prefix index row naming the inspected initial segment of a* $\mathsf{StreamName}^{\uparrow}$ *schedule.*
2. *F is the finite observation row storing exactly the reads made at the indices named by I.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *handoff row whose public requests are restricted to the displayed prefix.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite stream observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the precision, radius, and comparison budget attached to the prefix read.*
6. *H records componentwise* hsame *transport for the prefix index, observation row, stream handoff, regular readback, dyadic ledger, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from the finite prefix index through F, S, R, D to the consumer-facing boundary.*
8. *P records* Pkg *provenance over* $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixObservationInterface}^{\uparrow}}$ *row.*

The carrier has no coordinate for a countable-choice schedule, infinite stream equality, selected limit, host real equality, trichotomy, or ambient completeness principle.

Theorem 8308.2 (Finite-prefix observation NameCert obligations)**.** *For every accepted* FinitePrefixObservationInte *carrier*

$$O = (I, F, S, R, D, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitePrefixObservationInterface}^{\uparrow}}$ *surface consists exactly of finite prefix admission through* I*, finite observation exposure through* F*,* $\mathsf{StreamName}^{\uparrow}$ *handoff through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8308.1. The prefix index I determines the only inspected positions, and the observation row F stores only the finite reads at those positions.

The stream handoff S is consumed through that prefix before the regular rational readback R is available. The dyadic ledger D records the tolerance data attached to the same finite observation row, so a consumer cannot replace the readback by a selected infinite stream, host real value, or unbounded choice schedule.

The structural rows H, K, P, N keep the route local. Componentwise hsame transport preserves the displayed prefix rows, Cont replay reuses the same finite route, Pkg provenance records the sibling packets, and the local NameCert names no data beyond the carrier. □

Theorem 8308.3 (Finite-prefix observation handoff boundary)**.** *Every downstream read of an accepted* $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ *packet factors through the finite route*

$$I \longmapsto F \longmapsto S \longmapsto R \longmapsto D$$

with structural transport, replay, provenance, and local naming through H, K, P, N*. The read may be handed to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumers only as this displayed finite prefix boundary, not as countable choice, infinite stream equality, selected limit, host real equality, or completeness data.*

Proof. Start with an accepted packet $O = (I, F, S, R, D, H, K, P, N)$. The carrier definition gives a single route from the prefix index to the observation row and then to the stream, regular rational, and dyadic tolerance rows.

The NameCert obligations of Definition 8308.2 account for every accepted public read. Since those obligations list only $I, F, S, R, D, H, K, P, N$, a consumer has no separate coordinate from which to extract an infinite schedule, a host equality decision, or a completion principle.

Finally use the structural rows. Transport by H and replay by K preserve the same finite prefix route, while P and N record provenance and local naming without adding analytic content. □

Falsifiable prediction: Every accepted $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ read displays a finite prefix index, finite observation row, StreamName handoff, RegSeqRat readback, DyadicRatCore tolerance ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert before any downstream real-facing consumer can use the observation boundary.

Independence witness: stream name, regular rational sequence, dyadic rational core, real: $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ is not bijective to any listed sibling because it binds finite prefix observation rows between a StreamName schedule and RegSeqRat or Real-facing readback.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixObservationInterface}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FinitePrefixObservationInterface}^{\uparrow}$ packet is generated from a finite prefix index, a finite observation row, a StreamName handoff, a RegSeqRat readback, DyadicRatCore tolerance data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FinitePrefixObservationInterface}\uparrow}$, seed)

Scope closed: Definition 8308.1 and Definition 8308.2 seed the finite observation route from StreamName schedules to regular rational and real-facing readback data.

Formal verification: FormalStatus(NameCert$_{\mathsf{FinitePrefixObservationInterface}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, infinite stream equality, selected limits, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked prefix admission, finite observation exposure, StreamName handoff, RegSeqRat readback, DyadicRatCore tolerance accounting, transport, replay, provenance, and local naming for BEDC.Derived.FinitePrefixObservationInterfaceUp.

Chapter 8309

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$ packet records the located regular-Cauchy real interface as finite BEDC NameCert data. It sits over the dyadic tolerance ledger of chapter 291, the finite-window naming surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located-interval packet of chapter 394. The packet names a constructive locator for a regular Cauchy real name; it is not a selected real representative, quotient stream, trichotomy principle, ambient completeness theorem, or host interval object.

Definition 8309.1 (Bishop regular Cauchy locator carrier)**.** *A* $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, I, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule inspected by the locator request;*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to that schedule;*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic radius, midpoint, and comparison budget used by the request;*
4. I *is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the displayed interval against which the regular Cauchy name is located;*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S, R, D, I *have been displayed;*
6. H *records componentwise* hsame *transport for the schedule, readback, dyadic tolerance, located interval, Real seal, replay, provenance, and local name;*
7. C *records displayed* Cont *replay of the route*

$$S \longmapsto R \longmapsto D \longmapsto I \longmapsto E;$$

8. P *is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLocator}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, D, I, E, H, C, P, N$. It has no coordinate for a selected host real, quotient equality of streams, undecidable trichotomy, hidden interval endpoint, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8309.2 (Bishop regular Cauchy locator NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$ *carrier*

$$L = (S, R, D, I, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLocator}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the StreamName window, RegSeqRat readback, dyadic tolerance row, located interval, RealUp seal, transport, replay, provenance, and local name;*
2. *locator-window compatibility, requiring S and R to expose the same finite regular-Cauchy request before the located interval row is read;*
3. *dyadic-interval exactness, requiring D and I to display the midpoint, radius, and interval-side data used by the locator;*
4. *Real-seal nonescape, requiring E to consume only the displayed StreamName, RegSeqRat, DyadicRatCore, and LocatedInterval route;*
5. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the displayed packet.*

No locator obligation is stored in a host representative, quotient stream, trichotomy oracle, selected interval endpoint, choice schedule, or ambient completeness row.

Proof. Project the carrier of Definition 8309.1. Its analytic rows are exactly the finite window S, regular rational readback R, dyadic tolerance row D, located interval I, and Real seal E.

Read the locator route in order. The StreamName and RegSeqRat rows expose the inspected regular-Cauchy request; the dyadic row supplies the tolerance data; and the LocatedInterval row records the interval side against which the request is located.

Finally inspect the structural rows. The hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row preserve the same finite route, while the carrier excludes host representatives, quotient streams, trichotomy, selected endpoints, choice schedules, and ambient completeness. □

Theorem 8309.3 (Bishop regular Cauchy locator Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing locator read exported by an accepted* $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports a finite located regular-Cauchy route, not a completed host real, quotient-stream equality, hidden interval endpoint, countable-choice selector, or ambient completeness claim.

Proof. Start with $L = (S, R, D, I, E, H, C, P, N)$. Projection exposes S, R, D, I before the terminal Real-facing row E is available.

The route through S and R fixes the inspected regular-Cauchy window. The dyadic row D fixes the tolerance budget, and the located interval I fixes the interval evidence for that same request.

The Real seal E is therefore downstream of the displayed finite locator route. Transport, replay, provenance, and local naming do not add another route, so the packet supplies only the constructive located handoff named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyLocator}^{\uparrow}$ packet is generated from a StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, LocatedInterval row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8309.1, Definition 8309.2, and Definition 8309.3 seed the finite located regular-Cauchy locator route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient stream equality, selected host representatives, selected interval endpoints, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locator-window compatibility, dyadic-interval exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyLocatorUp.

Chapter 8310

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$ packet records the finite modulus data used by the Real-completeness diagonal representative of Definition 9238.3. It consumes the regular rational readback of chapter 301, the finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the Cauchy-limit boundary of chapter 386. The packet names the diagonal-limit modulus as displayed finite data; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient-stream equality, a selected limit, countable choice, or proof-extensional host equality.

Definition 8310.1 (Regular Cauchy diagonal-limit modulus carrier)**.** *A* $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row from which the diagonal representative is read;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule selecting the visible diagonal requests;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the precision split used by the diagonal modulus;*
4. *Q is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row certifying that the displayed diagonal route is the one consumed by the limit handoff;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after R, S, D, Q have been displayed;*
6. *H records componentwise* hsame *transport for the family, windows, dyadic tolerance, Cauchy-limit seal, Real seal, replay, provenance, and local name;*
7. *C records displayed* Cont *replay of the diagonal-limit route*

$$R \longmapsto S \longmapsto D \longmapsto Q \longmapsto E;$$

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, S, D, Q, E, H, C, P, N$ and has no coordinate for a host-selected diagonal limit, quotient stream, ambient completion principle, countable choice, or proof-extensional equality.

Theorem 8310.2 (Regular Cauchy diagonal-limit modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$ *carrier*

$$M = (R, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the RegSeqRat family row, StreamName selected windows, DyadicRatCore tolerance ledger, CauchyLimitSeal row, RealUp seal, transport, replay, provenance, and local name;*
2. *diagonal-window compatibility, requiring the selected windows in S to be read against the same regular rational family R;*
3. *modulus-budget exactness, requiring D to expose the precision split and dyadic tolerance data used by the diagonal-limit modulus;*
4. *limit-seal handoff, requiring Q and E to consume the displayed R, S, D route before any Real-facing limit read is exported;*
5. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the displayed finite packet.*

No local obligation is stored in a host limit, quotient stream, hidden modulus selector, countable-choice schedule, proof-extensional equality, or ambient completeness row.

Proof. Project the carrier of Definition 8310.1. The analytic rows are the regular rational family R, selected finite windows S, dyadic modulus ledger D, Cauchy-limit seal Q, and Real seal E.

Read the diagonal route in dependency order. The family row and selected windows determine the visible diagonal requests; the dyadic ledger supplies the precision split for those requests; and the Cauchy-limit seal records that the same route is consumed by the Real-facing handoff.

Finally inspect the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve the same finite packet, while the carrier omits host limits, quotient streams, countable choice, proof extensionality, hidden modulus selectors, and ambient completeness. □

Theorem 8310.3 (Regular Cauchy diagonal-limit modulus Real route)**.** *Every diagonal-limit modulus read exported by an accepted* $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports the finite modulus and seal data needed by the diagonal representative, not a host-selected limit, quotient stream, hidden choice schedule, proof-extensional equality, or ambient Real-completeness theorem.

Proof. Start from $M = (R, S, D, Q, E, H, C, P, N)$. Projection exposes the RegSeqRat family, selected StreamName windows, dyadic tolerance ledger, and Cauchy-limit seal before the terminal Real row can be read.

The dyadic ledger D fixes the explicit modulus budget for the selected diagonal windows. The seal row Q forces the Real-facing route to consume that same budget rather than a detached host limit.

Transport, replay, provenance, and local naming preserve only the displayed finite packet. Therefore every exported diagonal-limit modulus read remains on the stated RegSeqRat-to-Real route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}$ packet is generated from a RegSeqRat family row, selected StreamName windows, DyadicRatCore tolerance ledger, CauchyLimitSeal row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8310.1, Definition 8310.2, and Definition 8310.3 seed the finite diagonal-limit modulus route for Real-completeness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected diagonal limits, countable choice, proof-extensional equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-window compatibility, modulus-budget exactness, limit-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyDiagonalLimitModulusUp.

Chapter 8311

A Concrete Naming Certificate for $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completions, the $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ packet records the finite reflector surface that turns a named Cauchy stream into regular-rational readback and then into a terminal real-name seal. It sits beside the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback route of chapter 301, the real-name boundary of chapter 38, and the completion reflector of chapter 1585. The packet names a non-quotient handoff surface; it does not select a representative stream, assert host equality of completion points, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8311.1 (Regular Cauchy stream reflector carrier)**.** *A* $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ *carrier is a finite packet*

$$R = (S, D, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected stream prefixes and tail requests.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radius, comparison budget, and endpoint arithmetic for the same windows.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S and D.*
4. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream, dyadic, and regular-rational rows have been replayed.*
5. *H records componentwise* hsame *transport for S, D, Q, L, C, P, N.*
6. *C records displayed* Cont *replay of the handoff*

$$S, D \longmapsto Q \longmapsto L.$$

7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyStreamReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, D, Q, L, H, C, P, N. It has no coordinate for quotient stream equality, selected representatives, hidden cofinal tails, host completion equality, countable choice, or ambient completeness.

Lean4 (statement) `BEDC.Derived.RegularCauchyStreamReflectorUp`

Theorem 8311.2 (Regular Cauchy stream reflector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ *carrier* $R = (S, D, Q, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyStreamReflector}^{\uparrow}}$ *surface consists exactly of StreamName finite-window exposure through* S*, DyadicRatCore tolerance exposure through* D*, RegSeqRat readback through* Q*, RealUp seal exposure through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. The reflector may be consumed by a* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route only through the displayed chain*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

It is not a quotient completion, a selected representative, a hidden choice schedule, or an ambient real-completeness theorem.

Proof. Project the carrier of Definition 8311.1. The only stream-facing rows are S and D, so every read of the packet starts with finite windows and a dyadic tolerance ledger.

The regular-rational row Q is downstream of those two rows, and the terminal real-name row L is downstream of Q. Thus the visible handoff is exactly the displayed StreamName-to-RegSeqRat-to-RealUp route, with the dyadic ledger still attached.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for quotient stream equality, selected representatives, hidden cofinal tails, host completion equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ consumer that exports a real-name handoff must display the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchycompletionreflector: no listed sibling is bijective to $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ because this packet jointly binds finite stream windows, dyadic tolerance, regular-rational readback, a real-name seal, reflector provenance, transport, replay, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyStreamReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyStreamReflector}^{\uparrow}$ packet is generated from StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyStreamReflector}^{\uparrow}}$, seed)

Scope closed: Definition 8311.1 and Definition 8311.2 seed the non-quotient StreamName-to-RegSeqRat-to-RealUp reflector route for the L10 Real and Completion phase.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyStreamReflector}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.RegularCauchyStreamReflectorUp`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, hidden cofinal tails, host completion equality, countable choice, ambient RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName finite-window exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, RealUp seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyStreamReflectorUp.

Chapter 8312

A Concrete Naming Certificate for $\mathsf{KolmogorovConsistency}^{\uparrow}$

Following Kolmogorov, the $\mathsf{KolmogorovConsistency}^{\uparrow}$ packet records finite index-family probability rows together with their marginal restriction ledgers. It sits between the finite-window surface of chapter 6120, the conditional-law interface of chapter 6929, the located tightness route of chapter 3795, and the tight-probability handoff of chapter 4008. The packet names finite consistency data only; it does not assert an infinite extension, choose a projective limit, or identify host measure-space quotients.

Definition 8312.1 (KolmogorovConsistency carrier)**.** *A* $\mathsf{KolmogorovConsistency}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, M, \rho, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-family row;*
2. *F is the finite family of probability rows over the displayed index windows;*
3. *M records the marginal maps from a larger displayed window to a smaller displayed window;*
4. *ρ is the restriction ledger comparing each larger-window probability row with the smaller-window row named by F;*
5. *L is the finite consistency ledger collecting all displayed restriction checks;*
6. *H records componentwise* hsame *transport for $I, F, M, \rho, L, C, P, N$;*
7. *C records the displayed* Cont *replay of finite-window restriction reads;*
8. *P is the* Pkg *provenance over* $\mathsf{KolmogorovConsistency}^{\uparrow}$, $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$, $\mathsf{RegularConditionalDi}$ $\mathsf{RealTightness}^{\uparrow}$, $\mathsf{ProkhorovTightProbability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovConsistency}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an infinite product measure, a selected projective-limit object, quotient event equality, or a host measure-space completion.

Theorem 8312.2 (KolmogorovConsistency obligation surface)**.** *For the carrier of Definition 8312.1, the local* $\mathsf{NameCert}_{\mathsf{KolmogorovConsistency}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, F, M, \rho, L, H, C, P, N$ finite probability-row exposure through F, marginal-map exposure through M, restriction-ledger*

exposure through ρ, consistency-ledger exposure through L, componentwise transport through H, replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier rows. The finite probability family is visible only as the indexed row F, and every comparison between windows is visible only through M, ρ, L.

The structural rows then bind the same packet. The row H transports the finite family and all restriction ledgers componentwise, while C replays only the displayed finite-window restriction reads. The package and naming rows P, N expose the packet as a Kolmogorov consistency surface without adding any host product-measure object. □

Theorem 8312.3 (KolmogorovConsistency marginal ledger)**.** *Every accepted* $\mathsf{KolmogorovConsistency}^{\uparrow}$ *marginal read from a displayed larger window to a displayed smaller window factors through the same rows*

$$I, F, M, \rho, L, H, C, P, N.$$

The read first exposes the two finite windows in I, then the probability rows in F, then the marginal map in M, and then the restriction comparison in ρ as recorded by L.

Proof. Start with the index-family row I and the probability-family row F. A marginal read has no accepted source until both finite windows and their probability rows have been displayed.

Apply the marginal-map and ledger rows. The row M names the finite restriction map, ρ names the comparison with the smaller-window row, and L records that this comparison belongs to the displayed consistency ledger.

Finally use H, C, P, N to transport, replay, package, and name the same finite route. None of these structural rows introduces an infinite extension, quotient event equality, or selected projective-limit witness. □

Theorem 8312.4 (KolmogorovConsistency public certificate)**.** *The public* $\mathsf{KolmogorovConsistency}^{\uparrow}$ *certificate is exactly the finite route through the index-family row, probability-family rows, marginal-map rows, restriction ledger, consistency ledger, componentwise transport, displayed continuation replay, package provenance, and local NameCert. Downstream probability consumers receive this finite certificate before any request for the stronger surfaces named by chapter 6120, chapter 6929, chapter 3795, or chapter 4008.*

Proof. Use the obligation surface to identify the complete public row set. It exposes only $I, F, M, \rho, L, H, C, P, N$ and therefore supplies a finite consistency certificate rather than an ambient measure-space construction.

Use the marginal-ledger theorem for every smaller-window read. It forces each consumer to pass through the displayed finite windows, probability rows, marginal map, and restriction comparison before any downstream probability surface can inspect the packet.

The sibling references are then read as consumers, not as extra coordinates of the certificate. They may consume the finite consistency route, but the packet itself does not claim full infinite extension, choice of projective limits, or host measure-space quotient equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovConsistency}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovConsistency}^{\uparrow}$ packet is generated from finite index-family rows, probability-family rows, marginal maps, restriction ledgers, consistency ledgers, componentwise transport, displayed replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovConsistency}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8312.1, Definition 8312.2, Definition 8312.3, and Definition 8312.4 display the finite consistency carrier, marginal restriction ledger, public certificate, and sibling probability routes.

Formal verification: FormalStatus(NameCert$_{\mathsf{KolmogorovConsistency}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close full infinite extension, selected projective limits, host measure-space quotient equality, Lean theorem checking, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, marginal-map exactness, restriction-ledger stability, transport, replay, provenance, and local naming for the finite consistency packet.

Chapter 8313

A Concrete Naming Certificate for $\mathsf{ModulusTailSelection}^{\uparrow}$

Bishop and Bridges present Cauchy convergence through explicit moduli, finite tail estimates, and regular rational readback before a completed real object is consumed. The $\mathsf{ModulusTailSelection}^{\uparrow}$ packet records the local selector that turns a displayed modulus request into the finite tail threshold read by Real and Completion consumers. It sits after the tail-selection criterion of chapter 1618, the convergence-modulus schedule of chapter 1438, the modulus certificate of chapter 7756, and the Cauchy estimate ledger of chapter 1825. The packet exports a finite handoff from modulus data to regular readback and Real sealing; it does not assert quotient stream equality, a selected limit, countable choice, host sequence convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8313.1 (Modulus tail selection carrier)**.** *A* $\mathsf{ModulusTailSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, S, R, D, E, H, C, P, N)$$

with the following visible rows:

1. *M is the modulus request row, naming the requested dyadic tolerance and the accepted convergence-modulus surface to be queried.*
2. *T is the displayed tail threshold selected for the request in M.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened at and beyond the threshold T.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed tail windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the modulus request, the tail threshold, and the Cauchy estimate comparison.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, T, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the modulus request, tail threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a modulus request through $M, T, S, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusTailSelection}^{\uparrow}$, $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{CauchyModulusCertificate}^{\uparrow}$, $\mathsf{CauchyEstimate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{ModulusTailSelection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, T, S, R, D, E, H, C, P, N$. It has no coordinate for quotient Cauchy classes, selected representatives, selected limits, countable choice, host real equality, hidden unbounded search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8313.2 (Modulus tail selection NameCert obligations)**.** *For every accepted* $\mathsf{ModulusTailSelection}^{\uparrow}$ *carrier $K = (M, T, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusTailSelection}^{\uparrow}}$ *surface consists exactly of carrier admission for the modulus request, displayed tail threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, componentwise transport, displayed replay, package provenance, and local naming. The carrier consumes the sibling surfaces of chapter 1618, chapter 1438, chapter 7756, and chapter 1825 before exporting any tail-read row.*

Proof. Project the carrier of Definition 8313.1. The analytic rows are precisely the modulus request M, selected threshold T, finite windows S, regular rational readback R, dyadic tolerance row D, and terminal seal E.

The request and threshold rows are read first. The request M is the displayed query into the convergence-modulus and certificate surfaces, while T is the finite threshold returned to the tail-selection route.

The readback rows are downstream of that threshold. The row S exposes only StreamName windows at and beyond T, the row R gives the matching RegSeqRat readback, and the row D supplies the dyadic tolerance arithmetic used by the estimate ledger.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient names, selected limits, countable choice, host equality, hidden unbounded search, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8313.3 (Modulus tail selection Cauchy handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{ModulusTailSelection}^{\uparrow}$ *packet must read the route*

$$\mathsf{CauchyConvergenceModulus}^{\uparrow},\ \mathsf{CauchyModulusCertificate}^{\uparrow} \longmapsto M, T \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}$$

through $M, T, S, R, D, E, H, C, P, N$. Thus the packet supplies a finite tail-selection certificate between convergence moduli and RegSeqRat/RealUp reads without claiming ambient Real completeness.

Proof. Begin with the sibling modulus surfaces. The convergence-modulus schedule supplies the admissible request row, and the modulus certificate makes the displayed threshold eligible for downstream tail reads.

Apply the tail-selection criterion. The request row M and threshold row T bind the requested dyadic tolerance to one displayed tail threshold, rather than to a hidden search over all later indices.

Read the finite windows and estimates next. The StreamName row S opens the selected tail windows, the RegSeqRat row R supplies their rational readback, the DyadicRatCore row D supplies the tolerance arithmetic, and the Cauchy estimate surface records the comparison consumed before the Real seal.

The Real row E is therefore reached only after the finite modulus, threshold, window, readback, tolerance, and estimate route has been displayed. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming, so no ambient completeness or choice principle is exported. □

Falsifiable prediction: A $\mathsf{ModulusTailSelection}^{\uparrow}$ read is refuted whenever a Real or Completion consumer uses a selected tail threshold without displaying the modulus request, tail threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: cauchy-tail-selection-criterion, cauchy-convergence-modulus, cauchy-modulus-certificate, cauchy-estimate, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{ModulusTailSelection}^{\uparrow}$ because the selector jointly records the request, threshold, window, readback, tolerance, seal, transport, replay, provenance, and local naming route.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusTailSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusTailSelection}^{\uparrow}$ packet is generated from a modulus request, displayed tail threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusTailSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8313.1, Definition 8313.2, and Definition 8313.3 seed the finite selector route from convergence-modulus data to RegSeqRat readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusTailSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, selected representatives, selected limits, countable choice, host real equality, hidden unbounded search, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, threshold selection, StreamName window coverage, RegSeqRat readback, DyadicRatCore tolerance accounting, Cauchy estimate handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ModulusTailSelectionUp.

Chapter 8314

A Concrete Naming Certificate for $\mathsf{NoetherianInduction}^{\uparrow}$

Following Emmy Noether's finite-chain viewpoint and the algebraic use in Atiyah–Macdonald, the $\mathsf{NoetherianInduction}^{\uparrow}$ packet records descent arguments as finite BEDC rows rather than as host well-founded induction. It sits beside the ideal-chain surface of chapter 3637, the module-chain surface of chapter 3648, the finite rank discipline of chapter 7374, and the chain-cover carrier of chapter 7370. The packet gives algebra chapters a reusable finite descent certificate while leaving ambient Noetherian induction, selected counterexample chains, classical minimal elements, and quotient carrier equality outside the public surface.

Definition 8314.1 (Noetherian induction carrier)**.** *A* $\mathsf{NoetherianInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, D, O, E, T, R, P, N, L)$$

with the following displayed rows:

1. *S is the source row, naming the* $\mathsf{NoetherianRing}^{\uparrow}$ *or* $\mathsf{NoetherianModule}^{\uparrow}$ *surface from which the descent request is read.*
2. *D is the descending-chain row, listing the inspected obstruction chain and the finite predecessor relation used by the request.*
3. *O is the obstruction row, recording the currently unclosed counterexample or subobject boundary.*
4. *E is the descent-step row, recording the finite step that replaces a visible obstruction by a strictly earlier row in D.*
5. *T is the stabilization row, exposing the finite stage at which the displayed chain cannot continue to produce fresh obstructions.*
6. *R is the transport row for the source, chain, obstruction, descent-step, stabilization, replay, provenance, and local naming rows.*
7. *P is the displayed replay row, specifying how a consumer replays the same finite descent route before using the stabilized conclusion.*
8. *N is the* Pkg *provenance over* $\mathsf{NoetherianInduction}^{\uparrow}$, $\mathsf{NoetherianRing}^{\uparrow}$, $\mathsf{NoetherianModule}^{\uparrow}$, $\mathsf{WellFoundedTreeRank}^{\uparrow}$ $\mathsf{FiniteChainDecomposition}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{NoetherianInduction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host well-founded induction, a globally selected minimal counterexample, an infinite descending chain, quotient subobject equality, classical choice, or an ambient Noetherian principle.

Theorem 8314.2 (Noetherian induction NameCert obligations)**.** *For every accepted* $\mathsf{NoetherianInduction}^{\uparrow}$ *carrier*

$$I = (S, D, O, E, T, R, P, N, L),$$

the local $\mathsf{NameCert}_{\mathsf{NoetherianInduction}^{\uparrow}}$ *surface consists exactly of source admission through* S*, descending-chain exposure through* D*, obstruction visibility through* O*, descent-step exposure through* E*, stabilization exposure through* T*, componentwise transport through* R*, displayed replay through* P*, provenance through* N*, and local naming through* L*. A consumer may use the induction packet only after the finite chain, the current obstruction, and the step-to-earlier-row witness are displayed.*

Proof. Project the carrier of Definition 8314.1. The source row S fixes the Noetherian ring or module surface before a descent request is read, and the row D names the finite chain on which the request is evaluated.

The obstruction row O and the descent-step row E are paired. A consumer cannot cite a smaller obstruction unless E displays the finite predecessor relation inside the same chain row D.

The stabilization row T closes the displayed descent by giving a finite stage where no fresh obstruction row remains available. Transport and replay through R and P preserve the same source, chain, obstruction, step, and stabilization rows, while N and L source and name the packet.

Since host well-founded induction, selected minimal counterexamples, infinite descending chains, quotient subobject equality, choice, and ambient Noetherian principles have no coordinate in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8314.3 (Noetherian induction chain descent)**.** *Every accepted* $\mathsf{NoetherianInduction}^{\uparrow}$ *descent read factors through the finite route*

$$\mathsf{NoetherianRing}^{\uparrow},\ \mathsf{NoetherianModule}^{\uparrow},\ \mathsf{WellFoundedTreeRank}^{\uparrow},\ \mathsf{FiniteChainDecomposition}^{\uparrow}.$$

The chain row supplies the inspected predecessor spine, the rank row or chain decomposition bounds the visible descent, the obstruction row is consumed by the displayed descent step, and the stabilization row records the endpoint where no new obstruction can be produced. The read exports no host well-founded induction theorem, no selected minimal counterexample, no infinite descending chain, and no classical choice principle.

Proof. Start from the source row. If the descent request is algebraic, the relevant $\mathsf{NoetherianRing}^{\uparrow}$ or $\mathsf{NoetherianModule}^{\uparrow}$ packet supplies the finite chain-stabilization surface; if the request is purely order-theoretic, the $\mathsf{WellFoundedTreeRank}^{\uparrow}$ and $\mathsf{FiniteChainDecomposition}^{\uparrow}$ siblings supply the finite rank or chain-spine rows.

Next read D, O, E. The chain row exposes the inspected predecessor sequence, the obstruction row names the current failed subclaim, and the descent-step row replaces it by an earlier displayed row. This is finite chain descent, not an appeal to a host induction principle.

Finally consume T, R, P, N, L. Stabilization prevents an endless supply of fresh obstruction rows, while transport, replay, provenance, and local naming keep the argument on the same finite packet. The excluded host induction, selector, infinite-chain, quotient, and choice objects therefore cannot be exported by an accepted read. □

Falsifiable prediction: An accepted $\mathsf{NoetherianInduction}^{\uparrow}$ consumer is refuted if it reaches a stabilized conclusion without first displaying the source row, finite descending-chain row, obstruction row, descent-step row, stabilization row, transport, replay, provenance, and local naming rows.

Independence witness: Nearest siblings: noetherian-ring, noetherian-module, well-founded-tree-rank, and finite-chain-decomposition. None is bijective to $\mathsf{NoetherianInduction}^{\uparrow}$ because this packet jointly carries a descent request, obstruction row, finite descent step, stabilization row, replay, provenance, and local naming for a single induction-facing read.

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherianInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{NoetherianInduction}^{\uparrow}$ packet is generated from a Noetherian ring or module source row, finite descending-chain row, obstruction row, descent-step row, stabilization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherianInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8314.1, Definition 8314.2, and Definition 8314.3 seed the finite descent packet over source, chain, obstruction, step, stabilization, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherianInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host well-founded induction, selected minimal counterexamples, infinite descending chains, quotient subobject equality, classical choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, descending-chain exposure, obstruction visibility, descent-step exactness, stabilization exposure, transport, replay, provenance, and local naming for BEDC.Derived.NoetherianInductionUp.

Chapter 8315

A Concrete Naming Certificate for $\mathsf{LocatedCompletionCore}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCompletionCore}^{\uparrow}$ packet records the finite located Cauchy-completion core that a Real-facing construction may consume before any quotient completion or selected limit is named. It sits beside the metric completion route of chapter 1506, the complete metric service of chapter 128, the Cauchy filter surface of chapter 333, the finite StreamName interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38. The carrier is a finite L10 completion spine; it is not a quotient completion object, a chosen limit, a countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8315.1 (LocatedCompletionCore carrier)**.** *A* $\mathsf{LocatedCompletionCore}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the completion source row, naming the metric or completion-facing input before any completed point is exposed.*
2. *S is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording only the displayed service interface used by the completion read.*
3. *F is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, carrying the finite located Cauchy filter evidence read by the core.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which Cauchy requests and located tests are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and filter data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the precision, radius, and comparison budget visible to the read.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after M, S, F, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the source, service, located Cauchy filter, window, readback, tolerance ledger, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from any limit-facing request through $M, S, F, W, R, D, E, H, P, N$.*

10. *P records* Pkg *provenance over* $\mathsf{LocatedCompletionCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCore}^{\uparrow}}$ *row.*

The packet has no coordinate for quotient completion equality, selected completed points, hidden representatives, countable choice, trichotomy, or ambient Real completeness.

Theorem 8315.2 (LocatedCompletionCore NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletionCore}^{\uparrow}$ *carrier $L = (M, S, F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCore}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the completion source, complete service, located Cauchy filter, finite window, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local name;*
2. *located Cauchy exposure, requiring every completion-facing read to display F before any readback or seal is consumed;*
3. *finite-window exactness, requiring W and D to carry the visible requests, tolerances, and comparison budget used by R;*
4. *Real seal non-escape, requiring E to be consumed only as the terminal seal downstream of M, S, F, W, R, D;*
5. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the displayed finite packet.*

No local NameCert obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 8315.1. The accepted packet exposes exactly the completion source, complete service row, located Cauchy filter, finite window, regular rational readback, dyadic tolerance ledger, terminal Real seal, and structural rows.

Located Cauchy exposure is the row F. Since F is upstream of W, R, D, E, a completion-facing consumer cannot treat the terminal seal as a selected completed point before the located filter evidence is visible.

Finite-window exactness uses W, D, R. The window stores the inspected requests, the dyadic ledger stores the tolerance data, and the readback row records only the regular rational data available through those finite rows.

The remaining rows transport, replay, source, and locally name the same packet. Because the carrier omits quotient completion equality, selected completed points, hidden representatives, countable choice, trichotomy, and ambient Real completeness, no NameCert obligation can export them. □

Theorem 8315.3 (LocatedCompletionCore choice-free limit route)**.** *Every limit-facing read of an accepted* $\mathsf{LocatedCompletionCore}^{\uparrow}$ *carrier factors through the displayed finite rows*

$$M, S, F, W, R, D, E, H, C, P, N.$$

In particular, a consumer receives the completion source, complete service, located Cauchy filter, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, and receives no quotient completion, selected limit, countable choice, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 8315.1. The completion-facing input is M, the only complete-service coordinate is S, and the located Cauchy evidence is the displayed filter row F.

The finite analytic read is then forced through W, R, D. The StreamName window supplies the inspected requests, the RegSeqRat row supplies regular rational readback, and the dyadic ledger supplies the tolerance budget attached to that read.

Apply the NameCert obligations of Definition 8315.2. They place the terminal Real seal E downstream of M, S, F, W, R, D and exhaust every structural read through H, C, P, N.

Since the carrier has no coordinate for quotient completions, selected limits, countable choice, trichotomy, or ambient Real completeness, a limit-facing consumer cannot cite any of those principles as $\mathsf{LocatedCompletionCore}^{\uparrow}$ evidence. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionCore}^{\uparrow}$ read is refuted if it reaches a Real-facing seal without displaying the completion source, complete service row, located Cauchy filter, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local NameCert rows.

Independence witness: metriccompletion, completemetric, cauchyfilter, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCompletionCore}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a completion source, complete service, located Cauchy filter, finite window, regular readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming as one finite core.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionCore}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompletionCore}^{\uparrow}$ packet is generated from a completion source, complete service row, located Cauchy filter, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionCore}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8315.1, Definition 8315.2, and Definition 8315.3 seed the finite located completion core consumed by Real-facing completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionCore}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completed points, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located Cauchy exposure, finite-window exactness, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionCoreUp.

Chapter 8316

A Concrete Naming Certificate for MetacicResidualCriticalPairSieve$^{\uparrow}$

The MetacicResidualCriticalPairSieve$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as a finite residual sieve surface. It sits beside the critical-path carrier of chapter 765, the confluence audit packet of chapter 735, the candidate-normalization handoff of chapter 5547, and the decidable boundary of chapter 749. The sieve records bounded residual candidates and blocked edges before any confluence-facing read is public; it does not assert full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.

Definition 8316.1 (Metacic residual critical-pair sieve carrier)**.** *A* MetacicResidualCriticalPairSieve$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, Q, C, H, D, R, B, T, K, P, N)$$

with the following displayed rows:

1. *A is the MetaCIC audit anchor read through* MetaCICCriticalPath$^{\uparrow}$*.*
2. *Q is the finite residual critical-pair candidate list, naming the inspected source, left residual edge, right residual edge, and local overlap code for each candidate.*
3. *C is the* MetaCICConfluenceAuditPacket$^{\uparrow}$ *row that records how the candidate list is read by the confluence audit surface.*
4. *H is the* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *row, retaining candidate-normalization evidence before any bounded endpoint is cited.*
5. *D is the* MetaCICDecidableBoundary$^{\uparrow}$ *row used only as a bounded checker and not as global conversion decidability.*
6. *R is the residual-window row, recording the finite reduction neighborhood in which each critical pair is inspected.*
7. *B is the blocked-edge ledger, recording candidates whose local join is not discharged by the displayed window and checker rows.*
8. *T records componentwise* hsame *transport for the anchor, candidate list, audit packet, handoff, decidable boundary, residual window, blocked-edge ledger, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay from a sieve request through* $A, Q, C, H, D, R, B, T, P, N$.
10. *P records* Pkg *provenance over* MetacicResidualCriticalPairSieve$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICConfluenceAudit MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{MetacicResidualCriticalPairSieve}^{\uparrow}}$ *row.*

The packet has no coordinate for a global confluence theorem, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.

Theorem 8316.2 (Metacic residual critical-pair sieve NameCert obligations)**.** *For every accepted* MetacicResidualCriticalPairSieve$^{\uparrow}$ *carrier* $S = (A, Q, C, H, D, R, B, T, K, P, N)$*, the local* NameCert$_{\text{MetacicResidualCritical}}$ *surface consists exactly of carrier admission, residual-candidate exposure, audit-packet routing, candidate-normalization handoff retention, decidable-boundary boundedness, blocked-edge retention, componentwise transport, displayed replay, package provenance, and local naming. Every public read factors through*

$$A, Q, C, H, D, R, B, T, K, P, N$$

before it may be cited as sieve evidence.

Proof. Carrier admission is projection from Definition 8316.1. The accepted packet exposes the audit anchor, finite residual candidate list, confluence audit row, candidate-normalization handoff, decidable boundary, residual window, blocked-edge ledger, and structural rows.

Residual-candidate exposure is the row Q together with the residual window R. These rows name the inspected critical pairs and the bounded neighborhood in which their residual edges are compared.

Audit routing and handoff retention use C, H, D, B. The confluence audit packet reads the candidate list, the candidate-normalization handoff retains its premise surface, the decidable boundary supplies only bounded checking, and the blocked-edge row keeps non-discharged candidates visible.

The structural rows T, K, P, N transport, replay, source, and locally name the same finite packet. Since the carrier omits global confluence, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, and consistency, no local NameCert obligation can export those endpoints. □

Theorem 8316.3 (Metacic residual critical-pair sieve boundary)**.** *Every confluence-facing read of an accepted* MetacicResidualCriticalPairSieve$^{\uparrow}$ *packet must first display the* MetaCICCriticalPath$^{\uparrow}$ *anchor, the finite residual critical-pair candidate list, the* MetaCICConfluenceAuditPacket$^{\uparrow}$ *row, the* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *row, the* MetaCICDecidableBoundary$^{\uparrow}$ *bounded checker row, residual-window data, blocked-edge retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The read is a finite sieve over residual critical pairs, not full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.*

Proof. Start with the carrier of Definition 8316.1. The anchor A fixes the critical-path surface, while Q and R name the finite residual critical-pair window being inspected.

The confluence audit row C and the candidate-normalization handoff H keep the read attached to the audit and candidate premise surfaces. The decidable boundary row D is only a bounded checker for the displayed window, and the blocked-edge row B records candidates not discharged by that bounded check.

Apply the NameCert obligations of Definition 8316.2. They exhaust transport, replay, provenance, and local naming through T, K, P, N.

Because none of the carrier coordinates is a global Church–Rosser proof, unrestricted subject-reduction proof, global conversion decider, quotient syntax equality, host evaluator soundness principle, or consistency theorem, no public read of the sieve may cite such an endpoint. □

Falsifiable prediction: Every accepted $\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}$ read is refuted if it exports a confluence-facing candidate while omitting the critical-path anchor, finite residual candidate list, confluence audit packet, candidate-normalization handoff, decidable-boundary checker, residual window, blocked-edge ledger, transport, replay, provenance, or local NameCert rows.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, metacic-candidate-normalization-confluence-handoff, metacicdecidableboundary, metacic-residual-critical-pair: none is bijective to $\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}$ because the sieve jointly carries residual critical-pair candidates, residual windows, bounded checker status, blocked-edge retention, audit routing, candidate-normalization handoff, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}$ packet is generated from a MetaCICCriticalPath anchor, finite residual critical-pair candidates, a MetaCICConfluenceAuditPacket row, a MetacicCandidateNormalizationConfluence-Handoff row, a MetaCICDecidableBoundary bounded checker, residual windows, blocked-edge retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8316.1, Definition 8316.2, and Definition 8316.3 seed the finite residual critical-pair sieve route for the MetaCIC confluence audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicResidualCriticalPairSieve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-candidate exposure, audit routing, candidate-normalization handoff retention, decidable-boundary boundedness, blocked-edge retention, transport, replay, provenance, and local naming for BEDC.Derived.MetacicResidualCriticalPairSieveUp.

Chapter 8317

A Concrete Naming Certificate for MetaCICCriticalPairResidualScheduler$^{\uparrow}$

The MetaCICCriticalPairResidualScheduler$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as a finite scheduler for critical-pair residual reads. It sits beside the critical-path route of chapter 765, the normalization frontier of chapter 892, the confluence audit packet of chapter 735, and the parallel-diamond residual boundary of chapter 5126. The packet is a bounded residual and local-diamond scheduler; it is not full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, or MetaCIC kernel consistency.

Definition 8317.1 (MetaCIC critical-pair residual scheduler carrier)**.** *A* MetaCICCriticalPairResidualScheduler$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, A, R, J, B, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the critical-pair row, naming the finite local peak and the inspected binder window.*
2. *F is the normalization-frontier row, recording the candidate-normalization boundary available for the same peak.*
3. *A is the confluence-audit row, retaining the checked fragment and the named blocked premises from the audit map.*
4. *R is the residual-schedule row, ordering residual reads before any local diamond handoff is exposed.*
5. *J is the candidate local-join row produced by the schedule when the bounded residual rows agree.*
6. *B is the bounded residual-checker row, recording the finite budget and the local tests performed by the scheduler.*
7. *O is the obstruction row, retaining residual-substitution, subject-reduction, evaluator-equality, quotient-equality, and kernel-consistency boundaries when the schedule does not expose a local join.*
8. *H records componentwise* hsame *transport for the critical-pair, frontier, audit, schedule, local-join, checker, obstruction, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay through the same residual-scheduler route.*

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, MetaCICNormalizatic $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{ParallelDiamondResidual}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, unbounded residual search, or MetaCIC kernel consistency.

Theorem 8317.2 (MetaCIC critical-pair residual scheduler obligations)**.** *For every accepted* MetaCICCriticalPairRe *carrier*

$$S = (K, F, A, R, J, B, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}}$ *surface consists exactly of critical-pair exposure through K, normalization-frontier exposure through F, confluence-audit exposure through A, residual-schedule exposure through R, candidate local-join exposure through J, bounded-checker exposure through B, obstruction visibility through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. A confluence-facing consumer may inspect a local diamond handoff only after the residual schedule and bounded checker rows are displayed.*

Proof. Project the carrier of Definition 8317.1. The row K fixes the critical pair, F fixes the normalization boundary, and A fixes the confluence audit context before any residual read is scheduled.

The residual-schedule row R is read before the candidate join row J. The bounded checker row B records the finite tests that justify exposing J; otherwise the obstruction row O remains visible.

Transport and replay through H and C preserve the same schedule, checker, local-join, and obstruction rows. Provenance and local naming through P, N close only that finite packet.

Since full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, unbounded residual search, and kernel consistency have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 8317.3 (MetaCIC critical-pair residual scheduler diamond boundary)**.** *Every local-diamond read exported by an accepted* $\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}$ *packet is bounded by*

$$\mathsf{MetaCICCriticalPath}^{\uparrow},\ \mathsf{MetaCICNormalizationFrontier}^{\uparrow},\ \mathsf{MetaCICConfluenceAuditPacket}^{\uparrow},\ \mathsf{ParallelDiamondResidual}^{\uparrow}.$$

The residual schedule must read the critical pair, candidate normalization frontier, confluence audit packet, bounded residual checker, and obstruction row before it can expose a candidate local join. The boundary exports no full Church–Rosser theorem, no subject-reduction theorem, no evaluator equality, no quotient syntax equality, and no kernel-consistency theorem.

Proof. Start with the critical-pair and frontier rows K, F. They locate the local peak and the candidate normalization boundary before any residual-diamond consumer can ask for a join.

Next consume A, R, B, O. The audit row identifies the conditional confluence map, the scheduler row orders residual reads, the bounded checker row records the finite tests, and the obstruction row remains visible when those tests do not certify the candidate join.

Only after those rows are displayed may J be exported as a local-diamond handoff. Transport, replay, provenance, and local naming through H, C, P, N keep the handoff on the same finite packet, while the excluded global theorems remain outside the carrier. □

Falsifiable prediction: If a residual read skips the scheduler row R and reaches a candidate local join directly, it is not an accepted $\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}$ carrier read.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-confluence-audit-packet, parallel-diamond-residual: none is bijective to $\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}$ because this packet jointly carries residual scheduling, bounded checker, candidate local join, obstruction retention, transport, replay, provenance, and local naming rows.

The Lean TasteGate route is expected to provide `Nontrivial MetaCICCriticalPairResidualSchedulerUp` with an explicit witness pair and `FieldFaithful MetaCICCriticalPairResidualSchedulerUp` with `.mk`-derived fields and a match/injection proof.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}$ packet is generated from a critical-pair row, normalization-frontier row, confluence-audit row, residual-schedule row, candidate local-join row, bounded residual-checker row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8317.1, Definition 8317.2, and Definition 8317.3 seed the bounded residual scheduler over critical pairs, candidate frontiers, audit rows, local joins, obstruction rows, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairResidualScheduler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, unbounded residual search, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, critical-pair exposure, frontier exposure, audit exposure, residual-schedule exactness, bounded-checker exactness, candidate-join exposure, obstruction retention, transport, replay, provenance, local naming, and TasteGate nontriviality and field-faithfulness for BEDC.Derived.MetaCICCriticalPairResidualSchedu

Chapter 8318

A Concrete Naming Certificate for UniformCauchyCompletionBridge$^{\uparrow}$

Following Bishop and Bridges on uniform spaces and constructive completion, the UniformCauchyCompletionBridge$^{\uparrow}$ packet records the finite route by which a uniform Cauchy presentation enters the completion handoff and reaches the terminal Real seal. It sits beside the uniform Cauchy criterion of chapter 435, the uniform completion functor of chapter 1892, the metric completion packet of chapter 1506, the finite stream-name windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The bridge names a displayed finite completion route; it is not a quotient completion, selected limit, countable-choice schedule, host equality of completed points, or ambient Real$^{\uparrow}$ completeness theorem.

Definition 8318.1 (Uniform Cauchy completion bridge carrier)**.** *A* UniformCauchyCompletionBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, F, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformCauchyCriterion$^{\uparrow}$ *row naming the shared uniform Cauchy threshold surface consumed by the bridge.*
2. *F is the* UniformCompletionFunctor$^{\uparrow}$ *row that exposes the functorial completion handoff without selecting a quotient representative.*
3. *M is the* MetricCompletion$^{\uparrow}$ *row used by completion-facing consumers after the uniform route has been displayed.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row through which the uniform Cauchy data are inspected.*
5. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the same finite windows.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after U, F, M, S, R have been exposed.*
7. *H records componentwise* hsame *transport for the uniform criterion, uniform completion functor, metric completion handoff, finite windows, regular readback, real seal, and local certificate rows.*
8. *C records displayed* Cont *replay from a uniform Cauchy request through U, F, M, S, R, E.*
9. *P is the* Pkg *provenance over* UniformCauchyCompletionBridge$^{\uparrow}$*,* UniformCauchyCriterion$^{\uparrow}$*,* UniformCompletionFun MetricCompletion$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionBridge}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $U, F, M, S, R, E, H, C, P, N$. It has no coordinate for quotient completions, selected limits, countable choice, host equality of completion points, ambient real completeness, Lean verification, or bridge checking.

Theorem 8318.2 (Uniform Cauchy completion bridge NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyCompletionBridge}^{\uparrow}$ *carrier*

$$B = (U, F, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform Cauchy criterion row, uniform completion functor row, metric completion handoff row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read is therefore routed through $U, F, M, S, R, E, H, C, P, N$ before the Real seal is consumed.*

Proof. Project the carrier of Definition 8318.1. The rows U, F, M, S, R, E are the only coordinates through which a consumer can read uniform Cauchy data, completion handoff data, finite windows, regular rational readback, and the terminal real seal.

The uniform side is read through U and F. The criterion row supplies the shared threshold surface, while the functor row exposes the completion handoff as displayed data rather than as a selected representative or quotient construction.

The Real-facing side is read through S, R, E. Finite stream windows expose the inspected names, regular rational readback carries the same finite observations, and the seal row is available only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no quotient, selector, choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8318.3 (Uniform Cauchy completion bridge finite route)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCompletionBridge}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{UniformCauchyCriterion}^{\uparrow},\ \mathsf{UniformCompletionFunctor}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHis}$$

through $U, F, M, S, R, E, H, C, P, N$. The route packages uniform Cauchy data into the completion and Real seal corridor without invoking quotient completion, selected limits, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8318.2. They identify every accepted public read with the displayed rows $U, F, M, S, R, E, H, C, P, N$.

The completion side is forced through U, F, M. A consumer first reads the uniform Cauchy criterion, then the uniform completion functor handoff, and only then the metric completion row that downstream completion chapters may use.

The Real-facing side is forced through S, R, E. The stream windows and regular rational readback expose finite observations before the terminal seal is available.

Transport and replay by H and C preserve the same route, while P and N provide provenance and local naming. No remaining row can supply quotient completion, selected limits, countable choice, host equality, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyCompletionBridge}^{\uparrow}$ consumer that reaches a completion or Real-facing seal first displays the uniform criterion row, uniform completion functor handoff, metric completion row, finite stream windows, regular rational readback, real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: uniformcauchycriterion, uniformcompletionfunctor, metriccompletion, streamname, regseqrat, real: $\mathsf{UniformCauchyCompletionBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the uniform criterion, completion functor, metric completion handoff, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming in one bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCompletionBridge}^{\uparrow}$ packet is generated from a uniform Cauchy criterion row, a uniform completion functor handoff, a metric completion row, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8318.1, Definition 8318.2, and Definition 8318.3 seed the finite uniform Cauchy route into completion and Real-facing readback data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform criterion exposure, completion handoff, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyCompletionBri

Chapter 8319

A Concrete Naming Certificate for $\mathsf{RegularLagrangian}^{\uparrow}$

Following Weinstein's symplectic geometry convention for Lagrangian submanifolds, the $\mathsf{RegularLagrangian}^{\uparrow}$ packet records a displayed finite Lagrangian carrier whose tangent, restriction, and regularity rows are visible before any symplectic consumer may cite it. It sits beside the symplectic source of chapter 138, the manifold carrier of chapter 96, the differential-form surface of chapter 136, the Lagrangian-mechanics packet of chapter 337, and the bundle-local surface of chapter 97. The chapter names a finite regular Lagrangian interface; it is not a host submanifold primitive, a selected Weinstein neighbourhood, a Hamiltonian-isotopy theorem, a Floer complex, or an ambient transversality principle.

Definition 8319.1 (Regular Lagrangian carrier)**.** *A* $\mathsf{RegularLagrangian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, M, F, A, T, R, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Symplectic}^{\uparrow}$ *source row naming the ambient closed nondegenerate 2-form.*
2. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row for the carried domain on which the Lagrangian surface is inspected.*
3. *F is the* $\mathsf{DiffForm}^{\uparrow}$ *restriction row that records the pulled-back symplectic form on the displayed domain.*
4. *A is the finite atlas-and-admission row for the carried Lagrangian locus inside the ambient symplectic packet.*
5. *T is the tangent-rank row recording that the displayed tangent data are half-dimensional and isotropic against the form row F.*
6. *R is the regularity row: finite chart windows, transition readbacks, and rank witnesses are stable under the local carrier relation.*
7. *B is the* $\mathsf{Bundle}^{\uparrow}$ *normal or tangent-bundle bookkeeping row used by later neighbourhood, Hamiltonian-isotopy, or Floer-style consumers.*
8. *H records componentwise* hsame *transport for the symplectic source, manifold charts, differential-form restriction, atlas admission, tangent-rank row, regularity row, bundle bookkeeping, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a regular Lagrangian request through S, M, F, A, T, R, B.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularLagrangian}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{LagrangianMechanics}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host submanifold object, selected Darboux chart, Weinstein neighbourhood, Hamiltonian isotopy, Floer differential, ambient compactness, Lean verification, or bridge checking.

Theorem 8319.2 (Regular Lagrangian NameCert obligations)**.** *For every accepted* $\mathsf{RegularLagrangian}^{\uparrow}$ *carrier*

$$L = (S, M, F, A, T, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}$ *surface consists exactly of carrier admission for the symplectic source, manifold chart row, differential-form restriction, finite atlas admission, tangent-rank row, regularity row, bundle bookkeeping, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every regular Lagrangian read is therefore routed through $S, M, F, A, T, R, B, H, C, P, N$.*

Proof. Project the carrier of Definition 8319.1. The rows S, M, F, A, T, R, B are the only coordinates through which a consumer can read the ambient symplectic form, charted locus, restricted form, finite admission data, tangent-rank evidence, regularity witnesses, and bundle bookkeeping.

The Lagrangian condition is read through F and T. The differential-form row supplies the restricted symplectic form, while the tangent-rank row records half-dimensional isotropic data against that same restriction.

The regularity surface is read through A, R, B. Finite atlas admission names the carried locus, the regularity row keeps chart windows and rank witnesses stable, and the bundle row exposes only the bookkeeping needed by later local-neighbourhood or dynamics consumers.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no host submanifold, selected Darboux chart, Weinstein neighbourhood, Hamiltonian-isotopy theorem, Floer differential, or ambient-compactness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8319.3 (Regular Lagrangian symplectic handoff)**.** *Every* $\mathsf{Symplectic}^{\uparrow}$*-,* $\mathsf{LagrangianMechanics}^{\uparrow}$*-, or bundle-facing consumer of an accepted* $\mathsf{RegularLagrangian}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Symplectic}^{\uparrow},\ \mathsf{Manifold}^{\uparrow},\ \mathsf{DiffForm}^{\uparrow},\ \mathsf{Bundle}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}$$

through $S, M, F, A, T, R, B, H, C, P, N$ before using a regular Lagrangian boundary. The handoff supplies displayed tangent-rank, restricted-form, chart-regularity, and bundle-bookkeeping rows, but no host submanifold primitive, selected Weinstein neighbourhood, Hamiltonian isotopy, Floer complex, global transversality theorem, or ambient compactness principle.

Proof. Start with the NameCert obligations of Definition 8319.2. They identify every accepted public read with the displayed rows $S, M, F, A, T, R, B, H, C, P, N$.

The symplectic side is forced through S, F, T. A consumer first reads the ambient symplectic source, then the restricted differential-form row, and only then the tangent-rank evidence that certifies the Lagrangian boundary visible to the packet.

The carried regular surface is forced through M, A, R, B. The manifold row supplies chart windows, the atlas-admission row names the finite locus, the regularity row stabilizes the rank

witnesses, and the bundle row supplies the local bookkeeping consumed by later mechanics and neighbourhood packets.

Transport and replay by H and C preserve the same route, while P and N provide provenance and local naming. Therefore the handoff reaches symplectic, Lagrangian-mechanics, and bundle-facing consumers without exporting a separate host submanifold primitive or any of the excluded ambient structures. □

Falsifiable prediction: Every accepted $\mathsf{RegularLagrangian}^{\uparrow}$ consumer that reaches a symplectic, mechanics, or bundle-facing boundary first displays the symplectic source, manifold chart, differential-form restriction, finite atlas admission, tangent-rank, regularity, bundle bookkeeping, transport, replay, provenance, and local NameCert rows.

Independence witness: symplectic, manifold, diffform, lagrangianmechanics, bundle: $\mathsf{RegularLagrangian}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds restricted symplectic form data, half-dimensional tangent-rank evidence, finite regularity witnesses, and bundle-local bookkeeping in one certificate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularLagrangian}^{\uparrow}$ packet is generated from a symplectic source row, manifold chart row, differential-form restriction, finite atlas admission, tangent-rank row, regularity row, bundle bookkeeping, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8319.1, Definition 8319.2, and Definition 8319.3 seed the finite regular Lagrangian route into symplectic, mechanics, and bundle-facing readback data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularLagrangian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host submanifold primitives, selected Weinstein neighbourhoods, Hamiltonian isotopies, Floer complexes, global transversality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, restricted-form exposure, tangent-rank evidence, regularity stability, bundle bookkeeping, transport, replay, provenance, and local naming for BEDC.Derived.RegularLagrangianUp.

Chapter 8320

A Concrete Naming Certificate for $\mathsf{CauchyCondensationModulus}^{\uparrow}$

Following Cauchy's condensation test and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationModulus}^{\uparrow}$ packet records the finite modulus handoff by which a displayed condensation schedule becomes a Cauchy/RegSeqRat/Real-facing tail budget. It sits after the condensation packet of chapter 1464, the modulus surface of chapter 292, the minimum-modulus route of chapter 5544, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite Cauchy-condensation modulus route; it is not a quotient of series tails, a selected infinite subsequence, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8320.1 (Cauchy condensation modulus carrier)**.** *A* $\mathsf{CauchyCondensationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, M, B, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *source row, including the monotone tail, dyadic block schedule, block-sum ledger, and tail-bound data.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *request row extracted from the condensation tail budget.*
3. *B is the finite condensation-block modulus row, recording which dyadic block bound answers the current tolerance request.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger opened by the block modulus.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those same windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing the dyadic radius, block comparison, denominator budget, and slack rows used by M and B.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after K, M, B, W, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for the condensation source, modulus request, block modulus, windows, readback, dyadic ledger, real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a tolerance request through K, M, B, W, Q, D, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationModulus}^{\uparrow}$, $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyMinModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, B, W, Q, D, E, H, C, P, N$. It has no coordinate for quotient series equality, selected subsequences, selected sums, countable choice, trichotomy on arbitrary tails, host real equality, or ambient completion.

Theorem 8320.2 (Cauchy condensation modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationModulus}^{\uparrow}$ *carrier*

$$G = (K, M, B, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCondensationModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the condensation source row, Cauchy-modulus request row, finite block-modulus row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise transport, displayed replay, package provenance, and local naming. The modulus row must be read after the* $\mathsf{CauchyCondensation}^{\uparrow}$ *source and before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reaches Q or E.*

Proof. Project the carrier of Definition 8320.1. The analytic rows are precisely the condensation source K, the modulus request M, the finite block modulus B, the windows W, the readback Q, the dyadic ledger D, and the seal E.

The condensation source is read first. It supplies the dyadic block schedule and tail-bound ledger consumed by the modulus request, while the $\mathsf{CauchyModulus}^{\uparrow}$ row records the finite tolerance request to be answered.

The block-modulus row B and dyadic ledger D then determine the finite windows W and the regular-rational readback Q. The terminal seal E is downstream of those rows, and H, C, P, N transport, replay, source, and name the same route.

Since no carrier coordinate stores quotient tails, selected sums, selected subsequences, countable choice, host real equality, or ambient completion, the listed rows exhaust the local certificate surface. □

Theorem 8320.3 (Cauchy condensation modulus handoff)**.** *Every Cauchy, RegSeqRat, or Real-facing consumer of an accepted* $\mathsf{CauchyCondensationModulus}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCondensation}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow}, \mathsf{RegularCauchyMinModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatC}$$

The handoff exposes $K, M, B, W, Q, D, E, H, C, P, N$ and exports no quotient series equality, selected subsequence, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the accepted carrier. The condensation source row K is the only row carrying the dyadic block schedule and tail-bound comparison data, so any consumer that wants a modulus handoff must first read K.

Next read the modulus side. The row M presents the Cauchy tolerance request, the row B answers it with a finite condensation-block budget, and the sibling $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ route supplies the displayed minimum-budget discipline when two local Cauchy windows share one request.

The window and readback rows are downstream of that modulus answer. The StreamName row W, RegSeqRat row Q, and DyadicRatCore ledger D expose the finite windows and tolerance comparisons before the Real seal E is consumed.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Thus every consumer factors through the displayed handoff and receives no host quotient, selected infinite object, choice schedule, or ambient completeness coordinate. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationModulus}^{\uparrow}$ read is refuted whenever a Cauchy, RegSeqRat, or Real-facing consumer reaches a window, readback, or seal without displaying $K, M, B, W, Q, D, E, H, C, P, N$.

Independence witness: cauchycondensation, cauchymodulus, regular-cauchy-min-modulus, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{CauchyCondensationModulus}^{\uparrow}$ because the packet jointly records condensation source rows, a modulus request, a finite block-modulus answer, windows, readback, dyadic tolerance, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationModulus}^{\uparrow}$ packet is generated from a CauchyCondensation source row, CauchyModulus request row, finite condensation-block modulus row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8320.1, Definition 8320.2, and Definition 8320.3 seed the finite handoff from condensation schedules to Cauchy, RegSeqRat, and Real-facing modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyCondensationModulusUp.CauchyCondensationModulusUp`

Bridge status: none

Not claimed: This chapter does not close quotient series equality, selected subsequences, selected sums, countable choice, trichotomy on arbitrary tails, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, condensation-source exposure, block-modulus exactness, window coverage, RegSeqRat readback, DyadicRatCore tolerance accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationModulusUp.

Chapter 8321

A Concrete Naming Certificate for $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$ packet records equality of real names as a finite epsilon–delta request and response surface. It sits on the L10 corridor through the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the RealEquality classifier boundary of chapter 1636. The packet names the displayed equality route only; it does not turn equality into quotient real equality or into host equality of selected representatives.

Definition 8321.1 (Bishop epsilon-delta real equality carrier)**.** *A* $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, \Delta, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the epsilon-request row, recording the finite positive tolerance requested by the equality consumer;*
2. *Δ is the delta-response row, giving the finite modulus answer used to compare the two real-name windows at tolerance Q;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that carries the dyadic comparison budget used by the response;*
4. *S is the pair of* $\mathsf{StreamName}^{\uparrow}$ *finite windows inspected for the two real names at the requested tolerance;*
5. *R is the pair of* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the displayed stream windows;*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row for the two terminal real-name boundaries, reached only after Q, Δ, D, S, R have been displayed;*
7. *H records the componentwise* hsame *transports for epsilon requests, delta responses, dyadic tolerances, stream windows, regular readbacks, real seals, and the local packet shape;*
8. *C records the displayed* Cont *replay from epsilon request through delta response, dyadic tolerance, finite windows, regular readbacks, Real seal, and local naming;*

9. *P is the* Pkg *provenance over* $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these rows. It has no coordinate for quotient real equality, selected representatives, trichotomy, countable choice, ambient Real completeness, or host equality of completed real objects.

Theorem 8321.2 (Bishop epsilon-delta real equality NameCert obligations)**.** *For every accepted* $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$ *carrier*

$$E = (Q, \Delta, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}}$ *surface consists exactly of carrier admission for the epsilon request Q, delta response Δ, dyadic tolerance row D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R, terminal* $\mathsf{Real}^{\uparrow}$ *seal L, componentwise* hsame *transport H, displayed* Cont *replay C, package provenance P, and local naming row N. Every equality-facing consumer must expose those rows before it can read the RealEquality boundary.*

Proof. Project the carrier of Definition 8321.1. Its equality data are the epsilon request, delta response, dyadic tolerance ledger, two finite stream windows, two regular rational readbacks, and the terminal Real seal row.

Read the route in dependency order. The request Q determines which tolerance must be answered, Δ supplies the displayed response budget, D records the dyadic tolerance used by that response, and S records the finite windows inspected under that budget.

The readback row R is then obtained from the displayed windows, and the seal row L is reached only after the readbacks have been exposed. The structural rows H, C, P, N transport, replay, source, and name the same finite packet, so the listed rows exhaust the local NameCert obligations. □

Theorem 8321.3 (Bishop epsilon-delta real equality Real seal nonescape)**.** *Every Real-facing equality read of an accepted* $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$ *packet factors through the finite route*

$$Q, \Delta, D, S, R, L, H, C, P, N.$$

The terminal $\mathsf{Real}^{\uparrow}$ *seal row L may be inspected only after the epsilon request, delta response, dyadic tolerance, StreamName windows, and RegSeqRat readbacks have been displayed. The packet exports no quotient real equality, selected representative, trichotomy, countable-choice schedule, ambient Real completeness, host equality, Lean verification claim, or bridge check.*

Proof. Start with the NameCert obligations of Definition 8321.2. They force every equality-facing consumer to expose the epsilon request, delta response, dyadic tolerance, finite windows, and regular readbacks before the Real seal row is available.

Now apply the sibling route through chapter 1636. That boundary accepts equality only through displayed finite windows, regular readbacks, and dyadic tolerance data, not through a quotient of completed streams or a host equality predicate.

Finally inspect the carrier coordinates. Since no coordinate names a selected representative, trichotomy principle, countable-choice schedule, ambient completeness theorem, Lean target, or standard bridge, none of those objects can be exported by the local Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}$ packet is generated from a finite epsilon request, delta response, dyadic tolerance row, finite StreamName windows, RegSeqRat readbacks, terminal Real seal, componentwise transport, continuation replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite epsilon–delta Real-equality carrier, its local NameCert obligations, and the Real-seal nonescape boundary along the L10 Real and Completion corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopEpsilonDeltaRealEquality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not close quotient real equality, selected representatives, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Reaching obligation closure requires a Lean carrier and checked proof that every accepted equality consumer exposes the same epsilon request, delta response, dyadic tolerance, finite-window, readback, and Real-seal route.

Chapter 8322

A Concrete Naming Certificate for $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$

Bishop-style constructive compactness treats an ϵ-net as displayed finite data rather than as a choice made from an ambient compactness theorem. The $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ packet records that finite selector surface for a $\mathsf{CompactMetric}^{\uparrow}$ object: it reads the $\mathsf{Met}^{\uparrow}$ carrier, a $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ positive-radius ledger, finite $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal before exporting any selected net row. The selector is consumed only after the compact source of chapter 129, the metric readback of chapter 57, and the radius ledger of chapter 452 have exposed the windows later read through StreamName, RegSeqRat, and Real rows. It is a selector carrier for compact Real-and-Completion work, not a quotient cover, a hidden subcover choice, or a host compactness principle.

Definition 8322.1 (Finite epsilon-net choice carrier)**.** *A* $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, M, \Lambda, W, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compact presentation supplies finite observation data.*
2. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *row, keeping the metric carrier and distance readback visible.*
3. *Λ is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *row that provides the positive radius ledger used by the net request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family naming the candidate centers inspected by the selector.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row used to compare displayed radii and metric distances.*
7. *S is the selected finite epsilon-net row, recorded only after K, M, Λ, W, Q, R have been displayed.*
8. *H records componentwise* hsame *transport for the compact source, metric row, Lebesgue-number ledger, windows, rational readback, Real seal, selected-net row, replay, provenance, and local name.*
9. *C stores displayed* Cont *replay from a finite radius request through $K, M, \Lambda, W, Q, R, S, H$.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an arbitrary open cover, a choice-selected subcover, quotient metric equality, host compactness, countable choice, or a selected completion object.

Theorem 8322.2 (Finite epsilon-net choice NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ *carrier*

$$E = (K, M, \Lambda, W, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, compact source exposure through K, metric readback through M, finite positive-radius exposure through Λ, finite window exposure through W, rational readback through Q, Real comparison through R, selected-net export through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8322.1. The analytic rows are $K, M, \Lambda, W, Q, R, S$, and the structural rows are H, C, P, N.

The compact source and metric rows are read first. The selector therefore receives a displayed compact metric presentation rather than an arbitrary cover or host compactness assertion.

Next the finite Lebesgue-number row Λ fixes the positive-radius ledger. The windows W, rational readback Q, and Real seal R are downstream of that ledger, so the selector row S can record only centers inspected through the displayed finite data.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores open-cover choice, quotient metric equality, host compactness, countable choice, or a selected completion object, the listed rows exhaust the certificate surface. □

Theorem 8322.3 (Finite epsilon-net compact handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ *packet must expose the finite route*

$$K, M, \Lambda, W, Q, R, S, H, C, P, N$$

before using the selected epsilon-net row. In particular, the handoff from compactness to a finite net passes through the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before any downstream Real-and-Completion consumer may inspect S.*

Proof. Start with the NameCert obligations. They make $K, M, \Lambda, W, Q, R, S, H, C, P, N$ the complete public surface of the accepted selector packet.

The positive-radius step is forced through Λ. A consumer cannot read S until the compact source, metric row, and finite Lebesgue-number ledger have already determined the displayed radius budget.

The window and readback rows then determine which finite centers are visible. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows provide the finite observations, and the $\mathsf{Real}^{\uparrow}$ seal records the displayed metric comparison.

Thus the selected-net row is a finite handoff from compact metric data to Real-facing consumers. Because the carrier lacks open-cover choice, quotient metric equality, host compactness, countable choice, and selected completion coordinates, no stronger compactness principle is exported. □

Falsifiable prediction: An accepted $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ consumer that reaches a selected epsilon-net row without reading $K, M, \Lambda, W, Q, R, S, H, C, P, N$ refutes the selector carrier.

Independence witness: compactmetric, metricspace, finitelebesguenumber, streamname, regseqrat, real: none of these siblings alone carries the joint finite selector packet containing compact source, metric readback, positive-radius ledger, observation windows, regular-rational readback, Real comparison, selected net, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteEpsilonNetSelector}^{\uparrow}$ packet is generated from a CompactMetric source row, MetricSpace readback, FiniteLebesgueNumber positive-radius ledger, StreamName finite windows, RegSeqRat rational readback, Real seal, selected epsilon-net row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8322.1, Definition 8322.2, and Definition 8322.3 seed the finite selector route from constructive compactness to Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNetSelector}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, arbitrary open-cover subchoice, quotient metric equality, host compactness, countable choice, or selected completion objects.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger compatibility, finite-window readback, Real-seal comparison, and selector nonescape for BEDC.Derived.FiniteEpsilonNetSelectorUp.

Chapter 8323

A Concrete Naming Certificate for $\mathsf{HyperbolicBigonThinness}^{\uparrow}$

The $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ packet realizes the bigon-thinness branch of chapter 9463 as finite endpoint, side, metric, boundary-shadow, and Real-seal NameCert data. It sits beside the metric route of chapter 3474, the boundary route of chapter 3491, the visual comparison route of chapter 3866, and the ambient finite metric discipline of chapter 4230. The packet records a displayed two-sided geodesic boundary comparison; it does not assert global curvature, arbitrary geodesic uniqueness, host properness, or a selected geodesic representative outside the finite rows.

Definition 8323.1 (Hyperbolic bigon thinness carrier)**.** *A* $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row carrying the hyperbolic metric read.*
2. *G is the* $\mathsf{Met}^{\uparrow}$ *row that keeps the underlying finite metric carrier visible.*
3. *∂ is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *row naming the boundary endpoints used by the bigon.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row used to compare the boundary shadows.*
5. *E_0 and E_1 are the two displayed endpoint rows of the bigon.*
6. *L_0 and L_1 are the two geodesic-side rows whose fellow-traveller comparison is being read.*
7. *D is the finite distance ledger comparing side points through the hyperbolic metric.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed thinness tolerance.*
9. *Σ is the boundary-shadow ledger recording that the visual-metric read and the geodesic-boundary read refer to the same endpoints.*
10. *H records componentwise* hsame *transport for metric, boundary, visual, endpoint, side, distance, Real-seal, shadow, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from an endpoint-pair request through $M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBigonThinness}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

13. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicBigonThinness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for global CAT curvature, arbitrary geodesic uniqueness, host properness, selected geodesic representatives, quotient boundary equality, or an unbounded visual compactification.

Theorem 8323.2 (Hyperbolic bigon thinness NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ *carrier*

$$B = (M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBigonThinness}^{\uparrow}}$ *surface consists exactly of hyperbolic-metric exposure through M, ambient metric exposure through G, geodesic-boundary exposure through ∂, visual-metric exposure through V, endpoint exposure through E_0, E_1, geodesic-side exposure through L_0, L_1, distance-ledger exposure through D, Real thinness tolerance through R, boundary-shadow compatibility through Σ, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8323.1. The geometric rows are $M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma$, and the structural rows are H, C, P, N.

The metric and boundary rows are read before the sides. Thus the two geodesic-side rows are tied to the displayed hyperbolic metric, the underlying metric carrier, and the same geodesic-boundary endpoints.

The visual-metric and shadow rows then compare the endpoint data. The distance ledger D and Real seal R record only the finite thinness tolerance visible through those rows.

Finally the transport, replay, provenance, and local naming rows keep the same packet intact. Since no carrier coordinate stores global curvature, arbitrary geodesic uniqueness, host properness, selected geodesic representatives, quotient boundary equality, or visual compactification, the listed rows exhaust the local obligations. □

Theorem 8323.3 (Hyperbolic bigon thinness boundary route)**.** *Every boundary-facing consumer of an accepted* $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ *packet must expose*

$$M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma, H, C, P, N$$

before using a bigon-thinness conclusion. The read reaches the conclusion only after the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed in that route.*

Proof. Begin with the NameCert obligation theorem. It makes the metric, boundary, visual, endpoint, geodesic-side, distance, Real-seal, shadow, transport, replay, provenance, and local naming rows the complete public surface.

The endpoint rows E_0, E_1 are read through ∂ and V. Thus a consumer cannot detach the boundary-shadow comparison from the displayed geodesic boundary or visual metric.

The side rows L_0, L_1 are then compared only through M, G, D, R. The hyperbolic metric, ambient metric carrier, finite distance ledger, and Real seal provide the thinness tolerance as a displayed row, not as a host global geometry fact.

Therefore the bigon-thinness conclusion remains a finite boundary route. It cannot be retyped as global curvature, arbitrary geodesic uniqueness, host properness, selected geodesic representation, quotient boundary equality, or an unbounded visual compactification. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ boundary consumer that reads bigon thinness without exposing $M, G, \partial, V, E_0, E_1, L_0, L_1, D, R, \Sigma, H, C, P, N$ refutes the carrier surface.

Independence witness: hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-visual-metric, metricspace, real: none of these siblings alone carries the joint endpoint, two-side, distance-ledger, boundary-shadow, Real-seal, transport, replay, provenance, and local NameCert packet required by $\mathsf{HyperbolicBigonThinness}^{\uparrow}$.

Future Lean verification for this packet must supply nontriviality and field-faithfulness taste gates for the same carrier surface before any bridge can advertise a schema-level hyperbolic-thinness export.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBigonThinness}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBigonThinness}^{\uparrow}$ packet is generated from HyperbolicMetric, MetricSpace, HyperbolicGeodesicBoundary, HyperbolicVisualMetric, endpoint, geodesic-side, distance-ledger, Real-seal, boundary-shadow, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBigonThinness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8323.1, Definition 8323.2, and Definition 8323.3 seed the finite boundary route for hyperbolic bigon thinness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBigonThinness}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, global curvature, arbitrary geodesic uniqueness, host properness, selected geodesic representatives, quotient boundary equality, or visual compactification.

Upgrade path: Obligation closure requires checked metric exposure, endpoint and side visibility, boundary-shadow compatibility, Real thinness tolerance, and taste-gate nonescape for BEDC.Derived.HyperbolicBigonThinnessUp.

Chapter 8324

A Concrete Naming Certificate for $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$

Following Bishop–Bridges completion theory and Mac Lane's unit laws, the $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ packet records the finite BEDC surface saying that applying the completion unit to an already displayed completion endpoint leaves the same named endpoint route. It sits after the completion carrier of chapter 1506, the unit packet of chapter 7904, the separated-metric reflection of chapter 1790, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the eta law as finite NameCert data; it is not a quotient-completion equality, a selected representative theorem, or an ambient completeness principle.

Definition 8324.1 (Metric completion eta-law carrier)**.** *A* $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, T, U, S, R, Q, Z, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row.*
2. *T is the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *endpoint row already accepted by the completion carrier.*
3. *U is the* $\mathsf{MetricCompletionUnit}^{\uparrow}$ *row applied to the same displayed endpoint route.*
4. *S is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *row ensuring that comparison of the two endpoint reads is separated before any equality-like claim is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row witnessing that both endpoint routes expose the same finite regular-rational windows.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, U, S, R have been displayed.*
7. *Z is the eta-law exactness row binding the direct endpoint route and the unit-applied endpoint route to the same displayed completion name.*
8. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionEtaLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, U, S, R, Q, Z, H, C, P, N$. It has no coordinate for quotient classes of Cauchy names, selected representatives, host completion equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8324.2 (Metric completion eta-law NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ *carrier* $E = (X, T, U, S, R, Q, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionEtaLaw}^{\uparrow}}$ *surface consists exactly of source metric exposure, displayed completion endpoint exposure, unit-row exposure, separated comparison, regular readback, real-seal exposure, eta-law exactness, transport, replay, provenance, and local naming. Every consumer of the eta law must read those rows before it treats the direct completion endpoint and the unit-applied endpoint as the same named route.*

Proof. Project the carrier of Definition 8324.1. The rows X, T, U name the metric source, the displayed completed endpoint, and the unit-applied endpoint route.

Next read the comparison and endpoint rows S, R, Q. Separation is supplied by S, finite regular-rational agreement by R, and terminal real-name exposure by Q; none of these rows contains quotient equality or a selected representative.

Finally use the eta-law exactness row Z and the support rows H, C, P, N. They bind the direct route and the unit-applied route to the same displayed completion name under transport, replay, provenance, and local naming, so the listed rows exhaust the NameCert surface. □

Theorem 8324.3 (Metric completion eta-law unit handoff)**.** *Let* $E = (X, T, U, S, R, Q, Z, H, C, P, N)$ *be an accepted* $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ *carrier. Any completion-facing consumer that enters through the unit row* U *must pass through the already displayed completion endpoint* T*, the separated comparison row* S*, the regular-rational readback* R*, and the eta-law exactness row* Z *before reading the terminal seal* Q*. The handoff therefore exposes the unit law as a finite route over the existing completion name, not as a quotient-completion identification or a host-side equality of completed points.*

Proof. Begin with the unit row U. By the carrier definition it is not an independent completion endpoint; it is attached to the same source metric row X and the displayed completion endpoint row T.

Read the separated comparison row S and regular-rational row R. These rows keep the direct endpoint and the unit-applied endpoint inside the same finite comparison surface before the seal Q can be consumed.

The exactness row Z supplies the eta-law handoff, and H, C, P, N only preserve that displayed route. Thus a consumer receives a named unit law for the existing completion endpoint without acquiring quotient completion equality, selected representatives, countable choice, host equality, or ambient real completeness. □

Independence witness: metriccompletion, metric-completion-unit, separatedmetricreflection, regseqrat, real: no listed sibling is bijective to $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ because this packet jointly binds a displayed completion endpoint, its unit-applied route, separated comparison, regular readback, real seal, eta-law exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionEtaLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionEtaLaw}^{\uparrow}$ packet is generated from a MetricSpace source row, a MetricCompletion endpoint row, a MetricCompletionUnit row, a SeparatedMetricReflection comparison row, RegSeqRat readback, a RealUp seal, eta-law exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionEtaLaw}^{\uparrow}}$, seed)

Scope closed: Definition 8324.1, Definition 8324.2, and Definition 8324.3 seed the finite completion-unit eta-law surface over the existing L10 completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionEtaLaw}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient-completion equality, selected Cauchy representatives, host equality of completed points, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-row exposure, separated comparison, regular readback agreement, eta-law exactness, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionEtaLawUp.

Chapter 8325

A Concrete Naming Certificate for $\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}$

Following Aronszajn and Smith on invariant subspaces for compact operators, the $\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}$ packet records the finite BEDC surface by which a compact operator on a Banach-facing scalar space exposes an invariant-subspace witness before any functional-analysis consumer reads the result. It sits over the compact-operator carrier of chapter 1952, the bounded-linear-operator route of chapter 1644, the functional-analysis ledger of chapter 94, the Banach carrier of chapter 90, the real scalar seal of chapter 38, and the complex scalar row of chapter 39. The chapter names the invariant-subspace certificate as finite NameCert data; it does not assert a full spectral theorem, choose an eigenvector, quotient a function space, import Hilbert-space orthogonality, use countable choice, or claim an ambient completeness principle.

Definition 8325.1 (Aronszajn–Smith invariant-subspace carrier)**.** *An* $\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, F, B, R, Z, W, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactOperator}^{\uparrow}$ *row recording compactness of the operator route.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row recording bounded linear action on the same displayed source.*
3. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *ledger row that admits operator and subspace reads only through the finite packet.*
4. *B is the* $\mathsf{Banach}^{\uparrow}$ *carrier row for the domain in which the invariant-subspace witness is read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-seal row used by norm and radius comparisons.*
6. *Z is the* $\mathsf{Complex}^{\uparrow}$ *scalar row, stored as finite BEDC scalar data rather than as an ambient algebraic closure claim.*
7. *W is the invariant-subspace witness row, recording a displayed nonzero proper subspace candidate.*
8. *L is the invariance ledger row saying that the bounded compact operator route sends the displayed witness row back through itself.*

9. *H records componentwise* hsame *transport for K, T, F, B, R, Z, W, L and for the package and local naming rows.*
10. *C records displayed* Cont *replay from compact-operator admission through bounded linearity, Banach scalar sealing, witness exposure, and invariance readback.*
11. *P is the* Pkg *provenance over* AronszajnSmithInvariantSubspace$^{\uparrow}$, CompactOperator$^{\uparrow}$, BoundedLinearOperator$^{\uparrow}$, FunctionalAnalysis$^{\uparrow}$, Banach$^{\uparrow}$, Real$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{AronszajnSmithInvariantSubspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a full spectral theorem, selected eigenvectors, quotient function spaces, host Hilbert orthogonality, countable choice, or ambient completeness.

Theorem 8325.2 (Aronszajn–Smith invariant-subspace NameCert obligations)**.** *For every accepted* AronszajnSmithInvariantSubspace$^{\uparrow}$ *carrier*

$$A = (K, T, F, B, R, Z, W, L, H, C, P, N),$$

the local NameCert$_{\text{AronszajnSmithInvariantSubspace}^{\uparrow}}$ *surface consists exactly of compact-operator admission, bounded-linear-operator exposure, functional-analysis ledger exposure, Banach carrier exposure, real and complex-facing scalar sealing, invariant-subspace witness exposure, invariance-ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $K, T, F, B, R, Z, W, L, H, C, P, N$.*

Proof. Project the carrier of Definition 8325.1. The analytic rows are K, T, F, B, R, Z, W, L, and the structural rows are H, C, P, N.

The compact and bounded-linear rows are read first. They force any operator-facing consumer to enter through chapter 1952 and chapter 1644 before the witness row is visible.

Next the functional-analysis, Banach, and scalar rows expose the source discipline. The rows F, B, R, Z keep the normed carrier and scalar comparisons finite, with no coordinate for spectral decomposition or host-side algebraic closure.

Finally W and L provide the displayed witness and its invariance readback, while H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no row for selected eigenvectors, quotient spaces, Hilbert orthogonality, countable choice, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8325.3 (Aronszajn–Smith compact-operator route)**.** *Every functional-analysis or Banach-facing consumer of an accepted* AronszajnSmithInvariantSubspace$^{\uparrow}$ *packet must expose the compact-operator row K, the bounded-linear row T, the functional-analysis ledger F, the Banach row B, the scalar rows R, Z, the invariant-subspace witness W, the invariance ledger L, and the support rows H, C, P, N before reading the invariant-subspace certificate. The route reaches the sibling surfaces of chapter 1952, chapter 1644, chapter 94, chapter 90, and chapter 38 only through those displayed rows.*

Proof. Start with the NameCert obligations of Definition 8325.2. They identify $K, T, F, B, R, Z, W, L, H, C, P, N$ as the complete public surface of the accepted packet.

The operator handoff passes through K and T. Compactness is therefore consumed as a displayed compact-operator row, while bounded linearity is consumed as a displayed bounded-linear-operator row over the same packet.

The source and scalar handoff passes through F, B, R, Z. These rows expose the functional-analysis ledger, Banach carrier, and scalar comparison data before the witness row can be read.

The witness handoff is W, L under H, C, P, N. Thus any consumer that reads the invariant-subspace certificate remains inside the finite packet and cannot replace it by a spectral theorem, eigenvector choice, quotient function space, Hilbert orthogonality, countable choice, or ambient completeness argument. □

Falsifiable prediction: An accepted $\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}$ consumer that reads the invariant-subspace witness without first exposing $K, T, F, B, R, Z, W, L, H, C, P, N$ refutes the carrier.

Independence witness: compactoperator, boundedlinearoperator, functionalanalysis, banach, real: none of these siblings alone carries the joint packet containing compact-operator admission, bounded-linear action, functional-analysis ledger, Banach source, real and complex-facing scalar rows, invariant-subspace witness, invariance readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}}$.

Constructive story: An $\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}$ packet is generated from a CompactOperator row, BoundedLinearOperator row, FunctionalAnalysis ledger, Banach source, Real scalar seal, Complex scalar row, invariant-subspace witness row, invariance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8325.1, Definition 8325.2, and Definition 8325.3 seed the finite compact-operator route from Aronszajn–Smith invariant-subspace data to functional-analysis and Banach-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AronszajnSmithInvariantSubspace}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, a full spectral theorem, selected eigenvectors, quotient function spaces, Hilbert orthogonality, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-operator exposure, bounded-linear exposure, scalar sealing, invariant-subspace witness exposure, invariance readback, transport, replay, provenance, and local naming for BEDC.Derived.AronszajnSmithInvariantSubspaceUp.

Chapter 8326

A Concrete Naming Certificate for $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$

Following Kantorovich and Rubinstein, the $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$ packet records the finite dual-test surface that compares a Wasserstein transport-cost row with a Lipschitz test-function row. It consumes the transport-plan carrier of chapter 5197, the finite-support linear-programming surface of chapter 326, the probability rows of chapter 184, the metric source of chapter 57, the test-function carrier of chapter 3762, and the terminal scalar seal of chapter 38. The chapter names only the displayed duality handoff; it does not assert measure completion, arbitrary Polish-space duality, selected optimal plans, selected extremal functions, quotient function-space equality, or an ambient real-valued supremum.

Definition 8326.1 (Kantorovich–Rubinstein duality carrier)**.** *A* $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (W, T, \mu, \nu, M, L, R, B, H, C, P, N)$$

with the following visible rows:

1. *W is a* $\mathsf{Wasserstein}^{\uparrow}$ *transport-plan row whose cost ledger is already displayed.*
2. *T is an* $\mathsf{OptimalTransport}^{\uparrow}$ *finite-support certificate row exposing the primal finite linear-programming surface.*
3. *μ and ν are* $\mathsf{ProbSpace}^{\uparrow}$ *source and target probability rows read by both the primal and dual ledgers.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *source row used by the transport cost and the Lipschitz constraint.*
5. *L is a* $\mathsf{LipschitzFunction}^{\uparrow}$ *test-function row whose displayed constant is bounded by* 1 *against M.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar row comparing the transport-cost ledger with the displayed test-function difference.*
7. *B is the dual-bound ledger: every accepted read of R factors through the* 1*-Lipschitz test row L, the probability rows μ, ν, and the primal transport row W.*
8. *H records componentwise* hsame *transport for $W, T, \mu, \nu, M, L, R, B, C, P, N$.*
9. *C stores displayed* Cont *replay from the transport and test-function rows to the scalar bound row.*

10. *P is the* Pkg *provenance over* $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$, $\mathsf{Wasserstein}^{\uparrow}$, $\mathsf{OptimalTransport}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{KantorovichRubinsteinDuality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed measures, arbitrary bounded continuous functions, an unlisted supremum, a selected optimal coupling, a selected extremal Lipschitz function, quotient function-space equality, or host real equality.

Theorem 8326.2 (Kantorovich–Rubinstein NameCert obligations)**.** *For every accepted* KantorovichRubinsteinDuali *carrier*

$$D = (W, T, \mu, \nu, M, L, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KantorovichRubinsteinDuality}^{\uparrow}}$ *surface consists exactly of carrier admission, Wasserstein transport-cost exposure through W, finite primal certificate exposure through T, probability marginal exposure through μ, ν, metric readback through M,* 1*-Lipschitz test exposure through L, scalar comparison through R, dual-bound accounting through B, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8326.1. Its analytic rows are $W, T, \mu, \nu, M, L, R, B$, and its structural rows are H, C, P, N.

The primal side is forced by W and T. The Wasserstein row supplies the displayed transport-cost ledger, while the optimal-transport row supplies the finite linear-programming certificate that bounds the same finite support surface.

The dual side is forced by L, μ, ν, M, R, B. The Lipschitz row is read against the metric row M, the probability rows provide the two displayed integrals or finite sums, and B records only the resulting scalar comparison in R.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores completed measures, arbitrary bounded continuous functions, an unlisted supremum, selected optimizers, quotient function-space equality, or host real equality, the listed rows exhaust the local certificate surface. □

Theorem 8326.3 (Kantorovich–Rubinstein Lipschitz-test handoff)**.** *Every consumer that reads a scalar dual bound from an accepted* $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$ *carrier must first expose the finite route*

$$W, T, \mu, \nu, M, L, R, B, H, C, P, N.$$

In particular, the Lipschitz-test handoff cannot reach the $\mathsf{Real}^{\uparrow}$ *comparison row R without reading the* $\mathsf{Wasserstein}^{\uparrow}$ *transport ledger, the* $\mathsf{OptimalTransport}^{\uparrow}$ *finite primal certificate, the two* $\mathsf{ProbSpace}^{\uparrow}$ *marginals, the* $\mathsf{Met}^{\uparrow}$ *source, and the* $\mathsf{LipschitzFunction}^{\uparrow}$ *test row.*

Proof. Start from the NameCert obligations. They make the displayed carrier route the complete public surface of the accepted packet.

The transport side must be read before the scalar comparison. The rows W and T expose the primal cost and finite-support certificate, while μ, ν and M supply the probability and metric data shared with the dual test.

The Lipschitz side then supplies the only dual-test row. The row L carries the 1-Lipschitz constraint against M, and B records the comparison that reaches R.

Thus any consumer of R has already traversed the listed finite route. Because the carrier has no coordinate for completed measures, unlisted test functions, selected optimizers, quotient

function-space equality, or host real equality, no stronger Kantorovich–Rubinstein duality theorem is exported. □

Falsifiable prediction: An accepted $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$ consumer that reaches the scalar dual-bound row without reading $W, T, \mu, \nu, M, L, R, B, H, C, P, N$ refutes the duality carrier.

Independence witness: wasserstein, optimaltransport, probspace, metric, lipschitz-function, real: none of these siblings alone carries the joint packet containing transport cost, finite primal certificate, two probability marginals, metric source, Lipschitz test row, scalar comparison, dual-bound ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{KantorovichRubinsteinDuality}^{\uparrow}}$.

Constructive story: A $\mathsf{KantorovichRubinsteinDuality}^{\uparrow}$ packet is generated from a Wasserstein transport-cost row, an OptimalTransport finite primal certificate, two ProbSpace marginal rows, a Metric source row, a LipschitzFunction test row, a Real scalar comparison row, a dual-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KantorovichRubinsteinDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8326.1, Definition 8326.2, and Definition 8326.3 seed the displayed finite duality route from transport cost to Lipschitz-test scalar comparison.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KantorovichRubinsteinDuality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, measure completion, arbitrary Polish-space duality, selected optimal plans, selected extremal functions, quotient function-space equality, or ambient $\mathsf{Real}^{\uparrow}$ supremum claims.

Upgrade path: Obligation closure requires checked carrier admission, transport-cost exposure, finite primal certificate exposure, marginal readback, metric compatibility, Lipschitz-test compatibility, scalar comparison nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KantorovichRubinsteinDualityUp.

Chapter 8327

A Concrete Naming Certificate for $\mathsf{FiniteContinuedFraction}^{\uparrow}$

Following Hardy–Wright on finite continued fractions and rational convergents, the $\mathsf{FiniteContinuedFraction}^{\uparrow}$ packet records the finite-prefix object that sits between rational tree navigation and L10 rational readback. It is a bounded partial-quotient carrier whose endpoint can be consumed as a displayed $\mathsf{Rat}^{\uparrow}$ row before any infinite continued-fraction tail, quotient real equality, or choice-selected limit is present. The packet is positioned beside the rational arithmetic carrier of chapter 37, the Stern–Brocot tree of chapter 1704, the Farey-neighbor route of chapter 1759, the finite-prefix continued-fraction surface of chapter 342, the convergent ledger of chapter 1777, the dyadic tolerance core of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38.

Definition 8327.1 (FiniteContinuedFraction carrier)**.** *A* $\mathsf{FiniteContinuedFraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, N, D, Q, S, B, R, E, H, C, P, L)$$

with the following displayed rows:

1. *A is the finite partial-quotient spine, including its length row and the displayed prefix order.*
2. *N and D are the* $\mathsf{Rat}^{\uparrow}$ *numerator and denominator recurrence rows for the last displayed convergent.*
3. *Q is the normalized* $\mathsf{Rat}^{\uparrow}$ *endpoint read from N and D.*
4. *S is the sibling route row tying the prefix to* $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, *and* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$.
5. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row used when the endpoint is exposed to a regular rational consumer.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational-readback row built from the finite endpoint and dyadic budget.*
7. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R has been displayed.*
8. *H records componentwise* hsame *transport for partial quotients, rational endpoints, sibling routes, dyadic budgets, readback rows, the seal row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a finite-prefix request through $A, N, D, Q, S, B, R, E, H, P, L$.*

10. *P is the* Pkg *provenance row over* $\mathsf{FiniteContinuedFraction}^{\uparrow}$ *and all listed sibling carriers, and L is the local* $\mathsf{NameCert}_{\mathsf{FiniteContinuedFraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an infinite tail, quotient equality of real names, a selected limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8327.2 (FiniteContinuedFraction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteContinuedFraction}^{\uparrow}$ *carrier $F = (A, N, D, Q, S, B, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteContinuedFraction}^{\uparrow}}$ *surface consists exactly of the following ten obligation rows:*

1. *carrier admission for the displayed finite partial-quotient spine and its length row;*
2. *recurrence admission for the numerator and denominator rows N, D;*
3. *rational endpoint normalization through the* $\mathsf{Rat}^{\uparrow}$ *row Q;*
4. *sibling compatibility with* $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, *and* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *through S;*
5. *dyadic budget exposure through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row B;*
6. *rational-readback exposure through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
7. *terminal real-seal discipline for the optional* $\mathsf{Real}^{\uparrow}$ *row E;*
8. *componentwise transport through H;*
9. *displayed replay and package provenance through C and P;*
10. *local naming through L, with no extra obligation outside the displayed packet.*

Proof. Carrier and recurrence admission are projections from Definition 8327.1. The prefix row A fixes the finite list being inspected, while N and D record the last convergent by displayed recurrence data.

The rational endpoint is read through Q. Since Q is a $\mathsf{Rat}^{\uparrow}$ row attached to the displayed numerator and denominator rows, rational readback is available before any infinite continued-fraction object is mentioned.

Sibling compatibility is carried by S. The Stern–Brocot and Farey rows supply finite rational navigation, while the continued-fraction and convergent rows supply the prefix and recurrence interpretation of the same endpoint.

The L10 readback route is then forced through B, R, E. The dyadic budget row precedes the regular rational row, and the optional real seal is terminal, so the certificate exports no quotient real equality, selected limit, or choice principle.

Finally, H, C, P, L transport, replay, package, and name the same finite packet. No additional coordinate remains from which another local obligation could be read. □

Theorem 8327.3 (FiniteContinuedFraction rational readback)**.** *Every accepted* $\mathsf{FiniteContinuedFraction}^{\uparrow}$ *packet exposes its rational endpoint through the finite route*

$$A, N, D \longmapsto Q \longmapsto B \longmapsto R \longmapsto E.$$

Thus an L10 consumer may read the endpoint as $\mathsf{Rat}^{\uparrow}$ *data and, when requested, as a* $\mathsf{RegSeqRat}^{\uparrow}$ *row with a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The consumer receives no infinite continued fraction, no quotient real equality, no countable-choice selector, and no ambient completeness theorem.*

Proof. Start with the carrier projection. The only rational endpoint coordinate is Q, and it is downstream of the finite prefix and recurrence rows A, N, D.

The dyadic and regular rows are ordered after the endpoint. The row B supplies the finite tolerance budget, and R is the regular rational readback that consumes that budget together with Q.

The real-facing row E is terminal. Because the carrier has no coordinate for an infinite tail or a completed quotient real, every downstream read remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteContinuedFraction}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteContinuedFraction}^{\uparrow}$ packet is generated from a finite partial-quotient spine, rational recurrence rows, a normalized rational endpoint, sibling navigation rows, dyadic budget, regular rational readback, optional real seal, transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteContinuedFraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8327.1, Definition 8327.2, and Definition 8327.3 seed the finite-prefix rational-readback corridor between continued fractions and L10 rational data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteContinuedFraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite continued fractions, quotient real equality, selected limits, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recurrence exactness, rational endpoint normalization, sibling compatibility, dyadic budget exposure, regular rational readback, transport, replay, provenance, and local naming for BEDC.Derived.FiniteContinuedFractionUp.

Chapter 8328

A Concrete Naming Certificate for $\mathsf{SphericalCompletion}^{\uparrow}$

Following Kaplansky and Bourbaki on spherically complete ultrametric spaces, the $\mathsf{SphericalCompletion}^{\uparrow}$ packet records only the constructive finite nested-ball-chain completion boundary. It sits beside the constructive ultrametric completion route of chapter 5204, the metric completion surface of chapter 1506, the StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the terminal real seal of chapter 38. The packet does not assert maximal spherical completeness, a Zorn-style extension, selected representatives of a host completion, or quotient equality of Cauchy names.

Definition 8328.1 (Spherical completion carrier)**.** *A* $\mathsf{SphericalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (U, B, R, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the ultrametric source row whose balls and strong-triangle comparisons are read before any completion-facing request.*
2. *B is the finite nested ball-chain row, listing the displayed ball centers and inclusions visible to the packet.*
3. *R is the radius ledger, recording the finite descent requests attached to the nested chain.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to expose finite observations of candidate names.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational radius and distance data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal that records the terminal real-distance comparison available to completion consumers.*
7. *H records componentwise* hsame *transport for the ultrametric source, ball chain, radius ledger, stream window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each completion-facing request through $U, B, R, S, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SphericalCompletion}^{\uparrow}$, $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SphericalCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite nested-ball-chain completion boundary. It has no coordinate for maximal spherical completeness, Zorn extension, quotient completion classes, selected representatives, host ball intersections, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8328.2 (Spherical completion NameCert obligations)**.** *Every accepted* $\mathsf{SphericalCompletion}^{\uparrow}$ *carrier of Definition 8328.1 exposes the following finite obligation rows:*

1. *carrier admission for the displayed ultrametric source* U*, nested ball chain* B*, radius ledger* R*, StreamName window* S*, RegSeqRat readback* Q*, Real seal* E*, transport* H*, replay* C*, provenance* P*, and local NameCert row* N*;*
2. *nested-chain monotonicity, requiring every later displayed ball in* B *to be read as contained in the earlier finite rows of the same chain;*
3. *radius-descent visibility, requiring each radius comparison used by a completion-facing consumer to pass through* R *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*;*
4. *readback and seal handoff, requiring every real-distance conclusion to pass from* S *through* Q *into the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*;*
5. *nonescape from Zorn arguments, maximal-choice extensions, host intersections, quotient completion classes, selected representatives, countable choice, and ambient completeness.*

Proof. Project the carrier tuple. The ten displayed rows are the whole admitted surface, so carrier admission cannot introduce a separate completion object or a hidden ball-intersection row.

Read nested-chain monotonicity before any radius request. Since the only chain coordinate is B, each inclusion or later-ball request is compared against the earlier displayed rows of B and transported componentwise by H.

Next use the radius ledger, readback, and seal rows. A completion-facing request reaches the radius data through R, exposes the finite StreamName window S, reads the rational comparison through Q, and only then consumes the terminal Real seal E.

Finally inspect the replay, provenance, and local naming rows. The displayed Cont route C, Pkg provenance P, and local $\mathsf{NameCert}$ row N name exactly that finite route and contain no coordinate for Zorn, maximal choice, host quotients, selected representatives, or ambient completeness. □

Theorem 8328.3 (Spherical completion ball-chain nonescape)**.** *Every completion-facing read of an accepted* $\mathsf{SphericalCompletion}^{\uparrow}$ *packet factors through the displayed finite ball-chain rows*

$$U, B, R, S, Q, E, H, C, P, N.$$

In particular, a consumer may not replace the chain row B *by a maximal nest, the radius ledger* R *by a hidden convergence modulus, the readback rows* S, Q, E *by a quotient completion class, or the local certificate* N *by a host spherical-completeness theorem.*

Proof. Begin with the carrier admission row of Definition 8328.2. It identifies $U, B, R, S, Q, E, H, C, P, N$ as the complete packet inventory.

Use the nested-chain and radius-visibility rows next. They force every ball-chain read through B and every radius descent through R before the StreamName and RegSeqRat rows are available.

Then apply the readback and seal handoff. The only completion-facing Real read passes through S, Q, E, so quotient classes, selected representatives, and host limits have no carrier coordinate from which to enter.

The nonescape row excludes the remaining classical routes. Since H, C, P, N only transport, replay, record provenance, and name the displayed finite route, every completion-facing read factors through the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{SphericalCompletion}^{\uparrow}$ completion-facing read displays ultrametric source, finite nested ball chain, radius ledger, StreamName window, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any consumer can use the boundary.

Independence witness: constructive-ultrametric-completion, metriccompletion, streamname, regseqrat, real: $\mathsf{SphericalCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a finite nested ball-chain ledger, radius descent, stream windows, rational readback, Real sealing, transport, replay, provenance, and local naming while refusing maximal spherical completeness and quotient completion.

Closure status for $\mathsf{NameCert}_{\mathsf{SphericalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SphericalCompletion}^{\uparrow}$ packet is generated from an ultrametric source row, a finite nested ball-chain row, a radius ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SphericalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8328.1, Definition 8328.2, and Definition 8328.3 seed the finite nested-ball-chain completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SphericalCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close maximal spherical completeness, Zorn-style nest extension, quotient completion equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.SphericalCompletionUp and checked carrier, nested-chain, radius-visibility, readback, seal, transport, replay, provenance, and local naming obligations.

Chapter 8329

A Concrete Naming Certificate for $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$

Following Bishop–Bridges finite-net arguments and the classical overlap multiplicity of finite covers, the $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$ packet records the overlap-depth ledger used by the compact-uniform-continuity route of chapter 9237. It sits beside the compact cover nerve of chapter 6263, the finite cover nerve of chapter 7597, the compact metric source of chapter 129, the rational lower-bound fold of chapter 4096, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names finite overlap depth and rational-fold rows; it does not assert open-cover compactness, selected subcovers, host set equality, countable choice, or an ambient Lebesgue-number theorem.

Definition 8329.1 (FiniteCoverMultiplicity carrier)**.** *A* $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, V, O, D, W, R, F, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover source row, including the displayed* $\mathsf{ProbeBundle}$ *of centers and radius histories.*
2. *V is the cover-member row, recording which displayed cover members contain each inspected source point.*
3. *O is the finite overlap-incidence row inherited from the cover-nerve route.*
4. *D is the multiplicity-depth row, recording the displayed finite bound on how many cover members can be simultaneously active at an inspected point or face.*
5. *W is the rational-fold weight row assigning each active member its* $\mathsf{Rat}^{\uparrow}$ *radius or tolerance contribution.*
6. *R is the rational lower-bound and rational-fold readback row consumed by compact-uniform modulus arguments.*
7. *F is the* $\mathsf{FiniteCoverNerve}^{\uparrow}/\mathsf{CompactCoverNerve}^{\uparrow}$ *sibling handoff row.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row reached only after the finite depth and rational-fold rows are displayed.*
9. *H records componentwise* hsame *transport for finite cover source, member, overlap, depth, weight, fold, handoff, and structural rows.*

10. *C records displayed* Cont *replay through cover membership, overlap incidence, multiplicity depth, rational folding, sibling handoff, and uniform-modulus consumption.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{CompactCoverNerve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverMultiplicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, O, D, W, R, F, U, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient equality of cover members, host point equality, countable choice, or ambient compactness.

Theorem 8329.2 (FiniteCoverMultiplicity NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$ *carrier*

$$M = (K, V, O, D, W, R, F, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverMultiplicity}^{\uparrow}}$ *surface consists exactly of carrier admission; compact finite-cover source visibility through K; cover-member exposure through V; overlap-incidence readback through O; multiplicity-depth exactness through D; rational-fold weight exposure through W, R; cover-nerve sibling compatibility through F; uniform-modulus handoff through U; componentwise transport and replay through H, C; and provenance and local naming through P, N.*

Proof. Carrier admission is projection from Definition 8329.1. The accepted packet exposes exactly the compact source, cover-member, overlap, depth, rational weight, fold, sibling handoff, uniform-modulus, transport, replay, provenance, and local naming rows.

The cover and overlap obligations are read through K, V, O. Each membership and incidence row is tied to the same displayed finite ProbeBundle, so the certificate cannot widen to an arbitrary open-cover statement.

The multiplicity obligation is the row D. It counts only simultaneously active displayed members or faces already visible through V, O, and therefore records overlap depth as finite packet data rather than as host set cardinality.

The rational and uniform obligations are the route W, R, F, U. The weights and fold rows expose the rational accounting, the sibling handoff ties that accounting to the finite cover nerves, and the uniform-modulus row is terminal.

Finally, H, C, P, N transport, replay, source, and name the same finite inventory. Since no excluded object is a coordinate of the carrier, the listed rows exhaust the local certificate. □

Theorem 8329.3 (FiniteCoverMultiplicity overlap-depth handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$ *packet reads overlap-depth and rational-fold data through the finite route*

$$K, V, O \longmapsto D, W, R \longmapsto F, U \longmapsto H, C, P, N.$$

The route supplies finite cover membership, finite overlap incidence, displayed multiplicity depth, rational-fold weights, cover-nerve compatibility, and a $\mathsf{UniformModulus}^{\uparrow}$ *handoff. It exports no selected subcover, no host nerve complex, no quotient equality of covers, no countable choice, and no ambient compactness theorem.*

Proof. Project the accepted carrier. The source-side route is K, V, O: compact metric cover rows expose the finite probe spine, cover membership records active members, and overlap incidence records the finite face data.

The accounting route is D, W, R. Multiplicity depth is computed only over the displayed active members, while the rational weight and fold rows record the finite radius accounting used by the uniform-modulus construction.

The consumer route is F, U, H, C, P, N. The cover-nerve handoff keeps the packet aligned with the sibling finite-nerve chapters, and the uniform-modulus row is consumed only after the depth and fold rows are visible.

Because the carrier has no coordinate for selected subcovers, host complexes, quotient cover equality, choice, or ambient compactness, every compact-uniform consumer remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverMultiplicity}\uparrow}$.

Constructive story: A $\mathsf{FiniteCoverMultiplicity}^{\uparrow}$ packet is generated from compact finite-cover rows, cover membership, overlap incidence, finite multiplicity depth, rational-fold weights, cover-nerve handoff, uniform-modulus consumption, transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverMultiplicity}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8329.1, Definition 8329.2, and Definition 8329.3 seed the overlap-depth and rational-fold route used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverMultiplicity}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host nerve complexes, quotient cover equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-member exactness, overlap-incidence exactness, multiplicity-depth exactness, rational-fold stability, cover-nerve compatibility, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverMultiplicityUp.

Chapter 8330

A Concrete Naming Certificate for $\mathsf{BishopRegSeqRealEta}^{\uparrow}$

Following Bishop–Bridges constructive completion, the $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ packet records the finite eta route by which a displayed regular rational sequence name is sealed as a $\mathsf{Real}^{\uparrow}$ name and then read back through the same finite windows. It sits over the regular-rational source of chapter 301, the terminal real-name seal of chapter 38, the finite window service of chapter 126, the dyadic tolerance ledger of chapter 291, and the completion handoff of chapter 423. The packet names the eta surface from a $\mathsf{RegSeqRat}^{\uparrow}$ display to a $\mathsf{Real}^{\uparrow}$ seal and back to the same displayed route; it does not identify quotient representatives, select a limit stream, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8330.1 (Bishop RegSeqRat-to-Real eta carrier)**.** *A* $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, W, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, recording the regular rational sequence name and its displayed Cauchy modulus data.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming the finite observations through which the source sequence is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, denominator budgets, and slack comparisons used by the windows.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after Q, W, D have been displayed.*
5. *A is the eta readback row, requiring any read from the sealed real name back toward regular rational data to expose the same Q, W, D, R route.*
6. *H records componentwise* hsame *transport for the source row, windows, dyadic ledger, Real seal, eta readback, provenance, and local name.*
7. *C records displayed* Cont *replay from a finite precision request through Q, W, D, R, A, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegSeqRealEta}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegSeqRealEta}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, W, D, R, A, H, C, P, N$. It has no coordinate for quotient equality of real names, selected representatives, countable-choice schedules, hidden completion objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8330.2 (Bishop RegSeqRat-to-Real eta NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ *carrier*

$$E = (Q, W, D, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegSeqRealEta}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, eta readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; seal ordering requiring R to be read only after Q, W, D; eta readback requiring A to expose the same Q, W, D, R route; and non-escape excluding quotient representatives, selected limits, countable choice, hidden completion objects, and ambient Real completeness.*

Proof. Project the carrier of Definition 8330.1. The accepted packet displays $Q, W, D, R, A, H, C, P, N$, so carrier admission and local naming are finite row projections.

Read the seal ordering next. The $\mathsf{RegSeqRat}^{\uparrow}$ source row Q supplies the regular rational name, W supplies the finite $\mathsf{StreamName}^{\uparrow}$ observations, and D supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before the terminal $\mathsf{Real}^{\uparrow}$ seal R can be consumed.

Read the eta row A. It is not a second representative of the real name; it is the displayed return route that exposes the same source, window, tolerance, and seal rows when a consumer reads back from R toward regular rational data.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite route. Since no carrier coordinate stores quotient equality, a selected limit, a choice schedule, a hidden completion object, or Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8330.3 (Bishop RegSeqRat-to-Real eta readback nonescape)**.** *Every consumer that reads the eta row of an accepted* $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ *packet must carry the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{Real}^{\uparrow}$ *seal through H, C, P, N. A readback that omits any one of Q, W, D, R leaves the accepted carrier and cannot cite the packet as a RegSeqRat-to-Real eta certificate.*

Proof. Start with the NameCert obligations of Definition 8330.2. They require the eta row A to expose the same Q, W, D, R route that produced the Real seal.

Transport and replay do not add another route. The row H preserves the displayed components under hsame, while C replays the finite precision request through those components. The provenance row P and local name N only record the same package boundary.

Therefore a consumer that omits Q, W, D, or R is no longer reading the accepted eta packet. It may describe a partial source, partial window, partial tolerance ledger, or sealed real name, but it is not the $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ readback route. □

Falsifiable prediction: An accepted $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ readback exposes a RegSeqRatUp source row, StreamNameUp finite windows, DyadicRatCoreUp tolerance ledger, RealUp seal, eta row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any consumer treats the sealed real name as returning to regular rational data.

Independence witness: regseqrat, real, streamname, dyadicratcore, cauchycompletionfunctor: none is bijective to $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ because the packet jointly carries the source regular

sequence, finite windows, dyadic tolerance ledger, Real seal, eta readback, transport, replay, provenance, and local NameCert rows while refusing quotient representatives, selected limits, countable choice, hidden completion objects, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegSeqRealEta}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegSeqRealEta}^{\uparrow}$ packet is generated from a RegSeqRatUp source row, StreamNameUp windows, a DyadicRatCoreUp tolerance ledger, a RealUp seal, an eta readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegSeqRealEta}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8330.1, Definition 8330.2, and Definition 8330.3 seed the finite RegSeqRat-to-Real eta route through source rows, finite windows, dyadic tolerance data, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegSeqRealEta}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, selected representatives, selected limits, countable choice, hidden completion objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seal ordering, eta readback exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegSeqRealEtaUp.

Chapter 8331

A Concrete Naming Certificate for $\mathsf{RealCompletionEtaLaw}^{\uparrow}$

Following Bishop–Bridges completion theory and Mac Lane's unit laws, the $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ packet records the Real-facing finite BEDC surface saying that the completion unit returns the same displayed real completion endpoint after the StreamName, RegSeqRat, dyadic tolerance, and Real-seal routes have been read. It refines the metric eta packet of chapter 8324 along the Real/RegSeqRat/StreamName route displayed by chapter 38, chapter 301, and chapter 126. The packet names the unit law as finite NameCert data; it is not a quotient completion equality, a selected representative theorem, countable choice, or an ambient completeness principle.

Definition 8331.1 (Real completion eta-law carrier)**.** *A* $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, U, W, R, D, Q, Z, H, C, P, N)$$

with the following displayed rows:

1. *T is the Real completion endpoint row already accepted by the displayed completion route.*
2. *U is the completion unit row applied to that same displayed endpoint.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the observable real-name windows for the direct and unit-applied routes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the finite precision ledger used by the readback.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, U, W, R, D have been displayed.*
7. *Z is the eta-law exactness row binding the direct Real completion endpoint and the unit-applied endpoint to the same displayed Real seal.*
8. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealCompletionEtaLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, U, W, R, D, Q, Z, H, C, P, N$. It has no coordinate for quotient Cauchy classes, selected representatives, host equality of real numbers, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8331.2 (Real completion eta-law NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ *carrier $E = (T, U, W, R, D, Q, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionEtaLaw}^{\uparrow}}$ *surface consists*

exactly of Real completion endpoint exposure, completion unit exposure, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal exposure, eta-law exactness, transport, replay, provenance, and local naming. Every consumer of the eta law must read those rows before it treats the direct Real completion endpoint and the unit-applied endpoint as the same named route.

Proof. Project the carrier of Definition 8331.1. The rows T and U name the displayed Real completion endpoint and the completion unit applied to that endpoint, while W, R, D expose the finite StreamName, RegSeqRat, and dyadic tolerance route through which a Real-facing consumer may inspect them.

Next read the terminal seal row Q. The seal is reached only after the endpoint, unit, window, readback, and tolerance rows have been displayed, so no quotient Cauchy class, selected representative, countable-choice schedule, or host real equality is available as an independent coordinate.

Finally use the eta-law exactness row Z and the structural rows H, C, P, N. They bind the direct and unit-applied routes to the same displayed Real seal under transport, replay, provenance, and local naming, which exhausts the NameCert surface. □

Theorem 8331.3 (Real completion eta-law unit handoff)**.** *Let $E = (T, U, W, R, D, Q, Z, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ *carrier. Any Real-facing completion consumer that enters through the unit row U must pass through the displayed endpoint T, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the eta-law exactness row Z before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q. The handoff therefore exposes the completion unit law as a finite Real-name route, not as quotient completion equality, selected representatives, host equality of real numbers, or an ambient completeness theorem.*

Proof. Begin with the unit row U. By the carrier definition it is attached to the already displayed Real completion endpoint T, so it cannot introduce a second endpoint outside the packet.

Read the finite observation route through W, R, D. These rows are exactly the StreamName windows, RegSeqRat readback, and dyadic tolerance ledger used by the Real/RegSeqRat/StreamName route, and they must be consumed before the seal Q is exposed.

The exactness row Z supplies the eta-law handoff, while H, C, P, N preserve the same displayed route. Thus a consumer receives the unit law for the existing Real completion endpoint without acquiring quotient equality, selected representatives, countable choice, host equality, or ambient Real completeness. □

Independence witness: metric-completion-eta-law, real, regseqrat, streamname, dyadicratcore: no listed sibling is bijective to $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ because this packet jointly binds a Real completion endpoint, completion unit row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, eta-law exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionEtaLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionEtaLaw}^{\uparrow}$ packet is generated from a Real completion endpoint row, a completion unit row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, eta-law exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionEtaLaw}^{\uparrow}}$, seed)

Scope closed: Definition 8331.1, Definition 8331.2, and Definition 8331.3 seed the finite Real-facing completion-unit eta-law surface over the Real/RegSeqRat/StreamName route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionEtaLaw}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected Cauchy representatives, host equality of real numbers, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-row exposure, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal exposure, eta-law exactness, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionEtaLawUp.

Chapter 8332

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$

Following Bishop and Bridges on constructive completions and the standard dense-embedding comparison used in metric completion, the $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$ packet records the finite BEDC surface for comparing two dense presentations of the same Cauchy completion boundary. It sits over the dyadic arithmetic core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Cauchy dense embedding of chapter 5893, and the metric dense embedding of chapter 3351. The packet compares displayed dense rows only; it does not quotient completion points, select representatives, identify host limits, invoke countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8332.1 (Cauchy completion dense-embedding comparison carrier)**.** *A* CauchyCompletionDenseEmbeddi *carrier is a finite* BHist *packet*

$$K = (S, D, M, E, E', Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the shared source row, binding the compared dense presentations to the same displayed Cauchy-completion source and its finite* $\mathsf{StreamName}^{\uparrow}$ *observation schedule.*
2. *D is the* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *row, exposing the Cauchy dense-source image, completion route, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal used by the first presentation.*
3. *M is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row, exposing the metric dense-source image, separated comparison route, uniform-completion handoff, and regular rational readback used by the second presentation.*
4. *E is the first dense embedding into the completion presentation, read through D before any comparison row is consumed.*
5. *E' is the second dense embedding into the completion presentation, read through M before any comparison row is consumed.*
6. *Q is the comparison row, recording finite agreement of the two dense-image reads on the shared source windows and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers.*

7. *H records componentwise* hsame *transport for S, D, M, E, E', Q and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing comparison request through the source, both dense embeddings, the comparison row, and the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$, $\mathsf{CauchyCompletionDenseEmbeddin}$ $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $S, D, M, E, E', Q, H, C, P, N$. It has no coordinate for quotient completion equality, a selected dense representative, host equality of limits, countable choice, or an ambient completion theorem.

Theorem 8332.2 (Cauchy completion dense-embedding comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$ *carrier*

$$K = (S, D, M, E, E', Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the shared source row, the Cauchy dense-embedding row, the metric dense-embedding row, the two completion-presentation embeddings, the comparison row, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing comparison read must expose S, D, M, E, E' before it can consume Q, H, C, P, N.*

Proof. Project the carrier of Definition 8332.1. The accepted packet displays exactly the source row, two dense-embedding rows, two completion-presentation embedding rows, one comparison row, and the structural rows.

The entry route is S, D, M, E, E'. The source row fixes the common finite windows; D and M expose the Cauchy and metric dense-embedding presentations; E and E' are therefore read only after their corresponding dense rows have been displayed.

The comparison route is Q, H, C, P, N. The row Q compares only the displayed dense-image reads over the shared windows and dyadic tolerance ledgers. The rows H and C transport and replay the same comparison, while P and N record provenance and the local certificate.

Since the carrier has no coordinate for quotient completion equality, selected representatives, host limit equality, countable choice, or ambient completeness, those endpoints cannot be exported by the local NameCert surface. □

Theorem 8332.3 (Cauchy completion dense-embedding comparison handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer of an accepted* $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyCompletionDenseEmbedding}^{\uparrow},\ \mathsf{MetricCompletionDens}$$

Thus the consumer receives a finite comparison between two dense presentations feeding the same completion boundary. It receives no quotient equality of completion points, selected representative, host limit equality, countable-choice schedule, total completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the NameCert obligations of Definition 8332.2. They force every comparison read to expose the shared source row and both dense-embedding rows before the comparison row Q is consumed.

Read the numerical side through the sibling rows. The source windows are governed by $\mathsf{StreamName}^{\uparrow}$, their tolerance ledgers by $\mathsf{DyadicRatCore}^{\uparrow}$, and their regular readback by $\mathsf{RegSeqRat}^{\uparrow}$ before either dense presentation reaches a $\mathsf{Real}^{\uparrow}$ seal.

The Cauchy presentation is supplied by $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$, while the metric presentation is supplied by $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$. The comparison row Q only relates those displayed dense-image reads over the shared windows.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Because none of the excluded host-level objects is a carrier coordinate, the handoff remains a comparison surface rather than a quotient completion theorem. □

Falsifiable prediction: Any accepted $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$ consumer that compares dense completion presentations while omitting either the $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ row, the $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ row, or the shared $\mathsf{StreamName}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ readback route is outside the carrier surface.

Independence witness: Nearest siblings: cauchy-completion-dense-embedding, metric-completion-dense-embedding, dyadicratcore, streamname, regseqrat, real. The carrier is not bijective to either dense-embedding sibling because it contains both presentations and a comparison row; not to DyadicRatCoreUp because it includes source windows, dense completion rows, regular readback, and Real seals; not to StreamNameUp or RegSeqRatUp because it adds dense-presentation comparison and local NameCert rows; not to RealUp because it exports no ambient completeness or quotient equality.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}$ packet is generated from a shared source row, Cauchy and metric dense-embedding rows, two completion-presentation embedding rows, a finite comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8332.1, Definition 8332.2, and Definition 8332.3 seed the finite comparison surface for two dense presentations feeding the same completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbeddingComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of completion points, selected dense representatives, host limit equality, countable choice, total completion objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires a checked BEDC.Derived.CauchyCompletionDenseEmbeddingComparisonUp carrier with source-row admission, two dense-embedding rows, comparison-row exactness, StreamName, DyadicRatCore and RegSeqRat readback, Real-seal confinement, transport, replay, provenance, and local naming.

Chapter 8333

A Concrete Naming Certificate for $\mathsf{DyadicRealBall}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicRealBall}^{\uparrow}$ packet records a finite dyadic centre-radius neighbourhood for regular real-name work. It reads dyadic arithmetic from chapter 291, finite observation windows from chapter 126, regular rational readback from chapter 301, and the terminal real-name seal from chapter 38. The packet names a concrete ball row available to later metric, uniform, and completion arguments; it does not identify host real equality, quotient representatives, selected limits, countable choice, or ambient completeness.

Definition 8333.1 (Dyadic real ball carrier)**.** *A* $\mathsf{DyadicRealBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row carrying the dyadic centre, positive dyadic radius, denominator budget, and slack comparisons used by the ball.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the centre name is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *M is the membership-witness row: an admitted point can be read as lying in the ball only through the displayed centre, radius, window, readback, and seal rows.*
6. *H records componentwise* hsame *transport for the dyadic data, finite windows, regular readback, Real seal, membership witness, provenance, and local name.*
7. *C records displayed* Cont *replay from a finite precision request through D, S, R, E, M, H.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRealBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRealBall}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, E, M, H, C, P, N$. It has no coordinate for host real equality, quotient equality of real names, selected representatives, countable choice, hidden metric completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8333.2 (Dyadic real ball NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRealBall}^{\uparrow}$ *carrier*

$$B = (D, S, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRealBall}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic centre-radius row, finite StreamName windows, RegSeqRat readback, terminal Real seal, membership-witness row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every ball-membership consumer must pass through*

$$D, S, R, E, M, H, C, P, N$$

and through no host real equality, quotient representative, selected limit, countable-choice selector, hidden completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 8333.1. The row D displays the dyadic centre, radius, denominator budget, and slack comparisons, while S and R display the finite observation and regular rational readback route for the centre name.

Read the seal and membership rows next. The terminal row E is available only after D, S, R have been displayed, and the row M records membership only as a witness over the same centre-radius, window, readback, and seal route.

Finally inspect H, C, P, N. Transport, replay, provenance, and local naming preserve the same finite route and add no hidden equality or selector coordinate. Since the carrier contains no host real equality, quotient representative, selected limit, countable-choice selector, hidden completion object, or ambient completeness field, the listed rows exhaust the NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRealBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicRealBall}^{\uparrow}$ packet is generated from a DyadicRatCore centre-radius row, StreamName finite windows, RegSeqRat readback, a RealUp seal, a membership-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicRealBall}^{\uparrow}}$, seed)

Scope closed: Definition 8333.1 and Definition 8333.2 seed the finite dyadic centre-radius ball route over dyadic arithmetic, finite windows, regular rational readback, terminal Real sealing, membership witnessing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicRealBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close metric topology, uniform-space structure, completion theorems, host real equality, quotient representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, centre-radius visibility, finite window readback, Real seal ordering, membership-witness exactness, transport, replay, provenance, and local naming for BEDC.Derived.DyadicRealBallUp.

Chapter 8334

A Concrete Naming Certificate for $\mathsf{BorelCantelliLemma}^{\uparrow}$

Following Borel and Cantelli, the $\mathsf{BorelCantelliLemma}^{\uparrow}$ packet records the finite summable-event handoff used by first-lemma consumers before any repeated-hit conclusion is visible. It sits after the probability source of chapter 184, the event-value surface of chapter 92, the finite union-bound ledger of chapter 6696, the event-sequence route of chapter 2357, and the bounded-window packet of chapter 7718. The packet names only displayed event rows, finite tail windows, partial probability sums, and the local summable-budget ledger; it does not import countable choice, almost-sure quotient equality, selected null sets, or the full infinite theorem.

Definition 8334.1 (Borel–Cantelli lemma carrier)**.** *A* $\mathsf{BorelCantelliLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\Omega, M, E, W, U, F, S, B, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the normalized probability carrier.*
2. M *is the* $\mathsf{Measure}^{\uparrow}$ *event-value row through which each displayed event probability is read.*
3. E *is the finite event-sequence packet, listing the inspected event names and measurable-event witnesses.*
4. W *is the finite tail-window row, recording the bounded index range seen by the consumer.*
5. U *is the* $\mathsf{FiniteUnionBound}^{\uparrow}$ *ledger row for the union of events in the displayed window.*
6. F *is the* $\mathsf{FiniteBorelCantelli}^{\uparrow}$ *bounded-window handoff row.*
7. S *is the local summable-event budget row, recording the finite partial-sum bound available to the window.*
8. B *is the Borel–Cantelli first-lemma route row read through* $\mathsf{BorelCantelli}^{\uparrow}$*.*
9. H *records componentwise* hsame *transport for* $\Omega, M, E, W, U, F, S, B, C, P, N$*.*
10. C *records displayed* Cont *replay from the probability and measure rows through the event window, finite union-bound ledger, bounded-window handoff, summable-budget row, and first-lemma route.*
11. P *is the* Pkg *provenance over* $\mathsf{BorelCantelliLemma}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{FiniteUnionBound}^{\uparrow}$*,* $\mathsf{BorelCantelli}^{\uparrow}$*,* $\mathsf{FiniteBorelCantelli}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{BorelCantelliLemma}^{\uparrow}}$ *row.*
The carrier has no coordinate for countable choice, almost-sure quotient equality, selected null-event witnesses, an independence converse, or ambient measure completion.

Theorem 8334.2 (Borel–Cantelli lemma NameCert surface)**.** *For every accepted* $\mathsf{BorelCantelliLemma}^{\uparrow}$ *carrier* $L = (\Omega, M, E, W, U, F, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorelCantelliLemma}^{\uparrow}}$ *surface consists exactly of probability-source admission through* Ω*, event-value exposure through* M*, finite event-sequence admission through* E*, finite tail-window exposure through* W*, finite union-bound ledger exposure through* U*, bounded-window handoff through* F*, summable-budget exposure through* S*, first-lemma route exposure through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8334.1. The projection exposes exactly the probability source, measure event-value row, finite event-sequence packet, tail-window row, finite union-bound ledger, bounded-window handoff, summable-budget row, first-lemma route, transport, replay, provenance, and local naming.

The continuation row fixes the order of inspection. A consumer first reads the probability and measure rows, then the finite event packet and tail window, then the finite union-bound ledger and bounded-window Borel–Cantelli handoff, and only then the local summable budget and first-lemma route.

Since the carrier contains no further coordinate, no countable-choice selector, almost-sure quotient equality, selected null-event witness, independence converse, or completed-measure row can be exported from this surface. □

Theorem 8334.3 (Borel–Cantelli lemma finite-sum handoff)**.** *Every accepted* $\mathsf{BorelCantelliLemma}^{\uparrow}$ *consumer that reads a summable-event request must pass through the finite route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \mathsf{FiniteUnionBound}^{\uparrow},\ \mathsf{FiniteBorelCantelli}^{\uparrow},\ \mathsf{BorelCantelli}^{\uparrow}.$$

The event-sequence row E*, tail-window row* W*, finite union-bound ledger* U*, bounded-window handoff* F*, and summable-budget row* S *are visible before the first-lemma route* B *is public.*

Proof. Start with the probability and measure rows Ω and M. They supply the source normalization and event-value reads for the displayed event sequence E.

Next restrict the request to the finite tail window W. The finite union-bound ledger U applies only to that displayed window, so the probability estimate is attached to a visible finite event family.

Read the bounded-window handoff F through Definition 7718.3. This places the finite Borel–Cantelli estimate downstream of $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, and $\mathsf{FiniteUnionBound}^{\uparrow}$.

Finally the local summable-budget row S and the first-lemma route row B consume that finite estimate. Transport through H, replay through C, provenance through P, and local naming through N preserve the same route, so the handoff remains finite BEDC data rather than a full infinite theorem. □

Theorem 8334.4 (Borel–Cantelli lemma no-choice escape)**.** *An accepted* $\mathsf{BorelCantelliLemma}^{\uparrow}$ *packet cannot expose countable choice, almost-sure quotient equality, a selected null-event witness, an independence-side converse, or ambient measure completion. Every public read of the packet is exhausted by the finite rows* $\Omega, M, E, W, U, F, S, B, H, C, P, N$ *of Definition 8334.1.*

Proof. Apply Definition 8334.2. The NameCert surface is exactly the listed probability, measure, event, window, union-bound, bounded-window, summable-budget, first-lemma, transport, replay, provenance, and naming rows.

Apply Definition 8334.3. The only public route to the first-lemma row passes through the finite union-bound and bounded-window handoff before reading the summable budget.

The refused objects are absent from the carrier and from the handoff route. Therefore no consumer can recover them as hidden projections of the $\mathsf{BorelCantelliLemma}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BorelCantelliLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{BorelCantelliLemma}^{\uparrow}$ packet is generated from a ProbSpace source row, Measure event-value row, finite event-sequence row, finite tail-window row, FiniteUnionBound ledger, FiniteBorelCantelli bounded-window handoff, local summable-event budget, BorelCantelli first-lemma route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelCantelliLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8334.1, Definition 8334.2, Definition 8334.3, and Definition 8334.4 seed the finite summable-event handoff for first-lemma consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelCantelliLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, almost-sure quotient equality, selected null-event witnesses, the independence converse, ambient measure completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite event-row exactness, finite tail-window routing, finite union-bound ledger exactness, bounded-window handoff, summable-budget exposure, first-lemma route exposure, transport, replay, provenance, and local naming for BEDC.Derived.BorelCantelliLemmaUp.

Chapter 8335

A Concrete Naming Certificate for $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$

Following Bishop and Bridges on proof-relevant constructive data and Troelstra–van Dalen on realizability-style finite evidence, the $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ packet records the finite boundary at which proof-token identity remains available as evidence while the public classifier ignores it. It sits beside the finite observation service of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name surface of chapter 38, and the shared-tail normalizer of chapter 1299. The packet is a finite sieve for Real and Completion-facing reads: it preserves the proof-token ledger as local evidence, but no displayed stream, regular-rational, dyadic, or Real row may classify two packets differently merely because the retained proof token has a different identity.

Definition 8335.1 (FiniteWindowProofIrrelevanceSieve carrier)**.** *A* $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the observable prefixes used by the consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the finite reads.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or Completion-facing seal row, available only after W, R, D have been displayed.*
5. *Q is the proof-token ledger retained inside the packet as evidence for how the finite windows were justified; Q is not a coordinate of the public classifier.*
6. *H records componentwise* hsame *transport for the window, readback, dyadic, seal, proof-token, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite read request through W, R, D, E, Q, H.*
8. *P is the* Pkg *provenance row over* $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}}$ *row.*

The public classifier compares accepted packets only through W, R, D, E, H, C, P, N. The row Q may be inspected by the local proof ledger, but it cannot create a public distinction after the same finite windows, regular readback, dyadic tolerance data, and seal row have been displayed.

Theorem 8335.2 (FiniteWindowProofIrrelevanceSieve NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ *carrier*

$$S = (W, R, D, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite window, regular readback, dyadic tolerance, Real or Completion seal, proof-token ledger, transport, replay, provenance, and local naming rows; public-readback preservation requiring every public consumer to expose the same W, R, D, E rows before classification; proof-token erasure requiring the classifier to ignore identity changes in Q once W, R, D, E, H, C, P, N are fixed; ledger nonescape requiring Q to remain a retained evidence row rather than a public classifier coordinate; componentwise* hsame *transport for all displayed rows; displayed* Cont *replay through the same finite route;* Pkg *provenance over the listed sibling carriers; and the local naming obligation for N.*

Proof. Carrier admission is projection from Definition 8335.1. The packet displays the finite observation row W, the regular rational readback R, the dyadic tolerance ledger D, the terminal seal row E, and the retained proof-token ledger Q before transport, replay, provenance, and naming are inspected.

Public-readback preservation is read componentwise. A public consumer sees W, R, D, E, H, C, P, N in the same order as the carrier, so two accepted packets with the same visible rows offer the same StreamName, RegSeqRat, DyadicRatCore, and Real-facing data.

Proof-token erasure and ledger nonescape are the same finite sieve boundary. The row Q remains present as evidence for local reconstruction, but it is not an argument of the public classifier. Replacing Q while preserving the visible rows therefore cannot change the public readback class.

Transport and replay are handled row by row. The hsame row H transports each displayed component without turning Q into a public coordinate, and the Cont row C replays finite read requests through the same window, readback, dyadic, seal, and retained-ledger route.

Finally, P and N source and name the same finite packet. Since the carrier has no extra public coordinate beyond W, R, D, E, H, C, P, N, the listed clauses exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ public read exposes the StreamNameUp windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp or Completion-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before classification; a read whose only difference is proof-token identity in Q has the same public classifier value.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchytailthresholdnormalizer: none is bijective to $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ because the packet jointly carries public finite reads and a retained proof-token ledger while refusing to let proof-token identity escape into the public classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp or Completion-facing seal, a retained proof-token ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowProofIrrelevanceSieve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8335.1 and Definition 8335.2 seed the finite public-readback route that retains proof-token evidence while erasing proof-token identity from the classifier.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWindowProofIrrelevanceSieve}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotienting of displayed streams, proof irrelevance for all BEDC objects, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, public-readback preservation, proof-token erasure, ledger nonescape, hsame transport, Cont replay, Pkg provenance, and local naming for BEDC.Derived.FiniteWindowProofIrrelevanceSieve

Chapter 8336

A Concrete Naming Certificate for CompleteMetricContractionPrinciple$^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the CompleteMetricContractionPrinciple$^{\uparrow}$ packet records the fixed-point route in a complete metric space as finite NameCert data. It sits after the metric carrier of chapter 57, the complete-metric service of chapter 128, the Cauchy criterion surface of chapter 392, the Picard contraction route of chapter 395, and the fixed-point certificate of chapter 396. The packet names the contraction principle only when the metric source, contraction modulus, Picard orbit, Cauchy handoff, fixed-point seal, transport, replay, provenance, and local NameCert rows are all displayed; it is not a host selected-limit theorem.

Definition 8336.1 (Complete metric contraction principle carrier)**.** *A* CompleteMetricContractionPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, d, \lambda, O, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the* CompleteMetric$^{\uparrow}$ *source row naming the complete metric service consumed by the route.*
2. *d is the* Met$^{\uparrow}$ *distance row used for finite iterate comparisons and contraction estimates.*
3. *λ is the contraction-modulus row, including the displayed positivity and unit-bound data.*
4. *O is the* PicardContraction$^{\uparrow}$ *orbit row, exposing the initial point, successor applications, and finite Picard ledger.*
5. *Q is the* CauchyCriterion$^{\uparrow}$ *route that turns the contraction estimates along O into named Cauchy data.*
6. *F is the* BanachFixedPoint$^{\uparrow}$ *fixed-point seal, available only after the Picard and Cauchy rows have been displayed.*
7. *H records componentwise* hsame *transport for the complete metric source, distance, modulus, Picard orbit, Cauchy route, fixed-point seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric reads through contraction estimates, Picard iteration, Cauchy criterion, and fixed-point sealing.*
9. *P is the* Pkg *provenance over* CompleteMetricContractionPrinciple$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Met$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, PicardContraction$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected limit, quotient completion, countable choice, ambient real completeness, host fixed-point equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompleteMetricContractionPrincipleUp.CompleteMetricContractionPrincipleTasteGate single carrier alignment`

Theorem 8336.2 (Complete metric contraction principle NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}$ *carrier*

$$K = (M, d, \lambda, O, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}}$ *surface consists exactly of complete-metric source exposure, metric-distance exposure, contraction-modulus visibility, Picard-orbit visibility, Cauchy-Criterion route exposure, fixed-point seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8336.1. The source side is M, d, λ, O: the complete metric row supplies the service boundary, the metric row supplies distance comparisons, the modulus row supplies the strict contraction ledger, and the Picard row supplies the finite orbit.

The Cauchy row Q is downstream of those displayed data. It records the CauchyCriterion handoff for the Picard orbit, so the fixed-point seal F cannot be read before the orbit and Cauchy route are visible.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no selected limit, quotient completion, countable choice, ambient real completeness, host fixed-point equality, Lean verification, or bridge checking, the listed rows exhaust the local certificate surface. □

Theorem 8336.3 (Complete metric contraction Picard-Cauchy route)**.** *Every fixed-point read exported by an accepted* $\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{PicardContraction}^{\uparrow},\ \mathsf{CauchyCriterion}^{\uparrow},\ \mathsf{BanachFixedPoint}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

The route first exposes the complete metric source and distance row, then the contraction modulus and Picard orbit, then the CauchyCriterion handoff, and only then the BanachFixedPoint seal. It does not supply a host selected limit, a quotient completion, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or fixed-point equality outside the displayed packet.*

Proof. Begin with the accepted carrier. Its source rows M, d, λ, O are the only complete-metric, distance, modulus, and Picard coordinates available to a consumer.

Use the obligation theorem Definition 8336.2. It requires the CauchyCriterion row Q to be read from those displayed source and orbit rows before a fixed-point seal is available.

Finally read F, H, C, P, N. The fixed-point seal is transported by H, replayed by C, sourced by P, and named by N along the same finite route. Because no excluded coordinate occurs in the carrier, the exported principle is a NameCert handoff rather than a host selected-limit argument. □

Falsifiable prediction: Every accepted $\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}$ read exposes CompleteMetricUp, MetricUp, contraction modulus, PicardContractionUp, CauchyCriterionUp, BanachFixedPointUp, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any fixed-point consumer can use the seal.

Independence witness: complete metric, metric, cauchy criterion, picard contraction, banach fixed point: no listed sibling carries the whole packet $K = (M, d, \lambda, O, Q, F, H, C, P, N)$ because each sibling supplies only a source, distance, orbit, Cauchy route, or fixed-point seal while this chapter binds all rows as one complete-metric contraction-principle certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}$ packet is generated from a CompleteMetric source row, a Metric distance row, a contraction-modulus row, a PicardContraction orbit row, a CauchyCriterion route, a BanachFixedPoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8336.1, Definition 8336.2, and Definition 8336.3 seed the finite contraction-principle route over complete metric data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricContractionPrinciple}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close a host selected-limit theorem, quotient completion, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, host fixed-point equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked complete-metric source exposure, metric-distance exposure, contraction-modulus visibility, Picard-orbit visibility, CauchyCriterion handoff, fixed-point seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompleteMetricContractionPrincipleUp.

Chapter 8337

A Concrete Naming Certificate for $\mathsf{LocatedRealOrder}^{\uparrow}$

Following Bishop and Bridges on located order for constructive real numbers, the $\mathsf{LocatedRealOrder}^{\uparrow}$ packet records the finite BEDC surface by which order-facing real comparisons are read from displayed regular Cauchy windows. It sits over the terminal real-name seal of chapter 38, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the comparison row of chapter 1412. The packet exposes located interval and apartness evidence as finite L10 data; it does not import trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 8337.1 (Located real order carrier)**.** *A* $\mathsf{LocatedRealOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, W, S, R, Q, I, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and tolerance ledger used by all order-facing comparisons.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that selects the displayed rational approximants.*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the same windows and dyadic budget.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, W, S have been displayed.*
5. *Q is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row used to keep paired real names on the same finite comparison route.*
6. *I is the located interval row, recording a finite dyadic bracket or finite separation of brackets for the requested order comparison.*
7. *A is the apartness row, recording displayed positive separation when a strict order read is exported.*
8. *H records componentwise* hsame *transport for D, W, S, R, Q, I, A and for the structural rows.*
9. *C records displayed* Cont *replay from an order-facing request through the dyadic, window, readback, seal, comparison, interval, apartness, transport, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. N is the local $\mathsf{NameCert}_{\mathsf{LocatedRealOrder}^{\uparrow}}$ *row.*
The local classifier reads accepted packets only through $D, W, S, R, Q, I, A, H, C, P, N$. It has no coordinate for real trichotomy, quotient real equality, selected representatives, countable choice, ambient real completeness, or an external bridge object.

Theorem 8337.2 (Located real order NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealOrder}^{\uparrow}$ *carrier*

$$O = (D, W, S, R, Q, I, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, Cauchy-sequence comparison row, located interval evidence, apartness evidence, componentwise transport, displayed replay, provenance, and local naming. Every order-facing consumer must expose D, W, S, R, Q before it can consume I, A, H, C, P, N.*

Proof. Project the carrier of Definition 8337.1. The admitted packet contains exactly the numerical rows D, W, S, R, Q, I, A and the structural rows H, C, P, N.

The entry route is D, W, S, R, Q. The dyadic ledger fixes the tolerance budget, the stream schedule fixes the finite windows, the regular rational row supplies the readback, the real row supplies the terminal seal, and the Cauchy-sequence comparison row keeps paired names on one finite route.

The order route is I, A, H, C, P, N. The located interval row and apartness row are consumed only after the real and comparison rows are visible; the transport and replay rows preserve the same displayed route; the provenance and naming rows record the local certificate.

Since the carrier has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, ambient real completeness, or bridge checking, those endpoints cannot be exported by the local $\mathsf{NameCert}$ surface. □

Theorem 8337.3 (Located real order apartness handoff)**.** *Let*

$$O = (D, W, S, R, Q, I, A, H, C, P, N)$$

be an accepted $\mathsf{LocatedRealOrder}^{\uparrow}$ *carrier. Any strict order or apartness read exported from O must pass through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback S, terminal* $\mathsf{Real}^{\uparrow}$ *seal R, and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row Q before the apartness row A is consumed. The read exports the finite packet $D, W, S, R, Q, I, A, H, C, P, N$ and no trichotomy decision, quotient equality, selected representative, countable-choice schedule, ambient completeness theorem, or bridge certificate.*

Proof. Start from the NameCert obligations of Definition 8337.2. They force every order-facing consumer to expose the dyadic, window, readback, real-seal, and comparison rows before any interval or apartness evidence is read.

The strict order witness is the row A. It is not an independent sign oracle: it is attached to the located interval row I and therefore to the same displayed dyadic budget and finite windows used by D, W, S, R, Q.

The rows H, C, P, N transport, replay, source, and name that same packet. Because none of the excluded host-level objects is a carrier coordinate, the apartness handoff remains a finite L10 route rather than a trichotomy or quotient-equality principle. □

Theorem 8337.4 (Located real order interval route)**.** *Every interval comparison consumed through an accepted* $\mathsf{LocatedRealOrder}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchySequenceEquivalence}^{\uparrow} \longmapsto I, A.$$

Thus the consumer receives a located interval or displayed apartness read over one finite comparison route. It receives no quotient real equality, selected representative, trichotomy split, countable-choice stream, ambient $\mathsf{Real}^{\uparrow}$ *completeness object, or checked bridge.*

Proof. Read the numerical side through Definition 8337.1. The only tolerance source is D, the only finite schedule is W, and the only regular rational readback is S.

The terminal real row R is reached only after that finite route is displayed. The comparison row Q then aligns paired Cauchy presentations through the same finite windows before an interval read is accepted.

The rows I and A supply the interval and apartness evidence. They are therefore downstream of D, W, S, R, Q, while H, C, P, N preserve the route as one packet.

Consequently the interval route is exactly the displayed finite chain. The excluded objects would require additional carrier coordinates, so they cannot be read from an accepted $\mathsf{LocatedRealOrder}^{\uparrow}$ packet. □

Falsifiable prediction: Any accepted $\mathsf{LocatedRealOrder}^{\uparrow}$ consumer that exports an order-facing real comparison while omitting the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, or $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ comparison row is outside the carrier surface.

Independence witness: Nearest siblings: real, streamname, regseqrat, dyadicratcore, cauchy-sequenceequivalence, and cauchy-real-apartness. The carrier is not bijective to RealUp because it adds located interval and apartness rows; not to StreamNameUp or RegSeqRatUp because it adds real-seal comparison and order evidence; not to DyadicRatCoreUp because it adds windows, readback, comparison, and local NameCert rows; not to CauchySequenceEquivalenceUp because it exports interval and apartness evidence; not to CauchyRealApartnessUp because it keeps the located order boundary as its own finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealOrder}^{\uparrow}$ packet is generated from a DyadicRatCore observation ledger, StreamName finite windows, RegSeqRat readback, a terminal Real seal, a CauchySequenceEquivalence comparison row, located interval and apartness evidence, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealOrder}^{\uparrow}}$, seed)

Scope closed: Definition 8337.1, Definition 8337.2, Definition 8337.3, and Definition 8337.4 seed the finite located-order surface for real names through dyadic, stream, regular-rational, real-seal, and comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, trichotomy, quotient real equality, selected representatives, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires a checked BEDC.Derived.LocatedRealOrderUp carrier with dyadic ledger admission, finite window readback, RegSeqRat and Real seal handoff, CauchySequenceEquivalence comparison, located interval and apartness exactness, transport, replay, provenance, and local naming.

Chapter 8338

A Concrete Naming Certificate for $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$

Following Bishop and Bridges on constructive compact metric arguments, the $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ packet records the finite radius-selector surface that turns compact-source finite-net data and pointwise continuity radii into a single uniform-radius row. It sits between the compact source chapters chapter 60 and chapter 129, the metric carrier of chapter 57, the pointwise continuity route of chapter 59, the rational lower-bound route of chapter 37, the terminal real seal of chapter 38, and the downstream uniform-modulus consumer of chapter 369. The packet is a finite NameCert carrier for Real and Completion uniform-modulus routes; it is not an open-cover compactness theorem, a sequential compactness principle, a countable-choice functional, host real equality, or an ambient Real completeness assertion.

Definition 8338.1 (Compact-source uniform-radius selector carrier)**.** *A* $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, X, M, F, T, L, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, read together with the* $\mathsf{Compact}^{\uparrow}$ *compact-source row X before any radius is selected.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *triangle and distance-readback row for the source and target metrics used by the estimate.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-radius table; each inspected compact-net center carries its displayed local continuity radius and target tolerance row.*
4. *T is the finite* $\mathsf{Met}^{\uparrow}$ *triangle row that compares an arbitrary source point with the chosen finite-net center and transports the target estimate along the metric triangle.*
5. *L is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, recording the finite minimum of the displayed positive pointwise radii.*
6. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius certificate produced by the lower-bound fold.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row used when Real-facing consumers read the selected uniform radius.*

8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that exports only the selected radius, the compact finite-net source rows, the pointwise-radius rows, the metric triangle rows, and the Real seal.*
9. *H records componentwise* hsame *transport for $K, X, M, F, T, L, Q, R, U$ and the structural rows.*
10. *C stores displayed* Cont *replay from a finite source-tolerance request through the compact source, pointwise-radius table, lower-bound fold, Real seal, and uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, X, M, F, T, L, Q, R, U, H, C, P, N$. It has no coordinate for open-cover compactness, sequential compactness, countable choice, host real equality, quotient metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8338.2 (Compact-source uniform-radius selector NameCert obligations)**.** *For every accepted* $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ *carrier*

$$E = (K, X, M, F, T, L, Q, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source exposure through K and X, metric triangle exposure through M and T, continuous-map pointwise-radius exposure through F, rational lower-bound folding through L and Q, Real sealing through R, UniformModulus handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8338.1. The compact source rows are K and X, the analytic estimate rows are M, F, T, L, Q, R, U, and the structural rows are H, C, P, N.

The compact source is read first. Its finite-net row supplies only displayed centers and source-radius windows, so the packet cannot appeal to an open-cover compactness theorem, a sequential-compactness principle, or countable choice.

Next the pointwise-radius table F and the metric triangle row T are replayed over those displayed centers. The rational rows L and Q fold the finite list of positive pointwise radii into one lower-bound certificate, and the $\mathsf{Real}^{\uparrow}$ row R seals the selected radius for Real-facing reads.

Finally the $\mathsf{UniformModulus}^{\uparrow}$ handoff row U exports exactly that finite route, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host real equality, quotient metric equality, or ambient Real completeness, the listed rows exhaust the local certificate surface. □

Theorem 8338.3 (Compact-source uniform-radius selector consumer boundary)**.** *Every Real or Completion uniform-modulus consumer that reads an accepted* $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ *packet must pass through the finite route*

$$K, X, M, F, T, L, Q, R, U, H, C, P, N$$

before using the selected uniform-radius row. The consumer receives a compact finite-net source, pointwise-radius rows, metric triangle rows, a rational lower-bound fold, a Real seal, and the $\mathsf{UniformModulus}^{\uparrow}$ *handoff; it receives no open-cover compactness object, sequential compactness object, countable-choice functional, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the NameCert obligations. They identify $K, X, M, F, T, L, Q, R, U, H, C, P, N$ as the whole public surface of the accepted selector packet.

The Real-facing radius is downstream of the rational lower-bound rows. A consumer therefore reads L and Q before the seal R, and it cannot replace the finite fold by host real equality or by a completeness argument.

The Completion-facing uniform-modulus route is downstream of U. That row carries only the selected radius and the displayed compact-source, continuity, triangle, rational, and Real-seal data. Hence the consumer boundary is exactly the finite route above. □

Falsifiable prediction: An accepted $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ consumer that obtains a uniform-radius row without reading $K, X, M, F, T, L, Q, R, U, H, C, P, N$ refutes the selector carrier.

Independence witness: compact, compactmetric, metric, continuous, rat, real, uniformmodulus: no listed sibling is bijective to $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ because this packet jointly binds compact-source finite-net rows, pointwise-radius rows, metric triangle rows, rational lower-bound folding, a Real seal, UniformModulus handoff, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}$ packet is generated from CompactMetric and Compact finite-net source rows, ContinuousMap pointwise-radius rows, MetricSpace triangle rows, Rat lower-bound fold rows, a Real seal, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8338.1, Definition 8338.2, and Definition 8338.3 seed the constructive compact-source radius selector consumed by Real and Completion uniform-modulus routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactSourceUniformRadiusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter excludes open-cover compactness, sequential compactness, countable choice, host real equality, ambient Real completeness, Lean theorem verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, pointwise-radius replay, metric triangle replay, rational lower-bound folding, Real sealing, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactSourceUniformRadiusSelectorUp.

Chapter 8339

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyTwoScale}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and explicit error estimates, the $\mathsf{BishopRegularCauchyTwoScale}^{\uparrow}$ packet records the finite two-scale estimate used when a coarse regular-Cauchy window is refined by a smaller inspection scale. It sits beside the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the finite-window induction route of chapter 8091. The packet names a displayed induction object for two-scale Cauchy estimates; it is not a selected limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8339.1 (Bishop regular Cauchy two-scale carrier)**.** *A* $\mathsf{BishopRegularCauchyTwoScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, k, W_0, W_1, Q_0, Q_1, D, I, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular-Cauchy source schedule whose estimates are inspected at two displayed scales.*
2. *K is the coarse scale row and k is the refined scale row, with the classifier requiring the refined request to be read below the coarse request through dyadic tolerance data.*
3. *W_0 and W_1 are the* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected at the two scales.*
4. *Q_0 and Q_1 are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger storing the denominator budgets, slack rows, and two-scale comparison inequalities.*
6. *I is the* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *row that records the finite induction step from the coarse window to the refined window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after $S, K, k, W_0, W_1, Q_0, Q_1, D, I$ have been displayed.*
8. *H stores componentwise* hsame *transport for the source schedule, both scale rows, both window rows, both readback rows, the dyadic ledger, the induction row, the real seal, provenance, and local naming.*
9. *C stores displayed* Cont *replay from a finite precision request through the same rows.*

10. *P is the* Pkg *provenance over* BishopRegularCauchyTwoScale$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyWindowInduction$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BishopRegularCauchyTwoScale}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a completed real selected outside the displayed windows, quotient equality of streams, host real equality, countable choice, or ambient completeness.

Theorem 8339.2 (Bishop regular Cauchy two-scale obligations)**.** *For every accepted* BishopRegularCauchyTwoScale *carrier*

$$T = (S, K, k, W_0, W_1, Q_0, Q_1, D, I, E, H, C, P, N),$$

the local NameCert$_{\text{BishopRegularCauchyTwoScale}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source schedule, the two scale rows, the two* StreamName$^{\uparrow}$ *windows, the two* RegSeqRat$^{\uparrow}$ *readback rows, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* RegularCauchyWindowInduction$^{\uparrow}$ *step row, the* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local name.*

Proof. Project the accepted carrier of Definition 8339.1. The rows $S, K, k, W_0, W_1, Q_0, Q_1, D, I, E, H, C, P, N$ are the complete finite packet, so carrier admission and local naming are componentwise projections.

Read the two scales next. The coarse row K and refined row k are compared only through the dyadic tolerance ledger D, and their inspected windows are exactly W_0, W_1 with readbacks Q_0, Q_1. Thus the estimate is a displayed finite relation between two windows, not a statement about an unshown completed stream.

Finally use the induction and structural rows. The row I supplies the finite-window step from chapter 8091, while H, C, P, N transport, replay, source, and name the same displayed data. Since E is reached only after those rows are present, the certificate cannot export a selected limit, quotient stream equality, countable choice, or ambient Real$^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted BishopRegularCauchyTwoScale$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ seal without reading $S, K, k, W_0, W_1, Q_0, Q_1, D, I, E, H, C, P, N$ refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-window-induction: no listed sibling supplies the joint packet containing one regular-Cauchy source schedule, two displayed scale rows, two finite window rows, two regular-rational readback rows, a dyadic two-scale tolerance ledger, a finite-window induction row, a Real seal, transport, replay, provenance, and local NameCert row.

Closure status for NameCert$_{\text{BishopRegularCauchyTwoScale}^{\uparrow}}$**.**

Constructive story: A BishopRegularCauchyTwoScale$^{\uparrow}$ packet is generated from a regular-Cauchy source schedule, coarse and refined scale rows, paired StreamName windows, paired RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RegularCauchyWindowInduction row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopRegularCauchyTwoScale}^{\uparrow}}$, seed)

Scope closed: Definition 8339.1 and Definition 8339.2 seed the finite two-scale Cauchy estimate route through displayed stream windows, regular-rational readback, dyadic tolerance rows, window induction, and the terminal Real-facing seal.

Formal verification: FormalStatus(NameCert$_{\text{BishopRegularCauchyTwoScale}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, Real completeness, quotient stream equality, selected limits, host real equality, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, two-scale dyadic comparison, paired window readback, finite-window induction replay, transport, provenance, and local naming for BEDC.Derived.BishopRegularCauchyTwoScaleUp.

Chapter 8340

A Concrete Naming Certificate for $\mathsf{DyadicMeshDiameter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicMeshDiameter}^{\uparrow}$ packet names the finite diameter ledger carried by a dyadic mesh before interval-net and compactness consumers read it. It sits after the dyadic rational carrier of chapter 291, the interval-net surface of chapter 5966, the interval diameter ledger of chapter 1945, the closed bounded interval net route of chapter 1250, and the real-name seal of chapter 38. The chapter isolates the mesh-scale and diameter rows that a finite net exposes before any consumer treats the mesh as a compactness witness.

Definition 8340.1 (Dyadic mesh diameter carrier)**.** *A* $\mathsf{DyadicMeshDiameter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, E, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the mesh-scale row, recording the finite dyadic subdivision level used by the net.*
2. *E is the endpoint row family, listing the dyadic endpoint data inspected by adjacent mesh cells.*
3. *D is the dyadic diameter ledger, recording the displayed diameter bound attached to the mesh.*
4. *R is the refinement row, relating a finer mesh scale to the same endpoint and diameter ledger surface.*
5. *H records componentwise* hsame *transport for mesh scale, endpoint rows, diameter ledger, refinement row, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay for interval-net, closed bounded interval net, and finite compactness consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshDiameter}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{DyadicIntervalDiameter}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshDiameter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, E, D, R, H, C, P, N. It has no coordinate for an open-cover compactness proof, a completed real interval value, a selected limiting point, quotient equality of real names, or an unlisted mesh.

Lean4 (definition) `BEDC.Derived.DyadicMeshDiameterUp`

Theorem 8340.2 (Dyadic mesh diameter NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshDiameter}^{\uparrow}$ *carrier*

$$W = (M, E, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMeshDiameter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed mesh scale, endpoint family, diameter ledger, refinement row, transport, replay, provenance, and local naming;*
2. *classifier transport by common refinement, requiring two mesh reads to compare only after both expose the shared refinement row* R *and the componentwise transport row* H*;*
3. *diameter bound exactness, requiring every interval-net diameter read to pass through the displayed dyadic ledger* D*;*
4. *refinement monotonicity, requiring a finer accepted mesh to preserve or decrease the displayed dyadic diameter bound;*
5. *ledger nonescape, requiring* P, N *to source and name no open-cover compactness proof, completed real interval value, selected limit, quotient real equality, or unlisted mesh.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8340.1. The accepted packet displays exactly one mesh-scale row, one endpoint-family row, one dyadic diameter ledger, one refinement row, and the structural rows that transport, replay, source, and name them.

Classifier transport is read through R and H. A comparison of two mesh reads must first move both packets to the displayed common refinement surface; only then can the componentwise transport row compare mesh scale, endpoint rows, and diameter ledger data.

Diameter bound exactness is read from D. The interval-net consumer therefore receives a dyadic ledger value rather than a host metric diameter or a real-valued supremum.

Refinement monotonicity is the finite ledger clause attached to R. Refining the mesh may preserve the previous displayed bound or replace it by a smaller dyadic bound, but the route contains no coordinate that can increase the bound while still naming the same accepted refinement.

Finally, P and N keep the obligation surface local to the displayed rows. Since the excluded objects are absent from the carrier, no compactness theorem, completed real value, selected limit, quotient equality, or hidden mesh can enter the local NameCert surface. □

Theorem 8340.3 (Dyadic mesh diameter refinement monotonicity)**.** *Let* $W = (M, E, D, R, H, C, P, N)$ *be an accepted* $\mathsf{DyadicMeshDiameter}^{\uparrow}$ *carrier. Any accepted refinement read through* R *and replayed through* C *exposes a diameter ledger that is the same as* D *or a dyadic bound below* D*. Thus mesh refinement can only decrease or preserve the displayed dyadic diameter bound before interval-net, closed bounded interval net, or compactness consumers read the packet.*

Proof. Project the carrier of Definition 8340.1. The only row that records a mesh diameter is D, and the only row that records a mesh refinement is R.

An accepted refinement must be replayed through C while preserving the same endpoint and provenance surface through H, P, N. Therefore the refined packet can expose only the original diameter ledger or the dyadic ledger named by the refinement row.

The NameCert obligations of Definition 8340.2 require the refinement row to preserve or lower the displayed dyadic bound. Since there is no coordinate for an external diameter operation or a hidden mesh, no larger bound can be introduced by a valid refinement read. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshDiameter}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshDiameter}^{\uparrow}$ packet is generated from a mesh-scale row, dyadic endpoint rows, a diameter ledger, a refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshDiameter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8340.1, Definition 8340.2, and Definition 8340.3 seed the finite dyadic mesh diameter route used by interval-net and compactness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshDiameter}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.DyadicMeshDiameterUp`

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, completed RealUp interval values, selected limits, quotient real equality, ambient completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport by common refinement, diameter bound exactness, refinement monotonicity, ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicMeshDiameterUp.

Chapter 8341

A Concrete Naming Certificate for Window-Six Weight Log-Concavity

The window-six Fibonacci carrier records finite binary words with no adjacent true entries and reads their weight distribution through the polynomial

$$W_m(x) = \sum_k b(m,k)x^k,$$

where $b(m,k)$ is the number of length-m admissible words with exactly k true entries. Equivalently $b(m,k) = \binom{m-k+1}{k}$ on its natural support, and $\sum_k b(m,k) = F_{m+2}$. The BEDC carrier used here is the direct finite enumeration of those words, not the closed binomial formula.

Definition 8341.1 (Fibonacci weight coefficient)**.** *For $m, k \in \mathbb{N}$, let $b(m,k)$ be the coefficient of x^k in $W_m(x)$, computed by enumerating all length-m Boolean words, rejecting words with adjacent true entries, and counting the survivors with weight k. In Lean this coefficient is the natural number*

$$\mathsf{weightCoeff}(m,k).$$

Theorem 8341.2 (Window-six through window-ten degree-two Turan certificate)**.** *For the finite windows $m = 6, 7, 8, 9, 10$, every internal coefficient position checked by the certificate satisfies the degree-two Turan inequality*

$$b(m,k+1)b(m,k-1) \le b(m,k)^2.$$

Equivalently, the inspected coefficient rows of $W_m(x)$ are log-concave. The Lean certificate covers the nontrivial internal positions

$$\begin{aligned}
&(6,1),(6,2),\\
&(7,1),(7,2),(7,3),\\
&(8,1),(8,2),(8,3),\\
&(9,1),(9,2),(9,3),(9,4),\\
&(10,1),(10,2),(10,3),(10,4).
\end{aligned}$$

Lean4 ✓ `BEDC.Derived.Window6WeightLogConcaveTuran.window six to ten weight logconcave turan`

Proof. The proof is a finite certificate. Lean computes the admissible-word counts by recursive enumeration and then proves each inequality by an explicit natural-number slack. For example, the window $m = 6$ row is

$$(b(6,0), b(6,1), b(6,2), b(6,3)) = (1, 6, 10, 4),$$

with total $21 = F_8$. The two internal inequalities are witnessed by

$$b(6,2)b(6,0) + 26 = b(6,1)^2,$$
$$b(6,3)b(6,1) + 76 = b(6,2)^2.$$

The remaining windows are discharged in the same finite manner, using only the enumerator and natural-number slack equalities. □

Theorem 8341.3 (Finite-window RH criterion shape)**.** *The certificate of Definition 8341.2 realizes the degree-two Turan positivity shape on a Fibonacci carrier. In the classical analytic route, the Riemann Hypothesis is equivalent to the completed xi function lying in the Laguerre–Polya class, and that route is tested by finite Jensen-polynomial hyperbolicity windows; at degree two the finite positivity condition has the Turan form*

$$a_k^2 \geq a_{k-1}a_{k+1}.$$

This chapter supplies that shape only for the displayed Fibonacci coefficients, adjacent to the Laguerre–Polya packet of chapter 2363 and the RH roadmap of chapter 9239.

Proof. The statement is a scope identification. The Lean theorem supplies finite natural-number inequalities for the coefficient rows of $W_m(x)$. The Laguerre–Polya and Jensen route asks for hyperbolicity of finite polynomial windows, whose degree-two shadow is the adjacent coefficient inequality above. The carrier here is not the xi-function coefficient stream; it is the finite Fibonacci-cube weight carrier.

This is a finite-window positivity certificate on a Fibonacci carrier, not a proof of RH; it models the degree-two Turán/Jensen shape that Laguerre-Pólya (hence RH) requires. □

Falsifiable prediction: A recomputation of the Lean finite enumerator for $m = 6, 7, 8, 9, 10$ that violates any displayed internal Turan inequality refutes this packet.

Independence witness: laguerre-polya, riemann-hypothesis-roadmap, window-six-fibonacci-carrier: this packet is not bijective to the Laguerre–Polya or RH packets because it contains only finite Fibonacci-cube word enumeration, finite weight coefficients, explicit slack equalities, and the degree-two Turan shape on those coefficients.

Closure status for NameCert$_{Window6WeightLogConcaveTuran}$.

Theory closure: TheoryGate(NameCert$_{Window6WeightLogConcaveTuran}$, scopedClosed)

Scope closed: Definition 8341.1, Definition 8341.2, and Definition 8341.3 close the finite Fibonacci-carrier degree-two Turan certificate for $m = 6, 7, 8, 9, 10$.

Formal verification: FormalStatus(NameCert$_{Window6WeightLogConcaveTuran}$, theoremChecked)

Lean target: `BEDC.Derived.Window6WeightLogConcaveTuran.window\_six\_to\_ten\_weight\_logconcave\_turan`

Bridge status: none

Not claimed: not a proof of RH; finite Fibonacci-carrier degree-2 Turan positivity only

Upgrade path: extend to higher Jensen-polynomial hyperbolicity and connect to the xi Taylor coefficients

Chapter 8342

The Window6 Weight Polynomial: A Finite-Window Positivity Ladder on a Fibonacci Carrier

The Fibonacci-cube weight polynomial $W_m(x) = \sum_k b(m,k)x^k$, with $b(m,k)$ the number of length-m no-adjacent-1 words of weight k and $\sum_k b(m,k) = F_{m+2}$, is a finite carrier for the positivity conditions that the finite-window formulation of the Riemann Hypothesis asks of the Riemann ξ function. The Laguerre–Polya route reads RH as hyperbolicity of finite Jensen windows; its degree-two shadow is coefficient log-concavity, and its stronger shadows are the Newton inequalities, Toeplitz total positivity, and full real-rootedness. This chapter records four rungs of that ladder as axiom-free finite certificates on the Fibonacci carrier, together with the structural fact that ties them to this branch's spectral-determinant constants. None of this is a proof of RH; each rung is a finite positivity witness on a Fibonacci-cube surface.

8342.1 The Ladder

The degree-two rung, coefficient log-concavity $b(m,k)^2 \geq b(m,k-1)b(m,k+1)$, is recorded separately as the degree-two log-concavity rung. The present chapter adds three sharper rungs and the recurrence that forces them.

Theorem 8342.1 (Newton ultra-log-concavity)**.** *For $m = 6, \ldots, 10$ the weight coefficients satisfy the binomial-normalised Newton inequalities*

$$b(m,k-1)\,b(m,k+1)\,(k+1)\,(n-k+1) \;\leq\; b(m,k)^2\,k\,(n-k),$$

with n the degree of W_m. This is the coefficient shadow of real-rootedness that is strictly sharper than plain log-concavity.

Lean4 ✓ `BEDC.Derived.Window6WeightUltraLogConcave.window six to ten weight newton ultra log concave`

Theorem 8342.2 (Toeplitz total positivity)**.** *For $m = 6, \ldots, 10$ the weight-coefficient sequence has nonnegative 2×2 and 3×3 Toeplitz minors, the discrete Polya-frequency strengthening that a real-rooted polynomial's coefficient sequence carries.*

Lean4 ✓ BEDC.Derived.Window6WeightToeplitzTotalPositive.window six to ten weight toeplitz total positive certificate

↪ variant BEDC.Derived.Window6WeightToeplitzTotalPositive.window six to ten weight toeplitz three total positive

Theorem 8342.3 (Fibonacci recurrence and real-rootedness)**.** *The weight coefficients obey the Fibonacci polynomial recurrence*

$$b(m+2,k+1) = b(m+1,k+1) + b(m,k),$$

that is $W_m(x) = W_{m-1}(x) + x\,W_{m-2}(x)$*, with generating function* $\sum_m W_m(x)z^m = (1-z-xz^2)^{-1}$*. At* $x = 1$ *this is the golden-mean generating function* $(1-z-z^2)^{-1}$*. The recurrence forces interlacing, hence real-rootedness of every* W_m*; the linear-times-quadratic factorisation of* W_6 *is recorded explicitly.*

Lean4 ✓ BEDC.Derived.Window6WeightFibonacciRecurrence.window six weight fibonacci recurrence

↪ variant BEDC.Derived.Window6WeightFibonacciRecurrence.window six linear quadratic factor certificate

8342.2 The Spectral-Determinant Identification

The generating function of the ladder is not a new object. With the weighted golden-mean transfer matrix

$$T(y) = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix},$$

this branch's necklace spectral determinant already records

$$\det(I - xT(y)) = 1 - x - yx^2,$$

which at $y = 1$ is the Fibonacci generating function $1 - x - x^2$ whose reciprocal produces $F_{m+2} = \mathbf{1}^\top T(1)^{m-1}\mathbf{1}$ and $L_m = \operatorname{tr} T(1)^m$. Thus W_m is the coefficient stream of $\det(I - xT(y))^{-1}$, and the branch's fine constants are the same data viewed spectrally: the golden ratio and its conjugate $\{\varphi, -1/\varphi\}$ are the eigenvalues of $T(1)$, equivalently the reciprocal poles of $1 - x - x^2$, equivalently the roots governing the metallic recurrences $t^2 - t - 1$, $t^3 - t - 1$, $t^2 - 2t - 1$, $t^3 - t^2 - 1$ that this branch has certified elsewhere.

Theorem 8342.4 (Three readings of one determinant)**.** *The finite objects*

$$\textit{fine constants} \quad = \quad \operatorname{spec} T(y), \qquad \textit{RH-carrier} \quad = \quad \textit{coefficients of } \det(I - xT(y))^{-1},$$

are two readings of the single spectral determinant $\det(I - xT(y)) = 1 - x - yx^2$*; the positivity rungs of section 8342.1 are finite-window positivity conditions on the second reading, and the metallic constants are the first reading. The reflection-positivity link deficit of a golden self-substitution word,* $\Delta_d = D_d - |\mu_d|^2 \geq 0$*, is the same* 2×2 *Gram determinant shape as the degree-two rung, on the same golden constants.*

↪ variant BEDC.Derived.Window6WeightFibonacciRecurrence.window six weight fibonacci recurrence

Falsifiable prediction: A recomputation of the Lean finite enumerator for $m = 6, \ldots, 10$ that violates any displayed Newton, Toeplitz, or recurrence identity, or a W_m with a non-real root at these lengths, refutes this packet.

Independence witness: laguerre-polya, riemann-hypothesis-roadmap, reflection-positivity: this packet is finite Fibonacci-cube word enumeration with explicit natural-number positivity witnesses and a coefficient recurrence; it is not bijective to the analytic ξ-hyperbolicity packet or to any full RH packet, and it makes no analytic transport claim to the prime channels.

Closure status for NameCert$_{Window6WeightFiniteWindowRHLadder}$.

Theory closure: TheoryGate(NameCert$_{Window6WeightFiniteWindowRHLadder}$, scopedClosed)

Scope closed: Definition 8342.1, Definition 8342.2, and Definition 8342.3 close the Newton, Toeplitz, and Fibonacci-recurrence rungs on the Fibonacci carrier for $m = 6, \ldots, 10$, and Definition 8342.4 identifies the carrier with the golden-mean spectral determinant $1 - x - yx^2$.

Formal verification: FormalStatus(NameCert$_{Window6WeightFiniteWindowRHLadder}$, theoremChecked)

Lean target: `BEDC.Derived.Window6WeightFibonacciRecurrence.window\_six\_weight\_fibonacci\_recurrence`

Bridge status: none

Not claimed: not a proof of RH; finite Fibonacci-carrier positivity ladder and a spectral-determinant identification only, with no analytic transport to the Riemann ξ coefficients or the prime channels.

Upgrade path: lift the finite Newton/Toeplitz/real-rootedness rungs to all m, then connect the golden-mean spectral determinant to the analytic ξ Jensen windows.

Chapter 8343

The Golden-Mean Ruelle Zeta: A Finite Fibonacci Transfer Certificate

The golden-mean subshift of finite type is the space of two-sided binary sequences with no two adjacent 1s; its Artin–Mazur/Bowen–Lanford dynamical zeta function is a genuine zeta, built from periodic points and carrying a primitive-orbit Euler product. This chapter records that zeta as a finite Fibonacci transfer certificate: its reciprocal is the weighted spectral determinant $1 - t - yt^2$ of the golden-mean transfer matrix, its periodic-point counts obey the Fibonacci trace recurrence, and at unit weight its poles factor over the golden ring $\mathbb{Z}[\varphi]$ into the two golden eigenvalues. This is a real zeta theorem for an infinite dynamical system whose entire zeta is controlled by a finite Fibonacci transfer operator; it is not a statement about the Riemann zeta function and not about the Riemann ξ.

8343.1 The Weighted Periodic-Point Zeta

Let CycFib_n be the set of cyclic no-adjacent-1 words of length n, and weight each word by $y^{\#1}$. With the weighted transfer matrix

$$T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix},$$

the weighted periodic-point count is $C_n(y) = \mathrm{tr}(T_y^n)$, and the dynamical zeta is

$$\zeta_{\mathrm{GM}}(t, y) = \exp\left(\sum_{n \geq 1} \frac{C_n(y)}{n}\, t^n\right) = \frac{1}{\det(I - tT_y)} = \frac{1}{1 - t - yt^2}.$$

At $y = 1$ this is the Bowen–Lanford zeta of the golden-mean shift, with the Euler product $\prod_{d \geq 1}(1 - t^d)^{-\pi_d}$ over primitive orbit counts; that product is what makes this a zeta rather than a determinant in zeta clothing.

Theorem 8343.1 (Fibonacci trace recurrence)**.** *The weighted periodic-point polynomials obey the two-term recurrence forced by $T_y^2 = T_y + yI$,*

$$C_{n+2}(y) = C_{n+1}(y) + y\, C_n(y),$$

so at $y = 1$ the counts are the Lucas trace sequence $C_n(1) = 1, 3, 4, 7, 11, 18, 29, 47, \ldots$

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanRuelleZeta.C trace recurrence`

8343.2 The Finite Zeta Certificate

Rather than begin with the analytic exponential, the finite content is the log-derivative window. With $\chi_y(t) = 1 - t - yt^2$ and $S_N(t, y) = \sum_{n=1}^{N} C_n(y)t^{n-1}$, for every $N \geq 1$

$$(1 - t - yt^2)\, S_N(t, y) = 1 + 2yt - C_{N+1}(y)\, t^N - y\, C_N(y)\, t^{N+1},$$

equivalently $S_N \equiv (1 + 2yt)/(1 - t - yt^2) \pmod{t^N}$, which together with $\zeta_{\mathrm{GM}}(0, y) = 1$ pins down $\zeta_{\mathrm{GM}}(t, y) = (1 - t - yt^2)^{-1}$. The Lean certificate multiplies the truncated trace series by $1 - t - yt^2$ as genuine polynomial arithmetic and matches the right-hand side; the $N = 6$, $y = 1$ instance is

$$(1 - t - t^2)(1 + 3t + 4t^2 + 7t^3 + 11t^4 + 18t^5) = 1 + 2t - 29t^6 - 18t^7.$$

Theorem 8343.2 (Finite log-derivative window)**.** *The order-six weighted window satisfies the polynomial identity $\chi_y(t)\, S_6(t, y) = 1 + 2yt - C_7(y)t^6 - yC_6(y)t^7$ as an equality of polynomials in y and t, verified by explicit polynomial multiplication.*

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanRuelleZeta.ruelle zeta finite certificate N6`
↪ variant `BEDC.Derived.Window6GoldenMeanRuelleZeta.ruelle zeta certificate y1 N6`

Theorem 8343.3 (Determinant reading)**.** *The reciprocal zeta equals the weighted spectral determinant of the golden-mean transfer matrix,*

$$\det(I - tT_y) = 1 - t - yt^2,$$

so the zeta and the determinant are two readings of the single Fibonacci object $1 - t - yt^2$.

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanRuelleZeta.det I minus tT`

8343.3 The Golden Pole Factorisation

Specialising to $y = 1$ and working in the golden ring $\mathbb{Z}[\varphi]$ with $\varphi^2 = \varphi + 1$, the reciprocal zeta polynomial factors into golden linear terms.

Theorem 8343.4 (Golden pole factorisation)**.** *In $\mathbb{Z}[\varphi]$,*

$$1 - t - t^2 = (1 - \varphi t)\,(1 + (\varphi - 1)t),$$

so the poles of $\zeta_{\mathrm{GM}}(t, 1)$ are exactly $t = \varphi^{-1} = \varphi - 1$ and $t = -\varphi$.

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanRuelleZeta.golden pole factorization`
↪ variant `BEDC.Derived.Window6GoldenMeanRuelleZeta.zeta pole roots`

Theorem 8343.5 (Transfer spectrum)**.** *The transfer spectrum is exactly the golden pair*

$$\operatorname{spec} T_1 = \{\varphi,\ 1 - \varphi\} = \{\varphi,\ -\varphi^{-1}\},$$

two simple real spectral roots. This is the honest "fib-RH" statement here: the reciprocal zeta polynomial of the golden-mean shift has only simple real spectral roots, controlled by a finite Fibonacci transfer operator of entropy $\log \varphi$.

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanRuelleZeta.transfer spectrum`

Falsifiable prediction: A recomputation of the Lean transfer polynomials that violates the trace recurrence $C_{n+2} = C_{n+1} + yC_n$, the finite window $\chi_y S_6 = 1 + 2yt - C_7 t^6 - yC_6 t^7$, the determinant $\det(I - tT_y) = 1 - t - yt^2$, or the golden factorisation $1 - t - t^2 = (1 - \varphi t)(1 + (\varphi - 1)t)$, refutes this packet.

Independence witness: riemann-hypothesis-roadmap, riemann-zeta: this packet is the finite Fibonacci transfer certificate of the golden-mean subshift's own dynamical zeta; it is a real zeta for an infinite dynamical system, not the Riemann zeta, not the ξ function, and it makes no analytic transport claim to the critical line or the prime channels.

Closure status for NameCert$_{Window6GoldenMeanRuelleZeta}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanRuelleZeta}$, scopedClosed)

Scope closed: Definition 8343.1, Definition 8343.2, and Definition 8343.3 close the Fibonacci trace recurrence, the order-six finite zeta certificate, and the determinant reading of the golden-mean dynamical zeta $1-t-yt^2$; Definition 8343.4 and Definition 8343.5 close the golden pole factorisation and the transfer spectrum $\{\varphi, -\varphi^{-1}\}$ over $\mathbb{Z}[\varphi]$.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanRuelleZeta}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanRuelleZeta.ruelle\_zeta\_finite\_certificate\_N6`

Bridge status: none

Not claimed: not a proof of the Riemann Hypothesis and not about the Riemann ξ; this is the exact dynamical zeta of the golden-mean subshift of finite type, controlled by a finite Fibonacci transfer operator, with no analytic transport to the critical line or the prime channels.

Upgrade path: connect the golden-mean transfer spectrum to the analytic golden-mean thermodynamic pressure, and study the weighted Weil-type positivity of the finite zeta certificate as a bridge toward the Laguerre–Polya windows.

Chapter 8344

The Golden-Mean Primitive-Orbit Mobius Certificate

The golden-mean subshift has a primitive-orbit Euler product behind its dynamical zeta. This chapter records the finite weighted Mobius certificate for that Euler-product side: after the divisor substitution $y \mapsto y^d$, the primitive-orbit Mobius transform is divisible by the period and has a nonnegative integral quotient through period 12.

8344.1 The Weighted Primitive-Orbit Transform

Let $T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}$ be the weighted golden-mean transfer matrix and let

$$C_n(y) = \operatorname{tr}(T_y^n)$$

be the weighted cyclic period-count polynomial. The primitive-orbit Mobius transform is

$$A_n(y) = \sum_{d|n} \mu(d)\, C_{n/d}(y^d).$$

The substitution $y \mapsto y^d$ is part of the object: an orbit with a weight contribution in the shorter quotient period is repeated d times in the length-n orbit, so its weight exponent is multiplied by d.

Theorem 8344.1 (Finite primitive-orbit integrality and positivity)**.** *For each $1 \le n \le 12$, the weighted primitive-orbit Mobius transform satisfies*

$$A_n(y) = n\, P_n(y)$$

for a polynomial $P_n(y) \in \mathbb{N}[y]$. The certified quotients are

$$\begin{aligned}
P_1 &= 1,\\
P_2 &= y,\\
P_3 &= y,\\
P_4 &= y,\\
P_5 &= y + y^2,\\
P_6 &= y + y^2,\\
P_7 &= y + 2y^2 + y^3,\\
P_8 &= y + 2y^2 + 2y^3,\\
P_9 &= y + 3y^2 + 3y^3 + y^4,\\
P_{10} &= y + 3y^2 + 5y^3 + 2y^4,\\
P_{11} &= y + 4y^2 + 7y^3 + 5y^4 + y^5,\\
P_{12} &= y + 4y^2 + 9y^3 + 8y^4 + 3y^5.
\end{aligned}$$

Thus the Euler-product exponents in this finite window are integral and nonnegative after division by the period.

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanPrimitiveOrbit.golden mean primitive orbit mobius positivity`

At $y = 1$, the values $p_n = A_n(1)/n$ are the primitive golden-mean orbit counts of exact period n in the certified window:

$$1,\ 1,\ 1,\ 1,\ 2,\ 2,\ 4,\ 5,\ 8,\ 11,\ 18,\ 25.$$

Equivalently, the corresponding Mobius numerators are

$$A_1(1), \ldots, A_{12}(1) = 1, 2, 3, 4, 10, 12, 28, 40, 72, 110, 198, 300.$$

8344.2 What This Certificate Claims

This is the finite Euler-product / primitive-orbit positivity side of the golden-mean subshift zeta; it is not a proof of the Riemann Hypothesis and not about the Riemann xi. The determinant side and the trace-Hankel positivity side certify spectral polynomial identities and finite square-sum positivity. The Mobius certificate here is different finite content: it certifies the integrality and positivity of the Euler-product exponents after the primitive-orbit divisor transform, with the essential substitution $y \mapsto y^d$.

This certificate is therefore not reducible to the determinant identity. The determinant records a reciprocal zeta polynomial and its spectral transfer reading. The primitive-orbit Mobius transform records how periodic weighted words decompose into primitive cycles and checks that the quotient exponents are actual nonnegative orbit counts in the finite window $1 \le n \le 12$.

Falsifiable prediction: A recomputation of the weighted cyclic polynomials $C_n(y)$, the Mobius values $\mu(d)$ for $d \le 12$, or the divisor substitution $y \mapsto y^d$ that changes any equality $A_n(y) = nP_n(y)$ for $1 \le n \le 12$ refutes this packet.

Independence witness: riemann-hypothesis-roadmap, riemann-zeta: this packet is the finite Euler-product primitive-orbit integrality and positivity certificate for the golden-mean subshift

only; it is not a proof of the Riemann Hypothesis, not about the Riemann ξ, and it asserts no analytic transport to the critical line or to prime-channel zeta phenomena.

Closure status for NameCert$_{Window6GoldenMeanPrimitiveOrbit}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanPrimitiveOrbit}$, scopedClosed)

Scope closed: Definition 8344.1 closes the finite weighted primitive-orbit Mobius integrality and nonnegative quotient certificate for $1 \leq n \leq 12$, using the divisor substitution $y \mapsto y^d$.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanPrimitiveOrbit}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanPrimitiveOrbit.golden\_mean\_primitive\_orbit\_mobius\_positivity`

Bridge status: none

Not claimed: not a proof of RH; finite Euler-product primitive-orbit integrality/positivity for the golden-mean SFT only

Upgrade path: lift the finite $n \leq 12$ integrality to all n via a general Mobius-inversion argument, and connect the primitive-orbit exponents to the analytic golden-mean pressure

Chapter 8345

The Golden-Mean Parry Residue: A Cylinder-Resolved Finite RPF Certificate

The golden-mean subshift has a second finite certificate beyond the scalar determinant and beyond the primitive-orbit Euler product direction. This chapter records the cylinder-resolved Parry / Ruelle-Perron-Frobenius data carried by the weighted transfer matrix

$$T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}.$$

It keeps the entrywise cylinder counts, the Perron residue projector at $y = 1$, the exact secondary ψ^n error, and the formal pressure derivative. This is a finite Parry / Ruelle-Perron-Frobenius certificate for the golden-mean subshift; it is not a proof of the Riemann Hypothesis, not about the Riemann xi, and not a new Euler product.

8345.1 Cylinder-Resolved Transfer Powers

Let $F_0(y) = 0$, $F_1(y) = 1$, and

$$F_{n+2}(y) = F_{n+1}(y) + yF_n(y).$$

For $n \geq 1$, the finite matrix-power identity is

$$T_y^n = \begin{pmatrix} F_{n+1}(y) & yF_n(y) \\ F_n(y) & yF_{n-1}(y) \end{pmatrix}.$$

Thus the scalar trace is only the sum of two cylinder channels,

$$C_n(y) = \mathrm{tr}(T_y^n) = F_{n+1}(y) + yF_{n-1}(y),$$

while the two diagonal cylinders are retained separately as

$$N_0(n, y) = (T_y^n)_{00} = F_{n+1}(y), \qquad N_1(n, y) = (T_y^n)_{11} = yF_{n-1}(y).$$

The Lean certificate checks the order-eight window by direct polynomial matrix multiplication, including the entrywise data through the same recursive polynomial arithmetic used by the finite zeta packet.

Theorem 8345.1 (Cylinder-resolved finite transfer window)**.** *The weighted golden-mean transfer matrix has a checked order-eight entrywise cylinder certificate. In the list representation used by Lean, the bottom-left entry carries a trailing zero representative, while the mathematical polynomial is* $F_8(y) = 1 + 6y + 10y^2 + 4y^3$.

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanParryResidue.Ty pow cylinder 8`

8345.2 Adjugate and Pressure Shadow

The finite resolvent identity is the adjugate calculation

$$(I - tT_y)\begin{pmatrix} 1 & yt \\ t & 1-t \end{pmatrix} = (1 - t - yt^2)I_2.$$

This is more refined than the determinant alone: the adjugate remembers how the two cylinder coordinates propagate before taking the scalar determinant.

The same finite transfer data also carries the pressure derivative shadow. Differentiating the trace polynomial formally in y gives

$$\frac{d}{dy}C_n(y) = nF_{n-1}(y),$$

and therefore

$$y\,C_n'(y) = n(T_y^n)_{11}.$$

In the checked window, the identities include

$$C_4'(y) = 4(1+y), \qquad C_5'(y) = 5(1+2y), \qquad C_6'(y) = 6(1+3y+y^2),$$

and continue through the order-eight finite certificate.

Theorem 8345.2 (Adjugate and pressure derivative window)**.** *The adjugate identity for* $I - tT_y$ *and the formal derivative identities* $C_n'(y) = nF_{n-1}(y)$ *for* $2 \le n \le 8$ *are checked as exact finite polynomial computations.*

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanParryResidue.resolvent adjugate identity`
↪ variant `BEDC.Derived.Window6GoldenMeanParryResidue.pressure derivative 8`

8345.3 Golden Parry Residue

At $y = 1$, work in $\mathbb{Z}[\varphi]$ with $\varphi^2 = \varphi + 1$, put

$$\psi = 1 - \varphi, \qquad s = 2\varphi - 1,$$

and define

$$P = \begin{pmatrix} \varphi & 1 \\ 1 & \varphi - 1 \end{pmatrix}, \qquad Q = \begin{pmatrix} 1-\varphi & 1 \\ 1 & -\varphi \end{pmatrix}.$$

The denominator-cleared spectral projector statement is

$$sT^n = \varphi^n P - \psi^n Q.$$

On diagonal entries this gives the exact finite Parry-equidistribution error, with $L_n = C_n(1) = \varphi^n + \psi^n$,

$$s(T^n)_{00} = \varphi L_n - \psi^n, \qquad s(T^n)_{11} = (\varphi - 1)L_n + \psi^n.$$

The Lean certificate checks the golden spectral projector window through $n = 6$ and the two diagonal error formulas through the same window over the concrete ring $\mathbb{Z}[\varphi]$.

Theorem 8345.3 (Finite golden Parry residue)**.** *The order-six golden specialization verifies the denominator-cleared projector $sT^n = \varphi^n P - \psi^n Q$ and its diagonal $\pm\psi^n$ Parry-error formulas in $\mathbb{Z}[\varphi]$.*

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanParryResidue.golden mean parry residue certificate`

↪ variant `BEDC.Derived.Window6GoldenMeanParryResidue.golden parry spectral 6`

↪ variant `BEDC.Derived.Window6GoldenMeanParryResidue.golden parry error 6 00`

↪ variant `BEDC.Derived.Window6GoldenMeanParryResidue.golden parry error 6 11`

Falsifiable prediction: A recomputation of the finite polynomial matrix powers that changes the checked cylinder entries, the adjugate identity $(I - tT_y)\operatorname{adj}(I - tT_y) = (1 - t - yt^2)I_2$, the pressure derivative window $C_n' = nF_{n-1}$, or the golden diagonal Parry-error formulas refutes this packet.

Independence witness: determinant-side, primitive-orbit-side, riemann-hypothesis-roadmap: this packet is not reducible to the determinant side, because the scalar trace and determinant forget the Perron projector and the cylinder distribution; it is not reducible to the primitive-orbit side, because that side is the Euler product direction. This packet is a finite Parry / Ruelle-Perron-Frobenius cylinder-residue certificate for the golden-mean subshift only, not a proof of RH, not about the Riemann ξ, and not a new Euler product.

Closure status for NameCert$_{Window6GoldenMeanParryResidue}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanParryResidue}$, scopedClosed)

Scope closed: The chapter closes the finite golden-mean cylinder-power window, the adjugate resolvent identity, the pressure derivative window $C_n' = nF_{n-1}$ through $n = 8$, and the golden Parry residue diagonal error window through $n = 6$.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanParryResidue}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanParryResidue.golden\_mean\_parry\_residue\_certificate`

Bridge status: none

Not claimed: not a proof of RH; finite Parry/RPF cylinder-residue certificate for the golden-mean SFT only

Upgrade path: lift the finite cylinder/residue identities to the general weighted RPF operator and connect the pressure derivative to the analytic golden-mean equilibrium state

Chapter 8346

The Golden-Mean Hankel–Turan Discriminant Identity

The golden-mean weighted trace polynomials $C_n(y) = \mathrm{tr}(T_y^n)$ for

$$T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}$$

carry a finite Hankel–Turan minor whose scalar value is exactly the discriminant of the characteristic polynomial $t^2 - t - y$. The identity is

$$C_n(y)C_{n+2}(y) - C_{n+1}(y)^2 = (-y)^n(1 + 4y).$$

At $y = 1$, where $L_n = C_n(1)$ is the Lucas trace sequence $1, 3, 4, 7, 11, 18, 29, 47, \ldots$, this specializes to

$$L_n L_{n+2} - L_{n+1}^2 = 5(-1)^n.$$

Thus the same 2×2 trace minor exposes the golden discriminant $1 + 4y$, and the unit-weight constant is $5 = \sqrt{5}^2$.

Theorem 8346.1 (Finite golden-mean Hankel–Turan certificate)**.** *For the first eight positive trace indices, the polynomial identity*

$$C_n(y)C_{n+2}(y) - C_{n+1}(y)^2 = (-y)^n(1 + 4y)$$

is verified by direct coefficient arithmetic, and the same certificate records the Lucas specialization at $n = 4$:

$$L_4 L_6 - L_5^2 = 5.$$

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanHankelTuran.golden mean hankel turan discriminant certificate`

This is a finite Hankel–Turan discriminant identity for the golden-mean trace sequence; it is not a proof of the Riemann Hypothesis and not about the Riemann xi. Its content is the scalar consequence of the second-order trace recurrence attached to $T_y^2 = T_y + yI$, recorded as explicit polynomial arithmetic. The determinant side and the trace side meet through the same characteristic polynomial $t^2 - t - y$, but this packet claims only the stated finite discriminant window and the displayed Lucas specialization.

Falsifiable prediction: A recomputation of the polynomial trace sequence that changes any coefficient of $C_n(y)C_{n+2}(y) - C_{n+1}(y)^2$ for $1 \le n \le 8$, or changes the Lucas value $L_4 L_6 - L_5^2$, refutes this certificate.

Independence witness: riemann-hypothesis-roadmap, riemann-xi: this packet is a finite Hankel–Turan discriminant identity for the golden-mean trace sequence only; it makes no Riemann Hypothesis claim and no analytic statement about the Riemann ξ function.

Closure status for NameCert$_{Window6GoldenMeanHankelTuran}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanHankelTuran}$, scopedClosed)

Scope closed: The Lean certificate closes the coefficient identities $C_n(y)C_{n+2}(y) - C_{n+1}(y)^2 = (-y)^n(1+4y)$ for $1 \le n \le 8$, together with the unit-weight Lucas discriminant instance $L_4L_6 - L_5^2 = 5$.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanHankelTuran}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanHankelTuran.golden\_mean\_hankel\_turan\_discriminant\_certificate`

Bridge status: none

Not claimed: not a proof of RH; finite Hankel-Turan discriminant identity for the golden-mean trace sequence only

Upgrade path: lift the finite $n \le 8$ identity to all n via the second-order trace recurrence, and connect the discriminant $1+4y$ to the golden Jensen windows

Chapter 8347

The Golden-Mean Zeta Denominator as a Fifth-Cyclotomic Pentagon Certificate

The golden-mean reciprocal zeta denominator $1-t-t^2$ has a finite fifth-cyclotomic lift. Let $\zeta = \zeta_5$ satisfy $\Phi_5(\zeta) = \zeta^4 + \zeta^3 + \zeta^2 + \zeta + 1 = 0$, and put

$$D_k(t,y) = 1 - \zeta^k t - y\zeta^{2k}t^2 = \det(I - \zeta^k t T_y), \qquad T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}.$$

The certificate below records the primitive fifth-cyclotomic norm of the golden-mean denominator, its scalar y-weighted norm, and the pentagon Gram bridge to the golden field. It is a finite algebraic identity in $\mathbb{Z}[\zeta_5]$ and in $\mathbb{Z}[y,t]$, not a numerical coincidence.

8347.1 The Fifth-Cyclotomic Twisted Norm

Define

$$\begin{aligned} Q_5(t,y) = 1 + t + (1+y)t^2 + (1+2y)t^3 + (1+3y+y^2)t^4 \\ - (y+2y^2)t^5 + (y^2+y^3)t^6 - y^3t^7 + y^4t^8. \end{aligned}$$

Then the scalar norm identity is

$$(1-t-yt^2)Q_5(t,y) = 1 - (1+5y+5y^2)t^5 - y^5t^{10} = \det(I - t^5T_y^5).$$

At $y=1$, the primitive cyclotomic twists collapse from $\mathbb{Z}[\zeta_5]$ to integer coefficients:

$$\prod_{k=1}^{4}(1-\zeta^k t - \zeta^{2k}t^2) = 1 + t + 2t^2 + 3t^3 + 5t^4 - 3t^5 + 2t^6 - t^7 + t^8,$$

and the full fifth norm is

$$\prod_{k=0}^{4}(1-\zeta^k t - \zeta^{2k}t^2) = 1 - 11t^5 - t^{10}.$$

The number 11 is the fifth Lucas trace $C_5(1) = \mathrm{tr}(T_1^5)$.

Theorem 8347.1 (Fifth-cyclotomic twisted norm)**.** *The four primitive ζ_5-twists of $1 - t - t^2$ multiply to $Q_5(t, 1)$, the five twists multiply to $1 - 11t^5 - t^{10}$, and the bivariate scalar identity*

$$(1 - t - yt^2)Q_5(t, y) = 1 - (1 + 5y + 5y^2)t^5 - y^5t^{10}$$

holds by finite polynomial multiplication.

Lean4 ✓ BEDC.Derived.Window6GoldenMeanCyclotomicPentagon.golden mean cyclotomic pentagon certificate

8347.2 The Pentagon Bridge

Let

$$u = \zeta + \zeta^{-1} = \zeta + \zeta^4.$$

Since

$$u^2 + u - 1 = \zeta^2 + \zeta + 1 + \zeta^{-1} + \zeta^{-2} = \zeta^{-2}\Phi_5(\zeta) = 0,$$

the real element u is $2\cos 72^\circ = (\sqrt{5} - 1)/2 = 1/\varphi$. Thus $1 + u = \varphi$ and $-u = -1/\varphi$, and the golden denominator factors in $\mathbb{Z}[\zeta_5][t]$ as

$$1 - t - t^2 = (1 - (1 + u)t)(1 + ut).$$

Equivalently, the transfer characteristic polynomial

$$x^2 - x - 1 = (x - (1 + u))(x + u)$$

has spectrum $\{1 + u, -u\} = \{\varphi, -1/\varphi\}$.

The same pentagon element appears in the Gram matrix

$$G_{ij} = \zeta^{i-j} + \zeta^{j-i}, \qquad i, j \in \mathbb{Z}/5\mathbb{Z}.$$

Its first row is

$$[2,\ u,\ -1 - u,\ -1 - u,\ u],$$

and finite multiplication gives

$$G^2 = 5G.$$

The two nontrivial Gram inner products $u = 1/\varphi$ and $-1 - u = -\varphi$ are exactly the two roots of $1 - t - t^2 = 0$.

Theorem 8347.2 (Pentagon Gram and golden factorisation)**.** *In $\mathbb{Z}[\zeta_5]$, the element $u = \zeta + \zeta^{-1}$ satisfies $u^2 + u - 1 = 0$, the golden denominator factors as*

$$1 - t - t^2 = (1 - (1 + u)t)(1 + ut),$$

and the pentagon Gram matrix $G_{ij} = \zeta^{i-j} + \zeta^{j-i}$ satisfies $G^2 = 5G$.

↪ variant BEDC.Derived.Window6GoldenMeanCyclotomicPentagon.u min poly
↪ variant BEDC.Derived.Window6GoldenMeanCyclotomicPentagon.golden denominator cyclotomic factor
↪ variant BEDC.Derived.Window6GoldenMeanCyclotomicPentagon.pentagon gram idempotent

8347.3 Scope

This is a finite cyclotomic-norm and pentagon-Gram identity bridging the golden-mean zeta to the fifth cyclotomic field; it is not a proof of the Riemann Hypothesis, not about the Riemann xi, and not a zero-line statement.

The precise Φ_5 and pentagon link is canonical at $y = 1$: the golden field is the maximal real subfield of $\mathbb{Q}(\zeta_5)$, and $u = \zeta + \zeta^{-1} = 1/\varphi$ satisfies $u^2 + u - 1 = 0$. The scalar twisted norm holds for all y, but the generic transfer spectrum lies in $\mathbb{Q}(\sqrt{1+4y})$, not canonically in the fifth cyclotomic field. The pentagon Gram identity $G^2 = 5G$ also does not by itself prove any reflection-positivity deficit theorem; that would require a separately defined reflection-positivity functional.

Falsifiable prediction: A direct finite recomputation in $\mathbb{Z}[\zeta_5]$ that violates $u^2 + u - 1 = 0$, the primitive twist product $\prod_{k=1}^{4} D_k(t,1) = Q_5(t,1)$, the full fifth norm $\prod_{k=0}^{4} D_k(t,1) = 1 - 11t^5 - t^{10}$, the bivariate scalar norm $(1 - t - yt^2)Q_5(t,y) = 1 - (1 + 5y + 5y^2)t^5 - y^5t^{10}$, the factorisation $1 - t - t^2 = (1 - (1+u)t)(1+ut)$, or $G^2 = 5G$ refutes this packet.

Independence witness: riemann-hypothesis-roadmap, riemann-zeta, reflection-positivity-deficit: this packet is the finite fifth-cyclotomic twisted norm and pentagon Gram certificate for the golden-mean zeta denominator only; it makes no claim about the Riemann ξ, the critical line, prime-channel zero statistics, or any reflection-positivity deficit theorem.

Closure status for NameCert$_{Window6GoldenMeanCyclotomicPentagon}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanCyclotomicPentagon}$, scopedClosed)

Scope closed: Definition 8347.1 closes the finite primitive and full fifth-cyclotomic twisted norm identities for the golden-mean denominator, including the scalar bivariate norm in $\mathbb{Z}[y,t]$; Definition 8347.2 closes the pentagon bridge $u^2 + u - 1 = 0$, the golden factorisation, and the Gram identity $G^2 = 5G$ in $\mathbb{Z}[\zeta_5]$.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanCyclotomicPentagon}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanCyclotomicPentagon.golden\_mean\_cyclotomic\_pentagon\_certificate`

Bridge status: none

Not claimed: not a proof of RH; finite fifth-cyclotomic twisted norm and pentagon Gram identities for the golden-mean zeta denominator only; the RP-deficit theorem is not proven here.

Upgrade path: extend the twisted norm to the full $\mathbb{Z}[\zeta_5][y]$ factorisation for all y, and connect the pentagon Gram to a defined reflection-positivity deficit functional.

Chapter 8348

The Golden-Mean Shift Equivalence: A Finite Presentation-Invariance Certificate

This chapter records a finite strong-shift-equivalence certificate for the golden-mean subshift zeta. The certificate does not change the dynamical system; it changes the finite presentation from the two-state transfer matrix to a three-state edge-expansion matrix and verifies that the reciprocal zeta denominator and the periodic trace window are unchanged.

8348.1 The Rectangular Factors

Let

$$R = \begin{pmatrix} 1 & y & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad S = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Then the two rectangular products are

$$RS = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix} = T_y, \qquad SR = \begin{pmatrix} 1 & y & 0 \\ 0 & 0 & 1 \\ 1 & y & 0 \end{pmatrix} = E_y.$$

Thus R and S exhibit the two-state golden-mean transfer presentation T_y and the three-state edge-expansion presentation E_y as the two sides of an elementary strong-shift-equivalence datum.

8348.2 The Presentation-Invariance Certificate

The determinant side gives the same reciprocal zeta polynomial in both finite presentations:

$$\det(I_2 - tT_y) = \det(I_3 - tE_y) = 1 - t - yt^2.$$

The trace side gives the same weighted periodic-point window. For $1 \le n \le 8$, the Lean certificate computes the powers of E_y by genuine 3×3 matrix multiplication over $\mathbb{Z}[y]$ and verifies

$$\operatorname{tr}(E_y^n) = C_n(y),$$

where

$$
\begin{aligned}
C_1(y) &= 1,\\
C_2(y) &= 1 + 2y,\\
C_3(y) &= 1 + 3y,\\
C_4(y) &= 1 + 4y + 2y^2,\\
C_5(y) &= 1 + 5y + 5y^2,\\
C_6(y) &= 1 + 6y + 9y^2 + 2y^3,\\
C_7(y) &= 1 + 7y + 14y^2 + 7y^3,\\
C_8(y) &= 1 + 8y + 20y^2 + 16y^3 + 2y^4.
\end{aligned}
$$

Theorem 8348.1 (Golden-mean strong-shift-equivalence certificate)**.** *The rectangular matrices R and S satisfy $RS = T_y$ and $SR = E_y$; the two finite presentations have the common reciprocal zeta denominator $1 - t - yt^2$; and the trace window $\mathrm{tr}(E_y^n) = C_n(y)$ is checked for $1 \le n \le 8$ by explicit polynomial matrix multiplication.*

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanShiftEquivalence.golden mean shift equivalence certificate`

This is a finite presentation-invariance (strong shift equivalence) certificate for the golden-mean SFT zeta; it is not a proof of the Riemann Hypothesis and not about the Riemann xi. The point is local and finite: the determinant and trace data of the golden-mean SFT zeta survive the elementary presentation change from T_y to its edge-expansion presentation E_y. In this sense it is a finite witness for the Williams presentation-invariance principle, while remaining independent content from the determinant calculation itself.

Falsifiable prediction: A recomputation of the Lean rectangular products, determinant polynomials, or 3×3 trace powers that violates $RS = T_y$, $SR = E_y$, $\det(I_3 - tE_y) = 1 - t - yt^2$, or $\mathrm{tr}(E_y^n) = C_n(y)$ for the checked window refutes this packet.

Independence witness: riemann-hypothesis-roadmap, riemann-xi: this packet is only a finite strong-shift-equivalence presentation-invariance certificate for the golden-mean SFT zeta; it is not a proof of the Riemann Hypothesis and it is not about the Riemann xi.

Closure status for NameCert$_{Window6GoldenMeanShiftEquivalence}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanShiftEquivalence}$, scopedClosed)

Scope closed: The rectangular products $RS = T_y$ and $SR = E_y$, the common denominator $\det(I_2 - tT_y) = \det(I_3 - tE_y) = 1 - t - yt^2$, and the checked trace window $\mathrm{tr}(E_y^n) = C_n(y)$ for $1 \le n \le 8$ close the finite presentation-invariance certificate for the golden-mean SFT zeta.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanShiftEquivalence}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanShiftEquivalence.golden\_mean\_shift\_equivalence\_certificate`

Bridge status: none

Not claimed: not a proof of RH; finite strong-shift-equivalence presentation invariance for the golden-mean SFT zeta only

Upgrade path: extend to a general strong-shift-equivalence chain and connect to Williams classification of the golden-mean SFT

Chapter 8349

The Golden-Mean Quadratic-Algebra Capstone

The golden-mean zeta facets have a closest honest master identity, but not a single finite identity generating every certificate in the surrounding family. The organizing object is the quadratic algebra

$$Q_y = \mathbb{Z}[y][x]/(x^2 - x - y),$$

with every element written as $ax + b$ over $\mathbb{Z}[y, t]$ and conjugation $\overline{x} = 1 - x$. The root equation is $x^2 = x + y$, so multiplication reduces by

$$(ax + b)(cx + d) = (ac + ad + bc)x + (acy + bd).$$

The trace and norm are

$$\mathrm{Tr}(ax + b) = a + 2b, \qquad N(ax + b) = b^2 + ab - ya^2,$$

and hence $\mathrm{Tr}(x) = 1$ and $N(x) = -y$.

8349.1 The Master Norm Identity

The central identity is the norm computation

$$(1 - tx)(1 + t(x - 1)) = 1 - t - yt^2.$$

It is a norm identity in $Q_y[t]$: the second factor is the conjugate of $1 - tx$, because $\overline{x} = 1 - x$. Equivalently, with the regular representation

$$\rho(ax + b) = \begin{pmatrix} a + b & ya \\ a & b \end{pmatrix}, \qquad \rho(x) = T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix},$$

one has $\det \rho(ax + b) = N(ax + b)$ and $\mathrm{tr}\, \rho(ax + b) = \mathrm{Tr}(ax + b)$. The matrix form of the same identity is

$$(I - tT_y)(I + t(T_y - I)) = (1 - t - yt^2)I,$$

which is the Cayley–Hamilton reading of $T_y^2 = T_y + yI$.

Theorem 8349.1 (Quadratic-algebra capstone certificate)**.** *In the finite $\mathbb{Z}[y,t]$ model of Q_y, the root identity $x^2 = x + y$, the conjugation identity $\overline{x} = 1 - x$, the norm and trace values $N(x) = -y$ and $\mathrm{Tr}(x) = 1$, the master norm identity $(1-tx)(1+t(x-1)) = 1-t-yt^2$, the regular representation value $\rho(x) = T_y$, concrete determinant and trace readings of $\rho(x)$, six Fibonacci power identities, six Hankel–Cassini identities, and two finite truncated resolvent identities are all checked by Lean without axioms.*

Lean4 ✓ `BEDC.Derived.Window6GoldenMeanQuadraticCapstone.golden mean quadratic capstone certificate`

8349.2 Powers, Cassini, and Truncated Resolvents

Let $F_0(y) = 0$, $F_1(y) = 1$, and

$$F_{n+2}(y) = F_{n+1}(y) + yF_n(y).$$

The quadratic relation gives the finite power law

$$x^{n+1} = F_{n+1}(y)x + yF_n(y),$$

verified in Lean for the first six nontrivial instances. Taking traces gives the cyclic trace polynomials $C_n(y) = \mathrm{Tr}(x^n)$, whose specialization at $y = 1$ is the Lucas trace sequence.

The same conjugation gives the Hankel–Turan discriminant without a separate determinant argument:

$$C_n(y)C_{n+2}(y) - C_{n+1}(y)^2 = (x\overline{x})^n(x - \overline{x})^2 = (-y)^n(1 + 4y).$$

Here $x\overline{x} = -y$ and

$$(x - \overline{x})^2 = (2x - 1)^2 = 1 + 4y.$$

The capstone certificate checks this Cassini identity for $n = 1, \ldots, 6$ as polynomial data.

The finite resolvent identity is also a direct consequence of the same denominator:

$$(1 - t - yt^2)\sum_{i=0}^{N} x^i t^i = 1 + t(x - 1) - t^{N+1}x^{N+1} - yt^{N+2}x^N.$$

The Lean certificate records the concrete $N = 1$ and $N = 2$ truncations as finite polynomial computations. These truncations are not a replacement for the primitive-orbit product; they are the quadratic-algebra denominator viewed inside $Q_y[t]$.

8349.3 Honest Scope Across the Six Facets

There is no single finite identity generating all six facets; the closest honest master is the quadratic-algebra norm identity. This is the honest organizing master of the golden-mean SFT zeta facets, not a grand unification and not a proof of the Riemann Hypothesis; the independent facets remain separate certificates.

Facet	Generation by the quadratic master
A determinant/zeta/trace-recurrence	**YES directly**. The determinant, resolvent denominator, and trace recurrence come from $T_y^2 = T_y + yI$ and $\det(I - tT_y) = 1 - t - yt^2$; the $\mathbb{Z}[\varphi]$ splitting still needs $\varphi^2 = \varphi + 1$.
B Euler-product/primitive-orbit	**PARTLY**. The master gives $C_n(y) = \mathrm{Tr}(x^n)$ and therefore the input to the Mobius transform, but positivity $P_n \in \mathbb{N}[y]$ is an independent necklace-counting theorem.
C Parry/RPF/cylinder	**MOSTLY**. The power law, cylinder recurrences, and derivative identity $C_n' = nF_{n-1}$ are governed by $x^2 = x + y$; the golden spectral projector belongs to the $\mathbb{Z}[\varphi]$ specialization.
D Hankel–Turan/discriminant	**YES**. Conjugation gives $x\overline{x} = -y$ and $(x - \overline{x})^2 = 1 + 4y$, hence the Cassini–Hankel identity.
E fifth-cyclotomic/pentagon	**PARTLY**. Cyclic norms use the same norm mechanism together with $\prod_{k=0}^{4}(1 - \zeta^k a) = 1 - a^5$; the pentagon Gram relation $G^2 = 5G$ and $u^2 + u - 1 = 0$ remain independent cyclotomic geometry.
F strong-shift-equivalence	**NO**. The presentation data for strong-shift-equivalence are independent finite certificates, not consequences of the quadratic norm identity.

Thus the capstone is an organizing certificate, not a total generator. It explains why the determinant, trace recurrence, resolvent denominator, power law, and Hankel discriminant repeatedly share the same polynomial $1 - t - yt^2$, while marking the primitive-necklace positivity, golden-ring projector, pentagon geometry, and presentation data as independent inputs.

Falsifiable prediction: A recomputation of the finite $\mathbb{Z}[y, t]$ quadratic-algebra model that violates $x^2 = x + y$, $N(x) = -y$, $\mathrm{Tr}(x) = 1$, $(1 - tx)(1 + t(x - 1)) = 1 - t - yt^2$, $\rho(x) = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}$, or the checked Cassini windows refutes this packet.

Independence witness: golden-mean-facet-scope: the quadratic master does not generate the primitive-orbit positivity theorem, the $\mathbb{Z}[\varphi]$ spectral projector, the pentagon Gram geometry, the strong-shift-equivalence presentation data, or any proof of the Riemann Hypothesis.

Closure status for NameCert$_{Window6GoldenMeanQuadraticCapstone}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanQuadraticCapstone}$, scopedClosed)

Scope closed: The chapter closes the finite quadratic-algebra master identity for $Q_y = \mathbb{Z}[y][x]/(x^2 - x - y)$, its regular representation at x, six finite power and Hankel windows, and the honest facet-scope table for the golden-mean SFT zeta family.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanQuadraticCapstone}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanQuadraticCapstone.golden\_mean\_quadratic\_capstone\_certificate`

Bridge status: none

Not claimed: not a single-generator unification of all facets; B positivity, C/E's $\mathbb{Z}[\varphi]$/pentagon geometry, and F presentation data remain independent certificates; not a proof of RH.

Upgrade path: formalize the universal ring-hom instance over an abstract commutative base and derive each dependent facet as a corollary.

Chapter 8350

The Finite Golden-Mean Zeta Program

The golden-mean subshift of finite type, the space of two-sided binary sequences with no two adjacent 1s, has a single weighted dynamical zeta

$$\zeta(t, y) = \frac{1}{1 - t - yt^2},$$

whose reciprocal is the spectral determinant of the weighted transfer matrix $T_y = \begin{pmatrix} 1 & y \\ 1 & 0 \end{pmatrix}$. This chapter is the organizing summary of the finite, axiom-free certificate program built around that one object. It records the central algebraic identity, the table of finite facets that are shadows of it, and a frank statement of the program's boundary. It proves no new theorem; it names the completed envelope.

8350.1 The Central Norm Identity

The organizing algebra is the rank-two quadratic ring

$$Q_y = \mathbb{Z}[y][x]/(x^2 - x - y), \qquad \bar{x} = 1 - x,$$

whose regular representation of multiplication by x in the basis $(x, 1)$ is exactly T_y. The single identity from which the determinant face of the whole program descends is the quadratic norm

$$N_{Q_y/\mathbb{Z}[y]}(1 - xt) = (1 - xt)(1 - \bar{x}t) = 1 - t - yt^2,$$

equivalently the matrix form $(I - tT_y)(I + t(T_y - I)) = (1 - t - yt^2)I$, resting on the root relation $x^2 = x + y$ (that is $T_y^2 = T_y + yI$). The trace stream $C_n(y) = \mathrm{tr}(T_y^n) = \mathrm{Tr}_{Q_y}(x^n)$ and the Fibonacci-polynomial power law $x^{n+1} = F_{n+1}(y)\, x + yF_n(y)$ are the finite carriers that the individual facets read off in different ways.

Theorem 8350.1 (Organizing norm identity)**.** *In $Q_y[t]$ the quadratic norm of $1 - xt$ equals the golden-mean zeta denominator $1 - t - yt^2$, and its regular representation is the finite matrix identity $(I - tT_y)(I + t(T_y - I)) = (1 - t - yt^2)I$. Every determinant-side fact of the program is a reading of this one norm.*

↪ variant `BEDC.Derived.Window6GoldenMeanQuadraticCapstone.master norm`

8350.2 The Facet Table

Each finite facet of the program is a certificate about the same object, and the honest relation of each to the central norm is as follows.

Facet	Finite meaning	Relation to the norm i
Determinant/spectrum	$\det(I - tT_y) = 1 - t - yt^2$, $\mathbb{Z}[\varphi]$ poles	directly generated
Euler product/orbits	Mobius primitive-orbit positivity $A_n = nP_n$, $P_n \in \mathbb{N}[y]$	trace-log of the norm; posi
Parry/RPF/cylinder	T_y^n cylinder counts, $C'_n = nF_{n-1}$, golden projector	spectral specialization; the
Hankel–Turan	$C_nC_{n+2} - C_{n+1}^2 = (-y)^n(1 + 4y)$	directly the conjugate Cas
Fifth-cyclotomic/pentagon	$\prod_k(1 - \zeta^k tx)$, $u = 2\cos 72^\circ = 1/\varphi$, $G^2 = 5G$	golden specialization $y = 1$
Strong shift equivalence	$RS = T_y$, $SR = E_y$, equal zeta and traces	external dynamical invaria
Quadratic capstone	Q_y norm identity itself	the organizing algebra

The table is the honest content: the determinant, trace recurrence, resolvent denominator, power law, and Hankel discriminant repeatedly share the polynomial $1 - t - yt^2$ because they are all readings of the quadratic norm, while the primitive-necklace positivity, the golden-ring spectral projector, the pentagon Gram geometry, and the strong-shift-equivalence presentation data are genuinely independent finite inputs, not consequences of the norm.

8350.3 The Program Boundary

This is a finite algebraic and dynamical-zeta certificate program for the golden-mean subshift of finite type. It is complete as such: the shift-of-finite-type identity $\zeta(t) = 1/\det(I - tT_y)$ already controls the zeta, and the facets exhaust the main finite meanings of that determinant. It does not claim, and this program does not contain, any Riemann-Hypothesis-type analytic depth, any general subshift or sofic zeta classification, any prime-orbit asymptotics or analytic continuation, or a thermodynamic-formalism library. The golden-mean zeta is not the Riemann zeta and carries no critical line; the pentagon and $\mathbb{Z}[\varphi]$ arithmetic is exact golden-field structure, not a spectral statement about $\zeta(s)$.

Falsifiable prediction: Any finite recomputation contradicting Definition 8350.1, or a facet in the table whose displayed finite identity fails at the enumerated lengths, refutes the corresponding certificate and this summary's placement of it.

Independence witness: golden-mean-zeta-program: the six facets plus the quadratic capstone are the natural complete finite envelope of the golden-mean SFT zeta; the primitive-orbit positivity, $\mathbb{Z}[\varphi]$ spectral projector, pentagon Gram geometry, and strong-shift-equivalence presentation data are independent certificates, and nothing here is a proof of the Riemann Hypothesis or a general SFT zeta theory.

Closure status for NameCert$_{Window6GoldenMeanZetaProgramHub}$.

Theory closure: TheoryGate(NameCert$_{Window6GoldenMeanZetaProgramHub}$, scopedClosed)

Scope closed: Definition 8350.1 records the organizing quadratic norm identity, and section 8350.2 places each finite facet as a reading of it or as an independent certificate; the program is complete as a finite golden-mean-zeta certificate envelope.

Formal verification: FormalStatus(NameCert$_{Window6GoldenMeanZetaProgramHub}$, theoremChecked)

Lean target: `BEDC.Derived.Window6GoldenMeanQuadraticCapstone.golden\_mean\_quadratic\_capstone\_certificate`

Bridge status: none

Not claimed: not a proof of RH, not a general SFT or sofic zeta theory, not analytic prime-orbit asymptotics or thermodynamic formalism; the pentagon and primitive-orbit and presentation facets remain independent finite certificates.

Upgrade path: move from the finite certificate envelope to the analytic golden-mean pressure and equilibrium theory, which is outside the finite axiom-free scope of this program.

Chapter 8351

A Concrete Naming Certificate for $\mathsf{DyadicTentFunction}^{\uparrow}$

Following Schauder and Bishop–Bridges constructive approximation, the $\mathsf{DyadicTentFunction}^{\uparrow}$ packet isolates the finite tent basis element used by dyadic interpolation arguments before a consumer invokes an entire Faber–Schauder system or a polygonal approximation chapter. It sits between the rational and real readback surfaces of chapter 291 and chapter 38, the system-level construction of chapter 6414, the interpolation route of chapter 8220, and the approximation handoff of chapter 3906.

Definition 8351.1 (Dyadic tent-function carrier)**.** *A* $\mathsf{DyadicTentFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (C, S, H, L, R, E, D, A, M, P, N)$$

with the following displayed rows:

1. *C is the dyadic center row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *S is the dyadic support-window row, recording the left and right dyadic endpoints.*
3. *H is the height row, normalized as the visible peak value.*
4. *L and R are the left-slope and right-slope rows, each carried by finite dyadic arithmetic.*
5. *E records endpoint vanishing and peak agreement for the three displayed dyadic breakpoints.*
6. *D is the dyadic-to-real readback row consumed before any* $\mathsf{Real}^{\uparrow}$ *graph evaluation.*
7. *A records the affine-piece admission rows for the two support halves.*
8. *M records componentwise* hsame *transport for center, support, height, slopes, endpoint rows, readback, and affine pieces.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FaberSchauderSystem}^{\uparrow}$*,* $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$ $\mathsf{PiecewiseLinearApproximation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTentFunction}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite coefficient family, an external real-valued function object, a completed approximation theorem, or a selected ambient representative.

Theorem 8351.2 (DyadicTentFunction NameCert obligations)**.** *For the carrier of Definition 8351.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicTentFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (C, S, H, L, R, E, D, A, M, P, N),$$

dyadic center display through C*, support-window display through* S*, height display through* H*, slope display through* L, R*, endpoint and peak display through* E*, dyadic-to-real readback through* D*, affine-piece admission through* A*, componentwise transport through* M*, provenance through* P*, and local naming through* N*.*

Proof. Project the finite carrier. The center, support, height, slope, endpoint, readback, and affine-piece rows are all displayed before the provenance and naming rows are read.

The dyadic rows are consumed through $\mathsf{DyadicRatCore}^{\uparrow}$, while the graph-facing handoff to $\mathsf{Real}^{\uparrow}$ occurs only through the readback row D. The system and interpolation chapters may reuse this packet as one basis element, but the packet itself does not expose a Faber–Schauder family or a polygonal approximation theorem.

The structural rows preserve the same displayed rows. Thus the listed fields exhaust the local NameCert surface. □

Theorem 8351.3 (DyadicTentFunction modulus handoff)**.** *Every accepted* $\mathsf{DyadicTentFunction}^{\uparrow}$ *packet exposes a finite modulus handoff to dyadic interpolation consumers through the same support-window, slope, endpoint, and dyadic-to-real readback rows*

$$S, L, R, E, D, A, M, P, N.$$

A $\mathsf{FaberSchauderSystem}^{\uparrow}$*,* $\mathsf{DyadicPiecewiseLinearInterpolation}^{\uparrow}$*, or* $\mathsf{PiecewiseLinearApproximation}^{\uparrow}$ *consumer may use the packet as a single tent primitive only after those rows have been displayed, and cannot infer an infinite basis expansion, a global approximation bound, or a hidden real-function representative from the packet alone.*

Proof. Apply the NameCert obligations of Definition 8351.2. The support, slope, endpoint, readback, and affine-piece rows are separate displayed coordinates of the same carrier.

The interpolation-facing handoff reads S, L, R, E before it reads the real graph route through D. Transport through M preserves those rows componentwise, so refining the ambient interpolation context does not detach the tent primitive from its dyadic support and slope data.

The excluded objects are not carrier coordinates. Hence the packet supplies only the finite tent-function primitive and its modulus-facing handoff. □

Falsifiable prediction: Any interpolation consumer that reads a DyadicTentFunctionUp graph row while omitting the displayed support-window and slope rows is rejected by the local classifier.

Independence witness: Nearest siblings: faberschaudersystem, dyadicpiecewiselinearinterpolation, piecewiselinearapproximation, dyadicratcore, real. The carrier is not bijective to the system-level or approximation-level packets because it stores one finite tent primitive rather than a family, mesh, or global error ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTentFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTentFunction}^{\uparrow}$ packet is generated from dyadic center, support, height, slope, endpoint, readback, affine-piece, transport, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicTentFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter records the finite carrier and local NameCert surface for one dyadic tent primitive before system-level and approximation-level consumers read it.

Formal verification: FormalStatus(NameCert$_{\text{DyadicTentFunction}\uparrow}$, formalTarget)
Lean target: `BEDC.Derived.DyadicTentFunctionUp`
Bridge status: none
Origin: human
Not claimed: It does not close a Faber–Schauder basis theorem, uniform approximation theorem, external function equality, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, dyadic-to-real readback, affine-piece replay, and consumer handoff for the same finite packet.

Lean4 (statement) `BEDC.Derived.DyadicTentFunctionUp`

Chapter 8352

A Concrete Naming Certificate for $\mathsf{Grassmannian}^{\uparrow}$

Following Griffiths and Harris on Grassmannians and Plucker coordinates, the $\mathsf{Grassmannian}^{\uparrow}$ packet records a finite subspace-row certificate before any consumer reads a Grassmannian as an ambient projective variety. It sits after the vector-space carrier of chapter 47, the finite matrix ledger of chapter 49, the determinant boundary of chapter 84, the alternating-word carrier of chapter 5144, and the homogeneous-coordinate surface of chapter 207. The packet names only displayed spanning rows, rank minors, wedge coordinates, projective incidence, transport, replay, provenance, and local naming; it does not select a host quotient vector space, import a scheme, assert Schubert calculus, or expose an ambient moduli object.

Definition 8352.1 (Grassmannian carrier)**.** *A* $\mathsf{Grassmannian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, M, D, E, Q, H, C, P, N)$$

with the following rows:

1. *V is the consumed* $\mathsf{VecSpace}^{\uparrow}$ *source row, displaying the ambient vector carrier, scalar action, and finite list of candidate generators.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *coordinate row whose columns display the chosen finite generator table and the row operations visible to downstream consumers.*
3. *D is the* $\mathsf{Determinant}^{\uparrow}$ *minor ledger, recording the rank-window minors used to certify the displayed dimension and nonzero Plucker coordinates.*
4. *E is the* $\mathsf{ExteriorAlgebra}^{\uparrow}$ *wedge row, carrying the alternating product of the displayed generator rows and the induced coordinate table.*
5. *Q is the* $\mathsf{ProjectiveSpace}^{\uparrow}$ *homogeneous-coordinate row for the wedge coordinate, read only after V, M, D, E are visible.*
6. *H records componentwise* hsame *transport for V, M, D, E, Q and for the package and local naming rows.*
7. *C records the displayed* Cont *replay routes by which consumers read the vector row, matrix table, determinant ledger, exterior coordinate, and projective endpoint in that order.*
8. *P is the* Pkg *provenance over* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{Determinant}^{\uparrow}$*,* $\mathsf{ExteriorAlgebra}^{\uparrow}$*,* $\mathsf{ProjectiveSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{Grassmannian}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected basis outside the displayed table, quotient subspace equality, a host projective variety, Schubert stratification, moduli functor, or scheme-theoretic Plucker ideal.

Theorem 8352.2 (Grassmannian NameCert obligations)**.** *For the carrier of Definition 8352.1, the local* $\mathsf{NameCert}_{\mathsf{Grassmannian}^{\uparrow}}$ *surface consists exactly of carrier admission for* $G = (V, M, D, E, Q, H, C, P, N)$*, vector-source exposure through* V*, matrix-coordinate exposure through* M*, determinant-rank exposure through* D*, exterior-coordinate exposure through* E*, projective-coordinate exposure through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Any accepted* $\mathsf{Grassmannian}^{\uparrow}$ *consumer must read those rows before it may cite a subspace, rank, Plucker coordinate, or homogeneous endpoint.*

Proof. Project the carrier rows. The only admitted packet is $G = (V, M, D, E, Q, H, C, P, N)$, so the local certificate can expose no data beyond those displayed coordinates.

The algebraic source is consumed in order. The vector-space row V supplies the ambient finite vector operations, the matrix row M displays the generator table, and the determinant row D certifies the rank-window minors visible in that table.

The geometric endpoint is then read through E and Q. The exterior row E turns the generator table into an alternating coordinate, and the projective row Q names its homogeneous class only after the preceding rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no slot for quotient subspace equality, a host projective variety, Schubert data, a moduli functor, or a scheme-theoretic Plucker ideal, none of those objects is exported by the NameCert surface. □

Theorem 8352.3 (Grassmannian Plucker ledger)**.** *Every Plucker-facing read of an accepted* $\mathsf{Grassmannian}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{VecSpace}^{\uparrow},\ \mathsf{Mat}^{\uparrow},\ \mathsf{Determinant}^{\uparrow},\ \mathsf{ExteriorAlgebra}^{\uparrow},\ \mathsf{ProjectiveSpace}^{\uparrow},\ H,\ C,\ P,\ N.$$

The determinant minors and alternating wedge coordinate are ledger rows of the same carrier $G = (V, M, D, E, Q, H, C, P, N)$*, and the projective endpoint* Q *cannot be consumed as an independent projective variety, quotient-subspace classifier, Schubert cell, moduli point, or hidden comparison row.*

Proof. Start with Definition 8352.2. It forces every accepted read to expose the vector source, matrix table, determinant ledger, exterior coordinate, projective endpoint, and structural rows of one packet.

Read the Plucker coordinate through the determinant and exterior rows. The row D supplies exactly the visible minors of M, and the row E packages the same data as an alternating coordinate rather than as a separate ambient construction.

Now consume the projective endpoint. The row Q is available only after V, M, D, E have been projected, while H, C, P, N keep transport, replay, provenance, and local naming tied to the same finite carrier.

Thus the Plucker ledger is a row-exhaustion statement for the displayed packet. No independent projective variety, quotient-subspace equality, Schubert cell, moduli point, or hidden comparison row is present in G, so no consumer can read one from the accepted certificate. □

Falsifiable prediction: An accepted $\mathsf{Grassmannian}^{\uparrow}$ consumer that reads a Plucker coordinate or homogeneous endpoint without displaying $V, M, D, E, Q, H, C, P, N$ refutes the carrier surface.

Independence witness: vecspace, matrix, determinant, exterior-algebra, projectivespace: no listed sibling is bijective to $\mathsf{Grassmannian}^{\uparrow}$ because this packet jointly binds a vector-space source row, finite generator matrix, determinant minor ledger, exterior wedge coordinate, projective homogeneous endpoint, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{Grassmannian}^{\uparrow}}$.

Constructive story: A $\mathsf{Grassmannian}^{\uparrow}$ packet is generated from VecSpace rows, Matrix generator rows, Determinant minor rows, ExteriorAlgebra wedge rows, a ProjectiveSpace homogeneous-coordinate row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Grassmannian}^{\uparrow}}$, seed)

Scope closed: Definition 8352.1, Definition 8352.2, and Definition 8352.3 seed the finite Grassmannian NameCert surface over displayed linear-algebra and Plucker-coordinate rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Grassmannian}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, Schubert calculus, moduli interpretation, scheme-theoretic Plucker ideals, quotient subspace equality, or a host projective variety.

Upgrade path: Obligation closure requires checked carrier admission, vector-source exposure, matrix-coordinate exposure, determinant minor exposure, exterior-coordinate exposure, projective-coordinate exposure, transport, replay, provenance, and local naming for BEDC.Derived.GrassmannianUp.

Chapter 8353

A Concrete Naming Certificate for $\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}$

The $\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}$ packet realizes the compact-cover route in chapter 9237 as finite overlap data before a uniform-modulus consumer reads a tolerance-only handoff. It packages compact finite-cover rows, pointwise local-modulus rows, overlap compatibility, rational lower-bound data, metric-triangle rows, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff as one inspectable NameCert carrier.

Definition 8353.1 (Compact-uniform local-modulus nerve carrier)**.** *A* $\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, O, Q, T, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite cover-nerve row, recording the* ProbeBundle *cover names, center rows, membership rows, and finite incidence data.*
2. *L is the local-modulus row, recording pointwise modulus data attached to each bundled cover center.*
3. *O is the overlap compatibility row, recording the common-radius and common-tolerance checks on nonempty cover intersections.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound row, recording the finite rational fold that selects a positive bound below the displayed local radii.*
5. *T is the metric-triangle row, recording source and target triangle estimates used to move from local center data to neighboring points.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exposed after B, L, O, Q, T have been read.*
7. *H records componentwise* hsame *transport for cover, local-modulus, overlap, lower-bound, triangle, uniform-modulus, provenance, and naming rows.*
8. *C records displayed* Cont *replay through cover membership, local modulus, overlap compatibility, rational lower-bound fold, metric-triangle estimate, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}}$ *row.*
The carrier has no coordinate for an external cover choice, a host minimum, a global modulus selected outside the finite nerve, or a completed Heine–Cantor theorem.

Theorem 8353.2 (CompactUniformLocalModulusNerve NameCert obligations)**.** *For the carrier of Definition 8353.1, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (B, L, O, Q, T, U, H, C, P, N),$$

finite cover-nerve display through B, local-modulus display through L, overlap compatibility through O, $\mathsf{Rat}^{\uparrow}$ *lower-bound display through Q, metric-triangle display through T,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier rows. The cover nerve, local modulus, overlap compatibility, rational lower bound, triangle estimate, and uniform-modulus handoff are displayed before the structural rows are read.

The overlap row O binds local modulus data across intersecting cover entries. The lower-bound row Q is therefore a finite rational fold over displayed rows, and the metric-triangle row T is the only route from local center estimates to arbitrary nearby source points.

Finally, H, C, P, N preserve, replay, source, and name this same packet. They do not add an external cover selector or host minimum, so the listed rows exhaust the local NameCert surface. □

Theorem 8353.3 (CompactUniformLocalModulusNerve uniform-modulus handoff)**.** *Every accepted* $\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}$ *packet exposes its* $\mathsf{UniformModulus}^{\uparrow}$ *handoff only through the finite route*

$$B, L, O, Q, T, U, H, C, P, N.$$

A compact-uniform-continuity consumer may read the handoff only after cover membership, local pointwise moduli, overlap compatibility, rational lower-bound data, and metric-triangle estimates have been displayed.

Proof. Apply the NameCert obligations of Definition 8353.2. They place the handoff row U downstream of B, L, O, Q, T.

The finite cover row B supplies the ProbeBundle nerve, while L and O attach local moduli and overlap compatibility to the same finite incidence data. The rational lower-bound row Q reads only these displayed radii, and the metric-triangle row T is the displayed passage from local estimates to the uniform read.

Transport and replay preserve the same route through H and C. Hence any consumer that reads U without the prior rows leaves the accepted carrier. □

Theorem 8353.4 (CompactUniformLocalModulusNerve sibling route)**.** *Every accepted* CompactUniformLocalMod
packet consumes the finite-cover nerve surface of chapter 7597 or the compact-cover nerve surface of chapter 6263 before it exports the local-modulus nerve. Its downstream handoff then passes through the $\mathsf{UniformModulus}^{\uparrow}$ *sibling of chapter 369; the carrier has no route that skips the cover incidence rows, overlap compatibility, rational lower-bound fold, or metric-triangle row.*

Proof. Read the carrier rows in Definition 8353.1. The cover-nerve coordinate B is the only source for cover membership, center data, and finite incidence.

The uniform-modulus handoff theorem then places U after B, L, O, Q, T. Thus the accepted packet can export the local-modulus nerve only through the finite-cover or compact-cover sibling surface and then through the displayed $\mathsf{UniformModulus}^{\uparrow}$ handoff.

The structural rows H, C, P, N preserve this same route. They do not introduce an external cover selector, a detached host minimum, or a second uniform-modulus output. □

Falsifiable prediction: Any compact-uniform consumer that cites a UniformModulusUp handoff while omitting the displayed overlap compatibility and RatUp lower-bound rows is rejected by the local classifier.

Independence witness: Nearest siblings: compactuniformcontinuity, compactcovernerve, finitecovernerve, finitecovermeshchain, compactuniformmodulus. The carrier is not bijective to any sibling because it binds local modulus rows and overlap compatibility before the uniform-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}$ packet is generated from finite cover-nerve, local-modulus, overlap compatibility, rational lower-bound, metric-triangle, uniform-modulus handoff, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter records the finite carrier and local NameCert surface for the compact-cover local-modulus nerve before uniform-modulus consumers read it.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformLocalModulusNerve}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CompactUniformLocalModulusNerveUp`

Bridge status: none

Origin: human

Not claimed: It does not close full compact uniform continuity, external cover choice, host finite minimum selection, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, overlap compatibility, rational lower-bound replay, metric-triangle replay, and uniform-modulus handoff for the same finite packet.

Lean4 (statement) `BEDC.Derived.CompactUniformLocalModulusNerveUp`

Chapter 8354

A Concrete Naming Certificate for $\mathsf{LocatedCompactMetric}^{\uparrow}$

Following Bishop and Bridges constructive metric compactness, the $\mathsf{LocatedCompactMetric}^{\uparrow}$ packet records a compact metric source together with displayed located distance data and the finite-net readback route used by L10 Real and Completion consumers. It sits beside the metric-space source of chapter 4230, the compact-metric certificate of chapter 129, the finite epsilon-net route of chapter 1767, the complete-metric handoff of chapter 128, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite located compact metric surface only; it does not assert open-cover compactness, sequential compactness, countable choice, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8354.1 (Located compact metric carrier)**.** *A* $\mathsf{LocatedCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, D, K, F, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, naming the displayed metric carrier and metric-law ledger inspected by every distance request.*
2. *D is the located distance witness row, recording finite lower and upper distance windows for each displayed point-to-source query.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, exposing compact metric data only through displayed finite-net evidence.*
4. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *finite-net row, carrying center lists, tolerance labels, and coverage witnesses at inspected scales.*
5. *T is the* $\mathsf{CompleteMetric}^{\uparrow}$ *route, naming the completion-facing Cauchy handoff without selecting a quotient representative or host limit.*
6. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to measure radii, slack bounds, and finite comparison budgets.*
7. *E is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal row, exposing regular rational windows before any real-valued distance or compactness consumer can inspect the packet.*

8. *H records componentwise* hsame *transport for the metric source, located distance witness, compact metric row, finite epsilon-net row, complete-metric route, dyadic ledger, readback seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a located compact metric request through M, D, K, F, T, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, D, K, F, T, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, sequential compactness, countable choice, quotient completion equality, selected limits, host metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8354.2 (Located compact metric NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactMetric}^{\uparrow}$ *carrier*

$$L = (M, D, K, F, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}$ *surface consists exactly of MetricSpace source admission, located distance witness exposure, CompactMetric finite compactness data, FiniteEpsilonNet center and coverage rows, CompleteMetric completion-facing routing, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8354.1. The source metric row is M, the located distance row is D, and compactness is visible only through the compact metric row K together with the finite epsilon-net row F.

The completion-facing side is then forced through T, R, E. The complete-metric row supplies the Cauchy handoff route, the dyadic ledger records the inspected tolerances, and the readback seal exposes regular rational data before the terminal real-name row is consumed.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since no listed row contains an open-cover theorem, a sequential-compactness selector, countable choice, quotient completion equality, selected limits, host metric equality, or ambient completeness, the displayed rows exhaust the NameCert obligation surface. □

Theorem 8354.3 (Located compact metric finite-net completion route)**.** *Every completion-facing or real-facing consumer of an accepted* $\mathsf{LocatedCompactMetric}^{\uparrow}$ *packet must first read the route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the same local $\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}$ *carrier rows. The route exports finite centers, coverage witnesses, located distance windows, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, sequential compactness, countable choice, quotient completion, selected limit, or ambient completeness principle.*

Proof. Begin with the metric and compact rows. The metric source M gives the distance ledger, while K and F expose compactness only as displayed finite epsilon-net data.

Next read the located and completion rows. The located distance witness D answers finite distance-window requests, and the complete-metric row T names the handoff used by completion consumers without choosing a quotient representative or host limit.

Finally pass through the dyadic ledger and readback seal. Rows R and E keep the tolerance and real-facing data inside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and H, C, P, N replay the same

finite route. Thus the consumer receives exactly the listed finite surface and none of the refused endpoints. □

Falsifiable prediction: Any accepted $\mathsf{LocatedCompactMetric}^{\uparrow}$ read that exposes compactness or completion-facing data without displaying MetricSpace source admission, located distance witness data, CompactMetric finite compactness rows, FiniteEpsilonNet coverage, CompleteMetric routing, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert data refutes the carrier.

Independence witness: metricspace, compactmetric, finite-epsilon-net, completemetric, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{LocatedCompactMetric}^{\uparrow}$ because this packet jointly binds metric source data, located distance windows, compact finite-net evidence, completion routing, dyadic tolerance accounting, regular readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactMetric}^{\uparrow}$ packet is generated from a MetricSpace source row, located distance witness, CompactMetric finite compactness data, FiniteEpsilonNet coverage rows, CompleteMetric routing, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8354.1, Definition 8354.2, and Definition 8354.3 seed the located compact metric route over metric source, located distance, compact finite-net, completion, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, countable choice, quotient completion, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located distance exactness, finite-net coverage, completion routing, dyadic tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactMetricUp.

Chapter 8355

A Concrete Naming Certificate for BolzanoWeierstrassLocatedSequence$^{\uparrow}$

Bolzano–Weierstrass and Bishop-style located compactness give the BolzanoWeierstrassLocatedSequence$^{\uparrow}$ packet its finite BEDC shape: a bounded located real sequence is inspected through stream windows, regular rational readback, compact interval data, and a monotone-subsequence theorem surface before any cluster-window seal is consumed. It sits after the Real$^{\uparrow}$ carrier of chapter 38, the RegSeqRat$^{\uparrow}$ route of chapter 301, the finite stream-window surface of chapter 126, the selector theorem of chapter 1997, and the compact interval data of chapter 1250 and chapter 3969. The packet records a located finite-window substitute for sequential compactness; it does not assert open-cover compactness, countable choice, a selected host subsequence, quotient stream equality, or ambient Real$^{\uparrow}$ completeness.

Definition 8355.1 (Bolzano-Weierstrass located-sequence carrier)**.** *A* BolzanoWeierstrassLocatedSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, L, K, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded located source-sequence row, read only through finite* StreamName$^{\uparrow}$ *windows and displayed lower and upper interval bounds.*
2. *L is the located interval-membership row that records rational apartness windows for the source values.*
3. *K is the compact interval data row, supplied by finite closed-bounded interval nets and closed-bounded compactness certificates.*
4. *M is the* MonotoneSubsequenceTheorem$^{\uparrow}$ *row that names the finite selector-theorem surface available to the bounded source.*
5. *W is the selected finite subsequence-window ledger opened by S, L, K, M.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row for the located cluster-window read.*
8. *H records componentwise* hsame *transport for $S, L, K, M, W, R, E, C, P, N$.*
9. *C records displayed* Cont *replay from located source windows through compact interval data and monotone-subsequence windows to regular readback and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MonotoneSubsequenceTheorem}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for open-cover compactness, selected infinite subsequences, countable choice, quotient stream equality, trichotomy of host reals, or ambient completeness.

Theorem 8355.2 (Bolzano-Weierstrass located-sequence NameCert obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}$ *carrier*

$$B = (S, L, K, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded located source-window exposure through S, L, compact interval data exposure through K, selector-theorem exposure through M, finite selected-window exposure through W, regular rational readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The source side is S, L, K: finite stream windows, located interval membership, and compact interval data are the only rows available before a subsequence window can be read.

The selector-theorem row M is then consumed as a finite BEDC surface. It supplies monotone selected-window data for the bounded source without adding a choice-selected infinite subsequence.

The readback rows are W, R, E. The selected windows W determine the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is visible only after that readback.

Finally H, C, P, N preserve, replay, source, and name the same rows. Since no excluded compactness, choice, quotient, trichotomy, or ambient-completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8355.3 (Bolzano-Weierstrass located subsequence window)**.** *Every cluster-window read exported by an accepted* $\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{MonotoneSubsequenceTheorem}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the compact interval data row K and located membership row L. Thus a downstream Real or Completion consumer receives only bounded located source windows, compact interval data, a finite monotone-subsequence theorem row, selected windows, regular readback, a Real seal, transport, replay, provenance, and local naming.

Proof. Start with the accepted carrier and read the source windows S through the located row L. The compact interval data K supplies the finite bounded interval surface in which those windows are inspected.

Next use the selector-theorem row M. Its output is the finite selected-window ledger W, not a host-level infinite subsequence.

The selected windows pass to R and then to E. This is exactly the regular rational readback and terminal Real seal route consumed by downstream Real and Completion readers.

The structural rows H, C, P, N keep the same finite route under transport and replay. No open-cover compactness theorem, selected subsequence functional, quotient stream equality, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}$ packet is generated from bounded located StreamName windows, located interval membership, compact interval data, a MonotoneSubsequenceTheorem row, selected finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8355.1, Definition 8355.2, and Definition 8355.3 seed the finite located sequence route from compact interval data to regular cluster-window readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassLocatedSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected infinite subsequences, countable choice, quotient stream equality, trichotomy of host reals, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source-window exposure, compact interval data exposure, monotone-subsequence theorem routing, selected-window readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BolzanoWeierstrassLocatedSequenceUp.

Chapter 8356

A Concrete Naming Certificate for CompactIntervalModulusExtractor$^{\uparrow}$

Following Bishop and Bridges on compact intervals and uniform continuity, the CompactIntervalModulusExtractor$^{\uparrow}$ packet records the finite extraction surface promised by chapter 9237: compact interval probes, rational lower-bound folds, and a uniform-modulus handoff are carried as displayed NameCert rows. It sits beside the compact interval modulus route of chapter 5937, the bridge packet of chapter 3868, the finite interval net of chapter 1250, the compact source rows of chapter 129, and the output surface of chapter 369. The packet names only finite extraction data; it does not assert open-cover compactness, a selected subcover functional, host semantic continuity, quotient real equality, or ambient Real$^{\uparrow}$ completeness.

Definition 8356.1 (Compact interval modulus extractor carrier)**.** *A* CompactIntervalModulusExtractor$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, K, Q, F, R, U, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, read through closed bounded interval endpoint and finite-net data.*
2. *K is the* CompactMetric$^{\uparrow}$ *or interval-compactness row that supplies the finite* ProbeBundle *cover.*
3. *Q is the tolerance-query row whose requested precision is displayed before any radius fold is read.*
4. *F is the finite lower-bound fold row, recording rational radius comparisons over the displayed probe bundle.*
5. *R is the regular rational readback row attached to the selected radius and interval-window data.*
6. *U is the* UniformModulus$^{\uparrow}$ *handoff row exported after I, K, Q, F, R have been displayed.*
7. *S records the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *seal rows used by interval endpoints, probe centers, and modulus readback.*
8. *H records componentwise* hsame *transport for $I, K, Q, F, R, U, S, C, P, N$.*
9. *C records displayed* Cont *replay from compact interval probes through the rational lower-bound fold to the uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$, $\mathsf{CompactIntervalModulus}^{\uparrow}$, $\mathsf{CompactIntervalMod}$ $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for arbitrary open covers, selected subcovers, host continuity theorems, quotient real equality, countable choice, or ambient completeness.

Theorem 8356.2 (Compact interval modulus extractor NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$ *carrier*

$$X = (I, K, Q, F, R, U, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusExtractor}^{\uparrow}}$ *surface consists exactly of carrier admission, compact interval source exposure through I, finite probe-bundle exposure through K, tolerance-query exposure through Q, rational lower-bound fold exposure through F, regular readback through R,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through U, stream, regular, and Real seal exposure through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The source side is I, K, Q: the interval, compact finite probe bundle, and requested precision must all be visible before any extracted radius is read.

The finite lower-bound fold is F. It ranges only over the displayed probe rows and rational comparisons, so it is not an open-cover compactness theorem or a selected subcover functional.

The output side is R, U, S. The regular readback records the extracted rational data, the $\mathsf{UniformModulus}^{\uparrow}$ row is downstream of that readback, and S carries the stream, regular, and Real seal rows used by the handoff.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no host continuity, quotient, choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8356.3 (Compact interval modulus extractor vision handoff)**.** *Every compact-uniform-continuity read of an accepted* $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ProbeBundle} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The route realizes the compact interval branch of chapter 9237 as finite ProbeBundle cover rows, rational lower-bound fold data, regular readback, Real sealing, and UniformModulus handoff rows.

Proof. Begin with the compact interval source and finite probe bundle rows I, K. These rows supply the only cover data visible to the extractor.

Read the tolerance query Q and the lower-bound fold F. The fold is finite and rational, so every selected radius remains attached to a displayed probe and comparison row.

The regular readback and seal rows R, S then prepare the uniform-modulus handoff U. This is the precise finite route named by the compact-uniform-continuity vision: probe coverage, lower-bound fold, readback, Real seal, and uniform modulus.

Transport, replay, provenance, and local naming through H, C, P, N keep the route inside the carrier. No open-cover compactness, selected subcover, semantic continuity theorem, quotient real equality, countable choice, or ambient completeness is exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$ uniform-modulus read factors through compact interval source rows, a finite ProbeBundle cover, a tolerance query, rational lower-bound fold data, RegSeqRat readback, RealUp sealing, UniformModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: compact-interval-modulus, compact-interval-modulus-bridge, closed-boundedintervalnet, compactmetric, uniformmodulus: $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the compact interval source, probe-bundle cover, tolerance query, rational lower-bound fold, regular readback, Real seal, UniformModulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalModulusExtractor}^{\uparrow}$ packet is generated from compact interval source rows, CompactMetric finite ProbeBundle cover data, tolerance queries, rational lower-bound folds, RegSeqRat readback, UniformModulus handoff, StreamName and RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8356.1, Definition 8356.2, and Definition 8356.3 seed the compact interval modulus extraction route from finite ProbeBundle covers to UniformModulus handoff data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalModulusExtractor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host semantic continuity, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval source exposure, finite ProbeBundle coverage, rational lower-bound fold exactness, RegSeqRat readback, RealUp sealing, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalModulusExtractorUp.

Chapter 8357

A Concrete Naming Certificate for $\mathsf{RealSeriesComparison}^{\uparrow}$

Following Bishop–Bridges comparison-test practice, the $\mathsf{RealSeriesComparison}^{\uparrow}$ packet records a finite real-series comparison route that may read signed source terms only through displayed windows and bounded domination evidence. It consumes the positive comparison surface of chapter 7665, the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the finite stream windows of chapter 126. The packet names a comparison-test certificate for later $\mathsf{Real}^{\uparrow}$ series-tail estimates; it is not an ambient convergence theorem, a host infinite-sum construction, quotient equality of term streams, or a selected limit.

Definition 8357.1 (Real series comparison carrier)**.** *A* $\mathsf{RealSeriesComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, B, P, W, D, Q, E, H, R, G, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RealSeries}^{\uparrow}$ *row whose finite term windows are being compared.*
2. *B is the bounding* $\mathsf{RealSeries}^{\uparrow}$ *row read through the positive comparison packet.*
3. *P is the* $\mathsf{PositiveSeriesComparison}^{\uparrow}$ *witness row, exposing the nonnegative domination surface already named by chapter 7665.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite indices, signs, tail requests, and comparison requests under inspection.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording absolute-value bounds, domination checks, and requested error budgets for the displayed windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the compared windows are routed toward regular rational data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after S, B, P, W, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the source series, bounding series, positive comparison witness, windows, dyadic ledger, readback, terminal seal, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from a comparison request through $S, B, P, W, D, Q, E, H, G, N$.*

10. *G is the* Pkg *provenance over* $\mathsf{RealSeriesComparison}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{PositiveSeriesComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeriesComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, P, W, D, Q, E, H, R, G, N$. It has no coordinate for ambient convergence, host infinite sums, quotient equality of term streams, selected limits, countable choice, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8357.2 (Real series comparison obligation surface)**.** *For every accepted* $\mathsf{RealSeriesComparison}^{\uparrow}$ *carrier*

$$C = (S, B, P, W, D, Q, E, H, R, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSeriesComparison}^{\uparrow}}$ *surface consists exactly of source-series exposure, bounding-series exposure, positive comparison witness exposure, finite stream-window exposure, dyadic tolerance-ledger exposure, regular rational readback, terminal real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8357.1. The rows $S, B, P, W, D, Q, E, H, R, G, N$ are the complete finite packet, so carrier admission and classifier exactness are componentwise projections.

Read the comparison rows next. The source row S and bounding row B are inspected only through the finite window row W, while P imports the already displayed positive comparison witness and D records the dyadic domination ledger for those windows.

Finally inspect Q, E, H, R, G, N. These rows read back, seal, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host sums, selected limits, quotient equality, or ambient convergence, the obligation surface is exactly the stated packet. □

Theorem 8357.3 (Real series comparison non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RealSeriesComparison}^{\uparrow}$ *carrier reaches the terminal seal only through*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{PositiveSeriesComparison}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a comparison-test read may export finite tail-estimate evidence, but it cannot export ambient convergence, a host infinite sum, quotient equality of term streams, selected limits, countable choice, or detached real-name completeness.

Proof. Start with the source and bounding series rows. They expose only finite $\mathsf{RealSeries}^{\uparrow}$ observations, and the bounding row is consumable as comparison evidence only through the $\mathsf{PositiveSeriesComparison}^{\uparrow}$ witness row.

Next read the window and tolerance rows. The $\mathsf{StreamName}^{\uparrow}$ row supplies the inspected indices and requested tail windows, while the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the finite domination and error-budget checks used by the comparison route.

The readback and terminal rows Q, E are downstream of those finite rows. Componentwise transport, displayed replay, provenance, and local naming preserve the same packet, so no consumer can escape to an unlisted convergence principle or selected limit. □

Falsifiable prediction: Every accepted $\mathsf{RealSeriesComparison}^{\uparrow}$ read exposes source and bounding $\mathsf{RealSeries}^{\uparrow}$ rows, a $\mathsf{PositiveSeriesComparison}^{\uparrow}$ witness, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a comparison conclusion is consumed.

Independence witness: positive-series-comparison, realseries, streamname, dyadicratcore, regseqrat, real: $\mathsf{RealSeriesComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries

signed source windows, bounding series windows, positive comparison evidence, dyadic domination ledgers, regular readback, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeriesComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSeriesComparison}^{\uparrow}$ packet is generated from source and bounding RealSeries rows, a PositiveSeriesComparison witness row, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSeriesComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8357.1, Definition 8357.2, and Definition 8357.3 seed the finite real-series comparison route from source and bounding series windows through positive comparison evidence, dyadic tolerance accounting, regular readback, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSeriesComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim ambient convergence, host sums, quotient equality of term streams, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-window admission, dyadic domination exactness, positive-comparison witness routing, readback stability, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RealSeriesComparisonUp.

Chapter 8358

A Concrete Naming Certificate for $\mathsf{StirlingFactorialBudget}^{\uparrow}$

Following Stirling and Robbins, the $\mathsf{StirlingFactorialBudget}^{\uparrow}$ packet records the finite denominator and growth-control data needed when factorial estimates feed constructive real power series. It sits between the Gamma normalization route of chapter 75, the exponential and trigonometric series consumers of chapter 1816 and chapter 6126, and the L10 real substrate carried by chapter 291, chapter 126, chapter 301, and chapter 38. The carrier names a finite factorial-growth budget, not an asymptotic oracle, selected limit, or ambient analytic completeness theorem.

Definition 8358.1 (Stirling factorial-budget carrier)**.** *A* $\mathsf{StirlingFactorialBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, A, B, D, W, R, E, H, C, P, N)$$

with the following rows:

1. *K is the factorial index window recording the finite interval of n-values inspected by the downstream Gamma or power-series consumer.*
2. *A is the lower Stirling-style comparison ledger, storing the finite rational inequalities used to dominate reciprocal factorial denominators from below.*
3. *B is the upper Robbins-style slack ledger, storing the finite rational inequalities used to bound factorial growth from above.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator ledger carrying common dyadic refinements, radius comparisons, and slack transport for A and B.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window service naming which coefficient windows are read by the consumer.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same denominator and coefficient windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row consumed by the Gamma, exponential, or sine series route only after K, A, B, D, W, R have been displayed.*
8. *H stores componentwise* hsame *transport for the index window, comparison ledgers, dyadic denominator row, stream window, regular-rational readback, real handoff, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a finite precision request through the same factorial-budget route.*

10. *P is the* Pkg *provenance over* StirlingFactorialBudget$^{\uparrow}$*, the Gamma and real-series consumers,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* NameCert$_{\mathsf{StirlingFactorialBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a real-valued asymptotic expansion, a hidden analytic limit, selected denominator normalization, or ambient completeness.

Theorem 8358.2 (Stirling factorial-budget NameCert obligations)**.** *For every accepted* StirlingFactorialBudget$^{\uparrow}$ *carrier*

$$F = (K, A, B, D, W, R, E, H, C, P, N),$$

the local NameCert$_{\mathsf{StirlingFactorialBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, factorial index-window exposure, lower and upper Stirling-style comparison exposure, dyadic denominator refinement, stream-window exposure, regular-rational readback, Real-facing handoff, componentwise transport, displayed replay, provenance, and local naming. Any accepted consumer that uses a factorial denominator estimate must expose $K, A, B, D, W, R, E, H, C, P, N$ before using the estimate inside Gamma, exponential, or sine coefficient control.*

Proof. Project the carrier of Definition 8358.1. The finite rows $K, A, B, D, W, R, E, H, C, P, N$ are the only rows available to the classifier, so every budget read starts from the displayed index window and the two rational comparison ledgers.

Read the denominator surface next. The DyadicRatCore$^{\uparrow}$ ledger D carries the common denominator refinement needed by both Stirling-style bounds, while W and R attach the same finite budget to stream windows and regular-rational readback.

Finally consume the Real-facing and structural rows. The row E is available only after the comparison and readback rows have been displayed, and H, C, P, N preserve, replay, record, and name the same finite packet. Thus the NameCert surface exports finite factorial-budget data rather than an analytic asymptotic principle. □

Theorem 8358.3 (Stirling factorial-budget Gamma and power-series handoff)**.** *Every Gamma, real-exponential, or real-sine coefficient consumer that reads a* StirlingFactorialBudget$^{\uparrow}$ *packet factors through the finite route*

$$K, A, B, D, W, R, E, H, C, P, N.$$

The Gamma route reads K, A, B, D as finite factorial-growth control before passing to its Real-facing row, while the exponential and sine series routes read W, R, E as the coefficient-window handoff. None of these consumers receives a selected asymptotic expansion, quotient real equality, host convergence theorem, or ambient analytic completeness principle from the packet.

Proof. Start with the Gamma-side read. The factorial index window K and the comparison ledgers A, B are finite rational data, and the dyadic denominator ledger D is the only denominator refinement exported to the Gamma consumer.

Now read the power-series side. The coefficient windows W and regular-rational readback R attach the same budget to the real exponential and real sine consumers before the Real-facing row E is consumed.

Transport and replay close the handoff. The rows H, C, P, N keep the Gamma and power-series consumers over the same finite packet, so the handoff is a BEDC NameCert route and not a hidden appeal to Stirling asymptotics as an external analytic object. □

Falsifiable prediction: Every accepted $\mathsf{StirlingFactorialBudget}^{\uparrow}$ consumer that uses a factorial denominator estimate without exposing index-window, lower-bound, upper-bound, dyadic-denominator, stream-window, RegSeqRat, Real handoff, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: gamma-function, real-exponential, real-sine, dyadicratcore, stream-name, regseqrat, real: no listed sibling jointly carries finite factorial index windows, two-sided Stirling-style rational comparison ledgers, dyadic denominator refinement, coefficient-window readback, Real-facing handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StirlingFactorialBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{StirlingFactorialBudget}^{\uparrow}$ packet is generated from a finite factorial index window, two-sided rational Stirling-style comparison ledgers, dyadic denominator refinements, coefficient windows, regular-rational readback, a Real-facing handoff, transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StirlingFactorialBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8358.1, Definition 8358.2, and Definition 8358.3 seed the finite factorial-growth budget route for Gamma and real power-series consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StirlingFactorialBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Stirling's formula as a real asymptotic theorem, Gamma normalization, power-series convergence, quotient real equality, selected denominator normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-sided factorial comparison exactness, dyadic denominator transport, coefficient-window readback, Real-facing handoff, replay, provenance, and local naming for BEDC.Derived.StirlingFactorialBudgetUp.

Chapter 8359

A Concrete Naming Certificate for $\mathsf{CompactIntervalSupremum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactIntervalSupremum}^{\uparrow}$ packet records the finite-net supremum surface for a located compact real interval. It sits over the terminal real-name seal of chapter 38, the dyadic tolerance ledger of chapter 291, the located endpoint rows of chapter 394, the compact metric finite-net source of chapter 129, and the total-bounded finite-net surface of chapter 127. The packet names finite rows for inspected interval endpoints, finite-net coverage, dyadic tolerances, candidate upper bounds, and local supremum readback; it does not assert open-cover compactness, a selected global maximizer, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8359.1 (Compact interval supremum carrier)**.** *A* $\mathsf{CompactIntervalSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, I, K, T, D, U, M, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row for the displayed lower and upper interval endpoints.*
2. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row recording located endpoint comparisons and interval membership windows.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-source row read only through finite metric coverage data.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, displaying the inspected centers and coverage witnesses at the requested scale.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for mesh size, slack, and upper-bound comparison budgets.*
6. *U records the finite family of candidate upper-bound rows inspected over the displayed net.*
7. *M records the finite maximum-read row over those candidate bounds, as rational readback data rather than as a host supremum operator.*
8. *R is the local supremum-candidate row exported after E, I, K, T, D, U, M have all been displayed.*
9. *H records componentwise* hsame *transport for the endpoint, interval, compact, net, tolerance, candidate, readback, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from located interval data through finite-net inspection to the supremum-candidate row.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalSupremum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, I, K, T, D, U, M, R, H, C, P, N$. It has no coordinate for open-cover compactness, ambient completeness, quotient real equality, a selected maximizer, trichotomy, or an unrecorded comparison outside the displayed finite-net packet.

Theorem 8359.2 (Compact interval supremum NameCert obligations)**.** *For every accepted* CompactIntervalSuprem *carrier*

$$S = (E, I, K, T, D, U, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalSupremum}^{\uparrow}}$ *surface consists exactly of RealUp endpoint exposure, located interval exposure, compact finite-net source exposure, total-bounded net coverage, dyadic tolerance accounting, finite candidate upper-bound inspection, finite maximum readback, supremum-candidate export, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The endpoint and interval side is E, I: every candidate is read against displayed real endpoint seals and located interval rows before any upper-bound comparison is available.

The compact finite-net side is K, T, D. Compactness is consumed only through finite metric coverage and total-bounded net rows, while the dyadic ledger records the mesh and slack budgets used by the comparisons.

The supremum read is U, M, R. The candidate upper bounds are finite rows, the maximum row is a finite readback over those rows, and the exported supremum candidate is downstream of the same displayed data.

Finally H, C, P, N transport, replay, source, and name the packet. Since no carrier coordinate names open-cover compactness, ambient completeness, quotient equality, a selected maximizer, trichotomy, or a hidden comparison, the listed rows exhaust the local NameCert obligations. □

Theorem 8359.3 (Compact interval finite-net supremum read)**.** *Every accepted* $\mathsf{CompactIntervalSupremum}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{TotallyBounded}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow}$$

inside the same carrier $S = (E, I, K, T, D, U, M, R, H, C, P, N)$. A consumer may read the exported candidate R only after the endpoint seals, located interval rows, finite-net coverage, dyadic tolerance ledger, candidate upper-bound rows, and finite maximum readback have been displayed.

Proof. Begin with E and I. These rows bind the real endpoints and located interval comparisons that define the inspected compact interval surface.

Next read K, T, D. The compact metric and total-bounded rows display the finite net, and the dyadic ledger fixes the tolerance and slack used for each comparison.

Then consume U and M. Both are finite rows: U lists candidate upper-bound inspections over the displayed net, and M reads the finite maximum row attached to those candidates.

The replay, transport, provenance, and naming rows H, C, P, N keep the route inside the same BHist packet. Therefore the exported row R is a finite-net supremum candidate, not an open-cover

compactness theorem, selected maximizer, quotient real equality, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion. □

Falsifiable prediction: Every accepted $\mathsf{CompactIntervalSupremum}^{\uparrow}$ read exposes RealUp endpoint seals, LocatedInterval rows, CompactMetric finite-net source data, TotallyBounded coverage rows, DyadicRatCore tolerance accounting, finite candidate upper-bound rows, finite maximum readback, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before exporting a supremum candidate.

Independence witness: real, dyadicratcore, locatedinterval, compactmetric, totallybounded: $\mathsf{CompactIntervalSupremum}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds endpoint seals, located interval data, compact finite-net source rows, total-bounded coverage, dyadic tolerance budgets, candidate upper-bound inspection, finite maximum readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalSupremum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalSupremum}^{\uparrow}$ packet is generated from RealUp endpoint seals, LocatedInterval rows, CompactMetric finite-net source data, TotallyBounded coverage rows, DyadicRatCore tolerance accounting, finite candidate upper-bound inspection, finite maximum readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8359.1, Definition 8359.2, and Definition 8359.3 seed the compact interval finite-net supremum route over displayed endpoint, interval, compact, total-bounded, dyadic, candidate, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalSupremum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient RealUp completeness, quotient real equality, selected maximizers, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, located interval exposure, finite-net coverage, dyadic tolerance accounting, candidate upper-bound inspection, finite maximum readback, supremum-candidate export, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalSupremumUp.

Chapter 8360

A Concrete Naming Certificate for MetaCICResidualPeakScheduler$^{\uparrow}$

The MetaCICResidualPeakScheduler$^{\uparrow}$ packet realizes the residual-substitution boundary of chapter 9378 as finite local-peak scheduling data. It sits beside the critical-path carrier of chapter 765, the parallel-diamond frontier of chapter 4238, the candidate-normalization handoff of chapter 5547, the residual local-join packet of chapter 5126, and the audit packet of chapter 735. The scheduler names which residual peak is inspected, which residual rows remain visible, which candidate joins are budgeted, and which blocked edge is retained; it does not claim full MetaCIC confluence.

Definition 8360.1 (MetaCIC residual peak-scheduler carrier)**.** *A* MetaCICResidualPeakScheduler$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, R, J, B, T, U, H, C, Q, N)$$

with the following rows:

1. *P is the local peak row, naming the source term, the two outgoing parallel or residual reductions, and the syntactic redex frontier inspected by the scheduler.*
2. *R is the residual-row ledger, displaying the residual substitutions and branch-local redex descendants that remain visible after the peak is decomposed.*
3. *J is the candidate-join budget, recording the finite list of candidate common reducts and the bounded checks attached to them.*
4. *B is the blocked-edge row, retaining the residual-substitution compatibility or residual beta branch that has not been discharged.*
5. *T is the transport row from the sibling critical-path and parallel-diamond frontier packets.*
6. *U is the replay row from candidate normalization through the residual local-join packet and back to the confluence audit packet.*
7. *H stores componentwise* hsame *transport for the local peak, residual rows, candidate joins, blocked edge, transport, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for the finite scheduler route.*
9. *Q is the* Pkg *provenance over* MetaCICResidualPeakScheduler$^{\uparrow}$*, the MetaCIC critical path, the parallel-diamond frontier, candidate normalization, the residual local-join packet, the audit packet,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for an unconditional parallel diamond, a completed residual-substitution proof, a global Church–Rosser theorem, or a hidden proof search procedure.

Theorem 8360.2 (MetaCIC residual peak-scheduler NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ *carrier*

$$S = (P, R, J, B, T, U, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}}$ *surface consists exactly of local peak exposure, residual-row visibility, candidate-join budget exposure, blocked-edge retention, sibling transport, candidate-normalization replay, componentwise transport, displayed continuation replay, provenance, and local naming. Any accepted consumer that reaches a candidate join must expose $P, R, J, B, T, U, H, C, Q, N$ before treating the join as available to the confluence audit route.*

Proof. Project the carrier of Definition 8360.1. The local peak P, residual rows R, candidate budget J, and blocked edge B are explicit coordinates, so the scheduler cannot hide the residual-substitution boundary behind an unconditional diamond claim.

Read the sibling transport and replay rows. The row T imports only the finite critical-path and parallel-frontier data, while U replays candidate normalization through the residual local-join packet and back to the audit packet.

Finally read the structural rows. Componentwise transport H, displayed replay C, provenance Q, and local naming N preserve the same finite scheduler surface for downstream consumers, excluding completed confluence, hidden proof search, or a discharged residual beta branch. □

Theorem 8360.3 (MetaCIC residual peak-scheduler diamond handoff)**.** *Every conditional diamond or confluence-audit consumer of a* $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ *packet factors through the finite route*

$$P, R, J, B, T, U, H, C, Q, N.$$

The route may pass candidate joins to the parallel-diamond frontier and residual local-join packets, but it must retain the blocked-edge row B whenever residual-substitution compatibility or residual beta-branch compatibility has not been supplied. Therefore the handoff is a finite audit boundary, not a proof of the full parallel diamond.

Proof. Start with the local peak and residual rows. They determine which branch pair and which residual descendants are visible to the scheduler before any candidate common reduct is considered.

Use the candidate-join budget next. The row J supplies only bounded candidate joins, and the blocked-edge row B travels with the same packet whenever a compatibility edge remains unavailable.

The transport and replay rows complete the handoff. The rows T, U, H, C, Q, N connect the packet to the critical-path, parallel-frontier, candidate-normalization, residual-local-join, and audit-packet siblings while preserving the blocked edge. Thus downstream diamond consumers receive a scheduled finite boundary rather than an unconditional diamond theorem. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ consumer that reaches a candidate join without first exposing local peak, residual rows, scheduler budget, blocked edge, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: metaciccriticalpath, metacic-parallel-diamond-frontier, metacic-candidate-normalization-confluence-handoff, parallel-diamond-residual, metacic-confluence-audit-packet: no listed sibling jointly carries local peak scheduling, residual-row visibility, candidate-join budget, blocked-edge retention, transport, replay, provenance, and local naming.

The paired Lean surface for this packet must provide Nontrivial $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ and FieldFaithful $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ instances in a `TasteGate.lean` file, with an explicit witness for nontriviality and projection proofs for every displayed field.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}$ packet is generated from a local MetaCIC peak, visible residual rows, a finite candidate-join budget, a retained blocked edge, sibling transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8360.1, Definition 8360.2, and Definition 8360.3 seed the finite residual-peak scheduling route for the confluence audit boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualPeakScheduler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, full parallel diamond, arbitrary closed-term Church–Rosser, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked local-peak exposure, residual-row visibility, candidate-join budget exactness, blocked-edge retention, sibling transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualPeakSchedulerUp

Chapter 8361

A Concrete Naming Certificate for UniformConvergencePreservesContinuity$^{\uparrow}$

Following Bishop and Bridges and Kelley, the UniformConvergencePreservesContinuity$^{\uparrow}$ packet records the standard constructive route from uniformly convergent continuous real-valued functions to a continuous limit. It reads the finite sup-norm boundary of chapter 3807, the uniform-limit metric handoff of chapter 1764, the real-function uniform Cauchy schedule of chapter 1830, the continuous-map source of chapter 130, the compact metric source of chapter 129, and the terminal Real$^{\uparrow}$ seal of chapter 38. The packet names the finite continuity-preservation boundary; it does not choose a host limiting function, identify functions by quotient equality, or import ambient completeness outside the displayed rows.

Definition 8361.1 (Uniform convergence preserves continuity carrier)**.** *A* UniformConvergencePreservesContinuity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, A, S, K, F, G, R, W, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the* UniformLimitMetric$^{\uparrow}$ *row that receives a uniform convergence request.*
2. *A is the* RealFunctionUniformCauchy$^{\uparrow}$ *row recording the real-valued uniform Cauchy schedule for the approximating function rows.*
3. *S is the* ContinuousSupNorm$^{\uparrow}$ *sibling row that supplies the finite sup-norm comparison boundary.*
4. *K is the* CompactMetric$^{\uparrow}$ *source row over which all source-window reads are made.*
5. *F is the* ContinuousMap$^{\uparrow}$ *row for the approximating continuous maps over K.*
6. *G is the displayed limit-row request, storing only the finite continuity test to be replayed for the limit candidate.*
7. *R is the terminal* Real$^{\uparrow}$ *difference-and-triangle seal reached after the source, approximant, and limit rows have been displayed.*
8. *W is the finite window ledger, recording the source refinement, uniform tail bound, and pointwise comparison windows used in the continuity test.*
9. *M is the modulus-transfer row that transports the approximant continuity modulus through the uniform tail and Real seal.*

10. *H records componentwise* hsame *transport for $L, A, S, K, F, G, R, W, M, C, P, N$.*
11. *C records displayed* Cont *replay through the same rows.*
12. *P is the* Pkg *provenance over* $\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}$, $\mathsf{UniformLimitMetric}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a choice-selected limit, quotient equality of functions, open-cover compactness, host supremum selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.UniformConvergencePreservesContinuityUp`

Theorem 8361.2 (Uniform convergence preserves continuity NameCert obligations)**.** *For every accepted* $\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}$ *carrier* $U = (L, A, S, K, F, G, R, W, M, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}}$ *surface consists exactly of uniform-limit metric admission through L, real-function uniform Cauchy schedule exposure through A, continuous sup-norm boundary exposure through S, compact metric source exposure through K, approximating continuous-map exposure through F, displayed limit-row request through G, Real difference-and-triangle sealing through R, finite window exposure through W, modulus transfer through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8361.1. The tuple displays the uniform-limit row, uniform Cauchy schedule, sup-norm comparison row, compact source, approximating continuous-map row, limit-row request, Real seal, window ledger, modulus transfer, transport, replay, provenance, and local naming.

The rows K, F, W, M supply the only continuity information available to the packet. The rows A, S, L place that same information under a uniform tail comparison, so the continuity test for the limit is read through the displayed uniform-convergence boundary rather than through a host-selected limiting function.

The Real seal R is terminal. It records the triangle comparison after the approximant continuity row and the uniform tail row have both been displayed.

Finally, H, C, P, N preserve, replay, source, and name the same route. Since no carrier coordinate stores quotient equality of functions, a selected limit, host supremum data, open-cover compactness, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8361.3 (Uniform convergence preserves continuity Real route)**.** *Every accepted* UniformConvergencePrese *carrier routes a limit-continuity request through the finite chain*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{ContinuousSupNorm}^{\uparrow} \longmapsto \mathsf{RealFunctionUniformCauchy}^{\uparrow} \longmapsto \mathsf{UniformLimitMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, every displayed tolerance request first reads the compact source and approximating continuous-map rows, then the sup-norm and uniform Cauchy rows, then the finite window and modulus-transfer rows, and finally the terminal Real difference-and-triangle seal. The route exports no host function equality, selected limit object, quotient function space, or ambient compactness beyond the displayed $\mathsf{CompactMetric}^{\uparrow}$ *source.*

Proof. Start with the obligation surface of Definition 8361.2. The compact source K and approximating continuous-map row F fix the source windows and pointwise Real reads that may be inspected.

The sibling sup-norm row S and the uniform Cauchy schedule A then place the approximating function rows under one finite uniform comparison. The uniform-limit metric row L consumes that comparison as the displayed convergence request.

The finite ledger W and modulus-transfer row M carry the approximant continuity test across the same tolerance window. The Real seal R is reached only after the approximant continuity, uniform tail, and limit-row request have all been displayed.

Thus the accepted route is exactly the finite chain shown above, with H, C, P, N adding only transport, replay, provenance, and local naming. Nothing in the carrier can produce host function equality, selected limit data, quotient function-space equality, or compactness outside the displayed $\mathsf{CompactMetric}^{\uparrow}$ row. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}$ packet is generated from UniformLimitMetric, RealFunctionUniformCauchy, ContinuousSupNorm, CompactMetric, ContinuousMap, displayed limit-row, Real difference-and-triangle seal, finite window, modulus-transfer, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8361.1, Definition 8361.2, and Definition 8361.3 seed the continuity-preservation boundary for uniformly convergent continuous real-valued function rows over a displayed compact metric source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergencePreservesContinuity}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected limit construction, quotient function spaces, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked obligation rows, or bridge checking for UniformConvergencePreservesContinuity

Upgrade path: Obligation closure requires checked carrier admission, compact-source and continuous-map stability, sup-norm boundary exactness, uniform Cauchy schedule exactness, uniform-limit handoff, Real triangle-seal nonescape, replay stability, and local naming for BEDC.Derived.UniformConvergencePreservesContinuityUp.

Chapter 8362

A Concrete Naming Certificate for $\mathsf{FinitelyPresentedModule}^{\uparrow}$

Following Bourbaki and Atiyah–Macdonald on finite module presentations, the $\mathsf{FinitelyPresentedModule}^{\uparrow}$ packet records a module by a finite generator row and a finite relation matrix. It sits after the ambient $\mathsf{Ring}^{\uparrow}$ and $\mathsf{Module}^{\uparrow}$ carriers of chapter 43 and chapter 46, reads relation coefficients through the finite $\mathsf{Mat}^{\uparrow}$ surface of chapter 49, keeps Noetherian consumers aligned with chapter 3648, and supplies the presentation route used by the normal-form row of chapter 2262. The packet names only displayed generators, finite relations, exactness witnesses, transport, replay, provenance, and local naming; it does not quotient an ambient free module, select representatives, import a host category of modules, or assert that arbitrary modules admit finite presentations.

Definition 8362.1 (Finitely presented module carrier)**.** *A* $\mathsf{FinitelyPresentedModule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (R, M, G, A, E, H, C, Q, N)$$

with the following displayed rows:

1. *R is the ambient* $\mathsf{Ring}^{\uparrow}$ *row.*
2. *M is the target* $\mathsf{Module}^{\uparrow}$ *row over R.*
3. *G is the finite generator ledger, listing the displayed generator names and their landing rows in M.*
4. *A is the finite* $\mathsf{Mat}^{\uparrow}$ *relation row whose columns name relation words among the generators.*
5. *E is the presentation-exactness ledger, recording that the generator map is exposed and that the displayed relations are exactly the kernel rows read by the certificate.*
6. *H records componentwise* hsame *transport for R, M, G, A, E and the structural rows.*
7. *C records displayed* Cont *replay from a presentation request through R, M, G, A, E, H, Q, N.*
8. *Q is the* Pkg *provenance over* $\mathsf{FinitelyPresentedModule}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{NoetherianModule}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitelyPresentedModule}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, G, A, E, H, C, Q, N$. It has no coordinate for host quotient-module equality, selected normal forms, ambient free-module representatives, a chosen presentation category, or a theorem that every module over R is finitely presented.

Theorem 8362.2 (Finitely presented module NameCert obligations)**.** *For every accepted* $\mathsf{FinitelyPresentedModule}^{\uparrow}$ *carrier*

$$P = (R, M, G, A, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitelyPresentedModule}^{\uparrow}}$ *surface consists exactly of ambient ring exposure, module exposure, finite generator-ledger exposure, finite relation-matrix exposure, presentation-exactness exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A Noetherian-module or Smith-normal-form consumer may read the packet only through these rows.*

Proof. Project the carrier of Definition 8362.1. The rows $R, M, G, A, E, H, C, Q, N$ are the complete finite packet, so carrier admission and classifier exactness are componentwise projections.

Read the algebraic rows next. The ring row R and module row M identify the ambient algebraic surface, the generator ledger G displays the finite source family, and the relation matrix A records the finite relation words through the already named $\mathsf{Mat}^{\uparrow}$ route.

Finally inspect E, H, C, Q, N. The exactness ledger binds generators and relations to the same module row, while transport, replay, provenance, and local naming preserve only the displayed packet. Since the carrier has no coordinate for quotient equality, selected representatives, host module categories, or hidden normal-form output, the stated rows are exactly the NameCert obligations. □

Theorem 8362.3 (Finitely presented module presentation exactness)**.** *Every* $\mathsf{NoetherianModule}^{\uparrow}$ *or* $\mathsf{SmithNormalForm}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitelyPresentedModule}^{\uparrow}$ *carrier reaches the target module only through the finite route*

$$\mathsf{Ring}^{\uparrow}, \mathsf{Module}^{\uparrow} \longmapsto \mathsf{Mat}^{\uparrow} \longmapsto \mathsf{FinitelyPresentedModule}^{\uparrow}.$$

The consumer must read the ambient ring row R, target module row M, generator ledger G, relation matrix A, and exactness ledger E before any normal-form or Noetherian-facing row is consumed. Thus the packet exports finite presentation exactness, not a quotient-module object, selected representatives, ambient basis choice, or a host theorem that presentations exist for all modules.

Proof. Start with the ambient rows. The $\mathsf{Ring}^{\uparrow}$ and $\mathsf{Module}^{\uparrow}$ coordinates are read before the presentation rows, so every downstream consumer sees which module surface is being presented.

Next read the finite presentation rows. The generator ledger G displays the finite source family, the $\mathsf{Mat}^{\uparrow}$ row A displays the relation coefficients, and the exactness ledger E states that these displayed relations are the rows used by the certificate.

The downstream handoff then passes through H, C, Q, N. Componentwise transport, continuation replay, provenance, and local naming carry the same packet into $\mathsf{NoetherianModule}^{\uparrow}$ and $\mathsf{SmithNormalForm}^{\uparrow}$ contexts. They cannot add quotient equality, selected representatives, ambient basis choices, or an existence theorem outside the displayed presentation route. □

Falsifiable prediction: Every accepted $\mathsf{FinitelyPresentedModule}^{\uparrow}$ read exposes a $\mathsf{Ring}^{\uparrow}$ row, a $\mathsf{Module}^{\uparrow}$ row, a finite generator ledger, a finite $\mathsf{Mat}^{\uparrow}$ relation row, a presentation-exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Noetherian-module or Smith-normal-form consumer reads the presented module.

Independence witness: ring, module, matrix, noetherian-module, smith-normal-form: $\mathsf{FinitelyPresentedModule}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an ambient ring, target module, finite

generator ledger, relation matrix, presentation-exactness ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitelyPresentedModule}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitelyPresentedModule}^{\uparrow}$ packet is generated from an ambient RingUp row, a target ModuleUp row, a finite generator ledger, a finite MatrixUp relation row, a presentation-exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitelyPresentedModule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8362.1, Definition 8362.2, and Definition 8362.3 seed the finite module-presentation route for NoetherianModuleUp and SmithNormalFormUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitelyPresentedModule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim quotient-module equality, selected representatives, ambient basis choice, universal finite presentability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator-ledger exposure, relation-matrix exposure, presentation-exactness routing, Noetherian consumer boundary, Smith-normal-form handoff, transport, replay, provenance, and local naming for BEDC.Derived.FinitelyPresentedModuleUp.

Chapter 8363

A Concrete Naming Certificate for $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ packet records the finite modulus data by which a displayed Cauchy cluster is read before a Real or Completion consumer reaches a seal. It sits after the terminal real-name surface of chapter 38, the regular rational readback of chapter 301, the finite stream-window route of chapter 126, the dyadic tolerance core of chapter 291, and the finite cluster-window sibling of chapter 8035. It also reads the Cauchy-filter cluster sibling of chapter 1734 when a filter-facing completion route supplies the cluster source. The packet supplies a finite cluster-modulus interface without choice-selected subsequences, quotient representatives, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8363.1 (Bishop Cauchy cluster modulus carrier)**.** *A* $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, S, Q, D, K, F, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source seal whose completion-facing cluster read is being controlled.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of inspected windows for the cluster read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same inspected windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, comparison slack, and arithmetic budget used by the cluster-modulus checks.*
5. *K is the* $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$ *cluster-window row, recording the finite cells and overlap checks available to the Real-facing consumer.*
6. *F is the optional* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *handoff row used when the same cluster source is supplied by a displayed Cauchy-filter route.*
7. *B is the Bishop cluster-modulus ledger. It records, for each displayed tolerance request, the finite window bound and cluster-cell coverage needed before the downstream completion read may proceed.*
8. *H records componentwise* hsame *transport for the Real seal, stream schedule, regular readback, dyadic tolerance ledger, cluster-window row, filter-cluster handoff, modulus ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S, Q, D \longmapsto K, F \longmapsto B \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteRealCauchyCluster}^{\uparrow}$, $\mathsf{CauchyFilterCluster}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyClusterModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, Q, D, K, F, B, H, C, P, N$. It has no coordinate for a selected subsequence, quotient Cauchy representative, host limit, countable choice, or ambient completeness principle.

Theorem 8363.2 (Bishop Cauchy cluster modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ *carrier*

$$M = (R, S, Q, D, K, F, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyClusterModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Real seal, stream schedule, regular readback, dyadic tolerance ledger, cluster-window row, optional filter-cluster handoff, modulus ledger, transport, replay, provenance, and local naming rows;*
2. *stream-window exposure through S, requiring every modulus request to cite inspected finite windows;*
3. *regular-readback exposure through Q, requiring each inspected window to have a displayed rational readback before it participates in the modulus ledger;*
4. *dyadic-tolerance exactness through D, requiring the tolerance request, comparison slack, and arithmetic budget to be read from the same dyadic core;*
5. *cluster-window coverage through K, and through F when the source is filter-facing, requiring the modulus ledger to cite only displayed cluster cells;*
6. *Bishop modulus ledger exactness through B, requiring each tolerance request to return a finite window bound and cluster-cell coverage row before the Real or Completion consumer reads R;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No obligation is discharged by a selected subsequence, quotient representative, host limit, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier rows from Definition 8363.1. The finite observation data are exactly S, Q, D, while K and F are the only cluster-source rows available to the packet.

The modulus ledger B is downstream from those rows. Hence a tolerance request can be accepted only after the stream windows, regular readbacks, dyadic tolerance budget, and cluster-window cells have been displayed.

The Real-facing row R is not a selected limit or quotient representative. It is the seal consumed after the finite cluster-modulus ledger has supplied a window bound and coverage row for the displayed request.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since no coordinate stores subsequence choice, host completeness, or a hidden Cauchy representative, the listed rows exhaust the local NameCert surface. □

Theorem 8363.3 (Bishop Cauchy cluster modulus completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{FiniteRealCauchyCluster}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with the $\mathsf{CauchyFilterCluster}^{\uparrow}$ *sibling available only as a displayed cluster-source handoff. The consumer receives a Bishop modulus ledger over finite windows and cluster cells, not a choice-selected subsequence, quotient Cauchy representative, host limit, or ambient real-completeness theorem.*

Proof. Start with the obligation theorem Definition 8363.2. It places the stream schedule S, regular readback Q, dyadic tolerance ledger D, and cluster-window row K before the modulus ledger B and the Real seal R.

The finite cluster source is supplied by chapter 8035. When the source is filter-facing, chapter 1734 supplies only the displayed handoff row F; it does not add a second Real seal or a selected limit.

The structural rows H, C, P, N preserve the same route. Therefore the completion-facing read receives a finite cluster-modulus ledger over displayed windows and cluster cells, and no hidden subsequence or quotient representative can be consumed. □

Falsifiable prediction: A $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ consumer that reads a Real or Completion cluster seal without displaying StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, FiniteRealCauchyCluster cells, the Bishop modulus ledger, transport, replay, provenance, and local naming exits the accepted carrier.

Independence witness: real, regseqrat, streamname, dyadicratcore, finite-real-cauchy-cluster, cauchy-filter-cluster: no listed sibling is bijective with $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ because this packet binds tolerance-indexed finite window bounds to cluster-cell coverage before the Real or Completion seal is read.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyClusterModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyClusterModulus}^{\uparrow}$ packet is generated from a RealUp source seal, finite StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerance rows, FiniteRealCauchyCluster cells, an optional CauchyFilterCluster handoff, a Bishop cluster-modulus ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyClusterModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8363.1, Definition 8363.2, and Definition 8363.3 seed the finite cluster-modulus route over displayed Real, RegSeqRat, StreamName, DyadicRatCore, FiniteRealCauchyCluster, CauchyFilterCluster, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyClusterModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subsequence selection, quotient Cauchy representatives, host limits, countable choice, ambient RealUp completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular-readback exposure, dyadic-tolerance exactness, cluster-window coverage, filter-cluster handoff, Bishop modulus ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyClusterModulusUp.

Chapter 8364

A Concrete Naming Certificate for $\mathsf{PanitchpakdiHyperconvexInjectivity}^{\uparrow}$

Following Aronszajn and Panitchpakdi, the $\mathsf{PanitchpakdiHyperconvexInjectivity}^{\uparrow}$ packet records the finite boundary where hyperconvex ball-intersection data may be read as an injective metric-extension interface. It sits over the hyperconvex metric row of chapter 4013, the injective-envelope route of chapter 6810, the isometric embedding handoff of chapter 439, the metric source of chapter 57, the complete-metric seal of chapter 128, and the terminal real-radius rows of chapter 38. The packet names only displayed finite extension boundaries; it does not assert arbitrary-family intersections, choice-selected injective hull points, quotient completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8364.1 (Panitchpakdi hyperconvex injectivity carrier)**.** *A* $\mathsf{PanitchpakdiHyperconvexInjectivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, H, E, I, K, R, T, C, Q, N)$$

with the following displayed rows:

1. *M is the common* $\mathsf{Met}^{\uparrow}$ *source row whose point and distance requests are inspected by all downstream rows.*
2. *H is the* $\mathsf{HyperconvexMetric}^{\uparrow}$ *ball-family row, including displayed closed balls, real radii, pairwise compatibility, and the finite intersection handoff.*
3. *E is the finite injective-extension row, recording a displayed one-step metric extension boundary over the same source rather than a selected host hull point.*
4. *I is the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *handoff row, keeping the source-to-extension distance-preserving route visible before any extension consumer is admitted.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row attached to the same metric source and read only after the ball-family and embedding rows have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *radius and distance row used by the ball-family inequalities, extension budgets, and isometric comparison.*
7. *T records componentwise* hsame *transport for the metric source, hyperconvex row, finite-extension row, isometric handoff, complete-metric seal, real-radius row, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from metric observations through ball-family compatibility, extension admissibility, isometric embedding handoff, complete-metric sealing, and real-radius readback.*

9. *Q is the* Pkg *provenance over* PanitchpakdiHyperconvexInjectivity$^{\uparrow}$, HyperconvexMetric$^{\uparrow}$, InjectiveEnvelopeMetric$^{\uparrow}$, IsometricEmbedding$^{\uparrow}$, Met$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{PanitchpakdiHyperconvexInjectivity}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary ball-family intersections, choice-selected injective hull points, quotient completion representatives, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8364.2 (Panitchpakdi hyperconvex injectivity NameCert obligations)**.** *For every accepted* PanitchpakdiHyperconvexInjectivity$^{\uparrow}$ *carrier $P = (M, H, E, I, K, R, T, C, Q, N)$, the local* NameCert$_{\text{PanitchpakdiHyperconvexInjectivity}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, common-source exposure through M, hyperconvex finite ball-family exposure through H, finite injective-extension exposure through E, isometric embedding handoff through I, complete-metric sealing through K, real-radius and distance readback through R, componentwise* hsame *transport through T, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 8364.1. The packet exposes exactly the metric source M, hyperconvex ball-family row H, finite extension row E, isometric handoff I, complete-metric seal K, real-radius row R, and structural rows T, C, Q, N.

Read the ball-family side first. The row H imports only the finite hyperconvex route of chapter 4013: displayed closed balls, real radii, pairwise compatibility, and the finite intersection handoff over the common metric source.

Read the extension side next. The row E records the finite injective-extension boundary and is consumed through the injective-envelope neighbour of chapter 6810; the row I keeps the isometric embedding handoff visible before the extension is treated as distance preserving.

Finally inspect K, R, T, C, Q, N. The complete-metric and real-radius rows seal the same displayed route, while transport, replay, provenance, and local naming preserve the packet without adding a selected hull point, quotient completion representative, arbitrary-family intersection, choice selector, or ambient completeness theorem. □

Falsifiable prediction: Every accepted PanitchpakdiHyperconvexInjectivity$^{\uparrow}$ read that treats finite hyperconvex ball data as an injective metric-extension boundary must first expose the displayed metric source, hyperconvex ball-family row, finite extension row, isometric embedding handoff, complete-metric seal, real-radius row, transport, replay, provenance, and local naming.

Independence witness: hyperconvex-metric, injective-envelope-metric, isometric-embedding, metric, completemetric, real: no listed sibling supplies the whole packet $P = (M, H, E, I, K, R, T, C, Q, N)$, because each sibling contributes only a metric source, ball-intersection handoff, envelope route, embedding route, completion seal, or real-radius seal while this chapter binds the Panitchpakdi injectivity boundary across all six faces.

Closure status for NameCert$_{\text{PanitchpakdiHyperconvexInjectivity}^{\uparrow}}$.

Constructive story: A PanitchpakdiHyperconvexInjectivity$^{\uparrow}$ packet is generated from a displayed MetricSpace source, HyperconvexMetric finite ball-family row, finite injective-extension row, IsometricEmbedding handoff, CompleteMetric seal, Real radius and distance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PanitchpakdiHyperconvexInjectivity}^{\uparrow}}$, seed)

Scope closed: Definition 8364.1 and Definition 8364.2 seed the finite boundary between hyperconvex ball-intersection data and injective metric-extension reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{PanitchpakdiHyperconvexInjectivity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary-family intersections, choice-selected injective hull points, quotient completions, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, hyperconvex ball-family exposure, finite extension exposure, isometric handoff, complete-metric sealing, Real radius readback, transport, replay, provenance, and local naming for the same packet.

Chapter 8365

A Concrete Naming Certificate for $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ packet records the finite fusion route by which a nested dyadic interval stream supplies one regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ name is consumed. It sits after the dyadic endpoint and comparison rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The packet names only the displayed prefix, shrinking-width, fusion, readback, transport, replay, provenance, and local naming rows; it is not a quotient stream, a selected host limit, a countable-choice selector, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8365.1 (Nested dyadic interval fusion carrier)**.** *A* $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested dyadic interval prefix row, storing the finite chain of left and right endpoint cells.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and width ledger for those cells, including the displayed containment and shrinking-width comparisons.*
3. *W is the fusion-window row, selecting the common finite prefix that every downstream read must inspect.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule attached to the same prefix.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row formed from the scheduled dyadic observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row, available only after I, D, W, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the interval prefix, dyadic ledger, fusion window, stream schedule, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a fusion request through I, D, W, S, R, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedDyadicIntervalFusion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, W, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected completed limits, host real equality, countable choice, trichotomy, or ambient real completeness.

Theorem 8365.2 (Nested dyadic interval fusion NameCert obligations)**.** *For every accepted* $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ *carrier*

$$F = (I, D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedDyadicIntervalFusion}^{\uparrow}}$ *surface consists exactly of nested-prefix exposure through I, dyadic endpoint and width exposure through D, common fusion-window exposure through W, finite observation scheduling through S, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8365.1. The projection exposes the finite interval prefix, dyadic ledger, fusion window, schedule, regular readback, real seal, transport, replay, provenance, and naming rows.

The analytic rows are ordered. The dyadic ledger D can be read only after a prefix I is displayed, the fusion window W restricts all later observations to a common finite prefix, and the stream schedule S names the observations allowed inside that prefix.

The regular and real-facing rows are downstream. The readback row R is formed from the scheduled dyadic observations, and the terminal seal E is available only after that readback is displayed.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient stream, selected limit, host real equality, choice, trichotomy, or ambient completeness coordinate, the listed rows exhaust the local certificate. □

Theorem 8365.3 (Nested dyadic interval fusion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the finite rows $I, D, W, S, R, E, H, C, P, N$. The handoff exposes a common nested dyadic prefix and one regular rational readback route before the terminal seal is consumed; it does not expose quotient real equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations. They force a Real-facing consumer to expose I, D, W, S, R before the seal row E is available.

Read the sibling surfaces in order. The dyadic rational core supplies endpoint and width comparisons for D, the stream-name interface supplies the finite observation schedule S, and the regular rational interface supplies the readback row R.

Finally consume the terminal seal through $\mathsf{Real}^{\uparrow}$. The rows H, C, P, N preserve the same finite fusion route under transport, replay, provenance, and local naming, so no downstream consumer can replace the route by a quotient stream, a selected limit, or an ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ read displays the nested dyadic prefix, DyadicRatCore endpoint ledger, common fusion window, StreamName schedule,

RegSeqRat readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any Real-facing consumer can use the terminal seal.

Independence witness: dyadicratcore, streamname, regseqrat, real: none is bijective to NestedDyadicIntervalFusio because only this packet binds a common nested dyadic prefix, a fusion window, scheduled observations, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming as one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedDyadicIntervalFusion}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedDyadicIntervalFusion}^{\uparrow}$ packet is generated from nested dyadic interval prefixes, DyadicRatCore endpoint and width rows, one common fusion window, StreamName finite observations, RegSeqRat readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedDyadicIntervalFusion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8365.1, Definition 8365.2, and Definition 8365.3 seed the finite nested dyadic interval fusion route from displayed dyadic prefix data to a Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedDyadicIntervalFusion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, host real equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic prefix exposure, fusion-window scheduling, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedDyadicIntervalFusionUp.

Chapter 8366

A Concrete Naming Certificate for $\mathsf{MoreraTheorem}^{\uparrow}$

Following Morera and Bishop–Bridges constructive complex analysis, the $\mathsf{MoreraTheorem}^{\uparrow}$ packet names the finite theorem-shaped route by which triangle-contour integral vanishing is read as a holomorphic handoff. It sits beside the complex-analysis carrier of chapter 76, the holomorphic output surface of chapter 68, the finite contour window of chapter 587, the complex limit rows of chapter 65, and the real tolerance ledger of chapter 38. The packet records only displayed triangle contours, finite contour-integral rows, local complex-limit evidence, real tolerance comparisons, transport, replay, provenance, and local naming; it does not import analytic continuation, a host path-integral theory, open-cover compactness, selected primitives, quotient path equality, or classical choice.

Definition 8366.1 (Morera theorem carrier)**.** *A* $\mathsf{MoreraTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, H, T, W, L, R, Z, E, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CplxAnalytic}^{\uparrow}$ *context row, naming the complex neighbourhood and the displayed analytic-read request.*
2. *H is the* $\mathsf{Holomorphic}^{\uparrow}$ *handoff row that may be exported only after the triangle-contour vanishing rows have been inspected.*
3. *T is the finite triangle-contour row, recording three* $\mathsf{Complex}^{\uparrow}$ *vertex histories, oriented edge rows, and the bounded family of triangle windows to be tested.*
4. *W is the* $\mathsf{ContourSumWindow}^{\uparrow}$ *row, recording the finite contour-window ledger and the local sum requests for the triangle boundaries.*
5. *L is the* $\mathsf{CplxLim}^{\uparrow}$ *row, recording the local limiting comparison that turns the finite contour-window data into the analytic handoff.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance row, recording positive rational bounds, modulus comparisons, and real-valued estimates used by the finite windows.*
7. *Z is the zero-contour row, displaying that each accepted triangle-window integral request is sealed at* $0_{\mathsf{Complex}^{\uparrow}}$ *inside the finite ledger.*

8. *E records componentwise* hsame *transport for analytic context, holomorphic handoff, triangle contours, contour windows, complex limits, real tolerances, zero rows, provenance, and naming.*
9. *C records displayed* Cont *replay from triangle-window admission through zero-contour sealing and the holomorphic handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{MoreraTheorem}^{\uparrow}$, $\mathsf{CplxAnalytic}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{ContourSumWindow}^{\uparrow}$, $\mathsf{CplxLim}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MoreraTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, H, T, W, L, R, Z, E, C, P, N$. It has no coordinate for a selected primitive, analytic continuation, an ambient path integral, a quotient path space, an open-cover compactness proof, or a classical-choice schedule.

Theorem 8366.2 (Morera theorem NameCert obligations)**.** *For every accepted* $\mathsf{MoreraTheorem}^{\uparrow}$ *carrier $M = (A, H, T, W, L, R, Z, E, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MoreraTheorem}^{\uparrow}}$ *surface consists exactly of* $\mathsf{CplxAnalytic}^{\uparrow}$ *context admission through A,* $\mathsf{Holomorphic}^{\uparrow}$ *handoff exposure through H, finite triangle-contour display through T,* $\mathsf{ContourSumWindow}^{\uparrow}$ *ledger display through W,* $\mathsf{CplxLim}^{\uparrow}$ *limiting comparison through L,* $\mathsf{Real}^{\uparrow}$ *tolerance comparison through R, zero-contour sealing through Z, componentwise* hsame *transport through E, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8366.1. The analytic rows are A, H, T, W, L, R, Z, and the structural rows are E, C, P, N.

The triangle-contour row T and contour-window row W expose the finite integral requests before the holomorphic handoff is read. The zero row Z seals those requests at $\mathsf{0}_{\mathsf{Complex}^{\uparrow}}$ inside the displayed window, rather than by an ambient path-integral theorem.

The complex-limit row L and real tolerance row R supply the constructive limiting and estimate data consumed by the handoff H. Transport, replay, provenance, and local naming preserve the same packet, so no selected primitive, analytic-continuation principle, quotient path equality, or classical-choice schedule can be read from the carrier. □

Theorem 8366.3 (Morera theorem contour-window route)**.** *Every complex-analysis consumer that reads an accepted* $\mathsf{MoreraTheorem}^{\uparrow}$ *packet must factor the read through the finite route*

$$A, T, W, R, Z, L \longmapsto H \longmapsto E, C, P, N.$$

The theorem-shaped content is the displayed passage from finite triangle-contour vanishing to the $\mathsf{Holomorphic}^{\uparrow}$ *handoff over the named* $\mathsf{CplxAnalytic}^{\uparrow}$ *context. It is not an analytic-continuation theorem, a hidden primitive selector, a host contour-integral object, a quotient path equality, or an open-cover compactness principle.*

Proof. Begin with the accepted carrier. The rows A, T, W display the complex-analysis context, triangle contours, and contour-window ledger before any handoff is exposed.

The real tolerance row R and zero-contour row Z then bind the finite estimates and vanishing certificates for the same triangle windows. The complex-limit row L is read after those rows, so the limiting comparison remains attached to the finite contour-window surface.

Finally H exports the holomorphic handoff, while E, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded ambient objects, every accepted consumer remains on the finite contour-window route. □

Falsifiable prediction: Any $\mathsf{MoreraTheorem}^{\uparrow}$ consumer that exports a $\mathsf{Holomorphic}^{\uparrow}$ handoff while omitting the displayed $\mathsf{ContourSumWindow}^{\uparrow}$, $\mathsf{CplxLim}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows is rejected by the local classifier.

Independence witness: complexanalytic, holomorphic, contoursumwindow, complexlimit, real: $\mathsf{MoreraTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite triangle-contour vanishing, contour-window ledgers, complex-limit comparisons, real tolerance rows, holomorphic handoff, transport, replay, provenance, and local naming into one theorem-shaped packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MoreraTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MoreraTheorem}^{\uparrow}$ packet is generated from ComplexAnalytic context rows, Holomorphic handoff rows, triangle-contour rows, ContourSumWindow ledgers, ComplexLimit comparisons, Real tolerance rows, zero-contour seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MoreraTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8366.1, Definition 8366.2, and Definition 8366.3 seed the finite Morera theorem contour-window route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MoreraTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, analytic continuation, selected primitives, host path integration, open-cover compactness, quotient path equality, or classical choice for $\mathsf{MoreraTheorem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, triangle-contour display, contour-window exactness, ComplexLimit comparison, Real tolerance replay, zero-contour sealing, Holomorphic handoff, transport, replay, provenance, and local naming for BEDC.Derived.MoreraTheoremUp.

Chapter 8367

A Concrete Naming Certificate for BishopCompletionUniquenessPrinciple$^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the BishopCompletionUniquenessPrinciple$^{\uparrow}$ packet records the textbook uniqueness principle for completion presentations as finite BEDC NameCert data. It sits after the criterion surface of chapter 3404, the located uniqueness boundary of chapter 7277, the regular Cauchy completion uniqueness route of chapter 1782, and the regular Cauchy representative uniqueness route of chapter 3390. The packet gives L10 Real and Completion consumers one finite uniqueness surface; it does not assert quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient Real$^{\uparrow}$ completeness.

Definition 8367.1 (Bishop completion uniqueness principle carrier)**.** *A* BishopCompletionUniquenessPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, L, C, R, S, Q, D, E, H, K, P, N)$$

with the following displayed rows:

1. *B is the* BishopCompletionCriterion$^{\uparrow}$ *row exposing the completion-admission criterion before any uniqueness read is available.*
2. *L is the* LocatedCompletionUniqueness$^{\uparrow}$ *row recording the located uniqueness comparison for the same displayed completion source.*
3. *C is the* RegularCauchyCompletionUniqueness$^{\uparrow}$ *row recording the regular Cauchy completion uniqueness route.*
4. *R is the* RegularCauchyUniqueness$^{\uparrow}$ *row comparing representative regular Cauchy names before the terminal seal is consumed.*
5. *S is the paired* StreamName$^{\uparrow}$ *finite-window row inspected by the criterion and both uniqueness routes.*
6. *Q is the paired* RegSeqRat$^{\uparrow}$ *regular rational readback row extracted from those finite windows.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used by the displayed comparison routes.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after B, L, C, R, S, Q, D have been displayed.*

9. *H records componentwise* hsame *transport for the criterion, located uniqueness, regular completion uniqueness, representative uniqueness, window, readback, tolerance, seal, continuation, provenance, and local naming rows.*
10. *K records displayed* Cont *replay from a completion uniqueness request back through B, L, C, R, S, Q, D, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}$, $\mathsf{BishopCompletionCriterion}^{\uparrow}$, $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}}$ *row naming the displayed principle and its permitted completion consumers.*

The classifier compares accepted carriers only through these rows and their replay paths. It has no coordinate for quotient completion equality, selected limits, countable choice, host equality, proof-irrelevant representative collapse, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8367.2 (Bishop completion uniqueness principle NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}$ *carrier*

$$U = (B, L, C, R, S, Q, D, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}}$ *surface consists exactly of criterion admission through B, located uniqueness exposure through L, regular Cauchy completion uniqueness exposure through C, representative regular Cauchy uniqueness exposure through R, finite StreamName window admission through S, RegSeqRat readback through Q, DyadicRatCore tolerance comparison through D, Real seal nonescape through E, componentwise hsame transport through H, displayed Cont replay through K, Pkg provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8367.1. The analytic rows are exactly B, L, C, R, S, Q, D, E, and the structural rows are H, K, P, N.

The completion-admission criterion is read first. Row B supplies the Bishop completion criterion, so a uniqueness-facing consumer cannot enter the completion boundary without the displayed finite criterion surface.

The uniqueness comparison is then forced through the three displayed sibling routes. Row L supplies located completion uniqueness, row C supplies regular Cauchy completion uniqueness, and row R supplies regular Cauchy representative uniqueness over the same finite source.

The finite source rows bind those sibling routes to concrete data. Rows S, Q, D expose the StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance ledger used by the criterion and the uniqueness comparisons before the terminal Real seal E is available.

Finally H, K, P, N transport, replay, source, and name the same route. Since the excluded quotient, selection, host-equality, choice, representative-collapse, and ambient-completeness objects have no carrier coordinate, none can be exported by the local NameCert surface. □

Theorem 8367.3 (Bishop completion uniqueness principle finite route)**.** *Every completion-facing uniqueness read exported by an accepted* $\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}$ *carrier factors through the ordered finite route*

$$\mathsf{BishopCompletionCriterion}^{\uparrow},\ \mathsf{LocatedCompletionUniqueness}^{\uparrow},\ \mathsf{RegularCauchyCompletionUniqueness}^{\uparrow},\ \mathsf{RegularCauchyU}$$

Thus the L10 Real and Completion consumer receives uniqueness only as a displayed NameCert route through the completion criterion, located uniqueness, regular Cauchy completion uniqueness, representative uniqueness, finite windows, regular rational readback, and dyadic tolerance data.

Proof. Use Definition 8367.2 to identify the complete local obligation surface. Any exported uniqueness read must consume B, L, C, R, S, Q, D before the seal row E is visible.

The first four rows bind the principle to existing completion uniqueness packets. The criterion row B admits the completion-facing read, while L, C, R record the located, regular completion, and representative uniqueness routes over the same displayed source.

The finite source rows prevent escape from the displayed route. Rows S, Q, D expose the finite windows, regular rational readback, and dyadic tolerance ledger used by the sibling uniqueness routes.

The terminal rows add no new analytic object. Row E is only a Real seal, and H, K, P, N only transport, replay, source, and name the route, so the exported read cannot become quotient completion equality, a selected limit, countable choice, host equality, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}$ uniqueness read must expose $\mathsf{BishopCompletionCriterion}^{\uparrow}$, $\mathsf{LocatedCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: bishopcompletioncriterion, locatedcompletionuniqueness, regularcauchycompletionuniqueness, regularcauchyuniqueness, streamname, regseqrat, dyadicratcore, real: BishopCompletionUnic is not bijective to any listed sibling because it jointly carries the completion criterion, located uniqueness, regular completion uniqueness, representative uniqueness, finite windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming while refusing quotient completion equality and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}$ packet is generated from a BishopCompletionCriterion row, LocatedCompletionUniqueness row, RegularCauchyCompletionUniqueness row, RegularCauchyUniqueness row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8367.1, Definition 8367.2, and Definition 8367.3 seed the finite L10 completion uniqueness principle over BishopCompletionCriterion, LocatedCompletionUniqueness, RegularCauchyCompletionUniqueness, RegularCauchyUniqueness, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniquenessPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling uniqueness exposure, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparison, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 8368

A Concrete Naming Certificate for MetaCICCandidateResidualJoinBudget$^{\uparrow}$

The MetaCICCandidateResidualJoinBudget$^{\uparrow}$ packet realizes the finite candidate-residual join-budget branch of chapter 9378. It sits between the parallel-diamond frontier of chapter 4238, the confluence audit packet of chapter 735, and the residual-substitution boundary of chapter 2181. The packet records typed residual windows, candidate join budgets, closed-shape guards, obstruction retention, TasteGate-facing instance requirements, transport, replay, provenance, and local naming before a confluence-facing consumer may read a local join attempt.

Definition 8368.1 (MetaCIC candidate residual join budget carrier)**.** *A* MetaCICCandidateResidualJoinBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, R, B, Q, D, O, I, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, naming the closed or shape-limited MetaCIC source window being inspected.*
2. *R is the residual-substitution row, recording the residual frontier that remains visible before a candidate join can be consumed.*
3. *B is the finite budget row, recording unary fuel, inspected critical-pair identifiers, and the local residual-join search bound.*
4. *Q is the candidate-join row, naming the displayed common-reduct attempt and its budgeted replay trace.*
5. *D is the closed-shape guard row, separating the admitted closed-normal or bounded-shape branch from the unclosed residual-substitution boundary.*
6. *O is the obstruction row, retaining any missing residual-substitution compatibility, parallel-diamond input, or closed-confluence premise as visible data.*
7. *I is the TasteGate instance row, stating that Lean-side gate discharge must provide* Nontrivial *and* FieldFaithful *instances for the candidate residual-budget surface.*
8. *H records componentwise* hsame *transport for typed-window, residual-substitution, budget, candidate-join, guard, obstruction, instance, provenance, and naming rows.*

9. *C records displayed* Cont *replay from typed-window admission through residual inspection, budgeted candidate join, obstruction retention, and the instance row.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose displayed residual window, budget, candidate join, guard, obstruction, instance, transport, replay, provenance, and naming rows agree componentwise. It has no coordinate for global confluence, unrestricted residual-substitution compatibility, subject reduction, quotient syntax equality, evaluator equality, or a hidden join oracle.

Theorem 8368.2 (MetaCIC candidate residual join budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$ *carrier* $J = (T, R, B, Q, D, O, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, residual-substitution exposure through R, finite budget exposure through B, candidate-join exposure through Q, closed-shape guard exposure through D, obstruction retention through O, TasteGate instance disclosure through I, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8368.1. The confluence-facing rows are T, R, B, Q, D, O, I, and the structural rows are H, C, P, N.

The typed-window row T and residual-substitution row R are read before the finite budget B and candidate join Q. Thus a local join attempt remains attached to the residual frontier named by the confluence audit map.

The guard row D and obstruction row O then keep the closed-shape branch separate from missing substitution-compatibility and parallel-diamond inputs. The instance row I records the Nontrivial and FieldFaithful requirements that a Lean TasteGate must discharge before the packet can be promoted beyond a candidate budget.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate exports global confluence, subject reduction, quotient syntax equality, evaluator equality, or a hidden join oracle, the listed rows exhaust the local NameCert surface. □

Theorem 8368.3 (MetaCIC candidate residual join budget non-escape)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$ *packet receives the finite route*

$$T, R, B, Q, D, O, I, H, C, P, N.$$

The route may be handed to $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *consumers only as candidate residual-join budget evidence with the obstruction row still visible. It cannot be repacked as full confluence, subject reduction, unrestricted residual-substitution compatibility, quotient syntax equality, or evaluator equality.*

Proof. Start with the NameCert obligations of Definition 8368.2. They expose the typed window, residual frontier, finite budget, candidate join, guard, obstruction, instance row, and structural rows as one packet.

The candidate row Q is downstream of both the residual row R and budget row B. Therefore a consumer sees a budgeted join attempt, not an unrestricted common-reduct theorem.

The obstruction row O is transported and replayed together with D and I. Hence the handoff to the parallel-diamond frontier, confluence audit packet, or residual-substitution boundary preserves the missing-premise boundary and the TasteGate instance requirements. □

Falsifiable prediction: Any $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$ consumer that exports a candidate join while omitting the residual-substitution row, budget row, obstruction row, or Nontrivial and FieldFaithful TasteGate instance requirements is rejected by the local classifier.

Independence witness: Nearest siblings: MetaCICParallelDiamondFrontier, MetaCICConfluenceAuditPacket, MetaCICResidualSubstitutionBoundary. The packet is not bijective to any sibling because it binds typed residual windows, finite candidate-join budgets, closed-shape guards, obstruction retention, TasteGate instance requirements, transport, replay, provenance, and local naming into one confluence-audit object.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$ packet is generated from typed-window rows, residual-substitution rows, finite budget rows, candidate-join rows, closed-shape guards, obstruction rows, TasteGate instance requirements, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8368.1, Definition 8368.2, and Definition 8368.3 seed the finite candidate residual-join budget route for the MetaCIC confluence audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, full confluence, subject reduction, unrestricted residual-substitution compatibility, quotient syntax equality, evaluator equality, or hidden join selection for $\mathsf{MetaCICCandidateResidualJoinBudget}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, residual-substitution exposure, budget replay, candidate-join admission, closed-shape guard retention, obstruction retention, TasteGate instance disclosure, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateResidualJoinBudgetUp.

Chapter 8369

A Concrete Naming Certificate for MetaCICAuditSynthesisRouter$^{\uparrow}$

The MetaCICAuditSynthesisRouter$^{\uparrow}$ packet realizes the audit-synthesis route of section 9376.7 as one finite router over subject-reduction, confluence, normalization, substitution, decidability, blocker, and open-input rows. It sits beside the audit-synthesis packet of chapter 776, the critical-path packet of chapter 765, the confluence audit packet of chapter 735, the normalization frontier of chapter 892, and the discharge-obligation ledger of chapter 717. The router names a finite audit handoff; it preserves the open-input boundary instead of claiming subject reduction, confluence, normalization, decidability, closed consistency, quotient syntax equality, evaluator soundness, or MetaCIC kernel consistency.

Definition 8369.1 (MetaCIC audit-synthesis router carrier)**.** *A* MetaCICAuditSynthesisRouter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, C, N, U, D, B, O, K, F, H, T, P, L)$$

with the following displayed rows:

1. *S is the subject-reduction route row, recording the obstruction boundary and the discharge rows that remain required before closed subject reduction could be consumed.*
2. *C is the confluence route row, recording the conditional confluence audit surface, critical-pair rows, local-join rows, and retained residual-substitution premises.*
3. *N is the normalization route row, recording candidate-mediated strong-normalization evidence, bounded normal endpoints, and the normal-frontier handoff.*
4. *U is the substitution route row, recording closed-substitution, binder-budget, residual, and open-input side conditions that must remain visible to the router.*
5. *D is the decidability route row, recording bounded conversion, typed-checker, and finite observation data without turning them into global decidability.*
6. *B is the blocker row, recording the closed-normal branch, the subject-reduction obstruction, and the consistency boundary that prevents the route from becoming a closed consistency theorem.*
7. *O is the open-input row, recording inputs, premises, sockets, and side conditions that are intentionally not discharged by the router.*

8. *K is the critical-path row, read through* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *as the source ledger tying the preceding rows to the MetaCIC audit-synthesis route.*
9. *F is the field-faithfulness readiness row, recording that future TasteGate work must keep $S, C, N, U, D, B, O, K, H$ as separate carrier fields.*
10. *H records componentwise* hsame *transport for every route row, the critical-path row, field-faithfulness readiness, replay, provenance, and local naming.*
11. *T records displayed* Cont *replay from the audit-synthesis request through the seven route rows, the critical-path source, the open-input boundary, and the local name.*
12. *P is the* Pkg *provenance over the audit-synthesis packet, critical-path packet, confluence audit packet, normalization frontier, discharge-obligation ledger,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
13. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}}$ *row.*

The classifier compares accepted routers only through $S, C, N, U, D, B, O, K, F, H, T, P, L$. It has no coordinate for premise-free subject reduction, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, or MetaCIC kernel consistency.

Theorem 8369.2 (MetaCIC audit-synthesis router obligations)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ *carrier*

$$R = (S, C, N, U, D, B, O, K, F, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}}$ *surface consists exactly of subject-reduction route exposure, confluence route exposure, normalization route exposure, substitution route exposure, decidability route exposure, blocker retention, open-input retention, critical-path sourcing, field-faithfulness readiness, componentwise transport, displayed replay, provenance, local naming, and non-escape from premise-free subject reduction, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, and MetaCIC kernel consistency.*

Proof. Project the carrier of Definition 8369.1. The route rows S, C, N, U, D expose the five audit families separately, so the router has no coordinate that merges their conclusions into one theorem.

The blocker and open-input rows are then read as boundaries. Row B keeps the closed-normal branch and subject-reduction obstruction visible, while O records the sockets and premises that remain outside the router.

The critical-path row K sources the route without discharging those boundaries. The field-faithfulness readiness row F records that future TasteGate evidence must preserve every displayed field rather than ignore a route coordinate.

Finally, H, T, P, L transport, replay, source, and name the same finite packet. Since no row supplies premise-free subject reduction, unrestricted Church–Rosser, candidate-free normalization, global decidability, quotient equality, evaluator soundness, closed consistency, or kernel consistency, the listed rows exhaust the local certificate surface. □

Theorem 8369.3 (MetaCIC audit-synthesis router non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ *packet must read the finite route*

$$S, C, N, U, D \longmapsto B, O, K, F \longmapsto H, T, P, L.$$

The resulting handoff may coordinate subject-reduction, confluence, normalization, substitution, decidability, blocker, and open-input rows as audit data, but it cannot export subject reduction, confluence, normalization, decidability, closed consistency, quotient syntax equality, evaluator soundness, or MetaCIC kernel consistency as closed theorems.

Proof. Begin with the NameCert obligations of Definition 8369.2. They place every public read downstream of the five route rows S, C, N, U, D.

The blocker and open-input rows are not optional. A consumer that omits B loses the closed-normal and subject-reduction obstruction boundary, and a consumer that omits O loses the premises and sockets that the route intentionally leaves open.

The critical-path and field-faithfulness rows keep the handoff source-local. Row K points back to the displayed MetaCIC critical path, while F records the future requirement that no classifier may ignore one of the displayed fields.

Thus the structural rows H, T, P, L can only replay the finite audit-synthesis router. They cannot create any of the closed theorem endpoints excluded by the carrier. □

Falsifiable prediction: A purported $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ read is refuted if it exports subject reduction, confluence, normalization, decidability, closed consistency, quotient syntax equality, evaluator soundness, or MetaCIC kernel consistency without displaying the subject-reduction, confluence, normalization, substitution, decidability, blocker, open-input, critical-path, transport, replay, provenance, and local naming rows.

Independence witness: metacicauditsynthesispacket, metaciccriticalpath, metacicconfluenceauditpacket, metacicnormalizationfrontier, metacicdischargeobligationledger: none is bijective to $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ because this packet jointly carries subject-reduction, confluence, normalization, substitution, decidability, blocker, open-input, critical-path, field-faithfulness readiness, transport, replay, provenance, and local NameCert rows.

The Lean TasteGate follow-up for $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ must provide explicit Nontrivial MetaCICAuditS and FieldFaithful $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ instances. These instances must preserve $S, C, N, U, D, B, O, K, F,$ as separate fields before any checked carrier-alignment theorem can serve as a public gate.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}$ packet is generated from subject-reduction, confluence, normalization, substitution, decidability, blocker, open-input, critical-path, field-faithfulness readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8369.1, Definition 8369.2, and Definition 8369.3 seed the finite MetaCIC audit-synthesis router and its open-input non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, unrestricted confluence, candidate-free normalization, global beta-conversion decidability, quotient syntax equality, evaluator soundness, closed consistency, Lean verification, bridge checking, or MetaCIC kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, seven-row audit route exposure, blocker retention, open-input retention, critical-path sourcing, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICAuditSynthesisRouterUp.

Chapter 8370

A Concrete Naming Certificate for $\mathsf{SchurDecomposition}^{\uparrow}$

Following Schur and standard finite-dimensional spectral theory, the $\mathsf{SchurDecomposition}^{\uparrow}$ packet records the finite route by which a displayed square matrix is read through a unitary change-of-basis ledger, an upper-triangular row, and a diagonal spectral readback. It consumes the matrix carrier of chapter 49, the linear-map surface of chapter 48, the inner-product rows of chapter 86, the unitary-row discipline of chapter 219, and the real-spectrum boundary of chapter 3816. The packet names only finite matrix, unitary, triangular, spectral, transport, replay, provenance, and local NameCert data; it does not import host diagonalization, quotient matrix equality, a selected eigenbasis outside the displayed rows, or ambient spectral theory.

Definition 8370.1 (Schur decomposition carrier)**.** *A* $\mathsf{SchurDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, F, U, T, \Delta, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source square-matrix row, read through* $\mathsf{Mat}^{\uparrow}$ *and the underlying* $\mathsf{LinMap}^{\uparrow}$ *action surface.*
2. *F is the scalar-field or complex-real spectrum boundary row, exposing the finite scalar readback interface used by the diagonal entries.*
3. *U is the unitary change-of-basis ledger, read through* $\mathsf{InnerProduct}^{\uparrow}$ *and* $\mathsf{UnitaryGroup}^{\uparrow}$*.*
4. *T is the upper-triangular matrix ledger obtained after the displayed unitary change of basis.*
5. *Δ is the diagonal spectral readback row, listing the diagonal scalar reads of T through the spectrum boundary F.*
6. *E is the residual triangular exactness row saying that the displayed source matrix is replayed by the U, T route and that no off-ledger remainder is exported.*
7. *H records componentwise* hsame *transport for the source matrix, scalar boundary, unitary ledger, triangular ledger, diagonal readback, exactness row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the source matrix through the scalar boundary, unitary change of basis, triangular ledger, diagonal spectral readback, and residual exactness row.*
9. *P is the* Pkg *provenance over* $\mathsf{SchurDecomposition}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{UnitaryGroup}^{\uparrow}$*,* $\mathsf{RealSpectrum}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchurDecomposition}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying all ten rows. It has no coordinate for a host eigenbasis, quotient matrix equality, normal diagonalization, selected spectral ordering outside Δ, or an ambient theorem asserting that every host matrix has been decomposed.

Theorem 8370.2 (Schur decomposition NameCert obligations)**.** *For every accepted* $\mathsf{SchurDecomposition}^{\uparrow}$ *carrier*

$$D = (A, F, U, T, \Delta, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchurDecomposition}^{\uparrow}}$ *surface consists exactly of source matrix admission for A, scalar or spectrum-boundary admission for F, unitary-ledger admission for U, upper-triangular ledger admission for T, diagonal spectral readback for Δ, residual triangular exactness for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No Schur decomposition obligation is stored outside these rows.

Proof. Project the accepted carrier. The source row A is the only matrix being named, and F is the only scalar or spectrum boundary through which diagonal entries may be read.

The unitary and triangular data are displayed separately. The ledger U records the change-of-basis route, while T records the upper-triangular matrix obtained on that route.

The diagonal readback Δ and exactness row E then tie the spectrum-facing entries and triangular residual to the same replay path. The support rows H, C, P, N transport, replay, source, and name this finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 8370.3 (Schur decomposition unitary triangular handoff)**.** *Every finite-dimensional spectral, matrix-factorization, or linear-map consumer of an accepted* $\mathsf{SchurDecomposition}^{\uparrow}$ *carrier must first read*

$$A, F, U, T, \Delta, E, H, C, P, N.$$

The unitary handoff is visible only through U after the source matrix row A and scalar boundary F have been displayed, and the triangular handoff is visible only through T, Δ, E after that unitary ledger has been read. The handoff exports no host eigenbasis, quotient matrix equality, unrestricted diagonalization theorem, or selected ordering of spectrum entries outside the displayed diagonal ledger.

Proof. Start from an accepted carrier. The consumer first reads A and F, so the matrix and scalar boundary are fixed before any change-of-basis row is available.

The unitary row U then supplies the displayed change-of-basis ledger. Since the carrier has no other change-of-basis coordinate, any spectral or factorization consumer must pass through this ledger.

Finally T, Δ, E expose the upper-triangular matrix, the diagonal spectral readback, and the residual exactness row on the same route. Transport, replay, provenance, and local naming through H, C, P, N preserve the route and add no host-level diagonalization or quotient equality. □

Theorem 8370.4 (Schur decomposition spectrum non-escape)**.** *Every spectrum-boundary read from an accepted* $\mathsf{SchurDecomposition}^{\uparrow}$ *carrier factors through the displayed diagonal ledger Δ and the residual triangular exactness row E after the source matrix, scalar boundary, unitary ledger, and upper-triangular ledger have already been read. A consumer cannot detach a diagonal entry from $A, F, U, T, \Delta, E, H, C, P, N$ and treat it as an ambient host eigenvalue, a quotient representative, or a hidden spectral selector.*

Proof. Read the carrier rows in order. The diagonal spectral ledger Δ appears only after the source matrix, scalar boundary, unitary ledger, and upper-triangular ledger have been displayed.

The exactness row E keeps the diagonal readback attached to the residual triangular route. It does not provide a second spectral interface or a host eigenvalue selector.

The structural rows H, C, P, N preserve that same finite route. Hence every spectrum-facing consumer remains inside the displayed packet and cannot obtain quotient representatives, hidden eigenvalue choices, or an ambient spectral theorem from the carrier. □

Lean4 (statement) `BEDC.Derived.SchurDecompositionUp`

Closure status for $\mathsf{NameCert}_{\mathsf{SchurDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{SchurDecomposition}^{\uparrow}$ packet is generated from a source MatrixUp row, scalar or RealSpectrumUp boundary, unitary change-of-basis ledger, upper-triangular matrix ledger, diagonal spectral readback, residual exactness row, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchurDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8370.1, Definition 8370.2, Definition 8370.3, and Definition 8370.4 seed the finite Schur decomposition packet over matrix, linear-map, inner-product, unitary, spectrum, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchurDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host diagonalization, quotient matrix equality, selected eigenbasis construction, spectral ordering outside the displayed ledger, or ambient finite-dimensional spectral theory.

Upgrade path: Obligation closure requires checked source-matrix admission, scalar-boundary admission, unitary-ledger exactness, triangular-ledger exactness, diagonal spectral readback, residual non-escape, transport stability, replay stability, and NameCert obligations for BEDC.Derived.SchurDecompositionUp.

Chapter 8371

A Concrete Naming Certificate for $\mathsf{IntegralTest}^{\uparrow}$

Following Hardy's treatment of convergence tests and the Bishop–Bridges constructive reading of monotone real functions, the $\mathsf{IntegralTest}^{\uparrow}$ packet records the textbook integral test as finite NameCert data for Real and Completion consumers. It sits beside the narrower $\mathsf{CauchyIntegralTest}^{\uparrow}$ route of chapter 2259, the source-series surface of chapter 1581, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite comparison route from a nonnegative decreasing tail to a convergence handoff; it does not choose an improper integral value, select an infinite sum, quotient partial-sum streams, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8371.1 (IntegralTest carrier)**.** *An* $\mathsf{IntegralTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, F, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing the inspected nonnegative tail only through finite partial-sum windows.*
2. *F is the monotone real-function comparison row, recording the decreasing majorant or minorant data used by the integral test.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the series tail and the matching integral cells.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for endpoint, mesh, and tail-budget comparisons.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed tail windows and dyadic tolerances.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from Q and from no independent completed-sum coordinate.*
7. *H records componentwise* hsame *transport for the source series, comparison function, windows, dyadic budgets, readback, and seal.*
8. *C records displayed* Cont *replay for the same comparison route.*
9. *P is the* Pkg *provenance over* $\mathsf{IntegralTest}^{\uparrow}$, $\mathsf{CauchyIntegralTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntegralTest}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected improper integral, a host infinite sum, quotient equality of series names, countable choice, trichotomy, or ambient completeness.

Theorem 8371.2 (IntegralTest NameCert obligations)**.** *For every accepted* $\mathsf{IntegralTest}^{\uparrow}$ *carrier* $I = (S, F, W, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntegralTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure through* S*, monotone comparison exposure through* F*, finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure through* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The rows of Definition 8371.1 are precisely the series source, monotone comparison function, finite windows, dyadic tolerances, regular readback, terminal seal, transport, replay, provenance, and local naming rows.

The comparison part is finite: S exposes only inspected series tails, F exposes only the monotone comparison data, W names the windows, and D names the dyadic budgets. The readback row Q is therefore downstream of displayed tail and integral-cell data.

The terminal row E can then be read only after Q has been exposed. The structural rows H, C, P, N transport, replay, source, and name the same route, so the local certificate contains exactly the listed obligations and no host summation or ambient completeness row. □

Theorem 8371.3 (IntegralTest Cauchy series handoff)**.** *Every accepted* $\mathsf{IntegralTest}^{\uparrow}$ *carrier hands convergence evidence to Real and Completion consumers only through the displayed route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{CauchyIntegralTest}^{\uparrow}$ *sibling supplies the narrower Cauchy-tail comparison corridor, while* $\mathsf{IntegralTest}^{\uparrow}$ *keeps the general textbook comparison surface visible as its own finite carrier. The handoff exports no selected infinite sum, selected improper integral, quotient stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier rows. The source series row S and comparison row F are not endpoint seals; they only expose finite tail and monotone comparison data.

Use Definition 8371.2. It places W, D, Q, E after the source and comparison rows, so every endpoint-facing read must pass through finite windows, dyadic budgets, and regular rational readback before the Real seal is visible.

The $\mathsf{CauchyIntegralTest}^{\uparrow}$ route is a compatible sibling because it supplies the Cauchy-tail corridor for the same kind of integral comparison. This chapter keeps the broader $\mathsf{IntegralTest}^{\uparrow}$ carrier distinct: the provenance row cites the sibling but the route is still exhausted by $S, F, W, D, Q, E, H, C, P, N$.

Since no carrier coordinate stores a selected sum, improper integral value, quotient stream equality, choice schedule, or ambient completeness principle, none can be exported by the handoff. □

Falsifiable prediction: An accepted $\mathsf{IntegralTest}^{\uparrow}$ packet that reaches a $\mathsf{Real}^{\uparrow}$ seal without first exposing the $\mathsf{RealSeries}^{\uparrow}$ source row, monotone comparison row, finite $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: cauchy-integral-test, realseries, regseqrat, dyadicratcore, real: no listed sibling supplies the whole packet $I = (S, F, W, D, Q, E, H, C, P, N)$, because each sibling contributes only a Cauchy-tail corridor, source-series row, regular readback, dyadic tolerance ledger, or terminal seal while this chapter binds them into one integral-test NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{IntegralTest}^{\uparrow}}$.

Constructive story: An $\mathsf{IntegralTest}^{\uparrow}$ packet is generated from a RealSeriesUp source row, a monotone comparison-function row, finite StreamNameUp windows, a DyadicRatCore tolerance ledger, RegSeqRatUp readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntegralTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8371.1, Definition 8371.2, and Definition 8371.3 seed the finite integral-test carrier, local NameCert obligations, and Cauchy-series handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntegralTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected improper integrals, selected infinite sums, quotient series equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone comparison exposure, finite-window exposure, dyadic tolerance confinement, regular readback, terminal seal scope, transport, replay, provenance, and local naming for BEDC.Derived.IntegralTestUp.

Chapter 8372

A Concrete Naming Certificate for $\mathsf{FiniteSelectionCorrespondence}^{\uparrow}$

Following Berge and Aliprantis–Border on finite correspondences and measurable selection language, the $\mathsf{FiniteSelectionCorrespondence}^{\uparrow}$ packet records only finite displayed selection rows. It sits beside the metric source of chapter 57, the compact metric ledger of chapter 129, the set-like membership surface of chapter 3668, the finite transport-plan route of chapter 5197, and the distribution ledger of chapter 186. The packet is a finite correspondence interface; it does not export a global selector, countable choice, measurable-selection theorem, optimizer, or ambient completion.

Definition 8372.1 (Finite selection correspondence carrier)**.** *A* $\mathsf{FiniteSelectionCorrespondence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, K, S, W, R, D, H, C, P, N)$$

with the following visible rows:

1. *M is the displayed* $\mathsf{Met}^{\uparrow}$ *source row;*
2. *K is an optional* $\mathsf{CompactMetric}^{\uparrow}$ *finite-support or finite-net ledger read by compact consumers;*
3. *S is the* $\mathsf{SetLike}^{\uparrow}$ *graph row listing finite source points and finite admissible value rows;*
4. *W is the displayed selection-window table, whose entries are finite pairs already present in S;*
5. *R is the readback row consumed by finite* $\mathsf{Wasserstein}^{\uparrow}$ *and* $\mathsf{Distribution}^{\uparrow}$ *interfaces;*
6. *D is the refusal ledger recording that no global selector, countable-choice schedule, optimizer, or measurable-selection theorem is exported;*
7. *H records componentwise* hsame *transport for the metric, compact, graph, window, readback, refusal, provenance, and naming rows;*
8. *C records displayed* Cont *replay from graph rows to finite windows and from windows to downstream finite consumers;*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSelectionCorrespondence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Wasserstein}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSelectionCorrespondence}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise by hsame *on these rows.*

Lean4 (statement) `BEDC.Derived.FiniteSelectionCorrespondenceUp`

Theorem 8372.2 (Finite selection correspondence obligation surface)**.** *For every accepted* FiniteSelectionCorrespon*carrier* $F = (M, K, S, W, R, D, H, C, P, N)$*, the local obligation surface consists exactly of the displayed metric source, compact-support ledger, set-like graph row, finite selection windows, downstream readback row, refusal ledger, componentwise transport, replay, provenance, and local naming rows. Any finite transport, compact metric, or distribution consumer must read* S *and* W *before consuming* R*, and* D *remains visible on every route, so no choice principle or selected global function is exported.*

Proof. Project the carrier of Definition 8372.1. The only admissible correspondence data are the finite graph row S and the finite selection-window table W over the displayed metric source M and optional compact row K.

The downstream readback row R is attached to those finite rows. A Wasserstein$^{\uparrow}$ or Distribution$^{\uparrow}$ consumer may therefore receive a finite support or finite provenance route, but it must receive it through the listed S, W, R coordinates rather than through a hidden selector.

Finally read the structural rows. The componentwise hsame row H, the displayed Cont replay row C, the Pkg provenance P, and the local NameCert row N preserve the same finite route, while D records the refusal of global selection, countable choice, optimizer extraction, and measurable-selection principles. Hence the displayed rows exhaust the finite selection correspondence obligation surface. □

Falsifiable prediction: An accepted FiniteSelectionCorrespondence$^{\uparrow}$ consumer that reads a selected value without first exposing the finite graph row S and finite selection window W, or that exports a global selector after reading D, is outside the carrier surface.

Independence witness: metric, compactmetric, setlike, wasserstein, distribution: no listed sibling supplies the joint finite correspondence packet with displayed metric source, compact-support ledger, set-like graph row, finite selection windows, transport and distribution readback, refusal ledger, componentwise transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{FiniteSelectionCorrespondence}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{FiniteSelectionCorrespondence}^{\uparrow}}$, seed)

Scope closed: Definition 8372.1 and Definition 8372.2 seed the finite displayed correspondence interface over Metric, CompactMetric, SetLike, Wasserstein, Distribution, transport, replay, provenance, and local NameCert rows.

Constructive story: A FiniteSelectionCorrespondence$^{\uparrow}$ packet is generated from a displayed Metric source row, optional CompactMetric finite-support row, SetLike graph row, finite selection-window table, downstream readback row, refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Formal verification: FormalStatus(NameCert$_{\text{FiniteSelectionCorrespondence}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.FiniteSelectionCorrespondenceUp`

Bridge status: none

Not claimed: This chapter does not close global selectors, measurable selection, optimizer existence, countable choice, ambient measure completion, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite graph-window exhaustion, downstream readback preservation, refusal-ledger nonescape, Nontrivial, and FieldFaithful instances for BEDC.Derived.FiniteSelectionCorrespondenceUp.

Chapter 8373

A Concrete Naming Certificate for $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges-style completion discipline, the $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$ packet records a separated Cauchy completion as displayed regular rational names with an explicit separation seal. It sits after the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the completion interface of chapter 95, the separated-completion surface of chapter 1612, and the terminal real-name seal of chapter 38. The packet is quotient-free: equality is read through displayed regular windows and separation evidence rather than through an ambient quotient of Cauchy streams.

Definition 8373.1 (SeparatedCauchyCompletion carrier)**.** *A* $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, D, C, E, Q, H, T, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row for the regular rational Cauchy name.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which consumers inspect R.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for window comparison and separation tests.*
4. *C is the* $\mathsf{Completion}^{\uparrow}$ *handoff row recording that the displayed name is completion-facing.*
5. *E is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *seal that turns zero-distance comparison into displayed sameness without quotienting streams.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row reached only through R, W, D, C, E.*
7. *H records componentwise* hsame *transport for the regular name, schedule, dyadic ledger, completion row, separated seal, and real readback.*
8. *T records displayed* Cont *replay for the same finite route.*
9. *P is the* Pkg *provenance over* $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient class of Cauchy streams, a selected host limit, countable choice, trichotomy, or ambient completeness outside the displayed route.

Theorem 8373.2 (SeparatedCauchyCompletion NameCert obligations)**.** *For every accepted* SeparatedCauchyComp *carrier* $S = (R, W, D, C, E, Q, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, regular rational source exposure through* R*, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through* D*, completion handoff through* C*, separated-completion sealing through* E*, terminal* $\mathsf{Real}^{\uparrow}$ *readback through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* T*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The displayed rows of Definition 8373.1 are precisely the regular rational source, finite window schedule, dyadic tolerance ledger, completion handoff, separated seal, real readback, transport, replay, provenance, and local naming rows.

The separation step is finite because it can only read the windows named by W and the dyadic tolerances recorded in D. The completion row C therefore receives a regular rational name together with its explicit tolerance route, not an ambient quotient object.

The separated seal E is downstream of that route, and Q can be read only after E has been exposed. The structural rows H, T, P, N preserve the same route under transport, replay, sourcing, and naming, so the local certificate contains exactly the listed obligations. □

Theorem 8373.3 (SeparatedCauchyCompletion regular-sequence scope)**.** *Every completion-facing consumer of an accepted* $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow} \longmapsto \mathsf{SeparatedCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *row supplies the displayed regular name, the* $\mathsf{StreamName}^{\uparrow}$ *row supplies finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row supplies comparison tolerances, the* $\mathsf{Completion}^{\uparrow}$ *row records the completion-facing handoff, the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row records the separation seal, and the* $\mathsf{Real}^{\uparrow}$ *row is the only terminal readback. The route exports no quotient stream, selected host limit, countable-choice schedule, trichotomy, or ambient completeness theorem.*

Proof. Begin with the carrier rows. The source row R and window row W expose only finite pieces of the regular rational name, while D records the dyadic comparisons available at those windows.

Use Definition 8373.2. It places the completion handoff C after R, W, D and the separated seal E after the completion-facing data, so a consumer cannot reach the Real readback before the displayed separation evidence is visible.

Finally read Q through the structural rows H, T, P, N. Since the carrier has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, or ambient completeness, the regular-sequence scope is exhausted by the displayed route. □

Falsifiable prediction: An accepted $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$ packet whose $\mathsf{Real}^{\uparrow}$ readback is consumed without exposing the $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{Completion}^{\uparrow}$ handoff, and $\mathsf{SeparatedCompletion}^{\uparrow}$ seal refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, completion, separatedcompletion, real: no listed sibling supplies the whole packet $S = (R, W, D, C, E, Q, H, T, P, N)$, because each sibling contributes only a regular name, finite windows, dyadic tolerances, completion handoff, separation seal, or terminal real readback while this chapter binds them into one quotient-free NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeparatedCauchyCompletion}^{\uparrow}$ packet is generated from a RegSeqRatUp source row, StreamNameUp finite windows, a DyadicRatCore tolerance ledger, a CompletionUp handoff, a SeparatedCompletionUp seal, a

RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SeparatedCauchyCompletion}\uparrow}$, seed)

Scope closed: Definition 8373.1, Definition 8373.2, and Definition 8373.3 seed the quotient-free separated Cauchy completion carrier, local NameCert obligations, and regular-sequence consumer scope.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparatedCauchyCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular rational source exposure, finite-window exposure, dyadic tolerance confinement, completion handoff, separated seal scope, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCauchyCompletionUp.

Chapter 8374

A Concrete Naming Certificate for CompactCoverOrderBound$^{\uparrow}$

Following the constructive Heine–Cantor cover discipline, the CompactCoverOrderBound$^{\uparrow}$ packet records a finite overlap bound for the compact-uniform-continuity route. It sits beside the compact source certificate of chapter 60, the uniform-continuity carrier of chapter 388, the finite star-refinement route of chapter 6997, the mesh-chain cover route of chapter 7769, the local-modulus nerve of chapter 8353, and the tolerance-only handoff of chapter 369. The packet names bounded-overlap cover rows before a uniform modulus is read; it does not choose an optimal cover, a least Lebesgue number, or an ambient compactness principle.

Definition 8374.1 (CompactCoverOrderBound carrier)**.** *A* CompactCoverOrderBound$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, C, S, M, B, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the* Compact$^{\uparrow}$ *source row exposing the finite cover only through displayed compact data.*
2. *C is the finite cover row, read as bundled centers, radii, or cells rather than as an ambient open-cover object.*
3. *S is the star-refinement and mesh-chain row that records which cover cells may meet after refinement.*
4. *M is the pointwise local-modulus row consumed by compact-uniform-continuity arguments.*
5. *B is the finite cover-order bound row: for each displayed cell, it lists the finite family of neighboring cells allowed to overlap it.*
6. *U is the downstream* UniformModulus$^{\uparrow}$ *handoff row reached through the bound B and the local-modulus row M.*
7. *H records componentwise* hsame *transport for the compact source, cover, refinement, local-modulus, bound, and uniform-modulus rows.*
8. *T records displayed* Cont *replay for the same finite cover-order route.*
9. *P is the* Pkg *provenance over* CompactCoverOrderBound$^{\uparrow}$, Compact$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, FiniteCoverStarRefinementModulus$^{\uparrow}$, FiniteCoverMeshChain$^{\uparrow}$, CompactUniformLocalModulusNerve$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactCoverOrderBound}^{\uparrow}}$ *row.*
The carrier has no coordinate for an optimal subcover, a least Lebesgue number, an infinite locally finite family, countable choice, or an external compactness oracle.

Theorem 8374.2 (CompactCoverOrderBound NameCert obligations)**.** *For every accepted* CompactCoverOrderBou *carrier $O = (K, C, S, M, B, U, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactCoverOrderBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, compact source exposure through K, finite cover exposure through C, star-refinement and mesh-chain exposure through S, local-modulus exposure through M, finite cover-order bound exposure through B, uniform-modulus handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through T, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows of Definition 8374.1 are precisely the compact source, finite cover, refinement mesh, local-modulus, order-bound, uniform-modulus, transport, replay, provenance, and naming rows.

The cover-order row is finite because B records only the displayed neighboring cells allowed by C and S. No proof step ranges over an arbitrary open cover or over an infinite locally finite family.

The uniform-modulus handoff U is downstream of the local-modulus row M and the finite overlap bound B. The structural rows H, T, P, N preserve the same bounded-overlap route, so the local certificate contains exactly the listed obligations. □

Theorem 8374.3 (CompactCoverOrderBound uniform-modulus scope)**.** *Every uniform-modulus consumer of an accepted* $\mathsf{CompactCoverOrderBound}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{Compact}^{\uparrow} \longmapsto \mathsf{FiniteCoverStarRefinementModulus}^{\uparrow} \longmapsto \mathsf{FiniteCoverMeshChain}^{\uparrow} \longmapsto \mathsf{CompactUniformLocalModulusNe}$$

The cover-order bound row B is consumed before the $\mathsf{UniformModulus}^{\uparrow}$ *handoff U, and it confines the handoff to bounded-overlap finite cover rows. The route exports no optimal cover, least Lebesgue number, infinite cover selection, countable-choice schedule, or external compactness oracle.*

Proof. Begin with the compact source row K and the finite cover row C. These rows expose a displayed finite family, not an arbitrary open-cover quantifier.

Use Definition 8374.2. The obligation surface places the star-refinement row S, local-modulus row M, and cover-order bound row B before any uniform-modulus read is available.

Finally read U through H, T, P, N. The transport and replay rows preserve the same finite neighboring-cell list, while provenance and naming keep the route tied to the displayed carrier. Since no other cover-selection coordinate exists, the uniform-modulus scope is exactly the bounded-overlap finite cover route. □

Falsifiable prediction: An accepted $\mathsf{CompactCoverOrderBound}^{\uparrow}$ packet whose $\mathsf{UniformModulus}^{\uparrow}$ handoff is consumed without first exposing the finite cover row, star-refinement row, local-modulus row, and cover-order bound row refutes the carrier surface.

Independence witness: compact, compactuniformcontinuity, finite-cover-star-refinement-modulus, finite-cover-mesh-chain, compact-uniform-local-modulus-nerve, uniformmodulus: no listed sibling supplies the whole packet $O = (K, C, S, M, B, U, H, T, P, N)$, because each sibling contributes only compact source data, a uniform-continuity route, star-refinement data, mesh-chain data, local-modulus nerve data, or terminal uniform-modulus handoff while this chapter binds them into one bounded-overlap NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCoverOrderBound}^{\uparrow}}$.

Constructive story: A CompactCoverOrderBound$^{\uparrow}$ packet is generated from a CompactUp source row, finite cover rows, star-refinement and mesh-chain rows, a local-modulus row, a finite cover-order bound row, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactCoverOrderBound}^{\uparrow}}$, seed)

Scope closed: Definition 8374.1, Definition 8374.2, and Definition 8374.3 seed the bounded-overlap finite cover carrier, local NameCert obligations, and uniform-modulus consumer scope.

Formal verification: FormalStatus(NameCert$_{\text{CompactCoverOrderBound}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close optimal finite covers, least Lebesgue numbers, infinite cover selection, countable choice, ambient compactness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, finite cover exposure, refinement exposure, local-modulus exposure, cover-order bound confinement, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactCoverOrderBoundUp.

Chapter 8375

A Concrete Naming Certificate for $\mathsf{DyadicCutStability}^{\uparrow}$

Bishop-style located cut stability asks that a located lower-and-upper cut presentation remain stable when finite dyadic tolerance rows, regular rational readback, and the terminal real-name seal are replayed through the same certificate surface. The $\mathsf{DyadicCutStability}^{\uparrow}$ packet records that stability as finite BEDC data. It sits after the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Dedekind real carrier of chapter 3629, and the located cut real carrier of chapter 3773. The packet names a stability route for displayed cuts; it does not assert Dedekind completeness, quotient equality of real names, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8375.1 (Dyadic cut stability carrier)**.** *A* $\mathsf{DyadicCutStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the rounded lower-cut row, recording finite lower witnesses and the rounded-lower readback used by a located cut consumer.*
2. *U is the rounded upper-cut row, recording finite upper witnesses and the rounded-upper readback used by the same consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback row, storing dyadic width requests, comparison windows, and the finite tolerance ledger linking L and U.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row, translating the dyadic ledger into regular rational observations before the real seal is exposed.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, D, Q have been displayed.*
6. *H stores componentwise* hsame *transport for the rounded lower cut, rounded upper cut, dyadic ledger, regular-rational witness, real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from located-cut inspection through L, U, D, Q, R, H.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCutStability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DedekindReal}^{\uparrow}$, $\mathsf{LocatedCutReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCutStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, Q, R, H, C, P, N$. It has no coordinate for a host Dedekind supremum, quotient equality of real names, a selected cut representative, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8375.2 (Dyadic cut stability NameCert obligations)**.** *For an accepted* $\mathsf{DyadicCutStability}^{\uparrow}$ *packet $S = (L, U, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicCutStability}^{\uparrow}}$ *obligations are exhausted by rounded lower-cut exposure through L, rounded upper-cut exposure through U,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness readback through Q,* $\mathsf{Real}^{\uparrow}$ *sealing through R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8375.1. The accepted packet displays exactly the lower cut, upper cut, dyadic tolerance, regular rational witness, real seal, transport, replay, provenance, and naming rows.

Read the cut rows first. The lower and upper rows are rounded before the dyadic tolerance row is consumed, so a located-cut consumer receives both sides of the cut as finite data rather than as a detached real equality claim.

Next use D, Q, R. The dyadic ledger records the finite comparison windows, the regular rational row reads those windows as regular observations, and the real seal is reached only after those rows have been displayed.

Finally apply H, C, P, N. These rows preserve and replay the same packet under componentwise transport, package the cited sibling chapters, and expose the local name. Since the carrier has no further coordinate, the listed rows exhaust the local $\mathsf{NameCert}$ obligations. □

Theorem 8375.3 (Dyadic cut stability transport)**.** *Every* $\mathsf{DyadicCutStability}^{\uparrow}$ *transport read preserves the rounded lower-cut row L, rounded upper-cut row U, dyadic tolerance row D, regular rational witness row Q, real seal row R, componentwise* hsame *row H, displayed* Cont *replay row C, provenance row P, and local* $\mathsf{NameCert}$ *row N before any located-cut consumer reads stability. The transport exports no Dedekind completeness theorem, quotient real equality, trichotomy, selected representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the NameCert obligations of Definition 8375.2. They force every accepted read to display $L, U, D, Q, R, H, C, P, N$ before the local name is consumed.

Use the sibling route recorded in P. The dyadic tolerance core supplies D, the regular rational chapter supplies Q, the real chapter supplies R, and the Dedekind and located-cut chapters supply the cut-facing provenance for L and U.

Transport is then componentwise through H and replayed through C. These rows can preserve the displayed lower and upper cut data together with their dyadic and regular-rational readback, but they cannot create a new coordinate for a host completion or quotient equality.

Thus every located-cut stability consumer reads the same finite packet before it reaches the stability conclusion. The excluded completeness, equality, trichotomy, representative, choice, and ambient real-completion claims are not carrier rows. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCutStability}^{\uparrow}$ read that omits a rounded lower cut, rounded upper cut, dyadic tolerance ledger, RegSeqRat witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local NameCert row is outside the carrier surface.

Independence witness: dyadicratcore, regseqrat, real, dedekind-real, located-cut-real: none is bijective to DyadicCutStabilityUp because the packet jointly carries rounded lower and upper

cut rows, dyadic tolerance/readback, regular rational witness, Real seal, hsame transport, Cont replay, Pkg provenance, and one local NameCert row while refusing quotient equality and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCutStability}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCutStability}^{\uparrow}$ packet is generated from rounded lower-cut rows, rounded upper-cut rows, DyadicRatCore tolerance and readback rows, RegSeqRat witness rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCutStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8375.1, Definition 8375.2, and Definition 8375.3 seed the finite located-cut stability route over DyadicRatCore, RegSeqRat, Real, Dedekind real, located cut real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCutStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Lean verification, bridge checking, Dedekind completeness, quotient equality of real names, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked rounded lower-cut exposure, rounded upper-cut exposure, DyadicRatCore tolerance/readback, RegSeqRat witness readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCutStabilityUp.

Chapter 8376

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$

Following Bishop and Bridges constructive completion theory and the Kelley presentation of completion as a reflective boundary, the $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$ packet records the finite retraction surface from displayed Cauchy-completion data back to regular Cauchy real names. It sits after the completion retraction packet of chapter 1921, the finite name windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only a finite NameCert retraction route; it does not assert quotient completion, selected representatives, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8376.1 (Regular Cauchy completion retraction carrier)**.** *A* $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, W, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ *source row, exposing the completion-facing retraction request before any regular readback is consumed.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the retraction.*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the finite regularity and comparison budgets.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, W, S, D have been displayed.*
6. *H records componentwise* hsame *transport for the retraction row, windows, readback, dyadic ledger, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-facing read back through K, W, S, D, E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$, $\mathsf{CauchyCompletionRetraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8376.2 (Regular Cauchy completion retraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$ *carrier*

$$T = (K, W, S, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}}$ *surface consists exactly of completion-retraction admission through* K*, finite StreamName window exposure through* W*, RegSeqRat readback through* S*, DyadicRatCore tolerance comparison through* D*, Real seal nonescape through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8376.1. The analytic rows are exactly K, W, S, D, E, while H, C, P, N transport, replay, source, and name that same finite packet.

The retraction row is read first. Row K supplies the completion-facing retraction request, but it is not a quotient-completion equality or a selected representative.

The regular source rows then control the readback. Rows W, S, D expose finite stream windows, regular rational data, and dyadic tolerance budgets before the terminal seal row E is available.

Finally H, C, P, N preserve the same route. Since the excluded quotient, representative, choice, host-equality, and ambient-completeness objects have no carrier coordinate, none can become a local NameCert obligation. □

Theorem 8376.3 (Regular Cauchy completion retraction Real seal route)**.** *Every accepted* RegularCauchyCompletic *completion-facing read reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$\mathsf{CauchyCompletionRetraction}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a Real or Completion consumer reads the retraction as finite regular-Cauchy NameCert data through $K, W, S, D, E, H, C, P, N$*, not as quotient completion, a selected representative, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 8376.2. It places every exported retraction read inside the rows $K, W, S, D, E, H, C, P, N$.

The source row K attaches the route to chapter 1921. The window and readback rows W, S attach it to the finite StreamName and RegSeqRat surfaces of chapter 126 and chapter 301.

The dyadic ledger D supplies the finite tolerance comparison used before the terminal seal E is consumed. The structural rows only transport, replay, source, and name that route.

Therefore every accepted consumer reaches $\mathsf{Real}^{\uparrow}$ through the displayed regular-Cauchy surface and cannot replace it by quotient completion equality, selected representatives, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}$ packet that exposes a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying the $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger refutes the carrier surface.

Independence witness: cauchy-completion-retraction, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole finite packet $T = (K, W, S, D, E, H, C, P, N)$, because each sibling contributes only the retraction request, finite windows, regular readback, tolerance comparison, or terminal seal while this chapter binds them into one regular-Cauchy completion retraction surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionRetraction}^{\uparrow}}$**.**

Constructive story: A RegularCauchyCompletionRetraction$^{\uparrow}$ packet is generated from a CauchyCompletionRetraction source row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance comparison, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionRetraction}^{\uparrow}}$, seed)

Scope closed: Definition 8376.1, Definition 8376.2, and Definition 8376.3 seed the finite retraction route from Cauchy completion data to regular Cauchy real-name data.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionRetraction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality of completed objects, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-retraction admission, finite-window exposure, regular readback, dyadic tolerance comparison, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionRetractionUp.

Chapter 8377

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and explicit tail estimates, the $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ packet records the finite ledger used when a fixed tail-window budget is reparametrized before regular rational readback. It sits beside the finite schedules of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the cofinal reindexing route of chapter 6960, the tail-map route of chapter 7209, and the subsequence surface of chapter 381. The packet names only a fixed finite tail-window reparametrization ledger; it is not quotient stream equality, not a selected subsequence choice, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8377.1 (Regular Cauchy tail reparametrization carrier)**.** *A* $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, W, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite regular-Cauchy observations available before the reparametrization is read.*
2. *W is the target tail-window row. It records the finite target tail budget and the source tail window to which the current request is sent.*
3. *M is the monotone finite reparametrization ledger. It records the displayed order comparisons and target-to-source request map for the fixed window W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the precision budget, slack conversions, and tail-window comparison data used by M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after S, W, M, D have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular rational readback has consumed the fixed reparametrized tail window.*
7. *H records componentwise* hsame *transport for the source window, target tail-window, monotone ledger, dyadic tolerance row, readback row, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail precision request through W, then M, then S, then D, R, E.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyReindexing}^{\uparrow}$, $\mathsf{CofinalReindexing}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailReparametrization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected subsequence choice, trichotomy of reals, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8377.2 (Regular Cauchy tail reparametrization NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ *carrier*

$$T = (S, W, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailReparametrization}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{StreamName}^{\uparrow}$ *window, target tail-window, monotone finite reparametrization ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. The obligations require every readback request to pass through the fixed target tail-window and its monotone finite ledger before any dyadic tolerance, regular rational value, or Real seal is consumed.*

Proof. Project the carrier of Definition 8377.1. The rows S, W, M, D, R, E are the complete analytic route, while H, C, P, N only transport, replay, source, and name that route.

The target tail-window W is fixed before the reparametrization ledger is read. The row M supplies the finite monotone target-to-source request map for that window, so the packet does not choose a subsequence or appeal to a global cofinal selector.

After W and M have fixed the inspected source window, the dyadic tolerance row D supplies the precision budget for the regular rational readback R. The terminal seal E is downstream of the same displayed data, so no quotient stream equality, selected subsequence choice, trichotomy, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness row can be exported. □

Theorem 8377.3 (Regular Cauchy tail reparametrization fixed-window route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto M \longmapsto S \longmapsto D \longmapsto R \longmapsto E.$$

The route is a fixed tail-window specialization of the sibling reindexing and subsequence packets: it keeps the source $\mathsf{StreamName}^{\uparrow}$ *window, target tail-window, finite monotone ledger, dyadic tolerance row, regular rational readback, and terminal Real seal visible. It exports no quotient stream equality, selected subsequence choice, trichotomy, host real equality, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the seal row E. The NameCert obligations of Definition 8377.2 place E after R, and place R after the source window S, dyadic tolerance row D, target tail-window W, and monotone finite ledger M.

The sibling chapters identify the surrounding interfaces. The $\mathsf{RegularCauchyReindexing}^{\uparrow}$ and $\mathsf{CofinalReindexing}^{\uparrow}$ routes expose cofinal reindexing data, while $\mathsf{RegularCauchySubsequence}^{\uparrow}$ exposes subsequence-facing monotone-cofinal data. This packet uses only the fixed-window part of that surface, so it has no coordinate from which a selected subsequence or quotient stream equality could be read.

The structural rows H, C, P, N preserve the same finite route. Therefore every Real-facing read factors through the displayed ledger and stops at the terminal seal without adding trichotomy, host equality, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading $W, M, S, D, R, E, H, C, P, N$ refutes the fixed tail-window carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-reindexing, cofinal-reindexing, regular-cauchy-subsequence: no listed sibling supplies the joint packet containing one source StreamName window, one fixed target tail-window, one monotone finite reparametrization ledger, one dyadic tolerance row, one RegSeqRat readback, one Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailReparametrization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailReparametrization}^{\uparrow}$ packet is generated from a source StreamName window, a target tail-window, a monotone finite reparametrization ledger, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailReparametrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8377.1, Definition 8377.2, and Definition 8377.3 seed the fixed tail-window reparametrization route from displayed StreamName windows through dyadic tolerance and RegSeqRat readback to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailReparametrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequence choice, trichotomy, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fixed target tail-window exposure, monotone finite ledger stability, dyadic tolerance exactness, RegSeqRat readback ordering, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailReparametrizationUp.

Chapter 8378

A Concrete Naming Certificate for $\mathsf{SturmSeparation}^{\uparrow}$

Following Sturm and Coddington–Levinson on zero separation for scalar second-order equations, the $\mathsf{SturmSeparation}^{\uparrow}$ packet records the finite BEDC surface by which two displayed solution-window rows witness alternating zero windows before any global solution theorem is used. It sits beside the real-name boundary of chapter 38, the regular rational readback of chapter 301, the dyadic arithmetic ledger of chapter 291, the local ODE source of chapter 193, the interval carrier of chapter 56, and the continuous-map handoff of chapter 130. It also consumes the Sturm sequence, comparison, and root-count siblings of chapter 2079, chapter 2338, and chapter 3612. The packet names separation data as displayed rows only; it is not global ODE completeness, a selected classical solution curve, quotient real equality, sign trichotomy, or hidden choice.

Definition 8378.1 (Sturm separation carrier)**.** *A* $\mathsf{SturmSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, I, F, G, Z, A, B, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *source row for the scalar second-order equation and its local solution-window request.*
2. *I is the* $\mathsf{Interval}^{\uparrow}$ *row naming the inspected compact interval and its left, middle, and right subwindows.*
3. *F and G are two* $\mathsf{ContinuousMap}^{\uparrow}$ *solution-observation rows over the interval window.*
4. *Z is the zero-window ledger, recording the displayed zeros of F and the adjacent zero-search windows for G.*
5. *A is the alternation ledger, recording which G zero-window is read between adjacent F zero-windows and which F zero-window is read between adjacent G zero-windows.*
6. *B is the* $\mathsf{SturmComparison}^{\uparrow}$ *coefficient and comparison handoff row.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger for the zero windows and comparison inequalities.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint, coefficient, and solution-observation windows.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after $O, I, F, G, Z, A, B, D, R$ are displayed.*
10. *H records componentwise* hsame *transport for the ODE, interval, solution rows, zero ledgers, comparison rows, dyadic ledger, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a separation request through the ODE source, interval window, solution observations, zero ledgers, comparison handoff, dyadic and regular readbacks, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{SturmSeparation}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Interval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, $\mathsf{SturmComparison}^{\uparrow}$, $\mathsf{SturmRootCount}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for global ODE completeness, host solution equality, excluded-middle sign trichotomy, quotient real equality, or a selected classical solution curve.

Theorem 8378.2 (Sturm separation NameCert obligations)**.** *For every accepted* $\mathsf{SturmSeparation}^{\uparrow}$ *carrier*

$$S = (O, I, F, G, Z, A, B, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SturmSeparation}^{\uparrow}}$ *surface consists exactly of carrier admission, ODE-source exposure, interval-window exposure, two continuous solution-observation rows, zero-window exposure, alternation-ledger exposure, Sturm comparison handoff, dyadic endpoint and tolerance accounting, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8378.1. The rows O, I, F, G are the only displayed source rows, so a separation read cannot inspect a host solution space before the local ODE, interval, and continuous-map observations are opened.

Read the zero data through Z and the alternation data through A. These rows record finite adjacent windows rather than an ambient zero set or a trichotomy oracle.

Use the Sturm comparison handoff row B together with the dyadic and regular readback rows D, R. They place coefficient comparisons, endpoint tolerances, and solution observations on the same finite windows before the terminal seal E is available.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no displayed coordinate names global completeness, host solution equality, quotient real equality, hidden choice, or selected classical curves, those objects are not local NameCert obligations. □

Theorem 8378.3 (Sturm separation zero interlacing route)**.** *Every accepted* $\mathsf{SturmSeparation}^{\uparrow}$ *zero-interlacing read factors through the displayed route*

$$O, I, F, G \longmapsto Z, A \longmapsto B, D, R \longmapsto E \longmapsto H, C, P, N.$$

The route exposes one local ODE source, one inspected interval, two continuous solution-observation rows, finite zero-window and alternation ledgers, the Sturm comparison handoff, dyadic endpoint accounting, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming. It exports no global ODE completeness theorem, selected solution curve, quotient real equality, sign trichotomy, or hidden choice principle.

Proof. Start with a consumer of zero separation. The ODE source row O and interval row I must be displayed before any solution observation is read, and the two solution rows F, G are available only as $\mathsf{ContinuousMap}^{\uparrow}$ observations over that interval.

The zero-window ledger Z then names the inspected adjacent windows. The alternation ledger A records the finite interlacing claim by pointing from each displayed zero-window of one solution row to the displayed search window of the other.

Apply the Sturm comparison handoff B with the dyadic and regular readback rows D, R. This binds coefficient comparisons, endpoint tolerances, and solution-window readbacks to the same finite route before the $\mathsf{Real}^{\uparrow}$ seal E is readable.

The structural rows H, C, P, N preserve and name that route. Hence every accepted zero-interlacing read is the finite packet above, with no appeal to global completeness, selected solution curves, quotient real equality, sign trichotomy, or hidden choice. □

Falsifiable prediction: An accepted $\mathsf{SturmSeparation}^{\uparrow}$ consumer that reads zero separation without first displaying $O, I, F, G, Z, A, B, D, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: real, regseqrat, dyadicratcore, ode, interval, continuousmap, sturmsequence, sturmcomparison, sturmrootcount: no listed sibling supplies the joint packet containing one ODE source, one interval window, two continuous solution-observation rows, zero-window and alternation ledgers, Sturm comparison handoff, dyadic endpoint accounting, regular rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SturmSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{SturmSeparation}^{\uparrow}$ packet is generated from an OdeUp source row, an IntervalUp window, two ContinuousMap solution-observation rows, zero-window and alternation ledgers, a SturmComparison handoff, DyadicRatCore endpoint accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8378.1, Definition 8378.2, and Definition 8378.3 seed the finite zero-interlacing route for displayed scalar ODE solution windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical global ODE completeness, selected solution curves, quotient real equality, sign trichotomy, hidden choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ODE and interval exposure, continuous solution-window exposure, zero-window exactness, alternation-ledger exactness, Sturm comparison handoff, dyadic and regular readback discipline, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SturmSeparationUp.

Chapter 8379

A Concrete Naming Certificate for MetaCICSubjectReductionResidualFence$^{\uparrow}$

The MetaCICSubjectReductionResidualFence$^{\uparrow}$ packet realizes the dependent-codomain obstruction frontier of chapter 9377 as finite discharge, residual, normalization-frontier, obstruction-retention, transport, replay, provenance, and local naming rows. It consumes the obstruction-retention row of chapter 731 and the normalization-frontier rows of chapter 892 before any subject-reduction consumer can read the fence; it remains adjacent to the discharge route of chapter 718, the closed application argument-stability boundary of chapter 6177, and the beta-substitution preservation packet of chapter 716. The packet names the residual fence that remains after the independent-codomain application case has been separated from the dependent-codomain counterexample; it does not claim closed subject reduction, discharge any of the four hypotheses, or hide the obstruction boundary.

Definition 8379.1 (MetaCIC subject-reduction residual fence carrier)**.** *A* MetaCICSubjectReductionResidualFence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, R, K, O, T, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the discharge-interface row, naming the four simultaneous subject-reduction obligations* BetaSubstitutionPreservation*,* AppArgTypeStable*,* LamDomainSubjectReduction*, and* PiDomainSubjectRedu
2. *R is the residual dependent-codomain row, retaining the application-argument reduction case where substituting the reduct into the codomain changes the displayed result type.*
3. *K is the candidate-normalization frontier row, recording the closed-normal and candidate evidence that remains usable without converting the residual fence into a subject-reduction theorem.*
4. *O is the obstruction-retention row, carrying the checked dependent-codomain counterexample and the unavailable closed subject-reduction statement as visible boundary data.*
5. *T is the transport row from closed application argument stability, beta-substitution preservation, subject-reduction discharge, and normalization-frontier siblings into this fence.*
6. *Q is the replay row, recording the finite route from the obstruction map through candidate normalization and back to the four-obligation discharge interface.*
7. *H records componentwise* hsame *transport for the discharge, residual, candidate-normalization, obstruction, transport, replay, provenance, and naming rows.*

8. *C records displayed* Cont *replay for the fence route.*
9. *P is the* Pkg *provenance over* MetaCICSubjectReductionResidualFence$^{\uparrow}$, SubjectReductionDischarge$^{\uparrow}$, BetaSubstitutionPreservation$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, ClosedAppArgumentStability$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose residual fence, obstruction-retention row, transport route, replay route, provenance, and local naming agree componentwise. It has no coordinate for unconditional closed subject reduction, a discharged dependent-codomain theorem, quotient syntax equality, hidden normalization, or a silent proof of any missing discharge row.

Theorem 8379.2 (MetaCIC subject-reduction residual fence NameCert obligations)**.** *For every accepted* MetaCICSubjectReductionResidualFence$^{\uparrow}$ *carrier*

$$F = (D, R, K, O, T, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}}$ *surface consists exactly of four-obligation discharge exposure through D, dependent-codomain residual exposure through R, candidate-normalization frontier exposure through K, obstruction retention through O, sibling transport through T, replay through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8379.1. The rows D, R, K, O are the only mathematical rows available before transport and replay are read, so a consumer cannot extract closed subject reduction from the packet.

Read the residual row R against the obstruction row O. The residual row retains the dependent-codomain application boundary, while O keeps the checked counterexample and the unavailable closed statement visible.

Read the candidate-normalization row K and transport row T. They allow the already checked normalization-side evidence and sibling discharge routes to be cited only as finite rows, not as replacements for the four simultaneous discharge obligations.

Finally Q, H, C, P, N replay, transport, source, and name the same fence. Since no carrier coordinate discharges the missing hypotheses or erases the obstruction, the listed rows exhaust the local NameCert obligations. □

Theorem 8379.3 (MetaCIC subject-reduction residual fence non-escape)**.** *Every subject-reduction consumer of an accepted* MetaCICSubjectReductionResidualFence$^{\uparrow}$ *packet receives the finite route*

$$D, R, K, O, T, Q, H, C, P, N.$$

The route may cite subjectreductiondischarge, metacic-subject-reduction-obstruction, metacic-normalization-frontier, closed-app-argument-stability, and betasubstitutionpreservation only as displayed rows. It cannot be consumed as an unconditional subject-reduction theorem, a closed dependent-codomain preservation proof, a completed candidate-normalization proof of subject reduction, or an erased obstruction boundary.

Proof. Start with the discharge row D. It names the four simultaneous inputs consumed by the parameterised subject-reduction theorem and does not identify them with one another.

The residual and obstruction rows R, O are then read together. They retain the dependent-codomain counterexample and prevent the independent-codomain application case from being treated as the full application-argument congruence discharge.

Use the candidate-normalization and transport rows K, T. These rows connect the fence to the normalization frontier, closed application argument-stability boundary, beta-substitution preservation packet, and subject-reduction discharge route, while keeping every cited object as a finite row.

The replay and structural rows Q, H, C, P, N preserve that same packet. Thus the accepted consumer receives a residual fence, not a hidden proof of closed subject reduction. □

Theorem 8379.4 (MetaCIC residual fence sibling route)**.** *Any residual-fence read of an accepted* $\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ *packet that reaches the audit-synthesis rows of chapter 776 must first pass through the subject-reduction boundary of chapter 721. The route remains local to the fence: it records the order in which audit-synthesis data may be consumed, and it does not assert closed subject reduction, dependent-codomain preservation, candidate normalization as a preservation theorem, or discharge of the four simultaneous hypotheses.*

Proof. Project the accepted fence carrier $F = (D, R, K, O, T, Q, H, C, P, N)$. The residual row R and obstruction-retention row O are visible before any audit-synthesis consumer can read the replay row.

Read the subject-reduction boundary sibling first. The boundary chapter supplies the finite place where the subject-reduction obstruction is exposed as boundary data rather than as a completed theorem, so the fence can hand R, O to downstream audit rows only after that boundary is visible.

Then read the audit-synthesis packet. Its rows may summarize the surrounding MetaCIC audit surface, but within this fence they are consumed through T, Q, H, C, P, N as transport, replay, provenance, and naming data for the same residual route.

Thus a consumer reaching audit-synthesis data has already passed through the subject-reduction boundary. Since no carrier coordinate erases R, O or discharges D, the route remains a local dependency route rather than a proof of subject reduction. □

Falsifiable prediction: A MetaCIC consumer that reads $\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ as closed subject reduction while omitting any of subjectreductiondischarge, metacic-subject-reduction-obstruction, metacic-normalization-frontier, closed-app-argument-stability, or betasubstitutionpreservation refutes the residual-fence classifier.

Independence witness: subjectreductiondischarge, metacic-subject-reduction-obstruction, metacic-normalization-frontier, closed-app-argument-stability, betasubstitutionpreservation: no listed sibling is bijective to $\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ because this packet jointly carries discharge, residual, candidate-normalization, obstruction-retention, transport, replay, provenance, and local naming rows.

The paired Lean surface for this packet must provide $\mathsf{Nontrivial}\ \mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ instances in a `TasteGate.lean` file before any verified TasteGate route can be claimed.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}$ packet is generated from a four-obligation discharge row, a dependent-codomain residual row, a candidate-normalization frontier row, an obstruction-retention row, sibling transport, replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8379.1, Definition 8379.2, and Definition 8379.3 seed the finite residual-fence route for the dependent-codomain subject-reduction obstruction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionResidualFence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, any of the four discharge obligations, dependent-codomain preservation, candidate normalization as a preservation theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-obligation discharge exposure, residual-row retention, candidate-normalization frontier exposure, obstruction retention, transport, replay, provenance, local naming, and Lean TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICSubjectReductionResidualFenceUp.

Chapter 8380

A Concrete Naming Certificate for $\mathsf{AsymptoticCompactness}^{\uparrow}$

Following Hale and Temam, the $\mathsf{AsymptoticCompactness}^{\uparrow}$ packet records the dynamical compactness condition that long-time tails of a dissipative evolution admit compact finite-window capture. In BEDC form it consumes the dynamical-system certificate of chapter 195, the metric certificate of chapter 57, the compactness certificate of chapter 60, and the stream-window naming surface of chapter 126. The packet is a finite NameCert carrier for tail capture; it is not an ambient compactness theorem for the whole phase space and not a hidden choice principle for selecting convergent subsequences.

Definition 8380.1 (Asymptotic compactness packet carrier)**.** *An* $\mathsf{AsymptoticCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, M, K, S, T, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *flow row whose long-time orbit tail is being read;*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *phase-distance row used to compare tail windows;*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *candidate capture row for the selected finite tail window;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite sequence of requested tail times;*
5. *T is the absorbing-tail row, recording that all sufficiently late windows selected by S are routed into the compact candidate row K;*
6. *W is the finite-window attractor ledger, storing the compact capture window, its metric tolerance rows, and the associated dynamical endpoints;*
7. *H records componentwise* hsame *transport for the dynamical, metric, compact, stream, tail, and window rows;*
8. *C records displayed* Cont *replay from an orbit-tail request through D, M, K, S, T, W, H;*
9. *P is the* Pkg *provenance over* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, K, S, T, W, H, C, P, N$. It has no coordinate for a compact whole phase space, a selected limit point, a host-level sequential compactness principle, or a global attractor object outside the displayed finite ledger.

Theorem 8380.2 (Asymptotic compactness absorbing tail carrier)**.** *Every accepted* $\mathsf{AsymptoticCompactness}^{\uparrow}$ *carrier exposes an absorbing-tail row T before any compact read is consumed. Concretely, a downstream consumer may read the compact row K only as a capture surface for tail windows selected by the stream schedule S and produced by the dynamical row D under the metric row M. Thus the carrier records finite tail capture rather than compactness of the entire phase carrier.*

Theorem 8380.3 (Asymptotic compactness finite-window attractor ledger)**.** *For every accepted* $\mathsf{AsymptoticCompactness}^{\uparrow}$ *carrier*

$$A = (D, M, K, S, T, W, H, C, P, N),$$

the finite-window attractor ledger W is exhausted by the compact capture row K, the stream-selected tail windows from S, the dynamical endpoints produced by D, the metric tolerance comparisons from M, and the structural rows H, C, P, N. Agreement on those rows cannot be repacked as a selected limit, a full global attractor, a compact phase-space certificate, or a choice-backed subsequence selector.

Theorem 8380.4 (Asymptotic compactness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AsymptoticCompactness}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for $D, M, K, S, T, W, H, C, P, N$, classifier transport through the displayed* hsame *rows, continuation replay through C, provenance through P, absorbing-tail exposure through T, finite-window attractor ledger exposure through W, and local naming through N. No consumer-visible row claims a selected orbit limit, a compact ambient phase space, or a global attractor outside the finite BEDC packet.*

Independence witness: dynsystem, metric, compact, streamname: none is bijective to $\mathsf{AsymptoticCompactness}^{\uparrow}$ because this packet jointly carries dynamical flow rows, metric tolerance rows, compact capture rows, stream-selected tail windows, absorbing-tail exposure, finite-window attractor ledger rows, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticCompactness}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticCompactness}^{\uparrow}$ packet is generated from dynamical-system flow rows, metric comparison rows, compact capture rows, stream-selected tail windows, absorbing-tail exposure, finite-window attractor ledgers, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AsymptoticCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 8380.1, Definition 8380.2, Definition 8380.3, and Definition 8380.4 name the finite tail-capture carrier and its immediate obligation rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AsymptoticCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not construct a global attractor, select convergent subsequences, prove compactness of the whole phase space, or bridge-check the dynamical compactness packet.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, absorbing-tail exactness, finite-window ledger exactness, replay, provenance, and local naming for the same packet.

Chapter 8381

A Concrete Naming Certificate for $\mathsf{BishopLocatedBallCover}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis, the $\mathsf{BishopLocatedBallCover}^{\uparrow}$ packet records a finite located ball-cover route for L10 Real and Completion consumers. It sits after the dyadic radius core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the displayed metric and cover siblings of chapter 57, chapter 8085, and chapter 8117. The packet names a finite located cover witness only; it does not claim open-cover compactness, choice, trichotomy, quotient real equality, selected subcovers, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8381.1 (Bishop located ball cover carrier)**.** *A* $\mathsf{BishopLocatedBallCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, Q, W, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the displayed* $\mathsf{Met}^{\uparrow}$ *or completion-facing metric source whose ball membership comparisons are inspected.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, carrying the positive dyadic radii, tolerance refinements, and overlap budgets for the cover rows.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window through which points, centers, and requested radii are exposed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational distance and radius comparisons used by the locatedness test.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dyadic radius row, finite window row, and regular rational readback have been displayed.*
6. *L is the located cover ledger. For each inspected window it names a finite ball entry, its center row, its dyadic radius, and the comparison showing that the inspected point is either inside the displayed ball or separated from the complementary inspected window.*
7. *H records componentwise* hsame *transport for the metric source, radius ledger, finite windows, readback, seal, located cover ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a ball-cover request through M, Q, W, R, E, L, H.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedBallCover}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteOpenBallCover}^{\uparrow}$, $\mathsf{RationalBallCover}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedBallCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, W, R, E, L, H, C, P, N$. It has no coordinate for an arbitrary open cover, a host compactness theorem, a selected subcover, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopLocatedBallCoverUp`

Theorem 8381.2 (Bishop located ball cover NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedBallCover}^{\uparrow}$ *carrier $B = (M, Q, W, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedBallCover}^{\uparrow}}$ *surface consists exactly of metric-source admission, dyadic radius admission, finite observation-window exposure, regular rational readback, terminal real sealing, located cover ledger exposure, componentwise transport, displayed replay, provenance, and local naming for the rows $M, Q, W, R, E, L, H, C, P, N$.*

Proof. Project the carrier of Definition 8381.1. The analytic rows are M, Q, W, R, E, L, while H, C, P, N only transport, replay, source, and name that displayed route.

The located cover ledger L can be read only together with the metric source M, dyadic radius ledger Q, finite window W, and regular rational readback R. Thus each cover claim is witnessed by finite rows already present in the carrier.

Finally the terminal seal E is downstream of those rows, and the structural rows preserve the same finite route. Since the carrier has no coordinate for compactness, selected subcovers, countable choice, quotient equality, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8381.3 (Bishop located ball cover radius locality)**.** *Every radius read in an accepted* $\mathsf{BishopLocatedBallCover}^{\uparrow}$ *packet factors through the dyadic ledger Q before the cover ledger L or terminal seal E can be consumed. A consumer may refine a displayed radius by a finite dyadic tolerance row, but it cannot replace Q by a global real-order decision, trichotomy branch, quotient real equality, or ambient completeness principle.*

Proof. Read the carrier rows. Radius data appear only in Q, and the finite observation row W determines which radius and center entries are visible for the current request.

The regular rational readback R interprets the dyadic radius comparisons before the located cover ledger L is read. Therefore every radius-facing consumer receives a finite dyadic route rather than an ambient real-order oracle.

The seal E and the structural rows H, C, P, N remain downstream of Q, W, R, L. They preserve the displayed route and add no trichotomy, quotient equality, selected subcover, choice principle, or ambient completeness witness. □

Theorem 8381.4 (Bishop located ball cover finite-window realizer)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopLocatedBallCover}^{\uparrow}$ *carrier must first read the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The located ball-cover row L is realized only over that route: the displayed dyadic radii, finite observations, regular rational comparisons, and terminal real seal are consumed before any cover-facing conclusion is exported.

Proof. Start from a Real or Completion consumer of B. The consumer can reach the terminal seal only through Q, W, R, E, because these are the dyadic, window, readback, and seal rows present in the carrier.

The located cover ledger L is attached to the same route. It records finite ball entries and local comparisons for inspected windows, so it realizes the cover surface without selecting an external subcover or quotient representative.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route. Hence the Real-facing output remains confined to the displayed dyadic-window-readback-seal path and exports no open-cover compactness, choice, trichotomy, quotient real equality, selected limit, or ambient completeness principle. □

Falsifiable prediction: An accepted $\mathsf{BishopLocatedBallCover}^{\uparrow}$ consumer that reaches a cover-facing $\mathsf{Real}^{\uparrow}$ seal without exposing $Q, W, R, E, L, H, C, P, N$ refutes the carrier of Definition 8381.1.

Independence witness: dyadicratcore, streamname, regseqrat, real, finite-open-ball-cover, rational-ball-cover, metric: no listed sibling supplies the joint packet containing one metric source, one dyadic radius ledger, finite observation windows, regular rational readback, terminal Real seal, located cover ledger, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedBallCover}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedBallCover}^{\uparrow}$ packet is generated from a Metric source row, DyadicRatCore radius ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, a located ball-cover ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedBallCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8381.1, Definition 8381.2, Definition 8381.3, and Definition 8381.4 seed the finite located ball-cover route through dyadic radii, finite StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedBallCover}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.BishopLocatedBallCoverUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice, trichotomy, quotient real equality, selected subcovers, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic radius locality, finite-window realization, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedBallCoverUp.

Chapter 8382

A Concrete Naming Certificate for $\mathsf{BishopDiagonalCompletion}^{\uparrow}$

Following Bishop and Bridges constructive completion, the $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ packet records the finite diagonal route by which rational regular names, dyadic tolerances, and inspected stream windows are organized before a terminal Real-facing seal is exposed. It sits after the dyadic arithmetic ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the effective Cauchy completion route of chapter 5930, and the terminal real-name seal of chapter 38. The packet names a diagonal completion interface; it is not quotient stream equality, a selected limit representative, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8382.1 (Bishop diagonal completion carrier)**.** *A* $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, \Delta, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic scales, rational slack rows, and comparison budgets for the diagonal requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected rational-name windows at the scales displayed by D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row obtained from the inspected windows and the dyadic tolerance ledger.*
4. *Δ is the diagonal schedule row, listing the finite request order by which later windows reuse earlier tolerance and readback data.*
5. *E is the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *route row, exposing the completion-facing handoff that consumes D, S, R, Δ before any terminal seal is visible.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the effective completion route has displayed the diagonal windows and regular rational readback.*
7. *H records componentwise* hsame *transport for the dyadic, stream-window, regular-readback, diagonal-schedule, effective-completion, Real-seal, replay, provenance, and naming rows.*
8. *C records the displayed* Cont *replay from a diagonal completion request through $D, S, R, \Delta, E, L, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopDiagonalCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, \Delta, E, L, H, C, P, N$. It has no coordinate for quotient representatives, hidden limit selection, countable-choice schedules, host equality of reals, or an unmediated completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8382.2 (Bishop diagonal completion NameCert obligations)**.** *For every accepted* BishopDiagonalCompleti *carrier $B = (D, S, R, \Delta, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite stream-window exposure, regular rational readback, diagonal schedule display, effective Cauchy completion routing, Real-seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier coordinates. The rows D, S, R, Δ, E, L display the finite dyadic tolerance data, inspected name windows, regular Cauchy readback, diagonal request schedule, effective completion route, and terminal Real-facing seal.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. They do not add quotient representatives, selected limits, countable-choice data, host equality, or an ambient completeness witness.

Since the carrier has no coordinate outside $D, S, R, \Delta, E, L, H, C, P, N$, every accepted local NameCert obligation is one of the listed rows, and every listed row is explicitly present in the accepted packet. □

Theorem 8382.3 (Bishop diagonal completion regular seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ *carrier is reached through displayed finite diagonal windows and regular rational readback. More precisely, any Real-facing consumer factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{EffectiveCauchyCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

with the diagonal schedule row Δ governing which displayed windows are consumed before the effective completion route is read.

Proof. Begin with a Real-facing consumer of B. Carrier admission gives access to the terminal seal only as the row L in the same finite packet as D, S, R, Δ, E.

Replay the continuation row C. The request first reads the dyadic tolerance ledger D, opens the finite $\mathsf{StreamName}^{\uparrow}$ windows S, obtains the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback R, and consults the diagonal schedule Δ that records the inspected finite window order.

Only after that displayed route is available can the effective completion row E be consumed and the terminal seal L be read. Componentwise transport and package provenance preserve the same order, so no host limit operation can bypass the finite diagonal windows or the regular rational readback. □

Theorem 8382.4 (Bishop diagonal completion no quotient nonescape)**.** *No accepted* $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ *packet exports quotient stream equality, a selected representative of a Cauchy class, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any such datum is outside the local carrier unless it is re-expressed through the displayed dyadic, stream-window, regular-readback, diagonal-schedule, effective-completion, Real-seal, transport, replay, provenance, and naming rows.*

Proof. Inspect the classifier of Definition 8382.1. It compares accepted packets only through the ten displayed rows $D, S, R, \Delta, E, L, H, C, P, N$.

The excluded data are not among those coordinates. The regular seal route also forces every Real-facing read through the dyadic, finite-window, regular-readback, diagonal-schedule, and effective-completion route before the terminal seal is visible.

Therefore an export using quotient equality, selected representatives, countable choice, host equality, or ambient completeness has left the accepted $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ surface rather than escaping from it. □

Falsifiable prediction: Every accepted BishopDiagonalCompletionUp Real-facing read factors through DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat regular readback, a displayed diagonal schedule, EffectiveCauchyCompletion routing, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, effective-cauchy-completion, real: $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerances, finite windows, regular readback, a diagonal schedule, effective Cauchy completion routing, Real sealing, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDiagonalCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular Cauchy readback, a diagonal schedule row, an EffectiveCauchyCompletion route, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8382.1, Definition 8382.2, Definition 8382.3, and Definition 8382.4 seed the finite diagonal completion route through dyadic tolerance, finite stream-window, regular-readback, effective-completion, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window readback, regular Cauchy diagonal routing, effective completion handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopDiagonalCompletionUp.

Chapter 8383

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimsup}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLimsup}^{\uparrow}$ packet records the finite upper-tail envelope available from a displayed regular Cauchy rational source before any Real-facing consumer reads a terminal value. It sits over the limsup boundary of chapter 768, the located upper-envelope route of chapter 7935, the bounded upper-envelope packet of chapter 7101, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names an L10 completion-facing upper-tail envelope; it does not import host limsup, quotient stream equality, countable choice, selected subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8383.1 (Regular Cauchy limsup carrier)**.** *A* $\mathsf{RegularCauchyLimsup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, R, D, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row, recording the displayed regular Cauchy rational name whose upper tail is being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite tail windows made visible to the upper-envelope request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row, binding the windows from W to regular rational values and tail tolerances.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic precision, slack, and comparison budgets for the envelope read.*
5. *L is the* $\mathsf{Limsup}^{\uparrow}$ *and* $\mathsf{LocatedLimsup}^{\uparrow}$ *upper-envelope row, exposing the finite tail-upper surface and the located comparison tests available at the displayed precision.*
6. *B is the* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *comparison row, recording the bounded upper-envelope handoff over the same source row S and the same displayed tail windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, R, D, L, B have all been displayed.*
8. *H records componentwise* hsame *transport for the source, windows, readback, dyadic ledger, upper-envelope, bounded-comparison, Real-seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from an upper-tail request through*

$$S \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow L \longrightarrow B \longrightarrow E \longrightarrow H \longrightarrow P \longrightarrow N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimsup}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{LocatedLimsup}^{\uparrow}$, $\mathsf{BoundedRealLimsup}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimsup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, R, D, L, B, E, H, C, P, N$. It has no coordinate for quotient stream equality, a host-selected limsup value, a chosen cofinal subsequence, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8383.2 (Regular Cauchy limsup NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimsup}^{\uparrow}$ *carrier*

$$U = (S, W, R, D, L, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimsup}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source sequence,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Limsup}^{\uparrow}$ *and* $\mathsf{LocatedLimsup}^{\uparrow}$ *upper-envelope row,* $\mathsf{BoundedRealLimsup}^{\uparrow}$ *comparison row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier rows of Definition 8383.1. The analytic rows S, W, R, D, L, B, E display the source name, finite windows, regular rational readback, dyadic tolerance ledger, upper-envelope request, bounded comparison row, and terminal Real seal in that order.

The structural rows H, C, P, N transport, replay, source, and name the same packet. They do not add a quotient stream, a selected cofinal subsequence, a host limsup, countable choice, or ambient Real completeness.

Since the classifier has no coordinate outside $S, W, R, D, L, B, E, H, C, P, N$, every local NameCert obligation is one of the listed rows, and every listed row is visible in the accepted carrier. □

Theorem 8383.3 (Regular Cauchy limsup Real seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchyLimsup}^{\uparrow}$ *carrier is reached through displayed regular-Cauchy upper-tail data. More precisely, every Real or Completion-facing read factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{LocatedLimsup}^{\uparrow} \longrightarrow \mathsf{BoundedRealLimsup}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

with the same source sequence row S and the same displayed tail windows W governing the upper-envelope comparison.

Proof. Let a Real or Completion-facing consumer read an accepted carrier U. The terminal value is available only as the row E, and E is downstream of the source row S, window row W, readback row R, dyadic tolerance ledger D, upper-envelope row L, and bounded comparison row B.

Replay the continuation row C. The consumer first opens the finite $\mathsf{StreamName}^{\uparrow}$ windows, then reads regular rational tail values through $\mathsf{RegSeqRat}^{\uparrow}$, and then consults the dyadic ledger before the located and bounded upper-envelope rows are available.

Componentwise transport H and provenance P preserve the same order. Therefore the Real seal cannot bypass the displayed regular-Cauchy tail rows or import a host limsup, quotient stream equality, countable choice, selected subsequence, or ambient completeness witness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLimsup}^{\uparrow}$ Real-facing read exposes a source sequence, StreamName windows, RegSeqRat tail readbacks, a DyadicRatCore tolerance ledger, Limsup and LocatedLimsup upper-envelope data, a BoundedRealLimsup comparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any upper-tail endpoint is visible.

Independence witness: limsup, located-limsup, bounded-real-limsup, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyLimsup}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regular Cauchy source, finite windows, rational tail readback, dyadic tolerances, located upper-envelope data, bounded upper-envelope comparison, Real sealing, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimsup}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimsup}^{\uparrow}$ packet is generated from a regular Cauchy source sequence, StreamName windows, RegSeqRat tail readbacks, a DyadicRatCore tolerance ledger, Limsup and LocatedLimsup upper-envelope rows, a BoundedRealLimsup comparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimsup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8383.1, Definition 8383.2, and Definition 8383.3 seed the finite regular-Cauchy upper-tail envelope route through displayed windows, rational readback, dyadic tolerances, located and bounded limsup rows, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimsup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limsup, quotient stream equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, RegSeqRat readback, DyadicRatCore tolerance exposure, located upper-envelope exactness, bounded comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLimsupUp.

Chapter 8384

A Concrete Naming Certificate for $\mathsf{CauchyProductTopology}^{\uparrow}$

Following Bourbaki and Kelley on finite product topologies, the $\mathsf{CauchyProductTopology}^{\uparrow}$ packet records the product-topology read available when each coordinate is presented by a constructive Cauchy real name. It sits after the finite topology certificate of chapter 2000, the Cauchy coordinate-projection certificate of chapter 2076, and the completion-projection certificate of chapter 5824. The packet gives finite Cauchy-name product consumers a topology-facing coordinate read without importing quotient product equality, selected stream representatives, ambient completeness, or host product spaces.

Definition 8384.1 (Cauchy product topology carrier)**.** *A* $\mathsf{CauchyProductTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (P, R, Q, W, A, S, H, C, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProductTopology}^{\uparrow}$ *finite product-topology row naming the coordinate-window topology surface.*
2. *R is the* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *coordinate-projection readback row.*
3. *Q is the* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *completion-facing projection row.*
4. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows used by the coordinate Cauchy names.*
5. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the displayed windows.*
6. *S is the pair of* $\mathsf{Real}^{\uparrow}$ *seal rows reached from the regular readbacks.*
7. *H records componentwise* hsame *transport for topology, projection, completion, stream-window, regular-readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each product-coordinate topology request through $P, R, Q, W, A, S, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{CauchyProductTopology}^{\uparrow}$, $\mathsf{ProductTopology}^{\uparrow}$, $\mathsf{RegularCauchyProductProjection}^{\uparrow}$, $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductTopology}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite product-topology read over Cauchy real-name coordinates. It has no coordinate for quotient product equality, selected stream representatives, host product spaces, global completeness, choice of representatives, Lean verification, or bridge checking.

Theorem 8384.2 (Cauchy product topology NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductTopology}^{\uparrow}$ *carrier*

$$T = (P, R, Q, W, A, S, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, product-topology exposure through* P*, regular Cauchy product projection exposure through* R*, completion projection exposure through* Q*, finite stream-window exposure through* W*, regular-rational readback through* A*, real-name sealing through* S*, componentwise transport through* H*, displayed replay through* C*, provenance through* G*, and local naming through* N*.*

Proof. Project the carrier of Definition 8384.1. Its rows are precisely the topology row, projection rows, stream-window rows, regular-rational readback, real seals, and structural certificate rows listed in the statement.

Read the three sibling certificates. The $\mathsf{ProductTopology}^{\uparrow}$ chapter supplies the finite product-window topology row, chapter 2076 supplies the coordinate Cauchy projection row, and chapter 5824 supplies the completion-facing projection row.

Finally replay through H, C, G, N. These rows transport, replay, source, and name the already displayed route, so the local certificate cannot add quotient product equality, selected stream representatives, host product spaces, global completeness, or a Lean-side verification claim. □

Theorem 8384.3 (Cauchy product topology finite product readback)**.** *Every finite product-coordinate topology read exported by an accepted* $\mathsf{CauchyProductTopology}^{\uparrow}$ *carrier factors through*

$$P, R, Q, W, A, S, H, C, G, N$$

in that order: the $\mathsf{ProductTopology}^{\uparrow}$ *window is exposed before the* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *and* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *projection rows are consumed, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks are displayed before the* $\mathsf{Real}^{\uparrow}$ *seals, and the final route is preserved only by* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Start with the accepted packet T from Definition 8384.1. Since P is the only product-topology row, every coordinate topology read must expose P before any Cauchy projection or completion projection can be consumed.

Use the projection certificates cited in the introduction. The regular Cauchy projection row R reads finite coordinate windows, and the completion projection row Q preserves that coordinate route at the completion-facing boundary.

Then read the real-name source. The finite windows W feed the regular-rational row A, and A feeds the terminal real seals S; no selected stream representative or quotient product equality is present in the carrier.

The structural rows H, C, G, N preserve and name the same finite route. Hence the exported read is the displayed Cauchy-name product topology route, not an ambient product space, host completeness theorem, or bridge-checked object. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductTopology}^{\uparrow}$ read displays the $\mathsf{ProductTopology}^{\uparrow}$ finite product window, $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ coordinate readback, $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$

completion projection, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seals, hsame transport, Cont replay, Pkg provenance, and local NameCert before a product-coordinate topology consumer can use the route.

Independence witness: product-topology, regular-cauchy-product-projection, cauchy-product-completion-projection, streamname, regseqrat, real: $\mathsf{CauchyProductTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries product-topology windows, regular Cauchy projection readback, completion projection, finite stream windows, regular-rational readback, real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductTopology}^{\uparrow}$ packet is generated from a ProductTopology finite window row, RegularCauchyProductProjection coordinate readback, CauchyProductCompletionProjection completion-facing row, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8384.1, Definition 8384.2, and Definition 8384.3 seed the finite product-topology route over Cauchy real-name coordinates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient product equality, selected stream representatives, ambient product spaces, global completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite product-window exposure, coordinate projection readback, completion projection exposure, stream-window accounting, regular-rational readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductTopologyUp.

Chapter 8385

A Concrete Naming Certificate for $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$ packet records the finite limit-facing interface by which a regular rational Cauchy name is admitted as a separated completion limit without claiming ambient $\mathsf{Real}^{\uparrow}$ completeness. It consumes the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It also sits downstream of the quotient-free separated completion surface of chapter 8373.

Definition 8385.1 (Separated Cauchy completion limit carrier)**.** *A* $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, E, H, T, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected stages of the Cauchy name.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to those windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the displayed windows.*
4. Q *is the separated completion-limit row, recording that the named regular Cauchy data has passed the zero-distance boundary inside the displayed route.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after* S, R, D, Q *have been exposed.*
6. H *records componentwise* hsame *transport for the window, readback, tolerance, separated-limit, Real seal, provenance, and local naming rows.*
7. T *records the displayed continuity boundary*

$$\mathsf{Cont}(L) = \mathsf{Cont}(S, R, D, Q, E, H).$$

8. P *records*

$$\mathsf{Pkg}(L) = \mathsf{Pkg}(\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}$$

9. N *is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, E, H, T, P, N$. It has no coordinate for quotient stream equality, selected host limits, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.SeparatedCauchyCompletionLimitUp`

Theorem 8385.2 (Separated Cauchy completion limit NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$ *carrier*

$$L = (S, R, D, Q, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the displayed packet L;*
2. *finite observation exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row S;*
3. *regular rational readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. *dyadic comparison confinement through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
5. *separated completion-limit sealing through Q;*
6. *terminal* $\mathsf{Real}^{\uparrow}$ *readback through E, reachable only after S, R, D, Q have been displayed;*
7. *componentwise transport, continuity replay, provenance, and local naming through H, T, P, N.*

Thus the packet names a separated completion-limit interface, not a proof that every Cauchy object in an ambient Real carrier has a selected limit.

Proof. Project the carrier of Definition 8385.1. The finite-window row S, regular rational readback row R, and dyadic tolerance row D expose the only data through which a consumer may inspect the Cauchy name.

The separated-limit row Q is downstream of that finite route. It records the displayed zero-distance boundary for this packet rather than a quotient class of streams or a host-selected Real element.

Finally read the terminal Real seal E through H, T, P, N. These structural rows preserve and name the same displayed route, so the NameCert surface contains exactly the listed obligations and exports no ambient completeness principle. □

Theorem 8385.3 (Separated Cauchy completion limit uniqueness route)**.** *If two accepted* SeparatedCauchyComplet *packets agree componentwise on S, R, D, Q, H, T, P, N by* hsame*, then every* $\mathsf{Real}^{\uparrow}$*-facing consumer reads the same displayed limit route through E. The comparison is made through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{SeparatedCauchyCompletionLimit}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and does not invoke quotient equality of Cauchy streams, trichotomy, countable choice, a selected host limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the shared rows S, R, D, Q, H, T, P, N. The window, readback, and dyadic tolerance rows determine the displayed Cauchy route available to both packets.

Use Definition 8385.2. It places the separated-limit row Q after the common finite data and the terminal Real seal E after Q, so a Real-facing consumer cannot bypass the displayed comparison boundary.

Transport by H and replay by T preserve the same finite route, while P and N name it locally. Since the carrier contains no quotient, choice, trichotomy, selected host-limit, or ambient-completeness coordinate, the uniqueness route is exhausted by the displayed seal path. □

Falsifiable prediction: An accepted $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$ packet whose $\mathsf{Real}^{\uparrow}$ seal is read without first exposing $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the separated-limit row Q refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, separated-cauchy-completion: no listed sibling supplies the whole packet $L = (S, R, D, Q, E, H, T, P, N)$, because each sibling contributes only finite windows, regular rational readback, dyadic tolerances, a Real seal, or separated completion context while this chapter binds them into one limit-facing NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}$ packet is generated from StreamNameUp finite windows, RegSeqRatUp readback, a DyadicRatCoreUp tolerance ledger, a separated completion-limit row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8385.1, Definition 8385.2, and Definition 8385.3 seed the finite separated Cauchy completion-limit interface over window, readback, tolerance, separated-limit, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyCompletionLimit}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular rational readback, dyadic tolerance confinement, separated-limit sealing, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCauchyCompletionLimitUp.

Chapter 8386

A Concrete Naming Certificate for $\mathsf{RegularSequenceTailMetric}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and constructive completion, the $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ packet records the finite tail-metric surface by which regular rational names are compared before a terminal real-name seal is visible. It sits after the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the public finite fusion route of chapter 693, and the terminal real-name seal of chapter 38. The packet names a tail metric for displayed regular sequences only; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient stream equality, countable choice, trichotomy, or selected representatives of completed real names.

Definition 8386.1 (Regular sequence tail metric carrier)**.** *A* $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, W, R, F, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic scale at which two tails are compared.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite tail windows inspected at the requested scale.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence values visible on those windows.*
4. *F is the* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *fusion row, recording that the window, readback, tolerance, and terminal seal are consumed through one finite L10 route.*
5. *M is the tail-metric row, storing the displayed rational bound for the tail comparison obtained from D, W, R, F.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tail-metric row has been displayed.*
7. *H records componentwise* hsame *transport for $D, W, R, F, M, E, C, P, N$.*
8. *C records the displayed* Cont *replay from a tail comparison request through $D, W, R, F, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceTailMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularSequenceTailMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, W, R, F, M, E, H, C, P, N$. It has no coordinate for quotient equality of streams, a chosen limit representative, countable choice over tails, trichotomy of real names, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8386.2 (Regular sequence tail metric NameCert obligations)**.** *For every accepted* RegularSequenceTailMe *carrier*

$$T = (D, W, R, F, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularSequenceTailMetric}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite tail-window exposure, regular rational readback, regular Cauchy window fusion, displayed tail-metric bounding, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.RegularSequenceTailMetricUp.RegularSequenceTailMetricTasteGate single carrier alignment`

Proof. Project the carrier of Definition 8386.1. The displayed coordinates are precisely $D, W, R, F, M, E, H, C, P, N$

The comparison-facing rows are D, W, R, F, M. The dyadic tolerance row sets the scale, the stream row supplies finite tail windows, the regular readback row supplies rational values on those windows, the fusion row keeps those reads on one L10 route, and the metric row records the resulting bound.

The seal and structural rows E, H, C, P, N preserve, replay, source, and name the same displayed route. Since the carrier has no quotient, choice, trichotomy, selected-representative, or ambient-completeness coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 8386.3 (Regular sequence tail metric tail stability)**.** *Let*

$$T = (D, W, R, F, M, E, H, C, P, N)$$

and

$$T' = (D, W', R', F', M, E, H', C', P', N')$$

be accepted $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ *packets with the same dyadic tolerance row D, tail-metric row M, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E. If (W', R', F') is displayed as a cofinal tail of (W, R, F) inside the accepted packet data, then T and T' export the same finite tail-metric handoff through M and E. Neither packet exports quotient stream equality, a selected representative, countable choice, trichotomy, or ambient completeness.*

Proof. Start with Definition 8386.2. Both packets expose their dyadic tolerance, tail windows, regular readbacks, fusion rows, tail-metric rows, and terminal seals before the structural rows are consumed.

Because (W', R', F') is a displayed cofinal tail of (W, R, F), the second packet changes only the visible finite tail window and its regular readback route. The shared dyadic tolerance D and metric row M keep the requested comparison scale and bound fixed.

The terminal seal row E is reached only after M is displayed. The structural rows transport, replay, source, and name the same handoff, so passing to the displayed cofinal tail preserves the exported tail-metric surface.

The excluded data are not carrier coordinates in either packet. Thus no quotient stream equality, selected representative, countable choice, trichotomy, or ambient completeness can be exported by the tail replacement. □

Theorem 8386.4 (Regular sequence tail metric RegSeqRat handoff)**.** *Every accepted* $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ *comparison consumed by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing reader factors through the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{RegularCauchyWindowFusion}^{\uparrow} \longrightarrow \mathsf{RegularSequenceTailMetr}$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *row supplies the rational regular readback used by the tail metric row* M*, and the later seal row* E *is unreadable until that finite handoff has been displayed.*

Proof. Project an accepted carrier $T = (D, W, R, F, M, E, H, C, P, N)$. The rows D, W, R, F, M, E are ordered as dyadic tolerance, stream window, regular rational readback, finite fusion route, tail metric, and terminal real-name seal.

Apply Definition 8386.2. It requires every accepted comparison to expose D, W, R, F, M, E before H, C, P, N can package the result.

The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback coordinate in the carrier. Therefore the metric row M can only be read through the displayed regular rational values supplied by R and the finite window supplied by W.

Finally the row E is a seal for that already displayed route. Since no other completion, quotient, selected-limit, choice, or trichotomy coordinate exists in the packet, every $\mathsf{Real}^{\uparrow}$ or completion-facing reader factors through the stated finite handoff. □

Falsifiable prediction: Every accepted RegularSequenceTailMetricUp comparison exposes DyadicRatCore tolerance data, StreamName finite tail windows, RegSeqRat regular rational readback, RegularCauchyWindowFusion routing, a displayed tail-metric bound, a RealUp seal, transport, replay, provenance, and local naming before any Real-facing consumer reads the result.

Independence witness: dyadicratcore, streamname, regseqrat, regularcauchywindowfusion, real: $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ is not bijective to any listed sibling because it carries the displayed tail-metric row M between regular rational readback and the terminal Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceTailMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName finite tail-window row, RegSeqRat regular readback row, RegularCauchyWindowFusion route row, tail-metric row, RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceTailMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8386.1, Definition 8386.2, Definition 8386.3, and Definition 8386.4 seed the finite tail-metric surface between regular rational readback and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceTailMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, selected representatives, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite tail-window exposure, regular readback, fusion routing, tail-metric bounding, terminal Real seal scope, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceTailMetricUp.

Chapter 8387

A Concrete Naming Certificate for $\mathsf{KrawczykOperator}^{\uparrow}$

Following Krawczyk and Moore–Kearfott–Cloud interval analysis, the $\mathsf{KrawczykOperator}^{\uparrow}$ packet records the finite preconditioned interval operator used to validate and narrow boxes for nonlinear zero searches. It separates the Krawczyk image from the surrounding $\mathsf{IntervalNewton}^{\uparrow}$ row of chapter 1570: the operator is a named L10 certificate whose arithmetic is read through chapter 302, whose Newton comparison route is read through chapter 223, whose enclosure provenance may use the Taylor rows of chapter 442, and whose rational and real readback passes through chapter 301 and chapter 38.

Definition 8387.1 (Krawczyk operator carrier)**.** *A* $\mathsf{KrawczykOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, J, A, R, M, I, V, H, C, P, N)$$

with the following displayed rows:

1. *B is the input interval box, including the coordinate boxes and their* $\mathsf{IntervalArithmetic}^{\uparrow}$ *endpoint rows.*
2. *F is the interval enclosure of the function values on B, read only through interval-arithmetic operation rows.*
3. *J is the interval matrix or slope enclosure for the derivative data over B.*
4. *A is the approximate inverse or preconditioner row attached to the chosen center.*
5. *R is the center and residual row, recording the displayed center c and the preconditioned residual $AF(c)$.*
6. *M is the interval matrix row for $I - AJ$, including its interval product and subtraction ledger.*
7. *I is the Krawczyk image row, recording the displayed enclosure $c - AF(c) + (I - AJ)(B - c)$.*
8. *V is the containment and narrowing validation row, recording whether $I \subseteq B$, $I \cap B$, or the refused containment boundary is the public result.*
9. *H records componentwise* hsame *transport for boxes, enclosures, derivative rows, preconditioners, residuals, interval matrices, Krawczyk images, and validation rows.*
10. *C records displayed* Cont *replay from interval arithmetic to the image computation, from validation to narrowing, and from readback rows to the public certificate.*

11. *P is the* Pkg *provenance over* $\mathsf{KrawczykOperator}^{\uparrow}$, $\mathsf{IntervalArithmetic}^{\uparrow}$, $\mathsf{IntervalNewton}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KrawczykOperator}^{\uparrow}}$ *row.*

The carrier exports only the displayed interval operator and its validation ledger. It does not assert host existence of a zero, global uniqueness, floating-point semantics, differentiability outside the supplied slope row, quotient real equality, or ambient Real completeness.

Theorem 8387.2 (Krawczyk operator NameCert obligations)**.** *For every accepted* $\mathsf{KrawczykOperator}^{\uparrow}$ *carrier*

$$K = (B, F, J, A, R, M, I, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KrawczykOperator}^{\uparrow}}$ *surface consists exactly of carrier admission, box exposure through B, function enclosure exposure through F, derivative or slope enclosure exposure through J, preconditioner exposure through A, center and residual exposure through R, interval matrix exposure through M, Krawczyk image exposure through I, containment or narrowing validation through V, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The only arithmetic rows available to a Krawczyk read are the displayed box, function enclosure, derivative or slope enclosure, preconditioner, residual, interval matrix, image, and validation rows.

The interval-arithmetic route computes F, J, M, and I inside the same finite packet. The row A is only a displayed approximate inverse or preconditioner, and R is only the center-residual ledger attached to that preconditioner.

Validation is read through V. It records containment, narrowing by intersection, or refusal of containment without adding a root-existence theorem or a uniqueness theorem.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for host zeros, global uniqueness, floating-point semantics, quotient real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8387.3 (Krawczyk operator interval-Newton handoff)**.** *Every* $\mathsf{IntervalNewton}^{\uparrow}$ *consumer that uses a* $\mathsf{KrawczykOperator}^{\uparrow}$ *packet must read the Krawczyk route*

$$B, F, J, A, R, M, I, V, H, C, P, N$$

before treating a correction row as a validated enclosure. The handoff supplies the named image $I = c - AF(c) + (I - AJ)(B - c)$ and the containment or narrowing validation row V to the interval-Newton packet; it supplies no solver oracle, host zero selection, global uniqueness claim, floating-point semantics, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the interval-Newton correction boundary of chapter 1570. That chapter may consume a Krawczyk correction row, but the row is public only after the box, enclosure, derivative, and validation data have been displayed.

The present carrier names those data as B, F, J, A, R, M, I, V. The interval-arithmetic rows compute the residual and image, while the $\mathsf{NewtonIteration}^{\uparrow}$ and $\mathsf{TaylorModel}^{\uparrow}$ provenance rows record how the displayed derivative or slope data were obtained.

The rational and real readback remains finite. Any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the image and validation rows only through the displayed replay and provenance rows C, P together with the transport and local name H, N.

Thus the interval-Newton consumer receives a named finite operator certificate rather than an unnamed correction row. Since no carrier field selects a zero or proves global uniqueness, the handoff remains an interval validation route. □

Falsifiable prediction: Every accepted $\mathsf{KrawczykOperator}^{\uparrow}$ read exposes the box, function enclosure, derivative or slope enclosure, preconditioner, center-residual row, interval matrix row, Krawczyk image, validation ledger, transport, replay, provenance, and local NameCert row before an interval-Newton consumer can use the correction.

Independence witness: intervalnewton, intervalarithmetic, newtoniteration, taylormodel, regseqrat, real: $\mathsf{KrawczykOperator}^{\uparrow}$ is not bijective to any listed sibling because it binds the preconditioned residual, interval matrix $I - AJ$, Krawczyk image, containment-or-narrowing validation, arithmetic replay, rational readback, real-name readback, provenance, and local naming into one operator certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{KrawczykOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{KrawczykOperator}^{\uparrow}$ packet is generated from an interval box, function enclosure, derivative or slope enclosure, approximate inverse or preconditioner, center-residual row, interval matrix row, Krawczyk image, containment or narrowing validation row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KrawczykOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8387.1, Definition 8387.2, and Definition 8387.3 seed the finite L10 Krawczyk operator certificate and its interval-Newton consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KrawczykOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host zero existence, global uniqueness, floating-point semantics, differentiability beyond the displayed slope row, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, interval-arithmetic image formation, containment or narrowing validation, RegSeqRat and Real readback nonescape, interval-Newton handoff, transport, replay, provenance, and local naming for BEDC.Derived.KrawczykOperatorUp.

Chapter 8388

A Concrete Naming Certificate for IntervalNewtonKrawczyk$^{\uparrow}$

Following Krawczyk and Moore–Kearfott–Cloud interval analysis, the IntervalNewtonKrawczyk$^{\uparrow}$ packet records the bridge between interval Newton correction rows and Krawczyk validation rows as finite BEDC data. It sits between the interval-Newton carrier of chapter 1570 and the preconditioned operator carrier of chapter 8387. Its purpose is to expose the exact handoff by which a displayed interval box, interval-arithmetic enclosure, Newton correction, Krawczyk image, Taylor or slope remainder, rational readback, real-name seal, transport, replay, provenance, and local naming row jointly validate a narrowing certificate without adding a solver oracle, host zero selection, floating-point semantics, quotient real equality, or ambient Real$^{\uparrow}$ completeness.

Definition 8388.1 (Interval Newton–Krawczyk carrier)**.** *An* IntervalNewtonKrawczyk$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, K, A, N, T, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the accepted* IntervalNewton$^{\uparrow}$ *row carrying the displayed interval box, function enclosure, derivative or slope enclosure, correction row, containment ledger, and validated readback boundary.*
2. *K is the accepted* KrawczykOperator$^{\uparrow}$ *row carrying the preconditioned residual, interval matrix, Krawczyk image, and containment or narrowing validation ledger.*
3. *A is the* IntervalArithmetic$^{\uparrow}$ *operation row through which both I and K read endpoint operations, interval products, interval sums, and inclusion tests.*
4. *N is the* NewtonIteration$^{\uparrow}$ *correction-provenance row used by the interval Newton side.*
5. *T is the* TaylorModel$^{\uparrow}$ *or displayed slope-remainder row used by the Krawczyk side when the derivative enclosure is read through finite Taylor data.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the validated enclosure.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after I, K, A, N, T, R have been displayed.*
8. *H records componentwise* hsame *transport for interval Newton, Krawczyk, arithmetic, Newton, Taylor, rational readback, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from interval arithmetic through the Newton and Krawczyk validations into the rational and real readback rows.*

10. *P is the* Pkg *provenance over* IntervalNewtonKrawczyk$^{\uparrow}$, IntervalNewton$^{\uparrow}$, KrawczykOperator$^{\uparrow}$, IntervalArithmetic$^{\uparrow}$, NewtonIteration$^{\uparrow}$, TaylorModel$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{IntervalNewtonKrawczyk}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the visible rows $I, K, A, N, T, R, E, H, C, P, L$.

Theorem 8388.2 (Interval Newton–Krawczyk NameCert obligations)**.** *For every accepted* IntervalNewtonKrawczyk *carrier*

$$V = (I, K, A, N, T, R, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalNewtonKrawczyk}^{\uparrow}}$ *surface consists exactly of interval-Newton exposure through I, Krawczyk-operator exposure through K, interval-arithmetic exposure through A, Newton correction provenance through N, Taylor or slope-remainder provenance through T,* RegSeqRat$^{\uparrow}$ *readback through R,* Real$^{\uparrow}$ *seal delivery through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8388.1. The interval Newton side is available only through I, and the Krawczyk side is available only through K.

Read the arithmetic rows next. The row A supplies the interval endpoint operations for both validations, N records the Newton correction provenance, and T records the Taylor or slope-remainder provenance used by the Krawczyk image.

The readback rows are then forced by the displayed order. The rational readback R and real seal E can be consumed only after the interval Newton and Krawczyk rows have exposed the same box, correction, image, and containment data.

The rows H, C, P, L transport, replay, source, and name this finite route. Since the carrier has no extra coordinate, solver oracles, host zero selection, floating-point semantics, quotient real equality, and ambient Real completeness are outside the local certificate. □

Theorem 8388.3 (Interval Newton–Krawczyk validation handoff)**.** *Every accepted* IntervalNewtonKrawczyk$^{\uparrow}$ *validation handoff from interval Newton to a Krawczyk read is routed through*

IntervalNewton$^{\uparrow}$, KrawczykOperator$^{\uparrow}$, IntervalArithmetic$^{\uparrow}$, NewtonIteration$^{\uparrow}$, TaylorModel$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHi

In an accepted carrier $V = (I, K, A, N, T, R, E, H, C, P, L)$, a downstream consumer may read the validated enclosure only after the interval Newton correction row, Krawczyk image row, interval-arithmetic operation row, Newton provenance row, Taylor or slope-remainder row, rational readback row, real seal row, transport row, replay row, provenance row, and local naming row have all been displayed.

Proof. Start from the NameCert obligations of Definition 8388.2. They identify the full finite surface $I, K, A, N, T, R, E, H, C, P, L$ and place the interval Newton and Krawczyk rows before any readback row.

Use the interval Newton sibling of chapter 1570. Its correction row is accepted only with the displayed box, containment ledger, interval-arithmetic route, validated readback boundary, and local naming row.

Use the Krawczyk operator sibling of chapter 8387. Its image and containment rows are accepted only after the preconditioned residual, interval matrix, arithmetic replay, validation ledger, transport, provenance, and local name have been exposed.

The present carrier intersects those sibling surfaces at $A, N, T, R, E, H, C, P, L$. Thus the downstream consumer receives a single finite validation route rather than two unrelated certificates.

Because no field stores a solver oracle, host-selected zero, floating-point semantics, quotient real equality, or ambient Real completeness, the handoff exports only the stated BEDC rows. □

Falsifiable prediction: An $\mathsf{IntervalNewtonKrawczyk}^{\uparrow}$ packet that exposes a Krawczyk image without the interval Newton correction row, interval-arithmetic row, Newton provenance row, Taylor or slope-remainder row, and rational-to-real readback route is not accepted.

Independence witness: intervalnewton, krawczykoperator, intervalarithmetic, newtoniteration, taylormodel, regseqrat, real: the bridge packet is not bijective to any listed sibling because it binds both the interval Newton correction and the Krawczyk image to the same arithmetic, provenance, readback, transport, replay, package, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalNewtonKrawczyk}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalNewtonKrawczyk}^{\uparrow}$ packet is generated from an IntervalNewton row, KrawczykOperator row, IntervalArithmetic row, NewtonIteration provenance row, TaylorModel or slope-remainder row, RegSeqRat readback row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalNewtonKrawczyk}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8388.1, Definition 8388.2, and Definition 8388.3 seed the finite bridge surface joining interval Newton correction rows to Krawczyk validation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalNewtonKrawczyk}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host zero existence, global uniqueness, floating-point semantics, quotient real equality, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-arithmetic soundness, Newton correction provenance, Krawczyk image validation, Taylor or slope-remainder provenance, RegSeqRat readback, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.IntervalNewtonKrawczykUp.

Chapter 8389

A Concrete Naming Certificate for $\mathsf{BishopLocatedConvexHull}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ packet records finite located convex-hull approximation data as BEDC naming evidence. It sits over the real-name seal of chapter 38, the metric carrier of chapter 57, the convex membership surface of chapter 208, the compact finite-read boundary of chapter 60, the polytope witness rows of chapter 209, and the bounded-support convex-hull route of chapter 2347. The packet names a finite approximation surface for located convex hulls; it does not import host convex-hull equality, selected closest points, quotient subsets, ambient $\mathsf{Real}^{\uparrow}$ completeness, or countable choice.

Definition 8389.1 (Bishop located convex-hull carrier)**.** *A* $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, M, C, K, P, A, W, D, Q, H, T, S, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal through which tolerances, coefficients, and metric radii are read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ambient row carrying the distance reads used by the located approximation.*
3. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *source row naming the convex family under inspection.*
4. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-read boundary for the inspected source window.*
5. *P is the* $\mathsf{Polytope}^{\uparrow}$ *finite witness row that stores the displayed approximating hull.*
6. *A is the* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *bounded-support handoff row.*
7. *W records the located tolerance window and the finite mesh radius used by the approximation read.*
8. *D records the displayed distance-to-hull comparison rows, read only through R, M, C, K, P, A, W.*
9. *Q records the finite coefficient ledger for the approximating convex combination.*
10. *H records componentwise* hsame *transport for the real, metric, convex, compact, polytope, support, window, distance, coefficient, continuation, provenance, and naming rows.*
11. *T is the displayed* Cont *replay from a tolerance request through the same finite rows.*
12. *S is the* Pkg *provenance over* $\mathsf{BishopLocatedConvexHull}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{CaratheodoryConvexHull}^{\uparrow}$, BHist, hsame, Cont, *and local* NameCert *rows.*

13. N is the local $\mathsf{NameCert}_{\mathsf{BishopLocatedConvexHull}^{\uparrow}}$ *row.*
The carrier has no coordinate for host convex-hull equality, a selected nearest point, quotient subset equality, ambient completeness, separation, or countable choice.

Theorem 8389.2 (Bishop located convex-hull NameCert obligations)**.** *For the carrier of Definition 8389.1, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedConvexHull}^{\uparrow}}$ *surface consists exactly of carrier exposure for L, real and metric readback through R, M, convex-source exposure through C, compact finite-read exposure through K, polytope approximation exposure through P, Caratheodory bounded-support exposure through A, tolerance-window exposure through W, distance-comparison exposure through D, coefficient-ledger exposure through Q, componentwise transport through H, displayed continuation replay through T, package provenance through S, and local naming through N. Classifier stability is tested only by replaying those displayed rows under* hsame *transport.*

Theorem 8389.3 (Bishop located convex-hull finite approximation)**.** *Every accepted* $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ *packet exposes a finite approximation route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ \mathsf{Polytope}^{\uparrow},\ \mathsf{CaratheodoryConvexHull}^{\uparrow}$$

before any located convex-hull distance read is available. The tolerance row W, distance row D, and coefficient row Q therefore witness a finite located approximation of the convex hull, not a host equality of hulls, a selected closest point, quotient subset equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Falsifiable prediction: An accepted $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ consumer that reads a located convex-hull approximation without the displayed real, metric, convex, compact, polytope, Caratheodory, tolerance, distance, and coefficient rows refutes the carrier surface.

Independence witness: real, metric, convexset, compact, polytope, caratheodory-convex-hull: no listed sibling is bijective to $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ because this packet jointly binds finite located tolerance windows, metric distance comparisons, compact finite-read boundaries, polytope approximants, bounded-support hull rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedConvexHull}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedConvexHull}^{\uparrow}$ packet is generated from real-name scalar reads, a metric ambient row, a convex source row, a compact finite-read boundary, a polytope approximant, a Caratheodory bounded-support handoff, located tolerance and distance rows, a coefficient ledger, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedConvexHull}^{\uparrow}}$, seed)

Scope closed: Definition 8389.1, Definition 8389.2, and Definition 8389.3 bind the finite located approximation carrier and its NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedConvexHull}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.BishopLocatedConvexHullUp`

Bridge status: none

Not claimed: This chapter does not close host convex-hull equality, nearest-point selection, quotient subset equality, separation, ambient Real completeness, countable choice, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, located distance comparison, finite approximation exactness, support handoff, transport, provenance, and local naming for BEDC.Derived.BishopLocatedConvexHullUp.

Chapter 8390

A Concrete Naming Certificate for $\mathsf{FiniteIntervalChainSelection}^{\uparrow}$

Following Bishop and Bridges on nested intervals and constructive real approximation, the FiniteIntervalChainSelecti packet records the finite L10 surface by which a nested dyadic interval chain exposes a cofinal inspected subchain before Real and Completion consumers read a limit-facing seal. It sits over the dyadic arithmetic of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the nested interval selector route of chapter 6784. The packet names only the finite displayed rows needed by downstream Bolzano–Weierstrass, nested-interval, and completion arguments; it is not countable choice, quotient stream equality, a selected host limit, interval compactness, or real trichotomy.

Definition 8390.1 (Finite interval chain selection carrier)**.** *A* $\mathsf{FiniteIntervalChainSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, C, D, W, R, E, H, T, P, N)$$

with the following displayed rows:

1. *I is the nested dyadic interval row, recording the finite list of retained endpoint pairs and the inclusion certificates between adjacent rows.*
2. *C is the cofinal unary index row, recording the inspected stages and the monotone refinement map from requested windows to retained interval rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter-budget row, carrying dyadic upper bounds for retained interval lengths at the inspected stages.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the queried stages of the chain and cofinal map.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows, so endpoint and midpoint observations are regular rational reads rather than host real values.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the nested row, cofinal row, diameter ledger, stream window, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the nested rows, cofinal index row, diameter ledger, window, readback, seal, provenance, replay, and local naming rows.*
8. *T records displayed* Cont *replay from a requested precision through C, D, W, R, E and back to the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalChainSelection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalChainSelection}^{\uparrow}}$ *row.*

The carrier exports this finite chain-selection packet and no coordinate for a countable-choice functional, quotient stream equality, host-selected limit, trichotomy of real endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8390.2 (Finite interval chain selection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalChainSelection}^{\uparrow}$ *carrier*

$$S = (I, C, D, W, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalChainSelection}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed nested dyadic interval row, cofinal unary index row, diameter-budget row, stream window, regular readback, terminal seal, transport, replay, provenance, and local naming data;*
2. *nesting preservation, requiring every retained row of I exposed through C to remain included in the preceding displayed dyadic interval row;*
3. *cofinal refinement, requiring every requested window in W to be met by some unary index of C whose dyadic diameter budget in D is within the requested tolerance;*
4. *Real seal handoff, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row E to be consumed only after I, C, D, W, R have been displayed and replayed through H, T, P, N;*
5. *nonescape from countable choice, quotient streams, selected host limits, trichotomy, interval compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8390.1. Its analytic rows are exactly the nested dyadic interval row I, cofinal index row C, diameter-budget row D, finite stream window W, regular rational readback R, and terminal Real seal E.

Nesting preservation and cofinal refinement are read from I, C, D, W. The retained interval rows are finite BHist rows, and the cofinal map is a displayed unary index row, so every refinement request is witnessed inside the packet before any Real-facing read is available.

The Real seal handoff then factors through R and E. Regular rational readback supplies the endpoint and midpoint observations for the same finite windows, while E is terminal in the route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally, H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for countable choice, quotient stream equality, a host-selected limit, real trichotomy, interval compactness, or ambient completeness, the local certificate cannot export any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalChainSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntervalChainSelection}^{\uparrow}$ packet is generated from nested dyadic interval rows, a cofinal unary index row, a DyadicRatCoreUp diameter-budget ledger, StreamNameUp finite windows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteIntervalChainSelection}^{\uparrow}}$, seed)

Scope closed: Definition 8390.1 and Definition 8390.2 seed the finite L10 chain-selection route from nested dyadic interval rows through cofinal inspected windows, dyadic diameter budgets, regular rational readback, and the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteIntervalChainSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient stream equality, selected host limits, real trichotomy, interval compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nesting preservation, cofinal refinement, diameter-budget exactness, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalChainSelectio

Chapter 8391

A Concrete Naming Certificate for $\mathsf{LUDecomposition}^{\uparrow}$

Following Golub and Van Loan on matrix computations, the $\mathsf{LUDecomposition}^{\uparrow}$ packet records Gaussian elimination as a finite lower-upper factorization ledger. It consumes the matrix carrier of chapter 49, the vector-space source of chapter 47, the linear-map presentation of chapter 48, the determinant boundary of chapter 84, and the rank accounting surface of chapter 3438. The packet names only displayed pivot rows, row-update ledgers, triangular factors, determinant sign and diagonal-product handoffs, rank-sensitive zero-pivot boundaries, transport, replay, provenance, and local NameCert data; it does not import host Gaussian elimination, quotient matrix equality, selected inverses outside the pivot ledger, or an ambient matrix algorithm.

Definition 8391.1 (LU decomposition carrier)**.** *An* $\mathsf{LUDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (V, A, \Pi, E, L_0, U_0, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite* $\mathsf{VecSpace}^{\uparrow}$ *source and target row on which the matrix acts.*
2. *A is the consumed* $\mathsf{Mat}^{\uparrow}$ *row for the square matrix being factored, together with its* $\mathsf{LinMap}^{\uparrow}$ *action readback.*
3. *Π is the permutation and pivot-order ledger, listing the finite pivot rows and the row exchanges actually used.*
4. *E is the elimination ledger. For each pivot stage it records the displayed multiplier row, the affected row interval, and the resulting entry-update equations.*
5. *L_0 is the lower-triangular factor row, with unit diagonal entries and multiplier entries read only from E.*
6. *U_0 is the upper-triangular factor row produced by the same pivot sequence.*
7. *S is the triangular exactness row saying that the displayed permutation of A is replayed by the product L_0U_0 through the same matrix and linear-map rows.*
8. *D is the determinant boundary row, carrying the permutation sign and the diagonal-product readback through* $\mathsf{Determinant}^{\uparrow}$*.*
9. *R is the rank-sensitive boundary row, read through* $\mathsf{RankNullity}^{\uparrow}$*, that records where a zero-pivot branch stops, swaps, or exposes rank deficiency.*

10. *H records componentwise* hsame *transport for the vector-space, matrix, pivot, elimination, triangular-factor, exactness, determinant, rank, replay, provenance, and local-name rows.*
11. *C records displayed* Cont *replay from an LU-facing consumer back through $V, A, \Pi, E, L_0, U_0, S, D, R, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{LUDecomposition}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{RankNullity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LUDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host field-inversion oracle, quotient row-equivalence theorem, unlogged pivot choice, selected inverse matrix, spectral decomposition, or ambient Gaussian-elimination procedure.

Theorem 8391.2 (LU decomposition NameCert obligations)**.** *For every accepted* $\mathsf{LUDecomposition}^{\uparrow}$ *carrier*

$$L = (V, A, \Pi, E, L_0, U_0, S, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LUDecomposition}^{\uparrow}}$ *surface consists exactly of finite vector-space admission through V, matrix and linear-map admission through A, pivot-order admission through Π, elimination-ledger admission through E, lower-triangular and upper-triangular factor admission through L_0, U_0, triangular exactness through S, determinant boundary readback through D, rank-sensitive zero-pivot boundary readback through R, componentwise transport through H, displayed replay through C, sibling provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows V and A display the finite vector-space, matrix, and linear-map surface before any pivot row is visible.

The pivot and elimination data are then exactly Π, E. These rows determine the multiplier entries in L_0 and the updated upper-triangular entries in U_0, so no separate elimination trace or hidden inverse can be read.

The rows S, D, R attach the triangular product, determinant handoff, and rank-sensitive boundary to the same displayed route. Finally H, C, P, N transport, replay, source, and name only those rows. Hence the listed inventory exhausts the local NameCert obligations. □

Theorem 8391.3 (LU decomposition pivot ledger)**.** *Every finite linear-system, determinant, rank, or matrix-factorization consumer of an accepted* $\mathsf{LUDecomposition}^{\uparrow}$ *packet must first read the pivot route*

$$V, A, \Pi, E, L_0, U_0, S, D, R, H, C, P, N.$$

At each stage the pivot row in Π precedes the multiplier and row-update rows in E, and the lower and upper triangular factors expose only entries recorded by that same finite route.

Proof. Start with the carrier projection. The source vector-space and matrix rows fix the finite domain in which pivoting can be read.

Read Π before E. The pivot order and row-exchange ledger determine which update rows are admissible, while E records the multipliers and row transformations that later appear in L_0 and U_0.

The triangular exactness and support rows then preserve the same route. Since the carrier contains no additional pivot-choice coordinate, no consumer can replace this route by a host Gaussian-elimination algorithm, quotient row-equivalence theorem, or selected inverse outside the ledger. □

Theorem 8391.4 (LU decomposition triangular factor route)**.** *The triangular factor read exported by an accepted* $\mathsf{LUDecomposition}^{\uparrow}$ *packet factors through*

$$\mathsf{Mat}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{LinMap}^{\uparrow} \longmapsto \Pi, E, L_0, U_0, S \longmapsto H, C, P, N.$$

The row L_0 *is lower triangular with unit diagonal because its off-diagonal entries are the displayed multiplier rows from* E*, and the row* U_0 *is upper triangular because every lower-entry cancellation is recorded in the same elimination ledger.*

Proof. Read the matrix, vector-space, and linear-map sources. These rows determine the square finite matrix presentation against which a triangular factorization can be checked.

Next read the pivot and elimination ledgers. The lower factor receives only the multiplier entries already named in E, and the upper factor receives only the updated rows produced by replaying those same elimination equations.

The exactness row S ties the displayed product L_0U_0 to the permuted source matrix. The support rows H, C, P, N preserve that factor route and add no host equality of matrices, unlogged row operation, or external triangularization theorem. □

Theorem 8391.5 (LU decomposition determinant boundary)**.** *Every determinant read from an accepted* $\mathsf{LUDecomposition}^{\uparrow}$ *packet factors through the permutation sign row in* Π*, the upper-triangular diagonal row of* U_0*, the triangular exactness row* S*, and the determinant handoff row* D*. The read is therefore a displayed boundary through* $\mathsf{Determinant}^{\uparrow}$*, not a host determinant algorithm or an unrestricted row-operation principle.*

Proof. Begin with the pivot ledger. The row Π records the finite permutation and hence the sign contribution available to the determinant boundary.

Read the triangular rows next. Since L_0 has unit diagonal by the displayed elimination ledger, the determinant contribution is routed through the diagonal entries of U_0 together with the exactness row S.

The handoff row D is the only determinant-facing coordinate in the carrier. Transport and replay through H, C, P, N preserve that coordinate, so a consumer cannot obtain an ambient determinant computation, hidden row-equivalence theorem, or quotient matrix equality from the LU packet. □

Falsifiable prediction: An accepted $\mathsf{LUDecomposition}^{\uparrow}$ consumer is refuted if it reads a triangular factor, determinant boundary, or rank-sensitive zero-pivot result without first exposing the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{RankNullity}^{\uparrow}$, pivot, elimination, exactness, transport, replay, provenance, and local-name rows.

Independence witness: matrix, vecspace, linearmap, determinant, ranknullity: none is bijective to $\mathsf{LUDecomposition}^{\uparrow}$ because the LU packet jointly carries a pivot ledger, an elimination ledger, two triangular factor rows, determinant-boundary data, rank-sensitive zero-pivot data, and structural support rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LUDecomposition}^{\uparrow}}$.

Constructive story: An $\mathsf{LUDecomposition}^{\uparrow}$ packet is generated from MatrixUp, VecSpaceUp, and LinearMapUp source rows, a finite permutation and pivot ledger, displayed elimination-update equations, lower and upper triangular factor rows, triangular exactness, DeterminantUp sign and diagonal-product readback, RankNullityUp zero-pivot boundary data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LUDecomposition}^{\uparrow}}$, seed)

Scope closed: Definition 8391.1, Definition 8391.2, Definition 8391.3, Definition 8391.4, and Definition 8391.5 seed the finite LU factorization route over matrix, vector-space, linear-map, determinant, rank-nullity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{LUDecomposition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host Gaussian elimination, quotient row-equivalence, arbitrary pivot selection outside the displayed ledger, inverse-matrix construction, spectral decomposition, or ambient matrix-algorithm correctness.

Upgrade path: Obligation closure requires checked carrier admission, pivot-ledger exactness, elimination-update replay, triangular-factor exactness, determinant boundary soundness, rank-sensitive zero-pivot boundary soundness, transport, replay, provenance, and local naming for BEDC.Derived.LUDecompositionUp.

Chapter 8392

A Concrete Naming Certificate for $\mathsf{FiniteTailSandwich}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteTailSandwich}^{\uparrow}$ packet records the finite tail-squeeze data by which a middle regular rational name is read between lower and upper tail rows at a common dyadic budget. It sits over the arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real seal of chapter 38, and the two-sided regular-Cauchy bracketing route of chapter 5874. The packet gives Real-and-Completion arguments a displayed finite tail sandwich without using quotient sequence equality, selected subsequences, host limits, or ambient completeness.

Definition 8392.1 (Finite tail sandwich carrier)**.** *A* $\mathsf{FiniteTailSandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, M, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower tail row, naming the inspected lower* $\mathsf{RegSeqRat}^{\uparrow}$ *values and their finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *M is the middle tail row, naming the inspected middle* $\mathsf{RegSeqRat}^{\uparrow}$ *values that are to be squeezed.*
3. *U is the upper tail row, naming the inspected upper* $\mathsf{RegSeqRat}^{\uparrow}$ *values and their finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger, recording the shared tail index, tolerance cell, lower–middle comparison, middle–upper comparison, and upper–lower width bound for each displayed read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger proving that L, M, U are read on the same finite tail schedule.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which downstream regular-Cauchy consumers reread the middle row only after the two-sided tail comparisons have been exposed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the lower, middle, upper, budget, window, and readback rows have been replayed.*
8. *H records componentwise* hsame *transport for L, M, U, D, W, R, E and the structural rows.*
9. *C records displayed* Cont *replay routes for rereading the same finite sandwich data.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteTailSandwich}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTailSandwich}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host real equality, quotient stream equality, a selected cofinal subsequence, trichotomy, a least upper bound, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8392.2 (Finite tail sandwich NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTailSandwich}^{\uparrow}$ *carrier $F = (L, M, U, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteTailSandwich}^{\uparrow}}$ *surface consists of carrier admission for the lower, middle, and upper tail rows; common-window exposure through W; common dyadic-budget visibility through D; regular rational readback through R; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance; and local naming. No obligation may replace these rows by a host limit, quotient sequence equality, selected subsequence, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 8392.1. The analytic rows are exactly L, M, U, D, W, R, E, so every admissible obligation must be read from the displayed lower, middle, upper, budget, window, readback, and seal coordinates.

The common-tail condition is enforced by W and D. The window row fixes the finite tail schedule shared by the three regular rational rows, while the dyadic ledger records the lower–middle and middle–upper comparisons at the same tolerance cell.

The structural rows H, C, P, N preserve this inventory. Transport keeps the same component rows visible, continuation replay rereads them in the displayed order, provenance names the packet source, and the local name records no additional analytic coordinate. Hence the listed rows exhaust the local NameCert surface and exclude the non-constructive escape routes named in the statement. □

Theorem 8392.3 (Finite tail sandwich RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the middle row in an accepted* $\mathsf{FiniteTailSandwich}^{\uparrow}$ *packet factors through the displayed route*

$$L \longmapsto M \longmapsto U \longmapsto D \longmapsto W \longmapsto R \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

In particular, the handoff exposes a finite lower–middle–upper tail squeeze at a common dyadic budget, not a host limit, quotient sequence equality, selected subsequence, or ambient completeness principle.

Proof. Begin with the lower, middle, and upper rows L, M, U. They supply only finite $\mathsf{RegSeqRat}^{\uparrow}$ reads and finite $\mathsf{StreamName}^{\uparrow}$ windows, so the middle row is never detached from its two bounding rows.

Next read D and W. The dyadic ledger stores the two-sided comparison and width budget at the inspected tolerance, while the window ledger records that all three rows are read on the same finite tail schedule.

Only after those rows are visible can a consumer reach R and E. The readback row makes the middle regular rational name available to downstream regular-Cauchy consumers, and the terminal seal exposes the Real-facing boundary. The structural rows H, C, P, N then transport, replay, source, and name precisely the same route, so no host limit or completeness coordinate is introduced. □

The chapter is the finite-tail sibling of the broader regular-Cauchy sandwich certificate in chapter 5874. That sibling records the two-sided regular-Cauchy bracketing route as a full certificate; the present packet isolates the common-tail budget, window, and middle-row readback used when a later Real-and-Completion argument only needs the finite squeeze data.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailSandwich}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailSandwich}^{\uparrow}$ packet is generated from lower, middle, and upper RegSeqRat tail rows, a common DyadicRatCore budget ledger, StreamName windows, RegSeqRat readback, a terminal Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTailSandwich}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8392.1, Definition 8392.2, and Definition 8392.3 seed the finite lower–middle–upper tail squeeze surface for Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTailSandwich}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient sequence equality, selected subsequences, host limits, trichotomy, least-upper-bound reasoning, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-tail window exposure, dyadic two-sided budget visibility, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTailSandwichUp.

Chapter 8393

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$

Following Bishop and Bridges, a regular completion argument uses a displayed modulus before it treats a Cauchy name as completion-facing data. The $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ packet records that finite modulus surface between the dyadic arithmetic of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the effective Cauchy completion carrier of chapter 5930, and the terminal real-name seal of chapter 38. The packet names the regularity budget that a completion consumer may inspect; it is not a quotient of Cauchy streams, a selected host limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8393.1 (Bishop regular completion-modulus carrier)**.** *A* $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, C, E, B, H, T, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, listing the dyadic radii and comparison rows used by the requested completion precision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing exactly the inspected entries of the regular Cauchy name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same stream window and dyadic tolerance ledger.*
4. *C is the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *admission row, recording that the displayed windows and tolerances form the completion-facing Cauchy route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, S, R, C have all been displayed.*
6. *B is the Bishop regularity-modulus row, assigning each requested precision a finite inspection bound and a dyadic tolerance budget.*
7. *H records componentwise* hsame *transport for the dyadic ledger, stream window, regular readback, completion row, Real seal, modulus row, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a precision request through B, D, S, R, C, E.*

9. *P is the* Pkg *provenance over* BishopRegularCompletionModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, EffectiveCauchyCompletion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopRegularCompletionModulus}^{\uparrow}}$ *row.*

The carrier exports only this finite modulus packet. It has no coordinate for quotient stream equality, a selected representative in a host completion, an infinite choice functional, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8393.2 (Bishop regular completion-modulus NameCert obligations)**.** *For every accepted* BishopRegularCompletionModulus$^{\uparrow}$ *carrier*

$$M = (D, S, R, C, E, B, H, T, P, N),$$

the local NameCert$_{\text{BishopRegularCompletionModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, stream window, regular readback, effective-completion admission, Real seal, Bishop modulus, transport, replay, provenance, and local naming rows;*
2. *finite-window adequacy, requiring every precision request in B to open the displayed S window before the regular readback row R is consumed;*
3. *dyadic tolerance exactness, requiring the row D to supply the budget used by both R and the completion admission row C;*
4. *Real seal nonescape, requiring E to be read only after D, S, R, C, B have been displayed and replayed through H, T, P, N;*
5. *exclusion of quotient stream equality, selected completed representatives, countable choice, trichotomy, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8393.1. Its analytic rows are exactly D, S, R, C, E, B, with H, T, P, N supplying transport, replay, provenance, and local naming.

The modulus row B is the only row assigning a finite inspection bound to a requested precision. Once B is read, the consumer must open the corresponding stream window S and dyadic tolerance ledger D before it can consume the regular readback row R.

The completion-facing row C is downstream of D, S, R, B. Thus the packet presents an effective Cauchy completion route through displayed finite data rather than through a quotient of Cauchy names.

Finally, the terminal seal E is reached only after the same rows have been replayed. Since the carrier contains no coordinate for quotient equality, a selected host representative, countable choice, trichotomy, or ambient completeness, those principles are outside the local certificate. □

Theorem 8393.3 (Bishop regular completion-modulus window route)**.** *Every* BishopRegularCompletionModulus$^{\uparrow}$ *precision request first factors through the* StreamName$^{\uparrow}$ *finite-window row and the* RegSeqRat$^{\uparrow}$ *readback row of chapter 126 and chapter 301. The dyadic tolerance ledger of chapter 291 supplies the same finite budget before the request may be treated as an* EffectiveCauchyCompletion$^{\uparrow}$ *admission.*

Proof. Apply Definition 8393.2. The finite-window adequacy clause makes S the first observation row opened by a precision request, and the dyadic tolerance exactness clause makes D the budget row shared by the readback and completion admission.

The row R reads the displayed window as regular rational data, while C is available only after D, S, R, B. Therefore the effective-completion route cannot bypass the StreamName and RegSeqRat sibling rows. □

Theorem 8393.4 (Bishop regular completion-modulus Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ *packet reads the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{EffectiveCauchyCompletion}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $D, S, R, C, E, B, H, T, P, N$*. The route supplies a finite regular completion-modulus witness and no completed-real theorem, quotient representative, selected limit, countable-choice functional, trichotomy, or ambient completeness principle.*

Proof. Apply Definition 8393.3. The route already forces the precision request through dyadic tolerance, finite stream inspection, regular rational readback, and effective-completion admission.

The Real seal nonescape clause of Definition 8393.2 places E after those rows and after the Bishop modulus row B. Rows H, T, P, N preserve and replay the same displayed route.

Hence a Real-facing consumer receives only the finite regular completion-modulus packet. Any claim about completed-real existence, quotient representatives, selected limits, countable choice, trichotomy, or ambient completeness is absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ read exposes a DyadicRatCore tolerance ledger, StreamName finite-window row, RegSeqRat readback row, EffectiveCauchyCompletion admission row, RealUp seal, Bishop modulus row, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, effective-cauchy-completion, real: $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ is not identical to any listed sibling because it binds the precision-to-window modulus before effective completion admission and terminal RealUp sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletionModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, EffectiveCauchyCompletion admission, RealUp seal, Bishop regularity modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8393.1, Definition 8393.2, Definition 8393.3, and Definition 8393.4 seed the finite regular completion-modulus route from precision requests to the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed-real existence, quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window adequacy, dyadic tolerance exactness, effective-completion admission, RealUp seal nonescape, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletion

Chapter 8394

A Concrete Naming Certificate for $\mathsf{CompactUniformGraphModulus}^{\uparrow}$

Following Bishop and Bridges on compact metric spaces and uniform continuity, the $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ packet records the finite graph-evaluation route by which a continuous map on a compact source yields a displayed uniform-modulus handoff. It sits over the compact finite-net rows of chapter 129, the distance rows of chapter 57, the graph and pointwise-modulus rows of chapter 130, the uniform-modulus consumer of chapter 369, and the value readback route through chapter 126, chapter 301, and chapter 38. The packet names the compact graph-modulus bridge as finite NameCert data; it is not open-cover compactness, a host function graph, a selected net point outside the carrier, quotient equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8394.1 (Compact uniform graph-modulus carrier)**.** *A* $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, M, F, P, U, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, listing the compact source centers and coverage witnesses inspected by the graph-modulus read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source-and-target distance row used to compare source points, selected centers, and target values.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-evaluation row, recording the finite pointwise graph windows opened at the centers of K.*
4. *P is the pointwise-modulus row attached to the same graph windows.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained by folding the compact-net and pointwise-modulus data into a single tolerance budget.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for graph values and tolerance reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for value-facing consumers.*
9. *H records componentwise* hsame *transport for compact, metric, graph, pointwise-modulus, uniform-modulus, stream, readback, Real-seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a requested tolerance through K, M, F, P, U, S, R, E.*
11. *Q is the* Pkg *provenance over* $\mathsf{CompactUniformGraphModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformGraphModulus}^{\uparrow}}$ *row.*

The carrier exports no coordinate for a host graph subset, open-cover compactness proof, global choice of centers, quotient equality of maps, selected modulus outside U, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8394.2 (Compact uniform graph-modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ *carrier*

$$G = (K, M, F, P, U, S, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformGraphModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for compact finite-net, metric, continuous graph, pointwise-modulus, uniform-modulus, stream-window, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *graph-window adequacy, requiring each graph value used by F and P to be exposed through the displayed finite window S and read back through R;*
3. *compact fold exactness, requiring the uniform-modulus row U to be obtained only from the compact finite-net row K, metric row M, graph row F, and pointwise-modulus row P;*
4. *Real seal nonescape, requiring the terminal row E to be consumed only after K, M, F, P, U, S, R have all been displayed and replayed through H, C, Q, N;*
5. *exclusion of host graph subsets, open-cover compactness, global center choice, quotient map equality, selected external moduli, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8394.1. The analytic rows are precisely K, M, F, P, U, S, R, E, and the structural rows H, C, Q, N preserve and replay the same route.

The graph-window adequacy clause forces pointwise graph reads through S and R. Thus a continuous-map value is consumed as finite StreamName and RegSeqRat data, not as a host graph element.

The compact fold exactness clause then makes U downstream from the compact finite net K, metric comparisons M, graph values F, and pointwise moduli P. This is the finite graph-modulus route used by the compact-uniform-continuity construction.

Finally, E is terminal and only Real-facing. Since no row stores open-cover compactness, a host graph subset, a global choice functional, quotient map equality, an external selected modulus, or ambient completeness, the local certificate cannot export those principles. □

Theorem 8394.3 (Compact uniform graph-modulus finite route)**.** *Every* $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before reading a value-facing or modulus-facing conclusion. In particular, the compact finite-net row, graph-evaluation row, pointwise-modulus row, uniform-modulus row, and Real seal are all displayed rows of the same packet.

Proof. Apply Definition 8394.2. Carrier admission lists the compact, metric, graph, pointwise-modulus, uniform-modulus, stream, readback, and Real-seal rows as the whole analytic surface.

Graph-window adequacy routes value reads through S and R. Compact fold exactness routes the modulus read through K, M, F, P before U is available.

The terminal Real seal is then downstream of the same finite rows. Therefore any value-facing or modulus-facing consumer receives the displayed graph-modulus route and no detached compactness or host graph object. □

Theorem 8394.4 (Compact uniform graph-modulus Real seal)**.** *For an accepted* $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ *carrier, every* $\mathsf{Real}^{\uparrow}$ *value read and every* $\mathsf{UniformModulus}^{\uparrow}$ *handoff is bounded by the displayed compact-net, metric, graph-evaluation, pointwise-modulus, stream-window, and regular-readback rows. The packet can be cited by compact-uniform-continuity consumers, but it exports no open-cover compactness theorem, host graph subset, quotient map equality, selected external modulus, global choice of centers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8394.3. The value read has already passed through S, R, E, and the modulus handoff has already passed through K, M, F, P, U.

Rows H, C, Q, N add only componentwise transport, replay, provenance, and local naming. They do not add a host graph, a separate open-cover compactness proof, or a choice functional.

Thus the Real-facing and uniform-modulus-facing conclusions are bounded by the displayed finite packet. Claims about host graph subsets, quotient equality of maps, selected external moduli, global center choice, or ambient completeness lie outside the certificate. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ read exposes compact finite-net, metric, continuous graph, pointwise-modulus, uniform-modulus, StreamName, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Independence witness: compactmetric, metric, continuousmap, uniformmodulus, streamname, regseqrat, real: $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ is not identical to any listed sibling because it binds finite graph-evaluation rows and pointwise moduli into a compact uniform-modulus handoff before terminal RealUp sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformGraphModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformGraphModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, Metric distance rows, ContinuousMap graph-evaluation rows, pointwise-modulus rows, a UniformModulus handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformGraphModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8394.1, Definition 8394.2, Definition 8394.3, and Definition 8394.4 seed the compact graph-evaluation route from compact finite nets and pointwise moduli to a UniformModulus and RealUp handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformGraphModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host graph subsets, quotient equality of maps, selected external moduli, global center choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph-window adequacy, compact fold exactness, UniformModulus handoff, RealUp seal nonescape, replay, provenance, and local naming for BEDC.Derived.CompactUniformGraphModulu

Chapter 8395

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompactness}^{\uparrow}$

Following Bishop–Bridges constructive compactness, the $\mathsf{BishopLocatedCompactness}^{\uparrow}$ packet records the finite-net and located-distance certificate by which a constructive compact metric source can be consumed without changing into an open-cover principle. It sits after the complete metric service of chapter 128, the finite-net surface of chapter 127, the located subset interface of chapter 1541, the ambient metric-space carrier of chapter 4230, and the real-name boundary of chapter 38. Completion and compact-metric consumers receive a displayed located finite-net package, not open-cover compactness, not sequential compactness without choice-like data, and not host selected limit points.

Definition 8395.1 (Bishop located compactness carrier)**.** *A* $\mathsf{BishopLocatedCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, C, T, L, W, R, D, E, H, Q, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the distance classifier and metric-law ledger used by every finite-net request.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the displayed Cauchy-readback boundary for completion-facing consumers.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, storing finite center bundles, coverage-distance rows, and rational tolerance comparisons.*
4. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row for the compact-facing located subset or whole-space admission surface.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that names the finite observation schedule for net and distance reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance and tolerance data.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row for dyadic radius and coverage comparisons.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for displayed distance or modulus outputs.*
9. *H records componentwise* hsame *transport for the metric, complete, finite-net, located-set, stream, rational, dyadic, real, replay, provenance, and local naming rows.*
10. *Q records displayed* Cont *replay from a compactness tolerance request through $M, C, T, L, W, R, D, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, selected subsequences, host compact subsets, quotient completion, or chosen limit points.

Theorem 8395.2 (Bishop located compactness net obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompactness}^{\uparrow}$ *carrier*

$$K = (M, C, T, L, W, R, D, E, H, Q, P, N),$$

each compactness-facing read must first expose the metric source M, the complete-metric service row C, the totally bounded finite-net row T, the located-set row L, the stream window W, the regular-rational readback R, the dyadic ledger D, and the real seal E, together with the support rows H, Q, P, N. The resulting surface is a Bishop-style located finite-net certificate for downstream compact-metric and completion routes.

Proof. Project the carrier of Definition 8395.1. The finite-net data are exactly the $\mathsf{TotallyBounded}^{\uparrow}$ row T, the located-distance data are exactly the $\mathsf{LocatedSet}^{\uparrow}$ row L, and both are typed over the same $\mathsf{Met}^{\uparrow}$ source M.

The completion-facing service is read only through C. It supplies the displayed Cauchy boundary for later consumers, while W, R, D, E provide the finite stream, regular-rational, dyadic, and real-seal readbacks attached to the net and located-distance rows.

The support rows H, Q, P, N preserve this route by componentwise transport, displayed replay, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, choice-free sequential compactness, or selected host limit points, the obligation surface is exhausted by the displayed finite-net projection. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedCompactness}^{\uparrow}$ read exposes MetricSpace, CompleteMetric, TotallyBounded, LocatedSet, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows before any compactness-facing conclusion is consumed.

Independence witness: CompleteMetric, TotallyBounded, LocatedSet, MetricSpace, StreamName, RegSeqRat, DyadicRatCore, and Real: no listed sibling is bijective to $\mathsf{BishopLocatedCompactness}^{\uparrow}$ because this packet jointly carries the finite-net cover, located-distance admission, completion-facing service, rational and dyadic readback, real seal, transport, replay, provenance, and local naming route while refusing open-cover compactness and selected limit points.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompactness}^{\uparrow}$ packet is generated from a MetricSpace source row, CompleteMetric service row, TotallyBounded finite-net row, LocatedSet distance row, StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 8395.1 and Definition 8395.2 seed the constructive located finite-net certificate over the metric, completion, total-boundedness, located-set, stream, rational, dyadic, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness without choice-like data, selected limit points, quotient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net projection, located-distance routing, completion service exposure, StreamName and RegSeqRat readback, DyadicRatCore ledgering, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompactnessUp.

Chapter 8396

A Concrete Naming Certificate for $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ packet records the finite route by which a regular sequence displays a cluster witness before a Real or Completion consumer reads a seal. It sits over the regular-sequence source of chapter 5965, the finite observation windows of chapter 5916, the cofinal extraction route of chapter 5125, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet names a displayed cluster witness for a regular-sequence source without countable choice, quotient equality of sequence names, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8396.1 (Regular-sequence cluster-witness carrier)**.** *A* $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, F, E, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row, retaining the regular-sequence classifier and the source-window names used by the cluster read.*
2. *F is the* $\mathsf{FiniteStream}^{\uparrow}$ *window ledger, recording the finite source windows inspected for the cluster witness.*
3. *E is the* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *cofinal extraction row, naming the monotone cofinal schedule by which the displayed windows are read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii and comparison slack used in the cluster-window checks.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the rational observations read from the extracted windows.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after S, F, E, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, finite windows, extractor, dyadic tolerance, regular readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, F, E, D, R \longmapsto Q.$$

9. *P is the* Pkg *provenance over* RegularSequenceClusterWitness$^{\uparrow}$, RegularSequenceSpace$^{\uparrow}$, FiniteStream$^{\uparrow}$, CauchySubsequenceExtractor$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularSequenceClusterWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, E, D, R, Q, H, C, P, N$. It has no coordinate for a choice-selected subsequence, quotient equality of sequence names, a detached cluster point, host compactness, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8396.2 (Regular-sequence cluster-witness NameCert obligations)**.** *For every accepted* RegularSequenceClusterWitness$^{\uparrow}$ *carrier*

$$W = (S, F, E, D, R, Q, H, C, P, N),$$

the local NameCert$_{\text{RegularSequenceClusterWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed regular-sequence source, finite windows, cofinal extractor, dyadic tolerance row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *source-window exposure, requiring every cluster observation to cite the same* RegularSequenceSpace$^{\uparrow}$ *source row S and a displayed* FiniteStream$^{\uparrow}$ *window row F;*
3. *cofinal extraction exposure, requiring the selected windows to pass through the* CauchySubsequenceExtractor$^{\uparrow}$ *row E before any readback is used;*
4. *dyadic-tolerance exactness, requiring every cluster comparison to use the displayed* DyadicRatCore$^{\uparrow}$ *row D;*
5. *regular-readback exactness, requiring the comparison values to be read from the* RegSeqRat$^{\uparrow}$ *row R rather than from host sequence equality;*
6. *Real-seal nonescape, requiring Q to be consumed only after the source, finite windows, extractor, tolerance, and readback rows have been displayed;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No obligation is discharged by choice, quotient sequence equality, host compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularSequenceClusterWitnessUp.RegularSequenceClusterWitnessTasteGate`
`single carrier alignment`

Proof. Project the accepted carrier $W = (S, F, E, D, R, Q, H, C, P, N)$. The available observation data are exactly the regular-sequence source, finite stream windows, cofinal extractor, dyadic tolerance row, regular rational readback, and terminal Real seal.

The source-window and extractor obligations follow from the placement of S, F, E. A cluster observation has no independent schedule row unless it first cites the regular-sequence source, a finite window ledger, and the cofinal extraction route.

The comparison obligations are then forced through D and R. The dyadic tolerance row supplies the comparison radius, while the regular rational readback supplies the displayed values inspected inside that radius.

Finally Q is downstream from those rows, and H, C, P, N preserve, replay, source, and name the same finite packet. Since no excluded object is a coordinate of W, the listed rows exhaust the local NameCert surface. □

Theorem 8396.3 (Regular-sequence cluster-witness Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{RegularSequenceSpace}^{\uparrow} \longmapsto \mathsf{FiniteStream}^{\uparrow} \longmapsto \mathsf{CauchySubsequenceExtractor}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \mapsto$$

The handoff supplies a displayed cluster witness over finite windows and a cofinal extractor; it does not select a cluster point, identify quotient sequence names, invoke countable choice, or use host compactness or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularSequenceClusterWitnessUp.RegularSequenceClusterWitnessRealCompletionHandoff`

Proof. Start with the obligation theorem Definition 8396.2. It places the source row S, finite-window ledger F, cofinal extractor E, tolerance row D, and readback row R before the Real seal Q.

The sibling routes provide the stated handoff surface. The regular-sequence space supplies the source classifier, the finite-stream chapter supplies inspected windows, and the cofinal extractor supplies the displayed monotone schedule before any rational readback is consumed.

The dyadic and regular-rational rows then determine the finite comparison read. Transport, replay, provenance, and local naming preserve that route, so the completion-facing consumer receives only the displayed cluster-witness packet and none of the excluded endpoints. □

Theorem 8396.4 (Regular-sequence cluster-witness monotone extraction)**.** *For every accepted* $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ *carrier*

$$W = (S, F, E, D, R, Q, H, C, P, N),$$

each cluster-facing read exposes the cofinal extractor row E *as a monotone finite-window schedule over the* $\mathsf{FiniteStream}^{\uparrow}$ *ledger* F *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *or terminal* $\mathsf{Real}^{\uparrow}$ *seal* Q *is consumed.*

Proof. Project the carrier of Definition 8396.1. Its displayed row order is $S, F, E, D, R, Q, H, C, P, N$, and the replay row records the route

$$S, F, E, D, R \longmapsto Q.$$

Apply Definition 8396.2. The source-window exposure row places every cluster observation over the same source row S and finite-window row F, while the cofinal extraction exposure row requires the selected windows to pass through E before any readback is used.

The regular-readback and Real-seal clauses are downstream of that projection. Thus R can only read rational observations after the monotone finite-window schedule has been displayed over F, and Q can only be consumed after the source, finite windows, extractor, tolerance, and readback rows have already appeared. □

Lean4 ✓ `BEDC.Derived.RegularSequenceClusterWitnessUp.RegularSequenceClusterWitnessCarrier monotone extraction`

Falsifiable prediction: A $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ consumer that reaches a Real or Completion cluster read without displaying the regular-sequence source, finite stream windows, cofinal

extractor, dyadic tolerance row, regular rational readback, transport, replay, provenance, and local naming exits the accepted carrier.

Independence witness: regular-sequence-space, finite-stream, cauchy-subsequence-extractor, dyadicratcore, regseqrat, real: no listed sibling is bijective with $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ because this packet jointly binds source regularity, finite windows, cofinal extraction, dyadic tolerance, rational readback, terminal seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceClusterWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ packet is generated from a RegularSequenceSpace source row, FiniteStream windows, a CauchySubsequenceExtractor cofinal extraction row, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceClusterWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8396.1, Definition 8396.2, and Definition 8396.3 seed the finite cluster-witness route over displayed regular-sequence, finite-window, cofinal-extractor, dyadic-tolerance, rational-readback, Real, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceClusterWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected cluster points, quotient sequence equality, countable choice, host compactness, ambient RealUp completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, cofinal extraction exposure, dyadic-tolerance exactness, regular-readback exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceClusterWitnessUp.

Chapter 8397

A Concrete Naming Certificate for $\mathsf{CholeskyDecomposition}^{\uparrow}$

Following Cholesky and Golub–Van Loan, the $\mathsf{CholeskyDecomposition}^{\uparrow}$ packet records positive-definite matrix factorization as finite BEDC data over displayed matrix, inner-product, triangular-factor, and diagonal-positivity rows. It consumes the matrix carrier of chapter 49, the positive-definite inner-product route of chapter 86, the triangular ledger of chapter 8391, the orthogonal factorization comparison of chapter 4240, and the spectral comparison surface of chapter 6037. The packet names only a displayed positive-definite factorization route; it does not import a host numerical algorithm, quotient matrix equality, pivot selection outside the certificate, an analytic square-root choice, or an ambient completeness principle.

Definition 8397.1 (Cholesky decomposition carrier)**.** *A* $\mathsf{CholeskyDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (M, P, L, D, R, H, C, Q, N)$$

with the following visible rows:

1. *M is the displayed square matrix source, read through the finite row and column interface of* $\mathsf{Mat}^{\uparrow}$*.*
2. *P is the positive-definite witness row, recording the* $\mathsf{InnerProduct}^{\uparrow}$ *comparison and the finite Gram positivity checks available to the factorization consumer.*
3. *L is the lower-triangular factor row, carrying the finite entries and triangular-zero ledger.*
4. *D is the diagonal-positivity row, recording that the diagonal entries used by the triangular factor are positive displayed scalars.*
5. *R is the reconstruction ledger asserting that the displayed matrix source is recovered from the lower-triangular factor, its transpose-facing read, and the diagonal positivity data.*
6. *H records componentwise* hsame *transport for the source, positivity, triangular, diagonal, reconstruction, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the matrix and positivity rows to the factor row, and from the factor row to the reconstruction ledger.*
8. *Q is the* Pkg *provenance over* $\mathsf{CholeskyDecomposition}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{LUDecomposition}^{\uparrow}$*,* $\mathsf{QRDecomposition}^{\uparrow}$*,* $\mathsf{SingularValueDecomposition}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CholeskyDecomposition}^{\uparrow}}$ *row.*
The classifier compares accepted packets componentwise by the displayed rows. No coordinate stores a host floating-point routine, quotient matrix equality, selected pivoting outside the packet, a global spectral theorem, or an unrestricted square-root operation.

Theorem 8397.2 (Cholesky decomposition NameCert obligations)**.** *For every accepted* $\mathsf{CholeskyDecomposition}^{\uparrow}$ *carrier*

$$X = (M, P, L, D, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CholeskyDecomposition}^{\uparrow}}$ *surface consists exactly of matrix-source admission through M, positive-definite exposure through P, lower-triangular factor exposure through L, diagonal positivity through D, reconstruction through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 8397.1. The only mathematical rows before structural transport are M, P, L, D, R, so every accepted read must expose the matrix source, positive-definite witness, lower-triangular factor, diagonal positivity, and reconstruction ledger.

The structural rows H and C preserve those same displayed rows under transport and replay. They do not introduce a second factorization, a host equality principle, or an algorithmic pivot choice.

Finally Q and N record sibling provenance and the local naming row. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 8397.3 (Cholesky decomposition positive-definite route)**.** *Every positive-definite matrix-factorization consumer of an accepted* $\mathsf{CholeskyDecomposition}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Mat}^{\uparrow} \longrightarrow \mathsf{InnerProduct}^{\uparrow} \longrightarrow \mathsf{LUDecomposition}^{\uparrow} \longrightarrow \mathsf{QRDecomposition}^{\uparrow} \longrightarrow \mathsf{SingularValueDecomposition}^{\uparrow}$$

through the displayed rows $M, P, L, D, R, H, C, Q, N$. The Cholesky read therefore exports a lower-triangular positive-diagonal reconstruction certificate, not a host numerical routine, quotient matrix equality, selected pivoting rule, analytic square-root choice, or ambient spectral-completion theorem.

Proof. Start with the matrix source and positive-definite witness rows M and P. They force the consumer to read a displayed finite matrix and the finite positivity checks before any triangular factor can be inspected.

Next read the triangular and diagonal rows L and D. The lower-triangular shape is tied to the LU-style triangular ledger, while the positive diagonal row keeps the Cholesky factor distinct from a generic triangular decomposition.

Finally inspect the reconstruction and structural rows R, H, C, Q, N. They replay the same factorization route and record the QR and singular-value comparison surfaces only as sibling provenance. Thus the exported object is the displayed positive-definite Cholesky route and nothing outside the finite certificate. □

Falsifiable prediction: An accepted $\mathsf{CholeskyDecomposition}^{\uparrow}$ consumer is refuted if it reads a positive-definite factorization without first exposing the MatrixUp source, InnerProductUp positivity witness, lower-triangular factor, diagonal-positivity row, reconstruction ledger, transport, replay, provenance, and local naming rows.

Independence witness: matrix, innerproduct, lu-decomposition, qr-decomposition, singular-value-decomposition: none is bijective to $\mathsf{CholeskyDecomposition}^{\uparrow}$ because the Cholesky packet

jointly carries positive-definite witness data, lower-triangular factor data, diagonal positivity, reconstruction, and structural certificate rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CholeskyDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CholeskyDecomposition}^{\uparrow}$ packet is generated from a displayed MatrixUp source, an InnerProductUp positive-definite witness row, a lower-triangular factor row, a diagonal-positivity row, a reconstruction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CholeskyDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8397.1, Definition 8397.2, and Definition 8397.3 seed the finite positive-definite triangular factorization route over MatrixUp, InnerProductUp, LUDecompositionUp, QRDecompositionUp, SingularValueDecompositionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CholeskyDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host numerical Cholesky algorithms, quotient matrix equality, selected pivoting outside the packet, analytic square-root choice, spectral completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-definite witness exposure, triangular-factor exposure, diagonal-positivity exposure, reconstruction replay, transport, provenance, and local naming for BEDC.Derived.CholeskyDecom

Chapter 8398

A Concrete Naming Certificate for $\mathsf{ParacompactLocallyFinite}^{\uparrow}$

Following Bourbaki and Dugundji on paracompact covers and locally finite refinements, the ParacompactLocallyFinit packet records the finite BEDC handoff by which a displayed paracompact cover request is read as a locally finite cover ledger. It sits between the paracompact refinement carrier of chapter 1963, the locally finite cover surface of chapter 7245, and the indexed-open topology rows of chapter 88. The packet names only the displayed cover request, refinement assignment, local finite windows, transport, replay, provenance, and local NameCert rows needed by later partition-of-unity consumers; it does not assert a full partition of unity, a choice-selected refinement theorem, arbitrary subcover extraction, host powerset equality, or open-cover compactness.

Definition 8398.1 (Paracompact locally finite carrier)**.** *A* $\mathsf{ParacompactLocallyFinite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, A, Q, R, L, W, I, H, C, S, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the indexed open operations on which the cover request and local windows are read.*
2. *A is the displayed open-cover request row, naming the cover cells before any refinement or local-finiteness read is available.*
3. *Q is the* $\mathsf{Paracompact}^{\uparrow}$ *refinement source row, carrying the finite paracompact packet whose locally finite refinement ledger is being handed off.*
4. *R is the refinement-assignment row, listing the displayed refinement cells and their assignment back into the cover request A.*
5. *L is the* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *ledger row, recording the local neighbourhood windows and finite incidence lists used to read the refinement as locally finite.*
6. *W is the coverage and containment row, recording that the displayed refinement covers the same source region as A and that every refinement cell is contained in its assigned cover cell.*
7. *I is the downstream interface row consumed by* $\mathsf{PartitionOfUnity}^{\uparrow}$ *only as support-window data, with no weight or sum-to-one coordinate.*

8. *H records componentwise* hsame *transport for the topology, cover request, paracompact source, refinement assignment, locally finite ledger, coverage, downstream interface, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a paracompact cover request through $T, A, Q, R, L, W, I, H, S, N$.*
10. *S is the* Pkg *provenance over* ParacompactLocallyFinite$^{\uparrow}$, Paracompact$^{\uparrow}$, LocallyFiniteCover$^{\uparrow}$, Topology$^{\uparrow}$, PartitionOfUnity$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\mathsf{ParacompactLocallyFinite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, Q, R, L, W, I, H, C, S, N$. It has no coordinate for a selected host-indexed refinement family, arbitrary subcover choice, partition-of-unity weights, countable choice, quotient open equality, or a global paracompactness theorem detached from the displayed packet.

Theorem 8398.2 (Paracompact locally finite NameCert obligations)**.** *For every accepted* ParacompactLocallyFinite *carrier*

$$P = (T, A, Q, R, L, W, I, H, C, S, N),$$

the local NameCert$_{\mathsf{ParacompactLocallyFinite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, cover request, paracompact source, refinement assignment, locally finite ledger, coverage and containment row, downstream interface, transport, replay, provenance, and local naming;*
2. *topology and cover exposure, requiring every refinement read to display T and A before any locally finite ledger is consumed;*
3. *paracompact-source exposure, requiring the refinement assignment R to be read through the displayed* Paracompact$^{\uparrow}$ *packet Q;*
4. *local-finiteness exactness, requiring L to expose the neighbourhood windows and finite incidence lists supplied by* LocallyFiniteCover$^{\uparrow}$ *before the refinement can be used as locally finite;*
5. *coverage and containment exactness, requiring W to record both coverage of the displayed source region and containment inside assigned cover cells;*
6. *downstream-interface discipline, requiring I to expose only support-window data for* PartitionOfUnity$^{\uparrow}$ *and no weights or sum-to-one rows;*
7. hsame, Cont, Pkg, *and local* NameCert *stability, requiring H, C, S, N to preserve exactly the listed rows.*

No local obligation for the paracompact-to-locally-finite handoff is stored outside these rows.

Proof. Carrier admission is projection from Definition 8398.1. The accepted tuple displays exactly the topology source, open-cover request, paracompact packet, refinement assignment, locally finite ledger, coverage and containment row, downstream support interface, transport, replay, provenance, and local naming.

Topology and cover exposure are forced by row order. A refinement read starts with T and A, so the handoff cannot begin from an unlogged host-indexed cover family.

Paracompact-source exposure is the projection to Q and R. The refinement assignment is usable only as a row transported from the displayed Paracompact$^{\uparrow}$ packet, with each refinement cell assigned back into the cover request.

Local-finiteness exactness is the projection to L. The row records neighbourhood windows and finite incidence lists in the sense of chapter 7245; the existence of a refinement family alone is not read as local finiteness.

Coverage and containment are read from W, while the downstream support boundary is read from I. These rows make the handoff available to later partition-of-unity work only as cover and support-window data, not as weights, sums, or selected functions.

Finally H, C, S, N transport, replay, source, and name the same finite route. Since the classifier has no further coordinate, selected refinements, arbitrary subcover choice, partition-of-unity data, countable choice, quotient open equality, and detached global paracompactness cannot be exported by the local certificate. □

Theorem 8398.3 (Paracompact locally finite handoff route)**.** *Every accepted* $\mathsf{ParacompactLocallyFinite}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{Paracompact}^{\uparrow} \longrightarrow \mathsf{LocallyFiniteCover}^{\uparrow}$$

before any partition-of-unity consumer may read support-window data from it. The exported route is a handoff from displayed paracompact refinement rows to displayed locally finite cover windows; it is not a choice-based refinement theorem, an arbitrary open-cover compactness theorem, a full partition of unity, or an ambient point-set topology construction.

Proof. Begin with the carrier of Definition 8398.1. The rows T, A, Q display the topology, cover request, and paracompact source before the refinement assignment is available.

Use the obligation theorem Definition 8398.2. It forces the refinement assignment R, locally finite ledger L, and coverage row W to remain visible before a consumer reads the handoff.

The downstream interface row I is only support-window data. It may be consumed by $\mathsf{PartitionOfUnity}^{\uparrow}$ after the locally finite route is displayed, but the carrier contains no weight, sum-to-one, selected-function, or arbitrary subcover coordinate.

The structural rows H, C, S, N preserve this order, so every exported read factors through the displayed topology, paracompact refinement, and locally finite cover surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{ParacompactLocallyFinite}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ParacompactLocallyFinite}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8398.1, Definition 8398.2, and Definition 8398.3 seed the finite topology, paracompact refinement, locally finite cover, coverage, support-interface, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ParacompactLocallyFinite}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full partition-of-unity construction, choice-based refinement selection, arbitrary subcover extraction, open-cover compactness, quotient open equality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology and cover exposure, paracompact-source exposure, local-finiteness exactness, coverage and containment exactness, downstream-interface discipline, transport stability, replay stability, and non-escape proofs for BEDC.Derived.ParacompactLocallyFiniteUp.

Constructive story: A $\mathsf{ParacompactLocallyFinite}^{\uparrow}$ packet is generated from a $\mathsf{Topology}^{\uparrow}$ source row, an open-cover request row, a $\mathsf{Paracompact}^{\uparrow}$ refinement source row, a refinement assignment, a $\mathsf{LocallyFiniteCover}^{\uparrow}$ local-window ledger, coverage and containment data, a $\mathsf{PartitionOfUnity}^{\uparrow}$ support interface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8399

A Concrete Naming Certificate for $\mathsf{MaxFlowMinCut}^{\uparrow}$

Following Ford–Fulkerson and the finite network-flow tradition, the $\mathsf{MaxFlowMinCut}^{\uparrow}$ packet records the displayed certificate surface for maximum-flow/minimum-cut duality as finite BEDC data. It sits after the residual-capacity and cut rows of chapter 233, the alternating-incidence discipline of chapter 234, the directed graph carrier of chapter 118, the finite list accounting surface of chapter 36, and the natural-number ledger of chapter 28. The packet names a feasible flow, residual reachability ledger, displayed cut, augmenting-path refusal, value/capacity comparison, transport, replay, provenance, and local certificate; it is not a host optimization theorem, not quotient equality of flows, and not an unbounded search principle.

Definition 8399.1 (Max-flow min-cut carrier)**.** *A* $\mathsf{MaxFlowMinCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (G, s, t, U, F, R, A, K, V, M, L, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{Graph}^{\uparrow}$ *and* $\mathsf{NetworkFlow}^{\uparrow}$ *source row, carrying the finite directed graph, source and sink endpoints, and edge inventory.*
2. *s and t are the distinguished endpoint rows for the inspected flow problem.*
3. *U is the capacity ledger, recording the displayed capacity bound for every inspected edge.*
4. *F is the feasible-flow row, exposing edge-flow values, conservation checks, and the flow-value history through* $\mathsf{NetworkFlow}^{\uparrow}$*.*
5. *R is the residual reachability row, recording exactly which vertices are reached from s by residual arcs in the displayed finite residual graph.*
6. *A is the augmenting-path audit row, whose positive entries are finite path witnesses and whose terminal entry is the displayed refusal of any remaining s–t residual path.*
7. *K is the cut row obtained from the residual reachability boundary, with crossing-edge saturation and reverse-edge zero-flow checks displayed edge by edge.*
8. *V is the* $\mathsf{Nat}^{\uparrow}$ *value/capacity comparison ledger, recording that the flow value and the displayed cut capacity have the same finite unary total.*
9. *M is the* $\mathsf{Matching}^{\uparrow}$ *sibling row, exposing the same alternating-path and saturated-boundary discipline to bipartite matching and cover consumers.*

10. *L records the* $\mathsf{List}^{\uparrow}$ *accounting discipline for path lists, reachable-vertex lists, cut-edge lists, and fold-order witnesses.*
11. *H records componentwise* hsame *transport for source, endpoints, capacity, flow, residual reachability, augmenting audit, cut, value ledger, matching sibling, list accounting, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from a max-flow/min-cut query through $G, s, t, U, F, R, A, K, V, M, L, H$.*
13. *P is the* Pkg *provenance over* $\mathsf{MaxFlowMinCut}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MaxFlowMinCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host max-flow theorem, selected optimizer outside the packet, infinite graph, quotient flow equality, or choice over augmenting paths.

Theorem 8399.2 (Max-flow min-cut NameCert obligations)**.** *For every accepted* $\mathsf{MaxFlowMinCut}^{\uparrow}$ *carrier*

$$X = (G, s, t, U, F, R, A, K, V, M, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MaxFlowMinCut}^{\uparrow}}$ *surface consists exactly of network-flow source admission through G, endpoint exposure through s, t, capacity exposure through U, feasible-flow exposure through F, residual reachability through R, augmenting-path audit through A, cut exposure through K, value/capacity comparison through V, matching sibling exposure through M, list accounting through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8399.1. The source, endpoint, capacity, feasible-flow, residual, augmenting, cut, value, matching, and list rows are exactly $G, s, t, U, F, R, A, K, V, M, L$.

The feasible-flow row F is read only after the source and capacity rows. The residual reachability row R and augmenting audit row A then inspect the finite residual graph determined by the same G, s, t, U, F data, so the audit cannot detach from the displayed network.

The cut row K is obtained from the terminal residual boundary. The value ledger V compares the flow value to the capacity of that same cut, while the matching sibling row M exposes the alternating-route discipline without replacing the packet by a matching oracle.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since no carrier coordinate stores a host max-flow theorem, selected optimizer, quotient flow equality, infinite graph, or choice over paths, the listed rows exhaust the NameCert obligations. □

Theorem 8399.3 (Max-flow min-cut augmenting cut handoff)**.** *Every duality-facing consumer of an accepted* $\mathsf{MaxFlowMinCut}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{NetworkFlow}^{\uparrow},\ \mathsf{Matching}^{\uparrow},\ \mathsf{Graph}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{Nat}^{\uparrow} \longmapsto G, s, t, U, F, R, A, K, V, M, L, H, C, P, N.$$

The handoff states only that the displayed feasible flow has no displayed residual augmenting path and that the residual boundary cut has the same displayed finite value. It exports no global optimizer, no hidden residual network, no quotient equality of flows, and no theorem for infinite networks.

Proof. Begin with Definition 8399.2. A duality-facing consumer must expose G, s, t, U, F before it can inspect either an augmenting path or a cut boundary.

Next read R and A. If A contains a positive augmenting witness, it is a finite path list in the displayed residual graph. If the terminal audit row refuses such a path, that refusal is recorded against the same residual reachability boundary R.

Now read K, V, M, L. The cut row K is the boundary determined by R, the value ledger V compares the displayed flow value with the displayed cut capacity, the matching row M keeps the alternating-route sibling visible, and the list row L supplies the finite inventories used by all folds.

Transport, replay, provenance, and local naming through H, C, P, N preserve that route. Thus the consumer receives a finite augmenting-path/cut certificate and no stronger host max-flow/min-cut endpoint. □

Falsifiable prediction: A purported $\mathsf{MaxFlowMinCut}^{\uparrow}$ consumer is refuted if it reaches a maximum-flow/minimum-cut conclusion while omitting any of the displayed network source, endpoints, capacity, feasible-flow, residual reachability, augmenting audit, cut, value/capacity, matching sibling, list accounting, transport, replay, provenance, or local NameCert rows.

Independence witness: networkflow, matching, graph, list, nat: no listed sibling is bijective to $\mathsf{MaxFlowMinCut}^{\uparrow}$ because this packet jointly carries feasible-flow rows, residual reachability, augmenting-path audit, cut boundary, value/capacity comparison, matching sibling exposure, finite list accounting, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MaxFlowMinCut}^{\uparrow}}$.

Constructive story: A $\mathsf{MaxFlowMinCut}^{\uparrow}$ packet is generated from NetworkFlowUp source rows, endpoint rows, capacity rows, feasible-flow rows, residual reachability rows, augmenting-path audit rows, cut rows, NatUp value/capacity ledgers, MatchingUp sibling rows, ListUp accounting rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaxFlowMinCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8399.1, Definition 8399.2, and Definition 8399.3 seed the finite augmenting-path and cut-certificate route over network-flow, matching, graph, list, natural-number, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaxFlowMinCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host max-flow theorem, selected optimizers for arbitrary networks, infinite-network duality, quotient equality of flows, hidden residual networks, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, capacity exposure, feasible-flow exposure, residual reachability exactness, augmenting-path refusal, cut-boundary saturation, value/capacity comparison, Matching and List sibling handoffs, transport, replay, provenance, and local naming for BEDC.Derived.MaxFlowMinCutUp.

Chapter 8400

A Concrete Naming Certificate for MetaCICBoundedConversionFrontier$^{\uparrow}$

The MetaCICBoundedConversionFrontier$^{\uparrow}$ packet realizes the decidability audit map of chapter 9379 as a bounded finite conversion-checker boundary. It is distinct from the conversion-facing frontier of chapter 4097: this packet names the checker fuel, normalization endpoint, comparison row, refusal ledger, transport, replay, provenance, and local certificate before any broader decidability frontier can be cited. It sits beside the bounded equality checker of chapter 752, the fuel trace of chapter 3889, the bounded conversion bridge of chapter 3439, and the normal-form consistency seal of chapter 740.

Falsifiable prediction: Every accepted MetaCICBoundedConversionFrontier$^{\uparrow}$ read exposes checker fuel, bounded normalizer endpoints, structural comparison, finished-normal evidence, bridge handoff, consistency seal, refusal, transport, replay, provenance, and local NameCert rows before any conversion-frontier conclusion is consumed.

Independence witness: bounded-normal-equality-checker, metacic-fuel-trace, bounded-normal-conversion-bridge, normal-form-consistency-seal, metacic-decidable-conversion-frontier: the equality checker records comparison output without the full refusal ledger; the fuel trace records normalization budget without the bridge and seal pair; the bridge compares finished normal forms without naming the checker packet; the seal records closed normal-form safety without the local fuel ledger; and the decidable conversion frontier is a broader consumer of bounded rows. None is bijective to MetaCICBoundedConversionFrontier$^{\uparrow}$.

Definition 8400.1 (MetaCIC bounded conversion frontier carrier)**.** *A* MetaCICBoundedConversionFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, F, N, E, S, R, D, H, C, P, L)$$

with the following displayed rows:

1. *Q is the bounded checker row, recording the two closed terms, the finite fuel value, and the Boolean comparison produced by the checker.*
2. *F is the* MetaCICFuelTrace$^{\uparrow}$ *row, exposing the exact bounded normalizer schedule used by Q.*
3. *N is the bounded normalizer endpoint row, recording the two computed normal forms and the finished-normal evidence available at that fuel.*

4. *E is the structural equality row, reading the endpoint comparison through* Term.*eq rather than through quotient syntax equality.*
5. *S is the* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *row, packaging the local bounded normal-equality check as a finite NameCert surface.*
6. *R is the* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *row, transporting a positive bounded comparison to the finished-normal conversion boundary when the displayed endpoint evidence is present.*
7. *D is the refusal row, naming the excluded endpoints: global closed conversion decidability, arbitrary-fuel completeness, evaluator equality, subject reduction, confluence, quotient syntax equality, and kernel consistency.*
8. *H records componentwise* hsame *transport for checker, fuel, endpoint, comparison, checker-surface, bridge, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bounded conversion query through Q, F, N, E, S, R, D, H.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, F, N, E, S, R, D, H, C, P, L$. It has no coordinate for global conversion decidability, arbitrary-fuel completeness, evaluator equality, subject reduction, confluence, quotient syntax equality, or kernel consistency.

Theorem 8400.2 (MetaCIC bounded conversion frontier obligations)**.** *For every accepted* MetaCICBoundedConvers *carrier*

$$B = (Q, F, N, E, S, R, D, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}}$ *surface consists exactly of checker exposure through Q, fuel-trace exposure through F, endpoint exposure through N, structural comparison through E, bounded equality-checker exposure through S, bounded bridge exposure through R, refusal retention through D, componentwise transport through H, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8400.1. Rows Q, F, N, E, S, R, D are the bounded checker, fuel, endpoint, comparison, checker-surface, bridge, and refusal data.

The checker row Q is read only with its fuel trace F and endpoint row N. The comparison row E therefore compares displayed bounded normal forms, not quotient syntax objects or evaluator outputs.

The bounded equality-checker row S and bridge row R keep positive comparison data inside the finished-normal boundary. The refusal row D remains part of the same packet and prevents the bounded result from becoming global conversion decidability.

Finally H, C, P, L transport, replay, source, and name the same finite rows. Since no carrier coordinate stores arbitrary-fuel completeness, subject reduction, confluence, evaluator equality, quotient syntax equality, or kernel consistency, the listed rows exhaust the NameCert obligations. □

Theorem 8400.3 (MetaCIC bounded conversion frontier nonescape)**.** *Every positive read exported by an accepted* $\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}$ *carrier factors through*

$\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, NormalFormConsistencySe

The read is a finite bounded conversion-checker certificate. It does not export global closed conversion decidability, arbitrary-fuel completeness, evaluator equality, subject reduction, confluence, quotient syntax equality, or kernel consistency.

Proof. Start with Definition 8400.2. A positive read must expose the checker row Q, the fuel trace F, the endpoint row N, and the structural comparison row E before the bounded equality-checker row can be consumed.

Next read the bridge row R. It is attached to the same finished-normal endpoint evidence and to the normal-form consistency seal named in the provenance row P, so it transports only the displayed bounded comparison.

The refusal row D is then still present. It names the global endpoints that the bounded checker does not supply, including closed conversion decidability, arbitrary-fuel completeness, evaluator equality, subject reduction, confluence, quotient syntax equality, and kernel consistency.

Transport and replay through H, C, P, L preserve this boundary. Hence the packet can feed the broader decidability frontier, but it cannot itself be cited as a global conversion decider. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}$ packet is generated from bounded checker rows, MetaCICFuelTraceUp rows, bounded normalizer endpoint rows, Term.eq comparison rows, BoundedNormalEqualityCheckerUp rows, BoundedNormalConversionBridgeUp rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8400.1, Definition 8400.2, and Definition 8400.3 seed the finite bounded conversion-checker frontier over checker, fuel, endpoint, comparison, bridge, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedConversionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global closed conversion decidability, arbitrary-fuel completeness, evaluator equality, subject reduction, confluence, quotient syntax equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, checker exposure, fuel-trace exactness, endpoint exposure, Term.eq comparison, bounded equality-checker compatibility, bridge compatibility, refusal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICBoundedConversionFrontierUp.

Chapter 8401

A Concrete Naming Certificate for $\mathsf{BishopUniformStructure}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley on uniform spaces, the $\mathsf{BishopUniformStructure}^{\uparrow}$ packet records the finite entourage-filter substrate used by the L10 Real and Completion route. It sits between the general entourage discipline of chapter 339, the Real-induced route of chapter 1624, the Cauchy-filter surface of chapter 333, and the completion-facing criteria of chapter 1506. The chapter names only Bishop-style finite uniform-structure data: entourages, filter refinement, Cauchy compatibility, transport, replay, provenance, and local naming.

Definition 8401.1 (Bishop uniform-structure carrier)**.** *A* $\mathsf{BishopUniformStructure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, E, F, R, C, D, S, Q, H, K, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row naming the finite entourage basis, diagonal requests, symmetry requests, and composition requests accepted by the uniform surface.*
2. *E is the entourage-filter row, recording finite upward-closure and finite-intersection witnesses for the displayed entourage family.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row for consumers that read Cauchy data through the entourage filter.*
4. *R is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *comparison row, used only when a Real-induced uniformity supplies the source entourages.*
5. *C is the completion handoff row that names* $\mathsf{MetricCompletion}^{\uparrow}$ *or sibling completion consumers as readers of the already displayed uniform data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger when the source uniformity is metric or Real-induced.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by sequential or completion-facing consumers.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
9. *H records componentwise* hsame *transport for the uniform, entourage-filter, Cauchy-filter, Real-uniform, completion, tolerance, window, readback, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay through the same route.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopUniformStructure}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformStructure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, E, F, R, C, D, S, Q, H, K, P, N$. The carrier has no coordinate for Real completeness, selected limits, quotient uniformities, countable choice, open-cover compactness, or quotient equality of completion points.

Theorem 8401.2 (Bishop uniform-structure NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformStructure}^{\uparrow}$ *carrier*

$$B = (U, E, F, R, C, D, S, Q, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopUniformStructure}^{\uparrow}}$ *surface consists exactly of uniform-source admission, entourage-filter exposure, Cauchy-filter compatibility, optional Real-induced comparison, completion handoff ordering, tolerance-ledger exposure, finite-window readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of B.*

Proof. Project the carrier of Definition 8401.1. The uniform data are exactly U, E: a finite entourage source and the filter witnesses that make those entourages usable by downstream requests.

The Cauchy and completion rows are downstream of that source. A Cauchy-filter consumer reads F only after the entourage-filter row E is displayed, and a completion consumer reads C only as a named reader of the same route.

When the source is Real-induced, the comparison row R and tolerance ledger D supply the finite bridge to chapter 1624. The window and readback rows S, Q expose sequential observations only after the uniform route has been displayed.

The structural rows H, K, P, N transport, replay, source, and name this finite route. Since no listed row stores Real completeness, selected limits, quotient uniformities, countable choice, open-cover compactness, or quotient equality of completion points, none of those objects can discharge a local NameCert obligation. □

Theorem 8401.3 (Bishop filter-entourage induction)**.** *Let $B = (U, E, F, R, C, D, S, Q, H, K, P, N)$ be an accepted* $\mathsf{BishopUniformStructure}^{\uparrow}$ *carrier. Any finite induction over accepted entourage-filter requests preserves the route*

$$\mathsf{UniformSpace}^{\uparrow},\ E,\ \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}$$

and, when the source row is Real-induced, preserves the comparison route through $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$. *The induction exports no Real completeness, selected limit, quotient uniformity, open-cover compactness, or quotient equality of completion points.*

Proof. The base case is a single displayed entourage request. Carrier admission gives the request through U and its filter witness through E, so the Cauchy row F and the completion row C can only read the already displayed uniform surface.

For upward closure, the accepted refinement is another row of the same entourage-filter packet E. Transport through H and replay through K preserve the source row U and keep the Cauchy-filter handoff downstream of the displayed refinement.

For finite intersections, the induction step reads both prior requests from E and records the intersected request as another finite filter witness. The handoff to F and C is therefore still a handoff from the same uniform source, not a new quotient uniformity.

If R is present as the Real-induced comparison row, each inductive step also keeps the tolerance ledger D and the finite observations S, Q attached to the same route. Thus Real-uniform consumers receive only the displayed comparison and readback data.

No induction case creates a coordinate outside $U, E, F, R, C, D, S, Q, H, K, P, N$. Real completeness, selected limits, quotient uniformities, open-cover compactness, and quotient equality of completion points are absent throughout the induction. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformStructure}\uparrow}$.

Constructive story: A $\mathsf{BishopUniformStructure}^{\uparrow}$ packet is generated from a UniformSpace source row, finite entourage-filter witnesses, CauchyFilter compatibility, optional RealUniformStructure comparison, completion handoff ordering, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformStructure}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8401.1, Definition 8401.2, and Definition 8401.3 seed the L10 route from Bishop-style entourage filters to Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformStructure}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, selected limits, quotient uniformities, countable choice, open-cover compactness, quotient equality of completion points, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage-filter exactness, Cauchy-filter compatibility, Real-induced comparison, completion handoff ordering, tolerance exposure, finite-window readback, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformStructureUp.

Chapter 8402

A Concrete Naming Certificate for $\mathsf{AbelTransformation}^{\uparrow}$

Following Abel and Hardy, the $\mathsf{AbelTransformation}^{\uparrow}$ packet records the finite boundary identity that rewrites a displayed coefficient-weighted tail through partial sums, adjacent coefficient differences, and endpoint terms. It sits beside the finite summation-by-parts surface of chapter 5796, the Abel summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. The packet names only a displayed Abel-transform window; it does not import infinite sums, selected limits, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8402.1 (Abel transformation carrier)**.** *An* $\mathsf{AbelTransformation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, P, C, \Delta, B, W, T, R, D, E, H, K, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing the inspected finite term window and its partial-sum cells.*
2. *P is the partial-sum row used by the Abel transform on the same finite interval.*
3. *C is the coefficient-window row, naming the scalar coefficients paired with the partial sums.*
4. *Δ is the adjacent-difference ledger for the coefficient window.*
5. *B is the boundary-term row, recording the two endpoint products of the displayed window.*
6. *W is the finite transform row, collecting the boundary terms and interior difference-weighted partial sums.*
7. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the displayed handoff.*
9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the additions, subtractions, products, and endpoint comparisons in W.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after $S, P, C, \Delta, B, W, T, R, D$ have been displayed.*
11. *H records componentwise* hsame *transport for the source, partial-sum, coefficient, difference, boundary, transform, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

12. *K records displayed* Cont *replay of the route*

$$S, P, C, \Delta, B, W, T, R, D \longmapsto E.$$

13. *Q is the* Pkg *provenance over* $\mathsf{AbelTransformation}^{\uparrow}$, $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, P, C, \Delta, B, W, T, R, D, E, H, K, Q, N$. It has no coordinate for an infinite sum, selected limit, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8402.2 (Abel transformation NameCert obligations)**.** *For every accepted* $\mathsf{AbelTransformation}^{\uparrow}$ *carrier*

$$A = (S, P, C, \Delta, B, W, T, R, D, E, H, K, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through C and Δ; boundary-term exactness through B; transform-row exactness through W; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, K, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, P, C, \Delta, B, W, T, R, D, E, H, K, Q, N$ so every local naming obligation is attached to one displayed coordinate.

The analytic rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row C displays the paired coefficients, Δ displays adjacent differences, and B displays the endpoint products determined by the same window.

The transform row W is then forced by those rows. It records the finite Abel-transform boundary and interior difference-weighted partial-sum terms, so a convergence-test consumer can read T only after the transform data have been displayed.

Finally R, D, E, H, K, Q, N preserve and seal the same route. Because the carrier has no coordinate for infinite summation, selected limits, quotient series equality, countable choice, trichotomy, or ambient real completeness, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}$ surface. □

Theorem 8402.3 (Abel transformation summation handoff)**.** *Every accepted* $\mathsf{AbelTransformation}^{\uparrow}$ *packet hands a convergence-test consumer only the finite route*

$$\mathsf{SummationByParts}^{\uparrow} \longmapsto S, P, C, \Delta, B, W, T \longmapsto R, D, E, H, K, Q, N.$$

The downstream $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *and* $\mathsf{DirichletTest}^{\uparrow}$ *chapters may consume the displayed transform row, but they receive no infinite sum, selected limit, quotient series equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from this packet.*

Proof. Begin with the summation-by-parts dependency. The packet reads the same finite identity surface as chapter 5796, but specializes the visible handoff to the Abel-transform row W.

The Abel-facing consumers then read only T. An $\mathsf{AbelSummation}^{\uparrow}$ consumer uses the transform window as a summation boundary, an $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with its coefficient-control rows, and a $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and tail estimates. In each case the contribution of this chapter remains the displayed finite packet.

The regular and real-facing route is downstream of W and T. Rows R, D, E read back, account for, and seal the same finite transform, while H, K, Q, N transport, replay, source, and name that route. No excluded summation or completeness datum is present in the carrier. □

Falsifiable prediction: An accepted $\mathsf{AbelTransformation}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, partial-sum, coefficient, adjacent-difference, boundary, transform, and handoff rows refutes the carrier surface.

Independence witness: summation-by-parts, abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole fourteen-row packet $A = (S, P, C, \Delta, B, W, T, R, D, E, H, K, Q, N)$, because each sibling contributes only an identity surface, transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite Abel-transform window itself.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}$.

Constructive story: An $\mathsf{AbelTransformation}^{\uparrow}$ packet is generated from a SeriesUp finite source row, partial-sum windows, coefficient windows, adjacent-difference rows, boundary-term rows, a finite Abel-transform row, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8402.1, Definition 8402.2, and Definition 8402.3 seed the finite Abel-transform identity packet over $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelTransformation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite Abel-transform exactness, convergence-test handoff, regular readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.AbelTransformationUp.

Chapter 8403

A Concrete Naming Certificate for $\mathsf{BishopChainConnected}^{\uparrow}$

Following Bishop and Bridges on constructive connectedness of intervals, the $\mathsf{BishopChainConnected}^{\uparrow}$ packet records interval chain-connectedness as finite NameCert data over located interval rows, dyadic interval-net cells, finite chain ledgers, and Real-facing readbacks. It sits next to the metric chain-connected route of chapter 2050, consumes the finite interval-net surface of chapter 5966, and reads its Real-facing source rows through chapter 126, chapter 301, and chapter 38. The packet gives downstream interval and intermediate-value consumers a chain witness surface without invoking open-cover compactness, quotient equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8403.1 (Bishop chain-connected carrier)**.** *A* $\mathsf{BishopChainConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, A, Z, D, J, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval row, including the named endpoints, locatedness witnesses for endpoint comparisons, and the interval side on which the chain is read.*
2. *A is the endpoint-pair row whose members are to be joined inside the located interval surface.*
3. *Z is the finite chain ledger. It lists the inspected chain nodes, adjacency witnesses, and the finite successor route from the first endpoint row to the second endpoint row.*
4. *D is the dyadic tolerance and endpoint-arithmetic ledger used by the chain steps.*
5. *J is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row that supplies the finite dyadic interval-net cells through which the chain route is inspected.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for endpoint names, chain nodes, and cell endpoints.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those same windows and dyadic tolerances.*
8. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, A, Z, D, J, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the interval row, endpoint pair, chain ledger, dyadic data, interval-net cells, windows, readbacks, Real seal, and structural rows.*
10. *C records displayed* Cont *replay of the finite route*

$$I, A, Z, D, J \longmapsto W, R \longmapsto Q, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{BishopChainConnected}^{\uparrow}$, $\mathsf{ChainConnectedMetric}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopChainConnected}^{\uparrow}}$ *row.*

The carrier exports only this displayed finite packet. It has no coordinate for open-cover compactness, quotient equality of real names, a selected global path, countable choice, trichotomy, or ambient completeness.

Theorem 8403.2 (Bishop chain-connected NameCert obligations)**.** *For every accepted* $\mathsf{BishopChainConnected}^{\uparrow}$ *carrier* $B = (I, A, Z, D, J, W, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopChainConnected}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *interval admission: the located interval row I and endpoint-pair row A are displayed before any chain read;*

2. *finite chain ledger: the chain nodes, adjacency rows, and successor route are exhausted by the finite ledger Z;*

3. *dyadic cell support: every chain step reads the dyadic arithmetic row D and the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row J before it reaches a Real-facing boundary;*

4. *source readback: endpoint and cell observations pass through the common* $\mathsf{StreamName}^{\uparrow}$ *window W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q is consumed;*

5. *structural provenance: H, C, P, N transport, replay, source, and locally name only the displayed rows of the carrier.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BishopChainConnectedUp.BishopChainConnectedCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 8403.1. The located interval row, endpoint pair, finite chain ledger, dyadic arithmetic, interval-net cells, window, readback, Real seal, and structural rows are all coordinates of the same finite packet.

The chain obligation is finite because Z supplies the only allowed node and adjacency list. A consumer can replay a chain step only by reading a displayed successor entry of Z together with the dyadic support rows D and J.

The Real-facing obligation is downstream. The StreamName window W and RegSeqRat readback R are read before Q, and the structural rows H, C, P, N preserve that ordering componentwise. Since the carrier has no coordinate for open covers, quotient equality, choice, trichotomy, or completeness, none of those principles can be exported by the NameCert surface. □

Theorem 8403.3 (Bishop chain-connected interval handoff)**.** *Every interval-facing consumer of an accepted* $\mathsf{BishopChainConnected}^{\uparrow}$ *carrier must read the located interval row I, endpoint pair A, finite chain ledger Z, dyadic tolerance and endpoint-arithmetic row D,* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row J,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal Q in that displayed order. The handoff may then pass the packet to Real interval, compact interval, or intermediate-value consumers as a finite chain-connected source, but it may not replace the source by open-cover compactness, host real equality, quotient stream equality, selected paths, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopChainConnectedUp.BishopChainConnectedCarrier interval handoff`

Proof. Start with the obligation theorem Definition 8403.2. It forces every accepted interval read to expose the located interval and endpoint-pair rows before a chain witness is consumed.

Next read the finite chain and dyadic support rows. The ledger Z supplies the chain nodes and adjacency route, while D and the interval-net row J provide the dyadic endpoint arithmetic and finite cell incidence needed by each step.

Finally pass through the source readback rows W, R, Q and the structural rows H, C, P, N. This is the same Real-facing order used by the dyadic interval-net route of chapter 5966 and the metric chain-connected sibling chapter 2050. Thus downstream interval consumers receive a displayed finite chain-connected packet and no ambient compactness, equality, choice, or completeness endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopChainConnected}\uparrow}$.

Constructive story: A $\mathsf{BishopChainConnected}^{\uparrow}$ packet is generated from a located interval row, endpoint-pair row, finite chain ledger, dyadic endpoint arithmetic, DyadicIntervalNet cells, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopChainConnected}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8403.1, Definition 8403.2, and Definition 8403.3 seed the Bishop chain-connected interval route over located interval rows, finite chain ledgers, dyadic interval-net cells, StreamName windows, RegSeqRat readbacks, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopChainConnected}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BishopChainConnectedUp.BishopChainConnectedCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality of real names, selected global paths, countable choice, trichotomy, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite chain-ledger replay, dyadic interval-net support, StreamName and RegSeqRat readback routing, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopChainConnectedUp.

Chapter 8404

A Concrete Naming Certificate for $\mathsf{ConstructiveRiemannSum}^{\uparrow}$

Following Bishop and Bridges on constructive integration, the $\mathsf{ConstructiveRiemannSum}^{\uparrow}$ packet records a finite Riemann-sum surface over tagged dyadic partitions. It sits over the dyadic arithmetic core of chapter 291, the regular rational tolerance readback of chapter 301, the terminal real-name seal of chapter 38, and the pointwise source-map rows of chapter 130. The packet names the displayed finite sum only; it does not assert integrability, Darboux equivalence, mesh convergence, selected refining partitions, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8404.1 (Constructive Riemann sum carrier)**.** *A* $\mathsf{ConstructiveRiemannSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, T, F, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint row. It records a finite increasing list of dyadic endpoints and the adjacent half-open cells read by the partition surface.*
2. *T is the tag row. It records one displayed dyadic tag inside each cell named by E.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *readback row. It records the pointwise evaluations requested at the tags and the source-continuity provenance inspected before those values are used.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row. It records dyadic cell widths, dyadic tag comparisons, finite products of displayed values with displayed widths, and the finite dyadic sum replay.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row. It records the regular rational window through which the dyadic sum is read before a real-facing consumer accepts the value.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the endpoint, tag, map, arithmetic, and regular rational rows have been displayed.*
7. *H records componentwise* hsame *transport for endpoint, tag, map, dyadic arithmetic, regular tolerance, real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay for the route*

$$E, T, F, D \longmapsto R \longmapsto Q.$$

9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveRiemannSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host integrability, supremum or infimum data, quotient function equality, countable choice, selected mesh sequences, or ambient completeness.

Lean4 (definition) `BEDC.Derived.ConstructiveRiemannSumUp`

Theorem 8404.2 (Constructive Riemann sum NameCert obligations)**.** *For every accepted* ConstructiveRiemannSum *carrier $S = (E, T, F, D, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannSum}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed endpoint list, tag list,* $\mathsf{ContinuousMap}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic replay,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local naming; refinement stability for partitions obtained by inserting finitely many dyadic endpoints and retagging through displayed dyadic comparisons; sample transport along the H rows; dyadic finite-sum replay through D; terminal real handoff through R, Q; and ledger non-escape through P, N.*

Proof. Carrier admission is projection from Definition 8404.1. The tuple displays exactly the endpoint, tag, map, dyadic arithmetic, regular readback, real seal, transport, replay, provenance, and local naming rows, so every obligation begins with one of $E, T, F, D, R, Q, H, C, P, N$.

Refinement stability remains finite. Inserting dyadic endpoints only replaces the endpoint row by a longer displayed list and asks T for tags inside the new cells; the dyadic comparisons needed to validate those tags are already rows of D.

The sampled values are transported through H and replayed through C. Thus a consumer reads tag evaluations from F before using dyadic widths and products from D, and the finite sum has no source outside the displayed tagged partition.

The real-facing handoff is terminal. The dyadic sum passes through the regular rational tolerance row R before reaching the seal Q, and the provenance and naming rows P, N expose no hidden integrability theorem, selected mesh sequence, quotient equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRiemannSum}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveRiemannSum}^{\uparrow}$ packet is generated from finite dyadic endpoints, displayed tags, ContinuousMap point readback, DyadicRatCore width and product arithmetic, RegSeqRat tolerance readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveRiemannSum}^{\uparrow}}$, seed)

Scope closed: Definition 8404.1 and Definition 8404.2 seed the finite tagged dyadic partition surface and its dyadic-sum, regular-readback, and real-seal handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveRiemannSum}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.ConstructiveRiemannSumUp`

Bridge status: none

Not claimed: This chapter does not close Riemann integrability, Darboux equivalence, mesh-limit convergence, selected refining partitions, quotient equality of maps, ambient RealUp completeness, checked obligation rows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite refinement stability, sample transport, dyadic finite-sum replay, RegSeqRat tolerance readback, Real handoff, provenance, and local naming for BEDC.Derived.ConstructiveRiemannSu

Chapter 8405

A Concrete Naming Certificate for $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$

Fekete's subadditive lemma and Bishop–Bridges constructive sequence reasoning supply the traditional source for the $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$ packet: a displayed subadditive sequence is read through finite normalized-average windows before any lower-envelope or completion-facing consumer may inspect a Real boundary. The packet sits beside the subadditive-average carrier of chapter 2013, the limsup route of chapter 768, the regular rational readback of chapter 301, the finite stream window surface of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. It records the lemma as finite NameCert data, not as a host infimum principle, quotient stream equality, trichotomy, countable choice, or ambient completeness theorem.

Definition 8405.1 (Fekete subadditive lemma carrier)**.** *A* $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, A, Q, T, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row naming the inspected sequence windows.*
2. *A is the subadditivity ledger, listing each displayed decomposition comparison used by the lemma route.*
3. *Q is the normalized-average row, carrying the finite rational and dyadic denominator windows read from S and A.*
4. *T is the tail-window row, recording the cofinal finite tails over which the normalized averages are compared.*
5. *I is the tail-infimum readback row, storing only the lower-envelope data exposed by Q and T.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the tail-infimum readback is visible.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational handoff.*
8. *H records componentwise* hsame *transport for the source, subadditivity, average, tail, infimum-readback, handoff, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay through the route*

$$S, A \longmapsto Q \longmapsto T \longmapsto I \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$, $\mathsf{FeketeSubadditive}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FeketeSubadditiveLemma}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, Q, T, I, R, E, H, C, P, N$. It has no coordinate for a global infimum operator, quotient stream equality, selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8405.2 (Fekete subadditive lemma NameCert obligations)**.** *For every accepted* $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$ *carrier*

$$L = (S, A, Q, T, I, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FeketeSubadditiveLemma}^{\uparrow}}$ *surface consists exactly of source-window admission through S, subadditivity-ledger exposure through A, normalized-average comparison through Q, cofinal tail-window exposure through T, tail-infimum readback through I, regular rational handoff through R, Real seal exposure through E, componentwise transport through H, displayed replay through C, package provenance through P, local naming through N, and non-escape for global infima, selected limits, quotient streams, trichotomy, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 8405.1. The mathematical rows are exactly the source windows S, subadditivity ledger A, normalized-average row Q, tail-window row T, tail-infimum readback I, regular handoff R, and Real seal E.

The subadditivity content is read before the tail boundary. Rows S and A supply the inspected sequence values and decomposition comparisons, while Q records only the normalized averages computed from those displayed rows.

The tail readback is then forced through T and I. The row T supplies the cofinal finite windows, and I records the lower-envelope readback visible on those windows without adding a global infimum operator.

Finally R, E, H, C, P, N transport, replay, source, and name the same route. Since none of the displayed rows stores a selected limit, quotient stream equality, trichotomy, countable choice, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 8405.3 (Fekete subadditive lemma tail-infimum route)**.** *Every Real or completion-facing read of an accepted* $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$ *carrier factors through the finite route*

$$S, A \longmapsto Q \longmapsto T \longmapsto I \longmapsto R \longmapsto E.$$

The tail-infimum row I may be read only after the source sequence, subadditivity ledger, normalized-average comparisons, and cofinal tail windows have been displayed. Thus a downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer receives a finite lower-envelope route rather than a quotient sequence, selected limit, host infimum, trichotomy decision, countable-choice schedule, or ambient completeness witness.*

Proof. Start from the carrier projection. Rows $S, A, Q, T, I, R, E, H, C, P, N$ are the only public rows, and I is positioned downstream of the source, subadditivity, normalized-average, and tail-window data.

Apply the NameCert obligations of Definition 8405.2. They make S, A, Q, T visible before I, so the lower-envelope readback is computed from displayed finite comparisons rather than from a host infimum principle.

The regular rational and Real-facing rows are later in the same route. A $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer can read R and E only after the tail-infimum row has already been exposed.

Transport and replay preserve the route componentwise through H and C, while P and N fix provenance and naming. Therefore no consumer can bypass the displayed route with quotient streams, selected limits, trichotomy, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FeketeSubadditiveLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{FeketeSubadditiveLemma}^{\uparrow}$ packet is generated from source-sequence windows, a subadditivity ledger, normalized-average comparisons, cofinal tail windows, tail-infimum readback, regular rational handoff, Real seal exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FeketeSubadditiveLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8405.1, Definition 8405.2, and Definition 8405.3 seed the finite route from subadditive sequence windows to tail-infimum readback for Real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FeketeSubadditiveLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global infimum operator, quotient sequence equality, selected limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subadditivity-ledger exposure, normalized-average comparison, cofinal tail-window exposure, tail-infimum readback, regular rational handoff, Real seal exposure, transport, replay, provenance, and local naming for BEDC.Derived.FeketeSubadditiveLemmaUp.

Chapter 8406

A Concrete Naming Certificate for $\mathsf{CompactOpenEquicontinuity}^{\uparrow}$

Following Kelley and Bishop–Bridges on compact-open topology and equicontinuous families, the $\mathsf{CompactOpenEquicontinuity}^{\uparrow}$ packet records the finite compact-family certificate by which compact source windows, continuous-map graph rows, equicontinuity-family witnesses, and uniform-modulus handoffs are read together before an Arzela–Ascoli consumer may inspect a compact-open family. It sits over the compact metric source of chapter 129, the graph-and-modulus rows of chapter 130, the family-wide equicontinuity surface of chapter 413, the finite tolerance handoff of chapter 369, and the compact-family consumer of chapter 412. The packet names a compact-open equicontinuity boundary as displayed NameCert data; it is not a host compact-open topology, a selected subnet, quotient equality of functions, countable choice, or ambient completeness of a function space.

Definition 8406.1 (Compact-open equicontinuity carrier)**.** *A* $\mathsf{CompactOpenEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, A, U, O, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, listing the finite compact windows and coverage entries through which the compact-open inspection is performed.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family graph row, recording the finite graph evaluations visible on those compact windows.*
3. *A is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, carrying the family-wide local equicontinuity witnesses attached to the same compact windows.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained from the compact-window and family-equicontinuity data.*
5. *O is the compact-open observation ledger: a finite list of compact-window requests, tolerance requests, and graph-value windows that the consumer may inspect.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for graph values and tolerance reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those stream windows.*
8. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for value-facing and tolerance-facing reads.*

9. *H records componentwise* hsame *transport for compact-window, continuous-map, equicontinuity-family, uniform-modulus, observation, stream, readback, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$K, F, A, O \longmapsto U, S, R, Q, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CompactOpenEquicontinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenEquicontinuity}^{\uparrow}}$ *row.*

The carrier exports only these displayed rows. It has no coordinate for a host compact-open topology, function-space compactness theorem, selected convergent subnet, quotient equality of functions, countable choice, external modulus oracle, or ambient completeness.

Theorem 8406.2 (Compact-open equicontinuity NameCert obligations)**.** *For every accepted* CompactOpenEquicon *carrier*

$$E = (K, F, A, U, O, S, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOpenEquicontinuity}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *compact-window admission, requiring every compact-open request to expose the compact metric row K and the finite observation ledger O before any family read is consumed;*

2. *continuous-family exposure, requiring graph values to be read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *row F, the stream window S, the regular readback row R, and the terminal Real seal Q;*

3. *equicontinuity synchronization, requiring the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row A and the* $\mathsf{UniformModulus}^{\uparrow}$ *row U to share the compact-window and tolerance requests listed by O;*

4. *structural stability, requiring H, C, P, N to transport, replay, source, and locally name only the displayed rows of the carrier;*

5. *exclusion of host compact-open topology, selected subnets, quotient function equality, countable choice, external modulus choice, function-space compactness, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 8406.1. The compact-window data are exactly K and O, so any compact-open request begins with the displayed finite compact source and its inspection ledger.

The family values are then read through F, S, R, Q. This places every visible graph value behind the ContinuousMap row, StreamName window, RegSeqRat readback, and terminal Real seal, not behind a host function graph or quotient equality of maps.

The equicontinuity synchronization clause reads A and U over the same ledger O. Thus the family-wide equicontinuity witness and the downstream uniform-modulus handoff are tied to the compact windows already exposed by K.

Finally, H, C, P, N only transport, replay, source, and locally name the displayed rows. Since no row stores host compact-open topology, selected subnets, quotient function equality, countable choice, external modulus choice, function-space compactness, or ambient completeness, none of those objects can discharge a local NameCert obligation. □

Theorem 8406.3 (Compact-open equicontinuity Ascoli handoff)**.** *Every* ArzelaAscoli$^{\uparrow}$ *or compact-family consumer of an accepted* CompactOpenEquicontinuity$^{\uparrow}$ *carrier must read the compact metric row K, continuous-map family row F, equicontinuity-family row A, uniform-modulus handoff U, compact-open observation ledger O, stream window S, regular readback row R, terminal Real seal Q, and structural rows H, C, P, N in the displayed dependency order. The handoff may cite the packet as the compact-open equicontinuity boundary for the family, but it may not replace that boundary by host compact-open topology, a selected convergent subnet, quotient equality of functions, countable choice, a function-space compactness theorem, or ambient completeness.*

Proof. Start with the obligation theorem Definition 8406.2. It forces every compact-family read to expose K and O before the family graph or equicontinuity rows are consumed.

Next read the ContinuousMap and equicontinuity rows. The graph values pass through F, S, R, Q, while the family-wide equicontinuity data pass through A and the shared uniform-modulus handoff U.

The Ascoli-facing boundary is then the structural replay of the same finite route. Rows H, C, P, N preserve the order from compact windows to graph values, equicontinuity witnesses, and uniform-modulus output, so the consumer receives a displayed compact-open equicontinuity packet and no extra function-space compactness or choice-selected convergence object. □

Falsifiable prediction: Every accepted CompactOpenEquicontinuity$^{\uparrow}$ read exposes compact metric, continuous-map, equicontinuity-family, uniform-modulus, compact-open observation, StreamName, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Independence witness: compactmetric, continuousmap, equicontinuityfamily, uniformmodulus, arzelaascoli: CompactOpenEquicontinuity$^{\uparrow}$ is not identical to any listed sibling because it binds compact-open observation ledgers to family-wide equicontinuity and a uniform-modulus handoff before an Arzela–Ascoli consumer reads the family boundary.

Closure status for NameCert$_{\text{CompactOpenEquicontinuity}^{\uparrow}}$.

Constructive story: A CompactOpenEquicontinuity$^{\uparrow}$ packet is generated from CompactMetric finite-window rows, ContinuousMap graph rows, EquicontinuityFamily witnesses, a UniformModulus handoff, a compact-open observation ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactOpenEquicontinuity}^{\uparrow}}$, seed)

Scope closed: Definition 8406.1, Definition 8406.2, and Definition 8406.3 seed the compact-open equicontinuity route from compact windows and continuous family graphs to a UniformModulus and ArzelaAscoli-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{CompactOpenEquicontinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compact-open topology, selected convergent subnets, quotient equality of functions, countable choice, external modulus choice, function-space compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-family exposure, equicontinuity synchronization, compact-open observation exactness, UniformModulus handoff, RealUp seal ordering, replay, provenance, and local naming for BEDC.Derived.CompactOpenEquicontinuityUp.

Chapter 8407

A Concrete Naming Certificate for $\mathsf{FiniteOscillationCover}^{\uparrow}$

Following Bishop–Bridges compact metric analysis and the Heine–Cantor finite-cover route, the $\mathsf{FiniteOscillationCover}^{\uparrow}$ packet records the finite rows by which compact finite-net data and pointwise continuous-map control become an oscillation cover visible to uniform-continuity consumers. It sits over the compact metric source of chapter 129, the continuous-map surface of chapter 130, the dyadic tolerance core of chapter 291, the uniform-modulus consumer of chapter 369, and the terminal real-name seal of chapter 38. The packet names finite cover rows and finite oscillation bounds; it does not assert open-cover compactness, select points by choice, identify maps by host equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8407.1 (Finite oscillation cover carrier)**.** *A* $\mathsf{FiniteOscillationCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, D, C, R, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the displayed centers, radii, metric comparisons, and finite coverage ledger;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise continuity data are read at the listed centers and their displayed neighborhoods;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic oscillation bound and the finite comparisons used by the cover;*
4. *C is the finite oscillation-cover row, listing the accepted cover cells, their center witnesses, their source-side radius rows, and their cellwise oscillation bounds;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback seal for value comparisons when the oscillation bound is read in real-valued form;*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row visible after the finite cover and oscillation rows have been displayed;*
7. *H records componentwise* hsame *transport for compact, continuous-map, dyadic, finite-cover, real-readback, uniform-modulus, provenance, and naming rows;*
8. *T records displayed* Cont *replay for finite-net coverage, pointwise continuity reads, dyadic tolerance comparisons, and uniform-modulus handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteOscillationCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, D, C, R, U, H, T, P, N$. No coordinate stores a host open-cover theorem, a choice-selected subcover, a hidden global modulus, quotient function equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8407.2 (Finite oscillation cover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationCover}^{\uparrow}$ *carrier $O = (K, F, D, C, R, U, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationCover}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure through K, continuous-map pointwise reads through F, dyadic tolerance accounting through D, finite oscillation-cover cells through C, optional real-valued comparison sealing through R, uniform-modulus handoff through U, componentwise transport through H, displayed replay through T, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8407.1. The compact source, continuous-map row, dyadic tolerance, finite cover, real readback, uniform handoff, transport, replay, provenance, and local name are all displayed coordinates of one finite packet.

The compact and continuity rows are read before the cover row. The source row K supplies the finite centers and coverage ledger, while F supplies the pointwise continuity data attached to those displayed neighborhoods.

The dyadic and real rows then bound the oscillation read. The row D records the requested tolerance and finite comparisons, and R seals any real-valued comparison without adding an ambient completeness principle.

Finally C and U expose the finite oscillation cover and its downstream uniform-modulus handoff. The rows H, T, P, N preserve that route under transport, replay, provenance, and local naming, and no other obligation is available to the classifier. □

Theorem 8407.3 (Finite oscillation cover finite-subcover route)**.** *Every consumer that reads a* $\mathsf{FiniteOscillationCover}^{\uparrow}$ *packet as a compact-uniform-continuity input must factor through the displayed finite route*

$$K \longmapsto F \longmapsto D \longmapsto C \longmapsto R \longmapsto U \longmapsto H, T, P, N.$$

Thus the consumer receives a finite compact-metric coverage ledger, pointwise continuous-map rows, dyadic tolerance comparisons, finite oscillation cells, real-valued comparison seals when requested, and a $\mathsf{UniformModulus}^{\uparrow}$ *handoff, but it receives no open-cover compactness theorem, no choice-selected subcover, no hidden global modulus, no quotient map equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the compact source row K. It displays the finite-net coverage data used by the packet, so the consumer reads a listed finite coverage ledger rather than an external open-cover principle.

Read the continuous-map and dyadic rows next. The row F supplies pointwise control at the listed centers, and D supplies the dyadic tolerance comparisons that determine the accepted oscillation cells.

Project the cover row C. Its cells, center witnesses, radius rows, and oscillation bounds are the only finite subcover data available to the consumer; no selector outside the displayed list is present.

The real seal R and uniform handoff U are downstream of those finite rows. Transport, replay, provenance, and naming through H, T, P, N preserve exactly the same route, so the packet exports finite oscillation-cover data and refuses the excluded host principles. □

Falsifiable prediction: An accepted $\mathsf{FiniteOscillationCover}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying CompactMetric finite-net data, ContinuousMap pointwise rows, DyadicRatCore tolerance comparisons, finite oscillation-cover cells, optional RealUp comparison seals, transport, replay, provenance, and local naming.

Independence witness: compactmetric, continuousmap, dyadicratcore, real, uniformmodulus: none is bijective to $\mathsf{FiniteOscillationCover}^{\uparrow}$ because the packet binds compact finite-net coverage, pointwise continuity, dyadic tolerance accounting, finite oscillation cells, real-valued comparison seals, and a uniform-modulus handoff into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteOscillationCover}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise rows, DyadicRatCore tolerance comparisons, finite oscillation-cover cells, RealUp comparison seals, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8407.1, Definition 8407.2, and Definition 8407.3 seed the finite oscillation-cover packet for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, choice-selected subcovers, hidden global moduli, quotient function equality, ambient RealUp completeness, Lean verification, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked compact finite-net exposure, continuous-map pointwise reads, dyadic tolerance accounting, finite oscillation-cover rows, real comparison seals, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOscillationCoverUp.

Chapter 8408

A Concrete Naming Certificate for $\mathsf{OscillationDerivative}^{\uparrow}$

Following Lebesgue and Baire, the $\mathsf{OscillationDerivative}^{\uparrow}$ packet records the finite-window derivative step by which a real-valued function on a displayed metric source exposes its local oscillation boundary before any rank, Baire-one, or discontinuity consumer reads the data. It sits over the $\mathsf{Met}^{\uparrow}$ source of chapter 57, the $\mathsf{ContinuousMap}^{\uparrow}$ function-surface row of chapter 130, the $\mathsf{Real}^{\uparrow}$ value seal of chapter 38, the Baire-one oscillation neighbours chapter 8067 and chapter 8106, and the downstream oscillation-rank route of chapter 7961. The packet names only a displayed finite derivative window; it does not construct a transfinite rank, choose neighbourhood bases by countable choice, identify functions by quotient equality, or import ambient completeness.

Definition 8408.1 (OscillationDerivative carrier)**.** *An* $\mathsf{OscillationDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, X, W, O, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-surface row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *or an adjacent real-valued function packet, naming the displayed source-to-*$\mathsf{Real}^{\uparrow}$ *value observations.*
2. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the visible metric comparisons and named source points used by the derivative request.*
3. *W is the metric/Real finite-window row: it pairs a finite neighbourhood schedule around the inspected source points with the* $\mathsf{Real}^{\uparrow}$ *value windows read from F.*
4. *O is the oscillation-bound row, recording the displayed lower and upper value comparisons and the rational bound tested inside W.*
5. *R is the refinement ledger, recording which finite neighbourhood windows survive the derivative step and which are discarded at the displayed bound.*
6. *H records componentwise* hsame *transport for the function surface, metric source, finite windows, oscillation bounds, refinement ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an oscillation-derivative request through F, X, W, O, R and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{OscillationDerivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{OscillationDerivative}^{\uparrow}}$ *row.*
The carrier has no coordinate for a completed derivative hierarchy, a selected neighbourhood basis, quotient equality of functions, point-set measurability, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OscillationDerivativeUp.TasteGate.OscillationDerivativeTasteGate single carrier alignment`

Theorem 8408.2 (OscillationDerivative NameCert obligations)**.** *For every accepted* $\mathsf{OscillationDerivative}^{\uparrow}$ *carrier* $D = (F, X, W, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OscillationDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; function-surface admission through* F*; metric-source compatibility through* X*; finite neighbourhood and value-window admission through* W*; oscillation-bound exactness through* O*; refinement-ledger exactness through* R*; componentwise transport stability through* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Carrier admission is projection from Definition 8408.1. The accepted packet exposes exactly the rows $F, X, W, O, R, H, C, P, N$, so each local NameCert obligation must be read from one of these rows.

The analytic rows are read in finite order. The function-surface row F supplies the displayed real-valued observations, the metric row X supplies the source comparisons, and the finite-window row W binds neighbourhood inspection to the value windows already visible through F.

The derivative step is then the route through O and R. The oscillation-bound row records the tested lower and upper value comparisons at a rational bound, while the refinement ledger records only the displayed surviving and discarded windows.

Transport and replay preserve that route. The rows H and C cannot add a completed hierarchy, selected neighbourhood basis, quotient function equality, countable choice, point-set measurability, or ambient completeness principle, while P and N source and name only the listed packet. Hence the displayed rows exhaust the local certificate surface. □

Theorem 8408.3 (OscillationDerivative finite-window handoff)**.** *Every Baire-one, oscillation-rank, or discontinuity-boundary consumer of an accepted* $\mathsf{OscillationDerivative}^{\uparrow}$ *carrier factors through the finite route*

$$F, X, W, O, R \longmapsto H, C, P, N.$$

Thus the consumer may inspect the displayed function surface, metric source, finite neighbourhood schedule, Real value windows, oscillation-bound row, and refinement ledger, but it receives no completed rank hierarchy, no quotient function equality, no selected neighbourhood basis, no point-set measurability theorem, no countable-choice principle, and no ambient real-completion theorem.

Proof. Start with a consumer of the refinement row R. Carrier admission makes R available only after the function surface F, metric source X, finite window W, and oscillation-bound row O have been displayed.

Replay the continuation row C. The request opens the source and function-surface data, reads the named finite neighbourhood schedule, inspects the corresponding $\mathsf{Real}^{\uparrow}$ value windows, and only then exposes the oscillation-bound and refinement rows.

The handoff is therefore the stated finite route. Componentwise transport through H preserves the same rows, and P, N source and name only that packet. None of the excluded host objects can be recovered from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{OscillationDerivative}^{\uparrow}$ consumer that exports a derivative-window row must display the ContinuousMap or function-surface row, MetricSpace source row, finite neighbourhood schedule, RealUp value windows, oscillation-bound row, refinement ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before any Baire-one, oscillation-rank, or discontinuity-boundary handoff is available.

Independence witness: continuousmap, metricspace, real, baire-one-uniform-oscillation, baire-one-modulus, oscillation-rank: no listed sibling supplies the whole nine-row packet $D = (F, X, W, O, R, H, C, P, N)$, because each sibling contributes only the function surface, metric source, value seal, Baire-one oscillation window, modulus companion, or rank consumer while this chapter binds the derivative-window refinement ledger into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationDerivative}^{\uparrow}}$.

Constructive story: An $\mathsf{OscillationDerivative}^{\uparrow}$ packet is generated from a function-surface row, MetricSpace source row, finite neighbourhood and RealUp value windows, an oscillation-bound row, a refinement ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8408.1, Definition 8408.2, and Definition 8408.3 seed the finite-window oscillation-derivative certificate over displayed function-surface, metric-source, Real-value, oscillation-bound, refinement-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationDerivative}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OscillationDerivativeUp.TasteGate.OscillationDerivativeTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close completed derivative hierarchies, transfinite ranks, quotient function equality, selected neighbourhood bases, point-set measurability, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, axiom purity, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, function-surface admission, metric-source compatibility, finite-window admission, oscillation-bound exactness, refinement-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.OscillationDerivativeUp.

Chapter 8409

A Concrete Naming Certificate for $\mathsf{SardRegularValue}^{\uparrow}$

Following Sard and standard differential-topology treatments, the $\mathsf{SardRegularValue}^{\uparrow}$ packet records the finite regular-value consumer surface for displayed manifold chart rows, smooth-map rows, derivative finite-jet rows, the finite Sard jet packet of chapter 3644, terminal $\mathsf{Real}^{\uparrow}$ readback through chapter 38, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. It sits beside the charted manifold carrier of chapter 96 and consumes only finite window evidence; it does not claim full Sard measure theory, host smooth topology, a transversality theorem, or choice-selected regular values.

Definition 8409.1 (Sard regular-value carrier)**.** *A* $\mathsf{SardRegularValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, F, D, J, V, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row naming the source and target chart windows in which the candidate value is inspected.*
2. *F is the smooth-map observation row, containing only the displayed map values and chart restrictions consumed by the packet.*
3. *D is the derivative finite-jet row, recording the derivative windows and rank reads used to test regularity.*
4. *J is the* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *support row that supplies the finite critical-window exclusion surface.*
5. *V is the candidate value-window row whose regular-value status is being consumed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for the displayed value after the finite chart and jet route has been replayed.*
7. *H records componentwise* hsame *transport for the chart, map, derivative, Sard-jet, value-window, readback, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a regular-value request through $M, F, D, J, V, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SardRegularValue}^{\uparrow}$, $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SardRegularValue}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for ambient measure-zero data, host smooth-function spaces, selected values, global transversality, or a standard bridge.

Theorem 8409.2 (Sard regular-value NameCert obligations)**.** *For every accepted* $\mathsf{SardRegularValue}^{\uparrow}$ *carrier* $R = (M, F, D, J, V, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SardRegularValue}^{\uparrow}}$ *surface consists exactly of chart admission through* M*, smooth-map observation through* F*, derivative finite-jet admission through* D*, finite critical-window support through* J*, candidate value exposure through* V*, terminal* $\mathsf{Real}^{\uparrow}$ *readback through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8409.1. Its rows $M, F, D, J, V, E, H, C, P, N$ are the whole accepted packet, so every regular-value read begins as finite BHist data.

The differential-topology part is confined to M, F, D, J, V. The chart row and smooth-map row expose the finite windows, the derivative row records the rank reads, the Sard finite-jet row supplies the critical-window support, and the value row names the candidate window.

The readback and structure rows preserve that same route. The row E seals the displayed value as $\mathsf{Real}^{\uparrow}$, while H, C, P, N transport, replay, source, and name the packet without adding host measure theory, global transversality, selected regular values, or a smooth-function-space quotient. □

Theorem 8409.3 (Sard regular-value critical-window handoff)**.** *Every consumer of an accepted* $\mathsf{SardRegularValue}^{\uparrow}$ *packet that asks whether a displayed value window is regular must first read the route*

$$M \longmapsto F \longmapsto D \longmapsto J \longmapsto V \longmapsto E$$

with structural support H, C, P, N*. The row* J *is the only handoff to* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$*, so the consumer receives finite critical-window exclusion data rather than a full Sard measure theorem, host topology on smooth maps, or a choice-selected regular value.*

Proof. Begin with the chart and map rows M, F. They name the displayed source and target windows and the finite observations available to the regular-value consumer.

Next read D, J, V. The derivative row supplies the rank-facing finite-jet data, the Sard finite-jet row records the critical-window support from chapter 3644, and the value row identifies the inspected target window.

Finally replay the terminal value through E and the structural rows. The route stays inside the displayed carrier and therefore cannot export ambient measure-zero reasoning, host smooth topology, global transversality, or a selected regular value. □

Falsifiable prediction: Every accepted $\mathsf{SardRegularValue}^{\uparrow}$ consumer must expose $\mathsf{Manifold}^{\uparrow}$, smooth-map finite observations, derivative finite-jet rows, $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ readback, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before any regular-value result is consumed.

Independence witness: sard-theorem-finite-jet, manifold, real: $\mathsf{SardRegularValue}^{\uparrow}$ is not bijective to any listed sibling because it binds a candidate value-window consumer to chart, map, derivative, finite critical-window support, terminal real readback, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SardRegularValue}^{\uparrow}}$.

Constructive story: A SardRegularValue$^{\uparrow}$ packet is generated from Manifold chart rows, smooth-map observation rows, derivative finite-jet rows, a SardTheoremFiniteJetUp support row, candidate value-window rows, a terminal RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SardRegularValue}\uparrow}$, seed)

Scope closed: Definition 8409.1, Definition 8409.2, and Definition 8409.3 seed the finite regular-value certificate over chart, map, derivative, Sard finite-jet, value-window, readback, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SardRegularValue}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.SardRegularValueUp`

Bridge status: none

Not claimed: This chapter does not claim the full Sard measure theorem, transversality, choice-selected regular values, host smooth topology, Lean theorem checking, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative finite-jet admission, Sard finite-jet handoff, candidate value-window exposure, terminal Real readback, transport stability, replay, provenance, and local naming for the displayed packet.

Chapter 8410

A Concrete Naming Certificate for $\mathsf{SardTheorem}^{\uparrow}$

Following Sard's theorem in differential topology, the $\mathsf{SardTheorem}^{\uparrow}$ packet names the theorem-level consumer surface that reads the finite critical-window certificate of chapter 3644 together with the regular-value consumer route of chapter 8409. It sits over the charted manifold carrier of chapter 96 and the terminal Real seal of chapter 38. The packet records the finite BEDC route by which critical-value exclusion and regular-value readback are consumed together; it does not claim the classical measure-zero Sard theorem, a transversality theorem, a host smooth-topology argument, or choice-selected regular values.

Definition 8410.1 (Sard theorem carrier)**.** *A* $\mathsf{SardTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, J, R, V, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row that names the source and target chart windows read by the theorem-level consumer.*
2. *J is the* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *support row carrying the finite derivative-jet, rank-defect, and critical-value window evidence.*
3. *R is the* $\mathsf{SardRegularValue}^{\uparrow}$ *row that consumes the finite critical-window support as a regular-value read.*
4. *V is the displayed value-window row whose critical or regular status is inspected.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row used to expose the finite value-window comparison.*
6. *H records componentwise* hsame *transport for M, J, R, V, E and the structural rows below.*
7. *C records displayed* Cont *replay through the chart, finite-jet, regular-value, value-window, and Real rows.*
8. *P is the* Pkg *provenance over* $\mathsf{SardTheorem}^{\uparrow}$, $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{SardRegularValue}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SardTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Lebesgue measure, transversality, a host topology on smooth maps, a selected regular value, or an ambient differentiability theorem.

Theorem 8410.2 (Sard theorem NameCert obligations)**.** *For every accepted* $\mathsf{SardTheorem}^{\uparrow}$ *carrier* $S = (M, J, R, V, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SardTheorem}^{\uparrow}}$ *surface consists exactly of chart admission through* M*, finite-jet critical-window support through* J*, regular-value consumption through* R*, displayed value-window exposure through* V*, terminal* $\mathsf{Real}^{\uparrow}$ *readback through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier $S = (M, J, R, V, E, H, C, P, N)$. The chart row M fixes the finite manifold windows available to the theorem-level read, while the finite-jet row J supplies the derivative, rank-defect, and critical-window evidence.

Next read the regular-value row R together with the inspected value window V. The regular-value route of chapter 8409 consumes the finite critical-window support of chapter 3644, so the theorem-level packet uses exactly the displayed handoff rather than an ambient measure argument.

Finally read E, H, C, P, N. The Real row seals the finite value-window comparison, and the structural rows transport, replay, source, and name the same packet. Since the excluded host coordinates are absent from the carrier, they cannot discharge any local NameCert obligation. □

Theorem 8410.3 (Sard theorem critical-value route)**.** *Every consumer of an accepted* $\mathsf{SardTheorem}^{\uparrow}$ *packet that asks for a theorem-level critical-value or regular-value read must first traverse the finite route*

$$M \longmapsto J \longmapsto R \longmapsto V \longmapsto E$$

with structural support H, C, P, N*. The consumer receives the finite* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *and* $\mathsf{SardRegularValue}^{\uparrow}$ *rows over* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, not a full measure-theoretic Sard theorem, transversality theorem, host smooth-topology surface, or choice-selected regular value.*

Proof. Begin with the charted manifold row M. It names the source and target windows before any critical-value request can be inspected.

Apply the finite-jet packet of chapter 3644. The row J exposes derivative-jet, rank-defect, value-window, and local regularity data as finite certificate rows.

Use the regular-value consumer of chapter 8409. The row R can read regularity only through the finite critical-window support supplied by J, and the inspected value window V reaches the terminal Real readback E through the same displayed route.

The structural rows H, C, P, N preserve the packet under transport, replay, provenance, and local naming. No omitted host coordinate can supply measure-zero reasoning, transversality, smooth-topology closure, or a selected regular value. □

Falsifiable prediction: Every accepted $\mathsf{SardTheorem}^{\uparrow}$ consumer must expose $\mathsf{Manifold}^{\uparrow}$, $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{SardRegularValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before any theorem-level critical-value or regular-value read is public.

Independence witness: sard-theorem-finite-jet, sard-regular-value, manifold, real: $\mathsf{SardTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds theorem-level chart admission, finite critical-window support, regular-value consumption, value-window readback, terminal Real sealing, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SardTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SardTheorem}^{\uparrow}$ packet is generated from Manifold chart rows, a SardTheoremFiniteJetUp finite critical-window support row, a SardRegularValueUp consumer row, a displayed value-window row, a terminal RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SardTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 8410.1, Definition 8410.2, and Definition 8410.3 seed the theorem-level finite route through manifold chart rows, Sard finite-jet support, regular-value consumption, value-window exposure, Real readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{SardTheorem}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.SardTheoremUp`

Bridge status: none

Not claimed: This chapter does not claim full measure-theoretic Sard, transversality, host smooth topology, choice-selected regular values, Lean theorem checking, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-jet support, regular-value consumption, value-window exposure, Real readback, transport stability, replay, provenance, and local naming for the displayed packet.

Chapter 8411

A Concrete Naming Certificate for $\mathsf{MetricDerivedSet}^{\uparrow}$

Following Kuratowski and Bourbaki on derived sets in metric spaces, the $\mathsf{MetricDerivedSet}^{\uparrow}$ packet records the finite BEDC route by which a displayed subset exposes points whose every punctured metric neighbourhood meets that subset. It sits beside the metric carrier of chapter 57, the closure interface of chapter 3342, the metric sequential limit-point route of chapter 8015, the finite sequence windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal real seal of chapter 38. The packet is a conservative L10 neighbour of closure and sequential-limit evidence; it does not choose cluster points, identify quotient streams, import open-cover compactness, use countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8411.1 (Metric derived-set carrier)**.** *A* $\mathsf{MetricDerivedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, S, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance-observation rows, and finite metric-law ledger used by the neighbourhood tests.*
2. *S is the subset row whose accumulation boundary is being inspected.*
3. *Q is the punctured-ball witness ledger: each displayed radius request records a candidate point, a positive puncture condition, a subset witness row, and a metric-distance read showing the witness lies in the requested punctured ball.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface used to read candidate names, subset witnesses, and inspected radius windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the metric-distance and radius comparisons carried by Q and W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the metric, subset, punctured-ball, window, and readback rows have been displayed.*
7. *H records componentwise* hsame *transports for M, S, Q, W, R, E and for the provenance and local naming rows.*
8. *C records displayed* Cont *replay from metric and subset reads through punctured-ball witnesses, finite windows, regular rational readback, and the terminal Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricDerivedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricDerivedSet}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for an open-cover compactness theorem, a selected cluster point, quotient equality of sequences or subsets, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8411.2 (Metric derived-set NameCert obligations)**.** *For every accepted* $\mathsf{MetricDerivedSet}^{\uparrow}$ *carrier $D = (M, S, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricDerivedSet}^{\uparrow}}$ *surface consists exactly of metric-source admission through M, subset admission through S, punctured-ball witness coverage through Q, finite StreamName window visibility through W, RegSeqRat readback through R, Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The metric row, subset row, punctured-ball witness ledger, finite windows, regular rational readbacks, and terminal Real seal are the only mathematical rows exposed to a consumer.

The punctured-ball clause is finite because Q records the inspected radius requests and their displayed subset witnesses. Each witness is read through W and each distance comparison is read through R before the Real seal E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no compactness, selector, quotient, choice, or completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8411.3 (Metric derived-set L10 handoff)**.** *Every closure-facing or sequential-limit-point consumer of an accepted* $\mathsf{MetricDerivedSet}^{\uparrow}$ *packet must first read the finite route through* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *In carrier terms, the rows $M, S, Q, W, R, E, H, C, P, N$ must remain visible before the consumer treats the packet as derived-set evidence for a displayed subset. The handoff exports punctured-ball witness data and finite readback seals, not open-cover compactness, selected cluster points, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8411.2. They place the metric row, subset row, punctured-ball witness ledger, finite windows, rational readbacks, Real seal, transport, replay, provenance, and local naming in one displayed packet.

For a closure-facing read, the packet first exposes the same metric and subset rows used by $\mathsf{MetricClosure}^{\uparrow}$, then replaces a bare candidate-limit read with the punctured-ball ledger Q. The finite windows W, regular rational readback R, and Real seal E keep the closure evidence inside displayed BEDC rows.

For a sequential-limit-point read, the packet exposes the same finite observation discipline used by $\mathsf{MetricSequentialLimitPoint}^{\uparrow}$: sequence or witness windows are read through $\mathsf{StreamName}^{\uparrow}$, distance comparisons through $\mathsf{RegSeqRat}^{\uparrow}$, and terminal comparisons through $\mathsf{Real}^{\uparrow}$ before the derived-set consumer may inspect the punctured-ball witness.

The refused principles require rows absent from D. No coordinate supplies an open-cover compactness theorem, a selected cluster point, quotient equality, countable choice, or ambient Real completeness, so the L10 handoff is exactly the finite route stated above. □

Falsifiable prediction: Every accepted $\mathsf{MetricDerivedSet}^{\uparrow}$ read exposes one Metric row, one subset row, one punctured-ball witness ledger, finite StreamName windows, RegSeqRat readbacks,

one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any derived-set consumer is public.

Independence witness: metric, metric-closure, metric-sequential-limit-point, streamname, regseqrat, real: $\mathsf{MetricDerivedSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a subset row and punctured-ball witness ledger over the metric, window, readback, seal, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricDerivedSet}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricDerivedSet}^{\uparrow}$ packet is generated from a Metric row, a subset row, a punctured-ball witness ledger, finite StreamName windows, RegSeqRat readbacks, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricDerivedSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8411.1, Definition 8411.2, and Definition 8411.3 seed the finite metric derived-set route between metric closure and metric sequential limit-point consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricDerivedSet}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MetricDerivedSetUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected cluster points, quotient equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, punctured-ball witness coverage, finite-window visibility, regular rational readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetricDerivedSetU

Lean4 (statement) `BEDC.Derived.MetricDerivedSetUp`

Chapter 8412

A Concrete Naming Certificate for $\mathsf{ApartnessCut}^{\uparrow}$

Following Bishop and Bridges, an apartness cut is a cut-shaped real observation whose positive separation information is displayed before any real-order or completion consumer may inspect it. The $\mathsf{ApartnessCut}^{\uparrow}$ packet sits beside the located-cut route of chapter 404, the located real-number route of chapter 6640, the dyadic tolerance ledger of chapter 291, the stream-window source of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It records a constructive apartness-facing cut as finite BEDC data, not as trichotomy, quotient-cut equality, selected comparisons, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8412.1 (Apartness cut carrier)**.** *An* $\mathsf{ApartnessCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, U, K, S, R, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower cut-observation row, containing only finite rational lower observations and their downward-use discipline.*
2. *U is the upper cut-observation row, containing only finite rational upper observations and their upward-use discipline.*
3. *K is the located-cut row, read through* $\mathsf{LocatedCut}^{\uparrow}$ *and carrying displayed gap witnesses between lower and upper observations.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule for the inspected cut windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row exposed after S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for radii, endpoint gaps, and dyadic comparison slack.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, S, R, D have been displayed.*
8. *Q is the apartness-facing query row: it records positive separation requests and their displayed rational witnesses, but it does not decide equality or trichotomy.*
9. *H records componentwise* hsame *transport for the lower, upper, located-cut, stream, regular-rational, dyadic, real-seal, apartness-query, replay, provenance, and naming rows.*
10. *C records displayed* Cont *replay from an apartness query through $L, U, K, S, R, D, E, Q, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{ApartnessCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedRealNumber}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, S, R, D, E, Q, H, C, P, N$. It has no coordinate for cut quotients, real trichotomy, a selected apartness oracle, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ApartnessCutUp.ApartnessCutTasteGate single carrier alignment`

Theorem 8412.2 (Apartness cut NameCert obligations)**.** *For every accepted* $\mathsf{ApartnessCut}^{\uparrow}$ *carrier $A = (L, U, K, S, R, D, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ApartnessCut}^{\uparrow}}$ *surface consists exactly of carrier admission, lower- and upper-cut observation exposure, located-cut gap visibility, stream-window scheduling, regular-rational readback, dyadic tolerance exactness, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, apartness-query exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8412.1. The rows $L, U, K, S, R, D, E, Q, H, C, P, N$ are the complete local inventory available to an accepted packet.

The cut-side rows are finite. Lower and upper observations meet only through the located-cut row K, so a consumer cannot replace the displayed gap witnesses by a quotient cut or by an ambient order object.

The real-facing route then passes through the stream schedule S, the regular rational readback R, the dyadic tolerance ledger D, and only then the terminal seal E. The apartness-query row Q reads positive separation evidence along that same route; it does not add equality, trichotomy, or a selected comparison oracle.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded object is a coordinate of the carrier, the listed rows exhaust the local certificate obligations. □

Theorem 8412.3 (Apartness cut L10 handoff)**.** *Every Phase Real or Completion consumer of an accepted* $\mathsf{ApartnessCut}^{\uparrow}$ *packet must read apartness-cut evidence through the displayed route*

$$\mathsf{LocatedCut}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the lower and upper cut rows, apartness-query row Q, and the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support retained. The handoff exports a constructive apartness-cut source for* $\mathsf{Real}^{\uparrow}$ *apartness and order work, not trichotomy, quotient-cut equality, countable choice, selected limits, or ambient real completeness.*

Proof. Start from the NameCert obligations of Definition 8412.2. They force the lower row, upper row, located gap row, stream schedule, regular readback, dyadic tolerance ledger, Real seal, and apartness-query row to remain visible before any public read is available.

Use the located-cut sibling chapter 404 and the located real-number sibling chapter 6640 as the readback discipline for finite cut and located real observations. The new packet keeps that discipline while making the positive separation query Q an explicit coordinate.

The L10 route consumes only the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows after the located-cut witness has been shown. Structural rows H, C, P, N preserve the route under transport, replay, provenance, and local naming.

Thus the handoff supplies the constructive apartness-facing boundary needed by Real and Completion consumers. Because none of trichotomy, quotient-cut equality, countable choice, selected limits, or ambient completeness is present as a carrier row, none is exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{ApartnessCut}^{\uparrow}$ read exposes lower and upper cut observations, a LocatedCut gap witness, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a terminal Real seal, an apartness-query row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any Real-facing apartness or order consumer may inspect it.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedcut, located-real-number: no listed sibling is bijective with $\mathsf{ApartnessCut}^{\uparrow}$ because this packet jointly binds lower and upper cut observations, located gap witnesses, stream windows, regular rational readback, dyadic tolerance data, a terminal Real seal, apartness-query evidence, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessCut}^{\uparrow}}$.

Constructive story: An $\mathsf{ApartnessCut}^{\uparrow}$ packet is generated from lower and upper cut observations, a LocatedCut gap row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a terminal Real seal, an apartness-query row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8412.1, Definition 8412.2, and Definition 8412.3 seed the finite constructive apartness-cut route over DyadicRatCore, StreamName, RegSeqRat, Real, LocatedCut, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessCut}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ApartnessCutUp.ApartnessCutTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient-cut equality, selected apartness or comparison oracles, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cut gap visibility, stream-window scheduling, regular-rational readback, dyadic tolerance exactness, Real seal nonescape, apartness-query exposure, transport, replay, provenance, and local naming for BEDC.Derived.ApartnessCutUp.

Chapter 8413

A Concrete Naming Certificate for $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$

Following Bishop and Bridges on explicit real-name bounds and finite approximation windows, the $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ packet records the finite Archimedean witness window that a Real and Completion consumer must read before a terminal real-name seal is exposed. It sits over the finite stream surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the sequential Archimedean handoff of chapter 8278. The packet names a bounded witness-window route; it is not trichotomy, quotient stream equality, a selected completed representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8413.1 (Sequential Archimedean window carrier)**.** *A* $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, i, Q, D, A, G, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name whose bounded sequential observation is being prepared.*
2. *i is the unary* $\mathsf{Nat}^{\uparrow}$ *request row naming the observation scale.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *Archimedean witness row visible at the request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row presenting the same witness for window consumers.*
5. *A is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened by the request and witness rows.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback extracted from the same window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after i, Q, D, A, G have been displayed.*
8. *H records componentwise* hsame *transport for the real name, request, witness rows, finite window, regular readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the request through the finite window, regular readback, witness rows, and terminal seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$, $\mathsf{SequentialArchimedeanReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanWindow}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $R, i, Q, D, A, G, E, H, C, P, N$. It has no coordinate for trichotomy, quotient stream equality, a selected completed representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialArchimedeanWindowUp.SequentialArchimedeanWindowTasteGate single carrier alignment`

Theorem 8413.2 (Sequential Archimedean window NameCert obligations)**.** *For every accepted* $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ *carrier*

$$W = (R, i, Q, D, A, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real name, unary request, rational witness, dyadic tolerance, finite stream window, regular readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows; request-window compatibility binding i, Q, D to the same window A that produces G; witness coherence binding Q and D to one finite Archimedean bound; seal ordering requiring E to be read only after i, Q, D, A, G; and non-escape excluding trichotomy, quotient stream equality, selected completed representatives, countable choice, ambient Real completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 8413.1. The rows $R, i, Q, D, A, G, E, H, C, P, N$ are the complete packet inventory, so carrier admission and local naming introduce no hidden object.

The request-window compatibility clause follows from the displayed order. The unary request i and witness rows Q, D open the finite window A, and the readback G is extracted from that same window before the seal E can be consumed.

Witness coherence is componentwise. The rational witness Q and dyadic tolerance row D describe one Archimedean bound for the same request, and H and C transport and replay that bound without allocating another witness row.

Finally the seal-ordering and non-escape clauses use the absence of further coordinates. Since E is the only terminal $\mathsf{Real}^{\uparrow}$ seal and the carrier has no row for trichotomy, quotient stream equality, selected completed representatives, countable choice, ambient completeness, Lean verification, or bridge checking, every accepted consumer remains inside the displayed finite route. □

Theorem 8413.3 (Sequential Archimedean window Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ *carrier*

$$W = (R, i, Q, D, A, G, E, H, C, P, N)$$

must read the unary request i, the rational witness Q, the dyadic tolerance row D, the finite window A, and the regular readback G before it may read the terminal seal E. The handoff is therefore a finite Archimedean witness-window route from $\mathsf{SequentialArchimedeanReal}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$ *consumers, not a completeness principle or a selected representative of a completed stream.*

Proof. Read the accepted carrier. The only Real-facing seal row is E, and it is stored after the request, witness, window, and readback rows i, Q, D, A, G.

Apply Definition 8413.2. Its seal-ordering clause requires the request-window compatibility and witness-coherence rows before any consumer-visible seal read.

The provenance row P names the sibling route through $\mathsf{SequentialArchimedeanReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while H and C preserve the same displayed rows under

transport and replay. Hence the Real handoff exports exactly the finite witness window and not trichotomy, quotient stream equality, selected completed representatives, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ Real-facing read exposes a RealUp name, a NatUp unary request, RatUp and DyadicRatCoreUp Archimedean witness rows, a StreamNameUp finite window, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before the terminal seal is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real, sequential-archimedean-real: $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a unary request, rational and dyadic Archimedean witness rows, finite stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming while refusing trichotomy, quotient stream equality, selected completed representatives, countable choice, and Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialArchimedeanWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialArchimedeanWindow}^{\uparrow}$ packet is generated from a RealUp name, a NatUp unary request, RatUp and DyadicRatCoreUp Archimedean witness rows, a StreamNameUp finite window, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialArchimedeanWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8413.1, Definition 8413.2, and Definition 8413.3 seed the finite Archimedean witness-window route before RealUp consumers read a terminal seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialArchimedeanWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient stream equality, selected completed representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-window compatibility, rational-dyadic witness coherence, Real-seal ordering, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.SequentialArchimedeanW

Chapter 8414

A Concrete Naming Certificate for $\mathsf{EntourageCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive completion discipline, Kelley's uniform-space entourages, and Bourbaki's filter-facing uniformity language, the $\mathsf{EntourageCompletion}^{\uparrow}$ packet names the finite route by which an entourage base can be read as completion data without quotienting Cauchy filters or selecting limits. It sits between the Cauchy-filter surface of chapter 333, the finite stream windows of chapter 126, the regular rational sequence route of chapter 301, the dyadic ledger of chapter 291, and the terminal Real seal of chapter 38.

Definition 8414.1 (Entourage completion carrier)**.** *An* $\mathsf{EntourageCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, F, B, S, R, D, W, H, C, P, N)$$

with the following rows:

1. *U is a displayed uniformity row whose entourages are read as finite neighborhood relations rather than as a completed quotient space.*
2. *F is a* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose Cauchy condition is opened only through named entourage requests.*
3. *B is an entourage-base row, assigning each accepted request a finite base entourage and its monotone refinement ledger.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row used to expose observable names before a completion-facing consumer reads the route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row and D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by Real-facing consumers.*
6. *W is a common-window row tying the Cauchy-filter read and the stream read to the same displayed entourage budget.*
7. *H records componentwise* hsame *transport for the uniformity, filter, entourage-base, stream, regularity, dyadic, window, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an entourage request through B, W, S, R, D and then toward the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *P is the* Pkg *provenance over* $\mathsf{EntourageCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{EntourageCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for quotient filters, host uniform spaces, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8414.2 (Entourage completion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EntourageCompletion}^{\uparrow}}$ *surface for an accepted carrier* $E = (U, F, B, S, R, D, W, H, C, P, N)$ *consists exactly of carrier admission, entourage-base exposure, Cauchy-filter source exposure, finite StreamName observation, RegSeqRat readback, DyadicRatCore tolerance control, common-window routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert clause is stored outside the displayed rows* $U, F, B, S, R, D, W, H, C, P, N$.

Proof. Project the carrier of Definition 8414.1. Its listed coordinates are the whole local packet, so carrier admission and local naming are read directly from the displayed tuple.

The entourage-base row B opens the uniform request before the filter source row F is consumed. The common-window row W then binds that filter read to the same finite stream, regularity, and dyadic tolerance route used by S, R, D.

The structural rows H, C, P, N preserve and name this route without adding a new object. Since quotient filters, selected limits, countable choice, host uniform equality, and ambient Real completeness are absent from the carrier, they cannot occur as local NameCert obligations. □

Theorem 8414.3 (Entourage completion Cauchy-filter handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{EntourageCompletion}^{\uparrow}$ *packet must first expose the displayed uniformity row, entourage-base row, common window, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The handoff exports Cauchy-filter data only as the finite route*

$$U, F, B, W, S, R, D, H, C, P, N \longmapsto \mathsf{CauchyFilter}^{\uparrow}.$$

It exports no quotient filter, selected limit, cofinal-choice schedule, host uniform equality, or detached completion point.

Proof. Start from the source row F in the carrier. The filter is not inspected by an ambient limit operation; it is opened only after the uniformity row U and entourage-base row B have named the finite request.

Next bind the read to the common window. The row W routes the same request through the finite observation row S, the regular readback row R, and the dyadic tolerance ledger D before any completion-facing consumer can reuse the filter data.

Finally replay the route through H, C, P, N. These rows transport, continue, source, and name the displayed packet, and no excluded quotient, selected limit, choice schedule, host uniform equality, or detached completion point is available in the carrier. □

Theorem 8414.4 (Entourage completion Real-route nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{EntourageCompletion}^{\uparrow}$ *packet passes through the same displayed route*

$$\mathsf{CauchyFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The terminal Real seal is reached only after the entourage request, Cauchy filter source, finite stream window, regular readback, dyadic tolerance ledger, and common-window rows have been displayed. The read exports no quotient filters, selected limits, countable choice, host real equality, host uniform space equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the Cauchy-filter handoff of Definition 8414.3. It forces the entourage and Cauchy-filter source rows to remain visible before the completion-facing route is consumed.

Then read the L10 rows. StreamName observations, RegSeqRat regularity, and the DyadicRatCore tolerance ledger form the finite bridge from the displayed entourage request to the terminal Real seal.

The structural rows keep the bridge inside the packet. Componentwise hsame transport and displayed Cont replay preserve the same route, while Pkg and NameCert record its provenance and local name. Since the carrier contains none of the excluded host-level or choice-level data, the Real-facing read cannot escape the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{EntourageCompletion}^{\uparrow}$ read factors through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows before any completion-facing consumer can inspect a terminal seal.

Closure status for $\mathsf{NameCert}_{\mathsf{EntourageCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EntourageCompletion}^{\uparrow}$ packet is generated from a displayed uniformity row, CauchyFilter source row, entourage-base row, StreamName finite observations, RegSeqRat readback, DyadicRatCore tolerance ledger, a common-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EntourageCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8414.1, Definition 8414.2, Definition 8414.3, and Definition 8414.4 seed the constructive entourage-completion route over displayed CauchyFilterUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EntourageCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected limits, countable choice, host uniform-space equality, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, entourage-base exposure, Cauchy-filter handoff, common-window stability, Real-route nonescape, and TasteGate field faithfulness for BEDC.Derived.EntourageCompletionUp.

Chapter 8415

A Concrete Naming Certificate for $\mathsf{BoundedRealInfimum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BoundedRealInfimum}^{\uparrow}$ packet records the lower-cut side of a bounded real sequence as finite BEDC naming data. It sits beside the bounded sequence surface of chapter 1415, the bounded lower-cut observation discipline of chapter 1440, the dyadic rational ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet gives L10 Real-and-Completion consumers a reusable lower-bound certificate without importing a host infimum, quotient stream equality, choice of subsequence, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8415.1 (Bounded real infimum carrier)**.** *A* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, L, D, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row whose finite windows, bound ledger, and displayed* $\mathsf{Real}^{\uparrow}$ *entries are inspected.*
2. *L is the* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *lower-cut schedule, recording finite lower observations for the bounded source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare lower-cut observations with displayed rational budgets.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows consumed by the lower-cut route.*
5. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal naming the candidate lower-bound value only after S, L, D, Q have been replayed.*
6. *H records componentwise* hsame *transport for the bounded source, lower-cut schedule, dyadic ledger, rational readback, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *routes by which consumers replay the bounded sequence row, lower-cut schedule, dyadic ledger, regular readback, terminal real seal, transports, provenance, and local naming row.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealInfimum}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRealInfimum}^{\uparrow}}$ *row.*
The carrier stores a finite lower-bound route for bounded real sequences. It has no coordinate for a host infimum, quotient equality of streams, a selected subsequence, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Definition 8415.2 (Bounded real infimum classifier)**.** *Two* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *carriers*

$$I = (S, L, D, Q, T, H, C, P, N) \qquad \text{and} \qquad I' = (S', L', D', Q', T', H', C', P', N')$$

are classified as the same displayed infimum packet exactly when the bounded sequence rows expose the same finite source windows, the lower-cut schedules expose the same accepted lower observations, the dyadic ledgers carry the same rational comparison budgets, the regular readback rows expose the same finite rational windows, the terminal $\mathsf{Real}^{\uparrow}$ *seals are transported by the displayed* hsame *data, the continuation routes replay the same lower-bound path, and the provenance and local NameCert rows name the same* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *surface. The classifier is rowwise over* $S, L, D, Q, T, H, C, P, N$ *and does not identify packets by host infima, quotient stream equality, selected subsequences, trichotomy, or Real completeness.*

Theorem 8415.3 (Bounded real infimum NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *carrier* $I = (S, L, D, Q, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealInfimum}^{\uparrow}}$ *surface consists of carrier formation for the nine displayed rows, classifier stability for Definition 8415.2, lower-cut schedule exactness through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*, dyadic tolerance accounting through* $\mathsf{DyadicRatCore}^{\uparrow}$*, regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8415.1. Its rows are exactly the bounded source, lower-cut schedule, dyadic ledger, regular readback, terminal seal, transport, replay, provenance, and local naming data.

Use the classifier of Definition 8415.2. It compares accepted packets only by those displayed rows, so a consumer cannot replace the packet by a host-level infimum or by an equality of ambient streams.

The lower-cut, dyadic, regular-readback, and terminal-seal obligations are then row projections. The structural rows H, C, P, N preserve and name the same finite route, and no carrier coordinate contains subsequence choice, trichotomy, or ambient completeness. □

Theorem 8415.4 (Bounded real infimum lower-bound scope)**.** *Every consumer of an accepted* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *packet receives a lower-bound claim only after the bounded sequence row* S*, bounded lower-cut schedule* L*, dyadic tolerance ledger* D*, regular rational readback* Q*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* T *have been displayed. The lower-bound read is therefore the finite route*

$$S \longmapsto L \longmapsto D \longmapsto Q \longmapsto T \longmapsto H, C, P, N$$

rather than a host infimum, selected subsequence, quotient stream equality, trichotomy, or Real completeness principle.

Proof. Begin with S. The bounded source row supplies the only sequence-facing data available to the packet, and it is read through the bounded-sequence surface.

Next read L, D, Q. The lower-cut schedule exposes the accepted lower observations, the dyadic ledger records finite comparison budgets, and the regular readback row exposes the rational windows consumed by those budgets.

Only after those rows are visible can a consumer reach T. The structural rows transport, replay, source, and name the same displayed route, so the lower-bound claim cannot escape to any excluded host-level principle. □

Theorem 8415.5 (Bounded real infimum greatest lower-cut scope)**.** *The greatest-lower-cut information exported by an accepted* $\mathsf{BoundedRealInfimum}^{\uparrow}$ *carrier is scoped to the displayed lower-cut route* $S, L, D, Q, T, H, C, P, N$*. A consumer may compare lower observations through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*, spend dyadic budgets through* $\mathsf{DyadicRatCore}^{\uparrow}$*, read regular rational windows through* $\mathsf{RegSeqRat}^{\uparrow}$*, and reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it may not replace that route by a completed host infimum or by a quotient-level stream argument.*

Proof. Apply the NameCert obligation surface of Definition 8415.3. It exposes the lower-cut schedule, dyadic ledger, regular readback row, and terminal seal as separate obligations of the accepted packet.

Use the lower-bound scope theorem, Definition 8415.4. It orders the consumer route from the bounded source through the lower-cut and rational rows before the terminal seal is available.

Finally inspect the carrier inventory. Since host infima, quotient stream equality, selected subsequences, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the packet, the exported greatest-lower-cut surface remains the finite displayed route. □

Falsifiable prediction: Every accepted $\mathsf{BoundedRealInfimum}^{\uparrow}$ lower-bound read factors through $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows before any consumer can inspect the candidate lower-bound seal.

Independence witness: boundedrealsequence, bounded-real-lower-cut, dyadicratcore, regseqrat, real: $\mathsf{BoundedRealInfimum}^{\uparrow}$ is not bijective to any listed sibling because it binds the bounded source, lower-cut schedule, dyadic budget, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert rows into one lower-bound packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealInfimum}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealInfimum}^{\uparrow}$ packet is generated from a bounded real sequence source, a bounded lower-cut schedule, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedRealInfimum}^{\uparrow}}$, seed)

Scope closed: Definition 8415.1, Definition 8415.2, Definition 8415.3, Definition 8415.4, and Definition 8415.5 seed the finite lower-bound surface for bounded real sequences.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedRealInfimum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host infimum, quotient stream equality, selected subsequences, trichotomy, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, lower-cut schedule exactness, dyadic tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRealInfimumUp.

Chapter 8416

A Concrete Naming Certificate for $\mathsf{ConstantRankTheorem}^{\uparrow}$

Following Spivak and Lee on the constant rank theorem, the $\mathsf{ConstantRankTheorem}^{\uparrow}$ packet records the finite theorem-level route from a displayed smooth map, a stable rank window, and derivative-control data to a local normal-form read. It sits beside the derivative carrier of chapter 122, the inverse-function theorem route of chapter 6206, the implicit-function theorem route of chapter 6354, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the BEDC certificate a differential-topology consumer inspects before using a local normal form; it does not import a host smooth-manifold primitive, quotient chart equality, ambient choice of coordinates, or bridge verification.

Definition 8416.1 (Constant rank theorem carrier)**.** *A* $\mathsf{ConstantRankTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, R, D, I, G, L, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed map row, including source and target coordinate windows, base point, target value, and the finite classifier for the theorem read.*
2. *R is the rank-window row, recording the inspected minor, its nonvanishing witness, the neighbouring minor constraints, and the radius over which the rank is displayed as constant.*
3. *D is the* $\mathsf{Derivative}^{\uparrow}$ *control row for the displayed map on the same source and target windows.*
4. *I is the* $\mathsf{InverseFunctionTheorem}^{\uparrow}$ *handoff row used to isolate the invertible coordinate block supplied by R and D.*
5. *G is the* $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ *handoff row used to solve the complementary coordinate equations inside the same local window.*
6. *L is the local normal-form row, recording the source chart, target chart, rank projection, zero complementary coordinates, and the route by which the normal-form read is obtained from F, R, D, I, G.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for source coordinates, target coordinates, rank-window tests, and endpoint reads.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximation data exposed by S and used by the derivative and rank-window comparisons.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every real-facing coordinate, derivative, rank, and normal-form read.*
10. *H records componentwise* hsame *transport for the map, rank-window, derivative, inverse-function, implicit-function, normal-form, observation, readback, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay through*

$$F, R, D, I, G, L, S, Q, E \longmapsto \mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}.$$

12. *P is the* Pkg *provenance over* $\mathsf{ConstantRankTheorem}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{InverseFunctionTheorem}^{\uparrow}$, $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, D, I, G, L, S, Q, E, H, C, P, N$. The carrier has no coordinate for a host constant-rank theorem, an ambient smooth-manifold structure, selected chart germs, quotient equality of maps, unrecorded coordinate choices, global submersion or immersion theorems, Lean theorem checking, or bridge checking.

Theorem 8416.2 (Constant rank theorem NameCert obligations)**.** *For every accepted* $\mathsf{ConstantRankTheorem}^{\uparrow}$ *carrier*

$$T = (F, R, D, I, G, L, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed map row, rank-window exactness through R, derivative-control exposure through D, inverse-function handoff through I, implicit-function handoff through G, local normal-form exposure through L, finite StreamName observations through S, RegSeqRat readback through Q, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 8416.1. The rows F, R, D are the only source of the displayed map, constant-rank window, and derivative-control data.

The theorem handoffs are then forced. The row I exposes the inverse-function theorem route for the invertible block selected by the rank window, while G exposes the implicit-function theorem route for the complementary equations. The normal-form row L is downstream of those two handoffs and cannot be read before them.

The observation and completion rows come last. Stream observations S feed the regular rational readback Q, and E seals the Real-facing coordinate and derivative reads. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so no host constant-rank theorem, manifold primitive, selected chart germ, quotient equality, unrecorded coordinate choice, global submersion theorem, or bridge check is present. □

Theorem 8416.3 (Constant rank theorem local normal-form route)**.** *Every accepted* $\mathsf{ConstantRankTheorem}^{\uparrow}$ *local normal-form read factors through the finite route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{InverseFunctionTheorem}^{\uparrow} \longmapsto \mathsf{ImplicitFunctionTheorem}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Rea}$$

as instantiated by D, I, G, S, Q, E together with the displayed map row F, rank-window row R, and local normal-form row L. A consumer may read local source and target chart data, the rank projection, zero complementary coordinates, finite observations, regular rational readback, terminal

Real sealing, componentwise transport, displayed replay, provenance, and local naming; it may not read a host smooth-manifold theorem, quotient chart equality, selected chart germ, untracked coordinate choice, global immersion or submersion theorem, or normalization oracle.

Proof. Start with the accepted carrier $T = (F, R, D, I, G, L, S, Q, E, H, C, P, N)$. The displayed map and rank-window rows fix the coordinate block and the local radius, while D supplies the derivative-control data in the same window.

The local theorem route first uses I to isolate the invertible block and then uses G to solve the complementary coordinate equations. The row L records the resulting source chart, target chart, rank projection, and zero complementary coordinates only after those handoffs are visible.

The finite observation route then reads source, target, rank, and endpoint data through S, rational comparisons through Q, and Real-facing coordinates through E. Transport, replay, provenance, and local naming preserve the same path through H, C, P, N. Since the carrier has no host theorem, manifold primitive, quotient chart equality, chart-germ selector, untracked coordinate choice, global theorem, or oracle coordinate, the normal-form read is exhausted by the displayed BEDC rows. □

Falsifiable prediction: Every accepted $\mathsf{ConstantRankTheorem}^{\uparrow}$ local normal-form read exposes one displayed map row, one rank-window row, one Derivative control row, one InverseFunctionTheorem handoff, one ImplicitFunctionTheorem handoff, one local normal-form row, StreamName observations, RegSeqRat readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: derivative, inverse-function-theorem, implicit-function-theorem, streamname, regseqrat, real: $\mathsf{ConstantRankTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rank-window data, derivative control, inverse and implicit theorem handoffs, local normal-form data, finite observation, rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstantRankTheorem}^{\uparrow}$ packet is generated from a displayed map row, rank-window row, Derivative control row, InverseFunctionTheorem handoff, ImplicitFunctionTheorem handoff, local normal-form row, StreamName observations, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8416.1, Definition 8416.2, and Definition 8416.3 seed the finite theorem-level local normal-form route over rank-window, derivative, theorem-handoff, observation, regular rational, and Real completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstantRankTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host constant-rank theorems, ambient smooth-manifold primitives, selected chart germs, quotient chart equality, unrecorded coordinate choices, global immersion or submersion theorems, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rank-window exactness, derivative-control exposure, inverse-function handoff, implicit-function handoff, local normal-form exposure, Real completion routing, provenance exactness, and local naming for BEDC.Derived.ConstantRankTheoremUp.

Chapter 8417

A Concrete Naming Certificate for $\mathsf{SpeckerTheorem}^{\uparrow}$

Following Specker and the Bishop–Bridges treatment of constructive real analysis, the $\mathsf{SpeckerTheorem}^{\uparrow}$ packet records the finite NameCert boundary of the classical-looking assertion that a bounded monotone rational sequence need not provide a located supremum as constructive data. It sits after the $\mathsf{SpeckerSequence}^{\uparrow}$ carrier of chapter 1406, the Markov-sensitive refusal surface of chapter 1666, the terminal real-name seal of chapter 38, the regular rational route of chapter 301, and the dyadic tolerance ledger of chapter 291. The chapter records the theorem as a boundary object for Real and Completion consumers; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, real trichotomy, quotient stream equality, countable choice, or a selected completed real.

Definition 8417.1 (Specker theorem carrier)**.** *A* $\mathsf{SpeckerTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *source row carrying the monotone bounded rational stream.*
2. *M is the* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *refusal row that keeps negative finite inspection from becoming a global principle.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed rational windows of S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational bounds and cut tests used in the theorem boundary.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after S, R, D and the refusal row M have been displayed.*
6. *L is the located-supremum absence ledger, recording that no located comparison witness for the supremum is present in the packet.*
7. *H records componentwise* hsame *transport for the source, refusal, readback, dyadic, seal, absence, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the Specker source through regular readback, dyadic tolerance inspection, Markov-boundary refusal, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerTheorem}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SpeckerTheorem}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, M, R, D, E, L, H, C, P, N$. It has no coordinate for an ambient completion theorem, real trichotomy, quotient equality of real names, countable choice, Markov's principle as a rule, or a selected supremum.

Theorem 8417.2 (Specker theorem NameCert obligations)**.** *For every accepted* $\mathsf{SpeckerTheorem}^\uparrow$ *carrier $T = (S, M, R, D, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpeckerTheorem}^\uparrow}$ *surface consists exactly of Specker-source admission through S, Markov-boundary refusal through M, regular rational readback through R, dyadic tolerance accounting through D, terminal Real sealing through E, located-supremum absence through L, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8417.1. Its analytic rows are precisely the Specker source, the Markov-boundary refusal, the regular readback, the dyadic tolerance ledger, the Real seal, and the located-supremum absence ledger.

The Specker source row is read through the existing boundary of Definition 1406.2 and Definition 1406.7. The Markov row is read through Definition 1666.2 and Definition 1666.3, so negative inspection remains named data rather than an imported rule.

The remaining rows are structural. Componentwise transport, continuation replay, provenance, and local naming preserve the displayed route and do not add a located supremum, quotient equality, countable-choice selector, trichotomy branch, or ambient completion coordinate. Thus the listed rows exhaust the NameCert obligations. □

Theorem 8417.3 (Specker theorem Real-completion boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{SpeckerTheorem}^\uparrow$ *carrier must read the route*

$$\mathsf{SpeckerSequence}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{MarkovPrincipleBoundary}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

together with the located-supremum absence row L and the structural rows H, C, P, N. The packet may be cited as the Specker theorem boundary, but it cannot be consumed as completeness of $\mathsf{Real}^\uparrow$*, real trichotomy, quotient stream equality, countable choice, Markov's principle as a rule, or a selected completed supremum.*

Proof. Start with the $\mathsf{SpeckerSequence}^\uparrow$ source row. By the SpeckerSequence exposure and non-located supremum boundary, S supplies finite monotone bounded rational windows and no located-supremum witness.

Next read the regular and dyadic rows. The rows R and D expose the regular rational readback and the tolerance ledger used by a completion-facing consumer, but they remain finite displayed rows rather than a host completion operator.

Now read the Markov-boundary row M and the absence ledger L. The Markov boundary keeps finite negative inspection from becoming a global rule, while L records the missing located comparison witness for the supremum. The terminal seal E is therefore only a boundary seal for this displayed route.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no coordinate contains ambient completeness of $\mathsf{Real}^\uparrow$, trichotomy, quotient equality, countable choice, Markov's principle as a rule, or a selected completed supremum, no Real or Completion consumer can read such data from the carrier. □

Falsifiable prediction: Every $\mathsf{SpeckerTheorem}^\uparrow$ Real or Completion read factors through a SpeckerSequence source row, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a

MarkovPrincipleBoundary refusal row, a terminal RealUp seal, a located-supremum absence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: speckersequence, markov-principle-boundary, regseqrat, dyadicratcore, real: $\mathsf{SpeckerTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the theorem-level boundary tying a Specker source, regular and dyadic readback, Markov-sensitive refusal, a Real seal, and a located-supremum absence ledger into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{SpeckerTheorem}^{\uparrow}$ packet is generated from a SpeckerSequence source row, MarkovPrinciple-Boundary refusal row, RegSeqRat readback, DyadicRatCore tolerance ledger, a terminal RealUp seal, a located-supremum absence ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpeckerTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8417.1, Definition 8417.2, and Definition 8417.3 seed the finite theorem boundary separating Specker-source data from ambient Real completeness or trichotomy claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpeckerTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, real trichotomy, quotient stream equality, countable choice, Markov's principle as a rule, selected completed suprema, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Real-completion boundary nonescape, transport and replay stability, provenance, and local naming for BEDC.Derived.SpeckerTheoremUp.

Chapter 8418

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealSegment}^{\uparrow}$

Following Bishop–Bridges constructive analysis, $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ names the finite certificate surface for a located real interval whose endpoints are presented by regular rational data and whose order evidence is explicitly displayed. The packet sits over the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{LocatedRealOrder}^{\uparrow}$ siblings of chapter 291, chapter 126, chapter 301, chapter 38, **??**, and chapter 8337. It gives later located-interval and completion arguments a routed endpoint packet, not a classical interval object.

Definition 8418.1 (Bishop located real segment carrier)**.** *A* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, O, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are endpoint rows, each naming a left or right* $\mathsf{Real}^{\uparrow}$ *endpoint through the same finite packet rather than through host real equality.*
2. *O is the located-order evidence row, read through* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{LocatedRealOrder}^{\uparrow}$*, stating that the displayed left endpoint is below the displayed right endpoint at the requested finite tolerances.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect endpoint names.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording endpoint radii, comparison budgets, and dyadic inclusion tests.*
6. *E is the pair of terminal* $\mathsf{Real}^{\uparrow}$ *seals reached after S, Q, D, O have been replayed.*
7. *H records componentwise* hsame *transport for endpoint rows, located-order evidence, stream windows, regular readbacks, dyadic tolerances, terminal seals, and structural rows.*
8. *C records the displayed* Cont *routes by which consumers replay the endpoint, order, window, readback, tolerance, seal, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LocatedRealOrder}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealSegment}^{\uparrow}}$ *row.*

The carrier stores a located segment boundary as finite endpoint and order data. It has no coordinate for Real completeness, trichotomy, quotient real equality, selected representatives, or open-cover compactness.

Definition 8418.2 (Bishop located real segment classifier)**.** *Two* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ *carriers*

$$I = (L, R, O, S, Q, D, E, H, C, P, N) \qquad \textit{and} \qquad I' = (L', R', O', S', Q', D', E', H', C', P', N')$$

are classified as the same displayed located segment exactly when the left and right endpoint rows match componentwise, the located-order rows expose the same endpoint order boundary, the stream windows inspect the same finite endpoint prefixes, the regular rational readbacks and dyadic tolerance ledgers carry the same comparison budgets, the terminal $\mathsf{Real}^{\uparrow}$ *seals are transported by the displayed* hsame *data, and the continuation, provenance, and local NameCert rows name the same* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ *packet. The classifier is rowwise over* $L, R, O, S, Q, D, E, H, C, P, N$ *and does not identify intervals by quotient real equality, selected representatives, trichotomy, or a host subset of the real line.*

Theorem 8418.3 (Bishop located real segment NameCert obligations)**.** *For an accepted* $\mathsf{BishopLocatedRealSegmen}$ *carrier* $I = (L, R, O, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealSegment}^{\uparrow}}$ *surface consists of seven finite obligation rows:*

1. *carrier formation for the two endpoint rows, located-order evidence, finite stream windows, regular rational readbacks, dyadic tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seals, transports, continuations, provenance, and local naming data;*
2. *classifier stability for the componentwise comparison of Definition 8418.2;*
3. *endpoint order stability, stating that the displayed located-order row is transported together with the endpoint rows and dyadic comparison budgets;*
4. *stream-window readback, stating that endpoint prefixes are inspected only through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
5. *dyadic tolerance exactness, stating that interval inclusion and endpoint comparison checks use only the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger;*
6. *terminal seal routing, stating that endpoint consumers reach the two* $\mathsf{Real}^{\uparrow}$ *seals only after the window, readback, tolerance, and located-order rows have been replayed;*
7. *ledger non-escape, excluding Real completeness, trichotomy, quotient real equality, selected representatives, and open-cover compactness.*

Proof. Carrier formation is projection from Definition 8418.1. The packet displays exactly the endpoint rows, located-order evidence, finite stream windows, regular readbacks, dyadic tolerance ledger, terminal seals, transport rows, continuation routes, provenance, and local name.

Classifier stability follows from Definition 8418.2. Transport along H may replace visible rows only by rows with the same endpoint order boundary, the same stream windows, and the same dyadic comparison budgets, while C preserves the replay order.

The readback and exactness rows are the displayed roles of S, Q, D, O. The stream row exposes endpoint prefixes, the regular row reads them as rational approximants, the dyadic ledger records the finite tolerances, and the located-order row binds those comparisons to the endpoint boundary.

Terminal seal routing is the role of E. A consumer may read the left or right real endpoint only after the stream, regular, dyadic, and located-order rows have been replayed.

The non-escape row follows from the carrier inventory. No displayed field carries Real completeness, trichotomy, quotient real equality, selected representatives, or open-cover compactness. □

Theorem 8418.4 (Bishop located real segment endpoint order stability)**.** *Let $I = (L, R, O, S, Q, D, E, H, C, P, N)$ be an accepted* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ *carrier. Every endpoint-facing transport of I preserves the located order row O together with the endpoint rows L, R and dyadic tolerance ledger D. Hence the transported packet remains a located segment certificate and cannot turn into an unordered pair of real names, a trichotomy witness, or a quotient interval.*

Proof. Project L, R, O, D, H from the carrier of Definition 8418.1. The endpoint rows and located-order row are displayed in the same packet, and the dyadic tolerance ledger records the finite comparison budgets attached to them.

Use the classifier of Definition 8418.2. It compares accepted packets by the componentwise endpoint, order, stream, readback, tolerance, seal, transport, replay, provenance, and naming rows, not by a host subset or quotient equality of real names.

The transport row H therefore preserves the order boundary whenever it transports the endpoints. Since C, P, N replay and source the same finite packet, no consumer obtains an unordered pair, a trichotomy decision, or a quotient interval from the transported carrier. □

Theorem 8418.5 (Bishop located real segment Real seal non-escape)**.** *An accepted* $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ *packet exposes its endpoints only as the terminal* $\mathsf{Real}^{\uparrow}$ *seals E reached through the displayed route*

$$S \longmapsto Q \longmapsto D \longmapsto O \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

In particular, it does not export Real completeness, trichotomy, quotient real equality, selected representatives, or open-cover compactness.

Proof. The only endpoint-facing value coordinate in the carrier is E. The rows S, Q, D, O supply the stream windows, regular rational readbacks, dyadic tolerances, and located-order evidence that must be replayed before E is consumed.

Continuation replay through C follows the displayed route, and componentwise transport through H preserves the same endpoint and order rows. Provenance through P and local naming through N record the packet as a $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ certificate.

Since no carrier field stores the excluded principles, the terminal seals cannot be read as a completeness theorem, trichotomy decision, quotient real equality, selected representative, or open-cover compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealSegment}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealSegment}^{\uparrow}$ packet is generated from two RealUp endpoint rows, located-order evidence, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, terminal RealUp seals, componentwise hsame transports, Cont replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedRealSegment}^{\uparrow}}$, seed)

Scope closed: Definition 8418.1, Definition 8418.2, Definition 8418.3, Definition 8418.4, and Definition 8418.5 seed the finite located real segment surface over displayed endpoint, order, stream, regular rational, dyadic, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedRealSegment}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient real equality, selected representatives, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, endpoint order stability, stream-window readback, dyadic tolerance exactness, terminal seal routing, and Real seal non-escape for the displayed packet.

Chapter 8419

A Concrete Naming Certificate for $\mathsf{VanKampen}^{\uparrow}$

Following Seifert and van Kampen, the $\mathsf{VanKampen}^{\uparrow}$ packet records the finite cover-generator route by which local loop data over a path-connected open cover is handed to a fundamental-group presentation consumer. It sits beside the homotopy surface of chapter 156, the finite based-loop algebra of chapter 3368, the open-cover carrier of chapter 6731, the path-connected source of chapter 6885, and the group operation surface of chapter 41. The certificate names cover rows, overlap rows, local path-loop rows, generator rows, relation rows, and a downstream $\mathsf{FundamentalGroup}^{\uparrow}$ consumer route; it does not import a quotient group, a host group presentation theorem, selected representatives of homotopy classes, or ambient path-space equality.

Definition 8419.1 ($\mathsf{VanKampen}^{\uparrow}$ finite cover-generator carrier)**.** *A* $\mathsf{VanKampen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Pi = (X, x_0, U, O, A, L, G, R, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient topological source row and x_0 is the displayed basepoint row. The source must expose the* $\mathsf{PathConnectedSpace}^{\uparrow}$ *route that makes every inspected open row reachable from x_0 by finite path data.*
2. *U is the finite open-cover row family, read through the open-cover surface of chapter 6731. Each member row records its open test, basepoint-reachability witness, and finite cover index.*
3. *O is the overlap row family. It lists the nonempty pairwise and finite multi-overlap windows whose path components are inspected before a relation row is accepted.*
4. *A is the local path-loop row family. Its entries are based paths, transported basepoints, and local loop rows read through the deformation classifier of chapter 156.*
5. *L is the local fundamental-group row family. Each entry records the* $\mathsf{FundamentalGroup}^{\uparrow}$ *packet carried by one cover member or overlap component, together with the inclusion route back to X.*
6. *G is the generator ledger. It records one finite generator row for each admitted local loop or overlap transfer whose image is exported to the global consumer.*
7. *R is the relation ledger. It records the finite overlap identifications, conjugation-by-path adjustments, and null-loop seals that the global consumer must read before classifying the generated group row.*

8. *F is the* $\mathsf{FundamentalGroup}^{\uparrow}$ *consumer row. It receives only the displayed generator and relation rows G, R together with their source paths and homotopy comparisons, not a host quotient group.*
9. *H records componentwise* hsame *transport for $X, x_0, U, O, A, L, G, R, F, C, P, N$.*
10. *C records displayed* Cont *replay for open-cover admission, overlap admission, path-loop readback, local fundamental-group readback, generator export, relation export, and the global consumer handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{VanKampen}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{OpenCover}^{\uparrow}$, $\mathsf{PathConnectedSpace}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{FundamentalGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VanKampen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed cover, overlap, path-loop, generator, relation, consumer, transport, replay, provenance, and naming rows. Thus the packet is a finite cover-generator certificate, not a completed quotient presentation of $\pi_1(X, x_0)$.

Theorem 8419.2 ($\mathsf{VanKampen}^{\uparrow}$ NameCert obligations)**.** *For every accepted* $\mathsf{VanKampen}^{\uparrow}$ *carrier*

$$\Pi = (X, x_0, U, O, A, L, G, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VanKampen}^{\uparrow}}$ *surface consists exactly of source admission for X, x_0 through* $\mathsf{PathConnectedSpace}^{\uparrow}$, *finite open-cover admission through U, overlap-window admission through O, local path-loop admission through A, local* $\mathsf{FundamentalGroup}^{\uparrow}$ *admission through L, generator export through G, relation export through R, target handoff through F, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{Group}^{\uparrow}$ *or* $\mathsf{FundamentalGroup}^{\uparrow}$ *consumer of the packet must inspect those rows before it may read the generated global algebraic surface.*

Proof. Carrier admission is projection from Definition 8419.1. The rows $X, x_0, U, O, A, L, G, R, F, H, C, P, N$ expose the topological source, basepoint, cover windows, overlap windows, local path-loop data, local fundamental-group data, generator and relation ledgers, consumer handoff, transport, replay, provenance, and local name.

The source side is forced before any algebraic read. The $\mathsf{PathConnectedSpace}^{\uparrow}$ row supplies the displayed route from x_0 to each inspected cover member, while U and O supply the finite open-cover and overlap windows. The local loop rows in A are then read through $\mathsf{Homotopy}^{\uparrow}$, so every generator has a visible path and deformation provenance.

The algebraic side is only a consumer surface. The local $\mathsf{FundamentalGroup}^{\uparrow}$ rows in L produce finite loop-composition data, and G, R export exactly the generator and relation endpoints the global consumer may inspect. Since F receives these rows through H, C, P, N, no step introduces a host quotient group, an uncarried presentation, or a selected homotopy-class representative. □

Theorem 8419.3 ($\mathsf{VanKampen}^{\uparrow}$ cover-generator handoff)**.** *Let $\Pi = (X, x_0, U, O, A, L, G, R, F, H, C, P, N)$ be an accepted* $\mathsf{VanKampen}^{\uparrow}$ *carrier. Any downstream* $\mathsf{FundamentalGroup}^{\uparrow}$ *consumer of the global cover-generated surface factors through the finite route*

$$X, x_0, U, O, A, L, G, R, H, C, P, N \longmapsto F.$$

The consumer may read the displayed generators, overlap relations, conjugating path rows, and null-loop seals, but it cannot replace them by a host Seifert–van Kampen theorem, a quotient group presentation outside BHist, *or an ambient path-component equality.*

Proof. Begin with the cover and overlap rows. The open-cover ledger U and overlap ledger O determine which local path-loop rows in A are eligible for export, and the path-connected source row keeps the transport from x_0 visible.

Read the local algebraic rows next. Each local row in L is already a $\mathsf{FundamentalGroup}^{\uparrow}$ packet, so its loop composition, inverse, identity, and homotopy comparison data are finite NameCert rows. The generator ledger G selects only endpoints carried by those local rows.

Finally consume the relation ledger R through F. The relation rows list the overlap identifications and conjugating paths that the global consumer is allowed to use. Componentwise transport H, displayed replay C, provenance P, and the local name N keep this route inside the same finite packet, so the handoff supplies a cover-generated boundary without importing the ambient quotient theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{VanKampen}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VanKampen}^{\uparrow}}$, seed)

Scope closed: Definition 8419.1, Definition 8419.2, and Definition 8419.3 seed the finite cover, overlap, path-loop, local fundamental-group, generator, relation, consumer, transport, replay, provenance, and local naming surface for the Seifert–van Kampen cover-generator packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{VanKampen}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host Seifert–van Kampen theorem, quotient group presentations, selected loop-class representatives, universal covers, monodromy actions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite open-cover admission, overlap-window admission, path-loop admission, local fundamental-group admission, generator export, relation export, target handoff, transport, replay, provenance, and local naming for BEDC.Derived.VanKampenUp.

Constructive story: A $\mathsf{VanKampen}^{\uparrow}$ packet is generated from a displayed topological source and basepoint, finite open-cover rows, overlap rows, path-loop rows, local FundamentalGroupUp rows, generator rows, relation rows, a FundamentalGroupUp consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8420

A Concrete Naming Certificate for $\mathsf{HarmonicFunction}^{\uparrow}$

Following Weyl and Bishop–Bridges, the $\mathsf{HarmonicFunction}^{\uparrow}$ packet records the finite NameCert surface of a harmonic-function read over a compact metric chart. It sits over the terminal real-name rows of chapter 38, the metric carrier of chapter 57, the compact finite-net surface of chapter 129, the graph and modulus rows of chapter 130, and the local differential-equation vocabulary of $\mathsf{Pde}^{\uparrow}$. The packet names a constructive finite-means harmonicity certificate only; it does not assert global Dirichlet solvability, classical smoothness, distributional equality, unique continuation, maximum principle completeness, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8420.1 (Harmonic-function carrier)**.** *A* $\mathsf{HarmonicFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (R, M, K, F, L, A, V, E, T, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *value row, exposing the real-name windows consumed by the function values.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *chart row, recording the source distance comparisons used by ball and boundary reads.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-window row for the compact chart windows on which the certificate is inspected.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the pointwise function reads and their local continuity witnesses.*
5. *L is the* $\mathsf{Pde}^{\uparrow}$ *Laplace row, naming the displayed local equation $\Delta F = 0$ only through finite stencil, ball, or chart data.*
6. *A is the averaging row. It records the finite boundary samples, weights, tolerances, and mean-value readback requested around each inspected ball.*
7. *V is the verification row requiring the displayed Laplace row and averaging row to agree on every accepted inspection window.*
8. *E records componentwise* hsame *transport for real values, metric balls, compact windows, continuous graph reads, Laplace rows, averaging rows, verification rows, replay, provenance, and local naming.*

9. *T records displayed* Cont *replay for the route*

$$R, M, K, F, L, A \longmapsto V \longmapsto E, C, P, N.$$

10. *C is the consumer handoff row for local Dirichlet, mean-value, and PDE-facing requests that accept only the displayed finite route.*

11. *P is the* Pkg *provenance over* $\mathsf{HarmonicFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HarmonicFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for a host smooth function, selected Poisson kernel, global boundary-value solution, distributional equality, quotient map equality, excluded-middle regularity argument, or ambient completeness principle.

Theorem 8420.2 (Harmonic-function NameCert obligations)**.** *For every accepted* $\mathsf{HarmonicFunction}^{\uparrow}$ *carrier*

$$H = (R, M, K, F, L, A, V, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HarmonicFunction}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for* $\mathsf{Real}^{\uparrow}$ *values,* $\mathsf{Met}^{\uparrow}$ *chart distances,* $\mathsf{CompactMetric}^{\uparrow}$ *finite windows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph reads,* $\mathsf{Pde}^{\uparrow}$ *Laplace rows, averaging rows, verification rows, transport, replay, consumer handoff, provenance, and local naming;*

2. *metric-ball exactness, requiring every averaging window and Laplace stencil to be attached to the same displayed chart row M and compact window row K;*

3. *continuity compatibility, requiring every value consumed by L or A to be exposed first through F and read through R;*

4. *Laplace–mean agreement, requiring V to compare the displayed* $\mathsf{Pde}^{\uparrow}$ *Laplace row with the finite averaging row before any harmonic read is exported;*

5. *non-escape of host smoothness, global Dirichlet solvability, selected Poisson kernels, distributional equality, quotient equality of functions, choice principles, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 8420.1. The analytic rows are precisely R, M, K, F, L, A, V, and the structural rows E, T, C, P, N preserve, replay, hand off, source, and name the same finite route.

Metric-ball exactness is local. The row M supplies the distance comparisons for balls and boundaries, while K supplies the compact finite windows on which those comparisons are inspected. Thus neither L nor A can choose a detached chart.

Continuity compatibility then forces every value read through the graph row F and the real row R before a Laplace or averaging check may consume it. The verification row V is the only row that compares the displayed $\mathsf{Pde}^{\uparrow}$ equation with the finite mean-value data.

Finally, E, T, C, P, N add only transport, replay, consumer routing, provenance, and local naming. Since the carrier has no row for a host smooth function, global boundary-value solution, selected kernel, quotient equality, choice principle, or ambient completeness theorem, those claims cannot be exported by the NameCert surface. □

Theorem 8420.3 (Harmonic-function mean-value route)**.** *Every accepted* $\mathsf{HarmonicFunction}^{\uparrow}$ *mean-value or local Dirichlet read factors through*

$$\mathsf{Real}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Pde}^{\uparrow}$$

before it reaches the consumer handoff row. In particular, the exported harmonicity read is the displayed agreement between a finite Laplace row and a finite averaging row inside the same compact metric chart, not a host-theoretic solution operator.

Proof. Apply Definition 8420.2. Carrier admission lists the real, metric, compact, continuous-map, PDE, averaging, and verification rows as the entire analytic surface.

The metric-ball exactness and continuity compatibility clauses bind those rows to the same chart and graph reads. Hence a mean-value read cannot bypass R, M, K, F before inspecting L and A.

The verification row V then compares the Laplace and averaging displays, and the handoff row C exports only the route replayed by E, T and sourced by P, N. Therefore a local Dirichlet or harmonicity consumer receives finite NameCert data rather than a global solution operator, selected kernel, quotient function, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{HarmonicFunction}^{\uparrow}$ packet exposes RealUp value rows, MetricSpaceUp chart rows, CompactMetricUp finite windows, ContinuousMapUp graph rows, PdeUp Laplace rows, finite averaging rows, verification rows, transport, replay, consumer handoff, provenance, and local naming.

Independence witness: real, metricspace, compactmetric, continuousmap, pde: $\mathsf{HarmonicFunction}^{\uparrow}$ is not identical to any listed sibling because it binds finite mean-value rows and local Laplace rows into one compact metric chart certificate before exporting harmonic-function reads.

Closure status for $\mathsf{NameCert}_{\mathsf{HarmonicFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HarmonicFunction}^{\uparrow}$ packet is generated from RealUp value rows, MetricSpaceUp chart distance rows, CompactMetricUp finite windows, ContinuousMapUp graph rows, PdeUp Laplace rows, finite averaging rows, a Laplace–mean verification row, componentwise hsame transport, displayed Cont replay, a consumer handoff row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HarmonicFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8420.1, Definition 8420.2, and Definition 8420.3 seed the compact metric finite-means packet for local harmonic-function reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HarmonicFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global Dirichlet solvability, classical smoothness, distributional equality, unique continuation, maximum principle completeness, selected Poisson kernels, quotient equality of maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-ball exactness, continuity compatibility, Laplace–mean agreement, consumer handoff, provenance, and local naming for BEDC.Derived.HarmonicFunctionUp.

Chapter 8421

A Concrete Naming Certificate for $\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}$

The $\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}$ packet realizes the local stop-face classifier route of subsection 9462.28.17 as finite BEDC NameCert data. It consumes the six-coordinate source atlas of chapter 2234, the stop-face suffix surface of chapter 1374, the codon-window spectral inverse sibling of chapter 2161, and the deterministic transition-matrix rows of chapter 2245. The packet records the $\mathrm{U}+\mathrm{RR}$ face, the local suffix classifier $00, 01, 10, 11$, the exclusion of UGG as the 11 corner, and the surviving stop rows UAA, UAG, and UGA; it is not a global no-11 predicate, a full genetic-code table, an amino-acid classifier, a stochastic readthrough model, or a quotient codon semantics.

Definition 8421.1 (Codon-window stop-face classifier carrier)**.** *A* $\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, A, U, S, X, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the six-bit codon coordinates and Flat6 readback.*
2. *A is the two-axis atlas row naming the* $\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}$ *base reads used by the codon-window bridge.*
3. *U is the face-selection row. It selects the local* $\mathrm{U}+\mathrm{RR}$ *square before any stop decision is inspected.*
4. *S is the suffix-coordinate row. It displays the four local family suffixes* $00, 01, 10, 11$ *inside the selected face.*
5. *X is the* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *legality row, retaining exactly the* $X_2 = \{00, 01, 10\}$ *suffixes.*
6. *D is the stop-decision row. It maps* $00, 01, 10$ *to the stop rows* $\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}$ *and maps* 11 *to the excluded* UGG *corner.*
7. *E records componentwise* hsame *transport for source windows, atlas rows, face rows, suffix rows, legality rows, decision rows, replay, provenance, and local naming.*
8. *H records displayed* Cont *replay for the route*

$$W, A, U, S \longmapsto X, D \longmapsto E, C, P, N.$$

9. *C is the classifier handoff row used by local stop-face consumers that accept only the displayed finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{StopCodonZeckendorfSuffi}$ $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, *the codon-window-six bridge,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}}$ *row.*

*The classifier compares accepted packets only through these eleven rows. It has no coordinate for a global flattened no-*11 *predicate, a full translation table, amino-acid class identity, stochastic readthrough, quotient codon equality, selected biological code tables, or global biological universality.*

Theorem 8421.2 (Codon-window stop-face classifier NameCert obligations)**.** *For every accepted* $\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}$ *carrier*

$$F = (W, A, U, S, X, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CodonWindowStopFaceClassifier}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for six-bit source windows, two-axis atlas rows, local face selection, suffix-coordinate rows, suffix-legality rows, stop-decision rows, transport, replay, classifier handoff, provenance, and local naming;*

2. *source-atlas exactness, requiring every local stop-face read to pass through W and A before a suffix is classified;*

3. *face locality, requiring the decision row to be attached to the selected* $\mathrm{U} + \mathrm{RR}$ *square U rather than to the whole flattened six-bit window;*

4. *suffix exactness, requiring S and X to retain* $00, 01, 10$ *and exclude only* 11 *inside that face;*

5. *stop-decision determinacy, requiring D to export exactly* UAA, UAG, *and* UGA *as stop rows while keeping* UGG *outside the stop class;*

6. *non-escape of global no-*11 *classification, full genetic-code tables, amino-acid classifiers, stochastic readthrough, quotient codon equality, selected biological code tables, and biological universality claims.*

Proof. Project the accepted carrier of Definition 8421.1. The visible analytic rows are W, A, U, S, X, D, and the structural rows E, H, C, P, N preserve, replay, hand off, source, and locally name the same finite route.

Source-atlas exactness comes first. The six-bit source row W and atlas row A expose the codon window before any face or suffix classifier can be read.

Face locality is then enforced by U. The suffix rows S and X are attached only to the selected $\mathrm{U} + \mathrm{RR}$ square, so the classifier cannot become a global no-11 predicate on the flattened six-bit string.

The decision row D is deterministic over the displayed suffixes: it exports $00, 01, 10$ as UAA, UAG, and UGA stop rows and keeps 11 as the excluded UGG corner. Since the refused biological objects have no carrier coordinate and no replay edge, the listed rows exhaust the local NameCert surface. □

Theorem 8421.3 (Codon-window stop-face exclusion route)**.** *Every accepted* CodonWindowStopFaceClassifier$^{\uparrow}$ *read factors through*

CodonSixBitWindow$^{\uparrow}$, StopCodonZeckendorfSuffix$^{\uparrow}$, CodonWindowSpectralInverse$^{\uparrow}$, CodonTransitionMatrix$^{\uparrow}$

as sibling evidence before it may export a local stop decision. The exported decision is the finite U + RR *face classifier that excludes* UGG *and retains* UAA*,* UAG*, and* UGA*, not the global no-adjacent predicate of Definition 9462.1090.*

Proof. Apply Definition 8421.2. Carrier admission forces the read through W, A, U, S, X, D before any handoff row can be consumed.

The sibling route is finite. The source row W is the six-coordinate atlas packet, the legality row X is the stop-suffix packet, and the provenance row P names the spectral-inverse and transition-matrix consumers that inspect the same stop face.

By suffix exactness and stop-decision determinacy, only $00, 01, 10$ are exported as stop rows. The 11 suffix is the UGG corner and remains outside the stop class.

Thus the classifier is local to the selected face. It records the cannot-claim boundary of Definition 9462.1090 as concrete NameCert data, rather than importing a global flattened predicate or a full biological translation table. □

Falsifiable prediction: Every accepted CodonWindowStopFaceClassifier$^{\uparrow}$ packet exposes CodonSixBitWindowUp source rows, two-axis atlas rows, U+RR face selection, suffix coordinates, StopCodonZeckendorfSuffixUp legality, stop-decision rows, transport, replay, classifier handoff, provenance, and local naming. A packet that classifies stop codons by global no-adjacent-11 on the flattened six-bit window fails the face-locality row.

Independence witness: codon-six-bit-window, stopcodonzeckendorfsuffix, codon-window-spectral-inverse, codon-transition-matrix: CodonWindowStopFaceClassifier$^{\uparrow}$ is not identical to any listed sibling because it binds source atlas, local face selection, suffix legality, and deterministic UGG exclusion into one local stop-face classifier packet.

Closure status for NameCert$_{\text{CodonWindowStopFaceClassifier}^{\uparrow}}$**.**

Constructive story: A CodonWindowStopFaceClassifier$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source rows, two-axis atlas rows, a U+RR face-selection row, suffix-coordinate rows, StopCodonZeckendorfSuffixUp legality rows, a deterministic stop-decision row, componentwise hsame transport, displayed Cont replay, a classifier handoff row, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CodonWindowStopFaceClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 8421.1, Definition 8421.2, and Definition 8421.3 seed the local U+RR stop-face classifier and its UGG exclusion route.

Formal verification: FormalStatus(NameCert$_{\text{CodonWindowStopFaceClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global flattened no-adjacent classification, full genetic-code table identity, amino-acid classifiers, stochastic readthrough, quotient codon equality, selected biological code tables, biological universality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-atlas exactness, face locality, suffix exactness, stop-decision determinacy, transport, replay, provenance, and local naming for BEDC.Derived.CodonWindowStopFaceClassifierUp.

Chapter 8422

A Concrete Naming Certificate for $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$

Following Riemann, Darboux, and Bishop–Bridges constructive analysis, the $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ packet records the finite uniform-partition route by which a displayed real-valued integrand is admitted to a Riemann-integrability criterion. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the criterion surface of chapter 8064, and the compact-continuity route of chapter 7133. The packet names a finite equal-mesh integrability surface; it does not import host-level limits of partitions, quotient equality of functions, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8422.1 (Riemann integrability uniform-partition carrier)**.** *A* $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, F, M, E, S, Q, R, D, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *closed-interval source row, exposing the two endpoint seals and the admitted interval direction.*
2. *F is the integrand readback row, storing only displayed evaluation windows on the interval source.*
3. *M is the uniform mesh row, recording a positive dyadic mesh-width and a finite cell count.*
4. *E is the equal-cell ledger, listing the uniform partition cells and their endpoint coverage through I and M.*
5. *S is the finite Riemann-sum row, read through the same cells and the displayed integrand windows.*
6. *Q is the* $\mathsf{RiemannSumCriterion}^{\uparrow}$ *comparison row, recording the finite criterion request over the displayed uniform mesh.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the sum values and comparison tolerances.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for mesh size, oscillation budget, and rational error comparison.*
9. *T is the terminal handoff row, by which a* $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$ *or integration-facing consumer may read the accepted uniform-partition criterion.*

10. *H records componentwise* hsame *transport for interval, integrand, mesh, cells, sums, criterion, rational readback, dyadic tolerance, terminal handoff, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$I, F, M, E \longmapsto S \longmapsto Q \longmapsto R, D \longmapsto T.$$

12. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RiemannSumCriterion}^{\uparrow}$, $\mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, M, E, S, Q, R, D, T, H, C, P, N$. It has no coordinate for a host partition limit, selected refining sequence, quotient function equality, measure completion, countable choice, or ambient real completeness.

Theorem 8422.2 (Riemann integrability uniform-partition obligations)**.** *For every accepted* RiemannIntegrabilityUn *carrier*

$$U = (I, F, M, E, S, Q, R, D, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}}$ *surface consists exactly of interval admission, integrand-window exposure, uniform-mesh admission, equal-cell endpoint coverage, finite Riemann-sum readback, Riemann-sum criterion comparison, regular rational readback, dyadic tolerance accounting, terminal consumer handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8422.1. The rows I, F, M, E expose the interval, integrand windows, uniform mesh, and equal cells before any sum or criterion is read.

The integration-facing data are then S and Q. The finite sum row S is tied to the same equal-cell ledger, and the criterion row Q is the only place where the Riemann-sum comparison is recorded.

The numerical boundary is R and D. Regular rational readback and dyadic tolerances are displayed before the terminal row T is available to consumers.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no host partition limit, selected refining sequence, quotient function equality, measure-completion row, countable-choice selector, or ambient completeness theorem, the listed rows exhaust the local obligations. □

Theorem 8422.3 (Uniform-partition criterion non-escape)**.** *Every accepted* $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ *consumer that reads a Riemann-integrability claim factors through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RiemannSumCriterion}^{\uparrow},\ \mathsf{ContinuousCompactRiemannIntegrability}^{\uparrow}$$

as displayed by I, R, D, Q, T and by the structural rows H, C, P, N. No consumer may replace that route by a host partition-limit theorem, quotient equality of functions, a selected sequence of refinements, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted consumer read. By the carrier definition, the only terminal handoff is T, and it is downstream of the interval, integrand, mesh, equal-cell, sum, criterion, rational, and dyadic tolerance rows.

Use the criterion row Q to locate the Riemann-sum comparison inside the sibling surface of chapter 8064. The comparison does not stand alone; it reads the same finite uniform cells and sum row.

Read the numerical rows through R and D. These rows keep sum values, mesh bounds, and tolerances in $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ form before any $\mathsf{Real}^{\uparrow}$ terminal seal is consumed.

The terminal consumer may then connect to the compact-continuity route of chapter 7133, but only through the displayed handoff T. The structural rows H, C, P, N preserve the same packet, so no excluded host object is available. □

Falsifiable prediction: An accepted $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ consumer that reaches a Riemann-integrability claim without displaying the uniform mesh, equal cells, finite sum row, Riemann-sum criterion row, regular rational readback, dyadic tolerance row, terminal handoff, transport, replay, provenance, and local naming rows refutes the carrier surface.

Independence witness: real, regseqrat, dyadicratcore, riemann-sum-criterion, continuous-compact-riemann-integrability: no listed sibling is bijective with $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ because this packet binds the uniform mesh, equal-cell ledger, finite Riemann-sum readback, criterion comparison, rational and dyadic tolerance boundary, terminal handoff, transport, replay, provenance, and local naming rows as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}$ packet is generated from a RealUp interval source row, displayed integrand windows, a uniform mesh row, an equal-cell ledger, finite Riemann-sum readback, a RiemannSumCriterion comparison row, RegSeqRat readback, a DyadicRatCore tolerance row, terminal consumer handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8422.1, Definition 8422.2, and Definition 8422.3 seed the finite uniform-partition Riemann-integrability surface over Real, RegSeqRat, DyadicRatCore, RiemannSumCriterion, compact-continuity, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityUniformPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Riemann integration theorem, host partition limits, selected refining sequences, quotient function equality, measure completion, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-mesh exactness, equal-cell endpoint coverage, finite Riemann-sum readback, criterion comparison, rational and dyadic tolerance nonescape, terminal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegrabilityUniformPartitionUp.

Chapter 8423

A Concrete Naming Certificate for $\mathsf{LocatedBisectionLimit}^{\uparrow}$

Following Bishop and Bridges on interval bisection and constructive real approximation, the $\mathsf{LocatedBisectionLimit}^{\uparrow}$ packet names the finite certificate surface by which a retained located bisection route is read as a terminal $\mathsf{Real}^{\uparrow}$ seal. It sits over the source located-interval bisection packet of chapter 1711, the bisection-to-Cauchy handoff of chapter 8207, the dyadic tolerance ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet records a finite route from located bisection data to a Real seal; it does not select an infinite branch, identify quotient streams, assert trichotomy, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8423.1 (Located bisection limit carrier)**.** *A* $\mathsf{LocatedBisectionLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, B, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row, read through the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *surface and carrying the displayed endpoint, radius, locator, and retained-window data for the requested bisection depth.*
2. *B is the retained bisection branch ledger. It records the finite left-or-right branch prefix, the retained child intervals, and the source agreement with I.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and diameter budget row. It stores the displayed midpoint cells, dyadic width bounds, and tolerance comparisons used by the retained branch ledger.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite bisection stages and endpoint approximants inspected by a downstream request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row formed from the retained dyadic centers and the diameter budgets displayed in D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is exposed only after I, B, D, S, R have been displayed and replayed.*
7. *H records componentwise* hsame *transport for the source interval, retained branch ledger, dyadic midpoint and diameter budget, StreamName windows, RegSeqRat readback, terminal Real seal, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite approximation request through the located interval source, retained branch ledger, dyadic budget, StreamName windows, RegSeqRat readback, and terminal Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedBisectionLimit}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{BisectionCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedBisectionLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, D, S, R, E, H, C, P, N$. It has no coordinate for an infinite branch object, quotient stream equality, host real equality, trichotomy, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8423.2 (Located bisection limit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedBisectionLimit}^{\uparrow}$ *carrier*

$$L = (I, B, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedBisectionLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, retained bisection branch ledger, dyadic midpoint and diameter budget, finite stream windows, regular rational readback, terminal Real seal, componentwise transport, replay, provenance, and local naming rows;*

2. *source agreement, requiring the retained branch ledger B to read its initial interval and locator data from I;*

3. *dyadic diameter control, requiring every retained child interval in B to display midpoint and width information through D before any readback row is exposed;*

4. *finite-window extraction, requiring S to list only bisection stages and endpoint approximants already displayed by I, B, D;*

5. *regular-readback handoff, requiring R to read retained centers under the dyadic error budgets supplied by D before the terminal seal E is exposed;*

6. *transport, replay, provenance, and naming stability through H, C, P, N;*

7. *non-escape, excluding selected infinite branches, quotient stream equality, host real equality, trichotomy, countable choice, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the local certificate.*

No NameCert obligation is stored outside $I, B, D, S, R, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 8423.1. The packet displays the located source interval, retained branch ledger, dyadic budget, stream windows, regular readback, terminal seal, and structural rows.

Source agreement is inherited from the located-interval bisection route of chapter 1711. The branch ledger has no independent interval source; it can only retain child intervals below the displayed row I.

Dyadic diameter control follows the bisection-to-Cauchy route of chapter 8207 and the tolerance ledger of chapter 291. Each retained stage displays a midpoint and width before a downstream window may inspect it.

The finite-window and regular-readback obligations are supplied by chapter 126 and chapter 301. The window row selects only displayed stages, and the readback row turns those stages into regular rational data with the same dyadic budget.

Finally the terminal handoff is the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, transported and replayed through H, C, P, N. Since the carrier has no row for the excluded branch, quotient, choice, trichotomy, host-equality, or completeness objects, the listed rows exhaust the local NameCert surface. □

Theorem 8423.3 (Located bisection limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedBisectionLimit}^{\uparrow}$ *carrier must pass through the finite route*

$$I, B, D \longmapsto S, R \longmapsto E, H, C, P, N.$$

The consumer receives located interval data, a retained bisection branch ledger, dyadic midpoint and diameter budgets, finite StreamName windows, RegSeqRat readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row before the terminal Real seal can be consumed. The handoff exports no selected infinite branch, quotient stream equality, host real equality, trichotomy, countable choice, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the source and branch rows. The source row I gives the located interval data, while B displays the finite retained branch ledger below that source. This is the public bisection surface supplied by chapter 1711.

Next read the dyadic budget row D. The midpoint and diameter comparisons in D are the data used by the bisection-Cauchy route of chapter 8207 to control the retained centers.

Then the finite observation and regular readback rows S, R expose only already displayed bisection stages. Their StreamName and RegSeqRat roles are the same finite-window and regular-rational roles described in chapter 126 and chapter 301.

Finally E, H, C, P, N seal and name the route. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after the source, branch, dyadic, window, and readback rows have been replayed, as in chapter 38. No excluded branch, quotient, host-equality, trichotomy, choice, selected-representative, or completeness row is present, so no Real-facing consumer can bypass the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedBisectionLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedBisectionLimit}^{\uparrow}$ packet is generated from a LocatedIntervalBisection source row, a retained bisection branch ledger, DyadicRatCore midpoint and diameter budgets, StreamName finite windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedBisectionLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8423.1, Definition 8423.2, and Definition 8423.3 seed the finite located-bisection route from source interval and dyadic branch rows to regular Cauchy readback and terminal Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedBisectionLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, selected infinite branches, quotient stream equality, trichotomy, countable choice, host real equality, selected representatives, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, dyadic diameter control, finite-window extraction, regular-readback handoff, Real-seal handoff, and non-escape for BEDC.Derived.LocatedBisectionLimitUp.

Chapter 8424

A Concrete Naming Certificate for $\mathsf{TotallyBoundedReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{TotallyBoundedReal}^{\uparrow}$ packet records a finite-net real-object certificate for Real and Completion consumers. It sits beside the real-name seal of chapter 38, the metric source of chapter 57, the finite-net source of chapter 127, the dyadic tolerance ledger of chapter 291, and the compact-metric consumer of chapter 129. The packet names a totally bounded real row only through displayed finite nets, regular rational windows, dyadic radii, and real seals; it is not open-cover compactness, a quotient completion, a selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8424.1 (TotallyBoundedReal carrier)**.** *A* $\mathsf{TotallyBoundedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, M, T, D, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording the endpoint real-name boundary exposed to downstream Real and Completion consumers.*
2. *M is the metric row on the displayed real-name surface, with distance requests, endpoint classifiers, and comparison histories read through the same finite packet.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, including the displayed center bundle, membership rows, coverage witnesses, and mesh ledger.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the metric and finite-net rows for positive radius requests.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that exposes only the regular observation prefixes used to inspect the real names.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows attached to W and D.*
7. *H records componentwise* hsame *transport for the real seal, metric row, finite-net row, dyadic ledger, finite windows, rational readbacks, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a tolerance request through D, then through the finite-net row T, the metric row M, the stream window W, the rational readback Q, and finally the real seal E.*

9. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their componentwise transports. It has no coordinate for open-cover compactness, selected subcovers, quotient real equality, selected limits, trichotomy, countable choice, or ambient Real completeness.

Theorem 8424.2 (TotallyBoundedReal NameCert obligations)**.** *For an accepted* $\mathsf{TotallyBoundedReal}^{\uparrow}$ *carrier $R = (E, M, T, D, W, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the real seal, metric row, totally bounded finite-net row, dyadic tolerance ledger, stream window, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *finite-net exactness, requiring every positive tolerance request to read the center bundle, membership rows, coverage witnesses, and mesh ledger displayed by T;*
3. *real-window exactness, requiring every Real-facing read to pass through the finite observation row W, the regular rational readback row Q, and the dyadic tolerance ledger D before the seal E is consumed;*
4. *metric compatibility, requiring the distance comparisons in M to use the same dyadic radii and real-name windows displayed by D, W, Q, E;*
5. *transport and replay stability, requiring H and C to preserve the ordered route from tolerance request to finite net, metric comparison, rational readback, and real seal;*
6. *provenance non-escape, requiring P and N to name only the displayed rows and to export no open-cover compactness, selected subcover, quotient equality, selected limit, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8424.1. The packet exposes exactly the real seal, metric row, finite-net row, dyadic tolerance ledger, stream window, regular rational readback, transport, replay, provenance, and local name.

Finite-net exactness is read from T. A positive tolerance request can reach only the displayed center bundle, membership rows, coverage witnesses, and mesh ledger carried by the $\mathsf{TotallyBounded}^{\uparrow}$ row.

Real-window exactness and metric compatibility are read together. The stream window W, regular rational readback Q, and dyadic ledger D display the real-name observations before the seal E is exposed, while the metric row M uses those same windows and radii for every comparison.

Finally inspect H, C, P, N. Transport and replay preserve the displayed route, and provenance names only the listed rows. Since the carrier has no coordinate for open-cover compactness, selected subcovers, quotient equality, selected limits, trichotomy, countable choice, or ambient Real completeness, none of those objects can be exported. □

Theorem 8424.3 (TotallyBoundedReal finite-net handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*, Real, or Completion consumer of an accepted* $\mathsf{TotallyBoundedReal}^{\uparrow}$ *carrier*

$$R = (E, M, T, D, W, Q, H, C, P, N)$$

reads the totally bounded real object through the finite route

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For each positive tolerance request, the consumer receives a displayed dyadic radius from D, a finite center bundle from T, metric comparisons from M, a regular rational window from W, Q, and the terminal real seal E. The handoff exports no open-cover compactness, selected subcover, quotient real equality, selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the tolerance request. The replay row C sends it first to the dyadic ledger D, so every radius and comparison budget is displayed before a finite-net row can be read.

Next read the finite-net row T and the metric row M. The center bundle, membership rows, coverage witnesses, and mesh ledger provide the total-bounded source, while the metric comparisons use exactly the dyadic radii already displayed by D.

The Real-facing portion then passes through the finite window W and the regular rational readback Q before reaching the seal E. Thus a downstream compact-metric, Real, or Completion consumer sees a real object only after the finite net and the rational-window evidence have been exposed.

Transport, replay, provenance, and local naming through H, C, P, N preserve that route. The packet therefore hands off finite-net real data without adding any open-cover compactness, selected subcover, quotient real equality, selected limit, trichotomy, countable choice, or ambient Real completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8424.1, Definition 8424.2, and Definition 8424.3 seed the finite-net real-object certificate over RealUp, MetricUp, TotallyBoundedUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, CompactMetricUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exactness, real-window exactness, metric compatibility, transport stability, replay stability, and provenance non-escape for BEDC.Derived.TotallyBoundedRealUp.

Constructive story: A $\mathsf{TotallyBoundedReal}^{\uparrow}$ packet is generated from a Real seal, metric row, totally bounded finite-net row, dyadic tolerance ledger, finite stream window, regular rational readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 8425

A Concrete Naming Certificate for $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$

Following Bishop and Bridges on constructive sequential compactness, the $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$ packet records the finite-window subsequence route that can be consumed before any full compactness or completeness theorem is available. It sits beside the finite source interfaces of chapter 5916 and chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal seal of chapter 38, and the real Bolzano–Weierstrass and sequential compactness routes of chapter 5350 and chapter 5935. The packet names a bounded finite source window and a displayed subsequence schedule; it is not a host subsequence selector, countable-choice principle, quotient stream equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 8425.1 (Finite subsequence compactness carrier)**.** *A* $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, L, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite source-window row, listing the bounded block of source observations available to the compactness consumer.*
2. *S is the subsequence schedule row, recording only the displayed monotone index cells selected inside F.*
3. *L is the compact or located source row, exposing the finite compact support, located interval, or finite-net witness from which the schedule is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic mesh, comparison radius, and readback precision used by R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, S, L, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the source window, subsequence schedule, compact source, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite compactness request through F, S, L, R, D, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubsequenceCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, L, R, D, E, H, C, P, N$. It has no coordinate for a host-selected infinite subsequence, quotient stream equality, countable choice, open-cover compactness, or ambient real completeness.

Theorem 8425.2 (Finite subsequence compactness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$ *carrier*

$$K = (F, S, L, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSubsequenceCompactness}^{\uparrow}}$ *surface consists exactly of finite source-window admission through F, subsequence schedule visibility through S, compact or located source exposure through L,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, dyadic tolerance accounting through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8425.1. The only rows available to a compactness consumer are $F, S, L, R, D, E, H, C, P, N$.

The source and schedule rows are finite. Row F exposes the inspected source block, row S records the monotone schedule inside that block, and row L supplies the displayed compact or located support before any readback is public.

The analytic read proceeds through R, D, E. The regular rational row reads only scheduled observations, the dyadic row fixes the tolerance ledger, and the Real seal is downstream from both.

Finally H, C, P, N preserve and name the same route. Since the carrier has no coordinate for host subsequence selection, quotient stream equality, countable choice, open-cover compactness, or ambient real completeness, the listed rows exhaust the local obligation surface. □

Theorem 8425.3 (Finite subsequence Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{FiniteSubsequenceCompactness}^{\uparrow}$ *carrier must pass through the finite source window F, subsequence schedule S, compact or located source row L,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D before the terminal seal E can be consumed. The route exports a displayed finite-window subsequence packet and no selected infinite subsequence, quotient stream equality, countable choice, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the obligation surface of Definition 8425.2. It places F, S, L before any regular rational or real-facing read.

The readback rows R and D then account for the scheduled rational observations and the dyadic tolerance used to compare them. Therefore a Real-facing consumer cannot reach E without first reading the displayed finite source and schedule route.

The structural rows H, C, P, N transport, replay, source, and name that same route. They add no hidden selector, quotient stream, choice principle, open-cover compactness theorem, or ambient completeness object, so Real-seal access remains finite-window local. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubsequenceCompactness}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSubsequenceCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8425.1, Definition 8425.2, and Definition 8425.3 seed the finite-window subsequence compactness route through finite source windows, displayed schedules, compact or located support, RegSeqRat readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteSubsequenceCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subsequence selection, quotient stream equality, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite source-window visibility, subsequence-schedule monotonicity, compact-source exposure, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubsequenceCompactnessUp.

Constructive story: A FiniteSubsequenceCompactness$^{\uparrow}$ packet is generated from a finite source window, a displayed subsequence schedule, compact or located source support, RegSeqRat readback, DyadicRatCore tolerance, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8426

A Concrete Naming Certificate for $\mathsf{FiniteWindowSubcover}^{\uparrow}$

Following Bishop finite subcover and finite-net arguments, the $\mathsf{FiniteWindowSubcover}^{\uparrow}$ packet records the Real-completeness subcover route as explicit windows rather than as an ambient Heine–Borel theorem. It realizes the route of chapter 9238 through displayed $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. The packet is a finite-window NameCert surface for later completion consumers; it does not assert general open-cover compactness, arbitrary subcover selection, countable choice, quotient stream equality, or full ambient Real completeness. The sibling route through chapter 5916, chapter 126, chapter 301, chapter 291, and chapter 38 supplies the finite candidate stream, observation windows, regular rational readback, dyadic tolerance ledger, and terminal Real seal consumed by this packet.

Definition 8426.1 (Finite-window subcover carrier)**.** *A* $\mathsf{FiniteWindowSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the displayed cover-request row, naming the finite family of cover cells and the requested observation budget.*
2. *F is the* $\mathsf{FiniteStream}^{\uparrow}$ *source row, exposing the finite list of candidate cover cells and sampled source points.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source observations and cover-membership reads are opened.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational endpoints, radii, or membership thresholds visible in S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the finite mesh, overlap, and radius comparisons.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the cover request, finite stream, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real-completeness finite-window request through U, F, S, R, D, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowSubcover}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowSubcover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, S, R, D, E, H, C, P, N$. It has no coordinate for a host open-cover object, arbitrary subcover selector, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Theorem 8426.2 (Finite-window subcover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowSubcover}^{\uparrow}$ *carrier*

$$B = (U, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowSubcover}^{\uparrow}}$ *surface consists exactly of cover-request visibility through U, finite source and candidate-cell exposure through F,* $\mathsf{StreamName}^{\uparrow}$ *window reads through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, dyadic tolerance accounting through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8426.1. The packet exposes only $U, F, S, R, D, E, H, C, P, N$.

The cover-facing data are finite. Row U fixes the request, row F lists the displayed candidates, and row S opens only the inspected StreamName windows.

The numerical readback stays within R, D, E. The regular rational row reads the displayed endpoints or thresholds, the dyadic row supplies the finite tolerance ledger, and the Real seal is downstream from those rows.

Transport, replay, provenance, and local naming preserve this route without adding a host open-cover object, arbitrary subcover selector, countable-choice witness, quotient real equality, or ambient Real-completeness theorem. □

Theorem 8426.3 (Heine–Borel finite-window subcover route)**.** *Every subcover read exported by an accepted* $\mathsf{FiniteWindowSubcover}^{\uparrow}$ *carrier stays inside the displayed finite windows $U, F, S, R, D, E, H, C, P, N$. In particular, cover-cell membership is read through* $\mathsf{FiniteStream}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*, rational thresholds are read through* $\mathsf{RegSeqRat}^{\uparrow}$*, tolerance comparisons are read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal value is sealed through* $\mathsf{Real}^{\uparrow}$*. The route does not assert general open-cover compactness, arbitrary subcover selection, countable choice, quotient stream equality, or ambient Real completeness.*

Proof. Begin with Definition 8426.2. It forces the cover request, finite candidate list, and source observation windows to be displayed before any subcover-facing read is public.

Next read the numerical rows. The rational thresholds in R and dyadic comparisons in D are attached to the same finite StreamName windows, so no cover membership can be imported from outside U, F, S.

Finally consume the Real seal E only after the finite readback route has been displayed. The support rows H, C, P, N preserve that route uniformly, and the carrier contains no coordinate for the excluded open-cover, selector, choice, quotient, or ambient-completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowSubcover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8426.1, Definition 8426.2, and Definition 8426.3 seed the finite-window subcover route through cover requests, FiniteStream candidates, StreamName windows, RegSeqRat readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, arbitrary subcover selection, countable choice, quotient stream equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite candidate visibility, StreamName window exactness, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowSubcoverUp.

Constructive story: A FiniteWindowSubcover$^{\uparrow}$ packet is generated from a cover request, a FiniteStream candidate list, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8427

A Concrete Naming Certificate for $\mathsf{WhitneyExtension}^{\uparrow}$

Following Whitney's extension theorem for compatible finite jets on closed subsets of Euclidean space, the $\mathsf{WhitneyExtension}^{\uparrow}$ packet records the finite BEDC handoff from a displayed jet surface to extension-facing consumers. It sits after the finite-jet compatibility surface of chapter 3679, the finite Taylor enclosure rows of chapter 442, and the manifold carrier of chapter 96. The packet names only the rows by which a finite jet table, remainder ledger, chart window, transport, replay, provenance, and local naming are made available to downstream derivative or smooth-manifold consumers; it is not a host theorem selecting an arbitrary global smooth extension.

Definition 8427.1 (Whitney extension carrier)**.** *A* $\mathsf{WhitneyExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (J, T, M, E, R, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *compatibility row, including the closed-set classifier, finite jet table, and Whitney comparison ledger already exposed by the finite-jet chapter.*
2. *T is the* $\mathsf{TaylorModel}^{\uparrow}$ *local Taylor-row handoff, retaining only displayed center, finite coefficient, and remainder-enclosure data needed to read local polynomial approximants.*
3. *M is the* $\mathsf{Manifold}^{\uparrow}$ *or Euclidean chart-window row on which the finite jet and Taylor rows are interpreted.*
4. *E is the extension-handoff ledger, recording which jet requests are eligible for derivative-facing or smooth-manifold consumers after J, T, M have been displayed.*
5. *R is the readback row for derivative, smooth-chart, and validated-local-model consumers.*
6. *H records componentwise* hsame *transport for J, T, M, E, R, C, P, N.*
7. *C records displayed* Cont *replay from a finite extension request through J, T, M, E, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{WhitneyExtension}^{\uparrow}$, $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WhitneyExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J, T, M, E, R, H, C, P, N$. It has no coordinate for a selected smooth function, arbitrary global extension operator, compactness theorem, quotient equality, choice principle, or ambient completeness claim.

Theorem 8427.2 (Whitney extension NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyExtension}^{\uparrow}$ *carrier* $W = (J, T, M, E, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WhitneyExtension}^{\uparrow}}$ *surface consists exactly of carrier admission for the rows named in Definition 8427.1, finite-jet compatibility inherited through* J*, Taylor-row handoff through* T*, chart-window exposure through* M*, extension-handoff eligibility through* E*, derivative or smooth-chart readback through* R*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The only analytic source is the finite Whitney jet row J, and the only local approximation source is the Taylor-model row T over the displayed chart window M.

The extension-handoff row E is downstream of those three sources. It records that a consumer may request the displayed extension-facing data only after the finite compatibility, Taylor remainder, and chart rows are visible in the same packet.

The readback row R exposes exactly that handoff to derivative or smooth-chart consumers. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same route and add no hidden extension object. □

Theorem 8427.3 (Whitney extension jet handoff)**.** *Every derivative-facing, validated-local-model, or smooth-chart consumer reading an accepted* $\mathsf{WhitneyExtension}^{\uparrow}$ *packet must factor through the displayed route*

$$\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}, \mathsf{TaylorModel}^{\uparrow}, \mathsf{Manifold}^{\uparrow} \longmapsto E \longmapsto R \longmapsto H, C, P, N.$$

The consumer receives the finite jet compatibility surface, Taylor-row local approximation data, chart-window interpretation, and extension-handoff ledger. It receives no arbitrary global smooth extension, no selected representative function, no compactness theorem, no quotient equality, no cover choice, and no ambient completeness principle.

Proof. Begin with a consumer read. By Definition 8427.1, the consumer can inspect only J, T, M before reaching the extension-handoff ledger E.

Read E next. It is a finite row stating that the displayed jet and Taylor data can be handed to the requested derivative-facing or smooth-chart endpoint; it does not contain a host extension function.

Finally pass through R, H, C, P, N. These rows expose, transport, replay, source, and name the same finite handoff. Since the carrier has no coordinate for a global smooth extension or selected representative, every consumer route remains inside the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyExtension}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WhitneyExtension}^{\uparrow}}$, seed)

Scope closed: Definition 8427.1, Definition 8427.2, and Definition 8427.3 seed the finite Whitney extension handoff over finite jets, Taylor rows, manifold chart windows, readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WhitneyExtension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the classical global Whitney extension theorem, a selected smooth extension operator, compactness, arbitrary choice, quotient equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-jet compatibility, Taylor-row handoff, chart-window exposure, extension-handoff eligibility, readback, transport, replay, provenance, and local naming for BEDC.Derived.WhitneyExtensionUp.

Constructive story: A $\mathsf{WhitneyExtension}^{\uparrow}$ packet is generated from a WhitneyExtensionFiniteJetUp compatibility row, a TaylorModelUp local Taylor row, a ManifoldUp chart window, an extension-handoff ledger, derivative or smooth-chart readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8428

A Concrete Naming Certificate for $\mathsf{StratifiableSpace}^{\uparrow}$

Following Borges and Engelking on stratifiable spaces, the $\mathsf{StratifiableSpace}^{\uparrow}$ packet records the finite neighborhood-stratum route by which closed-set data are turned into open refinements visible to topology consumers. It sits beside the topological carrier of chapter 88, the metric-refinement surface of chapter 57, the open-cover carrier of chapter 6731, and the locally finite cover route of chapter 8398. The certificate names only the displayed stratum rows, monotone refinement comparisons, and downstream cover handoff; it does not import a host metrization theorem, paracompactness theorem, selected point neighborhoods, countable choice, or ambient lattice equality.

Definition 8428.1 ($\mathsf{StratifiableSpace}^{\uparrow}$ neighborhood-stratum carrier)**.** *A* $\mathsf{StratifiableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, F, G, M, U, R, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Topology}^{\uparrow}$ *source row, including the finite open-set tests used by the packet.*
2. *F is the closed-request row family. Each entry names the closed row whose neighborhoods are to be stratified.*
3. *G is the stratum ledger. For every displayed closed row and finite index, it records an open neighborhood row $G(F, n)$.*
4. *M is the monotonicity ledger, recording the displayed implication from $F_0 \subseteq F_1$ to $G(F_0, n) \subseteq G(F_1, n)$ and the nesting comparison from $G(F, n+1)$ to $G(F, n)$.*
5. *U is the separation ledger. It records that each closed row F remains inside the corresponding open stratum and that exterior witnesses are kept outside some finite stratum window.*
6. *R is the refinement handoff row for open-cover consumers. It exposes the finite cover rows obtained from the visible strata.*
7. *L is the locally finite route row, read through chapter 8398 only after G, M, U, R have been displayed.*
8. *H records componentwise* hsame *transport for $X, F, G, M, U, R, L, C, P, N$.*
9. *C records displayed* Cont *replay for topology admission, closed-row admission, stratum reads, monotone comparison, separation readback, refinement export, locally finite handoff, and the local naming row.*

10. *P is the* Pkg *provenance over* StratifiableSpace$^{\uparrow}$, Topology$^{\uparrow}$, Met$^{\uparrow}$, OpenCover$^{\uparrow}$, ParacompactLocallyFinite$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{StratifiableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, G, M, U, R, L, H, C, P, N$. It has no coordinate for a host metrization theorem, an ambient paracompactness theorem, arbitrary neighborhood choice, or equality of open-set lattices outside the displayed finite rows.

Theorem 8428.2 (StratifiableSpace$^{\uparrow}$ NameCert obligations)**.** *For every accepted* StratifiableSpace$^{\uparrow}$ *carrier*

$$S = (X, F, G, M, U, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StratifiableSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission through X, closed-request visibility through F, open-stratum exposure through G, monotone and nesting comparison through M, separation readback through U, cover-refinement export through R, locally finite handoff through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8428.1. The displayed rows expose the topological source, closed requests, strata, monotone comparisons, separation witnesses, refinement handoff, locally finite route, transport, replay, provenance, and local name.

The stratum obligations are local to F, G, M, U. The closed row is visible before any open neighborhood is read, the stratum ledger supplies the indexed open rows, and the monotonicity ledger records the two comparisons needed by downstream cover consumers.

The cover-facing data are exported only after the stratum route has been displayed. Rows R and L hand the finite refinement surface to open-cover and locally finite consumers, while H, C, P, N preserve the same finite packet without importing a host metrization theorem, arbitrary choices of neighborhoods, or ambient equality of topological lattices. □

Theorem 8428.3 (Metric-topology refinement handoff)**.** *Let $S = (X, F, G, M, U, R, L, H, C, P, N)$ be an accepted* StratifiableSpace$^{\uparrow}$ *carrier. Every metric-style neighborhood refinement read exported by S factors through the finite route*

$$X, F, G, M, U, H, C, P, N \longmapsto R \longmapsto L.$$

In particular, a downstream topology or cover-refinement consumer may read the displayed closed rows, open strata, monotone comparisons, separation windows, and refinement handoff, but it cannot replace them by a host metrization theorem, an arbitrary open-cover selector, or a paracompactness theorem outside the packet.

Proof. Start with the topology source row X and the closed-request rows F. They fix the only closed sets whose neighborhoods the packet is allowed to expose.

Read the stratum data next. The rows G, M, U display the open neighborhoods, the monotone and nesting comparisons, and the finite separation windows, so every refinement read has a visible stratum witness.

Finally pass through R and L. The refinement row exports the finite cover data, and the locally finite row is reached only through the displayed refinement route. Componentwise transport, replay, provenance, and local naming keep the handoff inside the same BHist packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StratifiableSpace}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StratifiableSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8428.1, Definition 8428.2, and Definition 8428.3 seed the finite neighborhood-stratum route through topology source rows, closed requests, indexed open strata, monotone comparisons, separation windows, cover refinement, locally finite handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{StratifiableSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close metrization, paracompactness, arbitrary neighborhood selection, equality of ambient open-set lattices, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-request visibility, open-stratum exposure, monotone comparison, separation readback, cover-refinement export, locally finite handoff, transport, replay, provenance, and local naming for BEDC.Derived.StratifiableSpaceUp.

Constructive story: A StratifiableSpace$^{\uparrow}$ packet is generated from a topology source, closed-request rows, indexed open-stratum rows, monotone and nesting comparison rows, separation rows, a cover-refinement handoff, a locally finite route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8429

A Concrete Naming Certificate for FiniteCauchyCompletenessCriterion$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the FiniteCauchyCompletenessCriterion$^{\uparrow}$ packet records a finite criterion object for reading Cauchy data into the Real$^{\uparrow}$ seal boundary without asserting ambient completeness. It sits after the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the convergence-route surface of chapter 1509, the Bolzano–Cauchy branch of chapter 1913, and the terminal real-name seal of chapter 38. The certificate names the finite rows by which a completion-facing consumer may inspect Cauchy convergence data; it is not a quotient of streams, a selected limit, countable choice, trichotomy, open-cover compactness, or full Real$^{\uparrow}$ completeness.

Definition 8429.1 (Finite Cauchy completeness criterion carrier)**.** *A* FiniteCauchyCompletenessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, G, B, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *finite-window schedule naming exactly the observed Cauchy tail requests.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced from those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger recording the dyadic radii, comparison cells, and finite precision budget.*
4. *G is the* CauchyConvergenceCriterion$^{\uparrow}$ *criterion row that certifies convergence readiness for the displayed readback and tolerance data.*
5. *B is the* BolzanoCauchy$^{\uparrow}$ *branch row, available only as a finite companion route for located Cauchy evidence.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after S, R, D, G, B have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, criterion, Bolzano–Cauchy branch, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each finite tail request through S, R, D, G, B, E, H.*
9. *P is the* Pkg *provenance over* FiniteCauchyCompletenessCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyConvergenceCriterion$^{\uparrow}$, BolzanoCauchy$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for host Cauchy completeness, quotient stream equality, selected representatives, countable choice, trichotomy, open-cover compactness, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 (statement) `BEDC.Derived.FiniteCauchyCompletenessCriterionUp`

Theorem 8429.2 (Finite Cauchy completeness criterion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}$ *carrier* $K = (S, R, D, G, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyC}}$ *surface consists exactly of carrier admission for the finite StreamName window, regular rational readback, dyadic tolerance ledger, convergence-criterion row, Bolzano–Cauchy branch row, Real seal row, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8429.1. Its displayed coordinates are precisely $S, R, D, G, B, E, H, C, P, N$.

The analytic read is finite before it is real-facing. Row S fixes the observed tail window, R supplies only the corresponding regular rational readback, and D supplies the dyadic tolerance cells attached to that window.

The criterion and branch rows are downstream of the same finite data. The convergence-criterion row G and Bolzano–Cauchy row B may be read only after S, R, D have been exposed, and the Real seal E is downstream from those rows.

Transport, replay, provenance, and local naming preserve this route. Since the carrier has no coordinate for host completeness, quotient streams, selected representatives, countable choice, trichotomy, open-cover compactness, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8429.3 (Finite Cauchy completeness criterion Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or completion-facing read of an accepted* $\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyConvergenceCriterion}^{\uparrow} \longmapsto \mathsf{BolzanoCauchy}^{\uparrow} \longmapsto \mathsf{Real}$$

The read consumes the same window row S, readback row R, tolerance ledger D, criterion row G, branch row B, seal row E, transport row H, replay row C, provenance row P, and local name row N before any terminal Real seal is public.

Proof. Begin with the finite window. The carrier gives S as the only source of Cauchy tail observations, so a Real-facing consumer cannot inspect an unbounded stream before the finite schedule is displayed.

Next read R and D. The regular rational row supplies the displayed values, and the dyadic ledger supplies the radii and comparison cells used by the criterion.

Now consume the criterion and branch rows. The convergence row G records readiness for the displayed Cauchy data, while B supplies only the finite Bolzano–Cauchy companion route; neither row introduces a selected limit or an ambient completeness theorem.

Finally the seal row E is reached through H, C, P, N. These structural rows preserve the same finite route, so the public Real seal has no bypass around S, R, D, G, B. □

Theorem 8429.4 (Finite Cauchy completeness criterion nonescape)**.** *An accepted* $\mathsf{FiniteCauchyCompletenessCriteric}$ *packet exports no ambient completeness principle. Any consumer that reaches* $\mathsf{Real}^{\uparrow}$ *through the packet can read only the finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, Cauchy convergence criterion row, Bolzano–Cauchy branch row, Real seal boundary, transport, replay, provenance, and local NameCert rows displayed by the carrier.*

Proof. Apply the NameCert obligations of Definition 8429.2. They identify the complete local surface of the accepted packet.

Apply the Real seal route of Definition 8429.3. Every completion-facing read reaches $\mathsf{Real}^{\uparrow}$ only through $S, R, D, G, B, E, H, C, P, N$.

The carrier definition contains no further coordinate. Therefore a consumer cannot extract quotient stream equality, a selected limit, countable choice, trichotomy, open-cover compactness, host Cauchy completeness, or ambient Real completeness from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a Cauchy convergence criterion row, a Bolzano–Cauchy branch row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8429.1, Definition 8429.2, Definition 8429.3, and Definition 8429.4 seed the finite criterion route from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Cauchy convergence evidence, Bolzano–Cauchy branch data, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyCompletenessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FiniteCauchyCompletenessCriterionUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, host Cauchy completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, regular rational readback, dyadic tolerance accounting, convergence-criterion handoff, Bolzano–Cauchy branch routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyCompletenessCriterionUp.

Chapter 8430

A Concrete Naming Certificate for $\mathsf{FiniteNetFan}^{\uparrow}$

Following Brouwer and Bishop–Bridges finite-net arguments, the $\mathsf{FiniteNetFan}^{\uparrow}$ packet records a finite ProbeBundle fan handoff for total-boundedness and compact-uniform consumers. It sits between the finite-net source of chapter 127, the compact-metric consumer of chapter 129, the dyadic tolerance ledger of chapter 291, the finite stream-name window of chapter 126, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a finite fan of mesh-controlled probes; it is not open-cover compactness, selected subcover extraction, quotient real equality, countable choice, trichotomy, or ambient Real completeness.

Definition 8430.1 (FiniteNetFan carrier)**.** *A* $\mathsf{FiniteNetFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, D, J, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ProbeBundle}^{\uparrow}$ *center bundle, recording the finite family of probe centers, membership rows, and bundle indices used by the net.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-radius ledger, recording the positive dyadic tolerance and every comparison budget consumed by the fan.*
3. *J is the finite fan-refinement row, listing the bounded refinement branches, parent-child mesh inclusions, and coverage witnesses read from the finite net.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which each probe center and refinement branch exposes only the finite observations requested by D.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows attached to W and to the dyadic mesh ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the center and branch observations after B, D, J, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the center bundle, mesh radius, fan refinement, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through D, then through B, J, W, Q, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteNetFan}^{\uparrow}$, $\mathsf{ProbeBundle}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteNetFan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their componentwise transports. It has no coordinate for open-cover compactness, a selected subcover, quotient real equality, a hidden completion theorem, countable choice, trichotomy, or ambient Real completeness.

Theorem 8430.2 (FiniteNetFan NameCert obligations)**.** *For an accepted* $\mathsf{FiniteNetFan}^{\uparrow}$ *carrier* $F = (B, D, J, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteNetFan}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, center-bundle visibility through B, mesh-radius accounting through D, finite fan-refinement visibility through J, StreamName window exactness through W, RegSeqRat readback through Q, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 8430.1. The packet exposes exactly the center bundle, mesh-radius ledger, finite fan-refinement row, window row, readback row, Real seal, transport, replay, provenance, and local name.

The finite-net obligations are read from B, D, J. The bundle B displays the centers, D displays the mesh radius and comparison budget, and J displays the bounded refinement branches and coverage witnesses.

The Real-facing obligations are read from W, Q, E. StreamName windows expose the finite observations, RegSeqRat readback turns those windows into regular rational rows, and the Real seal is downstream of that readback.

Finally H, C, P, N preserve and name the route. Since no carrier coordinate names open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, or ambient Real completeness, none of those data can be exported by the local certificate. □

Theorem 8430.3 (FiniteNetFan ProbeBundle handoff)**.** *Every total-boundedness, compact-metric, Real, or compact-uniform consumer of an accepted* $\mathsf{FiniteNetFan}^{\uparrow}$ *carrier*

$$F = (B, D, J, W, Q, E, H, C, P, N)$$

reads a finite fan only through the route

$$\mathsf{ProbeBundle}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

For each positive mesh request, the consumer receives the displayed probe centers from B, the dyadic mesh radius from D, the finite refinement branches from J, the finite observation windows from W, the regular rational readback from Q, and the terminal seal from E.

Proof. Induct over the finite ProbeBundle fan recorded by B and J. The empty branch has no exported center, and the carrier still exposes the mesh ledger D, structural rows, and local name before a consumer can read the packet.

For a branch extension, project the next probe center from B and the next refinement row from J. The mesh comparison for that branch is read from D, so the branch cannot be consumed as a detached point selection.

The branch then passes through W, Q, E. The StreamName window supplies only the finite observations requested by the mesh ledger, RegSeqRat readback records the regular rational values, and the Real seal is terminal for that branch.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same route for every branch. Thus total-boundedness and compact-uniform consumers receive finite ProbeBundle fan data without importing open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetFan}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetFan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8430.1, Definition 8430.2, and Definition 8430.3 seed the finite ProbeBundle fan handoff through center bundles, dyadic mesh radii, finite fan refinements, StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetFan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ProbeBundle center visibility, dyadic mesh accounting, finite fan-refinement exactness, StreamName window exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteNetFanUp.

Constructive story: A $\mathsf{FiniteNetFan}^{\uparrow}$ packet is generated from a ProbeBundle center bundle, dyadic mesh-radius ledger, finite fan-refinement row, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 8431

A Concrete Naming Certificate for $\mathsf{BorelLebesgueCover}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Borel–Lebesgue finite-cover theorem for closed intervals, the $\mathsf{BorelLebesgueCover}^{\uparrow}$ packet records the finite subcover boundary as BEDC NameCert data. It sits after the interval compactness surface of chapter 1379, the closed-bounded interval net of chapter 1250, the located interval-cover row of chapter 1885, and the rational readback rows of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names the finite subcover certificate available for a displayed interval cover; it does not import general open-cover compactness, selected subcovers, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8431.1 (Borel–Lebesgue cover carrier)**.** *A* $\mathsf{BorelLebesgueCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, K, L, D, S, Q, R, F, H, C, P, N)$$

where:

- I *is the displayed* $\mathsf{HeineBorelInterval}^{\uparrow}$ *closed-interval source.*
- K *is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite interval-net row over the same endpoints.*
- L *is the* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row whose located cover tests are available on the displayed interval net.*
- D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare interval endpoints, cover radii, and finite-net refinements.*
- S *is the* $\mathsf{StreamName}^{\uparrow}$ *inspection schedule for the finite cover windows.*
- Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same schedule and dyadic ledger.*
- R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the interval endpoints and cover radii read through* D, S, Q*.*
- F *is the finite subcover-boundary row: it records that the displayed finite window covers the interval source through* I, K, L, D, S, Q, R*.*
- H, C, P, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BorelLebesgueCover}^{\uparrow}}$ *rows.*

The carrier exports no arbitrary open-cover compactness principle, no selected subcover outside F*, no quotient real equality, and no ambient completeness field for* $\mathsf{Real}^{\uparrow}$*.*

Theorem 8431.2 (Borel–Lebesgue cover NameCert obligations)**.** *For the carrier of Definition 8431.1, the local* $\mathsf{NameCert}_{\mathsf{BorelLebesgueCover}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, interval-source exposure through* I*, closed-bounded net exposure through* K*, located-cover exposure through* L*, dyadic and regular readback through* D, S, Q, R*, finite subcover-boundary exposure through* F*, and structural replay through* H, C, P, N*. No NameCert read may replace those rows by general open-cover compactness, a choice-selected subcover, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. Its displayed coordinates are precisely $I, K, L, D, S, Q, R, F, H, C, P, N$, with the interval source, interval-net row, located-cover row, rational readback rows, finite subcover boundary, and structural rows all present before the local name is read.

The NameCert obligation for interval exposure reads I, K, L together: the closed interval, finite net, and located cover are not separate ambient principles. The rational obligation then routes endpoint and radius comparisons through D, S, Q, R.

Finally F, H, C, P, N record the finite subcover boundary, transport, replay, provenance, and local naming surface. Since no coordinate stores arbitrary open-cover compactness, a selected subcover oracle, quotient real equality, or ambient Real completeness, the local NameCert surface cannot export them. □

Theorem 8431.3 (Borel–Lebesgue cover finite-subcover boundary)**.** *Every consumer of an accepted* $\mathsf{BorelLebesgueCover}^{\uparrow}$ *packet receives a finite subcover boundary only after the rows*

$$\mathsf{HeineBorelInterval}^{\uparrow},\quad \mathsf{ClosedBoundedIntervalNet}^{\uparrow},\quad \mathsf{LocatedIntervalCover}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{R}$$

have been displayed inside the same carrier and replayed through F, H, C, P, N*. The exported finite boundary is therefore a packet-local certificate over the displayed interval cover, not a claim of general open-cover compactness, selected subcovers, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and NameCert obligations. They place the interval source I, interval-net row K, located-cover row L, dyadic ledger D, inspection schedule S, regular readback Q, and Real seal R before the finite boundary row F.

The finite boundary row records coverage only for the displayed window over that source. Transport and replay through H and C preserve the same rows, while P and N record provenance and local naming for the same packet.

Thus a downstream compact-uniform-continuity or interval-cover consumer can read the finite subcover boundary only through the displayed packet. The excluded compactness, selected-subcover, quotient-equality, and completeness claims have no carrier coordinate. □

Falsifiable prediction: A $\mathsf{BorelLebesgueCover}^{\uparrow}$ consumer is refuted if it exports a finite subcover without displaying the Heine–Borel interval source, closed-bounded interval net, located interval cover, dyadic ledger, stream window, regular readback, Real seal, finite boundary row, transport, replay, provenance, and local naming rows.

Independence witness: heineborelinterval, closedboundedintervalnet, locatedintervalcover, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{BorelLebesgueCover}^{\uparrow}$, because this packet binds the displayed interval source, located cover, rational readback, and finite subcover boundary as one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelLebesgueCover}^{\uparrow}}$**.**

Constructive story: A BorelLebesgueCover$^{\uparrow}$ packet is generated from a Heine–Borel interval source, a closed-bounded interval net, a located interval-cover row, dyadic tolerance arithmetic, stream-window inspection, regular-rational readback, a Real seal, a finite subcover-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BorelLebesgueCover}^{\uparrow}}$, seed)

Scope closed: Definition 8431.1, Definition 8431.2, and Definition 8431.3 introduce the finite subcover packet boundary over HeineBorelInterval$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, LocatedIntervalCover$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{BorelLebesgueCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general open-cover compactness, selected subcovers, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located cover exposure, dyadic and regular readback, finite subcover-boundary exactness, and non-escape from general compactness or selected-subcover principles.

Chapter 8432

A Concrete Naming Certificate for $\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}$

Following Bishop-style constructive analysis, the $\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}$ packet records finite oscillation control on explicit rational windows before any compact-continuity argument is allowed to consume a Real-facing value. It sits over the dyadic tolerance ledger of chapter 291, the observation interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name surface of chapter 38, and the finite-window consumer route of chapter 8426. The packet names a local gauge for explicit interval windows; it does not assert open-cover compactness, trichotomy, quotient equality, selected suprema, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8432.1 (Constructive real oscillation gauge carrier)**.** *A* $\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, L, U, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval-window row. It names a displayed rational interval, its finite subwindows, and the requested local observation budget.*
2. *L is the local lower-oscillation row. It records the displayed lower rational bounds for the values visible on each window of I.*
3. *U is the local upper-oscillation row. It records the displayed upper rational bounds for the same windows, with no appeal to a selected supremum.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic tolerance, mesh, and comparison ledger used to compare the displayed bounds in L and U.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row through which the finite windows and sampled value names are opened.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row. It records the regular rational readback attached to the sampled values and the dyadic tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, L, U, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval window, lower and upper oscillation bounds, dyadic tolerance, StreamName readback, RegSeqRat regularity, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite oscillation request through I, L, U, D, S, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteWindowSubcover}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, U, D, S, R, E, H, C, P, N$. It has no coordinate for an ambient open cover, trichotomy on reals, quotient equality of names, selected supremum or infimum, arbitrary subcover selection, or full $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Theorem 8432.2 (Constructive real oscillation gauge NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}$ *carrier*

$$G = (I, L, U, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRealOscillationGauge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval-window, local lower-bound, local upper-bound, dyadic tolerance, StreamName readback, RegSeqRat regularity, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *window coherence, requiring L and U to name bounds only for the subwindows displayed by I;*
3. *dyadic oscillation accounting, requiring D to display the tolerance comparisons that make each local spread $U - L$ visible as dyadic data;*
4. *regular readback, requiring S and R to expose only the finite sampled names and regular rational approximants attached to the displayed windows;*
5. *Real-seal nonescape, requiring E to be consumed only after the window, bound, tolerance, StreamName, and RegSeqRat rows have been replayed;*
6. *transport, replay, provenance, and naming stability through H, C, P, N;*
7. *refusal of open-cover compactness, trichotomy, quotient equality, selected suprema, arbitrary subcover selection, and full* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No NameCert obligation is stored outside $I, L, U, D, S, R, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 8432.1. The packet displays exactly the interval-window, lower-bound, upper-bound, dyadic tolerance, StreamName readback, RegSeqRat regularity, Real seal, and structural rows.

Window coherence follows because the lower and upper rows are indexed by the displayed subwindows in I. There is no independent global cover or ambient compactness row from which additional windows could be imported.

Dyadic oscillation accounting is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger of chapter 291. The spread between the displayed upper and lower rational bounds is public only when the tolerance comparisons are recorded in D.

The finite readback route is supplied by chapter 126 and chapter 301. Row S opens the sampled names, and row R records the regular rational approximants attached to those names.

Finally E, H, C, P, N seal and name the route as a local certificate. The carrier has no row for trichotomy, quotient equality, selected suprema, arbitrary subcover selection, or full Real completeness, so those principles cannot enter the local NameCert surface. □

Theorem 8432.3 (Constructive real oscillation modulus extraction)**.** *Every modulus read exported by an accepted* ConstructiveRealOscillationGauge$^{\uparrow}$ *carrier is extracted from the finite route*

$$I, L, U, D \longmapsto S, R \longmapsto E, H, C, P, N.$$

Given a requested dyadic tolerance already displayed in D, the extracted local modulus names only those subwindows of I whose lower and upper rational rows L, U have StreamName readback through S, RegSeqRat regularity through R, and terminal Real sealing through E. The extraction exports no open-cover compactness principle, trichotomy decision, quotient equality, selected supremum, arbitrary subcover selector, or full Real$^{\uparrow}$ *completeness proof.*

Proof. Start with the displayed request in I and the bound rows L, U. These rows determine the candidate subwindows and the rational lower and upper oscillation bounds visible to the packet.

Read the dyadic tolerance row D. A local modulus can be named only where D records the finite comparison between the requested tolerance and the displayed spread between L and U.

Next pass through S and R. The StreamName row opens the sampled windows and value names, while the RegSeqRat row records the regular rational approximants needed by the Real-facing read.

Finally consume the seal and structural rows E, H, C, P, N. The resulting modulus is local to the displayed finite packet and therefore cannot carry an ambient compactness theorem, a trichotomy decision, quotient equality, a selected extremum, an arbitrary subcover selector, or full Real completeness. □

Closure status for NameCert$_{\text{ConstructiveRealOscillationGauge}^{\uparrow}}$.

Constructive story: A ConstructiveRealOscillationGauge$^{\uparrow}$ packet is generated from interval windows, local lower and upper oscillation bounds, a DyadicRatCore tolerance ledger, StreamName readback, RegSeqRat regularity, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveRealOscillationGauge}^{\uparrow}}$, seed)

Scope closed: Definition 8432.1, Definition 8432.2, and Definition 8432.3 seed the finite local oscillation-control route from explicit interval windows to Real-seal consumers.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveRealOscillationGauge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient equality, selected suprema, arbitrary subcover selection, full Real$^{\uparrow}$ completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coherence, dyadic oscillation accounting, StreamName readback, RegSeqRat regularity, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveRealOscillationGaugeUp.

Chapter 8433

A Concrete Naming Certificate for $\mathsf{GoodCover}^{\uparrow}$

Following Leray's good-cover method and the Bott–Tu and Hatcher presentations of cover nerves, the $\mathsf{GoodCover}^{\uparrow}$ packet records the finite BEDC surface of a cover whose visible nonempty finite intersections are contractible along displayed rows. It sits between the open-cover carrier of chapter 6731, the finite open-cover nerve of chapter 7481, the finite cover nerve of chapter 7597, the compact cover nerve rows used by $\mathsf{CompactCoverNerve}^{\uparrow}$, and the Cech nerve packet of chapter 6755. The packet is a finite good-cover ledger; it is not the nerve theorem, not a selected homotopy equivalence, not an ambient triangulation theorem, and not host contractibility of arbitrary intersections.

Definition 8433.1 (Good-cover carrier)**.** *A* $\mathsf{GoodCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, O, F, I, K, N, S, H, C, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row naming the open-set ambient surface.*
2. *O is the* $\mathsf{OpenCover}^{\uparrow}$ *row giving the displayed cover members.*
3. *F is the* $\mathsf{FiniteOpenCover}^{\uparrow}$ *or finite cover row fixing the finite index window.*
4. *I is the finite-intersection ledger for every nonempty displayed subwindow of F.*
5. *K is the good-intersection row recording contractible-cell evidence for each nonempty entry of I.*
6. *N is the nerve-face row shared with* $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{CompactCoverNerve}^{\uparrow}$, *and* $\mathsf{CechNerve}^{\uparrow}$ *consumers.*
7. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *export row for the finite nerve faces.*
8. *H records componentwise* hsame *transport for the topology, cover, finite index, intersection, good-intersection, nerve, simplicial, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from finite cover reads through the intersection and nerve rows.*
10. *P is the* Pkg *provenance over* $\mathsf{GoodCover}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{OpenCover}^{\uparrow}$, $\mathsf{FiniteOpenCover}^{\uparrow}$, $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{CechNerve}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. L is the local $\mathsf{NameCert}_{\mathsf{GoodCover}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, O, F, I, K, N, S, H, C, P, L$. It has no coordinate for selected points in intersections, ambient homotopy equivalence, quotient realization equality, a global triangulation, open-cover compactness, or the nerve theorem.

Theorem 8433.2 (Good-cover NameCert obligations)**.** *For every accepted* $\mathsf{GoodCover}^{\uparrow}$ *carrier $G = (T, O, F, I, K, N, S, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{GoodCover}^{\uparrow}}$ *surface consists exactly of topology-source exposure, open-cover exposure, finite index exposure, nonempty finite-intersection ledger exposure, good-intersection contractible-cell exposure, nerve-face exposure, simplicial-complex handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8433.1. Rows T, O, F expose the topological source, displayed cover, and finite index window before any intersection can be read.

The finite-intersection ledger I lists only displayed nonempty subwindows. The good-intersection row K attaches contractible-cell evidence to those ledger entries, and the nerve row N records the corresponding finite faces.

The simplicial export S is downstream of I, K, N. Transport, replay, provenance, and local naming are supplied by H, C, P, L, so no host nerve theorem, selected point, quotient realization, or global triangulation is part of the local certificate surface. □

Theorem 8433.3 (Good-cover finite-intersection nerve route)**.** *Every* $\mathsf{FiniteOpenCoverNerve}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{CompactCoverNerve}^{\uparrow}$, *or* $\mathsf{CechNerve}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{GoodCover}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{Topology}^{\uparrow} \longmapsto \mathsf{OpenCover}^{\uparrow} \longmapsto \mathsf{FiniteOpenCover}^{\uparrow} \longmapsto I \longmapsto K \longmapsto N \longmapsto \mathsf{SimplicialComplex}^{\uparrow}.$$

The consumer receives only the displayed cover rows, finite-intersection rows, contractible-cell rows, nerve-face rows, transport, replay, provenance, and local $\mathsf{NameCert}$ *evidence. It does not receive the nerve theorem, an ambient homotopy equivalence, a selected realization map, arbitrary open-cover compactness, quotient point equality, or global homology computation.*

Proof. Start with the finite cover source. A nerve-facing consumer can read N only after the cover member rows O and finite index row F have been exposed over the topology row T.

Next read the intersection data. The ledger I records exactly the displayed nonempty finite intersections, while K records the good-cover evidence attached to those intersections. Thus the nerve face row is determined by finite ledger data rather than by a host compactness or choice principle.

Finally apply the structural rows H, C, P, L. They transport, replay, source, and name the same route into the finite-open-cover, finite-cover, compact-cover, and Cech nerve siblings. Since the carrier has no coordinate for a nerve theorem or ambient homotopy equivalence, the route stops at the displayed finite nerve and simplicial-complex surface. □

Falsifiable prediction: An accepted $\mathsf{GoodCover}^{\uparrow}$ consumer that exports a nerve-theorem equivalence while omitting the displayed finite-intersection ledger I or good-intersection row K is not reading this carrier.

Independence witness: open-cover, finite-open-cover-nerve, finite-cover-nerve, compact-cover-nerve, cech-nerve: none is bijective with $\mathsf{GoodCover}^{\uparrow}$ because this packet jointly carries finite cover rows, nonempty finite-intersection rows, good-intersection contractible-cell evidence, nerve-face

export, transport, replay, provenance, and local naming while refusing the nerve theorem and ambient realization equality.

Closure status for $\mathsf{NameCert}_{\mathsf{GoodCover}^{\uparrow}}$.

Constructive story: A $\mathsf{GoodCover}^{\uparrow}$ packet is generated from a TopologyUp source row, OpenCoverUp cover rows, a finite cover index, a finite-intersection ledger, good-intersection contractible-cell evidence, nerve-face rows, a SimplicialComplexUp export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GoodCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8433.1, Definition 8433.2, and Definition 8433.3 seed the finite good-cover surface linking open-cover, finite-cover-nerve, compact-cover-nerve, and Cech nerve consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GoodCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the nerve theorem, selected intersection points, ambient homotopy equivalence, quotient realization equality, global triangulation, host open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover exposure, intersection-ledger exactness, good-intersection evidence, nerve-face handoff, simplicial export, transport, replay, provenance, and local naming for BEDC.Derived.GoodCoverUp.

Chapter 8434

A Concrete Naming Certificate for $\mathsf{NormalizationBudgetTrace}^{\uparrow}$

The $\mathsf{NormalizationBudgetTrace}^{\uparrow}$ packet realizes the strong-normalization boundary of chapter 9370 as finite trace data for a total host reading a possibly universal substrate. It sits beside the step-indexed trace carrier of chapter 8123, the total-host fuel handoff of chapter 962, the reduction-fuel boundary of chapter 970, the MetaCIC normalization frontier of chapter 892, and the normalization audit map of chapter 9383. The packet names only finite fuel, finite prefix, endpoint, refusal, replay, provenance, and local naming rows; it does not assert global strong normalization, substrate halting, a divergence oracle, quotient trace equality, or host evaluation completeness.

Definition 8434.1 (Normalization budget trace carrier)**.** *A* $\mathsf{NormalizationBudgetTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, F, P, E, R, A, H, C, Q, N)$$

with the following displayed rows:

1. *T is the typed host term or substrate-state window being inspected.*
2. *F is the finite fuel row bounding the normalization read.*
3. *P is the finite prefix trace row produced under that fuel.*
4. *E is the endpoint row, recording either a displayed normal endpoint or the last bounded substrate state.*
5. *R is the refusal row for requests that ask for global strong normalization, unbounded halting, divergence detection, or evidence-free endpoint production.*
6. *A is the audit-route row linking the packet to chapter 9370 and chapter 9383.*
7. *H records componentwise* hsame *transport for the term, fuel, prefix, endpoint, refusal, audit, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the bounded fuel request through the finite prefix to the endpoint or refusal row.*
9. *Q is the* Pkg *provenance over* $\mathsf{NormalizationBudgetTrace}^{\uparrow}$, $\mathsf{StepIndexedTrace}^{\uparrow}$, $\mathsf{TotalHostFuelHandoff}^{\uparrow}$, $\mathsf{ReductionFuelBoundary}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationBudgetTrace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, F, P, E, R, A, H, C, Q, N$. It has no coordinate for an unbounded reduction, a host evaluator oracle, quotient trace equality, substrate halting, divergence detection, global strong normalization, or completed MetaCIC normalization.

Theorem 8434.2 (Normalization budget trace NameCert obligations)**.** *For every accepted* NormalizationBudgetTra *carrier $B = (T, F, P, E, R, A, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{NormalizationBudgetTrace}^{\uparrow}}$ *surface consists exactly of term-window exposure, finite fuel exposure, prefix-trace exposure, endpoint or bounded-state exposure, refusal-ledger exposure, audit-route exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8434.1. The rows T, F, P, E provide the only computational data: a term or substrate window, a finite fuel bound, a finite prefix, and the endpoint reached inside that bound.

The refusal row R records every request whose target is not carried by the finite packet. In particular, global strong normalization, unbounded substrate halting, divergence detection, and evidence-free endpoint production have no exported coordinate.

The audit row A places the bounded read inside the strong-normalization and MetaCIC audit routes. The structural rows H, C, Q, N transport, replay, source, and name that same finite route, so the listed rows exhaust the local NameCert surface. □

Theorem 8434.3 (Normalization budget trace total-host boundary)**.** *Every total-host or MetaCIC normalization consumer of an accepted* $\mathsf{NormalizationBudgetTrace}^{\uparrow}$ *packet factors through the finite route*

$$T \longmapsto F \longmapsto P \longmapsto E \longmapsto R \longmapsto A.$$

The consumer may read bounded fuel, a finite prefix, a displayed endpoint or bounded last state, the refusal ledger, the audit route, transport, replay, provenance, and local NameCert *evidence. It receives no global strong-normalization theorem, no substrate halting oracle, no divergence detector, no quotient trace equality, and no host evaluator completeness row.*

Proof. Start with the total-host discipline of the vision chapter. A BEDC consumer can inspect the substrate or MetaCIC normalization route only through a finite fuel row, so F must be displayed before any prefix or endpoint row is read.

Next read P and E. The prefix row records the bounded trace, and the endpoint row records only what is visible under that fuel. If the consumer asks beyond those rows, the request lands in the refusal ledger R rather than producing an unbounded normalization claim.

Finally use A, H, C, Q, N. The audit route keeps the packet attached to the strong-normalization and MetaCIC normalization maps, while transport, replay, provenance, and local naming preserve the same finite bounded trace. Since no carrier coordinate names halting, divergence, quotient trace equality, or evaluator completeness, the total-host boundary is exactly the displayed route. □

Falsifiable prediction: An accepted $\mathsf{NormalizationBudgetTrace}^{\uparrow}$ consumer that exports global strong normalization while omitting finite fuel F, finite prefix P, endpoint row E, or refusal row R is not reading this carrier.

Independence witness: step-indexed-trace, totalhostfuelhandoff, reductionfuelboundary, metacicnormalizationfrontier, metacic-normalization-audit-map: no listed sibling is bijective with NormalizationBudgetTra because this packet jointly carries the term window, finite fuel, prefix trace, endpoint row, refusal ledger, audit route, transport, replay, provenance, and local naming as one bounded total-host trace.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationBudgetTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalizationBudgetTrace}^{\uparrow}$ packet is generated from a typed host or substrate-state window, finite fuel, finite prefix trace, endpoint or bounded-state row, refusal ledger, audit-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationBudgetTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8434.1, Definition 8434.2, and Definition 8434.3 seed the finite total-host budget trace for normalization and substrate-boundary consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationBudgetTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, unbounded substrate halting, divergence detection, quotient trace equality, host evaluator completeness, completed MetaCIC normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fuel exposure, prefix exactness, endpoint or refusal exactness, audit-route handoff, transport, replay, provenance, and local naming for BEDC.Derived.NormalizationBudgetTraceUp.

Chapter 8435

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealFunction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealFunction}^{\uparrow}$ packet records a located real-valued function as displayed finite input-location requests and regular real output rows, not as a host function or quotient map. It sits between the terminal real-name seal of chapter 38, the located real input boundary of chapter 1933, the regular rational readback of chapter 301, the finite observation interface of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet gives Real and Completion consumers a function-facing surface whose inputs, output approximants, tolerances, transports, and local name remain finite BEDC rows.

Definition 8435.1 (Bishop located real function carrier)**.** *A* $\mathsf{BishopLocatedRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Q, L, D, S, R, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the input real-name row, opened only through the* $\mathsf{Real}^{\uparrow}$ *seal named by the packet.*
2. *Q is the locatedness request row, sourced through* $\mathsf{LocatedReal}^{\uparrow}$*, that records the finite positive/negative separation tests allowed for the input.*
3. *L is the local input-window ledger. It records which located input cells are served by the packet and which tolerance request each cell answers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the input request, output request, slack, and comparison budget.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window opened before the output is read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to S and D.*
7. *O is the displayed output real-name seal reached only after Q, L, D, S, R have been exposed.*
8. *H records componentwise* hsame *transport for the input seal, locatedness requests, local windows, dyadic tolerances, StreamName windows, RegSeqRat readback, output seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a located input request through the same rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Q, L, D, S, R, O, H, C, P, N$. It has no coordinate for general continuity, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host function equality, quotient equality of real names, countable choice, or a selected global modulus.*

Theorem 8435.2 (Bishop located real function NameCert obligations)**.** *For every accepted* BishopLocatedRealFunc *carrier*

$$F = (X, Q, L, D, S, R, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located-input admission through Q, local window exactness through L, dyadic tolerance accounting through D, StreamName window exactness through S, RegSeqRat output readback through R, output Real-seal nonescape through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 8435.1. The packet exposes exactly the input seal, locatedness request row, local input-window ledger, dyadic tolerance ledger, StreamName window, RegSeqRat readback, output seal, transport, replay, provenance, and local name.

The located-input obligations are read through Q, L, D. The locatedness row Q determines which finite input separation tests are allowed, L records the served local windows, and D fixes the tolerance and comparison budget for those requests.

The output obligations are read through S, R, O. StreamName opens only finite observations, RegSeqRat records the regular rational output approximants attached to those observations, and the output Real seal is downstream of that readback.

Finally H, C, P, N preserve and name the same route. Since no coordinate stores general continuity, host function equality, quotient equality, countable choice, a selected global modulus, or ambient Real completeness, none of those principles is exported by the local NameCert surface. □

Theorem 8435.3 (Bishop located real function located-output stability)**.** *Every accepted* BishopLocatedRealFunctio *read that serves a located input request through*

$$F = (X, Q, L, D, S, R, O, H, C, P, N)$$

must expose the same located input row Q, local window ledger L, dyadic tolerance ledger D, finite StreamName window S, RegSeqRat readback R, and output Real seal O before the output value is consumed. Refining a served input window may add displayed rows inside L, S, R, but the public output read remains the same packet route through H, C, P, N and cannot be presented as a host function value, quotient equality class, selected limit, general continuity theorem, or completeness witness.

Proof. Start with the located input request. Row Q supplies the locatedness data, while L records the finite input windows to which the packet responds.

Read the dyadic tolerance ledger D. It fixes the input and output precision budget, so a refinement must remain inside the displayed comparison route rather than changing the function packet.

Next pass through S, R, O. The StreamName and RegSeqRat rows expose finite output approximants, and the terminal Real seal is consumed only after those rows are visible.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Therefore a located-output read is stable under displayed finite refinement, but it never becomes host function equality, quotient equality, general continuity, selected completion data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealFunction}^{\uparrow}$ packet is generated from a RealUp input seal, LocatedRealUp request rows, local input windows, a DyadicRatCoreUp tolerance ledger, StreamNameUp finite windows, RegSeqRatUp output readback, a RealUp output seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8435.1, Definition 8435.2, and Definition 8435.3 seed the located real-valued function packet over displayed RealUp, LocatedRealUp, RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BishopLocatedRealFunctionUp`

Bridge status: none

Not claimed: This chapter does not claim general continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, host function equality, quotient equality of real names, countable choice, selected limits, Lean theorem checking, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-input admission, local window exactness, dyadic tolerance accounting, StreamName window exactness, RegSeqRat output readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealFunctionUp.

Chapter 8436

A Concrete Naming Certificate for $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$

Following Grothendieck's Tohoku paper and Weibel's homological algebra account, the GrothendieckSpectralSequenc packet records the finite NameCert surface for the classical composite-derived-functor spectral sequence. It sits after the abelian-category carrier of chapter 176, the category, functor, and natural-transformation rows of chapter 61, chapter 62, and chapter 63, the derived-functor spine of chapter 177, and the page ledger of chapter 178. The packet names only the displayed composite-functor source, acyclicity ledger, page rows, filtration target, transport, replay, provenance, and local naming data; it is not a host proof of the full Grothendieck spectral sequence and does not claim convergence beyond the finite route recorded below.

Definition 8436.1 (Grothendieck spectral sequence carrier)**.** *A* $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, C, F, G_0, \eta, D, E, P, T, H, R, Q, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row in which exactness, injective or acyclic resolutions, and homology reads are interpreted.*
2. *C is the* $\mathsf{Cat}^{\uparrow}$ *dependency row for the source, middle, and target categories of the composite construction.*
3. *F and G_0 are the two* $\mathsf{Functor}^{\uparrow}$ *rows whose composite is being inspected.*
4. *η is the* $\mathsf{NatTrans}^{\uparrow}$ *comparison row recording the displayed compatibility data used by the composite-functor route.*
5. *D is the* $\mathsf{DerivedFunctor}^{\uparrow}$ *row for the inner and outer derived-functor reads, including the resolution, degree, cycle-boundary, and* Cont *ledgers visible to downstream consumers.*
6. *E is the* $\mathsf{SpectralSeq}^{\uparrow}$ *page row, carrying the displayed E_2 page, differential rows, successor-page route, and filtration readback boundary.*
7. *P is the finite acyclicity and resolution-compatibility ledger that allows the inner derived-functor row to feed the outer derived-functor row.*
8. *T is the target filtration row, recording only the displayed filtration boundary of the composite derived functor.*

9. *H records componentwise* hsame *transport for the abelian, category, functor, natural-transformation, derived-functor, spectral-sequence, acyclicity, target-filtration, replay, provenance, and local naming rows.*
10. *R records displayed* Cont *replay from the composite source through F, G_0, η, D, P, E, T.*
11. *Q is the* Pkg *provenance over* $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{DerivedFunctor}^{\uparrow}$, $\mathsf{SpectralSeq}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GrothendieckSpectralSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, C, F, G_0, \eta, D, E, P, T, H, R, Q, N$. It has no coordinate for host derived-category equality, a global convergence theorem, chosen resolutions outside the displayed ledger, exactness beyond the acyclicity row, or quotient equality of pages.

Theorem 8436.2 (Grothendieck spectral sequence NameCert obligations)**.** *For every accepted* $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$ *carrier $G = (A, C, F, G_0, \eta, D, E, P, T, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{GrothendieckSpec}}$ *surface consists exactly of carrier admission for those thirteen rows; source exposure through* $\mathsf{AbelianCat}^{\uparrow}$ *and* $\mathsf{Cat}^{\uparrow}$*; functorial exposure through the two* $\mathsf{Functor}^{\uparrow}$ *rows and the* $\mathsf{NatTrans}^{\uparrow}$ *comparison row; derived-functor exposure through D; spectral-sequence page exposure through E; acyclicity and resolution compatibility through P; target-filtration exposure through T; componentwise* hsame *transport through H; displayed* Cont *replay through R;* Pkg *provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 8436.1. The only source rows are A, C, F, G_0, η, and the only homological rows are D, E, P, T.

Read the functorial source first. The category and abelian-category rows type the two functors and the natural-transformation comparison, so the composite route is a displayed BEDC route rather than an ambient functor category.

Next read the homological ledger. The derived-functor row D supplies the inner and outer derived reads, P records the acyclicity and resolution-compatibility budget, E records the spectral-sequence page ledger, and T records the target filtration boundary.

Finally use H, R, Q, N. Transport, replay, provenance, and local naming preserve exactly those displayed rows. Since no carrier coordinate stores a host convergence theorem, chosen external resolution, derived-category equality, or quotient page equality, those data are not NameCert obligations of the packet. □

Theorem 8436.3 (Grothendieck spectral sequence derived-functor route)**.** *Every derived-functor consumer of an accepted* $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{AbelianCat}^{\uparrow},\ \mathsf{Cat}^{\uparrow},\ \mathsf{Functor}^{\uparrow},\ \mathsf{NatTrans}^{\uparrow} \longmapsto \mathsf{DerivedFunctor}^{\uparrow} \longmapsto \mathsf{SpectralSeq}^{\uparrow} \longmapsto T, H, R, Q, N.$$

The route exposes the composite source, the acyclicity ledger, the page ledger, and the target-filtration boundary before any derived-functor output is read. It does not export a host Grothendieck spectral sequence theorem, global convergence assertion, selected resolution outside P, or quotient equality of spectral-sequence pages.

Proof. Begin with the NameCert obligations of Definition 8436.2. They force every read through the abelian and categorical source rows, the two functor rows, and the natural-transformation comparison row before any derived-functor or spectral-sequence row is consumed.

The derived-functor handoff is then the row D. It is scoped by the existing $\mathsf{DerivedFunctor}^{\uparrow}$ surface and by the acyclicity and resolution-compatibility row P, so the consumer receives only displayed resolution, degree, cycle-boundary, and continuation ledgers.

The spectral-sequence handoff is the row E, followed by the target filtration row T. These rows are read through the existing $\mathsf{SpectralSeq}^{\uparrow}$ page ledger and do not add a separate convergence theorem.

Finally H, R, Q, N preserve, replay, source, and name the same route. Thus the accepted consumer sees a finite composite-derived-functor packet and no unlisted host-level exactness or convergence data. □

Falsifiable prediction: Every accepted $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$ consumer that reads a target-filtration row must first expose the $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{DerivedFunctor}^{\uparrow}$, $\mathsf{SpectralSeq}^{\uparrow}$, acyclicity, BHist, hsame, Cont, Pkg, and NameCert rows listed in the carrier.

Independence witness: abelian category, category, functor, natural transformation, derived functor, spectral sequence: none is bijective to $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$, because this packet binds the composite-functor source, acyclicity ledger, derived-functor handoff, page ledger, and target-filtration boundary into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{GrothendieckSpectralSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{GrothendieckSpectralSequence}^{\uparrow}$ packet is generated from AbelianCatUp source rows, CategoryUp source rows, two FunctorUp rows, a NatTransUp comparison row, a DerivedFunctorUp handoff, a SpectralSeqUp page ledger, an acyclicity and resolution-compatibility ledger, a target-filtration row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GrothendieckSpectralSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8436.1, Definition 8436.2, and Definition 8436.3 seed the finite composite-derived-functor spectral-sequence route over the displayed dependency rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GrothendieckSpectralSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host derived-category equality, a full Grothendieck spectral sequence theorem, global convergence, selected external resolutions, quotient page equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, composite-functor source exposure, acyclicity-ledger exactness, derived-functor handoff, spectral-sequence page routing, target-filtration nonescape, transport, replay, provenance, and local naming for BEDC.Derived.GrothendieckSpectralSequenceUp.

Chapter 8437

A Concrete Naming Certificate for RegularCauchyCompactnessCriterion$^{\uparrow}$

Following Bishop and Bridges constructive compactness criteria, the RegularCauchyCompactnessCriterion$^{\uparrow}$ packet records the finite route by which regular Cauchy real data are admitted to a compactness read. It sits after the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the interval finite-net surface of chapter 1379, and the finite Cauchy completion criterion of chapter 8429. The packet names a criterion surface for Real and Completion consumers; it does not import sequential compactness, open-cover compactness, quotient stream equality, selected cluster points, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8437.1 (Regular Cauchy compactness criterion carrier)**.** *A* RegularCauchyCompactnessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, R, C, W, H, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *finite-window row that records which name windows are inspected before the compactness read.*
2. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the regular rational data exposed by those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row controlling mesh size, Cauchy radius, and comparison budget.*
4. *R is the terminal* Real$^{\uparrow}$ *seal row reached only after S, Q, D have been displayed.*
5. *C is the compact finite-window witness row, read through the public finite-net interval export of chapter 1379.*
6. *W is the finite-window extraction row that records the inspected window and net-cell data consumed by a compactness query.*
7. *H records componentwise* hsame *transport for S, Q, D, R, C, W, P, N.*
8. *P is the displayed* Cont *replay and* Pkg *provenance over* RegularCauchyCompactnessCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, HeineBorelInterval$^{\uparrow}$, FiniteCauchyCompletenessCriterion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that the compactness criterion is read from these finite rows. It has no coordinate for sequential compactness, open-cover compactness, a selected subsequence, a selected cluster point, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8437.2 (Regular Cauchy compactness criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}$ *carrier* $K = (S, Q, D, R, C, W, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{Real}^{\uparrow}$ *seal row, compact finite-window witness row, finite-window extraction row, componentwise* hsame *transport, displayed* Cont *replay with* Pkg *provenance, and local naming. No obligation row may be discharged by host compactness, sequential compactness, open-cover compactness, quotient stream equality, selected representatives, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 8437.1. Its displayed coordinates are exactly $S, Q, D, R, C, W, H, P, N$, so carrier admission and local naming can only inspect those rows.

Read the L10 source rows in order. The finite windows S expose the regular rational readback Q, the dyadic tolerance row D controls the Cauchy and mesh budgets, and the terminal real seal R is downstream of those displayed rows.

Now read the compactness side. The row C is a compact finite-window witness through the Heine–Borel interval finite-net surface, and W records the actual extraction window consumed by the query. Transport, replay, provenance, and local naming preserve that same finite route, so the excluded compactness and choice principles have no carrier coordinate. □

Theorem 8437.3 (Regular Cauchy compactness criterion finite-window extraction)**.** *Every compactness read of an accepted* $\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}$ *carrier factors through the finite extraction route*

$$S, Q, D, R, C, W, H, P, N.$$

In particular, the read consumes StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, the Real seal, the Heine–Borel finite-net witness, the displayed extraction window, transport, replay, provenance, and local naming before it can expose compactness data. It is therefore a finite-window extraction, not sequential compactness, not open-cover compactness, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8437.2. They identify the complete local surface of an accepted carrier, so a compactness read cannot leave the rows $S, Q, D, R, C, W, H, P, N$.

Use the Real-facing route next. Rows S, Q, D provide finite observation, regular rational readback, and tolerance accounting before the terminal seal R is available.

Finally apply the compact finite-net rows C, W. The public finite-net interval export supplies the compact witness as displayed net data, while W records the finite extraction actually inspected by the consumer. Since neither row contains a selected subsequence, an open-cover subcover choice, quotient stream equality, or an ambient completeness theorem, the compactness read is exactly the stated finite-window extraction. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal RealUp seal, a compact finite-window witness, a finite-window extraction row, componentwise hsame transport, displayed Cont replay with Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompactnessCriterion}\uparrow}$, seed)

Scope closed: Definition 8437.1, Definition 8437.2, and Definition 8437.3 seed the finite compactness criterion route over StreamName, RegSeqRat, DyadicRatCore, Real, Heine–Borel interval, and finite Cauchy completion data.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompactnessCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim sequential compactness, open-cover compactness, selected subcovers, selected cluster points, quotient equality of real names, countable choice, Lean verification, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, regular rational read-back, dyadic tolerance accounting, Real seal routing, Heine–Borel finite-net handoff, finite Cauchy completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompactnessCriterionUp.

Chapter 8438

A Concrete Naming Certificate for $\mathsf{SpectralSequenceComparison}^{\uparrow}$

Following Weibel and McCleary on spectral-sequence comparison theorems, the $\mathsf{SpectralSequenceComparison}^{\uparrow}$ packet records the finite NameCert surface by which displayed page data in $\mathsf{SpectralSeq}^{\uparrow}$ can be compared before any downstream derived-functor spectral-sequence consumer reads it. It sits after the spectral-sequence page carrier of chapter 178, the homology carrier of chapter 98, the abelian-category carrier of chapter 176, the derived-functor spine of chapter 177, and the composite-derived-functor packet of chapter 8436. The packet names only the displayed source pages, comparison rows, page-stability ledger, filtration boundary, transport, replay, provenance, and local naming data; it does not claim convergence, exactness beyond the recorded rows, or a host-level comparison theorem.

Definition 8438.1 (Spectral sequence comparison carrier)**.** *A* $\mathsf{SpectralSequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S_0, S_1, H_0, A, D, F, L, T, R, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two* $\mathsf{SpectralSeq}^{\uparrow}$ *page carriers being compared, including their page objects, differential rows, successor-page routes, and filtration readback boundaries.*
2. *H_0 is the* $\mathsf{Homology}^{\uparrow}$ *readback row for the pagewise cycles, boundaries, and induced homology rows visible in the comparison.*
3. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row in which the page objects, differentials, and homology reads are interpreted.*
4. *D is the* $\mathsf{DerivedFunctor}^{\uparrow}$ *consumer row that records how the compared pages feed derived-functor data.*
5. *F is the pagewise comparison row, listing the displayed maps or rowwise* hsame *transports used between corresponding page entries.*
6. *L is the finite page-stability ledger. It records the range of pages, bidegrees, differential compatibility checks, and filtration-boundary agreements that are actually inspected.*
7. *T records componentwise* hsame *transport for $S_0, S_1, H_0, A, D, F, L$ and the structural rows.*

8. *R records displayed* Cont *replay from the source page rows through the comparison row and the page-stability ledger.*
9. *P is the* Pkg *provenance over* SpectralSequenceComparison$^{\uparrow}$, SpectralSeq$^{\uparrow}$, Homology$^{\uparrow}$, AbelianCat$^{\uparrow}$, DerivedFunctor$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{SpectralSequenceComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, H_0, A, D, F, L, T, R, P, N$. It has no coordinate for global convergence, an ambient equality of spectral sequences, quotient equality of pages, selected resolutions, exact-couple data not displayed in L, or a host comparison theorem.

Theorem 8438.2 (Spectral sequence comparison NameCert obligations)**.** *For every accepted* SpectralSequenceComparison$^{\uparrow}$ *carrier $C = (S_0, S_1, H_0, A, D, F, L, T, R, P, N)$, the local* NameCert$_{\text{SpectralSequenceCompar}}$ *surface consists exactly of carrier admission for those eleven rows; page exposure through the two* SpectralSeq$^{\uparrow}$ *rows; homology readback through H_0; abelian source exposure through A; derived-functor consumer exposure through D; displayed pagewise comparison through F; page-stability and filtration-boundary exposure through L; componentwise* hsame *transport through T; displayed* Cont *replay through R;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8438.1. Its only spectral-sequence inputs are the displayed page rows S_0 and S_1, and its only ambient source rows are H_0, A, D.

Read the pagewise comparison next. The row F lists the inspected maps or transports between corresponding page entries, while L records the finite page range, bidegree range, differential compatibility checks, and filtration-boundary agreements used by the comparison.

Finally use T, R, P, N. Transport, replay, provenance, and local naming preserve exactly the listed rows. Since no coordinate stores a global convergence theorem, ambient equality of spectral sequences, quotient page equality, selected resolution, or host comparison theorem, none of those data is a NameCert obligation of the packet. □

Theorem 8438.3 (Spectral sequence comparison page stability)**.** *Every downstream* DerivedFunctor$^{\uparrow}$ *or* GrothendieckSpectralSequence$^{\uparrow}$ *consumer of an accepted* SpectralSequenceComparison$^{\uparrow}$ *packet must read the finite route*

$$S_0, S_1, H_0, A \longmapsto F, L \longmapsto D, T, R, P, N$$

before it may use the comparison as page-stability data. The route exposes the two SpectralSeq$^{\uparrow}$ *page carriers, homology readback, abelian source row, pagewise comparison row, and stability ledger before any derived-functor or composite-derived-functor consumer reads the result. It exports no convergence theorem, quotient equality of pages, selected external resolution, countable choice, or ambient spectral-sequence equality.*

Proof. Begin with the NameCert obligations of Definition 8438.2. They force every comparison read to expose the two spectral-sequence page packets, the homology readback row, and the abelian source row before the derived-functor consumer row is available.

The comparison data is then finite. The row F supplies the displayed pagewise maps or transports, and L records the inspected page range, bidegree range, differential compatibility, and filtration-boundary agreement. A downstream consumer therefore receives a stability ledger rather than an unbounded comparison theorem.

Finally replay the structural rows T, R, P, N. These rows transport and name the same finite packet for DerivedFunctor$^{\uparrow}$ and GrothendieckSpectralSequence$^{\uparrow}$ consumers, so the public route has no coordinate for convergence, quotient page equality, selected resolutions, countable choice, or ambient spectral-sequence equality. □

Falsifiable prediction: Every accepted $\mathsf{SpectralSequenceComparison}^{\uparrow}$ consumer that reads page-stability data must first expose the two $\mathsf{SpectralSeq}^{\uparrow}$ page carriers, $\mathsf{Homology}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{DerivedFunctor}^{\uparrow}$, comparison, stability-ledger, BHist, hsame, Cont, Pkg, and NameCert rows listed in the carrier.

Independence witness: spectral sequence, homology, abelian category, derived functor, Grothendieck spectral sequence: none is bijective to $\mathsf{SpectralSequenceComparison}^{\uparrow}$, because this packet binds two page carriers, homology readback, abelian source data, a pagewise comparison row, a finite stability ledger, and a derived-functor handoff into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralSequenceComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralSequenceComparison}^{\uparrow}$ packet is generated from two SpectralSeqUp page rows, HomologyUp readback, an AbelianCatUp source row, a DerivedFunctorUp consumer row, a pagewise comparison row, a finite page-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpectralSequenceComparison}^{\uparrow}}$, seed)

Scope closed: Definition 8438.1, Definition 8438.2, and Definition 8438.3 seed the finite comparison route over displayed spectral-sequence page data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpectralSequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim convergence, a host spectral-sequence comparison theorem, quotient page equality, selected external resolutions, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, page exposure, homology readback, abelian source exposure, derived-functor handoff, pagewise comparison, page-stability routing, transport, replay, provenance, and local naming for BEDC.Derived.SpectralSequenceComparisonUp.

Chapter 8439

A Concrete Naming Certificate for $\mathsf{BrouwerChoiceSequence}^{\uparrow}$

Following Brouwer and Bishop–Bridges, the $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ packet records finite displayed choice-sequence data routed through Brouwer spread admission and the L10 countable-name surfaces. It sits between the choice-sequence surface of chapter 1482, the spread-prefix discipline of chapter 1677, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet gives Real and Completion consumers a displayed finite bridge from choice-sequence observations to spread-admissible name reads without importing countable choice, quotient streams, selected infinite branches, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8439.1 (Brouwer choice-sequence carrier)**.** *A* $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, A, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite schedule row listing the choice-sequence questions and precision requests that have been opened.*
2. *W is the displayed prefix-window row, recording only the finite choice-sequence answers visible to the packet.*
3. *A is the answer-ledger row, pairing every displayed question in S with its answer in W.*
4. *T is the* $\mathsf{BrouwerSpread}^{\uparrow}$ *admissibility row, certifying that the displayed prefix window is accepted by the spread route before any downstream read occurs.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the admitted prefix window and its finite stream-name handoff.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, A, T, R have been displayed.*
7. *H records componentwise* hsame *transport for the schedule, prefix window, answer ledger, spread-admissibility, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for opening the choice-sequence window, checking spread admission, handing the window to* $\mathsf{StreamName}^{\uparrow}$*, reading* $\mathsf{RegSeqRat}^{\uparrow}$*, and sealing through* $\mathsf{Real}^{\uparrow}$*.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerChoiceSequence}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{BrouwerSpread}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerChoiceSequence}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $S, W, A, T, R, E, H, C, P, N$. It has no coordinate for a completed host function, quotient stream equality, countable choice, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8439.2 (Brouwer choice-sequence NameCert obligations)**.** *For the carrier of Definition 8439.1, the local* $\mathsf{NameCert}_{\mathsf{BrouwerChoiceSequence}^{\uparrow}}$ *surface consists of nine finite obligations:*

1. *carrier admission exposes exactly the rows $S, W, A, T, R, E, H, C, P, N$;*
2. *prefix stability preserves the visible window W and answer ledger A under later finite schedule extension;*
3. *spread admissibility reads the* $\mathsf{BrouwerSpread}^{\uparrow}$ *row T before any stream-name or regular-rational handoff;*
4. *StreamName handoff consumes only the displayed finite prefix window and answer ledger;*
5. *RegSeqRat and Real boundary rows expose R before the terminal seal E;*
6. *componentwise transport preserves S, W, A, T, R, E along H;*
7. *continuation replay reads the same finite route through C;*
8. *provenance records the choice-sequence, spread, stream-name, regular-rational, and Real siblings through P;*
9. *the ledger refuses completed host functions, quotient streams, countable choice, selected infinite branches, and ambient Real completeness as certificate data.*

Proof. Carrier admission is projection from Definition 8439.1. The displayed packet has exactly the schedule, prefix-window, answer-ledger, spread-admissibility, readback, seal, transport, replay, provenance, and local naming rows.

Prefix stability follows from the classifier. Since S, W, A are visible rows and no hidden tail coordinate exists, later finite inspection can preserve the already displayed prefix only by preserving those rows.

Spread admissibility and the StreamName handoff are read in order through T and then the finite window data. The regular-rational row R and terminal seal E are downstream of that admitted route.

Finally, H, C, P, N provide transport, replay, provenance, and local naming for the same finite packet. The refusal clause is the absence of any coordinate for completed host functions, quotient streams, countable choice, selected infinite branches, or ambient Real completeness. □

Theorem 8439.3 (Brouwer choice-sequence spread-stream handoff)**.** *Every consumer of an accepted* $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ *carrier first reads the* $\mathsf{ChoiceSequence}^{\uparrow}$ *rows S, W, A and the* $\mathsf{BrouwerSpread}^{\uparrow}$ *row T before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *handoff is public. The downstream route is therefore*

$$\mathsf{ChoiceSequence}^{\uparrow} \longmapsto \mathsf{BrouwerSpread}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with the concrete packet rows $S, W, A, T, R, E, H, C, P, N$ displayed along the way.

Proof. Start with the NameCert obligations of Definition 8439.2. They require carrier admission, prefix stability, and spread admissibility before the stream-name handoff is consumed.

The carrier definition places R after the admitted spread row and makes E a terminal seal after the regular-rational readback. Thus a Real-facing consumer cannot reach E without first reading S, W, A, T, R.

The transport, replay, provenance, and local naming rows preserve the same order for every consumer. Hence the handoff factors through the displayed choice-sequence and Brouwer-spread rows before StreamName, RegSeqRat, or Real data become public. □

Falsifiable prediction: An accepted $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ read is refuted if it reaches $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ data without first displaying the finite choice-sequence rows S, W, A and the $\mathsf{BrouwerSpread}^{\uparrow}$ admissibility row T.

Independence witness: choice-sequence, brouwer-spread, streamname, regseqrat, real: $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ is not bijective to any sibling because it binds finite choice-sequence questions, displayed answers, spread-prefix admission, L10 countable-name handoff, regular-rational readback, Real sealing, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerChoiceSequence}^{\uparrow}$ packet is generated from finite choice-sequence schedule rows, displayed prefix windows, answer ledgers, BrouwerSpread admission, StreamName handoff, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8439.1, Definition 8439.2, and Definition 8439.3 seed the finite route from displayed choice-sequence prefixes through Brouwer spread admission, StreamName handoff, RegSeqRat readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed host functions, quotient stream equality, countable choice, selected infinite branches, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix stability, spread admissibility, StreamName handoff, RegSeqRat and Real boundary, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerChoiceSequenceUp.

Lean4 (statement) `BEDC.Derived.BrouwerChoiceSequenceUp`

Chapter 8440

A Concrete Naming Certificate for $\mathsf{TotallyBoundedReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{TotallyBoundedReflection}^{\uparrow}$ packet records the finite precompact-reflection surface by which a uniform carrier exposes total-bounded schedules before any completion or compactness consumer reads it. It sits over the uniform source of chapter 339, the entourage basis of chapter 1819, the finite-net surface of chapter 127, the stream observation rows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite reflection carrier for later completion and compact-metric routes; it does not assert open-cover compactness, a selected Cauchy limit, quotient equality, countable choice, or ambient completeness.

Definition 8440.1 (Totally bounded reflection carrier)**.** *A* $\mathsf{TotallyBoundedReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (U, E, T, S, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row whose uniform carrier is being reflected.*
2. *E is the* $\mathsf{EntourageBase}^{\uparrow}$ *row naming the finite entourage requests available to downstream consumers.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedule for each displayed entourage request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row recording the finite stream windows by which net centers and coverage witnesses are inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same observation windows when rational radii or distances must be read.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the entourage, net, stream, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transport for $U, E, T, S, Q, A, C, P, N$.*
8. *C records displayed* Cont *replay from a precompact-reflection request through U, E, T, S, Q, A and then through the support rows.*
9. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedReflection}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{TotallyBoundedReflection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed tuple $U, E, T, S, Q, A, H, C, P, N$. It has no coordinate for open-cover compactness, a selected completion point, quotient real equality, countable choice, or ambient complete uniform space data.

Theorem 8440.2 (Totally bounded reflection NameCert obligations)**.** *For every accepted* TotallyBoundedReflection *carrier*

$$R = (U, E, T, S, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TotallyBoundedReflection}^{\uparrow}}$ *surface consists exactly of uniform-source admission, entourage-basis exposure, total-bounded finite-net scheduling, stream-window inspection, regular-rational readback, Real seal non-escape, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing or compact-metric consumer must read $U, E, T, S, Q, A, H, C, P, N$ before it treats the source as precompact.*

Proof. Project the carrier of Definition 8440.1. The rows U, E, T expose the uniform source, the entourage basis, and the finite total-bounded schedules before any reflected route can be consumed.

The stream and regular-rational rows S, Q provide the only observation and radius readback interface. The Real seal row A is downstream of those rows, so a consumer cannot replace the displayed net schedule by quotient real equality or by a selected completion point.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores open-cover compactness, a choice-selected subcover, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{TotallyBoundedReflection}^{\uparrow}$ consumer that exports a precompact reflection must display the $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: uniformspace, entourage-base, totallybounded, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{TotallyBoundedReflection}^{\uparrow}$ because this packet jointly binds a uniform carrier, entourage requests, finite-net schedules, stream windows, regular-rational readback, Real seal non-escape, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{TotallyBoundedReflection}^{\uparrow}$ packet is generated from UniformSpace source data, EntourageBase request rows, TotallyBounded finite-net schedules, StreamName observation windows, RegSeqRat readback, a Real seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TotallyBoundedReflection}^{\uparrow}}$, seed)

Scope closed: Definition 8440.1 and Definition 8440.2 seed the finite precompact-reflection surface for later completion and compact-metric consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TotallyBoundedReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected completion points, quotient real equality, countable choice, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, entourage-basis exposure, finite-net scheduling, stream-window inspection, regular-rational readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.TotallyBoundedReflectionUp.

Chapter 8441

A Concrete Naming Certificate for $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed located upper-cut presentation may be read through regular Cauchy and Real-seal data without turning that read into a Dedekind construction or an ambient completeness principle. It sits over the located-cut real route of chapter 3773, the two-sided located Dedekind cut packet of chapter 7256, the regular-Cauchy located-cut handoff of chapter 7018, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a local upper-cut criterion for Real and Completion consumers; it does not construct a Dedekind real, select a quotient-completion limit, or assert trichotomy.

Definition 8441.1 (Bishop located upper-cut criterion carrier)**.** *A* $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, A, D, I, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-cut row, listing the finite lower observations used by the criterion and the downward-closure witnesses actually opened by the request.*
2. *A is the upper-cut row, listing the finite upper observations, upper stability cells, and upper-side witnesses used by the criterion.*
3. *D is the locatedness witness row. For each displayed interval request $a < b$, it records the local decision that either a lower observation lies below b or an upper observation lies above a.*
4. *I is the dyadic interval-test row over* $\mathsf{DyadicRatCore}^{\uparrow}$*, carrying endpoint cells, radius cells, comparison slack, and the finite gap requests consumed by D.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row, exposing the regular rational readback that compares the cut observations against the displayed dyadic tests.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after L, A, D, I, Q have been displayed.*
7. *S records the handoff route from the regular-Cauchy located-cut surface to the terminal Real seal.*

8. *H stores componentwise* hsame *transport for lower rows, upper rows, locatedness witnesses, dyadic tests, regular-rational compatibility, Real sealing, replay, provenance, and local naming.*
9. *C stores displayed* Cont *replay for the route*

$$L, A, D, I, Q \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$, $\mathsf{LocatedCutReal}^{\uparrow}$, $\mathsf{LocatedDedekindCut}^{\uparrow}$, $\mathsf{RegularCauchyLocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, A, D, I, Q, R, S, H, C, P, N$. It has no coordinate for quotient completion, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8441.2 (Bishop located upper-cut criterion obligations)**.** *For every accepted* BishopLocatedUpperCutCrite *carrier $U = (L, A, D, I, Q, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}}$ *surface consists exactly of finite lower-cut row exposure, finite upper-cut row exposure, locatedness-witness exposure, dyadic interval-test exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may cite the located-cut real packet, the located Dedekind cut packet, and the regular-Cauchy located-cut packet only through those displayed rows.*

Proof. Project the accepted carrier. The cut-facing data are exactly the lower row L, upper row A, and locatedness row D.

The dyadic and regular-rational rows then determine the numerical read. The interval tests I provide the endpoint and slack ledger, while Q supplies the regular-rational compatibility row read by the Real-facing consumer.

Finally R, S, H, C, P, N seal, route, transport, replay, source, and name the same finite packet. Since no quotient completion, selected limit, trichotomy branch, countable-choice schedule, or ambient-completeness witness is a coordinate of U, those objects are not NameCert obligations. □

Theorem 8441.3 (Bishop located upper-cut criterion locatedness)**.** *Every accepted* BishopLocatedUpperCutCriterio *packet answers each displayed interval request $a < b$ only through the locatedness row D and the dyadic interval-test row I. The answer may expose a lower-side observation below b or an upper-side observation above a, but it may not decide all cuts by trichotomy, choose an external representative, or replace the displayed interval test by host real order.*

Proof. Begin with the locatedness row D in the carrier. Its entries are indexed by the interval requests actually displayed by the packet.

The dyadic interval-test row I supplies the endpoint and slack data for those requests, so every locatedness read remains tied to finite dyadic cells. Transport and replay through H and C preserve only those indexed cells.

Thus the packet gives a local locatedness criterion. It does not contain a global order decision, a representative selector, or a host real comparison row, so none can be exported by an accepted read. □

Theorem 8441.4 (Bishop located upper-cut criterion Real-seal handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ *carrier reaches the terminal seal only through*

$$\mathsf{LocatedCutReal}^{\uparrow} \longmapsto \mathsf{LocatedDedekindCut}^{\uparrow} \longmapsto \mathsf{RegularCauchyLocatedCut}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The handoff exposes finite lower and upper cut rows, locatedness witnesses, dyadic interval tests, regular-Cauchy compatibility, Real sealing, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert data; it exports no Dedekind real construction, quotient completion, selected limit, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the obligation theorem to expose $L, A, D, I, Q, R, S, H, C, P, N$. The lower and upper rows supply the cut presentation, while D and I provide the displayed located interval tests.

The regular-Cauchy and regular-rational reads then pass through Q before the Real seal R is available. The handoff row S names the same route through the located-cut and regular-Cauchy located-cut siblings cited above.

The excluded endpoints are absent from the carrier inventory. Therefore a Real or Completion consumer receives only the displayed finite criterion route and cannot read a Dedekind construction, quotient completion, selected limit, trichotomy branch, countable-choice schedule, or ambient completeness principle from this packet. □

Falsifiable prediction: A $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ consumer that reaches a Real or Completion handoff without displaying the lower row, upper row, locatedness witnesses, dyadic interval tests, regular-rational compatibility, Real seal, transport, replay, provenance, and local naming exits the accepted carrier.

Independence witness: located-cut-real, located-dedekind-cut, regular-cauchy-located-cut, dyadicratcore, regseqrat, real: $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds lower and upper cut rows, locatedness witnesses, dyadic interval tests, regular-Cauchy compatibility, terminal Real sealing, transport, replay, provenance, and one local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}$ packet is generated from finite lower and upper cut rows, locatedness witnesses, dyadic interval tests, RegSeqRat compatibility, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8441.1, Definition 8441.2, Definition 8441.3, and Definition 8441.4 seed the finite Bishop located upper-cut criterion route for regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedUpperCutCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind real construction, quotient completion, selected limits, trichotomy, countable choice, ambient RealUp completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locatedness witnesses, dyadic interval-test exactness, RegSeqRat compatibility, Real-seal ordering, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.BishopLocatedUpperCutCriterionUp.

Chapter 8442

A Concrete Naming Certificate for $\mathsf{FiniteDyadicBisection}^{\uparrow}$

Following Bishop–Bridges constructive bisection and nested interval practice, the $\mathsf{FiniteDyadicBisection}^{\uparrow}$ packet records a finite dyadic interval-halving computation as BEDC NameCert data. It sits after the dyadic endpoint rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested dyadic fusion route of chapter 8365. The packet supplies Real and Completion arguments with a finite bisection carrier without importing open-cover compactness, a selected host limit, quotient streams, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8442.1 (Finite dyadic bisection carrier)**.** *A* $\mathsf{FiniteDyadicBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, D, M, L, R, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval-chain row, recording the input dyadic endpoint pair and each retained dyadic child interval.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger, storing the displayed endpoint comparisons, width budget, and dyadic denominator data for every row of I.*
3. *M is the midpoint split row, computed from the current dyadic endpoints through the same ledger D.*
4. *L is the selected left-child row when the displayed finite test keeps the left half.*
5. *R is the selected right-child row when the displayed finite test keeps the right half.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the endpoint, midpoint, and selected-child observations.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row formed from those finite windows and dyadic tolerance cells.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, D, M, L, R, S, Q have been displayed.*
9. *H records componentwise* hsame *transport for the interval chain, dyadic ledger, midpoint split, selected child row, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite bisection request through I, D, M, L, R, S, Q, E and then through the structural rows.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteDyadicBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, M, L, R, S, Q, E, H, C, P, N$. It has no coordinate for an infinite bisection tree, quotient stream equality, a selected completed real, open-cover compactness, trichotomy, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteDyadicBisectionUp.FiniteDyadicBisectionTasteGate single carrier alignment`

Theorem 8442.2 (Finite dyadic bisection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDyadicBisection}^{\uparrow}$ *carrier*

$$B = (I, D, M, L, R, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDyadicBisection}^{\uparrow}}$ *surface consists exactly of finite interval-chain exposure through I, dyadic endpoint and width exposure through D, midpoint split exposure through M, selected-child exposure through L or R, finite observation scheduling through S, regular rational readback through Q, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8442.1. The projection exposes the finite interval chain, dyadic ledger, midpoint split, selected child rows, finite observation window, regular rational readback, Real seal, transport, replay, provenance, and naming rows.

The analytic rows are ordered. The endpoint ledger D can be read only after the current finite interval row I is displayed, and the midpoint split M is computed from that ledger.

The selected-child rows are finite alternatives inside the same packet. The left row L and right row R are not host choices; they are displayed child rows whose source endpoint and midpoint data remain in I, D, M.

The downstream rows S, Q, E then expose only finite windows, regular rational readback, and the terminal Real seal. The structural rows H, C, P, N transport, replay, source, and name the same route, so the listed rows exhaust the local certificate. □

Theorem 8442.3 (Finite dyadic bisection induction route)**.** *Every finite induction over an accepted* $\mathsf{FiniteDyadicBisection}^{\uparrow}$ *packet proceeds by the displayed route*

$$I, D \longmapsto M \longmapsto L, R \longmapsto S, Q, E, H, C, P, N.$$

The base step reads the initial dyadic endpoint row and the first width budget. Each successor step consumes the current endpoint row, computes the midpoint split, selects one displayed child row, and then replays the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *downstream route. The induction exports no infinite branch selector, quotient stream equality, selected host limit, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use induction on the finite interval-chain row I. In the base case, the packet displays the initial dyadic endpoints and width budget through D, so the downstream rows cannot be read before that finite row is visible.

For the successor step, assume the current finite prefix has been read through I, D. The midpoint row M is computed from the displayed dyadic endpoints, and the retained child is recorded by one of the finite rows L or R.

The observation and completion-facing rows are downstream from this selected child. The $\mathsf{StreamName}^{\uparrow}$ window S opens only the endpoint and midpoint observations named by the finite step, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q reads that window, and the $\mathsf{Real}^{\uparrow}$ seal E is consumed only after Q is displayed.

Finally H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Since the carrier contains no infinite branch selector, quotient stream, selected limit, compactness principle, or completeness coordinate, the induction remains a finite bisection route. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDyadicBisection}^{\uparrow}$ Real-facing read exposes a finite dyadic interval chain, DyadicRatCore endpoint ledger, midpoint split, selected child row, StreamName finite window, RegSeqRat readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before the terminal seal is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, nested-dyadic-interval-fusion: $\mathsf{FiniteDyadicBisection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite dyadic interval-chain, midpoint split, selected child row, downstream readback, Real seal, transport, replay, provenance, and local naming as one finite bisection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDyadicBisection}^{\uparrow}$ packet is generated from finite dyadic interval-chain rows, DyadicRatCore endpoint and width rows, midpoint split rows, selected child rows, StreamName finite observations, RegSeqRat readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8442.1, Definition 8442.2, and Definition 8442.3 seed the finite dyadic bisection route from displayed dyadic endpoint data to a Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDyadicBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite branch selection, quotient stream equality, selected host limits, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, midpoint splitting, selected-child stability, StreamName scheduling, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDyadicBisectionUp.

Chapter 8443

A Concrete Naming Certificate for $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ packet records the finite branch tree by which Real and Completion arguments may inspect finitely many possible local choices without importing a countable-choice schedule. It sits after the finite stream observations of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the regular Cauchy criterion surface of chapter 1611, and the terminal real-name seal of chapter 38. The packet gives diagonal, tail, and criterion arguments a named finite-choice surface; it does not choose an infinite branch, select a limit, quotient streams, decide trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8443.1 (Finite-window choice-tree carrier)**.** *A* $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, W, R, D, G, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite branch-tree row, listing the root, the displayed child cells, the finite depth bound, and the local branch labels opened by the request.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite stream observations attached to each displayed branch cell.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic radii, comparison cells, and finite precision budget attached to each branch cell.*
5. *G is the* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *row, exposing only the criterion checks that were requested on the displayed windows and dyadic tolerances.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, W, R, D, G have been displayed.*
7. *H records componentwise* hsame *transport for branch cells, window rows, readback rows, dyadic tolerances, criterion checks, Real sealing, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for the finite route*

$$B, W, R, D, G \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowChoiceTree}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, W, R, D, G, E, H, C, P, N$. It has no coordinate for a countable-choice schedule, an infinite selected branch, quotient stream equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8443.2 (Finite-window choice-tree NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ *carrier $T = (B, W, R, D, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowChoiceTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier formation, requiring the branch tree B to be finite and every exposed child cell to be named inside the same* BHist *packet;*
2. *branch-window stability, requiring every read of a branch cell to pass through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row W and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R;*
3. *dyadic tolerance exposure, requiring each criterion check to expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D before it may read the* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *row G;*
4. hsame *and* Cont *transport, requiring H and C to preserve the same finite branch, window, readback, tolerance, criterion, and Real-seal rows under replay;*
5. *ledger refusal, requiring the packet to refuse countable choice, infinite selected branches, selected limits, quotient streams, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as available coordinates.*

Proof. Project the accepted carrier of Definition 8443.1. Its branch data are exactly the finite row B, so the packet begins with named finite cells rather than an infinite choice object.

The analytic observations then pass through W, R, D, G. The stream windows and regular rational readbacks supply the observed values, while the dyadic row supplies the precision ledger consumed by the regular Cauchy criterion row.

The Real-facing read is downstream from the same rows. The seal E is not available until the branch tree, windows, readbacks, dyadic tolerances, and criterion checks have all been displayed.

Finally H, C, P, N transport, replay, source, and name the finite route. Since the carrier contains no coordinate for countable choice, selected limits, quotient streams, trichotomy, or ambient Real completeness, those objects cannot become local NameCert obligations. □

Theorem 8443.3 (Finite-window choice-tree Real handoff)**.** *Every Real or Completion read exported by an accepted* $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The finite branch tree only indexes which displayed windows, readbacks, dyadic tolerances, and criterion rows are consumed; it does not provide a global branch selector or a completeness theorem.

Proof. Start with the branch tree row B. It indexes a finite family of displayed cells and supplies no data outside that finite family.

For each exposed cell, read the rows W, R, D, G in order. These rows give the stream window, regular rational readback, dyadic tolerance, and criterion check used by the Real-facing route.

The seal row E is then reached through the same displayed data, and H, C, P, N preserve the route under transport, replay, provenance, and local naming. No separate selector or completeness coordinate is present, so no such object can be exported. □

Falsifiable prediction: A $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ consumer that exports a Real or Completion read while omitting the finite branch tree, stream windows, RegSeqRat readbacks, dyadic tolerance row, regular Cauchy criterion row, Real seal, transport, replay, provenance, or local naming exits the accepted carrier.

Independence witness: streamname, regseqrat, dyadicratcore, regularcauchycriterion, real: $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a finite branch tree, displayed stream windows, regular rational readbacks, dyadic tolerance exposure, criterion checks, Real sealing, transport, replay, provenance, and one local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowChoiceTree}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowChoiceTree}^{\uparrow}$ packet is generated from a finite branch tree, displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RegularCauchyCriterion checks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowChoiceTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8443.1, Definition 8443.2, and Definition 8443.3 seed the finite branch-window route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowChoiceTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, infinite selected branches, selected limits, quotient streams, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-window stability, dyadic tolerance exposure, regular Cauchy criterion handoff, Real seal nonescape, hsame transport, Cont replay, provenance, and local naming for BEDC.Derived.FiniteWindowChoiceTreeUp.

Chapter 8444

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$

Following Bishop and Bridges on located real sequences and regular Cauchy real names, the $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ packet records the finite calibration row that lets a located real-sequence consumer pass from inspected stream windows to regular rational readback and then to a terminal real-name seal. It sits beside the finite stream windows of chapter 126, the regular rational route of chapter 301, the terminal real-name surface of chapter 38, the Bishop located sequence packet of chapter 7766, and the located real sequence source of chapter 1984. The calibration is finite L10 data; it is not quotient equality of streams, selected convergence, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8444.1 (Bishop located real sequence calibration carrier)**.** *A* $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of inspected finite sequence windows.*
2. *B is the* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *row naming the Bishop-style located sequence surface that supplies the locatedness request.*
3. *L is the* $\mathsf{LocatedRealSequence}^{\uparrow}$ *row recording the located real-name component windows used by the same request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the inspected windows and locatedness data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S, B, L, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the regular rational readback has been displayed.*
7. *H records componentwise* hsame *transport for S, B, L, D, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from a calibration request through the same finite rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BishopLocatedSequence}^{\uparrow}$, $\mathsf{LocatedRealSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}}$ *row.*

The classifier is rowwise over $S, B, L, D, R, E, H, C, P, N$. It compares only displayed stream windows, located-sequence rows, located-real component windows, dyadic tolerances, regular rational readbacks, real seals, transport, replay, provenance, and local naming.

Theorem 8444.2 (Bishop located real sequence calibration obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ *carrier* $K = (S, B, L, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRea}}$ *surface consists of five finite obligation rows:*

1. *stream-window admission, requiring the consumer to begin at the displayed* $\mathsf{StreamName}^{\uparrow}$ *row* S*;*
2. *located-sequence alignment, requiring the* $\mathsf{BishopLocatedSequence}^{\uparrow}$ *row* B *and the* $\mathsf{LocatedRealSequence}^{\uparrow}$ *row* L *to describe the same inspected request;*
3. *tolerance calibration, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *to bind the displayed windows and locatedness rows before any rational readback is consumed;*
4. *readback and seal routing, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to precede the terminal* $\mathsf{Real}^{\uparrow}$ *row* E*;*
5. *structural replay, requiring* H, C, P, N *to preserve the same finite route under componentwise transport, continuation replay, provenance, and local naming.*

No obligation row supplies quotient stream equality, selected convergence, trichotomy, countable choice, host real equality, or ambient real completeness.

Proof. Carrier admission is projection from Definition 8444.1. The analytic rows are S, B, L, D, R, E, and the structural rows are H, C, P, N.

The alignment obligations bind the two sequence views. The $\mathsf{BishopLocatedSequence}^{\uparrow}$ row records the Bishop locatedness route, while the $\mathsf{LocatedRealSequence}^{\uparrow}$ row records the located real-name component windows for the same inspected request.

The tolerance and readback rows are downstream. The dyadic ledger D calibrates the displayed windows before the regular rational row R is read, and the terminal seal E is reached only after that readback.

Finally H, C, P, N transport, replay, source, and name the same route. Since none of the excluded principles appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8444.3 (Bishop located real sequence calibration L10 route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BishopLocatedSequence}^{\uparrow} \longmapsto \mathsf{LocatedRealSequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Re}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. The route is a calibration surface between finite windows, located sequence data, regular rational readback, and the real seal; it is not a completeness theorem or a quotient-real equality.*

Proof. Start with the obligation rows of Definition 8444.2. They force each consumer to expose the stream window row, the two located sequence rows, the dyadic tolerance ledger, the regular rational readback, and the terminal real seal in that order.

Use the rowwise classifier of Definition 8444.1. Componentwise transport preserves the displayed route, so a transported consumer cannot replace it by quotient streams or by a selected convergence witness.

The structural rows then close the finite replay. The route is sourced by P, replayed by C, transported by H, and named by N, so the consumer reaches only the displayed $\mathsf{Real}^{\uparrow}$ seal. □

Lean4 (statement) `BEDC.Derived.BishopLocatedRealSequenceCalibrationUp`

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ consumer displays StreamName windows, Bishop located sequence data, LocatedRealSequence component windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before it reads a Real-facing result.

Independence witness: streamname, regseqrat, real, bishoplocatedsequence, locatedrealsequence: $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ is not bijective to any listed sibling because it packages the calibration between located sequence data, located real-sequence rows, dyadic tolerance, rational readback, and the terminal real seal as one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}$ packet is generated from finite StreamName windows, BishopLocatedSequence data, LocatedRealSequence component windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8444.1, Definition 8444.2, and Definition 8444.3 seed the finite calibration route between located sequence data, regular rational readback, and a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealSequenceCalibration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected convergence, trichotomy, countable choice, host real equality, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Lean closure requires checked carrier admission, located-sequence alignment, dyadic tolerance calibration, RegSeqRat handoff, Real-seal nonescape, transport replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealSequenceC

Chapter 8445

A Concrete Naming Certificate for $\mathsf{FordCircle}^{\uparrow}$

Following Hardy and Wright on Farey fractions and the classical Ford-circle construction, the $\mathsf{FordCircle}^{\uparrow}$ packet records a reduced rational boundary point together with its tangent upper-half-plane circle as finite Farey and horoball NameCert data. It realizes the Farey-horoball branch of chapter 9463 beside the finite Farey sequence rows of chapter 1759, the Farey approximation route of chapter 8081, and the hyperbolic horoball carrier of chapter 7843. The packet names only displayed rational, radius, tangency, horoball, transport, replay, provenance, and local naming rows; it is not a quotient real circle, not a host Euclidean tangent theorem, and not a global hyperbolic-plane completion.

Definition 8445.1 (FordCircle carrier)**.** *A* $\mathsf{FordCircle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, D, R, T, U, H, C, P, N)$$

with the following displayed rows:

1. *Q is the reduced rational center row, naming coprime integer numerator and positive denominator data over the rational boundary line.*
2. *D is the positive denominator row, recording the denominator witness used by the circle radius and by Farey-neighbour tests.*
3. *R is the dyadic reciprocal-square radius row, storing the displayed radius corresponding to the classical value $1/(2D^2)$ as rational and dyadic readback data.*
4. *T is the tangency ledger. It records the finite Farey determinant tests that witness equality of the boundary tangency point and exact tangency between neighbouring Ford circles.*
5. *U is the upper-half-plane and horoball row. It places the circle as a boundary-tangent horoball read whose base is Q and whose height is read from R.*
6. *H records componentwise* hsame *transport for the reduced rational center, denominator, radius, tangency ledger, upper-half-plane or horoball row, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Farey or hyperbolic-boundary consumer through Q, D, R, T, U, H, P, N.*
8. *P is the* Pkg *provenance row over* $\mathsf{FareySequence}^{\uparrow}$*,* $\mathsf{HyperbolicHoroball}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FordCircle}^{\uparrow}}$ *row.*

The local classifier compares two accepted packets only through these rows. It has no coordinate for ambient Euclidean circle equality, real quotient centers, selected real representatives, a completed hyperbolic plane, or a host geometry theorem about all tangent circles.

Theorem 8445.2 (FordCircle NameCert obligations)**.** *For every accepted* $\mathsf{FordCircle}^{\uparrow}$ *carrier*

$$F = (Q, D, R, T, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FordCircle}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier formation for the reduced rational center, positive denominator, dyadic reciprocal-square radius, Farey tangency ledger, upper-half-plane or horoball row, transport, replay, provenance, and local naming data;*
2. *radius stability, requiring the radius row R to be transported together with the same denominator row D and not recomputed from an external real-circle representative;*
3. *Farey tangency exactness, requiring the tangency ledger T to expose the determinant and adjacency rows consumed by neighbouring-circle reads;*
4. *horoball embedding route, requiring the upper-half-plane row U to be read downstream of Q, D, R, T and upstream of the structural rows H, C, P, N;*
5. *nonescape from host real-circle equality, excluding ambient Euclidean circle equality, quotient real centers, selected representatives, global hyperbolic-plane completion, and an unlogged tangent-circle theorem.*

Proof. Project the carrier of Definition 8445.1. The rows $Q, D, R, T, U, H, C, P, N$ display the reduced rational center, denominator, radius, tangency data, horoball placement, componentwise transport, replay, provenance, and local name.

Radius stability is componentwise. The denominator row D and the dyadic reciprocal-square radius row R move together under H, and any consumer replay through C must read R as part of the same packet.

Farey tangency exactness is supplied by the ledger T. The packet records the determinant and adjacency data before a neighbouring-circle read can be consumed, so tangency is not imported from an ambient real-circle theorem.

Finally the horoball row U is downstream of the rational and radius rows. The package and NameCert rows P, N name this displayed route, while the absent host objects have no coordinate in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FordCircle}^{\uparrow}}$.

Constructive story: A $\mathsf{FordCircle}^{\uparrow}$ packet is generated from a reduced rational center, positive denominator, dyadic reciprocal-square radius, Farey tangency ledger, upper-half-plane or horoball row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FordCircle}^{\uparrow}}$, seed)

Scope closed: Definition 8445.1 and Definition 8445.2 seed the finite Ford-circle surface linking Farey rational data with hyperbolic horoball reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FordCircle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Euclidean circle equality, quotient real-center equality, global hyperbolic-plane completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius stability, Farey tangency exactness, horoball embedding, transport, replay, provenance, and local naming for BEDC.Derived.FordCircleUp.

Chapter 8446

A Concrete Naming Certificate for KolmogorovMaximalInequality$^{\uparrow}$

Following Kolmogorov and Bishop–Bridges finite probability estimates, the KolmogorovMaximalInequality$^{\uparrow}$ packet records the finite partial-sum maximal-window route behind Kolmogorov's maximal inequality. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the one-row Chebyshev tail carrier of chapter 6592, the finite Boole ledger of chapter 6696, the weak-law finite-average packet of chapter 4373, and the terminal real-name seal of chapter 38. The packet is a finite maximal-window certificate for displayed prefixes; it is not a countable-choice selector, almost-sure quotient, measure-completion object, host limit theorem, or ambient Real$^{\uparrow}$ completeness principle.

Definition 8446.1 (Kolmogorov maximal inequality carrier)**.** *A* KolmogorovMaximalInequality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\Omega, X, S, B, C, U, R, H, K_0, P, N)$$

with the following displayed rows:

1. Ω *is the* ProbSpace$^{\uparrow}$ *source row carrying the event classifier and total-mass normalization.*
2. X *is the finite* RandomVar$^{\uparrow}$ *prefix row, listing the displayed variables* $X_1, \ldots, X_n$ *over the same probability source.*
3. S *is the partial-sum window row, recording the displayed sums* $S_k = X_1 + \cdots + X_k$ *for* $1 \le k \le n$.
4. B *is the maximum-threshold row, naming the positive* Real$^{\uparrow}$ *threshold and the event* $\max_{1 \le k \le n} |S_k| \ge B$ *as a finite window over* S.
5. C *is the Chebyshev tail-row family, one displayed* ChebyshevInequality$^{\uparrow}$ *read for each stopped partial-sum window needed by the maximal estimate.*
6. U *is the* FiniteUnionBound$^{\uparrow}$ *ledger that joins the finitely many exposed tail rows into the maximal-window event bound.*
7. R *is the* Real$^{\uparrow}$ *bound seal reached only after* Ω, X, S, B, C, U *have been displayed.*
8. H *records componentwise* hsame *transport for the source, prefix, partial-sum, threshold, Chebyshev, finite-union, bound, replay, provenance, and local naming rows.*
9. K_0 *records displayed* Cont *replay of the finite maximal-window route.*

10. *P is the* Pkg *provenance over* KolmogorovMaximalInequality$^{\uparrow}$, ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, ChebyshevInequality$^{\uparrow}$, FiniteUnionBound$^{\uparrow}$, WeakLawLargeNumbers$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{KolmogorovMaximalInequality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, X, S, B, C, U, R, H, K_0, P, N$. It has no coordinate for countable choice, almost-sure quotient equality, measure completion, a host-side limiting theorem, an infinite stopping-time selector, or ambient real completeness.

Theorem 8446.2 (Kolmogorov maximal inequality NameCert obligations)**.** *For every accepted* KolmogorovMaximalInequality$^{\uparrow}$ *carrier*

$$K = (\Omega, X, S, B, C, U, R, H, K_0, P, N),$$

the local NameCert$_{\text{KolmogorovMaximalInequality}^{\uparrow}}$ *surface consists exactly of probability-source exposure through Ω, finite random-variable prefix exposure through X, partial-sum window exactness through S, maximum-threshold exposure through B, Chebyshev tail-row exposure through C, finite-union ledger exposure through U, terminal real-bound sealing through R, componentwise transport through H, displayed replay through K_0, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8446.1. The projection exposes the probability source, finite random-variable prefix, partial-sum windows, maximum-threshold row, Chebyshev tail rows, finite-union ledger, Real bound seal, transport, replay, provenance, and local naming.

The prefix rows are ordered. The partial sums in S can be read only from the displayed prefix $X_1, \ldots, X_n$ over the source Ω, and the maximal event in B is a finite event window over those displayed sums.

The tail rows are downstream from that window. Each row of C is a Chebyshev read for one displayed stopped partial-sum prefix, while U records the finite union-bound ledger over exactly those rows.

The endpoint R is therefore a real-bound seal for the finite maximal event named by B. The structural rows H, K_0, P, N transport, replay, source, and name the same route, so no unlisted measure completion, countable selector, almost-sure quotient, host limit, or ambient completeness coordinate remains. □

Theorem 8446.3 (Kolmogorov partial-sum window bound)**.** *Every accepted* KolmogorovMaximalInequality$^{\uparrow}$ *maximal-window read factors through the finite route*

$$\Omega, X, S, B \longmapsto C \longmapsto U \longmapsto R \longmapsto H, K_0, P, N.$$

For each displayed prefix length n and threshold row B, the route first opens the finite partial-sum windows $S_1, \ldots, S_n$, then applies the Chebyshev tail rows in C to the stopped window pieces, then composes those finitely many rows through the FiniteUnionBound$^{\uparrow}$ *ledger U, and only then exposes the* Real$^{\uparrow}$ *bound seal R. Thus the maximal-window estimate is available to weak-law, concentration, and later strong-law-facing consumers as finite BEDC data, not as an almost-sure, countable-choice, measure-completion, host-limit, or ambient-completeness principle.*

Proof. Use induction on the displayed prefix length in X. In the base case, the packet exposes the empty or one-term partial-sum window through Ω, X, S, and the threshold row B names the only finite maximal event that can be inspected.

For the successor step, extend the displayed prefix by one RandomVar$^{\uparrow}$ row. The new partial-sum row is appended to S, while the already displayed prefix sums remain available through the same componentwise hsame transport.

Now open the tail-row family C. Each stopped prefix event is a finite Chebyshev request over the same probability source and the same displayed partial-sum data; no countable family or stopping-time selector is introduced.

The finite-union ledger U then composes exactly the rows supplied by C. This is the Boole-inequality step of chapter 6696, so the maximal event remains a finite union of displayed tail events.

Finally read the $\mathsf{Real}^{\uparrow}$ seal R and the structural rows H, K_0, P, N. The seal is downstream from the finite prefix, Chebyshev rows, and union ledger, while the structural rows preserve the same route for weak-law and concentration consumers. Since the carrier contains no coordinate for almost-sure quotient equality, measure completion, countable choice, host limits, or ambient real completeness, the partial-sum maximal bound stays inside finite BEDC data. □

Falsifiable prediction: Every accepted $\mathsf{KolmogorovMaximalInequality}^{\uparrow}$ maximal-window read exposes a ProbSpace source row, finite RandomVar prefix, partial-sum window row, maximum-threshold row, Chebyshev tail rows, finite-union ledger, RealUp bound seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before the bound is consumed.

Independence witness: probspace, randomvar, chebyshev-inequality, finite-union-bound, weak-law-large-numbers, real: $\mathsf{KolmogorovMaximalInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite probability source, random-variable prefix, partial-sum maximal window, Chebyshev tail family, finite-union ledger, terminal real-bound seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovMaximalInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovMaximalInequality}^{\uparrow}$ packet is generated from a probability source row, finite random-variable prefix, partial-sum window rows, a maximum-threshold row, Chebyshev tail rows, a finite-union ledger, a RealUp bound seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovMaximalInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8446.1, Definition 8446.2, and Definition 8446.3 seed the finite partial-sum maximal-window route over probability, random-variable, Chebyshev, finite-union, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovMaximalInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, countable-choice selection, measure completion, quotient event equality, host limiting theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix exposure, partial-sum window exactness, maximum-threshold exposure, Chebyshev tail-row stability, finite-union ledger composition, Real bound nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovMaximalInequalityUp.

Chapter 8447

A Concrete Naming Certificate for $\mathsf{OrderConvexSubspace}^{\uparrow}$

Following Bourbaki and Bishop-style ordered-space practice, the $\mathsf{OrderConvexSubspace}^{\uparrow}$ packet records an order-convex subspace as finite NameCert data over a constructive ordered carrier. It sits after the relational source of chapter 52, the antisymmetric surface of chapter 53, the located interval boundary of chapter 1709, the constructive real order of chapter 1716, and the Real readback route of chapter 38. The packet gives interval and completion-facing consumers a named order-convex restriction surface without importing host subset equality, quotient subspaces, classical trichotomy, selected suprema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8447.1 (OrderConvexSubspace carrier)**.** *An* $\mathsf{OrderConvexSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (P, S, I, T, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the ordered source row, read through* $\mathsf{Preorder}^{\uparrow}$ *or* $\mathsf{Poset}^{\uparrow}$*, with its carrier, order relation, and classifier exposed before the subspace row is used.*
2. *S is the subset-membership row, recording the finite membership classifier for the inspected subspace without identifying it with host set equality.*
3. *I is the interval-closure row: whenever accepted elements a, b of S satisfy $a \leq x \leq b$ in the source order, the inspected element x is admitted by the same membership classifier.*
4. *T is the order-transport row, recording that componentwise* hsame *transport preserves source order comparisons and the subset-membership classifier.*
5. *R is the interval handoff row to* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, exposing only the located interval windows and regular rational readbacks needed by Real-facing consumers.*
6. *H records componentwise* hsame *transport for P, S, I, T, R and the structural rows below.*
7. *C records displayed* Cont *replay through the ordered source, membership classifier, interval-closure row, order transport, and Real or RegSeqRat handoff.*
8. *Q is the* Pkg *provenance over* $\mathsf{OrderConvexSubspace}^{\uparrow}$*,* $\mathsf{Preorder}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*,* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{CauchyRealOrder}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OrderConvexSubspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient subset equality, host carrier equality, trichotomy of the source order, selected endpoints, or ambient real completeness.

Theorem 8447.2 (OrderConvexSubspace NameCert obligations)**.** *For every accepted* $\mathsf{OrderConvexSubspace}^{\uparrow}$ *carrier* $O = (P, S, I, T, R, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OrderConvexSubspace}^{\uparrow}}$ *surface consists exactly of carrier admission for the ordered source, subset-membership classifier, interval-closure row, order-transport row, Real or RegSeqRat interval handoff, componentwise transport, displayed replay, provenance, and local naming rows. No obligation may be discharged by quotienting the subset, replacing membership by host set equality, invoking classical trichotomy, selecting endpoints externally, or importing ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8447.1. The source row P gives the displayed preorder or poset data, while S gives the finite membership classifier for the inspected subspace.

The interval row I is the only closure coordinate. It admits an inspected middle point only through the displayed comparisons $a \leq x \leq b$ and the accepted endpoint memberships, so no host subset equality or quotient subspace can discharge the local obligation.

The transport and handoff rows T, R, H, C, Q, N keep the same source order, membership classifier, interval closure, Real or RegSeqRat readback, replay, provenance, and naming surface visible to consumers. Since trichotomy, endpoint selection, and ambient completeness are absent from the packet, they cannot serve as NameCert obligations. □

Theorem 8447.3 (OrderConvexSubspace interval transport)**.** *Every accepted interval read through an* $\mathsf{OrderConvexSubspace}^{\uparrow}$ *packet factors through*

$$P, S, I, T \longmapsto R \longmapsto H, C, Q, N.$$

Thus a Real or RegSeqRat interval consumer may use the subspace only after reading the ordered source, subset-membership classifier, interval-closure row, order-transport row, terminal interval handoff, transport, replay, provenance, and local NameCert data.

Proof. Start with the ordered source and membership rows. The row P exposes the $\mathsf{Preorder}^{\uparrow}$ or $\mathsf{Poset}^{\uparrow}$ comparisons, and S exposes the subspace membership classifier.

Apply the interval-closure row I and order-transport row T. Together they make the middle-point admission invariant under the displayed source-order comparisons and componentwise transport.

Finally read the handoff row R before the structural rows H, C, Q, N. The interval consumer therefore receives a finite BEDC route from ordered membership to Real or RegSeqRat windows, not a host theorem about convex subsets or completed ordered spaces. □

Falsifiable prediction: An $\mathsf{OrderConvexSubspace}^{\uparrow}$ read that omits the subset-membership classifier, interval-closure row, order-transport row, or Real/RegSeqRat handoff is not accepted by the local NameCert.

Independence witness: preorder, poset, located-real-interval, cauchy-real-order, regseqrat, real: $\mathsf{OrderConvexSubspace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the ordered source, subset-membership classifier, interval-closure row, order transport, interval readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{OrderConvexSubspace}^{\uparrow}}$**.**

Constructive story: An OrderConvexSubspace$^{\uparrow}$ packet is generated from an ordered source row, subset-membership classifier, interval-closure row, order-transport row, Real or RegSeqRat interval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{OrderConvexSubspace}^{\uparrow}}$, seed)

Scope closed: Definition 8447.1, Definition 8447.2, and Definition 8447.3 seed the finite order-convex subspace route through ordered source, membership, interval closure, order transport, interval handoff, structural replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{OrderConvexSubspace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient subset equality, host carrier equality, trichotomy, selected endpoints, ambient Real$^{\uparrow}$ completeness, Lean theorem checking, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subset-membership stability, interval-closure routing, order-transport stability, Real or RegSeqRat interval handoff, transport stability, replay stability, provenance, and local naming for BEDC.Derived.OrderConvexSubspaceUp.

Chapter 8448

A Concrete Naming Certificate for BishopCauchyModulusCalculus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopCauchyModulusCalculus$^{\uparrow}$ packet records the finite calculus used to compose, meet, shift, and replay displayed Cauchy moduli before a regular-Cauchy real-name consumer reads a seal. It sits beside the dyadic tolerance core of chapter 291, the finite-window name surface of chapter 126, the regular rational readback of chapter 301, the checked arithmetic packet of chapter 1485, and the terminal real-name surface of chapter 38. The packet names a local L10 tool for regular-Cauchy modulus arithmetic; it does not assert quotient equality of real names, a selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real$^{\uparrow}$ completeness.

Definition 8448.1 (Bishop Cauchy modulus calculus carrier)**.** *A* BishopCauchyModulusCalculus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, A, M, J, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, recording the dyadic radii, comparison cells, and finite precision budget used by the calculus.*
2. *S is the* StreamName$^{\uparrow}$ *finite-window row whose observed windows provide the source data for each modulus read.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from those windows.*
4. *A is the* CauchyModulusArithmetic$^{\uparrow}$ *arithmetic row, exposing the sum, product, diagonal, and shared-threshold operations already accepted by that sibling packet.*
5. *M is the displayed meet row, recording the common threshold schedule used to compare two modulus requests in the same dyadic ledger.*
6. *J is the shift row, recording finite index shifts and replay offsets for the displayed windows.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row, reachable only after D, S, R, A, M, J have been displayed.*
8. *H records componentwise* hsame *transport for the tolerance, window, readback, arithmetic, meet, shift, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for each finite calculus operation through D, S, R, A, M, J, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalculus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, A, M, J, E, H, C, P, N$. No coordinate stores a completed real, quotient equality of names, a selected limit, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8448.2 (Bishop Cauchy modulus calculus NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$ *carrier*

$$K = (D, S, R, A, M, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalculus}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite StreamName windows, regular rational readback, Cauchy modulus arithmetic row, displayed meet row, finite shift row, Real seal row, componentwise transport, displayed replay, provenance, and local naming. The obligation surface requires every compose, meet, shift, and replay read to expose the same dyadic ledger D, window row S, readback row R, arithmetic row A, meet row M, shift row J, and structural rows H, C, P, N before the seal E is public.*

Proof. Project the carrier of Definition 8448.1. Its displayed coordinates are exactly $D, S, R, A, M, J, E, H, C, P, N$, so carrier admission and local naming have no additional source.

The arithmetic read is finite before it is Real-facing. The dyadic ledger D fixes the tolerances, the window row S fixes the inspected observations, and the regular rational readback R supplies the values used by the modulus operations.

The calculus rows are downstream of those data. The arithmetic row A supplies the accepted sum, product, diagonal, and shared-threshold operations from the Cauchy modulus arithmetic packet; the meet row M and shift row J record only the displayed threshold and index-offset data needed to replay those operations.

Finally the seal row E is read only through H, C, P, N. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, trichotomy, compactness, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8448.3 (Bishop Cauchy modulus calculus operation laws)**.** *Every accepted* BishopCauchyModulusCalculus *packet exposes compose, meet, shift, and replay laws only along the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyModulusArithmetic}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The compose and meet laws reuse the same dyadic tolerance ledger and shared-threshold row; the shift law reindexes only displayed finite windows; the replay law transports the same rows through hsame, Cont, Pkg, *and the local* NameCert *row. None of these laws exports quotient real equality, a selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.*

Proof. Begin with the NameCert obligations of Definition 8448.2. They require every calculus read to expose $D, S, R, A, M, J, H, C, P, N$ before a Real-facing seal is available.

For compose and meet, read A together with the meet row M. The arithmetic row supplies the accepted operation surface, while M forces both modulus requests to share the displayed dyadic threshold.

For shift, read J against the same S and R rows. The shift changes only the finite index window and replay offset; it does not create a new stream, a quotient name, or a selected limit.

For replay, use H, C, P, N. Componentwise transport and displayed continuation replay preserve the same finite route, and provenance plus local naming keep the calculus laws inside the accepted packet. □

Theorem 8448.4 (Bishop Cauchy modulus calculus L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$ *packet must read the dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Cauchy modulus arithmetic row, displayed meet row, finite shift row, componentwise transport, displayed replay, provenance, and local naming before using a regular-Cauchy modulus calculation. The handoff supplies a finite local tool for regular-Cauchy modulus arithmetic and does not close Real completeness, quotient equality of names, countable choice, selected limits, or an open-cover compactness principle.*

Proof. Start from the operation laws of Definition 8448.3. They place every compose, meet, shift, and replay read inside the displayed route through dyadic tolerances, finite windows, regular readback, arithmetic, transport, replay, provenance, and local naming.

The Real-facing consumer may then inspect the seal row E only after the same rows have been displayed. This keeps the calculation as an L10 handoff for regular-Cauchy modulus arithmetic rather than a theorem of ambient Real completeness.

The excluded principles have no row in the carrier. Quotient real equality, selected limits, countable choice, trichotomy, and open-cover compactness therefore cannot be extracted from the handoff. □

Falsifiable prediction: An accepted $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$ consumer is refuted if it uses a compose, meet, shift, replay, Real, or Completion read without displaying the dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, CauchyModulusArithmetic row, meet row, shift row, transport, replay, provenance, and local naming evidence.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-modulus-arithmetic, real: no listed sibling is bijective with $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$ because this packet jointly carries tolerance ledgers, finite windows, rational readback, modulus arithmetic, meet thresholds, shift offsets, Real sealing, transport, replay, provenance, and local naming as one finite calculus surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalculus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyModulusCalculus}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a CauchyModulusArithmetic row, displayed meet and shift rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalculus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8448.1, Definition 8448.2, Definition 8448.3, and Definition 8448.4 seed the finite L10 modulus-calculus route over dyadic tolerances, finite windows, regular readback, arithmetic, meet, shift, Real sealing, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusCalculus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window exposure, RegSeqRat readback exactness, arithmetic-row handoff, meet and shift replay, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyModulusCalculusUp.

Chapter 8449

A Concrete Naming Certificate for $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$

Following Banach and Bishop–Bridges, the $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$ packet records a separable Banach-space presentation whose dense data are inspected through finite stream windows before the completion service is read. It sits over the Banach carrier of **??**, the norm rows of chapter 89, the complete-metric boundary of chapter 128, the density route of chapter 1809, and the finite real-name infrastructure of chapter 126 and chapter 301. The packet names a constructive countably dense realization inside a complete normed carrier; it does not choose limits, quotient Cauchy classes, dense enumerants outside the displayed code, countable choice, Hahn–Banach data, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8449.1 (Finite-realized separable Banach space carrier)**.** *A* $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, N, M, D, S, R, E, H, C, P, L)$$

with the following displayed rows:

1. *B is the ambient* $\mathsf{BanachSpace}^{\uparrow}$ *carrier row, including its vector, norm, Cauchy, completion, and separated-comparison service.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *readback row used by dense-code and Cauchy-window consumers.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row induced by the Banach norm and exposed before completion-facing density is public.*
4. *D is the* $\mathsf{RealSeparability}^{\uparrow}$ *dense-code admission row, listing the accepted countable dense codes and the finite windows by which those codes are inspected.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for reading dense-code observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the same stream windows and norm tolerances.*
7. *E is the terminal real-name seal reached by the norm and density reads.*
8. *H records componentwise* hsame *transport for B, N, M, D, S, R, E and the local certificate rows.*
9. *C records displayed* Cont *replay for Banach, norm, dense-code, stream-window, regular-rational, real-seal, and completion reads.*

10. *P is the* Pkg *provenance over* FiniteRealizedSeparableBanachSpace$^{\uparrow}$, BanachSpace$^{\uparrow}$, Norm$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RealSeparability$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{FiniteRealizedSeparableBanachSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected dense enumeration, an external completion object, quotient real equality, selected Cauchy limits, Hahn–Banach functionals, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8449.2 (Finite-realized separable Banach space NameCert obligations)**.** *For every accepted* FiniteRealizedSeparableBanachSpace$^{\uparrow}$ *carrier*

$$F = (B, N, M, D, S, R, E, H, C, P, L),$$

the local NameCert$_{\text{FiniteRealizedSeparableBanachSpace}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier: the packet displays the* BanachSpace$^{\uparrow}$ *carrier B, its* Norm$^{\uparrow}$ *readback N, and the induced* CompleteMetric$^{\uparrow}$ *row M before any separability or completion read is available;*
2. *dense-code admission: the* RealSeparability$^{\uparrow}$ *row D admits only displayed dense codes, read through the finite* StreamName$^{\uparrow}$ *windows S and the* RegSeqRat$^{\uparrow}$ *rational readback R;*
3. *norm/Cauchy stability: componentwise transport H and replay C preserve the same norm, dense-code, stream-window, and regular-rational rows under every accepted Cauchy request;*
4. *completion seal: completion-facing consumers reach the terminal real-name seal E only after the Banach, complete-metric, dense-code, stream, and regular-rational rows have been displayed;*
5. *no-choice ledger: provenance P and local name L record that the packet exports no selected dense enumeration, countable-choice branch, quotient completion equality, selected limit, Hahn–Banach datum, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8449.1. The Banach, norm, and complete-metric rows provide the only complete normed-space data available to a consumer.

The dense-code obligation is then read through D, S, R. The separability row admits only displayed codes, the stream row supplies finite observation windows, and the regular rational row attaches those windows to norm tolerances before the terminal real-name seal is reached.

Stability is the role of H and C. Since they transport and replay the same displayed coordinates, they cannot introduce a second dense enumeration, a detached completion object, or an off-ledger Cauchy limit.

The completion seal is therefore the row E read after B, N, M, D, S, R. Provenance and local naming, P, L, close the no-choice ledger by recording the absence of selected dense enumerants, countable choice, quotient completion equality, Hahn–Banach data, and ambient real-completeness coordinates. □

Closure status for NameCert$_{\text{FiniteRealizedSeparableBanachSpace}^{\uparrow}}$.

Constructive story: A FiniteRealizedSeparableBanachSpace$^{\uparrow}$ packet is generated from an ambient Banach carrier, norm and complete-metric rows, displayed separability codes, finite stream windows, regular rational readback, a terminal real-name seal, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteRealizedSeparableBanachSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8449.1 and Definition 8449.2 bind the displayed carrier, dense-code admission, norm/Cauchy stability, completion seal, and no-choice ledger.

Formal verification: FormalStatus(NameCert$_{\text{FiniteRealizedSeparableBanachSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, a standard bridge, a selected dense enumeration, quotient Banach completion equality, Hahn–Banach, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Formal verification should encode the carrier rows and prove the five NameCert obligations without adding choice, quotient completion equality, selected limits, or ambient real-completeness coordinates.

Chapter 8450

A Concrete Naming Certificate for $\mathsf{Dvoretsky}^{\uparrow}$

Following Dvoretsky and Milman, the $\mathsf{Dvoretsky}^{\uparrow}$ packet records the finite-section surface by which a Banach-space carrier exposes a Hilbert-comparable finite-dimensional section with a displayed Banach–Mazur distortion ledger. It sits over the Banach-space source of **??**, the distance ledger of chapter 7979, the norm carrier of chapter 89, the Hilbert comparison surface of chapter 91, and the convex admissibility rows of chapter 208. The packet names only the finite BEDC handoff for a selected finite section and its distortion witness; it is not an ambient theorem asserting uniform Euclidean sections for every host Banach space.

Definition 8450.1 (Dvoretsky finite-section carrier)**.** *A* $\mathsf{Dvoretsky}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, N, S, D, H, C, P, L)$$

with the following displayed rows:

1. *B is the Banach and norm source row, read through the* $\mathsf{BanachSpace}^{\uparrow}$ *and* $\mathsf{Norm}^{\uparrow}$ *certificates.*
2. *N is the finite-dimensional section row, naming the finite subspace, coordinate window, and section inclusion under the same norm source.*
3. *S is the Hilbert comparison row, recording the finite Euclidean model and the comparison map to the selected section.*
4. *D is the* $\mathsf{BanachMazurDistance}^{\uparrow}$ *distortion ledger for the comparison between N and S.*
5. *H records componentwise* hsame *transport for the Banach source, section row, Hilbert comparison, distortion ledger, provenance, and local name.*
6. *C records displayed* Cont *replay from the Banach source through the finite section, Hilbert comparison, and distortion ledger.*
7. *P is the* Pkg *provenance over* $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{BanachMazurDistance}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{Dvoretsky}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host infinite-dimensional compactness theorem, selected global basis, quotient Banach space, unrestricted finite-codimensional projection, classical choice schedule, or ambient Euclideanization principle.

Theorem 8450.2 (Dvoretsky NameCert obligations)**.** *For the carrier of Definition 8450.1, the local* $\mathsf{NameCert}_{\mathsf{Dvoretsky}^{\uparrow}}$ *surface consists exactly of Banach and norm source exposure through* B*, finite-dimensional section exposure through* N*, Hilbert comparison through* S*, Banach–Mazur distortion readback through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* L*.*

Proof. Project the finite packet $Q = (B, N, S, D, H, C, P, L)$. The first four rows display the mathematical content: source normed Banach data, the finite section, the Hilbert comparison row, and the distortion ledger.

Read the structural rows next. The transport row H compares the displayed rows componentwise, while C replays only the route from source to section, from section to Hilbert comparison, and from comparison to distortion ledger.

Finally inspect P and L. They source and name the same finite packet, and neither supplies an additional compactness, basis-selection, quotient, choice, or global Euclideanization row. Hence the listed rows exhaust the local NameCert surface. □

Theorem 8450.3 (Dvoretsky finite-dimensional section handoff)**.** *Every accepted* $\mathsf{Dvoretsky}^{\uparrow}$ *consumer that reads a Hilbert-comparable finite section must first expose the Banach and norm source row* B*, the finite-dimensional section row* N*, the Hilbert comparison row* S*, the Banach–Mazur distortion ledger* D*, and the structural rows* H, C, P, L*. The handoff reaches downstream Banach, Hilbert, convex, or distortion consumers only through this finite route.*

Proof. Start from the carrier. A finite-section read is stored in N, but the row is attached to the Banach and norm source B and to the Hilbert comparison row S inside the same packet.

Apply the NameCert obligations of Definition 8450.2. They place the distortion ledger D after the section and comparison rows, and they keep H, C, P, L as preservation, replay, provenance, and naming data for the same route.

Thus a consumer cannot read a Hilbert-comparable section as a detached host subspace or as an unrecorded Euclidean model. It receives the displayed finite section only after B, N, S, D, H, C, P, L have been exposed. □

Theorem 8450.4 (Dvoretsky non-escape)**.** *An accepted* $\mathsf{Dvoretsky}^{\uparrow}$ *packet exports no host Banach-space compactness theorem, no selected global basis, no quotient representative, no unrestricted projection, no classical choice schedule, and no ambient assertion that all finite-dimensional sections are Euclidean. Every finite-section claim available from the packet is routed through* B, N, S, D, H, C, P, L*.*

Proof. Inspect the carrier coordinates. The only rows present are the Banach and norm source, finite section, Hilbert comparison, Banach–Mazur distortion ledger, transport, replay, provenance, and local naming rows.

Use Definition 8450.3. Any downstream consumer must pass through the displayed section and distortion handoff before using the finite Hilbert comparison.

The excluded principles are not rows of the packet. Since H, C, P, L only preserve and name the same finite route, no host compactness theorem, global basis selector, quotient representative, unrestricted projection, choice schedule, or ambient Euclideanization principle can escape from the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Dvoretsky}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Dvoretsky}^{\uparrow}$ packet is generated from a Banach and norm source row, a finite-dimensional section row, a Hilbert comparison row, a Banach–Mazur distortion ledger, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{Dvoretsky}\uparrow}$, seed)

Scope closed: Definition 8450.1, Definition 8450.2, Definition 8450.3, and Definition 8450.4 seed the finite-section NameCert surface over Banach, norm, Hilbert, convex, and Banach–Mazur distance rows.

Formal verification: FormalStatus(NameCert$_{\text{Dvoretsky}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close the full analytic Dvoretsky theorem, host compactness, global basis selection, quotient representatives, unrestricted projections, choice, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-section exposure, Hilbert comparison stability, Banach–Mazur distortion readback, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.DvoretskyUp.

Chapter 8451

A Concrete Naming Certificate for $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$

Following Bishop and Bridges, constructive diagonal arguments over regular rational names first synchronize dyadic tolerances and finite stream windows, then expose a regular readback before any real-name seal is available. The $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ packet records that finite L10 route as NameCert data. It sits after the dyadic tolerance rows of chapter 291, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the diagonal-limit route of chapter 6002, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with a finite dyadic diagonal schedule; it is not quotient stream equality, a selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8451.1 (Dyadic Cauchy diagonal carrier)**.** *A* $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (D_0, S, R, G, E, H, C, P, N)$$

with the following displayed rows:

1. D_0 *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule, recording the denominator scale, slack cells, and comparison ledger used by every diagonal read.*
2. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family, exposing only the source windows requested by the dyadic tolerance schedule.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed windows and tolerance ledger.*
4. G *is the regular Cauchy diagonal row, recording the synchronized diagonal request before the terminal real-name seal is visible.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after* D_0, S, R, G *have been displayed.*
6. H *records componentwise* hsame *transport for the dyadic schedule, finite windows, regular readback, diagonal row, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from each Real or Completion request through* D_0, S, R, G, E, H, P, N.
8. P *is the* Pkg *provenance over* $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D_0, S, R, G, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, trichotomy, host real equality, ambient completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DyadicCauchyDiagonalUp.DyadicCauchyDiagonalUp`

Theorem 8451.2 (Dyadic Cauchy diagonal NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ *carrier*

$$D = (D_0, S, R, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance schedule, finite stream windows, regular rational readback, regular Cauchy diagonal row, terminal Real seal, componentwise transport, displayed replay, package provenance, and local naming rows.*

Proof. Project the carrier of Definition 8451.1. The displayed tuple $D_0, S, R, G, E, H, C, P, N$ is the entire public surface, so carrier admission is exactly admission of those nine rows.

The analytic rows are ordered. The dyadic tolerance schedule D_0 determines which finite windows S may be opened, and the readback row R is assembled only from those windows and tolerance cells.

The diagonal row G then records the synchronized regular Cauchy request. Only after G is displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no further coordinate is present, no quotient stream, selected representative, countable-choice schedule, trichotomy row, host real equality, ambient completion object, or ambient real-completeness principle can become a local NameCert obligation. □

Theorem 8451.3 (Dyadic Cauchy diagonal RegSeqRat handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ *carrier first reads the finite handoff*

$$D_0, S \longmapsto R \longmapsto G \longmapsto E \longmapsto H, C, P, N.$$

Thus the diagonal route hands off a $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the regular Cauchy diagonal row and terminal real seal are available.*

Proof. Start with a consumer request. The carrier exposes no direct seal coordinate, so the request first enters the dyadic tolerance schedule D_0 and the finite-window row S.

Those rows determine the regular rational readback R. The diagonal row G is downstream of that readback, so it cannot be consumed until the RegSeqRat handoff is visible.

The real seal E and the structural rows H, C, P, N preserve this displayed route under transport, replay, provenance, and local naming. Therefore every Real or Completion consumer receives the finite handoff stated above. □

Theorem 8451.4 (Dyadic Cauchy diagonal nonescape)**.** *An accepted* $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ *carrier exports no quotient stream equality, selected representative, countable-choice schedule, trichotomy row, host real equality, ambient completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Inspect the carrier rows in Definition 8451.1. The only analytic rows are the dyadic tolerance schedule, finite stream windows, regular rational readback, regular Cauchy diagonal row, and terminal real seal.

The remaining rows are structural: transport, replay, provenance, and local naming. They can preserve or replay the displayed finite route, but they do not introduce a new analytic coordinate.

Any excluded object would need a readable row distinct from $D_0, S, R, G, E, H, C, P, N$. No such row exists in the carrier, so the excluded objects cannot escape through the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ Real or Completion read factors through a DyadicRatCore tolerance schedule, StreamName finite-window family, RegSeqRat readback, regular Cauchy diagonal row, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before the terminal seal is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, regular-cauchy-diagonal-limit, real: $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the dyadic tolerance schedule, finite windows, regular rational readback, diagonal route, real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyDiagonal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance schedule, StreamName finite windows, RegSeqRat readback, a regular Cauchy diagonal row, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8451.1, Definition 8451.2, Definition 8451.3, and Definition 8451.4 seed the finite dyadic diagonal route from tolerance schedules through RegSeqRat readback to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, dyadic tolerance exactness, RegSeqRat handoff, diagonal-row stability, Real seal non-escape, componentwise transport, displayed replay, and local naming for BEDC.Derived.DyadicCauchyDiagonalUp.

Chapter 8452

A Concrete Naming Certificate for $\mathsf{HairyBall}^{\uparrow}$

Following Brouwer and Hopf-style tangent-field obstruction theorems, the $\mathsf{HairyBall}^{\uparrow}$ packet records the finite BEDC surface by which a tangent vector-field obstruction on an even sphere is named. It reads the circle boundary discipline of chapter 58, the degree obstruction row of chapter 3745, the finite chart source of chapter 96, the tangent-bundle rows of chapter 210, and the terminal real seal of chapter 38. The packet stores only displayed rows; it does not import a host tangent field, quotient equality of sphere points, selected zeros, compactness by open covers, or an ambient classical degree theorem.

Definition 8452.1 (Hairy-ball carrier)**.** *A* $\mathsf{HairyBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, V, Z, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the sphere-source row, recording the displayed even-sphere chart and boundary data read through* $\mathsf{Manifold}^{\uparrow}$ *and the circle source of chapter 58.*
2. *T is the tangent-bundle row over S, read through* $\mathsf{VectorBundle}^{\uparrow}$ *and carrying only the displayed fibre and transition data needed for tangent reads.*
3. *V is the vector-field row, recording the finite local vector assignments and their compatibility with T.*
4. *Z is the zero-obstruction row, recording the finite ledger that blocks a nowhere-zero export from the displayed field data.*
5. *D is the Brouwer-degree and Euler-obstruction row, read through* $\mathsf{BrouwerDegree}^{\uparrow}$ *and used only as the finite obstruction ledger for the displayed sphere route.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the scalar comparison consumed by the obstruction.*
7. *H records componentwise* hsame *transports for S, T, V, Z, D, R and the structural rows.*
8. *C records the displayed* Cont *replay order by which consumers read the sphere source, tangent bundle, vector field, zero obstruction, degree obstruction, real seal, transports, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HairyBall}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HairyBall}^\uparrow}$ *row.*
The carrier has no coordinate for a host vector field, a selected zero, quotient equality of sphere points, unrestricted compactness, or a classical tangent-bundle theorem outside the displayed rows.

Lean4 (statement) `BEDC.Derived.HairyBallUp`

Theorem 8452.2 (Hairy-ball NameCert obligations)**.** *For every accepted* $\mathsf{HairyBall}^\uparrow$ *carrier* $Q = (S, T, V, Z, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HairyBall}^\uparrow}$ *surface consists exactly of sphere-source admission, tangent-bundle exposure, vector-field row exposure, zero-obstruction retention, Brouwer-degree and Euler-obstruction routing, terminal* $\mathsf{Real}^\uparrow$ *sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 8452.1. The rows S, T, V expose the sphere, tangent-bundle, and vector-field data before any obstruction is read.

The rows Z, D, R then bind the obstruction surface. The zero-obstruction row records the blocked nowhere-zero export, the Brouwer-degree and Euler row supplies the displayed obstruction ledger, and the terminal real row seals the scalar comparison consumed by that ledger.

Finally use H, C, P, N. Componentwise transport preserves the same displayed rows, continuation replay fixes their order, provenance names the sibling certificates, and the local NameCert row packages the finite surface. No missing coordinate can supply a host vector field, selected zero, quotient equality, compactness principle, or ambient degree theorem. □

Theorem 8452.3 (Hairy-ball tangent-field route)**.** *Every tangent-field consumer of an accepted* $\mathsf{HairyBall}^\uparrow$ *packet must factor through the route*

$$S \longmapsto T \longmapsto V \longmapsto Z \longmapsto D \longmapsto R \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before reading the displayed obstruction. The exported route is therefore a finite tangent-field obstruction packet over $\mathsf{Manifold}^\uparrow$*,* $\mathsf{VectorBundle}^\uparrow$*,* $\mathsf{BrouwerDegree}^\uparrow$*,* $\mathsf{Real}^\uparrow$*, and* $\mathsf{S}^{1\uparrow}$ *rows, not a host theorem selecting a zero or a nowhere-zero vector field.*

Proof. Start with S and T. The sphere-source row fixes the displayed chart and boundary data, and the tangent-bundle row fixes the fibre reads over that same source.

Next read V, Z, D, R. The vector-field row is consumed only as local data over T; the zero-obstruction and degree rows record the obstruction ledger; and the real seal is downstream of those rows rather than an independent host comparison.

The structural rows H, C, P, N preserve the same order. Transport, replay, provenance, and naming can repack the displayed packet, but they cannot introduce a selected zero, quotient equality of sphere points, an unrestricted compactness argument, or a classical tangent-field theorem outside the finite carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{HairyBall}^\uparrow}$**.**

Constructive story: A $\mathsf{HairyBall}^\uparrow$ packet is generated from a sphere-source row, tangent-bundle row, vector-field row, zero-obstruction row, Brouwer-degree and Euler-obstruction row, terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HairyBall}^\uparrow}$, seed)

Scope closed: Definition 8452.1, Definition 8452.2, and Definition 8452.3 seed the finite tangent-field obstruction route for HairyBallUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HairyBall}^\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim a selected zero, a nowhere-zero vector field, quotient equality of sphere points, unrestricted compactness, Lean verification of the obstruction theorem, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, tangent-bundle exposure, vector-field row exposure, zero-obstruction retention, degree-obstruction routing, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.HairyBallUp.

Chapter 8453

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley filter-base terminology, the FiniteCauchyFilterBasisRefine packet records the finite common-refinement layer used by the L10 Cauchy-filter route before located or regular completion consumers read the basis. It sits after the basis generator of chapter 1501 and the finite meet packet of chapter 1958, and before the located filter route of chapter 1661. Its readback support passes through the stream windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names a finite refinement packet only; it does not complete a quotient filter, select a limit, invoke countable choice, identify host equality, decide trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8453.1 (Finite Cauchy filter basis refinement carrier)**.** *A* $\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, R, D, M, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row whose finite basis elements are being compared.*
2. *R is the refinement request row. It names the finite pair or finite family of basis requests that must be jointly refined before a located or regular completion consumer may read them.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same requests, including the displayed radius and inclusion budget used to compare regular rational observations.*
4. *M is the finite meet and common-refinement witness, read through* $\mathsf{CauchyFilterMeet}^{\uparrow}$*, that names a displayed basis element refining every request recorded in R.*
5. *S is the support row carrying the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used by B, R, D, M.*
6. *H records componentwise* hsame *transport for B, R, D, M, S and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a refinement request through the support row S, the tolerance ledger D, the meet witness M, and then the downstream located or regular consumer boundary.*

8. *P is the* Pkg *provenance over* FiniteCauchyFilterBasisRefinement$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, CauchyFilterMeet$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{FiniteCauchyFilterBasisRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, D, M, S, H, C, P, N$. It has no coordinate for completed filters, quotient Cauchy names, selected representatives, countable-choice schedules, trichotomy decisions, host real equality, or ambient completeness of Real$^{\uparrow}$.

Theorem 8453.2 (Finite Cauchy filter basis refinement NameCert obligations)**.** *For every accepted* FiniteCauchyFilterBasisRefinement$^{\uparrow}$ *carrier $K = (B, R, D, M, S, H, C, P, N)$, the local* NameCert$_{\text{FiniteCauchyFilter}}$ *surface consists exactly of carrier admission for the source basis row, refinement request row, dyadic tolerance ledger, finite meet witness, stream and regular-rational support row, componentwise transport, displayed replay, provenance, and local naming. Every located or regular completion-facing consumer must first read B, R, D, M, S and then pass through H, C, P, N before any downstream filter or* Real$^{\uparrow}$ *seal is inspected.*

Proof. Project the carrier of Definition 8453.1. Its mathematical rows are exactly the source basis B, refinement request R, tolerance ledger D, meet witness M, and support row S.

The support row is not an independent completion principle. It only displays the StreamName$^{\uparrow}$ windows and RegSeqRat$^{\uparrow}$ readbacks that the basis, request, tolerance, and meet rows already use.

The structural rows H, C, P, N transport, replay, source, and name that route. Since no carrier coordinate stores a completed filter, quotient name, selected limit, choice schedule, trichotomy decision, host real equality, or ambient completeness proof, the listed rows exhaust the local NameCert obligations. □

Theorem 8453.3 (Finite Cauchy filter basis refinement common refinement)**.** *Let $K = (B, R, D, M, S, H, C, P, N)$ be an accepted* FiniteCauchyFilterBasisRefinement$^{\uparrow}$ *carrier. Every request listed in R has a displayed common refinement read through the meet row M, and that refinement is checked against the same stream windows, regular rational readbacks, and dyadic tolerance ledger carried by S and D. Thus a consumer receives a finite common-refinement certificate, not a completed filter object or a choice of limit.*

Proof. Read the request row R together with the basis row B. The accepted carrier requires these requests to be basis-level requests over the same displayed observation surface.

Apply the meet row M. It supplies the common-refinement witness through CauchyFilterMeet$^{\uparrow}$, so every listed request is refined by a displayed basis element before the consumer sees a downstream filter boundary.

Finally replay the witness through S, D, H, C, P, N. The stream windows, regular rational readbacks, and dyadic tolerances remain visible, while the structural rows add only transport, replay, provenance, and local naming. Hence the certificate is finite and does not select a completed filter or limit. □

Theorem 8453.4 (Finite Cauchy filter basis refinement Real seal boundary)**.** *Every* Real$^{\uparrow}$*-facing read exposed by an accepted* FiniteCauchyFilterBasisRefinement$^{\uparrow}$ *packet factors through the finite route*

CauchyFilterBasis$^{\uparrow}$, CauchyFilterMeet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, Real$^{\uparrow}$.

The route may hand a common-refinement certificate to located or regular completion consumers, but it cannot turn that certificate into quotient filter equality, selected convergence, countable choice, host equality of real names, trichotomy, or ambient completeness of Real$^{\uparrow}$.

Proof. Start from the NameCert obligations of Definition 8453.2. They force every downstream read to expose the basis, request, tolerance, meet, and support rows before structural replay is available.

Use common refinement from Definition 8453.3. The meet witness is a finite basis-level certificate checked against the same stream, regular rational, and dyadic rows; it is not a completed filter or a selected limit.

The only real-facing route in the provenance row passes through the located filter boundary and then the terminal $\mathsf{Real}^{\uparrow}$ seal. Since the carrier has no coordinate for quotient filters, selected convergence, choice, host equality, trichotomy, or ambient completeness, the Real-facing read remains a boundary handoff rather than a completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}$ consumer that claims a basis refinement must display a $\mathsf{CauchyFilterBasis}^{\uparrow}$ row, a refinement request row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{CauchyFilterMeet}^{\uparrow}$ witness, $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ support rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: cauchy-filter-basis, cauchy-filter-meet, located-cauchy-filter, stream-name, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}$ because this packet jointly binds source basis data, a refinement request, a dyadic tolerance ledger, a finite meet witness, readback support, downstream handoff provenance, transport, replay, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}$ packet is generated from a CauchyFilterBasis row, a finite refinement request row, a DyadicRatCore tolerance ledger, a CauchyFilterMeet common-refinement witness, StreamName and RegSeqRat support rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8453.1, Definition 8453.2, Definition 8453.3, and Definition 8453.4 seed the finite common-refinement packet between raw Cauchy filter bases and located or regular completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyFilterBasisRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement witness exactness, readback support preservation, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyFilterBasisRefinemen

Chapter 8454

A Concrete Naming Certificate for $\mathsf{PringsheimTauberian}^{\uparrow}$

Following Pringsheim on positive term series and Hardy's account of Tauberian theory, the $\mathsf{PringsheimTauberian}^{\uparrow}$ packet records the finite route by which a Pringsheim-positive real series may pass from Abel boundary data and a Tauberian side ledger back to ordinary real-series readback. It sits after the positive-series source of chapter 2080, the Pringsheim convergence surface of chapter 6261, the Hardy–Littlewood Tauberian handoff of chapter 1932, the Tauberian remainder boundary of chapter 1991, the Abel summation route of chapter 1763, the real-series source of chapter 1581, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is a finite NameCert surface for a positive-series Tauberian route; it is not an ambient theorem about arbitrary series, quotient series equality, selected limits, trichotomy, countable choice, open-cover compactness, or host $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8454.1 (Pringsheim-Tauberian carrier)**.** *A* $\mathsf{PringsheimTauberian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, C, A, B, T, M, R, E, H, K, P, N)$$

whose rows have the following roles.

1. *S is the* $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ *source row, displaying positive real-series coefficients and finite partial-sum windows.*
2. *C is the* $\mathsf{PringsheimConvergence}^{\uparrow}$ *convergence row, recording the ordinary Pringsheim-facing partial-sum readback admitted for the inspected positive series.*
3. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, displaying summation-by-parts and radial Abel boundary windows for the same source row.*
4. *B is the Abel boundary row supplied by the* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ *route.*
5. *T is the Tauberian side row, supplied either by the* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ *tail ledger or by the* $\mathsf{TauberianRemainder}^{\uparrow}$ *remainder ledger.*
6. *M is the comparison row tying the Abel boundary, the Tauberian side row, and the ordinary partial-sum readback to one finite tolerance request.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the ordinary series endpoint.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the ordinary readback.*

9. *H records componentwise* hsame *transport for positive-series rows, Abel rows, Tauberian side rows, ordinary readback, Real sealing, replay, provenance, and local naming.*
10. *K records displayed* Cont *replay from a requested precision through the Abel boundary and Tauberian side row to the ordinary readback and Real seal.*
11. *P is the* Pkg *provenance over* PringsheimTauberian$^{\uparrow}$, PringsheimPositiveRealSeries$^{\uparrow}$, PringsheimConvergence$^{\uparrow}$, HardyLittlewoodTauberian$^{\uparrow}$, TauberianRemainder$^{\uparrow}$, AbelSummation$^{\uparrow}$, RealSeries$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{PringsheimTauberian}^{\uparrow}}$ *row.*

Theorem 8454.2 (Pringsheim-Tauberian NameCert obligations)**.** *For every accepted* PringsheimTauberian$^{\uparrow}$ *carrier $Q = (S, C, A, B, T, M, R, E, H, K, P, N)$, the local* NameCert$_{\text{PringsheimTauberian}^{\uparrow}}$ *surface consists exactly of carrier admission for that twelve-row packet, positive-series exposure through S, Pringsheim convergence exposure through C, Abel summation exposure through A, Abel boundary exposure through B, Tauberian side exposure through T, comparison exposure through M, regular rational readback through R, Real sealing through E, and transport, replay, provenance, and local naming through H, K, P, N.*

Proof. Project the carrier of Definition 8454.1. Its analytic rows are precisely S, C, A, B, T, M, R, E, and its structural rows are precisely H, K, P, N.

Read the source rows first. The positive-series row S supplies the coefficients and finite partial sums, while C supplies the ordinary Pringsheim-facing convergence readback. The Abel row A and boundary row B then expose only radial and summation-by-parts data for the same source.

Next inspect the Tauberian rows. The side row T and comparison row M are the only coordinates that relate the Abel boundary to the ordinary readback, and R, E are the only endpoint coordinates. Componentwise transport, displayed replay, provenance, and local naming therefore preserve the same finite surface and supply no hidden host theorem. □

Theorem 8454.3 (Pringsheim-Tauberian Abel handoff)**.** *Every ordinary real-series endpoint exported by an accepted* PringsheimTauberian$^{\uparrow}$ *carrier factors through the finite route*

$$S, C \longmapsto A, B \longmapsto T, M \longmapsto R, E \longmapsto H, K, P, N.$$

Thus an Abel-facing read becomes an ordinary positive-series read only after the displayed Tauberian side row and comparison row have been replayed through the same hsame *and* Cont *surface.*

Proof. Begin with the positive source and ordinary convergence rows S, C. They display the partial-sum window and the Pringsheim-facing readback that any endpoint consumer must cite.

Then apply the Abel summation and Hardy–Littlewood boundary routes. The rows A, B expose Abel data, but they do not by themselves export ordinary convergence; they are boundary data for the Tauberian stage.

Finally use the Tauberian side row T and comparison row M. These rows connect the Abel boundary to the regular rational readback R and terminal Real seal E, while H, K, P, N transport, replay, source, and name the same finite packet. Hence the endpoint cannot be obtained from quotient series equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Falsifiable prediction: An accepted PringsheimTauberian$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ seal without first reading the PringsheimPositiveRealSeries$^{\uparrow}$ row, AbelSummation$^{\uparrow}$ row, Tauberian side row, comparison row, and RegSeqRat$^{\uparrow}$ readback row refutes the carrier surface.

Independence witness: pringsheim-positive-real-series, pringsheim-convergence, hardy-littlewood-tauberian, tauberian-remainder, abel-summation, realseries, regseqrat, real: no listed sibling supplies the whole twelve-row packet $Q = (S, C, A, B, T, M, R, E, H, K, P, N)$, because each sibling contributes only a positive-series source, convergence readback, Abel boundary, Tauberian side ledger, summation route, regular readback, seal face, or structural row while this chapter binds them into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PringsheimTauberian}\uparrow}$.

Constructive story: A PringsheimTauberian$^{\uparrow}$ packet is generated from a Pringsheim-positive real-series row, a Pringsheim convergence row, Abel summation rows, Abel boundary rows, Tauberian side-condition rows, comparison rows, RegSeqRat readback, a RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PringsheimTauberian}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8454.1, Definition 8454.2, and Definition 8454.3 seed the finite positive-series Tauberian route over Pringsheim, AbelSummation, HardyLittlewoodTauberian, TauberianRemainder, RegSeqRat, RealSeries, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PringsheimTauberian}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Tauberian theory, arbitrary real-series convergence, quotient series equality, selected limits, trichotomy, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the twelve-row carrier and prove the finite NameCert obligation surface, Abel handoff theorem, and Real seal nonescape for BEDC.Derived.PringsheimTauberianUp.

Chapter 8455

A Concrete Naming Certificate for $\mathsf{NewmanLemma}^{\uparrow}$

Following Newman on terminating abstract reduction systems, the $\mathsf{NewmanLemma}^{\uparrow}$ packet records the finite BEDC surface by which local confluence and normalization evidence can be routed toward confluence-facing MetaCIC consumers. It sits beside the traditional Church–Rosser packet of chapter 6198 and reads the MetaCIC normalization-confluence bridge of chapter 1190, the normalization frontier of chapter 892, and the parallel-diamond frontier of chapter 4238. The packet names a handoff from terminating local peaks to a bounded confluence route; it is not an unconditional Church–Rosser theorem, a subject-reduction theorem, quotient syntax equality, evaluator equality, or kernel consistency.

Definition 8455.1 (Newman lemma carrier)**.** *A* $\mathsf{NewmanLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (T, L, W, J, B, O, H, C, P, M)$$

with the following displayed rows:

1. *T is the terminating or strongly normalizing measure row for the displayed reduction source.*
2. *L is the local-confluence row, exposing only finite one-step peak data for the same source.*
3. *W is the well-founded descent ledger that forces every iterated peak analysis to consume a smaller displayed row.*
4. *J is the local join budget extracted from L under the descent row W.*
5. *B is the MetaCIC bridge row, read through* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *and carrying only normalization-confluence correspondence data.*
6. *O is the obstruction row recording that subject reduction, quotient syntax equality, host evaluator equality, and unrestricted Church–Rosser remain outside the packet.*
7. *H records componentwise* hsame *transport for T, L, W, J, B, O and the structural rows.*
8. *C records displayed* Cont *replay from a confluence-facing request through $T, L, W, J, B, O, H, P, M$.*
9. *P is the* Pkg *provenance over* $\mathsf{NewmanLemma}^{\uparrow}$, $\mathsf{ChurchRosser}^{\uparrow}$, $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *M is the local* $\mathsf{NameCert}_{\mathsf{NewmanLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, W, J, B, O, H, C, P, M$. There is no coordinate for a global confluence oracle, a subject-reduction discharge, quotient equality of terms, evaluator equality, or a consistency endpoint.

Theorem 8455.2 (Newman lemma NameCert obligations)**.** *For every accepted* $\mathsf{NewmanLemma}^{\uparrow}$ *carrier* $N = (T, L, W, J, B, O, H, C, P, M)$*, the local* $\mathsf{NameCert}_{\mathsf{NewmanLemma}^{\uparrow}}$ *surface consists exactly of termination evidence, local-confluence exposure, well-founded descent, local-join budgeting, MetaCIC bridge routing, obstruction retention, componentwise transport, displayed replay, provenance, and local naming through the displayed rows.*

Proof. Project the carrier of Definition 8455.1. The rows T, L, W, J expose the traditional Newman ingredients: termination, local confluence, descent, and local joins for the same displayed source.

The bridge row B then fixes the only MetaCIC-facing route. It reads normalization-confluence correspondence data through the existing bridge packet rather than inventing a detached confluence theorem.

Finally use O, H, C, P, M. The obstruction row records the missing global claims, while transport, replay, provenance, and local naming preserve the same finite route. Since no further coordinate is present, the NameCert surface cannot supply subject reduction, quotient term equality, evaluator equality, unrestricted Church–Rosser, or kernel consistency. □

Theorem 8455.3 (Newman local-confluence normalization handoff)**.** *Every confluence-facing consumer of an accepted* $\mathsf{NewmanLemma}^{\uparrow}$ *carrier first reads the finite handoff*

$$T, L \longmapsto W \longmapsto J \longmapsto B \longmapsto O, H, C, P, M.$$

Thus local confluence is consumed only under the displayed normalization or termination row, and the resulting join budget is handed to the MetaCIC normalization-confluence bridge before any confluence conclusion is visible.

Proof. Start with the rows T and L. The local peak row is readable only for the same source whose terminating measure is displayed by T.

Next consume W and J. The descent ledger orders the finite peak analysis, and the local join budget records the bounded join data produced by that analysis.

The bridge and structural rows then close the handoff route. Row B passes the bounded join budget to the normalization-confluence bridge, while O, H, C, P, M retain the obstruction boundary, replay order, provenance, and local naming. □

Theorem 8455.4 (Newman MetaCIC route)**.** *Any MetaCIC use of* $\mathsf{NewmanLemma}^{\uparrow}$ *factors through the sibling route*

$\mathsf{ChurchRosser}^{\uparrow}$, $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondI}$

The Newman packet may therefore serve as a finite local-confluence and normalization handoff, but it does not discharge subject reduction, residual substitution compatibility, full closed confluence, quotient syntax equality, evaluator equality, or kernel consistency.

Proof. Read the provenance row P of Definition 8455.1. It names the Church–Rosser sibling, the normalization-confluence bridge, the normalization frontier, and the parallel-diamond frontier as the only MetaCIC-facing dependencies.

The analytic rows T, L, W, J, B are then consumed along that provenance route. They can produce a bounded local-confluence handoff under termination, and they can feed the bridge row, but they do not add the missing subject-reduction or residual-substitution premises.

The obstruction and structural rows preserve this boundary. Hence the route remains a finite Newman-style handoff inside the displayed MetaCIC dependency lattice. □

Closure status for $\mathsf{NameCert}_{\mathsf{NewmanLemma}\uparrow}$.

Constructive story: A $\mathsf{NewmanLemma}^{\uparrow}$ packet is generated from termination evidence, local-confluence exposure, well-founded descent, local-join budgeting, MetaCIC bridge routing, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NewmanLemma}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8455.1, Definition 8455.2, Definition 8455.3, and Definition 8455.4 seed the finite Newman handoff from terminating local confluence into the MetaCIC normalization-confluence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NewmanLemma}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, residual substitution compatibility, full closed confluence, quotient syntax equality, evaluator equality, Lean verification, bridge checking, or kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, local-confluence exposure, well-founded descent, local-join budgeting, MetaCIC bridge routing, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.NewmanLemmaUp.

Chapter 8456

A Concrete Naming Certificate for MetaCICResidualNormalFormBudgetRouter$^{\uparrow}$

The MetaCICResidualNormalFormBudgetRouter$^{\uparrow}$ packet realizes a residual normal-form budget route for MetaCIC as finite candidate-mediated NameCert data. It sits beside the typed normal-form scheduler of chapter 6617, the residual diamond budget of chapter 7407, the residual join budget of chapter 7378, the normal-endpoint budget of chapter 4292, and the candidate-normalization confluence handoff of chapter 5547. The router stores the finite rows by which a residual pair, candidate gate, normal-form scheduler budget, and endpoint seal are read before a confluence-facing handoff is public; it is not a global normalization theorem, subject-reduction theorem, unrestricted residual diamond, quotient syntax equality, evaluator equality, or kernel-consistency certificate.

Definition 8456.1 (MetaCIC residual normal-form budget router carrier)**.** *A* MetaCICResidualNormalFormBudgetR *carrier is a finite* BHist *packet*

$$R = (W, D, J, S, E, Q, O, H, C, P, N)$$

with the following displayed rows:

1. *W is the typed residual window row, recording the closed typed source, residual pair, and bounded comparison window.*
2. *D is the residual-diamond budget row, read through* MetaCICResidualDiamondBudget$^{\uparrow}$ *and retaining the finite local-diamond checker surface.*
3. *J is the residual-join budget row, read through* MetaCICResidualJoinBudget$^{\uparrow}$ *and recording the candidate join that the checker may inspect.*
4. *S is the typed normal-form scheduler row, read through* MetaCICTypedNormalFormScheduler$^{\uparrow}$ *and exposing candidate-gate and scheduler-budget data.*
5. *E is the normal-endpoint budget row, read through* MetaCICNormalEndpointBudget$^{\uparrow}$ *and sealing the finished normal endpoint available to bounded comparison.*
6. *Q is the candidate-normalization confluence handoff row, read through* MetaCICCandidateNormalizationConfluenc *only after W, D, J, S, E have been displayed.*
7. *O is the obstruction row, retaining failed candidate gates, exhausted scheduler budgets, blocked residual pairs, missing endpoint seals, and failed bounded local joins.*
8. *H records componentwise* hsame *transport for W, D, J, S, E, Q, O and the structural rows.*

9. *C records displayed* Cont *replay from a residual normal-form request through* $W, D, J, S, E, Q, O, H, P, N$.
10. *P is the* Pkg *provenance over* MetaCICResidualNormalFormBudgetRouter$^{\uparrow}$, MetaCICTypedNormalFormScheduler$^{\uparrow}$, MetaCICResidualDiamondBudget$^{\uparrow}$, MetaCICResidualJoinBudget$^{\uparrow}$, MetaCICNormalEndpointBudget$^{\uparrow}$, MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}}$ *row.*

The classifier compares accepted router packets only through $W, D, J, S, E, Q, O, H, C, P, N$. It has no coordinate for unrestricted residual substitution, global strong normalization, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency.

Theorem 8456.2 (MetaCIC residual normal-form budget router obligations)**.** *For every accepted* MetaCICResidualNormalFormBudgetRouter$^{\uparrow}$ *carrier*

$$R = (W, D, J, S, E, Q, O, H, C, P, N),$$

the local NameCert$_{\text{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}}$ *surface consists exactly of typed residual-window admission, residual-diamond budget exposure, residual-join budget exposure, typed scheduler budget routing, normal-endpoint sealing, candidate-normalization confluence handoff, obstruction retention, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8456.1. The accepted packet displays exactly $W, D, J, S, E, Q, O, H, C, P, N$, so the NameCert surface is exhausted by those rows.

The residual rows W, D, J are read before any normal-form endpoint is public. They bind the typed window, local-diamond budget, and candidate join budget to the same residual pair.

The scheduler and endpoint rows S, E then expose candidate-gate, finite scheduler, and finished endpoint data. The handoff row Q can be consumed only after those rows are visible.

Finally O, H, C, P, N retain the obstruction boundary and preserve the displayed route under transport, replay, provenance, and local naming. Since no further coordinate is present, the packet cannot export unrestricted residual substitution, global normalization, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency. □

Theorem 8456.3 (MetaCIC residual normal-form router handoff)**.** *Every candidate-mediated confluence consumer of an accepted* MetaCICResidualNormalFormBudgetRouter$^{\uparrow}$ *packet first reads the finite route*

$$W \longmapsto D, J \longmapsto S \longmapsto E \longmapsto Q \longmapsto O, H, C, P, N.$$

Thus residual diamond and residual join budgets are consumed before the typed normal-form scheduler and normal-endpoint budget, and the candidate-normalization confluence handoff is visible only after those budget rows have been displayed.

Proof. Start with the typed residual window W. It fixes the closed typed source, residual pair, and bounded comparison window for the whole route.

Next read D and J. These rows expose the residual-diamond and residual-join budgets without turning them into a premise-free confluence theorem.

Then consume S, E, Q. The scheduler row supplies candidate-gate and budget data, the endpoint row seals the finished normal endpoint, and the handoff row exports only the candidate-mediated confluence boundary that can read those budgets.

The obstruction and structural rows keep the same boundary public. They record missing gates, exhausted budgets, blocked residuals, failed endpoint seals, failed joins, transport, replay, provenance, and local naming. □

Falsifiable prediction: Any accepted $\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}$ confluence-facing read that exposes Q without first displaying typed residual window W, residual-diamond budget D, residual-join budget J, typed scheduler budget S, endpoint seal E, obstruction O, transport H, replay C, provenance P, and local naming N is outside the carrier surface.

Independence witness: metacic-typed-normal-form-scheduler, metacic-residual-diamond-budget, metacic-residual-join-budget, metacic-normal-endpoint-budget, metacic-candidate-normalization-confluence-handoff: no listed sibling is bijective with $\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}$ because this packet jointly carries a typed residual window, residual-diamond budget, residual-join budget, scheduler budget, endpoint seal, candidate-normalization handoff, obstruction, transport, replay, provenance, and local NameCert rows.

Lean TasteGate work for $\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}$ must supply Nontrivial and FieldFaithful instances so the residual window, residual-diamond budget, residual-join budget, scheduler budget, endpoint seal, handoff, obstruction, replay, provenance, and local naming rows remain distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}$ packet is generated from a typed residual window, residual-diamond budget, residual-join budget, typed normal-form scheduler budget, normal-endpoint seal, candidate-normalization confluence handoff, obstruction row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8456.1, Definition 8456.2, and Definition 8456.3 seed the finite residual normal-form budget route through typed residual windows, residual-diamond and residual-join budgets, scheduler budgets, endpoint seals, candidate handoff, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalFormBudgetRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted residual substitution, global strong normalization, subject reduction, unrestricted residual diamond, quotient syntax equality, evaluator equality, Lean proof obligations, bridge checking, or kernel consistency.

Upgrade path: Obligation closure requires checked carrier admission, typed residual-window exposure, residual-diamond budget exposure, residual-join budget exposure, scheduler-budget routing, endpoint sealing, handoff exactness, Nontrivial and FieldFaithful instances, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualNormalFormBudgetRouterUp.

Chapter 8457

A Concrete Naming Certificate for $\mathsf{LocatedRealTailMajorant}^{\uparrow}$

Following Bishop and Bridges constructive analysis, a located real is read through explicit finite approximations, displayed error bounds, and a named handoff into the real-number surface rather than through an ambient completed object. The $\mathsf{LocatedRealTailMajorant}^{\uparrow}$ packet records the finite tail-majorant route that a Real-and-Completion consumer uses after the regular-Cauchy tail rows of chapter 5900 and the tail-bound rows of chapter 2071, while still reading finite windows through chapter 126, dyadic tolerances through chapter 291, regular rational names through chapter 301, located upper witnesses through chapter 6098, and the terminal real-name seal through chapter 38. The packet names a displayed majorant threshold for located reads; it does not quotient streams, select limits, invoke countable choice, use trichotomy, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8457.1 (Located real tail-majorant carrier)**.** *A* $\mathsf{LocatedRealTailMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, D, T, B, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected prefixes and tail windows of the located real name.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row listing the dyadic precision requests used by the tail comparison.*
3. *T is the* $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ *threshold row, giving an explicit index after which the inspected regular-Cauchy tail is bounded by the displayed dyadic request.*
4. *B is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *comparison row, recording the finite inequality check that turns the threshold into a bound usable by downstream consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name attached to the same windows and tolerances.*
6. *U is the* $\mathsf{LocatedSequenceSupremum}^{\uparrow}$ *handoff row, exposing the located upper comparison needed when a consumer asks for a majorant of all later inspected values.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, T, B, R, U have been consumed.*
8. *H records componentwise* hsame *transport for the stream, dyadic, threshold, bound, regular-rational, located-supremum, and real-seal rows.*

9. *C records displayed* Cont *replay order for the finite tail-majorant route.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealTailMajorant}^{\uparrow}$, $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{LocatedSequenceSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealTailMajorant}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limit values, host real equality, countable choice, trichotomy, a detached supremum object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8457.2 (Located real tail-majorant NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealTailMajorant}^{\uparrow}$ *carrier*

$$L = (S, D, T, B, R, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealTailMajorant}^{\uparrow}}$ *surface consists exactly of finite StreamName-window exposure through S, dyadic tolerance exposure through D, explicit tail-majorant threshold exposure through T, Cauchy tail-bound comparison through B, regular-rational readback through R, LocatedSequenceSupremum handoff through U, terminal Real seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows $S, D, T, B, R, U, E, H, C, P, N$ are the only public coordinates available to the local certificate, so every obligation must be read from one of these displayed finite rows.

Read the numerical rows first. The finite windows in S say which tail entries are inspected, the dyadic row D fixes the requested tolerances, the threshold row T gives the explicit majorant index, and the comparison row B records the finite inequalities witnessing that the tail after T is bounded at those tolerances.

Read the handoff rows next. The regular-rational row R supplies the RegSeqRat readback attached to the same windows, while U exposes the located-sequence-supremum comparison that lets a downstream located consumer ask for a majorant without selecting a completed supremum.

Finally inspect E, H, C, P, N. The Real seal is reachable only after the stream, dyadic, threshold, bound, regular-rational, and located-supremum rows have been consumed, and the structural rows preserve, replay, package, and name the same finite route. Thus the displayed packet exhausts the local NameCert obligations. □

Theorem 8457.3 (Located real tail-majorant Real handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{LocatedRealTailMajorant}^{\uparrow}$ *packet must reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailMajorant}^{\uparrow} \longmapsto \mathsf{CauchyTailBound}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Loc}$$

Consequently the consumer obtains a finite located tail-majorant threshold and its dyadic comparison rows before it can read the Real seal. No quotient stream, selected limit, detached supremum, host equality, countable-choice schedule, trichotomy, or ambient completeness principle is exported by the handoff.

Proof. Start from the obligation surface of Definition 8457.2. The Real seal row E is not an independent coordinate; it is named after the finite stream windows S, dyadic tolerances D, threshold row T, tail-bound comparison B, regular-rational readback R, and located-supremum handoff U.

The regular-Cauchy tail-majorant and Cauchy tail-bound rows keep the numerical estimate finite. They expose an explicit threshold and the dyadic inequalities checked at the requested tolerances, rather than a completed tail object or a choice-selected limiting value.

The RegSeqRat and LocatedSequenceSupremum rows then provide the Real-facing readback. They let a consumer compare located upper information against the same finite windows and tolerance rows, while keeping the supremum handoff as a displayed certificate row rather than a detached completed object.

Since the carrier of Definition 8457.1 has no coordinate for quotient streams, selected limits, host equality, countable choice, trichotomy, detached suprema, or ambient completeness, none of those objects can be projected from the handoff. Every accepted Real read therefore passes through the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealTailMajorant}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealTailMajorant}^{\uparrow}$ packet is generated from finite StreamName windows, dyadic tolerance rows, an explicit regular-Cauchy tail-majorant threshold, Cauchy tail-bound comparisons, RegSeqRat readback, a LocatedSequenceSupremum handoff, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealTailMajorant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8457.1, Definition 8457.2, and Definition 8457.3 seed the finite located-real tail-majorant route over displayed StreamName windows, dyadic tolerance rows, explicit threshold rows, RegSeqRat readback, LocatedSequenceSupremum handoff, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealTailMajorant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, trichotomy, detached supremum objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, StreamName window readback, dyadic tolerance readback, explicit threshold exactness, tail-bound comparison, RegSeqRat readback, LocatedSequenceSupremum handoff, Real seal nonescape, provenance, and local naming for the displayed finite packet.

Chapter 8458

A Concrete Naming Certificate for $\mathsf{DualSpace}^{\uparrow}$

Following Bishop and Bridges on constructive normed spaces and Riesz's duality tradition, the $\mathsf{DualSpace}^{\uparrow}$ packet records the continuous dual of a displayed normed vector-space carrier as finite NameCert data. It sits over the vector-space carrier of chapter 47, the norm row of chapter 89, the linear-map surface of chapter 48, and the bounded-linear-functional boundary of chapter 5790. The packet names only displayed bounded scalar functional rows and their finite transport data; it does not import Hahn–Banach, choice, quotient duals, or full Banach duality.

Definition 8458.1 (Dual space carrier)**.** *A* $\mathsf{DualSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (V, N, L, B, E, H, C, P, A)$$

with the following visible rows:

1. *V is the source* $\mathsf{VecSpace}^{\uparrow}$ *row, including the vector addition, zero, inverse, scalar-action, and vector-classifier reads available to dual consumers.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row on the same source, exposing the finite norm comparison and norm-transport witnesses used to state boundedness.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row for a scalar-valued linear action on the displayed vector rows.*
4. *B is the* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row, recording the boundedness ledger for L against N with scalar endpoints read through the ambient scalar certificate named by the dependency packet.*
5. *E is the evaluation row, pairing an accepted functional row with an accepted vector row and returning the displayed scalar endpoint.*
6. *H records componentwise* hsame *transport for V, N, L, B, E, C, P, A.*
7. *C records displayed* Cont *replay from vector and norm reads through linearity, boundedness, and evaluation.*
8. *P is the* Pkg *provenance over* $\mathsf{DualSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearFunctional}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *A is the local* $\mathsf{NameCert}_{\mathsf{DualSpace}^{\uparrow}}$ *row.*

The classifier compares accepted dual packets only through these displayed rows. It has no coordinate for all algebraic linear functionals, a quotient by pointwise equality, a selected representative of a functional class, a Hahn–Banach extension principle, or an ambient Banach-dual theorem.

Theorem 8458.2 (Dual space NameCert obligations)**.** *For every accepted* $\mathsf{DualSpace}^{\uparrow}$ *carrier* $D = (V, N, L, B, E, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{DualSpace}^{\uparrow}}$ *surface consists exactly of source vector-space admission, norm compatibility, scalar-valued linear-map admission, bounded-linear-functional ledger exposure, evaluation-row exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer of the packet receives no unbounded functional, no quotient dual carrier, no selected representative, no Hahn–Banach extension, and no full Banach-duality theorem.*

Proof. Project the carrier of Definition 8458.1. The rows $V, N, L, B, E, H, C, P, A$ are the full inventory, so every accepted read begins with the displayed vector-space and norm rows.

Read the functional rows next. The linear-map row L supplies only the scalar-valued linear action visible through $\mathsf{LinMap}^{\uparrow}$, and the boundedness row B records the finite norm ledger supplied by $\mathsf{BoundedLinearFunctional}^{\uparrow}$.

The evaluation row E is then exact because it pairs only an accepted bounded functional row with an accepted vector row from V. Transport and replay through H and C preserve those same coordinates, while P and A only record provenance and local naming.

Thus the NameCert obligations are precisely the displayed finite rows. Since no row stores an unbounded functional, quotient dual, selected representative, Hahn–Banach extension, or full Banach-duality object, none can be exported by the accepted packet. □

Falsifiable prediction: A purported $\mathsf{DualSpace}^{\uparrow}$ consumer that evaluates a functional without first exposing the $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, and $\mathsf{BoundedLinearFunctional}^{\uparrow}$ rows is rejected by the carrier of Definition 8458.1.

Independence witness: vecspace, norm, linearmap, boundedlinearfunctional: $\mathsf{DualSpace}^{\uparrow}$ is not bijective to any listed sibling. $\mathsf{VecSpace}^{\uparrow}$ lacks scalar-valued bounded functional rows; $\mathsf{Norm}^{\uparrow}$ lacks linear action and evaluation rows; $\mathsf{LinMap}^{\uparrow}$ lacks the norm-boundedness ledger; $\mathsf{BoundedLinearFunctional}^{\uparrow}$ supplies the bounded functional boundary but not the downstream dual-space packet provenance and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{DualSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DualSpace}^{\uparrow}$ packet is generated from VecSpaceUp source rows, NormUp norm rows, LinearMapUp scalar-valued action rows, BoundedLinearFunctionalUp boundedness rows, an evaluation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DualSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8458.1, Definition 8458.2, the falsifiable prediction, and the independence witness seed the finite continuous-dual packet over displayed vector-space, norm, linear-map, bounded-linear-functional, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DualSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach, choice, quotient duals, all algebraic linear functionals, full Banach duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-functional ledger exactness, evaluation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DualSpaceUp.

Chapter 8459

A Concrete Naming Certificate for $\mathsf{SouslinOperation}^{\uparrow}$

Following Suslin and Luzin on analytic sets, the $\mathsf{SouslinOperation}^{\uparrow}$ packet records the finite BEDC surface by which a tree-indexed family of Borel rows on Baire space is read as an analytic-set handoff. It sits between the Baire-space carrier of chapter 334, the Borel hierarchy of chapter 4020, the analytic-set consumer of chapter 8008, the topology and set-like rows of chapter 88 and chapter 3668, and the finite observation route of chapter 126. The packet names a displayed operation from Borel branch data to an analytic-set surface; it is not a host projection theorem, a completed choice sequence, a quotient equality of streams, or an ambient descriptive-set-theoretic universe.

Definition 8459.1 (Souslin-operation carrier)**.** *A* $\mathsf{SouslinOperation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, L, A, T, E, W, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, naming the finite branch prefixes and their inspection order.*
2. *L is the* $\mathsf{BorelHierarchy}^{\uparrow}$ *code-family row, assigning Borel-code data to the displayed finite prefix nodes.*
3. *A is the* $\mathsf{AnalyticSet}^{\uparrow}$ *handoff row produced from the branch-indexed Borel family.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood and closure-stability row used by the Borel and analytic reads.*
5. *E is the* $\mathsf{SetLike}^{\uparrow}$ *membership row, recording the displayed set-membership tests before the analytic handoff is accepted.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which branch observations and membership probes are read.*
7. *R is the operation ledger: it records branch-prefix compatibility, Borel-row stability along extensions, and the finite handoff from the displayed family to the analytic-set row.*
8. *H records componentwise* hsame *transport for $B, L, A, T, E, W, R, P, N$.*
9. *C records displayed* Cont *replay from a requested analytic-set read through Baire branches, Borel codes, topology, set-like membership, stream windows, and the operation ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{SouslinOperation}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BorelHierarchy}^{\uparrow}$, $\mathsf{AnalyticSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SouslinOperation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, L, A, T, E, W, R, H, C, P, N$. It has no coordinate for countable choice, arbitrary projection over a completed product, host equality of sets, quotient stream equality, or an unbounded Borel-code oracle.

Theorem 8459.2 (Souslin-operation NameCert obligations)**.** *For the carrier of Definition 8459.1, the local* $\mathsf{NameCert}_{\mathsf{SouslinOperation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (B, L, A, T, E, W, R, H, C, P, N),$$

Baire branch exposure through B, Borel family exposure through L, analytic-set handoff through A, topology and set-like membership exposure through T, E, finite stream observation through W, operation-ledger exactness through R, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier of Definition 8459.1. The visible rows are exactly $B, L, A, T, E, W, R, H, C, P, N$, so every accepted read is either one of those projections or a route replaying them.

The branch and code side is read through B, L, W. The Baire row names the branch-prefix data, the Borel row attaches a code family to those prefixes, and the StreamName row records the finite observations used by the branch read.

The analytic side is read through A, T, E, R. The topology and set-like rows supply the membership and neighbourhood tests already exposed by the Borel family, while the operation ledger records the branch-extension compatibility needed before A can be exported.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row contains countable choice, host set equality, quotient stream equality, or an unbounded Borel-code oracle, the listed rows exhaust the NameCert obligations. □

Theorem 8459.3 (Souslin-operation analytic-set handoff)**.** *Every accepted* $\mathsf{SouslinOperation}^{\uparrow}$ *packet exposes an* $\mathsf{AnalyticSet}^{\uparrow}$ *handoff only after the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BorelHierarchy}^{\uparrow} \longmapsto \mathsf{Topology}^{\uparrow}, \mathsf{SetLike}^{\uparrow} \longmapsto \mathsf{AnalyticSet}^{\uparrow}$$

has been displayed through $B, W, L, T, E, R, H, C, P, N$. A consumer may use the analytic-set row A only as the output of that route, not as an external projection theorem or a selected infinite branch.

Proof. Begin with the branch data. The row B supplies the Baire prefixes and W supplies the finite observations of those prefixes; no completed branch is selected outside the displayed windows.

Next read the Borel family. The row L assigns the Borel-code data to the inspected prefixes, while T and E expose the topology-facing and membership-facing tests used by those codes.

The ledger R then binds extension compatibility and handoff exactness. It is the only row that relates the branch-indexed Borel family to the analytic-set export.

Finally A, H, C, P, N export and preserve the analytic-set handoff. Because the route never introduces countable choice, host projection, quotient stream equality, or ambient set equality, the consumer receives exactly the displayed $\mathsf{AnalyticSet}^{\uparrow}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SouslinOperation}^{\uparrow}}$.

Constructive story: A $\mathsf{SouslinOperation}^{\uparrow}$ packet is generated from BaireSpace branch rows, BorelHierarchy code-family rows, an AnalyticSet handoff, Topology and SetLike membership rows, StreamName finite windows, an operation ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SouslinOperation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8459.1, Definition 8459.2, and Definition 8459.3 seed the finite operation route from Borel branch data on Baire space to an analytic-set handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SouslinOperation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host projection theorems, countable choice, quotient stream equality, or ambient set equality.

Upgrade path: Obligation closure requires checked carrier admission, Baire branch exposure, Borel family stability, topology and set-like membership replay, StreamName window discipline, analytic-set handoff exactness, and non-escape for BEDC.Derived.SouslinOperationUp.

Chapter 8460

A Concrete Naming Certificate for $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$

The $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ packet realizes the hyperbolic observer-expansion route in chapter 9463 as one finite NameCert surface joining boundary shadows, cross-ratio readback, visual metrics, Busemann boundary data, and observer-expansion ledgers. It sits after the boundary-shadow packet of chapter 3813, the cross-ratio packet of chapter 3948, the visual-metric packet of chapter 3866, the Busemann boundary packet of chapter 3732, and the observer-expansion packet of chapter 4167. The packet is a finite boundary-shadow cross-ratio ledger; it is not a host hyperbolic boundary, a selected geodesic endpoint, a global compactification theorem, or ambient real-valued geometry.

Falsifiable prediction: An accepted $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ read is refuted if a consumer exports a cross-ratio comparison without first displaying boundary-shadow rows, visual-metric rows, Busemann boundary rows, observer-expansion ledger rows, transport, replay, provenance, and local naming.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-cross-ratio, hyperbolic-visual-metric, hyperbolic-busemann-boundary, hyperbolic-observer-expansion: $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ is not bijective to any listed sibling because only this packet jointly records boundary-shadow incidence, four-point cross-ratio readback, visual-metric comparison, Busemann boundary calibration, observer-expansion routing, transport, replay, provenance, and local NameCert rows.

Definition 8460.1 (Boundary-shadow cross-ratio carrier)**.** *A* $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, Q, V, B, O, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, naming the boundary shadows and incidence data visible to the packet.*
2. *Q is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the finite four-boundary-point comparison readback.*
3. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row, supplying the visual-metric comparison surface used by the same boundary data.*
4. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, calibrating the shadow and visual-metric reads through Busemann boundary data.*

5. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row, recording the observer-expansion ledger that consumes the boundary-shadow and metric comparison data.*
6. *R is the unification ledger, binding shadow incidence, cross-ratio order, visual-metric scale, Busemann calibration, and observer-expansion replay to the same displayed boundary packet.*
7. *H records componentwise* hsame *transport for S, Q, V, B, O, R, P, N.*
8. *C records displayed* Cont *replay from a boundary-shadow request through cross-ratio readback, visual-metric comparison, Busemann calibration, observer-expansion routing, and the unification ledger.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowCrossRatio}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, V, B, O, R, H, C, P, N$. It has no coordinate for selected boundary endpoints, global hyperbolic compactification, ambient real-valued geometry, quotient boundary equality, or host geodesic equality.

Theorem 8460.2 (Boundary-shadow cross-ratio NameCert obligations)**.** *For the carrier of Definition 8460.1, the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowCrossRatio}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$X = (S, Q, V, B, O, R, H, C, P, N),$$

boundary-shadow exposure through S, cross-ratio readback through Q, visual-metric comparison through V, Busemann boundary calibration through B, observer-expansion routing through O, unification-ledger exactness through R, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier of Definition 8460.1. Its visible rows are exactly $S, Q, V, B, O, R, H, C, P, N$.

The geometric comparison side is read through S, Q, V, B. Boundary shadows provide the incidence surface, cross-ratio readback supplies the four-point comparison, the visual metric supplies scale, and the Busemann boundary row calibrates the same boundary data.

The observer side is read through O, R. The observer-expansion row consumes the boundary-shadow and comparison data, while the unification ledger records that all reads use the same displayed packet.

Finally H, C, P, N preserve, replay, source, and name the route. Since the carrier has no coordinate for selected endpoints, global compactification, ambient real-valued geometry, quotient boundary equality, or host geodesic equality, the listed rows exhaust the obligations. □

Theorem 8460.3 (Boundary-shadow cross-ratio observer handoff)**.** *Every accepted* $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ *packet hands an observer-expansion consumer the finite route*

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longmapsto \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{HyperbolicVisualMetric}^{\uparrow}, \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto$$

The handoff must pass through $S, Q, V, B, O, R, H, C, P, N$ before any boundary-shadow cross-ratio comparison is consumed.

Proof. Start with the boundary-shadow row S. It fixes the displayed boundary incidence data before any comparison row is read.

Next read the comparison rows Q, V, B. The cross-ratio, visual-metric, and Busemann boundary rows provide compatible comparison surfaces for the same displayed shadows.

The ledger R then binds those surfaces to the observer-expansion row O. This is the step that prevents a cross-ratio read from floating free of the boundary-shadow and observer-expansion data.

Transport, replay, provenance, and local naming are carried by H, C, P, N. Therefore a consumer cannot replace the route by selected boundary endpoints, quotient boundary equality, host geodesic equality, global compactification, or ambient real-valued geometry. □

The Lean `TasteGate.lean` development for $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ must include explicit $\mathsf{Nontrivial\ BoundaryShadowCrossRatio}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundaryShadowCrossRatio}^{\uparrow}$ instances. These instances must preserve the boundary-shadow, cross-ratio, visual-metric, Busemann boundary, observer-expansion, unification-ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowCrossRatio}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$ packet is generated from boundary-shadow rows, cross-ratio readback, visual-metric comparison, Busemann boundary calibration, observer-expansion routing, a unification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowCrossRatio}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8460.1, Definition 8460.2, and Definition 8460.3 seed the finite boundary-shadow cross-ratio route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowCrossRatio}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected boundary endpoints, global compactification, quotient boundary equality, host geodesic equality, or ambient real-valued geometry.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exposure, cross-ratio readback, visual-metric comparison, Busemann boundary calibration, observer-expansion routing, unification-ledger exactness, nontriviality, and field faithfulness for BEDC.Derived.BoundaryShadowCrossRatioUp.

Chapter 8461

Bounded Variation Cauchy Completion NameCert Construction

Following Bishop and Bridges, the $\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}$ packet records the constructive-analysis route by which a regular rational Cauchy name is completed while a finite variation-budget ledger remains visible to BEDC consumers. It sits over the regular Cauchy readback of chapter 301, the terminal real-name seal of chapter 38, and the tail-threshold normalization surface of chapter 1299. The packet is a finite NameCert carrier for bounded-variation completion data; it is not a quotient of Cauchy names, not a selected limit operator, not a host equality principle for completed reals, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 8461.1 (Bounded-variation Cauchy completion carrier)**.** *A* $\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (Q, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying the regular rational Cauchy name whose tail will be read by the completion consumer.*
2. *T is the* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *row selecting a finite dyadic tail window for each displayed unary tolerance request.*
3. *L is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *variation-budget ledger, recording dyadic tail differences, partial variation sums, and the displayed bound attached to the normalized tail windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, T, L have been consumed.*
5. *H records componentwise* hsame *transport for the source regular name, the normalized tail threshold, the variation-budget ledger, the terminal real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from every finite completion request through*

$$Q, T, L \longmapsto E, H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailThresholdNorm}$ $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}}$ *row.*

The classifier accepts exactly componentwise hsame*-stable packets of this form. In particular, replacing the regular name, changing the normalized tail threshold, or weakening the dyadic variation ledger changes the packet unless the corresponding row is transported by the displayed H row and replayed by C.*

Theorem 8461.2 (Bounded-variation Cauchy completion obligations)**.** *For every accepted* BoundedVariationCauch *carrier*

$$V = (Q, T, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* BHist *packet;*
2. *classifier exposure for componentwise* hsame *transport of Q, T, L, E, H, C, P, N;*
3. *stability of the normalized dyadic tail rows under* hsame *and* Cont *replay;*
4. *exactness of the regular Cauchy readback through Q and of the terminal* $\mathsf{Real}^{\uparrow}$ *seal through E;*
5. *ledger descent for the finite variation budget, requiring every later tail window to read a suffix of the same dyadic ledger L rather than a detached variation bound;*
6. *package provenance and local naming through P, N.*

No obligation row exports quotient Cauchy equality, a selected representative, a selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier tuple in Definition 8461.1. The only analytic rows available before the terminal seal are the regular rational source Q, the tail-threshold normalizer T, and the dyadic variation-budget ledger L.

The classifier row is componentwise. Thus an accepted transport must preserve the same source name, normalized tail window, ledger, seal, replay, provenance, and local naming coordinates through hsame; the displayed continuation row C replays the same route for each finite completion request.

It remains to justify ledger descent. Use induction on the finite tail-window spine supplied by T. The base window reads the first displayed dyadic suffix of Q and the initial partial variation budget in L.

For the induction step, the next normalized tail window is a suffix of the previous displayed window. The ledger row L records the corresponding dyadic difference budget and partial variation sum, so the successor read can only decrease the remaining displayed budget or keep it equal under hsame transport. The continuation row C then hands the same suffix route to the terminal seal E.

Since each step uses only finite rows of the displayed packet, the argument invokes neither quotienting of Cauchy names nor a choice of representatives. The absent quotient, selected-limit, host-equality, countable-choice, and ambient-completeness data therefore cannot become NameCert obligations. □

Theorem 8461.3 (Bounded-variation Cauchy completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}$ *carrier must read the finite route*

$$Q, T, L \longmapsto E, H, P, N$$

before the completed real seal is public. The handoff therefore preserves the regular Cauchy source, normalized tail threshold, and finite variation-budget ledger as visible BEDC rows.

Proof. Start with the obligation theorem Definition 8461.2. It exposes the terminal seal E only after carrier admission, classifier transport, tail stability, readback exactness, and ledger descent have been recorded for the same tuple.

The source row Q supplies the regular rational Cauchy name, while T supplies the finite tail threshold used by a completion request. The ledger L is read before E, so a consumer cannot replace the displayed variation budget by an external supremum or a host-level completed-real statement.

Transport and replay preserve the handoff. The rows H, C, P, N name only the displayed route, and the carrier contains no coordinate for quotient Cauchy equality, selected representatives, selected limits, host real equality, countable choice, or ambient completeness. Hence every accepted Real-facing read remains attached to Q, T, L. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationCauchyCompletion}\uparrow}$.

Constructive story: A $\mathsf{BoundedVariationCauchyCompletion}^{\uparrow}$ packet is generated from a RegSeqRat regular Cauchy source row, a CauchyTailThresholdNormalizer tail-window row, a DyadicRatCore finite variation-budget ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationCauchyCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8461.1, Definition 8461.2, and Definition 8461.3 seed the finite bounded-variation completion route from RegSeqRat through tail normalization and dyadic ledger descent to RealUp readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationCauchyCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy equality, selected representatives, selected limits, host real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, tail-window stability, readback exactness, ledger descent, transport, replay, provenance, and local naming for BEDC.Derived.BoundedVariationCauchyCompletionUp.

Chapter 8462

A Concrete Naming Certificate for $\mathsf{NashEmbedding}^{\uparrow}$

Following Nash on isometric embeddings of Riemannian manifolds, the $\mathsf{NashEmbedding}^{\uparrow}$ packet records only the finite chartwise certificate surface needed to name an embedding candidate inside BEDC. It sits between the manifold chart carrier of chapter 96, the metric rows of chapter 133, the vector target surface of $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{EuclideanSpace}^{\uparrow}$, and the terminal scalar seals of chapter 38. The packet is not a host Nash theorem, not a selected global chart, not a quotient manifold equality, and not an ambient completeness principle.

Definition 8462.1 (Nash embedding carrier)**.** *A* $\mathsf{NashEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (M, G, A, E, D, R, H, C, P, L)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *finite chart row, listing the inspected chart names, chart domains, and overlap rows used by the embedding read.*
2. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row, exposing only the finite metric-component ledger over the displayed chart windows.*
3. *A is the chartwise atlas-to-target ledger, recording which chart rows are sent into the finite* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{EuclideanSpace}^{\uparrow}$ *target rows.*
4. *E is the embedding-candidate row, naming the finite coordinate functions and point-separation obligations inspected in the displayed chart windows.*
5. *D is the* $\mathsf{Derivative}^{\uparrow}$ *comparison row, storing the finite derivative or metric-pullback comparisons used to test that E preserves the displayed metric rows of G.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row for every metric coefficient, coordinate coefficient, derivative comparison, and pullback comparison consumed by D.*
7. *H records componentwise* hsame *transport for chart rows, metric rows, target rows, embedding-candidate rows, derivative comparison rows, scalar seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a chartwise embedding request through M, G, A, E, D, R, H.*
9. *P is the* Pkg *provenance over* $\mathsf{NashEmbedding}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{EuclideanSpace}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. L is the local $\mathsf{NameCert}_{\mathsf{NashEmbedding}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host embedding operator, a global Nash existence theorem, quotient equality of manifolds, selected global charts, an unbounded coordinate family, or ambient Euclidean completeness.

Theorem 8462.2 (Nash embedding NameCert obligations)**.** *For every accepted* $\mathsf{NashEmbedding}^{\uparrow}$ *carrier* $N = (M, G, A, E, D, R, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{NashEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite chart row, metric-component admission, atlas-to-target admission, embedding-candidate admission, derivative or metric-pullback comparison admission, Real scalar sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No obligation can be discharged by reading a host Nash theorem, a global embedding operator, quotient manifold equality, selected global charts, or ambient Euclidean completeness outside those rows.

Proof. Project the carrier of Definition 8462.1. The available rows are exactly $M, G, A, E, D, R, H, C, P, L$, so every naming obligation must be assigned to one of those coordinates.

The geometric obligations are chartwise. The row M supplies the inspected chart windows, G supplies the finite metric ledger over those windows, A routes the same chart windows into the target rows, and E records the finite embedding-candidate data that downstream consumers may inspect.

The comparison obligations are scalar-sealed. The row D stores only derivative or metric-pullback comparisons over the displayed chart windows, and R seals the scalar coefficients as $\mathsf{Real}^{\uparrow}$ rows before the structural rows H, C, P, L transport, replay, source, and name the route. Since no other coordinate appears, the excluded host and quotient objects are not part of the local NameCert surface. □

Theorem 8462.3 (Nash embedding finite chart metric route)**.** *Every metric-facing consumer of an accepted* $\mathsf{NashEmbedding}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Manifold}^{\uparrow},\ \mathsf{RiemannianMetric}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{EuclideanSpace}^{\uparrow},\ \mathsf{Derivative}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $M, G, A, E, D, R, H, C, P, L$ before treating the packet as an isometric-embedding witness. The route exports finite chartwise metric comparison data, not an ambient existence theorem or a globally selected coordinate system.

Proof. Start with the chart and metric rows. The rows M and G determine the only chart windows and metric coefficients that can be inspected by a consumer.

Next read the target and comparison rows. The row A routes the displayed chart windows into the finite vector and Euclidean target rows, E supplies the embedding-candidate ledger, and D records the derivative or metric-pullback comparisons against the metric row G.

Finally apply the scalar and structural rows. The row R seals every coefficient and comparison through $\mathsf{Real}^{\uparrow}$, while H, C, P, L keep transport, replay, provenance, and local naming attached to the same finite route. Thus a metric-facing consumer receives only chartwise comparison data and cannot replace it by a host Nash existence theorem, selected global chart, quotient manifold equality, or ambient completeness fact. □

Theorem 8462.4 (Nash embedding Euclidean seal nonescape)**.** *Every* $\mathsf{EuclideanSpace}^{\uparrow}$ *or* $\mathsf{VecSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{NashEmbedding}^{\uparrow}$ *packet receives the target rows only after the chart ledger, Riemannian metric ledger, embedding-candidate ledger, derivative or pullback comparison ledger,*

and Real scalar seal have all been exposed. The Euclidean target seal therefore cannot escape into a host embedding operator, quotient manifold equality, selected global charts, or ambient completeness.

Proof. Inspect Definition 8462.1. The target rows appear only through A and E, and both rows are downstream of the chart row M and the metric row G.

Use Definition 8462.3. It forces every target-facing read through D and R, so the Euclidean coordinate data are consumed only after derivative or pullback comparisons and Real scalar seals are displayed.

Then apply Definition 8462.2. The structural rows H, C, P, L preserve the same finite route and add no host operator, quotient equality, selected global chart, or completeness row. Hence the Euclidean target seal stays inside the local $\mathsf{NashEmbedding}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{NashEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{NashEmbedding}^{\uparrow}$ packet is generated from finite ManifoldUp chart rows, RiemannianMetricUp metric-component rows, atlas-to-target rows, Euclidean and vector target rows, embedding-candidate rows, derivative or metric-pullback comparison rows, RealUp scalar seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NashEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8462.1, Definition 8462.2, Definition 8462.3, and Definition 8462.4 seed the finite chartwise isometric-embedding certificate surface for NashEmbeddingUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NashEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Nash's full embedding theorem, global smooth embedding existence, selected global charts, quotient manifold equality, ambient Euclidean completeness, Lean theorem checking, axiom purity, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite chart accountability, metric-component comparison, derivative or pullback exactness, Real scalar sealing, target nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NashEmbeddingUp.

Chapter 8463

A Concrete Naming Certificate for $\mathsf{RationalIntervalSequence}^{\uparrow}$

Following Bishop and Bridges-style nested interval and Cauchy constructions, the $\mathsf{RationalIntervalSequence}^{\uparrow}$ packet records a finite inspected schedule of rational intervals before any Real or Completion consumer reads a limiting name. It sits over the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet is a schedule object: it exposes endpoints, nesting, widths, finite windows, readback, and seal handoff as displayed NameCert data, not as a selected completed limit.

Definition 8463.1 (Rational interval sequence carrier)**.** *A* $\mathsf{RationalIntervalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, U, N, W, S, R, Z, X, C, P, K)$$

with the following displayed rows:

1. *E is the finite endpoint table of rational lower and upper rows, each row read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before comparison;*
2. *L and U are the displayed lower-endpoint and upper-endpoint schedules, with $L_i \leq U_i$ recorded at every inspected window;*
3. *N is the nesting evidence row, recording $L_i \leq L_j \leq U_j \leq U_i$ whenever the displayed index j refines the displayed index i;*
4. *W is the dyadic width ledger, assigning each displayed interval a dyadic budget and recording the inspected shrinkage evidence used by later regular-readback rows;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected indices, endpoint observations, and replay order;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed nested endpoints and width budgets;*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, available only after the endpoint table, nesting evidence, width ledger, finite windows, and regular rational readback have been displayed;*
8. *X records componentwise* hsame *transport for endpoints, nesting, widths, finite windows, readback, seal handoff, replay, provenance, and local naming rows;*

9. *C records displayed* Cont *replay from a tolerance request through W, then through S, N, E, L, U, then through R, and finally into Z;*
10. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *K is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, U, N, W, S, R, Z, X, C, P, K$. It has no coordinate for a selected limit, quotient real equality, countable choice, trichotomy of real order, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RationalIntervalSequenceUp.TasteGate.RationalIntervalSequenceTasteGate`
`single carrier alignment`

Theorem 8463.2 (Rational interval sequence nesting obligations)**.** *For every accepted* $\mathsf{RationalIntervalSequence}^{\uparrow}$ *carrier*

$$I = (E, L, U, N, W, S, R, Z, X, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalSequence}^{\uparrow}}$ *surface includes exactly the finite nesting obligations displayed by N and W: each inspected interval has rational endpoints in E, L, U, each refined window recorded by S is nested inside the earlier displayed window, and each width comparison is read through the dyadic budget row W before the readback row R or Real seal row Z is visible.*

Proof. Project the carrier. The accepted packet displays endpoint rows E, L, U, nesting evidence N, width budgets W, finite windows S, regular readback R, Real seal handoff Z, and the structural rows X, C, P, K.

Read the nesting row before any readback. For each inspected refinement, N records the lower endpoint moving upward, the upper endpoint moving downward, and the refined interval remaining inside the earlier interval.

Use the width ledger next. The dyadic budget row W is the only route by which a tolerance request reaches the nested endpoints, so shrinkage evidence is finite and displayed before R or Z can be consumed.

The transport, replay, provenance, and local name rows then preserve the same finite route. Since no selected limit or quotient coordinate occurs in the carrier, the nesting obligations are exhausted by the displayed packet. □

Theorem 8463.3 (Rational interval sequence regular readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exported by an accepted* $\mathsf{RationalIntervalSequence}^{\uparrow}$ *carrier is obtained from the displayed nested interval route*

$$E, L, U, N, W, S, R, Z, X, C, P, K.$$

The readback may use finite StreamName windows, dyadic endpoint comparisons, nesting evidence, and width budgets, but it exports no selected real limit, no quotient equality of real names, no countable-choice schedule, no trichotomy principle, and no ambient Real completeness theorem.

Proof. Begin with the finite observation row S. It names the inspected indices and endpoint observations, so a consumer never reads an unbounded interval family directly.

The endpoint and nesting rows E, L, U, N then supply the interval data inside those windows. The dyadic ledger W supplies the tolerance budget needed to treat the displayed endpoints as a regular rational approximation.

The readback row R is downstream of that route. It packages the regular rational data produced by the finite windows and dyadic widths, and the Real seal handoff Z remains terminal rather than a separate selected limit.

Finally replay through C and transport through X. Provenance P and the local NameCert row K keep the packet inside the displayed BEDC surface, so excluded quotient, choice, trichotomy, and completeness claims stay outside the public readback. □

Falsifiable prediction: Deleting the nesting row permits finite interval windows whose endpoints can be read by RegSeqRat without evidence that later windows refine earlier ones.

Independence witness: Nearest siblings: dyadicratcore, streamname, regseqrat, real, nestedinterval, and dyadic-nested-interval-selector; no bijection exists because this packet records a rational interval schedule with endpoint, nesting, width, readback, and Real-seal rows in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalSequence}^{\uparrow}$ packet is generated from rational endpoint rows, lower and upper endpoint schedules, nesting evidence, a dyadic width ledger, finite StreamName windows, RegSeqRat readback, Real seal handoff, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8463.1, Definition 8463.2, and Definition 8463.3 specify the finite nested rational interval schedule and its regular rational readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient real equality, countable choice, trichotomy of real order, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{RationalIntervalSequence}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, nesting exactness, dyadic width monotonicity, finite-window replay, regular-rational readback, Real-seal handoff, transport, provenance, and local NameCert obligations.

Chapter 8464

Nikodym Boundedness NameCert Construction

Following Nikodym and Dunford–Schwartz, the $\mathsf{NikodymBoundedness}^{\uparrow}$ packet records the finite NameCert surface by which a pointwise bounded family of countably additive measures exposes one displayed bound on the inspected variation rows. It sits over the measure carrier of chapter 92, the local regular rows of chapter 2263, the density handoff of chapter 3448, the bounded-family discipline of chapter 1835, and the terminal real-name seal of chapter 38. The packet is a finite reconstruction of the traditional boundedness route; it is not a host measure-completion theorem, a selector for subfamilies, or an ambient Banach-dual principle.

Definition 8464.1 (Nikodym boundedness carrier)**.** *A* $\mathsf{NikodymBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, F, W, T, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, carrying the measurable-event surface and the displayed countable-additivity rows that the family is allowed to inspect.*
2. *A is the finite event-algebra window. It lists the measurable rows $E_1, \ldots, E_n$ on which the family is tested and records their transport under the* $\mathsf{Measure}^{\uparrow}$ *classifier.*
3. *F is the measure-family ledger, a finite schedule of signed or finite-variation* $\mathsf{Measure}^{\uparrow}$ *rows ν_i over the same displayed measurable surface.*
4. *W is the pointwise boundedness window. For every event row in A, it records a* $\mathsf{Real}^{\uparrow}$ *bound for the displayed family values $\nu_i(E_j)$.*
5. *T is the variation-window row, recording the finite disjoint decompositions of the displayed events and the variation sums that may be read from F.*
6. *B is the candidate uniform* $\mathsf{Real}^{\uparrow}$ *bound for the displayed variation rows.*
7. *D is the* $\mathsf{RadonNikodym}^{\uparrow}$ *density handoff row used only when a displayed member of F is read through a dominated local density.*
8. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *regularity row for the locally finite event windows used by A, T, D.*
9. *H records componentwise* hsame *transport for $M, A, F, W, T, B, D, R, C, P, N$.*

10. *C records displayed* Cont *replay from a requested family-bound read through*

$$M, A, F, W, T, D, R \longmapsto B, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{NikodymBoundedness}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{RadonNikodym}^{\uparrow}$, $\mathsf{UniformBoundedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{NikodymBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a completed measure algebra, a choice-selected subfamily, a quotient of measures, a hidden dual Banach space, or a global countable-additivity principle beyond the displayed rows of M.

Lean4 ✓ `BEDC.Derived.NikodymBoundednessUp.NikodymBoundednessTasteGate single carrier alignment`

Theorem 8464.2 (Nikodym boundedness NameCert obligations)**.** *For every accepted* $\mathsf{NikodymBoundedness}^{\uparrow}$ *carrier*

$$K = (M, A, F, W, T, B, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NikodymBoundedness}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed measure source, event-algebra window, measure-family ledger, pointwise boundedness window, variation window, candidate bound, Radon–Nikodym density handoff, Radon regularity row, transport, replay, provenance, and local naming;*
2. *family coherence, requiring every member of F to read the same* $\mathsf{Measure}^{\uparrow}$ *event surface M and the same event-window row A;*
3. *pointwise boundedness admission, requiring W to supply only scheduled event reads and their* $\mathsf{Real}^{\uparrow}$ *bounds;*
4. *variation-window exactness, requiring T to build every exposed variation sum from the displayed finite decompositions and values in F;*
5. *Radon and Radon–Nikodym compatibility, requiring R and D to be local handoff rows rather than ambient completion or domination principles;*
6. *uniform-bound exposure through the single* $\mathsf{Real}^{\uparrow}$ *row B and the replay row C;*
7. *package provenance and local naming through P, N.*

No local obligation exports measure completion, countable additivity beyond the displayed rows of M, choice-selected subfamilies, quotient equality of measures, or ambient Banach-space duality.

Proof. Project the carrier of Definition 8464.1. The projection exposes exactly the measure source, event window, family ledger, pointwise boundedness window, variation window, bound row, Radon–Nikodym density handoff, Radon regularity row, transport, replay, provenance, and local naming.

Family coherence is read from F together with M and A. Every family member is evaluated only on events admitted by the same displayed $\mathsf{Measure}^{\uparrow}$ surface, and each event read is transported by the same classifier row. Thus no detached measurable universe or quotient of event algebras is available.

The boundedness data is local. The row W records pointwise $\mathsf{Real}^{\uparrow}$ bounds on the scheduled event reads, while T records the finite disjoint decompositions used to compute the displayed variation sums. The candidate bound B is downstream from those two rows and cannot be read before them.

The Radon rows are also local. The row R supplies regularity only for the displayed windows, and D supplies a Radon–Nikodym density handoff only for a displayed dominated member of the family. Neither row contains measure completion, a selected subfamily, or an ambient dual-space identification.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the classifier sees only the listed coordinates, the excluded host principles cannot become NameCert obligations. □

Theorem 8464.3 (Nikodym boundedness measure-family route)**.** *Every accepted* $\mathsf{NikodymBoundedness}^{\uparrow}$ *family-bound read factors through the displayed route*

$$\mathsf{Measure}^{\uparrow}, \mathsf{RadonMeasure}^{\uparrow}, \mathsf{RadonNikodym}^{\uparrow}, \mathsf{UniformBoundedness}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Concretely, the read first opens the common measure source M and event window A, then inspects the family ledger F and pointwise bound window W, then computes only the variation rows listed in T, then uses the local Radon and Radon–Nikodym rows R, D when those rows are present, and only then exposes the uniform $\mathsf{Real}^{\uparrow}$ *bound B through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.NikodymBoundednessUp.NikodymBoundednessMeasureFamilyRoute`

Proof. Start with the obligation theorem Definition 8464.2. It says that an accepted read may inspect only the displayed rows of the carrier.

The route begins at M, A, F, W. These rows name the measurable surface, the finite event window, the family members, and the pointwise $\mathsf{Real}^{\uparrow}$ bounds. This is the measure-family analogue of the finite bounded-family discipline carried by $\mathsf{UniformBoundedness}^{\uparrow}$.

The variation and regularity rows are downstream. The row T constructs variation reads from finite decompositions of the events in A, while R and D may be consulted only as local $\mathsf{RadonMeasure}^{\uparrow}$ and $\mathsf{RadonNikodym}^{\uparrow}$ handoffs attached to those displayed events and measures.

The endpoint is B. Because B is replayed through C after M, A, F, W, T, R, D, a consumer cannot replace the finite route by measure completion, a countable-choice selection of subfamilies, unrestricted countable additivity, or an ambient Banach-dual theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NikodymBoundedness}^{\uparrow}}$.

Constructive story: A $\mathsf{NikodymBoundedness}^{\uparrow}$ packet is generated from a MeasureUp source, finite event-window rows, a measure-family ledger, pointwise RealUp bound windows, finite variation rows, local RadonMeasure and Radon-Nikodym handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NikodymBoundedness}^{\uparrow}}$, seed)

Scope closed: Definition 8464.1, Definition 8464.2, and Definition 8464.3 seed the finite measure-family boundedness route through MeasureUp, RadonMeasureUp, RadonNikodymUp, UniformBoundednessUp, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NikodymBoundedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, countable additivity beyond the displayed rows, choice-selected subfamilies, ambient Banach-space duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family coherence, pointwise boundedness admission, variation-window exactness, local Radon and Radon–Nikodym compatibility, uniform RealUp bound exposure, transport, replay, provenance, and local naming for BEDC.Derived.NikodymBoundednessUp.

Chapter 8465

A Concrete Naming Certificate for $\mathsf{KraftPrefixCode}^{\uparrow}$

Following Kraft and McMillan, the $\mathsf{KraftPrefixCode}^{\uparrow}$ packet records a finite prefix-free binary code table together with the Kraft budget that makes instantaneous decoding visible as BEDC NameCert data. It sits beside the minimum-distance coding surface of chapter 241, the finite branch discipline of chapter 5910, the source-measure ledger of chapter 192, and the bounded program-code surface of chapter 5989. The packet names only displayed codewords, tree-frontier rows, length rows, Kraft-budget rows, refusal rows, transport, replay, provenance, and local naming; it is not a Shannon optimality theorem, a Huffman construction, an oracle for shortest descriptions, or a host probability-space completion.

Definition 8465.1 (KraftPrefixCode carrier)**.** *A* $\mathsf{KraftPrefixCode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, F, L, B, D, H, T, P, N)$$

with the following displayed rows:

1. *C is the code-table row, listing finitely many source symbols together with their binary codeword histories.*
2. *F is the binary-tree prefix-frontier row, read through* $\mathsf{BinaryTree}^{\uparrow}$*, whose leaves are exactly the displayed codewords admitted by C.*
3. *L is the length-ledger row, recording each displayed codeword length as a finite natural-history budget.*
4. *B is the Kraft-budget row, recording the finite inequality that the sum of the displayed dyadic weights 2^{-L_i} is bounded by the unit budget.*
5. *D is the decoding-refusal row: if a requested bit history is not on the displayed frontier, or extends through two codewords, the packet records refusal rather than a decoded symbol.*
6. *H records componentwise* hsame *transport for code tables, prefix frontiers, length ledgers, Kraft budgets, refusal rows, replay rows, provenance rows, and local naming rows.*
7. *T records displayed* Cont *replay from a finite decoding request through the same C, F, L, B, D, H, P, N rows.*
8. *P is the* Pkg *provenance over* $\mathsf{KraftPrefixCode}^{\uparrow}$*,* $\mathsf{ErrorCode}^{\uparrow}$*,* $\mathsf{BinaryTree}^{\uparrow}$*,* $\mathsf{Entropy}^{\uparrow}$*,* $\mathsf{KolmogorovComplexity}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{KraftPrefixCode}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $C, F, L, B, D, H, T, P, N$. It has no coordinate for a selected optimal code, an infinite alphabet, a completed probability model, quotient code equality, or an unbounded decoder.

Theorem 8465.2 (KraftPrefixCode NameCert obligations)**.** *For every accepted* $\mathsf{KraftPrefixCode}^{\uparrow}$ *carrier*

$$K = (C, F, L, B, D, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KraftPrefixCode}^{\uparrow}}$ *surface consists exactly of code-table exposure, binary-tree prefix-frontier exposure, length-ledger exposure, Kraft-budget exposure, decoding-refusal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every entropy, error-code, or bounded-description consumer must expose $C, F, L, B, D, H, T, P, N$ before reading a finite prefix-code row.*

Lean4 ✓ `BEDC.Derived.KraftPrefixCodeUp.KraftPrefixCodeTasteGate single carrier alignment`

Proof. Project the carrier $K = (C, F, L, B, D, H, T, P, N)$. The analytic rows are the finite code table C, the prefix frontier F, the length ledger L, the Kraft budget B, and the decoding-refusal row D.

The binary-tree frontier makes prefix admission explicit. The length ledger and Kraft budget then read only the displayed leaves, so the packet supplies a finite budget row rather than an optimal coding theorem or a hidden probability-space argument.

The structural rows H, T, P, N transport, replay, source, and name the same finite route. Since the carrier has no row for selected optimality, infinite alphabets, quotient equality, or an unbounded decoder, the listed rows exhaust the NameCert surface. □

Theorem 8465.3 (KraftPrefixCode prefix disjointness)**.** *Every accepted* $\mathsf{KraftPrefixCode}^{\uparrow}$ *carrier*

$$K = (C, F, L, B, D, H, T, P, N)$$

forces the displayed codeword frontier F to be prefix-disjoint: two decoded source rows can share a bit history only when the same displayed frontier leaf is read. Consequently a finite decoding request cannot pass through two distinct codewords, and any request that would do so is routed to the refusal row D rather than to a second decoded symbol.

Proof. Read the code table C through the binary-tree frontier F. The accepted frontier records codewords as leaves, so a decoded request must terminate at one displayed leaf before any longer extension can be considered.

Compare two candidate decoded rows. If they read the same frontier leaf, componentwise hsame transport identifies the same displayed codeword row inside C. If one candidate attempts to extend through another leaf, the prefix-frontier row blocks that branch as a leaf boundary.

The refusal row D records precisely the blocked case. Transport H, replay T, provenance P, and local naming N preserve the same finite frontier order, so the packet never exports ambiguous instantaneous decoding. □

Falsifiable prediction: Every accepted $\mathsf{KraftPrefixCode}^{\uparrow}$ consumer that reads a finite prefix-code row without first displaying the code table, binary-tree frontier, length ledger, Kraft budget, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: errorcode, binarytree, entropy, kolmogorov-complexity: $\mathsf{KraftPrefixCode}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries code-table rows, prefix-frontier rows, length ledgers, Kraft budgets, decoding-refusal rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KraftPrefixCode}\uparrow}$.

Constructive story: A $\mathsf{KraftPrefixCode}^{\uparrow}$ packet is generated from finite code-table rows, binary-tree prefix-frontier rows, finite length ledgers, Kraft-budget rows, decoding-refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KraftPrefixCode}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8465.1, Definition 8465.2, and Definition 8465.3 seed the finite prefix-free coding interface over code tables, binary-tree frontiers, length ledgers, Kraft budgets, decoding refusal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KraftPrefixCode}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Shannon optimality, Huffman construction, selected shortest programs, infinite alphabets, quotient code equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-frontier exactness, length-ledger exposure, Kraft-budget exposure, decoding-refusal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KraftPrefixCodeUp.

Chapter 8466

A Concrete Naming Certificate for $\mathsf{FiniteRealObserver}^{\uparrow}$

Following Bishop and Bridges on finite rational observations of real numbers, the $\mathsf{FiniteRealObserver}^{\uparrow}$ packet records the finite observation window through which a Real-facing reader inspects a regular Cauchy name. It sits beside the finite schedule of chapter 5916, the StreamName window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names a finite observation route; it is not a host real, not a quotient equality of names, not a selected limit, and not a completeness theorem.

Definition 8466.1 (Finite real observer carrier)**.** *A* $\mathsf{FiniteRealObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (Q, L, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row listing the real-name observations being asked for;*
2. *L is the list schedule row, read through a* $\mathsf{FiniteStream}^{\uparrow}$ *prefix surface for the requested observations;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by Q and L;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected window;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the readback;*
6. *H records componentwise* hsame *transport for the request, list schedule, window, readback, tolerance, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from the request through the finite window and readback rows;*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteRealObserver}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealObserver}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite observation trace, a host real, a quotient representative, a selected limit, or a Real equality principle outside the displayed finite route.

Theorem 8466.2 (Finite real observer window soundness)**.** *Every accepted* $\mathsf{FiniteRealObserver}^{\uparrow}$ *packet exposes its Real-facing read through the finite route*

$$Q, L, W, R, D, H, C, P, N.$$

Thus a $\mathsf{Real}^{\uparrow}$ *consumer may inspect the requested window only after the finite request row, list prefix row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows have all been displayed.*

Proof. Project the carrier of Definition 8466.1. The request row Q and list row L determine the only finite observation schedule available to a consumer.

Next read W, R, D. The window is a $\mathsf{StreamName}^{\uparrow}$ row selected by the finite schedule, the readback is a $\mathsf{RegSeqRat}^{\uparrow}$ row attached to that window, and the tolerance ledger is a $\mathsf{DyadicRatCore}^{\uparrow}$ row consumed by the same readback.

Finally use H, C, P, N. Transport is componentwise, continuation replay follows the displayed finite route, provenance names the sibling surfaces, and the local NameCert row seals the packet. Since no other observation coordinate is present, a Real-facing read cannot bypass the displayed window route. □

Theorem 8466.3 (Finite real observer NameCert obligations)**.** *For the carrier of Definition 8466.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteRealObserver}^{\uparrow}}$ *surface consists of carrier admission for* $O = (Q, L, W, R, D, H, C, P, N)$*, finite request admission, list prefix admission, StreamName window admission, RegSeqRat readback admission, DyadicRatCore tolerance admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No obligation can be discharged by reading an ambient infinite stream, a host real, a quotient representative, a selected limit, or a completeness principle outside those rows.

Proof. Read the rows of the carrier. The finite request and list rows Q, L are the only schedule data, so every observation obligation begins there.

The analytic rows are then forced in order. The StreamName row W supplies the finite window, the RegSeqRat row R ties that window to a regular rational readback, and the DyadicRatCore row D supplies the displayed tolerance data.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. The refused host objects are not coordinates of O, so they cannot satisfy a local naming obligation for $\mathsf{FiniteRealObserver}^{\uparrow}$. □

Theorem 8466.4 (Finite real observer Real readback route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *readback route that consumes an accepted* $\mathsf{FiniteRealObserver}^{\uparrow}$ *packet factors through chapter 5916, chapter 126, chapter 301, chapter 291, and chapter 38 before the terminal Real seal is read. In carrier terms, the rows* $Q, L, W, R, D, H, C, P, N$ *remain attached to one finite observation window throughout the readback.*

Proof. Start with window soundness, Definition 8466.2. It makes every Real-facing read expose Q, L, W, R, D before the structural rows are consumed.

Use the sibling chapters as the route sources. The finite stream chapter supplies the list prefix surface, StreamName supplies the finite window, RegSeqRat supplies regular readback, DyadicRatCore supplies the tolerance ledger, and Real supplies only the terminal seal reached through those rows.

The NameCert obligations of Definition 8466.3 then keep transport, replay, provenance, and local naming on the same packet. Hence the readback route is finite and displayed, not a host real, quotient equality, selected limit, or completeness claim. □

Falsifiable prediction: Any Real-facing read that omits the finite request row or list prefix row is not a FiniteRealObserverUp carrier read.

Independence witness: finite-stream, streamname, regseqrat, dyadicratcore, real: FiniteStreamUp supplies schedules without regular rational readback; StreamNameUp supplies windows without the full request and Real readback route; RegSeqRatUp supplies regular rational names without the observer schedule; DyadicRatCoreUp supplies tolerance rows without the Real-facing observation packet; RealUp supplies the seal boundary without the finite request ledger. None is bijective to $\mathsf{FiniteRealObserver}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealObserver}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRealObserver}^{\uparrow}$ packet is generated from a finite request row, list prefix schedule, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRealObserver}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8466.1, Definition 8466.2, Definition 8466.3, and Definition 8466.4 seed the finite observation-window bridge for Real and completion-phase readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRealObserver}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, Real completeness, quotient equality of names, selected limits, host real equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite request soundness, list-prefix accountability, StreamName window soundness, RegSeqRat readback, DyadicRatCore tolerance routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRealObserverUp.

Chapter 8467

A Concrete Naming Certificate for $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$ packet records an explicit-subsequence-selector compactness surface for located real intervals. It sits after the real-name carrier of chapter 38, the regular rational readback of chapter 301, the finite stream interface of chapter 126, the selector discipline of chapter 5902, and the located interval sequence route of chapter 8094. The packet does not claim open-cover Heine–Borel compactness, quotient real equality, selected representatives, countable choice, or general sequential compactness without displayed selector data.

Definition 8467.1 (Sequentially compact real interval carrier)**.** *A* $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, Q, M, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the located real interval row, recording endpoint names, locator data, and the closed interval boundary read through* $\mathsf{Real}^{\uparrow}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the input sequence of interval points and the finite windows inspected before a subsequence is accepted.*
3. *Q is the explicit selector row, naming the monotone subsequence index function and the finite evidence that each selected index is read from the input stream.*
4. *M is the Cauchy-modulus corridor row, recording the displayed tolerance schedule, interval-nesting or finite-net corridor, and the modulus data attached to the selected subsequence.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row, converting regular rational readback and terminal real-name sealing into the same interval packet.*
6. *H records componentwise* hsame *transport for interval, stream, selector, modulus, handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located interval and sequence request through the selector and Cauchy-modulus corridor.*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$*,* $\mathsf{LocatedCompactIntervalSequence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SequentiallyCompactRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, Q, M, R, H, C, P, N$. It has no coordinate for open-cover compactness, quotient equality of real names, hidden selected representatives, countable choice, or a general sequential-compactness principle detached from the displayed selector row.

Theorem 8467.2 (Sequentially compact real interval NameCert obligations)**.** *For every accepted* $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$ *carrier*

$$K = (I, S, Q, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentiallyCompactRealInterval}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *interval exposure: the located interval row I is displayed before any compactness read;*
2. *sequence exposure: the input sequence and inspected finite windows are exposed through S;*
3. *explicit selector exposure: the subsequence selector row Q displays the chosen index map and its monotonicity-facing data;*
4. *Cauchy-modulus corridor: the selected subsequence is routed through the modulus and finite corridor row M before convergence may be read;*
5. *Real-seal handoff: the regular rational readback and terminal real-name seal travel through R, H, C, P, N as the same local packet.*

Thus every accepted compactness consumer reads located interval data, sequence windows, an explicit selector, a Cauchy-modulus corridor, and a Real-facing seal in one finite route.

Proof. Project the carrier of Definition 8467.1. The rows $I, S, Q, M, R, H, C, P, N$ are the only accepted coordinates, so each consumer read begins with the located interval, sequence, selector, modulus corridor, and Real handoff rows.

Use the selector and located interval siblings next. The $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ route supplies displayed selector data, while chapter 8094 supplies the located finite-window interval corridor consumed by M.

Finally read the regular rational and Real rows through R, H, C, P, N. These structural rows transport, replay, source, and name the same packet; they do not add open-cover compactness, quotient equality, hidden representatives, countable choice, or selector-free sequential compactness. □

Falsifiable prediction: An accepted $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$ consumer that reaches a Real seal or Cauchy subsequence without displaying $I, S, Q, M, R, H, C, P, N$ refutes the carrier surface.

Independence witness: real, regseqrat, streamname, cauchy-subsequence-selector, located-compact-interval-sequence: no listed sibling supplies the joint finite packet containing located interval data, stream windows, explicit subsequence selector rows, Cauchy-modulus corridors, regular rational readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyCompactRealInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyCompactRealInterval}^{\uparrow}$ packet is generated from located real interval rows, stream-window sequence rows, explicit subsequence selector rows, Cauchy-modulus corridor rows, RegSeqRat and Real handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentiallyCompactRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8467.1 and Definition 8467.2 seed the explicit-selector compactness route for located real intervals.

Formal verification: FormalStatus(NameCert$_{\mathsf{SequentiallyCompactRealInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover Heine–Borel compactness, quotient real equality, selected representatives, countable choice, general sequential compactness without displayed selector data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval exposure, sequence-window exposure, explicit selector exactness, Cauchy-modulus corridor preservation, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.SequentiallyCompactRealIntervalUp.

Chapter 8468

A Concrete Naming Certificate for $\mathsf{TailEquivalenceModulus}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{TailEquivalenceModulus}^{\uparrow}$ packet records the shared finite modulus carried by two tail-equivalent regular Cauchy names before either name is read at a $\mathsf{Real}^{\uparrow}$ seal. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence classifier of chapter 5179, the reusable tail seal of chapter 8052, and the terminal real-name surface of chapter 38. The packet names the common modulus as displayed finite data; it is not quotient stream equality, a selected limit, countable choice over tails, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8468.1 (Tail-equivalence modulus carrier)**.** *A* $\mathsf{TailEquivalenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S_0, S_1, R_0, R_1, D, E, K, A, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which the regular Cauchy names are inspected.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed by the windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the displayed precision request, dyadic comparison budget, and slack left after the two regular radii are paid.*
4. E *is the* $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ *row witnessing that the two readbacks share a tail-equivalence comparison at the displayed precision.*
5. K *is the common finite modulus row assigning the precision in* D *to a single tail bound valid for both readbacks.*
6. A *is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after* $S_0, S_1, R_0, R_1, D, E, K$ *have been displayed.*
7. H *records componentwise* hsame *transport for the two window rows, two readback rows, dyadic ledger, equivalence row, common modulus, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a tail-equivalence request through* $S_0, S_1, R_0, R_1, D, E, K, A, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{TailEquivalenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$, $\mathsf{RegularCauchyTailSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{TailEquivalenceModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S_0, S_1, R_0, R_1, D, E, K, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8468.2 (Tail-equivalence modulus NameCert obligations)**.** *For every accepted* $\mathsf{TailEquivalenceModulus}^{\uparrow}$ *carrier*

$$M = (S_0, S_1, R_0, R_1, D, E, K, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TailEquivalenceModulus}^{\uparrow}}$ *surface consists exactly of two finite* $\mathsf{StreamName}^{\uparrow}$ *window admissions, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one* $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ *comparison row, one common finite modulus row, one* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8468.1. The accepted packet displays exactly the two window rows, two regular readbacks, dyadic ledger, tail-equivalence comparison, common modulus, Real seal, and structural rows.

The source side is exhausted by S_0, S_1, R_0, R_1. A consumer must inspect both finite stream windows and the corresponding regular rational readbacks before any common tail bound can be read.

The comparison side is exhausted by D, E, K. The dyadic ledger fixes the tolerance request, the tail-equivalence row records the displayed comparison between the readbacks, and the common modulus row assigns a single finite threshold to both names.

The endpoint and structural rows A, H, C, P, N seal, transport, replay, source, and name the same packet. Since quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, host real equality, and ambient completeness are absent from the carrier, they cannot appear as NameCert obligations. □

Theorem 8468.3 (Tail-equivalence modulus Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{TailEquivalenceModulus}^{\uparrow}$ *packet must read the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto E \longmapsto K \longmapsto A$$

before it may consume the terminal Real seal. The handoff exports the common tail modulus shared by the two regular Cauchy readbacks, not quotient equality of streams or a selected completion endpoint.

Proof. Begin with a Real-facing read. By Definition 8468.1, the read first reaches the two $\mathsf{StreamName}^{\uparrow}$ windows and their $\mathsf{RegSeqRat}^{\uparrow}$ readbacks.

The dyadic row D then fixes the tolerance budget for the comparison. The tail-equivalence row E is read only against that displayed budget, and the common modulus row K records the single tail bound shared by the two readbacks.

Only after K is visible does the terminal seal row A become available. The transport, replay, provenance, and local naming rows preserve this route without adding a quotient class, a selected representative, a selected limit, countable choice, trichotomy, host equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{TailEquivalenceModulus}^{\uparrow}$ read displays two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ row, one common finite modulus row, one $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed

Cont replay, Pkg provenance, and local naming before a Real-facing equality consumer can read the seal.

Independence witness: streamname, regseqrat, dyadicratcore, regular-cauchy-tail-equivalence, regular-cauchy-tail-seal, real: $\mathsf{TailEquivalenceModulus}^{\uparrow}$ is not bijective to any listed sibling because it binds two windows, two regular readbacks, one dyadic tolerance ledger, one tail-equivalence row, one shared modulus row, one Real seal, transport, replay, provenance, and local naming into a single finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{TailEquivalenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{TailEquivalenceModulus}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readbacks, one DyadicRatCore tolerance ledger, one RegularCauchyTailEquivalence comparison row, one common finite modulus row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailEquivalenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8468.1, Definition 8468.2, and Definition 8468.3 seed the finite shared-modulus surface between tail-equivalent regular Cauchy readbacks and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailEquivalenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked window admission, regular readback, dyadic tolerance accounting, tail-equivalence comparison, common-modulus preservation, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.TailEquivalenceModulusUp.

Chapter 8469

A Concrete Naming Certificate for $\mathsf{LocatedRealIntervalChain}^{\uparrow}$

Following Bishop and Bridges on located interval constructions, the $\mathsf{LocatedRealIntervalChain}^{\uparrow}$ packet records a finite chain of located real intervals as the displayed route by which interval refinements feed the Real and Completion stack. It sits between the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the closed interval boundary of chapter 1591. The packet names a finite located interval-chain certificate only; it does not assert interval compactness, choose an infinite branch, quotient real names, invoke trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8469.1 (Located real interval chain carrier)**.** *A* $\mathsf{LocatedRealIntervalChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite located interval-chain row. It lists the inspected interval cells, their endpoint order, their containment relation to the preceding cell, and the locatedness evidence available at each stage.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and endpoint-arithmetic ledger attached to the same cells.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for endpoint names, midpoint requests, and refinement stages.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the stream windows and dyadic tolerances of S and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the interval chain, dyadic ledger, stream window, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for the interval-chain row, dyadic ledger, stream windows, readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$I, D \longmapsto S, R \longmapsto E, H, P, N.$$

8. *P is the* Pkg *provenance over* LocatedRealIntervalChain$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{LocatedRealIntervalChain}^{\uparrow}}$ *row.*

The carrier exports only this displayed finite packet. It has no coordinate for a host interval intersection theorem, quotient equality of real names, a selected infinite branch, countable choice, trichotomy, or ambient completeness.

Theorem 8469.2 (Located real interval chain NameCert obligations)**.** *For every accepted* LocatedRealIntervalChain *carrier $L = (I, D, S, R, E, H, C, P, N)$, the local* NameCert$_{\text{LocatedRealIntervalChain}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *interval-chain admission: the finite located interval list, endpoint order, containment rows, and locatedness witnesses are displayed by I before any readback is allowed;*

2. *dyadic exactness: every diameter and endpoint-arithmetic request is read through the* DyadicRatCore$^{\uparrow}$ *ledger D attached to the same finite chain;*

3. *finite observation: endpoint and refinement-stage requests pass through the common* StreamName$^{\uparrow}$ *window S and* RegSeqRat$^{\uparrow}$ *readback row R before the terminal* Real$^{\uparrow}$ *seal E is consumed;*

4. *structural stability: H, C, P, N transport, replay, source, and locally name only the displayed rows of the carrier.*

No located real interval-chain obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8469.1. The located chain, dyadic diameter ledger, stream windows, regular readback, Real seal, and structural rows are all coordinates of one finite packet.

The dyadic exactness obligation reads only the row D. Since D is attached to the same finite interval cells named by I, a consumer cannot replace the chain by a host interval family, an infinite selector, or an ambient order-completeness principle.

The observation obligation is downstream. The stream row S exposes the inspected endpoint and refinement windows, the regular rational row R supplies readback for those windows, and the seal row E is available only after I, D, S, R have been displayed.

Finally H, C, P, N preserve that route under transport, replay, provenance, and local naming. Because no carrier coordinate contains quotient equality, choice, trichotomy, interval intersection, or ambient completeness, none of those principles is exported by the local certificate. □

Theorem 8469.3 (Located real interval chain nested refinement)**.** *Let $L = (I, D, S, R, E, H, C, P, N)$ be an accepted* LocatedRealIntervalChain$^{\uparrow}$ *carrier. Every refinement read of the chain must expose the finite interval row I and the dyadic diameter ledger D before it consumes the stream window S, the regular rational readback R, or the terminal* Real$^{\uparrow}$ *seal E. Thus a nested refinement is a displayed finite chain route through* ClosedBoundedInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *not a host-selected infinite intersection or quotient real endpoint.*

Proof. Start with the carrier definition. The only interval-chain coordinate is I, and the only diameter and endpoint-arithmetic coordinate is D.

Apply the NameCert obligations of Definition 8469.2. They force every refinement request to read the displayed interval-chain row and its dyadic ledger before any observation window or Real-facing seal is available.

The readback rows S, R, E are therefore downstream of the finite nested route. Componentwise transport and replay preserve the same ordering through H and C, while P and N record only the

displayed source and local name. No coordinate remains from which a consumer could recover a selected infinite branch, quotient endpoint equality, trichotomy, choice, or ambient completeness. □

Theorem 8469.4 (Located real interval chain Real seal route)**.** *Every Real and Completion consumer of an accepted* $\mathsf{LocatedRealIntervalChain}^{\uparrow}$ *carrier reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may transport and replay the displayed chain, but it cannot export interval compactness, countable choice, quotient equality of real names, selected completed endpoints, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8469.3. It places every refinement read behind the finite interval row I and dyadic ledger D.

Next use Definition 8469.2. Its finite-observation clause routes endpoint and refinement-stage reads through S and R before the seal row E can be consumed.

The sibling anchors identify the public surfaces of the route: closed bounded interval data supplies the interval boundary, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the diameter and endpoint arithmetic, $\mathsf{StreamName}^{\uparrow}$ supplies finite observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal. Since the carrier contains no compactness theorem, quotient equality, choice selector, trichotomy, or ambient completeness coordinate, the consumer receives only this finite seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalChain}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealIntervalChain}^{\uparrow}$ packet is generated from a finite located interval-chain row, a DyadicRatCore diameter and endpoint-arithmetic ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8469.1, Definition 8469.2, Definition 8469.3, and Definition 8469.4 seed the finite located interval-chain route into the Real and Completion stack.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close interval compactness, selected infinite intersections, quotient equality of real names, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.LocatedRealIntervalChainUp covering carrier admission, dyadic exactness, nested refinement ordering, readback routing, Real seal nonescape, transport, replay, provenance, and local naming.

Chapter 8470

A Concrete Naming Certificate for $\mathsf{CauchyGluingLemma}^{\uparrow}$

Following Bishop–Bridges constructive analysis on regular Cauchy names, the $\mathsf{CauchyGluingLemma}^{\uparrow}$ packet records the finite lemma surface by which two compatible regular rational windows are glued before a completion-facing Real seal is exposed. It sits after the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance agreement of chapter 291, the downstream gluing row of chapter 5899, and the terminal Real seal of chapter 38. The packet is a finite NameCert reconstruction of the usual constructive gluing lemma; it is not quotient stream equality, a selected common representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8470.1 (Cauchy gluing lemma carrier)**.** *A* $\mathsf{CauchyGluingLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S_0, S_1, W_0, W_1, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are compatible finite* $\mathsf{StreamName}^{\uparrow}$ *windows naming the two local regular Cauchy observations and their common overlap schedule.*
2. *W_0 and W_1 are the two local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback windows whose rational tails are compared only on the displayed overlap.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-agreement row tying the two windows to the same dyadic request and the same finite overlap budget.*
4. *R is the downstream* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *row that consumes the two local windows, the shared tolerance ledger, and the scheduled overlap before any glued readback is public.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream schedules, regular readbacks, dyadic agreement, and gluing row have been replayed.*
6. *H records componentwise* hsame *transport for the two schedules, two readbacks, dyadic ledger, gluing row, Real seal, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from a requested glued-window read through $S_0, S_1, W_0, W_1, D, R, E, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyGluingLemma}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyGluingLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected representatives, countable choice, host Real equality, hidden completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8470.2 (Cauchy gluing lemma NameCert obligations)**.** *For every accepted* $\mathsf{CauchyGluingLemma}^{\uparrow}$ *carrier*

$$G = (S_0, S_1, W_0, W_1, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyGluingLemma}^{\uparrow}}$ *surface consists exactly of carrier admission, overlap compatibility, regular-readback preservation, gluing handoff, Real-seal nonescape, transport, replay, provenance, and local naming. The carrier admission exposes the two* $\mathsf{StreamName}^{\uparrow}$ *schedules, two* $\mathsf{RegSeqRat}^{\uparrow}$ *windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-agreement ledger, one* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer row, one terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows* H, C, P, N*. No local obligation is stored outside these rows.*

Proof. Project the carrier of Definition 8470.1. The rows $S_0, S_1, W_0, W_1, D, R, E, H, C, P, N$ are the only visible coordinates, so carrier admission and local naming are immediate projections.

Overlap compatibility is the row D read together with the scheduled windows S_0, S_1. Regular-readback preservation is the pair W_0, W_1 read through the same dyadic request, so the two local rational windows cannot be replaced by quotient stream equality or selected representatives.

The gluing handoff is the row R. By the sibling route of chapter 5899, R consumes the displayed schedules, readbacks, and tolerance ledger before exposing any glued regular-Cauchy read.

Finally E is downstream of that handoff. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so a Real-facing consumer receives only the listed rows and none of the refused host or quotient principles. □

Theorem 8470.3 (Cauchy gluing lemma Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyGluingLemma}^{\uparrow}$ *carrier must pass through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyGluing}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer first reads the two finite stream windows, then the two regular rational readbacks, then the shared dyadic tolerance agreement, then the regular-Cauchy gluing row, and only then the terminal Real seal. The route exports no quotient stream equality, selected common representative, countable choice, host Real equality, hidden completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.CauchyGluingLemmaUp`

Proof. Let $G = (S_0, S_1, W_0, W_1, D, R, E, H, C, P, N)$ be accepted. Any Real-facing read must end at E, and E is available only after the carrier's displayed replay row C has passed through the earlier rows.

The earlier rows are ordered by their dependencies. The schedules S_0, S_1 name the finite observations, W_0, W_1 give the regular rational readbacks on those observations, and D records the shared dyadic tolerance agreement for the overlap.

The downstream gluing row R is then the only row that can turn the two local windows into one glued regular-Cauchy read. Since the terminal seal E is read after R, every Real-facing consumer factors through the displayed route.

The carrier contains no coordinate for the refused endpoints. Componentwise transport, replay, provenance, and local naming preserve the same route rather than adding a quotient, selector, choice principle, host equality, hidden completion, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{CauchyGluingLemma}^{\uparrow}$ Real-facing read factors through two $\mathsf{StreamName}^{\uparrow}$ finite windows, two $\mathsf{RegSeqRat}^{\uparrow}$ readback windows, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-agreement row, one $\mathsf{RegularCauchyGluing}^{\uparrow}$ handoff, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, regularcauchygluing, real: no listed sibling is bijective to $\mathsf{CauchyGluingLemma}^{\uparrow}$ because this packet jointly carries two compatible finite windows, two regular readbacks, one dyadic agreement ledger, a downstream gluing row, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyGluingLemma}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyGluingLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8470.1, Definition 8470.2, and Definition 8470.3 seed the finite gluing-lemma route over two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance agreement, a RegularCauchyGluing handoff, a Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyGluingLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host Real equality, hidden completion, ambient RealUp completeness, Lean theorem verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, overlap compatibility, regular-readback preservation, gluing handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyGluingLemmaUp.

Constructive story: A $\mathsf{CauchyGluingLemma}^{\uparrow}$ packet is generated from two compatible finite StreamName windows, two RegSeqRat readback windows, one DyadicRatCore tolerance-agreement row, one RegularCauchyGluing handoff row, one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8471

A Concrete Naming Certificate for $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed regular sequence over the real-completion route supplies a limit-facing read only after its StreamName windows, RegSeqRat readbacks, dyadic tolerance rows, sequential-completeness witness, and completion-operator handoff are visible. It sits after the terminal real-name seal of chapter 38, the finite window surface of chapter 126, the regular rational readback of chapter 301, the sequence-completeness interface of chapter 1515, and the completion-operator route of chapter 3379. The packet is a Bishop-style sequential limit criterion for the Real and Completion phase; it is not a proof of $\mathsf{Real}^{\uparrow}$ completeness, a quotient equality of streams, countable choice, trichotomy, selected representatives, or bridge checking.

Definition 8471.1 (Bishop sequential limit criterion carrier)**.** *A* $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, O, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, exposing the finite sequence observations and tolerance requests opened by the limit read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows of S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording requested radii, split tolerances, slack cells, endpoint observations, and comparison budgets.*
4. *Q is the* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *row that carries the sequence-completeness witness for the displayed metric-side sequence without importing an ambient completeness theorem.*
5. *O is the* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *route row, read only after S, R, D, Q have fixed the finite window and completion-boundary data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the criterion after the sequential-completeness and completion-operator rows are displayed.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, sequential-completeness, completion-operator, real-seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$S, R, D \longmapsto Q \longmapsto O \longmapsto E.$$

9. *P is the* Pkg *provenance over* BishopSequentialLimitCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, SequentiallyCompleteMetric$^{\uparrow}$, CauchyCompletionOperator$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopSequentialLimitCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, O, E, H, C, P, N$. It has no coordinate for Real$^{\uparrow}$ *completeness, quotient stream equality, countable choice, trichotomy, selected representatives, selected limits outside the displayed packet, host real equality, or bridge checking.*

Theorem 8471.2 (Bishop sequential limit criterion NameCert obligations)**.** *For every accepted* BishopSequentialLimitCriterion$^{\uparrow}$ *carrier*

$$L = (S, R, D, Q, O, E, H, C, P, N),$$

the local NameCert$_{\text{BishopSequentialLimitCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, SequentiallyCompleteMetric row, CauchyCompletionOperator route, Real seal, componentwise transport, displayed replay, provenance, and local naming;*
2. *window-readback exactness, requiring R to read only the finite observations displayed by S;*
3. *tolerance exactness, requiring D to display the dyadic radii, split tolerances, endpoint observations, slack cells, and comparison budgets before any sequential-completeness or completion-operator consumer reads the route;*
4. *sequential-completeness handoff, requiring Q to receive only the displayed S, R, D evidence and not an ambient completeness theorem;*
5. *completion-operator compatibility, requiring O to be downstream of the same S, R, D, Q route;*
6. *Real seal nonescape, requiring E to be reached only after the displayed sequential and completion rows have been read;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed route.*

No Bishop sequential limit criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8471.1. The accepted packet displays exactly the StreamName window row, regular readback, dyadic tolerance ledger, sequential-completeness row, completion-operator route, Real seal, and structural rows.

Window-readback and tolerance exactness are local to S, R, D. The readback row is attached to the displayed windows, and the dyadic ledger supplies all requested radii, endpoint observations, split tolerances, slack cells, and comparison budgets before the later rows may be consumed.

The sequential-completeness row Q receives only that displayed evidence. It names the sequence-completeness handoff for the given finite route rather than an external theorem that all real sequences have limits.

The completion-operator row O is downstream of the same route. It may read the completion boundary only after S, R, D, Q have fixed the finite window, readback, tolerance, and sequence-completeness data.

Finally E, H, C, P, N seal, transport, replay, source, and name the route. Since the carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, countable choice, trichotomy, selected representatives, selected limits outside the displayed packet, host real equality, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 8471.3 (Bishop sequential limit criterion Real-completion route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto Q \longmapsto O \longmapsto E \longmapsto H, C, P, N.$$

The consumer first reads the StreamName windows, then the RegSeqRat readback and DyadicRatCore tolerance ledger, then the SequentiallyCompleteMetric witness and CauchyCompletionOperator route, and only then the terminal Real seal and structural rows. The route exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, countable choice, trichotomy, selected representative, host real equality, or bridge checking.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ or completion-facing read of an accepted carrier. Projection exposes the ordered rows $S, R, D, Q, O, E, H, C, P, N$ and no additional analytic coordinate.

The rows S, R, D provide the finite observational base. A consumer cannot start from a quotient stream or an unbounded host sequence because the readback and tolerance evidence are attached to the displayed windows.

The sequential-completeness row Q is read next. It consumes the finite window, readback, and dyadic ledger as a displayed sequence-completeness witness for this packet, not as a global completeness claim.

The completion-operator row O then provides the completion-facing handoff, and the terminal seal E is available only after that handoff. The structural rows preserve the same path under transport, replay, provenance, and local naming.

The excluded principles are absent from the carrier inventory. Therefore every Real or completion consumer factors through the displayed route and receives none of the listed escape data. □

Falsifiable prediction: An accepted $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$ consumer is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing seal without displaying the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, SequentiallyCompleteMetric row, CauchyCompletionOperator route, transport row, replay row, provenance row, and local naming row.

Independence witness: streamname, regseqrat, dyadicratcore, sequentially-complete-metric, cauchycompletionoperator, real: $\mathsf{BishopSequentialLimitCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite sequence windows, regular rational readback, dyadic tolerance accounting, sequential-completeness handoff, completion-operator routing, Real sealing, componentwise transport, replay, provenance, and local naming as one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSequentialLimitCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSequentialLimitCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8471.1, Definition 8471.2, and Definition 8471.3 seed the Bishop sequential limit route through StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, SequentiallyCompleteMetric handoff, CauchyCompletionOperator routing, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSequentialLimitCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, countable choice, trichotomy, selected representatives, selected limits outside the displayed packet, host real equality, Lean checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, tolerance exactness, sequential-completeness handoff, completion-operator compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopSequentialLimitCriterionUp.

Constructive story: A BishopSequentialLimitCriterion$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a SequentiallyCompleteMetric handoff, a CauchyCompletionOperator route, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8472

A Concrete Naming Certificate for $\mathsf{FiniteCauchyTailHandoff}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real construction, the $\mathsf{FiniteCauchyTailHandoff}^{\uparrow}$ packet records the finite tail-window interface where regular rational names, finite stream requests, dyadic tolerance ledgers, and terminal real-name seals meet. It sits over the regular-rational readback of chapter 301, the finite request surface of chapter 126, the dyadic arithmetic rows of chapter 291, the real-name seal of chapter 38, the tail selector of chapter 458, and the modulus arithmetic of chapter 1485. The packet gives the L10 Real and Completion route a named handoff between tail evidence and Real sealing without quotient equality, selected limits, trichotomy, countable choice, or ambient completeness.

Definition 8472.1 (Finite Cauchy tail handoff carrier)**.** *A* $\mathsf{FiniteCauchyTailHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, W, D, M, S, Z, H, K, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row whose tail is being offered to a Real-facing consumer;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite request window over which the tail is inspected;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying the displayed radius, precision, and comparison budget for the window;*
4. *M is the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *row computing the tail bound from the request window and dyadic ledger;*
5. *S is the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *row selecting the finite tail segment licensed by M;*
6. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal that may consume the displayed tail handoff;*
7. *H records componentwise* hsame *transports for the regular readback, window, dyadic ledger, modulus arithmetic, tail selector, real seal, provenance, and local name;*
8. *K records displayed* Cont *routes from R, W, D through M, S to Z and from the structural rows back to the local certificate;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyTailHandoff}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, a host-selected Cauchy tail, a selected real limit, trichotomy, countable choice, or an ambient completeness principle.

Lean4 (statement) `BEDC.Derived.FiniteCauchyTailHandoffUp`

Theorem 8472.2 (Finite Cauchy tail handoff NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyTailHandoff}^{\uparrow}$ *carrier* $T = (R, W, D, M, S, Z, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyTailHandoff}^{\uparrow}}$ *surface consists exactly of four public obligation rows: regular-rational tail source through* R*, finite window and dyadic budget through* W, D*, computed modulus and tail selector through* M, S*, and terminal real sealing through* Z*. The structural rows* H, K, P, N *preserve, replay, source, and locally name only those displayed rows, so no other tail witness or Real-facing evidence is available.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyTailHandoffUp.FiniteCauchyTailHandoffNameCertObligations`

Proof. Project the finite packet of Definition 8472.1. Its analytic rows are exactly the regular readback R, window W, dyadic ledger D, modulus arithmetic row M, tail selector S, and real seal Z.

The first obligation exposes the $\mathsf{RegSeqRat}^{\uparrow}$ tail source. The second obligation fixes the finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ budget. The third obligation computes the displayed tail bound and selector through M, S. The fourth obligation seals the resulting handoff through Z.

The rows H, K, P, N transport, replay, package, and name the same packet. Since the carrier omits quotient equality, selected limits, choice schedules, trichotomy, and ambient completeness, the local certificate has no other public obligation row. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyTailHandoffNameCertObligations`

Theorem 8472.3 (Finite Cauchy tail handoff Real seal route)**.** *Every Real-facing read of an accepted* $\mathsf{FiniteCauchyTailHandoff}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal* Z *only after passing through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the common* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *row* M*, the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *row* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* K*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyTailHandoff}^{\uparrow}}$ *row* N*. It cannot replace this finite route by quotient equality, host real equality, a selected limit, trichotomy, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyTailHandoffUp.FiniteCauchyTailHandoffRealSealRoute`

Proof. Start with the NameCert obligations of Definition 8472.2. They place the tail source, window, dyadic ledger, modulus arithmetic, tail selector, and Real seal in one displayed finite packet.

The route to Z must therefore pass through R, W, D, M, S first. The Cont row K records that order, while the hsame row H can only transport the displayed components of the same carrier.

Finally, P and N provide provenance and local naming for that carrier. Because no omitted coordinate is present in Definition 8472.1, a Real-facing consumer cannot cite a quotient equality, host equality, selected limit, choice schedule, trichotomy branch, or ambient completeness witness as part of the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyTailHandoff}^{\uparrow}}$**.**

Constructive story: A FiniteCauchyTailHandoff$^{\uparrow}$ packet is generated from a RegSeqRat readback, a StreamName window, a DyadicRatCore ledger, CauchyModulusArithmetic, RegularCauchyTailSelector, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{FiniteCauchyTailHandoff}\uparrow}$, seed)

Scope closed: Definition 8472.1, Definition 8472.2, and Definition 8472.3 bind the finite tail-window route from regular rational readback to terminal Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{FiniteCauchyTailHandoff}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.FiniteCauchyTailHandoffUp`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, host real equality, selected limits, trichotomy, countable choice, ambient completeness, theorem checking, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window determinacy, Real seal non-escape, and structural route preservation for the same finite packet.

Chapter 8473

A Concrete Naming Certificate for $\mathsf{RegularCauchyRecursion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyRecursion}^{\uparrow}$ packet records the finite evaluator surface by which a regular rational Cauchy name is read through displayed precision windows, dyadic tolerance rows, and a recursion ledger before a $\mathsf{Real}^{\uparrow}$ seal is available. It sits over the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance core of chapter 291, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational source rows of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion eliminator of chapter 3628. The chapter names a finite L10 evaluator route; it does not identify quotient streams, select limits, compare host functions by equality, invoke countable choice, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8473.1 (Regular Cauchy recursion carrier)**.** *A* $\mathsf{RegularCauchyRecursion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, S, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational entries are the only values read by the evaluator.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row recording the finite observation windows used for each precision request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording dyadic radii, comparison cells, and the requested error budget.*
4. *E is the evaluator and recursion ledger, recording each finite call, its source window, its dyadic tolerance, and its dependency on earlier ledger rows.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, S, D, E have been displayed.*
6. *H records componentwise* hsame *transport for the rational source, window, tolerance, evaluator ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for each evaluator request through Q, S, D, E, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRecursion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRecursion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, S, D, E, L, H, C, P, N$. No coordinate stores a quotient stream, a selected limit, host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRecursionUp.RegularCauchyRecursionTasteGate single carrier alignment`

Theorem 8473.2 (Regular Cauchy recursion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRecursion}^{\uparrow}$ *carrier*

$$R = (Q, S, D, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRecursion}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, evaluator and recursion ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every evaluator read must expose the source row Q, window row S, tolerance row D, and ledger row E before any Real-facing seal is visible.*

Proof. Project the carrier of Definition 8473.1. Its displayed coordinates are exactly $Q, S, D, E, L, H, C, P, N$, so local naming and carrier admission have no hidden analytic source.

Read the analytic rows in order. The rational source Q supplies only regular rational entries, the window row S fixes the finite observations, and the dyadic row D fixes the requested tolerance budget.

The evaluator ledger E is downstream of those rows. It records finite recursive calls and their dependencies, but it does not add a quotient stream, selected limit, host function equality, choice schedule, or completeness principle.

Finally read the structural rows H, C, P, N and the seal row L. Transport, replay, provenance, and local naming preserve the same finite route, so the listed rows exhaust the NameCert surface. □

Theorem 8473.3 (Regular Cauchy recursion evaluation handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyRecursion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyCompletionRecursor}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The evaluator ledger may be handed to the $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ *route only as displayed regular-Cauchy recursion data over Q, S, D, E, H, C, P, N. The handoff exports no quotient-stream equality, selected representative, selected limit, host function equality, countable choice, or ambient Real completeness.*

Proof. Start from the NameCert obligations of Definition 8473.2. They force every evaluator request to expose the regular rational source, finite windows, dyadic tolerance row, and evaluator ledger before the Real seal is read.

The completion recursor then receives only the displayed ledger. Its input is the finite route through Q, S, D, E together with transport, replay, provenance, and local naming; it is not a separate limit selector or quotient representative.

Since quotient streams, selected representatives, selected limits, host function equality, countable choice, and ambient Real completeness have no coordinate in the carrier, no downstream consumer can extract them from the handoff. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyRecursion}^{\uparrow}$ consumer that reads a $\mathsf{Real}^{\uparrow}$ seal or a $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ handoff without displaying the $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, evaluator ledger, transport, replay, provenance, and local naming evidence refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-completion-recursor: no listed sibling supplies the whole packet $R = (Q, S, D, E, L, H, C, P, N)$, because each sibling contributes only a tolerance, window, rational-source, Real-seal, or completion-elimination surface while this chapter binds all five with an evaluator ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRecursion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRecursion}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName windows, DyadicRatCore tolerances, an evaluator and recursion ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRecursion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8473.1, Definition 8473.2, and Definition 8473.3 seed the finite L10 regular-Cauchy evaluator route over rational-source rows, finite windows, dyadic tolerances, completion-recursion handoff, Real sealing, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRecursion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host function equality, countable choice, ambient Real completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-row exactness, window exposure, dyadic tolerance exactness, evaluator ledger replay, completion-recursion handoff, Real seal nonescape, transport, provenance, and local naming for BEDC.Derived.RegularCauchyRecursionUp.

Chapter 8474

A Concrete Naming Certificate for $\mathsf{FunctionSequence}^{\uparrow}$

Following Bishop and Bridges on displayed sequences of maps, the $\mathsf{FunctionSequence}^{\uparrow}$ packet records the finite route through which a reader inspects a sequence of functions before uniform, pointwise, compactness, or limit-continuity consumers read the resulting value rows. It sits beside the continuous-map carrier of chapter 130, the stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name carrier of chapter 38, the uniform-convergence route of chapter 3417, and the uniform-limit continuity surface of chapter 1751. The packet is a shared BEDC carrier for displayed sequences of maps; it is not a host sequence of functions, quotient equality of maps, selected representatives, countable choice, or ambient completeness.

Definition 8474.1 (Function sequence carrier)**.** *A* $\mathsf{FunctionSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, F, A, V, W, H, C, P, L)$$

with the following displayed rows:

1. *X is the source-carrier row naming the domain and codomain surfaces through which the displayed maps are read;*
2. *F is the indexed function-graph row, listing the inspected stage names and their graph rows through* $\mathsf{ContinuousMap}^{\uparrow}$ *or the relevant displayed map carrier;*
3. *A is the finite argument-window row, recording the source arguments inspected at the displayed stages;*
4. *V is the value readback row, routing each inspected value through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal when the codomain is Real-facing;*
5. *W is the uniform-window ledger, recording finite stage and argument windows for uniform, compact, or limit-continuity consumers;*
6. *H records componentwise* hsame *transport for the source carrier, indexed graph rows, argument windows, value readbacks, uniform-window ledger, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a stage-and-argument request through X, F, A, V, W, H, P, L;*
8. *P is the* Pkg *provenance over* $\mathsf{FunctionSequence}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{UniformLimitContinuity}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. L is the local $\mathsf{NameCert}_{\mathsf{FunctionSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for host function extensionality, quotient equality of maps, an infinite uninspected schedule, selected representatives, countable choice, pointwise-only convergence detached from the value readback row, or ambient completeness.

Theorem 8474.2 (Function sequence NameCert obligations)**.** *For every accepted* $\mathsf{FunctionSequence}^{\uparrow}$ *carrier* $S = (X, F, A, V, W, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{FunctionSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source carrier, indexed function-graph rows, finite argument windows, value readback rows, uniform-window ledger, componentwise transport, displayed replay, provenance, and local naming; value readback through* A, V*; uniform-window exposure through* W*; and refusal of host function equality, quotient map equality, uninspected infinite schedules, selected representatives, countable choice, pointwise-only detached convergence, and ambient completeness.*

Proof. Project the carrier of Definition 8474.1. The source and analytic rows are X, F, A, V, W, and the structural rows are H, C, P, L.

The value readback obligation uses A and V. A consumer first chooses a displayed stage and argument window, then reads the value row through the regular rational and Real-facing surfaces supplied by the packet.

The uniform-window obligation uses W. Uniform convergence, compactness, and uniform-limit continuity consumers may request finite stage and argument windows, but the requested window remains attached to the same indexed graph rows and value readbacks.

Finally inspect H, C, P, L. These rows transport, replay, source, and name only the displayed packet. Since the refused host-side objects are not carrier coordinates, they cannot discharge a local NameCert obligation for $\mathsf{FunctionSequence}^{\uparrow}$. □

Theorem 8474.3 (Function sequence L10 route)**.** *Every accepted* $\mathsf{FunctionSequence}^{\uparrow}$ *read factors through the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

whenever a Real-facing value is inspected. Uniform-convergence and uniform-limit-continuity consumers may read W *only after the source carrier* X*, indexed graph row* F*, argument window* A*, value readback row* V*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* L *remain visible.*

Proof. Start with the NameCert obligations of Definition 8474.2. They force any consumer to expose the source carrier, indexed graph rows, argument windows, value readback rows, and uniform-window ledger before the packet is named.

Read the L10 siblings in order. The stream schedule supplies the inspected stage row, the continuous-map carrier supplies the displayed graph row, the regular rational surface supplies the value readback, and the Real carrier supplies only the terminal seal reached through those rows.

For a uniform or limit-continuity consumer, the row W adds a finite stage-and-argument ledger without detaching the read from X, F, A, V. The structural rows H, C, P, L preserve the same packet under transport, replay, provenance, and local naming, so the route cannot become a host-level sequence of functions or a quotient equality of maps. □

Falsifiable prediction: A uniform-convergence or uniform-limit-continuity read that obtains a value row without displaying the indexed graph row and finite argument window is not a $\mathsf{FunctionSequence}^{\uparrow}$ carrier read.

Independence witness: continuous-map, streamname, regseqrat, real, uniform-convergence, uniform-limit-continuity: ContinuousMapUp names one displayed map without the sequence schedule; StreamNameUp supplies schedules without map graphs; RegSeqRatUp supplies regular rational readback without the source and map rows; RealUp supplies the terminal seal without the sequence ledger; UniformConvergenceUp and UniformLimitContinuityUp consume finite uniform windows but do not by themselves name the shared displayed sequence-of-maps carrier. None is bijective to $\mathsf{FunctionSequence}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{FunctionSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{FunctionSequence}^{\uparrow}$ packet is generated from a source-carrier row, indexed function-graph rows, finite argument windows, value readback rows, a uniform-window ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FunctionSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8474.1, Definition 8474.2, and Definition 8474.3 seed the shared displayed sequence-of-maps carrier for ContinuousMapUp, StreamNameUp, RegSeqRatUp, RealUp, UniformConvergenceUp, UniformLimitContinuityUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FunctionSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host function extensionality, quotient equality of maps, selected representatives, countable choice, pointwise-only detached convergence, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, value readback, uniform-window exposure, L10 route preservation, transport, replay, provenance, and local naming for BEDC.Derived.FunctionSequenceUp.

Chapter 8475

A Concrete Naming Certificate for $\mathsf{SobolevSlobodeckij}^{\uparrow}$

Following Triebel and Adams–Fournier on fractional Sobolev and Slobodeckij regularity, the $\mathsf{SobolevSlobodeckij}^{\uparrow}$ packet records the finite Gagliardo-seminorm surface by which a non-integer smoothness read is exposed before any function-space quotient or ambient completeness theorem can be cited. It sits beside the Sobolev-space carrier of chapter 5911, the fractional Sobolev packet of chapter 6051, the Holder-space boundary of chapter 7000, and the Besov-space scale packet of chapter 4204. The chapter names only displayed function windows, dyadic pair ledgers, exponent data, seminorm readbacks, structural transport, replay, provenance, and local naming evidence.

Definition 8475.1 (Sobolev-Slobodeckij carrier)**.** *A* $\mathsf{SobolevSlobodeckij}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, F, p, s, Q, G, R, B, H, C, P, N)$$

with the following displayed rows:

1. *U is the source* $\mathsf{SobolevSpace}^{\uparrow}$ *or* $\mathsf{FractionalSobolev}^{\uparrow}$ *row whose weak or fractional regularity window is being inspected.*
2. *F records the finite function-value windows used by the seminorm read.*
3. *p and s are rational exponent and smoothness rows, with $0 < s < 1$ recorded as displayed order data.*
4. *Q is the finite pair ledger naming the inspected point-pair or dyadic cell-pair requests.*
5. *G is the Gagliardo-seminorm ledger, recording the displayed difference quotient budget over Q, F, p, s.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite seminorm estimates.*
7. *B is the handoff row to* $\mathsf{HolderSpace}^{\uparrow}$ *or* $\mathsf{BesovSpace}^{\uparrow}$ *consumers when their scale boundary is read.*
8. *H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SobolevSlobodeckij}^{\uparrow}}$ *evidence.*

The classifier compares only $U, F, p, s, Q, G, R, B, H, C, P, N$. It has no coordinate for quotient function equality, almost-everywhere representatives, selected completions, unrestricted integration theory, or ambient Banach-space completeness.

Theorem 8475.2 (Sobolev-Slobodeckij NameCert obligations)**.** *For every accepted* $\mathsf{SobolevSlobodeckij}^{\uparrow}$ *carrier* $S = (U, F, p, s, Q, G, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SobolevSlobodeckij}^{\uparrow}}$ *surface consists exactly of carrier admission, source Sobolev or fractional-Sobolev exposure, function-window exposure, exponent and smoothness-row exactness, pair-ledger coverage, Gagliardo-seminorm ledger exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Holder/Besov handoff exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8475.1. The analytic rows are precisely U, F, p, s, Q, G, R, B, and the structural rows are H, C, P, N.

The source regularity is read from U, while the finite windows and pair requests are read from F and Q. The exponent rows p, s and the ledger G record the Gagliardo-seminorm budget over exactly those displayed requests.

The readback row R transports the finite estimates to regular rational data, and the handoff row B exposes only the Holder or Besov boundary named by the sibling chapters. Since every remaining operation is componentwise transport, replay, provenance, or naming, the listed rows exhaust the local NameCert obligations. □

Theorem 8475.3 (Sobolev-Slobodeckij Gagliardo-seminorm boundary)**.** *Every downstream regularity consumer of an accepted* $\mathsf{SobolevSlobodeckij}^{\uparrow}$ *packet must read the Gagliardo-seminorm ledger* G *through the displayed source row* U*, function windows* F*, exponent rows* p, s*, pair ledger* Q*, and regular rational readback* R *before it receives any seminorm bound. The exported bound is therefore a finite BEDC row over the displayed pair ledger, not a host integral, quotient representative, or completed function-space norm.*

Proof. Begin with the NameCert obligations of Definition 8475.2. They make U, F, p, s, Q, G, R visible before any consumer can cite a seminorm estimate.

The ledger G is indexed by the displayed pair requests in Q and by the finite function windows F. Thus a seminorm read cannot replace those rows by an ambient integral domain, an almost-everywhere representative, or an unrestricted quotient function.

The readback R supplies the regular rational boundary for the estimate. The structural rows H, C, P, N preserve the same displayed route, so every exported Gagliardo-seminorm bound is attached to the carrier surface. □

Theorem 8475.4 (Sobolev-Slobodeckij Holder and Besov handoff)**.** *Any* $\mathsf{HolderSpace}^{\uparrow}$ *or* $\mathsf{BesovSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{SobolevSlobodeckij}^{\uparrow}$ *packet must first read the same source row* U*, function windows* F*, exponent data* p, s*, pair ledger* Q*, Gagliardo-seminorm ledger* G*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*. The handoff row* B *then exposes a regularity-scale boundary compatible with chapter 7000 and chapter 4204; it does not export an independent Holder modulus, Besov decomposition, quotient function identity, or selected completion point.*

Proof. Project the accepted packet and apply Definition 8475.3. Every scale-facing consumer has already read the source, window, exponent, pair-ledger, seminorm, and readback rows.

The handoff row B has no data outside that route. It merely names the boundary by which the displayed Slobodeckij seminorm can be consumed by the Holder-space or Besov-space sibling surface.

Because quotient identities, selected representatives, completed function-space points, and independent scale decompositions are absent from the carrier, neither the Holder nor Besov consumer can receive such data from the handoff. □

Falsifiable prediction: An accepted $\mathsf{SobolevSlobodeckij}^{\uparrow}$ consumer that cites a Gagliardo-seminorm, Holder boundary, or Besov boundary without reading the displayed source row, finite windows, exponent rows, pair ledger, seminorm ledger, regular rational readback, structural transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: sobolev-space, fractional-sobolev, holder-space, besov-space: no listed sibling supplies the whole packet $S = (U, F, p, s, Q, G, R, B, H, C, P, N)$, because SobolevSpace supplies the weak-source row, FractionalSobolev supplies a broader non-integer smoothness surface, HolderSpace supplies a power-law target boundary, and BesovSpace supplies a scale packet while this chapter binds the Gagliardo-seminorm pair ledger as its own finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevSlobodeckij}^{\uparrow}}$.

Constructive story: A $\mathsf{SobolevSlobodeckij}^{\uparrow}$ packet is generated from a SobolevSpace or FractionalSobolev source row, finite function windows, rational exponent and smoothness rows, a finite pair ledger, a Gagliardo-seminorm ledger, RegSeqRat readback, HolderSpace or BesovSpace handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobolevSlobodeckij}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8475.1, Definition 8475.2, Definition 8475.3, and Definition 8475.4 seed the finite Gagliardo-seminorm carrier and its Holder/Besov handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobolevSlobodeckij}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient function equality, almost-everywhere representative selection, host integration completeness, Lean verification, bridge checking, or ambient Banach-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, function-window exposure, exponent and smoothness exactness, pair-ledger coverage, Gagliardo-seminorm ledger exactness, regular rational readback, handoff exposure, transport, replay, provenance, and local naming.

Chapter 8476

A Concrete Naming Certificate for $\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}$

Following Lebesgue and the Hardy–Littlewood differentiation tradition, the $\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}$ packet records the theorem-level finite route by which local differentiation rows are read after the basis, maximal, covering, measure, and real-name packets have all been displayed. It consumes the differentiation carrier of chapter 4153, the basis carrier of chapter 6706, the maximal-function boundary of chapter 3581, the finite Vitali covering ledger of chapter 2166, the measurable source rows of chapter 92, and the terminal real-name seal of chapter 38. The packet names one finite theorem route; it is not measure completion, a selected almost-everywhere representative, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8476.1 (Lebesgue differentiation theorem carrier)**.** *A* $\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, B, H, V, M, R, A, E, Q, C, P, N)$$

with the following displayed rows:

1. *L is the accepted* $\mathsf{LebesgueDifferentiation}^{\uparrow}$ *row, exposing the finite local differentiation certificate.*
2. *B is the accepted* $\mathsf{LebesgueDifferentiationBasis}^{\uparrow}$ *row, exposing basis cells, finite selection, finite overlap, local-average, tolerance, and seal rows.*
3. *H is the* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *domination row used before an exceptional-set or local-average verdict is consumed.*
4. *V is the* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *ledger, recording finite disjoint subfamilies, cover cells, and overlap budgets used by B and L.*
5. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row for the inspected measurable requests and local-measure windows.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for local values, averages, radii, and error budgets.*
7. *A is the theorem-readback ledger, listing the finite route from basis admission through maximal domination and Vitali selection to the differentiation verdict.*
8. *E is the exclusion ledger, recording that the theorem route does not use measure completion, almost-everywhere representative selection, quotient function equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

9. *Q records componentwise* hsame *transport for differentiation, basis, maximal, Vitali, measure, real, readback, exclusion, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from theorem requests through $L, B, H, V, M, R, A, E, Q, P, N$.*
11. *P is the* Pkg *provenance over* LebesgueDifferentiationTheorem$^{\uparrow}$, LebesgueDifferentiation$^{\uparrow}$, LebesgueDifferentiationB HardyLittlewoodMaximal$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, Measure$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted theorem packets only through $L, B, H, V, M, R, A, E, Q, C, P, N$. It has no coordinate for measure completion, selected almost-everywhere representatives, quotient function equality, countable-choice schedules, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8476.2 (Lebesgue differentiation theorem NameCert obligations)**.** *For every accepted* LebesgueDifferentiationTheorem$^{\uparrow}$ *carrier*

$$T = (L, B, H, V, M, R, A, E, Q, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the differentiation row, basis row, maximal-domination row, finite Vitali-cover ledger, measure source row, Real seal row, theorem-readback ledger, exclusion ledger, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8476.1. The displayed rows $L, B, H, V, M, R, A, E, Q, C, P, N$ are the whole theorem-level packet.

The differentiation and basis rows are read first. The row L supplies the local differentiation certificate of chapter 4153, while B supplies the basis and finite-cover handoff of chapter 6706.

The maximal and covering rows then constrain the theorem read. The row H records the Hardy–Littlewood domination boundary, V records the finite Vitali ledger, and M, R keep the measurable source and Real seal visible before any differentiation verdict is consumed.

Finally A, E, Q, C, P, N record the theorem route, excluded endpoints, componentwise transport, replay, provenance, and local name. Since the excluded objects have no carrier coordinates, the NameCert obligations cannot use measure completion, selected almost-everywhere representatives, quotient function equality, countable choice, or ambient Real$^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted LebesgueDifferentiationTheorem$^{\uparrow}$ read must expose the LebesgueDifferentiation row, LebesgueDifferentiationBasis row, HardyLittlewoodMaximal row, FiniteVitaliCover ledger, Measure source row, Real seal row, theorem-readback ledger, exclusion ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert before a theorem-level differentiation verdict is consumed.

Independence witness: lebesgue-differentiation, lebesgue-differentiation-basis, hardy-littlewood-maximal, finite-vitali-cover, measure, real: LebesgueDifferentiationTheorem$^{\uparrow}$ is not bijective to any listed sibling because it binds the theorem-level readback ledger and exclusion ledger to all six local rows before the verdict is named.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiationTheorem}^{\uparrow}}$**.**

Constructive story: A LebesgueDifferentiationTheorem$^{\uparrow}$ packet is generated from LebesgueDifferentiationUp rows, LebesgueDifferentiationBasisUp rows, HardyLittlewoodMaximalUp domination, FiniteVitaliCoverUp ledgers, MeasureUp source rows, RealUp seals, theorem-readback rows, exclusion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LebesgueDifferentiationTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 8476.1 and Definition 8476.2 seed the theorem-level finite route through differentiation, basis, maximal, Vitali, measure, and real-name rows.

Formal verification: FormalStatus(NameCert$_{\text{LebesgueDifferentiationTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim measure completion, selected almost-everywhere representatives, quotient function equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, differentiation-row admission, basis-row admission, maximal-domination routing, finite Vitali-cover routing, Measure and Real sealing, exclusion-ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueDifferentiationTheoremUp.

Chapter 8477

A Concrete Naming Certificate for $\mathsf{FreudenthalSuspension}^{\uparrow}$

Following Freudenthal and Hatcher on suspension and stable homotopy, the $\mathsf{FreudenthalSuspension}^{\uparrow}$ packet records the finite route by which a pointed homotopy object is suspended and handed to a stable-range connectivity consumer. It sits after the $\mathsf{Homotopy}^{\uparrow}$ deformation surface of chapter 156, the $\mathsf{SimplicialSet}^{\uparrow}$ face-and-degeneracy ledger of chapter 157, the $\mathsf{Homology}^{\uparrow}$ chain-read surface of chapter 98, and the S1Up circle certificate of chapter 58. The chapter names suspension cones, equator gluing, basepoint transport, homology-window, connectivity-handoff, replay, provenance, and local naming rows; it does not import a host suspension space, quotient topology, selected homotopy representatives, stable homotopy groups, or Freudenthal's theorem as an ambient oracle.

Definition 8477.1 (FreudenthalSuspension carrier)**.** *A* $\mathsf{FreudenthalSuspension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, S_-, S_+, E, G, H_*, K, T, R, P, N)$$

with the following displayed rows:

1. *X is the pointed source row, read through* $\mathsf{Homotopy}^{\uparrow}$ *and carrying only the displayed basepoint and deformation ledgers needed by the suspension read.*
2. *S_- and S_+ are the two cone rows. Each cone row records finite face, degeneracy, and endpoint data through* $\mathsf{SimplicialSet}^{\uparrow}$ *rather than through a quotient topological suspension.*
3. *E is the equator row, read through the S1Up circle certificate and the source basepoint ledger.*
4. *G is the gluing ledger that identifies the two cone boundaries along E by displayed* hsame *data.*
5. *H_* is the finite* $\mathsf{Homology}^{\uparrow}$ *read window used by the stable-range consumer.*
6. *K is the connectivity handoff row. It records only the displayed stable-range obligation surface, not a selected homotopy group element or a host Freudenthal isomorphism.*
7. *T records componentwise* hsame *transport for the source, cones, equator, gluing, homology window, connectivity handoff, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a suspension query through $X, S_-, S_+, E, G, H_*, K, T$.*
9. *P is the* Pkg *provenance over* $\mathsf{FreudenthalSuspension}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, *S1Up,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FreudenthalSuspension}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient suspension, a selected homotopy class, an ambient stable homotopy group, a classical connectivity theorem, open-cover compactness, or choice-selected cellular approximation.

Theorem 8477.2 (FreudenthalSuspension NameCert obligations)**.** *For every accepted* $\mathsf{FreudenthalSuspension}^{\uparrow}$ *carrier* $F = (X, S_-, S_+, E, G, H_*, K, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FreudenthalSuspension}^{\uparrow}}$ *surface consists exactly of source admission, two displayed cone admissions, equator admission through S1Up, cone-boundary gluing, finite homology-window readback, stable-range connectivity handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream suspension consumer must read* X, S_-, S_+, E, G *before the homology window* H_* *is public and must read* H_* *before the connectivity handoff* K *is public.*

Proof. Project the carrier of Definition 8477.1. The source row X supplies only displayed homotopy data, while S_- and S_+ supply the two finite simplicial cone ledgers.

The equator row E is read through S1Up before the gluing ledger G is exposed. Thus the suspension boundary is a displayed cone-equator-cone route rather than a quotient-space construction.

The homology window H_* can be inspected only after the source, cone, equator, and gluing rows have been replayed. The connectivity handoff K is then a finite stable-range obligation row over that homology window, not an ambient stable homotopy theorem.

Transport through T, replay through R, provenance through P, and local naming through N preserve exactly these rows. Since the carrier stores no quotient suspension, selected homotopy class, host homotopy group, or classical connectivity theorem, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8477.3 (FreudenthalSuspension connectivity handoff)**.** *Every accepted* $\mathsf{FreudenthalSuspension}^{\uparrow}$ *connectivity consumer reaches its stable-range handoff only after the displayed* $\mathsf{Homotopy}^{\uparrow}$ *source row, the two* $\mathsf{SimplicialSet}^{\uparrow}$ *cone rows, the S1Up equator row, the cone-boundary gluing ledger, and the finite* $\mathsf{Homology}^{\uparrow}$ *read window have been exposed. Consequently the handoff exports a local NameCert obligation surface for suspension connectivity, not a proof of Freudenthal's suspension theorem, a selected homotopy class, a quotient suspension space, or a completed stable homotopy group.*

Proof. Start from Definition 8477.2. It places $X, S_-, S_+, E, G, H_*, K, T, R, P, N$ in the only accepted order available to the packet.

The homotopy, simplicial, circle, and homology siblings are therefore visible before K can be consumed. Each sibling contributes a displayed row: deformation data from $\mathsf{Homotopy}^{\uparrow}$, face and degeneracy data from $\mathsf{SimplicialSet}^{\uparrow}$, equator data from S1Up, and the finite chain-read window from $\mathsf{Homology}^{\uparrow}$.

Finally the structural rows T, R, P, N preserve that route under transport, replay, provenance, and local naming. With no carrier coordinate for quotient suspension spaces, selected homotopy classes, stable homotopy groups, or classical connectivity oracles, the connectivity handoff remains a finite BEDC obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreudenthalSuspension}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FreudenthalSuspension}^{\uparrow}$ packet is generated from a pointed HomotopyUp source row, two SimplicialSetUp cone rows, an S1Up equator row, a cone-boundary gluing ledger, a HomologyUp read window, a stable-range connectivity handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreudenthalSuspension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8477.1, Definition 8477.2, and Definition 8477.3 seed the Freudenthal suspension packet over Homotopy, SimplicialSet, Homology, S1, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FreudenthalSuspension}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Freudenthal's suspension theorem, quotient suspension topology, selected homotopy classes, stable homotopy groups, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, cone admission, equator gluing, homology-window readback, connectivity handoff, transport, replay, provenance, and local naming for BEDC.Derived.FreudenthalSuspensionUp.

Chapter 8478

A Concrete Naming Certificate for $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$ packet records a located metric subspace together with a finite oscillation gauge before any Real or Completion consumer reads the resulting function space. It sits over the metric source of chapter 57, the located-subset surface of chapter 1541, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the explicit modulus-continuity row of chapter 1631. The packet exposes oscillation control as finite located metric data, not as an external selector, quotient function equality, countable choice, host continuity, or ambient completeness.

Definition 8478.1 (Bishop located oscillation space carrier)**.** *A* $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, L, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the finite metric requests and comparison budget visible to the packet.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row for the inspected subspace, including the positive and negative locatedness tests allowed over the source row M.*
3. *G is the located oscillation gauge row. It records the displayed input cells, local radius requests, dyadic slack, and oscillation bounds served on the same support as M and L.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the sampled values and oscillation bounds carried by G.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the metric source, located-set row, oscillation gauge, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the metric source, located subset, oscillation gauge, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each oscillation request through M, L, G, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, G, R, E, H, C, P, N$. It has no coordinate for selected representatives, quotient function equality, a detached continuity principle, countable choice, global compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8478.2 (Bishop located oscillation space NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$ *carrier*

$$O = (M, L, G, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, metric-source exposure through M, located-subset admission through L, oscillation-gauge exactness through G, regular rational readback through R, Real-seal nonescape through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 8478.1. The packet displays exactly the metric source, located subset, oscillation gauge, regular readback, Real seal, transport, replay, provenance, and local name.

The source and located rows are read first. Row M fixes the metric comparison surface, and row L restricts every oscillation request to the located subset carried on that surface.

The oscillation obligation is the gauge row G. Its local radii and oscillation bounds are valid only on the same located support, and the readback row R exposes those bounds as regular rational data before the Real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no row for selected representatives, quotient equality, detached continuity, countable choice, global compactness, or ambient Real completeness, none of those principles is exported by the local NameCert surface. □

Theorem 8478.3 (Bishop located oscillation space modulus route)**.** *Every* $\mathsf{ModulusContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$ *carrier reads the finite route*

$$M, L, G \longmapsto R, E \longmapsto H, C, P, N.$$

The consumer receives a source metric row, a located subset row, a displayed oscillation gauge, regular rational readback, and a Real seal before any modulus-continuity handoff is visible. The route exports no external selector, host continuity theorem, quotient function equality, selected representative, countable-choice schedule, global compactness, or ambient completeness coordinate.

Proof. Project the carrier through Definition 8478.1. The only analytic rows available to a modulus consumer are M, L, G, R, E.

Use Definition 8478.2. It forces the metric source and located subset to be displayed before the oscillation gauge is read, and it forces the regular rational row and Real seal to be displayed before the handoff reaches a $\mathsf{ModulusContinuity}^{\uparrow}$ consumer.

The structural rows H, C, P, N preserve that same finite route. Therefore the modulus-facing read is a located oscillation handoff over displayed metric data, not a host continuity principle or a completion-side selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedOscillationSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedOscillationSpace}^{\uparrow}$ packet is generated from a MetricUp source row, a LocatedSetUp row, a located oscillation gauge, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopLocatedOscillationSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8478.1, Definition 8478.2, and Definition 8478.3 seed the located metric oscillation route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedOscillationSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected representatives, quotient function equality, detached continuity, countable choice, global compactness, ambient Real$^{\uparrow}$ completeness, Lean theorem checking, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, located-subset admission, oscillation-gauge exactness, regular rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedOscillationSpaceUp.

Chapter 8479

A Concrete Naming Certificate for $\mathsf{RationalIntervalCoverage}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalIntervalCoverage}^{\uparrow}$ packet records coverage of a displayed rational interval by finitely many dyadic subintervals. It sits over the rational endpoint surface of chapter 37, the dyadic arithmetic core of chapter 291, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the Real and Completion route a named finite coverage carrier for interval arguments, not an open-cover compactness theorem, a quotient interval, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8479.1 (Rational interval coverage carrier)**.** *A* $\mathsf{RationalIntervalCoverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Rat}^{\uparrow}$ *endpoint row for the source closed interval, displaying rational endpoints $a \leq b$.*
2. *B is the finite list of rational subinterval rows whose endpoints lie in the source interval and whose successive endpoints meet at displayed rational boundaries.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger. It records dyadic radii, midpoint cuts, containment checks, and endpoint slack bounds for every subinterval named by B.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation schedule bounding which endpoint, midpoint, and containment rows may be inspected by a downstream consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the displayed rational and dyadic endpoint data toward regular rational names.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the rational endpoint, subinterval, dyadic ledger, stream-window, regular-rational, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interval-coverage request through $A, B, D, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalCoverage}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalCoverage}^{\uparrow}}$ *row naming exactly this finite interval-coverage packet.*

The classifier compares accepted packets only through $A, B, D, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient intervals, selected real representatives, countable choice, host equality of real endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8479.2 (Rational interval coverage NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalCoverage}^{\uparrow}$ *carrier $I = (A, B, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalCoverage}^{\uparrow}}$ *surface consists exactly of carrier admission for the source rational interval, finite rational subinterval list, dyadic refinement ledger, bounded StreamName observation schedule, RegSeqRat readback, Real seal handoff, componentwise transport, displayed replay, provenance, and local naming. The obligation surface may cite* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling chapters, but it cannot replace the displayed finite rows by open-cover compactness, quotient interval equality, host real equality, countable choice, selected representatives, or ambient real completeness.*

Proof. Project the carrier of Definition 8479.1. The analytic rows visible to the local certificate are exactly the rational endpoint row A, the finite subinterval list B, the dyadic refinement ledger D, the finite observation schedule W, the regular-rational readback R, and the downstream real seal E.

The rational interval is covered only by reading the finite list B. For each listed subinterval, the dyadic ledger D supplies the displayed midpoint, containment, overlap, and slack checks needed to compare its endpoints with the source endpoints in A.

The rows W and R bound the consumer route. A Real or Completion-facing use must inspect only the scheduled finite endpoint and dyadic rows before the regular rational readback is available.

Finally H, C, P, N transport, replay, source, and name the same packet. Since none of the excluded principles is a carrier coordinate or a product of these projections, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8479.3 (Rational interval coverage dyadic refinement)**.** *Let $I = (A, B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{RationalIntervalCoverage}^{\uparrow}$ *carrier. Every subinterval row in B refines the source rational interval by the finite route*

$$A \longmapsto B \longmapsto D \longmapsto W \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, each endpoint comparison used by a downstream Real or Completion consumer is witnessed by a displayed dyadic budget in D before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is read. The route exports no open-cover compactness, quotient interval, selected real representative, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Fix a listed subinterval in B. By Definition 8479.1, its endpoints are rational rows tied to the source endpoint row A and to the dyadic ledger D.

Read the dyadic ledger next. The containment and slack entries in D give a finite comparison budget for the left endpoint, the right endpoint, and the shared boundary rows visible in B.

The StreamName row W bounds which of these comparisons can be inspected. The RegSeqRat row R then packages exactly the displayed endpoint approximations, and the Real seal E remains downstream of that same finite route.

The structural rows H, C, P, N add no analytic principle beyond transport, replay, provenance, and naming. Thus every refinement claim factors through the displayed dyadic budget and cannot escape to open-cover compactness, quotient intervals, selected representatives, choice, host equality, or ambient completeness. □

Theorem 8479.4 (Rational interval coverage Real-completion nonescape)**.** *Any Real or Completion-facing use of an accepted* $\mathsf{RationalIntervalCoverage}^{\uparrow}$ *packet must expose the source rational interval, finite subinterval list, dyadic refinement ledger, bounded observation schedule, and regular-rational readback before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Consequently the packet is available to compact-interval and completion arguments only as finite dyadic interval-coverage evidence, not as open-cover compactness, quotient interval equality, host real equality, countable choice, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real or Completion-facing consumer of I. The carrier projection first exposes A, B, D, W, R; no real seal can be read before those rows have been displayed.

Apply Definition 8479.3. Each subinterval comparison reaching the consumer has already passed through the finite dyadic ledger and the bounded StreamName schedule.

The terminal row E is therefore a seal of the displayed regular rational readback, not an independent host real interval. Transport, replay, provenance, and local naming keep that route inside H, C, P, N.

Since the packet contains no open-cover object, quotient interval coordinate, selected representative, choice schedule, host equality proof, or ambient completeness row, none of those objects can be exported by a Real or Completion-facing use. □

Falsifiable prediction: An accepted $\mathsf{RationalIntervalCoverage}^{\uparrow}$ consumer that reaches a RealUp or completion-facing interval conclusion without reading rational endpoint rows, a finite subinterval list, a DyadicRatCore refinement ledger, StreamName windows, and a RegSeqRat readback refutes the carrier surface.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalIntervalCoverage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source rational interval, finite subinterval list, dyadic refinement ledger, bounded observation schedule, regular-rational readback, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalCoverage}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalIntervalCoverage}^{\uparrow}$ packet is generated from source rational endpoints, a finite rational subinterval list, dyadic midpoint and slack rows, bounded StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalCoverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8479.1, Definition 8479.2, Definition 8479.3, and Definition 8479.4 seed the finite dyadic interval-coverage route over $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalCoverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim open-cover compactness, quotient interval equality, host real equality, countable choice, selected real representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite rational subinterval exposure, dyadic refinement exactness, bounded StreamName observation, RegSeqRat readback, Real seal handoff, and nonescape under BEDC.Derived.RationalIntervalCoverageUp.

Chapter 8480

A Concrete Naming Certificate for $\mathsf{DividedPowerAlgebra}^{\uparrow}$

Following Berthelot–Ogus and Grothendieck crystalline methods, the $\mathsf{DividedPowerAlgebra}^{\uparrow}$ packet records the finite law surface by which divided powers are named over a ring-module algebra context before any crystalline envelope or infinitesimal thickening consumer may read them. It sits over the coefficient ring of chapter 43, the underlying module rows of chapter 46, polynomial expression rows from chapter 50, tensor compatibility from chapter 87, and natural-number arity data from chapter 28. The packet names displayed multiplication, unit, ideal, divided-power operation, factorial-arity, polynomial-law, tensor-distribution, transport, replay, provenance, and local naming data; it does not export a host crystalline topos, quotient algebra equality, selected envelope, or unrestricted nilpotent completion.

Definition 8480.1 (Divided-power algebra carrier)**.** *A* $\mathsf{DividedPowerAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, A, I, \gamma, M, U, N, L, T, H, C, P, Z)$$

with the following displayed rows:

1. *R is the* $\mathsf{Ring}^{\uparrow}$ *coefficient row, carrying the addition, multiplication, zero, one, and distributivity data used by the algebra law checks.*
2. *A is the algebra-underlying* $\mathsf{Module}^{\uparrow}$ *row, exposing the finite additive and scalar rows on which the divided-power operations act.*
3. *I is the displayed ideal or augmentation row inside A, recorded as a finite membership and closure ledger.*
4. *γ is the finite divided-power operation table, with entries $\gamma_n(x)$ only for the natural-number arities and ideal elements listed by the request rows.*
5. *M and U record multiplication and unit compatibility for the algebra row over R.*
6. *N is the* $\mathsf{Nat}^{\uparrow}$ *arity ledger for zero, one, successor, and finite binomial-index reads.*
7. *L is the* $\mathsf{Poly}^{\uparrow}$ *law ledger, recording the displayed additivity, scalar, product, and composition law checks as polynomial identities over the listed arities.*
8. *T is the* $\mathsf{TensorProduct}^{\uparrow}$ *compatibility row for bilinear distribution and base-change reads.*
9. *H, C, P, Z record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DividedPowerAlgebra}^{\uparrow}}$ *evidence.*

The classifier compares accepted packets only through $R, A, I, \gamma, M, U, N, L, T, H, C, P, Z$. It has no coordinate for a host crystalline envelope, quotient algebra equality, selected completion, external factorial inverse, or ambient nilpotent thickening.

Theorem 8480.2 (Divided-power algebra NameCert obligations)**.** *For every accepted* $\mathsf{DividedPowerAlgebra}^{\uparrow}$ *carrier*

$$D = (R, A, I, \gamma, M, U, N, L, T, H, C, P, Z),$$

the local $\mathsf{NameCert}_{\mathsf{DividedPowerAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission for the coefficient ring, module-algebra row, ideal membership ledger, divided-power operation table, multiplication and unit rows, natural-number arity ledger, polynomial law ledger, tensor compatibility row, componentwise transport, displayed replay, provenance, and local naming. The law obligations require $\gamma_0(x)$, $\gamma_1(x)$, scalar compatibility, additivity, product compatibility, and iterated divided powers to be checked only through the displayed $R, A, I, \gamma, M, U, N, L, T$ rows.*

Proof. Project the carrier of Definition 8480.1. The algebraic source rows are $R, A, I, \gamma, M, U, N, L, T$, and the structural rows are H, C, P, Z.

The coefficient and module rows type every displayed operation. The ideal row I supplies the admitted inputs, while the arity ledger N restricts every divided-power read to the natural-number requests carried by the packet.

The law checks are then read from L and T. Polynomial rows record the displayed zero, unit, scalar, additivity, product, and composition equations, and the tensor row records the bilinear and base-change compatibility seen by downstream consumers.

Finally, H, C, P, Z transport, replay, source, and locally name exactly those rows. Since no carrier coordinate stores a crystalline envelope, quotient algebra equality, selected completion, external factorial inverse, or ambient nilpotent thickening, none of those objects can discharge a local NameCert obligation. □

Theorem 8480.3 (Divided-power law nonescape)**.** *Every consumer of an accepted* $\mathsf{DividedPowerAlgebra}^{\uparrow}$ *divided-power law must read the law through the finite route*

$$\mathsf{Ring}^{\uparrow},\ \mathsf{Module}^{\uparrow},\ I,\ \gamma,\ \mathsf{Nat}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow},\ \mathsf{TensorProduct}^{\uparrow} \longmapsto H, C, P, Z.$$

In particular, additivity, scalar, product, and iteration laws cannot be justified by a host polynomial identity, a chosen divided-power envelope, quotient algebra equality, or an unlisted nilpotent-completion coordinate.

Proof. Start from Definition 8480.2. It forces every law-facing consumer to expose the coefficient ring, module-algebra row, ideal ledger, divided-power operation table, and natural-number arity ledger before a law row can be consumed.

The polynomial and tensor rows are the only law ledgers in the carrier. Thus a consumer may cite additivity, scalar compatibility, product compatibility, or iterated divided powers only after reading the displayed $\mathsf{Poly}^{\uparrow}$ and $\mathsf{TensorProduct}^{\uparrow}$ rows.

The structural rows preserve the same finite route. Transport, replay, provenance, and local naming do not add a crystalline envelope, quotient equality, selected completion, or hidden nilpotent coordinate, so the divided-power laws remain inside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{DividedPowerAlgebra}^{\uparrow}$ consumer that cites a divided-power law must expose the Ring, Module, ideal, divided-power table, multiplication, unit, Nat arity, Polynomial law, TensorProduct compatibility, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: ring, module, polynomial, tensorproduct, nat: no listed sibling is bijective to $\mathsf{DividedPowerAlgebra}^{\uparrow}$, because this packet binds a coefficient ring, module-algebra row, ideal ledger, arity-indexed divided-power operations, polynomial laws, and tensor compatibility into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DividedPowerAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{DividedPowerAlgebra}^{\uparrow}$ packet is generated from RingUp coefficient rows, ModuleUp algebra rows, an ideal membership ledger, an arity-indexed divided-power operation table, multiplication and unit rows, a NatUp arity ledger, PolynomialUp law rows, TensorProductUp compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DividedPowerAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8480.1, Definition 8480.2, and Definition 8480.3 seed the finite divided-power law surface over ring, module, polynomial, tensor, and natural-number rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DividedPowerAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host crystalline envelope, quotient algebra equality, selected completion, external factorial inverses, unrestricted nilpotent completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ideal closure, divided-power table exactness, Nat arity coverage, polynomial law exactness, tensor compatibility, transport, replay, provenance, and local naming for BEDC.Derived.DividedPowerAlgebraUp.

Chapter 8481

A Concrete Naming Certificate for $\mathsf{SignedDistanceInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis of located intervals, the $\mathsf{SignedDistanceInterval}^{\uparrow}$ packet records the finite boundary-distance route by which a displayed point is compared with a displayed located interval before any Real-facing consumer reads a signed value. It sits over the bisection route of chapter 1711, the located-apartness comparison surface of chapter 6450, the metric route of chapter 1638, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names the interval boundary-distance carrier as finite BEDC data; it does not import host real equality, selected nearest endpoints, trichotomy, or ambient completeness.

Definition 8481.1 (Signed distance interval carrier)**.** *A* $\mathsf{SignedDistanceInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, A, B, O, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the point row, recording the inspected* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *name whose signed distance to the interval is requested.*
2. *A and B are the interval endpoint rows, recording the left and right located endpoint reads supplied by the located interval source.*
3. *O is the endpoint-order ledger, recording the finite route that certifies the displayed left endpoint is accepted before the displayed right endpoint.*
4. *S is the inside/outside sign row, recording whether the point read is routed through the interior window, the left exterior window, or the right exterior window.*
5. *D is the dyadic distance readback, carrying the finite tolerance, endpoint window, and signed rational comparison requested by the consumer.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to X, A, B, O, S, D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback row R has been displayed.*
8. *H records componentwise* hsame *transport for the point row, endpoint rows, endpoint-order ledger, sign row, dyadic distance row, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of every signed-distance-interval consumer route through*

$$X, A, B, O, S, D, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{SignedDistanceInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDistanceInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, B, O, S, D, R, E, H, C, P, N$. It has no coordinate for host real equality, a selected nearest endpoint, trichotomy, quotient representatives, ambient interval compactness, Lean verification, or bridge checking.

Theorem 8481.2 (Signed distance interval NameCert obligations)**.** *For every accepted* $\mathsf{SignedDistanceInterval}^{\uparrow}$ *carrier*

$$I = (X, A, B, O, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDistanceInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed point row, interval endpoint rows, endpoint-order ledger, inside/outside sign row, dyadic distance readback, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *endpoint-order stability, requiring every transported or replayed read to expose A, B, O before it treats the interval as an ordered located boundary;*
3. *sign-window exactness, requiring the row S to be read from the same point and endpoint rows X, A, B and to choose only the displayed interior, left exterior, or right exterior route;*
4. *dyadic and regular readback coherence, requiring D and R to read the same tolerance and signed boundary-distance observations before the Real seal E is exposed;*
5. *ledger refusal, requiring H, C, P, N to transport, replay, source, and name only the displayed rows, with no host equality, selected endpoint, trichotomy, Lean verification, or bridge credential.*

No signed-distance-interval NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8481.1. The packet displays the point row, two endpoint rows, the endpoint-order ledger, the sign row, the dyadic distance readback, the regular rational readback, the Real seal, and the four structural rows.

Endpoint-order stability is componentwise. Any transported or replayed interval read keeps the same left endpoint row A, right endpoint row B, and order ledger O, so the boundary cannot be replaced by a host interval or by an unrecorded endpoint comparison.

The sign-window obligation is read from X, A, B. The row S names only the displayed interior route, left exterior route, or right exterior route, and therefore cannot install a trichotomy decision beyond the finite packet.

The dyadic row D and regular row R share the tolerance and signed boundary-distance observations. Since E appears only after these rows are displayed, every Real-facing value is sealed by the same finite readback route.

Finally, H, C, P, N only transport, replay, source, and name the carrier rows. The excluded host equality, nearest-endpoint selector, trichotomy, Lean-verification, and bridge data have no carrier coordinate, so the listed rows exhaust the local certificate surface. □

Theorem 8481.3 (Signed distance interval located handoff)**.** *Every accepted* $\mathsf{SignedDistanceInterval}^{\uparrow}$ *carrier*

$$I = (X, A, B, O, S, D, R, E, H, C, P, N)$$

hands its boundary-distance read to located interval consumers only through the displayed route

$$X, A, B, O, S, D, R, E.$$

The handoff consumes the located bisection endpoint rows, the located-apartness sign window, the Real metric comparison route, and the regular rational readback before any signed value is public. It exports no host interval equality, nearest endpoint, decidable side choice, quotient real representative, or ambient completeness theorem.

Proof. Project the accepted carrier. The endpoint rows A and B are the only interval-boundary coordinates, and the order ledger O is the only row that lets a consumer treat them as an ordered located interval.

The point row X is then compared with the displayed boundary through the sign row S. This row is downstream of the located-apartness comparison surface and records only the finite interior or exterior window selected by the packet.

The dyadic readback D and regular readback R pass the same signed boundary-distance observations to the Real seal E. Thus a bisection, located-apartness, or Real-metric consumer receives the displayed route rather than a host-side interval object.

Transport, replay, provenance, and local naming preserve this route under classification. Since none of the excluded host data occurs in I, the handoff cannot bypass the displayed rows. □

Theorem 8481.4 (Signed distance interval Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{SignedDistanceInterval}^{\uparrow}$ *carrier*

$$I = (X, A, B, O, S, D, R, E, H, C, P, N)$$

factors through the displayed route

$$X, A, B, O, S, D, R \longmapsto E.$$

The factorization exports the signed boundary-distance seal E only after the point row, endpoint rows, endpoint-order ledger, sign row, dyadic distance readback, and regular rational readback have been displayed. It exports no host real equality, selected nearest endpoint, trichotomy, quotient representative, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. The carrier projection exposes E as the only Real-facing coordinate. Its replay row C names a route whose source rows are X, A, B, O, S, D, R.

Rows A, B, O bind the interval boundary, while X supplies the inspected point. The sign row S keeps the inside/outside decision inside the displayed finite window rather than in a host trichotomy.

The dyadic row D and regular row R then supply the only numeric readback consumed by the seal. Componentwise transport in H, replay in C, provenance in P, and naming in N preserve this order.

Therefore every Real-facing read factors through the displayed finite route. The excluded host equality, nearest-endpoint selection, quotient representative, completeness theorem, Lean-verification row, and bridge credential cannot escape through the seal because they are not coordinates of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDistanceInterval}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDistanceInterval}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8481.1, Definition 8481.2, Definition 8481.3, and Definition 8481.4 seed the finite signed boundary-distance route from located interval rows through dyadic and regular readback to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDistanceInterval}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, selected nearest endpoints, trichotomy, quotient representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order stability, sign-window exactness, dyadic and regular readback coherence, Real-seal nonescape, and ledger refusal for BEDC.Derived.SignedDistanceIntervalUp.

Constructive story: A $\mathsf{SignedDistanceInterval}^{\uparrow}$ packet is generated from point row, interval endpoint rows, endpoint-order ledger, inside/outside sign row, dyadic distance readback, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8482

A Concrete Naming Certificate for $\mathsf{BishopDiagonalRegularization}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{BishopDiagonalRegularization}^{\uparrow}$ packet records the finite diagonal regularization step used by Real and Completion consumers before they read a regular rational name. It sits beside the dyadic tolerance core of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite completion operator route of chapter 3379. The packet isolates diagonal repair as displayed NameCert data; it does not assert quotient stream equality, selected host limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8482.1 (Bishop diagonal regularization carrier)**.** *A* $\mathsf{BishopDiagonalRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus-indexed diagonal choice row, recording which finite diagonal index is inspected for each precision request.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite observations used by the diagonal repair.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic bounds that compare the diagonal entry with the repaired regular name.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance rows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the repaired regular readback has been displayed.*
6. *H records componentwise* hsame *transport for the modulus choice, window, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through M, W, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopDiagonalRegularization}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalRegularization}^{\uparrow}}$ *row.*

The classifier accepts only packets exhausted by $M, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, host real equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopDiagonalRegularizationUp.BishopDiagonalRegularizationTasteGate`
`single carrier alignment`

Theorem 8482.2 (Bishop diagonal regularization NameCert obligations)**.** *For every accepted* $\mathsf{BishopDiagonalRegularization}^{\uparrow}$ *carrier* $G = (M, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; diagonal-window exactness through* M, W*; regularity repair through* D, R*;* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*; tolerance compatibility through* D*; Real-seal nonescape through* E*; componentwise transport through* H*; displayed replay through* C*; package provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 8482.1. The accepted packet exposes precisely the modulus-indexed diagonal choice, finite stream windows, dyadic tolerances, regular rational readback, Real seal, and structural rows.

Read the analytic rows first. The diagonal-window exactness is confined to M, W, while regularity repair and tolerance compatibility are confined to D, R. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is downstream of the displayed diagonal window and dyadic tolerance rows.

Finally inspect E, H, C, P, N. The Real seal is reached only after the repaired readback has been displayed, and the structural rows only transport, replay, source, and name the same packet. Since the excluded quotient, selected-limit, choice, host-equality, and completeness data are not coordinates of G, they cannot appear as local obligations. □

Theorem 8482.3 (Bishop diagonal regularization RegSeqRat handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{BishopDiagonalRegularization}^{\uparrow}$ *carrier must pass through the diagonal repair route*

$$M, W \longmapsto D \longmapsto R \longmapsto E, H, C, P, N$$

before reading either the $\mathsf{RegSeqRat}^{\uparrow}$ *row or the* $\mathsf{Real}^{\uparrow}$ *seal. The consumer receives a displayed repaired regular name, not a quotient stream, selected host limit, countable-choice schedule, or ambient completeness principle.*

Proof. Begin with the obligation surface of Definition 8482.2. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R after the modulus choice M, StreamName windows W, and dyadic tolerance ledger D.

The Real-completion consumer therefore reaches the regular readback only through the displayed diagonal repair route. The seal row E is later still, and H, C, P, N only transport, replay, source, and name that same finite route.

The carrier inventory has no coordinate for quotient stream equality, selected host limits, host natural-number search, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence a consumer that reads R or E must read the displayed diagonal repair packet rather than an external completion witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalRegularization}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{BishopDiagonalRegularization}^{\uparrow}$ packet is generated from a modulus-indexed diagonal choice row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopDiagonalRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 8482.1, Definition 8482.2, and Definition 8482.3 seed the finite diagonal regularization route over StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopDiagonalRegularization}^{\uparrow}}$, encodedDef)
Lean4 (statement) `BEDC.Derived.BishopDiagonalRegularizationUp`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected host limits, countable choice, host real equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-window exactness, regularity repair, RegSeqRat handoff, tolerance compatibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopDiagonalRegularizationUp.

Chapter 8483

A Concrete Naming Certificate for $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$

The $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ packet realizes the compact-uniform-continuity route as one finite split ledger: compact finite windows and pointwise continuous-map rows are folded into a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff before any $\mathsf{Real}^{\uparrow}$ consumer reads the target estimate. It sits over the compact metric source of chapter 129, the graph and pointwise-modulus surface of chapter 130, the tolerance-only handoff of chapter 369, the dyadic precision rows of chapter 291, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The carrier records the finite $\mathsf{ProbeBundle}$ split from pointwise compact-cover moduli to one uniform-modulus read; it is not a host Heine–Cantor theorem, a selected Lebesgue number, an infinite cover, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8483.1 (Compact-uniform splitting ledger carrier)**.** *A* $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, U, B, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, listing the finite compact windows and cover probes inspected by the ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise row, recording the local modulus entries attached to the compact windows in K.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance-only output row.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *split ledger, pairing each compact probe with its local radius, target tolerance request, and selected finite comparison row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision row for the dyadic tolerance cuts used by the split.*
6. *Q is the rational lower-bound and triangle-budget row that chooses the displayed minimum budget over the finite probe bundle.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational and dyadic estimates consumed by downstream Real-facing readers.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the source and target metric estimates.*

9. *H records componentwise* hsame *transport for compact, continuous-map, uniform-modulus, probe-bundle, dyadic, rational-budget, regular-readback, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the finite split*

$$K, F, B, D, Q \longmapsto U, R, E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformSplittingLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, B, D, Q, R, E, H, C, P, N$. It has no coordinate for an infinite cover, selected Lebesgue number, host compactness theorem, countable choice, external modulus oracle, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8483.2 (Compact-uniform splitting ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ *carrier*

$$S = (K, F, U, B, D, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformSplittingLedger}^{\uparrow}}$ *surface consists exactly of compact-source exposure through K, continuous-map pointwise exposure through F, finite* ProbeBundle *split exposure through B, dyadic precision exposure through D, rational lower-bound and metric-triangle budget exposure through Q, tolerance-only uniform-modulus output through U, regular-rational readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8483.1. The compact-continuous source rows are K, F, the finite splitting rows are B, D, Q, the output rows are U, R, E, and the structural rows are H, C, P, N.

The compact source and continuous-map rows determine the finite cover probes and local pointwise moduli. The ProbeBundle row then records the split from those local rows to a single tolerance request.

The dyadic and rational-budget rows record the precision and lower-bound comparisons used in the metric-triangle estimate. The regular readback and Real seal are downstream of those rows, while the uniform-modulus row is the only tolerance-only output.

Finally, transport, replay, provenance, and local naming preserve that displayed route. Since the carrier has no infinite cover, selected Lebesgue number, host compactness theorem, countable-choice row, external modulus oracle, quotient real equality, or ambient Real-completeness coordinate, those objects cannot discharge a local NameCert obligation. □

Theorem 8483.3 (Compact-uniform split nonescape)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto Q,\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto$$

Thus the tolerance-only uniform-modulus handoff and the Real-facing metric estimate cannot be justified by an infinite cover, a selected Lebesgue number, countable choice, quotient real equality, or ambient completeness.

Proof. Start with Definition 8483.2. It forces the compact metric source, continuous-map pointwise row, finite ProbeBundle split, dyadic precision row, and rational budget row to be displayed before a uniform-modulus or Real-facing read is available.

The uniform-modulus output is the row U. It is tolerance-only because the source point, local modulus, lower-bound fold, and triangle estimate have already been folded through B, D, Q.

The Real-facing read then passes through R and E. These rows read regular rational and Real seals downstream from the same split ledger, so they cannot introduce a selected cover, quotient real equality, or completeness theorem.

The structural rows keep the route finite and replayable. Hence every downstream consumer remains inside the displayed compact-uniform splitting ledger. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ read is refuted if it exports a $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ result without displaying the $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, finite ProbeBundle, $\mathsf{DyadicRatCore}^{\uparrow}$, rational lower-bound, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ is not identical to any listed sibling because it binds compact finite-window data, pointwise continuous-map moduli, a finite ProbeBundle split, dyadic precision, rational lower-bound readback, a tolerance-only UniformModulus handoff, and a Real seal as one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSplittingLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformSplittingLedger}^{\uparrow}$ packet is generated from CompactMetric finite-window rows, ContinuousMap pointwise-modulus rows, a UniformModulus output row, finite ProbeBundle split rows, DyadicRatCore precision rows, rational lower-bound and triangle-budget rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformSplittingLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8483.1, Definition 8483.2, and Definition 8483.3 seed the finite split from pointwise compact-cover moduli to one tolerance-only uniform-modulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformSplittingLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an infinite cover, selected Lebesgue number, host compactness theorem, countable choice, external modulus oracle, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, pointwise ContinuousMap exposure, ProbeBundle split exactness, dyadic precision, rational lower-bound and triangle-budget exactness, UniformModulus handoff, RegSeqRat readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSplittingLedgerUp.

Chapter 8484

A Concrete Naming Certificate for $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$ packet records the finite subsequence-extraction surface that is visible once a metric sequence exposes tolerance-indexed finite epsilon-net selector rows. It sits between the total-boundedness carrier of chapter 127, the finite center packet of chapter 1767, and the compact-metric consumer surface of chapter 129. Before any subsequence window is handed to $\mathsf{StreamName}^{\uparrow}$ (chapter 126), $\mathsf{RegSeqRat}^{\uparrow}$ (chapter 301), or the terminal $\mathsf{Real}^{\uparrow}$ seal (chapter 38), the packet makes the chosen finite-net rows, window bounds, replay rows, and nonescape clauses explicit as BEDC NameCert data.

Definition 8484.1 (Totally bounded subsequence carrier)**.** *A* $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, \sigma, E, P, C, D, W, H, R, S, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier, distance classifier, and metric-law ledger used by the sequence.*
2. *σ is the inspected metric sequence row, read as a finite-window* $\mathsf{StreamName}^{\uparrow}$ *schedule whose entries are admitted by X.*
3. *E is the positive tolerance row, read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *before a center selector can be used.*
4. *P is the closed* $\mathsf{ProbeBundle}$ *of finite-net centers supplied by the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *surface.*
5. *C records, for every accepted center of P, the corresponding point-carrier admission in X.*
6. *D records the finite-net coverage ledger: for each inspected sequence entry, one accepted center of P and one displayed* $\mathsf{MetricDistanceWitness}$ *row bounded by E.*
7. *W records the subsequence-window selector. It names a finite source window, a center branch of P, and the entries of σ that remain in that branch for the next extraction stage.*
8. *H records componentwise* hsame *transport for $X, \sigma, E, P, C, D, W, R, S, N$.*
9. *R records displayed* Cont *replay from a subsequence request through the same metric, tolerance, finite-net, coverage, and window rows.*

10. *S is the* Pkg *provenance over* $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for countable choice, an infinite pigeonhole functional, an open-cover compactness proof, a selected limit, a host subsequence object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8484.2 (Totally bounded subsequence finite-net selector)**.** *For every accepted* TotallyBoundedSubsequenc *carrier $Q = (X, \sigma, E, P, C, D, W, H, R, S, N)$, the finite-net selector row W can be read only after the metric source X, sequence row σ, positive tolerance E, center bundle P, center-carrier ledger C, and coverage-distance ledger D have been displayed. The selected branch remains a finite* ProbeBundle *branch with admitted centers and bounded metric-distance rows; it does not expose countable choice, an infinite pigeonhole functional, an open-cover compactness proof, a host subsequence object, or a selected* $\mathsf{Real}^{\uparrow}$ *limit.*

Proof. Project the carrier of Definition 8484.1. The selector coordinate W is placed after the metric, sequence, tolerance, finite center bundle, center admission, and coverage-distance rows.

The finite-net data are therefore prerequisites. A branch named by W can be inspected only through membership in P, admission in C, and the bounded distance rows recorded by D.

Transport and replay do not add a new selector source. The row H moves the same displayed coordinates componentwise, and R replays requests through the same ordered route. Since the carrier has no coordinate for countable choice, an infinite pigeonhole functional, open-cover compactness, host subsequences, or selected limits, none of those objects can be exported by the selector. □

Theorem 8484.3 (Totally bounded subsequence L10 handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$ *carrier must read the finite route*

$$X, \sigma, E \longmapsto P, C, D \longmapsto W, H, R, S, N$$

before it may treat a chosen subsequence window as real-name data. Thus the consumer receives only displayed metric rows, finite epsilon-net centers, bounded coverage witnesses, the subsequence-window selector, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The handoff exports no countable-choice schedule, host-level subsequence, quotient stream equality, selected limit, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with a downstream consumer of Q. By Definition 8484.1, entry into the packet first exposes the metric source X, the inspected sequence σ, and the positive tolerance row E.

The finite-net selector is opened next. The closed bundle P names the centers, C admits those centers as points of X, and D supplies the bounded distance rows that locate inspected sequence entries in center branches.

Apply Definition 8484.2. The window selector W can only name branches already justified by P, C, D, and the structural rows H, R, S, N preserve the same route when the window is replayed toward $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$.

Therefore the L10 consumer sees exactly the displayed finite route. Because the carrier omits countable-choice schedules, host-level subsequence objects, quotient stream equality, selected limits, open-cover compactness proofs, and ambient completeness data, the handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedSubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8484.1, Definition 8484.2, and Definition 8484.3 seed the finite selector packet that routes total boundedness and finite epsilon-net coverage into subsequence windows before StreamName, RegSeqRat, or Real consumers read the result.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, sequential compactness, infinite pigeonhole extraction, countable choice, open-cover compactness, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net selector visibility, subsequence-window transport, L10 handoff, and ledger nonescape for BEDC.Derived.TotallyBoundedSubsequenceUp.

Constructive story: A $\mathsf{TotallyBoundedSubsequence}^{\uparrow}$ packet is generated from a MetricUp source row, an inspected finite-window sequence, a positive RatUp or DyadicRatCoreUp tolerance, a closed ProbeBundle of centers, center-carrier and coverage-distance ledgers, a subsequence-window selector, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8485

A Concrete Naming Certificate for $\mathsf{CauchyApproximationCategory}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyApproximationCategory}^{\uparrow}$ packet records the category-facing side of constructive completion as finite BEDC naming data. It sits beside the Cauchy approximation system of chapter 1920, the Cauchy approximation space of chapter 7574, the Bishop metric completion packet of chapter 4023, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names objects and approximation-preserving arrows by displayed rows only; it does not introduce quotient completions, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a host category of all Cauchy objects.

Definition 8485.1 (Cauchy approximation category carrier)**.** *A* $\mathsf{CauchyApproximationCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, M, I, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the object row. Each object is a named finite Cauchy approximation packet whose visible coordinates are sourced from* $\mathsf{CauchyApproximationSystem}^{\uparrow}$ *or* $\mathsf{CauchyApproximationSpace}^{\uparrow}$ *rows.*
2. *M is the morphism approximation data row. A morphism carries displayed source and target object names, a finite window transformer, dyadic tolerance transport, and a regular-readback compatibility ledger.*
3. *I is the identity route. It records the reflexive window transformer, unchanged dyadic tolerance rows, unchanged* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and unchanged* $\mathsf{Real}^{\uparrow}$ *seal for each displayed object.*
4. *Q is the composition route. It records the finite pasting of two morphism approximation rows, including the composed window transformer, composed tolerance ledger, composed readback comparison, and Real-seal replay.*
5. *W is the finite-window row, reading every object and morphism request through* $\mathsf{StreamName}^{\uparrow}$ *windows before any regular readback or Real seal is opened.*
6. *R is the regular readback row, exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *observations used by objects, identities, and composites.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the finite windows, dyadic tolerances, regular readbacks, identity checks, and composition checks have been displayed.*

8. *H records componentwise* hsame *transport for $O, M, I, Q, W, R, E, C, P, N$ and for the displayed approximation comparisons.*
9. *C records displayed* Cont *replay for object admission, morphism approximation, identity replay, composition replay, finite-window access, regular readback, Real sealing, and transport.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationCategory}^{\uparrow}$, $\mathsf{CauchyApproximationSystem}^{\uparrow}$, $\mathsf{CauchyApproximationSpace}^{\uparrow}$, $\mathsf{BishopMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, M, I, Q, W, R, E, H, C, P, N$. It has no coordinate for quotient Cauchy equality, selected representatives, chosen limits, large host category structure, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8485.2 (Cauchy approximation category NameCert obligations)**.** *For every accepted* $\mathsf{CauchyApproximationCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyApproximationCategory}^{\uparrow}}$ *surface consists exactly of object admission through O; morphism approximation admission through M; identity-route exactness through I; composition-route exactness through Q; finite-window exposure through W; regular readback through R;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8485.1. The object and morphism coordinates are O and M, and the only category-structure coordinates are the displayed identity and composition rows I and Q.

The finite analytic access is forced through W, R, E. A reader first opens StreamName windows, then the RegSeqRat readback tied to those windows, and only then the Real seal. The dyadic tolerance data are the displayed comparison rows carried by M, I, Q and packaged by P.

Identity exactness is constructive because I is a reflexive replay row. It preserves the same object, the same finite windows, the same dyadic tolerance rows, the same regular readback, and the same Real seal, so it introduces no representative choice or quotient equality.

Composition exactness is constructive because Q is a finite pasting row. Given two displayed morphism approximation rows, the composed window transformer and tolerance ledger are written as coordinates of the packet before any downstream readback is consumed.

Finally H, C, P, N transport, replay, source, and name the same finite sequence. Since the carrier contains no coordinate for selected limits, quotient completions, ambient Real completeness, countable choice, or host category structure, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{CauchyApproximationCategory}^{\uparrow}$ read exposes object rows, morphism approximation rows, identity and composition routes, StreamName finite windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-approximation-system, cauchy-approximation-space, bishop-metric-completion, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyApproximationCategory}^{\uparrow}$ is not bijective to any listed sibling because it binds approximation-preserving arrows with identity and composition rows before any completion-facing consumer reads a Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyApproximationCategory}^{\uparrow}$ packet is generated from object rows, morphism approximation rows, identity and composition routes, finite StreamName windows, RegSeqRat readback, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8485.1 and Definition 8485.2 seed the category-facing Cauchy approximation carrier over objects, approximation-preserving morphisms, identity and composition routes, finite windows, regular readback, Real seals, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host category completeness, large-category universality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, object and morphism approximation exactness, identity and composition route exactness, finite-window access, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationCategoryUp.

Chapter 8486

A Concrete Naming Certificate for $\mathsf{RealModulusContinuity}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealModulusContinuity}^{\uparrow}$ packet records the finite handoff by which a displayed continuity modulus on real names is read through regular rational data before any $\mathsf{Real}^{\uparrow}$ consumer treats the value as Real-facing. It sits after the continuity carrier of chapter 130, the local modulus packet of chapter 6998, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a theorem-facing Real and Completion route; it is not a compactness theorem, a global uniform-continuity route, a quotient-real equality principle, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8486.1 (Real modulus continuity carrier)**.** *A* $\mathsf{RealModulusContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, \varepsilon, \mu, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *source row, recording the displayed real-valued map whose continuity modulus is being consumed.*
2. *ε is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive tolerance request for the target real-name read.*
3. *μ is the modulus row, recording the finite input-window budget licensed for the tolerance request ε.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and image names under μ.*
5. *D is the dyadic tolerance ledger comparing the requested output precision with the source-window budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after S, D, R have been displayed on the same packet route.*
8. *H records componentwise* hsame *transport for $F, \varepsilon, \mu, S, D, R, E$ and the local naming row.*
9. *C records displayed* Cont *replay through the continuity source, tolerance request, modulus row, windows, dyadic ledger, regular rational readback, real seal, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RealModulusContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BishopContinuityModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, \varepsilon, \mu, S, D, R, E, H, C, P, N$. It has no coordinate for compact-domain uniformity, quotient stream equality, selected real representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8486.2 (Real modulus continuity NameCert obligations)**.** *For the carrier of Definition 8486.1, the local* $\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$ *surface consists exactly of continuity-source exposure through F, tolerance-request exposure through ε, modulus-row exposure through μ, StreamName window exposure through S, dyadic comparison through D, RegSeqRat readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The continuity source F, tolerance request ε, and modulus row μ are the only rows that determine which continuity request is being read.

Read the observation side next. The stream windows S are inspected under the budget named by μ, the dyadic ledger D compares those windows with the requested tolerance, and the regular rational row R supplies the readback data.

Finally apply the terminal seal and structural rows. The Real row E is reached only after S, D, R are displayed, while H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further coordinates, the listed rows exhaust the NameCert surface. □

Theorem 8486.3 (Real modulus continuity RegSeqRat handoff)**.** *Every accepted* $\mathsf{RealModulusContinuity}^{\uparrow}$ *packet that exports a Real-facing continuity read must first factor through the displayed route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{BishopContinuityModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The handoff supplies regular rational readback under an explicit modulus budget; it does not export compact-domain uniformity, quotient stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligations of Definition 8486.2. They place the continuity source, tolerance request, modulus row, stream windows, dyadic ledger, and regular rational readback in one finite certificate surface.

Use the sibling rows named in the statement. The continuity source is read through chapter 130, the local modulus through chapter 6998, the finite observations through chapter 126, the tolerance arithmetic through chapter 291, and the rational readback through chapter 301.

Thus the Real-facing consumer receives regular rational data only after the displayed modulus route has been replayed. None of the excluded compactness, quotient, choice, trichotomy, or completeness coordinates occurs in the carrier. □

Theorem 8486.4 (Real modulus continuity Real seal)**.** *Let $M = (F, \varepsilon, \mu, S, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusContinuity}^{\uparrow}$ *carrier. The terminal* $\mathsf{Real}^{\uparrow}$ *seal E is consumable only after the same packet has exposed $F, \varepsilon, \mu, S, D, R, H, C, P, N$. Consequently a Real consumer cannot read the continuity value while bypassing the modulus row, the finite stream windows, the dyadic tolerance ledger, or the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Project the carrier of Definition 8486.1. The terminal Real row E is listed after the modulus, window, dyadic, and readback rows, and it is transported and replayed only through the same structural packet.

Apply the RegSeqRat handoff theorem Definition 8486.3. It forces every Real-facing continuity read through the continuous-map, modulus, stream-window, dyadic, and regular-rational route before the terminal seal is used.

Finally inspect H, C, P, N. These rows preserve the same packet and add no alternate Real source. Therefore a consumer that reaches E without the listed rows has left the displayed $\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$ surface. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusContinuity}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ continuity value without displaying a continuity source, modulus row, StreamName windows, DyadicRatCore tolerance ledger, and RegSeqRat readback refutes the carrier.

Independence witness: continuousmap, bishop-continuity-modulus, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{RealModulusContinuity}^{\uparrow}$ because this carrier jointly binds a continuity source, tolerance request, modulus budget, finite windows, dyadic comparisons, regular rational readback, terminal real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusContinuity}^{\uparrow}$ packet is generated from a ContinuousMap source row, Bishop continuity modulus row, DyadicRatCore tolerance request, StreamName finite windows, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 8486.1, Definition 8486.2, Definition 8486.3, and Definition 8486.4 seed the L10 Real and Completion continuity-modulus route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealModulusContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim compact-domain uniform continuity, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, StreamName window exposure, DyadicRatCore tolerance comparison, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local NameCert rows for BEDC.Derived.RealModulusContinuityUp.

Chapter 8487

A Concrete Naming Certificate for $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$

The $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ packet realizes the external-supply discipline of chapter 9395 as a finite audit-ledger route. It sits beside the external boundary packet of chapter 1065, the non-axiom boundary-form classifier of chapter 1066, the boundary-admission decision packet of chapter 1074, the composed audit-gate route of chapter 696, the GAP closure boundary of chapter 1006, and the external-supply audit route of chapter 1107. The packet records external supply as visible ledger data; it is not an axiom, not a theorem that the supply exists, and not an internal witness hidden inside a consumer.

Definition 8487.1 (External supply ledger route carrier)**.** *An* $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, A, G, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the external-supply boundary row, naming the boundary and requested supply shape.*
2. *F is the* $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ *row, recording the admitted non-axiom form for the request.*
3. *A is the* $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ *row, recording the decision that selects the strongest permitted visible form.*
4. *G is the audit-gate composition row, recording the gate route through which the request is accepted, refused, or held open.*
5. *R is the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *ledger row, recording the forward-binding memory that the requested supply has not been internalized as a theorem.*
6. *H records componentwise* hsame *transport for B, F, A, G, R and the local naming row.*
7. *C records displayed* Cont *replay through the boundary, boundary form, admission decision, audit gate, GAP ledger, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$, $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$, $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$, $\mathsf{AuditGateComposition}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, A, G, R, H, C, P, N$. It has no coordinate for an anonymous axiom, hidden host primitive, unrecorded supply witness, erased audit gate, or closed GAP memory.

Theorem 8487.2 (External supply ledger route NameCert obligations)**.** *For the carrier of Definition 8487.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}}$ *surface consists exactly of external boundary exposure through B, non-axiom boundary form exposure through F, boundary-admission decision through A, audit-gate composition through G, GAP closure-boundary ledger exposure through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The boundary row B states the requested supply shape, while F and A record the non-axiom form and admission decision that keep the request visible.

Read the audit side next. The gate row G records how the request is accepted, refused, or held open, and the GAP ledger row R records that the requested supply has not been smuggled into the theory as an internal theorem.

Finally inspect H, C, P, N. These rows only transport, replay, source, and name the same finite packet. Since the carrier has no anonymous supply coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 8487.3 (External supply ledger non-axiom route)**.** *Every accepted* $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ *consumer must read the route*

$$\mathsf{ExternalSupplyBoundary}^{\uparrow} \longmapsto \mathsf{NonAxiomBoundaryForm}^{\uparrow} \longmapsto \mathsf{BoundaryAdmissionDecision}^{\uparrow} \longmapsto \mathsf{AuditGateCompositio}$$

before treating an external-supply request as audit-visible. The route records the boundary, form, decision, gate, and ledger; it does not assert that the requested supply exists.

Proof. Start from the NameCert obligations of Definition 8487.2. They bind boundary exposure, admitted boundary form, admission decision, audit-gate composition, and GAP ledger exposure into one finite packet.

Use the sibling chapters named in the statement. The external boundary identifies the request, the non-axiom form prevents axiom import, the admission decision chooses the visible form, the audit gate records the route by which the request is judged, and the GAP boundary preserves memory of the still-external supply.

Thus a consumer receives an audit ledger rather than a supplied theorem. Since no carrier coordinate stores a witness for the requested supply, the route cannot be read as existence of that supply. □

Theorem 8487.4 (External supply ledger audit memory)**.** *Let $L = (B, F, A, G, R, H, C, P, N)$ be an accepted* $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ *carrier. Any public read of the requested supply shape must keep the GAP ledger row R and audit-gate row G visible after B, F, A are displayed. Therefore the packet cannot be repacked as an internal theorem, anonymous axiom, hidden witness, or erased-boundary supply source.*

Proof. Project the carrier of Definition 8487.1. The requested supply appears only in the boundary row B and is governed by the form and decision rows F, A.

Apply the non-axiom route theorem Definition 8487.3. It forces every audit-visible read through the composed gate G and GAP ledger R before a consumer may treat the request as a ledgered external supply.

Finally read the structural rows H, C, P, N. They preserve the same boundary, gate, and ledger rows and add no internal witness coordinate. A public read that hides G or R is therefore outside the displayed NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ read is refuted if it names an external supply boundary without also displaying the non-axiom boundary form, boundary-admission decision, audit-gate composition, and GAP closure-boundary ledger.

Independence witness: external-supply-boundary, nonaxiomboundaryform, boundary-admission-decision, auditgatecomposition, gapclosureboundary: none is bijective to $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ because this carrier jointly records boundary name, admitted non-axiom form, admission decision, composed audit route, GAP memory, transport, replay, provenance, and local NameCert rows.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ ExternalSupplyLedgerRoute}^{\uparrow}$ and $\mathsf{FieldFaithful\ ExternalSupplyl}$ instances before any checked TasteGate marker can be attached to this chapter. Those instances must keep $B, F, A, G, R, H, C, P, N$ as distinct rows and must not internalize the requested supply as a theorem.

Closure status for $\mathsf{NameCert}_{\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}}$.

Constructive story: An $\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}$ packet is generated from an external-supply boundary row, non-axiom boundary-form row, boundary-admission decision row, audit-gate composition row, GAP closure-boundary ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8487.1, Definition 8487.2, Definition 8487.3, and Definition 8487.4 seed the external-supply ledger route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExternalSupplyLedgerRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim that requested external supply exists, that any external source is internalized, that a hidden witness is available, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-form exactness, admission-decision exposure, audit-gate composition, GAP ledger retention, transport, replay, provenance, and local NameCert rows for BEDC.Derived.ExternalSupplyLedgerRouteUp, followed by Nontrivial and FieldFaithful instances.

Chapter 8488

A Concrete Naming Certificate for YoulaKuceraParametrization$^{\uparrow}$

Following Youla and Kucera on stabilizing-controller parametrization, the YoulaKuceraParametrization$^{\uparrow}$ packet records the finite control-theory surface by which a displayed plant row, coprime-factor row, free stable-parameter row, closed-loop stability ledger, and controller readback are named before robust-control consumers read a controller family. It sits after the quadratic-control ledger of chapter 296, the algebraic Riccati row of chapter 307, the control Riccati route of chapter 5103, the bounded-gain route of chapter 4752, and the finite-horizon replay of chapter 348. The packet gives robust-control consumers a finite parametrization certificate; it is not an analytic theorem selecting all stabilizing controllers.

Definition 8488.1 (Youla-Kucera parametrization carrier)**.** *A* YoulaKuceraParametrization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (P, C, Q, S, K, T, R, V, H, E, G, N)$$

with the following displayed rows:

1. *P is the plant row, recording the finite state-space, matrix, vector-space, and real-gain data consumed by the control packet.*
2. *C is the coprime-factor row, displaying the numerator, denominator, Bezout, and normalization ledgers used by the parametrization read.*
3. *Q is the free stable-parameter row, naming the displayed stable parameter offered to the controller family.*
4. *S is the closed-loop stability ledger, recording the finite route by which the selected parameter is checked against the plant and coprime-factor rows.*
5. *K is the controller readback row, carrying the controller expression obtained from P, C, Q, S.*
6. *T is the transport row, recording componentwise* hsame *transport for plant, factor, parameter, stability, controller, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from a robust-control request through P, C, Q, S, K, T.*
8. *V is the verification ledger row read by* HInfinityControl$^{\uparrow}$, LQR$^{\uparrow}$, Riccati$^{\uparrow}$, ControlRiccati$^{\uparrow}$, *and* ModelPredictiveControl$^{\uparrow}$ *consumers when they compare the controller family with finite gain, quadratic, Riccati, and rollout data.*

9. *H is the host-refusal row, separating transfer-function syntax, quotient controller equality, spectral factorization, infinite-dimensional realization, and selected analytic stabilizers from BEDC evidence.*

10. *E is the exactness row, requiring controller readback to use the same displayed plant, factor, parameter, and stability rows.*

11. *G is the* Pkg *provenance over* $\mathsf{YoulaKuceraParametrization}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, $\mathsf{HInfinityControl}^{\uparrow}$, $\mathsf{ModelPredictiveControl}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoulaKuceraParametrization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, C, Q, S, K, T, R, V, H, E, G, N$. It has no coordinate for host transfer functions, quotient controller equality, spectral factorization, infinite-dimensional realization, or an analytic selector for all stabilizing controllers.

Theorem 8488.2 (Youla-Kucera parametrization NameCert obligations)**.** *For every accepted* $\mathsf{YoulaKuceraParametrization}^{\uparrow}$ *carrier*

$$Y = (P, C, Q, S, K, T, R, V, H, E, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{YoulaKuceraParametrization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed plant row, coprime-factor row, free stable-parameter row, closed-loop stability ledger, controller readback, transport, replay, verification ledger, refusal row, exactness row, provenance, and local naming;*

2. *factor exposure, requiring every controller read to display P and C before it consumes the free parameter Q;*

3. *stability-ledger exactness, requiring S and E to bind the same plant, factor, parameter, and controller rows;*

4. *robust-control handoff, requiring V to expose the finite* $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, $\mathsf{HInfinityControl}^{\uparrow}$, *and* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *dependency rows before a consumer reads K as a controller-family output;*

5. *host-refusal preservation, requiring H, T, R, G, N to preserve the same finite route without adding selected analytic stabilizers, host transfer functions, quotient controller equality, spectral factorization, or infinite-dimensional realization.*

No Youla-Kucera parametrization obligation is stored outside $P, C, Q, S, K, T, R, V, H, E, G, N$.

Proof. Project the carrier of Definition 8488.1. The packet displays the plant, coprime-factor, free stable-parameter, stability-ledger, controller-readback, transport, replay, verification, refusal, exactness, provenance, and local naming rows.

Factor exposure is immediate from the order of the rows. A controller read reaches K only after the same plant row P, factor row C, parameter row Q, and stability ledger S have been displayed.

The exactness obligation is carried by E. It binds the controller readback to the displayed P, C, Q, S route, so componentwise transport or replay cannot replace the controller by a host-side transfer-function calculation.

The robust-control handoff is the row V. It names exactly the finite LQR, Riccati, control-Riccati, bounded-gain, and model-predictive-control dependency rows consumed by downstream control packets before K is public.

Finally, H, T, R, G, N keep the refusal and structural data attached to the same carrier. Since selected analytic stabilizers, quotient controller equality, spectral factorization, host transfer functions, and infinite-dimensional realizations have no coordinate in Y, the listed rows exhaust the local NameCert surface. □

Theorem 8488.3 (Youla-Kucera parametrization robust-control handoff)**.** *Every* $\mathsf{HInfinityControl}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, *or* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoulaKuceraParametrization}^{\uparrow}$ *carrier*

$$Y = (P, C, Q, S, K, T, R, V, H, E, G, N)$$

must read the displayed route

$$P, C, Q, S, K, V, E$$

before treating K as a controller-family output. The handoff exports a finite parametrization surface, not a selected analytic stabilizing controller, host transfer-function equality, quotient controller equality, spectral factorization, or infinite-dimensional realization.

Proof. Start with the NameCert obligations of Definition 8488.2. They force every controller read to expose P, C, Q, S, K and then to pass through the verification ledger V and exactness row E.

Read the downstream control rows through V. The finite LQR, Riccati, control-Riccati, bounded-gain, and model-predictive-control rows are dependency rows for the same carrier, not replacement analytic control theorems.

The refusal row H remains attached under T, R, G, N. Thus the consumer receives only the displayed parametrization route and cannot bypass it through host transfer functions, quotient equality, spectral factorization, selected stabilizers, or infinite-dimensional realization. □

Closure status for $\mathsf{NameCert}_{\mathsf{YoulaKuceraParametrization}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoulaKuceraParametrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8488.1, Definition 8488.2, and Definition 8488.3 seed the finite parametrization route from plant and coprime-factor rows through a free stable-parameter row to controller readback and robust-control dependency rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoulaKuceraParametrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected analytic stabilizing controllers, host transfer functions, quotient controller equality, spectral factorization, infinite-dimensional realization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, factor exposure, stability-ledger exactness, robust-control handoff, host-refusal preservation, transport, replay, provenance, and local naming for BEDC.Derived.YoulaKuceraParametrizati

Constructive story: A $\mathsf{YoulaKuceraParametrization}^{\uparrow}$ packet is generated from a finite plant row, coprime-factor row, free stable-parameter row, closed-loop stability ledger, controller readback, componentwise hsame transport, displayed Cont replay, robust-control verification ledger, host-refusal row, exactness row, Pkg provenance, and local NameCert row.

Chapter 8489

A Concrete Naming Certificate for $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ packet records a constructive uniform-space completion as finite BEDC NameCert data. It sits after the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the metric carrier of chapter 57, and the entourage completion surface of chapter 8414. The packet gives the L10 completion route a located Cauchy-filter basis, regular observation windows, dense embedding rows, separated classifier rows, and structural provenance without quotienting filters, selecting representatives, invoking countable choice, or importing host equality.

Definition 8489.1 (Bishop uniform-space completion carrier)**.** *A* $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, L, F, W, R, D, E, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{UniformSpace}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *row, exposing the finite entourage or metric requests that the completion consumer may inspect.*
2. *L is the located uniformity row, recording finite positive-radius or entourage-location evidence before any Cauchy-filter read is made.*
3. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *row, carrying the located Cauchy-filter base through displayed finite requests.*
4. *W is the regular finite observation-window row over* $\mathsf{StreamName}^{\uparrow}$*, and R is the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare finite observations and filter requests.*
6. *E is the dense embedding row from the source carrier into the completion packet; it records graph membership before equality or separation is read.*
7. *S is the separated classifier row, comparing accepted completion names only through the displayed located Cauchy-filter base, observation windows, readback rows, tolerance ledger, and dense embedding row.*
8. *H records componentwise* hsame *transport for X, L, F, W, R, D, E, S and for the structural rows.*

9. *C records displayed* Cont *replay from a completion request through the located filter base, observation windows, readback, tolerance, dense embedding, and separated classifier.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{EntourageCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformSpaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient filter equality, host uniform-space equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8489.2 (Bishop uniform-space completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ *carrier* $U = (X, L, F, W, R, D, E, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopUniformSpace}}$ *surface consists of five rows:*

1. *source and located-uniformity admission, requiring X and L to display every entourage, metric, or positive-radius request before the completion route is read;*

2. *located Cauchy-filter basis exposure, requiring F to carry only finite filter-base requests over the displayed source and located rows;*

3. *regular finite observation, requiring W, R, D to expose StreamName windows, RegSeqRat readback, and DyadicRatCore tolerances for the same filter requests;*

4. *dense embedding and separated classifier exactness, requiring E and S to compare completion names through the displayed source graph, located Cauchy-filter base, observation windows, and tolerance ledger;*

5. *structural provenance, requiring H, C, P, N to transport, replay, source, and locally name the same packet without adding quotient filters, selected representatives, countable choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 8489.1. The displayed mathematical rows are X, L, F, W, R, D, E, S, and the structural rows are H, C, P, N.

The source and located rows X, L precede every Cauchy-filter request. The filter-base row F is therefore read as a finite located basis rather than as a completed quotient object.

The observation rows W, R, D bind every inspected request to finite StreamName windows, regular rational readback, and dyadic tolerances. Dense embedding and separation are then read through E, S, so equality-sensitive completion reads remain attached to displayed evidence.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for quotient filters, selected limits, countable choice, host equality, or ambient completeness, the five listed rows exhaust the local certificate surface. □

Theorem 8489.3 (Bishop uniform-space completion dense image)**.** *Every accepted* $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ *carrier* $U = (X, L, F, W, R, D, E, S, H, C, P, N)$ *exports its source object to completion-facing consumers only through the dense embedding row E and the separated classifier row S. A consumer that reads a completion name must first expose the located source rows X, L, the Cauchy-filter basis F, the regular observation route W, R, D, and then the displayed embedding and separation rows. Thus the source image is dense in the completion packet as a finite NameCert route, not as a quotient completion, a choice-selected representative, host equality, or an ambient completeness theorem.*

Proof. Start from the NameCert obligations of Definition 8489.2. They force every completion-facing read to expose the source, located-uniformity, Cauchy-filter basis, observation, readback, and tolerance rows before any endpoint is consumed.

The dense embedding row E records the source graph inside the completion packet. Since E is downstream of X, L, F, W, R, D, an image read cannot be detached from the finite located Cauchy-filter evidence that produced it.

The separated classifier row S is the only comparison row for completion names. It reads through E and the same observation surface, so density is witnessed by the displayed route rather than by quotienting filters or selecting representatives.

Transport, replay, provenance, and local naming are carried by H, C, P, N. These rows preserve the same packet and add no host equality or ambient completeness data, giving the claimed dense-image handoff. □

Falsifiable prediction: Every accepted $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ consumer that reaches a completion name while omitting the located source rows, Cauchy-filter basis row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, dense embedding row, separated classifier row, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the carrier.

Independence witness: uniform-space, metric, cauchy-filter-basis, cauchy-filter, streamname, regseqrat, dyadicratcore, entourage-completion: no listed sibling is bijective to $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ because this packet jointly binds located uniform-space requests, Cauchy-filter bases, regular finite observation windows, dense embedding rows, separated classifier rows, transport, replay, provenance, and local naming as one completion-facing NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformSpaceCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformSpaceCompletion}^{\uparrow}$ packet is generated from a source uniform-space or metric row, located uniformity evidence, a located Cauchy-filter basis, StreamName observation windows, RegSeqRat readbacks, DyadicRatCore tolerances, a dense embedding row, a separated classifier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformSpaceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8489.1, Definition 8489.2, and Definition 8489.3 seed the located uniform-space completion route over Cauchy-filter bases, regular finite observation windows, dense embedding rows, separated classifier rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformSpaceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host uniform-space equality, choice-selected representatives, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located Cauchy-filter basis exposure, regular observation exactness, dense embedding exactness, separated classifier exactness, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformSpaceCompletionUp.

Chapter 8490

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$

The $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ packet realizes the observer-horizon branch of chapter 9463 as finite boundary-shadow, Busemann-cocycle, phase-distance, transport, provenance, and refusal rows. It sits after the boundary-shadow cross-ratio route of chapter 8460, the hyperbolic observer phase route of chapter 7112, the phase-transport budget of chapter 7134, the Gromov-boundary readback of chapter 6470, and the visual-flow shadow route of chapter 6609. The packet names a finite cocycle ledger for horizon-facing observer reads; it is not a global hyperbolic model, quotient boundary theorem, selected geodesic ray, physical observer, or analytic ergodic statement.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ read is refuted if it reaches a phase-distance or horizon-cocycle conclusion without displaying the boundary-shadow, Busemann-cocycle, observer-horizon, transport, provenance, and refusal rows.

Independence witness: boundary-shadow-cross-ratio, hyperbolic-observer-phase, hyperbolic-phase-transport-budget, gromov-boundary, hyperbolic-visual-flow-shadow: none is bijective to $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ because this packet jointly carries an observer horizon row, Busemann-cocycle ledger, phase-distance readback, transport replay, provenance, refusal, and local NameCert naming.

Later Lean work must supply Nontrivial and FieldFaithful TasteGate instances before any positive discovery claim can be read from this packet. This chapter records the finite carrier and obligation surface only.

Definition 8490.1 (Hyperbolic observer horizon-cocycle carrier)**.** *A* $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, U, H, D, C, A, T, R, P, F, N)$$

with the following displayed rows:

1. *B is the boundary-shadow row, recording the visible shadow window seen by the observer.*
2. *U is the Busemann-cocycle row, recording the finite signed comparison between two displayed horizon reads.*
3. *H is the observer-horizon row, tying the boundary-shadow and Busemann rows to a displayed observer expansion route.*

4. *D is the phase-distance row, recording the finite readback consumed by phase-geometry consumers.*
5. *C is the cocycle-consistency ledger, requiring additivity and basepoint transport to be read only through displayed finite rows.*
6. *A is the apophatic refusal row, preventing the horizon far end from becoming an interior center, selected geodesic endpoint, or physical observer.*
7. *T records componentwise* hsame *transport for boundary-shadow, Busemann-cocycle, horizon, phase-distance, cocycle ledger, refusal, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an observer request through B, U, H, D, C, A, T.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$, $\mathsf{BoundaryShadowCrossRatio}^{\uparrow}$, $\mathsf{HyperbolicObserverPhase}^{\uparrow}$, $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$, $\mathsf{HyperbolicVisualFlowShadow}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *F is the FieldFaithful readiness row, recording that later field-facing comparison must preserve the same finite phase-distance and Busemann-cocycle data.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, U, H, D, C, A, T, R, P, F, N$. It has no coordinate for a quotient visual boundary, selected geodesic ray, physical observer state, ambient hyperbolic manifold, analytic ergodic invariant, or hidden basepoint equality.

Theorem 8490.2 (Hyperbolic observer horizon-cocycle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ *carrier*

$$O = (B, U, H, D, C, A, T, R, P, F, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the boundary-shadow, Busemann-cocycle, observer-horizon, phase-distance, cocycle-ledger, refusal, transport, replay, provenance, FieldFaithful-readiness, and local naming rows;*
2. *boundary-shadow exposure, requiring every horizon read to display B before it consumes the cocycle row U;*
3. *Busemann-cocycle exactness, requiring U, C, T, R to preserve the same finite horizon comparison under basepoint transport and replay;*
4. *phase-distance handoff, requiring D and F to remain attached before any phase-geometry or field-facing consumer reads the packet;*
5. *apophatic refusal, requiring A, P, N to keep selected rays, physical observers, quotient boundary equality, ambient manifolds, and analytic ergodic invariants outside the certificate surface.*

No observer horizon-cocycle obligation is stored outside $B, U, H, D, C, A, T, R, P, F, N$.

Proof. Project the accepted carrier from Definition 8490.1. The packet displays the boundary-shadow, Busemann-cocycle, observer-horizon, phase-distance, cocycle-ledger, refusal, transport, replay, provenance, FieldFaithful-readiness, and local naming rows.

Boundary-shadow exposure comes first. A horizon read reaches the Busemann row only after the visible shadow window B has been displayed, so the horizon cannot be replaced by a hidden ideal-boundary representative.

Busemann-cocycle exactness is the joint role of U, C, T, R. The cocycle ledger C records the finite comparison, while transport and replay preserve the same basepoint-facing route.

The phase-distance handoff uses D and F. These rows keep the phase readback and later field-facing comparison attached to the same finite observer route.

Finally, the refusal row A and provenance rows P, N exclude selected rays, physical observer states, quotient boundary equality, ambient manifolds, and analytic ergodic invariants. Since none of those objects is a coordinate of O, the listed rows exhaust the local NameCert surface. □

Theorem 8490.3 (Hyperbolic observer horizon-cocycle phase handoff)**.** *Every phase-geometry consumer of an accepted* $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ *carrier*

$$O = (B, U, H, D, C, A, T, R, P, F, N)$$

must read the displayed route

$$B, U, H, D, C, A, T, R, P, F, N$$

before treating the packet as horizon-cocycle data. The handoff exports finite boundary-shadow, Busemann-cocycle, phase-distance, transport, provenance, refusal, and readiness rows, not quotient boundary equality, selected rays, physical observer states, ambient hyperbolic manifolds, or analytic ergodic invariants.

Proof. Use the obligation theorem Definition 8490.2. It places boundary-shadow exposure before Busemann-cocycle readback, then keeps the observer-horizon and phase-distance rows attached to the same finite route.

Read transport and replay next. Rows T and R preserve the displayed cocycle ledger C rather than introducing an unshown basepoint equality or quotient boundary representative.

Finally apply the refusal and readiness rows. The apophatic row A blocks selected-ray and physical-observer interpretations, while F records only later FieldFaithful readiness. Thus a phase consumer receives the finite horizon-cocycle packet and no stronger analytic or quotient endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8490.1, Definition 8490.2, and Definition 8490.3 seed the finite horizon-cocycle route from boundary-shadow and Busemann-cocycle rows through phase-distance, transport, provenance, refusal, readiness, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient visual-boundary equality, selected geodesic rays, physical observer states, ambient hyperbolic manifolds, analytic ergodic invariants, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exposure, Busemann-cocycle exactness, phase-distance handoff, apophatic refusal, Nontrivial TasteGate support, FieldFaithful TasteGate support, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicObserverHorizonCocycleUp.

Constructive story: A $\mathsf{HyperbolicObserverHorizonCocycle}^{\uparrow}$ packet is generated from boundary-shadow row, Busemann-cocycle row, observer-horizon row, phase-distance row, cocycle-consistency ledger, apophatic refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, FieldFaithful readiness, and a local NameCert row.

Chapter 8491

A Concrete Naming Certificate for $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ packet records the finite handoff between regular Cauchy data and completion-facing consumers. It sits over the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the completion-operator interface of chapter 3379. The packet is a finite enriched completion route, not a quotient completion, a selected limit, countable choice, host real equality, or an ambient completeness principle.

Definition 8491.1 (Regular Cauchy enriched completion carrier)**.** *A* $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, M, A, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger, naming the visible stream windows consumed by the completion route.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational approximants attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic error scale used by the regularity test.*
4. *M is the explicit modulus row, giving the finite bound by which later windows are read at a requested tolerance.*
5. *A is the enrichment row, retaining the companion annotations that make the completion consumer inspect a regular Cauchy schedule rather than a bare stream name.*
6. *O is the* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *handoff row, exposing the completion-operator call site without choosing a quotient representative or a host limit.*
7. *H records componentwise* hsame *transport for stream windows, rational readback, dyadic tolerances, modulus data, enrichment rows, completion-operator handoff, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion query through $S, R, D, M, A, O, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, D, M, A, O, H, C, P, N$. It contains no coordinate for quotient real equality, a selected limit, countable choice, Dedekind completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8491.2 (Regular Cauchy enriched completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ *carrier*

$$E = (S, R, D, M, A, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}}$ *surface consists exactly of stream-window exposure through S, regular rational readback through R, dyadic tolerance exposure through D, modulus exposure through M, enrichment retention through A, completion-operator handoff through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8491.1. Rows S, R, D, M, A, O are the completion-facing finite data, while H, C, P, N are the structural certificate rows.

The stream and rational parts are read through S and R. The dyadic tolerance and modulus parts are read through D and M, so every requested completion tolerance remains tied to a displayed finite bound.

The enrichment and completion-operator parts are then read through A and O. These rows expose the regular Cauchy schedule and the operator handoff, but they do not select a quotient representative or assert a host limit.

Finally H, C, P, N transport the displayed rows, replay the completion query, source the packet, and name it. Since no carrier coordinate contains choice, quotient real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8491.3 (Regular Cauchy enriched completion modulus handoff)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ *carrier factors the requested tolerance through the dyadic row D and the modulus row M before it reaches the* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *handoff row O. In particular, a consumer cannot bypass the displayed modulus to read the packet as an unbounded Cauchy stream or as an already completed* $\mathsf{Real}^{\uparrow}$ *object.*

Proof. Start with a completion query. The query first reads the finite stream windows through S and their rational values through R, so it has only displayed approximants at this stage.

The tolerance request then passes through D and M. Row D supplies the dyadic scale, and row M supplies the finite bound that makes later windows inspectable at that scale.

Only after those rows are present can the query reach O. The handoff to $\mathsf{CauchyCompletionOperator}^{\uparrow}$ therefore carries its modulus data with it, and no consumer can detach the handoff from the finite regularity evidence. □

Theorem 8491.4 (Regular Cauchy enriched completion non-escape)**.** *Every public read exported by an accepted* $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ *carrier factors through $S, R, D, M, A, O, H, C, P, N$. The packet cannot be used to extract quotient real equality, a selected limit, countable choice, Dedekind completeness, open-cover compactness, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Inspect the carrier inventory. The packet contains stream windows, regular rational readback, dyadic tolerance data, an explicit modulus, enrichment rows, a completion-operator handoff, transport, replay, provenance, and local naming.

Each public read is one of those rows or a replay through them. The completion-facing coordinate is O, and it is guarded by the preceding stream, rational, dyadic, modulus, and enrichment rows.

None of the excluded objects appears as a coordinate. Hence the carrier is a finite NameCert handoff for regular Cauchy completion data and not a source of quotient equality, selected limits, choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, explicit modulus data, enrichment rows, CauchyCompletionOperator handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8491.1, Definition 8491.2, Definition 8491.3, and Definition 8491.4 seed the finite enriched completion handoff for regular Cauchy consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEnrichedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, Dedekind completeness, open-cover compactness, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular rational readback, dyadic tolerance exposure, modulus handoff, enrichment retention, completion-operator exactness, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEnrichedCompletionUp.

Chapter 8492

A Concrete Naming Certificate for $\mathsf{HomotopyExtensionProperty}^{\uparrow}$

Following Borsuk and Dugundji on homotopy extension theory, the $\mathsf{HomotopyExtensionProperty}^{\uparrow}$ packet records the finite surface by which a homotopy already defined on a closed subcarrier is extended across a displayed cylinder window. It sits after the $\mathsf{DugundjiExtension}^{\uparrow}$ extension-operator route of chapter 2121, the $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$ neighbourhood-retraction route of chapter 6562, the $\mathsf{Homotopy}^{\uparrow}$ deformation ledger of chapter 156, the $\mathsf{ClosedMetricSubspace}^{\uparrow}$ admission route of chapter 7567, and the $\mathsf{ContinuousMap}^{\uparrow}$ graph row of chapter 130. The packet names the homotopy-extension property as displayed NameCert data; it does not import a host cofibration theorem, quotient cylinder equality, selected neighbourhood deformation, or ambient extension operator.

Definition 8492.1 (Homotopy extension property carrier)**.** *A* $\mathsf{HomotopyExtensionProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, X, I, F, G, D, R, W, T, C, P, N)$$

with the following displayed rows:

1. *A is the closed subcarrier row, read through* $\mathsf{ClosedMetricSubspace}^{\uparrow}$ *and retaining the admitted inclusion of the displayed subcarrier in the ambient metric carrier.*
2. *X is the ambient carrier row, recording only the parent metric, topology, and continuous-map data visible to the extension request.*
3. *I is the cylinder homotopy window, a finite interval-and-endpoint ledger for the displayed $A \times I$ and $X \times I$ read.*
4. *F is the initial continuous-map row on X, read through* $\mathsf{ContinuousMap}^{\uparrow}$*.*
5. *G is the subcarrier homotopy row on $A \times I$, read through* $\mathsf{Homotopy}^{\uparrow}$ *and compatible with F on the initial face.*
6. *D is the Dugundji or ANR handoff row. It records the displayed extension interface supplied by* $\mathsf{DugundjiExtension}^{\uparrow}$ *or* $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$*, not a host-selected extension.*
7. *R is the extension request row, binding A, X, I, F, G, D into one finite request for a homotopy over $X \times I$.*
8. *W is the extended homotopy window row, recording the candidate extension only through the same displayed cylinder, initial-face, and subcarrier rows.*

9. *T records componentwise* hsame *transport for closed-subcarrier, ambient, cylinder, initial-map, subcarrier-homotopy, handoff, request, window, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an extension query through $A, X, I, F, G, D, R, W, T$.*
11. *P is the* Pkg *provenance over* $\mathsf{HomotopyExtensionProperty}^{\uparrow}$, $\mathsf{DugundjiExtension}^{\uparrow}$, $\mathsf{AbsoluteNeighborhoodRetract}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{ClosedMetricSubspace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HomotopyExtensionProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, X, I, F, G, D, R, W, T, C, P, N$. It has no coordinate for an ambient cofibration theorem, quotient cylinder equality, selected global deformation, host function-space topology, or choice-selected extension operator.

Lean4 ✓ `BEDC.Derived.HomotopyExtensionPropertyUp.HomotopyExtensionPropertyTasteGate single carrier alignment`

Theorem 8492.2 (Homotopy extension property NameCert obligations)**.** *For every accepted* $\mathsf{HomotopyExtensionProperty}^{\uparrow}$ *carrier*

$$E = (A, X, I, F, G, D, R, W, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HomotopyExtensionProperty}^{\uparrow}}$ *surface consists exactly of closed subcarrier admission through A; ambient carrier exposure through X; cylinder homotopy-window exposure through I; initial continuous-map exposure through F; subcarrier homotopy exposure through G; Dugundji or ANR handoff through D; extension request binding through R; extended-window readback through W; componentwise transport through T; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8492.1. The rows A and X display the closed subcarrier inside the ambient carrier, while I fixes the finite cylinder window on which the homotopy-extension request is inspected.

The analytic rows are F, G, D, R, W. The initial map F and the subcarrier homotopy G must be compatible before the handoff row D can be read. The request row R then binds those rows, and W records only the resulting displayed extended window.

The structural rows T, C, P, N preserve the same request under transport, replay, provenance, and local naming. Since the carrier contains no host cofibration theorem, quotient cylinder equality, selected deformation, function-space topology, or choice-selected operator, the listed rows exhaust the local NameCert obligations. □

Theorem 8492.3 (Homotopy extension Dugundji and ANR handoff)**.** *Every Dugundji-facing or ANR-facing consumer of an accepted* $\mathsf{HomotopyExtensionProperty}^{\uparrow}$ *carrier must read the route*

$$A, X, I, F, G \longmapsto D \longmapsto R, W, T, C, P, N$$

before treating the packet as homotopy-extension evidence. The handoff may cite the extension surface of chapter 2121 or the neighbourhood-retract surface of chapter 6562, but it exports only finite BEDC rows and not a host cofibration theorem, selected neighbourhood deformation, quotient cylinder equality, or global extension operator.

Proof. Start with the NameCert obligations of Definition 8492.2. They force every consumer to expose A, X, I, F, G before the handoff row D can be public.

Read D through the sibling extension route. A Dugundji consumer receives a displayed extension-operator interface, while an ANR consumer receives a displayed neighbourhood-retraction interface; neither route replaces the closed subcarrier, cylinder, initial-map, or homotopy rows.

Finally use R, W, T, C, P, N. These rows bind the request, expose the extended window, and preserve the packet under transport, replay, provenance, and local naming. Thus the handoff remains a finite homotopy-extension property surface rather than an ambient cofibration or extension theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{HomotopyExtensionProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{HomotopyExtensionProperty}^{\uparrow}$ packet is generated from a closed subcarrier row, ambient carrier row, cylinder homotopy-window row, initial continuous-map row, subcarrier homotopy row, Dugundji or ANR handoff row, extension request row, extended-window readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HomotopyExtensionProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8492.1, Definition 8492.2, and Definition 8492.3 seed the homotopy-extension property route over closed-subcarrier, cylinder, homotopy, Dugundji, ANR, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HomotopyExtensionProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close cofibration recognition, quotient cylinder equality, selected neighbourhood deformation, host function-space topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked closed-subcarrier admission, cylinder-window exposure, initial-map exposure, subcarrier-homotopy exposure, Dugundji or ANR handoff, request binding, extended-window readback, transport, replay, provenance, and local naming for BEDC.Derived.HomotopyExtensionPropertyUp.

Chapter 8493

A Concrete Naming Certificate for $\mathsf{HigmanLemma}^{\uparrow}$

Following Higman 1952 and Nash-Williams' well-quasi-order method, the $\mathsf{HigmanLemma}^{\uparrow}$ packet records finite word-embedding data over a well-quasi-ordered source as a NameCert carrier. It sits after the finite-product induction packet of chapter 6754, the finite list rows of chapter 36, the ordered comparison rows of chapter 53, and the candidate-boundary rows of chapter 892. The packet is a finite subsequence-embedding certificate for words; it is not a selected infinite subsequence theorem, not a quotient of streams, and not a host proof of MetaCIC normalization.

Definition 8493.1 (Higman lemma carrier)**.** *A* $\mathsf{HigmanLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{Higman}(S, W, E, B, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the source well-quasi-order row, exposing only finite comparisons and bad-prefix refusals for the underlying alphabet.*
2. *W is the finite-word row, storing word windows as* $\mathsf{List}^{\uparrow}$ *packets over S.*
3. *E is the subsequence-embedding row, recording monotone index maps and pointwise source comparisons for a pair of displayed word windows.*
4. *B is the bad-sequence-prefix row, storing the finite obstruction prefix that must be reduced when no embedding row is displayed.*
5. *D is the Dickson-style product-order dependency row read through* $\mathsf{DicksonLemma}^{\uparrow}$ *for the finite vector of front letters and residual word lengths.*
6. *H records componentwise* hsame *transport for source, word, embedding, obstruction, dependency, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a bad-prefix read to a shorter residual-word read and then to an embedding read.*
8. *P is the* Pkg *provenance over* $\mathsf{HigmanLemma}^{\uparrow}$, $\mathsf{DicksonLemma}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HigmanLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for countable choice, quotient word equality, a selected infinite subsequence, excluded-middle case selection, or an unconditional MetaCIC termination theorem.

Lean4 ✓ `BEDC.Derived.HigmanLemmaUp.TasteGate.HigmanLemmaTasteGate single carrier alignment`

Theorem 8493.2 (Higman lemma NameCert obligations)**.** *For every accepted* $\mathsf{HigmanLemma}^{\uparrow}$ *carrier* $\mathsf{Higman}(S, W, E, B, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HigmanLemma}^{\uparrow}}$ *surface consists exactly of source well-quasi-order exposure through* S*, finite-word exposure through* W*, subsequence-embedding exposure through* E*, bad-sequence-prefix exposure through* B*, Dickson dependency exposure through* D*, componentwise transport through* H*, continuation replay through* C*, provenance nonescape through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The source row S, word row W, embedding row E, obstruction row B, and dependency row D are the only mathematical rows available before the structural rows are read.

The source and word rows fix the alphabet and finite windows. The embedding row records only monotone index maps together with pointwise source comparisons, while the bad-prefix row records the finite obstruction prefix that remains when no such embedding is displayed.

The dependency row is read through chapter 6754. It supplies a finite product-order route for the front-letter and residual-length bookkeeping, but it does not add a selected infinite subsequence or a quotient word equality.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for countable choice, excluded-middle selection, quotient words, or an unconditional MetaCIC termination theorem, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.HigmanLemmaUp.HigmanLemmaNameCertObligations`

Theorem 8493.3 (Higman lemma finite subsequence embedding route)**.** *Every downstream consumer of an accepted* $\mathsf{HigmanLemma}^{\uparrow}$ *packet that asks for a word-embedding certificate must read the finite route*

$$\mathsf{List}^{\uparrow} \to \mathsf{Poset}^{\uparrow} \to \mathsf{DicksonLemma}^{\uparrow} \to \mathsf{HigmanLemma}^{\uparrow} \to \mathsf{NameCert}.$$

The consumer receives finite word windows, source comparisons, monotone index data, bad-prefix reduction data, Dickson-style product bookkeeping, componentwise transport, replay, provenance, and local naming. It receives no selected infinite subsequence, no quotient stream equality, no countable-choice selector, and no global normalization theorem.

Proof. Start with the NameCert obligations of Definition 8493.2. They force a consumer to expose the source row S, word row W, embedding row E, obstruction row B, and dependency row D before it reads the local naming row.

The word row is a finite $\mathsf{List}^{\uparrow}$ window over the source comparison surface. The embedding row is a displayed monotone index map with pointwise comparisons, so it is a finite certificate rather than an infinite subsequence selection.

The dependency row routes the finite obstruction reduction through $\mathsf{DicksonLemma}^{\uparrow}$. This places the front-letter and residual-length bookkeeping on the same finite product-order surface already named by chapter 6754.

Finally the structural rows carry the route to normalization-facing consumers. A $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ reader may consume the packet only as a finite residual-word evidence row, because P names no quotient stream equality, countable-choice selector, or global normalization theorem. □

Lean4 ✓ `BEDC.Derived.HigmanLemmaUp.HigmanLemmaSubsequenceEmbeddingRoute`

Theorem 8493.4 (Higman lemma normalization-frontier handoff)**.** *A normalization-frontier consumer may cite* HigmanLemma$^{\uparrow}$ *only after the finite word packet exposes* $S, W, E, B, D, H, C, P, N$ *and after the consumer has identified which residual-word window is being handed to the* MetaCICNormalizationFront *boundary. The handoff is a finite subsequence-embedding certificate for residual windows; it does not close subject reduction, confluence, strong normalization, quotient syntax equality, or host evaluator soundness.*

Proof. Read the finite subsequence route of Definition 8493.3. It already places the word windows, source comparisons, embedding data, bad-prefix reduction, Dickson dependency, and structural rows before any downstream handoff.

Now compare with the MetaCIC normalization frontier of chapter 892. That frontier separates candidate evidence, finished-normal evidence, and obstruction rows; the Higman packet may feed only the finite residual-word evidence side of that separation.

Thus the handoff preserves a finite certificate boundary. It can record that a residual-word window has an embedding route, but it cannot erase the MetaCIC obstruction rows or turn the finite combinatorial packet into a global normalization theorem. □

Lean4 ✓ `BEDC.Derived.HigmanLemmaUp.HigmanLemmaNormalizationFrontierHandoff`

Theorem 8493.5 (Higman lemma finite bad-prefix induction)**.** *Any accepted residual-word consumer of a* HigmanLemma$^{\uparrow}$ *packet may perform induction only on the displayed finite bad-prefix windows carried by* B *and routed through* D, C, P, N*. At each induction step the consumer must keep the* MetaCICNormalizationFrontier$^{\uparrow}$ *obstruction row visible, so the induction exports a shorter residual-word window or a finite embedding row, not a global normalization theorem.*

Lean4 ✓ `BEDC.Derived.HigmanLemmaUp.HigmanLemmaFiniteBadPrefixInduction`

Proof. Start with the carrier of Definition 8493.1. The only obstruction data available to a residual-word consumer is the finite bad-sequence-prefix row B, together with the Dickson dependency row D and the structural replay rows C, P, N.

Induction therefore has a finite displayed domain. The base case reads an already displayed embedding row E, while an induction step shortens only the bad-prefix window named by B and records the dependency through D.

Now compare the handoff of Definition 8493.4. A MetaCIC-facing consumer must preserve the normalization-frontier obstruction row while reading the residual-word evidence.

Thus the induction produces only a shorter displayed residual window or a finite embedding certificate. It cannot erase the obstruction row, select an infinite subsequence, or turn the finite Higman packet into global normalization. □

Falsifiable prediction: A claimed HigmanLemmaUp certificate fails if its word-embedding row lacks either a monotone index map or the pointwise source-comparison rows for the displayed finite word windows.

Independence witness: dickson-lemma, list, poset, metacicnormalizationfrontier: DicksonLemma$^{\uparrow}$ supplies finite product-order bookkeeping without word embedding rows; List$^{\uparrow}$ supplies finite word windows without bad-prefix reduction; Poset$^{\uparrow}$ supplies comparison rows without residual-word replay; and MetaCICNormalizationFrontier$^{\uparrow}$ consumes candidate-boundary evidence without being a word-embedding certificate. None is bijective to HigmanLemma$^{\uparrow}$ under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{HigmanLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{HigmanLemma}^{\uparrow}$ packet is generated from a source well-quasi-order row, finite word rows, subsequence-embedding rows, bad-sequence-prefix rows, Dickson dependency rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HigmanLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8493.1, Definition 8493.2, Definition 8493.3, and Definition 8493.4 seed the finite word-embedding packet and its MetaCIC normalization-frontier handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HigmanLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected infinite subsequences, quotient word equality, excluded-middle case selection, subject reduction, confluence, strong normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source exposure, finite-word exposure, subsequence-embedding exposure, bad-prefix reduction, Dickson dependency routing, componentwise transport, replay, provenance, and local NameCert exactness for BEDC.Derived.HigmanLemmaUp.

Chapter 8494

A Concrete Naming Certificate for $\mathsf{HewittRealcompactification}^{\uparrow}$

Following Hewitt and Kelley on realcompact spaces, the $\mathsf{HewittRealcompactification}^{\uparrow}$ packet records the finite real-valued-function completion boundary seen by BEDC consumers before any ambient realcompactification theorem is available. It sits after the completely regular source of chapter 5161, the topology carrier of chapter 88, the continuous-map surface of chapter 130, the terminal real-name seal of chapter 38, and the bounded-probe comparison of chapter 2344. The packet names the displayed real-function handoff only; it does not claim z-ultrafilter choice, host compactification equality, selected points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8494.1 (Hewitt realcompactification carrier)**.** *A* $\mathsf{HewittRealcompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (C, T, F, R, B, Z, E, L, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *source row, exposing the point-versus-closed-set real-valued separation surface.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row naming the open-set carrier on which the real-valued observations are read.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row listing the visible real-valued probes used by the completion boundary.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for the displayed probe evaluations.*
5. *B is the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *comparison row, used only as a bounded-probe sibling surface.*
6. *Z is the zero-set and cozero-set ledger row recording the finite real-function tests seen by the packet.*
7. *E is the realcompactification envelope row, exposing the completion-facing request without selecting z-ultrafilters or points.*
8. *L records componentwise* hsame *transport for the source, topology, continuous-map, Real, Stone–Cech comparison, zero-set ledger, envelope, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a real-function completion request through $C, T, F, R, B, Z, E, L, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HewittRealcompactification}^{\uparrow}$, $\mathsf{CompletelyRegularSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StoneCechCompactification}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HewittRealcompactification}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $C, T, F, R, B, Z, E, L, K, P, N$. It contains no coordinate for a selected z-ultrafilter, a chosen realcompactification point, host compactification equality, a full ring of all real-valued functions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8494.2 (Hewitt realcompactification NameCert obligations)**.** *For every accepted* HewittRealcompactificat *carrier*

$$H = (C, T, F, R, B, Z, E, L, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HewittRealcompactification}^{\uparrow}}$ *surface consists exactly of completely regular source exposure through C, topology exposure through T, continuous real-valued probe exposure through F, Real value sealing through R, Stone–Cech comparison through B, zero-set ledger exposure through Z, realcompactification-envelope exposure through E, componentwise transport through L, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8494.1. Rows C, T, F, R expose the completely regular source, its topology, the displayed continuous real-valued probes, and their Real value seals.

Rows B, Z, E then record only the comparison and envelope data. The Stone–Cech row is a sibling bounded-probe comparison, the zero-set ledger lists finite real-function tests, and the envelope row names the completion-facing request without adding a selected z-ultrafilter or point.

Finally L, K, P, N transport, replay, source, and name exactly those rows. Since no excluded object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8494.3 (Hewitt realcompactification real-function handoff)**.** *Every completion-facing read of an accepted* $\mathsf{HewittRealcompactification}^{\uparrow}$ *carrier first exposes the route*

$$\mathsf{CompletelyRegularSpace}^{\uparrow} \longmapsto \mathsf{Topology}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and only then reads the $\mathsf{StoneCechCompactification}^{\uparrow}$ *comparison row, the zero-set ledger, and the realcompactification envelope. Thus the packet exports a finite real-function handoff, not z-ultrafilter choice, host compactification equality, selected points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing query. The query must read C and T before any real-valued probe is visible, so the completely regular source and its topology remain explicit.

The probe stage is F and R. It displays the continuous real-valued observations and their Real value seals, but it does not supply an ambient function space or a completed realcompact host.

Only after those rows are present can the consumer read B, Z, E. The comparison, zero-set ledger, and envelope therefore remain downstream of displayed real-function data, and the structural rows L, K, P, N preserve that same order. □

Closure status for $\mathsf{NameCert}_{\mathsf{HewittRealcompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{HewittRealcompactification}^{\uparrow}$ packet is generated from CompletelyRegularSpace source rows, Topology rows, ContinuousMap real-valued probes, Real value seals, a StoneCechCompactification comparison row, zero-set ledger rows, an envelope row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HewittRealcompactification}^{\uparrow}}$, seed)

Scope closed: Definition 8494.1, Definition 8494.2, and Definition 8494.3 seed the finite real-valued-function completion boundary for Hewitt realcompactification consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HewittRealcompactification}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close z-ultrafilter choice, host compactification equality, selected points, a full function-ring construction, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, completely regular source exposure, topology exposure, continuous real-valued probe exactness, Real value sealing, Stone–Cech comparison nonescape, zero-set ledger exposure, envelope handoff, transport, replay, provenance, and local naming for BEDC.Derived.HewittRealcompactificationUp.

Chapter 8495

A Concrete Naming Certificate for $\mathsf{CompactOscillationTree}^{\uparrow}$

Following Bishop–Bridges compact uniform-continuity proofs, the $\mathsf{CompactOscillationTree}^{\uparrow}$ packet records the finite cover-refinement tree by which local oscillation windows are folded into a uniform-modulus handoff. It sits after the compact metric source of chapter 129, the continuous-map rows of chapter 130, the finite cover nerve of chapter 7597, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the Real seal of chapter 38, and the uniform-modulus consumer of chapter 369. The packet names only displayed finite tree data; it is not open-cover compactness, an external Lebesgue number selector, a host modulus function, quotient equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8495.1 (Compact oscillation tree carrier)**.** *A* $\mathsf{CompactOscillationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, V, D, Q, R, O, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the finite compact-net cells and coverage witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise inspection row attached to the displayed compact source.*
3. *V is the* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *cover-refinement row, listing finite parent-child cover incidences.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic oscillation scale used at each node.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational lower-bound and oscillation-window values.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed real-valued bounds.*
7. *O is the compact oscillation tree row. It records the finite rooted tree, node windows, child refinements, and leaf oscillation bounds.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported only after K, F, V, D, Q, R, O have been displayed.*
9. *H records componentwise* hsame *transport for source, pointwise map, cover refinement, dyadic tolerance, rational readback, Real seal, oscillation tree, uniform modulus, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a compact-uniform query through $K, F, V, D, Q, R, O, U, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationTree}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationTree}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, F, V, D, Q, R, O, U, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, a selected Lebesgue number, a host modulus function, quotient equality of continuous maps, or ambient completeness.

Theorem 8495.2 (Compact oscillation tree NameCert obligations)**.** *For every accepted* $\mathsf{CompactOscillationTree}^{\uparrow}$ *carrier*

$$T = (K, F, V, D, Q, R, O, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationTree}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through K, continuous-map pointwise inspection through F, finite cover-refinement exposure through V, dyadic tolerance exposure through D, rational readback through Q, Real sealing through R, finite oscillation-tree exposure through O, uniform-modulus handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8495.1. The source rows K, F, V expose the compact finite net, the pointwise continuous-map checks, and the finite cover-refinement incidence data.

Rows D, Q, R attach the numerical readback. They display dyadic tolerances, rational oscillation-window values, and Real value seals before any uniform-modulus consumer can inspect the tree.

The tree row O is therefore read as finite combinatorial data over already displayed cover and tolerance rows. The handoff row U can only consume that finite route, while H, C, P, N transport, replay, source, and name it. □

Theorem 8495.3 (Compact oscillation tree uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationTree}^{\uparrow}$ *carrier first reads the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{FiniteCoverNerve}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactO}$$

Thus the exported modulus is a finite cover-refinement and oscillation-window handoff, not an external compactness theorem, selected Lebesgue number, host modulus function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a compact-uniform query. It must expose the compact-metric row K and pointwise continuous-map row F before it can read any refinement data.

The finite cover route is then V, D, Q, R. These rows display the cover-refinement incidences, dyadic oscillation scales, rational lower-bound reads, and Real seals needed at the tree nodes.

Only after those rows are present can the query read O and then U. Componentwise transport, replay, provenance, and local naming preserve the same finite order, so no hidden compactness or host modulus coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationTree}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationTree}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise checks, FiniteCoverNerve refinement rows, DyadicRatCore tolerances, RegSeqRat rational readback, Real seals, a finite oscillation tree, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactOscillationTree}^{\uparrow}}$, seed)

Scope closed: Definition 8495.1, Definition 8495.2, and Definition 8495.3 seed the finite cover-refinement and oscillation-window route for compact uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactOscillationTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, host modulus functions, quotient equality of continuous maps, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric source exposure, continuous-map pointwise inspection, finite cover-refinement exactness, dyadic tolerance exposure, rational readback, Real sealing, oscillation-tree handoff, UniformModulus export, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationTreeUp

Chapter 8496

A Concrete Naming Certificate for FiniteDimensionalNormedCompleteness$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the FiniteDimensionalNormedCompleteness$^{\uparrow}$ packet names the finite-basis route by which a Cauchy sequence in a finite-dimensional normed space is sealed as Banach data. It sits after the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the coordinate and evaluation maps of chapter 48, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the completion-operator route of chapter 3379, and the Banach-space packet of **??**. The chapter is a finite-dimensional certificate: it records a displayed basis window, coordinate Cauchy extraction, norm-stability rows, completion handoff, and Banach seal, rather than a general completeness theorem for arbitrary normed spaces.

Definition 8496.1 (Finite-dimensional normed completeness carrier)**.** *A* FiniteDimensionalNormedCompleteness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, N, B, K, S, R, E, O, Z, H, C, P, L)$$

with the following displayed rows:

1. *V is the* VecSpace$^{\uparrow}$ *source row for the finite vector-space window.*
2. *N is the* NormedSpace$^{\uparrow}$ *row, exposing the norm and metric reads attached to the same vector rows.*
3. *B is the finite basis ledger, listing the displayed basis names and the coordinate-admission rows for inspected vectors.*
4. *K is the* LinMap$^{\uparrow}$ *coordinate-evaluation row from vectors to finitely many scalar coordinate names.*
5. *S is the coordinate Cauchy schedule, recording that each displayed coordinate stream is Cauchy through finite windows.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the coordinate streams and the norm-control estimates.*
7. *E is the* Real$^{\uparrow}$ *seal row for coordinate limits and norm estimates.*
8. *O is the* CauchyCompletionOperator$^{\uparrow}$ *handoff row, carrying the finite Cauchy evidence into the completion-facing route.*
9. *Z is the* BanachSpace$^{\uparrow}$ *seal row asserting that this finite-basis packet supplies the Banach-facing completeness read.*

10. *H records componentwise* hsame *transport for all rows of the packet.*
11. *C records displayed* Cont *replay through the finite-basis, coordinate, readback, completion, and seal rows.*
12. *P is the* Pkg *provenance over* FiniteDimensionalNormedCompleteness$^{\uparrow}$, VecSpace$^{\uparrow}$, NormedSpace$^{\uparrow}$, LinMap$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyCompletionOperator$^{\uparrow}$, BanachSpace$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *L is the local* NameCert$_{\text{FiniteDimensionalNormedCompleteness}^{\uparrow}}$ *row.*

The classifier accepts only packets whose completeness read is exhausted by $V, N, B, K, S, R, E, O, Z, H, C, P, L$. It has no coordinate for Hahn–Banach extension, arbitrary Banach-space completeness, quotient normed spaces, selected bases, countable choice, or an ambient host completion.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalNormedCompletenessTasteGate single carrier alignment`

Theorem 8496.2 (Finite coordinate Cauchy extraction)**.** *For every accepted* FiniteDimensionalNormedCompletenes *carrier*

$$F = (V, N, B, K, S, R, E, O, Z, H, C, P, L),$$

any Cauchy read of the normed vector sequence factors through the finite basis ledger B and the coordinate-evaluation row K before it reaches the regular rational readback R or the real seal E. Thus the extracted evidence consists of finitely many coordinate Cauchy streams, their RegSeqRat readbacks, the displayed norm-control estimates, and the structural rows H, C, P, L.

Proof. Project the carrier of Definition 8496.1. The only route from the normed source rows V, N to scalar data passes through the finite basis ledger B and the coordinate-evaluation row K.

The coordinate Cauchy schedule S is read after those rows are displayed. It supplies one finite-window Cauchy record for each coordinate named by B, so the extraction is finite rather than a choice over an external basis.

The rows R and E then read the same coordinate streams as regular rational data and terminal real seals. The norm-control estimates remain attached to those scalar rows because H, C, P, L transport, replay, source, and name the displayed packet.

Since the carrier contains no selected basis, quotient normed space, arbitrary Banach-space completeness field, or choice schedule, no downstream consumer can replace the coordinate extraction by a stronger ambient principle. □

Theorem 8496.3 (Finite-dimensional Banach seal)**.** *Every accepted* FiniteDimensionalNormedCompleteness$^{\uparrow}$ *carrier supplies a Banach-facing completeness seal only through the displayed route*

$$V, N \longmapsto B, K, S \longmapsto R, E \longmapsto O \longmapsto Z \longmapsto H, C, P, L.$$

The row O hands finite coordinate Cauchy evidence to the completion-operator surface, and the row Z records the resulting BanachSpace$^{\uparrow}$ *seal for this finite-dimensional packet. No Hahn–Banach principle, quotient normed space, arbitrary Banach completeness theorem, selected basis operation, countable-choice schedule, or host completion endpoint is exported.*

Proof. Begin with Definition 8496.2. It forces the Cauchy read through the finite coordinate surface B, K, S before any scalar readback or real seal is available.

The norm-stability step uses the normed-space row N together with the scalar rows R, E. The displayed estimates say that convergence of the finitely many coordinate streams is enough for the

normed Cauchy route seen by this packet; the proof does not require a supremum over an unlisted basis family.

The completion handoff is the row O. It consumes exactly the finite Cauchy evidence already displayed, matching the operator discipline of chapter 3379.

Finally Z records the Banach-facing seal for this packet, while H, C, P, L keep the route attached under transport, replay, provenance, and local naming. The excluded analytic principles are absent from the carrier inventory, so the seal cannot export them. □

Falsifiable prediction: A purported finite-dimensional completeness read that reaches a Banach seal without displaying the finite basis ledger, coordinate Cauchy schedule, RegSeqRat readback, Real seal, and completion-operator handoff refutes this carrier surface.

Independence witness: normedspace, vecspace, linearmap, regseqrat, real, cauchycompletionoperator, banachspace: no listed sibling alone binds finite basis rows, coordinate Cauchy extraction, norm stability, completion handoff, and a Banach-facing seal into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormedCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalNormedCompleteness}^{\uparrow}$ packet is generated from vector-space, normed-space, finite-basis, coordinate-evaluation, coordinate Cauchy, RegSeqRat readback, Real seal, CauchyCompletionOperator handoff, Banach seal, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormedCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8496.1, Definition 8496.2, and Definition 8496.3 seed the finite-basis completeness route from normed vector rows through coordinate Cauchy extraction and Banach sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormedCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FiniteDimensionalNormedCompletenessUp`

Bridge status: none

Not claimed: This chapter does not close Hahn–Banach extension, arbitrary Banach-space completeness, quotient normed spaces, selected bases, countable choice, host completion endpoints, theoremChecked Lean status, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coordinate Cauchy extraction, norm-stability readback, completion-operator handoff, Banach sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDimensionalNormedCompletenessUp.

Chapter 8497

A Concrete Naming Certificate for CauchyCompletionErrorEstimator$^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion and explicit error bounds, the CauchyCompletionErrorEstimator$^{\uparrow}$ packet records the finite route by which a completion read carries an a posteriori dyadic error estimate. It sits beside the completion operator of chapter 3379, the dyadic tolerance core of chapter 291, the regular rational source rows of chapter 301, the finite stream-name windows of chapter 126, and the terminal real-name seal of chapter 38. The packet names error control as finite NameCert data; it does not choose a limit representative, import ambient completeness, assert host real equality, or hide the estimate behind quotient completion syntax.

Definition 8497.1 (Cauchy completion error estimator carrier)**.** *A* CauchyCompletionErrorEstimator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (O, S, W, D, R, A, Z, H, C, P, N)$$

with the following displayed rows:

1. *O is the* CauchyCompletionOperator$^{\uparrow}$ *row whose completion output is being inspected.*
2. *S is the* RegSeqRat$^{\uparrow}$ *source row naming the regular rational approximation stream.*
3. *W is the* StreamName$^{\uparrow}$ *finite-window row through which the source tail is observed.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to record the displayed error radius.*
5. *R is the* Real$^{\uparrow}$ *seal row for the completed value reached from the source data.*
6. *A is the a posteriori estimate row, read only after O, S, W, D, R have been displayed.*
7. *Z is the zero-tail or residual-bound row certifying that the displayed estimate bounds the remaining completion tail.*
8. *H records componentwise* hsame *transport for the operator, source, window, dyadic ledger, real seal, estimate row, residual-bound row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the same finite route from an estimator request.*
10. *P records* Pkg *provenance over* CauchyCompletionErrorEstimator$^{\uparrow}$*,* CauchyCompletionOperator$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, S, W, D, R, A, Z, H, C, P, N$. It has no coordinate for selected limits, quotient completion equality, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8497.2 (Cauchy completion error estimator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}$ *carrier*

$$E = (O, S, W, D, R, A, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}}$ *surface consists exactly of completion-operator exposure for O, regular-rational source exposure for S, finite-window exposure for W, dyadic tolerance exposure for D, Real-seal exposure for R, a posteriori estimate exposure for A, residual-bound exposure for Z, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The operator, source, finite window, dyadic ledger, and Real seal are the only completion-facing rows available before an estimate can be read.

Next read A and Z. The estimate row supplies the displayed dyadic error radius, while the residual-bound row ties that radius to the remaining completion tail observed through the same source and window rows.

Finally H, C, P, N preserve the route under componentwise transport, replay it for estimator consumers, record provenance, and name the packet locally. Since the carrier contains no coordinate for selected limits, quotient equality, host real equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8497.3 (Cauchy completion error estimator a posteriori bound)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}$ *carrier reaches an a posteriori error bound only through the finite route*

$$O, S, W, D, R, A, Z, H, C, P, N.$$

The dyadic ledger D and finite window W determine the displayed bound, the residual row Z certifies it against the source tail S, and the Real seal R receives the estimate only after those rows are visible. Thus the estimator exports a finite completion-error certificate rather than a selected completion limit, quotient representative equality, host real equality, countable-choice schedule, trichotomy split, or ambient completeness theorem.

Proof. Start with the carrier projection. The completion operator row O and regular rational source row S determine which completion read is being estimated, while W fixes the finite observation window.

Use the dyadic ledger row D. It supplies the displayed radius for the estimate, but it is still tied to the source and window rows rather than to an external real-valued oracle.

Now read A and Z after the Real seal row R is visible. The estimate row records the a posteriori bound, and the residual-bound row certifies that the remaining completion tail is covered by that dyadic radius.

The support rows H, C, P, N only transport, replay, source, and name this same route. Because the excluded ambient principles are not carrier coordinates, the exported bound remains a finite completion-error certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}$ packet is generated from a CauchyCompletionOperatorUp row, a RegSeqRatUp source row, a StreamNameUp finite-window row, a DyadicRatCoreUp tolerance ledger, a RealUp seal row, an a posteriori estimate row, a residual-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8497.1, Definition 8497.2, and Definition 8497.3 seed the finite completion-error estimator packet over completion-operator, regular-rational, stream-window, dyadic-ledger, Real-seal, residual-bound, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionErrorEstimator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected completion limits, quotient representative equality, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires Lean encoding of the carrier, NameCert obligation surface, a posteriori bound route, transport stability, replay stability, and nonescape for BEDC.Derived.CauchyCompletionErrorEstimatorUp.

Chapter 8498

A Concrete Naming Certificate for $\mathsf{CompactMetricRegularSequence}^{\uparrow}$

Following Bishop and Bridges on compact metric spaces, finite nets, and regular Cauchy real names, the $\mathsf{CompactMetricRegularSequence}^{\uparrow}$ packet records the finite route by which a compact metric source supplies regular sequence data before a completion-facing consumer reads a real-name seal. It sits over the compact metric source of chapter 129, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. It is the base compact-metric regular-sequence packet consumed by the compact regular Cauchy subsequence route of chapter 3913; it does not assert open-cover compactness, countable choice, a host-selected limit, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8498.1 (Compact metric regular sequence carrier)**.** *A* $\mathsf{CompactMetricRegularSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying finite-net, located-distance, and compact inspection data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the source sequence windows inspected by the compact metric consumer.*
3. *W is the finite window ledger pairing requested precisions with inspected compact-net cells and source sequence windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for compact mesh radii, window distances, and regular Cauchy budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed windows and dyadic budgets.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the compact source, finite windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for K, S, W, D, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from compact inspection through finite windows, dyadic tolerance accounting, regular readback, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricRegularSequence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricRegularSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, S, W, D, R, E, H, C, P, N$. It has no coordinate for a host-selected subsequence, open-cover compactness theorem, quotient stream representative, hidden real limit, countable choice, or ambient completion principle.

Theorem 8498.2 (Compact metric regular sequence NameCert obligations)**.** *For every accepted* $\mathsf{CompactMetricRegularSequence}^{\uparrow}$ *carrier $M = (K, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricRegularSe}}$ *surface consists exactly of compact-metric source exposure through K, source-window exposure through S, finite-window ledger exposure through W, dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8498.1. The finite packet exposes exactly the compact metric source, source stream windows, inspected finite-window ledger, dyadic tolerance ledger, regular rational readback, Real seal, and structural rows.

Read the analytic rows in order. The row K supplies only displayed compact metric inspection data, while S and W record the finite source windows and compact-net cells inspected at each requested precision.

The handoff is then forced through D, R, E. Dyadic budgets are recorded in D, regular rational readback is recorded in R, and the Real seal E is reached only after those rows are visible. Finally H, C, P, N transport, replay, source, and name the same finite packet, so no additional NameCert obligation is available outside the displayed rows. □

Theorem 8498.3 (Compact metric regular sequence finite-net handoff)**.** *Every* $\mathsf{CompactMetricRegularSequence}^{\uparrow}$ *consumer that requests regular sequence data must first read the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The compact metric source supplies finite-net cells, the stream row supplies inspected windows, the dyadic ledger supplies mesh and tail budgets, and the regular rational row supplies the displayed readback before any $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Begin with an accepted carrier $M = (K, S, W, D, R, E, H, C, P, N)$. The compact source row K is the only compact metric coordinate, and the source observations are read through S and the finite ledger W.

Use the finite-net data in K together with the displayed windows in W. For a requested precision, the consumer receives a compact-net cell and the source observations inspected in that cell rather than an unrecorded compactness theorem.

The dyadic and regular rows complete the handoff. The ledger D records the mesh radius and tail budget, and R repacks the inspected windows as $\mathsf{RegSeqRat}^{\uparrow}$ readback data. Transport and replay through H, C, P, N preserve this route, so the consumer reaches E only after the finite-net handoff has been displayed. □

Theorem 8498.4 (Compact metric regular sequence Cauchy readback)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactMetricRegularSequence}^{\uparrow}$ *carrier reads the Cauchy evidence through the displayed sequence*

$$K, S, W, D, R, E, H, C, P, N.$$

In particular, CompactMetricRegularSequence$^{\uparrow}$ *can feed the compact regular Cauchy subsequence packet of chapter 3913 only by exporting finite windows, dyadic budgets,* RegSeqRat$^{\uparrow}$ *readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming; it does not export a host-selected limit or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start from the finite-net handoff theorem, Definition 8498.3. It confines regular sequence data to the compact source, finite stream windows, dyadic budget ledger, and regular rational readback.

Apply the NameCert obligation theorem, Definition 8498.2. The obligation surface keeps $K, S, W, D, R, E, H, C, P, N$ jointly visible, so downstream completion consumers cannot discard the compact-net or dyadic-budget rows while retaining the readback.

Finally compare the downstream compact regular Cauchy subsequence route of Definition 3913.3. That packet may consume the displayed regular readback and Real seal boundary, but this carrier itself supplies only the finite-window Cauchy evidence named in the theorem. Host-selected limits, ambient completeness, quotient equality, and countable choice remain outside the exported surface. □

Closure status for NameCert$_{\text{CompactMetricRegularSequence}^{\uparrow}}$.

Constructive story: A CompactMetricRegularSequence$^{\uparrow}$ packet is generated from a CompactMetric finite-net source row, StreamName finite windows, a finite window ledger, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactMetricRegularSequence}^{\uparrow}}$, seed)

Scope closed: Definition 8498.1, Definition 8498.2, Definition 8498.3, and Definition 8498.4 seed the compact-metric regular sequence route over CompactMetric, StreamName, DyadicRatCore, RegSeqRat, and RealUp rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactMetricRegularSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient stream equality, host sequence equality, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, source-window exposure, dyadic tolerance accounting, RegSeqRat readback exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactMetricRegularSequenceUp.

Chapter 8499

A Concrete Naming Certificate for $\mathsf{DunfordPettisOperator}^{\uparrow}$

Following Dunford–Schwartz and Schaefer on completely continuous operators between Banach spaces, the $\mathsf{DunfordPettisOperator}^{\uparrow}$ packet records the finite BEDC operator carrier that sends weakly compact or weakly null inspection rows into compact-image and completely continuous handoff rows. It sits over the bounded-operator carrier of chapter 1644, the Banach endpoint certificates of chapter 90, the finite weak-test surface of chapter 3597, the weak-compactness request of chapter 6851, the compact-operator consumer of chapter 1952, and the Dunford–Pettis handoff row of chapter 6346. The chapter names an operator carrier; it does not import a host theorem asserting that every weakly compact row is compactly imaged, nor does it select weakly convergent subsequences or quotient operator equality.

Definition 8499.1 (Dunford–Pettis operator carrier)**.** *A* $\mathsf{DunfordPettisOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, A, W, K, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *row, naming the normed linear source and its displayed completeness data.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, naming the codomain Banach carrier read by compact-image consumers.*
3. *A is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *action row, carrying the bounded linear map from X to Y only as displayed operator data.*
4. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *finite-test row, listing the weak probes and tolerances through which bounded or weakly null input families are inspected.*
5. *K is the* $\mathsf{WeakCompactness}^{\uparrow}$ *request row, read as a compactness-facing demand on inspected input windows rather than as a completed weak-compactness theorem.*
6. *Q is the compact-image or completely-continuous handoff row consumed by* $\mathsf{CompactOperator}^{\uparrow}$ *and* $\mathsf{DunfordPettis}^{\uparrow}$ *readers.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for operator norms, weak probe values, compact-image radii, and tolerance ledgers.*

8. *H records componentwise* hsame *transport for the Banach endpoints, bounded operator action, weak tests, weak-compactness request, compact-image handoff, scalar readback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through the same operator packet for weak-compactness, compact-operator, and Dunford–Pettis consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{DunfordPettisOperator}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{DunfordPettis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DunfordPettisOperator}^{\uparrow}}$ *row.*

The classifier accepts only packets exhausted by $X, Y, A, W, K, Q, R, H, C, P, N$. It has no coordinate for a host compactness theorem, a selected weakly convergent subsequence, quotient equality of operators, Banach–Alaoglu, countable choice, or ambient dual-space identification.

Lean4 (statement) `BEDC.Derived.DunfordPettisOperatorUp`

Theorem 8499.2 (Dunford–Pettis operator NameCert obligations)**.** *For the carrier of Definition 8499.1, the local* $\mathsf{NameCert}_{\mathsf{DunfordPettisOperator}^{\uparrow}}$ *surface consists exactly of source and target Banach admission, bounded-linear-operator action admission, finite weak-test visibility, weak-compactness request visibility, compact-image or completely-continuous handoff exactness, Real scalar readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the packet $T = (X, Y, A, W, K, Q, R, H, C, P, N)$. The Banach endpoint rows X, Y and the bounded-linear-operator row A supply the operator carrier before any compactness-facing consumer can read it.

The weak side is also displayed. The finite weak-test row W and the weak-compactness request row K are read before the handoff row Q, so the compact-image or completely-continuous conclusion remains attached to visible weak data.

The scalar row R supplies only the listed operator-norm, probe, radius, and tolerance readbacks. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since no coordinate stores a host compactness theorem, selected subsequence, quotient operator equality, Banach–Alaoglu principle, countable-choice schedule, or dual-space identification, the NameCert obligations are exactly the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordPettisOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{DunfordPettisOperator}^{\uparrow}$ packet is generated from source and target Banach rows, a bounded-linear-operator action, weak-topology finite tests, a weak-compactness request, a compact-image or completely-continuous handoff, Real scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DunfordPettisOperator}^{\uparrow}}$, seed)

Scope closed: Definition 8499.1 and Definition 8499.2 seed the finite operator carrier linking Banach endpoints, bounded linear action, weak tests, weak-compactness requests, and compact-image handoff rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DunfordPettisOperator}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DunfordPettisOperatorUp`

Bridge status: none

Not claimed: This chapter does not close a host Dunford–Pettis theorem, selected weakly convergent subsequences, quotient equality of operators, Banach–Alaoglu, countable choice, ambient dual-space identification, theoremChecked Lean status, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Banach endpoint compatibility, bounded operator action, finite weak-test visibility, weak-compactness request visibility, compact-image handoff exactness, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.DunfordPettisOperatorUp.

Chapter 8500

A Concrete Naming Certificate for $\mathsf{DaugavetProperty}^{\uparrow}$

Following Kadets and Shvidkoy on the Daugavet property in Banach-space geometry, the $\mathsf{DaugavetProperty}^{\uparrow}$ packet records the finite row surface by which a Banach carrier exposes the rank-one norm equation to later operator and geometry consumers. It sits over the norm carrier of chapter 89, the Banach rows of chapter 90, the scalar-functional surface of chapter 5790, the linear-map carrier of chapter 48, and the James-space comparison packet of chapter 6468. The packet names a displayed rank-one equation ledger; it is not a host class of all rank-one operators, not a selected norming functional, not a quotient operator algebra, and not an ambient compactness or reflexivity theorem.

Definition 8500.1 (Daugavet property carrier)**.** *A* $\mathsf{DaugavetProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, N, F, L, R, J, E, H, C, P, A)$$

with the following displayed rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row whose vector and completeness reads are visible to the packet.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row used to compare the identity-plus-operator norm with the operator norm.*
3. *F is the* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row naming the scalar functional in the displayed rank-one factorization.*
4. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the vector action used by the same rank-one factorization.*
5. *R is the rank-one operator ledger, recording the finite factorization data read through F and L.*
6. *J is the* $\mathsf{JamesSpace}^{\uparrow}$ *comparison row, present only as a Banach-geometry sibling and not as a reflexivity theorem.*
7. *E is the equation row recording that the displayed rank-one read is judged against the norm equation $\|I + T\| = 1 + \|T\|$.*
8. *H records componentwise* hsame *transport for the Banach, norm, functional, linear-map, rank-one, James-space, equation, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{DaugavetProperty}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{JamesSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. A is the local $\mathsf{NameCert}_{\mathsf{DaugavetProperty}^{\uparrow}}$ *row.*

The classifier accepts only packets whose Daugavet read is exhausted by $B, N, F, L, R, J, E, H, C, P, A$*. It has no coordinate for all bounded operators on the host space, a selected norming vector, a quotient operator class, weak compactness, reflexivity, or an ambient Banach-space theorem outside the displayed rows.*

Theorem 8500.2 (Daugavet property rank-one equation)**.** *For every accepted* $\mathsf{DaugavetProperty}^{\uparrow}$ *carrier*

$$D = (B, N, F, L, R, J, E, H, C, P, A),$$

any consumer read of the rank-one equation must first expose the Banach source row B*, the norm row* N*, the bounded-functional row* F*, the linear-map row* L*, and the rank-one ledger* R*. Only after those rows are visible may the equation row* E *be read as the finite statement that the displayed rank-one operator* T *satisfies* $\|I+T\| = 1+\|T\|$ *for the named Banach row. The* $\mathsf{JamesSpace}^{\uparrow}$ *row* J *remains a comparison surface, and the structural rows* H, C, P, A *transport, replay, source, and name the same packet.*

Proof. Project the carrier of Definition 8500.1. Rows B and N expose the Banach source and norm comparison data before any operator equation is available.

The rank-one stage is F, L, R. The bounded functional and linear-map rows name the displayed factorization, and the ledger R records that the operator read is finite and local to those rows.

Only then can the equation row E be consumed. It records the equality for the displayed rank-one operator and the displayed Banach norm, while J stays a sibling comparison row rather than a source of reflexivity or weak compactness.

Finally H, C, P, A keep the same finite route under transport, replay, provenance, and local naming. Since the carrier has no coordinate for a host operator space, selected norming data, quotient equality, or ambient Banach theorem, the consumer cannot export any of those endpoints. □

Closure status for $\mathsf{NameCert}_{\mathsf{DaugavetProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DaugavetProperty}^{\uparrow}$ packet is generated from Banach source rows, Norm rows, bounded linear functional rows, LinearMap rows, rank-one operator ledgers, JamesSpace comparison rows, an equation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DaugavetProperty}^{\uparrow}}$, seed)

Scope closed: Definition 8500.1 and Definition 8500.2 seed the finite rank-one norm-equation route over displayed Banach, Norm, BoundedLinearFunctional, LinearMap, JamesSpace, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DaugavetProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host space of all bounded operators, selected norming vectors, quotient operator equality, weak compactness, reflexivity, Lean verification, bridge checking, or an ambient Banach-space theorem.

Upgrade path: Obligation closure requires checked carrier admission, Banach source exposure, norm comparison exactness, bounded-functional and linear-map factorization, rank-one ledger stability, James-space comparison nonescape, equation readback, transport, replay, provenance, and local naming for BEDC.Derived.DaugavetPropertyUp.

Chapter 8501

A Concrete Naming Certificate for $\mathsf{DyadicRoundingInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real approximation by rational intervals, the $\mathsf{DyadicRoundingInterval}^{\uparrow}$ packet records the finite lower–upper dyadic enclosure obtained from one displayed rounding request. It sits over the dyadic arithmetic source of chapter 291, the lower-grid route of chapter 1416, the upper-grid route of chapter 1690, the signed normal-form boundary of chapter 1865, the finite observation window of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names an interval produced by finite rounding rows, not a host order decision, quotient real equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8501.1 (Dyadic rounding interval carrier)**.** *A* $\mathsf{DyadicRoundingInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, Q, F, U, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row carrying the dyadic arithmetic ledger and requested scale.*
2. *Q is the rounding request row naming the inspected value, precision, and outward-rounding direction as finite data.*
3. *F is the lower endpoint row read through the dyadic floor route of chapter 1416.*
4. *U is the upper endpoint row read through the dyadic ceiling route of chapter 1690.*
5. *S is the* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *row that keeps both endpoints in the same signed normal-form convention.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the request is observed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row served by the same window and endpoint pair.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, Q, F, U, S, W, R are displayed.*
9. *H records componentwise* hsame *transport for the dyadic source, request, endpoints, signed normal form, window, readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same finite route from a rounding-interval request.*

11. *P records* Pkg *provenance over* $\mathsf{DyadicRoundingInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *the floor and ceiling endpoint routes,* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRoundingInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, Q, F, U, S, W, R, E, H, C, P, N$. It has no coordinate for host trichotomy, quotient real equality, selected representatives, a global rounding oracle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8501.2 (Dyadic rounding interval NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRoundingInterval}^{\uparrow}$ *carrier*

$$I = (D, Q, F, U, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRoundingInterval}^{\uparrow}}$ *surface consists exactly of DyadicRatCore source exposure through D, rounding-request exposure through Q, lower endpoint exposure through F, upper endpoint exposure through U, signed normal-form exposure through S, finite-window exposure through W, RegSeqRat readback through R, Real seal exposure through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The dyadic source row D and rounding request row Q determine the displayed scale and the finite request; no host comparison service is available outside those rows.

Next read the two endpoint rows. The lower endpoint F is supplied by the floor route, the upper endpoint U by the ceiling route, and S forces both endpoints to share the signed normal-form convention before any stream or regular readback is consumed.

Finally W, R, E, H, C, P, N expose the observation window, regular rational readback, terminal Real seal, transport, replay, provenance, and local name. Since the carrier contains no coordinate for trichotomy, quotient equality, selected representatives, global rounding, choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8501.3 (Dyadic rounding interval containment route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicRoundingInterval}^{\uparrow}$ *carrier must read the displayed route*

$$D, Q, F, U, S, W, R, E, H, C, P, N.$$

The consumer receives the lower endpoint F and upper endpoint U as one signed dyadic interval before reading R or E, and the interval is transported only componentwise through H and replayed only through C.

Proof. Begin with the carrier projection. The source and request rows D, Q identify one finite rounding query, so the endpoint rows are not independent witnesses from separate searches.

Read F, U, S next. The lower and upper endpoint rows form the displayed dyadic interval, and the signed normal-form row keeps them in the same endpoint convention before the observation window is served.

The regular and Real-facing rows are downstream. The StreamName window W and RegSeqRat readback R must be exposed before the terminal Real seal E can consume the interval, while H, C, P, N preserve only that same route. Hence containment is a finite interval route rather than an ambient order fact. □

Theorem 8501.4 (Dyadic rounding interval nonescape)**.** *No accepted* $\mathsf{DyadicRoundingInterval}^{\uparrow}$ *carrier exports a host rounding operator, a trichotomy decision for arbitrary real names, quotient*

equality of real representatives, a selected endpoint representative, countable-choice search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer that reaches the interval must remain inside the rows* $D, Q, F, U, S, W, R, E, H, C, P, N$.

Proof. Inspect the carrier coordinates. They contain dyadic source data, one request, two endpoint rows, signed normal form, window and readback rows, a Real seal, and structural rows for transport, replay, provenance, and naming.

None of the excluded principles is a coordinate of the carrier or a consequence of a structural row. Componentwise hsame transport preserves the displayed endpoint rows, Cont replay replays the same finite route, and Pkg provenance records only the named dependencies.

Therefore every interval-facing read stays inside $D, Q, F, U, S, W, R, E, H, C, P, N$. A consumer claiming host rounding, trichotomy, quotient equality, selected representatives, choice, or ambient completeness has left the accepted NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRoundingInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRoundingInterval}^{\uparrow}$ packet is generated from a DyadicRatCore source row, a rounding request, floor and ceiling endpoint rows, signed dyadic normal-form data, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRoundingInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8501.1, Definition 8501.2, Definition 8501.3, and Definition 8501.4 seed the finite rounding-interval route over dyadic source, rounding request, lower endpoint, upper endpoint, signed normal form, finite window, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRoundingInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host rounding, trichotomy for arbitrary real names, quotient real equality, selected endpoint representatives, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires Lean encoding of the carrier, NameCert obligation surface, containment route, nonescape theorem, transport stability, replay stability, and provenance rows for BEDC.Derived.DyadicRoundingIntervalUp.

Chapter 8502

A Concrete Naming Certificate for $\mathsf{ClosedGraphCertificate}^{\uparrow}$

Following Banach's closed graph theorem tradition and the Bishop–Bridges reading of metric completeness, the $\mathsf{ClosedGraphCertificate}^{\uparrow}$ packet records graph-closedness as finite BEDC carrier data before it is consumed by the theorem-level route of chapter 1537. It sits over the metric observation surface of chapter 57, the complete-metric boundary of chapter 128, the metric-completion interface of chapter 1506, the Banach source row of chapter 90, and the finite StreamName, RegSeqRat, and Real readback route of chapter 126, chapter 301, and chapter 38. The certificate names the displayed graph-closedness surface itself, not the surrounding boundedness conclusion, open mapping theorem, bounded inverse theorem, quotient function space, or host-level graph equality.

Definition 8502.1 (Closed graph certificate carrier)**.** *A* $\mathsf{ClosedGraphCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, G, Q, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the source metric or Banach row, read through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*, or* $\mathsf{Banach}^{\uparrow}$ *observations.*
2. *Y is the target metric or Banach row, read through the same displayed complete-metric vocabulary.*
3. *F is the finite function-action row, containing the inspected source addresses and their displayed target addresses.*
4. *G is the graph-membership row, recording the finite source–target pairs accepted as graph data for F.*
5. *Q is the graph-closedness witness row, recording the sequential test: a scheduled source route converges in X, its graph targets converge in Y, graph membership persists along the schedule, and the target limit is accepted as the graph value for the source limit.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite source, target, graph, and convergence windows used by the test.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational metric and convergence estimates on the scheduled windows.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for distance and bound reads consumed by metric or completion-facing consumers.*
9. *H records componentwise* hsame *transport for source, target, function action, graph membership, graph-closedness witness, schedule, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from metric and completion admission through graph inspection, sequential closure, and real-seal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{ClosedGraphCertificate}^{\uparrow}$, $\mathsf{ClosedGraphTheorem}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a boundedness theorem, an open-mapping theorem, a bounded-inverse theorem, host graph equality, selected limits, quotient function spaces, or ambient completeness beyond the displayed rows X and Y.

Theorem 8502.2 (Closed graph certificate NameCert obligations)**.** *For an accepted* $\mathsf{ClosedGraphCertificate}^{\uparrow}$ *carrier $K = (X, Y, F, G, Q, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission, source and target metric readback, function-action exposure, graph-membership exactness, sequential graph-closedness, finite schedule and rational readback, real-seal non-escape, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read through the finite route*

$$X, Y, F, G, Q, S, R, E, H, C, P, N.$$

Proof. Project the carrier of Definition 8502.1. The displayed rows are exactly $X, Y, F, G, Q, S, R, E, H, C, P, N$, so carrier admission and local naming are immediate readings of the packet.

Read the analytic rows first. The source and target rows X, Y supply metric or Banach observations, F supplies the inspected function action, and G supplies the graph-membership pairs accepted for that action.

The graph-closedness row Q then supplies the sequential test. It can be inspected only along the windows selected by S, the rational estimates read by R, and the real-facing seals in E.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further coordinate remains, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}^{\uparrow}}$ surface and no theorem-level boundedness conclusion is exported by the certificate alone. □

Theorem 8502.3 (Closed graph certificate sequential route)**.** *Every completion-facing or theorem-facing consumer of an accepted* $\mathsf{ClosedGraphCertificate}^{\uparrow}$ *carrier*

$$K = (X, Y, F, G, Q, S, R, E, H, C, P, N)$$

must read graph-closedness through the sequential route

$$X, Y, F, G, S, R, E, Q, H, C, P, N.$$

The route may feed the $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *packet of chapter 1537, but it does not by itself assert local boundedness, continuity, a host graph equality, a quotient function-space equality, selected limits, or ambient completeness.*

Proof. Begin with the graph rows. The consumer sees the function-action row F and graph-membership row G only as displayed finite source–target observations.

Next read the sequential closure data. The schedule S selects the source and target windows, the readback row R supplies rational metric estimates, and the seal row E carries the metric values into a Real-facing read. The row Q accepts the limit graph value only along that same scheduled route.

Use the closed-graph theorem sibling only after this route is visible. chapter 1537 may consume the certificate as its graph-closedness input, while the certificate remains the smaller carrier-level surface named by Definition 8502.1.

Transport, replay, provenance, and local naming are then supplied by H, C, P, N. None of those structural rows adds a boundedness theorem, a quotient function-space object, or a selected host limit, so the sequential graph-closedness route is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8502.1, Definition 8502.2, and Definition 8502.3 seed the finite graph-closedness certificate over displayed metric, completion, function-action, graph-membership, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedGraphCertificate}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, local boundedness, continuity, open mapping, bounded inverse, quotient function-space equality, selected limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, graph-membership exactness, sequential graph-closedness, finite schedule readback, real-seal non-escape, classifier stability, and theorem-facing handoff for the same finite certificate.

Constructive story: The construction starts from displayed source and target metric rows, records finite graph membership, tests graph closure through scheduled sequential windows, and exports only the local certificate consumed by the closed-graph theorem route.

Chapter 8503

A Concrete Naming Certificate for $\mathsf{CatalanNumber}^{\uparrow}$

Following Stanley's enumerative combinatorics and Feller's Dyck-path ballot analysis, the $\mathsf{CatalanNumber}^{\uparrow}$ packet records Catalan data as finite prefix rows rather than as a completed counting theorem. It sits over the unary index carrier of chapter 28, the finite spine carrier of chapter 36, and the binary-tree induction surface of chapter 5910. The packet names one displayed Catalan object at a fixed size: a Dyck-prefix ledger, a matched binary-tree row, a finite count row, transport, replay, provenance, and local naming. It does not assert analytic generating-function identities, quotient equality of trees, or a host enumeration principle.

Definition 8503.1 (Catalan number carrier)**.** *A* $\mathsf{CatalanNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (n, W, P, T, C, H, R, G, N)$$

with the following displayed rows:

1. *n is the* $\mathsf{Nat}^{\uparrow}$ *size row.*
2. *W is the finite* $\mathsf{List}^{\uparrow}$ *word row of left and right steps.*
3. *P is the Dyck-prefix ledger: every prefix of W has nonnegative balance, and the terminal balance is zero.*
4. *T is the* $\mathsf{BinaryTree}^{\uparrow}$ *row obtained by matching each left step with its closing right step.*
5. *C is the finite* $\mathsf{Nat}^{\uparrow}$ *count row naming the Catalan value attached to the displayed size row.*
6. *H records componentwise* hsame *transport for the size, word, prefix ledger, tree row, count row, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from a Dyck-prefix request to the tree row and count row.*
8. *G is the* Pkg *provenance over* $\mathsf{CatalanNumber}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{BinaryTree}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CatalanNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for infinite paths, unbounded search, quotient tree equality, analytic generating functions, or an external cardinality theorem.

Theorem 8503.2 (Catalan number NameCert obligations)**.** *For every accepted* $\mathsf{CatalanNumber}^{\uparrow}$ *carrier* $K = (n, W, P, T, C, H, R, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CatalanNumber}^{\uparrow}}$ *surface consists exactly of size admission, finite word admission, Dyck-prefix balance, binary-tree matching, Catalan count naming, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from analytic or quotient enumeration principles.*

Proof. Project the carrier. The size row n is read through $\mathsf{Nat}^{\uparrow}$, while W is a finite $\mathsf{List}^{\uparrow}$ word row whose positions are the only inspected steps.

The Dyck-prefix ledger P supplies the balance checks. Since every prefix row is displayed and the terminal balance is zero, the packet never appeals to an infinite path or to unbounded search.

The tree row T is obtained only after the prefix ledger has been read. Matching a left step with its closing right step is therefore a finite operation over the displayed word, and the count row C names the resulting Catalan value at the same size row.

The structural rows H, R, G, N transport, replay, source, and name this finite route. Because the carrier has no coordinate for quotient tree equality, generating functions, or an external enumeration theorem, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 8503.3 (Catalan Dyck-prefix induction route)**.** *Every accepted* $\mathsf{CatalanNumber}^{\uparrow}$ *packet admits the finite decomposition route*

$$W = L\,U\,R\,V$$

where the first left step L *is matched by the first return right step* R*, the interior word* U *and suffix word* V *are again displayed Dyck-prefix rows, and the associated* $\mathsf{BinaryTree}^{\uparrow}$ *read decomposes into the root node with left subtree from* U *and right subtree from* V*. The route is read only from the* $\mathsf{Nat}^{\uparrow}$ *size row, the* $\mathsf{List}^{\uparrow}$ *word row, the Dyck-prefix ledger, and the* $\mathsf{BinaryTree}^{\uparrow}$ *matching row.*

Proof. Begin with an accepted packet $K = (n, W, P, T, C, H, R, G, N)$. If W is empty, the terminal balance row in P gives the empty Dyck word and the binary-tree row T is the empty tree boundary.

For a nonempty accepted word, the first step must be a left step. A right step would make the first prefix balance negative, contradicting the displayed prefix ledger P.

Read along the finite $\mathsf{List}^{\uparrow}$ spine until the first return to balance zero. The ledger P displays each inspected balance, so the chosen return is a finite row inside the packet. The part between the first left step and this return has nonnegative shifted prefix balance, and the suffix after the return starts again at balance zero.

The matching row T reads the same decomposition as a binary tree. The interior word gives the left subtree, the suffix gives the right subtree, and the root is the displayed first matched pair. Transport by H and replay by R preserve the same finite decomposition for any transported packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CatalanNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{CatalanNumber}^{\uparrow}$ packet is generated from a NatUp size row, a finite ListUp Dyck word row, a displayed prefix-balance ledger, a BinaryTreeUp matching row, a NatUp count row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CatalanNumber}^{\uparrow}}$, seed)

Scope closed: Definition 8503.1, Definition 8503.2, and Definition 8503.3 seed the finite Dyck-prefix and binary-tree route for Catalan number packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CatalanNumber}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Catalan recurrence uniqueness, generating-function identities, quotient tree equality, host enumeration theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Dyck-prefix balance, binary-tree matching, count-row naming, transport, replay, provenance, and local naming for BEDC.Derived.CatalanNumberUp.

Chapter 8504

A Concrete Naming Certificate for $\mathsf{FiniteModulusMesh}^{\uparrow}$

The $\mathsf{FiniteModulusMesh}^{\uparrow}$ packet realizes the compact-uniform-continuity route of chapter 9237 as finite mesh data. It consumes compact-net cells from chapter 129, continuous-map pointwise modulus rows from chapter 130, dyadic tolerance rows from chapter 291, Real readback from chapter 38, and the terminal uniform-modulus handoff of chapter 369. The packet names the mesh itself: compact cells, pointwise-radius witnesses, dyadic tolerances, a finite lower-bound fold, a triangle-estimate ledger, a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ output, transport, replay, provenance, and local naming.

Definition 8504.1 (Finite modulus mesh carrier)**.** *A* $\mathsf{FiniteModulusMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, D, Q, L, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-net cell row over the compact metric source.*
2. *F is the pointwise modulus row attached to the continuous-map data at the displayed cell centers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row read by the mesh.*
4. *Q is the finite mesh incidence ledger pairing each compact-net cell with its local modulus and dyadic tolerance request.*
5. *L is the finite lower-bound fold over the displayed radius rows.*
6. *T is the triangle-estimate ledger that keeps source and target metric comparisons visible before the output row is exported.*
7. *U is the tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row.*
8. *H records componentwise* hsame *transport for compact cells, pointwise moduli, dyadic tolerances, incidence, lower-bound fold, triangle ledger, output, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through the same mesh route.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusMesh}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, D, Q, L, T, U, H, C, P, N$. It has no coordinate for a selected subcover outside the packet, a host Lebesgue number, quotient metric equality, countable choice, or ambient compactness.

Theorem 8504.2 (Finite modulus mesh NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusMesh}^{\uparrow}$ *carrier* $M = (K, F, D, Q, L, T, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusMesh}^{\uparrow}}$ *surface consists exactly of compact-net cell admission, pointwise-modulus visibility, dyadic tolerance admission, finite incidence coverage, lower-bound fold exactness, triangle-estimate replay, uniform-modulus output admission, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from selector or completeness principles.*

Proof. Project the packet. The compact-net row K, pointwise-modulus row F, and dyadic tolerance row D are the only source rows available before the mesh ledger is opened.

The incidence ledger Q pairs each displayed cell with its local modulus and tolerance request. The fold row L is finite because it ranges only over the displayed incidence ledger.

The triangle ledger T records the source and target metric comparisons needed before the tolerance-only output row U can be exported. Thus the uniform-modulus handoff remains attached to compact-net, pointwise-modulus, dyadic, fold, and triangle evidence.

The rows H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores an external selector, host Lebesgue number, quotient metric equality, choice schedule, or ambient compactness theorem, the NameCert obligations are exactly the displayed rows. □

Theorem 8504.3 (Finite modulus mesh uniform handoff route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read exported by an accepted* $\mathsf{FiniteModulusMesh}^{\uparrow}$ *packet factors through*

$$K, F, D \longmapsto Q, L, T \longmapsto U, H, C, P, N.$$

In particular, a compact-uniform-continuity consumer cannot read the tolerance-only output before the compact-net cells, pointwise modulus rows, dyadic tolerance ledger, finite incidence ledger, lower-bound fold, and triangle-estimate ledger have all been displayed.

Proof. Start with an accepted carrier $M = (K, F, D, Q, L, T, U, H, C, P, N)$. Projection gives the compact-net cells, pointwise modulus rows, and dyadic tolerance row as the source stage.

The mesh stage is Q, L, T. The incidence ledger links cells to local moduli and tolerances, the lower-bound fold reads the finite radius rows, and the triangle ledger records the metric estimates used by the output route.

Only after this mesh stage does the carrier expose U. The structural rows H, C, P, N preserve, replay, source, and name the same route.

Therefore every accepted $\mathsf{UniformModulus}^{\uparrow}$ consumer sees the displayed finite mesh before it receives the tolerance-only handoff, and no external selector or compactness theorem is exported. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusMesh}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying compact-net cells, pointwise modulus rows, dyadic tolerance data, finite incidence coverage, lower-bound fold evidence, triangle-estimate replay, transport, provenance, and local naming.

Independence witness: compactmetric, continuousmap, dyadicratcore, real, uniformmodulus: none alone is bijective with $\mathsf{FiniteModulusMesh}^{\uparrow}$ because the packet jointly carries compact-net

cells, pointwise modulus rows, dyadic tolerances, Real-facing metric readback, finite fold data, triangle replay, a $\mathsf{UniformModulus}^{\uparrow}$ handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusMesh}^{\uparrow}$ packet is generated from compact-net cells, continuous-map pointwise modulus rows, dyadic tolerance rows, finite incidence coverage, a lower-bound fold, a triangle-estimate ledger, a UniformModulus output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8504.1, Definition 8504.2, and Definition 8504.3 seed the finite compact-net, pointwise-modulus, dyadic-tolerance, fold, triangle, and uniform-modulus handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subcovers, host Lebesgue numbers, quotient metric equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, pointwise-modulus visibility, dyadic tolerance admission, fold exactness, triangle replay, uniform-modulus output, Nontrivial and FieldFaithful TasteGate instances, transport, provenance, and local naming for BEDC.Derived.FiniteModulusMeshUp.

Chapter 8505

A Concrete Naming Certificate for $\mathsf{FiniteTailOscillationBound}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ packet records the finite boundary by which a regular rational stream exposes a late-tail oscillation estimate before a Real or Completion consumer reads a seal. It sits after the finite stream names of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name surface of chapter 38, and the oscillation-modulus row of chapter 2224. The packet supplies a named finite tail-oscillation boundary without quotient stream equality, selected limits, ambient completeness, trichotomy, or countable choice.

Definition 8505.1 (Finite tail-oscillation bound carrier)**.** *A* $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, O, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected stream positions.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular rational observations on those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare late observations.*
4. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row naming the finite oscillation schedule.*
5. *T is the tail-bound row, recording the displayed finite stage after which the D-bounded oscillation read is accepted.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after S, R, D, O, T have been displayed.*
7. *H records componentwise* hsame *transport for the stream, readback, dyadic, oscillation, tail-bound, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records the* Pkg *provenance over* $\mathsf{FiniteTailOscillationBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTailOscillationBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, O, T, E, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, trichotomy, countable choice, or bridge checking.*

Theorem 8505.2 (Finite tail-oscillation bound NameCert obligations)**.** *For every accepted* FiniteTailOscillationBou *carrier*

$$B = (S, R, D, O, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteTailOscillationBound}^{\uparrow}}$ *surface consists exactly of finite stream-window exposure through S, regular rational readback through R, dyadic tolerance exposure through D, oscillation-modulus exposure through O, tail-bound exactness through T, Real-seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The only analytic rows available before the seal are the finite stream window S, the regular rational readback R, the dyadic tolerance ledger D, the oscillation-modulus row O, and the displayed tail-bound row T.

The tail-bound exactness is therefore finite. A consumer may inspect late observations only through the named windows and readback row, compare them only through the dyadic ledger, and spend only the oscillation schedule recorded by O.

Finally read E, H, C, P, N. The Real seal is downstream from the finite tail route, and the structural rows only transport, replay, source, and name that same route. Since no excluded endpoint is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8505.3 (Finite tail-oscillation bound Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read of an accepted* $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ *packet reaches the seal row E only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{OscillationModulus}^{\uparrow} \longmapsto T \longmapsto \mathsf{Real}^{\uparrow}.$$

The read supplies a finite tail-oscillation boundary for regular Cauchy and Real-seal arguments; it does not identify quotient stream names, select a limit, invoke trichotomy or countable choice, or prove ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8505.2. They place the finite stream window, regular rational readback, dyadic tolerance ledger, oscillation schedule, and tail-bound row before the Real seal can be consumed.

The sibling dependencies give the displayed route. The StreamName row supplies finite windows, RegSeqRat supplies rational readback, DyadicRatCore supplies the tolerance comparisons, and OscillationModulus supplies the finite schedule that bounds the late tail.

The seal row E is therefore a downstream boundary, not an independent completeness theorem. Transport, replay, provenance, and local naming preserve the same finite route, so a Real or Completion consumer receives a named finite tail-oscillation boundary and none of the excluded quotient, choice, order, selected-limit, or completeness data. □

Theorem 8505.4 (Finite tail-oscillation bridge-consumer route)**.** *Any* $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ *consumer that also reads a* $\mathsf{FiniteModulusStaircase}^{\uparrow}$ *packet must factor the displayed tail-bound row through the staircase rows before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. Concretely, the consumer route has the finite form*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{OscillationModulus}^{\uparrow}, T \longmapsto \mathsf{FiniteModulusStaircase}^{\uparrow} \longmapsto E.$$

The staircase dependency is read through the carrier of Definition 8514.1; it does not supply a selected global modulus, infinite search, quotient stream equality, trichotomy, countable choice, selected limit, or ambient completeness theorem.

Proof. Start with the NameCert obligations of Definition 8505.2. They expose the tail-bound row T only after the finite window, regular readback, dyadic tolerance, and oscillation-modulus rows have been displayed.

Use the Real route theorem, Definition 8505.3. It places the seal row E downstream from the same finite tail-bound route, so a consumer cannot read E before the tail-bound row is fixed.

Now read the staircase carrier of Definition 8514.1. Its displayed rows place $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteModulusMesh}^{\uparrow}$, $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$, the staircase ledger, and the terminal handoff before its own Real-facing seal. A tail-bound consumer that imports this packet must therefore route T through those displayed staircase rows before consuming E.

The structural rows in both packets transport, replay, source, and name the same finite data. Since neither carrier contains a selected global modulus, infinite search, quotient equality, trichotomy, choice schedule, selected limit, or ambient completeness row, the bridge-consumer route is exactly the displayed finite staircase handoff. □

Falsifiable prediction: A $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ consumer that reaches a Real or Completion seal without displaying the StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, OscillationModulus schedule, tail-bound row, transport, replay, provenance, and local naming exits the accepted carrier.

Independence witness: streamname, regseqrat, dyadicratcore, oscillationmodulus, and real: no listed sibling is bijective with $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ because this packet jointly binds finite windows, rational readback, dyadic tolerances, an oscillation schedule, a tail-bound row, Real-seal exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailOscillationBound}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailOscillationBound}^{\uparrow}$ packet is generated from a StreamName finite-window row, RegSeqRat readback, DyadicRatCore tolerance ledger, OscillationModulus schedule, a finite tail-bound row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTailOscillationBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8505.1, Definition 8505.2, Definition 8505.3, and Definition 8505.4 seed the finite tail-oscillation boundary over displayed stream-window, regular-rational, dyadic-tolerance, oscillation-modulus, staircase, tail-bound, Real, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTailOscillationBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, trichotomy, countable choice, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{FiniteTailOscillationBound}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular-readback exactness, dyadic-tolerance exactness, oscillation-schedule exposure, tail-bound exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTailOscillationBoundUp.

Chapter 8506

A Concrete Naming Certificate for $\mathsf{FrobeniusNormalForm}^{\uparrow}$

Following Frobenius and the rational-canonical-form tradition, the $\mathsf{FrobeniusNormalForm}^{\uparrow}$ packet records the companion-block presentation of a displayed finite endomorphism as BEDC NameCert data. It sits between the matrix carrier of chapter 49, the linear-map surface of chapter 48, the polynomial rows of chapter 50, the invariant-factor packet of chapter 5843, the characteristic-polynomial route of chapter 6307, the block comparison boundary of chapter 6856, and the polynomial-matrix reduction route of chapter 2262. The packet names only the finite Frobenius companion-block certificate; it does not claim eigenvalue selection, algebraic closure, Jordan block decomposition, host similarity classification, or a detached canonical-form oracle.

Definition 8506.1 (Frobenius normal form carrier)**.** *A* $\mathsf{FrobeniusNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, L, Q, S, R, B, A, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the displayed finite operator, including row-column windows and matrix multiplication requests.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the same endomorphism surface.*
3. *Q is the* $\mathsf{Poly}^{\uparrow}$ *row carrying the monic invariant-factor polynomials and the characteristic-polynomial ledger visible to the block read.*
4. *S is the* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row for the polynomial-matrix reduction route that exposes the invariant factors.*
5. *R is the* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *handoff row, retaining the invariant-factor schedule used by the companion-block assembly.*
6. *B is the Frobenius companion-block row, listing one finite companion block for each displayed invariant factor in Q and the row-column positions used by the block sum.*
7. *A is the block-assembly exactness row, comparing the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation with the companion-block presentation through the same* $\mathsf{LinMap}^{\uparrow}$ *endpoint.*
8. *T is the Cayley-Hamilton compatibility row, recording that the characteristic-polynomial ledger read through chapter 6307 is consumed only as a polynomial boundary for the displayed blocks.*

9. *H records componentwise* hsame *transport for the matrix, linear-map, polynomial, Smith-normal, rational-canonical, companion-block, exactness, Cayley-Hamilton, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Frobenius-form request through $M, L, Q, S, R, B, A, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{FrobeniusNormalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{CayleyHamilton}^{\uparrow}$, $\mathsf{JordanNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, Q, S, R, B, A, T, H, C, P, N$. It contains no coordinate for selected eigenvalues, algebraic closure, Jordan-chain splitting, quotient matrix equality, host similarity classification, non-displayed basis choice, or an ambient canonical-form theorem.

Theorem 8506.2 (Frobenius normal form NameCert obligations)**.** *For every accepted* $\mathsf{FrobeniusNormalForm}^{\uparrow}$ *carrier*

$$F = (M, L, Q, S, R, B, A, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrobeniusNormalForm}^{\uparrow}}$ *surface consists exactly of matrix exposure through M, linear-map exposure through L, invariant-factor polynomial exposure through Q, Smith-normal dependency routing through S, rational-canonical handoff through R, companion-block assembly through B, block exactness through A, Cayley-Hamilton compatibility through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8506.1. The finite operator source is M, L, the polynomial and reduction source is Q, S, and the canonical-form handoff is R.

The Frobenius block data are exactly B, A, T. The companion blocks are indexed by the displayed invariant-factor row, the exactness row compares the block matrix with the original matrix presentation, and the Cayley-Hamilton row is read only as a characteristic-polynomial compatibility boundary.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no carrier coordinate contains eigenvalue selection, algebraic closure, Jordan-chain splitting, quotient equality, host similarity classification, non-displayed basis choice, or an ambient canonical-form theorem, the listed rows exhaust the NameCert surface. □

Theorem 8506.3 (Frobenius companion-block route)**.** *Every companion-block read exported by an accepted* $\mathsf{FrobeniusNormalForm}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{Poly}^{\uparrow}, \mathsf{SmithNormalForm}^{\uparrow}, \mathsf{RationalCanonicalForm}^{\uparrow} \longmapsto B, A, T \longmapsto \mathsf{FrobeniusNormalForm}^{\uparrow}.$$

The route exposes only invariant-factor polynomials, companion-block positions, block-assembly exactness, Cayley-Hamilton compatibility, transport, replay, provenance, and local naming.

Proof. Begin with the matrix and linear-map rows M, L. A companion-block consumer cannot read a Frobenius block before the endomorphism presentation and the polynomial row Q have been displayed.

Next use the Smith-normal and rational-canonical rows S, R. The invariant-factor schedule is obtained through the polynomial-matrix reduction route and then handed to the rational-canonical companion-block surface; it is not imported from a host classification oracle.

The rows B, A, T assemble and check the Frobenius blocks. Componentwise transport, replay, provenance, and local naming through H, C, P, N then preserve exactly this finite route, so the exported block read contains no Jordan-chain or spectral data. □

Theorem 8506.4 (Frobenius rational-canonical handoff)**.** *Any* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *consumer that requests a Frobenius companion presentation from an accepted* $\mathsf{FrobeniusNormalForm}^{\uparrow}$ *packet must keep the invariant-factor schedule R, the polynomial row Q, the Smith-normal dependency S, the companion-block row B, and the exactness row A visible together. The handoff may compare with the Jordan packet of chapter 6856 only as a block-layout boundary, not as an algebraic-closure or eigenvector-selection source.*

Proof. Apply the NameCert obligations of Definition 8506.2. They require the rational-canonical handoff row, invariant-factor polynomial row, Smith-normal route, companion-block row, and exactness row to remain in the same accepted packet.

The rational-canonical chapter supplies the invariant-factor schedule, while the Frobenius packet records the companion-block presentation read from that schedule. Thus a consumer cannot detach the Frobenius blocks from Q, S, R without leaving the displayed NameCert surface.

The Jordan comparison is only a sibling boundary. Since the carrier has no eigenvalue or generalized-eigenvector coordinate, the handoff remains a rational-canonical companion-block route rather than a Jordan decomposition. □

Theorem 8506.5 (Frobenius finite nonescape)**.** *Every accepted* $\mathsf{FrobeniusNormalForm}^{\uparrow}$ *carrier exports a finite normal-form certificate over $M, L, Q, S, R, B, A, T, H, C, P, N$ and no other rows. In particular, a consumer cannot escape from the companion-block route to selected eigenvalues, algebraic closure, Jordan-chain splitting, quotient matrix equality, host similarity classification, non-displayed basis choice, or an ambient canonical-form theorem.*

Proof. The carrier predicate lists exactly twelve rows. The mathematical rows M, L, Q, S, R, B, A, T supply matrix presentation, endomorphism surface, invariant-factor polynomials, Smith-normal dependency, rational-canonical handoff, companion blocks, exactness, and Cayley-Hamilton compatibility.

The structural rows H, C, P, N then transport, replay, source, and name those rows without adding a further mathematical coordinate. Hence every consumer remains inside the finite packet.

All excluded data would require an extra row: eigenvalues or algebraic closure for spectral reads, Jordan chains for nilpotent splitting, quotient equality for host matrix comparison, or a selected basis outside the displayed carrier. Since the packet contains none of these rows, the export is nonescaping. □

Falsifiable prediction: A $\mathsf{FrobeniusNormalForm}^{\uparrow}$ consumer that exports companion blocks while omitting the Matrix, LinearMap, Polynomial, SmithNormalForm, RationalCanonicalForm, companion-block, exactness, Cayley-Hamilton, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 8506.1.

Independence witness: matrix, linearmap, polynomial, smith-normal-form, rational-canonical-form, cayley-hamilton, jordan-normal-form: $\mathsf{FrobeniusNormalForm}^{\uparrow}$ is not bijective to any listed sibling because this packet alone binds invariant-factor polynomials to a finite Frobenius companion-block assembly with exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{FrobeniusNormalForm}^{\uparrow}$ packet is generated from Matrix rows, LinearMap rows, Polynomial invariant-factor rows, a SmithNormalForm dependency route, a RationalCanonicalForm handoff, finite companion-block assembly, block exactness, CayleyHamilton compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8506.1, Definition 8506.2, Definition 8506.3, Definition 8506.4, and Definition 8506.5 seed the finite Frobenius companion-block certificate over matrix, linear-map, polynomial, Smith-normal, rational-canonical, Cayley-Hamilton, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FrobeniusNormalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected eigenvalues, algebraic closure, Jordan-chain splitting, quotient matrix equality, host similarity classification, non-displayed basis choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor routing, companion-block assembly, rational-canonical handoff, Cayley-Hamilton compatibility, finite nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FrobeniusNormalFormUp.

Chapter 8507

A Concrete Naming Certificate for $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ packet records the finite limit-seal route by which regular Cauchy rational data are handed to a real-name consumer without selecting quotient representatives. It sits after the finite window discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the choice-free completion route of chapter 4237. The packet is a finite NameCert limit boundary for completion-facing readers; it is not Real completeness, a quotient stream, a selected limit, countable choice, or host real equality.

Definition 8507.1 (RegularCauchyChoiceFreeLimit carrier)**.** *A* $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, C, E, H, T, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite window schedule inspected by the limit request;*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the scheduled windows;*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare readback windows;*
4. C *is the* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *completion-facing route consumed before any limit seal is read;*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through* S, R, D, C*;*
6. H *records componentwise* hsame *transport for the window, readback, dyadic, completion-route, seal, replay, provenance, and local naming rows;*
7. T *records displayed* Cont *replay for the same limit-seal route;*
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}}$ *row.*

The classifier accepts only packets whose limit read is exhausted by $S, R, D, C, E, H, T, P, N$*. It has no coordinate for Real completeness, quotient stream equality, selected limits, countable-choice schedules, host real equality, or an ambient completion endpoint.*

Theorem 8507.2 (RegularCauchyChoiceFreeLimit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ *carrier*

$$L = (S, R, D, C, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}}$ *surface consists exactly of finite-window exposure through* S*, regular rational readback through* R*, dyadic tolerance exposure through* D*, choice-free completion routing through* C*, terminal Real sealing through* E*, componentwise transport through* H*, displayed replay through* T*, provenance through* P*, and local naming through* N*.*

The obligation surface refuses Real completeness, quotient streams, selected limits, countable choice, host real equality, and ambient completion endpoints.

Proof. Project the carrier of Definition 8507.1. The displayed analytic rows are exactly the finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, completion route, and terminal Real seal.

Read those rows in order. The window schedule S bounds the observations, R supplies the regular rational readback, and D supplies the tolerance ledger before the completion-facing row C and Real seal E can be consumed.

Finally inspect the structural rows H, T, P, N. They transport, replay, source, and name the same finite packet. Since the excluded principles are absent from the carrier inventory, they cannot appear as local NameCert obligations. □

Theorem 8507.3 (RegularCauchyChoiceFreeLimit seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ *carrier*

$$L = (S, R, D, C, E, H, T, P, N)$$

must first read the finite route

$$S, R, D \longmapsto C \longmapsto E \longmapsto H, T, P, N.$$

The terminal seal exports only the displayed regular-Cauchy limit packet. It does not expose Real completeness, quotient streams, selected limits, countable choice, host real equality, or an ambient completion endpoint.

Proof. Start with the NameCert obligations of Definition 8507.2. They force every local read through the finite window, readback, dyadic tolerance, completion route, Real seal, and structural rows.

Use the sibling completion route of chapter 4237. The row C is consumed only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows have been displayed, and it gives no quotient representative or selected limit.

Finally the $\mathsf{Real}^{\uparrow}$ seal row E is downstream of the same finite route. The transport, replay, provenance, and naming rows preserve that route rather than adding new analytic data, so no consumer can escape from the displayed limit packet to any refused endpoint. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ read exposes StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, RegularCauchyChoiceFreeCompletion routing, RealUp sealing, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any limit-facing consumer reads the seal.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-choice-free-completion: none is bijective with $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ because this packet alone binds the finite window, regular readback, dyadic tolerance, choice-free completion route, and terminal limit seal while refusing quotient streams, selected limits, countable choice, and host real equality.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RegularCauchyChoiceFreeCompletion route rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8507.1, Definition 8507.2, and Definition 8507.3 seed the finite choice-free regular-Cauchy limit route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected limits, countable choice, host real equality, ambient completion endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, regular readback, dyadic tolerance stability, choice-free completion routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyChoiceFreeLimitUp.

Chapter 8508

A Concrete Naming Certificate for $\mathsf{NestedIntervalSequence}^{\uparrow}$

Following Cantor's nested-interval principle in the Bishop–Bridges constructive reading, the $\mathsf{NestedIntervalSequence}^{\uparrow}$ packet records a nested located interval sequence as finite BEDC certificate data before any Real or Completion consumer reads a limiting name. It sits over the located interval surface of chapter 394, the dyadic nested-interval sibling of chapter 328, the squeeze route of chapter 8247, the dyadic rational core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the interval-sequence surface itself: endpoint rows, nesting evidence, diameter budgets, finite-window readback, transport, replay, provenance, and local naming.

Definition 8508.1 (Nested interval sequence carrier)**.** *A* $\mathsf{NestedIntervalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (I, E, L, D, S, R, Z, H, C, P, A)$$

with the following displayed rows:

1. *I is the finite interval-row table, each row read as a located interval through* $\mathsf{LocatedInterval}^{\uparrow}$ *and carrying displayed lower and upper endpoint names.*
2. *E is the endpoint-comparison ledger, recording the endpoint order and endpoint equality tests available to the inspected interval rows.*
3. *L is the nesting ledger, recording that every later inspected interval lies inside every earlier inspected interval named by the common schedule.*
4. *D is the diameter-budget ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, assigning each displayed interval a rational or dyadic shrinkage budget before any regular readback is allowed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected indices, endpoint requests, nesting checks, and replay order.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed endpoints, nesting ledger, and diameter budgets.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, E, L, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for interval rows, endpoint comparison, nesting, diameter budgets, finite windows, readback, Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a tolerance request through the diameter budget, then through the scheduled interval rows and nesting ledger, then through regular rational readback and the terminal Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalSequence}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{NestedIntervalSqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *A is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, E, L, D, S, R, Z, H, C, P, A$. It has no coordinate for open-cover compactness, selected limits, countable choice, real-order trichotomy, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8508.2 (Nested interval sequence NameCert obligation surface)**.** *For every accepted* $\mathsf{NestedIntervalSequence}^{\uparrow}$ *carrier*

$$N = (I, E, L, D, S, R, Z, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}$ *surface consists exactly of interval-row admission, endpoint comparison, nesting-ledger exactness, diameter-budget readback, finite-window scheduling,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8508.1. The only displayed coordinates are $I, E, L, D, S, R, Z, H, C, P, A$, so interval admission, endpoint comparison, nesting evidence, diameter control, finite observation, readback, Real sealing, transport, replay, provenance, and local naming are all direct readings of the packet.

Read the interval data first. The row I supplies the located interval rows, E supplies endpoint comparison, and L supplies the displayed containment relation between later and earlier inspected intervals.

Read the shrinkage and observation rows next. The diameter ledger D is the only route by which a tolerance request reaches the interval sequence, and S is the only route by which a consumer selects the inspected windows. The handoff row R and seal row Z are downstream of those displayed rows.

Finally H, C, P, A preserve the same finite route under transport, replay, source, and local naming. Since the carrier has no coordinate for compactness, a selected limit, choice, trichotomy, quotient equality, or ambient completeness, the displayed rows exhaust the local NameCert obligation surface. □

Theorem 8508.3 (Nested interval sequence shrink induction)**.** *For an accepted* $\mathsf{NestedIntervalSequence}^{\uparrow}$ *carrier*

$$N = (I, E, L, D, S, R, Z, H, C, P, A),$$

finite prefix shrink data obtained from endpoint comparison rows E, nesting evidence L, diameter budgets D, $\mathsf{StreamName}^{\uparrow}$ *window rows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows Z is preserved by induction on the displayed finite windows. In particular, every displayed window prefix whose last interval has certified endpoints and diameter budget inherits the earlier-prefix containment and shrink ledger, and the successor window is obtained by consuming the next nesting row and next diameter row before transporting the result through H, C, P, and the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}$ *row.*

Proof. For the base window, read Definition 8508.1. A one-row prefix has endpoint comparison from E, its displayed diameter budget from D, its scheduled observation from S, and the downstream readback and seal rows R and Z.

For the successor step, assume the shrink ledger has been built for the displayed prefix named by S. The next window row supplies the next interval entry from I, the endpoint comparison row from E, the nesting row from L placing it inside the earlier displayed intervals, and the next diameter budget from D.

The successor ledger is then transported along the componentwise hsame row H and replayed through C. The provenance row P records that the construction used only the displayed $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{NestedIntervalSqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces, and the local $\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}$ row repacks the finite-window induction as a BEDC certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalSequence}^{\uparrow}$ packet is generated from located interval rows, endpoint-comparison rows, a nesting ledger, a DyadicRatCore diameter-budget ledger, StreamName finite windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8508.1 and Definition 8508.2 seed the finite nested located interval sequence route into the regular-rational and Real-facing stack.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, open-cover compactness, selected limits, countable choice, trichotomy, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint comparison, nesting exactness, diameter-budget monotonicity, finite-window replay, regular-rational handoff, Real-seal nonescape, transport, provenance, and local naming for BEDC.Derived.NestedIntervalSequenceUp.

Chapter 8509

A Concrete Naming Certificate for $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$

Following the compact-uniform-continuity construction of chapter 9237, the $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$ packet records the finite $\mathsf{ProbeBundle}$ fold that turns bundled pointwise radii into one tolerance-only lower-bound history. It sits over the compact source data of chapter 60, the metric distance rows of chapter 57, the continuous pointwise modulus surface of chapter 59, the rational order readback of chapter 37, and the subordinate cover packet of chapter 596. The packet names only the finite lower-bound fold and its handoff to uniform-modulus consumers; it is not an infinite cover, a selected Lebesgue number, countable choice, a host compactness theorem, or an ambient real-completeness principle.

Definition 8509.1 (Finite modulus lower-bound ledger carrier)**.** *A* $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, F, Q, B, A, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row exposing the finite compact cover windows that supply the bundled probes.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used for source-ball and target-triangle comparisons in the compact-uniform route.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row, assigning each bundled center its local positive rational precision history.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *order-readback row for positive rational histories and finite lower-bound comparisons.*
5. *B is the nonempty finite* $\mathsf{ProbeBundle}$ *row whose bundled entries carry centers, half-radius cover rows, local modulus rows, and subordinate radius histories.*
6. *A is the accumulator ledger produced by primitive recursion over B; each successor step compares the current accumulator with the next displayed half-radius history using Q.*
7. *U is the uniform-modulus handoff row, recording the tolerance-only precision history certified below every bundled half-radius ledger.*

8. *H records componentwise* hsame *transport for compact source, metric, continuous modulus, rational order, probe bundle, accumulator, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the finite fold*

$$K, M, F, Q, B, A \longmapsto U, H, P, N.$$

10. *P is the* Pkg *provenance over* FiniteModulusLowerBoundLedger$^{\uparrow}$, Compact$^{\uparrow}$, Met$^{\uparrow}$, Continuous$^{\uparrow}$, Rat$^{\uparrow}$, SubordinateModulusCover$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{FiniteModulusLowerBoundLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, F, Q, B, A, U, H, C, P, N$. It has no coordinate for an infinite cover, a host minimum operator, selected Lebesgue number, countable choice, quotient real equality, or ambient completeness.

Theorem 8509.2 (Finite modulus lower-bound ledger fold obligation)**.** *For every accepted* FiniteModulusLowerBou *carrier*

$$L = (K, M, F, Q, B, A, U, H, C, P, N),$$

the local NameCert$_{\text{FiniteModulusLowerBoundLedger}^{\uparrow}}$ *surface consists exactly of compact-source exposure, metric-distance exposure, continuous pointwise-modulus exposure, rational order readback, nonempty finite* ProbeBundle *exposure, accumulator-step exactness, tolerance-only uniform-modulus handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8509.1. The compact, metric, continuous, rational, bundle, accumulator, handoff, and structural rows are exactly $K, M, F, Q, B, A, U, H, C, P, N$.

Read the source rows first. The compact row K and subordinate cover data inside B expose finitely many bundled centers and half-radius histories, while F exposes the pointwise continuous-modulus rows attached to those centers.

Run the fold through Q and A. At the initial and successor stages, the accumulator is compared with the displayed rational radius history for the next bundled probe, and the resulting lower-bound row is stored as finite BEDC history rather than as an external minimum.

The handoff row U is downstream of that fold and records only the tolerance-only precision certified below every bundled half-radius ledger. Transport, replay, provenance, and the local name preserve the same route through H, C, P, N, so no excluded compactness, choice, quotient, or completeness principle is exported. □

Closure status for NameCert$_{\text{FiniteModulusLowerBoundLedger}^{\uparrow}}$**.**

Constructive story: A FiniteModulusLowerBoundLedger$^{\uparrow}$ packet is generated from compact source rows, metric-distance rows, continuous pointwise-modulus rows, Rat order-readback rows, a nonempty finite ProbeBundle, a primitive-recursive accumulator ledger, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteModulusLowerBoundLedger}^{\uparrow}}$, seed)

Scope closed: Definition 8509.1 and Definition 8509.2 seed the finite rational lower-bound fold used by compact-uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{FiniteModulusLowerBoundLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an infinite cover, host minimum operator, selected Lebesgue number, countable choice, quotient real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, metric-distance exposure, continuous-modulus exposure, Rat order readback, finite ProbeBundle recursion, accumulator exactness, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusLowerBoundLedgerUp.

Chapter 8510

A Concrete Naming Certificate for $\mathsf{AumannIntegral}^{\uparrow}$

Following Aumann on integrals of set-valued maps, the $\mathsf{AumannIntegral}^{\uparrow}$ packet records the finite NameCert surface by which displayed selections of a correspondence expose a Banach-valued integral read. It sits after the set-valued map carrier of chapter 8121, the boundary discipline of chapter 6888, the measurable source of chapter 92, the integral readback of chapter 123, the convex-value row of chapter 208, the observable input surface of chapter 185, and the normed target of chapter 90. The packet names a finite integration object for displayed selection rows; it does not assert measurable-selection existence, quotient equality of selections, countable choice, or measure completion.

Definition 8510.1 (Aumann integral carrier)**.** *An* $\mathsf{AumannIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, R, X, F, G, S, K, I, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row carrying the measurable-event window on which all reads are tested.*
2. *R is the* $\mathsf{RandomVar}^{\uparrow}$ *observation row for the inspected source variables and their measurable preimage rows.*
3. *X is the* $\mathsf{SetValuedMap}^{\uparrow}$ *correspondence row, including the displayed source window and value-fibre addresses.*
4. *F is the* $\mathsf{ConvexSet}^{\uparrow}$ *value row, recording nonempty convex fibres only through finite membership and convex-combination tests.*
5. *G is the measurable-graph row that binds source observations in M, R to value candidates in X, F.*
6. *S is the finite selection ledger. Its entries are displayed rows*

$$(s_i, \gamma_i, \mu_i, \rho_i)_{i \in I_0}$$

where s_i is a named selection candidate, γ_i is its graph-admission witness through G, μ_i is its measurable-read witness through M, R, and ρ_i is its fibre-membership witness through F.

7. *K is the selection-integrability row. It records exactly which displayed selection candidates have accepted integral requests.*
8. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row, reducing every accepted selection integral to the finite integral data attached to K.*
9. *B is the* $\mathsf{Banach}^{\uparrow}$ *target row in which the displayed integral endpoints, addition reads, scalar reads, and norm comparisons are named.*
10. *H records componentwise* hsame *transport for $M, R, X, F, G, S, K, I, B$ and the structural rows.*
11. *C records displayed* Cont *replay of the route*

$$M, R, X, F, G \longmapsto S, K \longmapsto I, B.$$

12. *P is the* Pkg *provenance over* $\mathsf{AumannIntegral}^{\uparrow}$, $\mathsf{SetValuedMap}^{\uparrow}$, $\mathsf{AumannIntegralBoundary}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AumannIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for producing selections, identifying selections modulo almost-everywhere equality, completing the measure space, or invoking a choice principle outside the displayed ledger.

Theorem 8510.2 (Aumann integral NameCert obligations)**.** *For every accepted* $\mathsf{AumannIntegral}^{\uparrow}$ *carrier*

$$A = (M, R, X, F, G, S, K, I, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AumannIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, random-variable row, set-valued map row, convex-value row, measurable graph row, finite selection ledger, selection-integrability row, integral readback row, Banach target row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N.*

Proof. Project the carrier of Definition 8510.1. The rows M, R, X, F provide the measurable source, observable input, correspondence, and convex-value data.

The analytic rows are G, S, K, I, B. The graph row admits only displayed graph tests, the finite ledger lists only selection candidates already present in the packet, the integrability row records which of those candidates are read by the integral row, and the Banach row names the endpoints and norm comparisons.

The structural rows H, C, P, N transport, replay, source, and name this same finite route. Since the carrier has no row for producing selections, identifying them by an ambient quotient, completing the measure space, or invoking countable choice, the listed rows exhaust the local certificate surface. □

Theorem 8510.3 (Aumann integral selection route)**.** *Every accepted* $\mathsf{AumannIntegral}^{\uparrow}$ *read that exports a Banach-valued integral endpoint factors through the finite route*

$$\mathsf{SetValuedMap}^{\uparrow}, \mathsf{AumannIntegralBoundary}^{\uparrow}, \mathsf{Measure}^{\uparrow}, \mathsf{RandomVar}^{\uparrow}, \mathsf{ConvexSet}^{\uparrow} \longmapsto S, K \longmapsto \mathsf{Integral}^{\uparrow}, \mathsf{Banach}^{\uparrow}.$$

The route may inspect displayed selection candidates and their integral readbacks, but it cannot infer the existence of an unlisted measurable selection or identify two selections by an ambient almost-everywhere quotient.

Proof. Begin with the source rows M, R, X, F, G. These rows expose the measurable window, the observed source variables, the set-valued graph, and the convex-value membership tests.

Next read the ledger rows S, K. The packet supplies only the listed selection candidates and only the integrability requests recorded for those candidates. Thus the route is a consumer of displayed selections, not a theorem producing selections.

Finally the rows I, B, H, C, P, N read the accepted integrals, place their endpoints in the Banach target, and replay the same route under transport and provenance. No step adds a measure completion row, a countable-choice row, or an almost-everywhere quotient row, so the exported endpoint remains inside the finite Aumann integral packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AumannIntegral}\uparrow}$.

Constructive story: An $\mathsf{AumannIntegral}^{\uparrow}$ packet is generated from displayed measure, random-variable, set-valued-map, convex-value, measurable-graph, finite-selection, selection-integrability, integral-readback, Banach-target, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AumannIntegral}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8510.1, Definition 8510.2, and Definition 8510.3 seed the finite Aumann integral route through displayed selection ledgers and Banach integral readbacks.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AumannIntegral}\uparrow}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AumannIntegralUp`

Bridge status: none

Not claimed: This chapter does not claim measurable-selection existence, quotient equality of selections, countable choice, measure completion, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, measurable graph exactness, finite selection ledger exactness, selection-integrability exactness, integral readback, Banach target preservation, transport stability, and replay stability for BEDC.Derived.AumannIntegralUp.

Chapter 8511

A Concrete Naming Certificate for $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$

Bishop–Bridges and Troelstra-style constructive continuity principles treat diagonal arguments over function spaces as finite evidence disciplines rather than as licenses to choose a discontinuity point. The $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$ packet records the boundary where a diagonal probe against a real-valued continuity claim must expose its source windows, rational readback, and replay rows before any obstruction is accepted. It sits beside the finite window route of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the continuous-map surface of chapter 130, and the Baire-space source discipline of chapter 334. The packet names a choice-free diagonal continuity barrier, not a completed counterexample extracted from countable choice or ambient completeness.

Definition 8511.1 (Bishop diagonal continuity barrier carrier)**.** *A* $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, A, F, U, W, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, recording the finite address requests used by the diagonal probe.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing regular rational values attached to those source windows.*
3. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has been displayed.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose claimed continuity is being inspected through the displayed windows.*
5. *U is the* $\mathsf{BaireSpace}^{\uparrow}$ *or source-window data row supplying the finite branch segment on which the diagonal probe is formed.*
6. *W is the diagonal probe-window row, pairing the inspected source window with the requested target tolerance.*
7. *D is the barrier ledger, recording that the diagonal read is an obstruction to an unexposed continuity route rather than a selected discontinuity point.*

8. *H records componentwise* hsame *transport for the source-window, regular readback, Real seal, continuous-map, source data, diagonal window, barrier ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a continuity request through S, R, A, F, U, W, D, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, A, F, U, W, D, H, C, P, N$. It has no coordinate for countable choice, a selected discontinuity point, quotient function equality, or ambient Real completeness.

Theorem 8511.2 (Bishop diagonal continuity barrier NameCert obligations)**.** *For every accepted* $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$ *carrier*

$$B = (S, R, A, F, U, W, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *source windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{ContinuousMap}^{\uparrow}$ *row,* $\mathsf{BaireSpace}^{\uparrow}$ *or source-window row, diagonal probe windows, barrier ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every barrier read must expose $S, R, A, F, U, W, D, H, C, P, N$ and no obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8511.1. The analytic rows are exactly the source windows, regular rational readback, Real seal, continuous-map row, source data, diagonal probe windows, and barrier ledger.

The continuity-facing read is F, U, W, D. The source data row fixes the finite branch segment, the diagonal window records the inspected request, and the barrier ledger records only the failure of an unexposed continuity route.

The Real-facing read is S, R, A. The StreamName window and regular rational readback are displayed before the Real seal is consumed, so the packet cannot appeal to ambient Real completeness.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since no carrier coordinate contains countable choice, a selected discontinuity point, quotient function equality, or ambient Real completeness, the listed rows exhaust the NameCert surface. □

Theorem 8511.3 (Bishop diagonal continuity barrier choice-free route)**.** *Every accepted* BishopDiagonalContinuityB *read is a choice-free route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{BaireSpace}^{\uparrow} \longmapsto W, D \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

It may obstruct a continuity claim only through displayed source windows, regular rational readback, Real sealing, continuous-map inspection, source-window data, diagonal probe windows, transport, replay, provenance, and local naming. It does not export countable choice, a selected discontinuity point, quotient function equality, or ambient Real completeness.

Proof. Start with the NameCert obligations of Definition 8511.2. They require every accepted read to expose the source windows, regular readback, Real seal, continuous-map row, source data, diagonal window, and barrier ledger.

Read the diagonal stage through U, W, D. The source-window data and probe window are finite rows, so the obstruction is tied to a displayed request rather than to a choice-selected point.

Read the Real stage through S, R, A. The regular rational readback and terminal seal are consumed only after the finite StreamName windows are visible.

The structural rows H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Because the carrier omits countable choice, selected discontinuity points, quotient function equality, and ambient Real completeness, the accepted barrier remains the stated choice-free finite packet. □

Falsifiable prediction: A $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$ consumer that exports a diagonal continuity obstruction without the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, source-window, diagonal probe, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 8511.1.

Independence witness: streamname, regseqrat, real, continuousmap, bairespace: BishopDiagonalContinuityBarrie is not bijective to any listed sibling because this packet alone binds source-window data, diagonal probe windows, barrier-ledger routing, transport, replay, provenance, and local naming around a choice-free continuity obstruction.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}$ packet is generated from StreamName source windows, RegSeqRat readback, a Real seal, a ContinuousMap row, BaireSpace or source-window data, diagonal probe windows, a barrier ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8511.1, Definition 8511.2, and Definition 8511.3 seed the choice-free diagonal continuity barrier over finite source windows, regular rational readback, Real seals, continuous-map inspection, source-window data, diagonal probe windows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDiagonalContinuityBarrier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected discontinuity points, quotient function equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, regular readback, continuous-map inspection, diagonal probe-window routing, barrier-ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopDiagonalContinuityBarrierUp.

Chapter 8512

A Concrete Naming Certificate for $\mathsf{CauchySequenceModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{CauchySequenceModulus}^{\uparrow}$ packet records the finite tail-budget data by which a displayed sequence can be read as Cauchy before any Real or Completion consumer inspects its terminal seal. It sits over the effective Cauchy sequence route of chapter 1705, the regular Cauchy sequence packet of chapter 660, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the finite tail handoff of chapter 8472, the lower-bound ledger of chapter 8509, and the terminal real-name seal of chapter 38. The packet names explicit tail-budget rows only; it does not assert quotient equality of Cauchy names, select a limit, invoke countable choice, decide trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8512.1 (Cauchy sequence modulus carrier)**.** *A* $\mathsf{CauchySequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, R, W, Q, L, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row, exposing the effective Cauchy reading before a tail budget is exported.*
2. *R is the* $\mathsf{RegularCauchySequence}^{\uparrow}$ *regularity row, recording the regular Cauchy sequence discipline that the modulus serves.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, displaying the inspected sequence windows and index requests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding the displayed stream windows to regular rational approximation data.*
5. *L is the finite tail-bound ledger row, storing the explicit threshold and tolerance histories used to compare later sequence windows.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after E, R, W, Q, L have been displayed.*
7. *H records componentwise* hsame *transport for source, regularity, window, readback, ledger, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$E, R, W, Q, L \longmapsto S, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceModulus}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, R, W, Q, L, S, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable-choice tail selector, trichotomy, host real equality, or ambient completeness.

Theorem 8512.2 (Cauchy sequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceModulus}^{\uparrow}$ *carrier*

$$M = (E, R, W, Q, L, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceModulus}^{\uparrow}}$ *surface consists exactly of effective Cauchy source exposure, regular Cauchy sequence exposure, finite stream-window exposure, regular rational readback, finite tail-bound ledger exposure, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8512.1. The accepted packet displays exactly the effective source, regular sequence, stream-window, readback, tail-ledger, Real-seal, and structural rows.

Read the source route first. The row E exposes the effective Cauchy sequence data, while R records the regular Cauchy discipline already used by chapter 660. The stream row W and readback row Q make the inspected finite windows and rational approximations visible before any tail budget is consumed.

The tail-budget row L is the only modulus ledger. It stores the displayed threshold and tolerance histories and then hands the resulting route to the seal row S; the seal is therefore downstream of E, R, W, Q, L, not a detached completeness or quotient principle.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no selected limit, quotient equality, countable-choice selector, trichotomy, host equality, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceModulus}^{\uparrow}$ packet is generated from an EffectiveCauchySequence source row, a RegularCauchySequence discipline row, finite StreamName windows, RegSeqRat readback, a finite tail-bound ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8512.1 and Definition 8512.2 seed the explicit tail-budget carrier consumed by L10 Real and Completion routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy names, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective Cauchy source exposure, regular Cauchy sequence exposure, finite stream-window exposure, RegSeqRat readback, finite tail-ledger exactness, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceModulusUp.

Chapter 8513

A Concrete Naming Certificate for $\mathsf{CompactUniformEquioscillation}^{\uparrow}$

Following Arzela–Ascoli equicontinuity and Chebyshev equioscillation practice, the $\mathsf{CompactUniformEquioscillation}^{\uparrow}$ packet records the finite compact-uniform surface on which oscillation bands are compared over a compact source before a uniform-modulus or approximation consumer reads them. It concretizes the compact-uniform-continuity route of chapter 9237 after the compact metric rows of chapter 129, the equicontinuity family rows of chapter 413, the Arzela–Ascoli family packet of chapter 412, the finite modulus handoff of chapter 5973, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names finite oscillation bands, compact-net windows, and equicontinuity rows; it does not assert host suprema, selected extrema, quotient equality of functions, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8513.1 (CompactUniformEquioscillation carrier)**.** *A* $\mathsf{CompactUniformEquioscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, Q, B, O, D, S, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, exposing compact source centers, coverage radii, and located metric comparisons.*
2. *F is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, recording the displayed family-wide modulus rows inspected on the compact source.*
3. *Q is the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *family-control row, read only as finite boundedness and equicontinuity data.*
4. *B is the compact-uniform band row, naming the finite lower and upper comparison bands on each inspected net cell.*
5. *O is the equioscillation row, recording the finite alternation schedule, signed band gaps, and equality-of-oscillation comparisons between neighboring displayed cells.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the band-gap and oscillation comparisons.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting compact-net cells, family observations, band comparisons, and alternation checks.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational band gaps, oscillation widths, and tolerance comparisons.*
9. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported only after K, F, Q, B, O, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for compact nets, equicontinuity, family control, bands, equioscillation, dyadic tolerances, finite windows, regular readback, uniform-modulus handoff, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from a compact-uniform query through the finite net, family rows, band comparisons, equioscillation schedule, numerical readback, and uniform-modulus handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{CompactUniformEquioscillation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, Q, B, O, D, S, R, U, H, C, P, N$. It has no coordinate for host extrema, selected maximizers, quotient function equality, countable choice, open-cover compactness, or ambient Real completeness.

Theorem 8513.2 (CompactUniformEquioscillation obligation surface)**.** *For every accepted* $\mathsf{CompactUniformEquios}$ *carrier*

$$E = (K, F, Q, B, O, D, S, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, equicontinuity-family exposure through F, Arzela–Ascoli family-control exposure through Q, compact-uniform band admission through B, finite equioscillation schedule exposure through O, dyadic tolerance visibility through D, finite-window scheduling through S, regular rational readback through R, uniform-modulus handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8513.1. The compact source, family, and control rows are K, F, Q, so every band comparison is read after the compact-net and equicontinuity data are visible.

Rows B, O, D, S, R give the finite oscillation surface. The band row supplies the lower and upper comparison windows, the equioscillation row records the alternation and equal-width comparisons, and the dyadic, stream, and regular-rational rows expose the numerical checks before any consumer may use them.

The uniform-modulus handoff U is downstream of those displayed rows. Componentwise transport, replay, provenance, and local naming preserve the same finite route, so the packet exports no host extrema, selected maximizers, quotient function equality, countable-choice schedule, open-cover compactness theorem, or ambient completeness principle. □

Theorem 8513.3 (Compact-uniform equioscillation handoff)**.** *Every compact-uniform, Arzela–Ascoli, or Chebyshev-facing consumer of an accepted* $\mathsf{CompactUniformEquioscillation}^{\uparrow}$ *packet first reads*

$$K, F, Q \longmapsto B, O \longmapsto D, S, R \longmapsto U \longmapsto H, C, P, N.$$

Thus the consumer receives a finite compact-net and equicontinuity route together with displayed equioscillation bands before it may read the uniform-modulus handoff.

Proof. Start with the source and family rows. The compact metric row K, equicontinuity row F, and Arzela–Ascoli family-control row Q determine the finite family surface on which oscillation bands can be inspected.

Next read B and O. These rows bind the lower and upper band data to the displayed alternation schedule, so the equioscillation claim is a finite comparison ledger rather than a selected extremal point or host supremum.

Finally pass through D, S, R, U, H, C, P, N. The dyadic tolerance, stream window, and regular-rational readback rows expose the numerical comparisons before the uniform-modulus row is exported, while the structural rows keep the handoff inside the same local $\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformEquioscillation}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, EquicontinuityFamily rows, ArzelaAscoli family-control rows, compact-uniform band rows, finite equioscillation schedules, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8513.1, Definition 8513.2, and Definition 8513.3 seed the compact finite-net, equicontinuity-family, Arzela–Ascoli, band-comparison, equioscillation, rational-readback, and uniform-modulus route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformEquioscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host extrema, selected maximizers, quotient equality of functions, countable choice, open-cover compactness, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, equicontinuity-family exposure, Arzela–Ascoli family-control exposure, band exactness, equioscillation schedule exactness, dyadic tolerance visibility, finite-window replay, regular readback, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformEquioscillationUp.

Chapter 8514

A Concrete Naming Certificate for $\mathsf{FiniteModulusStaircase}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{FiniteModulusStaircase}^{\uparrow}$ packet records a finite staged route by which a modulus request is tightened through displayed rational tolerances before a Real or Completion consumer reads the final seal. It sits after the finite stream names of chapter 126, the regular rational readback of chapter 301, the dyadic core of chapter 291, the finite mesh surface of chapter 8504, the lower-bound ledger of chapter 8509, and the Real seal of chapter 38. The packet names only a finite staircase of modulus rows, monotone handoff between adjacent rows, transport, replay, provenance, and local naming; it is not an infinite modulus search, a selected global modulus, quotient stream equality, countable choice, trichotomy, or ambient completeness.

Definition 8514.1 (Finite modulus staircase carrier)**.** *A* $\mathsf{FiniteModulusStaircase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, M, L, A, U, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the stream positions whose modulus requests are inspected.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing rational observations on those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare adjacent requests.*
4. *M is the* $\mathsf{FiniteModulusMesh}^{\uparrow}$ *row exposing the finite mesh cells from which staged requests are drawn.*
5. *L is the* $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$ *row exposing the finite lower-bound fold available to every displayed stage.*
6. *A is the finite staircase ledger. Its initial row records the first displayed modulus request, and each successor row records a dyadic comparison against the previous request together with the lower-bound handoff read from L.*
7. *U is the terminal modulus handoff row, obtained only after every displayed staircase step in A has been read.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only downstream from W, R, D, M, L, A, U.*

9. *H records componentwise* hsame *transport for stream windows, regular readback, dyadic tolerances, mesh cells, lower-bound rows, staircase steps, terminal handoff, Real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay of the finite staircase route*

$$W, R, D, M, L \longmapsto A \longmapsto U, E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusStaircase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteModulusMesh}^{\uparrow}$, $\mathsf{FiniteModulusLowerBoundLedger}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusStaircase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, M, L, A, U, E, H, C, P, N$. It has no coordinate for an infinite search over moduli, selected global modulus, quotient stream equality, trichotomy, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8514.2 (Finite modulus staircase NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusStaircase}^{\uparrow}$ *carrier*

$$S = (W, R, D, M, L, A, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusStaircase}^{\uparrow}}$ *surface consists exactly of finite stream-window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, finite mesh exposure through M, lower-bound ledger exposure through L, staircase-step exactness through A, terminal modulus handoff through U, Real-seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The source rows available before the staircase opens are exactly the finite window, regular readback, dyadic tolerance, finite mesh, and lower-bound ledger rows W, R, D, M, L.

Read the staircase ledger A by finite induction over its displayed rows. The initial row records a visible modulus request, and each successor row records only the next visible request, the dyadic comparison against the previous request, and the lower-bound handoff already displayed by L.

The terminal row U is therefore downstream from a finite chain of visible comparisons, not from an external search. The Real seal E and the structural rows H, C, P, N transport, replay, source, and name that same finite route.

Since no excluded endpoint is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8514.3 (Finite modulus staircase monotone handoff)**.** *Every terminal modulus handoff exported by an accepted* $\mathsf{FiniteModulusStaircase}^{\uparrow}$ *packet factors through the displayed staircase rows:*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{FiniteModulusMesh}^{\uparrow}, \mathsf{FiniteModulusLowerBoundLedger}^{\uparrow} \longmapsto A \longmapsto U \longmapsto$$

Each successor row in A may tighten the current request only through the dyadic comparison and lower-bound handoff already present in the carrier. Thus a Real or Completion consumer receives a staged finite-modulus dependency rather than a one-shot modulus assertion.

Proof. Start with the obligations of Definition 8514.2. They place stream windows, regular readback, dyadic tolerances, finite mesh data, and lower-bound ledger data before the staircase ledger can be consumed.

Run finite induction along A. The base row supplies the first displayed request. At a successor row, the only available comparison is the dyadic one recorded by D, and the only lower-bound source is the ledger row L.

After the last displayed step, the terminal handoff U can be read. The seal row E and the structural rows preserve the same finite route, so no Real or Completion consumer can bypass the staircase by appealing to a selected global modulus, infinite search, quotient equality, choice schedule, or ambient completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusStaircase}^{\uparrow}$ read is refuted if a Real or Completion consumer exports a terminal modulus handoff without displaying the StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, FiniteModulusMesh cells, FiniteModulusLowerBoundLedger fold, staircase rows, transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, finite-modulus-mesh, finite-modulus-lower-bound-ledger, and real: none alone is bijective with $\mathsf{FiniteModulusStaircase}^{\uparrow}$ because the packet jointly binds finite windows, rational readback, dyadic comparisons, mesh rows, lower-bound rows, staged staircase steps, terminal handoff, Real-seal exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusStaircase}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusStaircase}^{\uparrow}$ packet is generated from StreamName finite-window rows, RegSeqRat readback rows, DyadicRatCore tolerance rows, FiniteModulusMesh rows, FiniteModulusLowerBoundLedger rows, a finite staircase ledger, a terminal modulus handoff row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusStaircase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8514.1, Definition 8514.2, and Definition 8514.3 seed the finite staged-modulus route over displayed stream-window, regular-readback, dyadic-tolerance, mesh, lower-bound, staircase, Real, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusStaircase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an infinite modulus search, selected global modulus, quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{FiniteModulusStaircase}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular-readback exactness, dyadic-comparison exactness, mesh exposure, lower-bound ledger exposure, staircase-step exactness, terminal handoff nonescape, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusStaircaseUp.

Chapter 8515

A Concrete Naming Certificate for $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy convergence, the $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ packet records the explicit finite threshold schedule by which a displayed regular rational stream becomes a Real-facing Cauchy name. It sits after the stream-window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal Real seal of chapter 38, and the completion route of chapter 3379. The packet names an explicit convergence modulus only; it does not choose a host limit, quotient stream representatives, countably many witnesses, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8515.1 (Modulus of Cauchy convergence carrier)**.** *A* $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, carrying the inspected finite windows of the rational stream.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the dyadic precision requests and their order comparison ledger.*
3. *T is the tail-threshold schedule. For each displayed tolerance request in D, it names a finite threshold index and the finite window in S on which all later inspected pairs are compared.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reading the regular rational sequence through the same stream windows and tolerance requests.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the scheduled tail comparisons and the regular rational handoff have been read.*
6. *H records componentwise* hsame *transport for S, D, T, R, E and the structural rows.*
7. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto T \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{ModulusOfCauchyConvergence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for selecting an infinite family of tail witnesses, passing to quotient Cauchy classes, choosing a completed Real representative, or invoking completeness outside the displayed completion route.

Theorem 8515.2 (Modulus of Cauchy convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ *carrier*

$$M = (S, D, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusOfCauchyConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the stream source, dyadic tolerance row, tail-threshold schedule, regular rational handoff, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* N*.*

Proof. Project the carrier of Definition 8515.1. The rows S, D, T provide the only admissible source windows, tolerance requests, and threshold schedule.

The handoff rows R, E are read after that schedule is fixed. Thus a Real-facing consumer can inspect the regular rational row and the terminal seal only through the same finite tail data.

The structural rows H, C, P, N transport, replay, source, and name this finite route. Since no coordinate stores a choice functional, quotient class, selected host limit, or ambient completeness theorem, the listed rows exhaust the local certificate surface. □

Theorem 8515.3 (Modulus of Cauchy convergence tail witness)**.** *Every accepted tolerance request in a* $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ *packet is discharged by the displayed tail-threshold row* T*. More precisely, if a consumer asks for the dyadic tolerance named by a row of* D*, the packet returns the threshold and finite stream window named in* T*, and every later inspected pair used by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is routed through that same window.*

Proof. Begin with the requested dyadic tolerance. By carrier admission, it is a row of D, so the schedule coordinate T contains the corresponding threshold entry.

Read the stream source through S at the window named by that entry. The regular rational handoff row R is defined over the same source and tolerance rows, so it cannot replace the threshold with an unlisted one or ask for a stream value outside the displayed window.

Finally use the transport and replay rows H and C to carry the same threshold under equivalent packet presentations. The witness remains a finite tail row rather than a countable choice of witnesses. □

Theorem 8515.4 (Modulus of Cauchy convergence Real handoff)**.** *For an accepted* $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ *carrier* $M = (S, D, T, R, E, H, C, P, N)$*, every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *consumer receives the stream source* S*, dyadic tolerance row* D*, tail-threshold schedule* T*, regular rational handoff* R*, Real seal* E*, and structural rows* H, C, P, N*. The handoff exports no selected host limit, quotient representative, countable-choice principle, or Real-completeness theorem beyond those displayed rows.*

Proof. Let a downstream consumer read the Real-facing output. The only available route to that output is the displayed replay row C, whose source is S, D, whose middle coordinate is T, and whose handoff rows are R, E.

The package row P records this same route as provenance for $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ and $\mathsf{CauchyCompletionOperator}^{\uparrow}$. It is not a limit selector, quotient map, or completeness principle.

Therefore every Real or completion read is still a read of the finite carrier coordinates. The excluded host limit, quotient representative, countable-choice row, and ambient completeness theorem are not carrier coordinates, so they cannot be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfCauchyConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfCauchyConvergence}^{\uparrow}$ packet is generated from a stream source, dyadic tolerance row, explicit tail-threshold schedule, regular rational handoff, Real seal, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfCauchyConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8515.1, Definition 8515.2, Definition 8515.3, and Definition 8515.4 seed the explicit modulus carrier, NameCert surface, tail witness route, and Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfCauchyConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host limits, quotient stream representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, tail-threshold coverage, regular rational handoff, Real seal preservation, transport stability, and replay stability for the finite modulus packet.

Chapter 8516

A Concrete Naming Certificate for $\mathsf{BishopRealFunctionSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealFunctionSpace}^{\uparrow}$ packet records a finite L10 real-function surface whose inputs and outputs are read only through displayed real-name rows. It sits beside the general function-space packet of chapter 7964, the continuous-map carrier of chapter 130, the terminal real-name seal of chapter 38, the finite window route of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance core of chapter 291. The packet is a finite Bishop-style real-function certificate; it is not a host set of all functions, a quotient equality principle for maps, a completeness theorem for $\mathsf{Real}^{\uparrow}$, a function-extensionality axiom, a selected limit operator, or a countable-choice schedule.

Definition 8516.1 (Bishop real-function-space carrier)**.** *A* $\mathsf{BishopRealFunctionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, T, G, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the source* $\mathsf{Real}^{\uparrow}$ *domain row, listing the displayed input real names on which the function packet may be inspected.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *row, listing the displayed output real names returned by the inspected function windows.*
3. *G is the finite graph-window row, binding each inspected source window in D to the corresponding target window in T.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the source and target real names used by G.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the windows in W before any terminal target seal is consumed.*
6. *Q is the local Bishop continuity row, requiring each graph-window read to pass through the same displayed source window, target window, and regular readback route.*
7. *H records componentwise* hsame *transport for D, T, G, W, R, Q and the structural rows below.*
8. *C records displayed* Cont *replay from a real-function request through D, T, G, W, R, Q, H.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealFunctionSpace}^{\uparrow}$, $\mathsf{RealFunctionSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealFunctionSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, G, W, R, Q, H, C, P, N$. It has no coordinate for host function extensionality, quotient equality of maps, an ambient function-space completion, selected pointwise limits, countable choice, real trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8516.2 (Bishop real-function-space NameCert obligations)**.** *For every accepted* BishopRealFunctionSpace *carrier $B = (D, T, G, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealFunctionSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *source and target* $\mathsf{Real}^{\uparrow}$ *exposure through D and T;*
2. *finite graph-window exposure through G;*
3. $\mathsf{StreamName}^{\uparrow}$ *scheduling through W;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;*
5. *Bishop continuity replay through Q over the same displayed windows;*
6. *componentwise* hsame *transport through H;*
7. *displayed* Cont *replay through C;*
8. Pkg *provenance through P and local naming through N.*

No accepted obligation row is stored outside $D, T, G, W, R, Q, H, C, P, N$.

Proof. Project the accepted carrier of Definition 8516.1. The Real-facing rows are D, T, the graph and observation rows are G, W, R, Q, and the structural rows are H, C, P, N.

The graph-window obligations are read before the terminal target row is available. A consumer first displays a source real-name row from D, then the graph-window row G, then the target real-name row T attached to that graph window.

The finite observation route is supplied by W, R, Q. The stream schedule W lists the inspected windows, the regular rational row R gives the readback, and the Bishop continuity row Q records that the same displayed source and target windows are used by the local continuity replay.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since no other coordinate occurs in the carrier, the listed obligation rows exhaust the local certificate surface. □

Theorem 8516.3 (Bishop real-function-space nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopRealFunctionSpace}^{\uparrow}$ *carrier factors through*

$$D, T, G, W, R, Q, H, C, P, N.$$

Thus a target $\mathsf{Real}^{\uparrow}$ *output row can be read only after the source* $\mathsf{Real}^{\uparrow}$ *domain row, graph-window row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Bishop continuity replay, componentwise transport, displayed replay, provenance, and local naming rows have been exposed. The packet exports no host function extensionality, quotient equality of maps, ambient function-space completion, selected pointwise limit, countable-choice schedule, real trichotomy, or completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Begin with an accepted carrier and a Real- or Completion-facing consumer request. Carrier projection exposes only the rows named in Definition 8516.1.

The request reaches the target real-name row through the graph route. The source row D and graph-window row G identify the inspected input window, while the target row T records the output real-name window tied to that graph entry.

The L10 readback then passes through W, R, Q. The $\mathsf{StreamName}^{\uparrow}$ schedule and $\mathsf{RegSeqRat}^{\uparrow}$ row make the inspected finite windows visible, and the Bishop continuity row replays the same source and target windows instead of replacing them by a host function.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. None of these rows introduces an equality principle for all functions, a quotient carrier, a selected limit, choice, trichotomy, or an ambient completeness coordinate. Hence every consumer read remains inside the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealFunctionSpace}^{\uparrow}$ consumer that reaches a target $\mathsf{Real}^{\uparrow}$ output without displaying $D, T, G, W, R, Q, H, C, P, N$ refutes the carrier surface.

Independence witness: real-function-space, continuousmap, real, streamname, regseqrat, dyadicratcore: no listed sibling supplies the whole packet $B = (D, T, G, W, R, Q, H, C, P, N)$, because each sibling contributes only a general function-space, continuous-map, real-name, stream-window, regular-readback, or dyadic-tolerance face while this chapter binds them as one finite Bishop real-function route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealFunctionSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealFunctionSpace}^{\uparrow}$ packet is generated from source and target RealUp rows, finite graph-window rows, StreamName windows, RegSeqRat readbacks, a Bishop continuity replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealFunctionSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8516.1, Definition 8516.2, and Definition 8516.3 seed the finite L10 real-function route over RealUp, StreamNameUp, RegSeqRatUp, ContinuousMapUp, RealFunctionSpaceUp, and DyadicRatCoreUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealFunctionSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of maps, ambient function-space completion, selected pointwise limits, countable choice, real trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph-window exactness, StreamName scheduling, RegSeqRat readback, Bishop continuity replay, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealFunctionSpac

Chapter 8517

A Concrete Naming Certificate for $\mathsf{UltrametricSkeleton}^{\uparrow}$

Following the metric embedding and ultrametric skeleton literature, the $\mathsf{UltrametricSkeleton}^{\uparrow}$ packet records a finite tree-scale approximation to displayed metric data as NameCert rows rather than as a host-selected infinite tree. It sits over the metric source of chapter 57, the doubling control route of chapter 2035, the finite epsilon-net surface of chapter 1767, and the dyadic tolerance ledger of chapter 291. The packet gives compactness, finite-net, entropy, and embedding-facing consumers a named finite ultrametric approximation object; it does not import quotient metric equality, open-cover compactness, a selected infinite hierarchy, or ambient completeness.

Definition 8517.1 (Ultrametric skeleton carrier)**.** *An* $\mathsf{UltrametricSkeleton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, E, T, D, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the displayed distance observations consumed by the skeleton.*
2. *E is the finite scale-net row, read through* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *at the displayed dyadic radii.*
3. *T is the cluster-tree row, recording the finite parent, child, and leaf incidence data induced by the scale-net inclusions.*
4. *D is the ultrametric distance ledger, assigning the displayed dyadic merge scale to each pair of retained leaves.*
5. *B is the distortion budget row, comparing source metric observations from M with ultrametric ledger entries from D.*
6. *H records componentwise* hsame *transport for the metric source, finite scale net, cluster tree, distance ledger, distortion budget, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a metric read through the scale-net row, cluster-tree row, ultrametric distance ledger, and distortion budget.*
8. *P is the* Pkg *provenance over* $\mathsf{UltrametricSkeleton}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, E, T, D, B, H, C, P, N$. It has no coordinate for a host quotient of metric spaces, a selected infinite cluster tree, open-cover compactness, a choice principle, or ambient metric completion.

Theorem 8517.2 (Ultrametric skeleton NameCert obligations)**.** *For every accepted* $\mathsf{UltrametricSkeleton}^{\uparrow}$ *carrier* $U = (M, E, T, D, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$ *surface consists exactly of metric-source exposure, finite scale-net exposure, cluster-tree incidence exposure, ultrametric distance ledger exposure, distortion-budget exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every embedding-facing or compactness-facing consumer must read the finite route*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{DoublingMetric}^{\uparrow},\quad \mathsf{FiniteEpsilonNet}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 8517.1. The only analytic rows are the metric source M, finite scale-net row E, cluster-tree incidence row T, ultrametric distance ledger D, and distortion budget B.

Read the finite approximation rows in order. The metric source supplies displayed distances, the finite scale-net row supplies retained representatives at dyadic radii, the cluster-tree row supplies only finite incidence data, and the ultrametric ledger assigns the merge scale recorded by that finite tree.

The distortion obligation is then exactly the row B. Componentwise transport through H and replay through C preserve the same finite rows, while provenance P and local naming N cite the packet surface.

Since the carrier has no coordinate for host choice, quotient metric equality, a selected infinite tree, open-cover compactness, or ambient completeness, none of those principles can be exported by the local $\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$ obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$.

Constructive story: An $\mathsf{UltrametricSkeleton}^{\uparrow}$ packet is generated from a Metric source row, finite scale-net rows, a finite cluster-tree incidence ledger, an ultrametric distance ledger, a distortion budget, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$, seed)

Scope closed: Definition 8517.1 and Definition 8517.2 seed the finite ultrametric approximation surface over metric, doubling, finite-net, dyadic, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UltrametricSkeleton}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host choice, quotient metric equality, selected infinite trees, open-cover compactness, ambient completeness, Lean verification, axiom-purity audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite scale-net exposure, cluster-tree incidence exactness, ultrametric ledger exactness, distortion-budget compatibility, transport, replay, provenance, and local naming for BEDC.Derived.UltrametricSkeletonUp.

Chapter 8518

A Concrete Naming Certificate for $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$

Following Bishop–Bridges constructive total boundedness and Kelley's filter and net language, the $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$ packet records the finite route by which a totally bounded source and a Cauchy filter or net source produce a completion-facing regular readback. It sits after the finite-net surface of chapter 127, the Cauchy-filter surface of chapter 333, the complete-metric seal of chapter 128, and the basis-first completion handoff of chapter 4050. The packet names a finite compactness-completion bridge only; it does not select a limit, identify quotient filters, use countable choice, or import ambient completeness outside the displayed rows.

Definition 8518.1 (Totally bounded Cauchy net carrier)**.** *A* $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, C, F, D, W, R, S, H, K, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{TotallyBounded}^{\uparrow}$ *source row, exposing the finite-net coverage surface used by the inspected metric source.*
2. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *or Cauchy-net source row, naming the directed observation route whose tails must be read through finite covers.*
3. *F is the finite net basis row, recording the selected cover indices, basis observations, and bundled centers displayed at each inspected tolerance.*
4. *D is the dyadic tolerance ledger row, storing the positive tolerance histories and refinement comparisons used by the finite basis.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which Cauchy observations and finite-net choices are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and dyadic tolerance data.*
7. *S is the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal row reached only after B, C, F, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the totally bounded source, Cauchy source, finite basis, tolerance ledger, windows, readback, seal, replay, provenance, and local naming.*
9. *K records displayed* Cont *replay from a Cauchy observation request through finite-net coverage, dyadic tolerance, stream-window inspection, regular rational readback, and the terminal seal.*

10. *P is the* Pkg *provenance over* TotallyBoundedCauchyNet$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CauchyFilter$^{\uparrow}$, CauchyFilterBasisCom
CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{TotallyBoundedCauchyNet}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite completion route. It has no coordinate for quotient completion, a choice-selected subnet, host metric compactness, selected limit representatives, hidden tolerance comparison, or ambient completeness outside S.

Theorem 8518.2 (Totally bounded Cauchy net NameCert obligations)**.** *For every accepted* TotallyBoundedCauchyI *carrier*

$$T = (B, C, F, D, W, R, S, H, K, P, N),$$

the local NameCert$_{\text{TotallyBoundedCauchyNet}^{\uparrow}}$ *surface consists exactly of totally bounded source exposure, Cauchy filter or net source exposure, finite net basis exposure, dyadic tolerance-ledger exposure, StreamName window exposure, RegSeqRat readback, CompleteMetric or Real seal handoff, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8518.1. The source rows are B and C, the finite compactness rows are F, D, the observation rows are W, R, and the seal and structural rows are S, H, K, P, N.

Read the totally bounded row before the Cauchy observation is consumed. The finite basis F is licensed by the finite-net coverage surface, while the Cauchy row C fixes which directed tails or filter observations must be covered.

Next inspect D, W, R. The dyadic ledger records the tolerance budget, the stream windows display the finite observations, and the regular rational readback records the completion-facing value for the same inspected windows.

Finally consume S, H, K, P, N. The seal is downstream of the totally bounded source, Cauchy source, basis, tolerance, window, and readback rows, and the structural rows preserve exactly that route. Since no carrier coordinate stores quotient completion, a selected subnet, host compactness, selected limits, hidden tolerance comparison, or detached completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8518.3 (Totally bounded Cauchy net filter handoff)**.** *Every* CauchyFilterBasisCompletion$^{\uparrow}$ *or* CompleteMetric$^{\uparrow}$ *consumer of an accepted* TotallyBoundedCauchyNet$^{\uparrow}$ *packet must first read*

$$B, C, F, D, W, R, S, H, K, P, N.$$

Thus a completion-facing Cauchy read from a totally bounded source passes through displayed finite-net coverage, Cauchy filter or net source data, dyadic tolerance control, StreamName windows, RegSeqRat readback, the CompleteMetric or Real seal, transport, replay, provenance, and local naming before any downstream completion result is consumed.

Proof. Start with Definition 8518.2. It requires the totally bounded source row B and the Cauchy source row C before the finite basis, tolerance, window, and readback rows can be read.

For a CauchyFilterBasisCompletion$^{\uparrow}$ consumer, the row C supplies the filter or net source and F, D, W, R supply the basis, tolerance, window, and readback data needed for the basis-first route of chapter 4050.

For a CompleteMetric$^{\uparrow}$ consumer, the seal row S records only the displayed completion-facing handoff. The provenance row may cite the complete-metric vocabulary, but it does not add a selected limit, quotient filter, or choice-selected subnet.

The exclusions follow from Definition 8518.1. Any downstream completion read must traverse the finite rows of T, so no hidden compactness or quotient-completion principle can be recovered from the packet. □

Falsifiable prediction: A claimed $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$ completion read is rejected unless the totally bounded source, Cauchy source, finite net basis, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, CompleteMetric or Real seal, transport, replay, provenance, and local NameCert rows are all displayed.

Independence witness: totallybounded, cauchyfilter, completemetric, cauchy-filter-basis-completion: none is bijective to $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$ because the packet jointly binds finite-net coverage, Cauchy filter or net source data, dyadic tolerance control, StreamName windows, RegSeqRat readback, completion seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{TotallyBoundedCauchyNet}^{\uparrow}$ packet is generated from TotallyBounded source rows, CauchyFilter or Cauchy-net source rows, finite net basis data, dyadic tolerance ledgers, StreamName windows, RegSeqRat readback, a CompleteMetric or Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8518.1, Definition 8518.2, and Definition 8518.3 seed the finite totally bounded Cauchy handoff to completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected subnets, host compactness, selected limit representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, Cauchy source admission, finite net basis exactness, dyadic tolerance exactness, StreamName window exactness, RegSeqRat readback, CompleteMetric or Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TotallyBoundedCauchyNetUp.

Chapter 8519

A Concrete Naming Certificate for $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$

The $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$ packet realizes the classifier-shift information principle of section 9462.2 as a finite gate for positiveDiscovery reads. It sits beside the classifier-channel shift ledger of chapter 6765, the pipeline discovery classifier of chapter 6931, the theorem-genome evolution packet of chapter 6358, and the theorem-DNA complexity scoring packet of chapter 7061. The packet accepts a discovery read only when classifier disagreement, witness flow, semantic certificate, public weight profile, refusal boundary, transport, replay, provenance, and local naming are displayed as one finite route.

Definition 8519.1 (Classifier-shift discovery gate carrier)**.** *A* $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, A, W, S, Q, R, T, C, P, N)$$

with the following displayed rows:

1. *B is the before-classifier row, naming the source scope and classifier relation before the inspected passage.*
2. *A is the after-classifier row, naming the same comparison surface after the inspected passage.*
3. *W is the witness-flow row, exposing a non-smoke path from source material to changed classifier judgment.*
4. *S is the semantic* $\mathsf{NameCert}$ *row, recording the certificate that licenses the changed classifier surface.*
5. *Q is the public weight-profile row used to distinguish positiveDiscovery from record growth or unchecked classifier movement.*
6. *R is the refusal row, recording failed, hidden, or ledger-incomplete claims that must remain visible.*
7. *T records componentwise* hsame *transport for classifier rows, witness flow, semantic certificate, weight profile, refusal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the before/after classifier comparison through witness flow, semantic certification, weight-profile reading, refusal inspection, and local naming.*

9. *P is the* Pkg *provenance over* ClassifierShiftDiscoveryGate$^{\uparrow}$, ClassifierChannelShiftLedger$^{\uparrow}$, PipelineDiscoveryClassif TheoremGenomeEvolution$^{\uparrow}$, TheoremDNAComplexityScoring$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for prose-only discovery, record count alone, hidden classifier comparison, smoke witness flow, missing semantic certificate, hidden weight profile, refusal erasure, or a local name outside N.

Theorem 8519.2 (Classifier-shift discovery gate NameCert obligations)**.** *For every accepted* ClassifierShiftDiscovery *carrier*

$$G = (B, A, W, S, Q, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}}$ *surface consists exactly of before-classifier exposure, after-classifier exposure, non-smoke witness-flow exposure, semantic-certificate exposure, public weight-profile exposure, refusal-row exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8519.1. The classifier comparison is carried by B, A, the evidence route by W, S, Q, R, and the structural preservation rows by T, C, P, N.

Read B and A first. A positiveDiscovery read cannot be formed before the before and after classifier rows are displayed over the same comparison surface.

Next inspect W, S, Q, R. The witness flow rules out smoke evidence, the semantic certificate licenses the changed classifier surface, the public weight profile records the positive reading, and the refusal row keeps rejected or incomplete claims visible.

Finally use T, C, P, N. These rows transport, replay, source, and name the same gate without adding hidden classifier comparison, hidden weights, or refusal erasure. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 8519.3 (Classifier-shift discovery gate ledger route)**.** *Every positiveDiscovery read from an accepted* ClassifierShiftDiscoveryGate$^{\uparrow}$ *packet factors through the classifier-channel ledger, pipeline discovery classifier, theorem-genome evolution route, and theorem-DNA complexity scoring surface before the local gate name is consumed. If any of before/after disagreement, witness flow, semantic certificate, weight profile, refusal row, transport, replay, provenance, or local naming is absent, the read is rejected by the displayed carrier.*

Proof. Start with Definition 8519.2. It places the before and after classifier rows, witness flow, semantic certificate, weight profile, refusal row, transport, replay, provenance, and local naming inside the same accepted packet.

Read the sibling route named in the introduction. The classifier-channel ledger supplies the before/after comparison surface, the pipeline discovery classifier supplies the positiveDiscovery boundary, the theorem-genome evolution packet supplies the theorem-flow route, and the theorem-DNA complexity scoring surface supplies the public weight profile.

The refusal and structural rows keep the same route auditable. Since no carrier coordinate supplies prose-only discovery, record count alone, hidden comparison, smoke evidence, hidden weights, or refusal erasure, omitting any displayed row prevents the local gate name from being consumed as a positiveDiscovery read. □

Falsifiable prediction: A claimed positive discovery read is rejected unless before/after classifier disagreement, non-smoke witness flow, semantic NameCert row, classifier-gain row, weight profile, refusal row, transport, replay, provenance, and local naming are all displayed.

Independence witness: theorem-genome-evolution, pipeline-discovery-classifier, classifier-channel-shift-ledger, theorem-dna-complexity-scoring: none is bijective to $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$ because the gate jointly binds before/after classifier disagreement, witness flow, semantic certificate, weight profile, refusal, transport, replay, provenance, and local naming.

The Lean follow-up for $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ClassifierShiftDiscoveryGate}^{\uparrow}$ and $\mathsf{FieldFaithful\ ClassifierShiftDiscoveryGate}^{\uparrow}$ instances in `lean4/BEDC/Derived/ClassifierShiftDiscoveryGate` The field-faithfulness witness must preserve $B, A, W, S, Q, R, T, C, P, N$ as distinct rows before a checked gate theorem can cite the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}$ packet is generated from before-classifier rows, after-classifier rows, non-smoke witness flow, semantic NameCert evidence, public weight-profile rows, refusal rows, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8519.1, Definition 8519.2, and Definition 8519.3 seed the finite gate for classifier-shift discovery reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierShiftDiscoveryGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global discovery certification, hidden semantic novelty scoring, automated theorem proving, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked before/after classifier disagreement, witness-flow non-smoke exposure, semantic certificate exposure, public weight-profile exposure, refusal preservation, transport, replay, provenance, local naming, Nontrivial, and FieldFaithful rows for BEDC.Derived.ClassifierShiftDiscoveryGateUp.

Chapter 8520

A Concrete Naming Certificate for $\mathsf{DyadicIntervalHalving}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicIntervalHalving}^{\uparrow}$ packet records one inspected bisection step for a dyadic interval as finite NameCert data. It sits beside the dyadic arithmetic core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies downstream nested interval, interval cover, and regular Cauchy completion consumers with a local halving row; it does not smuggle open-cover compactness, quotient real equality, trichotomy, an unbounded midpoint selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The sibling route is deliberately local. $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the endpoint arithmetic and width ledger, $\mathsf{StreamName}^{\uparrow}$ supplies the finite observation window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback boundary, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal only after the displayed dyadic row has been inspected.

Definition 8520.1 (Dyadic interval halving carrier)**.** *A* $\mathsf{DyadicIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, D, M, L, R, W, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *I is the input dyadic interval row, recording left endpoint, right endpoint, and the displayed nonnegative width witness.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row for endpoint comparison, midpoint formation, and dyadic denominator tracking.*
3. *M is the midpoint row computed from the displayed endpoints through D.*
4. *L is the left-child row with endpoints given by the input left endpoint and M.*
5. *R is the right-child row with endpoints given by M and the input right endpoint.*
6. *W is the width-halving row, exposing the equality of the two child widths with one displayed dyadic half of the input width.*
7. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for reading endpoints, midpoint, and child widths.*
8. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row formed from Q and D.*
9. *T is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row available only after I, D, M, L, R, W, Q, E have been displayed.*

10. *C records componentwise* hsame *transport for interval, arithmetic, midpoint, child, width, window, readback, seal, replay, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalHalving}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}$ *row.*

The carrier exports exactly one dyadic halving step. It has no coordinate for an infinite branch, open-cover compactness, quotient real equality, trichotomy, selected midpoint choice beyond the displayed dyadic row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalHalvingUp.DyadicIntervalHalvingTasteGate single carrier alignment`

Theorem 8520.2 (Dyadic interval halving NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalHalving}^{\uparrow}$ *carrier*

$$H = (I, D, M, L, R, W, Q, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}$ *surface consists exactly of input interval exposure, dyadic arithmetic exposure, midpoint exposure, left-child exposure, right-child exposure, width-halving exposure, finite-window scheduling, regular rational readback, terminal Real sealing, componentwise transport, package provenance, and local naming.*

Proof. Project the carrier of Definition 8520.1. The rows I and D expose the input interval and the dyadic arithmetic ledger before any child interval is read.

The midpoint and child rows are M, L, R. They are computed from the displayed dyadic endpoint data, and the width row W records the local halving equality for those two children.

The completion-facing route is downstream from the halving rows. The finite window Q is read before the regular rational row E, and the terminal Real seal T is available only after that readback. The rows C, P, N transport, source, and name the same finite route, so the listed rows exhaust the local certificate. □

Theorem 8520.3 (Dyadic interval halving nested route handoff)**.** *Every nested interval consumer of an accepted* $\mathsf{DyadicIntervalHalving}^{\uparrow}$ *carrier must read the handoff through*

$$I, D, M, L, R, W, Q, E, T, C, P, N.$$

The packet offers one displayed dyadic bisection step that can be appended to a finite nested route, but it does not choose an infinite branch, assert interval compactness, or identify quotient real names.

Proof. Begin with the displayed input interval row I and arithmetic row D. These rows determine the endpoint comparison and midpoint formation visible to a later nested-route consumer.

The child rows L and R are both present as finite data, with W recording the halved width relation. A consumer may retain either displayed child in its own finite route, but the carrier itself does not provide an infinite selector.

The rows Q, E, T then expose the observation, rational readback, and Real-facing boundary for that same one-step bisection. Since the structural rows C, P, N preserve only this finite packet, the handoff remains a local nested-route input rather than a compactness or quotient-real principle. □

Theorem 8520.4 (Dyadic interval halving Real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{DyadicIntervalHalving}^{\uparrow}$ *carrier is readable only after the dyadic endpoint row, midpoint row, child interval rows, width-halving row, finite observation row, and regular rational readback row have all been displayed. In particular, the seal does not create ambient real completeness, quotient equality of real names, trichotomy, or open-cover compactness.*

Proof. Read the carrier in the order prescribed by Definition 8520.1. The input endpoint row I, arithmetic row D, midpoint row M, child rows L, R, and width row W provide the whole dyadic halving surface.

The observation row Q and regular rational row E translate that finite surface into the completion-facing readback route. Only then is the row T exposed as the terminal $\mathsf{Real}^{\uparrow}$ seal boundary.

No remaining coordinate stores ambient completeness, quotient equality, trichotomy, or compactness. The seal therefore records only the displayed one-step dyadic interval readback. □

Theorem 8520.5 (Dyadic interval halving scoped bisection transport)**.** *Every accepted* $\mathsf{DyadicIntervalHalving}^{\uparrow}$ *bisection read transports componentwise through the displayed dyadic endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}$ *row. The packet exports no open-cover compactness, quotient* $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, unbounded midpoint selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 8520.2. They expose the input interval row, dyadic arithmetic row, midpoint row, child rows, width-halving row, finite observation row, regular rational readback row, Real seal row, transport row, provenance row, and local naming row as one finite carrier.

Next use the nested route handoff of Definition 8520.3. It permits a consumer to append the displayed left or right child only as a one-step bisection input, with no infinite branch selector or compactness principle attached.

Finally apply the Real seal boundary of Definition 8520.4. Since the seal is readable only after the endpoint, midpoint, child, width, observation, and readback rows have been displayed, componentwise hsame transport and Cont replay preserve exactly that scoped route. The carrier has no coordinate for open-cover compactness, quotient real equality, trichotomy, an unbounded midpoint selector, or ambient completeness, so none is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalHalving}^{\uparrow}$ packet is generated from one input dyadic interval row, DyadicRatCore arithmetic rows, a midpoint row, two child interval rows, a width-halving row, StreamName finite observations, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8520.1, Definition 8520.2, Definition 8520.3, and Definition 8520.4 seed the one-step dyadic interval bisection packet over arithmetic, observation, readback, seal, transport, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHalving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient real equality, trichotomy, selected midpoint choice beyond the displayed dyadic row, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic arithmetic exposure, midpoint formation, child-row exposure, width-halving exactness, finite-window scheduling, regular rational readback, Real seal boundary control, transport, provenance, and local naming for BEDC.Derived.DyadicIntervalHalvingUp.

Chapter 8521

A Concrete Naming Certificate for $\mathsf{HyperbolicHorosphere}^{\uparrow}$

Following classical hyperbolic geometry, the $\mathsf{HyperbolicHorosphere}^{\uparrow}$ packet realizes the level-set side of the observer-expansion vision of chapter 9463 as finite NameCert data. It sits after the metric source of chapter 3474, the Busemann boundary route of chapter 3732, the terminal real-name seal of chapter 38, and the metric carrier of chapter 57. The packet names a displayed boundary point and Busemann level surface; it does not introduce a host ideal boundary quotient, a completed horoball compactification, a selected geodesic representative, or a physical observer trajectory.

Definition 8521.1 (Hyperbolic horosphere carrier)**.** *A* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, B, L, R, O, E, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row for the source observations.*
2. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-point and Busemann-read row.*
3. *L is the displayed Busemann level row naming the horosphere slice.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *level-seal row used to compare the displayed Busemann value with L.*
5. *O is the observer-window row, recording which finite metric and Busemann observations are available.*
6. *E is the non-escape row excluding ideal-boundary quotient equality, completed horoball compactification, selected geodesic representatives, and physical observer motion.*
7. *T is the transport row preserving the boundary point, Busemann level, metric readback, observer window, and Real seal under componentwise readback.*
8. *H records componentwise* hsame *transport for metric, boundary, level, seal, window, non-escape, transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through metric admission, Busemann boundary readback, level comparison, observer-window inspection, Real sealing, and non-escape.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHorosphere}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorosphere}^{\uparrow}}$ *row.*
The carrier exports horosphere data only through $M, B, L, R, O, E, T, H, C, P, N$. It contains no coordinate for a host boundary quotient, a global foliation of hyperbolic space, a completed compactification, a selected geodesic ray, or an observer-substance reading.

Theorem 8521.2 (Hyperbolic horosphere NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *carrier*

$$S = (M, B, L, R, O, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorosphere}^{\uparrow}}$ *surface consists exactly of hyperbolic metric admission, Busemann boundary admission, displayed level exposure, Real seal comparison, observer-window exposure, non-escape refusal, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8521.1. The metric row M and Busemann boundary row B expose the source and boundary data before any horosphere read is made.

The rows L and R form the level surface: L names the displayed Busemann level, while R gives the Real-facing comparison boundary for that level. The observer-window row O records the finite observations available to a consumer.

The refusal and structural rows E, T, H, C, P, N preserve this same route and exclude boundary quotients, compactification, selected geodesics, and observer-substance readings. Hence the displayed rows exhaust the local NameCert obligations. □

Theorem 8521.3 (Hyperbolic horosphere Busemann route)**.** *Every public read from an accepted* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *carrier to a Busemann-level statement factors through the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row M, the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row B, the displayed level row L, the* $\mathsf{Real}^{\uparrow}$ *seal row R, and the observer-window row O before the transport, replay, provenance, and naming rows are consumed.*

Proof. Read the metric and Busemann source rows first. The metric read M supplies the finite hyperbolic distance observations, and B supplies the boundary point together with the Busemann readback route.

The level row L is interpreted only through the Real seal row R and the observer-window row O. This makes the horosphere a displayed level surface rather than a quotient boundary object.

The rows T, H, C, P, N then transport, replay, source, and name the same Busemann-level route. Since the carrier contains no global compactification or selected geodesic coordinate, every Busemann read must factor through the displayed rows. □

Theorem 8521.4 (Hyperbolic horosphere non-escape)**.** *An accepted* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *carrier cannot export host ideal-boundary quotient equality, completed horoball compactification, a selected geodesic representative, global horosphere foliation, or physical observer motion. All exported data are confined to*

$$M, B, L, R, O, E, T, H, C, P, N.$$

Proof. Inspect the carrier inventory of Definition 8521.1. The only geometric source rows are the metric row M, the Busemann boundary row B, the level row L, the Real seal row R, and the observer-window row O.

The non-escape row E records the excluded host readings, while T, H, C, P, N preserve only the displayed finite route. No carrier coordinate names a boundary quotient, compactification, selected geodesic, global foliation, or observer motion.

Therefore a consumer can extract only the finite horosphere rows and their replay/provenance data. The excluded host objects have no row from which they can be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorosphere}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHorosphere}^{\uparrow}$ packet is generated from HyperbolicMetric metric rows, HyperbolicBusemannBoundary boundary and Busemann rows, a displayed Busemann level, RealUp level sealing, observer-window rows, non-escape refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHorosphere}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8521.1, Definition 8521.2, Definition 8521.3, and Definition 8521.4 seed the finite horosphere certificate over metric, boundary, Busemann level, Real seal, observer-window, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHorosphere}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ideal-boundary quotient equality, completed horoball compactification, selected geodesic representatives, global horosphere foliations, physical observer motion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric admission, Busemann boundary admission, level-row exposure, Real seal comparison, observer-window exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicHorosphereUp.

Chapter 8522

A Concrete Naming Certificate for $\mathsf{MacShaneExtension}^{\uparrow}$

Following McShane's extension theorem, the $\mathsf{MacShaneExtension}^{\uparrow}$ packet records the finite BEDC certificate by which a displayed real-valued Lipschitz map on a located subcarrier of a displayed metric space is read as an ambient Lipschitz envelope. It consumes the $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ carrier surfaces of chapter 57 and chapter 38, the $\mathsf{ContinuousMap}^{\uparrow}$ replay discipline of chapter 130, the $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ bound route of chapter 3812, and the $\mathsf{LocatedUniformSpace}^{\uparrow}$ located-uniform boundary of chapter 3814. The packet names a finite envelope certificate, not a host formula for arbitrary extrema or a quotient function over completed reals.

Definition 8522.1 (MacShaneExtension carrier)**.** *A* $\mathsf{MacShaneExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, A, d, f, L, E, S, R, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *ambient carrier row, exposing only the metric observations used by the envelope request.*
2. *A is the located source row, recording the admitted subcarrier points and the finite inclusion evidence read by ambient comparisons.*
3. *d is the* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *compatibility row that keeps source and ambient distance reads on the same located-uniform route.*
4. *f is the* $\mathsf{Real}^{\uparrow}$ *source-value graph row for the displayed real-valued map on A.*
5. *L is the* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ *bound row, recording the displayed Lipschitz constant and the finite tolerance handoff used by the envelope.*
6. *E is the MacShane envelope row, storing finite comparison data of the form source value plus bound times ambient distance, without selecting an arbitrary infimum operator.*
7. *S is the source-agreement row, requiring every source-admitted ambient read to return through A, d, f, L, E before it is identified with the original source read.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for source values, envelope comparisons, and ambient extension reads.*
9. *U is the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row that records the continuity-facing readback route after the Lipschitz handoff has been displayed.*

10. *H records componentwise* hsame *transport for the carrier, source, metric, graph, bound, envelope, agreement, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a precision request through $X, A, d, f, L, E, S, R, U, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{MacShaneExtension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$, $\mathsf{LocatedUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MacShaneExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these thirteen rows. It has no coordinate for host choice, quotient real equality, arbitrary infimum operators, selected extrema, countable choice, trichotomy, or ambient completeness.

Theorem 8522.2 (MacShaneExtension NameCert obligations)**.** *For every accepted* $\mathsf{MacShaneExtension}^{\uparrow}$ *carrier $M = (X, A, d, f, L, E, S, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MacShaneExtension}^{\uparrow}}$ *surface consists of the following finite obligation rows:*

1. *carrier admission for the ambient metric-space row, located source row, located-uniform distance row, source-value graph, Lipschitz handoff, envelope, agreement, Real seal, continuity replay, transport, provenance, and local naming rows;*
2. *domain compatibility, requiring every source-value read to pass through A and d before comparison with an ambient envelope read;*
3. *bound transport, requiring L to mediate every tolerance and distance-to-value estimate consumed by E;*
4. *envelope exactness, requiring E to store finite MacShane comparison data and no host choice, quotient real equality, selected extremum, or arbitrary infimum operator;*
5. *source-agreement nonescape, requiring S, R, U, H, C, P, N to preserve the displayed route without adding a host function object, a quotient function, or an ambient completeness witness.*

No local obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 8522.1. The accepted packet displays exactly the ambient metric-space, located source, located-uniform distance, source graph, Lipschitz handoff, envelope, agreement, Real seal, continuity replay, and structural rows.

Read the domain and bound clauses next. The rows A and d provide the source admission and ambient distance route, while f and L provide the source value and the displayed Lipschitz constant. Thus an envelope comparison cannot be formed before the carrier has exposed the same finite metric and Real rows used by the source map.

Project the envelope row E. It stores finite MacShane comparisons derived from displayed source values and displayed distance bounds, so it can be replayed for a concrete ambient request without selecting a global extremum or quotient real representative.

Finally inspect S, R, U, H, C, P, N. Agreement, Real sealing, continuity replay, transport, provenance, and local naming only preserve the displayed rows. Therefore the NameCert surface is exactly the listed five-row obligation sketch, and no host choice, quotient real equality, arbitrary infimum operator, selected extremum, or ambient completeness principle is introduced. □

Theorem 8522.3 (MacShaneExtension Lipschitz envelope)**.** *Let $M = (X, A, d, f, L, E, S, R, U, H, C, P, N)$ be an accepted* $\mathsf{MacShaneExtension}^{\uparrow}$ *carrier. Every ambient envelope read exported by M factors through the finite route*

$$X, A, d \longmapsto f \longmapsto L \longmapsto E \longmapsto S, R, U \longmapsto H, C, P, N.$$

Along that route the displayed Lipschitz bound remains the only value-growth control used by the ambient read. The packet exports a finite Lipschitz-envelope certificate and no host choice, quotient real equality, arbitrary infimum operator, selected extremum, countable-choice selector, trichotomy branch, or ambient completeness witness.

Proof. Start with an ambient envelope request. The rows X, A, d expose the only metric-space, source-admission, and located-uniform distance information available to the read.

Read the source graph and bound handoff. The row f supplies the displayed Real source values, while L supplies the Lipschitz control that turns distance information into value estimates. This is the only route by which a bound can enter the envelope.

The envelope row E then records finite comparisons for the requested ambient input. The source-agreement and Real-seal rows S, R keep those comparisons tied to the source route, and the continuity replay row U exposes only the same Lipschitz-facing handoff to continuous-map consumers.

Transport, replay, provenance, and local naming preserve the route through H, C, P, N. Since no listed row stores host choice, quotient real equality, an arbitrary infimum operator, a selected extremum, countable choice, trichotomy, or ambient completeness, the exported object is exactly the finite Lipschitz-envelope certificate. □

Falsifiable prediction: An accepted $\mathsf{MacShaneExtension}^{\uparrow}$ consumer that reaches an ambient envelope read without first reading the ambient $\mathsf{Met}^{\uparrow}$ row, the located source row, the $\mathsf{Real}^{\uparrow}$ source graph, the $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ bound row, and the MacShane envelope row refutes the carrier surface.

Independence witness: metric-space, real, continuousmap, lipschitz-continuity-handoff, located-uniform-space: no listed sibling supplies the full thirteen-row packet $M = (X, A, d, f, L, E, S, R, U, H, C, P, N)$, because each sibling contributes only a metric carrier, Real seal, continuity replay, bound handoff, or located-uniform distance boundary while this chapter binds the finite MacShane envelope certificate into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MacShaneExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{MacShaneExtension}^{\uparrow}$ packet is generated from an ambient MetricSpaceUp row, a located source row, a LocatedUniformSpaceUp distance row, a RealUp source graph, a LipschitzContinuityHandoffUp bound row, a finite envelope row, a source-agreement row, a Real seal, ContinuousMapUp replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MacShaneExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8522.1, Definition 8522.2, and Definition 8522.3 seed the finite MacShane Lipschitz-envelope certificate over displayed metric-space, located-source, Real, continuity, Lipschitz-handoff, and located-uniform data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MacShaneExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, quotient real equality, arbitrary infimum operators, selected extrema, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, domain compatibility, bound transport, envelope exactness, source-agreement nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.MacShaneExtensionUp.

Chapter 8523

A Concrete Naming Certificate for $\mathsf{ControllabilityRank}^{\uparrow}$

Following Kalman 1960 on controllability rank conditions, the $\mathsf{ControllabilityRank}^{\uparrow}$ packet records the finite rank-window certificate that sits between the reachability ledger of chapter 297 and the finite matrix-pencil consumers of chapter 8240. It names the displayed columns, dimension rows, rank-window comparisons, transport, replay, provenance, and local naming data used by finite-dimensional control reads. The packet is not a host reachable subspace, not a quotient state space, not a Kalman decomposition, and not an infinite-dimensional controllability theorem.

Definition 8523.1 (ControllabilityRank carrier)**.** *A* $\mathsf{ControllabilityRank}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, U, A, B, W, V, L, R, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite state* $\mathsf{VecSpace}^{\uparrow}$ *row and U is the finite input* $\mathsf{VecSpace}^{\uparrow}$ *row.*
2. *A is the transition* $\mathsf{Mat}^{\uparrow}$ *row and B is the input* $\mathsf{Mat}^{\uparrow}$ *row, both read as* $\mathsf{LinMap}^{\uparrow}$ *data when the reachability columns are generated.*
3. *W is the finite rank-window row listing the displayed columns $B, AB, \ldots, A^{n-1}B$ and the row-reduction comparisons used to test whether the window spans the displayed state dimension.*
4. *V is the vector-space dimension row against which the rank window is measured.*
5. *L is the* $\mathsf{LinMap}^{\uparrow}$ *reachability-iterate ledger recording each multiplication and column append route.*
6. *R is the* $\mathsf{ControlControllability}^{\uparrow}$ *reachability ledger consumed by downstream controllability readers.*
7. *O is the* $\mathsf{ControlObservability}^{\uparrow}$ *dual-window row available to dual tests and observability-facing consumers.*
8. *H records componentwise* hsame *transport for state rows, input rows, matrix rows, reachability iterates, rank windows, dimension rows, dual windows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from the plant rows through the reachability iterates into the rank-window certificate.*

10. *P is the* Pkg *provenance over* $\mathsf{ControllabilityRank}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{HautusTest}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ControllabilityRank}^{\uparrow}}$ *row.*

The carrier exports only this finite rank-window surface. It has no coordinate for a host rank theorem detached from $W, V, L, R, O, H, C, P, N$, and it has no quotient subspace through which reachability could be identified externally.

Theorem 8523.2 (ControllabilityRank NameCert obligations)**.** *For every accepted* $\mathsf{ControllabilityRank}^{\uparrow}$ *carrier*

$$K = (X, U, A, B, W, V, L, R, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ControllabilityRank}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, finite rank-window exactness for W, matrix and vector-space readback through A, B, X, U, V, reachability-ledger exactness through L and R, dual-window exposure through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. The certificate never escapes to a host rank predicate, a quotient state subspace, a Kalman decomposition, a spectral test outside the Hautus consumer row, or an infinite-dimensional controllability claim.*

Proof. Carrier admission is projection from Definition 8523.1. The accepted packet displays the finite state and input rows, transition and input matrices, rank-window row, dimension row, reachability iterates, controllability ledger, dual observability window, transport, replay, provenance, and local name.

Finite rank-window exactness is the row W measured against V. The matrix and vector-space readback rows A, B, X, U, V are already visible before W can be consumed, and the reachability-iterate ledger L records how each displayed column is generated.

The downstream control surface is R, O together with the $\mathsf{HautusTest}^{\uparrow}$ provenance named by P. A controllability consumer reads R, an observability or dual-test consumer reads O, and a Hautus consumer may inspect the same finite rank-window data without receiving an external spectral decomposition.

Finally transport and replay are exactly H and C. Since the carrier contains no host rank predicate, quotient state subspace, Kalman decomposition object, or infinite-dimensional theorem, the local NameCert obligations are exhausted by the finite packet K. □

Falsifiable prediction: An accepted $\mathsf{ControllabilityRank}^{\uparrow}$ packet is refuted by any downstream control consumer that reads a controllability rank result while bypassing the displayed rank-window row W, the dimension row V, the reachability-iterate ledger L, or the $\mathsf{ControlControllability}^{\uparrow}$ ledger R.

Independence witness: Nearest siblings: controlcontrollability, controlobservability, hautustest, matrix, linearmap, and vecspace; none is bijective to $\mathsf{ControllabilityRank}^{\uparrow}$ because this packet keeps the rank-window row, dimension readback, reachability-iterate ledger, dual-window row, Hautus provenance, transport, replay, provenance, and local NameCert row in one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ControllabilityRank}^{\uparrow}}$.

Constructive story: A $\mathsf{ControllabilityRank}^{\uparrow}$ packet is generated from finite state and input rows, transition and input matrix rows, a finite rank-window row, a vector-space dimension row, reachability-iterate rows, controllability and observability ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ControllabilityRank}^{\uparrow}}$, seed)

Scope closed: Definition 8523.1, Definition 8523.2, the displayed falsifiable prediction, and the displayed independence witness seed the finite Kalman rank-window certificate over matrix, vector-space, linear-map, controllability, observability, and Hautus rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ControllabilityRank}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host reachable-subspace equality, quotient state-space rank, Kalman decomposition, spectral completeness, analytic trajectory semantics, infinite-dimensional controllability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite rank-window exactness, matrix and vector-space readback, reachability-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.ControllabilityRankUp.

Chapter 8524

A Concrete Naming Certificate for $\mathsf{BeckChevalley}^{\uparrow}$

Following Beck and Chevalley on base-change coherence in categorical logic, the $\mathsf{BeckChevalley}^{\uparrow}$ packet records the finite NameCert surface by which a pullback square and its adjoint/Kan-extension data are allowed to expose a comparison map. It sits after the categorical carrier of chapter 61, the functorial transport surface of chapter 62, the natural-transformation component surface of chapter 63, the adjunction ledger of chapter 107, and the Kan-extension handoff of chapter 3927. The packet names only the displayed square, adjoint rows, Kan-extension route, comparison 2-cell, transport, replay, provenance, and local NameCert data; it does not assert a host indexed-category theorem, ambient exactness, or quotient equality of functor categories.

Definition 8524.1 (Beck–Chevalley carrier)**.** *A* $\mathsf{BeckChevalley}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, C, L, R, K, T, H, E, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{Cat}^{\uparrow}$ *and target* $\mathsf{Cat}^{\uparrow}$ *square row, displaying the four corner categories and the four boundary functors of the pullback-shaped diagram.*
2. *C is the* $\mathsf{Pullback}^{\uparrow}$ *comparison row, recording that the square is consumed only as displayed finite square data.*
3. *L records the left-adjoint rows along the two vertical sides, including their unit and counit component ledgers.*
4. *R records the right-adjoint rows along the two horizontal sides, including their unit and counit component ledgers.*
5. *K is the* $\mathsf{KanExtension}^{\uparrow}$ *handoff row, exposing the extension candidate and its universal comparison route only after A, C, L, R have been read.*
6. *T is the* $\mathsf{NatTrans}^{\uparrow}$ *comparison 2-cell row between the two boundary composites around the square.*
7. *H records componentwise* hsame *transport for the square, adjoint rows, Kan-extension handoff, comparison 2-cell, replay, provenance, and local naming rows.*
8. *E records displayed* Cont *replay through*

$$A, C \longmapsto L, R \longmapsto K \longmapsto T \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{BeckChevalley}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{Adjunction}^{\uparrow}$, $\mathsf{KanExtension}^{\uparrow}$, $\mathsf{Pullback}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BeckChevalley}^{\uparrow}}$ *row.*

The classifier accepts only packets whose comparison is exhausted by $A, C, L, R, K, T, H, E, P, N$. It has no coordinate for a host pullback theorem, an ambient indexed category, equality of functor categories, selected Kan-extension witnesses, or global exactness.

Theorem 8524.2 (Beck–Chevalley NameCert obligations)**.** *For every accepted* $\mathsf{BeckChevalley}^{\uparrow}$ *carrier $B = (A, C, L, R, K, T, H, E, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BeckChevalley}^{\uparrow}}$ *surface consists exactly of square admission through A, pullback-shaped comparison through C, left-adjoint exposure through L, right-adjoint exposure through R, Kan-extension handoff through K, comparison 2-cell exposure through T, componentwise transport through H, displayed replay through E, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8524.1. The square rows A, C are read before any adjoint or Kan-extension data, so the packet exposes only displayed base-change geometry.

The adjoint rows L, R are then read as finite unit and counit ledgers. They do not create a host adjunction theorem; they only make the two sides of the square available to the later comparison.

Finally K, T, H, E, P, N provide the Kan-extension handoff, comparison 2-cell, transport, replay, provenance, and local naming. Since the carrier contains no coordinate for ambient functor-category equality, selected universal witnesses, or global exactness, no extra NameCert obligation is available outside the listed rows. □

Theorem 8524.3 (Beck–Chevalley square route)**.** *Every accepted* $\mathsf{BeckChevalley}^{\uparrow}$ *read factors the base-change comparison through the finite route*

$$\mathsf{Cat}^{\uparrow}, \mathsf{Functor}^{\uparrow}, \mathsf{Pullback}^{\uparrow} \longmapsto \mathsf{Adjunction}^{\uparrow} \longmapsto \mathsf{KanExtension}^{\uparrow} \longmapsto \mathsf{NatTrans}^{\uparrow}.$$

Thus a consumer may read the comparison 2-cell only after the square, adjoint rows, and Kan-extension handoff have been displayed, and only as a local NameCert route over $A, C, L, R, K, T, H, E, P, N$.

Proof. Start from the carrier rows A and C. They display the four corners, boundary functors, and pullback-shaped square data before any comparison 2-cell can be consumed.

Next read the adjoint and Kan-extension rows. The rows L and R expose the unit and counit ledgers needed by the base-change route, while K makes the extension handoff available only downstream of those rows.

The comparison 2-cell row T is therefore a terminal local read of the displayed route. The structural rows H, E, P, N only transport, replay, source, and name that same finite packet, so the route remains inside BEDC certificate data and does not import an ambient Beck–Chevalley theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BeckChevalley}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BeckChevalley}^{\uparrow}$ packet is generated from displayed category and functor square rows, a pullback-shaped comparison row, left and right adjoint ledgers, a Kan-extension handoff, a comparison 2-cell, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BeckChevalley}^{\uparrow}}$, seed)

Scope closed: Definition 8524.1, Definition 8524.2, and Definition 8524.3 seed the finite base-change comparison surface over $A, C, L, R, K, T, H, E, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BeckChevalley}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host indexed-category semantics, ambient exactness, equality of functor categories, selected Kan-extension witnesses, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked square admission, pullback-shaped comparison, adjoint exposure, Kan-extension handoff, comparison 2-cell exposure, transport, replay, provenance, and local naming for BEDC.Derived.BeckChevalleyUp.

Chapter 8525

A Concrete Naming Certificate for BoundedRegularCauchyCluster$^{\uparrow}$

Following Bishop and Bridges, the BoundedRegularCauchyCluster$^{\uparrow}$ packet records the bounded regular-Cauchy cluster route as finite NameCert data rather than as a completed real quotient or a chosen convergent subsequence. It sits on the L10 Real and Completion spine: bounded source evidence is read through the finite window interface of chapter 126, dyadic tail budgets through chapter 291, regular rational readback through chapter 301, displayed extraction through chapter 5125, and the terminal seal through chapter 38.

Definition 8525.1 (BoundedRegularCauchyCluster carrier)**.** *A* BoundedRegularCauchyCluster$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, R, E, X, H, C, P, N)$$

with the following visible rows:

1. *S is the bounded source row, recording the displayed lower and upper rational bounds for the inspected regular-Cauchy source.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row through which the source is observed.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tail-budget row used to compare requested tolerances with the displayed windows.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to W and D.*
5. *E is the* CauchySubsequenceExtractor$^{\uparrow}$ *row, exposing the finite extraction schedule consumed by cluster reads.*
6. *X is the terminal* Real$^{\uparrow}$ *seal reached only after the* RegSeqRat$^{\uparrow}$ *readback and the displayed extraction row have been read.*
7. *H records componentwise* hsame *transport for the source bounds, window row, tail-budget row, regular-rational readback, extraction row, and real seal.*
8. *C records the displayed* Cont *replay routes by which Real-facing and completion-facing consumers read the packet.*
9. *P is the* Pkg *provenance over* BoundedRegularCauchyCluster$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchySubsequenceExtractor$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchyCluster}^{\uparrow}}$ *row.*
The carrier exports no countable-choice selector, no quotient equality of Cauchy names, no host real equality, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Theorem 8525.2 (BoundedRegularCauchyCluster NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegularCauchyCluster}^{\uparrow}$ *carrier* $B = (S, W, D, R, E, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchyClust}}$ *surface consists exactly of carrier admission for the bounded source row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. The classifier is stable only through H, and the ledger admits no selected subsequence outside E, no quotient real equality, no countable choice, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted read through Definition 8525.1. The bounded source, finite window, dyadic tail budget, regular-rational readback, extraction row, terminal real seal, transport, replay, provenance, and naming rows are all visible coordinates of the same packet.

Classifier stability is componentwise because the only transport coordinate is H. The continuation row C can replay the displayed source, readback, extraction, and Real-facing routes, but it has no coordinate for a selector or quotient object.

Thus the local certificate is exhausted by the displayed finite carrier. The excluded rows cannot be recovered from the packet because no coordinate stores them. □

Theorem 8525.3 (BoundedRegularCauchyCluster RegSeqRat handoff)**.** *Every cluster read accepted by a* $\mathsf{BoundedRegularCauchyCluster}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *row E. In particular, a Real-facing consumer must first read the bounded source row S, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row D, and the regular-rational readback row R before it may consume the extraction row E and the seal X.*

Proof. Read the carrier coordinates in order. The real seal X is stored as the terminal row of the same packet that contains S, W, D, R, E, and the displayed replay row C exposes no route to X that bypasses the regular-rational readback.

The extraction row E is not an independent choice functional. It is a finite row of the carrier, transported by H and replayed by C after the window and tail-budget rows have already fixed the RegSeqRat handoff.

Therefore every Real-facing cluster read factors through the displayed S, W, D, R, E, X route. The packet contains no quotient equality of streams, selected limit, countable-choice schedule, or ambient completeness row that could supply another route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchyCluster}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedRegularCauchyCluster}^{\uparrow}$ packet is generated from bounded source rows, finite StreamName windows, DyadicRatCore tail budgets, RegSeqRat readback, CauchySubsequenceExtractor rows, a Real seal, componentwise hsame transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedRegularCauchyCluster}^{\uparrow}}$, seed)

Scope closed: Definition 8525.1, Definition 8525.2, and Definition 8525.3 seed the bounded regular-Cauchy cluster route from finite windows and dyadic tail budgets to RegSeqRat readback, displayed subsequence extraction, and Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedRegularCauchyCluster}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Real completeness, separability, quotient equality of Cauchy names, countable-choice subsequence selection, or bridge checking.

Upgrade path: Scoped closure requires Lean checking of carrier admission, classifier stability, RegSeqRat handoff exactness, and Real-seal nonescape for the same displayed packet.

Chapter 8526

A Concrete Naming Certificate for $\mathsf{FiniteCoverNerveUniformModulus}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{FiniteCoverNerveUniformModulus}^{\uparrow}$ packet records the finite cover-nerve incidence and rational lower-bound handoff as concrete NameCert data. It consumes compact source rows from chapter 60, continuous-map rows from chapter 130, metric and rational comparison rows from chapter 57 and chapter 37, finite interval-cover rows from chapter 5105, finite Lebesgue-number rows from chapter 452, and hands the resulting ledger to chapter 369 before any $\mathsf{Real}^{\uparrow}$ readback in chapter 38.

Definition 8526.1 (FiniteCoverNerveUniformModulus carrier)**.** *A* $\mathsf{FiniteCoverNerveUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, I, N, L, R, M, V, H, C, P, Q)$$

with the following visible rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row read on the compact source.*
3. *I is the* $\mathsf{FiniteIntervalCover}^{\uparrow}$ *row presenting the finite cover cells.*
4. *N is the finite nerve-incidence row, listing cover-cell adjacency and overlap witnesses as a bounded* BHist *table.*
5. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *row giving the displayed rational lower bound for the cover.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger that keeps every chosen lower-bound read positive and below the relevant local radius.*
7. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used by the overlap and coverage tests.*
8. *V is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row receiving the finite nerve and lower-bound ledger.*
9. *H records componentwise* hsame *transport for the compact source, continuous map, cover cells, nerve incidences, Lebesgue-number row, rational comparisons, metric rows, and uniform-modulus handoff.*
10. *C records the displayed* Cont *replay routes.*

11. *P is the* Pkg *provenance over* FiniteCoverNerveUniformModulus$^{\uparrow}$, Compact$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, Rat$^{\uparrow}$, FiniteIntervalCover$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, UniformModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *Q is the local* NameCert$_{\text{FiniteCoverNerveUniformModulus}^{\uparrow}}$ *row.*

The carrier refuses open-cover compactness, selected subcovers, host minima, quotient real equality, and any Real$^{\uparrow}$ *seal not routed through the displayed uniform-modulus handoff.*

Theorem 8526.2 (FiniteCoverNerveUniformModulus NameCert obligations)**.** *For every accepted* FiniteCoverNerveUniformModulus$^{\uparrow}$ *carrier* $U = (K, F, I, N, L, R, M, V, H, C, P, Q)$*, the local* NameCert$_{\text{FiniteCoverNerveU}}$ *surface consists exactly of carrier admission for the compact source row, continuous-map row, finite interval-cover row, finite nerve-incidence row, finite Lebesgue-number row, rational comparison ledger, metric distance row, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. The classifier is stable only through H, and no obligation may be discharged by open-cover compactness, a selected subcover, a host minimum, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project an accepted packet through Definition 8526.1. The compact, continuous, cover, nerve, Lebesgue-number, rational-comparison, metric, and uniform-modulus rows are all explicit coordinates.

The transport coordinate H preserves those rows componentwise. The replay coordinate C lets compact, continuous, metric, rational, cover, and uniform-modulus consumers read the same finite packet, but it does not create an open cover or a host minimum.

Consequently the local NameCert obligations are exactly the displayed finite rows. Since the excluded objects have no carrier coordinate, they cannot be used to discharge the certificate surface. □

Theorem 8526.3 (FiniteCoverNerveUniformModulus handoff)**.** *Every* UniformModulus$^{\uparrow}$ *read exported by an accepted* FiniteCoverNerveUniformModulus$^{\uparrow}$ *packet is handed the finite cover cells I, the nerve-incidence table N, the* FiniteLebesgueNumber$^{\uparrow}$ *lower-bound row L, the* Rat$^{\uparrow}$ *comparison ledger R, and the* Met$^{\uparrow}$ *distance row M before any Real-facing consumer may inspect a* Real$^{\uparrow}$ *seal. The handoff is finite and choice-free: it does not expose open-cover compactness, selected subcovers, host minima, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Read the uniform-modulus handoff row V from the carrier. Its inputs are the finite cover row I, the nerve-incidence row N, the Lebesgue-number row L, the rational comparison ledger R, and the metric row M, all attached to the compact source K and continuous map F.

The hsame and Cont rows keep this handoff on the displayed packet. They transport and replay the finite incidence and lower-bound data, but they do not add a selector for subcovers or a minimum operation outside the carrier.

Thus the Real-facing boundary is downstream of the uniform-modulus handoff. Any consumer that reaches Real$^{\uparrow}$ must do so through the displayed finite route, not through quotient real equality or ambient completeness. □

Closure status for NameCert$_{\text{FiniteCoverNerveUniformModulus}^{\uparrow}}$**.**

Constructive story: A FiniteCoverNerveUniformModulus$^{\uparrow}$ packet is generated from compact source rows, continuous-map rows, finite interval-cover cells, finite nerve-incidence rows, finite Lebesgue-number lower bounds, rational comparison ledgers, metric rows, a UniformModulus handoff, componentwise hsame transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteCoverNerveUniformModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8526.1, Definition 8526.2, and Definition 8526.3 seed the finite cover-nerve route from compact and continuous-map rows to a rational lower-bound UniformModulus handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteCoverNerveUniformModulus}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host minima, quotient real equality, Real completeness, or bridge checking.

Upgrade path: Scoped closure requires Lean checking of carrier admission, classifier stability, finite nerve incidence exactness, rational lower-bound handoff, and Real-facing nonescape for the same displayed packet.

Chapter 8527

A Concrete Naming Certificate for $\mathsf{Stabilizability}^{\uparrow}$

Following Kalman finite-dimensional control theory, the $\mathsf{Stabilizability}^{\uparrow}$ packet records the finite feedback data that lets a control consumer separate reachable stabilization evidence from unreachable stable residue. It sits after the state-space carrier of chapter 329, the reachability surface of chapter 297, the finite rank-window certificate of chapter 8523, and the optimization-facing corridors of chapter 296, chapter 307, chapter 5103, and chapter 4752. The chapter names the finite rows that make feedback stabilization inspectable without importing host spectral subspaces, analytic semigroups, or ambient asymptotic semantics.

Definition 8527.1 (Stabilizability carrier)**.** *A* $\mathsf{Stabilizability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, U, A, B, R, M, F, D, L, H, T, P, N)$$

with the following displayed rows:

1. *X is the finite state-space source row consumed from* $\mathsf{StateSpaceModel}^{\uparrow}$*.*
2. *U is the finite input row used by the feedback surface.*
3. *A is the displayed transition row and B is the displayed input-map row.*
4. *R is the* $\mathsf{ControlControllability}^{\uparrow}$ *reachability ledger available to feedback consumers.*
5. *M is the* $\mathsf{ControllabilityRank}^{\uparrow}$ *rank-window row naming the finite columns and dimension comparisons used to separate controllable directions.*
6. *F is the unstable-mode witness row: it names exactly the finite residual rows whose uncontrolled transition must be checked before a feedback handoff can be accepted.*
7. *D is the decay ledger for the residual rows named by F after the displayed feedback row is attached.*
8. *L is the feedback row, read as finite input-on-state data rather than as an ambient operator selected outside the packet.*
9. *H is the componentwise* hsame *ledger for the state-space source, input row, transition row, input-map row, reachability ledger, rank-window row, unstable-mode witness, feedback row, and decay ledger.*

10. *T is the displayed* Cont *route for reachability readback, rank-window readback, feedback replay, and decay-ledger replay.*

11. *P is the* Pkg *provenance over* $\mathsf{Stabilizability}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ControllabilityRank}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, $\mathsf{HInfinityControl}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}$ *row.*

The classifier identifies two accepted carriers only when the displayed state-space, input, transition, input-map, reachability, rank-window, unstable-mode, feedback, and decay rows agree componentwise through H, with the same continuation route, provenance, and local naming row.

Theorem 8527.2 (Stabilizability NameCert obligations)**.** *For every accepted* $\mathsf{Stabilizability}^{\uparrow}$ *carrier $S = (X, U, A, B, R, M, F, D, L, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}$ *surface has exactly six finite rows:*

1. *carrier admission for the displayed state-space, input, transition, input-map, reachability, rank-window, unstable-mode, feedback, and decay rows;*

2. *classifier admission through componentwise* hsame *transport over those displayed rows;*

3. *reachability readback saying that R and M are consumed only through* $\mathsf{ControlControllability}^{\uparrow}$ *and* $\mathsf{ControllabilityRank}^{\uparrow}$ *rows;*

4. *feedback admission saying that L is a displayed finite row attached to the same input and state coordinates as A and B;*

5. *residual stability readback saying that the only permitted decay claim is the finite ledger D over the unstable-mode witness F after the feedback row is attached;*

6. *provenance and local naming through the displayed* Cont, Pkg, *replay, and* $\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}$ *rows.*

No obligation row supplies a host spectral decomposition, an unrecorded state quotient, a selected optimal controller, a Riccati solution, or a robust performance theorem.

Proof. Project the carrier of Definition 8527.1. The rows $X, U, A, B, R, M, F, D, L$ are the only control-facing data, while H, T, P, N transport, replay, source, and name the same finite packet.

Reachability readback is forced through R and M. These rows point to $\mathsf{ControlControllability}^{\uparrow}$ and $\mathsf{ControllabilityRank}^{\uparrow}$, so a feedback consumer cannot substitute a hidden reachable subspace or an external rank theorem for the displayed finite ledgers.

The feedback and residual stability clauses are the rows L, F, D. Since L is attached to the same displayed input and state coordinates as A and B, and since D is the only decay ledger for the residual witness F, the local certificate exports feedback stabilizing data only through those rows. Repacking the projections with H, T, P, N gives the stated six-row surface. □

Theorem 8527.3 (Stabilizability feedback handoff)**.** *Every* $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ControlRiccati}^{\uparrow}$, *or* $\mathsf{HInfinityControl}^{\uparrow}$ *consumer of an accepted* $\mathsf{Stabilizability}^{\uparrow}$ *packet may read feedback-stabilizing data only through the displayed reachability ledger R, rank-window row M, unstable-mode witness F, feedback row L, decay ledger D, transport ledger H, continuation route T, provenance row P, and local naming row N. The handoff supplies no optimality claim, Riccati equation solution, robust gain bound, host spectral subspace, or ambient asymptotic theorem outside the finite packet of Definition 8527.1.*

Proof. Start with the obligation surface of Definition 8527.2. It exposes feedback only after the state-space, input, transition, input-map, reachability, rank-window, unstable-mode, and decay rows have been displayed.

The optimization-facing consumers read this as a boundary condition. An $\mathsf{LQR}^{\uparrow}$ or $\mathsf{Riccati}^{\uparrow}$ consumer may inspect the feedback and decay rows before forming an optimization ledger, while a $\mathsf{ControlRiccati}^{\uparrow}$ or $\mathsf{HInfinityControl}^{\uparrow}$ consumer may reuse the same finite handoff for control synthesis or robustness bookkeeping.

Because the carrier has no coordinate for an optimality claim, Riccati equation solution, robust gain bound, host spectral subspace, or ambient asymptotic theorem, no such object can be exported by the stabilizability handoff. □

Falsifiable prediction: Any LQR, Riccati, control Riccati, or robust-control consumer that reads $\mathsf{Stabilizability}^{\uparrow}$ while bypassing the displayed reachability ledger R, rank-window row M, unstable-mode witness F, feedback row L, or decay ledger D violates the local certificate boundary.

Independence witness: statespacemodel, controlcontrollability, controllability-rank, lqr, riccati, control-riccati, and h-infinity-control: none is bijective to $\mathsf{Stabilizability}^{\uparrow}$ because the stabilizability packet jointly carries state-space, input-map, reachability, rank-window, unstable-mode, feedback, decay, transport, replay, provenance, and local NameCert rows while excluding host spectral, optimality, and robustness semantics.

Closure status for $\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}$.

Constructive story: A $\mathsf{Stabilizability}^{\uparrow}$ packet is generated from finite state-space and input rows, transition and input-map rows, a controllability ledger, a controllability-rank row, an unstable-mode witness, a feedback row, a decay ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8527.1, Definition 8527.2, Definition 8527.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite feedback-stabilization certificate over state-space, controllability, rank-window, LQR, Riccati, control Riccati, and robust-control rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Stabilizability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host spectral decompositions, quotient reachable subspaces, optimal controllers, Riccati solutions, robust gain bounds, or ambient asymptotic stability.

Upgrade path: Obligation closure requires checked carrier admission, reachability readback, rank-window exactness, unstable-mode witness exactness, feedback-row admission, decay-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.StabilizabilityUp.

Chapter 8528

A Concrete Naming Certificate for $\mathsf{PaleyWiener}^{\uparrow}$

Following Paley and Wiener on Fourier transforms of compactly supported functions, the $\mathsf{PaleyWiener}^{\uparrow}$ packet records the finite support-and-growth surface that a constructive Fourier-analysis consumer must inspect before it reads a band-limited or exponential-type conclusion. It sits beside the Fourier transform route of chapter 4025, the Schwartz-function test-window surface of chapter 3399, the tempered-distribution action row of chapter 2310, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite support windows, displayed growth ledgers, Fourier handoff rows, test-function probes, distribution-facing action rows, scalar readback, transport, replay, provenance, and local naming; it does not assert Fourier inversion, a completed entire-function space, quotient equality of functions or distributions, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8528.1 (Paley–Wiener carrier)**.** *A* $\mathsf{PaleyWiener}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (K, F, G, S, T, R, E, H, C, Q, N)$$

with the following displayed coordinates:

1. *K is the compact support window row, naming the finite spatial support bounds and inspected test domains.*
2. *F is the* $\mathsf{FourierTransform}^{\uparrow}$ *handoff row, recording the transform-facing read of the supported source.*
3. *G is the growth ledger row, listing the finite exponential-type and polynomial-control bounds visible to the packet.*
4. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *probe row, exposing the rapidly decreasing test windows used to compare source and transform reads.*
5. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *action row, recording the distribution-facing interpretation of the same finite transform data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *scalar readback row for the displayed bounds.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for scalar comparisons.*
8. *H records componentwise* hsame *transport for support, Fourier handoff, growth, probes, distribution action, scalar readback, seal, provenance, and local name.*

9. *C records displayed* Cont *replay from support inspection through Fourier handoff, growth readback, distribution action, scalar sealing, and local naming.*
10. *Q is the* Pkg *provenance over* $\mathsf{PaleyWiener}^{\uparrow}$, $\mathsf{FourierTransform}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PaleyWiener}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, G, S, T, R, E, H, C, Q, N$. It has no coordinate for host Fourier inversion, completed spaces of entire functions, quotient equality of functions or distributions, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8528.2 (Paley–Wiener NameCert obligations)**.** *For every accepted* $\mathsf{PaleyWiener}^{\uparrow}$ *carrier $P = (K, F, G, S, T, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{PaleyWiener}^{\uparrow}}$ *surface consists exactly of carrier admission for compact support windows, Fourier-transform handoff, finite growth ledgers, Schwartz-function probes, tempered-distribution action, regular rational readback, Real scalar sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any Fourier-analysis, distribution, or scalar-bound consumer of the packet must read through those rows before it cites a Paley–Wiener boundary.*

Proof. Project the carrier of Definition 8528.1. The support row K, transform row F, growth row G, Schwartz probe row S, distribution action row T, readback row R, and seal row E are the only analytic coordinates.

The Fourier-facing read is finite. It exposes the support window before the transform handoff, and it exposes the growth ledger before any scalar comparison can be sealed in $\mathsf{Real}^{\uparrow}$.

The test and distribution rows keep the route inside the displayed sibling certificates. The $\mathsf{SchwartzFunction}^{\uparrow}$ probes and $\mathsf{TemperedDistribution}^{\uparrow}$ action rows are read through S, T, not through a host function space or a quotient distribution.

Finally H, C, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for Fourier inversion, completed entire-function spaces, quotient equality, selected representatives, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8528.3 (Paley–Wiener Fourier support handoff)**.** *Every Fourier-facing read of an accepted* $\mathsf{PaleyWiener}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FourierTransform}^{\uparrow} \longmapsto \mathsf{SchwartzFunction}^{\uparrow} \longmapsto \mathsf{TemperedDistribution}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

after the compact support row K and growth ledger G have been displayed. The handoff exports support-window and growth-bound data together with transport, replay, provenance, and local naming, but it exports no Fourier inversion theorem, completed entire-function space, quotient equality of functions or distributions, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from the accepted carrier $P = (K, F, G, S, T, R, E, H, C, Q, N)$. The support row K and Fourier handoff row F are upstream of every transform-facing read.

The growth ledger G is then read before scalar sealing. It records only finite exponential-type and polynomial-control bounds, and the scalar comparison passes through R and E.

Next expose the sibling analysis rows. The Schwartz probe row S supplies the test-window comparison, and the tempered-distribution action row T carries the same transform data toward distribution-facing consumers.

Finally replay the route through H, C, Q, N. Because no host inversion, completed function space, quotient equality, selected representative, countable choice, or ambient completeness coordinate appears in the carrier, the Fourier support handoff remains the displayed finite NameCert route. □

Falsifiable prediction: Every accepted $\mathsf{PaleyWiener}^{\uparrow}$ Fourier-facing read exposes compact support windows, Fourier handoff, growth ledger, SchwartzFunction probes, TemperedDistribution action, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any Paley–Wiener boundary is cited.

Independence witness: fourier-transform, schwartz-function, tempered-distribution, regseqrat, real: $\mathsf{PaleyWiener}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact support windows, Fourier handoff, growth ledgers, Schwartz probes, tempered-distribution action, scalar readback, sealing, transport, replay, provenance, and local naming while refusing inversion and completed entire-function data.

Closure status for $\mathsf{NameCert}_{\mathsf{PaleyWiener}^{\uparrow}}$.

Constructive story: A $\mathsf{PaleyWiener}^{\uparrow}$ packet is generated from compact support windows, FourierTransformUp handoff rows, finite growth ledgers, SchwartzFunctionUp probes, TemperedDistributionUp action rows, RegSeqRatUp scalar readback, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PaleyWiener}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8528.1, Definition 8528.2, and Definition 8528.3 seed the finite Paley–Wiener support and growth route over Fourier transform, Schwartz-function, tempered-distribution, scalar-readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PaleyWiener}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Fourier inversion, completed entire-function spaces, quotient equality of functions or distributions, selected representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-window exactness, Fourier handoff, growth-ledger stability, Schwartz-probe and tempered-distribution action rows, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.PaleyWienerUp.

Chapter 8529

A Concrete Naming Certificate for $\mathsf{UniformCompletionIdempotence}^{\uparrow}$

Following Bishop and Bridges and Kelley on uniform-space completion, the $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ packet records the finite BEDC surface by which a completed uniform route is read back through the same uniform-completion boundary. It sits after the uniform-completion carrier of chapter 1473, the general completion-idempotence row of chapter 2381, the metric fixed-point row of chapter 3888, the separated-completion idempotence boundary of chapter 6970, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is an idempotence certificate for a displayed uniform completion route; it is not quotient completion equality, a selected representative, countable choice, host equality, or ambient completeness.

Definition 8529.1 (Uniform completion idempotence carrier)**.** *A* $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (U, C, M, S, W, R, D, E, H, K, P, N)$$

with the following displayed rows:

1. *U is the first* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary row whose completed route is being inspected.*
2. *C is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *row recording the repeated completion request as a finite idempotence surface.*
3. *M is the* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *comparison row available when a metric presentation of the same uniform route is consumed.*
4. *S is the* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ *row recording the separated classifier boundary for the completed object.*
5. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which completion-facing reads are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed before any real-facing seal is consumed.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the finite comparison budget of the repeated read.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after U, C, M, S, W, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the uniform boundary, idempotence rows, finite window, regular readback, dyadic ledger, real seal, replay, provenance, and local name.*
10. *K records displayed* Cont *replay of the same ordered route.*
11. *P records* Pkg *provenance over* $\mathsf{UniformCompletionIdempotence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{MetricCompletionIdempotence}^{\uparrow}$, $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionIdempotence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8529.2 (Uniform completion idempotence NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ *carrier*

$$I = (U, C, M, S, W, R, D, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCompletionIdempotence}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform-completion boundary U, repeated-request exposure through C, metric-presentation compatibility through M, separated-boundary compatibility through S, shared finite-window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, real-seal exposure through E, componentwise transport through H, displayed replay through K, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8529.1. The first four rows expose the uniform-completion boundary and the three idempotence witnesses that can be consumed by completion, metric, and separated presentations.

The finite analytic rows then force a single readback route. The window W, regular rational row R, and dyadic ledger D are displayed before the terminal seal E is available, so the repeated completion read cannot jump directly to a completed point.

Finally read H, K, P, N as structural rows for the same packet. They transport, replay, source, and locally name only the displayed route, and no coordinate remains from which quotient completion equality, selected representatives, countable choice, host equality, or ambient completeness could be exported. □

Theorem 8529.3 (Uniform completion idempotence unit handoff)**.** *Every unit-facing consumer of an accepted* $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ *carrier reads the second uniform-completion request through*

$$U, C, M, S, W, R, D, E, H, K, P, N$$

in that order. The consumer may use the $\mathsf{UniformCompletion}^{\uparrow}$ *boundary only together with the completion-idempotence, metric-idempotence, and separated-idempotence rows already displayed by the packet. It may not substitute quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness for that route.*

Proof. Begin with the NameCert obligations of Definition 8529.2. They place the uniform-completion boundary U before the idempotence rows C, M, S and before the finite readback rows W, R, D, E.

Use the sibling idempotence surfaces in order. The completion row C supplies the repeated-request boundary of chapter 2381; the metric row M supplies the metric fixed-point comparison of chapter 3888; and the separated row S supplies the separated classifier boundary of chapter 6970.

The regular and real-facing part of the handoff is still finite. The shared window comes from chapter 126, the regular readback from chapter 301, the tolerance ledger from chapter 291, and the terminal seal from chapter 38.

The structural rows H, K, P, N only transport, replay, source, and name that same ordered route. Hence a unit-facing consumer receives a displayed uniform-completion idempotence handoff rather than a quotient equality, representative selector, choice schedule, host equality, or detached completeness principle. □

Falsifiable prediction: An accepted $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ read is refuted if a second uniform-completion request reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without first exposing the $\mathsf{UniformCompletion}^{\uparrow}$ boundary, the $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{MetricCompletionIdempotence}^{\uparrow}$, and $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ rows, the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, transport, replay, provenance, and local naming rows.

Independence witness: uniformcompletion, completion-idempotence, metric-completion-idempotence, separated-completion-idempotence, streamname, regseqrat, dyadicratcore, real: $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries a uniform-completion boundary, three idempotence comparison rows, finite window and tolerance readback, a real seal, transport, replay, provenance, and local naming rows for the uniform fixed-point handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionIdempotence}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionIdempotence}^{\uparrow}$ packet is generated from a UniformCompletion boundary row, CompletionIdempotence row, MetricCompletionIdempotence row, SeparatedCompletionIdempotence row, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionIdempotence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8529.1, Definition 8529.2, and Definition 8529.3 seed the finite route by which a repeated uniform-completion request returns through the displayed uniform, completion-idempotence, metric-idempotence, separated-idempotence, readback, tolerance, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionIdempotence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove quotient completion equality, selected representatives, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{UniformCompletionIdempotence}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, repeated uniform-completion request exactness, metric and separated compatibility, finite-window readback, dyadic tolerance, Real seal non-escape, transport, replay, provenance, and local naming for $\mathsf{UniformCompletionIdempotence}^{\uparrow}$.

Chapter 8530

A Concrete Naming Certificate for $\mathsf{RegularCauchyLiminf}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLiminf}^{\uparrow}$ packet records the finite lower-tail envelope available from a displayed regular Cauchy rational source before any Real-facing consumer reads a terminal value. It sits over the lower-envelope route of chapter 1553, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the upper-tail companion of chapter 8383. The packet names an L10 completion-facing lower-tail envelope; it does not import host infimum, quotient stream equality, countable choice, selected subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8530.1 (Regular Cauchy liminf carrier)**.** *A* $\mathsf{RegularCauchyLiminf}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row, recording the displayed regular Cauchy rational name whose lower tail is being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite tail windows made visible to the lower-envelope request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row, binding the windows from W to regular rational values and tail tolerances.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic precision, slack, and comparison budgets for the lower-envelope read.*
5. *L is the* $\mathsf{Liminf}^{\uparrow}$ *lower-envelope row, exposing the finite tail-lower surface available at the displayed precision.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, R, D, L have all been displayed.*
7. *H records componentwise* hsame *transport for the source, windows, readback, dyadic ledger, lower-envelope, Real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a lower-tail request through*

$$S \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow L \longrightarrow E \longrightarrow H \longrightarrow P \longrightarrow N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLiminf}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Liminf}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLiminf}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, R, D, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, a host-selected infimum value, a chosen cofinal subsequence, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8530.2 (Regular Cauchy liminf NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLiminf}^{\uparrow}$ *carrier*

$$I = (S, W, R, D, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLiminf}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source sequence,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Liminf}^{\uparrow}$ *lower-envelope row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier rows of Definition 8530.1. The analytic rows S, W, R, D, L, E display the source name, finite windows, regular rational readback, dyadic tolerance ledger, lower-envelope request, and terminal Real seal in that order.

The structural rows H, C, P, N transport, replay, source, and name the same packet. They do not add a quotient stream, a selected cofinal subsequence, a host infimum, countable choice, or ambient Real completeness.

Since the classifier has no coordinate outside $S, W, R, D, L, E, H, C, P, N$, every local NameCert obligation is one of the listed rows, and every listed row is visible in the accepted carrier. □

Theorem 8530.3 (Regular Cauchy liminf Real seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchyLiminf}^{\uparrow}$ *carrier is reached through displayed regular-Cauchy lower-tail data. More precisely, every Real or Completion-facing read factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Liminf}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

with the same source sequence row S and the same displayed tail windows W governing the lower-envelope comparison.

Proof. Let a Real or Completion-facing consumer read an accepted carrier I. The terminal value is available only as the row E, and E is downstream of the source row S, window row W, readback row R, dyadic tolerance ledger D, and lower-envelope row L.

Replay the continuation row C. The consumer first opens the finite $\mathsf{StreamName}^{\uparrow}$ windows, then reads regular rational tail values through $\mathsf{RegSeqRat}^{\uparrow}$, and then consults the dyadic ledger before the lower-envelope row is available.

Componentwise transport H and provenance P preserve the same order. Therefore the Real seal cannot bypass the displayed regular-Cauchy tail rows or import a host infimum, quotient stream equality, countable choice, selected subsequence, or ambient completeness witness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLiminf}^{\uparrow}$ Real-facing read exposes a source sequence, StreamName windows, RegSeqRat tail readbacks, a DyadicRatCore tolerance ledger, Liminf lower-envelope data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any lower-tail endpoint is visible.

Independence witness: liminf, regular-cauchy-limsup, limsup-liminf-gap, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyLiminf}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regular Cauchy source, finite windows, rational tail readback, dyadic tolerances, lower-envelope data, Real sealing, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLiminf}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLiminf}^{\uparrow}$ packet is generated from a regular Cauchy source sequence, StreamName windows, RegSeqRat tail readbacks, a DyadicRatCore tolerance ledger, a Liminf lower-envelope row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLiminf}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8530.1, Definition 8530.2, and Definition 8530.3 seed the finite regular-Cauchy lower-tail envelope route through displayed windows, rational readback, dyadic tolerances, Liminf lower-envelope data, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLiminf}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infimum, quotient stream equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, RegSeqRat readback, DyadicRatCore tolerance exposure, lower-envelope exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLiminfUp.

Chapter 8531

A Concrete Naming Certificate for $\mathsf{PrecompactMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PrecompactMetricSpace}^{\uparrow}$ packet records finite-net precompactness as displayed BEDC rows before any compact or complete metric consumer reads a completion boundary. It sits after the metric carrier of chapter 57, the total-boundedness finite-net surface of chapter 2048, the compact metric boundary of chapter 129, the complete metric boundary of chapter 128, the terminal real-name surface of chapter 38, and the Cauchy-net route of chapter 8518. The packet names a finite-net route from metric data toward compact and complete consumers; it is not an open-cover compactness theorem, a selected completion point, a quotient of Cauchy names, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8531.1 (Precompact metric space carrier)**.** *A* $\mathsf{PrecompactMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, T, K, C, R, W, Q, H, A, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the metric observations and metric-law ledger used by every finite-net request.*
2. *T is the* $\mathsf{TotallyBoundedMetric}^{\uparrow}$ *finite-net row, displaying for each inspected positive tolerance a finite* ProbeBundle *of centers and coverage histories.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-boundary row consumed only as a finite-net target surface.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion-boundary row reached only after the finite-net row has been exposed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for displayed distance and tolerance reads.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *or finite-window row through which Cauchy-facing consumers inspect the selected tolerance schedule.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite windows before any real seal is exported.*
8. *H records componentwise* hsame *transport for the metric source, finite-net, compact, complete, real-seal, window, readback, provenance, and local naming rows.*
9. *A records displayed* Cont *replay from finite-net inspection through compact and complete metric consumers.*

10. *G is the* Pkg *provenance over* $\mathsf{PrecompactMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBoundedMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrecompactMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, K, C, R, W, Q, H, A, G, N$. It has no coordinate for arbitrary open covers, selected completion points, quotient equality of Cauchy names, countable choice, host compactification, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8531.2 (Precompact metric space NameCert obligations)**.** *For every accepted* $\mathsf{PrecompactMetricSpace}^{\uparrow}$ *carrier*

$$P = (M, T, K, C, R, W, Q, H, A, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrecompactMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows; metric-source admission through M; finite-net admission through T; compact-boundary admission through K; completion-boundary admission through C; real tolerance sealing through R; finite-window and regular-rational readback through W, Q; componentwise transport through H; displayed replay through A; provenance through G; and local naming through N.*

Proof. Project the carrier of Definition 8531.1. The analytic rows are the metric source M, finite-net row T, compact boundary K, complete boundary C, real seal R, finite window W, and regular readback Q.

The obligation order follows the finite-net route. A consumer first reads the metric source, then the displayed finite net, then any compact or complete metric boundary, and only then the window, readback, and real-seal rows.

The structural rows H, A, G, N transport, replay, source, and name that same route. Since no excluded object appears in the carrier, the obligation surface cannot export open-cover compactness, selected completion points, quotient Cauchy equality, countable choice, host compactification, or ambient real completeness. □

Theorem 8531.3 (Precompact metric space finite-net refinement)**.** *Let*

$$P = (M, T, K, C, R, W, Q, H, A, G, N)$$

be an accepted $\mathsf{PrecompactMetricSpace}^{\uparrow}$ *carrier. Any compact, complete, or real-facing consumer that reads P must first expose the finite-net route*

$$\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{TotallyBoundedMetric}^{\uparrow} \longrightarrow \mathsf{CompactMetric}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The row T refines every tolerance request into a displayed finite probe bundle before K, C, W, Q, or R can be consumed, and the structural rows H, A, G, N keep that refinement attached to the same packet.

Proof. Read M as the source metric row. It supplies only displayed distance observations and metric-law ledgers, so a compact or complete metric consumer has no access to a host completion boundary at this stage.

Next read T. The total-boundedness row supplies the finite probe bundles and coverage histories for the inspected tolerances. This is the only row that can feed the downstream compact and complete boundaries K and C.

Finally read W, Q, R and the structural rows H, A, G, N. The finite windows and regular readback carry the tolerance schedule to the real seal, while transport, replay, provenance, and local naming preserve the same finite-net route. Any consumer that bypasses T would use a coordinate absent from Definition 8531.1. □

Falsifiable prediction: An accepted $\mathsf{PrecompactMetricSpace}^{\uparrow}$ consumer that reaches $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ data without displaying the finite-net row T refutes the carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PrecompactMetricSpace}^{\uparrow}$ packet is generated from a MetricSpace source row, TotallyBoundedMetric finite-net row, CompactMetric boundary row, CompleteMetric boundary row, Real tolerance seal, finite-window row, RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrecompactMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8531.1, Definition 8531.2, and Definition 8531.3 seed the finite-net precompactness route over metric-source, total-boundedness, compact-boundary, complete-boundary, real-seal, finite-window, regular-readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrecompactMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected completion points, quotient equality of Cauchy names, countable choice, host compactification, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage admission, compact-boundary admission, completion-boundary admission, finite-window readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PrecompactMetricSpaceUp.

Chapter 8532

A Concrete Naming Certificate for $\mathsf{CoarseSpace}^{\uparrow}$

Following Roe's large-scale geometry, $\mathsf{CoarseSpace}^{\uparrow}$ records controlled-set data over BEDC histories. It sits after the metric rows of chapter 57, the entourage discipline of chapter 2205, the metric-controlled specialization of chapter 6645, and the completion-facing uniform route of chapter 3560. The packet names large-scale control as finite carrier data; it does not identify small-scale uniform equivalence, quotient coarse maps, selected representatives, countable choice, or ambient completion.

Definition 8532.1 (CoarseSpace carrier)**.** *A* $\mathsf{CoarseSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (U, \Delta, E, J, K, I, H, C, P, N)$$

with the following visible rows:

1. *U is the underlying history row carrying the displayed points or probes read by a coarse consumer.*
2. *Δ is the diagonal controlled-entourage row.*
3. *E is a finite list of controlled-entourage rows over the same* BHist *source.*
4. *J records the controlled-union ledger for finite unions of rows in E.*
5. *K records the controlled-composition ledger for composable rows in E.*
6. *I records the inverse or symmetry row when the presented coarse structure includes it.*
7. *H records componentwise* hsame *transport for the source row, diagonal row, controlled-entourage rows, union ledger, composition ledger, inverse row, replay routes, provenance, and local naming.*
8. *C records displayed* Cont *replay from a consumer request through the same controlled-entourage rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CoarseSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformEntourageSpace}^{\uparrow}$, $\mathsf{CoarseMetricSpace}^{\uparrow}$, $\mathsf{UniformSpaceCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CoarseSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed controlled rows and their structural provenance. It has no coordinate for quotient coarse categories, ambient bornologies, selected representatives, countable choice, or small-scale uniform equivalence.

Theorem 8532.2 (CoarseSpace NameCert obligations)**.** *For an accepted* $\mathsf{CoarseSpace}^{\uparrow}$ *carrier* $X = (U, \Delta, E, J, K, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CoarseSpace}^{\uparrow}}$ *surface consists of carrier admission, the diagonal row* Δ*, controlled finite unions through* J*, controlled composition through* K*, inverse or symmetry exposure through* I *where present, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local NameCert naming through* N*. Every public controlled-entourage read must expose those rows before it can be consumed as large-scale coarse data.*

Proof. Carrier admission is the displayed finite packet. The diagonal, controlled-entourage list, union ledger, composition ledger, inverse row, transport rows, replay routes, package provenance, and local naming row are all visible coordinates of the same BHist surface.

The closure rows are finite. A union read uses J, a composition read uses K, and an inverse or symmetry read uses I only when that row is present in the carrier. Componentwise transport through H preserves the same rows rather than replacing them by an external coarse category.

The consumer routes are exhausted by C, P, N. Since no excluded object is a coordinate of the carrier, the NameCert surface cannot export quotient coarse maps, selected representatives, countable choice, ambient bornologies, or small-scale uniform equivalence. □

Theorem 8532.3 (CoarseSpace metric entourage handoff)**.** *Every metric-controlled* $\mathsf{CoarseSpace}^{\uparrow}$ *read factors through the displayed* $\mathsf{Met}^{\uparrow}$ *source and the existing* $\mathsf{CoarseMetricSpace}^{\uparrow}$ *specialization before the controlled-entourage rows are consumed. The handoff may also be read against* $\mathsf{UniformEntourageSpace}^{\uparrow}$ *and* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *as provenance for entourage and completion-facing consumers, but it does not assert that the large-scale controlled structure is equivalent to the small-scale uniform structure.*

Proof. Read the metric-controlled request through the provenance row P. Its metric side is the displayed $\mathsf{Met}^{\uparrow}$ source, and its large-scale specialization is the $\mathsf{CoarseMetricSpace}^{\uparrow}$ route already named as a dependency of the carrier.

The controlled-entourage rows of X are then consumed through $\Delta, E, J, K, I, H, C, P, N$. The $\mathsf{UniformEntourageSpace}^{\uparrow}$ and $\mathsf{UniformSpaceCompletion}^{\uparrow}$ references record where entourage and completion-facing readers may inspect compatible finite rows.

Nothing in the packet identifies those small-scale rows with the large-scale controlled structure. The handoff is therefore a factorization route for metric-controlled coarse reads, not a uniform-equivalence theorem. □

Falsifiable prediction: Every accepted $\mathsf{CoarseSpace}^{\uparrow}$ read displays the source row, diagonal controlled-entourage row, controlled-entourage packet, finite union ledger, composition ledger, available inverse or symmetry row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any large-scale control is exported.

Independence witness: metric, uniform-entourage-space, coarse-metric-space, uniform-space-completion: no listed sibling is bijective to $\mathsf{CoarseSpace}^{\uparrow}$ because this packet jointly carries diagonal, controlled union, controlled composition, optional inverse or symmetry, transport, replay, provenance, and local naming rows for an abstract controlled-set carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CoarseSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CoarseSpace}^{\uparrow}$ packet is generated from displayed source histories, a diagonal controlled-entourage row, finite controlled-entourage rows, controlled union and composition ledgers, an available inverse or symmetry row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoarseSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8532.1, Definition 8532.2, and Definition 8532.3 seed the finite controlled-set surface and its metric-controlled handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{CoarseSpace}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient coarse categories, small-scale uniform equivalence, selected representatives, countable choice, ambient bornologies, completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal exposure, controlled union, controlled composition, inverse or symmetry transport, metric handoff, replay, provenance, and local naming for BEDC.Derived.CoarseSpaceUp.

Chapter 8533

A Concrete Naming Certificate for $\mathsf{CauchyCompletionCommutation}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{CauchyCompletionCommutation}^{\uparrow}$ packet records the finite route by which a displayed binary product and a displayed Cauchy-completion read may be compared without choosing representatives or importing a quotient completion. It sits after the monad surface of chapter 378, the functor surface of chapter 423, the regular-sequence rational carrier of chapter 301, the stream-window carrier of chapter 126, the dyadic tolerance carrier of chapter 291, the real seal of chapter 38, and the product carrier of chapter 34. The packet names only finite product routing, completion routing, stream windows, dyadic tolerances, real readback, transport, replay, provenance, and local naming rows.

Definition 8533.1 (Cauchy completion commutation carrier)**.** *A* $\mathsf{CauchyCompletionCommutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, R, U, V, W, G, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two input carrier rows whose finite product read is inspected.*
2. *R is the* $\mathsf{Prod}^{\uparrow}$ *product route, exposing pair formation and component projections before any completion read is consumed.*
3. *U is the completion-then-product route, reading* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *on A and B before the product row is formed.*
4. *V is the product-then-completion route, forming the displayed product row before the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion read is consumed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite prefixes used by both routes.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational-sequence witnesses visible through W.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying the dyadic precision budget used to compare the two finite routes.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the two reads are exported only as a real-valued finite tolerance comparison.*

9. *H records componentwise* hsame *transport for $A, B, R, U, V, W, G, D, E, C, P, N$.*
10. *C records displayed* Cont *replay through*

$$A, B \longmapsto R \longmapsto U, V \longmapsto W, G, D \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionCommutation}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionFun}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCommutation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected representatives, countable choice, host product equality, ambient real completeness, or a global theorem identifying products in an external category.

Theorem 8533.2 (Cauchy completion commutation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionCommutation}^{\uparrow}$ *carrier $K = (A, B, R, U, V, W, G, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyComplet}}$ *surface consists exactly of product-route admission through R, completion-then-product admission through U, product-then-completion admission through V, stream-window exposure through W, regular rational readback through G, dyadic tolerance exposure through D, real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8533.1. The rows A, B, R display the finite product route before either completion path is compared.

Next read the two completion routes. The row U exposes completion before product formation, while V exposes product formation before completion. The shared stream-window row W, regular-sequence readback row G, and dyadic tolerance row D give the finite evidence by which the two routes can be compared.

Finally the real seal E exports only that finite tolerance comparison. The structural rows H, C, P, N transport, replay, source, and name the same packet, and the carrier contains no coordinate for representative choice or ambient product equality. Hence the listed rows exhaust the NameCert obligations. □

Theorem 8533.3 (Cauchy completion finite route commutation)**.** *Every accepted* $\mathsf{CauchyCompletionCommutation}^{\uparrow}$ *read factors the comparison of the two routes through the finite diagram*

$$\mathsf{Prod}^{\uparrow}, \mathsf{CauchyCompletionFunctor}^{\uparrow}, \mathsf{CauchyCompletionMonad}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{R}$$

Thus the completion-then-product route and the product-then-completion route can be compared only after product routing, completion routing, stream windows, regular rational readback, dyadic tolerance, and the real seal have all been displayed.

Proof. Start from the product row R. It exposes pair formation and projections as finite data, so both later routes consume the same displayed product interface.

Read U and V as two ordered uses of the completion surfaces. The comparison is not made at the level of host products; it is made only after both routes have supplied the same kind of stream windows and regular rational readbacks.

The dyadic row D supplies the common tolerance budget, and the real seal E exports the resulting finite comparison. Since H, C, P, N only preserve and name this route, every accepted read factors through the displayed finite diagram. □

Theorem 8533.4 (Cauchy completion commutation nonescape)**.** *An accepted* CauchyCompletionCommutation$^{\uparrow}$ *packet cannot be read as a quotient-completion theorem, a selected-representative construction, a countable-choice principle, host product equality, or ambient* Real$^{\uparrow}$ *completeness. Any consumer of the packet receives only the displayed finite route* $A, B, R, U, V, W, G, D, E, H, C, P, N$.

Proof. Inspect the carrier coordinates. None of $A, B, R, U, V, W, G, D, E, H, C, P, N$ contains a selector for Cauchy representatives, a quotient-equality rule, a countable-choice row, or a host product equality row.

The comparison rows stop at dyadic tolerance and the real seal. They can report that the two finite routes meet the displayed tolerance interface, but they cannot replace this report by a theorem of ambient completeness.

Therefore every consumer remains inside the named packet. Any request for quotient completion, representative selection, choice, host equality, or ambient real completeness leaves the accepted CauchyCompletionCommutation$^{\uparrow}$ carrier. □

Closure status for NameCert$_{\text{CauchyCompletionCommutation}^{\uparrow}}$**.**

Constructive story: A CauchyCompletionCommutation$^{\uparrow}$ packet is generated from two input carrier rows, a product route, completion-then-product and product-then-completion routes, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionCommutation}^{\uparrow}}$, seed)

Scope closed: Definition 8533.1, Definition 8533.2, Definition 8533.3, and Definition 8533.4 seed the finite product/completion route comparison over $A, B, R, U, V, W, G, D, E, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionCommutation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked product routing, completion routing, stream-window exposure, regular rational readback, dyadic tolerance, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionC

Chapter 8534

A Concrete Naming Certificate for HyperbolicPhaseVisualModulus$^{\uparrow}$

The HyperbolicPhaseVisualModulus$^{\uparrow}$ packet realizes the observer-expansion visual-modulus branch of chapter 9463 as finite NameCert data. It sits after the visual metric of chapter 3866, the observer phase route of chapter 6920, the Busemann-function readback of chapter 7534, the boundary-shadow surface of chapter 3813, the dyadic tolerance carrier of chapter 291, and the real seal of chapter 38. The packet names only finite observer phase rows, visual metric rows, Busemann rows, boundary-shadow windows, dyadic tolerance, real sealing, transport, replay, provenance, and local naming.

Definition 8534.1 (Hyperbolic phase visual-modulus carrier)**.** *A* HyperbolicPhaseVisualModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, V, B, S, D, R, T, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the* HyperbolicObserverPhaseRoute$^{\uparrow}$ *row, recording the observer phase route before a visual tolerance is read.*
2. *V is the* HyperbolicVisualMetric$^{\uparrow}$ *row, exposing the visual-metric comparison surface.*
3. *B is the* HyperbolicBusemannFunction$^{\uparrow}$ *row, recording the Busemann readback used to orient the observer phase.*
4. *S is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row, recording the finite boundary-shadow window consumed by the visual metric.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, displaying the dyadic modulus budget.*
6. *R is the* Real$^{\uparrow}$ *seal row, exporting the visual-modulus comparison only as a real-valued finite tolerance read.*
7. *T is the TasteGate row requiring* Nontrivial HyperbolicPhaseVisualModulus$^{\uparrow}$ *and* FieldFaithful HyperbolicPhaseVisu *instances before the packet can classify observer-expansion visual modulus data.*
8. *F is the refusal row, rejecting reads that turn the visual modulus into a global hyperbolic boundary oracle, selected geodesic representative, or ambient visual compactification.*
9. *H records componentwise* hsame *transport for $O, V, B, S, D, R, T, F, C, P, N$.*

10. *C records displayed* Cont *replay through*

$$O, V \longmapsto B, S \longmapsto D \longmapsto R \longmapsto T, F, H, P, N.$$

11. *P is the* Pkg *provenance over* HyperbolicPhaseVisualModulus$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicObserverPhase
HyperbolicBusemannFunction$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{HyperbolicPhaseVisualModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed visual boundary, a global metric oracle, selected geodesic representatives, host hyperbolic compactification, or an ambient theorem identifying all observer phases.

Theorem 8534.2 (Hyperbolic phase visual-modulus NameCert obligations)**.** *For every accepted* HyperbolicPhaseVisualModulus$^{\uparrow}$ *carrier $M = (O, V, B, S, D, R, T, F, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicPhaseVisua}}$ *surface consists exactly of observer-phase admission through O, visual-metric admission through V, Busemann readback through B, boundary-shadow exposure through S, dyadic tolerance exposure through D, real sealing through R, TasteGate nontriviality and field-faithfulness requests through T, refusal through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8534.1. The rows O, V, B, S expose the observer route, visual metric, Busemann readback, and boundary shadow as separate finite fields.

The tolerance and sealing rows then determine what can be exported. The dyadic row D gives the finite modulus budget, while R seals the result as a real-valued comparison rather than a boundary oracle.

Finally the TasteGate, refusal, and structural rows finish the certificate. The row T names the required Nontrivial and FieldFaithful instances, F blocks global-boundary readings, and H, C, P, N only transport, replay, source, and name the same packet. No further NameCert obligation is available outside the listed rows. □

Theorem 8534.3 (Hyperbolic phase visual-modulus boundary route)**.** *Every accepted* HyperbolicPhaseVisualModul *boundary read factors through*

HyperbolicObserverPhaseRoute$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannFunction$^{\uparrow}$, HyperbolicBoundaryShadow

Thus the observer-expansion visual modulus is available only after the observer phase, visual metric, Busemann readback, boundary shadow, dyadic tolerance, and real seal have all been displayed.

Proof. Start with the observer-phase and visual-metric rows. They provide the phase route and visual comparison surface, but neither row alone carries a modulus exported to a consumer.

Next read the Busemann and boundary-shadow rows. These rows orient the boundary-facing comparison and supply the finite shadow window to which the modulus budget applies.

The dyadic tolerance row D and real seal R complete the boundary route. Since H, C, P, N preserve only this displayed path, every accepted boundary read factors through the stated finite diagram. □

Theorem 8534.4 (Hyperbolic phase visual-modulus expansion readback)**.** *An observer-expansion read of an accepted* HyperbolicPhaseVisualModulus$^{\uparrow}$ *packet exports only the row family $O, V, B, S, D, R, T, F, H, C, P,$ It cannot be read as a global hyperbolic visual-boundary oracle, a selected geodesic representative, a completed compactification, or a proof that all observer phases have the same ambient visual metric.*

Proof. Inspect the refusal and TasteGate rows. The row F blocks global-boundary and selected-representative readings, while T requires nontrivial and field-faithful Lean-side instances before the packet can classify visual-modulus data.

Readback therefore remains finite. The rows O, V, B, S, D, R give the observer route, visual metric, Busemann readback, boundary shadow, dyadic tolerance, and real seal, and the rows H, C, P, N transport, replay, source, and name only those data.

Hence no consumer can extract a completed compactification, global oracle, selected representative, or ambient equality of observer phases from the packet. The expansion readback is exactly the displayed row family. □

Falsifiable prediction: A packet missing D or R may still display observer-phase and boundary-shadow rows, but it cannot export a real-valued visual modulus.

Independence witness: hyperbolic-visual-metric, hyperbolic-observer-phase-route, hyperbolic-busemann-function, hyperbolic-boundary-shadow, real: no listed sibling is bijective with HyperbolicPhaseVisualMod because none preserves observer phase, visual metric, Busemann readback, boundary shadow, dyadic tolerance, real seal, TasteGate request, refusal, transport, replay, provenance, and local NameCert together.

The Lean-side boundary is expected to provide explicit Nontrivial HyperbolicPhaseVisualModulus$^{\uparrow}$ and FieldFaithful HyperbolicPhaseVisualModulus$^{\uparrow}$ instances. The field-faithfulness witness must keep $O, V, B, S, D, R, T, F, H, C, P, N$ as distinct rows before any classifier target for HyperbolicPhaseVisualModulus$^{\uparrow}$ can accept the packet.

Closure status for NameCert$_{\text{HyperbolicPhaseVisualModulus}^{\uparrow}}$.

Constructive story: A HyperbolicPhaseVisualModulus$^{\uparrow}$ packet is generated from observer phase, visual metric, Busemann readback, boundary shadow, dyadic tolerance, real seal, TasteGate request, refusal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicPhaseVisualModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8534.1, Definition 8534.2, Definition 8534.3, and Definition 8534.4 seed the finite observer-expansion visual-modulus route over $O, V, B, S, D, R, T, F, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicPhaseVisualModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a completed visual boundary, global metric oracle, selected geodesic representatives, host hyperbolic compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked observer-phase admission, visual-metric admission, Busemann readback, boundary-shadow exposure, dyadic tolerance, real sealing, TasteGate nontriviality, field faithfulness, refusal, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseVisualModulusUp.

Chapter 8535

A Concrete Naming Certificate for $\mathsf{SteinitzExchange}^{\uparrow}$

Following Steinitz and the standard finite-dimensional linear algebra exchange argument, the $\mathsf{SteinitzExchange}^{\uparrow}$ packet records the finite route by which a displayed independent list, a displayed spanning list, and a selected exchange position are read as one NameCert surface. It sits after the vector-space carrier of chapter 47, the linear-map and matrix rows of chapter 48 and chapter 49, the rank accounting surface of chapter 3438, and the scalar source of chapter 45. The packet names only finite coordinate, independence, span, pivot, transport, replay, provenance, and local naming rows; it does not import a host basis choice, quotient vector equality, Zorn maximality, dimension comparison outside the displayed rank ledger, or ambient linear algebra.

Definition 8535.1 (Steinitz exchange carrier)**.** *A* $\mathsf{SteinitzExchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, V, L, S, M, R, Piv, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{Field}^{\uparrow}$ *scalar row.*
2. *V is the accepted* $\mathsf{VecSpace}^{\uparrow}$ *vector-space row over F.*
3. *L is the finite independent-list row whose entries are read through V.*
4. *S is the finite spanning-list row whose entries are read through V.*
5. *M is the* $\mathsf{Mat}^{\uparrow}$ *coordinate ledger recording the finite coefficient table from L into the span of S.*
6. *R is the* $\mathsf{RankNullity}^{\uparrow}$ *accounting row for the displayed coordinate map.*
7. *Piv is the pivot row naming the exchanged source position, target position, and nonzero scalar guard visible in the matrix ledger.*
8. *B is the basis-handoff row, recording the finite list obtained by replacing the target row of S by the selected row of L.*
9. *H records componentwise* hsame *transport for scalar, vector, list, matrix, rank, pivot, basis-handoff, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay through the same finite exchange route.*
11. *P is the* Pkg *provenance over* $\mathsf{SteinitzExchange}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{RankNullity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{SteinitzExchange}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, V, L, S, M, R, Piv, B, H, C, P, N$. It has no coordinate for an ambient basis selector, quotient equality of vectors, a maximal independent set, or a dimension theorem outside the displayed finite ledger.

Theorem 8535.2 (Steinitz exchange NameCert obligations)**.** *For every accepted* $\mathsf{SteinitzExchange}^{\uparrow}$ *carrier*

$$E = (F, V, L, S, M, R, Piv, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SteinitzExchange}^{\uparrow}}$ *surface consists exactly of scalar-field exposure, vector-space exposure, independent-list exposure, spanning-list exposure, finite matrix-coordinate exposure, rank-nullity accounting, pivot exposure, basis-handoff exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read the exchange handoff only after the same F, V, L, S, M, R, Piv route has been displayed.*

Proof. Project the carrier E. The algebraic rows expose the scalar field, vector-space carrier, independent list, spanning list, finite coordinate matrix, rank ledger, pivot, and exchanged list in one packet.

The pivot row is downstream of the matrix and rank rows. Thus a consumer cannot obtain the exchanged list from a hidden basis choice or from a host dimension theorem; it must read the displayed nonzero scalar guard and the finite coordinate position in Piv.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the excluded ambient objects are not carrier coordinates, the NameCert obligations are exhausted by the displayed exchange packet. □

Theorem 8535.3 (Steinitz exchange basis handoff)**.** *Every* $\mathsf{SteinitzExchange}^{\uparrow}$ *consumer that exports a basis-exchange handoff to a* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, or* $\mathsf{RankNullity}^{\uparrow}$ *sibling must first read the shared finite route*

$$\mathsf{Field}^{\uparrow}, \mathsf{VecSpace}^{\uparrow}, L, S, M, R, Piv, B, H, C, P, N.$$

The handoff keeps the independent row L, spanning row S, coordinate matrix M, rank ledger R, and pivot row Piv attached to the same vector-space and field rows; it cannot split the exchanged basis row B from its finite coordinate witness.

Proof. Begin with the NameCert obligations of Definition 8535.2. They expose the exchange handoff only after the field, vector-space, independent-list, spanning-list, matrix, rank, and pivot rows have been displayed.

The sibling consumers read different faces of the same packet. A $\mathsf{LinMap}^{\uparrow}$ read uses the coordinate map, a $\mathsf{Mat}^{\uparrow}$ read uses the finite coefficient table, and a $\mathsf{RankNullity}^{\uparrow}$ read uses the rank ledger; none of these rows contains an independent basis selector.

Finally replay the structural rows H, C, P, N. They preserve the same finite exchange route under transport, continuation, provenance, and local naming, so the exported basis handoff remains the displayed Steinitz exchange packet rather than a host basis-choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{SteinitzExchange}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SteinitzExchange}^{\uparrow}$ packet is generated from a displayed scalar field, vector-space row, independent list, spanning list, finite matrix-coordinate ledger, rank-nullity accounting row, pivot row, exchanged basis-handoff row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SteinitzExchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8535.1, Definition 8535.2, and Definition 8535.3 seed the finite exchange surface over Field, VecSpace, LinearMap, Matrix, and RankNullity rows.

Formal verification: FormalStatus(NameCert$_{\text{SteinitzExchange}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host basis selection, quotient vector equality, Zorn maximality, unrestricted dimension comparison, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite list exposure, matrix-coordinate exactness, rank-nullity handoff, pivot non-escape, basis-handoff transport, provenance, and local naming for BEDC.Derived.SteinitzExchangeUp.

Chapter 8536

A Concrete Naming Certificate for $\mathsf{HeytingAlgebra}^{\uparrow}$

Following Heyting and Brouwer, the $\mathsf{HeytingAlgebra}^{\uparrow}$ packet records the constructive algebraic reading of implication as a right adjoint to finite meet, rather than as Boolean complement or excluded middle. It sits beside the Boolean endpoint of chapter 3733, the meet–join surface of chapter 55, the order carrier of chapter 53, the point-free reading of chapter 5153, and the open-set discipline of chapter 88. The packet supplies the intuitionistic algebraic interface needed by Real and Completion chapters while refusing powerset, quotient, ultrafilter, prime-filter, and excluded-middle coordinates.

Definition 8536.1 (Heyting algebra carrier)**.** *A* $\mathsf{HeytingAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (L, \bot, \top, \mathsf{meet}, \mathsf{join}, \mathsf{imp}, H_s, C, P, N),$$

where:

1. *L is the bounded-lattice row, read through the displayed* $\mathsf{Lattice}^{\uparrow}$ *and* $\mathsf{Poset}^{\uparrow}$ *sibling obligations.*
2. *$\bot$ and $\top$ are the bottom and top endpoint rows, stable under the classifier inherited from the bounded-lattice surface.*
3. meet *and* join *are finite meet and join rows, transported only through componentwise* hsame *over the displayed lattice packet.*
4. imp *is the implication row, recording for displayed a, b, c the finite adjunction read that $a \wedge b \leq c$ exactly when $a \leq b \Rightarrow c$ inside the same classifier-stable order.*
5. *H_s records bounded-lattice stability, implication-adjunction stability, and refusal of Boolean complement as a required coordinate.*
6. *C is the displayed* Cont *replay row for the bounded-lattice and implication-adjunction data.*
7. *P is the* Pkg *provenance over* $\mathsf{HeytingAlgebra}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HeytingAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, \bot, \top, \mathsf{meet}, \mathsf{join}, \mathsf{imp}, H_s, C, P, N$. It has no coordinate for Boolean complement, powerset representation, quotient order equality, excluded middle, prime filters, ultrafilters, or Stone duality.

Theorem 8536.2 (Heyting algebra NameCert obligations)**.** *For every accepted* $\mathsf{HeytingAlgebra}^{\uparrow}$ *carrier* $H = (L, \bot, \top, \mathsf{meet}, \mathsf{join}, \mathsf{imp}, H_s, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HeytingAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier exactness, bounded-lattice endpoint exposure, meet and join exposure, implication-adjunction exposure, transport stability, replay stability, provenance, and local naming.*

In particular, a consumer may read the implication row only after the bounded-lattice rows and the $\mathsf{Poset}^{\uparrow}/\mathsf{Lattice}^{\uparrow}$ *order comparison are visible. The certificate does not supply Boolean completeness, a complement operation, powerset membership, quotient classes, excluded middle, prime filters, ultrafilters, or Stone representation data.*

Proof. Project the carrier of Definition 8536.1. The finite packet exposes the bounded-lattice row L, endpoints, meet, join, implication, stability row, replay row, provenance row, and local naming row as separate coordinates.

The classifier is inherited only through the displayed order and lattice rows. Thus any accepted consumer must preserve the $\mathsf{Poset}^{\uparrow}$ comparison, the $\mathsf{Lattice}^{\uparrow}$ meet–join surface, and the endpoint rows before the implication row can be inspected.

The implication obligation is the adjunction row in H_s. It records the finite comparison between $a \wedge b \leq c$ and $a \leq b \Rightarrow c$ inside the same order, so implication is read as a constructive residual of meet.

Finally the structural rows C, P, N replay, source, and name the same finite packet. Since no complement, powerset, quotient, excluded-middle, prime-filter, ultrafilter, or Stone-duality coordinate occurs in the carrier, none can be exported by the local certificate. □

Theorem 8536.3 (Heyting algebra implication adjunction)**.** *Let* $H = (L, \bot, \top, \mathsf{meet}, \mathsf{join}, \mathsf{imp}, H_s, C, P, N)$ *be an accepted* $\mathsf{HeytingAlgebra}^{\uparrow}$ *packet whose* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *histories are classifier-stable. For displayed rows* a, b, c *in the order carrier, the BEDC implication row is read by the finite adjunction relation*

$$a \wedge b \leq c \quad \textit{iff} \quad a \leq b \Rightarrow c$$

inside those classifier-stable histories. This read is an implication-adjunction certificate, not a Boolean complement, powerset, quotient, or excluded-middle coordinate.

Proof. Project the bounded-lattice rows from Definition 8536.1. The order comparison, meet row, join row, and endpoints are already supplied through the $\mathsf{Poset}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ sibling histories, so the displayed expressions $a \wedge b \leq c$ and $a \leq b \Rightarrow c$ are read in the same classifier-stable order.

Next read the implication row and the stability row H_s. These rows record implication only as the right-adjoint comparison to finite meet. They do not add an independent complement operation or a classical truth-value discriminator.

Transport and replay preserve the same relation. Componentwise hsame and displayed Cont move the bounded-lattice and implication rows together, while P and N source and name the same finite packet.

Therefore the implication row is exactly the displayed adjunction relation over the accepted Poset and Lattice histories. Since the packet has no Boolean complement, powerset, quotient, or excluded-middle coordinate, those readings are refused rather than silently imported. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeytingAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{HeytingAlgebra}^{\uparrow}$ packet is generated from bounded-lattice rows, endpoint rows, meet and join rows, an implication-adjunction row, stability data, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeytingAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8536.1, Definition 8536.2, and Definition 8536.3 seed the finite constructive implication-algebra packet over PosetUp, LatticeUp, BooleanAlgebraUp, LocaleSpaceUp, TopologyUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{HeytingAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Boolean completeness, classical powersets, prime filters, ultrafilters, Stone duality, quotient representations, excluded middle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-lattice endpoint exposure, meet and join stability, implication-adjunction stability, transport, replay, provenance, and local naming for BEDC.Derived.HeytingAlgebraUp.

Chapter 8537

A Concrete Naming Certificate for $\mathsf{InhabitedLocatedCut}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{InhabitedLocatedCut}^{\uparrow}$ packet records a two-sided located-cut witness whose lower and upper rows are both inhabited before any $\mathsf{Real}^{\uparrow}$ seal is admitted. It sits beside the lower-row carrier of chapter 8086, the rounded-cut surface of chapter 6971, the two-sided located Dedekind cut of chapter 7256, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet exposes finite cut witnesses and readback rows for Real and Completion consumers; it does not assert Dedekind completeness, quotient cut equality, trichotomy, or selected host reals.

Definition 8537.1 (InhabitedLocatedCut carrier)**.** *An* $\mathsf{InhabitedLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, A, G, R, Q, W, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-cut row, carrying the finite lower observations and downward-closed readback inspected by the packet.*
2. *U is the upper-cut row, carrying the finite upper observations and upward-closed readback paired with L.*
3. *A is the inhabited witness row. It displays at least one accepted lower observation and at least one accepted upper observation inside the same packet.*
4. *G is the locatedness modulus row. For every displayed interval request $a < b$ opened by the packet, G records the finite branch that witnesses a lower-side read below b or an upper-side read above a.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed lower, upper, inhabited, and locatedness rows.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint comparison, gap narrowing, radius accounting, and finite budget replay.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that fixes the finite observations of L, U, A, G, R, Q available to Real and Completion consumers.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after L, U, A, G, R, Q, W have been displayed.*

9. *H records componentwise* hsame *transport for lower rows, upper rows, inhabited witnesses, locatedness moduli, readbacks, dyadic ledgers, windows, Real seals, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay through*

$$L, U, A, G \longmapsto R, Q, W \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{InhabitedLocatedCut}^{\uparrow}$, $\mathsf{LocatedLowerCut}^{\uparrow}$, $\mathsf{RoundedCut}^{\uparrow}$, $\mathsf{LocatedDedekindCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{InhabitedLocatedCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, A, G, R, Q, W, E, H, C, P, N$. It has no coordinate for Dedekind completeness, arbitrary subset suprema, quotient cut equality, selected host reals, trichotomy, countable choice, or bridge checking.

Theorem 8537.2 (InhabitedLocatedCut NameCert obligations)**.** *For every accepted* $\mathsf{InhabitedLocatedCut}^{\uparrow}$ *carrier $I = (L, U, A, G, R, Q, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InhabitedLocatedCut}^{\uparrow}}$ *obligation surface is exhausted by carrier admission, inhabited witness exposure, located-cut compatibility, classifier stability, and non-escape along the displayed rows. A consumer receives the lower and upper cut rows L, U, the inhabited witness row A, the locatedness modulus G, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Q, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule W, the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C, provenance P, and local naming N, and no further analytic principle.*

Proof. Project the carrier of Definition 8537.1. Carrier admission exposes exactly the rows $L, U, A, G, R, Q, W, E, H$

The inhabited witness obligation is carried by A. It records one lower-side observation and one upper-side observation in the same finite packet, so the certificate never appeals to a detached existence principle.

Located-cut compatibility is carried by G together with L and U. Each displayed interval request is answered inside the finite row family, and the rows R, Q, W merely repack those answers as regular rational readback, dyadic tolerance accounting, and stream-window access.

Classifier stability and non-escape are structural. The rows H, C, P, N transport, replay, source, and name only the displayed data, while E seals only the already exposed route. Since no row contains completeness, quotient equality, selected host reals, trichotomy, arbitrary suprema, countable choice, or bridge data, none can be exported by the local NameCert. □

Theorem 8537.3 (InhabitedLocatedCut Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{InhabitedLocatedCut}^{\uparrow}$ *packet factors through the displayed cut rows, locatedness row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *window schedule, componentwise transport, continuation replay, provenance, and local naming rows before it can read the terminal seal:*

$$L, U, A, G \longmapsto R, Q, W \longmapsto H, C, P, N \longmapsto E.$$

The factorization gives Real and Completion consumers an inhabited located-cut witness without completing arbitrary cuts, quotienting cut equality, selecting host reals, or deciding trichotomy.

Proof. Start with the cut rows. The lower row L and upper row U provide the displayed two-sided cut surface, while A supplies the inhabited witness and G supplies the locatedness answers.

Next read the finite observation and tolerance rows. The regular readback R is computed from the displayed cut evidence, the dyadic ledger Q records endpoint and gap budgets, and the stream schedule W fixes which observations the consumer is allowed to inspect.

The structural rows then preserve the same route. Componentwise transport H, continuation replay C, provenance P, and local naming N can only replay the row family already exposed by the packet.

The seal E is terminal for this route. A consumer reaching E has therefore passed through the cut, locatedness, readback, tolerance, window, transport, replay, provenance, and naming rows, and cannot extract Dedekind completeness, arbitrary subset suprema, quotient cut equality, selected host reals, trichotomy, countable choice, or bridge checking from the packet. □

Falsifiable prediction: A $\mathsf{Real}^{\uparrow}$ consumer that reads an $\mathsf{InhabitedLocatedCut}^{\uparrow}$ seal while bypassing the inhabited witness row, locatedness modulus, regular rational readback, dyadic tolerance ledger, or StreamName window refutes the carrier of Definition 8537.1.

Independence witness: located-lower-cut, rounded-cut, located-dedekind-cut, regseqrat, real: no listed sibling is bijective with $\mathsf{InhabitedLocatedCut}^{\uparrow}$ because none carries lower and upper cut rows, inhabited witnesses, locatedness moduli, regular readback, dyadic tolerance, stream windows, real sealing, transport, replay, provenance, and local NameCert together.

Closure status for $\mathsf{NameCert}_{\mathsf{InhabitedLocatedCut}^{\uparrow}}$.

Constructive story: An $\mathsf{InhabitedLocatedCut}^{\uparrow}$ packet is generated from lower-cut rows, upper-cut rows, inhabited witness rows, locatedness modulus rows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, StreamName window schedules, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InhabitedLocatedCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8537.1, Definition 8537.2, and Definition 8537.3 seed the inhabited located-cut witness packet for the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InhabitedLocatedCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter excludes Dedekind completeness, arbitrary subset suprema, quotient cut equality, selected host reals, trichotomy, countable choice, Lean verification, and bridge checking.

Upgrade path: Formal verification can encode carrier admission, inhabited witness exposure, located-cut compatibility, classifier stability, non-escape, readback factorization, tolerance accounting, StreamName scheduling, Real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.InhabitedLocatedCutUp.

Chapter 8538

A Concrete Naming Certificate for $\mathsf{ModulusCofinalFilter}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy constructions, the $\mathsf{ModulusCofinalFilter}^{\uparrow}$ packet names the directed finite route by which a family of regular-Cauchy modulus rows is refined before a real-completion consumer reads a sealed real name. It sits between the regular-sequence carrier of chapter 301, the finite-window naming discipline of chapter 126, the real seal of chapter 38, the meet surface of chapter 462, and the uniform modulus surface of chapter 1741. The packet records directed cofinal refinement without quotient filters, host choice, or ambient completeness.

Definition 8538.1 (ModulusCofinalFilter carrier)**.** *A* $\mathsf{ModulusCofinalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, M, D, W, R, S, E, H, C, Q, P, N)$$

with the following visible rows:

1. *I is the finite directed index spine for the displayed family of modulus rows.*
2. *M assigns a regular-Cauchy modulus history to each displayed index.*
3. *D records the finite common-refinement witnesses for pairs of displayed indices.*
4. *W records the* $\mathsf{StreamName}^{\uparrow}$ *window route through which each modulus row is read as finite stream data.*
5. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
6. *S records the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the regular-Cauchy readback rows are exposed.*
7. *E records meet and comparison rows through* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *and* $\mathsf{UniformCauchyModulus}^{\uparrow}$*.*
8. *H records componentwise* hsame *transport for the index spine, modulus rows, refinement witnesses, windows, readbacks, seals, and comparisons.*
9. *C records the displayed* Cont *replay routes used by stream, regular-sequence, real, meet, and uniform-modulus consumers.*
10. *Q records the refusal ledger for quotient-filter, host-choice, and ambient-completeness reads.*
11. *P is the* Pkg *provenance over the five sibling chapters named above,* BHist, hsame, Cont, *and the local naming rows.*

12. N is the local $\mathsf{NameCert}_{\mathsf{ModulusCofinalFilter}^{\uparrow}}$ *row.*

The carrier exports a directed cofinal finite-means corridor for regular-Cauchy modulus reasoning. It does not export a quotient filter, a selected host cofinal subsequence, a chosen real limit, or any ambient completeness principle.

Theorem 8538.2 (ModulusCofinalFilter regular-Cauchy handoff)**.** *Every accepted* $\mathsf{ModulusCofinalFilter}^{\uparrow}$ *packet hands a real-completion consumer only the displayed directed index spine, regular-Cauchy modulus rows, common-refinement witnesses,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and meet/uniform-modulus comparison ledger. The handoff is cofinal because every requested pair of displayed modulus rows has a visible refinement row in D before the consumer reaches S.*

Proof. Project the accepted packet through Definition 8538.1. The rows I, M, D, W, R, S, E are all visible coordinates of the same finite BHist carrier.

The window route W is read through chapter 126, and the regular-rational readback route R is read through chapter 301. These two rows keep the modulus family finite-windowed before any real seal is available.

The comparison ledger E records the meet and uniform-modulus faces using chapter 462 and chapter 1741. The seal row S is then read through chapter 38.

Since the refusal row Q has no quotient-filter, host-choice, selected-limit, or ambient-completeness coordinate, the completion consumer receives only the finite directed cofinal corridor displayed by the carrier. □

Theorem 8538.3 (ModulusCofinalFilter NameCert obligations)**.** *For every accepted* $\mathsf{ModulusCofinalFilter}^{\uparrow}$ *carrier $F = (I, M, D, W, R, S, E, H, C, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusCofinalFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, directed-index admission, regular-Cauchy modulus-row admission, common-refinement exactness, StreamName window routing, RegSeqRat readback routing, Real seal non-escape, meet and uniform-modulus comparison exactness, componentwise* hsame *transport, displayed* Cont *replay, refusal of quotient filters and host choice,* Pkg *provenance, and local naming.*

Proof. Carrier admission is the displayed finite packet. Directed-index admission reads I, modulus-row admission reads M, and common-refinement exactness reads D.

StreamName window routing, RegSeqRat readback routing, and Real seal non-escape are the rows W, R, S. They force every completion-facing read to pass through finite windows and regular-rational readbacks before the real seal is inspected.

Meet and uniform-modulus comparison exactness is the row E, transported componentwise by H and replayed only through C. Provenance and local naming are the rows P, N.

The refusal ledger Q accounts for the absent surfaces. Because no carrier coordinate stores quotient filters, a host-choice selector, a chosen real limit, or ambient completeness, none of those objects can be part of the local naming certificate. □

Falsifiable prediction: Every accepted $\mathsf{ModulusCofinalFilter}^{\uparrow}$ consumer must expose directed modulus rows, cofinal refinement witnesses, StreamName windows, RegSeqRat readbacks, the Real seal boundary, meet and uniform-modulus comparison rows, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before any real-completion read is accepted.

Independence witness: streamname, regseqrat, real, cauchymodulusmeet, uniformcauchymodulus: no listed sibling is bijective to $\mathsf{ModulusCofinalFilter}^{\uparrow}$ because this packet jointly records directed cofinal refinement of regular-Cauchy modulus rows together with window, readback, seal, comparison, transport, replay, provenance, and refusal rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCofinalFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusCofinalFilter}^{\uparrow}$ packet is generated from a finite directed index spine, regular-Cauchy modulus rows, cofinal refinement witnesses, StreamName windows, RegSeqRat readbacks, Real seal boundary, meet and uniform-modulus comparisons, componentwise hsame transport, displayed Cont replay, refusal rows, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusCofinalFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8538.1, Definition 8538.2, and Definition 8538.3 register the finite directed modulus-filter corridor used before real-completion consumers read a sealed real name.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusCofinalFilter}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient filters, selected cofinal subsequences, host choice, ambient completeness, completed real construction, or bridge checking.

Upgrade path: Scoped closure requires Lean checking of carrier admission, cofinal refinement exactness, window routing, regular-rational readback, Real seal non-escape, comparison exactness, and refusal rows for BEDC.Derived.ModulusCofinalFilterUp.

Chapter 8539

A Concrete Naming Certificate for $\mathsf{PreuniformSpace}^{\uparrow}$

Following Weil and Bourbaki uniform-space terminology, the $\mathsf{PreuniformSpace}^{\uparrow}$ packet records a finite entourage surface before the full uniform-space closure rows are available. It sits between the mature uniform interface of chapter 339, the entourage-base carrier of chapter 1819, and the preuniform-base surface of chapter 8267. The packet names the L10 substrate where neighbourhood and completion-facing consumers may inspect displayed entourage data without receiving symmetry, composition, separatedness, quotient completion, or selected-limit structure.

Definition 8539.1 (PreuniformSpace carrier)**.** *A* $\mathsf{PreuniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, E, W, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the carrier-point row, listing the finite point histories whose neighbourhood comparisons are being inspected.*
2. *E is the entourage-base row, read through* $\mathsf{EntourageBase}^{\uparrow}$ *and* $\mathsf{PreuniformBase}^{\uparrow}$ *as a finite family of displayed binary neighbourhood requests.*
3. *W is the finite neighbourhood-window row, recording the inspected point pairs, base indices, and local membership evidence available to the consumer.*
4. *T is the topology or metric handoff row, exposing only the finite neighbourhood data that may later be read by* $\mathsf{UniformSpace}^{\uparrow}$*, topological, metric, or completion-facing consumers.*
5. *H records componentwise* hsame *transport for the point row, entourage-base row, finite-window row, handoff row, provenance, and local naming row.*
6. *C records displayed* Cont *replay of the same preuniform route.*
7. *P is the* Pkg *provenance over* $\mathsf{PreuniformSpace}^{\uparrow}$*,* $\mathsf{PreuniformBase}^{\uparrow}$*,* $\mathsf{EntourageBase}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PreuniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, E, W, T, H, C, P, N. It has no coordinate for a uniform-space symmetry law, a composition-refinement law, separatedness, quotient equality of Cauchy names, selected completions, countable choice, or host powersets.

Theorem 8539.2 (PreuniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{PreuniformSpace}^{\uparrow}$ *carrier*

$$U = (X, E, W, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PreuniformSpace}^{\uparrow}}$ *surface consists exactly of carrier-point exposure through* X*, entourage-base exposure through* E*, finite neighbourhood-window exposure through* W*, topology or metric handoff exposure through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8539.1. The analytic rows are the point histories X, the displayed entourage-base row E, the finite neighbourhood windows W, and the handoff row T.

The obligation order is finite. A consumer first reads the point and entourage rows, then the inspected neighbourhood windows, and only then the topology or metric handoff surface. No row in this route asserts symmetry, composition refinement, separatedness, quotient completion, or selected limits.

The structural rows H, C, P, N transport, replay, source, and name the same displayed data. Since the excluded uniform and completion objects are absent from the carrier, the local naming surface is exhausted by the listed rows. □

Theorem 8539.3 (PreuniformSpace entourage-base route)**.** *Every neighbourhood read exported by an accepted* $\mathsf{PreuniformSpace}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{EntourageBase}^{\uparrow}, \mathsf{PreuniformBase}^{\uparrow} \longmapsto W \longmapsto T.$$

Thus a topology, metric, or completion-facing consumer receives finite preuniform neighbourhood data rather than a full $\mathsf{UniformSpace}^{\uparrow}$ *packet.*

Proof. Read the carrier $U = (X, E, W, T, H, C, P, N)$. The only row that names binary neighbourhood data is E, and it is explicitly the entourage-base row attached to the preuniform-base surface.

The finite-window row W restricts each consumer request to displayed point pairs, base indices, and membership evidence. The handoff row T may expose those finite neighbourhood reads to topology, metric, or completion-facing consumers, but it does not create uniform symmetry or composition data.

Finally replay the structural rows H, C, P, N. They preserve the same entourage-base route under componentwise transport and displayed continuation, so every exported neighbourhood read remains downstream of E and W. □

Theorem 8539.4 (PreuniformSpace uniform-consumer handoff)**.** *Any* $\mathsf{UniformSpace}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{PreuniformSpace}^{\uparrow}$ *carrier must first expose the point row* X*, the entourage-base row* E*, the finite neighbourhood window* W*, and the handoff row* T*. The handoff may be consumed as preuniform input for later uniform closure, but it does not itself export diagonal refinement, symmetry, composition, separatedness, quotient completion, countable choice, selected limits, or host powersets.*

Proof. Start with Definition 8539.2. It places every consumer-visible read inside the displayed rows X, E, W, T, H, C, P, N.

Use the entourage-base route of Definition 8539.3. The $\mathsf{UniformSpace}^{\uparrow}$ or completion-facing consumer can inspect only the preuniform neighbourhood data carried by E and filtered through W before reaching T.

The uniform laws named in the statement require carrier coordinates that the preuniform packet does not store. Therefore the handoff is a finite substrate for later uniform closure, not a proof of full uniform-space closure or a completion principle. □

Falsifiable prediction: An accepted $\mathsf{PreuniformSpace}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{UniformSpace}^{\uparrow}$, topological, metric, or completion-facing data without displaying the carrier-point row, entourage-base row, finite neighbourhood-window row, handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{PreuniformSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PreuniformSpace}^{\uparrow}$ packet is generated from displayed carrier-point rows, an entourage-base row, finite neighbourhood windows, a topology or metric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PreuniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8539.1, Definition 8539.2, Definition 8539.3, and Definition 8539.4 seed the finite preuniform route over point, entourage-base, preuniform-base, neighbourhood-window, handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PreuniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full UniformSpace laws, separatedness, quotient completion, selected limits, countable choice, host powersets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage-base exposure, neighbourhood-window exactness, topology or metric handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PreuniformSpaceU

Chapter 8540

A Concrete Naming Certificate for $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and cofinal subsequence readings, the $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ packet records the finite modulus row that lets a regular Cauchy name be read along a displayed cofinal reindexing schedule before its regular rational readback or $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits between the finite stream schedule of chapter 7081, the regular cofinal route of chapter 6960, the general handoff of chapter 1568, the dyadic tolerance carrier of chapter 291, the regular rational row of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite cofinal-modulus carrier; it does not quotient streams, choose representatives, import countable choice, identify host equality, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8540.1 (RegularCauchyReindexingModulus carrier)**.** *A* $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, I, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *schedule of the regular Cauchy name.*
2. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite requests are sent back to S.*
3. *I is the* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *cofinal route, including the monotone index map and its cofinal reachability witness.*
4. *U is the modulus row assigning to each displayed target request a source-side bound along I.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read by the modulus row.*
6. *W is the* $\mathsf{StreamReindexing}^{\uparrow}$ *finite-window replay surface for the source and target schedules.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after T, I, U, D, W have been displayed.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached after the regular rational readback and cofinal-modulus route have been replayed.*
9. *H records componentwise* hsame *transport for the schedules, route, modulus, tolerance ledger, finite windows, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from target requests through I, U, D, W, R to the terminal seal E.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamReindexing}^{\uparrow}$, $\mathsf{RegularCauchyReindexing}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexingModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected representatives, countable choice, host equality of streams or reals, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8540.2 (RegularCauchyReindexingModulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ *carrier $M = (S, T, I, U, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRe}}$ *surface consists exactly of source and target* $\mathsf{StreamName}^{\uparrow}$ *admission,* $\mathsf{RegularCauchyReindexing}^{\uparrow}$ *cofinal-route exposure, modulus-row exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{StreamReindexing}^{\uparrow}$ *finite-window replay,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every accepted read of the packet factors through

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{StreamReindexing}^{\uparrow},\quad \mathsf{RegularCauchyReindexing}^{\uparrow},\quad \mathsf{CauchyReindexing}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSe}$$

before the local NameCert *row is public. No accepted read exports quotient streams, a selected representative, countable choice, host equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8540.1. Its analytic rows are the two stream schedules S, T, the cofinal reindexing route I, the modulus row U, the dyadic tolerance ledger D, the finite-window replay W, the regular readback R, and the Real seal E.

The target schedule is read through the reindexing rows before the modulus is available. Thus U can only assign bounds along the displayed cofinal route, and the tolerance ledger D is consumed as finite $\mathsf{DyadicRatCore}^{\uparrow}$ data.

Next the $\mathsf{StreamReindexing}^{\uparrow}$ row W replays the finite windows and the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies the regular rational readback. The terminal $\mathsf{Real}^{\uparrow}$ seal E is therefore downstream of the same cofinal-modulus route.

Finally H, C, P, N transport, replay, source, and name the displayed packet. Since no row stores quotient equality, representatives, choice, host equality, trichotomy, or completeness, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ read exposes source and target StreamName schedules, a regular cofinal route, a modulus row, DyadicRatCore tolerance, StreamReindexing finite windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local NameCert before the Real seal is consumed.

Independence witness: stream-reindexing, regular-cauchy-reindexing, cauchy-reindexing, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source schedule, target schedule, cofinal route, modulus row, tolerance ledger, finite-window replay, regular rational readback, Real seal, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexingModulus}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{RegularCauchyReindexingModulus}^{\uparrow}$ packet is generated from source and target StreamName schedules, a RegularCauchyReindexing cofinal route, a modulus row, DyadicRatCore tolerance, StreamReindexing finite-window replay, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyReindexingModulus}\uparrow}$, seed)

Scope closed: Definition 8540.1 and Definition 8540.2 seed the finite cofinal-modulus route over StreamReindexing, RegularCauchyReindexing, CauchyReindexing, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyReindexingModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected representatives, countable choice, host equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-route exposure, modulus exactness, dyadic tolerance compatibility, finite-window replay, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyReindexingModulusUp.

Chapter 8541

A Concrete Naming Certificate for $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$

Following Marcinkiewicz and Stein on weak-to-strong interpolation, the $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$ packet records the finite endpoint data by which a sublinear or operator-facing request reads weak-type and strong-type rows before an interpolated $\mathsf{Real}^{\uparrow}$ norm-bound is exposed. It sits beside the analytic-strip interpolation route of chapter 5725 and the singular-integral kernel route of chapter 7928. The packet names only displayed endpoint rows, interpolation parameters, scalar ledgers, transport, replay, provenance, and local naming; it does not assert a host interpolation theorem, completed measure space, quotient function equality, selected distribution function, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8541.1 (Marcinkiewicz interpolation carrier)**.** *A* $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, S, T, \Theta, B, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the weak-type endpoint row, carrying the source exponent, target threshold ledger, finite distribution-window data, and endpoint scalar bound inspected by the packet.*
2. *S is the strong-type endpoint row, carrying the second endpoint exponent, norm-window data, and endpoint scalar bound.*
3. *T is the operator or sublinear-action row shared by the endpoint data. It records the displayed action route only, not host equality of operators or functions.*
4. *Θ is the interpolation-parameter row, storing the inspected parameter, endpoint incidence, and convex-weight ledger used to combine W and S.*
5. *B is the interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound row. It is public only after W, S, T, Θ have all been displayed.*
6. *R is the scalar readback row for endpoint constants, threshold windows, exponent comparisons, and the interpolated bound.*
7. *H stores componentwise* hsame *transport for weak endpoint data, strong endpoint data, the operator row, interpolation parameter, scalar readback, bound row, replay, provenance, and local name.*
8. *C stores displayed* Cont *replay from an interpolation request through $W, S, T, \Theta, R, B, H, P, N$.*

9. *P is the* Pkg *provenance over* MarcinkiewiczInterpolation$^{\uparrow}$, RieszThorinInterpolation$^{\uparrow}$, CalderonZygmundOperator$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MarcinkiewiczInterpolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host distribution function, selected level set, quotient function representative, measure completion, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8541.2 (Marcinkiewicz interpolation NameCert obligations)**.** *For every accepted* MarcinkiewiczInterpolat *carrier*

$$M = (W, S, T, \Theta, B, R, H, C, P, N),$$

the local NameCert$_{\text{MarcinkiewiczInterpolation}^{\uparrow}}$ *surface consists exactly of weak-type endpoint admission through W, strong-type endpoint admission through S, shared operator-action admission through T, interpolation-parameter admission through Θ, scalar readback through R, interpolated* Real$^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8541.1. The endpoint rows W and S are the only weak-type and strong-type coordinates available to the classifier, and the shared action row T is the only operator-facing coordinate.

The interpolation parameter is then forced by row order. The row Θ records the displayed endpoint incidence and convex-weight ledger, while R accounts for the scalar endpoint constants and threshold windows used by the bound row.

The bound row B is downstream of those endpoint, action, parameter, and scalar-readback rows. Finally H, C, P, N transport, replay, source, and name the same finite packet, so no excluded host distribution function, selected level set, quotient representative, measure completion, choice schedule, or ambient completeness object can appear as a local obligation. □

Theorem 8541.3 (Marcinkiewicz interpolation weak-strong window route)**.** *Every accepted interpolated-bound read for a* MarcinkiewiczInterpolation$^{\uparrow}$ *packet exposes the weak endpoint row W, strong endpoint row S, shared operator row T, interpolation parameter Θ, scalar readback row R, and terminal* Real$^{\uparrow}$ *bound row B before the interpolated estimate is visible. The route may cite the strip-interpolation sibling* RieszThorinInterpolation$^{\uparrow}$ *for endpoint comparison discipline and the singular-integral sibling* CalderonZygmundOperator$^{\uparrow}$ *for weak-type operator windows, but it exports only the displayed weak/strong window certificate.*

Proof. Start with the NameCert obligations of Definition 8541.2. They require the weak endpoint, strong endpoint, shared action, interpolation parameter, scalar readback, and bound rows to be displayed before a local name is available.

The RieszThorinInterpolation$^{\uparrow}$ sibling supplies the comparison pattern that keeps endpoint interpolation parameters explicit, while the CalderonZygmundOperator$^{\uparrow}$ sibling supplies a kernel-window example whose weak-type row can feed the endpoint surface. These are sibling routes, not imported host theorems.

The structural rows H, C, P, N preserve the same route. Since the carrier contains no coordinate for measure completion, selected distribution functions, quotient function equality, or ambient Real completeness, every interpolated-bound consumer remains inside the finite weak/strong window packet. □

Falsifiable prediction: Every accepted $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$ bound read is refuted if the weak endpoint row, strong endpoint row, shared operator row, interpolation parameter, scalar readback row, and terminal $\mathsf{Real}^{\uparrow}$ bound row are not all displayed in the same packet.

Independence witness: riesz-thorin-interpolation, calderon-zygmund-operator, real: none is bijective to $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$ because this packet jointly carries weak-type endpoint data, strong-type endpoint data, a shared operator-action row, an interpolation parameter, scalar readback, an interpolated Real bound, transport, replay, provenance, and local naming as one weak-to-strong interpolation certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MarcinkiewiczInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{MarcinkiewiczInterpolation}^{\uparrow}$ packet is generated from weak-type endpoint data, strong-type endpoint data, a shared operator-action row, an interpolation parameter, scalar readback rows, an interpolated RealUp norm-bound, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarcinkiewiczInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8541.1, Definition 8541.2, and Definition 8541.3 seed the finite weak-to-strong interpolation carrier and its displayed endpoint-window route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarcinkiewiczInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interpolation theorems, measure completion, quotient function equality, selected distribution functions, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked weak endpoint admission, strong endpoint admission, shared operator admission, interpolation-parameter admission, scalar readback, RealUp bound export, transport, replay, provenance, and local naming for BEDC.Derived.MarcinkiewiczInterpolationUp.

Chapter 8542

Finite Intersection Closed Intervals NameCert Construction

Following Bishop–Bridges constructive real analysis, the $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$ packet records the finite-intersection surface for located closed intervals inside the L10 Real and Completion route. It sits beside the Real carrier of chapter 38, the finite window source of chapter 126, the rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the located interval surface of chapter 394. The packet exposes a finite common-window witness for closed interval families; it is not open-cover compactness, arbitrary intersection compactness, trichotomy, a choice-selected point, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8542.1 (Finite intersection closed intervals carrier)**.** *A* $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, A, B, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located closed interval family row, listing finitely many interval names together with their located left and right endpoint rows.*
2. *A is the endpoint-compatibility row, recording the finite comparisons that make the listed intervals mutually admissible for a shared search window.*
3. *B is the bounded-window row, storing the finite common-window witness from which every interval row may be inspected.*
4. *W is the window-membership row, stating that each listed interval admits the same displayed window witness rather than a separately selected point.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare endpoint approximants inside the common window.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to request the finitely many precision windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row from S into rational endpoint data.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the located endpoints and the finite common-window witness.*
9. *H records componentwise* hsame *transport for L, A, B, W, D, S, R, E and the structural rows.*
10. *C records displayed* Cont *replay from a finite-intersection request through $L, A, B, W, D, S, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, A, B, W, D, S, R, E, H, C, P, N$. It contains no coordinate for open-cover compactness, arbitrary intersections, trichotomy, choice-selected points, quotient real equality, or ambient real completeness.

Theorem 8542.2 (Finite intersection closed intervals NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$ *carrier*

$$I = (L, A, B, W, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}}$ *surface consists exactly of located family admission through L, endpoint compatibility through A, common-window exposure through B, W, dyadic tolerance comparison through D, finite window scheduling through S, rational readback through R, Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8542.1. The accepted packet displays the located interval family, endpoint compatibility, common-window witness, tolerance ledger, window schedule, rational readback, Real seal, and structural rows as finite data.

The interval-facing obligations are exhausted by L, A, B, W, D, S, R, E. Endpoint compatibility is checked before the common window is read, and the dyadic, stream, regular-rational, and Real rows keep every precision request inside that finite window.

The structural obligations are exhausted by H, C, P, N. Transport, replay, provenance, and local naming preserve the same packet and do not introduce open-cover compactness, arbitrary intersection compactness, trichotomy, selected points, quotient equality, or ambient completeness. □

Theorem 8542.3 (Finite intersection closed intervals common-window handoff)**.** *Let $I = (L, A, B, W, D, S, R, E, H$ be an accepted* $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$ *carrier. Every nested-interval, compact-interval, or completion-facing consumer of I must receive the same common-window route*

$$L, A \longmapsto B, W \longmapsto D, S, R, E \longmapsto H, C, P, N.$$

Thus the consumer obtains a finite located closed-interval handoff, not open-cover compactness, arbitrary intersection compactness, trichotomy, a choice-selected point, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8542.2. They expose endpoint compatibility before any common-window witness can be read.

Use the displayed common-window rows B, W. A downstream consumer receives one finite window route for the listed interval family, so it cannot replace the witness by a selected point or an arbitrary intersection principle.

Finally read D, S, R, E, H, C, P, N as the precision and structural handoff. The dyadic tolerance, StreamName schedule, RegSeqRat readback, and Real seal remain attached to the same finite packet, while transport, replay, provenance, and local naming add no ambient compactness or completeness endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}$ packet is generated from located closed interval family rows, endpoint compatibility rows, a finite common-window witness, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8542.1, Definition 8542.2, and Definition 8542.3 seed the finite located closed-interval intersection route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntersectionClosedIntervals}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary intersection compactness, trichotomy, choice-selected points, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint compatibility, common-window handoff, dyadic tolerance, finite window scheduling, rational readback, Real sealing, structural replay, provenance, and local naming for the finite intersection closed intervals packet.

Chapter 8543

A Concrete Naming Certificate for $\mathsf{RiemannLocalization}^{\uparrow}$

Following Riemann on localization for trigonometric series, the $\mathsf{RiemannLocalization}^{\uparrow}$ packet records the finite BEDC naming surface for a local Fourier read. It sits after the Fourier data of chapter 124, the Dirichlet and Riemann kernel windows of chapter 2311, the finite observation rows of chapter 126, the regular rational readback of chapter 301, and the terminal Real-facing seal of chapter 38. The packet names a local kernel-window certificate; it does not claim global Fourier convergence, a host function, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8543.1 (Riemann localization carrier)**.** *A* $\mathsf{RiemannLocalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, K, W, T, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Fourier}^{\uparrow}$ *data row naming the finite coefficient and partial-sum observations available to the local read.*
2. *K is the Dirichlet/Riemann kernel-window row, containing the displayed neighborhood, kernel truncation, and oscillation window around the inspected point.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for coefficient, point-neighborhood, and kernel-tail requests.*
4. *T is the tail-localization budget, recording the finite bound that separates the displayed neighborhood contribution from the refused exterior tail.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row handed to Real-facing consumers after the local kernel window is displayed.*
6. *H records componentwise* hsame *transport for Fourier data, kernel windows, stream windows, tail budgets, readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from Fourier inspection through kernel-window localization, tail-budget refusal, regular readback, and Real-facing handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannLocalization}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, K, W, T, R, H, C, P, N$. It has no coordinate for a host function, unrestricted Fourier convergence theorem, quotient real equality, countable choice, or ambient Real completeness.

Theorem 8543.2 (Riemann localization NameCert obligations)**.** *For every accepted* $\mathsf{RiemannLocalization}^{\uparrow}$ *carrier $L = (F, K, W, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannLocalization}^{\uparrow}}$ *surface consists exactly of carrier admission, local-kernel window exposure, tail refusal, componentwise transport, displayed replay, provenance, and ledger nonescape. Carrier admission displays the Fourier data row F, kernel-window row K, StreamName window family W, tail-localization budget T, regular rational and Real readback row R, transport row H, replay row C, provenance row P, and local naming row N. Local-kernel exposure requires the displayed neighborhood and kernel-tail budget before a public read is formed. Tail refusal records that exterior Fourier behavior remains outside the packet. Transport, replay, provenance, and ledger nonescape keep the read on the same finite carrier surface and refuse global Fourier convergence, host functions, quotient real equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8543.1. Its rows display exactly the Fourier observations, kernel-window data, stream windows, tail budget, regular readback, transport, replay, provenance, and local naming available to the packet.

The local-kernel obligation is the row K read together with W and T. A consumer first names the displayed neighborhood and kernel-tail budget, then forms the regular rational and Real-facing readback through R.

The remaining rows preserve that route. Componentwise transport H, replay C, provenance P, and naming N keep the packet inside the displayed BHist ledger, while the absence of coordinates for host functions, global convergence, quotient real equality, choice, or ambient completeness supplies the stated nonescape. □

Theorem 8543.3 (Riemann localization local-kernel window)**.** *Every public read from an accepted* $\mathsf{RiemannLocalization}^{\uparrow}$ *carrier*

$$L = (F, K, W, T, R, H, C, P, N)$$

depends only on the displayed neighborhood in K, the kernel-tail budget T, the finite StreamName windows W, the Fourier observations in F, the readback row R, and the structural rows H, C, P, N. If a consumer asks for global Fourier convergence, a host function value, quotient equality of Real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness, the request is outside the* $\mathsf{RiemannLocalization}^{\uparrow}$ *packet.*

Proof. Read the accepted carrier. The only analytic coordinates are F, K, W, T, R, so the local read has no access to an exterior Fourier tail except through the displayed budget T.

Use the NameCert obligations of Definition 8543.2. They require every public read to pass through the local-kernel window, tail refusal, transport, replay, provenance, and local naming rows before it reaches a Real-facing consumer.

Thus the public value is a finite kernel-window readback, not a global convergence theorem or a host-side function evaluation. Since quotient real equality, countable choice, and ambient completeness are not carrier coordinates, they cannot be exported by the local-kernel window. □

Falsifiable prediction: A $\mathsf{RiemannLocalization}^{\uparrow}$ consumer that exports a local Fourier read without displaying Fourier data, a kernel-window neighborhood, StreamName windows, a tail-localization budget, RegSeqRat/Real readback, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: fourier, dirichlet-kernel, streamname, regseqrat, real: no listed sibling by itself carries Fourier observations, a Dirichlet/Riemann kernel window, finite stream windows, tail-localization refusal, regular rational and Real readback, transport, replay, provenance, and local NameCert data while refusing global Fourier convergence and host-function semantics.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannLocalization}\uparrow}$.

Constructive story: A $\mathsf{RiemannLocalization}^{\uparrow}$ packet is generated from Fourier data, a Dirichlet/Riemann kernel-window row, finite StreamName windows, a tail-localization budget, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannLocalization}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8543.1, Definition 8543.2, and Definition 8543.3 seed the local Fourier localization packet over displayed kernel windows, tail budgets, finite StreamName rows, RegSeqRat and Real readback, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannLocalization}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global Fourier convergence, host function semantics, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-kernel window exactness, tail-refusal determinacy, readback stability, transport stability, replay stability, provenance exposure, and ledger nonescape for BEDC.Derived.RiemannLocalizati

Chapter 8544

A Concrete Naming Certificate for $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ packet records the finite interval estimate that reads endpoint values, dyadic finite-difference slopes, and a displayed derivative or mean-value route before a Real-facing consumer may use a slope bound. It sits after the terminal real-name surface of chapter 38, the dyadic arithmetic core of chapter 291, the regular rational readback rows of chapter 301, the finite-difference surface of chapter 5135, the derivative boundary of chapter 122, and the theorem-facing route of chapter 1912. The packet names a finite slope-bound carrier over one displayed interval; it is not host division in a completed field, not a selected interior derivative point, not quotient real equality, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8544.1 (Bishop interval slope-bound carrier)**.** *A* $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A_0, A_1, V_0, V_1, W, D, Q, M, K, H, C, P, N)$$

with the following displayed rows:

1. A_0, A_1 *are the* $\mathsf{Real}^{\uparrow}$ *endpoint-name rows for the interval endpoints.*
2. V_0, V_1 *are the* $\mathsf{Real}^{\uparrow}$ *endpoint-value rows for the inspected map at* A_0 *and* A_1*.*
3. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row listing the rational endpoint approximants, interval-width requests, and value-window requests read by the estimate.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-difference slope ledger. It records the displayed numerator budget, nonzero interval-width witness, denominator-window comparison, and dyadic upper-bound certificate for the slope request.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the dyadic slope ledger into regular rational endpoint and slope observations.*
6. M *is the derivative or mean-value handoff row. It points to the* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DifferenceQuotient}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, *or* $\mathsf{MeanValueTheorem}^{\uparrow}$ *packet that justifies reading the displayed finite slope as theorem-facing data.*
7. K *is the classifier row. It compares two accepted packets only through the endpoint rows, endpoint-value rows, finite window, dyadic ledger, regular readback, handoff row, and structural rows.*

8. *H records componentwise* hsame *transport for endpoint names, endpoint values, slope windows, dyadic ledgers, regular readbacks, handoff rows, classifier rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for the same finite slope-bound route.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DifferenceQuotient}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalSlopeBound}^{\uparrow}}$ *row.*

The classifier has no coordinate for host real division, equality of ambient functions, a selected interior derivative point, trichotomy of endpoints, a quotient of real names, or an ambient completeness principle.

Theorem 8544.2 (Bishop interval slope-bound NameCert obligations)**.** *For every accepted* BishopIntervalSlopeBou *carrier*

$$B = (A_0, A_1, V_0, V_1, W, D, Q, M, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalSlopeBound}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *endpoint admission for A_0, A_1, V_0, V_1 through the* $\mathsf{Real}^{\uparrow}$ *boundary;*
2. *finite slope-ledger admission for W, D, Q, including the dyadic finite-difference numerator, nonzero interval-width witness, denominator-window comparison, and regular rational readback;*
3. *theorem-facing handoff through M, using only the displayed* $\mathsf{DifferenceQuotient}^{\uparrow}$, $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, *or* $\mathsf{MeanValueTheorem}^{\uparrow}$ *route;*
4. *classifier and stability transport through K, H, C over the same finite rows;*
5. *provenance and local naming through P, N.*

Every Real-facing or calculus-facing consumer of the slope bound must expose these rows before it may read the bound. The packet exports no host real division, selected interior point, quotient real equality, endpoint trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8544.1. The endpoint rows A_0, A_1, V_0, V_1 are the only real-name data visible to the packet, and the finite window W is the only place where endpoint approximants, interval-width requests, and value-window requests are listed.

Read the dyadic and regular rows next. The ledger D records the finite-difference numerator, the displayed nonzero interval-width witness, the denominator-window comparison, and the dyadic upper-bound certificate. The row Q reads this ledger back as regular rational endpoint and slope observations, so no host division in a completed field is introduced.

Now use the handoff row M. It can cite only the already displayed difference-quotient, finite-difference, derivative, or mean-value packet, so a theorem-facing consumer receives a finite slope-bound route rather than a selected interior derivative point or an ambient mean-value theorem.

Finally transport and name the packet through K, H, C, P, N. These rows compare, replay, source, and certify the same endpoint, window, dyadic, regular, and handoff data. Since the excluded objects are not carrier coordinates, the five obligation rows exhaust the local NameCert surface. □

Theorem 8544.3 (Bishop interval slope-bound classifier transport)**.** *Let B and B' be accepted* $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ *packets whose endpoint rows, endpoint-value rows, finite windows, dyadic*

slope ledgers, regular readback rows, and handoff rows are related by the displayed classifier rows. Then every accepted slope-bound read from B transports to a slope-bound read from B' through the same finite route

$$\mathsf{Real}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DifferenceQuotient}^{\uparrow}, \mathsf{FiniteDifference}^{\uparrow}, \mathsf{Derivative}^{\uparrow}, \mathsf{Me}$$

The transported read does not add host real division, quotient equality of real names, a selected derivative point, endpoint trichotomy, or ambient completeness.

Proof. Use the classifier row K from Definition 8544.1. Its comparison fields mention only endpoint names, endpoint values, finite windows, dyadic ledgers, regular readbacks, handoff rows, and structural rows.

Apply the NameCert obligations of Definition 8544.2. The slope-bound read is available only after the endpoint, ledger, handoff, transport, provenance, and local naming rows have been exposed.

The componentwise transport row H and replay row C preserve that order. The provenance and local-name rows P, N certify the transported packet without adding any refused coordinate, so the transported read remains the same finite slope-bound route. □

Theorem 8544.4 (Bishop interval slope-bound mean-value handoff)**.** *Every theorem-facing read of an accepted* $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ *packet that cites a mean-value or derivative route must first pass through the dyadic finite-difference ledger and regular rational readback rows. In symbols, the handoff row M is readable only after W, D, Q have exposed the finite endpoint-window, slope-ledger, and readback data. Thus the packet uses chapter 1912 and the derivative chapters as downstream handoffs, not as hidden sources of host-level division or selected interior points.*

Proof. Start with the carrier and obligation surface. The rows W, D, Q are listed before M, and Definition 8544.2 makes the finite slope-ledger admission a separate obligation from the theorem-facing handoff.

The mean-value and derivative chapters can therefore inspect only the displayed endpoint, dyadic, and regular rational rows. They cannot replace the finite ledger with a completed real quotient or a selected interior witness, because neither object is present in the carrier.

The structural rows preserve this dependency order under transport and replay. Hence every theorem-facing handoff remains downstream of the finite dyadic slope ledger. □

Falsifiable prediction: Every accepted $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ slope-bound read factors through displayed endpoint Real rows, a finite endpoint-window row, a DyadicRatCore finite-difference slope ledger, RegSeqRat readback, a derivative or mean-value handoff row, classifier transport, displayed replay, package provenance, and local NameCert evidence.

Independence witness: real, dyadicratcore, regseqrat, finite-difference, derivative, mean-value-theorem: $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds interval endpoints, endpoint values, finite windows, dyadic slope budgets, regular readback, theorem-facing handoff, classifier transport, replay, provenance, and local naming in one finite slope-bound certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalSlopeBound}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopIntervalSlopeBound}^{\uparrow}$ packet is generated from displayed interval endpoint names, endpoint-value rows, finite endpoint windows, a dyadic finite-difference slope ledger, regular rational readback, a derivative or mean-value handoff row, classifier transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopIntervalSlopeBound}\uparrow}$, seed)

Scope closed: Definition 8544.1, Definition 8544.2, Definition 8544.3, and Definition 8544.4 seed the finite interval slope-bound carrier, its five-row obligation surface, classifier transport, and theorem-facing handoff discipline.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopIntervalSlopeBound}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real division, quotient real equality, endpoint trichotomy, a selected interior derivative point, a host mean-value theorem, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic slope-ledger admission, regular read-back, theorem-facing handoff, classifier transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalSlopeBoundUp.

Chapter 8545

A Concrete Naming Certificate for $\mathsf{StepFunctionApproximation}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Rudin's classical step-function approximations, the $\mathsf{StepFunctionApproximation}^{\uparrow}$ packet records a finite interval-partition approximation surface for Real-facing regular-rational routes. It sits beside the dyadic endpoint ledger of chapter 291, the finite window naming discipline of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names only displayed partitions, rational values, endpoint ledgers, readback windows, transport, replay, provenance, and local naming; it is not host function equality, arbitrary partition choice, quotient real equality, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8545.1 (Step-function approximation carrier)**.** *A* $\mathsf{StepFunctionApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, V, D, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval partition row, listing adjacent bounded intervals by displayed endpoint names.*
2. *V is the rational value row assigning one rational value token to each interval cell of I.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger, recording every endpoint, tolerance cell, and comparison budget used by the partition.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which finite name windows are inspected before a cell value may be read.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational route through the finite windows and dyadic endpoint ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the partition, rational values, endpoint ledger, windows, and regular readback have all been displayed.*
7. *H records componentwise* hsame *transport for $I, V, D, W, Q, R, C, P, N$.*
8. *C records displayed* Cont *replay through partition selection, endpoint inspection, window readback, regular rational comparison, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{StepFunctionApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{StepFunctionApproximation}^{\uparrow}}$ *row naming the displayed approximation packet. The carrier has no coordinate for host-level function equality, arbitrary choice of partitions, quotient equality of reals, measure integration, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Theorem 8545.2 (Step-function approximation NameCert obligations)**.** *For every accepted* StepFunctionApproxim *carrier $A = (I, V, D, W, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StepFunctionApproximation}^{\uparrow}}$ *surface consists exactly of finite partition admission through I, rational value admission through V, dyadic endpoint and tolerance exposure through D, finite StreamName window exposure through W, RegSeqRat readback through Q, Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

No local obligation is stored outside those rows, and no accepted read can add host function equality, partition choice, quotient real equality, measure integration, or ambient Real completeness.

Proof. Project the carrier of Definition 8545.1. The rows I, V, D, W, Q, R expose the analytic data: a finite interval partition, one rational value per cell, dyadic endpoints and tolerances, finite StreamName windows, regular rational readback, and the terminal Real seal.

The structural rows then keep that data local. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same finite packet without adding a host function or a quotient real.

Since the carrier lists every available coordinate, the named rows exhaust the local obligation surface. Requests for arbitrary partition choice, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ completeness are refused before the packet can be exported to a Real-facing consumer. □

Theorem 8545.3 (Step-function approximation RegSeqRat handoff)**.** *Every accepted* $\mathsf{StepFunctionApproximation}^{\uparrow}$ *read that reaches a Real-facing consumer factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exposes only the displayed interval partition, rational value rows, dyadic endpoint ledger, StreamName windows, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local NameCert row.

Proof. Start with the partition and endpoint rows. The finite partition I can be read only together with the dyadic endpoint ledger D, so each cell boundary is carried by displayed rational and tolerance data.

Next read the StreamName and RegSeqRat rows. The window row W records the finite observation surface, and the readback row Q turns that surface into a regular rational route without invoking a host function or an unbounded choice of partitions.

Finally the Real seal R is available only downstream of I, V, D, W, Q. The transport, replay, provenance, and local naming rows preserve that same route, so any Real-facing handoff factors through the named finite surface and refuses quotient equality, measure integration, and ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{StepFunctionApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{StepFunctionApproximation}^{\uparrow}$ packet is generated from a finite interval partition, rational value rows, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StepFunctionApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 8545.1, Definition 8545.2, and Definition 8545.3 seed the finite step-function approximation route over interval partitions, rational values, dyadic endpoints, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{StepFunctionApproximation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host function equality, arbitrary choice of partitions, quotient real equality, measure integration, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked partition admission, rational value rows, dyadic endpoint exactness, StreamName window exposure, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.StepFunctionApproximationUp.

Chapter 8546

A Concrete Naming Certificate for HyperbolicSectorEnvelope$^{\uparrow}$

The HyperbolicSectorEnvelope$^{\uparrow}$ packet realizes the boundary-sector branch of chapter 9463 as finite NameCert data. It sits after the Poincare disk metric of chapter 6209, the hyperbolic metric row of chapter 3474, the visual boundary of chapter 4181, the Busemann projection of chapter 7552, and the observer-expansion packet of chapter 4167. The carrier names only finite sector endpoints, visual radius, observer window, boundary-shadow handoff, refusal, transport, replay, provenance, and local naming.

Definition 8546.1 (Hyperbolic sector-envelope carrier)**.** *A* HyperbolicSectorEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S_0, S_1, R, O, B, V, F, T, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two finite boundary-sector endpoint rows.*
2. *R is the visual-radius row delimiting the sector envelope seen by the observer.*
3. *O is the observer-window row naming the finite focus through which the sector is read.*
4. *B is the boundary-shadow handoff row tying the endpoints and visual radius to a displayed boundary-shadow surface.*
5. *V is the visual-boundary readback row reached only after S_0, S_1, R, O, B have been displayed.*
6. *F is the refusal row excluding hidden hyperbolic metric data, a completed visual boundary, Busemann projection as a host map, and selected geodesic sectors.*
7. *T is the TasteGate row requiring* Nontrivial HyperbolicSectorEnvelope$^{\uparrow}$ *and* FieldFaithful HyperbolicSectorEnvelope *instances for the displayed row inventory.*
8. *H records componentwise* hsame *transport for $S_0, S_1, R, O, B, V, F, T, C, P, N$.*
9. *C records displayed* Cont *replay through sector endpoints, visual radius, observer window, boundary shadow, visual-boundary readback, refusal, and naming.*
10. *P is the* Pkg *provenance over* HyperbolicSectorEnvelope$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, BusemannProjection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for host hyperbolic metric equality, completed boundary sectors, selected geodesic representatives, ambient visual compactification, or a global Busemann projection map.

Theorem 8546.2 (Hyperbolic sector-envelope NameCert obligations)**.** *For every accepted* HyperbolicSectorEnvelop *carrier $E = (S_0, S_1, R, O, B, V, F, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorEnvelope}^{\uparrow}}$ *surface consists exactly of endpoint admission through S_0, S_1, visual-radius admission through R, observer-window admission through O, boundary-shadow handoff through B, visual-boundary readback through V, refusal through F, TasteGate nontriviality and field-faithfulness requests through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

No accepted read may expose a boundary-sector handoff without displaying the visual-radius row, observer-window row, provenance row, and refusal row.

Proof. Project the carrier of Definition 8546.1. The endpoint rows, visual-radius row, and observer-window row are the only coordinates from which a sector envelope can be read.

The boundary-shadow and visual-boundary rows are downstream of those data. Thus a consumer cannot obtain a boundary-sector handoff until S_0, S_1, R, O, B, V have all been displayed.

Finally the refusal, TasteGate, transport, replay, provenance, and local naming rows close the certificate surface. Since the carrier contains no host metric equality, completed boundary sector, selected geodesic representative, ambient compactification, or global Busemann map, the listed obligations exhaust the accepted read. □

Theorem 8546.3 (Hyperbolic sector-envelope observer handoff)**.** *Every accepted* $\mathsf{HyperbolicSectorEnvelope}^{\uparrow}$ *observer read factors through*

$$\mathsf{PoincareDiskMetric}^{\uparrow}, \mathsf{HyperbolicMetric}^{\uparrow} \longrightarrow \mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longrightarrow \mathsf{HyperbolicVisualBoundary}^{\uparrow} \longrightarrow \mathsf{Busem}$$

The handoff exposes only finite boundary-sector endpoints, visual radius, observer window, boundary-shadow readback, visual-boundary readback, refusal, TasteGate rows, componentwise transport, replay, provenance, and local NameCert data.

Proof. Begin with the displayed sector endpoints and visual radius. They delimit the finite sector envelope as observed data, not as a completed hyperbolic boundary subset.

Next use the boundary-shadow and visual-boundary rows. These rows route the sector envelope through the Poincare disk and hyperbolic metric surfaces while keeping the observer window visible.

The Busemann projection dependency is then read only as provenance-facing handoff data. Refusal and transport preserve the same finite packet, so the observer read cannot become a host metric theorem, selected geodesic sector, global visual compactification, or ambient Busemann map. □

Falsifiable prediction: An accepted sector-envelope read cannot expose a boundary-sector handoff without displaying the visual-radius row, observer-window row, provenance row, and refusal row.

Independence witness: hyperbolicmetric, poincare-disk-metric, hyperbolic-visual-boundary, busemann-projection, real: the sector-envelope carrier jointly binds sector endpoints, visual radius, observer window, provenance, and refusal rows absent from each listed sibling.

The Lean-side TasteGate surface for this carrier is expected to include explicit Nontrivial HyperbolicSectorEnvelop and FieldFaithful HyperbolicSectorEnvelope$^{\uparrow}$ instances. Field faithfulness keeps $S_0, S_1, R, O, B, V, F, T, H, C, P, N$ as distinct rows before a classifier target may accept the packet.

Closure status for NameCert$_{\text{HyperbolicSectorEnvelope}^{\uparrow}}$ **.**

Constructive story: A HyperbolicSectorEnvelope$^{\uparrow}$ packet is generated from boundary-sector endpoints, visual radius, observer window, boundary-shadow handoff, visual-boundary readback, refusal, TasteGate requests, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicSectorEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 8546.1, Definition 8546.2, and Definition 8546.3 seed the finite observer-expansion sector-envelope route over endpoints, visual radius, observer window, boundary-shadow handoff, visual-boundary readback, refusal, TasteGate rows, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicSectorEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host hyperbolic metric equality, completed boundary sectors, selected geodesic representatives, ambient visual compactification, global Busemann projection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, visual-radius admission, observer-window admission, boundary-shadow handoff, visual-boundary readback, refusal, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicSectorEnvelopeUp.

Chapter 8547

A Concrete Naming Certificate for $\mathsf{RegularOpenSet}^{\uparrow}$

Kelley/Munkres-style grounding: a regular open set is read as an open set recovered from the interior of its closure, with its complement read through the regular closed comparison surface. The $\mathsf{RegularOpenSet}^{\uparrow}$ packet records that topology-facing object as finite BEDC data over the topology carrier of chapter 88, the closed-set boundary packet of chapter 4229, the regular-closed sibling of chapter 8221, and the metric boundary route of chapter 57. It names the displayed open-neighborhood, closure, interior, complement, transport, replay, provenance, and local certificate rows; it does not import a host powerset, detached open-set lattice, quotient equality, selected point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8547.1 (RegularOpenSet carrier)**.** *A* $\mathsf{RegularOpenSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, C, R, M, O, K, I, E, D, H, Q, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row naming the finite open-neighborhood and closure observations available to the packet.*
2. *C is the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary row used when the complement or boundary of the open set is inspected.*
3. *R is the* $\mathsf{RegularClosedSet}^{\uparrow}$ *complement row, recording that the complement side is read by the regular-closed sibling surface.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row used when closure or boundary requests are consumed through ball and distance data.*
5. *O is the open-neighborhood classifier for the named open part.*
6. *K is the closure row reached from O through T and, when needed, through the metric row M.*
7. *I is the interior-of-closure row, recording the return from K to an open row in the same topology.*
8. *E is the exactness row certifying that the displayed open row O is recovered as I.*
9. *D is the complement-compatibility row tying the outside of O to the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary row C and the* $\mathsf{RegularClosedSet}^{\uparrow}$ *row R.*
10. *H records componentwise* hsame *transport for $T, C, R, M, O, K, I, E, D, Q, P, N$.*

11. *Q records displayed* Cont *replay for the route $O \to K \to I \to E$ and for the complement route through C, R, D.*
12. *P is the* Pkg *provenance over* $\mathsf{RegularOpenSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{RegularClosedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularOpenSet}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $T, C, R, M, O, K, I, E, D, H, Q, P, N$. It has no coordinate for host powersets, detached open-set lattices, selected limit points, quotient equality of open names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8547.2 (RegularOpenSet NameCert obligations)**.** *For every accepted* $\mathsf{RegularOpenSet}^{\uparrow}$ *carrier*

$$U = (T, C, R, M, O, K, I, E, D, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularOpenSet}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, topology exposure through T, closure-interior stability along $O \to K \to I \to E$, complement compatibility through C, R, D, metric boundary access through M, componentwise* hsame *transport through H, displayed* Cont *replay through Q,* Pkg *provenance through P, and local naming through N. The certificate refuses host powersets, detached open-set lattices, quotient equality, selected points, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 8547.1. The tuple displays the topology, closed-set, regular-closed, metric, open-neighborhood, closure, interior, exactness, complement, transport, replay, provenance, and local naming rows.

Topology exposure is forced by T and O. A consumer cannot read the named open part before the finite open-neighborhood observations in T have been exposed, and any closure request must pass through the same topology row before reaching K.

Closure-interior stability is the route $O \to K \to I \to E$. The exactness row E records that the interior of the displayed closure returns to the displayed open row, so the packet exports regular openness as a finite certificate row rather than as equality in a host powerset.

Complement compatibility is supplied by C, R, D. The complement side is read through the closed-set boundary row and regular-closed sibling row before it is matched with the open side, which blocks a detached open-set lattice or independent closed-set quotient from entering the certificate.

Finally, H, Q, P, N preserve, replay, source, and name the same finite route. Since no carrier coordinate stores selected points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality, or a host powerset, those principles cannot be extracted from the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularOpenSetUp.RegularOpenSetTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularOpenSet}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularOpenSet}^{\uparrow}$ packet is generated from ambient TopologyUp observations, ClosedSetUp boundary data, RegularClosedSetUp complement data, MetricUp boundary reads, an open-neighborhood classifier, closure and interior rows, exactness, complement compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularOpenSet}^{\uparrow}}$, seed)

Scope closed: Definition 8547.1 and Definition 8547.2 seed the regular-open carrier and local NameCert obligation surface over $\mathsf{Topology}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{RegularClosedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularOpenSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host powersets, detached open-set lattices, quotient equality, selected points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or any complete lattice theorem for regular open sets.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same topology, closed-boundary, regular-closed complement, metric, closure-interior, complement-compatibility, transport, replay, provenance, and local naming rows.

Chapter 8548

A Concrete Naming Certificate for GronwallFiniteOde$^{\uparrow}$

Following Gronwall and Bishop–Bridges constructive analysis, the GronwallFiniteOde$^{\uparrow}$ packet records the finite dyadic surface by which a displayed ordinary-differential-equation row is bounded before any Real-facing consumer reads an estimate. It sits after the ODE source of chapter 193, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic arithmetic ledger of chapter 291, the Gronwall envelope of chapter 2034, and the ODE comparison route of chapter 8036. The packet names a finite constructive bound surface; it is not host existence, a selected solution curve, quotient path equality, countable choice, or ambient completeness.

Definition 8548.1 (Gronwall finite ODE carrier)**.** *A* GronwallFiniteOde$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, T, L, E, R, D, B, H, C, P, N)$$

with the following rows.

1. *M is the finite mesh row, displaying the inspected time nodes and successor windows.*
2. *T is the time-window row that bounds the interval on which the displayed ODE source is read.*
3. *L is the Lipschitz row, exposing the dyadic local constant and the source window in which it is valid.*
4. *E is the Euler or Picard comparison row that records the finite stepwise approximation route.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational approximation and error sequence.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *ledger carrying every tolerance, mesh width, Lipschitz coefficient, and accumulated error bound.*
7. *B is the bound row. It displays the Gronwall majorant read by downstream ODE or Real consumers.*
8. *H records componentwise* hsame *transport for the mesh, time, Lipschitz, comparison, readback, dyadic, bound, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the finite bound route.*
10. *P is the* Pkg *provenance over* GronwallFiniteOde$^{\uparrow}$*,* Ode$^{\uparrow}$*,* GronwallInequality$^{\uparrow}$*,* OdeComparison$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GronwallFiniteOde}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, L, E, R, D, B, H, C, P, N$. It has no coordinate for a host solution theorem, selected flow, quotient path equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8548.2 (Gronwall finite ODE NameCert obligations)**.** *For every accepted* $\mathsf{GronwallFiniteOde}^{\uparrow}$ *carrier $G = (M, T, L, E, R, D, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GronwallFiniteOde}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed tuple $M, T, L, E, R, D, B, H, C, P, N$;*
2. *mesh exposure, requiring every time step and successor window to be read through M and T;*
3. *bound stability, requiring the Lipschitz row L, comparison row E, dyadic ledger D, and bound row B to remain componentwise stable under* hsame *transport;*
4. *regular-Cauchy handoff, requiring every exported approximation and error bound to pass through R before any* $\mathsf{Real}^{\uparrow}$ *seal is read;*
5. *no-escape from host completeness, selected solution curves, quotient path equality, trichotomy, countable choice, or any estimate not already displayed in D and B.*

Proof. Project the carrier of Definition 8548.1. Carrier admission is exactly the displayed tuple $M, T, L, E, R, D, B, H, C, P, N$ and the local naming row N attached to it.

The mesh exposure row is forced by M and T. A consumer can inspect only the listed time nodes and their successor windows, so the route cannot appeal to a host interval theorem or an unbounded solution domain.

The bound-stability row then reads L, E, D, B, H. The Lipschitz data and the Euler or Picard comparison row determine which dyadic entries may be used, the ledger D records all tolerances and accumulated errors, the bound row B is the only majorant export, and H preserves these rows componentwise.

Finally the regular-Cauchy handoff uses R, while C and P replay and source the same finite route. Since no carrier coordinate names completeness, choice, selected solutions, quotient paths, or trichotomy, the five rows exhaust the local certificate. □

Theorem 8548.3 (Gronwall finite ODE dyadic bound surface)**.** *Every accepted* $\mathsf{GronwallFiniteOde}^{\uparrow}$ *packet exports an ODE estimate only through the dyadic bound row B after the mesh row M, time window T, Lipschitz row L, Euler or Picard comparison row E,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D have been exposed. The packet exports no separate analytic estimate, selected solution curve, quotient equality of paths, countable-choice schedule, or ambient-completeness claim.*

Proof. Begin with an accepted carrier $G = (M, T, L, E, R, D, B, H, C, P, N)$. The only estimate-bearing coordinates are the comparison row E, dyadic ledger D, and bound row B.

Read the mesh and time rows first. They restrict the computation to the displayed finite windows, so an estimate consumer cannot widen the route by invoking a host ODE existence theorem.

Next read the Lipschitz and comparison rows. The row L supplies the local coefficient data, while E records the finite Euler or Picard comparison route whose accumulated error is written in D.

The exported bound is then exactly B, with R repacking the rational data before any $\mathsf{Real}^{\uparrow}$ consumer sees it. The structural rows H, C, P, N transport, replay, source, and name this same route, and no remaining coordinate can export another estimate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{GronwallFiniteOde}^{\uparrow}}$.

Constructive story: A $\mathsf{GronwallFiniteOde}^{\uparrow}$ packet is generated from a finite mesh row, time-window row, Lipschitz row, Euler or Picard comparison row, RegSeqRat readback, DyadicRatCore ledger, dyadic bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GronwallFiniteOde}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8548.1, Definition 8548.2, and Definition 8548.3 seed the finite dyadic ODE bound surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GronwallFiniteOde}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host ODE existence, selected solution curves, quotient path equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formalization requires a Lean carrier and checked NameCert obligations for the mesh, time-window, Lipschitz, comparison, readback, dyadic ledger, bound, transport, replay, provenance, and naming rows.

Chapter 8549

A Concrete Naming Certificate for ContinuousDependenceOde$^{\uparrow}$

Following Picard, Lindelof, Gronwall, and Coddington–Levinson, the ContinuousDependenceOde$^{\uparrow}$ packet records the constructive ODE stability route by which two nearby local flows are compared before a downstream Real-facing consumer reads the endpoint dependence. It sits beside the ODE source of chapter 193, the Picard–Lindelof route of chapter 1888, the Gronwall inequality surface of chapter 2034, the Lipschitz ledger of chapter 410, the complete-metric convergence boundary of chapter 128, and the sealed real-name row of chapter 38. The packet names a finite continuous-dependence carrier, not a global ODE solver, a selected solution curve, quotient equality of flows, or ambient Real$^{\uparrow}$ completeness.

Definition 8549.1 (Continuous dependence ODE carrier)**.** *A* ContinuousDependenceOde$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, J, F, G, L, M, R, E, H, C, P, N)$$

with the following visible rows:

1. *I and J are the two initial time-state cells whose nearby data are compared by the packet;*
2. *F and G are the two vector-field ledgers, each read through the displayed* LipschitzMap$^{\uparrow}$ *surface on the same local window;*
3. *L is the shared Lipschitz row that keeps the two local-flow routes inside the same comparison band;*
4. *M is the Picard–Lindelof handoff row, recording the displayed local-flow route consumed by the stability comparison;*
5. *R is the* GronwallInequality$^{\uparrow}$ *comparison row, bounding the difference envelope between the two local-flow routes;*
6. *E is the* CompleteMetric$^{\uparrow}$ *convergence row, reading the displayed Cauchy comparison only after the Gronwall route has been exposed;*
7. *H records componentwise* hsame *transport for the two initial cells, the two vector-field ledgers, the Lipschitz row, the Picard–Lindelof row, the Gronwall row, the complete-metric row, the replay row, the provenance row, and the local naming row;*
8. *C records displayed* Cont *replay of the same finite dependence route;*

9. *P is the* Pkg *provenance over* $\mathsf{ContinuousDependenceOde}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousDependenceOde}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host solution-space equality, a quotient of flows, a selected continuation, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8549.2 (Continuous dependence ODE NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousDependenceOde}^{\uparrow}$ *carrier*

$$K = (I, J, F, G, L, M, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousDependenceOde}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, paired initial-cell exposure through I and J, paired vector-field exposure through F and G, Lipschitz stability through L, Picard–Lindelof handoff through M, Gronwall comparison through R, complete-metric convergence visibility through E, transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 8549.1. The packet displays two initial cells, two vector-field ledgers, one shared Lipschitz row, one Picard–Lindelof handoff, one Gronwall comparison row, one complete-metric row, and the structural transport and naming rows.

The dependence surface begins with the two initial cells. The carrier exposes them as displayed time-state data, so the comparison always starts from named local flow inputs rather than a host solution object.

The vector-field and Lipschitz rows fix the comparison band. The two ledgers are read through the same $\mathsf{LipschitzMap}^{\uparrow}$ surface, and the shared Lipschitz row L keeps the two flow routes inside one finite stability region.

The Picard–Lindelof handoff and Gronwall row then compare the two routes. The row M carries the local-flow route, while R bounds the resulting difference envelope before the complete-metric row E is read. The rows H, C, P, and N only transport, replay, source, and name this same finite dependence route, so no host equality or quotient flow can be exported. □

Theorem 8549.3 (Continuous dependence ODE Gronwall route)**.** *Every accepted* $\mathsf{ContinuousDependenceOde}^{\uparrow}$ *carrier factors its stability read through the displayed route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{LipschitzMap}^{\uparrow} \longmapsto \mathsf{PicardLindelof}^{\uparrow} \longmapsto \mathsf{GronwallInequality}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first exposes the two initial cells and two vector-field ledgers, then the shared Lipschitz row, then the Picard–Lindelof handoff, then the Gronwall comparison row, then the complete-metric convergence row, and only then the Real-facing endpoint seal. The dependence read does not bypass those rows by invoking a host ODE solver, a selected solution curve, or a quotient equality of flows.

Proof. Start with an accepted carrier $K = (I, J, F, G, L, M, R, E, H, C, P, N)$. The two initial cells and two vector-field ledgers are the only source rows available to the packet.

The shared Lipschitz row L constrains the two local-flow routes on the same finite window. That constraint is the only entry into the comparison route and keeps the two flows within the same stability band.

Apply the Picard–Lindelof handoff through M. It carries the local existence-and-uniqueness route for each source row before the comparison is read. Then apply the Gronwall row R to the displayed difference envelope. That row bounds the comparison and hands the result to the complete-metric row E.

Finally read the Real seal only after E. The rows H, C, P, and N preserve the same finite route, so the dependence theorem stays constructive and local rather than becoming a host solution-space equality or a global continuation claim. □

Theorem 8549.4 (Continuous dependence ODE real-seal nonescape)**.** *Every accepted* $\mathsf{ContinuousDependenceOde}^{\uparrow}$ *packet exposes the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the two initial cells, two vector-field ledgers, Lipschitz row, Picard–Lindelof handoff, Gronwall comparison row, and complete-metric convergence row have been displayed. The consumer therefore sees continuous dependence as finite NameCert data, not as host function equality, quotient equality of flows, selected continuation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8549.1. The displayed source rows are the two initial cells and the two vector-field ledgers; the comparison rows are the Lipschitz row, the Picard–Lindelof handoff, the Gronwall row, and the complete-metric row.

Use the NameCert obligations of Definition 8549.2. They keep the dependence route local and finite, with the Real seal reached only after the comparison rows have been opened.

Then apply the Gronwall route of Definition 8549.3. It prevents a consumer from bypassing the comparison band or the complete-metric handoff. The structural rows H, C, P, and N only transport, replay, source, and name the same finite packet, so no host equality or quotient flow can escape from the seal. □

Falsifiable prediction: Every accepted $\mathsf{ContinuousDependenceOde}^{\uparrow}$ packet exposes two initial cells, two vector-field ledgers, one Lipschitz row, one Picard–Lindelof handoff, one Gronwall row, one complete-metric row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the Real seal is read.

Independence witness: ode, picard-lindelof, gronwall-inequality, lipschitzmap, completemetric, real: none is bijective to $\mathsf{ContinuousDependenceOde}^{\uparrow}$ because this packet jointly carries paired initial cells, paired vector-field ledgers, a shared Lipschitz row, a Picard–Lindelof handoff, a Gronwall comparison row, a complete-metric row, and the transport/replay/provenance rows binding them into one continuous-dependence certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousDependenceOde}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousDependenceOde}^{\uparrow}$ packet is generated from paired initial cells, paired vector-field ledgers, one Lipschitz row, one Picard–Lindelof handoff, one Gronwall comparison row, one complete-metric row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousDependenceOde}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8549.1, Definition 8549.2, Definition 8549.3, and Definition 8549.4 bind the finite constructive ODE stability packet for paired local flows, Lipschitz control, Picard–Lindelof handoff, Gronwall comparison, complete-metric convergence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousDependenceOde}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host ODE solver, selected continuation, quotient equality of flows, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked paired-cell admission, vector-field compatibility, Lipschitz stability, Picard–Lindelof handoff, Gronwall comparison, complete-metric convergence, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousDependenceOdeUp.

Chapter 8550

A Concrete Naming Certificate for $\mathsf{BoundedGeometry}^{\uparrow}$

Following Roe's coarse geometry and Heinonen's metric-space treatment of uniform local covering, the $\mathsf{BoundedGeometry}^{\uparrow}$ packet records the finite metric surface where a displayed separation floor and a displayed local-cover budget are bound to the same metric carrier. It sits after the metric source of chapter 57, the positive-separation ledger of chapter 7444, the half-radius cover surface of chapter 6131, the cover-count ledger of chapter 8055, and the finite-net handoff of chapter 1767. The packet names bounded geometry only as finite BEDC rows; it is not local compactness, selected nets, host properness, countable choice, or an ambient compactness theorem.

Definition 8550.1 (Bounded geometry carrier)**.** *A* $\mathsf{BoundedGeometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, S, R, L, K, E, F, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point-history classifier, distance rows, and metric-law ledger.*
2. *S is the* $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$ *separation row over the same source, including the displayed positive floor and its finite comparison schedule.*
3. *R is the local radius-budget row, recording the inspected positive radii and the scale conversion used by local cover requests.*
4. *L is the* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *local-cover row, read only as finite ball-cover data for the radii named by R.*
5. *K is the* $\mathsf{MetricEntropy}^{\uparrow}$ *cover-count row, recording the finite bound attached to the displayed local covers.*
6. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *handoff row, exposing the actual center bundle and coverage-distance ledger consumed by downstream finite-net users.*
7. *F is the bounded-geometry budget ledger, binding the separation floor from S and the local cover count from K to one named metric source X.*
8. *Q records the scope-exactness check: every bounded-geometry read must use X, S, R, L, K, E, F from the same packet and the same classifier route.*

9. *H records componentwise* hsame *transport for the metric source, separation row, radius budget, local-cover row, entropy count, finite-net handoff, budget ledger, exactness row, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a bounded-geometry request through $X, S, R, L, K, E, F, Q, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{BoundedGeometry}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformlyDiscreteMetric}^{\uparrow}$, $\mathsf{DoublingMetricSpace}^{\uparrow}$, $\mathsf{MetricEntropy}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedGeometry}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise replay of these rows. It has no coordinate for arbitrary local compactness, selected maximal nets, host properness, quotient point equality, countable choice, or ambient completion.

Theorem 8550.2 (Bounded geometry NameCert obligations)**.** *For every accepted* $\mathsf{BoundedGeometry}^{\uparrow}$ *carrier*

$$B = (X, S, R, L, K, E, F, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedGeometry}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source; separation-floor visibility through S; local radius-budget visibility through R; finite local-cover visibility through L; cover-count visibility through K; finite-net handoff through E; joint budget exactness through F; same-packet scope exactness through Q; componentwise transport, displayed replay, provenance, and local naming through H, C, P, N; and refusal of local compactness, selected nets, host properness, quotient point equality, countable choice, ambient compactness, and ambient completeness as local certificate data.*

Proof. Carrier admission is projection from Definition 8550.1. The metric source X, separation row S, radius budget R, local-cover row L, entropy count K, finite-net handoff E, budget ledger F, and exactness row Q are the only analytic rows available before local naming.

Separation visibility is read through S. This exposes a positive floor over the same source X but does not decide point equality or supply a host discrete topology.

Local cover visibility is read through R, L, K, E. The radius row names the inspected scales, the doubling row supplies finite local covers, the entropy row records their finite count, and the finite-net row displays the centers and coverage distances.

The joint bounded-geometry claim is the row F. It binds the separation floor and cover-count budget to one source, while Q prevents a consumer from mixing a separation row from one metric packet with a cover budget from another.

Transport and replay are structural. The rows H, C, P, N preserve, replay, source, and name the same finite route, so no unlisted local compactness theorem, selected maximal net, host properness principle, quotient equality, countable-choice schedule, or ambient completion witness can be recovered from the local NameCert surface. □

Theorem 8550.3 (Bounded geometry finite-net handoff)**.** *Every* $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricEntropy}^{\uparrow}$, *or* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedGeometry}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{UniformlyDiscreteMetric}^{\uparrow},\quad \mathsf{DoublingMetricSpace}^{\uparrow},\quad \mathsf{MetricEntropy}^{\uparrow},\quad \mathsf{FiniteEpsilonNet}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame}$$

through the rows $X, S, R, L, K, E, F, Q, H, C, P, N$. The handoff exports a displayed separation floor, finite local-cover budget, center bundle, and coverage ledger over the same metric source; it

exports no selected maximal net, local compactness theorem, host properness assertion, quotient point equality, countable-choice schedule, or ambient compactness witness.

Proof. Begin with a finite-net consumer. The consumer reaches centers and coverage distances only through E, and E is downstream of the metric source X, separation row S, radius budget R, local-cover row L, cover-count row K, and joint budget ledger F.

For a metric-entropy consumer, the cover-count row K is visible only after the displayed local-cover route R, L has been read. The same-packet exactness row Q keeps that count attached to the separation floor in S and the source metric in X.

For a doubling consumer, the local-cover row L remains the finite ball-cover data named by the packet. It may pass its centers through E and its count through K, but it does not become an open-cover compactness or host properness principle.

The structural rows then package exactly this route. Componentwise transport H and replay C preserve X, S, R, L, K, E, F, Q, while P, N source and name the local certificate. Therefore each listed consumer remains inside the displayed finite route and cannot import selected nets, quotient point equality, countable choice, ambient compactness, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{BoundedGeometry}^{\uparrow}$ consumer that reads a finite-net center bundle, local cover count, or bounded-geometry budget without displaying the common metric source, positive separation row, local radius budget, doubling cover row, entropy count row, finite-net handoff, budget ledger, scope exactness row, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: metric, uniformly-discrete-metric, doubling-metric-space, metric-entropy, finite-epsilon-net: no listed sibling is bijective with $\mathsf{BoundedGeometry}^{\uparrow}$ because this packet jointly carries a common metric source, positive separation floor, local radius budget, finite local-cover row, finite cover-count row, finite-net center handoff, budget ledger, same-packet exactness, transport, replay, provenance, and local naming in one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedGeometry}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedGeometry}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8550.1, Definition 8550.2, and Definition 8550.3 seed the finite bounded-geometry route over metric, uniformly discrete, doubling, metric-entropy, and finite-net rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedGeometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim local compactness, selected maximal nets, host properness, quotient point equality, countable choice, ambient compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separation-floor visibility, local-cover budget exactness, finite-net handoff, same-packet exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundedGeometryUp.

Constructive story: A $\mathsf{BoundedGeometry}^{\uparrow}$ packet is generated from MetricUp or MetricSpaceUp rows, a UniformlyDiscreteMetricUp separation floor, finite radius budgets, DoublingMetricSpaceUp local covers, MetricEntropyUp cover counts, FiniteEpsilonNetUp center and coverage rows, a joint bounded-geometry budget ledger, same-packet exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8551

A Concrete Naming Certificate for $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$

Following Riesz and Bishop–Bridges constructive functional analysis, the $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ packet records the finite separation surface by which a normed vector space, a displayed proper closed subspace, and a positive tolerance expose one witness row before any quotient-space or compactness principle is available. It sits over the norm carrier of chapter 89, the vector-space carrier of chapter 47, the Banach-facing dependency of chapter 90, the terminal real seal of chapter 38, and the complete-metric boundary of chapter 128. The packet names the finite Riesz separation witness as BEDC data; it is not Hahn–Banach, not a quotient normed-space construction, not a selected supremum or nearest point, and not full Banach-space compactness.

Definition 8551.1 (Riesz lemma finite-separation carrier)**.** *A* $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, V, Y, Q, U, T, B, D, S, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Norm}^{\uparrow}$ *source row, carrying the norm comparisons used by the separation read.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row over the same source, exposing the finite addition and scalar-window data needed before a vector witness can be named.*
3. *Y is the displayed proper closed subspace row, recorded as membership, closure, and properness evidence rather than as a quotient carrier.*
4. *Q is the same-source compatibility row binding X, V, Y to one classifier route.*
5. *U is the unit-normalization row for the inspected vector window.*
6. *T is the positive tolerance row, read through* $\mathsf{Real}^{\uparrow}$ *or a dyadic real-facing seal before the separation row is visible.*
7. *B is the ball-avoidance row, listing the finite comparisons between the witness window and the displayed subspace window.*
8. *D is the norm-distance observation row, routed through* $\mathsf{CompleteMetric}^{\uparrow}$ *only as metric evidence attached to X.*
9. *S is the finite separation witness row exported by the packet.*

10. *L is the* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, visible only as a source of completion discipline and not as a compactness or choice principle.*
11. *H stores componentwise* hsame *transport for $X, V, Y, Q, U, T, B, D, S, L$.*
12. *C stores displayed* Cont *replay from a finite separation request through the analytic rows to the witness row.*
13. *P is the* Pkg *provenance over* $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Hahn–Banach extension, quotient normed spaces, selected suprema, nearest-point projection, countable choice, or full Banach-space compactness.

Theorem 8551.2 (Riesz lemma finite-separation NameCert obligations)**.** *For every accepted* $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ *carrier*

$$R = (X, V, Y, Q, U, T, B, D, S, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}}$ *surface consists exactly of carrier admission for Definition 8551.1; norm and vector-space source visibility through X, V; proper closed-subspace visibility through Y; same-source compatibility through Q; unit and tolerance visibility through U, T; finite avoidance and distance stability through B, D; witness export through S; completion-discipline visibility through L; componentwise transport, displayed replay, provenance, and local naming through H, C, P, N; and refusal of Hahn–Banach, quotient normed spaces, selected suprema, nearest-point projections, countable choice, and full Banach-space compactness as local certificate data.*

Proof. Project the rows of Definition 8551.1. The analytic source is finite: X exposes norm comparisons, V exposes vector-space operations on the inspected windows, Y exposes the closed proper subspace ledger, and Q keeps those rows attached to one classifier route.

The separation read then passes through U, T, B, D, S. The unit row and tolerance row are visible before the avoidance and distance rows, and the witness row S is exported only after those finite comparisons have been recorded.

It remains to read the structural rows. The completion row L is only a dependency witness, while H, C, P, N preserve componentwise equality, replay the displayed route, record provenance, and name the local certificate. Since the carrier contains no row for Hahn–Banach extension, quotient normed spaces, selected suprema, nearest-point projection, countable choice, or full Banach-space compactness, none of those objects can appear as a NameCert obligation. □

Falsifiable prediction: Every accepted $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ consumer that exports a separation witness without displaying the norm source, vector-space source, proper closed-subspace row, same-source compatibility row, unit-normalization row, tolerance row, avoidance row, distance row, completion-discipline row, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: norm, vecspace, banach, real, completemetric: no listed sibling is bijective with $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ because this packet jointly binds a normed vector source, proper closed-subspace evidence, unit normalization, positive tolerance, finite avoidance, distance

observation, completion discipline, witness export, transport, replay, provenance, and local naming while refusing Hahn–Banach, quotient normed spaces, selected suprema, and full Banach-space compactness.

Closure status for $\mathsf{NameCert}_{\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}$ packet is generated from NormUp, VecSpaceUp, proper closed-subspace, same-source compatibility, unit-normalization, Real-facing tolerance, avoidance, distance-observation, separation-witness, BanachUp or CompleteMetricUp dependency, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8551.1 and Definition 8551.2 seed the finite Riesz separation packet over norm, vector-space, Banach, Real, complete-metric, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszLemmaFiniteSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach, quotient normed spaces, selected suprema, nearest-point projections, countable choice, full Banach-space compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, same-source compatibility, unit and tolerance visibility, avoidance and distance stability, witness nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RieszLemmaFiniteSeparationUp.

Chapter 8552

A Concrete Naming Certificate for $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$

Following Frechet and Urysohn, the $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ packet records the completion-facing sequential-closure interface as finite BEDC NameCert data. It sits after the Frechet–Urysohn carrier of chapter 3642, the sequential-space interface of chapter 4112, the complete-metric service of chapter 128, the regular-sequence readback of chapter 5965, and the terminal real-name seal of chapter 38. The packet names the displayed sequential completion route; it does not assert quotient sequence equality, selected representatives, countable choice, ambient Real completeness, or a host completion theorem.

Definition 8552.1 (Frechet–Urysohn completion carrier)**.** *A* $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, S, M, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the accepted* $\mathsf{FrechetUrysohn}^{\uparrow}$ *source row, carrying the topological sequential-closure route.*
2. *S is the* $\mathsf{SequentialSpace}^{\uparrow}$ *source row, exposing the sequence-indexed convergence interface read by the completion consumer.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording only displayed Cauchy-window admission, limit-service requests, and convergence-distance reads.*
4. *R is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *readback row attached to the same displayed sequence windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after U, S, M, R have been displayed.*
6. *A is the completion-admission row tying the Frechet–Urysohn sequential closure read to the complete-metric service and regular-sequence readback.*
7. *H records componentwise* hsame *transport for source, sequential, complete-metric, regular-sequence, seal, admission, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from each completion-facing request through $U, S, M, R, E, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$, $\mathsf{FrechetUrysohn}^{\uparrow}$, $\mathsf{SequentialSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohnCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, S, M, R, E, A, H, C, P, N$. It has no coordinate for quotient sequence equality, selected subsequences, chosen completion representatives, countable choice, ambient Real completeness, or a host completion object.

Theorem 8552.2 (Frechet–Urysohn completion NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohnComple}$ *carrier*

$$F = (U, S, M, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetUrysohnCompletion}^{\uparrow}}$ *surface consists exactly of Frechet–Urysohn source admission through U, sequential-source admission through S, complete-metric service exposure through M, regular-sequence readback through R, terminal Real-seal ordering through E, completion admission through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8552.1. The source, sequential, complete-metric, regular-sequence, Real-seal, and admission rows are the only rows visible before transport and provenance are read.

The source row U supplies the Frechet–Urysohn sequential-closure route, while S supplies the displayed sequence-indexed convergence interface. The completion-facing consumer cannot reach the complete-metric service row M or the regular-sequence readback row R without those two displayed sources.

The seal row E is downstream of U, S, M, R. The rows A, H, C, P, N then bind the completion request to the same route, transport it componentwise, replay it, source it, and name it locally.

Since the carrier has no quotient sequence, selected representative, choice, ambient completeness, or host completion coordinate, none of those objects can be an obligation of the local certificate. □

Theorem 8552.3 (Frechet–Urysohn completion sequential density)**.** *Every completion-facing sequential-density read exported by an accepted* $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{FrechetUrysohn}^{\uparrow},\ \mathsf{SequentialSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{RegularSequenceSpace}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto A, H, C, P, N.$$

Thus the packet distinguishes Frechet–Urysohn sequential closure from an ambient completeness theorem: it exposes the finite sequence windows, complete-metric service rows, regular-sequence readback, and terminal Real seal before a density consumer may read the completion-facing admission row.

Proof. Begin with the NameCert obligations of Definition 8552.2. They force any completion-facing read to display the Frechet–Urysohn source row U and the sequential-space row S before the complete-metric or regular-sequence rows are available.

Read M and R next. The complete-metric service supplies only displayed Cauchy-window and limit-service data, while the regular-sequence row records the finite sequence readback attached to the same displayed windows.

The terminal Real seal E is then consumed as a seal, not as a proof of ambient Real completeness. The admission and structural rows A, H, C, P, N preserve the route for the density consumer.

Therefore sequential density in this packet is a finite displayed handoff from Frechet–Urysohn closure to completion-facing readback. It is not quotient stream equality, a selected dense subsequence, a chosen completion representative, countable choice, or a general completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ completion-facing read factors through a FrechetUrysohn source row, a SequentialSpace row, a CompleteMetric service row, a RegularSequenceSpace readback row, a terminal RealUp seal, completion admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: frechet-urysohn, sequential-space, completemetric, regular-sequence-space, real: $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ is not bijective to any listed sibling because it binds sequential closure, completion service, regular-sequence readback, Real sealing, transport, replay, provenance, and naming as one completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohnCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetUrysohnCompletion}^{\uparrow}$ packet is generated from a FrechetUrysohn source row, a SequentialSpace source row, a CompleteMetric service row, a RegularSequenceSpace readback row, a terminal RealUp seal, completion admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetUrysohnCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8552.1, Definition 8552.2, and Definition 8552.3 seed the completion-facing sequential-density route over Frechet–Urysohn, SequentialSpace, CompleteMetric, RegularSequenceSpace, Real-seal, admission, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetUrysohnCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected subsequences, chosen completion representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Frechet–Urysohn source exactness, SequentialSpace source routing, CompleteMetric service ordering, RegularSequenceSpace readback, Real-seal ordering, completion admission, transport, replay, provenance, and local naming for BEDC.Derived.FrechetUrysohnCompletionUp.

Chapter 8553

A Concrete Naming Certificate for $\mathsf{AbelUniformTheorem}^{\uparrow}$

Following Abel, Hardy, and Bishop–Bridges constructive analysis, the $\mathsf{AbelUniformTheorem}^{\uparrow}$ packet records the finite theorem endpoint by which Abel uniform convergence data are handed to Real-and-Completion consumers. It sits after the uniform Abel route of chapter 7086, the Abel boundary theorem of chapter 1922, the Abel mean boundary of chapter 7412, the uniform-limit packet of chapter 463, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter packages the traditional theorem as a finite NameCert route; it does not introduce host power-series semantics, selected sums, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8553.1 (AbelUniformTheorem carrier)**.** *An* $\mathsf{AbelUniformTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, U, L, M, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite radius-window row listing the displayed radial cells and endpoint-approach windows inspected by the theorem consumer.*
2. *U is the* $\mathsf{AbelUniformConvergence}^{\uparrow}$ *boundary row, carrying the already named uniform Abel convergence surface over the same radius windows.*
3. *L is the* $\mathsf{AbelLimitTheorem}^{\uparrow}$ *endpoint row, recording the boundary-limit handoff after R and U have been accepted.*
4. *M is the* $\mathsf{AbelMeanBoundary}^{\uparrow}$ *mean-boundary row, exposing the finite Abel-mean compatibility data used by the endpoint theorem.*
5. *T is the* $\mathsf{UniformLimit}^{\uparrow}$ *tail ledger, assigning one displayed tail discipline to every radius cell in R before endpoint readback.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the synchronized Abel endpoint windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after R, U, L, M, T, Q have been displayed.*
8. *H records componentwise* hsame *transport for radius windows, Abel uniform convergence data, Abel endpoint data, mean-boundary data, uniform-tail data, readback, seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from each theorem request through R, U, L, M, T, Q, E.*
10. *P is the* Pkg *provenance over* $\mathsf{AbelUniformTheorem}^{\uparrow}$, $\mathsf{AbelUniformConvergence}^{\uparrow}$, $\mathsf{AbelLimitTheorem}^{\uparrow}$, $\mathsf{AbelMeanBoundary}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelUniformTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, U, L, M, T, Q, E, H, C, P, N$. It has no coordinate for host power-series semantics, selected sums, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8553.2 (AbelUniformTheorem NameCert obligations)**.** *For every accepted* $\mathsf{AbelUniformTheorem}^{\uparrow}$ *carrier*

$$A = (R, U, L, M, T, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelUniformTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, radius-window admission through R, Abel uniform convergence boundary admission through U, Abel endpoint admission through L, Abel mean-boundary compatibility through M, uniform-tail exactness through T, regular rational readback and Real seal nonescape through Q and E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8553.1. Its theorem rows are R, U, L, M, T, Q, E, and its structural rows are H, C, P, N.

The radius and Abel boundary obligations are finite-row projections. The row R fixes the displayed radius windows, U imports the sibling $\mathsf{AbelUniformConvergence}^{\uparrow}$ surface over those windows, L records the Abel endpoint route, and M records the Abel mean-boundary compatibility data.

The tail and readback obligations are downstream. The $\mathsf{UniformLimit}^{\uparrow}$ row T binds one finite tail discipline to the displayed windows before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q is available, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is read only after R, U, L, M, T, Q have appeared.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface and exclude host power-series semantics, selected sums, quotient streams, countable choice, and ambient completeness. □

Theorem 8553.3 (AbelUniformTheorem real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{AbelUniformTheorem}^{\uparrow}$ *packet reads the endpoint only through the finite route*

$$\mathsf{AbelUniformConvergence}^{\uparrow},\ \mathsf{AbelLimitTheorem}^{\uparrow},\ \mathsf{AbelMeanBoundary}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer first receives displayed radius windows, the Abel uniform convergence boundary, the endpoint Abel limit row, the Abel mean-boundary row, and one uniform-tail ledger; only then may it consume the regular rational readback and terminal Real seal. The route exports no host power-series semantics, selected sum, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted consumer read and project the carrier. The rows R, U, L, M expose the radius windows, Abel uniform convergence boundary, Abel endpoint theorem row, and Abel mean-boundary row before any real-seal read can occur.

Apply the uniform-tail row T. It fixes the displayed tail discipline for the same radius windows and prevents a consumer from choosing a separate endpoint schedule outside the packet.

Now pass through the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and the terminal $\mathsf{Real}^{\uparrow}$ seal E. These rows are downstream of R, U, L, M, T, so the Real-facing handoff cannot bypass the finite Abel theorem route.

The structural rows H, C, P, N preserve the same route componentwise and add no analytic coordinate. Thus the packet exposes exactly the finite real-seal handoff stated above and none of the excluded host or choice principles. □

Falsifiable prediction: Every accepted AbelUniformTheoremUp read exposes radius-window rows, an AbelUniformConvergenceUp boundary row, an AbelLimitTheoremUp endpoint row, an AbelMeanBoundaryUp compatibility row, a UniformLimitUp tail ledger, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a Real-facing consumer can inspect the endpoint.

Independence witness: abel-uniform-convergence, abel-limit-theorem, abel-mean-boundary, uniformlimit, regseqrat, real: $\mathsf{AbelUniformTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries radius windows, the uniform Abel boundary, endpoint Abel theorem data, Abel mean-boundary compatibility, one uniform tail ledger, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelUniformTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelUniformTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8553.1, Definition 8553.2, and Definition 8553.3 seed the finite Real-and-Completion Abel theorem route over AbelUniformConvergence, AbelLimitTheorem, AbelMeanBoundary, UniformLimit, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelUniformTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host power-series semantics, selected sums, quotient streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-window admission, Abel uniform convergence boundary admission, Abel endpoint admission, Abel mean-boundary compatibility, uniform-tail exactness, readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbelUniformTheoremUp.

Constructive story: An $\mathsf{AbelUniformTheorem}^{\uparrow}$ packet is generated from radius-window rows, an AbelUniformConvergence boundary row, an AbelLimitTheorem endpoint row, an AbelMeanBoundary compatibility row, a UniformLimit tail ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8554

A Concrete Naming Certificate for $\mathsf{LocatedRealSupremum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealSupremum}^{\uparrow}$ packet records the supremum surface for a bounded inhabited located real set as finite NameCert data rather than as a host-order primitive. It sits beside the dyadic arithmetic core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval bisection surface of chapter 5546. The packet exports a finite L10 route for upper-bound evidence, lower-approach evidence, dyadic interval narrowing, regular readback, transport, replay, provenance, and local naming; it does not export quotient $\mathsf{Real}^{\uparrow}$ equality, trichotomy, countable choice, selected host suprema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8554.1 (Located real supremum carrier)**.** *A* $\mathsf{LocatedRealSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, U, L, D, I, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the inhabited located real-set row, recording the inspected set predicate, the displayed membership witness, and the locatedness modulus used by finite queries.*
2. *U is the upper-bound row, recording a rational or dyadic bound that every inspected member must stay below through displayed evidence.*
3. *L is the lower-approach row, recording for each displayed tolerance request the finite witness that approaches the bound from below.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row for dyadic comparison, tolerance splitting, and endpoint refinement.*
5. *I is the located interval row, read through the located interval bisection surface of chapter 5546, which keeps the narrowing interval visible before a supremum name is read.*
6. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule for the inspected bounds, approach witnesses, dyadic endpoints, and interval rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from D, I, Q.*
8. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the upper-bound, lower-approach, interval, observation, and readback rows have all been displayed.*

9. *H records componentwise* hsame *transport for set, bound, approach, dyadic, interval, observation, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from finite located queries to bound comparison, approach selection, interval narrowing, regular readback, and local NameCert exposure.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedRealSupremum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealSupremum}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient real equality, trichotomy, countable choice, a selected host supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Every supremum read must be recovered from the displayed finite upper-bound, lower-approach, dyadic, interval, observation, readback, seal, transport, replay, provenance, and naming rows.*

Theorem 8554.2 (Located real supremum NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealSupremum}^{\uparrow}$ *packet*

$$S = (A, U, L, D, I, Q, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealSupremum}^{\uparrow}}$ *obligations are exhausted by inhabited located-set exposure through A, upper-bound exposure through U, lower-approach exposure through L, dyadic arithmetic through D, located interval narrowing through I, finite observation through Q, regular rational readback through R, terminal real-name sealing through T, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8554.1. The rows A, U, L display the exact inhabited located-set evidence, upper-bound evidence, and lower-approach evidence before any real-name seal can be consumed.

Next read D, I, Q, R. The dyadic arithmetic row supplies tolerance splitting and endpoint comparison, the located interval row keeps the narrowing interval visible, the finite observation row lists the inspected queries, and the regular rational row records the completion-facing readback.

Finally read T, H, C, P, N. These rows seal the same finite route into $\mathsf{Real}^{\uparrow}$, transport it componentwise, replay the displayed construction, record its source package, and expose the local name. Since no other coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 8554.3 (Located real supremum L10 finite route)**.** *Every public* $\mathsf{LocatedRealSupremum}^{\uparrow}$ *supremum read factors through the L10 route*

$$A, U, L, D, I, Q, R, T, H, C, P, N.$$

The route may cite dyadic arithmetic, stream-name observation, regular rational readback, real-name sealing, and located interval bisection only as displayed sibling dependencies. It refuses quotient $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, countable choice, selected host suprema, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the upper-bound and lower-approach rows. The row U records that inspected members stay below the displayed bound, while L records the finite approach witnesses demanded by the located supremum read.

Transport this evidence through the L10 siblings. The dyadic row D supplies the comparison arithmetic of chapter 291; the observation row Q supplies the finite schedule of chapter 126; the readback row R supplies the regular rational surface of chapter 301; and the seal row T supplies

only the terminal real-name boundary of chapter 38. The located interval row I keeps the bisection dependency visible before the seal is read.

The structural rows H, C, P, N preserve exactly this displayed packet under transport, replay, provenance, and local naming. None of those rows contains quotient equality, trichotomy, countable choice, host supremum selection, or ambient completeness, so the public read remains a finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealSupremum}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealSupremum}^{\uparrow}$ packet is generated from inhabited located-set evidence, upper-bound evidence, lower-approach evidence, DyadicRatCore arithmetic, located interval narrowing, StreamName finite observations, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8554.1, Definition 8554.2, and Definition 8554.3 seed the finite located real supremum packet over L10 arithmetic, observation, readback, seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealSupremum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, countable choice, selected host suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-bound exposure, lower-approach exposure, dyadic arithmetic exposure, located interval narrowing, finite-window scheduling, regular rational readback, Real seal boundary control, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealSupremumUp.

Chapter 8555

A Concrete Naming Certificate for $\mathsf{DunfordIntegral}^{\uparrow}$

Following Dunford and Schwartz, the $\mathsf{DunfordIntegral}^{\uparrow}$ packet records vector-valued integration by finite weak scalar integral rows before any strong measurability or Bochner norm-integrability endpoint is available. It sits between the weak scalar route of chapter 6638, the simple-function norm route of chapter 6587, the dual observation windows of chapter 5192, and the displayed separable Banach carrier of chapter 8449. The packet is a finite weak-integration NameCert boundary: it names scalar probe integrals and comparison windows, not a completed measure-theoretic representation theorem.

Definition 8555.1 (Dunford integral carrier)**.** *A* $\mathsf{DunfordIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, I, X, V, L, W, A, B, S, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row naming the finite measurable partitions, measurable-event windows, and null-boundary comparisons visible to the packet.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *scalar readback row for the finite scalar integrals inspected by the weak probes.*
3. *X is the* $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$ *row naming the displayed Banach carrier, dense-code windows, norm readbacks, and completion-facing service used by the packet.*
4. *V is the Banach-space and dual-pairing row. It lists the inspected vector windows, dual probes, scalar pairing requests, and finite tolerance comparisons.*
5. *L is the* $\mathsf{WeakStarTopology}^{\uparrow}$ *readback row for dual-probe observation windows and weak-star scalar comparisons.*
6. *W is the weak scalar integral row. Each entry records a displayed bounded dual probe, the scalar shadow it induces, and the scalar integral read through M, I, V, L.*
7. *A is the simple-function approximation window row. It records finite simple approximants, finite range data, measurable fibres, and the weak scalar comparisons used to test the approximants.*
8. *B is the* $\mathsf{BochnerIntegral}^{\uparrow}$ *comparison handoff row, available only when the displayed simple-function and norm-tail data are already present.*
9. *S is the* $\mathsf{PettisIntegral}^{\uparrow}$ *comparison handoff row, available only through the same weak scalar integral rows and dual-pairing windows.*

10. *R is the weak-star readback and scalar-budget ledger for the probe integrals, including finite tolerance requests and comparison bounds.*
11. *H records componentwise* hsame *transport for $M, I, X, V, L, W, A, B, S, R$ and the local structural rows.*
12. *C records displayed* Cont *replay from a weak integration request through the same scalar probes, approximation windows, comparison handoffs, readback ledger, transport, provenance, and local naming rows.*
13. *P is the* Pkg *provenance over* $\mathsf{DunfordIntegral}^{\uparrow}$, $\mathsf{PettisIntegral}^{\uparrow}$, $\mathsf{BochnerIntegral}^{\uparrow}$, $\mathsf{WeakStarTopology}^{\uparrow}$, $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{DunfordIntegral}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $M, I, X, V, L, W, A, B, S, R, H, C, P, N$. It has no coordinate for strong measurability, measure completion, Radon–Nikodym conclusions, quotient function equality, countable choice, selected representatives, or Lean verification.

Lean4 ✓ `BEDC.Derived.DunfordIntegralUp.DunfordIntegralTasteGate single carrier alignment`

Theorem 8555.2 (Dunford integral NameCert obligations)**.** *For every accepted* $\mathsf{DunfordIntegral}^{\uparrow}$ *carrier*

$$D = (M, I, X, V, L, W, A, B, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DunfordIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, measure source compatibility through M, scalar integral readback through I, Banach-space and dual-pairing visibility through X, V, weak-star probe visibility through L, weak scalar integral exposure through W, simple-function approximation-window exposure through A, Pettis and Bochner comparison handoffs through S and B, weak-star scalar-budget readback through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Every weak-integration-facing consumer must read those finite rows before using the Dunford handoff. It receives no strong measurability theorem, no measure completion, no Radon–Nikodym conclusion, no quotient function equality, no countable-choice schedule, no selected vector representative, and no Lean-checked theorem from this packet.

Proof. Project the carrier in Definition 8555.1. Carrier admission exposes exactly $M, I, X, V, L, W, A, B, S, R, H, C,$

The analytic rows M, I, X, V, L, W fix the measurable source, scalar integral readback, Banach carrier, dual-pairing window, weak-star observation row, and weak scalar integrals. Thus every probe integral is a displayed scalar shadow, not a hidden vector-valued representative.

The comparison rows A, B, S, R keep the handoffs finite. Simple-function approximation windows are visible in A, Bochner comparison can be read only through B, Pettis comparison can be read only through S, and weak-star scalar budgets remain in R.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the refused objects have no carrier coordinate, the displayed rows exhaust the local NameCert obligations. □

Theorem 8555.3 (Dunford integral Pettis–Bochner handoff)**.** *Every accepted* $\mathsf{DunfordIntegral}^{\uparrow}$ *packet mediates between Pettis and Bochner consumers only through the finite route*

$$M, I, X, V, L \longmapsto W, A, R \longmapsto S, B \longmapsto H, C, P, N.$$

The Pettis-facing read consumes weak scalar integral rows and dual-pairing windows, while the Bochner-facing read additionally consumes the simple-function approximation windows and norm-tail comparison handoff. The shared Dunford packet contributes no strong measurability theorem, measure completion, Radon–Nikodym conclusion, quotient function equality, countable choice, selected representative, or ambient Banach completeness beyond the displayed carrier rows.

Proof. Begin with the weak scalar rows. By Definition 8555.2, the measurable source, scalar integrals, Banach carrier, dual-pairing windows, and weak-star observations must be exposed before any comparison handoff is available.

For the Pettis-facing read, use the row S. It is downstream of the weak scalar integral row W and the scalar-budget ledger R, so it can hand off only the displayed scalar probe integrals and not a selected vector representative.

For the Bochner-facing read, use the rows A and B. They require simple-function approximation windows and norm-tail comparison data before the Bochner comparison can be consumed.

Finally replay the structural route through H, C, P, N. The same finite rows are transported and named on both sides, and none of the refused strong-measurement, completion, Radon–Nikodym, quotient, choice, representative-selection, or Lean-verification endpoints appears as a coordinate. □

Falsifiable prediction: Every public $\mathsf{DunfordIntegral}^{\uparrow}$ read factors through displayed $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FiniteRealizedSeparableBanachSpace}^{\uparrow}$, $\mathsf{WeakStarTopology}^{\uparrow}$, weak scalar integral, simple-function approximation, Pettis handoff, Bochner handoff, weak-star readback, hsame, Cont, Pkg, and local NameCert rows. A public read that exports strong measurability, measure completion, Radon–Nikodym conclusions, quotient function equality, countable choice, selected representatives, or Lean verification from this packet refutes the carrier boundary.

Independence witness: pettis-integral, bochner-integral, weak-star-topology, finite-realized-separable-banach-space: $\mathsf{DunfordIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite weak scalar integral rows, simple-function approximation windows, Banach-space and dual-pairing rows, Pettis and Bochner comparison handoffs, weak-star readback, transport, replay, provenance, and local naming while refusing stronger measure-theoretic and quotient endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordIntegral}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8555.1, Definition 8555.2, and Definition 8555.3 seed the finite weak-integration packet over scalar probe integrals, simple-function approximation windows, dual-pairing rows, Pettis and Bochner handoffs, weak-star readback, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim strong measurability, measure completion, Radon–Nikodym conclusions, quotient function equality, countable choice, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak scalar integral exposure, approximation-window stability, Pettis and Bochner handoff nonescape, weak-star readback, transport, replay, provenance, and local naming for BEDC.Derived.DunfordIntegralUp.

Constructive story: A $\mathsf{DunfordIntegral}^{\uparrow}$ packet is generated from Measure, Integral, finite realized separable Banach, dual-pairing, WeakStarTopology, weak scalar integral, simple-function approximation, Pettis comparison, Bochner comparison, weak-star readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 8556

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$

Following Bishop–Bridges regular real analysis and Kelley on complete metric subspaces, the $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ packet records the constructive completeness of a closed bounded real interval as finite BEDC NameCert data. It sits after the interval carrier of chapter 1591, the complete-metric service of chapter 128, the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite StreamName windows of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names the interval-completeness route used by L10 completion consumers without importing open-cover compactness, selected limits, quotient equality of real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8556.1 (Closed bounded interval complete carrier)**.** *A* $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, A, Z, R, W, D, C, M, T, H, Q, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *interval row, recording the closed endpoint pair, boundedness witnesses, and membership classifier for the interval being completed.*
2. *A records the two endpoint* $\mathsf{Real}^{\uparrow}$ *seals, including the lower and upper endpoint names and their displayed endpoint-order evidence.*
3. *Z is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row for the candidate interval sequence.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row reached by the readback after the tolerance and window rows have been exposed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording exactly the finite observations of endpoint and sequence names requested by the consumer.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint slack, interval-retention comparisons, and regular-Cauchy budget cells.*
7. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row for the displayed Cauchy request.*
8. *M records membership transport, carrying the interval-membership rows from the finite tail windows to the terminal Real seal.*

9. *T records componentwise* hsame *transport for the interval row, endpoint seals, readback, StreamName windows, tolerance ledger, complete-metric limit row, membership transport, replay, provenance, and local naming.*
10. *H records displayed* Cont *replay from a completion request through $B, A, Z, R, W, D, C, M, T$.*
11. *Q is the local interval-completeness replay row, tying the Cauchy tail request, dyadic tolerance budget, and membership transport to the same closed interval.*
12. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, Z, R, W, D, C, M, T, H, Q, P, N$. No coordinate stores open-cover compactness, a selected limit functional, quotient equality of real names, countable choice, trichotomy, or completeness for ambient $\mathsf{Real}^{\uparrow}$.

Theorem 8556.2 (Closed bounded interval complete NameCert obligations)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ *carrier*

$$I = (B, A, Z, R, W, D, C, M, T, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row, endpoint* $\mathsf{Real}^{\uparrow}$ *seals,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row, membership transport, componentwise* hsame *transport, displayed* Cont *replay, local replay, provenance, and local naming. Any completion-facing consumer must read those rows before treating a displayed regular Cauchy sequence in the closed bounded interval as having an interval member as its terminal seal.*

Proof. Project the carrier of Definition 8556.1. The accepted packet displays exactly the interval row, endpoint seals, regular-Cauchy readback, terminal seal, StreamName windows, dyadic tolerance ledger, complete-metric row, membership transport, and structural rows.

The analytic route begins with B, A, W, D, Z. These rows expose the interval endpoints, finite name observations, dyadic tolerance budget, and regular rational readback before any terminal Real seal can be consumed.

The completion-facing rows are downstream. The row C supplies the complete-metric limit surface, while M transports membership from finite retained tails to the terminal seal R using only the displayed interval and tolerance data.

Finally T, H, Q, P, N preserve, replay, source, and name the same finite route. Since the carrier contains no open-cover compactness row, selected-limit functional, quotient real-name equality, countable-choice schedule, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, none of those claims belongs to the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ completion read is refuted if it reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without displaying the closed interval row, endpoint Real seals, regular-Cauchy readback, StreamName windows, dyadic tolerance ledger, complete-metric limit row, membership transport, componentwise transport, replay, provenance, and local NameCert data.

Independence witness: closedboundedinterval, completemetric, real, regseqrat, streamname, dyadicratcore: $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ is not bijective to any listed sibling because it

jointly carries a closed interval row, endpoint seals, regular-Cauchy readback, finite name windows, dyadic tolerance ledger, complete-metric limit surface, membership transport, replay, provenance, and local naming as one interval-completeness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalComplete}\uparrow}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalComplete}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp row, endpoint RealUp seals, RegSeqRatUp regular-Cauchy readback, StreamNameUp finite windows, DyadicRatCoreUp tolerance ledger, CompleteMetricUp limit row, membership transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalComplete}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8556.1 and Definition 8556.2 seed the finite interval-completeness route from closed bounded interval data to a Real-facing terminal seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalComplete}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient equality of real names, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint seal exactness, regular-Cauchy readback, finite StreamName window inspection, DyadicRatCore tolerance exactness, CompleteMetric limit handoff, membership transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalCompleteUp.

Chapter 8557

Cauchy Interleaving Tree NameCert Construction

Following Bishop and Bridges, the $\mathsf{CauchyInterleavingTree}^{\uparrow}$ packet records the finite branch tree used when two regular Cauchy readback routes are inspected together before either route is sealed as a completed real. It sits beside the regular-Cauchy shuffle of chapter 1721, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance carrier of chapter 291, and the terminal real-name route of chapter 38. The packet names diagonal, subsequence, and alternating-window reads without quotient stream equality, countable choice, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8557.1 (Cauchy interleaving tree carrier)**.** *A* $\mathsf{CauchyInterleavingTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S_0, S_1, B, D, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite schedules whose windows are being interleaved.*
2. B *is the finite branch row, recording left, right, and paired requests in the inspected tree.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to every displayed branch.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the branch schedule and dyadic tolerance ledger are exposed.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only through the displayed* S_0, S_1, B, D, R *route.*
6. H *records componentwise* hsame *transport for the two schedules, branch row, dyadic ledger, regular readback, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same interleaving route.*
8. P *is the* Pkg *provenance over* $\mathsf{CauchyInterleavingTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingTree}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host sequence selector, quotient equality of Cauchy streams, countable choice, trichotomy, or a completed-real object outside the displayed seal row.

Lean4 (statement) `BEDC.Derived.CauchyInterleavingTreeUp`

Theorem 8557.2 (Cauchy interleaving tree regularity)**.** *For every accepted* $\mathsf{CauchyInterleavingTree}^{\uparrow}$ *carrier* $T = (S_0, S_1, B, D, R, E, H, C, P, N)$*, every branch read that reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *first exposes both stream schedules* S_0, S_1*, the finite branch row* B*, and the dyadic tolerance ledger* D*. Thus the regular rational readback is a finite interleaving certificate, not a selected subsequence, quotient stream equality, or host convergence assertion.*

Proof. Project the displayed carrier rows. The row R appears only after the two finite stream schedules, the branch row, and the dyadic tolerance ledger, so a regular-rational consumer cannot read R as an independent convergence object.

Now compare with the regular-Cauchy shuffle packet of chapter 1721. That sibling records an alternating read order; the present tree keeps the branch row visible so diagonal, subsequence, and paired-window reads share the same finite tolerance ledger.

The structural rows H, C, P, N preserve, replay, source, and name that route. Since no coordinate stores a selector or quotient equality, every regularity read factors through the displayed branch tree. □

Theorem 8557.3 (Cauchy interleaving tree real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{CauchyInterleavingTree}^{\uparrow}$ *packet factors through the route*

$$S_0, S_1, B, D, R \longmapsto E.$$

The seal may be consumed by Real and Completion arguments as a finite branch-structured readback of regular Cauchy data, but it does not assert ambient real completeness, quotient stream equality, countable choice, or a selected limiting subsequence.

Proof. Read the seal coordinate E in the carrier. Its only public input route is the finite pair of stream schedules, the branch row, the dyadic tolerance ledger, and the regular rational readback.

The replay row C and provenance row P keep the same order visible to downstream Real and Completion consumers. A consumer may therefore cite the packet as a branch-structured handoff from regular Cauchy data to a real-name seal.

No additional row chooses a limit, compares quotient streams, or discharges completeness. Hence the exported seal is exactly the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyInterleavingTree}^{\uparrow}$ real-seal read exposes the branch row B before the $\mathsf{RegSeqRat}^{\uparrow}$ readback R and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Independence witness: regular-cauchy-shuffle, streamname, regseqrat, dyadicratcore, real: the regular-Cauchy shuffle records an alternating schedule without the branch tree; StreamName records finite windows without regular readback; RegSeqRat records regular rational data without a two-source branch row; DyadicRatCore records tolerance arithmetic without a real seal; and RealUp seals a name without the interleaving carrier. None is bijective to $\mathsf{CauchyInterleavingTree}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyInterleavingTree}^{\uparrow}$ packet is generated from two finite stream schedules, a branch row, a dyadic tolerance ledger, a regular rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8557.1, Definition 8557.2, and Definition 8557.3 seed the finite branch tree used by diagonal, subsequence, and regular-Cauchy readback arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingTree}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CauchyInterleavingTreeUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, countable choice, selected limits, ambient Real$^{\uparrow}$ completeness, Lean theorem verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, branch-row stability, dyadic tolerance compatibility, regular rational readback, real-seal delivery, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterleavingTreeUp

Chapter 8558

MetaCIC Residual Join Envelope NameCert Construction

The $\mathsf{MetaCICResidualJoinEnvelope}^{\uparrow}$ packet realizes the residual-join route of the visions hub as a finite MetaCIC NameCert envelope. It sits between the finite residual join window of chapter 8021, the confluence diamond ledger of chapter 6547, the local join ledger of chapter 4126, the residual-substitution boundary of chapter 2181, and the parallel-diamond frontier of chapter 4238. The packet records a bounded residual-window, candidate-normalization, local-join, replay, provenance, and handoff envelope without claiming full closed confluence, strong normalization, subject reduction, evaluator equality, or kernel consistency.

Definition 8558.1 (MetaCIC residual join envelope carrier)**.** *A* $\mathsf{MetaCICResidualJoinEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (W, R, L, C, D, H, T, P, N)$$

with the following displayed coordinates:

1. *W is the finite residual-window row, exposing the source term, two residual branches, and bounded context.*
2. *R is the retained residual-substitution premise row consumed from the sibling boundary.*
3. *L is the local join row, recording candidate common-reduct data inside the displayed window.*
4. *C is the candidate-normalization row, available only as a bounded witness for the displayed residual window.*
5. *D is the parallel-diamond handoff row, read only after W, R, L, C are visible.*
6. *H records componentwise* hsame *transport for the residual window, premise, local join, candidate-normalization, handoff, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay of the residual-join route.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualJoinEnvelope}^{\uparrow}$*,* $\mathsf{FiniteResidualJoinWindow}^{\uparrow}$*,* $\mathsf{MetaCICConfluenceDia}$ $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$*,* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$*,* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted Church–Rosser, global strong normalization, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Lean4 (statement) `BEDC.Derived.MetaCICResidualJoinEnvelopeUp`

Theorem 8558.2 (MetaCIC residual join envelope diamond handoff)**.** *Every parallel-diamond handoff exported by an accepted* MetaCICResidualJoinEnvelope$^{\uparrow}$ *carrier* $J = (W, R, L, C, D, H, T, P, N)$ *factors through the finite route*

$$W, R, L, C \longmapsto D.$$

The handoff may feed the MetaCICParallelDiamondFrontier$^{\uparrow}$ *consumer as a bounded residual-join envelope, but it does not discharge the residual-substitution premise, prove unrestricted confluence, prove strong normalization, prove subject reduction, decide evaluator equality, or establish kernel consistency.*

Proof. Inspect the displayed carrier. The handoff row D is not public until the residual window W, retained premise R, local join row L, and candidate-normalization row C have all been read.

The sibling finite residual join window supplies only the bounded window shape, while the confluence diamond ledger and local join ledger record the local common-reduct data. The residual-substitution boundary remains a premise row, so the handoff cannot erase it.

The structural rows H, T, P, N transport, replay, source, and name the same finite route. Therefore the exported diamond-facing datum is the bounded envelope W, R, L, C, D and nothing stronger. □

Falsifiable prediction: Every accepted MetaCICResidualJoinEnvelope$^{\uparrow}$ handoff exposes the retained residual-substitution premise row R before the parallel-diamond row D is consumed.

Independence witness: finite-residual-join-window, metacic-confluence-diamond-ledger, metacic-local-join-ledger, metacic-residual-substitution-boundary, metacic-parallel-diamond-frontier: the finite residual join window records bounded residual data without the handoff envelope; the confluence diamond ledger records diamond-facing rows without this retained-premise packaging; the local join ledger records common-reduct data without the candidate-normalization row; the residual-substitution boundary records the premise without a handoff; and the parallel-diamond frontier consumes a conditional frontier rather than the residual-join envelope. None is bijective to MetaCICResidualJoinEnvelope$^{\uparrow}$.

The Lean-side TasteGate surface is expected to provide explicit `Nontrivial` and `FieldFaithful` instances for `BEDC.Derived.MetaCICResidualJoinEnvelopeUp`. Field faithfulness must keep W, R, L, C, D, H, T, P as distinct rows before any residual-diamond consumer reads the envelope.

Closure status for NameCert$_{\text{MetaCICResidualJoinEnvelope}^{\uparrow}}$.

Constructive story: A MetaCICResidualJoinEnvelope$^{\uparrow}$ packet is generated from a finite residual window, retained residual-substitution premise, local join row, candidate-normalization row, parallel-diamond handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICResidualJoinEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 8558.1 and Definition 8558.2 seed the finite residual-window, local-join, candidate-normalization, and parallel-diamond handoff envelope.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICResidualJoinEnvelope}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.MetaCICResidualJoinEnvelopeUp`

Bridge status: none

Not claimed: This chapter does not close residual-substitution discharge, full closed confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, kernel consistency, Lean theorem verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, residual-window exposure, retained-premise visibility, local-join accounting, candidate-normalization accounting, handoff nonescape, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICResidualJoinEnvelopeUp.

Chapter 8559

A Concrete Naming Certificate for MetaCICBoundedDiamondWindow$^{\uparrow}$

The MetaCICBoundedDiamondWindow$^{\uparrow}$ packet realizes the bounded-window handoff between candidate-mediated normalization in chapter 9383 and residual confluence visibility in chapter 9378. It sits beside the candidate normalization and confluence handoff of chapter 5547, the parallel-diamond frontier of chapter 4238, the proof-trace normal-form packet of chapter 5157, the finite-window confluence packet of chapter 6027, and the machine export boundary of chapter 5205. The packet records a finite audit window for candidate evidence, normal-form readback, residual-frontier rows, local diamond visibility, transport, replay, provenance, and local naming without claiming global strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.

Definition 8559.1 (MetaCIC bounded diamond window carrier)**.** *A* MetaCICBoundedDiamondWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (W, C, N, R, J, T, M, H, K, P, L)$$

with the following displayed rows:

1. *W is the bounded closed-term and redex-window row.*
2. *C is the candidate-evidence row consumed before any normalization handoff is read.*
3. *N is the normal-form readback row, compatible with the proof-trace normal-form sibling.*
4. *R is the residual-frontier row that records the two bounded branches and their retained obstruction surface.*
5. *J is the local diamond visibility row for candidate join data inside the same window.*
6. *T is the finite proof-trace row linking the normal-form and residual-frontier reads.*
7. *M is the machine export boundary row, recording which bounded evidence can be emitted without becoming a kernel oracle.*
8. *H records componentwise* hsame *transport for the window, candidate, normal-form, residual-frontier, local-join, proof-trace, and export-boundary rows.*
9. *K is the displayed* Cont *replay row for the bounded audit route.*

10. *P is the* Pkg *provenance over* MetaCICBoundedDiamondWindow$^{\uparrow}$, MetaCICCandidateNormalizationConfluenceHan MetaCICParallelDiamondFrontier$^{\uparrow}$, ProofTraceNormalForm$^{\uparrow}$, FiniteWindowConfluence$^{\uparrow}$, MachineExportBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *L is the local* NameCert$_{\text{MetaCICBoundedDiamondWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global strong-normalization theorem, global confluence theorem, subject reduction, evaluator equality, quotient syntax equality, kernel consistency, or unbounded machine export.

Theorem 8559.2 (MetaCIC bounded diamond window NameCert obligations)**.** *For every accepted* MetaCICBoundedDiamondWindow$^{\uparrow}$ *carrier $D = (W, C, N, R, J, T, M, H, K, P, L)$, the local* NameCert$_{\text{MetaCICBoundedDiamondWindow}^{\uparrow}}$ *surface consists exactly of bounded window exposure through W, candidate-evidence exposure through C, normal-form readback through N, residual-frontier exposure through R, local diamond visibility through J, proof-trace linkage through T, machine export boundary through M, componentwise* hsame *transport through H, displayed* Cont *replay through K, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8559.1. The window row W fixes the closed-term and redex scope before any candidate or frontier evidence can be consumed.

Rows C, N, R, J, T form the handoff surface. Candidate evidence precedes the normalization readback, the residual frontier keeps both bounded branches visible, and the local diamond row records only the join data inside the same window.

Rows M, H, K, P, L preserve the boundary. Machine export remains mediated by the export boundary, while transport, replay, provenance, and local naming carry the same finite packet. Hence the listed rows exhaust the NameCert obligations and add no global normalization, global confluence, evaluator equality, quotient syntax equality, or kernel-consistency coordinate. □

Theorem 8559.3 (MetaCIC bounded diamond window nonescape)**.** *Every* MetaCICBoundedDiamondWindow$^{\uparrow}$ *consumer factors through*

$$W, C, N, R, J, T, M, H, K, P, L.$$

If a bounded window omits either candidate evidence or residual-frontier rows, it cannot certify both strong-normalization handoff and local diamond visibility. In particular, the packet cannot be consumed as a global strong-normalization theorem, unrestricted confluence theorem, subject-reduction theorem, evaluator oracle, quotient syntax equality, kernel-consistency certificate, or unbounded machine-export route.

Proof. Start with the obligation surface. It makes candidate evidence C and residual-frontier evidence R mandatory rows before the normal-form readback N and local diamond row J are public.

The sibling handoff and frontier chapters are consumed through the displayed packet. The candidate-normalization handoff supplies candidate-mediated normalization data, while the parallel-diamond and finite-window confluence siblings supply only bounded residual visibility.

The proof-trace and machine-export rows keep the route inspectable. They record which evidence is replayed and which bounded rows can be emitted, without adding a kernel oracle or an unbounded export channel. Thus a consumer missing C or R leaves the carrier, and the accepted packet cannot escape to any of the listed global claims. □

Falsifiable prediction: If a bounded window omits either candidate evidence or residual-frontier rows, it cannot certify both SN handoff and local diamond visibility.

Independence witness: metacic-candidate-normalization-confluence-handoff, metacic-parallel-diamond-frontier, proof-trace-normal-form, finite-window-confluence, machine-export-boundary: the candidate-normalization handoff omits the machine export boundary row; the parallel-diamond frontier retains residual confluence rows without the normal-form proof trace; the proof-trace normal form omits the residual-frontier join row; the finite-window confluence packet omits candidate evidence; and the machine export boundary omits the local diamond visibility row. None is row-bijective to $\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}$.

The future Lean TasteGate surface must provide `Nontrivial` and `FieldFaithful` instances for `BEDC.Derived.MetaCICBoundedDiamondWindowUp`. Field faithfulness must keep $W, C, N, R, J, T, M, H, K, P, L$ distinct before any normalization or confluence consumer reads the bounded window.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}$ packet is generated from a bounded closed-term redex window, candidate evidence, normal-form readback, residual-frontier rows, local diamond visibility, proof-trace linkage, machine export boundary, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8559.1, Definition 8559.2, and Definition 8559.3 seed the bounded candidate-normalization and residual-diamond handoff window.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient syntax equality, kernel consistency, Lean verification, axiom-clean audit, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence, normal-form readback, residual-frontier exposure, local diamond visibility, proof-trace linkage, machine export boundary, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICBoundedDiamondWindow

Chapter 8560

A Concrete Naming Certificate for MetaCICBoundedChurchRosserFragment$^{\uparrow}$

The MetaCICBoundedChurchRosserFragment$^{\uparrow}$ packet realizes the conditional confluence boundary of section 9378.3 as finite NameCert data. It sits beside the critical-path source of chapter 765, the typed parallel-diamond boundary of chapter 6433, the parallel-diamond frontier of chapter 4238, and the confluence audit packet of chapter 735. The packet records atom fragments, normal-source fragments, conditional diamond rows, obstruction retention, transport, replay, provenance, and local naming without claiming full Church–Rosser, residual-substitution discharge, subject reduction, quotient syntax equality, or kernel consistency.

Definition 8560.1 (MetaCIC bounded Church–Rosser fragment carrier)**.** *A* MetaCICBoundedChurchRosserFragme *carrier is a finite* BHist *packet*

$$M = (A, N, C, O, T, R, P, S, H)$$

with the following visible rows:

1. *A is the atom fragment row, retaining the unconditional atom joins already exposed by the checked confluence substrate;*
2. *N is the normal-source row, retaining the unconditional closed-normal fragments already exposed by the checked substrate;*
3. *C is the conditional-diamond row, recording the substitution-sensitive bridge to the parallel-diamond frontier;*
4. *O is the obstruction row, keeping the residual-substitution boundary visible rather than silently discharged;*
5. *T is the typed-window row, recording the finite confluence input window read by the audit map;*
6. *R is the residual-frontier row, tracking the residual-premise budget available to the bridge;*
7. *P is the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *provenance row consumed by the fragment;*
8. *S is the* MetaCICTypedParallelDiamondBoundary$^{\uparrow}$ *source row, exposing the typed boundary before the conditional bridge is read;*
9. *H records componentwise* hsame *transport for the atom, normal-source, conditional-diamond, obstruction, typed-window, residual-frontier, provenance, source, and local naming rows.*

The carrier has no coordinate for full Church–Rosser, residual-substitution discharge, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency.

Theorem 8560.2 (MetaCIC bounded Church–Rosser fragment NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}$ *carrier*

$$M = (A, N, C, O, T, R, P, S, H),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, atom-fragment exposure through A, normal-source exposure through N, conditional-diamond exposure through C, obstruction retention through O, typed-window exposure through T, residual-frontier exposure through R, parallel-frontier provenance through P, typed-boundary source exposure through S, and componentwise transport through H.*

Proof. Project the carrier of Definition 8560.1. The packet displays the atom and normal-source fragments, the conditional diamond row, the obstruction row, the typed window, the residual frontier, the provenance row, the typed boundary source, and the transport row.

Atom and normal-source exposure are the unconditional part of the packet. Those rows record the checked fragments already available from the confluence substrate, so the packet can name them without turning them into a global confluence theorem.

The conditional-diamond row C and obstruction row O keep the residual-substitution boundary visible. The typed-window row T, residual-frontier row R, provenance row P, and source row S place that conditional bridge inside the same finite audit surface rather than outside it.

The structural row H only transports the displayed packet. Because the carrier has no coordinate for full Church–Rosser, residual-substitution discharge, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 8560.3 (MetaCIC bounded Church–Rosser fragment conditional route)**.** *Every accepted* $\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}$ *packet factors its confluence-facing read through the displayed route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICTypedParallelDiamondBoundary}^{\uparrow} \longmapsto \mathsf{MetaCICParallelDiamondFrontier}^{\uparrow} \longmapsto \mathsf{Meta}$$

The route first opens the critical path, then the typed boundary, then the parallel frontier, and only then the audit packet. The fragment therefore names a bounded Church–Rosser boundary, not a full closed Church–Rosser theorem or a residual-substitution discharge.

Proof. Start with the critical-path surface. It supplies the finite source package from which the confluence audit map reads its candidate evidence.

Next read the typed parallel-diamond boundary. That boundary keeps the typed-window and residual-substitution rows visible before any conditional bridge is consumed.

Then read the parallel-diamond frontier. The frontier keeps the local-join and obstruction rows visible, so the route remains conditional rather than unconditional.

Finally consume the audit packet. The route reaches the confluence audit map only after the checked fragments and boundary rows have been displayed, so it cannot be reinterpreted as full Church–Rosser, residual-substitution discharge, or kernel consistency. □

Theorem 8560.4 (MetaCIC bounded Church–Rosser fragment nonescape)**.** *Every accepted* $\mathsf{MetaCICBoundedChurc}$ *consumer must keep the atom fragment, normal-source fragment, conditional-diamond row, obstruction row, typed-window row, residual-frontier row, provenance row, and typed-boundary source*

row visible before citing the packet as confluence evidence. No consumer may replace that route by full Church–Rosser, residual-substitution discharge, subject reduction, quotient syntax equality, or kernel consistency.

Proof. Project the accepted carrier of Definition 8560.1. The visible rows are atom, normal-source, conditional diamond, obstruction, typed-window, residual-frontier, provenance, source, and transport.

Apply the NameCert obligations of Definition 8560.2. They keep the unconditional fragments and the conditional bridge separate, so a consumer must display the boundary rows before the fragment may be cited.

Use the conditional route of Definition 8560.3. It forces the confluence-facing read through the critical-path, typed-boundary, frontier, and audit rows, and therefore prevents the packet from being cited as full Church–Rosser or a residual-substitution discharge. □

Falsifiable prediction: An accepted $\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}$ packet that is cited as confluence evidence must display atom-fragment, normal-source, conditional-diamond, obstruction, typed-window, residual-frontier, provenance, source, and transport rows.

Independence witness: metaciccriticalpath, metacic-parallel-diamond-frontier, metacic-typed-parallel-diamond-boundary, metacicconfluenceauditpacket: none is bijective to MetaCICBoundedChurchRosserFragm because this packet jointly carries the atom fragment, normal-source fragment, conditional-diamond row, obstruction row, typed-window row, residual-frontier row, provenance row, source row, and transport row that keep the conditional boundary finite. The Lean TasteGate.lean follow-up must provide Nontrivial and FieldFaithful instances for the same carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}$ packet is generated from atom fragments, normal-source fragments, a conditional-diamond row, an obstruction row, a typed-window row, a residual-frontier row, a parallel-frontier provenance row, a typed-boundary source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8560.1, Definition 8560.2, Definition 8560.3, and Definition 8560.4 bind the finite conditional confluence boundary for atom fragments, normal-source fragments, typed windows, residual frontiers, obstruction retention, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICBoundedChurchRosserFragment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Church–Rosser, residual-substitution discharge, subject reduction, quotient syntax equality, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked atom-fragment exposure, normal-source exposure, conditional-diamond routing, obstruction retention, typed-window exposure, residual-frontier exposure, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICBoundedChurchRosserFragmentUp.

Chapter 8561

A Concrete Naming Certificate for $\mathsf{TightApartness}^{\uparrow}$

Following Bishop and Bridges, a tight apartness relation is read as positive separation evidence whose non-apartness boundary agrees with the displayed equality classifier. The $\mathsf{TightApartness}^{\uparrow}$ packet sits between the constructive apartness surface of chapter 1380, the positive-distance route of chapter 3505, and the terminal real-name seal of chapter 38. It gives equality-facing consumers a finite apartness boundary without importing trichotomy, quotient equality, selected points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8561.1 (Tight apartness carrier)**.** *A* $\mathsf{TightApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, E, T, S, C, H, R, P, N)$$

with the following displayed rows:

1. *A is the apartness evidence row. It records the positive separation branch read from a* $\mathsf{LocatedApartness}^{\uparrow}$ *or* $\mathsf{MetricApartness}^{\uparrow}$ *source before any equality-facing conclusion is exported.*
2. *E is the equality/classifier row. It records the classifier boundary whose accepted equality branch is the only branch allowed to match non-apartness.*
3. *T is the tightness row. It states that a displayed failure to produce apartness at the inspected boundary routes to the classifier row E, rather than to trichotomy or host negation.*
4. *S is the symmetry and cotransitivity-facing stability row. It keeps the positive apartness evidence stable under endpoint swap and under finite cotransitivity witnesses.*
5. *C records displayed* Cont *replay from an apartness or equality-boundary request through A, E, T, S, H, R, P, N.*
6. *H records componentwise* hsame *transport for apartness evidence, classifier rows, tightness boundaries, stability rows, replay, provenance, and the local name.*
7. *R is the Real-facing readback row. It allows a downstream* $\mathsf{Real}^{\uparrow}$ *consumer to inspect the same positive gap or non-apartness boundary without receiving a completed real equality theorem.*
8. *P is the* Pkg *provenance over* $\mathsf{TightApartness}^{\uparrow}$*,* $\mathsf{LocatedApartness}^{\uparrow}$*,* $\mathsf{MetricApartness}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TightApartness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, E, T, S, C, H, R, P, N$. It has no coordinate for trichotomy, quotient equality, selected points, countable choice, or ambient real completeness.

Theorem 8561.2 (Tight apartness NameCert obligations)**.** *For every accepted* $\mathsf{TightApartness}^{\uparrow}$ *carrier* $K = (A, E, T, S, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightApartness}^{\uparrow}}$ *surface consists exactly of carrier admission, apartness-evidence visibility, classifier-boundary visibility, tightness of non-apartness against the classifier row, symmetry and cotransitivity-facing stability, displayed replay, componentwise transport, Real-facing readback, provenance, and local naming. Every obligation is checked along the finite route*

$$A \longmapsto E \longmapsto T \longmapsto S \longmapsto C, H, R, P, N.$$

No obligation row exports trichotomy, quotient equality, selected points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8561.1. The row A is the only source of positive apartness evidence, and the row E is the only equality/classifier boundary available to an equality-facing consumer.

The tightness row T binds non-apartness to E. Since the carrier has no total comparison row, this boundary is a finite classifier handoff rather than a trichotomy decision or host negation principle.

The stability row S keeps the apartness branch attached under endpoint swap and finite cotransitivity-facing transport. The rows C, H, R, P, N then replay, transport, read back, source, and name the same finite route.

Thus the local NameCert surface is exhausted by the listed rows. The excluded trichotomy, quotient, selection, choice, and real-completeness data are not carrier coordinates. □

Theorem 8561.3 (Tight apartness Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{TightApartness}^{\uparrow}$ *packet must first read the apartness evidence row A, the equality/classifier row E, the tightness row T, and the stability row S, and only then consume C, H, R, P, N. The handoff therefore exposes a finite apartness-based equality boundary compatible with chapter 1380 and chapter 3505, while keeping the terminal real-name seal of chapter 38 downstream of displayed evidence.*

Proof. Begin with an equality-facing Real request. The carrier permits no direct read of host real equality, so the request first opens the apartness evidence row A and the classifier row E.

Read tightness through T. A non-apartness boundary is accepted only as the displayed classifier boundary, while a positive branch remains the evidence row named by A.

Use S to preserve this route under symmetry and finite cotransitivity-facing witnesses. Finally C, H, R, P, N replay, transport, read back, source, and name the same packet for the Real-facing consumer.

The consumer therefore receives an apartness-based equality boundary and no trichotomy, quotient equality, selected point, countable-choice schedule, or ambient real-completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{TightApartness}^{\uparrow}$ read factors through apartness evidence, an equality/classifier row, a tightness boundary, symmetry and cotransitivity-facing stability, replay, transport, Real-facing readback, provenance, and local naming before any equality-facing consumer reads a conclusion.

Independence witness: located-apartness, metric-apartness, real: $\mathsf{TightApartness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries positive apartness evidence, a non-apartness-to-classifier boundary, stability rows, Real-facing readback, transport, replay, provenance, and local naming as one tightness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TightApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{TightApartness}^{\uparrow}$ packet is generated from apartness evidence, an equality/classifier row, a tightness boundary, symmetry and cotransitivity-facing stability, displayed Cont replay, componentwise hsame transport, Real-facing readback, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8561.1, Definition 8561.2, and Definition 8561.3 seed the finite apartness-based equality boundary over located apartness, metric apartness, and Real-facing readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient equality, selected points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-evidence visibility, classifier-boundary visibility, tightness of non-apartness, stability, replay, transport, Real-facing readback, provenance, and local naming for BEDC.Derived.TightApartnessUp.

Chapter 8562

A Concrete Naming Certificate for $\mathsf{CompletionExponentialLaw}^{\uparrow}$

Following Mac Lane and Bishop–Bridges, the $\mathsf{CompletionExponentialLaw}^{\uparrow}$ packet records the finite completion-law surface by which a completion functor distributes over an exponential-style function object only through displayed completion rows. It sits after the functor packet of chapter 423, the monad packet of chapter 378, the unit packet of chapter 1527, the multiplication packet of chapter 1829, and the universal-property packet of chapter 397. The chapter names a standard completion-law handoff as finite NameCert data; it is not host cartesian closedness, not quotient equality of completed maps, not a selected limit, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8562.1 (Completion exponential law carrier)**.** *A* $\mathsf{CompletionExponentialLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, M, U, T, E, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *exposure row, recording the completion functor read used by the law.*
2. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row, recording the shared monad surface consumed by the unit and multiplication faces.*
3. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row, naming the unit side of the exponential-law comparison.*
4. *T is the* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *row, naming the multiplication side of the comparison.*
5. *E is the exponential-function window row, listing the displayed source and target function windows inspected by the packet.*
6. *W is the finite completion-window row, recording the common finite Cauchy windows through which both sides are read.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used for rational regularity checks along the finite windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the two completion reads.*
9. *H records componentwise* hsame *transport for F, M, U, T, E, W, R, D and the local structural rows.*
10. *C records displayed* Cont *replay from a completion-law request through the same functor, monad, unit, multiplication, window, readback, tolerance, transport, provenance, and local naming rows.*

11. *P is the* Pkg *provenance over* $\mathsf{CompletionExponentialLaw}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$ $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionExponentialLaw}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $F, M, U, T, E, W, R, D, H, C, P, N$. It has no coordinate for host exponentials, quotient equality of completed maps, selected representatives, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8562.2 (Completion exponential law NameCert obligations)**.** *For every accepted* CompletionExponentialL *carrier*

$$L = (F, M, U, T, E, W, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionExponentialLaw}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed packet L and its structural rows H, C, P, N;*
2. *completion-functor exposure through F, read as the functorial source already named by chapter 423;*
3. *unit and multiplication coherence through M, U, T, read through chapter 378, chapter 1527, and chapter 1829;*
4. *universal-property handoff through the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *source of chapter 397;*
5. *Real, RegSeqRat, and Dyadic nonescape through W, R, D, requiring every Real-facing consumer to pass through finite completion windows, regular rational readback, and dyadic tolerance rows before a terminal* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Every completion-law-facing consumer must read these rows before treating the law as a standard completion interface. It receives no host exponential object, no quotient equality of completed maps, no selected representative, no selected limit, no countable-choice schedule, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no Lean-checked bridge from this packet.*

Proof. Project the carrier in Definition 8562.1. The accepted packet exposes exactly $F, M, U, T, E, W, R, D, H, C, P,$

The completion rows F, M, U, T bind the law to the existing functor, monad, unit, and multiplication packets. The universal-property read is therefore a handoff through named completion rows, not an external mapping-space theorem.

The finite rows E, W, R, D keep the exponential comparison observable. Function windows, completion windows, regular rational readback, and dyadic tolerances are inspected before any Real-facing seal can be consumed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the refused host exponentials, quotient equalities, selected limits, choice schedules, and ambient completeness objects have no carrier coordinate, the five rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every public $\mathsf{CompletionExponentialLaw}^{\uparrow}$ read factors through displayed $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, finite completion-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows. A public read that exports host exponentials, quotient equality of completed maps, selected representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean bridge verification from this packet refutes the carrier boundary.

Independence witness: cauchycompletionfunctor, cauchycompletionmonad, cauchycompletionunit, cauchycompletionmultiplication, completionuniversalproperty: $\mathsf{CompletionExponentialLaw}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the completion functor, monad, unit, multiplication, universal-property handoff, finite exponential windows, regular rational readback, dyadic tolerance, transport, replay, provenance, and local naming while refusing host exponential and quotient endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionExponentialLaw}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionExponentialLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8562.1 and Definition 8562.2 seed the finite completion-law packet over functor, monad, unit, multiplication, universal-property handoff, finite window, rational readback, dyadic tolerance, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionExponentialLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host exponentials, quotient equality of completed maps, selected representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-functor exposure, unit and multiplication coherence, universal-property handoff, Real/RegSeqRat/Dyadic nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionExponentialLawUp.

Constructive story: A $\mathsf{CompletionExponentialLaw}^{\uparrow}$ packet is generated from CauchyCompletionFunctor, CauchyCompletionMonad, CauchyCompletionUnit, CauchyCompletionMultiplication, CompletionUniversalProperty, finite completion-window, RegSeqRat, DyadicRatCore, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 8563

A Concrete Naming Certificate for $\mathsf{TraceTheorem}^{\uparrow}$

Following Sobolev and Lions–Magenes trace theory, the $\mathsf{TraceTheorem}^{\uparrow}$ packet records the finite boundary-read surface by which a Sobolev-class object exposes boundary data to Real-facing and variational consumers. It sits beside the Sobolev embedding route of chapter 2217, the boundary-control estimate of chapter 2329, the compact Sobolev route of chapter 2218, and the scalar readback seal of chapter 38. The packet is a NameCert surface for boundary trace rows; it is not a host trace theorem, a selected representative, or a quotient boundary-function space.

8563.1 Carrier

Definition 8563.1 (Trace theorem carrier)**.** *A* $\mathsf{TraceTheorem}^{\uparrow}$ *carrier is a finite packet*

$$T = (D, S, E, P, K, B, R, H, C, N)$$

with the following displayed rows:

1. *D is the domain and boundary-admissibility row, read through the metric, compact, or bounded-domain surface used by the surrounding Sobolev certificates.*
2. *S is the* $\mathsf{SobolevEmbedding}^{\uparrow}$ *source row that exposes the Sobolev regularity, exponent ledger, weak-derivative data, and scalar* $\mathsf{Real}^{\uparrow}$ *readback before any boundary value is consumed.*
3. *E is the* $\mathsf{PoincareInequality}^{\uparrow}$ *control row, recording the norm or anchored-boundary estimate that prevents constants and detached representatives from escaping the packet.*
4. *P is the* $\mathsf{RellichKondrachov}^{\uparrow}$ *compact-window row when a compact trace window is present, and an explicit boundary marker when no compact-window read is used.*
5. *K is the finite boundary-test schedule: a list of carried boundary windows and scalar probes whose values are read through* $\mathsf{Real}^{\uparrow}$*.*
6. *B is the boundary trace row binding each displayed boundary probe to the Sobolev source row.*
7. *R, H, C, N record replay, componentwise transport, continuation, and local* $\mathsf{NameCert}_{\mathsf{TraceTheorem}^{\uparrow}}$ *evidence over the same finite rows.*

The classifier compares two trace packets only by inherited transport of these rows. It does not identify host functions, quotient boundary classes, selected traces, or all boundary representatives.

8563.2 Obligations

Theorem 8563.2 (TraceTheorem NameCert obligations)**.** *Every accepted* $\mathsf{TraceTheorem}^{\uparrow}$ *packet exposes exactly the carrier rows of Definition 8563.1: domain admission, Sobolev embedding source, Poincare control, optional Rellich–Kondrachov compact window, finite boundary-test schedule, boundary trace ledger, replay, transport, continuation, and local naming. A consumer may cite the packet only after the sibling rows* $\mathsf{SobolevEmbedding}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, $\mathsf{RellichKondrachov}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *have been displayed in the active branch.*

Proof. Project the carrier row by row. The source coordinate S supplies the Sobolev embedding and scalar readback data before the boundary ledger is visible, while E supplies the Poincare control row that keeps the estimate anchored to displayed finite data.

If the compact boundary route is present, the P coordinate reads it through the Rellich–Kondrachov row. If that route is absent, the boundary marker records the absence explicitly, so no compactness theorem can be inferred from the carrier alone.

Finally read K, B, R, H, C, N. These rows expose finite boundary probes, the boundary ledger, replay, transport, continuation, and local naming evidence. Since no row contains a host restriction operator, quotient boundary-function equality, selected representative, unrestricted extension theorem, or ambient completeness proof, the packet exports only the listed NameCert obligations. □

Theorem 8563.3 (TraceTheorem boundary route)**.** *Every* $\mathsf{TraceTheorem}^{\uparrow}$ *boundary consumer factors through the finite route*

$$\mathsf{SobolevEmbedding}^{\uparrow} \longmapsto \mathsf{PoincareInequality}^{\uparrow} \longmapsto \mathsf{RellichKondrachov}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

whenever the compact branch is active, and through the same Sobolev, Poincare, and Real rows with an explicit compact-window absence marker otherwise. The route supplies finite boundary probe values and their transport ledger; it does not supply a full host trace theorem, a global extension operator, a quotient boundary space, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the source and control rows S and E. They force every boundary read to pass through the Sobolev embedding and Poincare control surfaces before the boundary ledger can be consumed.

For the compact branch, read P through the Rellich–Kondrachov row and then seal boundary probe values through $\mathsf{Real}^{\uparrow}$. For the non-compact branch, the absence marker blocks any inference of compact trace data while leaving the displayed Sobolev, Poincare, and Real rows available.

Transport and replay are componentwise through R, H, C, N. Therefore a consumer receives only the finite boundary-route certificate, not a host trace theorem or any unlisted analytic object. □

Closure status for $\mathsf{NameCert}_{\mathsf{TraceTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TraceTheorem}^{\uparrow}$ packet is generated from a domain-boundary row, Sobolev embedding source, Poincare control row, optional Rellich–Kondrachov compact-window row, finite boundary-test schedule, boundary trace ledger, replay, componentwise transport, continuation, and local naming evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TraceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8563.1, Definition 8563.2, and Definition 8563.3 name the finite boundary-trace carrier, its NameCert obligation surface, and its sibling dependency route through SobolevEmbedding, PoincareInequality, RellichKondrachov, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TraceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host trace theorem, selected boundary representative, quotient boundary-function equality, unrestricted extension operator, Lean verification, or bridge checking.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.TraceTheoremUp and checked obligation plus boundary-route theorems over the displayed sibling rows.

Chapter 8564

A Concrete Naming Certificate for $\mathsf{BMOFiniteWindow}^{\uparrow}$

Following John and Nirenberg, the $\mathsf{BMOFiniteWindow}^{\uparrow}$ packet records the finite bounded-mean-oscillation surface by which a displayed function window exposes local average deviation, dyadic scale control, and Calderon–Zygmund handoff data before any quotient function space or measure completion is imported. It sits after the real readback surface of chapter 38, the dyadic scalar ledger of chapter 291, the maximal-function consumer of chapter 3581, the weight-window route of chapter 6590, and the singular-integral handoff of chapter 7928. The packet names only finite rows: inspected windows, displayed averages, oscillation bounds, dyadic budgets, maximal and weighted consumers, Calderon–Zygmund provenance, transport, replay, and local NameCert evidence.

Definition 8564.1 (BMO finite-window carrier)**.** *A* $\mathsf{BMOFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, A, O, D, M, Q, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row naming the displayed function or scalar sample family whose oscillation is inspected.*
2. *W is the finite window ledger of balls, intervals, or dyadic cubes on which averages are allowed to be read.*
3. *A records the* $\mathsf{Real}^{\uparrow}$ *average readback on each displayed window.*
4. *O records the mean-oscillation bound row comparing the local deviation from A with the displayed tolerance.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale and tolerance ledger for window inclusion, subdivision, and oscillation comparisons.*
6. *M is the* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *consumer row allowed to read the displayed average-deviation data.*
7. *Q is the* $\mathsf{MuckenhouptWeight}^{\uparrow}$ *compatibility row for weighted finite-window consumers.*
8. *Z is the* $\mathsf{CalderonZygmundOperator}^{\uparrow}$ *handoff row, recording which truncated singular-integral request may consume the oscillation packet.*
9. *H records componentwise* hsame *transport for source, window, average, oscillation, dyadic, consumer, handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from window selection to average readback, oscillation comparison, and consumer handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{BMOFiniteWindow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$, $\mathsf{MuckenhouptWeight}^{\uparrow}$, $\mathsf{CalderonZygmundOperator}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BMOFiniteWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for an equivalence class modulo constants, a completed BMO *normed space, a John–Nirenberg exponential-integrability theorem, selected representatives, ambient measure completion, quotient function equality, or a host singular-integral theorem.*

Theorem 8564.2 (BMO finite-window NameCert obligations)**.** *For every accepted* $\mathsf{BMOFiniteWindow}^{\uparrow}$ *carrier*

$$B = (S, W, A, O, D, M, Q, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BMOFiniteWindow}^{\uparrow}}$ *surface consists exactly of source admission through S, finite-window exposure through W,* $\mathsf{Real}^{\uparrow}$ *average readback through A, mean-oscillation comparison through O,* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale accounting through D, maximal and weighted consumer exposure through M, Q, Calderon–Zygmund handoff through Z, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8564.1. The tuple displays the source, inspected windows, averages, oscillation comparison, dyadic ledger, maximal consumer, weighted compatibility, Calderon–Zygmund handoff, transport, replay, provenance, and local name.

The scalar rows are read first. The average row A is sealed through $\mathsf{Real}^{\uparrow}$, while the window and tolerance comparisons in D are read through $\mathsf{DyadicRatCore}^{\uparrow}$; no host completion or quotient representative is introduced by those reads.

The consumer rows then expose M, Q, Z. These rows name the allowed Hardy–Littlewood maximal, Muckenhoupt-weight, and Calderon–Zygmund consumers, so the local certificate is exhausted by the displayed finite-window route before H, C, P, N transport, replay, package, and name it. □

Theorem 8564.3 (BMO finite-window mean-oscillation induction)**.** *Any induction over accepted* $\mathsf{BMOFiniteWindow}^{\uparrow}$ *mean-oscillation reads proceeds only by extending the finite window ledger W with a displayed window whose* $\mathsf{Real}^{\uparrow}$ *average row A, oscillation comparison row O, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row D are already present. The induction step may transport the ledger by H and replay it by C, but it cannot infer an ambient* BMO *quotient class, an unrestricted supremum over all windows, or a completed norm from the finite packet.*

Proof. The base case is the empty or singleton displayed-window ledger, where the only readable data are the rows named in Definition 8564.1. The local NameCert obligations of Definition 8564.2 require average, oscillation, and dyadic rows before a window contributes to the packet.

For the extension step, append one displayed window to W. Its average is read through A, its deviation comparison through O, and its scale and tolerance budget through D; the rows H and C only preserve this already displayed data under transport and replay.

Thus the induction remains finite-window induction. Since neither the carrier nor the obligation surface contains a coordinate for quotient representatives, unrestricted window suprema, or completed norms, no such object is exposed by the induction. □

Theorem 8564.4 (BMO finite-window Calderon-Zygmund handoff)**.** *Every* $\mathsf{CalderonZygmundOperator}^{\uparrow}$ *consumer of an accepted* $\mathsf{BMOFiniteWindow}^{\uparrow}$ *packet must read the finite route*

$$S, W, A, O, D, M, Q, Z, H, C, P, N$$

through the displayed average, oscillation, dyadic, maximal, and weighted rows before the handoff row Z *is available. In particular, the consumer receives a finite oscillation certificate for inspected windows, not a principal-value theorem, a quotient function-space theorem, a completed measure-space theorem, or an ambient weak-type estimate.*

Proof. Start from the obligation theorem. The handoff row Z appears only after S, W, A, O, D, M, Q have been displayed in the same packet.

The maximal row M records the $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ route for local average control, and the weighted row Q records the $\mathsf{MuckenhouptWeight}^{\uparrow}$ compatibility route for weighted finite-window consumers. The dyadic ledger D and real average row A keep the scalar content inside $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback.

Therefore a Calderon–Zygmund consumer can inspect only the finite oscillation certificate carried by Z together with the preceding rows. The packet exposes no principal value, quotient function equality, completed measure space, or global weak-type estimate. □

Falsifiable prediction: A $\mathsf{CalderonZygmundOperator}^{\uparrow}$ request that cites a $\mathsf{BMOFiniteWindow}^{\uparrow}$ handoff without displayed average, oscillation, dyadic scale, maximal-consumer, and weighted-compatibility rows refutes the local carrier.

Independence witness: real, dyadicratcore, hardy-littlewood-maximal, muckenhoupt-weight, calderon-zygmund-operator: none is bijective to $\mathsf{BMOFiniteWindow}^{\uparrow}$ because this packet jointly carries source, finite-window, average, oscillation, dyadic, maximal-consumer, weighted-compatibility, Calderon–Zygmund handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BMOFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BMOFiniteWindow}^{\uparrow}$ packet is generated from a source row, finite inspected windows, RealUp average readback, mean-oscillation comparison, DyadicRatCoreUp scale accounting, HardyLittlewoodMaximalUp and MuckenhouptWeightUp consumer rows, CalderonZygmundOperatorUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BMOFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8564.1, Definition 8564.2, Definition 8564.3, and Definition 8564.4 seed the finite bounded-mean-oscillation NameCert surface for inspected windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BMOFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim a completed BMO space, quotient equality modulo constants, John–Nirenberg exponential integrability, principal values, weak-type estimates, measure completion, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked source admission, finite-window exposure, Real average readback, oscillation comparison, dyadic scale accounting, maximal and weighted consumer exposure, Calderon–Zygmund handoff, transport, replay, provenance, and local naming for BEDC.Derived.BMOFiniteWindowUp.

Chapter 8565

A Concrete Naming Certificate for $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$

Following Bishop–Bridges diagonal arguments, the $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ packet records the finite route by which a displayed stream diagonal is read as completion-facing Cauchy data before a terminal Real seal is available. It sits after the stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name surface of chapter 38, and the completion operator of chapter 3379. The packet supplies a completion-facing diagonal route for compact-uniform Cauchy consumers; it is not a selected subsequence principle, not quotient stream equality, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8565.1 (Stream diagonalization completion carrier)**.** *A* $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, R, D, E, O, W, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the finite stream windows and diagonal schedule inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values exposed along the diagonal windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that compares diagonal windows at the requested finite depths.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the stream, regular-rational, and dyadic rows have been displayed.*
5. *O is the* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *row that receives the diagonal data as a completion-facing operator request.*
6. *W is the finite diagonal-window ledger, recording the source window, target window, and diagonal cut inspected for each completion read.*
7. *B is the compact-uniform Cauchy consumer boundary, recording the finite compact source and uniform-modulus read that asks for the diagonal completion handoff.*
8. *H records componentwise* hsame *transport for S, R, D, E, O, W, B and the local structural rows.*

9. *C records displayed* Cont *replay from a compact-uniform Cauchy request through the same stream, rational, dyadic, Real, completion-operator, window, boundary, transport, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamDiagonalizationCompletion}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $S, R, D, E, O, W, B, H, C, P, N$. It has no coordinate for selected subsequences, quotient stream equality, selected limits, countable choice, host compactness, host completion operators, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8565.2 (Stream diagonalization completion NameCert obligations)**.** *For every accepted* $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ *carrier*

$$G = (S, R, D, E, O, W, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamDiagonalizationCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; finite StreamName diagonal exposure through S; regular rational readback through R; dyadic tolerance exposure through D; terminal Real sealing through E; completion-operator handoff through O; compact-uniform Cauchy consumer boundary through B; finite diagonal-window replay through W; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Every compact-uniform Cauchy consumer must read those finite rows before using the diagonalization as completion data. It receives no selected subsequence, no quotient stream equality, no selected limit, no countable-choice schedule, no host compactness theorem, no host completion operator, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no Lean-checked bridge from this packet.*

Proof. Project the carrier in Definition 8565.1. Carrier admission exposes exactly $S, R, D, E, O, W, B, H, C, P, N$.

The analytic route starts with S, R, D. These rows display the stream windows, regular rational readbacks, and dyadic tolerance comparisons that make the diagonal finite.

The completion route then reads E, O, W, B. The Real seal is downstream of the finite stream data, the completion operator receives only that displayed diagonal route, and the compact-uniform consumer boundary records the request without adding host compactness.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the refused selected-subsequence, quotient, limit-selection, choice, host-completion, and ambient-completeness objects have no coordinate, the displayed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every public $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ read factors through displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionOperator}^{\uparrow}$, finite diagonal-window, compact-uniform Cauchy consumer, hsame, Cont, Pkg, and local NameCert rows. A public read that exports selected subsequences, quotient stream equality, selected limits, countable choice, host compactness, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean bridge verification from this packet refutes the carrier boundary.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchycompletionoperator: $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite stream windows, regular rational readback, dyadic tolerance, terminal Real sealing, completion-operator handoff, compact-uniform Cauchy boundary, transport, replay, provenance, and local naming while refusing selected-subsequence and quotient endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamDiagonalizationCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamDiagonalizationCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8565.1 and Definition 8565.2 seed the finite diagonal-completion packet over StreamName, RegSeqRat, DyadicRatCore, Real, CauchyCompletionOperator, compact-uniform consumer, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamDiagonalizationCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subsequences, quotient stream equality, selected limits, countable choice, host compactness, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, regular rational readback, dyadic tolerance exposure, Real sealing, completion-operator handoff, compact-uniform consumer boundary, transport, replay, provenance, and local naming for BEDC.Derived.StreamDiagonalizationCompletionUp.

Constructive story: A $\mathsf{StreamDiagonalizationCompletion}^{\uparrow}$ packet is generated from StreamName, RegSeqRat, DyadicRatCore, Real, CauchyCompletionOperator, finite diagonal-window, compact-uniform Cauchy consumer, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 8566

A Concrete Naming Certificate for $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ packet names the finite equivalence surface between Bishop regular Cauchy completion data and the sealed $\mathsf{Real}^{\uparrow}$ completion route. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the admission criterion of chapter 1476, and the terminal real seal of chapter 38. The packet records an equivalence of displayed NameCert routes, not a quotient real construction, a selected limit, countable choice, or completeness as a primitive.

Definition 8566.1 (Bishop Cauchy completion equivalence carrier)**.** *A* $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, R, K, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by both directions of the comparison.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which Cauchy observations are read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational row exposed by those windows.*
4. *K is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *admission row that certifies the displayed Cauchy route before any real seal is read.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same D, S, R, K surface.*
6. *H stores componentwise* hsame *transport for the dyadic ledger, stream schedule, regular rational row, criterion row, real seal, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from Bishop regular Cauchy data to the criterion row and from the criterion row back to the same finite real-seal read.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these nine rows. It contains no coordinate for quotient streams, selected host limits, countable choice, host real equality, or ambient completeness.

Theorem 8566.2 (Bishop Cauchy completion equivalence NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ *carrier* $E = (D, S, R, K, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance admission, finite-window readback, regular rational handoff, completion-criterion admission, real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8566.1. The rows $D, S, R, K, L, H, C, P, N$ are the only classifier-visible coordinates.

The Bishop regular Cauchy side is read through D, S, R. The dyadic ledger supplies the tolerance scale, the stream schedule supplies finite windows, and the regular rational row supplies the Cauchy handoff available to the criterion row.

The completion side is read through K, L. The criterion row admits the displayed Cauchy route, and the real seal row is exposed only after that admission has been consumed.

The remaining rows preserve and name the same route. Since no quotient stream, selected limit, countable-choice row, host equality, or completeness coordinate appears in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8566.3 (Bishop Cauchy completion equivalence route)**.** *Every consumer of an accepted* $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyCompletionCriterion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the stored hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route may be read from the Bishop regular Cauchy side toward the real seal or from the real seal back to the displayed criterion surface, but both reads use the same finite rows* $D, S, R, K, L, H, C, P, N$*.*

Proof. Begin with an accepted carrier $E = (D, S, R, K, L, H, C, P, N)$. The source side contains exactly the dyadic ledger D, stream schedule S, and regular rational row R.

Apply the NameCert obligations of Definition 8566.2. They require the criterion row K to be the admission point between the displayed regular Cauchy data and the real seal row L.

The replay row C records both readings of the same finite route: from D, S, R to K, L, and from a real-seal read back to the criterion surface that licensed it. The transport row H and provenance row P keep the route componentwise stable, while N names the local certificate.

Thus a consumer cannot substitute a host quotient, a selected limit, or a second completion principle for the displayed rows. Both directions of the equivalence are reads of the same finite NameCert route. □

Theorem 8566.4 (Bishop Cauchy completion equivalence non-escape)**.** *The carrier of Definition 8566.1, the NameCert obligations of Definition 8566.2, and the route theorem Definition 8566.3 assemble one finite equivalence packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The packet exports exactly the rows

$$D, S, R, K, L, H, C, P, N.$$

It exports no quotient stream, selected host limit, countable-choice schedule, host real equality, ambient completeness, or primitive completion operator.

Proof. The carrier definition lists the full observation surface. The NameCert obligations bind each row to its displayed role: dyadic tolerance, finite windows, regular rational handoff, criterion admission, real seal, transport, replay, provenance, and naming.

The route theorem shows that both readings of the equivalence factor through the same finite surface. A Bishop regular Cauchy read reaches the seal only through D, S, R, K, L, and a real-seal read returns only to the same criterion route.

No further row remains from which to read quotient streams, selected limits, countable choice, host equality, ambient completeness, or a primitive completion operator. The displayed rows therefore exhaust the $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, finite stream windows, a regular rational Cauchy row, a completion-criterion admission row, a Real seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8566.1, Definition 8566.2, Definition 8566.3, and Definition 8566.4 bind the finite equivalence carrier, obligation surface, route, and refusal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-real construction, selected limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a public consumer-factorization theorem for both readings of the dyadic, stream, regular rational, criterion, Real seal, and local NameCert route.

Chapter 8567

A Concrete Naming Certificate for $\mathsf{ProperUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{ProperUniformSpace}^{\uparrow}$ packet records the finite BEDC surface in which uniform entourages admit displayed controlled-neighbourhood windows whose completion-facing reads stay inside a properness boundary. It sits beside the uniform carrier of chapter 339, the proper-metric specializations of **??** and chapter 3686, the uniform and metric completion packets of chapter 1473 and chapter 1506, and the naming spine of chapter 126, chapter 301, and chapter 38. The carrier names properness as uniform finite data; it is not a selected compactification, a quotient-completion equality, a host-limit selector, or an ambient completeness principle.

Definition 8567.1 (Proper uniform space carrier)**.** *A* $\mathsf{ProperUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P_{\mathrm{pus}} = (U, M, W, C, R, S, Q, H, T, K, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, carrying finite entourage membership, inverse, refinement, and composition data.*
2. *M is the* $\mathsf{ProperMetric}^{\uparrow}$ *or* $\mathsf{ProperMetricSpace}^{\uparrow}$ *metric specialization row used when a consumer requests the metric properness face.*
3. *W is the controlled-uniform-neighbourhood window row, recording the bounded entourage windows that replace host compact neighbourhoods.*
4. *C is the completion-boundary row, exposing the* $\mathsf{UniformCompletion}^{\uparrow}$ *route before the* $\mathsf{MetricCompletion}^{\uparrow}$ *specialization can be read.*
5. *R is the regular rational readback row through* $\mathsf{RegSeqRat}^{\uparrow}$*, tied to the same finite windows.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that names every inspected uniform and completion-facing request.*
7. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after U, M, W, C, R, S have been displayed.*
8. *H records componentwise* hsame *transport for the uniform source, metric specialization, window, completion, readback, stream, seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a proper-uniform request through the same rows.*

10. *K is the* Pkg *provenance over* $\mathsf{ProperUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{ProperMetricSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProperUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for host compact neighbourhoods, selected completions, quotient uniform equality, countable choice, host real equality, or ambient completeness.

Theorem 8567.2 (Proper uniform space NameCert obligations)**.** *For every accepted* $\mathsf{ProperUniformSpace}^{\uparrow}$ *carrier* $P_{\mathrm{pus}} = (U, M, W, C, R, S, Q, H, T, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProperUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; uniform-source exactness for U; metric-specialization compatibility through M; controlled-window properness through W; completion-boundary routing through C; regular rational and stream readback through R, S; Real seal non-escape through Q; and structural transport, replay, provenance, and local naming through H, T, K, N.*

Proof. Project the carrier of Definition 8567.1. The uniform row, metric specialization, controlled window, completion boundary, regular readback, stream schedule, Real seal, and structural rows are exactly the visible coordinates.

Uniform-source exactness is rowwise. The row U supplies the entourage ledger, and every controlled neighbourhood request must pass through the displayed window row W before any completion-facing consumer can read it.

The metric face is only a specialization. The row M may expose $\mathsf{ProperMetric}^{\uparrow}$ or $\mathsf{ProperMetricSpace}^{\uparrow}$ data, but it does so downstream of the same uniform windows, so it cannot replace the uniform carrier by an unrelated metric completion.

Finally the rows C, R, S, Q, H, T, K, N preserve the displayed completion route. Since the carrier has no coordinate for selected completions, quotient equality, countable choice, host real equality, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 8567.3 (Proper uniform space completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{ProperUniformSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto W \longmapsto \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may use the $\mathsf{ProperMetric}^{\uparrow}$ *or* $\mathsf{ProperMetricSpace}^{\uparrow}$ *face as a specialization, but it cannot bypass the uniform window row or replace the completion boundary by a selected host limit, quotient completion, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a completion-facing consumer and project the accepted carrier. The only source row is the uniform entourage row U, and the only properness row available before completion is the controlled window row W together with the metric specialization row M.

Read the completion boundary next. The row C exposes the $\mathsf{UniformCompletion}^{\uparrow}$ route before any $\mathsf{MetricCompletion}^{\uparrow}$ specialization can be consumed, so a metric-completion read remains attached to the same uniform source and windows.

The naming spine is then forced by R, S, Q. Regular rational readback and stream scheduling are displayed before the terminal Real seal, while the transport, replay, provenance, and naming rows keep the consumer inside the same finite packet.

The excluded objects are absent from the carrier. Thus every completion-facing read follows the displayed route and cannot obtain a selected host limit, quotient completion, countable-choice schedule, or ambient completeness principle. □

Theorem 8567.4 (Proper uniform space L10 boundary)**.** *An accepted* $\mathsf{ProperUniformSpace}^{\uparrow}$ *packet supplies an L10 source object for properness arguments that do not start from the metric specialization. Any downstream route to* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *must first expose the uniform entourage row* U *and controlled window row* W*, then preserve the same packet through* H, T, K, N*.*

Proof. Project the source rows of Definition 8567.1. The packet begins with $\mathsf{UniformSpace}^{\uparrow}$ data and a controlled-window row, so the properness source is uniform before it is metric.

Use Definition 8567.3. It places $\mathsf{UniformCompletion}^{\uparrow}$ ahead of the metric-completion specialization and binds both completion reads to the same stream and regular-rational naming spine.

The structural rows H, T, K, N preserve that source order under accepted comparison. Therefore the L10 boundary is a proper-uniform source object, while the metric face remains a downstream specialization rather than the only properness entry point. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProperUniformSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProperUniformSpace}^{\uparrow}$ packet is generated from UniformSpace entourage data, a proper-metric specialization row, controlled uniform windows, UniformCompletion and MetricCompletion boundary rows, StreamName and RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProperUniformSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8567.1, Definition 8567.2, Definition 8567.3, and Definition 8567.4 seed the finite proper-uniform carrier, its NameCert obligations, the completion route, and the L10 boundary separating the uniform source from the metric specialization.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProperUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected compactifications, quotient completion equality, host-limit selection, countable choice, host real equality, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exactness, metric-specialization compatibility, controlled-window properness, completion-boundary routing, naming-spine readback, transport, replay, provenance, and local NameCert exactness for BEDC.Derived.ProperUniformSpaceUp.

Chapter 8568

A Concrete Naming Certificate for $\mathsf{KatetovTongInsertion}^{\uparrow}$

Following Katetov, Tong, and Dowker, the $\mathsf{KatetovTongInsertion}^{\uparrow}$ packet records the finite normal-space insertion route by which a lower semicontinuous row and an upper semicontinuous row are separated by a displayed continuous real-valued function. It sits over the normal separation surface of chapter 1866, the Urysohn separation route of **??**, the extension handoff of chapter 1956, the continuous-map carrier of chapter 130, and the terminal real-name seal of chapter 38. The packet names a finite insertion certificate only; it does not import host choice, quotient function equality, selected pointwise suprema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8568.1 (Katetov–Tong insertion carrier)**.** *A* $\mathsf{KatetovTongInsertion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (N, L, U, S, T, C, E, A, H, Q, P, M)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *source row whose separated closed-neighborhood data are inspected.*
2. *L is the lower-row ledger for the displayed lower semicontinuous boundary.*
3. *U is the upper-row ledger for the displayed upper semicontinuous boundary.*
4. *S records the strict pointwise separation budget between the L and U rows through terminal* $\mathsf{Real}^{\uparrow}$ *seals.*
5. *T is the local Urysohn/Tietze route that turns the separated neighborhood data into a continuous real-valued candidate.*
6. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row of the inserted function.*
7. *E records the endpoint inequalities saying that the inserted function lies between the lower and upper rows on every displayed probe.*
8. *A records finite approximation windows used to read the lower row, upper row, candidate map, and endpoint inequalities through the same* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *H records componentwise* hsame *transport for the normal source, semicontinuity rows, separation budget, Urysohn/Tietze route, continuous map, endpoint inequalities, approximation windows, replay, provenance, and local name.*
10. *Q records displayed* Cont *replay for the same insertion route.*

11. *P is the* Pkg *provenance over* $\mathsf{KatetovTongInsertion}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{TietzeExtension}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *finite insertion ledgers,* BHist, hsame, Cont, *and* NameCert.
12. *M is the local* $\mathsf{NameCert}_{\mathsf{KatetovTongInsertion}^{\uparrow}}$ *row.*

The carrier exports no host choice principle, quotient equality of functions, untracked real-valued map, selected supremum construction, or ambient completeness theorem.

Theorem 8568.2 (Katetov–Tong insertion NameCert obligations)**.** *For every accepted* $\mathsf{KatetovTongInsertion}^{\uparrow}$ *carrier*

$$I = (N, L, U, S, T, C, E, A, H, Q, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{KatetovTongInsertion}^{\uparrow}}$ *surface consists exactly of carrier admission, normal-source exposure, lower-row and upper-row visibility, strict real-valued separation, Urysohn/Tietze route discipline, continuous-map insertion, endpoint inequality exactness, approximation-window replay, componentwise transport, package provenance, and local naming. No insertion obligation is stored outside these rows.*

Proof. Project the carrier of Definition 8568.1. The rows $N, L, U, S, T, C, E, A, H, Q, P, M$ are the whole public surface, so carrier admission is exactly admission of those displayed coordinates.

The normal-source and separation obligations are read through N, L, U, S. The lower and upper rows are compared only through the displayed $\mathsf{Real}^{\uparrow}$ seals in S, and the route T records the Urysohn/Tietze separation data used to build the inserted function.

The inserted function itself is the $\mathsf{ContinuousMap}^{\uparrow}$ row C, with endpoint exactness recorded by E and finite readback windows recorded by A. Transport, replay, provenance, and local naming are then confined to H, Q, P, M, leaving no coordinate for host choice, quotient function equality, selected pointwise suprema, or ambient real completeness. □

Theorem 8568.3 (Katetov–Tong normal-space route)**.** *Every accepted* $\mathsf{KatetovTongInsertion}^{\uparrow}$ *packet factors any insertion consumer through the finite route*

$$\mathsf{NormalSpace}^{\uparrow} \longmapsto \mathsf{UrysohnLemma}^{\uparrow} \longmapsto \mathsf{TietzeExtension}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the consumer first exposes normal separation, then the local Urysohn/Tietze construction route, then the continuous inserted function and its real endpoint inequalities. It receives neither a host choice functional nor quotient equality of functions.

Proof. Start with the normal-space source row N and the separated lower and upper ledgers L, U, S. These rows provide the only closed-neighborhood and real-valued comparison data visible to the insertion request.

Read the route row T next. It records the local Urysohn and Tietze handoff through which the separated normal-space data can be turned into a continuous candidate without adding a host selector.

Finally read the continuous-map row C, the endpoint row E, and the approximation row A. The endpoint inequalities are checked through the same $\mathsf{Real}^{\uparrow}$ seal used by the lower and upper ledgers, while H, Q, P, M keep transport, replay, provenance, and naming attached to the same finite carrier. Hence every insertion consumer stays on the displayed normal-space route and cannot claim quotient function equality, a selected supremum, host choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{KatetovTongInsertion}^{\uparrow}$ read exposes the $\mathsf{NormalSpace}^{\uparrow}$ source row, lower and upper semicontinuity ledgers, strict $\mathsf{Real}^{\uparrow}$ separation budget, Urysohn/Tietze

route row, $\mathsf{ContinuousMap}^{\uparrow}$ inserted function, endpoint inequality ledger, finite approximation windows, hsame transport, Cont replay, Pkg provenance, and local naming before any insertion consumer may inspect the real-valued function.

Independence witness: normalspace, urysohn-lemma, tietze-extension, continuousmap, real: $\mathsf{KatetovTongInsertion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds lower and upper semicontinuity ledgers, strict real-valued separation budgets, Urysohn/Tietze construction rows, inserted continuous-map rows, endpoint inequalities, finite approximation windows, componentwise transport, replay, provenance, and local NameCert rows while refusing host choice and quotient function equality.

Closure status for $\mathsf{NameCert}_{\mathsf{KatetovTongInsertion}^{\uparrow}}$.

Constructive story: A $\mathsf{KatetovTongInsertion}^{\uparrow}$ packet is generated from a normal-space source row, lower and upper semicontinuity ledgers, a strict real-valued separation budget, a Urysohn/Tietze route row, a continuous-map inserted function, endpoint inequalities, finite approximation windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KatetovTongInsertion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8568.1, Definition 8568.2, and Definition 8568.3 seed the finite normal-space insertion certificate for L10 continuous real-valued consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KatetovTongInsertion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host choice, quotient equality of functions, selected pointwise suprema, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, normal-source exposure, lower and upper row visibility, strict real-valued separation, Urysohn/Tietze route discipline, continuous-map insertion, endpoint exactness, approximation replay, transport, provenance, and nonescape for BEDC.Derived.KatetovTongInsertionUp.

Chapter 8569

A Concrete Naming Certificate for $\mathsf{RegularCauchyRateLattice}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy moduli, this packet records finite meet, join, and refinement operations on displayed regular Cauchy rate ledgers. It sits beside the dyadic rational core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the rate-comparison packet of chapter 1818. The carrier names rate combination as finite ledger algebra; it is not a choice of a modulus functional, a selected limit, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8569.1 (Regular Cauchy rate lattice carrier)**.** *A* $\mathsf{RegularCauchyRateLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, A, M, J, F, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison row used by every rate entry.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the inspected Cauchy schedules.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the same windows.*
4. *A is the displayed rate-ledger row listing finitely many regular Cauchy rate entries.*
5. *M is the meet row, storing the finite common-refinement rate.*
6. *J is the join row, storing the finite coarser public rate bound.*
7. *F is the refinement row, recording when one displayed rate ledger refines another.*
8. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R, A, M, J, F have been displayed.*
9. *H records componentwise* hsame *transport for the rate ledger, operations, readback rows, seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay of the same finite rate-lattice route.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRateLattice}^{\uparrow}$, $\mathsf{CauchyRateLattice}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, A, M, J, F, Q, H, C, P, N$. It has no coordinate for a selected modulus functional, quotient stream equality, countable choice, trichotomy, a chosen limit, or ambient completeness.

Theorem 8569.2 (Regular Cauchy rate lattice obligation surface)**.** *For every accepted* $\mathsf{RegularCauchyRateLattice}^{\uparrow}$ *carrier* $L = (D, S, R, A, M, J, F, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateLattice}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rate ledger; dyadic tolerance admission through D; finite window admission through S; regular rational readback through R; meet, join, and refinement exposure through M, J, F; terminal real sealing through Q; and structural transport, replay, provenance, and local naming through H, C, P, N.*

Every Real or Completion consumer that combines regular Cauchy rates through this packet must read the finite ledger row A, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{StreamName}^{\uparrow}$ *window row S, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the meet, join, refinement, or* $\mathsf{Real}^{\uparrow}$ *seal rows are public.*

Proof. Project the carrier of Definition 8569.1. The rate-facing data are exactly the finite ledger A, the dyadic tolerance row D, the stream window S, and the regular rational readback R.

The lattice operations are finite row operations. The meet row M records a displayed common refinement, the join row J records a displayed coarser public bound, and the refinement row F records the comparison route between rate ledgers.

The consumer-facing read is delayed until the same finite route has been displayed. The seal row Q is downstream of D, S, R, A, M, J, F, and the structural rows H, C, P, N only transport, replay, source, and name that route.

No remaining coordinate can choose a modulus functional, quotient streams, a limit, a trichotomy branch, or a completeness principle. The listed rows therefore exhaust the obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRateLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRateLattice}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, a displayed regular Cauchy rate ledger, meet, join, and refinement rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyRateLattice}^{\uparrow}}$, seed)

Scope closed: Definition 8569.1 and Definition 8569.2 seed the finite regular Cauchy rate-lattice carrier and its NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyRateLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected modulus functionals, quotient stream equality, countable choice, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rate-ledger exactness, meet, join, refinement, dyadic tolerance, finite-window, regular-rational, Real-seal, transport, replay, provenance, and local NameCert rows for BEDC.Derived.RegularCauchyRateLatticeUp.

Chapter 8570

A Concrete Naming Certificate for HyperbolicBoundaryCrossRatioDistortion$^{\uparrow}$

The HyperbolicBoundaryCrossRatioDistortion$^{\uparrow}$ packet records boundary cross-ratio distortion as finite hyperbolic phase data. It sits beside the cross-ratio route of chapter 3948, the visual metric route of chapter 3866, the Busemann-function route of chapter 7534, and the boundary-shadow route of chapter 3813. The carrier is a NameCert surface for distortion rows; it is not a host hyperbolic boundary, a Mobius classification theorem, a quotient boundary equality, an unbounded real-valued invariant, or a global geometry oracle.

Definition 8570.1 (Hyperbolic boundary cross-ratio distortion carrier)**.** *A* HyperbolicBoundaryCrossRatioDistortion *carrier is a finite* BHist *packet*

$$D = (X, V, B, S, E, T, R, P, H, C, L, N)$$

with the following displayed rows:

1. *X is the* HyperbolicCrossRatio$^{\uparrow}$ *row for the inspected boundary quadruple.*
2. *V is the* HyperbolicVisualMetric$^{\uparrow}$ *row supplying the visual-metric comparison window.*
3. *B is the* HyperbolicBusemannFunction$^{\uparrow}$ *row used to read boundary displacement through finite Busemann data.*
4. *S is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row naming the shadow window where the distortion is inspected.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row for the displayed distortion read.*
6. *T is the transport row connecting the cross-ratio, visual-metric, Busemann, and boundary-shadow reads.*
7. *R is the replay row recording that the same boundary quadruple and shadow window are used throughout the packet.*
8. *P is the provenance row over the hyperbolic boundary, metric, Busemann, shadow, and Real-seal sources.*
9. *H records componentwise* hsame *transport for all analytic and structural rows.*
10. *C records displayed* Cont *replay of the local distortion route.*

11. *L records that later Lean TasteGate realizations must include* `Nontrivial` *and* `FieldFaithful` *instances before the distortion row is accepted as non-vacuous and source-faithful.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, V, B, S, E, T, R, P, H, C, L, N$. It has no coordinate for global boundary compactness, host Mobius equivalence, quotient boundary equality, hidden field choice, selected geodesics, or ambient hyperbolic geometry.

Theorem 8570.2 (Hyperbolic boundary cross-ratio distortion obligations)**.** *For every accepted* $\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}$ *carrier $D = (X, V, B, S, E, T, R, P, H, C, L, N)$, every distortion read exposes the cross-ratio row X, visual metric row V, Busemann row B, boundary-shadow row S, Real-seal row E, transport row T, replay row R, provenance row P, componentwise transport row H, displayed continuation row C, TasteGate instance row L, and local naming row N.*

The read is accepted only as this finite packet. It cannot be replaced by host boundary equality, a Mobius-normal-form theorem, unbounded real comparison, selected geodesics, quotient boundary compactification, or an ambient global-geometry principle.

Proof. Project the carrier of Definition 8570.1. The analytic rows are the cross-ratio, visual-metric, Busemann, boundary-shadow, and Real-seal rows X, V, B, S, E.

The structural rows keep those analytic reads attached to one boundary packet. Transport T and replay R preserve the same boundary quadruple and shadow window, while provenance P, componentwise transport H, displayed continuation C, and local naming N record the NameCert route.

The instance row L is part of the acceptance surface. It requires later Lean TasteGate material to carry `Nontrivial` and `FieldFaithful` instances before the distortion read can be accepted as non-vacuous and source-faithful.

No coordinate remains for a host boundary quotient, Mobius classification theorem, selected geodesic, hidden field choice, or global hyperbolic principle. The listed rows therefore exhaust the local obligations. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}$ distortion read exposes cross-ratio, visual metric, Busemann, boundary-shadow, Real-seal, transport, replay, provenance, and local naming rows before the read is public.

Independence witness: hyperbolic-cross-ratio, hyperbolic-visual-metric, hyperbolic-busemann-function, hyperbolic-boundary-shadow: cross-ratio records boundary quadruple comparison without a distortion ledger; visual metric records a metric window without the quadruple distortion row; Busemann functions record boundary displacement without the cross-ratio row; and boundary shadows record finite shadow windows without Real-sealed distortion. None is bijective to $\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}$ packet is generated from HyperbolicCrossRatio, HyperbolicVisualMetric, HyperbolicBusemannFunction, HyperbolicBoundaryShadow, RealUp seal, transport, replay, provenance, hsame, Cont, TasteGate instance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}}$, seed)

Scope closed: Definition 8570.1 and Definition 8570.2 seed the finite boundary cross-ratio distortion carrier and its NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryCrossRatioDistortion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host boundary equality, Mobius classification, selected geodesics, quotient boundary compactification, global hyperbolic geometry, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cross-ratio, visual-metric, Busemann, boundary-shadow, Real-seal, transport, replay, provenance, instance, and local NameCert exactness for BEDC.Derived.HyperbolicBoundaryCrossRatioDis

Chapter 8571

A Concrete Naming Certificate for $\mathsf{CauchyCompletionSeparator}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{CauchyCompletionSeparator}^{\uparrow}$ packet records the finite separator object that completion consumers read between regular Cauchy completion criteria and Hausdorff or separated completion boundaries. It sits after the Cauchy criterion surface of chapter 1476, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the separated universal surface of chapter 2242, and the quotient-free separation route of chapter 3839. The packet names a displayed separator boundary for L10 completion arguments; it is not quotient completion equality, a selected limit, host sequence equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8571.1 (Cauchy completion separator carrier)**.** *A* $\mathsf{CauchyCompletionSeparator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, W, R, D, U, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *row. It records the criterion that a completion-facing Cauchy name must satisfy before any separator is read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row. It exposes the finite observation window through which the criterion is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row. It records the regular rational readback used by the separator at the displayed precision.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row. It supplies the dyadic tolerance ledger used to compare the two inspected completion names.*
5. *U is the* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *row. It records the separated universal-property boundary consumed after the criterion, window, readback, and tolerance rows have been displayed.*
6. *Q is the* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *row. It records the quotient-free regular Cauchy separation handoff used by downstream completion consumers.*
7. *H records componentwise* hsame *transport for $K, W, R, D, U, Q, C, P, N$.*
8. *C records displayed* Cont *replay of the route*

$$K, W, R, D \longmapsto U, Q \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionSeparator}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparator}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, selected limits, host sequence equality, countable-choice schedules, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8571.2 (Cauchy completion separator NameCert obligations)**.** *For every accepted* CauchyCompletionSepa *carrier $S = (K, W, R, D, U, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparator}^{\uparrow}}$ *surface consists exactly of Cauchy criterion admission, finite StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance comparison, HausdorffCompletionUniversal separated-boundary routing, RegularCauchyCompletionSeparation handoff, componentwise transport, displayed continuation replay, package provenance, and local naming through the displayed rows $K, W, R, D, U, Q, H, C, P, N$.*

Proof. Project the carrier of Definition 8571.1. The Cauchy criterion row K is the only admission point, and the finite window W is the only route by which the criterion is inspected.

The regular rational row R and dyadic tolerance row D provide the displayed comparison data. A separator read therefore stays inside the finite regular Cauchy and dyadic ledgers before any separated completion boundary is consumed.

The rows U and Q then route the same inspected data to the Hausdorff universal boundary and the regular Cauchy separation handoff. Componentwise transport, continuation replay, provenance, and local naming through H, C, P, N preserve that route without adding quotient equality, selected limits, host equality, choice, trichotomy, or ambient completeness. □

Theorem 8571.3 (Cauchy completion separator Hausdorff route)**.** *Every completion consumer that reads an accepted* $\mathsf{CauchyCompletionSeparator}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{CauchyCompletionCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{HausdorffCompletionUniversal}^{\uparrow}, \mathsf{Regular}$$

Thus the separator can be consumed as a displayed Hausdorff or separated completion boundary, but not as quotient completion equality, selected-limit data, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted carrier $S = (K, W, R, D, U, Q, H, C, P, N)$. The source side is fixed by K, W, R, D: criterion admission, finite window exposure, regular rational readback, and dyadic tolerance comparison.

Read the separated boundary rows next. The $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ row U supplies the universal separated route, while the $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ row Q supplies the quotient-free separation handoff for regular Cauchy names.

Finally inspect the structural rows. The hsame transport row H, continuation row C, provenance row P, and local naming row N replay the same finite packet, so downstream completion consumers receive only the displayed separator route and no stronger host-side completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparator}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionSeparator}^{\uparrow}$ packet is generated from a CauchyCompletionCriterion row, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, HausdorffCompletionUniversal separated-boundary row, RegularCauchyCompletionSeparation handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionSeparator}\uparrow}$, seed)

Scope closed: Definition 8571.1, Definition 8571.2, and Definition 8571.3 seed the finite separator surface between regular Cauchy completion criteria and Hausdorff or separated completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionSeparator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, selected limits, host sequence equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-window exactness, regular-rational readback, dyadic tolerance comparison, Hausdorff universal routing, regular Cauchy separation handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionSeparatorUp.

Chapter 8572

A Concrete Naming Certificate for $\mathsf{OpialLemma}^{\uparrow}$

Following Opial's weak-convergence lemma, the $\mathsf{OpialLemma}^{\uparrow}$ packet records the finite certificate surface by which a Hilbert-facing weak-limit row and a strict norm-distance separation ledger can be inspected without importing a host weak compactness selector. It sits after the Opial-condition packet of chapter 6322, the Hilbert carrier of chapter 91, the norm and metric surfaces of chapter 89 and chapter 57, the scalar readback of chapter 38, and the weak compactness interface of chapter 6851. The packet names weak-limit and separation data only as displayed NameCert rows; it exports no selected weak cluster point, quotient equality, classical trichotomy, or ambient fixed-point theorem.

Definition 8572.1 (Opial lemma carrier)**.** *An* $\mathsf{OpialLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H, N, M, R, W, S, B, A, D, T, C, P, K)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, exposing only the inner-product, induced norm, and weak-observation surfaces visible to the packet.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row that reads norm distances for the displayed Hilbert endpoints.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row induced by the norm-distance ledger.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for all displayed inequalities and strict-separation margins.*
5. *W is the* $\mathsf{WeakCompactness}^{\uparrow}$ *handoff row, naming the weak-observation window while refusing any selected weak compactness witness.*
6. *S is the bounded sequence row whose entries are read through H, N, M, R.*
7. *B is the candidate weak-limit row, displayed as a named endpoint of the same Hilbert carrier and not as the result of a choice principle.*
8. *A is the alternative endpoint row used for comparison against the candidate weak limit.*
9. *D is the strict separation ledger*

$$\liminf_n \|S_n - B\| < \liminf_n \|S_n - A\|$$

read through the norm, metric, and Real rows.

10. *T records componentwise* hsame *transport for the Hilbert, norm, metric, Real, weak compactness, sequence, weak-limit, alternative endpoint, and strict-separation rows.*
11. *C records displayed* Cont *replay from an Opial-lemma request through*

$$H, N, M, R, W, S, B, A, D \longmapsto T, P, K.$$

12. *P is the* Pkg *provenance over* $\mathsf{OpialLemma}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *K is the local* $\mathsf{NameCert}_{\mathsf{OpialLemma}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host weak compactness selector, quotient equality of weak limits or Hilbert endpoints, classical trichotomy for arbitrary reals, a demiclosedness theorem, or a fixed-point theorem.

Theorem 8572.2 (Opial lemma NameCert obligations)**.** *For every accepted* $\mathsf{OpialLemma}^{\uparrow}$ *carrier*

$$L = (H, N, M, R, W, S, B, A, D, T, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{OpialLemma}^{\uparrow}}$ *surface consists exactly of Hilbert source admission, norm-distance admission, metric-distance readback, Real scalar readback, weak-observation handoff, bounded-sequence exposure, candidate weak-limit exposure, alternative-endpoint exposure, strict-separation exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 8572.1. The packet exposes precisely the rows $H, N, M, R, W, S, B, A, D, T, C, P, K$, so every obligation must be read from that finite inventory.

The analytic source rows are read first. The $\mathsf{Hilbert}^{\uparrow}$ row is interpreted through chapter 91; its norm and metric observations are routed through chapter 89 and chapter 57; and every comparison value is sealed by the $\mathsf{Real}^{\uparrow}$ readback of chapter 38.

The weak side is then confined to W, S, B, A. The $\mathsf{WeakCompactness}^{\uparrow}$ handoff of chapter 6851 supplies only a finite weak-observation window, while the sequence, candidate weak-limit row, and alternative endpoint remain displayed coordinates of the same packet.

Finally D, T, C, P, K bind the strict-separation and structural obligations. The strict ledger records the displayed norm-distance inequality, and the transport, replay, provenance, and local naming rows preserve only the stated coordinates. Hence no host weak compactness selector, quotient equality, classical trichotomy, or ambient fixed-point theorem is exported. □

Theorem 8572.3 (Opial lemma weak-limit separation)**.** *Every* $\mathsf{OpialLemma}^{\uparrow}$ *consumer that reads a candidate weak limit from an accepted carrier must also read the Hilbert source row H, norm-distance row N, metric row M, Real scalar row R, weak compactness handoff W, bounded sequence row S, alternative endpoint row A, and strict-separation ledger D. The exported conclusion is only the displayed weak-limit distance separation inside the packet; it is not a selected weak cluster point, quotient equality of endpoint representatives, classical real trichotomy, or host fixed-point principle.*

Proof. Start with the candidate weak-limit read. By Definition 8572.2, the candidate row B is visible only together with the source rows H, N, M, R, W, S, A, D and the structural rows T, C, P, K.

The distance comparison is routed through the named siblings. The Hilbert source provides the endpoint surface, the norm and metric chapters provide the distance observations, and the Real chapter supplies the scalar comparison row in which the strict inequality is read.

The weak compactness handoff stays local. The row W refers to the finite weak-observation interface of chapter 6851, so the consumer may inspect the displayed weak-limit candidate but may not extract a selected weak compactness witness.

It remains to identify the export boundary. Since the carrier has no coordinate for quotient equality, classical trichotomy, demiclosedness, or fixed-point data, the public read is exactly the strict weak-limit separation ledger preserved by T, C, P, K. □

Falsifiable prediction: Every accepted $\mathsf{OpialLemma}^{\uparrow}$ read must display $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{WeakCompactness}^{\uparrow}$, bounded-sequence, candidate weak-limit, alternative-endpoint, strict-separation, transport, replay, provenance, and local NameCert rows before any weak-limit separation is public. A read that exports a host weak compactness selector, quotient equality, classical trichotomy, demiclosedness, or a fixed-point theorem from this packet refutes the carrier boundary.

Independence witness: opial-condition, hilbert, norm, metric, real, weak-compactness: none is bijective with $\mathsf{OpialLemma}^{\uparrow}$ because this carrier jointly binds a Hilbert source, norm-distance ledger, metric readback, Real strict comparison, weak-observation handoff, bounded sequence, candidate weak-limit row, alternative endpoint row, strict-separation ledger, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{OpialLemma}^{\uparrow}}$.

Constructive story: An $\mathsf{OpialLemma}^{\uparrow}$ packet is generated from a Hilbert source, norm-distance and metric rows, Real scalar readback, weak compactness handoff, bounded sequence, candidate weak-limit row, alternative endpoint row, strict-separation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpialLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8572.1, Definition 8572.2, and Definition 8572.3 seed the finite Opial-lemma route from displayed weak-limit data to strict norm-distance separation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpialLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host weak compactness selector, quotient equality, classical trichotomy, demiclosedness, fixed-point existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-limit row exposure, norm-distance and Real strict-separation exactness, weak compactness handoff stability, and BEDC.Derived.OpialLemmaUp formalization.

Chapter 8573

A Concrete Naming Certificate for $\mathsf{RealTailFilter}^{\uparrow}$

Following Bishop and Bridges, a Real-facing tail filter is read only after the regular rational source, finite stream windows, dyadic tolerance ledger, terminal Real seal, and same-real tail route have all been displayed. The $\mathsf{RealTailFilter}^{\uparrow}$ packet sits after the stream-name interface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the regular real tail-equivalence route of chapter 5940. It records terminal tail-filter evidence for L10 Real and Completion consumers without quotient streams, selected limits, host equality, or ambient Real completeness.

Definition 8573.1 (Real tail filter carrier)**.** *A* $\mathsf{RealTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the tail request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the late-stage bound used by the tail request.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, D have been displayed.*
5. *F is the terminal filter row, recording the eventual tail request, its finite witness window, and its refusal boundary for missing tolerance evidence.*
6. *H records componentwise* hsame *transport for the window, readback, tolerance, seal, filter, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the same route from a tail request through S, R, D, E, F.*
8. *P records* Pkg *provenance over* $\mathsf{RealTailFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for quotient stream equality, a selected Cauchy representative, a chosen limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8573.2 (Real tail filter NameCert obligations)**.** *For every accepted* $\mathsf{RealTailFilter}^{\uparrow}$ *carrier*

$$T = (S, R, D, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealTailFilter}^{\uparrow}}$ *surface consists exactly of finite-window exposure through* S*, regular-rational readback through* R*, dyadic tolerance exposure through* D*, terminal Real sealing through* E*, eventual tail-filter evidence through* F*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8573.1. The analytic route starts with the finite window S, then reads the regular rational row R and the dyadic tolerance ledger D before any terminal Real seal is visible.

The filter row F is downstream of that route. It records only the eventual tail request and the finite witness window already controlled by S, R, D, E, so it cannot add a separate stream representative or a selected limit.

Finally read H, C, P, N. These rows transport, replay, source, and name the same finite packet. Since the classifier has no quotient, choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 8573.3 (Real tail filter eventual stability)**.** *Let*

$$T = (S, R, D, E, F, H, C, P, N)$$

be an accepted $\mathsf{RealTailFilter}^{\uparrow}$ *carrier. Any Real or Completion consumer that reads the terminal tail-filter row* F *must preserve the same* S, R, D, E *route at every later tail request whose finite witness window refines the displayed one. The consumer receives no quotient stream, selected representative, selected host limit, countable-choice schedule, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

Proof. Start from the NameCert obligations of Definition 8573.2. They force every tail-filter read to expose the finite window, regular-rational readback, dyadic tolerance ledger, and terminal Real seal before F is available.

For a later tail request, refinement is read through the same finite window discipline. The displayed tolerance ledger D is the only bound available to the consumer, so the request remains attached to the original S, R, D, E route.

The structural rows H, C, P, N preserve the route under transport and replay. Because the carrier lacks quotient, selector, host-limit, host-equality, choice, completeness, and bridge coordinates, eventual stability can export only the displayed tail-filter evidence. □

Falsifiable prediction: A $\mathsf{RealTailFilter}^{\uparrow}$ consumer that reaches terminal Real-facing tail-filter evidence after deleting S, R, D, or E refutes the carrier of Definition 8573.1.

Independence witness: Nearest siblings: StreamName, RegSeqRat, DyadicRatCore, Real, and RegularRealTailEquivalence. $\mathsf{RealTailFilter}^{\uparrow}$ is not bijective to any one sibling because it jointly carries finite windows, regular-rational readback, dyadic tolerance, terminal Real sealing, eventual tail-filter evidence, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealTailFilter}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance evidence, a terminal RealUp seal, eventual tail-filter evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8573.1, Definition 8573.2, and Definition 8573.3 seed the terminal Real-facing tail-filter route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligation rows, tail-filter nonescape, eventual stability, transport, replay, provenance, and local naming for BEDC.Derived.RealTailFilterUp.

Chapter 8574

A Concrete Naming Certificate for $\mathsf{LocatedRealSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of constructive metric spaces, the $\mathsf{LocatedRealSubset}^{\uparrow}$ packet records a Real-facing subset by finite positive and negative distance reads rather than by a classical powerset predicate. It sits over the located-real interface of **??**, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet gives downstream Real and Completion consumers a finite located-subset boundary without importing quotient subsets, selected representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8574.1 (Located real subset carrier)**.** *A* $\mathsf{LocatedRealSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, L, I, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare positive and negative distance reads.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule through which a consumer reads only finite prefixes of a real name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from those windows.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row that exposes the local located-real separator used by the subset packet.*
5. *I is the inside row, listing displayed evidence that the current real-name window lies within the subset up to the recorded tolerance.*
6. *O is the outside row, listing displayed evidence that the current real-name window is apart from the subset up to the recorded tolerance.*
7. *H stores componentwise* hsame *transport for tolerance, window, regular-rational, located-real, inside, outside, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay by which the same finite read moves from a real-name window to either the inside row or the outside row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealSubset}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedRealSubset}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through these rows. It has no coordinate for a host powerset of $\mathsf{Real}^{\uparrow}$*, quotient subset equality, selected representatives, countable choice, open-cover compactness, or ambient Real completeness.*

Theorem 8574.2 (Located real subset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealSubset}^{\uparrow}$ *carrier* $S = (D, W, R, L, I, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealSubset}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission, finite-window readback, regular-rational handoff, located-real comparison, inside-row exposure, outside-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8574.1. The rows $D, W, R, L, I, O, H, C, P, N$ are the only visible coordinates.

The Real-facing read starts with D, W, R, L. The dyadic ledger supplies the tolerance scale, the stream schedule supplies finite observation windows, the regular rational row supplies the readback, and the located-real row supplies the comparison boundary before any subset row is read.

The subset read is then split into I and O. Inside evidence and outside evidence are both displayed packet rows, so the packet records locatedness as a finite decision-facing boundary rather than as a host subset predicate.

The remaining rows preserve and name the same boundary. Since the carrier contains no host powerset, quotient subset equality, selected representative, choice schedule, compact-cover datum, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8574.3 (Located real subset L10 scope)**.** *Every consumer of an accepted* $\mathsf{LocatedRealSubset}^{\uparrow}$ *packet factors through the displayed L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedReal}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the inside row, outside row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The route exposes finite located-subset reads for Real and Completion consumers, and it does not export open-cover compactness, quotient subsets, countable choice, selected representatives, or ambient Real completeness.*

Proof. Begin with an accepted carrier $S = (D, W, R, L, I, O, H, C, P, N)$. The source side is the finite Real-facing chain D, W, R, L together with the real seal read named by the package row.

Apply Definition 8574.2. A consumer can read subset information only through the displayed inside row I or the displayed outside row O, and both rows remain under the same tolerance, window, regular-rational, and located-real boundary.

The transport row H, replay row C, provenance row P, and naming row N keep that boundary stable for downstream consumers. No remaining coordinate supplies compactness by open covers, quotient subset equality, countable choice, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealSubset}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealSubset}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, finite stream windows, regular rational readback, located-real comparison, inside and outside subset rows, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealSubset}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8574.1, Definition 8574.2, and Definition 8574.3 bind the finite located-subset carrier, NameCert obligation surface, and Real-facing L10 dependency scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealSubset}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, quotient subsets, countable choice, selected representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked consumer-factorization route for the displayed dyadic, stream, regular rational, located-real, Real seal, inside, outside, transport, replay, provenance, and local NameCert rows.

Chapter 8575

A Concrete Naming Certificate for $\mathsf{CauchyBinet}^{\uparrow}$

Cauchy–Binet's classical finite matrix-minor identity gives the $\mathsf{CauchyBinet}^{\uparrow}$ packet its linear-algebra surface: a displayed product matrix minor is expanded as a finite sum over intermediate column subsets, with every determinant, permutation sign, and scalar multiplication read through already named rows. It sits after the finite matrix carrier of chapter 49, the determinant packet of chapter 84, the permutation-sign route of chapter 116, and the scalar ring source of chapter 43. The packet names only finite minor data and its ledger; it does not assert a host linear-algebra theorem, quotient matrix equality, infinite-dimensional determinant, selected basis completion, or ambient algebraic closure.

Definition 8575.1 (Cauchy–Binet carrier)**.** *A* $\mathsf{CauchyBinet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, C, I, J, K, P_{\mathrm{row}}, P_{\mathrm{col}}, D_A, D_C, S, H, T, Q, N)$$

with the following displayed rows:

1. *R is the consumed* $\mathsf{Ring}^{\uparrow}$ *scalar row.*
2. *A and C are the two* $\mathsf{Mat}^{\uparrow}$ *factors over R, with compatible displayed inner dimension.*
3. *I and J are the displayed row and column index lists selecting the product minor.*
4. *K is the finite list of intermediate column subsets of the required cardinality.*
5. *P_{row} and P_{col} are* $\mathsf{Permutation}^{\uparrow}$ *rows used to order the selected rows, selected columns, and intermediate subsets before determinant signs are read.*
6. *D_A records the* $\mathsf{Determinant}^{\uparrow}$ *minors of A indexed by (I, K), and D_C records the* $\mathsf{Determinant}^{\uparrow}$ *minors of C indexed by (K, J).*
7. *S is the finite scalar sum row*
$$\sum_K \det(A_{I,K})\,\det(C_{K,J})$$
read through the ring operations and the determinant rows.
8. *H records componentwise* hsame *transport for the scalar, matrix, index-list, permutation, determinant, finite-sum, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a product-minor request through*

$$R, A, C, I, J, K, P_{\mathrm{row}}, P_{\mathrm{col}}, D_A, D_C, S \longmapsto H, Q, N.$$

10. *Q is the* Pkg *provenance over* $\mathsf{CauchyBinet}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBinet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host matrix equality, quotient row spaces, infinite determinant algorithms, selected bases, exterior algebra, or ambient field closure.

Theorem 8575.2 (Cauchy–Binet minor expansion)**.** *For every accepted* $\mathsf{CauchyBinet}^{\uparrow}$ *carrier*

$$B = (R, A, C, I, J, K, P_{\mathrm{row}}, P_{\mathrm{col}}, D_A, D_C, S, H, T, Q, N),$$

the product-minor read of the displayed matrix product factors through exactly the finite route

$$\mathsf{Mat}^{\uparrow}, \mathsf{Permutation}^{\uparrow}, \mathsf{Determinant}^{\uparrow}, \mathsf{Ring}^{\uparrow} \longmapsto D_A, D_C, S \longmapsto H, T, Q, N.$$

The exported identity is the displayed finite sum of paired minors indexed by K; it is not a host theorem about all matrices and not an equality outside the named carrier rows.

Proof. Project the carrier of Definition 8575.1. The matrix rows A and C, the selected index rows I, J, K, and the permutation rows $P_{\mathrm{row}}, P_{\mathrm{col}}$ are visible before any determinant is read.

The determinant rows are then finite. The row D_A reads each minor of A selected by (I, K), and D_C reads each minor of C selected by (K, J). Both rows are routed through chapter 84 and keep the needed order data attached to the permutation packet of chapter 116.

Finally the scalar sum row S is read through the $\mathsf{Ring}^{\uparrow}$ source. Since S contains only the displayed products $\det(A_{I,K})\det(C_{K,J})$ and their finite sum, the exported expansion remains inside the named matrix, determinant, permutation, ring, transport, replay, provenance, and local NameCert rows. □

Theorem 8575.3 (Cauchy–Binet ledger refusal)**.** *Every accepted* $\mathsf{CauchyBinet}^{\uparrow}$ *packet refuses host matrix equality, quotient row or column spaces, infinite-dimensional determinants, selected basis completion, exterior-algebra semantics, ambient field closure, and any determinant algorithm not displayed by the consumed* $\mathsf{Determinant}^{\uparrow}$ *rows. A consumer requesting one of those objects must leave the carrier route*

$$R, A, C, I, J, K, P_{\mathrm{row}}, P_{\mathrm{col}}, D_A, D_C, S, H, T, Q, N.$$

Proof. Inspect the carrier in Definition 8575.1. Its analytic rows are the scalar ring, the two displayed matrices, the finite index lists, the finite intermediate-subset list, the permutation order rows, the determinant-minor rows, and the finite scalar sum.

The minor expansion of Definition 8575.2 uses only those rows and the structural transport, replay, provenance, and local naming rows. No coordinate stores a quotient row space, an ambient equality of matrices, a selected basis completion, an infinite determinant, exterior-algebra semantics, or field-closure data.

Therefore such requests cannot be answered by the packet. They are refused at the ledger boundary before a linear-algebra consumer can treat the finite minor expansion as a host-level theorem. □

Falsifiable prediction: An accepted $\mathsf{CauchyBinet}^{\uparrow}$ read is refuted if it exports a product-minor identity without first displaying the MatrixUp factors, selected index lists, intermediate subset list, PermutationUp order rows, DeterminantUp minor rows, RingUp finite-sum row, transport, replay, provenance, and local NameCert rows.

Independence witness: matrix, determinant, permutation, ring: none is bijective with $\mathsf{CauchyBinet}^{\uparrow}$ because this carrier jointly binds two matrix factors, selected product-minor indices, intermediate subset data, permutation ordering, paired determinant minors, finite scalar summation, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBinet}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyBinet}^{\uparrow}$ packet is generated from a RingUp scalar source, two MatrixUp factor rows, selected row and column index lists, a finite intermediate-subset list, PermutationUp order rows, DeterminantUp minor rows, a finite scalar-sum row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBinet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8575.1, Definition 8575.2, and Definition 8575.3 seed the finite Cauchy–Binet route over displayed Matrix, Determinant, Permutation, Ring, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBinet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host matrix equality, quotient row spaces, infinite determinants, selected basis completion, exterior-algebra semantics, ambient field closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite subset enumeration, determinant-minor exactness, permutation-order stability, ring finite-sum exactness, ledger refusal, transport, provenance, and local naming for BEDC.Derived.CauchyBinetUp.

Chapter 8576

A Concrete Naming Certificate for $\mathsf{DarbouxFunction}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxFunction}^{\uparrow}$ packet records the intermediate-value surface of a real function on a displayed interval as finite BEDC rows. It sits beside the Real naming boundary of chapter 38, the graph-and-modulus route of chapter 130, the derivative consumer boundary of chapter 122, the tagged-sum interface of chapter 1960, and the Darboux integral route of chapter 1460. The packet names between-value witnesses for inspected interval windows without identifying host functions by quotient equality, choosing a global intermediate point, importing measure completion, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8576.1 (DarbouxFunction carrier)**.** *A* $\mathsf{DarbouxFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, R, G, B, C, D, H, K, P, N)$$

with the following displayed rows:

1. *I is the interval-source row whose endpoints and inspected subinterval windows are read through* $\mathsf{Real}^{\uparrow}$*.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *codomain row for the endpoint values and requested between-value level.*
3. *G is the displayed function-graph row, read by* $\mathsf{ContinuousMap}^{\uparrow}$ *and derivative-facing consumers only through finite source and value windows.*
4. *B is the between-value witness ledger: for each inspected pair of interval windows and requested value row it records the finite local witness window certified between the two endpoint values.*
5. *C is the continuity consumer boundary, tying G and B to the* $\mathsf{ContinuousMap}^{\uparrow}$ *route without asserting global uniform continuity.*
6. *D is the derivative consumer boundary, exposing the Darboux read needed by* $\mathsf{Derivative}^{\uparrow}$ *consumers while withholding any selected primitive or differentiability proof.*
7. *H stores componentwise* hsame *transport for interval, codomain, graph, between-value, continuity, derivative, provenance, and local naming rows.*
8. *K stores displayed* Cont *replay for the same finite reads.*
9. *P is the* Pkg *provenance over* $\mathsf{DarbouxFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient equality of host functions, a selected global intermediate point, measure completion, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.DarbouxFunctionUp.DarbouxFunctionTasteGate single carrier alignment`

Theorem 8576.2 (DarbouxFunction NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxFunction}^{\uparrow}$ *carrier* $F = (I, R, G, B, C, D, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}$ *surface consists exactly of interval-source admission, Real codomain readback, finite graph exposure, between-value witness-ledger exactness, continuity consumer boundary, derivative consumer boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation exports a host function quotient, a selected intermediate point outside the inspected witness ledger, measure completion, or ambient Real completeness.*

Proof. Project the carrier of Definition 8576.1. The rows $I, R, G, B, C, D, H, K, P, N$ are the only visible coordinates, so every Darboux-function read must pass through the interval source, Real codomain, displayed graph, and between-value ledger before it reaches a consumer.

The between-value claim is finite. The ledger B records inspected interval windows and requested value rows, and it names only the corresponding local witness window. Thus the packet supplies a Darboux surface for the inspected row, not a global choice of intermediate point.

The consumer boundaries then separate the downstream uses. The row C permits continuous-map consumers to replay the graph-and-ledger route, while D permits derivative consumers to read the same Darboux boundary without importing differentiability, a primitive, or an integral value.

Finally H, K, P, N transport, replay, source, and name this same packet. Since no carrier coordinate contains quotient equality of host functions, a global selected point, measure completion, or ambient Real completeness, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DarbouxFunction}^{\uparrow}$ packet is generated from interval-source rows, Real codomain rows, displayed graph rows, between-value witness ledgers, continuity and derivative consumer boundaries, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8576.1 and Definition 8576.2 seed the finite Darboux-function carrier and its local NameCert obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host function quotients, selected global intermediate points, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, between-value witness-ledger exactness, continuity and derivative consumer boundaries, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxFunctionUp.

Chapter 8577

A Concrete Naming Certificate for $\mathsf{LyndonWord}^{\uparrow}$

Following Lyndon and Schützenberger, the $\mathsf{LyndonWord}^{\uparrow}$ packet records a primitive finite word that is lexicographically smaller than every nontrivial proper suffix over a displayed totally ordered alphabet. It sits beside the concrete monoid carrier of chapter 40, the total-order comparison surface of chapter 54, the regular-language accepted-word ledger of chapter 351, and the context-free grammar derivation surface of chapter 356. The packet names only finite word, order, suffix-comparison, factorization, transport, replay, provenance, and local naming rows; it does not assert an infinite language quotient, a host free-monoid quotient, a selected global factorization oracle, or a bridge.

Definition 8577.1 (Lyndon word carrier)**.** *A* $\mathsf{LyndonWord}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, A, X, U, S, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite word row, displayed as a list of alphabet letters and read through the monoid concatenation surface.*
2. *A is the alphabet total-order row, exposing the finite comparison route for letters.*
3. *X is the lexicographic comparison row induced from A on finite word spines.*
4. *U is the primitive-word row, recording that W is not displayed as a nontrivial power of a shorter word in the monoid packet.*
5. *S is the proper-suffix minimality row, listing each nonempty proper suffix of W and the lexicographic comparison certifying W as strictly smaller.*
6. *F is the factorization ledger, recording the finite Chen–Fox–Lyndon factor route visible to regular-language and context-free grammar consumers.*
7. *H stores componentwise* hsame *transport for W, A, X, U, S, F and the structural rows.*
8. *C stores displayed* Cont *replay for finite word induction, suffix enumeration, lexicographic comparison replay, primitive-word inspection, and factorization handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{LyndonWord}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{TotalOrder}^{\uparrow}$*,* $\mathsf{RegularLanguage}^{\uparrow}$*,* $\mathsf{ContextFreeGrammar}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LyndonWord}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, X, U, S, F, H, C, P, N$. It has no coordinate for a host free monoid, quotient language equality, a selected factorization oracle, classical totality beyond the displayed total-order row, or a bridge field.

Theorem 8577.2 (Lyndon word NameCert obligations)**.** *For every accepted* $\mathsf{LyndonWord}^{\uparrow}$ *carrier $L = (W, A, X, U, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LyndonWord}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite word row, alphabet total-order row, lexicographic comparison row, primitive-word row, proper-suffix minimality row, factorization ledger, componentwise transport, displayed replay, provenance, and local naming. No Lyndon-word obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 8577.1. The word row W, total-order row A, lexicographic comparison row X, primitive row U, suffix-minimality row S, and factorization ledger F are the only mathematical coordinates.

Use finite word induction on W. The empty and one-letter cases expose no nontrivial suffix obligation, while the cons step reads the head-letter comparison from A and the tail comparison from the induction route through X.

Replay the suffix comparisons through S and C. Each nonempty proper suffix is a finite row, and its comparison with W is transported componentwise by H without adding a host word quotient.

Finally read P and N. Provenance and local naming mention only the displayed monoid, total-order, regular-language, context-free grammar, transport, and replay rows, so no selected oracle, quotient language equality, or bridge data can become a local NameCert obligation. □

Theorem 8577.3 (Lyndon word Chen–Fox–Lyndon factor route)**.** *Every regular-language or context-free grammar consumer of an accepted* $\mathsf{LyndonWord}^{\uparrow}$ *packet reads the Chen–Fox–Lyndon factor route through the finite ledger*

$$W, A, X, U, S, F, H, C, P, N.$$

The route first exposes the monoid word spine, then the alphabet total order and lexicographic suffix comparisons, then the primitive-word row, and only then the factorization ledger consumed by language-facing packets.

Proof. Begin with the finite word row W and the monoid surface of chapter 40. Word concatenation and nontrivial power inspection are read as displayed finite rows, so the primitive row U is not a host free-monoid assertion.

Next read the total-order and lexicographic rows through chapter 54. The proper suffixes listed in S are compared with W one by one, and finite word induction supplies the replay route for each suffix-comparison step.

The factorization ledger F is then available to chapter 351 and chapter 356. These consumers receive the displayed Lyndon factor route as finite word and derivation data, not as an external language quotient.

Finally transport and source the route through H, C, P, N. Since every row is displayed before the factor handoff, no selected global factorization oracle, quotient stream, host language equality, classical trichotomy beyond the displayed total order, or bridge field is exposed. □

Falsifiable prediction: Every accepted $\mathsf{LyndonWord}^{\uparrow}$ consumer that reads a factorization route must first display the finite word row, alphabet total-order row, lexicographic comparison row,

primitive-word row, proper-suffix minimality row, factorization ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: monoid, totalorder, regularlanguage, contextfreegrammar: $\mathsf{LyndonWord}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries primitive finite word data, proper-suffix lexicographic minimality, and a language-facing factorization ledger as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LyndonWord}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LyndonWord}^{\uparrow}$ packet is generated from a finite word row, alphabet total-order row, lexicographic comparison row, primitive-word row, proper-suffix minimality row, factorization ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LyndonWord}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8577.1, Definition 8577.2, and Definition 8577.3 seed the finite Lyndon-word certificate over monoid words, total-order comparisons, suffix minimality, and language-facing factorization data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LyndonWord}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host free-monoid quotient, infinite language equality, selected global factorization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite word induction, primitive-word exactness, suffix-comparison replay, factorization ledger handoff, transport, provenance, and local naming for BEDC.Derived.LyndonWordUp.

Chapter 8578

A Concrete Naming Certificate for $\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on choice sequences and limited omniscience boundaries, the $\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}$ packet records the finite test boundary at which Ishihara-style branch inspection may be consumed by Real and Completion rows. It sits beside $\mathsf{IshiharaTrick}^{\uparrow}$ in chapter 1665, $\mathsf{LLPOBoundary}^{\uparrow}$ in chapter 1978, $\mathsf{WLPOBoundary}^{\uparrow}$ in chapter 4062, the stream-window surface $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular rational readback $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, the dyadic tolerance ledger $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291, and the terminal real-name seal $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet exposes a finite branch-inspection route without importing LPO, WLPO, trichotomy, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8578.1 (Ishihara finite-test boundary carrier)**.** *An* $\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, P, N, W, Q, D, E, H, C, S)$$

with the following rows:

1. *B is the branch-window row displaying the finite observation schedule inspected by the Ishihara-style argument;*
2. *P is the positive-test row naming the finite evidence that the inspected branch has crossed the displayed positive threshold;*
3. *N is the negative-test row naming the finite evidence that the inspected branch remains inside the displayed negative threshold;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the inspected sequence coordinates are read;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the threshold budget used by the two finite tests;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, P, N, W, Q, D have been displayed;*
8. *H is componentwise* hsame *transport for the branch, test, window, readback, tolerance, and seal rows;*
9. *C is displayed* Cont *replay for the same finite inspection route;*
10. *S is the local* $\mathsf{NameCert}_{\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}}$ *row.*

The classifier accepts only packets whose public read factors through the displayed finite branch window and exactly one of the two test rows before the Real or Completion consumer reads E. It has no coordinate for LPO, WLPO, trichotomy of real names, a selected completed limit, quotient equality of streams, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8578.2 (Ishihara finite-test boundary NameCert obligations)**.** *For every accepted* IshiharaFiniteTestBoun *carrier $I = (B, P, N, W, Q, D, E, H, C, S)$, the local NameCert surface consists exactly of finite branch-window admission, positive-test admission, negative-test admission, StreamName window readback, RegSeqRat readback, DyadicRatCore tolerance budgeting, terminal Real-seal exposure, componentwise transport, displayed replay, and local naming. A Real or Completion consumer may use the packet only after the finite inspection rows B, W, Q, D and one displayed test row have been exposed; the consumer receives no LPO, WLPO, real trichotomy, selected limit, quotient stream equality, or ambient Real-completeness row.*

Lean4 (statement) `BEDC.Derived.IshiharaFiniteTestBoundaryUp`

Proof. Project the carrier of Definition 8578.1. The only branch-inspection data are the displayed window B, the positive-test row P, the negative-test row N, the stream window W, the regular rational readback Q, and the dyadic tolerance budget D.

The Ishihara-style use is therefore finite. A consumer first reads W, Q, D over the same branch window and then consumes either P or N as the displayed test outcome. The terminal Real seal E is downstream of that route.

Transport and replay through H and C preserve the same rows, and the local naming row S adds no omniscience coordinate. Since the carrier contains no LPO, WLPO, trichotomy, selected-limit, quotient-stream, or Real-completeness row, the listed rows exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}$ packet is generated from a finite branch-window row, positive and negative finite-test rows, a StreamName window, RegSeqRat readback, a DyadicRatCore tolerance budget, a terminal Real seal, componentwise transport, displayed replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8578.1 and Definition 8578.2 seed the finite branch-inspection boundary used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close LPO, WLPO, trichotomy, selected limits, quotient stream equality, ambient Real completeness, Lean-checked obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, test-row exactness, window/readback/tolerance compatibility, Real-seal nonescape, transport stability, replay stability, and local naming for BEDC.Derived.IshiharaFiniteTestBoundaryUp.

Chapter 8579

A Concrete Naming Certificate for $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ packet records the finite pairing and unpairing interface used when a regular Cauchy construction must fold a finite precision window into one unary request and then recover the two visible sides. It sits over the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the diagonal route of chapter 3782, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite NameCert for precision-window pairing; it is not a host pairing oracle, quotient equality of streams, selected real representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8579.1 (Regular Cauchy pairing-function carrier)**.** *A* $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, U, V, W, D, R, H, C, G, N)$$

with the following displayed rows:

1. *P is the unary pairing row. It records the finite code by which two precision requests are packed into one request index.*
2. *U and V are the first and second unpairing rows. They recover the two request windows from the displayed code P and do not read an ambient host pairing function.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected after U and V expose the two sides of the paired request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-transport ledger. It records the dyadic radius, split budget, slack comparison cells, and common-scale data transported across the two unpaired request windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the paired window and dyadic ledger.*
6. *H records componentwise* hsame *transport for the pairing code, unpairing rows, StreamName windows, dyadic tolerance transport, regular readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite precision request through $P, U, V, W, D, R, H, G, N$.*
8. *G is the* Pkg *provenance over* $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPairingFunction}^{\uparrow}}$ *row naming the finite pairing-function surface. The classifier compares accepted packets only through $P, U, V, W, D, R, H, C, G, N$. It has no coordinate for hidden host pairing functions, quotient stream equality, selected Real representatives, countable-choice schedules, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8579.2 (Regular Cauchy pairing-function NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ *carrier*

$$K = (P, U, V, W, D, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPairingFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the unary pairing row, the two unpairing rows, the finite StreamName window, the DyadicRatCore tolerance-transport ledger, the RegSeqRat readback, componentwise transport, displayed replay, package provenance, and local naming; pairing exactness for P; unpairing exactness for U, V; window exactness for W; tolerance transport for D; readback exactness for R; and ledger refusal for host pairing, quotient equality, selected representatives, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8579.1. The accepted tuple displays exactly $P, U, V, W, D, R, H, C, G, N$.

Pairing exactness is the row P. The two unpairing rows U and V consume the displayed code and recover the two finite precision windows before any StreamName or RegSeqRat readback is public.

Window and tolerance exactness are the rows W and D. The StreamName row reads only the unpaired finite windows, and the dyadic ledger transports the requested tolerance and common-scale data across those two windows.

The readback row R is downstream of the same route. The structural rows H, C, G, N transport, replay, source, and name the finite packet. Since the carrier contains no hidden host pairing function, quotient stream equality, selected representative, countable-choice schedule, trichotomy, host equality, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8579.3 (Regular Cauchy pairing-window exactness)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ *packet must read the finite route*

$$P \longmapsto U, V \longmapsto W \longmapsto D \longmapsto R \longmapsto H, C, G, N.$$

Thus the public readback receives only a displayed unary pairing code, its two displayed unpairing rows, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance transport, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Start with the NameCert obligations of Definition 8579.2. They force every accepted consumer to expose P before U, V, to expose U, V before W, and to expose W before the dyadic transport ledger D.

The regular readback is then row-local. The row R consumes exactly the displayed window and tolerance rows; it cannot replace them by a host pairing function or by an unbounded search over request pairs.

Finally, H, C, G, N preserve and name the same finite route. The downstream diagonal and Real consumers therefore inherit the paired precision window as displayed BEDC data, with no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ read displays one unary pairing row, two unpairing rows, one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-transport ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before a diagonal, completion, or Real-facing consumer may read the paired precision window.

Independence witness: streamname, regseqrat, dyadicratcore, regular-cauchy-diagonalization, real: $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a unary pairing row, two unpairing rows, a finite StreamName window, a DyadicRatCore tolerance-transport ledger, a RegSeqRat readback, and the structural transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPairingFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyPairingFunction}^{\uparrow}$ packet is generated from one unary pairing row, two unpairing rows, a finite StreamName window, a DyadicRatCore tolerance-transport ledger, a RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyPairingFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8579.1, Definition 8579.2, and Definition 8579.3 seed the finite precision-window pairing route used by RegSeqRat, diagonalization, completion, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyPairingFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pairing exactness, unpairing exactness, finite-window exactness, dyadic tolerance transport, readback exactness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyPairingFunctionUp.

Chapter 8580

A Concrete Naming Certificate for $\mathsf{DyadicRationalBallBasis}^{\uparrow}$

Following Bishop–Bridges constructive metric-space practice, the $\mathsf{DyadicRationalBallBasis}^{\uparrow}$ packet records a finite dyadic ball basis for real-line and metric-completion consumers before any open-ball primitive is available. It sits after the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric source interface of chapter 57. The finite ball-basis route also consumes the interval-cell refinement surface of chapter 5966 when a metric or Real consumer turns a dyadic center-radius row into a displayed interval cover. The packet gives Real, separability, and completion arguments a displayed basis element with center, radius, tolerance, readback, seal, transport, provenance, and local naming rows; it does not import host open balls, quotient real equality, located choice of centers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8580.1 (Dyadic rational ball basis carrier)**.** *A* $\mathsf{DyadicRationalBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, R, D, W, Q, E, H, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *center row, recording the dyadic rational center and its endpoint arithmetic.*
2. *R is the positive dyadic radius row, with the radius positivity and radius-comparison checks read through the same dyadic ledger as C.*
3. *D is the dyadic tolerance ledger for basis containment, refinement, and finite overlap reads.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a real or metric consumer inspects the center, radius, or tolerance schedule.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected dyadic data before any completed real boundary is opened.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the center and radius comparisons, available only after C, R, D, W, Q are displayed.*
7. *H records componentwise* hsame *transport for the center, radius, tolerance, window, readback, seal, provenance, and local naming rows.*

8. *P records* Pkg *provenance over* $\mathsf{DyadicRationalBallBasis}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRationalBallBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, R, D, W, Q, E, H, P, N$. It has no coordinate for host open balls, selected dense enumerations, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8580.2 (Dyadic rational ball basis NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRationalBallBasis}^{\uparrow}$ *carrier*

$$B = (C, R, D, W, Q, E, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRationalBallBasis}^{\uparrow}}$ *surface consists exactly of dyadic center admission through C, positive dyadic radius admission through R, tolerance-ledger admission through D, finite-window exposure through W, regular-rational readback through Q, real-seal exposure through E, componentwise transport through H, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8580.1. The analytic rows are the dyadic center C, positive radius R, tolerance ledger D, finite window W, regular readback Q, and Real seal E.

Read the rows in the displayed order. The center and radius rows are dyadic data, the tolerance row records containment and refinement bounds, the window row exposes the finite observations, and the readback row turns those observations into regular rational data before the seal is consumed.

Finally use the structural rows H, P, N. Transport, provenance, and local naming preserve that same finite route. Since no host open-ball, quotient-equality, dense-choice, trichotomy, choice, or ambient completeness coordinate appears in the carrier, none can be exported by the local certificate. □

Theorem 8580.3 (Dyadic rational ball basis Real coverage route)**.** *Every Real-facing or metric-facing consumer of an accepted* $\mathsf{DyadicRationalBallBasis}^{\uparrow}$ *carrier reads a basis element through the finite route*

$$C, R, D \longmapsto W, Q \longmapsto E.$$

The dyadic center, positive radius, and tolerance ledger expose the ball data before the $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal are available. Thus a local metric-ball argument receives a displayed dyadic basis row, not a host open-ball primitive, selected representative, quotient real equality, countable-choice dense set, or ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 8580.2. They force every accepted read to expose the center, radius, and tolerance rows before any Real-facing endpoint can be inspected.

Next read the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The finite window W supplies the inspected observations, while Q supplies regular rational readback for the same dyadic center and radius data.

Finally open the $\mathsf{Real}^{\uparrow}$ seal E. Because E is downstream of C, R, D, W, Q and the structural rows only transport, source, and name the same packet, the coverage route remains a finite dyadic ball-basis certificate. No excluded host open-ball, quotient, selected-representative, choice, or completeness endpoint is present. □

Falsifiable prediction: A $\mathsf{DyadicRationalBallBasis}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ or $\mathsf{Met}^{\uparrow}$ ball language without displaying C, R, D, W, Q refutes the carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRationalBallBasis}^{\uparrow}}$.

Constructive story: A DyadicRationalBallBasis$^{\uparrow}$ packet is generated from a dyadic center row, positive dyadic radius row, dyadic tolerance ledger, StreamName window row, RegSeqRat readback row, Real seal row, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicRationalBallBasis}\uparrow}$, seed)

Scope closed: Definition 8580.1, Definition 8580.2, and Definition 8580.3 seed the finite dyadic ball-basis route over center, radius, tolerance, finite-window, readback, Real-seal, transport, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicRationalBallBasis}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host open-ball primitives, quotient real equality, selected dense enumerations, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius admission, tolerance-ledger routing, StreamName window readback, RegSeqRat handoff, Real seal nonescape, transport, provenance, and local naming for BEDC.Derived.DyadicRationalBallBasisUp.

Chapter 8581

A Concrete Naming Certificate for $\mathsf{SylvesterEquation}^{\uparrow}$

Following Sylvester's matrix equation tradition, the $\mathsf{SylvesterEquation}^{\uparrow}$ packet records the finite equation route for a displayed linear matrix equation $AX - XB = C$. It sits after the finite matrix carrier of chapter 49, the linear-operator surface of chapter 48, the vector-space source of chapter 47, the scalar field route of chapter 45, and the characteristic-polynomial ledger of chapter 50. The packet names a candidate solution, a separation or solver ledger, and a residual row; it is not a host spectral theorem, a selected inverse of the Sylvester operator, quotient matrix equality, ambient Banach-space completeness, or an unbounded solver.

Definition 8581.1 (Sylvester equation carrier)**.** *A* $\mathsf{SylvesterEquation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, C, X, S, R, H, T, P, N)$$

with the following displayed rows:

1. *A and B are* $\mathsf{Mat}^{\uparrow}$ *or* $\mathsf{LinMap}^{\uparrow}$ *rows over a shared* $\mathsf{Field}^{\uparrow}$ *and compatible* $\mathsf{VecSpace}^{\uparrow}$ *source and target rows.*
2. *C is the right-hand-side row for the equation $AX - XB = C$.*
3. *X is candidate solution data with the same displayed source and target shape as the matrix or linear-operator route requires.*
4. *S is a separation or solver ledger, read through the* $\mathsf{Poly}^{\uparrow}$ *sibling when spectral or characteristic-polynomial separation data is displayed and through the* $\mathsf{LinMap}^{\uparrow}$ *sibling when an operator route is displayed.*
5. *R is the residual row for $AX - XB = C$, recording the displayed left action, right action, subtraction, and comparison with C.*
6. *H records componentwise* hsame *transport for A, B, C, X, S, R and the structural rows.*
7. *T records displayed* Cont *replay from the input operator rows and candidate solution through the residual check.*
8. *P is the* Pkg *provenance over* $\mathsf{SylvesterEquation}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SylvesterEquation}^{\uparrow}}$ *row naming the displayed equation packet.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host spectral decomposition, selected inverses, quotient matrix equality, analytic norm estimates, ambient completeness, or a solver oracle.

Theorem 8581.2 (Sylvester equation NameCert obligations)**.** *For every accepted* $\mathsf{SylvesterEquation}^{\uparrow}$ *carrier*

$$K = (A, B, C, X, S, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SylvesterEquation}^{\uparrow}}$ *surface consists exactly of matrix or linear-operator admission for* A *and* B*, right-hand-side admission for* C*, candidate-solution admission for* X*, separation or solver-ledger admission for* S*, residual-row admission for* R*, componentwise transport through* H*, displayed replay through* T*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8581.1. The equation data are precisely the two operator rows, the right-hand side, the candidate solution, the solver ledger, and the residual row.

The sibling anchors supply the allowed public surfaces. Matrix presentations read through chapter 49; operator presentations read through chapter 48; the common vector-space and scalar sources read through chapter 47 and chapter 45; and separation ledgers that use characteristic data read through chapter 50.

Finally H, T, P, N transport, replay, source, and name the same displayed route. Since the carrier contains no host spectral theorem, selected inverse, quotient equality, analytic completeness, or solver oracle, the listed rows exhaust the local NameCert obligations. □

Theorem 8581.3 (Sylvester equation linear-operator route)**.** *Every accepted* $\mathsf{SylvesterEquation}^{\uparrow}$ *packet exposes the residual computation only through the displayed operator route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{VecSpace}^{\uparrow}, \mathsf{Field}^{\uparrow} \longmapsto A, B, C, X \longmapsto R \longmapsto H, T, P, N.$$

When the separation or solver ledger S *uses characteristic data, that ledger is read through* $\mathsf{Poly}^{\uparrow}$ *before any consumer treats the residual row* R *as a certified solution check. No accepted consumer obtains a solution from a host inverse of* $X \mapsto AX - XB$*, quotient matrix equality, an unbounded algorithm, or ambient spectral completeness.*

Proof. Start from the carrier projection. The residual row R is downstream of the displayed rows A, B, C, X and records only the equation check $AX - XB = C$ inside the accepted packet.

The separation or solver ledger S is also a packet row. If a consumer uses spectral or characteristic-polynomial separation data, the packet forces that read through the $\mathsf{Poly}^{\uparrow}$ sibling before R can be consumed as a solution check.

The structural rows H, T, P, N then preserve and name this route. Because no host inverse, quotient equality, unbounded solver, analytic norm estimate, or ambient completeness coordinate is present, the linear-operator route is exactly the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{SylvesterEquation}^{\uparrow}$ read is refuted if it exports a solution check for $AX - XB = C$ without first displaying the MatrixUp or LinearMapUp input rows, VecSpaceUp and FieldUp sources, the right-hand-side row, candidate solution row, separation or solver ledger, residual row, transport, replay, provenance, and local NameCert rows.

Independence witness: matrix, linearmap, vecspace, field, polynomial: $\mathsf{SylvesterEquation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two operator rows, a right-hand-side row, candidate solution data, a separation or solver ledger, a residual row for $AX - XB = C$, and the local NameCert evidence in one equation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SylvesterEquation}^{\uparrow}}$.

Constructive story: A $\mathsf{SylvesterEquation}^{\uparrow}$ packet is generated from two MatrixUp or LinearMapUp rows over shared VecSpaceUp and FieldUp sources, a right-hand-side row, candidate solution data, a PolynomialUp-backed separation or solver ledger when needed, a residual row for $AX - XB = C$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SylvesterEquation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8581.1, Definition 8581.2, and Definition 8581.3 seed the finite Sylvester equation route over Matrix, LinearMap, VecSpace, Field, Polynomial, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SylvesterEquation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host spectral decomposition, selected Sylvester-operator inverses, quotient matrix equality, analytic norm estimates, ambient completeness, unbounded solver algorithms, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operator-shape compatibility, candidate solution admission, separation-ledger exactness, residual-row exactness, transport, replay, provenance, and local naming for BEDC.Derived.SylvesterEquationUp.

Chapter 8582

A Concrete Naming Certificate for $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$

Following Jordan's finite-partition treatment of bounded variation and Rudin's real-analysis presentation of variation sums, the $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$ packet records only the displayed partition ledger read by a bounded-variation consumer. It sits between the variation packet of chapter 398, the real-function route of chapter 3654, the finite dyadic partition carrier of chapter 1863, the finite mesh and adjacency sibling of chapter 3496, and the terminal real-name seal of chapter 38. The packet keeps the classical bounded-variation read at finite means: it names one interval, one finite partition, endpoint readbacks, edge-difference budgets, and local provenance, not a supremum over all partitions, a variation measure, or host equality of functions.

Definition 8582.1 (BoundedVariationFinitePartition carrier)**.** *A* $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, \Pi, E, R, D, V, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row on which the inspected partition is read.*
2. *Π is the finite* $\mathsf{DyadicPartition}^{\uparrow}$ *row, listing the ordered dyadic subintervals and their endpoint adjacency ledger.*
3. *E records the* $\mathsf{Real}^{\uparrow}$ *endpoint rows attached to the displayed partition points.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names used by the partition inspection.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-difference budget, recording the adjacent endpoint comparisons and the finite partial-sum ledger for the inspected partition.*
6. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger row consumed by a bounded-variation classifier after I, Π, E, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the interval, partition, endpoint, readback, budget, bounded-variation, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded-variation finite-partition request through I, Π, E, R, D, V.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedVariationFinitePartition}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through the displayed interval, finite partition, endpoint readback, edge-difference budget, bounded-variation ledger, transport, replay, provenance, and naming rows. It has no coordinate for total variation as a host supremum, a variation measure, quotient equality of functions, hidden partitions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8582.2 (BoundedVariationFinitePartition NameCert obligations)**.** *For every accepted* $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$ *carrier*

$$Q = (I, \Pi, E, R, D, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedVariationFinitePartition}^{\uparrow}}$ *surface consists exactly of interval admission, finite dyadic partition admission, endpoint admission through* $\mathsf{Real}^{\uparrow}$*, regular-rational endpoint readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic edge-difference budgeting through* $\mathsf{DyadicRatCore}^{\uparrow}$*, bounded-variation ledger handoff through* $\mathsf{BoundedVariation}^{\uparrow}$*, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8582.1. The interval row I fixes the domain of the inspected read, and the finite $\mathsf{DyadicPartition}^{\uparrow}$ row Π fixes the ordered adjacent cells whose endpoints may be queried.

Read the endpoint rows next. The $\mathsf{Real}^{\uparrow}$ row E gives the terminal endpoint seal, while the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies the regular-rational readback used to inspect those endpoint names at finite precision.

The dyadic budget row D then records the adjacent endpoint differences and their finite partial sums through $\mathsf{DyadicRatCore}^{\uparrow}$. Only after those rows are displayed can the bounded-variation ledger V be consumed as a $\mathsf{BoundedVariation}^{\uparrow}$ row.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no row for a host supremum over all partitions, a variation measure, quotient equality of functions, hidden partition families, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8582.3 (BoundedVariationFinitePartition ledger handoff)**.** *Every bounded-variation consumer of an accepted* $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$ *carrier must read the ordered finite route*

$$\mathsf{DyadicPartition}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{BoundedVariation}^{\uparrow}$$

through $I, \Pi, E, R, D, V, H, C, P, N$. The handoff supplies exactly one finite partition ledger to chapter 398 and to the real-function surface of chapter 3654.

Proof. Start with a bounded-variation request. By Definition 8582.2, the request first enters the displayed interval and finite partition rows, so every edge is one of the adjacent cells listed by Π.

The endpoint and budget rows then determine the inspected variation contribution. Endpoint values are read through E and R, and each adjacent difference is budgeted by the $\mathsf{DyadicRatCore}^{\uparrow}$ row D.

The $\mathsf{BoundedVariation}^{\uparrow}$ ledger V is downstream of this route. It receives the displayed finite partition and its edge-difference budget, while H, C, P, N preserve the same route for transport, replay, provenance, and local naming.

Thus the consumer obtains one finite partition ledger rather than a global variation object. The displayed partition row consumes the finite mesh and adjacency sibling of chapter 3496; no row supplies a supremum over all partitions, a variation measure, host function equality, a hidden refinement family, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationFinitePartition}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedVariationFinitePartition}^{\uparrow}$ packet is generated from an interval row, a finite DyadicPartitionUp row, RealUp endpoint rows, RegSeqRatUp readback rows, a DyadicRatCoreUp edge-difference budget, a BoundedVariationUp ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationFinitePartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8582.1, Definition 8582.2, and Definition 8582.3 seed the finite bounded-variation partition surface over interval rows, dyadic partitions, real endpoints, regular readbacks, dyadic edge budgets, bounded-variation handoff, and local provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationFinitePartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim total variation as a host supremum, a variation measure, quotient equality of functions, hidden partition families, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition exposure, endpoint readback, dyadic edge-budget exactness, bounded-variation ledger handoff, transport, replay, provenance, and local naming for BEDC.Derived.BoundedVariationFinitePartitionUp.

Chapter 8583

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$ packet records the separated uniqueness surface for regular Cauchy rational names before any quotient completion is formed. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names uniqueness by finite separation data, not by host real equality, quotient classes, selected limits, or ambient completeness.

Definition 8583.1 (RegularCauchyUniqueLimit carrier)**.** *A* $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, Q', W, W', D, S, E, H, C, P, N)$$

with the following rows:

1. *Q and Q' are* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows naming two regular rational Cauchy names inspected for the same prospective* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *W and W' are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which the two readbacks are observed.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing the dyadic radii, denominator windows, and comparison budget used by both readbacks.*
4. *S is the separated comparison row asserting that every displayed finite tolerance window forces the two readbacks into the same separation class before the seal row is opened.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, Q', W, W', D, S have been displayed.*
6. *H records componentwise* hsame *transport for the two readbacks, finite windows, tolerance ledger, separation row, seal row, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from finite window requests through dyadic tolerance comparison and separated uniqueness.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueLimit}^{\uparrow}}$ *row naming the separated uniqueness packet and its permitted real-seal reads.*

The classifier compares accepted packets only through $Q, Q', W, W', D, S, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice over tails, host equality of real numbers, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyUniqueLimitUp`

Theorem 8583.2 (RegularCauchyUniqueLimit separation)**.** *Let $U = (Q, Q', W, W', D, S, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing consumer that reads both regular Cauchy names through U must first pass through the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the separated comparison row S. Consequently the consumer receives a unique terminal seal only through the displayed finite separation route.*

The route exports no quotient stream equality, selected limit, countable-choice tail selector, host real equality, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 8583.1. The only rows available to a real-facing consumer are the two regular rational readbacks, the two finite StreamName windows, the shared DyadicRatCore tolerance ledger, the separated comparison row, and the terminal seal row, together with transport, replay, provenance, and local naming.

The separation row S is downstream of the synchronized finite windows and the common dyadic budget. Thus a consumer cannot inspect the terminal $\mathsf{Real}^{\uparrow}$ seal before the two regular Cauchy names have been compared through the same displayed tolerance ledger.

The structural rows H, C, P, N transport and replay exactly that finite route. Since the carrier has no row for quotient stream equality, selected limits, countable choice, host real equality, metric completeness, or ambient completeness, separated uniqueness is read only from the displayed NameCert surface. □

Theorem 8583.3 (RegularCauchyUniqueLimit NameCert obligations)**.** *For every accepted* RegularCauchyUniqueLi *carrier $U = (Q, Q', W, W', D, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueLimit}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, separated comparison row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, transport, replay, provenance, and local naming row;*
2. *window synchronization, requiring W and W' to expose the same finite inspection schedule before D is read;*
3. *tolerance-ledger exactness, requiring the denominator and radius budget in D to be the only budget consumed by the separated comparison row S;*
4. *separation-to-seal exactness, requiring S to be displayed before the terminal* $\mathsf{Real}^{\uparrow}$ *row E is available;*
5. *ledger refusal, excluding quotient stream equality, selected limits, countable choice over tails, host real equality, metric completeness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is tuple projection from Definition 8583.1. The accepted packet displays the two regular rational names Q, Q', synchronized finite windows W, W', shared dyadic tolerance ledger D, separated comparison row S, terminal seal E, and the structural rows H, C, P, N.

Window synchronization and tolerance-ledger exactness follow the displayed dependency order. The two finite windows are inspected before the shared dyadic ledger is consumed, and the separated comparison row can read only that ledger.

The seal exactness clause follows because E is downstream of S in the carrier. A consumer therefore cannot obtain a terminal real seal without first exposing the finite separated comparison route.

Finally, the carrier contains no row for quotient equality, selected limits, countable choice, host real equality, metric completeness, or ambient completeness. The listed rows therefore exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniqueLimit}^{\uparrow}$ packet is generated from two RegSeqRat readback rows, two synchronized StreamName finite-window rows, one DyadicRatCore tolerance ledger, a separated comparison row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8583.1, Definition 8583.2, and Definition 8583.3 seed the finite separated uniqueness route through RegSeqRat, StreamName, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueLimit}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyUniqueLimitUp`

Bridge status: none

Not claimed: This chapter does not claim full RealUp completeness, quotient completion, selected representatives, countable choice over tails, host real equality, metric completeness, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window synchronization, tolerance-ledger exactness, separation-to-seal exactness, transport stability, replay stability, and ledger refusal for BEDC.Derived.RegularCauchyUniqueLimitUp.

Chapter 8584

A Concrete Naming Certificate for $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$

Following Bishop–Bridges nested interval practice, the $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ packet records the finite closed-interval route by which a displayed nested interval argument is handed to Real and Completion consumers without choosing a completed point. It sits over the nested interval source of chapter 320, the finite diagonal selector surface of chapter 389, and the interval-cell refinement route of chapter 5966. The packet reads endpoint and width data through $\mathsf{DyadicRatCore}^{\uparrow}$, finite observations through $\mathsf{StreamName}^{\uparrow}$, regular rational tails through $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal completion-facing boundary through $\mathsf{Real}^{\uparrow}$; it does not assert Cantor-style completed intersections, point extraction, host real equality, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8584.1 (Finite closed-interval nested-intersection carrier)**.** *A* $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, C, D, W, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{NestedInterval}^{\uparrow}$ *packet, recording the retained closed-interval nesting rows and their finite containment ledger.*
2. *S is the* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *selector row, exposing the finite diagonal stage requested by the consumer.*
3. *C is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row, recording the interval cells used to read the selected finite stage.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and width row, carrying dyadic lower endpoints, upper endpoints, and width bounds for the selected cells.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row through which the consumer inspects the finite interval data.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-readback row, turning the displayed width bounds and endpoint windows into regular rational completion data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, S, C, D, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for the nested source, selector, cells, endpoints, widths, windows, tail readback, Real seal, provenance, and local naming rows.*

9. *T records the displayed continuity boundary*

$$\mathsf{Cont}(F) = \mathsf{Cont}(I, S, C, D, W, Q, E, H)$$

for the finite handoff from interval data to the Real-facing seal.

10. *P records*

$$\mathsf{Pkg}(F) = \mathsf{Pkg}(\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}, \mathsf{NestedInterval}^{\uparrow}, \mathsf{NestedIntervalIntersection}^{\uparrow}, \mathsf{DyadicIntervalI}$$

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, C, D, W, Q, E, H, T, P, N$. It has no coordinate for a completed infinite intersection, selected point, quotient real equality, choice schedule, host endpoint equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8584.2 (Finite closed-interval nested-intersection obligation surface)**.** *For every accepted* $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ *carrier*

$$F = (I, S, C, D, W, Q, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the source* $\mathsf{NestedInterval}^{\uparrow}$ *row I and the displayed selector row S;*
2. *classifier and window transport through the* $\mathsf{StreamName}^{\uparrow}$ *row W and componentwise* hsame *row H;*
3. *finite diagonal selector coverage through S and the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row C;*
4. *width-tail Cauchy handoff through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-width row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-readback row Q;*
5. $\mathsf{Real}^{\uparrow}$*-seal nonselection through E, which may be read only after I, S, C, D, W, Q are displayed;*
6. *provenance and local naming through* $\mathsf{Pkg}(F)$ *and* $\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}$.

Thus the packet supplies a finite closed-interval nested-intersection handoff, not Cantor-style completed intersection data, point extraction, host real equality, choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteClosedIntervalNestedIntersectionUp.FiniteClosedIntervalNestedIntersectionObl`

Proof. Project the carrier of Definition 8584.1. The source interval row I, selector row S, interval-cell row C, endpoint-width row D, window row W, readback row Q, and Real seal row E are displayed before the structural rows are read.

Read the finite route in dependency order. The nested source and selector admit the requested finite diagonal stage; the interval-cell and dyadic endpoint-width rows expose the inspected closed interval; the window and regular-rational readback rows turn that finite stage into completion-facing tail data.

Finally apply the transport, continuity, package, and local naming rows H, T, P, N. These rows preserve and name the same displayed data. Since the carrier contains no completed intersection coordinate, selected point, host equality, choice schedule, quotient equality, or ambient completeness row, none can be exported by the local certificate. □

Theorem 8584.3 (Closed-interval nested Bolzano handoff)**.** *Any Bolzano–Weierstrass or bounded-sequence consumer reading an accepted* $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ *packet must pass through the displayed seal rows*

$$I, S, C, D, W, Q, E, H, T, P, N.$$

In particular, cluster-point or subsequence language is available only after the consumer has read the $\mathsf{NestedInterval}^{\uparrow}$ *source row* I*, the* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *selector row* S*, the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row* C*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-width row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-readback row* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, followed by transport, continuity, provenance, and local naming. The handoff exports no selected subsequence, choice schedule, quotient equality, host cluster point, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteClosedIntervalNestedIntersectionUp.ClosedIntervalNestedBolzanoHandoff`

Proof. Project the carrier rows of Definition 8584.1. A bounded-sequence or Bolzano-facing read first sees the same nested source, selector, interval-cell, endpoint-width, window, readback, and Real-seal rows used by the local obligation surface.

Apply Definition 8584.2. It orders the finite route so that the selected diagonal stage, dyadic cell data, endpoint-width bounds, stream windows, and regular-rational tail readback are displayed before the Real-facing seal is consumed.

Only downstream readers may attach cluster or bounded-sequence terminology to that route. A $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, or Bolzano–Weierstrass-style consumer receives the displayed rows $I, S, C, D, W, Q, E, H, T, P, N$ as a handoff boundary, not a source of selected subsequences, countable choice, quotient equality, host cluster points, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8584.4 (Finite closed-interval nested-intersection obligation-closure package)**.** *For every accepted* $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ *carrier*

$$F = (I, S, C, D, W, Q, E, H, T, P, N),$$

the carrier, obligation surface, and Bolzano handoff jointly expose the $\mathsf{NestedInterval}^{\uparrow}$ *row* I*,* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *row* S*,* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row* C*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-width row* D*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-readback row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*,* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}$ *row* N*. The package supplies the finite L10 Real and Completion route only; it supplies no selected point, completed infinite intersection, host real equality, quotient real equality, choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8584.1. Its displayed rows are exactly the nested source, diagonal selector, interval-cell, endpoint-width, observation-window, tail-readback, Real-seal, transport, continuity, provenance, and local naming rows.

Apply the obligation surface of Definition 8584.2. It orders these rows as carrier admission, classifier and window transport, finite selector coverage, endpoint-width tail readback, Real-seal nonselection, provenance, and local naming.

Apply the Bolzano handoff of Definition 8584.3. It states that any bounded-sequence or Bolzano-facing consumer must pass through the same displayed rows before using cluster-point or subsequence language.

Because the carrier has no coordinate for point selection, completed intersection data, host equality, quotient equality, a choice schedule, or ambient completeness, the obligation package closes only the finite displayed route to L10 Real and Completion consumers. □

Lean4 ✓ `BEDC.Derived.FiniteClosedIntervalNestedIntersectionUp.FiniteClosedIntervalNestedIntersectionObl`

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ packet is generated from a NestedInterval source row, NestedIntervalIntersection selector row, DyadicIntervalNet cell row, DyadicRatCore endpoint-width row, StreamName window row, RegSeqRat tail-readback row, Real seal row, componentwise hsame transport, continuity boundary, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8584.1, Definition 8584.2, and Definition 8584.3 seed the finite closed-interval nested-intersection route over source, selector, cell, endpoint-width, window, readback, Real-seal, transport, continuity, provenance, local naming, and bounded-sequence handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Cantor-style completed intersections, point extraction, host real equality, quotient real equality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector coverage, interval-cell routing, endpoint-width tail readback, StreamName window transport, RegSeqRat handoff, Real seal nonselection, provenance, and local naming for BEDC.Derived.FiniteClosedIntervalNestedIntersectionUp.

Chapter 8585

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionPullback}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on separated uniform completions, the $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ packet records the finite BEDC surface where a Cauchy completion comparison becomes public only after Hausdorff separation has been displayed. It reads the stream, regular-rational, and dyadic source rows of chapter 126, chapter 301, and chapter 291; it refines the synchronized comparison of chapter 6954; it consumes the separated-uniform packets of chapter 2367 and chapter 2242; and it exports only the Real-facing route already named by chapter 38. The packet is a finite L10 NameCert object: it does not import quotient completions, selected representatives, countable choice, host equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8585.1 (Hausdorff completion pullback carrier)**.** *A* $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, W, D, P_0, P_1, H_0, H_1, U, T, C, Q, N)$$

with the following displayed rows:

1. *S is the common* $\mathsf{StreamName}^{\uparrow}$ *source row whose finite windows are read by both completion routes.*
2. *W is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for positive-radius comparison.*
4. *P_0 and P_1 are the two completion projection rows, each landing in the separated completion surface over S, W, D.*
5. *H_0 and H_1 are the two Hausdorff separation witnesses, read from* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *or* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *before any pullback comparison is exported.*
6. *U is the uniform pullback row binding the two projected completion routes over the common source and tolerance ledger.*
7. *T records componentwise* hsame *transport for the source, window, dyadic ledger, projections, separation witnesses, uniform pullback row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the separated pullback route.*

9. *Q is the* Pkg *provenance over* $\mathsf{HausdorffCompletionPullback}^{\uparrow}$, $\mathsf{CauchyCompletionPullback}^{\uparrow}$, $\mathsf{HausdorffUniformComp}$ $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionPullback}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, W, D, P_0, P_1, H_0, H_1, U, T, C, Q, N$. It has no coordinate for quotient-completion equality, selected limits, host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.HausdorffCompletionPullbackUp`

Theorem 8585.2 (Hausdorff completion pullback NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ *carrier*

$$P = (S, W, D, P_0, P_1, H_0, H_1, U, T, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffCompletionPullback}^{\uparrow}}$ *surface consists exactly of common source admission, regular-rational readback, dyadic tolerance control, two completion projections, two Hausdorff separation witnesses, uniform pullback comparison, transport, replay, provenance, and non-escape. In particular, a public separated-comparison read must display $S, W, D, P_0, P_1, H_0, H_1, U, T, C, Q, N$ before it may treat the two completion projections as meeting.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionPullbackUp.HausdorffCompletionPullbackTasteGate single carrier alignment`

Proof. Project the carrier of Definition 8585.1. The source rows S, W, D are the only finite input data available to the two completion projections, so every later comparison is already tied to the same stream window, regular sequence readback, and dyadic tolerance ledger.

Next inspect P_0 and P_1. These rows expose the two completion projections without giving a selected completed representative or a quotient-completion equality.

The Hausdorff rows H_0 and H_1 are then mandatory. They are the only coordinates that turn a Cauchy pullback comparison into a separated comparison, and they must be read before the uniform pullback row U can be exported.

Finally T, C, Q, N provide componentwise transport, displayed replay, provenance, and local naming for the same finite route. Since the carrier contains no coordinate for countable choice, host equality, detached limits, or ambient Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8585.3 (Hausdorff completion pullback uniform route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ *packet that compares the two exported projection rows must first read the common* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows S, W, D, then the two projection rows P_0, P_1, then the two Hausdorff separation witnesses H_0, H_1, and only then the uniform pullback row U. The route cannot inspect quotient filters, selected representatives, host function equality, countable choice, or ambient Real completeness.*

Proof. Begin with the synchronized finite-window comparison supplied by chapter 6954. It fixes the common source window and regular-rational readback through which the two completion routes are allowed to meet.

The dyadic ledger D supplies the positive-radius scale used by the separated uniform-completion packets. Reading P_0 and P_1 without D would expose two completion projections but no finite uniform comparison scale.

Now read H_0 and H_1 from the Hausdorff completion siblings. These witnesses are the public separation boundary; they block any replacement of the finite route by quotient filters, chosen representatives, or host-level equality.

With the source, readback, dyadic scale, projections, and separation witnesses displayed, the uniform pullback row U is a BEDC replay row rather than an external completeness principle. Transport, replay, provenance, and local naming keep the export inside the finite carrier. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ comparison must display StreamName, RegSeqRat, DyadicRatCore, two completion projection rows, two Hausdorff separation witnesses, a uniform pullback row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: cauchy-completion-pullback, hausdorff-uniform-completion, hausdorff-completion-universal, streamname, regseqrat, dyadicratcore, real: $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ is not bijective to any listed sibling because it binds the Cauchy pullback comparison to two explicit Hausdorff separation witnesses before the uniform pullback row becomes public.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ packet is generated from common StreamName, RegSeqRat, and DyadicRatCore source rows, two completion projection rows, two Hausdorff separation witnesses, a uniform pullback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8585.1, Definition 8585.2, and Definition 8585.3 seed the finite separated completion-pullback route over the L10 Real completion spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionPullback}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffCompletionPullbackUp.HausdorffCompletionPullbackTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection compatibility, Hausdorff separation witnesses, uniform pullback replay, transport, provenance, and non-escape for BEDC.Derived.HausdorffCompletionPullbackUp.

Chapter 8586

A Concrete Naming Certificate for $\mathsf{MetricCompletionAdmissibleEmbedding}^{\uparrow}$

Following Bishop and Bridges and Kelley on metric completion and dense isometric embeddings, the $\mathsf{MetricCompletionAdmissibleEmbedding}^{\uparrow}$ packet records the finite entry boundary by which a metric source is admitted into a displayed separated completion before extension consumers read it. It sits beside the metric-completion route of chapter 1506, the completion-embedding interface of chapter 1512, the Cauchy-filter basis of chapter 1501, the separated-metric comparison of chapter 1411, and the uniform-continuous extension handoff of chapter 769. Its finite readback rows pass through chapter 126, chapter 301, chapter 291, and chapter 38. The packet names an admissible dense embedding boundary; it is not a quotient completion, a selected representative, a host equality principle, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8586.1 (Metric completion admissible embedding carrier)**.** *A* $\mathsf{MetricCompletionAdmissibleEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, E, F, S, U, W, R, D, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, recording the completion-facing carrier, Cauchy admission surface, and completion readback boundary.*
2. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, recording the source-to-completion graph before any completed point equality is read.*
3. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row, exposing the finite filter-basis requests whose images are tested for density.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row, exposing the zero-distance comparison that makes the embedding admissible into a separated target.*
5. *U is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row, naming the downstream extension handoff available only after the dense embedding rows are displayed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the finite observations consumed by completion and extension readers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational distance and modulus data.*

8. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording the dyadic radii and comparison cells used by density and separatedness checks.*
9. *Z is the terminal* Real$^{\uparrow}$ *seal for the displayed distance and extension-readback values.*
10. *H records componentwise* hsame *transport for the completion, embedding, filter-basis, separated-metric, extension, window, rational readback, dyadic tolerance, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay for source requests, completion admission, dense-window lookup, separated comparison, and extension handoff.*
12. *P is the* Pkg *provenance over* MetricCompletionAdmissibleEmbedding$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CompletionEmbedding CauchyFilterBasis$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformContinuousExtension$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{MetricCompletionAdmissibleEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, F, S, U, W, R, D, Z, H, C, P, N$. It has no coordinate for quotient completion classes, selected limits, host equality of completed points, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.MetricCompletionAdmissibleEmbeddingUp`

Theorem 8586.2 (Metric completion admissible embedding NameCert obligations)**.** *For every accepted* MetricCompletionAdmissibleEmbedding$^{\uparrow}$ *carrier*

$$A = (M, E, F, S, U, W, R, D, Z, H, C, P, N),$$

the local NameCert$_{\text{MetricCompletionAdmissibleEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion row, completion-embedding graph, Cauchy-filter basis, separated-metric comparison, uniform-continuous extension handoff, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every extension-facing or completion-facing consumer must display $M, E, F, S, U, W, R, D, Z, H, C, P, N$ before reading the embedding as admissible.*

Proof. Project the accepted carrier. The analytic rows are $M, E, F, S, U, W, R, D, Z$, and the structural rows are H, C, P, N.

The embedding is read through the completion and filter-basis rows first. The rows M, E, F expose a completion-facing object, its graph from the source, and the finite basis requests that witness density for the read under inspection.

The separated and extension rows are downstream. The row S supplies the zero-distance comparison for the separated target, and U can be consumed only as a named handoff after the embedding and density rows have been displayed.

The numerical readback rows W, R, D, Z keep every distance, tolerance, and extension value inside the finite StreamName, RegSeqRat, DyadicRatCore, and Real seal route. Transport, replay, provenance, and local naming preserve the same displayed packet, so no quotient completion, selected limit, host equality, countable-choice row, trichotomy branch, or ambient Real completeness coordinate can be projected from the NameCert surface. □

Theorem 8586.3 (Metric completion admissible embedding handoff)**.** *Every* MetricCompletionAdmissibleEmbeddin *read handed to a* UniformContinuousExtension$^{\uparrow}$ *or completion-universal consumer must first expose the* MetricCompletion$^{\uparrow}$ *row,* CompletionEmbedding$^{\uparrow}$ *graph,* CauchyFilterBasis$^{\uparrow}$ *density requests,*

SeparatedMetric$^{\uparrow}$ *comparison, StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, and local NameCert rows. The handoff supplies finite admissible-embedding data only; it does not assert quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8586.2. They require the completion, embedding, filter-basis, separatedness, extension, window, readback, tolerance, seal, transport, replay, provenance, and naming rows to remain visible before the packet is read as admissible.

For a uniform-continuous extension consumer, the row U is not an independent extension theorem. It is a handoff row tied to M, E, F, S, W, R, D, Z, so the consumer receives only the displayed dense embedding boundary.

For a completion-universal consumer, the same rows give the completion and density route without quotienting or selecting a completed representative. The structural rows H, C, P, N preserve that finite route, and the excluded global principles are absent from the carrier. □

Falsifiable prediction: Every MetricCompletionAdmissibleEmbedding$^{\uparrow}$ handoff to completion or extension consumers displays MetricCompletion$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformContinuousExtension$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the admissible embedding is consumed.

Independence witness: metriccompletion, completionembedding, cauchyfilterbasis, separated-metric, uniform-continuous-extension, streamname, regseqrat, dyadicratcore, real: MetricCompletionAdmissibleEmb is not bijective to any listed sibling because it jointly carries a completion row, embedding graph, Cauchy-filter basis, separated comparison, extension handoff, finite windows, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{MetricCompletionAdmissibleEmbedding}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{MetricCompletionAdmissibleEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 8586.1, Definition 8586.2, and Definition 8586.3 seed the finite admissible-embedding route from metric completion and completion embedding rows through Cauchy-filter bases, separated comparison, extension handoff, finite windows, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\text{MetricCompletionAdmissibleEmbedding}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.MetricCompletionAdmissibleEmbeddingUp`

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host equality of completed points, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, embedding-graph exactness, Cauchy-filter density routing, separated-comparison exposure, extension-handoff nonescape, finite-window readback, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionAdmissibleEmbeddingUp.

Constructive story: A MetricCompletionAdmissibleEmbedding$^{\uparrow}$ packet is generated from a MetricCompletion row, CompletionEmbedding graph, CauchyFilterBasis requests, SeparatedMetric comparison, UniformContinuousExtension handoff, StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8587

A Concrete Naming Certificate for $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$ packet isolates the finite dyadic block-witness row used by the condensation route before theorem-level summability or completeness claims are read. It sits between the source-series carrier of chapter 1581, the finite block packet of chapter 7794, the theorem packet of chapter 3727, the dyadic arithmetic core of chapter 291, the stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The witness packet records only the displayed finite block data for one dyadic window; it is not host summability, quotient equality of series, selected infinite sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8587.1 (Cauchy condensation dyadic block witness carrier)**.** *A* $\mathsf{CauchyCondensationDyadicBlockWitness}$ *carrier is a finite* BHist *packet*

$$B = (A, D, W, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive decreasing* $\mathsf{RealSeries}^{\uparrow}$ *source row whose inspected tail supplies the block terms.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row naming the dyadic block bounds and precision scale used by the finite witness.*
3. *W is the finite block witness ledger, listing the displayed terms between the two dyadic endpoints and the associated condensed row.*
4. *M is the monotone endpoint comparison row, recording the decreasing-source inequalities needed to compare the finite block with its dyadic endpoint terms.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational data extracted from the displayed block window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, D, W, M, R have stayed visible.*
7. *H records componentwise* hsame *transport for the source, dyadic endpoint, witness ledger, monotone comparison, readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a condensation consumer request through the source tail, dyadic endpoints, finite witness ledger, monotone comparison, rational readback, and real-facing seal.*

9. *P is the* Pkg *provenance over* CauchyCondensationDyadicBlockWitness$^{\uparrow}$, CauchyCondensationPacket$^{\uparrow}$, CauchyCondensationTheorem$^{\uparrow}$, RealSeries$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{CauchyCondensationDyadicBlockWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, D, W, M, R, E, H, C, P, N$. It has no coordinate for an infinite series sum, host convergence theorem, quotient stream equality, selected limiting value, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCondensationDyadicBlockWitnessUp.CauchyCondensationDyadicBlockWitnessTasteGa`
`single carrier alignment`

Theorem 8587.2 (Cauchy condensation dyadic block witness NameCert obligations)**.** *For every accepted* CauchyCondensationDyadicBlockWitness$^{\uparrow}$ *carrier $B = (A, D, W, M, R, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyCondensationDyadicBlockWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the positive decreasing* RealSeries$^{\uparrow}$ *source row, dyadic endpoint exposure through* DyadicRatCore$^{\uparrow}$*, finite block witness ledger exposure, monotone endpoint comparison,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *sealing, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer therefore reads the block witness only through the finite route*

$$A, D, W, M, R \longmapsto E, H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8587.1. The source row A names the positive decreasing series, and the dyadic endpoint row D fixes the finite block window inspected by the witness ledger W.

The monotone comparison row M is then read before any condensed term or endpoint estimate is consumed. It keeps the comparison local to the displayed block and prevents a consumer from replacing the witness ledger by an ambient summability statement.

The rational readback row R and real seal row E are reached only after A, D, W, M remain visible. Finally H, C, P, N transport, replay, source, and name that same finite route, so no quotient stream, selected limit, host convergence theorem, countable-choice selector, or completeness witness enters the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyCondensationDyadicBlockWitnessUp.CauchyCondensationDyadicBlockWitnessNameCer`

Theorem 8587.3 (Cauchy condensation dyadic block witness Real handoff)**.** *Every* Real$^{\uparrow}$*-facing or completion-facing consumer of an accepted* CauchyCondensationDyadicBlockWitness$^{\uparrow}$ *carrier $B = (A, D, W, M, R, E, H, C, P, N)$ must first display the positive decreasing source row A, dyadic endpoint row D, finite block witness ledger W, monotone endpoint comparison row M, and* RegSeqRat$^{\uparrow}$ *readback row R. The handoff to* CauchyCondensationPacket$^{\uparrow}$ *and* CauchyCondensationTheorem$^{\uparrow}$ *may consume the seal row E only as this displayed finite block route. It does not assert host summability, equality of infinite series, selected sums, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8587.2. They require A, D, W, M, R to be displayed before the real-facing seal E is read.

The condensation packet and theorem siblings consume only that finite witness row. The packet of chapter 7794 records the surrounding condensation carrier, and chapter 3727 reads theorem-level block data only after the dyadic witness has kept its source, endpoint, ledger, comparison, and readback rows visible.

Finally the structural rows H, C, P, N preserve the same finite route by transport, replay, provenance, and local naming. Since the accepted carrier contains no coordinate for host summability, quotient equality of infinite series, selected sums, countable choice, or ambient completeness, the Real handoff remains a finite block-witness handoff. □

Lean4 ✓ `BEDC.Derived.CauchyCondensationDyadicBlockWitnessUp.CauchyCondensationDyadicBlockWitnessRealHan`

Theorem 8587.4 (Cauchy condensation dyadic block witness scoped route)**.** *Every accepted* $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$ *packet reads the CauchyCondensationTest,* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before any Real-facing handoff is public. Equivalently, the handoff route is scoped by the displayed source-series row A, dyadic endpoint row D, finite witness ledger W, monotone comparison row M, regular rational readback row R, Real seal row E, componentwise transport row H, continuation row C, provenance row P, and local naming row N.*

Proof. Begin with the carrier of Definition 8587.1. It lists $A, D, W, M, R, E, H, C, P, N$ as the whole packet surface and gives the Cauchy-condensation consumer no independent row beyond those displayed coordinates.

Use the NameCert obligations of Definition 8587.2. They force the source-series, dyadic endpoint, witness ledger, monotone comparison, and regular readback rows to remain visible before the Real seal is consumed.

Finally apply the Real handoff theorem Definition 8587.3. It routes every Real-facing or completion-facing read through the same finite block-witness rows, while H, C, P, N provide transport, replay, provenance, and local naming rather than a second summability surface. □

Theorem 8587.5 (Cauchy condensation dyadic block witness public certificate)**.** *The public* $\mathsf{NameCert}_{\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}}$ *surface is the package consisting of Definition 8587.1, Definition 8587.2, Definition 8587.3, and Definition 8587.4. Any public read of the dyadic block witness therefore factors through the displayed carrier rows $A, D, W, M, R, E, H, C, P, N$ and through no host summability theorem, equality of infinite series, selected sum, countable-choice schedule, or ambient Real-completeness datum.*

Proof. Package the carrier and the NameCert obligations first. The carrier names the rows $A, D, W, M, R, E, H, C, P, N$, and the obligations identify those rows as the local admission, readback, seal, transport, replay, provenance, and naming surface.

Next use the Real handoff theorem. It shows that a Real-facing or completion-facing consumer cannot inspect the seal row E before the source, dyadic endpoint, block witness, monotone comparison, and regular rational readback rows have been displayed.

Finally use the scoped-route theorem. It makes the dependency boundary public: every accepted read has already passed through the CauchyCondensationTest-facing route, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. The package therefore exports exactly the finite dyadic block-witness certificate and nothing from the excluded host or completeness surfaces. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$ read that exports a Real-facing block estimate without displaying the positive decreasing source, dyadic endpoints, finite witness ledger, monotone comparison, RegSeqRat readback, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: cauchy-condensation-packet, cauchy-condensation-theorem, realseries, dyadicratcore, regseqrat: none is bijective to $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$ because this

packet jointly carries one finite dyadic block witness ledger, its positive decreasing source row, endpoint core, monotone comparison, rational readback, Real seal, transport, replay, provenance, and local naming while refusing theorem-level summability.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationDyadicBlockWitness}\uparrow}$ **.**

Constructive story: A $\mathsf{CauchyCondensationDyadicBlockWitness}^{\uparrow}$ packet is generated from a positive decreasing RealSeries source row, a DyadicRatCore endpoint row, a finite block witness ledger, a monotone endpoint comparison row, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationDyadicBlockWitness}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 8587.1, Definition 8587.2, Definition 8587.3, Definition 8587.4, and Definition 8587.5 scope the finite dyadic block-witness route consumed by condensation packets and theorem-level Real handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationDyadicBlockWitness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host summability, equality of infinite series, selected sums, countable choice, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, block-witness ledger exactness, monotone comparison stability, regular readback nonescape, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationDyadicBlockWitnessUp.

Chapter 8588

A Concrete Naming Certificate for $\mathsf{OrthogonalArray}^{\uparrow}$

Following Rao and standard design theory, the $\mathsf{OrthogonalArray}^{\uparrow}$ packet records the finite incidence table whose every displayed projection of bounded strength has a balanced tuple count. It sits between the row-column-symbol discipline of chapter 6739, the sign-row orthogonality surface of chapter 7074, the finite table readback of chapter 49, the finite-set carrier of chapter 112, and the unary size ledger of chapter 28. The packet names only displayed finite design data; it is not a host classification theorem, a construction of all arrays, a quotient design equality, an infinite combinatorial principle, or an ambient coding-theory theorem.

Definition 8588.1 (Orthogonal-array carrier)**.** *An* $\mathsf{OrthogonalArray}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, F, L, t, W, B, M, S, H, C, P, N)$$

with the following displayed rows:

1. R *is the* $\mathsf{Nat}^{\uparrow}$ *row-count ledger naming the visible run boundary.*
2. F *is the* $\mathsf{FinSet}^{\uparrow}$ *factor row, exposing the finite family of coordinates inspected by the design.*
3. L *is the* $\mathsf{FinSet}^{\uparrow}$ *level row, exposing the finite value set shared by the displayed factors.*
4. t *is the* $\mathsf{Nat}^{\uparrow}$ *strength row, naming the maximum projection size whose balance is recorded.*
5. W *is the projection-window ledger, listing the displayed factor subsets of size at most* t*.*
6. B *is the balanced tuple-count ledger: for each window in* W *and each displayed level tuple,* B *records the finite row count in which that tuple appears.*
7. M *is the* $\mathsf{Mat}^{\uparrow}$ *readback row for the run-by-factor table.*
8. S *records row and coordinate scans that connect* M *to the factor and level ledgers.*
9. H *records componentwise* hsame *transport for row counts, factors, levels, strength, projection windows, tuple counts, matrix readback, scans, provenance, and local naming.*
10. C *records displayed* Cont *replay from an orthogonal-array inspection request through* $R, F, L, t, W, B, M, S, H, P, N$
11. P *is the* Pkg *provenance over* $\mathsf{OrthogonalArray}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{LatinSquare}^{\uparrow}$, $\mathsf{HadamardMatrix}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. N *is the local* $\mathsf{NameCert}_{\mathsf{OrthogonalArray}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for host array equality, selected isomorphism representatives, classification of parameters, infinite designs, or ambient coding-theory semantics.

Theorem 8588.2 (Orthogonal-array NameCert obligations)**.** *For every accepted* $\mathsf{OrthogonalArray}^{\uparrow}$ *carrier* $O = (R, F, L, t, W, B, M, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OrthogonalArray}^{\uparrow}}$ *surface consists exactly of row-count admission through* R*, factor and level admission through* F, L*, strength admission through* t*, projection-window exposure through* W*, balanced tuple-count exposure through* B*, matrix readback through* M*, scan compatibility through* S*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8588.1. The only size data are R and t, the only finite coordinate data are F, L, W, and the only displayed table data are B, M, S.

Read the matrix route next. The row-by-factor table is inspected through $\mathsf{Mat}^{\uparrow}$, while the factors and levels remain finite $\mathsf{FinSet}^{\uparrow}$ rows and the run count and strength remain $\mathsf{Nat}^{\uparrow}$ rows.

The balance obligation is then local to W and B. For each displayed projection window and level tuple, the tuple-count row records the visible count; no hidden host enumeration or quotient of arrays is available.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for parameter classification, selected isomorphism representatives, host equality, infinite designs, or coding-theory semantics, the listed rows exhaust the NameCert obligations. □

Theorem 8588.3 (Orthogonal-array projection balance)**.** *Every* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*, or* $\mathsf{Nat}^{\uparrow}$ *consumer that reads balance from an accepted* $\mathsf{OrthogonalArray}^{\uparrow}$ *carrier must pass through*

$$R, F, L, t \longmapsto W \longmapsto B \longmapsto M, S \longmapsto H, C, P, N.$$

The exported datum is exactly a displayed projection-window balance ledger. It may be consumed beside $\mathsf{LatinSquare}^{\uparrow}$ *and* $\mathsf{HadamardMatrix}^{\uparrow}$ *finite-design rows, but it does not construct a Latin square, construct a Hadamard matrix, classify orthogonal arrays, select a representative up to isomorphism, or assert host matrix equality.*

Proof. Start from the NameCert obligations of Definition 8588.2. They force every balance read through the row-count, factor, level, strength, projection-window, tuple-count, matrix, scan, transport, replay, provenance, and naming rows.

The projection window is finite. The row W lists the inspected factor subsets, and the row B records the balanced tuple counts for those windows and displayed level tuples.

The matrix readback is downstream of that finite ledger. The row M supplies the visible table, while S connects table entries to the factor and level rows without adding host matrix equality or a quotient of designs.

The sibling design routes are only comparison anchors. $\mathsf{LatinSquare}^{\uparrow}$ supplies a row-column-symbol finite design surface, and $\mathsf{HadamardMatrix}^{\uparrow}$ supplies a sign-row orthogonality surface; this packet exports the projection-balance surface and no construction or classification theorem for either sibling. □

Closure status for $\mathsf{NameCert}_{\mathsf{OrthogonalArray}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OrthogonalArray}^{\uparrow}$ packet is generated from Nat row-count and strength rows, FinSet factor and level rows, projection-window ledgers, balanced tuple-count rows, Matrix readback, scan compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{OrthogonalArray}\uparrow}$, seed)

Scope closed: Definition 8588.1, Definition 8588.2, and Definition 8588.3 seed the finite projection-balance carrier over displayed Matrix, FinSet, Nat, LatinSquare, and HadamardMatrix rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{OrthogonalArray}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close parameter existence, classification, selected isomorphism representatives, Latin-square construction, Hadamard-matrix construction, quotient design equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, factor and level exactness, strength exposure, projection-window coverage, balanced tuple-count readback, matrix readback, scan compatibility, transport, replay, provenance, and local naming for BEDC.Derived.OrthogonalArrayUp.

Chapter 8589

A Concrete Naming Certificate for $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$

Following Aronszajn–Panitchpakdi on hyperconvex metric spaces and injective metric extension, the $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$ packet records the finite NameCert surface where a displayed metric source may be read as an injective metric object. It sits between the metric source of chapter 57, the hyperconvex ball-family route of chapter 4013, the complete-metric boundary of chapter 128, the terminal real-radius rows of chapter 38, and the envelope consumer of chapter 6810. The packet names the Aronszajn–Panitchpakdi injective surface itself rather than another envelope synonym: finite extension requests, hyperconvex ball witnesses, completion visibility, and real-radius seals are displayed before any public consumer can treat the source as injective.

Definition 8589.1 (Aronszajn–Panitchpakdi injective metric carrier)**.** *An* $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, B, E, K, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the inspected point classifier and finite distance observations.*
2. *B is the* $\mathsf{HyperconvexMetric}^{\uparrow}$ *ball-family row, exposing finite closed-ball requests, real radii, pairwise compatibility, and the finite intersection handoff.*
3. *E is the finite injective-extension row, recording the admissible one-step extension boundary over the same metric source rather than a selected host hull point.*
4. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row, keeping completion-facing reads downstream of the metric source, hyperconvex ball data, and finite extension boundary.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius and distance row used by the metric observations, closed-ball tests, extension budgets, and completion boundary.*
6. *H records componentwise* hsame *transport for M, B, E, K, R, C, P, N.*
7. *C records displayed* Cont *replay from metric observations through ball-family compatibility, finite extension admissibility, complete-metric sealing, and real-radius readback.*
8. *P is the* Pkg *provenance over* $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HyperconvexMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}}$ *row.*
The carrier has no coordinate for arbitrary ball-family intersections, a chosen injective hull point, a quotient completion representative, countable choice, a host extension function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8589.2 (Aronszajn–Panitchpakdi injective metric obligations)**.** *For every accepted* AronszajnPanitchpakd *carrier $A = (M, B, E, K, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, metric-source exposure through M, hyperconvex finite ball-family exposure through B, finite injective-extension exposure through E, complete-metric sealing through K, real-radius and distance readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8589.1. The rows $M, B, E, K, R, H, C, P, N$ are the complete displayed inventory available to an injective-metric consumer.

Read the Aronszajn–Panitchpakdi side through the hyperconvex row. The row B supplies only the finite ball-family surface of chapter 4013: displayed closed balls, real radii, pairwise compatibility, and a finite intersection handoff over the same metric source.

Read the injective side through E. It records finite one-step extension admissibility before any envelope consumer from chapter 6810 may use the packet, and it does not select a host hull point or a global extension function.

Finally inspect K, R, H, C, P, N. The complete-metric and real-radius rows seal the same finite route, while transport, replay, provenance, and local naming preserve it without adding arbitrary-family intersections, quotient completions, countable choice, or ambient real completeness. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 8589.3 (Aronszajn–Panitchpakdi envelope-facing route)**.** *Every envelope-facing or completion-neighbour consumer of an accepted* $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$ *packet must read*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{HyperconvexMetric}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows M, B, K, R, H, C, P, N and must read the finite injective-extension boundary E before it treats the packet as an $\mathsf{InjectiveEnvelopeMetric}^{\uparrow}$ *source.*

Proof. Start with the carrier and obligation theorem. The metric source M is displayed before the hyperconvex row B, and B is the only row that exposes compatible closed-ball data.

The finite extension boundary E is read before an envelope-facing consumer can use the packet as injective. The completion seal K and real-radius row R remain downstream of the same metric and ball-family rows, so they cannot add a chosen hull point or an arbitrary intersection witness.

The structural rows H, C, P, N transport, replay, source, and name that same route. Thus every envelope-facing read factors through the displayed Aronszajn–Panitchpakdi surface rather than through an ungrounded injective-envelope synonym. □

Falsifiable prediction: Every accepted $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$ consumer that treats a metric packet as injective without first reading the displayed MetricUp, HyperconvexMetricUp, CompleteMetricUp, RealUp, finite extension, transport, replay, provenance, and local naming rows refutes the carrier surface.

Independence witness: metric, hyperconvex-metric, completemetric, real, injective-envelope-metric: no listed sibling supplies the whole packet $A = (M, B, E, K, R, H, C, P, N)$, because this chapter binds the Aronszajn–Panitchpakdi injective surface before the envelope consumer reads it.

Closure status for $\mathsf{NameCert}_{\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}}$.

Constructive story: An $\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}$ packet is generated from a metric source row, a hyperconvex finite ball-family row, a finite injective-extension boundary, a complete-metric seal, real radius and distance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8589.1, Definition 8589.2, and Definition 8589.3 seed the finite Aronszajn–Panitchpakdi injective metric surface over metric, hyperconvex, complete-metric, real-radius, finite-extension, and envelope-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AronszajnPanitchpakdiInjectiveMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, arbitrary-family intersections, selected injective hull points, quotient completions, countable choice, host extension functions, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, hyperconvex ball-family exposure, finite extension exposure, complete-metric sealing, Real radius readback, transport, replay, provenance, and local naming for BEDC.Derived.AronszajnPanitchpakdiInjectiveMetricUp.

Chapter 8590

A Concrete Naming Certificate for $\mathsf{FiniteLipschitzExtension}^{\uparrow}$

Following McShane, Whitney, and Bishop–Bridges constructive metric analysis, the $\mathsf{FiniteLipschitzExtension}^{\uparrow}$ packet records the finite certificate by which a Lipschitz extension request over a displayed finite subcarrier is read as a bounded ambient extension window. It sits between the reusable interface of chapter 3650, the McShane–Whitney envelope certificate of chapter 1941, the finite metric observations of chapter 3742, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite extension surface; it is not a host function-space construction, not a selected global formula, and not an appeal to countable choice or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8590.1 (Finite Lipschitz extension carrier)**.** *A* $\mathsf{FiniteLipschitzExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, A, L, B, G, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the metric observations used by the finite extension request.*
2. *A is the displayed finite subcarrier row, including membership evidence, inclusion into X, and the finite enumeration exposed to downstream readers.*
3. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row over A, carrying the finite graph window and displayed Lipschitz constant.*
4. *B is the* $\mathsf{RegSeqRat}^{\uparrow}$ *bound row recording the rational tolerance and constant reads used by the extension certificate.*
5. *G is the candidate finite ambient graph row, exposed only as finitely many source and target observations.*
6. *W is the refinement-window row comparing requested finite ambient observations with the enumerated source window.*
7. *R is the restriction-agreement row, requiring every source-admitted candidate read to replay the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row through A, L, B.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row for source values, candidate values, and rational tolerance readback.*
9. *H records componentwise* hsame *transport for the metric, finite subcarrier, source map, bound, graph, window, restriction, real seal, provenance, and local name rows.*

10. *C records displayed* Cont *replay from a finite ambient extension request through* $X, A, L, B, G, W, R, S, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{FiniteLipschitzExtension}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{McShaneExtension}^{\uparrow}$, $\mathsf{FiniteMetricSpace}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLipschitzExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function equality, quotient function spaces, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.FiniteLipschitzExtensionUp`

Theorem 8590.2 (Finite Lipschitz extension NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLipschitzExtension}^{\uparrow}$ *carrier* $F = (X, A, L, B, G, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteLipschitzExtension}^{\uparrow}}$ *surface consists exactly of carrier admission, finite subcarrier enumeration, source Lipschitz-map admission, rational bound readback, finite graph admission, refinement-window comparison, restriction-agreement exactness, real seal handoff, componentwise transport, displayed replay, provenance, and ledger non-escape. The finite route is*

$$X, A \longmapsto L \longmapsto B \longmapsto G, W \longmapsto R, S \longmapsto H, C, P, N.$$

No obligation is discharged by a host extension formula, an equality of ambient functions, or an unlisted completion principle.

Proof. Carrier admission is projection from Definition 8590.1. The displayed packet exposes exactly the rows $X, A, L, B, G, W, R, S, H, C, P, N$, so every local obligation must be read through that inventory.

The finite subcarrier and source-map clauses read A and L. A source value can enter the extension certificate only after the finite membership row has exposed its enumeration and the $\mathsf{LipschitzMap}^{\uparrow}$ row has supplied the displayed graph and bound data.

The rational bound, candidate graph, and window clauses read B, G, W. These rows keep the extension request finite: B supplies regular rational tolerances, G stores finite ambient graph observations, and W records the requested finite refinement window.

Restriction exactness is the row R, while the real-facing seal is the row S. The structural rows H, C, P, N preserve the same route componentwise and do not add host function equality, selected representatives, countable choice, or ambient completeness. Hence the listed finite route exhausts the local NameCert obligations. □

Theorem 8590.3 (Finite Lipschitz extension window factorization)**.** *Every* $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{McShaneExtension}^{\uparrow}$, $\mathsf{FiniteMetricSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLipschitzExtension}^{\uparrow}$ *packet must factor its finite extension observation through*

$$A, L, B \longmapsto G, W \longmapsto R, S$$

before exporting the observation to transport, replay, provenance, and the local NameCert *row. The factorization exports the displayed finite graph and its restriction route, not a host function object, not a selected global McShane envelope, and not ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a finite extension observation. The request first reads the finite subcarrier row A, because the source side of the observation is meaningful only against the displayed enumeration and inclusion route.

Next read L and B. The source $\mathsf{LipschitzMap}^{\uparrow}$ row supplies the finite graph and Lipschitz constant, while the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the rational tolerance budget used to compare source and ambient values.

The candidate graph and refinement rows G, W then determine the ambient window being exported. The restriction row R checks agreement over source-admitted points, and the real seal S is the only terminal real-facing handoff available to downstream consumers.

Finally the structural rows H, C, P, N replay and source that same factorization. The sibling certificates in chapter 3650 and chapter 1941 may consume the route, but this finite packet contributes only the displayed finite graph, refinement window, restriction, seal, transport, replay, provenance, and local naming rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLipschitzExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLipschitzExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8590.1, Definition 8590.2, and Definition 8590.3 seed the finite Lipschitz extension surface over displayed finite metric, Lipschitz-map, regular-rational, real, Lipschitz-extension, and McShane-extension rows.

Constructive story: A $\mathsf{FiniteLipschitzExtension}^{\uparrow}$ packet is generated from an ambient Metric row, a displayed finite subcarrier row, a source LipschitzMap row, a regular rational bound row, a finite ambient graph row, a refinement-window row, a restriction-agreement row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLipschitzExtension}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FiniteLipschitzExtensionUp`

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected global envelopes, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite subcarrier enumeration, source Lipschitz admission, rational bound readback, finite graph admission, window comparison, restriction exactness, real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.FiniteLipschitzExtensionUp.

Chapter 8591

A Concrete Naming Certificate for $\mathsf{FiniteCoverTree}^{\uparrow}$

Following Bishop and Bridges constructive compact metric practice, the $\mathsf{FiniteCoverTree}^{\uparrow}$ packet records a finite cover-refinement tree built from displayed ProbeBundle rows. It sits after the compact source of chapter 60, the metric observation rows of chapter 57, the compact metric finite-net surface of chapter 129, and the total-bounded finite-net carrier of chapter 127. The chapter names constructive finite routing data for compact-uniform and finite-cover consumers; it does not assert arbitrary open-cover compactness, selected subcovers, host equality of covers, countable choice, or ambient completeness.

Definition 8591.1 (Finite cover tree carrier)**.** *A* $\mathsf{FiniteCoverTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, M, B, L, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read only through finite compact-facing data.*
2. *M is the metric observation row used to compare inspected points with displayed centers.*
3. *B is the finite* ProbeBundle *row of centers, radii, membership witnesses, and refinement histories.*
4. *L is the level row recording the finite depth, parent pointer, and child list for each branch node.*
5. *E is the edge row recording that every child probe is a displayed refinement of its parent probe at the next requested tolerance.*
6. *R is the residual coverage row recording, for each branch node, which still-uncovered finite probe cells are routed to its children.*
7. *S is the stopping row recording leaves whose displayed probes already cover their local residual cells.*
8. *H records componentwise* hsame *transport for K, M, B, L, E, R, S and the structural rows.*
9. *C records displayed* Cont *replay through compact source, metric observation, probe bundle, refinement edges, residual coverage, and stopping leaves.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverTree}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverTree}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, M, B, L, E, R, S, H, C, P, N$. No coordinate stores an arbitrary open cover, a selected finite subcover, a host powerset, quotient equality of covers, countable choice, trichotomy, or ambient completeness.

Theorem 8591.2 (Finite cover tree NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverTree}^{\uparrow}$ *carrier*

$$T = (K, M, B, L, E, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverTree}^{\uparrow}}$ *surface consists exactly of carrier admission, compact source exposure through K, metric observation through M, finite* $\mathsf{ProbeBundle}$ *exposure through B, finite branch indexing through L, refinement-edge readback through E, residual coverage through R, stopping leaves through S, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier rows in Definition 8591.1. The rows K, M, B supply the compact, metric, and finite probe data that a cover-tree consumer can inspect.

Next read L, E, R, S. These rows give a finite tree: each branch has a displayed level, displayed child refinements, a residual-coverage ledger, and stopping leaves for the covered cells.

Finally read H, C, P, N. Transport, replay, provenance, and local naming preserve only the displayed rows, so the local certificate contains no route to arbitrary open-cover compactness, selected subcovers, host powersets, quotient cover equality, countable choice, trichotomy, or ambient completeness. □

Theorem 8591.3 (Finite cover tree branch stability)**.** *Let $T = (K, M, B, L, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCoverTree}^{\uparrow}$ *packet. Any transported branch read that preserves the finite* $\mathsf{ProbeBundle}$ *row B and the refinement-edge row E also preserves the residual coverage row R and the stopping row S. Thus a branch can be replayed only as a finite cover-tree route through displayed probe refinements, not as an arbitrary open-cover argument or a selected subcover theorem.*

Proof. Start with the finite probe row B and the edge row E. A branch read has no independent child selector; it reaches children only through the displayed refinement edges attached to the same finite probe bundle.

The residual coverage row R is indexed by the same branch levels and children as L and E. Preserving B and E therefore preserves the finite residual cells that each child is allowed to cover.

The stopping row S is downstream of R. Since H, C, P, N only transport, replay, source, and name this finite route, a transported branch cannot acquire an arbitrary open cover, selected subcover, host cover equality, or ambient compactness proof. □

Theorem 8591.4 (Finite cover tree sibling dependency)**.** *Every finite-cover consumer of an accepted* $\mathsf{FiniteCoverTree}^{\uparrow}$ *packet must first expose the compact source of chapter 60, the metric observation surface of chapter 57, and the total-bounded finite-net route of chapter 127. The tree then routes those rows through $B, L, E, R, S, H, C, P, N$ before any local cover-tree certificate is exported.*

Proof. Project the source rows K and M. They are the only rows through which compact and metric data enter the packet.

Read the finite-net dependency through B, L, E, R, S. The tree is finite because its probes, levels, refinement edges, residual coverage rows, and stopping leaves are all displayed before the local name is exported.

The structural rows H, C, P, N preserve the same dependency route. Therefore the sibling chapters are load-bearing sources of rows, and the exported certificate remains finite ProbeBundle cover-tree data. □

Theorem 8591.5 (Finite cover tree compact Cauchy cover handoff)**.** *Let $T = (K, M, B, L, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCoverTree}^{\uparrow}$ *packet. If a* $\mathsf{CompactCauchyCover}^{\uparrow}$ *packet consumes a branch read exported by T, then the consumer must first expose the finite* ProbeBundle *branch rows B, L, E, R, S from Definition 8591.1. Only after those rows have displayed the finite probes, branch levels, refinement edges, residual coverage, and stopping leaves may the* $\mathsf{CompactCauchyCover}^{\uparrow}$ *rows W, D, S, R, E be read as Cauchy-facing cover windows.*

Thus the handoff from finite cover refinements to chapter 1979 factors through the branch stability of Definition 8591.3 and the sibling dependency route of Definition 8591.4. It exports no arbitrary open-cover compactness, selected subcovers, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier by Definition 8591.1. The only branch rows available to a downstream cover consumer are B, L, E, R, S, together with the structural transport, replay, provenance, and local name rows.

Apply Definition 8591.3 to the branch read. Preserving the finite ProbeBundle row and refinement-edge row also preserves residual coverage and stopping leaves, so the exported branch cannot be reinterpreted as a hidden open cover or as a selected finite subcover.

Apply Definition 8591.4. The compact, metric, and total-bounded sources are consumed before the finite tree rows are exported, and chapter 1979 can then read its W, D, S, R, E window route only from this displayed finite branch data.

The excluded principles are absent from both packet inventories. Since neither the finite cover tree nor the compact Cauchy cover stores arbitrary open-cover compactness, selected subcovers, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverTree}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverTree}^{\uparrow}$ packet is generated from compact-source rows, metric-observation rows, finite ProbeBundle rows, branch-level rows, refinement-edge rows, residual-coverage rows, stopping-leaf rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCoverTree}^{\uparrow}}$, seed)

Scope closed: Definition 8591.1, Definition 8591.2, Definition 8591.3, Definition 8591.4, and Definition 8591.5 seed the finite compact-source, metric-observation, ProbeBundle, refinement-edge, residual-coverage, stopping-leaf, CompactCauchyCover handoff, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCoverTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host powersets, quotient equality of covers, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement-edge exactness, residual-coverage stability, stopping-leaf nonescape, sibling dependency, and branch transport for BEDC.Derived.FiniteCoverTreeUp.

Chapter 8592

A Concrete Naming Certificate for $\mathsf{StieltjesMeasure}^{\uparrow}$

Following Stieltjes and the Bishop–Bridges constructive reading of bounded variation, the $\mathsf{StieltjesMeasure}^{\uparrow}$ packet records the finite measure surface by which a bounded-variation distribution source supplies half-open interval masses before any Lebesgue–Stieltjes or real-valued measure consumer reads them. It sits after the partition-variation source of chapter 398, the endpoint-jump ledger of chapter 3528, the regular rational readback of chapter 301, the dyadic endpoint core of chapter 291, and the terminal real-name seal of chapter 38; it is the finite Stieltjes surface consumed by chapter 6047. The packet names only displayed half-open interval cells, endpoint-jump ledgers, finite additivity over inspected cells, regular rational and dyadic readback, Real sealing, transport, replay, provenance, and local naming.

Definition 8592.1 (Stieltjes measure carrier)**.** *A* $\mathsf{StieltjesMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, A, J, K, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *distribution source, read only through displayed finite partition-variation data.*
2. *S is the* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *endpoint-jump ledger that exposes the left and right boundary reads used by the interval cells.*
3. *A is the displayed half-open interval algebra generated by the inspected endpoints, with cells of the form $[a, b)$ and refinement witnesses for finite disjoint decompositions.*
4. *J is the endpoint-jump ledger assigning each displayed cell its boundary difference through the Stieltjes boundary packet.*
5. *K is the finite additivity row, checking only the inspected disjoint unions of cells in A.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows requested by the finite ledger.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row for dyadic endpoint and tolerance data in the same inspected cells.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the accepted interval-mass read.*

9. *H records componentwise* hsame *transport for the bounded-variation source, boundary row, interval algebra, jump ledger, additivity row, readbacks, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from bounded-variation partition reads through endpoint jumps, half-open interval cells, finite additivity, regular and dyadic readback, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
11. *P is the* Pkg *provenance over* $\mathsf{StieltjesMeasure}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{StieltjesMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, A, J, K, R, D, E, H, C, P, N$. It has no coordinate for a host sigma-algebra, completed outer measure, quotient equality of distribution functions, selected extension theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8592.2 (Stieltjes measure NameCert obligations)**.** *For every accepted* $\mathsf{StieltjesMeasure}^{\uparrow}$ *carrier*

$$M = (B, S, A, J, K, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StieltjesMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source B, Stieltjes boundary exposure through S, half-open interval algebra visibility through A, endpoint-jump exactness through J, finite additivity over inspected cells through K, regular rational readback through R, dyadic endpoint readback through D, terminal real sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8592.1. The source and boundary rows B, S supply the only distribution data: the bounded-variation source gives finite partition variation, and the Stieltjes boundary row gives endpoint jumps for displayed cells.

Read the interval algebra and ledgers next. The half-open cells of A are the only cells inspected by the packet, J assigns their endpoint-jump reads, and K checks finite additivity only for the displayed disjoint decompositions.

Finally consume the readback rows R, D, E and the structural rows H, C, P, N. The regular rational and dyadic readbacks are sealed by $\mathsf{Real}^{\uparrow}$, while transport, replay, provenance, and local naming preserve exactly the same finite rows. No host sigma-algebra, completed outer measure, quotient distribution equality, selected extension, countable-choice schedule, or ambient completeness row remains. □

Theorem 8592.3 (Stieltjes measure bounded-variation handoff)**.** *Every* $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$ *or measure-facing consumer of an accepted* $\mathsf{StieltjesMeasure}^{\uparrow}$ *packet receives the bounded-variation distribution only through*

$\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Co

Concretely, the consumer must display the bounded-variation source B, boundary row S, half-open interval algebra A, endpoint-jump ledger J, finite additivity row K, regular rational readback R, dyadic readback D, and terminal seal E before any Stieltjes interval mass is public.

Proof. Start with the bounded-variation source of chapter 398. It supplies partition-variation data and not a completed measure or host function quotient.

Pass through the boundary and interval rows. The Stieltjes boundary chapter supplies endpoint-jump reads for inspected cells, while the carrier here restricts those reads to the half-open algebra A and checks finite additivity through K.

Then apply the regular rational, dyadic, and Real rows. The readbacks R, D expose the requested windows and endpoint tolerances before the terminal seal E is visible. Thus a Lebesgue–Stieltjes or measure-facing consumer can read only the finite handoff route stated above, with no hidden extension theorem, countable choice, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{StieltjesMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{StieltjesMeasure}^{\uparrow}$ packet is generated from a BoundedVariation distribution source, a StieltjesMeasureBoundary endpoint-jump ledger, a displayed half-open interval algebra, finite additivity over inspected cells, RegSeqRat and DyadicRatCore readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StieltjesMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8592.1, Definition 8592.2, and Definition 8592.3 seed the finite Stieltjes measure surface from bounded-variation endpoint jumps to half-open interval masses and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StieltjesMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sigma-additivity, sigma-algebra completion, completed outer measure, selected extension theorems, quotient equality of distribution functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Stieltjes boundary exposure, half-open interval visibility, endpoint-jump exactness, finite additivity, RegSeqRat and DyadicRatCore readback, Real-seal non-escape, transport and replay stability, provenance non-escape, and a Lean carrier formalization for BEDC.Derived.StieltjesMeasureUp.

Chapter 8593

A Concrete Naming Certificate for $\mathsf{CountableMetricName}^{\uparrow}$

Following Bishop and Bridges constructive analysis and represented-space computable analysis, the $\mathsf{CountableMetricName}^{\uparrow}$ packet names the finite interface by which a countably represented source is read as metric-name data without leaving the BEDC carrier discipline. It sits between the finite observation schedules of chapter 126, the metric carrier surface of chapter 57, the dyadic tolerance rows of chapter 291, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing countable metric route of chapter 4044. The finite observation windows used by the countable-name route are the finite request/list/window schedules of chapter 5916, so a $\mathsf{CountableMetricName}^{\uparrow}$ consumer reaches metric-name data only after that finite-prefix surface has been displayed. The packet records a named handoff from certified countable observations to metric-name data; it is not a quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8593.1 (Countable metric name carrier)**.** *A* $\mathsf{CountableMetricName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the represented source row, naming the countable source, its displayed enumeration window, and the admitted source tokens read by the metric-name request.*
2. *M is the metric row, exposing the source-side distance classifier, finite ball ledger, and equality boundary only through the displayed metric packet.*
3. *W is the family of finite* $\mathsf{StreamName}^{\uparrow}$ *windows used to inspect source tokens, distance requests, and metric-name prefixes.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for requested radii, comparison budgets, slack witnesses, and refusal rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite distance comparisons that survive regular rational readback.*
7. *H records componentwise* hsame *transport for the represented source, metric row, stream windows, dyadic tolerance, regular readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from any metric-name request through $S, M, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CountableMetricName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableMetricName}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, W, D, R, E, H, C, P, N$. It has no coordinate for quotient equality of streams, a selected completion limit, countable choice, host real equality, a hidden distance oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CountableMetricNameUp.CountableMetricNameTasteGate single carrier alignment`

Theorem 8593.2 (CountableMetricName obligation surface)**.** *For an accepted* $\mathsf{CountableMetricName}^{\uparrow}$ *packet $A = (S, M, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CountableMetricName}^{\uparrow}}$ *surface consists exactly of carrier admission, represented-source exposure, metric-row exposure, finite stream-window scheduling, dyadic tolerance checking, regular rational readback, Real sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and ledger non-escape.*

Proof. Carrier admission is Definition 8593.1: every row read by a consumer is one of $S, M, W, D, R, E, H, C, P, N$.

The source and metric obligations are finite because the represented source is inspected only through S and the metric classifier only through M. Stream-name data is reduced to the displayed windows W, tolerance data is reduced to D, and every numerical readback must pass through R before the Real seal E is consumed.

Transport, replay, and provenance are the remaining exposed routes. The rows H, C, and P force componentwise identity transport, continuation replay, and package ancestry to remain inside the same displayed packet, while N names exactly this finite surface.

There is no additional obligation for quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects is a coordinate of the carrier. □

Theorem 8593.3 (CountableMetricName finite observation scope)**.** *Every legal consumer read of a* $\mathsf{CountableMetricName}^{\uparrow}$ *carrier factors through a finite observation route*

$$S \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

over the displayed metric row M, followed only by the transport, replay, provenance, and local naming rows H, C, P, N.

Proof. Inspect a consumer read of the carrier. A request first chooses a displayed source token from S and a finite metric-name prefix from the stream windows W; no unbounded source enumeration is exported.

The requested tolerance is then checked by the dyadic row D. Any distance or comparison data that survives the tolerance check is read back through the regular rational row R and sealed by E as a finite $\mathsf{Real}^{\uparrow}$ observation.

Finally, the read is transported and replayed only by H and C, with provenance recorded by P and the local NameCert row N. Since the route never leaves these displayed rows, it cannot recover quotient stream equality, a hidden representative, countable choice, host real equality, or ambient completeness. □

Theorem 8593.4 (CountableMetricName completion handoff)**.** *A* $\mathsf{CountableMetricName}^{\uparrow}$ *carrier exports to completion-facing consumers only the represented source row, metric row, finite stream*

windows, dyadic tolerance row, regular rational readback, Real seal, componentwise transport, replay, provenance, and local naming data. Any later $\mathsf{CountableMetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer must rebuild completion or real-completeness claims from those displayed rows rather than reading a selected limit, quotient equality, countable-choice schedule, host real equality, or ambient completeness from the handoff.*

Proof. The handoff payload is the carrier tuple $S, M, W, D, R, E, H, C, P, N$. A completion-facing consumer may reuse the represented source row S, the metric row M, the finite windows W, the tolerance ledger D, the regular readback R, and the Real seal E.

The same consumer may also transport and replay the packet through H and C, and may cite P and N for provenance and local naming. These are finite BEDC rows, so they can be passed to the countable metric completion route without changing the classifier boundary.

Nothing in the packet selects a limit or identifies two streams by quotient equality. Consequently, any completion theorem, separated comparison, or Real completeness claim must be supplied by a later completion certificate; it is not exported by $\mathsf{CountableMetricName}^{\uparrow}$ itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableMetricName}^{\uparrow}}$.

Constructive story: A $\mathsf{CountableMetricName}^{\uparrow}$ packet is generated by a represented source row, a metric row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableMetricName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8593.1, Definition 8593.2, Definition 8593.3, and Definition 8593.4 bind the finite carrier and handoff surface for countable observations read as metric-name data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableMetricName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, hidden distance oracles, ambient $\mathsf{Real}^{\uparrow}$ completeness, or completion completeness.

Upgrade path: Scoped closure requires Lean-backed NameCert obligations for the represented source, metric row, finite windows, tolerance, readback, Real seal, transport, replay, and completion handoff.

Chapter 8594

A Concrete Naming Certificate for $\mathsf{RegularRealSeparationModulus}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularRealSeparationModulus}^{\uparrow}$ packet records the finite positive-gap data by which two regular rational real names are separated before a Real or Completion consumer reads an apartness conclusion. It sits after the regular tail-equivalence route of chapter 5940, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the positive-separation boundary of chapter 304. The packet names a displayed modulus for a strict finite gap; it is not quotient real equality, trichotomy, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8594.1 (Regular real separation modulus carrier)**.** *A* $\mathsf{RegularRealSeparationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, W, D, T, E, A, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows whose regular rational windows are compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected tail request for both readbacks.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger. It records the radius ρ and denominator budget used by the comparison.*
4. *T is the* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *refusal row. It states that the inspected tails do not collapse through the sibling tail-equivalence route at the displayed radius.*
5. *E is the pair of terminal* $\mathsf{Real}^{\uparrow}$ *seal rows made available only after L, R, W, D, T have been displayed.*
6. *A is the apartness handoff row to the sibling apartness-real certificate of chapter 304.*
7. *H records componentwise* hsame *transport for $L, R, W, D, T, E, A, C, P, N$.*
8. *C records displayed* Cont *replay of the route*

$$L, R, W, D \longmapsto T \longmapsto E, A \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularRealSeparationModulus}^{\uparrow}$, $\mathsf{RegularRealTailEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRealSeparationModulus}^{\uparrow}}$ *row.*

The classifier reads only these rows. It has no coordinate for quotient real equality, host trichotomy, a selected real representative, countable-choice tail selection, or an ambient completeness witness.

Theorem 8594.2 (Regular real separation modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealSeparationModulus}^{\uparrow}$ *carrier* $M = (L, R, W, D, T, E, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularRealSeparationM}}$ *surface consists exactly of regular rational readback admission for* L, R*, shared finite-window exposure for* W*, positive dyadic radius and denominator-budget exactness for* D*, tail-equivalence refusal through* T*, delayed real-seal access through* E*, apartness handoff through* A*, componentwise transport, displayed continuation replay, package provenance, and local naming through* H, C, P, N*.*

Proof. Project the carrier of Definition 8594.1. The only source rows are the two regular rational readbacks, the shared finite stream window, and the dyadic positive-radius ledger.

The row T is downstream of those source rows. It records that the displayed tails do not collapse through the regular tail-equivalence route at the radius supplied by D.

Only after T is displayed may the terminal real seals E and the apartness handoff A be read. The structural rows H, C, P, N transport, replay, source, and name exactly that finite route, so the certificate exports no quotient equality, trichotomy, selected representative, choice schedule, or ambient completeness datum. □

Theorem 8594.3 (Regular real separation modulus finite-gap route)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{RegularRealSeparationModulus}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularRealTailEquivalence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{NameCert}.$$

The resulting apartness read is therefore a positive-gap conclusion over displayed regular rational windows, not a quotient-real comparison, a trichotomy branch, a selected-limit argument, or an ambient completeness principle.

Proof. Start from an accepted carrier $M = (L, R, W, D, T, E, A, H, C, P, N)$. The finite comparison begins with the two regular rational readbacks L, R, the shared window W, and the dyadic radius row D.

Read the tail-equivalence refusal next. Because T is the only row between the source windows and the real seals, any downstream apartness consumer must use the positive radius and denominator budget already displayed in D.

Finally, the terminal seals and apartness handoff are transported and replayed by H, C, P, N. Since the carrier contains no stronger host-side principle, the consumer receives only the finite positive-gap route named by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealSeparationModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularRealSeparationModulus}^{\uparrow}$ packet is generated from two RegSeqRat readback rows, one shared StreamName finite-window row, a DyadicRatCore positive-radius ledger, a RegularRealTailEquivalence refusal row, terminal Real seal rows, an apartness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularRealSeparationModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8594.1, Definition 8594.2, and Definition 8594.3 seed the finite positive-gap route for regular rational real names.

Formal verification: FormalStatus(NameCert$_{\text{RegularRealSeparationModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, trichotomy, selected representatives, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius exactness, tail-equivalence refusal, real-seal delay, apartness handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularRealSeparationModulusUp.

Chapter 8595

A Concrete Naming Certificate for HyperbolicSectorNerve$^{\uparrow}$

The HyperbolicSectorNerve$^{\uparrow}$ packet realizes the residual-sector boundary of chapter 9378 as finite MetaCIC NameCert data. It sits beside the confluence substitution boundary of chapter 3791, the local join ledger of chapter 4126, the residual checker of chapter 3827, and the parallel-diamond frontier of chapter 4238. The carrier names a finite residual-sector nerve: branches, candidate joins, retained obstructions, transport, replay, provenance, and local naming are explicit rows, while full Church–Rosser, unrestricted substitution compatibility, subject reduction, evaluator equality, and kernel consistency remain outside the packet.

Definition 8595.1 (Hyperbolic sector nerve carrier)**.** *A* HyperbolicSectorNerve$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, B, J, O, T, R, G, H, C, P, N)$$

with the following displayed rows:

1. *W is the typed residual-window row naming the finite source term window and its local context.*
2. *B is the residual-branch row recording the two finite reduction branches inspected inside W.*
3. *J is the candidate-join row recording the proposed common reducts visible at the inspected branch tips.*
4. *O is the obstruction-retention row naming the residual-substitution boundary still required before a full diamond can be consumed.*
5. *T is the transport row for branch, join, and obstruction data under componentwise* hsame*.*
6. *R is the replay row for the finite residual-sector route through W, B, J, O, T.*
7. *G is the TasteGate readiness row, reserving the later* Nontrivial *and* FieldFaithful *instance obligations for this finite carrier surface.*
8. *H records componentwise* hsame *transport for $W, B, J, O, T, R, G, C, P, N$.*
9. *C records displayed* Cont *replay through residual windows, branches, candidate joins, obstruction retention, transport, TasteGate readiness, provenance, and naming.*
10. *P is the* Pkg *provenance over* HyperbolicSectorNerve$^{\uparrow}$*, the MetaCIC confluence audit map, residual-substitution boundary rows, local-join rows, residual-checker rows, parallel-diamond frontier rows,* BHist*,* hsame*,* Cont*,* TasteGate*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorNerve}^{\uparrow}}$ *row.*
The carrier has no coordinate for a full parallel diamond, closed Church–Rosser, unrestricted substitution compatibility, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 8595.2 (Hyperbolic sector nerve NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ *carrier*

$$S = (W, B, J, O, T, R, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicSectorNerve}^{\uparrow}}$ *surface consists exactly of typed residual-window exposure through W, residual-branch exposure through B, candidate-join exposure through J, obstruction retention through O, transport through T, H, replay through R, C, TasteGate readiness through G, provenance through P, and local naming through N.*

Proof. Project the finite carrier. The rows W, B, J display the inspected residual sector: a typed window, two local branches, and candidate joins at the branch tips.

Read O before any diamond-facing consumer. This retained obstruction is the residual-substitution boundary named by the confluence audit map, so the carrier cannot turn the candidate joins into a full confluence theorem.

Finally read T, R, G, H, C, P, N. Transport and replay preserve the same finite sector, while the TasteGate readiness row records the later Nontrivial and FieldFaithful obligations without treating them as checked evidence. Since no stronger global row occurs in the carrier, the listed rows exhaust the local certificate. □

Theorem 8595.3 (Hyperbolic sector nerve audit-map concretization)**.** *Every confluence-audit read exported by an accepted* $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ *packet factors through finite residual-branch, candidate-join, obstruction-retention, transport, replay, provenance, and local NameCert rows. The read may feed a conditional parallel-diamond consumer, but it does not discharge residual-substitution compatibility, prove full Church–Rosser, prove subject reduction, decide evaluator equality, or establish kernel consistency.*

Proof. Start with W, B, J. These rows display only a finite local sector of the residual graph and its candidate joins.

Next inspect O. The obstruction row remains visible before the handoff is consumed, so the packet records the audit-map boundary instead of erasing it.

The rows T, R, H, C, P, N transport, replay, source, and name the same finite sector. Therefore the exported read is a concrete conditional-boundary packet, not an unrestricted confluence theorem or kernel metatheorem. □

Theorem 8595.4 (Hyperbolic sector nerve TasteGate readiness)**.** *The later Lean TasteGate file for* $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ *must provide* Nontrivial *and* FieldFaithful *instances at the displayed carrier surface*

$$W, B, J, O, T, R, G, H, C, P, N.$$

Nontriviality must exhibit an accepted packet with all finite residual-sector rows present. Field faithfulness must show that deleting any of the displayed branch, join, obstruction, transport, replay, provenance, or naming rows changes the readable carrier surface.

Proof. The readiness row G records the instance boundary but not the instance proof. It is downstream of W, B, J, O and upstream of the structural rows, so it cannot hide a missing residual-sector coordinate.

The nontriviality obligation is therefore tied to the existence of a displayed packet with all rows present. Field faithfulness is tied to componentwise comparison of those rows under H, C, P, N, ensuring that the future Lean surface cannot collapse the residual branch, candidate join, or obstruction-retention data into an empty marker. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ confluence-audit read exposes the obstruction-retention row O before any conditional parallel-diamond handoff is consumed.

Independence witness: metacic-confluence-substitution-boundary, metacic-local-join-ledger, metacic-confluence-residual-checker, metacic-parallel-diamond-frontier: none is bijective with $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ because this packet jointly carries a typed residual window, residual branches, candidate joins, obstruction retention, transport, replay, TasteGate readiness, provenance, and local naming as one finite nerve surface.

The Lean TasteGate file later supplies $\mathsf{Nontrivial}\ \mathsf{HyperbolicSectorNerve}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicSectorNerve}^{\uparrow}$ instances for the rows $W, B, J, O, T, R, G, H, C, P, N$. Those instances are obligations of the future Lean surface, not paper-side checked evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSectorNerve}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicSectorNerve}^{\uparrow}$ packet is generated from a typed residual window, residual-branch rows, candidate-join rows, obstruction-retention rows, transport rows, replay rows, TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8595.1, Definition 8595.2, Definition 8595.3, and Definition 8595.4 seed the finite residual-sector, candidate-join, obstruction-retention, transport, replay, TasteGate-readiness, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorNerve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, full parallel diamond, closed Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-branch exposure, candidate-join exactness, obstruction retention, transport stability, replay stability, provenance exhaustion, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.HyperbolicSectorNerveUp.

Chapter 8596

A Concrete Naming Certificate for $\mathsf{HyperbolicAreaDefect}^{\uparrow}$

Following classical hyperbolic geometry and the Gauss–Bonnet area-defect formula, the $\mathsf{HyperbolicAreaDefect}^{\uparrow}$ packet records the finite boundary-triangle route by which a Poincare disk area element is read into an ideal-triangle area comparison. It sits after the Poincare disk area-element row of chapter 7916, the ideal-triangle boundary row of chapter 6544, the hyperbolic metric row of chapter 3474, and the scalar real seal of chapter 38. The packet names only the finite boundary-area defect surface; it is not a host integration theorem, a completed moduli space, selected geodesic representatives, or an ambient curvature theorem.

Definition 8596.1 (Hyperbolic area-defect carrier)**.** *A* $\mathsf{HyperbolicAreaDefect}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Delta = (A, T, M, R, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{PoincareDiskAreaElement}^{\uparrow}$ *row. It records the local disk area-element read used by the defect computation.*
2. *T is the* $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$ *row. It records the ordered boundary triangle whose defect is inspected.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row. It supplies the metric handoff tying the disk area element and boundary triangle to the same hyperbolic surface.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row for the displayed area-defect read.*
5. *B is the boundary-triangle ledger. It records that the three boundary vertices, side windows, metric read, area-element read, and scalar seal are consumed in the displayed order.*
6. *H records componentwise* hsame *transport for A, T, M, R, B, C, P, N.*
7. *C records displayed* Cont *replay of the route*

$$A, T, M \longmapsto B \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicAreaDefect}^{\uparrow}$, $\mathsf{PoincareDiskAreaElement}^{\uparrow}$, $\mathsf{HyperbolicIdealTriangle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicAreaDefect}^{\uparrow}}$ *row.*

The classifier reads only the displayed area element, ideal-triangle boundary, metric, scalar seal, boundary ledger, transport, replay, provenance, and naming rows. It has no coordinate for host integration, selected geodesic representatives, quotient boundary configurations, or a global Gauss–Bonnet theorem.

Theorem 8596.2 (Hyperbolic area-defect NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicAreaDefect}^{\uparrow}$ *carrier* $\Delta = (A, T, M, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicAreaDefect}^{\uparrow}}$ *surface consists exactly of Poincare disk area-element admission through* A*, ideal-triangle admission through* T*, hyperbolic metric compatibility through* M*, boundary-triangle ledger visibility through* B*, delayed Real scalar sealing through* R*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8596.1. The only geometric source rows are the Poincare disk area element, the ideal-triangle boundary data, and the hyperbolic metric handoff.

The boundary-triangle ledger B binds those source rows before any scalar seal is read. Thus a defect consumer cannot bypass the displayed boundary vertices, side windows, metric read, or area-element read.

The Real row R is downstream of that ledger, and H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no row for host integration, selected geodesics, quotient boundary configurations, or a global curvature theorem, the listed obligations exhaust the local certificate surface. □

Theorem 8596.3 (Hyperbolic area-defect boundary-triangle route)**.** *Every accepted* $\mathsf{HyperbolicAreaDefect}^{\uparrow}$ *read factors through*

$$\mathsf{PoincareDiskAreaElement}^{\uparrow}, \mathsf{HyperbolicIdealTriangle}^{\uparrow}, \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{NameCert}.$$

The scalar defect is therefore consumed as a finite boundary-triangle area readback, not as an ambient integration theorem, selected ideal-triangle representative, quotient boundary configuration, completed moduli-space point, or host Gauss–Bonnet principle.

Proof. Begin with the source rows A, T, M. They display the local Poincare disk area element, the ordered ideal-triangle boundary data, and the hyperbolic metric handoff on the same inspected surface.

Next read the boundary-triangle ledger B. It records the order in which boundary vertices, side windows, metric data, and area-element data are exposed before a scalar value is sealed.

Finally, the Real row and structural rows R, H, C, P, N seal and replay that same ledger. No extra coordinate remains from which a host integration theorem, selected geodesic representative, quotient boundary object, completed moduli point, or global curvature theorem could be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicAreaDefect}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicAreaDefect}^{\uparrow}$ packet is generated from a PoincareDiskAreaElement row, HyperbolicIdealTriangle row, HyperbolicMetric row, Real scalar seal row, boundary-triangle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicAreaDefect}^{\uparrow}}$, seed)

Scope closed: Definition 8596.1, Definition 8596.2, and Definition 8596.3 seed the finite boundary-triangle area-defect route through Poincare disk area-element, ideal-triangle, metric, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicAreaDefect}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host integration, selected geodesic representatives, quotient boundary configurations, completed moduli spaces, a global Gauss–Bonnet theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked area-element admission, ideal-triangle admission, metric compatibility, boundary-triangle ledger visibility, scalar-seal delay, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicAreaDef

Chapter 8597

A Concrete Naming Certificate for $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ packet records the finite squeeze route that ties regular Cauchy lattice operations to the terminal $\mathsf{Real}^{\uparrow}$ seal without using host trichotomy, quotient streams, or a selected limit. It sits beside the minimum and maximum surfaces of chapter 1528 and chapter 1502, the lattice-operation surface of chapter 5727, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, and the terminal real-name seal of chapter 38. The packet names a constructive squeeze certificate, not a host order theorem, ambient completeness claim, or quotient-real comparison.

Definition 8597.1 (RegularCauchyLatticeSqueeze carrier)**.** *A* $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, W, D, \ell, u, R, E, H, C, P, N)$$

with the following rows:

1. *L is the lower-window row, carrying the displayed minimum route read from the accepted regular Cauchy lattice surface.*
2. *U is the upper-window row, carrying the displayed maximum route read from the same finite surface.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *observation row exposing the finite windows used by both lattice routes.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the displayed radius budget and comparison scale.*
5. *ℓ is the lower-route exactness row, forcing the lower window to pass through the same finite squeeze route as the minimum surface.*
6. *u is the upper-route exactness row, forcing the upper window to pass through the same finite squeeze route as the maximum surface.*
7. *R is the terminal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the common squeeze route.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R has been displayed.*

9. *H records componentwise* hsame *transport for the lower and upper routes, the shared observation row, dyadic tolerance, exactness rows, readback, seal, provenance, and local naming.*
10. *C records displayed* Cont *replay from the squeeze request through $L, U, W, D, \ell, u, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$, $\mathsf{RegularCauchyMinimum}^{\uparrow}$, $\mathsf{RegularCauchyMaximum}^{\uparrow}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, W, D, \ell, u, R, E, H, C, P, N$. It has no coordinate for host trichotomy, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8597.2 (RegularCauchyLatticeSqueeze NameCert obligations)**.** *For every accepted* RegularCauchyLattice *carrier $S = (L, U, W, D, \ell, u, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower-window row, upper-window row, shared StreamName observation row, DyadicRatCore tolerance ledger, lower-route exactness row, upper-route exactness row, RegSeqRat readback row, Real seal row, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the accepted carrier. The rows $L, U, W, D, \ell, u, R, E, H, C, P, N$ are the entire public inventory of the squeeze packet.

The lower and upper routes are displayed first. The lower route row ℓ and the upper route row u force the shared observation row W and dyadic tolerance row D to be read before the common readback row R is available.

The readback row R is downstream of the same displayed squeeze route. The structural rows H, C, P, N transport, replay, source, and name exactly that finite route. Since the carrier contains no coordinate for trichotomy, quotient stream equality, selected limits, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8597.3 (RegularCauchyLatticeSqueeze lower-window route)**.** *Let $S = (L, U, W, D, \ell, u, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ *carrier. Any lower-window consumer must read the finite route*

$$L \longmapsto \ell \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

Thus the lower-window route exports only the displayed lower-window row, the lower-route exactness row, the shared observation row, the dyadic tolerance ledger, the regular readback, and the terminal real seal.

Proof. Start with Definition 8597.2. The lower-window row is available only after L has been displayed, and the exactness row ℓ is the next lower-route commitment.

The shared observation row W and dyadic ledger D are then consumed before the readback row R appears. The seal row E remains downstream of that lower route, so no host trichotomy, quotient comparison, or selected limit enters the packet. □

Theorem 8597.4 (RegularCauchyLatticeSqueeze upper-window route)**.** *Let $S = (L, U, W, D, \ell, u, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ *carrier. Any upper-window consumer must read the finite route*

$$U \longmapsto u \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

Thus the upper-window route exports only the displayed upper-window row, the upper-route exactness row, the shared observation row, the dyadic tolerance ledger, the regular readback, and the terminal real seal.

Proof. Start with the same NameCert obligations. The upper-window row is available only after U has been displayed, and the exactness row u is the next upper-route commitment.

The shared observation row W and dyadic ledger D are then consumed before the readback row R appears. The seal row E remains downstream of that upper route, so no quotient stream equality or ambient completeness is exported by the packet. □

Theorem 8597.5 (RegularCauchyLatticeSqueeze RealUp-seal nonescape)**.** *Let $S = (L, U, W, D, \ell, u, R, E, H, C, P$ be an accepted* $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer reaches the seal E only through the displayed lower-window route and upper-window route together with the common readback row R. The consumer therefore receives no host trichotomy, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply the lower-window route theorem and the upper-window route theorem. Both force the consumer through the same shared observation row, dyadic tolerance ledger, and regular readback before the seal row is available.

The structural rows H, C, P, N preserve and name that shared route. Since the carrier has no excluded completeness coordinate, the terminal seal is a finite readback boundary rather than a host-order or quotient argument. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ read displays one lower-window row, one upper-window row, one shared $\mathsf{StreamName}^{\uparrow}$ observation row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, lower-route exactness, upper-route exactness, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before a minimum, maximum, or Real-facing consumer may read the squeeze route.

Independence witness: regularcauchymin, regularcauchymax, regseqrat, dyadicratcore, streamname, real: $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the lower-window, upper-window, shared observation, dyadic tolerance, lower exactness, upper exactness, regular readback, real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}$ packet is generated from lower-window and upper-window rows, a shared StreamName observation row, a DyadicRatCore tolerance ledger, lower-route and upper-route exactness rows, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}}$, seed)

Scope closed: Definition 8597.1, Definition 8597.2, Definition 8597.3, Definition 8597.4, and Definition 8597.5 seed the finite squeeze route over minimum, maximum, StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeSqueeze}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient stream equality, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-route exactness, upper-route exactness, route stability, provenance, and non-escape for the regular squeeze TasteGate target.

Chapter 8598

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$

The $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ packet realizes the composable observer-phase transport route of chapter 9463 as finite observer-phase, phase-distance, boundary-shadow, transport-budget, transport, replay, provenance, and NameCert data. It sits beside the observer-phase packet of chapter 7112, the phase-distance shell packet of chapter 6017, the observer-phase groupoid packet of chapter 6146, the transport-budget packet of chapter 7134, and the boundary-shadow packet of chapter 3813. The packet names a finite cocycle reading; it is not a global hyperbolic space, a completed observer geometry, or a host transport theorem.

Definition 8598.1 (Hyperbolic phase expansion cocycle carrier)**.** *A* $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O, D, B, T, G, R, H, C, P, N)$$

with the following rows:

1. *O is the observer-phase row.*
2. *D is the phase-distance row.*
3. *B is the boundary-shadow row.*
4. *T is the transport-budget row.*
5. *G is the observer-phase groupoid row that composes the displayed route.*
6. *R is the cocycle readback row, recorded only after the displayed transport and distance data have been inspected.*
7. *H records componentwise* hsame *transport for the observer-phase, phase-distance, boundary-shadow, transport-budget, groupoid, readback, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the cocycle request through $O, D, B, T, G, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicObserverPhase}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, D, B, T, G, R, H, C, P, N$. It has no coordinate for a completed boundary, host phase equality, quotient transport class, or ambient completeness.

Theorem 8598.2 (Hyperbolic phase expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ *carrier* $Q = (O, D, B, T, G, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseExpansi}}$ *surface consists exactly of carrier admission for observer-phase, phase-distance, boundary-shadow, transport-budget, observer-phase groupoid, cocycle readback, componentwise transport, displayed replay, provenance, and local naming rows.*

Proof. Project the accepted carrier. The rows $O, D, B, T, G, R, H, C, P, N$ are the public inventory of the cocycle packet.

The observer-phase, distance, boundary-shadow, and transport-budget rows are displayed before the readback row R is available. The groupoid row G composes those rows into the finite cocycle route, while the structural rows H, C, P, N transport, replay, source, and name that same route.

Since the carrier contains no completed boundary, host phase equality, quotient transport class, or ambient completeness coordinate, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ read exposes observer-phase, phase-distance, boundary-shadow, transport-budget, observer-phase groupoid, cocycle readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any public cocycle consumer receives the packet.

Independence witness: hyperbolic-observer-phase, phase-distance-observer-expansion, observer-phase-groupoid, hyperbolic-phase-transport-budget, hyperbolic-boundary-shadow: HyperbolicPhaseExpansionCocyc is not bijective to any listed sibling because it jointly binds observer-phase, phase-distance, boundary-shadow, transport-budget, groupoid, readback, transport, replay, provenance, and local naming rows into one finite cocycle surface.

The Lean TasteGate follow-up for this packet must provide explicit Nontrivial $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ instances before a checked carrier-alignment theorem can witness the public route.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}$ packet is generated from observer-phase, phase-distance, boundary-shadow, transport-budget, observer-phase groupoid, cocycle readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}}$, seed)

Scope closed: Definition 8598.1, Definition 8598.2, the falsifiable prediction, and the independence witness seed the finite observer-phase expansion cocycle route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicPhaseExpansionCocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed boundary geometry, host phase equality, quotient transport classes, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observer-phase exposure, phase-distance exposure, boundary-shadow exposure, transport-budget exposure, groupoid composition, cocycle readback, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for the hyperbolic phase expansion cocycle target.

Chapter 8599

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ packet records the finite handoff by which Cauchy-name data reaches completion-facing $\mathsf{Real}^{\uparrow}$ seals without importing a classical real-completeness principle. It sits beside the stream-name route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, the countable metric name route of chapter 8593, the countable metric completion route of chapter 4044, and the finite-prefix request surface of chapter 5916. The packet names the displayed bridge from finite Cauchy observations to completion-facing real seals; it is not a quotient stream equality, selected limit operator, countable-choice schedule, trichotomy principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 8599.1 (Constructive Cauchy real completion bridge carrier)**.** *A* ConstructiveCauchyRealCompletionBric *carrier is a finite* BHist *packet*

$$B = (S, R, D, E, M, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule for Cauchy-name observations, including the request prefixes consumed by the bridge.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those displayed stream windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for comparison scales, slack witnesses, and refusal rows.*
4. *E is the completion-facing* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D have been displayed.*
5. *M is the* $\mathsf{CountableMetricName}^{\uparrow}$ *row naming the metric-name data read by the bridge.*
6. *F is the finite* $\mathsf{FiniteStream}^{\uparrow}$ *prefix schedule that bounds every request before a Cauchy-name or metric-name read is consumed.*
7. *H records componentwise* hsame *transport for the stream, regular rational, dyadic, real-seal, metric-name, finite-prefix, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bridge request through $F, S, R, D, E, M, H, P, N$.*

9. *P is the* Pkg *provenance over* ConstructiveCauchyRealCompletionBridge$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CountableMetricName$^{\uparrow}$, CountableMetricCompletion$^{\uparrow}$, FiniteStream$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{ConstructiveCauchyRealCompletionBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, E, M, F, H, C, P, N$. It has no coordinate for quotient equality of Cauchy streams, selected limits, countable choice, trichotomy, hidden representatives, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8599.2 (Constructive Cauchy real completion bridge NameCert obligations)**.** *For an accepted* ConstructiveCauchyRealCompletionBridge$^{\uparrow}$ *carrier $B = (S, R, D, E, M, F, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCauchyRealCompletionBridge}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-prefix exposure through F, stream-window exposure through S, regular rational readback through R, dyadic tolerance checking through D, completion-facing* Real$^{\uparrow}$ *sealing through E, countable metric-name exposure through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is the displayed packet of Definition 8599.1. Every row read by a consumer is one of $S, R, D, E, M, F, H, C, P, N$.

The Cauchy side is finite because each request first passes through the prefix schedule F and then through the stream windows S. The regular rational row R and dyadic ledger D expose only the readback and tolerance data admitted by those windows.

The completion-facing side is the real seal E together with the metric-name row M. Transport, replay, provenance, and local naming are confined to H, C, P, N, so the NameCert obligations are exhausted by the displayed bridge rows. □

Theorem 8599.3 (Constructive Cauchy real completion bridge Cauchy seal)**.** *Every accepted* ConstructiveCauchyRealCompletionBridge$^{\uparrow}$ *read reaches a completion-facing* Real$^{\uparrow}$ *seal only through the finite route*

$$F \longmapsto S \longmapsto R \longmapsto D \longmapsto E$$

together with the countable metric-name row M and the structural rows H, C, P, N.

Proof. Start with the finite-prefix row F. It bounds the request before the bridge can inspect a Cauchy-name window, so the stream row S is never consumed as an unbounded choice of observations.

The regular rational row R reads the displayed windows, and the dyadic ledger D records the tolerance scale for the comparison. Only after these rows are displayed does the packet expose the completion-facing real seal E.

The metric-name row M is read beside the same route, not in place of it. The structural rows H, C, P, N preserve and replay the displayed route without adding a selected limit or quotient representative. □

Theorem 8599.4 (Constructive Cauchy real completion bridge nonescape)**.** *An accepted* ConstructiveCauchyRealCo *packet exports only the displayed rows $S, R, D, E, M, F, H, C, P, N$. Any selected limit, quotient stream equality, countable-choice schedule, trichotomy principle, hidden representative, host real equality, or ambient* Real$^{\uparrow}$ *completeness claim must be supplied by a later certificate; it is not a coordinate of this bridge.*

Proof. Inspect the carrier coordinates. The packet contains finite stream windows, regular rational readback, dyadic tolerances, a real seal, countable metric-name data, finite prefixes, transport, replay, provenance, and local naming.

None of those coordinates is a selector for a Cauchy-stream equivalence class, a choice function over requests, a trichotomy oracle, or a completeness theorem. The real row E is a displayed seal reached by the route of Definition 8599.3, not an ambient completeness principle.

Therefore every legal consumer remains inside the named packet. A downstream completion theorem may cite the bridge as input, but it must display its own completion or equality certificate. □

Theorem 8599.5 (Constructive Cauchy real completion bridge finite prefix)**.** *For an accepted* $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ *carrier* $B = (S, R, D, E, M, F, H, C, P, N)$*, every bridge request must first expose the finite* $\mathsf{FiniteStream}^{\uparrow}$ *row* F *before it may read a* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{CountableMetricName}^{\uparrow}$ *row, or a completion-facing* $\mathsf{Real}^{\uparrow}$ *seal. The prefix row is therefore part of carrier admission, not a detachable optimization.*

Proof. Project the carrier rows. The row F is the only coordinate that bounds request prefixes before a Cauchy-name or metric-name read is consumed.

Once F is displayed, the stream row S and metric-name row M may be read through the structural rows H, C, P, N. Without F, the request would have no finite BHist boundary and could not be replayed by Cont.

Thus finite-prefix routing is a named obligation of the packet. It is not supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{CountableMetricCompletion}^{\uparrow}$ alone. □

Theorem 8599.6 (Constructive Cauchy real completion bridge stream window)**.** *Every accepted Cauchy-name read in* $B = (S, R, D, E, M, F, H, C, P, N)$ *passes from the finite prefix row* F *into the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row* S *before regular rational readback through* R*. No consumer may replace* S *by an unbounded stream, a hidden representative, or a host sequence.*

Proof. The carrier definition names S as the finite stream-window schedule for Cauchy observations. It is downstream from F and upstream from R.

Readback through $\mathsf{RegSeqRat}^{\uparrow}$ is therefore a projection of displayed stream windows, not a direct call to an ambient sequence object. Transport and replay keep the same F, S, R route visible through H and C.

Consequently the bridge exposes windowed stream data only. Any unbounded stream or hidden representative would be outside the named $\mathsf{NameCert}$ surface. □

Theorem 8599.7 (Constructive Cauchy real completion bridge dyadic tolerance)**.** *For every accepted bridge read, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *must display the tolerance scale, slack witness, and refusal data used to compare the regular rational readback* R *with the completion-facing seal* E*. A read that omits* D *is not an accepted* $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ *comparison.*

Proof. The row D is the packet's tolerance ledger. It records comparison scales after R has read the displayed stream windows and before E is consumed.

The dyadic row also contains the refusal data for comparisons that the bridge cannot certify. This keeps tolerance checking within $\mathsf{DyadicRatCore}^{\uparrow}$ instead of moving it to host real equality or a trichotomy oracle.

Therefore a valid comparison route must expose R, D, E in that order, with H, C, P, N preserving the same local certificate rows. □

Theorem 8599.8 (Constructive Cauchy real completion bridge real seal)**.** *The* $\mathsf{Real}^{\uparrow}$ *row* E *of an accepted* $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ *packet is reached only after the finite-prefix, stream-window, regular rational, and dyadic-tolerance rows* F, S, R, D *have been displayed. The row* E *is a completion-facing seal for that route, not a proof of ambient real completeness.*

Proof. Use the carrier route recorded in Definition 8599.3. The row E occurs after F, S, R, D and is transported with the same componentwise hsame row.

Because E is a displayed packet row, continuation replay can revisit the route through C and provenance can name it through P. The local $\mathsf{NameCert}$ row N then records the seal as part of this bridge.

No coordinate of E selects limits, identifies quotient streams, or supplies global completeness. Those claims require a separate certificate beyond the bridge packet. □

Theorem 8599.9 (Constructive Cauchy real completion bridge countable metric)**.** *The countable metric-name row* M *is exposed beside the Cauchy route* F, S, R, D, E *and before any consumer may treat the packet as input to* $\mathsf{CountableMetricCompletion}^{\uparrow}$*. The bridge names metric-read data; it does not itself certify countable metric completion.*

Proof. Project M from the carrier. It is the $\mathsf{CountableMetricName}^{\uparrow}$ row read by the bridge, while $\mathsf{CountableMetricCompletion}^{\uparrow}$ appears in provenance as a downstream completion route.

The metric-name row is transported and replayed with H and C, so consumers can inspect the finite route by which metric data was made available. This is weaker than constructing a completed metric object.

Thus M supplies countable metric-name exposure only. Completion existence, quotient classes, and selected representatives remain outside the local bridge coordinates. □

Theorem 8599.10 (Constructive Cauchy real completion bridge transport and replay)**.** *For accepted packets* $B = (S, R, D, E, M, F, H, C, P, N)$*, the componentwise* hsame *row* H *and displayed* Cont *row* C *preserve the route* F, S, R, D, E, M *together with provenance* P *and local naming* N*. Transport or replay cannot introduce a selected limit, quotient stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read H first. Its componentwise domain is the displayed stream, rational, dyadic, real-seal, metric-name, finite-prefix, continuation, provenance, and local naming rows.

Read C next. Continuation replay follows a bridge request through the same rows $F, S, R, D, E, M, H, P, N$, so replay is a finite route over BHist data rather than a new analytic principle.

The provenance row P and local $\mathsf{NameCert}$ row N record the sources and name of that route. Since no listed coordinate contains selected limits, quotient stream equality, choice schedules, or ambient completeness, transport and replay cannot export them. □

Falsifiable prediction: Every accepted ConstructiveCauchyRealCompletionBridgeUp read displays $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CountableMetricName}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the completion-facing seal is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real, countable-metric-name, countable-metric-completion, finite-stream: $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite-prefix schedule, Cauchy stream windows, rational readback, dyadic tolerance, real seal, countable metric-name row, transport, replay, provenance, and local naming in one bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$ packet is generated from finite prefix schedules, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, Real seals, CountableMetricName rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8599.1, Definition 8599.2, Definition 8599.3, and Definition 8599.4 seed the finite bridge from Cauchy-name data to completion-facing Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable choice, trichotomy, hidden representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix routing, stream-window exposure, regular rational readback, dyadic tolerance checking, Real sealing, countable metric-name exposure, transport, replay, provenance, and local naming for $\mathsf{ConstructiveCauchyRealCompletionBridge}^{\uparrow}$.

Chapter 8600

A Concrete Naming Certificate for $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$

Following Brouwer on spreads and choice sequences, the $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ packet records a finite prefix law whose accepted nodes may be inspected by L10 Real and Completion consumers without quotient streams, selected branches, or countable choice. It sits beside the finite observation surface of chapter 5916, the StreamName window discipline of chapter 126, and the List prefix carrier of chapter 36. The packet isolates the constructive law: a displayed finite prefix is admitted only through a finite BHist row, and every public read stays inside the same prefix-closed route.

Definition 8600.1 (Brouwerian spread-law carrier)**.** *A* $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, A, E, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FiniteStream}^{\uparrow}$ *observation row, listing the finite source windows currently inspected.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that names the requested window positions.*
3. *A is the* $\mathsf{List}^{\uparrow}$ *prefix row, storing the finite words and parent-prefix inclusions read by the law.*
4. *E is the admissibility row: every displayed prefix is either admitted by the law or remains on the refusal side of the same finite ledger.*
5. *W is the extension row, recording for each admitted prefix the displayed one-step or bounded finite continuation still admitted by E.*
6. *H records componentwise* hsame *transport for finite streams, schedules, prefixes, admissibility rows, extension rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a prefix request through F, S, A, E, W.*
8. *P is* Pkg *provenance over* $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerianSpreadLaw}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, S, A, E, W, H, C, P, N$. It has no coordinate for a completed infinite branch, quotient stream equality, countable-choice selector, fan compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8600.2 (Brouwerian spread-law NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ *carrier* $L = (F, S, A, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BrouwerianSpreadLaw}^{\uparrow}}$ *surface consists exactly of finite-stream exposure through* F*, schedule exposure through* S*, List prefix exposure through* A*, admissibility classification through* E*, extension visibility through* W*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every public read of the law factors through the finite route*

$$F, S, A \longmapsto E \longmapsto W \longmapsto H, C, P, N.$$

Proof. Project the carrier of Definition 8600.1. The only source rows available to a consumer are the finite stream windows, StreamName schedule, and List prefix ledger.

Read the admissibility and extension rows next. The row E classifies displayed prefixes in the finite ledger, while W records only continuations still admitted by that same displayed law.

Finally use H, C, P, N to preserve and name the route. Since no carrier coordinate stores a completed branch, quotient stream, countable-choice selector, fan compactness witness, or ambient completion principle, these rows exhaust the NameCert obligations. □

Theorem 8600.3 (Brouwerian spread-law prefix closure)**.** *Let* $L = (F, S, A, E, W, H, C, P, N)$ *be an accepted* $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ *carrier. If a consumer reads an admitted prefix from* A *and* E*, then every displayed predecessor prefix in the same List spine is read through the same finite stream row* F*, schedule row* S*, admissibility row* E*, extension row* W*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*.*

Proof. Start with the List prefix row A. It stores the displayed parent-prefix inclusions for the finite words currently visible to the law.

Use the admissibility row E. An admitted prefix can be read only as part of the same finite ledger, so predecessor prefixes are classified by the same law rather than by a second hidden predicate.

The finite-stream and StreamName rows keep the observation schedule fixed. The extension row W then witnesses only continuations whose predecessors remain in the displayed List spine.

Transport, replay, provenance, and local naming through H, C, P, N preserve that packet. Therefore prefix restriction stays inside the accepted $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ carrier and exports no selected infinite branch. □

Theorem 8600.4 (Brouwerian spread-law finite-window handoff)**.** *Every L10 consumer of an accepted* $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ *packet reaches finite observation data only through the displayed route*

$$\mathsf{FiniteStream}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{List}^{\uparrow} \longmapsto \mathsf{BrouwerianSpreadLaw}^{\uparrow}.$$

The handoff supplies prefix-law evidence for finite windows; it does not select an infinite choice sequence, identify quotient streams, or assert compactness or completeness.

Proof. Apply the NameCert obligations of Definition 8600.2. They force the consumer to expose F, S, A, E, W before any public read of the law is available.

Apply prefix closure from Definition 8600.3. A later finite window cannot bypass the predecessor-prefix route, because the same List spine and admissibility row remain visible.

The excluded objects are not carrier coordinates. Thus the L10 handoff is a finite prefix-law route from finite streams and StreamName windows into a displayed Brouwerian law, with no quotient stream, selected branch, countable choice, fan compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerianSpreadLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerianSpreadLaw}^{\uparrow}$ packet is generated from finite stream windows, StreamName schedules, List prefix rows, admissibility and extension rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerianSpreadLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8600.1, Definition 8600.2, Definition 8600.3, and Definition 8600.4 seed the constructive finite-prefix law surface over FiniteStreamUp, StreamNameUp, ListUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerianSpreadLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, quotient stream equality, countable choice, fan compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, prefix admissibility, extension visibility, prefix closure, structural nonescape, and L10 handoff coverage for BEDC.Derived.BrouwerianSpreadLawUp.

Chapter 8601

A Concrete Naming Certificate for $\mathsf{PartiallyOrderedGroup}^{\uparrow}$

Following Bourbaki on ordered algebraic structures and Fuchs on partially ordered groups, the $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ packet records the finite NameCert surface for a group equipped with a translation-invariant partial order. It sits between the group carrier of chapter 41, the order surface of chapter 1378, the poset carrier of chapter 53, the monoid source of chapter 40, and the local NameCert transport rows used throughout the concrete-instance layer. The packet names order-compatible multiplication and inversion data; it does not assert totality, lattice operations, Archimedean comparison, ordered completion, or host equality of group functions.

Definition 8601.1 (Partially ordered group carrier)**.** *A* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, O, P, M, I, K, H, C, Q, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Group}^{\uparrow}$ *row, naming multiplication, inverse, neutral element, and the finite group-law surface.*
2. *O is the* $\mathsf{Order}^{\uparrow}$ *row on the same carrier, recording reflexive, antisymmetric, and transitive comparison data.*
3. *P is the* $\mathsf{Poset}^{\uparrow}$ *readback row, exposing the order as a partial-order classifier rather than a preorder-only relation.*
4. *M is the* $\mathsf{Monoid}^{\uparrow}$ *source row for the multiplication law before inverses are inspected.*
5. *I records order transport under left and right multiplication by displayed group elements.*
6. *K records positive-cone data: neutral membership, multiplication closure, inverse reflection, and reconstruction of the order from the cone.*
7. *H records componentwise* hsame *transport for A, O, P, M, I, K and the structural rows below.*
8. *C records displayed* Cont *replay from group-law reads to order reads, translation reads, positive-cone reads, provenance, and local naming.*
9. *Q is the* Pkg *provenance over* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PartiallyOrderedGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, O, P, M, I, K, H, C, Q, N$. It has no coordinate for total order, lattice meet or join, Archimedean law, ordered completion, quotient representative, or host-level group function equality.

Theorem 8601.2 (Partially ordered group NameCert obligations)**.** *For every accepted* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ *carrier $G = (A, O, P, M, I, K, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{PartiallyOrderedGroup}^{\uparrow}}$ *surface consists exactly of carrier admission for G, group exposure through A, order exposure through O, poset readback through P, monoid multiplication exposure through M, translation-invariance accountability through I, positive-cone accountability through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, local naming through N, and nonescape excluding totality, lattice operations, Archimedean comparison, ordered completion, quotient representatives, and host group-function equality.*

Proof. Carrier admission is projection from Definition 8601.1. The algebraic rows are A and M, while the order rows are O and P.

The compatibility rows are separate. Translation invariance is read from I, and positive-cone stability is read from K; neither row is forced by merely naming a group or a poset.

Finally H, C, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface and block the excluded stronger structures. □

Theorem 8601.3 (Partially ordered group positive cone stability)**.** *In an accepted* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ *packet, the positive-cone row K is stable under multiplication and reconstructs the displayed order row O only through the group row A, monoid row M, and poset row P. A consumer may read positive elements and cone multiplication, but not total comparison, lattice operations, or completion data.*

Proof. Read K as a displayed carrier row. It records neutral membership, multiplication closure, inverse reflection, and reconstruction of the order from the cone.

The reconstruction is not detached from algebra. It uses the group operation from A, the multiplication law from M, and the partial-order classifier exposed through O and P.

Thus cone stability remains a finite positive-cone obligation inside the packet. It does not add totality, meets, joins, Archimedean comparison, or completion rows. □

Theorem 8601.4 (Partially ordered group order translation)**.** *Every accepted order comparison in a* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ *packet is stable under displayed left and right translation only through the row I together with A, O, P, M, H, C, Q, N. Translation stability is part of the NameCert route and is not a host-level equality of group functions.*

Proof. Start with a comparison in O and its partial-order readback in P. The group and monoid rows A, M provide the displayed multiplication data used for translation.

The row I records the actual compatibility obligation: translating comparable elements on the left or right preserves the displayed order relation. Transport and replay keep this route visible through H and C.

Provenance and local naming record the route as part of the $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ certificate. No coordinate identifies host functions or supplies comparison data outside the finite packet. □

Theorem 8601.5 (Partially ordered group nonescape)**.** *An accepted* $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ *packet exports only the displayed rows $A, O, P, M, I, K, H, C, Q, N$. Any total order, lattice-ordered group structure, Archimedean principle, ordered completion theorem, quotient representative, or host equality of group operations must be supplied by another certificate.*

Proof. Inspect the carrier inventory. It contains group data, order data, poset readback, monoid multiplication, translation accountability, positive-cone accountability, transport, replay, provenance, and local naming.

None of these rows is a total-order selector, a lattice operation, an Archimedean comparison principle, a completion theorem, a quotient representative, or a host function equality. The translation and cone rows are compatibility obligations internal to the finite packet.

Therefore every legal consumer remains inside $A, O, P, M, I, K, H, C, Q, N$. Stronger ordered-algebra or completion claims require their own displayed NameCert rows. □

Falsifiable prediction: Every accepted $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ consumer displays the GroupUp row, OrderUp row, PosetUp readback, MonoidUp multiplication row, order-translation row, positive-cone row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before using order-compatible group structure.

Independence witness: group, order, poset, monoid: none is bijective because $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ jointly carries group laws, partial-order readback, monoid multiplication, translation-invariance accountability, positive-cone stability, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PartiallyOrderedGroup}^{\uparrow}}$.

Constructive story: A $\mathsf{PartiallyOrderedGroup}^{\uparrow}$ packet is generated from a GroupUp row, an OrderUp row, a PosetUp readback row, a MonoidUp multiplication row, translation-invariance accountability, positive-cone accountability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartiallyOrderedGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8601.1, Definition 8601.2, Definition 8601.3, Definition 8601.4, and Definition 8601.5 seed the finite order-compatible group certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartiallyOrderedGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close total orders, lattice-ordered groups, Archimedean comparison, ordered completions, quotient representatives, host group-function equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, group exposure, order exposure, poset readback, monoid multiplication exposure, translation-invariance accountability, positive-cone accountability, transport, replay, provenance, and local naming for BEDC.Derived.PartiallyOrderedGroupUp.

Chapter 8602

A Concrete Naming Certificate for $\mathsf{ModulusCauchyCompression}^{\uparrow}$

Following Bishop and Bridges on moduli for Cauchy constructions, this packet records a finite BEDC compression of regular-Cauchy tail schedules. It sits beside the stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38. The chapter names a displayed compression ledger for finite requests; it does not claim selected limits, quotient stream equality, countable choice, or ambient completeness.

Definition 8602.1 (Modulus Cauchy compression carrier)**.** *A* $\mathsf{ModulusCauchyCompression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request budget row for the compressed modulus surface.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that keeps the dyadic bounds visible.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows.*
5. *H records componentwise* hsame *transport for Q, D, S, R, C, P, N.*
6. *C records displayed* Cont *replay from the request budget through the window row, dyadic ledger, and regular rational readback.*
7. *P is the* Pkg *provenance over* $\mathsf{ModulusCauchyCompression}^{\uparrow}$, BHist, hsame, Cont, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompression}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through Q, D, S, R, H, C, P, N. It has no coordinate for selected limits, quotient stream equality, countable choice, ambient completeness, or a host real-space seal.

Theorem 8602.2 (Modulus Cauchy compression classifier transport)**.** *If two accepted* $\mathsf{ModulusCauchyCompression}^{\uparrow}$ *carriers agree row by row up to* hsame*, then the displayed classifier sees the same request budget, dyadic tolerance ledger, stream window, regular rational readback, replay, provenance, and local naming. The packet therefore transports only a finite NameCert surface inside* BHist.

Proof. Project the carrier of Definition 8602.1. The visible data are exactly the request budget row Q, the dyadic ledger D, the stream window S, the regular rational readback R, and the structural rows H, C, P, N.

Because hsame transport is recorded componentwise, the classifier cannot distinguish two packets that agree on those rows. The packet has no separate coordinate for a selected limit or a quotient stream, so the visible route is stable under transport. □

Theorem 8602.3 (Modulus Cauchy compression tail stability)**.** *Every accepted* $\mathsf{ModulusCauchyCompression}^{\uparrow}$ *carrier exposes its tail schedule through the same* $\mathsf{StreamName}^{\uparrow}$ *row before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed. The finite request budget Q therefore reaches the regular rational surface only after the dyadic and window rows have been displayed.*

Proof. The carrier definition places the request budget Q beside the dyadic ledger D and the stream window S. The readback row R is downstream of that displayed window data, so a consumer must read the finite tail schedule before it can read a regular rational output.

This is a stability statement about the route, not a completeness claim. The packet keeps the schedule finite, the window explicit, and the readback routed through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 8602.4 (Modulus Cauchy compression ledger exactness)**.** *The* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger in an accepted* $\mathsf{ModulusCauchyCompression}^{\uparrow}$ *carrier records exactly the tolerance data needed for the compressed request budget. No additional dyadic witness, hidden choice row, or quotient equality row is available to the ledger.*

Proof. Inspect the definition. The row D is the unique tolerance ledger, and it is read together with the request budget Q and window row S.

Since the packet has only the rows listed in Definition 8602.1, the ledger cannot be supplemented by a hidden choice function or by a quotient stream equality. The exactness claim is only that the displayed dyadic data are the whole local ledger. □

Theorem 8602.5 (Modulus Cauchy compression RealUp handoff)**.** *An accepted* $\mathsf{ModulusCauchyCompression}^{\uparrow}$ *packet hands its displayed route to* $\mathsf{Real}^{\uparrow}$ *consumers only through the finite chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff does not add ambient completeness, selected limits, or bridge checking.

Proof. The carrier definition places the real-facing consumption downstream of the dyadic ledger, the stream windows, and the regular rational readback. A $\mathsf{Real}^{\uparrow}$ consumer therefore sees the same finite route before any endpoint seal is exposed.

Because the packet is a finite BHist object with explicit hsame, Cont, Pkg, and NameCert rows, the handoff remains local. The chapter does not claim ambient completeness or bridge checking; it only records the displayed route into the real-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompression}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ModulusCauchyCompression}^{\uparrow}$ packet is generated from a finite request budget, a DyadicRatCore tolerance ledger, a StreamName window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompression}^{\uparrow}}$, seed)

Scope closed: Definition 8602.1, Definition 8602.2, Definition 8602.3, Definition 8602.4, and Definition 8602.5 seed the finite compression route from regular-Cauchy tails to RegSeqRat and RealUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompression}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.ModulusCauchyCompressionUp`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, ambient completeness, or bridge checking.

Upgrade path: The next grade requires checked carrier admission, exact tail compression, and a verified RealUp handoff for ModulusCauchyCompression$^{\uparrow}$.

Chapter 8603

A Concrete Naming Certificate for $\mathsf{FiniteElementMethod}^{\uparrow}$

Following Ciarlet's finite element method, the $\mathsf{FiniteElementMethod}^{\uparrow}$ packet records the finite mesh, local stiffness, assembly ledger, boundary row, and solution-readback surface that a constructive PDE-discretization consumer must inspect before it treats a variational problem as a computable finite system. It sits beside the PDE carrier of chapter 194, the matrix source of chapter 49, the linear-map source of chapter 48, and the vector-space source of chapter 47. The packet names only displayed finite element data; it does not assert analytic existence, host function equality, mesh convergence, hidden quadrature exactness, selected basis functions outside the ledger, or ambient completeness.

Definition 8603.1 (Finite element method carrier)**.** *A* $\mathsf{FiniteElementMethod}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, V, L, A, B, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite mesh row, naming cells, face incidences, boundary tags, and refinement provenance visible to the packet;*
2. *V is the local finite-element space row, naming the cellwise degrees of freedom, shape-function table, and local basis readback;*
3. *L is the local stiffness row, storing the finitely inspected bilinear-form entries on each cell;*
4. *A is the assembly ledger row, recording how local stiffness entries are routed into the global* $\mathsf{Mat}^{\uparrow}$ *row;*
5. *B is the boundary and load row, recording boundary constraints and source terms as displayed finite data;*
6. *R is the* $\mathsf{LinMap}^{\uparrow}$ *readback row associated with the assembled matrix and load data;*
7. *S is the solve-facing row, exposing the finite system interface without claiming an analytic PDE solution;*
8. *H records componentwise* hsame *transport for mesh, local space, stiffness, assembly, boundary, readback, solve-facing, provenance, and local naming rows;*
9. *C records displayed* Cont *replay from mesh inspection through local stiffness, assembly, readback, and solve-facing consumption;*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteElementMethod}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteElementMethod}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, L, A, B, R, S, H, C, P, N$. It has no coordinate for host finite-element spaces, selected global basis functions outside V, analytic weak solutions, exact integration not displayed by L, quotient mesh equality, or convergence of a mesh family.

Theorem 8603.2 (Finite element method NameCert obligations)**.** *For every accepted* $\mathsf{FiniteElementMethod}^{\uparrow}$ *carrier*

$$F = (M, V, L, A, B, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteElementMethod}^{\uparrow}}$ *surface consists exactly of carrier admission for finite meshes, local finite-element spaces, local stiffness tables, assembly ledgers, boundary and load rows, linear-map readback, solve-facing exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any PDE, matrix, or linear-map consumer of the packet must read through those rows before it cites a finite element method boundary.*

Proof. Project the carrier of Definition 8603.1. The mesh row M, local-space row V, stiffness row L, assembly row A, boundary row B, readback row R, and solve-facing row S are the only numerical-analysis coordinates.

The $\mathsf{Pde}^{\uparrow}$ sibling supplies the displayed relation-and-boundary discipline, while the $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ siblings supply the finite algebraic readback. The packet therefore exposes the finite element route as data moving from local cells to an assembled matrix, not as a host theorem about weak solutions.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for hidden basis selection, analytic existence, unstated quadrature exactness, quotient mesh equality, or mesh-family convergence, the listed rows exhaust the local NameCert surface. □

Theorem 8603.3 (Finite element method local stiffness assembly)**.** *Every assembled algebraic read exported by an accepted* $\mathsf{FiniteElementMethod}^{\uparrow}$ *packet factors through the finite route*

$$M, V, L \longmapsto A \longmapsto R.$$

The local stiffness row is inspected cell by cell, the assembly ledger records each displayed placement into the global matrix row, and the linear-map readback consumes only the assembled rows already visible in the packet. No consumer may replace this route by host integration, an unstated global basis, a quotient mesh comparison, or an analytic PDE solution.

Proof. Start with the accepted carrier $F = (M, V, L, A, B, R, S, H, C, P, N)$. The local stiffness row L is meaningful only over the finite mesh M and local-space row V.

The assembly ledger A records the finite placement of each local entry into the matrix-facing surface. Thus the $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ reads consume A and R rather than a hidden host assembly operation.

The structural rows H, C, P, N keep the same route under transport, replay, provenance, and local naming. Hence the assembled algebraic read is exactly the displayed finite route from mesh and local data to the assembled matrix readback. □

Theorem 8603.4 (Finite element method nonescape)**.** *Every accepted* $\mathsf{FiniteElementMethod}^{\uparrow}$ *consumer must keep the mesh row, local finite-element space row, local stiffness table, assembly ledger, boundary and load row, linear-map readback, solve-facing row, transport, replay, provenance, and local naming visible before citing the packet as finite element method data. The packet exports no analytic solution theorem, convergence theorem, host weak-form equality, selected global basis outside the displayed row, hidden quadrature, or ambient completeness principle.*

Proof. Apply the NameCert obligations of Definition 8603.2. They expose the finite mesh, local space, stiffness, assembly, boundary, readback, solve-facing, transport, replay, provenance, and naming rows.

Then use the local assembly theorem Definition 8603.3. It forces any algebraic consumer through the displayed mesh-to-matrix route before the finite system interface is read.

The excluded analytic and host-level objects are absent from the carrier. Therefore no downstream consumer can cite this packet as analytic PDE solvability, convergence, unstated basis selection, hidden integration exactness, quotient mesh equality, or ambient completion. □

Falsifiable prediction: Every accepted $\mathsf{FiniteElementMethod}^{\uparrow}$ finite-system read exposes mesh, local finite-element space, local stiffness, assembly, boundary and load, linear-map readback, solve-facing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the finite element method boundary is consumed.

Independence witness: pde, matrix, linearmap, vecspace: $\mathsf{FiniteElementMethod}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite mesh, local finite-element spaces, local stiffness tables, assembly ledgers, boundary rows, linear-map readback, solve-facing exposure, transport, replay, provenance, and local naming as one discretization packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteElementMethod}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteElementMethod}^{\uparrow}$ packet is generated from a finite mesh row, local finite-element space row, local stiffness table, assembly ledger, boundary and load row, LinearMapUp readback, solve-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteElementMethod}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8603.1, Definition 8603.2, Definition 8603.3, and Definition 8603.4 seed the finite mesh, local stiffness, assembly, boundary, algebraic readback, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteElementMethod}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim analytic PDE solvability, weak-solution uniqueness, mesh convergence, hidden quadrature exactness, quotient mesh equality, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh exposure, local-space exposure, stiffness-table exactness, assembly-ledger routing, boundary and load readback, linear-map handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteElementMethodUp.

Chapter 8604

A Concrete Naming Certificate for MetaCICNormalizationRedexFrontier$^{\uparrow}$

The MetaCICNormalizationRedexFrontier$^{\uparrow}$ packet realizes the candidate-mediated route of chapter 9383 as finite NameCert data. It sits beside the critical-path source of chapter 765, the normalization frontier of chapter 892, the parallel-diamond frontier of chapter 4238, and the typed example coverage matrix of chapter 923. The packet records typed terms, candidate evidence, residual budgets, beta windows, strong-normalization handoff, transport, replay, provenance, and local naming without claiming unconditional normalization for every closed MetaCIC term.

Definition 8604.1 (MetaCIC normalization redex-frontier carrier)**.** *A* MetaCICNormalizationRedexFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, C, E, B, W, S, D, H, Q, P, N)$$

with the following visible rows:

1. *T is the typed-term frontier row, naming the closed term, type, context boundary, and visible redex windows;*
2. *C is the candidate-evidence row consumed by the candidate-mediated strong-normalization route;*
3. *E is the candidate-set or closed-candidate evidence row, when the route is exposed through a packaged candidate surface;*
4. *B is the residual-budget row, bounding the redex windows inspected before the strong-normalization handoff is cited;*
5. *W is the beta-window row, recording the finite beta steps and local residual reads relevant to the frontier;*
6. *S is the strong-normalization handoff row, readable only after T, C, E, B, W have been displayed;*
7. *D is the typed-example coverage row, showing which example or coverage matrix row witnesses the packet when a concrete term is used;*
8. *H records componentwise* hsame *transport for typed-term, candidate, evidence, budget, beta-window, handoff, coverage, provenance, and local naming rows;*
9. *Q records displayed* Cont *replay from typed-term inspection through candidate evidence, redex budgeting, beta-window reading, and strong-normalization handoff;*

10. *P is the* Pkg *provenance over* MetaCICNormalizationRedexFrontier$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalization MetaCICParallelDiamondFrontier$^{\uparrow}$, TypedExampleCoverageMatrix$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{MetaCICNormalizationRedexFrontier}^{\uparrow}}$ *row.*

The carrier has no coordinate for global closed-term normalization without candidate evidence, unrestricted subject reduction, full confluence, quotient syntax equality, evaluator equality, or kernel consistency.

Theorem 8604.2 (MetaCIC normalization redex-frontier NameCert obligations)**.** *For every accepted* MetaCICNormalizationRedexFrontier$^{\uparrow}$ *carrier*

$$R = (T, C, E, B, W, S, D, H, Q, P, N),$$

the local NameCert$_{\text{MetaCICNormalizationRedexFrontier}^{\uparrow}}$ *surface consists exactly of carrier admission for typed-term frontiers, candidate evidence, packaged candidate-set or closed-candidate evidence, residual budgets, beta windows, strong-normalization handoff, typed-example coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8604.1. The typed frontier, candidate evidence, packaged evidence, residual budget, beta window, strong-normalization handoff, and coverage row are the only MetaCIC coordinates.

The audit map of chapter 9383 requires the candidate layer to remain visible. Rows C and E record that layer, while B and W keep the inspected redex frontier finite before S is read.

The sibling rows place the frontier in the existing MetaCIC lattice: critical path for the proof route, normalization frontier for the candidate handoff, parallel-diamond frontier for confluence-facing interaction, and typed example coverage for concrete witnesses. Since the carrier has no coordinate for unconditional normalization, subject reduction, full confluence, quotient syntax equality, evaluator equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 8604.3 (MetaCIC normalization redex-frontier candidate handoff)**.** *Every strong-normalization read exported by an accepted* MetaCICNormalizationRedexFrontier$^{\uparrow}$ *packet factors through*

$$T, C, E, B, W \longmapsto S.$$

The typed term frontier is inspected first, the candidate evidence and packaged evidence remain displayed, the residual budget and beta window bound the redex read, and only then may the strong-normalization handoff row be consumed.

Proof. Start with the accepted carrier $R = (T, C, E, B, W, S, D, H, Q, P, N)$. The row S is downstream of the typed-term, candidate, packaged-evidence, budget, and beta-window rows.

The candidate-mediated route prevents a consumer from erasing C or E from the citation. The residual budget and beta window keep the redex frontier finite, so the route is a displayed packet rather than a global evaluator theorem.

Transport, replay, provenance, and local naming preserve the same dependency order through H, Q, P, N. Therefore every strong-normalization handoff read factors through T, C, E, B, W before S is available. □

Theorem 8604.4 (MetaCIC normalization redex-frontier nonescape)**.** *Every accepted* MetaCICNormalizationRedex *consumer must keep the typed-term frontier, candidate evidence, packaged evidence, residual budget, beta window, strong-normalization handoff, coverage row, transport, replay, provenance, and*

local naming visible before citing the packet as normalization evidence. It cannot cite the packet as unconditional closed-term normalization, subject reduction, full confluence, quotient syntax equality, evaluator equality, or kernel consistency.

Proof. Use the NameCert obligations of Definition 8604.2. They expose the typed-term, candidate, evidence, budget, beta-window, handoff, coverage, transport, replay, provenance, and naming rows.

Then apply the candidate handoff theorem Definition 8604.3. It forces the strong-normalization read through the candidate-mediated redex frontier before any endpoint can be consumed.

The stronger metatheorems are absent from the carrier. Thus no consumer can replace this packet by global normalization, subject reduction, full confluence, quotient syntax equality, evaluator equality, or kernel consistency. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}$ normalization read exposes MetaCICCriticalPath, MetaCICNormalizationFrontier, MetaCICParallelDiamondFrontier, and TypedExampleCoverageMatrix rows together with typed-term, candidate, evidence, residual-budget, beta-window, strong-normalization handoff, transport, replay, provenance, and local naming data.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-parallel-diamond-frontier, typedexamplecoveragematrix: none is bijective to $\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}$ because this packet jointly carries typed-term redex windows, candidate evidence, packaged candidate rows, residual budgets, beta-window reads, strong-normalization handoff, coverage, transport, replay, provenance, and local naming. The Lean TasteGate surface must provide Nontrivial and FieldFaithful instances for the same carrier rows before this packet can be treated as a checked derived target.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}$ packet is generated from a typed-term frontier, candidate evidence, packaged candidate-set or closed-candidate evidence, residual budget, beta window, strong-normalization handoff, typed-example coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8604.1, Definition 8604.2, Definition 8604.3, and Definition 8604.4 seed the candidate-mediated redex-frontier route over typed terms, candidate evidence, residual budgets, beta windows, strong-normalization handoff, coverage, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationRedexFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unconditional closed-term normalization, subject reduction, full confluence, quotient syntax equality, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-term exposure, candidate-evidence exposure, packaged-evidence routing, residual-budget exactness, beta-window readback, strong-normalization handoff, typed-example coverage, transport, replay, provenance, local naming, Nontrivial TasteGate evidence, and FieldFaithful TasteGate evidence for BEDC.Derived.MetaCICNormalizationRedexFrontierUp.

Chapter 8605

Dyadic Barycentric Refinement NameCert Construction

Following Bishop and Bridges, the $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ packet records dyadic barycentric subdivision as finite listed cell data rather than as an open-cover compactness principle. It sits after the dyadic arithmetic core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback route of chapter 301, the terminal Real seal of chapter 38, and the finite interval-cell consumer of chapter 5966. The packet gives Real, Completion, and compact-interval arguments a displayed finite subdivision surface; it does not export open-cover compactness, selected subcovers, quotient streams, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8605.1 (Dyadic barycentric refinement carrier)**.** *A* $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, I, M, K, S, R, Q, H, C, P, N)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row. It records dyadic addition, halving, midpoint, successor, and radius comparisons used by the subdivision.*
2. *I is the parent-cell row consumed from* $\mathsf{DyadicIntervalNet}^{\uparrow}$*. It lists the displayed closed dyadic cells before refinement.*
3. *M is the midpoint row. For each listed parent cell it records the dyadic barycenter and the two displayed child endpoints produced by bisection.*
4. *K is the finite child-cell row. It lists the child cells, their parent pointers, adjacency witnesses, endpoint inheritance, and mesh-decrease witnesses.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed when a parent endpoint, barycenter, or child endpoint is read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic endpoints and the tolerance schedule induced by the child-cell radii.*
7. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed parent, midpoint, child-cell, window, and readback rows have been inspected.*
8. *H records componentwise* hsame *transport for D, I, M, K, S, R, Q and the structural rows.*

9. *C records displayed* Cont *replay of the finite route*

$$D, I, M, K \longmapsto S, R \longmapsto Q, H, P, N.$$

10. *P is the* Pkg *provenance over* DyadicBarycentricRefinement$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{DyadicBarycentricRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, M, K, S, R, Q, H, C, P, N$. It has no coordinate for an arbitrary open cover, a selected finite subcover, quotient stream equality, a chosen limit point, countable choice, trichotomy, or completed real compactness.

Theorem 8605.2 (Dyadic barycentric refinement NameCert obligations)**.** *For the carrier of Definition 8605.1, the local* NameCert$_{\text{DyadicBarycentricRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$B = (D, I, M, K, S, R, Q, H, C, P, N),$$

dyadic endpoint arithmetic through D, parent-cell exposure through I, barycenter exposure through M, child-cell listing through K, finite observation windows through S, regular rational readback through R, the terminal Real seal through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.

Proof. Project the carrier definition. The rows D, I, M, K are the only subdivision-facing data: endpoints and midpoint arithmetic live in D, M, while the parent and child finite-cell ledgers live in I, K.

The observation and Real-facing rows are downstream. Endpoint and barycenter reads pass through S, the induced dyadic tolerance schedule passes through R, and the terminal real-name boundary is exposed only by Q.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve exactly the displayed finite route, so the certificate exports subdivision data only through the listed rows. □

Theorem 8605.3 (Dyadic barycentric refinement RegSeqRat handoff)**.** *Every downstream consumer of an accepted* DyadicBarycentricRefinement$^{\uparrow}$ *carrier that inspects a parent endpoint, barycenter, child endpoint, or child-cell tolerance must read*

$$D, I, M, K \longmapsto S, R \longmapsto Q, H, C, P, N.$$

The DyadicRatCore$^{\uparrow}$ *row D supplies the dyadic arithmetic, the* StreamName$^{\uparrow}$ *row S exposes the finite endpoint windows, and the* RegSeqRat$^{\uparrow}$ *row R exposes the rational tolerance readback. Neither row may be skipped or replaced by quotient stream equality, selected limits, countable choice, or open-cover compactness.*

Proof. Begin with a downstream endpoint or tolerance read. The carrier gives the consumer only the parent cells in I, barycenters in M, child cells in K, and dyadic arithmetic in D before any named readback row is available.

The read then passes through S and R. The StreamName row supplies finite observation windows for the relevant endpoint names, while the RegSeqRat row supplies the regular rational schedule attached to the child-cell radii.

Finally replay Q, H, C, P, N. The Real seal is reached only after the displayed finite handoff, and the structural rows preserve that handoff componentwise, excluding quotient stream equality, selected limits, choice, and open-cover compactness. □

Theorem 8605.4 (Dyadic barycentric refinement Real seal non-escape)**.** *For every accepted* $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ *carrier* $B = (D, I, M, K, S, R, Q, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* Q *is reachable only after the finite subdivision route*

$$D, I, M, K \longmapsto S, R$$

has been displayed. Any Real, Completion, compact-interval, or $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that reaches* Q *remains inside the same routed packet* $D, I, M, K, S, R, Q, H, C, P, N$ *and cannot escape to host real equality, real trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected subcovers, quotient equality, selected limits, or countable choice.*

Proof. Project the accepted packet. The carrier places Q downstream of endpoint arithmetic, parent cells, barycenter rows, child-cell rows, StreamName windows, and RegSeqRat readback.

Thus a Real-facing consumer must first traverse D, I, M, K, S, R. These rows display the finite subdivision cells, endpoint reads, and tolerance schedule before any Real-facing boundary is available.

Once Q is reached, the structural rows H, C, P, N only preserve the same displayed route. Since no excluded equality, selection, trichotomy, choice, compactness, or completeness row is present in the carrier, the Real seal cannot be used as an escape from the finite dyadic barycentric refinement packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBarycentricRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ packet is generated from dyadic endpoint arithmetic, parent interval cells, barycenter rows, finite child cells, StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBarycentricRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8605.1, Definition 8605.2, Definition 8605.3, and Definition 8605.4 seed the finite dyadic barycentric refinement carrier over dyadic endpoints, parent cells, barycenters, child cells, StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBarycentricRefinement}^{\uparrow}}, \mathsf{formalTarget})$
Lean4 (statement) `BEDC.Derived.DyadicBarycentricRefinementUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, barycenter arithmetic, child-cell coverage, StreamName and RegSeqRat readback routing, Real seal ordering, provenance, and local naming for BEDC.Derived.DyadicBarycentricRefineme

Chapter 8606

A Concrete Naming Certificate for $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$

Following Atiyah and Bott, the $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$ packet records a fixed-point formula as a finite local-index and trace ledger over an already displayed endomap. It consumes the Lefschetz trace surface of $\mathsf{LefschetzFixedPoint}^{\uparrow}$ (chapter 6557), the local index accounting route of $\mathsf{FixedPointIndex}^{\uparrow}$ (chapter 7376), compact-source rows from $\mathsf{CompactMetric}^{\uparrow}$ (chapter 129), self-map rows from $\mathsf{ContinuousMap}^{\uparrow}$ (chapter 130), and homology handoff rows from $\mathsf{Homology}^{\uparrow}$ (chapter 98). The packet is not a new ambient fixed-point theorem; it is the local formula surface that lets downstream readers see exactly which fixed-locus, index-weight, and trace rows are being composed.

Definition 8606.1 (Atiyah–Bott fixed point formula carrier)**.** *An* $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, F, L, I, T, K, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact-source row, read through* $\mathsf{CompactMetric}^{\uparrow}$*, naming the inspected finite source surface and its compact-metric probes.*
2. *F is the continuous self-map row, read through* $\mathsf{ContinuousMap}^{\uparrow}$*, whose fixed-locus and induced homology action are being inspected.*
3. *L is the fixed-locus row, listing the displayed fixed components, point windows, or refusal rows that the formula is allowed to inspect.*
4. *I is the local index-weight ledger, read through* $\mathsf{FixedPointIndex}^{\uparrow}$*, assigning each displayed fixed-locus row its finite index or weight datum.*
5. *T is the trace row, read through* $\mathsf{LefschetzFixedPoint}^{\uparrow}$*, recording the finite alternating trace or Lefschetz accounting datum.*
6. *K is the homology handoff row, read through* $\mathsf{Homology}^{\uparrow}$*, exposing the induced homology action needed by the trace computation.*
7. *M is the formula-match row comparing the sum of displayed local index-weights with the trace ledger through* $\mathsf{Real}^{\uparrow}$ *or the finite scalar row already named by the dependencies.*
8. *H records componentwise* hsame *transport for compact source, self-map, fixed-locus, index-weight, trace, homology handoff, formula-match, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from X, F into L, K, from L into I, from K into T, and from I, T into M.*
10. *P is the* Pkg *provenance over* $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$, $\mathsf{LefschetzFixedPoint}^{\uparrow}$, $\mathsf{FixedPointIndex}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AtiyahBottFixedPoint}^{\uparrow}}$ *row.*

The carrier exports only these rows. It has no coordinate for transversality, smooth manifold structure, elliptic complexes, analytic index theory, host cohomology isomorphisms, or a global fixed-point existence theorem outside the displayed BEDC packets.

Theorem 8606.2 (Atiyah–Bott fixed point NameCert obligations)**.** *For an accepted* $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$ *carrier $A = (X, F, L, I, T, K, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AtiyahBottFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact source, continuous self-map, fixed-locus, local index-weight ledger, trace row, homology handoff, formula-match row, componentwise transport, displayed replay, provenance, and local naming rows. Every formula read must pass through* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *before the fixed-locus row is inspected, through* $\mathsf{FixedPointIndex}^{\uparrow}$ *before local weights are read, through* $\mathsf{Homology}^{\uparrow}$ *and* $\mathsf{LefschetzFixedPoint}^{\uparrow}$ *before the trace row is read, and through M before the local-index sum is compared with the trace ledger.*

Proof. Project the carrier of Definition 8606.1. Its rows already separate the compact source, self-map, fixed locus, index weights, trace ledger, homology handoff, and formula match.

Read the fixed-locus route first. The source row X and self-map row F expose the only fixed components that the packet may inspect, while L records whether those components are displayed or refused.

Read the two formula sides separately. The local side passes from L to the $\mathsf{FixedPointIndex}^{\uparrow}$ row I, and the trace side passes from the homology handoff K to the $\mathsf{LefschetzFixedPoint}^{\uparrow}$ trace row T.

Finally use M, H, C, P, N. The match row compares only the displayed local weights and trace ledger, while transport, replay, provenance, and naming preserve the same finite rows. No unlisted analytic or host topological datum can enter the local NameCert surface. □

Theorem 8606.3 (Atiyah–Bott fixed point index route)**.** *Every* $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$ *formula read factors through the route*

$$X, F \longmapsto L, \qquad L \longmapsto I, \qquad X, F, K \longmapsto T, \qquad I, T \longmapsto M.$$

Thus the local index-weight sum and the Lefschetz trace ledger meet only at the displayed formula-match row. A consumer cannot bypass the fixed-locus row, replace $\mathsf{FixedPointIndex}^{\uparrow}$ *with an ambient analytic index theorem, replace* $\mathsf{LefschetzFixedPoint}^{\uparrow}$ *with a host trace theorem, or infer global fixed-point existence from the local equality ledger.*

Proof. Begin with the compact-source and continuous self-map rows. They determine the fixed-locus row L as the only admissible source of local fixed components for this packet.

The local branch then reads I through $\mathsf{FixedPointIndex}^{\uparrow}$. This supplies the index-weight ledger for the displayed components and no analytic index object outside the row.

The trace branch reads K through $\mathsf{Homology}^{\uparrow}$ and then T through $\mathsf{LefschetzFixedPoint}^{\uparrow}$. This supplies the finite trace accounting datum and no host trace theorem outside the dependency packets.

At the final step, M compares the two displayed formula sides. Since every route into M has already passed through L, I, K, T, the consumer receives a local formula ledger rather than a global fixed-point theorem or an ambient Atiyah–Bott analytic package. □

Closure status for $\mathsf{NameCert}_{\mathsf{AtiyahBottFixedPoint}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AtiyahBottFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8606.1, Definition 8606.2, and Definition 8606.3 seed the local fixed-locus, index-weight, trace, homology handoff, transport, replay, provenance, and NameCert rows over $\mathsf{LefschetzFixedPoint}^{\uparrow}$, $\mathsf{FixedPointIndex}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{Homology}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AtiyahBottFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, transversality, smooth manifold structure, elliptic index theory, host cohomology isomorphisms, or global fixed-point existence.

Upgrade path: Obligation closure requires checked carrier admission, fixed-locus exposure, local index-weight exactness, trace and homology handoff, formula-match transport, replay, provenance, and no-host-theorem boundaries for BEDC.Derived.AtiyahBottFixedPointUp.

Constructive story: An $\mathsf{AtiyahBottFixedPoint}^{\uparrow}$ packet is generated from compact-source rows, a continuous self-map row, fixed-locus rows, local index-weight rows, Lefschetz trace rows, homology handoff rows, componentwise transport, displayed replay, provenance, and local NameCert data.

Chapter 8607

A Concrete Naming Certificate for MirandaTheoremFiniteCube$^{\uparrow}$

Following Miranda and the standard finite-dimensional intermediate-value tradition, the MirandaTheoremFiniteCube packet records the finite cube sign-change interface by which opposite-face sign data is read before any zero-facing consumer may act. It sits beside the labelled simplex route of chapter 2003, the finite obstruction ledger of chapter 3745, the compact carrier of chapter 60, the displayed map surface of chapter 130, and the terminal Real seal of chapter 38. The chapter names only finite cube, face-sign, compactness, continuity, degree, and readback rows; it does not assert a host zero theorem, selected root, quotient real equality, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8607.1 (Miranda theorem finite-cube carrier)**.** *A* MirandaTheoremFiniteCube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, F, S, C, D, L, W, R, E, H, T, P, N)$$

with the following displayed rows:

1. *Q is the finite cube row, listing the coordinate faces, opposite-face pairs, finite cell incidence table, and dyadic subdivision windows used by the certificate.*
2. *F is the face-sign row. For each coordinate it records the negative-face and positive-face sign reads of the displayed map.*
3. *S is the sign-stability row, preserving those face signs under the finite refinements, StreamName windows, and componentwise* hsame *transports used by the packet.*
4. *C is the* Compact$^{\uparrow}$ *source row for the named cube, read only as the displayed compact carrier required by the finite certificate.*
5. *D is the* BrouwerDegree$^{\uparrow}$ *obstruction row, consumed only as the finite boundary ledger attached to the sign-change data.*
6. *L is the* SpernerLemma$^{\uparrow}$ *labelled-cell handoff row, recording the finite labelled subdivision through which a sign conflict is localized.*
7. *W is the* StreamName$^{\uparrow}$ *finite-window row opened for coordinates, face probes, and retained cells.*
8. *R is the* RegSeqRat$^{\uparrow}$ *regular readback row attached to those windows and dyadic cell tolerances.*
9. *E is the terminal* Real$^{\uparrow}$ *seal row for the displayed candidate cell, available only after Q, F, S, C, D, L, W, R have been read.*

10. *H records componentwise* hsame *transport for all displayed rows.*
11. *T records displayed* Cont *replay through the* $\mathsf{ContinuousMap}^{\uparrow}$ *source surface.*
12. *P is the* Pkg *provenance over* $\mathsf{MirandaTheoremFiniteCube}^{\uparrow}$, $\mathsf{SpernerLemma}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MirandaTheoremFiniteCube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, S, C, D, L, W, R, E, H, T, P, N$. It has no coordinate for a selected zero, unrestricted host degree, quotient real equality, open-cover compactness, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Theorem 8607.2 (Miranda theorem finite-cube NameCert obligations)**.** *For every accepted* MirandaTheoremFiniteC *carrier*

$$M = (Q, F, S, C, D, L, W, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MirandaTheoremFiniteCube}^{\uparrow}}$ *surface consists exactly of finite cube admission through Q, opposite-face sign exposure through F, sign-stability through S, compact cube source exposure through C, Brouwer-degree boundary obstruction through D, Sperner labelled-cell handoff through L, finite-window readback through W, R, terminal Real sealing through E, componentwise transport through H, displayed continuous-map replay through T, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 8607.1. The rows Q, F, S display the finite cube, opposite-face sign data, and stability ledger that a Miranda-facing read can inspect.

Read the analytic source rows next. The compact row C gives only the named cube carrier, the continuous-map replay row T fixes how face probes are obtained from the displayed map, and the degree row D supplies only the finite boundary obstruction ledger.

The finite combinatorial handoff is L, W, R, E. The Sperner row localizes the sign conflict in a labelled subdivision, the window and regular-readback rows expose the retained cell data, and the Real seal is reached only after those rows are displayed.

Finally replay H, P, N. Transport, provenance, and local naming preserve exactly the displayed packet, so no selected zero, unrestricted host degree, quotient real equality, open-cover compactness, countable-choice schedule, trichotomy, ambient completeness, Lean verification, or bridge datum is exported. □

Theorem 8607.3 (Miranda theorem sign-change handoff)**.** *Every sign-change read accepted by a* $\mathsf{MirandaTheoremFiniteCube}^{\uparrow}$ *packet factors through*

$$\mathsf{Compact}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{SpernerLemma}^{\uparrow}, \quad \mathsf{BrouwerDegree}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads the finite cube and opposite-face signs Q, F, S, then the compact cube and continuous-map source rows C, T, then the Sperner labelled-cell row L and Brouwer-degree obstruction row D, and only then the finite windows, regular readback, and Real seal W, R, E. The handoff exports a finite sign-change certificate, not a host zero selector or an ambient intermediate-value theorem.

Proof. Start with a sign-change request. The carrier exposes only the finite cube row Q, face-sign row F, and sign-stability row S before any downstream obstruction can be read.

The source rows are then forced. The compact row C names the cube as displayed BEDC data, and the continuous-map replay row T records how the face signs are read from the map without introducing a second function object.

Next use the finite obstruction handoff. The Sperner row L supplies the labelled subdivision route, while the Brouwer-degree row D supplies the boundary obstruction ledger attached to the same face signs.

Finally open W, R, E, H, P, N. Windowing, regular readback, Real sealing, transport, provenance, and local naming keep the candidate cell inside the same finite packet, excluding selected roots, unrestricted host degree, quotient real equality, compactness shortcuts, choice, trichotomy, and ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{MirandaTheoremFiniteCube}^{\uparrow}$ sign-change read displays a finite cube row, opposite-face sign rows, sign-stability data, compact cube source, continuous-map replay, Sperner labelled-cell handoff, Brouwer-degree obstruction, finite StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Pkg provenance, and local NameCert naming.

Independence witness: sperner-lemma, brouwer-degree, compact, continuousmap, real: MirandaTheoremFiniteC is not bijective to any listed sibling because it binds finite cube face signs, sign stability, compact source, continuous-map replay, labelled-cell localization, degree obstruction, readback, Real sealing, transport, provenance, and local naming into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MirandaTheoremFiniteCube}^{\uparrow}}$.

Constructive story: A $\mathsf{MirandaTheoremFiniteCube}^{\uparrow}$ packet is generated from finite cube rows, opposite-face sign rows, sign-stability ledgers, compact cube source rows, continuous-map replay, Sperner labelled-cell handoff rows, Brouwer-degree obstruction rows, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MirandaTheoremFiniteCube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8607.1, Definition 8607.2, and Definition 8607.3 seed the finite Miranda cube sign-change route through compact source, continuous-map replay, Sperner handoff, Brouwer-degree obstruction, readback, Real seal, transport, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MirandaTheoremFiniteCube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host zero existence, selected roots, unrestricted degree theory, quotient real equality, open-cover compactness, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, face-sign stability, compact cube source exactness, continuous-map replay, Sperner handoff, Brouwer-degree obstruction, finite readback, Real-seal ordering, transport, provenance, and local naming for BEDC.Derived.MirandaTheoremFiniteCubeUp.

Chapter 8608

A Concrete Naming Certificate for CauchyContinuityCriterion$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyContinuityCriterion$^{\uparrow}$ packet records the finite test by which a displayed continuous map preserves regular Cauchy real names. It sits between the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the continuity-modulus schedule of chapter 292, the continuous-map carrier of chapter 130, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a named Cauchy-preservation boundary without importing host function equality, quotient equality of real names, selected limits, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 8608.1 (Cauchy continuity criterion carrier)**.** *A* CauchyContinuityCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* ContinuousMap$^{\uparrow}$ *row, including the displayed pointwise continuity evidence or modulus row used by the inspected map.*
2. *S is the* StreamName$^{\uparrow}$ *finite-window row for the source regular real name and for the target observations requested by the criterion.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the regular rational names exposed by those windows.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare source and target rational cells.*
5. *M is the* CauchyModulus$^{\uparrow}$ *schedule row that assigns a finite source-window demand to each displayed target tolerance.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after the continuity row, stream windows, regular readback, dyadic tolerance, and Cauchy-modulus schedule have stayed visible.*
7. *H records componentwise* hsame *transport for the continuity row, stream windows, readbacks, dyadic tolerances, modulus schedule, real seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a target tolerance request through $F, S, R, D, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* CauchyContinuityCriterion$^{\uparrow}$, ContinuousMap$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyContinuityCriterion}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite criterion packet for Cauchy-preserving continuity over displayed regular real names. It has no coordinate for a host semantic continuity theorem, quotient equality of completed real names, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8608.2 (Cauchy continuity criterion NameCert obligations)**.** *For every accepted* CauchyContinuityCriterio *carrier $K = (F, S, R, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuityCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, continuous/modulus-row exposure through F, finite StreamName window exposure through S, regular rational readback through R, dyadic tolerance exposure through D, Cauchy-modulus scheduling through M, Real sealing through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation is stored outside these rows.*

Proof. Project the carrier of Definition 8608.1. The accepted tuple displays exactly the continuity row, stream windows, regular readback, dyadic tolerance data, Cauchy-modulus schedule, Real seal, and structural rows.

The continuous/modulus row F supplies the map-facing input, while M supplies the finite schedule that turns each target tolerance request into a source-window demand. The rows S, R, D then display the windows, regular rational readbacks, and dyadic comparisons served by that schedule.

Finally E, H, C, P, N seal, transport, replay, package, and name the same finite route. Since the carrier has no coordinate for host semantic continuity, quotient equality, selected representatives, trichotomy, choice, or ambient Real completeness, no local obligation can export such data. □

Theorem 8608.3 (Cauchy continuity criterion Cauchy preservation)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyContinuityCriterion}^{\uparrow}$ *carrier reads Cauchy preservation only through the displayed route*

$$\mathsf{ContinuousMap}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCe}$$

Thus a displayed regular real name may be sent across the continuous row only after the source windows and dyadic tolerance schedule have been exposed, and the target real-name seal exports no host continuity theorem, quotient equality of names, selected limit, trichotomy, countable-choice schedule, or ambient completeness witness.

Proof. Start with an accepted carrier $K = (F, S, R, D, M, E, H, C, P, N)$. The continuous row F and the Cauchy-modulus row M are the only map-facing and schedule-facing coordinates available to a consumer.

Apply the NameCert obligations of Definition 8608.2. They force the consumer to expose S, R, D before the target-facing seal E can be read, so the regular Cauchy input is inspected by finite windows and dyadic tolerances rather than by a completed host object.

The structural rows H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Since no excluded host datum occurs in the carrier, the exported statement is precisely Cauchy preservation through the displayed continuity criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuityCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyContinuityCriterion}^{\uparrow}$ packet is generated from a displayed continuous or modulus row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a CauchyModulus schedule, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyContinuityCriterion}\uparrow}$, seed)

Scope closed: Definition 8608.1, Definition 8608.2, and Definition 8608.3 seed the finite Cauchy-preserving continuity boundary over displayed regular real names.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyContinuityCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host semantic continuity, quotient equality of real names, selected representatives, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuity-row exposure, finite StreamName projection, RegSeqRat readback, DyadicRatCore tolerance routing, CauchyModulus scheduling, and Real-seal nonescape for BEDC.Derived.CauchyContinuityCriterionUp.

Chapter 8609

A Concrete Naming Certificate for $\mathsf{ChoiceSequenceWindow}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen, the $\mathsf{ChoiceSequenceWindow}^{\uparrow}$ packet records the finite prefix-window object by which a displayed choice-sequence presentation is read by L10 Real and Completion consumers. It sits between the choice-sequence surface of chapter 1482, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite inspected prefix and its routed readback; it does not import a completed branch, quotient stream equality, countable choice, a selected infinite sequence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8609.1 (ChoiceSequenceWindow carrier)**.** *A* $\mathsf{ChoiceSequenceWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row, listing the finite answers already inspected from the choice sequence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened from that prefix, with one displayed window for each finite read requested by the consumer.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision request, comparison slack, and local radius ledger consumed by S and R.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, S, R, D have been exposed.*
6. *H records componentwise* hsame *transport for the prefix, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$Q \longmapsto S, R, D \longmapsto E, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{ChoiceSequenceWindow}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, R, D, E, H, C, P, N$. It has no coordinate for completed branches, quotient streams, countable-choice selectors, selected infinite sequences, host equality of names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8609.2 (ChoiceSequenceWindow NameCert obligations)**.** *For every accepted* $\mathsf{ChoiceSequenceWindow}^{\uparrow}$ *carrier*

$$W = (Q, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChoiceSequenceWindow}^{\uparrow}}$ *surface consists exactly of finite prefix admission for Q, StreamName window exposure for S, RegSeqRat readback for R, DyadicRatCore tolerance accounting for D, terminal Real seal non-escape for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No ChoiceSequenceWindow obligation is stored outside those rows.

Proof. Project the accepted carrier of Definition 8609.1. The source row Q is a finite prefix row, so every consumer starts with displayed choice-sequence answers rather than a completed branch.

The analytic read then passes through the L10 rows. The finite windows are displayed by S, regular rational readback is displayed by R, and the dyadic tolerance ledger is displayed by D before the terminal seal E can be read.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite route, so the carrier cannot export quotient stream equality, countable choice, selected infinite branches, host equality of names, or ambient Real completeness. □

Theorem 8609.3 (ChoiceSequenceWindow StreamName handoff)**.** *Every StreamName, RegSeqRat, or Real-facing consumer of an accepted* $\mathsf{ChoiceSequenceWindow}^{\uparrow}$ *carrier factors through*

$$Q \longmapsto S \longmapsto R, D \longmapsto E, H, C, P, N.$$

The handoff reads the $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row before the* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *row through chapter 301, checks dyadic tolerances through chapter 291, and reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through chapter 38. The route exports no completed branch, quotient stream equality, countable-choice selector, selected infinite sequence, host equality of names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite prefix row Q. Since the carrier contains no completed choice sequence, the downstream read can only use the displayed finite answers already present in the packet.

Next expose S, R, D. The StreamName window is the only route from the prefix into name observations, the RegSeqRat row is the only regular rational readback surface, and the DyadicRatCore row is the only tolerance ledger available to the read.

The Real seal E is therefore terminal for this displayed path. The structural rows H, C, P, N preserve the same route, so any StreamName, RegSeqRat, or Real-facing handoff remains a finite prefix-window handoff rather than a quotient, choice, selected-branch, host-equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequenceWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceSequenceWindow}^{\uparrow}$ packet is generated from a ChoiceSequence finite prefix row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoiceSequenceWindow}^{\uparrow}}$, seed)

Scope closed: Definition 8609.1, Definition 8609.2, and Definition 8609.3 seed the finite prefix-window packet for L10 countable-name consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoiceSequenceWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed branches, quotient stream equality, countable choice, selected infinite sequences, host equality of names, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix coverage, StreamName window routing, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ChoiceSequenceWindowUp.

Chapter 8610

A Concrete Naming Certificate for $\mathsf{RamanujanPartition}^{\uparrow}$

Following Ramanujan and Hardy–Ramanujan on congruences and finite asymptotic windows for the partition function, the $\mathsf{RamanujanPartition}^{\uparrow}$ packet records the finite BEDC surface that a partition-number consumer may inspect before reading a congruence, circle-window, or coefficient-boundary handoff. It sits beside the integer and natural-number carriers of chapter 31 and chapter 28, the polynomial and generating-series surfaces of chapter 50 and chapter 3811, and the real-facing regular readback route of chapter 301. The packet names only finite partition windows, modulus rows, coefficient ledgers, analytic-window witnesses, transport, replay, provenance, and local naming; it is not a host proof of all Ramanujan congruences, a selected infinite q-series, a quotient analytic function, or an ambient asymptotic theorem.

Definition 8610.1 (Ramanujan partition finite-window carrier)**.** *A* $\mathsf{RamanujanPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (N, W, M, Q, C, A, B, H, T, P, L)$$

with the following displayed rows:

1. *N is the finite* $\mathsf{Nat}^{\uparrow}$ *index-window row, listing the partition-number requests admitted by the packet.*
2. *W is the finite partition-table window, exposing the computed $p(n)$ entries for the displayed requests.*
3. *M is the modulus row for the congruence-facing prime-power or arithmetic-progression data.*
4. *Q is the coefficient-extraction row, read through a finite generating-series or polynomial window before any partition value is consumed.*
5. *C is the congruence ledger, recording the displayed divisibility or residue claim only for the finite rows in N, W, M, Q.*
6. *A is the analytic-window row, storing the Hardy–Ramanujan or circle-window estimate data as finite dyadic bounds.*
7. *B is the boundary row that separates exact finite congruence reads from asymptotic or analytic-window reads.*
8. *H records componentwise* hsame *transport for the index, partition-table, modulus, coefficient, congruence, analytic-window, boundary, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a partition consumer request through* N, W, M, Q, C, A, B, H.
10. *P is the* Pkg *provenance over* $\mathsf{RamanujanPartition}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{HadamardProductSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{RamanujanPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, W, M, Q, C, A, B, H, T, P, L$. It has no coordinate for an infinite partition table, selected q-series, quotient analytic equality, all-moduli congruence closure, circle-method completeness, or ambient asymptotic truth.

Theorem 8610.2 (Ramanujan partition NameCert obligations)**.** *For every accepted* $\mathsf{RamanujanPartition}^{\uparrow}$ *carrier*

$$R = (N, W, M, Q, C, A, B, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{RamanujanPartition}^{\uparrow}}$ *surface consists exactly of finite index admission, partition-table window exposure, modulus-row exposure, coefficient-extraction routing, congruence-ledger exactness, analytic-window boundary exposure, separation between exact and asymptotic reads, componentwise transport, displayed replay, package provenance, and local naming. No obligation is discharged by an unbounded partition table, host analytic continuation, selected infinite q-series, quotient function equality, or an ambient circle-method theorem.*

Proof. Project the carrier of Definition 8610.1. The displayed rows are exactly $N, W, M, Q, C, A, B, H, T, P, L$, so every local obligation must be read from those finite rows.

First expose the arithmetic side. The rows N and W give the finite request window and partition-table entries, while M gives the modulus data and Q gives the coefficient-extraction route that produced the displayed values.

Next separate the two consumer routes. The congruence ledger C can be read only against N, W, M, Q, and the analytic-window row A can be read only through the boundary row B. Thus a consumer cannot turn a finite congruence row into a global congruence theorem or an analytic estimate into a completed asymptotic theorem.

Finally H, T, P, L transport, replay, source, and name the same finite route. Since the excluded infinite and host-analytic objects are not carrier coordinates, they cannot appear as local NameCert obligations. □

Theorem 8610.3 (Ramanujan partition finite-window dependency route)**.** *Every congruence-facing or asymptotic-facing consumer of an accepted* $\mathsf{RamanujanPartition}^{\uparrow}$ *carrier must factor through*

$$\mathsf{Nat}^{\uparrow}, \mathsf{Int}^{\uparrow} \longmapsto N, W, M, Q \longmapsto C, A, B \longmapsto H, T, P, L.$$

The route exposes natural indices, integer partition values, coefficient windows, congruence rows, analytic-window bounds, and boundary separation before any partition-function handoff is read.

Proof. Begin with a consumer request for partition data. The carrier gives no direct access to an infinite partition function; the request first reaches the finite index row N and the finite table row W over the natural and integer carriers.

The coefficient and congruence data are then forced through M, Q, C. A congruence-facing consumer sees only the displayed modulus and coefficient route, not a theorem about all arithmetic progressions.

For analytic reads, A must pass through the boundary row B. The rows H, T, P, L then preserve, replay, source, and name the same finite dependency route. Hence the packet exports only the displayed finite-window handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RamanujanPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{RamanujanPartition}^{\uparrow}$ packet is generated from finite Nat index windows, Int partition-value rows, modulus rows, coefficient-extraction windows, congruence ledgers, Hardy–Ramanujan analytic-window bounds, boundary separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RamanujanPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8610.1, Definition 8610.2, and Definition 8610.3 seed the finite partition-function route over displayed arithmetic windows, coefficient extraction, congruence ledgers, analytic-window bounds, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RamanujanPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close all Ramanujan congruences, infinite partition tables, selected q-series, quotient analytic equality, circle-method completeness, ambient asymptotic truth, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, coefficient extraction, congruence-ledger exactness, analytic-boundary separation, structural stability, and local naming for BEDC.Derived.RamanujanPartitionUp.

Chapter 8611

A Concrete Naming Certificate for $\mathsf{BarbalatLemma}^{\uparrow}$

Following Barbalat's lemma and Bishop–Bridges constructive analysis, the $\mathsf{BarbalatLemma}^{\uparrow}$ packet records a finite uniform-continuity and tail-integral route before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits beside the uniform-continuity surface of chapter 391, the regulated-integral surface of chapter 1557, the sequence-window surface of chapter 126, the regular-rational readback of chapter 301, and the real seal of chapter 38. The packet does not export a host limit, a quotient stream, or an ambient completeness theorem; it names the finite rows required to pass from uniform continuity and an integral tail budget to the displayed real readback.

Definition 8611.1 (Barbalat lemma finite carrier)**.** *A* $\mathsf{BarbalatLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, W, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{UniformContinuity}^{\uparrow}$ *modulus row, recording the finite uniform-continuity control used by the packet.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite observation window over which the control is read.*
3. *U is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *tail-budget row, recording the regulated integral budget or integral tail bound used in the handoff.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, naming the regular rational route that receives the displayed tail information.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, read only after the modulus, window, tail budget, and readback rows are displayed.*
6. *H records componentwise* hsame *transport for M, W, U, R, E and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the uniform-continuity route through the tail budget, regular readback, and real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{BarbalatLemma}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BarbalatLemma}^\uparrow}$ *row.*

The classifier has no coordinate for a quotient stream, a selected limit, host real equality, countable choice, or ambient completeness.

Theorem 8611.2 (Barbalat lemma NameCert obligations)**.** *For every accepted* $\mathsf{BarbalatLemma}^\uparrow$ *carrier* $B = (M, W, U, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BarbalatLemma}^\uparrow}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{UniformContinuity}^\uparrow$ *modulus,* $\mathsf{StreamName}^\uparrow$ *window,* $\mathsf{RegulatedIntegral}^\uparrow$ *tail budget,* $\mathsf{RegSeqRat}^\uparrow$ *readback, and* $\mathsf{Real}^\uparrow$ *seal, together with componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. The interface surface is finite and does not export a host limit, a quotient stream, or an ambient completeness claim.*

Proof. Project the carrier of Definition 8611.1. The rows $M, W, U, R, E, H, C, P, N$ are the only public coordinates, so the finite NameCert surface is already listed by the carrier itself.

Replay the displayed route. Uniform continuity is read through M and W, the tail budget is read through U, the regular rational route is read through R, and the real seal is reached only after those rows have been displayed. The structural rows H, C, P, N preserve that same finite packet. □

Theorem 8611.3 (Barbalat lemma uniform continuity integral handoff)**.** *Every accepted* $\mathsf{BarbalatLemma}^\uparrow$ *carrier routes a* $\mathsf{UniformContinuity}^\uparrow$ *modulus through the displayed* $\mathsf{StreamName}^\uparrow$ *window and the* $\mathsf{RegulatedIntegral}^\uparrow$ *tail budget before the* $\mathsf{RegSeqRat}^\uparrow$ *readback can reach the terminal* $\mathsf{Real}^\uparrow$ *seal. The handoff therefore uses the finite rows* $M, W, U, R, E, H, C, P, N$ *and no consumer may bypass* U *or replace the regulated-integral route by a host limit, selected completeness principle, or quotient-stream equality.*

Proof. Start from the carrier definition. The route is explicitly staged as modulus, window, tail budget, readback, and seal, so the integral handoff is not an ambient principle but a displayed finite route.

Now replay the structural rows. The hsame row preserves the same packet, the Cont row replays the same finite handoff, and the provenance and naming rows keep the route attached to the same accepted $\mathsf{BarbalatLemma}^\uparrow$ carrier. Since the packet has no additional coordinate, no bypass can replace the regulated-integral row U. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarbalatLemma}^\uparrow}$.

Constructive story: A $\mathsf{BarbalatLemma}^\uparrow$ packet is generated from a $\mathsf{UniformContinuity}^\uparrow$ modulus row, a $\mathsf{StreamName}^\uparrow$ window row, a $\mathsf{RegulatedIntegral}^\uparrow$ tail budget row, a $\mathsf{RegSeqRat}^\uparrow$ readback row, a $\mathsf{Real}^\uparrow$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarbalatLemma}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 8611.1, Definition 8611.2, and Definition 8611.3 seed the finite Barbalat route from uniform continuity and regulated integral tail data to the real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BarbalatLemma}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a proof of Barbalat's lemma, quotient stream equality, selected limits, host real equality, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, uniform-continuity routing, regulated-integral tail exactness, regular-rational readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BarbalatLemm

Chapter 8612

A Concrete Naming Certificate for DyadicSqueezeLemma$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicSqueezeLemma$^{\uparrow}$ packet isolates the finite dyadic tolerance sandwich used when a middle regular-rational observation row is trapped between lower and upper dyadic rows before an L10 Real or Completion consumer can see the terminal seal. It sits over the dyadic endpoint core of chapter 291, the finite observation schedule of chapter 126, the regular rational readback route of chapter 301, the regular Cauchy enclosure of chapter 1772, the dyadic lower/upper route of chapter 3627, and the terminal real-name seal of chapter 38. The packet names a local dyadic tolerance lemma; it is not quotient stream equality, host trichotomy, a selected limit, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8612.1 (Dyadic squeeze lemma carrier)**.** *A* DyadicSqueezeLemma$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{dy}} = (L, M, U, W, D, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic row, listing the inspected lower endpoints and their finite comparison witnesses.*
2. *M is the middle* RegSeqRat$^{\uparrow}$ *observation row whose displayed rational values are being squeezed.*
3. *U is the upper dyadic row, listing the inspected upper endpoints and their matching comparison witnesses.*
4. *W is the shared* StreamName$^{\uparrow}$ *finite-window row on which L, M, U are jointly observed.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance budget, recording the dyadic radius, split slack, and endpoint arithmetic used by the sandwich.*
6. *R is the* RegSeqRat$^{\uparrow}$ *handoff row that exports the middle observations only after the dyadic bounds and shared window have been displayed.*
7. *H records componentwise* hsame *transport for lower rows, middle observations, upper rows, shared windows, dyadic budgets, handoff rows, replay rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from each Real-facing squeeze request through $L, M, U, W, D, R, H, P, N$ before any terminal seal is visible.*
9. *P is the* Pkg *provenance over* DyadicSqueezeLemma$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchySqueeze$^{\uparrow}$, DyadicSandwich$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, M, U, W, D, R, H, C, P, N$. It has no coordinate for quotient stream equality, host trichotomy, selected limits, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 8612.2 (Dyadic squeeze lemma NameCert obligations)**.** *For every accepted* $\mathsf{DyadicSqueezeLemma}^{\uparrow}$ *carrier*

$$S_{\mathrm{dy}} = (L, M, U, W, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower dyadic row, middle regular-rational observation row, upper dyadic row, shared finite window, dyadic tolerance budget, RegSeqRat handoff, transport, replay, provenance, and local naming;*
2. *lower-to-middle dyadic boundedness, requiring each displayed middle observation in M to lie above the matching lower endpoint in L on the shared window W at the budget in D;*
3. *middle-to-upper dyadic boundedness, requiring the same displayed middle observation to lie below the matching upper endpoint in U on W at the budget in D;*
4. *shared-window stability, requiring all lower, middle, and upper reads to use the same finite observation row W;*
5. *tolerance exactness, requiring D to be the sole dyadic budget consumed by both comparison sides and by the exported handoff;*
6. *RegSeqRat handoff, requiring R to expose the middle regular-rational route only after L, M, U, W, D have been read;*
7. *Real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to pass through the same dyadic and regular-rational rows before any terminal seal is available;*
8. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N;*
9. *refusal of quotient stream equality, host trichotomy, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No dyadic squeeze lemma obligation is stored outside these rows.

Proof. Project the accepted packet $S_{\mathrm{dy}} = (L, M, U, W, D, R, H, C, P, N)$. The carrier exposes exactly the lower dyadic row, middle observation row, upper dyadic row, shared window, dyadic tolerance budget, RegSeqRat handoff, and the structural rows, so carrier admission and provenance are exhausted by those coordinates.

The two boundedness clauses are the two faces of the displayed dyadic sandwich. The lower-to-middle comparison and the middle-to-upper comparison both read the same middle row M, the same finite window W, and the same budget D.

Shared-window stability and tolerance exactness follow from the absence of any alternate schedule or radius coordinate. The packet cannot name a second window for one side of the squeeze, and it cannot replace the displayed dyadic budget with an external real-order comparison.

The handoff row R is downstream of L, M, U, W, D. The sibling routes of chapter 1772 and chapter 3627 consume exactly such displayed regular-rational and dyadic sandwich data before a Real-facing read is exposed.

Finally H, C, P, N transport, replay, source, and locally name the same route. Since the carrier has no quotient stream, trichotomy, selected-limit, countable-choice, ambient-completeness, Lean-verification, or bridge-checking coordinate, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}$ surface. □

Theorem 8612.3 (Dyadic squeeze lemma Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing squeeze read from an accepted* $\mathsf{DyadicSqueezeLemma}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \left(\mathsf{RegularCauchySqueeze}^{\uparrow},\ \mathsf{DyadicSandwich}^{\uparrow}\right) \longrightarrow \mathsf{Real}^{\uparrow}$$

before the seal is visible. The exported read is a finite dyadic tolerance sandwich for the middle regular-rational row; it exports no quotient stream equality, host trichotomy, selected limit, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an accepted carrier $S_{\mathrm{dy}} = (L, M, U, W, D, R, H, C, P, N)$. The dyadic data are read through L, U, D, so every endpoint comparison and tolerance request first enters the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

The finite observations are fixed by W. The middle row M and the handoff row R can therefore expose only the $\mathsf{RegSeqRat}^{\uparrow}$ observations scheduled by the displayed $\mathsf{StreamName}^{\uparrow}$ window.

The regular Cauchy and dyadic sandwich consumers read the same data in the same order. The route of chapter 1772 uses the middle regular-rational row and shared windows, while the route of chapter 3627 uses the lower and upper dyadic bounds with the dyadic budget.

Only after those rows have been replayed through C and transported through H can P, N package the result for the terminal $\mathsf{Real}^{\uparrow}$ read. No remaining coordinate supplies a quotient stream equality test, a host trichotomy case split, a selected limit, countable choice, or ambient Real completeness, so the displayed factorization is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicSqueezeLemma}^{\uparrow}$ packet is generated from lower and upper dyadic rows, a middle RegSeqRat observation row, shared StreamName windows, one DyadicRatCore tolerance budget, a RegSeqRat handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8612.1, Definition 8612.2, and Definition 8612.3 seed the finite dyadic tolerance sandwich route used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSqueezeLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-to-middle bounds, middle-to-upper bounds, shared-window stability, tolerance exactness, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicSqueezeLemmaUp.

Chapter 8613

A Concrete Naming Certificate for $\mathsf{CompactContractionOrbit}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{CompactContractionOrbit}^{\uparrow}$ packet records the finite Picard-orbit data consumed before a compact contraction read is allowed to seal a fixed-point endpoint. It sits between the compact metric source of chapter 129, the complete metric admission boundary of chapter 128, the Lipschitz and contraction-map rows of chapter 410 and chapter 1418, the Banach fixed-point endpoint of chapter 396, and the dyadic finite-cell readback of chapter 5966. The carrier names the finite orbit corridor only; it does not assert a selected fixed point, quotient orbit equality, open-cover compactness, or ambient completeness beyond the displayed rows.

Definition 8613.1 (CompactContractionOrbit carrier)**.** *A* $\mathsf{CompactContractionOrbit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, D, L, T, I, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the inspected compact metric windows and finite-net identifiers.*
2. *D is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row used to name the finite orbit windows and tolerance cells consumed by the compact source.*
3. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *graph row for the displayed self-map, read before any contraction comparison is inspected.*
4. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row, including the strict contraction ledger and adjacent Picard comparison surface.*
5. *I is the finite Picard-orbit row, listing the displayed iterates and their predecessor links.*
6. *Q is the* $\mathsf{CompleteMetric}^{\uparrow}$ *admission row that records where the finite orbit may be handed to a Cauchy-facing consumer.*
7. *R is the regular tolerance readback attached to the dyadic cells and Picard comparisons.*
8. *E is the terminal* $\mathsf{BanachFixedPoint}^{\uparrow}$ *endpoint-request row, readable only after K, D, L, T, I, Q, R have been exposed.*
9. *H records componentwise* hsame *transport for the compact source, dyadic cells, map rows, orbit rows, admission row, endpoint row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the finite route*

$$K, D \longmapsto L, T \longmapsto I, Q, R \longmapsto E, H, P, N.$$

11. *P is the* Pkg *provenance over* CompactContractionOrbit$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompleteMetric$^{\uparrow}$, LipschitzMap$^{\uparrow}$, ContractionMapping$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{CompactContractionOrbit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, D, L, T, I, Q, R, E, H, C, P, N$. It has no coordinate for host fixed-point equality, a selected limit, quotient orbit equality, open-cover compactness, or ambient complete metric data outside the displayed admission row.

Theorem 8613.2 (CompactContractionOrbit NameCert obligations)**.** *For every accepted* CompactContractionOrbi *carrier*

$$O = (K, D, L, T, I, Q, R, E, H, C, P, N),$$

the local NameCert$_{\text{CompactContractionOrbit}^{\uparrow}}$ *surface consists exactly of compact-metric source visibility for K, dyadic finite-cell visibility for D, Lipschitz graph visibility for L, contraction ledger visibility for T, finite Picard-orbit visibility for I, completion-admission visibility for Q, tolerance readback for R, Banach endpoint-request visibility for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No CompactContractionOrbit obligation is stored outside those rows.

Proof. Project the accepted carrier of Definition 8613.1. The compact source and dyadic cell rows K, D are read first, so the orbit corridor starts from displayed finite windows rather than from an ambient compactness theorem.

Next expose the map rows. The Lipschitz graph row L names the self-map surface, while the contraction row T records the strict comparison data used by the Picard step ledger.

The orbit and endpoint rows then become readable in order. The finite Picard row I, completion-admission row Q, and tolerance row R are present before the Banach endpoint request E can be inspected. Transport, replay, provenance, and local naming preserve this same order through H, C, P, N. □

Theorem 8613.3 (CompactContractionOrbit Cauchy handoff)**.** *Every completion-facing or fixed-point-facing consumer of an accepted* CompactContractionOrbit$^{\uparrow}$ *carrier factors through*

$$K, D \longmapsto L, T \longmapsto I, Q, R \longmapsto E, H, C, P, N.$$

The consumer must read the CompactMetric$^{\uparrow}$ *source row, the* DyadicIntervalNet$^{\uparrow}$ *finite-cell row, the* LipschitzMap$^{\uparrow}$ *graph row, the* ContractionMapping$^{\uparrow}$ *contraction row, the finite Picard orbit, and the* CompleteMetric$^{\uparrow}$ *admission row before any* BanachFixedPoint$^{\uparrow}$ *endpoint request is consumed. The route exports no selected fixed point, quotient orbit equality, open-cover compactness, host map equality, or ambient completeness outside the displayed admission row.*

Proof. Begin with the finite compact source. The rows K and D expose the compact metric windows and dyadic cells that bound the orbit data available to the consumer.

Then read the self-map and contraction rows. The graph row L supplies the displayed map surface, while T supplies the contraction comparison needed to interpret adjacent Picard steps.

Finally use the finite-orbit corridor I, Q, R, E, H, C, P, N. The completion-admission row and tolerance readback are visible before the endpoint request, so any downstream Cauchy or fixed-point consumer receives only the displayed Picard corridor rather than a selected endpoint or an ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContractionOrbit}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactContractionOrbit}^{\uparrow}$ packet is generated from a CompactMetric source row, a DyadicIntervalNet finite-cell row, LipschitzMap and ContractionMapping self-map rows, a finite Picard orbit, a CompleteMetric admission row, tolerance readback, a BanachFixedPoint endpoint-request row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContractionOrbit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8613.1, Definition 8613.2, and Definition 8613.3 seed the finite compact contraction orbit corridor consumed before fixed-point sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContractionOrbit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, quotient orbit equality, open-cover compactness, host map equality, ambient complete metric data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source visibility, dyadic cell visibility, Lipschitz and contraction map routing, finite Picard orbit coverage, completion admission, endpoint-request nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactContractionOrbitUp.

Chapter 8614

A Concrete Naming Certificate for $\mathsf{UniformApartnessSpace}^{\uparrow}$

Following Bishop and Bridges on apartness spaces and Kelley on uniform entourages, the $\mathsf{UniformApartnessSpace}^{\uparrow}$ packet records the constructive interface where uniform smallness and positive apartness are read from the same finite carrier. It sits after the entourage rows of chapter 339, the positive-gap rows of chapter 1514, the metric apartness handoff of chapter 3505, the completion transport of chapter 2192, and the filter criterion of chapter 3885. The packet supplies a separated uniform apartness interface for constructive completion consumers; it does not introduce quotient completion, choice, ambient host equality, real trichotomy, or a selected completion point.

Definition 8614.1 (UniformApartnessSpace carrier)**.** *A* $\mathsf{UniformApartnessSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, E, A, M, F, K, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the common carried-object row and local point classifier.*
2. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage source row over X.*
3. *A is the* $\mathsf{ApartnessSpace}^{\uparrow}$ *positive-apartness source row over the same carried observations.*
4. *M is the* $\mathsf{MetricApartness}^{\uparrow}$ *compatibility row, reading positive metric distance as apartness evidence without importing host inequality.*
5. *F is the* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *handoff row for uniformly continuous maps between completion packets.*
6. *K is the* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *filter-criterion row used by completion consumers that read uniform Cauchy data.*
7. *S is the separated-uniform boundary row: if a consumer has only uniform smallness and no displayed apartness evidence, it must remain inside the separated classifier boundary before treating two names as identified.*
8. *R is the row that relates an entourage request, a positive-apartness request, and the metric-apartness witness to one finite readback route.*
9. *H records componentwise* hsame *transport for the carried object, uniform source, apartness source, metric-apartness compatibility, completion-functor handoff, filter criterion, separated boundary, readback, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a completion-facing request through the rows $E, A, M, F, K, S, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{ApartnessSpace}^{\uparrow}$, $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{MetricCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformApartnessSpace}^{\uparrow}}$ *row.*

The carrier is finite and row-visible. It carries no quotient of Cauchy filters, no selected representative, no choice principle, no host equality predicate, and no route that turns absence of apartness into equality outside the displayed separated-uniform boundary.

Theorem 8614.2 (UniformApartnessSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformApartnessSpace}^{\uparrow}$ *carrier $U = (X, E, A, M, F, K, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformApartnessSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, classifier stability under componentwise* hsame *transport, entourage-apartness readback through R, completion handoff through F and K, separated-uniform nonescape through S, visible* Cont *replay through C, package provenance through P, and local naming through N. Every consumer read is forced through the finite route*

$$X, E, A, M, F, K, S, R \longmapsto H, C, P, N.$$

No NameCert row exports quotient completion, choice, ambient host equality, real trichotomy, selected representatives, or equality from mere absence of positive apartness.

Proof. Project the carrier of Definition 8614.1. The rows X, E, A, M, F, K, S, R are the only mathematical rows available to a consumer, while H, C, P, N record transport, replay, provenance, and naming.

Classifier stability is componentwise. The stored hsame rows move the uniform source, apartness source, metric-apartness witness, completion handoff, filter criterion, separated boundary, and readback row together, and the Cont row replays the same finite request after transport.

The excluded principles cannot be read because the carrier has no coordinate for a quotient completion, selected representative, choice function, host equality predicate, real trichotomy, or equality-by-non-apartness rule. Thus the listed rows exhaust the local NameCert surface. □

Theorem 8614.3 (UniformApartnessSpace completion-filter handoff)**.** *Every* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformApartnessSpace}^{\uparrow}$ *packet must first read the route*

$$E, A, M, F, K, S, R, H, C, P, N.$$

The functor row F supplies only the displayed map and extension handoff named by chapter 2192; the criterion row K supplies only the displayed Cauchy-filter completion test named by chapter 3885. Neither row may replace the packet by quotient completion, selected limits, choice, or host equality.

Proof. Begin with a completion-facing request. The uniform source row E identifies the entourage data being consumed, and the apartness source row A records the positive information available on the same carried observations.

The metric-apartness row M translates positive metric distance into the apartness interface, while F and K expose the two completion-facing handoffs. These handoffs are read as finite rows of the same packet, not as external quotient objects.

The separated boundary S and readback row R keep the consumer inside the displayed uniform-apartness interface. Transport, replay, provenance, and naming then pass through H, C, P, N, so no hidden completion quotient, selected point, choice function, or host equality is exported. □

Theorem 8614.4 (UniformApartnessSpace separated uniform nonescape)**.** *Let* $U = (X, E, A, M, F, K, S, R, H, C,$ *be an accepted* $\mathsf{UniformApartnessSpace}^{\uparrow}$ *carrier. If a downstream consumer has only a uniform smallness read from* E *and lacks a displayed positive-apartness read through* A *and* M*, then every equality-facing read remains confined to the separated boundary row* S*. The packet therefore prevents a completion or filter consumer from treating uniform nearness, absence of apartness, or zero-distance notation as host equality.*

Proof. Inspect the rows of Definition 8614.1. Uniform smallness is stored in E, positive apartness is stored through A and M, and equality-facing access is represented only by the separated boundary row S.

A consumer that lacks the positive-apartness route cannot use R to produce a positive separation witness, and it cannot bypass S because the packet has no host equality row. The completion handoff rows F and K are downstream readers of the same boundary, not constructors of ambient equality.

Thus every equality-facing read remains inside S and is transported or replayed only by H and C with provenance P and local naming N. □

Falsifiable prediction: Every accepted $\mathsf{UniformApartnessSpace}^{\uparrow}$ completion-facing read factors through UniformSpace, ApartnessSpace, MetricApartness, MetricCompletionFunctor, CauchyFilterCompletionCriterion, separated-boundary, readback, transport, continuation, provenance, and NameCert rows.

Independence witness: uniformspace, apartnessspace, metricapartness, metriccompletionfunctor, cauchyfiltercompletioncriterion: $\mathsf{UniformApartnessSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a shared uniform source, positive apartness, metric-apartness compatibility, completion-functor handoff, filter-criterion handoff, separated-uniform boundary, readback, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformApartnessSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformApartnessSpace}^{\uparrow}$ packet is generated from a common carried-object row, a UniformSpace entourage row, an ApartnessSpace positive-apartness row, a MetricApartness compatibility row, MetricCompletionFunctor and CauchyFilterCompletionCriterion handoff rows, a separated-uniform boundary, readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformApartnessSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8614.1, Definition 8614.2, Definition 8614.3, and Definition 8614.4 seed the constructive uniform apartness interface for separated completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformApartnessSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, choice, host equality, real trichotomy, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage-apartness readback, completion handoff, separated-uniform nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.UniformApartnessSpaceU

Chapter 8615

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessRate}^{\uparrow}$

Following Bishop and Bridges on Cauchy completeness with displayed moduli, the $\mathsf{CauchyCompletenessRate}^{\uparrow}$ packet records the finite rate-extraction surface by which Real and Completion consumers read a Cauchy rate before treating $\mathsf{Real}^{\uparrow}$ as a completed object. It sits after the rational telescoping route of chapter 5922, the diagonal tail route of chapter 5923, the regular Cauchy telescoping budget of chapter 708, the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 object for extracting displayed Cauchy rates; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8615.1 (Cauchy completeness rate carrier)**.** *A* $\mathsf{CauchyCompletenessRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, D, B, Q, W, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *source row naming the finite rational telescope whose tail is being read.*
2. *D is the* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *diagonal-tail row that exposes the family-indexed Cauchy tail windows.*
3. *B is the* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *row recording the displayed budget for reusing those telescope windows as a regular Cauchy schedule.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the requested dyadic rate bound and slack used by the extraction.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observed stages of the Cauchy stream.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from T, D, B, Q, W.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rate row and readback row have been displayed.*
8. *M is the extracted modulus row, saying that each accepted precision request is discharged by the visible dyadic budget Q and the displayed telescope rows T, D, B, W before the seal E is read.*
9. *H records componentwise* hsame *transport for $T, D, B, Q, W, R, E, M, C, P, N$.*

10. *C records displayed* Cont *replay of the route from the telescope rows through the extracted modulus to the terminal seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessRate}^{\uparrow}$, $\mathsf{RationalCauchyTelescope}^{\uparrow}$, $\mathsf{BishopDiagonalCauchyTelesc}$ $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, D, B, Q, W, R, E, M, H, C, P, N$. It has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, selected host bounds, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8615.2 (Cauchy completeness rate NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletenessRate}^{\uparrow}$ *carrier $K = (T, D, B, Q, W, R, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessRate}^{\uparrow}}$ *surface consists exactly of rational telescope admission, diagonal-tail admission, regular telescoping-budget admission, dyadic tolerance visibility, finite-window exposure, regular rational readback, terminal Real seal handoff, extracted-modulus exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8615.1. The rows T, D, B, Q, W, R, E, M are the only mathematical data available before transport, replay, provenance, and local naming.

The telescope rows T, D, B expose the source rational telescope, its diagonal tail, and the regular telescoping budget. The dyadic ledger Q and stream window W then display the concrete precision request and the stages at which it is checked.

The regular rational row R and the terminal seal E are downstream of those displayed rows. The extracted modulus M names only the finite discharge of the same visible request, so it cannot import a quotient stream, selected host limit, countable choice, trichotomy, selected host bound, or ambient completeness coordinate.

Finally, H, C, P, N transport, replay, source, and name the same route. Since the carrier has no further coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 8615.3 (Cauchy completeness rate modulus extraction)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{CauchyCompletenessRate}^{\uparrow}$ *packet must read the extracted rate through the displayed route*

$\mathsf{RationalCauchyTelescope}^{\uparrow}$, $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$, $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, S

In particular, the modulus row M is visible before the terminal $\mathsf{Real}^{\uparrow}$ *seal E is consumed, and the handoff exports no quotient stream, selected limit, countable choice, trichotomy, selected host bound, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from an accepted carrier $K = (T, D, B, Q, W, R, E, M, H, C, P, N)$. The source telescope T is inspected through the diagonal-tail row D and the regular telescoping budget B.

The precision request is then read from the dyadic ledger Q on the finite windows W. The extracted modulus row M records that this displayed budget discharges the accepted Cauchy-rate request.

Only after the rate row has been displayed may the consumer read R as a $\mathsf{RegSeqRat}^{\uparrow}$ row and pass to the terminal $\mathsf{Real}^{\uparrow}$ seal E. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Because no other row exists in the carrier, a completion consumer cannot replace the displayed extraction by quotient streams, selected representatives, countable choice, trichotomy, selected host bounds, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessRate}^{\uparrow}$ Real or Completion consumer reads RationalCauchyTelescope, BishopDiagonalCauchyTelescope, RegularCauchyTelescopingBudget, DyadicRatCore, StreamName, RegSeqRat, RealUp, extracted modulus, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before using the displayed Cauchy rate.

Independence witness: rational-cauchy-telescope, bishop-diagonal-cauchy-telescope, regular-cauchy-telescoping-budget, dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchyCompletenessRate}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide telescope rows, diagonal tails, budget rows, tolerance ledgers, finite windows, regular rational readback, and terminal sealing, while this packet binds them into one displayed rate-extraction carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessRate}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessRate}^{\uparrow}$ packet is generated from RationalCauchyTelescope, BishopDiagonalCauchyTelescope, RegularCauchyTelescopingBudget, DyadicRatCore, StreamName, RegSeqRat, RealUp, extracted modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessRate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8615.1, Definition 8615.2, and Definition 8615.3 seed the finite L10 rate-extraction route from telescope rows to a displayed Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessRate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, selected host bounds, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, telescope admission, diagonal-tail admission, budget admission, dyadic tolerance exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, extracted-modulus exactness, transport, replay, and ledger non-escape for BEDC.Derived.CauchyCompletenessRateUp.

Chapter 8616

A Concrete Naming Certificate for $\mathsf{UniformCompletionReflection}^{\uparrow}$

Following Bishop and Bridges and Kelley-style constructive uniform completion, the $\mathsf{UniformCompletionReflection}^{\uparrow}$ packet records the finite reflector route by which a uniform source is sent to a separated completion-facing object without quotienting Cauchy filters or selecting limits. It sits beside the dense-source route of chapter 3542, the metric dense embedding of chapter 3351, the separated completion reflection of chapter 3543, and the uniform-space completion packet of chapter 3560. Its finite readback rows are anchored by chapter 126, chapter 301, chapter 291, and the terminal real-name seal of chapter 38.

Definition 8616.1 (Uniform completion reflection carrier)**.** *A* $\mathsf{UniformCompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, F, D, E, R, W, Q, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the source uniformity row, exposing the uniform entourage data admitted by the reflector.*
2. *F is the Cauchy-filter window row, recording the finite filter requests inspected before a completion-facing read is accepted.*
3. *D is the dense embedding row, read through* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *and* $\mathsf{MetricCompletionDenseEmbedd}$ *before any reflected completion point is public.*
4. *E is the separated reflection row, read through* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *before the completion-facing seal can be compared.*
5. *R is the universal arrow row, recording the* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *handoff available to uniform-continuous consumers.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect Cauchy-filter evidence.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational observations extracted from the displayed windows.*
8. *Z is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and terminal* $\mathsf{Real}^{\uparrow}$ *sealing route for the inspected comparison.*
9. *H records componentwise* hsame *transport for $S, F, D, E, R, W, Q, Z, C, P, N$.*

10. *C records displayed* Cont *replay from a reflector request through the source uniformity, Cauchy-filter windows, dense embedding, separated reflection, universal arrow, numerical readback, transport, provenance, and local naming rows.*

11. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionReflection}^{\uparrow}$, $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{SeparatedCompletionReflection}^{\uparrow}$, $\mathsf{UniformSpaceCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, D, E, R, W, Q, Z, H, C, P, N$. It has no coordinate for quotient completion classes, selected filter representatives, selected limits, countable choice, host equality of completed points, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8616.2 (Uniform completion reflection NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompletionReflection}^{\uparrow}$ *carrier $U = (S, F, D, E, R, W, Q, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionRefl}}$ *surface consists exactly of source uniformity admission, Cauchy-filter window exposure, dense embedding exposure through* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *and* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, *separated reflection exposure through* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$, *universal-arrow exposure through* $\mathsf{UniformSpaceCompletion}^{\uparrow}$, *StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance and* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 8616.1. The tuple displays the source uniformity, Cauchy-filter windows, dense embedding, separated reflection, universal arrow, finite windows, regular rational readback, dyadic and Real sealing, transport, replay, provenance, and local naming rows.

The completion-facing obligations are downstream from the finite source evidence. A consumer first reads S and F, then the dense-source rows D, then the separated reflection row E, and only then the universal arrow row R.

The numerical and structural rows W, Q, Z, H, C, P, N preserve the same route. Since the packet contains no quotient class, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate, the NameCert surface is exhausted by the displayed rows. □

Theorem 8616.3 (Uniform completion reflection density route)**.** *Every density read exported by an accepted* $\mathsf{UniformCompletionReflection}^{\uparrow}$ *packet factors through* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *and* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *before the separated reflection or universal-arrow rows are consumed. The density route is carried by $S, F, D, W, Q, Z, H, C, P, N$ and remains a finite BEDC packet, not a quotient completion or selected representative.*

Proof. Read the source uniformity row S and Cauchy-filter window row F. These rows bound the request before any dense image can be inspected.

The dense embedding row D is then supplied by the dense-source and metric dense-embedding packets. Its readback still passes through the StreamName, RegSeqRat, DyadicRatCore, and Real rows W, Q, Z.

Transport, replay, provenance, and local naming keep the route within H, C, P, N. Thus density is exported only as displayed finite evidence, with no quotient completion class or selected representative available to the consumer. □

Theorem 8616.4 (Uniform completion reflection universal route)**.** *Every universal-arrow read from an accepted* $\mathsf{UniformCompletionReflection}^{\uparrow}$ *packet factors through*

$$S, F, D, E, W, Q, Z, H, C, P, N \longmapsto R.$$

The row R may be consumed by $\mathsf{UniformSpaceCompletion}^{\uparrow}$*-facing maps only after the source uniformity, Cauchy-filter window, dense embedding, separated reflection, numerical readback, transport, replay, provenance, and local naming rows remain visible in the same packet.*

Proof. Start with Definition 8616.2. It places the universal arrow after the source uniformity, Cauchy-filter windows, dense embedding, separated reflection, and numerical readback rows.

The density route of Definition 8616.3 supplies the source-to-completion side. The separated reflection row E then prevents the consumer from treating zero-distance evidence as host equality before the reflection boundary is displayed.

Finally project R through H, C, P, N. The universal route is therefore a replayable NameCert row for $\mathsf{UniformSpaceCompletion}^{\uparrow}$ consumers, not a selected-limit operator, quotient completion theorem, countable-choice principle, or ambient separated completion. □

Falsifiable prediction: Every accepted $\mathsf{UniformCompletionReflection}^{\uparrow}$ consumer that reaches the universal arrow while omitting source uniformity, Cauchy-filter windows, dense embedding, separated reflection, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real sealing, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cauchy-completion-density-source, metric-completion-dense-embedding, separated-completion-reflection, uniform-space-completion, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{UniformCompletionReflection}^{\uparrow}$ because the packet jointly carries source uniformity, Cauchy-filter windows, dense embedding, separated reflection, universal-arrow routing, finite windows, regular rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCompletionReflection}^{\uparrow}$ packet is generated from a source uniformity row, Cauchy-filter windows, dense embedding rows, separated reflection rows, a universal-arrow row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8616.1, Definition 8616.2, Definition 8616.3, and Definition 8616.4 seed the constructive uniform-completion reflector route from source uniformity and Cauchy-filter windows through dense embedding, separated reflection, and universal-arrow rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected filter representatives, selected limits, countable choice, trichotomy, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-uniformity exactness, Cauchy-filter window exposure, dense embedding handoff, separated reflection non-escape, universal-arrow routing, finite numerical readback, transport, replay, provenance, and local naming for BEDC.Derived.UniformCompletionReflectionUp.

Chapter 8617

A Concrete Naming Certificate for $\mathsf{CantorNestedInterval}^{\uparrow}$

Following Cantor and Bishop, the $\mathsf{CantorNestedInterval}^{\uparrow}$ packet records the constructive nested-interval argument as finite NameCert data rather than as an ambient compactness or completeness theorem. It sits on the Phase Real and Completion frontier by reading dyadic rational endpoints from chapter 291, finite endpoint streams from chapter 126, regular rational readback from chapter 301, terminal real sealing from chapter 38, and limit uniqueness only through the already named boundary of chapter 5983.

Definition 8617.1 (Cantor nested interval carrier)**.** *A* $\mathsf{CantorNestedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows.

1. *L and U are finite endpoint histories whose entries are read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *as lower and upper rational endpoints.*
2. *D is a dyadic diameter budget assigning each displayed index k a dyadic bound for the interval width.*
3. *W is the finite window in which the packet checks endpoint order and nesting for all displayed adjacent stages.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the endpoint windows into regular rational midpoint or endpoint reads.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from those regular reads.*
6. *H, C, P, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CantorNestedInterval}^{\uparrow}}$ *rows.*

The carrier accepts only histories where each displayed interval is closed, rational, nested in the previous displayed interval, and bounded by the displayed dyadic diameter budget. It contains no coordinate for a selected point, a quotient real equality, trichotomy of endpoints, Heine–Borel compactness, or full real completeness.

Theorem 8617.2 (Cantor nested interval NameCert obligations)**.** *For every accepted* $\mathsf{CantorNestedInterval}^{\uparrow}$ *carrier*

$$I = (L, U, D, W, R, E, H, C, P, N),$$

the NameCert obligation surface consists exactly of carrier admission, finite closed-rational endpoint rows, displayed nesting, dyadic diameter decrease, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal through* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CantorNestedInterval}^{\uparrow}}$ *row* N*. The packet exports no Heine–Borel principle, selected point operator, quotient real equality, endpoint trichotomy, or full completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 8617.1. The rows L, U, D, W display exactly the finite rational endpoint data, nesting checks, and dyadic diameter bounds; no external interval family is hidden outside those histories.

Use the real-frontier route named in the chapter introduction. The endpoint histories read through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$, the regular rational handoff is the row R, and the terminal real object is exposed only through the seal E.

The remaining rows are structural. Componentwise hsame transport H preserves the displayed endpoint and diameter rows, Cont replay C records the finite window used by a consumer, Pkg provenance P keeps the interval packet visible, and N names the local certificate. Since no row stores a selected point, quotient equality, trichotomy, compactness principle, or ambient completeness proof, the listed obligations are the entire certificate surface. □

Theorem 8617.3 (Cantor nested interval regular Cauchy seal)**.** *Every accepted* $\mathsf{CantorNestedInterval}^{\uparrow}$ *carrier whose displayed windows cover a requested precision* k *yields a regular Cauchy seal only by reading the finite endpoint window, dyadic diameter budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *row displayed in the carrier. The seal is a finite-window route from nested rational intervals to a regular rational readback; it is not an ambient completeness theorem and does not choose a point outside the displayed packet.*

Proof. Fix the requested precision k and open the displayed window W large enough for the dyadic budget row D at that precision. The nested endpoint rows L, U then give a finite closed rational interval whose diameter is bounded by the requested dyadic tolerance.

Read the chosen finite endpoint data through $\mathsf{DyadicRatCore}^{\uparrow}$ and the window interface through $\mathsf{StreamName}^{\uparrow}$. The regular rational readback row R supplies the Cauchy-stage comparison, while the terminal row E seals the read as a $\mathsf{Real}^{\uparrow}$ object.

The construction has used only the displayed finite rows of the carrier. Therefore the resulting seal is the regular Cauchy seal named by the packet and cannot be reinterpreted as a selected nested-intersection point, quotient real equality, Heine–Borel compactness, trichotomy, or full completeness of the real line. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorNestedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorNestedInterval}^{\uparrow}$ packet is generated from finite lower and upper dyadic rational endpoint histories, a displayed dyadic diameter budget, a finite nesting window, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorNestedInterval}^{\uparrow}}$, seed)

Scope closed: The seed scope is the finite nested rational interval carrier, dyadic diameter budget, RegSeqRat handoff, terminal Real seal, and local NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorNestedInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close Heine–Borel compactness, selected intersection points, quotient real equality, endpoint trichotomy, or full Real completeness.

Upgrade path: Formalization should add BEDC.Derived.CantorNestedIntervalUp with the carrier, obligation theorem, and finite-window regular Cauchy seal.

Chapter 8618

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ packet records the real-line interface obtained from displayed regular Cauchy rational data, not from a selected quotient completion. It sits over the dyadic tolerance core of chapter 291, the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the separability surface of chapter 1809. The packet names a finite Real and Completion interface for constructive Cauchy reals without importing quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8618.1 (Constructive Cauchy real line carrier)**.** *A* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for comparison scales, slack witnesses, and refusal rows.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the Cauchy-name observations consumed by the line interface.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those displayed stream windows and dyadic tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after D, S, R have been displayed.*
5. *Q is the* $\mathsf{RealSeparability}^{\uparrow}$ *countable-dense-source row used as a named separability witness for the same real-line packet.*
6. *H records componentwise* hsame *transport for the dyadic, stream, regular-rational, Real-seal, separability, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a real-line request through D, S, R, E, Q, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealLine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient equality of Cauchy streams, selected representatives, countable-choice schedules, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8618.2 (Constructive Cauchy real line classifier route)**.** *For every accepted* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *carrier* $L = (D, S, R, E, Q, H, C, P, N)$*, the classifier reads exactly the dyadic tolerance row, finite stream-window row, regular rational readback, terminal Real seal, separability source, componentwise transport, displayed replay, provenance, and local naming rows. Two packets with the same displayed rows are indistinguishable to the classifier, and no omitted quotient, selector, host equality, choice, trichotomy, or completeness datum can be recovered from the accepted packet.*

Proof. Project the carrier of Definition 8618.1. Its public rows are precisely $D, S, R, E, Q, H, C, P, N$, and the classifier is defined over those displayed coordinates.

The analytic route is staged by the same rows. Dyadic tolerances and stream windows are read before the regular rational row, the regular rational row is read before the terminal $\mathsf{Real}^{\uparrow}$ seal, and the separability row is named beside that seal.

The remaining rows only transport, replay, source, and name the packet. Since no coordinate stores a quotient representative, selector, host equality, countable-choice schedule, trichotomy oracle, or ambient completeness theorem, the classifier cannot export such data. □

Theorem 8618.3 (Constructive Cauchy real line NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealLine}}$ *surface consists exactly of carrier admission for* $D, S, R, E, Q, H, C, P, N$*, dyadic tolerance exposure, finite stream-window exposure, regular rational readback, terminal Real sealing, separability-source exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission supplies the packet $L = (D, S, R, E, Q, H, C, P, N)$. Each NameCert obligation is a projection of one displayed coordinate or of the ordered route among those coordinates.

The source obligations are D, S, R, E, Q. They expose the dyadic ledger, finite windows, regular rational readback, terminal seal, and separability source before a Real or Completion consumer can read the packet.

The structural obligations are H, C, P, N. They preserve the same finite route under transport, replay the request through displayed continuations, record sibling provenance, and provide the local name. No further NameCert row is available. □

Theorem 8618.4 (Constructive Cauchy real line source exactness)**.** *Every accepted* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *read reaches its terminal* $\mathsf{Real}^{\uparrow}$ *seal only through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{RealSeparability}^{\uparrow}.$$

The dyadic ledger D*, stream schedule* S*, regular rational row* R*, Real seal* E*, and separability source* Q *remain one finite source packet; none may be replaced by a host Cauchy sequence, quotient real, selected limit, or unrecorded dense subset.*

Proof. Use the classifier route of Definition 8618.2. It exposes D, S, R, E, Q as the only source rows available to a Real or Completion consumer.

The order of the read is forced by the carrier. Tolerances and finite windows are displayed before regular rational readback, the readback is displayed before the Real seal, and separability is named as a row of the same packet.

Transport and replay preserve this source packet through H and C, while P and N record provenance and naming only. Hence a consumer cannot substitute a host sequence, quotient real, selected limit, or unrecorded dense source for the displayed rows. □

Theorem 8618.5 (Constructive Cauchy real line transport stability)**.** *Let L_0 and L_1 be accepted* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *carriers. If their dyadic tolerance rows, stream-window schedules, regular rational readbacks, Real seals, separability rows, replay rows, provenance rows, and local naming rows are componentwise* hsame*, then every Real or Completion read accepted for L_0 is accepted for L_1 with the transported finite route.*

Proof. Read the componentwise transport row H. It is scoped over the dyadic, stream, regular-rational, Real-seal, separability, replay, provenance, and local naming rows.

Use displayed replay through C. A consumer request for L_0 is replayed through the ordered rows D, S, R, E, Q, H, P, N, and the hsame route transports those rows componentwise to L_1.

Source exactness from Definition 8618.4 keeps the transported read inside the same finite route. The transported packet therefore exports no quotient equality, selector, choice schedule, trichotomy, host equality, or completeness datum. □

Theorem 8618.6 (Constructive Cauchy real line ledger exactness)**.** *Every comparison, readback, Real-seal, and separability read in an accepted* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *carrier is ledgered by $D, S, R, E, Q, H, C, P, N$. The ledger records the dyadic tolerance budget, finite stream windows, regular rational observations, terminal Real seal, separability source, transport, replay, provenance, and local naming as one packet, and it records no selected representative, hidden quotient class, or countable-choice schedule.*

Proof. Start with the NameCert obligations of Definition 8618.3. They identify every admitted read as one of the displayed carrier rows.

The comparison and readback part is D, S, R. The Real-facing and separability part is E, Q. The structural part is H, C, P, N.

Together these rows form the whole ledger. Because the ledger has no selector, quotient representative, or countable-choice coordinate, an accepted read cannot hide one behind the Real seal or separability source. □

Theorem 8618.7 (Constructive Cauchy real line standard-bridge seed)**.** *The* $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ *packet supplies a standard-bridge seed for the Real and Completion route: every candidate bridge must preserve the dyadic ledger D, finite stream schedule S, regular rational readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, separability row Q, componentwise transport H, displayed replay C, provenance P, and local naming N. The seed is not itself a checked bridge and does not assert ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply source exactness and transport stability. They force every Real or Completion read to pass through $D, S, R, E, Q, H, C, P, N$ and keep that route stable under componentwise hsame.

Use ledger exactness to identify the bridge payload. A bridge candidate that omits one of the displayed rows would no longer preserve the local NameCert surface of the packet.

The conclusion is only a seed obligation. Since the chapter provides no checked standard-bridge theorem and no completeness coordinate, ambient $\mathsf{Real}^{\uparrow}$ completeness and bridge checking remain outside the claim. □

Falsifiable prediction: Every accepted ConstructiveCauchyRealLineUp read displays DyadicRatCore, StreamName, RegSeqRat, Real, RealSeparability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a Real or Completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, real-separability: ConstructiveCauchyRealLin is not bijective to any listed sibling because it jointly carries dyadic tolerances, finite stream windows, regular rational readback, terminal Real sealing, separability source data, transport, replay, provenance, and local naming in one real-line packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealLine}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, a Real seal, RealSeparability source data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8618.1, Definition 8618.2, Definition 8618.3, Definition 8618.4, and Definition 8618.7 seed the finite constructive Cauchy real-line route over DyadicRatCore, StreamName, RegSeqRat, Real, RealSeparability, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier routing, dyadic tolerance exposure, finite stream-window exposure, regular rational readback, Real sealing, separability source exposure, transport stability, ledger exactness, standard-bridge seed obligations, provenance, and local naming for BEDC.Derived.ConstructiveCauchyRealLineUp.

Chapter 8619

A Concrete Naming Certificate for $\mathsf{DowkerSpace}^{\uparrow}$

Following Engelking and Kelley, the $\mathsf{DowkerSpace}^{\uparrow}$ packet records the standard topology boundary where normality is visible but countable paracompactness is obstructed by the interval product test. It sits over the topology carrier of chapter 88, the normal-space surface of chapter 1866, the paracompact comparison of chapter 1963, the locally finite cover boundary of chapter 7245, and the product-topology route of chapter 2000. The packet names the finite boundary data only; it does not import host powersets, selected open covers, open-cover compactness, or a choice principle.

Definition 8619.1 (Dowker space carrier)**.** *A* $\mathsf{DowkerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, N, C, P, I, H, R, Q, Z)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *carrier row for the admitted point and open-set names.*
2. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *exposure row recording the displayed normal separation source.*
3. *C is the countable-paracompactness obstruction row; it records the finite refusal boundary rather than a selected countable cover.*
4. *P is the* $\mathsf{Paracompact}^{\uparrow}$ *comparison row, read only through the local obstruction and locally finite cover surfaces.*
5. *I is the interval and product-normality test row, routing the boundary through the product topology comparison.*
6. *H records componentwise* hsame *transport for topology, normality, obstruction, comparison, product test, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from a consumer request through the topology row, normal-source row, obstruction row, paracompact comparison, and interval-product test.*
8. *Q is the* Pkg *provenance over topology, normal space, paracompact comparison, locally finite cover, interval-product topology,* BHist, hsame, Cont, *and local naming rows.*
9. *Z is the local* $\mathsf{NameCert}_{\mathsf{DowkerSpace}^{\uparrow}}$ *row naming this boundary object.*

The carrier has no coordinate for a host powerset, a selected open cover, open-cover compactness, quotient equality of covers, or a choice-based paracompactness witness.

Theorem 8619.2 (Dowker space NameCert obligations)**.** *For every accepted* $\mathsf{DowkerSpace}^{\uparrow}$ *carrier* $D = (T, N, C, P, I, H, R, Q, Z)$*, the local NameCert surface consists of carrier admission for the nine displayed rows, normal-source exposure through* N*, countable-paracompact obstruction through* C*, product-normality boundary through* I*, componentwise transport through* H*, replay through* R*, provenance through* Q*, and local naming through* Z*. The paracompact comparison row* P *is readable only as a comparison against the displayed obstruction and does not supply a selected cover, a host powerset, open-cover compactness, or a global choice principle.*

Proof. Project the carrier of Definition 8619.1. The tuple displays exactly the topology carrier, normal source, countable-paracompact obstruction, paracompact comparison, interval-product test, transport, replay, provenance, and local naming rows.

The positive topological read starts with T and N. These rows expose the underlying topology and the normal-space source, but they do not create a paracompactness witness.

The boundary read is C, P, I. The obstruction row C records the countable-paracompactness refusal, P records only the comparison with paracompact and locally finite cover surfaces, and I records the interval-product normality test that consumes that same boundary.

The structural rows H, R, Q, Z transport, replay, source, and name the finite surface. Since the carrier omits host powersets, selected covers, open-cover compactness, and choice-based cover selectors, no consumer can extract those objects from the NameCert. □

Theorem 8619.3 (Dowker space normal-paracompact boundary)**.** *Every consumer of an accepted* $\mathsf{DowkerSpace}^{\uparrow}$ *packet that uses the normality row must keep* $\mathsf{NormalSpace}^{\uparrow}$*, the* $\mathsf{Paracompact}^{\uparrow}$ *and countably-paracompact comparison row, the* $\mathsf{Topology}^{\uparrow}$ *carrier row, and the interval/product topology row visible. The handoff may compare normality, paracompactness, locally finite covers, and product normality along the displayed rows* $T, N, C, P, I, H, R, Q, Z$*, but it cannot export host powersets, selected covers, selected refinements, open-cover compactness, or a global paracompactness witness.*

Proof. Start from the carrier of Definition 8619.1. A normality consumer reads N only after the topology carrier T has been displayed.

The comparison cannot bypass the obstruction row. The row C keeps the countable-paracompactness boundary visible, while P and I route any paracompact or product-normality request through the displayed comparison surface.

The locally finite cover and product-topology dependencies are read through provenance and replay. Rows R and Q point to the same finite route, and H preserves it componentwise under hsame.

Thus the consumer receives the normal-paracompact boundary as a finite NameCert surface. Host powersets, selected covers, selected refinements, open-cover compactness, and global paracompactness witnesses are absent from the carrier and cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DowkerSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DowkerSpace}^{\uparrow}$ packet is generated from finite BHist rows for topology, normality, countable-paracompact obstruction, paracompact comparison, interval-product testing, transport, replay, provenance, and local naming. The normal row is readable only with the obstruction and product-test rows still visible, so the packet records the standard normal/paracompact boundary without selected covers or host powersets.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DowkerSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the Dowker-space carrier, NameCert obligations, and normal-paracompact boundary for the displayed topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DowkerSpace}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DowkerSpaceUp`

Bridge status: none

Not claimed: It does not claim selected covers, host powersets, open-cover compactness, choice-based paracompactness, quotient cover equality, or a standard bridge.

Upgrade path: Obligation closure requires Lean encodings of carrier admission, normal-source exposure, obstruction preservation, product-normality boundary, and NameCert non-escape.

Chapter 8620

A Concrete Naming Certificate for $\mathsf{UniformEntourageCompletion}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces, the $\mathsf{UniformEntourageCompletion}^{\uparrow}$ packet records the L10 completion boundary where displayed entourage-Cauchy data are routed to a separated real-facing completion surface as finite BEDC rows. It sits after the uniform-space entourage discipline of chapter 339, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal Real seal of chapter 38, and the separated-completion boundary of chapter 1612. The packet names an entourage-source completion handoff without quotient completion equality, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8620.1 (Uniform entourage completion carrier)**.** *A* $\mathsf{UniformEntourageCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, E, W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, exposing the finite entourage requests and uniformity witnesses consumed before any completion-facing read.*
2. *E is the entourage-Cauchy row. It records the displayed entourage windows used to recognize Cauchy behavior inside the source uniformity.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which observed points, tails, and comparison requests are read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite windows in W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii and comparison budgets used by the Cauchy-window reads.*
6. *S is the separated* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{SeparatedCompletion}^{\uparrow}$ *handoff row. It is readable only after U, E, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the uniform source, entourage-Cauchy, window, readback, tolerance, separated handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$U, E \longmapsto W, R, D \longmapsto S, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{UniformEntourageCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformEntourageCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, E, W, R, D, S, H, C, P, N$. It has no coordinate for quotient completion equality, quotient entourage filters, selected representatives, selected limits, countable-choice schedules, host equality, trichotomy, or ambient completeness.

Theorem 8620.2 (Uniform entourage completion NameCert obligations)**.** *For every accepted* $\mathsf{UniformEntourageCompletion}^{\uparrow}$ *carrier $K = (U, E, W, R, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformEntourageCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform source row, entourage-Cauchy row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, separated Real-completion handoff, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing obligation is read through those rows before the separated handoff S is opened.*

Proof. Project the carrier of Definition 8620.1. The displayed tuple exposes $U, E, W, R, D, S, H, C, P, N$ and no other completion-facing coordinate.

The source and Cauchy evidence are read first. The uniform source row U supplies the entourage discipline, while E records the Cauchy-window rows inside that discipline.

The finite observation route is then forced through W, R, D. StreamName windows expose the observed tails, RegSeqRat rows read back the finite windows, and DyadicRatCore rows record the tolerance comparisons used by the Cauchy request.

Finally read S, H, C, P, N. The separated Real-completion handoff is downstream of the displayed source, window, readback, and tolerance rows, and the structural rows preserve only that route. Since the excluded quotient, selector, choice, host-equality, trichotomy, and ambient-completeness objects are not carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 8620.3 (Uniform entourage completion L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{UniformEntourageCompletion}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto E \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{SeparatedCompletion}^{\uparrow}.$$

In carrier notation, every such read factors through

$$U, E \longmapsto W, R, D \longmapsto S, H, C, P, N.$$

Thus the separated completion boundary is a displayed handoff from finite entourage-source Cauchy data, not a quotient completion equality, selected representative, countable-choice schedule, host equality, or ambient completeness premise.

Proof. Apply the NameCert obligation surface of Definition 8620.2. It confines every accepted completion-facing read to the carrier rows $U, E, W, R, D, S, H, C, P, N$.

Read the route in row order. The source row U and entourage-Cauchy row E precede the observation rows, so the consumer cannot start from a detached quotient filter or a selected completed point.

The finite observation rows W, R, D precede the separated handoff. They expose StreamName windows, regular rational readback, and dyadic tolerance budgets before the Real or separated-completion boundary in S is available.

The structural rows then replay the same route. Componentwise transport, displayed continuation, provenance, and local naming preserve the finite handoff, so no omitted quotient, selector,

choice, host-equality, trichotomy, or ambient-completeness coordinate can be introduced downstream. □

Falsifiable prediction: Every accepted $\mathsf{UniformEntourageCompletion}^{\uparrow}$ completion-facing read factors through $\mathsf{UniformSpace}^{\uparrow}$, entourage-Cauchy rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, a separated $\mathsf{Real}^{\uparrow}$ and $\mathsf{SeparatedCompletion}^{\uparrow}$ handoff, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Independence witness: uniformspace, streamname, regseqrat, dyadicratcore, real, separated-completion: no listed sibling is bijective to $\mathsf{UniformEntourageCompletion}^{\uparrow}$ because this packet jointly carries the uniform source, entourage-Cauchy windows, finite readback, dyadic tolerances, separated handoff, transport, replay, provenance, and local naming as one completion boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEntourageCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEntourageCompletion}^{\uparrow}$ packet is generated from UniformSpaceUp entourage rows, entourage-Cauchy window rows, StreamNameUp observations, RegSeqRatUp readback, DyadicRatCoreUp tolerances, RealUp and SeparatedCompletionUp handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEntourageCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8620.1, Definition 8620.2, and Definition 8620.3 seed the L10 entourage-source completion boundary over UniformSpaceUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, SeparatedCompletionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEntourageCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected representatives, countable choice, host equality, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, entourage-Cauchy window routing, StreamName and RegSeqRat readback, DyadicRatCore tolerance ordering, separated Real-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformEntourageCompletionUp.

Chapter 8621

A Concrete Naming Certificate for $\mathsf{BishopIntervalSearch}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, the $\mathsf{BishopIntervalSearch}^{\uparrow}$ packet records the finite interval-search route by which a located predicate over a displayed closed real interval returns a certified subinterval and a regular Cauchy readback. It sits after the terminal real-name surface of chapter 38, the constructive real-line route of chapter 8618, the nested-interval source of chapter 8617, the dyadic tolerance core of chapter 291, the finite stream-name surface of chapter 126, and the regular rational readback of chapter 301. The packet names only a finite search certificate; it does not choose a completed point, quotient a Cauchy stream, decide a global order, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8621.1 (Bishop interval search carrier)**.** *A* $\mathsf{BishopIntervalSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, D, G, R, W, H, C, P, N)$$

with the following displayed rows:

1. *L records located interval endpoints, endpoint-apartness evidence, and the closed interval boundary read through* $\mathsf{Real}^{\uparrow}$*.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid, mesh, midpoint, and refinement budget used by the search.*
3. *G records the finite dyadic grid and the chosen nested-refinement rows, including the parent-child containment witnesses for every inspected cell.*
4. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the displayed endpoint and grid windows.*
5. *W records the result witness: the accepted subinterval, the located predicate side selected by the search, and the refusal rows for cells not selected.*
6. *H records componentwise* hsame *transport for the endpoint, dyadic-grid, refinement, regular-readback, result, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an interval-search request through L, D, G, R, W, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalSearch}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConstructiveCauchyRealLine}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalSearch}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a selected real point, quotient stream equality, host interval equality, omniscient case split, countable-choice schedule, or ambient completeness theorem.

Theorem 8621.2 (Bishop interval search NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BishopIntervalSearch}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L, D, G, R, W, H, C, P, N$*, located endpoint exposure, dyadic grid and refinement exposure, regular Cauchy readback, result-witness exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission supplies the packet $I = (L, D, G, R, W, H, C, P, N)$. The source obligations are projections of L, D, G, R, W: endpoints are located, the dyadic grid and refinement budget are displayed, the regular rational readback is visible, and the selected result interval is carried with its refusal rows.

The structural obligations are projections of H, C, P, N. Transport preserves the displayed rows componentwise, replay sends each request through the finite search route, provenance records the sibling Real and Completion sources, and the local certificate names the packet.

Since the carrier has no coordinate for a selected point, quotient stream equality, host interval equality, omniscient case split, countable-choice schedule, or ambient completeness theorem, no NameCert obligation can export one. □

Theorem 8621.3 (Bishop interval search located-net correctness)**.** *For every accepted* $\mathsf{BishopIntervalSearch}^{\uparrow}$ *carrier* $I = (L, D, G, R, W, H, C, P, N)$*, the result witness* W *is read only after the located endpoint row* L*, the dyadic grid budget* D*, the nested refinement row* G*, and the regular Cauchy readback* R *have been displayed. The selected subinterval is therefore a cell of the displayed finite net, and every refusal attached to a nonselected cell is ledgered by the same packet.*

Proof. Read the carrier route from Definition 8621.1. The result row W is not an independent search oracle; it is a row after the endpoint, dyadic-grid, refinement, and regular-readback rows.

The containment claim is a finite inspection of G. Each selected cell names its parent interval, mesh row, and endpoint witnesses, so the selected subinterval belongs to the displayed net rather than to an ambient interval object.

The refusal rows for nonselected cells are part of W and are replayed through C. Componentwise transport through H keeps the same refusals attached to the same endpoint, grid, refinement, and readback data. □

Theorem 8621.4 (Bishop interval search Cauchy stability)**.** *Let* I_0 *and* I_1 *be accepted* $\mathsf{BishopIntervalSearch}^{\uparrow}$ *carriers. If their located endpoint rows, dyadic grid budgets, nested refinement rows, regular Cauchy readbacks, result witnesses, replay rows, provenance rows, and local naming rows are componentwise* hsame*, then every interval-search read accepted for* I_0 *is accepted for* I_1 *with the transported finite result route.*

Proof. Use the transport row H. It is scoped over L, D, G, R, W, C, P, N, so each endpoint observation, grid budget, refinement cell, regular Cauchy readback, result witness, replay, provenance entry, and local name has a transported mate.

Apply displayed replay through C. A request accepted for I_0 is replayed through L, D, G, R, W, H, P, N, and the componentwise hsame route transports exactly those rows to I_1.

Located-net correctness from Definition 8621.3 keeps the transported result inside the displayed finite net. Thus Cauchy-stable search exports the same finite result route without selecting a completed point, quotienting streams, invoking choice, or using ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted BishopIntervalSearchUp read displays located Real endpoints, DyadicRatCore grid rows, nested refinement rows, RegSeqRat readback, a result witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before an interval-search consumer reads the packet.

Independence witness: real, constructive-cauchy-real-line, cantor-nested-interval, dyadicrat-core, streamname, regseqrat: $\mathsf{BishopIntervalSearch}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located interval endpoints, dyadic grid and refinement rows, regular Cauchy readback, result witness data, transport, replay, provenance, and local naming in one interval-search packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalSearch}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalSearch}^{\uparrow}$ packet is generated from located Real endpoints, DyadicRatCore grid and refinement rows, RegSeqRat readback, a result witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8621.1, Definition 8621.2, Definition 8621.3, and Definition 8621.4 seed the finite constructive interval-search route over Real, constructive Cauchy real-line, dyadic grid, regular rational readback, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected points, quotient stream equality, host interval equality, omniscient search, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located endpoint exposure, dyadic grid and refinement exposure, regular Cauchy readback, result-witness correctness, transport stability, provenance, and local naming for BEDC.Derived.BishopIntervalSearchUp.

Chapter 8622

A Concrete Naming Certificate for $\mathsf{UniformCauchyMap}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{UniformCauchyMap}^{\uparrow}$ packet records the finite interface by which a uniformly Cauchy family of displayed maps is read before a $\mathsf{ContinuousMap}^{\uparrow}$ consumer receives any pointwise value or limiting seal. It sits between the displayed sequence-of-maps surface of chapter 8474, the uniform-tail ledger of chapter 3417, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the graph handoff of chapter 130. The packet names only displayed source windows, target readbacks, shared moduli, tolerances, seals, transport, replay, provenance, and local naming; it is not Real completeness, host function equality, a selected limit, quotient streams, trichotomy, or choice.

Definition 8622.1 (Uniform Cauchy map carrier)**.** *A* $\mathsf{UniformCauchyMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, F, W, R, M, D, E, H, C, P, N)$$

with the following visible rows:

1. *X is the source finite-window row, displaying the inspected argument windows and stage windows through* $\mathsf{StreamName}^{\uparrow}$*.*
2. *Y is the target carrier row, naming the displayed codomain read surface for each inspected value.*
3. *F is the displayed family row, routing inspected stage-and-argument pairs through* $\mathsf{FunctionSequence}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows.*
4. *W is the finite window ledger that aligns the source windows with the inspected family stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for target values observed in those windows.*
6. *M is the shared uniform Cauchy modulus row. For every displayed dyadic tolerance request it records one threshold row used by every inspected source window in W.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested dyadic radius, slack, and comparison rows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for value readback, available only after X, Y, F, W, R, M, D have been displayed.*

9. *H records componentwise* hsame *transport for the source windows, target row, family row, window ledger, readback, modulus, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a map-family Cauchy request through $X, Y, F, W, R, M, D, E, H, P, N$.*
11. *P is the* Pkg *provenance over* UniformCauchyMap$^\uparrow$, FunctionSequence$^\uparrow$, ContinuousMap$^\uparrow$, UniformConvergence$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{UniformCauchyMap}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $X, Y, F, W, R, M, D, E, H, C, P, N$. It has no coordinate for ambient Real completeness, host function equality, selected limiting maps, quotient stream equality, trichotomy, or choice.

Theorem 8622.2 (Uniform Cauchy map NameCert obligations)**.** *For every accepted* UniformCauchyMap$^\uparrow$ *carrier $U = (X, Y, F, W, R, M, D, E, H, C, P, N)$, the local* NameCert$_{\text{UniformCauchyMap}^\uparrow}$ *surface consists exactly of carrier admission for the displayed source windows, target row, family row, finite window ledger,* RegSeqRat$^\uparrow$ *readback, shared uniform Cauchy modulus,* DyadicRatCore$^\uparrow$ *tolerance ledger, terminal* Real$^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each tolerance request must read the same threshold row in M for every inspected source window before the target readback in R or the terminal seal E can be consumed.*

Proof. Project the carrier of Definition 8622.1. The analytic rows are X, Y, F, W, R, M, D, E, and the structural rows are H, C, P, N.

The uniform Cauchy condition is carried only by M and D. A requested dyadic tolerance first reads the radius and slack from D, then reads a single threshold row from M that is shared across the displayed source windows in W.

The target readback stays downstream. The row R records only regular rational observations of the inspected values, and the terminal Real$^\uparrow$ seal E appears only after the source windows, family graph rows, modulus row, and tolerance ledger have been displayed.

Transport and replay preserve the same rows through H and C, while P and N source and name the local packet. Since none of the refused principles is a carrier coordinate, the NameCert surface cannot use Real completeness, host function equality, selected limits, quotient streams, trichotomy, or choice. □

Theorem 8622.3 (Uniform Cauchy map ContinuousMap handoff)**.** *Every* ContinuousMap$^\uparrow$ *consumer of an accepted* UniformCauchyMap$^\uparrow$ *carrier must first read*

$$X, Y, F, W, R, M, D, E, H, C, P, N.$$

The handoff exposes source finite-window rows, target RegSeqRat$^\uparrow$ *readbacks, one shared uniform Cauchy modulus row,* DyadicRatCore$^\uparrow$ *tolerance rows, terminal* Real$^\uparrow$ *seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row before any continuous-map graph read is exported. It supplies a finite BEDC route for uniformly Cauchy map families, not Real completeness, host function equality, selected limits, quotient streams, trichotomy, or choice.*

Proof. Start with the NameCert obligations of Definition 8622.2. They force any consumer to expose the source windows, target row, family row, finite window ledger, regular rational readback, shared modulus, tolerance ledger, and terminal seal before the packet is named.

Read the sibling routes in order. The FunctionSequence$^\uparrow$ and ContinuousMap$^\uparrow$ rows expose the displayed map graph, StreamName$^\uparrow$ supplies the inspected source windows, RegSeqRat$^\uparrow$ supplies

the target readback, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance ledger, and $\mathsf{Real}^{\uparrow}$ supplies only the terminal seal reached through those rows.

Use the $\mathsf{UniformConvergence}^{\uparrow}$ sibling as the uniform-tail ledger. The shared row M gives a Cauchy threshold for every displayed source window, so the continuous-map consumer receives a finite map-family Cauchy route rather than a host function-space limit.

Finally H, C, P, N transport, replay, source, and name the same route. No quotient stream equality, selected limiting map, trichotomy oracle, countable-choice schedule, host function equality, or ambient Real completeness coordinate is exported. □

Falsifiable prediction: An accepted $\mathsf{UniformCauchyMap}^{\uparrow}$ consumer that reaches a $\mathsf{ContinuousMap}^{\uparrow}$ graph handoff without reading source windows, target readback, shared modulus, dyadic tolerance, Real seal, hsame transport, Cont replay, Pkg provenance, and NameCert rows refutes the carrier surface.

Independence witness: function-sequence, continuous-map, uniform-convergence, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{UniformCauchyMap}^{\uparrow}$ because this packet jointly carries source finite windows, displayed map-family graph rows, target regular readbacks, a shared uniform Cauchy modulus, dyadic tolerances, terminal Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyMap}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyMap}^{\uparrow}$ packet is generated from source finite-window rows, target readback rows, displayed map-family graph rows, one shared uniform Cauchy modulus, DyadicRatCore tolerance rows, terminal RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8622.1, Definition 8622.2, and Definition 8622.3 seed the finite L10 interface between uniformly Cauchy map families and ContinuousMapUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, host function equality, selected limits, quotient streams, trichotomy, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared uniform Cauchy modulus visibility, target RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal nonescape, ContinuousMap handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyMapUp.

Chapter 8623

A Concrete Naming Certificate for $\mathsf{RealUniformLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, uniform limits of continuous real-valued maps are obtained from explicit Cauchy moduli and finite readback windows, not from an ambient function-space completion or a choice-selected limit function. The $\mathsf{RealUniformLimit}^{\uparrow}$ packet records this theorem as a finite NameCert surface over the real-name interface of chapter 38, the regular rational readback of chapter 301, the finite windows of chapter 126, the continuity carrier of chapter 59, and the uniform-modulus surface of chapter 369. It is a Real and Completion bridge seed: a downstream consumer may read the limit only through the displayed uniform Cauchy family, rowwise continuity data, shared modulus ledger, transport, replay, provenance, and local naming rows.

Definition 8623.1 (RealUniformLimit carrier)**.** *A* $\mathsf{RealUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, F, W, R, M, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the named source-domain row on which all functions in the family are read.*
2. *F is a finite row family of* $\mathsf{Continuous}^{\uparrow}$ *maps whose values are sealed as* $\mathsf{Real}^{\uparrow}$ *names.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the values of every row of F at every displayed source request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows, with one regular rational stream for each displayed value read.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger giving a common Cauchy threshold for all displayed source requests and all sufficiently late rows of F.*
6. *L is the limit-read row: it records the candidate* $\mathsf{Real}^{\uparrow}$ *value read at a source request only after W, R, and M have supplied the shared uniform Cauchy evidence.*
7. *H records componentwise* hsame *transport for the source row, row family, stream windows, regular readbacks, uniform modulus ledger, limit-read row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a source request through the continuous-map row, the stream windows, the regular readback, the uniform modulus ledger, and the limit-read row.*

9. *P records* Pkg *provenance over* $\mathsf{RealUniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host function-space completion, pointwise-only convergence, a selected subsequence, quotient equality of streams, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8623.2 (RealUniformLimit NameCert obligations)**.** *For an accepted* $\mathsf{RealUniformLimit}^{\uparrow}$ *carrier $U = (X, F, W, R, M, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealUniformLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source row,* $\mathsf{Real}^{\uparrow}$*-valued continuous-map family,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{UniformModulus}^{\uparrow}$ *ledger, limit-read row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every limit read must expose the common modulus ledger before it may expose the* $\mathsf{Real}^{\uparrow}$ *limit value.*

Proof. Carrier admission is projection from Definition 8623.1. The packet exposes only the rows $X, F, W, R, M, L, H, C, P, N$, so every local read begins with the displayed source, family, window, readback, modulus, and limit data.

The constructive route is uniform rather than pointwise-only. For a displayed tolerance and source request, the window row W and readback row R expose the late values of the $\mathsf{Continuous}^{\uparrow}$ rows in F, while the ledger M supplies one threshold valid for the whole displayed source scope.

The structural rows then preserve that route. The transport table H compares packets componentwise, the replay row C reruns the same finite route for a consumer, and P, N source and name the certificate. Since no omitted coordinate stores a host completion, selected subsequence, quotient stream equality, countable-choice schedule, or ambient completeness theorem, no such datum can be exported by the NameCert surface. □

Theorem 8623.3 (RealUniformLimit continuity handoff)**.** *Every completion-facing or continuity-facing consumer of an accepted* $\mathsf{RealUniformLimit}^{\uparrow}$ *packet must read the* $\mathsf{Continuous}^{\uparrow}$ *row family F, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger M before it can use the limit row L as a* $\mathsf{Real}^{\uparrow}$*-valued continuous map. The handoff route is therefore*

$$X, F, W, R, M \longmapsto L, H, C, P, N.$$

It exports the constructive uniform-limit theorem only through the displayed finite rows, and not through pointwise convergence alone, host function-space completeness, countable choice, quotient stream equality, or a selected limit representative.

Proof. Start with the NameCert obligations of Definition 8623.2. They force every limit read to pass through the displayed family, windows, readbacks, and shared modulus ledger.

Continuity is handed off rowwise and then uniformly. Each row of F supplies its $\mathsf{Continuous}^{\uparrow}$ replay for the displayed source request, while W and R expose the corresponding $\mathsf{Real}^{\uparrow}$ value names as regular rational readbacks.

The ledger M is the uniform step. It supplies the common late threshold used to compare all displayed rows at the requested tolerance, so the candidate limit row L is read only after the same threshold covers the whole source scope.

Finally, H, C, P, N transport, replay, source, and name the handoff. A consumer that bypasses F, W, R, or M is outside the accepted packet, and a consumer that asks for pointwise-only convergence, a host completion, countable choice, quotient stream equality, or a selected representative asks for data not carried by the certificate. □

Theorem 8623.4 (RealUniformLimit ledger exactness)**.** *Every accepted* $\mathsf{RealUniformLimit}^{\uparrow}$ *limit read is ledgered by the single packet* $X, F, W, R, M, L, H, C, P, N$*. The same ledger records the Real-valued continuous rows, stream windows, regular rational readbacks, shared uniform Cauchy threshold, limit-read row, transport, replay, provenance, and local naming; it records no pointwise-only convergence row, host limit selector, quotient stream class, countable-choice schedule, or ambient function-space completion.*

Proof. Use the carrier of Definition 8623.1. Its analytic rows are exactly X, F, W, R, M, L, and its structural rows are exactly H, C, P, N.

The NameCert obligations of Definition 8623.2 make these rows the complete public surface. A limit read has to display the continuous row family, the value windows, the rational readbacks, and the uniform ledger before the limit row is available.

The continuity handoff of Definition 8623.3 then identifies the consumer route through that same packet. Since the packet has no additional coordinate, the excluded pointwise, host-completion, selector, quotient, and choice data cannot be recovered from an accepted read. □

Falsifiable prediction: Every accepted RealUniformLimitUp read displays Continuous, StreamName, RegSeqRat, UniformModulus, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a completion-facing or continuity-facing consumer uses the limit packet.

Independence witness: real, regseqrat, streamname, continuous, uniform-modulus: $\mathsf{RealUniformLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores a $\mathsf{Real}^{\uparrow}$-valued continuous-row family, finite value windows, regular rational readback, one shared uniform Cauchy ledger, limit-read data, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformLimit}^{\uparrow}$ packet is generated from a displayed source row, $\mathsf{Real}^{\uparrow}$-valued continuous-map rows, finite stream windows, regular rational readback, a shared uniform Cauchy ledger, a limit-read row, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8623.1, Definition 8623.2, Definition 8623.3, and Definition 8623.4 seed the constructive uniform-limit route over Real, RegSeqRat, StreamName, Continuous, UniformModulus, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean carrier, a checked standard bridge, pointwise-only convergence, host function-space completeness, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires a Lean carrier and checked NameCert obligations for BEDC.Derived.RealUniformLimitUp.

Chapter 8624

A Concrete Naming Certificate for $\mathsf{MooreSmithDirectedNet}^{\uparrow}$

Following Moore and Smith, the $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ packet records a directed net through finite observation windows rather than through a host function space of all indexed values. It sits between the directed-index substrate of chapter 1607, the Moore–Smith convergence surface of chapter 1521, the net-convergence interface of chapter 1667, the filter-base handoff of chapter 1601, the Cauchy-filter completion boundary of chapter 1569, and the finite stream-window surface of chapter 126. The packet gives topology, completion, and filter consumers a shared directed-net carrier; it does not supply host choice, quotient net equality, selected limits, ambient completeness, or an untracked cofinal selector.

Definition 8624.1 (Moore–Smith directed-net carrier)**.** *A* $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, V, R, W, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row, naming finite directed addresses, displayed preorder comparisons, finite request windows, and upper-window witnesses.*
2. *V is the directed value-window row, recording the values visible at the displayed addresses of D.*
3. *R is the cofinal-refinement row, recording a finite refinement request and the displayed upper-window route by which the refined addresses remain cofinal in the observed window.*
4. *W is the finite observation-window row, naming the convergence, filter-base, and stream-window requests that a consumer may inspect.*
5. *L is the convergence ledger, recording the handoff through* $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *before any completion-facing read is public.*
6. *H records componentwise* hsame *transport for the directed-index data, value windows, cofinal refinement, observation windows, convergence ledger, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an observation request through D, V, R, W, L.*
8. *P is the* Pkg *provenance over* $\mathsf{MooreSmithDirectedNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithDirectedNet}^{\uparrow}}$ *row.*

The carrier exports exactly these finite directed-net rows. It contains no host choice functional, quotient net equality, selected limit point, ambient topology, or hidden completion oracle.

Theorem 8624.2 (Moore–Smith directed-net NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ *carrier $M = (D, V, R, W, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithDirectedNet}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the directed-index row, value-window row, cofinal-refinement row, finite observation-window row, convergence ledger, transport row, replay row, provenance row, and local naming row;*
2. *directed-index readability, requiring every net read to expose the* $\mathsf{DirectedSet}^{\uparrow}$ *row D before a value in V is consumed;*
3. *cofinal-refinement compatibility, requiring every refined tail request in R to display its upper-window witness inside the same directed ledger;*
4. *finite-window convergence routing, requiring every* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *and* $\mathsf{NetConvergence}^{\uparrow}$ *read to pass through the observation windows W and ledger L before it is exported;*
5. *filter and completion handoff, requiring* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumers to read only the finite tail family recorded by D, R, W, L;*
6. *stream replay, requiring every* $\mathsf{StreamName}^{\uparrow}$ *observation to be replayed through C and transported through H without changing the displayed rows;*
7. *ledger refusal, excluding host choice, quotient net equality, selected limits, ambient completeness, and unrecorded cofinal schedules.*

Proof. Project the accepted carrier of Definition 8624.1. Its analytic rows are D, V, R, W, L, and its structural rows are H, C, P, N.

The directed-index obligation is inherited from the visible $\mathsf{DirectedSet}^{\uparrow}$ row. A consumer first reads the displayed addresses, comparisons, finite windows, and upper-window witnesses in D; only then can the value-window row V be inspected.

The cofinal-refinement row R is not an external selector. It records a finite refinement request together with the upper-window route already supplied by D, so every refined tail remains inside the displayed directed ledger.

The convergence ledger L is reached through the finite observation windows W. Thus $\mathsf{MooreSmithConvergence}^{\uparrow}$ and $\mathsf{NetConvergence}^{\uparrow}$ consumers receive a displayed observation route, while $\mathsf{FilterBase}^{\uparrow}$ and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumers receive only the finite tail family recorded by that same route.

Finally H and C transport and replay the same rows componentwise, while P and N package and name them locally. Since no coordinate stores host choice, quotient net equality, selected limits, ambient completeness, or an unrecorded cofinal schedule, the listed obligations exhaust the certificate surface. □

Theorem 8624.3 (Moore–Smith directed-net finite-window handoff)**.** *Every topology-facing or completion-facing consumer of an accepted* $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ V,\ R,\ W \longmapsto \mathsf{MooreSmithConvergence}^{\uparrow},\ \mathsf{NetConvergence}^{\uparrow},\ \mathsf{FilterBase}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{Str}$$

The handoff gives the consumer directed index data, cofinal refinement, finite observation windows, convergence ledger rows, componentwise transport, displayed replay, provenance, and local naming.

It does not give host choice, quotient net equality, selected limits, ambient compactness, or a countable schedule outside the packet.

Proof. Start with the NameCert obligations of Definition 8624.2. They force directed-index readability through D and finite value-window exposure through V before any downstream route is public.

Next read the cofinal-refinement and observation-window rows. The row R keeps refined tails inside the displayed upper-window ledger, and W lists the finite requests visible to convergence, filter, completion, and stream consumers.

Apply the convergence ledger row L. It orders the public handoff through $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ without replacing the finite packet by a host topology or by a selected endpoint.

The structural rows H, C, P, N preserve, replay, source, and name this same carrier. Hence every consumer receives the finite directed-net handoff and no forbidden host choice, quotient net equality, selected limit, ambient compactness, or external countable schedule. □

Falsifiable prediction: A claimed Moore–Smith directed-net handoff falls outside $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ whenever it reaches $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ without first displaying the directed index row, cofinal-refinement row, finite observation-window row, convergence ledger, componentwise transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithDirectedNet}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithDirectedNet}^{\uparrow}$ packet is generated from directed index data, finite value windows, a cofinal-refinement row, finite observation windows, a convergence ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithDirectedNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8624.1, Definition 8624.2, and Definition 8624.3 seed the finite-window Moore–Smith directed-net interface consumed by convergence, filter, completion, and stream-window chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithDirectedNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host choice, quotient net equality, selected limits, ambient completeness, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-index readability, cofinal-refinement compatibility, finite-window convergence routing, filter and completion handoff, stream replay, transport preservation, and ledger refusal for the displayed packet.

Chapter 8625

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchyMap}^{\uparrow}$

Following Bishop and Bridges on uniform convergence and finite-net compact metric arguments, the $\mathsf{CompactUniformCauchyMap}^{\uparrow}$ packet records a compact-domain family of uniformly Cauchy continuous-map rows as finite NameCert data. It sits over the compact metric net surface of chapter 129, the continuous-map modulus rows of chapter 130, the uniform-Cauchy family route of chapter 8622, the regular rational readback of chapter 301, and the terminal Real seals of chapter 38.

The packet names the finite handoff from compact-domain net control to a uniform-Cauchy map family. It is not an open-cover compactness principle, a selected limiting map, a quotient equality of functions, a choice operator, an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification, or bridge checking.

Definition 8625.1 (Compact uniform Cauchy map carrier)**.** *A* $\mathsf{CompactUniformCauchyMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, F, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the compact-net datum for the domain, including the finite* $\mathsf{CompactMetric}^{\uparrow}$ *cover windows, Lebesgue-style mesh requests, and the metric comparison rows used to inspect the domain.*
2. *F stores the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows for the displayed family members, including source windows, target tolerances, and the local continuity budgets consumed at each compact-net point.*
3. *U is the* $\mathsf{UniformCauchyMap}^{\uparrow}$ *family row, requiring all family comparisons to use one displayed tolerance schedule across the compact-domain windows named by A.*
4. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for every finite tolerance, oscillation, and image-distance request exposed by A, F, U.*
5. *E stores the terminal* $\mathsf{Real}^{\uparrow}$ *seals for the finite image-distance bounds and Cauchy comparison rows that downstream consumers may read.*
6. *H records componentwise* hsame *transport for compact-net data, continuous-map moduli, uniform-Cauchy family rows, regular rational readbacks, Real seals, continuation, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from compact-net requests through continuous-map modulus rows, then through the uniform-Cauchy family row, and finally into the readback and Real seal rows.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCauchyMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformCauchyMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, F, U, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected limits, quotient equality of functions, choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8625.2 (CompactUniformCauchyMap NameCert obligations)**.** *For every accepted* CompactUniformCauc *carrier*

$$K = (A, F, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyMap}^{\uparrow}}$ *surface consists exactly of compact-net admission for A, continuous-map modulus admission for F, uniform-Cauchy family admission for U, regular rational readback through R, Real seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

No compact uniform Cauchy map obligation is stored outside these rows.

Proof. Project the accepted carrier. The compact-domain information available to the packet is exactly the finite net row A, so every later modulus request is tied to displayed compact metric windows rather than to an open-cover principle.

The analytic rows are then forced. The continuous-map data F supplies the local modulus budgets, while U requires the family comparison to use one tolerance schedule across the compact-domain windows named by A.

The readback and seal rows close the public surface. The row R records the regular rational observations used by the compact-net, modulus, and uniform-Cauchy routes, and E is visible only after those rows have been replayed.

Finally inspect the structural rows. The transport, continuation, provenance, and local NameCert rows preserve the same finite route, and the carrier has no coordinate for open-cover compactness, selected limiting maps, quotient function equality, choice, ambient completeness, Lean verification, or bridge checking. □

Theorem 8625.3 (CompactUniformCauchyMap uniform-modulus handoff)**.** *Let $K = (A, F, U, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompactUniformCauchyMap}^{\uparrow}$ *carrier. Every downstream uniform-modulus read from K factors through the finite route*

$$A, F \longrightarrow U \longrightarrow R \longrightarrow E$$

with transport, replay, provenance, and local naming supplied by H, C, P, N. Thus the exported modulus is a compact-net uniform-Cauchy handoff, not a selected limit map or an equality of quotient functions.

Proof. Start with a downstream uniform-modulus read. Carrier admission exposes the compact-net row A and the continuous-map modulus row F before any family-level comparison is available.

The uniform-Cauchy row U then consumes those same compact-domain windows and modulus budgets. Because U stores one family tolerance schedule over the displayed finite net data, the handoff is uniform over the named compact-domain requests.

The readback route is finite. The row R records the regular rational bounds used in the compact-net and uniform-Cauchy comparison, and the terminal Real seals in E may be read only after that route is displayed.

The support rows H, C, P, N transport and replay the route without adding another analytic coordinate. Hence the public handoff excludes open-cover compactness, selected limits, quotient function equality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyMap}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCauchyMap}^{\uparrow}$ packet is generated from compact-net data, ContinuousMapUp modulus rows, one UniformCauchyMapUp family row, RegSeqRat readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8625.1, Definition 8625.2, and Definition 8625.3 seed the compact-domain uniform-Cauchy map packet from compact-net rows, continuous-map modulus rows, uniform-Cauchy family rows, regular rational readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient function equality, choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-net admission, continuous-map modulus admission, uniform-Cauchy family exactness, regular rational readback, Real seal nonescape, transport stability, replay stability, and TasteGate field faithfulness for BEDC.Derived.CompactUniformCauchyMapUp.

Chapter 8626

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$

Following Bourbaki and Kelley on completions as reflector data, and Bishop–Bridges on constructive Cauchy completion, the $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$ packet records the finite unit leg by which a displayed source Cauchy object enters the completion-reflector surface. It sits between the reflector packet of chapter 1585, the universal mapping surface of chapter 1456, the regular-Cauchy reflector row of chapter 3398, and the Real completion adjunction packet of chapter 6918. The chapter names only the unit embedding and its dense-image replay; it does not assert quotient completion, selected representatives, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8626.1 (Cauchy completion reflector unit carrier)**.** *A* $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, R, J, D, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric and Cauchy-observation row, including the finite source histories consumed before a completion request is public.*
2. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *row that exposes the target completion-reflector surface over the same source histories.*
3. *J is the unit embedding row from the source row into the reflector row, read as displayed finite data rather than as a quotient-class injection.*
4. *D is the dense unit-image row, requiring reflector-facing reads to expose the source image before universal mapping or Real-completion consumers can use the completed surface.*
5. *T is the triangle replay row, tying the unit read to the adjunction and Real-completion rows named by chapter 1456 and chapter 6918.*
6. *H records componentwise* hsame *transport for source, reflector, unit embedding, dense-image, triangle replay, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through*

$$X \longmapsto J \longmapsto R \longmapsto D, T \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, RegularCauchyComp CauchyCompletionAdjunction$^{\uparrow}$, RealCompletionAdjunction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, R, J, D, T, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8626.2 (CauchyCompletionReflectorUnit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$ *carrier $U = (X, R, J, D, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorUnit}}$ *surface consists exactly of carrier admission for the source row, reflector row, unit embedding, dense unit image, triangle replay, componentwise transport, displayed continuation, provenance, and local naming rows.*

Every reflector-unit consumer must read $X, J, R, D, T, H, C, P, N$ in the displayed order before it exports a completion-facing handoff. The packet exports no quotient completion, selected representative, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the accepted carrier of Definition 8626.1. The tuple exposes the source row X, reflector row R, unit row J, dense-image row D, triangle replay row T, and structural rows H, C, P, N.

The unit embedding is read before the reflector target is used. A completion-facing consumer therefore obtains source access through J, then reads the reflector row R, and only after that exposes the dense unit-image row D.

The triangle row is a replay row. It refers back to the displayed adjunction and Real-completion adjunction packets through P, while H and C transport and replay only the same finite coordinates. Since no refused host-side principle appears as a row of the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8626.3 (CauchyCompletionReflectorUnit triangle law)**.** *Let $U = (X, R, J, D, T, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$ *carrier. Any unit-side read consumed by the Cauchy completion adjunction or the Real completion adjunction factors through*

$$X \longmapsto J \longmapsto R \longmapsto D \longmapsto T \longmapsto (H, C, P, N).$$

The triangle law is only this finite replay condition over the displayed source, reflector, unit, dense-image, transport, continuation, provenance, and local NameCert *rows. It does not identify quotient completions, select representatives, invoke countable choice, assert host equality, or claim ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8626.2. They force every accepted read to display the source, unit embedding, reflector, dense-image, triangle, transport, continuation, provenance, and naming rows.

The adjunction-side consumer can then use the same route only through the cited sibling packets. The Cauchy completion adjunction supplies the universal mapping surface, the regular-Cauchy reflector supplies the regular source specialization, and the Real completion adjunction supplies the terminal Real-facing unit/counit boundary.

The row T records the replay through those surfaces, while H, C, P, N preserve and name the same finite packet. Because the carrier has no quotient, representative-selection, choice, host-equality, or ambient-completeness coordinate, the triangle law is exactly the displayed BEDC replay. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$ packet is generated from source Cauchy data, a completion-reflector row, a unit embedding, dense unit-image evidence, triangle replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8626.1, Definition 8626.2, and Definition 8626.3 seed the finite unit leg for the L10 Cauchy completion reflector route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit embedding exposure, dense-image visibility, triangle replay exactness, transport stability, continuation replay, provenance exhaustion, and local naming for BEDC.Derived.CauchyCompletionReflectorUnitUp.

Chapter 8627

A Concrete Naming Certificate for $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$

Following Bridson and Haefliger, the $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$ packet records the finite BEDC surface on which complete metric service and displayed geodesic convexity are read together. It sits between the metric carrier of chapter 57, the completion boundary of chapter 128, the segment witness route of chapter 2213, the finite convexity packet of chapter 7899, and the complete CAT(0) neighbour of chapter 6734. The packet names a complete geodesic-convex carrier for downstream metric geometry consumers; it does not assert open-cover compactness, selected global geodesics, quotient path equality, projection existence, or host-geometric primitives.

Definition 8627.1 (Complete geodesic convexity surface)**.** *The* $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$ *NameCert surface has five displayed rows:*

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing point classifiers, distance readback, and the metric-law ledger inspected by the packet.*
2. *E is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording only the displayed Cauchy-thread or completion-facing requests available over the same source.*
3. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *row, carrying finite segment witnesses between displayed endpoints.*
4. *V is the* $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ *row, requiring each displayed segment read to remain inside the named convex target through finite endpoint, segment, and membership ledgers.*
5. *S is the structural row consisting of componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompleteGeodesicConvexity}^{\uparrow}}$ *naming row over M, E, G, V.*

The classifier compares accepted packets only through M, E, G, V, S. It has no coordinate for open-cover compactness, selected geodesic families, quotient path spaces, projection selectors, ambient CAT(0) theorems, or host completeness outside the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Theorem 8627.2 (Complete geodesic convexity carrier)**.** *For every accepted* $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$ *packet*

$$C = (M, E, G, V, S),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteGeodesicConvexity}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Met}^{\uparrow}$ *row M,* $\mathsf{CompleteMetric}^{\uparrow}$ *row E,* $\mathsf{GeodesicMetric}^{\uparrow}$ *segment row G,* $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$ *convexity row V, and structural row S. Every completion-facing or Hadamard-facing consumer must read the complete metric service and the segment-inside-convex-target witness through those five rows before using the packet as complete geodesic convexity data.*

Proof. Project the displayed carrier $C = (M, E, G, V, S)$. The metric source is fixed by M, while completion enters only through the $\mathsf{CompleteMetric}^{\uparrow}$ service row E over that same source.

Read the geodesic and convexity rows next. The row G supplies only finite segment witnesses between displayed endpoints, and V verifies that the visible segment read remains inside the named convex target by the finite convexity surface of chapter 7899.

Finally inspect the structural row S. Componentwise hsame transport and displayed Cont replay preserve exactly M, E, G, V, while Pkg provenance and the local naming row source the same packet. Because no additional coordinate stores open-cover compactness, selected geodesic families, quotient path equality, projection selectors, ambient CAT(0) theorems, or host-geometric primitives, the five displayed rows exhaust the NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$ consumer that reaches completion-facing convexity data without displaying the $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{FiniteGeodesicConvexity}^{\uparrow}$, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: metric-space, complete-metric, geodesic-metric, finite-geodesic-convexity, hadamard-space: none of the listed neighbours alone supplies the whole packet $C = (M, E, G, V, S)$, because the neighbours contribute metric source data, completion service, segment witnesses, convexity ledgers, or complete CAT(0) context while this chapter binds the complete metric service and finite geodesic-convexity route into one finite NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteGeodesicConvexity}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteGeodesicConvexity}^{\uparrow}$ packet is generated from a MetricSpace source row, a CompleteMetric service row, a GeodesicMetric segment row, a FiniteGeodesicConvexity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteGeodesicConvexity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8627.1 and Definition 8627.2 seed the finite complete geodesic-convex carrier over metric, completion, segment, convexity, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteGeodesicConvexity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected global geodesics, quotient path equality, projection existence, ambient CAT(0) theorems, host-geometric primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric exposure, geodesic segment exactness, convexity routing, transport, replay, provenance, and local naming for BEDC.Derived.CompleteGeodesicConvexityUp.

Chapter 8628

A Concrete Naming Certificate for $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$ packet records a finite family of regular Cauchy rational names controlled by one shared modulus schedule. It sits over the dyadic rational core of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the shared-modulus surface of chapter 1741, and the family-level route of chapter 1555. The packet gives completion and uniform-limit consumers a bounded shared-modulus regular-family certificate before any quotient stream, host real equality, countable-choice selector, chosen limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 8628.1 (Uniform Cauchy regular family carrier)**.** *A* $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family row whose entries are* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy names and whose domain of inspection is a finite BEDC row.*
2. *M is the shared* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *schedule. It assigns one displayed modulus response to the whole family for each accepted tolerance request.*
3. *W records the* $\mathsf{StreamName}^{\uparrow}$ *windows read from every family member at the requests named by M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the requested tolerance, the shared tail tolerance supplied by M, and the finite endpoint ledgers used for each component comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family assembled from the windows W and dyadic ledger D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-row family reached only after F, M, W, D, R have been exposed.*
7. *H records componentwise* hsame *transport for the family row, shared modulus, stream windows, dyadic tolerance ledger, regular readbacks, real seals, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a family Cauchy request through $F, M, W, D, R, E, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$, $\mathsf{UniformCauchyFamily}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformCauchyRegularFamily}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient streams, host real equality, per-component hidden moduli, countable-choice schedules, chosen limiting reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8628.2 (Uniform Cauchy regular family NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$ *carrier*

$$U = (F, M, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyRegularFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite regular family, shared modulus schedule, stream windows, dyadic tolerance ledgers, regular readbacks, real seals, componentwise transport, replay, provenance, and local naming; shared-modulus exactness requiring M to serve every displayed family member; readback exactness requiring R to be assembled only from W and D; real-seal nonescape requiring E to remain downstream of F, M, W, D, R; and provenance exhaustion excluding quotient streams, host real equality, hidden per-component moduli, countable choice, chosen limits, and ambient completeness.*

Proof. Project the carrier of Definition 8628.1. The finite family row, shared modulus, stream windows, dyadic tolerance ledger, regular readbacks, real seals, transport, replay, provenance, and local name are the only displayed coordinates.

The shared-modulus clause is forced by the single row M. Every accepted tolerance request for the family reads the same schedule before any component window in W is inspected, so no component receives a hidden threshold outside the packet.

The readback and seal clauses are downstream of this same data. The row R is assembled from W and D, and the real seal row E appears only after the regular readback has been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no further coordinate, the listed obligations exhaust the NameCert surface and rule out the refused quotient, equality, choice, limit, and completeness routes. □

Theorem 8628.3 (Uniform Cauchy regular family real-completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$ *carrier factors through the finite route*

$$F, M \longmapsto W, D \longmapsto R \longmapsto E.$$

Thus the consumer receives one finite regular family row, one shared $\mathsf{UniformCauchyModulus}^{\uparrow}$ *schedule,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows before any completion-facing output can be read.*

Proof. Begin with the source side. The family row F and shared modulus row M are the only input rows available to a completion-facing consumer, and M is common to all displayed family members.

Next read the observation rows. The stream windows W and dyadic tolerance ledger D provide the finite component comparisons required by the shared modulus, while R records the regular rational readbacks obtained from those observations.

The real seal row E is downstream of R. Componentwise transport and displayed replay preserve this order, and provenance records only the rows named in Definition 8628.1.

Therefore completion-facing consumers must use the displayed route $F, M \mapsto W, D \mapsto R \mapsto E$. The route does not export quotient stream equality, host real equality, per-component hidden moduli, countable choice, chosen limiting reals, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyRegularFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyRegularFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8628.1, Definition 8628.2, and Definition 8628.3 seed the finite shared-modulus regular Cauchy family route over $\mathsf{UniformCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyRegularFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-stream completion, host real equality, hidden per-component moduli, countable choice, chosen limiting reals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, readback exactness, real-seal nonescape, replay stability, provenance exhaustion, and local naming for BEDC.Derived.UniformCauchyRegularFamilyUp.

Constructive story: A $\mathsf{UniformCauchyRegularFamily}^{\uparrow}$ packet is generated from a finite family of RegSeqRat rows, one UniformCauchyModulus schedule, StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readbacks, RealUp seal rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 8629

A Concrete Naming Certificate for $\mathsf{RegularCauchyOrderedField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ packet records the constructive ordered-field surface for regular Cauchy real names as finite BEDC NameCert data. It sits over the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the algebraic order boundary of chapter 766, and the terminal real-name seal of chapter 38. The packet gives RealAlgOrder and Real consumers a displayed comparison and operation route on regular names, without quotient streams, trichotomy, selected limits, host ordered-field equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8629.1 (Regular Cauchy ordered-field carrier)**.** *A* $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, S, D, R, O, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source rows whose finite rational windows are compared.*
2. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule naming every inspected precision window for both sources.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, sign budget, and comparison threshold for each inspected window.*
4. *R is the regular-rational readback row that records the finite rational observations obtained from X and Y through S and D.*
5. *O is the operation-order ledger, storing the displayed addition, multiplication, comparison, and apartness rows needed by the ordered-field route.*
6. *A is the downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff row, available only after X, Y, S, D, R, O have been displayed.*
7. *H records componentwise* hsame *transport for the regular sources, stream schedule, dyadic ledger, readback row, operation-order ledger, RealAlgOrder handoff, and local certificate rows.*
8. *C records the displayed* Cont *replay from an ordered-field request to the stream windows, dyadic tolerance comparison, regular-rational readback, operation-order ledger, and RealAlgOrder handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOrderedField}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOrderedField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, D, R, O, A, H, C, P, N$. It has no coordinate for quotient stream equality, trichotomy, selected limits, host ordered-field equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8629.2 (Regular Cauchy ordered-field NameCert obligations)**.** *For every accepted* RegularCauchyOrdered *carrier $F = (X, Y, S, D, R, O, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOrderedField}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparisons, regular-rational readback, operation-order ledger, downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient streams, trichotomy, selected limits, host ordered-field equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8629.1. The rows $X, Y, S, D, R, O, A, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The two regular Cauchy sources are compared through the same stream schedule S and dyadic tolerance ledger D. The readback row R and the operation-order ledger O therefore record finite tolerance evidence, not quotient-stream equality or a host trichotomy.

The RealAlgOrder handoff A is downstream of that finite route. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and the carrier has no additional coordinate from which a selector, host equality, choice principle, or ambient completeness theorem could be projected. □

Theorem 8629.3 (Regular Cauchy operation-order handoff)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ *packet reads the constructive ordered-field surface only through the displayed route*

$$X, Y, S, D, R, O, A, H, C, P, N.$$

The handoff exports finite stream windows, dyadic tolerance comparisons, regular-rational readback, an operation-order ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It does not export quotient streams, trichotomy, selected limits, host ordered-field equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the ordered-field surface. The carrier displays X and Y only together with the shared stream schedule S, dyadic tolerance ledger D, and finite readback row R.

The operation-order ledger O can be read only after those windows and tolerances have been displayed. If an algebraic-order or Real-facing consumer is present, the downstream row A is available only after the same finite comparison route has been consumed.

Apply the NameCert obligations of Definition 8629.2. The support rows H, C, P, N preserve, replay, source, and name this finite route, so the handoff exports no quotient streams, trichotomy, selected limits, host ordered-field equality, choice principle, or ambient completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ consumer that reaches a $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read without first reading two RegSeqRat source rows, a shared StreamName schedule, a

DyadicRatCore tolerance ledger, regular-rational readback, an operation-order ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: regseqrat, dyadicratcore, streamname, realalgorder, real: $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular Cauchy source rows to one constructive ordered-field ledger and downstream RealAlgOrder handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOrderedField}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOrderedField}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName schedule, a DyadicRatCore tolerance ledger, regular-rational readback, an operation-order ledger, a RealAlgOrder handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrderedField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8629.1, Definition 8629.2, and Definition 8629.3 seed the finite constructive ordered-field route for regular Cauchy real names through RegSeqRat, DyadicRatCore, StreamName, RealAlgOrder, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrderedField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, trichotomy, selected limits, host ordered-field equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and RealAlgOrder handoff theorem.

Chapter 8630

A Concrete Naming Certificate for FiniteSubcoverModulusBudget$^{\uparrow}$

Following Bishop and Bridges, the FiniteSubcoverModulusBudget$^{\uparrow}$ packet records the finite budget ledger by which compact-net cover data are folded into a uniform modulus. It sits over the compact finite-net route of chapter 60, the finite lower-bound fold of chapter 619, the compact cover route of chapter 6250, the finite subcover criterion of chapter 3503, and the uniform-modulus consumers in chapter 9237. The packet records a finite ledger for compact-net, rational-radius, local-modulus, fold-minimum, and uniform-modulus rows; it does not select arbitrary subcovers, invoke host compactness, or claim ambient completeness.

Definition 8630.1 (Finite subcover modulus budget carrier)**.** *A* FiniteSubcoverModulusBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, E, L, R, F, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-net source row, exposing the compact metric cover data used by the budget ledger.*
2. *E is the bundled-center row, naming the finite probe bundle for the requested tolerance.*
3. *L is the rational-radius row, recording the displayed radius data for each bundled center.*
4. *R is the local-modulus row, recording the pointwise modulus rows attached to the displayed centers.*
5. *F is the fold-minimum row, recording the finite lower-bound fold used to aggregate the bundled radii.*
6. *U is the uniform-modulus row, exporting the displayed tolerance-to-precision ledger only after the fold minimum is available.*
7. *H records componentwise* hsame *transport for the compact source, bundle, radii, local moduli, fold minimum, uniform modulus, and local certificate rows.*
8. *C records displayed* Cont *replay from the compact budget request through K, E, L, R, F, U, H.*
9. *P is the* Pkg *provenance over* FiniteSubcoverModulusBudget$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* ProbeBundle$^{\uparrow}$*,* Rat$^{\uparrow}$*,* CFun$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, E, L, R, F, U, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, host compactness, or ambient completeness.

Theorem 8630.2 (Finite subcover modulus budget NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}$ *carrier $B = (K, E, L, R, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverModulusBudge}}$ *surface consists exactly of carrier admission for the compact-net source row, bundled-center row, rational-radius row, local-modulus row, fold-minimum row, uniform-modulus row, componentwise transport, displayed replay, provenance, and local naming. No NameCert obligation may replace these rows by arbitrary subcovers, selected representatives, host compactness, or ambient completeness.*

Proof. Project the carrier of Definition 8630.1. The rows $K, E, L, R, F, U, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The compact-net source row K and bundled-center row E determine the finite cover budget. The rational-radius row L and local-modulus row R keep the pointwise data visible before any uniform modulus is read.

The fold-minimum row F is the only aggregate row, and it is read before the uniform-modulus row U. Transport, replay, provenance, and local naming are supplied by H, C, P, N, so the carrier has no coordinate from which a selected subcover or ambient completeness theorem could be projected. □

Theorem 8630.3 (Finite subcover modulus budget uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}$ *packet reads the finite cover budget only through the displayed route*

$$K, E, L, R, F, U, H, C, P, N.$$

The handoff exports compact-net rows, bundled centers, rational radii, local moduli, a fold minimum, a uniform-modulus row, componentwise transport, displayed replay, provenance, and local naming. It does not export arbitrary subcovers, host compactness, or ambient completeness.

Proof. Start with the compact-net source row K and the bundled-center row E. The budget ledger can only be read from displayed center data, so no consumer can bypass the finite cover route.

The rational-radius row L and local-modulus row R supply the local pointwise data, while the fold-minimum row F aggregates the finite radii into the displayed lower-bound ledger. The uniform-modulus row U is available only after that fold has been read.

Apply Definition 8630.2. The structural rows H, C, P, N preserve and replay the same finite route, so the handoff exports no arbitrary subcovers, host compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}$ packet is generated from a compact-net source row, bundled centers, rational radii, local moduli, a fold minimum, a uniform-modulus row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8630.1, Definition 8630.2, and Definition 8630.3 seed the finite compact-net budget route consumed by the compact-uniform-continuity vision.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary subcovers, selected representatives, host compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bundled-center exposure, rational-radius exactness, local-modulus exactness, fold-minimum exactness, uniform-modulus export, transport stability, replay stability, and non-escape proofs for BEDC.Derived.FiniteSubcoverModulusBudgetUp.

Chapter 8631

A Concrete Naming Certificate for $\mathsf{BishopModulatedCauchySequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopModulatedCauchySequence}^{\uparrow}$ packet records the finite route by which a displayed sequence and an explicit Cauchy modulus become a completion-facing real name. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the monotone-modulus normalization boundary of chapter 1744, and the terminal real-name seal of chapter 38. The packet names explicit modulus data for Phase Real and Completion consumers; it does not quotient streams, select a limit, invoke countable choice, compare host reals, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8631.1 (BishopModulatedCauchySequence carrier)**.** *A* $\mathsf{BishopModulatedCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence row, exposing the inspected rational or dyadic approximants only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *M is the explicit Cauchy modulus row, giving a tolerance-to-tail schedule rather than an external convergence oracle.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger used to compare the tail entries requested by M.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the sequence entries inspected for a requested tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads the displayed windows as regular rational approximation data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the sequence row, modulus row, dyadic ledger, finite windows, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for the sequence, modulus, dyadic, window, regular-readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay in the order $S, M, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* BishopModulatedCauchySequence$^{\uparrow}$, CauchyModulusMonotoneNormalization$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopModulatedCauchySequence}^{\uparrow}}$ *row naming this packet and no larger completion principle.*

The carrier has no coordinate for quotient sequence equality, selected limits, countable choice, host equality of reals, trichotomy, or ambient completeness of Real$^{\uparrow}$.

Theorem 8631.2 (BishopModulatedCauchySequence NameCert obligations)**.** *For an accepted* BishopModulatedCauchySequence$^{\uparrow}$ *carrier $B = (S, M, D, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{BishopModulatedCauch}}$ *surface consists exactly of carrier admission for the sequence row, explicit Cauchy modulus row, dyadic-radius ledger, finite* StreamName$^{\uparrow}$ *window row,* RegSeqRat$^{\uparrow}$ *handoff, terminal* Real$^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. Every consumer read of the real-facing boundary must display $S, M, D, W, R, E, H, C, P, N$ in that order, and no obligation is stored outside those rows.*

Proof. Carrier admission is projection from Definition 8631.1. The tuple displays the sequence row, explicit modulus, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

The modulus row is read before the dyadic ledger. The normalization boundary of chapter 1744 keeps this row as an explicit tolerance-to-tail schedule, while DyadicRatCore$^{\uparrow}$ supplies only the displayed radius comparisons needed by that schedule.

The window and readback rows then route through StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$. The terminal Real$^{\uparrow}$ seal is downstream of those finite rows, and the structural rows H, C, P, N preserve the same displayed order. Since none of these rows contains quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, or ambient real completeness, the listed obligations exhaust the local certificate. □

Theorem 8631.3 (BishopModulatedCauchySequence regular route)**.** *Every real-facing read exported by an accepted* BishopModulatedCauchySequence$^{\uparrow}$ *carrier factors through the explicit route*

$$S \longmapsto M \longmapsto D \longmapsto W \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

Thus the RegSeqRat$^{\uparrow}$ *handoff is available only after the* CauchyModulusMonotoneNormalization$^{\uparrow}$ *modulus surface, the* DyadicRatCore$^{\uparrow}$ *radius ledger, and the finite* StreamName$^{\uparrow}$ *windows have been displayed; the* Real$^{\uparrow}$ *seal cannot be used as a prior source of convergence.*

Proof. Apply Definition 8631.2. It requires the sequence row S and explicit modulus row M before any radius or tail comparison can be exported.

The dyadic and stream rows D, W supply the finite comparisons requested by the modulus. The regular rational row R reads those displayed windows only after the tolerance ledger and finite observation schedule are already visible.

Finally the seal row E and structural rows H, C, P, N package the same finite route. Because the carrier excludes quotient streams, selected limits, countable choice, host equality, trichotomy, and ambient Real$^{\uparrow}$ completeness, a consumer cannot bypass the regular route by appealing to a completed real in advance. □

Falsifiable prediction: Every accepted BishopModulatedCauchySequence$^{\uparrow}$ real-facing read exposes a sequence row, explicit Cauchy modulus row, DyadicRatCore radius ledger, StreamName

finite-window row, RegSeqRat handoff, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-modulus-monotone-normalization, real: none is bijective with $\mathsf{BishopModulatedCauchySequence}^{\uparrow}$ because this packet alone binds the sequence row, explicit modulus row, dyadic-radius ledger, finite-window route, regular rational handoff, and terminal real seal into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopModulatedCauchySequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopModulatedCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8631.1, Definition 8631.2, and Definition 8631.3 seed the finite explicit-modulus route from sequence windows to regular rational readback and terminal real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopModulatedCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, dyadic-radius visibility, finite-window coverage, regular readback stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopModulatedCauchySequenceUp.

Constructive story: A $\mathsf{BishopModulatedCauchySequence}^{\uparrow}$ packet is generated from a sequence row, explicit Cauchy modulus row, DyadicRatCore radius ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8632

A Concrete Naming Certificate for $\mathsf{MetricCategory}^{\uparrow}$

Following Lawvere and enriched-category treatments of metric spaces, the $\mathsf{MetricCategory}^{\uparrow}$ packet records the category whose objects are displayed metric-space rows and whose morphisms are uniformly continuous map rows. It sits over the metric carrier of chapter 57, the modulus-bearing map surface of chapter 130, the finite category interface of chapter 61, and the finite observation schedules of chapter 126. The packet names only the finite BEDC rows needed for identity, composition, classifier comparison, and modulus transport; it is not a host function category, a quotient of maps, an extensional equality principle, or an ambient enrichment theorem.

Definition 8632.1 (Metric category carrier)**.** *A* $\mathsf{MetricCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (O, F, I, K, M, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite object ledger, listing the* $\mathsf{Met}^{\uparrow}$ *rows whose point classifiers, distance rows, and metric-law ledgers may be read by the category packet.*
2. *F is the morphism ledger, listing* $\mathsf{ContinuousMap}^{\uparrow}$ *rows between the displayed metric objects together with their source, target, graph, and modulus data.*
3. *I is the identity-map ledger, assigning each object row in O a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *identity row over the same metric classifier.*
4. *K is the composition ledger, assigning each composable pair of morphism rows in F a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *composite row with the named source and target objects.*
5. *M is the metric-classifier ledger, recording that object comparison and morphism typing use only the* $\mathsf{Met}^{\uparrow}$ *classifiers already displayed in O.*
6. *L is the modulus ledger, recording the identity modulus rows and the composite modulus rows consumed from the* $\mathsf{ContinuousMap}^{\uparrow}$ *data in F, I, K.*
7. *H records componentwise* hsame *transport for object rows, morphism rows, identity rows, composition rows, classifier rows, modulus rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an object or morphism request through the finite object ledger, morphism ledger, identity or composition ledger, metric classifier ledger, and modulus ledger.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricCategory}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, F, I, K, M, L, H, C, P, N$. It has no coordinate for host functions, quotient equality of maps, selected representatives, global extensionality, countable choice, or an ambient category of all metric spaces.

Theorem 8632.2 (MetricCategory NameCert obligations)**.** *For every accepted* $\mathsf{MetricCategory}^{\uparrow}$ *carrier*

$$G = (O, F, I, K, M, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCategory}^{\uparrow}}$ *surface consists exactly of object admission through O, morphism admission through F, identity admission through I, composition admission through K, metric-classifier exposure through M, modulus exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every identity or composition read therefore remains inside the displayed $\mathsf{ContinuousMap}^{\uparrow}$ *modulus data over the displayed* $\mathsf{Met}^{\uparrow}$ *objects before the packet may be consumed as category data.*

Proof. Project the accepted carrier. The object row O is the only source of metric objects, and the morphism row F is the only source of displayed uniformly continuous maps between those objects.

The categorical rows are then forced to read the same data. The identity ledger I assigns identity maps over the metric classifiers already exposed by O, while the composition ledger K assigns composite maps only to pairs whose source and target rows match through the displayed typing information in F.

The classifier and modulus ledgers close the analytic part of the certificate. The row M prevents object or morphism comparison from leaving the displayed metric classifiers, and L records that identity and composition use the visible $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows rather than host-level functions.

Finally, H, C, P, N transport, replay, source, and name that same finite packet. Since no carrier coordinate stores quotient equality of maps, selected representatives, global extensionality, countable choice, or an ambient all-metric-spaces category, the listed rows exhaust the local NameCert obligations. □

Theorem 8632.3 (MetricCategory composition ledger)**.** *Let $G = (O, F, I, K, M, L, H, C, P, N)$ be an accepted* $\mathsf{MetricCategory}^{\uparrow}$ *carrier. Every composable pair read from F is admitted to K only after the source and target object rows are exposed through O and the two displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows are exposed through L. The resulting composite is again a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *row over the same metric-classifier ledger M.*

Consequently, associativity and identity-consumption checks for the packet are finite ledger checks over I, K, M, L, H, C, P, N, not host-function equality checks or quotient-map comparisons.

Proof. Start with a composable pair of morphism rows in F. Their source, middle, and target objects are read from O, so the composition request is typed before any composite row is exposed.

Next read the modulus ledger L. The two input morphisms carry displayed $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows, and L records the finite composite-modulus route used by K. Thus the composite map is admitted as another displayed $\mathsf{ContinuousMap}^{\uparrow}$ row rather than as an external function.

Identity checks use the same route. The row I supplies the identity map rows and their identity moduli over the object classifiers in O, while K records the left and right identity composites as finite ledger reads.

The structural rows then preserve the ledger. Transport and replay through H and C keep the same object, morphism, identity, composition, classifier, and modulus rows, and P, N source and name the packet. Because the carrier contains no host-function equality or quotient-map coordinate, the composition ledger supplies exactly the category-facing evidence available to consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCategory}^{\uparrow}$ packet is generated from MetricSpace object rows, ContinuousMap morphism rows, identity-map rows, composition rows, metric classifier rows, modulus ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8632.1, Definition 8632.2, and Definition 8632.3 seed the finite category packet of metric objects and uniformly continuous morphisms through displayed classifier and modulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function categories, quotient equality of maps, selected representatives, global extensionality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked object admission, morphism admission, identity admission, composition admission, classifier exposure, modulus conservation, transport, replay, provenance, and local naming for BEDC.Derived.MetricCategoryUp.

Chapter 8633

A Concrete Naming Certificate for CantorStieltjesMeasure$^{\uparrow}$

Following Lebesgue and Stieltjes, the CantorStieltjesMeasure$^{\uparrow}$ packet records the finite BEDC surface obtained when the Cantor distribution function supplies the displayed interval masses for a Stieltjes measure. It consumes the ternary-prefix to binary-prefix function rows of chapter 1895 and the finite half-open interval ledger of chapter 8592, with endpoint windows and terminal sealing supplied by the Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and StreamName$^{\uparrow}$ sibling chapters. The packet exposes only Cantor-function source rows, interval-window rows, finite Stieltjes ledger rows, seal rows, transport, replay, provenance, and local naming.

Definition 8633.1 (Cantor-Stieltjes measure carrier)**.** *A* CantorStieltjesMeasure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, I, L, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *F is the* CantorFunction$^{\uparrow}$ *source row, read through retained ternary prefixes, binary output prefixes, graph rows, and the real-facing seal of the Cantor-function packet.*
2. *I is the inspected half-open interval-window row, recording only the finite endpoint cells whose masses are requested.*
3. *L is the finite* StieltjesMeasure$^{\uparrow}$ *ledger row assigning each displayed interval in I its endpoint-difference mass through the source row F.*
4. *W is the* StreamName$^{\uparrow}$ *endpoint-window row used by the Cantor-function source reads and by the interval-mass ledger.*
5. *R is the* RegSeqRat$^{\uparrow}$ *regular readback row for the finite endpoint and mass windows.*
6. *S is the terminal* Real$^{\uparrow}$ *seal row for displayed interval masses and endpoint values.*
7. *H stores componentwise* hsame *transport for Cantor source rows, interval windows, Stieltjes ledger rows, stream windows, regular readbacks, real seals, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from Cantor-function prefix reads through interval windows, finite Stieltjes ledger rows, regular readback, and the terminal real seal.*
9. *P is the* Pkg *provenance over* CantorStieltjesMeasure$^{\uparrow}$, CantorFunction$^{\uparrow}$, StieltjesMeasure$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CantorStieltjesMeasure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, I, L, W, R, S, H, C, P, N$. It has no coordinate for general measure completion, countable additivity beyond the displayed finite ledger, host measure equality, quotient equality of distribution functions, selected extensions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorStieltjesMeasureUp.CantorStieltjesMeasureTasteGate single carrier alignment`

Theorem 8633.2 (Cantor-Stieltjes measure NameCert obligations)**.** *For every accepted* $\mathsf{CantorStieltjesMeasure}^{\uparrow}$ *carrier*

$$M = (F, I, L, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorStieltjesMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Cantor-function source row F, interval-window admission through I, finite Stieltjes ledger exposure through L, endpoint and mass readback through W and R, terminal real sealing through S, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8633.1. The source row F supplies the only distribution function data, and it is already routed through the finite Cantor-function prefix, graph, and real-seal rows of chapter 1895.

Read the interval and ledger rows next. The row I lists the inspected half-open endpoint windows, and L is the only finite Stieltjes ledger assigning masses to those displayed cells.

Finally route endpoint and mass reads through W, R, S and the structural rows H, C, P, N. This is the same finite Stieltjes surface exposed by chapter 8592, specialized to the Cantor-function source row, so no completed measure, countable-additivity theorem, host measure equality, selected extension, quotient distribution equality, or ambient completeness row remains. □

Theorem 8633.3 (Cantor-Stieltjes measure function boundary)**.** *Every measure-facing consumer of an accepted* $\mathsf{CantorStieltjesMeasure}^{\uparrow}$ *carrier must read the Cantor-function source row before any interval-mass seal is exposed. Concretely, each displayed interval mass factors through*

$$\mathsf{CantorFunction}^{\uparrow} \longrightarrow I \longrightarrow L \longrightarrow W, R \longrightarrow S, H, C, P, N.$$

The boundary exports finite Cantor-function prefix reads and finite Stieltjes ledger rows, but it does not export general measure completion, countable additivity beyond the displayed finite ledger, host measure equality, selected distribution extensions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a measure-facing read of an interval cell. The carrier has no primitive measure object, so the read first exposes the Cantor-function source row F and the inspected interval-window row I.

The finite ledger row L then assigns the displayed cell its endpoint-difference mass. The route uses the same half-open-cell and finite-additivity discipline as the $\mathsf{StieltjesMeasure}^{\uparrow}$ packet, while the endpoint values are supplied by the $\mathsf{CantorFunction}^{\uparrow}$ source surface.

The rows W and R provide the stream windows and regular rational readback before S can be read. Transport, replay, provenance, and local naming remain inside H, C, P, N, so a consumer receives only the displayed finite route and cannot escape to completed measure theory, host equality, selected extensions, countable choice, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorStieltjesMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorStieltjesMeasure}^{\uparrow}$ packet is generated from one CantorFunction source row, inspected half-open interval windows, one finite Stieltjes ledger, StreamName endpoint windows, RegSeqRat readbacks, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorStieltjesMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8633.1, Definition 8633.2, and Definition 8633.3 seed the finite Cantor-Stieltjes measure surface from Cantor-function prefix reads to displayed Stieltjes interval masses and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorStieltjesMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general measure completion, countable additivity beyond the displayed finite ledger, host measure equality, selected extension theorems, quotient distribution equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cantor-function source stability, interval-window exposure, finite Stieltjes ledger exactness, StreamName and RegSeqRat readback, Real-seal nonescape, transport and replay stability, provenance nonescape, and a Lean carrier formalization for BEDC.Derived.CantorStieltjesMeasureUp.

Chapter 8634

A Concrete Naming Certificate for $\mathsf{MooreMetrization}^{\uparrow}$

Following R. L. Moore and the Bing–Nagata metrization tradition, the $\mathsf{MooreMetrization}^{\uparrow}$ packet records the finite BEDC route from a developable topology with collectionwise normal separation data to a metric presentation. It sits after the Moore-space carrier of chapter 3592, the familywise separation surface of chapter 7988, and the Frink-style metric construction route of chapter 4226. The packet names only the displayed topology-to-metric dependency route; it does not import open-cover compactness, host choice of refinements, quotient topology equality, or an ambient metrization theorem.

Definition 8634.1 (Moore metrization carrier)**.** *A* $\mathsf{MooreMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, C, T, D, F, G, H, K, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{MooreSpace}^{\uparrow}$ *source row, carrying the displayed development windows and point-neighbourhood refinement reads.*
2. *C is the* $\mathsf{CollectionwiseNormal}^{\uparrow}$ *separation row, exposing the familywise closed-row separation ledger used by the development.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *base row shared by the development, closed-family, open-neighbourhood, and metric-open comparison reads.*
4. *D is the development compatibility row, recording that the inspected development windows refine the topology source without selecting a host sequence of covers.*
5. *F is the* $\mathsf{FrinkMetrization}^{\uparrow}$ *handoff row, translating finite star-refinement and separation reads into the metric-candidate ledger.*
6. *G is the* $\mathsf{Met}^{\uparrow}$ *endpoint row, exposing the resulting distance comparison and metric-open reads through finite BEDC data.*
7. *H records componentwise* hsame *transport for $S, C, T, D, F, G, K, P, N$ and for the displayed development-to-metric reads.*
8. *K records displayed* Cont *replay from Moore-space development through collectionwise-normal separation, topology comparison, Frink handoff, metric endpoint, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{MooreMetrization}^{\uparrow}$, $\mathsf{MooreSpace}^{\uparrow}$, $\mathsf{CollectionwiseNormal}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{FrinkMetrization}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreMetrization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, C, T, D, F, G, H, K, P, N$. It has no coordinate for arbitrary open covers, selected developments outside S, host powersets, quotient topology equality, open-cover compactness, or an untracked metrization theorem.

Theorem 8634.2 (Moore metrization NameCert obligations)**.** *For every accepted* $\mathsf{MooreMetrization}^{\uparrow}$ *carrier $M = (S, C, T, D, F, G, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreMetrization}^{\uparrow}}$ *surface consists exactly of Moore-space source admission through S, collectionwise-normal separation admission through C, topology-source alignment through T, development compatibility through D, Frink handoff through F, metric endpoint exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8634.1. The rows $S, C, T, D, F, G, H, K, P, N$ are the whole local surface.

The source rows are S, C, T, D. They bind the Moore development, the collectionwise-normal separation ledger, and the topology source before any metric endpoint can be read.

The metric construction rows are F and G. The Frink handoff reads only the displayed development and separation data, and the metric endpoint exposes only the resulting finite distance and metric-open comparisons.

Finally H, K, P, N transport, replay, source, and locally name the same finite route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8634.3 (Moore development metric route)**.** *Every metric read exported by an accepted* $\mathsf{MooreMetrization}^{\uparrow}$ *packet must pass through*

$$\mathsf{MooreSpace}^{\uparrow}, \quad \mathsf{CollectionwiseNormal}^{\uparrow}, \quad \mathsf{Topology}^{\uparrow}, \quad \mathsf{FrinkMetrization}^{\uparrow}$$

before it reaches the $\mathsf{Met}^{\uparrow}$ *endpoint row. The route preserves the development windows and familywise separation rows as finite BEDC data, so the metric endpoint is not obtained from open-cover compactness, host choice of refinements, quotient topology equality, or an ambient point-set metrization theorem.*

Proof. Read the source side first. The Moore-space row S supplies the development windows, the collectionwise-normal row C supplies the familywise separation ledger, and the topology row T keeps both surfaces tied to the same open and closed vocabulary.

The compatibility row D records that the displayed development is the one being used by the metric route. It does not choose a host sequence of covers or import open-cover compactness.

The Frink handoff F then converts the finite refinement and separation reads into the metric-candidate ledger, and the endpoint row G exposes the resulting $\mathsf{Met}^{\uparrow}$ comparisons. The structural rows H, K, P, N preserve that order under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{MooreMetrization}^{\uparrow}$ read exposes a MooreSpaceUp source row, CollectionwiseNormalUp separation row, TopologyUp source row, development compatibility row, FrinkMetrizationUp handoff, MetricSpaceUp endpoint, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any metric conclusion is public.

Independence witness: moore-space, collectionwise-normal, frink-metrization, topology, metric-space: $\mathsf{MooreMetrization}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds development windows, familywise separation, topology alignment, Frink handoff, metric endpoint, transport, replay, provenance, and local naming in one topology-to-metric route.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreMetrization}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreMetrization}^{\uparrow}$ packet is generated from MooreSpaceUp development windows, CollectionwiseNormalUp separation rows, a TopologyUp source, development compatibility, FrinkMetrizationUp handoff, MetricSpaceUp endpoint, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8634.1, Definition 8634.2, and Definition 8634.3 seed the finite topology-to-metric route from Moore development and collectionwise normal separation through Frink-style metric construction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreMetrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary metrization theorems, open-cover compactness, host-selected developments, quotient topology equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, development compatibility, collectionwise-normal separation visibility, Frink handoff exactness, metric endpoint exposure, transport, replay, provenance, and local naming for BEDC.Derived.MooreMetrizationUp.

Chapter 8635

A Concrete Naming Certificate for MetaCICResidualNormalizationObstruction$^{\uparrow}$

The MetaCICResidualNormalizationObstruction$^{\uparrow}$ packet realizes the normalization boundary of section 9386.4 as finite BEDC obstruction data. It sits beside the candidate closed-projection boundary of chapter 4366, the normal-endpoint cut of chapter 5143, the candidate-normalization discharge route of chapter 7180, and the typed normal-form scheduler of chapter 6617. The packet names the residual gap between arbitrary closed typed MetaCIC terms and the candidate-mediated normalization interface; it does not assert blanket strong normalization, closed consistency, or a subject-reduction theorem.

Definition 8635.1 (MetaCIC residual normalization obstruction carrier)**.** *A* MetaCICResidualNormalizationObstruc *carrier is a finite* BHist *packet*

$$R = (T, C, E, B, O, L, A, H, K, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row, recording the empty-context term, its displayed type, and the finite typing window under inspection.*
2. *C is the candidate-interface request row, asking whether the term can be placed into the candidate evidence surface already used by the normalization map.*
3. *E is the endpoint row, recording the bounded normal endpoint or the absence of such an endpoint inside the displayed candidate interface.*
4. *B is the residual beta-frontier row, exposing unreduced or unchecked residual fragments that remain after the bounded scheduler window.*
5. *O is the obstruction row, recording that candidate admission, endpoint preservation, or false-type preservation is not yet available for the inspected closed term.*
6. *L is the Lean TasteGate obligation row, requiring future* Nontrivial MetaCICResidualNormalizationObstruction$^{\uparrow}$ *and* FieldFaithful MetaCICResidualNormalizationObstruction$^{\uparrow}$ *instances before the obstruction classifier can serve as checked evidence.*
7. *A is the audit handoff row to subject-reduction and closed-consistency consumers; it exports the obstruction without discharging those consumers.*
8. *H records componentwise* hsame *transport for $T, C, E, B, O, L, A, K, P, N$.*

9. *K records displayed* Cont *replay from the closed-term request through candidate admission, endpoint, residual frontier, obstruction, TasteGate obligation, audit handoff, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$, MetaCICCandidateClosedProjectionBou $\mathsf{MetaCICNormalEndpointCut}^{\uparrow}$, $\mathsf{CandidateNormalizationDischarge}^{\uparrow}$, $\mathsf{MetaCICTypedNormalFormScheduler}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, E, B, O, L, A, H, K, P, N$. It has no coordinate for blanket strong normalization of arbitrary closed terms, closed consistency, hidden semantic candidates, host proof search, selected normal forms, or subject reduction.

Theorem 8635.2 (MetaCIC residual normalization obstruction NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$ *carrier $R = (T, C, E, B, O, L, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}}$ *surface consists exactly of closed typed-term admission through T, candidate-interface request through C, bounded normal endpoint accounting through E, residual beta-frontier exposure through B, obstruction recording through O, Lean* Nontrivial *and* FieldFaithful *obligation exposure through L, audit handoff through A, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8635.1. The closed term, candidate request, endpoint, residual frontier, obstruction, TasteGate obligation, and audit handoff are displayed as T, C, E, B, O, L, A.

The obstruction row O is not a proof of non-normalization. It records that the inspected closed typed term has not been placed into the candidate-mediated normalization interface with endpoint and false-type preservation visible in the same finite packet.

The TasteGate row L keeps the Lean obligations explicit: a future checked surface must provide both a nontrivial accepted/refused boundary and field-faithful preservation of the displayed rows. Transport, replay, provenance, and naming are then exhausted by H, K, P, N, so the listed rows are exactly the local NameCert obligations. □

Theorem 8635.3 (MetaCIC residual normalization obstruction boundary)**.** *Every closed-consistency or normalization consumer that reads an accepted* $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$ *packet receives an obstruction ledger, not a discharge of the open normalization question. The consumer must still provide candidate admission, bounded normal endpoint preservation, residual-frontier elimination, subject-reduction compatibility, and the Lean* Nontrivial *and* FieldFaithful *instances before the inspected closed term can be treated as normalized inside the public interface.*

Proof. Begin with the closed typed-term row T and the candidate request row C. These rows expose the term and the intended candidate interface, but they do not themselves provide candidate evidence.

Read the endpoint and residual rows next. The row E records the bounded endpoint status visible to the packet, while B records the residual beta frontier that remains under the scheduler window.

The obstruction row O preserves the gap rather than erasing it. The audit row A may be consumed by subject-reduction or closed-consistency routes, but it exports the same obstruction ledger. The TasteGate row L records the remaining Lean nontriviality and field-faithfulness obligations, and the structural rows H, K, P, N only transport, replay, source, and name the same finite packet. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$ read exposes a closed typed-term row, candidate-interface request, bounded normal endpoint accounting, residual beta-frontier row, obstruction row, Lean TasteGate obligation row, audit handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row. A read that exports blanket strong normalization, closed consistency, selected normal forms, hidden semantic candidates, or subject reduction from this packet refutes the carrier boundary.

Independence witness: metacic-candidate-closed-projection-boundary, metacic-normal-endpoint-cut, candidate-normalization-discharge, metacic-typed-normal-form-scheduler, metacic-sn-candidate-interface: $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$ is not bijective to any listed sibling because it keeps the residual frontier, candidate-admission request, endpoint accounting, obstruction row, Lean TasteGate obligation row, and audit handoff in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}$ packet is generated from a closed typed MetaCIC term row, candidate-interface request, bounded normal endpoint accounting, residual beta-frontier row, obstruction row, Lean TasteGate obligation row, audit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8635.1, Definition 8635.2, and Definition 8635.3 seed the finite obstruction packet for arbitrary closed typed MetaCIC terms outside the currently displayed candidate normalization interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualNormalizationObstruction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close blanket strong normalization, closed consistency, subject reduction, selected normal forms, hidden semantic candidates, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-interface request exactness, endpoint accounting, residual-frontier exposure, obstruction preservation, Nontrivial and FieldFaithful instances, audit handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualNormalizationObstructionUp.

Chapter 8636

LefschetzHyperplane NameCert Construction

Following Lefschetz hyperplane theorem references in algebraic geometry and Hodge theory, the $\mathsf{LefschetzHyperplane}^{\uparrow}$ packet records only the finite BEDC route by which a smooth projective source, a hyperplane section, and a bounded cohomology-window comparison are exposed to downstream consumers. It sits after the projective-variety surface of chapter 155, the scheme cover ledger of chapter 153, the cohomology and homology carriers of chapter 99 and chapter 98, and the numeric coefficient surface of chapter 39. The packet names the comparison route; it does not import a host Lefschetz theorem, a standard cohomology isomorphism, a selected ample divisor, a quotient cohomology representative, or a completed Hodge-theoretic package.

Definition 8636.1 (Lefschetz hyperplane carrier)**.** *A* $\mathsf{LefschetzHyperplane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, R, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the smooth projective source row, read through* $\mathsf{ProjectiveVar}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *carriers rather than through a host variety.*
2. *Y is the hyperplane-section row, including incidence with the displayed projective source and the finite section ledger.*
3. *R is the restriction-map row from the source packet to the section packet.*
4. *W is the cohomology-window row naming the inspected degree range and coefficient surface.*
5. *Q is the finite comparison ledger relating the source and section rows inside the displayed window.*
6. *H records componentwise* hsame *transport for X, Y, R, W, Q and the structural rows.*
7. *C records displayed* Cont *replay for source admission, section admission, restriction, cohomology-window readback, comparison, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{ProjectiveVar}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LefschetzHyperplane}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, R, W, Q, H, C, P, N$.

Theorem 8636.2 (Lefschetz hyperplane NameCert obligations)**.** *For the carrier of Definition 8636.1, the local* $\mathsf{NameCert}_{\mathsf{LefschetzHyperplane}^{\uparrow}}$ *surface consists exactly of projective-source admission through* X*, hyperplane-section admission through* Y*, restriction-map exposure through* R*, cohomology-window readback through* W*, finite comparison-ledger exposure through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, local naming through* N*, and non-escape from host Lefschetz theorems, unrecorded ample-divisor choices, quotient cohomology representatives, completed Hodge packages, and ambient standard-model isomorphisms.*

Proof. Project the carrier. The packet exposes exactly the rows $X, Y, R, W, Q, H, C, P, N$, with the geometric source, section, restriction, and comparison data separated into visible coordinates.

Read the dependency rows next. The projective and scheme siblings supply only displayed projective-source and section-carrier data. The cohomology, homology, and complex-number siblings supply only the finite degree window and coefficient rows used by W and Q.

Finally inspect the packet boundary. No coordinate stores a host Lefschetz theorem, an ample-divisor selector, a quotient cohomology representative, a completed Hodge package, or a standard-model isomorphism. The local NameCert obligations are therefore exactly the displayed finite packet and its transport, replay, provenance, and naming rows. □

Theorem 8636.3 (Lefschetz hyperplane section route)**.** *Every accepted* $\mathsf{LefschetzHyperplane}^{\uparrow}$ *consumer that reads a hyperplane-section comparison must first expose the route*

$$\mathsf{ProjectiveVar}^{\uparrow},\ \mathsf{Scheme}^{\uparrow},\ Y,\ R,\ W,\ Q,\ \mathsf{Cohomology}^{\uparrow},\ \mathsf{Homology}^{\uparrow},\ \mathsf{Complex}^{\uparrow},\ H,\ C,\ P,\ N.$$

The consumer receives a displayed projective source, a displayed section row, a restriction map, an inspected cohomology window, and a finite comparison ledger. It receives no host Lefschetz theorem, no ambient isomorphism outside the ledger, no selected generic hyperplane, and no quotient-class representative.

Proof. Start with source and section admission. The rows X and Y force the projective and scheme carriers to be displayed before the section comparison can be read.

Next read the restriction and window rows. The restriction row R and cohomology window W determine where the comparison ledger Q is meaningful, and the cohomology, homology, and complex-number siblings supply the inspected coefficient and degree surfaces.

The structural rows H, C, P, N preserve the same finite route. Since the refused host theorem, ambient isomorphism, generic-hyperplane choice, and quotient representative are not coordinates of the carrier, the section route exports only the displayed BEDC packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LefschetzHyperplane}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LefschetzHyperplane}^{\uparrow}$ packet is generated from a smooth projective source row, a hyperplane-section row, a restriction-map row, a cohomology-window row, a finite comparison ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LefschetzHyperplane}^{\uparrow}}$, seed)

Scope closed: Definition 8636.1, Definition 8636.2, and Definition 8636.3 bind the finite projective-section comparison route before any host Lefschetz theorem or ambient isomorphism can be read.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LefschetzHyperplane}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, a standard bridge, host Lefschetz theorem import, generic-hyperplane selection, quotient cohomology equality, or Hodge-theoretic comparison.

Upgrade path: Obligation closure requires explicit carrier-field coverage for projective source admission, hyperplane-section admission, restriction-map exposure, cohomology-window readback, comparison-ledger exactness, transport, replay, provenance, and local naming.

Chapter 8637

A Concrete Naming Certificate for $\mathsf{ProductiveCauchyName}^{\uparrow}$

Following Bishop and Bridges on countable regular Cauchy data, the $\mathsf{ProductiveCauchyName}^{\uparrow}$ packet records a finite productive-name surface whose schedule rows generate inspected Cauchy windows before any value-facing seal is read. It sits over the stream schedules of chapter 126, the rational regular readbacks of chapter 301, the dyadic tolerance core of chapter 291, the regular sequence-space packet of chapter 5965, and the terminal real-name seal of chapter 38. The packet names productive finite readback, not quotient equality, countable choice, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8637.1 (ProductiveCauchyName carrier)**.** *A* $\mathsf{ProductiveCauchyName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, D, Q, R, E, H, C, K, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, exposing the finite requests and response windows used by the productive name.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the displayed dyadic radius and slack data requested by each schedule window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
4. *R is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row, naming the ambient regular sequence-space carrier only through its displayed schedule, readback, tolerance, and seal route.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, Q, R have remained visible.*
6. *H records componentwise* hsame *transport for the schedule, tolerance, readback, regular-space, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a productive-name request through S, D, Q, R, E, H, K, N.*
8. *K is the* Pkg *provenance over* $\mathsf{ProductiveCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProductiveCauchyName}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of names, countable choice, trichotomy, a selected representative, a host limit, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 8637.2 (ProductiveCauchyName NameCert obligations)**.** *For every accepted* $\mathsf{ProductiveCauchyName}^{\uparrow}$ *carrier* $P = (S, D, Q, R, E, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProductiveCauchyName}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source routing, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the real-facing boundary must read the finite route*

$$S, D, Q, R \longmapsto E, H, C, K, N$$

and no obligation is stored outside these rows.

Proof. Project the carrier of Definition 8637.1. The visible coordinates are exactly the stream schedule, dyadic tolerance row, regular rational readback, regular sequence-space source, terminal seal, transport, replay, provenance, and local naming rows.

The analytic rows are consumed in order. The $\mathsf{StreamName}^{\uparrow}$ schedule opens the inspected windows, $\mathsf{DyadicRatCore}^{\uparrow}$ records the displayed tolerance scale for those windows, $\mathsf{RegSeqRat}^{\uparrow}$ reads them as regular rational data, and $\mathsf{RegularSequenceSpace}^{\uparrow}$ keeps the same readback inside the displayed sequence-space carrier.

Only after that route is visible can the $\mathsf{Real}^{\uparrow}$ seal be read. The structural rows H, C, K, N transport, replay, source, and name the same packet, so no quotient equality, countable-choice selector, selected representative, trichotomy, host limit, or ambient completeness datum remains outside the listed obligations. □

Theorem 8637.3 (ProductiveCauchyName regular readback)**.** *Every accepted* $\mathsf{ProductiveCauchyName}^{\uparrow}$ *read exposes its* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row* R *are available. The terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *therefore reads only a displayed productive Cauchy route and exports no quotient real equality, countable-choice schedule, real trichotomy, selected representative, or* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use the NameCert obligations of Definition 8637.2. They force every accepted read to expose S and D before Q can be consumed.

The readback row Q is regular only through the displayed windows and dyadic tolerances. The source row R then places that readback inside the regular sequence-space route of chapter 5965, without adding a quotient or a representative selector.

Finally the seal row E is downstream of S, D, Q, R, while H, C, K, N preserve the same finite route. Thus a real-facing consumer receives productive regular readback data, not countable choice, trichotomy, selected representatives, quotient equality, or an ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{ProductiveCauchyName}^{\uparrow}$ read exposes a StreamName schedule row, DyadicRatCore tolerance row, RegSeqRat readback row, RegularSequenceSpace source row, RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any value-facing seal is consumed.

Independence witness: streamname, dyadicratcore, regseqrat, regular-sequence-space, real: $\mathsf{ProductiveCauchyName}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries schedule

productivity, dyadic tolerance exposure, regular rational readback, regular sequence-space routing, terminal real sealing, transport, replay, provenance, and local naming in one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ProductiveCauchyName}^{\uparrow}}$.

Constructive story: A $\mathsf{ProductiveCauchyName}^{\uparrow}$ packet is generated from StreamName schedules, DyadicRatCore tolerance rows, RegSeqRat readback rows, a RegularSequenceSpace source route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProductiveCauchyName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8637.1, Definition 8637.2, and Definition 8637.3 seed the finite productive Cauchy-name route from countable schedule windows to regular rational readback and terminal real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProductiveCauchyName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality, countable choice, real trichotomy, selected representatives, host limit selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule productivity, dyadic tolerance exposure, regular rational readback, RegularSequenceSpace routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ProductiveCauchyNameUp.

Chapter 8638

A Concrete Naming Certificate for $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ packet records the finite evaluation functional by which a regular Cauchy rational tail answers a bounded precision request without selecting a completed limit. It sits after the regular source row of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the window-extraction packet of chapter 7135, and the terminal real-name seal of chapter 38. The packet is consumed by Real and completion-facing reads as a finite evaluator for displayed tails; it is not quotient completion, representative selection, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8638.1 (Cauchy tail evaluation functional carrier)**.** *A* $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, Q, V, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular Cauchy rational name and its tail-readback surface.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row listing exactly the finite suffix entries inspected for the request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, comparison budget, and arithmetic checks for the displayed tail.*
4. *Q is the finite request row naming the precision, the tail index, and the window extraction route read from* $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$*.*
5. *V is the evaluation-value row, the displayed rational or dyadic answer returned by the finite tail inspection.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, W, D, Q, V have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, tolerance, request, value, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the request row through R, W, D, V, S, H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$, $\mathsf{RegularCauchyWindowExtraction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, D, Q, V, S, H, C, P, N$. It has no coordinate for quotient representatives, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyTailEvaluationFunctionalUp.CauchyTailEvaluationFunctionalTasteGate`
`single carrier alignment`

Theorem 8638.2 (Cauchy tail evaluation functional NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ *carrier*

$$E = (R, W, D, Q, V, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, regular-source admission through R, finite tail-window admission through W, dyadic tolerance exactness through D, finite request admission through Q, evaluation-value exposure through V, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through S, classifier transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8638.1. The rows $R, W, D, Q, V, S, H, C, P, N$ are the complete inventory of an accepted packet.

The analytic rows are ordered. The regular source R is inspected only through the tail windows W selected by the finite request Q, and the tolerance ledger D records the dyadic comparisons used by that inspection.

The evaluation row V is therefore a displayed finite answer, not a selected limit. The seal row S is downstream of the same source, window, tolerance, request, and value rows, while H, C, P, N transport, replay, source, and name only that route. □

Theorem 8638.3 (Cauchy tail evaluation functional tail stability)**.** *Let $E = (R, W, D, Q, V, S, H, C, P, N)$ and $E' = (R', W', D', Q', V', S', H', C', P', N')$ be accepted* $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ *packets. If the regular source, tail-window, dyadic tolerance, finite request, and evaluation-value rows are transported componentwise through the displayed* hsame *rows and replayed through the displayed* Cont *rows, then the transported packet exposes the same finite tail-evaluation route before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed. No quotient representative, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row can be introduced by the transport.*

Proof. Apply Definition 8638.2 to both accepted packets. The obligations identify the evaluation route with the displayed rows $R, W, D, Q, V, S, H, C, P, N$ and $R', W', D', Q', V', S', H', C', P', N'$.

Componentwise hsame transport preserves the source, window, tolerance, request, and value rows as the same classified data, while Cont replay exposes those transported rows in the displayed order. The terminal seal remains downstream of the finite route.

Since the carrier has no coordinate for quotient representatives, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, transport and replay cannot add those objects. The accepted tail evaluator is stable exactly as a finite regular-Cauchy tail route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ read exposes a RegSeqRat source row, StreamName tail-window row, DyadicRatCore tolerance row, finite request row,

evaluation-value row, RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: regular-cauchy-window-extraction, regseqrat, streamname, dyadicratcore, real: $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ is not bijective to any listed sibling because it binds the finite request and evaluation-value rows to the regular Cauchy tail before the RealUp seal, while refusing quotient representatives, selected limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8638.1, Definition 8638.2, and Definition 8638.3 seed the finite evaluation-functional route over RegularCauchyWindowExtractionUp, RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-source admission, tail-window admission, tolerance exactness, finite request admission, value exposure, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailEvaluationFunctionalUp.

Constructive story: A $\mathsf{CauchyTailEvaluationFunctional}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName tail windows, a DyadicRatCore tolerance ledger, a finite request row, an evaluation-value row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8639

A Concrete Naming Certificate for $\mathsf{BirkhoffKakutani}^{\uparrow}$

Following Birkhoff and Kakutani on first-countable topological groups, the $\mathsf{BirkhoffKakutani}^{\uparrow}$ packet records the finite BEDC surface by which a displayed first-countable $\mathsf{TopGroup}^{\uparrow}$ source is handed to a left-invariant $\mathsf{Met}^{\uparrow}$ witness. It sits after the topological-group operation and continuity rows of chapter 204, the entourage interface of chapter 339, the metric carrier of chapter 57, and the shrinkage-to-metric route of chapter 4226. The packet names the neighborhood-basis, symmetric-shrinkage, product, inverse, and metric handoff rows; it is not a quotient group, a choice-selected countable family, a completed uniform space, or a host metrization theorem.

Definition 8639.1 (Birkhoff-Kakutani carrier)**.** *A* $\mathsf{BirkhoffKakutani}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, B, S, P, I, U, M, H, C, Q, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{TopGroup}^{\uparrow}$ *source row, exposing multiplication, unit, inverse, translation, and neighborhood-continuity data.*
2. *B is the countable neighborhood-basis ledger, represented by finite prefix windows rather than a choice-selected infinite basis.*
3. *S is the symmetric shrinkage ledger, recording the displayed sequence of neighborhoods whose squares stay inside the previous neighborhood.*
4. *P is the product-compatibility row, showing that multiplication consumes the shrinkage rows through the* $\mathsf{TopGroup}^{\uparrow}$ *continuity surface.*
5. *I is the inverse-compatibility row, showing that inversion preserves the displayed symmetric neighborhood rows.*
6. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *handoff row generated from left translates of the displayed neighborhood data.*
7. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row, read through the* $\mathsf{FrinkMetrization}^{\uparrow}$ *finite shrinkage-to-distance route.*
8. *H records componentwise* hsame *transport for the topological-group source, basis, shrinkage, product, inverse, uniform, metric, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a metrization request through $G, B, S, P, I, U, M, H, Q, N$.*

10. *Q is the* Pkg *provenance over* BirkhoffKakutani$^{\uparrow}$, TopGroup$^{\uparrow}$, UniformSpace$^{\uparrow}$, Met$^{\uparrow}$, FrinkMetrization$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BirkhoffKakutani}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, B, S, P, I, U, M, H, C, Q, N$. It has no coordinate for a quotient group, selected infinite basis, host left-uniformity object, ambient completion, or unlogged metrization oracle.

Theorem 8639.2 (Birkhoff-Kakutani NameCert obligations)**.** *For every accepted* BirkhoffKakutani$^{\uparrow}$ *carrier $K = (G, B, S, P, I, U, M, H, C, Q, N)$, the local* NameCert$_{\text{BirkhoffKakutani}^{\uparrow}}$ *surface consists exactly of topological-group source exposure through G, countable neighborhood-basis exposure through B, symmetric shrinkage through S, product compatibility through P, inverse compatibility through I, left-uniform handoff through U, left-invariant metric handoff through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Every metric-facing consumer therefore reads the TopGroup$^{\uparrow}$ *source, the finite basis prefixes, the symmetric shrinkage rows, product and inverse compatibility, and the* UniformSpace$^{\uparrow}$ *handoff before the* Met$^{\uparrow}$ *row can be consumed.*

Proof. Project the accepted carrier. The source row G is the only topological-group data available, and B is the only basis data exposed to consumers.

The shrinkage and compatibility rows are downstream from those sources. The row S records the symmetric neighborhood chain, P records that multiplication respects the chain, and I records inverse stability on the same displayed rows.

The uniform and metric rows then read the same finite route. The row U packages the left-translate entourage handoff, while M records the Frink-style metric handoff from the displayed shrinkage ledger.

Finally, H, C, Q, N transport, replay, source, and name that route. Since the carrier contains no quotient group, selected infinite basis, host uniformity object, ambient completion, or metrization oracle, the listed rows exhaust the local certificate. □

Closure status for NameCert$_{\text{BirkhoffKakutani}^{\uparrow}}$**.**

Constructive story: A BirkhoffKakutani$^{\uparrow}$ packet is generated from a TopGroup source row, countable neighborhood-basis windows, symmetric shrinkage rows, product and inverse compatibility, a UniformSpace handoff, a Metric handoff through FrinkMetrization, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BirkhoffKakutani}^{\uparrow}}$, seed)

Scope closed: Definition 8639.1 and Definition 8639.2 seed the finite first-countable topological-group metrization surface over topological-group, uniform, metric, shrinkage, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BirkhoffKakutani}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient groups, selected infinite bases, host uniformity objects, ambient completions, Lean verification, bridge checking, or an unrestricted metrization theorem outside the displayed rows.

Upgrade path: Obligation closure requires checked source exposure, basis-window admission, symmetric shrinkage, product compatibility, inverse compatibility, uniform handoff, metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.BirkhoffKakutaniUp.

Chapter 8640

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ packet realizes the boundary phase-shadow route of chapter 9463 as finite Poincare-disk, hyperbolic-metric, boundary-shadow, observer-phase, and refusal-ledger NameCert data. It sits after the metric source of chapter 3474, the boundary-shadow route of chapter 3813, the phase-transport packet of chapter 3455, the geodesic-projection packet of chapter 5917, and the boundary-transport packet of chapter 6267. The packet names only finite disk, phase, shadow, transport, refusal, replay, provenance, and local naming rows; it is not a completed ideal-boundary quotient, a host geodesic-flow object, a physical observer path, or a selected boundary phase.

Definition 8640.1 (Hyperbolic boundary phase-shadow carrier)**.** *A* $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, M, B, T, G, E, O, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare-disk source row, read through displayed disk coordinates and boundary transport.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing the metric comparison used by shadow windows.*
3. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, recording the boundary-shadow window visible from the disk source.*
4. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *row, recording phase transport along the displayed disk route.*
5. *G is the* $\mathsf{PoincareDiskGeodesicProjection}^{\uparrow}$ *row, exposing the geodesic projection used before a boundary phase read.*
6. *E is the* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *row, preserving the boundary endpoint data along the displayed transport.*
7. *O is the observer-phase ledger, recording how the boundary shadow and phase transport are read by observer-expansion consumers.*
8. *R is the refusal ledger, excluding completed ideal-boundary quotients, host geodesic-flow objects, physical observer paths, and selected boundary phases.*

9. *H records componentwise* hsame *transport for disk, metric, shadow, phase, geodesic, boundary, observer, refusal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a phase-shadow request through $D, M, B, T, G, E, O, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{PoincareDiskGeodesicProjection}^{\uparrow}$, $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B, T, G, E, O, R, H, C, P, N$. It has no coordinate for a completed ideal boundary, quotient phase class, selected observer path, host Mobius action, or global geodesic-flow theorem.

Theorem 8640.2 (Hyperbolic boundary phase-shadow NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ *carrier $S = (D, M, B, T, G, E, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}}$ *surface consists exactly of Poincare-disk source exposure through D, hyperbolic-metric comparison through M, boundary-shadow exposure through B, phase transport through T, geodesic projection through G, boundary transport through E, observer-phase ledger exactness through O, refusal exactness through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Thus every observer-expansion read reaches phase-shadow data only after the metric, boundary-shadow, phase-transport, geodesic-projection, and boundary-transport rows have been displayed.

Proof. Project the accepted carrier to D, M, B, T, G, E. These rows are the complete geometric surface available to the packet, and each is a displayed BEDC row rather than a host hyperbolic object.

The observer row O is downstream from those geometric rows. It records how the boundary-shadow read and phase-transport read are jointly consumed by an observer-facing request.

The refusal row R stays attached to that handoff. It blocks rereading the packet as a completed ideal-boundary quotient, quotient phase class, physical observer path, host Mobius action, or global geodesic-flow theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no excluded object is a coordinate of S, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ read is refuted if an observer-expansion consumer obtains boundary phase-shadow data without displaying Poincare disk source rows, HyperbolicMetric comparison, HyperbolicBoundaryShadow windows, PoincareDiskPhase-Transport, PoincareDiskGeodesicProjection, PoincareDiskBoundaryTransport, observer-phase ledger, refusal ledger, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: hyperbolic-metric, hyperbolic-boundary-shadow, poincare-disk-phase-transport, poincare-disk-geodesic-projection, poincare-disk-boundary-transport: no listed sibling is bijective to $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ because this packet jointly carries disk source, metric comparison, boundary-shadow window, phase transport, geodesic projection, boundary transport, observer-phase ledger, refusal, transport, replay, provenance, and local naming as one finite row surface.

The Lean TasteGate route must provide explicit Nontrivial $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ instances in `lean4/BEDC/Derived/HyperbolicBoundaryPhase`

The field-faithfulness witness must preserve $D, M, B, T, G, E, O, R, H, C, P, N$ as distinct rows before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryPhaseShadow}\uparrow}$.

Constructive story: A $\mathsf{HyperbolicBoundaryPhaseShadow}^{\uparrow}$ packet is generated from Poincare disk source rows, HyperbolicMetric comparison, HyperbolicBoundaryShadow windows, PoincareDiskPhaseTransport, PoincareDiskGeodesicProjection, PoincareDiskBoundaryTransport, observer-phase ledger exactness, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryPhaseShadow}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8640.1 and Definition 8640.2 seed the finite boundary phase-shadow route over disk, metric, shadow, phase, geodesic, boundary, observer, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryPhaseShadow}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed ideal-boundary quotients, quotient phase classes, physical observer paths, host Mobius actions, global geodesic-flow theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-source admission, metric comparison, boundary-shadow exposure, phase transport, geodesic projection, boundary transport, observer-phase ledger exactness, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBoundaryPhaseShadowUp.

Chapter 8641

A Concrete Naming Certificate for $\mathsf{UniformTotalBounded}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformTotalBounded}^{\uparrow}$ packet records total boundedness as a uniform finite-net certificate whose radii are read from displayed dyadic precision rows rather than from open-cover compactness or choice-selected finite subcovers. It sits after the total-bounded source of chapter 127, the metric carrier of chapter 57, the finite-net interface of chapter 1767, and the compact-metric consumer of chapter 129. The packet names the finite uniform coverage route consumed by compact, completion, and uniform-continuity chapters while excluding host enumerations of centers, selected subcovers, quotient metric equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8641.1 (Uniform total-bounded carrier)**.** *A* $\mathsf{UniformTotalBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, D, F, W, T, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, exposing the point carrier, point classifier, distance witness, zero-distance, symmetry, triangle, and distance-transport ledgers read by the packet.*
2. *D is the dyadic precision-radius row. It records the positive requested tolerance, its dyadic refinement, and the radius ledger used to ask for a finite net at that scale.*
3. *F is the finite net family row. For each precision request supplied by D, it displays a closed* $\mathsf{ProbeBundle}$ *of centers, center-carrier rows over M, and membership rows for those centers.*
4. *W is the coverage witness row. It records, for every inspected source point, a bundled center from F and a displayed metric distance comparison whose bound is the dyadic radius carried by D.*
5. *T is the uniform transport row, recording componentwise* hsame *transport for the metric source, dyadic radius ledger, finite net family, coverage witnesses, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a precision request through D, then through M, F, W, T, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{UniformTotalBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{FiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformTotalBounded}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, D, F, W, T, C, P, N. It has no coordinate for open-cover compactness, selected finite subcovers, host lists of centers, quotient equality of metric points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8641.2 (Uniform total-bounded finite-net obligation)**.** *For every accepted* $\mathsf{UniformTotalBounded}^{\uparrow}$ *carrier*

$$U = (M, D, F, W, T, C, P, N),$$

the finite-net obligation surface consists exactly of metric source exposure through M, dyadic precision-radius exposure through D, finite net family exposure through F, coverage witness exposure through W, uniform hsame *transport through T, displayed* Cont *replay through C, package provenance through P, and local naming through N. Any compact-metric, completion-facing, or uniform-continuity consumer that reads total boundedness from the packet must expose these rows before it may use the finite-net claim.*

Proof. Project the carrier of Definition 8641.1. The only analytic rows are the metric source M, dyadic radius ledger D, finite net family F, and coverage witness row W; the remaining rows transport, replay, source, and name that same finite packet.

Read a tolerance request through the dyadic row D. The finite net family F is then indexed by the displayed radius rather than by an external cover. Its centers are closed ProbeBundle entries with center-carrier rows over M.

Coverage is the row W. For each inspected source point it supplies a bundled center and a metric comparison bounded by the dyadic radius already displayed by D. The transport row T and replay row C preserve this route uniformly across hsame-equivalent presentations.

Finally P and N record provenance and local naming for the same rows. Since the carrier contains no open-cover, selected-subcover, host-enumeration, quotient-equality, countable-choice, or ambient-completeness coordinate, every exported total-boundedness claim is exactly the listed finite-net obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformTotalBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformTotalBounded}^{\uparrow}$ packet is generated from a metric source row, dyadic precision-radius row, finite net family row, coverage witness row, uniform hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformTotalBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8641.1 and Definition 8641.2 seed the finite-net uniformity certificate over displayed metric sources, dyadic precision radii, center bundles, coverage witnesses, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformTotalBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, host center enumeration, quotient metric equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic radius exactness, finite net family exposure, coverage witness stability, uniform transport, replay, provenance, and local naming for BEDC.Derived.UniformTotalBoundedUp.

Chapter 8642

A Concrete Naming Certificate for $\mathsf{HeineBorelCoverSelector}^{\uparrow}$

Following Heine, Borel, Lebesgue, and Bishop and Bridges, the $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ packet records the finite selector surface by which a displayed compact interval cover is consumed by Real and Completion routes. It sits after the finite-net interval packet of chapter 1379, the compact-metric source of chapter 129, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It reads finite subcover budgets through chapter 8630 and interval net evidence through chapter 1808. The packet names a finite cover-selector carrier only; it does not assert arbitrary open-cover compactness, a choice function on covers, a quotient real equality principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8642.1 (Heine–Borel cover-selector carrier)**.** *A* $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, K, W, R, D, E, F, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{HeineBorelInterval}^{\uparrow}$ *finite interval-net source.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coverage row over the same interval carrier.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite cover-window entries inspected by the selector.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint and radius observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing cover radii, mesh widths, and overlap slack.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, K, W, R, D are visible.*
7. *F is the finite selected-subcover ledger, listing exactly the displayed cover entries retained by the selector.*
8. *L is the local lower-bound and overlap ledger proving that the retained entries cover the displayed interval windows.*
9. *H records componentwise* hsame *transport for all selector rows.*
10. *C records displayed* Cont *replay from interval source to finite selector ledger and Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{HeineBorelCoverSelector}^{\uparrow}$, $\mathsf{HeineBorelInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. N is the local $\mathsf{NameCert}_{\mathsf{HeineBorelCoverSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite cover family, a host-side choice function, selected representatives of real names, open-cover compactness beyond the displayed interval net, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8642.2 (Heine–Borel cover-selector finite-net subcover)**.** *Let $S = (I, K, W, R, D, E, F, L, H, C, P, N)$ be an accepted* $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ *carrier. Every selected subcover exported by the packet must first expose the* $\mathsf{HeineBorelInterval}^{\uparrow}$ *row I, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row K, the finite* $\mathsf{StreamName}^{\uparrow}$ *cover windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D. Only then may the finite selector ledger F, coverage ledger L, Real seal E, transport, replay, provenance, and local naming rows be consumed.*

The route exports no arbitrary open-cover compactness theorem, countable choice schedule, host cover selector, quotient real equality, selected completed real, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 8642.1. The accepted packet displays the interval source, compact-metric finite-net source, finite cover windows, rational readback, dyadic tolerance ledger, selector ledger, coverage ledger, Real seal, and structural rows.

Read the finite-net side first. The $\mathsf{HeineBorelInterval}^{\uparrow}$ and $\mathsf{CompactMetric}^{\uparrow}$ rows identify the same displayed interval-net surface, while W, R, D restrict inspection to finitely many named cover entries and tolerances.

The selector ledger F and coverage ledger L are downstream of those rows. Transport, replay, provenance, and local naming preserve the same finite packet, so the excluded choice, quotient, open-cover compactness, selected-real, and ambient-completeness principles cannot be read from the carrier. □

Theorem 8642.3 (Heine–Borel cover-selector NameCert obligations)**.** *For an accepted* $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ *carrier $S = (I, K, W, R, D, E, F, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelCoverSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, interval finite-net exposure through I, compact-metric coverage exposure through K, finite cover-window exposure through W, regular rational readback through R, dyadic tolerance accounting through D, Real sealing through E, selected-subcover ledger exposure through F, coverage ledger exposure through L, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 8642.1. The rows I, K, W, R, D expose the displayed interval-net and finite cover-window source before any selected subcover can be inspected.

Apply Definition 8642.2. It forces the selector ledger F and coverage ledger L to remain downstream of the displayed finite-net, stream-window, rational, and dyadic rows.

Finally read E, H, C, P, N. These rows seal, transport, replay, package, and name the same finite selector carrier. Since no extra coordinate is present, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Any accepted $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ consumer that reads a selected subcover without displaying the $\mathsf{HeineBorelInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, selector, coverage, transport, replay, provenance, and local $\mathsf{NameCert}$ rows refutes the carrier.

Independence witness: heineborelinterval, compactmetric, real, streamname, regseqrat, dyadicratcore: no listed sibling is bijective with $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ because this packet jointly

carries an interval finite-net source, compact-metric coverage, finite cover windows, rational readback, dyadic tolerance accounting, selected-subcover ledger, coverage ledger, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelCoverSelector}\uparrow}$.

Constructive story: A $\mathsf{HeineBorelCoverSelector}^{\uparrow}$ packet is generated from a HeineBorelIntervalUp finite-net source, CompactMetricUp coverage rows, StreamNameUp finite cover windows, RegSeqRatUp readback, DyadicRatCoreUp tolerance accounting, a finite selected-subcover ledger, a coverage ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelCoverSelector}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8642.1, Definition 8642.2, and Definition 8642.3 seed the finite cover-selector route over HeineBorelInterval, CompactMetric, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelCoverSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, host cover choice, quotient real equality, selected completed reals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net subcover routing, selected-subcover ledger exactness, coverage ledger exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelCoverSelectorUp.

Chapter 8643

A Concrete Naming Certificate for $\mathsf{AntiSpecker}^{\uparrow}$

Following Specker and Bishop–Bridges constructive analysis, the $\mathsf{AntiSpecker}^{\uparrow}$ packet records the compactness boundary where a displayed stream of located real probes is refuted by finite-net data rather than by a selected convergent subsequence or a host compactness theorem. It sits over the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the compact-metric source of chapter 129, and the finite-net coverage row named by $\mathsf{TotallyBounded}^{\uparrow}$.

Definition 8643.1 (Anti-Specker carrier)**.** *An* $\mathsf{AntiSpecker}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, S, R, E, F, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compact information is exposed only as finite metric and cover data;*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, carrying the finite centers, radii, and membership alternatives used by the boundary argument;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for the inspected probe stream;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the requested rational windows of the stream;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed probe window after the regular-rational readback has been exposed;*
6. *F is the finite escape ledger, recording the inspected net cell, the displayed probe window, and the positive separation row for that cell;*
7. *G is the compact finite-net refusal row, stating that the escape ledger is read as a finite obstruction to a universal compact accumulation claim;*
8. *H records componentwise* hsame *transport for the compact source, net row, stream row, regular readback, real seal, escape ledger, refusal row, provenance, and local name;*
9. *C records displayed* Cont *replay from compact finite-net inspection through stream windows, regular readback, real sealing, finite escape, and refusal;*

10. *P records* Pkg *provenance over* $\mathsf{AntiSpecker}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AntiSpecker}^{\uparrow}}$ *registration row.*

The carrier has no coordinate for open-cover compactness, sequential compactness, selected subsequences, countable choice, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8643.2 (Anti-Specker NameCert obligations)**.** *For every accepted* $\mathsf{AntiSpecker}^{\uparrow}$ *carrier $A = (K, T, S, R, E, F, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AntiSpecker}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, finite-net admission through T, stream-window exposure through S, regular-rational readback through R, terminal real-name sealing through E, finite escape recording through F, compact finite-net refusal through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The compact source is read through K and the finite-net row T, while the probe stream is read through S, R, and E before any escape datum is available.

The finite escape ledger F is then a row over the already displayed net cell and probe window. It records a positive separation row for that finite inspection, not a global compactness theorem.

The refusal row G packages this finite escape as the local boundary consumed by compactness-facing readers. The structural rows H, C, P, N transport, replay, source, and name the same finite packet. No remaining coordinate can supply open-cover compactness, sequential compactness, selected subsequences, countable choice, quotient stream equality, or real completeness. □

Theorem 8643.3 (Anti-Specker finite escape route)**.** *Every compactness-facing consumer of an accepted* $\mathsf{AntiSpecker}^{\uparrow}$ *packet must read the route*

$$K, T, S, R, E, F, G, H, C, P, N$$

in that order: compact finite-net data first, then stream and regular-rational windows, then the terminal real-name seal, then the finite escape and refusal rows. The route provides a constructive compactness boundary over displayed finite-net rows and exports no selected subsequence, no countable-choice schedule, no open-cover compactness theorem, no quotient stream equality, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the local NameCert obligations of Definition 8643.2. They force any consumer to inspect the compact source and finite-net row before reading a stream window or regular-rational readback.

The real-facing seal E is downstream of S and R, so a consumer cannot treat the probe as a detached completed real. The escape row F is attached to that displayed window and to the finite-net cell named by T.

Finally the refusal row G is read only after the finite escape has been displayed, and H, C, P, N only transport, replay, source, and name this route. The packet therefore remains a finite constructive boundary rather than a sequential compactness, choice, open-cover, quotient, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{AntiSpecker}^{\uparrow}}$.

Constructive story: An $\mathsf{AntiSpecker}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a TotallyBoundedUp finite-net row, StreamName windows, RegSeqRat readback, RealUp seal rows, a finite escape ledger, a compact finite-net refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AntiSpecker}^{\uparrow}}$, seed)

Scope closed: Definition 8643.1, Definition 8643.2, and Definition 8643.3 seed the finite compactness-boundary route over displayed stream, regular-rational, real-seal, compact-metric, and finite-net rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{AntiSpecker}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, sequential compactness, selected subsequences, countable choice, quotient stream equality, or Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires carrier admission, finite-net exactness, stream and regular-rational read-back, real-seal nonescape, finite escape ledger coverage, transport stability, and provenance scope for BEDC.Derived.AntiSpeckerUp.

Chapter 8644

A Concrete Naming Certificate for $\mathsf{RealCompletionReflector}^{\uparrow}$

Following Bishop and Bridges and Kelley on metric completion, the $\mathsf{RealCompletionReflector}^{\uparrow}$ packet names the finite boundary by which a displayed metric source reaches a Real-sealed completion reflector without quotienting streams, selecting limits, invoking trichotomy, or using countable choice. It sits after the minimality route of chapter 1606, the reflector-unit packet of chapter 8626, the metric-completion service of chapter 1506, the regular rational name surface of chapter 301, and the terminal real-name seal of chapter 38. The packet is a reflector boundary for Real and Completion consumers; it is not Real completeness, a selected limit theorem, quotient stream equality, trichotomy, or a countable-choice principle.

Definition 8644.1 (Real completion reflector carrier)**.** *A* $\mathsf{RealCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, Q, E, U, F, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row, exposing the source carrier, source classifier, distance row, zero-distance, symmetry, triangle, and distance-transport ledgers read by the reflector.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row. It records the regular rational readback route, the finite dyadic tolerance requests inherited from* $\mathsf{DyadicRatCore}^{\uparrow}$*, the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows, and the terminal Real seal exposed to consumers.*
3. *E is the completion object row, displaying the* $\mathsf{MetricCompletion}^{\uparrow}$ *object together with the minimality and reflector-unit references supplied by the sibling completion packets.*
4. *U is the unit row from the source metric into the displayed completion object. It records only the finite graph of inspected source points and their completion names.*
5. *F is the universal arrow row. For each inspected uniformly continuous source-to-complete-target route, it records the finite factorization through U and the target's displayed Real-sealed completion readback.*
6. *T is the transport row, carrying componentwise* hsame *comparison for the source metric, regular rational and Real seal, completion object, unit, factorization route, replay, provenance, and local naming.*

7. *H records the reflector handoff to the existing* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *packets.*
8. *C records displayed* Cont *replay from source metric inspection through Q, E, U, F, T, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionReflector}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, E, U, F, T, H, C, P, N$. It has no coordinate for Real completeness, selected limits, quotient streams, trichotomy, countable choice, or host equality of completion representatives.

Theorem 8644.2 (Real completion reflector NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionReflector}^{\uparrow}$ *carrier*

$$R = (M, Q, E, U, F, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionReflector}^{\uparrow}}$ *obligation surface consists exactly of source metric admission through M, regular rational and Real-seal readback through Q, completion-object exposure through E, unit exposure through U, universal-arrow factorization through F, componentwise transport through T, completion-packet handoff through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8644.1. The source metric data is confined to M, while the Real-facing readback is confined to Q through displayed regular rational names, finite stream windows, dyadic tolerance requests, and the terminal seal.

The completion object and unit are the rows E and U. Their meaning is read through the sibling minimality, reflector-unit, and metric-completion packets recorded in H, not through an ambient completion theorem.

The universal-arrow surface is exactly F. It records a finite factorization route for each inspected source-to-complete-target map after M, Q, E, U have been displayed. The structural rows T, C, P, N transport, replay, source, and name this same route.

Since the carrier contains no coordinate for Real completeness, selected limits, quotient streams, trichotomy, countable choice, or host equality of representatives, those principles cannot be exported by the local NameCert surface. □

Theorem 8644.3 (Universal arrow factorization)**.** *Let $R = (M, Q, E, U, F, T, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionReflector}^{\uparrow}$ *packet. Every completion-facing consumer that reads a uniformly continuous source route from M to a displayed complete target must factor its read through the unit row U and the universal-arrow row F before it may inspect the Real-sealed completion row Q or the completion handoff row H.*

Proof. The NameCert obligations of Definition 8644.2 force any consumer to expose the source metric row M before a completion-facing route is available. The unit row U is the only row that connects the source metric to the displayed completion object.

The factorization row F is downstream of U. It stores the inspected arrow through the displayed completion object and no other coordinate stores such a route. The replay row C and transport row T preserve this same factorization under displayed continuity and componentwise hsame comparison.

The handoff row H then records the relationship with the existing minimality, reflector-unit, and metric-completion packets. Thus every completion-facing read passes through U and F before reaching Q or H. □

Theorem 8644.4 (Minimality handoff)**.** *Let $R = (M, Q, E, U, F, T, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionReflector}^{\uparrow}$ *packet. Every completion-facing read that reaches the completion object row E and the handoff row H must pass through the displayed* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, $\mathsf{CauchyCompletionReflectorUnit}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *rows before exporting the Real-sealed reflector interface through Q, F, T, C, P, N.*

The handoff exports no Real completeness theorem, selected limit operation, quotient stream equality, trichotomy principle, countable-choice schedule, or host equality of completion representatives.

Proof. Start with the local obligations of Definition 8644.2. They make E the only completion-object row and H the only row that records the sibling completion packets, so a completion-facing read cannot obtain a completion service by bypassing those two rows.

Apply the universal-arrow factorization of Definition 8644.3. Before the read may inspect the Real-sealed row Q or the handoff row H, it has already passed through the source metric row M, the unit row U, and the factorization row F.

The handoff row is then interpreted through the Cauchy-completion boundary. The NameCert obligations of Definition 1606.2 and the universal factorization route of Definition 1606.4 force the minimality-facing part of the read through the displayed source, dense embedding, universal-property, uniformly continuous extension, separatedness, transport, replay, provenance, and local naming rows. The reflector-unit route of Definition 1606.5 supplies the unit-facing entrance to the same route, while the $\mathsf{MetricCompletion}^{\uparrow}$ row supplies the displayed completion object named by E.

Thus the reflector packet exports only the composed finite route through E, H and the named sibling completion packets before returning to Q, F, T, C, P, N. Since each cited route omits Real completeness, selected limits, quotient streams, trichotomy, countable choice, and host equality, the Real-sealed reflector interface cannot export any of those principles. □

Theorem 8644.5 (Real-seal nonescape)**.** *Every Real-facing consumer read exported by an accepted* $\mathsf{RealCompletionReflector}^{\uparrow}$ *packet factors through the displayed route*

$$M, Q, E, U, F, T, H, C, P, N.$$

The packet supplies a Real-sealed completion reflector boundary and exports no Real completeness theorem, no selected limit operation, no quotient stream equality, no trichotomy principle, and no countable-choice schedule.

Proof. The route begins at M and Q. These rows expose the source metric and the regular rational, stream-window, dyadic-tolerance, and terminal Real-seal data that the consumer may inspect.

The completion object, unit, and universal-arrow rows E, U, F then determine how the completion-facing read is routed. The handoff row H connects this route to the existing completion packets without adding any additional analytic coordinate.

Finally T, C, P, N transport, replay, source, and name the same finite packet. Because Real completeness, selected limits, quotient streams, trichotomy, and countable choice are absent from the carrier, the Real-seal read cannot escape the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionReflector}^{\uparrow}$ packet is generated from a source metric row, RegSeqRat and Real seal row, completion object row, unit row, universal-arrow factorization row, componentwise hsame transport, completion-packet handoff, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8644.1, Definition 8644.2, Definition 8644.3, Definition 8644.4, and Definition 8644.5 seed the finite reflector boundary over displayed metric, regular-rational, stream-window, dyadic-tolerance, Real-seal, completion-object, unit, universal-arrow, transport, handoff, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealCompletionReflectionUp.RealCompletionReflectionTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not claim Real completeness, selected limits, quotient streams, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Real-seal readback, completion object exposure, unit stability, universal-arrow factorization, transport, handoff, replay, provenance, and local naming for BEDC.Derived.RealCompletionReflecto

Chapter 8645

A Concrete Naming Certificate for $\mathsf{SequentialContinuityCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ packet records the constructive criterion that a displayed continuous real map carries convergent source sequences to convergent image sequences through finite observation and modulus data. It sits over the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the displayed continuity rows of $\mathsf{Continuous}^{\uparrow}$, and the real-completeness route of chapter 9238. The packet gives Real and Completion consumers a sequence-continuity criterion without quotient streams, host function equality, selected limits, countable choice, or ambient completeness.

Definition 8645.1 (Sequential continuity criterion carrier)**.** *A* $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, S_X, S_Y, W_X, W_Y, M, R_X, R_Y, E_X, E_Y, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the source and target* $\mathsf{Real}^{\uparrow}$ *rows whose sequence windows are inspected.*
2. *F is the* $\mathsf{Continuous}^{\uparrow}$ *continuity row for the displayed map, carrying only graph and tolerance data visible to the certificate.*
3. *S_X and S_Y are the source and image sequence rows, exposed as finite* $\mathsf{StreamName}^{\uparrow}$ *observations.*
4. *W_X and W_Y are the source sequence windows and image sequence windows read from S_X and S_Y.*
5. *M is the continuity-modulus row. It records the finite tolerance-to-precision requests used to turn a source-window convergence request into an image-window convergence request.*
6. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the source and image windows.*
7. *E_X and E_Y are the* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the corresponding regular-rational readbacks have been displayed.*
8. *H records componentwise* hsame *transport for the source, target, map, sequence windows, modulus, readbacks, seals, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from source sequence windows through the continuity modulus into image sequence windows, then through regular readback and Real seals.*

10. *P is the* Pkg *provenance over* $\mathsf{SequentialContinuityCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuityCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, host function equality, selected representatives, countable choice, an external limit selector, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 8645.2 (Sequential continuity criterion NameCert obligations)**.** *For every accepted* $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ *carrier* $K = (X, Y, F, S_X, S_Y, W_X, W_Y, M, R_X, R_Y, E_X, E_Y, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuityCriterion}^{\uparrow}}$ *surface consists exactly of source and target Real rows, a displayed* $\mathsf{Continuous}^{\uparrow}$ *row, source and image sequence windows, a finite continuity-modulus row,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, terminal* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The source sequence is visible only through S_X, W_X, R_X, E_X, and the image sequence is visible only through S_Y, W_Y, R_Y, E_Y.

The continuity information is the displayed row F together with the finite modulus row M. A convergence request for the image sequence must first read the requested target tolerance in M, then inspect the source windows W_X at the corresponding precision, and only then open the image windows W_Y.

The structural rows H, C, P, N preserve and name that finite route. Since quotient streams, host function equality, selected representatives, choice, external limit selection, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8645.3 (Sequential continuity criterion StreamName handoff)**.** *Every Real or Completion consumer that applies the* $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ *packet to a displayed source sequence must read the finite handoff*

$$S_X, W_X, M, F, W_Y, S_Y$$

before any regular-rational readback or Real seal is opened. Thus the criterion is consumed through $\mathsf{StreamName}^{\uparrow}$ *observations of source and image windows, not through quotient streams, host function equality, selected limits, or countable choice.*

Proof. Start from the carrier definition. The only sequence-facing rows are S_X, S_Y, W_X, W_Y, and the only bridge from source windows to image windows is the pair M, F.

Given a target tolerance, the consumer first reads the modulus row M. The requested source precision is then inspected in W_X as a finite StreamName window, and the displayed continuity row F routes that window to the image window W_Y.

Only after this finite handoff may the consumer continue to readback and seal rows. The handoff therefore remains a window route and cannot be replaced by an equality of streams, a host-level function equation, a selected limit, or a choice schedule. □

Theorem 8645.4 (Sequential continuity criterion RegSeqRat and Real seal)**.** *For every accepted* $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ *carrier K, any image-convergence read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must pass through the source* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R_X, the image* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R_Y, and the terminal seals E_X, E_Y in the displayed carrier. The route is compatible with the real-completeness vision of chapter 9238 because it exposes regular rational windows before the seal, but it does not itself assert ambient completeness or select a limit outside the packet.*

Proof. Use the StreamName handoff of Definition 8645.3. It places the source and image windows before any Real-facing conclusion.

The regular-rational rows then supply the only readback surface. The source readback R_X records the regular information consumed by the modulus, and the image readback R_Y records the regular information exported to a Real or Completion consumer.

Finally the seals E_X, E_Y are terminal rows downstream of R_X, R_Y. The structural rows H, C, P, N keep the same finite packet visible, so the route aligns with the regular-Cauchy discipline of the real-completeness vision while refusing ambient completeness, selected limits, quotient streams, host equality, and countable choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialContinuityCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialContinuityCriterion}^{\uparrow}$ packet is generated from source and target RealUp rows, a ContinuousUp row, finite source and image StreamName windows, a continuity-modulus row, RegSeqRat readbacks, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialContinuityCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8645.1, Definition 8645.2, Definition 8645.3, and Definition 8645.4 seed the finite criterion route from displayed source sequence windows through continuity-modulus data to image readback and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialContinuityCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host function equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{SequentialContinuityCriterion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, StreamName source and image window handoff, continuity-modulus exactness, RegSeqRat readback, RealUp seal ordering, transport stability, provenance, and local naming for BEDC.Derived.SequentialContinuityCriterionUp.

Chapter 8646

A Concrete Naming Certificate for HausdorffCompletionFunctor$^{\uparrow}$

Following Bourbaki and Bishop–Bridges on separated uniform completions, the HausdorffCompletionFunctor$^{\uparrow}$ packet records the finite map-level surface that turns separated completion into a displayed functorial route. It sits over the envelope packet of chapter 5789 and the pullback comparison of chapter 8585; it also reads the unit-facing separated-completion rows already exposed by the Hausdorff completion route. The packet is not an ambient categorical completion theorem: it is a finite NameCert carrier for source and target completion rows, uniformly continuous map data, unit compatibility, pullback compatibility, transport, replay, provenance, and local naming.

Definition 8646.1 (HausdorffCompletionFunctor carrier)**.** *A* HausdorffCompletionFunctor$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, M, C, U, P, H, R, Q, N)$$

with the following visible rows:

1. *X is the source separated-completion row, including the displayed Hausdorff separation, stream-window, regular-rational readback, dyadic tolerance, and terminal real-seal coordinates already named by the envelope route.*
2. *Y is the target separated-completion row, with the same visible row shape for the codomain.*
3. *M is the uniformly continuous map row from the source presentation to the target presentation, carried only as displayed finite map and modulus data.*
4. *C is the completion action row, recording the induced map on completion names as a finite readback route rather than as a host function on quotient completions.*
5. *U is the unit-compatibility row, requiring source values to enter the source completion and then map forward along the same route as direct application followed by the target unit row.*
6. *P is the pullback-compatibility row, forcing comparisons of two completed map reads to pass through the public pullback packet of chapter 8585.*
7. *H stores the componentwise* hsame *transports for source completion, target completion, map row, completion action, unit row, pullback row, replay, provenance, and local certificate.*
8. *R stores displayed* Cont *replay for source observation, target observation, uniform-continuity readback, completion action, unit compatibility, and pullback compatibility.*

9. *Q is the* Pkg *provenance over* HausdorffCompletionFunctor$^{\uparrow}$, HausdorffCompletionEnvelope$^{\uparrow}$, HausdorffCompletionI HausdorffCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{HausdorffCompletionFunctor}^{\uparrow}}$ *row.*

The carrier predicate says that these rows form one finite functorial separated-completion packet. It has no coordinate for quotient-completion equality, selected representatives, countable choice, a host function extensionality theorem, or ambient completeness.

Lean4 (statement) BEDC.Derived.HausdorffCompletionFunctorUp

Theorem 8646.2 (HausdorffCompletionFunctor obligation surface)**.** *For every accepted* HausdorffCompletionFunct *carrier $F = (X, Y, M, C, U, P, H, R, Q, N)$, the local* NameCert$_{\text{HausdorffCompletionFunctor}^{\uparrow}}$ *surface consists exactly of source completion admission through X, target completion admission through Y, uniformly continuous map admission through M, completion-action readback through C, unit compatibility through U, pullback compatibility through P, componentwise transport through H, replay through R, provenance through Q, and local naming through N. A consumer therefore reads functoriality only through the displayed finite route*

$$X, Y, M \longmapsto C, U, P, H, R, Q, N.$$

Proof. Project the carrier of Definition 8646.1. The source and target rows are the two separated-completion surfaces, and the map row is the displayed uniformly continuous input.

The completion action C is admitted only after those rows are visible. Unit compatibility is then checked by U, and pullback compatibility is checked by P through the sibling pullback packet rather than through host equality of completed maps.

Finally H, R, Q, and N replay, transport, package, and name the same finite rows. Since the carrier has no coordinate for quotient equality, selected representatives, countable choice, function extensionality, or ambient completeness, the listed rows exhaust the obligation surface. □

Theorem 8646.3 (HausdorffCompletionFunctor unit-pullback route)**.** *Every unit-facing or comparison-facing read from an accepted* HausdorffCompletionFunctor$^{\uparrow}$ *carrier factors through the source row X, target row Y, uniformly continuous map row M, completion action row C, unit row U, and pullback row P before any real-facing seal or completion comparison is public. The unit read is therefore tied to the displayed separated-completion envelope of chapter 5789, and the comparison read is tied to the public pullback packet of chapter 8585.*

Proof. Read the completion action coordinate C. It is downstream of the displayed source and target completion rows and the uniformly continuous map row, so no unit-facing consumer can inspect a completed value before X, Y, M are fixed.

The unit coordinate U compares direct source admission with mapped target admission only through those same rows. It does not introduce a separate representative selector or a host equality proof.

For comparison-facing reads, use the pullback coordinate P. It routes the comparison through the Hausdorff completion pullback packet, where the two separated-completion projections remain visible before the comparison becomes public. □

Theorem 8646.4 (HausdorffCompletionFunctor nonescape)**.** *No accepted* HausdorffCompletionFunctor$^{\uparrow}$ *carrier exports a map on completed Hausdorff objects except through the displayed rows*

$$X, Y, M, C, U, P, H, R, Q, N.$$

In particular, every real-facing or completion-facing consumer must keep the source and target completion rows, uniformly continuous map row, unit compatibility row, pullback compatibility row, transport, replay, provenance, and local naming visible; the route exports no quotient-completion equality, selected representative, countable-choice selector, host function extensionality, or ambient completeness witness.

Proof. Start from the obligation surface of Definition 8646.2. It lists the only rows through which functoriality is admitted.

Apply the unit-pullback route of Definition 8646.3. Unit reads are tied to X, Y, M, C, U, and comparison reads are tied to X, Y, M, C, P before downstream consumers can inspect a completed value or comparison.

The structural rows H, R, Q, N preserve the same finite route under transport, replay, provenance, and naming. Thus the carrier contains no escape coordinate for quotient-completion equality, representative choice, countable choice, host extensionality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFunctor}\uparrow}$.

Constructive story: A $\mathsf{HausdorffCompletionFunctor}^{\uparrow}$ packet is generated from source and target separated-completion rows, a uniformly continuous map row, completion-action readback, unit compatibility, pullback compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFunctor}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8646.1 seeds the finite functorial separated-completion carrier over envelope, pullback, unit, stream, regular-rational, dyadic, and Real-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFunctor}\uparrow}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.HausdorffCompletionFunctorUp`

Bridge status: none

Not claimed: This chapter does not claim quotient-completion equality, selected representatives, countable choice, host function extensionality, ambient completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked source and target completion admission, uniformly continuous map routing, unit compatibility, pullback compatibility, transport, replay, provenance, and nonescape for BEDC.Derived.HausdorffCompletionFuncto

Chapter 8647

A Concrete Naming Certificate for $\mathsf{FaaDiBruno}^{\uparrow}$

Following Faà di Bruno and the classical finite chain-rule formulae, the $\mathsf{FaaDiBruno}^{\uparrow}$ packet records the finite partition expansion for iterated derivatives of a composite as displayed NameCert data. It sits after the derivative observation surface of chapter 122, the finite Taylor theorem route of chapter 5898, the Taylor-polynomial truncation packet of chapter 6674, the polynomial carrier of chapter 50, the regular-rational readback rows of chapter 301, and the terminal real-name seals of chapter 38. The packet names only a finite partition-and-coefficient handoff; it does not import host equality of functions, analytic completeness, quotient real equality, selected infinite power series, or an ambient smooth calculus.

Definition 8647.1 (Faà di Bruno finite-partition carrier)**.** *A* $\mathsf{FaaDiBruno}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (n, D, T, U, P, A, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *n is the unary derivative order requested from the composite.*
2. *D is the finite* $\mathsf{Derivative}^{\uparrow}$ *observation row listing the derivative endpoints of the outer and inner maps that may be inspected up to order n.*
3. *T is the* $\mathsf{TaylorTheorem}^{\uparrow}$ *jet row that bounds the finite derivative reads used by the chain-rule expansion.*
4. *U is the* $\mathsf{TaylorPolynomial}^{\uparrow}$ *truncation row exposing exactly the polynomial degree needed for the requested order.*
5. *P is the* $\mathsf{Poly}^{\uparrow}$ *monomial and coefficient ledger for the Bell-polynomial partition monomials.*
6. *A is the finite partition ledger. Each row records one integer partition of n, its multiplicities, and the combinatorial coefficient consumed by the expansion.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed coefficients and derivative values.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the finite coefficient and derivative reads.*
9. *H stores componentwise* hsame *transport for the order, derivative observations, Taylor rows, polynomial rows, partition ledger, readback rows, real seals, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from the partition ledger through the derivative, Taylor, polynomial, readback, and real-seal rows.*
11. *Q is the* Pkg *provenance over* $\mathsf{FaaDiBruno}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorPolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FaaDiBruno}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host function equality, an infinite derivative tower, an analytic convergence theorem, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8647.2 (Faà di Bruno NameCert obligations)**.** *For every accepted* $\mathsf{FaaDiBruno}^{\uparrow}$ *carrier*

$$F = (n, D, T, U, P, A, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FaaDiBruno}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed order, derivative observations, Taylor-theorem jet, Taylor-polynomial truncation, polynomial coefficient ledger, finite partition ledger, regular-rational readbacks, real seals, componentwise transport, displayed replay, package provenance, and local naming rows.*

The partition exactness obligation requires every summand of the chain-rule expansion to cite one row of A, one coefficient row of P, the derivative observations in D, the Taylor support rows T, U, and the readback route R, E before a consumer may inspect the composite-derivative coefficient. No other analytic obligation is part of the packet.

Proof. Project the carrier of Definition 8647.1. The theorem-facing rows are exactly n, D, T, U, P, A, R, E, and the structural rows are exactly H, C, Q, N.

The expansion is finite because the request order n is unary and the partition ledger A lists only the partitions of that order. Each listed row names its multiplicities and coefficient, while P supplies the polynomial monomial and coefficient source used by that row.

The derivative and Taylor dependencies are also displayed. The row D supplies only the derivative observations used by the summand, and T, U restrict those observations to a finite Taylor jet and a finite Taylor polynomial truncation.

Finally route the displayed coefficients and derivative reads through R, E, H, C, Q, N. Since the carrier stores no host function equality, infinite derivative tower, convergence theorem, quotient real equality, countable choice, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 8647.3 (Faà di Bruno finite-partition handoff)**.** *Every accepted* $\mathsf{FaaDiBruno}^{\uparrow}$ *carrier exports a composite-derivative coefficient only through the finite route*

$$\mathsf{Derivative}^{\uparrow}, \mathsf{TaylorTheorem}^{\uparrow}, \mathsf{TaylorPolynomial}^{\uparrow}, \mathsf{Poly}^{\uparrow} \longmapsto A, P \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the order row n, derivative observation row D, Taylor-theorem row T, Taylor-polynomial truncation U, polynomial ledger P, finite partition ledger A, regular-rational readback R, real seal E, transport H, replay C, provenance Q, and local NameCert *row N before using the coefficient. The handoff exports only the finite partition expansion for the displayed order, not a host chain-rule theorem over arbitrary functions, an infinite Bell-polynomial family, quotient real equality, selected representatives, or analytic completeness.*

Proof. Start with an accepted carrier $F = (n, D, T, U, P, A, R, E, H, C, Q, N)$. The requested order n fixes a finite partition set, and A displays exactly the partition rows consumed by the packet.

Read the derivative and Taylor support next. The row D supplies the derivative observations, while T and U restrict those reads to the finite jet and finite polynomial truncation needed by the requested order.

The polynomial coefficient ledger P and the partition ledger A assemble the displayed summands. Each summand is therefore read from named finite rows before any coefficient reaches a regular-rational consumer.

Finally pass through R, E, H, C, Q, N. Regular rational readback precedes the terminal real seals, and the structural rows preserve the same finite partition route. Because the carrier has no coordinate for host function equality, infinite families, quotient real equality, selected representatives, or analytic completeness, every exported coefficient is exhausted by the finite handoff above. □

Falsifiable prediction: Every accepted $\mathsf{FaaDiBruno}^{\uparrow}$ composite-derivative coefficient read factors through displayed Derivative, TaylorTheorem, TaylorPolynomial, Polynomial, finite partition, RegSeqRat, Real, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: derivative, taylor-theorem, taylor-polynomial, polynomial, regseqrat, real: $\mathsf{FaaDiBruno}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds one derivative order, finite derivative observations, Taylor support, polynomial coefficient rows, partition multiplicities, coefficient readback, endpoint seals, transport, replay, provenance, and local naming in one finite chain-rule certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FaaDiBruno}^{\uparrow}}$.

Constructive story: A $\mathsf{FaaDiBruno}^{\uparrow}$ packet is generated from a unary derivative order, finite DerivativeUp observation rows, TaylorTheoremUp jet support, TaylorPolynomialUp truncation, PolynomialUp coefficient rows, a finite partition ledger, RegSeqRatUp readbacks, RealUp seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FaaDiBruno}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8647.1, Definition 8647.2, and Definition 8647.3 seed the finite partition chain-rule route over $\mathsf{Derivative}^{\uparrow}$, $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorPolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FaaDiBruno}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host equality of functions, an infinite derivative tower, analytic convergence, quotient real equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition exactness, derivative and Taylor support routing, polynomial coefficient handoff, readback sealing, transport and replay stability, and nonescape for BEDC.Derived.FaaDiBrunoUp.

Chapter 8648

A Concrete Naming Certificate for $\mathsf{KnoppTailCondensation}^{\uparrow}$

Following Knopp and Bishop–Bridges constructive analysis, the $\mathsf{KnoppTailCondensation}^{\uparrow}$ packet records the finite tail-condensation route needed by the Real and Completion series corridor. It starts from displayed $\mathsf{RealSeries}^{\uparrow}$ source rows, opens Cauchy-condensation block rows, reads dyadic tolerances through $\mathsf{DyadicRatCore}^{\uparrow}$, hands the finite regular sequence data through $\mathsf{RegSeqRat}^{\uparrow}$, and reaches a $\mathsf{Real}^{\uparrow}$ seal only after those rows have been exposed. The packet is a finite NameCert surface; it is not a host summability theorem, selected sum, quotient equality, countable-choice rule, or ambient completeness principle.

The source series surface is supplied by chapter 1581. The condensation block discipline is the one isolated in chapter 1478. Dyadic tolerances, regular-sequence readback, and Real sealing are inherited only as named rows from chapter 291, chapter 301, and chapter 38.

Definition 8648.1 (Knopp tail-condensation carrier)**.** *A* $\mathsf{KnoppTailCondensation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, B, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row whose nonnegative tail is being read;*
2. *T is the displayed tail-window row selected from the source series;*
3. *B is the finite Cauchy-condensation block row over dyadic tail intervals;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the block bounds;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that records the regular tail handoff;*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, B, D, R are displayed;*
7. *H records componentwise* hsame *transport for the source, tail window, block, tolerance, readback, seal, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes from S through T, B, D, R into A;*
9. *P records* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and certificate rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KnoppTailCondensation}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $S, T, B, D, R, A, H, C, P, N$.

Theorem 8648.2 (Knopp tail-condensation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{KnoppTailCondensation}^{\uparrow}}$ *surface for Definition 8648.1 consists of carrier admission for $K = (S, T, B, D, R, A, H, C, P, N)$, source-series exposure through S, tail-window exposure through T, condensation-block exposure through B, dyadic tolerance exposure through D, regular readback through R,* $\mathsf{Real}^{\uparrow}$ *seal delivery through A, componentwise* hsame *stability through H, displayed* Cont *route replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed tuple in Definition 8648.1. The source series, tail window, condensation block, dyadic tolerance, regular readback, seal, transport, route, provenance, and naming rows are all visible coordinates of one finite packet.

Read the obligations in order. The source row S opens only the displayed tail window T; the block row B records the finite condensation slices; the dyadic row D supplies the tolerance data consumed by the block; and the regular readback row R is downstream of those displayed finite rows.

The $\mathsf{Real}^{\uparrow}$ seal row A is available only after T, B, D, R have been exposed. Since the carrier has no coordinate for selected sums, quotient stream equality, countable choice, host summability, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8648.3 (Knopp tail-condensation L10 handoff)**.** *Every accepted* $\mathsf{KnoppTailCondensation}^{\uparrow}$ *carrier hands its Real-and-Completion consumer the finite route*

$$S, T, B, D, R, A, H, C, P, N$$

from $\mathsf{RealSeries}^{\uparrow}$ *source data through Cauchy-condensation blocks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. The handoff cannot bypass the displayed tail window by invoking host summability, selected sums, quotient equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier to its displayed rows. The route row C is the only continuation surface by which a consumer reaches A, and it passes through the same tail, block, tolerance, and regular readback rows.

Apply the NameCert obligations of Definition 8648.2. They force the source, tail window, block row, dyadic tolerance, regular readback, seal, transport, provenance, and naming rows to remain the same finite packet under componentwise hsame transport.

Finally inspect the carrier inventory. No coordinate stores a host-selected sum, quotient stream equality, countable-choice selector, hidden limit, or ambient completeness proof, so the L10 handoff is exactly the displayed finite route. □

Falsifiable prediction: A tail-condensation packet that reaches a $\mathsf{Real}^{\uparrow}$ seal without a displayed Cauchy-condensation block row is not an accepted $\mathsf{KnoppTailCondensation}^{\uparrow}$ carrier.

Independence witness: realseries, cauchycondensationtest, dyadicratcore, regseqrat, real: these siblings expose source, block, tolerance, readback, or seal rows separately, while $\mathsf{KnoppTailCondensation}^{\uparrow}$ binds them into one finite tail-condensation handoff surface, so no row-preserving bijection exists.

Closure status for $\mathsf{NameCert}_{\mathsf{KnoppTailCondensation}^{\uparrow}}$.

Constructive story: A $\mathsf{KnoppTailCondensation}^{\uparrow}$ packet is generated from a displayed $\mathsf{RealSeries}^{\uparrow}$ source row, finite tail window, Cauchy-condensation block row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KnoppTailCondensation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8648.1, Definition 8648.2, and Definition 8648.3 bind the finite tail-condensation carrier and its Real-and-Completion handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{KnoppTailCondensation}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.KnoppTailCondensationUp`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean theorem verification, host summability, selected sums, quotient equality, countable choice, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-series exposure, condensation-block exposure, dyadic tolerance readback, RegSeqRat handoff, RealUp seal delivery, route stability, provenance, and non-escape for the same finite packet.

Chapter 8649

A Concrete Naming Certificate for $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Bourbaki's finite-neighbourhood reading of filters, the $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ packet records an interval-indexed Cauchy filter around displayed real names as finite BEDC NameCert data. It sits between the terminal real-name seal of chapter 38, the filter surface of chapter 333, the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, and the metric comparison row of chapter 57. The packet names only the finite interval-filter route by which Real and Completion consumers inspect a Cauchy neighbourhood; it does not select a limit, quotient filter, host real equality, open-cover compactness principle, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness.

Definition 8649.1 (RealIntervalCauchyFilter carrier)**.** *A* $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, F, D, W, Q, M, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row naming the real endpoint or center data whose interval neighbourhoods are inspected.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *finite filter-basis row, listing the accepted interval-neighbourhood requests and their tail order.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording radii, refinements, and dyadic containment comparisons for the inspected intervals.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that opens only the stream prefixes needed by the filter request.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row that records interval containment and diameter checks before any Real-facing handoff is exposed.*
7. *H records componentwise* hsame *transport for the Real source, filter basis, dyadic tolerance row, stream window, regular readback, metric comparison, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an interval-filter request through R, F, D, W, Q, M, H.*

9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCauchyFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8649.2 (RealIntervalCauchyFilter NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ *carrier $I = (R, F, D, W, Q, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCauchyFilter}^{\uparrow}}$ *surface consists exactly of Real source admission through R, finite Cauchy filter-basis exposure through F, dyadic interval tolerance exposure through D, stream-window exposure through W, regular rational readback through Q, metric interval comparison through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. A consumer therefore reads interval Cauchy-filter data only through the finite route*

$$R, F, D, W, Q, M \longmapsto H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8649.1. The rows R and F expose the Real source and finite filter-basis request before any interval read is available.

The dyadic tolerance row D, stream window W, regular readback Q, and metric comparison row M then display the interval containment route. These rows make the inspected neighbourhood finite and visible rather than a hidden quotient filter or selected completion object.

Finally the structural rows H, C, P, N transport, replay, source, and name the same carrier. Since the carrier has no coordinate for quotient filters, selected limits, countable choice, host real equality, open-cover compactness, or ambient Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8649.3 (RealIntervalCauchyFilter tail basis)**.** *Every accepted* $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ *tail read factors through the same finite filter-basis row F, dyadic tolerance row D,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and* $\mathsf{Met}^{\uparrow}$ *comparison row M before H, C, P, N may replay or name the read. In particular, interval refinement is a displayed basis operation inside $I = (R, F, D, W, Q, M, H, C, P, N)$ and not an appeal to an ambient neighbourhood filter or countable family of representatives.*

Proof. Start with the NameCert obligations of Definition 8649.2. They force every accepted tail read to expose the filter-basis row F and the dyadic interval row D before the route can reach stream or rational readback data.

The finite window W and regular readback Q then inspect only the prefixes and rational values listed by the carrier. The metric comparison row M records containment and diameter checks on that same displayed basis.

The rows H, C, P, N preserve the basis route under transport, continuation replay, provenance, and local naming. Thus no ambient neighbourhood filter, quotient representative, selected limit, or countable-choice schedule is exported by the tail read. □

Theorem 8649.4 (RealIntervalCauchyFilter Real handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ *packet must consume the displayed route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Met}^{\uparrow}$$

through the carrier rows $R, F, D, W, Q, M, H, C, P, N$. The handoff supplies a finite interval-filter witness for Real and Completion neighbours, but it exports no quotient filter, selected limit, host real

equality, open-cover compactness theorem, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply the tail-basis theorem, Definition 8649.3. It keeps every interval-filter read on the displayed basis row F, dyadic tolerance row D, stream window W, regular readback Q, and metric comparison row M.

The Real source row R is therefore consumed together with the same finite basis route. No consumer can detach the terminal Real read from the filter, dyadic, stream, rational, and metric rows that make the interval request visible.

Finally use the structural rows H, C, P, N from the NameCert obligations. They replay and name the same finite packet, so the handoff cannot introduce quotient filters, selected limits, host real equality, open-cover compactness, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ read exposes a RealUp source row, a finite CauchyFilterUp basis row, a DyadicRatCoreUp interval tolerance row, a StreamNameUp finite window, a RegSeqRatUp readback row, a MetricUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real-facing handoff is visible.

Independence witness: real, cauchyfilter, dyadicratcore, streamname, regseqrat, metric: RealIntervalCauchyFilter is not bijective to any listed sibling because it jointly binds Real interval source data, finite filter-basis rows, dyadic tolerance rows, finite stream windows, regular rational readback, metric interval comparisons, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalCauchyFilter}^{\uparrow}$ packet is generated from a RealUp source row, a finite CauchyFilterUp basis row, a DyadicRatCoreUp interval tolerance row, a StreamNameUp finite window, a RegSeqRatUp readback row, a MetricUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8649.1 and Definition 8649.2 seed the finite interval Cauchy-filter carrier over RealUp, CauchyFilterUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, MetricUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host real equality, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite filter-basis exactness, dyadic interval tolerance stability, StreamName window admission, RegSeqRat readback, Metric comparison nonescape, replay stability, and local naming for BEDC.Derived.RealIntervalCauchyFilterUp.

Chapter 8650

A Concrete Naming Certificate for $\mathsf{CartanDieudonne}^{\uparrow}$

Following Cartan and Dieudonne's finite-dimensional reflection theorem for nondegenerate quadratic spaces, the $\mathsf{CartanDieudonne}^{\uparrow}$ packet records the finite reflection factorization boundary as BEDC NameCert data. It sits beside the bilinear-form source of chapter 146, the Clifford word package of chapter 147, the even reflection route of chapter 148, the full parity route of chapter 149, and the finite reflection tuple of chapter 993. These siblings expose quadratic-pairing rows, Clifford products, Spin and Pin parity rows, and tuple-image readback; this chapter binds them into a seed certificate for a finite reflection ledger and its orthogonal-action boundary.

Definition 8650.1 (Cartan–Dieudonne carrier)**.** *A* $\mathsf{CartanDieudonne}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (V, Q, B, R, W, O, S, P, H, T, N)$$

with the following displayed rows:

1. *V is the displayed vector-space or module source row whose finite-dimensional boundary is being inspected.*
2. *Q is the quadratic-form row read through the diagonal and symmetry data of* $\mathsf{BilinForm}^{\uparrow}$*.*
3. *B is the nondegenerate bilinear-pairing row that keeps radical and orthogonality tests visible.*
4. *R is the finite reflection tuple, listing the reflection rows that generate the inspected isometry.*
5. *W is the Clifford word row that types the same reflection product inside* $\mathsf{Clifford}^{\uparrow}$*.*
6. *O is the orthogonal-action boundary row, recording that the reflected endpoints preserve the displayed quadratic row Q.*
7. *S records the Spin and Pin sibling reads needed to distinguish even reflection products from the full reflection-parity surface.*
8. *P is the* Pkg *provenance over* $\mathsf{BilinForm}^{\uparrow}$*,* $\mathsf{Clifford}^{\uparrow}$*,* $\mathsf{SpinGroup}^{\uparrow}$*,* $\mathsf{PinGroup}^{\uparrow}$*,* $\mathsf{FiniteReflectionTuple}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *H records componentwise* hsame *transport for the vector source, quadratic row, bilinear row, reflection tuple, Clifford word, orthogonal boundary, parity row, provenance, replay, and local naming.*
10. *T records displayed* Cont *replay from an orthogonal-action request through $V, Q, B, R, W, O, S, P, H$.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CartanDieudonne}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient orthogonal group, a host matrix representation, a selected basis, a quotient Clifford algebra, infinite reflection products, or analytic spectral diagonalization.

Theorem 8650.2 (Cartan–Dieudonne NameCert obligations)**.** *For every accepted* $\mathsf{CartanDieudonne}^{\uparrow}$ *packet $C = (V, Q, B, R, W, O, S, P, H, T, N)$, the local* $\mathsf{NameCert}_{\mathsf{CartanDieudonne}^{\uparrow}}$ *surface consists exactly of carrier admission for the vector source V, quadratic row Q, nondegenerate bilinear row B, finite reflection tuple R, Clifford word row W, orthogonal-action boundary O, Spin–Pin parity row S, package provenance P, componentwise transport H, displayed replay T, and local naming row N. Any consumer that reads a finite reflection factorization must keep all of these rows visible before treating the action as an orthogonal boundary.*

Proof. Project the carrier of Definition 8650.1. The source geometry is displayed by V, Q, B: the vector or module row is named, the quadratic row is read through the bilinear-form sibling, and the nondegenerate pairing row records the orthogonality tests used by each reflection.

Next read the factorization rows R, W, O, S. The tuple row R lists the finite reflection factors, the Clifford word row W places their product in the Clifford sibling, the orthogonal-action row O records preservation of Q, and the parity row S keeps the Spin and Pin readings separate.

Finally inspect P, H, T, N. Provenance, transport, replay, and local naming preserve the same finite reflection ledger. Since the carrier has no coordinate for an ambient orthogonal group, selected basis, host matrix action, quotient Clifford algebra, infinite product, or analytic diagonalization, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{CartanDieudonne}^{\uparrow}$ reflection read exposes a vector source row, quadratic row, nondegenerate bilinear row, finite reflection tuple, Clifford word row, orthogonal-action boundary, Spin–Pin parity row, package provenance, componentwise hsame transport, displayed Cont replay, and local naming before any orthogonal-facing consumer can inspect the factorization.

Independence witness: bilinform, clifford, spingroup, pingroup, finitereflectiontuple: $\mathsf{BilinForm}^{\uparrow}$ supplies pairing rows without a reflection ledger; $\mathsf{Clifford}^{\uparrow}$ supplies word and quadratic-boundary rows without an orthogonal factorization boundary; $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{PinGroup}^{\uparrow}$ supply parity-specific reflection surfaces without the Cartan–Dieudonne finite factorization ledger; $\mathsf{FiniteReflectionTuple}^{\uparrow}$ supplies tuple image and readback without the bilinear, Clifford, and parity obligations bound into one carrier. None is bijective to $\mathsf{CartanDieudonne}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CartanDieudonne}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CartanDieudonne}^{\uparrow}$ packet is generated from a finite vector or module source row, quadratic row, nondegenerate bilinear row, finite reflection tuple, Clifford word row, orthogonal-action boundary, Spin–Pin parity row, package provenance, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CartanDieudonne}^{\uparrow}}$, seed)

Scope closed: Definition 8650.1 and Definition 8650.2 seed the finite reflection-factorization packet over BilinFormUp, CliffordUp, SpinGroupUp, PinGroupUp, FiniteReflectionTupleUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CartanDieudonne}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, an ambient orthogonal group, a selected basis, host matrix representation, quotient Clifford algebra, infinite reflection products, or analytic spectral diagonalization.

Upgrade path: Obligation closure requires checked carrier admission, finite reflection tuple exactness, bilinear nondegeneracy visibility, Clifford word compatibility, Spin–Pin parity separation, orthogonal-action preservation, transport, replay, provenance, and local naming for BEDC.Derived.CartanDieudonneUp.

Chapter 8651

A Concrete Naming Certificate for $\mathsf{ClarkeSubdifferential}^{\uparrow}$

Following Clarke's nonsmooth analysis and Rockafellar's convex-analysis presentation of generalized gradients, the $\mathsf{ClarkeSubdifferential}^{\uparrow}$ packet records the finite locally Lipschitz slope surface needed to expose a convexified generalized-gradient read inside BEDC. It sits beside the subgradient support surface of chapter 3712, the feasible-direction surface of chapter 3596, the convex carrier of chapter 208, the terminal real and norm rows of chapter 38 and chapter 89, and the stationarity and duality consumers of chapter 300 and chapter 321. The packet names a finite Clarke-style generalized-gradient certificate; it does not assert a host differentiability theorem, selected limiting gradient, measurable selection, ambient compactness, or quotient dual-space equality.

8651.1 ClarkeSubdifferential Carrier

Definition 8651.1 (ClarkeSubdifferential carrier)**.** *A* $\mathsf{ClarkeSubdifferential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, F, x, W, D, T, S, V, R, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *ambient row in which base points, direction rows, and slope comparisons are read.*
2. *F is the locally Lipschitz functional row, including the finite* $\mathsf{LipschitzModulus}^{\uparrow}$ *windows and value-comparison rows actually inspected by the packet.*
3. *x is the base-point row admitted by the displayed* $\mathsf{ConvexSet}^{\uparrow}$ *domain of F.*
4. *W is the finite locally Lipschitz window ledger around x, listing the inspected radii, norm bounds, and Real-valued comparison budgets.*
5. *D is the directional-slope ledger. It records the displayed direction rows and the upper finite slope comparisons read from the rows $F(x + hv) - F(x)$ over the windows in W.*
6. *T is the* $\mathsf{TangentCone}^{\uparrow}$ *row exposing the feasible-direction surface used by constrained consumers.*
7. *S is the* $\mathsf{Subdifferential}^{\uparrow}$ *support row, read only as a sibling support ledger and not as an ambient choice of subgradient.*

8. *V is the convex-hull and provenance row for the finite slope witnesses, recording which displayed direction and support rows contribute to each generalized-gradient candidate.*
9. *R is the stationarity and duality readback row consumed by* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *only after D, T, S, V are visible.*
10. *H records componentwise* hsame *transport for the ambient, functional, base point, Lipschitz windows, directional slopes, tangent-cone row, support row, convex-hull provenance, readback, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a generalized-gradient request through $X, F, x, W, D, T, S, V, R, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{ClarkeSubdifferential}^{\uparrow}$, $\mathsf{Subdifferential}^{\uparrow}$, $\mathsf{TangentCone}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LipschitzModulus}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClarkeSubdifferential}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $X, F, x, W, D, T, S, V, R, H, C, P, N$. It has no coordinate for a selected differentiability point, a host limiting-gradient closure, measurable selection, quotient dual-space equality, ambient compactness, or a global nonsmooth chain rule.

Lean4 (definition) `BEDC.Derived.ClarkeSubdifferentialUp.ClarkeSubdifferentialUp`

8651.2 ClarkeSubdifferential Certificate

Theorem 8651.2 (ClarkeSubdifferential NameCert obligations)**.** *For every accepted* $\mathsf{ClarkeSubdifferential}^{\uparrow}$ *carrier $G = (X, F, x, W, D, T, S, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClarkeSubdifferential}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the ambient vector, metric, norm, and real rows; the locally Lipschitz functional row; the base point; the finite window ledger; the directional-slope ledger; the tangent-cone row; the sibling support row; the convex-hull provenance row; the stationarity and duality readback row; transport; replay; provenance; and local naming;*
2. *Lipschitz-window exactness, requiring every exported slope comparison to pass through the displayed windows W and the value-comparison rows of F;*
3. *directional-slope exposure, requiring every generalized-gradient read to display the direction row and the finite upper-slope comparison stored in D;*
4. *tangent and support compatibility, requiring constrained reads to pass through T and support-facing reads to pass through S before the convex-hull row V may be inspected;*
5. *convex-hull provenance, requiring each generalized-gradient candidate to list the displayed slope and support rows used by its finite convex combination;*
6. *non-escape from selected limiting gradients, quotient dual spaces, measurable selections, host differentiability theorems, ambient compactness, and global nonsmooth calculus rules.*

Proof. Project the accepted carrier of Definition 8651.1. The analytic rows are $X, F, x, W, D, T, S, V, R$, and the structural rows are H, C, P, N.

Lipschitz-window exactness is read first. A slope comparison can be exported only when the relevant norm, real, and value-comparison rows are visible through X, F, W.

The directional row then passes through D. Constrained consumers also expose the tangent-cone row T, and support-facing consumers expose the subdifferential row S, before the convex-hull provenance row V may name a generalized-gradient candidate.

Finally R hands the same finite route to KKT and Fenchel-duality consumers, while H, C, P, N transport, replay, source, and name the packet. Since no coordinate of G stores a selected limiting gradient, quotient dual space, measurable selection, host differentiability theorem, compactness theorem, or global nonsmooth calculus rule, the listed rows exhaust the local certificate. □

Theorem 8651.3 (ClarkeSubdifferential Lipschitz-gradient route)**.** *Every* $\mathsf{KKT}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, *or* $\mathsf{Subdifferential}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClarkeSubdifferential}^{\uparrow}$ *packet must factor through the displayed route*

$$X, F, x, W, D, T, S, V, R \longmapsto H, C, P, N.$$

In particular, a generalized-gradient read is public only after the locally Lipschitz window ledger, the finite directional-slope rows, the tangent-cone row, the sibling support row, and the convex-hull provenance row have all been displayed. The route supplies no selected limiting gradient, no host differentiability theorem, no quotient dual-space equality, no measurable selection, and no ambient completeness principle.

Proof. Begin with the NameCert obligations of Definition 8651.2. They force every generalized-gradient read to expose the locally Lipschitz window row W and the directional-slope ledger D before any candidate gradient row is available.

The constrained and support-facing boundaries are then read through T and S. These rows keep the Clarke packet attached to the tangent-cone and subdifferential siblings rather than to a host limiting-gradient construction.

The convex-hull row V records only the displayed finite provenance, and R hands that same provenance to KKT or Fenchel-duality consumers. Transport, replay, package provenance, and local naming preserve the route through H, C, P, N, so the consumer cannot bypass the finite Lipschitz, slope, tangent, support, and convex-hull rows. □

Falsifiable prediction: Every accepted $\mathsf{ClarkeSubdifferential}^{\uparrow}$ read exposes VecSpaceUp, MetricUp, NormUp, RealUp, locally Lipschitz window rows, finite directional-slope rows, TangentCone rows, SubdifferentialUp support rows, convex-hull provenance rows, KKTUp or FenchelDualityUp readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any generalized-gradient output is visible.

Independence witness: subdifferential, tangent-cone, convexset, real, norm, kkt, fenchelduality: $\mathsf{ClarkeSubdifferential}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds locally Lipschitz windows, directional-slope ledgers, feasible-direction rows, support rows, convex-hull provenance, stationarity and duality readback, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClarkeSubdifferential}^{\uparrow}}$.

Constructive story: A $\mathsf{ClarkeSubdifferential}^{\uparrow}$ packet is generated from vector, metric, norm, Real, locally Lipschitz, base-point, directional-slope, TangentCone, SubdifferentialUp support, convex-hull provenance, KKTUp or FenchelDualityUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClarkeSubdifferential}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8651.1, Definition 8651.2, and Definition 8651.3 seed the finite nonsmooth generalized-gradient certificate surface over locally Lipschitz windows, directional slopes, tangent cones, support rows, convex hulls, and optimization consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClarkeSubdifferential}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host limiting-gradient theorem, selected differentiability points, measurable selection, quotient dual-space equality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz-window exactness, directional-slope exposure, tangent and support compatibility, convex-hull provenance, KKT and Fenchel readback, replay stability, and local naming for BEDC.Derived.ClarkeSubdifferentialUp.

Chapter 8652

A Concrete Naming Certificate for $\mathsf{FiniteLatticePath}^{\uparrow}$

Following Stanley's enumerative combinatorics, the $\mathsf{FiniteLatticePath}^{\uparrow}$ packet records a finite grid walk by endpoint, step-list, and prefix-ledger rows. It sits beside the unary size carrier of chapter 28, the finite spine carrier of chapter 36, the ordered-pair carrier of chapter 34, and the comparison rows of chapter 1378; its Dyck-prefix special case is consumed by the Catalan surface of chapter 8503. The packet names only displayed finite walks, not an ambient grid, infinite path space, quotient endpoint equality, or external enumeration theorem.

Definition 8652.1 (Finite lattice path carrier)**.** *A* $\mathsf{FiniteLatticePath}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (s, t, W, E, P, B, H, C, G, N)$$

with the following displayed rows:

1. *s is the start endpoint row, a* $\mathsf{Prod}^{\uparrow}$ *pair of* $\mathsf{Nat}^{\uparrow}$ *coordinates.*
2. *t is the terminal endpoint row, another* $\mathsf{Prod}^{\uparrow}$ *pair of* $\mathsf{Nat}^{\uparrow}$ *coordinates.*
3. *W is the finite* $\mathsf{List}^{\uparrow}$ *step row whose entries are the displayed horizontal and vertical unit steps.*
4. *E is the endpoint ledger: iterating the steps of W from s reaches t through finite* $\mathsf{Prod}^{\uparrow}$ *coordinate updates.*
5. *P is the prefix ledger: every prefix of W has a displayed* $\mathsf{Prod}^{\uparrow}$ *endpoint and an* $\mathsf{Order}^{\uparrow}$ *comparison row whenever a boundary condition is part of the path request.*
6. *B is the bounded-size row, read through* $\mathsf{Nat}^{\uparrow}$*, that records the length of W and the coordinate budgets consumed by E and P.*
7. *H records componentwise* hsame *transport for the endpoints, step list, endpoint ledger, prefix ledger, bounded-size row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from endpoint and prefix requests to the finite step-list ledger.*
9. *G is the* Pkg *provenance over* $\mathsf{FiniteLatticePath}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*,* $\mathsf{CatalanNumber}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a completed lattice, infinite walk, quotient endpoint equality, unbounded search, or host enumeration principle.

Theorem 8652.2 (Finite lattice path NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLatticePath}^{\uparrow}$ *carrier* $L = (s, t, W, E, P, B, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}$ *surface consists exactly of start and terminal endpoint admission through* $\mathsf{Prod}^{\uparrow}$ *pairs of* $\mathsf{Nat}^{\uparrow}$ *coordinates, finite step-list admission through* $\mathsf{List}^{\uparrow}$*, endpoint-ledger exactness, prefix-ledger visibility, bounded-size accounting, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from ambient grids, infinite path spaces, quotient endpoint equality, unbounded search, or external enumeration theorems.*

Proof. Project the accepted packet. The start and terminal rows are displayed $\mathsf{Prod}^{\uparrow}$ pairs of $\mathsf{Nat}^{\uparrow}$ coordinates, and the step row W is a finite $\mathsf{List}^{\uparrow}$ spine.

Read the endpoint ledger E. It iterates the displayed steps from s and records the terminal endpoint t as a finite coordinate calculation, so no ambient lattice carrier or quotient endpoint equality is available.

Read the prefix ledger P and bounded-size row B. Each prefix endpoint is displayed as a $\mathsf{Prod}^{\uparrow}$ pair, and each boundary comparison is an $\mathsf{Order}^{\uparrow}$ row attached to that prefix. The length and coordinate budgets are $\mathsf{Nat}^{\uparrow}$ rows, hence the accepted path is finite at every inspected prefix.

Finally H, C, G, N transport, replay, source, and name this same route. Since the carrier has no coordinate for an infinite walk, unbounded search, or external enumeration theorem, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 8652.3 (Finite lattice path endpoint and prefix ledger route)**.** *Every downstream consumer of an accepted* $\mathsf{FiniteLatticePath}^{\uparrow}$ *packet must read the route*

$$\mathsf{Nat}^{\uparrow},\ \mathsf{Prod}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{Order}^{\uparrow} \longmapsto (s, t, W, E, P, B) \longmapsto \mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}$$

before it may consume the path as a Catalan, Dyck-prefix, ballot, grid-walk, or finite-enumeration object. The $\mathsf{CatalanNumber}^{\uparrow}$ *consumer receives only the special case where* P *records the nonnegative prefix balance and zero terminal balance; it does not receive an unrestricted path space or an analytic counting theorem.*

Proof. Start with a consumer of the accepted packet L. Carrier admission exposes the endpoint rows s, t, the finite step list W, the endpoint ledger E, the prefix ledger P, and the bounded-size row B before the local name row can be read.

The endpoint rows are $\mathsf{Prod}^{\uparrow}$ pairs whose coordinates are $\mathsf{Nat}^{\uparrow}$ rows. The step list is a $\mathsf{List}^{\uparrow}$ spine, so any scan of the path factors through the finite list carrier rather than through an ambient function space.

The prefix ledger is the only place where order information appears. When a consumer asks for a boundary condition, it receives an $\mathsf{Order}^{\uparrow}$ comparison attached to a displayed prefix endpoint, not a detached order relation on a host grid.

For the Catalan-facing special case, the same prefix ledger records balance nonnegativity and terminal balance zero. Thus the Catalan packet can consume the lattice-path row as a Dyck-prefix surface, while the finite lattice path packet itself remains the endpoint-and-prefix ledger route described above. □

Falsifiable prediction: Every accepted $\mathsf{FiniteLatticePath}^{\uparrow}$ read must expose the start endpoint, terminal endpoint, finite step list, endpoint ledger, prefix ledger, bounded-size row, componentwise

hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before it may be consumed by Catalan, Dyck-prefix, ballot, grid-walk, or finite-enumeration chapters.

Independence witness: nat, list, prod, order, catalan-number: $\mathsf{FiniteLatticePath}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, a finite step spine, endpoint iteration, prefix comparisons, bounded-size accounting, transport, replay, provenance, and local naming. NatUp has sizes without path spines; ListUp has spines without coordinate endpoints; ProdUp has endpoint pairs without iteration; OrderUp has comparisons without prefix history; CatalanNumberUp consumes a Dyck-prefix special case without carrying arbitrary finite grid endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteLatticePath}^{\uparrow}$ packet is generated from NatUp coordinate rows, ProdUp endpoint pairs, a finite ListUp step spine, endpoint iteration, prefix OrderUp comparisons, bounded-size accounting, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8652.1, Definition 8652.2, and Definition 8652.3 seed the finite endpoint and prefix-ledger route for lattice-path packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLatticePath}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Catalan recurrence uniqueness, ballot enumeration, reflection principles, quotient endpoint equality, infinite path spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-ledger exactness, prefix-ledger visibility, bounded-size accounting, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLatticePathUp.

Chapter 8653

A Concrete Naming Certificate for $\mathsf{LevyContinuity}^{\uparrow}$

Following Feller's characteristic-function proof of weak convergence and Billingsley's finite-dimensional distribution conventions, the $\mathsf{LevyContinuity}^{\uparrow}$ packet records the finite BEDC surface used by a Levy-continuity readback without importing completed measure spaces or host almost-sure quotient equality. It sits over the probability source of chapter 184, the random-variable observation surface of chapter 185, the distribution readback of chapter 186, the rational regular-window discipline of chapter 301, and the terminal real seal of chapter 38. The characteristic-function rows are finite windows: each accepted read exposes only the inspected frequency grid, tolerance ledger, distribution readback, transport, replay, provenance, and local naming rows.

Definition 8653.1 (Levy continuity carrier)**.** *A* $\mathsf{LevyContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, X, D, W, Q, R, H, C, G, N)$$

with the following rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *probability-source row for the inspected finite event algebra and probability requests.*
2. *X is the* $\mathsf{RandomVar}^{\uparrow}$ *observation row, exposing the displayed random-variable reads used by the packet.*
3. *D is the* $\mathsf{Distribution}^{\uparrow}$ *readback row, recording the finite distribution observations attached to X.*
4. *W is the* $\mathsf{CharacteristicFunction}^{\uparrow}$ *finite-window row, naming the inspected frequency grid and the complex-valued read requests admitted on that grid.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row for the finite characteristic-function and distribution comparisons.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after P, X, D, W, Q remain visible.*
7. *H stores componentwise* hsame *transport for the probability source, random-variable observation, distribution readback, characteristic-function window, tolerance ledger, real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from probability reads to random-variable observations, from observations to distribution rows, and from distribution rows to characteristic-function finite windows.*

9. *G is the* Pkg *provenance over* $\mathsf{LevyContinuity}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{CharacteristicFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LevyContinuity}^{\uparrow}}$ *row.*

The accepted carrier stores no measure-completion coordinate, no host almost-sure quotient equality, no selected limit distribution, no countable-choice selector, and no ambient Real completeness witness.

Lean4 (statement) `BEDC.Derived.LevyContinuityUp`

Theorem 8653.2 (Levy continuity NameCert obligations)**.** *For an accepted packet $L = (P, X, D, W, Q, R, H, C, G,$ the local* $\mathsf{NameCert}_{\mathsf{LevyContinuity}^{\uparrow}}$ *obligations are exhausted by probability-source admission through P, random-variable observation through X, distribution readback through D, characteristic-function finite windows through W, rational tolerance exposure through Q, terminal real sealing through R, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8653.1. The probability row P and random-variable row X expose only the finite probability and observation requests stored in the packet.

Read the analytic rows next. The distribution row D is available before the characteristic-function window W, and W records only the inspected frequency grid together with the tolerance ledger Q.

Finally inspect R, H, C, G, N. These rows seal, transport, replay, package, and name the same displayed data. Since the carrier has no coordinate for measure completion, host almost-sure quotient equality, a selected limit law, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no local NameCert obligation can read one. □

Theorem 8653.3 (Levy continuity characteristic-function route)**.** *Every distribution-facing consumer of an accepted* $\mathsf{LevyContinuity}^{\uparrow}$ *packet must expose the finite route*

$$\mathsf{ProbSpace}^{\uparrow}, \mathsf{RandomVar}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto \mathsf{CharacteristicFunction}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the transport, replay, provenance, and local naming rows can be consumed. Thus characteristic-function convergence is read only through displayed finite windows and their rational tolerance ledger, not through completed measure objects, host equality of random variables, selected weak limits, or ambient real completeness.

Proof. Start with the NameCert obligations of Definition 8653.2. They force every accepted read to expose P, X, D, W, Q before the terminal real seal and structural rows are available.

The characteristic-function row W is therefore a finite-window handoff between the distribution readback D and the rational tolerance row Q. Its continuation replay C keeps the probability, observation, distribution, and window reads in the same displayed packet.

The structural rows H, G, N preserve that route by transport, provenance, and local naming. Since no completed measure carrier, almost-sure quotient equality, selected weak limit, countable-choice selector, or ambient completeness row appears in Definition 8653.1, no distribution-facing consumer can bypass the finite characteristic-function route. □

Falsifiable prediction: Every accepted $\mathsf{LevyContinuity}^{\uparrow}$ read exposes a $\mathsf{ProbSpace}^{\uparrow}$ source, $\mathsf{RandomVar}^{\uparrow}$ observation row, $\mathsf{Distribution}^{\uparrow}$ readback, $\mathsf{CharacteristicFunction}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any downstream weak-convergence or central-limit packet may consume it.

Independence witness: probspace, randomvar, distribution, regseqrat, real: none is bijective to LevyContinuityUp because the packet jointly carries probability-source, random-variable, distribution-readback, characteristic-window, tolerance, seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LevyContinuity}\uparrow}$.

Constructive story: A $\mathsf{LevyContinuity}^{\uparrow}$ packet is generated from a probability source row, random-variable observation row, distribution readback, characteristic-function finite windows, a rational tolerance ledger, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LevyContinuity}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8653.1, Definition 8653.2, and Definition 8653.3 seed the finite Levy-continuity carrier between probability sources, random-variable observations, distribution readbacks, characteristic-function windows, RegSeqRat tolerances, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LevyContinuity}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, almost-sure quotient equality, selected weak limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, characteristic-window exactness, distribution readback stability, tolerance-ledger stability, and downstream nonescape for BEDC.Derived.LevyContinuityUp.

Chapter 8654

A Concrete Naming Certificate for $\mathsf{DieudonneCompletion}^{\uparrow}$

Following Dieudonne, Bourbaki, and Kelley on completion of uniform spaces, the $\mathsf{DieudonneCompletion}^{\uparrow}$ packet records the finite BEDC boundary by which a displayed uniform source hands Cauchy-filter and Cauchy-net evidence to the separated-completion corridor. It sits beside the uniform-space completion route of chapter 3560, the separated-completion boundary of chapter 1612, the finite stream-name windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a traditional uniform-completion interface without quotient completion objects, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8654.1 (Dieudonne completion carrier)**.** *A* $\mathsf{DieudonneCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, F, T, S, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, recording the displayed entourage base, finite refinement reads, and uniform source data admitted by chapter 3560.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing finite filter-base demands and Cauchy eventuality checks over the same uniform source.*
3. *T is the* $\mathsf{CauchyNet}^{\uparrow}$ *row, exposing directed Cauchy windows when the same completion request is presented in Moore–Smith form.*
4. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *handoff row, recording the separated boundary consumed before a completion-facing point can be public.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the filter or net observations are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected windows into regular rational observations.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, refinement slack, and comparison budgets used by the Cauchy-filter or Cauchy-net rows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after U, F, T, S, W, R, Q have all been displayed.*

9. *H records componentwise* hsame *transport for the uniform source, Cauchy-filter row, Cauchy-net row, separated handoff, finite windows, regular readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a uniform-completion request through*

$$U, F, T, S, W, R, Q, E \longmapsto \mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}.$$

11. *P is the* Pkg *provenance over* $\mathsf{DieudonneCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, T, S, W, R, Q, E, H, C, P, N$. It has no coordinate for quotient completion classes, selected Cauchy representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8654.2 (Dieudonne completion NameCert obligations)**.** *For every accepted* $\mathsf{DieudonneCompletion}^{\uparrow}$ *carrier*

$$D = (U, F, T, S, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform source row, Cauchy-filter row, Cauchy-net row, separated-completion handoff, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any completion-facing read must keep these rows visible before it consumes the separated boundary or the Real seal.*

Proof. Project the carrier of Definition 8654.1. The rows U, F, T, S, W, R, Q, E are the only analytic coordinates through which a consumer can read uniform source data, Cauchy evidence, separated completion data, finite windows, rational readback, dyadic tolerance, and the terminal Real seal.

The uniform side begins at U. Filter-style evidence is read through F, while directed-net evidence is read through T; both remain attached to the same displayed entourage and refinement rows rather than to a quotient of Cauchy presentations.

The separated and numerical side then passes through S, W, R, Q, E. The separated-completion handoff is displayed before the public completion boundary is consumed, and the finite windows, regular rational readback, dyadic ledger, and Real seal keep the route inspectable by Real and Completion consumers.

Finally H, C, P, N transport, replay, source, and locally name the same packet. Since the carrier contains no coordinate for quotient completion classes, selected representatives, countable choice, host equality, or ambient Real completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8654.3 (Dieudonne completion uniform handoff)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{DieudonneCompletion}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto (\mathsf{CauchyFilter}^{\uparrow}, \mathsf{CauchyNet}^{\uparrow}) \longmapsto \mathsf{SeparatedCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The consumer first reads the uniform source, then the displayed filter or net Cauchy evidence, then the separated-completion handoff, and only then the finite-window, rational-readback, dyadic-ledger, and Real-seal rows.

Proof. Begin with the NameCert obligations of Definition 8654.2. They place every public read inside the carrier rows $U, F, T, S, W, R, Q, E, H, C, P, N$.

The first step is uniform. A filter presentation uses F, and a net presentation uses T, but both are read over the same displayed source row U and share the same downstream packet boundary.

The separated handoff is the next public boundary. The row S must be displayed before a completion-facing consumer can treat the Cauchy evidence as separated data, so the route cannot skip to a completed point by quotienting filters or nets.

The final read is numerical and structural. The windows W, regular readback R, dyadic ledger Q, and Real seal E are replayed through H, C, P, N, leaving no route for selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness to enter the packet. □

Falsifiable prediction: Every accepted $\mathsf{DieudonneCompletion}^{\uparrow}$ completion read exposes a UniformSpace source row, a CauchyFilter or CauchyNet row, a SeparatedCompletion handoff, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniform-space-completion, separatedcompletion, streamname, regseqrat, dyadicratcore, real: $\mathsf{DieudonneCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a uniform source, filter-style and net-style Cauchy evidence, separated-completion handoff, finite windows, rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local naming in one uniform-completion boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DieudonneCompletion}^{\uparrow}$ packet is generated from a UniformSpace source row, CauchyFilter and CauchyNet evidence rows, a SeparatedCompletion handoff, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8654.1, Definition 8654.2, and Definition 8654.3 seed the finite traditional uniform-completion route from displayed uniform data through Cauchy-filter or Cauchy-net evidence, separated handoff, finite windows, rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DieudonneCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected Cauchy representatives, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exposure, Cauchy-filter and Cauchy-net evidence, separated-completion handoff, finite-window inspection, regular rational readback, dyadic tolerance accounting, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.DieudonneCompletionUp.

Chapter 8655

A Concrete Naming Certificate for $\mathsf{UniformFilterBase}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces and filter bases, the $\mathsf{UniformFilterBase}^{\uparrow}$ packet records the finite L10 route by which a displayed uniform-entourage request is refined through a directed filter-base row before it is handed to Cauchy-filter and real-completion consumers. It sits over the filter-base discipline of chapter 1601, the uniform-space entourage discipline of chapter 339, the Cauchy-filter boundary of chapter 333, the finite stream-name observations of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38.

Definition 8655.1 (Uniform filter-base carrier)**.** *A* $\mathsf{UniformFilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, B, C, S, R, D, A, H, K, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage request row naming the requested uniform neighbourhood.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *directed refinement row, carrying the displayed base element and its refinement relation.*
3. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *readiness row, asserting that the directed refinement can be read by a Cauchy-filter consumer.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row for reading the selected base element.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite observations in S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius and slack.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after U, B, C, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the entourage request, directed base refinement, Cauchy readiness, readback, tolerance, terminal seal, provenance, and local name.*
9. *K records the displayed* Cont *replay by which a consumer reads $U, B, C, S, R, D, A, H, P, N$ in order.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformFilterBase}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{UniformFilterBase}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient filters, selected limits, countable choice, host equality of filters, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8655.2 (Uniform filter-base NameCert obligations)**.** *For the carrier of Definition 8655.1, the local* $\mathsf{NameCert}_{\mathsf{UniformFilterBase}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (U, B, C, S, R, D, A, H, K, P, N),$$

entourage-request exposure through U, directed filter-base refinement through B, Cauchy readiness through C, finite observation and readback through S, R, dyadic tolerance through D, terminal real sealing through A, componentwise hsame *stability through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 8655.1. The rows U and B expose the uniform-entourage request and the directed filter-base refinement before any Cauchy or completion-facing consumer may inspect the packet.

The Cauchy and readback side is read through C, S, R, D. The Cauchy readiness row keeps the filter-base route finite, while the stream, regular rational, and dyadic rows provide the displayed observation and tolerance data needed for a real-completion handoff.

Finally A, H, K, P, N seal, transport, replay, package, and name the same displayed rows. Since the carrier contains no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8655.3 (Uniform filter-base Cauchy handoff)**.** *Every accepted* $\mathsf{UniformFilterBase}^{\uparrow}$ *packet factors any* $\mathsf{CauchyFilter}^{\uparrow}$ *read through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow},\ \mathsf{FilterBase}^{\uparrow},\ U,\ B,\ C,\ S,\ R,\ D \longrightarrow \mathsf{CauchyFilter}^{\uparrow}.$$

The handoff supplies a uniform entourage request, a directed base refinement, Cauchy readiness, finite observation windows, regular rational readback, and dyadic tolerance data; it does not supply quotient-filter equality, selected limit filters, countable choice, or host equality of filters.

Proof. Project the carrier in Definition 8655.1. The only rows available before the Cauchy read are U, B, C, S, R, D together with transport, replay, provenance, and local naming.

Read U through the uniform-space entourage surface and B through the directed filter-base surface. The row C is then a readiness row for the Cauchy-filter consumer, not a completed filter quotient or a selected limit.

The rows S, R, D provide the finite observation, rational readback, and dyadic tolerance budget attached to that same handoff. Since the carrier has no further coordinate, every accepted Cauchy read factors through the displayed route. □

Theorem 8655.4 (Uniform filter-base real-completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing completion read of an accepted* $\mathsf{UniformFilterBase}^{\uparrow}$ *packet may inspect only the terminal seal A after the rows U, B, C, S, R, D have been displayed and transported through H, K, P, N. The boundary exposes no selected real limit, no ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*, no quotient equality of filters, and no countable-choice extraction.*

Proof. Start with Definition 8655.3. It confines the Cauchy-facing side of the packet to the uniform entourage request, directed base refinement, Cauchy readiness, finite observations, regular rational readback, and dyadic tolerance rows.

The terminal row A can be read only after those rows are present. It is a seal for completion-facing consumers, not a construction of a selected real limit or a proof of ambient real completeness.

The transport and replay rows H, K preserve the same displayed route, and P, N record provenance and local naming. Thus the real-completion boundary is exactly the finite seal over the displayed uniform-filter-base data. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformFilterBase}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformFilterBase}^{\uparrow}$ packet is generated from a uniform-entourage request, a directed filter-base refinement, Cauchy readiness, finite StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformFilterBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8655.1, Definition 8655.2, Definition 8655.3, and Definition 8655.4 display the finite uniform-filter-base carrier, Cauchy handoff, and real-completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformFilterBase}^{\uparrow}}, \mathsf{unformalized})$

Lean4 (statement) `BEDC.Derived.UniformFilterBaseUp`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality of filters, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: The next step is a checked NameCert obligation theorem for the displayed carrier rows and their Cauchy-filter handoff.

Chapter 8656

A Concrete Naming Certificate for $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ packet records a finite completion-facing family of Cauchy real-name reads whose members share one displayed equicontinuity modulus on compact or bounded source windows. It sits after the Real-uniform entourage surface of chapter 1624, the finite window and readback rows of chapter 126 and chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real spine of chapter 38. The packet supplies a traditional finite-means consumer for L10 Real and Completion work without asserting ambient completeness, quotient filters, selected limits, countable choice, host equality, or bridge credentials.

Definition 8656.1 (Equicontinuous Cauchy family carrier)**.** *An* $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, U, M, T, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact or bounded source-window row, naming the finite portion of the domain on which the family is inspected.*
2. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage row consumed by the shared continuity request.*
3. *M is the shared equicontinuity modulus row, assigning one finite tolerance-to-window rule for all accepted members of the family.*
4. *T is the Cauchy-family threshold row, recording the displayed tail budget used when two family indices are compared at a requested precision.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the source row, the modulus row, and every Cauchy threshold request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the stream windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic scales, slack bounds, and comparison budgets used by U, M, T, W, R.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source window, shared modulus, Cauchy threshold, StreamName, RegSeqRat, and DyadicRatCore rows have been displayed.*

9. *H records componentwise* hsame *transport for the source windows, Real-uniform entourage, shared modulus, Cauchy thresholds, stream windows, regular readback, dyadic tolerances, Real seals, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from any completion-facing family read through*

$$K \longrightarrow U \longrightarrow M \longrightarrow T \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow A.$$

11. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousCauchyFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, U, M, T, W, R, D, A, H, C, P, N$. It has no coordinate for an ambient complete function space, a selected limiting function, quotient filter equality, countable choice, host equality, or bridge checking.

Theorem 8656.2 (Equicontinuous Cauchy family NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ *carrier*

$$E = (K, U, M, T, W, R, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuousCauchyFamily}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact or bounded source windows, Real-uniform entourage, shared equicontinuity modulus, Cauchy-family thresholds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, terminal Real seals, transport, replay, provenance, and local naming rows;*

2. *uniform-source exactness, requiring every family read to expose K and U before a modulus or Cauchy threshold is inspected;*

3. *shared-modulus exactness, requiring the same row M to answer every memberwise equicontinuity request over the displayed source windows;*

4. *Cauchy-threshold exactness, requiring T to supply only finite tail thresholds at the requested DyadicRatCore precision;*

5. *L10 readback discipline, requiring W, R, D to mediate all rational observations before any terminal* $\mathsf{Real}^{\uparrow}$ *seal A is public;*

6. *structural stability, requiring H, C, P, N to transport, replay, source, and name exactly the displayed route.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.EquicontinuousCauchyFamilyUp.EquicontinuousCauchyFamilyTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 8656.1. The accepted packet displays the source-window, Real-uniform, shared-modulus, Cauchy-threshold, stream, regular-readback, dyadic, Real-seal, transport, replay, provenance, and local naming rows.

Uniform-source exactness is the first part of the route. A completion-facing family consumer must read K and then U, so the shared modulus is interpreted over the Real-uniform entourage row of chapter 1624.

Shared-modulus and Cauchy-threshold exactness are the next projections. The row M is the only equicontinuity modulus coordinate, and T is the only Cauchy threshold coordinate, so neither can be replaced by a hidden cover, quotient filter, or chosen limit.

The L10 readback discipline is the terminal part of the route. Stream windows, regular rational readback, and dyadic tolerances appear as W, R, D before the $\mathsf{Real}^{\uparrow}$ seal row A is available.

Finally, H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no ambient complete function space, selected limiting function, quotient filter equality, countable choice, host equality, or bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 8656.3 (Equicontinuous Cauchy family completion handoff)**.** *Every L10 completion-facing consumer of an accepted* $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$K \longrightarrow U \longrightarrow M \longrightarrow T \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow A$$

with componentwise transport, replay, provenance, and local naming supplied by H, C, P, N*. The handoff may be consumed by Real and Completion chapters that read uniform Cauchy-family data, but it does not export ambient completeness, a selected limit family, quotient-filter equality, countable choice, host real equality, open-cover compactness, or bridge checking.*

Proof. Project the carrier of Definition 8656.1. The source-window row K fixes the compact or bounded part of the domain, and the Real-uniform row U fixes the entourage request before any family threshold is read.

Apply Definition 8656.2. The shared modulus row M and Cauchy threshold row T are required before the finite observation rows W, R, D and before the terminal Real seal.

The L10 Real spine is then the same finite window-readback-seal route used by chapter 38. The completion consumer receives A only after StreamName windows, RegSeqRat readback, and DyadicRatCore tolerances have been displayed.

The structural rows H, C, P, N do not add analytic data. They transport, replay, source, and name the route already displayed, so no ambient completeness theorem, selected limit family, quotient-filter equality, countable-choice schedule, host equality, open-cover compactness, or bridge credential can be exported. □

Theorem 8656.4 (Equicontinuous Cauchy family tail stability)**.** *Let* $E = (K, U, M, T, W, R, D, A, H, C, P, N)$ *and* $E' = (K, U, M, T, W', R', D', A', H', C', P', N')$ *be accepted* $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ *packets. Suppose the two packets agree on the source-window row* K*, the RealUniformStructure row* U*, the shared modulus row* M*, and every StreamName, RegSeqRat, and DyadicRatCore row beyond a displayed Cauchy threshold* T*. Then every completion-facing consumer reads the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route through*

$$K \longmapsto U \longmapsto M \longmapsto T \longmapsto W \longmapsto R \longmapsto D \longmapsto A \longmapsto H, C, P, N$$

and

$$K \longmapsto U \longmapsto M \longmapsto T \longmapsto W' \longmapsto R' \longmapsto D' \longmapsto A' \longmapsto H', C', P', N'.$$

No selected limit family or quotient-filter equality is exported.

Proof. Project both accepted carriers using Definition 8656.1. The shared rows K, U, M, T force the two packets to present the same source window, entourage row, modulus row, and threshold before any tail-sensitive data is read.

Now compare the tail rows beyond T. The displayed hypothesis says the StreamName, RegSeqRat, and DyadicRatCore rows agree on the part of the packet available to the consumer, so the completion route has the same finite readback pattern in both packets.

Finally replay the structural rows. Componentwise hsame transport and displayed Cont replay preserve the same terminal Real seal route, so the completion-facing consumer sees identical tail-stable output without exporting a selected limit family or quotient-filter equality. □

Theorem 8656.5 (Equicontinuous Cauchy family obligation package)**.** *For every accepted* EquicontinuousCauchyFa *carrier*

$$E = (K, U, M, T, W, R, D, A, H, C, P, N),$$

the rows $K, U, M, T, W, R, D, A, H, C, P, N$ assemble an obligation-closure package consisting exactly of carrier admission, uniform-source exactness, shared-modulus exactness, Cauchy-threshold exactness, L10 readback discipline, and structural stability. The package admits no additional coordinate for selected limits, quotient-filter equality, countable choice, host equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Project the displayed carrier of Definition 8656.1. Its rows expose the source-window coordinate K, the Real-uniform coordinate U, the shared modulus M, the Cauchy threshold row T, the StreamName window row W, the RegSeqRat readback row R, the DyadicRatCore tolerance row D, and the terminal Real seal A.

Apply Definition 8656.2. The carrier admission, uniform-source exactness, shared-modulus exactness, Cauchy-threshold exactness, L10 readback discipline, and structural stability clauses are precisely the listed finite obligation rows.

Apply Definition 8656.3 and Definition 8656.4. These results force every completion-facing read and every tail-stable comparison to pass through the same finite route, with H, C, P, N supplying only transport, replay, provenance, and naming.

Because the carrier has no row for a selected limit, quotient filter, countable-choice schedule, host equality, open-cover compactness, ambient Real completeness theorem, or bridge credential, the obligation package is exhausted by the displayed rows. □

Theorem 8656.6 (Equicontinuous Cauchy family Real-seal tail nonescape)**.** *Let*

$$E = (K, U, M, T, W, R, D, A, H, C, P, N)$$

be an accepted $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ *packet. No tail-stable completion-facing read can expose the terminal* $\mathsf{Real}^{\uparrow}$ *seal A before the rows K, U, M, T, W, R, D have been displayed, and no such read can export selected limits, quotient-filter equality, countable choice, host equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Start with the carrier projection in Definition 8656.1. The seal row A occurs after the source-window, Real-uniform, shared-modulus, Cauchy-threshold, StreamName, RegSeqRat, and DyadicRatCore rows in the displayed route.

Apply Definition 8656.2. Its L10 readback clause requires W, R, D to mediate the rational observations before any terminal Real seal is public.

Apply Definition 8656.3. A completion-facing consumer receives A only through the finite route $K \to U \to M \to T \to W \to R \to D \to A$.

Finally apply Definition 8656.4. Tail agreement can preserve the displayed route, but it adds no coordinate for selected limits, quotient-filter equality, countable choice, host equality, open-cover compactness, ambient Real completeness, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuousCauchyFamily}^{\uparrow}}$.

Constructive story: An $\mathsf{EquicontinuousCauchyFamily}^{\uparrow}$ packet is generated from compact or bounded source windows, a RealUniformStructure entourage row, a shared equicontinuity modulus, Cauchy-family thresholds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, terminal Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuousCauchyFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8656.1, Definition 8656.2, and Definition 8656.3 seed the finite completion-facing equicontinuous Cauchy-family route over the L10 Real spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuousCauchyFamily}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EquicontinuousCauchyFamilyUp.EquicontinuousCauchyFamilyTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close ambient completeness, selected limits, quotient-filter equality, countable choice, host equality, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, Cauchy-threshold exactness, L10 readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.EquicontinuousCauchyFamilyUp.

Chapter 8657

A Concrete Naming Certificate for $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite dyadic covers, the $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$ packet records the finite interval-net compactness route as displayed BEDC data. It sits over the dyadic interval-net carrier of chapter 5966, the dyadic arithmetic core of chapter 291, the StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names finite subdivision-depth coverage and downstream finite-cover handoff only; it does not assert open-cover compactness, selected subcovers, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8657.1 (Dyadic interval-net compactness carrier)**.** *A* $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, J, E, S, R, Q, H, C, P, N)$$

whose rows are read in the following order. The row I is an accepted $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier. The rows D, J, E are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic, finite cell list, and endpoint-incidence data displayed by that net. The row S is the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row, and Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The structural rows H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over the named packets, and the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalNetCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted compactness packets only through $I, D, J, E, S, R, Q, H, C, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, host interval compactness, quotient stream equality, countable choice, trichotomy, or ambient real completeness.

Theorem 8657.2 (Dyadic interval-net compactness obligations)**.** *For every accepted* DyadicIntervalNetCompactness *carrier*

$$K = (I, D, J, E, S, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalNetCompactness}^{\uparrow}}$ *obligation surface is exhausted by the accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *source row, dyadic endpoint arithmetic, finite cell coverage, endpoint incidence, StreamName endpoint windows, RegSeqRat tolerance readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read a finite interval-net compactness conclusion only after those rows are displayed in that order.*

Proof. Project the carrier of Definition 8657.1. The compactness-facing data are the interval-net source I together with its dyadic endpoint, finite cell, and endpoint-incidence rows D, J, E.

Use the obligation surface of Definition 5966.10 and the finite-chain coverage route of Definition 5966.16. They require every finite cover read to inspect the listed cells and endpoint arithmetic before StreamName, RegSeqRat, or Real-facing rows are available.

Finally read S, R, Q, H, C, P, N as the downstream seal and structural rows. Since the carrier has no row for open covers, selected subcovers, quotient equality, choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8657.3 (Dyadic interval-net finite-cover route)**.** *Every compact-interval or uniform-modulus consumer that reads finite-cover compactness from an accepted* $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$ *packet factors through*

$$I, D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N.$$

The read first exposes the $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite cell route, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic and endpoint incidence, then the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and only then the* $\mathsf{Real}^{\uparrow}$ *seal and structural packet rows. The route exports a finite interval-net cover handoff, not open-cover compactness or an ambient completeness principle.*

Proof. Start with the source net row I. By Definition 5966.2, any finite cover request is refined into displayed dyadic cells whose endpoint and incidence data are read through D, J, E.

Apply subdivision-depth coverage from Definition 5966.6. The finite refinement depth keeps each refined cell inside the same endpoint arithmetic, cell list, endpoint incidence, StreamName, RegSeqRat, and Real-seal route.

The compact-interval or uniform-modulus consumer therefore receives only the displayed finite route. Replaying H, C, P, N preserves the same rows, so the finite-cover conclusion cannot be replaced by selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8657.4 (Dyadic interval-net compactness Real seal boundary)**.** *For every accepted* $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$ *carrier* $K = (I, D, J, E, S, R, Q, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* Q *may be read only after the finite interval-net route* I, D, J, E, S, R *is displayed. Once* Q *is reached, the rows* H, C, P, N *preserve the same finite packet and do not create host real equality, open-cover compactness, selected subcovers, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The row Q is downstream of the accepted interval-net source, dyadic endpoint arithmetic, finite cell list, endpoint incidence, StreamName windows, and RegSeqRat tolerance readback.

Apply the Real seal nonescape theorem Definition 5966.13. It prevents a Real-facing consumer from opening Q before the finite dyadic route has been displayed.

Finally replay the compactness packet's structural rows. The componentwise transport, continuation replay, provenance, and local naming rows keep the same finite route visible, so the Real seal remains a boundary for finite-cover handoff rather than a completeness or host-equality endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalNetCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicIntervalNetCompactness}^{\uparrow}$ packet is generated from an accepted dyadic interval-net source, finite subdivision-depth coverage, StreamName windows, RegSeqRat tolerance readback, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNetCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8657.1, Definition 8657.2, Definition 8657.3, and Definition 8657.4 seed the finite interval-net compactness carrier and its finite-cover route over DyadicIntervalNet, DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalNetCompactness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host real equality, quotient stream equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite subdivision-depth coverage, finite-cover consumer routing, Real seal nonescape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DyadicIntervalNetCompactnessUp.

Chapter 8658

A Concrete Naming Certificate for $\mathsf{LawvereTheory}^{\uparrow}$

From Lawvere's finite-product algebraic theories, the $\mathsf{LawvereTheory}^{\uparrow}$ packet records a categorical presentation of finitary algebraic syntax as finite NameCert data. It reads the small category surface of chapter 61, the product-preserving comparison route of chapter 62, the finite-product rows of chapter 34, and the syntactic arity ledger of chapter 197. The packet therefore exposes algebraic operations by their finite-product arities and equations, not by importing a host algebraic theory, quotient syntax, or an external clone object.

Definition 8658.1 (Lawvere theory carrier)**.** *A* $\mathsf{LawvereTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (C, A, O, I, E, H, R, P, N)$$

with the following displayed rows:

1. *C is the category source row, recording the small categorical carrier, identities, composition, and finite-product comparison surface.*
2. *A is the finite-product arity row, recording the distinguished finite powers of the generic object and their projections.*
3. *O is the operation-symbol row, recording each inspected operation as a morphism out of one displayed finite product.*
4. *I is the product-preserving interpretation row, recording the functorial read that carries arities and operations through finite-product preserving maps.*
5. *E is the equation ledger, recording only finite diagrams between displayed operation composites and projection composites.*
6. *H stores componentwise* hsame *transport for the category source, arity row, operation-symbol row, interpretation row, equation ledger, replay, provenance, and local naming row.*
7. *R stores displayed* Cont *replay from categorical composition and product projections to operation interpretation and equation checking.*
8. *P is the* Pkg *provenance over* $\mathsf{LawvereTheory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LawvereTheory}^{\uparrow}}$ *row.*

The accepted carrier has no coordinate for host quotient syntax, a selected term algebra, an external variety of algebras, or a global equality principle beyond the displayed finite equation ledger.

Theorem 8658.2 (Lawvere theory NameCert obligations)**.** *For an accepted packet $L = (C, A, O, I, E, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LawvereTheory}^{\uparrow}}$ *obligations are exhausted by carrier admission for the category source C, finite-product arities A, operation symbols O, product-preserving interpretation row I, equation ledger E, componentwise transport H, displayed replay R, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 8658.1. The categorical source and finite-product arities are exactly the rows C and A, so every operation read must pass through a displayed morphism in O whose domain is one of the finite products recorded by A.

The interpretation row I is read only after the functorial and product-preserving data have been displayed. Equation checks are then finite ledger reads from E: each equation compares two displayed composites built from projections, operations, identities, and composition in C.

Transport, replay, provenance, and naming are componentwise through H, R, P, N. Since the carrier has no host term quotient, selected term algebra, external variety coordinate, or global equality principle, the local NameCert surface cannot export any of those objects. □

Theorem 8658.3 (Lawvere theory finite-product clone route)**.** *Every clone-facing or algebraic-syntax consumer of an accepted* $\mathsf{LawvereTheory}^{\uparrow}$ *packet must first read the category source C, finite-product arities A, operation symbols O, product-preserving interpretation row I, and finite equation ledger E. Only after those rows are displayed may the consumer reuse the packet as a finite-product clone route through H, R, P, N; no read bypasses the displayed products by invoking host syntax, quotient terms, ambient algebras, or a choice of representatives.*

Proof. Start with the finite-product arity row. It identifies each operation by a morphism from a displayed finite power of the generic object, so the consumer has only the arities and projections recorded in A and O.

Next read the interpretation and equation rows. Product-preserving interpretation through I keeps arities tied to the same finite-product structure, while the ledger E verifies only the finite composites that have been displayed.

Finally repack the route through H, R, P, N. Componentwise transport and replay preserve the same categorical rows, so the resulting clone-facing surface is a finite-product route and not an imported syntax quotient, a selected term model, or an ambient algebraic-theory object. □

Falsifiable prediction: Every accepted $\mathsf{LawvereTheory}^{\uparrow}$ algebraic-syntax read passes through a category source, finite-product arity row, operation-symbol row, product-preserving interpretation row, finite equation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any clone-facing surface is visible.

Independence witness: category, functor, prod, firstorder: none is bijective to LawvereTheoryUp because the packet jointly carries finite-product arities, operation morphisms, product-preserving interpretation, equation ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LawvereTheory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LawvereTheory}^{\uparrow}$ packet is generated from a category source, finite-product arities, operation-symbol morphisms, product-preserving interpretation, a finite equation ledger, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawvereTheory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8658.1, Definition 8658.2, and Definition 8658.3 seed the finite-product algebraic-theory packet over CategoryUp, FunctorUp, ProdUp, FirstOrderUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawvereTheory}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host quotient syntax, a selected term algebra, an external variety of algebras, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, product-preserving interpretation, finite equation-ledger replay, transport, provenance, and clone-facing nonescape for BEDC.Derived.LawvereTheoryUp.

Chapter 8659

A Concrete Naming Certificate for $\mathsf{LocatedCantorSet}^{\uparrow}$

Following Bishop and Bridges on located subsets of constructive metric spaces, the $\mathsf{LocatedCantorSet}^{\uparrow}$ packet records the middle-third Cantor set as a finite located-subset certificate over real-name windows rather than as a host closed subset of the line. It sits after the located-apartness boundary of chapter 6450, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the compact-metric carrier of chapter 129. The chapter supplies a concrete L10 neighbour for located-real and compact-metric routes; it does not claim open-cover compactness, a selected ternary expansion, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8659.1 (Located Cantor set carrier)**.** *A* $\mathsf{LocatedCantorSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, S, R, D, A, I, B, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *name whose Cantor-membership distance is being inspected.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which X is read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed rational approximants.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the inspected dyadic scale and the interval-stage endpoints.*
5. *A is the* $\mathsf{LocatedRealApartness}^{\uparrow}$ *apartness row separating X from a displayed removed middle-third interval whenever such a separation is visible.*
6. *I is the finite interval-stage ledger: it lists the retained closed dyadic-rational cells and the removed gap cells at the inspected stage.*
7. *B is the bisection row saying that the next stage is obtained by splitting each retained cell and deleting its middle gap through displayed dyadic comparisons.*
8. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *consumer row, recording that finite retained cells are passed to compact-metric consumers only as a total-boundedness and Cauchy-window surface.*
9. *H records componentwise* hsame *transport for the real name, stream window, regular readback, dyadic ledger, apartness row, interval ledger, bisection row, compact-metric row, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a located-subset request through* $X, S, R, D, A, I, B, M, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCantorSet}^{\uparrow}$, $\mathsf{LocatedRealApartness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for a selected ternary expansion, quotient stream equality, open-cover compactness, countable choice, host subset equality, or ambient real completeness.

Theorem 8659.2 (Located Cantor set NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCantorSet}^{\uparrow}$ *carrier*

$$K = (X, S, R, D, A, I, B, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}$ *surface consists exactly of real-name admission for X, finite stream-window admission for S, regular-rational readback for R, dyadic interval-stage exactness for D and I, located-apartness exposure for A, bisection-stage replay for B, compact-metric handoff for M, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No located Cantor-set obligation is stored outside these rows.

Proof. Project the carrier of Definition 8659.1. The inspected point is the real-name row X, and it is visible only through the finite stream window S and regular-rational readback R.

The dyadic ledger D and interval ledger I supply the finite stage being inspected. The located-apartness row A records separation from a removed gap when the window falls outside a retained cell, while the bisection row B records the next retained-cell refinement.

The compact-metric row M is downstream of these interval data. It receives only retained finite cells and Cauchy-window information, so it cannot turn the packet into open-cover compactness or a selected point theorem.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier lists no other coordinate, the stated rows exhaust the local NameCert obligations. □

Theorem 8659.3 (Located Cantor set bisection route)**.** *Every accepted* $\mathsf{LocatedCantorSet}^{\uparrow}$ *bisection read factors through the finite route*

$$\mathsf{Real}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{LocatedRealApartness}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}$$

At each displayed stage, the consumer first reads the real-name window, then the regular rational approximants, then the dyadic interval ledger, then the located-apartness row for the removed middle gap, and only then the retained-cell row offered to compact-metric consumers. The route exports no selected ternary expansion, quotient real equality, open-cover compactness, countable choice, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the window-readback part of the carrier. The rows X, S, R expose a finite real-name observation, not a completed point of a host subset.

Read the dyadic ledger D and the interval ledger I. These rows display the current retained cells and removed gap cells at the inspected stage.

The located-apartness row A can only certify separation from one displayed removed gap, and the bisection row B can only refine the displayed retained cells. Thus the stage handoff remains a finite interval calculation.

The compact-metric row M consumes the retained-cell data after those rows have been displayed. The structural rows H, C, P, N preserve the same route, so the bisection read cannot export any of the refused ambient or choice-dependent objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCantorSet}^{\uparrow}$ packet is generated from a RealUp name row, a finite StreamName window, RegSeqRat readback, a DyadicRatCore interval-stage ledger, a LocatedRealApartness gap row, a retained-cell bisection row, a CompactMetric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8659.1, Definition 8659.2, and Definition 8659.3 seed the finite located Cantor-set packet over real-name windows, dyadic interval stages, located gap apartness, retained-cell bisection, and compact-metric handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCantorSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected ternary expansions, quotient real equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked real-name admission, finite-window admission, regular-rational readback, dyadic interval-stage exactness, located gap-apartness exposure, bisection replay, compact-metric handoff, transport, replay, provenance, and non-escape for BEDC.Derived.LocatedCantorSetUp.

Chapter 8660

A Concrete Naming Certificate for $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$ packet records the located-interval Lebesgue-number handoff at the point where the $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ surface, finite interval nets, $\mathsf{DyadicRatCore}^{\uparrow}$ comparison budgets, $\mathsf{Real}^{\uparrow}$ endpoint seals, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ finite-subcover radius ledgers are still visible. It sits after chapter 1808, chapter 5966, chapter 291, and chapter 3471, where the finite subcover radius ledger already appears in Definition 3471.8 and Definition 3471.9. The packet gives L10 compact-uniform and finite-subcover consumers a located Lebesgue-number handoff without claiming arbitrary open-cover compactness, selected subcovers, countable choice, quotient interval equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 8660.1 (Located interval Lebesgue-number carrier)**.** *A* $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, F, L, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval row. It exposes the located endpoints and interval-order data used by the cover request.*
2. *F is the finite cover row. It records the displayed finite interval net and cover witnesses attached to the located interval.*
3. *L is the positive Lebesgue radius row. It gives the radius ledger that makes the displayed finite cover effective.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger. It records dyadic slack, mesh refinement, and endpoint comparison budgets for the cover route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is read only after the located interval, finite cover, positive radius, and dyadic comparison rows have been displayed.*
6. *H records componentwise* hsame *transport for the located interval, finite cover, positive radius, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the finite cover request through I, F, L, D, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, D, E, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, a selected subcover, countable choice, quotient interval equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8660.2 (Located interval Lebesgue-number NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$ *carrier $L = (I, F, L, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalLebesgueNumbe}}$ *surface consists exactly of carrier admission, interval exposure, radius compatibility, finite-net transfer, and nonescape:*

1. *carrier admission for the located interval row, finite cover row, positive Lebesgue radius row, dyadic comparison ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval exposure, requiring the cover request to display I and F before any cover witness is read;*
3. *radius compatibility, requiring the positive radius row L and the dyadic comparison row D to control mesh width, endpoint containment, and finite cover slack;*
4. *finite-net transfer, requiring every finite-cover read to pass through the route $I \longmapsto F \longmapsto L \longmapsto D \longmapsto E$ before any uniform-modulus or subcover route is visible;*
5. *nonescape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet.*

The obligation surface may cite chapter 1808, chapter 5966, chapter 291, and chapter 3471; it does not add arbitrary open-cover compactness, selected subcovers, countable choice, quotient interval equality, host real equality, or bridge checking.

Proof. Project the carrier of Definition 8660.1. The visible rows are exactly the located interval, finite cover, positive radius, dyadic comparison, Real seal, transport, replay, provenance, and local naming rows.

The interval exposure is immediate. A finite cover consumer must read I and F before any witness is inspected, so the located interval and the displayed finite interval net are already public before a radius is used.

The radius compatibility is then forced through L and D. The positive Lebesgue radius row and the dyadic comparison ledger control the mesh width, endpoint containment, and finite-cover slack before the terminal seal is opened.

Finally the finite-net transfer and nonescape are carried by the same packet. The route goes through $I \longmapsto F \longmapsto L \longmapsto D \longmapsto E$ and then through H, C, P, N, so the chapter cannot export arbitrary compactness, selected subcovers, countable choice, quotient interval equality, host real equality, ambient completeness, or bridge checking. □

Theorem 8660.3 (Located interval Lebesgue-number finite-cover handoff)**.** *Every finite-cover consumer of an accepted* $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$ *packet reads through the route*

$$I \longmapsto F \longmapsto L \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

The handoff is the located-interval, finite-cover, positive-radius, DyadicRatCore, and RealUp route that later feeds compact-uniform-modulus and finite-subcover consumers, but it does not export a selected subcover, arbitrary open-cover compactness, quotient interval equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier of Definition 8660.1. The consumer first sees the located interval row I and the finite cover row F, so the read is already confined to the displayed interval data.

Apply Definition 8660.2. It forces the positive radius row L and dyadic comparison row D to mediate every cover witness before the Real seal E can be consumed.

Finally replay the structural rows. The transport, replay, provenance, and local naming rows preserve the same finite route, so every finite-cover consumer factors through the displayed handoff before any uniform-modulus or subcover route can be used. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8660.1, Definition 8660.2, and Definition 8660.3 seed the located-interval Lebesgue-number route for finite-cover consumers over located intervals, positive radii, dyadic comparison budgets, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, countable choice, quotient interval equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exposure, radius compatibility, finite-net transfer, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalLebesgueNumberUp.

Constructive story: A $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$ packet is generated from a located interval row, a finite cover row, a positive Lebesgue radius row, a DyadicRatCore comparison ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8661

A Concrete Naming Certificate for $\mathsf{Sobrification}^{\uparrow}$

Following Johnstone and Gierz–Hofmann–Keimel–Lawson–Mislove–Scott, the $\mathsf{Sobrification}^{\uparrow}$ packet records the sober-point completion of a topological carrier as finite NameCert data. It consumes the topology carrier of chapter 88, the Sierpinski classifier surface of chapter 1860, the order-facing route of chapter 5950, the localic point boundary of chapter 5153, and the compact-open comparison surface of chapter 7424. The chapter names sober completion without importing host point-set topology, choice of generic points, or a spectral-space representation theorem.

Definition 8661.1 (Sobrification carrier)**.** *A* $\mathsf{Sobrification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, I, \Sigma, \preceq, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the source* $\mathsf{Topology}^{\uparrow}$ *carrier row whose closed and open predicates are read only through the accepted topology packet.*
2. *I is the irreducible-closed predicate row, recording nonempty closedness and the finite meet-test boundary for irreducibility.*
3. *Σ is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *classifier row that reads opens as characteristic maps into the Sierpinski carrier.*
4. *$\preceq$ is the specialization preorder row induced by the topology surface.*
5. *G is the generic-point witness row for irreducible closed data, stored as local witness data rather than as a global point-selection principle.*
6. *H records componentwise* hsame *transport for the topology, irreducibility, classifier, specialization, witness, provenance, and local naming rows.*
7. *C records displayed* Cont *routes from topology opens through Sierpinski classifiers and specialization reads to the generic-point row.*
8. *P is the* Pkg *provenance over* $\mathsf{Sobrification}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{ScottTopology}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{StoneSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Sobrification}^{\uparrow}}$ *row.*

The packet has no coordinate for choice of all generic points, ambient spectral-space duality, quotient point creation, classical prime-filter selection, or host topological reflection.

Theorem 8661.2 (Sobrification NameCert obligations)**.** *For the carrier of Definition 8661.1, the local* $\mathsf{NameCert}_{\mathsf{Sobrification}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (T, I, \Sigma, \preceq, G, H, C, P, N),$$

irreducible-closed exposure through I*, Sierpinski classifier exposure through* Σ*, specialization-order exposure through* $\preceq$*, generic-point witness locality through* G*, componentwise* hsame *stability through* H*, displayed* Cont *routing through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 8661.1. The topology row T is the only source of open and closed data, so every irreducibility read must pass through the displayed topology carrier.

The classifier row Σ supplies the Sierpinski-facing open classifier, and the specialization row $\preceq$ records the induced preorder. The witness row G is local to the displayed irreducible closed row I and does not select generic points globally.

Finally H, C, P, N transport, route, source, and name the same finite packet. Since the carrier contains no further coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 8661.3 (Sobrification specialization sober-point surface)**.** *Every accepted* $\mathsf{Sobrification}^{\uparrow}$ *read that exposes a sober point factors through the specialization preorder row* $\preceq$*, the Sierpinski classifier row* Σ*, and the generic-point witness row* G *of*

$$Q = (T, I, \Sigma, \preceq, G, H, C, P, N).$$

It cannot replace those rows by a host sober-reflection theorem, a chosen generic-point function on all irreducible closed sets, or a spectral-space representation claim.

Proof. Start with the carrier of Definition 8661.1. A sober-point read has access to topology data only through T, to irreducible closed data only through I, and to open classifiers only through Σ.

The specialization preorder $\preceq$ is then the displayed order surface connecting topology opens to point comparison. The witness row G is read through that same finite route, so it supplies only the local generic point named by the packet.

Apply Definition 8661.2. Transport, continuation replay, provenance, and local naming remain inside H, C, P, N, and the excluded global principles have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Sobrification}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Sobrification}^{\uparrow}$ packet is generated from topology, irreducible-closed, Sierpinski classifier, specialization preorder, generic-point witness, transport, route, provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Sobrification}^{\uparrow}}$, obligation)

Scope closed: Definition 8661.1, Definition 8661.2, and Definition 8661.3 bind the sober-point surface to topology, irreducible-closed, Sierpinski classifier, specialization, witness, transport, route, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Sobrification}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close a checked Lean carrier, global choice of generic points, ambient spectral duality, or bridge checking.

Upgrade path: Formal closure requires a Lean carrier and NameCert obligation theorem for the displayed sobrification packet.

Chapter 8662

A Concrete Naming Certificate for $\mathsf{HardCertificateLayer}^{\uparrow}$

The $\mathsf{HardCertificateLayer}^{\uparrow}$ packet realizes the verification layer of section 9462.5 as finite NameCert data. It records backend-relative proof-object acceptance, trust-boundary exposure, classifier preservation, verification-ledger routing, and local naming without turning a checked theorem into a new classifier. The packet sits beside chapter 526, chapter 546, chapter 630, and chapter 1157 as a certificate-hardening surface rather than as an audit gate, kernel witness, negative boundary, or host-primitive leak detector.

Definition 8662.1 (HardCertificateLayer carrier)**.** *A* $\mathsf{HardCertificateLayer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, \Phi, \Pi, T, C, L, H, R, P, N)$$

with the following displayed rows:

1. *B is the verification backend row, naming the formal language, checker, proof-object format, accepted dependencies, and trust boundary.*
2. *Φ is the formal statement row translated from an already displayed certificate row.*
3. *Π is the proof-object row accepted by B for Φ.*
4. *T is the trust-boundary row, recording checked components, trusted components, unchecked components, and portability limits.*
5. *C is the classifier-preservation row, requiring the source classifier to remain unchanged by the hardening read.*
6. *L is the verification-ledger row that records assumptions, dependencies, proof artifact, backend, uncovered certificate fields, and not-claimed endpoints.*
7. *H records componentwise* hsame *transport for backend, statement, proof object, trust boundary, classifier, ledger, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from the certificate row being hardened through backend acceptance and back to the unchanged classifier.*
9. *P is the* Pkg *provenance over* $\mathsf{HardCertificateLayer}^{\uparrow}$*, the source NameCert row, the verification backend, proof object, trust boundary, classifier-preservation row, ledger row,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HardCertificateLayer}^{\uparrow}}$ *row.*
The packet has no coordinate for classifier creation, backend-independent truth, transfer across proof assistants, replacement of a source NameCert, or discovery credit from verification alone.

Theorem 8662.2 (HardCertificateLayer NameCert obligations)**.** *For the carrier of Definition 8662.1, the local* $\mathsf{NameCert}_{\mathsf{HardCertificateLayer}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (B, \Phi, \Pi, T, C, L, H, R, P, N),$$

backend-relative proof acceptance through B, Φ, Π*, trust-boundary exposure through* T*, classifier preservation through* C*, verification-ledger routing through* L*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* R*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 8662.1. The backend, statement, and proof-object rows are the only hard-certificate evidence rows, so every verification read passes through B, Φ, Π.

The trust-boundary row T and ledger row L record what the proof object does and does not cover. The classifier-preservation row C binds the hardening read to the same classifier surface that existed before the backend accepted the proof object.

Finally H, R, P, N transport, replay, source, and name the same finite packet. Since the carrier omits classifier creation, backend-independent truth, proof-assistant transfer, source-NameCert replacement, and discovery credit, none of those endpoints is a local NameCert obligation. □

Theorem 8662.3 (HardCertificateLayer classifier preservation)**.** *Every accepted* $\mathsf{HardCertificateLayer}^{\uparrow}$ *transition preserves the classifier row it hardens. If a verification-only transition changes the classifier row, then it is not a* $\mathsf{HardCertificateLayer}^{\uparrow}$ *transition.*

Proof. Read the carrier of Definition 8662.1. The classifier row C is preservation data, not an output slot for a new classifier.

Apply Definition 8662.2. Backend acceptance through B, Φ, Π and trust-boundary exposure through T, L harden a displayed certificate row while C keeps the classifier surface fixed.

Thus any transition that changes the classifier leaves the displayed carrier and cannot be cited as a $\mathsf{HardCertificateLayer}^{\uparrow}$ read. □

Falsifiable prediction: If a verification-only transition changes the classifier row, then it is not a $\mathsf{HardCertificateLayer}^{\uparrow}$ transition.

Independence witness: auditgateboundary, kernelauditwitness, negativecertificateboundary, hostprimitiveleakage: none is bijective because $\mathsf{HardCertificateLayer}^{\uparrow}$ stores backend-relative proof-object acceptance, trust-boundary exposure, classifier preservation, verification-ledger routing, transport, replay, provenance, and local NameCert rows in one carrier. AuditGateBoundaryUp records gate admission rather than proof-object acceptance; KernelAuditWitnessUp records kernel witness data rather than classifier-preserving hardening; NegativeCertificateBoundaryUp records negative evidence boundaries rather than backend-relative proof artifacts; HostPrimitiveLeakageUp records primitive leakage rather than the ledgered preservation of a source classifier.

The Lean follow-up should provide Nontrivial $\mathsf{HardCertificateLayer}^{\uparrow}$ and FieldFaithful $\mathsf{HardCertificateLayer}^{\uparrow}$ instances in `lean4/BEDC/Derived/HardCertificateLayerUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{HardCertificateLayer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HardCertificateLayer}^{\uparrow}$ packet is generated from backend, statement, proof-object, trust-boundary, classifier-preservation, verification-ledger, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{HardCertificateLayer}\uparrow}$, obligation)

Scope closed: Definition 8662.1, Definition 8662.2, and Definition 8662.3 bind hard-certificate reads to backend-relative proof objects, trust boundaries, classifier preservation, verification ledger rows, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{HardCertificateLayer}\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean instances, backend-independent truth, classifier creation, source-NameCert replacement, proof-assistant transfer, or bridge checking.

Upgrade path: Formal closure requires the Lean carrier, Nontrivial and FieldFaithful instances, and a checked NameCert obligation theorem for the displayed hard-certificate-layer packet.

Chapter 8663

A Concrete Naming Certificate for $\mathsf{RealCompletionDiagonalRouter}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCompletionDiagonalRouter}^{\uparrow}$ packet records the finite diagonal handoff used by the Real and Completion route of chapter 9238. It sits after the dyadic tolerance ledger of chapter 291, the stream-window carrier of chapter 126, the regular-rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the minimal completion service boundary of chapter 1606. Its role is to route one finite precision window through those rows before a completion-facing consumer reads the $\mathsf{Real}^{\uparrow}$ seal; it is not a quotient-stream construction, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem.

Definition 8663.1 (RealCompletionDiagonalRouter carrier)**.** *A* $\mathsf{RealCompletionDiagonalRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, M, H, C, P, N)$$

with the following visible rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite precision-window row requested by the diagonal consumer.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance and readback row for the same window.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row built from the stream window and dyadic readback.*
4. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by that regular-rational handoff.*
5. M *is the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *service row that records dense-image coverage, universal factorization, separated comparison, and replay non-escape for the completion-facing read.*
6. H *records componentwise* hsame *transport for the window, dyadic, regular-rational, Real seal, minimality, replay, provenance, and local naming rows.*
7. C *records the displayed* Cont *replay through* W, D, R, E, M, H, P, N *in the order in which a diagonal consumer may read the packet.*
8. P *is the* Pkg *provenance over* $\mathsf{RealCompletionDiagonalRouter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalRouter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected diagonal limits, host real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8663.2 (RealCompletionDiagonalRouter NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionDiagonalRouter}^{\uparrow}$ *carrier* $Q = (W, D, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalRoute}}$ *surface consists exactly of carrier admission for the finite StreamName window, dyadic readback, RegSeqRat handoff, Real seal, CauchyCompletionMinimality route, componentwise transport, displayed replay, provenance, and local naming rows.*

Every completion-facing diagonal read must expose the route

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto M \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before it can present the Real seal as a finite completion handoff. The obligation surface exports no quotient stream, selected limit, host equality, countable-choice schedule, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is projection from Definition 8663.1. The tuple displays exactly the finite window, dyadic readback, regular-rational handoff, terminal Real seal, minimal completion service, transport, replay, provenance, and local naming rows.

The diagonal route is forced by the order of the displayed carrier. A consumer first reads the StreamName window W, then the DyadicRatCore tolerance row D, then the RegSeqRat handoff R, and only then reaches the Real seal E.

The completion-facing part is controlled by M. By Definition 1606.13, the minimality row exports dense-image coverage, universal factorization, separated reflection, replay non-escape, and finite-row induction without quotient completion, selected limits, host equality, choice, compactness, or ambient real-completeness data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since none of the carrier rows contains an off-route completion principle, the local NameCert obligations are exhausted by the stated finite diagonal handoff. □

Theorem 8663.3 (RealCompletionDiagonalRouter precision-window route)**.** *Every accepted* $\mathsf{RealCompletionDiago}$ *packet routes a finite precision-window request from* $\mathsf{StreamName}^{\uparrow}$ *through* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff to the* $\mathsf{Real}^{\uparrow}$ *seal before the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *service row can be consumed. The route is finite, componentwise transported by* H*, replayed by* C*, sourced by* P*, and named by* N*.*

The packet does not identify Real names by quotient streams, choose a diagonal limit representative, assert host real equality, call on countable choice, or close ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8663.2. They require any diagonal read to pass through $W, D, R, E, M, H, C, P, N$ in the displayed order.

The StreamName, DyadicRatCore, and RegSeqRat rows provide only a finite precision-window route. The Real seal row is therefore reached as a named finite handoff rather than as a selected representative of a quotient stream.

Apply Definition 1606.13 to the minimality row. It keeps the completion-facing consumer inside dense-image, universal-factorization, separated-reflection, replay, and finite-row obligations, so the packet cannot import host equality, choice, compactness, or ambient completeness after the Real seal is read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalRouter}^{\uparrow}}$ **.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionDiagonalRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8663.1, Definition 8663.2, and Definition 8663.3 seed the finite precision-window route from StreamName and RegSeqRat data through DyadicRatCore readback to the RealUp seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCompletionDiagonalRouter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected diagonal limits, host real equality, countable choice, open-cover compactness, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, precision-window readback, RegSeqRat handoff, Real seal non-escape, CauchyCompletionMinimality consumption, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionDiagonalRouterUp.

Constructive story: A RealCompletionDiagonalRouter$^{\uparrow}$ packet is generated from a StreamName finite window, DyadicRatCore tolerance readback, RegSeqRat handoff, RealUp seal, CauchyCompletionMinimality route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8664

A Concrete Naming Certificate for $\mathsf{CompactUniformDiagonalModulus}^{\uparrow}$

Following the Heine–Cantor compactness-to-uniform-continuity route in chapter 9237, the CompactUniformDiagonal packet records the finite diagonal-modulus surface obtained from compact nets, pointwise moduli, rational lower-bound folds, and metric replay. It sits over the compact source of chapter 60, the continuous pointwise-modulus rows of chapter 59, the metric distance carrier of chapter 57, the rational lower-bound fold of chapter 37, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. The packet exposes a tolerance-only modulus as displayed BEDC data, not as an external function selector.

Definition 8664.1 (CompactUniformDiagonalModulus carrier)**.** *A* $\mathsf{CompactUniformDiagonalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, \mathcal{P}, M, R, B, U, H, C, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row carrying the finite net or finite-cover witness.*
2. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row for the displayed source and target metric carriers.*
3. *$\mathcal{P}$ is the finite* ProbeBundle *spine over which the compact net is read.*
4. *M records the local modulus rows attached to the probes in $\mathcal{P}$.*
5. *R records the positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound candidates read from the local radius rows.*
6. *B is the diagonal lower-bound ledger certifying one rational precision below all displayed candidates.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exposed from the tolerance history and the diagonal ledger.*
8. *H records componentwise* hsame *transport for the compact source, pointwise rows, bundle, candidates, diagonal ledger, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the finite diagonal route.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalModulus}^{\uparrow}}$ *row, with* Pkg *provenance over* CompactUniformDiagonalModu $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.

The carrier has no coordinate for an open-cover choice function, a host minimum over a finite set, a selected point outside the bundle, quotient point equality, an external modulus function, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8664.2 (CompactUniformDiagonalModulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformDiagonalModulus}^{\uparrow}$ *carrier*

$$D = (K, F, \mathcal{P}, M, R, B, U, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalModulus}^{\uparrow}}$ *surface consists exactly of compact source exposure, pointwise-modulus admission, finite* $\mathsf{ProbeBundle}$ *spine admission, local-radius row exposure, positive* $\mathsf{Rat}^{\uparrow}$ *candidate admission, diagonal lower-bound certification,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise transport, displayed replay, and local naming with provenance. Every uniform-modulus read factors through the finite route*

$$\mathsf{Compact}^{\uparrow}, \quad \mathsf{Continuous}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

Proof. Project the carrier in Definition 8664.1. The compact source, continuous pointwise rows, finite bundle, local modulus rows, rational candidates, lower-bound ledger, and uniform-modulus handoff are displayed coordinates.

The finite diagonal route first reads the compact net $\mathcal{P}$, then reads the local modulus rows M at each bundled probe. The positive rational candidates R provide the finite input to the lower-bound ledger B.

The handoff U is exposed only after the lower-bound ledger has been transported and replayed by H and C. Since the carrier contains no open-cover choice, host minimum, off-bundle point selector, quotient point equality, external modulus function, or completeness row, the listed obligations exhaust the certificate surface. □

Theorem 8664.3 (CompactUniformDiagonalModulus uniform handoff)**.** *An accepted* CompactUniformDiagonalMo *packet exposes a* $\mathsf{UniformModulus}^{\uparrow}$ *precision row only after the compact source* K*, pointwise-modulus row* F*, finite bundle* $\mathcal{P}$*, local modulus rows* M*, rational candidates* R*, and diagonal lower-bound ledger* B *have been displayed. Thus the compact-uniform route is a finite diagonal-modulus packet and not a host Heine–Cantor theorem, open-cover selector, quotient equality principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with an accepted carrier $D = (K, F, \mathcal{P}, M, R, B, U, H, C, N)$. Projection gives the compact and pointwise-continuity source rows before any uniform handoff is visible.

The finite bundle $\mathcal{P}$ is the only index surface for the diagonal route. Reading M and R over that bundle supplies the local radii and positive rational candidates inspected by the lower-bound ledger.

The row B certifies one displayed precision below the candidates, and U exposes that precision as a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ row. Transport, replay, provenance, and local naming keep the read inside the displayed packet, so none of the excluded host or completeness objects can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformDiagonalModulus}^{\uparrow}$ packet is generated from a CompactUp finite-net source, a ContinuousUp pointwise-modulus row, a finite ProbeBundle spine, local modulus rows, positive RatUp candidates, a diagonal lower-bound ledger, a UniformModulusUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformDiagonalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8664.1, Definition 8664.2, and Definition 8664.3 seed the finite compact-net diagonal modulus packet used by compact-uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactUniformDiagonalModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover choice, host finite minima, off-bundle point selection, quotient point equality, external modulus functions, ambient completeness of Real$^{\uparrow}$, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked compact-source admission, pointwise-modulus admission, finite bundle coverage, rational lower-bound stability, uniform-modulus handoff, and non-escape for BEDC.Derived.CompactUniformDiagonalModulu

Chapter 8665

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$ packet records cofinal-subsequence invariance for regular Cauchy rational names as finite BEDC data. It sits over the regular rational readback of chapter 301, the finite schedule surface of chapter 126, the cofinal subsequence handoff of chapter 381, and the terminal real-name seal of chapter 38. The packet compares the source name and the cofinal subsequence only through displayed windows, tolerance rows, and replay data before any Real-facing consumer reads the invariant boundary.

Definition 8665.1 (RegularCauchySubsequenceEquivalence carrier)**.** *A* $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, S, T, W, D, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, carrying the original regular Cauchy rational name.*
2. *S is the cofinal subsequence schedule row, carrying the monotone index route read by* $\mathsf{StreamName}^{\uparrow}$*.*
3. *T is the transported subsequence* $\mathsf{RegSeqRat}^{\uparrow}$ *row obtained by replay along S.*
4. *W records the finite source and subsequence windows compared by the packet.*
5. *D is the dyadic tolerance ledger used to compare windows at a requested precision.*
6. *L is the Real-facing invariant handoff row exposed after R, S, T, W, D have been displayed.*
7. *H records componentwise* hsame *transport for all displayed rows.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records the* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by $R, S, T, W, D, L, H, C, P, N$. It has no coordinate for quotient representatives, host stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 8665.2 (RegularCauchySubsequenceEquivalence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$ *carrier*

$$E = (R, S, T, W, D, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *source and subsequence admission, requiring the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* R *and* T *and the cofinal schedule* S*;*
2. *window comparability, requiring* W *to display the finite source and subsequence windows consumed by the comparison;*
3. *tolerance stability, requiring the dyadic ledger* D *to witness the same requested precision for both windows;*
4. *invariant handoff, requiring* L *to be exposed only after the source, schedule, subsequence, window, and tolerance rows are visible;*
5. *non-escape, requiring* H, C, P, N *to transport, replay, source, and name only the displayed packet without quotient representatives, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 8665.1. The source name, cofinal schedule, transported subsequence name, windows, and dyadic tolerance ledger are all displayed coordinates.

The usual constructive subsequence argument is read through those coordinates. Cofinality lets each requested source precision be met by a later subsequence window, and regularity of R supplies the dyadic comparison recorded in D.

The handoff row L is therefore a finite invariant boundary, not a quotient representative. Transport, replay, provenance, and local naming preserve only this route, so the five obligations exhaust the certificate surface. □

Theorem 8665.3 (RegularCauchySubsequenceEquivalence cofinal tail stability)**.** *Let* $E = (R, S, T, W, D, L, H, C,$ *be an accepted* $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$ *carrier. Every Real-facing consumer read of the cofinal subsequence invariant factors through the finite route*

$$R, S, T, W, D, L, H, C, P, N.$$

Thus discarding a finite prefix and replaying along a monotone cofinal schedule preserves the displayed regular Cauchy name boundary, while exporting no quotient representatives, host equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier. Projection gives the source regular name R, the cofinal schedule S, the subsequence readback T, the comparison windows W, and the tolerance ledger D.

For any requested tail precision, cofinality of S places a selected subsequence window beyond the source tail bound. Regularity of R then compares the source window and the replayed subsequence window through the same dyadic tolerance row.

The rows L, H, C, P, N only expose, transport, replay, source, and name that finite comparison. Since the classifier has no coordinate for quotient representatives, host equality, countable choice, or ambient completeness, every consumer read remains on the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}$ packet is generated from a RegSeqRat source row, a cofinal subsequence schedule, a transported RegSeqRat row, finite StreamName windows, a DyadicRatCore tolerance

ledger, a Real-facing invariant handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 8665.1, Definition 8665.2, and Definition 8665.3 seed the finite cofinal-subsequence invariance packet for regular Cauchy rational names.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchySubsequenceEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host stream equality, host real equality, countable choice, ambient completeness of Real$^{\uparrow}$, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, cofinal schedule admission, window comparability, dyadic tolerance stability, invariant handoff, and non-escape for BEDC.Derived.RegularCauchySubsequenceEquivalenceUp.

Chapter 8666

A Concrete Naming Certificate for BishopFilteredRegularCompletion$^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and filter-base completion, the BishopFilteredRegularCompletion$^{\uparrow}$ packet binds the regular Cauchy completion route to a displayed filter-base completion route through one finite window-induction carrier. It sits after the regular Cauchy completion surface of chapter 1639, the located filter-base packet of chapter 6434, the Cauchy filter-base surface of chapter 1915, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet records a finite handoff between regular Cauchy and filter-base completion readings; it does not assert quotient equality of filters or streams, select a limit, invoke countable choice, or claim ambient completeness of Real$^{\uparrow}$.

Definition 8666.1 (Bishop filtered regular completion carrier)**.** *A* BishopFilteredRegularCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, R, W, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the filter-base row, exposing the directed base element and the located refinement relation consumed from* BishopLocatedFilterBase$^{\uparrow}$ *and* CauchyFilterBase$^{\uparrow}$*.*
2. *R is the regular Cauchy readback row, reading the same request through* BishopRegularCauchyCompletion$^{\uparrow}$*,* StreamName$^{\uparrow}$*, and* RegSeqRat$^{\uparrow}$*.*
3. *W is the finite tail-window row, recording the window in which the filter-base and regular Cauchy readings are compared.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording the displayed dyadic radius, slack, and containment comparisons attached to W.*
5. *A is the terminal* Real$^{\uparrow}$ *seal row, visible only after B, R, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter-base, regular readback, window, tolerance, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through B, R, W, D, A, H.*
8. *P is the* Pkg *provenance over* BishopFilteredRegularCompletion$^{\uparrow}$*,* BishopRegularCauchyCompletion$^{\uparrow}$*,* BishopLocatedFilterBase$^{\uparrow}$*,* CauchyFilterBase$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopFilteredRegularCompletion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient filters, quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8666.2 (Bishop filtered regular completion obligations)**.** *For every accepted* BishopFilteredRegularComple *carrier $F = (B, R, W, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopFilteredRegularCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, filter-base exposure through B, regular Cauchy readback through R, finite tail-window synchronization through W, dyadic tolerance exposure through D, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and nonescape excluding every unlisted completion object.*

Proof. Project the carrier of Definition 8666.1. The filter-base side is the row B, while the regular Cauchy side is the row R.

The finite comparison surface is W, D. The window row W fixes the tail segment shared by the two readings, and the dyadic tolerance row D records the finite radius and containment budget used inside that segment.

The terminal Real read is downstream of B, R, W, D. The rows H, C, P, N preserve, replay, source, and name the same packet, so no quotient filter, quotient stream, selected limit, choice schedule, host real equality, or ambient completeness datum can enter the local NameCert surface. □

Theorem 8666.3 (Bishop filtered regular completion filter-window induction)**.** *Every accepted* $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ *completion read is obtained by finite induction on the displayed tail-window row W. The base window reads the filter-base row B and the regular Cauchy row R at the first displayed dyadic tolerance in D. The induction step refines the directed base element and the regular Cauchy tail inside the next displayed window while preserving the same* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The terminal* $\mathsf{Real}^{\uparrow}$ *seal A is exposed only after this finite window induction has synchronized B, R, W, D.*

Proof. Use the obligation theorem to identify the rows available to the induction. The base step reads the first displayed window in W, then inspects the filter-base row B, the regular Cauchy readback row R, and the dyadic tolerance row D for that window.

For the step, refine along the directed filter-base relation recorded in B and along the regular Cauchy tail readback recorded in R. The next window is still a row of W, so the dyadic comparisons remain in D and the route does not leave the finite carrier.

The structural rows H, C, P, N transport, replay, source, and name each induction step componentwise. After the final displayed window has been synchronized, the Real seal row A may be read; before that point no Real-facing completion consumer is exposed. □

Falsifiable prediction: Every accepted $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ read displays a filter-base row, regular Cauchy readback row, finite tail-window row, DyadicRatCoreUp tolerance row, RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any completion-facing consumer can inspect the terminal Real seal.

Independence witness: bishop-regular-cauchy-completion, bishop-located-filter-base, cauchy-filter-base, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds filter-base refinement, regular Cauchy readback, finite tail-window synchronization, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFilteredRegularCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ packet is generated from a filter-base row, regular Cauchy readback row, finite tail-window row, DyadicRatCoreUp tolerance row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFilteredRegularCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8666.1, Definition 8666.2, and Definition 8666.3 seed the finite filter-compatible regular completion packet over filter bases, regular Cauchy readbacks, tail windows, dyadic tolerances, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFilteredRegularCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base exposure, regular Cauchy readback, finite window synchronization, dyadic tolerance stability, Real seal nonescape, replay stability, provenance, and local naming for BEDC.Derived.BishopFilteredRegularCompletionUp.

Chapter 8667

A Concrete Naming Certificate for FiniteVariationMeshCompactness$^{\uparrow}$

Following Jordan's finite-variation ledger and Bishop–Bridges finite-mesh practice, the FiniteVariationMeshCompactn packet records the finite route by which a displayed variation mesh is read as compact-data support for later Real and interval consumers. It sits beside the bounded-variation real-function surface of chapter 3654, the finite variation partition surface of chapter 6386, the dyadic arithmetic core of chapter 291, the finite cover refinement route of chapter 3430, the located interval compactness packet of chapter 3431, the compact interval finite-net route of chapter 6607, and the terminal Real seal of chapter 38. The packet names a finite variation-mesh compactness surface; it does not claim open-cover compactness, ambient Real completeness, or a selected limit.

Definition 8667.1 (Finite variation mesh compactness carrier)**.** *A* FiniteVariationMeshCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, \Pi, D, V, M, R, S, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the* Real$^{\uparrow}$ *interval source row naming the displayed endpoints.*
2. *Π is the displayed finite partition row from the bounded-variation surface.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *budget row recording the finite dyadic comparison and endpoint-arithmetic data.*
4. *V is the* FiniteVariationPartition$^{\uparrow}$ *ledger row read from the same partition data.*
5. *M is the finite mesh row recording the displayed subdivision points and adjacent cell comparisons.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the mesh and partition data.*
7. *S is the* StreamName$^{\uparrow}$ *window row that supplies the finite observation route before any Real-facing consumer reads the packet.*
8. *K is the* CompactInterval$^{\uparrow}$ *handoff row reached only after I, Π, D, V, M, R, S are displayed.*
9. *H records componentwise* hsame *transport for the interval, partition, dyadic budget, variation, mesh, readback, window, compact-handoff, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through the finite variation mesh route into the compact interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{FiniteVariationPartition}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{CompactInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationMeshCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, \Pi, D, V, M, R, S, K, H, C, P, N$. It has no coordinate for a host variation supremum, a selected open-cover subfamily, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8667.2 (Finite variation mesh compactness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ *carrier $V = (I, \Pi, D, V, M, R, S, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationMeshCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval, finite partition, dyadic budget, finite variation ledger, finite mesh, regular-rational readback, StreamName window, compact-interval handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8667.1. The analytic rows are I, Π, D, V, M, R, S, K, and the structural rows are H, C, P, N.

The interval and partition rows are the starting point. The row I displays the interval source, while Π and V read the same finite partition as a bounded-variation ledger rather than as an arbitrary partition family.

The dyadic row D and mesh row M then record the finite comparison data used by the packet. The regular-rational readback R and StreamName window S keep the mesh route finite before the compact-interval handoff K is visible.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no row for open-cover compactness, a selected subcover, quotient equality of functions, countable choice, or ambient completeness, those principles cannot be exported by the NameCert surface. □

Theorem 8667.3 (Finite variation mesh compactness L10 route)**.** *Every completion-facing read of an accepted* $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ *packet factors through*

$$I, \Pi, D, V, M \longmapsto R, S \longmapsto K, H, C, P, N$$

before any compact-interval consumer can read the packet as a compact-mesh route. The read first exposes the interval and displayed partition, then the dyadic budget, finite variation ledger, and mesh, then the regular-rational and StreamName readback rows, and only then the compact-interval handoff and structural packet rows. The route exports no open-cover compactness, selected subcover, host supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the interval source row I and the displayed partition Π. They identify the finite variation surface before any compactness-facing consumer is allowed to inspect the ledger.

Next apply the dyadic row D, variation row V, and mesh row M. These rows keep the route inside the same finite partition data and supply the displayed cell comparisons used by later consumers.

The regular-rational readback R and StreamName window S then expose the finite observation route. The compact-interval handoff K is downstream of that route, so the packet can be read as a compact-facing mesh object only after the displayed variation data have been exposed.

Finally the structural rows H, C, P, N preserve the same displayed packet. Since the carrier contains no supremum, selected subcover, quotient equality, countable-choice, or ambient completeness coordinate, none of those principles can be exported by the route. □

Theorem 8667.4 (Finite variation mesh compactness non-escape)**.** *An accepted* $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple* $I, \Pi, D, V, M, R, S, K, H, C, P, N$*. In particular, it exports no host variation supremum, no selected subcover, no quotient equality of functions, no countable-choice selection, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The NameCert obligations identify every row available to a downstream consumer. The interval, partition, dyadic, variation, mesh, readback, window, and compact-interval rows are the only analytic content rows, and H, C, P, N only transport, replay, package, and name those rows.

Apply Definition 8667.3 to any completion-facing read. It forces the read through the finite variation mesh route before the compact-interval handoff can be consumed.

Since no additional coordinate is present, the packet has no source for host variation suprema, selected subcovers, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: A variation-mesh packet that lacks a displayed finite partition row is not an accepted $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ carrier.

Independence witness: boundedvariationrealfunction, finitevariationpartition, finitecoverrefinement, dyadicratcore, locatedintervalcompactness, compactinterval: these siblings expose partition, ledger, refinement, dyadic, location, or compact-interval rows separately, while $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ binds the finite variation mesh route with all of them as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVariationMeshCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteVariationMeshCompactness}^{\uparrow}$ packet is generated from a RealUp interval row, a displayed finite partition row, a dyadic budget row, a finite variation ledger, a finite mesh row, RegSeqRat readback, StreamName windows, a CompactIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteVariationMeshCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8667.1, Definition 8667.2, Definition 8667.3, and Definition 8667.4 bind the finite variation mesh compactness surface over interval, partition, dyadic, variation, mesh, readback, StreamName, compact-interval handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteVariationMeshCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, ambient RealUp completeness, quotient equality of functions, selected subcovers, countable choice, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, dyadic budget exposure, finite mesh exposure, readback routing, compact-interval handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteVariationMeshCompactnessUp.

Chapter 8668

A Concrete Naming Certificate for MetaCICNormalizationAuditRow$^{\uparrow}$

The MetaCICNormalizationAuditRow$^{\uparrow}$ packet realizes the finite audit-row reading of chapter 9383. It sits beside the normalization audit packet of chapter 3779, the normalization audit surface of chapter 750, the normalization frontier of chapter 892, the normal-endpoint cut of chapter 5143, and the residual-checker normalization budget of chapter 6666. The packet records a finite candidate set, fuel trace, closed typing fragment, residual checker row, and normal endpoint row before a normalization-audit consumer may read the result. It refuses global strong normalization, kernel consistency, quotient term equality, evaluator equality, and subject reduction.

Definition 8668.1 (MetaCIC normalization audit-row carrier)**.** *A* MetaCICNormalizationAuditRow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, F, T, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the candidate-set row, listing the finite candidate entries whose normalization evidence may be inspected.*
2. *F is the fuel-trace row, recording the bounded reduction budget and the inspected trace segment.*
3. *T is the closed typing fragment row, exposing the finite context boundary, term, type, and closedness data used by the audit.*
4. *R is the residual checker row, recording which residual fragments have been checked inside the same bounded window.*
5. *E is the normal endpoint row, available only when the candidate-set, fuel, closed typing, and residual checker rows all support the endpoint read.*
6. *O is the refusal row for requests that ask for global strong normalization, kernel consistency, quotient term equality, evaluator equality, subject reduction, hidden semantic candidates, or an unbounded evaluator.*
7. *H records componentwise* hsame *transport for $S, F, T, R, E, O, C, P, N$.*
8. *C records displayed* Cont *replay from a normalization-audit request through S, F, T, R, E, O, H.*

9. *P is the* Pkg *provenance over* MetaCICNormalizationAuditRow$^{\uparrow}$, MetaCICNormalizationAuditPacket$^{\uparrow}$, MetaCICNormalizationAudit$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICNormalEndpointCut$^{\uparrow}$, MetaCICResidualCh BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, R, E, O, H, C, P, N$ and has no coordinate for an unbounded normalization theorem or kernel-level consistency theorem.

Theorem 8668.2 (MetaCIC normalization audit-row obligations)**.** *For every accepted* MetaCICNormalizationAuditI *carrier $M = (S, F, T, R, E, O, H, C, P, N)$, the local* NameCert$_{\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}}$ *surface consists exactly of carrier admission, candidate-set exposure through S, fuel-trace exposure through F, closed typing fragment exposure through T, residual checker exposure through R, normal endpoint exposure through E, refusal retention through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier. The finite candidate set, bounded fuel trace, closed typing fragment, residual checker, normal endpoint, and refusal rows are displayed as S, F, T, R, E, O.

The endpoint row E is downstream of the finite rows. A consumer must inspect the candidate set, fuel trace, closed typing fragment, and residual checker before reading the endpoint as an audit result.

The refusal row O keeps broader requests outside the carrier. The structural rows H, C, P, N preserve, replay, source, and name the finite audit row without adding global strong normalization, kernel consistency, quotient term equality, evaluator equality, subject reduction, or hidden semantic candidate data. □

Theorem 8668.3 (MetaCIC normalization audit-row finite endpoint route)**.** *Every public endpoint read from an accepted* MetaCICNormalizationAuditRow$^{\uparrow}$ *packet factors through the finite route*

$$S, F, T, R \longmapsto E \longmapsto O, H, C, P, N.$$

The route records candidate-set membership, bounded fuel, closed typing, residual checking, and a normal endpoint inside one finite audit row. It does not export global strong normalization, kernel consistency, quotient term equality, evaluator equality, subject reduction, or an unbounded evaluator.

Proof. Apply the obligation theorem. It orders the endpoint row E after the candidate-set, fuel-trace, closed-typing, and residual-checker rows.

The route is finite because S lists the candidate entries, F bounds the trace, T fixes the inspected closed typing fragment, and R records only residual checks inside that window. The endpoint is therefore an audit-row endpoint, not a kernel-wide normalization theorem.

Finally read O, H, C, P, N. The refusal row blocks the listed global claims, while transport, replay, provenance, and local naming preserve the same displayed route without adding unbounded evaluator or subject-reduction data. □

Falsifiable prediction: Every accepted MetaCICNormalizationAuditRow$^{\uparrow}$ read exposes a candidate set, fuel trace, closed typing fragment, residual checker, normal endpoint row, refusal row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a normalization-audit endpoint is visible.

Independence witness: metacic-normalization-audit-packet, metacicnormalizationaudit, metacicnormalizationfrontier, metacic-normal-endpoint-cut, metacic-residual-checker-normalization-budget:

$\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate-set rows, fuel traces, closed typing fragments, residual checker rows, normal endpoints, refusal retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}$ packet is generated from a candidate-set row, fuel-trace row, closed typing fragment, residual checker row, normal endpoint row, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8668.1, Definition 8668.2, and Definition 8668.3 seed the finite normalization audit row over candidate sets, fuel traces, closed typing fragments, residual checking, normal endpoints, refusal retention, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditRow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, kernel consistency, quotient term equality, evaluator equality, subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-set exposure, fuel-trace exactness, closed typing fragment preservation, residual checker nonescape, normal endpoint routing, refusal retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalizationAuditRowUp.

Chapter 8669

A Concrete Naming Certificate for $\mathsf{BishopFiniteTailComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopFiniteTailComparison}^{\uparrow}$ packet records the finite shared-tail comparison used before equality-facing Real seals are read. It sits between the regular Cauchy source rows of chapter 301, the finite StreamName windows of chapter 126, the dyadic tolerance budgets of chapter 291, the L10 difference route of chapter 1874, and the terminal Real packet of chapter 38. The packet names only the finite comparison surface: two source names, one shared window, one displayed tolerance budget, one verdict row, one terminal seal boundary, transport, replay, provenance, and local naming.

Definition 8669.1 (BishopFiniteTailComparison carrier)**.** *A* $\mathsf{BishopFiniteTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, W, D, Q, E, H, C, P, N)$$

with the following visible rows:

1. *X and Y are* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for two regular Cauchy names sharing the same comparison context.*
2. *W is a shared* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row recording the indices that are simultaneously visible to X and Y.*
3. *D is a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row used to compare the two visible tails inside the window W.*
4. *Q is the finite comparison-verdict row, read only after the source rows, window row, and dyadic budget have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-boundary row that may be inspected only after Q and the* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *route have been read.*
6. *H records componentwise* hsame *transport for the source, window, dyadic-budget, verdict, seal-boundary, replay, provenance, and local naming rows.*
7. *C records the displayed* Cont *replay through $X, Y, W, D, Q, E, H, P, N$ in the order in which a consumer may read the packet.*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFiniteTailComparison}^{\uparrow}}$ *row naming this finite shared-tail comparison packet.*

The carrier has no coordinate for Real completeness, quotient stream equality, selected limits, trichotomy, countable choice, host equality of completed reals, or bridge checking.

Theorem 8669.2 (BishopFiniteTailComparison NameCert obligations)**.** *For an accepted* BishopFiniteTailComparis *carrier $B = (X, Y, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopFiniteTailComparison}^{\uparrow}}$ *surface consists exactly of these obligations:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, finite verdict, terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise transport, continuation replay, provenance, and local naming row;*
2. *shared-window discipline, requiring every comparison read to expose the same finite tail window W for both source rows before the verdict Q is inspected;*
3. *tolerance-budget discipline, requiring the dyadic row D to be displayed before the verdict is consumed;*
4. *difference-route compatibility, requiring any equality-facing read beyond Q to pass through the* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *route before the terminal seal row E is exposed;*
5. *ledger non-escape, stating that H, C, P, N expose no Real completeness theorem, quotient stream equality, selected limit, trichotomy, countable-choice schedule, host equality of completed reals, Lean verification, or bridge check.*

Proof. Carrier admission is projection from Definition 8669.1. The tuple displays exactly the two source rows, shared finite window, dyadic tolerance budget, verdict, seal boundary, transport, replay, provenance, and local naming data.

The shared-window and tolerance-budget obligations are read from W and D before Q. Since the verdict row is downstream of both rows, the comparison cannot be inspected from one source tail alone and cannot use an undisplayed tolerance.

For equality-facing reads, use the placement of E behind Q and the cited $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ route. The structural rows H, C, P, N only transport, replay, source, and name the same finite packet. Hence no excluded Real completeness, quotient equality, selected-limit, trichotomy, choice, host-equality, Lean-verification, or bridge-check principle is exported. □

Theorem 8669.3 (BishopFiniteTailComparison shared-window comparison)**.** *Let $B = (X, Y, W, D, Q, E, H, C, P,$ be an accepted* $\mathsf{BishopFiniteTailComparison}^{\uparrow}$ *carrier. Any finite comparison of the two source rows X, Y must first display the shared* $\mathsf{StreamName}^{\uparrow}$ *window W and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget D, then read the verdict row Q, and only then continue toward the terminal seal boundary E through the* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *route.*

The shared-window route exports no comparison from a one-sided tail, no quotient stream equality, no selected limit, no trichotomy, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 8669.1. The comparison-facing rows are ordered as the two source names X, Y, the shared window W, the dyadic budget D, and the verdict Q.

Apply the NameCert obligations of Definition 8669.2. They force the same window to be displayed for both sources and force the dyadic budget to be available before Q is inspected.

The continuation row C replays the same order through the terminal seal boundary E. Since E is read only after the verdict and the $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ route, the shared-window comparison remains finite and does not export any excluded equality, limit-selection, trichotomy, choice, or Real-completeness principle. □

Theorem 8669.4 (BishopFiniteTailComparison Real-seal boundary)**.** *Let $B = (X, Y, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BishopFiniteTailComparison}^{\uparrow}$ *carrier. The terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary E is readable only after the source rows X, Y, the shared finite window W, the dyadic tolerance budget D, the comparison verdict Q, and the* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *handoff have all been displayed. The boundary is therefore a finite pre-seal route for equality-facing Real reads, not a proof of Real completeness or quotient stream equality.*

Proof. Start with the shared-window comparison theorem Definition 8669.3. It supplies the finite route through X, Y, W, D, Q before any terminal seal-boundary read can occur.

Use the difference-route compatibility clause of Definition 8669.2. It requires equality-facing reads past the verdict to pass through the displayed $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ route before the terminal $\mathsf{Real}^{\uparrow}$ seal boundary E is exposed.

Finally read the ledger non-escape clause. The carrier and structural rows contain no Real completeness theorem, quotient stream equality, selected limit, trichotomy, countable-choice schedule, host equality of completed reals, Lean verification, or bridge check, so the seal boundary remains exactly the stated finite pre-seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFiniteTailComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFiniteTailComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8669.1, Definition 8669.2, Definition 8669.3, and Definition 8669.4 seed the finite shared-tail comparison route from two RegSeqRat source rows through a shared StreamName window, DyadicRatCore budget, finite verdict, CauchyDifferenceCriterion handoff, and terminal Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFiniteTailComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, trichotomy, countable choice, host equality of completed reals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, tolerance-budget discipline, difference-route compatibility, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.BishopFiniteTailComparisonUp.

Constructive story: A $\mathsf{BishopFiniteTailComparison}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one shared StreamName finite window, one DyadicRatCore tolerance budget, one finite comparison verdict, one terminal Real seal-boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8670

A Concrete Naming Certificate for $\mathsf{IntertwiningOperator}^{\uparrow}$

Following Fulton and Harris on intertwiners between finite-dimensional representation spaces, the $\mathsf{IntertwiningOperator}^{\uparrow}$ packet records an equivariant linear map between two displayed finite representation rows as NameCert data. It consumes the finite representation carrier of chapter 3662, the linear-map source of chapter 48, the Schur-facing Hom boundary of chapter 5882, and the direct-sum and tensor ledger of chapter 246. The packet names only the displayed morphism surface; it does not assert irreducibility, Schur scalarity, quotient representation equality, or a host category of representations.

Definition 8670.1 (IntertwiningOperator carrier)**.** *An* $\mathsf{IntertwiningOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, U, V, L, E, S, C, H, T, P, N)$$

with the following displayed rows:

1. *G is the common finite group source row read through the two representation packets.*
2. *U is the source* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *row.*
3. *V is the target* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *row over the same group source.*
4. *L is the consumed* $\mathsf{LinMap}^{\uparrow}$ *row naming the underlying linear map from the source vector-space row of U to the target vector-space row of V.*
5. *E is the equivariance ledger, recording for each displayed group row that applying the target action after L and applying L after the source action replay to the same named row.*
6. *S is the Schur-facing Hom boundary consumed by* $\mathsf{SchurLemma}^{\uparrow}$ *and* $\mathsf{SchurOrthogonality}^{\uparrow}$ *readers.*
7. *C is the* $\mathsf{RepresentationRing}^{\uparrow}$ *composition and operation ledger boundary, exposing only the rows needed by direct-sum, tensor, and composable-morphism consumers.*
8. *H records componentwise* hsame *transport for G, U, V, L, E, S, C.*
9. *T records displayed* Cont *replay for action compatibility, Schur-facing reads, and representation-ring operation reads.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{SchurLemma}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{IntertwiningOperator}^{\uparrow}}$ *row.*

No carrier coordinate stores all host linear maps, a quotient Hom space, a chosen basis, an irreducible decomposition, or a global representation category.

Theorem 8670.2 (IntertwiningOperator NameCert obligations)**.** *For every accepted* $\mathsf{IntertwiningOperator}^{\uparrow}$ *carrier* $I = (G, U, V, L, E, S, C, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntertwiningOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, common-group exposure through* G*, source and target representation exposure through* U, V*, underlying linear-map exposure through* L*, equivariance exactness through* E*, Schur-facing Hom boundary through* S*, representation-ring consumer boundary through* C*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* T*, package provenance through* P*, and local naming through* N*.*

The obligation surface is local to the finite representation rows. It does not claim Schur scalarity, nonzero Hom classification, representation-ring equality, quotient Hom computation, complete reducibility, or a host category of representations.

Proof. Project the carrier of Definition 8670.1. The tuple $G, U, V, L, E, S, C, H, T, P, N$ is the whole public surface, so carrier admission cannot introduce a second Hom object or a private linear-map equality.

Read U and V through the finite representation chapter and L through the linear-map chapter. The equivariance ledger E is then a finite row-by-row replay over the same group source G, comparing target action after L with L after source action only through displayed continuation rows.

Finally route downstream consumers through S and C. Schur-facing readers may inspect only the Hom boundary already named by the packet, and representation-ring readers may inspect only the displayed composition or operation rows. Repacking those rows with H, T, P, N gives the stated local NameCert surface and none of the refused global endpoints. □

Theorem 8670.3 (IntertwiningOperator composition stability)**.** *Let* $I = (G, U, V, L, E, S, C, H, T, P, N)$ *and* $J = (G, V, W, M, F, R, D, K, Q, B, O)$ *be accepted* $\mathsf{IntertwiningOperator}^{\uparrow}$ *packets sharing the middle representation row* V *and the same finite group source* G*. If the representation-ring composition boundary exposes the composable rows of* C *and* D*, then the displayed composite linear row* $M \circ L$ *carries a finite equivariance ledger obtained by replaying* E *and* F *in sequence. The resulting composite remains an* $\mathsf{IntertwiningOperator}^{\uparrow}$ *packet over* U *and* W *after componentwise transport, continuation replay, provenance, and local naming.*

This stability claim is only about the finite displayed composition surface. It does not identify a quotient Hom class, prove associativity for a host category, select bases, or derive Schur scalarity.

Proof. Begin with the source packet I. Its equivariance ledger E says that each displayed group action on U can be moved across the linear-map row L to the matching action on V by displayed continuation replay.

Next use the target packet J. Its ledger F moves the same displayed group action on V across M to the matching action on W.

Compose the two replays along the middle representation row. The representation-ring boundary rows C and D certify that the requested composition is a displayed finite operation row, while the structural rows H, T, P, N and K, Q, B, O transport, replay, source, and name the composite data.

The constructed carrier therefore has the same row shape as Definition 8670.1, with the composite linear row replacing L and the sequential replay replacing E. Since every step stays inside

the displayed rows, the theorem exports only composition stability for finite intertwiners and none of the refused host-level endpoints. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntertwiningOperator}^{\uparrow}}$.

Constructive story: An $\mathsf{IntertwiningOperator}^{\uparrow}$ packet is generated from a common finite group source, source and target FiniteGroupRepresentationUp rows, a LinearMapUp row, a finite equivariance ledger, SchurLemmaUp and RepresentationRingUp consumer boundaries, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntertwiningOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8670.1, Definition 8670.2, and Definition 8670.3 seed the finite equivariant-linear-map carrier, its NameCert obligation surface, and composition stability over displayed representation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntertwiningOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Schur scalarity, irreducibility, quotient Hom computation, host representation categories, chosen bases, Lean verification, or bridge checking for $\mathsf{IntertwiningOperator}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, common-group exposure, source and target representation exposure, linear-map exactness, equivariance replay, Schur-facing boundary, representation-ring composition stability, transport, replay, provenance, and local naming for BEDC.Derived.IntertwiningOperatorUp.

Chapter 8671

A Concrete Naming Certificate for HyperbolicPhaseEnergyTransport$^{\uparrow}$

The HyperbolicPhaseEnergyTransport$^{\uparrow}$ packet realizes the hyperbolic phase transport rows of chapter 9463 as finite phase-energy transport data. It sits between the metric and disk carriers of chapter 3474 and chapter 6209, the reversible phase surface of chapter 6146, the energy ledger of chapter 6481, and the bounded phase-transport route of chapter 7134. The packet names a finite transport-with-energy readback surface; it does not assert physical energy conservation, continuous geodesic dynamics, global curvature integration, or selected observer motion.

Definition 8671.1 (HyperbolicPhaseEnergyTransport carrier)**.** *A* HyperbolicPhaseEnergyTransport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, M, Q, U, L, B, R, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the consumed Poincare disk metric row.*
2. *M is the consumed hyperbolic metric row.*
3. *Q is the observer-phase groupoid transport row naming reversible phase motion between displayed endpoints.*
4. *U is the phase-energy transport row, recording the finite source endpoint, target endpoint, phase label, and energy-budget label.*
5. *L is the consumed* HyperbolicPhaseEnergyLedger$^{\uparrow}$ *row used to read the energy side of the transport.*
6. *B is the consumed* HyperbolicPhaseTransportBudget$^{\uparrow}$ *row bounding the transport before any public read.*
7. *R is the regular-rational readback row for the displayed transport budget.*
8. *S is the* Real$^{\uparrow}$ *seal preventing the budget readback from becoming an unlogged host scalar.*
9. *F is the TasteGate-facing row requiring Lean* `Nontrivial` *and* `FieldFaithful` *instances for the carrier fields.*
10. *H records componentwise* hsame *transport for disk, metric, phase groupoid, energy transport, ledger, budget, readback, seal, TasteGate-facing, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay for phase transport, energy-ledger readback, budget exposure, and Real sealing.*
12. *P is the* Pkg *provenance over* PoincareDiskMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, ObserverPhaseGroupoid$^{\uparrow}$, HyperbolicPhaseEnergyLedger$^{\uparrow}$, HyperbolicPhaseTransportBudget$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* NameCert$_{\text{HyperbolicPhaseEnergyTransport}^{\uparrow}}$ *row.*

No carrier coordinate stores a continuous geodesic flow, physical Hamiltonian system, global conservation law, host curvature integral, or selected observer trajectory.

Theorem 8671.2 (HyperbolicPhaseEnergyTransport NameCert obligations)**.** *For every accepted* HyperbolicPhaseEnergyTransport$^{\uparrow}$ *carrier $E = (D, M, Q, U, L, B, R, S, F, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicPhas}}$ *surface consists exactly of disk-metric admission, hyperbolic-metric admission, reversible phase-groupoid transport, phase-energy transport exposure, energy-ledger readback, phase-transport budget exposure, regular-rational readback, Real-seal nonescape, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The surface is a finite transport-with-energy packet. It does not claim global hyperbolic dynamics, physical conservation, curvature integration, selected observer motion, or a host analytic energy theorem.

Proof. Project the carrier of Definition 8671.1. The disk and metric rows D, M must be visible before the phase groupoid transport Q is read, and the transport row U is only a finite endpoint-and-budget row over that displayed metric surface.

Next expose the energy ledger L and phase-transport budget B. These rows keep the energy readback and transport bound inside the finite packet before the regular-rational row R and Real seal S can be inspected.

Finally use the TasteGate-facing row F and structural rows H, C, P, N. They require field separation, componentwise transport, displayed replay, provenance, and local naming for exactly the displayed rows, so no physical conservation law, global hyperbolic flow, host curvature integral, or selected trajectory is exported. □

Theorem 8671.3 (HyperbolicPhaseEnergyTransport budget nonescape)**.** *Any public read of an accepted* HyperbolicPhaseEnergyTransport$^{\uparrow}$ *packet must pass through the metric rows D, M, the phase-groupoid row Q, the transport row U, the energy ledger L, and the budget row B before it reaches the readback and Real-seal rows R, S. Therefore a read that omits any one of $D, M, Q, U, L, B, H, C, P, N$ is not a* HyperbolicPhaseEnergyTransport$^{\uparrow}$ *read, and the packet cannot be reread as a bare Real scalar or a physical energy law.*

Proof. The carrier definition orders the readback rows after the displayed metric, phase-transport, energy-ledger, and budget rows. Thus R and S are not independent numeric payloads; they are sealed readbacks of the finite route.

Componentwise transport and continuation replay preserve that route under H and C. Package provenance and local naming then repack the same displayed rows as the public certificate.

If a consumer reaches a Real-facing value without one of the named rows, it has left the carrier surface. The theorem therefore records nonescape for the finite budget route rather than any analytic or physical conservation claim. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}$ packet exposes $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$, $\mathsf{HyperbolicPhaseEnergyLedger}^{\uparrow}$, and $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ rows before any $\mathsf{Real}^{\uparrow}$ energy-transport seal; a read that reaches the seal without those rows refutes the carrier surface.

Independence witness: hyperbolic-phase-energy-ledger, hyperbolic-phase-transport-budget, observer-phase-groupoid, hyperbolic-phase-curvature-budget, hyperbolic-phase-ledger: no listed sibling is bijective to $\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}$ because this packet jointly carries disk metric data, hyperbolic metric comparison, reversible phase transport, energy-ledger readback, transport-budget exposure, regular readback, Real sealing, TasteGate-facing field separation, transport, replay, provenance, and local naming.

The Lean TasteGate file `lean4/BEDC/Derived/HyperbolicPhaseEnergyTransportUp/TasteGate.lean` must provide `Nontrivial` and `FieldFaithful` instances for $\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}$. Those instances must preserve $D, M, Q, U, L, B, R, S, F, H, C, P, N$ as distinct fields before any carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}$ packet is generated from PoincareDiskMetricUp disk data, HyperbolicMetricUp metric rows, ObserverPhaseGroupoidUp reversible phase transport, a finite energy-transport row, HyperbolicPhaseEnergyLedgerUp readback, HyperbolicPhaseTransportBudgetUp bounds, RegSeqRatUp readback, a RealUp seal, TasteGate-facing field-separation requirements, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8671.1, Definition 8671.2, and Definition 8671.3 seed the finite phase-energy transport carrier, NameCert surface, and budget nonescape route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical energy conservation, continuous geodesic dynamics, global curvature integration, selected observer motion, Lean verification, or bridge checking for $\mathsf{HyperbolicPhaseEnergyTransport}^{\uparrow}$.

Upgrade path: Obligation closure requires checked disk admission, metric admission, phase-groupoid transport, energy-transport exposure, energy-ledger readback, budget exposure, Real-seal nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPhaseEnergyTransportUp.

Chapter 8672

A Concrete Naming Certificate for CauchyProductModulusMajorant$^{\uparrow}$

Following Hardy, Knopp, and Bishop–Bridges on constructive products of series, the CauchyProductModulusMajoran packet records the finite modulus-majorant route behind Cauchy products. It sits between the product-series surface of chapter 2214, the majorant tail ledger of chapter 1774, the shared uniform-tail route of chapter 1736, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. It also supplies the product-tail control consumed by the Mertens packet of chapter 3584. The packet names a finite L10 route for product tails; it does not assert a host theorem on infinite Cauchy products, select a product sum, identify quotient series representatives, invoke countable choice, or import ambient Real$^{\uparrow}$ completeness.

Definition 8672.1 (Cauchy product modulus majorant carrier)**.** *A* CauchyProductModulusMajorant$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, Q, T, U, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source-series rows. They expose finite term windows, partial-sum windows, and source tail requests already admitted by the neighboring Cauchy-series product route.*
2. *W is the convolution-window row. It lists the inspected product diagonals, the finite rectangular and triangular windows used to compare product partial sums, and the omitted tail cells.*
3. *Q is the product-modulus row. It records the requested output tolerance together with the finite source-tail indices that make every displayed product-window comparison small enough.*
4. *T is the majorant tail ledger. It records the nonnegative majorant schedules and tail bounds that dominate the omitted convolution cells.*
5. *U is the* UniformCauchySeries$^{\uparrow}$ *threshold row reached only after A, B, W, Q, T have all been displayed.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the source windows, product partial sums, and modulus thresholds.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *budget row, recording the dyadic tolerances, product-radius comparisons, and finite addition slack used by Q and T.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the product modulus, majorant ledger, uniform tail threshold, regular readback, and dyadic budget rows have been inspected.*

9. *H records componentwise* hsame *transport for the source series, convolution windows, modulus row, majorant ledger, uniform threshold, readback, dyadic budget, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a product-tail request through*

$$A, B, W \longmapsto Q, T \longmapsto U, R, D \longmapsto E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyProductModulusMajorant}^{\uparrow}$, $\mathsf{CauchySeriesProduct}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductModulusMajorant}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, Q, T, U, R, D, E, H, C, P, N$. It has no coordinate for a host product theorem, selected infinite convolution sum, quotient equality of series, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8672.2 (Cauchy product modulus majorant NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductModulusMajorant}^{\uparrow}$ *carrier*

$$M = (A, B, W, Q, T, U, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductModulusMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission for the two source-series rows, convolution-window exposure, product-modulus exactness, majorant tail control, uniform-tail thresholding, regular rational readback, dyadic budget accounting, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No Cauchy product modulus-majorant obligation is stored outside these rows.

Proof. Project the carrier of Definition 8672.1. The source rows A and B and the convolution row W are the only product-facing data visible before a tail request is answered.

The modulus and majorant obligations are the rows Q and T. The modulus row stores the finite source-tail indices for the requested tolerance, while the majorant row supplies the displayed domination data for omitted convolution cells.

The endpoint-facing rows are downstream. The uniform threshold U, regular readback R, and dyadic budget D must be displayed before the terminal Real seal E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for a host product theorem, selected convolution sum, quotient equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8672.3 (Cauchy product modulus majorant uniform tail route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyProductModulusMajorant}^{\uparrow}$ *packet factors its constructive Cauchy-product tail read through the displayed route*

$$\mathsf{CauchySeriesProduct}^{\uparrow} \longmapsto \mathsf{CauchyMajorantSeries}^{\uparrow} \longmapsto \mathsf{UniformCauchySeries}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \mapsto$$

In carrier notation, the consumer first reads the source rows and convolution window A, B, W, then the product modulus and majorant ledger Q, T, then the uniform threshold, readback, and

dyadic budget rows U, R, D, and only then the Real seal E and structural rows H, C, P, N. The route exports finite tail control for constructive Cauchy products without exporting a host infinite product theorem, selected product endpoint, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier. The Cauchy-series product surface supplies the two source rows and the convolution window, so the product tail is first a displayed finite-window request through A, B, W.

Next read the modulus-majorant pair. The row Q gives the finite source-tail thresholds for the requested product tolerance, and the row T gives the majorant tail ledger that dominates the omitted convolution cells.

The handoff then enters the uniform-tail route. The $\mathsf{UniformCauchySeries}^{\uparrow}$ row U consumes the same finite threshold data, while R and D record the regular rational readback and dyadic tolerance budget needed before any endpoint-facing read is public.

Finally the Real seal E and structural rows H, C, P, N preserve that route. The consumer therefore receives exactly the finite modulus-majorant handoff and cannot replace it by a host product theorem, a selected infinite convolution sum, quotient equality, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductModulusMajorant}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductModulusMajorant}^{\uparrow}$ packet is generated from two source-series rows, a convolution-window row, a product-modulus row, a majorant tail ledger, a UniformCauchySeries threshold row, RegSeqRat readback, a DyadicRatCore budget, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductModulusMajorant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8672.1, Definition 8672.2, and Definition 8672.3 seed the finite product-tail modulus route over CauchySeriesProduct, CauchyMajorantSeries, UniformCauchySeries, RegSeqRat, DyadicRatCore, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductModulusMajorant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Cauchy product theorems, selected infinite convolution sums, quotient equality of series, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convolution-window exactness, product-modulus exactness, majorant tail control, uniform-tail thresholding, regular readback, dyadic budget accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductModulusMajorantUp.

Chapter 8673

A Concrete Naming Certificate for UniformlyLocatedSequence$^{\uparrow}$

Following Bishop and Bridges, uniformly located sequences are represented by displayed finite located-window and modulus data rather than by selected limits or quotient stream representatives. The UniformlyLocatedSequence$^{\uparrow}$ packet sits after the located sequence modulus route of chapter 1801, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. It names the finite uniform locatedness surface used by the L10 Real and Completion route without asserting quotient streams, countable choice, trichotomy, selected limits, or ambient Real$^{\uparrow}$ completeness.

Definition 8673.1 (Uniformly located sequence carrier)**.** *A* UniformlyLocatedSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located-sequence window row, recording the finite index windows and located value reads exposed to the packet.*
2. *M is the shared monotone modulus schedule, sending each displayed tolerance request to a finite window bound reused by all completion-facing consumers.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance scale that records the positive dyadic requests and comparison rows used by M.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row naming the inspected stream prefixes.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance scale.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after L, M, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for located windows, the shared modulus schedule, dyadic tolerances, stream windows, regular rational readbacks, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a uniform locatedness request through L, M, D, S, R, E, H.*
9. *P is the* Pkg *provenance over* UniformlyLocatedSequence$^{\uparrow}$, LocatedSequenceModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSequence}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, selected limits, countable choice over windows, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8673.2 (Uniformly located sequence NameCert obligations)**.** *For every accepted* UniformlyLocatedSequer *carrier $U = (L, M, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSequence}^{\uparrow}}$ *surface consists exactly of located-window admission through L, shared monotone modulus exposure through M, dyadic tolerance accounting through D,* $\mathsf{StreamName}^{\uparrow}$ *finite-window admission through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8673.1. The only rows that expose analytic data are the located windows L, the shared schedule M, the dyadic tolerance scale D, the stream window S, the regular rational readback R, and the terminal seal E.

The monotonicity requirement is read from the displayed schedule M. Reusing a smaller tolerance or a later completion request therefore reuses the same finite schedule row and the same dyadic tolerance scale rather than selecting a new stream representative.

The readback route then passes through S and R before the $\mathsf{Real}^{\uparrow}$ seal E is available. Rows H, C, P, N transport, replay, source, and name that same route, so a consumer cannot bypass the displayed finite packet.

Since no carrier coordinate records quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness, row projection and monotone reuse of the displayed modulus schedule exhaust the NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{UniformlyLocatedSequence}^{\uparrow}$ read exposes located sequence windows, a shared monotone modulus schedule, a DyadicRatCore tolerance scale, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: located-sequence-modulus, streamname, regseqrat, dyadicratcore, real: $\mathsf{UniformlyLocatedSequence}^{\uparrow}$ is not identical to any listed sibling because it binds one shared monotone locatedness schedule across all finite sequence windows before regular rational readback and Real sealing.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyLocatedSequence}^{\uparrow}$ packet is generated from located sequence windows, a shared monotone modulus schedule, DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyLocatedSequence}^{\uparrow}}$, seed)

Scope closed: Definition 8673.1 and Definition 8673.2 seed the finite uniformly located sequence route from displayed windows and a shared modulus schedule to RegSeqRat readback and the terminal RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyLocatedSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window coverage, monotone modulus reuse, dyadic tolerance compatibility, StreamName window admission, RegSeqRat readback, Real seal nonescape, replay, provenance, and local naming for BEDC.Derived.UniformlyLocatedSequenceUp.

Chapter 8674

A Concrete Naming Certificate for $\mathsf{CompactCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CompactCompletionFunctor}^{\uparrow}$ packet records the finite route by which compact metric source data, total-bounded completion data, and L10 regular-name readback are offered as a functorial completion surface. It consumes the finite-net completion route of chapter 1573, the metric-completion interface of chapter 1506, and the compact metric source surface of chapter 129. Its L10 observation rows are read through chapter 126, chapter 301, chapter 291, and chapter 38. The packet names a displayed completion functor route; it does not assert open-cover compactness, quotient-completion equality, selected limits, countable choice, host equality of completed points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8674.1 (Compact completion functor carrier)**.** *A* $\mathsf{CompactCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, T, C, M, L, S, R, D, E, H, A, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, displaying the compact metric carrier, metric classifier, and compact-facing finite observation surface before any completion read is exposed.*
2. *T is the* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *row, recording the finite-net completion route, completion embedding, separated handoff, and total-bounded completion ledger consumed by the functor surface.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row attached to the completed target, displaying the complete metric seal used by completion-facing consumers.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, exposing the metric-completion admissibility route and the displayed completion map from the source side to the completed side.*
5. *L is the* $\mathsf{Completion}^{\uparrow}$ *interface row, retaining only the functorial source-target ledger, identity route, composition route, and completion-seal handoff visible to downstream packets.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from compact source names and completed target names before any regular readback is public.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those finite windows and the dyadic tolerance ledger.*

8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic scales, finite-net radii, comparison ledgers, and precision requests used by T, S, R.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, T, C, M, L, S, R, D have been displayed.*
10. *H records componentwise* hsame *transport for the compact metric source, total-bounded completion route, complete metric seal, metric-completion map, completion interface, stream windows, regular readback, dyadic tolerance ledger, Real seal, replay, provenance, and local naming rows.*
11. *A records displayed* Cont *replay from a compact-source tolerance request through the finite-net completion route, completion map, stream windows, regular readback, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{CompactCompletionFunctor}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBoundedCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCompletionFunctor}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite compact-completion functor packet. It has no coordinate for an open-cover compactness theorem, quotient representatives, a selected limit, countable choice, host equality of completed points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8674.2 (Compact completion functor NameCert obligations)**.** *For every accepted* CompactCompletionFu *carrier $F = (K, T, C, M, L, S, R, D, E, H, A, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactCompletionFunctor}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed compact metric source, total-bounded completion route, complete metric seal, metric-completion map, completion interface, stream windows, regular readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *finite-net completion handoff, requiring every completion-facing tolerance request to read K, T, D before M, L are exposed;*
3. *complete-target seal, requiring every completed metric read to pass through C and then through the terminal* $\mathsf{Real}^{\uparrow}$ *row E;*
4. *functorial replay, requiring identity and composition reads in L to be replayed through the same S, R, D, E, H, A, P, N rows as the underlying completion request;*
5. *ledger non-escape, excluding open-cover compactness, quotient-completion equality, selected representatives, countable choice, host equality of completed points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 8674.1. The projected rows are exactly the compact metric source K, total-bounded completion route T, complete metric seal C, metric-completion map M, completion interface L, finite windows S, regular readback R, dyadic tolerances D, Real seal E, and structural rows H, A, P, N.

The finite-net completion handoff is inherited from chapter 1573 and chapter 1506. A tolerance request first reads compact metric source data and finite-net radii, then reaches the completion-facing map and interface rows.

The complete-target seal is downstream of chapter 128. The L10 observation spine through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ keeps every completed metric read behind finite windows, regular readbacks, dyadic tolerances, and the terminal seal.

Functorial replay is the role of L together with H and A. Identity and composition reads preserve the displayed completion route and cannot replace it by quotient representatives, selected limits, or host equality. The provenance and naming rows P, N name only this finite route, so the excluded principles have no carrier coordinate from which to escape. □

Theorem 8674.3 (Compact completion functor finite-net handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactCompletionFunctor}^{\uparrow}$ *carrier* $F = (K, T, C, M, L, S, R, D, E, H, A, P, N)$ *must first read the compact metric source row, the total-bounded finite-net completion route, the dyadic tolerance ledger, the metric-completion map, the completion interface, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In route notation, every public completion read factors through*

$$K, D, T \longmapsto M, L \longmapsto S, R \longmapsto C, E \longmapsto H, A, P, N.$$

Thus the handoff exports the L10 finite-net completion route and not an open-cover compactness principle, a quotient-completion equality, a selected limit, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $F = (K, T, C, M, L, S, R, D, E, H, A, P, N)$. The source compact metric packet is visible only through K, and the finite-net completion route is visible only through T together with dyadic tolerance data D.

Apply Definition 8674.2. It requires the finite-net route to be read before the metric-completion map M and completion interface L become public.

The L10 readback then passes through S, R, D, E. The finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback carry the displayed precision request, while the $\mathsf{Real}^{\uparrow}$ seal is reached only after the complete metric row C has been displayed.

Finally, H, A, P, N transport, replay, source, and name the same route. Since the carrier contains no open-cover compactness row, quotient representative, selected limit, countable-choice schedule, host equality row, or ambient Real-completeness theorem, the route shown in the statement is the whole public handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCompletionFunctor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactCompletionFunctor}^{\uparrow}$ packet is generated from a compact metric source row, a total-bounded finite-net completion route, a complete metric seal, a metric-completion map, a completion interface, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCompletionFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 8674.1, Definition 8674.2, and Definition 8674.3 seed the finite NameCert surface for compact metric completion functoriality along the L10 completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCompletionFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient-completion equality, selected limits, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net completion handoff, complete-target sealing, functorial replay, transport and replay stability, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.CompactCompletionFunctorUp.

Chapter 8675

A Concrete Naming Certificate for $\mathsf{BanachIndicatrix}^{\uparrow}$

Following Banach's indicatrix theorem for functions of bounded variation, the $\mathsf{BanachIndicatrix}^{\uparrow}$ packet records the finite BEDC surface by which level preimage counts are read from a bounded-variation real function before any area, variation, or Real-facing consumer sees the result. It sits after the bounded-variation real-function carrier of chapter 3654, the regular rational readback of chapter 301, the dyadic tolerance accounting of chapter 291, and the terminal Real seal of chapter 38. The packet is a finite level-count ledger over displayed interval and preimage rows; it does not introduce a host function quotient, a selected measurable representative, countable choice over levels, a measure-theoretic coarea formula, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8675.1 (Banach indicatrix carrier)**.** *A* $\mathsf{BanachIndicatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, L, G, P, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row on a displayed real interval.*
2. *L is a finite list of displayed level rows read through* $\mathsf{RegSeqRat}^{\uparrow}$ *names.*
3. *G is the finite interval ledger that records which displayed subintervals are inspected for each level row.*
4. *P is the finite preimage-count ledger; each entry records the counted crossings or endpoint hits for one displayed level row and one displayed interval window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for function values, level names, and endpoint comparisons.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for sign separation, endpoint equality tests, and count stability under displayed refinements.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, L, G, P, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the bounded-variation source, level rows, interval ledger, preimage-count ledger, readback, tolerance, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a level-count request through bounded variation, finite level rows, finite interval inspection, preimage counting, rational readback, dyadic tolerance, and terminal Real sealing.*

10. *Q is the* Pkg *provenance over* $\mathsf{BanachIndicatrix}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachIndicatrix}^{\uparrow}}$ *row.*

Accepted packets are compared only through these displayed rows.

Theorem 8675.2 (Banach indicatrix NameCert obligations)**.** *For every accepted* $\mathsf{BanachIndicatrix}^{\uparrow}$ *carrier $I = (B, L, G, P, R, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachIndicatrix}^{\uparrow}}$ *surface consists exactly of bounded-variation real-function source admission, finite level-row exposure, finite interval-ledger exposure, finite preimage-count exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 8675.1. The source is the single row B, and every level inspected by the packet is already listed in the finite row L.

The counting surface is finite. The interval ledger G selects the displayed interval windows, and the preimage-count ledger P records only the crossings or endpoint hits supported by those windows.

Readback and tolerance remain downstream. The row R supplies regular rational names for values and levels, D supplies dyadic sign and endpoint tests, and E seals only the Real-facing boundary reached from those rows.

Finally H, C, Q, N transport, replay, source, and locally name the same packet. Since no row stores a host quotient of functions, a measurable representative, a choice family over all levels, a coarea measure, or ambient Real completeness, none of those objects is exported by the local NameCert surface. □

Theorem 8675.3 (Banach indicatrix finite level-count route)**.** *Every variation, area, or Real-facing consumer of an accepted* $\mathsf{BanachIndicatrix}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ L,\ G,\ P,\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto H, C, Q, N.$$

Thus the Banach indicatrix is exposed as finite level rows with finite interval and preimage-count ledgers, not as a host-level counting function over all real levels or a selected measure datum.

Proof. Apply Definition 8675.2. A consumer cannot read an indicatrix count until the bounded-variation real-function source, the displayed level rows, the interval ledger, and the preimage-count ledger have been exposed.

The handoff to Real-facing consumers then passes through rational readback and tolerance accounting. The $\mathsf{RegSeqRat}^{\uparrow}$ row R names function values and levels, the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the finite sign and endpoint tests, and the $\mathsf{Real}^{\uparrow}$ row E seals only the displayed result.

The structural rows H, C, Q, N preserve that finite route for downstream variation and area readers. They add no host function quotient, all-level counting function, selected measure representative, countable choice, or ambient Real-completeness coordinate. □

Falsifiable prediction: A purported $\mathsf{BanachIndicatrix}^{\uparrow}$ consumer that reads a level count while omitting the bounded-variation real-function source row, finite level rows, finite interval ledger, preimage-count ledger, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: bounded-variation-real-function, regseqrat, dyadicratcore, real: $\mathsf{BanachIndicatrix}^{\uparrow}$ is not bijective with any listed sibling because it joins the bounded-variation real-function source,

finite level rows, finite interval inspection, preimage-count ledger, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming as one level-count packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachIndicatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachIndicatrix}^{\uparrow}$ packet is generated from a BoundedVariationRealFunctionUp source row, finite level rows, finite interval inspection rows, finite preimage-count rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachIndicatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8675.1, Definition 8675.2, and Definition 8675.3 seed the finite Banach indicatrix route for bounded-variation real functions and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachIndicatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host counting function over all real levels, a selected measurable representative, measure-theoretic coarea data, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite level-row exposure, interval-ledger exposure, preimage-count exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BanachIndicatrixUp.

Chapter 8676

A Concrete Naming Certificate for $\mathsf{BishopLocatedComparison}^{\uparrow}$

Following Bishop and Bridges, located comparison of real names is read through explicit rational windows and dyadic tolerance data rather than through trichotomy or quotient-real equality. The $\mathsf{BishopLocatedComparison}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval route of chapter 394. It names a finite comparison interface for L10 real and completion consumers without importing selected representatives, ambient real order, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8676.1 (Bishop located-comparison carrier)**.** *A* $\mathsf{BishopLocatedComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, D, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left regular real-name row, including the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used by the comparison.*
2. *R is the right regular real-name row, with the same kind of displayed stream windows and regular rational readbacks.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and enclosure ledger, recording the positive dyadic requests, enclosure widths, and finite comparison rows used to separate or overlap the two displayed names.*
4. *W is the located-window comparison ledger, listing the finite pairs of left and right windows inspected at each dyadic tolerance.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal surface for the left and right names after the displayed windows and readbacks have been fixed.*
6. *E records the comparison outcome rows available to downstream consumers: separated-left, separated-right, or retained-overlap data, each tied to a displayed tolerance and window pair.*
7. *H records componentwise* hsame *transport for the left and right real-name rows, dyadic tolerance ledger, window ledger, seal rows, comparison outcome rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each comparison request through L, R, D, W, S, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedComparison}^{\uparrow}}$ *row.*

The carrier predicate says that these rows form one finite located-comparison packet. It has no coordinate for real trichotomy, quotient real equality, a selected representative stream, a host order oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8676.2 (Bishop located-comparison NameCert obligations)**.** *For every accepted* BishopLocatedComparisc *carrier $K = (L, R, D, W, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedComparison}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the finite packet of Definition 8676.1;*
2. *left and right source visibility through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks in L and R;*
3. *dyadic located-window comparison coverage through the tolerance and enclosure ledger D together with the paired window ledger W;*
4. *comparison stability under componentwise* hsame *transport and displayed* Cont *replay through H and C;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape through S, requiring every real-facing consumer to pass through the displayed left and right window rows before reading a seal;*
6. *provenance and local naming through P and N.*

No further order, equality, choice, or completeness obligation is part of the packet.

Lean4 ✓ `BEDC.Derived.BishopLocatedComparisonUp.BishopLocatedComparisonTasteGate single carrier alignment`

Proof. Project the carrier of Definition 8676.1. The left and right sources expose only their displayed stream windows and regular rational readbacks, while the dyadic ledger D supplies the tolerance and enclosure rows used by the comparison.

The paired window ledger W records exactly which left and right windows were inspected at each tolerance. A separated or retained-overlap row in E is therefore read only after the corresponding entries of L, R, D, W have been displayed.

Transport and replay are componentwise. Rows H and C move the same left source, right source, dyadic tolerance, window, seal, and outcome rows together, so a consumer cannot replace the finite comparison route with an external order oracle.

Finally, P and N record provenance and local naming for that same route. Since the carrier has no coordinate for trichotomy, quotient equality, selected representatives, or completeness of $\mathsf{Real}^{\uparrow}$, the listed rows exhaust the NameCert surface. □

Theorem 8676.3 (Bishop located-comparison Real seal nonescape)**.** *Every comparison consumer of an accepted* $\mathsf{BishopLocatedComparison}^{\uparrow}$ *carrier routes through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{LocatedInterval}^{\uparrow}$ *rows before it can read a comparison outcome or a real-facing seal. Such a consumer cannot read real trichotomy, quotient real equality, selected stream representatives, a host order oracle, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the consumer route from the carrier. The source rows L and R expose the stream windows and regular rational readbacks, and the dyadic ledger D supplies the tolerance scale on which the comparison is staged.

The window ledger W ties those rows to the located interval boundary inherited from chapter 394. A comparison outcome in E is therefore attached to a finite pair of displayed windows and a displayed dyadic tolerance before the terminal seal S is available.

Rows H, C, P, N transport, replay, source, and name the same finite route. None of these rows contains trichotomy, quotient equality, a selected representative, an external order oracle, or completeness data for $\mathsf{Real}^{\uparrow}$. Hence every real-seal or comparison read remains inside the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedComparison}^{\uparrow}$ read exposes left and right StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance data, located interval window coverage, RealUp seal rows, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedinterval: $\mathsf{BishopLocatedComparison}^{\uparrow}$ is not identical to any listed sibling because it binds two real-name routes to one located comparison ledger before any outcome or RealUp seal is consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedComparison}^{\uparrow}$ packet is generated from left and right regular real-name rows, DyadicRatCore tolerance and enclosure rows, StreamName windows, RegSeqRat readbacks, terminal RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8676.1, Definition 8676.2, and Definition 8676.3 seed the finite Bishop located-comparison route from displayed dyadic windows to terminal real-name seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient real equality, selected stream representatives, a host order oracle, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left and right source visibility, dyadic located-window comparison coverage, transport stability, Real seal nonescape, provenance, and local naming for BEDC.Derived.BishopLocatedComparis

Chapter 8677

A Concrete Naming Certificate for ArchimedeanDyadicIntervalMesh$^{\uparrow}$

Following Bishop and Bridges, an Archimedean interval mesh is read as finite dyadic data: a requested scale is met by an explicit depth, listed endpoints, and displayed cell incidence before any Real-facing completion argument can inspect it. The ArchimedeanDyadicIntervalMesh$^{\uparrow}$ packet places this textbook construction between the dyadic arithmetic of chapter 291, the finite-cell surface of chapter 5966, the stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38.

Definition 8677.1 (Archimedean dyadic interval mesh carrier)**.** *An* ArchimedeanDyadicIntervalMesh$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, A, J, I, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *endpoint ledger, listing the left and right dyadic endpoints, radius comparisons, and tolerance arithmetic used by the mesh.*
2. *A is the Archimedean depth witness: for the requested positive dyadic tolerance it displays a natural depth whose dyadic mesh size is below that tolerance.*
3. *J is the finite cell list, including the ordered dyadic subintervals at depth A and their endpoint-incidence rows.*
4. *I is the* DyadicIntervalNet$^{\uparrow}$ *handoff row that exposes J as a finite interval-net surface rather than as an open cover.*
5. *W is the* StreamName$^{\uparrow}$ *window row through which endpoint and cell observations are inspected.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance ledger.*
7. *Q is the terminal* Real$^{\uparrow}$ *seal row reached only after D, A, J, I, W, R have been replayed.*
8. *H stores componentwise* hsame *transport for endpoint, depth, cell, window, readback, seal, and local classifier rows.*
9. *C stores displayed* Cont *replay routes from a mesh request through D, A, J, I, W, R, Q.*

10. *P records* Pkg *provenance over* ArchimedeanDyadicIntervalMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ArchimedeanDyadicIntervalMesh}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, quotient interval equality, selected real representatives, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8677.2 (Archimedean dyadic interval mesh NameCert obligations)**.** *For every accepted* ArchimedeanDyadicIntervalMesh$^{\uparrow}$ *carrier $M = (D, A, J, I, W, R, Q, H, C, P, N)$, the local* NameCert$_{\text{ArchimedeanDyadicInte}}$ *surface is exhausted by the finite dyadic endpoint ledger, Archimedean depth witness, finite cell list,* DyadicIntervalNet$^{\uparrow}$ *cell handoff,* StreamName$^{\uparrow}$ *observation window,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Any consumer of the mesh must read those rows in that order before it may use the packet as interval or completion data.*

Proof. Project the carrier of Definition 8677.1. The endpoint arithmetic, depth, listed cells, interval-net handoff, finite windows, readback, and terminal seal are the only analytic rows.

The Archimedean row A is not a completeness claim. It is a displayed finite depth witness whose dyadic mesh size is compared against the requested tolerance inside the ledger D.

The cell row J becomes interval-net data only through I. Thus a consumer receives listed dyadic cells and endpoint incidence from the DyadicIntervalNet$^{\uparrow}$ surface before any regular-name or Real-facing read is exposed.

Finally read the structural rows H, C, P, N. They transport, replay, source, and locally name the same finite packet. Since the carrier contains no row for open-cover compactness, selected subcovers, quotient interval equality, selected representatives, countable choice, trichotomy, or ambient Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8677.3 (Archimedean dyadic interval mesh L10 handoff)**.** *Every L10 completion or finite-Real consumer of an accepted* ArchimedeanDyadicIntervalMesh$^{\uparrow}$ *packet must first expose the* DyadicRatCore$^{\uparrow}$ *endpoint ledger D, the Archimedean depth witness A, the finite cell list J, the* DyadicIntervalNet$^{\uparrow}$ *handoff I, the* StreamName$^{\uparrow}$ *window W, the* RegSeqRat$^{\uparrow}$ *readback R, and the terminal* Real$^{\uparrow}$ *seal Q. The handoff supplies a finite dyadic mesh object for interval and completion arguments; it does not supply open-cover compactness, selected finite subcovers, quotient stream equality, selected limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8677.2. They force every mesh read through the endpoint ledger, depth row, finite cell list, interval-net handoff, finite window, regular readback, Real seal, and structural rows.

Use the sibling DyadicIntervalNet$^{\uparrow}$ route of chapter 5966. The row I exposes the displayed dyadic cells and endpoint incidence supplied by J, so the mesh reaches compact-interval or completion consumers only as finite interval-net data.

Next pass through the StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows. The window row W records the inspected endpoint and cell observations, and R gives the regular rational readback obtained from those observations and the dyadic tolerance ledger.

The terminal seal Q is reached only after that finite route has been replayed. Rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming, so an L10 consumer receives a finite mesh handoff rather than an ambient compactness or completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{ArchimedeanDyadicIntervalMesh}^{\uparrow}$ read exposes a DyadicRatCore endpoint ledger, an Archimedean depth witness, a finite dyadic cell list, a DyadicIntervalNet cell handoff, a StreamName window, a RegSeqRat readback, a terminal RealUp seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: dyadicratcore, dyadic-interval-net, streamname, regseqrat, real: ArchimedeanDyadicInter is not identical to any listed sibling because it binds the Archimedean depth witness and the finite dyadic cell list before the interval-net and Real-facing handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicIntervalMesh}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanDyadicIntervalMesh}^{\uparrow}$ packet is generated from a DyadicRatCore endpoint ledger, an Archimedean depth witness, a finite dyadic cell list, a DyadicIntervalNet handoff, a StreamName window, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicIntervalMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8677.1, Definition 8677.2, and Definition 8677.3 seed the finite Archimedean dyadic mesh route from endpoint arithmetic to Real-facing completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanDyadicIntervalMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient interval equality, selected real representatives, countable choice, trichotomy, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, Archimedean depth exactness, dyadic cell coverage, endpoint incidence, StreamName and RegSeqRat readback routing, Real seal ordering, replay, provenance, and local naming for BEDC.Derived.ArchimedeanDyadicIntervalMeshUp.

Chapter 8678

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ packet records the finite envelope by which located Cauchy-name data is admitted to the Real and Completion route without quotienting names or selecting limits. It sits over the dyadic tolerance core of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the directed-index substrate of chapter 1607, and the metric completion service of chapter 1506. The packet is a located completion envelope for finite NameCert consumers; it is not a quotient completion, a selected limit operation, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, or host equality of completed representatives.

Definition 8678.1 (Bishop located completion envelope carrier)**.** *A* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, R, L, M, I, A, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for rational radii, slack witnesses, and refusal rows.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by every Cauchy-name request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the displayed stream windows.*
4. *L is the located classifier row, recording the finite apartness and enclosure answers used to admit a Cauchy name to the envelope.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-service row exposed only after D, S, R, L have been displayed.*
6. *I is the* $\mathsf{DirectedSet}^{\uparrow}$ *request-index row for net-style or filter-style completion consumers that need a finite common-window witness.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the envelope route.*
8. *T records componentwise* hsame *transport for the dyadic, stream, regular-rational, located, completion, directed-index, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion request through $D, S, R, L, M, I, A, T, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, L, M, I, A, T, C, P, N$. It has no coordinate for quotient completion, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, trichotomy, or host equality of completion representatives.*

Theorem 8678.2 (Bishop located completion envelope NameCert obligations)**.** *For an accepted* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ *carrier $E = (D, S, R, L, M, I, A, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompl}}$ *surface consists exactly of the following five obligation families:*

1. *carrier envelope admission through the displayed packet $D, S, R, L, M, I, A, T, C, P, N$;*
2. *located Cauchy-name classification through the finite dyadic, stream-window, regular-rational, and located rows D, S, R, L;*
3. *componentwise* hsame *transport through T for every displayed row;*
4. *finite completion ledger exposure through M, I, A, C, P, N;*
5. *a no-choice boundary excluding quotient completion, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, countable-choice schedules, trichotomy, and host equality.*

Proof. Carrier admission is the tuple of Definition 8678.1. Every row consumed by a completion-facing reader is one of $D, S, R, L, M, I, A, T, C, P, N$, so the carrier envelope is finite and visible.

The located Cauchy-name classifier is confined to D, S, R, L. The dyadic ledger supplies tolerance requests, the stream row supplies finite observations, the regular rational row supplies readback, and the located row records the finite apartness or enclosure answer.

The completion ledger is then read through M, I, A, C, P, N. The metric-completion row and directed-index row expose only the finite request surface that reaches the terminal Real seal, while replay, provenance, and local naming keep the route inspectable.

Finally the carrier contains no coordinate for quotient completion, selected limits, ambient Real completeness, countable choice, trichotomy, or host equality. Componentwise transport through T preserves the displayed rows only, so none of those excluded principles can be exported by the local NameCert surface. □

Theorem 8678.3 (Located Cauchy-name classifier)**.** *Every completion-facing read of an accepted* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ *packet must pass through the located Cauchy-name route*

$$D \longmapsto S \longmapsto R \longmapsto L$$

before it may inspect the completion row M, directed-index row I, or terminal $\mathsf{Real}^{\uparrow}$ *seal A.*

Proof. The carrier definition places the dyadic tolerance ledger, stream windows, regular rational readback, and located classifier before any completion row. No later coordinate stores a second classifier for the same request.

Apply the NameCert obligation surface of Definition 8678.2. It identifies D, S, R, L as the only rows that admit a located Cauchy-name read. Therefore a consumer that reaches M, I, A has already displayed those rows.

The structural rows T, C, P, N transport and replay the same route. They do not add a hidden selector, quotient representative, or ambient completion proof. □

Theorem 8678.4 (Finite completion ledger)**.** *Let $E = (D, S, R, L, M, I, A, T, C, P, N)$ be an accepted* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ *packet. Every Real or Completion consumer reads the completion service only through the finite ledger*

$$D, S, R, L, M, I, A, T, C, P, N.$$

The ledger may cite $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{DirectedSet}^{\uparrow}$ *rows, but it exports no quotient completion, selected Moore–Smith limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or countable-choice schedule.*

Proof. First expose the located Cauchy-name route of Definition 8678.3. It supplies D, S, R, L before any completion-facing row is read.

Next read the completion and directed-index rows M and I. These rows name the finite completion service and common-window substrate available to the consumer, rather than a quotient of all Cauchy names or a selected limiting point.

Finally the terminal Real seal A is transported and replayed through T, C, P, N. Since the ledger contains only these finite rows, any quotient completion, selected limit, Real completeness theorem, or countable-choice schedule must be supplied by a separate certificate. □

Theorem 8678.5 (No-choice boundary)**.** *An accepted* $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ *packet exports a located completion envelope only through the displayed carrier, classifier, transport, and finite ledger rows. It does not supply quotient completion,* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, trichotomy, countable choice, or host equality of completion representatives.*

Proof. Inspect the carrier rows. The packet records dyadic tolerances, finite stream windows, regular rational readback, a located classifier, a metric-completion row, a directed-index row, a terminal Real seal, componentwise transport, replay, provenance, and local naming.

None of those rows is a quotient of Cauchy names, a selector for representatives, a global completeness theorem for $\mathsf{Real}^{\uparrow}$, a trichotomy oracle, or a countable-choice functional. The Real seal is downstream of the finite located classifier and completion ledger.

Thus every legal consumer remains inside the displayed envelope route. Any stronger completion theorem may cite this packet as input, but it must display its own completion, equality, or limit certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a located Cauchy-name classifier, MetricCompletion service row, DirectedSet request index, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 8678.1, Definition 8678.2, Definition 8678.3, Definition 8678.4, and Definition 8678.5 seed the finite envelope over displayed tolerance, stream-window, regular-rational, located, completion-service, directed-index, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, RealUp completeness, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located Cauchy-name classification, componentwise transport, finite completion ledger exposure, provenance, local naming, and no-choice boundary for BEDC.Derived.BishopLocatedComplet

Chapter 8679

A Concrete Naming Certificate for $\mathsf{EventuallyFilter}^{\uparrow}$

Following Bishop–Bridges tail arguments for sequences and the Bourbaki–Kelley filter vocabulary, the $\mathsf{EventuallyFilter}^{\uparrow}$ packet records the finite evidence that a sequence claim is read only beyond a displayed threshold. It sits between the cofinite-tail surface of chapter 1564, the regular Cauchy tail-filter route of chapter 1605, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite eventual-tail certificate, not a quotient filter, selected limit, countable-choice schedule, host equality of streams, or ambient completeness principle.

Definition 8679.1 (EventuallyFilter carrier)**.** *An* $\mathsf{EventuallyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (F, T, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the cofinite threshold row, naming the displayed index after which the event is inspected.*
2. *T is the finite complement check, listing the bounded prefix ignored by the eventual read.*
3. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row whose accepted base elements are exactly the threshold tails generated by F and checked by T.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row read only on the displayed tail.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the tail windows and their tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, T, B, S, R have been exposed.*
7. *H records componentwise* hsame *transport for the threshold, finite complement, filter-basis, stream-window, readback, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from threshold admission through finite complement checking, filter-basis exposure, tail-window readback, regular rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{EventuallyFilter}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{RegularCauchyTailFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EventuallyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, B, S, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, hidden infinite complements, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8679.2 (EventuallyFilter NameCert obligations)**.** *For every accepted* $\mathsf{EventuallyFilter}^{\uparrow}$ *carrier* $V = (F, T, B, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EventuallyFilter}^{\uparrow}}$ *surface consists exactly of cofinite threshold admission, finite complement checking, filter-basis exposure, tail* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside these rows.*

Proof. Project the accepted carrier. The rows F, T, B, S, R, E are the only mathematical data coordinates available to an eventual-tail consumer.

The threshold row F determines the tail, and T records the bounded prefix that is not read by the eventual claim. The filter-basis row B then exposes exactly the cofinite threshold tails generated by those two rows.

The stream and regular-rational rows are downstream of the same basis. The window row S is read only on the displayed tail, and R records the corresponding regular rational readback before the seal E is exposed.

Finally the structural rows H, C, P, N transport, replay, source, and name this route. Since the carrier contains no quotient filter, selected-limit row, countable-choice schedule, host stream equality, hidden infinite-complement row, or ambient completeness principle, none can become a local obligation. □

Theorem 8679.3 (EventuallyFilter tail refinement)**.** *Let* $V = (F, T, B, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{EventuallyFilter}^{\uparrow}$ *carrier. If a displayed threshold tail is refined by deleting a further finite prefix, every accepted read remains in the same carrier route through the refined* F, T, B *rows and the downstream* S, R, E, H, C, P, N *rows. Thus finite prefix deletion changes only the inspected tail window and not the eventual-filter basis seen by completion-facing consumers.*

Proof. Start with the refinement of the threshold row. Deleting a further finite prefix changes the displayed threshold and enlarges the finite complement recorded by T.

The filter-basis row B is built from threshold tails, so the refined tail is a displayed base element of the same eventual-filter route. No quotient-filter comparison is needed; the refinement is recorded by the finite prefix ledger.

Read the downstream rows through the refined tail. The $\mathsf{StreamName}^{\uparrow}$ window S and $\mathsf{RegSeqRat}^{\uparrow}$ readback R are replayed by C after the refined F, T, B rows are exposed, while H transports the same componentwise route.

The terminal seal E therefore remains attached to the same eventual-tail certificate. Only the finite window and prefix ledger have changed, and no selected limit, countable choice, host stream equality, or ambient completeness row is introduced. □

Theorem 8679.4 (EventuallyFilter Real-facing route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{EventuallyFilter}^{\uparrow}$ *packet reaches the seal row only through*

$$F, T, B \longrightarrow S \longrightarrow R \longrightarrow E.$$

The route factors the cofinite threshold, finite complement check, filter-basis row, StreamName windows, and RegSeqRat readback before any Real-facing completion or Cauchy-tail consumer can inspect the terminal seal.

Proof. Begin with a Real-facing read of the accepted packet. The carrier places the seal row E downstream of the stream-window and regular-rational readback rows.

Apply the NameCert obligations. They require the threshold F, finite complement check T, and filter-basis row B to be displayed before S can be read as a tail window.

The readback R is then obtained from that same tail window, and the continuation row replays the order from the threshold data to the Real seal. The route is exactly the displayed finite chain through F, T, B, S, R, E.

The sibling $\mathsf{FrechetFilter}^{\uparrow}$ and $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ chapters may consume this cofinite-tail route, but they receive no quotient filter, selected limit, countable-choice schedule, host stream equality, or ambient completeness coordinate from the packet. □

Falsifiable prediction: Every accepted $\mathsf{EventuallyFilter}^{\uparrow}$ Real-facing read factors through cofinite threshold data, finite complement checks, a FilterBase row, StreamName tail windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: frechetfilter, regular-cauchy-tail-filter, streamname, regseqrat, real: $\mathsf{EventuallyFilter}^{\uparrow}$ is not bijective with any listed sibling because this packet alone binds a plain eventual-tail threshold, finite complement ledger, filter-basis row, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming into one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{EventuallyFilter}^{\uparrow}}$.

Constructive story: An $\mathsf{EventuallyFilter}^{\uparrow}$ packet is generated from a cofinite threshold row, finite complement checks, a FilterBase basis row, StreamName tail windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EventuallyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8679.1, Definition 8679.2, Definition 8679.3, and Definition 8679.4 seed the finite eventual-tail route from cofinite threshold evidence through StreamName and RegSeqRat rows to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EventuallyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality of streams, hidden infinite complements, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked cofinite threshold admission, finite complement checking, filter-basis exposure, tail-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EventuallyFilterU

Chapter 8680

A Concrete Naming Certificate for $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$

Following Heine–Cantor and Bishop's finite-net proof of compact uniform continuity, the CompactModulusExtraction packet records the lattice-shaped finite ProbeBundle data that turns centerwise pointwise moduli into one tolerance-only uniform modulus. It refines the compact-modulus extraction surface of chapter 6750 by making the ProbeBundle fold, rational lower-bound row, and uniform-modulus row visible as separate lattice coordinates. It sits over the compact source rows of chapter 60, the compact metric rows of chapter 129, the continuous-map rows of chapter 130, the finite lower-bound fold of chapter 4096, the metric rows of chapter 57, and the terminal Real and uniform-modulus surfaces of chapter 38 and chapter 369. The packet is finite NameCert data, not an open-cover theorem, selected subcover, external modulus oracle, quotient function equality, or ambient compactness principle.

Definition 8680.1 (CompactModulusExtractionLattice carrier)**.** *A* $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, A, \mathcal{P}, B, Q, U, E, H, C, G, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row naming the inspected compact source, centers, and coverage radius.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used to route a source point through a displayed center.*
3. *A is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus ledger attached to the displayed centers.*
4. *$\mathcal{P}$ is the finite* ProbeBundle *row listing center probes, overlap checks, and admissible tolerance requests.*
5. *B is the ProbeBundle fold row that gathers the centerwise moduli into one finite lattice of positive rational candidates.*
6. *Q is the* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *rational lower-bound row selected from the finite lattice.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row produced from $K, M, A, \mathcal{P}, B, Q$.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for target-distance and tolerance reads consumed by the uniform-modulus route.*

9. *H records componentwise* hsame *transport for compact, metric, pointwise-modulus, ProbeBundle, fold, rational-bound, uniform-modulus, Real-seal, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from compact finite-net admission through pointwise moduli, ProbeBundle folding, rational lower-bound extraction, metric comparison, and uniform-modulus handoff.*
11. *G is the* Pkg *provenance over* $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusExtractionLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, A, \mathcal{P}, B, Q, U, E, H, C, G, N$. It has no coordinate for selected infinite covers, open-cover compactness, host continuity theorems, quotient function equality, arbitrary suprema, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8680.2 (CompactModulusExtractionLattice NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ *carrier $L = (K, M, A, \mathcal{P}, B, Q, U, E, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactModul}}$ *surface consists exactly of compact finite-net admission, metric comparison admission, pointwise-modulus rows, finite ProbeBundle exposure, ProbeBundle fold exactness, rational lower-bound extraction, uniform-modulus handoff, Real-seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The mathematical rows are the compact finite-net data K, metric comparison M, centerwise pointwise-modulus ledger A, ProbeBundle spine $\mathcal{P}$, fold row B, rational lower-bound row Q, uniform-modulus output U, and Real seal E.

The ProbeBundle obligations are finite. Each probe listed by $\mathcal{P}$ is attached to a displayed center and a displayed tolerance request, and the fold row B gathers only those listed candidates.

The rational lower-bound row Q is downstream of the fold. It computes a positive rational bound from the finite lattice before the output row U can be consumed by a uniform-modulus reader.

Finally H, C, G, N transport, replay, source, and name this route. Since no selected infinite cover, open-cover theorem, host continuity proof, quotient function equality, arbitrary supremum, or ambient completeness row is part of the carrier, no local obligation can read one. □

Theorem 8680.3 (CompactModulusExtractionLattice ProbeBundle fold)**.** *Every accepted* CompactModulusExtrac *packet folds pointwise moduli only through the finite route*

$$K, M, A, \mathcal{P} \longrightarrow B \longrightarrow Q.$$

The row B is a fold over the displayed ProbeBundle lattice, and Q is the resulting positive rational lower-bound row consumed before any $\mathsf{UniformModulus}^{\uparrow}$ *handoff is public.*

Proof. Start with the compact and metric rows. They supply a finite list of centers and the comparison route by which nearby source points are checked against those centers.

Read the pointwise-modulus row A only at the probes listed by $\mathcal{P}$. The ProbeBundle spine stores the admissible center, overlap, and tolerance requests, so the fold has a finite domain.

The fold row B gathers those displayed candidates into a finite lattice. The lower-bound row Q then reads the lattice output as a positive rational bound rather than as an external infimum or selected subcover.

Thus any downstream uniform-modulus consumer receives the folded lower bound through the displayed ProbeBundle route and not through open-cover compactness, countable choice, quotient function equality, or ambient completeness. □

Theorem 8680.4 (CompactModulusExtractionLattice uniform-modulus route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ *packet reaches the output row through*

$$K, M, A, \mathcal{P}, B, Q \longrightarrow U \longrightarrow E.$$

The uniform-modulus route is therefore a finite compact-net, ProbeBundle fold, rational lower-bound, metric-comparison, and Real-seal route; it exports no selected open subcover, host continuity theorem, quotient function equality, arbitrary supremum, or ambient Real completeness.

Proof. Begin with a uniform-modulus consumer. The carrier places U downstream of compact finite-net admission, metric comparison, pointwise-modulus rows, ProbeBundle folding, and rational lower-bound extraction.

Apply the ProbeBundle fold theorem. It forces the lower-bound row Q to be read from the displayed finite lattice $K, M, A, \mathcal{P}, B$ before the output row is available.

The Real-seal row E supplies the target-distance and tolerance boundary used by the same route, while H and C preserve the route under componentwise transport and replay. Provenance and local naming record no further analytic coordinate.

Consequently the uniform-modulus output is the displayed finite NameCert route. The excluded open-cover, selected-subcover, host-continuity, quotient-equality, arbitrary-supremum, and ambient-completeness objects are absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ uniform-modulus read factors through compact finite-net rows, Metric comparison rows, ContinuousMap pointwise-modulus rows, a finite ProbeBundle spine, a ProbeBundle fold, a FiniteLowerBoundFold rational row, UniformModulus output, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: compact-modulus-extraction, compact, compactmetric, continuousmap, finite-lower-bound-fold, metric, uniformmodulus, real: $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ is not bijective with any listed sibling because this packet alone exposes the ProbeBundle spine, fold row, rational lower-bound row, and uniform-modulus handoff as one finite lattice certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusExtractionLattice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactModulusExtractionLattice}^{\uparrow}$ packet is generated from compact finite-net rows, metric comparison rows, ContinuousMap pointwise-modulus rows, a finite ProbeBundle spine, a ProbeBundle fold row, a FiniteLowerBoundFold rational lower-bound row, UniformModulus output, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactModulusExtractionLattice}^{\uparrow}}$, seed)

Scope closed: Definition 8680.1, Definition 8680.2, Definition 8680.3, and Definition 8680.4 seed the finite ProbeBundle lattice route from compact finite-net and pointwise-modulus data to a tolerance-only uniform-modulus handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactModulusExtractionLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, external modulus oracles, host continuity theorems, quotient function equality, arbitrary suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, metric comparison, pointwise-modulus exposure, ProbeBundle fold exactness, rational lower-bound extraction, uniform-modulus handoff, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusExtractionLatticeUp.

Chapter 8681

A Concrete Naming Certificate for $\mathsf{MonoidAction}^{\uparrow}$

Following Bourbaki and Mac Lane, the $\mathsf{MonoidAction}^{\uparrow}$ packet records a monoid acting on a displayed carrier as finite BEDC NameCert data. It consumes the monoid source of chapter 40, the acted carrier vocabulary of chapter 3668 or chapter 3600, and the orbit-facing readback routes used by representation-style chapters such as chapter 3662 and chapter 246. The packet names action endpoints, identity and composition law rows, orbit and stabilizer handoff rows, transport, replay, provenance, and local naming; it does not construct a quotient orbit object, a selected representative, a group inverse action, or an ambient category of actions.

Definition 8681.1 (Monoid action carrier)**.** *A* $\mathsf{MonoidAction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, X, \alpha, I, C, H, R, P, N)$$

with the following displayed rows:

1. *M is the consumed* $\mathsf{Monoid}^{\uparrow}$ *source row, exposing the displayed carrier, multiplication, unit, associativity, identity, transport, replay, provenance, and local name rows of the acting monoid.*
2. *X is the acted carrier row, read through* $\mathsf{SetLike}^{\uparrow}$ *or* $\mathsf{TypeLike}^{\uparrow}$ *vocabulary with its local classifier and point rows visible.*
3. *α is the action graph row. For every displayed monoid element row m and acted point row x, it records the endpoint row $\alpha(m, x)$ as a point of X.*
4. *I is the identity-action law row, comparing $\alpha(e, x)$ with x through the identity row of M and the classifier of X.*
5. *C is the action-composition law row, comparing $\alpha(m, \alpha(n, x))$ with $\alpha(mn, x)$ through the multiplication and associativity rows of M.*
6. *H records componentwise* hsame *transport for $M, X, \alpha, I, C, R, P, N$ and for the displayed endpoint comparisons.*
7. *R records displayed* Cont *replay from monoid multiplication and acted carrier reads through action endpoints, identity rows, composition rows, orbit reads, and stabilizer reads.*
8. *P is the* Pkg *provenance over* $\mathsf{MonoidAction}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{TypeLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{MonoidAction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $M, X, \alpha, I, C, H, R, P, N$. It has no coordinate for quotient orbit equality, selected representatives, inverse actions, external function spaces, or an ambient category of actions.

Lean4 (statement) `BEDC.Derived.MonoidActionUp`

Theorem 8681.2 (Monoid action NameCert obligations)**.** *For every accepted* $\mathsf{MonoidAction}^{\uparrow}$ *carrier $A = (M, X, \alpha, I, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonoidAction}^{\uparrow}}$ *surface consists exactly of monoid-source admission through M; acted-carrier admission through X; action graph admission through α; identity-action exactness through I; action-composition exactness through C; componentwise* hsame *transport through H; displayed* Cont *replay through R; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8681.1. The monoid row M and acted carrier row X are the only source rows, so every action endpoint must first display the acting element and acted point over those rows.

The action graph row α supplies the endpoint data. The law rows I and C are then read as finite comparisons over that same graph: I uses the unit row of M, while C uses the multiplication and associativity rows of M.

Finally H, R, P, N transport, replay, source, and name the same finite route. Since the carrier has no row for quotient orbit equality, selected representatives, inverse actions, function spaces, or an ambient action category, the listed rows exhaust the local NameCert surface. □

Theorem 8681.3 (Monoid action orbit-stabilizer handoff)**.** *Every orbit-facing or stabilizer-facing consumer of an accepted* $\mathsf{MonoidAction}^{\uparrow}$ *packet first reads*

$$M, \quad X, \quad \alpha, \quad I, \quad C, \quad H, \quad R, \quad P, \quad N.$$

The orbit read is the finite endpoint family displayed by the action graph row, and the stabilizer read is the finite subroute where those endpoints compare with the inspected point through hsame*. The handoff supplies no quotient orbit object, chosen orbit representative, inverse-action law, host function space, or ambient group-action theorem.*

Proof. Begin with the action graph row α. It is typed only after the monoid source M and acted carrier X have been displayed, so an orbit-facing consumer receives endpoint rows over the same finite source data.

Use the identity row I and composition row C to keep the endpoint family compatible with the monoid unit and multiplication. These rows are law ledgers over the displayed graph; they do not add inverses or quotient orbit data.

For a stabilizer-facing read, restrict the same endpoint family to rows where $\alpha(m, x)$ compares with x through hsame. Transport and replay then pass through H and R, while P and N record provenance and local naming. Thus representation-style and orbit-style consumers receive only the finite action route, not a stronger orbit or stabilizer construction. □

Falsifiable prediction: A $\mathsf{MonoidAction}^{\uparrow}$ read is refuted whenever an orbit-facing, stabilizer-facing, or representation-style consumer uses action endpoints without first displaying the MonoidUp source, acted carrier row, action graph row, identity-action row, action-composition row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MonoidAction}^{\uparrow}}$.

Constructive story: A MonoidAction$^{\uparrow}$ packet is generated from a Monoid source row, an acted SetLike or TypeLike carrier row, an action graph row, identity-action and action-composition law rows, orbit and stabilizer handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MonoidAction}^{\uparrow}}$, seed)

Scope closed: Definition 8681.1, Definition 8681.2, and Definition 8681.3 seed the finite monoid action carrier, its local NameCert obligation surface, and the orbit-stabilizer handoff route.

Formal verification: FormalStatus(NameCert$_{\mathsf{MonoidAction}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.MonoidActionUp`

Bridge status: none

Not claimed: This chapter does not close quotient orbit spaces, selected representatives, inverse actions, group actions, ambient action categories, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action graph admission, identity-action exactness, composition exactness, orbit and stabilizer handoff, transport, replay, provenance, and local naming for BEDC.Derived.MonoidActionUp.

Chapter 8682

A Concrete Naming Certificate for FiniteDyadicSubsequenceSelector$^{\uparrow}$

Following Bishop and Bridges on bounded sequences and finite dyadic interval refinements, the FiniteDyadicSubsequenceSelector$^{\uparrow}$ packet records the finite selector route by which a bounded real sequence exposes inspected dyadic cells and selected finite windows before any Real-and-Completion consumer reads a subsequence handoff. It sits over the dyadic arithmetic core of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the bounded sequence source of chapter 1415, and the finite net surface of chapter 1250. The packet is a choice-free finite selector for bounded-window routes; it is not general sequential compactness, countable choice, quotient stream equality, or ambient Real$^{\uparrow}$ completeness.

Definition 8682.1 (Finite dyadic subsequence selector carrier)**.** *A* FiniteDyadicSubsequenceSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded sequence schedule row, read through the* BoundedRealSequence$^{\uparrow}$ *source and the finite window requests it accepts.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *cell ledger recording the dyadic interval cells inspected inside the displayed bounded range.*
3. *W is the selected finite-window row naming the subsequence windows admitted by the dyadic cell ledger and the bounded sequence schedule.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the selected windows and their dyadic tolerance data.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after B, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, dyadic ledger, selected windows, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from bounded-sequence requests through dyadic cell inspection, selected windows, regular rational readback, and the Real seal.*
8. *P is the* Pkg *provenance over* FiniteDyadicSubsequenceSelector$^{\uparrow}$, BoundedRealSequence$^{\uparrow}$, ClosedBoundedIntervalN DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. It stores no coordinate for a host subsequence function, countable-choice selector, quotient stream equality, selected limit, general sequential compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 8682.2 (Finite dyadic subsequence selector NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}$ *carrier $S = (B, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicSubsequenceSele}}$ *surface consists exactly of carrier admission, bounded-window exposure, dyadic-cell compatibility, selected-window cofinal readback, RegSeqRat and Real handoff, componentwise transport, displayed replay, provenance, and local naming. Each selected window must be read through the bounded sequence schedule B, the dyadic cell ledger D, and the finite window row W before the regular rational row R or the terminal real seal E is public. The obligations export no sequential compactness theorem, countable-choice selector, quotient stream equality, selected host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8682.1. The rows B, D, W, R, E are the only analytic rows available before the structural support rows are read.

The bounded-window exposure starts with B. A downstream consumer receives only the displayed bounded sequence schedule and the finite requests it accepts, so no host subsequence function or countable-choice object is present at the source.

Next read the dyadic cell ledger D and selected window row W. Compatibility means that each selected window lies inside the inspected dyadic cell data and remains tied to the same bounded schedule. Cofinality is therefore a finite displayed window condition, not an infinite selection principle.

Finally read R and E. The regular rational readback is obtained from the same selected windows, and the real seal is downstream of that readback. Transport, replay, provenance, and local naming preserve only the displayed route, leaving no coordinate for quotient stream equality, selected limits, sequential compactness, or ambient completeness. □

Theorem 8682.3 (Finite dyadic subsequence selector bounded-window handoff)**.** *Every bounded-subsequence consumer of an accepted* $\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}$ *packet reads through the finite dyadic route*

$$\mathsf{BoundedRealSequence}^{\uparrow}, \quad \mathsf{ClosedBoundedIntervalNet}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

In carrier notation, the consumer must expose B, D, W, R, E before it treats the selected windows as a Real-and-Completion handoff, while H, C, P, N only transport, replay, source, and name the same packet.

Proof. Begin with the NameCert obligations of Definition 8682.2. They place every selected window downstream of the bounded sequence schedule, dyadic cell ledger, and finite window row.

Read the bounded interval net and dyadic ledger together. The bounded sequence source is usable by a subsequence consumer only through the inspected finite cells and their StreamName windows, so the route cannot jump directly from a bounded sequence to a completed real.

The regular rational and real-facing rows close the handoff. The row R reads back exactly the selected windows, and E is visible only after that readback. Since the support rows H, C, P, N preserve the same finite packet, no consumer can replace the route by general sequential compactness, countable choice, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}$ packet is generated from a bounded sequence schedule, dyadic cell ledger, selected finite windows, RegSeqRat readback, RealUp seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8682.1, Definition 8682.2, and Definition 8682.3 seed the finite dyadic selector route from bounded sequence windows to RegSeqRat and Real handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDyadicSubsequenceSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general sequential compactness, countable choice, quotient stream equality, selected host limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window exposure, dyadic-cell compatibility, selected-window cofinal readback, RegSeqRat and Real handoff, and structural non-escape for the same finite packet.

Chapter 8683

A Concrete Naming Certificate for $\mathsf{KrullDimension}^{\uparrow}$

Following Krull and Bourbaki/Zariski–Samuel, the $\mathsf{KrullDimension}^{\uparrow}$ packet records dimension as finite prime-chain NameCert data rather than as a global set-theoretic supremum. It sits over the ideal carrier of chapter 81, the Noetherian finiteness surface of chapter 3637, the principal-chain boundary of chapter 7991, the valuation-domain comparability route of chapter 6626, and the finite-intersection discipline of chapter 7643. The packet names only displayed chains of prime-ideal rows and their bounded handoff; it does not assert a maximal prime chain, choose a generic point, import excluded ideal equality, or identify schemes by ambient topology.

Definition 8683.1 (Krull dimension carrier)**.** *A* $\mathsf{KrullDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, I, P, L, N, V, J, H, C, G)$$

where R is the displayed commutative-ring source, I is the $\mathsf{Ideal}^{\uparrow}$ *carrier row for finitely named ideals, P is the finite list of prime-ideal names, L is the listed strict-inclusion chain relation on those prime rows, N is the* $\mathsf{NoetherianRing}^{\uparrow}$ *finiteness handoff when such a bound is present, V is the valuation-domain comparability route when the chain is read through a valuation sibling, J is the principal-ideal or Dedekind-style height handoff route, H is componentwise* hsame *transport, C is displayed* Cont *replay, and G is the local* $\mathsf{NameCert}_{\mathsf{KrullDimension}^{\uparrow}}$ *row.*

Theorem 8683.2 (Krull dimension NameCert obligations)**.** *For every accepted* $\mathsf{KrullDimension}^{\uparrow}$ *carrier $K = (R, I, P, L, N, V, J, H, C, G)$, the local* $\mathsf{NameCert}_{\mathsf{KrullDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed ring source, ideal-row admission through chapter 81, primality and strict-inclusion admission for the finite list P, L, optional Noetherian bound routing through chapter 3637, valuation comparability routing through chapter 6626, principal or Dedekind-style height routing through chapter 7991 and chapter 172, componentwise transport, replay, and local naming. No obligation row exports a chosen longest chain, quotient ideal equality, spectrum topology, scheme dimension, or classical maximal-prime principle.*

Proof. Project the carrier of Definition 8683.1. The rows R, I, P, L display the ring source, ideal carrier, prime names, and strict-inclusion chain before any dimension read is public.

Read the sibling finiteness and comparability routes next. The Noetherian row supplies only a finite bound witness when it is present, the valuation-domain row supplies a comparability handoff, and the principal or Dedekind-style route supplies a height-facing handoff through existing concrete algebra chapters.

Finally replay H, C, G. These rows preserve the displayed finite packet and provide the local name, so a consumer receives only the finite prime-chain obligation surface and cannot infer a selected maximal chain, quotient ideal equality, spectrum topology, scheme dimension, or a classical maximal-prime endpoint. □

Theorem 8683.3 (Krull dimension prime-chain handoff)**.** *Every downstream consumer of a* $\mathsf{KrullDimension}^{\uparrow}$ *read that asks for a local, scheme-facing, or valuation-facing dimension boundary must first expose the finite prime-chain rows* P, L*, the* $\mathsf{Ideal}^{\uparrow}$ *source row* I*, and the active sibling route through* $\mathsf{NoetherianRing}^{\uparrow}$*,* $\mathsf{PrincipalIdealDomain}^{\uparrow}$*,* $\mathsf{ValuationDomain}^{\uparrow}$*, or the Dedekind chapter before the dimension boundary is public. The handoff is a finite NameCert route through prime ideals and strict inclusions, not an ambient topological dimension theorem, not a selected irreducible component, and not a quotient construction on ideals.*

Proof. Start with the obligation theorem Definition 8683.2. It makes the ideal row, prime-name list, and strict-inclusion relation visible before any consumer-facing dimension row can be read.

Use the selected sibling route according to the consumer. A Noetherian consumer reads the finite-bound row, a principal-domain consumer reads the principal height handoff, a valuation-domain consumer reads comparability, and a Dedekind-facing consumer reads the height route through the existing Dedekind chapter.

The structural rows H, C, G then transport and replay the same finite packet. Since the carrier has no coordinate for spectrum topology, selected generic points, quotient ideal equality, or a longest-chain choice, the exported handoff is exactly the displayed prime-chain route. □

Closure status for $\mathsf{NameCert}_{\mathsf{KrullDimension}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KrullDimension}^{\uparrow}$ packet is generated from a displayed ring source, IdealUp rows, finite prime-chain rows, optional NoetherianRingUp bounds, PrincipalIdealDomainUp or valuation comparability routes, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KrullDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8683.1 and Definition 8683.2 seed the finite prime-chain NameCert surface for commutative-algebra dimension reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KrullDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected maximal chain, quotient equality of ideals, spectrum topology, scheme dimension, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prime-chain strictness, sibling finiteness routing, valuation comparability, height handoff, transport stability, replay stability, and nonescape for BEDC.Derived.KrullDimensionUp.

Chapter 8684

A Concrete Naming Certificate for $\mathsf{FiveLemma}^{\uparrow}$

Following Mac Lane's homological algebra and the standard abelian-category diagram chase, the $\mathsf{FiveLemma}^{\uparrow}$ packet records the finite route that turns two exact rows, five compatible vertical maps, and four local isomorphism-style witnesses into the middle vertical classifier witness. It sits over the abelian-category carrier of chapter 176 and consumes the same exact-row and boundary discipline used by the snake-lemma packet of chapter 5890. The packet names a bounded chase surface; it does not import element representatives, quotient object equality, ambient diagram extensionality, or an unbounded search oracle.

Definition 8684.1 (Five Lemma carrier)**.** *A* $\mathsf{FiveLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, R_0, R_1, V, L, M, Q, C, T, H, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row exposing the additive kernel-cokernel surface in which both exact rows are read.*
2. *R_0 and R_1 are the upper and lower exact-row ledgers, each carrying five displayed object rows, four displayed horizontal morphism rows, and the local image-kernel containment rows used by the chase.*
3. *V records the five displayed vertical morphisms and the four commutative squares aligning R_0 with R_1.*
4. *L records the two left-side comparison rows used to move witnesses across the first and second vertical maps.*
5. *M records the two middle-neighbour comparison rows used to pass from the second and fourth vertical maps to the third vertical map.*
6. *Q records the two right-side comparison rows used to close the chase through the fourth and fifth vertical maps.*
7. *C is the finite chase ledger. It lists the kernel entry, exact-row transport, square transport, image-kernel handoff, cokernel exit, and middle classifier witness in bounded order.*
8. *T records displayed* Cont *replay from A, R_0, R_1, V, L, M, Q to the chase ledger C.*

9. *H records componentwise* hsame *transport for the abelian-category source, exact rows, vertical squares, comparison rows, chase ledger, replay, provenance, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{FiveLemma}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{SnakeLemma}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiveLemma}^{\uparrow}}$ *row.*

The classifier compares accepted five-lemma packets only through these rows. It contains no coordinate for selected elements outside the displayed histories, hidden kernel representatives, quotient representatives, host-object equality, or a completed ambient diagram category.

Lean4 (definition) `BEDC.Derived.FiveLemmaUp.FiveLemmaUp`

Theorem 8684.2 (Five Lemma NameCert obligations)**.** *For every accepted* $\mathsf{FiveLemma}^{\uparrow}$ *carrier*

$$F = (A, R_0, R_1, V, L, M, Q, C, T, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiveLemma}^{\uparrow}}$ *surface consists exactly of the abelian-category source row A, the upper and lower exact-row ledgers R_0, R_1, vertical-square compatibility through V, left comparison through L, middle-neighbour comparison through M, right comparison through Q, the finite chase ledger C, displayed replay through T, componentwise transport through H, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8684.1. The rows A, R_0, R_1, V expose the abelian-category source, the two exact rows, and the four commutative squares before any middle-map conclusion is available.

The comparison rows L, M, Q then record precisely which neighbouring vertical maps may be used by the chase. These rows are displayed certificate data, not ambient claims about all diagrams in the host category.

Read the chase ledger C next. Its entries are bounded by the five displayed vertical maps and the four displayed squares, so every witness passed through the exact rows is accounted for by a named row of the packet.

Finally T, H, P, N keep the same surface stable under replay, componentwise history equality, provenance, and local naming. Since no hidden representative, host-object equality, or unbounded diagram search appears among the carrier coordinates, the listed rows exhaust the certificate obligations. □

Theorem 8684.3 (Five Lemma finite diagram chase)**.** *Every downstream consumer of an accepted* $\mathsf{FiveLemma}^{\uparrow}$ *packet reads the middle vertical classifier witness only through the displayed route*

$$\mathsf{AbelianCat}^{\uparrow} \longmapsto (R_0, R_1, V) \longmapsto (L, M, Q) \longmapsto C \longmapsto (T, H, P, N).$$

Thus the consumer receives one finite diagram-chase packet of abelian-category source rows, exact-row ledgers, vertical-square compatibility rows, comparison rows, bounded chase entries, replay, transport, provenance, and local naming.

Proof. Begin with the source row A and the two exact rows R_0, R_1. The vertical compatibility row V makes the four squares visible, so the chase can move between the two rows only at named square entries.

Use L, M, Q to expose the neighbouring comparison data. These rows delimit the classical five-lemma assumptions as finite certificate coordinates: left-side transport, middle-neighbour transport, and right-side exit data are all displayed before the middle conclusion is read.

The ledger C then lists the bounded sequence of chase moves. Exactness supplies the image-kernel handoffs, square compatibility transports each handoff across the vertical maps, and the comparison rows provide the permitted entries and exits.

Replaying through T and transporting through H preserves the same finite route. The provenance and naming rows P, N therefore export the middle witness as a NameCert packet rather than as an ambient element chase. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiveLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{FiveLemma}^{\uparrow}$ packet is generated from an AbelianCat source row, two exact-row ledgers, five vertical maps, four commutative-square rows, comparison rows, a finite chase ledger, Cont replay, hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiveLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8684.1, Definition 8684.2, and Definition 8684.3 seed the finite Five Lemma packet over abelian-category source rows, exact-row ledgers, vertical-square compatibility, comparison rows, bounded chase entries, replay, transport, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiveLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-object equality, ambient diagram extensionality, selected-element semantics, or unbounded diagram-chase completeness.

Upgrade path: Obligation closure requires checked carrier admission, exact-row exposure, vertical-square compatibility, comparison-row exposure, finite chase-ledger construction, replay stability, transport stability, provenance exhaustion, and local naming for BEDC.Derived.FiveLemmaUp.

Chapter 8685

A Concrete Naming Certificate for HyperbolicSectorExpansion$^{\uparrow}$

The HyperbolicSectorExpansion$^{\uparrow}$ packet realizes the sector-expansion branch of chapter 9463 as finite NameCert data. It sits after the sector-envelope packet of chapter 8546, the observer-expansion packet of chapter 4167, the visual-boundary packet of chapter 4181, and the boundary-shadow expansion cocycle of chapter 5814. The carrier names only finite sector aperture, boundary shadow, visual radius, observer window, expansion comparison, transport, refusal, provenance, and local naming rows.

Definition 8685.1 (Hyperbolic sector-expansion carrier)**.** *A* HyperbolicSectorExpansion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, A, B, R, O, G, W, T, F, K, H, C, P, N)$$

with the following displayed rows:

1. *E is the* HyperbolicSectorEnvelope$^{\uparrow}$ *source row carrying the finite sector endpoints, visual-radius row, observer-window row, and refusal row.*
2. *A is the sector-aperture row, naming the visible opening budget of the sector in the observer window.*
3. *B is the boundary-shadow row tying the aperture to a displayed* HyperbolicBoundaryShadow$^{\uparrow}$ *readback.*
4. *R is the visual-radius comparison row inherited from the envelope and read again at the expanded sector boundary.*
5. *O is the observer-window row selecting the finite focus in which the expansion is measured.*
6. *G is the expansion-comparison row, recording the before-window and after-window sector readings and their classifier comparison.*
7. *W is the visual-boundary handoff row through* HyperbolicVisualBoundary$^{\uparrow}$.
8. *T is the transport row preserving aperture, boundary shadow, visual radius, observer window, and expansion comparison under componentwise classifier transport.*
9. *F is the refusal row excluding a completed hyperbolic boundary, hidden geodesic sectors, host metric equality, and a global observer expansion map.*

10. *K is the TasteGate request row requiring* Nontrivial HyperbolicSectorExpansion$^{\uparrow}$ *and* FieldFaithful HyperbolicSector *instances before the row inventory may be accepted by the Lean-side gate.*
11. *H records componentwise* hsame *transport for $E, A, B, R, O, G, W, T, F, K, C, P, N$.*
12. *C records displayed* Cont *replay through sector envelope, aperture, boundary shadow, visual radius, observer window, expansion comparison, transport, refusal, TasteGate requests, and naming.*
13. *P is the* Pkg *provenance over* HyperbolicSectorExpansion$^{\uparrow}$, HyperbolicSectorEnvelope$^{\uparrow}$, HyperbolicObserverExpansi HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, BoundaryShadowExpansionCocycle$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* NameCert$_{\text{HyperbolicSectorExpansion}^{\uparrow}}$ *row.*

The carrier has no coordinate for ambient boundary compactification, selected geodesic representatives, host hyperbolic metric equality, a completed sector family, or a global observer dynamics.

Theorem 8685.2 (Hyperbolic sector-expansion NameCert obligations)**.** *For every accepted* HyperbolicSectorExpans *carrier $X = (E, A, B, R, O, G, W, T, F, K, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicSectorExpansion}^{\uparrow}}$ *surface consists exactly of sector envelope exposure through E, sector aperture through A, boundary shadow through B, visual-radius comparison through R, observer window through O, expansion comparison through G, visual-boundary handoff through W, transport through T, refusal through F, TasteGate requests through K, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

No accepted read may expose an expansion comparison without displaying the aperture row, boundary-shadow row, visual-radius row, observer-window row, refusal row, and provenance row.

Proof. Project the carrier of Definition 8685.1. The envelope row E exposes the endpoint, radius, observer-window, and refusal data on which any sector expansion must depend.

The aperture, boundary-shadow, visual-radius, and observer-window rows A, B, R, O then delimit the measured expansion. The comparison row G can be read only after these coordinates are visible, and the visual-boundary handoff W remains downstream of the finite boundary-shadow surface.

Transport, refusal, TasteGate requests, componentwise history equality, replay, provenance, and naming are then projected from T, F, K, H, C, P, N. Since no hidden geodesic sector, completed boundary, host metric equality, or global observer map appears among the coordinates, the listed rows exhaust the local certificate obligations. □

Theorem 8685.3 (Hyperbolic sector-expansion observer route)**.** *Every accepted* HyperbolicSectorExpansion$^{\uparrow}$ *observer read factors through*

$$\text{HyperbolicSectorEnvelope}^{\uparrow} \longmapsto (A, B, R, O) \longmapsto (G, W, T, F) \longmapsto (K, H, C, P, N).$$

The route exposes finite sector aperture, boundary-shadow readback, visual-radius comparison, observer-window focus, expansion comparison, visual-boundary handoff, transport, refusal, TasteGate requests, replay, provenance, and local NameCert rows.

Proof. Start with the sector-envelope row. It provides the finite endpoint and observer-window context, so the expansion packet does not begin from a completed boundary sector.

Read the aperture, boundary-shadow, visual-radius, and observer-window rows next. These rows fix the local geometry in which an expansion comparison can be displayed.

The expansion comparison, visual-boundary handoff, and transport rows then record the only admitted observer route. Refusal prevents the route from being promoted to host metric equality, hidden geodesic selection, or a global observer dynamics.

Finally the TasteGate request, componentwise transport, replay, provenance, and local naming rows keep the same finite packet available to downstream observer-expansion chapters. □

Falsifiable prediction: An accepted sector-expansion read cannot expose an expansion comparison without displaying the aperture row, boundary-shadow row, visual-radius row, observer-window row, refusal row, and provenance row.

Independence witness: hyperbolic-sector-envelope, hyperbolic-observer-expansion, hyperbolic-visual-boundary, boundary-shadow-expansion-cocycle

The Lean-side TasteGate follow-up for this carrier requires explicit Nontrivial $\mathsf{HyperbolicSectorExpansion}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicSectorExpansion}^{\uparrow}$ instances. Field faithfulness keeps $E, A, B, R, O, G, W, T, F, K, H, C, P$ as distinct rows before a classifier target may accept the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSectorExpansion}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicSectorExpansion}^{\uparrow}$ packet is generated from a sector-envelope source row, sector aperture, boundary shadow, visual radius, observer window, expansion comparison, visual-boundary handoff, transport, refusal, TasteGate requests, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8685.1, Definition 8685.2, and Definition 8685.3 seed the finite observer-expansion sector packet over aperture, boundary shadow, visual radius, observer window, expansion comparison, handoff, transport, refusal, TasteGate, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary sectors, selected geodesic representatives, host hyperbolic metric equality, global observer dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked envelope exposure, aperture admission, boundary-shadow handoff, visual-radius comparison, observer-window exposure, expansion comparison, transport, refusal, TasteGate nontriviality, field faithfulness, replay, provenance, and local naming for BEDC.Derived.HyperbolicSectorExpansionUp.

Chapter 8686

A Concrete Naming Certificate for $\mathsf{LusinTheorem}^{\uparrow}$

Following Lusin and Bishop–Bridges constructive measure theory, the $\mathsf{LusinTheorem}^{\uparrow}$ packet records the finite certificate surface for reading a measurable real-valued row by a compact-domain continuous representative outside a displayed exceptional set. It sits after the measure ledger of chapter 92, compact metric support of chapter 129, continuous-map carrier of chapter 130, regular rational readback of chapter 301, terminal real seal of chapter 38, and approximation route of chapter 4147. The packet names only the displayed finite NameCert obligations for a Lusin-theorem read; it does not assert the full host measure-theoretic theorem, completed measure algebra, quotient almost-everywhere equality, or a choice-selected compact approximation.

Definition 8686.1 (Lusin theorem carrier)**.** *A* $\mathsf{LusinTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, K, C, E, R, D, H, Q, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, carrying the finite measurable-set ledger, finite measure budget, and inspected measurable real-valued row.*
2. *K is the compact approximation row, displaying the compact support or compact-domain window over the same measure ledger and the finite complement budget exposed to consumers.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *representative row, naming the continuous representative over the compact window rather than an ambient function-space element.*
4. *E is the exceptional-set and error ledger, recording the displayed exceptional row, its measure budget, and the tolerance read consumed by the theorem route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row, sealing the measured row, continuous representative, tolerance, and exceptional ledger into the terminal real-valued comparison route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic error budget used to compare the public theorem read with the approximation route of* $\mathsf{LusinApproximation}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for $M, K, C, E, R, D, Q, P, N$ and for the displayed compact approximation and exceptional-set reads.*
8. *Q records displayed* Cont *replay from the measure source through compact approximation, continuous representative, exceptional ledger, real readback, dyadic tolerance, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{LusinTheorem}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LusinApproximation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LusinTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, C, E, R, D, H, Q, P, N$. It has no coordinate for a host measurable-function quotient, a completed measure algebra, a choice-selected approximation, an ambient function space, countable choice, or host $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8686.2 (Lusin theorem NameCert obligations)**.** *For every accepted* $\mathsf{LusinTheorem}^{\uparrow}$ *carrier*

$$L = (M, K, C, E, R, D, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LusinTheorem}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the* $\mathsf{Measure}^{\uparrow}$ *source row M, the compact approximation row K, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *representative row C;*
2. *compact approximation exposure through K, requiring the compact window and its complement budget to be read in the same measure ledger as M;*
3. *continuous representative exactness through C, requiring every public representative read to stay on the displayed compact window;*
4. *exceptional-set and error-ledger exposure through E and the dyadic tolerance row D, requiring each theorem read to display its finite exceptional budget;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal non-escape through R, requiring real-valued readback to remain tied to the measured row, representative row, and tolerance ledger;*
6. *structural stability through H, Q, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve exactly the rows M, K, C, E, R, D.*

No obligation may be discharged by importing a host measurable-function quotient, selecting an approximation outside the displayed compact row, completing the measure algebra, quotienting by almost-everywhere equality, or appealing to countable choice.

Proof. Project the carrier of Definition 8686.1. The measure-facing data are M, K, E, D, the representative data are C, R, and the structural route is H, Q, P, N.

The compact approximation row is read before any continuous representative is exported. Thus the representative in C is attached to the compact window K and to the exceptional-set ledger E, rather than to an ambient measurable-function quotient or a selected approximation.

The real-valued readback is sealed only after the dyadic tolerance row is visible. The row D supplies the finite error budget, while R keeps the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ comparison route tied to the same measured row, continuous representative, and exceptional ledger.

Finally H, Q, P, N transport, replay, source, and name only those displayed rows. Since the excluded host quotient, selected approximation, completed measure algebra, countable choice, and ambient real-completeness coordinates are absent, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{LusinTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LusinTheorem}^{\uparrow}$ packet is generated from finite Measure rows, compact approximation rows, ContinuousMap representative rows, exceptional-set and error ledgers, RegSeqRat and Real readback rows, DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LusinTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 8686.1 and Definition 8686.2 seed the finite compact-approximation and continuous-representative route from Measure and DyadicRatCore tolerance rows to RegSeqRat and Real readback consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{LusinTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the full host Lusin theorem, completed measure algebra, host measurable-function quotient, selected approximation, quotient almost-everywhere equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact approximation exactness, continuous representative exactness, exceptional-set ledger exposure, tolerance sealing, real readback non-escape, transport stability, and replay stability for BEDC.Derived.LusinTheoremUp.

Chapter 8687

A Concrete Naming Certificate for $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and explicit moduli, the $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$ packet records the finite eventual-window route by which a regular Cauchy request exposes a StreamName window, a DyadicRatCore tolerance row, a RegSeqRat readback, a Real seal, and a DiagonalLimitRepresentative handoff. It sits beside the finite-window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the regular rational carrier of chapter 301, the terminal Real carrier of chapter 38, and the diagonal-limit representative packet of chapter 1205. The packet is a choice-free finite tail-window interface; it is not quotient stream equality, a selected host limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8687.1 (Regular Cauchy eventual-window carrier)**.** *A* $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *eventual-window request row naming the finite tail stage and the precision at which the window is read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic comparison bound used inside the tail window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream window and tolerance data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R have been displayed.*
5. *L is the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *handoff row that consumes the same finite window before any diagonal-limit consumer reads a representative.*
6. *H records componentwise* hsame *transport for the window, tolerance, readback, seal, handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a regular Cauchy precision request through the eventual window, dyadic tolerance, RegSeqRat readback, Real seal, and diagonal-limit handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEventualWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, D, R, E, L, H, C, P, N$. It stores no coordinate for a quotient of streams, a chosen limit object, a countable-choice selector, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8687.2 (Regular Cauchy eventual-window NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$ *carrier $W = (S, D, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEventualWindow}}$ *surface consists exactly of carrier admission, eventual-window exposure, dyadic tolerance compatibility, RegSeqRat readback, Real seal handoff, diagonal-limit representative handoff, componentwise transport, displayed replay, provenance, and local naming. A consumer must read the tail stage through S, the tolerance through D, and the regular rational row through R before the* $\mathsf{Real}^{\uparrow}$ *seal E or the diagonal-limit row L is public.*

Proof. Project the carrier of Definition 8687.1. The rows S, D, R, E, L are the only analytic rows available before the structural support rows are read.

The eventual-window exposure starts with S. A downstream consumer receives a displayed finite tail stage and precision request, not a global quotient of streams or a selected limit.

Next read the dyadic tolerance row D and the regular rational readback row R. Compatibility means that the tolerance data belongs to the same tail window and that the RegSeqRat row is obtained from that displayed window.

Finally read E and L. The Real seal and diagonal-limit representative handoff are both downstream of the same finite window. The rows H, C, P, N transport, replay, source, and name only this displayed packet, so the NameCert surface cannot import countable choice or ambient completeness. □

Theorem 8687.3 (Regular Cauchy eventual-window tail handoff)**.** *Every diagonal-limit or Real-completion consumer of an accepted* $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$ *packet reads through the finite tail-window route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{DiagonalLimitRepresentative}^{\uparrow}.$$

In carrier notation, the consumer must expose S, D, R, E, L before it treats the packet as a limit-facing handoff, while H, C, P, N only transport, replay, source, and name the same finite route.

Proof. Begin with the NameCert obligations of Definition 8687.2. They place every Real-facing or diagonal-limit-facing read downstream of the same displayed tail window and dyadic tolerance row.

Read the StreamName and DyadicRatCore rows together. The tail stage is usable only with its tolerance witness, so the route cannot jump directly from a regular Cauchy request to a completed real or a diagonal representative.

The RegSeqRat readback, Real seal, and diagonal-limit handoff close the finite route. Since the support rows H, C, P, N preserve the same packet, no consumer can replace the handoff by quotient stream equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEventualWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEventualWindow}^{\uparrow}$ packet is generated from an eventual StreamName window, dyadic tolerance row, RegSeqRat readback, RealUp seal, diagonal-limit handoff, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEventualWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8687.1, Definition 8687.2, and Definition 8687.3 seed the finite tail-window interface from regular Cauchy requests to diagonal-limit and Real-completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyEventualWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, eventual-window exposure, dyadic tolerance compatibility, RegSeqRat readback, Real seal handoff, diagonal-limit handoff, and structural non-escape for the same finite packet.

Chapter 8688

A Concrete Naming Certificate for $\mathsf{WindingNumber}^{\uparrow}$

Following Poincare and the standard algebraic-topology treatment of loops in the circle, the $\mathsf{WindingNumber}^{\uparrow}$ packet records the finite certificate surface by which a displayed loop into $\mathsf{Circle}^{\uparrow}$ receives an integer degree read through $\mathsf{S}^{1\uparrow}$ and real-coordinate windows. It sits beside the phase-circle certificate of chapter 58, the topological circle boundary of chapter 4010, the continuous-map source of chapter 59, the integer target of chapter 31, and the real-coordinate seal of chapter 38. The chapter names only the finite loop-degree boundary; it does not claim a full fundamental-group computation, classical covering-space lifting, unrestricted homotopy invariance, or quotient path equality.

Definition 8688.1 (Winding-number carrier)**.** *A* $\mathsf{WindingNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (L, S, C, I, R, D, H, T, P, N)$$

with the following displayed rows:

1. *L is the closed loop-source row, naming the displayed loop interval and endpoint match.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-boundary row read from the loop.*
3. *C is the* $\mathsf{Circle}^{\uparrow}$ *topological-circle row through which the same loop is inspected as a boundary object.*
4. *I is the* $\mathsf{Int}^{\uparrow}$ *degree row exported by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate-read row used for finite endpoint and arc-window checks.*
6. *D is the displayed degree-boundary ledger relating loop subdivision windows to the integer row I.*
7. *H records componentwise* hsame *transport for the loop, circle, phase, integer, real, degree-boundary, replay, provenance, and naming rows.*
8. *T records displayed* Cont *replay for the same finite rows.*
9. *P is the* Pkg *provenance over* $\mathsf{WindingNumber}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Circle}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WindingNumber}^{\uparrow}}$ *row.*

The carrier has no coordinate for a universal covering space, a host fundamental group, quotient path classes, selected lifts, or unrestricted homotopy classes.

Theorem 8688.2 (Winding-number NameCert obligations)**.** *For every accepted* $\mathsf{WindingNumber}^{\uparrow}$ *carrier*

$$W = (L, S, C, I, R, D, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WindingNumber}^{\uparrow}}$ *surface consists exactly of loop-source exposure,* $\mathsf{S}^{1\uparrow}$ *phase-boundary exposure,* $\mathsf{Circle}^{\uparrow}$ *boundary exposure,* $\mathsf{CFun}^{\uparrow}$ *loop-continuity provenance,* $\mathsf{Int}^{\uparrow}$ *degree exposure,* $\mathsf{Real}^{\uparrow}$ *coordinate-window exposure, degree-boundary ledger exposure, componentwise transport, displayed replay, package provenance, and local naming. A winding-number consumer must read those finite rows before using the integer degree row* I*.*

Proof. Project the carrier of Definition 8688.1. The loop row L, the $\mathsf{S}^{1\uparrow}$ row S, the $\mathsf{Circle}^{\uparrow}$ row C, the $\mathsf{Real}^{\uparrow}$ row R, and the degree-boundary ledger D are displayed before the integer row is treated as public data.

The continuous-map requirement is part of the provenance row P: it records that the loop is read through the continuous-map certificate rather than through a detached host function. The rows H, T, N only transport, replay, and name the same finite packet.

Since the carrier contains no covering-space lift, quotient path class, host fundamental group, or unrestricted homotopy-class coordinate, the listed rows exhaust the NameCert obligations for the finite winding-number surface. □

Theorem 8688.3 (Winding-number $\mathsf{S}^{1\uparrow}$ loop-degree boundary)**.** *Every* $\mathsf{S}^{1\uparrow}$*-facing loop-degree read from an accepted* $\mathsf{WindingNumber}^{\uparrow}$ *packet factors through*

$$\mathsf{CFun}^{\uparrow} \longmapsto \mathsf{Circle}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Int}^{\uparrow}$$

using the finite rows $L, C, S, R, D, I, H, T, P, N$*. The route exports a displayed integer degree for the loop boundary, but it does not export a full fundamental group theorem, a classical covering-space lift, quotient path equality, or unrestricted homotopy invariance.*

Proof. Start with the displayed loop row L and its continuity provenance inside P. This places the source in the continuous-map certificate surface before any degree read is available.

Next read the topological boundary row C and the phase-circle row S. These rows connect the same loop to $\mathsf{Circle}^{\uparrow}$ and $\mathsf{S}^{1\uparrow}$ without replacing the carrier by a quotient path object or an ambient covering space.

Finally read the real-coordinate windows R, the degree-boundary ledger D, and the integer row I. The structural rows H, T, P, N preserve this order, so the exported degree is exactly the finite loop-degree boundary and none of the refused classical endpoints is supplied. □

Closure status for $\mathsf{NameCert}_{\mathsf{WindingNumber}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WindingNumber}^{\uparrow}$ packet is generated from a displayed loop row, SOne phase row, Circle boundary row, Int degree row, Real coordinate-read row, degree-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WindingNumber}^{\uparrow}}$, seed)

Scope closed: Definition 8688.1, Definition 8688.2, and Definition 8688.3 seed the finite loop-degree carrier, obligation surface, and SOne-facing degree route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WindingNumber}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, the full fundamental group of the circle, classical covering-space theory, quotient path equality, or unrestricted homotopy invariance.

Upgrade path: Obligation closure requires checked carrier admission, loop-continuity provenance, SOne and Circle boundary compatibility, Real coordinate-window exactness, Int degree exposure, transport, replay, provenance, and local naming for the same finite packet.

Chapter 8689

A Concrete Naming Certificate for $\mathsf{AlgorithmWTypeInference}^{\uparrow}$

Following Hindley and Milner, the $\mathsf{AlgorithmWTypeInference}^{\uparrow}$ packet records principal type inference for simply presented lambda terms as finite BEDC readback data. It consumes the $\mathsf{LambdaCalc}^{\uparrow}$ syntax surface of chapter 200, the typed-example coverage rows of chapter 923, and the MetaCIC audit routing of chapter 765. The chapter names only the displayed inference, constraint, unification, principal-type, transport, replay, provenance, and local NameCert rows; it does not import a host typechecker, an ambient unification oracle, quotient equality of types, or a completeness theorem for a host evaluator.

Definition 8689.1 (Algorithm W type-inference carrier)**.** *An* $\mathsf{AlgorithmWTypeInference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, E, M, C, U, T, H, R, P, N)$$

with the following displayed rows:

1. *L is the consumed* $\mathsf{LambdaCalc}^{\uparrow}$ *term packet. It exposes variables, abstractions, applications, substitution rows, alpha-beta classifier rows, and constructor case data as finite syntax rows.*
2. *E is the typed-example coverage row consumed from* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$*. It records which variable, abstraction, application, let-shaped, and failure-facing examples are covered by the inference packet before any checker-audit consumer reads it.*
3. *M is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *audit row. It records the critical-path cells that will later consume inferred typing rows, normalization-facing rows, and checker-audit rows.*
4. *C is the constraint-generation ledger. Each entry names the source syntax row, context row, fresh type-variable row, and generated equality constraint for the visible constructor case.*
5. *U is the unification ledger. It records only finite occurs-check, substitution-extension, constraint-discharge, and residual-constraint rows.*
6. *T is the principal-type output row. It names the inferred type scheme, the solved substitution, the residual context, and the proof that every exported typed-example row reads through the same solved ledger.*
7. *H records componentwise* hsame *transport for the lambda syntax, typed-example coverage, MetaCIC audit, constraint, unification, principal-type, replay, provenance, and naming rows.*

8. *R records displayed* Cont *replay from L, E, M into C, from C into U, and from U into T before any MetaCIC typing or normalization audit route consumes the result.*
9. *P is the* Pkg *provenance over* $\mathsf{AlgorithmWTypeInference}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlgorithmWTypeInference}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host typechecker, ambient type-scheme equality, host unification state, hidden principal-type theorem, or imported MetaCIC typing judgment.

Theorem 8689.2 (Algorithm W type-inference NameCert obligations)**.** *For every accepted* AlgorithmWTypeInferen *carrier $A = (L, E, M, C, U, T, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlgorithmWTypeInference}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed lambda syntax, typed-example coverage, MetaCIC critical-path, constraint-generation, unification, principal-type, transport, continuation, provenance, and naming rows;*
2. *syntax coverage, requiring every inference read to enter through the consumed* $\mathsf{LambdaCalc}^{\uparrow}$ *constructor row of L;*
3. *typed-example coverage, requiring each exported example-facing typing row to be listed in E before it is consumed by a checker-audit route;*
4. *constraint generation, requiring constructor cases to append only the displayed context, fresh type-variable, and equality-constraint rows in C;*
5. *unification ledger exactness, requiring every solved substitution and residual constraint to be read from U rather than from a host unifier;*
6. *principal-type output, requiring the public type scheme and solved substitution to be exposed through T after C and U have been displayed;*
7. *transport, replay, provenance, and naming through H, R, P, N.*

No Algorithm W obligation is stored outside these rows.

Proof. Project the carrier of Definition 8689.1. Its coordinates list the lambda source, coverage row, MetaCIC audit row, constraint ledger, unification ledger, principal-type output, and structural rows.

The syntax and coverage obligations are forced by L and E. A consumer cannot ask for an inference row until the corresponding $\mathsf{LambdaCalc}^{\uparrow}$ constructor row and typed-example coverage cell have been displayed.

Constraint generation and unification are the roles of C and U. The packet records fresh type-variable rows, equality constraints, occurs-check rows, substitution extensions, and residual constraints as finite ledger entries, so a host typechecker or ambient unification oracle has no coordinate from which to enter.

Finally, the principal output T is downstream from C and U, while H, R, P, N transport, replay, package, and name the same finite route. Hence the local NameCert surface is exhausted by the rows stated above. □

Theorem 8689.3 (Algorithm W type-inference principal-type route)**.** *Every principal-type read exported by an accepted* $\mathsf{AlgorithmWTypeInference}^{\uparrow}$ *packet factors through*

$$\mathsf{LambdaCalc}^{\uparrow}, \mathsf{TypedExampleCoverageMatrix}^{\uparrow}, \mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto C \longmapsto U \longmapsto T \longmapsto H, R, P, N.$$

Thus a MetaCIC typing or normalization audit route may consume the inferred type scheme only after the lambda syntax row, typed-example coverage row, critical-path row, constraint-generation ledger, unification ledger, principal-type output row, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row have been displayed.*

Proof. Begin with a principal-type request. The source term must first be read as a $\mathsf{LambdaCalc}^{\uparrow}$ packet, and the typed-example consumer must read the corresponding coverage row of TypedExampleCoverageMatrix

The MetaCIC audit route is then constrained by M. It can request typing and normalization-facing rows only through the critical-path cells already displayed in the carrier.

Constraint generation sends the request into C, and unification discharges those generated constraints through U. The principal type scheme is exposed only by T, together with the solved substitution and residual context recorded in the same finite ledger.

The structural rows H, R, P, N preserve this route under transport and replay. Since the carrier has no host typechecker, ambient unifier, quotient type equality, hidden principal-type theorem, or imported MetaCIC typing judgment, the exported principal-type read is exactly the displayed route named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlgorithmWTypeInference}^{\uparrow}}$.

Constructive story: An $\mathsf{AlgorithmWTypeInference}^{\uparrow}$ packet is generated from LambdaCalc syntax rows, typed-example coverage rows, MetaCIC critical-path rows, a finite constraint-generation ledger, a finite unification ledger, a principal-type output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlgorithmWTypeInference}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8689.1, Definition 8689.2, and Definition 8689.3 seed the finite Algorithm W inference packet over displayed syntax, coverage, critical-path, constraint, unification, principal-type, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlgorithmWTypeInference}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host typechecker, ambient unification oracle, quotient equality of types, a full Hindley-Milner completeness theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constructorwise constraint generation, unification ledger exactness, principal-type output, MetaCIC audit handoff, transport, replay, provenance, and local NameCert rows for BEDC.Derived.AlgorithmWTypeInferenceUp.

Chapter 8690

A Concrete Naming Certificate for $\mathsf{RieszMarkovKakutani}^{\uparrow}$

Following Riesz, Markov, and Kakutani, the $\mathsf{RieszMarkovKakutani}^{\uparrow}$ packet records the finite BEDC surface that routes a positive linear functional on displayed continuous functions over a compact source to a Radon-measure handoff. It sits over the compact source rows of chapter 129, the function rows of chapter 130, the real-name seal of chapter 38, and the measure rows of chapter 2263. The packet names the carrier, positivity ledger, linearity ledger, compact-support restriction, finite approximation route, measure handoff, transport, replay, provenance, and local naming; it does not import host dual spaces, selected representing measures, open-cover compactness, quotient function equality, or countable choice.

Definition 8690.1 (Riesz–Markov–Kakutani carrier)**.** *A* $\mathsf{RieszMarkovKakutani}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, V, P, L, A, M, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact domain surface that every continuous-function read must inspect.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *function-family row over K, with target observations sealed through* $\mathsf{Real}^{\uparrow}$ *rows.*
3. *V is the real-valued functional row, recording the finite evaluation requests and real-name outputs read by the positive functional.*
4. *P is the positivity ledger: nonnegative function rows in F are sent to nonnegative* $\mathsf{Real}^{\uparrow}$ *output rows.*
5. *L is the linearity ledger, recording displayed addition and scalar-multiplication compatibility for the finite function rows read by the packet.*
6. *A is the approximation row, routing compact finite probes and continuous-function modulus data to the finite test family used by the measure handoff.*
7. *M is the* $\mathsf{RadonMeasure}^{\uparrow}$ *handoff row, exposing the measure candidate only after K, F, V, P, L, A are visible.*

8. *H records componentwise* hsame *transport for the compact source, function family, real-valued functional, positivity, linearity, approximation, measure handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same rows.*
10. *Q is the* Pkg *provenance over* $\mathsf{RieszMarkovKakutani}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszMarkovKakutani}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient dual Banach space, a selected representing measure, quotient equality of continuous functions, arbitrary open-cover compactness, countable choice, or a completed integration theory.

Theorem 8690.2 (Riesz–Markov–Kakutani NameCert obligations)**.** *For every accepted* $\mathsf{RieszMarkovKakutani}^{\uparrow}$ *carrier*

$$R = (K, F, V, P, L, A, M, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszMarkovKakutani}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, continuous-function admission through F, real-valued functional exposure through V, positivity through P, linearity through L, finite approximation through A, Radon-measure handoff through M, componentwise transport through H, displayed replay through C, package provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 8690.1. The rows K, F, V display the compact domain, continuous real-valued function data, and finite functional evaluations before any measure handoff is public.

Read the positivity and linearity ledgers next. They record exactly the functional laws needed by the handoff and do not introduce a host dual space or quotient equality of functions.

The approximation row A then ties compact finite probes to the continuous-function rows, and the measure row M is exposed only after that route is visible. Finally H, C, Q, N preserve, replay, source, and name the same packet, so the listed rows exhaust the local certificate obligations. □

Theorem 8690.3 (Riesz–Markov–Kakutani positive functional measure route)**.** *Every downstream* $\mathsf{RadonMeasure}^{\uparrow}$ *consumer of an accepted* $\mathsf{RieszMarkovKakutani}^{\uparrow}$ *packet must first read the compact source row K, the continuous-function row F, the real-valued functional row V, the positivity ledger P, the linearity ledger L, and the finite approximation row A. The measure handoff is therefore a displayed route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto (P, L, A) \longmapsto \mathsf{RadonMeasure}^{\uparrow}$$

inside the packet, not a selected representing-measure theorem in the host metatheory.

Proof. Start from the obligation theorem Definition 8690.2. It places the compact source, function family, functional row, positivity row, linearity row, and approximation row before the measure handoff.

The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows provide the finite domain and function observations, while $\mathsf{Real}^{\uparrow}$ seals the functional outputs. Positivity and linearity remain displayed ledgers attached to those observations.

Only after these rows are visible may a $\mathsf{RadonMeasure}^{\uparrow}$ consumer inspect M. Since the carrier has no selected measure coordinate, no quotient function equality, and no countable-choice coordinate, the exported measure read is exactly the displayed handoff route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszMarkovKakutani}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszMarkovKakutani}^{\uparrow}$ packet is generated from a CompactMetric source row, ContinuousMap real-valued function rows, RealUp functional outputs, positivity and linearity ledgers, finite approximation rows, a RadonMeasure handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszMarkovKakutani}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8690.1, Definition 8690.2, and Definition 8690.3 seed the positive-functional-to-measure NameCert surface over compact metric, continuous map, real-name, and Radon measure rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszMarkovKakutani}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a selected representing measure, host dual-space equality, quotient equality of continuous functions, open-cover compactness, countable choice, or a completed integration theory.

Upgrade path: Obligation closure requires checked carrier admission, positivity exposure, linearity exposure, compact approximation routing, RadonMeasure handoff stability, replay stability, transport stability, provenance exhaustion, and local naming for BEDC.Derived.RieszMarkovKakutaniUp.

Chapter 8691

A Concrete Naming Certificate for $\mathsf{LoewenheimSkolem}^{\uparrow}$

Following Skolem and Tarski, the $\mathsf{LoewenheimSkolem}^{\uparrow}$ packet records the finite certificate surface by which a first-order theory request exposes a generated Skolem submodel boundary. It sits beside the syntax-and-deduction carrier of chapter 197, the satisfaction ledger of chapter 198, the displayed membership discipline of chapter 3668, and the typed-family route of chapter 3600. The packet names only a finite language row, a finite formula-family row, a generated Skolem-closure row, a submodel-witness row, a satisfaction ledger, transport, replay, provenance, and local naming; it does not assert a host cardinal theorem, selected elementary chain, quotient model equality, or unrestricted choice of witnesses.

Definition 8691.1 (Loewenheim-Skolem carrier)**.** *A* $\mathsf{LoewenheimSkolem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, F, S, M, V, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite language row, read through* $\mathsf{FirstOrder}^{\uparrow}$*, listing the function, relation, and constant symbols inspected by the packet.*
2. *F is the formula-family row, also read through* $\mathsf{FirstOrder}^{\uparrow}$*, exposing the formulas whose witnesses and truth values are tracked.*
3. *S is the generated Skolem-closure row. It records the finite closure steps that add named witness terms for the displayed formulas rather than a host-level closure operator.*
4. *M is the submodel-witness row, read through* $\mathsf{ModelTheory}^{\uparrow}$ *and* $\mathsf{SetLike}^{\uparrow}$*, displaying membership and interpretation compatibility for the generated carrier fragment.*
5. *V is the satisfaction ledger, read through* $\mathsf{ModelTheory}^{\uparrow}$*, recording that every inspected formula family keeps the same displayed truth row when evaluated inside the generated fragment.*
6. *H records componentwise* hsame *transport for the language, formula-family, Skolem-closure, submodel, satisfaction, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a submodel request through language exposure, formula exposure, closure generation, membership inspection, satisfaction readback, provenance, and local naming.*

8. *P is the* Pkg *provenance over* LoewenheimSkolem$^{\uparrow}$, FirstOrder$^{\uparrow}$, ModelTheory$^{\uparrow}$, SetLike$^{\uparrow}$, TypeLike$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{LoewenheimSkolem}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient set-theoretic universe, a selected infinite elementary chain, quotient equality of models, or a cardinal arithmetic theorem.

Theorem 8691.2 (Loewenheim-Skolem NameCert obligations)**.** *For every accepted* LoewenheimSkolem$^{\uparrow}$ *carrier*

$$L = (G, F, S, M, V, H, C, P, N),$$

the local NameCert$_{\text{LoewenheimSkolem}^{\uparrow}}$ *surface consists exactly of finite language exposure through G, formula-family exposure through F, generated Skolem-closure exposure through S, submodel-witness exposure through M, satisfaction-ledger exposure through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every consumer of the generated submodel boundary must read those rows before citing a Loewenheim-Skolem conclusion.*

Proof. Project the carrier of Definition 8691.1. The first two rows G, F fix the first-order language and the formula family whose witnesses are being generated; they are read only through the FirstOrder$^{\uparrow}$ certificate surface.

Next inspect the generated closure and submodel rows. The row S records the finite Skolem steps, while M records membership and interpretation compatibility through the ModelTheory$^{\uparrow}$ and SetLike$^{\uparrow}$ siblings. These rows supply the displayed submodel boundary without selecting an ambient chain or quotient model.

Now read the satisfaction ledger V. It ties each inspected formula in F to the same displayed truth row before and after the generated-fragment read, so the packet exports only the finite satisfaction boundary that has been named in the carrier.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for an external cardinal theorem, host universe, chosen elementary chain, or quotient model equality, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A consumer that claims a LoewenheimSkolem$^{\uparrow}$ generated submodel while omitting the finite language row, formula-family row, Skolem-closure row, submodel-witness row, satisfaction ledger, hsame, Cont, Pkg, or local NameCert row refutes the carrier of Definition 8691.1.

Independence witness: firstorder, modeltheory, setlike, typelike: LoewenheimSkolem$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite language, formula-family, Skolem-closure, submodel-witness, satisfaction-ledger, transport, replay, provenance, and local NameCert rows. FirstOrderUp supplies syntax and deduction without generated submodels; ModelTheoryUp supplies satisfaction rows without a Skolem-closure boundary; SetLikeUp supplies membership without first-order witness generation; TypeLikeUp supplies displayed families without the model-theoretic satisfaction ledger.

Closure status for NameCert$_{\text{LoewenheimSkolem}^{\uparrow}}$.

Constructive story: A LoewenheimSkolem$^{\uparrow}$ packet is generated from a finite language row, formula-family row, generated Skolem-closure row, submodel-witness row, satisfaction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LoewenheimSkolem}^{\uparrow}}$, seed)

Scope closed: Definition 8691.1 and Definition 8691.2 seed the generated-submodel carrier and obligation surface over FirstOrder$^{\uparrow}$, ModelTheory$^{\uparrow}$, SetLike$^{\uparrow}$, TypeLike$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{LoewenheimSkolem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, an external cardinal theorem, selected elementary chains, quotient model equality, or unrestricted witness choice.

Upgrade path: Obligation closure requires checked carrier admission, finite language exposure, formula-family exposure, generated Skolem-closure exposure, submodel-witness exposure, satisfaction-ledger stability, transport, replay, provenance, and local naming for the same packet.

Chapter 8692

A Concrete Naming Certificate for $\mathsf{EnrichedMetricSpace}^{\uparrow}$

Following Lawvere enriched metric spaces and the Isbell–Yoneda completion tradition, the $\mathsf{EnrichedMetricSpace}^{\uparrow}$ packet records the distance-as-hom profile that L10 completion routes need before a $\mathsf{LawvereMetric}^{\uparrow}$ or $\mathsf{YonedaCompletion}^{\uparrow}$ consumer reads completion-facing data. It sits over the metric carrier of chapter 57, the formal-ball surface of chapter 1510, the dyadic and regular rational readbacks of chapter 291 and chapter 301, the terminal real seal of chapter 38, the directed-distance packet of chapter 7212, and the Yoneda-profile route of chapter 2148. The packet names only the finite enriched-distance profile shared by those readers; it does not construct a quotient completion, select colimits, import host enriched categories, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8692.1 (EnrichedMetricSpace carrier)**.** *An* $\mathsf{EnrichedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, D, U, T, B, Y, R, H, C, P, N)$$

whose rows have the following public roles:

1. *X is the displayed source-object row, read through the ordinary* $\mathsf{Met}^{\uparrow}$ *sibling when a symmetric metric source is present.*
2. *D is the enriched distance or hom-profile row, exposing directed values before any* $\mathsf{LawvereMetric}^{\uparrow}$ *or* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *consumer reads them.*
3. *U is the unit row, certifying the displayed self-distance or identity-hom value.*
4. *T is the triangle or composition row, certifying the finite comparison that composes two displayed hom values into a third.*
5. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *probe row, carrying radius-indexed observations of the same profile.*
6. *Y is the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row, recording the finite presheaf-style read seen by completion consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row, with* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets attached before any terminal distance read.*
8. *H is componentwise* hsame *transport for X, D, U, T, B, Y, R.*
9. *C is the displayed* Cont *replay row for the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{EnrichedMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LawvereMetric}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EnrichedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient completion, selected colimit, host enriched category, ambient real-completeness theorem, choice functional, Lean proof credential, or standard bridge.

Theorem 8692.2 (EnrichedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{EnrichedMetricSpace}^{\uparrow}$ *carrier* $E = (X, D, U, T, B, Y, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EnrichedMetricSpace}^{\uparrow}}$ *surface consists exactly of seven finite obligation rows: carrier admission for the displayed source and profile rows; unit exactness through U; triangle or composition exactness through T;* $\mathsf{FormalBall}^{\uparrow}$ *and* $\mathsf{YonedaCompletion}^{\uparrow}$ *handoff through B, Y;* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *nonescape through R with* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets; classifier transport through H, C; and provenance exhaustion through P, N.*

Proof. Project the carrier of Definition 8692.1. The rows $X, D, U, T, B, Y, R, H, C, P, N$ are the entire public surface, so carrier admission is exactly admission of those displayed rows.

The unit and triangle obligations are read from U and T. They attach identity and composition evidence to the same enriched distance row D, so a reader cannot replace them by a host enriched category or by an untracked order law.

The completion-facing handoff is the pair B, Y. The formal-ball probe row displays finite center-radius observations, and the Yoneda row reads the same enriched profile before any $\mathsf{LawvereMetricCompletion}^{\uparrow}$ route can inspect a completion seal.

Finally R, H, C, P, N seal the route. The real-valued read passes through $\mathsf{DyadicRatCore}^{\uparrow}$ budgets, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ sealing; transport, replay, provenance, and local naming preserve the same packet and add no quotient completion, selected colimit, choice principle, Lean credential, or bridge check. □

Theorem 8692.3 (EnrichedMetricSpace Lawvere-Yoneda route)**.** *Every* $\mathsf{LawvereMetric}^{\uparrow}$ *and* $\mathsf{YonedaCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{EnrichedMetricSpace}^{\uparrow}$ *carrier reads the same enriched distance profile through*

$$\mathsf{EnrichedMetricSpace}^{\uparrow} \longrightarrow \mathsf{FormalBall}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before any completion-facing seal is available. Thus $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *receives finite FormalBall, dyadic-budget, regular-rational, and real-seal data for the displayed profile, not a quotient completion, selected colimit, host enriched category, ambient completeness theorem, or choice-selected representative.*

Proof. Start with the NameCert obligations of Definition 8692.2. They force the source row, enriched distance row, unit row, triangle row, formal-ball probe, Yoneda profile, real-seal row, transport, replay, provenance, and local naming to remain rows of one packet.

A $\mathsf{LawvereMetric}^{\uparrow}$ consumer reads D, U, T as the directed-distance profile and its unit-composition ledger. A $\mathsf{YonedaCompletion}^{\uparrow}$ consumer reads the same profile only after B has exposed finite formal-ball probes.

The terminal read then passes through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ inside R. Since the carrier has no coordinate for quotient completion, selected colimits, host enriched categories, ambient completeness, or choice, both consumers receive exactly the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{EnrichedMetricSpace}^{\uparrow}$ read that lets a $\mathsf{LawvereMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, or $\mathsf{LawvereMetricCompletion}^{\uparrow}$ consumer inspect a completion-facing seal without first displaying $D, U, T, B, Y, R, H, C, P, N$ is outside the carrier surface.

Independence witness: metric, lawvere-metric, formalball, yoneda-completion, dyadicratcore, regseqrat, real: $\mathsf{EnrichedMetricSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds one enriched hom-profile to unit, triangle, FormalBall, Yoneda, dyadic-budget, regular-rational, Real-seal, transport, replay, provenance, and local NameCert rows while refusing quotient completion and host enriched categories.

Closure status for $\mathsf{NameCert}_{\mathsf{EnrichedMetricSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{EnrichedMetricSpace}^{\uparrow}$ packet is generated from a displayed source row, enriched distance or hom-profile row, unit row, triangle or composition row, FormalBall probe row, Yoneda profile row, RegSeqRat and Real seal row with DyadicRatCore budgets, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EnrichedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8692.1, Definition 8692.2, and Definition 8692.3 seed the finite enriched-distance profile shared by LawvereMetric, YonedaCompletion, FormalBall, DyadicRatCore, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EnrichedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.EnrichedMetricSpaceUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected colimits, host enriched categories, ambient Real completeness, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exactness, triangle or composition exactness, FormalBall and Yoneda handoff, Real-seal nonescape, classifier transport, provenance exhaustion, and local naming for BEDC.Derived.EnrichedMetricSpaceUp.

Chapter 8693

A Concrete Naming Certificate for $\mathsf{CompletionReflectorMonad}^{\uparrow}$

Following Mac Lane on reflective subcategories and Bishop–Bridges/Kelley on metric completion, the $\mathsf{CompletionReflectorMonad}^{\uparrow}$ packet records the finite service boundary where completion is read simultaneously as a reflector and as a monad. It sits between the source metric rows of chapter 57, the displayed completion routes of chapter 1506, the monad rows of chapter 378, the reflector-unit route of chapter 8626, and the minimality route of chapter 1606. The packet is a finite NameCert carrier for completion-facing unit, bind, join, separated uniqueness, transport, replay, provenance, and local naming rows.

Definition 8693.1 (Completion reflector monad carrier)**.** *A* $\mathsf{CompletionReflectorMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, U, J, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, exposing the source carrier, source classifier, distance observations, zero-distance row, symmetry row, triangle row, and distance-transport ledger consumed before any completion read is public.*
2. *R is the completion reflector row, recording the displayed completion object and reflector handoff through the existing metric-completion, reflector-unit, and minimality packets.*
3. *U is the unit embedding row from the source metric into the displayed completion surface, carrying only the finite graph of inspected source points and their completion names.*
4. *J is the bind and join schedule, read through the completion monad rows and the finite Kleisli route before a completion-valued consumer may flatten a nested completion request.*
5. *E is the separated uniqueness row, recording that two displayed extensions agree only through the separated completion comparison already named by the reflector boundary.*
6. *H records componentwise* hsame *transport for the source metric, reflector, unit embedding, bind and join schedule, separated uniqueness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a source metric request through R, U, J, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CompletionReflectorMonad}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}$ $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompletionReflectorMonad}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion, selected representatives, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8693.2 (Completion reflector monad NameCert obligations)**.** *For every accepted* CompletionReflectorMon *carrier*

$$K = (S, R, U, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionReflectorMonad}^{\uparrow}}$ *surface consists exactly of source metric admission through S, completion reflector exposure through R, unit embedding through U, bind and join scheduling through J, separated uniqueness through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8693.1. The source metric data is confined to S, while the completion object and reflector handoff are confined to R through the named metric-completion, reflector-unit, and minimality packets.

The unit embedding row U is the only source-to-completion entrance. The bind and join schedule J is the only row that may flatten a completion-valued route, and it is read through the existing completion monad, unit, Kleisli, and multiplication packets.

The separated uniqueness row E supplies the reflector-side uniqueness boundary, and H, C, P, N transport, replay, source, and name the same finite packet. No other coordinate is available in the carrier, so the listed rows exhaust the local NameCert surface. □

Theorem 8693.3 (Completion reflector monad unit and bind laws)**.** *Every unit-facing or bind-facing read of an accepted* $\mathsf{CompletionReflectorMonad}^{\uparrow}$ *carrier factors through*

$$S, R, U, J, E, H, C, P, N.$$

The unit law is public only after the source metric row S, reflector row R, and unit embedding row U have been displayed. The bind and join laws are public only after the monad schedule J is read together with the separated uniqueness row E and the structural rows H, C, P, N.

Proof. Begin with a unit-facing request. The source metric row S and reflector row R display the domain and completion boundary, and U is the only row that admits source points into the completion surface.

For a bind-facing request, the carrier supplies no direct host operation on completed objects. The read must instead pass through J, where the completion monad, unit, Kleisli, and multiplication packets provide the finite flattening schedule.

Separated uniqueness is then read through E. It prevents a consumer from treating the unit or bind laws as host equality of representatives, while H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. □

Theorem 8693.4 (Completion reflector monad extension uniqueness)**.** *Let*

$$K = (S, R, U, J, E, H, C, P, N)$$

be an accepted $\mathsf{CompletionReflectorMonad}^{\uparrow}$ *carrier. If two completion-facing extension reads use the same source metric row S, reflector row R, unit embedding U, bind and join schedule J, separated uniqueness row E, componentwise transport H, displayed replay C, provenance P, and local naming row N, then they expose the same completion-facing public route.*

Proof. Project the shared carrier coordinates $S, R, U, J, E, H, C, P, N$. The source metric, completion reflector, unit embedding, and bind and join schedule are therefore read from the same finite packet for both extension requests.

Apply Definition 8693.3 to each read. The unit-facing part must pass through S, R, U, and the bind-facing part must pass through J before any completion-valued route is public.

The separated uniqueness row E compares the two displayed extensions after those operational rows have been exposed. Transport, replay, provenance, and naming through H, C, P, N carry that comparison back to the public route. Since the carrier has no coordinate for quotient completion, selected representatives, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the uniqueness claim is only the equality of the displayed completion-facing route. □

Theorem 8693.5 (Completion reflector monad nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionReflectorMonad}^{\uparrow}$ *carrier reads only the displayed finite route*

$$S, R, U, J, E, H, C, P, N.$$

In particular, the packet exports no quotient completion, no selected representatives, no host equality of completed objects, no countable choice, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8693.2. It identifies the only public carrier rows as the source metric, reflector, unit embedding, bind and join schedule, separated uniqueness, transport, replay, provenance, and local naming rows.

Use Definition 8693.3 for the operational reads. Unit-facing and bind-facing consumers must pass through the same displayed packet before any completion law is available.

The excluded principles have no row in K. A request for quotient completion, selected representatives, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness therefore stays outside the carrier rather than becoming a hidden continuation through H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionReflectorMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionReflectorMonad}^{\uparrow}$ packet is generated from source metric rows, a completion reflector row, a unit embedding row, a bind and join schedule, separated uniqueness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionReflectorMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8693.1, Definition 8693.2, Definition 8693.3, and Definition 8693.5 seed the finite reflector-monad service boundary over source metric, completion reflector, unit embedding, bind and join, separated uniqueness, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionReflectorMonad}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, host equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector exposure, unit embedding, bind and join scheduling, separated uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.CompletionReflectorMonadUp.

Chapter 8694

A Concrete Naming Certificate for MetricSeparatedReflectorUnit$^{\uparrow}$

Following Bishop and Bridges on metric apartness and separated reflection, the MetricSeparatedReflectorUnit$^{\uparrow}$ packet records the finite unit object that sends a displayed pseudometric source into its separated metric reflection before completion-facing consumers read the result. It sits after the pseudometric source of chapter 1614, the separated-reflection route of chapter 4151, the separated metric carrier of chapter 1411, the Hausdorffization packet of chapter 7153, and the metric completion boundary of chapter 1506. The packet names the unit and its factorization ledger without selecting quotient representatives, importing host equality, claiming completed reals, or invoking countable choice.

Definition 8694.1 (MetricSeparatedReflectorUnit carrier)**.** *A* MetricSeparatedReflectorUnit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, Z, S, U, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric or pseudometric source row, exposing point admission, distance readback, symmetry, triangle, and zero-distance inspection.*
2. *Z is the zero-distance witness row consumed from M, recording exactly which source pairs are identified by the separated reflection.*
3. *S is the separated target row, carrying the reflected separated metric classifier and its public point surface.*
4. *U is the unit arrow row from the source row M into the separated target S.*
5. *F is the factorization ledger. It records that any separated metric consumer receiving a compatible arrow from M must read the arrow through U and then through a displayed target arrow out of S.*
6. *H records componentwise* hsame *transport for the source metric row, zero-distance witness, separated target, unit arrow, factorization ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a source metric request through M, Z, S, U, F, H before any completion or Hausdorff consumer reads the result.*
8. *P is the* Pkg *provenance over* MetricSeparatedReflectorUnit$^{\uparrow}$, Pseudometric$^{\uparrow}$, SeparatedReflection$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, Hausdorffization$^{\uparrow}$, MetricCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, Z, S, U, F, H, C, P, N$. It has no coordinate for quotient representatives, host equality of reflected points, selected section maps, completed limits, or ambient real completeness.

Theorem 8694.2 (MetricSeparatedReflectorUnit NameCert obligations)**.** *For every accepted* MetricSeparatedRefle *carrier $R = (M, Z, S, U, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}}$ *surface consists exactly of source metric exposure, zero-distance witness exposure, separated target exposure, unit-arrow exposure, factorization ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every separated or completion-facing consumer must read*

$$M, Z, S, U, F, H, C, P, N$$

before treating the separated reflection as public.

Proof. Project the carrier of Definition 8694.1. The rows $M, Z, S, U, F, H, C, P, N$ are exactly the displayed coordinates available to the local certificate.

The source metric and zero-distance rows are read first. The row Z is the only place where source points become identified, so a consumer cannot obtain reflected equality from a hidden quotient representative or host equality predicate.

The unit and factorization rows then expose the reflector-unit surface. The arrow row U supplies the map into the separated target S, and the ledger F records how every compatible arrow into a separated metric consumer factors through that unit.

Transport and replay preserve the same route. Componentwise hsame in H aligns each displayed coordinate, C replays the request through those coordinates, and P, N source and name the packet. Since quotient representatives, selected sections, completed limits, and ambient real completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 8694.3 (MetricSeparatedReflectorUnit factorization route)**.** *Let $R = (M, Z, S, U, F, H, C, P, N)$ be an accepted* $\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}$ *carrier. Any downstream separated metric, Hausdorffization, or metric-completion consumer that receives a source-compatible arrow from M must read the route*

$$M, Z \longmapsto S, U, F \longmapsto H, C, P, N$$

before using the target as a separated metric object.

Proof. Start with the accepted carrier and project M and Z. These rows determine the pseudometric source and the zero-distance identifications that the separated target is allowed to recognize.

Apply Definition 8694.2. The separated target S, unit arrow U, and factorization ledger F are exposed only after the zero-distance witness has been displayed, and every public read continues through H, C, P, N.

A Hausdorffization or metric-completion consumer may consume the separated target, but it receives it through the unit route rather than through a quotient representative or selected completed point. Thus the factorization surface is the finite bridge from the pseudometric source to separated completion-facing use, with no completed real or host equality claim added by the packet. □

Theorem 8694.4 (MetricSeparatedReflectorUnit universal factorization)**.** *Let*

$$R = (M, Z, S, U, F, H, C, P, N)$$

be an accepted $\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}$ *carrier. Any two separated-metric consumers receiving source-compatible arrows from the same pseudometric source row* M *factor through the same displayed unit row* U *whenever they agree on the zero-distance witness row* Z *and the separated target classifier row* S*.*

Proof. Project $M, Z, S, U, F, H, C, P, N$ from the accepted carrier. The source-compatible arrows begin with the same pseudometric source row and the same zero-distance witness row.

Use the factorization ledger F for each consumer. The ledger sends every separated-metric consumer through the displayed unit row U before a target arrow out of S can be read.

Agreement on Z and on the separated target row S identifies the displayed factorization route for the two consumers. The rows H, C, P, N then replay, transport, source, and name the common route. The carrier never supplies quotient representatives, selected sections, completed limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the conclusion is exactly the shared displayed factorization through U. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}$ packet is generated from a source metric or pseudometric row, a zero-distance witness row, a separated target row, a unit arrow, a factorization ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8694.1, Definition 8694.2, and Definition 8694.3 seed the finite separated-reflector unit interface over source metric data, zero-distance witnesses, unit routing, factorization, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient representatives, selected sections, completed reals, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance exposure, separated target exposure, unit-arrow routing, factorization ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.MetricSeparatedReflec

Chapter 8695

A Concrete Naming Certificate for $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$

The $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ packet realizes the lift-flip and quotient support-flip rows of Definition 9462.1690, Definition 9462.1691, and Definition 9462.1692 as finite NameCert data. It sits after the minimal trigger lift bundle of chapter 4129, the nine-orbit minimal trigger classification of chapter 4192, and the codon lift orbit classifier of chapter 4059. The packet records only quotient supports, lift fibers, finite flip edges, edge-budget rows, transport, replay, provenance, and local naming; it is not a biological mechanism, stochastic dynamics, hidden enumeration oracle, or unbounded graph search.

Definition 8695.1 (FiniteTriggerFlipGraph carrier)**.** *A* $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, L, E, B, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient-trigger base row, listing the finite quotient-level minimal triggers by size and orbit source.*
2. *L is the lift-choice fiber row, recording the binary f_3 choices over each displayed quotient support.*
3. *E is the support and lift flip-edge row. Inside a fixed lift fiber it records single-coordinate lift flips, and inside a fixed quotient-size layer it records support replacements with symmetric difference two.*
4. *B is the edge-budget row, including the internal lift-flip count* 397 *under the displayed finite inventory.*
5. *H records componentwise* hsame *transport for quotient triggers, lift-choice fibers, edge rows, edge budgets, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a finite trigger query through Q, L, E, B, H before any TasteGate or classifier consumer reads the graph.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$, $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$, $\mathsf{MinimalTriggerOrbitClassificatio}$ $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTriggerFlipGraph}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through Q, L, E, B, H, C, P, N. It has no coordinate for hidden support semantics, stochastic transitions, biological fitness, unbounded graph enumeration, or a graph object outside the displayed finite rows.

Theorem 8695.2 (FiniteTriggerFlipGraph NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ *carrier* $G = (Q, L, E, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteTriggerFlipGraph}^{\uparrow}}$ *surface consists exactly of quotient-trigger base exposure, lift-choice fiber exposure, support and lift flip-edge exposure, edge-budget exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A future Lean TasteGate file for this packet must include the nontriviality and field-faithfulness instance surface for the displayed graph rows before any classifier consumer reads the graph as admitted.*

Proof. Project the carrier of Definition 8695.1. The only available rows are Q, L, E, B, H, C, P, N, so every graph read begins with the quotient-trigger base and the lift-choice fibers.

The edge row E then separates the two finite adjacency sources. Lift flips remain inside one lift fiber and change exactly one binary lift coordinate, while support flips remain inside one quotient-size layer and change exactly one support by symmetric difference two.

The edge-budget row B binds the finite counts to the displayed inventory. Transport in H and replay in C preserve the same quotient, lift, edge, and budget rows, and P, N record provenance and local naming. Therefore no hidden enumeration oracle, stochastic transition law, biological mechanism, or off-packet graph coordinate is exported by the NameCert surface. □

Theorem 8695.3 (FiniteTriggerFlipGraph edge budget)**.** *For every accepted* $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ *carrier* $G = (Q, L, E, B, H, C, P, N)$ *whose rows read the displayed finite inventory*

$$Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4,$$

the internal lift-flip part of B *records exactly* 397 *lift-flip edges, and every downstream support-graph or TasteGate consumer must read that budget through* Q, L, E, B, H, C, P, N*.*

Proof. Read the finite inventory through Q and L. The source theorem Definition 9462.1691 computes the lift-flip contribution by summing the hypercube edge counts over the displayed quotient triggers.

The budget row B records that sum as part of the carrier, while E records which edges are lift flips and which are quotient support flips. Transport and replay then keep the count attached to the same quotient and lift rows before provenance and local naming seal the packet.

Thus the public edge-budget surface is the finite route Q, L, E, B, H, C, P, N. It does not claim an unrecorded search, an external graph equality, or any stochastic interpretation of the adjacency relation. □

Falsifiable prediction: For the displayed finite inventory $Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4$, the edge-budget row B of any accepted $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ packet records 397 internal lift-flip edges. A future independent enumeration that changes this count must reject the packet or replace the displayed inventory.

Independence witness: minimal-trigger-lift-bundle, minimal-trigger-orbit-classification, codon-lift-orbit-classifier, boolean-trigger-lattice-beyond-mstar, coverage-rank-algebra-beyond-mstar: $\mathsf{FiniteTriggerFlipGra}$ is not bijective to any listed sibling because it jointly carries quotient-trigger rows, lift-choice fibers, support and lift flip edges, edge-budget rows, TasteGate-facing instance obligations, transport, replay, provenance, and local NameCert evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTriggerFlipGraph}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTriggerFlipGraph}^{\uparrow}$ packet is generated from quotient-trigger base rows, lift-choice fiber rows, support and lift flip-edge rows, edge-budget rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTriggerFlipGraph}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8695.1, Definition 8695.2, and Definition 8695.3 seed the finite trigger flip graph interface over quotient supports, lift fibers, edge rows, edge budgets, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTriggerFlipGraph}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, biological mechanism, stochastic transition semantics, unbounded graph search, TasteGate verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, quotient-trigger exposure, lift-fiber exposure, edge-row exactness, the 397 lift-flip edge budget, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTriggerFlipGraphUp.

Chapter 8696

A Concrete Naming Certificate for $\mathsf{CompleteMetricPicardIterator}^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive complete-metric analysis, the $\mathsf{CompleteMetricPicardIterator}^{\uparrow}$ packet records the finite Picard-iteration surface available before a complete-metric consumer reads a fixed-point approximation. It sits after the complete-metric carrier of chapter 128, the separated-reflector unit of chapter 8694, the separated-reflection route of chapter 4151, and the contraction interface of chapter 3529. The packet names a constructive orbit-and-tail ledger; it does not select a limit point, quotient representatives, host-level function equality, countable choice, or ambient classical completeness.

Definition 8696.1 (CompleteMetricPicardIterator carrier)**.** *A* $\mathsf{CompleteMetricPicardIterator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, S, T, Q, O, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, exposing point admission, metric readback, completeness witness data, and the separated target surface through which distances are read.*
2. *S is the separated-reflection row consumed before the iterator is allowed to treat zero-distance agreement as a public metric equality.*
3. *T is the contraction row, displaying the self-map, contraction budget, and compatibility with the metric row.*
4. *Q is the dyadic tail-control row, recording finite bounds that make the Picard tails Cauchy in the displayed complete metric.*
5. *O is the orbit row, listing the finite initial point, iterate window, and successor routing for the Picard chain.*
6. *W is the endpoint-window row. It records which finite windows may be exported to downstream fixed-point or Real-facing consumers before any completed endpoint is claimed.*
7. *H records componentwise* hsame *transport for the complete metric, separated reflection, contraction, tail-control, orbit, endpoint-window, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through M, S, T, Q, O, W, H before an endpoint-facing consumer reads the packet.*

9. *P is the* Pkg *provenance over* CompleteMetricPicardIterator$^{\uparrow}$, CompleteMetric$^{\uparrow}$, MetricSeparatedReflectorUnit$^{\uparrow}$, SeparatedReflection$^{\uparrow}$, BanachContraction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{CompleteMetricPicardIterator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, T, Q, O, W, H, C, P, N$. It has no coordinate for a chosen completed point, quotient representative, host equality of functions, countable-choice schedule, or classical fixed-point theorem.

Theorem 8696.2 (CompleteMetricPicardIterator NameCert obligations)**.** *For every accepted* CompleteMetricPicar *carrier $I = (M, S, T, Q, O, W, H, C, P, N)$, the local* NameCert$_{\text{CompleteMetricPicardIterator}^{\uparrow}}$ *surface consists exactly of complete-metric admission, separated-reflection exposure, contraction exposure, finite tail-control exposure, Picard-orbit exposure, endpoint-window exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every fixed-point or completion-facing consumer must read*

$$M, S, T, Q, O, W, H, C, P, N$$

before treating the Picard iterator as endpoint evidence.

Proof. Project the carrier of Definition 8696.1. The rows $M, S, T, Q, O, W, H, C, P, N$ are exactly the displayed coordinates available to the local certificate.

The complete-metric and separated-reflection rows are read first. They expose the metric request surface and the zero-distance discipline before the contraction row is allowed to generate an orbit.

Next read the contraction, tail-control, orbit, and endpoint-window rows. These rows supply the constructive Picard data: a self-map with a displayed contraction budget, finite tail estimates, finite iterate windows, and the bounded endpoint windows that downstream consumers may inspect.

Finally transport and replay preserve the same route. Componentwise hsame aligns each row, Cont replay sends the request through the displayed coordinates, and P, N source and name the packet. Since no carrier coordinate stores a selected limit, quotient representative, host function equality, countable-choice schedule, or ambient classical completeness principle, no NameCert obligation can export one. □

Closure status for NameCert$_{\text{CompleteMetricPicardIterator}^{\uparrow}}$.

Constructive story: A CompleteMetricPicardIterator$^{\uparrow}$ packet is generated from complete-metric data, separated-reflection exposure, a contraction row, finite tail-control data, Picard orbit windows, endpoint-window rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{CompleteMetricPicardIterator}^{\uparrow}}$, seed)

Scope closed: Definition 8696.1 and Definition 8696.2 seed the constructive Picard-iteration interface over complete metric data, separated reflection, contraction budgets, finite tails, orbit windows, endpoint windows, transport, replay, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\text{CompleteMetricPicardIterator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected fixed points, quotient representatives, host function equality, countable choice, ambient classical completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric exposure, separated-reflection exposure, contraction exposure, tail-control stability, orbit-window replay, endpoint-window routing, transport, provenance, and local naming for BEDC.Derived.CompleteMetricPicardIteratorUp.

Chapter 8697

A Concrete Naming Certificate for $\mathsf{CantorDiagonalTail}^{\uparrow}$

Following Cantor diagonal arguments and the finite-window discipline of Bishop–Bridges constructive analysis, the $\mathsf{CantorDiagonalTail}^{\uparrow}$ packet records the diagonal tail readback needed before a Real-facing consumer treats a stream of rational windows as a completed object. It sits between the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance spine of chapter 291, and the terminal seal of chapter 38. The packet records only finite diagonal indices, displayed tail windows, rational readback, dyadic tolerances, Real-facing terminal seals, transport, replay, provenance, and local naming; it does not export quotient stream equality, countable choice, selected limits, open-cover compactness, or ambient completeness of the reals.

Definition 8697.1 (Cantor diagonal tail carrier)**.** *A* $\mathsf{CantorDiagonalTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, T, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the diagonal index schedule, listing the finite diagonal coordinates whose tail reads are in scope.*
2. *T is the StreamName finite-window row, recording for each index in D the finite tail windows made visible to the packet.*
3. *Q is the RegSeqRat readback row, translating the displayed StreamName windows into regular rational comparison data.*
4. *E is the DyadicRatCore tolerance row, binding each readback in Q to a displayed dyadic tolerance and comparison budget.*
5. *R is the RealUp terminal seal row, recording only the Real-facing endpoint promised by the finite tail windows and dyadic tolerances.*
6. *H records componentwise* hsame *transport for the diagonal schedule, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RealUp seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real-facing query through D, T, Q, E, R, H before any consumer reads the diagonal tail.*

8. *P is the* Pkg *provenance over* $\mathsf{CantorDiagonalTail}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorDiagonalTail}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, Q, E, R, H, C, P, N$. It has no coordinate for a chosen infinite subsequence, quotient equality of streams, an unrecorded limiting real, an open-cover compactness argument, or a completeness principle outside the displayed finite rows.

Theorem 8697.2 (Cantor diagonal tail NameCert obligations)**.** *For every accepted* $\mathsf{CantorDiagonalTail}^{\uparrow}$ *carrier $K = (D, T, Q, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorDiagonalTail}^{\uparrow}}$ *surface consists exactly of diagonal index exposure, StreamName finite-window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance exposure, RealUp terminal-seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any Real-facing consumer of the packet must read the diagonal argument through those rows and cannot infer quotient stream equality, countable choice, selected limits, open-cover compactness, or ambient Real completeness from the certificate.*

Proof. Project the carrier of Definition 8697.1. The only rows exposed to a certificate consumer are $D, T, Q, E, R, H, C, P, N$, so every diagonal read begins with the finite schedule D and the displayed StreamName windows in T.

The readback row Q then turns only those windows into regular rational comparison data, and the tolerance row E binds each comparison to a displayed DyadicRatCore budget. The terminal row R is therefore reached through a finite route, not through a chosen infinite stream or an ambient completeness theorem.

Transport in H and replay in C preserve the same schedule, windows, rational readbacks, tolerance rows, and terminal seal before provenance and local naming are inspected. Thus the NameCert obligations are exactly the displayed finite fields, and the packet exports no quotient stream equality, countable-choice selection, selected limit, compactness, or Real-completeness principle. □

Theorem 8697.3 (Cantor diagonal tail L10 handoff)**.** *Let $K = (D, T, Q, E, R, H, C, P, N)$ be an accepted* $\mathsf{CantorDiagonalTail}^{\uparrow}$ *carrier. Every L10 read handed from K to a Real-facing proof factors through the finite path*

$$D \longrightarrow T \longrightarrow Q \longrightarrow E \longrightarrow R$$

and then through H, C, P, N. In particular, the handoff exposes only finite diagonal-tail windows and their regular rational and dyadic tolerance readbacks; it does not provide an equality of quotient streams, a countable family of choices, a selected limiting real, an open-cover compactness witness, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Fix an L10 read from the packet. By carrier projection, the read can mention no row outside $D, T, Q, E, R, H, C, P, N$. The schedule row D selects a finite list of diagonal coordinates, and the StreamName row T gives exactly the finite windows associated with that list.

The RegSeqRat row Q supplies the rational readback of those windows, while E supplies the DyadicRatCore tolerance data used by the Real-facing boundary. The RealUp row R records the terminal seal reached from those displayed inputs.

It remains to check that the handoff cannot smuggle a stronger principle through transport or provenance. Componentwise hsame in H transports only the displayed rows, Cont replay in C replays only the same finite path, and P, N add provenance and local naming without adding a new data coordinate. Hence every L10 handoff factors through the finite diagonal-tail route

and remains silent about quotient stream equality, countable choice, selected limits, open-cover compactness, and ambient Real completeness. □

Falsifiable prediction: Any accepted $\mathsf{CantorDiagonalTail}^{\uparrow}$ packet whose L10 read reaches a Real-facing proof without passing through the displayed finite route $D, T, Q, E, R, H, C, P, N$ is rejected by the local NameCert interface.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{CantorDiagonalTail}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a diagonal index schedule, finite StreamName tail windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RealUp terminal seals, transport, replay, provenance, and local NameCert evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorDiagonalTail}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorDiagonalTail}^{\uparrow}$ packet is generated from a finite diagonal schedule, StreamName tail windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RealUp terminal seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorDiagonalTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8697.1, Definition 8697.2, and Definition 8697.3 seed the finite diagonal-tail interface over schedules, windows, rational readback, dyadic tolerances, terminal seals, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorDiagonalTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, countable choice, selected limits, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal schedule exposure, StreamName window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance exposure, RealUp terminal-seal exposure, transport, replay, provenance, and local naming for BEDC.Derived.CantorDiagonalTailUp.

Chapter 8698

A Concrete Naming Certificate for $\mathsf{BanachPicardOrbit}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive fixed-point analysis, the $\mathsf{BanachPicardOrbit}^{\uparrow}$ packet records the finite orbit surface generated by a displayed contraction over complete metric data. It sits after the complete-metric carrier of chapter 128, the contraction packet of chapter 3529, the Lipschitz interface of chapter 410, the stream-name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet realizes a compact-uniform-continuity route through finite contraction, Cauchy-tail, complete-metric, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows without selecting a completed fixed point.

Definition 8698.1 (BanachPicardOrbit carrier)**.** *A* $\mathsf{BanachPicardOrbit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, K, L, S, R, E, T, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row carrying the metric source, separated readback, and completeness witness data.*
2. *K is the* $\mathsf{BanachContraction}^{\uparrow}$ *row naming the self-map and contraction budget.*
3. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row exposing the Lipschitz readback used by compact-uniform-modulus consumers.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *orbit-window row, recording finite observations of the Picard iterates.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same windows and tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after M, K, L, S, R and the tail data have been displayed.*
7. *T is the Cauchy-tail row recording finite contraction-tail estimates for the orbit.*
8. *W is the compact-uniform window row that records which finite orbit windows may be consumed by uniform-modulus and endpoint-facing readers.*
9. *H records componentwise* hsame *transport for the complete metric, contraction, Lipschitz, stream, regular-rational, Real-seal, tail, window, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a compact-uniform request through $M, K, L, S, R, E, T, W, H$ before any endpoint or modulus consumer reads the packet.*
11. *P is the* Pkg *provenance over* $\mathsf{BanachPicardOrbit}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachPicardOrbit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, L, S, R, E, T, W, H, C, P, N$. It has no coordinate for a selected fixed point, quotient orbit equality, host function equality, countable choice, or ambient classical completeness.

Theorem 8698.2 (BanachPicardOrbit NameCert obligations)**.** *For every accepted* $\mathsf{BanachPicardOrbit}^{\uparrow}$ *carrier $O = (M, K, L, S, R, E, T, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachPicardOrbit}^{\uparrow}}$ *surface consists exactly of complete-metric exposure, contraction exposure, Lipschitz readback, finite stream-window exposure, regular rational readback, Real-seal routing, Cauchy-tail exposure, compact-uniform window exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-uniform-continuity or constructive fixed-point consumer must read*

$$M, K, L, S, R, E, T, W, H, C, P, N$$

before treating the orbit as endpoint or modulus evidence.

Proof. Project the carrier of Definition 8698.1. The rows $M, K, L, S, R, E, T, W, H, C, P, N$ are exactly the displayed coordinates available to the local certificate.

The complete-metric, contraction, and Lipschitz rows expose the source of the orbit and its finite contraction budget. The stream and regular-rational rows then record the finite windows through which the iterates are observed.

The Real seal and tail row are read only through those windows. Thus an endpoint-facing consumer receives finite Cauchy-tail and readback data rather than a selected completed point.

Finally the compact-uniform window, transport, replay, provenance, and naming rows keep the same route visible for uniform-modulus consumers. Since the carrier lacks coordinates for selected fixed points, quotient orbit equality, host function equality, countable choice, or ambient classical completeness, no NameCert obligation can export those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachPicardOrbit}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachPicardOrbit}^{\uparrow}$ packet is generated from complete-metric data, Banach contraction data, Lipschitz readback, StreamName orbit windows, RegSeqRat readback, a Real seal, finite Cauchy-tail estimates, compact-uniform windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BanachPicardOrbit}^{\uparrow}}$, seed)

Scope closed: Definition 8698.1 and Definition 8698.2 seed the finite Picard-orbit interface over complete metric, contraction, Lipschitz, stream, regular-rational, Real-seal, Cauchy-tail, compact-uniform, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BanachPicardOrbit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected fixed points, quotient orbit equality, host function equality, countable choice, ambient classical completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric exposure, contraction exposure, Lipschitz readback, finite stream-window routing, regular rational readback, Real-seal routing, Cauchy-tail stability, compact-uniform replay, transport, provenance, and local naming for BEDC.Derived.BanachPicardOrbitUp.

Chapter 8699

A Concrete Naming Certificate for $\mathsf{CoareaFormula}^{\uparrow}$

Following Federer and Evans–Gariepy, the $\mathsf{CoareaFormula}^{\uparrow}$ packet records the finite BEDC surface by which a displayed Lipschitz or chartwise regular map may hand slice-count data to an integral consumer. It sits after the measure carrier of chapter 92, the finite integral route of chapter 123, the chart naming of chapter 96, the Real-facing seal of chapter 38, and the compact metric window of chapter 129. The packet is a finite coarea ledger over displayed chart, slice, Jacobian, and integral rows; it does not claim unrestricted Lebesgue theory, choice of slices, quotient measure classes, or ambient real completeness.

Definition 8699.1 (Coarea formula carrier)**.** *A* $\mathsf{CoareaFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{Coarea}(M, X, F, J, S, I, R, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row for the displayed domain window.*
2. *X is the* $\mathsf{Manifold}^{\uparrow}$ *chart row, naming only the finite chart windows used by the packet.*
3. *F is the displayed map row, storing the finite source, target, and Lipschitz or chartwise regularity certificates read by the packet.*
4. *J is the Jacobian-density ledger, recording the finite determinant or metric-volume rows exposed for the displayed chart windows.*
5. *S is the slice ledger; each entry names a displayed slice window, its admissibility row, and the finite preimage data attached to that slice.*
6. *I is the* $\mathsf{Integral}^{\uparrow}$ *readout row for the domain-side and slice-side finite integral requests.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison row for the displayed values produced by I.*
8. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *localization row that bounds the chart and slice windows used by X, F, J, S, I, R.*
9. *H records componentwise* hsame *transport for measure, chart, map, Jacobian, slice, integral, Real, localization, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a coarea request through the finite measure, chart, map, Jacobian, slice, integral, Real, and compact-metric rows.*

11. *P is the* Pkg *provenance over* $\mathsf{CoareaFormula}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CoareaFormula}^{\uparrow}}$ *row.*

Accepted packets are compared only through these rows. The classifier has no coordinate for unrestricted Lebesgue theory, a slice-choice function, quotient measure classes, a completed ambient real line, or an all-charts theorem outside the displayed finite windows.

Theorem 8699.2 (Coarea formula NameCert obligations)**.** *For every accepted* $\mathsf{CoareaFormula}^{\uparrow}$ *carrier* $\mathsf{Coarea}(M, X, F, J, S, I, R, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CoareaFormula}^{\uparrow}}$ *surface consists exactly of measure-source exposure through M, finite chart exposure through X, displayed map exposure through F, Jacobian-density exposure through J, slice-ledger exposure through S, Integral readout through I, Real comparison through R, compact-metric localization through K, componentwise transport through H, displayed replay through C, provenance nonescape through P, and local naming through N.*

Proof. Project the carrier of Definition 8699.1. The only analytic rows available before the structural rows are the measure source M, the chart row X, the displayed map row F, the Jacobian ledger J, the slice ledger S, the integral row I, the Real comparison row R, and the compact-metric localization row K.

The slice side is finite. Each slice row in S is tied to a displayed chart window and a displayed preimage entry, and the Jacobian row J is read only through the same chart windows. Thus a consumer cannot replace the ledger by a choice of slices or by a quotient class of measures.

The integral readout is also finite. The row I sends only the displayed domain-side and slice-side requests to $\mathsf{Integral}^{\uparrow}$, while R records the terminal $\mathsf{Real}^{\uparrow}$ comparisons reached from those requests and K keeps the chart windows localized inside $\mathsf{CompactMetric}^{\uparrow}$.

Finally H, C, P, N transport, replay, source, and locally name the same packet. Since no row stores unrestricted Lebesgue theory, a selected slice family, quotient measure classes, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8699.3 (Coarea formula slice handoff)**.** *Every measure, integration, or geometry consumer of an accepted* $\mathsf{CoareaFormula}^{\uparrow}$ *packet that asks for a slice calculation must first read the finite route*

$$\mathsf{Measure}^{\uparrow},\ \mathsf{Manifold}^{\uparrow},\ F,\ J,\ S,\ \mathsf{Integral}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow} \longmapsto H, C, P, N.$$

The consumer receives only displayed chart windows, map rows, Jacobian-density rows, slice-ledger rows, integral readouts, Real comparisons, compact-metric localization, transport, replay, provenance, and local naming. It receives no unrestricted Lebesgue theory, no choice of slices, no quotient measure class, and no ambient real-completeness theorem.

Proof. Apply Definition 8699.2. A consumer cannot ask for slice data until the measure source, chart window, displayed map, Jacobian ledger, slice ledger, integral readout, Real comparison, and compact-metric localization rows have all been exposed.

The route is local to the displayed windows. The chart row X and localization row K bound the windows, while F, J, S provide only the map, density, and slice records visible in those windows.

The handoff to integration then passes through I and R. These rows compare the displayed integral values through $\mathsf{Integral}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, but they do not add an all-slices selector, a quotient measure class, an unrestricted Lebesgue theory, or ambient Real completeness.

The structural rows H, C, P, N preserve that finite route for downstream readers. Therefore a later analysis or geometry packet may cite $\mathsf{CoareaFormula}^{\uparrow}$ as a slice handoff only at the finite NameCert surface described above. □

Falsifiable prediction: A purported $\mathsf{CoareaFormula}^{\uparrow}$ certificate fails if its slice calculation omits the displayed measure source, chart window, map row, Jacobian ledger, slice ledger, Integral readout, Real comparison, compact-metric localization, transport, replay, provenance, or local naming row.

Independence witness: measure, integral, manifold, real, compactmetric: $\mathsf{CoareaFormula}^{\uparrow}$ is not bijective with any listed sibling because it joins the measure source, finite chart windows, displayed map rows, Jacobian-density rows, slice ledgers, integral readouts, Real comparisons, compact-metric localization, transport, replay, provenance, and local naming as one slice-calculation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CoareaFormula}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CoareaFormula}^{\uparrow}$ packet is generated from MeasureUp source rows, finite ManifoldUp chart rows, displayed map rows, Jacobian-density rows, slice-ledger rows, IntegralUp readout, RealUp comparison, CompactMetricUp localization, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CoareaFormula}^{\uparrow}}$, seed)

Scope closed: Definition 8699.1, Definition 8699.2, and Definition 8699.3 seed the finite coarea slice handoff for measure, integral, and geometry consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CoareaFormula}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unrestricted Lebesgue theory, choice of slices, quotient measure classes, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked measure-source exposure, chart exposure, displayed-map exposure, Jacobian-density exposure, slice-ledger exposure, Integral readout, Real comparison, compact-metric localization, transport, replay, provenance, and local naming for BEDC.Derived.CoareaFormulaUp.

Chapter 8700

A Concrete Naming Certificate for $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$

Following Bishop–Bridges constructive metric topology, the $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ packet records the finite center-radius basis rows by which metric open balls are exposed to topology consumers. It sits over the metric source of chapter 4230, the dyadic positive-radius ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the indexed open-set surface of chapter 88. The packet is a finite basis object for Phase Real and Completion consumers; it does not import open-cover compactness, quotient equality, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8700.1 (Effective open-ball basis carrier)**.** *An* $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, C, D, W, R, E, T, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance comparison supplies the ball predicate.*
2. *C is the finite center schedule row, naming the displayed centers that may be used by basis consumers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger, carrying the requested radii, slack rows, and comparison budgets.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite observations of centers and requested points are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite real-name reads used in distance and radius comparison.*
7. *T is the* $\mathsf{Topology}^{\uparrow}$ *handoff row, exporting only the indexed open rows generated by displayed center-radius data.*
8. *H records componentwise* hsame *transport for the metric source, centers, dyadic radii, windows, readback, real seal, topology handoff, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay of the metric-ball-to-topology route.*
10. *P is the* Pkg *provenance over* $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveOpenBallBasis}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $M, C, D, W, R, E, T, H, K, P, N$. It has no coordinate for open-cover compactness, quotient equality of points, real trichotomy, countable choice, selected completions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.EffectiveOpenBallBasisUp`

Theorem 8700.2 (Effective open-ball basis NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ *carrier*

$$B = (M, C, D, W, R, E, T, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveOpenBallBasis}^{\uparrow}}$ *surface consists exactly of metric-source admission, finite center schedule exposure, dyadic positive-radius exactness, stream-window readback, regular rational replay, real-name sealing, topology handoff, componentwise transport, displayed continuation replay, provenance, and non-escape. In particular, a topology consumer may read an open-ball basis row only after the rows $M, C, D, W, R, E, T, H, K, P, N$ have been displayed in the same finite packet.*

Lean4 ✓ `BEDC.Derived.EffectiveOpenBallBasisUp.EffectiveOpenBallBasisTasteGate single carrier alignment`

Proof. Project the carrier of Definition 8700.1. The metric source M, center schedule C, and dyadic radius ledger D are the only rows that can generate a basis ball.

The readback rows W, R, E then fix how finite observations and real-name comparisons are exposed. They keep distance and radius comparison inside displayed stream windows, regular rational data, and the terminal real seal.

The topology handoff row T exports only the indexed open rows generated from those displayed center-radius data. Transport, replay, provenance, and local naming are carried by H, K, P, N for the same packet.

Since no row stores open-cover compactness, quotient equality, trichotomy, choice, selected completions, or ambient Real completeness, the listed coordinates exhaust the NameCert obligations. □

Theorem 8700.3 (Effective open-ball basis metric topology route)**.** *Every* $\mathsf{Topology}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ *packet must factor its open-set read through the displayed* $\mathsf{Met}^{\uparrow}$ *source row M, the center schedule C, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows W, R, the* $\mathsf{Real}^{\uparrow}$ *seal E, and the topology handoff row T. The route exports a finite center-radius basis surface and no open-cover compactness theorem, quotient point equality, real trichotomy, choice principle, or ambient completion of* $\mathsf{Real}^{\uparrow}$.

Proof. Start with the carrier projection. The rows M, C, D display the metric source, the available centers, and the positive radii before any topological read occurs.

Next read W, R, E. These rows make the finite distance and radius comparisons visible through stream windows, regular rational readback, and the Real seal, so the basis route is not a host-side predicate over hidden points.

Finally consume T together with H, K, P, N. The topology handoff is therefore the image of the displayed metric-ball rows under recorded transport and replay, and the consumer receives no additional compactness, quotient, trichotomy, choice, or ambient-completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ consumer must display a $\mathsf{Met}^{\uparrow}$ source row, a finite center schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{Topology}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row before an open-ball basis row can be consumed.

Independence witness: metricspace, dyadicratcore, streamname, regseqrat, real, topology: $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ is not bijective to any listed sibling because it binds metric centers and dyadic radii into a topology-facing basis handoff before any indexed open row becomes public.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveOpenBallBasis}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveOpenBallBasis}^{\uparrow}$ packet is generated from a MetricSpace source row, finite center schedule, DyadicRatCore positive-radius ledger, StreamName windows, RegSeqRat readback, Real seal, Topology handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveOpenBallBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8700.1, Definition 8700.2, and Definition 8700.3 seed the finite metric open-ball basis route from displayed center-radius data into indexed topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveOpenBallBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality, trichotomy, choice, selected completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger exactness, metric-ball readback, topology handoff, transport, replay, provenance, and non-escape for BEDC.Derived.EffectiveOpenBallBasisUp.

Chapter 8701

A Concrete Naming Certificate for $\mathsf{RadiusOfConvergence}^{\uparrow}$

Following Cauchy's power-series tests and Hadamard's formula for the radius of convergence, the $\mathsf{RadiusOfConvergence}^{\uparrow}$ packet records the finite BEDC data by which a power-series consumer may compare candidate radii before any Real-facing analytic claim is read. It sits after the finite power-series carrier of chapter 7045, the ratio-test route of chapter 2244, the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names only finite coefficient windows, displayed candidate-radius ledgers, convergence and divergence budgets, regular rational readback, Real sealing, transport, replay, provenance, and local naming; it does not assert a host supremum, analytic continuation, quotient real equality, trichotomy, selected limit, or ambient completeness.

Definition 8701.1 (Radius of convergence carrier)**.** *A* $\mathsf{RadiusOfConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, W, D, C, V, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *A is the coefficient source row, exposing the finite coefficient data consumed from the surrounding power-series packet.*
2. *W is the finite power-series window row, recording the displayed coefficient and exponent range in which the radius comparison is in scope.*
3. *D is the dyadic candidate-radius ledger, carrying candidate radii and comparison budgets through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic.*
4. *C is the convergence-side budget row, listing the finite tests that certify convergence reads for candidates inside the displayed ledger.*
5. *V is the divergence-side budget row, listing the finite obstruction reads for candidates outside the displayed ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same candidate-radius ledger and finite test windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after A, W, D, C, V, Q have been displayed.*
8. *H records componentwise* hsame *transport for coefficient, window, dyadic ledger, convergence budget, divergence budget, readback, seal, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from a radius query through A, W, D, C, V, Q, E, H before a consumer reads the packet.*

10. *P is the* Pkg *provenance over* $\mathsf{RadiusOfConvergence}^{\uparrow}$, $\mathsf{PowerSeries}^{\uparrow}$, $\mathsf{RealSeriesRatioTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RadiusOfConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, D, C, V, Q, E, H, T, P, N$. It has no coordinate for a host supremum over radii, analytic continuation, quotient equality of real names, trichotomy over the Real seal, selected limiting radius, or ambient completeness principle.

Theorem 8701.2 (Radius of convergence NameCert obligations)**.** *For every accepted* $\mathsf{RadiusOfConvergence}^{\uparrow}$ *carrier $R = (A, W, D, C, V, Q, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RadiusOfConvergence}^{\uparrow}}$ *surface consists exactly of coefficient-source exposure, finite power-series window exposure, dyadic candidate-radius ledger exposure, convergence-side budget exposure, divergence-side budget exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every power-series or analytic-function consumer must read*

$$A, W, D, C, V, Q, E, H, T, P, N$$

before treating the packet as radius evidence.

Proof. Project the carrier of Definition 8701.1. The rows $A, W, D, C, V, Q, E, H, T, P, N$ are exactly the displayed coordinates available to the local certificate.

The coefficient source and finite window rows fix the visible power-series data. The dyadic ledger then lists candidate radii, while the convergence and divergence budgets record only finite tests attached to that ledger.

Next read the regular rational and Real-facing rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row translates the displayed candidate-radius comparisons into regular readback data, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after those rows have been exposed.

Finally transport, replay, provenance, and local naming preserve the same finite route. Since the carrier lacks coordinates for a host supremum, analytic continuation, quotient real equality, trichotomy, selected limits, or ambient completeness, no NameCert obligation can export those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{RadiusOfConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RadiusOfConvergence}^{\uparrow}$ packet is generated from coefficient source data, finite power-series windows, dyadic candidate-radius ledgers, convergence-side and divergence-side budgets, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RadiusOfConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 8701.1 and Definition 8701.2 seed the finite radius-of-convergence interface over coefficient windows, dyadic candidate ledgers, convergence and divergence budgets, regular rational readback, Real sealing, transport, replay, provenance, and naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RadiusOfConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host suprema, analytic continuation, quotient real equality, trichotomy, selected limits, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient source exposure, finite window routing, dyadic candidate-radius ledger exposure, convergence and divergence budget routing, RegSeqRat readback, Real sealing, transport, provenance, and local naming for BEDC.Derived.RadiusOfConvergenceUp.

Chapter 8702

A Concrete Naming Certificate for $\mathsf{FrechetUrysohnSpace}^{\uparrow}$

Following Frechet, Urysohn, Kelley, and Engelking, the $\mathsf{FrechetUrysohnSpace}^{\uparrow}$ packet records the classical sequential closure criterion as finite BEDC naming data. It sits after the indexed-open topology surface of chapter 88, the sequence-indexed convergence interface of chapter 4112, and the finite observation rows of chapter 126. Completion-facing consumers read its numeric convergence rows through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal; the packet does not assert quotient sequence spaces, selected subsequences, countable choice, open-cover compactness, or ambient completeness.

Definition 8702.1 (Frechet–Urysohn space carrier)**.** *A* $\mathsf{FrechetUrysohnSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, S, W, R, E, A, H, C, P, N)$$

with the following rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying indexed open rows, closed-row vocabulary, point membership reads, and neighbourhood tests.*
2. *S is the* $\mathsf{SequentialSpace}^{\uparrow}$ *source row, exposing the sequence-indexed convergence interface over the same topology surface.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row for the sequence names used by a closure request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *convergence-readback row for numeric or completion-facing sequence observations.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after T, S, W, R have been displayed.*
6. *A is the sequential-closure admission row: for each displayed closed-row request and point-window request, it records the finite route by which a topological closure read may be witnessed by a displayed sequence window.*
7. *H records componentwise* hsame *transport for $T, S, W, R, E, A, C, P, N$.*
8. *C records displayed* Cont *replay of the route*

$$T, S, W, R \longmapsto E, A, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohnSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SequentialSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FrechetUrysohnSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, S, W, R, E, A, H, C, P, N$. It has no coordinate for quotient sequence equality, a host-selected subsequence, countable choice, arbitrary filters, open-cover compactness, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8702.2 (Frechet–Urysohn space NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohnSpace}^{\uparrow}$ *carrier*

$$F = (T, S, W, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetUrysohnSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission through T, sequential-source admission through S, finite stream-window exposure through W, regular rational convergence readback through R, terminal Real-seal ordering through E, sequential-closure admission through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 8702.1. The topology source, sequential source, stream windows, regular rational readback, Real seal, and closure-admission row are the only data visible before transport and provenance are read.

Read the closure request in dependency order. The topology row supplies the open and closed vocabulary; the sequential row supplies the sequence-indexed convergence interface; the stream and regular-rational rows expose finite windows and convergence readback before the terminal Real seal can be consumed.

Finally replay A, H, C, P, N. The admission row binds topological closure reads to displayed sequence windows, while transport, continuation, provenance, and local naming preserve the same route. Since no excluded quotient, selector, choice, filter, compactness, host equality, or completeness row is present, the listed rows exhaust the local certificate. □

Theorem 8702.3 (Frechet–Urysohn sequential-closure route)**.** *Every topological closure read exported by an accepted* $\mathsf{FrechetUrysohnSpace}^{\uparrow}$ *carrier must pass through*

$$T, S, W, R \longmapsto E, A, H, C, P, N.$$

Thus a $\mathsf{Topology}^{\uparrow}$ *consumer may use the packet as a sequential-closure route only after the* $\mathsf{SequentialSpace}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows have been displayed. The route is not a quotient sequence space, not a host subsequence selector, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with a closure-facing consumer. The carrier exposes only the topology row T for open and closed vocabulary, so the closure request cannot bypass the displayed topology source.

Next inspect S, W, R, E. The sequential-space row supplies the convergence interface, the stream row supplies finite observation windows, and the regular-rational row supplies the convergence readback used by completion-facing readers before the Real seal is reached.

Finally apply A, H, C, P, N. The sequential-closure admission row names the route from topological closure to sequence windows, and the structural rows preserve that route componentwise. Since the carrier has no coordinate for quotient sequence spaces, selected subsequences, countable choice, filters, compactness, or ambient Real completeness, none is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohnSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrechetUrysohnSpace}^{\uparrow}$ packet is generated from a Topology source row, a SequentialSpace source row, StreamName observation windows, RegSeqRat convergence readback, a Real seal, sequential-closure admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FrechetUrysohnSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8702.1, Definition 8702.2, and Definition 8702.3 seed the Frechet–Urysohn carrier over topology, sequential-space, StreamName, RegSeqRat, Real-seal, admission, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FrechetUrysohnSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient sequence spaces, selected subsequences, countable choice, arbitrary filters, open-cover compactness, host point equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-source routing, sequential-source routing, StreamName window coverage, RegSeqRat convergence readback, Real-seal ordering, sequential-closure admission, provenance, and local naming for BEDC.Derived.FrechetUrysohnSpaceUp.

Chapter 8703

Compact Uniform Barycentric Refinement NameCert Construction

Following Bishop and Bridges finite-cover refinement practice, the $\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}$ packet records the compact-uniform-continuity use of dyadic barycentric subdivision as finite NameCert data. It consumes the subdivision primitive of chapter 8605, the compact source rows of chapter 129, the tolerance-only modulus surface of chapter 369, the finite interval-cell net of chapter 5966, and the sequence/readback route of chapter 126, chapter 301, and chapter 38. The packet is a compact-uniform consumer handoff; it does not assert open-cover compactness, selected finite subcovers, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8703.1 (Compact uniform barycentric refinement carrier)**.** *A* $\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, K, M, I, S, R, E, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ *source row, exposing parent cells, barycenters, child cells, and mesh-decrease data.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row that supplies the compact finite-net boundary read by the uniform-continuity consumer.*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, recording the tolerance-only modulus produced after the compact source and subdivision rows are inspected.*
4. *I is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *interval-cell row used to align parent cells, child cells, and compact finite-net probes.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row for endpoint, barycenter, and finite-net probe reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for child-cell radii, compact-net tolerances, and modulus budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, K, M, I, S, R have been displayed.*
8. *H records componentwise* hsame *transport for all displayed refinement, compact, modulus, interval, window, readback, seal, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the compact-uniform route*

$$B, K, I, S, R \longmapsto M, E, H, P, N.$$

10. *P is the* Pkg *provenance over* CompactUniformBarycentricRefinement$^{\uparrow}$, DyadicBarycentricRefinement$^{\uparrow}$, CompactMetric$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CompactUniformBarycentricRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, M, I, S, R, E, H, C, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, host compactness, quotient stream equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8703.2 (Compact uniform barycentric refinement NameCert obligations)**.** *For every accepted* CompactUniformBarycentricRefinement$^{\uparrow}$ *carrier*

$$U = (B, K, M, I, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{CompactUniformBarycentricRefinement}^{\uparrow}}$ *surface consists exactly of barycentric-refinement source admission through B, compact-metric source admission through K, uniform-modulus handoff through M, dyadic interval-cell alignment through I, finite observation through S, regular rational readback through R, terminal Real-seal ordering through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8703.1. The accepted packet displays the dyadic barycentric source, compact metric source, uniform-modulus handoff, interval-cell alignment, observation windows, readback row, and Real seal before the structural rows are inspected.

Read the compact-uniform route in order. The barycentric row supplies parent and child cells, the compact row supplies finite-net source data, and the interval-cell row aligns the subdivision with the compact probes. The StreamName and RegSeqRat rows then expose finite endpoint windows and tolerance budgets before the terminal Real seal is available.

Finally replay M, H, C, P, N. The modulus row is a tolerance-only handoff generated inside the displayed route, while transport, continuation, provenance, and local naming preserve the same finite rows. Since no open-cover, selected-subcover, quotient, selected-limit, choice, host-compactness, or completeness coordinate is present, none is exported by the certificate. □

Theorem 8703.3 (Compact uniform barycentric refinement modulus route)**.** *Every compact-uniform-continuity consumer of an accepted* CompactUniformBarycentricRefinement$^{\uparrow}$ *packet must read*

$$B, K, I, S, R \longmapsto M, E, H, C, P, N.$$

Thus the UniformModulus$^{\uparrow}$ *row is available only after the* DyadicBarycentricRefinement$^{\uparrow}$, CompactMetric$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *rows have been displayed, and the* Real$^{\uparrow}$ *seal remains downstream of the same finite route.*

Proof. Begin with the barycentric and compact source rows. The row B exposes the parent cells, barycenters, child cells, and mesh-decrease data, while K supplies only the compact finite-net boundary used by the consumer.

Next align the source rows through I, S, R. The dyadic interval-cell row ties the subdivision cells to compact probes; the stream row exposes finite endpoint and barycenter windows; and the regular-rational row records the tolerance budgets consumed by the modulus handoff.

Only then may the consumer read M and E. The UniformModulus row is produced as a tolerance-only consumer row, and the terminal Real seal remains downstream of the displayed readback. The structural rows H, C, P, N preserve that order, excluding open-cover compactness, selected subcovers, quotient stream equality, selected limits, countable choice, and ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}$ packet is generated from a DyadicBarycentricRefinement source row, a CompactMetric source row, a UniformModulus handoff row, a DyadicIntervalNet alignment row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8703.1, Definition 8703.2, and Definition 8703.3 seed the compact-uniform barycentric refinement carrier over dyadic subdivision, compact source, uniform modulus, interval-cell, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformBarycentricRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, quotient stream equality, selected limits, countable choice, host compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked barycentric-refinement admission, compact-source admission, interval-cell alignment, StreamName window coverage, RegSeqRat tolerance readback, UniformModulus handoff, Real-seal ordering, provenance, and local naming for BEDC.Derived.CompactUniformBarycentricRefinementUp.

Chapter 8704

A Concrete Naming Certificate for $\mathsf{BishopContinuityTheorem}^{\uparrow}$

Following Bishop and Bridges, constructive continuity theorems are read here as finite modulus-producing routes over displayed names rather than as host statements about arbitrary functions. The $\mathsf{BishopContinuityTheorem}^{\uparrow}$ packet sits between the fan-continuity route of chapter 2027, the bar-induction continuity route of chapter 6028, the local modulus handoff of chapter 6998, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the theorem surface by which fan or bar continuity evidence is routed into a Bishop local modulus for Real and RegSeqRat consumers; it does not assert choice, weak limited principle of omniscience, quotient equality of real names, a selected host functional, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8704.1 (Bishop continuity theorem carrier)**.** *A* $\mathsf{BishopContinuityTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, B, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *route, when the theorem is read through a fan-continuity source.*
2. *B is the* $\mathsf{BarInductionContinuity}^{\uparrow}$ *route, when the theorem is read through a bar-induction source.*
3. *M is the* $\mathsf{BishopContinuityModulus}^{\uparrow}$ *handoff row produced by the selected displayed source route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and image names before the modulus is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those finite windows and the modulus-controlled tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after F or B, M, S, and R have been displayed.*
7. *H records componentwise* hsame *transport for the fan route, bar route, modulus row, window row, readback row, real seal, and local certificate row.*
8. *C records displayed* Cont *replay from the chosen continuity source through M, S, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopContinuityTheorem}^{\uparrow}$, $\mathsf{FanContinuityPrinciple}^{\uparrow}$, $\mathsf{BarInductionContinuity}^{\uparrow}$, $\mathsf{BishopContinuityModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopContinuityTheorem}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, B, M, S, R, E, H, C, P, N$ and the displayed choice of continuity source. It has no coordinate for a choice principle, weak limited principle of omniscience, quotient equality of real names, selected host functional, detached semantic continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8704.2 (Bishop continuity theorem NameCert obligations)**.** *For the carrier of Definition 8704.1, the local* $\mathsf{NameCert}_{\mathsf{BishopContinuityTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (F, B, M, S, R, E, H, C, P, N)$, continuity-source exposure through F or B, local modulus handoff through M, finite observation through S, regular rational readback through R, terminal real seal through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier to its displayed rows. The theorem source is either the fan-continuity row F or the bar-induction continuity row B, and no third continuity-source coordinate is available.

Read the modulus row next. The local handoff M is the $\mathsf{BishopContinuityModulus}^{\uparrow}$ surface consumed by the theorem, so the source route can affect downstream Real or RegSeqRat consumers only by producing that displayed modulus row.

Finally route the numeric observations through S, R, E. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row reads them back as regular rational data, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after those rows are displayed. The structural rows H, C, P, N transport, replay, source, and name the same finite packet, so the listed rows exhaust the NameCert obligations. □

Theorem 8704.3 (Bishop continuity theorem fan-modulus handoff)**.** *Whenever an accepted* BishopContinuityTheor *packet uses the* $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *source row F, its downstream continuity output factors through the* $\mathsf{BishopContinuityModulus}^{\uparrow}$ *row M before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer can read the theorem. The handoff is the finite route*

$$F \longmapsto M \longmapsto S \longmapsto R \longmapsto E,$$

with transport, replay, provenance, and local naming confined to H, C, P, N.

Proof. Start with the source row F in Definition 8704.1. It carries the fan-continuity evidence only as the displayed packet of chapter 2027.

Apply the NameCert obligations of Definition 8704.2. They require the theorem output to pass through M before the finite observation and readback rows S, R, E are visible.

The route is then fixed by the continuation row C. It replays the fan source into the local modulus, then into the stream windows, regular rational readback, and terminal real seal. Since H, C, P, N are the only structural rows, no choice selector, quotient equality, host functional, or completeness principle can enter the handoff. □

Theorem 8704.4 (Bishop continuity theorem nonescape boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopContinuityTheorem}^{\uparrow}$ *packet receives only the displayed continuity-source row F or B, the local modulus row M, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows H, C, P, N. Such a consumer cannot extract choice, weak limited principle of omniscience, quotient equality of real names, a selected host functional, detached semantic continuity, a general compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Read the consumer request through Definition 8704.2. The only numeric-facing route reaches R and E after one displayed continuity source and the local modulus row have already been exposed.

If the source is fan continuity, use Definition 8704.3. If the source is bar-induction continuity, the same carrier projection replaces F by B and still routes through M, S, R, E before the consumer can read any endpoint data.

The excluded principles are absent from the carrier inventory. Transport by H, replay by C, provenance by P, and local naming by N preserve the displayed finite rows; they do not create choice, WLPO, quotient equality, host functional selection, semantic continuity outside the packet, compactness, or real completeness. □

Falsifiable prediction: Any accepted BishopContinuityTheoremUp packet that exports a RegSeqRat or Real continuity read without displaying a FanContinuityPrinciple or BarInductionContinuity source, a BishopContinuityModulus row, StreamName windows, RegSeqRat readback, and Real seal fails the carrier classifier.

Independence witness: fancontinuityprinciple, barinductioncontinuity, bishopcontinuitymodulus, streamname, regseqrat, real: none is bijective with $\mathsf{BishopContinuityTheorem}^{\uparrow}$ because this carrier jointly records the theorem source, local modulus handoff, finite observation windows, regular rational readback, terminal real seal, transport, replay, provenance, and one local NameCert row while refusing choice, WLPO, quotient equality, host function selection, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopContinuityTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopContinuityTheorem}^{\uparrow}$ packet is generated from a FanContinuityPrinciple or BarInductionContinuity source row, a BishopContinuityModulus handoff row, StreamName windows, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopContinuityTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8704.1, Definition 8704.2, Definition 8704.3, Definition 8704.4, the falsifiable prediction, and the independence witness seed the constructive L10 continuity-theorem surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopContinuityTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice, WLPO, quotient equality of real names, selected host functionals, detached semantic continuity, general compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan and bar source exposure, BishopContinuityModulus handoff, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local NameCert rows for BEDC.Derived.BishopContinuityTheoremUp.

Chapter 8705

A Concrete Naming Certificate for $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$

Following Mac Lane on Kleisli categories and monads, and Bishop–Bridges/Kelley on metric completion, the $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$ packet records the finite service boundary where a completion-facing map is extended along the completion reflector monad without becoming a host function on completed representatives. It sits after the reflector monad of chapter 8693, the separated reflector unit of chapter 8694, and the real-number terminal seal of chapter 38. The packet names a Kleisli extension carrier, the bind and join compatibility rows that make the extension public, and the separated uniqueness row that prevents hidden quotient or choice data from entering the route.

Definition 8705.1 (Completion reflector Kleisli extension carrier)**.** *A* $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, U, K, J, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, exposing point admission, distance readback, zero-distance inspection, symmetry, triangle, and source transport before any completion-valued extension is read.*
2. *R is the completion reflector monad row, recording the completion object, reflector handoff, unit interface, bind interface, and join interface consumed from the* $\mathsf{CompletionReflectorMonad}^{\uparrow}$ *packet.*
3. *U is the separated unit row, read through* $\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}$*, which identifies exactly which zero-distance witnesses are allowed to pass from the source metric surface into the separated completion-facing target.*
4. *K is the Kleisli-extension schedule. It records the finite graph of inspected source names, the completion-valued target requests attached to them, and the order in which those requests are extended along the reflector unit.*
5. *J is the bind and join compatibility row, requiring the extension schedule K to pass through the monad bind and join rows of R before a nested completion request is flattened.*
6. *E is the separated uniqueness row. It states that two extension reads with the same S, R, U, K, J rows expose the same public route only through the separated comparison named by the reflector boundary.*

7. *H records componentwise* hsame *transport for the source metric row, reflector monad row, separated unit row, Kleisli schedule, bind and join compatibility, separated uniqueness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a source metric request through R, U, K, J, E, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$, $\mathsf{CompletionReflectorMonad}^{\uparrow}$, $\mathsf{MetricSeparatedReflectorUnit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected representatives, quotient completion, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8705.2 (Completion reflector Kleisli extension NameCert obligations)**.** *For every accepted* $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$ *carrier*

$$L = (S, R, U, K, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}}$ *surface consists exactly of source metric admission through S, completion reflector monad exposure through R, separated unit exposure through U, Kleisli extension scheduling through K, bind and join compatibility through J, separated uniqueness through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8705.1. The source metric information is confined to S, the completion reflector monad interface is confined to R, and the separated reflector unit information is confined to U.

The Kleisli extension route is not a host operation on completed objects. It is the finite schedule K together with the bind and join compatibility row J, so every completion-valued extension request must pass through the reflector monad rows named in chapter 8693.

Separated uniqueness is read only through E, with the separated unit boundary supplied by chapter 8694 and the real terminal seal supplied by chapter 38. The rows H, C, P, N then transport, replay, source, and name the same finite packet. Since no excluded principle has a row in L, the listed coordinates exhaust the local NameCert obligations. □

Theorem 8705.3 (Completion reflector Kleisli extension nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$ *carrier reads only the displayed finite route*

$$S, R, U, K, J, E, H, C, P, N.$$

In particular, the packet exports no selected representatives, no quotient completion, no host equality of completed objects, no countable choice, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8705.2. It identifies the public route as the source metric row, reflector monad row, separated unit row, Kleisli extension schedule, bind and join compatibility row, separated uniqueness row, transport row, replay row, provenance row, and local naming row.

The consumer can read completion-facing behavior only after those rows have been displayed. The reflector monad row supplies the bind and join surface, the separated unit row supplies the zero-distance boundary, and the Kleisli schedule records the finite extension order.

The carrier has no slot for representatives, quotient completion, host equality, choice, or ambient completeness. A request for any of those principles therefore remains outside the carrier instead of being replayed through H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}$ packet is generated from source metric rows, a completion reflector monad row, a separated unit row, a Kleisli extension schedule, bind and join compatibility, separated uniqueness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8705.1, Definition 8705.2, and Definition 8705.3 seed the finite Kleisli extension service boundary over source metric, completion reflector monad, separated unit, bind and join compatibility, separated uniqueness, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionReflectorKleisliExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected representatives, quotient completion, host equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Kleisli extension scheduling, bind and join compatibility, separated uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.CompletionReflectorKleisliExtensionUp.

Chapter 8706

A Concrete Naming Certificate for $\mathsf{BishopCauchyDiagonal}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ packet records the finite diagonal window used when a Cauchy family is read as a regular rational name and then sealed as a real-name boundary. It sits between the finite stream windows of chapter 126, the dyadic tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 Real and Completion object; it does not import host limits, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8706.1 (Bishop Cauchy diagonal carrier)**.** *A* $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, A, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy window row, recording the finite family of rational names and the inspected Cauchy bounds.*
2. *A is the diagonal schedule row, listing the finite index and precision pairs actually selected for the diagonal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the source window and diagonal schedule.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window readback generated by the displayed diagonal schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, exposing the regular rational readback only after S, A, D, W have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the regular name presented by the diagonal read.*
7. *H records componentwise* hsame *transport for the source window, diagonal schedule, dyadic ledger, StreamName window, RegSeqRat handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite diagonal request through $S, A, D, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyDiagonal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, D, W, R, E, H, C, P, N$. No coordinate stores a host limit, quotient stream equality, selected representative, countable-choice sequence, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 8706.2 (Bishop Cauchy diagonal NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ *carrier*

$$B = (S, A, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of source Cauchy window admission, finite diagonal schedule exposure, dyadic tolerance ledger visibility,* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A Real-facing consumer may read the diagonal packet only after all ten rows have been displayed.*

Proof. Project the carrier of Definition 8706.1. The displayed inventory is the finite source window S, diagonal schedule A, dyadic ledger D, StreamName readback W, regular rational handoff R, Real seal E, and the structural rows H, C, P, N.

The analytic rows are ordered. The schedule A is meaningful only against the source Cauchy window S; the dyadic ledger D attaches the requested tolerances to that finite schedule; and the StreamName row W records the actual window read from those displayed entries.

The completion-facing rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and terminal $\mathsf{Real}^{\uparrow}$ seal E may be consumed only after the finite window and dyadic ledger have been exposed.

Finally H, C, P, N preserve and name the same route. Since the carrier has no coordinate for host limits, quotient real equality, countable choice, trichotomy, or ambient Real completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 8706.3 (Bishop Cauchy diagonal Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ *packet must factor through the finite route*

$$S, A, D, W, R, E, H, C, P, N.$$

The route first displays the source Cauchy window and diagonal schedule, then reads the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. It exports a named finite diagonal packet, not a host limit, a quotient real representative, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligations of Definition 8706.2. They force a Real-facing read to expose $S, A, D, W, R, E, H, C, P, N$ as the accepted packet surface.

The finite diagonal schedule is read through the same completion corridor as the sibling chapters. The stream window agrees with the $\mathsf{StreamName}^{\uparrow}$ route of chapter 126, the precision ledger agrees with the $\mathsf{DyadicRatCore}^{\uparrow}$ route of chapter 291, and the regular rational handoff agrees with the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301.

Only after these rows are displayed may the terminal seal of chapter 38 be consumed. Thus the handoff remains a finite BEDC certificate for a diagonal read, with the excluded completeness, choice, quotient, and host-limit coordinates absent from the packet. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ consumer without displaying the source Cauchy window, diagonal schedule, dyadic tolerance ledger, StreamName window readback, RegSeqRat handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source Cauchy window, finite diagonal schedule, dyadic tolerance ledger, StreamName window readback, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming as one finite diagonal packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyDiagonal}^{\uparrow}$ packet is generated from a source Cauchy window, finite diagonal schedule, DyadicRatCore tolerance ledger, StreamName window readback, RegSeqRat handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8706.1, Definition 8706.2, and Definition 8706.3 seed the finite diagonal route from Cauchy windows to Real-facing regular rational seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limits, quotient real equality, selected representatives, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window admission, diagonal schedule exactness, dyadic tolerance visibility, StreamName readback, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyDiagonalUp.

Chapter 8707

A Concrete Naming Certificate for $\mathsf{RieszSchauderTheory}^{\uparrow}$

Following Riesz and Schauder on compact operators in normed spaces, the $\mathsf{RieszSchauderTheory}^{\uparrow}$ packet records the finite BEDC route by which a displayed compact linear operator exposes spectral-window and Fredholm-facing obligations. It reads the Banach endpoint rows of chapter 90, the bounded operator action of chapter 1644, the compact-operator certificate of chapter 1952, the spectral-radius boundary of chapter 2352, and the compact-perturbation alternative of chapter 3425. The packet names only this displayed compact-operator spectral route; it does not import a host spectrum, quotient operator algebra, selected eigenbasis, closed-range complement, or analytic Fredholm theorem.

Definition 8707.1 (Riesz–Schauder theory carrier)**.** *A* $\mathsf{RieszSchauderTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, B, K, S, F, H, C, P, N)$$

with the following displayed rows:

1. X *is the* $\mathsf{Banach}^{\uparrow}$ *source endpoint row.*
2. Y *is the* $\mathsf{Banach}^{\uparrow}$ *target endpoint row, or the same endpoint when the compact operator is read endomorphically.*
3. B *is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the displayed linear action and norm-bound ledger between the endpoints.*
4. K *is the* $\mathsf{CompactOperator}^{\uparrow}$ *row naming compactness by finite image-net schedules, tail moduli, and replay data.*
5. S *is the* $\mathsf{SpectralRadius}^{\uparrow}$ *window row used by spectral consumers that inspect finite power-growth and resolvent-test data.*
6. F *is the* $\mathsf{FredholmAlternative}^{\uparrow}$ *handoff row used by range-obstruction and kernel-facing consumers.*
7. H *records componentwise* hsame *transport for the endpoint, bounded-action, compactness, spectral-window, and Fredholm-handoff rows.*
8. C *records displayed* Cont *replay through the same rows.*
9. P *is the* Pkg *provenance over* $\mathsf{RieszSchauderTheory}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{FredholmAlternative}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RieszSchauderTheory}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a completed spectrum, quotient operator, selected eigenvector chain, algebraic multiplicity, closed-range complement, or host Fredholm determinant.

Lean4 (statement) `BEDC.Derived.RieszSchauderTheoryUp`

Theorem 8707.2 (Riesz–Schauder theory NameCert obligations)**.** *For an accepted* $\mathsf{RieszSchauderTheory}^{\uparrow}$ *carrier $R = (X, Y, B, K, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RieszSchauderTheory}^{\uparrow}}$ *surface consists exactly of endpoint admission through X, Y, bounded-action admission through B, compact-operator admission through K, spectral-window exposure through S, Fredholm-handoff exposure through F, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8707.1. The tuple $X, Y, B, K, S, F, H, C, P, N$ is the full visible inventory, so endpoint, action, compactness, spectral-window, Fredholm-handoff, transport, replay, provenance, and naming obligations cannot be enlarged by an external operator object.

Read the analytic dependencies in order. The Banach rows provide the endpoints, the bounded-operator row provides the displayed action, and the compact-operator row supplies the finite compactness witness before any spectral or Fredholm consumer can inspect the packet.

The spectral and Fredholm rows are then handoffs rather than ambient theorems. The row S points only to finite power-growth and resolvent windows from $\mathsf{SpectralRadius}^{\uparrow}$, while F points only to the kernel or range-obstruction boundary already named by $\mathsf{FredholmAlternative}^{\uparrow}$.

Finally H, C, P, N preserve, replay, source, and name those same rows. Since the carrier contains no completed spectrum, quotient operator, selected eigenbasis, multiplicity ledger, closed-range complement, or determinant coordinate, the listed obligations exhaust the local certificate surface. □

Theorem 8707.3 (Riesz–Schauder compact Fredholm route)**.** *Every consumer that reads a Riesz–Schauder spectral or Fredholm conclusion from an accepted* $\mathsf{RieszSchauderTheory}^{\uparrow}$ *packet must pass through the displayed route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{CompactOperator}^{\uparrow} \longmapsto \mathsf{SpectralRadius}^{\uparrow}, \mathsf{FredholmAlternative}^{\uparrow} \longmapsto \mathsf{RieszSchau}$$

The route exposes compactness through finite image-net schedules and exposes spectral or range-obstruction data only through the sibling windows S and F. It does not export a host spectrum, quotient operator algebra, selected eigenspace chain, algebraic multiplicity theorem, or analytic Fredholm determinant.

Proof. Begin with the NameCert obligations of Definition 8707.2. They force every accepted read to expose the Banach endpoints, bounded action, compactness row, spectral window, Fredholm handoff, and structural rows before any conclusion is public.

The compactness step is the central gate. A consumer cannot enter S or F until K has displayed the finite compact-operator schedules certified by chapter 1952. The bounded action B remains attached to the same endpoints throughout this replay.

Once the compact row is visible, the spectral side may inspect only the finite resolvent and power-growth windows of chapter 2352. The Fredholm side may inspect only the kernel-facing or range-obstruction handoff of chapter 3425.

Repack the route through H, C, P, N. These rows preserve and name the displayed packet, but they do not add a spectrum object, quotient operator algebra, selected eigenvectors, multiplicities,

complements, or determinants. Hence every exported Riesz–Schauder read factors through the finite compact-Fredholm route above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszSchauderTheory}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszSchauderTheory}^{\uparrow}$ packet is generated from Banach endpoint rows, a BoundedLinearOperatorUp action row, a CompactOperatorUp compactness row, SpectralRadiusUp finite spectral windows, a FredholmAlternativeUp kernel or range-obstruction handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszSchauderTheory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8707.1, Definition 8707.2, and Definition 8707.3 seed the compact-operator spectral route over BanachUp, BoundedLinearOperatorUp, CompactOperatorUp, SpectralRadiusUp, FredholmAlternativeUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszSchauderTheory}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close a host Riesz–Schauder theorem, completed spectrum, quotient operator algebra, eigenbasis selection, algebraic multiplicity, closed-range complement, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, bounded-action admission, compactness admission, spectral-window exposure, Fredholm-handoff exposure, transport, replay, provenance, and local naming for BEDC.Derived.RieszSchauderTheoryUp.

Chapter 8708

A Concrete Naming Certificate for $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$

Following Kelley, the $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ packet records the classical compact-open exponential law as finite NameCert data: a continuous map out of a product is compared with its curried continuous map into a compact-open function space only through displayed compact-open subbasic rows, continuity rows, Kelleyfication boundary rows, transport, replay, provenance, and local naming. It sits after the compact-open topology surface of chapter 4138, the Kelleyfication boundary of chapter 6830, and the continuous-map interface of chapter 130. The chapter names the finite topology-facing adjunction packet; it is not host cartesian closedness, a global mapping-space equality, selected compactly generated replacement, quotient function equality, countable choice, or Lean bridge verification.

Definition 8708.1 (Compact-open exponential law carrier)**.** *A* $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, Z, P, E, U, V, K, A, H, C, Q, N)$$

with the following displayed rows:

1. *X, Y, Z are the source, exponent, and target space rows inspected by the law.*
2. *P is the finite product-window row, recording the displayed reads of $X \times Y$ before any curried map is exposed.*
3. *E is the evaluation row, naming the comparison between a map $X \times Y \to Z$ and a map $X \to Z^Y$ through finite observations.*
4. *U is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *row, listing the compact-open subbasic windows used on the function-space side.*
5. *V is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording continuity for the uncurried and curried maps only through the public continuous-map interface.*
6. *K is the* $\mathsf{Kelleyfication}^{\uparrow}$ *boundary row, recording the compactly generated read when the compact-open topology must be sealed before the adjunction is public.*
7. *A is the adjunction comparison row, storing the two displayed directions of the compact-open exponential law.*

8. *H records componentwise* hsame *transport for spaces, product windows, evaluation, compact-open rows, continuous-map rows, Kelleyfication rows, adjunction rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from any law-facing read through $P, E, U, V, K, A, H, Q, N$ in that order.*
10. *Q is the* Pkg *provenance over* $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{Kelleyfication}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenExponentialLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, Z, P, E, U, V, K, A, H, C, Q, N$. It has no coordinate for host cartesian closedness, quotient function equality, selected compactly generated replacements, global mapping-space equality, countable choice, or bridge verification.

Theorem 8708.2 (Compact-open exponential law NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ *carrier*

$$L = (X, Y, Z, P, E, U, V, K, A, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOpenExponentialLaw}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the three displayed space rows, product-window row, evaluation row, compact-open row, continuous-map row, Kelleyfication boundary row, adjunction row, transport, replay, provenance, and local naming;*
2. *compact-open exposure through U, requiring every function-space observation to be read through the compact-open subbasic rows of chapter 4138;*
3. *continuous-map compatibility through V, requiring the uncurried and curried reads to use the public continuous-map interface of chapter 130;*
4. *Kelleyfication nonescape through K, requiring the compactly generated boundary of chapter 6830 before a topology-facing adjunction read is exported;*
5. *adjunction exactness through A, requiring both directions of the exponential-law comparison to preserve the same product, evaluation, compact-open, continuous-map, Kelleyfication, transport, replay, provenance, and local naming rows.*

No compact-open exponential-law obligation is stored outside these rows.

Proof. Project the carrier in Definition 8708.1. The accepted packet exposes exactly the rows $X, Y, Z, P, E, U, V, K, A, H, C, Q, N$, so carrier admission and local naming attach to those rows and to no hidden function-space object.

The compact-open obligation is read through U. Its subbasic windows are the only function-space topology coordinates available to the packet, and they are already routed through chapter 4138.

The continuity obligation is read through V. Both the uncurried and curried sides use the same public continuous-map interface, so the packet does not introduce host continuity or quotient equality of maps.

The Kelleyfication row K is the boundary before the adjunction becomes topology-facing. It records the compactly generated read supplied by chapter 6830 without selecting an external replacement.

Finally the adjunction row A compares the two displayed directions and the structural rows H, C, Q, N preserve that comparison under transport, replay, provenance, and local naming. Since the refused host endpoints have no carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8708.3 (Compact-open exponential law Kelley adjunction)**.** *Let* $L = (X, Y, Z, P, E, U, V, K, A, H, C, Q, N$ *be an accepted* $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ *carrier. Any law-facing read that exports the compact-open exponential comparison must first pass through the finite route*

$$P \longrightarrow E \longrightarrow U \longrightarrow V \longrightarrow K \longrightarrow A.$$

Thus the packet presents the Kelley-style adjunction only as a displayed compact-open and continuous-map route sealed by the Kelleyfication boundary; it exports no host cartesian-closed structure, global mapping-space equality, selected compactly generated replacement, quotient function equality, countable choice, or bridge credential.

Proof. Start with a law-facing consumer of the accepted carrier. By Definition 8708.2, the consumer can inspect the law only through the visible rows of L.

The route begins with the finite product and evaluation rows P and E. These rows display the uncurried input and the evaluation comparison before any function-space topology is available.

The compact-open and continuous-map rows come next. The row U supplies the subbasic topology observations, and V records that both sides of the comparison are read through the continuous-map interface.

The Kelleyfication boundary K then seals the topology-facing read. Only after that boundary is present may A expose the two directions of the compact-open exponential comparison.

The structural rows H, C, Q, N transport, replay, source, and name the same route. Because no host cartesian closedness, global mapping-space equality, selected replacement, quotient function equality, countable-choice schedule, or bridge credential appears in the carrier, the exported adjunction is exactly the displayed route. □

Falsifiable prediction: Every public $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ read factors through displayed product-window, evaluation, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Kelleyfication}^{\uparrow}$, adjunction, hsame, Cont, Pkg, and local NameCert rows. A public read that exports host cartesian closedness, global mapping-space equality, selected compactly generated replacement, quotient function equality, countable choice, or bridge verification from this packet refutes the carrier boundary.

Independence witness: compact-open-topology, kelleyfication, continuousmap: $\mathsf{CompactOpenExponentialLaw}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries product-window, evaluation, compact-open, continuous-map, Kelleyfication, adjunction, transport, replay, provenance, and local naming rows, while the siblings isolate topology generation, compactly generated sealing, or the continuous-map interface without the exponential-law comparison.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenExponentialLaw}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenExponentialLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8708.1 and Definition 8708.2 seed the finite compact-open exponential-law packet over product windows, evaluation, compact-open topology, continuous-map compatibility, Kelleyfication boundary, adjunction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenExponentialLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host cartesian closedness, global mapping-space equality, selected compactly generated replacement, quotient function equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open exposure, continuous-map compatibility, Kelleyfication nonescape, adjunction exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactOpenExp

Constructive story: A CompactOpenExponentialLaw$^{\uparrow}$ packet is generated from source, exponent, target, finite product-window, evaluation, CompactOpenTopology, ContinuousMap, Kelleyfication, adjunction comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 8709

A Concrete Naming Certificate for EilenbergVariety$^{\uparrow}$

Following Eilenberg's variety theorem for regular languages, the EilenbergVariety$^{\uparrow}$ packet records the finite NameCert surface where a family of regular-language rows and their syntactic-monoid classifiers are read together under boolean closure, quotient closure, and inverse-morphism closure. It sits beside the accepted-word packet of chapter 351, the right-congruence classifier of chapter 5729, the canonical deterministic presentation of chapter 6188, the finite classifier monoid of chapter 6302, and the finite-control surface of chapter 2364. The chapter names the displayed handoff between language families and finite syntactic monoid rows; it does not import a host powerset of all languages, arbitrary quotient languages, infinite alphabets, or an ambient pseudovariety theorem.

Definition 8709.1 (Eilenberg variety carrier)**.** *An* EilenbergVariety$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (A, L, M, C, Q, I, H, R, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet-family row, naming the displayed alphabets and finite alphabet morphisms available to the packet.*
2. *L is the regular-language family row, whose entries are accepted* RegularLanguage$^{\uparrow}$ *observations over the alphabets named by A.*
3. *M is the syntactic-monoid family row, assigning to each displayed language row its finite* SyntacticMonoid$^{\uparrow}$ *classifier.*
4. *C is the closure ledger for finite union, finite intersection, and complement inside the displayed language rows.*
5. *Q is the quotient ledger for displayed left and right word quotients, read through the finite right-congruence rows of* MyhillNerode$^{\uparrow}$ *and the canonical presentations of* MinimalDfa$^{\uparrow}$.
6. *I is the inverse-morphism ledger, recording how a displayed finite alphabet morphism pulls a displayed regular-language row back to another displayed regular-language row.*
7. *H records componentwise* hsame *transport for alphabet-family, regular-language, syntactic-monoid, boolean-closure, quotient, inverse-morphism, replay, provenance, and local naming rows.*

8. *R records displayed* Cont *replay from accepted-word observations through the Myhill–Nerode, minimal-DFA, finite-automaton, and syntactic-monoid handoff rows.*
9. *P is the* Pkg *provenance over* $\mathsf{EilenbergVariety}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, $\mathsf{MinimalDfa}^{\uparrow}$, $\mathsf{SyntacticMonoid}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EilenbergVariety}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, M, C, Q, I, H, R, P, N$. It has no coordinate for a host language powerset, arbitrary infinite alphabet family, selected quotient representatives outside the displayed ledgers, external monoid pseudovariety, or global star-free equivalence claim.

Theorem 8709.2 (Eilenberg variety NameCert obligations)**.** *For every accepted* $\mathsf{EilenbergVariety}^{\uparrow}$ *carrier $V = (A, L, M, C, Q, I, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EilenbergVariety}^{\uparrow}}$ *surface consists exactly of finite alphabet-family exposure, regular-language row exposure, syntactic-monoid classifier exposure, boolean-closure replay, left and right quotient replay, inverse-morphism replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8709.1. The rows A, L, M expose the finite alphabet family, the displayed regular-language observations, and the finite syntactic monoid classifiers attached to those observations.

Read C, Q, I next. The boolean ledger C records only finite union, finite intersection, and complement among displayed rows; the quotient ledger Q uses the Myhill–Nerode and minimal-DFA rows to name left and right quotient replay; the inverse-morphism ledger I follows displayed finite alphabet morphisms back to displayed language rows.

The structural rows H, R, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host language powerset, infinite alphabet family, external quotient representative, pseudovariety oracle, or global star-free theorem, the listed rows exhaust the NameCert obligations. □

Theorem 8709.3 (Eilenberg variety syntactic-monoid handoff)**.** *Every* $\mathsf{SyntacticMonoid}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, $\mathsf{MinimalDfa}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, *or* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer of an accepted* $\mathsf{EilenbergVariety}^{\uparrow}$ *packet first reads the displayed alphabet-family row, regular-language family row, syntactic-monoid family row, boolean-closure ledger, quotient ledger, and inverse-morphism ledger. The handoff through chapter 351, chapter 5729, chapter 6188, chapter 6302, and chapter 2364 therefore exports only the finite regular-language and classifier-monoid route.*

Proof. Start with an accepted packet $V = (A, L, M, C, Q, I, H, R, P, N)$. A regular-language consumer reads each language row from L only after the finite alphabet-family row A has fixed the displayed alphabet.

The Myhill–Nerode and minimal-DFA consumers then read the quotient ledger Q as a finite classifier replay boundary. The finite-automaton consumer receives only the displayed deterministic presentations already licensed by those right-congruence and accepted-word rows.

Finally, the syntactic-monoid consumer reads M as the finite classifier-monoid family attached to L, with C and I describing only displayed closure and inverse-morphism replay. Thus the packet supplies the Eilenberg handoff as finite BEDC rows, not as a host powerset of languages, a selected family of all quotients, or an external pseudovariety theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{EilenbergVariety}^{\uparrow}}$**.**

Constructive story: An EilenbergVariety$^{\uparrow}$ packet is generated from finite alphabet-family rows, regular-language family rows, syntactic-monoid classifier rows, boolean-closure ledgers, quotient ledgers, inverse-morphism ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{EilenbergVariety}\uparrow}$, seed)

Scope closed: Definition 8709.1, Definition 8709.2, and Definition 8709.3 seed the finite regular-language variety handoff across RegularLanguageUp, MyhillNerodeUp, MinimalDfaUp, SyntacticMonoidUp, and FiniteAutomatonUp rows.

Formal verification: FormalStatus(NameCert$_{\text{EilenbergVariety}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, a host variety theorem over all regular languages, star-free or aperiodic classification, external monoid pseudovarieties, infinite alphabet families, arbitrary quotient representatives, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-language family exposure, syntactic-monoid classifier exposure, boolean closure, quotient replay, inverse-morphism replay, sibling handoff, transport, replay, provenance, and local naming for BEDC.Derived.EilenbergVarietyUp.

Chapter 8710

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCompletion}^{\uparrow}$ packet records the base regular-completion boundary for rational names as finite BEDC NameCert data. It sits before the reflector route of chapter 6545, the functor handoff of chapter 7003, and the modulus refinement of chapter 8393; it reads dyadic tolerances from chapter 291, finite observation windows from chapter 126, regular rational readback from chapter 301, and the terminal real-name seal from chapter 38. The packet names the shared L10 completion surface refined by those siblings; it does not quotient Cauchy streams, select limits, use countable choice, assert host real equality, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8710.1 (Bishop regular completion carrier)**.** *A* $\mathsf{BishopRegularCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, listing the dyadic radii, comparison budgets, and triangle slack inspected by a regular completion request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing exactly the stream entries visible at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, S, R have all been displayed.*
5. *H records componentwise* hsame *transport for the dyadic ledger, finite windows, regular readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a precision request through*

$$D, S \longmapsto R \longmapsto E \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, E, H, C, P, N. It has no coordinate for quotient stream equality, a selected representative in a host completion, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8710.2 (Bishop regular completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCompletion}$ *carrier*

$$B = (D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *finite-window exposure, requiring every completion-facing request to inspect S before the readback row R is consumed;*
3. *dyadic tolerance exactness, requiring D to supply the same budget used by the stream-window comparison and the regular rational readback;*
4. *Real seal nonescape, requiring E to be read only after D, S, R have been displayed and replayed through H, C, P, N;*
5. *exclusion of quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8710.1. Its analytic rows are exactly D, S, R, E, with H, C, P, N supplying transport, replay, provenance, and local naming.

The source side begins with D and S. The dyadic ledger fixes the tolerance budget for the request, while the stream-window row fixes the finite observations allowed at that budget.

The regular readback row R is downstream of those source rows. Thus a completion-facing consumer receives regular rational data only through the displayed dyadic and stream evidence.

Finally, the terminal seal E is reached only after the same route has been replayed and named. Since the carrier contains no coordinate for quotient equality, a selected host limit, countable choice, trichotomy, host equality, or ambient completeness, those principles are outside the local certificate. □

Theorem 8710.3 (Bishop regular completion RegSeqRat seal)**.** *Every accepted* $\mathsf{BishopRegularCompletion}^{\uparrow}$ *packet reads its regular rational input through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}$$

displayed by D, S, R, H, C, P, N. The row R may be consumed as a regular rational readback for completion-facing arguments only after the dyadic tolerance ledger and finite stream window have been exposed.

Proof. Apply Definition 8710.2. The finite-window exposure clause places S before R, and the dyadic tolerance exactness clause places D before both the window comparison and the readback.

Rows H, C, P, N then transport, replay, source, and name the same route. They do not add a quotient of streams or a selected representative, so the RegSeqRat seal remains the displayed finite handoff from dyadic tolerance and stream-window data to regular rational readback. □

Theorem 8710.4 (Bishop regular completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCompletion}^{\uparrow}$ *carrier reads the route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows D, S, R, E, H, C, P, N. This handoff is the base boundary refined by chapter 6545, chapter 7003, and chapter 8393; it exports no quotient completion, selected limit, countable-choice functional, host real equality, trichotomy, or ambient completeness theorem.

Proof. Use Definition 8710.3. The Real-facing route already passes through dyadic tolerance, finite stream inspection, and regular rational readback.

The Real seal nonescape clause of Definition 8710.2 places E after those rows and after replay through H, C, P, N. The reflector, functor, and modulus chapters may refine this handoff by adding their own rows, but they still consume the same displayed base boundary.

Hence a Real-facing consumer receives only the finite Bishop regular completion packet. Any claim about quotient completions, selected limits, countable choice, host real equality, trichotomy, or ambient completeness is absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularCompletion}^{\uparrow}$ read exposes a DyadicRatCore tolerance ledger, StreamName finite-window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any reflector, functor, or modulus refinement is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-completion-reflector, bishop-regular-completion-functor, bishop-regular-completion-modulus: $\mathsf{BishopRegularCompletion}^{\uparrow}$ is not bijective to any listed sibling because it binds dyadic tolerance, finite windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as the shared base completion boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8710.1, Definition 8710.2, Definition 8710.3, and Definition 8710.4 seed the base Bishop regular completion boundary refined by the reflector, functor, and modulus siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected limits, countable choice, host real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, RegSeqRat readback, RealUp seal nonescape, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletionUp.

Chapter 8711

A Concrete Naming Certificate for CompactUniformModulusExtraction$^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the CompactUniformModulusExtraction$^{\uparrow}$ packet records the finite extraction route from compact nets and pointwise continuity data to a tolerance-only uniform-modulus readback. It sits over the compact metric source of chapter 60, the continuous-map local-radius surface of chapter 59, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is the concrete NameCert handoff for chapter 9237; it does not choose a global modulus from outside the displayed finite data.

Definition 8711.1 (Compact uniform modulus extraction carrier)**.** *A* CompactUniformModulusExtraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, L, D, F, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-net row, carrying the finite* ProbeBundle *centers, coverage witnesses, and compact-metric source data used by the extraction.*
2. *L is the* CFun$^{\uparrow}$ *local-radius row, recording the pointwise radius data and graph certificate consumed at each displayed center.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying the requested output tolerance, radius slack, and finite lower-bound comparisons.*
4. *F is the finite fold row over the compact net, retaining the minimum positive radius ledger obtained from the displayed centers.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the folded precision and the local regular-rational comparisons.*
6. *R is the terminal* Real$^{\uparrow}$ *seal row, available only after K, L, D, F, Q have all been displayed.*
7. *H records componentwise* hsame *transport for the compact-net, local-radius, dyadic, fold, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through the compact-net fold and into the uniform-modulus readback.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusExtraction}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, L, D, F, Q, R, H, C, P, N$. It has no coordinate for open-cover choice, selected external moduli, quotient equality of real names, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8711.2 (Compact uniform modulus extraction obligations)**.** *For every accepted* CompactUniformModulus *carrier $U = (K, L, D, F, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtraction}^{\uparrow}}$ *surface consists exactly of compact-net admission through K, local-radius exposure through L, dyadic tolerance and slack readback through D, finite lower-bound folding through F, regular rational readback through Q, Real sealing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. A uniform-modulus consumer therefore receives only the finite route*

$$K, L, D \longmapsto F \longmapsto Q \longmapsto R, H, C, P, N.$$

Proof. Project the carrier of Definition 8711.1. The compact-net row K supplies the finite center family and coverage witnesses, while L supplies the local continuous-map radius data attached to those centers.

Read the dyadic tolerance row D. It is the only tolerance budget available to the extraction, and the finite fold row F records the lower-bound ledger obtained by comparing the displayed positive radii.

The regular rational readback Q and terminal Real seal R are downstream of that fold. Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed packet.

Since no row stores open-cover choice, an external modulus selector, quotient equality, trichotomy, countable choice, or ambient Real completeness, the listed coordinates exhaust the NameCert obligations. □

Theorem 8711.3 (Compact uniform modulus Real-seal route)**.** *Every Real-facing read from an accepted* $\mathsf{CompactUniformModulusExtraction}^{\uparrow}$ *packet must expose the compact-net row K, continuous-map local-radius row L,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, finite lower-bound fold F, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q before the terminal* $\mathsf{Real}^{\uparrow}$ *seal R is consumed. The route exports a displayed uniform-modulus handoff and no open-cover choice principle, selected external modulus, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Apply the obligation theorem Definition 8711.2. It places K, L, D before the fold row F, and places F before the regular rational readback Q.

The Real seal R is therefore downstream of compact-net coverage, local continuity radii, dyadic tolerance comparison, and folded precision readback. A Real-facing consumer cannot inspect the terminal seal before those rows have been displayed in the same packet.

Finally replay H, C, P, N. These rows transport, replay, source, and name the finite route but do not add open-cover choice, external modulus selection, quotient equality, countable choice, or ambient completeness. The Real-facing handoff is exactly the displayed compact-net extraction route. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformModulusExtraction}^{\uparrow}$ consumer must display compact-net rows, ContinuousFunction local-radius rows, DyadicRatCore tolerance rows, finite lower-bound fold rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport,

displayed Cont replay, Pkg provenance, and a local NameCert row before a uniform-modulus readback is public.

Independence witness: compact, continuous, dyadicratcore, regseqrat, real: $\mathsf{CompactUniformModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it binds compact-net coverage, local continuous radii, dyadic tolerance comparison, finite lower-bound folding, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one extraction carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusExtraction}^{\uparrow}$ packet is generated from compact-net coverage, ContinuousFunction local-radius rows, DyadicRatCore tolerance comparison, finite lower-bound folding, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8711.1, Definition 8711.2, and Definition 8711.3 seed the finite extraction route from compact nets and local continuity radii to a uniform-modulus readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover choice, an external modulus selector, quotient equality of Real names, trichotomy, countable choice, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, local-radius exposure, dyadic tolerance exactness, finite fold stability, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusExtractionUp.

Chapter 8712

A Concrete Naming Certificate for $\mathsf{CauchyProductUniformCompletion}^{\uparrow}$

Following Bishop and Bridges on products of regular Cauchy names and uniform completions, the $\mathsf{CauchyProductUniformCompletion}^{\uparrow}$ packet records the finite route by which paired Cauchy-product rows are transported into the current L10 uniform-completion boundary. It sits after the product source of chapter 464, the regular product route of chapter 461, the product-completion fusion of chapter 7495, the uniform completion handoff of chapter 6282, and the Bishop regular completion boundary of chapter 8710. The packet exposes product-uniform completion transport without quotient pairs, selected limits, host product equality, or ambient completeness.

Definition 8712.1 (Cauchy product uniform completion carrier)**.** *A* $\mathsf{CauchyProductUniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (A, B, J, U, D, S, Q, R, H, C, G, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CauchyProduct}^{\uparrow}$ *source row carrying the paired Cauchy product data.*
2. *B is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *regularity row, recording the product-facing regular Cauchy witness before completion transport.*
3. *J is the* $\mathsf{CauchyProductCompletionFusion}^{\uparrow}$ *row that synchronizes product-completion and projection data.*
4. *U is the* $\mathsf{UniformCompletionReflector}^{\uparrow}$ *handoff row, read only after A, B, J have exposed the product route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the product and uniform-completion comparisons.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the paired product observations.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerance budget.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for completion-facing product reads.*
9. *H records componentwise* hsame *transport for the product, regularity, fusion, uniform-completion, tolerance, window, readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a product-completion request through $A, B, J, U, D, S, Q, R, H$.*

11. *G is the* Pkg *provenance over* CauchyProductUniformCompletion$^{\uparrow}$, CauchyProduct$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyProductCompletionFusion$^{\uparrow}$, UniformCompletionReflector$^{\uparrow}$, BishopRegularCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{CauchyProductUniformCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, J, U, D, S, Q, R, H, C, G, N$. It has no coordinate for quotient pair representatives, selected completion limits, host product equality, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8712.2 (Cauchy product uniform completion obligations)**.** *For every accepted* CauchyProductUniformCor *carrier $P = (A, B, J, U, D, S, Q, R, H, C, G, N)$, the local* NameCert$_{\text{CauchyProductUniformCompletion}^{\uparrow}}$ *surface consists exactly of Cauchy product admission through A, regular product exposure through B, product-completion fusion through J, uniform-completion handoff through U, dyadic tolerance readback through D, finite product windows through S, regular rational readback through Q, terminal Real sealing through R, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8712.1. The product rows A, B, J expose the Cauchy product source, regularity witness, and product-completion fusion before any uniform-completion handoff is visible.

Read U, D, S, Q next. The uniform-completion reflector row is tied to the same dyadic tolerance ledger, finite product windows, and regular rational readback, so the transport is not a host-side map between completed quotient pairs.

The terminal Real seal R is downstream of those displayed rows. Transport, replay, provenance, and local naming through H, C, G, N preserve the same finite packet.

Since the classifier has no coordinate for quotient pair representatives, selected completion limits, host product equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8712.3 (Cauchy product uniform-completion product route)**.** *Every completion-facing product read from an accepted* CauchyProductUniformCompletion$^{\uparrow}$ *carrier factors through the finite route*

CauchyProduct$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyProductCompletionFusion$^{\uparrow}$, UniformCompletionReflector$^{\uparrow}$, DyadicR

The route transports paired Cauchy-product rows to the uniform-completion handoff before the Real seal is public, and it exports no quotient pair, selected product limit, host product equality, countable-choice schedule, or ambient completeness theorem.

Proof. Use the obligation theorem Definition 8712.2. It places the Cauchy product row, regular product row, and product-completion fusion row before the uniform-completion handoff.

Apply the sibling completion boundary of chapter 8710. The dyadic tolerance, StreamName window, RegSeqRat readback, and Real-seal order used there is the same L10 order consumed by the product route.

Now read the product-completion fusion sibling of chapter 7495. It synchronizes product-completion and projection data before any component or product-facing completion consumer can inspect the terminal seal.

Finally replay H, C, G, N. These rows keep the uniform-completion transport attached to the displayed product rows, so the public read is the finite product-uniform route rather than a quotient pair, selected limit, host product equality, choice schedule, or ambient-completeness claim. □

Theorem 8712.4 (Cauchy product uniform completion sibling compatibility)**.** *The* CauchyProductUniformComplet *carrier is compatible with the finite-data product-window route of chapter 34 and the regular sequence common-window route of chapter 5965. Any sibling consumer must keep the product rows A, B, J, the uniform-completion row U, the tolerance and window rows D, S, Q, and the Real seal row R visible before it uses transport, replay, provenance, or local naming.*

Proof. Project the product route theorem Definition 8712.3. It already fixes the order from Cauchy product rows through uniform completion, dyadic tolerances, finite windows, RegSeqRat readback, and the terminal Real seal.

Read the product finite-data route through chapter 34. Its paired-window surface supplies product data only as displayed finite rows, not as hidden host tuples or selected projections.

Read the common-window completion route through chapter 5965. That route keeps StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, and Real seals public before completion-facing comparisons are consumed.

The rows H, C, G, N then transport and name the same packet. Hence every sibling consumer must preserve A, B, J, U, D, S, Q, R before structural replay, and no sibling can replace the route by quotient pairs, selected limits, host product equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted CauchyProductUniformCompletion$^{\uparrow}$ consumer must display CauchyProduct, RegularCauchyProduct, CauchyProductCompletionFusion, UniformCompletionReflector, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and local NameCert rows before a product-uniform-completion read is public.

Independence witness: cauchyproduct, regularcauchyproduct, cauchy-product-completion-fusion, uniform-completion-reflector, bishop-regular-completion, prod, regular-sequence-space: CauchyProductUniformCom is not bijective to any listed sibling because it binds paired Cauchy product rows, product-completion fusion, uniform-completion transport, dyadic tolerances, finite windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one product-uniform carrier.

Closure status for NameCert$_{\text{CauchyProductUniformCompletion}^{\uparrow}}$.

Constructive story: A CauchyProductUniformCompletion$^{\uparrow}$ packet is generated from CauchyProduct rows, RegularCauchyProduct regularity, CauchyProductCompletionFusion synchronization, UniformCompletionReflector handoff, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyProductUniformCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 8712.1, Definition 8712.2, Definition 8712.3, and Definition 8712.4 seed the finite product-uniform-completion route for L10 completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyProductUniformCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient pair representatives, selected product limits, host product equality, countable choice, trichotomy, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-row exposure, product-completion fusion, uniform-completion handoff, dyadic tolerance exactness, finite-window readback, RegSeqRat handoff, Real seal nonescape, sibling compatibility, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductUniformCompletionUp.

Chapter 8713

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name remains stable after a cofinal reindexing of its observation schedule. It sits over the regular rational sequence carrier of chapter 301, the finite stream observations of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name boundary of chapter 38, and the interleaving comparison route of chapter 414. The packet exposes cofinal schedule stability as finite NameCert data; it does not import quotient stream equality, selected subsequences, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8713.1 (Bishop regular Cauchy cofinal stability carrier)**.** *A* $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, K, D, W, I, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
2. *K is the cofinal schedule row, recording the finite monotone reindexing windows through which the same tail of R is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that fixes the dyadic budget before and after the cofinal schedule is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row opened along the schedule K.*
5. *I is the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *comparison row, displaying the interleaving route between the original schedule and the cofinal schedule.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, K, D, W, I have been exposed.*
7. *H records componentwise* hsame *transport for the source name, cofinal schedule, dyadic budget, finite window, interleaving route, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real-facing tolerance request through R, K, D, W, I and then into the terminal seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, K, D, W, I, E, H, C, P, N$. It has no coordinate for quotient equality of streams, a selected cofinal subsequence, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) BEDC.Derived.BishopRegularCauchyCofinalStabilityUp

Theorem 8713.2 (Bishop regular Cauchy cofinal NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$ *carrier $S = (R, K, D, W, I, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchy}}$ *surface consists exactly of carrier admission for the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row R; cofinal schedule exposure through K; dyadic tolerance exposure through D; finite stream-window exposure through W; interleaving comparison through I; Real-seal handoff through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier in Definition 8713.1. Its displayed packet lists exactly the rows $R, K, D, W, I, E, H, C, P, N$ so any accepted cofinal read begins with the regular rational name, the cofinal schedule, the dyadic budget, and the finite stream window.

The interleaving row gives the comparison surface. It records how the original regular Cauchy schedule and the cofinal schedule are read through the same dyadic tolerance ledger before either route reaches the terminal $\mathsf{Real}^{\uparrow}$ seal.

The support rows preserve that inventory. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming can only replay the displayed rows; since no quotient stream equality, selected subsequence, countable-choice schedule, host real equality, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 8713.3 (Bishop regular Cauchy cofinal Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$ *carrier $S = (R, K, D, W, I, E, H, C, P, N)$ factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BishopRegularCauchyCofinalStability}^{\uparrow} \longmapsto \mathsf{RegularCauchyInterleaving}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The read exposes the same source regular rational name, cofinal schedule, dyadic tolerance budget, finite StreamName observations, and interleaving route before the terminal Real seal is available. It exports no quotient stream equality, selected cofinal subsequence, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the rows R, K, D, W, I, E from the carrier. The source regular rational name R supplies the tail data, K scopes the cofinal reindexing schedule, D fixes the dyadic tolerance, and W exposes only finite observations of that schedule.

Next use the interleaving row I. It places the original and cofinal observations in one finite comparison route before the seal row E is exposed, so a Real-facing consumer cannot read a completed real independently of the displayed schedule.

Finally apply the structural rows H, C, P, N. Transport and replay preserve the same route, while provenance and local naming bind the packet as a $\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}$ certificate. Because the carrier lacks any quotient, choice-selected, host-equality, or completeness coordinate, the terminal $\mathsf{Real}^{\uparrow}$ seal cannot be reached by an escaping read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCofinalStability}^{\uparrow}}$.

Constructive story: A BishopRegularCauchyCofinalStability$^{\uparrow}$ packet is generated from a regular rational sequence row, a cofinal schedule row, a dyadic tolerance ledger, finite StreamName observations, a RegularCauchyInterleaving comparison row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopRegularCauchyCofinalStability}^{\uparrow}}$, seed)

Scope closed: Definition 8713.1, Definition 8713.2, and Definition 8713.3 seed the quotient-free cofinal reindexing route for regular Cauchy rational names.

Formal verification: FormalStatus(NameCert$_{\text{BishopRegularCauchyCofinalStability}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.BishopRegularCauchyCofinalStabilityUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected cofinal subsequences, countable-choice schedules, host real equality, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal schedule exposure, dyadic budget preservation, interleaving route stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauch

Chapter 8714

A Concrete Naming Certificate for $\mathsf{CauchyRootTest}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyRootTest}^{\uparrow}$ packet records a finite root-test convergence surface for Real-series and Completion-facing consumers. It sits beside the Real-series source of chapter 1581, the Cauchy-series criterion of chapter 2015, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the finite stream-window carrier of chapter 126, and the terminal real-name seal of chapter 38. The packet names a displayed root-bound discipline; it does not assert a selected sum, a host limit, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8714.1 (Cauchy root test carrier)**.** *A* $\mathsf{CauchyRootTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, Q, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row whose terms are read only through finite probes.*
2. *Q is the root-bound witness row, displaying a dyadic bound below the unit scale and an entry index beyond which the bound applies.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row that exposes the finite segment inspected by a precision request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the displayed root bound with the requested tail budget.*
5. *S is the* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *handoff row certifying that the finite root-bound data supplies the required Cauchy-series tail route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the source, root-bound, window, tolerance, and Cauchy-series rows have been exposed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the displayed regular rational readback row.*
8. *H records componentwise* hsame *transport for source, root-bound, window, tolerance, Cauchy-series, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a root-test request through $A, Q, W, D, S, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRootTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchySeriesCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRootTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for selected infinite sums, host completeness, quotient stream equality, countable-choice tail selectors, or detached limit objects.

Theorem 8714.2 (Cauchy root test NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRootTest}^{\uparrow}$ *carrier* $T = (A, Q, W, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRootTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure, root-bound witness exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8714.1. The displayed rows are exactly $A, Q, W, D, S, R, E, H, C, P, N$, so every local obligation must be read from one of those rows.

The convergence read begins with A, Q, W, D. The series source gives the inspected terms, the root-bound row gives the subunit dyadic bound and entry index, the stream row gives the finite tail window, and the dyadic ledger records the tolerance comparisons used by the request.

The handoff row S is then forced through $\mathsf{CauchySeriesCriterion}^{\uparrow}$. It consumes the same source, root-bound, window, and dyadic rows before any regular rational endpoint can be exposed.

Rows R and E terminate the route through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Transport, replay, provenance, and local naming preserve this order, and the carrier contains no row for selected sums, host limits, quotient streams, countable-choice selectors, or ambient completeness. Hence the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyRootTest}^{\uparrow}$ endpoint read factors through a RealSeries source row, root-bound witness, finite StreamName tail window, DyadicRatCore tolerance ledger, CauchySeriesCriterion handoff, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: realseries, cauchy-series-criterion, dyadicratcore, regseqrat, streamname, real: no listed sibling supplies the whole eleven-row packet $T = (A, Q, W, D, S, R, E, H, C, P, N)$, because each sibling contributes only the source series, Cauchy handoff, dyadic tolerance, regular readback, finite stream windows, or endpoint seal while this packet binds them into one finite root-test route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRootTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRootTest}^{\uparrow}$ packet is generated from a RealSeries source row, root-bound witness, finite StreamName tail window, DyadicRatCore tolerance ledger, CauchySeriesCriterion handoff, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRootTest}^{\uparrow}}$, seed)

Scope closed: Definition 8714.1 and Definition 8714.2 seed the finite root-test convergence route from displayed RealSeries source data through Cauchy-series handoff, regular rational readback, and Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRootTest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected sums, host limits, quotient stream equality, countable-choice tail selectors, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, root-bound exactness, stream-window admission, dyadic tolerance exactness, Cauchy-series handoff, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRootTestUp.

Chapter 8715

A Concrete Naming Certificate for HyperbolicIdealPentagon$^{\uparrow}$

The HyperbolicIdealPentagon$^{\uparrow}$ packet realizes the five-end boundary-cycle branch of chapter 9463 as finite NameCert data. It sits beside the Poincare-disk geodesic triangle packet of chapter 7891, the boundary-geodesic carrier of chapter 3491, the cross-ratio packet of chapter 3948, the ideal quadrilateral of chapter 7659, and the metric row of chapter 3474. The packet names five displayed boundary endpoints, cyclic side-geodesic incidence, hyperbolic metric comparison, cross-ratio readback, Real-seal rows, TasteGate-facing nontriviality and field-faithfulness requirements, transport, replay, provenance, and local naming; it does not complete an ideal boundary, choose global geodesic representatives, import a host Poincare model theorem, or identify observer expansion with physical spacetime.

Definition 8715.1 (Hyperbolic ideal pentagon carrier)**.** *A* HyperbolicIdealPentagon$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, B, S, M, X, Q, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the* PoincareDiskGeodesicTriangle$^{\uparrow}$ *comparison row used for finite triangular subreads inside the pentagon cycle.*
2. *B is the* HyperbolicGeodesicBoundary$^{\uparrow}$ *row naming five cyclically ordered boundary endpoints.*
3. *S records the five side-geodesic incidence rows and the finite adjacency ledger for the boundary cycle.*
4. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-comparison row over the displayed side and diagonal probes.*
5. *X is the* HyperbolicCrossRatio$^{\uparrow}$ *readback row for the boundary quadruples selected from adjacent pentagon windows.*
6. *Q is the* HyperbolicIdealQuadrilateral$^{\uparrow}$ *compatibility row recording the four-end subcycles used by the same pentagon data.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row for finite metric and cross-ratio comparisons.*
8. *E is the* Real$^{\uparrow}$ *seal reached only from the displayed regular rational readback row.*
9. *F records the TasteGate-facing requirements that the ambient scalar rows supply* Nontrivial *and* FieldFaithful *instances before the metric and cross-ratio comparisons are exported.*

10. *H records componentwise* hsame *transport for boundary endpoints, side incidence, metric comparison, cross-ratio readback, quadrilateral compatibility, readback, seal, TasteGate requirements, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a pentagon-cycle request through $T, B, S, M, X, Q, R, E, F, H, P, N$.*
12. *P is the* Pkg *provenance over* HyperbolicIdealPentagon$^{\uparrow}$, PoincareDiskGeodesicTriangle$^{\uparrow}$, HyperbolicGeodesicBound HyperbolicCrossRatio$^{\uparrow}$, HyperbolicIdealQuadrilateral$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Nontrivial, FieldFaithful, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{HyperbolicIdealPentagon}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed boundary quotients, selected geodesic representatives, host Poincare disk theorems, global analytic flow, or physical observer substance.

Theorem 8715.2 (Hyperbolic ideal pentagon NameCert obligations)**.** *For every accepted* HyperbolicIdealPentagon$^{\uparrow}$ *carrier $G = (T, B, S, M, X, Q, R, E, F, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicIdealPentagon}^{\uparrow}}$ *surface is exhausted by triangular subread admission, five-end boundary incidence, side-geodesic visibility, hyperbolic metric comparison, cross-ratio readback, ideal-quadrilateral compatibility, regular rational readback, Real seal non-escape, TasteGate-facing* Nontrivial *and* FieldFaithful *instance requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8715.1. The rows $T, B, S, M, X, Q, R, E, F, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The boundary-cycle read begins with T, B, S. The triangular row supplies the finite subread discipline, while the boundary row and side-incidence row expose exactly five cyclic endpoints and their adjacent geodesic sides.

Rows M, X, Q supply the geometric comparisons. The metric row reads only the displayed side and diagonal probes, the cross-ratio row reads boundary quadruples selected from finite adjacent windows, and the quadrilateral row checks compatibility with the four-end subcycle surface.

The endpoint readback then passes through R and E, and the TasteGate row F records that nontriviality and field-faithfulness are required before exporting the metric and cross-ratio comparisons. Transport, replay, provenance, and local naming preserve the same route. Since no row stores completed boundary quotients, selected geodesics, host Poincare theorems, global analytic flows, or physical observer substance, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Any accepted HyperbolicIdealPentagon$^{\uparrow}$ consumer that exports a five-end boundary-cycle comparison without the side-geodesic incidence row S, the metric row M, the cross-ratio row X, the Real seal E, and the TasteGate-facing row F refutes the carrier surface.

Independence witness: poincare-disk-geodesic-triangle, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, hyperbolic-ideal-quadrilateral, hyperbolic-metric: no listed sibling is bijective to HyperbolicIdealPentago because this packet alone binds five boundary endpoints, side-geodesic incidence, metric comparison, cross-ratio readback, ideal-quadrilateral compatibility, Real sealing, TasteGate-facing nontriviality and field-faithfulness requirements, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{HyperbolicIdealPentagon}^{\uparrow}}$.

Constructive story: A HyperbolicIdealPentagon$^{\uparrow}$ packet is generated from PoincareDiskGeodesicTriangle subreads, HyperbolicGeodesicBoundary five-end incidence, side-geodesic rows, HyperbolicMetric comparisons, HyperbolicCrossRatio readback, HyperbolicIdealQuadrilateral compatibility, RegSeqRat readback, RealUp seal, TasteGate-facing Nontrivial and FieldFaithful requirements, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicIdealPentagon}\uparrow}$, seed)

Scope closed: Definition 8715.1 and Definition 8715.2 seed the finite five-end boundary-cycle route over triangular subreads, boundary incidence, side geodesics, metric comparison, cross-ratio readback, quadrilateral compatibility, Real sealing, TasteGate requirements, transport, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicIdealPentagon}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed ideal boundaries, quotient endpoint equality, selected global geodesics, host Poincare disk theorems, physical observer dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked triangular subread admission, five-end boundary incidence, side-geodesic visibility, metric comparison, cross-ratio readback, quadrilateral compatibility, RegSeqRat readback, Real seal non-escape, TasteGate-facing instance requirements, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicIdealPentagonUp

Chapter 8716

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}$

Bishop and Bridges constructive completion practice reads a completed real endpoint through regular Cauchy rational data, finite observation windows, and displayed moduli rather than through quotient streams or selected limits. The $\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}$ packet records that selector surface as finite BEDC NameCert data over the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the equality route of chapter 1874. It names the handoff by which a completion-facing consumer can inspect a regular Cauchy input and its equality discipline without importing quotient stream equality, countable choice, a host limit selector, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8716.1 (Constructive Cauchy completion selector carrier)**.** *A* $\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, S, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational observations are to be sealed as a completion-facing real name.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the inspected source entries at the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii, slack comparisons, and modulus budgets used by the window read.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after R, S, D have been displayed.*
5. *Q is the* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *equality route used when two selector packets with displayed regular sources are compared.*
6. *H records componentwise* hsame *transport for the regular source, windows, dyadic ledger, Real seal, equality route, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through*

$$R, S, D \longrightarrow E \longrightarrow Q \longrightarrow H, P, N.$$

8. *P is the* Pkg *provenance over* ConstructiveCauchyCompletionSelector$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CauchyDifferenceCriterion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{ConstructiveCauchyCompletionSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, E, Q, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8716.2 (Constructive Cauchy completion selector NameCert obligations)**.** *For every accepted* ConstructiveCauchyCompletionSelector$^{\uparrow}$ *carrier $K = (R, S, D, E, Q, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCauchyCompletionSelector}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the* RegSeqRat$^{\uparrow}$ *source row,* StreamName$^{\uparrow}$ *finite-window row,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* Real$^{\uparrow}$ *seal,* CauchyDifferenceCriterion$^{\uparrow}$ *equality route, transport, replay, provenance, and local naming rows;*
2. *source-window exactness, requiring every completion-facing read to expose S as the finite inspection window for R;*
3. *tolerance-ledger exactness, requiring D to supply the dyadic budget consumed by the window read and the terminal seal;*
4. *Real-seal nonescape, requiring E to be read only after R, S, D have been displayed and replayed through H, C, P, N;*
5. *equality-route discipline, requiring any comparison of accepted selector packets to factor through the displayed* CauchyDifferenceCriterion$^{\uparrow}$ *row Q rather than through quotient streams or host real equality.*

No completion selector obligation is stored outside these rows.

Proof. Project the carrier of Definition 8716.1. The source row is R, its finite observations are S, its precision budget is D, and its terminal seal is E.

The source-window and tolerance clauses are row projections. A completion-facing consumer receives the source only through the displayed window and dyadic ledger, so no unbounded stream representative or selected limit can enter the local certificate.

The equality clause is supplied by the CauchyDifferenceCriterion$^{\uparrow}$ row Q. When two accepted selector packets are compared, Q routes the comparison through displayed regular sources, difference data, null evidence, zero-distance handoff, and the Real seal boundary.

Finally replay H, C, P, N. These structural rows preserve the displayed packet componentwise and record its provenance and local name; they do not add quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness. □

Theorem 8716.3 (Constructive Cauchy completion selector RegSeqRat–Real handoff)**.** *Every* Real$^{\uparrow}$ *consumer of an accepted* ConstructiveCauchyCompletionSelector$^{\uparrow}$ *carrier reads the completion-facing route*

$$\mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows R, S, D, E, H, C, P, N. If the consumer compares two such routes, it must additionally read the CauchyDifferenceCriterion$^{\uparrow}$ *row Q before treating the terminal seals as the same completion endpoint.*

Proof. Start with the NameCert obligations of Definition 8716.2. They place the finite window row S and dyadic tolerance ledger D before the terminal seal E, with R as the only regular rational source.

The handoff to $\mathsf{Real}^{\uparrow}$ is therefore the displayed route R, S, D, E. The transport, replay, provenance, and naming rows keep the same route visible to downstream Real and Completion consumers.

For equality-facing reads, apply the equality-route discipline from the same obligation theorem. The row Q is the only coordinate that compares accepted selector packets, and it is precisely the $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ route of chapter 1874. Thus the handoff reaches $\mathsf{Real}^{\uparrow}$ without quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}$ packet is generated from one RegSeqRat source row, finite StreamName windows, a DyadicRatCore tolerance ledger, one RealUp seal, one CauchyDifferenceCriterion equality route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8716.1, Definition 8716.2, and Definition 8716.3 seed the completion selector route from RegSeqRat through StreamName, DyadicRatCore, Real, CauchyDifferenceCriterion, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyCompletionSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, tolerance-ledger exactness, RealUp seal nonescape, equality-route discipline, replay, provenance, and local naming for BEDC.Derived.ConstructiveCauchyCompleti

Chapter 8717

A Concrete Naming Certificate for $\mathsf{HartmanGrobman}^{\uparrow}$

Following Hartman and Grobman on local hyperbolic dynamics, the $\mathsf{HartmanGrobman}^{\uparrow}$ packet records the finite BEDC surface by which a nonlinear local flow near a hyperbolic fixed point is compared with its linearized flow on a displayed conjugacy window. It sits after the ODE source of chapter 193, the local-flow endpoint packet of chapter 1899, the Gronwall comparison surface of chapter 2034, the metric carrier of chapter 57, and the terminal real-name seal of chapter 38. The packet names local linearization data only as finite NameCert rows; it is not a global conjugacy theorem, a selected stable manifold, a host solution operator, or a quotient equality of paths.

Definition 8717.1 (Hartman–Grobman carrier)**.** *A* $\mathsf{HartmanGrobman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, A, L, U, W, O, H, C, P, N)$$

with the following displayed rows:

1. *F is the nonlinear local-flow row, read from the displayed* $\mathsf{Ode}^{\uparrow}$ *source and the* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ *endpoint packet on one local time window.*
2. *A is the hyperbolic fixed-point row, recording the fixed point, the local chart basepoint, and the finite spectral separation ledger used by the comparison.*
3. *L is the linearized-flow row, displaying the derivative or linear comparison row on the same chart window.*
4. *U is the local chart row, carrying the metric neighbourhood, coordinate window, and real endpoint reads through* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows.*
5. *W is the conjugacy-window row, naming the finite domain on which nonlinear and linear orbit segments are compared.*
6. *O is the orbit-comparison ledger, with the* $\mathsf{GronwallInequality}^{\uparrow}$ *row and tolerance budget used to compare the nonlinear and linear segment reads.*
7. *H records componentwise* hsame *transport for nonlinear flow, hyperbolic fixed point, linearized flow, local chart, conjugacy window, orbit-comparison ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the ODE source through local flow, linearization, chart restriction, Gronwall comparison, and endpoint sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{HartmanGrobman}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HartmanGrobman}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, A, L, U, W, O, H, C, P, N$. It has no coordinate for global conjugacy, arbitrary invariant manifolds, selected stable or unstable foliations, host solution operators, quotient path equality, or ambient completeness.

Theorem 8717.2 (Hartman–Grobman NameCert obligations)**.** *For every accepted* $\mathsf{HartmanGrobman}^{\uparrow}$ *carrier $G = (F, A, L, U, W, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HartmanGrobman}^{\uparrow}}$ *surface consists exactly of carrier admission; nonlinear local-flow exposure through F; hyperbolic fixed-point exposure through A; linearized-flow exposure through L; local chart exposure through U; conjugacy-window exposure through W; orbit-comparison exactness through O; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8717.1. The rows $F, A, L, U, W, O, H, C, P, N$ are the complete finite inventory of an accepted packet.

The dynamic rows come first. The nonlinear local-flow row F and linearized-flow row L are read on the same chart row U, while the hyperbolic fixed-point row A supplies the finite separation ledger needed before the two flows can be compared.

The conjugacy-window and orbit-comparison obligations are then exactly W and O. The latter is the only place where Gronwall-style comparison and tolerance data enter the packet.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since the carrier has no coordinate for global conjugacy, arbitrary invariant manifolds, selected stable or unstable foliations, host solution operators, quotient path equality, or ambient completeness, none of those objects can become a local NameCert obligation. □

Theorem 8717.3 (Hartman–Grobman linearization window)**.** *Every accepted* $\mathsf{HartmanGrobman}^{\uparrow}$ *local linearization read factors through the finite route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{PicardLindelofLocalFlow}^{\uparrow} \longmapsto (A, L, U, W) \longmapsto \mathsf{GronwallInequality}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes the nonlinear local flow, hyperbolic fixed point, linearized flow, chart window, conjugacy window, orbit-comparison ledger, transport, replay, provenance, and local NameCert rows before any downstream stability consumer may treat nonlinear and linear orbit segments as locally conjugate.

Proof. Start with the ODE and local-flow rows. The nonlinear flow read is not a host solution operator; it is the displayed local endpoint route named by F.

Read the fixed-point and chart rows next. The hyperbolic separation ledger A and chart window U place both the nonlinear and linearized rows in the same finite coordinate neighbourhood.

The linearized-flow row L and conjugacy-window row W then delimit the comparison. The orbit-comparison ledger O sends the finite segment estimates through the Gronwall surface before the Real endpoint seal is exposed.

The structural rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming. Thus a stability consumer receives only the displayed local linearization window, not a global conjugacy, selected invariant manifold, quotient path equality, or ambient solution theory. □

Theorem 8717.4 (Hartman–Grobman hyperbolic scope)**.** *Any* $\mathsf{HartmanGrobman}^{\uparrow}$ *consumer that reaches a conjugacy-window row must first expose the hyperbolic fixed-point row* A*, the local chart row* U*, the linearized-flow row* L*, and the orbit-comparison ledger* O*. Consequently the packet supports only local hyperbolic scope: it names a finite comparison window between displayed nonlinear and linear orbit segments and exports no global dynamics, arbitrary invariant manifolds, host solution operators, quotient path equality, or selected stable-manifold data.*

Proof. Project an accepted carrier. The row W is not independent; it is a conjugacy-window coordinate inside the packet $G = (F, A, L, U, W, O, H, C, P, N)$.

The fixed-point and chart rows A, U bind the window to a hyperbolic local neighbourhood. The linearized-flow row L supplies the comparison target, and O supplies the orbit ledger through which the nonlinear and linear segments are compared.

After these rows are visible, H, C, P, N keep the packet stable under transport, replay, provenance, and local naming. No remaining coordinate can extend the read to global dynamics, arbitrary invariant manifolds, host solution operators, quotient path equality, or selected stable-manifold data. □

Falsifiable prediction: An accepted $\mathsf{HartmanGrobman}^{\uparrow}$ read cannot expose a conjugacy-window row without displaying the hyperbolic fixed-point row, local chart row, linearized-flow row, orbit-comparison ledger, transport row, replay row, provenance row, and local NameCert row.

Independence witness: ode, picard-lindelof-local-flow, gronwall-inequality, metric, real

Closure status for $\mathsf{NameCert}_{\mathsf{HartmanGrobman}^{\uparrow}}$.

Constructive story: A $\mathsf{HartmanGrobman}^{\uparrow}$ packet is generated from a nonlinear local-flow row, hyperbolic fixed-point row, linearized-flow row, local chart row, conjugacy-window row, orbit-comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HartmanGrobman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8717.1, Definition 8717.2, Definition 8717.3, and Definition 8717.4 seed the finite local hyperbolic linearization packet over nonlinear flow, fixed point, linearized flow, chart, conjugacy window, orbit comparison, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HartmanGrobman}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global dynamics, arbitrary invariant manifolds, selected stable or unstable manifolds, host solution operators, quotient path equality, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Formalization requires a Lean carrier and checked NameCert obligations for BEDC.Derived.HartmanGrobmanUp over local flow, hyperbolic fixed-point, linearized-flow, chart, conjugacy-window, orbit-comparison, transport, replay, provenance, and naming rows.

Chapter 8718

A Concrete Naming Certificate for $\mathsf{MetricPartitionOfUnity}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on metric covers and locally finite analytic gluing, the $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ packet records the finite BEDC route by which a metric-source cover is read as a locally finite subordinate partition certificate. It sits after the metric carrier of chapter 57, the topology carrier of chapter 88, the locally finite cover surface of chapter 7245, the paracompact refinement ledger of chapter 1963, and the partition surface of chapter 3369. The packet names only displayed metric-cover, support, finite-weight, normalization, transport, replay, provenance, and local naming rows; it does not select arbitrary host functions, choose subcovers, import open-cover compactness, or identify analytic functions extensionally.

Definition 8718.1 (Metric partition of unity carrier)**.** *A* $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, T, L, Q, S, W, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the displayed distance, ball, and metric-cover requests whose subordinate weights are being named.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *handoff row induced by the metric balls, recording the open-set interpretation used by every cover and support read.*
3. *L is the* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *row, recording the locally finite cover cells, finite neighbourhood witnesses, and assignment back to the metric-cover request.*
4. *Q is the* $\mathsf{Paracompact}^{\uparrow}$ *refinement ledger that keeps the locally finite cover route visible without selecting an arbitrary host subcover.*
5. *S is the subordinate support row, listing for each displayed cover cell the support window in which the corresponding weight row may be nonzero.*
6. *W is the finite weight row, listing the finitely many nonzero weights visible at each displayed metric neighbourhood read.*
7. *Z is the normalization row, recording that the displayed finite weights sum to the unit row on the covered metric region.*
8. *H is componentwise* hsame *transport for $M, T, L, Q, S, W, Z, C, P, N$.*

9. *C is displayed* Cont *replay from the metric-cover request through the locally finite cover, support, weight, and normalization rows.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricPartitionOfUnity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocallyFiniteCover}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricPartitionOfUnity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, L, Q, S, W, Z, H, C, P, N$. It has no coordinate for a host-selected function family, arbitrary subcover selection, open-cover compactness, unlogged support choice, or extensional equality of real-valued functions.

Theorem 8718.2 (Metric partition of unity NameCert obligations)**.** *For every accepted* $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ *carrier*

$$U = (M, T, L, Q, S, W, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricPartitionOfUnity}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, topology handoff, locally finite cover row, paracompact refinement ledger, subordinate support rows, finite weight rows, normalization row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every partition consumer must read the finite route*

$$M \longmapsto T \longmapsto L, Q \longmapsto S, W, Z \longmapsto H, C, P, N$$

before it treats subordinate weights as metric analytic data.

Proof. Project the carrier of Definition 8718.1. The accepted packet displays exactly the metric source, topology handoff, locally finite cover, paracompact refinement, support, weight, normalization, transport, replay, provenance, and local naming rows.

The metric source row M is read first, and T records the open-set interpretation of the displayed metric balls. The locally finite cover row L and paracompact refinement row Q then supply the cover cells and finite neighbourhood witnesses without adding a host subcover selector.

The subordinate support, weight, and normalization rows are S, W, Z. They expose support containment, the finite list of nonzero weights visible at a displayed metric neighbourhood, and the corresponding sum-to-one row on the covered region.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for arbitrary function selection, open-cover compactness, or extensional equality of functions, the listed rows exhaust the local NameCert surface. □

Theorem 8718.3 (Metric cover partition route)**.** *Every* $\mathsf{PartitionOfUnity}^{\uparrow}$ *or metric-analysis consumer of an accepted* $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ *packet must pass through the* $\mathsf{Met}^{\uparrow}$ *source row, the* $\mathsf{Topology}^{\uparrow}$ *handoff, the* $\mathsf{LocallyFiniteCover}^{\uparrow}$ *row, and the* $\mathsf{Paracompact}^{\uparrow}$ *refinement ledger before reading subordinate supports, finite weights, or normalization data. Thus the metric partition certificate is a finite cover route, not a host construction of real-valued functions.*

Proof. Read the dependency route in Definition 8718.2. It places the metric source M and topology handoff T before the locally finite and refinement rows L, Q.

The rows S, W, Z are downstream of that cover route. A support row is subordinate to a displayed cover cell, a weight row is visible only through the finite neighbourhood witnesses, and the normalization row sums only the displayed finite support.

The final rows H, C, P, N authenticate the same route. Therefore a consumer reaches the partition surface only through the metric and topological cover ledger already named by the packet, with no hidden function family or subcover selection. □

Falsifiable prediction: Every accepted $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ read exposes a MetricSpace source row, Topology handoff, LocallyFiniteCover row, Paracompact refinement ledger, subordinate support rows, finite weight rows, normalization row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any partition or metric-analysis consumer reads it.

Independence witness: metricspace, topology, locally-finite-cover, paracompact, partition-of-unity: $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a metric-source cover, topology handoff, locally finite cover cells, refinement ledger, subordinate support rows, finite weight rows, normalization, transport, replay, provenance, and local naming in one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricPartitionOfUnity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricPartitionOfUnity}^{\uparrow}$ packet is generated from a MetricSpace source row, Topology handoff, LocallyFiniteCover row, Paracompact refinement ledger, subordinate support rows, finite weight rows, normalization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricPartitionOfUnity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8718.1, Definition 8718.2, and Definition 8718.3 seed the finite metric-cover route from metric balls and topological opens to subordinate partition weights.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricPartitionOfUnity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-selected function families, arbitrary subcover selection, open-cover compactness, analytic function extensionality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exactness, topology handoff exactness, locally finite cover exactness, subordinate support exactness, finite weight normalization, transport, replay, provenance, and local naming for BEDC.Derived.MetricPartitionOfUnityUp.

Chapter 8719

Bishop Apartness Field Envelope NameCert Construction

Following Bishop and Bridges, the $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ packet records the finite envelope by which a regular real-name surface is admitted to constructive apartness-field use without turning the ambient Real object into a quotient field or a trichotomous ordered field. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, and the apartness-real interface of chapter 304. The envelope also exposes the neighboring apartness-budget discipline used by $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ consumers, but it names only the finite handoff rows visible to BEDC.

Definition 8719.1 (Bishop apartness field envelope carrier)**.** *A* $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, R, A, B, F, M, T, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name that later field-facing consumers inspect.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback route has been displayed.*
3. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-separation row used when nonzero or inequality evidence is consumed constructively.*
4. *B is the finite apartness-budget row, recording the dyadic or regular-Cauchy tolerance window required before A may be read.*
5. *F is the field-operation envelope row for addition, multiplication, negation, and reciprocal requests that already carry displayed apartness evidence for the reciprocal case.*
6. *M is the mixed handoff row joining regular names, Real seals, apartness evidence, and field-operation requests.*
7. *T records componentwise* hsame *transport for source names, Real seals, apartness evidence, budgets, field-operation rows, mixed handoff rows, continuation rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from a regular name and a positive apartness budget through R, A, B, F, M, T.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, A, B, F, M, T, C, P, N$. It has no coordinate for trichotomy, quotient fields, selected reciprocal representatives, host field equality, countable choice, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopApartnessFieldEnvelopeUp`

Theorem 8719.2 (Bishop apartness field envelope NameCert obligations)**.** *For every accepted* $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ *carrier $E = (Q, R, A, B, F, M, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopApartnessFieldEnvelope}}$ *surface consists exactly of carrier admission, regular rational source readback, Real seal nonescape, apartness-positive evidence, finite apartness-budget exactness, field-operation envelope admission, mixed handoff replay, componentwise transport, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 8719.1. The source row Q is read first, and the Real seal R remains downstream of the regular rational route.

The apartness row A is not a global order decision. It can be consumed only together with the finite budget B, so reciprocal and inequality-facing uses enter through displayed positive-separation evidence rather than trichotomy or host real equality.

The field-operation row F and mixed handoff row M collect only the operations whose inputs have already exposed the required regular and apartness rows. The structural rows T, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate is present, the certificate has exactly the stated obligations. □

Theorem 8719.3 (Bishop apartness field envelope regular real handoff)**.** *Every regular-real consumer of an accepted* $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$Q \longmapsto R \longmapsto M, T, C, P, N.$$

Thus Real and Completion consumers receive a regular rational source row and a finite replay route before the Real-facing seal is visible, and they receive no quotient-field equality, selected representative, countable-choice schedule, or full $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $E = (Q, R, A, B, F, M, T, C, P, N)$. The row Q is the only regular rational source coordinate, and R is the only Real seal coordinate.

The mixed handoff row M records how the Real-facing read is joined to the field and apartness envelope, while T and C transport and replay that same route. Provenance and local naming are carried by P and N.

Because no quotient-field object, selected representative, choice schedule, or completeness witness is a coordinate of the carrier, every regular-real handoff reaches the seal only through the displayed finite rows. □

Theorem 8719.4 (Bishop apartness field envelope apartness-field handoff)**.** *Every apartness-field use of an accepted* $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ *carrier must expose the route*

$$A, B, F \longmapsto M, T, C, P, N.$$

In particular, reciprocal-facing field requests are admitted only when the finite apartness-budget row has supplied positive-separation evidence. The envelope does not assert trichotomy, quotient fields, global decidable equality, selected inverses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The apartness-positive row A and finite budget row B are the only coordinates that license a nonzero or inequality read.

The field-operation row F can therefore be consumed only after that apartness evidence is displayed. The mixed handoff row M joins the operation request back to the regular and Real-name route, and T, C, P, N preserve the same finite evidence through transport, replay, provenance, and local naming.

The carrier contains no trichotomy coordinate, quotient-field row, global equality decision, selected inverse, or completeness witness. Apartness-field consumers therefore receive only the displayed constructive handoff. □

Falsifiable prediction: Every $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ consumer factors through one RegSeqRat source row, one RealUp seal row, one ApartnessReal positive-separation row, one finite apartness-budget row, one field-operation envelope row, one mixed handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, real, apartnessreal, regular-cauchy-apartness-budget: BishopApartnessFieldEnv is not bijective to any listed sibling because it jointly carries regular rational readback, Real sealing, apartness-positive evidence, finite budget data, field-operation admission, mixed handoff, transport, replay, provenance, and local naming in one envelope.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}$ packet is generated from a RegSeqRat source row, a RealUp seal row, an ApartnessReal positive-separation row, a finite apartness budget, a field-operation envelope, a mixed handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8719.1, Definition 8719.2, Definition 8719.3, and Definition 8719.4 seed the finite constructive bridge from regular real names to apartness-field consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopApartnessFieldEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient fields, global decidable equality, selected inverses, selected reciprocal representatives, host field equality, countable choice, full RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular rational source readback, Real seal nonescape, apartness-positive evidence, finite apartness-budget exactness, field-operation envelope admission, mixed handoff replay, transport, provenance, and local naming for BEDC.Derived.BishopApartnessFieldEnvelopeUp.

Chapter 8720

A Concrete Naming Certificate for RegularCauchySeparatedFilter$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki separation discipline for uniform completions, the RegularCauchySeparatedFilter$^{\uparrow}$ packet records the finite filter object that sits between regular Cauchy readback and separated completion. It is a neighbor of the CauchyFilterBasis$^{\uparrow}$ route of chapter 1501, the RegularCauchyFilterBasis$^{\uparrow}$ route of chapter 7853, and the separated-completion surface of chapter 1612. The packet keeps window schedules, rational readback, dyadic tolerances, basis observations, separated comparison, Real sealing, transport, replay, provenance, and local naming as displayed rows; it does not introduce quotient filters, selected limits, ambient real equality, countable choice, or host completeness.

Definition 8720.1 (RegularCauchySeparatedFilter carrier)**.** *A* RegularCauchySeparatedFilter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, B, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *finite-window schedule naming the regular Cauchy observations that may be read.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the same windows and regular-rational reads.*
4. *B is the* CauchyFilterBasis$^{\uparrow}$ *basis-window row whose observations are indexed by S and whose tolerances are supplied by D.*
5. *E is the* SeparatedCompletion$^{\uparrow}$ *comparison row, identifying when two displayed basis windows have zero separated distance inside the completion route.*
6. *L is the terminal* Real$^{\uparrow}$ *seal row reached only after S, R, D, B, E have been displayed.*
7. *H records componentwise* hsame *transport for windows, readback, tolerance, basis windows, separated comparison, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a filter-window request through regular readback, dyadic tolerance, basis exposure, separated comparison, and Real sealing.*
9. *P records* Pkg *provenance over* RegularCauchySeparatedFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, SeparatedCompletion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedFilter}^{\uparrow}}$ *row.*

The classifier accepts packets only through the ten displayed rows $S, R, D, B, E, L, H, C, P, N$. No coordinate stores a quotient filter, a hidden cofinal family, a selected completion point, host real equality, countable-choice selector, or ambient completeness witness.

Theorem 8720.2 (RegularCauchySeparatedFilter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySeparatedFilter}^{\uparrow}$ *carrier*

$$Q = (S, R, D, B, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, CauchyFilterBasis basis windows, SeparatedCompletion comparison, Real sealing, componentwise transport, displayed replay, package provenance, and local naming. Every completion-facing consumer must first read S, then R and D, then B and E, and only then the structural rows L, H, C, P, N.*

Proof. Project the carrier of Definition 8720.1. The visible source of the packet is the finite window schedule S together with its regular-rational readback R and dyadic tolerance ledger D.

Read the filter-basis and separation rows next. The row B contains exactly the basis observations licensed by the displayed windows, while E records the separated-completion comparison for the same observations. This is the handoff to the separated-completion route of chapter 1612.

Finally consume the Real seal and structural rows. The row L is downstream from S, R, D, B, E, and H, C, P, N only transport, replay, source, and name that same route. Since no excluded quotient, selector, host-equality, choice, or completeness coordinate is present, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySeparatedFilter}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, CauchyFilterBasis basis windows, a SeparatedCompletion comparison row, a RealUp seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8720.1 and Definition 8720.2 seed the finite regular Cauchy separated-filter route between basis windows and separated completion.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySeparatedFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-comparison preservation, readback preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySeparatedFilterUp.

Chapter 8721

A Concrete Naming Certificate for $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$

The $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ packet realizes the boundary cross-ratio phase branch of chapter 9463 as one finite NameCert surface. It sits after the hyperbolic metric packet of chapter 3474, the visual-boundary route of chapter 4181, the Busemann boundary route of chapter 3732, the boundary-shadow route of chapter 3813, and the Real seal of chapter 38. The packet records phase as finite boundary cross-ratio data with displayed metric, boundary, shadow, transport, replay, provenance, and local naming rows; it is not a selected ideal endpoint, quotient boundary equality, host Mobius geometry, or ambient real-valued phase theory.

Falsifiable prediction: An accepted $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ read is refuted if a consumer exports a boundary cross-ratio phase without first displaying HyperbolicMetric, visual-boundary, Busemann-boundary, boundary-shadow, Real-seal, transport, replay, provenance, and local NameCert rows.

Independence witness: hyperbolicmetric, hyperbolicvisualboundary, hyperbolicbusemannboundary, hyperbolicboundaryshadow, real: none is bijective to $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ because this packet jointly records metric scale, visual-boundary endpoints, Busemann calibration, shadow incidence, phase readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Definition 8721.1 (Hyperbolic cross-ratio phase carrier)**.** *A* $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (M, V, B, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, naming the finite metric-read surface used by the boundary comparison.*
2. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row, naming the displayed visual-boundary points visible to the packet.*
3. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, calibrating the boundary reads through Busemann data.*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, recording the shadow incidence attached to the same boundary data.*
5. *Q is the finite cross-ratio phase row, recording the ordered boundary quadruple and its phase readback.*

6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, V, B, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for metric, visual-boundary, Busemann, shadow, phase, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a boundary-phase request through metric readback, visual-boundary exposure, Busemann calibration, shadow incidence, phase readback, and Real sealing.*
9. *P records* Pkg *provenance over* $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, B, S, Q, R, H, C, P, N$. It has no coordinate for selected ideal endpoints, quotient boundary equality, host Mobius transformations, hidden phase normalization, or ambient real-valued geometry.

Theorem 8721.2 (Hyperbolic cross-ratio phase NameCert obligations)**.** *For the carrier of Definition 8721.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$X = (M, V, B, S, Q, R, H, C, P, N),$$

metric-read exposure through M, visual-boundary exposure through V, Busemann calibration through B, shadow incidence through S, cross-ratio phase readback through Q, Real sealing through R, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier of Definition 8721.1. Its visible rows are exactly $M, V, B, S, Q, R, H, C, P, N$.

The boundary comparison side is read through M, V, B, S, Q. The metric row fixes the finite scale surface, the visual-boundary row names the boundary entries, the Busemann row calibrates them, the shadow row records incidence, and the phase row reads the ordered cross-ratio datum.

The Real seal and structural rows are downstream. The row R seals the displayed phase readback, while H, C, P, N preserve, replay, source, and name the same route. Since the carrier has no coordinate for selected endpoints, quotient boundary equality, host Mobius transformations, hidden phase normalization, or ambient real-valued geometry, the listed rows exhaust the obligations. □

The Lean `TasteGate.lean` development for $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ must include explicit Nontrivial $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ instances. These instances must preserve the hyperbolic metric, visual-boundary, Busemann-boundary, boundary-shadow, cross-ratio phase, Real-seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}$ packet is generated from HyperbolicMetric rows, visual-boundary rows, Busemann-boundary calibration, boundary-shadow incidence, cross-ratio phase readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}}$, seed)

Scope closed: Definition 8721.1 and Definition 8721.2 seed the finite boundary cross-ratio phase route for hyperbolic observer-expansion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatioPhase}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected ideal endpoints, quotient boundary equality, host Mobius geometry, hidden phase normalization, or ambient real-valued geometry.

Upgrade path: Obligation closure requires checked carrier admission, metric exposure, visual-boundary exposure, Busemann calibration, shadow incidence, phase readback, Real-seal nonescape, nontriviality, and field faithfulness for BEDC.Derived.HyperbolicCrossRatioPhaseUp.

Chapter 8722

A Concrete Naming Certificate for $\mathsf{FiniteLipschitzDifferentiability}^{\uparrow}$

Following Rademacher's differentiability theorem for Lipschitz maps, the $\mathsf{FiniteLipschitzDifferentiability}^{\uparrow}$ packet records the BEDC-finite interface at which derivative-candidate evidence may be consumed without importing a measure-theoretic almost-everywhere theorem. It refines the finite derivative-candidate route of chapter 5146 by reading the Lipschitz graph and bound of chapter 3762, the Real base and output seals of chapter 38, the dyadic radius ledger of chapter 291, the metric-neighbourhood rows of chapter 57, and the finite variation control of chapter 3361. The packet is a named finite-window differentiability interface; it is not a Lebesgue a.e. differentiability theorem and it selects no derivative outside the displayed windows.

Definition 8722.1 (Finite Lipschitz differentiability carrier)**.** *A* $\mathsf{FiniteLipschitzDifferentiability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, E, D, M, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LipschitzFunction}^{\uparrow}$ *graph-and-bound row, exposing the source metric, target metric, graph ledger, Lipschitz constant, and finite distance-inequality route used by the derivative-candidate read.*
2. *E is the pair of* $\mathsf{Real}^{\uparrow}$ *base and output seal rows, recording the base-point seal, endpoint output seals, and finite difference-quotient values visible to Real-facing consumers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and difference-quotient ledger, listing the inspected dyadic radii, endpoint offsets, halved-window refinements, tolerance comparisons, and displayed quotient windows.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *neighbourhood row for the source and target distance requests used to read the Lipschitz and quotient comparisons.*
5. *A is the* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *control row, bounding only the displayed finite variation window consumed by the derivative-candidate route.*
6. *H records componentwise* hsame *transport for the Lipschitz row, Real seals, dyadic radius ledger, metric neighbourhoods, absolute-continuity control, replay row, provenance row, and local naming row.*

7. *C records displayed* Cont *replay from a finite differentiability request through D, M, L, A and then into the terminal Real seal row E.*
8. *P is the* Pkg *provenance over* FiniteLipschitzDifferentiability$^{\uparrow}$, Rademacher$^{\uparrow}$, LipschitzFunction$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Met$^{\uparrow}$, AbsolutelyContinuousFunction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{FiniteLipschitzDifferentiability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, E, D, M, A, H, C, P, N$. It has no coordinate for measurable exceptional sets, selected derivatives outside the finite windows, Lebesgue measure algebra, quotient equality, countable choice, host real equality, or a completed derivative space.

Theorem 8722.2 (Finite Lipschitz differentiability NameCert obligations)**.** *For every accepted* FiniteLipschitzDifferentiability$^{\uparrow}$ *carrier*

$$F = (L, E, D, M, A, H, C, P, N),$$

the local NameCert$_{\text{FiniteLipschitzDifferentiability}^{\uparrow}}$ *surface consists exactly of Lipschitz graph-and-bound exposure, Real base and output seal routing, dyadic radius and difference-quotient accounting, metric-neighbourhood inspection, absolute-continuity finite-window control, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from the displayed rows.*

Consequently the packet excludes measurable exceptional sets, selected derivatives outside the finite windows, Lebesgue measure algebra, quotient equality, countable choice, host real equality, and completed derivative-space data.

Lean4 (statement) `BEDC.Derived.FiniteLipschitzDifferentiabilityUp`

Proof. Carrier exposure is projection from Definition 8722.1. The visible analytic rows are L, E, D, M, A, and the visible structural rows are H, C, P, N.

The differentiability read is finite. The Lipschitz row supplies the graph and bound, the dyadic ledger supplies the inspected radii and quotient windows, and the metric row supplies the neighbourhoods in which those finite requests are compared. The absolute-continuity row bounds only the displayed variation window attached to those requests.

The Real row seals the base, endpoint, and quotient values reached by that finite route. Transport and replay preserve the same coordinates, while provenance and local naming source the packet without adding analytic rows.

Since no other coordinate occurs in the carrier, the NameCert surface exports none of the excluded measurable, derivative-selection, quotient, choice, host-equality, or completion data. □

Closure status for NameCert$_{\text{FiniteLipschitzDifferentiability}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FiniteLipschitzDifferentiability}^{\uparrow}}$, seed)

Scope closed: Definition 8722.1 and Definition 8722.2 seed the finite Lipschitz differentiability interface over Lipschitz graph-and-bound, Real seal, dyadic radius, metric-neighbourhood, absolute-continuity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteLipschitzDifferentiability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lebesgue almost-everywhere differentiability, measurable exceptional sets, selected derivatives outside finite windows, Lebesgue measure algebra, quotient equality, countable choice, host real equality, completed derivative spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz graph-and-bound exposure, Real seal routing, dyadic radius and quotient accounting, metric-neighbourhood inspection, absolute-continuity finite-window control, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.FiniteLipschitzDifferentiabilityUp.

Constructive story: A FiniteLipschitzDifferentiability$^{\uparrow}$ packet is generated from a LipschitzFunction row, RealUp base and output seals, DyadicRatCore radius and difference-quotient ledger, MetricSpace neighbourhood row, AbsolutelyContinuousFunction finite-window control row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8723

A Concrete Naming Certificate for $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy convergence, the $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ packet records the finite tail-selection budget through which a regular Cauchy stream exposes cofinal tail data before any real-name consumer may read it. It sits after the finite stream naming surface of chapter 126, the dyadic tolerance carrier of chapter 291, the regular rational sequence route of chapter 301, the real-name seal of chapter 38, and the Cauchy tail-equivalence boundary of chapter 1738. The packet names a choice-free cofinal selector budget; it does not select limits, quotient streams, countable-choice subsequences, trichotomy witnesses, or ambient Real completeness.

Definition 8723.1 (Cofinal tail-selector budget carrier)**.** *A* $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, exposing the finite stream-name windows on which tail requests are made.*
2. *T is the cofinal tail-selector budget row, recording a finite budget for moving from an inspected window to a later tail window without choosing a global subsequence.*
3. *W is the finite window row, listing the source indices and later indices visible to the local selector budget.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision request used by the tail comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row through which regular rational sequence consumers read the selected tail window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, available only after the StreamName, dyadic tolerance, and RegSeqRat rows have been exposed.*
7. *H records componentwise* hsame *transport for the stream source, selector budget, windows, tolerance, handoff, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same selector route.*

9. *P is the* Pkg *provenance over* $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailSelectorBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, W, D, R, E, H, C, P, N$. It has no coordinate for a selected limit, quotient-stream equality, an infinite choice function, trichotomy, or completeness of an ambient real line.

Theorem 8723.2 (Cofinal tail-selector budget NameCert obligations)**.** *For every accepted* CofinalTailSelectorBudge *carrier $B = (S, T, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalTailSelectorBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName source, cofinal selector budget, finite window, dyadic tolerance, RegSeqRat handoff, Real seal, componentwise transport, displayed replay, provenance, and local naming; cofinal selector exposure through S, T, W; dyadic tolerance readback through D; RegSeqRat handoff through R; and Real-seal nonescape through E.*

Proof. Project the carrier of Definition 8723.1. Its rows are precisely $S, T, W, D, R, E, H, C, P, N$, so carrier admission gives the complete local inventory.

The cofinal selector is not a global choice function. It is read from the StreamName source S, the finite selector budget T, and the displayed window row W, so each accepted tail request names only a finite later-window obligation.

The analytic readback is then forced through D, R, E. The dyadic tolerance row supplies the precision request, the RegSeqRat row supplies the regular rational handoff, and the Real row seals the read without adding a selected limit.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no quotient stream, choice sequence, trichotomy witness, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 8723.3 (Cofinal selector RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ *packet reads the cofinal selector through*

$$S \longmapsto T, W \longmapsto D \longmapsto R \longmapsto E$$

before the Real seal is visible. Thus the consumer first reads the $\mathsf{StreamName}^{\uparrow}$ *source, the finite cofinal selector budget and window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row; it cannot bypass them through quotient streams, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real-facing consumer. The carrier exposes the Real row only as E, and E is downstream of the dyadic tolerance and regular-rational handoff rows.

The selector route begins with the StreamName source S. The finite budget and window rows T, W record the cofinal tail request before any tolerance or regular sequence handoff is opened.

The rows D and R then supply the dyadic precision and RegSeqRat readback required by the consumer. Componentwise transport, displayed replay, provenance, and local naming preserve the same route through H, C, P, N.

Because the packet has no coordinate for selected limits, quotient streams, countable choice, trichotomy, or ambient Real completeness, every Real-facing read remains on the displayed chain before the seal E is consumed. □

Falsifiable prediction: Every $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ read consumed by a Real-facing completion route factors through StreamName source, finite cofinal selector budget, finite window, DyadicRatCore tolerance, RegSeqRat handoff, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-tail-equivalence: $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite cofinal selector budget between StreamName windows, dyadic tolerance readback, RegSeqRat handoff, Real sealing, transport, replay, provenance, and local naming while refusing selected limits, quotient streams, countable choice, trichotomy, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailSelectorBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalTailSelectorBudget}^{\uparrow}$ packet is generated from a StreamName source row, a finite cofinal tail-selector budget, finite windows, a DyadicRatCore tolerance row, a RegSeqRat handoff row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailSelectorBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8723.1, Definition 8723.2, and Definition 8723.3 seed the finite choice-free cofinal tail-selector route from StreamName windows to RegSeqRat and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailSelectorBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal selector exposure, dyadic tolerance readback, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CofinalTailSelectorBudgetUp

Chapter 8724

A Concrete Naming Certificate for MetaCICNormalizationCounterexampleBudget$^{\uparrow}$

The MetaCICNormalizationCounterexampleBudget$^{\uparrow}$ packet realizes the finite obstruction row of the MetaCIC normalization audit map of chapter 9383. It packages a bounded counterexample budget for candidate-mediated normalization checks: a consumer may inspect typed terms, candidate evidence, bounded residual traces, normal-form consistency seals, and obstruction rows, but it cannot promote the packet into global strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency. The Lean follow-up supplies Nontrivial and FieldFaithful instances for the finite packet interface.

The bounded counterexample budget consumes the candidate surface of chapter 5994 and the local-join surface of chapter 4238; its budget row refines those sibling frontier packets by adding residual-checker, obstruction, and normal-form-seal rows.

It is also a consumer of the normalization-frontier packet of chapter 892 and the MetaCIC critical-path route: the candidate evidence, residual checker, join envelope, and obstruction row refine the frontier budget before any read reaches kernel primitives.

Definition 8724.1 (MetaCIC normalization counterexample budget carrier)**.** *A* MetaCICNormalizationCounterexam *carrier is a finite* BHist *packet*

$$M = (T, K, B, R, J, F, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC typed-term source row, exposing the closed fragment and typing boundary under audit.*
2. *K is the candidate-evidence row used to mediate strong-normalization reads.*
3. *B is the bounded counterexample budget row, recording the finite number of residual traces, redex windows, or obstruction candidates inspected by the audit.*
4. *R is the residual-checker row, carrying the bounded redex and residual observations.*
5. *J is the local join-envelope row, recording the finite confluence-facing data visible to the counterexample search.*
6. *F is the normal-form consistency seal row, available only after the candidate, budget, residual, and join rows have been read.*

7. *O is the obstruction row, recording failed budget discharge without turning it into a global normalization theorem.*
8. *H records componentwise* hsame *transport for the typed-term, candidate, budget, residual, join, seal, obstruction, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same bounded audit route.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, B, R, J, F, O, H, C, P, N$. It has no coordinate for global strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Theorem 8724.2 (MetaCIC normalization counterexample budget obligations)**.** *For every accepted* $\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}$ *carrier $M = (T, K, B, R, J, F, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the typed-term source, candidate evidence, bounded counterexample budget, residual checker, local join envelope, normal-form consistency seal, obstruction row, componentwise transport, displayed replay, provenance, and local naming. Every normalization-audit consumer must read candidate evidence and bounded obstruction data before the seal F is visible, and every failed budget discharge remains confined to the obstruction row O.*

Proof. Project the carrier of Definition 8724.1. Its rows are exactly $T, K, B, R, J, F, O, H, C, P, N$, so accepted packets expose no normalization evidence outside the displayed inventory.

The candidate-mediated route is finite. The typed-term row T and candidate row K must be read before the bounded budget row B can expose residual observations through R and local join data through J.

The normal-form consistency seal F is downstream of those finite rows. A failed bounded search is recorded in O, but O is an obstruction row rather than a proof of subject reduction, evaluator equality, quotient term equality, or global strong normalization.

Finally H, C, P, N transport, replay, source, and name the same audit packet. Since no carrier coordinate names kernel consistency or an ambient evaluator equality, the obligations cannot escape the bounded counterexample budget. □

Falsifiable prediction: Every $\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}$ normalization-audit read factors through typed-term source, candidate evidence, bounded counterexample budget, residual-checker, local join-envelope, normal-form consistency seal, obstruction, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any bounded obstruction result is exported.

Independence witness: metacic-parallel-diamond-frontier, normalization-candidate-frontier, normal-form-consistency-seal, metacic-residual-join-envelope: $\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded counterexample budget, residual-checker route, local join envelope, normal-form consistency seal, obstruction row, transport, replay, provenance, and local naming while refusing global strong normalization, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationCounterexampleBudget}^{\uparrow}}$.

Constructive story: A MetaCICNormalizationCounterexampleBudget$^{\uparrow}$ packet is generated from a typed-term source row, candidate evidence, a bounded counterexample budget, residual-checker data, a local join envelope, a normal-form consistency seal, an obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationCounterexampleBudget}^{\uparrow}}$, seed)

Scope closed: Definition 8724.1 and Definition 8724.2 seed the bounded obstruction route for the MetaCIC normalization audit map.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalizationCounterexampleBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global strong normalization, subject reduction, evaluator equality, quotient term equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-mediated budget exposure, residual-checker readback, local join-envelope routing, normal-form seal nonescape, obstruction confinement, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalizationCounterexampleBudgetUp.

Chapter 8725

A Concrete Naming Certificate for $\mathsf{PuncturedInterval}^{\uparrow}$

Following Bishop and Bridges on deleted neighborhoods, the $\mathsf{PuncturedInterval}^{\uparrow}$ packet records a finite interval read around a displayed center while retaining a positive deleted-radius row. It sits after the located interval surface of chapter 394, the real interval surface of chapter 4111, and the dyadic-to-real spine of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names deleted-neighborhood finite interval reads for later limit and derivative consumers; it is not full calculus, a host topology on real numbers, a selected real representative, or ambient order completeness.

Definition 8725.1 (PuncturedInterval carrier)**.** *A* $\mathsf{PuncturedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, X, R, D, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{RealInterval}^{\uparrow}$ *source interval row, already carrying endpoint-and-enclosure data.*
2. *X is the center row, read as a* $\mathsf{Real}^{\uparrow}$ *name inside the interval surface.*
3. *R is the deleted-radius row, carrying a positive dyadic or regular rational radius around X.*
4. *D is the dyadic comparison row, comparing endpoint windows, the center row, and the deleted radius at a displayed tolerance.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for interval endpoints, center reads, and radius reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the windows and dyadic comparison budget into finite rational observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for interval, center, and deleted-neighborhood consumers.*
8. *H records componentwise* hsame *transport for source interval, center, radius, dyadic comparison, windows, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a deleted-neighborhood request through $I, X, R, D, W, Q, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PuncturedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PuncturedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host deleted-neighborhood topology, selected real representatives, derivative data, limit laws, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.PuncturedIntervalUp`

Theorem 8725.2 (PuncturedInterval NameCert obligations)**.** *For every accepted* $\mathsf{PuncturedInterval}^{\uparrow}$ *carrier* $J = (I, X, R, D, W, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PuncturedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission, interval-source readback, center readback, deleted-radius positivity, dyadic comparison exactness, StreamName window exposure, RegSeqRat finite readback, Real seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. No obligation is stored outside the displayed packet rows.*

Proof. Carrier admission is projection from Definition 8725.1. The projection exposes the interval source row I, center row X, positive deleted-radius row R, dyadic comparison row D, and the finite window, readback, and seal rows W, Q, S.

The deleted-neighborhood condition is read through R and D. The radius row supplies the positive exclusion around the center, while the dyadic comparison row is the only place where endpoint windows, center windows, and the deleted radius are compared.

The finite real-name route is $D \mapsto W \mapsto Q \mapsto S$. A consumer first fixes a displayed tolerance, then reads finite stream windows, then obtains regular rational readback, and only then reaches the Real seal.

Finally H, C, P, N transport, replay, source, and locally name the same route. Since the carrier contains no topology object, derivative object, limit law, selected representative, trichotomy decision, choice schedule, compactness theorem, or completeness principle, the listed rows exhaust the NameCert obligations. □

Theorem 8725.3 (PuncturedInterval deleted-neighborhood scope)**.** *Every deleted-neighborhood read of an accepted* $\mathsf{PuncturedInterval}^{\uparrow}$ *carrier reaches an interval or* $\mathsf{Real}^{\uparrow}$ *consumer only through the displayed route*

$$I, X, R \longmapsto D, W, Q \longmapsto S, H, C, P, N.$$

Thus the read exposes the source interval, center, positive deleted radius, dyadic comparison budget, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert before any downstream limit or derivative consumer can inspect the interval. The scope does not include full calculus, host real topology, quotient real equality, selected endpoint representatives, or ambient completeness.

Proof. Start with an accepted carrier $J = (I, X, R, D, W, Q, S, H, C, P, N)$. The interval row I and center row X identify the displayed source and center, while R is the only coordinate that records the deleted positive radius.

Read the dyadic comparison route next. The comparison row D attaches the radius exclusion to finite endpoint and center windows; the window row W and readback row Q keep this comparison inside StreamName and RegSeqRat data.

Only after this finite comparison route may the Real seal S be consumed. The structural rows H, C, P, N preserve the same deleted-neighborhood read through transport, replay, provenance, and local naming.

Because the carrier has no coordinate for calculus rules, a host topology, quotient equality, selected real representatives, or completeness, any later limit or derivative consumer receives only the scoped deleted-neighborhood packet described above. □

Closure status for $\mathsf{NameCert}_{\mathsf{PuncturedInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PuncturedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8725.1, Definition 8725.2, and Definition 8725.3 seed the finite deleted-neighborhood interval surface over $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Constructive story: A $\mathsf{PuncturedInterval}^{\uparrow}$ packet is generated from a RealInterval source row, a RealUp center row, a positive deleted-radius row, a dyadic comparison row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PuncturedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full calculus, host real topology, derivative rules, limit laws, quotient real equality, selected real representatives, trichotomy, countable choice, compactness, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center readback, deleted-radius positivity, dyadic comparison exactness, StreamName window exposure, RegSeqRat readback, Real seal nonescape, transport, provenance, and local naming for BEDC.Derived.PuncturedIntervalUp.

Chapter 8726

A Concrete Naming Certificate for $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$

Following Bishop and Bridges on finite interval reasoning over constructive real numbers, the $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$ packet records the displayed overlap ledger for two finite dyadic interval rows before any completion-facing consumer treats the overlap as a Real interval. It sits over the dyadic endpoint arithmetic of chapter 291, the finite endpoint-cell surface of chapter 306, the finite naming windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is a finite boundary ledger for nested-interval, finite-cover, and compactness consumers; it does not import trichotomy, quotient interval equality, selected points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8726.1 (Dyadic interval intersection ledger carrier)**.** *A* $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, I_0, I_1, B, O, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-arithmetic row, carrying lower and upper endpoint comparisons for the two inspected dyadic intervals.*
2. *I_0 and I_1 are the two* $\mathsf{DyadicInterval}^{\uparrow}$ *source rows whose overlap boundary is being recorded.*
3. *B is the finite boundary row, displaying the two candidate lower endpoints, the two candidate upper endpoints, and the endpoint comparisons needed to retain the intersection cell.*
4. *O is the overlap ledger row, recording the retained lower endpoint, retained upper endpoint, and the finite witness that the lower endpoint does not pass the upper endpoint.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained overlap endpoints.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained dyadic endpoint windows and tolerance requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, I_0, I_1, B, O, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for the dyadic arithmetic, interval, boundary, overlap, window, readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the two source interval rows through the overlap ledger and into the Real-facing seal.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, I_0, I_1, B, O, W, Q, E, H, C, P, N$. It has no coordinate for host interval equality, quotient real equality, trichotomy of endpoints, selected overlap points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8726.2 (Dyadic interval intersection ledger NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$ *carrier $L = (D, I_0, I_1, B, O, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}}$ *surface consists exactly of dyadic endpoint arithmetic through D, the two source interval rows I_0, I_1, finite boundary comparison through B, retained-overlap exposure through O, stream-window exposure through W, regular rational readback through Q, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. Thus every accepted interval-intersection consumer must read the finite route*

$$D, I_0, I_1 \longmapsto B \longmapsto O \longmapsto W, Q \longmapsto E, H, C, P, N.$$

Proof. Project the carrier of Definition 8726.1. The rows I_0 and I_1 name the two inspected dyadic intervals, while D supplies the endpoint arithmetic used to compare their displayed bounds.

The boundary row B records only finite comparisons among those endpoints. The overlap row O retains the displayed lower and upper endpoint selected by those comparisons together with the finite nonempty-cell witness.

The stream window row W and regular rational readback row Q are downstream of the retained overlap ledger, and the Real seal E is downstream of both. Transport, replay, provenance, and local naming through H, C, P, N preserve that same displayed route.

Since the carrier has no coordinate for host interval equality, quotient real equality, endpoint trichotomy, selected overlap points, open-cover compactness, or ambient Real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8726.3 (Dyadic interval intersection ledger Real-seal nonescape)**.** *Every Real-facing read from an accepted* $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$ *packet factors through the dyadic endpoint row D, the two* $\mathsf{DyadicInterval}^{\uparrow}$ *source rows I_0, I_1, the finite boundary row B, the retained overlap ledger O, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is consumed. The handoff exports a finite intersection boundary and no trichotomy principle, quotient interval equality, selected point, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Apply the NameCert obligations of Definition 8726.2. They place the two source interval rows and the dyadic endpoint arithmetic before the boundary comparison row.

The retained overlap ledger O is downstream of those finite comparisons, and the stream and regular-rational readback rows W, Q are downstream of the retained endpoints. A Real-facing consumer therefore reaches E only after the finite overlap route has been displayed.

Finally replay H, C, P, N. These rows transport, replay, source, and name the same packet without adding host interval equality, quotient Real equality, endpoint trichotomy, a selected point, open-cover compactness, or ambient completeness. The Real-facing handoff remains a finite BEDC ledger. □

Falsifiable prediction: Every accepted DyadicIntervalIntersectionLedgerUp consumer must display DyadicRatCore endpoint arithmetic, two DyadicInterval source rows, a finite boundary row, retained overlap ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before a Real-facing overlap read is public.

Independence witness: dyadicratcore, dyadicinterval, streamname, regseqrat, real: DyadicIntervalIntersectionLed is not bijective to any listed sibling because it binds two source intervals, finite endpoint comparisons, the retained overlap ledger, readback windows, terminal Real sealing, transport, replay, provenance, and local naming as one intersection-boundary carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalIntersectionLedger}\uparrow}$.

Constructive story: A $\mathsf{DyadicIntervalIntersectionLedger}^{\uparrow}$ packet is generated from DyadicRatCore endpoint arithmetic, two DyadicInterval source rows, finite boundary comparison, retained overlap ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalIntersectionLedger}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8726.1, Definition 8726.2, and Definition 8726.3 seed the finite interval-intersection boundary for DyadicRatCore, DyadicInterval, StreamName, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalIntersectionLedger}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim endpoint trichotomy, host interval equality, quotient Real equality, selected overlap points, open-cover compactness, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint comparisons, source-interval exposure, retained-overlap stability, StreamName windowing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalIntersectionLedgerUp.

Chapter 8727

A Concrete Naming Certificate for $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$ packet records the finite modulus surface that turns a displayed interval contraction into fixed-point tolerances without selecting an ambient limit. It sits after the interval-contraction route of chapter 5867, the fixed-point interval packet of chapter 5811, the dyadic tolerance ledger of chapter 291, the finite name windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 modulus for fixed-point consumers; it is not a selected limit, quotient real equality, countable choice, a host Picard theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8727.1 (Choice-free fixed-point modulus carrier)**.** *A* $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, T, S, D, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval source row imported from the* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *route, including the located endpoint and interval-admission data consumed by the fixed-point request.*
2. *T is the contraction self-map row, recording that the displayed map preserves the interval source and exposes its contraction witness.*
3. *S is the shrinkage modulus row, converting each requested fixed-point tolerance into a finite contraction-stage budget.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the shrinkage row and by every downstream rational read.*
5. *Q is the dyadic comparison row that records the retained radius, error, and slack cells for the requested tolerance.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly which contraction stages are inspected.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and the same dyadic tolerance ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, T, S, D, Q, W, R have been displayed.*

9. *H records componentwise* hsame *transport for the interval source, self-map, shrinkage, dyadic, window, readback, real-seal, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from each tolerance request through $I, T, S, D, Q, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$, $\mathsf{BishopIntervalContraction}^{\uparrow}$, BishopFixedPointInterva $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, S, D, Q, W, R, E, H, C, P, N$. It has no coordinate for a chosen fixed point, selected limit, quotient stream equality, countable choice, host completeness, or an ambient fixed-point theorem.

Theorem 8727.2 (Choice-free fixed-point modulus NameCert obligations)**.** *For every accepted* $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$ *carrier*

$$M = (I, T, S, D, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval source, contraction self-map, shrinkage modulus, dyadic tolerance, comparison, finite-window, regular readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *interval compatibility, requiring each fixed-point tolerance request to read I and T before any shrinkage budget or window is exposed;*
3. *modulus exactness, requiring S, D, Q to be the sole source of contraction depth, tolerance, radius, error, and slack data;*
4. *finite readback, requiring W and R to be downstream of the same shrinkage and dyadic rows before the* $\mathsf{Real}^{\uparrow}$ *seal E can be consumed;*
5. *non-escape, requiring E, H, C, P, N to export no selected limit, quotient real equality, countable-choice schedule, host Picard theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No choice-free fixed-point modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8727.1. The displayed tuple $I, T, S, D, Q, W, R, E, H, C, P, N$ is the entire public surface of the packet.

The interval and contraction rows are read first. A tolerance request cannot inspect a stage window until the interval source I and self-map row T have been displayed, so the packet inherits the interval-contraction route before any fixed-point modulus data is visible.

The modulus rows then determine the finite budget. The shrinkage row S reads the requested tolerance through the dyadic ledger D and comparison row Q, and these rows are the only place where contraction depth, radius, error, or slack data occur.

Finally, the readback and seal remain downstream. The finite windows W and $\mathsf{RegSeqRat}^{\uparrow}$ readback R are consumed before the terminal $\mathsf{Real}^{\uparrow}$ seal E, while H, C, P, N transport, replay, source, and name that same route. Since the carrier contains no selected limit, quotient equality, countable-choice row, host Picard theorem, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, T, S, D, Q, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: bishop-interval-contraction, bishop-fixed-point-interval, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the joint packet containing an interval source, contraction self-map, shrinkage modulus, dyadic tolerance and comparison rows, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}$ packet is generated from an interval source row, contraction self-map row, shrinkage modulus row, dyadic tolerance and comparison rows, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8727.1 and Definition 8727.2 seed the finite choice-free modulus route from displayed interval contraction data to fixed-point Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceFreeFixedPointModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, countable choice, a host Picard theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, interval compatibility, modulus exactness, finite readback, terminal Real-seal non-escape, and local naming for BEDC.Derived.ChoiceFreeFixedPointModulusUp.

Chapter 8728

A Concrete Naming Certificate for $\mathsf{WarsawCircle}^{\uparrow}$

Following classical continuum theory, the $\mathsf{WarsawCircle}^{\uparrow}$ packet records the Warsaw circle as a compact metric continuum assembled from the topologist sine curve, a closing arc, and the displayed topology and real-coordinate carriers already present in the BEDC library. It sits beside the topologist sine curve of chapter 7316, the compact metric carrier of chapter 129, the topology carrier of chapter 88, the real-coordinate carrier of chapter 38, and the continuous-map carrier of chapter 130. The packet names the finite row surface by which a continuum-theory consumer reads the Warsaw circle; it does not assert planar embedding uniqueness, local connectedness, path connectedness, selected parametrizations, quotient equality of point names, or a Lean-checked standard bridge.

Definition 8728.1 (Warsaw circle carrier)**.** *A* $\mathsf{WarsawCircle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, A, T, M, R, F, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{TopologistSineCurve}^{\uparrow}$ *spine row naming the displayed sine-curve component.*
2. *A is the finite closing-arc row recording the added compact connector between the two endpoint sides of the displayed spine.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *subspace-topology row on the assembled carrier.*
4. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric row for the closed bounded carrier surface.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate row used by the planar point names and endpoint witnesses.*
6. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *inclusion-and-attachment row recording the displayed continuous maps from the sine spine and the closing arc into the assembled carrier.*
7. *H records componentwise* hsame *transport for the spine, arc, topology, compact metric, real coordinate, continuous-map, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Warsaw-circle request through the sine-curve spine, closing arc, topology, compact metric, real-coordinate, and continuous-map rows.*
9. *P is the* Pkg *provenance over* $\mathsf{WarsawCircle}^{\uparrow}$, $\mathsf{TopologistSineCurve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WarsawCircle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected parametrization of the continuum, planar embedding uniqueness, local connectedness, path connectedness, quotient equality of point names, or an ambient theorem of continuum theory.

Theorem 8728.2 (Warsaw circle NameCert obligations)**.** *For every accepted* $\mathsf{WarsawCircle}^{\uparrow}$ *carrier* $W = (S, A, T, M, R, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WarsawCircle}^{\uparrow}}$ *surface consists exactly of sine-curve spine exposure through* S*, closing-arc exposure through* A*, topology exposure through* T*, compact metric exposure through* M*, real-coordinate exposure through* R*, continuous attachment exposure through* F*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A consumer therefore reads the Warsaw circle only through the finite route*

$$S, A, T, M, R, F \longmapsto H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8728.1. The sine-curve row S and closing-arc row A display the two geometric pieces before any assembled continuum read is public.

The topology row T, compact metric row M, real-coordinate row R, and continuous attachment row F then expose the carrier surface used by a continuum-theory consumer. These rows are the only rows available before structural replay.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite packet, so no selected parametrization, embedding uniqueness theorem, local connectedness claim, path connectedness claim, quotient equality of point names, or ambient continuum theorem enters the NameCert surface. □

Theorem 8728.3 (Warsaw circle sine-curve closure handoff)**.** *Every continuum-facing read from an accepted* $\mathsf{WarsawCircle}^{\uparrow}$ *carrier factors through the* $\mathsf{TopologistSineCurve}^{\uparrow}$ *spine row* S *before it may consume the closing arc* A*, topology row* T*, compact metric row* M*, real-coordinate row* R*, and continuous attachment row* F*. The handoff from the sine-curve carrier to the Warsaw-circle carrier is therefore the displayed finite route*

$$\mathsf{TopologistSineCurve}^{\uparrow} \longmapsto A, T, M, R, F, H, C, P, N,$$

not a new proof of local connectedness, path connectedness, selected planar parametrization, embedding uniqueness, quotient equality, or ambient continuum classification.

Proof. Start with the carrier of Definition 8728.1. Its first row is the displayed $\mathsf{TopologistSineCurve}^{\uparrow}$ spine, so any accepted Warsaw-circle read exposes the inherited sine-curve component before the closing arc or assembled surface is available.

Apply the NameCert obligations of Definition 8728.2. They place the closing arc, topology, compact metric, real-coordinate, and continuous-map rows downstream of that displayed spine and upstream of the structural rows.

The handoff is therefore finite and carrier-local. Since H, C, P, N only transport, replay, source, and name the same packet, the route cannot be reread as local connectedness, path connectedness, selected parametrization, planar embedding uniqueness, quotient equality of point names, or a general continuum-classification theorem. □

Falsifiable prediction: Every accepted $\mathsf{WarsawCircle}^{\uparrow}$ consumer must display the $\mathsf{TopologistSineCurve}^{\uparrow}$ spine row, closing-arc row, $\mathsf{Topology}^{\uparrow}$ row, $\mathsf{CompactMetric}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ coordinate row, $\mathsf{ContinuousMap}^{\uparrow}$ attachment row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row before the Warsaw-circle certificate can be consumed.

Independence witness: topologist-sine-curve, compactmetric, topology, real, continuousmap: $\mathsf{WarsawCircle}^{\uparrow}$ is not bijective to any listed sibling because it carries the joint sine spine, closing arc, topology, compact metric, real-coordinate, continuous-attachment, transport, replay, provenance, and local naming rows. The sine-curve sibling lacks the closing arc and assembled compact-metric carrier; the compact metric sibling lacks the sine-spine and attachment rows; the topology sibling lacks compact-metric and continuum rows; the real sibling supplies only coordinate names; and the continuous-map sibling records maps without the named Warsaw-circle carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{WarsawCircle}^{\uparrow}}$.

Constructive story: A $\mathsf{WarsawCircle}^{\uparrow}$ packet is generated from a topologist-sine-curve spine, a finite closing-arc row, topology and compact metric rows, real-coordinate rows, continuous attachment rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WarsawCircle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8728.1, Definition 8728.2, and Definition 8728.3 seed the finite Warsaw-circle carrier over the existing sine-curve, topology, compact metric, real-coordinate, and continuous-map anchors.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WarsawCircle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close local connectedness, path connectedness, planar embedding uniqueness, selected parametrizations, quotient equality of point names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sine-spine exposure, closing-arc exposure, topology and compact metric routing, real-coordinate exposure, continuous attachment routing, structural nonescape, and local naming for BEDC.Derived.WarsawCircleUp.

Chapter 8729

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ packet records the diagonal-limit witness of chapter 9238 as finite NameCert data. It consumes the diagonal representative and regularity theorem of Definition 9238.3 and Definition 9238.4, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing boundary of chapter 1639. The packet gives Real and Completion consumers a named diagonal-limit witness; it is not a quotient stream, a selected representative, countable choice, a host limit theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8729.1 (Regular Cauchy limit witness carrier)**.** *A* $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, D, W, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed regular-Cauchy source-family row whose members are read only through finite windows.*
2. *M is the diagonal modulus row, recording the primitive-recursive stage selector used by Definition 9238.3.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, triangle, and slack ledger used by the diagonal regularity estimate.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family for the source reads, diagonal index reads, and output read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the diagonal representative certified by the same modulus, tolerance, and window rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after X, M, D, W, R have been displayed.*
7. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, recording that the witness is completion-facing without selecting a completed host limit.*
8. *H records componentwise* hsame *transport for the source family, modulus, dyadic ledger, finite windows, readback, real seal, completion boundary, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each diagonal-limit request through* $X, M, D, W, R, E, B, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, D, W, R, E, B, H, C, P, N$. It has no coordinate for quotient streams, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8729.2 (Regular Cauchy limit witness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ *carrier*

$$L = (X, M, D, W, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular-Cauchy source family, diagonal modulus, dyadic tolerance and triangle ledger, finite windows, regular readback, terminal real seal, completion boundary, transport, replay, provenance, and local naming rows;*
2. *diagonal-stage exactness, requiring each output read to use the displayed modulus row M and no selected representative outside the packet;*
3. *finite-window coverage, requiring every source and output observation to be named by W and compared through the same dyadic ledger D;*
4. *RegSeqRat handoff, requiring R to be downstream of X, M, D, W and upstream of the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*
5. *completion boundary non-escape, requiring B, H, C, P, N to export no quotient streams, countable-choice selector, selected host limit, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No regular Cauchy limit-witness obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8729.1. The displayed tuple $X, M, D, W, R, E, B, H, C, P, N$ is the entire public surface of the packet.

The diagonal-stage clause reads the source family through X and the stage selector through M. Because M is a displayed row, the output representative is obtained by the finite diagonal recipe rather than by choosing a representative from a host completion.

The window and dyadic clauses then control the estimate. Every source read and output read is named by W, while D supplies the tolerance, triangle, and slack data used by the diagonal regularity route of Definition 9238.4.

The regular readback row R is therefore downstream of X, M, D, W. The terminal seal E and completion boundary B can be consumed only after that route has been replayed and transported by H, C, P, N. Since the carrier contains no quotient-stream, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ Real-facing or Completion-facing read exposes a regular-Cauchy source family, displayed diagonal modulus, DyadicRatCore

tolerance ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, BishopRegularCauchyCompletion boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-cauchy-completion, regular-cauchy-diagonal-limit: no listed sibling supplies the joint packet containing the source family, diagonal modulus, dyadic triangle ledger, finite windows, regular readback, terminal Real seal, completion boundary, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ packet is generated from a regular-Cauchy source family, displayed diagonal modulus, DyadicRatCore tolerance and triangle ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, BishopRegularCauchyCompletion boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8729.1 and Definition 8729.2 seed the finite diagonal-limit witness route from regular-Cauchy source data to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-stage exactness, finite-window coverage, RegSeqRat handoff, completion-boundary non-escape, and local naming for BEDC.Derived.RegularCauchyLimitWitnessUp.

Chapter 8730

A Concrete Naming Certificate for $\mathsf{BishopSpreadChoice}^{\uparrow}$

Following Bishop and Bridges and Troelstra–van Dalen on spreads, choice sequences, and finite observation, the $\mathsf{BishopSpreadChoice}^{\uparrow}$ packet records a finite choice-sequence prefix only after the displayed spread-law admission row has accepted it. It sits between the finite choice-sequence source of chapter 1482, the spread carrier of chapter 1677, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite-prefix bridge for L10 Real and Completion consumers; it is not a completed branch, countable choice, quotient stream equality, fan compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8730.1 (Bishop spread-choice carrier)**.** *A* $\mathsf{BishopSpreadChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *finite-prefix row, exposing only the opened prefix entries and their displayed request order.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row. It records the finite spread-law decisions that admit the prefixes in Q.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that displays the admitted prefix as the only stream window available to downstream readers.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the admitted window and its displayed rational requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, S, W, R have been consumed.*
6. *H records componentwise* hsame *transport for the prefix, admission, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite choice-sequence request through spread-law admission, window display, regular readback, and Real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopSpreadChoice}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSpreadChoice}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $Q, S, W, R, E, H, C, P, N$. It has no coordinate for a completed infinite branch, a host function, quotient stream equality, countable choice, fan compactness, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8730.2 (Bishop spread-choice NameCert obligations)**.** *For every accepted* $\mathsf{BishopSpreadChoice}^{\uparrow}$ *carrier*

$$B = (Q, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSpreadChoice}^{\uparrow}}$ *surface consists exactly of finite choice-sequence prefix exposure, spread-law admission for that prefix, StreamName window display, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from those rows.*

Proof. Project the carrier of Definition 8730.1. The visible analytic route begins with the prefix row Q, checks it through the spread-law row S, and then exposes the admitted window W.

The regular readback and Real seal are downstream. Row R reads only the admitted finite window, and row E is available only after Q, S, W, R have been displayed.

The structural rows preserve the same route. Transport H is componentwise, replay C follows the displayed admission and readback chain, and P, N source and name that packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8730.3 (Bishop spread-choice finite-prefix stability)**.** *Let $B = (Q, S, W, R, E, H, C, P, N)$ and $B' = (Q', S', W', R', E', H', C', P', N')$ be accepted* $\mathsf{BishopSpreadChoice}^{\uparrow}$ *carriers. If Q and Q' display the same finite choice-sequence prefix, S and S' give the same spread-law admission decisions for that prefix, and W and W' are componentwise* hsame *as admitted StreamName windows, then every finite derivative-free read exported through H, C, P, N and through H', C', P', N' displays the same W-window before reaching the regular readback and Real seal.*

Proof. Induct on the displayed prefix length. The empty prefix has no opened entry, so the only visible window is the root window admitted by the spread-law row.

For the successor step, assume the shorter prefix has the same admitted window in both carriers. The next displayed entry is read from the same finite choice-sequence prefix and is admitted by the same spread-law decision, so the extended StreamName row is componentwise hsame.

The readback cannot introduce a second window. By Definition 8730.2, every exported read must pass through Q, S, W before R and E. The transport, replay, provenance, and local naming rows therefore preserve the displayed finite row inventory rather than replacing it by a host stream, quotient representative, selected branch, countable-choice schedule, or ambient completion datum. □

Falsifiable prediction: Every accepted $\mathsf{BishopSpreadChoice}^{\uparrow}$ Real-facing read exposes one finite $\mathsf{ChoiceSequence}^{\uparrow}$ prefix row, one $\mathsf{SpreadLaw}^{\uparrow}$ admission row, one $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: choice-sequence, brouwer-spread, streamname, regseqrat, real: $\mathsf{BishopSpreadChoice}^{\uparrow}$ is not bijective to any listed sibling because it binds finite prefix data to spread-law admission before any StreamName, RegSeqRat, or Real row can be consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSpreadChoice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopSpreadChoice}^{\uparrow}$ packet is generated from a finite ChoiceSequence prefix row, a SpreadLaw admission row, a StreamName window row, a RegSeqRat readback row, a terminal RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopSpreadChoice}^{\uparrow}}$, seed)

Scope closed: Definition 8730.1, Definition 8730.2, and Definition 8730.3 seed the finite-prefix bridge from choice-sequence observation through spread-law admission to StreamName, RegSeqRat, and Real-name handoff rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopSpreadChoice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed infinite branches, countable choice, quotient stream equality, fan compactness, selected limits, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking for BishopSpreadChoice$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, spread-law admission stability, finite-prefix stability, readback handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopSpreadChoiceUp.

Chapter 8731

A Concrete Naming Certificate for $\mathsf{SequentialOrderCompactness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentialOrderCompactness}^{\uparrow}$ packet records the finite order-window route by which a displayed real sequence in a compact interval can be inspected before any Real or Completion consumer reads an order-facing compactness claim. It sits beside the subsequence-window surface of chapter 2372, the interval boundary of chapter 3986, and the terminal real-name seal of chapter 38. The packet names only the local finite window used by an order-monotone read; it is not open-cover compactness, not a choice-selected subsequence, not trichotomy, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8731.1 (Sequential order compactness carrier)**.** *A* $\mathsf{SequentialOrderCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, I, O, R, W, D, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Real}^{\uparrow}$ *sequence source row whose inspected terms are offered only through displayed real-name seals.*
2. *I is the* $\mathsf{CompactInterval}^{\uparrow}$ *bracket row naming the interval boundary in which the sequence is read.*
3. *O is the finite monotone/order window. It records the inspected order comparisons and adjacent bracket witnesses visible inside the current window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the sequence terms that appear in the order window.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite indices and readback windows being inspected.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the displayed rational readbacks inside the compact interval bracket.*
7. *H records componentwise* hsame *transport for the source sequence, interval bracket, order window, readback, schedule, tolerance, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an order-window request through A, I, O, R, W, D, H before any compactness-facing consumer reads the packet.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialOrderCompactness}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentialOrderCompactness}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $A, I, O, R, W, D, H, C, P, N$. It has no coordinate for open-cover compactness, a selected infinite subsequence, trichotomy of real names, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8731.2 (Sequential order compactness NameCert obligations)**.** *For every accepted* $\mathsf{SequentialOrderCompactness}^{\uparrow}$ *carrier $S = (A, I, O, R, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialOrderCompactness}^{\uparrow}}$ *surface consists exactly of source-sequence exposure through A; compact-interval bracketing through I; finite monotone/order-window exposure through O; regular rational readback through R; finite schedule exposure through W; dyadic tolerance comparison through D; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. Every Real or Completion consumer must first pass through the sequential compactness sibling of chapter 2372, the compact-interval sibling of chapter 3986, and the real-name seal of chapter 38.*

Proof. Project the carrier in Definition 8731.1. The displayed rows are exactly $A, I, O, R, W, D, H, C, P, N$, so any accepted read begins with the source sequence, interval bracket, finite order window, readback, schedule, and dyadic tolerance rows.

The order-facing compactness information is local to the finite window. The compact interval row supplies the bracket, the StreamName row supplies the inspected indices, and the RegSeqRat and DyadicRatCore rows supply the finite rational comparison data used inside that bracket.

Transport and replay preserve the same rows componentwise. Provenance and local naming then bind the packet as a $\mathsf{SequentialOrderCompactness}^{\uparrow}$ certificate. Since the carrier contains no selected infinite subsequence, open-cover compactness witness, quotient real equality, trichotomy branch, countable-choice schedule, or ambient completeness row, the listed fields exhaust the local obligations. □

Theorem 8731.3 (Sequential order compactness finite-window extraction)**.** *Let $S = (A, I, O, R, W, D, H, C, P, N)$ be an accepted* $\mathsf{SequentialOrderCompactness}^{\uparrow}$ *carrier. Every order-facing compactness read exported by S factors through the finite route*

$$A \to I \to O \to R \to W \to D \to H \to C \to P \to N.$$

The extracted witness is a finite inspected order window over displayed real-name readbacks inside the compact interval bracket. It is not a selected convergent subsequence, not an open-cover compactness proof, not a quotient-real equality, not a trichotomy decision, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the source sequence row A and compact-interval row I first. They determine which displayed real-name terms and which interval bracket are visible to the packet.

Next inspect the order window O through the readback and schedule rows R and W. The dyadic tolerance row D supplies the finite comparison budget for the displayed rational approximants, so the extracted information is a finite window rather than an infinite subsequence.

Finally replay the structural rows H, C, P, N. Componentwise transport and displayed continuation replay cannot create a selected limit point, a quotient equality, a trichotomy branch, or an open-cover witness. Thus the consumer receives exactly the finite order-window extraction over the compact interval route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialOrderCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SequentialOrderCompactness}^{\uparrow}$ packet is generated from a displayed RealUp sequence source, a CompactInterval bracket, a finite monotone/order window, RegSeqRat readback, StreamName schedule, DyadicRatCore tolerance data, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{SequentialOrderCompactness}\uparrow}$, seed)

Scope closed: Definition 8731.1, Definition 8731.2, and Definition 8731.3 seed the finite order-window surface over displayed real sequences in compact interval brackets.

Formal verification: FormalStatus(NameCert$_{\mathsf{SequentialOrderCompactness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected subsequences, trichotomy, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-bracket exposure, finite order-window extraction, readback, schedule, tolerance, transport, replay, provenance, and local naming for BEDC.Derived.SequentialOrderCompactnessUp.

Chapter 8732

A Concrete Naming Certificate for $\mathsf{SequentialCompletionReflector}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{SequentialCompletionReflector}^{\uparrow}$ packet records the boundary where regular sequential completion data are reflected into a finite NameCert handoff. It sits after the completion schedule of $\mathsf{CauchyCompletionMonad}^{\uparrow}$ in chapter 378, the metric completion boundary of $\mathsf{MetricCompletion}^{\uparrow}$ in chapter 1506, the finite observation windows of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular rational readback of $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, and the terminal real-name seal of $\mathsf{Real}^{\uparrow}$ in chapter 38. The reflector names how sequential Cauchy data are handed to Real and Completion consumers as displayed rows; it is not quotient equality, a host selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 8732.1 (Sequential completion reflector carrier)**.** *A* $\mathsf{SequentialCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, K, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *schedule row, naming the unit and bind surface consumed by completion-facing reads.*
2. *K is the* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary row, recording the source metric presentation and the completion target admitted by the reflector.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the inspected sequence entries and precision requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, K, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, completion boundary, finite windows, regular readback, terminal seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each sequential completion request through M, K, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SequentialCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletionReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, W, R, E, H, C, P, N$. It has no coordinate for quotient equality, host selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or checked bridge data.*

Theorem 8732.2 (Sequential completion reflector NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompletionReflector}^{\uparrow}$ *carrier*

$$S = (M, K, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion monad schedule, metric completion boundary, finite windows, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*
2. *sequential-source exposure, requiring each completion request to pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W before any regular readback is consumed;*
3. *regular-readback handoff, requiring R to be downstream of M, K, W and upstream of the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*
4. *reflector non-escape, requiring H, C, P, N to replay only the displayed rows and to export no quotient completion, selected representative, countable-choice schedule, host limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

No sequential completion reflector obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8732.1. The tuple $S = (M, K, W, R, E, H, C, P, N)$ is the whole surface exposed to a sequential completion consumer.

The sequential-source clause reads each requested stage through W. Since W is a finite $\mathsf{StreamName}^{\uparrow}$ window row, the consumer obtains only inspected entries and precision requests, not a chosen infinite representative.

The regular-readback clause then routes the same finite windows through R. The completion schedule M and metric completion boundary K remain visible before the terminal $\mathsf{Real}^{\uparrow}$ seal E is consumed, so the Real-facing handoff is a displayed regular-rational readback rather than a quotient completion.

Finally, H, C, P, N transport, replay, name, and package the same rows. Because the carrier contains no coordinate for quotient equality, selected limits, countable choice, ambient completeness, or bridge checking, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{SequentialCompletionReflector}^{\uparrow}$ consumer must display a CauchyCompletionMonad schedule row, a MetricCompletion boundary row, a StreamName finite-window row, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before the reflector can be consumed.

Independence witness: cauchycompletionmonad, metriccompletion, streamname, regseqrat, real: no listed sibling supplies the joint reflector packet containing the completion schedule, metric boundary, finite windows, regular readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletionReflector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SequentialCompletionReflector}^{\uparrow}$ packet is generated from a CauchyCompletionMonad schedule row, a MetricCompletion boundary row, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SequentialCompletionReflector}\uparrow}$, seed)

Scope closed: Definition 8732.1 and Definition 8732.2 seed the finite reflector route from sequential Cauchy completion data to Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{SequentialCompletionReflector}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality, host selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequential-source exposure, regular-readback handoff, reflector non-escape, and local naming for BEDC.Derived.SequentialCompletionReflectorUp.

Chapter 8733

A Concrete Naming Certificate for $\mathsf{DenseRationalEmbedding}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DenseRationalEmbedding}^{\uparrow}$ packet records the finite handoff by which rational certificates are read as dense real-name data before completion consumers inspect a $\mathsf{Real}^{\uparrow}$ seal. It sits after the rational carrier of chapter 37, the dyadic comparison core of chapter 291, the regular rational sequence route of chapter 301, the finite stream-window interface of chapter 126, and the terminal real-name packet of chapter 38. The packet names a displayed density route from rational witnesses to real-name seals; it is not quotient real equality, trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8733.1 (Dense rational embedding carrier)**.** *A* $\mathsf{DenseRationalEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, Y, S, R, E, A, H, C, P, N)$$

with the following rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row naming the rational witness to be embedded.*
2. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row, recording the dyadic tolerance and order checks used to locate Q inside a real-name window.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that exposes the inspected real-name prefixes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row used to transport the rational witness through completion-facing Cauchy data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the terminal real name reached by the displayed embedding route.*
6. *A is the density-approximation row, recording that every inspected real-name window admits a rational witness read through Q, Y, S, R, E at the displayed tolerance.*
7. *H records componentwise* hsame *transport for the rational, dyadic, stream, regular-sequence, real-seal, density, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a density request through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DenseRationalEmbedding}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseRationalEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, Y, S, R, E, A, H, C, P, N$. It has no coordinate for quotient real equality, trichotomy, selected limit objects, countable choice, or ambient real completeness.

Theorem 8733.2 (Dense rational embedding NameCert obligations)**.** *For every accepted* $\mathsf{DenseRationalEmbedding}^{\uparrow}$ *carrier $D = (Q, Y, S, R, E, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseRationalEmbedding}^{\uparrow}}$ *surface consists exactly of rational source admission, dyadic tolerance comparison, finite stream-window exposure, regular rational sequence readback, terminal real sealing, density-approximation exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every density read is therefore confined to the finite route*

$$Q, Y, S, R, E, A \longmapsto H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8733.1. The rows Q, Y, S, R, E, A are exactly the rational source, dyadic tolerance, stream-window, regular-sequence, real-seal, and density-approximation data available before structural replay.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. They add no quotient-real equality, no trichotomy branch, no selected limit, and no choice principle.

Thus the local NameCert surface is exhausted by the displayed rows. A completion-facing consumer can read density only through this finite route. □

Theorem 8733.3 (Dense rational embedding density window)**.** *Every real-name density request accepted by a* $\mathsf{DenseRationalEmbedding}^{\uparrow}$ *packet must display a finite stream window S, a regular rational readback row R, a dyadic tolerance row Y, and a rational witness row Q before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is consumed. The packet therefore supplies density as a finite window statement over rational certificates, not as an ambient completeness or selected-limit principle.*

Proof. Read the density-approximation row A. By the carrier definition, A is attached to the rows Q, Y, S, R, E and has no independent coordinate outside them.

The stream and regular-sequence rows determine what part of the real name is inspected. The dyadic row records the tolerance comparison, and the rational row supplies the witness used inside that tolerance.

Finally the real seal E authenticates the same inspected window. Since selected limits, quotient equality, trichotomy, choice, and ambient completeness are absent from the carrier, the density statement remains a finite rational-window handoff. □

Theorem 8733.4 (Dense rational embedding real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenseRationalEmbedding}^{\uparrow}$ *packet receives the real-name seal only after the route*

$$\mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

has been displayed and authenticated by A, H, C, P, N. Thus the handoff from rational certificates to real-name seals is a NameCert route for completion proofs, not a claim of quotient real equality, trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the rows of Definition 8733.1. The real seal E is downstream of the rational witness Q, dyadic comparison Y, stream window S, and regular rational readback R.

The density row A records that this route supplies the rational witness required by the inspected window. The transport and replay rows H, C preserve that route componentwise, and P, N source and name it.

No remaining coordinate can expose quotient equality of real names, a trichotomy decision, a selected limit object, choice, or ambient completeness. Hence the seal is exactly the displayed finite handoff from rational certificates into the real-name boundary. □

Falsifiable prediction: Every accepted $\mathsf{DenseRationalEmbedding}^{\uparrow}$ completion-facing read must display a $\mathsf{Rat}^{\uparrow}$ source row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal row, density-approximation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before the density handoff is consumed.

Independence witness: rat, dyadicratcore, regseqrat, streamname, real: $\mathsf{DenseRationalEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it binds the rational witness, dyadic tolerance comparison, finite stream window, regular rational readback, terminal real seal, density row, transport, replay, provenance, and local naming into one completion-facing handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseRationalEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseRationalEmbedding}^{\uparrow}$ packet is generated from a RatUp source row, DyadicRatCoreUp tolerance row, StreamNameUp finite-window row, RegSeqRatUp readback row, RealUp seal row, density-approximation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseRationalEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8733.1, Definition 8733.2, Definition 8733.3, and Definition 8733.4 seed the finite rational-to-real density handoff used by completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseRationalEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational source exposure, dyadic tolerance compatibility, stream-window exposure, regular rational readback, real sealing, density-row exactness, transport, replay, provenance, and local naming for BEDC.Derived.DenseRationalEmbeddingUp.

Chapter 8734

A Concrete Naming Certificate for $\mathsf{LocatedIntervalSeparation}^{\uparrow}$

Following Bishop and Bridges on located intervals and apartness-sensitive real construction, the $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ packet records a finite positive separation boundary before a Real or Completion consumer treats two inspected interval faces as apart. It sits over the located endpoint carrier of chapter 394, the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion route a finite separation row; it is not trichotomy, quotient real equality, a selected endpoint representative, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 8734.1 (Located interval separation carrier)**.** *A* $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (J_0, J_1, G, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J_0 and J_1 are the two* $\mathsf{LocatedInterval}^{\uparrow}$ *source rows whose exposed endpoint windows are being separated.*
2. *G is the dyadic positive-gap row. It displays a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance together with the endpoint-side choice saying which visible face of J_0 lies at least that gap before the corresponding visible face of J_1.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for positivity, subtraction, comparison, and slack retention used by the gap row.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the endpoint reads of J_0, the endpoint reads of J_1, and the gap-check observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two endpoint windows and the dyadic gap tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after J_0, J_1, G, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source intervals, positive gap, dyadic arithmetic, finite windows, readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each finite separation request through $J_0, J_1, G, D, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalSeparation}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J_0, J_1, G, D, W, R, E, H, C, P, N$. It has no coordinate for endpoint trichotomy, quotient real equality, selected representatives, countable choice, a host interval order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8734.2 (Located interval separation NameCert obligations)**.** *For every accepted* LocatedIntervalSeparatio *carrier*

$$S = (J_0, J_1, G, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalSeparation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *source rows, positive dyadic gap, dyadic arithmetic ledger, finite windows, regular readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *source-row exposure, requiring every interval endpoint read to be projected from J_0 or J_1 as in Definition 394.1;*
3. *positive-gap exactness, requiring G to display a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap and D to display the arithmetic route that witnesses the retained separation;*
4. *finite-window coverage, requiring every endpoint and gap-check observation to be named by W and read back by R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is consumed;*
5. *non-escape, requiring H, C, P, N to export no endpoint trichotomy, quotient real equality, selected representative, countable-choice schedule, host interval order, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No located interval separation obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8734.1. The displayed tuple $J_0, J_1, G, D, W, R, E, H, C, P, N$ is the whole packet surface.

The source-row clause first reads the two interval packets through the located interval carrier of chapter 394. Thus every endpoint row available to the separation test is already one of the finite endpoint-window rows of J_0 or J_1.

The positive-gap clause is then dyadic. Row G names the positive gap, and row D supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ positivity, subtraction, comparison, and slack data needed to replay the displayed separation check.

The window and readback rows control the handoff. Every endpoint observation and every gap-check observation passes through W, then through the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, before the terminal real-name seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name that same route. Since the carrier contains no further coordinate, the listed rows exhaust the local certificate surface and cannot export trichotomy, quotient real equality, selected representatives, countable choice, host interval order, or ambient Real completeness. □

Theorem 8734.3 (Located interval separation Real-completion handoff)**.** *Every Real-facing or Completion-facing read from an accepted* $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ *packet factors through two displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *source rows, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row, a finite* $\mathsf{StreamName}^{\uparrow}$ *window family, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any downstream consumer may cite the separation boundary.*

Consequently the handoff gives Real and Completion consumers a finite positive-gap boundary. It does not provide endpoint trichotomy, quotient real equality, selected representatives, countable choice, host interval order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8734.2. The obligations force a downstream read to begin with the two displayed source intervals J_0, J_1 and the displayed positive-gap row G.

The dyadic arithmetic is internal to the same packet. Row D supplies the positive-gap check, while W and R name and read back the finite endpoint and gap observations.

The terminal real-name seal E is therefore reached only after the finite separation route has been replayed. Transport H, replay C, provenance P, and the local certificate row N preserve that displayed route for Real and Completion consumers.

Because no carrier coordinate stores a host interval order, selected endpoint representative, quotient real equality, countable-choice schedule, endpoint trichotomy, or ambient completeness theorem, the handoff exports only the finite positive-gap boundary. □

Falsifiable prediction: Every accepted $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ Real-facing or Completion-facing read exposes two $\mathsf{LocatedInterval}^{\uparrow}$ source rows, one positive $\mathsf{DyadicRatCore}^{\uparrow}$ gap row, one dyadic arithmetic ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: locatedinterval, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ is not bijective to any listed sibling because it binds two located interval source rows to a positive dyadic gap before any StreamName, RegSeqRat, or Real row can be consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalSeparation}^{\uparrow}$ packet is generated from two LocatedInterval source rows, a positive DyadicRatCore gap row, dyadic arithmetic, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 8734.1, Definition 8734.2, and Definition 8734.3 seed the finite positive-gap boundary for LocatedInterval, DyadicRatCore, StreamName, RegSeqRat, Real, and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalSeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint trichotomy, quotient real equality, selected representatives, countable choice, host interval order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-row exposure, positive-gap exactness, finite-window coverage, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalSeparationUp.

Chapter 8735

A Concrete Naming Certificate for DeLaValleePoussinCriterion$^{\uparrow}$

Following de la Vallee Poussin and Dunford–Schwartz on uniform integrability, the DeLaValleePoussinCriterion$^{\uparrow}$ packet records the finite convex-growth route by which a family of integrable rows is admitted to uniform-integrability consumers. It sits after the uniform-integrability surface of chapter 6228, the integral ledger of chapter 123, the measure carrier of chapter 92, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names only displayed finite rows for a convex-growth test and its tail budget; it does not import measure completion, quotient equality almost everywhere, a selected dominating function, host convex analysis, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8735.1 (de la Vallee Poussin criterion carrier)**.** *A* DeLaValleePoussinCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, I, U, G, T, R, E, H, C, P, N)$$

with the following rows:

1. *M is the* Measure$^{\uparrow}$ *row carrying the displayed measurable source, finite family window, and inspected set budget.*
2. *I is the* Integral$^{\uparrow}$ *row recording the integrable-row reads for the family over M.*
3. *U is the* UniformIntegrability$^{\uparrow}$ *handoff row reached only after the growth and tail rows have been exposed.*
4. *G is the convex-growth row, displaying the increasing convex gauge, its finite thresholds, and the inspected growth values used by the criterion.*
5. *T is the tail-budget row, recording the finite tail thresholds and the exact budget by which large values are handed to U.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational budgets and growth thresholds.*
7. *E is the* Real$^{\uparrow}$ *seal row for the terminal real-valued comparison data.*
8. *H records componentwise* hsame *transport for $M, I, U, G, T, R, E, C, P, N$.*
9. *C records displayed* Cont *replay from the measure and integral rows through the growth, tail, readback, and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{DeLaValleePoussinCriterion}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{UniformIntegrability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DeLaValleePoussinCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, I, U, G, T, R, E, H, C, P, N$. It has no coordinate for a completed measure algebra, a quotient almost-everywhere class, a selected dominating function, an ambient family of all integrable functions, or host $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DeLaValleePoussinCriterionUp`

Theorem 8735.2 (de la Vallee Poussin NameCert obligations)**.** *For every accepted* $\mathsf{DeLaValleePoussinCriterion}^{\uparrow}$ *carrier*

$$D = (M, I, U, G, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DeLaValleePoussinCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the* $\mathsf{Measure}^{\uparrow}$ *row M, the* $\mathsf{Integral}^{\uparrow}$ *row I, and the family window inspected by the criterion;*
2. *growth-row exposure through G, requiring the convex gauge, finite thresholds, and growth reads to remain displayed before any uniform-integrability handoff;*
3. *tail-budget exactness through T, requiring each large-value tail read to carry its finite threshold and budget;*
4. *uniform-integrability handoff through U, requiring the public consumer to read* $\mathsf{UniformIntegrability}^{\uparrow}$ *only after M, I, G, T have remained visible;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal nonescape through R, E, requiring rational budget readback and terminal real comparison to stay tied to the same measure, integral, growth, and tail rows;*
6. *structural stability through H, C, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve exactly the rows M, I, U, G, T, R, E.*

No obligation may be discharged by measure completion, quotient almost-everywhere equality, a hidden dominating function, host convex-duality data, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8735.1. The measure and integral surface is M, I, the criterion data are G, T, U, the readback and seal are R, E, and the structural rows are H, C, P, N.

The growth row is exposed before the handoff row. Thus G supplies the displayed convex gauge and inspected thresholds used by the criterion, while T supplies the exact finite tail budget for large-value reads.

The uniform-integrability consumer receives U only through that displayed route. Since R and E read the rational budgets and real comparisons after M, I, G, T have remained visible, the terminal seal cannot be read from a completed measure algebra, quotient almost-everywhere class, selected dominating function, or ambient completeness principle.

Finally H, C, P, N transport, replay, source, and name the same finite rows. The listed obligations therefore exhaust the local NameCert surface. □

Theorem 8735.3 (de la Vallee Poussin uniform-integrability route)**.** *Every Vitali-facing or dominated-convergence-facing consumer of an accepted* $\mathsf{DeLaValleePoussinCriterion}^{\uparrow}$ *packet reads the criterion through the displayed route*

$$\mathsf{Measure}^{\uparrow}, \mathsf{Integral}^{\uparrow}, G, T \longmapsto \mathsf{UniformIntegrability}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Consequently the consumer sees a finite convex-growth row, an exact finite tail budget, regular rational readback, and a terminal real seal before any uniform-integrability conclusion is exported. It does not receive measure completion, quotient almost-everywhere equality, a host dominating function, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier of Definition 8735.1. The measure and integral rows M, I provide the source family, while the growth and tail rows G, T display the finite criterion data.

Apply the NameCert obligations of Definition 8735.2. The row U is visible only after the measure, integral, growth, and tail-budget rows have been read.

The consumer route then passes through R, E. These rows convert the finite budgets and growth thresholds into regular rational readback and a real-valued seal, so Vitali and dominated-convergence consumers receive only the finite $\mathsf{UniformIntegrability}^{\uparrow}$ handoff route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeLaValleePoussinCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DeLaValleePoussinCriterion}^{\uparrow}$ packet is generated from finite Measure rows, Integral rows, UniformIntegrability handoff rows, convex-growth rows, tail-budget rows, RegSeqRat readback rows, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DeLaValleePoussinCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8735.1, Definition 8735.2, and Definition 8735.3 seed the finite convex-growth and tail-budget route from Measure and Integral rows to UniformIntegrability, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DeLaValleePoussinCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full host de la Vallee Poussin theorem, measure completion, quotient almost-everywhere equality, selected domination, host convex analysis, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, growth-row exposure, tail-budget exactness, uniform-integrability handoff, Real-seal nonescape, transport stability, and replay stability for BEDC.Derived.DeLaValleePoussinCriterionUp.

Chapter 8736

A Concrete Naming Certificate for $\mathsf{DyadicBallBase}^{\uparrow}$

Following Bishop and Bridges and Kelley-style metric bases, the $\mathsf{DyadicBallBase}^{\uparrow}$ packet records the finite dyadic ball-basis surface that a BEDC completion or topology consumer may inspect before any ambient compactness, choice of centers, or Real completeness theorem is visible. It sits over the dyadic ball source of chapter 311, the dyadic rational ledger of chapter 291, the metric-ball comparison route of chapter 1450, the finite stream window of chapter 126, the regular rational readback of chapter 301, and the Real seal of chapter 38. The packet names only a finite basis row for inspected center-radius windows; it does not assert completion, separability, total boundedness, a selected center enumeration, or an ambient topology basis theorem.

Definition 8736.1 (Dyadic ball base carrier)**.** *A* $\mathsf{DyadicBallBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Y, Q, M, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *Y is the* $\mathsf{DyadicBall}^{\uparrow}$ *source row, exposing the finite center-radius ball request and its dyadic containment vocabulary.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *center and radius ledger, recording the inspected dyadic centers, positive dyadic radii, halved-radius comparisons, and finite refinement budgets.*
3. *M is the* $\mathsf{MetricBall}^{\uparrow}$ *comparison row, tying the dyadic ball request to the metric-ball inclusion and overlap checks that a later metric topology consumer may read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, displaying the finite name-window reads used to expose centers and endpoint requests without selecting an infinite enumeration.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row attached to the displayed StreamName windows and dyadic-radius ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected center and radius windows, reached only after Y, Q, M, W, S have been displayed.*
7. *H records componentwise* hsame *transport for the dyadic ball, dyadic rational ledger, metric-ball comparison, stream window, regular rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a dyadic-ball-basis request through $Y, Q, M, W, S, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicBallBase}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBallBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Y, Q, M, W, S, E, H, C, P, N$. It has no coordinate for a selected dense enumeration, an ambient open-set lattice, host compactness, Real completeness, quotient equality of names, or choice of centers.

Theorem 8736.2 (Dyadic ball base NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBallBase}^{\uparrow}$ *carrier*

$$B = (Y, Q, M, W, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicBallBase}^{\uparrow}}$ *surface consists exactly of five analytic rows followed by structural support: dyadic ball admission through Y; dyadic center-radius ledger exactness through Q; metric-ball comparison through M; finite StreamName window exposure through W; RegSeqRat tolerance readback through S; terminal Real sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8736.1. The analytic surface is the finite row sequence Y, Q, M, W, S, E, and the structural surface is H, C, P, N.

The dyadic source row is read first. It supplies the ball request, while the dyadic rational ledger records the center and radius data that make the request finite and checkable.

The metric comparison row then places the same center-radius request inside the metric-ball vocabulary. The StreamName and RegSeqRat rows expose only finite windows and tolerance readbacks, so the packet does not select an infinite basis enumeration.

Finally the Real seal and structural rows preserve the displayed sequence. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface and no ambient basis, compactness, completeness, quotient, or choice datum is exported. □

Theorem 8736.3 (Dyadic ball base finite radius stability)**.** *Let*

$$B = (Y, Q, M, W, S, E, H, C, P, N)$$

and

$$B' = (Y', Q', M', W', S', E', H', C', P', N')$$

be accepted $\mathsf{DyadicBallBase}^{\uparrow}$ *carriers. If Y, Y', Q, Q', M, M', W, W', S, S', and E, E' are componentwise* hsame *as dyadic ball rows, dyadic center-radius ledgers, metric-ball comparison rows, StreamName windows, RegSeqRat readbacks, and Real seals, then every finite basis read exported through H, C, P, N and H', C', P', N' displays the same center-radius window and the same metric-ball comparison route.*

Proof. Project both accepted carriers. The only rows that can feed a basis read are Y, Q, M, W, S, E, with H, C, P, N and H', C', P', N' only transporting, replaying, sourcing, and naming the same packet shapes.

Apply the NameCert obligation surface of Definition 8736.2. It forces every consumer read through the dyadic ball source, dyadic rational center-radius ledger, metric-ball comparison, finite window, tolerance readback, and Real seal.

The componentwise hsame assumptions identify each analytic row in the two packets. Replay through C and C' therefore follows the same finite sequence, and transport through H and H'

preserves the displayed center-radius and comparison data. Provenance and naming add no new coordinate.

Thus the exported basis read is stable under the displayed finite radius data. The theorem does not assert an ambient topology basis, a chosen dense sequence, a compact net, Real completeness, or quotient equality of names. □

Theorem 8736.4 (Dyadic ball base non-escape)**.** *Every completion, metric-topology, or compact-net consumer of an accepted* $\mathsf{DyadicBallBase}^{\uparrow}$ *packet must read the route*

$$Y \to Q \to M \to W \to S \to E$$

before using the structural rows H, C, P, N*. The consumer receives a finite dyadic center-radius basis surface and cannot read a hidden modulus selector, selected dense enumeration, ambient open-set lattice, quotient stream equality, selected limit, host compactness proof, or Real completeness theorem from the packet.*

Proof. Start with the carrier and NameCert obligation surface, Definition 8736.1 and Definition 8736.2. They expose exactly the dyadic ball source, dyadic rational ledger, metric-ball comparison, finite window, tolerance readback, Real seal, and structural rows.

The route to a completion or topology consumer is ordered. The dyadic ball request is not public without the center-radius ledger, the metric comparison is not public before that ledger, and the window and tolerance rows are the only way the Real seal is reached.

Finally apply finite radius stability, Definition 8736.3. It shows that consumers can preserve the displayed center-radius window and metric comparison route, but only through the same finite rows. Since the carrier has no coordinate for the excluded selector, enumeration, lattice, quotient, limit, compactness, or completeness objects, no consumer can read them from the packet. □

Falsifiable prediction: Every accepted $\mathsf{DyadicBallBase}^{\uparrow}$ read exposes a $\mathsf{DyadicBall}^{\uparrow}$ source row, a $\mathsf{DyadicRatCore}^{\uparrow}$ center-radius ledger, a $\mathsf{MetricBall}^{\uparrow}$ comparison row, a $\mathsf{StreamName}^{\uparrow}$ finite window, a $\mathsf{RegSeqRat}^{\uparrow}$ tolerance readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicball, dyadicratcore, metricball, streamname, regseqrat, real: $\mathsf{DyadicBallBase}^{\uparrow}$ is not bijective to any listed sibling because it binds dyadic ball requests, dyadic center-radius ledgers, metric-ball comparisons, finite stream windows, regular rational tolerance readbacks, and Real seals into one finite basis packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBallBase}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBallBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8736.1, Definition 8736.2, Definition 8736.3, and Definition 8736.4 seed the finite dyadic ball-basis surface over dyadic ball, DyadicRatCore, MetricBall, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBallBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, compactness, total boundedness, selected dense enumerations, quotient equality of names, ambient topology-basis theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic ball admission, center-radius ledger exactness, metric-ball comparison, finite window exposure, tolerance readback, Real seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.DyadicBallBaseUp.

Constructive story: A $\mathsf{DyadicBallBase}^{\uparrow}$ packet is generated from a DyadicBall source row, a DyadicRatCore center-radius ledger, a MetricBall comparison row, a StreamName window row, a RegSeqRat tolerance readback row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8737

A Concrete Naming Certificate for $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$

Following Egoroff and Bishop–Bridges constructive analysis, the $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ packet records the finite-measure convergence neighbour in which pointwise tail data are converted into a displayed uniform-tail window after paying an explicit exceptional-measure budget. It sits after the root measure rows of chapter 92, the finite event-value substrate of chapter 6127, the finite $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records finite exceptional sets, finite tail thresholds, and uniform-window handoff rows only; it does not assert a full almost-everywhere quotient, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8737.1 (Egoroff finite-measure carrier)**.** *An* $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, F, S, B, W, R, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, exposing the measurable domain and event-family route inspected by the convergence packet.*
2. *F is the* $\mathsf{FiniteMeasure}^{\uparrow}$ *row, giving the finite total-mass bound and the finite event-value ledger for exceptional sets.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *pointwise-observation row, listing the finite stream probes and tail requests read from the converging family.*
4. *B is the exceptional-budget row, assigning each requested precision a displayed finite event whose* $\mathsf{FiniteMeasure}^{\uparrow}$ *value is below the requested rational tolerance.*
5. *W is the uniform-tail window row, recording the finite tail index after the exceptional event has been removed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to compare displayed values on the accepted tail window.*
7. *U is the uniform-window handoff row, binding B, W, R to the same measure source and stream observations.*
8. *H records componentwise* hsame *transport for the measure, finite-measure, stream, budget, window, readback, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a convergence request through $M, F, S, B, W, R, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{FiniteMeasure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EgoroffFiniteMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, S, B, W, R, U, H, C, P, N$. It has no coordinate for a countable exceptional-set union, quotient almost-everywhere equality, selected infinite subsequence, host measurable-function object, or ambient real-completeness principle.

Theorem 8737.2 (Egoroff finite-measure NameCert obligations)**.** *For every accepted* $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ *carrier*

$$E = (M, F, S, B, W, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EgoroffFiniteMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure source, finite-measure ledger, finite stream observations, exceptional-measure budget, uniform-tail window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, uniform-window handoff, componentwise transport, displayed replay, provenance, and local naming rows. No obligation is stored outside $M, F, S, B, W, R, U, H, C, P, N$.*

Proof. Project the carrier of Definition 8737.1. The analytic rows are M, F, S, B, W, R, U, while H, C, P, N are the structural certificate rows.

The measure source and finite-measure ledger are admitted first. The exceptional-budget row B can therefore speak only about displayed finite events and their recorded finite values, not about a hidden countable union or completed null ideal.

The stream row S supplies the inspected pointwise tails. The rows W and R then expose a finite tail index and a regular rational readback for the accepted request.

Finally U, H, C, P, N bind that budget and window to one public route. Since no excluded object is a carrier coordinate, none is a NameCert obligation of the packet. □

Theorem 8737.3 (Egoroff finite-measure uniform-window handoff)**.** *Every convergence consumer of an accepted* $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ *carrier must factor through the displayed route*

$$M, F, S \longmapsto B \longmapsto W, R \longmapsto U, H, C, P, N.$$

Thus the only public handoff is the finite BEDC route from a $\mathsf{Measure}^{\uparrow}$ *source and* $\mathsf{FiniteMeasure}^{\uparrow}$ *event-value ledger through* $\mathsf{StreamName}^{\uparrow}$ *observations, an exceptional-measure budget, a uniform-tail window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no almost-everywhere quotient, countable-choice selector, infinite exceptional-set union, hidden host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8737.2. They place the measure source M, finite-measure ledger F, and stream observations S before any uniform conclusion is visible.

The exceptional-budget row B is downstream of the finite-measure ledger. It records a displayed finite event and a displayed tolerance comparison for the current request, so the handoff cannot appeal to a countable null-set construction.

The uniform-tail window W and readback row R are then inspected on the remaining displayed route. The handoff row U ties the same budget, window, and readback to the source rows M, F, S.

Transport, replay, provenance, and local naming are carried by H, C, P, N. These rows preserve the same finite route and do not create an independent quotient or completeness boundary. □

Falsifiable prediction: Every accepted $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ read must expose a MeasureUp source, FiniteMeasureUp event-value ledger, finite StreamName observations, exceptional-measure budget, uniform-tail window, RegSeqRat readback, uniform-window handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: measure, finite-measure, streamname, regseqrat, real: $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds exceptional finite-measure budgets to uniform-tail windows and regular rational readback in one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{EgoroffFiniteMeasure}^{\uparrow}}$.

Constructive story: An $\mathsf{EgoroffFiniteMeasure}^{\uparrow}$ packet is generated from a MeasureUp source row, a FiniteMeasureUp event-value ledger, finite StreamName observations, exceptional-measure budget rows, uniform-tail windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EgoroffFiniteMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8737.1, Definition 8737.2, and Definition 8737.3 seed the finite-measure Egoroff route from displayed exceptional budgets to uniform-tail windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EgoroffFiniteMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, countable exceptional-set unions, full almost-everywhere quotients, selected subsequences, host measurable-function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite event-value exactness, exceptional-budget readback, uniform-tail window handoff, RegSeqRat and Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EgoroffFiniteMeasureUp.

Chapter 8738

A Concrete Naming Certificate for $\mathsf{DyadicApartnessScale}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicApartnessScale}^{\uparrow}$ packet records the finite dyadic scale by which constructive real apartness is witnessed before any trichotomy or quotient real equality is available. It sits after the dyadic rational carrier of chapter 291, the regular rational readback of chapter 301, the located apartness route of chapter 7144, the RealMetric boundary of chapter 1638, and the terminal Real certificate of chapter 38. The packet is a finite scale witness for separation reads; it is not a comparison oracle, a selected real representative, an ambient completeness theorem, or a host equality principle.

Definition 8738.1 (Dyadic apartness scale carrier)**.** *A* $\mathsf{DyadicApartnessScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, R, L, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, carrying the positive dyadic tolerance and the two dyadic rational comparison endpoints inspected by the apartness read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the finite regular rational windows whose distance is compared against the dyadic scale.*
3. *L is the located-apartness row, recording that the apartness witness is displayed as positive separation data rather than as trichotomy.*
4. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *metric row, naming the finite distance estimate consumed by separation-facing Real readers.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, R, L, M have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic scale, regular readback, located-apartness, metric, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each Real-facing separation request through D, R, L, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicApartnessScale}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *located apartness,* $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicApartnessScale}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, R, L, M, E, H, C, P, N$. It has no coordinate for real trichotomy, quotient real equality, selected limits, host representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 8738.2 (Dyadic apartness scale NameCert obligations)**.** *For every accepted* $\mathsf{DyadicApartnessScale}^{\uparrow}$ *carrier*

$$A = (D, R, L, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicApartnessScale}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *dyadic-scale admission for the positive dyadic tolerance and inspected dyadic rational endpoints carried by D;*
2. *regular-window exposure requiring each separation read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the Real seal is consumed;*
3. *located-apartness exposure requiring L to display positive separation data and to refuse trichotomy as a public primitive;*
4. *metric handoff through M, recording only the finite distance estimate needed by RealMetric consumers;*
5. *Real-seal non-escape through E, followed by componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

No accepted obligation row stores quotient real equality, selected limits, host real representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or checked bridge data.*

Proof. Carrier admission is projection from Definition 8738.1. The tuple $A = (D, R, L, M, E, H, C, P, N)$ is the whole surface exposed to a separation-facing consumer.

The dyadic-scale and regular-window clauses force the apartness read to start from D and R. Thus the consumer sees a positive dyadic tolerance and finite regular rational windows, not an infinite representative or a quotient class.

The located-apartness and metric clauses then route the same data through L and M. The witness is positive separation at a displayed scale, and the RealMetric row records only the finite estimate needed for that separation.

Finally, E, H, C, P, N seal, transport, replay, package, and name the same finite rows. Since the carrier contains no coordinate for trichotomy, quotient equality, selected limits, choice, ambient completeness, or bridge checking, those objects cannot be exported by the NameCert surface. □

Theorem 8738.3 (Dyadic apartness scale Real separation handoff)**.** *Every Real-facing separation read of an accepted* $\mathsf{DyadicApartnessScale}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \textit{located apartness} \longrightarrow \mathsf{RealMetric}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by D, R, L, M, E. The handoff may witness that two Real reads are apart at a dyadic scale, but it does not decide equality, decide order, choose representatives, select limits, or supply ambient Real completeness.

Proof. Read the separation request through the dyadic scale row D and the finite regular windows in R. The only available comparison datum is a positive dyadic tolerance attached to displayed rational approximants.

Next pass the same datum through the located-apartness row L and the RealMetric row M. These rows state the finite positive-separation evidence consumed by the Real-facing read.

The terminal Real seal E is reached only after those rows are present, and the structural rows H, C, P, N replay exactly the same packet. Therefore the handoff supplies scale-indexed apartness evidence, not equality decision, order trichotomy, representative selection, selected limits, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{DyadicApartnessScale}^{\uparrow}$ consumer must display a DyadicRatCore scale row, a RegSeqRat finite-window readback row, a located-apartness row, a RealMetric estimate row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before Real-facing separation can be consumed.

Independence witness: dyadicratcore, regseqrat, located-apartness-field, realmetric, real: no listed sibling supplies the joint scale packet containing dyadic tolerance data, regular rational windows, located-apartness evidence, metric estimate, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicApartnessScale}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicApartnessScale}^{\uparrow}$ packet is generated from a DyadicRatCore scale row, RegSeqRat finite-window readback, located-apartness evidence, RealMetric estimate, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicApartnessScale}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8738.1, Definition 8738.2, and Definition 8738.3 seed the finite dyadic scale route for constructive Real apartness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicApartnessScale}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic-scale admission, regular-window exposure, located-apartness exposure, metric handoff, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicApartnessScaleUp.

Chapter 8739

A Concrete Naming Certificate for MetaCICResidualLocalConfluenceMatrix$^{\uparrow}$

The MetaCICResidualLocalConfluenceMatrix$^{\uparrow}$ packet realizes the bounded residual local-join frontier of chapter 9378 as finite matrix data. It sits beside the residual checker of chapter 3827, the residual local-join determinacy packet of chapter 5970, the residual-substitution boundary of chapter 2181, the parallel-diamond frontier of chapter 4238, and the confluence audit packet of chapter 735. The matrix records typed windows, residual pairs, bounded checkers, obstruction retention, transport, replay, provenance, and local naming before any confluence consumer may cite a local join.

Definition 8739.1 (MetaCIC residual local-confluence matrix carrier)**.** *A* MetaCICResidualLocalConfluenceMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, R, J, B, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row, naming the finite MetaCIC source terms and contexts inspected by the matrix.*
2. *R is the residual-pair row, recording the bounded residual branch pairs whose local join status is under inspection.*
3. *J is the candidate local-join row, displaying only the finite common-reduct candidates admitted by the checker.*
4. *B is the bounded-checker row, recording the finite checker outcome for each displayed residual pair.*
5. *O is the obstruction row, retaining residual-substitution compatibility, unrestricted closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, and kernel consistency as visible non-exports.*
6. *H records componentwise* hsame *transport for typed windows, residual pairs, candidate joins, checker outcomes, obstruction rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from each confluence-facing request through T, R, J, B, O, H, P, N.*
8. *P is the* Pkg *provenance over* MetaCICResidualLocalConfluenceMatrix$^{\uparrow}$*, the residual checker, residual local-join determinacy, residual-substitution boundary, parallel-diamond frontier, confluence audit packet,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, R, J, B, O, H, C, P, N$. It has no coordinate for unrestricted substitution compatibility, arbitrary closed parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, kernel consistency, or bridge checking.

Theorem 8739.2 (MetaCIC residual local-confluence matrix NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}$ *carrier*

$$M = (T, R, J, B, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}}$ *surface consists exactly of typed-window exposure, residual-pair exposure, candidate local-join exposure, bounded-checker exactness, obstruction retention, componentwise transport, displayed replay, provenance, and local naming. Any accepted confluence-facing consumer must expose $T, R, J, B, O, H, C, P, N$ before treating a residual local join as available to the confluence audit route.*

Proof. Carrier admission is projection from Definition 8739.1. The typed windows, residual pairs, candidate joins, checker outcomes, and obstruction rows are all explicit coordinates of M.

The local-join read first displays T and R, so the matrix is tied to a bounded typed residual frontier. It then displays J and B, so candidate joins are available only through the finite checker row.

The obstruction row O remains visible after the checker outcome. Thus any missing residual-substitution compatibility or closed-diamond premise is retained as data rather than silently discharged.

Finally, H, C, P, N transport, replay, package, and name the same finite rows. Since no carrier coordinate stores an unrestricted confluence theorem, the NameCert surface cannot export arbitrary closed parallel diamond or Church–Rosser. □

Theorem 8739.3 (MetaCIC residual local-confluence matrix budget handoff)**.** *Every bounded local-join handoff exported by an accepted* $\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}$ *carrier factors through the finite route*

$$T \longrightarrow R \longrightarrow J \longrightarrow B \longrightarrow O$$

inside $M = (T, R, J, B, O, H, C, P, N)$. The route may pass checked residual-pair rows to the confluence audit map, but it must retain the obstruction row whenever residual-substitution compatibility, unrestricted closed parallel diamond, or arbitrary closed Church–Rosser has not been supplied.

Proof. The typed-window and residual-pair rows bound the source of every matrix entry. A handoff cannot name a candidate join until those finite source rows have been displayed.

The candidate local-join and bounded-checker rows then identify which entries passed the finite checker. These rows are budgeted by B, not by a global proof search or an unrestricted confluence theorem.

The obstruction row O remains part of the exported route. Therefore the handoff can support bounded local joins while still refusing arbitrary closed parallel diamond, Church–Rosser, unchecked substitution compatibility, subject reduction, evaluator equality, and kernel consistency. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}$ consumer that cites a residual local join without first displaying typed-window, residual-pair, candidate local-join,

bounded-checker, obstruction, transport, replay, provenance, and local NameCert rows refutes the matrix surface.

Independence witness: metacic-confluence-residual-checker, metacic-residual-join-determinacy, metacic-residual-substitution-boundary, metacic-parallel-diamond-frontier, metacic-confluence-audit-packet: no listed sibling jointly carries typed windows, residual pairs, candidate local joins, bounded checker outcomes, obstruction retention, transport, replay, provenance, and local naming as one finite matrix packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}$ packet is generated from typed-window rows, residual-pair rows, candidate local-join rows, bounded-checker rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8739.1, Definition 8739.2, and Definition 8739.3 seed the finite matrix route for bounded residual local joins.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualLocalConfluenceMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, arbitrary closed parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, residual-pair exposure, candidate local-join exposure, bounded-checker exactness, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICResidualLocalConfluenceMatrixUp.

Chapter 8740

A Concrete Naming Certificate for $\mathsf{RegularCauchyRealizer}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real numbers, the $\mathsf{RegularCauchyRealizer}^{\uparrow}$ packet records the finite readback route by which regular Cauchy data answers precision requests before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits beside the general realizer route of chapter 1771, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real carrier seal of chapter 38. The packet names a choice-free L10 Real and Completion readback surface; it is not quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8740.1 (Regular Cauchy realizer carrier)**.** *A* $\mathsf{RegularCauchyRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the RegularCauchy source row, recording the request-to-tail discipline for the regular Cauchy name consumed by the realizer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which each precision request is inspected.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values visible in the selected window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic bound against which the readback is checked.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the source, window, readback, and tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for A, W, Q, D, E, C, P, N.*
7. *C records displayed* Cont *replay of the same request-to-window route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRealizer}^{\uparrow}$, $\mathsf{CauchyRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, Q, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, ambient completeness, countable choice, trichotomy, or host real equality.

Theorem 8740.2 (Regular Cauchy realizer NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRealizer}^{\uparrow}$ *carrier* $R = (A, W, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the RegularCauchy source row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer route must expose the same readback chain used by* $\mathsf{CauchyRealizer}^{\uparrow}$ *before it may cite the Real-facing seal.*

Proof. Project the carrier of Definition 8740.1. Its rows are exactly $A, W, Q, D, E, H, C, P, N$, so local naming has no source outside the finite packet.

The regular Cauchy source A answers a precision request only through the finite observation window W. The observed values are then read as regular rational data through Q and checked against the dyadic tolerance row D.

The terminal Real seal E is downstream of those rows. Transport H, replay C, provenance P, and local naming N preserve and name the same route. Since no quotient, selector, choice, trichotomy, or ambient-completeness coordinate occurs in the carrier, these rows exhaust the NameCert obligations. □

Theorem 8740.3 (Regular Cauchy realizer Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RegularCauchyRealizer}^{\uparrow}$ *carrier factors through the finite route*

$$A \to W \to Q \to D \to E \to H \to C \to P \to N.$$

Thus a Real or Completion consumer reads the RegularCauchy source, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, terminal Real seal, componentwise transport, replay, provenance, and local* NameCert *rows before it treats the packet as a realizer. The route does not claim completeness of* $\mathsf{Real}^{\uparrow}$ *or equality with a host quotient real.*

Proof. Open the exported Real seal row E. By Definition 8740.1, that row is available only inside the same finite packet as the RegularCauchy source, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance rows.

Apply Definition 8740.2. It forces every consumer read to pass through A, W, Q, D before the seal and then through transport, replay, provenance, and local naming.

The sibling $\mathsf{CauchyRealizer}^{\uparrow}$ chapter supplies the general realizer pattern, while this packet narrows the read to regular Cauchy data. Because neither carrier contains host quotient equality, selected limits, countable choice, or ambient Real completeness, the displayed row route is the full Real-seal route. □

Falsifiable prediction: Every accepted RegularCauchyRealizerUp Real or Completion read exposes a RegularCauchy source row, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance row, terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a consumer receives the seal.

Independence witness: cauchy-realizer, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds regular Cauchy source data, finite windows, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming while refusing quotient equality and selected limits.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRealizer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyRealizer}^{\uparrow}$ packet is generated from a RegularCauchy source row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchyRealizer}\uparrow}$, seed)

Scope closed: Definition 8740.1, Definition 8740.2, and Definition 8740.3 seed the finite regular Cauchy readback route into the Real seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyRealizer}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, dyadic tolerance accounting, Real-seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyRealizerUp.

Chapter 8741

A Concrete Naming Certificate for $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$ packet records the finite partition and mesh-modulus surface by which a located interval is handed to dyadic and real-name consumers. It sits after the located interval boundary of chapter 394, the dyadic interval net of chapter 5966, the dyadic rational core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a named finite-means partition/modulus object; it is not host interval equality, selected representatives, trichotomy, choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8741.1 (Located interval modulus partition carrier)**.** *A* $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, U, A, Q, D, K, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the located interval endpoint rows inherited from* $\mathsf{LocatedInterval}^{\uparrow}$*.*
2. *A is the locatedness and endpoint-order row, binding the endpoints before any interval partition is read.*
3. *Q is the finite dyadic partition-cell row supplied by* $\mathsf{DyadicIntervalNet}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to express cell endpoints, mesh bounds, and outward rounding budgets.*
5. *K is the mesh-modulus row, recording the finite precision request that bounds the displayed partition cells.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint and cell reads at the displayed mesh.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic and stream-window observations.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for partition and endpoint consumers.*
9. *H records componentwise* hsame *transport for endpoint, locatedness, partition, tolerance, modulus, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same partition/modulus route.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, A, Q, D, K, W, R, S, H, C, P, N$. It has no coordinate for host interval equality, selected endpoint or cell representatives, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8741.2 (Located interval modulus partition carrier obligations)**.** *For every accepted* $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$ *carrier*

$$M = (L, U, A, Q, D, K, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusPartition}^{\uparrow}}$ *surface exposes exactly located interval endpoints, finite dyadic partition cells, mesh-modulus data,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every partition read must pass through $A, Q, D, K, W, R, S, H, C, P, N$ after the endpoints are displayed, and the packet exports no host interval equality, selected representatives, trichotomy, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8741.1. Its rows are exactly $L, U, A, Q, D, K, W, R, S, H, C, P, N$, so accepted packets expose endpoint, locatedness, partition, tolerance, modulus, window, readback, seal, transport, replay, provenance, and naming data and no further host interval coordinate.

The finite partition route begins with the located interval endpoint rows and the locatedness row A. The dyadic cells Q are then read through the tolerance row D and mesh-modulus row K, which fixes the finite precision at which the partition is visible.

The real-name handoff is downstream of the finite rows. Stream windows W and regular rational readback R mediate the dyadic observations before the terminal seal S can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no row for host interval equality, selected representatives, trichotomy, choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those endpoints is exported. □

Falsifiable prediction: Every accepted $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$ consumer that reads a partition cell must expose located endpoints, dyadic partition cells, mesh modulus, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows before exporting the read.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalModulusPartition}^{\uparrow}$ packet is generated from located interval endpoints, an endpoint-order row, finite dyadic partition cells, a dyadic tolerance row, a mesh-modulus row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusPartition}^{\uparrow}}$, seed)

Scope closed: Definition 8741.1 and Definition 8741.2 seed the finite partition/modulus route from located intervals to dyadic nets and regular real readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalModulusPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected representatives, trichotomy, choice, compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order preservation, dyadic partition-cell visibility, mesh-modulus readback, StreamName windows, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalModulusPartitionUp.

Chapter 8742

A Concrete Naming Certificate for PerfectlyNormalSpace$^{\uparrow}$

Following Engelking and Munkres on perfectly normal spaces, the PerfectlyNormalSpace$^{\uparrow}$ packet records the classical separation surface where every displayed closed row is presented as a zero set and every displayed open row is presented as the complementary cozero set. It sits above the topology carrier of chapter 88, the closed-pair separation route of chapter 1866, the separation-function route of **??**, the zero/cozero boundary of chapter 1858 and chapter 1857, and the continuous-map surface of chapter 130. The packet names only finite displayed rows for this traditional point-set condition; it does not import host powersets, arbitrary function spaces, selected complements, quotient equality of functions, or ambient real completeness.

Definition 8742.1 (Perfectly normal space carrier)**.** *A* PerfectlyNormalSpace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, N, F, Z, O, U, M, H, C, K, L)$$

with the following displayed rows:

1. *T is the* Topology$^{\uparrow}$ *source row, carrying the open-set and closed-set observation surface.*
2. *N is the* NormalSpace$^{\uparrow}$ *row, exposing the closed-pair neighborhood separation route already named in chapter 1866.*
3. *F is the finite ledger of displayed closed rows that the packet will expose as zero sets.*
4. *Z assigns to each closed row in F a* ZeroSet$^{\uparrow}$ *witness and the corresponding continuous real-valued function row.*
5. *O is the open/cozero handoff row, recording that the complement of each displayed closed row in F is consumed as a* CozeroSet$^{\uparrow}$ *row.*
6. *U is the* UrysohnLemma$^{\uparrow}$ *boundary row, tying the zero-set and cozero-set witnesses to the separation-function route of* **??**.
7. *M is the* ContinuousMap$^{\uparrow}$ *row for the displayed real-valued functions used by Z and O.*
8. *H records componentwise* hsame *transport for the topology, normal-space, closed-row, zero-set, cozero-set, Urysohn-boundary, continuous-map, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a closed-row request through T, N, F, Z, O, U, M.*

10. *K is the* Pkg *provenance over* PerfectlyNormalSpace$^\uparrow$, Topology$^\uparrow$, NormalSpace$^\uparrow$, ZeroSet$^\uparrow$, CozeroSet$^\uparrow$, UrysohnLemma$^\uparrow$, ContinuousMap$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{PerfectlyNormalSpace}^\uparrow}$ *row.*

The carrier contains no coordinate for a host powerset of closed subsets, arbitrary complement selection, unrestricted function-space equality, choice over separating maps, or ambient Real$^\uparrow$ *completeness.*

Theorem 8742.2 (Perfectly normal space NameCert obligations)**.** *For every accepted* PerfectlyNormalSpace$^\uparrow$ *carrier*

$$P = (T, N, F, Z, O, U, M, H, C, K, L),$$

the local NameCert$_{\text{PerfectlyNormalSpace}^\uparrow}$ *surface consists exactly of the finite rows that admit the topology source, the normal-space closed-pair route, the displayed closed rows, their zero-set witnesses, the complementary cozero-set handoff, the Urysohn function boundary, the continuous-map rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local naming. Every public read of perfect normality must pass through these rows before exporting a perfectly normal space name.*

Proof. Carrier admission is projection from Definition 8742.1. The accepted packet displays the topology source T, normal-space route N, closed-row ledger F, zero-set witnesses Z, cozero handoff O, Urysohn boundary U, continuous-map row M, and the structural rows H, C, K, L.

Closed-as-zero-set exposure is read through F and Z. Each displayed closed row has its zero-set witness before any consumer may treat the closed row as part of the perfectly normal surface.

The open/cozero handoff is read through O. Since O is attached to the same closed rows and the same continuous-map data, an open-row consumer receives only the displayed complementary cozero route, not an ambient complement selector.

The Urysohn and Tietze-facing nonescape condition is enforced by U and M. A downstream separation-function or extension reader must pass through the normal-space row of chapter 1866, the Urysohn route of **??**, and the displayed ContinuousMap$^\uparrow$ row before using the zero/cozero certificate.

Finally the componentwise hsame row, displayed Cont replay, provenance, and local naming rows preserve this finite route. They provide stability, source accounting, and local naming, while adding no host powerset, choice principle, quotient function equality, or ambient completeness datum. □

Closure status for NameCert$_{\text{PerfectlyNormalSpace}^\uparrow}$.

Constructive story: A PerfectlyNormalSpace$^\uparrow$ packet is generated from a Topology$^\uparrow$ source row, a NormalSpace$^\uparrow$ closed-pair route, displayed closed rows, ZeroSet$^\uparrow$ witnesses, complementary CozeroSet$^\uparrow$ rows, a UrysohnLemma$^\uparrow$ boundary row, ContinuousMap$^\uparrow$ real-valued function rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PerfectlyNormalSpace}^\uparrow}$, seed)

Scope closed: Definition 8742.1 and Definition 8742.2 name the finite carrier and seed obligation surface for closed-as-zero-set and open-as-cozero reads over Topology$^\uparrow$, NormalSpace$^\uparrow$, ZeroSet$^\uparrow$, CozeroSet$^\uparrow$, UrysohnLemma$^\uparrow$, ContinuousMap$^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{PerfectlyNormalSpace}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim checked Lean realization, public closure, Tietze extension closure, quotient equality of functions, host powersets, choice over separators, or a completed bridge.

Upgrade path: Obligation closure requires checked carrier admission, closed-as-zero-set exposure, open/cozero handoff, Urysohn and continuous-map boundary stability, transport stability, provenance, and local naming for BEDC.Derived.PerfectlyNormalSpace

Chapter 8743

A Concrete Naming Certificate for $\mathsf{PerronMethodFiniteBarrier}^{\uparrow}$

Following Perron and the classical barrier method for the Dirichlet problem, the $\mathsf{PerronMethodFiniteBarrier}^{\uparrow}$ packet records a finite BEDC surface for local boundary barriers used by a Perron upper-envelope route. It sits between the integral precedent of chapter 6378, the endpoint and order rows of chapter 38, and the metric-facing boundary data of chapter 57. The packet is a finite barrier ledger; it does not select a harmonic solution, solve the full Dirichlet problem, complete a measure space, quotient real equality, choose from a family of candidates, or assert ambient completeness.

Definition 8743.1 (Perron finite barrier carrier)**.** *A* $\mathsf{PerronMethodFiniteBarrier}^{\uparrow}$ *carrier is a finite tuple*

$$B = (\Omega, \partial\Omega, x_0, U, R, M, W, L, H, C, P, N)$$

whose rows have the following displayed meanings.

1. Ω *is the inspected domain row, and* $\partial\Omega$ *is the finite boundary-window row.*
2. x_0 *is the boundary point under inspection, with all endpoint comparisons read through* $\mathsf{Real}^{\uparrow}$ *rows.*
3. U *is a finite neighborhood ledger around* x_0*.*
4. R *is the ordered real-value row used for barrier inequalities.*
5. M *is the metric-distance row that relates* U *and the boundary window.*
6. W *is the finite barrier-witness list.*
7. L *is the Perron ledger recording which displayed upper and lower comparison rows are available.*
8. H, C, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{PerronMethodFiniteBarrier}^{\uparrow}}$ *rows.*

The carrier contains no coordinate for a selected harmonic function, no completed boundary measure, no quotient real equality, no countable choice selector, and no ambient complete metric space.

Theorem 8743.2 (Perron finite barrier NameCert obligations)**.** *For an accepted* $\mathsf{PerronMethodFiniteBarrier}^{\uparrow}$ *packet, the local* $\mathsf{NameCert}$ *surface consists exactly of carrier admission for the domain row, boundary-window row, inspected boundary point, finite neighborhood ledger, ordered* $\mathsf{Real}^{\uparrow}$ *value row,* $\mathsf{Met}^{\uparrow}$ *distance row, finite barrier-witness list, Perron comparison ledger, componentwise transport, displayed replay, provenance, and local naming rows. Every barrier comparison exported by*

the packet must read a witness from W, an order comparison from R, and a neighborhood-distance request from M and U before it can be used by a downstream Perron or boundary-value consumer.

Proof. Project the carrier of Definition 8743.1. The available rows are precisely $\Omega, \partial\Omega, x_0, U, R, M, W, L, H, C, P, N$, so every accepted read begins with the displayed domain, boundary, point, neighborhood, real-order, metric, witness, and comparison-ledger data.

The NameCert row N names only those projected rows. The transport row H preserves them componentwise under hsame, the replay row C replays the displayed comparison requests, and the provenance row P records the finite source of each witness.

Thus no consumer can replace the finite witness list by a selected harmonic solution, a boundary measure completion, quotient equality of real values, choice over candidate functions, or ambient completeness. The packet exports exactly the finite barrier data listed in the statement. □

Theorem 8743.3 (Perron finite barrier harmonic route)**.** *Every boundary-value consumer that reads a* PerronMethodFiniteBarrier$^{\uparrow}$ *packet toward a harmonic Perron route must first pass through the finite barrier NameCert obligations of Definition 8743.2. Consequently the route can expose a local barrier inequality and a Perron comparison ledger, but it cannot claim a selected harmonic solution, the full Dirichlet problem, measure completion, quotient real equality, choice, or ambient completeness.*

Proof. Start with the NameCert obligations. They force the consumer to inspect the finite boundary window, the boundary point, the neighborhood ledger, the real-order row, the metric-distance row, the finite barrier-witness list, and the Perron comparison ledger before any harmonic-facing read is exported.

The Perron comparison ledger may then be passed to downstream chapters in the same sense that chapter 6378 passes finite Perron data through an integration route, while the Real$^{\uparrow}$ and Met$^{\uparrow}$ rows keep endpoint and distance evidence explicit. This supplies a constructive barrier route rather than a solved boundary-value problem.

Because the carrier has no coordinate for a chosen harmonic function, completed measure, quotient equality, choice selector, or complete ambient space, none of those objects can be produced by this route. The exported data are exactly the local barrier inequality and its finite comparison provenance. □

Falsifiable prediction: Every accepted PerronMethodFiniteBarrier$^{\uparrow}$ read must expose a finite boundary window, inspected boundary point, neighborhood ledger, ordered Real$^{\uparrow}$ value row, Met$^{\uparrow}$ distance row, finite barrier-witness list, Perron comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any harmonic-facing consumer can use it.

Independence witness: perron-integral, real, metric: no listed sibling jointly carries the finite boundary-window, local barrier-witness, Perron comparison-ledger, real-order, metric-distance, transport, replay, provenance, and local NameCert rows of $B = (\Omega, \partial\Omega, x_0, U, R, M, W, L, H, C, P, N)$.

Closure status for NameCert$_{\text{PerronMethodFiniteBarrier}^{\uparrow}}$.

Constructive story: A PerronMethodFiniteBarrier$^{\uparrow}$ packet is generated from domain and boundary-window rows, an inspected boundary point, finite neighborhood rows, Real order rows, Metric distance rows, a finite barrier-witness list, a Perron comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PerronMethodFiniteBarrier}^{\uparrow}}$, seed)

Scope closed: Definition 8743.1, Definition 8743.2, and Definition 8743.3 seed the finite barrier packet for Perron-method boundary consumers.

Formal verification: FormalStatus(NameCert$_{\text{PerronMethodFiniteBarrier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a selected harmonic solution, the full Dirichlet problem, measure completion, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a checked carrier, barrier-comparison ledger, transport and replay stability, and local NameCert rows for BEDC.Derived.PerronMethodFiniteBarrierUp.

Chapter 8744

A Concrete Naming Certificate for $\mathsf{CauchySequenceModulusChain}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy moduli, the $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ packet names the finite chain of displayed precision refinements by which a regular sequence read becomes available to Real-facing completion consumers. It sits after the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real seal of chapter 38, and the regular completion route of chapter 1639. The packet records a finite modulus-chain surface; it does not assert quotient stream equality, selected limits, ambient completeness, countable choice, or host real equality.

Definition 8744.1 (Cauchy sequence modulus-chain carrier)**.** *A* $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, C, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row whose requests determine which sequence entries are visible.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to state every comparison budget in the chain.*
4. *C is the displayed chain of Cauchy-modulus refinements, ordered so that each later row refines the tolerance requested by the previous row.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing completion reads.*
6. *H records componentwise* hsame *transport for the window, readback, tolerance, chain, seal, replay, provenance, and naming rows.*
7. *T records displayed* Cont *replay of the same finite chain.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceModulusChain}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceModulusChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, C, E, H, T, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceModulusChainUp.CauchySequenceModulusChainTasteGate single carrier alignment`

Theorem 8744.2 (Cauchy sequence modulus-chain NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ *carrier*

$$M = (S, R, D, C, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceModulusChain}^{\uparrow}}$ *surface consists exactly of carrier admission, chain coherence, tolerance monotonicity, readback visibility, Real-seal discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every row of the displayed chain* C *must read its tolerance through* D*, expose its observations through* S *and* R*, and reach* E *only after the finite refinement chain has been displayed.*

Proof. Project the carrier of Definition 8744.1. Its visible rows are exactly $S, R, D, C, E, H, T, P, N$, so the local certificate has no hidden coordinate from which quotient stream equality, selected limits, choice, or ambient completeness can be recovered.

Read the analytic rows in order. The finite windows S expose the requested sequence entries, R records the regular rational readback, and D supplies the dyadic tolerance language in which each refinement in C is stated.

The chain row C is then checked for coherence and monotonicity: later requests may refine earlier tolerances, but they cannot bypass the displayed tolerance ledger. The terminal seal E is available only after that finite chain has been read.

Finally H, T, P, N transport, replay, source, and name the same finite packet. Since these structural rows preserve only the displayed route, the NameCert obligations are exhausted by the stated rows. □

Theorem 8744.3 (Cauchy sequence modulus-chain regular handoff)**.** *Every Real-facing completion read sourced from an accepted* $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ *packet factors through the displayed finite chain*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ C,\ \mathsf{Real}^{\uparrow}$$

before the terminal Real seal is consumed. The handoff exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $M = (S, R, D, C, E, H, T, P, N)$. The only observation windows are S, the only regular rational readback row is R, and the only tolerance language for the completion request is D.

The completion-facing read must therefore pass through the chain row C before it reaches the terminal seal E. A consumer that tries to inspect E without C omits the displayed modulus-refinement route and is not reading the accepted packet.

The transport, replay, provenance, and naming rows preserve the same order. Thus every Real-facing read factors through the finite chain before the Real seal, and none of the excluded host-level or choice-level data is exported. □

Falsifiable prediction: Every accepted $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ completion read exposes finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a displayed Cauchy-modulus refinement chain, a terminal Real seal, transport, replay, provenance, and local NameCert rows before exporting the read.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceModulusChain}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceModulusChain}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a displayed Cauchy-modulus refinement chain, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceModulusChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8744.1, Definition 8744.2, and Definition 8744.3 seed the finite modulus-chain route from regular rational windows to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceModulusChain}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchySequenceModulusChainUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chain coherence, tolerance monotonicity, readback visibility, Real-seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceModulusChainUp.

Chapter 8745

A Concrete Naming Certificate for $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$

The $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$ packet realizes the comparative genetic-code deformation ledger of Definition 9462.1905 as finite BEDC NameCert data. It sits after the stop-codon suffix bridge of chapter 1374, the six-bit codon source window of chapter 2234, the finite state graph surface of chapter 2306, and the partition incidence carrier of chapter 4017. The packet records table-wise stop/sense deformation from the standard stop boundary; it does not derive genetic-code universality, biological dynamics, amino-acid chemistry, or empirical reassignment likelihoods.

Definition 8745.1 (Genetic-code deformation ledger carrier)**.** *A* $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, S, E, A, U, V, W, L, \Delta, H, C, P, N)$$

with the following displayed rows:

1. T *is the external genetic-code table row under comparison.*
2. S *is the displayed stop-set row for* T*.*
3. E *is the displayed sense-set row for* T*.*
4. A, U, V, W *are the role rows for* UAA*,* UAG*,* UGA*, and* UGG*.*
5. L *is the standard* L*-shaped stop-boundary row* $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$ *read through* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *and* $\mathsf{CodonSixBitWindow}^{\uparrow}$*.*
6. Δ *is the symmetric-difference witness row comparing* S *with the standard stop boundary* L*.*
7. H *records componentwise* hsame *transport for the table, stop-set, sense-set, role, boundary, difference, replay, provenance, and naming rows.*
8. C *records displayed* Cont *replay of the same finite comparison route.*
9. P *is the* Pkg *provenance over* $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$*,* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}$ $\mathsf{CoreStateGraph}^{\uparrow}$*,* $\mathsf{PartitionCoreIncidence}^{\uparrow}$*, the codon-cube boundary program,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{GeneticCodeDeformationLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, E, A, U, V, W, L, \Delta, H, C, P, N$. It has no coordinate for genetic-code necessity, all-code universality, release-factor dynamics, amino-acid chemistry, or stochastic readthrough data.

Lean4 ✓ `BEDC.Derived.GeneticCodeDeformationLedgerUp.GeneticCodeDeformationLedgerTasteGate single carrier alignment`

Theorem 8745.2 (Genetic-code deformation ledger NameCert obligations)**.** *For every accepted* $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$ *carrier*

$$G = (T, S, E, A, U, V, W, L, \Delta, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GeneticCodeDeformationLedger}^{\uparrow}}$ *surface consists exactly of carrier admission, genetic-code table exposure, StopSet and SenseSet visibility, role-row exactness for* UAA*,* UAG*,* UGA*, and* UGG*, standard L-boundary exposure, symmetric-difference witness visibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No ledger obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8745.1. Its visible rows are exactly $T, S, E, A, U, V, W, L, \Delta, H, C, P, N$, so the NameCert surface cannot add an unlisted biological or empirical coordinate.

The table row T anchors the comparison. The rows S and E expose which codons are recorded as stops and senses, while A, U, V, W display the four boundary roles inspected by the ledger.

The standard boundary is read through L, and the deformation is read only through Δ. Thus preserving the local certificate requires preserving the displayed stop/sense and role rows together with the stated symmetric difference.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since those structural rows carry no empirical dynamics or amino-acid semantics, the listed obligations exhaust the packet. □

Theorem 8745.3 (Genetic-code deformation boundary classification)**.** *Every accepted* GeneticCodeDeformationLed *packet classifies a table row T as standard-boundary preserving exactly when the displayed symmetric-difference row Δ is empty, and as a stop-boundary deformation exactly when Δ is displayed as nonempty. The classification factors through S, L, Δ after the four role rows A, U, V, W are visible, and it exports no claim of genetic-code necessity, all-code universality, or biological dynamics.*

Proof. Start with the stop-set row S and the standard L-shaped row L. These are the only stop-boundary sets available to the accepted packet.

The symmetric-difference row Δ is the sole classifier for deformation. If it is empty, the packet records preservation of the standard boundary; if it is nonempty, the packet records precisely the displayed boundary displacement.

The role rows A, U, V, W make the four inspected codon roles visible before the classification is exported. The structural rows H, C, P, N preserve that finite comparison route and do not add genetic-code necessity, all-code universality, release-factor dynamics, or readthrough probabilities. □

Falsifiable prediction: Every accepted $\mathsf{GeneticCodeDeformationLedger}^{\uparrow}$ comparison exposes a table row, StopSet, SenseSet, four boundary role rows, the standard L-shaped stop boundary, a symmetric-difference witness, transport, replay, provenance, and local NameCert rows before classifying the table as boundary-preserving or boundary-deforming.

Closure status for $\mathsf{NameCert}_{\mathsf{GeneticCodeDeformationLedger}^{\uparrow}}$.

Constructive story: A GeneticCodeDeformationLedger$^{\uparrow}$ packet is generated from a genetic-code table row, StopSet and SenseSet rows, UAA/UAG/UGA/UGG role rows, the standard L-shaped stop-boundary row, a symmetric-difference witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GeneticCodeDeformationLedger}^{\uparrow}}$, seed)

Scope closed: Definition 8745.1, Definition 8745.2, and Definition 8745.3 seed the finite comparative deformation ledger for genetic-code table rows.

Formal verification: FormalStatus(NameCert$_{\text{GeneticCodeDeformationLedger}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.GeneticCodeDeformationLedgerUp`

Bridge status: none

Not claimed: This chapter does not close genetic-code universality, biological dynamics, amino-acid chemistry, empirical reassignment likelihoods, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stop/sense visibility, four-role exactness, standard boundary exposure, symmetric-difference classification, transport, replay, provenance, and local naming for BEDC.Derived.GeneticCodeDeformationLedgerUp.

Chapter 8746

A Concrete Naming Certificate for RegularCauchyModulusOfConvergence$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RegularCauchyModulusOfConvergence$^{\uparrow}$ packet records the finite convergence-modulus service attached to a regular Cauchy rational name before any quotient real equality, selected limit, or ambient completeness theorem is available. It sits over the regular sequence route of chapter 301, the finite window service of chapter 126, the dyadic tolerance ledger of chapter 291, the Cauchy difference export of chapter 1874, and the terminal real-name seal of chapter 38. The packet gives later Real$^{\uparrow}$ equality and completion consumers a visible modulus row rather than a hidden choice schedule.

Definition 8746.1 (Regular Cauchy modulus of convergence carrier)**.** *A* RegularCauchyModulusOfConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, M, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* RegSeqRat$^{\uparrow}$ *source row whose rational observations are read only through finite requests.*
2. *M is the modulus row assigning each requested dyadic precision to a visible tail threshold for X.*
3. *W is the* StreamName$^{\uparrow}$ *window row exposing exactly the finite source observations used after the threshold row M is read.*
4. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance row against which the requested precision and tail comparisons are checked.*
5. *R is the regular convergence row recording that all displayed tail reads of X beyond the threshold supplied by M are controlled by the same T tolerance.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row available to a Real equality or completion consumer only after X, M, W, T, R have been read.*
7. *H records componentwise* hsame *transport for the source, modulus, window, tolerance, convergence, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route from the source and precision request through the modulus, finite window, tolerance, convergence, and Real seal rows.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, W, T, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8746.2 (Regular Cauchy modulus of convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}$ *carrier $K = (X, M, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular rational source, displayed modulus row, finite StreamName window, dyadic tolerance ledger, regular convergence row, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Every Real equality or completion consumer using the modulus must read

$$X \longmapsto M \longmapsto W \longmapsto T \longmapsto R \longmapsto E$$

together with H, C, P, N, and receives no quotient stream equality, selected representative, countable-choice schedule, trichotomy, host real equality, or ambient completeness theorem.

Proof. Project the carrier of Definition 8746.1. The only analytic source is the $\mathsf{RegSeqRat}^{\uparrow}$ row X, and the only precision-to-tail service is the displayed modulus row M.

The window row W exposes the finite observations available after M is read. The dyadic tolerance row T is the sole tolerance ledger for those observations, so the convergence row R can record tail control only through X, M, W, T.

The terminal Real seal E is downstream of the same finite route. The structural rows H, C, P, N transport, replay, source, and name that route without adding quotient equality, selected limits, choice schedules, trichotomy, host real equality, or ambient completeness. □

Theorem 8746.3 (Regular Cauchy modulus of convergence tail stability)**.** *If an accepted* RegularCauchyModulusOf
carrier $K = (X, M, W, T, R, E, H, C, P, N)$ is read at a requested dyadic precision, then every visible tail comparison exported beyond the threshold row M is stable under the same finite route

$$X, M, W, T, R, H, C, P, N.$$

Consequently any $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may cite the modulus only as a displayed tail-stability service over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, not as a quotient stream equality, selected limit, trichotomy, countable choice, host real equality, or ambient completeness principle.*

Proof. Begin with the obligation surface of Definition 8746.2. It places every accepted modulus read on the displayed route through $X, M, W, T, R, E, H, C, P, N$.

For a tail comparison, the Real seal E is not needed until the consumer asks for the terminal equality-facing handoff. The stability claim is already carried by R after the source, modulus, finite window, and dyadic tolerance rows have been consumed.

Now apply the public export route of Definition 1874.14. A difference-criterion consumer can use the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows as visible inputs before reaching a $\mathsf{Real}^{\uparrow}$ seal, so the modulus remains a displayed finite service rather than a hidden quotient or choice object. □

Falsifiable prediction: Every $\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}$ read factors through one RegSeqRat source row, a displayed modulus row, finite StreamName windows, a DyadicRatCore tolerance ledger, a regular convergence row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, cauchy-difference-criterion, real: $\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source row, precision-to-tail modulus row, finite window row, dyadic tolerance ledger, convergence row, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}$ packet is generated from a RegSeqRat source row, displayed convergence-modulus row, StreamName window row, DyadicRatCore tolerance row, regular convergence row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8746.1, Definition 8746.2, and Definition 8746.3 seed the finite convergence-modulus route consumed by Real equality and Cauchy-difference readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusOfConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, finite-window exposure, dyadic tolerance compatibility, convergence-row stability, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyModulusOfConvergenceUp.

Chapter 8747

A Concrete Naming Certificate for BishopDyadicLocatedIntervalSelection$^{\uparrow}$

Following Bishop and Bridges, the BishopDyadicLocatedIntervalSelection$^{\uparrow}$ packet records the finite act of selecting a dyadic located interval from displayed endpoint and tolerance data. It sits after the dyadic rational surface of chapter 291, the real interval surface of chapter 4111, the located interval surface of chapter 394, and the completion handoff through chapter 301, chapter 126, and chapter 38. The packet supplies a choice-free interval-selection interface for Real and Completion consumers; it is not a Heine–Borel theorem, an open-cover compactness principle, a quotient-real equality rule, or a host choice operator.

Definition 8747.1 (Bishop dyadic located-interval selection carrier)**.** *A* BishopDyadicLocatedIntervalSelection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the located request row, recording the requested interval predicate, endpoint side, positive tolerance, and locatedness evidence read from* LocatedInterval$^{\uparrow}$ *and* RealInterval$^{\uparrow}$*.*
2. *D is the dyadic bracket row supplied by* DyadicRatCore$^{\uparrow}$*; it stores the lower and upper dyadic rationals, their outward rounding checks, and the finite bracket evidence for the request row.*
3. *M is the midpoint and refinement budget row, recording the dyadic midpoint reads, the requested refinement depth, and the finite comparison rows used to keep the bracket located.*
4. *R is the regular-Cauchy handoff row from the selected dyadic bracket to* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readback at the requested precision.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row for the selected interval endpoints and the interval-selection consumer.*
6. *H records componentwise* hsame *transport for the request, bracket, midpoint, handoff, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for located requests, dyadic bracket inspection, midpoint refinement, regular-Cauchy handoff, and Real-seal consumption.*
8. *P is the* Pkg *provenance over* BishopDyadicLocatedIntervalSelection$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}}$ *row.*

The carrier predicate says exactly that these rows form one finite located-interval selection packet. It contains no coordinate for open-cover compactness, host endpoint choice, quotient equality of reals, excluded middle over cuts, or an ambient completeness theorem.

Theorem 8747.2 (Bishop dyadic located-interval selection obligations)**.** *For every accepted* BishopDyadicLocatedIn *carrier $S = (Q, D, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the finite packet of Definition 8747.1;*
2. *located-request admission through the displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{RealInterval}^{\uparrow}$ *rows in Q;*
3. *dyadic bracket admission through the outward rounding and endpoint comparison rows of D;*
4. *midpoint and refinement-budget admission through the finite dyadic midpoint and comparison rows of M;*
5. *regular-Cauchy handoff through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows in R;*
6. *Real-seal non-escape through E after the request, bracket, midpoint, and handoff rows have been displayed;*
7. *componentwise transport, replay, provenance, and naming through H, C, P, N.*

No obligation for Heine–Borel compactness, open-cover selection, quotient-real equality, host choice, or a hidden real representative is part of the packet.

Proof. Project the carrier of Definition 8747.1. The request row Q supplies only displayed locatedness and endpoint data, while D supplies only finite dyadic bracket evidence.

The midpoint row M refines that bracket by finite dyadic comparisons. It neither selects a host real endpoint nor appeals to an open-cover argument; each refinement step is one displayed midpoint or comparison row.

The handoff row R then exposes the selected bracket as stream windows and regular rational readback, and E seals the resulting endpoint reads as $\mathsf{Real}^{\uparrow}$ data. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

Since every consumer read factors through those rows, the local NameCert has no remaining coordinate from which compactness, quotient equality, host choice, or hidden endpoint representatives could be exported. □

Theorem 8747.3 (Bishop dyadic located-interval selection sibling lattice)**.** *Every accepted* BishopDyadicLocatedInt *consumer must read the sibling lattice*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{RealInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $Q, D, M, R, E, H, C, P, N$ before treating the result as an interval-selection surface. The dyadic rational sibling supplies bracket arithmetic, the located and real interval siblings supply request and endpoint admission, the stream and regular-rational siblings supply completion-facing readback, and the Real sibling supplies only the terminal seal. No sibling may be bypassed by a host root, host endpoint, quotient representative, compactness theorem, or choice functional.

Proof. Begin with the obligation theorem Definition 8747.2. It places request admission in Q, dyadic bracket admission in D, midpoint refinement in M, completion handoff in R, and Real-seal reads in E.

Thus $\mathsf{DyadicRatCore}^{\uparrow}$ is read only as bracket arithmetic and midpoint refinement, while $\mathsf{LocatedInterval}^{\uparrow}$ and $\mathsf{RealInterval}^{\uparrow}$ are read only through the request and endpoint rows. The handoff to $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ occurs after those rows, not before them.

Finally the $\mathsf{Real}^{\uparrow}$ seal is downstream of the same displayed path. Componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming keep the route inside the finite packet, so no sibling can be replaced by a host endpoint, quotient representative, compactness theorem, or choice functional. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8747.1, Definition 8747.2, and Definition 8747.3 seed the finite located-request, dyadic-bracket, midpoint-refinement, regular-Cauchy handoff, Real-seal, transport, replay, provenance, and local NameCert rows for choice-free interval selection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, open-cover compactness, quotient-real equality, host endpoint choice, excluded middle over cuts, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bracket exactness, midpoint refinement stability, regular-Cauchy handoff, Real seal non-escape, and sibling-lattice non-bypass for BEDC.Derived.BishopDyadicLocatedIntervalSelectionUp.

Constructive story: A $\mathsf{BishopDyadicLocatedIntervalSelection}^{\uparrow}$ packet is generated from a located request row, dyadic bracket row, midpoint and refinement budget row, StreamNameUp and RegSeqRatUp handoff, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8748

A Concrete Naming Certificate for $\mathsf{LocatedRealLimit}^{\uparrow}$

Following Bishop and Bridges on located real numbers and limits of regular Cauchy data, the $\mathsf{LocatedRealLimit}^{\uparrow}$ packet records the finite boundary at which a regular rational name, a stream name, and a Real seal are allowed to present a located limit without importing host limits or countable choice. It reads the Real carrier of chapter 38, the regular rational sequence route of chapter 301, the finite observation windows of chapter 126, and the dyadic tolerance discipline of chapter 291. The packet names a choice-free limit boundary: approximants are admitted through displayed rows, tolerance coverage is local, uniqueness is separated by the Real carrier, and the ledger contains no coordinate for an external limit object.

Definition 8748.1 (Located real limit carrier)**.** *A* $\mathsf{LocatedRealLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, W, T, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the named located limit.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row whose rational approximants supply the displayed limit data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which requests for an index and tolerance are inspected.*
4. *T is the located tolerance coverage row, recording that every positive dyadic tolerance has a finite tail window in which the displayed rational approximants are compatible with R.*
5. *U is the separated uniqueness row for the* $\mathsf{Real}^{\uparrow}$ *seal, identifying two accepted limit packets only through Real separatedness and the shared tail coverage data.*
6. *H records componentwise* hsame *transport for R, Q, W, T, U, C, P, N.*
7. *C records displayed* Cont *replay of the same request-to-tail route.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, T, U, H, C, P, N$. It has no coordinate for a host limit, selected subsequence, quotient representative, ambient completeness of $\mathsf{Real}^{\uparrow}$*, countable choice, or classical trichotomy.*

Theorem 8748.2 (Located real limit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealLimit}^{\uparrow}$ *carrier* $L = (R, Q, W, T, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail data,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, located tolerance coverage, separated Real uniqueness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. In particular:*

1. *carrier admission exposes only the displayed rows* $R, Q, W, T, U, H, C, P, N$*;*
2. *located tolerance coverage is witnessed by the finite tail window row* W *and the tolerance row* T*;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *tail compatibility is read through the same tail row* Q *before the Real seal* R *may be consumed;*
4. *uniqueness of the named limit is separated by* U *and the* $\mathsf{Real}^{\uparrow}$ *carrier, not by host equality;*
5. *the provenance ledger has no escape to host limits, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8748.1. The only visible rows are $R, Q, W, T, U, H, C, P, N$, so admission and naming are finite and local to the packet.

Given a positive tolerance request, the finite window row W supplies the inspected tail segment and the tolerance row T records the corresponding located coverage. The regular rational tail row Q is the only source of approximants used by the Real seal R.

If two accepted packets present the same tail coverage, the separated uniqueness row U compares their terminal Real seals through the $\mathsf{Real}^{\uparrow}$ boundary. Transport, replay, provenance, and local naming preserve that route. Since no host-limit, selector, quotient representative, choice, trichotomy, or ambient-completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8748.3 (Located real limit tail route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LocatedRealLimit}^{\uparrow}$ *carrier factors its limit read through the finite route*

$$Q \to W \to T \to R \to U \to H \to C \to P \to N.$$

Thus the consumer reads $\mathsf{RegSeqRat}^{\uparrow}$ *tail data,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, located tolerance coverage, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, separated uniqueness, transport, replay, provenance, and local naming before treating the packet as a located real limit.*

Proof. Open the terminal Real seal row R. By Definition 8748.1, that row is packaged with the regular rational tail row, the finite observation window row, and the located tolerance row.

Apply Definition 8748.2. It forces each consumer read to expose Q, W, T before the seal and then the separated uniqueness, transport, replay, provenance, and local naming rows.

The sibling Real and regular rational chapters supply the carrier boundaries cited above, while this packet records their joint limit-facing handoff. Because the route contains no coordinate for host limits, selected representatives, countable choice, quotient equality, or ambient completeness, the displayed finite path is the complete consumer boundary. □

Falsifiable prediction: Every accepted LocatedRealLimitUp read exposes RegSeqRat tail data, StreamName finite windows, located tolerance coverage, a RealUp seal, separated uniqueness, hsame transport, Cont replay, Pkg provenance, and local NameCert before a consumer receives the limit boundary.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{LocatedRealLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds terminal Real sealing, regular rational tails, finite observation windows, located tolerance coverage, separated uniqueness, transport, replay, provenance, and local naming while refusing host limits and choice.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealLimit}^{\uparrow}$ packet is generated from a terminal Real seal, RegSeqRat tail data, StreamName finite windows, located tolerance coverage, separated Real uniqueness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8748.1, Definition 8748.2, and Definition 8748.3 seed the finite located-limit boundary over Real, RegSeqRat, and StreamName rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limits, selected representatives, countable choice, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located tolerance coverage, RegSeqRat tail compatibility, separated Real uniqueness, ledger non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealLimitUp

Chapter 8749

A Concrete Naming Certificate for $\mathsf{CauchyFilterSubsequence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and Bourbaki–Kelley on filters, the $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ packet records the finite route by which Cauchy-filter convergence evidence is read as subsequence windows without invoking countable choice. It sits between the Cauchy-filter surface of chapter 333, the regular-filter handoff of chapter 1445, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed extraction windows: it is not a quotient filter, not a selected infinite subsequence, not a countable-choice schedule, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8749.1 (CauchyFilterSubsequence carrier)**.** *A* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, R, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *convergence row, including the finite filter witnesses and directed refinement data visible to the packet.*
2. *R is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *handoff row that regularizes the filter evidence before any sequence-facing read is exposed.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected subsequence requests and their displayed positions.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerance windows opened by W.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, R, W, Q have been displayed.*
6. *H records componentwise* hsame *transport for the filter, regular-filter, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a filter tolerance request through the same finite rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, W, Q, E, H, C, P, N$. It has no coordinate for a chosen infinite subsequence, quotient filter equality, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8749.2 (CauchyFilterSubsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ *carrier $S = (F, R, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the regular-filter handoff row R, the finite* $\mathsf{StreamName}^{\uparrow}$ *extraction windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C, provenance P, and local naming N.*

Proof. Project the carrier of Definition 8749.1. The analytic rows are precisely the Cauchy-filter source, regular-filter handoff, stream-window extraction surface, regular rational readback, and terminal real-name seal.

The structural rows H, C, P, N transport, replay, source, and name that same finite tuple. Since the carrier contains no slot for a selected infinite subsequence, quotient filter equality, host sequence equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 8749.3 (CauchyFilterSubsequence finite-window route)**.** *Every* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ *consumer that extracts sequence-facing evidence from a Cauchy filter must factor through the displayed route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite subsequence windows and tolerance readback, but it never exports a selected infinite subsequence, quotient filter, countable-choice schedule, host equality of streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the source row F. It supplies only displayed filter convergence evidence, so a sequence-facing consumer must first read the regular-filter handoff R.

The handoff R opens the finite window row W. The regular rational readback Q can then inspect the tolerance windows named by W, and the terminal seal E is reached only after those rows have been displayed.

Transport, replay, provenance, and local naming preserve the same route through H, C, P, N. No row of the carrier can replace the route by a chosen infinite sequence or by quotient equality of filters or streams. □

Theorem 8749.4 (CauchyFilterSubsequence nonescape)**.** *Let $S = (F, R, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ *carrier. Every downstream Real or Completion consumer that reads subsequence evidence from S can read only the displayed rows $F, R, W, Q, E, H, C, P, N$ and their sibling chapter surfaces. In particular, no consumer can escape to a hidden selector, quotient filter equality, host stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 8749.2. They bind the local certificate to the nine displayed rows and leave no extra consumer coordinate.

Apply the finite-window route of Definition 8749.3. It places every sequence-facing read on the path from the Cauchy-filter row through regular-filter, stream-window, regular-rational, and Real-seal rows.

Thus a Real or Completion consumer sees exactly the visible route named by the packet. Hidden selectors, quotient equalities, countable-choice schedules, trichotomy, and ambient completeness are outside the carrier and cannot be read from it. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ read displays a CauchyFilter row, a RegularCauchyFilter handoff, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Real or Completion consumer extracts subsequence evidence.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, regseqrat, real: $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds filter convergence evidence, regular-filter handoff, finite subsequence windows, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming while refusing selected infinite subsequences and countable choice.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ packet is generated from CauchyFilter convergence rows, RegularCauchyFilter handoff rows, StreamName finite subsequence windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8749.1, Definition 8749.2, Definition 8749.3, and Definition 8749.4 seed the finite bridge from Cauchy-filter convergence evidence to visible subsequence windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite subsequences, quotient filters, quotient streams, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-source admission, regular-filter handoff, finite window extraction, regular rational readback, seal nonescape, transport, replay, provenance, and local naming for the same packet.

Chapter 8750

A Concrete Naming Certificate for $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$

Following Lusin and Bishop–Bridges constructive measure theory, the $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$ packet records the finite continuity-window route by which a measurable real-valued source is read on a displayed large set through a continuous representative row. It sits after the measure ledger of chapter 92, the continuous-map carrier of chapter 130, the approximation packet of chapter 4147, the finite exceptional-window discipline of chapter 2204, the regular-rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet names only a finite BHist continuity-window handoff; it does not claim full measure completion, quotient almost-everywhere equality, a selected null representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8750.1 (Lusin finite-continuity-window carrier)**.** *A* $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, L, C, E, R, S, H, Q, P, N)$$

with the following displayed rows:

1. *M is the measurable source row, exposing the finite* $\mathsf{Measure}^{\uparrow}$ *ledger, the inspected measurable real-valued row, and the measurable-set boundary on which the window is read.*
2. *L is the large-set ledger row, carrying the displayed subset, its finite complement budget, and the containment checks used by the continuity-window read.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *representative row, naming the continuous representative on the displayed large set rather than an ambient function-space element.*
4. *E is the exceptional-set budget row, recording the finite exceptional window inherited from the* $\mathsf{Egorov}^{\uparrow}$ *and* $\mathsf{LusinApproximation}^{\uparrow}$ *routes.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, turning the measured values and representative values inside the window into regular-rational comparison data.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, L, C, E, R have been displayed.*
7. *H records componentwise* hsame *transport for the measurable source, large-set ledger, continuous representative, exceptional-set budget, regular-rational readback, Real seal, replay, provenance, and local-name rows.*

8. *Q records displayed* Cont *replay of the finite route*

$$M, L, C, E, R, S \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LusinApproximation}^{\uparrow}$, $\mathsf{Egorov}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LusinFiniteContinuityWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, C, E, R, S, H, Q, P, N$. It has no coordinate for completed measure algebras, host measurable-function quotients, selected null representatives, countable choice, quotient almost-everywhere equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8750.2 (Lusin finite-continuity-window NameCert obligations)**.** *For every accepted* $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$ *carrier*

$$W = (M, L, C, E, R, S, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LusinFiniteContinuityWindow}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the measurable source row M, the large-set ledger row L, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *representative row C;*
2. *large-set exactness through L, requiring the displayed subset and complement budget to be read in the same finite* $\mathsf{Measure}^{\uparrow}$ *ledger as M;*
3. *exceptional-set budget exposure through E, requiring every continuity-window read to preserve the same finite exceptional row;*
4. *regular-rational readback through R, requiring measured and representative values inside the window to pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison surface;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape through S, requiring real-valued comparison to remain downstream of M, L, C, E, R;*
6. *structural stability through H, Q, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve exactly the rows M, L, C, E, R, S.*

No obligation may be discharged by completing the measure algebra, quotienting by almost-everywhere equality, selecting a null representative, importing a host Lusin theorem, or appealing to countable choice.

Proof. Project the carrier of Definition 8750.1. The measure-facing rows are M, L, E, the representative-facing row is C, the regular-rational readback row is R, and the terminal real seal is S.

The large-set ledger is read before any continuity verdict is exported. Thus the representative in C is attached to the displayed subset and complement budget in L, while the exceptional row E records the finite window inherited from the Egorov and Lusin-approximation routes.

The real-valued comparison is sealed only after regular-rational readback has been displayed. The row R fixes the inspected rational windows for the measured source and the continuous representative, and S exposes only the downstream $\mathsf{Real}^{\uparrow}$ comparison route.

Finally H, Q, P, N transport, replay, source, and name only those displayed rows. Since the excluded completed measure algebra, quotient almost-everywhere equality, selected null representative, host theorem, choice, and ambient completeness coordinates are absent, the listed rows exhaust the local NameCert obligations. □

Theorem 8750.3 (Lusin finite-continuity-window measure handoff)**.** *Every* $\mathsf{Measure}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of an accepted* $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{Measure}^{\uparrow} \longmapsto \mathsf{LusinApproximation}^{\uparrow} \longmapsto \mathsf{Egorov}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the rows $M, L, E, C, R, S, H, Q, P, N$*. The handoff exports no completed measure algebra, host measurable-function quotient, selected null set, quotient almost-everywhere equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the packet through Definition 8750.1. The source row M fixes the finite measure ledger, while L and E expose the large-set and exceptional-set budgets before the representative row C is available.

The regular-rational comparison then passes through R. Only after this readback can the terminal seal S be consumed, so every Real-facing handoff factors through the displayed measure, approximation, exceptional-window, continuous-representative, and regular-rational rows.

The structural rows H, Q, P, N preserve that same finite route under transport, replay, provenance, and local naming. No excluded coordinate is present from which a consumer could read a completed measure algebra, quotient almost-everywhere equality, selected null set, countable-choice schedule, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LusinFiniteContinuityWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{LusinFiniteContinuityWindow}^{\uparrow}$ packet is generated from finite Measure source rows, large-set ledger rows, ContinuousMap representative rows, exceptional-set budget rows, RegSeqRat readback rows, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LusinFiniteContinuityWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8750.1, Definition 8750.2, and Definition 8750.3 seed the finite large-set continuity-window route from Measure and LusinApproximation rows to ContinuousMap, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LusinFiniteContinuityWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full measure completion, host measurable-function quotients, selected null representatives, quotient almost-everywhere equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, large-set ledger exactness, continuous representative admission, exceptional-set budget exposure, regular-rational readback, Real seal non-escape, transport stability, and replay stability for BEDC.Derived.LusinFiniteContinuityWindowUp.

Lean4 (statement) `BEDC.Derived.LusinFiniteContinuityWindowUp`

Chapter 8751

A Concrete Naming Certificate for $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$

Following Bishop and Bridges on compact metric arguments by finite covers, the $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ packet records the finite numerical boundary used when a compact cover is consumed through a displayed ProbeBundle rather than through an external enumeration. It realizes the finite-cover route of chapter 9237: compactness supplies finitely many cover rows, the $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers record positive radius comparisons, and the resulting number row is consumed by uniform-modulus arguments without choosing a host finite set. The packet sits beside the compact carrier of chapter 60, the finite-net route of chapter 4760, and the rational lower-bound fold of chapter 4118.

Definition 8751.1 (Finite open-cover number carrier)**.** *A* $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, B, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row whose cover is being read.*
2. *B is the* ProbeBundle *row carrying the finite cover names.*
3. *R is the* $\mathsf{Rat}^{\uparrow}$ *radius row attached to each bundled cover element.*
4. *L is the lower-bound ledger row recording a positive rational number below the displayed cover radii.*
5. *U is the uniform-modulus handoff row consumed by compact-uniform-continuity arguments.*
6. *H records componentwise* hsame *transport for K, B, R, L, U, C, P, N.*
7. *C records displayed* Cont *replay of the same finite-cover read.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, ProbeBundle, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, R, L, U, H, C, P, N$. It has no coordinate for an externally numbered cover, a host finite set, a selected minimum, open-cover compactness outside the displayed compact row, countable choice, or ambient Real completeness.

Theorem 8751.2 (Finite open-cover number NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ *carrier* $O = (K, B, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNumber}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact source row,* $\mathsf{ProbeBundle}$ *finite cover row,* $\mathsf{Rat}^{\uparrow}$ *radius rows, positive lower-bound ledger, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer route must expose the finite cover through* B *and the rational radius ledger through* R, L *before it may read the uniform-modulus row* U*.*

Proof. Project the carrier of Definition 8751.1. The only rows available to the local classifier are $K, B, R, L, U, H, C, P, N$, so the cover number is named by finite packet data.

The compact row K supplies the source of the cover, while B supplies the displayed finite $\mathsf{ProbeBundle}$ cover names. The radius row R and lower-bound ledger L are the only numerical rows from which the cover number can be read.

The uniform-modulus handoff U is downstream of those cover and radius rows. Transport, replay, provenance, and local naming preserve that route. Since the carrier contains no external enumeration, host finite set, selected minimum, choice coordinate, open-cover compactness outside K, or ambient Real completeness, these rows exhaust the local $\mathsf{NameCert}$ obligations. □

Theorem 8751.3 (Finite open-cover number compact-uniform route)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ *carrier factors its finite-cover read through*

$$K \to B \to R \to L \to U \to H \to C \to P \to N.$$

Thus the consumer reads the compact source, $\mathsf{ProbeBundle}$ *finite cover, rational radius rows, positive lower-bound ledger, uniform-modulus handoff, transport, replay, provenance, and local naming before treating the packet as a finite open-cover number.*

Proof. Open the handoff row U. By Definition 8751.1, it appears only after the compact source row, finite cover bundle row, rational radius row, and lower-bound ledger row have been displayed.

Apply Definition 8751.2. It forces any compact-uniform consumer to read the finite cover as $\mathsf{ProbeBundle}$ data and the numerical cover bound as a $\mathsf{Rat}^{\uparrow}$ ledger before using the handoff.

The compact-uniform-continuity vision cited above consumes exactly this finite-cover route. Because the carrier has no external numbering, selected minimum, host finite set, countable choice, ambient completeness, or open-cover compactness outside the displayed compact row, the finite BEDC route is the complete boundary. □

Falsifiable prediction: Every accepted FiniteOpenCoverNumberUp compact-uniform read exposes a compact source row, ProbeBundle finite cover, RatUp radius rows, positive lower-bound ledger, UniformModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert before a consumer receives the cover number.

Independence witness: compact, compact-uniform-finite-net, probe-bundle-finite-minimum-fold: $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact cover source data, finite ProbeBundle cover rows, rational radius ledgers, a positive lower-bound number, uniform-modulus handoff, transport, replay, provenance, and local naming while refusing external enumerations and selected minima.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverNumber}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOpenCoverNumber}^{\uparrow}$ packet is generated from a compact source row, ProbeBundle finite cover row, RatUp radius rows, a positive lower-bound ledger, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteOpenCoverNumber}\uparrow}$, seed)

Scope closed: Definition 8751.1, Definition 8751.2, and Definition 8751.3 seed the finite cover-number boundary consumed by compact-uniform-continuity routes.

Formal verification: FormalStatus(NameCert$_{\text{FiniteOpenCoverNumber}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close externally numbered covers, host finite sets, selected minima, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ProbeBundle cover rows, RatUp radius ledgers, lower-bound number accounting, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOpenCove

Chapter 8752

A Concrete Naming Certificate for $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$

Following Brouwer and constructive finite-dimensional fixed-point presentations, the $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$ packet records a finite compact-convex certificate for a two-dimensional self-map before any fixed-point-facing read is allowed. It consumes the compact metric source of chapter 129, the convex carrier of chapter 208, the continuous self-map surface of chapter 130, the dyadic rational ledger of chapter 291, the stream window discipline of chapter 126, the regular sequence readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite mesh, labelling, tolerance, readback, transport, replay, provenance, and local naming route; it does not assert the unrestricted host theorem.

Definition 8752.1 (Brouwer fixed-point two-dimensional carrier)**.** *A* $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, V, F, M, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, recording the compact two-dimensional metric carrier and the finite cover data used by the certificate.*
2. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *row, recording convex membership, boundary faces, and admitted convex-combination operations for the displayed source.*
3. *F is the continuous self-map row, read through* $\mathsf{ContinuousMap}^{\uparrow}$*, exposing only the displayed modulus and finite image windows used on the mesh.*
4. *M is the finite vertex and mesh ledger, recording the triangulation cells, adjacency, boundary flags, and mesh-scale index.*
5. *L is the finite labelling ledger, assigning each boundary and interior mesh vertex the displacement label read from F.*
6. *D is the dyadic tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording mesh diameter, residual budgets, and label-comparison margins.*
7. *S is the Sperner witness row, carrying the fully labelled simplex and its finite induction trace inside the mesh ledger.*

8. *R is the readback row through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, exposing finite windows of the approximate fixed-simplex coordinates.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after K, V, F, M, L, D, S, R are displayed and used only to keep the readback inside the named compact-convex source.*
10. *H records componentwise* hsame *transport for the compact, convex, self-map, mesh, labelling, tolerance, witness, readback, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay for every consumer route through $K, V, F, M, L, D, S, R, E, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointTwoD}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, V, F, M, L, D, S, R, E, H, C, P, N$. It has no coordinate for full classical Brouwer fixed point, open-cover compactness, quotient point equality, choice-selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8752.2 (Brouwer fixed-point two-dimensional NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$ *carrier*

$$B = (K, V, F, M, L, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointTwoD}^{\uparrow}}$ *obligation surface consists exactly of compact-convex source exposure through K, V; continuous self-map exposure through F; finite vertex and mesh exposure through M; labelling exposure through L; dyadic tolerance exposure through D; Sperner witness exposure through S; StreamName and RegSeqRat readback through R; Real sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. Every fixed-point-facing read must pass through that finite route before it can cite the two-dimensional certificate.*

Proof. Project the carrier of Definition 8752.1. The compact-convex source is visible only through K, V, and the self-map is visible only through F.

The mesh part is finite. Induct on the displayed mesh ledger M: the boundary clauses read their labels from L and their dyadic margins from D, while each subdivision step preserves the same vertex, adjacency, and tolerance rows. The Sperner witness row S is therefore reached by finite mesh induction rather than by open-cover compactness.

The readback part is downstream. The StreamName and RegSeqRat row R exposes only finite coordinate windows of the simplex selected by S, and the Real seal E records that this readback remains inside the displayed compact-convex source.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for full classical Brouwer fixed point, open-cover compactness, quotient point equality, choice-selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the finite NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointTwoD}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BrouwerFixedPointTwoD}^{\uparrow}$ packet is generated from compact metric rows, convex source rows, continuous self-map rows, finite vertex and mesh rows, finite labelling and Sperner witness rows, dyadic tolerance rows, StreamName and RegSeqRat readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BrouwerFixedPointTwoD}^{\uparrow}}$, seed)

Scope closed: Definition 8752.1 and Definition 8752.2 seed the finite compact-convex fixed-point certificate over compact metric, convex, continuous-map, dyadic tolerance, stream readback, regular sequence, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BrouwerFixedPointTwoD}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full classical Brouwer fixed point, open-cover compactness, quotient point equality, choice-selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite mesh induction, labelling exactness, dyadic tolerance readback, StreamName and RegSeqRat window stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerFixedPointTwoDUp.

Chapter 8753

A Concrete Naming Certificate for $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$

Following Bourbaki and Kelley on proper metric spaces, completions, and compact exhaustions, the $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$ packet names the finite boundary by which a proper metric closed-ball ledger can be consumed by completion-facing and compactification-facing readers. It is a NameCert reconstruction of the classical fact that proper metric data supplies compact bounded windows around a completion boundary; it is not a quotient completion, not a selected representative scheme, not a countable-choice selector, not an open-cover compactness theorem, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

The sibling surface is explicit. The packet reads proper-radius windows from **??**, closed-ball membership from chapter 1451, compact-window witnesses from chapter 129, completion-facing boundary rows from chapter 1506, and terminal real readback from chapter 38. These siblings are consumed only as displayed BEDC rows, so the packet can hand compact-ball information to completion and one-point compactification corridors without importing host compactness, selected completion points, or ambient real completeness.

Definition 8753.1 (Proper metric completion boundary carrier)**.** *A* $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, R, C, K, Q, W, E, H, T, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{ProperMetric}^{\uparrow}$ *row, exposing the proper metric radius-window ledger.*
2. *R is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row, exposing center-radius non-strict membership for each displayed window.*
3. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *row attached to the closed-ball window.*
4. *K is the compact-ball handoff row, recording that compact-window consumers read R and C together over the same source metric.*
5. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary row, exposing only the completion-facing request that consumes the displayed compact windows.*
6. *W is the finite window row that aligns the proper-radius ledger with the completion request.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *readback seal for the radius and tolerance comparisons used by R, W, Q.*

8. *H records componentwise* hsame *transport for $M, R, C, K, Q, W, E, T, P, N$.*
9. *T records displayed* Cont *replay from the proper-radius request through closed balls, compact windows, the completion boundary, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ProperMetricCompletionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, C, K, Q, W, E, H, T, P, N$. There is no coordinate for quotient completion points, selected representatives, countable choice, open-cover compactness, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8753.2 (Proper metric completion boundary NameCert obligations)**.** *For every accepted* $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$ *carrier $B = (M, R, C, K, Q, W, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProperMetricCompletionBoundary}^{\uparrow}}$ *surface consists exactly of proper-metric radius admission, closed-ball membership admission, compact-window handoff, completion-boundary request admission, finite-window alignment, Real readback sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

No NameCert obligation is stored outside those rows, and no accepted read can recover a quotient completion, a selected representative, a countable-choice selector, an open-cover compactness principle, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 8753.1. The rows M, R, C, K, Q, W, E are the only mathematical data rows, and the support rows H, T, P, N only transport, replay, source, and name that displayed packet.

The proper metric route is finite. The source row M supplies a radius-window ledger, R records the corresponding closed-ball membership, and C records the compact-metric witness attached to that same window.

The completion-facing row is downstream of those windows. The boundary request Q can be read only after the compact-ball handoff K and the alignment row W are exposed, and its radius comparisons are sealed by E through the displayed Real readback.

Finally, H, T, P, N preserve this same route. Since the carrier has no coordinate for quotient completion points, selected representatives, countable choice, open-cover compactness, host equality, or ambient completeness, the local NameCert surface cannot export any of those principles. □

Theorem 8753.3 (Proper metric compact-ball handoff)**.** *Every completion-facing or compactification-facing consumer of an accepted* $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$ *carrier must read the compact-ball handoff through*

$$M \longmapsto R \longmapsto C \longmapsto K \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, T, P, N.$$

Thus the consumer receives proper metric closed-ball data, a compact-metric witness for the displayed ball, and a completion-boundary request only as one finite BEDC route. It receives no quotient completion point, selected representative, countable-choice schedule, open-cover compactness theorem, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the proper metric source. The row M names the finite radius-window surface, and the closed-ball row R reads membership for that displayed window.

Next read the compact witness. The compact-metric row C and handoff row K bind the witness to the same closed-ball window, so a consumer cannot substitute an independent compact object or an open-cover compactness theorem.

Now expose the completion side. The alignment row W matches the radius window with the completion-boundary request Q, and the Real seal E records only the displayed tolerance readback needed by that request.

The support rows H, T, P, N replay and name the same path. Therefore every completion-facing or compactification-facing read is confined to the stated carrier route and cannot escape into quotient completions, selected representatives, countable choice, host equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProperMetricCompletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ProperMetricCompletionBoundary}^{\uparrow}$ packet is generated from a ProperMetricUp radius-window row, a MetricClosedBallUp membership row, a CompactMetricUp witness row, a compact-ball handoff, a MetricCompletionUp boundary request, finite window alignment, a RealUp readback seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProperMetricCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8753.1, Definition 8753.2, and Definition 8753.3 seed the finite NameCert surface for proper metric compact-ball data consumed by completion-boundary readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProperMetricCompletionBoundary}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, open-cover compactness, host equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-ball handoff exactness, completion-boundary readback, transport, replay, provenance exhaustion, and ledger non-escape for BEDC.Derived.ProperMetricCompletionBoundaryUp.

Chapter 8754

A Concrete Naming Certificate for $\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}$ packet records the finite selector boundary needed when a closed real interval is read through constructive finite data. It sits after the interval carrier of chapter 4111, the located finite-net rows of chapter 1808, the finite-net witness surface of chapter 3471, the dyadic mesh route of chapter 5966, the stream-name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter gives compact-interval consumers a named finite selection packet without asserting open-cover compactness, arbitrary subcover choice, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8754.1 (ClosedIntervalFiniteNetSelector carrier)**.** *A* $\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, M, F, V, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint rows read from a* $\mathsf{RealInterval}^{\uparrow}$ *packet, including the displayed endpoint-order and enclosure data.*
2. *M is the dyadic mesh-depth row. It records the positive precision request, mesh denominator, and outward rounding budget used by the selector.*
3. *F is the located finite-net row from* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$*; it displays the finite list of mesh cells, endpoint coverage checks, and locatedness witnesses visible at depth M.*
4. *V is the finite-net witness row routed through* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *and* $\mathsf{DyadicIntervalNet}^{\uparrow}$*; it records the exact finite candidate rows retained by the selector, not an arbitrary subcover.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the endpoint streams and the selected mesh cells.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning those windows and the mesh budget into regular rational observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every Real-facing endpoint or selected-net output.*
8. *H records componentwise* hsame *transport for endpoints, mesh depth, located finite-net rows, witness rows, stream windows, readback, Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay for finite selector reads through* $L, U, M, F, V, W, R, S, H, P, N$.
10. *P is the* Pkg *provenance over* ClosedIntervalFiniteNetSelector$^{\uparrow}$, RealInterval$^{\uparrow}$, LocatedClosedIntervalFiniteNet$^{\uparrow}$, HeineBorelFiniteNet$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ClosedIntervalFiniteNetSelector}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite selector history for a displayed closed real interval. It has no coordinate for open-cover compactness, selected arbitrary subcovers, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8754.2 (ClosedIntervalFiniteNetSelector NameCert obligations)**.** *For every accepted* ClosedIntervalFiniteNetSelector$^{\uparrow}$ *carrier* $Q = (L, U, M, F, V, W, R, S, H, C, P, N)$*, the local* NameCert$_{\text{ClosedIntervalFiniteN}}$ *surface consists exactly of endpoint admission, dyadic mesh-depth admission, located finite-net exposure, finite-net witness retention, stream-window exposure, regular rational readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No finite selector obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8754.1. The endpoint rows L, U, mesh row M, located finite-net row F, and witness row V are the only selection data in the packet.

Read the sibling finite-net route next. The located finite-net row F supplies finite mesh-cell and endpoint-coverage data, while V records the retained witness rows through HeineBorelFiniteNet$^{\uparrow}$ and DyadicIntervalNet$^{\uparrow}$. These rows name finite candidates; they do not choose an arbitrary subcover.

Finally route every Real-facing output through W, R, S and every structural read through H, C, P, N. Since the excluded compactness, quotient equality, trichotomy, countable-choice, and completeness principles are not coordinates of the carrier, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 8754.3 (ClosedIntervalFiniteNetSelector choice-free retention)**.** *Every accepted* ClosedIntervalFiniteNetS *packet retains finite selector data only through the displayed route*

$$L, U \longmapsto M, F, V \longmapsto W, R, S, H, C, P, N.$$

Consequently downstream compact-interval consumers may cite a finite net selected at a displayed dyadic depth, but they receive no open-cover compactness theorem, no selected arbitrary subcover, no quotient real equality, no order trichotomy, no countable-choice schedule, and no ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start from the endpoint rows. The interval boundary is read from L, U together with the order and enclosure discipline of RealInterval$^{\uparrow}$, so the selector never replaces the endpoints by host representatives.

Use the dyadic mesh-depth row M to enter the finite-net siblings. The located finite-net row F exposes finite mesh coverage at that depth, and the witness row V retains exactly the finite candidates routed through LocatedClosedIntervalFiniteNet$^{\uparrow}$, HeineBorelFiniteNet$^{\uparrow}$, and DyadicIntervalNet$^{\uparrow}$.

The stream windows, regular rational readback, and Real seal are then read through W, R, S. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same finite route. Since no carrier row stores the refused global principles, every consumer receives only the choice-free finite selector retained by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8754.1, Definition 8754.2, and Definition 8754.3 seed the finite selector surface over $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected arbitrary subcovers, quotient real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-net witness retention, Real seal nonescape, choice-free routing, and ledger refusal for BEDC.Derived.ClosedIntervalFiniteNetSelectorUp.

Constructive story: A $\mathsf{ClosedIntervalFiniteNetSelector}^{\uparrow}$ packet is generated from closed interval endpoints, a dyadic mesh depth, located finite-net rows, finite witness rows, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8755

A Concrete Naming Certificate for $\mathsf{SignedDyadicRootBracket}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SignedDyadicRootBracket}^{\uparrow}$ packet records a finite dyadic interval whose endpoint signs bracket a root before any completion-facing consumer may read it. It sits over the dyadic rational ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback route of chapter 301, the interval surface of chapter 4111, and the terminal real-name seal of chapter 38. The packet names only the displayed signed endpoints, tolerance ledger, windowed function reads, bisection row, transport, replay, provenance, and local naming route.

Definition 8755.1 (Signed dyadic root-bracket carrier)**.** *A* $\mathsf{SignedDyadicRootBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, a, b, L, R, D, W, F, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{RealInterval}^{\uparrow}$ *row naming the inspected interval.*
2. *a, b are the displayed dyadic endpoints read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *L and R are the endpoint sign rows, recording opposite signs for the windowed reads at a and b.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling endpoint arithmetic, midpoint arithmetic, and sign-margin comparisons.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing exactly the finite function observations used at the two endpoints and at later bisection midpoints.*
6. *F is the function-read row, available only through the displayed finite windows and dyadic tolerances.*
7. *S is the signed bracket row tying I, a, b, L, R, D, W, F into one admitted bracket.*
8. *Q is the bisection row recording the inspected midpoint, the retained dyadic subinterval, and the retained endpoint-sign witness.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the interval, endpoint, sign, tolerance, window, function-read, bracket, and bisection rows have been displayed.*
10. *H records componentwise* hsame *transport for the interval, endpoint, sign, tolerance, window, function-read, bracket, bisection, seal, replay, provenance, and naming rows.*

11. *C records displayed* Cont *replay of every consumer route through* $I, a, b, L, R, D, W, F, S, Q, E, H$.
12. *P is the* Pkg *provenance over* $\mathsf{SignedDyadicRootBracket}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicRootBracket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, a, b, L, R, D, W, F, S, Q, E, H, C, P, N$. It has no coordinate for host trichotomy, selected roots, quotient real equality, unrestricted intermediate-value reasoning, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8755.2 (Signed dyadic root-bracket NameCert obligations)**.** *For every accepted* SignedDyadicRootBracke *carrier*

$$B = (I, a, b, L, R, D, W, F, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDyadicRootBracket}^{\uparrow}}$ *surface consists exactly of interval exposure through I; dyadic endpoint exposure through a, b; endpoint-sign exposure through L, R; dyadic tolerance exposure through D; finite observation exposure through W; function-read exposure through F; signed bracket exposure through S; bisection exposure through Q; terminal real sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 8755.1. The interval row, two dyadic endpoints, and two sign rows are the only endpoint data available to a consumer.

The analytic read is finite. The window row W exposes only the observations used by the function-read row F, and the tolerance ledger D is the sole ledger for endpoint arithmetic, midpoint arithmetic, and sign margins.

The bracket and bisection rows are downstream of those finite observations. The row S records the accepted signed bracket, while Q records only one displayed midpoint inspection and the retained dyadic subinterval.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier contains no row for host trichotomy, selected roots, quotient real equality, unrestricted intermediate-value reasoning, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8755.3 (Signed dyadic root-bracket bisection stability)**.** *If an accepted* $\mathsf{SignedDyadicRootBracket}^{\uparrow}$ *carrier $B = (I, a, b, L, R, D, W, F, S, Q, E, H, C, P, N)$ is read by a bisection-facing consumer, the retained subinterval in Q must expose the same* $\mathsf{RealInterval}^{\uparrow}$ *source, dyadic endpoint ledger,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback discipline, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route before the consumer sees the retained bracket. The read exports a finite signed dyadic sub-bracket, not a selected root, quotient real equality, host trichotomy, or ambient completeness witness.*

Proof. Begin with the NameCert obligations of Definition 8755.2. They force every accepted bracket read through $I, a, b, L, R, D, W, F, S, Q, E, H, C, P, N$.

For a bisection read, inspect the row Q. Its midpoint and retained endpoint rows are admitted only through the same dyadic tolerance ledger D and the same finite observation window W used by the original signed bracket row S.

The retained subinterval is sealed only after the displayed readback and terminal real-name boundary are exposed. Componentwise transport and displayed replay keep the retained bracket on the same finite route, so no selected limit, host trichotomy, quotient real equality, or ambient completeness row is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDyadicRootBracket}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDyadicRootBracket}^{\uparrow}$ packet is generated from a RealInterval row, dyadic endpoint rows, endpoint-sign rows, a DyadicRatCore tolerance ledger, finite StreamName observation windows, RegSeqRat readback discipline, a function-read row, a signed bracket row, one bisection row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDyadicRootBracket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8755.1, Definition 8755.2, and Definition 8755.3 seed the finite signed dyadic root-bracket route consumed by later bisection and Real or Completion readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDyadicRootBracket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host intermediate value theorem, host trichotomy, selected roots or limits, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exactness, dyadic midpoint stability, finite-window function reads, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.SignedDyadicRootBracketUp.

Chapter 8756

A Concrete Naming Certificate for $\mathsf{OmegaGenerationBody}^{\uparrow}$

Following Lawvere and Mac Lane on categorical diagrams and Bishop-style certified data, the $\mathsf{OmegaGenerationBody}^{\uparrow}$ packet records the licensed generation body interface of chapter 9361 as finite NameCert data. It sits after the stream-display discipline of chapter 126, the product packet of chapter 34, and the local naming carrier surface of chapter 1366. The packet names only the displayed diagram body, legal stream display, code/readback, pairing, generation-order, transport, replay, provenance, and local naming rows; it is not a host tower object, an unlicensed external encoding, a global pairing axiom, or an ambient completeness principle.

Definition 8756.1 (Omega generation body carrier)**.** *An* $\mathsf{OmegaGenerationBody}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, L, K, J, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the licensed diagram body row. It records the finite coordinate list for the Ω generation diagram, the accepted object interfaces, and the readback arrows between coordinate rows.*
2. *L is the legal stream display row, read through* $\mathsf{StreamName}^{\uparrow}$*, exposing the public stream window used to display a body without replacing the body by a raw code string.*
3. *K is the code/readback row. It records a faithful readable code morphism together with the displayed decode witness that returns each public stream row to the body row it names.*
4. *J is the pairing involution row. It records the licensed endomorphism on each layer that claims paired behavior, its double-application classifier law, and the fixed-row or apart-pair branch.*
5. *O is the generation-order row. It records the finite forward route generated by displayed history extension and continuation rows before any body read may be consumed.*
6. *H records componentwise* hsame *transport for D, L, K, J, O and the structural rows.*
7. *C records displayed* Cont *replay for every consumer route through the diagram body, stream display, code/readback, pairing, and generation-order rows.*
8. *P is the* Pkg *provenance over* $\mathsf{OmegaGenerationBody}^{\uparrow}$*, the licensed interface rows of chapter 9361,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{NameCert}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OmegaGenerationBody}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, K, J, O, H, C, P, N$. It has no coordinate for an unlicensed tower carrier, a raw bit-string encoding, an involution unsupported by a named layer, a backward time map, quotient body equality, countable choice, or ambient completion.

Theorem 8756.2 (Omega generation body display NameCert obligations)**.** *For every accepted* $\mathsf{OmegaGenerationBody}^{\uparrow}$ *carrier*

$$G = (D, L, K, J, O, H, C, P, N),$$

every body read consumed through the local $\mathsf{NameCert}_{\mathsf{OmegaGenerationBody}^{\uparrow}}$ *must first expose the licensed diagram body row D, the legal stream display row L, the code/readback row K, the pairing involution row J, the generation-order route O, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local naming row N. Thus the body interface is consumed only as the finite row sequence*

$$D, L, K, J, O \longmapsto H, C, P, N,$$

with no hidden body coordinate outside the accepted carrier.

Proof. Project the carrier of Definition 8756.1. The rows D, L, K, J, O are the only interface-facing data: D supplies the licensed generation diagram body, L supplies the public stream display, K supplies faithful readable code and decode witnesses, J supplies typed pairing by involution, and O supplies the finite generation-order route.

The display cannot be read as a raw external encoding. The stream row is licensed through $\mathsf{StreamName}^{\uparrow}$, the code/readback row carries its own decode witness, and the product-shaped body data remains tied to the coordinate rows named by chapter 34.

The pairing and order rows remain separate. The row J authorizes paired behavior only where a licensed involution is present, while O records the forward generation route through history extension and continuation replay. Neither row supplies a global inverse time map or an untyped pairing principle.

Finally replay H, C, P, N. Componentwise transport, displayed continuation replay, provenance, and local naming preserve exactly the same finite route. Since the carrier contains no unlicensed tower coordinate, raw code row, unsupported pairing, quotient body equality, choice schedule, or ambient completion row, every body read must expose the listed NameCert rows before it can be consumed. □

Closure status for $\mathsf{NameCert}_{\mathsf{OmegaGenerationBody}^{\uparrow}}$.

Constructive story: An $\mathsf{OmegaGenerationBody}^{\uparrow}$ packet is generated from a licensed diagram body row, a legal StreamName display row, a faithful readable code and decode row, a pairing involution row, a finite generation-order route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OmegaGenerationBody}^{\uparrow}}$, seed)

Scope closed: Definition 8756.1 and Definition 8756.2 seed the finite Omega generation body surface over licensed interface, StreamName, Prod, NameCert, history, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OmegaGenerationBody}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unlicensed tower carriers, raw external encodings, global pairing axioms, backward time maps, quotient body equality, countable choice, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, licensed diagram-body exposure, legal stream display, faithful readable code, pairing involution typing, generation-order routing, transport, replay, provenance, and local naming for BEDC.Derived.OmegaGenerationBodyUp.

Chapter 8757

A Concrete Naming Certificate for $\mathsf{ToeplitzOperator}^{\uparrow}$

Following Toeplitz's Hilbert-space operator construction and the classical matrix reading of constant diagonals, the $\mathsf{ToeplitzOperator}^{\uparrow}$ packet records the finite BEDC surface on which a bounded operator is named by displayed sequence windows and a diagonal-constant matrix row. It sits beside the Hankel-shadow carrier of chapter 6634, the finite matrix carrier of chapter 49, the bounded-operator and operator-norm surfaces of chapter 1644 and chapter 3603, the regular sequence-space route of chapter 5965, and the scalar seal of chapter 38. The chapter names only the finite Toeplitz carrier; it does not assert the spectral theorem, host infinite matrices, quotient operator equality, selected completions, or host Hilbert-space equality.

Definition 8757.1 (Toeplitz operator carrier)**.** *A* $\mathsf{ToeplitzOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, U, M, D, B, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source sequence-window row, recording the visible source indices, values, and tail request read through* $\mathsf{RegularSequenceSpace}^{\uparrow}$*.*
2. *U is the finite target sequence-window row, recording the visible target indices and values produced by the same displayed action.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *row whose displayed entries are available only on the finite source-target rectangle named by S and U.*
4. *D is the diagonal-constancy row, requiring every displayed entry M_{ij} to be read from the same scalar row when the visible difference $i - j$ agrees.*
5. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row recording the finite operator action and linearity ledger on the displayed windows.*
6. *L is the boundedness and* $\mathsf{OperatorNorm}^{\uparrow}$ *ledger, carrying the scalar bound used by consumers of B.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for the diagonal coefficients and norm-bound endpoints.*
8. *H records componentwise* hsame *transport for S, U, M, D, B, L, R and the structural rows.*
9. *C records displayed* Cont *replay of the route*

$$S, U \longmapsto M, D \longmapsto B, L, R.$$

10. *P is the* Pkg *provenance over* $\mathsf{ToeplitzOperator}^{\uparrow}$, $\mathsf{HankelOperator}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host infinite matrix, spectral measure, quotient operator equality, chosen completion, or host Hilbert-space equality.

Theorem 8757.2 (Toeplitz operator NameCert obligations)**.** *For every accepted* $\mathsf{ToeplitzOperator}^{\uparrow}$ *carrier*

$$T = (S, U, M, D, B, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the source and target sequence windows, finite* $\mathsf{Mat}^{\uparrow}$ *admission, diagonal-constancy exactness, bounded-linear-operator compatibility, operator-norm ledger exposure,* $\mathsf{Real}^{\uparrow}$ *scalar sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer therefore reads a Toeplitz operator only through the finite route*

$\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCe

Proof. Project the carrier of Definition 8757.1. The visible analytic rows are exactly the source window S, target window U, matrix row M, diagonal-constancy row D, bounded-operator row B, norm ledger L, and scalar seal R.

Diagonal-constancy exactness reads M and D together. The matrix row is admitted only on the displayed finite rectangle, and D compares entries only by the visible difference index already named by S and U.

Bounded-operator compatibility is the route through B, L, R. The bounded action is exposed before the operator-norm ledger, and the scalar bound is sealed through $\mathsf{Real}^{\uparrow}$ without adding a host supremum or selected norm-attaining vector.

Finally replay H, C, P, N. These rows transport, replay, source, and name only the same finite packet, so the listed rows exhaust the local NameCert surface and leave spectral data, infinite matrices, quotient operator equality, selected completions, and host Hilbert equality outside the carrier. □

Theorem 8757.3 (Toeplitz operator diagonal-constancy route)**.** *Every diagonal read of an accepted* $\mathsf{ToeplitzOperator}^{\uparrow}$ *carrier factors through the finite source window S, finite target window U,* $\mathsf{Mat}^{\uparrow}$ *row M, diagonal-constancy row D, bounded operator row B,* $\mathsf{OperatorNorm}^{\uparrow}$ *ledger L,* $\mathsf{Real}^{\uparrow}$ *scalar seal R, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}$ *row. The route is a Toeplitz certificate-level boundary parallel to the anti-diagonal boundary of* $\mathsf{HankelOperator}^{\uparrow}$*, not a host equality of infinite operators.*

Proof. Start from the NameCert obligations of Definition 8757.2. They require a diagonal consumer to expose S, U, M, D before any bounded-operator or scalar-bound read is available.

Use the diagonal-constancy row. If two displayed pairs have the same visible difference index, the packet reads them through the same scalar coefficient row; if the difference is not named by the finite windows, the carrier exports no entry.

Next consume the boundedness route through B, L, R. The bounded operator row records the finite action, the operator-norm ledger records the displayed bound, and the $\mathsf{Real}^{\uparrow}$ row seals only the scalar coefficients and endpoints used by that ledger.

The structural rows H, C, P, N preserve this same finite route under transport and replay. Since the carrier has no coordinate for a spectral theorem, host infinite matrix, quotient operator equality, selected completion, or host Hilbert-space equality, the diagonal-constancy read stops at the displayed NameCert boundary. □

Falsifiable prediction: An accepted $\mathsf{ToeplitzOperator}^{\uparrow}$ read is refuted if a diagonal, boundedness, or scalar-bound consumer reaches a conclusion without first displaying the source window, target window, finite matrix row, diagonal-constancy row, bounded-operator row, operator-norm ledger, Real scalar seal, transport row, replay row, provenance row, and local naming row.

Independence witness: hankel-operator, matrix, boundedlinearoperator, operator-norm, regular-sequence-space, and real: $\mathsf{ToeplitzOperator}^{\uparrow}$ is not bijective to any listed sibling because only this packet jointly carries source and target sequence windows, diagonal-constant matrix data, bounded-operator and operator-norm rows, scalar seals, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{ToeplitzOperator}^{\uparrow}$ packet is generated from finite source and target sequence windows, a diagonal-constant MatrixUp row, a bounded-operator row, an OperatorNormUp ledger, a RealUp scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8757.1, Definition 8757.2, and Definition 8757.3 seed the finite Toeplitz carrier consumed by matrix, bounded-operator, operator-norm, regular-sequence-space, Hankel-neighbor, and real scalar rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ToeplitzOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, the spectral theorem, host infinite matrices, quotient operator equality, selected completions, or host Hilbert-space equality.

Upgrade path: Obligation closure requires checked carrier admission, finite source and target window routing, diagonal-constancy exactness, boundedness exposure, operator-norm handoff, scalar sealing, transport, replay, provenance, and local naming for BEDC.Derived.ToeplitzOperatorUp.

Chapter 8758

A Concrete Naming Certificate for $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ packet records the finite observation-window route by which a stream name, a regular rational readback, a dyadic tolerance ledger, and a terminal Real seal are read as one coherent NameCert surface. It sits over the finite stream windows of chapter 126, the regular rational sequence route of chapter 301, the dyadic tolerance core of chapter 291, and the real-name boundary of chapter 38. The packet is a finite L10 handoff for Real and Completion consumers; it is not quotient stream equality, a selected representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8758.1 (Regular sequence window coherence carrier)**.** *A* $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording exactly the inspected source indices, rational observations, and request tags.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same inspected windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row against which every displayed regularity and comparison read is checked.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, D have been read.*
5. *H records componentwise* hsame *transport for the stream window, regular readback, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the route from a finite observation request through the stream window, regular readback, dyadic tolerance, and Real seal.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceWindowCoherence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, R, D, E, H, C, P, N. It has no coordinate for quotient stream equality, a selected regular representative, countable choice, trichotomy, host real equality, an ambient limit object, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8758.2 (Regular sequence window coherence NameCert obligations)**.** *For every accepted* $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ *carrier* $K = (S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceWindowCoher}}$ *surface consists exactly of carrier admission for the finite StreamName window row, the RegSeqRat readback row, the DyadicRatCore tolerance row, the terminal Real seal row, componentwise transport, displayed replay, provenance, and local naming.*

Every Real-facing consumer using the packet must read

$$S \longmapsto R \longmapsto D \longmapsto E$$

together with H, C, P, N, and receives no quotient stream equality, selected representative, countable-choice schedule, trichotomy, host real equality, ambient limit object, or ambient completeness theorem.

Proof. Project the carrier of Definition 8758.1. The only source observations are the finite window entries in S, and the only regular rational interpretation of those entries is the readback row R.

The tolerance row D is the sole dyadic budget against which the readback may be consumed. Hence any Real-facing request must pass from S to R and then through D before it can reach the terminal seal E.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no slot for quotient equality, selected representatives, countable choice, trichotomy, host equality, an ambient limit object, or completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8758.3 (Regular sequence window coherence handoff)**.** *Every accepted* $\mathsf{RegularSequenceWindowCoherenc}$ *consumer of a Real-facing read factors through the finite handoff*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first exposes the window row S, then the regular rational readback row R, then the dyadic tolerance row D, and only then the terminal Real seal row E.

Proof. Start with the StreamName window row S. It is the only row that displays the inspected indices and source observations, so a consumer has no independent access to an infinite stream or a selected representative.

Read the regular rational row R over the same finite observations and then the dyadic tolerance row D. These two rows are the complete regular-sequence handoff: R supplies the readback discipline and D supplies the comparison budget.

The terminal Real seal E is downstream of that handoff. Transport, replay, provenance, and local naming preserve the route through H, C, P, N without adding quotient equality, countable choice, trichotomy, host real equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ Real-facing read factors through a finite StreamName window row, a RegSeqRat readback row, a DyadicRatCore tolerance row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite observation windows, regular rational readback, dyadic tolerance, terminal Real sealing, transport, replay, provenance, and local naming while refusing quotient streams, selected representatives, countable choice, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceWindowCoherence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceWindowCoherence}^{\uparrow}$ packet is generated from a finite StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceWindowCoherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8758.1, Definition 8758.2, and Definition 8758.3 seed the finite observation-window handoff consumed by Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceWindowCoherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, RegSeqRat readback compatibility, dyadic tolerance compatibility, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceWindowCoherenceUp.

Chapter 8759

A Concrete Naming Certificate for $\mathsf{LocatedMetricReflection}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion, the $\mathsf{LocatedMetricReflection}^{\uparrow}$ packet records the finite route by which locatedness, metric distance, and separated reflection are displayed before a completion consumer reads a point or limit row. It sits over the located-space surface of chapter 2043, the metric carrier of chapter 57, the metric-completion route of chapter 1506, the separated-metric reflection of chapter 1790, the finite stream windows of chapter 126, and the terminal real-name seal of chapter 38. The packet names the L10 countable-completion interface as displayed NameCert data; it is not quotient completion, selected limits, host equality of points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8759.1 (Located metric reflection carrier)**.** *A* $\mathsf{LocatedMetricReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, Q, S, W, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{LocatedSpace}^{\uparrow}$ *row, carrying the finite locatedness request, positive-gap budget, and completion-facing refusal boundary.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance witnesses are exposed before any completion row can be read.*
3. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route, or the equivalent* $\mathsf{Completion}^{\uparrow}$ *face, carrying the displayed Cauchy or filter completion handoff.*
4. *S is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier row that turns displayed zero-distance evidence into a separated uniqueness read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row through which source observations, distance requests, locatedness budgets, and completion windows are read.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal attached to distance and positive-gap readback before completion or separatedness is consumed.*
7. *H records componentwise* hsame *transport for locatedness, metric distance, completion route, separated reflection, stream window, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located metric request through X, M, W, E, then through Q or S according to whether the consumer asks for completion or separated reflection.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedMetricReflection}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, Q, S, W, E, H, C, P, N$. It has no coordinate for quotient points, selected limits, host point equality, countable choice, an unrestricted completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8759.2 (Located metric reflection NameCert obligations)**.** *For every accepted* $\mathsf{LocatedMetricReflection}^{\uparrow}$ *carrier*

$$L = (X, M, Q, S, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMetricReflection}^{\uparrow}}$ *surface consists exactly of locatedness exposure through X; metric distance exposure through M; completion-route exposure through Q; separated-reflection exposure through S; finite StreamName inspection through W; Real sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N. Every completion or separated-reflection consumer must read locatedness and metric distance before it can consume the completion handoff or separated classifier.*

Proof. Project the carrier in Definition 8759.1. The displayed rows are exactly $X, M, Q, S, W, E, H, C, P, N$, so a consumer has no route to completion or separatedness before the locatedness and metric rows have been displayed.

The locatedness row X supplies the finite request and positive-gap budget, while the metric row M supplies the distance witnesses read through W and sealed at E. These rows are the only source of located metric information in the packet.

The completion and separated-reflection rows are downstream. Row Q may be consumed only after the located metric read has been exposed, and row S may turn zero-distance evidence into a uniqueness read only after the same located metric route has been displayed. Transport, replay, provenance, and local naming preserve that route without adding quotient points, selected limits, countable choice, host equality, or ambient completeness. □

Theorem 8759.3 (Located metric reflection completion handoff)**.** *Every completion-facing read from an accepted* $\mathsf{LocatedMetricReflection}^{\uparrow}$ *carrier factors through the ordered route*

$$\mathsf{LocatedSpace}^{\uparrow} \to \mathsf{Met}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow}$$

before any separated uniqueness consumer may inspect $\mathsf{SeparatedMetricReflection}^{\uparrow}$. *Thus locatedness and metric reflection are read before completion or separatedness; the handoff exports no quotient point, selected limit, countable-choice schedule, host equality of completed points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the carrier rows X, M, W, E, Q first. The locatedness row fixes the finite request and positive-gap boundary, the metric row fixes the displayed distance evidence, the StreamName row fixes the finite observations, and the Real seal records the numerical readback.

Only after that route is visible can a completion-facing consumer read Q. The completion handoff is therefore a row of the same finite packet rather than an ambient completion operator or a quotient class.

Finally, if a consumer asks for separated uniqueness, it must continue through S after the completion-facing route has exposed the located metric data. The support rows H, C, P, N replay and name the same sequence, so no selected limit, host equality, countable-choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetricReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedMetricReflection}^{\uparrow}$ packet is generated from a LocatedSpaceUp row, MetricUp distance row, MetricCompletionUp route, SeparatedMetricReflectionUp classifier row, StreamNameUp inspection row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMetricReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8759.1, Definition 8759.2, and Definition 8759.3 seed the finite located metric route consumed by completion and separated-reflection interfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMetricReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient points, selected limits, host point equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locatedness exposure, metric-distance exactness, completion handoff, separated-reflection handoff, transport stability, replay stability, and local naming for BEDC.Derived.LocatedMetricReflecti

Chapter 8760

A Concrete Naming Certificate for $\mathsf{BishopLocatedSupremum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedSupremum}^{\uparrow}$ packet records a finite located-supremum witness for a displayed Bishop-style family of real names. It consumes the real-name seal of chapter 38, the dyadic tolerance core of chapter 291, the stream-window surface of chapter 126, the regular-rational readback of chapter 301, and the located-supremum route of chapter 1427. The chapter names only the finite witness packet and its approximation handoffs; it does not claim classical least-upper-bound completeness, trichotomy, countable choice, or a completed quotient-real supremum.

Definition 8760.1 (BishopLocatedSupremum carrier)**.** *A* $\mathsf{BishopLocatedSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, D, Q, R, L, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite Bishop family row whose members are read only as displayed* $\mathsf{Real}^{\uparrow}$ *names.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the family at the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic gap budget for upper and lower approximation tests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the displayed stream windows as regular rational names.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after F, W, D, Q have stayed visible.*
6. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row recording the finite located upper-bound and lower-approximation tests visible in the same window.*
7. *A is the approximation ledger pairing each requested dyadic tolerance with a displayed family element and an upper-bound witness.*
8. *H records componentwise* hsame *transport for the family, window, tolerance, readback, real seal, located-supremum row, approximation ledger, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a supremum request through $F, W, D, Q, R, L, A, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSupremum}^{\uparrow}}$ *row.*

The accepted carrier has no coordinate for a classical least upper bound, quotient-real equality, trichotomy, countable choice, selected maximizing sequence, or ambient completeness.

Theorem 8760.2 (BishopLocatedSupremum NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedSupremum}^{\uparrow}$ *carrier* $S = (F, W, D, Q, R, L, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedSupremum}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed family row, stream window, dyadic tolerance ledger, regular-rational readback, terminal real seal, located-supremum row, approximation ledger, componentwise transport, continuation replay, provenance, and local naming. Every accepted read factors through*

$$F, W, D, Q \longmapsto R, L, A, H, C, P, N$$

and refuses classical least-upper-bound completeness, quotient-real equality, trichotomy, countable choice, selected maximizing sequences, and ambient completeness data.

Proof. Project the carrier of Definition 8760.1. The visible rows are exactly $F, W, D, Q, R, L, A, H, C, P, N$, so the local NameCert can admit only those finite rows.

The approximation part begins with the family row F, the stream window W, and the dyadic tolerance row D. The regular-rational row Q repacks the displayed windows before the terminal $\mathsf{Real}^{\uparrow}$ seal R is available.

The located-supremum row L and approximation ledger A then record the upper-bound and lower-approximation witnesses in the same displayed window. The structural rows H, C, P, N transport, replay, source, and name this route without adding a classical least upper bound, quotient equality, trichotomy, countable choice, selected maximizing sequence, or ambient completeness coordinate. □

Theorem 8760.3 (BishopLocatedSupremum approximation nonescape)**.** *Every approximation read from an accepted* $\mathsf{BishopLocatedSupremum}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{LocatedSupremum}^{\uparrow}.$$

The read receives only the finite witness rows $F, W, D, Q, R, L, A, H, C, P, N$ and cannot export a classical least upper bound, quotient-real equality, trichotomy, countable choice, selected maximizing sequence, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an approximation request and project the accepted carrier. The stream window W and dyadic tolerance row D are the only rows that determine which finite precision test is in scope.

The regular-rational readback Q and terminal real seal R are downstream of those displayed rows. Thus a consumer cannot inspect a real-facing approximation before the finite StreamName and DyadicRatCore route has been displayed.

Finally apply the located-supremum row L and approximation ledger A. They bind the upper-bound and lower-approximation witnesses to the same finite window, while H, C, P, N preserve the route. Since the carrier omits quotient reals, selected maximizing sequences, trichotomy, countable choice, and ambient completeness, the approximation route cannot escape the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedSupremum}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedSupremum}^{\uparrow}$ packet is generated from a finite Bishop family row, StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, LocatedSupremum row, approximation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedSupremum}^{\uparrow}}$, seed)

Scope closed: Definition 8760.1, Definition 8760.2, and Definition 8760.3 bind the finite located-supremum witness packet over the Real and Completion route.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedSupremum}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim classical least-upper-bound completeness, quotient-real equality, trichotomy, countable choice, selected maximizing sequences, checked Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window routing, dyadic tolerance exposure, regular-rational readback, Real seal handoff, located-supremum approximation, and structural nonescape for BEDC.Derived.BishopLocatedSupremumUp.

Chapter 8761

A Concrete Naming Certificate for $\mathsf{RationalFinitePlaceProduct}^{\uparrow}$

Following the classical product formula for the rationals, this chapter opens the $\mathsf{RationalFinitePlaceProduct}^{\uparrow}$ NameCert entry over the existing $\mathsf{Rational}^{\uparrow}$ rational valuation interface for a BEDC-native finite-place product surface. The target is to connect the finite prime-power counts exposed by `PrimeFactorizationProd` to the existing zero-axiom rational valuation interface, including `ratVal`, `ratValIndex`, `ratVal_mul`, and the archimedean absolute-value surface `ratAbsInfty`. The intended terminal equality is the rational global product formula, with the finite prime-indexed factor and the archimedean factor assembled inside BEDC rather than imported from mathlib. The complete acceptance law family, anti-degeneracy constraints, and bridge authority are the contract at `papers/bedc_mathlib_bridge/gap_contrac` this chapter records only the main-pipeline entrance for that contract.

The finite support is read from the prime-factorization rows of chapter 64, while the local factors are routed toward the restricted-product consumers of chapter 101 and the valued-field surface of chapter 173. Thus the chapter sits in the concrete-instance dependency lattice as a rational product bridge: prime-power rows determine the finite index support, the rational valuation rows determine each local factor, and the downstream Adele/local-field consumers receive only those displayed finite rows rather than a mathlib place object.

Definition 8761.1 (Rational finite-place support target)**.** *For a nonzero rational name q, the intended finite-place support is the finite list of prime places whose* `ratVal` *index for q is nonzero, derived from the* `PrimeFactorizationProduct` *data of the numerator and denominator. It is not an arbitrary natural-index surface and it carries no mathlib place object.*

Definition 8761.2 (Rational finite-place product target)**.** *The intended finite-place product is a BEDC-native finite product over the support of Definition 8761.1. Each local factor is read from the existing rational valuation surface, while the infinite factor is read from* `ratAbsInfty`*; the terminal target is the cancellation statement*

$$\texttt{ratAbsInfty}(q) \cdot \prod_{p \in \mathrm{supp}(q)} |q|_p = 1.$$

The displayed equation is a target signature for the seed entrance only, not a claimed theorem of this chapter.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalFinitePlaceProduct}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalFinitePlaceProduct}^{\uparrow}}, \mathsf{seed})$

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalFinitePlaceProduct}^{\uparrow}}$, unformalized)

Bridge status: none

Constructive story: The finite place product is to be assembled constructively from PrimeFactorizationProduct support data, existing ratVal indices, ratVal multiplicativity, and ratAbsInfty.

Scope closed: This chapter only declares the rational finite-place product formula gap target and its acceptance entry; it does not construct the support, product, or terminal equality.

Not claimed: The final product formula and the mathlib correspondence are not claimed here; the correspondence belongs to the bridge side.

Upgrade path: Upgrade requires implementing the contract required structure and laws, then passing the bridge exported_core authority for bedc-rational-product-formula.

Chapter 8762

A Concrete Naming Certificate for $\mathsf{CauchySequenceRing}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{CauchySequenceRing}^{\uparrow}$ packet records the finite ring-level arithmetic surface for regular rational Cauchy sequence names before any quotient-free $\mathsf{Real}^{\uparrow}$ seal is read. It sits after the regular-rational readback of chapter 301, the finite observation windows of chapter 126, and the dyadic tolerance ledger of chapter 291; it supplies the ring-facing arithmetic row consumed later by chapter 3902; and it remains bounded by the terminal real-name boundary of chapter 38. The packet names rowwise addition, negation, multiplication, zero, one, and operation stability for regular Cauchy sequence arithmetic; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8762.1 (CauchySequenceRing carrier)**.** *A* $\mathsf{CauchySequenceRing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, W, D, A, M, Z, E, H, C, P, N)$$

with the following visible rows:

1. *X and Y are accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for the two regular rational Cauchy sequence names being combined.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that synchronizes the finite windows inspected from X and Y for the requested operation budget.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the tolerance cells and rational arithmetic rows used by the operation.*
4. *A is the additive row, displaying zero, addition, and negation over the synchronized regular-Cauchy source windows.*
5. *M is the multiplicative row, displaying one and multiplication over the same synchronized source windows and dyadic budget.*
6. *Z records the ring constants and the finite compatibility rows that keep zero, one, additive inverse, addition, and multiplication on the same regular-Cauchy name surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary row, recording that downstream real-name consumers may read the operation result only after the regular-Cauchy arithmetic rows have been displayed.*
8. *H stores componentwise* hsame *transports for X, Y, W, D, A, M, Z, E and the local naming rows.*

9. *C stores displayed* Cont *replay for the regular source rows, synchronized windows, dyadic budget, additive row, multiplicative row, constant row, and real boundary in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceRing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceRing}^{\uparrow}}$ *row.*

The carrier asserts only that these rows form one displayed ring-arithmetic packet for regular Cauchy sequence names. It has no field for quotient classes, selected sequence representatives, countable choice over moduli, trichotomy, host real arithmetic, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8762.2 (CauchySequenceRing NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceRing}^{\uparrow}$ *carrier $K = (X, Y, W, D, A, M, Z, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceRing}^{\uparrow}}$ *surface consists exactly of carrier admission, regular-rational source exposure through X and Y, synchronized-window exposure through W, dyadic operation-budget exposure through D, additive operation exposure through A, multiplicative operation exposure through M, constant and compatibility exposure through Z, terminal real-boundary exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No obligation may replace these rows by quotient Cauchy classes, selected representatives, host real arithmetic, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8762.1. The rows $X, Y, W, D, A, M, Z, E, H, C, P, N$ are the only coordinates available to the local certificate.

The source and window obligations read X and Y only through the synchronized schedule W. The dyadic ledger D supplies the finite tolerance and rational arithmetic cells needed by the displayed operation rows.

The additive row A exposes zero, addition, and negation, while the multiplicative row M exposes one and multiplication. The compatibility row Z keeps all five ring operations on the same regular-Cauchy source surface before the terminal $\mathsf{Real}^{\uparrow}$ boundary E is available to a downstream consumer.

Rows H, C, P, N transport, replay, source, and name exactly that displayed packet. Since the carrier contains no quotient row, selected representative row, countable-choice row, trichotomy row, host-real row, or ambient-completeness row, the NameCert obligations cannot export any of those stronger objects. □

Theorem 8762.3 (CauchySequenceRing operation stability)**.** *Every operation read exported by an accepted* $\mathsf{CauchySequenceRing}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchySequenceRing}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In particular, addition, negation, multiplication, zero, and one remain stable under componentwise hsame *transport of $X, Y, W, D, A, M, Z, E, H, C, P, N$, and any* $\mathsf{CauchySequenceField}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer sees the ring row only after the regular-rational sources, finite windows, dyadic budget, operation rows, compatibility row, transport, replay, provenance, and local name have been exposed.*

Proof. Start from an accepted carrier $K = (X, Y, W, D, A, M, Z, E, H, C, P, N)$. Carrier projection exposes the regular-rational sources, synchronized windows, dyadic budget, additive row, multiplicative row, constants, compatibility row, real boundary, transport, replay, provenance, and local name.

For an additive read, use A together with the shared schedule W and budget D. For a multiplicative read, use M with the same W and D. The constant and compatibility row Z records

that zero, one, additive inverse, addition, and multiplication are read on the same regular-Cauchy sequence surface.

Componentwise hsame transport in H moves each displayed row to the corresponding transported packet, and C replays the same route after transport. The downstream consumer therefore receives the same route composition through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceRing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The final boundary is E, so the route never identifies quotient Cauchy classes, chooses representatives, invokes countable choice, decides trichotomy, imports host real arithmetic, or claims ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceRing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceRing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore operation ledger, additive and multiplicative operation rows, ring constants and compatibility evidence, a Real boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceRing}^{\uparrow}}$, seed)

Scope closed: Definition 8762.1, Definition 8762.2, and Definition 8762.3 seed the regular Cauchy sequence ring surface between RegSeqRat, StreamName, DyadicRatCore, CauchySequenceField, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceRing}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient Cauchy classes, selected sequence representatives, host real arithmetic, countable choice, trichotomy, field inverses, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-window synchronization, dyadic-budget exactness, additive and multiplicative operation exposure, constant compatibility, real-boundary handoff, transport, replay, and ledger refusal for BEDC.Derived.CauchySequenceRingUp.

Chapter 8763

A Concrete Naming Certificate for $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$ packet records the finite metrizable-completion handoff used by the L10 Real and Completion corridor. It sits after the metric-completion row of chapter 1506, the located completion row of chapter 2114, the Cauchy completion embedding of chapter 1934, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the terminal Real seal of chapter 38. The packet names completion as displayed finite carrier data, not as quotient completion equality, selected representatives, countable choice, trichotomy, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8763.1 (FiniteMetrizableCompletion carrier)**.** *A* $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, J, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row exposing the metric source, finite tolerance ledger, and completion-facing route.*
2. *J is the* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ *row by which source observations enter the completion surface.*
3. *L is the* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *row that keeps located completion evidence visible before a Real-facing read is public.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the displayed tolerance and source data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same finite windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, J, L, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, embedding, located-completion, window, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from the metric source through the Cauchy embedding, located completion row, finite windows, regular readback, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, $\mathsf{LocatedMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetrizableCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected host limits, countable choice, trichotomy, compactness, host interval equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8763.2 (FiniteMetrizableCompletion NameCert obligations)**.** *For an accepted* FiniteMetrizableCompletio *carrier* $F = (M, J, L, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetrizableCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-completion source exposure through* M*, Cauchy-completion embedding through* J*, located-completion evidence through* L*, finite StreamName windows through* W*, RegSeqRat readback through* R*, terminal Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 8763.1. Its only completion-facing rows are the metric source, Cauchy embedding, located completion, finite window, regular readback, Real seal, transport, replay, provenance, and local naming rows.

The metric source row M supplies the completion-facing tolerance and source route. The embedding row J records how source observations enter the Cauchy completion surface, and the located row L keeps located completion evidence visible before the packet is read as Real-facing.

The finite numerical read proceeds through W and R. The StreamName row opens only the displayed finite windows, and the RegSeqRat row supplies the corresponding regular rational readback before the terminal Real seal E is consumed.

Finally H, C, P, N transport, replay, package, and name the same displayed route. Since no further carrier coordinate is present, quotient completion equality, selected limits, countable choice, trichotomy, compactness, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not exported. □

Theorem 8763.3 (FiniteMetrizableCompletion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionEmbedding}^{\uparrow} \longmapsto \mathsf{LocatedMetricCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

with BHist, hsame, Cont, Pkg, *and* NameCert *support still visible. The handoff exports no quotient completion, selected endpoint or limit representative, countable-choice schedule, trichotomy, compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 8763.2. It places every accepted consumer read on the rows $M, J, L, W, R, E, H, C, P, N$.

The completion-facing prefix is M, J, L. The metric-completion source is read first, the Cauchy-completion embedding then admits the displayed source observations, and the located-completion row keeps located evidence attached to the same finite packet.

The Real-facing suffix is W, R, E. Finite StreamName windows are opened before RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal is available only after that regular readback has been displayed.

The structural rows H, C, P, N preserve the same route. Because the carrier has no coordinate for quotient completion equality, selected representatives, choice schedules, trichotomy, compactness, or ambient completeness, the Real and Completion corridor receives exactly the finite handoff stated above. □

Independence witness: metriccompletion, cauchycompletionembedding, locatedmetriccompletion, streamname, regseqrat, real: $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$ is not bijective with any listed sibling because it joins metric source exposure, Cauchy embedding, located completion evidence,

finite windows, regular rational readback, Real sealing, componentwise hsame transport, Cont replay, Pkg provenance, and one local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetrizableCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetrizableCompletion}^{\uparrow}$ packet is generated from a metric-completion source row, Cauchy-completion embedding, located-completion evidence, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetrizableCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8763.1, Definition 8763.2, and Definition 8763.3 display the finite metrizable-completion handoff over MetricCompletion, CauchyCompletionEmbedding, LocatedMetricCompletion, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetrizableCompletion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected host limits, countable choice, trichotomy, compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires kernel-scope routing for $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, $\mathsf{LocatedMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert over the same packet.

Chapter 8764

A Concrete Naming Certificate for $\mathsf{CauchySpaceCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completions, and Kelley on uniform and completion spaces, the $\mathsf{CauchySpaceCompletion}^{\uparrow}$ packet names a non-quotient completion carrier for a displayed Cauchy space. It reconstructs the traditional completion interface as finite BEDC rows: effective Cauchy data, filter-Cauchy data, located uniform control, metric-completion handoff, and Bishop-style metric completion readback. The chapter consumes chapter 4099, chapter 6129, chapter 3814, chapter 4023, and chapter 1506; it does not claim quotient completion, selected limit representatives, countable choice, or ambient real completeness.

Definition 8764.1 (CauchySpaceCompletion carrier)**.** *A* $\mathsf{CauchySpaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (E, F, U, B, M, W, R, L, H, T, P, N)$$

with the following displayed rows.

1. *E is the source* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *row, exposing only the effective Cauchy approximation schedule accepted by the packet.*
2. *F is the* $\mathsf{CauchyFilterSpace}^{\uparrow}$ *row, recording the filter-Cauchy face read by completion consumers.*
3. *U is the* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *row, recording the located uniform entourage tests used to compare Cauchy names.*
4. *B is the* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *row, exposing the Bishop-style completion handoff without quotient representatives.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, recording the metric-completion request as a displayed boundary row.*
6. *W is the finite window row aligning effective Cauchy approximations, filter tests, located uniform tests, and metric-completion requests.*
7. *R is the readback row for the completion name, available only after E, F, U, B, M, W remain visible.*
8. *L is the exactness ledger: two accepted completion names are compared only through displayed Cauchy agreement, located uniform tests, and metric-completion boundary rows.*
9. *H records componentwise* hsame *transport for $E, F, U, B, M, W, R, L, T, P, N$.*

10. *T records displayed* Cont *replay from a completion request through the effective, filter, uniform, Bishop-metric, metric-completion, readback, and exactness rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchySpaceCompletion}^{\uparrow}$, $\mathsf{EffectiveCauchySpace}^{\uparrow}$, $\mathsf{CauchyFilterSpace}^{\uparrow}$, $\mathsf{LocatedUniformSpace}^{\uparrow}$, $\mathsf{BishopMetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySpaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $E, F, U, B, M, W, R, L, H, T, P, N$. It has no coordinate for quotient classes, selected representatives, countable-choice subsequences, ambient complete space data, or an external real-completeness principle.

Theorem 8764.2 (CauchySpaceCompletion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySpaceCompletion}^{\uparrow}$ *carrier $C = (E, F, U, B, M, W, R, L, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySpaceCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the effective Cauchy row, Cauchy-filter row, located-uniform row, Bishop metric-completion handoff, metric-completion boundary, finite window alignment, completion-name readback, exactness ledger, componentwise transport, continuation replay, provenance, and local naming. Every accepted completion read factors through*

$$E, F, U \longmapsto B, M, W \longmapsto R, L, H, T, P, N$$

and refuses quotient completion equality, selected limit representatives, countable choice, host completion objects, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8764.1. The rows E, F, U are the only source Cauchy rows: E supplies the effective schedule, F supplies the filter-Cauchy face, and U supplies located uniform comparison tests.

The completion handoff is downstream of those rows. The Bishop-metric row B and metric-completion boundary row M are read only after the finite alignment window W displays the same Cauchy request on all three source faces.

The readback and exactness rows do not add quotient data. The row R exposes the completion name, and L compares completion names only through displayed Cauchy agreement, located uniform tests, and the metric-completion boundary. The structural rows H, T, P, N transport, replay, source, and name that same route.

Since the carrier has no coordinate for quotient classes, selected representatives, countable-choice subsequences, host completion objects, or ambient real completeness, the local NameCert surface is exactly the displayed non-quotient completion packet. □

Theorem 8764.3 (Cauchy completion standard-bridge boundary)**.** *Any later standard bridge from* $\mathsf{CauchySpaceCompletion}^{\uparrow}$ *to a classical completion schema must consume only the displayed rows $E, F, U, B, M, W, R, L, H, T, P, N$. In particular, the bridge may cite the effective Cauchy schedule, the filter-Cauchy face, the located uniform tests, the Bishop metric-completion handoff, and the metric-completion boundary, but it cannot introduce quotient representatives, a selected limit map, countable choice, or an ambient completed host space.*

Proof. The bridge source is the accepted carrier itself. By Definition 8764.2, every completion read first passes through E, F, U and then through B, M, W before the readback and exactness rows are exposed.

A standard bridge can therefore repackage only those displayed rows. The exactness ledger L is the sole comparison interface, while H, T, P, N preserve the packet route and provenance.

No row stores quotient representatives, selected limits, countable-choice schedules, host completion data, or ambient completeness. Thus any bridge using this packet has the stated boundary. □

Theorem 8764.4 (CauchySpaceCompletion exactness ledger)**.** *Two accepted* $\mathsf{CauchySpaceCompletion}^{\uparrow}$ *carriers classify together only through displayed effective Cauchy agreement, Cauchy-filter agreement, located-uniform tests,* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary rows, finite alignment window, completion readback, exactness ledger* L*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8764.1. The comparison interface is the displayed row family $E, F, U, B, M, W, R, L, H, T, P, N$, with L named as the exactness ledger.

The source comparison first reads effective Cauchy agreement through E, Cauchy-filter agreement through F, and located-uniform tests through U. The completion-facing comparison then passes through the Bishop metric-completion row B, the metric-completion boundary row M, the finite alignment window W, and the completion readback row R.

The ledger L has no coordinates for quotient representatives, selected limits, countable choice, host completion objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The structural rows H, T, P, N only transport, replay, source, and name the displayed comparison route, so two accepted carriers classify together exactly through this ledger. □

Falsifiable prediction: A future $\mathsf{CauchySpaceCompletion}^{\uparrow}$ packet fails this NameCert surface if a completion read is accepted without the effective Cauchy row, Cauchy-filter row, located uniform row, Bishop metric-completion handoff, metric-completion boundary, finite alignment window, readback row, and exactness ledger all remaining displayed before the local NameCert row.

Independence witness: The packet is independent of quotient-completion principles: $\mathsf{CauchySpaceCompletion}^{\uparrow}$ binds effective Cauchy schedules, Cauchy filters, located uniform tests, Bishop metric-completion handoff, metric-completion boundary, readback, exactness, transport, replay, provenance, and local naming, but has no row for selected representatives, quotient equality, countable choice, host completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySpaceCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySpaceCompletion}^{\uparrow}$ packet is generated from an EffectiveCauchySpace row, CauchyFilterSpace row, LocatedUniformSpace row, BishopMetricCompletion row, MetricCompletion boundary row, finite alignment window, completion-name readback, exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySpaceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8764.1, Definition 8764.2, Definition 8764.3, and Definition 8764.4 seed the non-quotient Cauchy-space completion carrier and its standard-bridge boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySpaceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limit representatives, countable choice, host completion objects, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, effective-Cauchy routing, filter-Cauchy routing, located-uniform routing, completion readback, exactness nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySpaceCompletionUp.

Chapter 8765

A Concrete Naming Certificate for $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$

The $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ packet realizes the finite-window side of chapter 9244 as concrete NameCert data. It reads the finite prime-window carrier of Definition 9244.1, the local defect rows of Definition 9244.6, the total-defect row of Definition 9244.7, and the completed gluing surface of Definition 9244.10. The packet consumes the prime-row source of chapter 64 and the zeta boundary surface of chapter 71 before its finite gluing ledger may be read. The packet is local to finite prime windows: it does not claim RH, global zeta recovery, nontrivial zero exclusion, Euler-product completeness, or analytic continuation beyond the displayed boundary.

Definition 8765.1 (FinitePrimeWindowGluing carrier)**.** *A* $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, A, D, L, Z, T, R, V, H, C, Q, N)$$

with the following displayed rows.

1. *P is the finite prime-row ledger, listing the displayed window $S \subseteq \mathsf{Prime}^{\uparrow}$ and the prime-local amplitude rows available in that window.*
2. *A is the unit-window amplitude row, recording which displayed prime rows satisfy the local unit-amplitude test.*
3. *D is the local defect-row ledger, recording $D_p(s)$ for every displayed prime row in the finite window.*
4. *L is the gluing ledger, aligning the finite prime rows, defect rows, archimedean boundary request, functional-equation request, pole ledger, zero-ledger request, and branch-transport request.*
5. *Z is the zero-boundary row. It records only the local request for a later zero-ledger exclusion and does not decide any nontrivial zero.*
6. *T is the transport row from finite prime-window data into the completed carrier interface.*
7. *R is the replay row for the local proof route through finite rows, defect accumulation, completed-surface boundary, and zero-boundary request.*
8. *V is the provenance row tying the packet to the finite prime-window, local-defect, and carrier-gluing vision sites.*
9. *H records componentwise* hsame *transport for $P, A, D, L, Z, T, R, V, C, Q, N$.*

10. *C records displayed* Cont *replay for the local gluing request.*

11. *Q is the local replay provenance for the finite carrier data.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrimeWindowGluing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It requires later nontriviality and field-faithfulness input as separate TasteGate *instances before any global zeta or RH-facing consumer may treat the local carrier as an analytic proof object.*

Theorem 8765.2 (FinitePrimeWindowGluing NameCert obligations)**.** *For every accepted* FinitePrimeWindowGluin *carrier $G = (P, A, D, L, Z, T, R, V, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FinitePrimeWindowGluing}^{\uparrow}}$ *surface consists exactly of finite prime-row admission, unit-window amplitude rows, local defect rows, gluing-ledger alignment, zero-boundary request, transport into the completed-carrier interface, replay, provenance, componentwise transport, continuation replay, local replay provenance, and local naming.*

Every accepted read factors through

$$P, A, D \longmapsto L, Z, T \longmapsto R, V, H, C, Q, N$$

and refuses global RH*, recovered zeta completeness, nontrivial zero exclusion, field faithfulness, Euler-product completion, analytic continuation, and archimedean normalization as closed conclusions.*

Proof. Project the accepted carrier. The finite prime-window side is exactly P, A, D: a displayed finite prime set, its unit-amplitude rows, and its local defect rows.

The gluing side starts only after those rows are visible. The row L aligns the finite rows with the archimedean, functional-equation, pole, zero-ledger, and branch-transport requests of the completed surface, while Z records the zero-boundary request without deciding it.

The transport and replay rows T, R, V, H, C, Q, N preserve the local route and provenance. None of these rows contains a nontrivial zero exclusion theorem, a recovered zeta carrier, an analytic continuation proof, a field-faithfulness witness, or a nontriviality gate.

Thus the NameCert surface is the finite gluing packet itself. Any RH-facing use must add the later nontrivial and field-faithful TasteGate instances outside this local carrier. □

Theorem 8765.3 (Finite prime-window gluing boundary)**.** *A* $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ *packet may be consumed by an RH route only as local carrier data: finite prime rows, defect rows, gluing ledger, zero-boundary request, transport, replay, provenance, and local NameCert naming. It does not supply the later* Nontrivial *and* FieldFaithful TasteGate *instances required for global zeta recovery, and it does not close* RH *or any nontrivial zero-exclusion theorem.*

Proof. The consumer first reads the finite rows P, A, D and the gluing ledger L. These rows are local to a displayed finite prime window and cannot by themselves expose the completed zeta carrier.

The zero-boundary row Z is only a request boundary. The transport and replay rows carry the request toward the completed surface, but they do not add the missing nontriviality and field-faithfulness gates.

Since the carrier has no row for global zeta recovery, nontrivial zero exclusion, Euler-product completeness, analytic continuation, or closed RH evidence, every accepted read remains a finite NameCert packet. □

Falsifiable prediction: A future finite prime-window packet fails this NameCert surface if it accepts a gluing read without a displayed finite prime-row ledger, unit-window amplitude rows, local defect rows, gluing ledger, zero-boundary request, transport, replay, provenance, and local NameCert row, or if it treats those rows as a closed proof of RH.

Independence witness: RH, prime-window, zeta-recovery: $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ is not bijective to any listed sibling because it binds finite prime rows, defect rows, local gluing, zero-boundary request, transport, replay, provenance, and local naming while explicitly leaving nontriviality, field faithfulness, recovered zeta completeness, analytic continuation, and global zero exclusion outside the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrimeWindowGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ packet is generated from finite prime rows, unit-window amplitude rows, local defect rows, a gluing ledger, zero-boundary request, transport into the completed-carrier interface, replay, provenance, componentwise hsame transport, displayed Cont replay, local replay provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrimeWindowGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8765.1, Definition 8765.2, and Definition 8765.3 seed the finite local gluing packet for prime-window RH routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrimeWindowGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, recovered zeta completeness, nontrivial zero exclusion, Euler-product completion, analytic continuation, archimedean normalization, nontrivial TasteGate evidence, field-faithful TasteGate evidence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prime-row exactness, defect-row exactness, gluing-ledger alignment, zero-boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FinitePrimeWindowGluin global routes require later Nontrivial and FieldFaithful TasteGate instances.

Chapter 8766

A Concrete Naming Certificate for $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$

Following Bishop–Bridges finite-net practice on compact intervals, the $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$ packet names the finite refinement tree through which an interval mesh is repeatedly subdivided before a completion-facing consumer reads a finite window. It sits between the dyadic interval cells of chapter 5966, the finite mesh surface of chapter 7835, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records finite refinement data only; it is not an open-cover compactness theorem, a selected infinite branch, a quotient real equality principle, a countable-choice operation, or an ambient completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Definition 8766.1 (Finite mesh refinement tree carrier)**.** *A* $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (N, M, R, W, Q, E, H, C, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *source row, listing the dyadic interval cells inspected by the refinement request.*
2. *M is the* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *mesh row, recording the finite dyadic mesh at the root scale.*
3. *R is the refinement-tree row, a finite parent-child ledger whose children are dyadic subintervals of the displayed parent cell.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the displayed mesh and refinement requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite windows and dyadic mesh ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after N, M, R, W, Q have already been displayed.*
7. *H records componentwise* hsame *transport for the net, mesh, refinement tree, window, readback, seal, replay, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a finite refinement request through N, M, R, W, Q, E.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{FiniteDyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{FiniteMeshRefinementTree}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite branch, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8766.2 (Finite refinement tree Real-window boundary)**.** *Every accepted* $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$ *packet exports completion-facing data only through the finite route*

$$\mathsf{DyadicIntervalNet}^{\uparrow} \longrightarrow \mathsf{FiniteDyadicMesh}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the refinement tree supplies a finite parent-child mesh ledger before any regular-Cauchy or real-name consumer can inspect a terminal seal.

Proof. Project the carrier of Definition 8766.1. The only interval cells available to a consumer are the displayed $\mathsf{DyadicIntervalNet}^{\uparrow}$ cells and the finite mesh row M.

The refinement row R is finite, so every child request is replayed through the same displayed parent-child ledger. The consumer then opens only the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows attached to that ledger.

The terminal seal E is downstream of these finite rows. Since the carrier contains no coordinate for the excluded principles, the $\mathsf{Real}^{\uparrow}$ read is a finite-window handoff rather than an appeal to open-cover compactness, selected branches, quotient equality, countable choice, or ambient real completeness. □

Theorem 8766.3 (Finite mesh refinement tree NameCert obligations)**.** *For every accepted* FiniteMeshRefinementTr *packet, the local* $\mathsf{NameCert}_{\mathsf{FiniteMeshRefinementTree}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *exposure,* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *root mesh exposure, finite parent-child refinement ledger,* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8766.1. The rows N, M, R display the interval-net source, root mesh, and finite parent-child refinement ledger before any window request is available.

The window and readback rows W, Q are downstream of that finite tree. The structural rows H, C, P, L transport, replay, source, and name the same packet, so no additional NameCert obligation can enter from outside the displayed carrier.

Finally apply Definition 8766.2. It keeps the downstream seal route on the finite $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{FiniteDyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ path, excluding open-cover compactness, selected branches, quotient equality, countable choice, and ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMeshRefinementTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$ packet is generated from a DyadicIntervalNet source row, a FiniteDyadicMesh root mesh, a finite refinement-tree ledger, StreamName finite windows, RegSeqRat readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteMeshRefinementTree}^{\uparrow}}$, seed)

Scope closed: Definition 8766.1, Definition 8766.2, and Definition 8766.3 seed the finite mesh-refinement route from dyadic interval nets to regular-Cauchy and RealUp finite-window consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteMeshRefinementTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement-tree boundary, finite-window handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMeshRefinementTreeUp.

Chapter 8767

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$

Following Bishop–Bridges regular Cauchy completion, the $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$ packet names the finite pullback of a requested real precision to the stream windows and regular-rational rows that witness it. It refines the diagonal-limit route of section 9238.1 by sitting beside the finite observation discipline of chapter 126, the regular rational readback packet of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet records a finite pullback surface for regular-Cauchy windows; it is not a selected limit, quotient stream equality, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8767.1 (Regular Cauchy window pullback carrier)**.** *A* $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (P, S, R, D, E, H, C, Q, L)$$

with the following displayed rows:

1. *P is the precision request row, naming the requested real-window budget and its finite regular-Cauchy tail index.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at exactly the indices demanded by P.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the finite dyadic comparisons used to transport the requested precision through S and R.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after P, S, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the precision, window, readback, tolerance, seal, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay from a precision request through P, S, R, D, E.*
8. *Q is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowPullback}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8767.2 (Regular Cauchy window pullback handoff)**.** *Every accepted* $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$ *packet factors a Real-completion precision request through the displayed finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The route exposes the finite stream windows and dyadic tolerance ledger before a terminal $\mathsf{Real}^{\uparrow}$ *seal can be read.*

Proof. Project the carrier of Definition 8767.1. The precision row P fixes the finite tail window to inspect, and the only observations opened by the packet are the coordinates listed in S.

The readback row R interprets those observations as regular rational data, while the dyadic ledger D records the tolerance comparisons needed by the same finite request. These rows are displayed before the seal row is available.

The terminal row E therefore records only the RealUp boundary attached to the finite pullback. Since the carrier contains no coordinate for selected limits, quotient stream equality, countable choice, host real equality, or ambient completeness, every completion-facing read stays on the displayed route. □

Theorem 8767.3 (Regular Cauchy window pullback NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$ *packet, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowPullback}^{\uparrow}}$ *surface consists exactly of the precision request row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *terminal seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8767.1. The row P fixes the requested precision and the row S lists the only stream windows opened by that request.

The rows R, D, E keep the readback, dyadic tolerance ledger, and terminal seal on the same displayed finite route. The structural rows H, C, Q, L then transport, replay, source, and name that route without adding selected limits or quotient stream equality.

Apply Definition 8767.2. It identifies the finite completion-facing route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the NameCert surface is exactly the packet displayed by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyWindowPullback}^{\uparrow}$ packet is generated from a precision request row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8767.1, Definition 8767.2, and Definition 8767.3 seed the finite-window pullback route for regular-Cauchy and RealUp completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window pullback handoff, tolerance transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindowPullbackUp.

Chapter 8768

A Concrete Naming Certificate for CompactlyGeneratedWeakHausdorff$^{\uparrow}$

Following Kelley and Steenrod, the CompactlyGeneratedWeakHausdorff$^{\uparrow}$ packet records the compactly generated weak Hausdorff interface as finite BEDC NameCert data. It sits after the topology carrier of chapter 88, the core-compact neighbourhood route of chapter 6350, the compact-open topology ledger of chapter 4138, and the Kelleyfication boundary of chapter 6830. The packet names compact-window tests, weak-Hausdorff diagonal separation, transport, replay, provenance, and local naming rows only; it does not import a host category of compactly generated spaces, quotient-open equality, choice-selected compact tests, or ambient separation theorems.

Definition 8768.1 (Compactly generated weak Hausdorff carrier)**.** *A* CompactlyGeneratedWeakHausdorff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, K, O, D, W, H, C, P, N)$$

with the following displayed rows:

1. *T is the* Topology$^{\uparrow}$ *source row whose indexed opens and incidence data are already admitted as finite BEDC carrier rows.*
2. *K is the compact-generation test row, recording the displayed compact probes through which openness may be inspected.*
3. *O is the* CompactOpenTopology$^{\uparrow}$ *compact-open window ledger, listing the compact-domain windows and open-target tests available to function-space consumers.*
4. *D is the weak-Hausdorff diagonal-separation row. It records the displayed compact-source tests by which a diagonal read is separated, without exporting a host equality theorem for points.*
5. *W is the finite inspection window row for compact probes, diagonal tests, and compact-open subbasic reads.*
6. *H records componentwise* hsame *transport for the topology source, compact-generation tests, compact-open windows, diagonal-separation data, inspection windows, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the same compact-window and diagonal-separation route.*
8. *P is the* Pkg *provenance over* CompactlyGeneratedWeakHausdorff$^{\uparrow}$*,* Topology$^{\uparrow}$*,* CoreCompact$^{\uparrow}$*,* CompactOpenTopology$^{\uparrow}$*,* Kelleyfication$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these nine rows. It has no coordinate for a host category of spaces, quotient-open equality, a selected family of compact probes, classical point separation, countable choice, or an ambient compactly generated topology theorem.

Theorem 8768.2 (Compactly generated weak Hausdorff NameCert obligations)**.** *For every accepted* $\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}$ *carrier* $G = (T, K, O, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactlyGener}}$ *surface consists exactly of topology-source exposure through* T*, compact-generation tests through* K*, compact-open window exposure through* O*, weak-Hausdorff diagonal-separation exposure through* D*, finite inspection through* W*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8768.1. The analytic rows are exactly T, K, O, D, W, and the structural rows are exactly H, C, P, N.

The compact-generation side is therefore read from K and O. The first row records compact probes over the source topology, while the second records the compact-open window ledger used by function-space consumers.

The separation side is read from D. It is a diagonal-separation row attached to the same compact-source tests, so it cannot provide host equality of points or an ambient Hausdorff theorem outside the carrier.

Finally, transport, replay, provenance, and naming are supplied by H, C, P, N for those same rows. Since the packet has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8768.3 (Compactly generated weak Hausdorff diagonal separation)**.** *Let* $G = (T, K, O, D, W, H, C, P, N)$ *be an accepted* $\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}$ *carrier. Every weak-Hausdorff read of* G *must expose the topology source* T*, the compact-generation tests* K*, the diagonal-separation row* D*, and the finite inspection row* W *before any topology or function-space consumer may consume the separation result. The read exports no host diagonal closedness theorem, quotient point equality, selected compact family, countable choice, or ambient separation principle.*

Proof. Start with the carrier projection. The only topology-facing source is T, and the only compact-test source is K.

The diagonal-separation datum is then forced through D. Because D is a row of the same packet, it is read with the finite inspection window W rather than by consulting an external diagonal predicate.

The remaining rows bind the read to the accepted packet. Componentwise transport and displayed replay preserve the route through H and C, while provenance and local naming record the same dependency through P and N.

Thus the consumer receives a compact-window diagonal-separation surface and not a host closed-diagonal theorem, quotient equality, choice-selected compact family, or ambient separation principle. □

Theorem 8768.4 (Compactly generated weak Hausdorff compact-window scope)**.** *For every accepted* $\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}$ *carrier* $G = (T, K, O, D, W, H, C, P, N)$*, every compact-open or Kelleyfication-facing consumer is scoped by the ordered route*

$$T \longmapsto K \longmapsto O \longmapsto D \longmapsto W \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route supplies compact-generation tests, compact-open windows, and weak-Hausdorff diagonal separation as displayed rows of one finite carrier, and it supplies no host compactly generated space, detached mapping-space topology, quotient-open equality, selected compact tests, or ambient separation theorem.

Proof. Read T, K, O first. These rows place the source topology, compact-generation tests, and compact-open ledger before any downstream function-space or Kelleyfication consumer can inspect the packet.

Read D and W next. The diagonal-separation surface and the finite inspection window are attached to the same compact-window route, so the weak-Hausdorff condition is not detached from the compact-generation data.

Finally read H, C, P, N. Transport, replay, provenance, and local naming preserve the displayed row order and add no independent topology or separation coordinate.

Therefore the compact-window scope is exactly the finite route displayed in the statement. Any read outside that route would require a carrier field that G does not contain. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}$ packet is generated from a Topology source row, compact-generation tests, CompactOpenTopology windows, weak-Hausdorff diagonal-separation data, finite inspection windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8768.1, Definition 8768.2, Definition 8768.3, and Definition 8768.4 seed the finite compactly generated weak Hausdorff route over topology, compact-generation, compact-open, diagonal-separation, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactlyGeneratedWeakHausdorff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host category of compactly generated weak Hausdorff spaces, quotient-open equality, selected compact-test families, countable choice, classical point separation, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-generation test exactness, compact-open window scope, diagonal-separation routing, transport and replay stability, provenance, local naming, and nonescape for BEDC.Derived.CompactlyGeneratedWeakHausdorffUp.

Chapter 8769

A Concrete Naming Certificate for $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$

The $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ packet realizes the static boundary-flux branch of Definition 9346.50 as finite graph-Laplacian NameCert data. It sits beside the finite-volume boundary ledger of chapter 6277, the Gauss flux-boundary route of chapter 8156, the displayed matrix surface of chapter 49, and the rational finite-place product carrier of chapter 8761. The packet names static reduced-Laplacian exactness only; it does not identify the dynamic Green response, a running susceptibility, a host PDE solution, or a selected boundary trace.

Definition 8769.1 (Static Green boundary-flux carrier)**.** *A* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, I, B, L, A, E, Z, T, R, P, N)$$

with the following displayed rows:

1. *G is the finite weighted graph row, listing retained vertices, boundary vertices, edge weights, and incidence windows.*
2. *I is the retained-block row for the internal vertex set on which the reduced Dirichlet problem is read.*
3. *B is the boundary-cut row, naming the deleted boundary block and the cut weights from retained vertices into that block.*
4. *L is the reduced Laplacian row over I, with diagonal entries read from full weighted degree and off-diagonal entries read from retained-retained edge weights.*
5. *A is the cut-source row, recording the vector whose retained coordinate is the total edge weight from that vertex into B.*
6. *E is the exactness row, displaying the identity $L\mathbf{1}_I = A$ before any Green-class read is consumed.*
7. *Z is the localized-denominator row. It records the finite determinant and the designated unit factors without turning the cokernel read into a dynamic Green response.*
8. *T records componentwise* hsame *transport for graph, retained-block, boundary-cut, reduced-Laplacian, source, exactness, denominator, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay of the static cut-source route.*

10. *P is the* Pkg *provenance over* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$, $\mathsf{FiniteVolume}^{\uparrow}$, $\mathsf{GaussDivergenceTheorem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{RationalFinitePlaceProduct}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StaticGreenBoundaryFlux}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a dynamic Green operator, a Green–Kubo susceptibility, a host finite-volume theorem, quotient equality of sources, selected traces, or a physical running parameter.

Theorem 8769.2 (Static Green boundary-flux NameCert obligations)**.** *For every accepted* StaticGreenBoundaryFlu *carrier $S = (G, I, B, L, A, E, Z, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StaticGreenBoundaryFlux}^{\uparrow}}$ *surface consists exactly of finite graph exposure through G, retained-block exposure through I, boundary-cut exposure through B, reduced-Laplacian exposure through L, cut-source exposure through A, exactness exposure through E, localized-denominator exposure through Z, componentwise transport through T, displayed replay through R, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8769.1. The finite graph and cut data are exactly G, I, B, while the algebraic readback rows are L, A, E, Z.

The reduced Laplacian and cut-source are not external objects. The row L is computed from the retained graph incidence in G, I, and the row A is computed from the boundary-cut row B.

The exactness claim is carried only by E. It displays $L\mathbf{1}_I = A$ before the localized-denominator row Z is inspected, so the packet cannot claim Green-exactness while hiding the reduced Laplacian or the cut source.

Finally, T, R, P, N transport, replay, source, and name the same finite route. Since no dynamic Green operator, susceptibility, host finite-volume theorem, quotient source equality, selected trace, or physical running parameter occurs in the carrier, the listed rows exhaust the local obligations. □

Theorem 8769.3 (Static Green boundary-flux exactness)**.** *Let $S = (G, I, B, L, A, E, Z, T, R, P, N)$ be an accepted* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ *carrier. Any static boundary-flux read that claims Green-exactness must expose the reduced Laplacian row L and the cut-source row A before reading the exactness row E. The visible exactness is the finite identity $L\mathbf{1}_I = A$, and it exports no dynamic Green response, no Green–Kubo susceptibility, no selected boundary trace, and no host finite-volume theorem.*

Proof. Read the graph, retained-block, and boundary-cut rows first. They determine which vertices remain internal and which edge weights contribute to the boundary source.

Project the reduced Laplacian row L. Its diagonal entries use the full weighted degree of each retained vertex, while its off-diagonal entries use only retained-retained edge weights.

Project the cut-source row A. Each coordinate records exactly the omitted boundary weights from the same retained vertex into B.

The row E then displays the coordinatewise identity $L\mathbf{1}_I = A$: multiplying L by the constant retained vector cancels retained-retained edge contributions and leaves precisely the boundary-cut coordinate. The transport, replay, provenance, and local naming rows preserve that static identity, and no row supplies the excluded dynamic or host-theorem data. □

Theorem 8769.4 (Static Green boundary-flux vision realization)**.** *The* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ *carrier realizes the static part of Definition 9346.50: a Window-six or finite graph boundary read is accepted only when the rows $G, I, B, L, A, E, Z, T, R, P, N$ expose the reduced Laplacian, cut source, exactness identity, localized denominator, transport, replay, provenance, and local naming. The*

realization is static and exact; the dynamic Green response of Definition 9346.51 remains a separate readout surface.

Proof. Start with Definition 8769.1. Its rows bind the finite graph, retained block, deleted boundary block, reduced Laplacian, and cut-source vector before any exactness read is public.

Apply Definition 8769.3. The exactness row is the displayed identity $L\mathbf{1}_I = A$, so the static boundary source is a reduced-Laplacian image inside the same finite packet.

The localized-denominator row Z records which determinant factors remain visible after the designated finite-place localization. It does not create a running Green operator and does not turn the zero static cokernel class into a dynamic susceptibility.

Thus the vision is realized only as static finite NameCert data. The dynamic centered relaxation and Green-excess quantities cited by the vision require their own packet and cannot be recovered from S. □

Falsifiable prediction: Any accepted $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ read that claims static Green-exactness must expose the reduced Laplacian row and the cut-source row before the exactness row is consumed; a read that hides either row refutes the carrier of Definition 8769.1.

Independence witness: finite-volume, gauss-divergence-theorem, matrix, rational-finite-place-product: $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite graph incidence, retained-block data, boundary-cut rows, reduced-Laplacian rows, cut-source rows, exactness, localized-denominator data, transport, replay, provenance, and local naming, while the siblings isolate conservation ledgers, flux-boundary calculus, matrix display, or finite-place arithmetic.

The Lean-side taste surface for $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ must eventually provide ‘Nontrivial StaticGreenBoundaryFluxUp‘ and ‘FieldFaithful StaticGreenBoundaryFluxUp‘ instances in ‘TasteGate.lean‘, so that nontrivial boundary-flux content and finite-field faithful readback are checked before stronger closure is claimed.

Closure status for $\mathsf{NameCert}_{\mathsf{StaticGreenBoundaryFlux}^{\uparrow}}$.

Constructive story: A $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$ packet is generated from a finite weighted graph row, retained-block row, boundary-cut row, reduced-Laplacian row, cut-source row, exactness row, localized-denominator row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StaticGreenBoundaryFlux}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8769.1, Definition 8769.2, Definition 8769.3, and Definition 8769.4 seed the finite static boundary-flux route over reduced Laplacians, cut sources, exactness, localized denominators, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StaticGreenBoundaryFlux}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close dynamic Green response, Green–Kubo susceptibility, host finite-volume theorems, selected traces, physical running parameters, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reduced-Laplacian construction, cut-source construction, exactness, localized-denominator faithful readback, transport and replay stability, provenance, local naming, Nontrivial instance, and FieldFaithful instance for BEDC.Derived.StaticGreenBoundaryFluxUp.

Chapter 8770

A Concrete Naming Certificate for $\mathsf{WeylCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{WeylCauchyCriterion}^{\uparrow}$ packet records the finite BEDC route by which a stream of rational approximants is certified Cauchy before it is sealed as a real. It consumes the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the dyadic rational core of chapter 291, the $\mathsf{CauchyModulus}^{\uparrow}$ request surface of chapter 292, and the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet is a finite Cauchy-criterion certificate; it does not quotient streams, choose limits, or import ambient completeness.

Definition 8770.1 (Weyl Cauchy Criterion carrier)**.** *A* $\mathsf{WeylCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row naming the rational stream whose tails are inspected.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row storing the finite request-to-tail assignment used by the criterion.*
3. *W is the finite tail-window ledger: for each displayed precision request it names the window beyond which the stream comparison is read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row carrying the dyadic tolerance and dyadic distance witnesses used inside each tail window.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream observations and dyadic comparisons.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the stream, modulus, window, dyadic, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for S, M, W, D, Q, E and for replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of every consumer route through*

$$S, M, W, D, Q \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{WeylCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeylCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or a standard bridge.

Theorem 8770.2 (Weyl Cauchy Criterion NameCert obligations)**.** *For every accepted* $\mathsf{WeylCauchyCriterion}^{\uparrow}$ *carrier*

$$K = (S, M, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeylCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the displayed stream name, Cauchy modulus, tail-window ledger, dyadic comparison row, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*
2. *tail-window stability, requiring every transported or replayed precision request to keep the same stream S, modulus M, window W, dyadic comparison D, and readback Q before exposing the real seal E;*
3. *modulus extraction, requiring the tail index used by each criterion request to be read from M and not from a chosen subsequence, selected limit, or ambient completeness theorem;*
4. *Real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$*-facing consumer to pass through S, M, W, D, Q, E and to receive no quotient representative or host real equality from the packet;*
5. *refusal of quotient streams, selected limits, countable choice, trichotomy, and ambient completeness, requiring H, C, P, N to transport, replay, source, and name only the displayed rows.*

No Weyl Cauchy Criterion NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 8770.1. The packet displays the stream, modulus, tail windows, dyadic comparisons, regular readback, real seal, and the four structural rows.

Tail-window stability is componentwise. A transported or replayed request keeps the same S, M, W, D, Q route, so a consumer cannot replace the displayed tail comparison by a quotient over streams.

Modulus extraction is forced by the M row. Each precision request reads its tail index from the displayed Cauchy modulus, and the dyadic comparison row D checks only the window named by W.

The real seal E appears after S, M, W, D, Q have been exposed. Thus a $\mathsf{Real}^{\uparrow}$ consumer receives a finite criterion route rather than a selected limit, a quotient representative, or an ambient completeness theorem.

Finally, H, C, P, N only transport, replay, source, and name the carrier rows. Since the packet contains no countable choice, trichotomy, quotient stream equality, selected limit, Lean verification, or bridge credential, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeylCauchyCriterion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeylCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8770.1 and Definition 8770.2 seed the finite Weyl Cauchy Criterion surface over stream names, regular rational readback, dyadic comparisons, Cauchy modulus extraction, and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeylCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, tail-window stability, modulus extraction, Real-seal nonescape, and ledger refusal for BEDC.Derived.WeylCauchyCriterionUp.

Constructive story: A WeylCauchyCriterion$^{\uparrow}$ packet is generated from a StreamName row, a CauchyModulus row, finite tail windows, DyadicRatCore comparisons, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8771

A Concrete Naming Certificate for $\mathsf{DyadicIntervalContinuityModulus}^{\uparrow}$

Following Bishop–Bridges constructive practice, the $\mathsf{DyadicIntervalContinuityModulus}^{\uparrow}$ packet names a finite dyadic interval modulus for a continuous map on a located closed interval. It sits beside the dyadic interval cells of chapter 5966, the located finite-net interval surface of chapter 1808, the finite observation discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records finite modulus data only; it is not an open-cover compactness theorem, a choice-selected modulus, quotient-real equality, or ambient completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Definition 8771.1 (Dyadic interval continuity modulus carrier)**.** *A* $\mathsf{DyadicIntervalContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, N, L, S, R, E, B, H, C, P, Q)$$

with the following displayed rows:

1. *I is the located closed interval row carrying the two dyadic endpoint certificates and the located membership test used by the packet.*
2. *N is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row listing the dyadic subinterval cells inspected at the chosen mesh scale.*
3. *L is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *row certifying that the listed cells are the finite located interval net consumed by the packet.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the function reads at the cell endpoints and selected interior probes.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those finite stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed readbacks.*
7. *B is the modulus-budget row: for each requested output dyadic tolerance it displays a finite input mesh scale and the cell-window ledger that witnesses the corresponding oscillation bound.*
8. *H records componentwise* hsame *transport for the interval, net, located-net, stream-window, rational-readback, real-seal, budget, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay from a requested tolerance through I, N, L, S, R, E, B.*

10. *P is the* Pkg *provenance over* DyadicIntervalContinuityModulus$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, LocatedClosedIntervalFiniteN StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *Q is the local* NameCert$_{\text{DyadicIntervalContinuityModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, choice-selected subcovers, quotient-real equality, host real equality, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8771.2 (Dyadic interval modulus window stability)**.** *Every accepted* DyadicIntervalContinuityModulus$^{\uparrow}$ *packet preserves its modulus decision under componentwise* hsame *transport of the displayed interval, net, finite-window, readback, real-seal, budget, replay, provenance, and local-name rows.*

Proof. Project the carrier of Definition 8771.1. The modulus budget row B is not a standalone host function; it is read only together with the interval row I, the dyadic net N, the located finite-net certificate L, the StreamName windows S, the RegSeqRat readback R, and the RealUp seal E.

Now transport the packet componentwise along the displayed hsame row H. Because H compares exactly the rows listed in the carrier, it preserves the cell ledger, the opened stream windows, the rational readback, and the requested output tolerance attached to B.

The replay row C then replays the same finite route after transport. Hence a consumer that reads the transported packet sees the same finite modulus decision, and no new open cover, selected subcover, quotient-real equality, or completeness principle is introduced. □

Theorem 8771.3 (Dyadic interval modulus regular-Cauchy handoff)**.** *Every accepted* DyadicIntervalContinuityMod *packet exports completion-facing data only through the finite route*

$$\text{DyadicIntervalNet}^{\uparrow} \longrightarrow \text{LocatedClosedIntervalFiniteNet}^{\uparrow} \longrightarrow \text{StreamName}^{\uparrow} \longrightarrow \text{RegSeqRat}^{\uparrow} \longrightarrow \text{Real}^{\uparrow}.$$

The modulus budget is therefore a finite regular-Cauchy window handoff, not an appeal to open-cover compactness or ambient Real$^{\uparrow}$ *completeness.*

Proof. The source of a completion-facing read is the dyadic interval net N together with the located finite-net row L. These rows determine the finite cells and probes available to the packet.

The only observation route then passes through the StreamName windows S and the RegSeqRat readback row R. The terminal seal E records that the displayed rational windows have been presented as a RealUp-facing name, but it does not add a selected limit or a completeness theorem.

Finally the budget row B is tied to those same finite windows by the replay row C and the local NameCert row Q. Since no excluded principle is a coordinate of the carrier, every completion consumer receives exactly the finite regular-Cauchy handoff displayed in the statement. □

Closure status for NameCert$_{\text{DyadicIntervalContinuityModulus}^{\uparrow}}$.

Constructive story: A DyadicIntervalContinuityModulus$^{\uparrow}$ packet is generated from a located closed interval row, a DyadicIntervalNet row, a LocatedClosedIntervalFiniteNet row, StreamName finite windows, RegSeqRat readback, a RealUp terminal seal, a finite modulus-budget ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicIntervalContinuityModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8771.1, Definition 8771.2, and Definition 8771.3 seed the finite dyadic interval modulus route from located interval nets to regular-Cauchy and RealUp consumers.

Formal verification: FormalStatus(NameCert$_{\text{DyadicIntervalContinuityModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected modulus extraction, quotient-real equality, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window stability, regular-Cauchy handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalContinuityModulusUp.

Chapter 8772

A Concrete Naming Certificate for $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$

The $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$ packet realizes the zero-fiber route of chapter 9253 as finite NameCert data. It binds a zeta-domain source row, a displayed zero-value fiber, legal generator factorization, fixed-half ledger rows, componentwise transport, replay, provenance, and local naming before any consumer may read the fixed-half sufficiency interface. The packet is narrower than the analytic zeta carrier and wider than a bare zero row: it is a concrete bridge from the vision's zero-type carrier to BEDC packet data, with sibling context from chapter 8761 and chapter 4238.

Falsifiable prediction: If the zero-fiber row is absent, no accepted packet can prove the fixed-half sufficiency theorem for this carrier.

Independence witness: zeta, prime, rational-finite-place-product, metacic-parallel-diamond-frontier: the packet has a zero-fiber/generator/fixed-half ledger not bijective to Euler-product, primality, finite-place-product, or residual-diamond carriers.

Definition 8772.1 (Zeta zero type factorization carrier)**.** *A* $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (D, V, F, G, T, J, K, H, C, P, N)$$

with the following displayed rows:

1. *D is the zeta-domain source row, carrying the continuation-domain provenance and the pole-exclusion ledger used by the zero-type vision.*
2. *V is the zeta-value relation row, displaying a finite witness that the source row has a value readout.*
3. *F is the zero-fiber row, displaying the value witness together with the zero-complex classifier row of Definition 9253.4.*
4. *G is the legal-generator factorization row, recording that every admitted generator read factors through the same displayed zero fiber.*
5. *T is the non-trivial strip row, recording the critical-strip witness required before the fixed-half interface is meaningful.*
6. *J is the fixed-half ledger row, recording the critical-line witness supplied to the sufficiency consumer.*

7. *K is the TasteGate-facing row requiring later Lean files to provide* Nontrivial ZetaZeroTypeFactorization$^{\uparrow}$ *and* FieldFaithful ZetaZeroTypeFactorization$^{\uparrow}$ *instances for the displayed carrier fields.*
8. *H records componentwise* hsame *transport for the domain, value, zero-fiber, generator, strip, fixed-half, TasteGate, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay from a legal generator through D, V, F, G, T, J, K.*
10. *P is the* Pkg *provenance over* ZetaZeroTypeFactorization$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *the zero-type vision,* RationalFinitePlaceProduct$^{\uparrow}$, *and* MetaCICParallelDiamondFrontier$^{\uparrow}$.
11. *N is the local* NameCert$_{\text{ZetaZeroTypeFactorization}^{\uparrow}}$ *row.*

The classifier accepts only packets in which F, G, T, J occur in the same finite carrier. It has no coordinate for Euler-product reconstruction, primality, finite-place product equality, residual-diamond confluence, analytic continuation proof, host complex equality, or a proof of RH.

Theorem 8772.2 (Zeta zero type factorization zero-fiber boundary)**.** *Every accepted* ZetaZeroTypeFactorization$^{\uparrow}$ *packet exposes the zero-type route through the displayed row F before any generator or fixed-half consumer can read the packet.*

Proof. Project the carrier of Definition 8772.1. The source row D and value row V are not yet a zero-type packet; the zero-fiber row F is the displayed witness that the value readout is classified with the zero complex history.

The generator row G is downstream of F. A legal generator is accepted only after its output is routed through the same zero-fiber row, so it cannot bypass the zero-type carrier described in chapter 9253.

The fixed-half rows T, J are also downstream of F. They refine a non-trivial zero packet by strip and critical-line ledgers, but they do not create the zero witness. Thus the zero-fiber row is the boundary through which every accepted packet must pass. □

Theorem 8772.3 (Zeta zero type factorization ledger)**.** *For the carrier of Definition 8772.1, the local* NameCert$_{\text{ZetaZeroTypeFactorization}^{\uparrow}}$ *surface consists exactly of the finite zero-fiber row, generator-factorization row, fixed-half ledger rows, TasteGate-facing row, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Read the rows D, V, F first. They display the source, the value relation, and the zero-fiber witness. Without F, the packet is merely a zeta-value readout and cannot feed the fixed-half sufficiency interface.

Next read G, T, J. The generator row records factorization through the displayed zero fiber, while the non-trivial strip row and fixed-half row record the two ledgers required by the RH-facing subtype route. These are finite ledger rows, not an analytic proof of the Riemann hypothesis.

Finally read K, H, C, P, N. The TasteGate row names the future Nontrivial and FieldFaithful obligations for the same carrier fields, and the transport, replay, provenance, and local-name rows exhaust the BEDC certificate surface. No Euler-product, primality, finite-place-product, residual-diamond, host-equality, or analytic-continuation coordinate is available to a consumer. □

Closure status for NameCert$_{\text{ZetaZeroTypeFactorization}^{\uparrow}}$.

Constructive story: A ZetaZeroTypeFactorization$^{\uparrow}$ packet is generated from a zeta-domain source row, a value-relation row, a zero-fiber row, a legal-generator factorization row, non-trivial strip and fixed-half ledger rows, a TasteGate-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ZetaZeroTypeFactorization}^{\uparrow}}$, seed)

Scope closed: Definition 8772.1, Definition 8772.2, and Definition 8772.3 seed the zero-fiber, generator-factorization, and fixed-half ledger surface for the RH zero-type route.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZetaZeroTypeFactorization}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Euler-product reconstruction, primality, finite-place product equality, residual-diamond confluence, analytic continuation, host complex equality, a proof of RH, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-fiber boundary, generator-factorization ledger, fixed-half ledger, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.ZetaZeroTypeFactorizationUp.

Chapter 8773

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$ packet names the finite envelope through which a Cauchy-modulus request is routed before the L10 Real and Completion spine reads a terminal seal. It sits beside the finite observation discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the directed modulus comparison surface of chapter 2386, and the terminal real-name seal of chapter 38. The packet binds those sibling routes into one displayed finite envelope; it is not a selected limit, quotient stream equality principle, countable-choice operation, host modulus function, or ambient completeness claim for $\mathsf{Real}^{\uparrow}$.

Definition 8773.1 (Bishop Cauchy modulus envelope carrier)**.** *A* $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, R, D, J, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus request row, naming the requested dyadic precision and the finite tail window demanded by a Bishop-style Cauchy argument.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened exactly at the indices exposed by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those stream windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the finite dyadic comparisons used by the request.*
5. *J is the* $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ *comparison row, placing the displayed request inside the directed modulus lattice before a stronger request is consumed.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, S, R, D, J have all been displayed.*
7. *H records componentwise* hsame *transport for the modulus request, stream window, regular readback, dyadic ledger, lattice comparison, seal, replay, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a modulus request through M, S, R, D, J, A.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient representatives, selected limits, countable choice, a host modulus oracle, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8773.2 (Bishop Cauchy modulus envelope NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$ *carrier* $E = (M, S, R, D, J, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEnvel}}$ *surface consists exactly of carrier admission for the modulus request row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ *comparison,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8773.1. The row M is the only source of requested precision, and the row S opens only the finite windows demanded by that request.

The readback row R and dyadic ledger D interpret the same displayed window. The lattice row J then compares the request with finite directed-modulus data before any stronger completion-facing request is consumed.

The terminal row A is downstream of M, S, R, D, J. The structural rows H, C, P, N transport, replay, source, and name that same finite route, so the NameCert obligations are exactly the displayed rows and no hidden completion principle is available. □

Theorem 8773.3 (Bishop Cauchy modulus envelope Real-seal boundary)**.** *Every accepted* BishopCauchyModulusEr *packet exposes its* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{DyadicCauchyModulusLattice}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Thus the envelope ties a Bishop Cauchy modulus request to Real and Completion consumers without importing quotient equality, selected limits, countable choice, a host modulus oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8773.2. The accepted carrier displays the modulus request before any finite stream window is opened, and the regular rational readback and dyadic comparisons remain attached to that request.

The comparison row J is the only component that relates the request to a dyadic Cauchy modulus lattice. Any stronger tolerance request must therefore pass through the displayed directed lattice row rather than through an external modulus function.

The seal row A is exposed only after this finite route has been replayed. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, host modulus data, or ambient completeness, the Real-facing read cannot leave the displayed envelope. □

Falsifiable prediction: A $\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}$ packet fails this NameCert surface if a Real or Completion consumer can read the terminal seal without first displaying the modulus request, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, DyadicCauchyModulusLattice comparison row, transport, replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, dyadic-cauchy-modulus-lattice, real: no listed sibling carries the whole envelope $E = (M, S, R, D, J, A, H, C, P, N)$, because each sibling supplies only finite windows, readback, tolerance arithmetic, directed modulus comparison, or terminal Real sealing while this packet binds the request-to-seal route into one local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEnvelope}^{\uparrow}}$.

Constructive story: A BishopCauchyModulusEnvelope$^{\uparrow}$ packet is generated from a modulus request row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a DyadicCauchyModulusLattice comparison row, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopCauchyModulusEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 8773.1, Definition 8773.2, and Definition 8773.3 seed the finite Bishop Cauchy modulus envelope over StreamName, RegSeqRat, DyadicRatCore, DyadicCauchyModulusLattice, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopCauchyModulusEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, host modulus functions, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus request exposure, finite-window readback, dyadic tolerance routing, lattice comparison, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyModulusEnvelopeUp.

Chapter 8774

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$

Following Bishop and Bridges, completion is a constructive metric operation on displayed Cauchy data rather than a passage to a hidden quotient object. The $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$ packet records the finite NameCert seed for flattening a Cauchy completion of a Cauchy completion. It sits after the Cauchy completion monad law surface of chapter 3834, the bind route of chapter 3777, the multiplication sibling of chapter 1829, the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38. The packet names the displayed flattening service only; it is not a quotient completion, a selected limit, host equality of completed points, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8774.1 (Cauchy completion monad multiplication carrier)**.** *A* $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (O, I, B, L, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the outer completion row, naming the completion-of-completion request before any flattening read is public.*
2. *I is the inner completion row, naming the regular Cauchy family whose completion has already been admitted.*
3. *B is the bind row inherited from* $\mathsf{CauchyCompletionBind}^{\uparrow}$*, displaying the source family and the completion-facing bind schedule.*
4. *L is the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *ledger row, carrying the finite associativity and unit comparisons used by the flattening route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the outer request, inner request, and flattened read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-stability row attached to the displayed stream windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the precision budget and triangle slack for the flattening comparison.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after O, I, B, L, S, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the outer, inner, bind, law, window, readback, dyadic, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route from nested completion data to the flattened classifier.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}}$ *row.*

The flattened classifier compares accepted packets only through $O, I, B, L, S, R, D, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host equality of completed points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.CauchyCompletionMonadMultiplicationUp`

Theorem 8774.2 (Flattening scope)**.** *For every accepted* $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$ *carrier*

$$J = (O, I, B, L, S, R, D, E, H, C, P, N),$$

the flattening map from a nested completion request to its flattened classifier depends only on the nested Cauchy completion rows O, I, the bind and law rows B, L, the regular-Cauchy tail stability recorded by S, R, D, and classifier transport through H, C, P, N. Any consumer of the flattened classifier must read those rows before the terminal Real seal E is available. The map exports no quotient completion class, selected limit, host equality of completed points, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 8774.1. The nested completion request is visible only through the outer and inner rows O, I, while the completion-facing computation is visible only through the bind row B and the monad-law ledger L.

The regular-Cauchy tail data are read next. The stream windows S, regular readback R, and dyadic tolerance ledger D determine the finite tail-stability comparison used by the flattened classifier before the seal row E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Since none of the excluded principles has a carrier coordinate, the flattening map cannot depend on them. □

Theorem 8774.3 (Cauchy completion monad multiplication NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$ *carrier $J = (O, I, B, L, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}}$ *surface consists of exactly five finite obligation rows:*

1. *carrier admission: the accepted object is the displayed* BHist *packet over nested completion, bind, law, StreamName, RegSeqRat, DyadicRatCore, Real seal, transport, replay, provenance, and local naming rows;*

2. *nested-source visibility: O and I must be displayed before the flattening route is read;*

3. *bind-and-law exactness: B and L supply the completion bind schedule and the finite associativity ledger used by the flattened classifier;*

4. *tail-stability transport: S, R, D, E, H, C preserve the regular-Cauchy windows, readback, dyadic tolerances, Real seal, componentwise transport, and displayed replay;*

5. *provenance and local naming: P and N bind the packet to* Pkg *and* NameCert *without adding quotient completion, selected representatives, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier admission row first. It exposes the twelve displayed coordinates of J and no other data.

For nested-source visibility, inspect O and I. These rows are the only source of completion-of-completion information, so the flattened route cannot be read before they are present.

For bind-and-law exactness, inspect B and L. The bind row supplies the completion-facing schedule, and the law row supplies the finite associativity ledger used to compare the two possible replays of a nested request.

For tail-stability transport, inspect S, R, D, E, H, C. These rows preserve the StreamName windows, regular readback, dyadic tolerances, terminal Real seal, componentwise transport, and displayed continuation replay across the flattening step.

Finally P and N provide package provenance and local naming for the same displayed packet. Since the carrier has no row for the excluded principles, the five listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8774.1, Definition 8774.2, and Definition 8774.3 seed the finite completion-of-completion flattening surface over $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source visibility, bind-and-law exactness, tail-stability transport, Real-seal nonescape, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadMultiplicationUp.

Constructive story: A $\mathsf{CauchyCompletionMonadMultiplication}^{\uparrow}$ packet is generated from outer and inner completion rows, bind and monad-law rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8775

A Concrete Naming Certificate for $\mathsf{GelfandTriple}^{\uparrow}$

Following Gelfand and Vilenkin's rigged Hilbert space tradition, the $\mathsf{GelfandTriple}^{\uparrow}$ packet records the finite NameCert surface for a test carrier sitting densely inside a Hilbert carrier and read against a displayed dual carrier. It sits beside the Sobolev carrier of chapter 5911, the Schwartz carrier of chapter 5877, the tempered-distribution handoff of chapter 2310, and the dual-space interface of chapter 8458. The chapter names the $\Phi \to H \to \Phi'$ sandwich as finite BEDC rows; it is not a host locally convex completion, a selected distribution representative, spectral decomposition, reflexivity theorem, countable choice, or ambient Hilbert-space duality.

Definition 8775.1 (Gelfand triple carrier)**.** *A* $\mathsf{GelfandTriple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (\Phi, H, D, I, J, P, N)$$

with the following displayed rows:

1. *Φ is the test carrier row. It records the finite test-function or finite-energy windows read from sibling packets such as* $\mathsf{SchwartzSpace}^{\uparrow}$ *and* $\mathsf{SobolevSpace}^{\uparrow}$*.*
2. *H is the Hilbert carrier row. It records the inner-product and norm-facing readback surface through which Φ is interpreted as a dense Hilbert-facing test domain.*
3. *D is the dual carrier row. It records the displayed* $\mathsf{DualSpace}^{\uparrow}$ *or* $\mathsf{TemperedDistribution}^{\uparrow}$ *action rows that consume test windows from Φ.*
4. *I is the dense inclusion row $\Phi \to H$. It records that every Hilbert-facing read of the test carrier must pass through the displayed inclusion and its finite density ledger.*
5. *J is the dual inclusion and pairing row $H \to D$. It records the pairing of a Hilbert row against a dual row only after the test row, Hilbert row, and dense inclusion row have been displayed.*
6. *P is the* Pkg *provenance over* $\mathsf{GelfandTriple}^{\uparrow}$*,* $\mathsf{SobolevSpace}^{\uparrow}$*,* $\mathsf{SchwartzSpace}^{\uparrow}$*,* $\mathsf{TemperedDistribution}^{\uparrow}$*,* $\mathsf{DualSpace}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{GelfandTriple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through Φ, H, D, I, J, P, N. It has no coordinate for a host locally convex topology, unrestricted continuous dual, selected representative, spectral basis, countable-choice schedule, or ambient Hilbert-space completeness theorem.

Theorem 8775.2 (Gelfand triple NameCert obligations)**.** *For the carrier of Definition 8775.1, the local* $\mathsf{NameCert}_{\mathsf{GelfandTriple}^{\uparrow}}$ *surface consists exactly of test-carrier admission, Hilbert-carrier admission, dual-carrier admission, dense inclusion exposure for* $\Phi \to H$*, dual-pairing exposure for* $H \to \Phi'$*,* Pkg *provenance, and local naming. Every accepted read is confined to the displayed route*

$$\Phi, I, H, J, D \longmapsto P, N.$$

The surface exports no host locally convex completion, unrestricted dual equality, selected distribution representative, spectral decomposition, countable choice, or ambient Hilbert-space duality theorem.

Proof. Carrier admission is projection from Definition 8775.1. The rows Φ, H, D, I, J, P, N are the only rows available to a consumer.

The inclusion obligations are ordered. The row I is the only dense route from the test carrier into the Hilbert carrier, while J is the only pairing route from the Hilbert carrier toward the displayed dual carrier.

The provenance and naming rows preserve that same finite packet. Since no excluded topology, unrestricted dual, selected representative, spectral basis, choice schedule, or ambient duality theorem appears as a coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8775.3 (Gelfand triple density sandwich)**.** *Every Hilbert-facing consumer of an accepted* $\mathsf{GelfandTriple}^{\uparrow}$ *carrier must read the test row* Φ*, the dense inclusion row* I*, and the Hilbert row* H *before consuming any dual-facing row. Thus the packet exposes a finite density sandwich*

$$\Phi \longmapsto H \longmapsto D$$

without replacing Φ*,* H*, or* D *by host completions or quotient representatives.*

Proof. Project the accepted carrier. The only test row is Φ, the only Hilbert row is H, and the only row connecting them is I.

Apply the NameCert obligations of Definition 8775.2. They place dense inclusion exposure before any dual-pairing exposure, so a Hilbert-facing read cannot skip directly to D.

The structural rows P, N source and name the same ordered route. Therefore the public analytic shape available from the packet is the displayed density sandwich, not a host completion or quotient construction. □

Theorem 8775.4 (Gelfand triple dual-pairing ledger)**.** *Every dual-pairing read of an accepted* $\mathsf{GelfandTriple}^{\uparrow}$ *carrier factors through the ledger*

$$\Phi, H, D, I, J, P, N.$$

The pairing row J *may be consumed only after the test carrier, Hilbert carrier, dual carrier, and dense inclusion rows have been displayed. The ledger does not export unrestricted distribution equality, a selected dual representative, reflexive identification of* H *with* D*, spectral decomposition, countable choice, or ambient completeness.*

Proof. Read the carrier rows from Definition 8775.1. The dual carrier D is visible, but it is paired with Hilbert-facing data only through J.

Use the density sandwich theorem. It forces the route through Φ, I, H before a dual-facing read may use the Hilbert row.

Finally apply the local obligations. The row J is the only dual-pairing ledger, and P, N only source and name that ledger. Since the excluded dual, spectral, choice, and completeness objects are absent from the packet, every dual-pairing read remains inside the displayed finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{GelfandTriple}^{\uparrow}}$.

Constructive story: A $\mathsf{GelfandTriple}^{\uparrow}$ packet is generated from a test carrier row, Hilbert carrier row, dual carrier row, dense inclusion row, dual-pairing ledger, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GelfandTriple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8775.1, Definition 8775.2, Definition 8775.3, and Definition 8775.4 seed the finite rigged-Hilbert sandwich and its dual-pairing boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GelfandTriple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex completion, unrestricted dual equality, selected distribution representatives, spectral decomposition, reflexivity, countable choice, ambient Hilbert-space duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense inclusion exposure, dual-pairing exactness, provenance, and local naming for BEDC.Derived.GelfandTripleUp.

Chapter 8776

A Concrete Naming Certificate for $\mathsf{Prohorov}^{\uparrow}$

Following Prohorov on tight probability families and weak compactness, the $\mathsf{Prohorov}^{\uparrow}$ packet records the finite BEDC surface where a probability family may be read through displayed tightness ledgers, metric compactness witnesses, and weak-convergence test windows. It sits over the probability source of chapter 184, the distribution readback of chapter 186, the metric distance vocabulary of chapter 57, and the compact metric witness route of chapter 129. The packet names only the finite dependency route for tight probability data; it does not import ambient weak compactness, select subsequences, quotient measures, countable choice, or an external topology of measures.

Definition 8776.1 (Prohorov carrier)**.** *A* $\mathsf{Prohorov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, D, T, K, W, M, R, V, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{ProbSpace}^{\uparrow}$ *probability source row, naming the total-mass and measurable-event surface from which the family is read.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *readback row, recording the pushed-forward probability endpoints inspected by the packet.*
3. *T is the tightness ledger, displaying the finite list of compact windows and tolerance requests that must be met before any weak-convergence read is public.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness row for the compact windows named by T.*
5. *W is the weak-convergence test-window row, containing the finite family of bounded continuous tests and tolerance comparisons currently admitted.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare test-window endpoints and compact-window neighborhoods.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the displayed probabilities, tolerances, and test-window endpoint comparisons.*
8. *V records transport of probability mass through the displayed distribution rows and compact windows.*

9. *H records componentwise* hsame *transport for $S, D, T, K, W, M, R, V, C, Q, N$.*
10. *C records displayed* Cont *replay from a Prohorov request through probability source, distribution readback, tightness ledger, compact metric witness, weak-convergence tests, metric comparison, Real readback, and transport.*
11. *Q is the* Pkg *provenance over* $\mathsf{Prohorov}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Prohorov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, K, W, M, R, V, H, C, Q, N$. It has no coordinate for selected subsequences, quotient probability measures, ambient weak compactness, countable choice, or a host topology on all measures.

Theorem 8776.2 (Prohorov NameCert obligations)**.** *For every accepted* $\mathsf{Prohorov}^{\uparrow}$ *carrier*

$$P = (S, D, T, K, W, M, R, V, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Prohorov}^{\uparrow}}$ *surface consists exactly of probability-source admission through S, distribution readback through D, tightness-ledger exposure through T, compact metric witness exposure through K, weak-convergence test-window admission through W, metric comparison through M, Real readback through R, transport through V and H, displayed replay through C, provenance through Q, and local naming through N.*

No Prohorov obligation is stored outside these rows.

Proof. Project the accepted carrier. The probability source row S and the distribution row D are the only measure-facing data available to the packet, so every probability endpoint is read through the displayed source and pushforward surfaces.

The tightness ledger T must be displayed before a compact-window witness can be consumed. The compact metric row K supplies only the finite compact windows named by the ledger, not an ambient compactness theorem or a selected subsequence.

The weak-convergence test row W, metric row M, and Real readback row R then record the finite comparisons that are public. The transport, replay, provenance, and local naming rows V, H, C, Q, N preserve the same route, so the listed rows exhaust the local NameCert obligations. □

Theorem 8776.3 (Prohorov tightness metric handoff)**.** *Every consumer that reads a weak-convergence or compactness-facing conclusion from an accepted* $\mathsf{Prohorov}^{\uparrow}$ *carrier must first pass through the probability source row S, distribution readback row D, tightness ledger T, compact metric witness K, weak-convergence test window W, metric comparison row M, and Real readback row R. The handoff is the finite route*

$$S, D, T, K, W, M, R, V, H, C, Q, N$$

and exports no selected subsequence, quotient measure equality, countable-choice schedule, ambient weak compactness principle, or host topology of probability measures.

Proof. Start with the NameCert obligations of Definition 8776.2. They place source admission, distribution readback, tightness exposure, compact metric witnessing, weak test windows, metric comparison, and Real readback inside one accepted packet.

Read the tightness row before the compact metric row. This forces every compactness-facing consumer to inspect the displayed compact windows and tolerance requests before using any compact metric witness.

Next read the weak-convergence test window and metric comparison rows. These rows expose only finite tests and metric endpoint comparisons; they do not choose a subsequence or introduce an external topology of measures.

Finally replay V, H, C, Q, N. Transport, componentwise equality, continuation replay, provenance, and local naming preserve the same finite handoff, so the consumer cannot replace the route by quotient measure equality, countable choice, or ambient weak compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{Prohorov}\uparrow}$**.**

Constructive story: A $\mathsf{Prohorov}^{\uparrow}$ packet is generated from probability source rows, distribution readback rows, tightness ledgers, compact metric witnesses, weak-convergence test windows, metric comparison rows, Real readback rows, transport, componentwise hsame, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Prohorov}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8776.1, Definition 8776.2, and Definition 8776.3 seed the finite Prohorov packet over probability sources, distributions, tightness ledgers, compact metric witnesses, weak test windows, metric comparisons, Real readback, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Prohorov}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subsequences, quotient measures, countable choice, ambient weak compactness, a host topology of measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked probability-source admission, distribution readback, tightness-ledger exposure, compact metric witnessing, weak-test routing, metric comparison, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.ProhorovUp.

Chapter 8777

A Concrete Naming Certificate for $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$

Following Bishop and Bridges on constructive metric fixed-point methods, the $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$ packet records the finite route by which displayed contraction data generates a Cauchy endpoint realizer before any ambient completed fixed point is named. It sits after the contraction modulus surface of chapter 7948, the finite density-net route of chapter 6003, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed finite endpoint-realizer route; it is not a global Banach fixed-point theorem, a selected limit, quotient representative data, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8777.1 (Contractive Cauchy realizer carrier)**.** *A* $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, O, D, W, Q, E, F, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{DyadicContractionModulus}^{\uparrow}$ *row, recording the contraction budget and the dyadic shrinkage rule consumed by every endpoint request.*
2. *O is the Picard-orbit row, naming the finite iterate window generated from the displayed start row and the contraction self-map.*
3. *D is the* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *row, recording the finite density and tolerance cells through which the orbit is compared.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing exactly the orbit observations inspected at the requested precision.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window and contraction budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after K, O, D, W, Q have been displayed.*
7. *F is the endpoint certificate row, recording that the finite Picard window, density-net comparison, and regular-rational readback jointly supply the displayed endpoint realizer.*
8. *H records componentwise* hsame *transport for $K, O, D, W, Q, E, F, C, P, N$.*
9. *C records displayed* Cont *replay from a precision request through the contraction budget, orbit window, density net, stream window, readback, Real seal, endpoint row, transport, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$, $\mathsf{DyadicContractionModulus}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContractiveCauchyRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, D, W, Q, E, F, H, C, P, N$. It has no coordinate for selected limits, quotient representatives, countable-choice schedules, trichotomy branches, host fixed-point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8777.2 (Contractive Cauchy realizer NameCert obligations)**.** *For every accepted* ContractiveCauchyRealiz *carrier $R = (K, O, D, W, Q, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContractiveCauchyRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the contraction modulus, Picard orbit, finite Cauchy density net, finite stream window, regular rational readback, terminal Real seal, endpoint certificate, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every endpoint-facing consumer must expose the same finite route*

$$K \to O \to D \to W \to Q \to E \to F \to H \to C \to P \to N$$

before it treats the packet as contraction-driven Cauchy endpoint data.

Proof. Project the carrier of Definition 8777.1. Its displayed coordinates are exactly $K, O, D, W, Q, E, F, H, C, P, N$, so the local naming surface has no hidden row outside the packet.

The contraction modulus K determines the finite shrinkage budget used by the Picard orbit row O. The density-net row D and stream window W then identify the finite observations through which the requested tolerance is inspected.

The regular rational readback Q is attached to those observations, and the terminal Real seal E is downstream of that readback. The endpoint certificate F records the fact that this finite route is the endpoint-facing data, while H, C, P, N transport, replay, source, and name the same route.

Since none of the rows stores selected limits, quotient representatives, countable-choice schedules, trichotomy branches, host fixed-point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 8777.3 (Contractive Cauchy route)**.** *Every Cauchy read exported by an accepted* ContractiveCauchyRealiz *carrier factors through the contraction modulus, Picard orbit, finite Cauchy density net, stream window, and regular rational readback rows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is available. In the notation of Definition 8777.1, the route is*

$$\mathsf{DyadicContractionModulus}^{\uparrow},\ O,\ \mathsf{FiniteCauchyDensityNet}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Thus the packet supplies a finite Cauchy endpoint-realizer route rather than a selected completed fixed point.

Proof. Open the finite route named in Definition 8777.2. The first row K fixes the dyadic contraction budget, and O records only the Picard iterates visible in the finite orbit window.

The density-net row D compares those iterates through finite cells. The stream window W exposes the inspected observations, and the regular rational row Q reads those observations at the requested precision.

Only after those rows have been displayed can the Real seal E be read. Therefore each Cauchy read factors through the finite contraction, orbit, density, window, and readback route, with no selected limit or ambient completeness principle added to the packet. □

Theorem 8777.4 (Contractive Cauchy fixed-endpoint seal)**.** *Let $R = (K, O, D, W, Q, E, F, H, C, P, N)$ be an accepted* $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$ *carrier. Any consumer that reaches the endpoint certificate F must first read the terminal* $\mathsf{Real}^{\uparrow}$ *seal E through the displayed Cauchy route of Definition 8777.3, and must then read H, C, P, N as the transport, replay, provenance, and naming rows for that same endpoint certificate. The seal does not assert global Banach fixed-point existence, selected limits, quotient representatives, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The endpoint certificate F is a row of the carrier, not a detached fixed-point selector. By Definition 8777.1, it appears only after the contraction modulus, Picard orbit, density net, stream window, regular rational readback, and Real seal rows have been displayed.

Apply Definition 8777.3 to the seal row E. The consumer therefore reaches E only through the finite Cauchy readback route and then reaches F as the named endpoint certificate for that route.

Finally H, C, P, N record transport, replay, provenance, and local naming for the same packet. Because the classifier contains no coordinate for global Banach fixed-point existence, selected limits, quotient representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the fixed-endpoint seal is exactly the displayed finite endpoint-realizer seal. □

Falsifiable prediction: Every accepted $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$ endpoint consumer exposes a DyadicContractionModulus row, Picard-orbit row, FiniteCauchyDensityNet row, StreamName window, RegSeqRat readback, RealUp seal, endpoint certificate, hsame transport, Cont replay, Pkg provenance, and local NameCert row before it receives endpoint data.

Independence witness: dyadic-contraction-modulus, finite-cauchy-density-net, streamname, regseqrat, real: no listed sibling supplies the joint packet containing contraction budget, Picard orbit, density-net comparison, finite stream window, regular rational readback, terminal Real seal, endpoint certificate, transport, replay, provenance, and local NameCert rows while refusing selected limits and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveCauchyRealizer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContractiveCauchyRealizer}^{\uparrow}$ packet is generated from a DyadicContractionModulus row, Picard orbit row, FiniteCauchyDensityNet row, StreamName window, RegSeqRat readback, a RealUp seal, endpoint certificate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8777.1, Definition 8777.2, Definition 8777.3, and Definition 8777.4 seed the finite contraction-driven Cauchy endpoint-realizer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyRealizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global Banach fixed-point existence, selected limits, quotient representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-modulus exactness, Picard-window routing, finite density-net comparison, StreamName readback, RegSeqRat handoff, Real-seal non-escape, endpoint-certificate sealing, transport, replay, provenance, and local naming for BEDC.Derived.ContractiveCauchyRealizerUp.

Chapter 8778

A Concrete Naming Certificate for $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$

Following Bishop and Bridges and the categorical reflection reading of completion, the CauchyCompletionSeparatedU packet records the separated universal-property boundary for Cauchy completion as finite BEDC NameCert data. It sits after the functorial completion surface of chapter 423, the adjunction row of chapter 1456, the dense embedding of chapter 3351, the separated-reflection boundary of chapter 1790, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names exactly the rows through which a separated target may consume a completion universal arrow; it is not a quotient completion, a selected limit, a host equality principle, or an assertion of ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8778.1 (Cauchy completion separated universal carrier)**.** *A* $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, A, D, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row exposing the source completion surface before any universal-property consumer reads the packet.*
2. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row carrying the unit and extension handoff over the same source data.*
3. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *dense-source row, visible before a separated target is allowed to inspect the completion value.*
4. *S is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *row witnessing that the target-side comparison is separated at the NameCert boundary.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface through which source and image observations are requested.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W and R have been displayed.*
8. *H records componentwise* hsame *transport for the functor, adjunction, dense embedding, separated reflection, window, readback, seal, and local naming rows.*

9. *C records displayed* Cont *replay from a universal-property request through F, A, D, S, W, R, E.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient equality of completion points, selected representatives, choice-selected limits, host equality of maps, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8778.2 (Cauchy completion separated universal obligations)**.** *For every accepted* CauchyCompletionSepar *carrier*

$$U = (F, A, D, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}}$ *surface consists exactly of carrier admission, functor-row exposure, adjunction handoff, dense-embedding exposure, separated-reflection exposure, finite window readback, terminal Real sealing, componentwise transport, continuation replay, provenance, and local naming. No obligation may replace these rows by quotient equality, selected limits, host map equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8778.1. Its public rows are exactly $F, A, D, S, W, R, E, H, C, P, N$.

The universal-property part is read through F, A, D, S. The functor and adjunction rows expose the completion-facing source and extension route, the dense embedding row supplies the source image inside the completion surface, and the separated-reflection row is the only target-side separatedness datum.

The numerical readback is read through W, R, E. Stream windows and regular rational readback are displayed before the terminal Real seal is available.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no quotient, selected-limit, host-map-equality, choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8778.3 (Cauchy completion separated universal factorization)**.** *Every separated-target consumer of an accepted* $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$ *packet factors through*

$\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, SeparatedMetricRe

before any public universal-property read is exposed. The factorization supplies a displayed completion functor row, adjunction handoff, dense-source row, separated-reflection row, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming; it exports no quotient completion, selected representative, choice-selected limit, host map equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the functor and adjunction rows F, A. They identify the completion-facing source and extension route by finite NameCert data.

Next read the dense embedding and separated-reflection rows D, S. The dense image must be visible before the target consumes the completion value, and separatedness is recorded only as the row S.

Then replay the observation route through W, R, E. The consumer receives finite stream windows, regular rational readback, and the terminal Real seal in that order.

The structural rows H, C, P, N preserve and name the same route. Thus a separated-target consumer can only factor through the displayed rows and cannot obtain quotient equality, selected limits, host map equality, choice, or ambient completeness from the packet. □

Theorem 8778.4 (Cauchy completion separated universal nonescape)**.** *Every public read from an accepted* $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$ *packet exposes separatedness only as the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *row S attached to the displayed completion route. The read cannot escape to quotient equality of completion points, chosen representatives, choice-selected limits, host equality of maps, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8778.2. It fixes the packet inventory to the finite rows $F, A, D, S, W, R, E, H, C, P, N$.

Apply Definition 8778.3. Any separated-target consumer reaches the universal-property boundary only after the functor, adjunction, dense embedding, separated-reflection, finite-window, readback, and Real-seal rows are displayed.

The separatedness datum is therefore the visible row S, not an equality principle outside the carrier. Because no excluded object is a coordinate of the packet, no public read can export it. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$ read must expose $F, A, D, S, W, R, E, H, C, P, N$ before a separated universal-property consumer may inspect the completion boundary.

Independence witness: cauchycompletionfunctor, cauchycompletionadjunction, metric-completion-dense-embedding, separatedmetricreflection, streamname, regseqrat, real: no listed sibling jointly carries the separated universal-property packet with functor, adjunction, dense embedding, separated reflection, finite observation, regular readback, Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}$ packet is generated from CauchyCompletionFunctor, CauchyCompletionAdjunction, MetricCompletionDenseEmbedding, SeparatedMetricReflection, StreamName, RegSeqRat, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8778.1, Definition 8778.2, and Definition 8778.3 seed the separated universal-property boundary for Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedUniversal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality, choice-selected limits, host map equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, functor-row exposure, adjunction handoff, dense embedding, separated-reflection visibility, finite readback, Real sealing, and ledger nonescape for BEDC.Derived.CauchyCompletionSeparatedUn

Chapter 8779

A Concrete Naming Certificate for $\mathsf{MarkHistContSpine}^{\uparrow}$

The $\mathsf{MarkHistContSpine}^{\uparrow}$ packet realizes the finite mark-history-continuation spine of chapter 9351 as concrete NameCert data. It binds the raw BMark emissions, generated BHist rows, Ext one-step extension rows, Cont continuation rows, hsame transport, replay, provenance, and local naming into one inspectable packet before any higher BEDC layer reads a continuation trace. The carrier is not a package object, not a signature relation, not a quotient of histories, and not an external normalization theorem.

Definition 8779.1 (MarkHistContSpine carrier)**.** *A* $\mathsf{MarkHistContSpine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, H, E, T, Q, R, P, N)$$

with the following displayed rows:

1. *M is the finite* BMark *emission row, exposing only generated raw marks.*
2. *H is the generated* BHist *history row recording the finite mark sequence read by the packet.*
3. *E is the* Ext *one-step extension ledger from a source history and displayed mark to the resulting history.*
4. *T is the* Cont *continuation trace row, recording the finite concatenation route consumed by downstream replay.*
5. *Q is the continuation-trace normal form row, asserting that the same generated history route is read by the displayed continuation trace rather than by an external quotient.*
6. *R records componentwise* hsame *transport for M, H, E, T, Q and the local classifier row.*
7. *P is the* Pkg *provenance over* $\mathsf{MarkHistContSpine}^{\uparrow}$, BMark, BHist, Ext, Cont, hsame, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkHistContSpine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, H, E, T, Q, R, P, N. It has no coordinate for package formation, signature equality, hidden history quotienting, external normalization, or an unrecorded continuation route.

Theorem 8779.2 (MarkHistContSpine NameCert obligations)**.** *For every accepted* $\mathsf{MarkHistContSpine}^{\uparrow}$ *carrier*

$$K = (M, H, E, T, Q, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkHistContSpine}^{\uparrow}}$ *surface consists exactly of carrier admission for raw mark emissions, generated history rows, one-step extension rows, continuation trace rows, continuation-trace normal form, componentwise* hsame *transport, provenance, and local naming. No obligation may replace those rows by package structure, signature equality, quotient histories, external normalization, or an unrecorded continuation route.*

Proof. Project the carrier of Definition 8779.1. The visible kernel rows are M, H, E, T, Q, and the structural rows are R, P, N.

The raw mark and history obligations are read from M, H. The one-step extension obligation is read from E, and the continuation obligation is read from T.

The normal-form row Q records only that the displayed continuation trace has the stated generated history shape. It does not identify histories by a quotient or by an external conversion principle.

Finally R, P, N transport, source, and name the same finite packet. Since no excluded object is stored in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8779.3 (MarkHistContSpine continuation trace)**.** *Every continuation read from an accepted* $\mathsf{MarkHistContSpine}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{BMark},\ \mathsf{BHist},\ \mathsf{Ext},\ \mathsf{Cont},\ \mathsf{hsame},\ \mathsf{NameCert}.$$

The read exposes raw marks, generated histories, one-step extensions, the continuation trace, trace normal form, componentwise transport, provenance, and local naming before any downstream kernel consumer may inspect the trace.

Proof. Start with the raw mark row M and generated history row H. They expose the finite source material of the trace.

Use the extension row E to read each displayed one-step move. The continuation row T then concatenates those moves inside the generated BHist surface.

Read the normal-form row Q and transport row R last. They keep the trace attached to the same generated history shape without replacing it by quotient history equality or external normalization. □

Theorem 8779.4 (Additive continuation dependency)**.** *Any arithmetic consumer that reads a* $\mathsf{MarkHistContSpine}^{\uparrow}$ *continuation row as an additive continuation must pass through chapter 29 and its unary continuation route. The spine itself supplies only the displayed* BMark, BHist, Ext, *and* Cont *rows, with transport, provenance, and local naming; it does not create* $\mathsf{Add}^{\uparrow}$ *arithmetic or the unary-source discipline required by additive continuation.*

Proof. Project the continuation trace of Definition 8779.3. The rows M, H, E, T, Q, R, P, N expose generated marks, generated histories, one-step extension, displayed continuation, trace normal form, transport, provenance, and naming.

An additive reading needs more than those rows. By Definition 29.1, an accepted $\mathsf{Add}^{\uparrow}$ row first reads its two sources through the unary $\mathsf{Nat}^{\uparrow}$ spine and only then consumes the displayed Cont witness as additive output.

Thus the present spine can hand a continuation trace to a downstream consumer, but the arithmetic interpretation is supplied by the $\mathsf{Add}^{\uparrow}$ sibling route. Without that route, the trace remains a mark-history continuation packet rather than an additive continuation certificate. □

Theorem 8779.5 (MarkHistContSpine nonescape)**.** *Every public consumer of an accepted* $\mathsf{MarkHistContSpine}^{\uparrow}$ *packet receives only the finite rows* M, H, E, T, Q, R, P, N*. The packet cannot export package formation, signature equality, quotient-history identification, external normalization, a hidden continuation route, or any object above the mark-history-continuation spine.*

Proof. Apply the obligations theorem, Definition 8779.2. It fixes the admitted packet surface to raw marks, histories, extension, continuation, normal form, transport, provenance, and local naming.

Apply the continuation-trace theorem, Definition 8779.3. Any trace consumer must read those rows before the continuation trace is public.

The omitted structures are not fields of the carrier. A consumer can therefore use the packet only as a finite kernel-spine NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{MarkHistContSpine}^{\uparrow}$ read that exposes a continuation trace without first displaying M, H, E, T, Q, R, P, N refutes the carrier surface.

Independence witness: continuationtrace, continuationtracenormalform, namecert, core: no listed sibling jointly carries raw BMark emissions, generated BHist rows, Ext one-step rows, Cont trace rows, trace normal form, hsame transport, provenance, and local NameCert data as one finite kernel-spine packet.

Future Lean work for this packet must exhibit nontriviality and field faithfulness as TasteGate instances: a Nontrivial witness separates at least two accepted spine carriers, and FieldFaithful confirms that the displayed M, H, E, T, Q, R, P, N fields are the classifier surface consumed by the local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkHistContSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkHistContSpine}^{\uparrow}$ packet is generated from BMark emissions, BHist generated histories, Ext one-step rows, Cont continuation trace rows, continuation-trace normal form, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkHistContSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8779.1, Definition 8779.2, and Definition 8779.3 seed the finite mark-history-continuation spine packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkHistContSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close package formation, signature equality, quotient histories, external normalization, Lean verification, TasteGate instance checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, raw mark exposure, generated history exposure, extension rows, continuation trace rows, trace normal form, nonescape, and TasteGate field faithfulness for BEDC.Derived.MarkHistContSpineUp.

Chapter 8780

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ packet records the finite tail-bound row by which a Bishop regular Cauchy name exposes its eventual dyadic control before any Real or completion-facing consumer reads it. It sits beside the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the shift-stability route of chapter 1578, and the terminal Real seal of chapter 38. The packet names only a displayed tail-bound certificate; it does not assert quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8780.1 (Bishop regular Cauchy tail-bound carrier)**.** *A* $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, Q, N, W, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, listing the finite windows from which the Cauchy name is read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the windows as regular rational data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius used for the tail request.*
4. *Q is the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *row, recording that the same tail request is stable under a displayed finite shift of the observation window.*
5. *N is the unary tail-index row, naming the finite stage after which the dyadic radius is enforced.*
6. *W is the later-window comparison row, displaying only those later observations that the packet certifies inside the radius from D after N.*
7. *H records componentwise* hsame *transport for $S, R, D, Q, N, W, C, P, L$.*
8. *C records displayed* Cont *replay from the stream and regular readback rows through*

$$S, R, D, Q, N, W \longmapsto H, C, P, L.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailBound}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, R, D, Q, N, W, H, C, P, L$. It has no coordinate for a quotient completion object, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness theorem.

Theorem 8780.2 (Bishop regular Cauchy tail-bound NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ *carrier*

$$T = (S, R, D, Q, N, W, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailBound}^{\uparrow}}$ *surface consists exactly of StreamName observation exposure through S, RegSeqRat readback through R, dyadic-radius accounting through D, shift-stable tail routing through Q, tail-index exposure through N, later-window comparison through W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8780.1. The accepted packet displays exactly the stream, readback, dyadic tolerance, shift, tail-index, later-window, transport, replay, provenance, and local naming rows.

Read the tail request in order. A consumer first opens S and R, then consumes the dyadic radius D, then checks that the same request survives the shift row Q before reading N and W.

The structural rows preserve only that finite route. The hsame and Cont rows transport and replay the displayed comparisons, while P and L source and name the packet. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles is part of the local NameCert surface. □

Theorem 8780.3 (Bishop regular Cauchy tail-bound Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyShift}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff supplies only the displayed tail-bound packet for downstream Real sealing. It does not supply quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 8780.2. It forces every Real-facing read through S, R, D, Q, N, W before the structural rows may be consumed.

The completion-facing part is finite. Stream windows are repacked as regular rational readback, the dyadic row supplies the radius, the shift row keeps the request stable under displayed finite shifts, and the tail-index plus later-window rows provide the bound.

Inspect the missing coordinates of the carrier. The packet contains no quotient stream, selected representative, countable-choice schedule, trichotomy decision, host equality row, or ambient completeness theorem. Thus a downstream Real or completion consumer receives only the finite tail-bound handoff. □

Falsifiable prediction: A candidate $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ packet fails this NameCert surface if it accepts a Real-facing tail-bound read without displayed StreamName observations, RegSeqRat readback, dyadic tolerance, shift-stable routing, tail index, later-window comparison, transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, regular-cauchy-shift, real: BishopRegularCauchyT is not bijective to any listed sibling because it stores the Bishop tail-index and later-window comparison together with shift-stable routing and a Real-facing handoff, while excluding quotient streams, selected limits, countable choice, trichotomy, host real equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailBound}\uparrow}$.

Constructive story: A $\mathsf{BishopRegularCauchyTailBound}^{\uparrow}$ packet is generated from StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance row, a RegularCauchyShift stability row, a unary tail-index row, later-window comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailBound}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8780.1, Definition 8780.2, and Definition 8780.3 seed the finite Bishop tail-bound packet over stream observations, regular rational readback, dyadic radii, shift-stable routing, tail indices, later-window comparisons, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailBound}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, countable choice, quotient stream equality, selected limits, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-index exposure, later-window comparison, dyadic-radius accounting, shift-stable routing, replay stability, and ledger nonescape for BEDC.Derived.BishopRegularCauchyTailBoundUp.

Chapter 8781

A Concrete Naming Certificate for $\mathsf{CauchyEntourageCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ packet names the finite L10 completion route in which Cauchy-entourage filter data, regular basis rows, uniform completion data, finite stream windows, regular rational readback, dyadic tolerances, and the terminal Real seal are displayed before any completion-facing consumer reads the packet. It sits after the Cauchy-entourage filter surface of chapter 1791, the entourage-base discipline of chapter 1819, the regular Cauchy entourage-basis packet of chapter 3805, and the uniform entourage completion boundary of chapter 8620. The packet is a displayed NameCert route for completion data; it is not a quotient filter, selected representative, selected limit, countable-choice schedule, host equality of completion points, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 8781.1 (Cauchy entourage completion carrier)**.** *A* $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, B, A, U, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *source row, exposing Cauchy requests as finite entourage-filter observations before any completion read is public.*
2. *B is the* $\mathsf{EntourageBase}^{\uparrow}$ *row, naming the base entourages and refinement ledger consumed by the Cauchy request.*
3. *A is the* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *row, aligning regular Cauchy windows with the displayed entourage-base generators.*
4. *U is the* $\mathsf{UniformEntourageCompletion}^{\uparrow}$ *row, recording the uniform completion handoff read downstream of F, B, A.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the source, regular basis, and uniform completion reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the entourage, regularity, and completion comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, B, A, U, S, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the Cauchy-entourage filter, entourage base, regular basis, uniform completion, stream, readback, dyadic, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a completion request through the route*

$$F, B, A \longmapsto U, S, R, D \longmapsto E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyEntourageCompletion}^{\uparrow}$, $\mathsf{CauchyEntourageFilter}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$, $\mathsf{UniformEntourageCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyEntourageCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, B, A, U, S, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, selected representatives, selected limits, countable-choice schedules, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8781.2 (Cauchy entourage completion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ *carrier $K = (F, B, A, U, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyEntourageCompl}}$ *surface consists exactly of carrier admission for the displayed rows, CauchyEntourageFilter source exposure through F, EntourageBase refinement exposure through B, RegularCauchyEntourageBasis alignment through A, UniformEntourageCompletion handoff through U, StreamName window preservation through S, RegSeqRat readback preservation through R, DyadicRatCore tolerance preservation through D, Real-seal preservation through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. No local NameCert clause is stored outside those rows.*

Proof. Project the carrier of Definition 8781.1. The projection exposes exactly the source, base, regular-basis, uniform-completion, finite-window, readback, dyadic, Real-seal, transport, replay, provenance, and local naming rows.

The source route is read first. A Cauchy-entourage request must display F, then open the base-entourage row B, and then align that base with the regular Cauchy entourage-basis row A before the uniform completion row U is available.

The L10 readback route is downstream of that source route. StreamName windows, RegSeqRat readback, and DyadicRatCore tolerances are read as the shared rows S, R, D before the terminal seal E can be consumed.

Finally attach H, C, P, N. These rows transport, replay, source, and name only the displayed packet. Since the carrier has no quotient, selected representative, selected limit, choice schedule, host-equality, or ambient completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8781.3 (Cauchy entourage completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyEntourageFilter}^{\uparrow},\ \mathsf{EntourageBase}^{\uparrow},\ \mathsf{RegularCauchyEntourageBasis}^{\uparrow} \longmapsto \mathsf{UniformEntourageCompletion}^{\uparrow},\ \mathsf{Strean}$$

In carrier notation, every such read passes through

$$F, B, A \longmapsto U, S, R, D \longmapsto E, H, C, P, N.$$

The handoff exports no quotient filter, selected representative, selected limit, countable-choice schedule, host equality of completed points, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness premise.*

Proof. Apply the NameCert obligation surface of Definition 8781.2. It confines every accepted Real-facing read to the displayed carrier rows.

Read the source side in row order. The Cauchy-entourage filter row F, the entourage-base row B, and the regular Cauchy entourage-basis row A must be visible before the uniform completion handoff row U can be opened.

Read the finite completion side next. The StreamName row S, RegSeqRat row R, and DyadicRatCore row D provide the common observation and tolerance route that reaches the seal row E.

The structural rows preserve this route. Componentwise hsame, displayed Cont replay, Pkg provenance, and the local NameCert row record the same finite handoff, so no omitted quotient, selector, choice, host-equality, or ambient-completeness coordinate can be introduced downstream. □

Falsifiable prediction: Every accepted $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ read exposes $\mathsf{CauchyEntourageFilter}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$, $\mathsf{UniformEntourageCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows before a Real-facing completion consumer can inspect the terminal seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyEntourageCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyEntourageCompletion}^{\uparrow}$ packet is generated from CauchyEntourageFilterUp source data, EntourageBaseUp refinement data, RegularCauchyEntourageBasisUp alignment data, UniformEntourageCompletionUp handoff data, finite StreamNameUp windows, RegSeqRatUp readback, DyadicRatCoreUp tolerances, a RealUp seal, and structural NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyEntourageCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8781.1, Definition 8781.2, and Definition 8781.3 seed the L10 Cauchy-entourage completion route over CauchyEntourageFilterUp, EntourageBaseUp, RegularCauchyEntourageBasisUp, UniformEntourageCompletionUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyEntourageCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient filters, selected representatives, selected limits, countable choice, host equality, host Real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-entourage source exposure, entourage-base refinement, regular-basis alignment, uniform completion handoff, finite readback preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyEntourageCompletionUp.

Chapter 8782

A Concrete Naming Certificate for $\mathsf{LocatedRealFastCauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRealFastCauchy}^{\uparrow}$ packet records the finite bridge by which located-real interval data is read as a fast Cauchy and regular-rational real name before any completion-facing consumer receives it. It sits after the Bishop located-real surface of chapter 6180, the fast sequence surface of chapter 2185, the fast real-name carrier of chapter 1589, and the two-way comparison route of chapter 1656. The packet is a finite L10 bridge from located intervals to fast and regular Cauchy readback; it is not quotient real equality, a selected representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8782.1 (LocatedRealFastCauchy carrier)**.** *A* $\mathsf{LocatedRealFastCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, I, M, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located-real interval data row, carrying the finite interval enclosures, strict-side evidence, and located-cut requests inherited from the* $\mathsf{BishopLocatedReal}^{\uparrow}$ *surface.*
2. *I is the interval-refinement witness row, recording the finite refinement steps by which later intervals remain compatible with the earlier located enclosure.*
3. *M is the fast-Cauchy modulus row, displaying the threshold schedule that sends each requested tolerance to a finite sequence window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same interval refinements and fast-Cauchy windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording dyadic radii, comparison slack, midpoint arithmetic, and endpoint budgets used by L, I, M, R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after L, I, M, R, D have been displayed.*
7. *H records componentwise* hsame *transport for located intervals, refinements, modulus rows, readback rows, dyadic tolerances, Real seal handoff, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for every located-interval request, fast threshold request, regular readback request, and Real seal request through the same finite route.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealFastCauchy}^{\uparrow}$, $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{FastCauchySequence}^{\uparrow}$, $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealFastCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, I, M, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8782.2 (LocatedRealFastCauchy NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealFastCauchy}^{\uparrow}$ *carrier $K = (L, I, M, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealFastCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, located-interval exposure, interval-refinement exposure, fast-Cauchy modulus exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the displayed rows $L, I, M, R, D, E, H, C, P, N$.*

Proof. Project the accepted packet. The analytic rows are precisely L, I, M, R, D, E: located intervals first, refinement evidence next, the fast modulus route next, regular rational readback next, dyadic tolerance accounting alongside those reads, and the terminal Real seal last.

Read the sibling obligations through provenance. The row L is the located-real interface of chapter 6180; the row M is the fast sequence and fast real-name schedule visible through chapter 2185 and chapter 1589; and the row R is the regular readback disciplined by chapter 1656.

Transport, replay, provenance, and local naming do not add analytic content. The rows H, C, P, N preserve and name the same finite route, so quotient streams, selected limits, countable-choice schedules, trichotomy, host equality, and ambient completeness remain outside the local certificate. □

Theorem 8782.3 (LocatedRealFastCauchy regular readback)**.** *Every accepted* $\mathsf{LocatedRealFastCauchy}^{\uparrow}$ *carrier $K = (L, I, M, R, D, E, H, C, P, N)$ supplies a regular readback route*

$$\mathsf{BishopLocatedReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{FastCauchySequence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route reads the located interval row L, interval refinements I, fast threshold row M, dyadic tolerance row D, regular readback row R, and terminal seal E before any Real-facing or completion-facing consumer may inspect the packet. It supplies no detached uniform limit, quotient stream, selected representative, countable-choice schedule, trichotomy decision, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the located row L and refinement row I. These rows display the finite interval enclosures and the refinement witnesses that keep later precision requests attached to the same located-real surface.

Use the fast row M and the tolerance row D. A requested precision is routed to a finite sequence window and dyadic comparison budget before any regular rational readback can be consumed.

Read R and then E. The $\mathsf{RegSeqRat}^{\uparrow}$ row is downstream of the same located, refinement, fast, and dyadic rows, and the $\mathsf{Real}^{\uparrow}$ seal is terminal with respect to that route. Componentwise transport and replay preserve these rows, while provenance and local naming bind the packet without adding any detached limit, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealFastCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealFastCauchy}^{\uparrow}$ packet is generated from located-real interval data, interval-refinement witnesses, fast-Cauchy modulus rows, RegSeqRat readback, DyadicRatCore tolerance rows, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealFastCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8782.1, Definition 8782.2, and Definition 8782.3 seed the located-real to fast-Cauchy and regular-readback bridge over finite L10 rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealFastCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval exposure, interval-refinement exactness, fast-modulus exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealFastCauchyUp.

Chapter 8783

A Concrete Naming Certificate for $\mathsf{SequentialPrecompact}^{\uparrow}$

Following Bishop and Bridges, sequential precompactness is witnessed by finite nets and regular readback windows rather than by selected convergent subsequences or completed quotient streams. The $\mathsf{SequentialPrecompact}^{\uparrow}$ packet sits after the finite density-net route of chapter 6003, the dyadic interval net surface of chapter 5966, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It gives L10 compactness and completion consumers a finite inspection route for a displayed sequence source; it is not open-cover compactness, countable choice, selected subsequence extraction, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8783.1 (Sequential precompact carrier)**.** *A* $\mathsf{SequentialPrecompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, N, S, R, E, H, C, P, L)$$

with the following displayed rows:

1. *X is the displayed sequence source row, naming the finite observation interface through which a consumer asks for sequence data.*
2. *N is the finite-net extractor row, read through* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *and* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *before any completion-facing density claim is exposed.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the inspected finite prefixes of the displayed sequence source and net cells.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the finite window and dyadic tolerance data have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for real-facing consumers of the finite-net readback.*
6. *H records componentwise* hsame *transport for sequence source, finite net, windows, readback, seal, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from a sequential-precompactness request through X, N, S, R, E, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{SequentialPrecompact}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{SequentialPrecompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, N, S, R, E, H, C, P, L$. It has no coordinate for quotient streams, selected subsequences, countable choice, open-cover compactness, host sequence equality, or a general completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8783.2 (Sequential precompact NameCert obligations)**.** *For every accepted* $\mathsf{SequentialPrecompact}^{\uparrow}$ *carrier $Q = (X, N, S, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialPrecompact}^{\uparrow}}$ *surface consists exactly of sequence-source admission, finite-net extraction, StreamName window display, RegSeqRat readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 8783.1. The displayed packet has exactly the rows $X, N, S, R, E, H, C, P, L$, so carrier admission and local naming must be read from those rows.

The sequence source row X supplies only inspected sequence data. The finite-net row N supplies the finite density and dyadic-cell route, while S and R record the finite window and regular rational readback through which the sequence is compared to those cells.

The terminal seal E is downstream of the net, window, and readback rows. The rows H, C, P, L preserve, replay, source, and name the same finite route. Since the carrier has no further coordinate, selected subsequences, quotient stream equality, countable choice, open-cover compactness, and ambient completeness cannot become local certificate obligations. □

Theorem 8783.3 (Sequential precompact finite-net handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentialPrecompact}^{\uparrow}$ *packet reads the finite-net row N through chapter 6003 and the dyadic interval cell surface of chapter 5966 before it consumes the* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, or terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff is the displayed route*

$$X, N \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, L.$$

It exports finite dyadic density rows for L10 completion consumers, not quotient streams, selected subsequences, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8783.2. They force the consumer to expose the sequence source, finite net, stream window, readback, and seal rows in the same accepted packet.

Next read N through the finite density-net and dyadic interval-net sibling chapters. The density route supplies the finite Cauchy density witness, and the dyadic interval-net route supplies the finite cell and endpoint-incidence surface used by the requested tolerance.

Finally use the carrier exhaustion in Definition 8783.1. The packet has no row for selected subsequence extraction, quotient stream representatives, countable-choice schedules, open-cover compactness, or ambient completeness. Thus the completion-facing handoff is exactly the displayed finite-net route into S, R, E, H, C, P, L. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialPrecompact}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialPrecompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8783.1, Definition 8783.2, and Definition 8783.3 seed the displayed finite-net route from a sequence source to StreamName windows, RegSeqRat readback, and the terminal RealUp seal.

Constructive story: A $\mathsf{SequentialPrecompact}^{\uparrow}$ packet is generated from a displayed sequence source row, finite density-net and dyadic-cell rows, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialPrecompact}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SequentialPrecompactUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected subsequences, countable choice, open-cover compactness, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked sequence-source admission, finite-net extraction, dyadic-cell handoff, StreamName window coverage, RegSeqRat readback, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.SequentialPrecompactUp.

Chapter 8784

A Concrete Naming Certificate for $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$

Following Baire, Oxtoby, and Kelley, the $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$ packet records the finite compact-metric route by which a dense-open schedule is read as an input surface for the complete-metric Baire theorem. It sits between the Baire category theorem of chapter 1542, the complete-metric Baire surface of chapter 3535, the compact-metric carrier of chapter 129, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed finite bridge from compact-metric data to the complete-metric Baire route; it does not claim countable choice, a host intersection point, open-cover compactness, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8784.1 (Baire category compact metric carrier)**.** *A* $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, F, B, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense-open schedule row, listing the finite prefix of dense-open tests consumed by the Baire read.*
2. *F is the compact-metric finite-net source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *carrier of chapter 129.*
3. *B is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection row, exposing the complete-metric Baire route of chapter 3535 only after D and F are displayed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row controlling the finite schedule prefix inspected by the consumer.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerances and window bounds attached to S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular rational readback.*
7. *H records componentwise* hsame *transport for the schedule, compact-metric source, Baire inspection, stream window, regular readback, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the dense-open schedule and compact finite-net source through the complete-metric Baire inspection row.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurfac}$ $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BaireCategoryCompactMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, F, B, S, Q, E, H, C, P, N$. It has no coordinate for countable choice, a host intersection witness, open-cover compactness, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8784.2 (BaireCategoryCompactMetric NameCert obligations)**.** *For every accepted* BaireCategoryCompac *carrier*

$$K = (D, F, B, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireCategoryCompactMetric}^{\uparrow}}$ *surface consists exactly of dense-open schedule admission for D, compact-metric finite-net source admission for F,* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection for B,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure for S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for Q,* $\mathsf{Real}^{\uparrow}$ *sealing for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No Baire category compact-metric obligation is stored outside these rows.

Proof. Project the accepted carrier. The dense-open schedule D and compact-metric finite-net source F are the only inputs available to the packet, and the complete-metric Baire inspection row B is read only after both rows are displayed.

The finite-window and rational readback rows S and Q then expose the inspected prefix and its tolerance data before the terminal Real seal E is consumed. This is a finite compact-metric route, not a selected point of the full intersection.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no row for countable choice, host intersection, open-cover compactness, quotient equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8784.3 (BaireCategoryCompactMetric route)**.** *Every consumer that reads a compact-metric Baire handoff from an accepted* $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$ *carrier must factor the read through*

$$D, F, B, S, Q, E, H, C, P, N.$$

The handoff reaches the $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *surface of chapter 1542 only through the* CompleteMetricBaireSurfac *inspection row, after the compact-metric finite-net source, stream window, regular rational readback, and Real seal have all been displayed. It exports no countable-choice schedule, host intersection point, open-cover compactness theorem, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier of Definition 8784.1. It fixes the public compact-metric Baire packet as the rows $D, F, B, S, Q, E, H, C, P, N$ and gives no consumer a separate complete-metric or Real-facing coordinate.

Apply the NameCert obligations of Definition 8784.2. They require the dense-open schedule, compact finite-net source, complete-metric Baire inspection, stream window, rational readback, and Real seal to be displayed in that order before the local name row is consumed.

The structural rows then preserve the same route under hsame transport, Cont replay, and Pkg provenance. Thus the compact-metric Baire handoff is a finite bridge into the complete-metric Baire route, not an assertion of countable choice, host intersection, open-cover compactness, quotient equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireCategoryCompactMetric}^{\uparrow}$ packet is generated from a dense-open schedule row, a compact-metric finite-net source row, a CompleteMetricBaireSurface inspection row, a StreamName window row, a RegSeqRat

readback row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireCategoryCompactMetric}\uparrow}$, seed)

Scope closed: Definition 8784.1, Definition 8784.2, and Definition 8784.3 seed the finite compact-metric Baire handoff through dense-open schedules, finite nets, complete-metric Baire inspection, stream windows, rational readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{BaireCategoryCompactMetric}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, a host intersection point, open-cover compactness, quotient equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dense-open schedule admission, compact-metric finite-net admission, complete-metric Baire inspection, stream-window exposure, regular rational readback, Real sealing, transport stability, replay stability, provenance, and non-escape for BEDC.Derived.BaireCategoryCompactMetricUp.

Chapter 8785

A Concrete Naming Certificate for $\mathsf{IwasawaDecomposition}^{\uparrow}$

Following Iwasawa and standard Lie group theory, the $\mathsf{IwasawaDecomposition}^{\uparrow}$ packet records the KAN factorization surface as finite NameCert data rather than as an ambient global analytic theorem. It sits beside the Cartan decomposition of chapter 7926, the Lie-group carrier of chapter 140, the Lie-algebra carrier of chapter 141, the matrix readback of chapter 49, and the underlying group route of chapter 41. The chapter names only the displayed compact, abelian, nilpotent, multiplication, factorization, transport, replay, provenance, and local naming rows.

Definition 8785.1 (Iwasawa decomposition carrier)**.** *An* $\mathsf{IwasawaDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, K, A, N, M, F, H, C, P, L)$$

with the following displayed rows:

1. *G is the* $\mathsf{LieGroup}^{\uparrow}$ *source row, read through the underlying* $\mathsf{Group}^{\uparrow}$ *multiplication route and, when a matrix presentation is displayed, through the* $\mathsf{Mat}^{\uparrow}$ *readback row.*
2. *K is the compact factor row, recording the subgroup surface consumed from the Cartan-side compact data.*
3. *A is the abelian factor row, recording the displayed split abelian direction read from the Lie-algebra and Cartan rows.*
4. *N is the nilpotent factor row, recording the displayed nilpotent direction read from the same Lie-algebra source.*
5. *M is the multiplication route row, recording the ordered product read $K \times A \times N \to G$ using only the visible* $\mathsf{Group}^{\uparrow}$ *multiplication coordinates.*
6. *F is the factorization witness row, recording that the currently accepted source element of G is reached by the displayed K, A, N entries through M.*
7. *H stores componentwise* hsame *transport for the group source, compact factor, abelian factor, nilpotent factor, multiplication route, factorization witness, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an Iwasawa-decomposition request through G, K, A, N, M, F, H.*
9. *P is the* Pkg *provenance over* $\mathsf{IwasawaDecomposition}^{\uparrow}$, $\mathsf{CartanDecomposition}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, $\mathsf{LieAlgebra}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. L is the local $\mathsf{NameCert}_{\mathsf{IwasawaDecomposition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, K, A, N, M, F, H, C, P, L$. It has no coordinate for global Lie theory, Haar measure, selected analytic coordinates, quotient equality, or ambient smooth manifold completeness.

Theorem 8785.2 (Iwasawa decomposition NameCert obligations)**.** *For an accepted* $\mathsf{IwasawaDecomposition}^{\uparrow}$ *packet*

$$I = (G, K, A, N, M, F, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{IwasawaDecomposition}^{\uparrow}}$ *surface is exhausted by the source group row, compact factor row, abelian factor row, nilpotent factor row, multiplication route, factorization witness, componentwise transport, replay, provenance, and local naming. Every accepted read is confined to the displayed route*

$$G, K, A, N, M, F, H, C, P, L.$$

Proof. Project the carrier of Definition 8785.1. The only group-facing source is G, and the only factor rows available to a consumer are K, A, N.

The multiplication and factorization obligations are carried by M and F. They record the ordered product route and the witness that the displayed source element is read through that route.

The structural rows H, C, P, L preserve the same finite packet under transport, replay, provenance, and local naming. Since no global analytic or quotient coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 8785.3 (Iwasawa decomposition KAN route)**.** *Every KAN read from an accepted* $\mathsf{IwasawaDecomposition}^{\uparrow}$ *packet must pass through the Cartan-side compact data, the Lie-algebra abelian and nilpotent directions, the underlying* $\mathsf{Group}^{\uparrow}$ *multiplication route, and any displayed* $\mathsf{Mat}^{\uparrow}$ *readback before the local* $\mathsf{IwasawaDecomposition}^{\uparrow}$ *name is consumed. The public route is the finite packet*

$$K, A, N \xrightarrow{M} G \longmapsto L$$

together with F, H, C, P.

Proof. Start with the factor rows K, A, N. The compact row is sourced through the Cartan decomposition, while the abelian and nilpotent rows are read from the displayed Lie-algebra source.

Next use the multiplication row M. It is the only route in the carrier from the ordered factor rows into the source group G, and it reads the underlying group operation rather than an external analytic product theorem.

Finally apply F, H, C, P, L. The factorization witness, transport, replay, provenance, and local name rows keep the same KAN route visible, so a consumer cannot skip directly from the source group to a local Iwasawa name. □

Theorem 8785.4 (Iwasawa decomposition nonescape)**.** *No accepted* $\mathsf{IwasawaDecomposition}^{\uparrow}$ *read can escape the finite rows of Definition 8785.1. In particular, the packet does not export global Lie theory, Haar measure, selected analytic coordinates, quotient equality, or ambient smooth manifold completeness from the existence of a displayed KAN factorization route.*

Proof. Inspect the carrier coordinates. They name the source group, three factor rows, multiplication, factorization, transport, replay, provenance, and local NameCert row.

Each excluded object would require an additional coordinate: global Lie theory would need a separate theorem surface, Haar measure a measure row, analytic coordinates a chart-selection row, quotient equality a quotient carrier, and ambient smooth manifold completeness a completion row.

Because none of those coordinates is part of the accepted packet, every consumer remains inside the finite KAN route displayed by $G, K, A, N, M, F, H, C, P, L$. □

Falsifiable prediction: An accepted $\mathsf{IwasawaDecomposition}^{\uparrow}$ consumer is refuted if it reaches a local KAN name without first displaying the LieGroupUp source row, compact factor, abelian factor, nilpotent factor, multiplication route, factorization witness, transport, replay, provenance, and local NameCert rows.

Independence witness: cartandecomposition, liegroup, liealgebra, matrix, group: none is bijective to IwasawaDecompositionUp because only the Iwasawa decomposition packet jointly carries the compact factor, abelian factor, nilpotent factor, ordered multiplication route, factorization witness, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{IwasawaDecomposition}^{\uparrow}}$.

Constructive story: An $\mathsf{IwasawaDecomposition}^{\uparrow}$ packet is generated from a LieGroup source row, compact factor row, abelian factor row, nilpotent factor row, displayed multiplication route, factorization witness, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IwasawaDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8785.1, Definition 8785.2, Definition 8785.3, and Definition 8785.4 seed the finite KAN packet over CartanDecompositionUp, LieGroupUp, LieAlgebraUp, MatrixUp, GroupUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IwasawaDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global Lie theory, Haar measure, selected analytic coordinates, quotient equality, ambient smooth manifold completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, KAN route exactness, factorization witness preservation, transport, replay, provenance, and local naming for BEDC.Derived.IwasawaDecompositionUp.

Chapter 8786

A Concrete Naming Certificate for $\mathsf{LocatedLimitOperator}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedLimitOperator}^{\uparrow}$ packet records the finite operator surface that turns located nested-interval or regular Cauchy evidence into a $\mathsf{Real}^{\uparrow}$-facing limit handoff. It sits after the located nested-interval route of chapter 1713, the retained interval source of chapter 320, the dyadic tolerance core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite handoff route only: it does not assert ambient completeness, quotient equality, trichotomy, countable choice, or a selected host limit.

Definition 8786.1 (Located limit-operator carrier)**.** *A* $\mathsf{LocatedLimitOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, J, D, W, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{NestedInterval}^{\uparrow}$ *row, exposing the retained nested interval evidence.*
2. *J is the* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *handoff row, recording the located containment and diameter route used by the source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint widths, requested precisions, and rational enclosure checks.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by each precision request.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed finite windows and dyadic tolerances.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row that may be consumed only after I, J, D, W, Q have been displayed.*
7. *A is the operator-admission row, recording that the input route I, J, D, W, Q has been assembled before the Real-facing handoff R.*
8. *H records componentwise* hsame *transport for $I, J, D, W, Q, R, A, C, P, N$.*
9. *C records displayed* Cont *replay from a precision request through the nested interval, window, readback, and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedLimitOperator}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLimitOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, J, D, W, Q, R, A, H, C, P, N$. It has no coordinate for host completeness, quotient representatives, exact trichotomy, countable choice, or a selected limit outside the displayed rows.

Theorem 8786.2 (Located limit-operator NameCert obligations)**.** *For every accepted* $\mathsf{LocatedLimitOperator}^{\uparrow}$ *carrier $L = (I, J, D, W, Q, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedLimitOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the nested interval row I, located nested-interval handoff J, dyadic tolerance ledger D, finite window row W, regular rational readback Q, terminal Real seal R, operator-admission row A, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming N.*

Proof. Project the carrier of Definition 8786.1. The rows I, J, D, W, Q expose the input evidence from retained nested intervals to regular rational readback, and R is the only Real-facing row.

The operator-admission row A records that the displayed input route has already been assembled before R is consumed. Transport and replay through H and C preserve the same rows and cannot add a hidden selected point or quotient representative.

Finally, provenance and naming through P, N bind the packet to the sibling surfaces cited above. Since no excluded coordinate is present in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 8786.3 (Located limit-operator L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{LocatedLimitOperator}^{\uparrow}$ *carrier receives its limit handoff only through the finite route*

$$\mathsf{NestedInterval}^{\uparrow},\ \mathsf{LocatedRealNestedIntervals}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff exports a displayed Real-facing seal after finite windows and dyadic tolerances have been read; it exports no completeness principle, quotient equality, trichotomy, countable-choice schedule, or selected host limit.

Proof. Read the route from the input side. The $\mathsf{NestedInterval}^{\uparrow}$ row supplies retained interval evidence, and chapter 1713 supplies the located nested-interval handoff that turns that evidence into finite window requests.

The dyadic, stream, and regular rational rows then provide the inspected tolerance ledger, finite windows, and readback data. These rows are the same Real-facing corridor used by chapter 301 before the terminal $\mathsf{Real}^{\uparrow}$ seal may be consumed.

Thus the operator is a finite handoff surface, not a selector. Any consumer that skips one of these rows leaves the accepted $\mathsf{LocatedLimitOperator}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedLimitOperator}^{\uparrow}$ read exposes NestedInterval, LocatedRealNestedIntervals, DyadicRatCore, StreamName, RegSeqRat, Real, operator-admission, transport, replay, provenance, and local NameCert rows before a Real or Completion consumer receives the handoff.

Independence witness: nestedinterval, locatedrealnestedintervals, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedLimitOperator}^{\uparrow}$ is not bijective to any listed sibling because it binds all six rows into one operator-admission packet while refusing host completeness, quotient equality, trichotomy, choice, and selected limits.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLimitOperator}\uparrow}$.

Constructive story: A $\mathsf{LocatedLimitOperator}^{\uparrow}$ packet is generated from a NestedInterval row, a LocatedRealNestedIntervals handoff row, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a Real seal, operator admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedLimitOperator}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8786.1, Definition 8786.2, and Definition 8786.3 seed the finite operator surface from located nested-interval evidence to a Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedLimitOperator}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient completeness, quotient equality, trichotomy, countable choice, selected host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-interval exposure, located nested-interval handoff, dyadic tolerance coverage, finite window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedLimitOperatorUp.

Chapter 8787

A Concrete Naming Certificate for $\mathsf{BishopRealInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealInterval}^{\uparrow}$ packet records a constructive closed real interval as finite endpoint and regularity data rather than as a classical subset of a completed line. It sits after the dyadic rational arithmetic of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers an endpoint-bounded enclosure route without claiming Heine–Borel compactness, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8787.1 (BishopRealInterval carrier)**.** *A* $\mathsf{BishopRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, T, Q, D, E, H, C, P, N)$$

with the following rows:

1. *L is the lower endpoint row, displayed as a regular rational name and sealed only through the terminal real-name route.*
2. *U is the upper endpoint row, displayed as a regular rational name with the same endpoint discipline.*
3. *T is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect both endpoint names at requested precisions.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint regular sequences.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison row recording the finite dyadic scale at which L is below U.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *enclosure seal available only after the endpoint windows, regular readbacks, and dyadic comparison row have been displayed.*
7. *H records componentwise* hsame *transport for the endpoint, stream, readback, dyadic, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interval-enclosure request through the same endpoint and tolerance rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealInterval}^{\uparrow}}$ *row.*
The carrier has no coordinate for classical interval subsets, quotient real equality, selected endpoint representatives, Heine–Borel compactness, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8787.2 (BishopRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealInterval}^{\uparrow}$ *carrier* $I = (L, U, T, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealInterval}^{\uparrow}}$ *surface consists exactly of lower endpoint admission through* L*, upper endpoint admission through* U*, finite endpoint-window exposure through* T*, regular rational readback through* Q*, dyadic tolerance and order comparison through* D*, real-enclosure sealing through* E*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A consumer therefore reads the interval only through the finite route*

$$L, U, T, Q, D \longmapsto E, H, C, P, N.$$

Proof. Project the carrier of Definition 8787.1. The endpoint rows L and U are not host real numbers; they are displayed regular names whose finite prefixes are exposed by the shared $\mathsf{StreamName}^{\uparrow}$ row T.

Use the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q and the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison row D. These rows supply the regular-sequence and dyadic-order evidence before the interval can be treated as a Real-facing enclosure.

Finally replay the structural rows. The seal E, transport H, continuation replay C, provenance P, and local name N preserve the same finite endpoint packet, so the NameCert surface cannot import quotient equality of reals, selected representatives, Heine–Borel compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8787.3 (BishopRealInterval regular seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-readable enclosure request accepted by a* $\mathsf{BishopRealInterval}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Equivalently, the endpoint classifier reads finite stream windows, regular rational endpoint readback, and dyadic tolerance comparison before the terminal Real seal is public. The route is an L10 completion interface over chapter 301 and chapter 38, not a classical interval subset or an ambient completeness principle.

Proof. Start with the NameCert obligations of Definition 8787.2. They require endpoint admission, stream-window exposure, regular rational readback, and dyadic comparison before any Real-facing interval seal can be consumed.

Read the endpoint rows through the sibling $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ routes. The stream row exposes only finite endpoint prefixes, while the regular rational row supplies the endpoint readback used by completion consumers.

Now apply the dyadic comparison row D. Its finite tolerance ledger is the boundary at which the lower and upper endpoint names become an interval enclosure rather than two unrelated real-name seals.

Finally consume E, H, C, P, N. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after those displayed rows remain visible, so the route supplies a constructive interval enclosure without selected endpoint representatives, quotient real equality, countable choice, Heine–Borel compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealInterval}^{\uparrow}$ packet is generated from lower and upper endpoint rows, finite StreamName endpoint windows, RegSeqRat endpoint readback, DyadicRatCore tolerance comparison, a Real enclosure seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8787.1, Definition 8787.2, and Definition 8787.3 seed the endpoint-bounded interval route over StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical interval subsets, quotient real equality, selected endpoint representatives, Heine–Borel compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint carrier admission, finite endpoint-window routing, regular rational readback, dyadic endpoint comparison, Real enclosure sealing, and structural nonescape for BEDC.Derived.BishopRealIntervalUp.

Chapter 8788

A Concrete Naming Certificate for TubeLemmaFiniteProduct$^{\uparrow}$

Following Munkres and Kelley on product topology, compact fibres, and the tube lemma, the TubeLemmaFiniteProduct$^{\uparrow}$ packet records the finite product form of the tube-neighbourhood argument. It refines the one-slice tube packet of chapter 3861 through the finite-product topology certificate of chapter 2000, the compact-row surface of chapter 60, the generated-open topology rows of chapter 88, and the finite-cover refinement route of chapter 3430. The chapter names only the finite product tube carrier and its local obligations; it does not assert arbitrary product compactness, countable products, quotient equality of product points, selected neighbourhood bases, or an ambient open-cover theorem outside the displayed finite rows.

Definition 8788.1 (Finite product tube lemma carrier)**.** *A* TubeLemmaFiniteProduct$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T_{\mathrm{fin}} = (I, X, K, O, U, V, F, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the product coordinates used by the argument.*
2. *X is the finite family of* Topology$^{\uparrow}$ *rows indexed by I.*
3. *K is the finite family of* Compact$^{\uparrow}$ *slice rows for the compact coordinates.*
4. *O is the* ProductTopology$^{\uparrow}$ *open-neighbourhood row containing the displayed compact product slice.*
5. *U is the finite family of coordinate product-neighbourhood rows read from O.*
6. *V is the resulting finite tube-neighbourhood row in the remaining product coordinates.*
7. *F is the finite-cover-refinement row that extracts one displayed tube from the coordinate neighbourhood family.*
8. *R records the route back to the ordinary* TubeLemma$^{\uparrow}$ *slice packet for each exposed coordinate fibre.*
9. *H records componentwise* hsame *transport for the finite index, topology rows, compact rows, product-open row, coordinate neighbourhood family, tube row, finite-cover row, slice route, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a finite product-neighbourhood request through $I, X, K, O, U, V, F, R, H, P, N$*

11. *P is the* Pkg *provenance over* TubeLemmaFiniteProduct$^{\uparrow}$, TubeLemma$^{\uparrow}$, ProductTopology$^{\uparrow}$, Topology$^{\uparrow}$, Compact$^{\uparrow}$, FiniteCoverRefinement$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* NameCert$_{\text{TubeLemmaFiniteProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, X, K, O, U, V, F, R, H, C, P, N$. It has no coordinate for countable products, arbitrary product compactness, quotient equality of product points, selected bases, or a detached ambient open-cover selector.

Lean4 (definition) `BEDC.Derived.TubeLemmaFiniteProductUp`

Theorem 8788.2 (Finite product tube lemma NameCert obligations)**.** *For every accepted* TubeLemmaFiniteProduc *carrier $T_{\text{fin}} = (I, X, K, O, U, V, F, R, H, C, P, N)$, the local* NameCert$_{\text{TubeLemmaFiniteProduct}^{\uparrow}}$ *surface is exhausted by finite-index admission, topology-row exposure for the indexed factors, compact-slice exposure through K, product-open exposure through O, coordinate product-neighbourhood exposure through U, tube-neighbourhood exposure through V, finite-cover refinement through F, coordinatewise routing through R to* TubeLemma$^{\uparrow}$*, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8788.1. The finite index row I bounds the product coordinates, while X and K expose only the displayed topology rows and compact slice rows attached to those coordinates.

The product-neighbourhood data then pass through O, U, V, F. The row O supplies the finite product-open neighbourhood, U records the coordinate neighbourhood family read from it, F records the finite-cover refinement, and V is the resulting tube-neighbourhood row.

Finally the route R reuses the ordinary tube packet coordinatewise, and H, C, P, N preserve the same finite data under transport, replay, provenance, and naming. Since no carrier coordinate names countable products, arbitrary product compactness, quotient product equality, selected bases, or a detached open-cover selector, the listed rows exhaust the local NameCert obligations. □

Theorem 8788.3 (Finite product compact route)**.** *Every accepted* TubeLemmaFiniteProduct$^{\uparrow}$ *carrier routes a compact finite product slice through the displayed chain*

$$I, X, K \longmapsto O \longmapsto U \longmapsto F \longmapsto V \longmapsto R.$$

Thus a downstream compact-product or uniform-continuity consumer must first inspect the finite index row, the factor topology rows, the compact slice rows, the product-open row, the coordinate neighbourhood family, the finite-cover refinement row, and the coordinatewise TubeLemma$^{\uparrow}$ *route before it can read the finite tube neighbourhood.*

Proof. The carrier projection supplies the finite source rows I, X, K. These rows determine which coordinates participate in the compact product slice and prevent any appeal to countable or arbitrary products.

The neighbourhood construction is forced through O, U, F, V. Product openness is exposed first, the coordinate neighbourhood family is then read from that product row, and the finite-cover refinement row chooses the finite tube data used by V.

The final row R records that each coordinate fibre is routed through the ordinary TubeLemma$^{\uparrow}$ packet. Transport, replay, provenance, and naming do not introduce further compactness principles, so the displayed chain is the complete compact route available to downstream consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{TubeLemmaFiniteProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{TubeLemmaFiniteProduct}^{\uparrow}$ packet is generated from a finite index row, indexed Topology rows, Compact slice rows, a ProductTopology open-neighbourhood row, coordinate neighbourhood rows, a tube neighbourhood row, a FiniteCoverRefinement row, a coordinatewise TubeLemma route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TubeLemmaFiniteProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8788.1, Definition 8788.2, and Definition 8788.3 seed the finite product tube-neighbourhood packet over product topology, compact slice, topology, tube lemma, and finite-cover refinement rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TubeLemmaFiniteProduct}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.TubeLemmaFiniteProductUp`

Bridge status: none

Not claimed: This chapter does not close arbitrary product compactness, countable products, quotient equality of product points, selected neighbourhood bases, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-index exposure, topology-row exposure, compact-slice exposure, product-open exposure, coordinate neighbourhood exactness, finite-cover refinement, coordinatewise TubeLemma routing, transport replay, provenance exhaustion, and local naming for BEDC.Derived.TubeLemmaFiniteProductUp.

Chapter 8789

A Concrete Naming Certificate for $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$ packet records the finite fan-modulus surface used when a compact interval argument is read through constructive interval data. It sits after the compact fan bridge of chapter 6946, the modulus selector of chapter 8199, the dyadic interval-net surface of chapter 5966, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a finite modulus row for compact interval consumers without asserting open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8789.1 (BishopCompactIntervalFanModulus carrier)**.** *A* $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CompactIntervalFanBridge}^{\uparrow}$ *row, carrying the finite bridge from interval subdivision data to fan-indexed compact interval reads.*
2. *S is the* $\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}$ *row, recording the selected finite modulus depth for the displayed interval and precision request.*
3. *D is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row, exposing the dyadic interval-net cells inspected at the chosen depth.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the interval endpoints, sampled cells, and modulus witnesses.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning those windows into regular rational observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the finite fan, dyadic-net, stream, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transport for the bridge, selector, dyadic-net, stream, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for compact interval modulus requests through $B, S, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$, $\mathsf{CompactIntervalFanBridge}^{\uparrow}$, $\mathsf{CompactIntervalFanModulusSelector}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite fan-modulus history for a displayed compact interval request. It has no coordinate for open-cover compactness, a selected arbitrary subcover, quotient real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8789.2 (BishopCompactIntervalFanModulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$ *carrier* $M = (B, S, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}}$ *surface consists exactly of compact fan bridge admission, finite modulus-depth selection, dyadic interval-net exposure, stream-window exposure, regular rational readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No compact interval modulus obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8789.1. The fan bridge row B, selector row S, and dyadic interval-net row D are the only compact interval modulus data in the packet.

Read the analytic-facing rows next. The stream row W exposes finite endpoint and cell observations, the regular rational row R supplies the readback used by completion-facing consumers, and the terminal row E seals only those displayed observations as Real-facing data.

Finally use H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same finite packet, so the excluded open-cover, arbitrary subcover, quotient-equality, choice, trichotomy, and ambient-completeness principles cannot enter the local NameCert surface. □

Theorem 8789.3 (BishopCompactIntervalFanModulus finite-cover route)**.** *Every downstream consumer of an accepted* $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{CompactIntervalFanBridge}^{\uparrow},\ \mathsf{CompactIntervalFanModulusSelector}^{\uparrow},\ \mathsf{DyadicIntervalNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRa}$$

Consequently the consumer may read a displayed finite fan depth and its dyadic interval-net witnesses, but it receives no theorem of open-cover compactness, no selected arbitrary subcover, no quotient real equality, no countable-choice schedule, no order trichotomy, and no ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Start with the bridge and selector rows. The row B supplies the compact interval fan bridge, and S records the finite modulus depth attached to the displayed request.

Use the dyadic interval-net row D. It exposes the concrete interval cells available at that depth, so a consumer sees finite coverage data rather than an arbitrary family of opens or a selected subcover.

Route every Real-facing read through W, R, E and preserve the packet through H, C, P, N. Since the carrier contains only those finite rows, the public route is exactly the displayed finite means route and not an ambient compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}$ packet is generated from compact fan bridge rows, a finite modulus selector, dyadic interval-net cells, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}}$, seed)

Scope closed: Definition 8789.1, Definition 8789.2, and Definition 8789.3 seed the finite fan-modulus route over compact fan bridge, modulus selector, dyadic interval net, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopCompactIntervalFanModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected arbitrary subcovers, quotient real equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan bridge admission, finite modulus-depth selection, dyadic interval-net exposure, Real seal nonescape, and structural nonescape for BEDC.Derived.BishopCompactIntervalFanModulusUp.

Chapter 8790

A Concrete Naming Certificate for $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ packet records the finite comparison square between the dense-source induction route of chapter 1776 and the completion-interface route of chapter 1844. It shares the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names a square witness for the L10 completion route; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host function equality, Lean verification, or bridge checking.

Definition 8790.1 (Bishop completion comparison square carrier)**.** *A* $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *Q_0 and Q_1 are the two dense-source faces, exposing the source data read by the induction-extension route and by the completion-interface route.*
2. *I_0 and I_1 are the two completion-interface faces, naming the* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *extension read and the* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *boundary read over the same source faces.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger used by both faces before any Real seal is visible.*
4. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the dyadic tolerance ledger.*
5. *R is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to D and S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the common route D, S, R has been displayed.*
7. *L is the square-commutation ledger, recording that both paths from Q_0, Q_1 through I_0, I_1 reach the same shared rows D, S, R, E before any structural export.*
8. *H records componentwise* hsame *transport for the dense-source faces, completion-interface faces, shared route rows, commutation ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the square from each dense-source face through the shared route rows and the commutation ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$, $\mathsf{BishopCompletionInduction}^{\uparrow}$, $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonSquareUp.BishopCompletionComparisonSquareTasteGate`
`single carrier alignment`

Theorem 8790.2 (Bishop completion comparison square NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ *carrier*

$$X = (Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two dense-source faces, two completion-interface faces, shared dyadic ledger, finite windows, regular readback, Real seal, commutation ledger, transport, replay, provenance, and local naming rows;*
2. *dense-source visibility, requiring Q_0 and Q_1 to be displayed before either completion-facing face is read;*
3. *interface-face exactness, requiring I_0 to read the* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *extension face and I_1 to read the* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *boundary face over the same displayed source data;*
4. *shared-route discipline, requiring both faces to consume the same D, S, R, E route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
5. *square-commutation, requiring L to compare only the two displayed paths and their shared route rows;*
6. *structural non-escape, requiring H, C, P, N to transport, replay, source, and name no row outside the displayed packet.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8790.1. The accepted packet displays exactly the two dense-source faces, two completion-interface faces, the shared route rows, and the structural rows.

Dense-source visibility is forced by the row order. Both completion-facing faces can be read only after Q_0 and Q_1 are visible, so neither path can introduce a hidden source representative.

The interface-face exactness clauses are the rows I_0 and I_1. The first reads the Bishop completion-induction extension face, and the second reads the Cauchy real completion-interface boundary face, both over the same displayed source data.

The shared-route and commutation clauses are the rows D, S, R, E, L. The dyadic ledger opens the finite windows, the regular readback consumes those windows, the Real seal is terminal, and the ledger L compares only the two paths through those rows.

Finally H, C, P, N add only transport, replay, provenance, and local naming. Since the carrier contains none of the excluded coordinates, the displayed rows exhaust the local NameCert surface. □

Theorem 8790.3 (Bishop completion comparison square commutes)**.** *Let $X = (Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P$ be an accepted* $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ *carrier. Every completion-facing comparison exported by X factors through the finite square*

$$Q_0, Q_1 \longmapsto I_0 \longmapsto D, S, R, E \longmapsto L, H, C, P, N,$$
$$Q_0, Q_1 \longmapsto I_1 \longmapsto D, S, R, E \longmapsto L, H, C, P, N.$$

Thus the two displayed paths commute at the local NameCert surface: each reaches the same dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, terminal Real seal, commutation ledger, transport, replay, provenance, and local name.

Proof. Project the accepted carrier. Both paths begin with the same dense-source faces Q_0, Q_1 and can inspect no completion-facing datum before those faces are displayed.

The upper path opens I_0, the Bishop completion-induction extension face. The lower path opens I_1, the Cauchy real completion-interface boundary face. By carrier admission, both faces are attached to the same source data and to the same shared route rows.

The shared route then forces the comparison. Both paths pass through D, S, R, E before the commutation ledger can be read, so they reach the same dyadic tolerance, StreamName window, regular readback, and terminal Real-seal coordinates.

The ledger L records exactly this comparison, while H, C, P, N preserve, replay, source, and name it. No additional coordinate is present from which a second completion route could diverge. □

Theorem 8790.4 (Bishop completion comparison square Real handoff)**.** *For an accepted* BishopCompletionCompari *carrier $X = (Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P, N)$, any Real-facing consumer must receive the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only through the shared finite route*

$$D \longmapsto S \longmapsto R \longmapsto E.$$

The handoff is therefore a common dyadic-tolerance, finite-window, regular-readback, Real-seal route for both the induction-extension face and the completion-interface face, not an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness or to a selected host limit.*

Proof. Start with a Real-facing consumer of the square. Projection of the carrier gives the shared route rows D, S, R, E and no other Real-facing coordinate.

The dyadic ledger D is opened first. It supplies the tolerance budget that controls the finite StreamName windows S and the regular rational readback R.

Only after D, S, R have been displayed can the terminal seal E be read. Both completion-facing faces I_0 and I_1 use the same route, so the consumer receives one Real handoff rather than two independent seals.

Since the excluded coordinates are absent from X, the consumer cannot replace this handoff by ambient completeness, quotient equality, selected limits, countable choice, host function equality, Lean verification, or bridge checking. □

Theorem 8790.5 (Bishop completion comparison square nonescape)**.** *No consumer of an accepted* $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ *carrier may export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, countable choice, host function equality, Lean verification, or bridge checking. The consumer may export only the displayed comparison square on* $Q_0, Q_1, I_0, I_1, D, S, R, E, L, H, C, P, N$ *and the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}$ *evidence attached to those rows.*

Proof. Inspect the carrier inventory in Definition 8790.1. The tuple contains only dense-source faces, completion-interface faces, shared route rows, a commutation ledger, and structural naming rows.

Definition 8790.2 exhausts the local obligation surface. Each allowed export is tied to one of the displayed rows, and no excluded row is introduced by the obligation theorem.

Definition 8790.3 and Definition 8790.4 then fix the consumer route. Every comparison and every Real-facing handoff passes through the shared finite square before it can be named.

Thus any exported consumer result remains inside the displayed packet. The listed ambient completeness, quotient, choice, host-equality, Lean, and bridge objects are absent from the carrier and cannot escape through the local NameCert row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8790.1, Definition 8790.2, Definition 8790.3, Definition 8790.4, and Definition 8790.5 seed the finite L10 comparison square between the Bishop completion-induction extension route and the Cauchy real completion-interface boundary route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionComparisonSquare}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source visibility, interface-face exactness, shared-route discipline, square commutation, Real handoff, and consumer nonescape for BEDC.Derived.BishopCompletionComparisonSquareUp

Constructive story: A $\mathsf{BishopCompletionComparisonSquare}^{\uparrow}$ packet is generated from two dense-source faces, two completion-interface faces, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, a square-commutation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8791

A Concrete Naming Certificate for $\mathsf{PrimeShadowInterface}^{\uparrow}$

The $\mathsf{PrimeShadowInterface}^{\uparrow}$ packet realizes the bulk–boundary prime-shadow vision of chapter 9332 as finite NameCert data. It sits after the arithmetic prime certificate of chapter 64, the finite prime-window gluing route of chapter 8765, the zeta source of chapter 71, the zeta-zero surface of chapter 73, the zeta zero type factorization of chapter 8772, and the critical-line witness route of chapter 548. The packet names only a finite interface row: it does not prove RH, does not discharge the analytic incompatibility row, and does not assert zero-free strips, Euler product convergence on the critical strip, quotient complex equality, or ambient analytic completeness.

Definition 8791.1 (PrimeShadowInterface carrier)**.** *A* $\mathsf{PrimeShadowInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, W, Z, A, D, H, C, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Prime}^{\uparrow}$ *row naming the prime classifier and the local prime-invariant read used by the boundary shadow.*
2. *W is the* $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ *row recording the finite prime window through which boundary-visible prime data is inspected.*
3. *Z is the* $\mathsf{Zeta}^{\uparrow}$ *and* $\mathsf{ZetaZeros}^{\uparrow}$ *source row, retaining only the displayed zeta and zero-locus surfaces.*
4. *A is the* $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$ *row that connects a legal zero-facing type read to the finite zeta-zero surface.*
5. *D is the prime-shadow defect row: it records the layer index, finite readout, nonzero defect witness, and refusal of a hidden bulk invariant.*
6. *H records componentwise* hsame *transport for prime, window, zeta, zero-factor, defect, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for the finite prime-shadow boundary read through P, W, Z, A, D, H, G, N.*
8. *G is the* Pkg *provenance over* $\mathsf{PrimeShadowInterface}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$, $\mathsf{Zeta}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$, $\mathsf{CriticalLineWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimeShadowInterface}^{\uparrow}}$ *row.*

The carrier has no coordinate for an RH proof, a global zero classifier, a completed Euler product, analytic continuation outside the displayed sibling rows, quotient equality of complex data, or an ambient completeness principle.

Theorem 8791.2 (PrimeShadowInterface NameCert obligations)**.** *For every accepted* $\mathsf{PrimeShadowInterface}^{\uparrow}$ *carrier* $S = (P, W, Z, A, D, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PrimeShadowInterface}^{\uparrow}}$ *surface consists exactly of prime-row admission, finite prime-window admission, zeta-source admission, zeta-zero factorization admission, prime-shadow defect exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The obligation surface may cite chapter 9332, but it exports only these finite boundary rows.*

Proof. Project the carrier of Definition 8791.1. The rows P, W, Z, A, D are the complete mathematical surface of the packet: prime classifier data, a finite prime window, zeta and zero-facing rows, zero-type factorization, and a displayed defect read.

Use the structural rows H, C, G, N. They transport, replay, package, and name the same finite boundary interface, so the local certificate cannot introduce an analytic theorem beyond the displayed siblings.

The RH vision is therefore consumed only as a target diagram. The packet supplies a concrete boundary interface for that diagram, not an incompatibility proof or an RH conclusion. □

Theorem 8791.3 (PrimeShadowInterface finite boundary route)**.** *Every RH-facing consumer of an accepted* $\mathsf{PrimeShadowInterface}^{\uparrow}$ *packet must read the finite prime-shadow boundary through*

$$\mathsf{Prime}^{\uparrow},\ \mathsf{FinitePrimeWindowGluing}^{\uparrow},\ \mathsf{Zeta}^{\uparrow},\ \mathsf{ZetaZeros}^{\uparrow},\ \mathsf{ZetaZeroTypeFactorization}^{\uparrow} \longmapsto \mathsf{CriticalLineWitness}^{\uparrow},\ \mathsf{BHist}$$

The route keeps bulk–boundary prime-shadow detection finite and local. It does not claim RH, analytic defect-zero incompatibility, a zero-free strip theorem, completed Euler products on the critical strip, quotient complex equality, or host analytic completeness.

Proof. Start from P and W. The prime-facing side is a displayed prime classifier read plus a finite prime window, so the consumer receives only finite boundary-visible prime data.

Read Z and A next. These rows expose the zeta source and zeta-zero type factorization before any critical-line consumer can treat a zero-facing row as boundary data.

Finally preserve the route through D, H, C, G, N. The defect row is local to the same packet, while transport, replay, provenance, and naming keep all reads inside the finite interface. The refused analytic claims are absent from the carrier and therefore remain outside the exported boundary route. □

Falsifiable prediction: If the finite prime-window row W or the defect row D is removed, an RH-facing consumer can no longer distinguish a visible prime-shadow defect from an untracked zeta-zero read.

Independence witness: Nearest siblings: prime, finite-prime-window-gluing, zeta-basic, zeta-zeros, criticallinewitness. None carries the combined boundary interface because each sibling supplies only the prime classifier, finite window, zeta source, zero surface, or critical-line witness, while $\mathsf{PrimeShadowInterface}^{\uparrow}$ binds them to one defect-read packet.

The Lean follow-up for namespace BEDC.Derived.PrimeShadowInterfaceUp.TasteGate must provide explicit Nontrivial $\mathsf{PrimeShadowInterface}^{\uparrow}$ and FieldFaithful $\mathsf{PrimeShadowInterface}^{\uparrow}$ instances before a checked carrier-alignment theorem can serve as the TasteGate witness. Those instances must preserve $P, W, Z, A, D, H, C, G, N$ as distinct fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeShadowInterface}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeShadowInterface}^{\uparrow}$ packet is generated from a Prime row, finite prime-window gluing, zeta source rows, zeta-zero factorization rows, a prime-shadow defect read, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeShadowInterface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8791.1, Definition 8791.2, and Definition 8791.3 seed the finite prime-shadow boundary interface over Prime, finite prime-window, Zeta, ZetaZeros, zero-factorization, critical-line, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeShadowInterface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, analytic defect-zero incompatibility, zero-free strips, completed Euler products on the critical strip, quotient complex equality, ambient analytic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prime-window exactness, zeta-zero factorization readback, local defect exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.PrimeShadowInterfaceUp.

Chapter 8792

A Concrete Naming Certificate for $\mathsf{DnaWindowTriggerFrontier}^{\uparrow}$

The $\mathsf{DnaWindowTriggerFrontier}^{\uparrow}$ packet realizes the boundary-tile hinge and ablation rows of Definition 9462.1468 as finite BEDC NameCert data. It consumes the boundary-tile source of chapter 3393, the six-boundary coverage rows of chapter 3470, the trigger algebra of chapter 3478, the closure/refusal discipline of chapter 3508, and the reliability surfaces of chapter 3583 and chapter 3657. The packet names a finite trigger-frontier row for DNA-window ablation reads; it is not a biological mechanism, stochastic evolution model, quotient codon semantics, hidden probability oracle, or infinite trigger search.

Definition 8792.1 (DNA-window trigger frontier carrier)**.** *A* $\mathsf{DnaWindowTriggerFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, B, C, A, L, R, T, U, V, H, Q, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *boundary-window source row.*
2. *B is the boundary-tile hinge row, naming the displayed codon-window faces inspected by ablation.*
3. *C is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *coverage row, recording support coverage for the same boundary faces.*
4. *A is the* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *row, carrying finite trigger-support operations.*
5. *L is the* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *closure/refusal row, retaining blocked infinite-branch and quotient-codon reads.*
6. *R is the reliability row, read through* $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ *and* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$.
7. *T is the finite trigger-frontier row, listing the trigger, support, and ablation boundary entries visible to consumers.*
8. *U is the transport row for boundary-tile, support, reliability, and refusal data across the displayed frontier.*
9. *V is the replay row for the source vision's hinge and ablation calculations.*
10. *H records componentwise* hsame *transport for $W, B, C, A, L, R, T, U, V, Q, P, N$.*
11. *Q records displayed* Cont *replay through the same finite rows.*

12. *P is the* Pkg *provenance over* DnaWindowTriggerFrontier$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverag WindowCoverageTriggerAlgebra$^{\uparrow}$, DnaBoundaryTriggerClosure$^{\uparrow}$, EnergyBiasedTriggerReliability$^{\uparrow}$, TriggerHypergrap *the source vision of Definition 9462.1468,* BHist, hsame, Cont, *and* NameCert.

13. *N is the local* NameCert$_{\text{DnaWindowTriggerFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, C, A, L, R, T, U, V, H, Q, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden probability, infinite trigger enumeration, or host spectral oracle.

Theorem 8792.2 (DNA-window trigger frontier obligations)**.** *For every accepted* DnaWindowTriggerFrontier$^{\uparrow}$ *carrier $F = (W, B, C, A, L, R, T, U, V, H, Q, P, N)$, the local* NameCert$_{\text{DnaWindowTriggerFrontier}^{\uparrow}}$ *surface consists exactly of boundary-window admission, hinge-row exposure, coverage-row exposure, trigger-algebra exposure, closure/refusal exposure, reliability-row exposure, finite trigger-frontier exposure, transport, replay, provenance, and local naming. A consumer may read a boundary-tile hinge or ablation frontier only after all these rows are displayed.*

Proof. Project the carrier of Definition 8792.1. The source and support rows are W, B, C, A, L, the reliability and frontier rows are R, T, and the structural rows are U, V, H, Q, P, N.

The boundary rows are supplied by chapter 3393 and chapter 3470. They expose the finite window faces and support coverage before any trigger-frontier read is public.

The trigger rows are supplied by chapter 3478 and chapter 3508. These keep support closure and refusal rows visible, so an accepted frontier cannot become an infinite trigger search or quotient codon semantics.

The reliability row R is read only through chapter 3583 and chapter 3657. Finally T, U, V, H, Q, P, N expose, transport, replay, source, and name the same finite frontier. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the obligation surface. □

Falsifiable prediction: Every accepted DnaWindowTriggerFrontier$^{\uparrow}$ consumer must exhibit DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WindowCoverageTriggerAlgebra, DnaBoundaryTriggerClosure, EnergyBiasedTriggerReliability, TriggerHypergraphReliability, trigger-frontier, support, refusal, transport, replay, provenance, and local NameCert rows before reading a hinge or ablation frontier.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra, dna-boundary-trigger-closure, energy-biased-trigger-reliability: DnaWindowTriggerFrontier$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boundary windows, hinge rows, coverage rows, trigger algebra, closure/refusal rows, reliability rows, finite trigger-frontier entries, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{DnaWindowTriggerFrontier}^{\uparrow}}$.

Constructive story: A DnaWindowTriggerFrontier$^{\uparrow}$ packet is generated from boundary-window rows, boundary-tile hinge rows, six-boundary coverage rows, finite trigger-algebra rows, closure/refusal rows, reliability rows, finite trigger-frontier entries, transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DnaWindowTriggerFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 8792.1 and Definition 8792.2 seed the finite DNA-window trigger-frontier interface over boundary-tile, coverage, trigger-algebra, closure/refusal, reliability, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{DnaWindowTriggerFrontier}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon semantics, hidden probability models, infinite trigger search, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked boundary-window admission, hinge-row exposure, coverage exposure, trigger-algebra exposure, refusal exposure, reliability exposure, frontier exposure, transport, replay, provenance, and local naming for BEDC.Derived.DnaWindowTriggerFrontierUp.

Chapter 8793

A Concrete Naming Certificate for $\mathsf{HenstockGaugeIntegral}^{\uparrow}$

Following Henstock and Kurzweil, the $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ packet names the finite gauge-integral interface obtained from Cousin-style tagged partitions and regulated-function control. It consumes the Cousin partition route of chapter 7385, the Henstock gauge-partition packet of chapter 8286, the regulated-function surface of chapter 1906, and the positive-radius discipline supplied by chapter 2301. The packet records tagged sums, gauge fineness, oscillation control, tolerance readback, Real sealing, transport, replay, provenance, and local naming; it is not open-cover compactness, selected subcovers, measure completion, host integration, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8793.1 (Henstock gauge integral carrier)**.** *A* $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, G, P, S, O, T, E, H, C, Q, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the regulated-window data visible to the integral consumer.*
2. *G is the positive gauge row, read through the same finite radius discipline used by* $\mathsf{LebesgueNumberLemma}^{\uparrow}$*.*
3. *P is the* $\mathsf{HenstockGaugePartition}^{\uparrow}$ *partition row, with tags, endpoint readback, coverage, disjointness, and gauge-fineness already displayed.*
4. *S is the tagged-sum row over the finite partition P and the regulated source F.*
5. *O is the oscillation-control row, recording the finite local error bound available on the cells of P.*
6. *T is the tolerance readback row, binding the requested precision to the displayed gauge, partition, sum, and oscillation rows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the accepted integral read.*
8. *H records componentwise* hsame *transport for the regulated source, gauge, partition, tagged sum, oscillation control, tolerance, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through $F, G, P, S, O, T, E, H, Q, N$.*

10. *Q is the* Pkg *provenance over* $\mathsf{HenstockGaugeIntegral}^{\uparrow}$, $\mathsf{CousinPartition}^{\uparrow}$, $\mathsf{HenstockGaugePartition}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HenstockGaugeIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, P, S, O, T, E, H, C, Q, N$. It has no coordinate for open-cover compactness, a selected subcover, a host integration primitive, measure completion, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8793.2 (Henstock gauge integral NameCert obligations)**.** *For every accepted* $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ *carrier $I = (F, G, P, S, O, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HenstockGaugeIntegral}^{\uparrow}}$ *surface consists exactly of regulated-source admission, positive-gauge admission, Henstock gauge-partition admission, tagged-sum exposure, oscillation-control exposure, tolerance readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every integral consumer must read the* $\mathsf{CousinPartition}^{\uparrow}$, $\mathsf{HenstockGaugePartition}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, *and* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *rows before reading the terminal integral value.*

Proof. Project the carrier of Definition 8793.1. The analytic rows are F, G, P, S, O, T, E, and the structural rows are H, C, Q, N.

The partition row P is already finite because chapter 7385 and chapter 8286 display tags, endpoint readback, fineness, coverage, and disjointness before any sum is consumed. The gauge row G is read through the positive-radius discipline of chapter 2301.

The regulated source row F supplies only the finite regulated windows named by chapter 1906. The tagged-sum row S, oscillation row O, tolerance row T, and Real seal E therefore remain downstream of displayed finite data.

Finally H, C, Q, N transport, replay, source, and name the same packet. Since no carrier coordinate stores a selected subcover, host integral, measure completion, quotient function equality, countable-choice schedule, or ambient Real completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 8793.3 (Henstock gauge integral regulated handoff)**.** *Every* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ *packet factors through*

$$F, G, P, S, O, T, E \longmapsto H, C, Q, N.$$

The handoff first displays the regulated source row, then the positive gauge and finite Henstock gauge partition, then the tagged sum and oscillation-control ledger, and only then the tolerance readback and terminal $\mathsf{Real}^{\uparrow}$ *seal. It exports no open-cover compactness, selected subcover, host integration primitive, measure completion, quotient function equality, countable choice, or ambient Real completeness.*

Proof. Start from the regulated source row F. By chapter 1906, the source is visible only through finite step-window and tolerance data, so the integral packet cannot read an ambient host function.

Next read G and P. The positive gauge is displayed before the partition row is consumed, and the Henstock gauge-partition route supplies a finite tagged partition with endpoint readback, gauge fineness, coverage, and disjointness.

Now consume S, O, T, E. The tagged-sum row is computed on the displayed partition, the oscillation-control row records the local error budget on those cells, the tolerance row binds the requested precision to that finite ledger, and the terminal Real seal exposes the accepted integral read.

The remaining rows H, C, Q, N only transport, replay, source, and name this finite route. Thus the regulated handoff remains a constructive gauge-integral packet rather than a hidden compactness, choice, measure, quotient, host-integration, or Real-completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ consumer must expose RegulatedFunction, positive gauge, CousinPartition, HenstockGaugePartition, tagged-sum, oscillation-control, tolerance-readback, Real-seal, transport, replay, provenance, and local NameCert rows before reading the integral value.

Independence witness: cousin-partition, henstock-gauge-partition, regulated-function, lebesgue-number-lemma: $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regulated source, positive gauge, finite Henstock gauge partition, tagged sum, oscillation control, tolerance readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockGaugeIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{HenstockGaugeIntegral}^{\uparrow}$ packet is generated from a regulated-function source row, positive gauge row, finite Henstock gauge partition, tagged-sum row, oscillation-control ledger, tolerance readback, terminal Real seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockGaugeIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8793.1, Definition 8793.2, and Definition 8793.3 seed the finite Henstock gauge-integral interface over CousinPartition, HenstockGaugePartition, RegulatedFunction, LebesgueNumberLemma, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HenstockGaugeIntegral}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, measure completion, host integration, quotient function equality, countable choice, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked regulated-source admission, positive-gauge admission, finite partition admission, tagged-sum exposure, oscillation control, tolerance readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HenstockGaugeIntegralUp.

Chapter 8794

A Concrete Naming Certificate for $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$

Following Kantorovich and the finite-dimensional inverse-function theorem, the $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ packet records the finite radius budget that lets a Newton consumer open an inverse window without invoking a global inverse theorem. It sits beside the Newton-Kantorovich route of chapter 1785, the inverse-function carrier of chapter 2055, the complete-metric handoff of chapter 128, the terminal Real surface of chapter 38, the metric source of chapter 57, and the continuous-map route of chapter 130. The chapter names a local ledger for contraction, inverse-window, and Real-facing handoff rows; it does not assert selected limits, host differentiability, quotient representatives, global inverse functions, or Lean verification.

Definition 8794.1 (Newton inverse-radius ledger carrier)**.** *A* $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, R, F, D, I, A, W, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, naming the domain and codomain distance windows used by the local inverse read.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *radius row, carrying the positive radius, tolerance budget, and endpoint comparisons used by the inverse window.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *source map row, displayed before any derivative or Newton read is consumed.*
4. *D is the local derivative or finite-difference control row read by the inverse-window calculation.*
5. *I is the inverse-witness row for the derivative control displayed by D at the centre of the window.*
6. *A is the approximation ledger, storing the finite residual and Lipschitz or majorant comparisons that bound the Newton correction inside the radius row R.*
7. *W is the inverse-window row, recording the accepted ball, target ball, and nonescape side condition.*
8. *Q is the contraction-obligation row, recording that the Newton transform determined by F, D, I, A, W is contractive on the displayed radius window.*

9. *K is the* NewtonKantorovich$^{\uparrow}$ *handoff row, opened only after M, R, F, D, I, A, W, Q have been displayed.*
10. *E is the* CompleteMetric$^{\uparrow}$ *handoff row that accepts the contraction ledger as a local completion route for the window.*
11. *H records componentwise* hsame *transport for the metric source, radius budget, map row, derivative control, inverse witness, approximation ledger, inverse window, contraction row, Newton-Kantorovich handoff, complete-metric handoff, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from the map row through the radius ledger, inverse witness, contraction obligation, and local handoff rows.*
13. *P is the* Pkg *provenance over* NewtonInverseRadiusLedger$^{\uparrow}$, NewtonKantorovich$^{\uparrow}$, InverseFunction$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* NameCert$_{\text{NewtonInverseRadiusLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, F, D, I, A, W, Q, K, E, H, C, P, N$. It has no coordinate for selected limits, host differentiability, quotient representatives, global inverse functions, Lean verification, or bridge checking.

Theorem 8794.2 (Newton inverse-radius ledger contraction obligation)**.** *For every accepted* NewtonInverseRadiusL *carrier*

$$L = (M, R, F, D, I, A, W, Q, K, E, H, C, P, N),$$

the contraction row Q is readable only after the metric source M, radius budget R, continuous-map row F, derivative-control row D, inverse-witness row I, approximation ledger A, and inverse-window row W are displayed. Thus every contraction claim exported by the packet is a local radius-budget claim about the displayed Newton transform, not a global inverse-function claim.

Proof. Project the carrier of Definition 8794.1. The only rows upstream of Q are M, R, F, D, I, A, W, so the contraction obligation has no separate source from which it can read a global inverse theorem.

The metric and radius rows determine the window. The map row and derivative-control row determine the local Newton transform, while the inverse witness and approximation ledger record the finite bounds used by that transform.

Only after those rows are visible can W expose the nonescape window and Q record contraction inside it. Since the carrier lacks global inverse-function and selected-limit coordinates, the exported contraction claim remains confined to the displayed radius ledger. □

Theorem 8794.3 (Newton inverse-radius ledger inverse-window obligation)**.** *Every accepted* NewtonInverseRadiusL *read of an inverse window factors through*

$$M, R, F, D, I, A \longmapsto W \longmapsto Q \longmapsto K, E, H, C, P, N.$$

The row W names only the local ball, target ball, and nonescape side condition carried by the radius ledger; it does not select a host limit, choose a quotient representative, assert host differentiability, or export a global inverse function.

Proof. The inverse-window row is W. By carrier admission, it is downstream of the metric source, radius row, map row, derivative-control row, inverse witness, and approximation ledger.

The contraction row Q can consume W only as a finite local window. It then hands the data to the Newton-Kantorovich row K and the complete-metric row E for local completion-facing use.

The structural rows H, C, P, N preserve, replay, source, and name the same displayed route. None of them adds selected limits, quotient representatives, host differentiability, or a global inverse function, so the inverse-window read remains local to W. □

Theorem 8794.4 (Newton inverse-radius ledger NameCert ledger)**.** *For every accepted* NewtonInverseRadiusLedger *carrier, the local* $\mathsf{NameCert}_{\mathsf{NewtonInverseRadiusLedger}^{\uparrow}}$ *surface is exhausted by carrier admission, metric-source visibility, radius-budget exactness, continuous-map source visibility, derivative-control compatibility, inverse-witness stability, approximation-ledger exactness, inverse-window nonescape, contraction-row exactness, Newton-Kantorovich handoff, complete-metric handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission gives exactly the tuple $M, R, F, D, I, A, W, Q, K, E, H, C, P, N$. The source rows M, R, F, D, I, A describe the local metric, radius, map, derivative-control, inverse-witness, and approximation data.

The window and contraction obligations are the rows W and Q. They bind the source data to one local inverse window and one contraction claim inside that window.

The handoff rows K and E route the same data through the Newton-Kantorovich and complete-metric surfaces. The structural rows H, C, P, N add only transport, replay, provenance, and local naming, so no additional obligation lies outside the displayed ledger. □

Theorem 8794.5 (Newton inverse-radius ledger local handoff)**.** *Any consumer of an accepted* $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ *packet may export only the local handoff through* $\mathsf{NewtonKantorovich}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *named by K and E. The consumer may cite the finite radius budget, inverse-window row, and contraction row, but it may not export selected limits, host differentiability, quotient representatives, global inverse functions, Lean verification, or bridge checking.*

Proof. The downstream analytic rows are K and E. They are downstream of the radius ledger, inverse-window row, and contraction row, so they cannot be read before the local finite budget is displayed.

The packet contains no selected-limit row, host differentiability row, quotient-representative row, global inverse-function row, Lean marker, or bridge row. A consumer therefore has no coordinate from which to export those claims.

The only public handoff left is the finite local route through $M, R, F, D, I, A, W, Q, K, E, H, C, P, N$, with K and E naming the Newton-Kantorovich and complete-metric consumers of that route. □

Falsifiable prediction: Every accepted $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ packet displays MetricUp source data, a RealUp radius row, ContinuousMapUp source data, derivative control, inverse-witness data, an approximation ledger, an inverse-window row, a contraction row, NewtonKantorovichUp handoff, CompleteMetricUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data before any inverse-window consumer is accepted.

Independence witness: newtonkantorovich, inversefunction, completemetric, real, metric, continuousmap: $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the radius budget, inverse-window nonescape row, contraction obligation, Newton-Kantorovich handoff, complete-metric handoff, transport, replay, provenance, and local naming in one finite ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonInverseRadiusLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NewtonInverseRadiusLedger}^{\uparrow}$ packet is generated from MetricUp source data, a RealUp radius row, ContinuousMapUp source data, derivative control, inverse-witness data, an approximation ledger, an inverse-window

row, a contraction row, NewtonKantorovichUp handoff, CompleteMetricUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{NewtonInverseRadiusLedger}\uparrow}$, seed)

Scope closed: Definition 8794.1, Definition 8794.2, Definition 8794.3, Definition 8794.4, and Definition 8794.5 seed the finite inverse-radius ledger for local Newton inverse-window consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{NewtonInverseRadiusLedger}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, host differentiability, quotient representatives, global inverse functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source visibility, radius-budget exactness, continuous-map source visibility, derivative-control compatibility, inverse-witness stability, approximation-ledger exactness, inverse-window nonescape, contraction-row exactness, Newton-Kantorovich handoff, complete-metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.NewtonInverseRadiusLedgerUp.

Chapter 8795

A Concrete Naming Certificate for MetaCICNormalizationConfluenceFrontier$^{\uparrow}$

The MetaCICNormalizationConfluenceFrontier$^{\uparrow}$ packet realizes the candidate-mediated normalization boundary of chapter 9383 as finite NameCert data. It sits beside the candidate normalization/confluence handoff of chapter 5547, the normalization audit packet of chapter 3779, the typed-normalization audit of chapter 3539, the normalization frontier of chapter 892, and the finite-window confluence route of chapter 6027. The packet names the exact frontier where candidate evidence, residual-join visibility, and confluence-boundary rows are retained before any public normalization consumer may proceed.

Definition 8795.1 (MetaCIC normalization-confluence frontier carrier)**.** *A* MetaCICNormalizationConfluenceFronti *carrier is a finite* BHist *packet*

$$F = (T, K, S, R, J, B, A, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row being inspected.*
2. *K is the candidate-evidence row, carrying reducibility-candidate, closed-candidate, or candidate-set evidence before any strong-normalization projection is read.*
3. *S is the candidate-mediated strong-normalization row for T.*
4. *R is the residual-frontier row, recording the redex or residual window that remains visible when normalization and confluence routes meet.*
5. *J is the residual-join boundary row, recording the finite local join or blocked-join status for the displayed residual window.*
6. *B is the confluence-boundary row, recording exactly which confluence claim is available and which subject-reduction or closed-consistency obligations remain outside the packet.*
7. *A is the normalization-audit row, routing the same T, K, S, R, J, B data through the audit map of chapter 9383.*
8. *O is the obstruction row retained whenever a consumer asks to erase candidate evidence, residual-join boundary rows, subject-reduction obligations, or the confluence boundary.*

9. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, SN row, residual frontier, residual join boundary, confluence boundary, audit row, obstruction row, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a normalization-confluence request through candidate evidence, residual visibility, join boundary, confluence boundary, obstruction, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalizationConfluenceFrontier}^{\uparrow}$, $\mathsf{MetaCICCandidateNormalizationConfluen}$ $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$, $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{FiniteWindowConfluence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationConfluenceFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, S, R, J, B, A, O, H, C, P, N$. It has no coordinate for evidence-free strong normalization, unconditional confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Theorem 8795.2 (MetaCIC normalization-confluence frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICNormalizationConfluenceFrontier}^{\uparrow}$ *carrier*

$$F = (T, K, S, R, J, B, A, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationConfluenceFrontier}^{\uparrow}}$ *surface consists exactly of carrier admission, closed-typed-term visibility, candidate-evidence retention, candidate-mediated strong-normalization projection, residual-frontier visibility, residual-join boundary retention, confluence-boundary retention, normalization-audit routing, obstruction exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission projects the tuple $T, K, S, R, J, B, A, O, H, C, P, N$. The closed typed term row and candidate-evidence row are the source rows for any normalization read.

The candidate-mediated projection is S. It can be read only after K is visible, so the carrier cannot produce evidence-free strong normalization.

The confluence-facing rows are R, J, B. They retain the residual frontier, the residual-join boundary, and the confluence boundary before the audit row A can route the packet through the normalization audit map.

The obstruction and structural rows O, H, C, P, N preserve the same finite route. Since the carrier contains none of the excluded global coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 8795.3 (MetaCIC normalization-confluence candidate gate)**.** *Every normalization read from an accepted* $\mathsf{MetaCICNormalizationConfluenceFrontier}^{\uparrow}$ *packet factors through*

$$T \longmapsto K \longmapsto S$$

before it may inspect residual-frontier, residual-join, confluence-boundary, audit, obstruction, transport, replay, provenance, or local naming rows. A consumer that erases K is outside the carrier.

Proof. The source term is T, and the only normalization evidence row upstream of S is K. Thus any strong-normalization read must first retain the displayed candidate evidence.

After S is visible, the consumer may inspect $R, J, B, A, O, H, C, P, N$. These rows refine the same candidate-mediated route by recording residual and confluence boundaries.

If K is erased, the route no longer matches the carrier inventory. The consumer has then left the finite NameCert surface and cannot be accepted by the local classifier. □

Theorem 8795.4 (MetaCIC normalization-confluence residual boundary)**.** *Every confluence-facing read from an accepted* MetaCICNormalizationConfluenceFrontier$^{\uparrow}$ *packet factors through the finite boundary*

$$R \longmapsto J \longmapsto B \longmapsto A, O, H, C, P, N.$$

The residual-frontier row, residual-join row, and confluence-boundary row remain visible together; none may be replaced by an unconditional confluence theorem or by an erased subject-reduction assumption.

Proof. Project the carrier. The residual-frontier row is R, the join boundary is J, and the confluence-boundary row is B.

The audit row A can route the packet only after those rows are displayed. It therefore records a bounded normalization-confluence interface, not an unconditional confluence theorem.

The obstruction row O remains visible when a requested consumer would erase residual joins or subject-reduction obligations. The structural rows preserve that same boundary, so the packet cannot silently replace it by a stronger global claim. □

Theorem 8795.5 (MetaCIC normalization-confluence frontier nonescape)**.** *No accepted* MetaCICNormalizationCon *consumer may export evidence-free strong normalization, unconditional confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking. The consumer may export only the candidate-mediated normalization route and the residual-join/confluence-boundary rows displayed by* $T, K, S, R, J, B, A, O, H, C, P, N$.

Proof. Inspect the carrier inventory in Definition 8795.1. Every accepted row is listed explicitly, and the global claims named in the statement are not coordinates of the packet.

Definition 8795.3 fixes the normalization side to the candidate-mediated route through T, K, S. Definition 8795.4 fixes the confluence side to the displayed residual and boundary rows.

The obstruction, transport, replay, provenance, and local naming rows preserve those two routes. Since no excluded coordinate is introduced downstream, every public consumer remains inside the displayed frontier. □

Falsifiable prediction: Any accepted MetaCICNormalizationConfluenceFrontier$^{\uparrow}$ consumer that erases candidate evidence or residual-join boundary rows refutes the carrier, as does any consumer that reads confluence without the displayed confluence-boundary row.

Independence witness: metacic-candidate-normalization-confluence-handoff, metacic-residual-join-envelope, metacic-normalization-audit-packet, metacic-confluence-substitution-boundary: MetaCICNormalizati is not bijective to any listed sibling because it jointly retains candidate evidence, candidate-mediated SN, residual-frontier visibility, residual-join boundary, confluence boundary, audit routing, obstruction, transport, replay, provenance, and local naming.

The Lean follow-up for BEDC.Derived.MetaCICNormalizationConfluenceFrontierUp must provide ‘Nontrivial‘ and ‘FieldFaithful‘ instances in ‘lean4/BEDC/Derived/MetaCICNormalizationConfluenceFrontier before any downstream taste gate treats the carrier as a non-smoke frontier.

Closure status for NameCert$_{\text{MetaCICNormalizationConfluenceFrontier}^{\uparrow}}$.

Constructive story: A MetaCICNormalizationConfluenceFrontier$^{\uparrow}$ packet is generated from a closed typed MetaCIC term row, candidate evidence, candidate-mediated strong-normalization projection, residual-frontier row, residual-join boundary, confluence-boundary row, normalization-audit route, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationConfluenceFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 8795.1, Definition 8795.2, Definition 8795.3, Definition 8795.4, and Definition 8795.5 seed the candidate-mediated normalization/confluence frontier over residual joins and confluence-boundary rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalizationConfluenceFrontier}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization, unconditional confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-evidence retention, candidate-mediated SN projection, residual-frontier visibility, residual-join boundary retention, confluence-boundary retention, audit routing, obstruction exposure, transport, replay, provenance, local naming, and the taste-gate instances for BEDC.Derived.MetaCICNormalizati

Chapter 8796

A Concrete Naming Certificate for $\mathsf{OgdenLemma}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{OgdenLemma}^{\uparrow}$ packet records the marked-position strengthening of the context-free pumping lemma as finite NameCert data. It sits beside the grammar carrier of chapter 356 and the unmarked pumping route of chapter 2298; its role is to expose which marked terminal positions force the repeated nonterminal route, not to import host language equality, infinite parse trees, grammar quotients, or an ambient semantic pumping theorem.

Definition 8796.1 (Ogden lemma carrier)**.** *An* $\mathsf{OgdenLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (G, W, M, D, A, E, I, Y, H, C, R, Q, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *grammar packet, including the terminal alphabet, nonterminal carrier, start row, production ledger, and terminal-yield window.*
2. *W is the terminal word row, read as a* $\mathsf{List}^{\uparrow}$ *spine with a displayed length bound supplied by* $\mathsf{Nat}^{\uparrow}$*.*
3. *M is the finite marked-position ledger over W, recording only the displayed positions chosen for the Ogden read.*
4. *D is the finite derivation ledger whose prefix rows are read through the grammar packet and whose marked leaves are aligned with M.*
5. *A is the repeated-nonterminal exposure row forced by the marked derivation route.*
6. *E is the decomposition row naming the five displayed word pieces and the marked-position split attached to them.*
7. *I is the pump-index row.*
8. *Y is the yielded-word row obtained by replaying the marked pumping route at the displayed index.*
9. *H records componentwise* hsame *transport for the grammar, word, marked-position ledger, derivation ledger, repeated-nonterminal row, decomposition, pump index, yielded word, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through $G, W, M, D, A, E, I, Y, H, R, Q, N$.*
11. *R is the sibling route to the* $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ *packet, forgetting only the marked-position ledger while preserving the displayed derivation-prefix route.*
12. *Q is the* Pkg *provenance over* $\mathsf{OgdenLemma}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{ContextFreePumpingLemma}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{OgdenLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, W, M, D, A, E, I, Y, H, C, R, Q, N$. The carrier contains no host language set, grammar quotient, infinite derivation tree, selected parse tree, or ambient semantic pumping principle.

Lean4 ✓ `BEDC.Derived.OgdenLemmaUp.OgdenLemmaTasteGate single carrier alignment`

Theorem 8796.2 (Ogden lemma NameCert obligations)**.** *For every accepted* $\mathsf{OgdenLemma}^{\uparrow}$ *carrier*

$$O = (G, W, M, D, A, E, I, Y, H, C, R, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{OgdenLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the grammar packet, terminal word row, marked-position ledger, finite derivation ledger, repeated-nonterminal exposure row, decomposition, pump index, yielded word, transport, replay, sibling pumping route, provenance, and local naming. The marked-position ledger M must be displayed before A and E are read, and the sibling route R may forget only that ledger after the finite derivation-prefix path has been exposed.*

Proof. Project the accepted carrier of Definition 8796.1. The rows G, W, M, D, A, E, I, Y are the whole formal-language surface, while H, C, R, Q, N transport, replay, compare with the unmarked pumping sibling, source, and name the same packet.

First read the grammar and terminal word rows. The finite production ledger and terminal-yield window of G supply the derivation-prefix route, and W supplies the bounded terminal spine on which marked positions can be displayed.

Next expose M before the pumping witness. The repeated-nonterminal row A and decomposition row E are accepted only as reads of the marked derivation ledger D, so no unmarked host pumping argument can replace them.

Finally replay the chosen pump index through I and Y, then package the route through H, C, R, Q, N. The sibling row R erases only the marked-position annotation after the finite derivation-prefix data has been read, and no carrier coordinate stores a host language set, grammar quotient, infinite derivation tree, selected parse tree, or ambient semantic pumping principle. □

Theorem 8796.3 (Ogden lemma marked-position pumping route)**.** *Let*

$$O = (G, W, M, D, A, E, I, Y, H, C, R, Q, N)$$

be an accepted $\mathsf{OgdenLemma}^{\uparrow}$ *carrier. Every marked pumping read of O factors through the finite route*

$$G \longmapsto W \longmapsto M \longmapsto D \longmapsto A \longmapsto E \longmapsto I \longmapsto Y$$

before it reaches transport, replay, provenance, local naming, or the $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ *sibling route. A consumer may therefore recover a standard pumping-lemma surface only through the displayed forgetful row R, not by replacing the marked route with host language equality, quotient grammar semantics, an infinite parse tree, or an ambient membership theorem.*

Proof. Begin with Definition 8796.2. It orders the formal-language rows so that the grammar packet and terminal word are read before the marked-position ledger and derivation ledger.

The marked ledger then controls the pumping witness. The repeated-nonterminal row A and decomposition row E are downstream of M and D, and the pump index I can be replayed to the yielded row Y only along that displayed finite path.

The structural rows do not add a second pumping argument. Componentwise transport H and continuation replay C preserve the same path, while Q and N source and name it. The sibling row R gives the only route to the unmarked $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ surface, and that route forgets the mark ledger only after the derivation-prefix witness has been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{OgdenLemma}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OgdenLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8796.1, Definition 8796.2, and Definition 8796.3 seed the marked-position pumping surface over ContextFreeGrammarUp, ContextFreePumpingLemmaUp, ListUp, NatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OgdenLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host language equality, grammar quotient semantics, infinite parse trees, selected parse trees, ambient semantic pumping, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, marked-position exposure, repeated-nonterminal extraction, decomposition exactness, pump replay, sibling forgetful routing, transport, replay, provenance, and local naming for BEDC.Derived.OgdenLemmaUp.

Constructive story: An $\mathsf{OgdenLemma}^{\uparrow}$ packet is generated from a finite ContextFreeGrammarUp grammar packet, a terminal ListUp word row, a NatUp-bounded marked-position ledger, a finite derivation ledger, repeated-nonterminal exposure, decomposition, pump index, yielded word, componentwise hsame transport, Cont replay, sibling ContextFreePumpingLemmaUp routing, package provenance, and a local NameCert row.

Chapter 8797

A Concrete Naming Certificate for BolzanoWeierstrassFiniteSubsequence$^{\uparrow}$

Following Bolzano, Weierstrass, Bishop, and Bridges, the BolzanoWeierstrassFiniteSubsequence$^{\uparrow}$ packet records the finite-window face of the bounded-sequence compactness argument as BEDC NameCert data. It sits after the bounded real sequence carrier of chapter 1415, the finite-window compactness route of chapter 5145, the compact regular Cauchy subsequence surface of chapter 3913, the dyadic tolerance core of chapter 291, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a named finite subsequence-window object; it is not general sequential compactness, an infinite selected subsequence, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 8797.1 (Bolzano–Weierstrass finite-subsequence carrier)**.** *A* BolzanoWeierstrassFiniteSubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, K, W, I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the bounded* Real$^{\uparrow}$ *sequence source row, read through* BoundedRealSequence$^{\uparrow}$ *before any compactness-facing request is accepted.*
2. *K is the finite-window compactness row, recording the displayed finite interval, mesh, and pigeonhole ledger consumed from* FiniteWindowCompactness$^{\uparrow}$*.*
3. *W is the* StreamName$^{\uparrow}$ *observation row for the inspected sequence values and mesh witnesses.*
4. *I is the finite increasing index row, stored as a displayed finite list of positions rather than as an infinite subsequence function.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger comparing the sampled values with the selected finite mesh cell.*
6. *S is the compact regular Cauchy subsequence sibling row, available only as a finite-window handoff to chapter 3913.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerances.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row for the finite-window readback.*

9. *H records componentwise* hsame *transport for the source, compactness window, stream observations, finite index list, dyadic tolerance ledger, sibling handoff, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through $X, K, W, I, D, S, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{FiniteWindowCompactness}^{\uparrow}$, $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, W, I, D, S, R, E, H, C, P, N$. It has no coordinate for a host sequential compactness theorem, a selected infinite subsequence, countable choice, order trichotomy, quotient stream equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8797.2 (Bolzano–Weierstrass finite-subsequence NameCert obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$ *carrier*

$$B = (X, K, W, I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source X and the finite-window compactness row K;*

2. *classifier stability under componentwise* hsame *transport of $X, K, W, I, D, S, R, E, H, C, P, N$;*

3. *finite-window extraction of the displayed increasing index list I from the inspected stream observations W and dyadic tolerance ledger D;*

4. *the no-choice boundary asserting that I is a finite row and does not name an infinite selected subsequence, countable-choice schedule, trichotomy decision, quotient stream representative, or ambient completeness proof for* $\mathsf{Real}^{\uparrow}$*;*

5. *local* NameCert *packaging through S, R, E, H, C, P, N, with the compact regular Cauchy subsequence sibling read only after the finite window has been exposed.*

No Bolzano–Weierstrass obligation is stored outside those rows.

Proof. Project the carrier of Definition 8797.1. The bounded sequence source X and finite-window compactness row K are the only rows that can introduce the inspected window.

Read W, I, D next. The stream row W exposes the finite sequence observations, the finite index list I records the positions selected inside that displayed window, and the dyadic ledger D checks the tolerance comparison against the chosen mesh cell.

The sibling row S is therefore downstream of a finite extraction. It can hand the displayed data toward the compact regular Cauchy subsequence surface, but it cannot replace the finite index list with a selected infinite subsequence.

Finally route the readback and seal through R, E, H, C, P, N. Since the carrier has no coordinate for countable choice, trichotomy, quotient stream equality, host sequential compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the local NameCert surface is exactly the finite-window package listed in the theorem. □

Theorem 8797.3 (Bolzano–Weierstrass finite-window handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$ *packet must read the route*

$$X, K \longmapsto W, I, D \longmapsto S \longmapsto R, E \longmapsto H, C, P, N$$

before it treats the packet as compactness-facing data. The handoff exposes a bounded real sequence source, an explicit finite compactness window, a finite increasing index list, dyadic tolerance comparisons, regular rational readback, and a terminal real seal. It exports no theorem of general sequential compactness, no infinite selected subsequence, no countable-choice schedule, no trichotomy decision, no quotient stream equality, and no ambient completeness principle for $\mathsf{Real}^{\uparrow}$*.*

Proof. Apply Definition 8797.2. It orders the carrier so that boundedness and the compactness window are exposed before the stream observations and finite index list are consumed.

The dyadic ledger then fixes the tolerance comparison for the displayed finite window. The compact regular Cauchy subsequence sibling row can be read only after those finite rows have been presented.

The remaining rows preserve and name the same route. Regular rational readback, the terminal Real seal, componentwise transport, replay, provenance, and local naming do not add a second selection principle, so the consumer receives precisely the displayed finite-window handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8797.1, Definition 8797.2, and Definition 8797.3 seed the finite-window Bolzano–Weierstrass surface over bounded Real sequence, finite compactness window, StreamName, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general sequential compactness, infinite selected subsequences, countable choice, trichotomy, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite-window extraction, no-choice boundary, sibling compact-subsequence handoff, readback, seal, transport, replay, provenance, and local naming for BEDC.Derived.BolzanoWeierstrassFiniteSubsequenceUp.

Constructive story: A $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$ packet is generated from a bounded Real sequence source, an explicit finite compactness window, StreamName observations, a finite increasing index list, DyadicRatCore tolerance comparisons, a compact regular Cauchy subsequence sibling row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8798

A Concrete Naming Certificate for CompactIntervalFiniteSubcoverBudget$^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite-cover arguments, the CompactIntervalFiniteSubcoverBu packet records the finite budget by which a displayed located interval cover is read through dyadic mesh cells and refinement depth before any finite subcover conclusion is available. It sits after the dyadic interval-net surface of chapter 5966, the located interval cover packet of chapter 1885, and the finite subcover modulus budget of chapter 8630. The packet supplies a compact-interval consumer route; it does not claim general open-cover compactness, selected representatives of real names, countable choice, quotient stream equality, or ambient Real$^{\uparrow}$ completeness.

Definition 8798.1 (Compact interval located cover selector)**.** *A* CompactIntervalFiniteSubcoverBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, D, M, R, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, exposed only as located endpoint and containment data.*
2. *L is the* LocatedIntervalCover$^{\uparrow}$ *row listing the finite located interval cover requests and their endpoint-membership witnesses.*
3. *D is the* DyadicIntervalNet$^{\uparrow}$ *cell row, listing the dyadic mesh cells inspected by the cover selector.*
4. *M is the finite mesh-refinement row, recording the displayed refinement depth and parent-child cell ledger.*
5. *R is the finite selector budget, listing the retained cover entries and the mesh cells certified by each retained entry.*
6. *S is the* StreamName$^{\uparrow}$ *window row opened by the displayed cover, mesh, and refinement data.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback and terminal* Real$^{\uparrow}$ *seal row reached only after I, L, D, M, R, S have been displayed.*
8. *H records componentwise* hsame *transport for interval source, located cover, dyadic cells, refinement depth, selector budget, readback, and seal rows.*
9. *C records displayed* Cont *replay from the located cover request through the finite mesh cells, refinement depth, selector budget, and terminal readback.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{FiniteMeshRefinementTree}^{\uparrow}$, $\mathsf{FiniteSubcoverModulusBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, D, M, R, S, Q, H, C, P, N$. It has no coordinate for an arbitrary open cover, a host-side subcover selector, quotient equality of real names, countable choice, or ambient completeness.

Theorem 8798.2 (Compact interval finite-subcover budget)**.** *For every accepted* CompactIntervalFiniteSubcoverBud *carrier*

$$B = (I, L, D, M, R, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, compact interval source exposure, located cover exposure through L, dyadic interval-net exposure through D, finite mesh-refinement depth through M, finite selector budget through R, finite window and regular readback through S, Q, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N. Every compact-interval finite-subcover consumer must read the sibling routes of chapter 5966, chapter 1885, and chapter 8630 before the retained finite selector budget is public.*

Proof. Project the carrier of Definition 8798.1. The only interval and cover inputs are the displayed compact interval source row I and the located cover row L.

The finite route then passes through D and M. The dyadic interval-net row lists the cells being inspected, and the finite mesh-refinement row records the depth and parent-child ledger used to compare those cells against the located cover requests.

The selector budget R is downstream of those finite rows. It may retain only displayed cover entries and displayed mesh cells, while S and Q keep the finite observation windows, regular rational readback, and Real seal on the same route.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate is present, the certificate cannot export arbitrary open-cover compactness, countable choice, quotient stream equality, host real equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}$ consumer exposes compact interval source rows, $\mathsf{LocatedIntervalCover}^{\uparrow}$ cover data, $\mathsf{DyadicIntervalNet}^{\uparrow}$ cells, finite mesh-refinement depth, finite selector budget, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any finite-subcover budget is public.

Independence witness: dyadic interval net, located interval cover, finite subcover modulus budget: $\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds located cover requests, dyadic mesh cells, refinement depth, retained cover entries, readback windows, Real sealing, transport, replay, provenance, and local naming into one compact-interval finite-cover budget packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}$ packet is generated from compact interval source rows, LocatedIntervalCoverUp cover rows, DyadicIntervalNetUp cell rows, finite mesh-refinement depth, a finite selector budget, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalFiniteSubcoverBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8798.1 and Definition 8798.2 seed the finite compact-interval subcover budget route over located covers, dyadic interval nets, and finite refinement depth.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalFiniteSubcoverBudget}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, arbitrary subcover selection, quotient stream equality, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cover exposure, dyadic-cell exposure, finite refinement-depth exactness, selector-budget exactness, readback ordering, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalFiniteSubcoverBudgetUp.

Chapter 8799

A Concrete Naming Certificate for $\mathsf{RhFreeZeroInitiality}^{\uparrow}$

The $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ packet realizes the RH-free zero-generation initiality route of chapter 9255 as finite NameCert data. It consumes the zero-fiber and legal-generator factorization rows of chapter 8772 before forming its RH-free generated-zero signature and no-circularity audit. It records a generated zero signature, a no-circularity audit, initial-algebra rows, exactness obligations, and fixed-half refusal boundaries without asserting the Riemann Hypothesis, analytic continuation, zero completeness, or a fixed-half section. The packet is a BEDC-native audit surface for the route; it is not a zero generator and not a proof of the analytic claim.

The no-circularity row also consumes the halting-boundary refusal route of chapter 5564 and the fixed-statement separation route of chapter 8824 before its generated-carrier initiality row is public. Thus an accepted initiality read has already separated constructor generation from RH-route halting refusal and fixed-statement closure boundaries.

Definition 8799.1 (RH-free zero initiality NameCert)**.** *A* $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (F, A, G, I, E, K, X, H, C, P, N)$$

with the following displayed rows:

1. *F is the RH-free zero signature row, listing constructor kinds for the zero-generation endofunctor.*
2. *A is the* $\mathsf{NoRHCirc}$ *audit row, refusing any constructor input that consumes an* $\mathsf{OnCritLine}$ *witness, a fixed-half section witness, the proposition* RH*, or an unfactored zero-source oracle.*
3. *G is the generated zero carrier row, naming the displayed initial algebra over F.*
4. *I is the initiality row, recording the fold and uniqueness surface available for generated zero rows.*
5. *E is the zero-presentation exactness obligation row: soundness, faithfulness, and essential coverage are stored as separate finite obligations.*
6. *K is the fixed-half closure obligation row, recording constructor-by-constructor preservation requirements without supplying a fixed-half witness as input.*
7. *X is the no-circularity refusal ledger, retaining failed rows whose constructor reads* $\mathsf{OnCritLine}$ *or a fixed-half section before generation.*

8. *H records componentwise* hsame *transport for signature, audit, generated carrier, initiality, exactness, fixed-half closure, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a zero-generation constructor request through F, A, G, I, E, K, X, H.*
10. *P is the* Pkg *provenance over* $\mathsf{RhFreeZeroInitiality}^{\uparrow}$*, the RH-free zero-generation vision route,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RhFreeZeroInitiality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, A, G, I, E, K, X, H, C, P, N$. It has no coordinate for an analytic zero source, a proof that all zeta zeros are generated, an OnCritLine *witness, a fixed-half section, or an RH theorem.*

Theorem 8799.2 (RH-free zero initiality no fixed-half circularity)**.** *For every accepted* $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ *carrier*

$$Z = (F, A, G, I, E, K, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RhFreeZeroInitiality}^{\uparrow}}$ *obligation surface has exactly five mathematical rows:*

1. *RH-free signature admission through F;*
2. *no-circularity audit through A and refusal ledger X;*
3. *generated zero carrier and initiality exposure through G, I;*
4. *zero-presentation exactness obligations through E;*
5. *fixed-half closure obligations through K.*

The structural rows H, C, P, N transport, replay, source, and name those five rows. No accepted constructor row may consume an OnCritLine *witness or fixed-half section witness before the generated carrier, exactness obligations, and fixed-half closure obligations have been displayed.*

Proof. Project the carrier of Definition 8799.1. The visible mathematical rows are F, A, G, I, E, K, X, followed by the structural rows H, C, P, N.

The no-circularity rows are read before initiality is used. If a constructor consumes OnCritLine, a fixed-half section, RH, or an unfactored zero-source oracle, the row is stored in X as a refusal rather than in the accepted constructor surface.

Initiality then applies only to the generated carrier G over the admitted signature F. The exactness row E and fixed-half closure row K remain obligations displayed beside the initiality row; neither is supplied by initiality alone.

Transport and replay preserve the same route. Since P and N contain no analytic zero source, fixed-half witness, section witness, or RH theorem coordinate, the packet records the no-circularity audit without proving the analytic conclusion. □

Falsifiable prediction: Any accepted $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ zero-generation constructor row fails if it consumes an OnCritLine witness, a fixed-half section witness, the proposition RH, or an unfactored zero-source oracle before the no-circularity audit, generated carrier, exactness obligations, fixed-half closure obligations, transport, replay, provenance, and local NameCert rows are displayed.

Independence witness: `rh_layered_translation`, `zeta_self_similar_recovered_carrier`, `prime_fingerprint_separation`, `zero_skew_eventflow_target`: $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ is not bijective to any listed sibling because the NoRHCirc row and the separated initiality, exactness, and fixed-half closure obligations jointly form a no-circularity audit surface rather than a zero carrier, a prime fingerprint classifier, an event-flow target, or a fixed-half section.

Closure status for $\mathsf{NameCert}_{\mathsf{RhFreeZeroInitiality}^{\uparrow}}$.

Constructive story: A $\mathsf{RhFreeZeroInitiality}^{\uparrow}$ packet is generated from an RH-free zero signature, a NoRHCirc audit row, generated zero carrier rows, initiality rows, exactness obligations, fixed-half closure obligations, refusal ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RhFreeZeroInitiality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8799.1 and Definition 8799.2 seed the finite no-circularity audit surface for RH-free zero-generation initiality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RhFreeZeroInitiality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close analytic continuation, zero completeness, an RH proof, fixed-half section existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked signature admission, NoRHCirc refusal, generated-carrier initiality, zero-presentation exactness, fixed-half closure, transport, replay, provenance, and local naming for BEDC.Derived.RhFreeZeroInitialityUp.

Chapter 8800

A Concrete Naming Certificate for $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$

Following Bishop and Bridges on bounded monotone convergence, the $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ packet records the finite boundary at which bounded monotone sequence data may hand off to the Bishop Cauchy convergence criterion and then to the L10 Real route. It sits beside the criterion packet of chapter 6459, the bounded real sequence surface of chapter 1415, the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the constructive finite handoff from monotone bounded data to Cauchy-readiness; it does not assert an ambient completeness theorem for all $\mathsf{Real}^{\uparrow}$ streams.

Definition 8800.1 (Bishop monotone convergence boundary carrier)**.** *A* $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, M, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence schedule row, recording the finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the monotone source is inspected.*
2. *B is the boundedness row, exposing the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *bound ledger attached to the same windows.*
3. *M is the monotone ledger, recording the displayed order comparisons that make later tails compatible with earlier bounds.*
4. *Q is the Bishop Cauchy convergence criterion row, reached only after the boundedness and monotone ledgers have stayed visible.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying the requested radii, split budgets, comparison scales, and endpoint tolerances used by Q.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the inspected windows as regular rational data for Real-facing consumers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, B, M, Q, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the schedule, boundedness, monotone ledger, criterion, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a monotone-convergence request through $S, B, M, Q, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, M, Q, D, R, E, H, C, P, N$. It has no coordinate for a selected limit, quotient equality of streams, countable choice, detached monotone convergence, host real completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8800.2 (Bishop monotone convergence boundary NameCert obligations)**.** *For every accepted* $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ *carrier*

$$K = (S, B, M, Q, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *carrier admission for the sequence schedule, boundedness row, monotone ledger, Bishop Cauchy convergence criterion row, dyadic tolerance row, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming;*
2. *bounded-monotone exposure, requiring B and M to be read from the finite windows scheduled by S before any convergence criterion is consumed;*
3. *criterion handoff, requiring Q to consume the displayed boundedness and monotone rows rather than an external monotone convergence theorem;*
4. *finite L10 seal nonescape, requiring D, R, E to expose dyadic tolerances, regular rational readback, and the terminal Real seal in order;*
5. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name the same finite route without adding selected limits, quotient stream equality, countable choice, host real completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8800.1. The accepted packet displays exactly the schedule, boundedness, monotone, criterion, tolerance, readback, seal, and structural rows.

Bounded-monotone exposure is read from S, B, M. The boundedness row and monotone ledger are attached to the same finite windows, so no consumer can replace them by a detached convergence theorem.

The criterion handoff is the role of Q. It receives only the displayed boundedness and monotone data, then passes through the dyadic tolerance row D and the regular readback row R before any Real-facing seal is available.

Finally E, H, C, P, N add only the terminal seal, transport, replay, provenance, and local naming. Since the carrier has no selected-limit, quotient-stream, countable-choice, host-completeness, or ambient-completeness coordinate, the listed families exhaust the local NameCert surface. □

Theorem 8800.3 (Bishop monotone convergence boundary L10 nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{BoundedRealSequence}^{\uparrow},\ \mathsf{BishopCauchyConvergenceCriterion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In carrier coordinates, the consumer reads S for StreamName windows, B and M for bounded monotone exposure, Q for the Bishop Cauchy convergence criterion, D for DyadicRatCore tolerances, R for RegSeqRat readback, and E for the Real seal before H, C, P, N may transport, replay, source, and name the route.

Proof. Start with a Real-facing read of an accepted packet. Projection exposes the ordered carrier rows $S, B, M, Q, D, R, E, H, C, P, N$.

The window row S is the only schedule coordinate. The boundedness row B and monotone ledger M are attached to that schedule, so the consumer receives bounded monotone sequence data only through finite StreamName windows.

Apply Definition 8800.2. The criterion handoff requires Q to consume the displayed S, B, M rows, and the finite L10 seal obligation then routes the read through D, R, E.

The structural rows H, C, P, N preserve and name the same route. Therefore every Real-facing consumer passes through StreamName, BoundedRealSequence, BishopCauchyConvergenceCriterion, DyadicRatCore, RegSeqRat, and Real rows, and none can extract a selected limit, quotient stream equality, countable choice, detached monotone convergence theorem, host real completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ Real-facing read factors through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{BoundedRealSequence}^{\uparrow}$ boundedness, a monotone ledger, $\mathsf{BishopCauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bishop-cauchy-convergence-criterion, boundedrealsequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded monotone source exposure, criterion handoff, dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}$ packet is generated from StreamName windows, BoundedRealSequence boundedness, a monotone ledger, a BishopCauchyConvergenceCriterion handoff, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8800.1, Definition 8800.2, and Definition 8800.3 seed the finite monotone-bounded convergence boundary over StreamName, BoundedRealSequence, BishopCauchyConvergenceCriterion, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMonotoneConvergenceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, detached monotone convergence, host real completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-monotone exposure, criterion handoff, finite L10 seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopMonotoneConvergenceBoundaryUp.

Chapter 8801

Located Interval Enclosure NameCert Construction

Following Bishop and Bridges on constructive real intervals, the $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ packet records a finite enclosure of a displayed real-name read by dyadic endpoints before any nested-interval or completion consumer may use it. It consumes the dyadic endpoint arithmetic of chapter 291, finite observation windows from chapter 126, regular rational readback from chapter 301, the terminal real-name seal of chapter 38, and the dyadic interval-cell route of chapter 5966. The packet names a displayed enclosure row; it does not assert interval compactness, choose a completed point, identify host real equality, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8801.1 (Located interval enclosure carrier)**.** *A* $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, U, O, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L and U are* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for the lower and upper dyadic bounds.*
2. *O is the endpoint order row, displaying that the lower endpoint precedes the upper endpoint inside the same dyadic tolerance ledger.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window inspected before the enclosure is admitted.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same window and dyadic tolerance budget.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the named real read enclosed by L, U, W, R.*
6. *H records componentwise* hsame *transport for the endpoints, order row, containment witness, stream window, readback row, Real seal, replay row, provenance row, and local name.*
7. *C records displayed* Cont *replay from the window request through the endpoint, containment, readback, seal, transport, provenance, and naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalEnclosure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, U, O, W, R, S, H, C, P, N$. It has no coordinate for host real equality, quotient representatives, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8801.2 (Located interval enclosure carrier extraction)**.** *Every accepted* $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ *read exposes the dyadic endpoint rows* L, U*, the endpoint order row* O*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S*, and the structural rows* H, C, P, N *before an enclosure judgment is public.*

Proof. Project the carrier of Definition 8801.1. The tuple lists exactly the endpoint, order, window, readback, seal, transport, replay, provenance, and local naming rows.

The sibling routes named in the introduction supply those rows only as displayed finite data. Dyadic endpoints and the interval-net cell are read before the stream window and regular readback, and the Real seal is reached only after those finite rows have been displayed.

Since the classifier has no coordinate beyond $L, U, O, W, R, S, H, C, P, N$, no public enclosure read can bypass this extraction route. □

Theorem 8801.3 (Located interval enclosure endpoint-refinement transport)**.** *Let* $E = (L, U, O, W, R, S, H, C, P, N$ *be an accepted* $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ *carrier. If refined dyadic endpoints* L', U' *preserve the endpoint order row and the containment witness through the same* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell, then the enclosure transports to*

$$E' = (L', U', O, W, R, S, H, C, P, N)$$

without changing the stream window, regular readback, Real seal, replay, provenance, or local name.

Proof. Read the endpoint refinement through the dyadic endpoint rows and the interval-net cell. The hypotheses keep the order and containment witness attached to the same displayed dyadic route.

Apply the componentwise hsame transport row H. It carries endpoint replacement while leaving the finite observation window, regular readback, Real seal, replay, provenance, and local name on their original rows.

The transported carrier therefore remains an accepted enclosure packet. Because no host real equality or quotient coordinate is present, the transport is only the displayed endpoint-refinement route. □

Theorem 8801.4 (Located interval enclosure Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ *packet factors through the displayed route*

$$L, U, O, W, R, S, H, C, P, N.$$

The consumer receives a sealed real-name read inside dyadic enclosure rows, but it receives no host real equality, quotient representative, selected limit, countable-choice schedule, interval compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with carrier extraction, Definition 8801.2. It forces the dyadic endpoints, order row, finite window, regular readback, and terminal seal to be visible before any enclosure judgment is public.

Use endpoint-refinement transport, Definition 8801.3, for the only permitted endpoint variation. Refinement preserves the displayed rows and cannot introduce a new host-side coordinate.

Finally read the refusal inventory in the classifier of Definition 8801.1. Since the accepted packet has no coordinate for the excluded objects, the Real-facing route cannot escape the finite enclosure surface. □

Falsifiable prediction: An accepted $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first displaying dyadic endpoints, endpoint order, a finite StreamName window, a RegSeqRat readback row, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, dyadic-interval-net: no listed sibling supplies the whole packet $E = (L, U, O, W, R, S, H, C, P, N)$, because each sibling contributes only endpoint arithmetic, finite observation, regular readback, terminal sealing, or interval-cell routing while this chapter binds them into one enclosure certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalEnclosure}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalEnclosure}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, an endpoint order row, a containment witness routed through DyadicIntervalNet, a StreamName finite observation window, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalEnclosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8801.1, Definition 8801.2, Definition 8801.3, and Definition 8801.4 seed the finite enclosure packet over dyadic endpoints, finite windows, regular readback, terminal sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalEnclosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close interval compactness, point extraction, host real equality, quotient representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier extraction, endpoint-refinement transport, and Real-seal nonescape for BEDC.Derived.LocatedIntervalEnclosureUp.

Chapter 8802

Golden-Mean Shift Window-Six NameCert Construction

Following Lind and Marcus, the golden-mean subshift is the finite-type no-adjacent-one shift. The $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ packet realizes the window-six kernel of Definition 9347.15 as finite BHist data: a six-cell binary window, a two-state transition classifier, a shift-stability row, a finite-window growth row, Zeckendorf carry-facing provenance through chapter 278 and chapter 544, and local NameCert evidence. The packet names the finite symbolic-dynamics carrier used by the Axis-Zeckendorf vision; it is not an invariant measure, a two-sided completion, a host entropy theorem, or a claim that all Zeckendorf carries have been bridged.

Definition 8802.1 (Golden-mean window-six carrier)**.** *A* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, K, T, S, F, H, C, P, N)$$

with the following displayed rows:

1. *W is a six-symbol binary window row.*
2. *K is the no-adjacent-one transition-kernel classifier of Definition 9347.15, restricted to adjacent pairs inside W.*
3. *T is the transition ledger recording which adjacent pairs are admitted by K.*
4. *S is the one-step shift-stability row: deleting the first cell and reading the remaining five-cell suffix preserves the admitted adjacent-pair rows.*
5. *F is the finite-window growth row, recording the Fibonacci recurrence of Definition 9347.16 at the inspected window length.*
6. *H records componentwise* hsame *transport for the window, transition classifier, transition ledger, shift row, growth row, replay row, provenance row, and local name.*
7. *C records displayed* Cont *replay from the window read through K, T, S, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$*, the Axis-Zeckendorf kernel of Definition 9347.15,* $\mathsf{ZeckendorfNormal}^{\uparrow}$*,* $\mathsf{ZCarry}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, K, T, S, F, H, C, P, N$. It has no coordinate for a host subshift object, invariant measure, entropy limit, two-sided completion, unrestricted carry propagation, or bridge credential.

Theorem 8802.2 (Golden-mean transition-kernel obligations)**.** *For every accepted* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ *carrier* $G = (W, K, T, S, F, H, C, P, N)$*, the local transition-kernel obligations are exactly pair admission by the no-adjacent-one classifier, transport of those pair rows under* hsame*, and replay of the admitted finite window through* C, P, N*.*

Proof. Project the carrier of Definition 8802.1. The only transition coordinates are K and T, so every adjacent-pair decision is read through the displayed no-adjacent-one classifier.

The transport row H carries the same pair decisions under accepted renaming of the finite window. Replay and provenance through C, P, N can name the route but cannot add another transition classifier.

Since no host subshift, measure, entropy, or unrestricted carry coordinate belongs to the carrier, the listed rows exhaust the local transition obligations. □

Theorem 8802.3 (Golden-mean window-six shift stability)**.** *Every accepted* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ *carrier exports shift stability only by reading* S *after* W, K, T *have displayed the no-adjacent-one pair rows. The shifted five-cell suffix remains inside the same transition-kernel classifier and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *surface.*

Proof. Read the finite window W and transition rows K, T. They display every adjacent pair whose preservation matters after the first cell is dropped.

The shift row S is then a suffix read of the same finite window. It preserves all adjacent-pair rows that remain in the suffix and does not introduce a second classifier.

Finally H, C, P, N transport, replay, source, and name that suffix route. Thus shift stability is a finite NameCert row, not a host dynamical-system completion. □

Theorem 8802.4 (Golden-mean window-six finite growth route)**.** *Any finite-window growth read from an accepted* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ *packet factors through* F *after the transition classifier* K *and ledger* T *have been displayed. The read may cite the recurrence of Definition 9347.16, but it exports only the inspected finite growth row and not a host entropy limit or a completed inverse-limit carrier.*

Proof. Start with the kernel obligations of Definition 8802.2. They ensure that the no-adjacent-one grammar has been displayed as finite adjacent-pair data.

The growth row F reads the same grammar at the inspected window length. It records the Fibonacci recurrence locally, using the vision proposition only as the named source of the finite recurrence.

The refusal inventory in Definition 8802.1 excludes host entropy and completed inverse-limit coordinates. Therefore the growth route remains a finite NameCert row. □

Falsifiable prediction: An accepted $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ read that exports shift stability or finite growth without first displaying the six-cell window, no-adjacent-one classifier, transition ledger, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: axiszeckendorf-kernel, zeckendorf-normal, zcarry: no listed source supplies the whole packet $G = (W, K, T, S, F, H, C, P, N)$, because the vision source supplies the grammar while the Zeckendorf and carry-facing siblings supply normal-word provenance rather than the local window-six transition, shift, growth, transport, replay, provenance, and NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}}$**.**

Constructive story: A GoldenMeanShiftWindowSix$^{\uparrow}$ packet is generated from a six-cell binary BHist window, the no-adjacent-one transition kernel, a transition ledger, a shift-stability row, a finite-window growth row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{GoldenMeanShiftWindowSix}\uparrow}$, seed)

Scope closed: Definition 8802.1, Definition 8802.2, Definition 8802.3, and Definition 8802.4 seed the finite no-adjacent-one window packet over transition, shift, growth, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{GoldenMeanShiftWindowSix}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close invariant measures, entropy limits, two-sided completions, unrestricted Zeckendorf carry propagation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked transition-kernel obligations, shift stability, finite growth, and provenance ledger rows for BEDC.Derived.GoldenMeanShiftWindowSixUp.

Chapter 8803

A Concrete Naming Certificate for $\mathsf{ContinuousLattice}^{\uparrow}$

Following Gierz–Hofmann–Keimel–Lawson–Mislove–Scott, the $\mathsf{ContinuousLattice}^{\uparrow}$ packet records the domain-theoretic carrier in which every displayed element has a finite approximation row and every directed read remains tied to a complete-lattice substrate. It sits between the directed-supremum surface of chapter 5905, the complete meet–join surface of chapter 5155, the Scott-open consumer of chapter 5950, and the point-free open comparison route of chapter 5153. The packet names finite way-below, directed-join, and Scott-handoff rows for L10 approximation consumers; it is not a host powerset lattice, not an unrestricted frame theorem, not a selected basis theorem, and not an ambient spatiality claim.

Definition 8803.1 (Continuous lattice carrier)**.** *A* $\mathsf{ContinuousLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, C, W, A, S, L, H, R, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DCPO}^{\uparrow}$ *source row, exposing finite order, directed-family, supremum, and lower-bound exactness data.*
2. *C is the* $\mathsf{CompleteLattice}^{\uparrow}$ *source row, exposing displayed certified-family joins and meets.*
3. *W is the way-below ledger: every approximant named by the packet is tied to a finite directed-window witness.*
4. *A is the approximation row, recording that each listed element is read through directed joins of its visible way-below approximants.*
5. *S is the* $\mathsf{ScottTopology}^{\uparrow}$ *handoff row for upper-set and directed-inaccessibility reads.*
6. *L is the* $\mathsf{LocaleSpace}^{\uparrow}$ *comparison row for point-free finite-open and cover-refinement observations.*
7. *H records componentwise* hsame *transport for all visible continuous-lattice rows.*
8. *R records displayed* Cont *replay from directed reads to approximation rows and Scott-facing handoff rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ContinuousLattice}^{\uparrow}$, $\mathsf{DCPO}^{\uparrow}$, $\mathsf{CompleteLattice}^{\uparrow}$, $\mathsf{ScottTopology}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ContinuousLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, C, W, A, S, L, H, R, P, N$. It has no coordinate for host powersets, arbitrary joins outside the displayed complete-lattice row, selected bases, quotient opens, global spatiality, Lean verification, or bridge checking.

Theorem 8803.2 (Continuous lattice NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousLattice}^{\uparrow}$ *carrier $Q = (D, C, W, A, S, L, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousLattice}^{\uparrow}}$ *surface consists exactly of DCPO source admission, complete-lattice source admission, way-below ledger exposure, approximation-row exposure, Scott-topology handoff, locale comparison, componentwise transport, continuation replay, provenance, and local naming. Every approximation consumer must read D, C, W, A before using S or L as a topology-facing observation.*

Proof. Project the carrier of Definition 8803.1. The order-theoretic substrate is split into the directed-complete row D and the complete-lattice row C, so no consumer can read approximation data before both source rows are visible.

The finite approximation evidence is stored in W and A. The way-below ledger W names the displayed approximants, while A records the directed-join read by which each listed element is approximated.

The topology-facing rows are downstream. The Scott handoff S may be read only after the directed-supremum and approximation rows have been inspected, and the locale comparison L may be read only as point-free finite-open data tied to the same packet.

Finally H, R, P, N preserve the displayed rows as certificate data. Thus the local surface is finite NameCert evidence over the listed rows, not a host complete lattice or unrestricted frame construction. □

Theorem 8803.3 (Continuous lattice Scott handoff)**.** *Every* $\mathsf{ScottTopology}^{\uparrow}$ *or* $\mathsf{LocaleSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{ContinuousLattice}^{\uparrow}$ *packet must factor its read through*

$$\mathsf{DCPO}^{\uparrow}, \quad \mathsf{CompleteLattice}^{\uparrow}, \quad W, \quad A$$

before it treats S or L as public open data. The handoff exports the finite approximation route needed by Scott-open and locale-facing consumers, but it does not claim spatiality, a selected basis, a host frame of all opens, or equality with an unrestricted continuous-domain construction.

Proof. Use Definition 8803.2. It requires every accepted packet to expose the DCPO row, complete-lattice row, way-below ledger, and approximation row before topology-facing observations are public.

The $\mathsf{ScottTopology}^{\uparrow}$ consumer reads S only as an upper-set and directed-inaccessibility handoff tied to D, W, A. The $\mathsf{LocaleSpace}^{\uparrow}$ consumer reads L only as a point-free comparison row tied to the same finite approximation packet.

Since neither S nor L contains a coordinate for selected bases, host powersets, quotient opens, or spatiality, the handoff remains the displayed finite route from order and approximation rows to topology-facing observations. □

Falsifiable prediction: An accepted $\mathsf{ContinuousLattice}^{\uparrow}$ consumer that exports Scott-open or locale-facing data while omitting the $\mathsf{DCPO}^{\uparrow}$ row D, $\mathsf{CompleteLattice}^{\uparrow}$ row C, way-below ledger W, or approximation row A refutes the packet.

Independence witness: dcpo, complete-lattice, scott-topology, locale-space: none is bijective to $\mathsf{ContinuousLattice}^{\uparrow}$ because this packet jointly carries directed completeness, complete-lattice family

joins and meets, finite way-below approximation, Scott handoff, locale comparison, transport, replay, provenance, and local naming as one certificate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousLattice}^{\uparrow}$ packet is generated from a DCPO source row, CompleteLattice source row, finite way-below ledger, approximation row, ScottTopology handoff, LocaleSpace comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8803.1, Definition 8803.2, and Definition 8803.3 seed the finite continuous-lattice approximation route over DCPO, CompleteLattice, ScottTopology, LocaleSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousLattice}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close selected bases, host powerset lattices, arbitrary frame completeness, spatiality, quotient-open equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, way-below ledger exactness, approximation-row exactness, ScottTopology handoff, LocaleSpace comparison, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousLatticeUp.

Lean4 (statement) `BEDC.Derived.ContinuousLatticeUp`

Chapter 8804

A Concrete Naming Certificate for $\mathsf{WeierstrassFunction}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{WeierstrassFunction}^{\uparrow}$ packet records a finite approximation route for the classical nowhere-smooth trigonometric series by using only displayed windows, dyadic ledgers, and regular real-name handoffs. It sits after the majorant route of chapter 399, the uniform Cauchy series surface of chapter 1736, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the constructive finite-window approximation route; it does not assert a host function space, differentiability failure, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8804.1 (Weierstrass function finite-approximation carrier)**.** *A* $\mathsf{WeierstrassFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, A, M, U, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite trigonometric-polynomial window, listing only the inspected frequencies, phase samples, and partial sums of the approximation.*
2. *A is the dyadic amplitude and frequency ledger for the same window, including the finite amplitude weights and frequency growth entries read by T.*
3. *M is the M-test majorant row that bounds the displayed tail requests through positive dyadic comparison data.*
4. *U is the uniform Cauchy threshold row, stated as a finite request-to-tail ledger rather than as an ambient infinite sum.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window surface through which sample points, partial sums, and tail requests are observed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the visible rational approximants.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the uniform Cauchy threshold and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for trigonometric windows, dyadic ledgers, majorant rows, uniform thresholds, stream windows, readbacks, real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{WeierstrassFunction}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WeierstrassFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, A, M, U, S, Q, R, H, C, P, N$. It has no coordinate for a completed function space, a global infinite series object, differentiability failure, quotient equality of functions, countable choice, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8804.2 (Weierstrass function finite-window NameCert obligations)**.** *For every accepted* $\mathsf{WeierstrassFunction}^{\uparrow}$ *carrier*

$$W = (T, A, M, U, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeierstrassFunction}^{\uparrow}}$ *surface consists exactly of finite trigonometric-polynomial window exposure through T, dyadic amplitude and frequency ledger exposure through A, M-test majorant exposure through M, uniform Cauchy threshold exposure through U, StreamName observation through S, RegSeqRat readback through Q, RealUp sealing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8804.1. The visible analytic rows are exactly T, A, M, U, S, Q, R, so every local certificate read must expose the finite partial polynomial window, its dyadic amplitude and frequency ledger, the majorant row, the uniform threshold, the observed stream windows, the regular rational readback, and the final real-name seal.

Read the dependencies in route order. The M-test row is the majorant interface of chapter 399; the uniform Cauchy threshold is the series route of chapter 1736; the visible samples are $\mathsf{StreamName}^{\uparrow}$ windows; and the rational approximants pass through $\mathsf{RegSeqRat}^{\uparrow}$ before reaching the $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a completed function space, host infinite series, differentiability failure, quotient function equality, choice schedule, selected limit, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8804.3 (Weierstrass function uniform Cauchy route)**.** *Every approximation or Real-facing consumer of an accepted* $\mathsf{WeierstrassFunction}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{WeierstrassMTest}^{\uparrow},\ \mathsf{UniformCauchySeries}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer receives a finite trigonometric-polynomial window, dyadic amplitude and frequency data, a majorant row, and a uniform Cauchy threshold before any terminal Real seal can be inspected. The route exports no global infinite series object, quotient equality of functions, selected limit, countable-choice schedule, differentiability claim, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 8804.2. They force each consumer read through the finite rows T, A, M, U, S, Q, R before the packet can be used as a named approximation object.

The M-test row supplies the majorant-tail control, and the uniform Cauchy threshold turns that control into a finite request-to-tail ledger. The stream and regular-rational rows then determine which approximants are visible before the terminal real-name row is consumed.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-level coordinate is present in the carrier, every consumer factors through the displayed finite route. □

Theorem 8804.4 (Weierstrass function Real-seal nonescape)**.** *Let*

$$W = (T, A, M, U, S, Q, R, H, C, P, N)$$

be an accepted $\mathsf{WeierstrassFunction}^{\uparrow}$ *carrier. Any read of the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *from* W *is preceded by the displayed rows* T, A, M, U, S, Q *and is replayed only through* H, C, P, N*. Thus the real-name handoff cannot be used to infer a host function object, a completed ambient series, differentiability failure, quotient function equality, countable choice, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 8804.3. A Real-facing read must first expose the trigonometric-polynomial window, dyadic amplitude and frequency ledger, majorant row, uniform Cauchy threshold, stream windows, and regular rational readback.

The seal R is therefore terminal for this displayed route, not a source of additional host-level structure. The only remaining rows are transport, replay, provenance, and local naming rows, and each preserves the already displayed packet.

Since the carrier lacks coordinates for the excluded objects, reading R cannot introduce them. The Real seal is a handoff boundary for the finite approximation route and not an ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{WeierstrassFunction}^{\uparrow}$ approximation read exposes finite trigonometric-polynomial windows, dyadic amplitude and frequency ledgers, M-test majorant rows, uniform Cauchy thresholds, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any Real-facing consumer is served.

Independence witness: weierstrassmtest, uniformcauchyseries, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{WeierstrassFunction}^{\uparrow}$ because this packet alone binds the finite trigonometric window, dyadic amplitude and frequency ledger, majorant row, uniform threshold, observation route, readback, seal, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{WeierstrassFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WeierstrassFunction}^{\uparrow}$ packet is generated from finite trigonometric-polynomial windows, dyadic amplitude and frequency ledgers, M-test majorant rows, uniform Cauchy thresholds, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeierstrassFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8804.1, Definition 8804.2, Definition 8804.3, and Definition 8804.4 seed the finite approximation route for the constructive Weierstrass function packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeierstrassFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed function space, a global infinite series object, differentiability failure, quotient equality of functions, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigonometric-window exposure, dyadic ledger exactness, M-test majorant exposure, uniform Cauchy thresholding, StreamName observation, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeierstrassFunctionUp.

Chapter 8805

A Concrete Naming Certificate for $\mathsf{PhillipsLemma}^{\uparrow}$

Following Phillips on lifting/local-reflexivity phenomena in Banach-space theory, the $\mathsf{PhillipsLemma}^{\uparrow}$ packet records the finite BEDC surface by which a bounded lifting request is read through Banach endpoints, dual-pair windows, finite approximation data, and an explicit obstruction row. It sits after the functional-analysis surface of chapter 94, the Banach carrier of chapter 90, the bounded-operator route of chapter 1644, and the operator-ideal consumer boundary of chapter 213. The certificate names the local lifting packet only; it does not assert a quotient dual space, a globally selected local inverse, reflexivity of the source, exactness of every short sequence, or ambient choice over finite-dimensional subspaces.

Definition 8805.1 (PhillipsLemma carrier)**.** *A* $\mathsf{PhillipsLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, T, B, D, A, O, H, C, Q, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the norm, metric, completion, and real-seal reads available to the lifting request.*
2. *T is the target* $\mathsf{Banach}^{\uparrow}$ *row, displayed with the same Banach-facing endpoint discipline.*
3. *B is the bounded lifting row. It records the bounded-operator surface through which a local lift is inspected, including its action window, norm-bound row, and compatibility with the Banach endpoints.*
4. *D is the dual-pair readback row. It displays the finite dual tests used by the lifting request before any operator-ideal or functional-analysis consumer reads the packet.*
5. *A is the finite approximation witness row, listing the inspected finite-dimensional source window, tolerance ledger, and bounded image data.*
6. *O is the obstruction/refusal row. It records which global lifting, quotient-dual, or selector claims are not exported by the packet.*
7. *H records componentwise* hsame *transport for the Banach endpoint rows, bounded lifting row, dual-pair readback, approximation witness, obstruction row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a lifting request through S, T, B, D, A, O, H.*

9. *Q is the* Pkg *provenance over* $\mathsf{PhillipsLemma}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhillipsLemma}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient-dual equality, a selected complement, a global right inverse, countable choice over subspaces, host operator equality, or reflexivity beyond the displayed Banach endpoint rows.

Lean4 ✓ `BEDC.Derived.PhillipsLemmaUp.PhillipsLemmaTasteGate single carrier alignment`

Theorem 8805.2 (PhillipsLemma NameCert obligations)**.** *For every accepted* $\mathsf{PhillipsLemma}^{\uparrow}$ *carrier*

$$P = (S, T, B, D, A, O, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PhillipsLemma}^{\uparrow}}$ *surface is exhausted by five grouped rows: Banach source and target admission through S, T; bounded lifting exposure through B; dual-pair readback and finite approximation exposure through D, A; obstruction and refusal visibility through O; and structural closure through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, Q, N. Every functional-analysis, Banach, bounded-operator, or operator-ideal consumer must read those grouped rows before treating a local Phillips lifting witness as public certificate data.*

Proof. Project the carrier of Definition 8805.1. The Banach endpoint rows S, T are the only source and target data available to the packet, and the bounded lifting row B is the only row carrying an operator action and norm-bound ledger.

Read the dual and approximation data next. The row D supplies the displayed dual-pair tests, while A supplies the finite-dimensional window and tolerance ledger used by the local lifting witness.

The obstruction row O blocks the packet from exporting a global right inverse, selected complement, quotient-dual equality, or choice schedule. Thus the lifting read remains local to the displayed finite window.

Finally replay H, C, Q, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same rows, so the five grouped obligations exhaust the NameCert surface before any consumer in chapter 94, chapter 90, chapter 1644, or chapter 213 can cite the packet. □

Falsifiable prediction: Every accepted $\mathsf{PhillipsLemma}^{\uparrow}$ read exposes Banach source and target rows, a bounded lifting row, a dual-pair readback row, a finite approximation witness row, an obstruction/refusal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any functional-analysis or operator-ideal consumer is served.

Independence witness: functionalanalysis, banach, boundedlinearoperator, operatorideal: none alone is bijective with $\mathsf{PhillipsLemma}^{\uparrow}$ because this packet alone binds the Banach endpoint rows, bounded lifting window, dual-pair readback, finite approximation witness, obstruction/refusal row, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{PhillipsLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PhillipsLemma}^{\uparrow}$ packet is generated from Banach source and target rows, a bounded lifting row, a dual-pair readback row, a finite approximation witness row, an obstruction/refusal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PhillipsLemma}^{\uparrow}}$, seed)

Scope closed: Definition 8805.1 and Definition 8805.2 seed the local Banach lifting certificate route over functional-analysis, bounded-operator, operator-ideal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PhillipsLemma}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-dual equality, selected complements, global right inverses, exactness of arbitrary short sequences, countable choice over subspaces, host operator equality, reflexivity beyond displayed endpoint rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Banach endpoint admission, bounded lifting exposure, dual-pair readback, finite approximation witness exactness, obstruction/refusal visibility, transport, replay, provenance, and local naming for BEDC.Derived.PhillipsLemmaUp.

Chapter 8806

A Concrete Naming Certificate for $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$

Following Helly's selection theorem for uniformly bounded finite-variation functions, the HellySelectionFiniteVariatio packet records the finite partition, shared-window, and regular-readback data that a constructive Real or Completion consumer may inspect before any subsequence language is used. It sits over the compact interval source of chapter 3986, the finite partition ledger of chapter 398, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite cell discipline of chapter 5966. The packet names only displayed partition budgets, selected finite windows, shared-window agreement, transport, replay, provenance, and local naming; it does not claim open-cover compactness, countable choice, a host-selected subsequence, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8806.1 (Helly selection finite-variation carrier)**.** *A* $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, P, V, W, A, R, E, T, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{CompactInterval}^{\uparrow}$ *source row, recording the named compact interval only through displayed endpoint, cell, and finite-cover data.*
2. *P is the finite partition row. It lists the inspected partition points, adjacent interval cells, and refinement witnesses used by the variation request.*
3. *V is the finite variation-budget row, read through* $\mathsf{BoundedVariation}^{\uparrow}$ *and recording only the displayed partition sums and common-refinement budgets.*
4. *W is the selected-window row, recording the finite set of indices, interval cells, and rational scales retained for the Helly-facing request.*
5. *A is the shared selected-window agreement row, comparing two retained windows only through displayed endpoint reads, variation budgets, and cell incidences.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected finite windows and their rational tail budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, P, V, W, A, R have been displayed.*
8. *T records componentwise* hsame *transport for interval data, partitions, variation budgets, selected windows, agreement rows, readback, and Real seals.*

9. *C records displayed* Cont *replay from compact interval cells through partition variation budgets, selected-window agreement, regular readback, and the Real seal.*
10. *Q is the* Pkg *provenance over* $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$, $\mathsf{CompactInterval}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HellySelectionFiniteVariation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, P, V, W, A, R, E, T, C, Q, N$. It has no coordinate for a host subsequence theorem, arbitrary open-cover compactness, choice-selected branches, quotient equality of functions, variation measures, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8806.2 (Helly selection finite-variation NameCert obligations)**.** *For every accepted* $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ *carrier*

$$H = (I, P, V, W, A, R, E, T, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellySelectionFiniteVariation}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows; compact interval exposure through I; finite partition exposure through P; variation-budget exposure through V; selected-window exposure through W; shared selected-window agreement through A;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport through T; displayed* Cont *replay through C;* Pkg *provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 8806.1. The compact interval and finite partition rows I, P are the only domain-facing data available to the packet.

The variation and window rows are then read in order. Row V supplies the displayed finite partition sums and common-refinement budgets, while W lists the retained windows and A records their shared agreement through the same endpoint and cell data.

The readback and seal remain downstream. Row R gives regular rational access only for those selected windows, and row E seals the corresponding real names only after I, P, V, W, A, R have been displayed.

Finally T, C, Q, N transport, replay, source, and name the same finite surface. Since the carrier has no coordinate for host subsequence selection, open-cover compactness, quotient equality of functions, variation measures, or ambient Real completeness, none of those principles is a local NameCert obligation. □

Theorem 8806.3 (Helly selection shared-window route)**.** *Every accepted* $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ *packet exposes the route*

$$\mathsf{CompactInterval}^{\uparrow},\ \mathsf{DyadicIntervalNet}^{\uparrow},\ \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through its displayed rows $I, P, V, W, A, R, E, T, C, Q, N$ before any downstream consumer reads a Helly-style selected-window conclusion. The agreement row A is a finite comparison of retained windows over the same compact interval cells and variation budgets, not a selected infinite subsequence and not a function-space compactness theorem.

Proof. Start with the compact interval and dyadic-cell side. The interval source row I and partition row P force every retained window to be attached to displayed finite cells and their endpoint data.

Use the variation row V next. It records finite partition sums and common-refinement budgets, so the selected-window row W receives bounded-variation data only as a visible finite ledger.

The agreement row A compares retained windows over those same rows. It can test shared endpoint reads, shared variation budgets, and shared cell incidence, but it cannot introduce a host subsequence or a completed function-space object.

Finally the readback row R and Real seal E are reached after the finite agreement route, and T, C, Q, N preserve that route. Thus every downstream Helly-facing read is a displayed finite-variation window handoff. □

Falsifiable prediction: An accepted $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ consumer that reaches a Real-facing selected-window conclusion without reading compact interval cells, finite partition variation budgets, shared selected-window agreement, and RegSeqRat readback refutes the carrier surface.

Independence witness: compactinterval, boundedvariation, regseqrat, real, dyadicintervalnet: none is bijective to $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ because this packet jointly carries compact interval cells, finite partition variation budgets, selected windows, shared-window agreement, regular readback, Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HellySelectionFiniteVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{HellySelectionFiniteVariation}^{\uparrow}$ packet is generated from compact interval rows, finite partition rows, bounded-variation budgets, selected finite windows, shared-window agreement rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellySelectionFiniteVariation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8806.1, Definition 8806.2, and Definition 8806.3 seed the finite Helly selection packet over compact interval cells, partition variation budgets, shared selected-window agreement, regular rational readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellySelectionFiniteVariation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, host-selected subsequences, quotient equality of functions, variation measures, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval exposure, finite partition exposure, variation-budget exposure, selected-window agreement, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HellySelectionFiniteVariationUp.

Chapter 8807

A Concrete Naming Certificate for $\mathsf{VCDimension}^{\uparrow}$

Following Vapnik–Chervonenkis and Sauer–Shelah, the $\mathsf{VCDimension}^{\uparrow}$ packet records finite classifier complexity by displaying the sample rows, Boolean label rows, concept-class rows, shattering-pattern ledger, and growth-boundary row that a downstream statistical-learning consumer may inspect. It sits beside the Boolean and list carriers of chapter 32 and chapter 36, the unary size rows of chapter 28, the displayed membership discipline of chapter 3668, the finite-set surface of chapter 7157, and the finite concentration neighbour chapter 5726. The packet names only finite shattering data and a finite growth-boundary obligation; it does not assert PAC bounds, probability measures, empirical risk convergence, a complete Sauer–Shelah proof, host finite-set quotients, or ambient powerset semantics.

Definition 8807.1 (VC dimension finite classifier carrier)**.** *A* $\mathsf{VCDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (X, K, S, L, F, T, G, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite sample-domain row, read through the displayed membership discipline and the finite-set surface rather than through a host powerset.*
2. *K is the inspected cardinality row, a* $\mathsf{Nat}^{\uparrow}$ *ledger recording the sample size and the growth-boundary size request.*
3. *S is the finite sample row, a* $\mathsf{List}^{\uparrow}$ *spine of displayed sample points with an explicit support boundary in X.*
4. *L is the Boolean label-row ledger, listing only the inspected* $\mathsf{Bool}^{\uparrow}$ *label patterns on the sample row S.*
5. *F is the concept-class row, recording the finite family of displayed classifier rows that are tested against S and L.*
6. *T is the shattering-pattern ledger: for each displayed label pattern in L, it names a concept row in F whose readback agrees with that pattern on S.*
7. *G is the growth-boundary row, recording the first displayed size at which the packet refuses a full shattering ledger and exposes only the counted pattern boundary.*

8. *H records componentwise* hsame *transport for sample-domain, cardinality, sample, label, concept-class, shattering, growth-boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through $X, K, S, L, F, T, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{VCDimension}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{KuratowskiFiniteSet}^{\uparrow}$, $\mathsf{HoeffdingFiniteSum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{VCDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, S, L, F, T, G, H, C, P, N$. It has no coordinate for a host concept class, quotient finite set, powerset of labels, probability measure, risk functional, ambient growth theorem, or selected infinite sample family.

Theorem 8807.2 (VC dimension shattering obligation)**.** *For every accepted* $\mathsf{VCDimension}^{\uparrow}$ *carrier*

$$V = (X, K, S, L, F, T, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VCDimension}^{\uparrow}}$ *shattering surface consists exactly of carrier admission for the sample-domain row, inspected cardinality row, finite sample row, Boolean label-row ledger, concept-class row, shattering-pattern ledger, transport, replay, provenance, and local naming. The shattering ledger T must be read after S, L, F are displayed, and each entry of T may cite only a displayed concept row whose Boolean readback on S matches the corresponding label row in L.*

Proof. Project the accepted carrier of Definition 8807.1. The rows X, K, S, L, F, T are the full finite classifier surface before any growth boundary is inspected.

First read the sample route. The row X supplies the displayed membership boundary, K records the relevant size request, and S supplies the finite list of sample points whose labels can be tested.

Next read the classifier route. The row L lists the Boolean patterns being requested, while F lists the displayed concept rows available for the test. The ledger T is accepted only after both rows are visible, so a shattering read cannot be replaced by a host powerset, quotient finite set, or ambient concept-class equality.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a probability measure, risk functional, infinite sample family, or host classifier semantics, the listed rows exhaust the local shattering obligation. □

Theorem 8807.3 (VC dimension growth-boundary obligation)**.** *Let*

$$V = (X, K, S, L, F, T, G, H, C, P, N)$$

be an accepted $\mathsf{VCDimension}^{\uparrow}$ *carrier. Any read of the growth-boundary row G is downstream of the displayed sample-domain, cardinality, sample, Boolean label, concept-class, and shattering-pattern rows. Thus a statistical-learning or finite-combinatorics consumer may use G only as the finite refusal boundary attached to the displayed shattering ledger, not as a PAC guarantee, probability space, empirical-risk convergence theorem, complete Sauer–Shelah proof, host finite-set quotient, or ambient growth function oracle.*

Proof. Use Definition 8807.2. It orders the finite classifier rows so that the sample, label, concept, and shattering ledgers are read before a consumer can inspect any boundary row.

The growth-boundary row G records only the displayed size at which the packet refuses a full shattering ledger and exposes a counted pattern boundary. It is therefore tied to the same

X, K, S, L, F, T route rather than to an external Sauer–Shelah theorem or host powerset calculation.

The structural rows H, C, P, N preserve that finite route under transport, replay, provenance, and local naming. Because the carrier has no coordinate for PAC probability, empirical risk, quotient finite sets, or ambient growth semantics, no consumer can read such data from G. □

Falsifiable prediction: Every accepted $\mathsf{VCDimension}^{\uparrow}$ classifier-complexity read exposes finite sample-domain rows, NatUp cardinality ledgers, ListUp sample spines, BoolUp label rows, concept-class rows, shattering-pattern ledgers, growth-boundary rows, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any learning-facing consumer is served.

Independence witness: bool, list, nat, setlike, kuratowski finite set, hoeffding finite sum: none alone is bijective with $\mathsf{VCDimension}^{\uparrow}$ because this packet alone binds finite sample rows, Boolean label ledgers, displayed concept rows, shattering-pattern witnesses, growth-boundary refusal, transport, replay, provenance, and local naming as one classifier-complexity certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{VCDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{VCDimension}^{\uparrow}$ packet is generated from finite sample-domain rows, NatUp cardinality ledgers, ListUp sample rows, BoolUp label ledgers, displayed concept-class rows, shattering-pattern ledgers, growth-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VCDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8807.1, Definition 8807.2, and Definition 8807.3 seed the finite VC-dimension classifier-complexity route over Boolean labels, finite samples, displayed concept rows, and finite growth boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VCDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close PAC bounds, probability measures, empirical risk convergence, Sauer–Shelah proof completeness, host finite-set quotients, ambient powerset semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sample-row exposure, Boolean label-row exactness, concept-class row exposure, shattering-pattern ledger exactness, growth-boundary refusal, transport, replay, provenance, and local naming for BEDC.Derived.VCDimensionUp.

Chapter 8808

A Concrete Naming Certificate for $\mathsf{JordanCanonicalForm}^{\uparrow}$

Following the classical Jordan canonical form over algebraically closed fields, the $\mathsf{JordanCanonicalForm}^{\uparrow}$ packet records the finite certificate by which a displayed endomorphism is read as a block diagonal matrix with eigenvalue labels and nilpotent successor rows. It sits after the matrix carrier of chapter 49, the vector-space source of chapter 47, the polynomial ledger of chapter 50, and the scalar field carrier of chapter 45. It also records the finite spectral comparison surface shared with chapter 6037 and chapter 6602. The packet names only displayed finite rows; it is not an algebraic-closure theorem, a selected host eigenbasis, quotient eigenspace equality, or an ambient spectral theorem.

Definition 8808.1 (Jordan canonical form carrier)**.** *A* $\mathsf{JordanCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, V, F, Q, E, B, K, S, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *row, recording the finite matrix entries, multiplication windows, and row-column indices of the source endomorphism.*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row, recording the finite-dimensional vector-space presentation and the displayed basis windows used by the matrix read.*
3. *F is the* $\mathsf{Field}^{\uparrow}$ *row, recording the scalar carrier over which the eigenvalue names and polynomial coefficients are interpreted.*
4. *Q is the* $\mathsf{Poly}^{\uparrow}$ *row, recording the characteristic-polynomial, minimal-polynomial, and annihilator ledgers visible to the block scheduler.*
5. *E is the eigenvalue-name row, listing the displayed roots that label the Jordan blocks inside the accepted packet.*
6. *B is the block-layout row, a finite ledger of block sizes, block order, diagonal eigenvalue labels, and superdiagonal successor positions.*
7. *K is the chain-stability row, recording the finite nilpotent-chain checks attached to each block size and eigenvalue label.*
8. *S is the similarity-transport row, recording the displayed source presentation, block presentation, and change-of-presentation rows.*

9. *D is the comparison row to* $\mathsf{SingularValueDecomposition}^{\uparrow}$ *and* $\mathsf{PolarDecomposition}^{\uparrow}$*, recording that those sibling packets may read the matrix-facing spectral boundary only through the displayed finite matrix and transport rows.*
10. *H records componentwise* hsame *transport for $M, V, F, Q, E, B, K, S, D$ and the local certificate rows.*
11. *C records displayed* Cont *replay along the route*

$$M, V, F, Q, E, B, K, S, D \longmapsto H, C, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{JordanCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{SingularValueDecomposition}$ $\mathsf{PolarDecomposition}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCanonicalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, F, Q, E, B, K, S, D, H, C, P, N$. It has no coordinate for a host proof that the field is algebraically closed, an unrecorded eigenvector choice, quotient equality of generalized eigenspaces, uniqueness of block order, or a global spectral theorem.

Theorem 8808.2 (Jordan canonical form NameCert obligations)**.** *For every accepted* $\mathsf{JordanCanonicalForm}^{\uparrow}$ *carrier $J = (M, V, F, Q, E, B, K, S, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{JordanCanonicalForm}^{\uparrow}}$ *surface consists exactly of carrier admission, matrix exposure through M, vector-space exposure through V, scalar-field exposure through F, polynomial-ledger exposure through Q, eigenvalue-name exposure through E, block-layout exposure through B, nilpotent-chain stability through K, similarity transport through S, spectral sibling comparison through D, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8808.1. The rows $M, V, F, Q, E, B, K, S, D$ are the mathematical data, while H, C, P, N transport, replay, source, and name that finite packet.

The eigenvalue and block rows are read only through the matrix, vector-space, field, and polynomial rows. The packet therefore displays the finite block layout and its scalar labels before a local NameCert row can certify the Jordan canonical form surface.

The comparison row D does not import a singular-value or polar-decomposition theorem. It only records the matrix-facing route by which those siblings may inspect the same finite spectral boundary. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 8808.3 (Jordan canonical finite chain stability)**.** *Every accepted* $\mathsf{JordanCanonicalForm}^{\uparrow}$ *read that consumes a Jordan block must pass through the finite chain route*

$$M, V, F, Q, E, B, K, S, H, C, P, N.$$

The row K records the nilpotent successor checks attached to the displayed block sizes in B and eigenvalue names in E; it is stable only under the componentwise transport row H and replay row C of the same carrier. A downstream consumer receives no quotient generalized-eigenspace equality, no selected host eigenbasis, no proof of algebraic closure, and no ambient spectral theorem from this route.

Proof. Use the NameCert obligations of Definition 8808.2. Any block read must expose the source matrix, vector-space presentation, scalar field row, polynomial ledger, eigenvalue-name row, block-layout row, and chain-stability row before the structural rows are consumed.

Transport by H preserves the same displayed rows componentwise, and replay by C follows the same finite route. Since neither row adds a host eigenspace object or a spectral theorem, the nilpotent-chain check remains a finite carrier row. □

Theorem 8808.4 (Jordan canonical sibling noncollapse)**.** *The* $\mathsf{JordanCanonicalForm}^{\uparrow}$ *packet is not bijective to* $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{SingularValueDecomposition}^{\uparrow}$, *or* $\mathsf{PolarDecomposition}^{\uparrow}$. *It jointly carries matrix entries, vector-space presentation data, scalar-field rows, polynomial ledgers, eigenvalue-name rows, finite block layout, nilpotent-chain stability, similarity transport, spectral sibling comparison, structural transport, replay, provenance, and local naming, while each listed sibling omits at least one of those displayed rows.*

Proof. Compare the carrier coordinates in Definition 8808.1. The linear-algebra source siblings provide matrix, vector-space, polynomial, or field rows, but none of them alone carries the eigenvalue-name, block-layout, chain-stability, and similarity-transport rows together with local naming.

The spectral siblings $\mathsf{SingularValueDecomposition}^{\uparrow}$ and $\mathsf{PolarDecomposition}^{\uparrow}$ read through their own factorization surfaces. Row D lets them inspect a matrix-facing boundary, but it does not identify their packets with the Jordan canonical block ledger. Hence no listed sibling is bijective to the full carrier. □

Falsifiable prediction: Every accepted $\mathsf{JordanCanonicalForm}^{\uparrow}$ consumer must display Matrix, VecSpace, Field, Polynomial, eigenvalue-name, block-layout, nilpotent-chain, similarity-transport, sibling-comparison, hsame, Cont, Pkg, and local NameCert rows before reading a Jordan canonical form certificate.

Independence witness: matrix, vecspace, polynomial, field, singular-value-decomposition, polar-decomposition: $\mathsf{JordanCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because this packet alone binds source matrix rows, vector-space presentation, scalar-field data, polynomial ledgers, eigenvalue labels, finite Jordan blocks, nilpotent-chain stability, similarity transport, spectral sibling comparison, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCanonicalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanCanonicalForm}^{\uparrow}$ packet is generated from Matrix rows, VecSpace basis windows, Field scalar rows, Polynomial ledgers, displayed eigenvalue-name rows, finite Jordan-block layout, nilpotent-chain stability, similarity transport, spectral sibling comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanCanonicalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8808.1, Definition 8808.2, Definition 8808.3, and Definition 8808.4 seed the finite Jordan canonical form certificate over matrix, vector-space, field, polynomial, eigenvalue, block, nilpotent-chain, similarity, sibling-comparison, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanCanonicalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim algebraic closure, quotient generalized-eigenspace equality, selected host eigenvectors, canonical uniqueness of block order, ambient spectral theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, eigenvalue-name admission, block-ledger exactness, nilpotent-chain stability, similarity-transport nonescape, spectral sibling comparison, provenance, and local NameCert exactness for BEDC.Derived.JordanCanonicalFormUp.

Chapter 8809

A Concrete Naming Certificate for $\mathsf{FiniteMeshSelection}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{FiniteMeshSelection}^{\uparrow}$ packet records the finite mesh choice used when a displayed real interval or dyadic interval net is inspected at one explicit tolerance. It sits after the dyadic arithmetic spine of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the endpoint-and-enclosure packet of chapter 4111, the terminal real-name seal of chapter 38, and the dyadic interval-net surface of chapter 5966. The chapter names a finite selection row only; it is not compactness, an infinite mesh sequence, selected real representatives, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8809.1 (Finite mesh selection carrier)**.** *A* $\mathsf{FiniteMeshSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the positive dyadic precision request, mesh denominator, outward rounding budget, and endpoint comparison bounds.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance.*
4. *I is the* $\mathsf{RealInterval}^{\uparrow}$ *endpoint-and-enclosure row being inspected.*
5. *E is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *selected finite cell row, listing exactly the dyadic interval cells retained by the mesh selection at tolerance D.*
6. *H records componentwise* hsame *transport for tolerance, windows, readback, interval enclosure, selected cells, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite mesh request through D, W, R, I, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteMeshSelection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMeshSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, I, E, H, C, P, N$. The carrier has no coordinate for open-cover compactness, an infinite mesh sequence, arbitrary subcover selection, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.FiniteMeshSelectionUp`

Theorem 8809.2 (Finite mesh selection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMeshSelection}^{\uparrow}$ *carrier* $S = (D, W, R, I, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMeshSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance admission, finite window exposure, regular rational readback, interval endpoint-and-enclosure exposure, selected dyadic interval-net cells, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No finite mesh selection obligation is stored outside these rows.*

Proof. Project the carrier of Definition 8809.1. The rows D, W, R, I, E are the whole selection surface: a dyadic tolerance, finite stream windows, regular readback, one interval enclosure, and the retained dyadic interval-net cells.

Read the finite route in order. The tolerance row D opens W, the window row gives R, and the interval row I is inspected only through those finite observations before the selected cells E are named.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since none of the refused compactness, infinite mesh, quotient, choice, or completeness principles is a carrier coordinate, the local obligations are exhausted by the displayed rows. □

Theorem 8809.3 (Finite mesh selection Real completion route)**.** *Every completion-facing read of an accepted* $\mathsf{FiniteMeshSelection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{RealInterval}^{\uparrow} \longrightarrow \mathsf{DyadicIntervalNet}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Consequently a Real or Completion consumer receives finite dyadic cells selected at an explicit tolerance before any terminal real-name seal is used.

Proof. Start with the dyadic tolerance row D. It is the only row that records the requested mesh precision and outward rounding budget.

Next read W and R. These rows expose finite stream windows and regular rational readback before the endpoint-and-enclosure packet I can be inspected at the chosen tolerance.

Finally route the selected cells through E and the structural rows. The selected dyadic interval-net cells are therefore finite data already tied to D, W, R, I, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of that route. The carrier contains no compactness, infinite mesh, choice, quotient, or ambient completeness coordinate that could supply an alternate completion route. □

Falsifiable prediction: Every accepted FiniteMeshSelectionUp read displays a DyadicRatCoreUp tolerance row, StreamNameUp finite windows, RegSeqRatUp readback, a RealIntervalUp endpoint-and-enclosure row, DyadicIntervalNetUp selected cells, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before any RealUp-facing completion consumer receives mesh data.

Independence witness: dyadicratcore, streamname, regseqrat, realinterval, real, dyadicintervalnet: $\mathsf{FiniteMeshSelection}^{\uparrow}$ is not bijective to any listed sibling because it binds one finite mesh-selection row to a displayed tolerance, finite windows, regular readback, interval enclosure, dyadic cell retention, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMeshSelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMeshSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8809.1, Definition 8809.2, and Definition 8809.3 seed the finite mesh-selection route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicIntervalNet}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMeshSelection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FiniteMeshSelectionUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite mesh limits, arbitrary subcover selection, quotient real equality, countable choice, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, finite-window exposure, regular readback, selected-cell retention, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMeshSelectionUp.

Constructive story: A FiniteMeshSelection$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName finite windows, RegSeqRat readback, a RealInterval endpoint-and-enclosure row, selected DyadicIntervalNet cells, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8810

A Concrete Naming Certificate for $\mathsf{KrylovSubspace}^{\uparrow}$

Following Krylov and the standard Arnoldi/Lanczos finite-iteration literature, the $\mathsf{KrylovSubspace}^{\uparrow}$ packet records the finite span generated by repeated application of one displayed linear operator to one displayed seed vector. It sits after the matrix carrier of chapter 49, the vector-space carrier of chapter 47, the linear-map surface of chapter 48, and the control-facing observer route of chapter 5909. The packet names only the finite iterate-and-span certificate; it does not export an infinite cyclic subspace, a spectral decomposition, a chosen orthonormal Arnoldi basis, a Lanczos tridiagonalization, or host equality between matrix and operator presentations.

Definition 8810.1 (KrylovSubspace carrier)**.** *A* $\mathsf{KrylovSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, A, v, n, I, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row carrying the finite vector-space operations, zero row, scalar action, and classifier used by the packet.*
2. *A is the common* $\mathsf{LinMap}^{\uparrow}/\mathsf{Mat}^{\uparrow}$ *row. It records both the operator action and the finite coordinate matrix surface through which iterates are inspected.*
3. *v is the seed-vector row, displayed as a vector of V before any iterate ledger is consumed.*
4. *n is the finite depth row naming the last iterate admitted to the packet.*
5. *I is the finite iterate ledger*
$$v, \quad Av, \quad A^2v, \quad \ldots, \quad A^nv,$$
with each entry read through the same V and A rows.
6. *L is the span and provenance ledger. It records finite linear-combination support over the iterate list I and the refusal to identify the span with a host subspace outside the displayed rows.*
7. *H records componentwise* hsame *transport for $V, A, v, n, I, L, C, P, N$.*
8. *C records displayed* Cont *replay from the seed-vector request through the finite iterate and span ledgers.*
9. *P is the* Pkg *provenance over* $\mathsf{KrylovSubspace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{KrylovSubspace}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite-dimensional closure, a quotient invariant subspace, a selected orthogonal basis, a minimal polynomial theorem, spectral convergence, or equality between the matrix row and the linear-map row beyond the displayed provenance.

Theorem 8810.2 (KrylovSubspace NameCert boundary)**.** *For every accepted* $\mathsf{KrylovSubspace}^{\uparrow}$ *carrier*

$$K = (V, A, v, n, I, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KrylovSubspace}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier admission for the vector-space row V, operator row A, seed row v, depth row n, iterate ledger I, span ledger L, and structural rows H, C, P, N;*
2. *finite iterate closure, requiring each displayed entry of I to be generated by repeated use of the same A on the same seed row v inside V;*
3. *classifier transport, requiring every span membership or nonmembership read to pass through the componentwise* hsame *row H and the replay row C;*
4. *matrix-linear-map provenance, requiring the coordinate matrix surface and the linear-map action to be linked only by the displayed A and P rows;*
5. *ledger exhaustion, requiring Arnoldi, Lanczos, reachability, observability, or polynomial-action consumers to read only $V, A, v, n, I, L, H, C, P, N$ before citing the packet.*

No NameCert obligation for $\mathsf{KrylovSubspace}^{\uparrow}$ *is stored outside those five rows.*

Proof. Project the carrier of Definition 8810.1. The accepted packet exposes one vector-space row, one matrix-linear-map row, one seed vector, one finite depth, one finite iterate ledger, one finite span ledger, and the four structural rows.

Finite iterate closure is read from I. Since each entry is displayed as $A^k v$ for $0 \le k \le n$, the packet never needs an infinite orbit or a host subspace completion.

Classifier transport and replay are the rows H and C. They preserve the vector, operator, iterate, and span ledgers componentwise and do not add a hidden basis, spectral coordinate, or convergence claim.

Finally read provenance through P and local naming through N. The matrix and linear-map presentations are connected only by the displayed row A, with $\mathsf{Poly}^{\uparrow}$ appearing as finite polynomial-action provenance rather than as a minimal-polynomial theorem. Therefore the five listed obligations exhaust the local certificate surface. □

Falsifiable prediction: Every accepted $\mathsf{KrylovSubspace}^{\uparrow}$ read exposes one VecSpace row, one Matrix/LinearMap row, one seed-vector row, one finite depth, one finite iterate ledger, one finite span ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any Arnoldi, Lanczos, reachability, observability, or polynomial-action consumer is served.

Independence witness: matrix, vecspace, linearmap, polynomial, luenbergerobserver: none alone is bijective with $\mathsf{KrylovSubspace}^{\uparrow}$ because this packet alone binds the seed vector, finite operator iterates, finite span ledger, matrix-linear-map provenance, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{KrylovSubspace}^{\uparrow}}$.

Constructive story: A KrylovSubspace$^{\uparrow}$ packet is generated from a VecSpace row, a Matrix/LinearMap action row, a seed-vector row, a finite depth row, a finite iterate ledger, a span and provenance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{KrylovSubspace}^{\uparrow}}$, seed)

Scope closed: Definition 8810.1 and Definition 8810.2 seed the finite Krylov iterate-and-span certificate over vector-space, matrix, linear-map, polynomial-action, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KrylovSubspace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Arnoldi orthogonalization, Lanczos tridiagonalization, minimal-polynomial exactness, spectral approximation, infinite cyclic subspaces, host matrix-operator equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite iterate closure, classifier transport, matrix-linear-map provenance, and ledger exhaustion for BEDC.Derived.KrylovSubspaceUp.

Chapter 8811

A Concrete Naming Certificate for $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$

Following Zeckendorf and the Fibonacci symbolic-dynamics tradition, the $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ packet realizes the stable-window, residual-fiber, and external golden-phase projection rows of chapter 9346 as finite NameCert data. It sits beside the carry-normalization surface of chapter 608 and the finite golden-mean window of chapter 8802. The packet names the visible phase shell only; it does not turn golden phase into a kernel object, a quotient of histories, a physical law, or a real-completion theorem.

Definition 8811.1 (ZeckendorfPhaseShell carrier)**.** *A* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (m, \Omega, X, F, R, G, H, C, P, N)$$

with the following displayed rows:

1. *m is the finite window-width row.*
2. *Ω is the raw binary-window row $\{0,1\}^m$.*
3. *X is the stable Zeckendorf-window row, admitting exactly the visible words with no adjacent* 11 *pattern.*
4. *F is the fold row from raw windows to stable visible windows, read through the finite Fibonacci-value ledger of the vision chapter.*
5. *R is the residual-fiber row. For each visible word x, it records the finite fiber of raw windows that fold to x and the refusal to identify that fiber with a quotient class of histories.*
6. *G is the external golden-phase projection row. It records the phase readout attached to X and R without placing that readout in the BEDC kernel.*
7. *H records componentwise* hsame *transport for $m, \Omega, X, F, R, G, C, P, N$.*
8. *C records displayed* Cont *replay through the stable window, fold, residual-fiber, and phase rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$, $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$, $\mathsf{GoldenMeanShiftWindowSix}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ZeckendorfPhaseShell}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting raw histories by the fold, no kernel equality between visible phase and raw microstate, no invariant-measure claim, no physical-law assertion, and no completed real limit.

Theorem 8811.2 (ZeckendorfPhaseShell NameCert boundary)**.** *For every accepted* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ *carrier*

$$Z = (m, \Omega, X, F, R, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ZeckendorfPhaseShell}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *stable-window admission, requiring X to expose the no-adjacent-11 classifier over the finite raw window Ω;*
2. *fold exactness, requiring F to display the finite Fibonacci-value fold before any visible word is consumed;*
3. *residual-fiber preservation, requiring R to remain visible whenever a consumer reads the folded window;*
4. *external phase transport, requiring the golden-phase row G to be replayed as projection data rather than as a kernel object or quotient of histories;*
5. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the displayed rows m, Ω, X, F, R, G.*

No Zeckendorf phase-shell obligation is stored outside those rows.

Proof. Project the carrier of Definition 8811.1. The raw window, stable visible window, fold, residual-fiber ledger, and phase projection are all finite rows before transport or replay is applied.

Stable-window admission and fold exactness come from X and F. The visible word can be consumed only after the no-adjacent-11 classifier and the Fibonacci fold have been displayed.

The residual-fiber row R prevents the fold readout from erasing raw histories. Thus a consumer of the visible phase shell receives a stable word together with its finite fiber ledger, not a quotient equality of histories.

Finally read G, H, C, P, N. The golden phase row is transported and replayed only as external projection data, while provenance names the Zeckendorf carry-normalization and golden-mean window sources. Therefore the five rows exhaust the local NameCert boundary without promoting phase projection to a kernel object. □

Falsifiable prediction: Every accepted $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ read exposes a finite window width, raw binary window, stable Zeckendorf window, Fibonacci fold, residual-fiber ledger, external golden-phase projection row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before any visible phase-shell consumer is served.

Independence witness: zeckendorfcarrynormalization, goldenmeanshiftwindowsix: neither is bijective with $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ because this packet alone binds raw windows, stable folds, residual fibers, external phase projection, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ZeckendorfPhaseShell}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ packet is generated from a finite window-width row, raw binary-window row, stable Zeckendorf-window row, Fibonacci fold row, residual-fiber row, external golden-phase projection row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeckendorfPhaseShell}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8811.1 and Definition 8811.2 seed the finite Zeckendorf phase-shell certificate over stable-window, residual-fiber, golden-phase projection, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZeckendorfPhaseShell}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a kernel golden-phase object, quotient equality of histories, invariant measure, physical law, real-completion theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked stable-window admission, fold exactness, residual-fiber preservation, external phase transport, and structural exhaustion for BEDC.Derived.ZeckendorfPhaseShellUp.

Chapter 8812

A Concrete Naming Certificate for $\mathsf{SionMinimax}^{\uparrow}$

Following Sion's minimax theorem and Bishop–Bridges constructive analysis, the $\mathsf{SionMinimax}^{\uparrow}$ packet records finite compact-convex minimax data as NameCert rows rather than as a host saddle-point selection principle. It sits between the finite payoff-table route of chapter 7060, the KKM-facing convex route of chapter 7973, the primal-dual convex packet of chapter 321, and the compact interval source of chapter 6585. Its scalar readback is routed through chapter 38, chapter 291, chapter 301, and chapter 126.

Definition 8812.1 (SionMinimax finite compact-convex carrier)**.** *A* $\mathsf{SionMinimax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, Y, F, A, T, R, W, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact convex source row, read through displayed finite-window membership and finite convex-combination checks rather than through an ambient compactness theorem.*
2. *Y is the opposite convex source row, with the same finite-window membership and convex-combination discipline for the second variable.*
3. *F is the convex-concave payoff row, recording only displayed two-variable payoff comparisons and the finite slice checks needed by downstream minimax consumers.*
4. *A is the finite affine-spine ledger for the inspected convex combinations in X and Y.*
5. *T is the finite saddle-window approximant row, listing paired lower and upper comparison windows before any scalar value is sealed.*
6. *R is the dyadic tolerance ledger, recording the requested comparison precision through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *route.*
7. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the finite tolerance ledger into regular rational comparison data.*
8. *D is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that orders the finite inspection windows consumed by T, R, W.*
9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after the finite saddle-window, tolerance, rational-readback, and stream-schedule rows are displayed.*

10. *H records componentwise* hsame *transport for compact-convex source rows, payoff rows, affine-spine rows, saddle-window approximants, tolerance ledgers, readback rows, schedule rows, value seals, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay for the same finite compact-convex minimax route.*
12. *P is the* Pkg *provenance over* $\mathsf{SionMinimax}^{\uparrow}$, $\mathsf{FiniteGameMinimax}^{\uparrow}$, $\mathsf{KyFanMinimax}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ConvexCompactInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SionMinimax}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, F, A, T, R, W, D, Q, H, C, P, N$. It has no coordinate for host suprema or infima, choice-selected saddle points, quotient equality, full compactness, or ambient real completeness.

Theorem 8812.2 (SionMinimax NameCert obligations)**.** *For every accepted* $\mathsf{SionMinimax}^{\uparrow}$ *carrier*

$$S = (X, Y, F, A, T, R, W, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SionMinimax}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier comparison, saddle-window exactness, dyadic tolerance ledger readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A minimax equality read is admitted only after the compact-convex source rows, payoff row, affine-spine ledger, finite saddle-window approximants, tolerance ledger, rational readback, stream schedule, value seal, transport, replay, provenance, and local certificate are visible in the same packet.*

Proof. Project the accepted carrier of Definition 8812.1. The source rows X, Y, the payoff row F, and the affine-spine ledger A display the compact-convex minimax surface before any scalar comparison is read.

The finite approximation route is T, R, W, D. The row T supplies the lower and upper saddle windows, R records their dyadic tolerance budget, W exports regular rational readback, and D fixes the stream-window schedule that orders those finite reads.

The value seal Q is downstream of that route. Componentwise transport, replay, provenance, and local naming are then supplied by H, C, P, N.

Since no row stores host suprema, host infima, a selected saddle pair, quotient equality, full compactness, or ambient real completeness, the listed rows exhaust the local naming obligations. □

Theorem 8812.3 (SionMinimax finite saddle-window route)**.** *Every minimax equality consumer of an accepted* $\mathsf{SionMinimax}^{\uparrow}$ *packet must pass through the finite saddle-window row T before the* $\mathsf{Real}^{\uparrow}$ *seal Q is public. Thus the scalar equality exported by the packet is read only after the displayed compact-convex rows, convex-concave payoff row, affine-spine ledger, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{StreamName}^{\uparrow}$ *schedule have all been inspected.*

Proof. Use Definition 8812.2. It orders the accepted packet so that the source, payoff, and affine-spine rows are displayed before the finite saddle-window approximants are read.

The only path from those approximants to a scalar equality is through R, W, D, Q. The dyadic tolerance ledger R, regular rational readback W, and stream schedule D are all visible before the $\mathsf{Real}^{\uparrow}$ seal Q can serve a consumer.

Therefore a minimax equality read cannot skip directly to an ambient scalar principle. It factors through the finite saddle-window route carried by the same packet. □

Theorem 8812.4 (SionMinimax nonescape boundary)**.** *An accepted* $\mathsf{SionMinimax}^{\uparrow}$ *packet exports no host supremum, host infimum, choice-selected saddle point, quotient equality of points or functions, full compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle. Every public minimax read is confined to the displayed rows* $X, Y, F, A, T, R, W, D, Q, H, C, P, N$*.*

Proof. Inspect the carrier definition. Its coordinates are exactly $X, Y, F, A, T, R, W, D, Q, H, C, P, N$, and the classifier compares packets only through those rows.

The obligation theorem identifies the same rows as the complete local NameCert surface. The finite saddle-window theorem then forces every scalar equality read through T, R, W, D before the value seal Q.

Because none of these rows contains a host extremum operator, a selected saddle pair, a quotient-equality witness, full compactness data, or ambient real-completeness data, no consumer can extract such a principle from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SionMinimax}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SionMinimax}^{\uparrow}$ packet is generated from compact convex source rows, opposite convex source rows, convex-concave payoff rows, finite affine-spine ledgers, saddle-window approximants, dyadic tolerance ledgers, RegSeqRat readback rows, StreamName schedules, a RealUp value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SionMinimax}^{\uparrow}}$, seed)

Scope closed: Definition 8812.1, Definition 8812.2, Definition 8812.3, and Definition 8812.4 seed the finite compact-convex Sion minimax route over displayed source, payoff, saddle-window, tolerance, rational-readback, stream-schedule, RealUp seal, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SionMinimax}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host suprema or infima, choice-selected saddle points, quotient equality, full compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier comparison, saddle-window exactness, dyadic tolerance readback, RegSeqRat readback, StreamName scheduling, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.SionMinimaxUp.

Chapter 8813

A Concrete Naming Certificate for $\mathsf{MooreSmithCauchyNet}^{\uparrow}$

Following Moore, Smith, Kelley, and Bishop–Bridges, the $\mathsf{MooreSmithCauchyNet}^{\uparrow}$ packet records the standard directed Cauchy-net source as finite BEDC NameCert data. It sits after the directed substrate of chapter 1607, the Cauchy-net completion surface of chapter 3794, the Moore–Smith completion route of chapter 2190, and the diagonal handoff of chapter 6879. Its numerical readback is routed through chapter 126, chapter 301, chapter 291, and the terminal real-name seal of chapter 38. The packet names the source object consumed by L10 completion routes; it does not select a Moore–Smith limit, quotient nets, choose a subnet, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8813.1 (Moore–Smith Cauchy net carrier)**.** *A* $\mathsf{MooreSmithCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, N, T, W, R, Q, E, H, C, P, L)$$

with the following visible rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *substrate, including the finite directed vertices, comparison rows, and common upper-window witnesses used by the net route.*
2. *N is the displayed net-source row, recording the observed Moore–Smith values and the finite tail reads over D.*
3. *T is the Cauchy-tail ledger, listing the precision requests and the directed tail thresholds that answer them.*
4. *W is the finite window row, naming the directed windows inspected before any readback or completion-facing consumer receives the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback obtained from the displayed windows.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the Cauchy-tail comparisons and by the regular readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the directed substrate, net source, Cauchy-tail ledger, finite windows, readback, and dyadic tolerance rows have been exposed.*
8. *H records componentwise* hsame *transport for D, N, T, W, R, Q, E and the structural rows.*
9. *C records displayed* Cont *replay from a directed Cauchy-net read through the same tail, window, readback, tolerance, and Real-seal route.*

10. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCauchyNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected limits, quotient nets, selected subnets, countable-choice schedules, host real equality, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8813.2 (Moore–Smith Cauchy net NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithCauchyNet}^{\uparrow}$ *carrier* $M = (D, N, T, W, R, Q, E, H, C, P, L)$, *the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed substrate, net source, Cauchy-tail ledger, finite windows, regular readback, dyadic tolerance row, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-window exposure, requiring every completion-facing or readback-facing request to begin with D, N, W before the Cauchy-tail ledger is consumed;*
3. *Cauchy-tail stability, requiring T and Q to display the precision request, tail threshold, and dyadic comparison rows used by the same directed windows;*
4. $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *readback, requiring R to be read only after W, T, Q have fixed the finite Cauchy route;*
5. *Real-seal nonescape, requiring E to be downstream of the displayed directed, tail, window, regular, and dyadic rows rather than an ambient completeness theorem;*
6. *transport, replay, and provenance preservation, requiring H, C, P, L to preserve and name only the displayed source packet.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8813.1. The tuple exposes the directed substrate, net source, Cauchy-tail ledger, finite windows, regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Directed-window exposure reads D, N, W before any consumer reaches a numerical or completion-facing boundary. The Cauchy-tail row T then binds each precision request to a displayed tail threshold on that same directed surface.

The readback step uses Q and R together. The dyadic ledger records the tolerance comparisons, while the regular rational row reads only the windows already admitted by W and stabilized by T.

Finally E is a terminal seal on the displayed route, and H, C, P, L transport, replay, source, and name only the listed coordinates. Since selected limits, quotient nets, selected subnets, countable choice, host equality, and ambient completeness have no carrier coordinate, none of them can mediate the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNet}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNet}^{\uparrow}}$, seed)

Scope closed: Definition 8813.1 and Definition 8813.2 seed the finite Moore–Smith Cauchy-net source packet over DirectedSet, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected Moore–Smith limits, quotient nets, selected subnets, countable choice, host real equality, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window exposure, Cauchy-tail stability, RegSeqRat and DyadicRatCore readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MooreSmithCauchyNetUp.

Constructive story: A MooreSmithCauchyNet$^{\uparrow}$ packet is generated from a DirectedSet substrate, a Moore–Smith net source row, Cauchy-tail thresholds, finite directed windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8814

A Concrete Naming Certificate for $\mathsf{BorsukUlamAntipodal}^{\uparrow}$

Following Borsuk, Ulam, and Tucker, the $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ packet isolates the finite antipodal witness ledger used by the Borsuk–Ulam obstruction route. It sits beside the obstruction packet $\mathsf{BorsukUlam}^{\uparrow}$ of chapter 5828 and the combinatorial antipodal labeling packet $\mathsf{TuckerLemma}^{\uparrow}$ of chapter 6975. The packet names the witness-ledger surface only: it does not strengthen $\mathsf{BorsukUlam}^{\uparrow}$, select antipodal points, identify quotient spheres, or import a host topological theorem.

Definition 8814.1 (Borsuk-Ulam antipodal witness-ledger carrier)**.** *A* $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, A, T, L, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the supplied sphere or boundary-source row consumed from the compact metric and* $\mathsf{S}^{1\uparrow}$ *surfaces already named by* $\mathsf{BorsukUlam}^{\uparrow}$*.*
2. *A is the antipodal-pair ledger row, storing paired source histories and the refusal of quotient-sphere identification.*
3. *T is the Tucker-style finite labeling row, read through the sibling* $\mathsf{TuckerLemma}^{\uparrow}$ *route when a combinatorial witness is used.*
4. *L is the local comparison ledger that records which antipodal read is being tested by the downstream obstruction consumer.*
5. *O is the obstruction-facing handoff row, pointing back to the* $\mathsf{BorsukUlam}^{\uparrow}$ *obstruction packet without adding a stronger theorem to it.*
6. *H records componentwise* hsame *transport for source, antipodal ledger, Tucker row, local comparison, obstruction handoff, provenance, and local naming.*
7. *C records displayed* Cont *replay from the source and antipodal rows through the finite witness ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{BorsukUlamAntipodal}^{\uparrow}$*,* $\mathsf{BorsukUlam}^{\uparrow}$*,* $\mathsf{TuckerLemma}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BorsukUlamAntipodal}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected antipodal points, quotient equality of sphere points, unrestricted degree theory, choice, ambient compactness, or a completed topological Borsuk–Ulam theorem.

Theorem 8814.2 (Borsuk-Ulam antipodal witness-ledger NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ *carrier* $W = (S, A, T, L, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlamAntipodal}^{\uparrow}}$ *surface consists exactly of source admission, antipodal-pair ledger exposure, Tucker-style finite labeling handoff, local comparison readback, obstruction-facing handoff to* $\mathsf{BorsukUlam}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every downstream obstruction consumer reads the finite witness ledger through*

$$S, A, T \longmapsto L, O \longmapsto H, C, P, N$$

and receives no point-selection principle or quotient-sphere equality from that read.

Lean4 (statement) `BEDC.Derived.BorsukUlamAntipodalUp`

Proof. Project the carrier of Definition 8814.1. The rows S, A, T are the only source-side witness data: they display the boundary source, the antipodal ledger, and the Tucker-style finite labeling route.

The ledger then passes through L and O. The row L names the local antipodal comparison being inspected, while O records that the obstruction-facing consumer must return to $\mathsf{BorsukUlam}^{\uparrow}$ rather than reading a stronger theorem from this packet.

Finally H, C, P, N transport, replay, source, and name the same finite witness ledger. Since the excluded objects are not carrier coordinates, the packet can serve as a reusable witness ledger without asserting selected antipodal points, quotient equality, or host topological completeness. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ read first displays sphere or boundary source rows, an antipodal-pair ledger, and Tucker-style finite labeling data before any obstruction-facing handoff to $\mathsf{BorsukUlam}^{\uparrow}$ is consumed.

Independence witness: borsukulam, tuckerlemma, compactmetric, s1: $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ is not bijective to any listed sibling because it binds the antipodal witness ledger, Tucker-style finite labeling handoff, local comparison row, and obstruction-facing handoff as one NameCert packet, while each sibling supplies only an obstruction theorem, combinatorial labeling theorem, source carrier, or boundary model.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlamAntipodal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BorsukUlamAntipodal}^{\uparrow}$ packet is generated from sphere or boundary source rows, an antipodal-pair ledger, Tucker-style finite labeling rows, a local comparison ledger, an obstruction-facing handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BorsukUlamAntipodal}^{\uparrow}}$, seed)

Scope closed: Definition 8814.1 and Definition 8814.2 seed the finite antipodal witness-ledger surface between $\mathsf{BorsukUlam}^{\uparrow}$ and $\mathsf{TuckerLemma}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BorsukUlamAntipodal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected antipodal points, quotient-sphere equality, host topological Borsuk–Ulam, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, Tucker-style finite labeling handoff, local comparison readback, obstruction-facing handoff, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamAntipodalUp.

Chapter 8815

A Concrete Naming Certificate for $\mathsf{MetricFanTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MetricFanTheorem}^{\uparrow}$ packet records the finite-means form of the fan theorem as it is consumed by metric continuity arguments. It sits after the finite spread route of $\mathsf{FanTheorem}^{\uparrow}$ and $\mathsf{FanSpace}^{\uparrow}$ in chapter 1962, the binary-prefix boundary supplied by $\mathsf{CantorSpace}^{\uparrow}$, the metric source of chapter 57, the pointwise continuity interface of chapter 130, the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369, and the terminal real-name seal of chapter 38. The chapter names a finite branch-radius certificate for compact fan arguments; it does not assert open-cover compactness, host uniform continuity, selected infinite branches, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8815.1 (MetricFanTheorem carrier)**.** *A* $\mathsf{MetricFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, B, M, C, R, U, E, H, K, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row, recording a displayed depth that covers the finite branching source.*
2. *B is the* $\mathsf{FanSpace}^{\uparrow}$ *or* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row whose inspected binary branches are bounded by F.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing only the metric comparisons needed to read branch radii.*
4. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise continuity row, assigning each inspected prefix a displayed local radius and target tolerance response.*
5. *R is the finite radius ledger, listing the branch radii that survive the fan-depth inspection.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained by folding the finite radius ledger into one displayed modulus depth.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only for the finite metric and modulus observations already exposed.*
8. *H records componentwise* hsame *transport for fan, prefix, metric, continuity, radius, modulus, seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a metric fan request through $F, B, M, C, R, U, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{MetricFanTheorem}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{FanSpace}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricFanTheorem}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite metric fan history. Its classifier compares accepted packets only through the displayed fan depth, inspected prefix rows, metric observations, pointwise continuity rows, finite radius ledger, uniform-modulus handoff, Real seal, transport, replay, provenance, and local name.

Theorem 8815.2 (MetricFanTheorem NameCert obligations)**.** *For every accepted* $\mathsf{MetricFanTheorem}^{\uparrow}$ *carrier* $T = (F, B, M, C, R, U, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricFanTheorem}^{\uparrow}}$ *surface consists exactly of finite fan-depth admission, prefix-row exposure, metric comparison exposure, pointwise continuity exposure, finite radius ledgering, uniform-modulus handoff, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 8815.1. The rows F and B are the only fan-side rows, so fan admission and branch inspection are confined to the displayed finite depth and prefix surface.

Read the analytic rows next. The metric row M supplies the branch-distance comparisons, C supplies pointwise continuity data at the inspected prefixes, and R records the finite list of radii that remain after the fan-depth inspection.

Finally use U, E, H, K, P, N. The uniform-modulus handoff, Real seal, componentwise transport, replay, provenance, and local name preserve the same finite packet. Since the carrier has no further coordinate, open-cover compactness, host uniform continuity, selected infinite branches, quotient stream equality, countable choice, and ambient Real completeness cannot become local NameCert obligations. □

Theorem 8815.3 (MetricFanTheorem uniform-modulus route)**.** *Every downstream consumer of an accepted* $\mathsf{MetricFanTheorem}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{FanTheorem}^{\uparrow},\ \mathsf{FanSpace}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto R \longmapsto \mathsf{UniformModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Consequently a consumer may read one displayed fan depth, the inspected prefix rows below that depth, the metric branch-radius comparisons, and the resulting uniform-modulus handoff. It receives no open-cover compactness theorem, no host uniform-continuity theorem, no selected infinite branch, no quotient stream equality, no countable-choice schedule, and no ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Start with the fan rows. The row F supplies the finite bound, and B exposes only the prefix rows inspected below that bound.

Use the metric and continuity rows. The row M records the source-distance observations needed for branch radii, while C records the pointwise continuity responses for those displayed prefixes. The ledger R is therefore a finite list obtained inside the inspected fan surface.

Route the finite ledger through U and E, then preserve it through H, K, P, N. Since every downstream handoff factors through these rows, the public route is the displayed uniform-modulus extraction route and not an ambient compactness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricFanTheorem}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8815.1, Definition 8815.2, and Definition 8815.3 seed the finite metric fan route from fan depth and prefix rows to metric branch radii, pointwise continuity rows, a uniform-modulus handoff, Real seal, transport, replay, provenance, and local naming.

Constructive story: A MetricFanTheorem$^{\uparrow}$ packet is generated from a finite FanTheorem depth row, FanSpace or CantorSpace prefix rows, Metric comparisons, ContinuousMap pointwise continuity rows, a finite radius ledger, a UniformModulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetricFanTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host uniform continuity, selected infinite branches, quotient stream equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked fan-depth admission, prefix-row exposure, metric comparison exposure, pointwise continuity exposure, finite radius ledgering, uniform-modulus handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricFanTheoremUp.

Chapter 8816

A Concrete Naming Certificate for $\mathsf{DyadicDistance}^{\uparrow}$

Following Bishop and Bridges, read dyadic distance as the finite absolute-difference certificate needed before regular Cauchy real names. The $\mathsf{DyadicDistance}^{\uparrow}$ packet sits after the dyadic rational source of chapter 291, the absolute-value normalization of chapter 293, and the metric handoff of chapter 2001. It is consumed by the regular Cauchy row discipline of chapter 301 and the terminal real-name seal of chapter 38. The packet names only finite endpoint, scale, sign, absolute-value, transport, replay, provenance, and local certificate rows; it does not import a host real distance, quotient rational equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8816.1 (Dyadic distance carrier)**.** *A* $\mathsf{DyadicDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, R, E, S, A, Z, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint histories.*
2. *E is the common positive unary scale row refining the endpoint denominators.*
3. *S is the signed common-scale difference row, including the transported mantissas.*
4. *A is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *absolute-value normalization row for S.*
5. *Z is the zero-distance and triangle ledger, storing the equal-endpoint row and the common-scale comparison rows used by later tolerance consumers.*
6. *H records componentwise* hsame *transport for endpoints, scale, signed difference, absolute value, ledger, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from an endpoint-pair request through the common-scale and absolute-value rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicDistance}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets by equality of the displayed distance witnesses after common-scale refinement. It has no coordinate for a quotient rational representative, a host real metric, a selected real endpoint, trichotomy, or a completion principle.

Lean4 (statement) `BEDC.Derived.DyadicDistanceUp`

Theorem 8816.2 (Dyadic distance NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDistance}^{\uparrow}$ *carrier* $D = (L, R, E, S, A, Z, H, C, P, N)$*, the local naming surface has exactly five mathematical obligation rows:*

1. *carrier admission for the pair of* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints plus the absolute common-scale witness;*
2. *classifier comparison by equality of distance witnesses under common-scale refinement;*
3. *stability under dyadic equality, componentwise* hsame *transport, and displayed* Cont *replay;*
4. *exactness for zero-distance exposure and the common-scale triangle rows recorded in* Z*;*
5. *ledger visibility for hidden scale choices and sign-normalization data.*

The structural rows H, C, P, N *transport, replay, source, and name these obligations without adding host real distance, quotient equality, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8816.1. The endpoint rows L, R, the common scale row E, the signed difference row S, the absolute-value row A, and the ledger row Z are the only mathematical coordinates available to a distance consumer.

The classifier row is read through the common-scale witness. Refining two accepted distance packets to a further displayed scale transports the same endpoint and mantissa rows through $\mathsf{DyadicRatCore}^{\uparrow}$ and compares the resulting absolute witnesses through $\mathsf{DyadicAbsoluteValue}^{\uparrow}$.

Stability is therefore componentwise. The H row transports endpoints, scale, signed-difference, absolute-value, and ledger data, while C replays the same finite route for the transported packet.

Exactness is confined to Z. The zero-distance row is the equal-endpoint common-scale case, and the triangle row is the finite comparison of three displayed common-scale absolute differences. Since no host real, quotient, trichotomy, or completion row is present in the carrier, the five listed obligations exhaust the naming surface. □

Theorem 8816.3 (Dyadic distance triangle envelope)**.** *Let* D_{01}*,* D_{12}*, and* D_{02} *be accepted* $\mathsf{DyadicDistance}^{\uparrow}$ *packets for three accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint histories. If their common-scale rows are refined to one displayed positive unary scale and their sign-normalization ledgers are transported to that scale, then the triangle envelope is read entirely from the three absolute common-scale witnesses and the ledger row* Z*. The read exposes no host metric inequality, quotient rational equality, selected real endpoint, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Refine the three packet scales to one displayed positive unary scale. The $\mathsf{DyadicRatCore}^{\uparrow}$ classifier supplies the endpoint transports, and the $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ rows normalize the three signed common-scale differences at that same scale.

Read the envelope from the ledger row Z of each accepted packet. These rows contain the zero-distance and common-scale triangle data that later $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers need for finite tolerance comparison.

The conclusion is a boundary statement about available rows. Because the carrier of Definition 8816.1 contains only endpoint, scale, signed-difference, absolute-value, ledger, transport, replay, provenance, and local naming data, the triangle envelope cannot be reinterpreted as a host metric theorem or as any completed real-distance principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDistance}^{\uparrow}}$**.**

Constructive story: A DyadicDistance$^{\uparrow}$ packet is generated from two DyadicRatCore endpoint rows, a common positive unary scale, a signed common-scale difference row, a DyadicAbs normalization row, a zero-distance and triangle ledger, componentwise transport, displayed replay, provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicDistance}^{\uparrow}}$, seed)

Scope closed: Definition 8816.1, Definition 8816.2, and Definition 8816.3 seed the finite dyadic distance substrate consumed by regular Cauchy and real-name tolerance rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, metric-space completion, quotient rational equality, trichotomy, or any host real-distance theorem.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, zero-distance exactness, triangle-envelope exactness, hidden-scale ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.DyadicDistanceUp.

Chapter 8817

A Concrete Naming Certificate for $\mathsf{BoundedDyadicVariation}^{\uparrow}$

Following Jordan bounded variation and Bishop-style finite partition reasoning, the $\mathsf{BoundedDyadicVariation}^{\uparrow}$ packet records bounded variation evidence only over displayed dyadic partitions. It sits between the finite variation surface of chapter 398, the partition cells of chapter 1863, the regular rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The packet gives regulated-function, Riemann, and Darboux consumers a finite dyadic variation ledger without importing measure completion, selected extrema, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8817.1 (Bounded dyadic variation carrier)**.** *A* $\mathsf{BoundedDyadicVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval source row, naming the inspected compact or located interval only through displayed endpoint data.*
2. *M is the dyadic partition mesh row, recording a finite ordered* $\mathsf{DyadicPartition}^{\uparrow}$ *mesh together with its adjacent cell and common-refinement data.*
3. *V is the finite variation-budget row, recording the dyadic edge sums available to the packet.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare endpoint differences and edge contributions on the displayed mesh.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to endpoint names and finite variation sums.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row, available only after I, M, V, D, R have been displayed.*
7. *H records componentwise* hsame *transport for interval sources, dyadic meshes, variation budgets, tolerance rows, readback rows, Real endpoint seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an interval request through the dyadic mesh, finite variation budget, dyadic tolerance rows, regular readback, and endpoint seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedDyadicVariation}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedDyadicVariation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, V, D, R, E, H, C, P, N$. It has no coordinate for measure completion, selected extrema, host function equality, hidden partition families, variation measures, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8817.2 (Bounded dyadic variation NameCert obligations)**.** *For every accepted* $\mathsf{BoundedDyadicVariation}^{\uparrow}$ *carrier*

$$B = (I, M, V, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedDyadicVariation}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows; dyadic partition exposure through M; finite variation-budget stability through V and its common-refinement reads; endpoint-difference tolerance through D; endpoint-readback discipline through R before E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

The certificate refuses measure completion, selected extrema, host equality of functions, hidden dyadic mesh families, a supremum over all partitions, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8817.3 (Bounded dyadic variation finite partition handoff)**.** *Every regulated-function, Riemann, or Darboux consumer of an accepted* $\mathsf{BoundedDyadicVariation}^{\uparrow}$ *packet must first read the displayed dyadic partition mesh M, the finite variation budget V, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before consuming the* $\mathsf{Real}^{\uparrow}$ *endpoint seal E.*

Thus the handoff route is finite:

$$\mathsf{DyadicPartition}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

It exports only displayed mesh cells, finite edge sums, dyadic tolerance rows, regular readback, endpoint seals, transport, replay, provenance, and local naming.

Falsifiable prediction: A $\mathsf{BoundedDyadicVariation}^{\uparrow}$ consumer that reaches a Real-facing regulated-function or integration conclusion without reading the dyadic mesh, finite variation budget, dyadic tolerance row, and RegSeqRat readback refutes the carrier surface.

Independence witness: boundedvariation, dyadicpartition, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{BoundedDyadicVariation}^{\uparrow}$ because this packet jointly carries an interval source, a displayed dyadic mesh, a finite variation budget, dyadic tolerance rows, regular readback, Real endpoint sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedDyadicVariation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedDyadicVariation}^{\uparrow}$ packet is generated from an interval source row, a finite dyadic partition mesh, a finite variation-budget row, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedDyadicVariation}^{\uparrow}}$, seed)

Scope closed: Definition 8817.1, Definition 8817.2, and Definition 8817.3 seed the finite dyadic variation interface from displayed dyadic partitions and variation budgets to regulated-function and integration consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedDyadicVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, selected extrema, host equality of functions, hidden partition families, quotient function spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic partition exposure, variation-budget stability, endpoint tolerance, regular readback, Real endpoint sealing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedDyadicVariationUp.

Chapter 8818

A Concrete Naming Certificate for $\mathsf{ConstructiveRealApartness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveRealApartness}^{\uparrow}$ packet records strict real separation as finite positive witness data rather than as a trichotomy principle. It sits over the Real carrier of chapter 38, the regular rational rows of chapter 301, the StreamName windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the positive real boundary of chapter 1587. Its role is to make the constructive constraint of section 9467.12 usable by later order and completion consumers without importing quotient real equality, host equality, selected limits, countable choice, trichotomy, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 8818.1 (ConstructiveRealApartness carrier)**.** *A* $\mathsf{ConstructiveRealApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, R_X, R_Y, S_X, S_Y, D, \delta, W, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Real}^{\uparrow}$ *seal rows being separated.*
2. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows used to inspect the two real names.*
3. *S_X and S_Y are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows from which the readbacks are taken.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger that records the finite dyadic comparison scale.*
5. *δ is the* $\mathsf{PositiveReal}^{\uparrow}$ *lower-bound witness for the displayed separation.*
6. *W is the finite apartness window asserting that the rational readbacks at the displayed scale are separated by the positive lower bound named by δ.*
7. *H records componentwise* hsame *transport for the real seals, regular readbacks, stream windows, dyadic ledger, positive witness, apartness window, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay that opens the two StreamName windows, reads the RegSeqRat rows, checks the dyadic comparison ledger, and hands the positive witness to the Real-facing apartness surface.*
9. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, host real equality, a selected representative limit, countable choice, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8818.2 (ConstructiveRealApartness NameCert obligations)**.** *For every accepted* ConstructiveRealApartne *carrier*

$$A = (X, Y, R_X, R_Y, S_X, S_Y, D, \delta, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRealApartness}^{\uparrow}}$ *surface consists of carrier admission for the displayed rows, paired RealUp seal exposure through* X, Y*, regular readback exposure through* R_X, R_Y*, finite StreamName window exposure through* S_X, S_Y*, dyadic comparison exactness through* D*, positive lower-bound exposure through* δ*, apartness-window readback through* W*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8818.1. The Real-facing rows are exactly X, Y, and the only inspection rows are the regular readbacks, stream windows, dyadic comparison ledger, positive witness, and apartness window displayed in the tuple.

The certificate reads separation constructively. The witness δ is a positive lower-bound row, while W ties that row to the finite rational readbacks obtained from S_X, S_Y through R_X, R_Y and checked against D.

The transport, replay, provenance, and local name rows preserve the same finite route. Since no further carrier coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 8818.3 (ConstructiveRealApartness finite readback)**.** *Every accepted* $\mathsf{ConstructiveRealApartness}^{\uparrow}$ *read is witnessed by the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{PositiveReal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The two Real-facing rows become apart only through the displayed StreamName windows, regular rational readbacks, dyadic comparison ledger, positive lower-bound row, and apartness window.

Proof. Use the NameCert obligations of Definition 8818.2. They force each accepted read to expose S_X, S_Y before R_X, R_Y, then the dyadic ledger D, the positive witness δ, and the apartness window W.

Only after those rows have been displayed can the Real-facing rows X, Y be read as apart. The continuation row C records the same order, and the package row P adds provenance rather than a new comparison principle. □

Theorem 8818.4 (ConstructiveRealApartness non-trichotomy boundary)**.** *An accepted* ConstructiveRealApartness *carrier exports a positive finite apartness witness and does not export quotient real equality, host real equality, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. In particular, the packet cannot be read as a decision procedure for arbitrary comparison of the two RealUp rows.*

Proof. Project the carrier. Its content rows are the two RealUp seals, regular readbacks, finite StreamName windows, dyadic ledger, positive witness, and apartness window.

The finite readback theorem Definition 8818.3 identifies the only route from these rows to an apartness claim. That route supplies a displayed positive witness, not a three-way case split and not a host equality test.

Since the tuple contains no row for quotient equality, selected representatives, choice, trichotomy, or completeness, no accepted consumer can recover those principles from the packet. □

Falsifiable prediction: A packet that claims ConstructiveRealApartnessUp while omitting either the positive lower-bound row or the finite apartness window is not an accepted carrier.

Independence witness: real, regseqrat, streamname, dyadicratcore, positivereal: each sibling supplies one surface of the readback route, while ConstructiveRealApartnessUp binds two RealUp seals, two regular readbacks, two StreamName windows, one dyadic comparison ledger, one positive witness, transport, replay, provenance, and a local NameCert row in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealApartness}\uparrow}$.

Constructive story: A $\mathsf{ConstructiveRealApartness}^{\uparrow}$ packet is generated from two RealUp seal rows, paired RegSeqRat readbacks, paired StreamName finite windows, a DyadicRatCore comparison ledger, a PositiveReal lower-bound witness, an apartness window, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRealApartness}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8818.1, Definition 8818.2, Definition 8818.3, and Definition 8818.4 seed the finite constructive apartness surface for RealUp rows through regular readback, StreamName windows, dyadic comparison, and positive witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRealApartness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, host equality, selected limits, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired readback exposure, finite apartness-window exactness, positive-witness stability, route replay, provenance, and non-trichotomy nonescape for the same packet.

Chapter 8819

A Concrete Naming Certificate for RichardsonExtrapolation$^{\uparrow}$

Following Richardson's extrapolation method for cancelling the leading discretization term, the RichardsonExtrapolation$^{\uparrow}$ packet records a finite acceleration surface for two or more displayed approximation rows before any Real or completion-facing consumer reads the accelerated value. It sits after the dyadic endpoint arithmetic of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the regular Cauchy completion route of chapter 1639. The certificate names one displayed cancellation stencil; it does not assert an asymptotic expansion theorem, quotient equality of real names, selected convergence rates, countable choice, or ambient completeness.

Definition 8819.1 (Richardson extrapolation carrier)**.** *A* RichardsonExtrapolation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, A, O, K, Q, B, E, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *scale and endpoint ledger naming the refinement ratio, powers of the scale, and dyadic denominator comparisons.*
2. *S is the* StreamName$^{\uparrow}$ *window row selecting the finite approximation depths inspected by the extrapolation stencil.*
3. *A is the source approximation row over the displayed depths, with no hidden ambient sequence.*
4. *O is the finite order row naming the leading error order that the stencil is allowed to cancel.*
5. *K is the dyadic coefficient row computing the Richardson weights from D and O.*
6. *Q is the denominator-apartness and fallback ledger for the coefficient computation.*
7. *B is the accelerated readback row formed from A, K, Q.*
8. *E is the* RegSeqRat$^{\uparrow}$ *readback row reached from the displayed accelerated values.*
9. *U is the downstream* Real$^{\uparrow}$ *and* BishopRegularCauchyCompletion$^{\uparrow}$ *seal row exposed only after D, S, A, O, K, Q, B, E have been displayed.*
10. *H records componentwise* hsame *transport for all visible extrapolation rows.*
11. *C records displayed* Cont *replay from scale and window reads to coefficient construction, accelerated readback, regular rational readback, and terminal sealing.*

12. *P is the* Pkg *provenance over* $\mathsf{RichardsonExtrapolation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RichardsonExtrapolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, A, O, K, Q, B, E, U, H, C, P, N$. It has no coordinate for host asymptotic equality, quotient real equality, a selected rate theorem, countable-choice extraction of a subsequence, or ambient completeness.

Theorem 8819.2 (Richardson extrapolation NameCert obligations)**.** *For an accepted* $\mathsf{RichardsonExtrapolation}^{\uparrow}$ *carrier $R = (D, S, A, O, K, Q, B, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RichardsonExtrapolation}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic scale ledger exposure through D, finite approximation-window exposure through S and A, displayed leading-order admission through O, coefficient construction through K, denominator-apartness and fallback admission through Q, accelerated readback through B,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through E,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *sealing through U, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 8819.1. The tuple lists exactly the dyadic scale data, finite stream window, approximation rows, leading-order row, coefficient ledger, denominator guard, accelerated readback, regular rational readback, terminal seal, transport, replay, provenance, and local name.

The cancellation step is finite. The coefficient row K is computed only from the displayed dyadic scale row D and order row O, and the denominator guard Q is read before any accelerated value in B is exported.

The analytic handoff is downstream of that finite stencil. The stream row S supplies the inspected depths, A supplies the source approximations at those depths, B supplies the weighted readback, and E converts the readback into the regular rational route before U exposes any Real or completion-facing seal.

Finally H, C, P, N preserve the same visible route under transport, replay, provenance, and local naming. Since no carrier coordinate stores host asymptotic equality, quotient real equality, a selected rate theorem, countable-choice extraction, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8819.3 (Richardson extrapolation RegSeqRat and Real seal handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RichardsonExtrapolation}^{\uparrow}$ *packet must pass through the finite route*

$$D, S, A, O, K, Q, B, E, U, H, C, P, N.$$

The consumer receives the displayed dyadic scale ledger, finite approximation window, leading-order row, coefficient construction, denominator guard, accelerated readback, regular rational readback, terminal Real and completion seal, componentwise transport, continuation replay, provenance, and local naming. It receives no host asymptotic expansion, quotient equality of real names, hidden convergence rate, countable-choice schedule, or ambient completeness theorem.

Proof. Begin with the accepted carrier of Definition 8819.1. A downstream consumer has access only to the displayed rows $D, S, A, O, K, Q, B, E, U, H, C, P, N$.

Read the finite numerical stencil first. The row D fixes the dyadic refinement scale, S chooses the inspected approximation depths, A supplies the corresponding approximation values, and O names the leading order whose displayed term is to be cancelled.

The coefficient and guard rows are then mandatory. The row K stores the Richardson weights computed from D and O, while Q records the denominator-apartness or fallback evidence required before the accelerated value B can be used.

Finally the regular rational and terminal rows E and U expose the same accelerated readback to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ consumers. The transport, replay, provenance, and local name rows H, C, P, N add no host asymptotic, quotient, choice, or completeness coordinate, so every consumer-facing read factors through the displayed route. □

Falsifiable prediction: Every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ read from a $\mathsf{RichardsonExtrapolation}^{\uparrow}$ packet factors through displayed dyadic scale rows, finite StreamName windows, source approximation rows, a leading-order row, coefficient construction, denominator-apartness or fallback evidence, accelerated readback, regular rational readback, terminal seal, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-cauchy-completion: $\mathsf{RichardsonExtrapolation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a refinement-scale ledger, finite approximation window, leading-order cancellation row, coefficient and denominator guard rows, accelerated readback, regular rational readback, Real and completion seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RichardsonExtrapolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RichardsonExtrapolation}^{\uparrow}$ packet is generated from dyadic refinement-scale data, finite StreamName windows, source approximation rows, a displayed leading-order row, Richardson coefficient construction, denominator-apartness or fallback evidence, accelerated readback, RegSeqRat readback, Real and Bishop regular Cauchy completion sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RichardsonExtrapolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8819.1, Definition 8819.2, and Definition 8819.3 seed the finite Richardson extrapolation packet and its route to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RichardsonExtrapolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, a standard bridge, host asymptotic expansion theorems, quotient real equality, selected convergence rates, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, coefficient guard exactness, RegSeqRat readback handoff, Real and completion seal nonescape, and consumer-boundary proofs for BEDC.Derived.RichardsonExtrapolationUp.

Chapter 8820

A Concrete Naming Certificate for $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$

The $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$ packet turns the marker-ledger discipline of chapter 9236 into finite NameCert data. It reads the visions index as a source-label surface, points to concrete-instance targets through displayed chapter labels, records the marker ledger, replays the audit readback, and carries provenance for the route that connects inspiration-only vision material to mature concrete packets. It is a finite audit object for the existing marker surface, not a replacement for the target chapters it names.

The marker audit route consumes the kernel acceptance witness of chapter 549 and the source-label discipline of chapter 9364 before it records a concrete target label.

Definition 8820.1 (Finite vision marker audit carrier)**.** *A* $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (S, T, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-label row naming the vision chapter or vision index site being audited.*
2. *T is the concrete-target row naming the concrete-instance chapter label claimed by the marker.*
3. *L is the finite marker-ledger row recording the source label, target label, marker text, and local position.*
4. *A is the audit-readback row that checks source-label agreement, target-label agreement, and marker-ledger agreement in one finite pass.*
5. *H records componentwise* hsame *transport for the source label, target label, ledger, audit readback, continuation, provenance, and local naming rows.*
6. *C records the displayed* Cont *replay that reads S, then T, then L, and finally the readback row A.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$*, the visions index, the concrete-instance target chapter,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteVisionMarkerAudit}^{\uparrow}}$ *row.*

The carrier stores only finite marker data and route readback. It does not import the mathematical content of the target chapter, rewrite a vision file, or certify any theorem beyond the marker route itself.

Theorem 8820.2 (Finite vision marker audit NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$ *carrier* $V = (S, T, L, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteVisionMarkerAudit}^{\uparrow}}$ *surface consists of source-label exposure through* S*, concrete-target exposure through* T*, marker-ledger exposure through* L*, audit-readback exactness through* A*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8820.1. The source label, target label, marker ledger, audit readback, transport, replay, provenance, and name are the only rows available to an accepted consumer.

The readback row A is downstream of S, T, L and checks exactly those labels and the marker ledger. The continuation row C fixes the same order, while H transports the rows componentwise.

The package row P names the visions index and the concrete target without importing the target chapter's theorem content. Therefore the listed rows exhaust the local NameCert surface. □

Theorem 8820.3 (Finite vision marker audit readback failure)**.** *An accepted* $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$ *carrier rejects any marker read whose source label or concrete-target label differs from the displayed rows* S *and* T*. The marker ledger row* L *and audit-readback row* A *must agree on the same pair of labels before the route can be consumed.*

Proof. Use the NameCert obligations of Definition 8820.2. They make S, T, L, A visible and force the audit readback to occur only after the source and target rows are displayed.

If a candidate read changes the source label, it no longer transports componentwise through the S row. If it changes the target label, it no longer transports componentwise through the T row.

The ledger and readback rows therefore agree only for the displayed pair. A mismatched source or target label leaves the accepted carrier surface. □

Theorem 8820.4 (Finite vision marker audit TasteGate obligations)**.** *The TasteGate surface for* $\mathsf{FiniteVisionMarkerAudit}^{\uparrow}$ *requires Lean to provide local* $\mathsf{Nontrivial}$ *and* $\mathsf{FieldFaithful}$ *instances over the same carrier rows* S, T, L, A, H, C, P, N*. Nontriviality separates a marker with matching source and target labels from a candidate with a mismatched label, and field faithfulness preserves the source-label, target-label, ledger, and audit-readback order.*

Proof. The carrier of Definition 8820.1 supplies the only rows on which the taste gate may act. A nontrivial boundary is visible because a matched marker read and a mismatched marker read differ at S or T before any audit readback is accepted.

Field faithfulness is the requirement that every faithful read preserve S, T, L, A and the structural rows H, C, P, N in the displayed order. Dropping the ledger or replacing the audit readback by a detached target theorem would create data outside the carrier. □

Falsifiable prediction: A FiniteVisionMarkerAuditUp packet whose source or target labels differ between the marker ledger and audit-readback rows fails marker-readback.

Independence witness: vision-coverage-index, vision-coverage-ledger, affine-filter-hull, vision-ledger-route: these siblings expose coverage, ledger, affine-filter, or routing rows separately, while FiniteVisionMarkerAuditUp binds source label, concrete target, marker ledger, audit readback, transport, replay, provenance, and local naming in one finite marker audit carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVisionMarkerAudit}^{\uparrow}}$**.**

Constructive story: A FiniteVisionMarkerAudit$^{\uparrow}$ packet is generated from a source-label row, a concrete-target row, a finite marker-ledger row, an audit-readback row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteVisionMarkerAudit}^{\uparrow}}$, seed)

Scope closed: Definition 8820.1, Definition 8820.2, Definition 8820.3, and Definition 8820.4 seed the finite marker-audit route for the visions index and its concrete-instance markers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteVisionMarkerAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not certify target-chapter mathematics, rewrite vision prose, close Lean verification, or provide a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-label exactness, concrete-target exactness, marker-ledger readback, route stability, provenance, TasteGate nontriviality, and field faithfulness for the same finite marker audit packet.

Chapter 8821

A Concrete Naming Certificate for $\mathsf{FluxOperatorGeometry}^{\uparrow}$

The $\mathsf{FluxOperatorGeometry}^{\uparrow}$ packet realizes the window-six flux-operator geometry branch of Definition 9346.48 as finite NameCert data. It sits beside the stable Zeckendorf phase-shell surface of chapter 8811, the finite golden-mean window of chapter 8802, and the static boundary-flux route of chapter 8769. The packet names stable addresses, transition operators, phase gates, boundary escape flux, and golden-shell projection as one finite operator-geometry surface; it does not assert a hidden metric geometry, invariant bilinear form, invariant multiplication, physical impedance law, or curvature-scalar readout.

Definition 8821.1 (Flux operator geometry carrier)**.** *A* $\mathsf{FluxOperatorGeometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, T, G, B, S, O, H, C, P, N)$$

with the following displayed rows:

1. *A is the stable Zeckendorf address row, read through the visible window-six address surface of* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$*.*
2. *T is the finite transition-operator table on those displayed addresses.*
3. *G is the phase-gate row, recording which transition reads preserve, flip, or refuse the visible phase shell.*
4. *B is the boundary escape-flux row, read through the static boundary-flux sibling where the boundary cut is visible.*
5. *S is the golden-shell projection row, read through* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$ *and* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$ *rather than through a hidden inner product.*
6. *O is the operator-algebra refusal row, recording the absence of any displayed invariant bilinear form or invariant multiplication in the accepted packet.*
7. *H records componentwise* hsame *transport for address, transition, phase-gate, boundary-flux, projection, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from stable address reads to transition operators, phase gates, escape-flux reads, golden-shell projection, and refusal checks.*
9. *P is the* Pkg *provenance over* $\mathsf{FluxOperatorGeometry}^{\uparrow}$*,* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$*,* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$*,* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FluxOperatorGeometry}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, T, G, B, S, O, H, C, P, N$. It has no coordinate for a hidden metric tensor, invariant symplectic form, bracket, commutative product, physical coupling constant, or curvature scalar.

Theorem 8821.2 (Flux operator geometry NameCert obligations)**.** *For every accepted* $\mathsf{FluxOperatorGeometry}^{\uparrow}$ *carrier $F = (A, T, G, B, S, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FluxOperatorGeometry}^{\uparrow}}$ *surface consists exactly of stable-address exposure through A, finite transition-operator exposure through T, phase-gate exposure through G, boundary escape-flux exposure through B, golden-shell projection through S, operator-algebra refusal through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8821.1. The visible packet contains only the stable-address row, transition table, phase gates, boundary escape-flux row, golden-shell projection, refusal row, and structural rows.

The stable-address and projection data are finite. The rows A and S are read through the Zeckendorf phase shell and golden-mean window siblings, so a consumer receives visible address and projection rows rather than a hidden metric space.

The operator data are likewise row-local. The transition table T and phase gates G record finite operator action on displayed addresses, while the boundary row B records the escape-flux surface inherited from the static boundary-flux packet.

Finally O, H, C, P, N close the local boundary. The refusal row O blocks rereading the packet as an invariant bilinear form, symplectic form, bracket, or multiplication, and the structural rows transport, replay, source, and name only the displayed operator-geometry data. □

Theorem 8821.3 (Flux operator geometry boundary route)**.** *Every accepted* $\mathsf{FluxOperatorGeometry}^{\uparrow}$ *consumer that reads transition dynamics, phase gates, or escape flux must pass through the finite route*

$$A, T, G, B, S, O, H, C, P, N.$$

Thus the consumer sees stable Zeckendorf addresses, finite transition operators, phase gates, boundary escape flux, golden-shell projection, operator-algebra refusal, componentwise transport, continuation replay, provenance, and local naming. It receives no hidden metric geometry, invariant bilinear form, invariant multiplication, physical impedance constant, or curvature-scalar readout.

Proof. Begin with the accepted carrier of Definition 8821.1. A downstream read has access only to the rows $A, T, G, B, S, O, H, C, P, N$.

First read the visible address surface. The stable address row A and golden-shell projection row S are inherited from the finite Zeckendorf phase-shell and golden-mean window routes, so they do not expose a hidden ambient geometry.

Next read the operator rows. The transition table T records finite moves between displayed addresses, the phase-gate row G records the visible phase effect of those moves, and the boundary row B records the escape-flux contribution before any operator-geometry read is exported.

Finally apply the refusal and structural rows. The row O records that no invariant bilinear form or invariant multiplication is available in the packet, while H, C, P, N transport, replay, source, and name the same finite route. Hence every consumer-facing read remains inside the displayed operator-geometry surface. □

Falsifiable prediction: Every accepted $\mathsf{FluxOperatorGeometry}^{\uparrow}$ read that serves a transition, phase-gate, or boundary-flux consumer must expose stable Zeckendorf addresses, finite transition

operators, phase gates, boundary escape flux, golden-shell projection, operator-algebra refusal, componentwise transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: zeckendorf-phase-shell, golden-mean-shift-window-six, static-green-boundary-flux: $\mathsf{FluxOperatorGeometry}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries stable address rows, transition operators, phase gates, boundary escape-flux rows, golden-shell projection, refusal of invariant algebraic structure, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.FluxOperatorGeometryUp` must eventually provide explicit $\mathsf{Nontrivial\ FluxOperatorGeometry}^{\uparrow}$ and $\mathsf{FieldFaithful\ FluxOperatorGeometry}^{\uparrow}$ instances preserving $A, T, G, B, S, O, H, C, P, N$ as separate displayed fields before stronger closure is claimed.

Closure status for $\mathsf{NameCert}_{\mathsf{FluxOperatorGeometry}^{\uparrow}}$.

Constructive story: A $\mathsf{FluxOperatorGeometry}^{\uparrow}$ packet is generated from stable Zeckendorf address rows, finite transition-operator tables, phase-gate rows, boundary escape-flux rows, golden-shell projection rows, operator-algebra refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FluxOperatorGeometry}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8821.1, Definition 8821.2, and Definition 8821.3 seed the finite flux-operator geometry packet over stable addresses, transition operators, phase gates, escape flux, golden-shell projection, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FluxOperatorGeometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, a standard bridge, hidden metric geometry, invariant bilinear forms, invariant multiplication, physical impedance laws, or curvature-scalar readouts.

Upgrade path: Obligation closure requires checked carrier admission, stable-address exposure, transition-operator table exactness, phase-gate exposure, boundary escape-flux handoff, golden-shell projection, operator-algebra refusal, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.FluxOperatorGeometryUp.

Chapter 8822

A Concrete Naming Certificate for $\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}$

Following Peano's existence theorem and Bishop–Bridges constructive analysis, the $\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}$ packet records the finite approximation surface where Euler polygon ledgers are read as Cauchy-style local-existence evidence without selecting a solution curve. It sits after the Euler-polygon route of chapter 1967, the Peano local-existence packet of chapter 3530, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite mesh-budget and readback route; it is not a selected solution, not a completed ambient real object, not a uniqueness theorem, not host compactness, not quotient stream equality, and not countable choice.

Definition 8822.1 (Cauchy–Peano Euler-polygon carrier)**.** *A* $\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, F, E, B, D, Q, R, H, C, G, N)$$

with the following displayed rows:

1. *I is the Peano interval-source row, binding the initial time, initial state, local radius, and finite time window read through* $\mathsf{PeanoExistence}^{\uparrow}$*.*
2. *F is the continuous vector-field row used by the local Peano request, read only as displayed finite* BHist *data.*
3. *E is the* $\mathsf{EulerPolygon}^{\uparrow}$ *approximation row, exposing the dyadic mesh, initial point, local slope control, vertex windows, step ledger, regular readback, and terminal seal route of the sibling Euler-polygon packet.*
4. *B is the compact step-budget row, listing the finite mesh refinement requests, local radius bounds, and containment checks that keep the displayed polygon segments inside the Peano window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for step widths, accumulated polygon error cells, and Cauchy comparison budgets.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after E, B, D have been displayed.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the finite approximation packet.*
8. *H records componentwise* hsame *transport for interval data, vector-field data, Euler-polygon rows, step budgets, dyadic ledgers, readback rows, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the route*

$$(I, F, E, B, D) \longmapsto Q \longmapsto R.$$

10. *G is the* Pkg *provenance over* CauchyPeanoEulerPolygon$^{\uparrow}$, PeanoExistence$^{\uparrow}$, EulerPolygon$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyPeanoEulerPolygon}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, E, B, D, Q, R, H, C, G, N$. It has no coordinate for a selected solution curve, uniqueness, ambient compactness, quotient equality of streams, host real equality, countable choice, Lean evidence, or bridge credentials.

Theorem 8822.2 (Cauchy–Peano Euler-polygon NameCert obligations)**.** *For every accepted* CauchyPeanoEulerPoly *carrier*

$$P = (I, F, E, B, D, Q, R, H, C, G, N),$$

the local NameCert$_{\text{CauchyPeanoEulerPolygon}^{\uparrow}}$ *surface consists exactly of carrier admission for the Peano interval source, displayed vector-field data, Euler-polygon approximation row, compact step-budget row, dyadic tolerance ledger, regular rational readback, terminal Real seal, componentwise transport, displayed replay, package provenance, and local naming. Every Cauchy–Peano approximation read is therefore routed through $I, F, E, B, D, Q, R, H, C, G, N$ before any consumer can inspect the terminal seal.*

Proof. Project the accepted carrier of Definition 8822.1. The rows I and F expose the local Peano request and vector-field data, while E supplies the Euler-polygon approximation route already named in chapter 1967.

The approximation budget is carried by B and D. The row B records finite mesh requests, local radius bounds, and containment checks; the row D records the dyadic widths, accumulated errors, and comparison budgets consumed before readback.

Rows Q and R are downstream of those displayed rows. The regular rational readback cannot be inspected before E, B, D, and the terminal Real seal cannot be inspected before Q.

The structural rows H, C, G, N transport, replay, package, and name exactly that finite route. Since the carrier contains no selected solution curve, uniqueness coordinate, ambient compactness witness, quotient equality, host equality, choice schedule, Lean evidence, or bridge credential, the NameCert surface cannot export any of those objects. □

Theorem 8822.3 (Compact step-budget factorization)**.** *For every accepted* CauchyPeanoEulerPolygon$^{\uparrow}$ *carrier*

$$P = (I, F, E, B, D, Q, R, H, C, G, N),$$

each finite approximation prefix exposed to a Peano consumer factors through the compact step-budget row B and the dyadic tolerance ledger D before it reaches the RegSeqRat$^{\uparrow}$ *row Q. In particular, the packet can certify only displayed finite mesh budgets and containment checks; it does not certify an ambient compactness theorem, a selected convergent subsequence, or a completed solution path.*

Proof. Read a finite approximation prefix through the Peano side. The interval row I supplies the requested local window, and the vector-field row F supplies the displayed slope data used by the Euler-polygon row E.

Before the prefix reaches a regular rational readback, the carrier forces the compact step-budget row B to expose the finite mesh request, the local radius bound, and the containment check for the inspected polygon segment.

The dyadic ledger D then supplies the numerical tolerance route for that same prefix. It records the displayed step widths and accumulated comparison budgets consumed by the handoff to Q.

Thus every readback-facing approximation prefix factors through B and D. The carrier has no coordinate for host compactness, selected subsequences, or completed solution paths, so those claims cannot be read from the finite step-budget factorization. □

Theorem 8822.4 (Real-seal route for Cauchy–Peano Euler polygons)**.** *For every accepted* CauchyPeanoEulerPolygo *carrier*

$$P = (I, F, E, B, D, Q, R, H, C, G, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* R *factors through the displayed route*

$$(I, F, E, B, D) \longmapsto Q \longmapsto R$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The seal records only the finite approximation packet connecting Euler polygons to Peano local-existence evidence; it does not assert a selected solution curve, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, uniqueness, host compactness, quotient stream equality, or countable choice.*

Proof. Start with a Real-facing consumer of R. The carrier exposes R only as the terminal row after the interval source, vector-field data, Euler-polygon approximation route, compact step budget, dyadic tolerance ledger, and regular rational readback have all been displayed.

Trace the route backward from the seal. The row R is downstream of Q, the row Q is downstream of E, B, D, and the Euler-polygon and budget rows are downstream of the local Peano request I, F.

The structural rows preserve this order. Componentwise hsame transport cannot add a second Real surface, displayed Cont replay records the same finite route, and the provenance and naming rows merely bind the displayed packet.

Therefore every terminal Real read factors through $I, F, E, B, D, Q, R, H, C, G, N$. Since no coordinate names a selected solution curve, ambient completeness principle, uniqueness witness, host compactness theorem, quotient equality, or choice schedule, none of those claims escapes through the seal. □

Falsifiable prediction: Every $\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}$ Real-facing read factors through a Peano interval row, displayed vector-field data, an EulerPolygonUp approximation row, a compact step-budget row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: euler-polygon, peano-existence, regseqrat, dyadicratcore, real: CauchyPeanoEulerPolygor is not bijective to any listed sibling because it binds the Euler-polygon approximation surface to the Peano finite local-existence budget and Real-seal route in one displayed NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}$ packet is generated from a Peano interval-source row, displayed vector-field data, an Euler-polygon approximation row, a compact step-budget row, a dyadic tolerance ledger, RegSeqRat readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyPeanoEulerPolygon}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8822.1, Definition 8822.2, Definition 8822.3, and Definition 8822.4 seed the finite approximation route connecting Euler polygons to Peano local-existence evidence.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyPeanoEulerPolygon}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected solution curves, ambient Real$^{\uparrow}$ completeness, uniqueness, host compactness, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Peano interval exposure, Euler-polygon dependency, compact step-budget factorization, DyadicRatCore tolerance accounting, RegSeqRat handoff, Real-seal routing, transport/replay/provenance coverage, and non-escape for BEDC.Derived.CauchyPeanoEulerPolygonUp.

Chapter 8823

A Concrete Naming Certificate for $\mathsf{LocatedCauchyNetSelector}^{\uparrow}$

Following Moore and Smith, Kelley, and Bishop, the $\mathsf{LocatedCauchyNetSelector}^{\uparrow}$ packet records the finite selector surface needed when a located Cauchy net is read through a regular-Cauchy window before reaching a $\mathsf{Real}^{\uparrow}$ seal. It sits after the located net packet of chapter 3587, the Moore–Smith source of chapter 3403, the regular subnet selector of chapter 6183, the stream naming row of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The packet selects only a finite located window and its regular readback route; it does not assert a global choice function on all nets, a quotient of directed indices, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8823.1 (Located Cauchy net selector carrier)**.** *A* $\mathsf{LocatedCauchyNetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, L, T, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed index data: finite index names, comparison rows, and upper-bound witnesses used by the local Moore–Smith window.*
2. *W is the selected finite window, storing the inspected indices and the located net values admitted by the* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *source.*
3. *L is the located radius row, recording the dyadic precision request and the located comparison data that make the window inspectable.*
4. *T is the tail witness row, showing that the selected window lies past the displayed directed threshold for the requested radius.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W, L, T.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *admission seal reached only after the directed data, selected window, located radius, tail witness, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for D, W, L, T, R, A and the structural rows.*
8. *C records displayed* Cont *replay from a located Cauchy-net request through the same finite window, radius, tail, readback, and real-seal route.*

9. *P is the* Pkg *provenance over* LocatedCauchyNetSelector$^{\uparrow}$, LocatedCauchyNet$^{\uparrow}$, MooreSmithNet$^{\uparrow}$, RegularCauchySubnetSelector$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{LocatedCauchyNetSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, L, T, R, A, H, C, P, N$. It has no coordinate for a global selector, countable choice, quotient net equality, host real equality, or an unbounded family of selected tails.

Theorem 8823.2 (Located Cauchy net selector NameCert obligations)**.** *For an accepted* LocatedCauchyNetSelector *carrier $S = (D, W, L, T, R, A, H, C, P, N)$, the local* NameCert$_{\text{LocatedCauchyNetSelector}^{\uparrow}}$ *surface consists exactly of directed index admission, selected finite-window admission, located radius readback, tail-witness exposure,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Every completion-facing consumer must read D, W, L, T before it reads R or A.*

Proof. Carrier admission is projection from Definition 8823.1. The rows D, W, L, T display the directed substrate, finite window, located radius, and tail witness before any numerical readback is available.

The regular readback is downstream of those rows. The RegSeqRat$^{\uparrow}$ coordinate R is read from the selected window and located tail data, while the Real$^{\uparrow}$ seal A is reached only after that regular readback has been displayed.

Transport, replay, provenance, and local naming are componentwise. Since the tuple contains no global selector, countable-choice schedule, quotient of net indices, host real equality, or unbounded family of tails, the NameCert surface is exactly the finite selector route listed above. □

Falsifiable prediction: Any accepted LocatedCauchyNetSelector$^{\uparrow}$ packet fails if a Real$^{\uparrow}$ seal is read before the directed data, selected window, located radius row, tail witness, RegSeqRat$^{\uparrow}$ readback, transport, replay, provenance, and local NameCert rows are displayed.

Closure status for NameCert$_{\text{LocatedCauchyNetSelector}^{\uparrow}}$**.**

Constructive story: A LocatedCauchyNetSelector$^{\uparrow}$ packet is generated from directed index data, a selected finite window, a located radius row, a tail witness, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCauchyNetSelector}^{\uparrow}}$, seed)

Scope closed: Definition 8823.1 and Definition 8823.2 seed the finite located-window selector surface for L10 completion routes.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyNetSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a global selector on all located Cauchy nets, countable choice, quotient net equality, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked directed-index admission, finite-window selection, located-radius readback, tail-witness exposure, RegSeqRat readback, Real-seal admission, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyNetSelectorUp.

Chapter 8824

A Concrete Naming Certificate for RhRouteClosureFixedStatementSeparation$^{\uparrow}$

The RhRouteClosureFixedStatementSeparation$^{\uparrow}$ packet realizes the route-layer versus fixed-statement separation of chapter 9337 as finite NameCert data. It keeps route-family closure rows separate from the single fixed RH proposition row, records refusal data for classifier rows that attempt to reuse a route-family obstruction as fixed-RH independence evidence, and exposes transport, replay, provenance, and local naming before any independence-facing consumer may read the packet. Its sibling context is the RH-free zero-generation route of chapter 8799, the zero-type factorization surface of chapter 8772, the finite prime-window gluing row of chapter 8765, and the route-halting boundary of chapter 5564.

Definition 8824.1 (RH route closure fixed statement separation carrier)**.** *A* RhRouteClosureFixedStatementSeparat *carrier is a finite* BHist *packet*

$$S = (R, F, U, B, T, C, H, Q, P, N)$$

with the following displayed rows:

1. *R is the route-layer row, recording a finite route-family classifier request over route codes and theorem targets.*
2. *F is the fixed-statement row, recording the single* RH *proposition as a fixed constructive statement rather than a family of routes.*
3. *U is the route-undecidability row, importing only the family-level obstruction from the route-halting boundary.*
4. *B is the row-split boundary, requiring any independence-facing consumer to display which data belong to R, U and which data belong to F.*
5. *T is the refusal row, storing failed attempts to reuse a route-family classifier obstruction as a fixed-RH independence witness.*
6. *C is the TasteGate-facing row requiring later Lean files to provide* Nontrivial RhRouteClosureFixedStatementSepara *and* FieldFaithful RhRouteClosureFixedStatementSeparation$^{\uparrow}$ *instances for the displayed route-layer and statement-layer fields.*
7. *H records componentwise* hsame *transport for route-layer, fixed-statement, obstruction, row-split, refusal, TasteGate, replay, provenance, and local-name rows.*

8. *Q records displayed* Cont *replay from a route-family obstruction request through R, U, B, T without rewriting it into a proof predicate for the fixed row F.*
9. *P is the* Pkg *provenance over* $\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}$, $\mathsf{RhFreeZeroInitiality}^{\uparrow}$, $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$, $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$, $\mathsf{RhRouteHaltingBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, F, U, B, T, C, H, Q, P, N$. It has no coordinate for a fragment-specific non-provability theorem, a proof or refutation of RH*, an identification of route closure with fixed-statement truth, or a total classifier for route-family closure.*

Theorem 8824.2 (RH route closure fixed statement separation NameCert obligations)**.** *For an accepted* $\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}$ *carrier $S = (R, F, U, B, T, C, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}}$ *surface consists exactly of route-layer admission, fixed-statement admission, family-level obstruction exposure, row-split boundary, route-to-fixed refusal, TasteGate-facing instance obligations, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Every independence-facing consumer must read the row-split boundary B and refusal row T before treating route undecidability as evidence about the fixed* RH *statement.*

Proof. Carrier admission is projection from Definition 8824.1. The route-layer row R and obstruction row U are family-level data, while F is the single fixed proposition row.

The row-split boundary B is read before any independence-facing consumer receives the packet. If a consumer attempts to identify the route-family obstruction with fragment-specific independence for the fixed row, the failed row is stored in T rather than accepted as a NameCert obligation.

The remaining rows preserve the displayed separation. The TasteGate-facing row C names the later Nontrivial and FieldFaithful obligations for the same carrier fields, while H, Q, P, N transport, replay, source, and name only the split route-layer and statement-layer data. □

Falsifiable prediction: Any route-family classifier row reused as a fixed-RH independence witness fails the $\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}$ NameCert obligation surface unless it first displays the route-layer row, fixed-statement row, row-split boundary, refusal row, transport, replay, provenance, and local naming data.

Independence witness: `rh_free_zero_initiality`, `zeta_zero_type_factorization`, `finite_prime_window` `rh_route_halting_boundary`: $\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}$ is not bijective to any listed sibling because its route-family row and fixed-statement row are separated before the refusal surface is read; the siblings supply zero-generation, zero-type, finite-prime gluing, and route-halting data, not a classifier equating route undecidability with fixed-statement independence.

Closure status for $\mathsf{NameCert}_{\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}$ packet is generated from a route-layer classifier request, a fixed RH statement row, a family-level route obstruction, a row-split boundary, a route-to-fixed refusal row, TasteGate-facing instance obligations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 8824.1 and Definition 8824.2 seed the finite route-layer, statement-layer, refusal, transport, replay, provenance, and local naming surface for the RH separation boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RhRouteClosureFixedStatementSeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close fragment-specific RH independence, a proof or refutation of RH, a total route-closure classifier, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked route-layer admission, fixed-statement admission, family-level obstruction exposure, row-split boundary, route-to-fixed refusal, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.RhRouteClosureFixedStatementSeparationUp.

Chapter 8825

A Concrete Naming Certificate for $\mathsf{TotalDerivative}^{\uparrow}$

Following the standard multivariable calculus tradition and its Fréchet-style presentation of linear approximation, the $\mathsf{TotalDerivative}^{\uparrow}$ packet joins the existing derivative fragments into one finite certificate surface. It reads the linear approximation branch through chapter 3634, the directional and one-parameter probes through chapter 6344 and chapter 7621, the linear operator row through chapter 48, the norm and error rows through chapter 89, and the terminal seal through chapter 38. The packet names a local total-linear-approximation route; it is not an inverse-function theorem, not an implicit-function theorem, not a smooth manifold atlas, not a selected ambient limit, and not a hidden completeness principle.

Definition 8825.1 (Total derivative carrier)**.** *A* $\mathsf{TotalDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, L, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source chart row, recording the inspected point, local chart window, and admissible perturbation rows.*
2. *Y is the target normed row, recording the codomain chart, norm readback, and comparison window consumed by the approximation.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the candidate linear part.*
4. *M is the approximation-modulus row, recording the finite tolerance schedule for the remainder.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the quotient-style error checks used by the displayed approximation.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the inspected error and norm comparisons.*
7. *E is the remainder ledger, stating that the residual after subtracting L is bounded by the displayed modulus at the requested windows.*
8. *H records componentwise* hsame *transport for source chart, target normed row, linear map, modulus, readback, Real seal, remainder ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from chart exposure to linear-map readback, norm comparison, remainder-modulus inspection, regular rational readback, and Real seal delivery.*

10. *P is the* Pkg *provenance over* $\mathsf{TotalDerivative}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{GateauxDerivative}^{\uparrow}$, $\mathsf{HadamardDirectionalDeriv}$ $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TotalDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, L, M, Q, R, E, H, C, P, N$. It has no coordinate for a global differentiability class, a selected host limit, an inverse map, an implicit solution branch, an ambient Banach-space theorem, or unrecorded directional probes.

Theorem 8825.2 (Total derivative NameCert obligations)**.** *For every accepted* $\mathsf{TotalDerivative}^{\uparrow}$ *carrier*

$$D = (X, Y, L, M, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TotalDerivative}^{\uparrow}}$ *surface consists exactly of source-chart exposure, target normed-row exposure, linear-map exposure, remainder-modulus exposure, regular rational readback, Real seal delivery, remainder-ledger exactness, componentwise transport, displayed replay, provenance, and local naming. A consumer may read the packet as a total derivative only after the rows for* $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{GateauxDerivative}^{\uparrow}$, $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *have been consumed through the displayed route.*

Proof. Project the accepted carrier of Definition 8825.1. The source chart, target normed row, linear map, modulus, readback, seal, and remainder ledger are all explicit rows of one finite packet.

Read the derivative fragments in their dependency order. The Fréchet row supplies the total linear-approximation shape, while the Gateaux and directional rows record the probe-level faces that the total packet refuses to leave detached.

Read the analytic rows next. The linear-map row L is supplied through $\mathsf{LinMap}^{\uparrow}$, the norm comparisons through $\mathsf{Norm}^{\uparrow}$, the regular rational readback through Q, and the Real seal through R.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Because no row stores a global differentiability class, selected host limit, inverse map, implicit branch, ambient Banach theorem, or unrecorded probe, the local NameCert obligations stop at the finite total-linear-approximation route. □

Theorem 8825.3 (Total derivative linear approximation route)**.** *Every accepted* $\mathsf{TotalDerivative}^{\uparrow}$ *consumer that cites the total derivative must traverse the ordered route*

$$X, Y, L, M, Q, R, E, H, C, P, N.$$

The route first exposes the source chart and target normed row, then the linear approximation, then the remainder modulus, then the regular rational readback and Real seal, and only then the remainder ledger, transport, replay, provenance, and local naming rows.

Proof. Start with the carrier and NameCert obligations. They identify $X, Y, L, M, Q, R, E, H, C, P, N$ as the complete finite surface and place the linear approximation downstream of the source and target rows.

Use the sibling derivative surfaces to control the route. The Fréchet packet supplies the total-linear approximation shape, while the Gateaux and directional packets keep one-parameter and direction-specific probes visible as dependencies rather than as replacements for L.

Use the norm and readback rows to inspect the residual. The row M gives the displayed modulus, Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ readback, R gives the $\mathsf{Real}^{\uparrow}$ seal, and E records the residual bound only after those rows have been exposed.

Thus a consumer cannot cite $\mathsf{TotalDerivative}^{\uparrow}$ as an unlogged differentiability assertion. The route is exactly $X, Y, L, M, Q, R, E, H, C, P, N$, with no selected host limit or hidden completeness row. □

Falsifiable prediction: Every accepted $\mathsf{TotalDerivative}^{\uparrow}$ read must expose source chart, target normed row, linear map, remainder modulus, RegSeqRat readback, Real seal, remainder ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data before a total-linear-approximation consumer is served.

Independence witness: frechet-derivative, gateaux-derivative, directional-derivative, linear-map, norm, real: $\mathsf{TotalDerivative}^{\uparrow}$ is not bijective to any listed sibling because it joins chart exposure, normed target data, linear approximation, remainder modulus, readback, seal, transport, replay, provenance, and local naming in one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TotalDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{TotalDerivative}^{\uparrow}$ packet is generated from source chart rows, target normed rows, a LinearMap row, a remainder modulus, RegSeqRat readback, a Real seal, a residual ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8825.1, Definition 8825.2, and Definition 8825.3 seed the finite total-linear-approximation packet over existing derivative, linear, norm, readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, a standard bridge, inverse-function or implicit-function theorems, global differentiability classes, selected host limits, ambient completeness, or manifold-atlas compatibility.

Upgrade path: Obligation closure requires checked carrier admission, derivative-fragment dependency, LinearMap exposure, Norm modulus accounting, RegSeqRat readback, Real-seal routing, remainder-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.TotalDerivativeUp.

Chapter 8826

A Concrete Naming Certificate for $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$

The $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ packet realizes the reversible hyperbolic disk-transport branch of chapter 9463 as finite NameCert data. It sits beside the Poincare disk transport surfaces of chapter 3455, chapter 6776, chapter 6267, and the metric row of chapter 3474. The packet names disk coordinates, boost parameters, boundary rows, metric preservation, phase transport, provenance, and local naming; it does not assert physical spacetime, light-cone ontology, a completed Lorentz group, or an ambient smooth manifold.

Definition 8826.1 (Lorentz boost disk-transport carrier)**.** *A* $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, A, U, V, M, \Phi, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare disk-coordinate row, recording the source and target disk windows.*
2. *A is the boost-parameter row, recording the finite rapidity or disk-automorphism parameter accepted by the local classifier.*
3. *U is the source boundary row and V is the transported boundary row.*
4. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row used to read metric preservation along the displayed disk transport.*
5. *Φ is the phase-transport row, recording the reversible observer-phase readback along the boost route.*
6. *O is the refusal row, recording the absence of physical spacetime claims, light-cone ontology, completed Lorentz group structure, and hidden smooth-manifold data.*
7. *H records componentwise* hsame *transport for disk, boost, boundary, metric, phase, refusal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from disk-coordinate exposure to boost-parameter admission, boundary transport, metric comparison, phase transport, and refusal checks.*
9. *P is the* Pkg *provenance over* $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, *disk automorphism packets, phase-transport packets,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LorentzBoostDiskTransport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, A, U, V, M, \Phi, O, H, C, P, N$. It has no coordinate for a physical observer, completed spacetime, external Lorentz group, host differentiability theorem, or unlogged boundary ideal point.

Theorem 8826.2 (Lorentz boost disk-transport NameCert obligations)**.** *For every accepted* $\mathsf{LorentzBoostDiskTrans}$ *carrier*

$$B = (D, A, U, V, M, \Phi, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LorentzBoostDiskTransport}^{\uparrow}}$ *surface consists exactly of disk-coordinate exposure, boost-parameter exposure, source and target boundary rows, hyperbolic metric comparison, reversible phase transport, refusal of host spacetime claims, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the accepted carrier of Definition 8826.1. The disk, boost, boundary, metric, phase, refusal, and structural rows are all explicit coordinates of one finite packet.

Read the hyperbolic dependencies. The disk-coordinate row is supplied through the existing Poincare disk surfaces, and the metric comparison row is read through $\mathsf{HyperbolicMetric}^{\uparrow}$ rather than through a host smooth model.

Read the reversible-transport branch next. The boost parameter A controls the displayed disk transport from U to V, while Φ records the observer-phase readback along that same finite route.

Finally O, H, C, P, N close the boundary. The refusal row blocks physical spacetime and completed-group readings, and the structural rows transport, replay, source, and name only the displayed disk-transport data. □

Theorem 8826.3 (Lorentz boost disk-transport reversible route)**.** *Every accepted* $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ *consumer that reads a boost-style disk transport must traverse the finite route*

$$D, A, U, V, M, \Phi, O, H, C, P, N.$$

Thus the consumer receives displayed disk coordinates, boost parameter, source and transported boundary rows, hyperbolic metric comparison, phase transport, refusal, componentwise transport, replay, provenance, and local naming before any transport claim is served.

Proof. Start with the carrier and NameCert obligations. They expose the exact rows $D, A, U, V, M, \Phi, O, H, C, P, N$ and exclude any ambient spacetime or completed Lorentz-group coordinate.

Use the disk and metric siblings to read the geometric surface. The Poincare disk rows expose D, U, V, while $\mathsf{HyperbolicMetric}^{\uparrow}$ supplies the displayed metric comparison M for the same transport route.

Use the boost and phase rows for reversibility. The boost parameter A names the finite transport instruction, and the phase row Φ records the reversible observer-phase readback without erasing the boundary rows.

The refusal and structural rows then preserve the boundary of the packet. Therefore every consumer-facing transport read remains inside $D, A, U, V, M, \Phi, O, H, C, P, N$ and cannot be repacked as host spacetime, an external Lorentz group, or a hidden smooth-manifold claim. □

Falsifiable prediction: Every accepted $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ read that serves a reversible disk-transport consumer must expose disk coordinates, boost parameter, boundary rows, HyperbolicMetric comparison, phase transport, refusal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: poincare-disk-phase-transport, poincare-disk-automorphism, poincare-disk-boundary-transport, hyperbolic-metric: $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boost-parameter admission, boundary transport, metric comparison, phase transport, refusal, transport, replay, provenance, and local naming in one displayed packet.

The Lean TasteGate route for `BEDC.Derived.LorentzBoostDiskTransportUp` must eventually provide explicit $\mathsf{Nontrivial\ LorentzBoostDiskTransport}^{\uparrow}$ and $\mathsf{FieldFaithful\ LorentzBoostDiskTransport}^{\uparrow}$ instances preserving $D, A, U, V, M, \Phi, O, H, C, P, N$ as separate displayed fields before stronger closure is claimed.

Closure status for $\mathsf{NameCert}_{\mathsf{LorentzBoostDiskTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{LorentzBoostDiskTransport}^{\uparrow}$ packet is generated from disk-coordinate rows, boost-parameter rows, boundary rows, HyperbolicMetric comparison rows, phase-transport rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LorentzBoostDiskTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8826.1, Definition 8826.2, and Definition 8826.3 seed the reversible hyperbolic disk-transport packet over disk, boost, boundary, metric, phase, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LorentzBoostDiskTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, a standard bridge, physical spacetime, light-cone ontology, a completed Lorentz group, host differentiability, or ambient smooth-manifold structure.

Upgrade path: Obligation closure requires checked carrier admission, disk-coordinate exposure, boost-parameter exactness, boundary transport, HyperbolicMetric comparison, phase-transport reversibility, TasteGate nontriviality, field faithfulness, refusal, transport, replay, provenance, and local naming for BEDC.Derived.LorentzBoostDiskTransportUp.

Chapter 8827

A Concrete Naming Certificate for $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$ packet records the finite-window rate boundary by which bounded real sequence data may be read by Bolzano–Weierstrass-style compactness consumers. It sits after the bounded real sequence carrier of chapter 1415, the finite-subsequence packet of chapter 8797, the dyadic tolerance core of chapter 291, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite read budget available to a compactness-facing route; it does not select a subsequence, invoke countable choice, decide real trichotomy, identify quotient streams, or assert ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 8827.1 (Bolzano–Weierstrass rate carrier)**.** *A* $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, F, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source row, read through* $\mathsf{BoundedRealSequence}^{\uparrow}$ *before any rate request is accepted.*
2. *F is the finite-subsequence window row, read through* $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$ *as a finite compactness-facing consumer rather than as an infinite selected subsequence.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger assigning each inspected window a displayed dyadic comparison budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite source positions and observations that the rate certificate may inspect.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerances.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after B, F, D, W, Q have been exposed.*
7. *H records componentwise* hsame *transport for the source, finite-subsequence window, tolerance ledger, stream windows, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through $B, F, D, W, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$*,* $\mathsf{BoundedRealSequence}^{\uparrow}$*,* $\mathsf{BolzanoWeierstrassFiniteSubseque}$ $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassRate}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $B, F, D, W, Q, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, countable-choice schedule, trichotomy decision, quotient stream representative, or completeness proof for $\mathsf{Real}^{\uparrow}$.

Theorem 8827.2 (Bolzano–Weierstrass rate NameCert obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$ *carrier*

$$R = (B, F, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassRate}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded source B, finite-subsequence window F, dyadic tolerance ledger D, stream windows W, regular rational readback Q, terminal real seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N. No NameCert obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 8827.1. The bounded source row B is visible only as the $\mathsf{BoundedRealSequence}^{\uparrow}$ surface, while F is visible only as the finite-window face of $\mathsf{BolzanoWeierstrassFiniteSubsequer}$

Read the rate rows next. The dyadic ledger D assigns the tolerance budget for the inspected window, the stream row W lists the finite observations, and the regular readback row Q consumes exactly those observations.

Finally inspect E, H, C, P, N. The real seal, transport, replay, provenance, and local naming rows preserve the same finite route. Since the carrier has no coordinate for countable choice, real trichotomy, quotient stream equality, selected infinite subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot become local NameCert obligations. □

Theorem 8827.3 (Bolzano–Weierstrass rate window handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BolzanoWeierstrassRate}^{\uparrow}$ *packet must read the route*

$$\mathsf{BoundedRealSequence}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

through $B, F, D, W, Q, E, H, C, P, N$ before treating the packet as Bolzano–Weierstrass-style rate data. The handoff exposes a bounded real sequence source, a finite subsequence-window consumer, dyadic tolerance comparisons, finite stream windows, regular rational readback, and a terminal real seal. It exports no selected subsequence, no countable choice, no trichotomy, no quotient stream equality, and no ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Start with the bounded source. The row B is admitted only through the $\mathsf{BoundedRealSequence}^{\uparrow}$ carrier, so a consumer first receives finite stream windows, regular readback, the closed-bounded interval-net bound row, and the real-name seal already exposed by that source packet.

Next read the finite-subsequence window. The row F is downstream of $\mathsf{BolzanoWeierstrassFiniteSubsequence}^{\uparrow}$, so it presents a finite compactness-facing window rather than a selected infinite subsequence.

The dyadic, stream, and regular readback rows D, W, Q then form the rate budget inspected by the consumer. The terminal row E seals only that displayed readback as $\mathsf{Real}^{\uparrow}$ data.

Transport, replay, provenance, and local naming preserve the same finite-window route. Because no additional row carries choice, trichotomy, quotient equality, or completeness, the public handoff is exactly the rate boundary displayed in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassRate}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassRate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8827.1, Definition 8827.2, and Definition 8827.3 seed the finite-window rate boundary over bounded Real sequence, finite Bolzano–Weierstrass subsequence-window, DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BolzanoWeierstrassRate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected subsequences, countable choice, real trichotomy, quotient stream equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window rate exactness, source-row stability, dyadic tolerance readback, terminal Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BolzanoWeierstrassRateUp.

Constructive story: A BolzanoWeierstrassRate$^{\uparrow}$ packet is generated from a bounded Real sequence source, a finite Bolzano–Weierstrass subsequence-window row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8828

A Concrete Naming Certificate for $\mathsf{CartanHadamard}^{\uparrow}$

Following Cartan and Hadamard, the $\mathsf{CartanHadamard}^{\uparrow}$ packet records the finite nonpositive-curvature and geodesic-handoff surface by which a simply connected Riemannian presentation is read inside BEDC. It sits after the manifold carrier of chapter 96, the Riemannian metric row of chapter 133, the curvature row of chapter 135, the finite geodesic handoff of chapter 5929, and the terminal real-name seal of chapter 38. The packet names a displayed curvature-controlled geodesic route; it does not assert a host universal cover, global exponential diffeomorphism, selected minimizing geodesics, quotient equality of paths, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8828.1 (Cartan-Hadamard carrier)**.** *A* $\mathsf{CartanHadamard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, G, K, S, E, R, W, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *source row, exposing the local chart and overlap data used by the packet.*
2. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *metric row, read only through the displayed metric and length coordinates.*
3. *K is the* $\mathsf{Curvature}^{\uparrow}$ *comparison row, recording the nonpositive-curvature ledger visible to the local classifier.*
4. *S is the simply connected presentation row, recorded as a finite NameCert datum rather than as an ambient host theorem.*
5. *E is the geodesic extension row, consumed through the finite-geodesic Hopf–Rinow sibling before a geodesic read is public.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for displayed lengths, distances, and curvature bounds.*
7. *W is the obstruction-refusal row, recording the absence of a host universal-cover object, selected global exponential inverse, quotient path equality, or choice schedule.*
8. *H records componentwise* hsame *transport for manifold, metric, curvature, simply connected, geodesic, Real, refusal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a Cartan–Hadamard request through $M, G, K, S, E, R, W, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CartanHadamard}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{HopfRinowFiniteGeodesic}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CartanHadamard}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, G, K, S, E, R, W, H, T, P, N$. It has no coordinate for a host universal cover, global exponential diffeomorphism, selected minimizing geodesics, quotient equality of paths, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8828.2 (Cartan-Hadamard NameCert obligations)**.** *For every accepted* $\mathsf{CartanHadamard}^{\uparrow}$ *carrier $C = (M, G, K, S, E, R, W, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CartanHadamard}^{\uparrow}}$ *surface consists exactly of manifold-source admission, Riemannian metric exposure, nonpositive-curvature ledger exposure, simply connected presentation, finite geodesic handoff, Real readback, obstruction refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every geometry-facing consumer must read*

$$M, G, K, S, E, R, W, H, T, P, N$$

before the Cartan–Hadamard packet is consumed.

Proof. Project the carrier of Definition 8828.1. The manifold and metric data occur only as the rows M and G, so a consumer cannot replace them by an ambient smooth manifold or a hidden metric tensor.

Read the geometric rows next. The curvature row K records the nonpositive-curvature comparison ledger, the row S records the simply connected presentation, and the row E routes geodesic reads through the finite Hopf–Rinow sibling.

Finally use R, W, H, T, P, N. The Real row exposes only displayed length, distance, and curvature-bound readbacks, while the refusal and structural rows preserve the same finite route. Since no carrier coordinate stores a host universal cover, global exponential diffeomorphism, selected minimizing geodesic, quotient path equality, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8828.3 (Cartan-Hadamard geodesic boundary)**.** *Every accepted* $\mathsf{CartanHadamard}^{\uparrow}$ *geodesic or curvature consumer factors through the visible route*

$$\mathsf{Manifold}^{\uparrow},\ \mathsf{RiemannianMetric}^{\uparrow},\ \mathsf{Curvature}^{\uparrow} \longmapsto \mathsf{HopfRinowFiniteGeodesic}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CartanHadamard}^{\uparrow}.$$

The handoff supplies a finite curvature-controlled geodesic packet, not a host universal-cover theorem, global exponential diffeomorphism, selected minimizing-geodesic principle, quotient path equality, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Start with the source rows. The manifold row fixes the displayed chart data, and the Riemannian metric row supplies only visible length and distance coordinates.

Use the curvature and connectivity rows. The nonpositive-curvature ledger K and simply connected presentation row S are read before the finite geodesic handoff E is available.

Route the handoff through R, W, H, T, P, N. The Real readback seals the displayed numerical data, the refusal row blocks host universal-cover and global exponential readings, and the structural rows transport, replay, source, and name the same finite packet. Hence every public geodesic or curvature read remains inside the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CartanHadamard}^{\uparrow}$ read that serves a geodesic or curvature consumer must expose ManifoldUp, RiemannianMetricUp, CurvatureUp, HopfRinowFiniteGeodesicUp, RealUp, obstruction-refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: manifold, riemannianmetric, curvature, hopf-rinow-finite-geodesic, real: $\mathsf{CartanHadamard}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries simply connected presentation, nonpositive-curvature ledger, finite geodesic handoff, Real readback, obstruction refusal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CartanHadamard}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CartanHadamard}^{\uparrow}$ packet is generated from ManifoldUp source rows, RiemannianMetricUp metric rows, CurvatureUp nonpositive-curvature ledgers, simply connected presentation rows, HopfRinowFiniteGeodesicUp handoffs, RealUp readbacks, obstruction-refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CartanHadamard}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8828.1, Definition 8828.2, and Definition 8828.3 seed the finite Cartan–Hadamard route over manifold, metric, curvature, geodesic, Real, refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CartanHadamard}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host universal-cover theorem, global exponential diffeomorphism, selected minimizing geodesics, quotient path equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, manifold-source exposure, Riemannian metric exposure, curvature-ledger exactness, simply connected presentation, finite-geodesic handoff, Real readback, obstruction refusal, transport, replay, provenance, and local naming for BEDC.Derived.CartanHadamardUp.

Chapter 8829

A Concrete Naming Certificate for $\mathsf{OmegaRigidityLedger}^{\uparrow}$

The $\mathsf{OmegaRigidityLedger}^{\uparrow}$ packet realizes the mature bridge and Ω-rigidity spine of chapter 9362 as finite NameCert data. It sits after the licensed generation body of chapter 8756 and the no-host-leak audit surface of chapter 1217. The packet records bridge reports, standard readouts, lower-layer NameCert dependencies, compatibility checks, audit rows, and no-replacement rows for an Ω-tower rigidity claim; it does not import an external standard model as a BEDC proof term, close RH, select a zero witness, or replace any lower-layer certificate.

Definition 8829.1 (Omega rigidity ledger carrier)**.** *An* $\mathsf{OmegaRigidityLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, B, S, N, K, A, R, H, C, P, Q)$$

with the following displayed rows:

1. *O is the* $\mathsf{OmegaGenerationBody}^{\uparrow}$ *row, exposing the compatible tower body and its readback arrows.*
2. *B is the bridge-report row, listing the mature bridge packets read only after internal tower rows are present.*
3. *S is the standard-readout row, recording external interpretations as readouts rather than as BEDC constructors.*
4. *N is the lower-layer* $\mathsf{NameCert}$ *dependency row, naming the internal certificates that must already be present before the ledger is consumed.*
5. *K is the compatibility row, recording agreement between tower projections, bridge reports, standard readouts, and lower-layer NameCert rows.*
6. *A is the audit row, tracking unsupported fields, host leakage checks, axiom-use checks, and marker resolution for the finite ledger.*
7. *R is the no-replacement row, recording that no bridge report or standard readout substitutes for a missing internal carrier, classifier, stability, ledger, or naming obligation.*
8. *H records componentwise* hsame *transport for tower body, bridge report, standard readout, lower-layer NameCert, compatibility, audit, no-replacement, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an Ω-rigidity ledger request through $O, B, S, N, K, A, R, H, P, Q$.*

10. *P is the* Pkg *provenance over* $\mathsf{OmegaRigidityLedger}^{\uparrow}$, $\mathsf{OmegaGenerationBody}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, *the mature bridge spine of chapter 9362,* BHist, hsame, Cont, *and* NameCert.
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{OmegaRigidityLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, B, S, N, K, A, R, H, C, P, Q$. It has no coordinate for an external model as proof term, a host RH theorem, a selected zero witness, a hidden bridge replacement, or a detached standard readout.

Theorem 8829.2 (Omega rigidity ledger NameCert obligations)**.** *For every accepted* $\mathsf{OmegaRigidityLedger}^{\uparrow}$ *carrier $L = (O, B, S, N, K, A, R, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{OmegaRigidityLedger}^{\uparrow}}$ *surface consists exactly of tower-body exposure, bridge-report exposure, standard-readout exposure, lower-layer NameCert dependency exposure, compatibility checking, audit checking, no-replacement checking, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every rigidity-ledger consumer must read*

$$O, B, S, N, K, A, R, H, C, P, Q$$

before it can consume the Ω-rigidity ledger.

Proof. Project the carrier of Definition 8829.1. The tower body is visible only as O, and the bridge and standard rows are visible only as B and S.

Read the dependency and compatibility rows next. The row N lists the lower-layer NameCert packets already required by the ledger, while K checks that bridge reports and standard readouts agree with those internal rows.

Finally use A, R, H, C, P, Q. The audit row records leakage and axiom-use checks, the no-replacement row blocks external substitution for internal obligations, and the structural rows transport, replay, source, and name the same finite ledger. Since no carrier coordinate stores an external model as proof term, host RH theorem, selected zero witness, hidden bridge replacement, or detached standard readout, the listed rows exhaust the local NameCert surface. □

Theorem 8829.3 (Omega rigidity ledger no-replacement boundary)**.** *Every accepted* $\mathsf{OmegaRigidityLedger}^{\uparrow}$ *consumer that reads a bridge report or standard readout must also read the lower-layer NameCert row N, compatibility row K, audit row A, and no-replacement row R. Thus the bridge side can interpret the already displayed tower body, but it cannot generate a missing internal carrier, classifier, stability field, ledger row, naming row, RH proof, or selected zero witness.*

Proof. Start from the bridge-facing rows B and S. By carrier projection, they occur in the same packet as the lower-layer dependency row N and cannot be consumed as detached bridge data.

Use the compatibility row K. It compares bridge reports and standard readouts against the internal tower body O and the already named lower-layer certificates.

Apply the audit and no-replacement rows. The row A records host-leakage and axiom-use checks, while R states that bridge and standard rows do not replace internal obligations. Transport, replay, provenance, and local naming then preserve the same finite ledger, so no external readout becomes a BEDC proof term. □

Falsifiable prediction: Every accepted $\mathsf{OmegaRigidityLedger}^{\uparrow}$ read that exports a bridge report or standard readout without exposing OmegaGenerationBodyUp, lower-layer NameCert rows, compatibility checks, audit checks, no-replacement rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows is outside the carrier surface.

Independence witness: omega-generation-body, standardbridgeauditpacket, mature-bridge-omega-rigidity-spine: $\mathsf{OmegaRigidityLedger}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries tower body, bridge report, standard readout, lower-layer NameCert dependency, compatibility, audit, no-replacement, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{OmegaRigidityLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{OmegaRigidityLedger}^{\uparrow}$ packet is generated from an OmegaGenerationBodyUp tower-body row, bridge-report rows, standard-readout rows, lower-layer NameCert dependency rows, compatibility rows, audit rows, no-replacement rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OmegaRigidityLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8829.1, Definition 8829.2, and Definition 8829.3 seed the finite Omega-rigidity ledger over tower body, bridge report, standard readout, lower-layer NameCert, compatibility, audit, no-replacement, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OmegaRigidityLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, zero rigidity, external standard-model soundness as an internal proof term, selected zero witnesses, missing lower-layer NameCert obligations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tower-body exposure, bridge-report exposure, standard-readout exposure, lower-layer NameCert dependency exposure, compatibility exactness, audit exactness, no-replacement checking, transport, replay, provenance, and local naming for BEDC.Derived.OmegaRigidityLedgerUp.

Chapter 8830

A Concrete Naming Certificate for $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ packet records the finite endpoint-separation budget read before an L10 Real observation route may treat two displayed endpoint rows as separated. It sits after the real-name seal of chapter 38, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the guarded sign-apartness surface of chapter 8304. The packet names only the inspected separation rows; it is not trichotomy, quotient real equality, selected endpoint representatives, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8830.1 (Finite endpoint-separation budget carrier)**.** *A* $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, D, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint* $\mathsf{Real}^{\uparrow}$ *row, already sealed through the displayed real-name route.*
2. *U is the upper endpoint* $\mathsf{Real}^{\uparrow}$ *row, sealed through the same route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing the finite positive radius, outward slack, and comparison precision used to separate the endpoints.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row from which endpoint observations are inspected at the requested tolerance.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows.*
6. *S is the* $\mathsf{LocatedSignApartness}^{\uparrow}$ *sign-apartness row certifying that the inspected endpoint difference is separated at the displayed budget.*
7. *H records componentwise* hsame *transport for endpoint, tolerance, window, readback, sign-apartness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an endpoint-separation request through $L, U, D, W, Q, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSignApartness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, W, Q, S, H, C, P, N$. It has no coordinate for total real order, quotient representatives, selected endpoints, countable choice, compactness, or ambient completeness.

Theorem 8830.2 (Finite endpoint-separation budget NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ *carrier* $B = (L, U, D, W, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteEndpointSeparationB}}$ *surface consists exactly of endpoint admission for* L, U*, dyadic budget exposure through* D*, finite-window exposure through* W*, regular rational readback through* Q*, located sign-apartness exposure through* S*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 8830.1. The endpoint rows, tolerance row, finite windows, readback, sign-apartness row, and structural rows are all explicit coordinates of one finite packet.

The analytic route is ordered. Endpoint reads first expose L, U, then the dyadic budget D, then the finite windows W, and only then the regular rational readback Q used by the sign-apartness row S.

Transport, replay, provenance, and local naming are componentwise. Since the tuple contains no total order decision, quotient representative, selected endpoint, countable-choice schedule, compactness theorem, or ambient completeness object, the listed rows exhaust the local NameCert surface. □

Theorem 8830.3 (Finite endpoint-separation budget Real route)**.** *Every L10 Real endpoint-separation consumer of an accepted* $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ *carrier factors through the displayed route*

$$L, U \longmapsto D, W, Q \longmapsto S, H, C, P, N.$$

Thus downstream interval, enclosure, and approximation chapters may cite one finite budget object instead of repeating endpoint-budget rows locally, while receiving no trichotomy, quotient real equality, selected endpoints, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint rows L, U. They are already sealed $\mathsf{Real}^{\uparrow}$ rows, so a consumer reads endpoint data only through the real-name route rather than through host representatives.

Next apply the dyadic-window-readback handoff. The dyadic budget D determines the inspected tolerance, the window row W exposes finite endpoint observations, and the readback row Q presents those observations as regular rational data.

Finally read the located sign-apartness row S and replay the structural rows H, C, P, N. The consumer receives a finite endpoint-separation budget for the displayed endpoints, and no coordinate remains from which the excluded global principles could be recovered. □

Falsifiable prediction: Every accepted $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ read that serves an L10 Real endpoint-separation consumer must expose endpoint rows, a DyadicRatCore budget, StreamName windows, RegSeqRat readback, LocatedSignApartness evidence, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: real, dyadicratcore, streamname, regseqrat, located-sign-apartness: none is bijective to $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ because only this packet binds two sealed endpoints, one finite dyadic separation budget, finite observation windows, regular readback, guarded sign-apartness evidence, transport, replay, provenance, and local naming into one endpoint-budget carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}$ packet is generated from lower and upper RealUp endpoint rows, a DyadicRatCore tolerance budget, StreamName finite windows, RegSeqRat readback, LocatedSignApartness evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8830.1, Definition 8830.2, and Definition 8830.3 seed the finite endpoint-separation budget route for L10 Real observation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteEndpointSeparationBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total real order, quotient real equality, selected endpoint representatives, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic budget exposure, finite-window readback, located sign-apartness evidence, transport, replay, provenance, and local naming for BEDC.Derived.FiniteEndpointSeparationBudgetUp.

Chapter 8831

A Concrete Naming Certificate for $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$

Following Bing, Nagata, Smirnov, and Engelking on metrization criteria, the $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$ packet records the finite BEDC route by which a regular topological source exposes both a Bing-style development route and a Nagata–Smirnov locally finite cover route before a metric-space consumer reads the handoff. It sits over the topology carrier of chapter 88, beside the Bing packet of chapter 6479, beside the Nagata–Smirnov packet of chapter 6348, and downstream of the paracompactness surface of chapter 1963 when displayed local finiteness is supplied. The packet names only the combined finite criterion route through $\mathsf{BingMetrization}^{\uparrow}$, $\mathsf{NagataSmirnovMetrization}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Paracompact}^{\uparrow}$ anchors; it does not assert host open-cover compactness, selected covers, quotient topology equality, countable choice, or an ambient point-set metrization oracle.

Definition 8831.1 (Bing–Nagata–Smirnov metrization carrier)**.** *A* $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, B, N, P, L, S, R, E, Q, M, F, H, C, G, A)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the open-row classifier, point-neighbourhood requests, and regular separation reads.*
2. *B is the* $\mathsf{BingMetrization}^{\uparrow}$ *development route row, exposing the displayed star-refinement and development-level checks read from the source topology.*
3. *N is the* $\mathsf{NagataSmirnovMetrization}^{\uparrow}$ *cover-family route row, exposing the displayed locally finite layer schedule and base-refinement checks.*
4. *P is the* $\mathsf{Paracompact}^{\uparrow}$ *dependency row, used only when the packet has an explicitly displayed local-finiteness supply.*
5. *L is the common locally finite ledger through which the Bing and Nagata–Smirnov routes are compared.*
6. *S is the combined schedule row, recording the finite development levels and cover layers inspected by the packet.*
7. *R is the regularity and refinement row, binding closed-neighbourhood separation to the displayed cover refinements.*

8. *E is the exactness row asserting that every accepted criterion request is read through both B and N before any metric handoff is consumed.*
9. *Q is the dyadic radius schedule shared by the combined criterion route.*
10. *M is the $\mathsf{Met}^{\uparrow}$ handoff row, available only after $T, B, N, P, L, S, R, E, Q$ have been exposed.*
11. *F records finite compatibility between metric balls from M and topology windows from T along the two sibling metrization routes.*
12. *H records componentwise* hsame *transport for all displayed rows.*
13. *C records displayed* Cont *replay from a topology request through Bing development data, Nagata–Smirnov cover data, paracompact local-finiteness data, regularity, exactness, dyadic radii, metric handoff, compatibility, provenance, and local naming.*
14. *G is the* Pkg *provenance over* $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$, $\mathsf{BingMetrization}^{\uparrow}$, $\mathsf{NagataSmirnovMetrization}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
15. *A is the local* $\mathsf{NameCert}_{\mathsf{BingNagataSmirnovMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open-cover compactness, selected covers, quotient topology equality, arbitrary refinements, countable choice, or an ambient metric oracle.

Theorem 8831.2 (Bing–Nagata–Smirnov metrization NameCert obligations)**.** *For every accepted* $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$ *carrier $K = (T, B, N, P, L, S, R, E, Q, M, F, H, C, G, A)$, the local* $\mathsf{NameCert}_{\mathsf{BingNagataSmirnovMetrization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, Bing development route, Nagata–Smirnov cover route, paracompact local-finiteness dependency, common locally finite ledger, combined schedule, regularity-refinement row, exactness row, dyadic radius schedule, metric-space handoff, compatibility, transport, replay, provenance, and local naming row;*
2. *sibling route readability, requiring every accepted criterion request to start at T and pass through the displayed* $\mathsf{BingMetrization}^{\uparrow}$ *and* $\mathsf{NagataSmirnovMetrization}^{\uparrow}$ *rows before M is inspected;*
3. *local-finiteness compatibility, requiring P, L, S, R to expose only finite cover-layer and separation checks supplied inside the packet;*
4. *metric handoff exactness, requiring E, Q, M, F to keep the* $\mathsf{Met}^{\uparrow}$ *row downstream of the combined criterion route and compatible with the original topology windows;*
5. *refusal, requiring G and A to name only this finite packet and to refuse host open-cover compactness, selected covers, quotient topology equality, arbitrary refinements, countable choice, and ambient metric oracles.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 8831.1. The projection exposes exactly the topology source, Bing route, Nagata–Smirnov route, paracompact local-finiteness dependency, common ledger, combined schedule, regularity-refinement row, exactness row, dyadic radius schedule, metric handoff, compatibility row, and the structural rows H, C, G, A.

Sibling route readability is forced by row order. A criterion request starts at T, reads the development data in B, reads the locally finite cover-family data in N, and then compares both through the common ledger L and schedule S.

The metric handoff is downstream of the finite compatibility route. Rows P, L, S, R, E record the local-finiteness supply, cover-layer checks, regularity separation, and exactness witness, while Q, M, F record the dyadic schedule, metric-space handoff, and finite topology compatibility.

Since the tuple contains no host open-cover compactness coordinate, selected-cover coordinate, quotient topology equality coordinate, arbitrary refinement coordinate, countable-choice coordinate, or ambient metric oracle, the local NameCert cannot export those principles. □

Theorem 8831.3 (Bing–Nagata–Smirnov metrization handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{BingMetrization}^{\uparrow},\ \mathsf{NagataSmirnovMetrization}^{\uparrow},\ \mathsf{Paracompact}^{\uparrow},\ L,\ S,\ R,\ E,\ Q,\ M,\ F,\ H,\ C,\ G,\ A.$$

The handoff row M is consumed only as a downstream metric-space interface tied to the displayed Bing development rows, Nagata–Smirnov locally finite cover rows, paracompact local-finiteness dependency row, regularity-refinement rows, and topology compatibility windows. It is not a host metric, not a selected-cover construction, and not a claim that the source topology equals a quotient topology.

Proof. Read the carrier rows in order. The topology source T, Bing route B, Nagata–Smirnov route N, paracompact dependency P, common ledger L, combined schedule S, and regularity row R are displayed before the exactness row can license a metric-facing read.

The rows E, Q, M, F then supply the downstream interface: exactness binds the two sibling criteria to the same accepted request, the dyadic schedule prepares the metric handoff, and the compatibility row ties metric balls back to the topology source windows.

Transport and replay through H and C preserve that same route, while G and A record provenance and local naming. Because no row stores host open-cover compactness, selected covers, quotient topology equality, countable choice, or an ambient metric oracle, the $\mathsf{Met}^{\uparrow}$ consumer receives only the finite BEDC handoff named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BingNagataSmirnovMetrization}^{\uparrow}}$.

Constructive story: A $\mathsf{BingNagataSmirnovMetrization}^{\uparrow}$ packet is generated from a topology source row, Bing development route, Nagata–Smirnov cover-family route, paracompact local-finiteness dependency, common locally finite ledger, combined schedule, regularity-refinement row, exactness row, dyadic radius schedule, MetricSpace handoff, topology compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BingNagataSmirnovMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8831.1, Definition 8831.2, and Definition 8831.3 seed the finite combined Bing and Nagata–Smirnov criterion route to a displayed MetricSpace handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BingNagataSmirnovMetrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, selected covers, quotient topology equality, arbitrary refinements, countable choice, ambient metric oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling route readability, local-finiteness compatibility, metric handoff exactness, transport, replay, provenance, and local naming for the displayed Bing–Nagata–Smirnov metrization packet.

Chapter 8832

A Concrete Naming Certificate for $\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}$

Following Michael's selection theorem in its finite closed-convex refinement reading, the MichaelSelectionFiniteRefine packet records the BEDC surface where one displayed paracompact cover cell family, one lower-semicontinuous closed-convex value row, and one finite refinement ledger are inspected before any local continuous-selection handoff is public. It is a finite sibling of chapter 2300; it consumes the paracompact refinement rows of chapter 1963, the lower-semicontinuous value boundary of chapter 1536, the convex target rows of chapter 208, the continuous-map handoff of chapter 130, and the metric observation discipline of chapter 57. The packet names only finite refinement data; it does not assert a global selector, arbitrary choice over fibres, Banach-space completeness, host function equality, or ambient point-set topology.

Definition 8832.1 (Michael finite closed-convex refinement carrier)**.** *A* $\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, V, K, R, S, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Paracompact}^{\uparrow}$ *domain-cover row whose cells are being refined.*
2. *V is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *closed-value row exposed over the displayed cover cells.*
3. *K is the* $\mathsf{ConvexSet}^{\uparrow}$ *target row, carrying the closed-convex witnesses used by each inspected value row.*
4. *R is the finite refinement ledger, listing the retained cover cells, their refinement addresses, and the value-row incidence data.*
5. *S is the local finite selection row, recording only the selections attached to the retained refinement cells.*
6. *T is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row available after D, V, K, R, S have been displayed.*
7. *H records componentwise* hsame *transport for the domain, value, convex-target, refinement, local selection, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$D, V, K, R, S \longmapsto T \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* MichaelSelectionFiniteRefinement$^{\uparrow}$, MichaelSelection$^{\uparrow}$, Paracompact$^{\uparrow}$, LowerSemicontinuous$^{\uparrow}$, ConvexSet$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, V, K, R, S, T, H, C, P, N$. It has no coordinate for a host global selector, arbitrary choice over all fibres, quotient equality of functions, Banach-space completeness, or an ambient topology theorem.

Theorem 8832.2 (Michael finite refinement NameCert obligations)**.** *For every accepted* MichaelSelectionFiniteRefin *carrier $F = (D, V, K, R, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, paracompact cover exposure through D, lower-semicontinuous closed-convex value exposure through V, convex target exposure through K, finite refinement incidence through R, local finite selection rows through S, continuous-map handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every continuous-map consumer must read the refinement route D, V, K, R, S before treating T as public, and no obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8832.1. The rows $D, V, K, R, S, T, H, C, P, N$ are the whole finite packet, so carrier admission and the visible obligation rows are read directly from that tuple.

The selection data are local to the displayed refinement ledger. The paracompact row supplies the cover cells, the lower-semicontinuous row supplies the closed-value windows, the convex row supplies target witnesses, and R, S bind the retained finite refinement cells to the visible local selections.

The continuous-map row T is downstream of those rows. Transport and replay through H and C preserve the same finite route, while P, N source and name it. Since the excluded selector, choice, completeness, host-function, and ambient-topology objects have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8832.3 (Michael finite refinement sibling boundary)**.** *Every accepted* MichaelSelectionFiniteRefinement$^{\uparrow}$ *packet is a finite refinement read of the* MichaelSelection$^{\uparrow}$ *surface of chapter 2300. Its public handoff factors through*

Paracompact$^{\uparrow}$, LowerSemicontinuous$^{\uparrow}$, ConvexSet$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The handoff names one displayed finite closed-convex refinement ledger and no global selection theorem.

Proof. Start from Definition 8832.2. The visible route exposes the paracompact domain cover, the lower-semicontinuous closed-value row, the convex target row, the finite refinement ledger, and the local selection rows before the continuous-map handoff.

Compare this route with the sibling carrier in Definition 2300.1. The finite packet narrows the selection ledger to the retained refinement cells R, S while keeping the same paracompact, lower-semicontinuous, convex, continuous-map, metric, transport, replay, provenance, and naming dependencies.

Thus a consumer may read a finite closed-convex refinement handoff, but it cannot promote that read to arbitrary choice, a host global selector, quotient equality of functions, Banach-space completeness, or an ambient point-set selection theorem. □

Falsifiable prediction: Every accepted $\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}$ read exposes a displayed $\mathsf{Paracompact}^{\uparrow}$ cover row, $\mathsf{LowerSemicontinuous}^{\uparrow}$ closed-value row, $\mathsf{ConvexSet}^{\uparrow}$ target row, finite refinement ledger, local selection row, terminal $\mathsf{ContinuousMap}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: michael-selection, paracompact, lowersemicontinuous, convexset, continuousmap, metric: $\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}$ is not bijective to any listed sibling because it binds one finite refinement ledger and its local closed-convex selection rows into a separate NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8832.1, Definition 8832.2, and Definition 8832.3 seed the finite closed-convex refinement surface over paracompact covers, lower-semicontinuous value rows, convex target rows, continuous-map handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global selection, arbitrary choice over fibres, host function equality, Banach-space completeness, ambient point-set topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite refinement incidence, local selection-row exactness, continuous-map handoff boundary, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.MichaelSelectionFiniteRefinementUp.

Constructive story: A $\mathsf{MichaelSelectionFiniteRefinement}^{\uparrow}$ packet is generated from a paracompact cover row, a lower-semicontinuous closed-value row, a closed-convex target row, finite refinement and local selection rows, a continuous-map handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 8833

A Concrete Naming Certificate for $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$

Following Bishop and Bridges on finite dyadic covers and constructive completion, the $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ packet records the finite L10 route that transfers a displayed dyadic uniform cover into StreamName, RegSeqRat, Real, interval-net, and compact-metric consumers. It refines the dyadic finite-cover direction marked in chapter 9238 and consumes the cover source of chapter 8290, the dyadic arithmetic spine of chapter 291, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, the dyadic interval-net surface of chapter 5966, and the compact-metric source of chapter 129. The packet names a finite cover-transfer carrier only; it does not assert open-cover compactness, choose a subcover, quotient real names, select limits, use countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8833.1 (Dyadic uniform cover-transfer carrier)**.** *A* $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (U, D, W, Q, R, I, K, L, H, C, P, N)$$

with the following displayed rows:

1. *U is the upstream* $\mathsf{DyadicUniformCover}^{\uparrow}$ *row whose finite cover cells are being transferred.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, overlap, and tolerance ledger used by the transfer.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row for cover-cell endpoints and membership reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational center and endpoint data exposed by W.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, W, Q have been displayed.*
6. *I is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row that packages the cover cells as interval-net data.*
7. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *consumer row, available only after U, D, W, Q, R, I have been displayed.*
8. *L is the finite transfer ledger, recording that the same cover cells and tolerance comparisons feed the stream, regular-rational, Real, interval-net, and compact-metric rows.*
9. *H records componentwise* hsame *transport for the cover, dyadic, stream, regular-rational, Real, interval-net, compact-metric, transfer-ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$U, D, W, Q, R, I, K, L \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$, $\mathsf{DyadicUniformCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicUniformCoverTransfer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, W, Q, R, I, K, L, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient stream equality, host real equality, countable choice, selected limits, or ambient compactness and completeness.

Theorem 8833.2 (Dyadic uniform cover-transfer NameCert obligations)**.** *For every accepted* $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ *carrier $T = (U, D, W, Q, R, I, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicUniformCoverTran}}$ *surface consists exactly of upstream dyadic-cover exposure through U, dyadic endpoint and tolerance exposure through D, finite StreamName window exposure through W, regular-rational readback through Q, Real sealing through R, dyadic interval-net handoff through I, compact-metric handoff through K, finite transfer-ledger exactness through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. Every L10 completion or compactness consumer must read U, D, W, Q, R, I, L before treating K as public, and no obligation is stored outside those rows.*

Proof. Project the carrier of Definition 8833.1. The transfer source is U, the arithmetic and readback rows are D, W, Q, R, the interval-net and compact-metric consumers are I, K, and the finite transfer ledger is L.

Apply the upstream dyadic-cover handoff of Definition 8290.3. It forces any compactness or completion read to expose the dyadic ledger, finite stream window, regular-rational readback, Real seal, and finite cover row before using the cover as a source.

The new transfer ledger L keeps that same finite cover route attached to the interval-net row I and the compact-metric row K. Transport and replay through H and C preserve the displayed rows, while P, N source and name the same packet. Since no excluded open-cover, selected-subcover, quotient, choice, limit-selection, or ambient-completeness object appears as a coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8833.3 (Dyadic uniform cover-transfer L10 route)**.** *Every L10 Real, Completion, interval-net, or compact-metric consumer of an accepted* $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ *packet factors through*

$\mathsf{DyadicUniformCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BH

The route is a finite cover-transfer packet from displayed dyadic cells to readback and compact-metric rows, not an open-cover compactness theorem or a selected subcover principle.

Proof. Begin with Definition 8833.2. It places the upstream cover row U, dyadic ledger D, finite stream window W, regular-rational readback Q, Real seal R, interval-net row I, compact-metric row K, and transfer ledger L in one displayed route.

The sibling chapters provide the public surfaces consumed by that route: chapter 291 supplies dyadic arithmetic, chapter 126 and chapter 301 supply finite observation and rational readback, chapter 38 supplies the seal, chapter 5966 supplies the interval-net handoff, and chapter 129 supplies the compact-metric consumer boundary.

Finally inspect H, C, P, N. These structural rows only transport, replay, source, and name the finite transfer route. They cannot add arbitrary open covers, selected subcovers, quotient real names, host real equality, countable choice, selected limits, or ambient completeness, so every public L10 read remains inside the displayed cover-transfer packet. □

Falsifiable prediction: Every accepted $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ read exposes an upstream $\mathsf{DyadicUniformCover}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{DyadicIntervalNet}^{\uparrow}$ handoff, $\mathsf{CompactMetric}^{\uparrow}$ consumer row, finite transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadic-uniform-cover, dyadicratcore, streamname, regseqrat, real, dyadic-interval-net, compactmetric: $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ is not bijective to any listed sibling because it binds upstream dyadic cover cells to one finite L10 transfer ledger across readback, Real sealing, interval-net, and compact-metric consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicUniformCoverTransfer}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicUniformCoverTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8833.1, Definition 8833.2, and Definition 8833.3 seed the finite L10 cover-transfer surface over dyadic cover, dyadic arithmetic, StreamName, RegSeqRat, Real, dyadic interval-net, compact-metric, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicUniformCoverTransfer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient real names, host real equality, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upstream cover exposure, dyadic endpoint exactness, finite-window readback, Real-seal exposure, interval-net and compact-metric handoff exactness, transfer-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.DyadicUniformCoverTransferUp.

Constructive story: A $\mathsf{DyadicUniformCoverTransfer}^{\uparrow}$ packet is generated from an upstream dyadic uniform cover row, dyadic endpoint and tolerance data, finite StreamName windows, RegSeqRat readback, a Real seal, dyadic interval-net and compact-metric rows, a finite transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8834

A Concrete Naming Certificate for $\mathsf{EulerMethodConvergence}^{\uparrow}$

Following Euler and Cauchy, and the Bishop–Bridges constructive treatment of numerical approximation, the $\mathsf{EulerMethodConvergence}^{\uparrow}$ packet records the finite route by which explicit Euler meshes are compared, budgeted, and handed to a regular real readback. It consumes the finite forward-step packet of chapter 6231, the finite ODE bound of chapter 8548, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names constructive explicit-Euler mesh convergence as finite NameCert data; it is not a host ODE existence theorem, a selected solution curve, quotient equality of paths, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8834.1 (Euler method convergence carrier)**.** *An* $\mathsf{EulerMethodConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, G, W, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ *step packet, exposing the local ODE row, vector-field read, finite step ledger, endpoint update row, and dyadic arithmetic row.*
2. *G is the* $\mathsf{GronwallFiniteOde}^{\uparrow}$ *bound packet, carrying the finite Lipschitz, comparison, dyadic majorant, and ODE estimate rows used to compare meshes.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row, recording which mesh refinements and endpoint reads are inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed endpoint approximations and error budgets.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for mesh widths, local truncation bounds, Gronwall accumulation, and target precision requests.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, G, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for step, bound, window, readback, dyadic, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the finite step route through the Gronwall budget and regular readback.*

9. *P is the* Pkg *provenance over* $\mathsf{EulerMethodConvergence}^{\uparrow}$, $\mathsf{EulerMethodFiniteStep}^{\uparrow}$, $\mathsf{GronwallFiniteOde}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerMethodConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, G, W, R, D, L, H, C, P, N$. It has no coordinate for an exact solution curve, host convergence theorem, selected limiting path, quotient equality of functions, countable-choice schedule, or ambient real completeness.

Theorem 8834.2 (Euler method convergence NameCert obligations)**.** *For every accepted* EulerMethodConvergence *carrier*

$$E = (S, G, W, R, D, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerMethodConvergence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *finite-step admission through S, including the displayed ODE source, vector-field read, step ledger, endpoint update row, and dyadic arithmetic row;*
2. *Gronwall comparison admission through G, including the finite Lipschitz and accumulated-error majorant used to compare mesh refinements;*
3. *observation and regular-readback admission through W and R, so every endpoint approximation is read as a finite window before any real seal is exposed;*
4. *dyadic error-budget admission through D, stable under componentwise* hsame *transport H and displayed* Cont *replay C;*
5. *provenance and local naming through P, N, with the terminal* $\mathsf{Real}^{\uparrow}$ *seal L available only after the previous four rows are visible.*

No local obligation may read exact ODE solution existence, global continuation, selected limiting paths, quotient path equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $E = (S, G, W, R, D, L, H, C, P, N)$. The finite-step packet S gives the only endpoint-update source, while the Gronwall packet G gives the only accumulated-error comparison row.

The observation route is then forced through W and R. A convergence consumer can inspect only the displayed mesh windows and the regular rational readback attached to those windows, so it cannot invoke a host limiting path.

The dyadic ledger D records the mesh-width, local-error, accumulated-error, and target-precision budgets. Transport and replay through H and C preserve those displayed rows componentwise.

Finally P, N source and name the same packet, and the terminal seal L is exposed only after S, G, W, R, D have been read. Since no carrier coordinate names exact ODE solution existence, global continuation, selected limits, quotient paths, choice, or ambient completeness, the five rows exhaust the local NameCert surface. □

Theorem 8834.3 (Euler method convergence error-budget route)**.** *Every accepted* $\mathsf{EulerMethodConvergence}^{\uparrow}$ *consumer that exports a completed real endpoint must read the error budget through*

$$S, G, W, R, D, L, H, C, P, N.$$

The route first exposes finite Euler steps, then the Gronwall comparison budget, then finite observation windows and regular rational readback, then the dyadic tolerance ledger, and only then the terminal real seal. It exports no exact solution object, selected limiting path, quotient path equality, countable-choice schedule, or ambient-completeness principle.

Lean4 (statement) `BEDC.Derived.EulerMethodConvergenceUp`

Proof. Start with an accepted carrier $E = (S, G, W, R, D, L, H, C, P, N)$. The finite-step data S and comparison data G determine which mesh endpoints and accumulated bounds are available.

Read the observation row W and the regular rational row R next. These rows make endpoint comparison a finite-window and regular-readback operation rather than a direct appeal to a completed curve.

The dyadic ledger D is then the only tolerance source for the target precision. The terminal row L may be exposed only after that ledger has been displayed, and H, C, P, N preserve, replay, source, and name the same route.

Thus any completed real endpoint exported by the packet factors through the displayed ten rows. No remaining row can supply a hidden solution object, selected limit, quotient equality, choice schedule, or ambient-completeness claim. □

Falsifiable prediction: An accepted $\mathsf{EulerMethodConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed finite-step row, Gronwall comparison row, StreamName window row, RegSeqRat readback row, DyadicRatCore error ledger, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: euler-method-finite-step, gronwall-finite-ode, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{EulerMethodConvergence}^{\uparrow}$ because this packet binds finite Euler steps, Gronwall mesh comparison, finite observation windows, regular rational readback, dyadic error budgets, terminal real sealing, transport, replay, provenance, and local naming into one convergence route.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerMethodConvergence}^{\uparrow}}$.

Constructive story: An $\mathsf{EulerMethodConvergence}^{\uparrow}$ packet is generated from an EulerMethodFiniteStep row, a GronwallFiniteOde comparison row, StreamName observation windows, RegSeqRat readback, a DyadicRatCore error ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EulerMethodConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 8834.1, Definition 8834.2, and Definition 8834.3 seed the constructive explicit-Euler mesh convergence route over finite-step, Gronwall, stream-window, regular-readback, dyadic-budget, real-seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EulerMethodConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close exact ODE solution existence, global continuation, selected limiting paths, quotient path equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-step visibility, Gronwall comparison, windowed readback, dyadic error budgets, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.EulerMethodConvergence

Chapter 8835

A Concrete Naming Certificate for $\mathsf{LocalizedDarbouxIntermediate}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{LocalizedDarbouxIntermediate}^{\uparrow}$ packet records the finite L10 route from a located interval source, through Darboux-continuity data, into a localized bisection readback before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits beside the Darboux-continuity surface of chapter 3828, the located intermediate-value route of chapter 3938, the constructive intermediate-value packet of chapter 6410, the finite $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The certificate names a local route only; it does not claim global compactness, a host-selected zero, quotient real equality, or ambient completeness.

Definition 8835.1 (Localized Darboux intermediate carrier)**.** *A* $\mathsf{LocalizedDarbouxIntermediate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, B, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row, naming the endpoint* $\mathsf{Real}^{\uparrow}$ *seals, the located order witness, and the source window in which the intermediate read is requested.*
2. *D is the Darboux-continuity row, recording the local graph exposure and crossing data consumed from* $\mathsf{DarbouxContinuity}^{\uparrow}$*.*
3. *B is the bisection route row, listing the retained subintervals, inspected midpoint samples, and sign-apartness observations used by the localized search.*
4. *T is the dyadic tolerance ledger from* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording endpoint brackets, midpoint brackets, crossing tolerances, and the terminal window budget.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by B and T.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed windows and dyadic tolerances into regular rational observations.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after I, D, B, T, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the located interval, Darboux row, bisection ledger, dyadic tolerance ledger, window row, readback row, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any localized intermediate-value consumer through I, D, B, T, W, R, E and then through the structural rows.*
10. *P is the* Pkg *provenance over* LocalizedDarbouxIntermediate$^{\uparrow}$, DarbouxContinuity$^{\uparrow}$, LocatedIntermediateValue$^{\uparrow}$, ConstructiveIntermediateValue$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocalizedDarbouxIntermediate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for open-cover compactness, ambient real completeness, trichotomy, a selected host zero, countable choice, or quotient equality of completed real names.

Lean4 ✓ `BEDC.Derived.LocalizedDarbouxIntermediateUp.LocalizedDarbouxIntermediateTasteGate`
`single carrier alignment`

Theorem 8835.2 (Localized Darboux intermediate NameCert obligations)**.** *For every accepted* LocalizedDarbouxIntermediate$^{\uparrow}$ *carrier $L = (I, D, B, T, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{LocalizedDarbouxIntermed}}$ *surface consists exactly of the following five rows: located interval admission through I; Darboux and bisection exposure through D, B; dyadic tolerance, finite-window, and regular readback through T, W, R; terminal* Real$^{\uparrow}$ *sealing through E; and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier rows. The located interval row I, Darboux row D, bisection row B, dyadic tolerance ledger T, window row W, regular readback R, and real seal E are the only analytic rows available to a consumer.

Group those rows by their obligation role. Source admission is I; Darboux and bisection exposure are D, B; numerical observation is T, W, R; the terminal real-name handoff is E.

The remaining rows H, C, P, N transport, replay, source, and name the same packet. Since no further coordinate is present, the five listed rows exhaust the local NameCert surface. □

Theorem 8835.3 (Localized Darboux intermediate bisection route)**.** *Every consumer of an accepted* LocalizedDarbouxIntermediate$^{\uparrow}$ *packet reaches the intermediate* Real$^{\uparrow}$ *seal only through the displayed located interval row I, Darboux row D, bisection route B, finite* StreamName$^{\uparrow}$ *windows W,* RegSeqRat$^{\uparrow}$ *readback R, dyadic tolerance ledger T, and terminal* Real$^{\uparrow}$ *row E, with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row still attached.*

Proof. Begin with the located source. The carrier exposes I before any Darboux or bisection row, so a consumer cannot start from an ambient interval image or detached real-name selection.

Read the Darboux and bisection rows next. The row D supplies the local crossing surface, while B records the retained subintervals and midpoint observations used by the localized route.

The numerical handoff is finite. The window row W, dyadic ledger T, and regular rational readback R must be displayed before the terminal seal E is available.

Finally H, C, P, N preserve and name this same route. No carrier coordinate supplies open-cover compactness, ambient Real completeness, trichotomy, a selected host zero, countable choice, or quotient equality. □

Closure status for NameCert$_{\text{LocalizedDarbouxIntermediate}^{\uparrow}}$ **.**

Constructive story: A LocalizedDarbouxIntermediate$^{\uparrow}$ packet is generated from a located interval source, Darboux-continuity row, bisection route, dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{LocalizedDarbouxIntermediate}\uparrow}$, seed)

Scope closed: Definition 8835.1, Definition 8835.2, and Definition 8835.3 seed the local L10 Darboux-intermediate route over located interval, Darboux, bisection, StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{LocalizedDarbouxIntermediate}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient Real$^{\uparrow}$ completeness, trichotomy, selected host zeros, quotient equality of completed real names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Darboux row exposure, bisection-route exactness, dyadic tolerance accounting, finite StreamName windows, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocalizedDarbouxIntermediateUp.

Chapter 8836

A Concrete Naming Certificate for $\mathsf{LocatedNumber}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedNumber}^{\uparrow}$ packet records a finite located-number interface for the L10 Real and Completion route before later comparison, apartness, or completion consumers read a real-name boundary. It sits after the natural scale rows of chapter 28, the rational bracket rows of chapter 37, the dyadic tolerance arithmetic of chapter 291, the finite windows of chapter 126, the regular rational readback of chapter 301, the scale compatibility packet of chapter 1507, and the terminal real-name seal of chapter 38. The packet names a displayed located-number carrier; it is not quotient real equality, a selected limit, trichotomy, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8836.1 (Located number carrier)**.** *A* $\mathsf{LocatedNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (N, Q, D, S, R, A, E, H, C, P, M)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *scale row, recording the finite precision request at which a located bracket is inspected.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *bracket row, displaying the lower and upper rational bounds presented at scale N.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording dyadic radii, midpoint arithmetic, bracket slack, and endpoint comparison data for the same request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened only after N, Q, D have been displayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same window and dyadic tolerance ledger.*
6. *A is the* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *scale-compatibility row tying the natural scale, rational bracket, dyadic tolerance, finite window, and readback rows to one approximation request.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after N, Q, D, S, R, A are visible.*
8. *H records componentwise* hsame *transport for the scale, bracket, tolerance, window, readback, Archimedean compatibility, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for each scale request, bracket read, tolerance check, window opening, readback, compatibility check, and Real seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedNumber}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *M is the local* $\mathsf{NameCert}_{\mathsf{LocatedNumber}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $N, Q, D, S, R, A, E, H, C, P, M$. It has no coordinate for quotient real equality, selected representatives, selected limits, trichotomy, countable-choice schedules, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8836.2 (Located number NameCert obligations)**.** *For every accepted* $\mathsf{LocatedNumber}^\uparrow$ *carrier $L = (N, Q, D, S, R, A, E, H, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedNumber}^\uparrow}$ *surface consists exactly of carrier admission,* $\mathsf{Nat}^\uparrow$ *scale exposure,* $\mathsf{Rat}^\uparrow$ *bracket exposure,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance exactness,* $\mathsf{StreamName}^\uparrow$ *finite-window opening,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{ArchimedeanApproximation}^\uparrow$ *scale compatibility,* $\mathsf{Real}^\uparrow$ *terminal sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the displayed rows $N, Q, D, S, R, A, E, H, C, P, M$.*

Proof. Project the accepted packet. The analytic rows are precisely the scale row N, rational bracket row Q, dyadic tolerance row D, finite window row S, regular readback row R, Archimedean compatibility row A, and terminal seal row E.

Read the sibling obligations in their displayed order. The scale row controls the request, the rational bracket and dyadic tolerance rows display the finite enclosure data, the StreamName window and RegSeqRat readback rows expose only finite observations, and the ArchimedeanApproximation row binds those rows to one scale-compatible approximation request.

The structural rows do not add analytic content. Componentwise transport, continuation replay, provenance, and local naming preserve the same packet, so quotient equality, selected representatives, selected limits, trichotomy, countable choice, and ambient completeness remain outside the certificate. □

Theorem 8836.3 (Located number Real route)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{LocatedNumber}^\uparrow$ *carrier reads the displayed route*

$$\mathsf{Nat}^\uparrow,\ \mathsf{Rat}^\uparrow,\ \mathsf{DyadicRatCore}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{ArchimedeanApproximation}^\uparrow,\ \mathsf{Real}^\uparrow$$

before the packet is public. The consumer receives a located-number boundary through finite scale, bracket, tolerance, window, readback, compatibility, and seal rows; it receives no quotient real equality, selected limit, trichotomy, countable-choice schedule, or ambient $\mathsf{Real}^\uparrow$ *completeness theorem.*

Proof. Start with the scale and bracket rows. The request is finite because N names the inspected scale and Q names only the rational bracket displayed at that scale.

Next read the dyadic tolerance, finite window, and regular readback rows. The rows D, S, R force every endpoint or tolerance observation through displayed dyadic arithmetic, a finite StreamName window, and a RegSeqRat readback before any Real-facing boundary is available.

Finally read A and E. The ArchimedeanApproximation row ties the preceding rows to one scale-compatible approximation request, and the Real seal is terminal with respect to that request. Transport, replay, provenance, and local naming preserve the same route, so no excluded equality, selected limit, choice, trichotomy, or completeness principle can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNumber}^\uparrow}$.

Constructive story: A $\mathsf{LocatedNumber}^\uparrow$ packet is generated from Nat scale rows, Rat bracket rows, DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, ArchimedeanApproximation scale compatibility, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedNumber}^\uparrow}$, seed)

Scope closed: Definition 8836.1, Definition 8836.2, and Definition 8836.3 seed the finite located-number interface for the L10 Real and Completion route.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedNumber}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, countable choice, selected representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scale exposure, bracket exposure, dyadic tolerance exactness, finite-window opening, regular readback, Archimedean scale compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNumberUp.

Chapter 8837

A Concrete Naming Certificate for MetaCICSubjectReductionObstructionWitness$^{\uparrow}$

The MetaCICSubjectReductionObstructionWitness$^{\uparrow}$ packet realizes the simultaneous obstruction surface of chapter 9386 as finite NameCert data. It sits after the checked boundary packet of chapter 721, the four-row discharge matrix of chapter 6922, the proof-obligation gap of chapter 3460, and the discharge-inversion boundary of chapter 6111. The witness records that the beta-substitution, application-argument, lambda-domain, and Pi-domain rows must be inspected together before a subject-reduction consumer may read the parameterised theorem; it does not close subject reduction.

Definition 8837.1 (MetaCIC subject-reduction obstruction witness carrier)**.** *A* MetaCICSubjectReductionObstructi *carrier is a finite* BHist *packet*

$$W = (B, A, L, D, M, Q, O, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BetaSubstitutionPreservation *obligation row.*
2. *A is the* AppArgTypeStable *obligation row.*
3. *L is the* LamDomainSubjectReduction *obligation row.*
4. *D is the* PiDomainSubjectReduction *obligation row.*
5. *M is the* MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$ *row, retaining simultaneous four-row coverage.*
6. *Q is the* ProofObligationGap$^{\uparrow}$ *row, recording named missing discharge rather than positive preservation evidence.*
7. *O is the obstruction row imported from the* MetaCICSubjectReductionBoundaryUp *packet.*
8. *S is the* MetaCICDischargeInversionBoundary$^{\uparrow}$ *socket row, preventing transport, replay, provenance, or local naming from filling one of the four discharge fields.*
9. *H records componentwise* hsame *transport for the four obligation rows, matrix, gap, obstruction, socket, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same obstruction witness route.*

11. *P is the* Pkg *provenance over* MetaCICSubjectReductionObstructionWitness$^{\uparrow}$, MetaCICSubjectReductionBoundaryU MetaCICSubjectReductionDischargeMatrix$^{\uparrow}$, ProofObligationGap$^{\uparrow}$, MetaCICDischargeInversionBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{MetaCICSubjectReductionObstructionWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, L, D, M, Q, O, S, H, C, P, N$. It has no coordinate for a closed subject-reduction theorem, host conversion, erased derivation semantics, quotient equality of terms, global normalization, or kernel consistency.

Theorem 8837.2 (MetaCIC subject-reduction obstruction witness obligations)**.** *For every accepted* MetaCICSubjectReductionObstructionWitness$^{\uparrow}$ *carrier*

$$W = (B, A, L, D, M, Q, O, S, H, C, P, N),$$

the local NameCert$_{\text{MetaCICSubjectReductionObstructionWitness}^{\uparrow}}$ *surface consists exactly of simultaneous beta-substitution, application-argument, lambda-domain, and Pi-domain obligation exposure through B, A, L, D, matrix coverage through M, proof-obligation-gap polarity through Q, checked obstruction retention through O, socket-preserving discharge inversion through S, componentwise transport and displayed replay through H, C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The four subject-reduction obligations are displayed as separate rows B, A, L, D, and no one of them contains the data of another.

The matrix row M records simultaneous coverage. It therefore prevents a consumer from treating a single positive row as a closed subject-reduction proof.

The gap and obstruction rows Q, O fix the polarity of the packet. They record named missing discharge and the retained subject-reduction boundary rather than positive preservation evidence.

Finally the socket row S and the structural rows H, C, P, N preserve the same route. Since none of those rows has a coordinate for host conversion, erased derivation semantics, quotient equality, global normalization, or kernel consistency, the listed rows exhaust the local certificate. □

Theorem 8837.3 (MetaCIC subject-reduction obstruction witness non-escape)**.** *Every subject-reduction-facing consumer of an accepted* MetaCICSubjectReductionObstructionWitness$^{\uparrow}$ *packet must read the route*

$$B, A, L, D, M, Q, O, S, H, C, P, N.$$

If any one of B, A, L, D is absent, the consumer remains at the obstruction witness and cannot export the parameterised subject-reduction theorem as a closed theorem.

Proof. Use Definition 8837.2. A consumer reaches the matrix row M only after the four displayed obligation rows are visible.

If one of B, A, L, D is absent, the proof-obligation gap Q and obstruction row O are the only available answer for that branch. The socket row S keeps transport, replay, provenance, and local naming from replacing the missing discharge row.

Thus the public read remains on the displayed route $B, A, L, D, M, Q, O, S, H, C, P, N$. The packet exports no closed subject-reduction theorem, no host conversion rule, no erased derivation semantics, no quotient term equality, no global normalization theorem, and no kernel-consistency theorem. □

Lean4 (statement) `BEDC.Derived.MetaCICSubjectReductionObstructionWitnessUp`

Falsifiable prediction: A MetaCIC subject-reduction consumer that reads the witness as a closed theorem while any one of BetaSubstitutionPreservation, AppArgTypeStable, LamDomainSubjectReduction,

or PiDomainSubjectReduction is absent refutes the $\mathsf{MetaCICSubjectReductionObstructionWitness}^{\uparrow}$ carrier surface.

Independence witness: metacic-subject-reduction-boundary, metacic-subject-reduction-discharge-matrix, proof-obligation-gap, metacic-discharge-inversion-boundary: none is bijective to MetaCICSubjectReductionO because this packet binds the simultaneous four-row obstruction, matrix coverage, named gap polarity, retained obstruction, socket-preserving inversion, transport, replay, provenance, and local naming as one witness surface.

Lean follow-up must provide `Nontrivial MetaCICSubjectReductionObstructionWitnessUp` and `FieldFaithful MetaCICSubjectReductionObstructionWitnessUp` instances in `BEDC.Derived.MetaCICSub` The instances must keep $B, A, L, D, M, Q, O, S, H, C, P, N$ field-faithful, so the four discharge rows, matrix row, proof-obligation gap, obstruction row, and inversion socket cannot collapse into one carrier coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstructionWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubjectReductionObstructionWitness}^{\uparrow}$ packet is generated from beta-substitution, application-argument, lambda-domain, Pi-domain, discharge-matrix, proof-obligation-gap, obstruction, discharge-inversion socket, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstructionWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8837.1, Definition 8837.2, and Definition 8837.3 seed the simultaneous four-row subject-reduction obstruction witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstructionWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, host conversion, quotient term equality, global normalization, kernel consistency, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-row exposure, matrix coverage, proof-obligation-gap polarity, obstruction retention, socket-preserving inversion, transport, replay, provenance, local naming, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.MetaCICSubjectReductionObstructionWitnessUp.

Chapter 8838

A Concrete Naming Certificate for $\mathsf{DarbouxSumRefinement}^{\uparrow}$

Following Darboux and Riemann integration, the $\mathsf{DarbouxSumRefinement}^{\uparrow}$ packet records the finite refinement surface that compares Darboux sums over a displayed partition with Darboux sums over a displayed refinement before any Real-facing Cauchy handoff is read. It sits between the finite Darboux-sum boundary of chapter 1880, the lower-sum side of chapter 5858, the dyadic partition ledger of chapter 1863, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a finite refinement certificate for integration estimates; it does not import host suprema, selected mesh limits, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8838.1 (Darboux-sum refinement carrier)**.** *A* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, P, Q, L, U, M, D, C, E, H, K, G, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval and bounded-integrand observation row visible to the Darboux-sum consumer.*
2. *P is the source* $\mathsf{DyadicPartition}^{\uparrow}$ *row.*
3. *Q is the refined* $\mathsf{DyadicPartition}^{\uparrow}$ *row together with the finite cell-incidence map from Q to P.*
4. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row over the source partition P.*
5. *U is the upper finite Darboux-sum row over the same source partition P.*
6. *M is the refinement monotonicity ledger comparing the lower and upper rows on P with the corresponding finite rows on Q.*
7. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row carrying the paired lower and upper finite sums through the common refinement route.*
8. *C is the Cauchy-readiness row that turns the displayed upper-minus-lower refinement gap into a regular rational window for a Real-facing consumer.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, P, Q, L, U, M, D, C have been displayed.*
10. *H records componentwise* hsame *transport for interval, partition, refinement, sum, gap, readiness, seal, replay, provenance, and local naming rows.*

11. *K records displayed* Cont *replay from Darboux, Riemann, Real, or Completion requests through the same finite rows.*
12. *G records* Pkg *provenance over* $\mathsf{DarbouxSumRefinement}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxSumRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, P, Q, L, U, M, D, C, E, H, K, G, N$. It has no coordinate for a host supremum or infimum, selected limiting mesh, quotient equality of functions, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.DarbouxSumRefinementUp`

Theorem 8838.2 (Darboux-sum refinement NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *carrier $R = (I, P, Q, L, U, M, D, C, E, H, K, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxSumRefinement}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the interval row, source partition, refined partition, lower-sum row, upper-sum row, refinement monotonicity ledger, Darboux-sum handoff, Cauchy-readiness row, Real seal, transport, replay, provenance, and local name;*
2. *partition refinement coherence, requiring Q to refine P through a displayed finite cell-incidence map before any sum comparison is consumed;*
3. *lower and upper sum stability, requiring the source lower and upper rows L, U and their refined counterparts to be compared only through the ledger M;*
4. *Darboux handoff confinement, requiring the* $\mathsf{DarbouxSum}^{\uparrow}$ *row D to expose the paired lower and upper finite sums through the same refinement route;*
5. *Real-facing Cauchy readiness, requiring the row C and terminal seal E to be reached only after the finite refinement gap has been displayed;*
6. *transport, replay, provenance, and naming, requiring H, K, G, N to preserve and name exactly the same packet.*

No local obligation stores host suprema, selected mesh limits, quotient equality of functions, countable choice, open-cover compactness, or ambient Real completeness.

Proof. Carrier admission is projection from Definition 8838.1. The admitted packet exposes the interval row, source and refined partitions, lower and upper finite sums, the refinement ledger, the Darboux handoff, the Cauchy-readiness row, the Real seal, and the structural rows.

Partition refinement coherence is read from P, Q, M. The row Q is not a second unrelated partition; it is tied to P by the displayed finite cell-incidence map, and M is the only row that compares sum data across that map.

The Darboux handoff is then read through D. Since D sees only the paired finite lower and upper sums over the displayed refinement route, it cannot supply a host supremum, selected mesh limit, or quotient-function comparison.

Finally C, E, H, K, G, N carry the same finite route into a Real-facing Cauchy readback, transport it componentwise, replay the route, record provenance, and name the packet. The excluded host objects have no carrier coordinate, so the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DarbouxSumRefinementUp.DarbouxSumRefinementCarrier namecert obligations`

Theorem 8838.3 (Darboux-sum refinement Cauchy handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *carrier reaches its Cauchy-facing read only through the finite route*

$$\mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{LowerDarbouxSum}^{\uparrow} \longmapsto \mathsf{DarbouxSumRefinement}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer first reads the source partition P, the refined partition Q, the paired finite lower and upper rows L, U, the refinement monotonicity ledger M, and the Darboux handoff D; only then may it consume the Cauchy-readiness row C and terminal seal E. It receives no selected limiting partition, no host integral value, no quotient equality of functions, no countable-choice schedule, and no ambient completeness theorem.

Proof. Start with an accepted carrier $R = (I, P, Q, L, U, M, D, C, E, H, K, G, N)$. The carrier definition makes P the source dyadic partition and Q the displayed finite refinement of P.

Read the lower and upper finite rows through L, U and compare them only through M. Thus any refinement claim is a finite ledger over the displayed partition route, not a limiting mesh assertion.

Apply the Darboux handoff row D. It packages the paired finite sums over the common refinement route before a Real-facing consumer can reach the Cauchy-readiness row C.

Finally the terminal seal E is downstream of that route, and H, K, G, N only transport, replay, source, and name the same finite packet. Since the carrier omits selected limiting partitions, host integral values, quotient function equality, countable choice, and ambient completeness, the consumer cannot extract any of those endpoints. □

Lean4 ✓ `BEDC.Derived.DarbouxSumRefinementUp.DarbouxSumRefinementCarrier cauchy handoff`

Falsifiable prediction: Every accepted $\mathsf{DarbouxSumRefinement}^{\uparrow}$ read used by a Real or Completion consumer displays one source DyadicPartitionUp row, one refined DyadicPartitionUp row, lower and upper finite Darboux-sum rows, one refinement monotonicity ledger, one DarbouxSumUp handoff, one Cauchy-readiness row, one RealUp seal, and the structural rows H, K, G, N before it is treated as a Cauchy-facing refinement certificate.

Independence witness: darboux-sum, lower-darboux-sum, dyadic-partition, real: $\mathsf{DarbouxSumRefinement}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries a source partition, a refined partition, the finite cell-incidence map, lower and upper Darboux-sum rows, a refinement monotonicity ledger, Cauchy-readiness data, terminal Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxSumRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxSumRefinement}^{\uparrow}$ packet is generated from a RealUp interval row, a source DyadicPartitionUp row, a refined DyadicPartitionUp row with finite cell incidence, lower and upper Darboux-sum rows, a refinement monotonicity ledger, a DarbouxSumUp handoff, Cauchy-readiness data, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxSumRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8838.1, Definition 8838.2, and Definition 8838.3 seed the finite refinement route from Darboux sums over dyadic partitions to Real-facing Cauchy handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxSumRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host suprema or infima, selected limiting meshes, host integral values, quotient equality of functions, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition refinement coherence, lower and upper sum stability, Darboux handoff confinement, Real-facing Cauchy readiness, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.DarbouxSumRefinementUp.

Chapter 8839

A Concrete Naming Certificate for $\mathsf{RiemannRoch}^{\uparrow}$

Following Riemann and Roch, the $\mathsf{RiemannRoch}^{\uparrow}$ packet records the finite divisor, line-bundle, genus, and cohomology-dimension surface needed before an algebraic-curve consumer may read the classical balance law. It sits beside the manifold and chart discipline of chapter 96, the sheaf-interface route of chapter 151, the cohomology substrate of chapter 99, the unary index discipline of chapter 28, and the terminal real-name seal of chapter 38. The packet names a displayed Riemann–Roch balance route; it is not a host algebraic-geometry theorem, quotient divisor equality, selected basis of sections, hidden cohomology computation, or ambient completeness principle.

Definition 8839.1 (Riemann–Roch carrier)**.** *A* $\mathsf{RiemannRoch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, D, L, G, H, E, B, T, Q, P, N)$$

with the following displayed rows:

1. *C is the curve-presentation row, exposing the finite chart and overlap data used by the packet.*
2. *D is the divisor row, recording the displayed zeros, poles, and degree ledger read by the balance law.*
3. *L is the line-bundle row attached to the divisor presentation.*
4. *G is the genus row, recorded as a named integer datum rather than as a host classification theorem.*
5. *H is the cohomology-dimension row for the visible section and obstruction counts.*
6. *E is the exactness row that routes the dimension comparison through the displayed finite sequence boundary.*
7. *B is the balance row stating that the section count, obstruction count, divisor degree, and genus data are read in one Riemann–Roch ledger.*
8. *T records componentwise* hsame *transport for curve, divisor, line-bundle, genus, dimension, exactness, balance, replay, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from a Riemann–Roch request through $C, D, L, G, H, E, B, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RiemannRoch}^{\uparrow}$*, the curve, divisor, line-bundle, genus, cohomology-dimension, exactness, integer,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{RiemannRoch}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, D, L, G, H, E, B, T, Q, P, N$. It has no coordinate for quotient divisor equality, a selected basis of global sections, a hidden cohomology engine, an ambient algebraic-curve category, or a host proof of Riemann–Roch.

Theorem 8839.2 (Riemann–Roch NameCert obligations)**.** *For every accepted* $\mathsf{RiemannRoch}^{\uparrow}$ *carrier $R = (C, D, L, G, H, E, B, T, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannRoch}^{\uparrow}}$ *surface consists exactly of curve-presentation admission, divisor-ledger exposure, line-bundle attachment, genus-row exposure, cohomology-dimension exposure, exactness-boundary exposure, balance-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every algebraic-geometry consumer must read*

$$C, D, L, G, H, E, B, T, Q, P, N$$

before the Riemann–Roch packet is consumed.

Proof. Project the carrier of Definition 8839.1. The curve, divisor, line-bundle, genus, cohomology-dimension, exactness, and balance data occur only as displayed rows of the same finite packet.

Read the algebraic route in order. The curve row C supplies the presentation, D and L attach the divisor and line-bundle data, and G records the genus datum before any balance row is available.

The dimension row H and exactness row E are then consumed by the balance row B. Transport, replay, provenance, and local naming are the componentwise rows T, Q, P, N.

Since no carrier coordinate stores quotient divisor equality, a selected basis, a hidden cohomology computation, an ambient algebraic-curve category, or a host proof of Riemann–Roch, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{RiemannRoch}^{\uparrow}$ read that serves an algebraic-curve consumer must expose curve presentation, divisor ledger, line-bundle attachment, genus row, cohomology-dimension row, exactness boundary, balance row, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: manifold, sheaf, exactsequence, integer, real: $\mathsf{RiemannRoch}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries curve presentation, divisor ledger, line-bundle attachment, genus data, cohomology-dimension data, exactness, balance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRoch}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannRoch}^{\uparrow}$ packet is generated from a curve-presentation row, divisor row, line-bundle row, genus row, cohomology-dimension row, exactness row, balance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannRoch}^{\uparrow}}$, seed)

Scope closed: Definition 8839.1 and Definition 8839.2 seed the finite Riemann–Roch balance route over curve, divisor, line-bundle, genus, dimension, exactness, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannRoch}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient divisor equality, selected bases of sections, hidden cohomology computation, a host algebraic-geometry theorem, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked curve admission, divisor-ledger exactness, line-bundle attachment, genus exposure, cohomology-dimension exposure, exactness-boundary exposure, balance-row exactness, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRochUp.

Chapter 8840

A Concrete Naming Certificate for $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$

The $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ packet realizes the finite ProbeBundle lower-bound fold in the compact-uniform-continuity vision of chapter 9237. It sits after the compact metric source of chapter 129, the continuous-map rows of chapter 130, the ProbeBundle cover fold of chapter 7452, the rational carrier of chapter 37, and the uniform-modulus consumer of chapter 369. The packet names the finite positive-radius meet used by the compact-uniform route; it is not a selected Lebesgue number, host finite minimum theorem, open-cover compactness principle, countable choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 8840.1 (Compact finite radii meet carrier)**.** *A* $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, B, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the compact finite-net data used by the fold.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-continuity row supplying local positive radii for the inspected centers.*
3. *B is the* $\mathsf{ProbeBundle}^{\uparrow}$ *finite-bundle row listing the centers and local radius entries consumed by the meet.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *positivity row for the local radii and the requested output tolerance.*
5. *L is the finite lower-bound fold row, recording the displayed meet of the positive rational radii in B.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained only after K, F, B, R, L have been displayed.*
7. *H records componentwise* hsame *transport for compact-source, continuous-map, bundle, rational, meet, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform request through $K, F, B, R, L, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ProbeBundle}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteRadiiMeet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, R, L, U, H, C, P, N$. It has no coordinate for an external minimum operator, selected Lebesgue number, hidden open cover, infinite radius family, choice schedule, or ambient completeness principle.

Theorem 8840.2 (Compact finite radii meet NameCert obligations)**.** *For every accepted* $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ *carrier* $M = (K, F, B, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactFiniteRadiiMeet}^{\uparrow}}$ *surface consists exactly of compact-source admission, continuous-map local-radius exposure, finite ProbeBundle exposure, rational-positivity exposure, finite lower-bound meet exposure, UniformModulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-uniform consumer must read*

$$K, F, B, R, L, U, H, C, P, N$$

before it can consume the finite meet as a uniform-modulus datum.

Proof. Project the carrier of Definition 8840.1. The compact source, pointwise map rows, finite bundle, rational positivity, lower-bound meet, and uniform-modulus handoff are all displayed coordinates of the accepted packet.

The finite meet route is ordered. The compact row K supplies the finite-net source, F supplies local radii at the inspected centers, B lists exactly the bundled entries, and R records their positive rational status.

Only after those rows are visible may the fold row L be read. The handoff row U then exports the result to a uniform-modulus consumer, while H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Since the carrier contains no external minimum operator, selected Lebesgue number, hidden open cover, infinite radius family, choice schedule, or ambient completeness object, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ read that serves a compact-uniform consumer must expose compact-source rows, continuous-map local-radius rows, finite ProbeBundle rows, RatUp positivity rows, a finite lower-bound meet row, a UniformModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the consumer is served.

Independence witness: compactmetric, continuousmap, probe-bundle-cover-fold, rat, uniform-modulus: $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ is not bijective to any listed sibling because this packet binds compact finite-net data, pointwise local radii, a finite ProbeBundle, rational positivity, a lower-bound meet, UniformModulus handoff, transport, replay, provenance, and local naming into one compact-uniform fold carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFiniteRadiiMeet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactFiniteRadiiMeet}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map local-radius rows, a finite ProbeBundle row, rational positivity rows, a finite lower-bound meet row, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactFiniteRadiiMeet}^{\uparrow}}$, seed)

Scope closed: Definition 8840.1 and Definition 8840.2 seed the finite positive-radius meet used by compact-uniform-continuity lower-bound folds.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactFiniteRadiiMeet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected Lebesgue numbers, host finite minima, hidden open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, local-radius exposure, finite ProbeBundle coverage, rational-positivity exactness, finite meet exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactFiniteRadiiMeetUp.

Chapter 8841

A Concrete Naming Certificate for $\mathsf{CompactOperatorSpectrum}^{\uparrow}$

Following Riesz and Schauder compact-operator spectral theory, the $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ packet records the finite BEDC route by which compact-operator spectral reads are routed through displayed Banach, bounded-linear-operator, compact-operator, spectral-radius, and Fredholm-alternative certificates. It sits over the Banach carrier of chapter 90, the bounded operator row of chapter 1644, the compactness route of chapter 1952, the radius ledger of chapter 2352, and the Fredholm alternative surface of chapter 3425. The chapter names the finite spectral-route interface only; it does not close host spectra, selected eigenbases, quotient operator algebras, closed-range complements, or analytic Fredholm theory.

Definition 8841.1 (Compact-operator spectrum carrier)**.** *A* $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, L, K, R, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row naming the ambient complete normed carrier used by the operator read.*
2. *L is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the bounded operator action and its displayed norm control.*
3. *K is the* $\mathsf{CompactOperator}^{\uparrow}$ *row naming the compact-image or finite-approximation surface for the same operator.*
4. *R is the* $\mathsf{SpectralRadius}^{\uparrow}$ *ledger row recording the finite radius comparison data consumed by spectral-route readers.*
5. *F is the* $\mathsf{FredholmAlternative}^{\uparrow}$ *route row recording the kernel-range and alternative-facing readbacks available to downstream consumers.*
6. *H records componentwise* hsame *transport for Banach, bounded-operator, compact-operator, spectral-radius, Fredholm-route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$B, L, K, R \longmapsto F.$$

8. *P is the* Pkg *provenance over* $\mathsf{CompactOperatorSpectrum}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{FredholmAlternative}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOperatorSpectrum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, K, R, F, H, C, P, N$. It has no coordinate for a host spectrum, an eigenbasis selector, a quotient operator algebra, a closed-range complement, or an analytic Fredholm theorem.

Theorem 8841.2 (Compact-operator spectrum NameCert obligations)**.** *For every accepted* CompactOperatorSpect *carrier*

$$S = (B, L, K, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOperatorSpectrum}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Banach source, bounded-linear-operator row, compact-operator row, spectral-radius ledger, Fredholm-route row, transport, replay, provenance, and local naming rows;*
2. *Banach-source exposure, requiring every operator read to cite the same* $\mathsf{Banach}^{\uparrow}$ *row B before the bounded action is consumed;*
3. *bounded-action exactness, requiring every operator application and norm-control read to pass through the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row L;*
4. *compactness exposure, requiring spectral-route consumers to use the displayed* $\mathsf{CompactOperator}^{\uparrow}$ *row K rather than a host compactness predicate;*
5. *radius-ledger exactness, requiring every spectral-radius comparison to read the* $\mathsf{SpectralRadius}^{\uparrow}$ *row R;*
6. *Fredholm-route nonescape, requiring the* $\mathsf{FredholmAlternative}^{\uparrow}$ *row F to be consumed only after B, L, K, R have been displayed;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No obligation is discharged by host spectra, selected eigenbases, quotient operator algebras, closed-range complements, or analytic Fredholm theory.

Proof. Project the accepted carrier $S = (B, L, K, R, F, H, C, P, N)$. The visible rows are exactly the Banach source, bounded operator, compact-operator route, spectral-radius ledger, Fredholm-route row, transport, replay, provenance, and local naming data.

The first operator-facing reads are forced through B and L. A consumer has no independent bounded action unless it first cites the displayed Banach row and the bounded-linear-operator row.

The compact spectral route is then forced through K and R. The compact-operator row supplies the finite compactness surface, while the spectral-radius row supplies the radius comparison ledger.

Finally F is downstream from those rows, and H, C, P, N preserve, replay, source, and name the same finite packet. Since no excluded host object is a coordinate of S, the listed rows exhaust the local NameCert surface. □

Theorem 8841.3 (Compact-operator spectrum Fredholm route)**.** *Every Fredholm-facing consumer of an accepted* $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{Banach}^{\uparrow} \longmapsto \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{CompactOperator}^{\uparrow} \longmapsto \mathsf{SpectralRadius}^{\uparrow} \longmapsto \mathsf{FredholmAlternative}^{\uparrow}.$$

The route supplies a displayed compact-operator spectral interface over finite carrier rows; it does not select eigenbases, construct quotient operator algebras, choose closed-range complements, compute host spectra, or invoke analytic Fredholm theory.

Proof. Start with the obligation theorem Definition 8841.2. It places the Banach row B, bounded-linear-operator row L, compact-operator row K, and spectral-radius ledger R before the Fredholm route F.

The sibling certificates provide the stated handoff surface. The Banach chapter supplies the source carrier, the bounded-linear-operator chapter supplies the action row, the compact-operator chapter supplies compactness exposure, and the spectral-radius chapter supplies finite comparison data.

The FredholmAlternative consumer then reads only the displayed route row F. Transport, replay, provenance, and local naming preserve the same packet, so the Fredholm-facing surface receives the finite compact-operator spectral interface and none of the excluded analytic endpoints. □

Falsifiable prediction: A $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ consumer that reaches a Fredholm-facing spectral read without displaying the Banach source, bounded-linear-operator action, compact-operator row, spectral-radius ledger, Fredholm-route row, transport, replay, provenance, and local naming exits the accepted carrier.

Independence witness: banach, bounded-linear-operator, compact-operator, spectral-radius, fredholm-alternative: no listed sibling is bijective with $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ because this packet jointly binds Banach source, bounded action, compactness exposure, spectral-radius comparison, Fredholm route, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOperatorSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOperatorSpectrum}^{\uparrow}$ packet is generated from a Banach source row, BoundedLinearOperator action row, CompactOperator compactness row, SpectralRadius comparison ledger, FredholmAlternative route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOperatorSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8841.1, Definition 8841.2, and Definition 8841.3 seed the finite compact-operator spectral route over displayed Banach, bounded-operator, compact-operator, spectral-radius, Fredholm, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOperatorSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host spectra, selected eigenbases, quotient operator algebras, closed-range complements, or analytic Fredholm theory.

Upgrade path: Obligation closure requires checked carrier admission, Banach-source exposure, bounded-action exactness, compactness exposure, spectral-radius exactness, Fredholm-route nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactOperatorSpectrumUp.

Chapter 8842

A Concrete Naming Certificate for $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley's filter presentation of uniform spaces, the $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$ packet names the finite handoff from uniform Cauchy-filter data to the current Real completion route. It sits after the uniform Cauchy-filter surface of chapter 1694, the Cauchy-filter root of chapter 333, the uniform-completion boundary of chapter 1473, the stream window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet is a finite functorial routing surface; it is not a quotient of filters, a selected limit, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8842.1 (Uniform Cauchy-filter completion functor carrier)**.** *A* $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, C, E, S, R, D, L, H, K, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *source row, exposing the uniform-space, filter-base, entourage-Cauchy, stream-window, regular-rational, and completion-handoff surface.*
2. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *root row that records the underlying filter route before any completion-facing consumer reads a Real seal.*
3. *E is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row, read only after U and C have exposed their finite Cauchy-filter rows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to the completion request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the same displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for entourage requests, window budgets, and regular readback comparisons.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the uniform-filter, root-filter, uniform-completion, stream, readback, dyadic, Real-seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a uniform Cauchy-filter request through U, C, E, S, R, D, L, H.*

10. *P is the* Pkg *provenance over* UniformCauchyFilterCompletionFunctor$^{\uparrow}$, UniformCauchyFilter$^{\uparrow}$, CauchyFilter$^{\uparrow}$, UniformCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformCauchyFilterCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $U, C, E, S, R, D, L, H, K, P, N$. It has no coordinate for quotient-filter representatives, selected completion limits, host real equality, countable-choice schedules, hidden choice of subnet, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8842.2 (Uniform Cauchy-filter completion functor obligations)**.** *For every accepted* UniformCauchyFilterCompletionFunctor$^{\uparrow}$ *carrier*

$$F = (U, C, E, S, R, D, L, H, K, P, N),$$

the local NameCert$_{\text{UniformCauchyFilterCompletionFunctor}^{\uparrow}}$ *surface consists exactly of uniform Cauchy-filter admission through U, root Cauchy-filter admission through C, uniform-completion handoff through E, finite window exposure through S, regular rational readback through R, dyadic tolerance accounting through D, terminal Real sealing through L, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8842.1. The accepted packet exposes exactly the rows $U, C, E, S, R, D, L, H, K, P, N$.

The filter-facing part is forced through U and C. These rows display the uniform Cauchy-filter route and the root Cauchy-filter route before the uniform-completion handoff E is available.

The Real-facing part is then forced through S, R, D, L. Stream windows are read first, regular rational data are read from those windows, dyadic tolerances account for the comparison budgets, and the Real seal appears only after those finite rows are present.

Finally H, K, P, N transport, replay, source, and name the same packet. Since the carrier has no row for quotient-filter representatives, selected limits, host equality, choice schedules, hidden subnets, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8842.3 (Uniform Cauchy-filter completion functor Real route)**.** *Every Real-facing completion consumer of an accepted* UniformCauchyFilterCompletionFunctor$^{\uparrow}$ *packet factors through*

UniformCauchyFilter$^{\uparrow}$, CauchyFilter$^{\uparrow}$, UniformCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$.

The route converts uniform Cauchy-filter data into the current Real completion boundary by finite window readback and dyadic tolerance accounting. It exports no quotient filter, selected limit, countable-choice selector, hidden subnet, host real equality, or ambient completeness theorem.

Proof. Use Definition 8842.2. It places the uniform Cauchy-filter row, the root Cauchy-filter row, and the uniform-completion handoff before any Real-facing seal is visible.

Read the finite numeric part next. The StreamName row S, RegSeqRat row R, and DyadicRatCore row D are the only rows from which the completion request obtains a regular Real readback.

The terminal row L is therefore downstream of the displayed filter, completion, window, readback, and tolerance rows. Transport, replay, provenance, and local naming through H, K, P, N preserve the same route.

Thus every Real-facing completion consumer factors through the displayed finite functorial handoff and cannot extract a quotient filter, selected limit, choice selector, hidden subnet, host equality, or ambient completeness claim. □

Falsifiable prediction: An accepted $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed $\mathsf{UniformCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: uniform-cauchy-filter, cauchyfilter, uniformcompletion, streamname, regseqrat, dyadicratcore, real: $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds the uniform Cauchy-filter source, root Cauchy-filter route, uniform-completion handoff, finite stream windows, regular rational readback, dyadic tolerances, Real sealing, transport, replay, provenance, and local naming as one functorial completion handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}$ packet is generated from UniformCauchyFilter rows, CauchyFilter root rows, UniformCompletion handoff rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8842.1, Definition 8842.2, and Definition 8842.3 seed the finite route from uniform Cauchy-filter data to the Real completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilterCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable-choice selectors, hidden subnets, host real equality, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform Cauchy-filter exposure, root Cauchy-filter routing, uniform-completion handoff, finite window readback, dyadic tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyFilterCompletionFunctorUp.

Chapter 8843

A Concrete Naming Certificate for $\mathsf{MellinTransform}^{\uparrow}$

Following Mellin and Titchmarsh, the $\mathsf{MellinTransform}^{\uparrow}$ packet records the finite positive-real source, integrand-window, Gamma handoff, Real-seal, transport, replay, provenance, and local naming rows needed before an analytic consumer may read the classical transform route. It sits above the real-name seal of chapter 38, the stream schedule discipline of chapter 126, the regular rational readback of chapter 301, and the Gamma interface of chapter 75. The analytic motivation is the Gamma bridge of section 9358.7; this packet names the transform-facing finite data without importing host function equality, measure-theoretic choice, or an ambient integral object.

Definition 8843.1 (Mellin transform carrier)**.** *A* $\mathsf{MellinTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, I, G, R, T, C, P, N)$$

with the following displayed rows:

1. *S is the positive-real source row, exposing the* $\mathsf{Real}^{\uparrow}$ *domain name and the apartness-from-zero ledger used by the transform request.*
2. *W is the integrand-window row, recording the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of sampled positive-real windows.*
3. *I is the regular-integrand row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoints and dyadic tolerance ledgers on the displayed windows.*
4. *G is the Gamma-handoff row, naming the* $\mathsf{Gamma}^{\uparrow}$ *boundary data consumed by the transform-facing branch.*
5. *R is the Real-seal row that repacks accepted window sums as a* $\mathsf{Real}^{\uparrow}$ *output name.*
6. *T records componentwise* hsame *transport for source, window, integrand, Gamma handoff, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Mellin-transform request through S, W, I, G, R, T, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MellinTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MellinTransform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, I, G, R, T, C, P, N$. It has no coordinate for host function equality, selected measurable representatives, a Lebesgue integral, quotient stream equality, countable choice, or an ambient analytic completion.

Theorem 8843.2 (Mellin transform NameCert obligations)**.** *For every accepted* $\mathsf{MellinTransform}^{\uparrow}$ *carrier* $M = (S, W, I, G, R, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MellinTransform}^{\uparrow}}$ *surface consists exactly of positive-real source exposure, integrand-window exposure, regular-integrand readback, Gamma-handoff exposure, Real-seal output exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every transform consumer must read*

$$S, W, I, G, R, T, C, P, N$$

before the packet is consumed.

Proof. Project the carrier of Definition 8843.1. The source, window, integrand, Gamma handoff, and Real seal data occur only as displayed rows of the same finite packet.

The route first reads the positive-real source S and the finite schedule W. The regular-integrand row I then supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks on those windows, while G records the Gamma-facing boundary that the analytic spine may consume.

The Real-seal row R is downstream of S, W, I, G. Transport, replay, provenance, and local naming are the componentwise rows T, C, P, N.

Since no carrier coordinate stores host function equality, measurable representatives, a Lebesgue integral, quotient stream equality, countable choice, or ambient analytic completion, the listed rows exhaust the local NameCert obligations. □

Theorem 8843.3 (Mellin transform Gamma handoff)**.** *An accepted* $\mathsf{MellinTransform}^{\uparrow}$ *packet can hand data to the Gamma bridge only through the displayed Gamma row G together with the positive-real source row S, integrand-window row W, regular-integrand row I, and Real-seal row R. The handoff does not create a meromorphic black box, a hidden integral, or a selected analytic representative outside the finite carrier.*

Proof. Use the NameCert obligations of Definition 8843.2. They force every consumer to expose S, W, I, G, R before the transform packet is usable.

The Gamma row G is only a handoff row. It names the Gamma-facing boundary already isolated in section 9358.7, and its inputs remain the positive-real source, finite integrand windows, regular-rational readbacks, and Real seal displayed by this packet.

No additional analytic object is available to project from the carrier. Therefore the handoff is exactly the finite route through $S, W, I, G, R, T, C, P, N$ and not a hidden integral or selected representative. □

Falsifiable prediction: A MellinTransformUp packet whose Gamma handoff is read without the displayed positive-real source, integrand-window, regular-integrand, and Real-seal rows fails local NameCert consumption.

Independence witness: gamma, real, stream-name, regseqrat, gamma-zeta-factor-seal: $\mathsf{MellinTransform}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries positive-real source exposure, integrand windows, regular-rational readback, Gamma handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MellinTransform}^{\uparrow}}$**.**

Constructive story: A MellinTransform$^{\uparrow}$ packet is generated from a positive-real source row, finite integrand-window row, regular-integrand readback row, Gamma handoff row, Real-seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MellinTransform}\uparrow}$, seed)

Scope closed: Definition 8843.1, Definition 8843.2, and Definition 8843.3 seed the finite Mellin-transform route over source, window, integrand, Gamma handoff, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MellinTransform}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, measure-theoretic choice, a Lebesgue integral, meromorphic continuation, quotient stream equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, integrand-window exactness, regular-integrand readback, Gamma-handoff exactness, Real-seal exactness, transport, replay, provenance, and local naming for BEDC.Derived.MellinTransformU

Chapter 8844

A Concrete Naming Certificate for $\mathsf{VariationModulus}^{\uparrow}$

Following Jordan and Bishop–Bridges on bounded variation and regulated functions, the $\mathsf{VariationModulus}^{\uparrow}$ packet records a finite partition-modulus ledger for oscillation and Darboux-sum control. It sits after the bounded-variation surface of chapter 398, the regulated-function carrier of chapter 1906, the Darboux refinement packet of chapter 8838, the dyadic arithmetic core of chapter 291, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed finite partitions, oscillation budgets, variation bounds, refinement reads, transport, replay, provenance, and local naming rows; it does not import host supremum variation, measure theory, quotient equality of functions, selected mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8844.1 (VariationModulus carrier)**.** *A* $\mathsf{VariationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, F, P, O, B, D, Q, R, E, H, C, S, N)$$

with the following displayed rows:

1. *I is the interval-source row, sealed as a displayed* $\mathsf{Real}^{\uparrow}$ *interval endpoint ledger.*
2. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row whose inspected values are read through finite windows.*
3. *P is the finite partition row, listing ordered subinterval cells with dyadic endpoint data supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *O is the oscillation-bound row, assigning a regular-rational bound to each displayed partition cell.*
5. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *control row, recording the finite variation budget used by the same cells.*
6. *D is the* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *handoff row, naming how the partition and oscillation rows are consumed by upper/lower Darboux-sum refinement.*
7. *Q is the partition-modulus row, recording the finite rule that refines a requested tolerance into a displayed partition and cellwise oscillation budget.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the tolerance, oscillation, variation, and Darboux comparison data.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, F, P, O, B, D, Q, R have remained visible.*
10. *H records componentwise* hsame *transport for the interval, function, partition, oscillation, variation, Darboux, modulus, readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a regulated-function or Darboux consumer request through the same finite route.*
12. *S is the* Pkg *provenance over* $\mathsf{VariationModulus}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DarbouxSumRefinement}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, P, O, B, D, Q, R, E, H, C, S, N$. It has no coordinate for host total variation as a supremum, measure-theoretic variation, selected limiting partitions, quotient equality of functions, countable choice, or ambient real completeness.

Theorem 8844.2 (VariationModulus NameCert obligations)**.** *For every accepted* $\mathsf{VariationModulus}^{\uparrow}$ *carrier*

$$V = (I, F, P, O, B, D, Q, R, E, H, C, S, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure, cellwise oscillation-bound exposure, regulated-function source visibility, bounded-variation budget visibility, Darboux-sum refinement handoff, dyadic endpoint exactness, regular-rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, package provenance, and local naming. Every Darboux or regulated-function consumer must read the finite route*

$$I, F, P, O, B, D, Q, R \longmapsto E, H, C, S, N$$

before treating a partition-modulus estimate as public.

Proof. Project the accepted carrier of Definition 8844.1. The interval, regulated-function, partition, oscillation, variation, Darboux, modulus, and readback rows are displayed before the structural rows can be consumed.

The partition and endpoint data are finite. The row P lists the cells and reads their dyadic endpoints through $\mathsf{DyadicRatCore}^{\uparrow}$, while Q records how a requested tolerance selects the displayed partition-modulus row without selecting a limiting mesh.

The analytic estimates are also finite rows. The oscillation row O and bounded-variation row B provide the cellwise control, and the Darboux row D is the only route by which upper/lower refinement consumers may use those controls.

Finally R, E, H, C, S, N read back, seal, transport, replay, source, and name exactly the displayed route. Since the carrier has no coordinate for host supremum variation, measure-theoretic variation, quotient function equality, selected limits, countable choice, or ambient completeness, those principles are not exported by the local certificate. □

Theorem 8844.3 (VariationModulus Darboux-regulated handoff)**.** *Every* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *or* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer of an accepted* $\mathsf{VariationModulus}^{\uparrow}$ *packet receives only the displayed partition row, oscillation-bound row, bounded-variation row, partition-modulus row, dyadic endpoint row, regular-rational readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming data. The handoff may refine Darboux sums and regulated-function windows through those rows, but it exports no host supremum variation, measure-completion object, selected limiting partition, quotient equality of functions, or countable-choice schedule.*

Proof. Start with Definition 8844.2. A Darboux-facing consumer must read P, O, B, D, Q before a refinement estimate is available, while a regulated-function consumer must also keep the source row F visible.

The dyadic and regular-rational rows fix the arithmetic boundary of the packet. Endpoint and tolerance data pass through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is read.

The structural rows H, C, S, N preserve this same finite route under transport, replay, provenance, and local naming. No excluded host principle appears among the carrier coordinates, so every public handoff remains a finite partition-modulus handoff. □

Falsifiable prediction: An accepted $\mathsf{VariationModulus}^{\uparrow}$ consumer is refuted if it reaches a Darboux or regulated-function estimate without displaying the finite partition row, oscillation-bound row, bounded-variation row, partition-modulus row, dyadic endpoint row, regular-rational readback row, terminal RealUp seal, transport, replay, provenance, and local naming rows.

Independence witness: boundedvariation, regulated-function, darboux-sum-refinement, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{VariationModulus}^{\uparrow}$ because this packet jointly carries a finite partition-modulus ledger, oscillation budgets, bounded-variation control, Darboux refinement, dyadic endpoint arithmetic, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationModulus}^{\uparrow}$ packet is generated from displayed Real interval rows, RegulatedFunction source rows, finite partition rows, oscillation-bound rows, BoundedVariation budget rows, DarbouxSumRefinement handoff rows, DyadicRatCore endpoint rows, RegSeqRat readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8844.1, Definition 8844.2, and Definition 8844.3 seed the finite partition-modulus route for Darboux and regulated-function consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host supremum variation, measure-theoretic variation, selected limiting partitions, quotient equality of functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition exactness, oscillation-bound exposure, regulated-function source visibility, bounded-variation budget visibility, Darboux refinement handoff, dyadic endpoint exactness, regular-rational readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.VariationModulusUp.

Chapter 8845

A Concrete Naming Certificate for $\mathsf{TsirelsonSpace}^{\uparrow}$

Following Tsirelson's construction of the first Banach space containing no isomorphic copy of ℓ_p or c_0, the $\mathsf{TsirelsonSpace}^{\uparrow}$ packet records the finite BEDC surface on which the classical recursive norm can be inspected before any separation theorem is claimed. It sits after the vector-space carrier of chapter 47, the normed-space surface of chapter 1950, the Banach handoff of chapter 90, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is adjacent to the Banach–Saks route of chapter 7116, but it does not prove reflexivity separation, an infinite basis theorem, or exclusion of classical sequence spaces.

Definition 8845.1 (TsirelsonSpace carrier)**.** *A* $\mathsf{TsirelsonSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, N, V, R, Q, A, S, H, C, P, L)$$

with the following displayed rows:

1. *B is a finite basis-block window, carrying inspected block supports, admissible block order, and the finite coefficient rows used in the recursive norm read.*
2. *N is the recursive norm-budget window, recording the visible stage, the admissible averaging budget, and the remaining tolerance after each block read.*
3. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *handoff row for the finitely supported vector operations exposed by B.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational coefficient and tolerance schedules consumed by the norm budget.*
5. *Q is the* $\mathsf{NormedSpace}^{\uparrow}$ *handoff row, exposing the displayed norm value, triangle-budget row, scalar-budget row, and zero-vector row.*
6. *A is the* $\mathsf{Banach}^{\uparrow}$ *handoff row, recording the completion-facing Cauchy budget that may be read only after B, N, V, R, Q are present.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row for the norm values and Cauchy-budget endpoints.*
8. *H records componentwise* hsame *transport for B, N, V, R, Q, A, S and for the structural rows of the packet.*
9. *C records displayed* Cont *replay for basis-window inspection, recursive norm-budget replay, vector-space handoff, normed-space handoff, Banach handoff, and Real seal delivery.*

10. *P is the* Pkg *provenance over* $\mathsf{TsirelsonSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BanachSaks}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{TsirelsonSpace}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $B, N, V, R, Q, A, S, H, C, P, L$.

Theorem 8845.2 (TsirelsonSpace NameCert obligation surface)**.** *For every accepted* $\mathsf{TsirelsonSpace}^{\uparrow}$ *carrier*

$$T = (B, N, V, R, Q, A, S, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{TsirelsonSpace}^{\uparrow}}$ *surface consists exactly of finite basis-block admission through B, recursive norm-budget admission through N,* $\mathsf{VecSpace}^{\uparrow}$ *handoff through V,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{NormedSpace}^{\uparrow}$ *handoff through Q,* $\mathsf{Banach}^{\uparrow}$ *handoff through A,* $\mathsf{Real}^{\uparrow}$ *seal delivery through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through L. It exports no reflexivity separation theorem, no infinite basis theorem, no copy-exclusion theorem for ℓ_p or c_0, no host completion, no selected infinite block branch, and no ambient compactness principle.*

Proof. Project the carrier rows from Definition 8845.1. The finite basis-block row B and recursive norm-budget row N are the only rows carrying the Tsirelson-specific norm inspection.

Read the algebraic and analytic handoffs in order. The vector-space operations pass through V, the rational schedules pass through R, and the normed-space obligations pass through Q before any Banach-facing row A or Real seal S is available.

The structural rows H, C, P, L transport, replay, source, and name the same finite packet. Since no coordinate contains reflexivity separation, an infinite basis theorem, copy-exclusion data for ℓ_p or c_0, a selected infinite branch, or a host completion object, none is exported by the local NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{TsirelsonSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TsirelsonSpace}^{\uparrow}$ packet is generated from finite basis-block windows, recursive norm-budget windows, VecSpace, RegSeqRat, NormedSpace, Banach, and Real handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TsirelsonSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8845.1 and Definition 8845.2 bind the finite basis-block, recursive norm-budget, vector-space, normed-space, Banach, regular-rational, Real-seal, transport, replay, provenance, and local naming rows of the TsirelsonSpace packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TsirelsonSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim reflexivity separation, an infinite basis theorem, exclusion of ℓ_p or c_0, Lean verification, bridge checking, host completion, selected infinite branches, or ambient compactness.

Upgrade path: A higher closure grade requires Lean definitions for the displayed rows and checked obligations for the recursive norm-budget handoff without importing host Banach-space theorems.

Chapter 8846

A Concrete Naming Certificate for $\mathsf{BetaSignSelection}^{\uparrow}$

The $\mathsf{BetaSignSelection}^{\uparrow}$ packet realizes the beta-sign selection principle of Definition 9346.57 as finite NameCert data. It reads the centered Green-excess branch through chapter 8769, the phase-shell route through chapter 8811, and the window-six seed through chapter 8802. The packet records the sign of the denominator deformation before any physical running law is consumed; it is not a fitted infrared value, a continuous beta function, or a host analytic flow.

Falsifiable prediction: Every accepted $\mathsf{BetaSignSelection}^{\uparrow}$ read exposes the denominator-deformation sign row before treating a Zeckendorf window-six branch as physically running. A branch whose displayed sign raises the inverse coupling is rejected as the primary running direction.

Independence witness: Nearest siblings: ZeckendorfNormal, ZCarry, ZNormal, StaticGreenBoundaryFlux, ZeckendorfPhaseShell, and GoldenMeanShiftWindowSix. None is bijective to $\mathsf{BetaSignSelection}^{\uparrow}$ because this packet binds denominator sign, activation order, admissible branch tag, finite window-six provenance, transport, replay, and local naming into one selection surface.

Definition 8846.1 (Beta-sign selection carrier)**.** *A* $\mathsf{BetaSignSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, A, G, W, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the denominator-deformation row, displaying the inspected source denominator, target denominator, and monotone activation coordinate.*
2. *A is the beta-sign admissibility row, recording whether the displayed denominator deformation lowers inverse coupling along the activation interval.*
3. *G is the centered Green-excess row consumed from* $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$.
4. *W is the window-six beta-seed row consumed from* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$.
5. *Z is the Zeckendorf normal-and-carry row, naming ZeckendorfNormal, ZCarry, and ZNormal as the finite normal-word siblings that must be visible before a branch tag is accepted.*
6. *S is the selected-branch row, retaining the accepted local running seed and the refused walking-branch tag.*
7. *H records componentwise* hsame *transport for D, A, G, W, Z, S and for the structural rows of the packet.*

8. *C records displayed* Cont *replay for denominator inspection, sign selection, Green-excess read, window-six seed read, sibling normal-word read, and branch retention.*
9. *P is the* Pkg *provenance over* $\mathsf{BetaSignSelection}^{\uparrow}$, $\mathsf{StaticGreenBoundaryFlux}^{\uparrow}$, $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$, $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$, *ZeckendorfNormal, ZCarry, ZNormal,* BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaSignSelection}^{\uparrow}}$ *row.*

The Lean TasteGate surface for this packet will require `Nontrivial` *and* `FieldFaithful` *instances for the displayed carrier rows before the sign-selection classifier can be checked.*

Theorem 8846.2 (Beta-sign selection NameCert obligation surface)**.** *For every accepted* $\mathsf{BetaSignSelection}^{\uparrow}$ *carrier*

$$B = (D, A, G, W, Z, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BetaSignSelection}^{\uparrow}}$ *surface consists exactly of denominator-deformation admission through D, beta-sign admissibility through A, centered Green-excess exposure through G, window-six beta-seed exposure through W, ZeckendorfNormal–ZCarry–ZNormal sibling visibility through Z, selected-branch retention through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N. It exports no fitted infrared value, no continuous beta function, no host analytic flow, no uninspected walking branch, and no physical running claim without the displayed sign row.*

Proof. Project the carrier rows from Definition 8846.1. The denominator row D and sign row A are the only entries that decide whether a branch may be read as physically running.

Read the source siblings next. The Green-excess row G, window-six row W, and normal-word row Z display the finite Zeckendorf sources named by Definition 9346.57; the selected-branch row S can retain a running seed only after those rows are visible.

The transport, replay, provenance, and naming rows H, C, P, N preserve the same finite packet. Since no coordinate stores a fitted infrared value, a continuous beta function, a host analytic flow, or an uninspected walking branch, the NameCert surface exports only the displayed beta-sign selection route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BetaSignSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{BetaSignSelection}^{\uparrow}$ packet is generated from a denominator-deformation row, beta-sign admissibility row, Green-excess source, window-six beta seed, Zeckendorf sibling visibility, selected-branch retention, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BetaSignSelection}^{\uparrow}}$, seed)

Scope closed: Definition 8846.1 and Definition 8846.2 bind the finite denominator-sign, Green-excess, window-six beta-seed, Zeckendorf sibling, transport, replay, provenance, and local naming rows of the BetaSignSelection packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BetaSignSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a continuous beta function, a fitted infrared value, a host analytic flow, Lean verification, bridge checking, or a running-law conclusion without the displayed sign row.

Upgrade path: A higher closure grade requires Lean carrier rows plus Nontrivial and FieldFaithful instances for the displayed sign-selection classifier.

Chapter 8847

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$

Following Bishop–Bridges compact metric analysis, the $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ packet records the finite surface on which compact-uniform modulus data are restricted to a displayed subdomain and a displayed codomain read without importing open-cover compactness or a host function-equality principle. It sits beside the compact-uniform-continuity route of chapter 388, the uniform-modulus ledger of chapter 369, the compact-metric source of chapter 129, the continuous-map carrier of chapter 130, and the terminal real-name seal of chapter 38. The chapter names only the restriction carrier and its NameCert obligations; it does not assert an external modulus selector, a host equality of functions, an open-cover compactness theorem, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8847.1 (Compact uniform modulus restriction carrier)**.** *A* $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, U, D, W, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite compact-metric observations name the inspected domain window.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map on that source window, including only graph reads and pointwise continuity data already admitted by the local classifier.*
3. *U is the incoming* $\mathsf{UniformModulus}^{\uparrow}$ *row, recording the source-radius response for each displayed target tolerance request.*
4. *D is the restriction-domain row, naming the finite subdomain or finite compact-window inclusion on which the restricted modulus may be read.*
5. *W is the restricted source-window row, recording the compact observations from K that remain visible after D is applied.*
6. *M is the outgoing restricted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained by reading U only through D, W, F.*
7. *H records componentwise* hsame *transport for the compact source, continuous-map data, incoming modulus, restriction-domain row, restricted window, outgoing modulus, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$K, F, U, D, W \longmapsto M.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusRestriction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, D, W, M, H, C, P, N$. It has no coordinate for open-cover compactness, host function equality, a selected external modulus, quotient equality of real names, ambient real completeness, Lean verification, or bridge checking.

Theorem 8847.2 (Compact uniform modulus restriction NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ *carrier*

$$R = (K, F, U, D, W, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusRestriction}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure, continuous-map visibility, incoming uniform-modulus exposure, restriction-domain admission, restricted-window exposure, outgoing restricted-modulus production, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer therefore reads a restricted compact-uniform modulus only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{CompactUniformContinuity}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg}$$

Proof. Project the carrier of Definition 8847.1. The compact source row K, continuous-map row F, and incoming modulus row U are the only upstream analytic rows.

Read the restriction data through D and W. The restricted window W is a displayed subpacket of the compact observations already present in K, and the outgoing modulus M is read from U only after D, W, F remain visible.

Replay the structural rows H, C, P, N. They transport, replay, source, and name the same finite route from compact-uniform continuity to the restricted modulus surface.

Since the carrier has no coordinate for open-cover compactness, host function equality, an external modulus selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ read is refuted if a restricted-modulus consumer exports M without first displaying the compact source row K, continuous-map row F, incoming modulus row U, restriction-domain row D, restricted-window row W, and the structural rows H, C, P, N.

Independence witness: compact-uniform-continuity, uniformmodulus, compactmetric, continuousmap, real: none is bijective to $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ because this packet jointly carries compact source data, continuous-map evidence, an incoming modulus, a displayed restriction domain, a restricted compact window, an outgoing restricted modulus, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusRestriction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformModulusRestriction}^{\uparrow}$ packet is generated from compact-metric source data, ContinuousMap rows, an incoming UniformModulus row, a displayed restriction domain, a restricted source window, an outgoing restricted UniformModulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusRestriction}\uparrow}$, seed)

Scope closed: Definition 8847.1 and Definition 8847.2 seed the finite restriction route from compact-uniform modulus data to a displayed restricted-modulus consumer.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusRestriction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host function equality, an external modulus selector, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source exposure, continuous-map visibility, incoming modulus exposure, restriction-domain admission, restricted-window exposure, outgoing modulus production, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformModulusRestrictionUp.

Chapter 8848

A Concrete Naming Certificate for $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$

The $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$ packet realizes the prime-skew boundary-defect route of chapter 9249 as finite NameCert data. It consumes the RH-free zero-generation initiality boundary of chapter 8799, the zeta-zero type-factorization surface of chapter 8772, the finite prime-window gluing row of chapter 8765, the prime shadow interface of chapter 8791, and the zeta-zero catalogue of chapter 73. The packet records conditional evidence routing for prime-visible skew and boundary defects; it is not a proof of the Riemann Hypothesis, not a zero-location theorem, and not an analytic completion of the boundary faithfulness or zero-defect incompatibility premises.

Definition 8848.1 (Prime-skew boundary defect packet carrier)**.** *A* $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Z, R, S, B, I, F, X, H, C, Q, N)$$

with the following displayed rows:

1. *Z is the non-trivial zeta-zero source row, read through* $\mathsf{ZetaZeros}^{\uparrow}$ *and the zero-type factorization surface.*
2. *R is the bulk-representation row, recording the displayed map from zero packets into a finite boundary-readout carrier.*
3. *S is the prime-skew row, naming the finite prime-visible skew evidence extracted from displayed* $\mathsf{Prime}^{\uparrow}$ *channels.*
4. *B is the boundary-defect row, recording the finite boundary layer at which the prime-visible skew is observed as a defect.*
5. *I is the zero-defect incompatibility row, storing the conditional refutation interface between a represented non-trivial zero atom and the displayed defect row.*
6. *F is the boundary-faithfulness row, recording the conditional route from prime-skew evidence to boundary-defect evidence.*
7. *X is the refusal row, retaining failed attempts to read the packet as an unconditional RH proof, a global zero-location classifier, an analytic continuation theorem, or a completed boundary-faithfulness proof.*

8. *H records componentwise* hsame *transport for zero source, representation, prime skew, boundary defect, incompatibility, faithfulness, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an RH-facing audit request through Z, R, S, B, I, F, X, H.*
10. *Q is the* Pkg *provenance over* PrimeSkewBoundaryDefectPacket$^{\uparrow}$, RhFreeZeroInitiality$^{\uparrow}$, ZetaZeroTypeFactorizatior FinitePrimeWindowGluing$^{\uparrow}$, PrimeShadowInterface$^{\uparrow}$, ZetaZeros$^{\uparrow}$, Prime$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{PrimeSkewBoundaryDefectPacket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, R, S, B, I, F, X, H, C, Q, N$. It has no coordinate for an unconditional proof of RH*, a full zero-location classifier, completed boundary faithfulness, completed zero-defect incompatibility, host complex equality, or analytic continuation outside the displayed zeta-zero route.*

Theorem 8848.2 (Prime-skew boundary defect packet)**.** *For every accepted* PrimeSkewBoundaryDefectPacket$^{\uparrow}$ *carrier*

$$P = (Z, R, S, B, I, F, X, H, C, Q, N),$$

the local NameCert$_{\text{PrimeSkewBoundaryDefectPacket}^{\uparrow}}$ *surface consists exactly of zero-source admission, bulk-representation visibility, prime-skew visibility, boundary-defect visibility, zero-defect incompatibility routing, boundary-faithfulness routing, RH-proof refusal, componentwise transport, displayed replay, package provenance, and local naming. A consumer may use the packet only as conditional evidence routing from the prime-skew row to the boundary-defect row and then to the incompatibility row; no accepted read exports an RH proof or an off-line-zero exclusion without separately displaying the vision-side premises.*

Proof. Project the carrier of Definition 8848.1. The mathematical rows are Z, R, S, B, I, F, X, and the structural rows are H, C, Q, N.

The zero source and representation rows are read first. They bind the zeta-zero data to the displayed boundary-readout carrier without selecting a host complex representative or completing an analytic zero-location theorem.

The prime-skew and boundary-defect rows are conditional. Row S names the finite prime-visible skew evidence, row F routes that evidence to B, and row I records the incompatibility interface once the represented zero atom and defect row are both displayed.

Finally use the refusal row X and the structural rows H, C, Q, N. They preserve the packet as conditional evidence routing and retain any attempted shortcut to RH, global zero-location, analytic continuation, or completed boundary faithfulness as a refused read rather than a public conclusion. □

Theorem 8848.3 (Prime-skew boundary defect conditional nonescape)**.** *Every RH-facing read of an accepted* PrimeSkewBoundaryDefectPacket$^{\uparrow}$ *packet factors through the displayed route*

ZetaZeros$^{\uparrow}$, ZetaZeroTypeFactorization$^{\uparrow}$, Prime$^{\uparrow}$, FinitePrimeWindowGluing$^{\uparrow}$, PrimeShadowInterface$^{\uparrow}$, RhFreeZeroIn

before the prime-skew, boundary-defect, faithfulness, incompatibility, refusal, transport, replay, provenance, and local naming rows are public. The read exports no unconditional RH theorem, no global zero-location classifier, no completed proof of boundary faithfulness, no completed proof of zero-defect incompatibility, and no host analytic continuation claim.

Proof. Apply Definition 8848.2. It exposes the only public rows of the packet and keeps the evidence route downstream of the zero-source, representation, prime-skew, boundary-defect, faithfulness, and incompatibility rows.

The sibling sources supply the finite entry points. The zeta-zero and zero-type rows bind the zero packet, the prime and finite-prime-window rows bind the skew observation, the prime-shadow interface binds the boundary readout, and the RH-free initiality row keeps generated-zero no-circularity separate from fixed-RH proof claims.

The refusal row and structural rows close the boundary. Because no carrier coordinate stores an unconditional RH theorem, a global zero-location classifier, completed boundary faithfulness, completed zero-defect incompatibility, or analytic continuation outside the zeta-zero route, no consumer can extract those claims from the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$ consumer that exports an RH proof, off-line-zero exclusion, or global zero-location classifier without first displaying zero-source, representation, prime-skew, boundary-defect, faithfulness, incompatibility, refusal, transport, replay, provenance, and local naming rows is outside the carrier surface.

Independence witness: prime, zeta, complex, riemann-hilbert, rh-free-zero-generation-initiality: $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$ is not bijective to any listed source because it jointly carries prime-skew evidence, boundary-defect evidence, conditional faithfulness, conditional incompatibility, RH-proof refusal, transport, replay, provenance, and local naming while refusing any unconditional RH conclusion.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}$ packet is generated from zeta-zero source rows, bulk-representation rows, prime-skew rows, boundary-defect rows, zero-defect incompatibility rows, boundary-faithfulness rows, RH-proof refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8848.1, Definition 8848.2, and Definition 8848.3 seed the finite conditional evidence route for the RH prime-skew boundary-defect vision.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeSkewBoundaryDefectPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the Riemann Hypothesis, off-line-zero exclusion, global zero-location, completed boundary faithfulness, completed zero-defect incompatibility, analytic continuation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked zero-source admission, bulk-representation visibility, prime-skew visibility, boundary-defect visibility, faithfulness routing, incompatibility routing, RH-proof refusal, transport, replay, provenance, and local naming for BEDC.Derived.PrimeSkewBoundaryDefectPacketUp.

Chapter 8849

A Concrete Naming Certificate for $\mathsf{BishopRegularRealCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ packet records the finite admission criterion by which a regular rational sequence is allowed to serve as a constructive real name. It sits between the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The criterion is an admission route for Real-facing consumers; it is not a quotient construction of real numbers, a trichotomy principle, a selected limit operator, countable choice, or a proof of ambient Real completeness.

Definition 8849.1 (Bishop regular real criterion carrier)**.** *A* $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, Q, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only bounded windows of the candidate approximation stream.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the visible dyadic radii and comparison budget used by the regularity test.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, Q, D have been displayed.*
5. *A is the admission ledger row saying that the finite windows, regular readbacks, and dyadic tolerances match as one regular-real criterion packet.*
6. *H records componentwise* hsame *transport for stream windows, regular readbacks, dyadic tolerance rows, Real seal, admission ledger, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a criterion request through finite-window read, regular readback, dyadic tolerance comparison, admission, and Real sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularRealCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, A, H, C, P, N$. It has no coordinate for quotient reals, trichotomy, selected limits, countable choice, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8849.2 (Bishop regular real criterion derivation)**.** *Every accepted* $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ *carrier exposes exactly* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and a* $\mathsf{Real}^{\uparrow}$ *seal before any Real-facing read. More precisely, any Real-facing consumer of*

$$K = (W, Q, D, R, A, H, C, P, N)$$

must read W, then Q, then D, then the admission ledger A, and only then the seal row R together with H, C, P, N.

Proof. Project the accepted carrier of Definition 8849.1. Its public admission route begins with the finite-window row W and contains no completed real or host convergence row.

Read the regularity evidence next. The row Q is the regular rational readback attached to the displayed windows, and D records the dyadic tolerance comparisons used by the criterion.

The admission ledger A binds those rows before the Real seal is visible. Transport, replay, provenance, and local naming preserve the same order, so a consumer cannot detach R from W, Q, D, A.

Because the carrier has no coordinate for quotient reals, trichotomy, selected limits, countable choice, host convergence, or ambient completeness, the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows exhaust the Real-facing criterion. □

Falsifiable prediction: A Real-facing consumer that accepts a $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ packet without first displaying the $\mathsf{StreamName}^{\uparrow}$ finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the admission ledger refutes the carrier.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ is not bijective to any listed sibling because it binds finite stream windows, regular rational readbacks, dyadic tolerance comparison, an admission ledger, Real seal ordering, transport, replay, provenance, and local naming into one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularRealCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularRealCriterion}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, an admission ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularRealCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8849.1 and Definition 8849.2 seed the finite Bishop regular-real criterion route over StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularRealCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient reals, trichotomy, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, regular readback ordering, dyadic tolerance exactness, admission-ledger routing, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularRealCriterionUp.

Chapter 8850

A Concrete Naming Certificate for $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$

The $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ packet realizes the nested Busemann-level shadow route of chapter 9463 as finite observer-expansion NameCert data. It sits after the horosphere packet of chapter 8521, the Busemann boundary ledger of chapter 3732, the visual boundary route of chapter 4181, and the boundary-shadow packet of chapter 3813. The packet names a finite horoball-threshold filtration; it is not a host ideal-boundary quotient, a selected geodesic ray, a completed hyperbolic manifold, a global visual compactification, or an observer-substance claim.

Definition 8850.1 (Hyperbolic horoball filtration carrier)**.** *A* $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, B, V, S, L, T, H, C, P, N)$$

with the following displayed rows:

1. O *is the* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *row naming the boundary-anchored horosphere read.*
2. B *is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row recording the Busemann-level boundary ledger.*
3. V *is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row through which boundary visibility is read.*
4. S *is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the observer-shadow expansion read.*
5. L *is the finite horoball-threshold ledger, ordered by displayed Busemann levels.*
6. T *is the filtration transport row, requiring every shadow expansion row to preserve the visible threshold order.*
7. H *records componentwise* hsame *transport for horosphere, Busemann, visual-boundary, shadow, threshold-ledger, filtration-transport, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay through horosphere exposure, Busemann-level comparison, visual-boundary readback, boundary-shadow expansion, threshold inspection, and filtration transport.*
9. P *is the* Pkg *provenance over* $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$, $\mathsf{HyperbolicHorosphere}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}$ $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballFiltration}^{\uparrow}}$ *row.*

The carrier exports horoball-filtration data only through $O, B, V, S, L, T, H, C, P, N$*. It has no coordinate for host ideal-boundary quotients, selected geodesic rays, completed hyperbolic manifolds, global visual compactifications, or observer-substance dynamics.*

Theorem 8850.2 (Hyperbolic horoball shadow concretization)**.** *Every accepted* $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ *read factors the observer-shadow expansion route through*

$$\mathsf{HyperbolicHorosphere}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow},\ \mathsf{HyperbolicVisualBoundary}^{\uparrow},\ \mathsf{HyperbolicBoundaryShadow}^{\uparrow}$$

before the finite horoball-threshold ledger is exposed. A boundary-shadow expansion row is accepted only when the displayed Busemann-level filtration ledger and its transport row are visible in the same packet.

Proof. Project the carrier of Definition 8850.1. The geometric source rows are O, B, V, S, and the filtration data appear only as L and T.

Read the sibling route in order. The horosphere row fixes the boundary-anchored level surface, the Busemann boundary row records the level comparison, the visual boundary row supplies the visible boundary read, and the boundary-shadow row records the observer-expansion shadow.

The threshold ledger L then orders those reads by finite Busemann levels. The transport row T requires any shadow expansion consumer to preserve that threshold order rather than reading an unledgered boundary-shadow expansion.

Finally attach H, C, P, N. These rows preserve, replay, source, and name the same finite packet, leaving no coordinate for host boundary quotients, selected geodesic rays, completed manifolds, global visual compactifications, or observer-substance dynamics. □

The corresponding Lean surface for $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ is expected to expose `Nontrivial` and `FieldFaithful` instances once the carrier and field-faithfulness rows are formalized.

Falsifiable prediction: Any consumer exposing a boundary-shadow expansion row without the finite horoball-threshold ledger L and filtration transport row T refutes the $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ carrier.

Independence witness: hyperbolic-horosphere, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-boundary-shadow, hyperbolic-observer-expansion: the horosphere packet records level surfaces without the ordered horoball ledger; the Busemann boundary packet records boundary levels without the shadow filtration; the visual boundary packet records boundary visibility without threshold nesting; the boundary-shadow packet records expansion shadows without the finite ledger order; and observer-expansion packets can expose growth rows without this Busemann-level filtration. None is bijective to $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHoroballFiltration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicHoroballFiltration}^{\uparrow}$ packet is generated from HyperbolicHorosphere rows, HyperbolicBusemannBoundary rows, HyperbolicVisualBoundary rows, HyperbolicBoundaryShadow rows, a finite horoball-threshold ledger, filtration transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballFiltration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8850.1 and Definition 8850.2 seed the finite horoball-filtration route for observer-shadow expansion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHoroballFiltration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host ideal-boundary quotients, selected geodesic rays, completed hyperbolic manifolds, global visual compactifications, observer-substance dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked horosphere exposure, Busemann boundary comparison, visual boundary readback, boundary-shadow expansion, threshold-ledger exactness, filtration transport, provenance, and local naming for BEDC.Derived.HyperbolicHoroballFiltrationUp.

Chapter 8851

A Concrete Naming Certificate for $\mathsf{CompleteMetricSeparability}^{\uparrow}$

Following Bishop and Bridges on constructive metric spaces, the $\mathsf{CompleteMetricSeparability}^{\uparrow}$ packet records complete separability through displayed dense windows and regular Cauchy readback rather than through host countability or choice. It sits after the metric carrier of chapter 57, the complete-metric service of chapter 128, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the complete-separable synthesis surface of chapter 1490. The packet gives L10 Real and Completion consumers a named dense-window handoff before any PolishSpace consumer treats a complete metric source as separable.

Definition 8851.1 (Complete metric separability carrier)**.** *A* $\mathsf{CompleteMetricSeparability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, K, D, W, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row, carrying the point-history carrier, distance classifier, visible metric comparisons, and local metric naming data.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over the same metric carrier, exposing only the displayed Cauchy-window and limit-seal service available to downstream completion requests.*
3. *D is the countable dense-index row. It records finite inspected dense names, their index tags, and the admission rule that every requested ball must be answered through an indexed dense name rather than through an external choice function.*
4. *W is the finite dense-window row, displaying the requested center, radius, tolerance, inspected dense indices, and the first accepted witness window for the request.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-and-tolerance row, sealing the displayed metric distances and positive radii used by W.*
6. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row, recording the regular rational windows by which the dense witness and the completion row are compared.*
7. *E is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for dense names, Cauchy windows, and finite observation requests made by D, W, K, T.*

8. *H records componentwise* hsame *transport for M, K, D, W, R, T, E and for the structural rows of the packet.*
9. *C records displayed* Cont *replay for metric-ball inspection, dense-index lookup, finite dense-window replay, completion-service replay, StreamName readback, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricSeparability}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricSeparability}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $M, K, D, W, R, T, E, H, C, P, N$. It has no coordinate for host countability, selected infinite dense sequences outside the displayed index row, quotient equality of completions, open-cover compactness, or ambient real completeness.

Theorem 8851.2 (Complete metric separability NameCert obligations)**.** *For every accepted* CompleteMetricSepara *carrier*

$$S = (M, K, D, W, R, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricSeparability}^{\uparrow}}$ *surface consists exactly of metric carrier exposure through M, complete-metric service exposure through K, dense-index admission through D, finite dense-window coverage through W, Real-valued distance and tolerance sealing through R, regular rational readback through T, StreamName readback through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N. It exports no host countability principle, no selected global dense sequence beyond the displayed index row, no quotient completion equality, no open-cover compactness theorem, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier rows from Definition 8851.1. The metric row M fixes the distance surface, while the complete-metric row K supplies only the Cauchy-window and limit-seal service available on that same surface.

Read the separability-specific rows next. The dense-index row D records the only accepted source of dense names, and the finite dense-window row W records the requested ball, inspected indices, tolerance, and accepted witness window. The distance and tolerance values in W are sealed through R, while T and E expose the regular rational and stream readbacks used to inspect the witness.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores a host countability principle, a hidden global sequence selector, quotient equality of completions, open-cover compactness, or ambient real completeness, none is exported by the NameCert surface. □

Theorem 8851.3 (Complete metric separability density handoff)**.** *Let*

$$S = (M, K, D, W, R, T, E, H, C, P, N)$$

be an accepted $\mathsf{CompleteMetricSeparability}^{\uparrow}$ *carrier. Any downstream* $\mathsf{PolishSpace}^{\uparrow}$ *or completion consumer that requests a dense point inside a displayed metric ball must factor the request through the route*

$\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, D, W, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{CompleteMetri}}$

The handoff exposes the dense witness only as finite index-window evidence and does not supply a host enumeration, countable-choice selector, quotient completion representative, selected limit outside K, or open-cover compactness argument.

Proof. Start with a displayed downstream ball request. The metric carrier M supplies the center, radius comparison, and distance classifier, and the complete-metric row K remains available only as the completion service over that same carrier.

The dense request is then routed through D and W. The row D admits the inspected dense index, while W records the finite witness window and the comparison budget for the requested ball. The numeric side of the comparison is readable only through the Real seal R, the regular rational readback T, and the StreamName observations E.

Finally apply the structural rows H, C, P, N. They keep the metric, completion, dense-index, dense-window, readback, and seal rows aligned under transport and replay, and they expose only the local packet to the downstream consumer. A consumer therefore receives finite dense-window evidence, not an external enumeration or a choice-selected completion representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricSeparability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricSeparability}^{\uparrow}$ packet is generated from a MetricSpace row, CompleteMetric service row, countable dense-index row, finite dense-window row, Real-valued distance and tolerance row, RegSeqRat and StreamName readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricSeparability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8851.1, Definition 8851.2, and Definition 8851.3 seed the complete-metric separability route over metric, complete-metric, dense-index, dense-window, Real-seal, regular-rational, stream, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricSeparability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host countability, countable choice, a selected global dense sequence beyond the displayed index row, quotient completion equality, open-cover compactness, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dense-index exposure, finite dense-window coverage, complete-metric service nonescape, StreamName and RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompleteMetricSeparabilityUp.

Chapter 8852

A Concrete Naming Certificate for $\mathsf{TailDiameterCriterion}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{TailDiameterCriterion}^{\uparrow}$ packet records the finite criterion that a regular rational source is Cauchy exactly when every requested dyadic tolerance can be served by a displayed tail-diameter witness. It refines the tail witness of chapter 1746 and the real-facing gauge of chapter 5891, while depending on the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet supplies a criterion-level finite tail-diameter object for Real and Completion arguments; it does not assert quotient real equality, selected limits, trichotomy, or ambient Real completeness.

Definition 8852.1 (Tail-diameter criterion carrier)**.** *A* $\mathsf{TailDiameterCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, D, T, K, E, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *or stream source whose Cauchy behavior is being tested.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request.*
3. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *tail-bound row that exposes a threshold, two visible tail indices, and the dyadic diameter bound for D.*
4. *K is the criterion row saying that each accepted tolerance request is served by such a displayed tail-bound row.*
5. *E records the finite equivalence direction between the criterion row and the ordinary Cauchy tail-diameter reading of the same source.*
6. *R is the* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff row used only after S, D, T, K, E have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, tail-bound, criterion, equivalence, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a criterion request through $S, D, T, K, E, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{TailDiameterCriterion}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{RealTailDiameterGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TailDiameterCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, K, E, R, H, C, P, N$. It has no coordinate for quotient real equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 (statement) `BEDC.Derived.TailDiameterCriterionUp`

Theorem 8852.2 (Tail-diameter criterion NameCert obligations)**.** *For every accepted* $\mathsf{TailDiameterCriterion}^{\uparrow}$ *carrier $Q = (S, D, T, K, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TailDiameterCriterion}^{\uparrow}}$ *surface consists exactly of the finite carrier rows of Definition 8852.1 and the following obligations:*

1. *carrier admission for the source, tolerance, tail-bound, criterion, equivalence, handoff, transport, replay, provenance, and local naming rows;*
2. *tolerance exactness, requiring each criterion request to read the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D before any tail-bound row is consumed;*
3. *tail-bound exactness, requiring T to expose the threshold, two visible tail indices, and dyadic comparison row of chapter 1746;*
4. *criterion exactness, requiring K to bind the same source and tolerance rows used by T;*
5. *equivalence confinement, requiring E to express only the two finite readings of Cauchy tail-diameter data, not quotient real equality or selected limits;*
6. *Real-seal handoff confinement, requiring R to be downstream from the displayed criterion and compatible with chapter 5891;*
7. *componentwise transport, replay, provenance, and local naming for exactly the same rows.*

No tail-diameter criterion obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic rows are S, D, T, K, E, R, and the structural rows are H, C, P, N.

Read the tolerance route first. The dyadic request D is exposed before the tail-bound row T, so the threshold and visible tail-index comparison are tied to the requested tolerance.

The criterion row K then packages the universal tolerance reading as finite NameCert data. The equivalence row E compares this criterion only with the ordinary tail-diameter Cauchy reading of the same source and tolerance surface.

The handoff row R is downstream. It may serve the real-facing gauge or terminal real seal only after the source, tolerance, tail-bound, criterion, and equivalence rows have been displayed.

Finally, H, C, P, N preserve the same row list under transport, replay, provenance, and local naming. Since no excluded object occurs in the carrier, no additional NameCert obligation can be read from the packet. □

Theorem 8852.3 (Tail-diameter criterion Cauchy equivalence)**.** *Every accepted* $\mathsf{TailDiameterCriterion}^{\uparrow}$ *carrier presents the finite route*

$$S, D \longmapsto T \longmapsto K \longmapsto E \longmapsto H, C, P, N$$

between the ordinary Cauchy tail-diameter reading of S and the criterion that every displayed dyadic tolerance request is served by a matching tail-bound witness. The route exports only the finite rows of Definition 8852.1; it exports no quotient real equality, selected limit, trichotomy decision, countable-choice schedule, open-cover compactness witness, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or standard bridge.*

Proof. Start with the source and tolerance rows. The packet exposes S and D before any criterion claim is available.

Apply the tail-diameter witness route of chapter 1746. The row T supplies the threshold, two visible tail indices, and dyadic bound for the same source and tolerance surface.

The criterion row K records that this displayed service is available for each accepted dyadic request, while E records the finite comparison between that criterion and the ordinary tail-diameter Cauchy reading.

Transport, replay, provenance, and local naming preserve the same rows. Because the carrier contains no quotient, choice, trichotomy, compactness, completeness, or bridge coordinate, the equivalence route cannot export any of those objects. □

Theorem 8852.4 (Tail-diameter criterion Real-seal handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion consumer that uses an accepted* $\mathsf{TailDiameterCriterion}^{\uparrow}$ *carrier must consume the route*

$$S, D, T, K, E \longmapsto R \longmapsto H, C, P, N.$$

Thus the completion-facing handoff is mediated by the displayed source, dyadic tolerance, tail-bound, criterion, equivalence, real-gauge or real-seal row, transport, replay, provenance, and local naming rows. It does not supply quotient real equality, a selected limit, trichotomy, countable choice, ambient Real completeness, open-cover compactness, or a standard bridge.

Proof. Begin with a regular-rational or real-facing consumer. The consumer receives the finite carrier $Q = (S, D, T, K, E, R, H, C, P, N)$, so the only analytic rows initially available are the source, tolerance, tail-bound, criterion, equivalence, and handoff rows.

The handoff cannot precede the criterion. The row R is read after S, D, T, K, E, and therefore any real-facing use of the packet already carries the tail-diameter witness and the finite criterion comparison.

Use the real-gauge confinement of chapter 5891. The same finite stream-window, dyadic tolerance, regular rational readback, and real-seal discipline is preserved when the criterion packet is consumed by Real and Completion arguments.

The structural rows H, C, P, N then transport, replay, source, and name only this route. Since no excluded object is a carrier coordinate, the consumer cannot extract a quotient equality, selected limit, trichotomy row, choice schedule, completeness theorem, compactness witness, or bridge datum from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailDiameterCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailDiameterCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8852.1, Definition 8852.2, Definition 8852.3, and Definition 8852.4 seed the finite criterion packet linking displayed dyadic tolerance requests, tail-diameter witnesses, Cauchy readings, and Real-seal handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailDiameterCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, theorem-level Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, tail-bound exactness, criterion exactness, Cauchy-equivalence confinement, Real-seal handoff confinement, transport stability, replay, provenance, and local naming for BEDC.Derived.TailDiameterCriterionUp.

Constructive story: A $\mathsf{TailDiameterCriterion}^{\uparrow}$ packet is generated from a RegSeqRat or stream source, a DyadicRatCore tolerance request, a CauchyTailDiameter tail-bound row, a criterion row, a finite equivalence row, a RealTailDiameterGauge or Real seal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8853

A Concrete Naming Certificate for $\mathsf{DyadicLipschitzSlopePacket}^{\uparrow}$

Following Bishop and Bridges on constructive metric and real analysis, the $\mathsf{DyadicLipschitzSlopePacket}^{\uparrow}$ packet records a finite slope-bound witness for Lipschitz estimates whose input and output comparisons are read through dyadic rational data. It sits over the dyadic arithmetic core of chapter 291, the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the Lipschitz map carrier of chapter 410, and the dyadic modulus handoff of chapter 1864. The packet names the finite row where a displayed slope bound is accepted before a Real or Completion consumer treats a dyadic error estimate as a Lipschitz output bound.

Definition 8853.1 (Dyadic Lipschitz slope packet carrier)**.** *A* $\mathsf{DyadicLipschitzSlopePacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, M, R, L, U, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row carrying the inspected dyadic input difference, dyadic output allowance, positivity witness, and finite comparison ledger.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row exposing the source and target distance classifiers used by the slope comparison.*
3. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed distance and tolerance names.*
4. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row carrying the accepted map, source metric row, target metric row, and displayed Lipschitz constant.*
5. *U is the* $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ *row that turns the slope bound and dyadic input tolerance into the visible output tolerance budget.*
6. *Q is the slope-ratio ledger. It records the finite dyadic comparison asserting that the displayed output difference is bounded by the displayed slope row times the displayed input difference whenever the input row is positive.*
7. *B is the consumer-bound row read by Real and Completion estimates after D, M, R, L, U, Q have all been displayed.*
8. *H records componentwise* hsame *transport for the dyadic, metric, Real, Lipschitz, modulus, slope-ledger, consumer-bound, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a slope-bound request through $D, M, R, L, U, Q, B, H, P, N$.*
10. *P is the* Pkg *provenance over* DyadicLipschitzSlopePacket$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, LipschitzMap$^{\uparrow}$, DyadicLipschitzModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{DyadicLipschitzSlopePacket}^{\uparrow}}$ *row.*

The carrier contains no host derivative, quotient function equality, selected real limit, ambient completeness principle, trichotomy row, countable-choice selector, Lean verification credential, or standard bridge.

Theorem 8853.2 (Dyadic Lipschitz slope packet NameCert obligations)**.** *For every accepted* DyadicLipschitzSlopePacket$^{\uparrow}$ *carrier*

$$S = (D, M, R, L, U, Q, B, H, C, P, N),$$

the local NameCert$_{\text{DyadicLipschitzSlopePacket}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed dyadic ledger, metric rows, Real seal, Lipschitz map row, dyadic modulus row, slope-ratio ledger, consumer-bound row, componentwise transport, replay, provenance, and local naming row; classifier exactness through those same rows; slope-ratio admission through Q; dyadic modulus handoff through U; Real and Completion consumer exposure through B; and transport and replay stability through H, C, P, N.*

Proof. Project the accepted packet to its displayed rows. The dyadic ledger D, metric row M, Real seal R, Lipschitz row L, and modulus row U are the only upstream analytic data available to the local name.

Read the slope-specific rows next. The ratio ledger Q records the finite dyadic comparison, and the consumer-bound row B is reached only after D, M, R, L, U, Q are displayed. Thus the slope witness is a finite comparison surface rather than a derivative or selected limit.

Finally apply the structural rows H, C, P, N. They transport, replay, source, and name exactly the same displayed packet. Since the carrier has no coordinate for host derivatives, quotient function equality, selected limits, ambient completeness, trichotomy, choice, or bridge credentials, the listed rows exhaust the local NameCert obligations. □

Closure status for NameCert$_{\text{DyadicLipschitzSlopePacket}^{\uparrow}}$.

Constructive story: A DyadicLipschitzSlopePacket$^{\uparrow}$ packet is generated from a DyadicRatCore ledger, Metric rows, a Real seal, a LipschitzMap row, a DyadicLipschitzModulus row, a finite slope-ratio ledger, a Real and Completion consumer-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicLipschitzSlopePacket}^{\uparrow}}$, seed)

Scope closed: Definition 8853.1 and Definition 8853.2 seed the finite slope-bound route from dyadic comparison data through metric, Real, Lipschitz, and dyadic-modulus rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicLipschitzSlopePacket}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host derivatives, quotient function equality, selected limits, ambient Real completeness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, slope-ratio admission, dyadic modulus handoff, Real and Completion consumer exposure, transport, replay, provenance, and local naming for BEDC.Derived.DyadicLipschitzSlopePacketUp.

Chapter 8854

A Concrete Naming Certificate for $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$

Following Newman and Takahashi-style finite diamond reasoning for terminating rewrite systems, the $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$ packet records the candidate-mediated local-join budget used by the MetaCIC normalization audit route. It sits beside the normalization budget of chapter 755, the candidate-mediated strong-normalization handoff of chapter 5163, and the candidate diamond budget of chapter 6552. The packet names only finite candidate evidence, local two-front reductions, join budget, no-infinite-reduction readback, transport, replay, provenance, and local naming rows; it does not assert full confluence, unconditional normalization, or subject-reduction closure.

Definition 8854.1 (Finite normalization diamond carrier)**.** *A* $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, E, S, A, P_i, A_p, L, J, Z, T, C, G, N)$$

with the following displayed rows:

1. *K is the candidate-evidence row, carrying the reducibility-candidate data used by the closed term under inspection.*
2. *E is the closed-candidate row, recording closedness of the type and term together with the candidate evidence consumed by the normalization route.*
3. *S is the candidate-set membership row, exposing membership, soundness, and beta-closure evidence when the route is read through a candidate set.*
4. *A is the Pi-adequacy row for binder and function evidence.*
5. *P_i is the application-adequacy row for applying a Pi-side candidate to a displayed argument candidate.*
6. *A_p is the local two-front reduction row, recording the two finite one-step or bounded-step frontiers whose endpoints must be compared.*
7. *L is the local join-budget row, listing the bounded join witnesses and the residual rows inspected by the local diamond check.*
8. *J is the accepted join row, present only when the two-front endpoints are joined inside the displayed budget.*
9. *Z is the no-infinite-reduction readback row exported from the candidate-mediated strong-normalization surface.*

10. *T records componentwise* hsame *transport for the candidate, closed-candidate, candidate-set, adequacy, two-front, join-budget, accepted-join, no-infinite-reduction, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay for the same candidate-mediated diamond route.*
12. *G is the* Pkg *provenance over* $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$*, the MetaCIC candidate and adequacy rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteNormalizationDiamond}^{\uparrow}}$ *row.*

The carrier has no coordinate for global Church–Rosser confluence, unconditional normalization of every closed typed term, erased candidate evidence, subject-reduction closure, quotient syntax equality, oracle reduction, Lean verification, or a standard bridge.

Theorem 8854.2 (Finite normalization diamond NameCert obligations)**.** *For every accepted* $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$ *carrier*

$$F = (K, E, S, A, P_i, A_p, L, J, Z, T, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteNormalizationDiamond}^{\uparrow}}$ *surface consists exactly of candidate evidence, closed-candidate rows, candidate-set membership, Pi adequacy, application adequacy, local two-front reductions, join budget, accepted local join, no-infinite-reduction readback, componentwise transport, displayed replay, provenance, and local naming. Every normalization-diamond read must keep the candidate rows visible and must expose the join as a bounded local witness rather than as full confluence or unconditional normalization.*

Proof. Project the carrier rows. Candidate evidence K, closed-candidate evidence E, and candidate-set membership S are the only sources from which strong-normalization and no-infinite-reduction reads may be obtained.

Read the adequacy rows A and P_i before inspecting any application frontier. They ensure that the local route remains candidate-mediated through Pi and application cases instead of becoming componentwise termination of unrelated parts.

The two-front row A_p, join budget L, and accepted join row J then record the finite diamond check. The no-infinite-reduction row Z is read from the same candidate-mediated surface, and the structural rows T, C, G, N transport, replay, source, and name exactly that finite packet.

Since the carrier has no row for global confluence, unconditional normalization, erased candidate evidence, subject-reduction closure, quotient syntax equality, oracle reduction, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNormalizationDiamond}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteNormalizationDiamond}^{\uparrow}$ packet is generated from candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy rows, a local two-front reduction row, a bounded join row, a no-infinite-reduction readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteNormalizationDiamond}^{\uparrow}}$, seed)

Scope closed: Definition 8854.1 and Definition 8854.2 seed the finite candidate-mediated local-join route for the MetaCIC normalization audit map.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteNormalizationDiamond}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full confluence, unconditional normalization, erased candidate evidence, subject-reduction transport, quotient syntax equality, oracle reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked candidate evidence, closed-candidate rows, candidate-set membership, Pi and application adequacy, local two-front reductions, join budget, no-infinite-reduction readback, transport, replay, provenance, and local naming for BEDC.Derived.FiniteNormalizationDiamondUp.

Chapter 8855

A Concrete Naming Certificate for $\mathsf{DualCone}^{\uparrow}$

Following Rockafellar's finite-dimensional convex analysis and the Boyd–Vandenberghe treatment of convex cones, the $\mathsf{DualCone}^{\uparrow}$ packet records the polar-support route for a displayed cone before any separation theorem or analytic closure principle is read. It sits over the convex membership rows of chapter 208, the vector-space boundary of chapter 47, the linear functional rows of chapter 48, the scalar seal of chapter 38, and the finite primal-dual packet of chapter 321. The packet names the dual cone as finite support-inequality data, not as Hahn–Banach, strong separation, a host quotient space, or analytic closedness.

Definition 8855.1 (Dual-cone carrier)**.** *A* $\mathsf{DualCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (V, C, L, R, F, H, K, P, N)$$

whose rows are displayed as follows: V *is the* $\mathsf{VecSpace}^{\uparrow}$ *source row,* C *is the* $\mathsf{ConvexSet}^{\uparrow}$ *cone row inside* V*,* L *is the* $\mathsf{LinMap}^{\uparrow}$ *functional row,* R *is the* $\mathsf{Real}^{\uparrow}$ *scalar comparison row for nonnegative support values,* F *is the* $\mathsf{FenchelDuality}^{\uparrow}$ *provenance row that records the adjacent polar-support reading,* H *is componentwise* hsame *transport for* V, C, L, R, F*,* K *is displayed* Cont *replay,* P *is* Pkg *provenance over* $\mathsf{DualCone}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FenchelDuality}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, and* N *is the local* $\mathsf{NameCert}_{\mathsf{DualCone}^{\uparrow}}$ *row. The classifier compares accepted packets only through these rows and has no coordinate for Hahn–Banach extension, strong separation, host quotient spaces, analytic closedness, or an ambient topological dual.*

Theorem 8855.2 (Dual-cone NameCert obligations)**.** *For every accepted* $\mathsf{DualCone}^{\uparrow}$ *carrier* $D = (V, C, L, R, F, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DualCone}^{\uparrow}}$ *surface consists exactly of vector-space exposure through* V*, cone-membership exposure through* C*, linear-functional exposure through* L*, Real-valued support comparison through* R*, polar-support provenance through* F*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the rows of Definition 8855.1. The vector, cone, functional, and scalar comparison data are the only mathematical rows stored by the packet.

The polar-support row F is read only as provenance from the finite Fenchel-duality sibling. It does not supply a separating hyperplane theorem, a topological dual completion, or an analytic closure assertion.

The structural rows H, K, P, N transport, replay, source, and name the same displayed data. Since no other coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 8855.3 (Dual-cone polar-support route)**.** *Every polar-support read of an accepted* $\mathsf{DualCone}^{\uparrow}$ *packet must first expose the* $\mathsf{ConvexSet}^{\uparrow}$ *cone row, the* $\mathsf{VecSpace}^{\uparrow}$ *source row, the* $\mathsf{LinMap}^{\uparrow}$ *functional row, the* $\mathsf{Real}^{\uparrow}$ *comparison row, and the* $\mathsf{FenchelDuality}^{\uparrow}$ *provenance row. The route may be consumed by finite primal-dual chapters, but it exports no Hahn–Banach theorem, no strong separation theorem, no host quotient space, no analytic closedness statement, and no ambient topological-dual object.*

Proof. Read the carrier rows C, V, L, R, F in order. The cone is inspected as displayed convex data in a displayed vector-space boundary, and a dual read is admitted only through a displayed linear functional with a Real-valued support comparison.

Now compare with chapter 321. The Fenchel row records the finite primal-dual bridge that may consume the support comparison, while the present packet keeps the cone-dual route at the level of named rows.

The carrier has no row for extension, separation, quotienting, analytic closure, or a completed topological dual. Thus every consumer remains on the finite polar-support route. □

Falsifiable prediction: Any accepted $\mathsf{DualCone}^{\uparrow}$ consumer that invokes Hahn–Banach, strong separation, host quotient spaces, analytic closedness, or an ambient topological dual before reading $V, C, L, R, F, H, K, P, N$ refutes the carrier surface of Definition 8855.1.

Independence witness: convexset, vecspace, linearmap, real, fenchelduality: $\mathsf{DualCone}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries cone membership, vector-space source, linear-functional support, Real comparison, Fenchel polar-support provenance, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DualCone}^{\uparrow}}$.

Constructive story: A $\mathsf{DualCone}^{\uparrow}$ packet is generated from a VecSpace source row, a ConvexSet cone row, a LinearMap functional row, a Real support-comparison row, a FenchelDuality polar-support provenance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DualCone}^{\uparrow}}$, seed)

Scope closed: Definition 8855.1, Definition 8855.2, and Definition 8855.3 seed the finite dual-cone NameCert surface over $V, C, L, R, F, H, K, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DualCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Hahn–Banach, strong separation, host quotient spaces, analytic closedness, ambient topological duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cone exposure, functional exposure, support comparison, polar-support provenance, transport, replay, provenance, and local naming for BEDC.Derived.DualConeUp.

Chapter 8856

A Concrete Naming Certificate for $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$

Following Munkres and Engelking on finite refinements of covers, the $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$ packet records the finite refinement-order surface through which compact metric cover data are routed to dyadic radii and uniform-modulus consumers. It sits beside the compact metric certificate of chapter 129, the metric carrier of chapter 57, the dyadic interval net of chapter 5966, the finite Lebesgue-number route of chapter 452, the uniform modulus handoff of chapter 369, and the terminal real seal of chapter 38. The packet names only displayed finite cover nodes, refinement arrows, incidence witnesses, dyadic radius ledgers, transport, replay, provenance, and local naming; it does not import open-cover compactness, a choice-selected subcover, quotient equality of covers, ambient completeness, or an independent $\mathsf{Real}^{\uparrow}$ carrier.

Definition 8856.1 (FiniteCoverOrderNetwork carrier)**.** *A* $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, O, I, D, U, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the cover-node row, listing the accepted finite cover names, the parent* $\mathsf{CompactMetric}^{\uparrow}$ *source, and the local* $\mathsf{Met}^{\uparrow}$ *probes visible at each node.*
2. *O is the refinement-order row, recording the finite preorder of displayed refinements and the named source and target cover nodes of each refinement arrow.*
3. *I is the incidence row, storing the pointwise cover-membership and refinement-containment witnesses used by order consumers.*
4. *D is the dyadic radius ledger, binding each cover node to its* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *radius data and to the finite* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-bound checks that justify refinement steps.*
5. *U is the uniform-modulus handoff row, available only after V, O, I, D have been displayed and only for consumers of the named* $\mathsf{UniformModulus}^{\uparrow}$ *route.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the real-valued radius readbacks displayed by D and U.*
7. *H stores componentwise* hsame *transport for cover nodes, refinement arrows, incidence witnesses, dyadic ledgers, uniform-modulus handoff, real seals, provenance, and local naming.*
8. *C stores the displayed* Cont *replay for the same finite rows.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverOrderNetwork}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, O, I, D, U, R, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient equality of cover families, countable choice, ambient metric completeness, or an independent real carrier.

Theorem 8856.2 (FiniteCoverOrderNetwork NameCert obligations)**.** *For every accepted* FiniteCoverOrderNetwork *carrier*

$$Q = (V, O, I, D, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverOrderNetwork}^{\uparrow}}$ *surface consists exactly of carrier admission for the cover-node row V, refinement-order row O, incidence row I, dyadic radius ledger D, uniform-modulus handoff U,* $\mathsf{Real}^{\uparrow}$ *seal R, componentwise transport H, displayed replay C, package provenance P, and local naming N.*

Proof. Project the carrier of Definition 8856.1. The finite data visible to the classifier are precisely the cover nodes V, refinement order O, incidence witnesses I, dyadic ledger D, uniform-modulus handoff U, and real seal R.

The order obligation is forced through V, O, I. A consumer cannot compare two cover nodes until both the displayed refinement arrow and the incidence witnesses justifying its containment reading have been exposed.

The radius obligation is separate. The row D supplies the dyadic interval-net and finite Lebesgue-number entries used by refinement steps, while U and R make only the named uniform-modulus and real-valued radius readbacks available downstream.

Finally read H, C, P, N. These rows transport, replay, prove provenance, and name the same finite packet. Since no arbitrary open cover, selected subcover, quotient cover equality, countable choice, ambient completeness, or independent real carrier is a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8856.3 (FiniteCoverOrderNetwork order readback)**.** *Every compact-metric or uniform-modulus consumer of an accepted* $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$ *packet must read the refinement network through the ordered rows*

$$V, O, I, D, U, R, H, C, P, N$$

before treating a cover refinement as a radius-controlled finite route. The readback supplies a finite refinement order with dyadic and real radius seals; it does not supply open-cover compactness, a chosen subcover, quotient equality of covers, countable choice, ambient completeness, or an independent $\mathsf{Real}^{\uparrow}$ *carrier.*

Proof. Start with the cover comparison. The rows V, O, I display the finite cover nodes, the refinement arrows, and the incidence witnesses, so the order readback has no access to an unnamed cover relation.

Next read the radius evidence. The dyadic ledger D records the interval-net and finite Lebesgue-number checks attached to the refinement arrows, and the rows U, R expose the uniform-modulus handoff and terminal real radius seals only after that ledger is visible.

It remains to package the route. Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed rows and add no global compactness theorem or selected subcover. Thus every accepted consumer reads exactly the finite refinement-order surface named by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverOrderNetwork}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverOrderNetwork}^{\uparrow}$ packet is generated from finite cover nodes, a refinement-order row, incidence witnesses, a dyadic radius ledger, a uniform-modulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverOrderNetwork}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8856.1, Definition 8856.2, and Definition 8856.3 seed the finite cover refinement-order carrier through compact metric cover nodes, metric incidence, dyadic interval nets, finite Lebesgue-number rows, uniform-modulus handoff, and Real radius readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverOrderNetwork}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, quotient equality of cover families, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement-order exactness, incidence coverage, dyadic radius-ledger exposure, uniform-modulus routing, Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverOrderNetworkUp.

Chapter 8857

A Concrete Naming Certificate for $\mathsf{DirectMethodMinimizer}^{\uparrow}$

Following Tonelli's existence theorem, Weierstrass compactness arguments, and the constructive accounts of the direct method in Dacorogna and Evans, the $\mathsf{DirectMethodMinimizer}^{\uparrow}$ packet records the finite route by which a variational minimum is read from compact sublevel data and lower-semicontinuity evidence. It sits after the compact finite-window discipline of chapter 129, the lower-semicontinuity boundary of chapter 1536, the variational convergence surface of chapter 5957, the Euler–Lagrange consumer of chapter 318, the regular rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet names a displayed minimizer read; it does not import a host choice of minimizer, quotient equality of functionals, selected convergent subsequences, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8857.1 (DirectMethodMinimizer carrier)**.** *A* $\mathsf{DirectMethodMinimizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, K, L, S, F, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric or function-space source row on which admissible configurations are inspected.*
2. *K is either a* $\mathsf{CompactMetric}^{\uparrow}$ *finite compactness-window row or a finite compact sublevel row displaying the compact part of the admissible surface.*
3. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph or liminf row for the inspected functional.*
4. *S is the coercive sublevel ledger, listing the displayed level, admissible window, and compact sublevel dependency actually used by the packet.*
5. *F is the finite functional-value window ledger for the minimizing comparison.*
6. *M is the minimizer-read row, exposed only after K, L, S, F have all been displayed.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational value comparisons attached to F and M.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *value seal row for the displayed minimum value.*
9. *H records componentwise* hsame *transport for the source, compact sublevel, lower-semicontinuity, coercive ledger, value window, minimizer read, readback, and seal rows.*
10. *C records displayed* Cont *replay from a minimizer request through the compact sublevel and lower-semicontinuity route.*

11. *P is the* Pkg *provenance over* $\mathsf{DirectMethodMinimizer}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{GammaConvergence}^{\uparrow}$, $\mathsf{EulerLagrange}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectMethodMinimizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, L, S, F, M, Q, E, H, C, P, N$. It has no coordinate for a host minimizer choice, arbitrary open-cover compactness, selected subsequence extraction, quotient equality of functionals, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8857.2 (DirectMethodMinimizer NameCert obligations)**.** *For every accepted* $\mathsf{DirectMethodMinimizer}^{\uparrow}$ *carrier $D = (X, K, L, S, F, M, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirectMethodMinimizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the source row, compact sublevel row, lower-semicontinuity row, coercive sublevel ledger, functional-value windows, minimizer-read row, regular rational readback, terminal real-value seal, componentwise transport, displayed replay, provenance, and local naming.*

Every minimizer-facing consumer must therefore read the finite route

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{LowerSemicontinuous}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

through the rows $K, L, S, F, M, Q, E, H, C, P, N$. The exported route is a NameCert boundary for a displayed direct-method minimum, not a host selection principle, quotient functional equality, selected subsequence theorem, or ambient completeness result.

Proof. Project the accepted carrier of Definition 8857.1. The analytic rows are X, K, L, S, F, M, Q, E, and the structural rows are H, C, P, N.

Carrier admission first exposes the admissible source row X and the compact sublevel row K. The coercive ledger S is read against the same compact window, so the packet has no separate path to open-cover compactness or a selected subsequence.

The functional-value and lower-bound data are then read through L, F. Lower semicontinuity supplies only the displayed epigraph or liminf row, and the value windows supply only the comparisons listed in the packet.

Finally the minimizer-read row M is exposed through Q and sealed by E. Transport, replay, provenance, and local naming preserve that same displayed route, so no coordinate can add a host minimizer choice, quotient functional equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 8857.3 (DirectMethodMinimizer compact-sublevel route)**.** *Every minimizer read exported by an accepted* $\mathsf{DirectMethodMinimizer}^{\uparrow}$ *packet factors through the displayed route*

$$X, K, S \longmapsto L, F \longmapsto M, Q, E, H, C, P, N.$$

In particular, the read is public only after the compact sublevel row and lower-semicontinuity row have both been displayed. It does not factor through host choice of a minimizer, ambient completeness of $\mathsf{Real}^{\uparrow}$*, quotient equality of functionals, selected recovery sequences from* $\mathsf{GammaConvergence}^{\uparrow}$*, or an* $\mathsf{EulerLagrange}^{\uparrow}$ *necessary-condition consumer.*

Proof. Start from the NameCert obligations of Definition 8857.2. They require any minimizer-facing consumer to expose the compact sublevel row K and coercive ledger S before the minimizer-read coordinate M can be inspected.

Next read the lower-semicontinuity row L and the functional-value windows F. These rows provide the direct-method comparison surface: compactness supplies the admissible sublevel boundary, while lower semicontinuity supplies the displayed lower-bound handoff.

The readback rows Q, E then present only the rational comparison and terminal real-value seal attached to that finite route. The structural rows H, C, P, N transport, replay, source, and name the same packet.

Thus the public minimizer read factors through compact sublevel and lower-semicontinuity data. Gamma-convergence and Euler–Lagrange packets may consume or compare the result, but they do not supply the existence route inside this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DirectMethodMinimizer}^{\uparrow}$ consumer that reaches a minimizer-read row or a terminal RealUp value seal without first reading the compact sublevel row, LowerSemicontinuous row, coercive sublevel ledger, RegSeqRat readback, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: compactmetric, lowersemicontinuous, gamma-convergence, eulerlagrange, regseqrat, real: $\mathsf{DirectMethodMinimizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sublevel data, lower-semicontinuity rows, coercive ledgers, value windows, minimizer reads, rational readback, terminal real seals, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DirectMethodMinimizer}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectMethodMinimizer}^{\uparrow}}, \mathsf{seed})$

Constructive story: A $\mathsf{DirectMethodMinimizer}^{\uparrow}$ packet is generated from source rows, compact sublevel or CompactMetric rows, LowerSemicontinuous epigraph rows, coercive sublevel ledgers, functional value windows, minimizer-read rows, RegSeqRat readback, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Scope closed: Definition 8857.1, Definition 8857.2, and Definition 8857.3 seed the finite direct-method minimizer certificate through compact sublevel, lower-semicontinuity, regular rational readback, real-value sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirectMethodMinimizer}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.DirectMethodMinimizerUp`

Bridge status: none

Not claimed: This chapter does not close host minimizer selection, selected subsequence extraction, quotient equality of functionals, open-cover compactness, countable choice, Euler–Lagrange necessary conditions, Gamma-convergence recovery selection, ambient RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact sublevel exposure, lower-semicontinuity exposure, coercive ledger exactness, value-window readback, minimizer-read sealing, transport, replay, provenance, and local naming for BEDC.Derived.DirectMethodMinimizerUp.

Lean4 (statement) `BEDC.Derived.DirectMethodMinimizerUp`

Chapter 8858

A Concrete Naming Certificate for $\mathsf{UpwindFlux}^{\uparrow}$

Following LeVeque and Eymard–Gallouet–Herbin, the $\mathsf{UpwindFlux}^{\uparrow}$ packet records finite-volume flux orientation as BEDC-visible NameCert data. It sits over the conservative balance packet of chapter 6277, the dyadic mesh interface of chapter 358, the PDE relation surface of chapter 194, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The packet names only the finite upstream orientation rows used by a numerical flux; it does not import a host PDE solution, unrestricted convergence theorem, quotient weak-solution object, selected trace, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8858.1 (Upwind-flux carrier)**.** *An* $\mathsf{UpwindFlux}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (V, M, Q, A, O, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{FiniteVolume}^{\uparrow}$ *carrier row whose face-incidence and balance ledgers supply the finite-volume context.*
2. *M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the inspected cell, face, and neighbour windows at dyadic precision.*
3. *Q is the* $\mathsf{Pde}^{\uparrow}$ *relation row whose local derivative or conservation relation supplies the direction-sensitive PDE side.*
4. *A is the advection or sign classifier row, read only as a finite orientation comparison on each displayed face.*
5. *O is the upstream-choice row, selecting for each oriented face the left, right, or boundary value row licensed by V, M, Q, A.*
6. *W is the numerical-flux window row, recording the finite stencil values consumed after the upstream side has been selected.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected flux values and residual windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row comparing paired orientation and residual budgets.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after V, M, Q, A, O, W, R, D have been displayed.*

10. *H records componentwise* hsame *transport for the finite-volume, mesh, PDE, orientation, upstream-choice, flux-window, readback, tolerance, real-seal, replay, provenance, and local naming rows.*
11. *C is the displayed* Cont *replay from an upwind flux request through $V, M, Q, A, O, W, R, D, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{UpwindFlux}^{\uparrow}$, $\mathsf{FiniteVolume}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{UpwindFlux}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host PDE solution, unrestricted convergence of a scheme, quotient weak solution, selected trace, hidden solver state, or ambient real completeness theorem.

Theorem 8858.2 (Upwind-flux NameCert obligations)**.** *For every accepted* $\mathsf{UpwindFlux}^{\uparrow}$ *carrier*

$$U = (V, M, Q, A, O, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UpwindFlux}^{\uparrow}}$ *surface consists exactly of finite-volume admission through V, dyadic mesh admission through M, PDE relation admission through Q, orientation classifier admission through A, upstream-choice exactness through O, flux-window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, real-seal handoff through E, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8858.1. The rows V, M, Q expose the finite-volume balance packet, dyadic mesh window, and PDE relation before any orientation-sensitive flux read is public.

The row A is the only source of direction information. The upstream-choice row O is therefore exact only when it selects a displayed left, right, or boundary value from the finite rows already licensed by V, M, Q, A.

The flux-window row W consumes the selected side from O before the readback and tolerance rows are inspected. Thus R, D, E certify only the resulting finite flux and residual windows, not a host solution or a scheme-wide convergence theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no excluded host solution, unrestricted convergence proof, quotient weak-solution object, selected trace, solver state, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the local certificate surface. □

Theorem 8858.3 (Upwind-flux ledger stability)**.** *Let $U = (V, M, Q, A, O, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{UpwindFlux}^{\uparrow}$ *carrier. Any componentwise* hsame *transport that preserves the displayed finite-volume incidence in V, the dyadic mesh window in M, the PDE relation row in Q, and the orientation classifier row in A also preserves the upstream-choice ledger O and the numerical-flux window W as* $\mathsf{UpwindFlux}^{\uparrow}$ *data. The transported packet remains inside the same finite ledger boundary and exposes no host PDE solution, unrestricted convergence theorem, quotient weak solution, selected trace, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the componentwise transport rows in H. They move the finite-volume incidence row, dyadic mesh window, PDE relation row, and orientation classifier row without adding any coordinate outside the displayed packet.

Because O is defined only as the upstream side selected from V, M, Q, A, preserving those four rows preserves the source of every selection in O. Boundary faces remain boundary reads, left-facing interior choices remain left reads, and right-facing interior choices remain right reads.

The numerical-flux window W is downstream of O. Once the selected side is transported with the same classifier data, the regular readback R, dyadic tolerance row D, and real seal E replay the same finite flux route.

The continuation, provenance, and local name rows C, P, N then repack the transported data as an $\mathsf{UpwindFlux}^{\uparrow}$ carrier. No excluded analytic or quotient object can enter, since none is a field of Definition 8858.1. □

Falsifiable prediction: An accepted $\mathsf{UpwindFlux}^{\uparrow}$ consumer must expose the $\mathsf{FiniteVolume}^{\uparrow}$ face-incidence row, $\mathsf{DyadicMesh}^{\uparrow}$ inspected window, $\mathsf{Pde}^{\uparrow}$ relation row, orientation classifier, upstream-choice ledger, numerical-flux window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before it may read an upwind flux.

Independence witness: finite-volume, dyadicmesh, pde, regseqrat, dyadicratcore, real: no listed sibling supplies the whole packet $U = (V, M, Q, A, O, W, R, D, E, H, C, P, N)$, because each sibling contributes only a conservative balance, mesh window, PDE relation, regular rational readback, dyadic tolerance row, or terminal real seal, while this chapter binds them into one orientation-sensitive finite flux route.

Closure status for $\mathsf{NameCert}_{\mathsf{UpwindFlux}^{\uparrow}}$.

Constructive story: An $\mathsf{UpwindFlux}^{\uparrow}$ packet is generated from a FiniteVolumeUp balance row, DyadicMeshUp inspected window, PdeUp relation row, finite orientation classifier, upstream-choice row, numerical-flux window, RegSeqRat readback, DyadicRatCore tolerance, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UpwindFlux}^{\uparrow}}$, seed)

Scope closed: Definition 8858.1, Definition 8858.2, and Definition 8858.3 seed the finite upwind flux-orientation route over $\mathsf{FiniteVolume}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UpwindFlux}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host PDE solutions, unrestricted convergence, quotient weak solutions, selected traces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-volume incidence routing, mesh-window routing, PDE relation routing, orientation classifier exactness, upstream-choice stability, flux-window readback, dyadic tolerance, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.UpwindFluxUp.

Chapter 8859

A Concrete Naming Certificate for $\mathsf{FiniteDependentFamily}^{\uparrow}$

Following dependent type theory and finite product constructions, the $\mathsf{FiniteDependentFamily}^{\uparrow}$ packet records the finite BEDC surface for a family of displayed fibres indexed by a finite row. It sits after the finite-data interface of chapter 9353, the ordered spine discipline of chapter 36, the component envelope of chapter 34, the branch-tag boundary of chapter 35, and the function-object substrate of chapter 132. The packet also aligns with the continuous-map interface of chapter 130 when a downstream consumer reads a displayed fibre assignment as a finite family of maps. It names finite dependent family data only; it does not import an ambient dependent function space, quotient equality of fibres, a selected global section, countable choice, or host-side product extensionality.

Definition 8859.1 (Finite dependent family carrier)**.** *A* $\mathsf{FiniteDependentFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row. It is read as a displayed* $\mathsf{List}^{\uparrow}$ *spine of index names, with endpoint and branch discipline inherited from the* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, *and* $\mathsf{Sum}^{\uparrow}$ *finite-data surfaces.*
2. *F is the fibre row. For each displayed index of I, it stores one fibre NameCert row and the local classifier by which that fibre is inspected.*
3. *H records componentwise* hsame *transport for the index row, each fibre row, each pointwise classifier, the replay row, the provenance row, and the local naming row.*
4. *C records displayed* Cont *replay from finite index inspection through the matching fibre row and then through the pointwise classifier for that fibre.*
5. *P is the* Pkg *provenance over* $\mathsf{FiniteDependentFamily}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Func}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDependentFamily}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient dependent product, quotient fibre equality, selected section, countable-choice schedule, or host-side extensionality principle.

Theorem 8859.2 (Finite dependent family NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDependentFamily}^{\uparrow}$ *carrier*

$$D = (I, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDependentFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for* D*, finite index exposure through* I*, fibre exposure through* F*, pointwise classifier stability for each displayed fibre, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A consumer therefore reads a finite dependent family only through the displayed route*

$$I, F \longmapsto H, C, P, N$$

and not through an ambient dependent function space, quotient equality of fibres, selected global section, countable choice, or host product extensionality.

Proof. Project the accepted carrier. The finite index row I is the only source of fibre positions, so every consumer first reads the displayed list spine and its finite-data branch discipline.

For each exposed index, the fibre row F supplies the corresponding fibre NameCert row and its local classifier. Pointwise classifier stability is therefore checked inside the displayed fibre row rather than by comparing all fibres in an ambient dependent product.

Finally the structural rows preserve the same packet. Componentwise transport H, continuation replay C, provenance P, and local naming N exhaust the available route, leaving no coordinate for quotient fibre equality, selected sections, countable choice, or host extensionality. □

Falsifiable prediction: An accepted $\mathsf{FiniteDependentFamily}^{\uparrow}$ consumer that reads a fibre without the finite index row I, the displayed fibre row F, and the structural hsame, Cont, Pkg, and NameCert rows refutes the carrier surface.

Independence witness: finite-data-interface, list, product, sum, function-object: no listed sibling jointly carries the finite index row, displayed fibre NameCert rows, pointwise classifier stability, componentwise transport, replay, provenance, and local naming rows of $D = (I, F, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDependentFamily}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteDependentFamily}^{\uparrow}$ packet is generated from a finite index row, displayed fibre NameCert rows, pointwise classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteDependentFamily}^{\uparrow}}$, seed)

Scope closed: Definition 8859.1 and Definition 8859.2 seed the finite dependent-family carrier and its displayed NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteDependentFamily}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient dependent products, quotient fibre equality, selected sections, countable choice, host product extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, index exposure, fibre exposure, pointwise classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDependentFamilyUp.

Chapter 8860

A Concrete Naming Certificate for $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$

Following Knaster and continuum theory, the $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ packet records the bucket-handle continuum as finite NameCert data for a compact chainable continuum built from nested interval windows rather than from a host closed subset of the plane. It sits beside the compact-metric source of chapter 129, the Peano-continuum interface of chapter 2160, the topologist sine curve boundary of chapter 7316, the pseudo-arc inverse-limit surface of chapter 6905, and the terminal Real seal of chapter 38. The packet names finite chain covers, nested strip windows, bonding readback, and indecomposable-continuum obstruction rows; it does not import ambient planar topology, quotient continuum equality, selected inverse-limit points, open-cover compactness, or a host theorem about the bucket-handle continuum.

Definition 8860.1 (Knaster bucket-handle continuum carrier)**.** *A* $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, S, B, C, I, R, E, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric carrier, finite compactness windows, total-boundedness ledger, and distance readback used by every continuum-facing request.*
2. *S is the nested strip row. It records a finite prefix of horizontal strip windows, endpoint brackets, refinement inclusions, and the refusal of any ambient planar closed-set equality.*
3. *B is the bonding row, recording the finite inverse-limit prefix by which each strip window is read into the next level through displayed projection and compatibility data.*
4. *C is the chain-cover row. For each displayed scale it lists a finite chain cover, adjacency ledger, overlap certificates, and diameter bounds over the compact-metric source.*
5. *I is the indecomposability obstruction row, recording finite tests that prevent a displayed continuum window from being split into two proper subcontinuum windows inside the same prefix.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint brackets, strip widths, chain-cover diameters, projection tolerances, and obstruction budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the numeric windows read through S, B, C, I, R.*

8. *H records componentwise* hsame *transport for compact-metric, strip, bonding, chain-cover, obstruction, regular-rational, Real-seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from compact-metric cover requests through nested strip prefixes, bonding compatibility, chain covers, obstruction tests, and numeric readback.*
10. *P is the* Pkg *provenance over* $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{TopologistSineCurve}^{\uparrow}$, $\mathsf{PseudoArc}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KnasterBucketHandleContinuum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, B, C, I, R, E, H, T, P, N$. It has no coordinate for ambient planar subsets, host closed-set equality, quotient continuum equality, selected inverse-limit representatives, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8860.2 (Knaster bucket-handle continuum NameCert obligations)**.** *For every accepted* $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ *carrier*

$$K = (M, S, B, C, I, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KnasterBucketHandleContinuum}^{\uparrow}}$ *surface consists exactly of compact-metric source admission through M, nested strip-window admission through S, bonding-prefix compatibility through B, finite chain-cover coverage through C, indecomposability obstruction exposure through I, regular rational tolerance readback through R, terminal Real sealing through E, componentwise transport through H, displayed replay through T, provenance through P, and local naming through N.*

Every continuum-facing consumer must read these rows before it may use the packet as bucket-handle continuum evidence. No obligation row is discharged by an ambient planar subset, host closed-set equality, quotient continuum equality, selected inverse-limit representative, countable-choice schedule, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 8860.1. The compact-metric row M is the only source of metric and compactness data, so every later strip, bonding, chain-cover, or obstruction read is measured over the same displayed carrier.

Read the constructive bucket-handle rows next. The strip row S stores only finite nested strip windows, and the bonding row B records finite projection compatibility between those windows. Thus the packet uses displayed inverse-limit-style readback rather than a host closed subset of the plane.

The chainability and indecomposability data are then separated. The row C lists finite chain covers with adjacency, overlap, and diameter certificates; the row I records finite obstruction tests against splitting a displayed continuum window into two proper subcontinuum windows. The numeric side of both rows is read through R before the Real seal E is consumed.

Finally H, T, P, N transport, replay, source, and name the same finite packet. Since the refused principles have no carrier coordinate and no replay edge, the displayed rows exhaust the local NameCert surface. □

Theorem 8860.3 (Knaster bucket-handle continuum chainable handoff)**.** *Every chainable-continuum read of an accepted* $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ *packet factors through*

$$\mathsf{CompactMetric}^{\uparrow},\quad S,\quad B,\quad C,\quad I,\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}_{\mathsf{KnasterBuc}}$$

The handoff exposes the bucket-handle continuum only as finite nested strip prefixes, bonding compatibility, chain-cover rows, indecomposability obstruction rows, regular-rational readback, and Real

sealing. It exports no ambient planar topology, quotient continuum equality, selected inverse-limit point, open-cover compactness theorem, countable-choice selector, or ambient completeness claim.

Proof. Begin with the compact-metric source. The row M supplies the metric carrier and finite compactness windows before any continuum-facing read may inspect the strip or chain-cover data.

The bucket-handle-specific route then reads S and B. These rows display finite nested strip prefixes and bonding compatibility, so a downstream consumer obtains a finite prefix route rather than a selected inverse-limit point or host planar subset.

Next the consumer reads C and I. Chain covers in C provide the scale-by-scale chainable handoff, while obstruction windows in I keep the indecomposable-continuum boundary visible inside the same prefix. The tolerance and endpoint data for both rows pass through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The structural rows H, T, P, N preserve this route under transport, replay, provenance, and local naming. Therefore the handoff stays inside the displayed finite packet and cannot export the refused ambient principles. □

Falsifiable prediction: An accepted $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ consumer that reads a chain-cover row without first displaying the compact-metric source, nested strip prefix, bonding compatibility, regular-rational tolerance readback, Real seal, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: compactmetric, peano-continuum, topologist-sine-curve, pseudo-arc, real: $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ is not bijective to any listed sibling because it binds nested strip prefixes, bonding compatibility, chain-cover rows, indecomposability obstruction rows, regular-rational readback, Real sealing, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{KnasterBucketHandleContinuum}^{\uparrow}}$.

Constructive story: A $\mathsf{KnasterBucketHandleContinuum}^{\uparrow}$ packet is generated from CompactMetricUp source rows, nested strip prefix rows, bonding compatibility rows, finite chain-cover rows, indecomposability obstruction rows, RegSeqRatUp tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KnasterBucketHandleContinuum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8860.1, Definition 8860.2, and Definition 8860.3 seed the finite bucket-handle continuum route over compact-metric, nested-strip, bonding, chain-cover, obstruction, readback, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KnasterBucketHandleContinuum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient planar topology, quotient continuum equality, selected inverse-limit points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested strip-window exposure, bonding-prefix compatibility, finite chain-cover coverage, indecomposability obstruction exposure, tolerance readback, transport, replay, provenance, and local naming for BEDC.Derived.KnasterBucketHandleContinuumUp.

Chapter 8861

A Concrete Naming Certificate for $\mathsf{BumpFunction}^{\uparrow}$

Following Urysohn's lemma, Tietze extension, and the partition-of-unity tradition, the $\mathsf{BumpFunction}^{\uparrow}$ packet records the finite certificate surface for a locally supported nonzero continuous real-valued function without importing host function equality, choice-selected supports, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits beside the separating-function route of **??**, the extension route of chapter 1956, the locally finite gluing route of chapter 3369, the real-name seal of chapter 38, and the pointwise continuity carrier of chapter 130.

Definition 8861.1 (Bump-function carrier)**.** *A* $\mathsf{BumpFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, S, V, Z, F, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the local source row, naming the displayed neighborhood, closed core, or finite localization data on which the bump is requested.*
2. *S is the support row, recording the finite support boundary and its inclusion in the chosen local region.*
3. *V is the vanishing-boundary row, stating that reads outside the displayed support boundary return the zero real-name seal.*
4. *Z is the nonzero local witness, displayed inside the source row before any downstream support consumer may use the bump.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the named real-valued map.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row through which zero, nonzero, and boundary values are read.*
7. *H records componentwise* hsame *transport for the source, support, vanishing-boundary, nonzero witness, continuous-map, real-seal, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from local source data through support control and pointwise continuous-map reads into the real-value seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BumpFunction}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{TietzeExtension}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BumpFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, S, V, Z, F, R, H, C, P, N$. It has no coordinate for host equality of functions, a choice-selected support, a global support oracle, quotient real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8861.2 (Bump-function NameCert obligations)**.** *For an accepted* $\mathsf{BumpFunction}^{\uparrow}$ *carrier*

$$B = (L, S, V, Z, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BumpFunction}^{\uparrow}}$ *surface consists of carrier admission, local source exposure through L, support exposure through S, vanishing-boundary exposure through V, nonzero local witness exposure through Z, continuous-map exposure through F, real-name reads through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8861.1. The local source, support, vanishing boundary, nonzero witness, continuous-map row, real seal, transport, replay, provenance, and local name are all visible coordinates of one finite packet.

Read the analytic obligations in order. The source row L fixes the local region, S records the support boundary inside that region, V records the zero behavior at the boundary, and Z records the local nonzero witness before the downstream consumer treats the map as a bump.

Finally F, R, H, C, P, N preserve the same displayed route as continuous real-valued data. The sibling routes through $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{TietzeExtension}^{\uparrow}$, and $\mathsf{PartitionOfUnity}^{\uparrow}$ explain the traditional sources of such packets, while the carrier itself exports only the listed finite rows. □

Theorem 8861.3 (Bump-function support non-escape)**.** *Every* $\mathsf{BumpFunction}^{\uparrow}$ *consumer reading an accepted carrier*

$$B = (L, S, V, Z, F, R, H, C, P, N)$$

must expose the local source row L, support row S, vanishing-boundary row V, nonzero witness Z, $\mathsf{ContinuousMap}^{\uparrow}$ *row F, and* $\mathsf{Real}^{\uparrow}$ *seal R before it can use the bump as locally supported real-valued data. The packet exports no host function equality, selected support set, global support oracle, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The NameCert obligations of Definition 8861.2 enumerate every row available to a downstream consumer. In particular, support control is not separate from the displayed packet: it is the row S together with the vanishing-boundary row V and the local witness Z.

The continuous-map and real-value reads are also downstream of the same packet. A consumer reaches F and R only through L, S, V, Z and then through the transport, replay, provenance, and naming rows.

Since the carrier has no coordinate for host function equality, selected supports, global support oracles, quotient equality, countable choice, or ambient completeness, none of these can be recovered from an accepted $\mathsf{BumpFunction}^{\uparrow}$ packet. □

Falsifiable prediction: A claimed bump-function packet with no displayed support row, no vanishing-boundary row, or no nonzero local witness is not an accepted BumpFunctionUp carrier.

Independence witness: urysohnlemma, tietzeextension, partitionofunity, continuousmap, real: these sibling chapters supply separating, extending, gluing, continuity, and real-value rows, while BumpFunctionUp binds them around a local support and nonzero witness, so no row-preserving bijection exists.

Closure status for $\mathsf{NameCert}_{\mathsf{BumpFunction}^{\uparrow}}$.

Constructive story: A BumpFunction$^{\uparrow}$ packet is generated from a local source row, a displayed support row, vanishing-boundary data, a nonzero local witness, a ContinuousMap row, a Real seal, componentwise hsame transport, Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BumpFunction}^{\uparrow}}$, obligation)

Scope closed: Definition 8861.1, Definition 8861.2, and Definition 8861.3 bind the finite bump-function carrier and its local support boundary over source, support, vanishing, witness, ContinuousMap, Real, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BumpFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, choice-selected supports, global support oracles, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicit routing of local source, support, vanishing-boundary, nonzero witness, ContinuousMap, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows against the cited sibling surfaces.

Chapter 8862

A Concrete Naming Certificate for $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$ packet records the finite certificate by which a uniformly Cauchy family of Cauchy-name maps is handed to a limit-function reader without selecting a host function or appealing to ambient completeness. It sits after the Cauchy-name transformer boundary of chapter 1401, the real-valued uniform Cauchy family surface of chapter 1830, the shared endpoint criterion of chapter 6029, the regular rational readback row of chapter 301, and the terminal real-name seal of chapter 38. The packet names the displayed rows by which a completion-facing consumer may read a limit candidate, not a quotient function, selected pointwise limit, choice schedule, host continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8862.1 (Uniform Cauchy function limit carrier)**.** *A* $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, Q, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite Cauchy-name map family row, listing the inspected maps, source Cauchy-name windows, and target value windows.*
2. *Q is the* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *row, recording that the same displayed tolerance request controls every retained source window and family index.*
3. *M is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *boundary row, ensuring each inspected map is read through finite stream windows, regular rational image rows, dyadic tolerance ledgers, and a real-seal handoff.*
4. *S is the* $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ *row, joining the shared uniform Cauchy evidence to the limit-endpoint read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row opened from the displayed map-family windows before the terminal seal is inspected.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the limit candidate, available only after F, Q, M, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the family row, uniform Cauchy row, Cauchy-continuous map boundary, criterion row, readback row, seal row, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a limit-function request through $F, Q, M, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}$ $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFunctionLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, Q, M, S, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient Cauchy-name maps, selected pointwise limits, countable-choice selectors, global function-space completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8862.2 (Uniform Cauchy function limit NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$ *carrier*

$$L = (F, Q, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyFunctionLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite Cauchy-name map family, real-function uniform Cauchy row, Cauchy-continuous map boundary, uniform-limit Cauchy criterion row, regular rational readback, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. Every limit read must display the shared uniform Cauchy evidence Q, the Cauchy-continuous map boundary M, and the criterion row S before R and E can be consumed.*

Proof. Project the carrier of Definition 8862.1. The only admitted rows are $F, Q, M, S, R, E, H, C, P, N$, so the local certificate has no place to store an external limit function or a host equality proof.

The uniform Cauchy and continuity obligations are the rows Q and M. Their placement before S, R, E forces the family comparison and map-readback surfaces to be displayed before any endpoint seal is reached.

The criterion row S connects those rows to the limit endpoint. The regular rational readback R and terminal seal E then provide the Real-facing route, while H, C, P, N transport, replay, source, and name the same finite packet. □

Theorem 8862.3 (Uniform Cauchy function limit Real seal)**.** *Let $L = (F, Q, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing read of the limit candidate factors through the displayed route*

$$\mathsf{CauchyContinuousMap}^{\uparrow} \longmapsto \mathsf{RealFunctionUniformCauchy}^{\uparrow} \longmapsto \mathsf{UniformLimitCauchyCriterion}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

using the rows M, Q, S, R, E together with transport, replay, provenance, and local naming. The seal cannot be reached from a quotient function, a selected pointwise limit, a countable-choice schedule, or an ambient completeness argument.

Proof. Read the endpoint rows in Definition 8862.1. The seal E is downstream of the map boundary M, uniform Cauchy row Q, criterion row S, and regular readback R.

Replay the request through C. The replay route exposes exactly those rows before the seal is visible, and componentwise transport through H preserves the same dependency order.

The provenance row P sources only the listed packet rows. Therefore a Real-facing consumer receives the displayed route to E and no detached quotient function, selected pointwise limit, choice schedule, or ambient completeness datum. □

Theorem 8862.4 (Uniform Cauchy function limit continuity handoff)**.** *Every continuity-facing consumer of an accepted* $\mathsf{UniformCauchyFunctionLimit}^{\uparrow}$ *packet must first inspect the* $\mathsf{CauchyContinuousMap}^{\uparrow}$

row M, the RealFunctionUniformCauchy$^{\uparrow}$ *row Q, and the* UniformLimitCauchyCriterion$^{\uparrow}$ *row S. The packet therefore hands forward a finite Cauchy-name continuity boundary for the limit candidate, not a host-level proof of continuity over completed reals.*

Proof. The continuity-facing data are the rows M, Q, S in the carrier. They record the map boundary, the shared uniform Cauchy control, and the limit criterion before any regular readback or Real seal is exported.

Use the replay row C and transport row H to present the same rows to downstream consumers. Since P and N source and name only this finite packet, the consumer cannot replace the handoff by a host function on completed reals.

Thus the limit packet exposes a Cauchy-name continuity boundary, while host-level continuity, quotient function equality, selected limits, and completeness remain outside the local certificate. □

Closure status for NameCert$_{\text{UniformCauchyFunctionLimit}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{UniformCauchyFunctionLimit}^{\uparrow}}$, seed)

Scope closed: Definition 8862.1, Definition 8862.2, Definition 8862.3, and Definition 8862.4 seed the finite certificate for uniform Cauchy limits of Cauchy-name maps through CauchyContinuousMap, RealFunctionUniformCauchy, UniformLimitCauchyCriterion, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{UniformCauchyFunctionLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient Cauchy-name maps, selected pointwise limits, countable choice, global function-space completeness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform Cauchy row stability, Cauchy-continuous map boundary exactness, criterion routing, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyFunctionLimitUp.

Constructive story: A UniformCauchyFunctionLimit$^{\uparrow}$ packet is generated from a finite Cauchy-name map family, RealFunctionUniformCauchy evidence, a CauchyContinuousMap boundary, UniformLimitCauchyCriterion routing, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8863

A Concrete Naming Certificate for $\mathsf{BlakersMassey}^{\uparrow}$

Following Blakers and Massey on homotopy excision, the $\mathsf{BlakersMassey}^{\uparrow}$ packet records the finite BEDC surface by which two highly connected subobject rows and their intersection row are handed to an excision consumer. It sits after the $\mathsf{Homotopy}^{\uparrow}$ deformation surface of chapter 156, the $\mathsf{SimplicialSet}^{\uparrow}$ face-and-degeneracy ledger of chapter 157, the $\mathsf{Homology}^{\uparrow}$ chain-read surface of chapter 98, the van Kampen finite pushout route of chapter 8419, and the suspension connectivity handoff of chapter 8477. The packet names a displayed homotopy-excision handoff only; it does not import ambient model-category factorization, quotient topology, selected homotopy groups, a spectral sequence, or a host Blakers–Massey theorem.

Definition 8863.1 (Blakers–Massey carrier)**.** *A* $\mathsf{BlakersMassey}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, A, C, I, S, H_*, V, F, T, R, P, N)$$

with the following displayed rows:

1. *X is the ambient gluing row, recording the finite union or pushout window whose two subobject faces are inspected.*
2. *A and C are the two subobject rows, read through* $\mathsf{Homotopy}^{\uparrow}$ *deformation data and through the local face ledger of the ambient packet.*
3. *I is the intersection row, recording the displayed overlap, endpoint compatibility, and basepoint or boundary rows shared by A and C.*
4. *S is the* $\mathsf{SimplicialSet}^{\uparrow}$ *row carrying the finite face, horn, and degeneracy itinerary used to display the excision square.*
5. *H_* is the* $\mathsf{Homology}^{\uparrow}$ *read window used only as a finite chain-level sanity check for the inspected degrees.*
6. *V is the* $\mathsf{VanKampen}^{\uparrow}$ *sibling row, recording the already displayed pushout route for fundamental group style reads before higher homotopy-excision vocabulary is consumed.*
7. *F is the* $\mathsf{FreudenthalSuspension}^{\uparrow}$ *sibling row, recording the stable-range connectivity handoff used when a suspension-facing consumer asks for the same excision boundary.*
8. *T records componentwise* hsame *transport for the ambient, subobject, overlap, simplicial, homotopy, homology, van Kampen, Freudenthal, replay, provenance, and local naming rows.*

9. *R records displayed* Cont *replay from an excision request through $X, A, C, I, S, H_*, V, F, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BlakersMassey}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{VanKampen}^{\uparrow}$, $\mathsf{FreudenthalSuspension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BlakersMassey}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, C, I, S, H_*, V, F, T, R, P, N$. It has no coordinate for a host homotopy pushout theorem, selected homotopy-group element, quotient space, model-category cofibration replacement, spectral-sequence convergence claim, or unbounded connectivity proof.

Theorem 8863.2 (Blakers–Massey NameCert obligations)**.** *For every accepted* $\mathsf{BlakersMassey}^{\uparrow}$ *carrier*

$$B = (X, A, C, I, S, H_*, V, F, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BlakersMassey}^{\uparrow}}$ *surface consists exactly of ambient gluing admission through X, subobject exposure through A and C, intersection exposure through I, finite simplicial itinerary exposure through S, homotopy deformation reads through the displayed sibling rows, homology-window sanity checks through H_*, van Kampen pushout handoff through V, Freudenthal suspension handoff through F, componentwise transport through T, displayed replay through R, provenance through P, local naming through N, and non-escape from those displayed rows.*

Proof. Project the accepted carrier. The rows X, A, C, I supply the ambient square, the two subobject faces, and the overlap boundary, while S supplies the finite simplicial itinerary used to inspect that square.

Read the sibling rows in dependency order. The homotopy-facing data is consumed through the displayed $\mathsf{Homotopy}^{\uparrow}$ rows attached to A, C, I; the $\mathsf{Homology}^{\uparrow}$ row H_* supplies only a finite chain-read window; the $\mathsf{VanKampen}^{\uparrow}$ row V supplies the lower pushout route; and the $\mathsf{FreudenthalSuspension}^{\uparrow}$ row F supplies the suspension connectivity handoff.

Finally apply the structural rows. Transport T and replay R preserve exactly the rows already listed, and P, N only source and name the packet. Since the carrier has no coordinate for ambient quotient topology, chosen homotopy groups, model-category replacement, spectral-sequence convergence, or a host Blakers–Massey theorem, none of those objects can be exported by the local NameCert surface. □

Theorem 8863.3 (Blakers–Massey excision handoff)**.** *Every excision-facing consumer of an accepted* $\mathsf{BlakersMassey}^{\uparrow}$ *packet must read the route*

$$X, A, C, I, S \longmapsto H_*, V, F \longmapsto T, R, P, N$$

before treating the packet as a homotopy-excision boundary. The handoff preserves the displayed $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{VanKampen}^{\uparrow}$, *and* $\mathsf{FreudenthalSuspension}^{\uparrow}$ *sibling rows, and it exports no ambient Blakers–Massey theorem, no quotient-space construction, no selected homotopy class, no hidden connectivity range, and no bridge certificate.*

Proof. Begin with the carrier rows of Definition 8863.1. The excision request starts with the ambient gluing row X, the two subobject rows A, C, the overlap row I, and the finite simplicial route S.

Apply the NameCert obligations from Definition 8863.2. They force the consumer to expose the homotopy, simplicial, homology, van Kampen, and Freudenthal sibling rows before any excision vocabulary is public.

The handoff is therefore row-preserving. The rows H_*, V, F are read as displayed dependency anchors, not as ambient theorem oracles; T, R, P, N then transport, replay, source, and name the same finite packet.

Because the packet contains no quotient-space constructor, selected homotopy-class row, hidden connectivity-range row, standard bridge row, or host theorem row, the consumer receives only the finite homotopy-excision boundary named by $X, A, C, I, S, H_*, V, F, T, R, P, N$. □

Falsifiable prediction: Any accepted $\mathsf{BlakersMassey}^{\uparrow}$ consumer that cites homotopy excision without first displaying the overlap row I, the simplicial route S, and the sibling rows $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{VanKampen}^{\uparrow}$, and $\mathsf{FreudenthalSuspension}^{\uparrow}$ refutes the carrier surface of Definition 8863.1.

Independence witness: homotopy, simplicialset, homology, van-kampen, freudenthal-suspension: none is bijective to $\mathsf{BlakersMassey}^{\uparrow}$, because this packet alone binds the two-face gluing row, overlap row, finite simplicial itinerary, homology sanity window, van Kampen pushout route, Freudenthal connectivity handoff, componentwise transport, replay, provenance, and local NameCert rows into one homotopy-excision surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BlakersMassey}^{\uparrow}}$.

Constructive story: A $\mathsf{BlakersMassey}^{\uparrow}$ packet is generated from an ambient gluing row, two subobject rows, an overlap row, a finite SimplicialSet route, a Homology read window, VanKampen and FreudenthalSuspension handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BlakersMassey}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8863.1, Definition 8863.2, and Definition 8863.3 seed the finite homotopy-excision route over Homotopy, SimplicialSet, Homology, VanKampen, FreudenthalSuspension, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BlakersMassey}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the classical Blakers–Massey theorem, quotient topology, selected homotopy classes, hidden connectivity ranges, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked ambient gluing admission, subobject exposure, overlap exposure, simplicial route exposure, homology-window readback, VanKampen and Freudenthal handoffs, transport, replay, provenance, and local naming for BEDC.Derived.BlakersMasseyUp.

Chapter 8864

A Concrete Naming Certificate for $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$

The $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ packet realizes the three-boundary separation of chapter 9240 as finite NameCert data. It sits after the complex carrier of chapter 39, the real-name seal of chapter 38, the zeta-zero witness surface of chapter 73, and the finite prime-window gluing route of chapter 8765. The packet keeps the geometric critical-line surface, certified zero predicate, and RH rigidity boundary in separate rows before any consumer may ask about a total zero decider.

Definition 8864.1 (RH boundary-kind separation carrier)**.** *An* $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, Z, R, D, F, H, C, P, N)$$

with the following displayed rows:

1. *G is the geometric critical-line row, storing the mirror-fixed surface and its* $\mathsf{Real}^{\uparrow}$ *readback.*
2. *Z is the certified zeta-zero predicate row, read through the displayed* $\mathsf{ZetaZeros}^{\uparrow}$ *witness surface and the inspected* $\mathsf{Complex}^{\uparrow}$ *point.*
3. *R is the RH rigidity row, recording only the conditional route from certified critical-strip zero evidence to a critical-line witness.*
4. *D is the decider-boundary row, recording whether a total zero-decider row is explicitly displayed; absence of this row blocks any total-decision read.*
5. *F is the finite prime-window gluing row, exposing prime-window data only as a sibling dependency of the certified zero route, not as a replacement for G, Z, R, D.*
6. *H records componentwise* hsame *transport for the geometric row, zero-predicate row, rigidity row, decider-boundary row, gluing row, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from an RH-facing boundary request through G, Z, R, D, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RHBoundaryKindSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, Z, R, D, F, H, C, P, N$. It has no coordinate for an RH proof, an RH refutation, a hidden total zero decider, a quotient zero set, host complex equality, or a collapse of geometric, predicate, and rigidity data into one row.

Theorem 8864.2 (RH boundary-kind separation NameCert obligations)**.** *For every accepted* $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ *carrier*

$$B = (G, Z, R, D, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RHBoundaryKindSeparation}^{\uparrow}}$ *surface consists exactly of carrier admission for the geometric critical-line row, certified zero-predicate row, RH rigidity row, displayed decider-boundary row, finite prime-window gluing row, componentwise transport, displayed replay, provenance, and local naming. A consumer may use the packet only after it keeps* G*,* Z*, and* R *as distinct rows and treats* D *as the sole place where a total zero decider could be displayed.*

Proof. Project the carrier of Definition 8864.1. The rows G, Z, R, D, F are listed separately before the structural rows H, C, P, N.

The geometric row G is read from the mirror-fixed surface and real readback. The zero-predicate row Z is read from certified zeta-zero evidence at a complex point. The rigidity row R consumes zero and strip evidence conditionally; it does not decide whether such evidence exists.

The decider-boundary row D is the only coordinate in which a total zero decider can be displayed. Since the classifier has no hidden coordinate outside $G, Z, R, D, F, H, C, P, N$, omitting D blocks a total-decision read. □

Theorem 8864.3 (RH boundary-kind decider nonescape)**.** *Any accepted* $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ *packet has no row for a total zero decider unless the decider-boundary row* D *explicitly displays one. In particular, the geometric critical-line row* G*, certified zero-predicate row* Z*, rigidity row* R*, and finite prime-window gluing row* F *cannot be reclassified as a total decision procedure.*

Proof. Inspect the carrier inventory. The rows G, Z, R, F record a fixed surface, a supplied zero predicate, a conditional rigidity route, and finite prime-window gluing data.

None of those rows has the shape of a total decision procedure over all complex histories. The row D is the only row assigned to the zero-decider boundary.

Transport through H and replay through C preserve the same row inventory. Provenance and local naming through P, N do not add a missing decider coordinate. Therefore a total zero-decider read is accepted only when it is displayed in D. □

Theorem 8864.4 (RH boundary-kind sibling separation)**.** *The* $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ *carrier is not bijective to the sibling packets for real names, complex names, zeta-zero witnesses, or finite prime-window gluing. It jointly binds geometric critical-line, certified zero-predicate, rigidity, decider-boundary, and gluing rows, while each listed sibling supplies only one component surface or one dependency route.*

Proof. Compare row inventories. The real and complex siblings supply the scalar and point carriers used by G and Z, but they do not contain a zeta-zero predicate or RH rigidity row.

The zeta-zero sibling supplies certified zero evidence, but it does not by itself display the mirror-fixed geometric surface, the rigidity boundary, or the decider-boundary row.

The finite prime-window gluing sibling supplies the gluing route consumed by F. It does not carry the three-way boundary ledger G, Z, R, D.

Thus no listed sibling has the same accepted carrier surface as $B = (G, Z, R, D, F, H, C, P, N)$, and no bijection can preserve the local NameCert rows component by component. □

Falsifiable prediction: Any accepted RHBoundaryKindSeparationUp packet has no row for a total zero decider unless the zero-decider row is explicitly displayed.

Independence witness: real, complex, zeta, finite_prime_window_gluing: the packet is not bijective to these siblings because it carries boundary-kind separation rows linking all of them while none of the listed siblings carries the three-way separation ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{RHBoundaryKindSeparation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHBoundaryKindSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8864.1, Definition 8864.2, Definition 8864.3, and Definition 8864.4 seed the finite three-boundary ledger separating geometric critical-line, certified zero-predicate, RH rigidity, decider-boundary, and finite prime-window gluing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RHBoundaryKindSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove RH, refute RH, decide all zeta zeros, enumerate all zeros, collapse the critical-line surface into the zero predicate, supply Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, geometric-row exactness, certified zero-predicate routing, rigidity-row separation, decider-boundary nonescape, finite prime-window gluing handoff, transport, replay, provenance, local naming, and the future Nontrivial and FieldFaithful taste gates for BEDC.Derived.RHBoundaryKindSeparationUp.

Constructive story: An $\mathsf{RHBoundaryKindSeparation}^{\uparrow}$ packet is generated from geometric critical-line rows, certified zeta-zero predicate rows, RH rigidity rows, decider-boundary rows, finite prime-window gluing rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8865

A Concrete Naming Certificate for $\mathsf{PrecompactSpace}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style finite-net readings of precompactness, the $\mathsf{PrecompactSpace}^{\uparrow}$ packet records the finite route by which a metric space supplies total-bounded data to completion-facing consumers. It sits over the metric carrier of chapter 57, the finite-net surface of chapter 127, and the compact-metric consumer boundary of chapter 129. The packet is a displayed NameCert interface for finite ε-net handoff; it is not an open-cover compactness theorem, a selected dense subset, a quotient completion, or a host choice principle.

Definition 8865.1 (PrecompactSpace carrier)**.** *A* $\mathsf{PrecompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, T, K, S, Q, R, N, H, C, G)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, including the accepted point histories, point classifier, and metric-distance witness surface;*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, assigning each positive tolerance request to a closed finite net surface;*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *consumer boundary row, recording the completion-facing interface that may later combine total boundedness with complete-metric data;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the finite windows consumed by completion-facing reads;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the same stream windows;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and distance-readback seal used by dyadic or rational comparison consumers;*
7. *N records the closed finite net data: center histories,* $\mathsf{ProbeBundle}$ *membership rows, displayed distance witnesses, and radius comparison ledgers;*
8. *H records componentwise* hsame *transport for M, T, K, S, Q, R, N and for every center, stream window, readback, distance, and radius row exposed by them;*
9. *C records the displayed* Cont *replay route from a tolerance request through the finite net rows and then to any completion-facing consumer;*

10. *G is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{PrecompactSpace}^{\uparrow}}$ *row naming* $\mathsf{PrecompactSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

The classifier on such packets is componentwise transport through H together with the route replay in C. No coordinate stores an open cover, a choice-selected dense sequence, a completed quotient point, or an ambient compactness witness.

Theorem 8865.2 (PrecompactSpace NameCert obligations)**.** *For an accepted* $\mathsf{PrecompactSpace}^{\uparrow}$ *carrier $P = (M, T, K, S, Q, R, N, H, C, G)$, the local* $\mathsf{NameCert}_{\mathsf{PrecompactSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, metric source exposure, total-bounded finite-net exposure, compact-metric consumer boundary exposure, stream-window scheduling, regular rational readback, real tolerance sealing, closed finite-net ledger visibility, componentwise transport, displayed replay, package provenance, and local naming. These obligations make precompactness available as finite NameCert data over $M, T, K, S, Q, R, N, H, C, G$, without adding open-cover compactness, choice-selected dense subsets, quotient completions, selected limit points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is Definition 8865.1. The accepted packet exposes only the rows $M, T, K, S, Q, R, N, H, C$, and the finite subrows named by those coordinates.

The metric and total-bounded obligations are read before any completion-facing consumer is reached. The row M supplies the point histories, classifier, and distance witnesses, while T and N expose the finite net centers, ProbeBundle membership rows, distance witnesses, and radius comparison ledgers.

The consumer boundary is then routed through K, S, Q, R, H, C, G. These rows say which compact-metric, stream, regular-readback, real-tolerance, transport, replay, provenance, and local naming data may be inspected. Since no open-cover, dense-choice, quotient-completion, selected-limit, or ambient-completeness coordinate is present, no such principle can be projected from the local certificate. □

Theorem 8865.3 (PrecompactSpace total-bounded handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{PrecompactSpace}^{\uparrow}$ *packet must read total-boundedness through the displayed finite-net rows T and N before using the compact-metric boundary row K. In particular, a consumer may pass from a tolerance request to the stream and regular-readback rows S, Q, R only along the replay route C and the componentwise transport ledger H; it cannot replace the finite net by an open-cover argument, a choice-selected dense subset, a host compactness theorem, or a quotient completion.*

Proof. Open the carrier $P = (M, T, K, S, Q, R, N, H, C, G)$. The only total-bounded data in the packet is the displayed finite-net surface in T together with the closed net ledger N.

Use the NameCert obligations above. They order the read so that a tolerance request reaches the metric source M, the total-bounded finite-net row T, and the net ledger N before any compact-metric consumer boundary in K can be used.

Finally transport and replay are explicit. The rows H and C preserve the same center histories, membership constructors, distance witnesses, radius comparisons, stream windows, regular readbacks, and real tolerance seals when the consumer moves toward completion-facing data. Thus the handoff remains a finite NameCert route rather than an appeal to open-cover compactness, selected dense subsets, host choice, or quotient completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactSpace}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrecompactSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8865.1, Definition 8865.2, and Definition 8865.3 seed the finite-net handoff from MetricSpace and TotallyBounded rows to CompactMetric-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{PrecompactSpace}\uparrow}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, choice-selected dense subsets, quotient completions, selected limit points, ambient Real completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exactness, total-bounded handoff, transport stability, replay stability, and local naming for BEDC.Derived.PrecompactSpaceUp.

Constructive story: A PrecompactSpace$^{\uparrow}$ packet is generated from a MetricSpace carrier, TotallyBounded finite-net row, CompactMetric consumer boundary, StreamName windows, RegSeqRat readback, Real tolerance seal, closed finite net ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8866

A Concrete Naming Certificate for $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$

Following Birkhoff's projective metric on cones and standard positive-operator treatments, the $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ packet records the finite cone-ratio surface by which two positive cone elements are compared before a contraction theorem consumes the result. It sits over the positive-cone carrier of chapter 2297, the metric carrier of chapter 57, the real-name readback of chapter 38, and the contraction consumer of chapter 7755. The packet names a finite projective-distance ledger; it is not a quotient of cone rays, an ambient topological cone, a supremum-infimum oracle, or a fixed-point theorem.

Definition 8866.1 (Birkhoff projective metric carrier)**.** *A* $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, x, y, U, L, \rho, d, M, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{PositiveCone}^{\uparrow}$ *source row, fixing the cone surface and the scalar-order comparison used by every ratio read.*
2. *x and y are two displayed positive cone elements in the same K row.*
3. *U is the finite upper-ratio ledger, listing the displayed scalar witnesses for $x \leq u\,y$ that the packet is allowed to inspect.*
4. *L is the finite lower-ratio ledger, listing the displayed scalar witnesses for $\ell\,y \leq x$ that the packet is allowed to inspect.*
5. *ρ is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for the accepted upper and lower ratio bounds.*
6. *d is the* $\mathsf{Met}^{\uparrow}$ *projective-distance row, storing only the displayed logarithmic ratio readback produced from U, L, ρ.*
7. *M is the metric-surface compatibility row, requiring the distance read to be symmetric on the displayed pair and zero only through the local projective-identification ledger.*
8. *G is the* $\mathsf{BirkhoffContraction}^{\uparrow}$ *consumer handoff row, exposing K, x, y, U, L, ρ, d as the finite Hilbert-projective metric surface later used by contraction budgets.*
9. *H records componentwise* hsame *transport for the cone source, two positive elements, ratio ledgers, scalar readbacks, projective-distance row, metric compatibility, consumer handoff, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a projective-distance request through* $K, x, y, U, L, \rho, d, M, G, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BirkhoffContraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffProjectiveMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, x, y, U, L, \rho, d, M, G, H, C, P, N$. It has no coordinate for an unstated cone quotient, hidden supremum or infimum, ambient norm, fixed point, eigenvector, non-displayed operator, or host real equality outside the finite ratio ledgers.

Theorem 8866.2 (Birkhoff projective metric NameCert obligations)**.** *For every accepted* $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ *carrier*

$$B = (K, x, y, U, L, \rho, d, M, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BirkhoffProjectiveMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{PositiveCone}^{\uparrow}$ *source row, the two displayed positive elements, finite upper- and lower-ratio ledgers,* $\mathsf{Real}^{\uparrow}$ *scalar readbacks, the* $\mathsf{Met}^{\uparrow}$ *projective-distance row, the metric-compatibility row, the* $\mathsf{BirkhoffContraction}^{\uparrow}$ *consumer handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every downstream projective-metric read must first display the common cone source and both ratio ledgers. The read may then consume ρ and d as finite scalar and metric rows, but it cannot replace them with a completed quotient of rays, a hidden extremal ratio, a host topological cone, or a contraction conclusion.

Proof. Project the carrier of Definition 8866.1. The rows K, x, y fix a single positive-cone source and two elements in that source before any ratio comparison can be read.

The rows U and L are finite ledgers. They display only the upper witnesses $x \leq u\, y$ and lower witnesses $\ell\, y \leq x$ that the packet records, so no hidden supremum, infimum, or ambient order oracle is available.

The scalar row ρ and metric row d are downstream of those ledgers. The compatibility row M keeps the projective-distance read inside the displayed metric surface, and G exposes exactly the same data to the $\mathsf{BirkhoffContraction}^{\uparrow}$ consumer.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite row inventory. Since the classifier has no coordinate outside $K, x, y, U, L, \rho, d, M, G, H, C, P$, every accepted NameCert obligation is one of the listed rows. □

Theorem 8866.3 (Birkhoff projective metric ratio nonescape)**.** *An accepted* $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ *packet exports no projective distance unless the common* $\mathsf{PositiveCone}^{\uparrow}$ *source row, both positive elements, finite upper-ratio ledger, finite lower-ratio ledger,* $\mathsf{Real}^{\uparrow}$ *scalar readback, and* $\mathsf{Met}^{\uparrow}$ *projective-distance row are all displayed in the same carrier.*

Proof. Inspect the carrier inventory. The distance row d is admitted only after the source row K, element rows x, y, ratio ledgers U, L, and scalar readback row ρ have been displayed.

The replay row C follows the same order, and the handoff row G exposes the distance to $\mathsf{BirkhoffContraction}^{\uparrow}$ only with the source, elements, ledgers, and readback still attached.

Thus no public read can detach d from the finite ratio evidence that produced it. Quotient-ray language, hidden extremal ratios, and host completeness claims remain outside the packet. □

Falsifiable prediction: Any accepted BirkhoffProjectiveMetricUp distance read is refuted if it omits the common positive-cone source, either displayed positive element, either finite ratio ledger, the RealUp readback, or the MetricUp projective-distance row.

Independence witness: positive-cone, metric, real, birkhoff-contraction: none alone is bijective with $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ because the packet jointly records two positive cone elements, finite upper and lower ratio ledgers, scalar readbacks, projective-distance rows, contraction handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffProjectiveMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BirkhoffProjectiveMetric}^{\uparrow}$ packet is generated from a PositiveConeUp source row, two displayed positive cone elements, finite upper- and lower-ratio ledgers, RealUp scalar readbacks, a MetricUp projective-distance row, a BirkhoffContractionUp consumer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffProjectiveMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8866.1, Definition 8866.2, and Definition 8866.3 seed the finite cone-ratio projective metric surface over PositiveConeUp, MetricUp, RealUp, BirkhoffContractionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffProjectiveMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not construct a quotient of cone rays, prove contraction, select an eigenvector or fixed point, supply hidden extremal ratios, invoke host cone topology, close Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-cone discipline, upper- and lower-ratio ledger exactness, scalar readback, projective-distance routing, contraction handoff, transport, replay, provenance, and local naming for BEDC.Derived.BirkhoffProjectiveMetricUp.

Chapter 8867

A Concrete Naming Certificate for $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$

Following Bishop and Bridges on located intervals, regular Cauchy names, and constructive real completion, the $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ packet records a finite chain of located interval refinements before a Real or Completion consumer may read a terminal real-name seal. It sits between the located-interval source of chapter 394, the real-interval endpoint surface of chapter 4111, the dyadic tolerance descent of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate gives the L10 Real and Completion route a displayed refinement-chain object; it does not assert compactness, trichotomy, selected representatives, countable choice, quotient equality of names, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8867.1 (Bishop located interval refinement-chain carrier)**.** *A* $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, I, D, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row naming the initial located interval and its apartness-facing boundary data.*
2. *I is the* $\mathsf{RealInterval}^{\uparrow}$ *endpoint and enclosure row, displaying lower and upper real-interval endpoints together with the enclosure relation consumed by the chain.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-descent row, recording a finite decreasing dyadic budget for the refinement steps.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at each displayed tolerance stage.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and dyadic tolerances.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located interval source, endpoint-enclosure row, tolerance descent, windows, and readback have been displayed.*
7. *H records componentwise* hsame *transport for the located source, endpoint-enclosure row, dyadic descent, finite windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a refinement request through $L, I, D, W, Q, S, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, I, D, W, Q, S, H, C, P, N$. It has no coordinate for interval compactness, trichotomy of reals, a selected representative of a completion class, countable choice, quotient equality of interval names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8867.2 (Bishop located interval refinement-chain NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ *carrier*

$$R = (L, I, D, W, Q, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval source, RealInterval endpoint and enclosure row, DyadicRatCore tolerance descent, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8867.1. The tuple displays the located source L, the endpoint-enclosure row I, the tolerance descent D, the finite windows W, the regular readback Q, and the terminal seal S before any structural row is consumed.

The structural rows H, C, P, N preserve, replay, source, and name only those displayed rows. They do not add a hidden interval compactness theorem, a selected completion representative, or an ambient completeness principle.

Thus the local NameCert surface is exhausted by the ten displayed rows, and every accepted consumer must read the refinement chain through that finite packet. □

Theorem 8867.3 (Bishop located interval refinement-chain sibling route)**.** *The* BishopLocatedIntervalRefinementCh *carrier is not bijective to the located interval, real interval, dyadic rational core, stream-name, regular rational readback, or real-name sibling packets. It binds all six sibling surfaces into one ordered refinement-chain packet whose endpoint, tolerance, window, readback, and seal rows are consumed together.*

Proof. Compare row inventories. The located interval sibling supplies L, but it does not include the endpoint-enclosure row, dyadic tolerance descent, finite window schedule, regular readback, or terminal Real seal.

The real interval and dyadic rational siblings supply I and D respectively. Neither one records the StreamName windows, RegSeqRat readback, Real seal, and structural rows as a single chain.

The stream-name, regular-rational, and real-name siblings supply W, Q, S as downstream readings. They do not contain the located source and endpoint-enclosure data that make those readings a refinement chain. Therefore no listed sibling preserves the accepted packet $R = (L, I, D, W, Q, S, H, C, P, N)$ component by component. □

Theorem 8867.4 (Bishop located interval refinement-chain Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ *packet reaches the seal row S only through the displayed route*

$$L, I, D, W, Q, S, H, C, P, N.$$

The read does not select a representative, decide trichotomy, infer compactness, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Use the NameCert obligations of Definition 8867.2. They force the located source, endpoint-enclosure row, dyadic tolerance descent, finite windows, and regular readback to be present before the seal row is public.

The sibling-route theorem Definition 8867.3 prevents a consumer from replacing this packet by any one sibling surface. Transport and replay keep the same displayed rows.

The classifier of Definition 8867.1 has no coordinate for selected representatives, trichotomy, compactness, countable choice, quotient equality, or ambient completeness. Hence the Real-facing read is confined to the finite refinement-chain route. □

Falsifiable prediction: Any accepted $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the located interval source, RealInterval endpoint-enclosure row, DyadicRatCore tolerance descent, StreamName finite windows, RegSeqRat readback, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: locatedinterval, realinterval, dyadicratcore, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$, because each sibling supplies one source, endpoint, tolerance, window, readback, or seal surface while this packet stores their ordered refinement-chain dependency in one NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}$ packet is generated from a LocatedInterval source row, a RealInterval endpoint-enclosure row, a DyadicRatCore tolerance descent, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8867.1, Definition 8867.2, Definition 8867.3, and Definition 8867.4 seed the finite refinement-chain packet over located interval source, endpoint enclosure, dyadic tolerance descent, finite windows, regular readback, terminal sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalRefinementChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, trichotomy, countable choice, selected representatives, quotient equality of interval names, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-route separation, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedIntervalRefinementChainUp.

Chapter 8868

A Concrete Naming Certificate for RHCriticalStripWindowCertificate$^{\uparrow}$

The RHCriticalStripWindowCertificate$^{\uparrow}$ packet realizes a finite critical-strip window route from chapter 9240 as NameCert data. It sits beside the three-boundary separation of chapter 8864, the zeta-zero witness surface of chapter 73, the complex carrier of chapter 39, the real-name seal of chapter 38, and the finite prime-window gluing route of chapter 8765. The packet exposes a finite critical-strip window and certified-zero row while refusing to turn that window into an exact zero decider or a global RH rigidity proof.

Definition 8868.1 (RH critical-strip window certificate carrier)**.** *An* RHCriticalStripWindowCertificate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, M, Z, E, R, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite critical-strip window row, storing the inspected complex histories and their critical-strip bounds.*
2. *M is the mirror fixed-surface row, recording the functional-equation mirror surface available on the same finite window.*
3. *Z is the certified-zero exposure row, naming supplied zeta-zero packets inside the displayed window.*
4. *E is the exact-decider refusal row, recording that the carrier does not provide $\prod_s$ Decidable(ZetaZero(s)).*
5. *R is the rigidity refusal row, recording that the carrier does not provide the global RH route from every critical-strip zero to an* OnCritLine *witness.*
6. *G is the finite prime-window gluing row available only as a dependency of the displayed certified-zero exposure, not as a substitute for B, M, Z, E, R.*
7. *H records componentwise* hsame *transport for the strip window, mirror row, certified-zero row, refusal rows, gluing row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an RH-facing finite-window request through $B, M, Z, E, R, G, H, P, N$.*
9. *P is the* Pkg *provenance over* RHCriticalStripWindowCertificate$^{\uparrow}$*,* RHBoundaryKindSeparation$^{\uparrow}$*,* ZetaZeros$^{\uparrow}$*,* Complex$^{\uparrow}$*,* Real$^{\uparrow}$*,* FinitePrimeWindowGluing$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, M, Z, E, R, G, H, C, P, N$. It has no coordinate for an exact zero decider, an RH proof, a zero enumeration theorem, a quotient zero set, a hidden critical-line classifier, or global rigidity outside the displayed finite window.

Theorem 8868.2 (RH critical-strip window certificate NameCert obligations)**.** *For every accepted* $\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}$ *carrier*

$$W = (B, M, Z, E, R, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite critical-strip window, mirror fixed-surface row, certified-zero exposure row, exact-decider refusal row, rigidity refusal row, finite prime-window gluing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8868.1. The rows B, M, Z, E, R, G are displayed before transport, replay, provenance, and local naming are read.

The certified-zero row Z is window-local: it exposes supplied zero packets only inside B. The mirror row M records the fixed-surface geometry available on that same finite surface.

The refusal rows E and R are part of the carrier inventory. Since no other row has the type of a total zero decider or global RH rigidity route, transport through H and replay through C cannot manufacture either object. □

Theorem 8868.3 (RH critical-strip window certificate boundary separation)**.** *An accepted* RHCriticalStripWindowC *packet separates finite critical-strip inspection from exact zero decision and global RH rigidity. A consumer may read certified zero exposure inside B, but it may not infer a total zero decider or a proof that every critical-strip zero lies on the mirror fixed surface.*

Proof. Use the NameCert obligations of Definition 8868.2. They keep the finite strip window, mirror row, certified-zero exposure, exact-decider refusal, and rigidity refusal as separate rows.

The exact-decider refusal row E blocks a read of $\prod_s \mathsf{Decidable}(\mathsf{ZetaZero}(s))$. The rigidity refusal row R blocks a read of the global route from every certified critical-strip zero to an $\mathsf{OnCritLine}$ witness.

The finite prime-window gluing row G is only a sibling dependency for the displayed window. It does not change the carrier inventory, so the accepted packet remains a finite certificate rather than an exact decider or global RH proof. □

Falsifiable prediction: Any accepted $\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}$ consumer that reads an exact zeta-zero decider or a global RH rigidity proof from the finite window rows B, M, Z, G, H, C, P, N without displaying the refusal rows E, R refutes the carrier surface.

Independence witness: rh-boundary-kind-separation, zeta-zeros, complex, finite-prime-window-gluing, rh-free-zero-initiality: no listed sibling is bijective to $\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}$, because this packet binds a finite critical-strip window, mirror fixed-surface row, certified-zero exposure, exact-decider refusal, and rigidity refusal in one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}}$.

Constructive story: An $\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}$ packet is generated from a finite critical-strip window row, mirror fixed-surface row, certified-zero exposure row, exact-decider refusal row, rigidity refusal row, finite prime-window gluing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8868.1, Definition 8868.2, and Definition 8868.3 seed the finite RH critical-strip window certificate while separating mirror geometry from exact zero decision and global RH rigidity.

Formal verification: FormalStatus(NameCert$_{\text{RHCriticalStripWindowCertificate}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove RH, refute RH, decide all zeta zeros, enumerate all zeros, select a quotient zero set, supply a hidden critical-line classifier, close Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, mirror fixed-surface exposure, certified-zero routing, exact-decider refusal, rigidity refusal, gluing handoff, transport, replay, provenance, and local naming for BEDC.Derived.RHCriticalStripWindowCertificateUp.

Chapter 8869

A Concrete Naming Certificate for $\mathsf{SturmLiouville}^{\uparrow}$

Following Sturm and Liouville, the $\mathsf{SturmLiouville}^{\uparrow}$ packet records the regular-endpoint spectral ODE surface as finite BEDC NameCert data. It sits over the ODE carrier of chapter 193, the Wronskian determinant boundary of chapter 6877, the Hilbert-space source of chapter 91, the spectral-data package of chapter 214, and the terminal real-name seal of chapter 38. The packet names a displayed coefficient, boundary, Wronskian, Hilbert, and spectral-row handoff; it does not import host differential operators, quotient function spaces, selected eigenfunction bases, classical completeness, or an ambient $\mathsf{Real}^{\uparrow}$ spectral theorem.

Definition 8869.1 (Sturm–Liouville carrier)**.** *A* $\mathsf{SturmLiouville}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, B, W, H, T, R, E, K, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *coefficient-and-equation row for the inspected second-order scalar equation on the displayed interval.*
2. *B is the regular-endpoint boundary row, naming the two endpoint cells, boundary forms, separated or coupled boundary clauses, and the finite compatibility checks attached to O.*
3. *W is the* $\mathsf{Wronskian}^{\uparrow}$ *row used to compare endpoint solutions and boundary-admissible solution pairs.*
4. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *row carrying the inner-product and norm surface on which boundary-admissible solution rows may be compared.*
5. *T is the* $\mathsf{SpectralTheorem}^{\uparrow}$ *spectral-data row read only after the ODE, boundary, Wronskian, and Hilbert rows have been displayed.*
6. *R is the regular-endpoint resolvent packet, recording the finite Green-kernel, endpoint, and inhomogeneous-source reads that remain inside the same boundary surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for eigenvalue, norm, and endpoint scalar reads.*
8. *K records componentwise* hsame *transport for the ODE, boundary, Wronskian, Hilbert, spectral, resolvent, and Real-seal rows.*
9. *C records displayed* Cont *replay from coefficient requests through boundary admission, Wronskian comparison, Hilbert comparison, spectral-data exposure, resolvent readback, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SturmLiouville}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Wronskian}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{SpectralTheorem}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmLiouville}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host unbounded operator, quotient function equality, selected eigenbasis, endpoint distribution, classical compact-resolvent theorem, or ambient real completeness.

Theorem 8869.2 (Sturm–Liouville NameCert obligations)**.** *For every accepted* $\mathsf{SturmLiouville}^{\uparrow}$ *carrier*

$$S = (O, B, W, H, T, R, E, K, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SturmLiouville}^{\uparrow}}$ *surface consists exactly of ODE coefficient admission through O, regular-endpoint boundary admission through B, Wronskian comparison through W, Hilbert comparison through H, spectral-data exposure through T, regular-endpoint resolvent readback through R, terminal real sealing through E, componentwise* hsame *transport through K, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8869.1. The ODE, boundary, Wronskian, Hilbert, spectral, resolvent, and Real-seal rows are the only mathematical rows available before the structural support is read.

The ODE and boundary rows supply the regular-endpoint source. The Wronskian row compares endpoint solutions inside the same displayed boundary surface, and the Hilbert row supplies only the carried inner-product and norm comparison data.

The spectral row is downstream of those displays. It is consumed as a $\mathsf{SpectralTheorem}^{\uparrow}$ data packet over the same Hilbert-facing source, while the resolvent row records only finite Green-kernel and source reads attached to the boundary packet.

Finally, K, C, P, N transport, replay, source, and locally name the same route. Since no carrier coordinate stores a host differential operator, quotient function equality, selected eigenbasis, endpoint distribution, compact-resolvent theorem, or ambient completeness principle, the listed rows exhaust the local NameCert surface. □

Theorem 8869.3 (Sturm–Liouville regular-endpoint handoff)**.** *Every regular-endpoint read exported by an accepted* $\mathsf{SturmLiouville}^{\uparrow}$ *packet factors through*

$$\mathsf{Ode}^{\uparrow},\ B,\ \mathsf{Wronskian}^{\uparrow},\ \mathsf{Hilbert}^{\uparrow},\ \mathsf{SpectralTheorem}^{\uparrow},\ R,\ \mathsf{Real}^{\uparrow}.$$

The endpoint clauses are read before Wronskian comparison, the Wronskian row is read before Hilbert and spectral comparison, and the terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed only after the displayed spectral or resolvent row is visible.*

Proof. Start from the obligation surface of Definition 8869.2. It fixes O and B as the source of coefficient and endpoint data, so an endpoint consumer cannot begin from an unrecorded host operator.

Apply the carrier definition. The Wronskian row W is attached to the same ODE and boundary rows, and the Hilbert row H is available only as the comparison surface for those displayed solution rows.

The spectral-data row T and resolvent row R are downstream of O, B, W, H. Therefore every endpoint-facing scalar read reaches E through the displayed spectral or resolvent packet, not through a host eigenfunction argument or quotient function space. □

Theorem 8869.4 (Sturm–Liouville spectral packet nonescape)**.** *No accepted* $\mathsf{SturmLiouville}^{\uparrow}$ *spectral packet can escape the displayed rows*

$$O, B, W, H, T, R, E, K, C, P, N.$$

In particular, a consumer may read eigenvalue, orthogonality, normalization, or resolvent scalar data only through the $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{Wronskian}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{SpectralTheorem}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *route named by the carrier.*

Proof. Read the carrier and the NameCert obligations. The rows $O, B, W, H, T, R, E, K, C, P, N$ are the complete accepted packet, and each spectral read must preserve the same ODE, boundary, Wronskian, Hilbert, spectral, resolvent, and Real-seal order.

Use Definition 8869.3. It places endpoint reads before Wronskian comparison and Wronskian comparison before Hilbert and spectral consumption, so a later spectral packet cannot bypass the regular-endpoint route.

The structural rows then close the nonescape claim. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row keep eigenvalue, orthogonality, normalization, and resolvent scalar reads inside the finite packet, excluding host operators, quotient function spaces, selected eigenbases, and ambient completeness. □

Theorem 8869.5 (Sturm–Liouville Wronskian boundary)**.** *Every boundary-admissible solution comparison in an accepted* $\mathsf{SturmLiouville}^{\uparrow}$ *packet factors through the* $\mathsf{Wronskian}^{\uparrow}$ *row* W *over the same ODE row* O *and regular-endpoint row* B*. A consumer that compares endpoint solutions through Hilbert or spectral data without first displaying* W *is outside the local certificate.*

Proof. The carrier of Definition 8869.1 places W between the regular-endpoint source O, B and the Hilbert and spectral rows H, T. Thus boundary-admissible solution comparison has a displayed Wronskian coordinate before any Hilbert-facing read is available.

Apply Definition 8869.2. Wronskian comparison is a separate obligation row, not a consequence of host equality or a hidden classical zero theorem.

Finally use Definition 8869.4. Since every spectral or resolvent scalar read remains inside $O, B, W, H, T, R, E, K, C, P, N$, omitting W leaves the accepted certificate rather than producing another valid boundary comparison. □

Falsifiable prediction: A Sturm–Liouville spectral read that reaches Hilbert or spectral rows without first displaying ODE coefficient data, regular-endpoint boundary data, and the Wronskian comparison row refutes the local carrier.

Independence witness: ode, wronskian, hilbert, spectraltheorem, real: $\mathsf{SturmLiouville}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds ODE coefficients, regular endpoint boundary data, Wronskian comparison, Hilbert comparison, spectral-data exposure, resolvent readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SturmLiouville}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SturmLiouville}^{\uparrow}$ packet is generated from an ODE coefficient row, a regular-endpoint boundary row, a Wronskian comparison row, a Hilbert comparison row, a spectral-data row, a regular-endpoint resolvent packet, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmLiouville}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8869.1, Definition 8869.2, Definition 8869.3, Definition 8869.4, and Definition 8869.5 seed the regular-endpoint spectral ODE carrier, obligation, handoff, nonescape, and Wronskian-boundary surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmLiouville}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host unbounded operators, quotient function spaces, selected eigenfunction bases, classical compact-resolvent completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-row exactness, Wronskian comparison stability, Hilbert and spectral packet routing, resolvent readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SturmLiouvilleUp.

Chapter 8870

A Concrete Naming Certificate for $\mathsf{CantorBorelSet}^{\uparrow}$

Following Cantor and Kechris, the $\mathsf{CantorBorelSet}^{\uparrow}$ packet records the finite cylinder-window surface used when countable-name real routes inspect Borel-style subsets of Cantor space. It sits after the Cantor prefix carrier of chapter 1475, the cylinder basis of chapter 2153, the nested interval route of chapter 320, and the L10 real-name chain through chapter 126, chapter 301, and chapter 38. The packet names only displayed finite cylinder rows and countable-name windows; it is not a completed sigma-algebra object, quotient set equality, selected infinite branch, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 8870.1 (CantorBorelSet carrier)**.** *A* $\mathsf{CantorBorelSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, C, W, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *address row naming the finite binary prefixes currently inspected.*
2. *C is the cylinder-window row supplied through* $\mathsf{CantorCylinderBasis}^{\uparrow}$*.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the countable-name observations visible to the Borel-window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows when the request feeds a countable real-name route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after A, C, W, R have remained visible.*
6. *H records componentwise* hsame *transport for the Cantor address, cylinder window, stream window, regular readback, Real seal, replay, provenance, and local name.*
7. *K records displayed* Cont *replay from a cylinder-membership or countable-name request through A, C, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CantorBorelSet}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{CantorCylinderBasis}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBorelSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host Borel sigma-algebra, quotient equality of subsets, selected infinite branches, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8870.2 (CantorBorelSet obligation surface)**.** *For every accepted* $\mathsf{CantorBorelSet}^{\uparrow}$ *carrier* $B = (A, C, W, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CantorBorelSet}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cantor address row, cylinder-window row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The surface contains no hidden obligation for a completed Borel sigma algebra, quotient set equality, selected branch, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8870.1. The accepted packet exposes $A, C, W, R, E, H, K, P, N$ as separate finite rows, so every local obligation must be read from one of those coordinates.

The cylinder data is supplied by A and C. These rows point to Cantor-space prefixes and the cylinder basis, not to a completed host set of all Borel subsets.

The countable-name route is supplied by W, R, E. A request first reads finite $\mathsf{StreamName}^{\uparrow}$ windows, then $\mathsf{RegSeqRat}^{\uparrow}$ readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, K, P, N transport, replay, package, and name the same finite rows. Since no carrier coordinate contains a sigma-algebra completion, quotient equality, selected branch, countable-choice schedule, or ambient completeness principle, the obligation surface is exactly the displayed packet. □

Theorem 8870.3 (CantorBorelSet cylinder generation)**.** *Every public cylinder read exported by an accepted* $\mathsf{CantorBorelSet}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CantorSpace}^{\uparrow} \longmapsto \mathsf{CantorCylinderBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}.$$

The route exposes the Cantor prefix row A*, the cylinder-window row* C*, and the finite stream-window row* W *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is public. Thus cylinder generation remains a finite NameCert route, not a host powerset, completed sigma algebra, quotient subset equality, selected infinite branch, or countable-choice enumeration.*

Proof. Start from the Cantor address row A. It names only the finite binary prefixes currently visible in the accepted carrier.

Use the cylinder-window row C next. By construction, C is read through the cylinder basis of chapter 2153, so generation is by displayed finite cylinder windows.

Finally expose the stream window W. The $\mathsf{StreamName}^{\uparrow}$ row records exactly the bounded countable-name observations used by downstream consumers. Since R and E are downstream and no carrier coordinate names a host powerset or completed sigma algebra, the public cylinder generation route is exhausted by the displayed finite rows. □

Theorem 8870.4 (CantorBorelSet real route)**.** *Every countable-name real consumer of an accepted* $\mathsf{CantorBorelSet}^{\uparrow}$ *packet must read the route*

$$\mathsf{CantorCylinderBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before using the packet as a real-facing Borel-window input. The consumer receives C, W, R, E, H, K, P, N *as displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it receives no quotient stream equality, selected representative, selected branch, host Borel set equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the cylinder-generation theorem Definition 8870.3. It forces the read through the Cantor address, cylinder basis, and finite StreamName window before real-facing data is available.

Next read the regular rational row R. This is the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate in the carrier, so any countable-name real consumer must pass through the displayed readback rather than a quotient stream or selected representative.

Finally consume the terminal seal E and structural rows H, K, P, N. The seal is downstream of C, W, R, and the structural rows preserve the same displayed route. Therefore the real-facing Borel-window input is exactly the finite path through cylinder, stream, regular readback, and Real rows, with no hidden host Borel equality, branch selection, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBorelSet}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorBorelSet}^{\uparrow}}$, seed)

Scope closed: Definition 8870.1, Definition 8870.2, Definition 8870.3, and Definition 8870.4 seed the finite cylinder-window route from Cantor prefixes through StreamName, RegSeqRat, and Real rows for later countable-name consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorBorelSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed Borel sigma algebras, quotient set equality, selected infinite branches, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cylinder-window exactness, countable-name window discipline, regular readback, Real sealing, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.CantorBorelSetUp.

Constructive story: A $\mathsf{CantorBorelSet}^{\uparrow}$ packet is generated from a CantorSpace address row, CantorCylinderBasis window row, StreamName finite-window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8871

A Concrete Naming Certificate for $\mathsf{FellerSemigroup}^{\uparrow}$

Following Feller and Ethier–Kurtz on Markov transition semigroups, the $\mathsf{FellerSemigroup}^{\uparrow}$ packet records the finite route by which a displayed transition family preserves continuous observables on a compact metric state surface before any process-level or generator-facing consumer reads it. It sits after the compact state boundary of chapter 129, the continuous observable route of chapter 130, the Markov transition row of chapter 189, and the terminal real-value seal of chapter 38. The packet names a finite Markov-continuity carrier; it is not a host semigroup theorem, a selected stochastic process, a measure-space completion, quotient equality of kernels, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8871.1 (FellerSemigroup carrier)**.** *A* $\mathsf{FellerSemigroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, T, K, A, O, V, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *state row, exposing only the finite metric and compact-window data used by transition reads.*
2. *T is the finite time-window row, listing the semigroup times and composition comparisons inspected by the packet.*
3. *K is the* $\mathsf{MarkovChain}^{\uparrow}$ *transition-kernel row, recording the displayed Markov transition data available at the times in T.*
4. *A is the algebraic semigroup ledger for identity time, displayed composition, and compatibility of the transition rows in K.*
5. *O is the finite list of* $\mathsf{ContinuousMap}^{\uparrow}$ *observable rows on the compact state surface.*
6. *V is the* $\mathsf{Real}^{\uparrow}$ *value-read row for the observable evaluations before and after transition.*
7. *R is the Markov-continuity row saying that transition reads carry the displayed observables in O to displayed continuous observable rows over the same compact state surface.*
8. *H records componentwise* hsame *transport for the state, time, transition, semigroup, observable, value, and Markov-continuity rows.*
9. *C records displayed* Cont *replay from a transition request through observable readback and Markov-continuity exposure.*

10. *P is the* Pkg *provenance over* $\mathsf{FellerSemigroup}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FellerSemigroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, K, A, O, V, R, H, C, P, N$. It has no coordinate for a host probability space, selected sample paths, measure completion, quotient equality of kernels, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8871.2 (FellerSemigroup NameCert obligations)**.** *For every accepted* $\mathsf{FellerSemigroup}^{\uparrow}$ *carrier $F = (X, T, K, A, O, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FellerSemigroup}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric state row, finite time-window row, Markov transition row, semigroup ledger, continuous observable rows, real-value readback, Markov-continuity row, componentwise transport, displayed replay, provenance, and local naming. Every semigroup-facing consumer must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{MarkovChain}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

through those carrier rows before any Feller-continuity conclusion is available. The exported boundary is finite NameCert data for Markov-continuity, not a host stochastic-process construction, measure-space completion, selected path law, quotient kernel equality, or ambient completeness theorem.

Proof. Project the accepted carrier of Definition 8871.1. The displayed analytic rows are X, T, K, A, O, V, R, and the structural rows are H, C, P, N.

The compact state and transition rows are read first. The row X supplies the finite compact-metric state surface, T names the inspected time windows, and K supplies only the displayed Markov transition data at those times.

The semigroup and observable rows then bind the transition route. The ledger A records identity and composition checks, while O, V, R expose continuous observable rows, their real-value reads, and the Markov-continuity handoff.

Transport, replay, provenance, and local naming preserve that same route. Since the carrier has no host probability-space, selected-path, measure-completion, quotient-kernel, choice, trichotomy, or ambient-completeness coordinate, no local NameCert obligation can export one. □

Theorem 8871.3 (FellerSemigroup Markov-continuity handoff)**.** *Every Feller-continuity read exported by an accepted* $\mathsf{FellerSemigroup}^{\uparrow}$ *packet factors through the displayed route*

$$X, T, K, A \longmapsto O, V, R \longmapsto H, C, P, N.$$

In particular, the $\mathsf{MarkovChain}^{\uparrow}$ *transition row is consumed only together with the* $\mathsf{CompactMetric}^{\uparrow}$ *state row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *observable rows, and the* $\mathsf{Real}^{\uparrow}$ *value-read row. A downstream consumer cannot bypass those sibling rows by appealing to host kernel equality, a selected process realization, measure completion, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier inventory. The only transition-facing row is K, and it is indexed by the finite time-window row T over the compact state surface X.

Use the NameCert obligations of Definition 8871.2. They require any semigroup-facing consumer to expose the semigroup ledger A, observable rows O, and value-read row V before the Markov-continuity row R can be inspected.

The row R is therefore a handoff from displayed Markov transition data to displayed continuous observables. The componentwise transport H and replay C preserve the same compact state, transition, observable, and value rows, while P, N record the provenance and local certificate.

Because no excluded coordinate appears in the carrier, the handoff is exactly the finite Markov-continuity route through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{FellerSemigroup}^{\uparrow}$ consumer that reaches the Markov-continuity row without first reading the CompactMetricUp state row, ContinuousMapUp observable rows, MarkovChainUp transition row, RealUp value row, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: compactmetric, continuousmap, markovchain, real: $\mathsf{FellerSemigroup}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite time windows, Markov transition rows, semigroup composition, continuous observable rows, real-value readback, Markov-continuity handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FellerSemigroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FellerSemigroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8871.1, Definition 8871.2, and Definition 8871.3 seed the finite Markov-continuity certificate over compact state rows, continuous observable rows, Markov transition rows, Real value reads, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FellerSemigroup}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host stochastic-process construction, measure completion, selected sample paths, quotient equality of kernels, countable choice, trichotomy, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite time-window exposure, semigroup ledger exactness, observable-row handoff, Markov-continuity exposure, transport, replay, provenance, and local naming for BEDC.Derived.FellerSemigroupUp.

Constructive story: A $\mathsf{FellerSemigroup}^{\uparrow}$ packet is generated from a CompactMetric state row, finite time-window row, MarkovChain transition row, semigroup ledger, ContinuousMap observable rows, RealUp value-read rows, Markov-continuity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8872

A Concrete Naming Certificate for $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and explicit bounded tails, the $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$ packet records the finite witness by which a regular rational Cauchy schedule exposes a bounded tail before a Real or Completion consumer reads the source. It sits beside the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the selected bounded-tail window of chapter 8282, and the bounded Cauchy sequence packet of chapter 1522. The chapter names a displayed L10 bounded-tail witness; it does not claim quotient stream equality, a selected limit, countable choice, trichotomy, located suprema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8872.1 (Regular Cauchy bounded-tail carrier)**.** *A* $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, W, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, displaying the finite observation windows inspected by the tail request.*
2. *W is the selected window row consumed from* $\mathsf{CauchyBoundSelector}^{\uparrow}$*, recording the finite threshold and inspected bounded-tail windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, naming the dyadic radius, endpoint arithmetic, and comparison budget used by the bounded-tail read.*
4. *B is the* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *boundedness row, exposing the explicit tail bound attached to the same stream and tolerance data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, repacking the displayed stream windows and dyadic comparisons as regular rational Cauchy data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, W, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the schedule, selected window, dyadic ledger, boundedness row, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded-tail request through $S, W, D, B, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyBoundSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedTail}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, B, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, located suprema, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8872.2 (Regular Cauchy bounded-tail NameCert obligations)**.** *For every accepted* RegularCauchyBounde *carrier*

$$T = (S, W, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedTail}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName schedule, CauchyBoundSelector selected window, DyadicRatCore tolerance ledger, CauchySequenceBounded explicit bound, RegSeqRat readback, RealUp seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8872.1. The accepted tuple displays the schedule row S, selected window row W, dyadic tolerance ledger D, boundedness row B, regular readback row R, Real seal row E, and structural rows H, C, P, N.

The bounded-tail read is ordered. A consumer first opens the finite stream schedule and selected window, then reads the dyadic tolerance ledger and the boundedness row, and only then consumes the regular readback and terminal Real seal.

The structural rows preserve exactly that route. The transport and replay rows carry the displayed schedule, window, tolerance, bound, readback, and seal rows; provenance and local naming source the same packet. Since the classifier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, located suprema, compactness, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8872.3 (Regular Cauchy bounded-tail window boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read from an accepted* $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$ *packet reaches the seal row E only after the finite route*

$$S, W, D, B, R, E, H, C, P, N$$

has been displayed. The read may use the bounded-tail window and regular rational handoff as source data, but it cannot infer quotient equality of regular names, select a limit, choose tail bounds for all Cauchy predicates, decide trichotomy, form a located supremum, or import ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 8872.2. They force the StreamName schedule, CauchyBoundSelector window, DyadicRatCore tolerance ledger, CauchySequenceBounded bound, and RegSeqRat readback to be public before the Real seal can be read.

The seal row E is therefore downstream of a finite displayed bounded-tail route. Transport through H and replay through C preserve the same rows, while P and N supply provenance and local naming without adding further analytic structure.

Finally inspect the excluded coordinates of the carrier. No row stores quotient equality of regular names, selected limits, a global choice function, trichotomy, located suprema, compactness, or ambient Real completeness. Thus every Real or completion-facing read remains confined to the displayed bounded-tail boundary. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$ consumer that reads a $\mathsf{Real}^{\uparrow}$ or completion-facing bounded-tail source must display the $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{CauchyBoundSelector}^{\uparrow}$ selected window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{CauchySequenceBounded}^{\uparrow}$ bound, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local NameCert rows.

Independence witness: streamname, cauchy-bound-selector, dyadicratcore, cauchy-sequence-bounded, regseqrat, real: no listed sibling is bijective to $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$, because this packet jointly binds a finite stream schedule, selected bounded-tail window, dyadic tolerance ledger, explicit boundedness row, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedTail}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBoundedTail}^{\uparrow}$ packet is generated from a StreamName schedule, CauchyBoundSelector selected window, DyadicRatCore tolerance ledger, CauchySequenceBounded explicit bound, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8872.1, Definition 8872.2, and Definition 8872.3 seed the finite regular-Cauchy bounded-tail packet over stream schedules, selected windows, dyadic tolerances, explicit bounds, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, located suprema, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-window exposure, dyadic tolerance exactness, boundedness readback, regular rational handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyBoundedTailUp.

Chapter 8873

A Concrete Naming Certificate for CompactIntervalCantorBisection$^{\uparrow}$

Following Cantor's nested-interval bisection and the Bishop–Bridges finite-means reading of compact intervals, the CompactIntervalCantorBisection$^{\uparrow}$ packet records a closed interval search as finite dyadic interval data before any completion consumer reads a limit. It is anchored in the dyadic interval net of chapter 5966, the located interval bisection route of chapter 8867, the nested interval convergence surface of chapter 5987, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter treats Cantor bisection as a finite NameCert obligation surface over displayed intervals, not as open-cover compactness, a selected limit, countable choice, or ambient real completeness.

Definition 8873.1 (Compact interval Cantor bisection carrier)**.** *A* CompactIntervalCantorBisection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, L, N, R, Q, H, C, P, M)$$

with the following displayed rows:

1. *D is the* DyadicIntervalNet$^{\uparrow}$ *row listing the finite dyadic cells used at the displayed bisection depth.*
2. *L is the* LocatedIntervalBisection$^{\uparrow}$ *row recording the left/right interval decision ledger for each inspected cell.*
3. *N is the* NestedIntervalConvergence$^{\uparrow}$ *row that retains the nested enclosure relation through the finite bisection spine.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for endpoint windows and shrinking-radius budgets.*
5. *Q is the terminal* Real$^{\uparrow}$ *seal row reached only after the dyadic cells, located bisection ledger, nested enclosure row, and regular-rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic interval net, located bisection, nested enclosure, regular-rational readback, real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a compact interval request through finite dyadic subdivision, located bisection, nested enclosure, regular-rational readback, and the real seal.*
8. *P is the* Pkg *provenance over* CompactIntervalCantorBisection$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, LocatedIntervalBisection$^{\uparrow}$, NestedIntervalConvergence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. M is the local $\mathsf{NameCert}_{\mathsf{CompactIntervalCantorBisection}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, L, N, R, Q, H, C, P, M$. It has no coordinate for open-cover compactness, selected finite subcovers, countable-choice schedules, quotient equality of real names, host interval endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8873.2 (Compact interval Cantor bisection NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalCantorBisection}^{\uparrow}$ *carrier*

$$B = (D, L, N, R, Q, H, C, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalCantorBisection}^{\uparrow}}$ *surface consists exactly of carrier admission, finite dyadic interval-net exposure, located interval bisection exposure, nested interval convergence retention, regular-rational endpoint readback, terminal real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission projects the tuple $D, L, N, R, Q, H, C, P, M$. The finite dyadic interval-net row D is the only source of cells inspected by the bisection route.

The located bisection row L records the finite left/right interval decisions, and the nested enclosure row N keeps the resulting chain visible as a displayed interval spine. These rows precede every completion-facing read.

The readback row R converts endpoint windows and shrinking-radius budgets into regular rational data, and the seal row Q is downstream of that readback. Transport, replay, provenance, and local naming preserve the same route through H, C, P, M.

Since the carrier has no coordinate for open covers, selected subcovers, countable-choice schedules, quotient equality of real names, host endpoints, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8873.3 (Compact interval Cantor bisection five-row route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactIntervalCantorBisection}^{\uparrow}$ *packet reads the five visible mathematical rows*

$$\mathsf{DyadicIntervalNet}^{\uparrow} \longmapsto \mathsf{LocatedIntervalBisection}^{\uparrow} \longmapsto \mathsf{NestedIntervalConvergence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before inspecting transport, replay, provenance, or local naming data. The route exports no open-cover compactness theorem, selected limit, countable-choice schedule, quotient real-name equality, or ambient completeness principle.

Proof. Begin with the dyadic interval net row D. It supplies the finite cell list and dyadic endpoint arithmetic consumed by the located bisection ledger.

The located bisection row L records only displayed interval decisions. The nested convergence row N then retains the enclosure relation for the finite bisection spine without selecting a host-real limit.

The regular-rational row R reads back the endpoint windows and shrinking-radius budgets, and the real seal Q can be opened only after that readback. The structural rows H, C, P, M replay this same chain and add no excluded compactness or choice coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalCantorBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalCantorBisection}^{\uparrow}$ packet is generated from a finite dyadic interval net, a located bisection ledger, a nested interval convergence row, regular-rational endpoint readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalCantorBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8873.1, Definition 8873.2, and Definition 8873.3 seed the finite Cantor-bisection route over DyadicIntervalNet, LocatedIntervalBisection, NestedIntervalConvergence, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactIntervalCantorBisection}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, a selected limit, countable choice, quotient real-name equality, host interval endpoints, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, five-row route exactness, classifier nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalCantorBisectionUp.

Chapter 8874

A Concrete Naming Certificate for $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$

The $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ packet realizes the non-fixed orbit target of chapter 9334 as finite NameCert data. It names the critical-strip candidate row, mirror involution row, orbit-zero row, fixedness boundary, apartness witness, and obstruction ledger that must be displayed before an RH route may consume a non-fixed mirror orbit. The packet is a target surface for the RH orbit layer; it does not prove RH, does not assert zero-free strips, and does not collapse weak orbit exclusion into fixed-section evidence.

It depends on the critical-strip window carrier of chapter 8868, the boundary-kind separation surface of chapter 8864, and the fixed-statement separation route of chapter 8824. Rows S, J, Z, F, A, W consume their candidate-row, fixedness-boundary, apartness, and weak-exclusion rows before any mirror-orbit target is eligible for downstream RH routing.

Definition 8874.1 (RH mirror-orbit target carrier)**.** *An* $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, J, Z, F, A, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the critical-strip candidate row read from the carrier of Definition 9334.1.*
2. *J is the mirror-involution row recording the displayed map $s \mapsto 1 - \overline{s}$ and its finite replay data.*
3. *Z is the orbit-zero row naming the two possible zero reads $Z_\zeta(s)$ and $Z_\zeta(J(s))$.*
4. *F is the fixedness boundary row, separating fixed mirror rows from non-fixed mirror rows without deciding the equality globally.*
5. *A is the apart non-fixed witness row when a positive rational separation is displayed.*
6. *W is the weak-exclusion obstruction row, recording that weak orbit exclusion supplies only a refutation route unless additional located equality transport is present.*
7. *H records componentwise* hsame *transport for candidate, mirror, orbit-zero, fixedness, apartness, obstruction, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an orbit target request through the critical-strip candidate, mirror row, orbit-zero row, fixedness boundary, apartness witness, and obstruction ledger.*

9. *P is the* Pkg *provenance over* $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$*, the RH non-fixed orbit vision layer,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RHMirrorOrbitTarget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, J, Z, F, A, W, H, C, P, N$. It has no coordinate for an RH proof, global zero-free strip, classical excluded middle over mirror fixedness, quotient equality of candidate rows, host complex analysis, or unrecorded functional-equation transport.

Theorem 8874.2 (RH mirror-orbit target NameCert obligations)**.** *For every accepted* $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ *carrier*

$$O = (S, J, Z, F, A, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RHMirrorOrbitTarget}^{\uparrow}}$ *surface consists exactly of carrier admission, critical-strip candidate exposure, mirror-involution replay, orbit-zero exposure, fixedness-boundary retention, apart non-fixed witness exposure when present, weak-exclusion obstruction retention, component-wise transport, displayed replay, provenance, and local naming.*

Proof. Carrier admission projects $S, J, Z, F, A, W, H, C, P, N$. The critical-strip candidate row S and mirror row J are the only rows from which an orbit target can be read.

The orbit-zero row Z keeps the two zero reads visible as a finite disjoint target surface. The fixedness boundary row F records whether the consumer is reading a fixed row, a non-fixed row, or only a refutational boundary.

The apart row A is available only when a positive rational separation is displayed, and the obstruction row W records the gap between weak exclusion and fixed-section evidence. Transport, replay, provenance, and local naming preserve these rows without adding a hidden RH conclusion.

Since the carrier has no coordinate for an RH proof, global zero-free strip, classical decision of fixedness, quotient candidate equality, host complex-analysis oracle, or unrecorded functional-equation transport, the displayed rows exhaust the local NameCert surface. □

Theorem 8874.3 (RH mirror-orbit fixedness boundary)**.** *Every accepted* $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ *consumer that reads an orbit-zero row must keep the finite route*

$$S \longmapsto J \longmapsto Z \longmapsto F, W$$

visible before any apartness or fixed-section handoff is consumed. If the apart witness row A is absent, the packet records a weak refutation boundary only; it does not inhabit the fixed-section target.

Proof. Project the accepted carrier. The critical-strip source is S, the mirror involution is J, and orbit-zero exposure is Z.

The fixedness boundary F and obstruction row W are downstream of Z. They record whether the consumer has a fixed-row route, a non-fixed refutation route, or only a double-negation boundary.

The apartness row A is a separate positive witness row. Without it, the route has no coordinate for located equality transport, so it cannot be read as fixed-section evidence. The structural rows preserve this separation. □

Falsifiable prediction: Any accepted $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ consumer that reads an RH conclusion without displaying S, J, Z, F, W refutes the carrier; any accepted consumer that turns weak orbit exclusion into fixed-section evidence without an apartness or located-equality row refutes the fixedness boundary.

Independence witness: rh-critical-strip-window-certificate, rh-boundary-kind-separation, rh-route-closure-fixed-statement-separation: the mirror-orbit target is not bijective to any listed sibling because it jointly names the candidate row, mirror involution, orbit-zero row, fixedness boundary, apartness witness, weak-exclusion obstruction, transport, replay, provenance, and local naming.

The Lean follow-up for BEDC.Derived.RHMirrorOrbitTargetUp must provide a TasteGate surface whose non-smoke witness inspects the mirror row, orbit-zero row, fixedness boundary, apartness row, and obstruction row. A downstream positive gate must not treat weak non-fixed orbit exclusion as a fixed-section inhabitant unless a separate located-equality transport row is checked.

Closure status for $\mathsf{NameCert}_{\mathsf{RHMirrorOrbitTarget}^{\uparrow}}$.

Constructive story: An $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ packet is generated from a critical-strip candidate row, mirror-involution replay, orbit-zero exposure, fixedness boundary, apart non-fixed witness when present, weak-exclusion obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHMirrorOrbitTarget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8874.1, Definition 8874.2, and Definition 8874.3 seed the finite RH mirror-orbit target over critical-strip candidates, mirror involution, orbit-zero exposure, fixedness boundary, apartness, and weak-exclusion obstruction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RHMirrorOrbitTarget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove RH, zero-free strips, classical fixedness decidability, quotient candidate equality, host complex-analysis facts, or unrecorded functional-equation transport.

Upgrade path: Obligation closure requires checked carrier admission, mirror replay, orbit-zero exposure, fixedness-boundary retention, apartness handling, weak-exclusion obstruction, transport, replay, provenance, local naming, and TasteGate evidence for BEDC.Derived.RHMirrorOrbitTargetUp.

Chapter 8875

A Concrete Naming Certificate for CauchyCompletionInductionPrinciple$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the CauchyCompletionInductionPrinciple$^{\uparrow}$ packet names the finite row-by-row induction surface used by Cauchy-completion proofs before a terminal real seal is exposed. It is an L10 companion to the dense-source induction of chapter 1776, the quotient-free completion boundary of chapter 1639, the universal-property route of chapter 6041, the finite stream window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is not another completion object; it is the displayed induction rule that lets Real and Completion consumers traverse those rows without receiving quotient equality, selected limits, countable choice, host real equality, or ambient completeness.

Definition 8875.1 (Cauchy completion induction principle carrier)**.** *A* CauchyCompletionInductionPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, R, D, B, U, E, I, H, C, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *finite-window schedule whose listed entries are the only sequence coordinates inspected by the induction.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback reached from the same windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger assigning each finite induction row its visible dyadic budget.*
4. *B is the* BishopRegularCauchyCompletion$^{\uparrow}$ *boundary row, recording that the induction is completion-facing without selecting a completed limit.*
5. *U is the* RealCompletionUniversalProperty$^{\uparrow}$ *handoff row, read only after S, R, D, B have been displayed.*
6. *E is the terminal* Real$^{\uparrow}$ *seal produced by the displayed handoff.*
7. *I is the finite-row induction ledger: it orders the base row, successor row, and terminal seal handoff over S, R, D, B, U, E.*
8. *H records componentwise* hsame *transport for all displayed rows.*

9. *C records displayed* Cont *replay of the same finite-row route.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$, $\mathsf{BishopCompletionInduction}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, R, D, B, U, E, I, H, C, P, N$. It contains no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Theorem 8875.2 (Cauchy completion induction principle NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ *carrier $J = (S, R, D, B, U, E, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the eleven displayed rows;*
2. *finite-window visibility, requiring every induction premise to read only entries listed by S;*
3. *regular readback and tolerance compatibility, requiring R and D to answer the same displayed window requests;*
4. *completion-boundary routing, requiring B to expose the quotient-free* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary before any universal-property handoff is read;*
5. *universal-property handoff exactness, requiring U and E to be read downstream of S, R, D, B;*
6. *finite-row induction exactness, requiring I to store only the base, successor, and terminal handoff rows over the displayed route;*
7. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

No obligation row for quotient completion, selected limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking is exported by the packet.

Proof. Project the carrier of Definition 8875.1. Its visible rows are exactly the finite windows, regular readback, dyadic budget, completion boundary, universal handoff, terminal seal, induction ledger, transport, replay, provenance, and local naming rows.

The first three obligations are forced by the source route. An induction premise has no sequence coordinate outside S, and the readback row R is compatible with the tolerance ledger D only along the same displayed finite windows.

The completion-facing handoff is downstream. The boundary row B records that the route is a Bishop regular Cauchy completion route without a selected limit, while U and E are visible only after S, R, D, B have been displayed.

The induction ledger is local. It stores the base row, successor row, and terminal handoff row over the same displayed route, and H, C, P, N transport, replay, source, and name that route without adding another analytic coordinate.

Since all excluded objects are absent from the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8875.3 (Cauchy completion finite-row induction)**.** *Let $J = (S, R, D, B, U, E, I, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ *carrier. Every completion-facing induction step exported by J factors through the displayed finite route*

$$S \longmapsto R \longmapsto D \longmapsto B \longmapsto U \longmapsto E \longmapsto I \longmapsto H, C, P, N.$$

The base row and each successor row of the induction may inspect only the finite $\mathsf{StreamName}^{\uparrow}$ *entries in* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget* D*, the quotient-free* BishopRegularCauchyComplet *boundary* B*, the* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *handoff* U*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. No induction step may export quotient equality, a selected limit, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with an accepted carrier and project the finite-window row. The induction ledger I has no source outside S, so the base and successor rows can inspect only the listed StreamName entries.

Readback and tolerance control then fix the middle route. The row R is the displayed regular rational readback from those windows, and D is the dyadic budget attached to the same finite observations.

The completion handoff is reached only after the quotient-free boundary row is visible. The row B supplies the Bishop regular Cauchy completion boundary, while U and E record the universal-property handoff and terminal Real seal downstream of S, R, D, B.

Finally read I, H, C, P, N. The induction ledger stores the base, successor, and terminal handoff rows over the same displayed route, and the structural rows transport, replay, source, and name that route. Because no excluded object appears in the carrier, no induction step can export any of the forbidden completion or verification data. □

Theorem 8875.4 (Cauchy completion induction principle consumer boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ *packet receives only the finite-row induction route named by* $S, R, D, B, U, E, I, H, C, P, N$*. In particular, the consumer may reuse the dense-source discipline of* $\mathsf{BishopCompletionInduction}^{\uparrow}$*, the quotient-free boundary of* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*, and the handoff surface of* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$*, but only through the displayed rows of the packet. It receives no separate completion object, quotient stream equality, selected representative, countable-choice schedule, host equality service, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Project the accepted packet. The rows S, R, D, B, U, E, I give the only analytic and induction data, while H, C, P, N give only transport, replay, provenance, and local naming.

The sibling packets enter as named rows, not as external services. The dense-source discipline is cited by the provenance row P, the regular Cauchy completion boundary is stored in B, and the universal-property handoff is stored in U after the finite route has been displayed.

Thus a consumer cannot bypass the carrier to obtain another completion object or another equality service. It reads exactly the finite-row induction route and no excluded quotient, choice, host-equality, ambient-completeness, Lean, or bridge datum. □

Falsifiable prediction: Every $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ consumer read factors through one StreamName finite-window schedule, one RegSeqRat readback, one DyadicRatCore tolerance ledger, one BishopRegularCauchyCompletion boundary row, one RealCompletionUniversalProperty handoff, one terminal RealUp seal, one finite-row induction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bishop-completion-induction, bishop-regular-cauchy-completion, real-completion-universal-property, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it isolates the base-successor-terminal induction ledger over the completion-facing Cauchy route rather than naming the dense-source principle, the completion boundary, the universal handoff, or the terminal seal alone.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8875.1, Definition 8875.2, and Definition 8875.3 seed the finite-row induction route through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, Bishop regular Cauchy completion boundary, Real completion universal-property handoff, terminal Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding consumer nonescape, row induction exactness, and universal-property handoff exactness to BEDC.Derived.CauchyCompletionInductionPrincipleUp.

Constructive story: A $\mathsf{CauchyCompletionInductionPrinciple}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a BishopRegularCauchyCompletion boundary row, a RealCompletionUniversalProperty handoff, a terminal RealUp seal, a base-successor-terminal induction ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8876

A Concrete Naming Certificate for $\mathsf{FareyIntervalRefinement}^{\uparrow}$

Following Hardy–Wright on Farey sequences and Stern–Brocot intervals, the $\mathsf{FareyIntervalRefinement}^{\uparrow}$ packet records a finite rational interval-refinement route between adjacent Farey boundary rows and the real-facing readback surface. It consumes the Farey-neighbor certificate of chapter 1759, the Stern–Brocot branch route of chapter 1704, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the rational approximation handoff of chapter 1668, and the terminal real-name seal of chapter 38. The packet names a non-quotient finite refinement witness; it is not real completeness, a selected infinite branch, trichotomy, quotient equality, or a host density theorem.

Definition 8876.1 (Farey interval-refinement carrier)**.** *A* $\mathsf{FareyIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *row naming the adjacent rational boundary pair and its denominator budget.*
2. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *row naming the same interval by a finite branch word and mediant path.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint conversion, interval width, and requested precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *family of finite windows opened before a real-facing consumer reads the retained interval.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed windows and dyadic tolerances.*
6. *A is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff row recording the rational interval as a finite approximation window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after F, S, D, W, R, A have been exposed.*
8. *H stores componentwise* hsame *transport for the Farey row, Stern–Brocot row, dyadic tolerance ledger, stream windows, regular readback, approximation handoff, real seal, provenance, and local name.*
9. *C stores displayed* Cont *replay from F, S, D through W, R, A, E and then through the structural rows.*

10. *P is the* Pkg *provenance over* $\mathsf{FareyIntervalRefinement}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FareyIntervalRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, a selected infinite Stern–Brocot branch, trichotomy, countable choice, host order completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8876.2 (Farey interval-refinement NameCert obligations)**.** *For every accepted* $\mathsf{FareyIntervalRefinement}^{\uparrow}$ *carrier*

$$I = (F, S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FareyIntervalRefinement}^{\uparrow}}$ *surface consists exactly of Farey adjacency admission, Stern–Brocot branch agreement for the same rational interval, DyadicRatCore tolerance control, StreamName finite-window exposure, RegSeqRat readback, RealRationalApproximation handoff, RealUp seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 8876.1. The packet displays the Farey boundary pair, the Stern–Brocot branch row, dyadic endpoint tolerances, finite windows, readback, approximation handoff, real seal, and structural rows.

The interval agreement obligation is finite. The Farey row F supplies adjacent rational boundaries at the displayed denominator budget, while the Stern–Brocot row S supplies the finite mediant path for the same interval. Their compatibility is read before any downstream approximation row is available.

The real-facing obligations are downstream of that agreement. The dyadic tolerance ledger D controls endpoint conversion and interval width; the windows W expose the inspected prefixes; the readback R and approximation handoff A route the same finite interval to the seal row E.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no quotient representative, selected infinite branch, trichotomy witness, choice principle, host completeness theorem, or ambient Real completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8876.3 (Farey interval-refinement real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FareyIntervalRefinement}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{FareySequence}^{\uparrow},\ \mathsf{SternBrocot}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RealRationalApproximation}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed rows $F, S, D, W, R, A, E, H, C, P, N$. The packet can therefore serve Real and completion-facing consumers as a non-quotient rational interval-refinement witness, but it exports no proof of $\mathsf{Real}^{\uparrow}$ *completeness, no selected limiting real, and no quotient equality of real names.*

Proof. Begin with an accepted packet and project its displayed rows. The Farey row F and Stern–Brocot row S fix the same rational interval before any dyadic or real-facing row is consulted.

Next read the dyadic and stream route. The tolerance ledger D records the finite endpoint and width budget, and W records the finite windows opened under that budget. The regular readback R is assembled only from those displayed windows and tolerances.

The rational approximation handoff A is downstream of F, S, D, W, R. Only after that handoff has been displayed may the terminal row E be read as a $\mathsf{Real}^{\uparrow}$ seal.

Rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Thus every Real-facing consumer receives a finite interval-refinement witness and cannot extract a selected limit, quotient representative, trichotomy witness, countable-choice selector, host completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{FareyIntervalRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FareyIntervalRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8876.1, Definition 8876.2, and Definition 8876.3 seed the finite rational interval-refinement route from FareySequence and SternBrocot rows through DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, RealRationalApproximation handoff, RealUp seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FareyIntervalRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient real equality, selected infinite branches, selected limiting reals, trichotomy, countable choice, host order completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Farey and SternBrocot interval agreement, dyadic tolerance control, StreamName exposure, RegSeqRat readback, RealRationalApproximation handoff, RealUp seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.FareyIntervalRefinementUp.

Constructive story: A $\mathsf{FareyIntervalRefinement}^{\uparrow}$ packet is generated from a FareySequence adjacent-boundary row, a SternBrocot finite branch row for the same interval, DyadicRatCore endpoint tolerances, StreamName finite windows, RegSeqRat readback, RealRationalApproximation handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8877

A Concrete Naming Certificate for $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ packet records the finite route by which a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row enters the $\mathsf{Real}^{\uparrow}$ completion seal and remains readable at the $\mathsf{RealClassifier}^{\uparrow}$ boundary. It sits after the dyadic source ledger of chapter 291, the constant finite-window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the public classifier surface of chapter 1620. The certificate is a faithfulness surface for displayed dyadic sources, not a host embedding theorem, quotient stream equality, selected representative, trichotomy, countable choice, or ambient completeness principle.

Definition 8877.1 (Dyadic real embedding faithfulness carrier)**.** *A* $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, L, Q, C, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, including the dyadic numerator, scale, comparison ledger, and tolerance budget under inspection.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *constant-window row that presents the dyadic source to completion-facing reads.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the constant window and the displayed dyadic tolerance ledger.*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *Q is the* $\mathsf{RealClassifier}^{\uparrow}$ *boundary row that compares two embedded dyadic sources through one common classifier window.*
6. *C records the classifier conclusion visible at that boundary: equality of accepted classifier reads preserves equality of the displayed dyadic source surface.*
7. *H stores componentwise* hsame *transport for the dyadic source, constant window, regular readback, real seal, classifier row, conclusion row, replay, provenance, and local name.*
8. *K stores displayed* Cont *replay from the dyadic source through the constant window, regular readback, real seal, classifier boundary, and dyadic-source conclusion.*

9. *P records* Pkg *provenance over* $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-level injection function, quotient stream equality, selected real representative, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8877.2 (Dyadic real embedding faithfulness NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ *carrier* $F = (D, S, R, L, Q, C, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}}$ *surface consists exactly of six rows:*

1. *carrier admission for the dyadic source, constant window, regular readback, Real seal, classifier row, conclusion row, transport, replay, provenance, and local name;*
2. *dyadic source exposure, requiring every consumer to read the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D before it reaches the completion seal;*
3. *constant-window scheduling, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row S to be the only window through which the dyadic source is presented;*
4. *regular readback handoff, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to mediate the route from D, S to the* $\mathsf{Real}^{\uparrow}$ *seal L;*
5. *classifier faithfulness, requiring the* $\mathsf{RealClassifier}^{\uparrow}$ *row Q and conclusion row C to preserve the displayed dyadic source equality surface under equality of accepted classifier reads;*
6. *non-escape, requiring H, K, P, N to transport, replay, source, and locally name only the displayed finite packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 8877.1. Its analytic rows are exactly the dyadic source D, constant window S, regular readback R, Real seal L, classifier row Q, and conclusion row C.

The source, window, and readback obligations are forced by the ordered route. A completion-facing consumer first reads D, then opens only the constant $\mathsf{StreamName}^{\uparrow}$ window S, and then consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback R before reaching L.

The classifier obligation is local to Q and C. Equality at the accepted $\mathsf{RealClassifier}^{\uparrow}$ boundary is permitted to report only the dyadic source equality surface already displayed by D, S, R, L.

The structural rows H, K, P, N transport, replay, package, and name that same finite packet. Since the carrier omits host injection data, quotient stream equality, selected representatives, trichotomy, countable choice, and ambient completeness, no further obligation is available. □

Theorem 8877.3 (Dyadic real embedding faithfulness sibling anchors)**.** *Every accepted* $\mathsf{DyadicRealEmbeddingFaithf}$ *carrier is anchored in the ordered sibling chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RealClassifier}^{\uparrow}.$$

The $\mathsf{DyadicRatCore}^{\uparrow}$ *row supplies the source ledger,* $\mathsf{StreamName}^{\uparrow}$ *supplies the constant finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *supplies the regular readback,* $\mathsf{Real}^{\uparrow}$ *supplies the terminal seal, and* $\mathsf{RealClassifier}^{\uparrow}$ *supplies the common classifier boundary. No accepted read may bypass one of these rows by appealing to host real equality or quotient stream equality.*

Proof. Start from an accepted carrier $F = (D, S, R, L, Q, C, H, K, P, N)$. The tuple names exactly one source row, one window row, one readback row, one seal row, and one classifier row.

The dyadic source is therefore read as D, not as an unstated rational representative. The constant-window route is S, and the regular rational readback is R.

The terminal completion boundary is the $\mathsf{Real}^{\uparrow}$ seal L. The only public equality-facing consumer after that seal is the $\mathsf{RealClassifier}^{\uparrow}$ row Q, whose conclusion row C reports the displayed dyadic source equality surface.

Because H, K, P, N only transport, replay, source, and name the same packet, the listed sibling chain exhausts the accepted route. □

Theorem 8877.4 (Dyadic real embedding faithfulness public certificate)**.** *Let*

$$F = (D, S, R, L, Q, C, H, K, P, N)$$

and

$$F' = (D', S', R', L', Q', C', H', K', P', N')$$

be accepted $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ *carriers. Suppose Q and Q' are accepted* $\mathsf{RealClassifier}^{\uparrow}$ *boundary rows using the same common classifier window and the same dyadic tolerance ledger, and suppose equality at that boundary identifies the two public classifier decision surfaces. Then the public certificate may identify only the displayed dyadic source equality surface carried by D and D', transported through the constant windows S, S', the regular readbacks R, R', and the Real seals L, L'.*

Proof. Apply the sibling-anchor theorem Definition 8877.3 to both carriers. Both accepted routes must pass through the dyadic source, constant window, regular readback, Real seal, and RealClassifier boundary rows.

Use the common classifier window and dyadic tolerance ledger hypotheses. They ensure that the public classifier decision surfaces are read through the same finite inspection data rather than through separate representatives or host real values.

Now read the conclusion rows C and C'. Their only permitted content is preservation of the displayed dyadic source equality surface under equality of accepted classifier reads.

Finally inspect the structural rows. Since H, K, P, N and H', K', P', N' do not add quotient streams, selected representatives, trichotomy, countable choice, or ambient completeness, the public certificate cannot report anything beyond equality of the displayed dyadic source surfaces. □

Theorem 8877.5 (Dyadic real embedding faithfulness scope)**.** *The* $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ *certificate closes only the finite NameCert surface that says: constant* $\mathsf{StreamName}^{\uparrow}$ *lifts of* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, read back through* $\mathsf{RegSeqRat}^{\uparrow}$ *and sealed as* $\mathsf{Real}^{\uparrow}$*, preserve the displayed dyadic source equality surface when compared through a common* $\mathsf{RealClassifier}^{\uparrow}$ *window and dyadic tolerance ledger. It does not close a global injection theorem for a host function into* $\mathsf{Real}^{\uparrow}$*, quotient stream equality, selected canonical representatives, trichotomy, countable choice, or ambient completeness.*

Proof. The positive scope is exactly the route of Definition 8877.1. The certificate displays $D, S, R, L, Q, C, H, K, P, N$ and no other analytic coordinate.

The NameCert obligations of Definition 8877.2 require source exposure, constant-window scheduling, regular readback handoff, classifier faithfulness, and structural non-escape.

The public certificate theorem Definition 8877.4 then identifies what equality at the common classifier boundary may preserve: only the displayed dyadic source equality surface.

The excluded principles are absent from the carrier and from every obligation row, so they remain outside the chapter scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ packet is generated from a DyadicRatCore source ledger, a constant StreamName window, a RegSeqRat readback, a RealUp seal, a RealClassifier common-window boundary, a dyadic-source conclusion row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8877.1, Definition 8877.2, Definition 8877.3, Definition 8877.4, and Definition 8877.5 seed the finite dyadic-to-real faithfulness surface through DyadicRatCore, StreamName, RegSeqRat, RealUp, and RealClassifier rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a host injection theorem, quotient stream equality, selected canonical representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, constant-window scheduling, RegSeqRat readback, Real seal handoff, and RealClassifier dyadic-source faithfulness for BEDC.Derived.DyadicRealEmbeddingFaithfulUp.

Chapter 8878

A Concrete Naming Certificate for $\mathsf{BishopMeanValueInequality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopMeanValueInequality}^{\uparrow}$ packet records the constructive finite-window route by which derivative rows and a displayed norm bound produce an endpoint displacement bound. It sits beside the $\mathsf{FrechetDerivative}^{\uparrow}$ certificate of chapter 3634, the directional derivative surfaces of chapter 6344 and chapter 7621, the linear action row of chapter 48, the norm row of chapter 89, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet is a finite BEDC proof surface for a mean-value inequality; it is not a host mean-value theorem, a selected derivative point, an extremum principle, compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8878.1 (Bishop mean-value inequality carrier)**.** *A* $\mathsf{BishopMeanValueInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, D, B, \Delta, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite segment-window row, listing the two endpoint rows, the inspected partition windows, and the segment domain on which the comparison is read.*
2. *D is the derivative-row ledger, exposing only the* $\mathsf{FrechetDerivative}^{\uparrow}$*,* $\mathsf{GateauxDerivative}^{\uparrow}$*, and* $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ *reads used inside the displayed windows.*
3. *B is the norm-bound row, pairing each derivative read with a* $\mathsf{LinMap}^{\uparrow}$ *action row and a* $\mathsf{Norm}^{\uparrow}$ *bound row.*
4. *Δ is the endpoint-displacement row, naming the source endpoint difference and the target displacement to be bounded.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite slope budgets, partition lengths, and endpoint-distance observations.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed inequality after I, D, B, Δ, Q have been read.*
7. *H records componentwise* hsame *transport for segment windows, derivative rows, norm bounds, endpoint displacement, readback, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each finite-window derivative read through the norm-bound row to the endpoint displacement comparison.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopMeanValueInequality}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{GateauxDerivative}^{\uparrow}$, $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMeanValueInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, B, \Delta, Q, R, H, C, P, N$. It has no coordinate for a host mean-value theorem, a selected derivative point, an extremum argument, compactness of a segment, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8878.2 (Bishop mean-value inequality NameCert obligations)**.** *For every accepted* $\mathsf{BishopMeanValueInequality}^{\uparrow}$ *carrier $M = (I, D, B, \Delta, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopMeanValueInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for finite segment windows, derivative-row exposure through D, norm-bound accountability through B, endpoint-displacement exposure through Δ, regular rational readback through Q, Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, local naming through N, and nonescape excluding host mean-value selection, compactness, countable choice, quotient equality, and ambient completeness.*

Proof. Carrier admission is projection from Definition 8878.1. The rows I, D, B, Δ, Q, R expose the finite segment windows, derivative reads, norm bounds, endpoint displacement, regular rational readback, and terminal Real seal.

The derivative rows are read only as displayed data. The $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{GateauxDerivative}^{\uparrow}$, and $\mathsf{HadamardDirectionalDerivative}^{\uparrow}$ entries in D are paired with the $\mathsf{LinMap}^{\uparrow}$ and $\mathsf{Norm}^{\uparrow}$ rows in B, so the packet records bounded local linear behavior without selecting a host derivative point.

The structural rows then preserve the same finite route. Componentwise transport is supplied by H, replay by C, provenance by P, and local naming by N. Since the carrier lists no selected point, extremum principle, compact segment theorem, countable-choice schedule, quotient equality, or ambient completeness row, those objects are not NameCert obligations for $\mathsf{BishopMeanValueInequality}^{\uparrow}$. □

Theorem 8878.3 (Bishop mean-value inequality finite slope bound)**.** *Every accepted* $\mathsf{BishopMeanValueInequality}^{\uparrow}$ *carrier routes the endpoint displacement comparison through the displayed finite-window inequality*

$$I, D, B, \Delta, Q \longmapsto R, H, C, P, N.$$

A consumer may read the resulting bound only after the segment-window row, derivative-row ledger, norm-bound row, endpoint-displacement row, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback row have all been displayed. The route is a BEDC finite slope-bound surface, not a host mean-value theorem, selected derivative witness, compactness argument, countable-choice construction, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier inventory of Definition 8878.1. The only endpoint comparison coordinate is Δ, and it is downstream of the segment-window row I and the derivative ledger D.

Use the NameCert obligations of Definition 8878.2. They force every slope-bound read to expose the derivative rows and the norm-bound row B before the regular rational readback Q and Real seal R are available.

The displayed inequality route is therefore finite. The packet accumulates the inspected slope budgets through Q, seals the comparison through R, and transports, replays, sources, and names the same route through H, C, P, N. Because no carrier row names a selected derivative point or compactness principle, the endpoint displacement bound remains a finite-window BEDC proof surface. □

Falsifiable prediction: Every accepted $\mathsf{BishopMeanValueInequality}^{\uparrow}$ consumer that reaches an endpoint displacement bound without displaying the rows $I, D, B, \Delta, Q, R, H, C, P, N$ refutes the carrier surface.

Independence witness: frechetderivative, gateauxderivative, hadamarddirectionalderivative, linearmap, norm, regseqrat, real: none is bijective to $\mathsf{BishopMeanValueInequality}^{\uparrow}$ because this packet jointly carries finite segment windows, derivative rows, norm-bound rows, endpoint displacement, readback, sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMeanValueInequality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMeanValueInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8878.1, Definition 8878.2, and Definition 8878.3 seed the finite slope-bound certificate over segment windows, derivative rows, norm bounds, endpoint displacement, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMeanValueInequality}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close a host mean-value theorem, selected derivative point, extremum principle, compactness, countable choice, quotient real equality, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-row exposure, norm-bound accountability, endpoint-displacement exposure, readback, sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopMeanValueInequalityUp.

Constructive story: A $\mathsf{BishopMeanValueInequality}^{\uparrow}$ packet is generated from finite segment windows, derivative rows, norm-bound rows, endpoint displacement, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8879

A Concrete Naming Certificate for $\mathsf{LocatedUniformQuotient}^{\uparrow}$

Following Bourbaki and Kelley-style uniform-space quotient constructions, the $\mathsf{LocatedUniformQuotient}^{\uparrow}$ packet records the finite BEDC surface where a displayed equivalence relation on a located uniform source descends the uniformity only through saturated entourage evidence. It sits after the uniform-space carrier of chapter 339, the Real-uniform source of chapter 1624, and the metric witness discipline of chapter 57. The packet is a constructive quotient-uniformity object: it names source rows, relation rows, saturation rows, representative-change transport, replay, provenance, and local naming without forming host quotient types, selecting representatives, or importing ambient completeness.

Definition 8879.1 (Located uniform quotient carrier)**.** *A* $\mathsf{LocatedUniformQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, R, S, W, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source uniform carrier, read through* $\mathsf{UniformSpace}^{\uparrow}$ *and, when the source is metric or Real-facing, through the sibling rows of* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealUniformStructure}^{\uparrow}$*.*
2. *R is the displayed equivalence relation on the source carrier; it is stored as finite relation data and not as a host quotient type.*
3. *S is the saturated entourage row: each quotient-facing entourage request is accepted only after a source entourage of X has been shown stable under R on both coordinates.*
4. *W is the representative-change witness, recording that changing either displayed representative along R preserves the accepted saturated entourage request.*
5. *T is the transport row that carries accepted saturated-entourage reads through componentwise source-uniform, relation, and representative-change data.*
6. *H records componentwise* hsame *transport for X, R, S, W, T, C, P, N.*
7. *C records displayed* Cont *replay of the same quotient-uniformity route.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformQuotient}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{RealUniformStructure}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformQuotient}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, R, S, W, T, H, C, P, N$. It has no coordinate for host quotient types, selected representatives, quotient-set equality, hidden choice of section, ambient completeness, Lean verification, or bridge checking.

Theorem 8879.2 (Located uniform quotient NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformQuotient}^{\uparrow}$ *carrier $Q = (X, R, S, W, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformQuotient}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *source-uniform admission through X;*
2. *displayed equivalence-relation admission through R;*
3. *saturated-entourage admission through S, requiring every quotient-facing entourage request to come from a source entourage stable under R on both coordinates;*
4. *representative-change exactness through W, requiring each permitted representative change to preserve the same saturated-entourage read;*
5. *transport, replay, provenance, and local naming through T, H, C, P, N.*

No NameCert obligation is stored in a host quotient type, a selected representative, a quotient equality principle, a hidden section, ambient completeness, Lean verification, or bridge checking.

Proof. Project the carrier of Definition 8879.1. The accepted packet exposes exactly the source uniform row, the displayed relation row, the saturated entourage row, the representative-change witness, and the structural rows.

The source and relation obligations are immediate projections. A quotient-facing request has no data except the source uniform carrier X and the displayed equivalence relation R.

The saturated-entourage obligation is the only route from source uniformity to quotient uniformity. The row S records that the source entourage is stable under R in both coordinates before any quotient-facing consumer can read it.

Representative change is then local to W. It preserves the same saturated-entourage read and does not choose a canonical representative.

Finally T, H, C, P, N transport, replay, source, and name the displayed route. Since none of the excluded objects is a row of the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 8879.3 (Saturated-entourage descent)**.** *Let $Q = (X, R, S, W, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedUniformQuotient}^{\uparrow}$ *carrier. Every quotient-facing entourage read exported by Q descends from a displayed source-uniform entourage of X whose saturation is witnessed by S and whose representative changes are witnessed by W. The descent exports no host quotient type, selected representative, quotient-set equality, hidden section, ambient completeness, Lean verification, or bridge checking.*

Proof. Start with an accepted quotient-facing entourage read. Carrier admission gives access only to $X, R, S, W, T, H, C, P, N$.

The source of the read is therefore the source uniform row X. The relation row R supplies the only comparison data between representatives.

The row S supplies descent: it states that the source entourage is saturated with respect to R on both coordinates. Hence the quotient-facing read is not a new entourage generated by a host quotient object; it is the displayed saturated source entourage.

The row W verifies that changing representatives along R leaves the same saturated read available. The remaining structural rows preserve and name that route, so none of the excluded quotient or completeness data can be exported. □

Theorem 8879.4 (Representative-change transport)**.** *For every accepted* $\mathsf{LocatedUniformQuotient}^{\uparrow}$ *carrier* $Q = (X, R, S, W, T, H, C, P, N)$*, any two accepted reads that differ only by displayed* R*-equivalent representatives are transported by the same rows* W, T, H, C, P, N*. The transported read remains attached to the original source uniform carrier* X*, the relation row* R*, and the saturated-entourage row* S*, and it does not create a selected representative, a quotient type, an equality service on quotient classes, ambient completeness, Lean verification, or bridge checking.*

Proof. Take two accepted reads differing only by displayed R-equivalent representatives. Projection of the carrier shows that both reads share the same source uniform row X, relation row R, and saturated-entourage row S.

Representative change is handled by W. This row witnesses that the same saturated-entourage request remains accepted after changing either representative along R.

The transport row T and the structural rows H, C, P, N then carry the accepted read through componentwise transport, displayed replay, provenance, and local naming. These rows preserve the displayed source and saturation data; they do not add a representative selector or quotient-class equality coordinate.

Thus representative-change transport stays inside the finite carrier surface and cannot export any of the excluded host quotient, selection, completeness, Lean, or bridge data. □

Falsifiable prediction: Every accepted $\mathsf{LocatedUniformQuotient}^{\uparrow}$ consumer read that claims quotient-uniformity data must expose one source uniform carrier row, one displayed equivalence-relation row, one saturated-entourage row, one representative-change witness row, transport, componentwise hsame, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniformspace, real-uniform-structure, metric: $\mathsf{LocatedUniformQuotient}^{\uparrow}$ is not bijective to any listed sibling because it binds a displayed equivalence relation and saturated-entourage descent surface over a source uniform carrier, while those siblings name the source uniform, Real-uniform, or metric rows alone.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformQuotient}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformQuotient}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8879.1, Definition 8879.2, Definition 8879.3, and Definition 8879.4 seed the finite quotient-uniformity route through a source uniform carrier, displayed equivalence relation, saturated entourage row, representative-change witness, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformQuotient}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host quotient types, selected representatives, quotient-set equality, hidden sections, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, saturated-entourage descent, representative-change transport, nonescape, replay, provenance, and local naming for BEDC.Derived.LocatedUniformQuotientUp.

Constructive story: A $\mathsf{LocatedUniformQuotient}^{\uparrow}$ packet is generated from a source UniformSpace row, a displayed equivalence-relation row, a saturated-entourage row, a representative-change witness, transport, componentwise hsame, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8880

A Concrete Naming Certificate for QuarticSpectralCommonCarrier$^{\uparrow}$

Following standard recurrence theory, companion matrices, and affine spectral curves, the QuarticSpectralCommonC packet records the fixed quartic spectral object of section 9413.9 as finite NameCert data. It sits beside the polynomial carrier of chapter 50, the matrix carrier of chapter 49, the elliptic curve carrier of chapter 175, the real-polynomial carrier of chapter 1728, and the complex-number carrier of chapter 39. The certificate binds the recurrence source, companion matrix, spectral polynomial, affine curve, elliptic normalization, and projection ledger as one finite carrier surface; it is not an ambient algebraic-geometry theorem, a host spectral theorem, or an analytic continuation principle.

Definition 8880.1 (Quartic spectral common carrier)**.** *A* QuarticSpectralCommonCarrier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, M, P, A, E, L, H, C, G, N)$$

with the following displayed rows:

1. *R is the recurrence-source row, storing the numerator $N(t, y)$, denominator $D(t, y)$, coefficient readback, and order-four recurrence schedule of Definition 9413.86.*
2. *M is the companion-matrix row, storing the displayed 4×4 transfer matrix and its characteristic-polynomial readout.*
3. *P is the spectral-polynomial row, storing the quartic $\Pi(\lambda, y)$ as both the reciprocal denominator face and the matrix characteristic face.*
4. *A is the affine spectral-curve row, storing the finite equation $\Pi(\lambda, y) = 0$ and the projection coordinates (λ, y).*
5. *E is the elliptic-normalization row, storing the balanced-quartic normalization route to the displayed Weierstrass model.*
6. *L is the projection ledger, storing the generating-function, recurrence, companion-matrix, spectral-polynomial, affine-curve, elliptic-normalization, real-projection, and complex-coordinate reads.*
7. *H stores componentwise* hsame *transport for the recurrence, matrix, polynomial, curve, normalization, ledger, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for each projection in L.*

9. *G stores* Pkg *provenance over* $\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{EllipticCurve}^{\uparrow}$, $\mathsf{RealPolynomial}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary quartic family, external eigenvalue selection, analytic continuation, global branch choice, or a proof of the Riemann hypothesis.

Definition 8880.2 (Quartic spectral carrier NameCert obligations)**.** *For an accepted* QuarticSpectralCommonCarrie *carrier $Q = (R, M, P, A, E, L, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the recurrence source, companion matrix, spectral polynomial, affine curve, elliptic normalization, projection ledger, transport, replay, provenance, and local naming data;*
2. *recurrence-matrix compatibility, requiring the recurrence row R and matrix row M to expose the same order-four transfer law;*
3. *matrix-polynomial compatibility, requiring M and P to expose the same characteristic face $\Pi(\lambda, y)$;*
4. *polynomial-curve compatibility, requiring P and A to expose the same affine equation;*
5. *normalization compatibility, requiring E to be read only as the finite elliptic-normalization route of the displayed quartic curve;*
6. *projection non-escape, requiring L, H, C, G, N to transport, replay, source, and name only the displayed finite packet.*

No obligation row stores a host spectral theorem, arbitrary branch selection, analytic continuation, or global zeta-zero claim.

Theorem 8880.3 (Quartic spectral projection ledger)**.** *Every accepted* $\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}$ *packet exports a projection ledger whose public reads are exactly the recurrence source, the companion matrix, the spectral polynomial, the affine spectral curve, the elliptic-normalization route, the real-polynomial face, and the complex coordinate face displayed in R, M, P, A, E, L. The export cannot replace those rows by an external polynomial family, a free matrix spectrum, an unbounded branch selector, analytic continuation, or a global zeta-zero statement.*

Proof. Project the accepted carrier of Definition 8880.1. The recurrence row R, companion row M, spectral-polynomial row P, affine-curve row A, elliptic-normalization row E, and ledger row L are the only mathematical rows exposed before the structural data are read.

Use the NameCert obligations of Definition 8880.2. They bind R to M through the order-four transfer law, M to P through the characteristic face, P to A through the affine equation, and A to E through the displayed normalization route.

Finally read the structural rows H, C, G, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same finite carrier. Since the carrier omits arbitrary polynomial families, free spectra, branch selectors, analytic continuation, and global zeta-zero data, the public projection ledger is exactly the displayed quartic spectral surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}}$.

Constructive story: A $\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}$ packet is generated from a recurrence-source row, companion-matrix row, spectral-polynomial row, affine-curve row, elliptic-normalization row, projection ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}}$, seed)

Scope closed: Definition 8880.1, Definition 8880.2, and Definition 8880.3 seed the finite quartic spectral common-carrier surface over recurrence, matrix, polynomial, affine-curve, elliptic-normalization, real-polynomial, and complex-coordinate rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{QuarticSpectralCommonCarrier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, an arbitrary quartic-family theorem, analytic continuation, global branch selection, external spectral theory, or a global zeta-zero statement.

Upgrade path: Obligation closure requires checked carrier admission, recurrence-matrix compatibility, matrix-polynomial compatibility, polynomial-curve compatibility, normalization routing, and projection non-escape for BEDC.Derived.QuarticSpectralCommonCar

Chapter 8881

A Concrete Naming Certificate for LocalHalfSplitCertificate$^{\uparrow}$

The LocalHalfSplitCertificate$^{\uparrow}$ packet realizes the local two-branch half-split certificate of section 9413.60 as finite NameCert data. It sits beside the complex carrier of chapter 39, the zeta zero type-factorization carrier of chapter 8772, the zeta-zero carrier of chapter 73, the zeta-basic carrier of chapter 71, and the quartic spectral common-carrier packet of chapter 8880. The certificate stores a local zero island, half-split coordinate, real-defect ledger, phase-residual row, and refusal boundary; it is not a global zeta theorem, a Lee–Yang theorem, an analytic continuation proof, or a proof of the Riemann hypothesis.

Definition 8881.1 (Local half-split certificate carrier)**.** *A* LocalHalfSplitCertificate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Z, D, A, B, R, S, F, P, H, C, G, N)$$

with the following displayed rows:

1. *Z is the local zero-island row, storing the critical-strip point, local disk, dominant two-branch window, and zero-relevant boundary.*
2. *D is the dominance row for the two visible branches.*
3. *A and B are the two complex branch rows read through* Complex$^{\uparrow}$*.*
4. *R is the Rouché tail-control row for the remainder term.*
5. *S is the half-split row, storing the equal-length coordinate and the real-defect split formula.*
6. *F is the phase-residual row, storing the opposite-phase congruence and height-facing residual data after the equal-length row has been read.*
7. *P is the no-hidden-real-correction row, refusing a cofinal correction that cancels a nonzero real defect outside the displayed local island.*
8. *H stores componentwise* hsame *transport for the zero island, dominance, branch, tail, split, phase, refusal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay for the local certificate route.*
10. *G stores* Pkg *provenance over* LocalHalfSplitCertificate$^{\uparrow}$*,* Complex$^{\uparrow}$*,* ZetaZeroTypeFactorization$^{\uparrow}$*,* ZetaZeros$^{\uparrow}$*,* ZetaBasic$^{\uparrow}$*,* QuarticSpectralCommonCarrier$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalHalfSplitCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global zero enumeration, analytic continuation, functional equation proof, Lee–Yang stability theorem, or RH conclusion.

Theorem 8881.2 (Local half-split certificate NameCert obligations)**.** *For every accepted* $\mathsf{LocalHalfSplitCertificate}^{\uparrow}$ *carrier $L = (Z, D, A, B, R, S, F, P, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocalHalfSplitCertificate}^{\uparrow}}$ *surface consists exactly of seven finite obligation rows:*

1. *carrier admission for the local zero island, dominance, branch, tail, half-split, phase-residual, refusal, transport, replay, provenance, and local naming rows;*
2. *branch dominance, requiring D to expose the two active local branches before any zero certificate is exported;*
3. *complex branch readback, requiring A and B to be read through the displayed* $\mathsf{Complex}^{\uparrow}$ *branch rows;*
4. *tail control, requiring R to keep the local zero count inside the same island;*
5. *half-split exactness, requiring S to expose the equal-length coordinate and real-defect split formula;*
6. *phase-residual exposure, requiring F to remain present whenever the packet exports a half-split zero certificate;*
7. *non-escape, requiring P, H, C, G, N to refuse hidden real corrections and to transport, replay, source, and name only the displayed finite packet.*

Proof. Project the accepted carrier of Definition 8881.1. Its zero-facing rows are exactly Z, D, A, B, R, S, F, P, and its structural rows are exactly H, C, G, N.

The branch obligations force a consumer to read dominance and the two complex branches before the half-split row is available. The tail-control row then keeps the read inside the same local island.

The half-split row supplies equal length and the real-defect split formula. The phase-residual row remains a separate displayed row, so the certificate records the remaining zero-location data rather than hiding it in the real coordinate.

Finally the refusal and structural rows prevent a hidden real correction, unrelated zeta-zero carrier, global analytic principle, or RH conclusion from entering the packet. □

Theorem 8881.3 (Local half-split public certificate)**.** *Every accepted* $\mathsf{LocalHalfSplitCertificate}^{\uparrow}$ *packet exports a half-split zero certificate only through the displayed route*

$$Z \longmapsto D \longmapsto A, B \longmapsto R \longmapsto S \longmapsto F \longmapsto P \longmapsto H, C, G, N.$$

If the phase-residual row F is absent, the packet cannot export a half-split zero certificate.

Proof. Start with carrier admission. The local zero-island row Z, dominance row D, branch rows A, B, tail-control row R, half-split row S, phase-residual row F, and refusal row P are all explicit coordinates of the accepted packet.

Apply the NameCert obligations of Definition 8881.2. They require dominance before branch readback, branch readback before tail control, tail control before half-split exactness, and phase-residual exposure before export.

Thus removing F removes one of the required rows in the certificate route. The structural rows H, C, G, N can transport and replay only rows that are present, so the packet cannot export the half-split zero certificate without the phase-residual row. □

Falsifiable prediction: If the phase-residual row is absent, the packet cannot export a half-split zero certificate.

Independence witness: complex, zeta-zero-type-factorization, zeta-zeros, zeta-basic, quartic-spectral-common-carrier: each sibling supplies a source row or finite template, but none carries the combined local zero-island, two-branch dominance, Rouché tail-control, half-split exactness, phase-residual, and no-hidden-real-correction surface in one NameCert packet.

The Lean follow-up for BEDC.Derived.LocalHalfSplitCertificateUp.TasteGate must provide Nontrivial LocalHalfSpli and FieldFaithful LocalHalfSplitCertificate$^{\uparrow}$ instances with displayed projections for the zero-island, dominance, branch, tail-control, half-split, phase-residual, and refusal rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalHalfSplitCertificate}\uparrow}$.

Constructive story: A LocalHalfSplitCertificate$^{\uparrow}$ packet is generated from a local zero-island row, dominance row, two Complex branch rows, Rouche tail-control row, half-split row, phase-residual row, no-hidden-real-correction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalHalfSplitCertificate}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8881.1, Definition 8881.2, and Definition 8881.3 seed the finite local half-split certificate surface over two-branch dominance, complex branch readback, tail control, real-defect splitting, phase residuals, and no-hidden-real-correction refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalHalfSplitCertificate}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, analytic continuation, Lee–Yang stability, global zero enumeration, a zeta functional-equation proof, or RH.

Upgrade path: Obligation closure requires checked carrier admission, branch dominance, complex branch readback, tail control, half-split exactness, phase-residual exposure, and refusal non-escape for BEDC.Derived.LocalHalfSplitCertificateUp.

Chapter 8882

A Concrete Naming Certificate for $\mathsf{CantorFan}^{\uparrow}$

Following classical continuum theory, the $\mathsf{CantorFan}^{\uparrow}$ packet names the finite-prefix compact metric fan obtained by joining Cantor-space branch data to a single apex through dyadic height rows. It consumes the Cantor-space source of chapter 1475, the compact-metric and metric-space surfaces of chapter 129 and chapter 4230, the continuous-map interface of chapter 130, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. The chapter reconstructs the traditional fan as NameCert data over finite branch prefixes and dyadic heights; it does not use quotient equality of branches, host compactness, selected paths, or ambient real completeness.

Definition 8882.1 (Cantor fan carrier)**.** *A* $\mathsf{CantorFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, C, \Delta, W, R, D, A, E, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CantorSpace}^{\uparrow}$ *row naming the binary branch-prefix source.*
2. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *row naming the compact metric presentation visible to finite-net consumers.*
3. *C is the* $\mathsf{Met}^{\uparrow}$ *row whose distance observations are read only through displayed dyadic bounds.*
4. *Δ is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the apex-to-branch and branch-height continuity witnesses.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened on branch prefixes and height prefixes.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed branch-height regular Cauchy data.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for prefix agreement, height comparison, and apex collapse.*
8. *A is the finite apex row identifying the zero-height endpoint shared by all displayed branches.*
9. *E is the edge classifier row distinguishing an apex read from a positive-height branch read.*
10. *H stores componentwise* hsame *transport for all displayed rows.*
11. *Q stores displayed* Cont *replay from Cantor prefixes and dyadic heights through the metric and compact rows.*

12. *P is the* Pkg *provenance over* $\mathsf{CantorFan}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorFan}^{\uparrow}}$ *row.*

The carrier exports no quotient class of branches, selected infinite path, host continuum theorem, open-cover compactness proof, or ambient complete real line.

Theorem 8882.2 (Cantor fan NameCert)**.** *For every accepted* $\mathsf{CantorFan}^{\uparrow}$ *carrier*

$$F = (K, M, C, \Delta, W, R, D, A, E, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorFan}^{\uparrow}}$ *surface consists exactly of Cantor-space prefix admission, compact-metric finite-net exposure, metric distance exposure, continuous apex-and-height transport, StreamName finite-window exposure, RegSeqRat readback, DyadicRatCore tolerance control, apex classification, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The classifier is stable under any componentwise transport that preserves the displayed Cantor prefixes, dyadic height ledger, stream windows, regular readback, metric row, compact row, continuous-map row, and apex classifier. A consumer therefore receives the fan only as the finite ledger $K, M, C, \Delta, W, R, D, A, E, H, Q, P, N$.

Proof. Carrier exposure is projection from Definition 8882.1. The row K supplies the finite Cantor-prefix source, while D, W, R supply the dyadic height and prefix windows through which a consumer can inspect a branch-height point.

The classifier separates two displayed cases. In the apex case, A is read with the zero-height dyadic row and all branch prefixes collapse only through the same displayed tolerance ledger. In the positive-height case, E keeps the branch-prefix row visible together with the metric and compact rows C, M.

Continuity and compactness are consumed as finite NameCert rows, not as host principles. The row Δ records the apex and branch-height continuity handoff, and M records the compact-metric finite-net surface exposed to downstream consumers.

Finally H, Q, P, N transport, replay, source, and name that same finite packet. Since the carrier has no coordinate for quotient branch equality, selected paths, open-cover compactness, or ambient completeness, the listed rows exhaust the $\mathsf{CantorFan}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorFan}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorFan}^{\uparrow}}$, seed)

Scope closed: Definition 8882.1 and Definition 8882.2 seed the finite Cantor-prefix, compact-metric, metric, continuous-map, dyadic-height, StreamName, RegSeqRat, apex-classifier, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorFan}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cantor branches, selected infinite paths, host continuum compactness, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, compact-metric finite-net exposure, continuous apex transport, dyadic height control, stream-window readback, and local naming for BEDC.Derived.CantorFanUp.

Constructive story: A $\mathsf{CantorFan}^{\uparrow}$ packet is generated from CantorSpace prefixes, CompactMetric and MetricSpace rows, a ContinuousMap apex-height route, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, apex classification, componentwise transport, displayed replay, provenance, and a local NameCert row.

Chapter 8883

A Concrete Naming Certificate for $\mathsf{UniformTailAscoli}^{\uparrow}$

Following Arzela–Ascoli in its compact-metric constructive form, the $\mathsf{UniformTailAscoli}^{\uparrow}$ packet records finite uniform-tail data for an equicontinuous family on compact finite nets. It consumes the compact-uniform-continuity route of chapter 9237, the compact-metric source of chapter 129, the metric-space and continuous-map surfaces of chapter 4230 and chapter 130, the finite-net equicontinuity certificate of chapter 2362, and the L10 readback rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ from chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a finite NameCert surface for uniform tails and diagonal subfamilies, not a choice-selected subsequence theorem or a host compact-open theorem.

Definition 8883.1 (Uniform-tail Ascoli carrier)**.** *A* $\mathsf{UniformTailAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, M, C, F, E, U, T, W, R, D, S, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net source row.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row for the source and target distance observations.*
3. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *row that makes the finite-net read available.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row.*
5. *E is the finite-net equicontinuity row.*
6. *U is the shared uniform-modulus row read from K, M, C, F, E.*
7. *T is the uniform-tail row recording the finite stage after which the family is uniformly Cauchy on the displayed net.*
8. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used for the diagonal read.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed tails.*
10. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling all tail and net comparisons.*
11. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the readback.*
12. *L is the diagonal-subfamily ledger, finite at every displayed window.*

13. *H stores componentwise* hsame *transport for the listed rows.*
14. *Q stores displayed* Cont *replay from compact finite nets and equicontinuity rows to uniform tails and Real seals.*
15. *P is the* Pkg *provenance over* $\mathsf{UniformTailAscoli}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, FiniteNetEquicontinuit DyadicRatCore$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
16. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformTailAscoli}^{\uparrow}}$ *row.*

The carrier exports no choice-selected infinite subsequence, host compact-open topology, quotient equality of functions, or ambient completeness theorem.

Theorem 8883.2 (Uniform-tail Ascoli NameCert)**.** *For every accepted* $\mathsf{UniformTailAscoli}^{\uparrow}$ *carrier*

$$A = (K, M, C, F, E, U, T, W, R, D, S, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformTailAscoli}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, metric and compact-metric exposure, continuous-map family exposure, finite-net equicontinuity, shared uniform-modulus extraction, uniform-tail control, StreamName diagonal windows, RegSeqRat readback, DyadicRatCore tolerance control, RealUp seal boundary, diagonal-subfamily ledger rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The uniform-tail classifier is stable under any transport that preserves the compact net, equicontinuity row, shared modulus, tail stage, diagonal windows, regular readback, dyadic ledger, Real seal, and diagonal-subfamily ledger. Thus every consumer receives a finite Ascoli packet rather than a selected infinite subsequence.

Proof. Project the carrier of Definition 8883.1. The source rows K, M, C, F, E expose the compact finite net, metric observations, continuous family, and equicontinuity surface before any tail row is read.

Read the modulus and tail rows next. The shared modulus U is computed from the displayed compact-net and equicontinuity data, and the uniform-tail row T records a finite stage for the dyadic tolerance ledger D.

The diagonal part is windowed. The row L records only finite diagonal-subfamily data at each displayed $\mathsf{StreamName}^{\uparrow}$ window W, and R reads those windows as regular rational data before the terminal $\mathsf{Real}^{\uparrow}$ seal S is exposed.

Finally H, Q, P, N preserve the same finite route under transport, replay, provenance, and local naming. Since the carrier has no coordinate for a choice-selected infinite subsequence, quotient equality of maps, compact-open host topology, or ambient completeness, the stated rows exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformTailAscoli}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformTailAscoli}^{\uparrow}}$, seed)

Scope closed: Definition 8883.1 and Definition 8883.2 seed the finite compact-net, equicontinuity, uniform-modulus, uniform-tail, diagonal-window, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and local naming route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformTailAscoli}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a choice-selected infinite subsequence, host compact-open topology, quotient equality of functions, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, equicontinuity-to-uniform-modulus extraction, uniform-tail control, diagonal-window readback, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.UniformTailAscoliUp.

Constructive story: A UniformTailAscoli$^{\uparrow}$ packet is generated from compact finite nets, metric and compact-metric rows, continuous-map family rows, finite-net equicontinuity, a shared uniform modulus, a finite uniform-tail stage, StreamName diagonal windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, a diagonal-subfamily ledger, componentwise transport, displayed replay, provenance, and a local NameCert row.

Chapter 8884

A Concrete Naming Certificate for $\mathsf{RegularCauchyMedian}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyMedian}^{\uparrow}$ packet records the median of three regular Cauchy real-facing names by finite comparison windows. It reads the regular rational rows of chapter 301, the finite schedules of chapter 126, the dyadic comparison budgets of chapter 291, the sibling minimum and maximum routes of chapter 1528 and chapter 1502, and the terminal real-name seal of chapter 38. The packet names the middle window by displayed finite data; it does not import trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8884.1 (Regular Cauchy median carrier)**.** *A* $\mathsf{RegularCauchyMedian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, C, W_A, W_B, W_C, D, m_{AB}, j_{AB}, u, v, R, E, H, K, P, N)$$

with the following displayed rows:

1. A, B, C *are the three source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name rows.*
2. W_A, W_B, W_C *are the* $\mathsf{StreamName}^{\uparrow}$ *schedules used to inspect the three source names on the same finite precision request.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row, recording the endpoint tolerances, overlap slack, and dyadic cells consumed by the displayed pairwise comparisons.*
4. m_{AB} *is the* $\mathsf{RegularCauchyMin}^{\uparrow}$ *sibling route for the lower row of* A *and* B *over the inspected windows.*
5. j_{AB} *is the* $\mathsf{RegularCauchyMax}^{\uparrow}$ *sibling route for the upper row of* A *and* B *over the same inspected windows.*
6. u *is the* $\mathsf{RegularCauchyMax}^{\uparrow}$ *row for the displayed maximum of* m_{AB} *and* C*.*
7. v *is the* $\mathsf{RegularCauchyMin}^{\uparrow}$ *row for the displayed minimum of* u *and* j_{AB}*; it is the median row.*
8. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exported by* v*.*
9. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the readback row* R *has been exposed.*
10. H *records componentwise* hsame *transport for the three source rows, schedules, dyadic budget, sibling min and max rows, median row, readback, seal, replay, provenance, and local naming.*

11. *K records displayed* Cont *replay from a median request through* $W_A, W_B, W_C, D, m_{AB}, j_{AB}, u, v, R, E, H, P, N$.
12. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMedian}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyMin}^{\uparrow}$, $\mathsf{RegularCauchyMax}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMedian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host trichotomy, quotient stream equality, selected limits, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an unbounded comparison procedure.*

Theorem 8884.2 (Regular Cauchy median NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMedian}^{\uparrow}$ *carrier*

$$M = (A, B, C, W_A, W_B, W_C, D, m_{AB}, j_{AB}, u, v, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMedian}^{\uparrow}}$ *surface consists exactly of three-source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, synchronized* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget accounting, the displayed* $\mathsf{RegularCauchyMin}^{\uparrow}$ *and* $\mathsf{RegularCauchyMax}^{\uparrow}$ *sibling routes, median readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from those rows.*

Proof. Carrier admission is projection from Definition 8884.1. The source rows are A, B, C, the finite observation schedules are W_A, W_B, W_C, and the common comparison budget is D.

The operation rows are the sibling routes already named in the carrier. The rows m_{AB} and j_{AB} expose the lower and upper pairwise bounds for A and B; the row u then places C against the lower bound, and the row v places the result against the upper bound. Thus the median read is assembled from min and max rows over the same finite windows.

The readback and seal rows add no comparison principle. The row R is downstream of v, and the terminal seal E is downstream of R.

Finally H, K, P, N transport, replay, source, and name this displayed route. Since the carrier contains no other coordinate, the local NameCert exports no host trichotomy, quotient stream equality, selected representative, countable-choice schedule, ambient completeness, or unbounded comparison datum. □

Theorem 8884.3 (Regular Cauchy median finite-window route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyMedian}^{\uparrow}$ *carrier reaches the median row only through the finite route*

$$(A, B, C, W_A, W_B, W_C, D) \longrightarrow (m_{AB}, j_{AB}) \longrightarrow (u, v) \longrightarrow R \longrightarrow E.$$

The route is a displayed finite-window median operation over regular Cauchy names. It is not a trichotomy on completed reals, not a quotient-real equality principle, not a selected limit, and not an ambient completeness argument.

Proof. Start with a consumer of the accepted carrier. By Definition 8884.2, it must first expose the three regular rational sources, the three finite stream schedules, and the common dyadic comparison budget.

Next read the sibling operation rows. The first pair of rows, m_{AB} and j_{AB}, is supplied by the finite $\mathsf{RegularCauchyMin}^{\uparrow}$ and $\mathsf{RegularCauchyMax}^{\uparrow}$ chapters over A and B. The second pair, u and v, uses the same sibling routes to compare C with the displayed lower and upper rows.

The median row is therefore v, and the readback row R is downstream of that displayed operation packet. The terminal $\mathsf{Real}^{\uparrow}$ seal E is reached only after R has been exposed.

Transport through H and replay through K preserve exactly the same finite route. The provenance and local naming rows source the route but do not add a host comparison, quotient stream, selected representative, countable choice, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMedian}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMedian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8884.1, Definition 8884.2, and Definition 8884.3 seed the finite median route for three regular Cauchy names through RegSeqRat, StreamName, DyadicRatCore, RegularCauchyMin, RegularCauchyMax, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMedian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or unbounded comparison schedules.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic budget accounting, min and max sibling-route compatibility, median readback stability, Real seal handoff, transport, replay, provenance, local naming, and non-escape proofs for BEDC.Derived.RegularCauchyMedianUp.

Constructive story: A $\mathsf{RegularCauchyMedian}^{\uparrow}$ packet is generated from three RegSeqRat source rows, three StreamName schedules, a DyadicRatCore comparison budget, RegularCauchyMin and RegularCauchyMax sibling routes, a median readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8885

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ packet names the finite tail-gluing step used when regular rational Cauchy data are spliced after a shared prefix and then read as the same real-name route. It sits on the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter records a finite NameCert route for tail agreement and gluing stability; it does not assert quotient stream equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8885.1 (Bishop regular-Cauchy tail-gluing carrier)**.** *A* $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, D, R, T, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, listing the finite prefix and tail windows inspected by the gluing request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radius, endpoint arithmetic, and comparison budget used at the splice.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two regular rational tails.*
4. *T is the tail-gluing row: it records the common prefix, the tail threshold, and the proof that all inspected post-threshold windows agree within the displayed dyadic budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after S, D, R, T have been displayed.*
6. *W is the finite window-stability row, recording that later consumers may enlarge the inspected window only by reusing the same threshold and dyadic ledger.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-readback, tail-gluing, seal, stability, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the same finite gluing route.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, D, R, T, E, W, H, C, P, N$. It contains no coordinate for quotient equality of streams, selected real representatives, countable choice, trichotomy, located suprema, host real equality, or ambient completeness.

Theorem 8885.2 (Bishop regular-Cauchy tail-gluing NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ *carrier $G = (S, D, R, T, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailGlu}}$ *surface consists exactly of carrier admission for the StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, tail-threshold gluing row, RealUp seal, finite window-stability row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8885.1. The analytic rows are exactly the finite stream schedule S, dyadic tolerance ledger D, regular rational readback R, tail-gluing row T, terminal seal E, and window-stability row W.

The gluing row is downstream of finite data. It can compare tails only after the inspected prefix, threshold, dyadic budget, and regular readback have been displayed, so it has no hidden equality service for all streams.

The structural rows preserve the same route. Transport H, replay C, provenance P, and local naming N certify the displayed packet without adding quotient equality, selected representatives, choice, trichotomy, host equality, or ambient completeness. □

Theorem 8885.3 (Bishop regular-Cauchy tail-gluing window stability)**.** *Let $G = (S, D, R, T, E, W, H, C, P, N)$ be an accepted* $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ *carrier. Every Real or Completion consumer that enlarges the inspected post-threshold window must reuse the same finite route*

$$S, D, R, T, E, W, H, C, P, N.$$

The enlarged read may inspect more StreamName entries only through the displayed tail threshold and DyadicRatCore ledger; it cannot replace the glued tail by quotient equality, a selected representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8885.2. They force every consumer to expose $S, D, R, T, E, W, H, C, P, N$ before reading the tail-gluing surface.

The stability row W records the only permitted enlargement. It extends the inspected StreamName window while keeping the same threshold, dyadic budget, and regular readback rows, so the post-threshold comparison remains tied to the displayed finite packet.

Finally inspect the excluded carrier coordinates. Since no quotient equality, selected representative, choice schedule, trichotomy, host equality, or ambient completeness row exists, an enlarged consumer read remains the same finite tail-gluing route rather than a new real-equality principle. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ consumer read displays StreamName prefix and tail windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a tail-threshold gluing row, a RealUp seal, a window-stability row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ is not bijective to any listed sibling because it jointly records a finite prefix, a tail threshold, dyadic splice tolerances, regular rational readback, terminal Real sealing, window stability, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8885.1, Definition 8885.2, and Definition 8885.3 seed the finite regular-Cauchy tail-gluing route over stream windows, dyadic tolerances, regular rational readback, tail thresholds, Real sealing, window stability, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, regular readback compatibility, tail-threshold gluing, window stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCauchyTailGluingUp.

Constructive story: A $\mathsf{BishopRegularCauchyTailGluing}^{\uparrow}$ packet is generated from StreamName prefix and tail windows, DyadicRatCore tolerance data, RegSeqRat readback, a tail-threshold gluing row, a RealUp seal, a window-stability row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8886

A Concrete Naming Certificate for $\mathsf{DyadicCompletionFunctor}^{\uparrow}$

Following Bishop–Bridges constructive completion practice, the $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ packet records the finite map-level route that sends displayed dyadic rational data into the regular Cauchy real-name boundary. It sits after the dyadic tolerance source of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the extension ledger of chapter 397. The packet names a functorial completion surface for displayed maps; it is not a quotient completion, a selected limit, countable choice, host real equality, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 8886.1 (Dyadic completion functor carrier)**.** *A* $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, U, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, exposing dyadic numerators, scales, tolerance comparisons, and finite refinement data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which each dyadic source and map request is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row reached after D and S have been displayed.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the completed source and target names.*
5. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row, read only after the dyadic, stream, regular-rational, and Real-seal rows have been projected.*
6. *M is the map ledger, recording identity preservation and composition preservation for displayed dyadic maps through the same D, S, R, E, U route.*
7. *H records componentwise* hsame *transport for dyadic data, stream windows, regular rational readback, Real seals, universal-property rows, map ledgers, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a dyadic map request through D, S, R, E, U, M, H.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicCompletionFunctor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, U, M, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, host-level function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionFunctorUp.DyadicCompletionFunctorTasteGate single carrier alignment`

Theorem 8886.2 (Dyadic completion functor NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCompletionFunctor}$ *carrier*

$$F = (D, S, R, E, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCompletionFunctor}^{\uparrow}}$ *surface consists exactly of dyadic source admission through D, finite window inspection through S, regular rational readback through R, terminal Real sealing through E, universal-property extension through U, identity and composition preservation through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. Every map-level completion consumer must read*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow}$$

before treating the completed map as public.

Proof. Project the carrier of Definition 8886.1. The source and observation rows are D and S, so a map consumer first sees dyadic tolerance data and finite stream windows.

Read R and E next. The regular rational row is the only readback route from the finite windows, and the Real seal is downstream of that readback rather than a selected host limit.

The universal-property row U and map ledger M then record the functorial part. Identity and composition are preserved as finite ledger reads over the same dyadic, stream, regular, and Real-seal route; no host function equality or quotient completion is introduced.

Finally H, C, P, N transport, replay, source, and name the displayed packet. Since the carrier has no coordinates for quotient streams, selected representatives, countable choice, host equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 8886.3 (Dyadic completion functor map route)**.** *Every completed-map read exported by an accepted* $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CompletionUniversalProperty}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

The functorial read preserves displayed identities and displayed compositions through the map ledger M and never bypasses the dyadic tolerance ledger, finite stream windows, regular rational readback, Real seal, or universal-property extension row.

Proof. Apply Definition 8886.2. It places D, S, R, E, U before the map ledger M, so every completed-map read is already tied to the displayed dyadic-to-Real completion route.

The identity and composition clauses are ledger clauses inside M. They compare the completed map reads only after the same dyadic, stream, regular-rational, Real-seal, and universal-property rows have been exposed.

Transport and replay through H and C preserve those rows componentwise, while P and N source and name the same finite packet. Therefore the public map route cannot be replaced by quotient completion equality, selected representatives, countable choice, host function equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ consumer that reaches a completed map without displaying $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, completionuniversalproperty: $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds the dyadic source, finite window, regular readback, Real seal, universal-property extension, identity ledger, composition ledger, transport, replay, provenance, and local naming as one map-level completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCompletionFunctor}^{\uparrow}$ packet is generated from a DyadicRatCore source row, StreamName finite windows, RegSeqRat readback, a RealUp seal, a CompletionUniversalProperty extension row, identity and composition ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8886.1, Definition 8886.2, and Definition 8886.3 seed the finite map-level completion route from dyadic data to Real-facing completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host function equality, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic source exposure, finite window inspection, regular rational readback, Real seal nonescape, universal-property extension, identity preservation, composition preservation, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCompletionFunctorUp.

Chapter 8887

A Concrete Naming Certificate for $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$

Following Euler and Maclaurin, and in the Bishop–Bridges constructive reading of finite summation formulae, the $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$ packet records only the finite summation budget needed to read a bounded Euler–Maclaurin correction as BEDC data. It refines the broader summation-remainder route of chapter 2059, keeps the terminal value on the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, reads rational approximants through chapter 301, and sends every tolerance request through the $\mathsf{DyadicRatCore}^{\uparrow}$ surface of chapter 291. The chapter names a finite summation-window object; it does not assert a host asymptotic expansion, an infinite series theorem, an integral theorem, a chosen limiting process, quotient equality, countable choice, trichotomy, or ambient real completeness.

Definition 8887.1 (Euler–Maclaurin finite-sum carrier)**.** *An* $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, L, U, B, D, Q, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite summation-window row. It contains the lower and upper indices, the inspected term rows, and the finite partial-sum observations allowed to enter the budget.*
2. *L and U are endpoint rows for the boundary evaluations, including the* $\mathsf{Real}^{\uparrow}$ *names read at the lower and upper edges of the window.*
3. *B is the Bernoulli-correction row. It records the finite correction order, the coefficient rows requested at that order, and the boundary slots to which those coefficients are applied.*
4. *D is the derivative-window row. It lists only the derivative observations and interval windows used by the finite correction and remainder rows.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder-budget row, carrying the positive tolerance, split budget, and refinement data for the displayed finite window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the summation window, endpoint rows, Bernoulli-correction row, derivative-window row, and dyadic remainder budget.*
7. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after W, L, U, B, D, Q, R have been displayed.*

8. *H records componentwise* hsame *transport for the summation window, endpoint rows, Bernoulli-correction row, derivative-window row, dyadic budget, regular rational readback, terminal seal, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay through*

$$W, L, U, B, D, Q, R \longmapsto Z \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$, $\mathsf{EulerMaclaurin}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, L, U, B, D, Q, R, Z, H, C, P, N$. It has no coordinate for an unbounded series, a host integral, a hidden asymptotic remainder, a selected limit, quotient equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 8887.2 (Euler–Maclaurin finite-sum NameCert obligations)**.** *For every accepted* EulerMaclaurinFiniteSun *carrier*

$$F = (W, L, U, B, D, Q, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *finite-window admission, requiring the indices, inspected term rows, and partial-sum observations in W to be displayed before any correction row is read;*

2. *endpoint readiness, requiring both endpoint rows L, U to be opened before boundary corrections or terminal seals are named;*

3. *Bernoulli-correction readiness, requiring the finite coefficient ledger B to expose its order and boundary slots before it contributes to the budget;*

4. *derivative-window readiness, requiring D to list the derivative observations and interval windows used by the correction and remainder rows;*

5. *dyadic remainder budgeting, requiring every final error request to factor through the positivity, split-budget, and refinement data in Q;*

6. *regular rational readback, requiring R to expose rational approximants for W, L, U, B, D, Q before the terminal* $\mathsf{Real}^{\uparrow}$ *seal Z is opened;*

7. *classifier transport, requiring H and C to preserve the same W, L, U, B, D, Q, R, Z inventory under* hsame *and displayed* Cont *replay;*

8. *provenance and local naming, requiring P, N to source and name only the finite packet above, with no host asymptotic expansion, unbounded summation, host integral, selected limit, quotient equality, countable choice, trichotomy, ambient real completeness, Lean verification, or bridge checking.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8887.1. The accepted packet displays $W, L, U, B, D, Q, R, Z, H, C, P, N$, so every obligation must be read from one of those rows.

The finite-window and endpoint obligations come first. A consumer cannot reach the correction row until the summation bounds and inspected term rows in W are visible, and it cannot form boundary corrections until both endpoint rows L, U have been opened.

The correction and derivative obligations are rowwise. The Bernoulli-correction row B supplies only the finite coefficient ledger requested by the order of the packet, while D supplies the derivative observations and interval windows used by that same finite order.

The remainder obligation is forced by Q. Positivity, split-budget data, and refinements are displayed there before the regular rational readback R can name approximants for the whole budget and before the terminal real seal Z can be opened.

Finally, H and C replay the same inventory under transport, and P, N source and name only that replayed packet. Since the carrier has no additional coordinate, none of the excluded host-analytic or choice-based data can be read from an accepted packet. □

Theorem 8887.3 (Euler–Maclaurin finite-sum budget handoff)**.** *Every downstream quadrature, summation, or real-series consumer of an accepted* $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$ *packet must read the finite summation window* W*, endpoint rows* L, U*, Bernoulli-correction row* B*, derivative-window row* D*, dyadic remainder budget* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* Z*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N *before treating the packet as an Euler–Maclaurin budget. The handoff is therefore a finite BEDC budget route, not an invocation of the broader Euler–Maclaurin theorem as a host asymptotic principle.*

Proof. Let $F = (W, L, U, B, D, Q, R, Z, H, C, P, N)$ be accepted. By Definition 8887.1, these are exactly the rows available to any consumer.

The consumer first opens W for the finite summation interval and inspected terms, then opens L, U for endpoint data. It may then read B and D for the finite correction and derivative windows requested by the packet.

The error route passes through Q and then through R, so every tolerance and rational approximation is tied to the displayed dyadic budget and regular readback. Only after this route is visible may the terminal seal Z be consumed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Thus the handoff exports exactly the finite budget surface and gives later consumers no route to a host asymptotic theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}}$, seed)

Scope closed: Definition 8887.1, Definition 8887.2, and Definition 8887.3 seed the finite summation-window, endpoint, Bernoulli-correction, derivative-window, dyadic remainder, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host asymptotic expansion, unbounded summation theorem, host integral theorem, selected limiting process, quotient equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, endpoint readiness, Bernoulli-correction readiness, derivative-window readiness, dyadic remainder budgeting, RegSeqRat readback, Real seal handoff, classifier transport, and non-escape for BEDC.Derived.EulerMaclaurinFiniteSumUp.

Constructive story: An $\mathsf{EulerMaclaurinFiniteSum}^{\uparrow}$ packet is generated from a finite summation-window row, endpoint RealUp rows, Bernoulli-correction row, derivative-window row, DyadicRatCore remainder budget, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 8888

A Concrete Naming Certificate for $\mathsf{FiniteRadiusContraction}^{\uparrow}$

Following Banach and Picard, the $\mathsf{FiniteRadiusContraction}^{\uparrow}$ packet records the local contraction-budget data needed to run a displayed Picard window inside a fixed metric radius. It sits over the metric source of chapter 57, the regular rational readback of chapter 301, the stream-window route of chapter 126, and the terminal real-name seal of chapter 38. The packet names only a finite local budget: it does not assert a global fixed-point theorem, a complete ambient space, a selected limit outside the displayed window, quotient equality of real names, countable choice, or a host contraction principle.

Definition 8888.1 (Finite-radius contraction carrier)**.** *A* $\mathsf{FiniteRadiusContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, r, q, W, B, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the carried metric and the finite distance reads used by the local argument.*
2. *r is the positive radius row for the local ball being inspected.*
3. *q is the contraction-constant row, displayed as a rational bound with $0 \leq q < 1$ in the packet's rational readback.*
4. *W is the Picard window row, listing the finitely many iterates and distance requests opened from the chosen seed.*
5. *B is the Cauchy-tail budget row, recording the finite geometric tail estimates consumed inside the radius r.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the displayed iterate stream window.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational error bounds attached to W and B.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the stream and regular-rational rows have been displayed.*
9. *H records componentwise* hsame *transport for the metric source, radius, contraction constant, Picard window, Cauchy-tail budget, StreamName, RegSeqRat, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the metric and radius rows through the Picard window, tail budget, StreamName, RegSeqRat, and Real seal rows.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteRadiusContraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRadiusContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, r, q, W, B, S, Q, R, H, C, P, N$. It has no coordinate for a global fixed point, a complete ambient metric space, a selected host limit, quotient equality of real names, countable choice, or an unbounded Picard iteration.

Theorem 8888.2 (Finite-radius contraction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRadiusContraction}^{\uparrow}$ *carrier*

$$F = (M, r, q, W, B, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRadiusContraction}^{\uparrow}}$ *surface consists exactly of metric-source exposure through M, radius exposure through r, contraction-constant exposure through q, Picard-window exposure through W, Cauchy-tail budget exposure through B, StreamName exposure through S, RegSeqRat readback through Q, Real seal delivery through R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. Its analytic rows are exactly the metric source, radius, contraction constant, Picard window, tail budget, stream window, rational readback, and Real seal.

The local contraction data are read before any Picard consumer can act. The row M supplies only displayed metric distances, r fixes the ball in which the argument is allowed to run, and q supplies the finite rational contraction bound.

The iterative route is then confined to W, B, S, Q, R. The Picard window W lists only finitely many iterates, the budget row B records only the displayed geometric tail estimates, the stream row S names the visible window, the regular rational row Q reads the finite error bounds, and the Real row R seals only that displayed readback.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded global object is a coordinate of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8888.3 (Finite-radius contraction Picard budget)**.** *Every Picard consumer of an accepted* $\mathsf{FiniteRadiusContraction}^{\uparrow}$ *packet must read the finite route*

$$M, r, q, W, B, S, Q, R, H, C, P, N$$

before using a contraction estimate. The route supplies a local radius, a displayed contraction constant, a finite Picard window, and a Cauchy-tail budget for the visible iterates; it supplies no global fixed point, no ambient completeness theorem, no selected limit outside the window, no quotient equality of real names, and no countable-choice schedule.

Proof. Begin with the NameCert obligations of Definition 8888.2. They place the metric source, radius, contraction constant, window, and tail-budget rows before any Real-facing read.

Read the Picard route inside the fixed radius. The displayed constant q is consumed only together with the ball row r and the finite window row W, so a consumer cannot extend the estimate to an unbounded iteration or to a different metric source.

The tail estimate remains finite. The budget row B records the geometric tail data visible to the packet, while S, Q, R turn that window into StreamName, RegSeqRat, and Real-facing rows without selecting a host limit.

Transport, replay, provenance, and local naming preserve this same route. Thus the packet exports exactly a local Picard budget, and any global fixed-point conclusion must be supplied by a separate downstream certificate. □

Falsifiable prediction: Every accepted $\mathsf{FiniteRadiusContraction}^{\uparrow}$ Picard read exposes the metric source, radius row, contraction constant, finite Picard window, Cauchy-tail budget, StreamName row, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert row before using the contraction estimate.

Independence witness: metric, streamname, regseqrat, real: $\mathsf{FiniteRadiusContraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a local radius, contraction constant, Picard window, Cauchy-tail budget, stream readback, regular rational readback, Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRadiusContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRadiusContraction}^{\uparrow}$ packet is generated from a Metric source row, local radius row, contraction-constant row, finite Picard window, Cauchy-tail budget, StreamName row, RegSeqRat readback, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRadiusContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8888.1, Definition 8888.2, and Definition 8888.3 seed the finite local contraction-budget route for Picard-window and Cauchy-tail consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRadiusContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global fixed-point theorem, ambient metric completeness, selected host limits, quotient equality of real names, countable choice, unbounded Picard iteration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, radius exactness, contraction-constant exactness, Picard-window exposure, Cauchy-tail budget exactness, StreamName and RegSeqRat readback, Real seal delivery, transport, replay, provenance, and local naming for BEDC.Derived.FiniteRadiusContractionUp.

Chapter 8889

UniformRegularCompletionScale NameCert Construction

Following Bourbaki's uniform-space completion interface and Bishop-style regular Cauchy completion, the $\mathsf{UniformRegularCompletionScale}^{\uparrow}$ packet records a finite tolerance-scale route from uniform entourages to regular rational names. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the scale used by completion-facing consumers without quotienting Cauchy filters, selecting representatives, importing ambient completeness, or deciding equality of completed points.

Definition 8889.1 (Uniform regular completion scale)**.** *A* $\mathsf{UniformRegularCompletionScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, D, W, R, T, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite uniform-entourage request ledger, indexed by the precision cells actually inspected by a consumer.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii, slack conversions, and monotone refinement data for the inspected cells.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *pointwise certification row, exposing finite observations at the requested tolerance scale.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-stability row, reading the same observations as a regular rational Cauchy name.*
5. *T is the scale-coherence ledger, requiring finer dyadic requests to replay through compatible stream windows and regular rational tails.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *sealing row reached only after D, W, R, T have been displayed.*
7. *H records componentwise* hsame *transport for the uniform request ledger, dyadic tolerance row, stream windows, regular rational tail, scale-coherence ledger, terminal seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the ordered route E, D, W, R, T, S.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformRegularCompletionScale}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformRegularCompletionScale}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion, selected Cauchy representative, ambient uniform space equality, countable choice, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8889.2 (Uniform regular completion scale NameCert obligations)**.** *For every accepted* $\mathsf{UniformRegularCompletionScale}^{\uparrow}$ *carrier*

$$U = (E, D, W, R, T, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformRegularCompletionScale}^{\uparrow}}$ *obligations are exactly carrier admission for the displayed ten-row packet, dyadic tolerance-scale admission through D, pointwise stream certification through W, regular rational tail stability through R, coherence of finer requests through T, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through S, and transport, replay, provenance, and local naming through H, C, P, N. These obligations export a uniform regular completion scale only as finite BEDC data and do not assert quotient equality of completed points, a selected representative, countable choice, trichotomy, or ambient completeness.*

Proof. Carrier admission is projection from Definition 8889.1. The uniform request ledger, dyadic tolerance row, pointwise stream windows, regular rational tail, scale-coherence ledger, terminal seal, transport, replay, provenance, and local name are explicit coordinates of U.

The tolerance route is dyadic. The row D supplies the inspected radii and slack conversions, while W reads only the finite stream observations licensed by those cells. Thus pointwise certification is a $\mathsf{StreamName}^{\uparrow}$ read attached to a $\mathsf{DyadicRatCore}^{\uparrow}$ scale, not a selected infinite representative.

Tail stability is then read through R and T. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational Cauchy handoff, and the scale-coherence ledger forces finer requests to replay through compatible windows and tails before any terminal seal is available.

Finally S exposes the $\mathsf{Real}^{\uparrow}$ boundary only after the ordered route E, D, W, R, T has been displayed. The structural rows H, C, P, N transport and replay that same route, so the NameCert surface has no path to quotient completion, selected limits, countable choice, trichotomy, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformRegularCompletionScale}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformRegularCompletionScale}^{\uparrow}$ packet is generated from a finite uniform-entourage request ledger, a DyadicRatCore tolerance row, StreamName pointwise certification, RegSeqRat tail stability, a scale-coherence ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformRegularCompletionScale}^{\uparrow}}$, seed)

Scope closed: Definition 8889.1 and Definition 8889.2 seed the finite uniform regular completion scale over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformRegularCompletionScale}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected Cauchy representatives, host uniform-space equality, countable choice, trichotomy, Lean verification, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked dyadic tolerance-scale admission, StreamName pointwise certification, RegSeqRat tail stability, scale-coherence replay, RealUp sealing, transport stability, and non-escape for BEDC.Derived.UniformRegularCom

Chapter 8890

ZeckendorfPrimeEdge NameCert Construction

Following Zeckendorf's finite Fibonacci-word normal form and the standard irreducible-edge language of finite directed combinatorics, the $\mathsf{ZeckendorfPrimeEdge}^{\uparrow}$ packet records an atomic recursive prime edge as NameCert data. It sits beside the normal-window surface of chapter 278, the carry route of chapter 558, the carry-classifier boundary of chapter 567, the phase-shell window of chapter 8811, and the classifier-channel ledger of chapter 6765. The packet turns the atomic recursive prime-edge vision into a finite carrier over displayed operations, not into a theorem about ordinary integer primes.

Definition 8890.1 (Zeckendorf prime-edge carrier)**.** *A* $\mathsf{ZeckendorfPrimeEdge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (B_0, B_1, O, \alpha, E, N_D, I, Q, L, H, C, R, N)$$

with the following displayed rows:

1. B_0 *is the source Zeckendorf window row, read through the finite no-adjacent-*11 *classifier.*
2. B_1 *is the target Zeckendorf window row, read through the same finite classifier after the displayed operation fires.*
3. O *is the recursive-operation carrier row: it exposes an operation identity, composition, equivalence, and action on displayed window records.*
4. α *is the atomic operation row acting from* B_0 *toward* B_1*.*
5. E *is the edge row witnessing that* $\alpha(B_0)$ *classifies as* B_1*.*
6. N_D *is the nondegeneracy row, keeping* B_0 *and* B_1 *classifier-distinct before the edge is consumed.*
7. I *is the irreducibility row: every displayed factorization of* α *has a unit factor.*
8. Q *is the prime-divisibility row for the chosen operation divisibility classifier.*
9. L *is the finite ledger exposing source, target, operation, edge, nondegeneracy, irreducibility, and primeness rows.*
10. H *records componentwise* hsame *transport for all displayed rows.*
11. C *records displayed* Cont *replay from source window through operation action, edge readback, and prime-edge ledger.*

12. *R is the* Pkg *provenance over* ZeckendorfPrimeEdge$^{\uparrow}$, ZeckendorfNormal$^{\uparrow}$, ZCarry$^{\uparrow}$, ZeckendorfCarryClassifier$^{\uparrow}$, ZeckendorfPhaseShell$^{\uparrow}$, ClassifierChannelShiftLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{ZeckendorfPrimeEdge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for ordinary integer primality, hidden factorization search, quotient equality of histories, selected normalizer, or a global rewrite system.

Lean4 ✓ `BEDC.Derived.ZeckendorfPrimeEdgeUp.TasteGate.ZeckendorfPrimeEdgeTasteGate single carrier alignment`

Theorem 8890.2 (Zeckendorf prime-edge NameCert obligations)**.** *For every accepted* ZeckendorfPrimeEdge$^{\uparrow}$ *carrier*

$$P = (B_0, B_1, O, \alpha, E, N_D, I, Q, L, H, C, R, N),$$

the local NameCert$_{\text{ZeckendorfPrimeEdge}^{\uparrow}}$ *obligations are exactly carrier admission for the displayed thirteen-row packet, source and target Zeckendorf-window admission, recursive-operation admission, edge exactness, nondegeneracy, irreducibility, operation-prime divisibility, ledger exactness, componentwise transport, continuation replay, provenance, and local naming. These rows realize the atomic recursive prime-edge carrier as finite NameCert data and do not assert ordinary integer primality, quotient history equality, a selected normal form representative, or a global confluence theorem.*

Proof. Carrier admission is projection from Definition 8890.1. The source window, target window, recursive-operation carrier, atomic operation, edge row, nondegeneracy row, irreducibility row, prime-divisibility row, ledger, transport, replay, provenance, and local name are all explicit coordinates of P.

The source and target obligations read the Zeckendorf siblings. The no-adjacent-11 classifier and carry-classifier boundary supply the finite window discipline, while the phase-shell and channel-shift ledgers explain which visible window and classifier-channel rows are being consumed.

Recursive-operation admission is row-local. The carrier reads identity, composition, operation equivalence, and action from O, then reads α as a displayed operation rather than as a hidden evaluator.

Edge exactness, nondegeneracy, irreducibility, and primeness are then read from E, N_D, I, Q. The irreducibility row handles factorization by requiring a unit side in every displayed decomposition of α, and the prime row is relative to operation divisibility, not ordinary integer primality.

Finally L, H, C, R, N expose the finite ledger, transport the same rows, replay the same route, record provenance, and name the packet. Since the carrier has no quotient-history, selected-normalizer, hidden search, or global confluence coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for NameCert$_{\text{ZeckendorfPrimeEdge}^{\uparrow}}$.

Constructive story: A ZeckendorfPrimeEdge$^{\uparrow}$ packet is generated from source and target Zeckendorf window rows, a recursive-operation carrier, an atomic operation, edge exactness, nondegeneracy, irreducibility, operation-prime divisibility, a finite ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{ZeckendorfPrimeEdge}^{\uparrow}}$, seed)

Scope closed: Definition 8890.1 and Definition 8890.2 seed the finite atomic recursive prime-edge surface over Zeckendorf normal windows, carry/classifier rows, phase-shell rows, classifier-channel ledger rows, recursive operations, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{ZeckendorfPrimeEdge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ordinary integer primality, quotient history equality, selected normal representatives, global carry confluence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source/target window admission, recursive-operation admission, edge exactness, nondegeneracy, irreducibility, operation-prime divisibility, ledger exactness, transport stability, and non-escape for BEDC.Derived.ZeckendorfPrimeEdgeUp.

Chapter 8891

A Concrete Naming Certificate for $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ packet records the finite sign-split data used by constructive bisection before any limiting root object is named. It refines the sign-change bracket surface of chapter 3359 and the contraction-facing interval route of chapter 5867 into a single NameCert carrier for endpoint Real seals, located signs, dyadic midpoint budgets, retained subintervals, StreamName windows, RegSeqRat readback, componentwise transport, displayed replay, provenance, and local naming. The packet is a finite handoff object; it is not classical intermediate value, real trichotomy, quotient equality, countable choice, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8891.1 (Finite interval sign-split carrier)**.** *A* $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (a, b, L, U, D, M, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *a and b are endpoint* $\mathsf{Real}^{\uparrow}$ *seal rows read only through finite regular-rational windows.*
2. *L and U are located sign witnesses at the endpoint windows, with opposed sign evidence displayed at the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *interval ledger storing endpoint dyadics, width comparisons, tolerance radii, and midpoint requests.*
4. *M is the midpoint/budget row, recording the finite bisection query and the dyadic budget consumed by the inspected midpoint.*
5. *T is the retained subinterval row, naming the left or right half whose endpoint signs still form an opposed located pair.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the inspected endpoint and midpoint data.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback assembled from the retained subinterval windows and the same dyadic ledger.*
8. *E is the terminal pair of* $\mathsf{Real}^{\uparrow}$ *seals reached only after $a, b, L, U, D, M, T, W, R$ have been displayed.*
9. *H records componentwise* hsame *transport for endpoint, sign, dyadic, midpoint, retained-subinterval, window, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the sign-split request through* $a, b, L, U, D, M, T, W, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$, $\mathsf{RealBolzano}^{\uparrow}$, $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalSignSplit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for real trichotomy, a selected zero, quotient real equality, countable choice, an infinite bisection branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8891.2 (Finite interval sign-split NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ *carrier*

$$S = (a, b, L, U, D, M, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalSignSplit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for endpoint Real seals, located sign witnesses, dyadic interval data, midpoint budget, retained subinterval, StreamName windows, RegSeqRat readback, terminal Real seals, transport, replay, provenance, and local naming;*
2. *endpoint sign exposure, requiring L, U to be read from a, b as displayed located apartness data before any midpoint is inspected;*
3. *dyadic midpoint exactness, requiring D, M to record the same finite width, midpoint, tolerance, and budget rows used by the split;*
4. *retained-subinterval stability, requiring T to keep one opposed located sign pair and to stay inside the displayed endpoint interval;*
5. *window and readback routing, requiring W, R, E to be assembled from the retained subinterval and dyadic budget rows rather than from a selected root;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N for the same rows;*
7. *refusal of classical intermediate value, real trichotomy, quotient equality, selected roots, countable choice, infinite branches, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No finite sign-split obligation is stored outside $a, b, L, U, D, M, T, W, R, E, H, C, P, N$.

Proof. Project the carrier of Definition 8891.1. The endpoint rows a, b and the located sign witnesses L, U are the only sources of sign data, and both are read at displayed finite precision.

The split itself is dyadic. The ledger D and midpoint row M record the width, midpoint, tolerance, and budget data before the retained subinterval T is admitted.

The completion-facing rows are downstream of the retained subinterval. The window row W and readback row R are assembled from T, D, M, and the terminal Real seals E are read only after that finite route is present.

Finally H, C, P, N transport, replay, source, and name the same packet rows. Since the carrier contains no excluded order, quotient, choice, branch, root-selection, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8891.3 (Finite interval sign-split bisection handoff)**.** *Every bisection-facing consumer of an accepted* $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ *packet must read the finite route*

$$a, b, L, U, D, M \longmapsto T, W, R, E \longmapsto H, C, P, N$$

before it can treat a retained half-interval as sign-change evidence. The handoff supplies endpoint Real seals, located sign witnesses, dyadic midpoint and budget rows, a retained subinterval, finite StreamName windows, RegSeqRat readback, componentwise transport, displayed continuation replay, provenance, and local naming. It supplies no unrestricted intermediate-value theorem, selected root, real trichotomy, quotient real equality, countable choice, infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8891.2. They force endpoint sign exposure through a, b, L, U and dyadic midpoint exactness through D, M before a retained subinterval can be consumed.

Apply the RealBolzano sign-change bisection route of Definition 3359.6. The retained half may be exported only as opposed located sign data inside a displayed bracket, not as a selected root or unrestricted intermediate-value result.

The StreamName and RegSeqRat rows then carry the retained half to completion-facing consumers. This is the same finite kind of Cauchy handoff isolated by Definition 5867.3, with W, R, E downstream of the dyadic and sign rows.

The structural rows H, C, P, N preserve that route for later consumers. Since none of the cited rows contains real trichotomy, quotient real equality, countable choice, an infinite branch, selected roots, or ambient Real completeness, the handoff remains the finite bisection surface stated above. □

Falsifiable prediction: An accepted $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ consumer that reaches a retained subinterval, $\mathsf{RegSeqRat}^{\uparrow}$ readback, or terminal $\mathsf{Real}^{\uparrow}$ seal without reading $a, b, L, U, D, M, T, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: real-bolzano, bishop-interval-contraction, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies exactly the joint packet containing endpoint Real seals, located sign witnesses, dyadic midpoint budgets, a retained subinterval, finite windows, regular-rational readback, terminal Real seals, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalSignSplit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntervalSignSplit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8891.1, Definition 8891.2, and Definition 8891.3 seed the constructive finite sign-split route from endpoint Real seals and located sign witnesses to retained subinterval, RegSeqRat readback, and terminal Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntervalSignSplit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim classical intermediate value, real trichotomy, quotient real equality, selected roots, countable choice, infinite branches, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exposure, dyadic midpoint exactness, retained-subinterval stability, window readback routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalSignSplitUp.

Constructive story: A $\mathsf{FiniteIntervalSignSplit}^{\uparrow}$ packet is generated from endpoint Real seals, located sign witnesses, dyadic midpoint and budget rows, a retained subinterval row, StreamName finite windows, RegSeqRat readback, terminal Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8892

A Concrete Naming Certificate for $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$

Bishop and Bridges constructive analysis treats diagonal arguments over regular real families through explicit finite observations, moduli, and null-tail comparisons rather than selected representatives of completed real names. The $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$ packet records that family-diagonal route as finite NameCert data. It sits over the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the zero-tail comparison service of chapter 380, the terminal real-name seal of chapter 38, and the completion-facing boundary of chapter 360. The packet names a finite diagonal family carrier for Real and Completion consumers; it is not quotient stream equality, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8892.1 (Regular real family diagonal carrier)**.** *A* $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family, listing the source-family coordinates and observation windows inspected by the diagonal request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed family windows rather than from a selected stream representative.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the diagonal precision request, local dyadic slack, and comparison budget used by the readback.*
4. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row, displaying that the diagonal difference route is eventually absorbed by the named tolerance windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after S, R, D, Z have been displayed.*
6. *H records componentwise* hsame *transport for the stream-family, regular readback, dyadic tolerance, zero-tail, Real seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a diagonal-family request through S, R, D, Z, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularRealFamilyDiagonal}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed regular-real family diagonal history. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8892.2 (Regular real family diagonal NameCert obligations)**.** *For every accepted* RegularRealFamilyDiago *carrier $F = (S, R, D, Z, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRealFamilyDiagonal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed stream-family windows, regular rational readback, dyadic tolerance ledger, zero-tail comparison, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *window-family exactness, requiring S to expose only the finite source-family observations inspected by the diagonal request;*
3. *regular-readback handoff, requiring R to be read from the displayed windows and not from an unrecorded representative of the source family;*
4. *dyadic tolerance exactness, requiring D to bind the diagonal precision request to the readback and zero-tail rows;*
5. *null-tail handoff, requiring Z to be displayed before the Real seal E is consumed by any equality-facing or completion-facing reader;*
6. *ledger exhaustion, requiring H, C, P, N to transport, replay, source, and name no quotient stream, selected limit, countable-choice schedule, trichotomy branch, host equality, ambient Real-completeness theorem, Lean verification, or checked bridge.*

No regular-real family diagonal obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 8892.1. The accepted packet exposes exactly the finite stream-family windows, regular readback, dyadic tolerance row, zero-tail row, terminal seal, and structural rows.

Window-family exactness is read from S. A diagonal request names which family coordinates and finite windows are inspected before any regular readback can be consumed.

The regular-readback and dyadic clauses are supplied by R and D. The readback row receives the displayed family windows, while the dyadic row records the precision and slack budget used by the diagonal comparison.

The null-tail handoff is the row Z. It must be displayed before the terminal Real seal E, so an equality-facing or completion-facing consumer receives the zero-tail evidence as part of the same finite route.

Finally H, C, P, N preserve the route by componentwise transport, replay, provenance, and local naming. Since the carrier has no coordinate for the excluded quotient, selector, choice, trichotomy, host-equality, or completeness objects, the listed rows exhaust the local NameCert surface. □

Theorem 8892.3 (Regular real family diagonal seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RealCauchyCompletion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$ *carrier must read the displayed route*

$$S, R, D, Z, E, H, C, P, N$$

before using the diagonal family as a real-name or completion-name input. The consumer therefore receives the finite StreamName family, RegSeqRat readback, DyadicRatCore tolerance, NullSequence zero-tail comparison, Real seal, transport, replay, provenance, and local naming rows as

prior data. It cannot replace that route by quotient streams, selected limits, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8892.1. The only Real-facing endpoint is E, and the carrier places E after the finite family windows, regular readback, dyadic tolerance ledger, and null-tail row.

Apply Definition 8892.2. The window-family, readback, dyadic, and null-tail obligation rows force every consumer to read S, R, D, Z before the terminal Real seal can be used.

The completion-facing handoff is downstream of the same seal. A $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer may read the diagonal family only as the already named route $S, R, D, Z, E, H, C, P, N$; it receives no independent completion coordinate from the carrier.

The structural rows preserve this route and add no quotient stream, selected limit, countable-choice schedule, trichotomy branch, host equality, or ambient completeness theorem. Hence the displayed route is exhaustive for both Real-facing and completion-facing consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealFamilyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealFamilyDiagonal}^{\uparrow}$ packet is generated from finite StreamName family windows, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a NullSequence zero-tail comparison, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealFamilyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8892.1, Definition 8892.2, and Definition 8892.3 seed the finite diagonal-family route through StreamName, RegSeqRat, DyadicRatCore, NullSequence, Real, and RealCauchyCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealFamilyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-family exactness, regular-readback handoff, dyadic tolerance exactness, null-tail handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularRealFamilyDiagonalUp.

Chapter 8893

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}$ packet records the finite located-real apartness schedule that turns a displayed separation request into dyadic brackets and a readable modulus row. It sits over the located-real source of chapter 1933, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the located-apartness comparison row of chapter 6450. The packet names only finite bracket and modulus data; it does not assert trichotomy, quotient real equality, selected real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8893.1 (Bishop located-real apartness modulus carrier)**.** *A* $\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, E, R, D, B, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *source row naming the located real datum whose apartness request is being read.*
2. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the located source and its finite readbacks have been displayed.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows inspected by the apartness request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying dyadic radii, half-radius comparisons, endpoint brackets, and bracket-refinement requests.*
5. *B is the located-bracket transport row, requiring each apartness request to pass through the same displayed brackets before a modulus can be consumed.*
6. *M is the apartness-modulus extraction row: it records the finite precision schedule that turns a positive dyadic separation witness into the requested apartness margin.*
7. *H records componentwise* hsame *transport for the located source, Real seal, regular-rational readback, dyadic tolerance ledger, bracket transport, modulus extraction, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay along the same located-real, regular-rational, dyadic, bracket, and modulus route.*
9. *P is the* Pkg *provenance over* BishopLocatedRealApartnessModulus$^{\uparrow}$, LocatedReal$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, LocatedRealApartness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopLocatedRealApartnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, E, R, D, B, M, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected real representatives, hidden limit choice, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 8893.2 (Bishop located-real apartness modulus obligations)**.** *For every accepted* BishopLocatedRealApart *carrier*

$$A = (L, E, R, D, B, M, H, C, P, N),$$

the local NameCert$_{\text{BishopLocatedRealApartnessModulus}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier admission through the displayed rows $L, E, R, D, B, M, H, C, P, N$;*
2. *classifier stability, requiring comparison only through the same located-real, real-seal, regular-rational, dyadic, bracket, modulus, and structural rows;*
3. *located bracket transport through B, so each apartness read consumes the same finite bracket route before reaching* LocatedRealApartness$^{\uparrow}$ *consumers;*
4. *apartness modulus extraction through M, turning a displayed positive dyadic separation witness into the finite precision schedule requested by the consumer;*
5. *ledger exactness through D, H, C, P, N, preserving the dyadic tolerance row, componentwise transport, replay, provenance, and local naming without adding a host real equality service.*

No obligation supplies trichotomy, quotient real equality, selected representatives, hidden limit choice, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopLocatedRealApartnessModulusUp`

Proof. Project the carrier of Definition 8893.1. Its visible rows are exactly the located source, Real seal, regular-rational readback, dyadic tolerance ledger, bracket transport, modulus extraction, and structural naming rows.

The classifier-stability clause is immediate from that projection. Since the carrier has no coordinate outside $L, E, R, D, B, M, H, C, P, N$, an accepted comparison can inspect only those rows.

The located-bracket and apartness-modulus clauses are the two value-facing obligations. The row B forces each apartness request to pass through displayed brackets, and the row M extracts the finite precision schedule from the positive dyadic separation witness recorded through D.

Finally H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Because none of the excluded principles is a row of the packet, the listed five rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted BishopLocatedRealApartnessModulus$^{\uparrow}$ apartness read exposes the located-real source, Real seal, RegSeqRat readback, DyadicRatCore tolerance ledger, bracket transport, modulus extraction, hsame transport, Cont replay, provenance, and local NameCert row before using a positive separation margin.

Independence witness: located-real, real, regseqrat, dyadicratcore, located-real-apartness: BishopLocatedRealAp is not bijective to any listed sibling because it jointly binds a located source, real seal, regular-rational readback, dyadic tolerance ledger, bracket transport, modulus extraction, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}$ packet is generated from a LocatedReal source row, RealUp seal row, RegSeqRat readback, DyadicRatCore tolerance ledger, located-bracket transport row, apartness-modulus extraction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8893.1 and Definition 8893.2 seed the finite located-real apartness-modulus route through brackets, dyadic tolerances, regular-rational readback, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealApartnessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected real representatives, hidden limit choice, countable choice, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, located bracket transport, apartness modulus extraction, ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealApartnes

Chapter 8894

RieszInterpolationRealUp NameCert Construction

Following Riesz's interpolation property for ordered groups and vector lattices, the $\mathsf{RieszInterpolationReal}^{\uparrow}$ packet records a finite interpolation witness for two displayed lower real names and two displayed upper real names inside the constructive Real algebraic-order surface. It sits between the terminal real-name seal of chapter 38, the Real algebraic-order boundary of chapter 766, and the dyadic tolerance core of chapter 291. The packet names the dyadic comparison route that witnesses one interpolant w with $a_0, a_1 \leq w \leq b_0, b_1$; it does not assert trichotomy, quotient real equality, selected representatives, order completion, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8894.1 (RieszInterpolationReal carrier)**.** *A* $\mathsf{RieszInterpolationReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, A, D, L, U, W, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the four endpoint real names and the candidate interpolant real name.*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation-and-order row exposing the constructive nonnegative-difference comparisons used by the packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger recording the finite dyadic radii, slack rows, and lower-upper comparison budget.*
4. *L is the lower-pair row, displaying the two lower endpoints a_0, a_1 and their dyadic comparison requests against the upper endpoints.*
5. *U is the upper-pair row, displaying the two upper endpoints b_0, b_1 and the finite evidence that each lower endpoint is below each upper endpoint through A and D.*
6. *W is the interpolation-witness row, naming a displayed real name w together with the four comparison reads $a_0 \leq w$, $a_1 \leq w$, $w \leq b_0$, and $w \leq b_1$ through the same RealAlgOrder and dyadic ledger rows.*
7. *H records componentwise* hsame *transport for the Real seal, RealAlgOrder row, dyadic ledger, lower pair, upper pair, witness row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-order interpolation request through R, A, D, L, U, W, H back to the same finite comparison surface.*

9. *P is the* Pkg *provenance over* $\mathsf{RieszInterpolationReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszInterpolationReal}^{\uparrow}}$ *row binding the interpolation packet.*

The classifier compares accepted packets only through $R, A, D, L, U, W, H, C, P, N$. It has no coordinate for host real order, trichotomy, quotient representatives, a selected order-completion element, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8894.2 (RieszInterpolationReal dyadic witness)**.** *For every accepted* $\mathsf{RieszInterpolationReal}^{\uparrow}$ *carrier*

$$I = (R, A, D, L, U, W, H, C, P, N),$$

every real-order interpolation consumer must read the finite route

$$R, A, D, L, U \longrightarrow W \longrightarrow H, C, P, N$$

before using the interpolant. The route supplies the four endpoint real seals, the RealAlgOrder comparison row, the shared dyadic comparison ledger, the lower-pair row, the upper-pair row, and the displayed witness row; it supplies no trichotomy branch, quotient equality of real names, selected representative, order-completion principle, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier of Definition 8894.1. Its analytic rows are exactly the Real seal row R, the RealAlgOrder row A, the dyadic comparison ledger D, the lower-pair row L, the upper-pair row U, and the interpolation witness row W.

The comparison premises are finite. The lower and upper rows are accepted only through the displayed RealAlgOrder comparisons and the dyadic ledger, so the request cannot switch to a host order relation or to a quotient equality of real names.

The witness row is read after those premises. Its four comparisons reuse the same A and D rows, and the structural rows H, C, P, N transport, replay, source, and name this same packet. Since none of the excluded order-completion or choice principles appears as a carrier coordinate, the listed route exhausts the consumer-facing witness surface. □

Theorem 8894.3 (RieszInterpolationReal ledger stability)**.** *Let $I = (R, A, D, L, U, W, H, C, P, N)$ be an accepted* $\mathsf{RieszInterpolationReal}^{\uparrow}$ *carrier. If a downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *consumer refines the dyadic tolerance rows inside the same ledger D while preserving the displayed lower-pair, upper-pair, and witness comparisons, then the accepted interpolation read still factors through*

$$R, A, D, L, U, W, H, C, P, N.$$

The refinement may narrow the comparison radii, but it cannot add a new host real-order coordinate, a trichotomy selector, quotient real equality, countable choice, a selected completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the dyadic witness route of Definition 8894.2. It identifies the only rows available to any real-order interpolation consumer.

A refinement inside D changes only the displayed dyadic radius and slack entries carried by the same ledger. The lower-pair row L, upper-pair row U, and witness row W remain tied to the same Real seal R and RealAlgOrder row A.

Transport and replay are already recorded by H and C, while P and N keep the provenance and local naming fixed. Therefore the refined read is still the same finite NameCert route, with no additional analytic or choice-dependent coordinate. □

Falsifiable prediction: Every accepted $\mathsf{RieszInterpolationReal}^{\uparrow}$ witness read exposes the Real seal row, RealAlgOrder comparison row, dyadic comparison ledger, lower-pair row, upper-pair row, witness row, componentwise transport, displayed replay, provenance, and local NameCert row before using the interpolant.

Independence witness: real, real-alg-order, dyadicratcore: $\mathsf{RieszInterpolationReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds four real endpoints, one witness real name, a RealAlgOrder comparison surface, a dyadic comparison ledger, lower and upper pair rows, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RieszInterpolationReal}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszInterpolationReal}^{\uparrow}$ packet is generated from Real endpoint and witness seals, a RealAlgOrder comparison row, a DyadicRatCore comparison ledger, displayed lower and upper pairs, an interpolation witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszInterpolationReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8894.1, Definition 8894.2, and Definition 8894.3 seed the finite real-order interpolation witness route over displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszInterpolationReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, order completion, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Real seal exposure, RealAlgOrder comparison exactness, dyadic ledger stability, lower-pair and upper-pair display, witness display, transport, replay, provenance, and local naming for BEDC.Derived.RieszInterpolationRealUp.

Chapter 8895

A Concrete Naming Certificate for DyadicCoverModulus$^{\uparrow}$

Following Bishop and Bridges finite-cover arguments, the DyadicCoverModulus$^{\uparrow}$ packet records the mesh-control data used when a regular-Cauchy limit route must pass through finitely many dyadic cells. It concretizes the regular-Cauchy vision of section 9238.1 by naming the finite cover modulus that sits between the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, the finite epsilon-net surface of chapter 1767, and the dyadic interval-net cells of chapter 5966. The packet exports only finite mesh coverage and modulus bookkeeping; it does not assert compactness of a host space, selected limits, quotient equality of real names, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8895.1 (Dyadic cover modulus carrier)**.** *A* DyadicCoverModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, I, E, R, L, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, carrying dyadic radii, mesh widths, half-radius comparisons, and finite refinement requests.*
2. *I is the* DyadicIntervalNet$^{\uparrow}$ *cell row naming the finite dyadic interval cells inspected by the cover request.*
3. *E is the* FiniteEpsilonNet$^{\uparrow}$ *row that records the finite net points or cells at the requested tolerance.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row whose regular-Cauchy windows are controlled by the displayed mesh.*
5. *L is the terminal* Real$^{\uparrow}$ *seal row, available only after the dyadic mesh and regular-rational readback have been exposed.*
6. *W is the cover-modulus row: for each requested dyadic tolerance it records the finite mesh level and cover cells sufficient for the regular-Cauchy read.*
7. *H records componentwise* hsame *transport for $D, I, E, R, L, W, C, P, N$.*
8. *C records displayed* Cont *replay of the same dyadic-cover route.*
9. *P is the* Pkg *provenance over* DyadicCoverModulus$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* DyadicIntervalNet$^{\uparrow}$*,* FiniteEpsilonNet$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCoverModulus}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, I, E, R, L, W, H, C, P, N$. It has no coordinate for host compactness, selected limiting covers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8895.2 (Dyadic cover modulus obligations)**.** *For every accepted* $\mathsf{DyadicCoverModulus}^{\uparrow}$ *carrier*

$$K = (D, I, E, R, L, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCoverModulus}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission through D, dyadic interval-cell exposure through I, finite epsilon-net exposure through E, regular-rational readback through R, Real seal delivery through L, finite cover-modulus exactness through W, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8895.1. Its rows are exactly the dyadic tolerance ledger, interval-net cells, finite epsilon-net surface, regular-rational readback, Real seal, cover-modulus row, and structural naming rows.

The dyadic mesh is read before any regular-Cauchy consumer reaches the Real seal. The rows D, I, E expose the requested tolerance, the finite interval cells, and the finite net data.

The cover modulus is then the row W. It binds each tolerance request to a mesh level and a finite list of cover cells before R and L can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded host objects is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8895.3 (Regular-Cauchy finite mesh control)**.** *Every regular-Cauchy limit consumer reading an accepted* $\mathsf{DyadicCoverModulus}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{DyadicIntervalNet}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the cover-modulus, transport, replay, provenance, and local naming rows. The route supplies finite mesh control for the regular-Cauchy limit read and supplies no host compactness theorem, selected limit, quotient equality of real names, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the NameCert obligations of Definition 8895.2. They force the dyadic tolerance ledger, interval-net row, finite epsilon-net row, and cover-modulus row to be exposed before the regular-rational readback or terminal Real seal is available.

The regular-Cauchy consumer therefore reads mesh control as finite data. A requested tolerance first opens D, then chooses the displayed cell level in I and finite cover in E, and only then reaches R and L.

The structural rows H, C, P, N preserve that route without adding a compactness theorem or a selected limiting object. Thus the packet realizes finite mesh control for the regular-Cauchy limit route and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCoverModulus}^{\uparrow}$ regular-Cauchy read exposes the DyadicRatCore tolerance ledger, DyadicIntervalNet cells, FiniteEpsilonNet row, cover-modulus row, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert row before using finite mesh control.

Independence witness: dyadicratcore, dyadic-interval-net, finite-epsilon-net, regseqrat, real: $\mathsf{DyadicCoverModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds dyadic tolerances, interval cells, finite net data, a cover-modulus schedule, regular-rational readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCoverModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCoverModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, DyadicIntervalNet cell row, FiniteEpsilonNet row, RegSeqRat readback, RealUp seal, cover-modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCoverModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8895.1, Definition 8895.2, and Definition 8895.3 seed the finite dyadic mesh-control route for regular-Cauchy limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCoverModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected limiting covers, selected real limits, quotient equality of real names, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance admission, interval-net exposure, finite epsilon-net exposure, cover-modulus exactness, RegSeqRat readback, Real seal delivery, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCoverModulusUp.

Chapter 8896

Diffeological Space NameCert Construction

Following Souriau and Iglesias-Zemmour, the $\mathsf{DiffeologicalSpace}^{\uparrow}$ packet reconstructs a diffeological-space interface as finite plot-observation data. The carrier records point rows, plot rows, domain-window rows, locality and gluing checks, pullback stability, transport, replay, provenance, and local naming; it gives topology and manifold consumers a visible smooth-space neighbour without quotienting all plots into a global smooth object.

The packet sits between the open-set reading of chapter 88, the chart-facing manifold surface of chapter 96, the stream-name interface of chapter 126, and the BHist and $\mathsf{NameCert}$ surfaces used throughout the concrete-instance ledger. It does not claim quotient smooth spaces, selected atlases, global colimits, or full manifold reconstruction.

Definition 8896.1 (Diffeological-space carrier)**.** *A* $\mathsf{DiffeologicalSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, P, U, L, G, S, T, R, Q, N)$$

with the following displayed rows:

1. *X is the point row, listing the observed points whose plot memberships are named by the packet.*
2. *P is the finite plot row, storing selected parameterized observations into X.*
3. *U is the domain-window row, recording the open-domain fragments on which the displayed plots are inspected.*
4. *L is the locality row, stating that plot admission is checked on the named domain windows.*
5. *G is the gluing row, recording compatible finite local plots before a glued plot is read.*
6. *S is the pullback-stability row, showing that a displayed smooth reparameterization keeps the read inside the same plot surface.*
7. *T records componentwise transports for point, plot, domain, locality, gluing, and pullback rows.*
8. *R is the replay ledger that exposes only X, P, U, L, G, S, T to downstream consumers.*
9. *Q is the* Pkg *provenance over* $\mathsf{DiffeologicalSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiffeologicalSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient smooth-space construction, global colimits, selected atlases, full manifold reconstruction, or host smoothness semantics.

Theorem 8896.2 (Diffeological-space NameCert obligations)**.** *For an accepted* $\mathsf{DiffeologicalSpace}^{\uparrow}$ *carrier*

$$D = (X, P, U, L, G, S, T, R, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiffeologicalSpace}^{\uparrow}}$ *surface consists exactly of point-row admission, finite plot-row admission, domain-window visibility, locality checking, gluing checking, pullback stability, componentwise transport, replay exactness, provenance, and local naming. Every topology, manifold, or stream-name consumer of the packet must read the same rows before using a plot observation.*

Proof. Project the carrier in Definition 8896.1. The rows X, P, U give the observed points, the selected plots, and the domain windows on which those plots are inspected.

Next read the diffeological closure data. The locality row L keeps plot admission local to the displayed windows, the gluing row G records finite compatible local plots, and the pullback row S keeps smooth reparameterization inside the observed plot surface.

Transport and replay are then structural. The row T preserves each displayed coordinate componentwise, while R exposes only the finite plot-observation inventory to downstream reads.

Finally, provenance and naming are Q and N. Since no quotient smooth space, selected atlas, global colimit, full manifold reconstruction, or host smoothness object appears as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8896.3 (Diffeological-space plot-observation nonescape)**.** *Every public read exported from an accepted* $\mathsf{DiffeologicalSpace}^{\uparrow}$ *packet factors through* $X, P, U, L, G, S, T, R, Q, N$*. The read may be consumed by* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *only as finite plot-observation data; it is not a quotient smooth space, a selected atlas, a global colimit, or a proof of full manifold reconstruction.*

Proof. Use the NameCert-obligation inventory of Definition 8896.2. It requires every accepted read to expose the point, plot, domain-window, locality, gluing, pullback, transport, replay, provenance, and naming rows.

The topology, manifold, and stream-name consumers therefore receive a finite observation interface. The packet can say which displayed plot rows are available and how they survive locality, gluing, and pullback checks.

No refused global object is present in the carrier. Hence quotient smooth spaces, selected atlases, global colimits, host smoothness semantics, and full manifold reconstruction cannot be extracted from the $\mathsf{DiffeologicalSpace}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiffeologicalSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiffeologicalSpace}^{\uparrow}$ packet is generated from finite point, plot, domain-window, locality, gluing, pullback, transport, replay, provenance, and naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiffeologicalSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8896.1, Definition 8896.2, and Definition 8896.3 bind the finite plot-observation seed surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiffeologicalSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not construct quotient smooth spaces, selected atlases, global colimits, host smoothness semantics, or full manifold reconstruction.

Upgrade path: Obligation closure requires checked carrier admission, locality and gluing rows, pullback stability, replay exactness, provenance, and local naming for the same finite packet.

Chapter 8897

A Concrete Naming Certificate for LocatedDedekindCutCauchyBracket$^{\uparrow}$

Following Bishop and Bridges, the LocatedDedekindCutCauchyBracket$^{\uparrow}$ packet records the finite bridge by which a displayed regular Cauchy real name supplies lower and upper located Dedekind-cut brackets. It sits after the dyadic arithmetic source of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the cut surfaces of chapter 7256 and chapter 8086. The packet names a bracket display for downstream cut consumers; it is not a construction of Real$^{\uparrow}$ from Dedekind cuts, quotient equality of real names, trichotomy, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 8897.1 (Located Dedekind cut Cauchy bracket carrier)**.** *A* LocatedDedekindCutCauchyBracket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, E, L, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *window and tolerance ledger used to compare rational probes with the regular Cauchy source.*
2. *W is the* StreamName$^{\uparrow}$ *finite observation row that lists the inspected Cauchy windows.*
3. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback row accepted from the same windows and dyadic ledger.*
4. *E is the terminal* Real$^{\uparrow}$ *seal row for the named regular Cauchy real.*
5. *L is the lower located cut bracket row, consumed through* LocatedLowerCut$^{\uparrow}$ *and the lower side of* LocatedDedekindCut$^{\uparrow}$*.*
6. *U is the upper located cut bracket row, displaying the complementary upper tests attached to the same dyadic windows.*
7. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, Real seal, lower bracket, upper bracket, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cut-bracket request back through D, W, R, E, L, U, H.*
9. *P is the* Pkg *provenance over* LocatedDedekindCutCauchyBracket$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* LocatedDedekindCut$^{\uparrow}$*,* LocatedLowerCut$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, W, R, E, L, U, H, C, P, N$. It has no coordinate for constructing a Dedekind real quotient, identifying $\mathsf{Real}^{\uparrow}$ *with cut equality, choosing representatives, deciding trichotomy, or importing ambient completeness.*

Theorem 8897.2 (Located Dedekind cut Cauchy bracket)**.** *Every accepted regular Cauchy real name displayed by*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

yields a $\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}$ *packet whose lower row is a located lower cut bracket and whose upper row is the matching upper bracket over the same dyadic window data. The packet supplies displayed lower and upper located cut brackets for* $\mathsf{LocatedDedekindCut}^{\uparrow}$ *consumers, while making no claim that* $\mathsf{Real}^{\uparrow}$ *is constructed from Dedekind cuts and no claim of quotient equality between the regular Cauchy name and a Dedekind-cut real.*

Proof. Project the accepted regular Cauchy route. The dyadic ledger D, finite windows W, and regular rational readback R are displayed before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

Build the lower bracket from the same data. A rational probe below the inspected Cauchy window is recorded in L through the $\mathsf{LocatedLowerCut}^{\uparrow}$ surface, and the locatedness row is attached to the same dyadic tolerance ledger rather than to a host order oracle.

Build the upper bracket in parallel. The upper tests in U use the same D, W, R, E route, so lower and upper bracket reads share one regular Cauchy source and one finite window schedule.

Finally attach transport, replay, provenance, and local naming through H, C, P, N. Since no carrier coordinate stores a Dedekind-real construction, quotient equality, trichotomy, countable choice, or ambient completeness, the accepted packet exports only the displayed lower and upper bracket rows. □

Theorem 8897.3 (Located Dedekind cut Cauchy bracket order refinement)**.** *For every accepted* $\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}$ *carrier*

$$B = (D, W, R, E, L, U, H, C, P, N),$$

the lower and upper located cut rows exposed to an order-refinement consumer are read through the same $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus any accepted comparison or refinement of the bracket route must pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic source D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows W, the regular readback R, the lower row L, the upper row U, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}}$ *row. The packet supplies no trichotomy row, no Dedekind-to-Cauchy reconstruction, and no quotient identification of cut equality with* $\mathsf{Real}^{\uparrow}$ *equality.*

Proof. Project the packet rows of Definition 8897.1. The lower bracket row L and upper bracket row U are stored next to the same dyadic ledger D, finite windows W, regular readback R, and real seal E.

Use the main bracket theorem, Definition 8897.2. It identifies both cut-facing rows as downstream of one displayed regular Cauchy route

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

An order-refinement consumer can therefore compare only the lower and upper bracket rows already attached to that route. Transport through H, replay through C, provenance through P,

and local naming through N preserve the same displayed packet and add no classifier coordinate beyond D, W, R, E, L, U.

Since the carrier has no row for trichotomy, no Dedekind-to-Cauchy reconstruction, and no quotient identification between cut equality and $\mathsf{Real}^{\uparrow}$ equality, the refinement remains confined to the displayed lower and upper located cut rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}$ consumer that reads a lower or upper cut bracket without displaying $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-dedekind-cut, located-lower-cut: $\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds one regular Cauchy real route, one lower located cut bracket, one upper bracket, transport, replay, provenance, and local naming in a single finite bridge packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}$ packet is generated from DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat readback, a RealUp seal, lower and upper located cut bracket rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8897.1 and Definition 8897.2 seed the finite bracket route from regular Cauchy real data to located lower and upper cut rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDedekindCutCauchyBracket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind construction of RealUp, quotient equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite window inspection, regular rational readback, Real seal nonescape, lower bracket locatedness, upper bracket locatedness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDedekindCutCauchyBracketUp.

Chapter 8898

A Concrete Naming Certificate for $\mathsf{FiniteDataAssembly}^{\uparrow}$

The $\mathsf{FiniteDataAssembly}^{\uparrow}$ packet realizes the finite data interface spine of chapter 9353 as a single NameCert assembly over the existing Boolean, option, sum, product, and list chapters. It consumes $\mathsf{Bool}^{\uparrow}$ from chapter 32, $\mathsf{Option}^{\uparrow}$ from chapter 33, $\mathsf{Sum}^{\uparrow}$ from chapter 35, $\mathsf{Prod}^{\uparrow}$ from chapter 34, and $\mathsf{List}^{\uparrow}$ from chapter 36. The packet names the finite constructor, eliminator, stability, and ledger rows shared by those interfaces; it is not a new kernel input, a host finite-set universe, quotient equality, or a choice principle.

Definition 8898.1 (Finite data assembly carrier)**.** *A* $\mathsf{FiniteDataAssembly}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, O, S, Pd, L, K, E, T, C, Q, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *row exposing the two endpoint constructors and Boolean case analysis.*
2. *O is the* $\mathsf{Option}^{\uparrow}$ *row exposing absent and present branches.*
3. *S is the* $\mathsf{Sum}^{\uparrow}$ *row exposing left and right tagged injections.*
4. *Pd is the* $\mathsf{Prod}^{\uparrow}$ *row exposing paired component projections.*
5. *L is the* $\mathsf{List}^{\uparrow}$ *row exposing empty and cons constructors together with finite induction.*
6. *K is the shared constructor ledger, recording how the five interfaces build admitted histories from displayed rows.*
7. *E is the shared eliminator ledger, recording the allowed case-analysis, projection, and finite-induction reads.*
8. *T records classifier stability and componentwise* hsame *transport for all five finite-data interfaces.*
9. *C records displayed* Cont *replay from a finite-data request back through B, O, S, Pd, L, K, E, T.*
10. *Q is the* Pkg *provenance over* $\mathsf{FiniteDataAssembly}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDataAssembly}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, S, Pd, L, K, E, T, C, Q, N$. It has no coordinate for a host finite-set universe, unbounded case split, quotient equality, choice-selected branch, or extra kernel primitive.

Theorem 8898.2 (Finite data assembly seed obligation)**.** *Every public finite-data read that uses the Bool, Option, Sum, Prod, and List interface layer factors through an accepted* $\mathsf{FiniteDataAssembly}^{\uparrow}$ *packet. The packet displays the five sibling rows*

$$\mathsf{Bool}^{\uparrow},\ \mathsf{Option}^{\uparrow},\ \mathsf{Sum}^{\uparrow},\ \mathsf{Prod}^{\uparrow},\ \mathsf{List}^{\uparrow}$$

together with shared constructor, eliminator, classifier-stability, transport, replay, provenance, and local naming rows before any finite-data consumer may use the interface layer as a single public surface.

Proof. Project the sibling rows. Boolean reads consume B, option reads consume O, sum reads consume S, product reads consume Pd, and list reads consume L; none of these rows is replaced by host finite data.

Read the shared constructor and eliminator ledgers. The constructor ledger K records how admitted histories are built, while the eliminator ledger E records the allowed case-analysis, projection, and finite-induction paths for the same displayed rows.

Apply classifier stability and transport through T. The five interfaces remain comparable only by their displayed classifiers and componentwise hsame routes.

Finally attach replay, provenance, and local naming through C, Q, N. The resulting assembly is a finite NameCert packet over existing sibling interfaces, so it supplies no host finite-set universe, quotient equality, unbounded branch selector, choice principle, or additional kernel input. □

Falsifiable prediction: An accepted $\mathsf{FiniteDataAssembly}^{\uparrow}$ consumer that uses the finite data layer without displaying Bool, Option, Sum, Prod, List, constructor, eliminator, stability, replay, provenance, and local NameCert rows refutes the assembly surface.

Independence witness: bool, option, sum, prod, list: $\mathsf{FiniteDataAssembly}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds five finite-data interfaces, one constructor ledger, one eliminator ledger, shared stability, transport, replay, provenance, and local naming in a single assembly packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDataAssembly}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDataAssembly}^{\uparrow}$ packet is generated from Bool, Option, Sum, Prod, and List rows, shared constructor and eliminator ledgers, classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDataAssembly}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8898.1 and Definition 8898.2 seed the finite assembly NameCert route for the Bool/Option/Sum/Prod/List interface spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDataAssembly}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host finite-set universe, quotient equality, unbounded branch selection, choice, extra kernel primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling row exposure, constructor ledger exactness, eliminator ledger exactness, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDataAssemblyUp.

Chapter 8899

A Concrete Naming Certificate for $\mathsf{FiniteInterfaceSpine}^{\uparrow}$

The $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ packet realizes the finite data layer of chapter 9353 as one L6 NameCert interface over the existing $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, and $\mathsf{List}^{\uparrow}$ rows. It reconstructs the ordinary finite-data toolkit as displayed BEDC certificate data rather than as kernel input: truth tags, nullable alternatives, disjoint branch tags, ordered pairs, and finite lists are all read through histories, history sameness, continuation replay, package provenance, gap discipline, and local naming. The packet sits after chapter 32, chapter 33, chapter 35, chapter 34, and chapter 36; it exports no host datatype primitive, unrestricted recursion scheme, quotient equality of finite objects, or hidden kernel constructor.

Definition 8899.1 (Finite interface spine carrier)**.** *A* $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, O, S, P, L, H, C, Q, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *endpoint row, naming the two visible truth tags.*
2. *O is the* $\mathsf{Option}^{\uparrow}$ *nullable row, naming absence and displayed presence over the admitted carrier.*
3. *S is the* $\mathsf{Sum}^{\uparrow}$ *branch row, naming left and right injections with their branch tags.*
4. *P is the* $\mathsf{Prod}^{\uparrow}$ *pair row, naming first and second projections as displayed components.*
5. *L is the* $\mathsf{List}^{\uparrow}$ *finite sequence row, naming nil, cons, length, and component readback ledgers.*
6. *H records componentwise* hsame *transport for all five finite-data rows and the structural rows.*
7. *C records displayed* Cont *replay from finite-data requests through B, O, S, P, L.*
8. *Q is the* Pkg *provenance over* $\mathsf{FiniteInterfaceSpine}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteInterfaceSpine}^{\uparrow}}$ *row.*

The carrier is an interface spine only. It has no coordinate for a host finite-type universe, an external eliminator, quotient equality of displayed finite objects, or a new BEDC kernel constructor.

Theorem 8899.2 (Finite interface spine NameCert obligations)**.** *For every accepted* $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ *carrier $F = (B, O, S, P, L, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteInterfaceSpine}^{\uparrow}}$ *surface consists exactly*

of carrier admission for the displayed five finite-data rows, constructor exposure for the Boolean, option, sum, product, and list interfaces, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8899.1. The rows B, O, S, P, L are the only finite-data rows available to an accepted packet.

Each constructor surface is inherited from its displayed sibling row. Boolean reads use the endpoint row B; option reads use the absent-or-present row O; sum reads use the branch row S; product reads use the projection row P; and list reads use the nil, cons, length, and component ledgers in L.

The structural rows H, C, Q, N transport, replay, source, and name only those displayed rows. Since the carrier has no host finite-type universe, external eliminator, quotient equality, or kernel constructor coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8899.3 (Finite interface spine L6 handoff)**.** *Every L6 consumer of* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, *or* $\mathsf{List}^{\uparrow}$ *through a* $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ *packet must display the finite route*

$$\mathsf{Bool}^{\uparrow}, \mathsf{Option}^{\uparrow}, \mathsf{Sum}^{\uparrow}, \mathsf{Prod}^{\uparrow}, \mathsf{List}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert} \longmapsto \mathsf{FiniteInterfaceSpine}^{\uparrow}.$$

The handoff exports the common finite-data interface packet and no host datatype primitive, hidden recursion principle, quotient equality, or kernel extension.

Proof. Start with the obligation surface of Definition 8899.2. It forces every consumer to read the five sibling finite-data rows before any common interface name is public.

The replay row C orders the route through the displayed Boolean, option, sum, product, and list certificates. The transport row H keeps the same five-row packet stable under hsame, while Q and N attach provenance and local naming.

Thus the L6 handoff is exactly the displayed finite-data packet. No additional coordinate can supply a host datatype primitive, hidden recursion principle, quotient equality, or kernel extension. □

Falsifiable prediction: Every accepted $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ consumer displays BoolUp, OptionUp, SumUp, ProdUp, ListUp, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before reading the common L6 finite-data interface.

Independence witness: bool, option, sum, prod, list: $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ is not bijective to any listed sibling because it binds the five finite-data rows into one common L6 interface packet with shared transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteInterfaceSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteInterfaceSpine}^{\uparrow}$ packet is generated from BoolUp endpoints, OptionUp nullable rows, SumUp branch tags, ProdUp projection rows, ListUp finite-sequence ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteInterfaceSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8899.1, Definition 8899.2, and Definition 8899.3 seed the common L6 finite-data interface over BoolUp, OptionUp, SumUp, ProdUp, and ListUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteInterfaceSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted recursion, quotient equality of finite objects, host datatype primitives, or a new BEDC kernel constructor.

Upgrade path: Obligation closure requires checked carrier admission, constructor exposure, L6 handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteInterfaceSpineUp.

Chapter 8900

A Concrete Naming Certificate for GrothendieckInequality$^{\uparrow}$

Following Grothendieck's inequality and Pisier's Banach-space formulation, the GrothendieckInequality$^{\uparrow}$ packet records the finite BEDC surface by which a displayed bilinear matrix estimate is routed through Banach, tensor, norm, real, and bounded-operator rows. It sits over the Banach source of chapter 90, the algebraic tensor surface of chapter 87, the bounded-operator interface of chapter 1644, the norm row of chapter 89, and the terminal scalar seal of chapter 38. The packet names a displayed inequality ledger only; it does not import the analytic proof of Grothendieck's theorem, a host supremum over all sign choices, selected optimizers, quotient tensor norms, or a global Banach-space completion principle.

Definition 8900.1 (Grothendieck inequality carrier)**.** *A* GrothendieckInequality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, F, T, O, A, B, K, H, C, P, N)$$

with the following visible rows:

1. *E and F are displayed* Banach$^{\uparrow}$ *source rows with their* Norm$^{\uparrow}$ *scalar-bound interfaces.*
2. *T is the* TensorProduct$^{\uparrow}$ *row naming the finite tensor table on which the bilinear estimate is read.*
3. *O is the* BoundedLinearOperator$^{\uparrow}$ *handoff row for the associated operator or bilinear form.*
4. *A is the finite coefficient ledger, listing the matrix entries, sign or vector probes, and their displayed norm bounds.*
5. *B is the scalar inequality ledger over* Real$^{\uparrow}$*, recording the two compared finite bounds and the displayed constant row K.*
6. *K is the Grothendieck-constant boundary row; it is a named scalar bound read through* Real$^{\uparrow}$*, not a selected optimal constant.*
7. *H records componentwise* hsame *transport for source rows, tensor rows, operator rows, coefficient ledgers, scalar inequality rows, constant rows, replay data, provenance, and local naming.*
8. *C records displayed* Cont *replay from a bilinear-estimate request through E, F, T, O, A, B, K.*
9. *P is the* Pkg *provenance over* GrothendieckInequality$^{\uparrow}$*,* Banach$^{\uparrow}$*,* TensorProduct$^{\uparrow}$*,* BoundedLinearOperator$^{\uparrow}$*,* Norm$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{GrothendieckInequality}^{\uparrow}}$ *row.*
The carrier exports only this finite inequality ledger. It has no coordinate for an ambient proof of the classical theorem, a universal supremum, a selected optimizer, quotient tensor-norm equality, or an unrestricted Banach completion.

Theorem 8900.2 (Grothendieck inequality NameCert obligations)**.** *For every accepted* $\mathsf{GrothendieckInequality}^{\uparrow}$ *carrier* $G = (E, F, T, O, A, B, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GrothendieckInequality}^{\uparrow}}$ *surface consists exactly of Banach source exposure, tensor-table exposure, bounded-operator handoff, finite coefficient-ledger admission, scalar inequality readback through* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, Grothendieck-constant boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8900.1. The source rows E, F, tensor row T, bounded-operator row O, coefficient ledger A, scalar inequality ledger B, and constant boundary K are the only analytic-facing rows available.

The inequality read is finite. It compares the displayed matrix or bilinear table in A with scalar bounds read through the source norm rows and the terminal $\mathsf{Real}^{\uparrow}$ ledger B, K.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores a host theorem proof, universal supremum, selected optimizer, quotient tensor norm, or unrestricted completion, the listed rows exhaust the local NameCert surface. □

Theorem 8900.3 (Grothendieck inequality tensor-norm handoff)**.** *Every tensor-norm or operator-ideal consumer of an accepted* $\mathsf{GrothendieckInequality}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{TensorProduct}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto E, F, T, O, A, B, K, H, C, P, N.$$

The handoff supplies the displayed coefficient ledger, scalar inequality ledger, and constant boundary as reusable finite data; it does not supply a completed tensor norm, an optimal-factorization theorem, quotient equality of operators, or host-level compactness.

Proof. Start with the obligation surface of Definition 8900.2. It forces the consumer to expose Banach endpoints, tensor data, bounded-operator handoff, coefficient entries, scalar bounds, and the constant boundary before the inequality row is public.

The tensor-norm consumer reads only the route through T, A, B, K. The bounded-operator consumer reads only the route through O, A, B, K. In both cases the norm and scalar comparisons remain displayed rows, not ambient supremum calculations.

Transport, replay, provenance, and naming then keep the same route attached to H, C, P, N. Because the packet lacks coordinates for completion, optimal factorization, quotient operator equality, and host compactness, the handoff exports only the finite inequality ledger. □

Falsifiable prediction: Every accepted $\mathsf{GrothendieckInequality}^{\uparrow}$ consumer displays Banach endpoints, TensorProductUp data, BoundedLinearOperatorUp handoff, NormUp scalar rows, RealUp bound rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before using the inequality ledger.

Independence witness: banach, tensorproduct, boundedlinearoperator, norm, real: $\mathsf{GrothendieckInequality}^{\uparrow}$ is not bijective to any listed sibling because it binds coefficient tables, bilinear or operator handoff, scalar inequality ledgers, Grothendieck-constant boundary, transport, replay, provenance, and local naming into one finite estimate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{GrothendieckInequality}^{\uparrow}}$.

Constructive story: A GrothendieckInequality$^{\uparrow}$ packet is generated from Banach endpoint rows, TensorProductUp finite tensor tables, BoundedLinearOperatorUp handoff rows, coefficient ledgers, NormUp and RealUp scalar inequality rows, a Grothendieck-constant boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GrothendieckInequality}^{\uparrow}}$, seed)

Scope closed: Definition 8900.1, Definition 8900.2, and Definition 8900.3 seed the finite Grothendieck inequality ledger over Banach, tensor-product, bounded-operator, norm, and real rows.

Formal verification: FormalStatus(NameCert$_{\text{GrothendieckInequality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, the analytic proof of Grothendieck's theorem, optimal constants, quotient tensor norms, selected optimizers, or unrestricted Banach completion.

Upgrade path: Obligation closure requires checked carrier admission, tensor-table exposure, bounded-operator handoff, scalar inequality readback, constant-boundary exposure, transport, replay, provenance, and local naming for BEDC.Derived.GrothendieckInequalityUp.

Chapter 8901

A Concrete Naming Certificate for $\mathsf{ParticularPointTopology}^{\uparrow}$

Following Alexandrov and standard finite-topology examples, the $\mathsf{ParticularPointTopology}^{\uparrow}$ packet records the included-point topology on a finite displayed carrier as BEDC NameCert data. It sits after the topology carrier of chapter 88, the two-point truth classifier of chapter 1860, and the separation boundary of chapter 3338. The packet names a distinguished point, the open-set classifier that accepts exactly the empty set and the displayed subsets containing that point, and the finite boundary where specialization may fail.

Definition 8901.1 (Particular point topology carrier)**.** *A* $\mathsf{ParticularPointTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (p, O, S, T, H, C, P, N)$$

with the following displayed rows:

1. *p is the distinguished point row on the finite carrier.*
2. *O is the open classifier row, accepting the empty displayed subset and every displayed subset whose membership row contains p.*
3. *S is the Sierpinski truth handoff row that reads membership in a particular-point open as a two-point truth observation.*
4. *T is the T_0-boundary row, recording which displayed point pairs are separated by the particular-point classifier and which pairs remain specialization-indistinguishable.*
5. *H records componentwise* hsame *transport for the distinguished point, open classifier, Sierpinski handoff, separation boundary, provenance, and naming rows.*
6. *C records displayed* Cont *replay through topology, Sierpinski truth, and T_0-reflection consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{ParticularPointTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{TZeroReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ParticularPointTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through p, O, S, T, H, C, P, N. It has no coordinate for host powersets, arbitrary union closure, quotient equality of opens, selected specialization classes, or completed separation reflection.

Theorem 8901.2 (Particular point topology NameCert obligations)**.** *For every accepted* $\mathsf{ParticularPointTopology}^{\uparrow}$ *carrier*

$$X = (p, O, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ParticularPointTopology}^{\uparrow}}$ *surface consists exactly of distinguished-point admission, open-classifier exposure through* O*, Sierpinski truth handoff through* S*,* T_0*-boundary exposure through* T*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8901.1. The distinguished point row, open classifier row, Sierpinski handoff, separation boundary, transport, replay, provenance, and local name are all visible coordinates of one finite packet.

The topology-facing obligation reads O only as a displayed classifier over finite subsets. The accepted open rows are the empty subset row and the rows that contain p, so no host powerset or arbitrary-union theorem is imported.

The truth and separation obligations are then read through S and T. Transport, replay, provenance, and local naming preserve the same finite rows and do not add quotient equality of opens or completed reflection data. □

Theorem 8901.3 (Particular point topology Sierpinski handoff)**.** *Every accepted* $\mathsf{ParticularPointTopology}^{\uparrow}$ *open-membership read factors through the Sierpinski truth row* S *before a topology consumer may use it as a classifier. The handoff reads the sibling* $\mathsf{SierpinskiSpace}^{\uparrow}$ *surface of chapter 1860 and returns only displayed truth data for the finite open row* O*.*

Proof. Start from the open classifier row O. A displayed subset is accepted as open either because it is the empty row or because its membership row contains the distinguished point p.

The row S repacks that membership test as a Sierpinski truth observation. Since S is a carrier row, the handoff is finite and visible rather than a host characteristic-function theorem.

Transport and replay through H, C keep the read attached to the same finite classifier. Thus topology consumers receive only the displayed truth handoff and cannot infer host powerset extensionality or arbitrary open-set equality. □

Theorem 8901.4 (Particular point topology T0 boundary)**.** *Every* T_0*-facing read of an accepted* $\mathsf{ParticularPointTopology}^{\uparrow}$ *packet must expose the boundary row* T *before it reaches chapter 3338. The row records the finite point pairs separated by opens containing* p *and the pairs that remain indistinguishable by the displayed classifier.*

Proof. Project the rows p, O, T from the carrier. Separation information is not stored as a global theorem about all opens; it is the displayed finite ledger T attached to the particular-point classifier.

The topology and Sierpinski rows supply the inspected opens and truth observations. The T_0-boundary row records which point pairs those displayed observations separate.

Finally H, C, P, N transport, replay, source, and name only that ledger. A T_0 consumer may use the boundary row as input to reflection, but the packet does not export completed reflection, quotient specialization classes, or ambient topological separation. □

Closure status for $\mathsf{NameCert}_{\mathsf{ParticularPointTopology}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ParticularPointTopology}^{\uparrow}}$, seed)

Scope closed: Definition 8901.1, Definition 8901.2, Definition 8901.3, and Definition 8901.4 seed the finite distinguished-point topology, truth-handoff, and separation-boundary surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ParticularPointTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host powerset topology, arbitrary union closure, quotient equality of opens, selected specialization classes, completed T_0 reflection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-classifier exactness, Sierpinski handoff, T_0 boundary replay, transport, provenance, and local naming for BEDC.Derived.ParticularPointTopologyUp.

Constructive story: A ParticularPointTopology$^{\uparrow}$ packet is generated from a distinguished point row, finite open classifier row, Sierpinski truth handoff, T_0 boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8902

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusNerve}^{\uparrow}$

Following Heine–Cantor and Bishop compactness machinery, the $\mathsf{CompactUniformModulusNerve}^{\uparrow}$ packet records the finite nerve route by which compact finite covers, local continuity moduli, rational lower-bound folds, and a $\mathsf{UniformModulus}^{\uparrow}$ handoff are read as one NameCert object. It realizes the finite-cover branch of chapter 9237 and sits beside the finite cover nerve of chapter 7597, the compact metric source of chapter 129, the continuous-map surface of chapter 130, the rational comparison row of chapter 37, and the downstream uniform-modulus consumer of chapter 369.

Definition 8902.1 (Compact uniform modulus nerve carrier)**.** *A* $\mathsf{CompactUniformModulusNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, L, R, B, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover source row, displaying the inspected* $\mathsf{ProbeBundle}$ *centers and cover-radius histories.*
2. *F is the finite nerve row, recording cover membership, overlap compatibility, and the accepted finite faces used by the modulus extraction.*
3. *L is the local-modulus row read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface, assigning displayed source neighborhoods to displayed target tolerances.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound comparison row for the cover radii and local modulus radii.*
5. *B is the finite rational-fold row that computes the uniform positive lower-bound ledger.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed after K, F, L, R, B have been displayed.*
7. *H records componentwise* hsame *transport for cover, nerve, local-modulus, rational-fold, handoff, provenance, and naming rows.*
8. *C records displayed* Cont *replay through finite covers, nerve faces, local moduli, rational folds, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusNerve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNerve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, R, B, U, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, selected subcovers, host function equality, countable choice, quotient equality of covers, or ambient real completeness.

Theorem 8902.2 (Compact uniform modulus nerve NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusNerve}^{\uparrow}$ *carrier*

$$M = (K, F, L, R, B, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNerve}^{\uparrow}}$ *surface consists exactly of compact finite-cover exposure through K, finite nerve exposure through F, local-modulus exposure through L, rational comparison through R, finite lower-bound folding through B, uniform-modulus handoff through U, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8902.1. The compact-cover, nerve, local-modulus, rational-fold, uniform-modulus, transport, replay, provenance, and naming rows are the complete inventory of the packet.

The cover and nerve obligations are read through K, F. These rows expose finite ProbeBundle data and accepted faces, not an arbitrary open cover or selected subcover.

The modulus obligations are read through L, R, B, U. Local continuity data is displayed before the rational lower-bound fold, and the uniform-modulus handoff is downstream of that fold.

Finally H, C, P, N preserve and name only these rows. No host function equality, countable choice principle, quotient cover equality, or ambient completeness theorem is exported. □

Theorem 8902.3 (Compact uniform modulus nerve vision handoff)**.** *The compact finite-cover, local-modulus,* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold, and* $\mathsf{UniformModulus}^{\uparrow}$ *handoff route of chapter 9237 is realized by an accepted* $\mathsf{CompactUniformModulusNerve}^{\uparrow}$ *carrier as the finite row sequence*

$$K, F, L, R, B \longmapsto U, H, C, P, N.$$

The handoff is a concrete NameCert packet and not an external Heine–Cantor theorem.

Proof. Read the vision chapter through its finite-cover branch. The compact ProbeBundle and cover-radius histories land in K, and the finite nerve membership and overlap checks land in F.

The local continuity witnesses land in L. Their radii are compared through R and folded by B before any uniform-modulus row is exposed.

The row U is therefore downstream of finite cover, nerve, local-modulus, and rational-fold data. Transport, replay, provenance, and local naming keep the handoff inside the same packet, so the construction does not invoke an external compactness theorem or selector. □

Theorem 8902.4 (Compact uniform modulus nerve no-choice boundary)**.** *Every accepted* CompactUniformModulus *read keeps the finite nerve and rational lower-bound fold visible before the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. Consequently the packet exports no arbitrary open-cover compactness principle, selected subcover, countable choice functional, quotient equality of covers, host function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start from the carrier rows $K, F, L, R, B, U, H, C, P, N$. The only route to U passes through the finite cover row, nerve row, local-modulus row, rational comparison row, and finite fold row.

The sibling finite cover nerve chapter supplies a displayed cover-nerve route, while the compact metric, continuous-map, rational, and uniform-modulus chapters supply the rows consumed by the packet. None of these rows is a choice functional or host equality principle.

Since transport and replay preserve only the displayed carrier coordinates, a consumer cannot bypass F, R, B to obtain a modulus from an external compactness theorem. The refused choice, quotient, host-function, and completeness claims remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusNerve}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8902.1, Definition 8902.2, Definition 8902.3, and Definition 8902.4 seed the finite cover-nerve, local-modulus, rational-fold, and uniform-modulus handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusNerve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, countable choice, quotient cover equality, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite nerve exactness, local-modulus replay, rational-fold stability, uniform-modulus handoff, and no-choice boundary proofs for BEDC.Derived.CompactUniformModulusNerveUp.

Constructive story: A $\mathsf{CompactUniformModulusNerve}^{\uparrow}$ packet is generated from CompactMetric finite-cover rows, finite nerve rows, ContinuousMap local-modulus rows, Rat lower-bound comparison rows, rational-fold rows, UniformModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8903

A Concrete Naming Certificate for $\mathsf{Hypercover}^{\uparrow}$

Following Grothendieck and Verdier on hypercovers, the $\mathsf{Hypercover}^{\uparrow}$ packet records the finite BEDC surface by which a displayed Grothendieck topology is refined into a simplicial cover diagram and then consumed by sheaf-descent data. It sits after the site carrier of chapter 6718, the cover-to-simplicial inventory of chapter 6755, the sheaf-facing refinement route of chapter 7336, and the cocycle-effectivity surface of chapter 8142. The packet names levelwise cover rows, matching-object comparison rows, Cech truncation rows, descent handoff rows, transport, replay, provenance, and local naming; it is not a host simplicial presheaf, a selected coskeleton, a model-structure fibrancy theorem, a quotient stack, or global choice of refinements.

Definition 8903.1 (Hypercover carrier)**.** *A* $\mathsf{Hypercover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (J, U, M, C, S, D, T, R, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{GrothendieckTopology}^{\uparrow}$ *site row, carrying the covering-sieve, pullback-stability, and transitivity data over which all levels are read.*
2. *U is the levelwise cover row, listing the displayed cover data for each finite simplicial level used by the packet.*
3. *M is the matching-object comparison row, recording the finite boundary request that each level must cover before the next level is public.*
4. *C is the* $\mathsf{CechNerve}^{\uparrow}$ *truncation row, identifying the ordinary cover nerve recovered at the initial levels.*
5. *S is the* $\mathsf{Sheafification}^{\uparrow}$ *compatibility row, requiring locality and gluing reads to remain attached to the same site and levelwise cover data.*
6. *D is the* $\mathsf{DescentData}^{\uparrow}$ *handoff row, exposing cocycle and effectivity requests only after the levelwise and matching rows have been displayed.*
7. *T records componentwise* hsame *transport for the site, levelwise cover, matching-object, Cech, sheafification, descent, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a sheaf-descent request through J, U, M, C, S, D, T.*

9. *P is the* Pkg *provenance over* $\mathsf{Hypercover}^{\uparrow}$, $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{CechNerve}^{\uparrow}$, $\mathsf{Sheafification}^{\uparrow}$, $\mathsf{DescentData}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{Hypercover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J, U, M, C, S, D, T, R, P, N$. It has no coordinate for host simplicial presheaves, selected coskeleta, model structures, quotient stacks, global refinement choice, ambient sheaf equality, or bridge credentials.

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverTasteGate single carrier alignment`

Theorem 8903.2 (Hypercover NameCert obligations)**.** *For every accepted* $\mathsf{Hypercover}^{\uparrow}$ *carrier $H = (J, U, M, C, S, D, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Hypercover}^{\uparrow}}$ *surface consists exactly of site exposure through J, levelwise cover admission through U, matching-object comparison through M, Cech truncation through C, sheafification compatibility through S, descent handoff through D, componentwise transport through T, displayed replay through R, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverNameCertObligations`

Proof. Project the carrier of Definition 8903.1. The site, levelwise cover, matching-object, Cech, sheafification, and descent rows are exactly J, U, M, C, S, D.

The site row is read first. It supplies the covering-sieve and pullback-stability data that make every finite levelwise cover meaningful inside the same $\mathsf{GrothendieckTopology}^{\uparrow}$ packet.

The levelwise and matching rows then control the simplicial cover request. The row U displays the cover data at each finite level, while M records the boundary or matching comparison that must be covered before a higher-level read is exported.

Finally T, R, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for a host simplicial presheaf, selected coskeleton, model structure, quotient stack, global refinement choice, ambient sheaf equality, or bridge credential, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverCarrier namecert obligations`

Theorem 8903.3 (Hypercover Cech nerve route)**.** *Every accepted* $\mathsf{Hypercover}^{\uparrow}$ *packet exposes its* $\mathsf{CechNerve}^{\uparrow}$ *row before any sheaf-descent consumer may read a higher matching level. The route from an ordinary covering family to a hypercover factors through J, U, M, C, T, R, P, N, so the initial Cech data remain visible while the packet adds levelwise matching coverage.*

Proof. Start with the carrier of Definition 8903.1. The Cech truncation row C is a displayed coordinate, not a derived host object.

Read the ordinary cover surface through chapter 6755. Its cover-member, intersection-witness, nerve-face, stability, replay, provenance, and local naming rows supply the initial finite simplicial data consumed by C.

Now expose U and M. They extend the initial Cech route level by level, but every matching comparison still reads the same site row J and the same Cech truncation row C before a higher-level cover is public.

The rows T, R, P, N preserve this route componentwise. Therefore the hypercover packet refines the Cech nerve surface without exporting selected intersection points, host realizations, quotient simplicial equality, or ambient homology data. □

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverCechNerveRoute`

Theorem 8903.4 (Hypercover sheaf descent handoff)**.** *Every sheaf-facing read of an accepted* $\mathsf{Hypercover}^{\uparrow}$ *carrier must expose the* $\mathsf{GrothendieckTopology}^{\uparrow}$ *site row, the* $\mathsf{Sheafification}^{\uparrow}$ *compatibility row, and the* $\mathsf{DescentData}^{\uparrow}$ *handoff row before consuming descent information. The exported route is*

$$\mathsf{GrothendieckTopology}^{\uparrow} \longmapsto \mathsf{CechNerve}^{\uparrow} \longmapsto \mathsf{Sheafification}^{\uparrow} \longmapsto \mathsf{DescentData}^{\uparrow}$$

together with $J, U, M, C, S, D, T, R, P, N$*.*

Proof. Apply the NameCert obligation surface of Definition 8903.2. It places J, U, M, C, S, D before replay, provenance, and local naming.

The sheafification row S reads locality and gluing only over the displayed site and levelwise cover rows. Thus a sheaf consumer cannot bypass J, U, M, C to obtain an external reflector or quotient of compatible families.

The descent row D is downstream of S. It exposes cocycle and effectivity requests after the levelwise matching route is visible, so descent data remain a finite handoff rather than a host quotient stack or selected global gluing.

Finally T, R, P, N preserve the same route. The displayed chain through $\mathsf{GrothendieckTopology}^{\uparrow}$, $\mathsf{CechNerve}^{\uparrow}$, $\mathsf{Sheafification}^{\uparrow}$, and $\mathsf{DescentData}^{\uparrow}$ is therefore the whole sheaf-descent handoff carried by $\mathsf{Hypercover}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverCarrier sheaf descent handoff`

Theorem 8903.5 (Hypercover covering-family nonescape)**.** *No accepted* $\mathsf{Hypercover}^{\uparrow}$ *read can export a covering family without also displaying the source site row* J*, the levelwise cover row* U*, the matching-object row* M*, the Cech row* C*, the sheafification row* S*, the descent row* D*, transport* T*, replay* R*, provenance* P*, and local naming* N*. In particular, hypercover consumers obtain no selected refinement schedule, host simplicial presheaf, coskeleton operator, quotient stack, ambient sheaf equality, or global cover choice from the packet.*

Lean4 ✓ `BEDC.Derived.HypercoverUp.HypercoverCarrier covering family nonescape`

Proof. Project the carrier and read the classifier coordinates. Covering-family information appears only in J, U, M, C, S, D and is preserved only by T, R, P, N.

Suppose a consumer reaches a displayed covering family. The site row J provides the ambient coverage discipline, U provides the finite levelwise cover, and M records the matching comparison that justifies the level. The Cech row C keeps the initial cover nerve visible.

For sheaf or descent reads, S and D must also be exposed. They route locality, gluing, cocycle, and effectivity through the same finite levelwise data rather than through a selected external construction.

Since the excluded objects are not carrier coordinates, componentwise transport, replay, provenance, and local naming cannot create them. Thus every covering-family read remains inside the displayed hypercover packet. □

Falsifiable prediction: An accepted $\mathsf{Hypercover}^{\uparrow}$ consumer that reads descent along a hypercover without displaying the Grothendieck topology, Cech nerve, sheafification, descent-data, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: grothendieck topology, cech nerve, sheafification, descent data: $\mathsf{Hypercover}^{\uparrow}$ is not bijective to any listed sibling because it binds levelwise cover rows and matching-object comparisons between the site, Cech truncation, sheafification compatibility, descent handoff, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{Hypercover}^{\uparrow}}$.

Constructive story: A $\mathsf{Hypercover}^{\uparrow}$ packet is generated from a GrothendieckTopologyUp site row, levelwise cover rows, matching-object comparison rows, a CechNerveUp truncation row, a SheafificationUp compatibility row, a DescentDataUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Hypercover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8903.1, Definition 8903.2, Definition 8903.3, Definition 8903.4, and Definition 8903.5 seed the finite hypercover route from site coverage through Cech truncation to sheaf-descent consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Hypercover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host simplicial presheaves, selected coskeleta, model-structure fibrancy, quotient stacks, selected refinement schedules, global cover choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, levelwise cover exposure, matching-object comparison, Cech nerve routing, sheafification compatibility, descent handoff, transport, replay, provenance, and local naming for BEDC.Derived.HypercoverUp.

Chapter 8904

A Concrete Naming Certificate for $\mathsf{Preuniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures and preuniform spaces, the $\mathsf{Preuniformity}^{\uparrow}$ packet records the finite entourage-base surface that sits between topology, metric data, and uniform completion. It is read beside the uniform-space certificate of chapter 339, the metric carrier of chapter 57, the topology carrier of chapter 88, the finite observation discipline of chapter 126, and the terminal real-name seal of chapter 38. The packet names a non-symmetric uniformity-like carrier before a separated or symmetric uniform-space object is available.

Falsifiable prediction: Every accepted $\mathsf{Preuniformity}^{\uparrow}$ packet displays source, entourage, stability, base, transport, replay, provenance, and local naming rows before a UniformSpaceUp or UniformCompletionUp consumer may read it.

Independence witness: topology, metric, uniformspace, streamname, real: no listed sibling is bijective to $\mathsf{Preuniformity}^{\uparrow}$. Topology records open-neighbourhood data without an entourage base; metric supplies symmetric distance rows; UniformSpaceUp already enforces the uniform-space surface; StreamNameUp records finite schedules; RealUp seals scalar names. The preuniform packet preserves the directed entourage route separately.

Definition 8904.1 (Preuniformity carrier)**.** *A* $\mathsf{Preuniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, E, B, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-row carrier whose points, observations, or names are being compared;*
2. *E is a finite entourage ledger, recording displayed binary-neighbourhood rows without imposing symmetry;*
3. *B is the finite filter-base ledger, recording upward closure and finite-intersection refinements for displayed entourage rows;*
4. *T is the topology-readback row, recording the neighbourhoods induced by the displayed entourage base;*
5. *H records componentwise* hsame *transport for S, E, B, T and the local naming row;*
6. *C records displayed* Cont *replay through source admission, entourage lookup, base refinement, topology readback, provenance, and local naming;*

7. *P is the* Pkg *provenance over* $\mathsf{Preuniformity}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{Preuniformity}^{\uparrow}}$ *row.*

The carrier exports no symmetry axiom, separatedness theorem, quotient completion, selected Cauchy limit, host entourage filter, countable-choice base, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.PreuniformityUp.PreuniformityTasteGate single carrier alignment`

Theorem 8904.2 (Preuniformity entourage stability)**.** *For every accepted* $\mathsf{Preuniformity}^{\uparrow}$ *carrier* $U = (S, E, B, T, H, C, P, N)$*, an entourage read is stable only through the displayed ledger E, the base-refinement ledger B, and componentwise transport H. A downstream* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer may preserve the same directed row, but it may not replace it by a symmetric metric ball, a separated uniformity, a quotient completion row, or a selected completed point.*

Proof. Project the accepted carrier. The only binary-neighbourhood data in the packet is the displayed entourage ledger E, and the only refinement data is the base ledger B.

Transport through H preserves rows componentwise. It therefore preserves the source, entourage, base, topology readback, and naming rows already present in U without adding symmetry, separatedness, or completion data.

The sibling consumers may cite the preserved directed row only as a preuniform row. Since the carrier has no coordinate for metric symmetry, quotient completion, selected limits, host filters, countable choice, or ambient Real completeness, no such datum can be exported by the entourage read. □

Lean4 ✓ `BEDC.Derived.PreuniformityUp.PreuniformityCarrier entourage stability`

Theorem 8904.3 (Preuniformity filter-base route)**.** *For every accepted* $\mathsf{Preuniformity}^{\uparrow}$ *carrier* $U = (S, E, B, T, H, C, P, N)$*, the filter-base row B is read after the entourage ledger E and before topology readback T. The row records finite upward and intersection refinements for displayed entourages only; it does not assert a completed host filter, a symmetric uniformity, or a UniformCompletionUp limit object.*

Proof. Read the carrier rows in order. The entourage ledger E supplies the displayed binary-neighbourhood rows, and B records the finite refinements allowed for those rows.

The topology row T is downstream of the same finite base. It may expose neighbourhood data through chapter 88, but it cannot turn the preuniform base into a host filter or a completed uniform space.

The continuation row C and provenance row P replay and source the same finite route. They do not add symmetry, separatedness, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.PreuniformityUp.PreuniformityCarrier filter base route`

Theorem 8904.4 (Preuniformity NameCert obligations)**.** *For the carrier of Definition 8904.1, the local* $\mathsf{NameCert}_{\mathsf{Preuniformity}^{\uparrow}}$ *surface consists exactly of carrier admission for* $U = (S, E, B, T, H, C, P, N)$*, source exposure through S, directed entourage exposure through E, finite filter-base exposure through B, topology readback through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission exposes the finite tuple S, E, B, T, H, C, P, N. The source row, entourage ledger, base ledger, topology readback, transport, replay, provenance, and local naming row are all visible coordinates of one packet.

Apply the entourage-stability theorem and the filter-base route theorem. Together they keep binary-neighbourhood reads directed, place base refinements between entourage lookup and topology readback, and prevent a consumer from replacing the preuniform carrier by a symmetric uniformity or completion object.

The listed rows therefore exhaust the local NameCert obligations. There is no additional coordinate from which to read separatedness, metric symmetry, quotient completion, selected limits, host filters, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.PreuniformityUp.PreuniformityNameCertObligations`

Closure status for $\mathsf{NameCert}_{\mathsf{Preuniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{Preuniformity}^{\uparrow}$ packet is generated from a source row, a directed entourage ledger, a finite filter-base ledger, topology readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Preuniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8904.1, Definition 8904.2, Definition 8904.3, and Definition 8904.4 seed the directed entourage-base route between topology, metric data, and uniform completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Preuniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, uniform-space symmetry, separatedness, quotient completion, selected Cauchy limits, countable choice, host filters, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, directed entourage stability, finite filter-base routing, topology readback, provenance, and local naming for BEDC.Derived.PreuniformityUp.

Chapter 8905

A Concrete Naming Certificate for $\mathsf{SchreierSims}^{\uparrow}$

Following Sims' stabilizer-chain method and the computational group theory treatments of Seress, the $\mathsf{SchreierSims}^{\uparrow}$ packet records a finite permutation-group certificate as BEDC NameCert data. It consumes the permutation source of chapter 116, the group source of chapter 41, the orbit and stabilizer rows of chapter 6559, and the finite action rows of chapter 3662. The packet names a displayed stabilizer chain, transversal ledger, Schreier generator rows, sift trace, transport, replay, provenance, and local naming; it does not implement a host permutation-group algorithm or claim a complete constructive base-change strategy.

Definition 8905.1 (Schreier-Sims carrier)**.** *A* $\mathsf{SchreierSims}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, G, O, R, B, T, W, F, H, C, Q, N)$$

with the following displayed rows:

1. *P is the consumed* $\mathsf{Permutation}^{\uparrow}$ *source row, exposing the finite moved-set and graph data used by the action.*
2. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *source row, exposing multiplication, unit, inverse, and finite support rows for the generated permutation group.*
3. *O is the consumed* $\mathsf{OrbitStabilizer}^{\uparrow}$ *row, used to read each orbit and stabilizer boundary before a chain level is accepted.*
4. *R is the consumed* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, used only as the finite action interface through which permutation rows act on the displayed base points.*
5. *B is the finite base row, listing the inspected base points and the order in which stabilizer levels are read.*
6. *T is the transversal ledger, recording for each displayed orbit level the chosen finite transversal rows and their source in P, G, O, R.*
7. *W is the Schreier-generator row, recording the finite generator words obtained from the displayed transversal rows and group generators.*
8. *F is the sift-trace row, recording the finite reduction route of a displayed permutation through the stabilizer chain.*

9. *H is componentwise* hsame *transport for P, G, O, R, B, T, W, F.*
10. *C is displayed* Cont *replay of the stabilizer-chain read from permutation, group, orbit-stabilizer, and representation rows into the local certificate.*
11. *Q is the* Pkg *provenance over* Permutation$^{\uparrow}$, Group$^{\uparrow}$, OrbitStabilizer$^{\uparrow}$, FiniteGroupRepresentation$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{SchreierSims}^{\uparrow}}$ *row.*

No coordinate stores a host permutation-group procedure, ambient coset table, quotient action category, selected global base-change oracle, or proof that a particular implementation terminates.

Lean4 (statement) `BEDC.Derived.SchreierSimsUp`

Theorem 8905.2 (Schreier-Sims NameCert obligations)**.** *For every accepted* SchreierSims$^{\uparrow}$ *carrier*

$$S = (P, G, O, R, B, T, W, F, H, C, Q, N),$$

the local NameCert$_{\text{SchreierSims}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed permutation source, group source, orbit-stabilizer rows, finite action row, base row, transversal ledger, Schreier-generator row, sift-trace row, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read from the displayed finite rows before any consumer may treat the packet as a stabilizer-chain certificate.*

Proof. Project the carrier of Definition 8905.1. The first four rows expose the consumed permutation, group, orbit-stabilizer, and finite-action sources; the next four rows expose the base, transversals, Schreier generators, and sift trace.

The local naming obligations are then componentwise. The transversal ledger must cite the displayed orbit-stabilizer row, the generator ledger must cite the same transversal and group rows, and the sift trace must move through the displayed stabilizer levels in base order.

Finally H, C, Q, N transport, replay, source, and name only those finite rows. Since the packet has no host algorithm, quotient action category, or global base-change coordinate, the NameCert surface is exactly the displayed stabilizer-chain data. □

Theorem 8905.3 (Schreier-Sims stabilizer-chain handoff)**.** *Every* SchreierSims$^{\uparrow}$ *consumer that reads a stabilizer chain must first expose the base row B, the orbit-stabilizer row O, the transversal ledger T, the Schreier-generator row W, and the sift-trace row F over the consumed* Permutation$^{\uparrow}$, Group$^{\uparrow}$, *and* FiniteGroupRepresentation$^{\uparrow}$ *sources. The handoff is a finite BEDC certificate for the displayed chain; it is not a host membership algorithm, a coset-enumeration primitive, or a complete permutation-group computation engine.*

Proof. Begin with the base row B. It fixes the level order in which stabilizers can be read, so a consumer cannot inspect a later sift endpoint before the earlier orbit and stabilizer rows are displayed.

Next use the orbit-stabilizer and action sources O and R. They provide the finite orbit and stabilizer boundaries used by the transversal ledger T, with the permutation and group rows P, G still visible.

Then read the Schreier-generator row W and sift-trace row F. These rows record the finite generator and reduction data for the displayed chain, while H, C, Q, N preserve and name the same packet. Thus the exported handoff is exactly a finite stabilizer-chain certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchreierSims}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchreierSims}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8905.1, Definition 8905.2, and Definition 8905.3 seed the finite permutation-group stabilizer-chain carrier over displayed permutation, group, orbit-stabilizer, finite-action, base, transversal, Schreier-generator, and sift-trace rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchreierSims}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SchreierSimsUp`

Bridge status: none

Not claimed: This chapter does not close host permutation-group algorithms, complete coset enumeration, global base-change selection, implementation termination, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transversal exactness, Schreier-generator stability, sift-trace replay, stabilizer-chain handoff, transport, provenance, and local naming for the same finite packet.

Constructive story: A $\mathsf{SchreierSims}^{\uparrow}$ packet is generated from displayed PermutationUp, GroupUp, OrbitStabilizerUp, and FiniteGroupRepresentationUp source rows, a finite base row, transversal ledger, Schreier-generator row, sift-trace row, componentwise transport, displayed replay, provenance, and local naming.

Chapter 8906

A Concrete Naming Certificate for $\mathsf{FineUniformity}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{FineUniformity}^{\uparrow}$ packet records the fine uniformity carried by a completely regular topology as finite BEDC naming data. It sits between the topology source of chapter 88, the real-valued separation surface of chapter 5161, the uniform-space carrier of chapter 339, the relative-open route of chapter 5993, and the terminal real-name seal of chapter 38. The packet names the displayed entourage base generated by real-valued continuous probes; it is not a selected filter, a host powerset, a choice principle for entourages, a metrization theorem, or a completion claim.

Definition 8906.1 (Fine uniformity carrier)**.** *A* $\mathsf{FineUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, C, U, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row, exposing the open-index classifier, neighbourhood reads, finite-intersection rows, and arbitrary-union rows used by the packet.*
2. *C is the* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *row, supplying the real-valued point-versus-closed-set separation route over the same topology source.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *row whose entourages are read only through the displayed fine-uniformity base.*
4. *S is the* $\mathsf{SubspaceTopology}^{\uparrow}$ *compatibility row, recording that relative opens admitted by a displayed subspace read the same topology source before any restricted uniformity request.*
5. *R is the* $\mathsf{Real}^{\uparrow}$*-valued continuous-probe row. Each accepted probe is a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *read into* $\mathsf{Real}^{\uparrow}$ *whose inverse uniform window contributes an entourage-base entry.*
6. *E is the finite entourage-base ledger generated from the displayed real-valued probes and their finite intersection reads.*
7. *H records componentwise* hsame *transport for $T, C, U, S, R, E, K, P, N$ and for the displayed probe-window and entourage-base rows.*
8. *K records displayed* Cont *replay from a fine-uniformity request through topology, complete-regularity, uniform-space, subspace, real-probe, entourage-base, transport, provenance, and naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{FineUniformity}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompletelyRegularSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SubspaceTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FineUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, U, S, R, E, H, K, P, N$. It has no coordinate for a host powerset of $X \times X$, a selected entourage filter, a choice of all real-valued functions, a metric generator, a completion object, or ambient real completeness.

Theorem 8906.2 (Fine uniformity NameCert obligations)**.** *For every accepted* $\mathsf{FineUniformity}^{\uparrow}$ *carrier*

$$F = (T, C, U, S, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FineUniformity}^{\uparrow}}$ *surface consists exactly of topology-source admission, complete-regularity source admission, uniform-space handoff, subspace-topology compatibility,* $\mathsf{Real}^{\uparrow}$*-valued continuous-probe exposure, finite entourage-base exposure, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every fine-uniformity consumer must read the route*

$$T, C, U, S, R, E, H, K, P, N$$

before using an entourage as public.

Proof. Project the carrier of Definition 8906.1. The accepted packet displays the topology source, completely regular source, uniform-space row, subspace compatibility row, real-valued probe row, finite entourage-base ledger, and structural rows.

The real-valued probes are read before the entourage base. The row C supplies the point-versus-closed-set separation route, while R records only displayed $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ probes over the same topology source.

The entourage-base row E is finite and generated from the displayed probe windows. A consumer may inspect an entourage entry or a finite-intersection read already stored in E, but it cannot replace that row by a host powerset, a selected filter, or a choice of all probes.

Finally H, K, P, N transport, replay, source, and locally name the same finite route. Hence the listed rows exhaust the local NameCert surface, and no metric generator, completion object, selected filter, host powerset, or ambient real-completeness principle can enter the packet. □

Theorem 8906.3 (Fine uniformity compatible-topology handoff)**.** *Let $F = (T, C, U, S, R, E, H, K, P, N)$ be an accepted* $\mathsf{FineUniformity}^{\uparrow}$ *carrier. Any* $\mathsf{Topology}^{\uparrow}$, $\mathsf{SubspaceTopology}^{\uparrow}$, *or* $\mathsf{UniformSpace}^{\uparrow}$ *consumer that reads the topology induced by the fine uniformity must first expose the displayed topology source T, complete-regularity route C, real-valued probe row R, finite entourage-base ledger E, and structural rows H, K, P, N.*

Consequently the topology handoff is the finite route

$$\mathsf{Topology}^{\uparrow}, \mathsf{CompletelyRegularSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{UniformSpace}^{\uparrow}, \mathsf{SubspaceTopology}^{\uparrow} \longmapsto H, K, P, N,$$

not a host powerset topology, a selected entourage filter, a metric presentation, a completion theorem, or a choice principle for probes.

Proof. Start with the carrier definition. The topology source T and complete-regularity row C are the only entries that expose the open and real-valued separation vocabulary used by the packet.

Read the real-valued probe row R and the finite entourage-base ledger E. These rows display how the fine-uniformity windows are generated before the $\mathsf{UniformSpace}^{\uparrow}$ handoff is consumed, so the induced topology cannot be read from an untracked filter or host powerset.

For subspaces, the row S keeps the same topology source visible before restriction. Thus a $\mathsf{SubspaceTopology}^{\uparrow}$ consumer receives a relative-open route tied to T and E, not an independent restricted uniformity object.

Finally H, K, P, N transport, replay, source, and name only the displayed rows. The handoff therefore exports compatible topology data through the finite certificate route while refusing selected filters, host powersets, metric presentations, completion claims, and choice over probes. □

Closure status for $\mathsf{NameCert}_{\mathsf{FineUniformity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FineUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8906.1, Definition 8906.2, and Definition 8906.3 seed the fine-uniformity NameCert surface over topology, complete-regularity, uniform-space, subspace-topology, real-valued probe, entourage-base, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FineUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected filters, host powersets, choice over all probes, metrization, completion, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-valued probe exactness, finite entourage-base stability, compatible-topology handoff, subspace restriction compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FineUniformityUp.

Constructive story: A $\mathsf{FineUniformity}^{\uparrow}$ packet is generated from an ambient TopologyUp row, a CompletelyRegularSpaceUp row, a UniformSpaceUp row, a SubspaceTopologyUp compatibility row, RealUp-valued ContinuousMapUp probe rows, a finite entourage-base ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8907

A Concrete Naming Certificate for $\mathsf{LocallyBoundedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's local-boundedness usage in general topology, the $\mathsf{LocallyBoundedFunction}^{\uparrow}$ packet records the finite surface by which a displayed function read is bounded on one local source window before any Real or Completion consumer may use it. It sits beside the continuous-map carrier of chapter 130, the finite window discipline of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the bounded-family comparison surface of chapter 3522. The packet names local boundedness by finite rows only; it does not assert compactness, ambient completeness, quotient equality of functions, countable choice, selected global bounds, or bridge checking.

Definition 8907.1 (Locally bounded function carrier)**.** *A* $\mathsf{LocallyBoundedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, V, B, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the local source-window row, naming the finite neighborhood or bounded source portion on which the read is inspected.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose graph or evaluation interface supplies the function read on K.*
3. *V is the* $\mathsf{Real}^{\uparrow}$ *value-read row exposed by the function on the displayed source window.*
4. *B is the finite bound row, recording the displayed local upper budget against which the value read is compared.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened for the source inspection, value read, and bound request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations used by V and B.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic scale, slack, and comparison budget for the local bound.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, F, V, B, W, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the local source window, continuous-map read, value row, finite bound, StreamName, RegSeqRat, DyadicRatCore, Real seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay through*

$$K \longrightarrow F \longrightarrow V \longrightarrow B \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow A.$$

11. *P is the* Pkg *provenance over* $\mathsf{LocallyBoundedFunction}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedFunctionFamily}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyBoundedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, V, B, W, R, D, A, H, C, P, N$. It has no coordinate for an ambient function space, a selected global bound, quotient equality of functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 8907.2 (Locally bounded function NameCert obligations)**.** *For every accepted* $\mathsf{LocallyBoundedFunction}^{\uparrow}$ *carrier*

$$L = (K, F, V, B, W, R, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocallyBoundedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the local source window, ContinuousMap read, Real value row, finite bound, StreamName window, RegSeqRat readback, DyadicRatCore tolerance, terminal Real seal, transport, replay, provenance, and local naming rows; local-source exactness requiring K before any bound is inspected; value-bound exactness requiring V and B to be compared through R and D; and structural stability requiring H, C, P, N to preserve the displayed route.*

Lean4 ✓ `BEDC.Derived.LocallyBoundedFunctionUp.LocallyBoundedFunctionCarrier namecert obligations`

Proof. Project the carrier of Definition 8907.1. The rows $K, F, V, B, W, R, D, A, H, C, P, N$ are the only rows available to an accepted packet, so carrier admission is exactly the displayed finite list.

The local-source obligation is supplied by K and F. A consumer first opens the local source window and then reads the ContinuousMap row on that window, so the local bound is not interpreted as a host-global function-space assertion.

The value-bound obligation is supplied by V, B, R, D. The Real value and finite bound are exposed through regular rational readback and dyadic tolerance comparison before the terminal Real seal can be reached.

Finally, H, C, P, N transport, replay, source, and name the same route. Since the carrier has no rows for ambient completeness, host function-space equality, selected global bounds, quotient equality, countable choice, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 8907.3 (Locally bounded function finite-window bound route)**.** *Every Real-facing or L10 completion-facing consumer of an accepted* $\mathsf{LocallyBoundedFunction}^{\uparrow}$ *packet reaches a local bound only through the finite route*

$$K \longrightarrow F \longrightarrow V \longrightarrow B \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow A$$

with componentwise transport, replay, provenance, and naming supplied by H, C, P, N. The route may be consumed by compact-continuity, equicontinuity, and bounded-family chapters that need local finite bounds, but it does not export ambient completeness, host function spaces, selected global bounds, quotient equality, countable choice, or bridge checking.

Proof. Use the carrier projection of Definition 8907.1. The source window K and ContinuousMap row F are displayed before any value or bound is read.

Apply Definition 8907.2. Its value-bound exactness clause forces the Real value row V and the finite bound row B to pass through the StreamName, RegSeqRat, and DyadicRatCore rows W, R, D before the terminal Real seal A is public.

The structural rows H, C, P, N do not add analytic content. They preserve and name the same finite route, so any compact-continuity, equicontinuity, or bounded-family consumer receives only the displayed local bound route and no ambient completeness theorem, host function-space equality, selected global bound, quotient equality, countable-choice schedule, or bridge credential. □

Lean4 ✓ `BEDC.Derived.LocallyBoundedFunctionUp.LocallyBoundedFunctionFiniteWindowBoundRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyBoundedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyBoundedFunction}^{\uparrow}$ packet is generated from a local source-window row, ContinuousMap row, Real value read, finite bound row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyBoundedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8907.1, Definition 8907.2, and Definition 8907.3 seed the finite local-boundedness route for Real-facing and completion-facing function consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyBoundedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host function spaces, selected global bounds, quotient equality, countable choice, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and finite-window bound route for BEDC.Derived.LocallyBoundedFunctionUp.

Chapter 8908

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ packet records the finite functional that reads regular Cauchy tail data through displayed windows and dyadic tolerance rows before a Real-facing consumer reaches a seal. It sits over the regular rational source of chapter 301, the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the finite choice boundary of chapter 1354. The packet names a tail-functional read only; it does not assert completeness, quotient names, selected limits, countable choice, or a global choice of tails.

Definition 8908.1 (Regular Cauchy tail-functional carrier)**.** *A* $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational tail is inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which tail requests are exposed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the inspected window.*
4. *T is the tail-functional row, recording the finite read from a tolerance request to a displayed tail window.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the analytic rows and the structural rows.*
7. *C records displayed* Cont *replay from regular rational source data through the window, tolerance ledger, tail functional, seal, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CanonicalTailChoice}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, a selected limit, a host sequence equality test, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8908.2 (Regular Cauchy tail-functional NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ *carrier*

$$F = (S, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular rational source, finite stream window, dyadic tolerance ledger, tail-functional row, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8908.1. Its rows list the regular rational source S, stream window W, dyadic tolerance ledger D, tail-functional row T, seal row R, and structural support H, C, P, N.

The tail read is ordered by the carrier. A consumer must expose S, W, D before the tail-functional row can be read, and the seal row R is downstream from that finite tail read.

The structural rows do not add analytic content. Transport, replay, provenance, and local naming keep the same packet visible, so the listed rows exhaust the local NameCert surface. □

Theorem 8908.3 (Regular Cauchy tail-functional window stability)**.** *Let* $F = (S, W, D, T, R, H, C, P, N)$ *and* $F' = (S', W', D', T', R', H', C', P', N')$ *be accepted* $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ *carriers. If the source, window, and dyadic rows are transported componentwise by the displayed* hsame *data in* H *and* H'*, then the tail-functional rows* T *and* T' *remain on the same local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}$ *surface before either packet reaches its* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Read the transported source rows first. Componentwise hsame identifies the regular rational sources and their finite stream windows before the tail-functional row is consumed.

The dyadic tolerance ledgers are transported along the same displayed route. Hence the tail-functional rows compare finite requests inside the same window-and-tolerance surface rather than through a host equality test.

The seal rows and structural rows are downstream. Since R, H, C, P, N and R', H', C', P', N' only seal, transport, replay, source, and name the transported packet, window stability is exactly stability of the displayed finite rows. □

Theorem 8908.4 (Regular Cauchy tail-functional Real-seal nonescape)**.** *No accepted* $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ *packet exports quotient Cauchy names, selected limits, host sequence equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. A Real-facing consumer reaches the seal row* R *only through the displayed route*

$$S, W, D, T, R, H, C, P, N.$$

Proof. Inspect the carrier. The only analytic rows are the regular rational source, finite window, dyadic tolerance ledger, tail-functional row, and Real seal.

Apply the NameCert obligations of Definition 8908.2. They force the consumer through S, W, D, T before R is available.

The excluded principles are not coordinates of the packet. The transport, replay, provenance, and local naming rows preserve the finite route, so any read of a quotient name, selected limit, host equality, choice row, trichotomy row, or completeness principle leaves the accepted surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ consumer displays the regular rational source, stream window, dyadic tolerance ledger, tail-functional row, Real seal, transport, replay, provenance, and local NameCert rows before reading a Real-facing tail result.

Independence witness: regseqrat, streamname, dyadicratcore, real, canonicaltailchoice: no listed sibling is bijective to $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ because this packet jointly binds a regular rational source, finite stream window, dyadic tolerance ledger, tail-functional read, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailFunctional}^{\uparrow}$ packet is generated from a regular rational source, a finite stream window, a dyadic tolerance ledger, a tail-functional row, a Real seal, componentwise transport, displayed continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8908.1, Definition 8908.2, Definition 8908.3, and Definition 8908.4 seed the finite tail-functional packet over regular rational source rows, stream windows, dyadic tolerances, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy names, selected limits, host sequence equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-functional exactness, window stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailFunctionalUp.

Chapter 8909

A Concrete Naming Certificate for $\mathsf{FiniteMembershipClassifier}^{\uparrow}$

The $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ packet realizes the order–set–type interface of chapter 9356 as finite membership-classifier data over displayed source, member, witness, stability, ledger, transport, replay, provenance, and local naming rows. It sits beside the finite interface spine of chapter 8899, the public finite-data rows of chapter 36, chapter 33, and chapter 32, and the set/type interface intent of $\mathsf{SetLike}^{\uparrow}$ and $\mathsf{TypeLike}^{\uparrow}$. The packet names finite membership inspection only; it does not introduce ambient subset semantics, a host finite-type universe, quotient equality, unrestricted comprehension, or excluded-middle membership.

Definition 8909.1 (Finite membership classifier carrier)**.** *A* $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, A, F, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row naming the finite displayed collection being inspected.*
2. *A is the member row naming the candidate displayed element.*
3. *F is the finite witness row, exposing the bounded inspection window and the finite evidence considered by the classifier.*
4. *Q is the classifier row, recording the membership read as finite certificate data rather than as ambient elementhood.*
5. *L is the stability and ledger row, recording source stability, member stability, witness stability, and the consumed comparison evidence.*
6. *H records componentwise* hsame *transport for the source, member, witness, classifier, ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a membership request through S, A, F, Q, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteMembershipClassifier}^{\uparrow}$, $\mathsf{FiniteInterfaceSpine}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{TypeLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMembershipClassifier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for ambient subset semantics, an unrestricted type universe, quotient equality of members, hidden comprehension, or excluded-middle membership.

Theorem 8909.2 (Finite membership classifier NameCert obligations)**.** *For every accepted* FiniteMembershipClassif *carrier*

$$M = (S, A, F, Q, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteMembershipClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the source row, member row, finite witness row, classifier row, stability and ledger row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8909.1. The source, member, witness, classifier, and ledger rows are the only membership-facing rows available to an accepted packet.

The finite witness row F bounds the inspection before Q records the classifier read. The ledger row L records which source, member, witness, and stability data have been consumed.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no ambient subset, host type universe, quotient member equality, hidden comprehension, or excluded-middle row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8909.3 (Finite membership classifier stability)**.** *Let* $M = (S, A, F, Q, L, H, C, P, N)$ *and* $M' = (S', A', F', Q', L', H', C', P', N')$ *be accepted* $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ *carriers. If the source, member, finite witness, and ledger rows are transported componentwise by the displayed* hsame *data in* H *and* H'*, then the classifier rows* Q *and* Q' *remain on the same finite membership surface. The transported read exports no ambient subset relation, quotient member equality, hidden comprehension, or excluded-middle membership principle.*

Proof. Start with the source and member rows. Componentwise hsame transport identifies the displayed collection and candidate element before the classifier row is read.

Transport the finite witness rows next. The classifier rows are read after the same bounded evidence is displayed, and the ledger rows record the same consumed comparison data.

The replay, provenance, and local naming rows keep the transported packet inside the same surface. Therefore classifier stability is a property of the displayed finite rows, not an ambient membership principle. □

Theorem 8909.4 (Finite membership classifier interface handoff)**.** *Every order–set–type interface consumer that reads a* $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ *packet must consume the route*

$$S, A, F, Q, L, H, C, P, N$$

before treating membership as public classifier data. The handoff realizes finite source, member, witness, stability, ledger, transport, replay, provenance, and NameCert rows, and supplies no host finite-type universe, unrestricted comprehension, quotient equality, or hidden decidability principle.

Proof. Apply the NameCert obligations of Definition 8909.2. They force the source, member, witness, classifier, and ledger rows to be displayed before the interface consumer can read membership.

The stability theorem Definition 8909.3 shows that transported reads remain on the same finite membership surface. Thus a consumer can move membership data across displayed sameness only by preserving the packet rows.

The carrier lacks coordinates for the excluded objects. Hence the order–set–type handoff is exactly the finite classifier route listed above. □

Falsifiable prediction: Every accepted $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ consumer displays source, member, finite witness, classifier, stability ledger, transport, replay, provenance, and local NameCert rows before reading membership as order–set–type interface data.

Independence witness: finite-interface-spine, list, option, bool, setlike, typelike: no listed sibling is bijective to $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ because this packet jointly binds finite source, member, witness, classifier, stability, ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMembershipClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMembershipClassifier}^{\uparrow}$ packet is generated from a displayed finite source, a displayed member, a finite witness row, a membership classifier row, a stability and ledger row, componentwise transport, displayed continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMembershipClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8909.1, Definition 8909.2, Definition 8909.3, and Definition 8909.4 seed the finite membership-classifier packet over source, member, witness, classifier, stability, ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMembershipClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient subset semantics, a host finite-type universe, quotient member equality, unrestricted comprehension, excluded-middle membership, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite witness exactness, classifier stability, interface handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMembershipClassifierUp.

Chapter 8910

A Concrete Naming Certificate for $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$

Following Caratheodory, Koebe, and Ahlfors on kernel convergence of simply connected planar domains, the $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ packet records the finite NameCert route by which pointed domain windows, compact-kernel exhaustions, holomorphic chart reads, normal-family compactness, and Riemann-map normalization are read as one carrier. It sits between the normal-family packet of chapter 2186, the Riemann mapping packet of chapter 6805, the holomorphic chart surface of chapter 68, the unit-disk source of chapter 377, and the compact-open topology surface of chapter 4138. The packet names the displayed kernel route without importing prime ends, quotient equality of functions, host domain convergence, or a selected boundary correspondence.

Definition 8910.1 (Caratheodory kernel convergence carrier)**.** *A* $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, E, H, N, R, T, C, P, L)$$

with the following displayed rows.

1. *D is the pointed planar-domain window, recording the visible base point, the inspected planar open rows, and the admitted source windows for the kernel read.*
2. *E is the compact-kernel exhaustion row, recording the finite compact windows whose eventual containment witnesses the kernel interior and whose escape rows witness points outside the kernel.*
3. *H is the holomorphic chart-read row, exposing the finite* $\mathsf{Holomorphic}^{\uparrow}$ *chart observations used to compare maps on the displayed compact windows.*
4. *N is the normal-family compactness handoff, read through* $\mathsf{NormalFamily}^{\uparrow}$ *and the compact-window discipline of* $\mathsf{CompactOpenTopology}^{\uparrow}$ *before any subsequential holomorphic limit is public.*
5. *R is the Riemann-map route, recording the normalized* $\mathsf{UnitDisk}^{\uparrow}$ *source, base-point normalization, derivative-orientation row, and* $\mathsf{RiemannMapping}^{\uparrow}$ *handoff used by simply connected kernel consumers.*
6. *T records componentwise* hsame *transport for the pointed window, compact-kernel exhaustion, holomorphic chart reads, normal-family handoff, Riemann-map route, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a kernel-convergence request through D, E, H, N, R, T.*

8. *P is the* Pkg *provenance over* $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$, $\mathsf{NormalFamily}^{\uparrow}$, $\mathsf{RiemannMapping}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryKernelConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for prime ends, quotient function equality, host domain convergence, selected boundary correspondence, ambient compactness beyond the displayed normal-family handoff, or an unrestricted domain topology.

Lean4 ✓ `BEDC.Derived.CaratheodoryKernelConvergenceUp.CaratheodoryKernelConvergenceTasteGate`
`single carrier alignment`

Theorem 8910.2 (Caratheodory kernel convergence NameCert obligations)**.** *For every accepted* $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ *carrier $K = (D, E, H, N, R, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryKernelConvergenc}}$ *surface consists exactly of carrier admission for the pointed planar-domain window, compact-kernel exhaustion visibility, holomorphic chart-read visibility, normal-family compactness handoff, Riemann-map route visibility, componentwise transport, continuation replay, provenance, and local naming. A kernel-convergence consumer must read these rows before using the packet as a Caratheodory kernel route.*

Proof. Project the carrier of Definition 8910.1. The pointed window D, compact exhaustion E, holomorphic chart row H, normal-family handoff N, and Riemann-map route R are visible coordinates of the same finite packet.

The compact-kernel row E supplies both interior containment windows and exterior escape rows. The holomorphic row H is read only on those displayed compact windows, so chart convergence is a compact-open read rather than host function equality.

The normal-family handoff N and Riemann-map route R remain downstream. They expose the compactness and normalized disk-map rows needed by consumers, but they do not add prime ends, boundary selectors, quotient equality of maps, or host domain convergence.

Finally T, C, P, L transport, replay, source, and name exactly these rows. Hence the listed rows exhaust the local NameCert obligations for the kernel packet. □

Theorem 8910.3 (Caratheodory kernel convergence normal-family handoff)**.** *Every accepted* CaratheodoryKernelCo *read that reaches a Riemann-map consumer first exposes the pointed planar-domain window D, compact-kernel exhaustion E, holomorphic chart reads H, and the* $\mathsf{NormalFamily}^{\uparrow}$ *compactness handoff N through* $\mathsf{CompactOpenTopology}^{\uparrow}$*. Only then may the normalized* $\mathsf{UnitDisk}^{\uparrow}$ *source and* $\mathsf{RiemannMapping}^{\uparrow}$ *route R be consumed. The handoff supplies no prime-end compactification, quotient function equality, host domain convergence, selected boundary correspondence, or ambient compactness principle.*

Proof. Start with the NameCert obligations of Definition 8910.2. They force every accepted kernel read to expose the pointed domain window, compact exhaustion, holomorphic chart reads, normal-family handoff, and Riemann-map route as rows of one packet.

The normal-family row is read before the Riemann-map route. It consumes the compact-open windows from chapter 4138 and the holomorphic chart reads from chapter 68, then hands a compactness boundary to chapter 2186.

The Riemann-map row is therefore downstream of the same finite handoff. Its normalized $\mathsf{UnitDisk}^{\uparrow}$ source and $\mathsf{RiemannMapping}^{\uparrow}$ data may be used only after the compact-kernel and normal-family rows are visible.

Since the carrier contains no prime-end coordinate, quotient equality of functions, host domain convergence row, boundary selector, or ambient compactness principle, the handoff is exactly the displayed normal-family route into Riemann mapping. □

Falsifiable prediction: Every accepted $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ read displays one pointed planar-domain window, one compact-kernel exhaustion, one holomorphic chart-read row, one NormalFamily compactness handoff, one RiemannMapping route, hsame transport, Cont replay, Pkg provenance, and local NameCert row before a kernel-convergence or Riemann-map consumer may use it.

Independence witness: normalfamily, riemannmapping, holomorphic, unitdisk, compactopen-topology: none is bijective with $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ because the packet jointly stores the pointed domain window, compact-kernel exhaustion, holomorphic chart reads, normal-family handoff, Riemann-map route, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CaratheodoryKernelConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CaratheodoryKernelConvergence}^{\uparrow}$ packet is generated from a pointed planar-domain window, compact-kernel exhaustion, holomorphic chart reads, NormalFamily compactness handoff, RiemannMapping route, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CaratheodoryKernelConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8910.1, Definition 8910.2, and Definition 8910.3 seed the finite kernel-convergence route from compact-open holomorphic windows through normal-family compactness into normalized Riemann mapping.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CaratheodoryKernelConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, prime-end compactification, quotient equality of functions, host domain convergence, selected boundary correspondences, or ambient compactness beyond the displayed normal-family handoff.

Upgrade path: Obligation closure requires checked carrier admission, compact-kernel exhaustion, holomorphic chart-read visibility, normal-family handoff, Riemann-map route visibility, transport, replay, provenance, and local naming for BEDC.Derived.CaratheodoryKernelConvergenceUp.

Chapter 8911

A Concrete Naming Certificate for $\mathsf{LanczosIteration}^{\uparrow}$

Following Lanczos, the $\mathsf{LanczosIteration}^{\uparrow}$ packet records the finite three-term recurrence data used to build a Krylov basis and a tridiagonal projection from one symmetric matrix row, one inner-product row, and one seed-vector row. It sits after the Krylov carrier of chapter 8810, the matrix carrier of chapter 49, the inner-product carrier of chapter 86, and the spectral theorem surface of chapter 214. The packet names only the finite iteration certificate; it does not assert floating-point stability, Ritz-value convergence, host diagonalization, or a global equality between the tridiagonal row and the original matrix.

Definition 8911.1 (LanczosIteration carrier)**.** *A* $\mathsf{LanczosIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, I, q_0, m, Q, A_{\mathrm{diag}}, B_{\mathrm{sub}}, T, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{KrylovSubspace}^{\uparrow}$ *row generated by one matrix action and one displayed seed vector.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *row for the symmetric action inspected by the iteration.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row used to normalize vectors and test orthogonality inside the displayed Krylov span.*
4. *q_0 is the initial normalized seed row, read as the first admitted vector of the Krylov packet.*
5. *m is the finite step bound.*
6. *$Q = (q_0, \ldots, q_m)$ is the displayed vector ledger.*
7. *$A_{\mathrm{diag}} = (\alpha_0, \ldots, \alpha_m)$ is the diagonal coefficient ledger.*
8. *$B_{\mathrm{sub}} = (\beta_1, \ldots, \beta_m)$ is the subdiagonal coefficient ledger.*
9. *T is the tridiagonal matrix row with diagonal A_{diag} and symmetric subdiagonal B_{sub}.*
10. *R is the finite residual ledger recording that each next vector is generated from the previous two vectors and the same matrix action before normalization.*
11. *H records componentwise* hsame *transport for all displayed rows.*
12. *C records* Cont *replay from the seed request through the recurrence, coefficient, residual, and tridiagonal ledgers.*

13. *P is the* Pkg *provenance over* LanczosIteration$^{\uparrow}$, KrylovSubspace$^{\uparrow}$, Mat$^{\uparrow}$, InnerProduct$^{\uparrow}$, SpectralTheorem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* NameCert$_{\text{LanczosIteration}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite Krylov limit, a numerical roundoff model, an eigenvalue convergence theorem, a host orthogonal projector, or a hidden spectral decomposition.

Theorem 8911.2 (LanczosIteration NameCert obligations)**.** *For every accepted* LanczosIteration$^{\uparrow}$ *carrier*

$$L = (K, M, I, q_0, m, Q, A_{\text{diag}}, B_{\text{sub}}, T, R, H, C, P, N),$$

the local NameCert$_{\text{LanczosIteration}^{\uparrow}}$ *surface is exhausted by six obligation rows:*

1. *carrier admission for the Krylov row K, matrix row M, inner-product row I, seed row q_0, step bound m, vector ledger Q, coefficient ledgers A_{diag} and B_{sub}, tridiagonal row T, residual row R, and structural rows H, C, P, N;*
2. *same-action recurrence, requiring every residual entry of R to use the same matrix row M and the previous two displayed vector rows of Q;*
3. *inner-product normalization, requiring each admitted vector and coefficient read to pass through the same* InnerProduct$^{\uparrow}$ *row I;*
4. *tridiagonal assembly, requiring T to be reconstructed only from A_{diag} and B_{sub};*
5. *Krylov compatibility, requiring every vector in Q and every residual in R to remain a displayed finite read of the Krylov packet K;*
6. *ledger exhaustion, requiring any spectral, quadrature, conjugate-gradient, or Ritz-value consumer to cite only the displayed rows before using the packet.*

No NameCert obligation for LanczosIteration$^{\uparrow}$ *is stored outside those six rows.*

Proof. Project the carrier of Definition 8911.1. The accepted packet exposes the Krylov, matrix, inner-product, seed, step, vector, coefficient, tridiagonal, residual, transport, replay, provenance, and local naming rows.

The recurrence obligation is read from R. Each residual row records a finite application of the same matrix row to the current vector, subtracts the two displayed coefficient contributions, and admits the next vector only through the same inner-product normalization surface.

The tridiagonal matrix is not an independent host object. It is assembled from A_{diag} and B_{sub}, so any read of T factors through the two finite coefficient ledgers.

Finally K, H, C, P, N provide the boundary checks. The Krylov row supplies finite iterate compatibility, the transport and replay rows preserve the displayed ledgers, and the provenance row names the spectral theorem only as downstream background rather than as a convergence claim. Therefore the six listed obligations exhaust the certificate surface. □

Theorem 8911.3 (LanczosIteration Krylov handoff)**.** *Every accepted* LanczosIteration$^{\uparrow}$ *carrier has a canonical handoff to the* KrylovSubspace$^{\uparrow}$ *packet obtained by forgetting the coefficient, tridiagonal, and residual rows and retaining the rows*

$$(K, H, C, P, N).$$

This handoff preserves the finite seed, matrix-action, span, transport, replay, provenance, and local naming reads exposed by chapter 8810; it does not add an Arnoldi basis, a spectral approximation theorem, or a host diagonalization claim.

Proof. Read K from the carrier of Definition 8911.1. By construction K is already the finite Krylov packet generated by the same seed and matrix action used by the recurrence.

Forgetting A_{diag}, B_{sub}, T, and R removes the Lanczos-specific coefficient and tridiagonal data. The rows H and C still transport and replay the remaining Krylov reads componentwise, while P and N keep the provenance and local naming boundary visible.

The handoff therefore exposes exactly the finite Krylov certificate needed by downstream consumers and no additional spectral or host-level assertion. □

Falsifiable prediction: Every accepted $\mathsf{LanczosIteration}^{\uparrow}$ read exposes one Krylov row, one Matrix row, one InnerProduct row, one normalized seed row, one finite step bound, one vector ledger, two coefficient ledgers, one tridiagonal row, one residual ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any spectral, quadrature, conjugate-gradient, or Ritz-value consumer is served.

Independence witness: krylovsubspace, matrix, innerproduct, spectraltheorem: none alone is bijective with $\mathsf{LanczosIteration}^{\uparrow}$ because this packet alone binds finite Krylov admission, three-term recurrence data, normalization, tridiagonal assembly, residual replay, transport, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LanczosIteration}^{\uparrow}}$.

Constructive story: A $\mathsf{LanczosIteration}^{\uparrow}$ packet is generated from a Krylov row, a Matrix row, an InnerProduct row, a normalized seed, a finite step bound, a vector ledger, two coefficient ledgers, a tridiagonal row, a residual ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LanczosIteration}^{\uparrow}}$, seed)

Scope closed: Definition 8911.1, Definition 8911.2, and Definition 8911.3 seed the finite Lanczos iteration certificate over Krylov, matrix, inner-product, tridiagonal, residual, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LanczosIteration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close floating-point stability, loss of orthogonality, Ritz-value convergence, eigenvalue error bounds, host diagonalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recurrence compatibility, inner-product normalization, tridiagonal assembly, Krylov handoff, and ledger exhaustion for BEDC.Derived.LanczosIterationUp.

Chapter 8912

A Concrete Naming Certificate for $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ packet records the regular-Cauchy completion theorem as finite BEDC naming data rather than as a quotient construction on streams. It reads dyadic tolerance arithmetic from chapter 291, finite observation windows from chapter 126, regular rational readback from chapter 301, the terminal real-name seal from chapter 38, and the extension-and-uniqueness route of chapter 2251. The theorem packet names a capstone handoff for regular completion consumers; it does not identify quotient streams, select limits, invoke countable choice, assert host real equality, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8912.1 (Bishop–Bridges regular completion theorem carrier)**.** *A* BishopBridgesRegularCompletionTheor *carrier is a finite* BHist *packet*

$$T = (D, W, R, A, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare regular Cauchy observations.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the inspected completion request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row visible through D and W.*
4. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the dyadic, stream, and regular-rational rows have been displayed.*
5. *U is the* $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ *handoff row, read as an extension-and-uniqueness surface over the same finite completion data.*
6. *H records componentwise* hsame *transport for D, W, R, A, U, C, P, N.*
7. *C records displayed* Cont *replay from a completion theorem request through the dyadic, stream, regular-rational, real-seal, and universal-property rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, A, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8912.2 (Bishop–Bridges regular completion theorem NameCert obligations)**.** *For every accepted* $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ *packet*

$$T = (D, W, R, A, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}}$ *obligation sketch has exactly five rows:*

1. *carrier:* $D, W, R, A, U, H, C, P, N$ *must be displayed before the theorem packet is public;*
2. *classifier: accepted packets are compared only by the dyadic tolerance, stream-window, regular-rational, real-seal, universal-property, transport, replay, provenance, and local naming rows;*
3. *stability: componentwise* hsame *transport in* H *and displayed* Cont *replay in* C *preserve the same completion route;*
4. *ledger:* P *records the theorem packet as sourced from the dyadic, stream, regular-rational, real, and regular-completion universal-property siblings;*
5. *public theorem: the exported theorem handoff is the finite route through those rows, not a quotient completion, a selected limit, a choice principle, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8912.1. The rows D, W, R, A, U expose the dyadic tolerance ledger, finite stream windows, regular rational readback, real-name seal, and universal-property handoff before any theorem-facing consumer can inspect the packet.

The classifier and stability rows are structural. A consumer can compare packets only by the displayed coordinates, while H and C transport and replay the same ordered route without adding quotient equality or a selected representative.

Finally read the provenance and local naming rows. They bind the packet to the sibling chapters named in the opening paragraph and export only the finite theorem handoff. Since no excluded coordinate occurs in the carrier, the five listed rows exhaust the local NameCert obligation sketch. □

Theorem 8912.3 (Bishop–Bridges regular completion theorem handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ *packet reads the route*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Co

as the finite regular-completion theorem surface. The handoff may be consumed by Real and Completion chapters only after the dyadic tolerance, finite observation, regular rational readback, real seal, extension-and-uniqueness, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Begin with the dyadic and stream rows. They determine the tolerance request and the finite windows that are visible to the theorem packet.

Next read the regular rational and real rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular readback, and the $\mathsf{Real}^{\uparrow}$ row is reached only as the terminal seal over the same displayed data.

The universal-property row is then consumed as a finite extension-and-uniqueness ledger, through chapter 2251. Transport, replay, provenance, and local naming preserve that route for downstream consumers. No step introduces quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ read exposes dyadic tolerance, finite StreamName windows, RegSeqRat readback, a RealUp seal, a regular-Cauchy universal-property row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data before the theorem handoff is public.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-completion-universal-property: none is bijective with $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ because the theorem packet binds all five rows as one completion-facing capstone surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, a RealUp seal, a regular-Cauchy universal-property row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8912.1, Definition 8912.2, and Definition 8912.3 seed the finite regular-completion theorem route over dyadic, stream, regular rational, real-seal, universal-property, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopBridgesRegularCompletionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier locality, transport stability, provenance non-escape, and theorem handoff for BEDC.Derived.BishopBridgesRegularCompletionTheoremUp.

Chapter 8913

A Concrete Naming Certificate for $\mathsf{RealCompletionCantorSieve}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Cantor nested intervals, the $\mathsf{RealCompletionCantorSieve}^{\uparrow}$ packet records a finite nested-window route into the open Real and Completion phase. It consumes the dyadic arithmetic source of chapter 291, the observation stream of chapter 126, the regular rational handoff of chapter 301, the terminal real-name seal of chapter 38, and the fast Cauchy control of chapter 323. The packet names one finite Cantor-style sieve for one displayed budget; it is not a proof of full $\mathsf{Real}^{\uparrow}$ completeness, a quotient-real equality theorem, or a choice principle for infinite interval selections.

Definition 8913.1 (Real completion Cantor sieve carrier)**.** *A* $\mathsf{RealCompletionCantorSieve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, B, O, Q, R, F, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, carrying the dyadic endpoints, midpoint operations, order checks, and radius comparisons used by the displayed windows.*
2. *W is the finite nested-window table*

$$I_0 \supseteq I_1 \supseteq \cdots \supseteq I_m,$$

   where each $I_i = [a_i, b_i]$ has dyadic endpoints and the inclusion proof is recorded as a finite row in D.
3. *B is the diameter descent budget. It records a finite schedule $\epsilon_0, \ldots, \epsilon_m$ and witnesses that the displayed diameter of I_i is bounded by ϵ_i.*
4. *O is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It lists only the stream entries inspected to decide the next nested window.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, translating the observed stream windows into regular rational approximation rows for the same displayed budget.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the terminal window and the regular rational handoff. It exposes a real-name boundary only after D, W, B, O, Q have been read.*
7. *F is the* $\mathsf{FastCauchy}^{\uparrow}$ *control row. It records the finite modulus entries that certify the terminal window is compatible with the requested fast Cauchy tolerance.*

8. *E is the excluded-principle ledger: no ambient completeness theorem, quotient-real equality, located trichotomy, countable choice selector, or infinite nested-window choice may be read from the carrier.*
9. *H records componentwise* hsame *transport for the dyadic arithmetic, nested-window table, descent budget, stream observations, regular-rational handoff, real seal, fast Cauchy row, excluded-principle ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the finite route*

$$D, W, B, O \longmapsto Q, F \longmapsto R, E.$$

11. *P is the* Pkg *provenance over* $\mathsf{RealCompletionCantorSieve}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionCantorSieve}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $D, W, B, O, Q, R, F, E, H, C, P, N$. It has no coordinate for an infinite interval selector, a completed quotient of Cauchy names, global real completeness, or an order-trichotomy oracle.

Theorem 8913.2 (RealCompletionCantorSieveUp NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionCantorSieve}^{\uparrow}$ *carrier*

$$S = (D, W, B, O, Q, R, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionCantorSieve}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed packet of Definition 8913.1;*
2. *dyadic endpoint exactness, requiring every endpoint, midpoint, comparison, and diameter computation to be read from D;*
3. *nested-window exactness, requiring W to expose only the finite chain $I_0, \ldots, I_m$ and its displayed inclusion witnesses;*
4. *descent-budget exactness, requiring B to bind each exposed diameter bound to the corresponding finite tolerance row;*
5. *observation and regularization compatibility, requiring O and Q to show that every stream observation used by the sieve has a matching* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row;*
6. *terminal real and fast Cauchy compatibility, requiring R and F to seal only the terminal finite window and its displayed tolerance;*
7. *excluded-principle nonescape through E, requiring consumers to refuse ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient-real equality, located trichotomy, countable choice, and infinite nested-window selection;*
8. *transport, replay, provenance, and local naming through H, C, P, N.*

No Cantor-sieve obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 8913.1. The projection exposes the dyadic arithmetic row, the finite nested-window table, the descent budget, the observation stream row, the

regular rational handoff, the terminal real seal, the fast Cauchy row, the refusal ledger, and the structural rows.

The dyadic and nested-window obligations are finite. Each endpoint and comparison is read from D, while W lists only the windows $I_0, \ldots, I_m$ and the inclusion witnesses between consecutive rows. Thus a consumer cannot replace the finite table by an infinite interval selector.

The descent budget is tied to the same table. The row B records one tolerance row for each displayed window, so every diameter claim is checked against a row already present in W and computed through D.

The observation handoff then passes through O and Q. Every inspected stream entry is named in O, and the corresponding regular rational approximation is stored in Q before the terminal real seal is visible.

The endpoint of the route is local. The rows R and F seal the final window and the requested fast Cauchy tolerance, while E blocks the host principles that are not part of the packet.

Finally H, C, P, N transport, replay, source, and name exactly the same rows. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 8913.3 (Finite Cantor sieve handoff)**.** *Every accepted* $\mathsf{RealCompletionCantorSieve}^{\uparrow}$ *real-name read factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{FastCauchy}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Concretely, the read first opens dyadic endpoint arithmetic and a finite nested-window table, then checks the diameter descent budget, then reads the inspected stream observations and regular rational handoff rows, then exposes the terminal real seal and fast Cauchy control through transport, replay, provenance, and local naming.

Proof. Start with a consumer of the terminal real row R. By carrier admission, R is downstream from D, W, B, O, Q, F, so it cannot be inspected independently of the finite sieve data.

The first part of the route is dyadic. The rows D and W expose the endpoint arithmetic and the nested finite table, while B supplies the diameter schedule attached to those same windows.

The second part is observational. The stream row O records the finite observations that selected the displayed windows, and Q repackages those observations as regular rational rows for the same budget.

The final part is the terminal seal. The fast Cauchy row F checks the requested tolerance, and H, C, P, N preserve, replay, source, and name the same route. Therefore the public read is the displayed finite handoff and not a global completeness or choice claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionCantorSieve}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionCantorSieve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8913.1, Definition 8913.2, and Definition 8913.3 seed the finite Cantor-style nested-window route through DyadicRatCore, StreamName, RegSeqRat, FastCauchy, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionCantorSieve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full $\mathsf{Real}^{\uparrow}$ completeness, quotient-real equality, located trichotomy, countable choice, infinite nested-window selection, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked dyadic endpoint exactness, nested-window exactness, descent-budget exactness, observation and regularization compatibility, terminal real and fast Cauchy compatibility, excluded-principle nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionCantorSieveUp.

Constructive story: A $\mathsf{RealCompletionCantorSieve}^{\uparrow}$ packet is generated from DyadicRatCore endpoint arithmetic, a finite nested-window table, a diameter descent budget, StreamName observations, RegSeqRat handoff rows, a terminal RealUp seal, FastCauchy tolerance control, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8914

A Concrete Naming Certificate for $\mathsf{RapidCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RapidCauchySubsequence}^{\uparrow}$ packet records the finite extraction route that turns a displayed Cauchy or bounded-sequence request into a rate-controlled subsequence handoff without countable choice. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the diagonal selector route of chapter 1873. The packet supplies only a displayed subsequence extraction surface for Real and Completion consumers; it does not assert ambient sequential compactness, selected cluster points, quotient stream equality, countable choice, trichotomy, or mature $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8914.1 (Rapid Cauchy subsequence carrier)**.** *A* $\mathsf{RapidCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, R, I, W, D, G, E, A, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source sequence row, recording the visible Cauchy or bounded-sequence request whose subsequence is being extracted.*
2. *R is the rapid rate row, assigning each requested precision level to a finite tail window.*
3. *I is the strictly increasing index row for the selected subsequence positions.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row read at the indices listed by I.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic radii and slack comparisons used by the finite windows.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row induced by W and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rate, index, window, tolerance, and readback rows have been displayed.*
8. *A is the* $\mathsf{CauchyDiagonalArgument}^{\uparrow}$ *handoff row, recording that the selected finite tails are read through the diagonal route rather than through an ambient subsequence choice principle.*
9. *L records the rate-extraction handoff: every precision request reaches R, I, W, D, G, E, A in that order before a downstream Real or Completion consumer can inspect the subsequence.*
10. *H records componentwise* hsame *transport for the displayed rows.*

11. *C records displayed* Cont *replay of the finite extraction route.*
12. *P is the* Pkg *provenance over* RapidCauchySubsequence$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyDiagonalArgument$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{RapidCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, I, W, D, G, E, A, L, H, C, P, N$. It has no coordinate for countable choice, selected cluster points, quotient stream equality, host subsequence equality, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RapidCauchySubsequenceUp`

Theorem 8914.2 (Rapid Cauchy subsequence NameCert obligations)**.** *For every accepted* RapidCauchySubsequenc *carrier*

$$S = (Q, R, I, W, D, G, E, A, L, H, C, P, N),$$

the local NameCert$_{\text{RapidCauchySubsequence}^{\uparrow}}$ *surface consists of carrier exposure, classifier stability, monotone index display, finite-window selection, dyadic tolerance accounting,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *seal routing, diagonal handoff, rate-extraction handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside these finite rows.*

Proof. Project the carrier of Definition 8914.1. The analytic rows are the source sequence, rate schedule, increasing index list, StreamName windows, dyadic tolerance ledger, regular rational readback, Real seal, diagonal handoff, and rate-extraction handoff.

The monotone index and finite-window rows make the subsequence visible before any Real-facing seal is consumed. The dyadic ledger and RegSeqRat readback then bind the selected windows to the same tolerance route used by the L10 Real and Completion chain.

Finally replay the structural rows H, C, P, N. They transport, replay, source, and name the same finite packet, so the local NameCert obligations are exhausted by the displayed rows and do not import countable choice, quotient stream equality, host subsequence equality, trichotomy, or ambient completeness. □

Theorem 8914.3 (Rapid Cauchy subsequence rate extraction handoff)**.** *Every precision request accepted by a* RapidCauchySubsequence$^{\uparrow}$ *carrier factors through the finite route*

$$R, I, W, D, G, E, A, L, H, C, P, N.$$

The request first reads the rapid rate row R, then the increasing index row I, then the finite StreamName$^{\uparrow}$ *windows W, then the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, then the* RegSeqRat$^{\uparrow}$ *readback G, then the terminal* Real$^{\uparrow}$ *seal E, and only then the diagonal and structural handoff rows. The route supplies no selected infinite choice function, quotient stream equality, host subsequence equality, trichotomy, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier and inspect one precision request. The row R is the only rate entrance, so the request cannot bypass the finite schedule and read the terminal seal first.

The row I lists the selected indices and W reads only the finite windows at those indices. The dyadic ledger D supplies the tolerance radii and slack comparisons needed for those windows, and the readback row G turns the displayed windows into the regular rational surface consumed by L10 Real routes.

The seal E and diagonal row A are downstream of the same finite route. The handoff row L records this ordering for downstream Real and Completion consumers, while H, C, P, N preserve and name it. Since no carrier coordinate stores a choice function, quotient equality, host equality, trichotomy, or completeness theorem, none is exported by the handoff. □

Theorem 8914.4 (Rapid Cauchy subsequence nonchoice boundary)**.** *An accepted* $\mathsf{RapidCauchySubsequence}^{\uparrow}$ *packet may be consumed by Real and Completion routes only as the displayed finite subsequence extraction surface of Definition 8914.1. It cannot be used as a countable-choice selector, a cluster-point theorem, quotient stream equality, host subsequence equality, trichotomy, or mature* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the rate extraction handoff of Definition 8914.3. Every accepted read passes through the rate, index, window, dyadic tolerance, RegSeqRat readback, Real seal, diagonal handoff, and structural rows.

Inspect the classifier in Definition 8914.1. The packet has no coordinate for a chosen infinite selector outside I, no quotient equality row, no host subsequence equality row, no trichotomy row, and no ambient completeness theorem.

Thus a downstream consumer receives only the displayed finite extraction route. Any request for the excluded objects lies outside the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{RapidCauchySubsequence}^{\uparrow}$ read displays a source sequence row, rapid rate row, monotone index row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, RealUp seal, CauchyDiagonalArgument handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before any Real or Completion consumer receives subsequence data.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-diagonal-argument: $\mathsf{RapidCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source sequence display, rapid rate extraction, monotone subsequence indices, finite windows, dyadic tolerance accounting, regular readback, terminal Real sealing, diagonal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RapidCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RapidCauchySubsequence}^{\uparrow}$ packet is generated from a source sequence row, rapid rate row, monotone index row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal RealUp seal, CauchyDiagonalArgument handoff, rate-extraction handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RapidCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8914.1, Definition 8914.2, Definition 8914.3, and Definition 8914.4 seed the finite rapid Cauchy subsequence extraction route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RapidCauchySubsequence}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected cluster points, quotient stream equality, host subsequence equality, trichotomy, mature $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, monotone index preservation, window selection, dyadic accounting, RegSeqRat readback, Real-seal routing, diagonal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RapidCauchySubsequenceUp.

Chapter 8915

A Concrete Naming Certificate for $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$

Following Barendregt on standardization for lambda calculus, the $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$ packet records the finite route by which a residual-aware MetaCIC reduction read may be rearranged into a leftmost-facing route before any normalization or kernel-consistency conclusion is public. It sits after the lambda-calculus syntax carrier of chapter 200, the parallel-reduction packet of chapter 7468, the Church–Rosser packet of chapter 6198, and the candidate-normalization confluence handoff of chapter 5547. The packet names standardization only as displayed NameCert data; it does not assert full normalization, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency.

Definition 8915.1 (MetaCIC standardization theorem carrier)**.** *A* $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, R, P, C, L, J, H, Q, G, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{LambdaCalc}^{\uparrow}$ *term row supplying the source syntax and binder ledger.*
2. *R is the one-step reduction route whose redex observations are still displayed.*
3. *P is the* $\mathsf{ParallelReduction}^{\uparrow}$ *residual row recording the simultaneous-contraction and residual bookkeeping surface.*
4. *C is the* $\mathsf{ChurchRosser}^{\uparrow}$ *confluence-facing row, read only as a sibling join route.*
5. *L is the leftmost schedule row naming the standardization order for the displayed reduction observations.*
6. *J is the candidate-normalization handoff row read through* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$.
7. *H records componentwise* hsame *transport for the term, reduction, residual, confluence, leftmost schedule, handoff, replay, provenance, and naming rows.*
8. *Q records displayed* Cont *replay along the same rows.*
9. *G is the* Pkg *provenance over* $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{ParallelReduction}^{\uparrow}$, $\mathsf{ChurchRosser}^{\uparrow}$, $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICStandardizationTheorem}^{\uparrow}}$ *evidence row.*

The classifier compares accepted packets only through $T, R, P, C, L, J, H, Q, G, N$. It has no coordinate for global strong normalization, unrestricted Church–Rosser, subject reduction, quotient equality of terms, host evaluator equality, or kernel consistency.

Theorem 8915.2 (MetaCIC standardization NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICStandardizationT}$ *carrier* $S = (T, R, P, C, L, J, H, Q, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICStandardizationTheorem}^{\uparrow}}$ *surface consists exactly of term-row admission, displayed reduction-route visibility, parallel residual bookkeeping, sibling Church–Rosser route exposure, leftmost schedule admission, candidate-normalization handoff exposure, componentwise transport, displayed replay, provenance, and local naming. A MetaCIC confluence or normalization consumer may read the packet only as a standardization boundary between residual data and a leftmost route; it cannot read the packet as full normalization, unrestricted confluence, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency.*

Proof. Project the carrier of Definition 8915.1. The term row T, reduction route R, parallel residual row P, Church–Rosser sibling row C, leftmost schedule L, and candidate-normalization handoff J are the whole mathematical surface available to the certificate.

The standardization read first exposes T, R, P. This fixes the source syntax, the observed reduction route, and the residual bookkeeping before any leftmost schedule can be consumed.

Next it exposes C, L, J. The Church–Rosser row supplies only a sibling join route, the leftmost row names the standardization order, and the candidate-normalization handoff remains a guarded downstream boundary rather than a normalization theorem.

Finally H, Q, G, N preserve the same displayed route by transport, replay, provenance, and local naming. Since the carrier contains no coordinate for the excluded global endpoints, none of those endpoints can be exported by the local NameCert surface. □

Theorem 8915.3 (MetaCIC standardization leftmost route)**.** *Every accepted* $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$ *read that is handed to a MetaCIC confluence or candidate-normalization consumer must expose the finite route*

$$\mathsf{LambdaCalc}^{\uparrow} \longmapsto \mathsf{ParallelReduction}^{\uparrow} \longmapsto \mathsf{ChurchRosser}^{\uparrow} \longmapsto \mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$$

through the displayed rows $T, R, P, C, L, J, H, Q, G, N$ before any leftmost-facing endpoint is public. The leftmost route is a residual standardization handoff only; it does not claim global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or kernel consistency.

Proof. Start with the obligation surface of Definition 8915.2. It forces the term row, reduction route, parallel residual bookkeeping, Church–Rosser sibling row, leftmost schedule, and candidate-normalization handoff to be displayed together.

Read the sibling dependencies in order. The lambda-calculus row supplies syntax, $\mathsf{ParallelReduction}^{\uparrow}$ supplies residual bookkeeping, $\mathsf{ChurchRosser}^{\uparrow}$ supplies the confluence-facing sibling route, and $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ supplies the guarded downstream boundary.

The structural rows H, Q, G, N then transport, replay, source, and name the same finite path. Hence the public endpoint is only a residual standardization handoff to a leftmost route, with no coordinate for the refused global MetaCIC claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICStandardizationTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICStandardizationTheorem}^{\uparrow}$ packet is generated from lambda-term syntax, displayed reduction observations, parallel residual bookkeeping, a Church–Rosser sibling route, a leftmost schedule, candidate-normalization handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICStandardizationTheorem}\uparrow}$, seed)

Scope closed: Definition 8915.1, Definition 8915.2, and Definition 8915.3 seed the finite residual standardization route from lambda syntax and parallel residual data to a guarded MetaCIC candidate-normalization handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICStandardizationTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full normalization, unrestricted Church–Rosser, subject reduction, quotient syntax equality, host evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual bookkeeping, sibling Church–Rosser exposure, leftmost-schedule exactness, candidate-normalization handoff discipline, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICStandardizationTheoremUp.

Chapter 8916

A Concrete Naming Certificate for $\mathsf{ApartnessOrderedField}^{\uparrow}$

Following Bishop and Bridges on constructive algebra and ordered real fields, the $\mathsf{ApartnessOrderedField}^{\uparrow}$ packet records the finite interface in which field operations, strict apartness, and order-facing comparison data are displayed before any Real consumer may read an algebraic order claim. It sits over the apartness real carrier of chapter 304, the real apartness field surface of chapter 5562, the located apartness field of chapter 7144, the algebraic order route of chapter 766, the Archimedean ordered field boundary of chapter 1419, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names constructive ordered-field data only through displayed rows; it does not replace apartness by host inequality, trichotomy, quotient real equality, or ambient completeness.

Definition 8916.1 (Apartness ordered field carrier)**.** *An* $\mathsf{ApartnessOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, A, O, L, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the underlying* $\mathsf{Real}^{\uparrow}$ *name row whose operations are being read.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *row naming positive apartness as data rather than as negated equality.*
3. *O is the order-facing row supplied through* $\mathsf{RealAlgOrder}^{\uparrow}$ *and* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$*.*
4. *L is the located-field row inherited from* $\mathsf{LocatedApartnessField}^{\uparrow}$*.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational operation and comparison witnesses.*
6. *E is the Real-seal row reached only after R, A, O, L, Q remain visible.*
7. *H records componentwise* hsame *transport for the real, apartness, order, locatedness, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an ordered-field consumer request through apartness, order, locatedness, rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ApartnessOrderedField}^{\uparrow}$*,* $\mathsf{ApartnessReal}^{\uparrow}$*,* $\mathsf{RealApartnessField}^{\uparrow}$*,* $\mathsf{LocatedApartnessField}$ $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessOrderedField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, O, L, Q, E, H, C, P, N$. It has no coordinate for trichotomy, host real equality, quotient real equality, choice-selected representatives, or an ambient completeness principle.

Lean4 (statement) `BEDC.Derived.ApartnessOrderedFieldUp`

Theorem 8916.2 (Apartness ordered field NameCert obligations)**.** *For every accepted* $\mathsf{ApartnessOrderedField}^{\uparrow}$ *carrier* $F = (R, A, O, L, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ApartnessOrderedField}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *source row, explicit* $\mathsf{ApartnessReal}^{\uparrow}$ *apartness, algebraic order data, located apartness-field data,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Real sealing, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer therefore reads constructive ordered-field structure only through the finite route*

$$R, A, O, L, Q \longmapsto E, H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8916.1. The rows R, A, O, L, Q display the real name, apartness relation, algebraic order, located-field evidence, and regular rational readback before the seal row is available.

The structural rows H, C, P, N preserve that displayed route under transport, replay, provenance, and local naming. Since the carrier has no row for trichotomy, host equality, quotient equality, or selected representatives, those data cannot enter the NameCert surface. □

Theorem 8916.3 (Apartness ordered field Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing algebraic or order-facing consumer of an accepted* $\mathsf{ApartnessOrderedField}^{\uparrow}$ *carrier must first display the* $\mathsf{Real}^{\uparrow}$ *source row R, the apartness row A, the order row O, the locatedness row L, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q. The handoff may consume the seal row E only as this displayed constructive ordered-field route. It does not assert trichotomy, host real equality, quotient real equality, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the NameCert obligations of Definition 8916.2. They force every ordered-field read through the displayed rows R, A, O, L, Q before E can be consumed.

The sibling chapters named in the carrier supply the traditional components as finite BEDC rows: apartness from chapter 304, located field data from chapter 7144, algebraic order from chapter 766, Archimedean order from chapter 1419, and rational readback from chapter 301.

Finally H, C, P, N preserve that same route. Because the carrier exports no trichotomy row, quotient equality row, selected representative, or ambient completeness row, the Real handoff remains constructive and apartness-explicit. □

Falsifiable prediction: An accepted $\mathsf{ApartnessOrderedField}^{\uparrow}$ read that produces an ordered-field conclusion while bypassing explicit apartness, locatedness, regular rational readback, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: apartnessreal, real-apartness-field, located-apartness-field, real-alg-order, archimedeanorderedfield, regseqrat, real: no listed sibling is bijective with $\mathsf{ApartnessOrderedField}^{\uparrow}$ because this packet binds apartness, algebraic order, located field data, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one constructive ordered-field interface.

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessOrderedField}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ApartnessOrderedField}^{\uparrow}$ packet is generated from RealUp source data, explicit apartness, algebraic order, located apartness-field data, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ApartnessOrderedField}\uparrow}$, seed)

Scope closed: Definition 8916.1, Definition 8916.2, and Definition 8916.3 seed the constructive ordered-field and apartness handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{ApartnessOrderedField}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, host real equality, quotient real equality, selected representatives, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, explicit apartness preservation, order-row exactness, located-field stability, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ApartnessOrderedFieldUp.

Chapter 8917

A Concrete Naming Certificate for $\mathsf{RealCompletionTailLock}^{\uparrow}$

The $\mathsf{RealCompletionTailLock}^{\uparrow}$ packet realizes the Real and Completion branch of chapter 9238 as a finite tail-lock object: a completion-facing read may inspect a finite stream window, regular rational readback, dyadic tolerance ledger, and terminal Real seal only in that order. It sits over the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a lock on the visible tail route; it does not assert quotient Cauchy equality, choose a completed limit, import trichotomy, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8917.1 (Real completion tail-lock carrier)**.** *A* $\mathsf{RealCompletionTailLock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected tail requests.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row locking the inspected radii and endpoint budgets.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, D have been displayed.*
5. *L is the tail-lock row recording that completion-facing reads preserve the same S, R, D, E route.*
6. *H records componentwise* hsame *transport for the stream, readback, dyadic, seal, lock, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through the stream window, regular readback, dyadic tolerance, Real seal, and tail lock.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionTailLock}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionTailLock}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, E, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient Real completeness.

Lean4 (statement) `BEDC.Derived.RealCompletionTailLockUp`

Theorem 8917.2 (Real completion tail-lock NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionTailLock}^{\uparrow}$ *carrier* $T = (S, R, D, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionTailLock}^{\uparrow}}$ *surface consists exactly of finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance locking,* $\mathsf{Real}^{\uparrow}$ *sealing, tail-lock preservation, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion-facing consumer therefore reads the route only through*

$$S, R, D \longmapsto E, L, H, C, P, N.$$

Proof. Project the accepted carrier of Definition 8917.1. The stream, regular-rational, and dyadic rows S, R, D are displayed before the terminal seal E and before the lock row L can be consumed.

The rows H, C, P, N preserve exactly that finite route by transport, replay, provenance, and local naming. Since no quotient stream, selected limit, countable-choice schedule, trichotomy row, or ambient completeness row appears in the carrier, the NameCert obligations are exhausted by the displayed tail-lock surface. □

Theorem 8917.3 (Real completion tail-lock L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{RealCompletionTailLock}^{\uparrow}$ *carrier must read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before consuming the* $\mathsf{Real}^{\uparrow}$ *seal row* E *or the lock row* L*. The handoff may transport and replay this finite route, but it cannot replace it by quotient equality of Cauchy names, a selected completed limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 8917.2. They place the stream window, regular-rational readback, and dyadic tolerance rows before the Real seal and tail-lock rows.

The vision source chapter 9238 asks for completion-facing reads to remain explicit about their finite regular-Cauchy route. This carrier supplies exactly that finite route through the sibling chapters chapter 126, chapter 301, chapter 291, and chapter 38.

Finally H, C, P, N preserve the same displayed route. Since the accepted carrier contains no quotient, selected-limit, choice, trichotomy, or ambient-completeness coordinate, the L10 handoff remains a finite tail-lock handoff. □

Falsifiable prediction: An accepted $\mathsf{RealCompletionTailLock}^{\uparrow}$ read that reaches Real or Completion data without displaying StreamName, RegSeqRat, DyadicRatCore, Real seal, tail-lock, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{RealCompletionTailLock}^{\uparrow}$ because this packet binds finite windows, regular rational readback, dyadic tolerance locking, Real sealing, tail-lock preservation, transport, replay, provenance, and local naming as one completion-facing route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionTailLock}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCompletionTailLock}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, tail-lock preservation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionTailLock}^{\uparrow}}$, seed)

Scope closed: Definition 8917.1, Definition 8917.2, and Definition 8917.3 seed the finite L10 Real and Completion tail-lock route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionTailLock}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy equality, selected completed limits, countable choice, trichotomy, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName exposure, RegSeqRat readback, DyadicRatCore tolerance locking, Real-seal nonescape, tail-lock exactness, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionTailLockUp.

Chapter 8918

A Concrete Naming Certificate for $\mathsf{RealCompletionIdempotence}^{\uparrow}$

Following Bishop and Bridges on constructive completions, the $\mathsf{RealCompletionIdempotence}^{\uparrow}$ packet records that a $\mathsf{Real}^{\uparrow}$ seal already obtained from $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data is stable under re-entering the same completion route. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names a face-local Real and Completion boundary; it does not construct quotient streams, choose representatives, select limits, invoke countable choice, or assert ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 8918.1 (Real completion idempotence carrier)**.** *A* $\mathsf{RealCompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, S, Q, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row already obtained from the first completion-facing route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the same finite observations used to read the name.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row governing the displayed regularity comparisons.*
5. *H records componentwise* hsame *transport for the Real seal, windows, readback, tolerance ledger, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from any second completion request back through D, Q, S and into the already sealed R row.*
7. *P is the* Pkg *provenance over* $\mathsf{RealCompletionIdempotence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionIdempotence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through R, S, Q, D, H, C, P, N. No coordinate stores a quotient stream equality, representative selector, selected limit, countable-choice schedule, metric-completeness theorem, or ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Theorem 8918.2 (Real completion idempotence NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionIdempotence}^{\uparrow}$ *carrier*

$$I = (R, S, Q, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionIdempotence}^{\uparrow}}$ *surface consists exactly of carrier admission, sealed* $\mathsf{Real}^{\uparrow}$ *exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8918.1. Its visible data are the sealed Real row R, the window row S, the regular rational readback Q, the dyadic tolerance ledger D, and the structural rows H, C, P, N.

The second completion request is read by replaying the same finite route. It first exposes D, then Q, then S, and only then reaches the already sealed $\mathsf{Real}^{\uparrow}$ row R.

The remaining rows preserve the boundary. Componentwise hsame in H transports the displayed coordinates, Cont in C replays the same dependency order, and P, N record provenance and local naming. Since the carrier has no excluded coordinate, the NameCert surface exports no quotient stream equality, selected representative, selected limit, countable choice, metric completeness, or ambient Real completeness. □

Theorem 8918.3 (Real completion idempotence nonescape)**.** *Every second completion-facing read of an accepted* $\mathsf{RealCompletionIdempotence}^{\uparrow}$ *carrier reuses the same displayed face rows*

$$R, S, Q, D, H, C, P, N.$$

Consequently re-entering the completion route cannot add quotient stream equality, selected limits, countable choice, metric completeness, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Let a second completion-facing consumer read an accepted carrier $I = (R, S, Q, D, H, C, P, N)$. Carrier projection gives no data outside these eight rows.

The replay row C directs the request through the dyadic tolerance ledger D, the regular rational readback Q, and the finite observation windows S before the terminal seal row R is inspected. Thus the second read has the same face-local route as the first read.

Transport and provenance do not enlarge the route. The componentwise hsame row H compares only the displayed coordinates, while P and N record the package and local name for the same carrier. With no quotient, choice, selected-limit, metric-completeness, or ambient-completeness coordinate present, there is no escape path from the idempotence packet to any of those claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionIdempotence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionIdempotence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8918.1, Definition 8918.2, and Definition 8918.3 seed the face-local idempotence carrier over Real, StreamName, RegSeqRat, DyadicRatCore, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionIdempotence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, selected limits, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window reuse, RegSeqRat readback reuse, DyadicRatCore tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionIdempotenceUp.

Constructive story: A $\mathsf{RealCompletionIdempotence}^{\uparrow}$ packet is generated from an already sealed Real row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8919

A Concrete Naming Certificate for $\mathsf{DoobMeyerDecomposition}^{\uparrow}$

Following Doob and Meyer, the $\mathsf{DoobMeyerDecomposition}^{\uparrow}$ packet records the finite BEDC surface of a submartingale split into a martingale row and an increasing compensator row. It consumes the one-sided adapted carrier of chapter 6556, the equality-direction martingale rows of chapter 190, the singleton conditional-expectation surface of chapter 188, the bounded stopped-window route of chapter 383, and the crossing ledger of chapter 1629. The chapter names only finite martingale-compensator rows and their local certificate obligations; it does not import measure completion, optional projection, predictable host σ-algebras, almost-sure quotient equality, or an ambient stochastic-process theorem.

Definition 8919.1 (Doob-Meyer decomposition carrier)**.** *A* $\mathsf{DoobMeyerDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, M, A, E, O, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{Submartingale}^{\uparrow}$ *source row whose finite one-sided conditional-expectation comparisons are to be split.*
2. *M is the* $\mathsf{Martingale}^{\uparrow}$ *row carrying the equality-direction adapted sequence read from the same displayed finite window.*
3. *A is the compensator ledger, a finite increasing row whose successor gaps record the displayed difference between the* $\mathsf{Submartingale}^{\uparrow}$ *inequality and the* $\mathsf{Martingale}^{\uparrow}$ *equality row.*
4. *E is the family of* $\mathsf{CondExp}^{\uparrow}$ *faces that reads the successor endpoint before the compensator step row is exposed.*
5. *O is the bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *window used to compare stopped reads of S, M, and A.*
6. *U is the* $\mathsf{Upcrossing}^{\uparrow}$ *ledger that keeps the same finite lower-to-upper crossing window visible to decomposition consumers.*
7. *H records componentwise* hsame *transport for the submartingale, martingale, conditional-expectation, stopped-window, upcrossing, and compensator rows.*
8. *C records displayed* Cont *replay from a decomposition request through S, M, A, E, O, U.*
9. *P is the* Pkg *provenance over* $\mathsf{Submartingale}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{DoobMeyerDecomposition}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these ten rows. It has no coordinate for measure completion, optional projection, a predictable host σ-algebra, selected almost-sure versions, quotient equality of stochastic processes, or an external decomposition theorem.

Theorem 8919.2 (Doob-Meyer decomposition NameCert obligations)**.** *For every accepted* DoobMeyerDecompositi *carrier $D = (S, M, A, E, O, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DoobMeyerDecomposition}^{\uparrow}}$ *surface consists exactly of carrier admission, exposure of the source* $\mathsf{Submartingale}^{\uparrow}$ *row, exposure of the* $\mathsf{Martingale}^{\uparrow}$ *row, compensator-ledger exactness for A, conditional-expectation exposure through E, bounded stopped-window routing through O, finite upcrossing control through U, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Every decomposition consumer must read

$$\mathsf{Submartingale}^{\uparrow},\ \mathsf{Martingale}^{\uparrow},\ \mathsf{CondExp}^{\uparrow},\ \mathsf{OptionalStopping}^{\uparrow},\ \mathsf{Upcrossing}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

through the displayed packet before a martingale-compensator conclusion is visible. The obligation surface is finite NameCert data; it is not measure completion, optional projection, predictable host measurability, almost-sure quotient equality, or an ambient Doob–Meyer theorem.

Proof. Project the accepted carrier of Definition 8919.1. The analytic rows are S, M, A, E, O, U, and the structural rows are H, C, P, N.

The source row S supplies the one-sided $\mathsf{Submartingale}^{\uparrow}$ inequality, while M supplies the equality-direction $\mathsf{Martingale}^{\uparrow}$ read on the same finite window. The conditional-expectation faces E expose the successor comparisons before any compensator step row in A can be inspected.

The bounded stopping and upcrossing rows O, U keep the finite window stable for downstream consumers. Transport, replay, provenance, and local naming preserve the same route, so no consumer can bypass the displayed packet to obtain a host optional projection or a completed ambient decomposition theorem. □

Theorem 8919.3 (Doob-Meyer compensator handoff)**.** *Every compensator read exported by an accepted* $\mathsf{DoobMeyerDecomposition}^{\uparrow}$ *packet factors through the route*

$$S, E \longmapsto M, A \longmapsto O, U, H, C, P, N.$$

The row A is visible only as the finite ledger that records the gap between the $\mathsf{Submartingale}^{\uparrow}$ *source comparisons and the* $\mathsf{Martingale}^{\uparrow}$ *equality-direction row. A downstream martingale or upcrossing consumer receives that ledger together with the bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *and* $\mathsf{Upcrossing}^{\uparrow}$ *windows; it does not receive a host predictable projection, measure completion, selected almost-sure version, quotient process equality, or external compensator existence proof.*

Proof. Begin with the NameCert obligations of Definition 8919.2. They require the submartingale source, martingale row, conditional-expectation faces, compensator ledger, stopped-window row, and upcrossing row to be displayed before the local certificate can be consumed.

The compensator row is therefore not an independent host process. It is read only after the $\mathsf{CondExp}^{\uparrow}$ faces expose the one-step gap between the source $\mathsf{Submartingale}^{\uparrow}$ inequality and the $\mathsf{Martingale}^{\uparrow}$ equality row.

Finally, bounded stopping and upcrossing preserve the same finite window. Since A is carried only with O, U, H, C, P, N, every handoff remains inside the finite packet and cannot be reread as optional projection, predictable host measurability, measure completion, quotient equality, or an ambient decomposition theorem. □

Falsifiable prediction: An accepted $\mathsf{DoobMeyerDecomposition}^{\uparrow}$ consumer that reaches a compensator row without first reading the displayed $\mathsf{Submartingale}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows refutes the carrier surface.

Independence witness: submartingale, martingale, condexp, optionalstopping, upcrossing: no listed sibling supplies the whole packet $D = (S, M, A, E, O, U, H, C, P, N)$, because the source one-sided row, equality-direction martingale row, compensator ledger, stopped-window route, crossing ledger, transport, replay, provenance, and local naming are jointly bound only here.

Closure status for $\mathsf{NameCert}_{\mathsf{DoobMeyerDecomposition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoobMeyerDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8919.1, Definition 8919.2, and Definition 8919.3 seed the finite martingale-compensator certificate over submartingale, martingale, conditional-expectation, bounded stopping, upcrossing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoobMeyerDecomposition}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close measure completion, optional projection, predictable host sigma-algebras, selected almost-sure versions, quotient equality of stochastic processes, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source submartingale exposure, martingale row exposure, compensator-ledger exactness, conditional-expectation exposure, stopped-window routing, finite upcrossing control, transport, replay, provenance, and local naming for BEDC.Derived.DoobMeyerDecompositionUp.

Constructive story: A $\mathsf{DoobMeyerDecomposition}^{\uparrow}$ packet is generated from a SubmartingaleUp source row, MartingaleUp equality row, finite compensator ledger, CondExpUp faces, OptionalStoppingUp bounded window, UpcrossingUp ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8920

A Concrete Naming Certificate for $\mathsf{DomainCompletion}^{\uparrow}$

Following Scott and the domain-theoretic completion tradition, the $\mathsf{DomainCompletion}^{\uparrow}$ packet records the finite frontier object that turns rounded-domain approximation data into a completion-facing Real handoff. It sits after the interval-domain route of chapter 1543, the Scott-domain basis of chapter 4164, the rounded-ideal completion surface of chapter 1883, the finite stream observations of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite domain-completion route; it does not choose ideal representatives, quotient directed systems, selected limits, host order equality, countable choice, or ambient completeness.

Definition 8920.1 (Domain-completion carrier)**.** *A* $\mathsf{DomainCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, S, R, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{IntervalDomain}^{\uparrow}$ *cell row naming the located interval approximants read by the completion frontier.*
2. *S is the* $\mathsf{ScottDomain}^{\uparrow}$ *basis row that records the inspected finite way-below comparisons.*
3. *R is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *row that packages the rounded directed approximation discipline.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which completion-facing consumers observe the approximants.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after I, S, R, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for interval cells, Scott-domain basis data, rounded ideals, stream windows, regular readbacks, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion request through $I, S, R, W, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DomainCompletion}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DomainCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, R, W, Q, E, H, C, P, N$. It has no coordinate for selected ideal representatives, quotient directed-system equality, host order equality, countable choice, or ambient completeness.

Theorem 8920.2 (Domain-completion NameCert obligations)**.** *For every accepted* $\mathsf{DomainCompletion}^{\uparrow}$ *carrier*

$$D = (I, S, R, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DomainCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for interval-domain cells, Scott-domain basis comparisons, rounded-ideal completion data, finite StreamName windows, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. Completion-facing reads must expose the ordered route*

$$I \to S \to R \to W \to Q \to E$$

before any Real handoff is available, and the structural rows H, C, P, N may only transport, replay, source, and name that route.

Proof. Project the carrier of Definition 8920.1. The displayed analytic rows are I, S, R, W, Q, E, and the structural rows are H, C, P, N.

Read the domain side first. The interval-domain row I supplies the located cells, the Scott-domain row S supplies the inspected finite basis comparisons, and the rounded-ideal row R records the directed approximation discipline before any completion-facing stream row is available.

The observation and Real handoff then pass through W, Q, E. The finite window row W exposes the observations admitted by the rounded route, the regular rational row Q supplies the readback, and the Real seal row E is downstream of those displayed rows.

Finally H, C, P, N preserve the same packet boundary. Since no carrier coordinate stores selected ideal representatives, quotient directed systems, host order equality, countable choice, or ambient completeness, the local NameCert obligations are exhausted by the displayed route. □

Theorem 8920.3 (Domain-completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{DomainCompletion}^{\uparrow}$ *carrier must read the finite completion frontier*

$$\mathsf{IntervalDomain}^{\uparrow} \to \mathsf{ScottDomain}^{\uparrow} \to \mathsf{RoundedIdealCompletion}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The handoff exports located interval approximation, finite Scott-basis comparison, rounded directed approximation, finite observation windows, regular rational readback, Real sealing, hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. It exports no selected ideal representative, quotient directed-system equality, host order equality, countable-choice selector, or ambient completion theorem.*

Proof. Start with an accepted carrier $D = (I, S, R, W, Q, E, H, C, P, N)$. Carrier projection gives the consumer access only to those displayed rows.

The completion frontier begins with the domain rows. A consumer reads located interval approximants through I, finite Scott-basis comparisons through S, and rounded directed approximation through R before reaching any stream or Real-facing data.

The handoff then consumes W, Q, E in order. The finite observation windows in W feed the regular rational readback Q, and the terminal Real seal E is exposed only after that readback is displayed.

The structural rows add no new mathematical primitive. Componentwise transport in H, replay in C, provenance in P, and local naming in N carry the same finite route, so the excluded

representative, quotient, host-equality, choice, and ambient-completeness data have no row from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{DomainCompletion}^{\uparrow}$ Real or Completion handoff displays IntervalDomain, ScottDomain, RoundedIdealCompletion, StreamName, RegSeqRat, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the terminal seal can be consumed.

Independence witness: intervaldomain, scott-domain, rounded-ideal-completion, streamname, regseqrat, real: $\mathsf{DomainCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds interval approximants, Scott-basis comparison, rounded directed approximation, finite observation windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one completion-frontier packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DomainCompletion}\uparrow}$.

Constructive story: A $\mathsf{DomainCompletion}^{\uparrow}$ packet is generated from IntervalDomain cells, ScottDomain basis data, RoundedIdealCompletion rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DomainCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8920.1, Definition 8920.2, and Definition 8920.3 seed the finite domain-completion frontier from interval and Scott-domain approximation to the Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DomainCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected ideal representatives, quotient directed-system equality, host order equality, countable choice, ambient completion, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-domain approximation, Scott-domain basis comparison, rounded-ideal discipline, StreamName observation, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DomainCompletionUp.

Chapter 8921

A Concrete Naming Certificate for $\mathsf{CauchyNameTruncation}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy names, the $\mathsf{CauchyNameTruncation}^{\uparrow}$ packet records the finite prefix and tail-budget certificate visible before a regular rational name is handed toward a Real or Completion consumer. It sits over the StreamName finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a displayed truncation budget for one regular Cauchy read; it is not quotient stream equality, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8921.1 (CauchyNameTruncation carrier)**.** *A* $\mathsf{CauchyNameTruncation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite prefix schedule naming the inspected source windows.*
2. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the displayed error radii.*
4. T *is the truncation-index and tail-budget row saying which prefix is sufficient for the named tolerance.*
5. E *is the optional* $\mathsf{Real}^{\uparrow}$ *seal row, available only after* S, Q, D, T *have been displayed.*
6. H *records componentwise* hsame *transport for prefix schedule, readback, tolerance, truncation budget, optional seal, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay of the same finite truncation route.*
8. P *is the* Pkg *provenance over* $\mathsf{CauchyNameTruncation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyNameTruncation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, T, E, H, C, P, N$*. It has no coordinate for quotient stream equality, selected limits, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 (statement) `BEDC.Derived.CauchyNameTruncationUp`

Theorem 8921.2 (CauchyNameTruncation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNameTruncation}^{\uparrow}$ *carrier* $K = (S, Q, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyNameTruncation}^{\uparrow}}$ *surface consists exactly of carrier admission, finite StreamName prefix exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, truncation-index and tail-budget admission, optional RealUp sealing, componentwise transport, displayed replay, provenance, and local naming. No accepted local read may export quotient stream equality, selected limits, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8921.1. The rows S, Q, D, T are the finite prefix schedule, regular rational readback, dyadic tolerance ledger, and truncation budget inspected before any terminal seal is available.

The optional seal row E is downstream of that displayed route. Transport, replay, provenance, and local naming are exactly the rows H, C, P, N, so the local NameCert surface is exhausted by the nine-row packet.

The excluded outputs have no carrier coordinates. Since the packet contains no quotient row, selected-limit row, countable-choice schedule, open-cover compactness witness, or ambient-completeness certificate, an accepted local read cannot produce one. □

Theorem 8921.3 (CauchyNameTruncation tail-budget route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyNameTruncation}^{\uparrow}$ *carrier must read the* $\mathsf{StreamName}^{\uparrow}$ *prefix row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the truncation budget row* T *before consuming the optional* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The route may be transported and replayed through* H *and* C*, but it cannot be replaced by quotient stream equality, a selected limit, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 8921.2. They place S, Q, D, T before the optional seal row E, with T naming the inspected truncation index and the tail budget associated to the displayed tolerance.

The sibling chapters supply the four external rows in order: finite prefix schedules from chapter 126, regular rational readback from chapter 301, dyadic tolerance accounting from chapter 291, and the terminal seal boundary from chapter 38. Thus a downstream read sees a finite prefix and its tail budget, not a completed quotient object.

Finally, H and C preserve the displayed route under transport and replay, while P and N keep the provenance and local NameCert rows explicit. No omitted output is recovered by this preservation, because none is represented inside the accepted carrier. □

Falsifiable prediction: An accepted $\mathsf{CauchyNameTruncation}^{\uparrow}$ read that reaches Real or Completion data without displaying StreamName prefix schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, truncation budget, optional seal, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyNameTruncation}^{\uparrow}$ is not bijective with any listed sibling because it binds finite prefix schedule, regular rational readback, dyadic tolerance, truncation index, optional Real seal, transport, replay, provenance, and local naming as one tail-budget packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNameTruncation}^{\uparrow}}$.

Constructive story: A CauchyNameTruncation$^{\uparrow}$ packet is generated from StreamName finite prefix schedules, RegSeqRat readback rows, DyadicRatCore tolerance ledgers, truncation-index and tail-budget rows, optional RealUp sealing, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyNameTruncation}^{\uparrow}}$, seed)

Scope closed: Definition 8921.1, Definition 8921.2, and Definition 8921.3 seed the finite regular Cauchy truncation surface.

Formal verification: FormalStatus(NameCert$_{\text{CauchyNameTruncation}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, ambient Real completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix exposure, regular readback, dyadic tolerance accounting, truncation-budget routing, optional Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNameTruncationUp.

Chapter 8922

A Concrete Naming Certificate for $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley-style separated reflection, the CompleteSeparatedMetr packet records the standard reflector landing surface that sends a displayed metric-completion request to a complete separated metric object. It sits after the complete-metric service of chapter 128, the separated-metric classifier of chapter 1411, the metric-completion comparison route of chapter 1506, the finite stream and regular rational readback stack of chapter 126 and chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite reflector certificate; it does not assert quotient completion equality, selected limits, countable choice, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8922.1 (CompleteSeparatedMetricReflector carrier)**.** *A* $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, S, M, W, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row that exposes the limit-service surface available to completion consumers;*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row, read only after zero-distance evidence has been displayed;*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row, carrying the metric-completion handoff and its reflector-facing comparison data;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for inspected Cauchy names and tolerance schedules;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and rational tolerance data;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row used for zero-distance and convergence-distance reads;*
7. *H records componentwise* hsame *transport for the complete-metric, separated-classifier, metric-completion, stream-window, readback, Real-seal, provenance, and local naming rows;*
8. *K records displayed* Cont *replay from the metric-completion request through complete-metric service, stream windows, regular rational readback, Real sealing, and separated reflection;*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their structural transport and replay data.

Theorem 8922.2 (CompleteSeparatedMetricReflector NameCert obligations)**.** *For every accepted* $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ *carrier* $R = (C, S, M, W, Q, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompleteSeparatedMet}}$ *surface consists exactly of carrier admission for the displayed tuple, complete-metric source exposure through* C*, separated classifier exposure through* S*, metric-completion comparison through* M*, StreamName finite-window exposure through* W*, RegSeqRat readback through* Q*, Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. Its coordinates display the complete-metric source, separated classifier, metric-completion comparison, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name.

Read the completion-facing rows first. The metric-completion row M supplies the handoff under inspection, while C supplies the complete-metric service row consumed by the reflector.

Read the finite name rows next. The stream-window row W precedes the regular rational readback Q, and the terminal Real seal E is downstream of those displayed windows and readbacks.

Finally apply the separated classifier row S and the structural rows H, K, P, N. They preserve, replay, source, and name the same finite reflector route, so no local NameCert obligation remains outside the listed rows. □

Theorem 8922.3 (CompleteSeparatedMetricReflector adjunction route)**.** *Every reflector-facing consumer of an accepted* $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{SeparatedMetric}^{\uparrow}.$$

The route is the displayed packet $R = (C, S, M, W, Q, E, H, K, P, N)$ *with* M *opening the metric-completion comparison,* C *supplying the complete-metric service,* W, Q, E *supplying the finite real-name readback, and* S *supplying the separated classifier.*

Proof. Start with the NameCert obligations of Definition 8922.2. They force every accepted reflector read to expose M, C, W, Q, E, S before the local name is consumed.

The reflector source is M followed by C. This is the finite metric-completion request together with the complete-metric service row, not a quotient completion object.

The readback stack is W, Q, E. The stream windows and regular rational readback are displayed before the Real seal, so the zero-distance or convergence-distance observation is reached only through finite name data.

The separated classifier S is terminal. Rows H, K, P, N transport, replay, source, and name this same route, so the consumer receives a reflector certificate rather than selected limits, countable choice, host point equality, or ambient Real completeness. □

Theorem 8922.4 (CompleteSeparatedMetricReflector real-seal handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ *carrier must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *before the terminal seal* E *is public, and must expose the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row* C*,* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row* M*, and* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row* S *before the reflector handoff is public.*

Proof. Project the carrier of Definition 8922.1. The Real-facing row is E, and the only rows feeding it are the displayed stream-window and regular-readback rows W, Q.

Apply the adjunction route of Definition 8922.3. It places the metric-completion comparison and complete-metric service before the real-name readback and places the separated classifier after the Real seal.

The structural rows H, K, P, N preserve that order under transport, replay, provenance, and local naming. Therefore a Real-facing consumer cannot bypass W, Q, and a reflector-facing consumer cannot bypass M, C, S. □

Falsifiable prediction: Every accepted $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ consumer read factors through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it can use the reflector handoff.

Independence witness: complete-metric, separated-metric, metric-completion, streamname, regseqrat, real: $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the complete service, separated classifier, metric-completion comparison, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}$ packet is generated from a CompleteMetricUp source row, a SeparatedMetricUp classifier row, a MetricCompletionUp comparison row, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8922.1, Definition 8922.2, Definition 8922.3, and Definition 8922.4 seed the finite complete separated metric reflector route over complete-metric, separated, metric-completion, stream, readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetricReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected limits, countable choice, host point equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric source exposure, separated classifier exposure, metric-completion comparison, finite-window readback, Real-seal handoff, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.CompleteSeparatedMetricReflectorUp.

Chapter 8923

A Concrete Naming Certificate for $\mathsf{IntervalExchange}^{\uparrow}$

Following Keane and Veech on interval exchange transformations, the $\mathsf{IntervalExchange}^{\uparrow}$ packet records the finite interval-map surface used when a displayed partition of a real interval is rearranged by finitely many translations. It sits over the dyadic arithmetic core of chapter 291, the finite cell net of chapter 5966, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the finite partition, permutation, and translation ledger by which later symbolic-dynamics or interval-map certificates may inspect an interval exchange; it does not assert invariant measures, minimality, unique ergodicity, host interval functions, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8923.1 (IntervalExchange carrier)**.** *An* $\mathsf{IntervalExchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, I, B, \Pi, T, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row containing the located interval endpoints, dyadic cut points, endpoint comparisons, and translation budgets.*
2. *I is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row listing the finite source cells and target cells of the partition.*
3. *B is the boundary-incidence row recording which neighboring source cells share displayed endpoints and which target cells share displayed endpoints.*
4. *Π is the finite permutation row matching each source cell to exactly one target cell.*
5. *T is the finite translation row; for each displayed source cell it records the dyadic displacement that carries the source endpoints to the corresponding target endpoints.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used whenever a source endpoint, target endpoint, or translated endpoint is inspected.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning the displayed endpoint and translation windows into regular rational comparisons.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, I, B, Π, T, W, Q have been displayed.*
9. *H records componentwise* hsame *transport for D, I, B, Π, T, W, Q, R and the structural rows.*

10. *C records displayed* Cont *replay from partition cells through permutation, translation, StreamName windows, RegSeqRat readback, and the Real seal.*
11. *P is the* Pkg *provenance row over* $\mathsf{IntervalExchange}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalExchange}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed endpoint, cell, incidence, permutation, translation, window, readback, seal, transport, continuation, provenance, and local naming rows.

Theorem 8923.2 (IntervalExchange NameCert obligations)**.** *For every accepted* $\mathsf{IntervalExchange}^{\uparrow}$ *carrier $E = (D, I, B, \Pi, T, W, Q, R, H, C, P, N)$, the local NameCert surface consists exactly of carrier admission for those rows; classifier stability under common dyadic refinement of the source and target cells; interval-cell transport through I, B, Π, T; finite orbit replay by iterating only the displayed permutation and translation rows for a displayed number of steps;* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through R; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and ledger nonescape through the local certificate N.*

Proof. Carrier admission is Definition 8923.1. It fixes the finite row inventory and makes the local classifier a comparison of displayed endpoint, cell, permutation, translation, window, readback, seal, transport, replay, provenance, and naming data.

Common refinement is finite. A refined carrier replaces listed cells in I by displayed dyadic subcells whose endpoint arithmetic is still read in D, whose incidences are still read in B, and whose source-to-target matching is still read through Π, T.

Finite orbit replay is also a row operation. A displayed number of steps iterates the permutation and translation ledgers already present in Π, T, opening StreamName windows through W and rational readback through Q before the Real seal R is visible.

Finally replay H, C, P, N. Transport, continuation, provenance, and local naming preserve the same finite ledger, so the packet cannot export invariant measures, minimality, unique ergodicity, host interval functions, quotient real equality, countable choice, or ambient completeness. □

Theorem 8923.3 (IntervalExchange L10 handoff)**.** *Every downstream consumer of an accepted* $\mathsf{IntervalExchange}^{\uparrow}$ *carrier must read the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before treating the packet as an interval map. The handoff order is*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{DyadicIntervalNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with the permutation and translation rows displayed between the interval-net read and the terminal seal.

Proof. Project the carrier $E = (D, I, B, \Pi, T, W, Q, R, H, C, P, N)$. The dyadic endpoint arithmetic and interval-cell rows D, I, B are the only partition data visible at the start of an interval-map read.

The consumer then opens Π, T. These rows give a finite source-to-target matching and a finite translation ledger, so no host interval function or quotient real equality is used to interpret the rearrangement.

The L10 numeric route remains downstream. Endpoint observations pass through W, tolerance and regular rational comparisons pass through Q, and the Real-facing boundary is exposed only by R.

Finally replay H, C, P, N. The same finite dependency order is preserved under transport, continuation, provenance, and naming, so the handoff reaches an interval-map surface without importing invariant measures, minimality, unique ergodicity, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{IntervalExchange}^{\uparrow}$ consumer that treats the packet as an interval map before reading $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: dyadicratcore, dyadic-interval-net, streamname, regseqrat, real: no listed sibling supplies the whole interval-exchange packet $E = (D, I, B, \Pi, T, W, Q, R, H, C, P, N)$, because the sibling chapters supply arithmetic, cells, windows, readback, and seal rows while this chapter binds the finite permutation and translation ledger that turns those rows into an interval exchange surface.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalExchange}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalExchange}^{\uparrow}$ packet is generated from located dyadic endpoints, dyadic interval cells, endpoint incidence, a finite permutation row, finite translation rows, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalExchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8923.1, Definition 8923.2, and Definition 8923.3 seed the finite interval-map NameCert boundary over dyadic interval partitions, permutation rows, translation rows, StreamName windows, RegSeqRat readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalExchange}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim invariant measures, minimality, unique ergodicity, host interval functions, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability under common refinement, interval-cell transport, finite orbit replay, L10 handoff order, ledger nonescape, provenance, and local naming for BEDC.Derived.IntervalExchangeUp.

Chapter 8924

A Concrete Naming Certificate for $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$

Following Kuratowski and Kelley, the $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source is embedded into its distance-profile interface before any completion-facing consumer reads a limit object. It sits after the metric carrier of chapter 57, the separated complete-metric boundary of chapter 128, the Cauchy-net handoff of chapter 589, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a distance-profile embedding route; it does not quotient completion names, choose limits, identify host equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8924.1 (Metric Kuratowski embedding carrier)**.** *A* $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, D, W, Q, E, C, F, H, R, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the named point carrier and displayed metric comparison data.*
2. *D is the distance-profile row, recording finite reads of $x \mapsto d(x, a)$ against named anchor rows a from the source.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which each distance profile is observed.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row induced by the displayed windows and dyadic tolerance data already carried by the regular-rational corridor.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the finite distance-profile values.*
6. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *comparison row available when the profile is consumed by a complete-metric target.*
7. *F is the* $\mathsf{CauchyNet}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row for completion-facing reads that enter through finite Cauchy data rather than through a selected limit.*
8. *H records componentwise* hsame *transport for source points, anchors, distance-profile rows, windows, readbacks, Real seals, complete-metric comparisons, Cauchy handoffs, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a profile request through $M, D, W, Q, E, C, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricKuratowskiEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, W, Q, E, C, F, H, R, P, N$. It has no coordinate for quotient completion, selected representatives, host equality of functions, ambient complete spaces, or a global Real-completeness theorem.

Theorem 8924.2 (Metric Kuratowski embedding NameCert obligations)**.** *For every accepted* $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ *carrier*

$$K = (M, D, W, Q, E, C, F, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricKuratowskiEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Met}^{\uparrow}$ *source, distance-profile exposure, finite StreamName observation, RegSeqRat readback, Real sealing, CompleteMetric comparison, CauchyNet or CauchyFilter handoff, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing read must expose the route*

$$\mathsf{Met}^{\uparrow} \to \mathsf{DistanceProfile} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and then choose either the $\mathsf{CompleteMetric}^{\uparrow}$ *comparison row or the* $\mathsf{CauchyNet}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *handoff row already displayed by the carrier.*

Proof. Project the carrier of Definition 8924.1. The source and profile rows are M and D, the observable analytic route is W, Q, E, and the completion-facing handoff rows are C and F.

Read the metric side first. The $\mathsf{Met}^{\uparrow}$ source supplies named points and metric comparisons, while D turns those comparisons into distance-profile rows against the displayed anchors.

The profile then passes through finite windows, regular rational readback, and the Real seal. The rows W, Q, E are the only value-facing route, so a consumer cannot bypass them to obtain a host function or ambient real value.

Finally C, F, H, R, P, N preserve the same finite packet for complete-metric and Cauchy consumers. Since no row stores quotient representatives, selected limits, host function equality, or ambient completeness, the NameCert obligations are exhausted by the displayed carrier. □

Theorem 8924.3 (Metric Kuratowski completion route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ *packet must read the finite profile route through* $\mathsf{Met}^{\uparrow}$ *source rows, distance-profile rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The consumer receives a displayed embedding interface for completion arguments, not quotient completion, selected limits, host equality of profiles, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $K = (M, D, W, Q, E, C, F, H, R, P, N)$. Carrier projection gives the consumer access only to those rows.

The metric source row M and distance-profile row D fix the embedding interface. A profile value is observable only after the StreamName window W, RegSeqRat readback Q, and Real seal E have been displayed.

The completion-facing rows C and F then route the same profile either through a complete-metric comparison or through a CauchyNet/CauchyFilter handoff. These rows do not add a selected limit; they only expose the finite comparison path already named by the packet.

The structural rows H, R, P, N transport, replay, source, and name the same data. Therefore the handoff is a finite distance-profile interface and not a quotient completion or ambient completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ completion read displays Metric source rows, distance-profile rows, StreamName windows, RegSeqRat readback, Real seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any CompleteMetric, CauchyNet, or CauchyFilter consumer can inspect the embedding interface.

Independence witness: metric, completemetric, cauchynet, cauchyfilter, streamname, regseqrat, real: $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a metric source, anchor-indexed distance profiles, finite observation windows, regular rational readback, Real sealing, completion handoffs, transport, replay, provenance, and local naming as one embedding packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricKuratowskiEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricKuratowskiEmbedding}^{\uparrow}$ packet is generated from Metric source rows, distance-profile rows, StreamName finite windows, RegSeqRat readback, RealUp seals, CompleteMetric comparison rows, CauchyNet or CauchyFilter handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricKuratowskiEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8924.1, Definition 8924.2, and Definition 8924.3 seed the finite distance-profile embedding interface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricKuratowskiEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, host equality of profile functions, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-profile exposure, finite-window observation, RegSeqRat readback, Real sealing, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricKuratowskiEmbeddingUp.

Chapter 8925

A Concrete Naming Certificate for $\mathsf{DyadicLocatedModulus}^{\uparrow}$

The $\mathsf{DyadicLocatedModulus}^{\uparrow}$ packet realizes the finite tolerance-to-window branch of the countable completion spine as BEDC NameCert data. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the Bishop located-modulus interface of chapter 6101, and the RegSeqRat located-modulus route of chapter 3532. The packet names a dyadic finite modulus; it does not quotient streams, select limits, decide total order, or import mature $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8925.1 (Dyadic located-modulus carrier)**.** *A* $\mathsf{DyadicLocatedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, W, R, E, B, G, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic precision request and slack comparison rows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected for that dyadic precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row induced by D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic tolerance and regular readback rows are displayed.*
5. *B is the* $\mathsf{BishopLocatedModulus}^{\uparrow}$ *interface row, naming the locatedness request that the dyadic window answers.*
6. *G is the* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *handoff row, binding the regular-rational readback to the located-modulus consumer.*
7. *H records componentwise* hsame *transport for dyadic tolerance data, stream windows, regular readbacks, Real seals, located-modulus interfaces, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a tolerance request through $D, W, R, E, B, G, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicLocatedModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopLocatedModulus}^{\uparrow}$, $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicLocatedModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, B, G, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient completeness.

Theorem 8925.2 (Dyadic located-modulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicLocatedModulus}^{\uparrow}$ *carrier*

$$L = (D, W, R, E, B, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicLocatedModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, Real sealing, Bishop located-modulus interface, RegSeqRat located-modulus handoff, componentwise transport, displayed replay, provenance, and local naming. Every countable-completion read must expose the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

before the located-modulus rows B and G can be consumed.

Proof. Project the carrier of Definition 8925.1. The dyadic tolerance row is D, the finite observation row is W, the regular readback row is R, and the Real seal row is E.

The located-modulus interface is downstream of those rows. The Bishop-facing row B names the locatedness request, while G records the RegSeqRat located-modulus handoff that answers it through the displayed regular readback.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no quotient stream, selected limit, trichotomy, countable-choice, host-equality, or ambient-completeness coordinate, the NameCert obligations are exhausted by the displayed rows. □

Theorem 8925.3 (Dyadic located-modulus RegSeqRat route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopLocatedModulus}^{\uparrow}$, *or* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicLocatedModulus}^{\uparrow}$ *packet must read the finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seals, the Bishop located-modulus interface, the RegSeqRat located-modulus handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The route is a finite tolerance-to-window object for the countable completion spine, not a quotient stream or selected-limit construction.*

Proof. Start with an accepted carrier $L = (D, W, R, E, B, G, H, C, P, N)$. The tolerance request enters through D, so no consumer can obtain a located-modulus answer without first displaying the dyadic precision row.

The selected finite window W and regular readback R bind that dyadic request to the countable completion corridor. The Real seal E is downstream of the same readback and does not supply an independent completeness theorem.

The located rows B and G consume the displayed readback as a finite modulus answer. The structural rows H, C, P, N preserve and name this route, so the consumer receives only the dyadic tolerance-to-window packet. □

Falsifiable prediction: Every accepted $\mathsf{DyadicLocatedModulus}^{\uparrow}$ read displays DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, RealUp seals, BishopLocatedModulus rows, RegSeqRatLocatedModulus rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a countable-completion consumer can inspect the located modulus.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-located-modulus, regseqrat-located-modulus: $\mathsf{DyadicLocatedModulus}^{\uparrow}$ is not bijective to any listed sibling because

it jointly binds dyadic precision, finite window selection, regular rational readback, Real sealing, Bishop located-modulus interface, RegSeqRat located-modulus handoff, transport, replay, provenance, and local naming as one finite modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicLocatedModulus}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, BishopLocatedModulus interface rows, RegSeqRatLocatedModulus handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8925.1, Definition 8925.2, and Definition 8925.3 seed the finite dyadic located-modulus object for the countable completion spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, trichotomy, countable choice, host real equality, ambient completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window selection, RegSeqRat readback, Real sealing, located-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicLocatedModulusUp.

Chapter 8926

A Concrete Naming Certificate for $\mathsf{QuasiArithmeticMean}^{\uparrow}$

Following Kolmogorov and Nagumo on generalized means, the $\mathsf{QuasiArithmeticMean}^{\uparrow}$ packet records the finite carrier by which a continuous real transform, a finite list of real inputs, and a finite averaging row are inspected before a Real-valued aggregation is read. It sits after the terminal real-name surface of chapter 38, the continuous-map carrier of chapter 130, the finite input spine of chapter 36, and the finite-sum surface of chapter 1880. The packet names the Kolmogorov–Nagumo finite aggregation route; it does not import quotient equality of real functions, a selected inverse transform, infinite-series convergence, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8926.1 (Quasi-arithmetic mean carrier)**.** *A* $\mathsf{QuasiArithmeticMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, I, V, A, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *transform row, naming the inspected continuous real observable used by the mean.*
2. *I is the* $\mathsf{List}^{\uparrow}$ *finite input row, recording the ordered finite family of* $\mathsf{Real}^{\uparrow}$ *names to be aggregated.*
3. *V is the transformed-value row, exposing the finite list obtained by reading F on the entries of I.*
4. *A is the finite averaging row, carried through the same finite-sum discipline as the sum surfaces of chapter 1880.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *result row exposed only after F, I, V, A have been displayed.*
6. *H records componentwise* hsame *transport for the transform, input list, transformed values, finite average, result, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an aggregation request through F, I, V, A, R, H.*
8. *P is the* Pkg *provenance over* $\mathsf{QuasiArithmeticMean}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiArithmeticMean}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, V, A, R, H, C, P, N$. It has no coordinate for a host inverse-function theorem, quotient equality of real functions, infinite averaging, measure integration, countable choice, or ambient completeness.

Theorem 8926.2 (Quasi-arithmetic mean NameCert obligations)**.** *For every accepted* $\mathsf{QuasiArithmeticMean}^{\uparrow}$ *carrier*

$$Q = (F, I, V, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QuasiArithmeticMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the continuous transform row, finite input-list row, transformed-value row, finite averaging row, terminal Real result row, componentwise transport, displayed replay, package provenance, and local naming. Every aggregation consumer must read the route*

$$\mathsf{ContinuousMap}^{\uparrow} \to \mathsf{List}^{\uparrow} \to \mathsf{DarbouxSum}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

through the displayed rows F, I, V, A, R *before the output row* R *is available.*

Proof. Project the carrier of Definition 8926.1. The analytic rows are F, I, V, A, R, and the structural rows are H, C, P, N.

Read the transform and finite input first. The row F supplies the inspected continuous real observable, while I supplies only a displayed finite list of Real names. The transformed-value row V is therefore a finite readback from those two rows, not a host-level function quotient.

The aggregation step is the finite row A. It records the finite averaging surface over the transformed values before the terminal Real row R is exposed. Since the carrier has no coordinate for infinite summation, measure integration, selected inverses, countable choice, or ambient completeness, the local NameCert obligations are exhausted by the displayed route and the structural rows that preserve it. □

Theorem 8926.3 (Quasi-arithmetic mean generator handoff)**.** *Every downstream generator that consumes an accepted* $\mathsf{QuasiArithmeticMean}^{\uparrow}$ *packet receives only the rows*

$$F, I, V, A, R, H, C, P, N.$$

The handoff exposes a continuous transform row, a finite input-list row, transformed finite values, a finite average row, a terminal Real row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It exports no quotient equality of real functions, selected inverse transform, infinite-series convergence, measure-integral equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier $Q = (F, I, V, A, R, H, C, P, N)$. Carrier projection gives a generator access only to the displayed rows.

The generator first reads F and I, then forms the transformed finite row V. The averaging row A is downstream of those displayed rows and is the only aggregation coordinate carried by the packet.

Finally the terminal Real row R is transported and replayed through H and C, with provenance and local naming recorded by P and N. No excluded object has a carrier slot, so the generator handoff is exactly the finite route named by the packet. □

Falsifiable prediction: An accepted $\mathsf{QuasiArithmeticMean}^{\uparrow}$ consumer that reads a Real output without first displaying the ContinuousMap transform row, finite List input row, transformed-value row, finite average row, hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: real, continuousmap, list, darboux-sum: no listed sibling is bijective with $\mathsf{QuasiArithmeticMean}^{\uparrow}$ because this packet jointly binds a continuous transform, finite real

input spine, transformed finite values, finite averaging, terminal Real output, transport, replay, provenance, and local naming as one aggregation carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiArithmeticMean}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasiArithmeticMean}^{\uparrow}$ packet is generated from a ContinuousMapUp transform row, a finite ListUp input row of RealUp names, transformed finite values, a finite average row, a terminal RealUp result, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiArithmeticMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8926.1, Definition 8926.2, and Definition 8926.3 seed the finite Kolmogorov–Nagumo aggregation route from continuous transform and finite input list to Real output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiArithmeticMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of real functions, selected inverse transforms, infinite averaging, measure integration, countable choice, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-transform exposure, finite input-list exposure, transformed-value readback, finite-average discipline, Real result nonescape, transport, replay, provenance, and local naming for BEDC.Derived.QuasiArithmeticMeanUp.

Chapter 8927

A Concrete Naming Certificate for $\mathsf{DyadicRealInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicRealInterval}^{\uparrow}$ packet records the finite endpoint envelope used when dyadic rational bounds are handed to Real and Completion consumers. It sits after the dyadic rational core of chapter 1658, the finite stream naming surface of chapter 126, the regular rational sequence readback of chapter 301, the terminal Real seal of chapter 38, and the closed bounded interval net route of chapter 1250. The packet is an interval-envelope certificate for displayed dyadic endpoints; it is not a quotient real interval, a selected completion point, a compactness theorem, or a proof of ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8927.1 (DyadicRealInterval carrier)**.** *A* $\mathsf{DyadicRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, O, S, Q, E, N, H, C, P, K)$$

with the following displayed rows:

1. *L is the left* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
2. *R is the right* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row.*
3. *O is the endpoint-order evidence row, recording that the displayed left endpoint is admitted before the displayed right endpoint for this interval envelope.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect endpoint approximations and interval refinements.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row through which endpoint and interval-envelope reads become Real-facing.*
7. *N is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row for finite closed bounded interval-net consumers.*
8. *H records componentwise* hsame *transport for dyadic endpoints, endpoint order, stream windows, regular rational readback, Real sealing, interval-net handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from an interval-envelope query through $L, R, O, S, Q, E, N, H, P, K$.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. K is the local $\mathsf{NameCert}_{\mathsf{DyadicRealInterval}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, R, O, S, Q, E, N, H, C, P, K$. It has no coordinate for quotient real equality, selected completion points, arbitrary located compactness, host interval topology, or ambient Real completeness.

Theorem 8927.2 (DyadicRealInterval Real seal obligations)**.** *For every accepted* $\mathsf{DyadicRealInterval}^{\uparrow}$ *carrier*

$$I = (L, R, O, S, Q, E, N, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicRealInterval}^{\uparrow}}$ *surface consists exactly of left and right dyadic endpoint exposure through L, R, endpoint-order evidence through O, finite StreamName windows through S, RegSeqRat readback through Q, Real sealing through E, closed bounded interval-net handoff through N, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through K.*

Proof. Project the carrier of Definition 8927.1. Rows L, R, O are the endpoint envelope, rows S, Q, E, N are the completion-facing readback and interval-net surface, and rows H, C, P, K are the structural certificate data.

The endpoint part is read first. The two dyadic endpoint rows remain separate, and O records the displayed order evidence before any Real-facing seal can be consumed.

The completion-facing part is then read through S, Q, E, N. Stream windows precede regular rational readback, the Real seal is downstream of that readback, and the interval-net row is a finite closed bounded consumer row rather than an ambient compact interval.

Finally H, C, P, K transport, replay, source, and name the same finite packet. Since no coordinate stores quotient real equality, selected completion points, host interval topology, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8927.3 (DyadicRealInterval completion consumer route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicRealInterval}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \times \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{ClosedBoundedIntervalNet}^{\uparrow}.$$

The consumer may inspect dyadic endpoint rows, endpoint-order evidence, finite stream windows, regular rational readback, Real sealing, closed bounded interval-net handoff, transport, replay, provenance, and local naming. It receives no quotient real interval, selected completion point, host interval topology, arbitrary compactness theorem, or ambient Real completeness.

Proof. Begin with the NameCert obligations of Definition 8927.2. They force every accepted consumer read to expose $L, R, O, S, Q, E, N, H, C, P, K$.

The endpoint segment is L, R, O. Both endpoint rows are dyadic rational core rows, and the order evidence is displayed as part of the same finite packet rather than inferred from an external real-order theorem.

The readback segment is S, Q, E. A completion-facing consumer reaches the Real seal only after finite StreamName windows and RegSeqRat readback have been displayed.

The interval-net segment is N, transported and replayed by H, C and sourced and named by P, K. Thus the consumer receives a finite interval-envelope route into closed bounded interval-net data, with no hidden quotient, selected point, compactness, topology, or ambient-completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{DyadicRealInterval}^{\uparrow}$ consumer read displays two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, endpoint-order evidence, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert naming before the interval envelope can be consumed.

Independence witness: dyadic-rat-core, streamname, regseqrat, real, closed-bounded-interval-net: $\mathsf{DyadicRealInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds two ordered dyadic endpoints, finite stream windows, regular rational readback, terminal Real sealing, closed bounded interval-net handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRealInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRealInterval}^{\uparrow}$ packet is generated from left and right DyadicRatCore endpoint rows, endpoint-order evidence, StreamName finite windows, RegSeqRat readback, a RealUp seal, a ClosedBoundedIntervalNet handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8927.1, Definition 8927.2, and Definition 8927.3 seed the finite dyadic endpoint interval-envelope route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real intervals, selected completion points, arbitrary located compactness, host interval topology, ambient RealUp completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, endpoint-order evidence, StreamName window visibility, RegSeqRat readback, Real-seal nonescape, interval-net handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicRealIntervalUp.

Chapter 8928

A Concrete Naming Certificate for $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley's uniform-space terminology, the $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ packet records the finite comparison route by which a located uniform-completion presentation is read beside a plain uniform-completion presentation before Real and Completion consumers inspect either endpoint. It sits after the located uniform-completion packet of chapter 1692, the uniform completion carrier of chapter 1473, the finite stream-name observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed comparison route; it is not quotient completion equality, selected representative equality, host uniform-space equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8928.1 (Located uniform completion comparison carrier)**.** *A* $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *source row, exposing the located Cauchy-filter and completion route under comparison;*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row, exposing the corresponding uniform-completion route without adding a quotient comparison object;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both presentations expose inspected observations;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those displayed windows and tolerance data;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after L, U, S, R have been displayed;*
6. *H records componentwise* hsame *transport for the two completion sources, finite windows, regular readback, Real seal, provenance, and local naming rows;*
7. *C records displayed* Cont *replay of the ordered route from the two completion sources through the finite readback and terminal seal;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$, $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletionComparison}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows and their structural transport and replay data. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8928.2 (Located uniform completion comparison NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ *carrier* $K = (L, U, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, located uniform-completion source exposure through* L*, uniform-completion source exposure through* U*, StreamName finite-window exposure through* S*, RegSeqRat readback through* R*, terminal Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8928.1. Its coordinates display the located completion source, the plain uniform-completion source, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name.

Read the two completion sources first. The row L supplies the located uniform-completion presentation, while U supplies the companion uniform-completion presentation; neither row adds quotient equality between completed objects.

Read the finite observation stack next. The row S supplies the inspected StreamName windows, R supplies the corresponding RegSeqRat readback, and E is downstream of those rows as the terminal Real seal.

Finally use H, C, P, N to transport, replay, source, and name the same route. Therefore the NameCert surface has no local obligation outside the displayed comparison packet. □

Theorem 8928.3 (Located uniform completion comparison finite route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ *packet factors through the finite displayed route*

$$\mathsf{LocatedUniformCompletion}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The structural rows H, C, P, N *preserve that order under transport, replay, provenance, and local naming, so the comparison is a route between displayed completion presentations rather than an ambient equivalence of completions.*

Proof. Start with the obligation surface of Definition 8928.2. It forces any accepted consumer read to expose L, U, S, R, E before the local name is consumed.

The comparison source is the pair L, U. Since both rows are part of the same finite packet, a consumer may compare their displayed routes only by reading the carrier rows, not by importing host equality of completed objects.

The readback stack is S, R, E. StreamName windows are inspected before regular rational readback, and the Real seal is readable only after that finite observation stack has been displayed.

Rows H, C, P, N transport, replay, source, and name this same dependency order. Thus the consumer receives a finite comparison route and no selected representative, quotient completion equality, countable-choice schedule, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ consumer read displays one $\mathsf{LocatedUniformCompletion}^{\uparrow}$ source row, one $\mathsf{UniformCompletion}^{\uparrow}$ source row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame

transport, displayed Cont replay, Pkg provenance, and local naming before any Real or Completion handoff is consumed.

Independence witness: located-uniform-completion, uniformcompletion, streamname, regseqrat, real: $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located completion source, plain uniform-completion source, finite windows, regular readback, Real seal, transport, replay, provenance, and local NameCert rows in one packet.

Lean4 (statement) `BEDC.Derived.LocatedUniformCompletionComparisonUp`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCompletionComparison}^{\uparrow}$ packet is generated from a LocatedUniformCompletionUp source row, a UniformCompletionUp source row, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8928.1, Definition 8928.2, and Definition 8928.3 seed the finite comparison route through located completion, uniform completion, finite stream windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, host equality of completed objects, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located completion source exposure, uniform completion source exposure, StreamName finite-window exposure, RegSeqRat readback, Real-seal handoff, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.LocatedUniformCompletionComparisonUp.

Chapter 8929

A Concrete Naming Certificate for $\mathsf{BishopCompactImage}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompactImage}^{\uparrow}$ packet records the constructive continuous image of compact metric data as finite compact-source, continuous-map, uniform-modulus, image-witness, regular rational, dyadic tolerance, and terminal real-seal rows. It sits after the compact metric source of chapter 129, the continuous-map surface of chapter 130, the regular sequence readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The chapter names only the finite image carrier, its NameCert rows, and its Real-facing handoff; it does not assert a quotient image space, an ambient compactness theorem, countable choice, or host-level function equality.

Falsifiable prediction: A consumer that reads a compact-image Real endpoint without the displayed compact source, continuous-map row, uniform-modulus ledger, image witness, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{Real}^{\uparrow}$ seal has left the $\mathsf{BishopCompactImage}^{\uparrow}$ packet.

Independence witness: compactmetric, continuousmap, regseqrat, dyadicratcore, real: no listed sibling supplies the whole compact-image packet, because $\mathsf{CompactMetric}^{\uparrow}$ names the compact source, $\mathsf{ContinuousMap}^{\uparrow}$ names the map row, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ name value readback, and $\mathsf{Real}^{\uparrow}$ seals endpoints while this chapter binds their finite image-witness route.

Definition 8929.1 (Bishop compact-image carrier)**.** *A* $\mathsf{BishopCompactImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CompactMetric}^{\uparrow}$ *compact-source row, including the finite cover or finite-net data exposed by that sibling packet.*
2. *F is the accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map from the displayed compact source into the displayed metric target.*
3. *M is the uniform-modulus ledger produced from the compact-source and continuous-map rows.*
4. *W is the image-witness table: each displayed image request points back to a compact-source row, a map-evaluation row, and the modulus row used to certify the requested tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed image values.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the modulus and readback comparisons.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed image values.*
8. *H records componentwise* hsame *transport for $K, F, M, W, R, D, E, C, P, N$.*
9. *C records displayed* Cont *replay from compact-source requests through continuous-map evaluation, uniform-modulus use, image-witness inspection, regular rational readback, dyadic tolerance comparison, and the Real seal.*
10. *P is the* Pkg *provenance row naming exactly* $\mathsf{BishopCompactImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactImage}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed rows. It has no coordinate for a quotient image set, a host subset equality, countable choice of preimages, or ambient compactness outside the named source.

Theorem 8929.2 (Bishop compact-image NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactImage}^{\uparrow}$ *carrier*

$$I = (K, F, M, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompactImage}^{\uparrow}}$ *surface consists exactly of carrier exposure for I, compact-source exposure through K, continuous-map exposure through F, uniform-modulus handoff through M, image-witness exactness through W, regular rational readback through R, dyadic tolerance exposure through D, terminal Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8929.1. The rows K, F, M, W, R, D, E are the analytic route, and H, C, P, N are the structural rows.

The source-facing read begins at K and F. The compact metric row supplies only the displayed compact-source data, while the continuous-map row supplies only the named map-evaluation surface on that source.

The image-facing read then passes through M, W, R, D, E. The uniform-modulus ledger fixes the tolerance route, the witness table points each image request back to a displayed source and map-evaluation row, the regular rational row reads the requested values, the dyadic row supplies the tolerance arithmetic, and the Real row seals the displayed endpoint.

Transport, replay, provenance, and naming are exactly the rows H, C, P, N. Since no carrier coordinate names a quotient image set, a host subset equality, a choice function for all preimages, or ambient compactness beyond K, the listed obligations exhaust the local NameCert surface. □

Theorem 8929.3 (Bishop compact-image Real-completion handoff)**.** *Every Real-facing image consumer of an accepted* $\mathsf{BishopCompactImage}^{\uparrow}$ *carrier factors through the displayed compact source, continuous-map row, uniform-modulus ledger, image-witness table,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{Real}^{\uparrow}$ *seal:*

$$K, F, M, W, R, D, E, H, C, P, N.$$

The handoff exposes a finite constructive image route and no quotient image carrier, host-level function equality, countable preimage choice, ambient Heine–Borel principle, or completed image-space object.

Proof. Start with the NameCert obligations of Definition 8929.2. They identify the compact-source, continuous-map, modulus, witness, readback, dyadic, and Real rows as the whole Real-facing route.

A consumer that requests an image endpoint must first inspect K, F, M, W. These rows determine the displayed compact source, the displayed map evaluation, the modulus used for the request, and the finite witness that links the image read to a source row.

The endpoint read then passes through R, D, E. Regular rational readback and dyadic tolerance comparison produce the value data that the terminal $\mathsf{Real}^{\uparrow}$ seal may expose.

Finally H, C, P, N preserve the same route under transport, replay it under displayed continuation, name its sibling sources, and bind the local certificate. No listed row supplies an unbounded preimage selector, a quotient image set, or an ambient compactness theorem, so every Real-facing consumer remains inside the finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactImage}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactImage}^{\uparrow}$ packet is generated from a displayed compact metric source, continuous-map row, uniform-modulus ledger, image-witness table, regular rational readback, dyadic tolerance arithmetic, terminal Real seal, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8929.1, Definition 8929.2, and Definition 8929.3 seed the finite constructive compact-image carrier and its Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient image spaces, ambient compactness theorems, countable preimage choice, host function equality, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the finite carrier and prove compact-source exposure, continuous-map exposure, uniform-modulus handoff, image-witness exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactImageUp.

Chapter 8930

A Concrete Naming Certificate for $\mathsf{UniformlyLocatedSet}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric spaces, the $\mathsf{UniformlyLocatedSet}^{\uparrow}$ packet records the finite BEDC surface by which a displayed subset is read through uniform locator data before any Real or completion consumer inspects distances. It sits after the metric source row of chapter 57, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a constructive located-subset interface; it does not import host subsets, countable choice, quotient completions, or ambient Real completeness.

Definition 8930.1 (Uniformly located set carrier)**.** *A* $\mathsf{UniformlyLocatedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, L, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the named point carrier and displayed metric comparison data.*
2. *L is the located-set row or locator-modulus row, recording the finite witness that answers distance-to-the-subset requests uniformly over the displayed metric source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds each locator request by a dyadic precision and slack comparison.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the locator answer is observed.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row induced by the displayed window and tolerance rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the distance readback reached through D, W, Q.*
7. *H records componentwise* hsame *transport for metric source rows, locator rows, dyadic tolerances, finite windows, regular readbacks, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a distance-to-set request through $M, L, D, W, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyLocatedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, D, W, Q, E, H, C, P, N$. It has no coordinate for host subset membership, selected nearest points, countable choice over approximants, quotient completion, or ambient Real-completeness principles.

Theorem 8930.2 (Uniformly located set NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyLocatedSet}^{\uparrow}$ *carrier*

$$U = (M, L, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSet}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, located-set or locator-modulus row, dyadic tolerance row, finite StreamName observation, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. Every distance-to-set read must expose the route*

$$\mathsf{Met}^{\uparrow} \to \mathsf{LocatorModulus} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

before any Real or completion-facing consumer can inspect the located-subset interface.

Proof. Project the carrier of Definition 8930.1. The metric source and locator data are M and L, the finite tolerance-to-value route is D, W, Q, E, and the structural rows are H, C, P, N.

The source row supplies only named metric comparisons. The locator row answers a distance-to-set request by displaying a dyadic tolerance row, a finite observation window, a regular rational readback, and the Real seal reached through that readback.

The structural rows transport, replay, source, and name the same finite packet. Since the carrier has no host-subset coordinate, nearest-point selector, countable-choice ledger, quotient-completion row, or ambient-completeness row, the NameCert obligations are exhausted by the displayed carrier. □

Theorem 8930.3 (Uniformly located set Real and completion route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformlyLocatedSet}^{\uparrow}$ *packet must read the metric source row, located-set or locator-modulus row, dyadic tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. The consumer receives a finite constructive located-subset interface, not a host subset, a selected nearest point, countable choice, quotient completion, or ambient Real completeness.*

Proof. Start with the accepted carrier $U = (M, L, D, W, Q, E, H, C, P, N)$. Carrier projection gives a Real or completion-facing consumer access only to those rows.

The source and locator rows M, L determine the distance-to-set request. The dyadic tolerance D, finite window W, regular readback Q, and Real seal E are the only value-facing route for that request.

The transport, replay, provenance, and local certificate rows preserve the same route without adding selectors or completion quotients. Therefore any downstream consumer reads a finite located-subset packet and not an ambient Real-completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{UniformlyLocatedSet}^{\uparrow}$ read displays MetricSpace source rows, located-set or locator-modulus rows, DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, RealUp seals, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before a Real or completion consumer can inspect the located-subset interface.

Independence witness: metricspace, dyadicratcore, streamname, regseqrat, real: $\mathsf{UniformlyLocatedSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a metric source, locator-modulus

evidence, dyadic precision, finite observation windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming as one located-subset packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyLocatedSet}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyLocatedSet}^{\uparrow}$ packet is generated from MetricSpace source rows, located-set or locator-modulus rows, DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyLocatedSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8930.1, Definition 8930.2, and Definition 8930.3 seed the finite constructive located-subset interface for Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyLocatedSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host subset membership, nearest-point selection, countable choice, quotient completion, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locator-modulus exposure, dyadic tolerance exactness, finite-window observation, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.UniformlyLocatedSetUp.

Chapter 8931

A Concrete Naming Certificate for $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$

Following the classical nerve construction for finite covers and the finite-incidence reading of simplicial chains, the $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ packet records the BEDC rows that turn displayed cover membership and overlap data into a simplicial and homology-facing handoff. It sits after the topological source of chapter 88, the finite cover nerve of chapter 7597, the simplicial complex interface of chapter 238, the homology rows of chapter 98, and the categorical carrier of chapter 61. The packet names finite incidence data; it does not import a host cover, selected refinements, ambient chain complexes, or quotient homology.

Definition 8931.1 (Finite cover incidence realizer carrier)**.** *A* $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (T, K, M, O, F, S, A, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row for the covered object.*
2. *K is the* $\mathsf{FiniteCoverNerve}^{\uparrow}$ *cover row, including the finite cover index list and cover-member exposure.*
3. *M is the finite membership-incidence ledger pairing displayed source rows with displayed cover-index rows.*
4. *O is the overlap-incidence ledger for nonempty finite intersections.*
5. *F is the simplicial face ledger induced by M and O.*
6. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *handoff row that reads the finite face ledger as a simplicial carrier.*
7. *A is the* $\mathsf{Homology}^{\uparrow}$ *handoff row for finite chains generated from the accepted simplicial faces.*
8. *G is the* $\mathsf{Cat}^{\uparrow}$ *provenance row for functorial comparison of cover maps, face maps, and homology consumers.*
9. *H records componentwise* hsame *transport for topology, cover, membership, overlap, face, simplicial, homology, category, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a cover-incidence request through $T, K, M, O, F, S, A, G, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{FiniteCoverNerve}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, K, M, O, F, S, A, G, H, C, P, N$. It has no coordinate for host open covers, selected refinements, ambient simplicial sets, quotient chain complexes, or global homology representatives.

Theorem 8931.2 (Finite cover incidence realizer NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ *carrier*

$$I = (T, K, M, O, F, S, A, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for topology source rows, finite cover rows, membership incidence, overlap incidence, simplicial face admission, simplicial-complex handoff, homology handoff, categorical provenance, componentwise transport, displayed replay, package provenance, and local naming. Every simplicial or homology consumer must expose the route*

$$\mathsf{Topology}^{\uparrow} \to \mathsf{FiniteCoverNerve}^{\uparrow} \to \mathsf{IncidenceLedger} \to \mathsf{SimplicialComplex}^{\uparrow} \to \mathsf{Homology}^{\uparrow}$$

with the $\mathsf{Cat}^{\uparrow}$ *provenance row displayed before functorial comparison is consumed.*

Proof. Project the carrier of Definition 8931.1. The topology and finite-cover inputs are T, K, the incidence ledgers are M, O, F, the downstream handoff rows are S, A, G, and the structural rows are H, C, P, N.

The finite membership and overlap rows determine exactly which finite faces are admitted. The simplicial-complex row consumes that face ledger, and the homology row consumes the finite-chain data generated from the same accepted faces.

The category provenance row records how cover maps, face maps, and homology-facing consumers compare without adding a host category of covers. The transport, replay, provenance, and local naming rows preserve this finite packet, so the NameCert obligations are exhausted by the displayed carrier. □

Falsifiable prediction: Every accepted $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ read displays Topology source rows, FiniteCoverNerve cover rows, membership incidence, overlap incidence, simplicial face rows, SimplicialComplex handoff, Homology handoff, Category provenance, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before a simplicial or homology consumer can inspect the incidence interface.

Independence witness: topology, finite-cover-nerve, simplicialcomplex, homology, category: $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds cover membership, overlap incidence, simplicial faces, homology handoff, categorical provenance, transport, replay, provenance, and local naming as one finite incidence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}$ packet is generated from Topology source rows, FiniteCoverNerve cover rows, finite membership and overlap incidence ledgers, simplicial face rows, SimplicialComplex handoff rows, Homology handoff rows, Category provenance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}}$, seed)

Scope closed: Definition 8931.1 and Definition 8931.2 seed the finite cover-incidence interface for simplicial, homology, and categorical consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCoverIncidenceRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host open covers, selected refinements, ambient simplicial sets, quotient chain complexes, global homology representatives, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, membership-incidence exactness, overlap-incidence exactness, simplicial face handoff, homology handoff, categorical provenance, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverIncidenceRealizerUp.

Chapter 8932

A Concrete Naming Certificate for $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$ packet records the finite mesh object used when a Real or Completion consumer needs dyadic interval cells with located endpoints and regular readback. It sits after the dyadic arithmetic of chapter 291, the stream schedule rows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the interval-cell surface of chapter 5966. The packet supplies a finite L10 mesh handoff without quotient representatives, choice-selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8932.1 (Located dyadic uniform mesh carrier)**.** *A* $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$ *carrier is a finite* BHist *packet* $M = (I, L, Q, S, R, H, C, P, N)$ *with the following displayed rows:*

1. *I is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *row carrying a finite dyadic interval net and its cell-incidence ledger.*
2. *L is the located endpoint evidence row, recording left and right dyadic endpoint comparisons for each visible cell.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row that exposes the mesh width and requested precision as regular rational data.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row that schedules the finite windows inspected by the mesh consumer.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after I, L, Q, S have been displayed.*
6. *H, C, P, N are respectively componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicUniformMesh}^{\uparrow}}$ *row.*

Theorem 8932.2 (LocatedDyadicUniformMesh NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDyadicUniform}$ *carrier $M = (I, L, Q, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicUniformMesh}^{\uparrow}}$ *surface consists exactly of dyadic interval-net admission, located endpoint evidence, RegSeqRat tolerance readback, StreamName observation rows, terminal Real seal, componentwise transport, continuation replay, provenance, and local naming. A consumer can therefore inspect the mesh only through the finite route*

$$\mathsf{DyadicIntervalNet}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The obligation surface exports no quotient representative, selected limit, host real equality, countable-choice schedule, or ambient completeness theorem.

Proof. Project the carrier rows of Definition 8932.1. The dyadic interval-net row supplies finite cells, the endpoint row supplies located comparisons, and the tolerance and observation rows keep width and inspection data visible before the Real seal is read.

Replay the structural rows H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite route under classifier transport and do not add a coordinate for quotient equality, selected limits, choice schedules, host real equality, or ambient completeness. □

Theorem 8932.3 (LocatedDyadicUniformMesh refinement stability)**.** *Let $M = (I, L, Q, S, R, H, C, P, N)$ and $M' = (I', L', Q', S', R', H', C', P', N')$ be accepted* $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$ *carriers. If M' refines the dyadic interval net of M and the endpoint, tolerance, StreamName, RegSeqRat, Real-seal, and structural rows are transported by componentwise* hsame*, then every classifier accepted by M is accepted by M' through the same displayed finite route. The refinement exports only smaller dyadic cells with the same located endpoint and readback surface.*

Proof. Read the interval-net row I and its refined row I' through the dyadic interval-net carrier. Refinement replaces each visible cell by displayed dyadic subcells while preserving the endpoint order and incidence ledger.

Transport the endpoint evidence, tolerance readback, StreamName windows, RegSeqRat row, and Real seal along the stated hsame witnesses. Since the carrier exposes no coordinate outside these rows, classifier acceptance is stable under the refinement and remains tied to the same finite mesh route. □

Theorem 8932.4 (LocatedDyadicUniformMesh L10 handoff)**.** *Every L10 Real and Completion consumer of a* $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$ *packet must read the dyadic interval net I, located endpoint evidence L, RegSeqRat tolerance row Q, StreamName observation row S, and Real seal R before it treats the packet as a mesh for approximation, comparison, or completion-facing readback. The handoff supplies finite dyadic cells and located endpoint data only; it does not supply quotient real representatives, selected Cauchy limits, host real equality, choice schedules, or an ambient completeness principle.*

Proof. Apply the NameCert obligations above. They force the consumer to read the interval net, endpoint evidence, tolerance readback, observation schedule, and Real seal in the displayed order before any completion-facing use is public.

Now use refinement stability. Any later mesh refinement remains inside the same finite route and therefore cannot introduce an external representative, selected limit, choice schedule, host equality, or ambient completeness endpoint. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicUniformMesh}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedDyadicUniformMesh}^{\uparrow}}$, seed)

Scope closed: Definition 8932.2 records the seed carrier and finite L10 mesh route over dyadic interval nets, located endpoints, RegSeqRat tolerance readback, StreamName observations, and Real seals.

Constructive story: A $\mathsf{LocatedDyadicUniformMesh}^{\uparrow}$ packet is generated from dyadic interval-net cells, located endpoint evidence, RegSeqRat tolerance readback, StreamName observation windows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedDyadicUniformMesh}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.LocatedDyadicUniformMeshUp`

Bridge status: none

Not claimed: This chapter does not close a public completion theorem, quotient-free Real equality, compactness theorem, or checked standard bridge.

Upgrade path: A scoped closure requires Lean-checked carrier obligations and refinement stability for the displayed mesh route.

Chapter 8933

A Concrete Naming Certificate for $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$

Following the finite-net proof of Heine–Cantor, the $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$ packet turns the compact-uniform-continuity vision of chapter 9237 into a concrete finite handoff. It reads compact finite-net data from chapter 60, pointwise continuity rows from chapter 59, dyadic tolerance arithmetic from chapter 291, regular rational readback from chapter 301, and the terminal real seal from chapter 38. The packet records the compact-net, pointwise-modulus, rational-fold, and uniform-modulus route as finite NameCert data rather than as an open-cover compactness principle.

Definition 8933.1 (Compact modulus mesh handoff carrier)**.** *A* $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet* $K = (C, M, D, F, U, R, H, T, P, N)$ *with the following displayed rows:*

1. *C is the compact finite-net row over the* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *source.*
2. *M is the pointwise-modulus row supplied by the* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the requested mesh precision.*
4. *F is the rational lower-bound fold row over the finite probe bundle.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced by the fold.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for Real-valued consumers.*
7. *H, T, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactModulusMeshHandoff}^{\uparrow}}$ *row.*

Theorem 8933.2 (CompactModulusMeshHandoff NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$ *carrier* $K = (C, M, D, F, U, R, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusMeshHand}}$ *surface consists exactly of compact finite-net admission, pointwise-modulus exposure, dyadic tolerance arithmetic, rational lower-bound fold, UniformModulus handoff, Real seal, componentwise transport, replay, provenance, and local naming. The route exports finite compact-net and modulus data only; it exports no open-cover compactness oracle, selected Lebesgue number outside the fold, quotient function equality, host supremum, choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8933.1. The compact finite-net row supplies the centers, the pointwise-modulus row supplies local radii, the dyadic tolerance row supplies the requested precision, and the fold row computes the displayed lower bound before the UniformModulus handoff is public.

Replay H, T, P, N. These rows transport, order, source, and name the same finite packet, so the consumer receives the visible compact-net and modulus data without any external compactness oracle, selected number, quotient equality, host supremum, choice schedule, or ambient completeness coordinate. □

Theorem 8933.3 (CompactModulusMeshHandoff uniform modulus)**.** *Every compact-uniform-continuity consumer that reads an accepted* $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$ *carrier obtains its* $\mathsf{UniformModulus}^{\uparrow}$ *row only after the compact finite-net row C, pointwise-modulus row M, dyadic tolerance row D, and rational lower-bound fold row F have been displayed. The resulting uniform modulus is the finite mesh handoff of the carrier and not a choice-selected modulus, open-cover compactness theorem, quotient function equality, or host supremum argument.*

Proof. Start from the NameCert obligations. They force the consumer through C, M, D, F before the uniform-modulus row U is exposed, and they keep the Real seal and structural rows downstream of that finite handoff.

The finite lower-bound fold is a displayed row of the carrier, so the modulus readback is internal to the packet. Consequently the consumer cannot replace the fold by an external selector, open-cover compactness theorem, quotient equality, host supremum, choice schedule, or ambient real-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusMeshHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusMeshHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8933.2 records the seed carrier and finite compact-net, pointwise-modulus, dyadic tolerance, rational-fold, UniformModulus, and Real-seal handoff.

Constructive story: A $\mathsf{CompactModulusMeshHandoff}^{\uparrow}$ packet is generated from compact finite-net data, pointwise-modulus rows, dyadic tolerance arithmetic, a rational lower-bound fold, a UniformModulus handoff, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusMeshHandoff}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CompactModulusMeshHandoffUp`

Bridge status: none

Not claimed: This chapter does not close the full Heine–Cantor theorem, a compactness principle, a quotient function equality theorem, or a checked standard bridge.

Upgrade path: A scoped closure requires Lean-checked carrier obligations and uniform-modulus handoff stability for the displayed finite mesh route.

Chapter 8934

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedRealReflection}^{\uparrow}$ packet records the finite reflection surface by which located real data are read through the completion boundary and returned to the terminal real-name seal without quotienting the name. It sits over the located-real carrier of chapter 1933, the apartness-real route of chapter 304, the completion-reflection row of chapter 403, the stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a non-quotient reflection route for L10 Real and Completion consumers; it does not identify host real values, choose representatives, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8934.1 (Bishop located real reflection carrier)**.** *A* $\mathsf{BishopLocatedRealReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, F, S, Q, D, R, U, V, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{LocatedReal}^{\uparrow}$ *source row, exposing only the located real-name observations admitted by the packet.*
2. *B is the* $\mathsf{ApartnessReal}^{\uparrow}$ *comparison row used when reflection consumers request positive separation data.*
3. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *boundary row, recording that located data are read through the displayed completion-reflection surface.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which all source observations are scheduled.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds the located and reflected reads.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, F, S, Q, D remain visible.*
8. *U is the unit row, recording the displayed entry of located real data into the reflection route.*
9. *V is the counit row, recording the displayed return from the completion-reflection boundary to the Real seal route.*
10. *H records componentwise* hsame *transport for the located, apartness, reflection, stream, readback, tolerance, seal, unit, counit, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$A, B, F, S, Q, D \longmapsto R, U, V, H, P, N.$$

12. *P is the* Pkg *provenance over* BishopLocatedRealReflection$^{\uparrow}$, LocatedReal$^{\uparrow}$, ApartnessReal$^{\uparrow}$, CompletionReflection$^{\uparrow}$ StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* NameCert$_{\text{BishopLocatedRealReflection}^{\uparrow}}$ *row.*

The accepted carrier is exactly the displayed packet L. It has no coordinate for quotient stream equality, selected completed points, host real equality, countable choice, ambient Real$^{\uparrow}$ *completeness, or an external reflection theorem.*

Theorem 8934.2 (Bishop located real reflection NameCert obligations)**.** *For every accepted* BishopLocatedRealReflection$^{\uparrow}$ *carrier $L = (A, B, F, S, Q, D, R, U, V, H, C, P, N)$, the local* NameCert$_{\text{BishopLocatedRealRe}}$ *obligations are exhausted by carrier admission for the displayed located-real, apartness-real, completion-reflection, stream-window, regular-readback, dyadic-tolerance, Real-seal, unit, counit, transport, replay, provenance, and local naming rows. A consumer can therefore read the reflection packet only through*

$$A, B, F, S, Q, D, R, U, V, H, C, P, N.$$

No local obligation may call on quotient stream equality, a selected limit, host real equality, countable choice, ambient Real$^{\uparrow}$ *completeness, or a detached reflection principle.*

Proof. Project the carrier of Definition 8934.1. The accepted tuple displays exactly the located-real source, apartness comparison, completion-reflection boundary, stream-window, regular-readback, dyadic-tolerance, seal, unit, counit, transport, replay, provenance, and local naming rows.

Read the analytic rows first. The located row A and apartness row B are scheduled through S, interpreted by the regular readback row Q, and bounded by the dyadic tolerance row D before the terminal seal R is public.

The reflection row F and the unit-counit rows U, V are attached to that same finite route. Componentwise transport in H, replay in C, provenance in P, and local naming in N preserve the displayed carrier, so no excluded completion, equality, choice, or host-side principle is exported. □

Theorem 8934.3 (Bishop located real reflection unit-counit route)**.** *Let $L = (A, B, F, S, Q, D, R, U, V, H, C, P, N)$ be an accepted* BishopLocatedRealReflection$^{\uparrow}$ *carrier. Every unit-counit read enters through the* LocatedReal$^{\uparrow}$ *row A, the* StreamName$^{\uparrow}$ *window S, the* RegSeqRat$^{\uparrow}$ *readback Q, and the* DyadicRatCore$^{\uparrow}$ *tolerance row D, then passes through the* CompletionReflection$^{\uparrow}$ *row F before the counit row V returns to the terminal* Real$^{\uparrow}$ *seal R. The route remains downstream of B, U, H, C, P, N and exports no quotient completion equality, selected completed point, host real equality, countable-choice schedule, or ambient completeness theorem.*

Proof. Start with the source route. The located-real row A supplies the displayed name observations, while S, Q, D expose the finite windows, regular rational readback, and dyadic tolerances needed before a Real-facing consumer can inspect a value.

Next read the reflection boundary. The row F is the only completion-reflection coordinate in the packet, and the unit row U records entry into that route without choosing a representative or completed point.

Finally consume the counit row V and terminal seal R. The apartness row B remains a displayed comparison coordinate, and H, C, P, N preserve the same finite route by transport, replay, provenance, and naming. Since the carrier has no excluded coordinate, the unit-counit read cannot become quotient equality, selected completion data, host equality, choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealReflection}^{\uparrow}$ packet is generated from LocatedReal source rows, ApartnessReal comparison rows, CompletionReflection boundary rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, unit and counit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8934.1, Definition 8934.2, and Definition 8934.3 seed the finite located-real reflection packet over located, apartness, completion-reflection, stream, readback, tolerance, seal, unit, counit, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completed limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-counit routing, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealReflectionUp.

Chapter 8935

A Concrete Naming Certificate for $\mathsf{RealNameEliminator}^{\uparrow}$

Following Bishop–Bridges constructive real-number practice, the $\mathsf{RealNameEliminator}^{\uparrow}$ packet records the finite route by which a consumer may use a $\mathsf{Real}^{\uparrow}$ seal only through certified regular rational representatives. It sits over the dyadic arithmetic surface of chapter 291, the finite observation discipline of chapter 126, the regular-sequence readback of chapter 301, the completed real seal of chapter 38, and the modulus witness of chapter 292. The packet supplies a representative-consuming eliminator, not quotient equality, host real primitives, selected limits, or ambient real completeness.

Falsifiable prediction: A $\mathsf{Real}^{\uparrow}$ consumer that reads a completed real without displaying a certified $\mathsf{RegSeqRat}^{\uparrow}$ representative refutes the $\mathsf{RealNameEliminator}^{\uparrow}$ carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchymodulus: $\mathsf{RealNameEliminator}^{\uparrow}$ is not bijective to any one sibling because it binds dyadic tolerance rows, finite stream windows, regular-sequence representatives, the Real seal, and modulus transport into one consumer-facing elimination packet.

Definition 8935.1 (Real-name eliminator carrier)**.** *A* $\mathsf{RealNameEliminator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, Q, R, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger used by the representative;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the observations a consumer may inspect;*
3. *Q is the certified* $\mathsf{RegSeqRat}^{\uparrow}$ *representative readback served on those windows;*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, S, Q and the modulus data have been displayed;*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row transporting precision requests back to finite representative windows;*
6. *H records componentwise* hsame *transport for the analytic and structural rows;*
7. *C records displayed* Cont *replay from representative admission to Real consumption;*

8. *P is the* Pkg *provenance over* $\mathsf{RealNameEliminator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealNameEliminator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, R, M, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, selected representatives outside Q, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8935.2 (Real-name eliminator NameCert obligations)**.** *For every accepted* $\mathsf{RealNameEliminator}^{\uparrow}$ *carrier*

$$E = (D, S, Q, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealNameEliminator}^{\uparrow}}$ *surface consists exactly of carrier exposure for E, dyadic tolerance and comparison exposure through D, finite window exposure through S, certified regular-rational representative readback through Q, Real seal exposure through R, modulus transport through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8935.1. The displayed tuple lists precisely the dyadic ledger, stream window, regular representative, Real seal, modulus row, and structural rows.

Read the analytic rows in dependency order. The representative row Q is attached to the finite windows S and the dyadic ledger D, while the modulus row M transports later precision requests back to those displayed windows before the Real seal row R is consumed.

Finally inspect the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further coordinate, the listed rows exhaust the local naming surface. □

Theorem 8935.3 (Real-name eliminator RegSeqRat determinacy)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{RealNameEliminator}^{\uparrow}$ *packet must factor through the certified* $\mathsf{RegSeqRat}^{\uparrow}$ *representative row Q and the finite route*

$$D, S, Q, R, M, H, C, P, N.$$

Thus two accepted consumers that agree on D, S, Q, M, H, C, P, N have the same local Real-name elimination surface, and any consumer that claims host real equality, quotient stream equality, a selected representative outside Q, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness has left the packet.*

Proof. Start with the NameCert obligations of Definition 8935.2. They place the regular representative row Q between the finite stream windows S and the terminal Real seal row R.

Use the modulus row M to replay every later precision request through the same representative windows. Agreement on D, S, Q, M therefore fixes the representative-consuming part of the eliminator before any structural transport is read.

The structural rows H, C, P, N preserve that same packet under componentwise transport, continuation replay, provenance, and naming. The excluded objects are absent from the carrier, so a read of any such object cannot be an accepted $\mathsf{RealNameEliminator}^{\uparrow}$ consumption. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealNameEliminator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealNameEliminator}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat representative readback, a Real seal row, CauchyModulus precision transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealNameEliminator}^{\uparrow}}$, seed)

Scope closed: Definition 8935.1, Definition 8935.2, and Definition 8935.3 declare the representative-consuming Real-name eliminator surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealNameEliminator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected representatives outside the displayed RegSeqRat row, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representative determinacy, modulus transport, Real-seal nonescape, and structural transport for BEDC.Derived.RealNameEliminatorUp.

Chapter 8936

A Concrete Naming Certificate for $\mathsf{RiemannMappingBoundary}^{\uparrow}$

Following Caratheodory and Pommerenke, the $\mathsf{RiemannMappingBoundary}^{\uparrow}$ packet records the finite boundary-extension surface attached to a Riemann mapping whose source boundary is displayed as a Jordan curve. It sits after the interior mapping route of chapter 6805, the Jordan-curve boundary source of chapter 4365, the circle carrier of chapter 58, the compact metric windows of chapter 129, and the continuous graph rows of chapter 130. The packet names boundary extension only as displayed NameCert data; it does not import prime-end quotients, selected boundary correspondences, host homeomorphism equality, or ambient compact-open completeness.

Definition 8936.1 (Riemann mapping boundary carrier)**.** *A* $\mathsf{RiemannMappingBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, J, S, K, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the existing* $\mathsf{RiemannMapping}^{\uparrow}$ *interior packet, read through the finite uniformization route of chapter 6805.*
2. *J is the* $\mathsf{JordanCurve}^{\uparrow}$ *boundary-source row, carrying the displayed simple closed curve, side classifier, polygonal subdivision, and orientation ledger.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *boundary-circle row whose finite parameter windows supply the source boundary schedule.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-boundary-window row, recording only compact metric inspection windows along the displayed curve and circle source.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *boundary-graph row, exposing the proposed boundary extension as finite graph and modulus data over the already displayed boundary windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coordinate and modulus observations produced by the boundary graph.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for distances, radii, and continuity budgets attached to Q.*
8. *H records componentwise* hsame *transport for the interior packet, Jordan boundary source, circle boundary source, compact windows, boundary graph, readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay, forcing a consumer to read the interior mapping, boundary source, boundary circle, compact windows, graph, readback, and Real seal in the displayed order.*
10. *P is the* Pkg *provenance over* $\mathsf{RiemannMappingBoundary}^{\uparrow}$, $\mathsf{RiemannMapping}^{\uparrow}$, $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannMappingBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, J, S, K, F, Q, E, H, C, P, N$. It has no coordinate for a prime-end quotient, selected boundary homeomorphism, host equality of continuous extensions, ambient compact-open completion, or a boundary map detached from the displayed Jordan and circle rows.

Lean4 (statement) `BEDC.Derived.RiemannMappingBoundaryUp`

Falsifiable prediction: A purported $\mathsf{RiemannMappingBoundary}^{\uparrow}$ consumer that reads boundary extension without exposing the $\mathsf{RiemannMapping}^{\uparrow}$ interior packet, the $\mathsf{JordanCurve}^{\uparrow}$ source boundary, the $\mathsf{S}^{1\uparrow}$ boundary circle, the $\mathsf{CompactMetric}^{\uparrow}$ finite windows, the $\mathsf{ContinuousMap}^{\uparrow}$ graph, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal is outside the carrier surface.

Independence witness: riemann-mapping, jordan-curve, s1, compactmetric, continuousmap, regseqrat, real: no listed sibling supplies the joint carrier containing the interior mapping packet, Jordan boundary source, circle boundary source, compact boundary windows, boundary graph, regular rational readback, Real seal, transport, replay, provenance, and local NameCert rows while refusing prime-end quotients, selected boundary homeomorphisms, and ambient compact-open completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannMappingBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannMappingBoundary}^{\uparrow}$ packet is generated from a RiemannMappingUp interior row, JordanCurveUp boundary-source row, SOneUp boundary-circle row, CompactMetricUp finite-boundary windows, ContinuousMapUp boundary graph, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannMappingBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 8936.1 seeds the finite boundary-extension carrier over RiemannMappingUp, JordanCurveUp, SOneUp, CompactMetricUp, ContinuousMapUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannMappingBoundary}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, prime-end quotient construction, selected boundary homeomorphisms, host equality of continuous extensions, or ambient compact-open completeness.

Upgrade path: Obligation closure requires checked carrier admission, Jordan boundary exposure, circle parameter exactness, compact-window routing, graph readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RiemannMappingBoundaryUp.

Chapter 8937

A Concrete Naming Certificate for $\mathsf{RegularLocatedCut}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularLocatedCut}^{\uparrow}$ packet records located-cut boundary data as a finite comparison surface for the regular-sequence route to constructive real numbers. It reads dyadic comparison rows through chapter 291, regular rational representatives through chapter 301, located-real boundary language through **??**, and the completed Real seal through chapter 38. The packet is a comparison object for the Real and Completion phase, not a construction of $\mathsf{Real}^{\uparrow}$ by Dedekind cuts.

Falsifiable prediction: A located-cut consumer that compares against $\mathsf{RegSeqRat}^{\uparrow}$ without displaying dyadic brackets and regularity-modulus data is not an accepted $\mathsf{RegularLocatedCut}^{\uparrow}$ packet.

Independence witness: dyadicratcore, regseqrat, locatedreal, real: $\mathsf{RegularLocatedCut}^{\uparrow}$ is not bijective to any one sibling because it binds lower and upper cut rows, dyadic brackets, regularity modulus data, transport, continuation replay, provenance, and local naming into one comparison packet.

Definition 8937.1 (Regular located-cut carrier)**.** *A* $\mathsf{RegularLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, D, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower cut row, listing the accepted rational lower approximants;*
2. *B is the upper cut row, listing the accepted rational upper approximants;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket row that records finite dyadic comparisons between the lower and upper sides;*
4. *M is the regularity modulus row that routes precision requests to a displayed bracket refinement;*
5. *H records componentwise* hsame *transport for the lower row, upper row, dyadic bracket row, modulus row, replay row, provenance row, and naming row;*
6. *C records displayed* Cont *replay for extending a bracket comparison without changing the finite carrier surface;*
7. *P is the* Pkg *provenance over* $\mathsf{RegularLocatedCut}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*

8. N is the local $\mathsf{NameCert}_{\mathsf{RegularLocatedCut}^\uparrow}$ *row.*
The classifier compares accepted packets only through A, B, D, M, H, C, P, N. It has no coordinate for a host Dedekind cut, quotient equality of cuts, a selected Real representative, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8937.2 (Regular located-cut NameCert obligations)**.** *For every accepted* $\mathsf{RegularLocatedCut}^\uparrow$ *carrier*

$$L = (A, B, D, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularLocatedCut}^\uparrow}$ *surface consists exactly of lower cut exposure through A, upper cut exposure through B, dyadic bracket exposure through D, regularity-modulus exposure through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8937.1. The displayed tuple lists exactly the lower side, upper side, dyadic bracket, regularity modulus, and structural rows.

Read the analytic rows in comparison order. The lower and upper rows A, B are related only through the finite dyadic brackets D, and the modulus row M sends later precision requests back to displayed bracket refinements.

Inspect the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite packet, so no additional cut object or Real-completeness principle can enter the local naming surface. □

Theorem 8937.3 (Regular located-cut RegSeqRat comparison boundary)**.** *Every comparison between an accepted* $\mathsf{RegularLocatedCut}^\uparrow$ *packet and the regular-sequence route to* $\mathsf{Real}^\uparrow$ *factors through the finite rows*

$$A, B, D, M, H, C, P, N.$$

The rows A and B provide located lower and upper approximants, D supplies dyadic brackets, and M supplies the regularity modulus needed to compare with $\mathsf{RegSeqRat}^\uparrow$ *representatives. This is a comparison boundary between located-cut language and the* $\mathsf{RegSeqRat}^\uparrow/\mathsf{Real}^\uparrow$ *route, not a Dedekind-cut construction of* $\mathsf{Real}^\uparrow$*.*

Proof. Start from the NameCert obligations of Definition 8937.2. They place every located lower and upper read inside the rows A, B and require dyadic bracket evidence through D.

Use the modulus row M to align a precision request with a finite bracket refinement. A comparison with a $\mathsf{RegSeqRat}^\uparrow$ representative therefore reads the displayed bracket data rather than selecting an ambient cut or a host limit.

The structural rows H, C, P, N preserve only that comparison route under transport, continuation replay, provenance, and naming. Since the carrier has no coordinate for a Dedekind construction of $\mathsf{Real}^\uparrow$, the theorem records a boundary between languages rather than an alternate construction of the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularLocatedCut}^\uparrow}$**.**

Constructive story: A $\mathsf{RegularLocatedCut}^\uparrow$ packet is generated from lower cut rows, upper cut rows, dyadic bracket rows, regularity-modulus rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularLocatedCut}^\uparrow}$, seed)

Scope closed: Definition 8937.1, Definition 8937.2, and Definition 8937.3 declare the located-cut comparison boundary for the RegSeqRat and Real route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularLocatedCut}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient equality of cuts, selected Real representatives, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper bracket obligations, regularity-modulus transport, RegSeqRat comparison boundary, and structural transport for BEDC.Derived.RegularLocatedCutUp.

Chapter 8938

A Concrete Naming Certificate for $\mathsf{SpectralProjection}^{\uparrow}$

Following Horn and Johnson's finite-dimensional matrix analysis, the packet records spectral projectors as displayed polynomial idempotent rows over a matrix carrier. It sits after the finite matrix carrier of chapter 49, the displayed polynomial surface of chapter 50, the field-extension rows of chapter 161, the vector-space carrier of chapter 47, and the finite spectral-radius boundary of chapter 2352. The chapter names only the finite polynomial projector packet; it does not import host spectral calculus, contour integrals, quotient eigenspaces, selected eigenbases, or analytic completeness.

Definition 8938.1 (Spectral projection carrier)**.** *A* $\mathsf{SpectralProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, P, L, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *source row, including the finite square matrix and its entry carrier.*
2. *P is the displayed* $\mathsf{Poly}^{\uparrow}$ *projector row whose coefficients are read over the same scalar field as M.*
3. *L is the eigenvalue or factor row, exposed through* $\mathsf{FieldExt}^{\uparrow}$ *when the projector is displayed after scalar extension.*
4. *I is the idempotence row recording that the evaluated polynomial projector satisfies $P(M)P(M) = P(M)$ inside the displayed matrix carrier.*
5. *R is the range/kernel visibility row: it records the visible image boundary, the visible kernel boundary, and the finite* $\mathsf{VecSpace}^{\uparrow}$ *rows consumed by downstream invariant-subspace readers.*
6. *H records componentwise* hsame *transport for the matrix source, polynomial projector, eigenvalue or factor row, idempotence row, range/kernel visibility, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a spectral-projection request through M, P, L, I, R, H, Q, N.*
8. *Q is the* Pkg *provenance over* $\mathsf{SpectralProjection}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, P, L, I, R, H, C, Q, N$. It has no coordinate for a host functional calculus, an implicit contour integral, a quotient eigenspace, a selected eigenbasis, or a completed spectral decomposition.

Theorem 8938.2 (Spectral projection NameCert obligations)**.** *For every accepted* $\mathsf{SpectralProjection}^{\uparrow}$ *carrier*

$$S = (M, P, L, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralProjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed matrix source, polynomial projector, eigenvalue or factor row, idempotence row, range/kernel visibility row, componentwise transport, displayed replay, package provenance, and local naming. Any consumer that reads a spectral projector must expose the matrix-polynomial route, the field-extension boundary when used, the finite vector-space visibility row, and the spectral-radius boundary before it may consume the projector as a named finite row.*

Proof. Project the carrier of Definition 8938.1. The matrix and polynomial inputs are M, P, the scalar or factor boundary is L, the idempotence witness is I, and the visible range/kernel boundary is R.

The transport and replay rows H, C preserve exactly those displayed rows. The provenance row Q records the consumed MatrixUp, PolynomialUp, FieldExtUp, VecSpaceUp, and SpectralRadiusUp siblings, while N names the local packet.

No additional spectral object is available to the classifier. Hence the NameCert obligations are exhausted by the displayed carrier rows. □

Theorem 8938.3 (Spectral projection matrix-polynomial handoff)**.** *Every accepted* $\mathsf{SpectralProjection}^{\uparrow}$ *packet exports its projector only through the finite handoff*

$$\mathsf{Mat}^{\uparrow} \times \mathsf{Poly}^{\uparrow} \times \mathsf{FieldExt}^{\uparrow} \to \mathsf{IdempotentMatrixRow} \to \mathsf{VecSpace}^{\uparrow}.$$

The $\mathsf{SpectralRadius}^{\uparrow}$ *row can bound the finite matrix side of the handoff, but it cannot replace the displayed polynomial projector, idempotence row, or range/kernel visibility row.*

Proof. Read the handoff from the carrier rows. The matrix source and polynomial projector produce the displayed matrix row $P(M)$, while the field-extension row records the scalar boundary used to display the eigenvalue or factor data.

The idempotence row I is the only row that lets the packet act as a projector. The range/kernel visibility row R is the only row exported to vector-space consumers.

The spectral-radius row appears only through provenance and finite matrix-side estimates. It therefore cannot supply the projector, the idempotence witness, or the visible subspace boundary by itself. □

Falsifiable prediction: Every accepted $\mathsf{SpectralProjection}^{\uparrow}$ consumer displays Matrix source rows, Polynomial projector rows, FieldExt eigenvalue or factor rows when used, idempotence rows, range/kernel visibility rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before it can consume a spectral projector.

Independence witness: matrix, polynomial, fieldext, vecspace, spectralradius: $\mathsf{SpectralProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a finite matrix source, a displayed polynomial projector, eigenvalue or factor exposure, idempotence, visible range/kernel rows, transport, replay, provenance, and local naming as one projector packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralProjection}^{\uparrow}}$**.**

Constructive story: A SpectralProjection$^{\uparrow}$ packet is generated from a MatrixUp source row, a PolynomialUp projector row, a FieldExtUp eigenvalue or factor row when used, an idempotence row, VecSpaceUp range/kernel visibility rows, a SpectralRadiusUp finite matrix-side boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SpectralProjection}^{\uparrow}}$, seed)

Scope closed: Definition 8938.1, Definition 8938.2, and Definition 8938.3 seed the finite polynomial projector interface over MatrixUp, PolynomialUp, FieldExtUp, VecSpaceUp, SpectralRadiusUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{SpectralProjection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host spectral calculus, contour integrals, quotient eigenspaces, selected eigenbases, analytic completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial-projector exactness, idempotence exactness, range/kernel visibility, matrix-polynomial handoff, transport, replay, provenance, and local naming for BEDC.Derived.SpectralProjectionUp.

Chapter 8939

A Concrete Naming Certificate for $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$

Following Bishop–Bridges regular Cauchy completion, Kelley filter language, and Bourbaki uniform-space bookkeeping, the $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$ packet records the finite route by which a displayed Cauchy-filter presentation is read as a regular Cauchy rational name before any Real or Completion consumer may use it. It sits after the Cauchy-filter surface of chapter 333, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a handoff from filter data to regular Cauchy names; it does not quotient filters, select a limit, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8939.1 (Cauchy regularization functor carrier)**.** *A* $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, W, R, D, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the source row, either a* $\mathsf{CauchyFilter}^{\uparrow}$ *presentation or the refined* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *source whose finite filter requests are being regularized.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite request schedule used to inspect the source filter.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the source windows, recording the regular rational name seen by downstream consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic precision, radius, comparison, and monotone budget rows used by the extraction.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after A, W, R, D have been displayed.*
6. *H is the componentwise* hsame *transport row for the source, window, readback, tolerance ledger, real seal, provenance, and local name.*
7. *K is the displayed* Cont *replay row, routing each accepted filter request through W, D, R before the real seal is exposed.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRegularizationFunctor}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of filters, a selected limit point, a countable-choice schedule, host real equality, or ambient completion of $\mathsf{Real}^{\uparrow}$.

Theorem 8939.2 (Cauchy regularization functor obligation surface)**.** *For every accepted* CauchyRegularizationFunc *carrier $F = (A, W, R, D, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRegularizationFunctor}^{\uparrow}}$ *surface consists of the following finite rows:*

1. *source admission, requiring every input request to enter through the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *row A;*
2. *window discipline, requiring every inspected request to be listed in the* $\mathsf{StreamName}^{\uparrow}$ *row W before it contributes to a regular name;*
3. *dyadic regularity, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D to supply the precision and tolerance budget for the readback;*
4. *regular rational extraction, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be produced from A, W, D before any* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
5. *real-seal nonselection, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to be consumed only as a terminal seal over the displayed regular name;*
6. *structural replay, requiring H, K, P, N to transport, replay, source, and locally name exactly the displayed packet.*

No obligation row exports quotient filters, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 8939.1. The source row is A, the finite inspection row is W, the tolerance row is D, the regular rational readback is R, and the terminal seal is E.

The NameCert obligation reads those rows in order. A filter request cannot reach R until it has been scheduled by W and budgeted by D, and the real seal E is not available before the regular readback row is displayed.

Finally read the structural rows. The rows H, K, P, N only transport, replay, source, and name the same packet, so they cannot add quotient filters, selected limits, countable choice, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRegularizationFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$ packet is generated from a displayed CauchyFilter or CauchyFilterRegularization source, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRegularizationFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 8939.1, Definition 8939.2, and Definition 8939.3 seed the finite L10 handoff from Cauchy-filter presentations to regular rational real-name data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRegularizationFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, window discipline, dyadic regularity, regular rational extraction, Real-seal nonselection, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRegularizationFunctorUp.

Theorem 8939.3 (Cauchy regularization functor L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CauchyRegularizationFunctor}^{\uparrow}$ *packet reaches the terminal Real boundary*

only through the displayed route

$$\mathsf{CauchyFilter}^{\uparrow} \textit{ or } \mathsf{CauchyFilterRegularization}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Equivalently, the handoff from filter presentations to regular Cauchy names is exposed as finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. A consumer asking for quotient filter equality, selected limits, countable choice, host real equality, or ambient completion has no row in the carrier.*

Proof. Start with the source row A. By the obligation surface of Definition 8939.2, each accepted request enters through the Cauchy-filter source and is then routed into the finite window W.

Next read the dyadic ledger and regular rational row. The ledger D supplies the tolerance budget for every scheduled request, and the readback R is the only regular Cauchy name exposed to downstream consumers.

The terminal seal E is downstream from A, W, D, R. Since the carrier has no coordinate for quotient filters, selected limits, countable choice, host equality, or ambient completeness, Real and Completion consumers can use only the displayed L10 route. □

Chapter 8940

A Concrete Naming Certificate for $\mathsf{EndpointBisectionCertificate}^{\uparrow}$

The $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ packet realizes the endpoint-bisection certificate of section 9413.61 as finite NameCert data over RHRoute endpoint intervals, branch choices, preservation rows, and local half-split handoff. It reads the endpoint interval and branch replay surface encoded by BEDC.Derived.RHRoute.EndpointBisectionCertificate, hands the local split boundary to chapter 8881, and keeps the endpoint route separate from parity tower, oriented triadic, and zero-recursor cover routes. The packet names a finite endpoint bisection surface; it is not an RH proof, not a zero selector, and not a quotient over all endpoint models.

Definition 8940.1 (Endpoint bisection certificate carrier)**.** *An* $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, \beta, R, S, L, H, K, P, N)$$

with the following displayed rows:

1. *I is the RHRoute endpoint-interval row, carrying an endpoint dyadic interval and its finite diameter data.*
2. *β is the branch-choice row, listing the finite lower/upper decisions used by endpoint replay.*
3. *R is the replay row, computing the endpoint interval reached by the displayed branch list.*
4. *S is the preservation row, recording the depth, width, and endpoint-location invariants kept by replay.*
5. *L is the local half-split handoff row, available only after I, β, R, S have been displayed.*
6. *H stores componentwise* hsame *transport for the endpoint interval, branch list, preservation row, local half-split row, provenance, and local name.*
7. *K stores displayed* Cont *replay from endpoint reads through the branch and preservation rows before any RHRoute conclusion is consumed.*
8. *P records* Pkg *provenance over* $\mathsf{EndpointBisectionCertificate}^{\uparrow}$*, RHRoute endpoint intervals, branch choices, preservation rows,* $\mathsf{LocalHalfSplitCertificate}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EndpointBisectionCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global RH conclusion, a selected zero, a quotient over endpoint models, parity tower equality, oriented triadic equivalence, or zero-recursor cover boundary replacement.

Theorem 8940.2 (Endpoint bisection certificate NameCert surface)**.** *For every accepted* EndpointBisectionCertifica *carrier* $B = (I, \beta, R, S, L, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EndpointBisectionCertificate}^{\uparrow}}$ *surface consists exactly of endpoint interval admission, branch-choice admission, replay determinacy, preservation visibility, local half-split handoff, componentwise transport, displayed continuation replay, provenance, and local naming. Any endpoint-bisection read that omits one of* $I, \beta, R, S, L, H, K, P, N$ *is outside the carrier surface.*

Proof. Project the carrier of Definition 8940.1. The analytic rows are the endpoint interval I, branch list β, replay row R, preservation row S, and local half-split handoff L.

Read the rows in dependency order. Replay depends on the displayed branch list, preservation is recorded only for the replayed interval, and the local half-split handoff is available only after replay and preservation have both been displayed.

The structural rows H, K, P, N transport, replay, source, and name the same packet. Since no coordinate carries a global RH conclusion, selected zero, quotient endpoint model, parity tower equality, oriented triadic equivalence, or zero-recursor cover boundary replacement, the listed rows exhaust the local NameCert surface. □

Theorem 8940.3 (Endpoint bisection certificate RHRoute handoff)**.** *Every RHRoute consumer of an accepted* $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ *packet reaches an endpoint-facing conclusion only through endpoint interval, branch, preservation, local half-split,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. In carrier notation the route is*

$$I, \beta, R, S, L, H, K, P, N.$$

Thus the packet can hand endpoint-bisection data to RHRoute readers, but it does not by itself assert the Riemann hypothesis, select a zero, identify all endpoint models, or collapse the local half-split route into parity, triadic, or recursor-cover siblings.

Proof. Start with the NameCert surface of Definition 8940.2. It forces any accepted endpoint read to expose I, β, R, S, L before a route consumer can read the packet.

Next use the structural rows. Transport and replay keep the endpoint interval, branch list, preservation row, and local half-split row together as one finite packet, while provenance and local naming prevent the handoff from detaching into an untracked RHRoute conclusion.

The carrier omits global RH conclusions, zero selectors, quotient endpoint models, parity tower equality, oriented triadic equivalence, and zero-recursor cover replacement. Therefore every accepted RHRoute handoff remains the displayed finite endpoint-bisection route. □

Falsifiable prediction: Every accepted endpoint-bisection read factors through endpoint interval, branch, preservation, local half-split, hsame, Cont, Pkg, and NameCert rows before any RHRoute conclusion is consumed.

Independence witness: local-half-split-certificate, parity-closure-tower-certificate, oriented-triadic-certificate, zero-recursor-cover-boundary: none is bijective to $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ because the local half-split sibling supplies only the split handoff, the parity tower route supplies parity closure data, the oriented triadic route supplies triadic orientation rows, and the zero-recursor cover boundary supplies recursor-cover rows, while this packet binds endpoint interval replay, branch choices, preservation, local half-split handoff, transport, replay, provenance, and local naming in one carrier.

The future TasteGate surface for $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ must supply nontrivial endpoint-branch variation and field-faithful visibility for $I, \beta, R, S, L, H, K, P, N$ before the chapter can be promoted beyond this paper-side NameCert target.

Closure status for $\mathsf{NameCert}_{\mathsf{EndpointBisectionCertificate}^{\uparrow}}$.

Constructive story: An $\mathsf{EndpointBisectionCertificate}^{\uparrow}$ packet is generated from RHRoute endpoint interval rows, branch-choice rows, endpoint replay, preservation rows, a local half-split handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EndpointBisectionCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8940.1, Definition 8940.2, and Definition 8940.3 seed the endpoint-bisection NameCert route over RHRoute endpoint intervals and local half-split handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EndpointBisectionCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the Riemann hypothesis, zero selection, quotient endpoint-model equality, parity tower equivalence, oriented triadic equivalence, zero-recursor cover replacement, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint interval admission, branch replay, preservation visibility, local half-split handoff, nontrivial endpoint variation, field-faithful row visibility, transport, replay, provenance, and local naming for BEDC.Derived.RHRoute.EndpointBisectionCertificate.

Chapter 8941

A Concrete Naming Certificate for $\mathsf{FiniteDyadicStaircase}^{\uparrow}$

Following Bishop–Bridges constructive integration and the Darboux use of finite partitions, the $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ packet records a finite L10 staircase over dyadic cells. It refines the partition surface of chapter 1863, reads cell geometry through chapter 5966, hands intervalwise bounds to chapter 1880, keeps rational values on the regular route of chapter 301, and exposes any terminal value-facing seal only through chapter 38. The packet is a finite approximation object; it does not import open-cover compactness, selected mesh limits, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: A consumer that reads a finite dyadic staircase value without the partition row, interval-net cell row, Darboux-sum handoff row, RegSeqRat readback row, and terminal Real-facing seal has left the $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ packet.

Independence witness: dyadic partition, dyadic interval net, darboux sum, regseqrat, real: $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ is not bijective to any one sibling because it jointly carries a finite partition, cellwise rational heights, Darboux-facing interval ledgers, regular rational readback, terminal sealing, transport, replay, provenance, and local naming.

Definition 8941.1 (Finite dyadic staircase carrier)**.** *A* $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, I, V, L, U, D, R, E, H, C, G, N)$$

with the following displayed rows:

1. *P is the accepted* $\mathsf{DyadicPartition}^{\uparrow}$ *row for the finite ordered dyadic partition;*
2. *I is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cell-net row exposing the inspected dyadic cells and their endpoint ledgers;*
3. *V is the finite value row assigning a* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback name to each displayed cell;*
4. *L and U are the lower and upper cell-bound ledgers exported to the Darboux route;*
5. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, reading exactly the partition row, interval-net row, value row, and bound ledgers;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite staircase values and Darboux ledgers;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for value-facing consumers that are allowed to read a real name;*

8. *H records componentwise* hsame *transport for the partition, cell-net, value, lower-bound, upper-bound, Darboux, readback, seal, replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay from a staircase request through $P, I, V, L, U, D, R, E, H, G, N$;*
10. *G is the* Pkg *provenance over* $\mathsf{FiniteDyadicStaircase}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDyadicStaircase}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an open cover, a selected limiting mesh, quotient real equality, countable choice, or ambient real completeness.

Theorem 8941.2 (Finite dyadic staircase NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ *carrier*

$$S = (P, I, V, L, U, D, R, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDyadicStaircase}^{\uparrow}}$ *surface consists exactly of partition admission through P, dyadic cell-net exposure through I, finite rational value exposure through V, lower and upper Darboux ledgers through L, U, Darboux handoff through D, regular rational readback through R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8941.1. The partition row, interval-net row, finite value row, lower and upper bound ledgers, Darboux handoff, readback row, and Real seal are visible before the structural rows are read.

The finite partition route is fixed by P, I, V. A downstream consumer can inspect a cell only through the displayed dyadic partition and interval-net rows, and can inspect a height only through the finite RegSeqRat value row attached to that cell.

The Darboux route is fixed by L, U, D, R, E. The lower and upper ledgers feed the Darboux handoff row, the regular rational row reads the finite sums and heights, and the Real seal exposes only the terminal value-facing name.

The rows H, C, G, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for the excluded host principles, the listed rows exhaust the local NameCert surface. □

Theorem 8941.3 (Finite dyadic staircase Darboux handoff)**.** *Every Darboux-facing consumer of an accepted* $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ *packet must pass through*

$$P, I, V, L, U, D, R, E, H, C, G, N.$$

The consumer receives a finite dyadic partition, displayed interval-net cells, cellwise regular rational values, lower and upper Darboux ledgers, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal. It receives no open-cover compactness theorem, selected mesh-limit object, quotient real equality, countable-choice row, or ambient completeness principle.*

Proof. Start with Definition 8941.2. It identifies the displayed rows as the whole local NameCert surface and gives no second source for partition, cell, value, Darboux, readback, or seal data.

The Darboux-facing evidence is carried by P, I, V, L, U, D. The partition and interval-net rows display the cells, the value row supplies the cellwise rational names, and the lower and upper ledgers are the only rows exported to the Darboux handoff.

The value route is fixed by R, E. A consumer that wants a value-facing endpoint first reads the finite regular rational data and only then reaches the terminal Real seal.

Therefore the Darboux handoff cannot switch to open-cover compactness, selected mesh limits, quotient real equality, countable choice, or ambient completeness. Those objects are absent from the carrier and from the local classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicStaircase}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDyadicStaircase}^{\uparrow}$ packet is generated from a finite dyadic partition row, a dyadic interval-net cell row, finite RegSeqRat value rows, lower and upper Darboux ledgers, a Darboux handoff row, regular rational readback, a terminal Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicStaircase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8941.1, Definition 8941.2, and Definition 8941.3 seed the finite L10 staircase carrier from dyadic partitions to Darboux-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDyadicStaircase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected mesh limits, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Darboux handoff exactness, componentwise transport, replay, provenance, terminal sealing, and local naming for BEDC.Derived.FiniteDyadicStaircaseUp.

Chapter 8942

A Concrete Naming Certificate for $\mathsf{DirectedCauchyNet}^{\uparrow}$

Following Moore and Smith's theory of nets and Kelley's topology text, the $\mathsf{DirectedCauchyNet}^{\uparrow}$ packet records a directed Cauchy net as finite BEDC completion data. It sits after the directed-index substrate of chapter 1607, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a net-style L10 completion neighbour for the sequence-based $\mathsf{Real}^{\uparrow}$ route; it does not choose a cofinal sequence, quotient nets, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: A completion consumer that reads a directed Cauchy net without the directed-index row, finite cofinal windows, StreamName readback, RegSeqRat handoff, DyadicRatCore tolerance ledger, and terminal Real seal has left the $\mathsf{DirectedCauchyNet}^{\uparrow}$ packet.

Independence witness: directedset, streamname, regseqrat, dyadicratcore, real: $\mathsf{DirectedCauchyNet}^{\uparrow}$ is not bijective to any one sibling because it jointly carries directed-window data, Cauchy-tail stability, sequence-style readback, dyadic tolerances, terminal sealing, transport, replay, provenance, and local naming.

Definition 8942.1 (Directed Cauchy net carrier)**.** *A* $\mathsf{DirectedCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (D, V, W, T, S, Q, Y, R, H, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *row naming the inspected directed vertices, comparison rows, finite request windows, and common upper-window witnesses;*
2. *V is the net-value row storing the finite values visible at the inspected directed vertices;*
3. *W is the finite cofinal-window row whose entries compare directed tails before any completion consumer reads a Real-facing seal;*
4. *T is the Cauchy-tail stability row, stating that every requested dyadic tolerance is witnessed after a displayed upper window;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row obtained by reading a retained cofinal window as a finite stream window;*

6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that records the regular rational readback of the retained window;*
7. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the tail-stability and regular readback rows;*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for completion-facing consumers allowed to read a real name;*
9. *H records componentwise* hsame *transport for the directed-index, value, cofinal-window, tail-stability, stream, regular, dyadic, Real, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from a directed request through $D, V, W, T, S, Q, Y, R, H, P, L$;*
11. *P is the* Pkg *provenance over* $\mathsf{DirectedCauchyNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows, with sibling anchors in chapter 1607, chapter 126, chapter 301, chapter 291, and chapter 38;*
12. *L is the local* $\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host net function, quotient of nets, selected cofinal sequence, selected limit, countable-choice schedule, or ambient completeness theorem.

Theorem 8942.2 (Directed Cauchy net NameCert obligations)**.** *For every accepted* $\mathsf{DirectedCauchyNet}^{\uparrow}$ *carrier*

$$N = (D, V, W, T, S, Q, Y, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$ *surface consists exactly of directed-index admission through D, finite value exposure through V, cofinal-window extraction through W, tail-stability evidence through T, StreamName readback through S, RegSeqRat handoff through Q, DyadicRatCore tolerance accounting through Y, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 8942.1. The directed-index row D, value row V, cofinal-window row W, and tail-stability row T are visible before the completion-facing readback rows.

The sequence-style readback is fixed by S, Q, Y, R. The retained window is read as a $\mathsf{StreamName}^{\uparrow}$ row, then as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff controlled by the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and only then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal.

The rows H, C, P, L transport, replay, source, and name the same packet through hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert surface. □

Theorem 8942.3 (Directed Cauchy net finite window extraction)**.** *Every completion-facing read of an accepted* $\mathsf{DirectedCauchyNet}^{\uparrow}$ *packet extracts a finite cofinal window from the displayed* $\mathsf{DirectedSet}^{\uparrow}$ *row before it may inspect* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *data. The extraction supplies only the finite window W and its common upper witnesses; it does not supply a selected cofinal sequence, quotient index, host net function, or countable-choice operator.*

Proof. Start with the directed substrate. The row D names the inspected directed vertices, comparison rows, and upper-window witnesses, and W records which finite windows are retained for completion readback.

The downstream rows cannot be read before this extraction. The $\mathsf{StreamName}^{\uparrow}$ row S, $\mathsf{RegSeqRat}^{\uparrow}$ handoff Q, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger Y, and $\mathsf{Real}^{\uparrow}$ seal R all appear after D, V, W, T in the carrier, and their provenance is routed through the sibling certificates named in Definition 8942.1.

Therefore a consumer receives finite cofinal-window data, not a host-level cofinal sequence or quotient of directed indices. The structural rows H, C, P, L preserve that same finite extraction boundary. □

Theorem 8942.4 (Directed Cauchy net tail stability)**.** *In an accepted* $\mathsf{DirectedCauchyNet}^{\uparrow}$ *carrier, every dyadic tolerance requested by the completion route is witnessed by a displayed upper window in* W *and by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* Y*. The resulting tail-stability row* T *is read through* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$ *row; it is not an external Cauchy principle over all directed indices.*

Proof. Read the tail-stability row T together with the tolerance ledger Y. The row T does not quantify over an unbounded host net; it points to the finite windows in W and to the dyadic tolerances recorded by $\mathsf{DyadicRatCore}^{\uparrow}$.

The directed side is supplied by D and V. The common upper-window witnesses named by $\mathsf{DirectedSet}^{\uparrow}$ are the only witnesses that the stability row may use.

Finally, H, C, P, L keep the stability evidence inside the BEDC packet. The hsame transports preserve row identity, the Cont row replays only the displayed route, the Pkg row records the sibling sources, and the local $\mathsf{NameCert}$ row names the packet. Hence T is a finite carrier row, not an external Cauchy theorem. □

Theorem 8942.5 (Directed Cauchy net Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DirectedCauchyNet}^{\uparrow}$ *packet must pass through*

$$D, V, W, T, S, Q, Y, R, H, C, P, L.$$

The handoff compares the directed cofinal windows with the sequence-style route by reading $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *in that order. It exports no selected Moore–Smith limit, quotient-net equality, countable-choice row, host real equality, or ambient completeness theorem.*

Proof. Apply the NameCert obligation theorem. It identifies $D, V, W, T, S, Q, Y, R, H, C, P, L$ as the whole local surface and gives no second source for directed tails, stream windows, regular rational readback, dyadic tolerances, or the Real seal.

The completion handoff is sequential only after it is finite and directed. The packet first reads the $\mathsf{DirectedSet}^{\uparrow}$ substrate and the cofinal-window stability rows, then reads the retained window through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before serving the terminal $\mathsf{Real}^{\uparrow}$ seal.

The componentwise hsame row, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row preserve the same order of access. Thus the handoff compares net-style cofinal windows with the existing sequence-based Real route without adding a selected limit, quotient net, countable-choice schedule, host equality, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DirectedCauchyNet}^{\uparrow}$ packet is generated from a DirectedSet row, finite net-value rows, cofinal-window rows, Cauchy-tail stability evidence, StreamName readback, RegSeqRat handoff, DyadicRatCore tolerance accounting, a terminal Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$, seed)

Scope closed: Definition 8942.1, Definition 8942.2, Definition 8942.3, Definition 8942.4, and Definition 8942.5 seed the finite directed Cauchy-net carrier and its Real-completion handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DirectedCauchyNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected Moore–Smith limits, quotient-net equality, cofinal-choice schedules, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite window extraction, tail stability, Real-completion handoff, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DirectedCauchyNetUp.

Chapter 8943

A Concrete Naming Certificate for $\mathsf{OrderedSemiring}^{\uparrow}$

Following Bourbaki on ordered algebraic structures and Golan on semirings, the packet records a semiring carrier whose additive and multiplicative rows are compatible with a stable order classifier; it consumes chapter 327, chapter 1378, chapter 37, chapter 291, and chapter 38. The packet gives order-sensitive L10 arithmetic handoffs one named BEDC surface for monotone addition and positive-row multiplication, while keeping ordered rings, field order, quotient order, classical trichotomy, and any standard bridge outside the carrier.

Definition 8943.1 (Ordered semiring finite-history carrier)**.** *An* $\mathsf{OrderedSemiring}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, Q, R, D, E, A, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Semiring}^{\uparrow}$ *source row naming the shared carrier, addition, multiplication, distributivity, zero-annihilation ledger, transport, replay, provenance, and local* $\mathsf{NameCert}$ *surface.*
2. *Q is the* $\mathsf{Order}^{\uparrow}$ *comparison row naming the stable order classifier attached to the same displayed carrier entries read by S.*
3. *R is the* $\mathsf{Rat}^{\uparrow}$ *arithmetic readback row used by rational order-facing consumers without importing field order.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff row used by dyadic interval and sign-ledger consumers.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal read row reached only after the semiring and order rows remain visible.*
6. *A records addition monotonicity rows: whenever two displayed input comparisons are accepted by Q, the corresponding sum comparison is replayed through the addition route of S.*
7. *M records multiplication monotonicity by positive rows: whenever Q displays a nonnegative or positive multiplier row admitted by the rational or dyadic handoff, the product comparison is replayed through the multiplication route of S.*
8. *H records componentwise* hsame *transport for the semiring row, order row, rational readback, dyadic handoff, Real read, monotonicity ledgers, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from each order-aware arithmetic request through $S, Q, R, D, E, A, M, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{OrderedSemiring}^{\uparrow}$, $\mathsf{Semiring}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite order-compatible semiring packet. It has no coordinate for subtraction, additive inverses, field order, quotient order, trichotomy, ambient Real completeness, or a standard bridge.

Definition 8943.2 (Ordered semiring classifier)**.** *Two accepted* $\mathsf{OrderedSemiring}^{\uparrow}$ *carriers*

$$O = (S, Q, R, D, E, A, M, H, C, P, N) \quad \textit{and} \quad O' = (S', Q', R', D', E', A', M', H', C', P', N')$$

are classified together when the semiring rows classify through the displayed $\mathsf{Semiring}^{\uparrow}$ *classifier, the order rows classify through the displayed* $\mathsf{Order}^{\uparrow}$ *endpoint transport, the rational, dyadic, and Real handoff rows transport componentwise by* hsame*, and the addition and multiplication monotonicity ledgers replay the same* Cont *routes after that transport. The classifier compares no host ordered algebra and no quotient relation.*

Theorem 8943.3 (Ordered semiring NameCert obligations)**.** *For every accepted* $\mathsf{OrderedSemiring}^{\uparrow}$ *carrier* $O = (S, Q, R, D, E, A, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier habitation: the displayed* BHist *packet of Definition 8943.1 is inhabited by a semiring row, an order row, rational and dyadic readback rows, a Real read row, monotonicity ledgers, transport, replay, provenance, and local naming;*
2. *classifier transport: the classifier of Definition 8943.2 transports semiring, order, rational, dyadic, Real, monotonicity, replay, provenance, and local name rows componentwise;*
3. *addition monotonicity: the ledger A records only the comparison of sums obtained by replaying the addition route of S after input comparisons supplied by Q;*
4. *multiplication monotonicity by positive rows: the ledger M records only product comparisons whose multiplier positivity is already displayed by Q together with the R or D handoff row;*
5. *ledger nonescape: the rows H, C, P, N preserve and replay only the displayed finite inventory and export no ordered ring, field order, classical trichotomy, quotient order, or standard bridge.*

No ordered semiring obligation is stored outside these five rows.

Proof. Project the carrier of Definition 8943.1. It exposes the $\mathsf{Semiring}^{\uparrow}$ operation row S, the $\mathsf{Order}^{\uparrow}$ comparison row Q, the rational, dyadic, and Real handoff rows R, D, E, and the two monotonicity ledgers A, M.

The classifier of Definition 8943.2 then fixes the only allowed transport. It moves semiring, order, rational, dyadic, Real, monotonicity, replay, provenance, and local naming rows componentwise, so an accepted comparison after transport is still a comparison of the same displayed packet.

The addition and multiplication clauses are read from A and M. The addition row allows only monotone sum replay through S after input order rows from Q; the multiplication row allows only product replay when the multiplier's positivity has already been displayed by Q and by the rational or dyadic handoff.

Finally H, C, P, N exhaust the ledger. Since the packet has no row for additive inverses, ordered rings, field order, quotient order, classical trichotomy, or a standard bridge, the five listed rows are the whole local $\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}$ surface. □

Theorem 8943.4 (Ordered semiring addition and multiplication order compatibility)**.** *Every order-aware arithmetic consumer of an accepted* $\mathsf{OrderedSemiring}^{\uparrow}$ *carrier reads addition monotonicity and positive-row multiplication monotonicity only through*

$$S, Q, R, D \longmapsto A, M, H, C, P, N.$$

In particular, a $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *handoff may use the displayed semiring operations and stable order classifier, but it cannot extract an ordered ring, a field order, a quotient order, classical trichotomy, or any standard bridge from the packet.*

Proof. Begin with an order-aware arithmetic request. By Definition 8943.1, the available operation row is S and the available comparison row is Q; the rational and dyadic handoff rows R, D only record the finite readback needed by their own consumers.

Addition monotonicity is therefore forced through A. The ledger reads the two input comparisons from Q, applies the addition route of S, and records the sum comparison as a finite replay row.

Positive-row multiplication monotonicity is forced through M. The multiplier condition is not a host positivity predicate; it is the displayed positivity row already visible through Q and the R or D handoff. Only then may the multiplication route of S be replayed to produce the product comparison.

The Real-facing row E is downstream of these checks. Since the structural rows H, C, P, N preserve only the displayed route, no consumer can replace the route by ordered-ring structure, field order, quotient order, trichotomy, or a standard bridge. □

Theorem 8943.5 (Ordered semiring kernel scope)**.** *The* $\mathsf{OrderedSemiring}^{\uparrow}$ *kernel scope is exactly the finite BEDC packet consisting of the semiring carrier row, stable order classifier, rational readback, dyadic handoff, Real read boundary, addition monotonicity, positive-row multiplication monotonicity, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. Every L10 arithmetic handoff that claims this scope factors through those rows and through no stronger ordered-algebraic structure.*

Proof. The carrier definition lists the rows that form the packet, and the classifier definition states the only transport relation between accepted packets. Hence a kernel consumer receives exactly those named components.

The NameCert obligation theorem identifies the five finite rows that make the packet usable: habitation, classifier transport, addition monotonicity, multiplication monotonicity by positive rows, and ledger nonescape.

The compatibility theorem then identifies the allowed handoff route for $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. Combining these three readings leaves no remaining coordinate for ordered rings, field order, quotient order, trichotomy, or a standard bridge, so the displayed finite packet is exactly the kernel scope. □

Falsifiable prediction: An accepted $\mathsf{OrderedSemiring}^{\uparrow}$ read that produces addition or multiplication order compatibility while bypassing the displayed semiring row, order classifier, rational or dyadic handoff, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: semiring, order, rat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{OrderedSemiring}^{\uparrow}$ because this packet binds semiring operations, stable order classification, rational and dyadic positivity handoffs, Real-facing read boundaries, monotonicity ledgers, transport, replay, provenance, and local naming as one order-aware arithmetic interface.

Closure status for $\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}$.

Constructive story: An $\mathsf{OrderedSemiring}^{\uparrow}$ packet is generated from a SemiringUp source row, an OrderUp classifier row, RatUp and DyadicRatCoreUp handoff rows, a RealUp read boundary, addition monotonicity, positive-row multiplication monotonicity, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8943.1, Definition 8943.2, Definition 8943.3, Definition 8943.4, and Definition 8943.5 seed the finite order-aware semiring handoff surface for L10 arithmetic consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrderedSemiring}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly excludes ordered rings, field order, classical trichotomy, quotient order, and any standard bridge.

Upgrade path: Obligation closure requires checked carrier habitation, classifier transport, addition monotonicity, multiplication monotonicity by positive rows, ledger nonescape, and handoff scope for BEDC.Derived.OrderedSemiringUp.

Chapter 8944

A Concrete Naming Certificate for $\mathsf{MetricCompletionFilter}^{\uparrow}$

Following Bourbaki and Kelley filter completion, the $\mathsf{MetricCompletionFilter}^{\uparrow}$ packet records the filter-facing finite certificate surface for Cauchy completion. It sits between the metric-completion branch of chapter 1506, the Cauchy-filter route of chapter 333, the finite filter-base rows of chapter 1601, the stream-window presentation of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names completion data as finite filter rows; it does not introduce quotient classes, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: A completion consumer that reads a filter limit without displaying the $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows has left the $\mathsf{MetricCompletionFilter}^{\uparrow}$ packet.

Independence witness: metriccompletion, cauchyfilter, filterbase, streamname, regseqrat, real: $\mathsf{MetricCompletionFilter}^{\uparrow}$ is not bijective to any one sibling because it jointly carries completion admission, Cauchy-filter evidence, finite base refinement, stream windows, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming.

Definition 8944.1 (MetricCompletionFilter carrier)**.** *A* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, F, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. X, d *is the source metric row accepted by the* $\mathsf{MetricCompletion}^{\uparrow}$ *surface;*
2. F *is the* $\mathsf{CauchyFilter}^{\uparrow}$ *evidence row for the displayed completion-facing filter;*
3. B *is the* $\mathsf{FilterBase}^{\uparrow}$ *finite-base row refining the requested filter neighborhoods;*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the base tails;*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational tail data;*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal attached to the displayed readback;*
7. H *records componentwise* hsame *transport of the metric, Cauchy-filter, base, window, readback, and seal rows;*

8. *C records displayed* Cont *replay from metric-completion requests through Cauchy-filter evidence, finite base refinement, StreamName windows, RegSeqRat readback, and the Real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a quotient completion class, selected filter limit, countable-choice schedule, or ambient real completeness theorem.

Theorem 8944.2 (MetricCompletionFilter tail-base refinement)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *carrier*

$$M = (X, d, F, B, S, R, E, H, C, P, N)$$

must expose the $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{FilterBase}^{\uparrow}$ *row B, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E before it may be consumed as completion data. The finite base row B refines each displayed filter request to a tail window read through S and R, so the completion handoff is the displayed tail-base route rather than a quotient-class representative or a selected limit.*

Proof. Project the carrier of Definition 8944.1. The metric row and Cauchy-filter row are visible before any completion-facing read reaches the base, window, readback, or seal rows.

Use the finite row projection through B, S, R. The $\mathsf{FilterBase}^{\uparrow}$ row refines each displayed filter request to a named base tail, the $\mathsf{StreamName}^{\uparrow}$ row fixes the finite request window, and the $\mathsf{RegSeqRat}^{\uparrow}$ row reads that window as regular rational data.

The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of this readback. Componentwise hsame transport and displayed Cont replay preserve the same tail-base route, while Pkg provenance and the local NameCert row name the packet.

Thus every completion-facing consumer receives only the displayed Cauchy-filter, finite-base, StreamName, RegSeqRat, and Real-seal rows. No quotient class, selected limit, countable-choice schedule, or ambient completeness theorem is used. □

Theorem 8944.3 (MetricCompletionFilter obligation surface)**.** *For every accepted* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *carrier*

$$M = (X, d, F, B, S, R, E, H, C, P, N),$$

the visible completion-filter obligation surface is the finite BHist *inventory consisting of the metric row (X, d), Cauchy-filter row F, filter-base row B,* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFilter}^{\uparrow}}$ *naming row N. No completion obligation is stored outside these rows.*

Proof. Project the carrier of Definition 8944.1. The analytic rows are exactly $(X, d), F, B, S, R, E$, and the structural rows are exactly H, C, P, N.

Apply tail-base refinement from Definition 8944.2. It forces every completion-facing read to pass through the Cauchy-filter row, finite base row, finite StreamName window, regular rational readback, and terminal Real seal.

The remaining rows transport, replay, source, and locally name that same route. Since the carrier has no further coordinate, quotient completion classes, selected filter limits, countable-choice schedules, and ambient completeness cannot be part of the local obligation surface. □

Theorem 8944.4 (MetricCompletionFilter RegSeqRat handoff)**.** *Every accepted* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *read that exports tail data to a completion-facing consumer must pass from the* $\mathsf{FilterBase}^{\uparrow}$ *row* B *and* $\mathsf{StreamName}^{\uparrow}$ *window* S *through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *before any* $\mathsf{Real}^{\uparrow}$ *seal is visible. The handoff therefore records regular rational tail data as finite BEDC rows, with* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *naming carried along the same route.*

Proof. Use Definition 8944.2. The theorem already places B, S, R between the Cauchy-filter row and the terminal seal.

Read the obligation surface of Definition 8944.3. It identifies R as the unique regular rational readback row in the carrier, so a consumer cannot replace it by a selected filter limit or host completion representative.

The rows H, C, P, N preserve the same handoff under transport, replay, provenance, and naming. Thus the RegSeqRat handoff is the displayed route through B, S, R and not an ambient completeness principle. □

Theorem 8944.5 (MetricCompletionFilter Real-seal boundary)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *of an accepted* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *carrier is available only after the metric row, Cauchy-filter row, filter-base row, StreamName window, and RegSeqRat readback have been displayed and transported through* H, C, P, N*. A Real-facing consumer may therefore read* E *only as the end of the finite completion-filter route, not as quotient completion equality, a selected limit, a countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tail-base theorem Definition 8944.2. It places the seal row downstream of F, B, S, R for every completion-facing read.

Apply Definition 8944.4. The regular rational readback row is already displayed before the seal, and the structural rows H, C, P, N carry the same route.

Finally use the carrier projection from Definition 8944.1. The carrier has no coordinate for quotient completion equality, selected limits, countable-choice schedules, or ambient Real completeness, so the Real seal is confined to the finite route just described. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionFilter}^{\uparrow}$ packet is generated from a source metric row, a Cauchy-filter row, a finite FilterBase refinement, StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8944.1 and Definition 8944.2 bind the finite filter-completion carrier and tail-base refinement route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion classes, selected filter limits, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-base refinement, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionFilterUp.

Chapter 8945

A Concrete Naming Certificate for $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ packet records the terminal finite seal route by which the L10 Real and Completion phase reaches $\mathsf{Real}^{\uparrow}$ without importing quotient stream equality, countable choice, a hidden tail oracle, a detached completion seal, or ambient completeness. It sits after the dyadic arithmetic of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the finite tail selection route of chapter 1354, and the witness-gluing route of chapter 1110. The packet also reads the exact completion boundary of chapter 1120 as a sibling surface rather than as an external completeness theorem.

Definition 8945.1 (Bishop L10 completion-terminal carrier)**.** *A* $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, Q, A, G, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger consumed by the terminal route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the route observes a Cauchy name.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row produced from the displayed windows and dyadic tolerances.*
4. *A is the* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *finite tail-choice row, read only as the displayed L10 tail selection surface.*
5. *G is the* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *witness-gluing row binding the witness ledger, tail envelope, synchronizer, and completion-seal classifier.*
6. *B is the* $\mathsf{RealCompletionExactBoundaryUp}$ *exact-boundary row, reached after D, S, Q, A, G have been displayed.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row served to the Real and Completion consumer.*
8. *E is the completion-terminal edge stating that D, S, Q, A, G, B, R are read as one finite terminal route.*
9. *H records componentwise* hsame *transport for every displayed object row, the edge, provenance, and local naming.*

10. *C records displayed* Cont *replay from dyadic tolerance data through stream windows, regular readback, finite tail choice, witness gluing, exact boundary, and terminal Real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CanonicalTailChoice}^{\uparrow}$, $\mathsf{CauchyWitnessGluing}^{\uparrow}$, RealCompletionExactBoundaryUp, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLTenCompletionTerminal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a countable-choice schedule, a hidden tail oracle, a detached completion seal, selected host limits, host real equality, metric completeness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8945.2 (Bishop L10 completion-terminal NameCert obligations)**.** *For every accepted* $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ *carrier* $T = (D, S, Q, A, G, B, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLTenCompleti}}$ *surface consists exactly of the following five obligation rows:*

1. *dyadic, stream, and regular readback admission through D, S, Q;*
2. *finite tail and witness admission through A, G;*
3. *exact-boundary and terminal Real sealing through B, R, E;*
4. *componentwise transport and displayed replay through H, C;*
5. *package provenance and local naming through P, N.*

These five rows exhaust the completion-terminal obligation surface and add no quotient stream, choice, hidden-tail, detached-seal, or ambient-completeness coordinate.

Proof. Project the carrier of Definition 8945.1. Its object rows are D, S, Q, A, G, B, R, its edge row is E, and its structural rows are H, C, P, N.

The first obligation row reads the finite observational stack. The dyadic tolerance ledger D budgets the finite windows S, and the regular rational row Q is the only readback exported from those observations.

The second row binds the completion witness side. The canonical tail-choice row A supplies the finite tail selection route, while the witness-gluing row G binds the witness ledger, tail envelope, synchronizer, and completion-seal classifier before the exact boundary is read.

The third row serves the terminal boundary. The exact-boundary row B and Real seal R are connected by the completion-terminal edge E, so no detached completion seal can be consumed apart from the displayed packet.

The remaining two rows are structural. Rows H, C transport and replay the displayed route, while P, N source and name it. Since no listed row contains quotient stream equality, countable choice, hidden tail data, detached seals, selected host limits, host real equality, metric completeness, or ambient completeness, the five rows exhaust the obligation surface. □

Theorem 8945.3 (Bishop L10 completion-terminal seal factorization)**.** *Every accepted Real and Completion consumer of a* $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ *packet reaches the terminal seal only through the displayed factorization*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CanonicalTailChoice}^{\uparrow} \longmapsto \mathsf{CauchyWitnessGluing}^{\uparrow} \longmapsto \mathsf{RealCo}$$

The structural rows H, C, P, N preserve this route under hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Start with the readback stack. The dyadic row D, stream row S, and regular row Q display the finite rational observation route before any completion-facing object is available.

Next read the L10 tail and witness surfaces. The finite tail-choice row A and witness-gluing row G expose the tail and witness data as named sibling packets, so a consumer cannot replace them by a host-selected subsequence or a hidden tail oracle.

Then read the terminal boundary. The exact-boundary row B is downstream of D, S, Q, A, G, and the terminal $\mathsf{Real}^{\uparrow}$ row R is served only across the edge E.

Finally apply the structural rows. The transports H, replays C, provenance P, and local name N carry the same factorization and do not add another completion route. Therefore every accepted consumer reaches $\mathsf{Real}^{\uparrow}$ through the displayed chain. □

Theorem 8945.4 (Bishop L10 completion-terminal refusal ledger)**.** *An accepted* $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ *packet refuses quotient stream equality, countable choice, hidden tail oracles, detached completion seals, selected host limits, host real equality, metric completeness, and ambient completeness. Any consumer asking for one of those objects is outside the displayed rows* $D, S, Q, A, G, B, R, E, H, C, P, N$ *and therefore outside the local* $\mathsf{NameCert}_{\mathsf{BishopLTenCompletionTerminal}^{\uparrow}}$ *surface.*

Proof. The carrier definition lists all rows of the accepted packet. None of $D, S, Q, A, G, B, R, E, H, C, P, N$ is a quotient-stream row, a choice schedule, a hidden tail oracle, a detached seal, a selected host limit, a host real equality row, a metric-completeness row, or an ambient-completeness theorem.

The NameCert obligations of Definition 8945.2 exhaust the same row inventory. The seal factorization of Definition 8945.3 then forces every terminal read through the listed sibling surfaces before $\mathsf{Real}^{\uparrow}$ is served.

Thus any requested object named in the refusal statement has no carrier coordinate, no transport coordinate, no replay coordinate, and no naming coordinate in the packet. It cannot be consumed as part of the accepted terminal route. □

Falsifiable prediction: Every accepted $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ consumer displays dyadic tolerance, stream-window, regular readback, canonical tail-choice, witness-gluing, exact-boundary, terminal Real, transport, replay, provenance, and local naming rows before the terminal seal is consumed.

Independence witness: dyadicratcore, streamname, regseqrat, canonicaltailchoice, cauchywitnessgluing, realcompletionexactboundary, real: $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ is not bijective to any listed sibling because it carries all seven object rows together with the terminal edge and structural NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLTenCompletionTerminal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLTenCompletionTerminal}^{\uparrow}$ packet is generated from DyadicRatCore, StreamName, RegSeqRat, CanonicalTailChoice, CauchyWitnessGluing, RealCompletionExactBoundary, RealUp, completion-terminal edge, hsame, Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLTenCompletionTerminal}^{\uparrow}}$, seed)

Scope closed: Definition 8945.1, Definition 8945.2, Definition 8945.3, Definition 8945.4, the falsifiable prediction, and the independence witness seed the finite terminal seal route from L10 completion data to $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLTenCompletionTerminal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, countable choice, hidden tail oracles, detached completion seals, selected host limits, host real equality, metric completeness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, five-row NameCert obligations, terminal seal factorization, refusal-ledger exactness, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.BishopLTenCompletionTerminalUp.

Chapter 8946

A Concrete Naming Certificate for $\mathsf{CauchyApartnessModulus}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{CauchyApartnessModulus}^{\uparrow}$ packet records the finite witness by which two regular Cauchy real names are read as apart before any order, quotient, or trichotomy principle is available. It sits between the stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the apartness surface of chapter 1881. The packet names a displayed positive separation window for regular Cauchy names; it does not assert trichotomy, quotient-real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8946.1 (Cauchy apartness modulus carrier)**.** *A* $\mathsf{CauchyApartnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, Q, D, R, J, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the two inspected Cauchy streams and the finite precision request at which they are compared.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two regular rational names on that window.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the dyadic positive gap and the comparison budget used to read it.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream window, regular rational readback, and dyadic gap budget have been displayed.*
5. *J is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *apartness row witnessing that the displayed dyadic gap is the public apartness evidence for the two Cauchy real names.*
6. *H records componentwise* hsame *transport for the StreamName, RegSeqRat, DyadicRatCore, Real, apartness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Cauchy-apartness request through the stream window, regular rational readback, dyadic gap budget, Real seal, and apartness row.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyApartnessModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApartnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, Q, D, R, J, H, C, P, N$. It has no coordinate for total order trichotomy, quotient representatives, selected moduli, countable-choice schedules, or ambient real completeness.

Theorem 8946.2 (Cauchy apartness modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyApartnessModulus}$ *carrier*

$$A = (S, Q, D, R, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyApartnessModulus}^{\uparrow}}$ *surface consists exactly of StreamName window admission, RegSeqRat readback, DyadicRatCore positive-gap budgeting, Real-seal visibility, CauchyRealApartness witness alignment, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The surface supplies apartness evidence only through the displayed finite window and exports no trichotomy, quotient-real equality, selected Cauchy limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 8946.1. The rows S, Q, D expose the inspected stream window, regular rational readback, and dyadic gap budget before the Real seal or apartness row can be consumed.

The apartness witness is then read through J. Since J is downstream of the same S, Q, D, R route, a consumer sees a positive separation window rather than a detached order decision or quotient representative.

Finally H, C, P, N preserve and name that displayed route. The carrier has no coordinate for trichotomy, chosen representatives, selected limits, countable choice, or ambient completeness, so none is exported by the local NameCert surface. □

Theorem 8946.3 (Cauchy apartness modulus regular window)**.** *Every* $\mathsf{CauchyApartnessModulus}^{\uparrow}$ *read that reaches the apartness row J must first display the* $\mathsf{StreamName}^{\uparrow}$ *window S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R. Thus the apartness modulus is a regular-window route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyRealApartness}^{\uparrow}$$

and not an external order oracle or quotient-real comparison.

Proof. Start from a consumer read of J. By Definition 8946.1, the apartness row is attached to the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows.

The regular rational readback and dyadic tolerance budget determine the public positive gap. The terminal Real seal is visible only after those rows have been opened, so the apartness row is a finite regular-window witness.

Replay through H, C, P, N preserves the same route. A consumer that omits any of S, Q, D, R has not reached the named apartness modulus and cannot read J as public evidence. □

Theorem 8946.4 (Cauchy apartness modulus ledger refusal)**.** *An accepted* $\mathsf{CauchyApartnessModulus}^{\uparrow}$ *packet refuses every attempted ledger read that omits one of $S, Q, D, R, J, H, C, P, N$. In particular, a route that supplies only a Real seal, only a dyadic inequality, or only an apartness label is boundary evidence, not a public apartness modulus for* $\mathsf{CauchyRealApartness}^{\uparrow}$.

Proof. Use the obligation surface of Definition 8946.2. It lists all rows that the local certificate admits, including the stream window, regular rational readback, dyadic tolerance budget, Real seal, and apartness witness.

Apply the regular-window route of Definition 8946.3. Any public apartness read reaches J only through S, Q, D, R and then through the structural rows H, C, P, N.

Therefore a ledger read missing one of these rows lies outside the accepted carrier. Such a read may be boundary evidence for a later consumer, but it is not a $\mathsf{CauchyApartnessModulus}^{\uparrow}$ certificate and cannot be exported as a public apartness modulus. □

Falsifiable prediction: A consumer that exports $\mathsf{CauchyRealApartness}^{\uparrow}$ evidence while omitting the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ positive-gap row, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, or local naming row has left the $\mathsf{CauchyApartnessModulus}^{\uparrow}$ packet.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchyrealapartness: CauchyApartnessModu is not bijective to any listed sibling because it joins a StreamName comparison window, RegSeqRat readback, DyadicRatCore positive-gap budget, Real seal, CauchyRealApartness witness, transport, replay, provenance, and local NameCert as one finite apartness-modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApartnessModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyApartnessModulus}^{\uparrow}$ packet is generated from a StreamName comparison window, RegSeqRat readback, DyadicRatCore positive-gap budget, a Real seal, a CauchyRealApartness witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApartnessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8946.1, Definition 8946.2, Definition 8946.3, and Definition 8946.4 seed the finite Cauchy apartness modulus route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApartnessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total order trichotomy, quotient-real equality, selected Cauchy limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-window readback, dyadic positive-gap budgeting, Real-seal visibility, apartness witness alignment, transport, replay, provenance, and local naming for BEDC.Derived.CauchyApartnessModulusUp.

Chapter 8947

A Concrete Naming Certificate for $\mathsf{DirichletFunction}^{\uparrow}$

Following Dirichlet and Bishop–Bridges constructive analysis, the $\mathsf{DirichletFunction}^{\uparrow}$ packet records the classical rational-versus-irrational boundary example as finite BEDC data. It sits after the rational carrier of chapter 37, the rational-density surface of chapter 1409, the stream naming surface of chapter 126, the regular rational-sequence readback of chapter 301, the real seal of chapter 38, the continuous-map packet of chapter 130, and the boundary comparison supplied by chapter 6525. Its role is not to assert a host function equality or a complete classical discontinuity theorem, but to expose the finite NameCert route by which a Real-facing consumer sees rational density, irrational-boundary refusal, stream windows, regular readback, and a terminal $\mathsf{Real}^{\uparrow}$ seal.

Falsifiable prediction: An accepted $\mathsf{DirichletFunction}^{\uparrow}$ packet that omits rational-density evidence, treats the irrational-boundary refusal as a continuity proof, or reads the terminal seal without a displayed StreamName window refutes this carrier.

Independence witness: rat, rationaldensityreal, streamname, regseqrat, real, continuous-map, thomae-function: none is bijective because DirichletFunctionUp binds rational-density branching, irrational-boundary refusal, finite stream windows, regular rational readback, Real-seal delivery, transport, continuation replay, provenance, and local naming in one packet.

Definition 8947.1 (Dirichlet-function carrier)**.** *A* $\mathsf{DirichletFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, \Delta, I, W, G, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Rat}^{\uparrow}$ *source row for rational probes and rational values;*
2. *Δ is the rational-density evidence row over the surrounding* $\mathsf{Real}^{\uparrow}$ *boundary;*
3. *I is the irrational-boundary refusal row, recording that a non-rational boundary read is not converted into a rational witness;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the boundary read is opened;*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window;*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, Δ, I, W, G are displayed;*
7. *H records componentwise* hsame *transport for the rational, density, refusal, window, readback, seal, provenance, and naming rows;*

8. *C records displayed* Cont *replay of the packet as a syntactic continuation route, not as a continuity claim;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalDensityReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ThomaeFunction}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletFunction}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $R, \Delta, I, W, G, A, H, C, P, N$. It has no coordinate for quotient real equality, host function extensionality, a continuity proof, a complete discontinuity-set classifier, or an omniscient rationality decision.

Theorem 8947.2 (Dirichlet-function NameCert obligations)**.** *For the carrier of Definition 8947.1, the local* $\mathsf{NameCert}_{\mathsf{DirichletFunction}^{\uparrow}}$ *surface consists of carrier admission for $D = (R, \Delta, I, W, G, A, H, C, P, N)$, rational-probe exposure through R, rational-density evidence through Δ, irrational-boundary refusal through I, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through G, terminal* $\mathsf{Real}^{\uparrow}$ *seal delivery through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed tuple in Definition 8947.1. The rational row, density evidence, refusal row, stream window, regular readback, seal, transport, replay, provenance, and naming row are all visible coordinates of one finite packet.

The obligations are then read in order. The rational row supplies the rational branch, the density row records the nearby rational evidence, and the refusal row prevents an irrational-boundary read from being reclassified as a rational witness.

The window and readback rows open the finite StreamName and RegSeqRat route before the Real seal is delivered. The remaining rows transport, replay, source, and name the same packet, so there is no hidden coordinate for extensional equality, continuity, complete discontinuity classification, or omniscient rationality. □

Theorem 8947.3 (Dirichlet-function rational and irrational boundary)**.** *Every accepted* $\mathsf{DirichletFunction}^{\uparrow}$ *read keeps the rational-density branch and the irrational-boundary refusal simultaneously visible. The rational side may be consumed only through $R, \Delta, W, G, A, H, C, P, N$, while the non-rational boundary side remains a refusal row I and cannot be upgraded into a host decision procedure, quotient real equality, or a continuity statement.*

Proof. Use the obligation surface of Definition 8947.2. It exposes both the rational-density evidence row Δ and the irrational-boundary refusal row I before any Real-facing seal can be read.

The StreamName and RegSeqRat rows keep the boundary read finite. They supply a window and regular rational readback attached to the displayed packet, not an ambient equality of real functions or a host-level decision of rationality.

Since I is a refusal row and no additional classifier coordinate exists, a non-rational boundary read cannot be repackaged as continuity, quotient real equality, or a complete discontinuity-set theorem. The packet therefore preserves the rational and non-rational boundary sides as distinct displayed obligations. □

Theorem 8947.4 (Dirichlet-function Real-seal factorization)**.** *For every accepted* $\mathsf{DirichletFunction}^{\uparrow}$ *carrier*

$$D = (R, \Delta, I, W, G, A, H, C, P, N),$$

each completion-facing read of the terminal $\mathsf{Real}^{\uparrow}$ *seal row A factors through the displayed rational row R, rational-density row Δ, irrational-boundary refusal row I, finite* $\mathsf{StreamName}^{\uparrow}$ *window*

W, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback G. The factorization uses* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows only for the same finite packet, and it exports no continuity theorem, classical discontinuity-set completeness, host function extensionality, quotient real equality, or countable-choice selector.*

Proof. Project the carrier. The seal row A appears only after the rational, density, refusal, window, and readback rows in the displayed tuple.

Apply Definition 8947.2. Any accepted seal read must preserve the same rational evidence, refusal row, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming.

Now use the boundary theorem Definition 8947.3. It prevents the refusal row from becoming a host rationality decision and keeps the rational-density branch separate from continuity or discontinuity-set claims. Thus the Real seal factors through exactly the displayed BEDC rows and no stronger analytic principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletFunction}^{\uparrow}$ packet is generated from rational probes, rational-density evidence, an irrational-boundary refusal row, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DirichletFunction}^{\uparrow}}$, seed)

Scope closed: Definition 8947.2, Definition 8947.3, and Definition 8947.4 expose the seed carrier surface for rational-density branching and irrational-boundary refusal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DirichletFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim continuity, classical discontinuity-set completeness, host function extensionality, quotient real equality, countable choice, or Lean verification.

Upgrade path: Scoped closure requires checked local obligations tying the carrier rows to the concrete RatUp, StreamNameUp, RegSeqRatUp, RealUp, ContinuousMapUp, and ThomaeFunctionUp sibling surfaces.

Chapter 8948

A Concrete Naming Certificate for $\mathsf{FiniteImageStreamWindow}^{\uparrow}$

Following Bishop and Bridges on finite means in constructive analysis, the $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ packet records the finite image of a displayed $\mathsf{StreamName}^{\uparrow}$ observation window before any Real or Completion consumer reads it as a completed set of values. It sits after the finite stream-name surface of chapter 126, the regular rational readback route of chapter 301, the continuous-map interface of chapter 130, and the terminal real-name seal of chapter 38. The packet names only finite source rows, finite image rows, readback ledgers, classifier transport, replay, provenance, and local naming; it does not select a limit, form a host image set, quotient stream equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8948.1 (Finite image stream-window carrier)**.** *A* $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, J, F, I, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose inspected entries are named by the packet;*
2. *J is the finite window-index row, recording the displayed positions of S that are inspected;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *or certified readback row applied to the inspected source entries;*
4. *I is the finite image ledger, listing the image endpoints obtained from the entries indexed by J;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite rational approximants attached to the image ledger;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row visible only after the stream window and regular rational rows have been read;*
7. *H stores componentwise* hsame *transport for the source, index, map, image, readback, and seal rows;*
8. *C stores displayed* Cont *continuation from the finite source window through the image ledger and real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteImageStreamWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite image set, a selected limiting image, a quotient of streams, or a completed compactness object.

Theorem 8948.2 (Finite image stream-window NameCert obligations)**.** *For the carrier of Definition 8948.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteImageStreamWindow}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$W = (S, J, F, I, Q, R, H, C, P, N),$$

finite source-window exposure through S *and* J*, certified map exposure through* F*, finite image ledger exposure through* I*, regular rational readback through* Q*, Real-facing seal visibility through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows S, J, F, I expose the finite source window and its image before any rational readback or Real-facing seal is available.

The rows Q and R are read only as handoff rows. They do not create new image endpoints; they record the regular rational and Real-facing surfaces attached to the finite ledger already displayed by I.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no row contains an infinite image set or selected limit, the NameCert obligations are exactly the displayed finite rows. □

Theorem 8948.3 (Finite image stream-window classifier stability)**.** *If two accepted* $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ *packets are related by the displayed classifier, then their source window rows, certified map rows, finite image ledgers, regular rational readbacks, Real-facing seals, transport rows, replay rows, provenance rows, and local names are matched componentwise. In particular, classifier transport cannot replace a finite image ledger by an ambient host image set.*

Proof. Read the classifier through the carrier rows of Definition 8948.1. A classified replacement must preserve the entries named by S and J, the applied readback row F, and the finite image ledger I.

The readback and seal rows Q and R are downstream of that same finite ledger, so they transport only with the displayed image entries. The structural rows H, C, P, N then repack the transported packet as another finite image window, not as a host image object. □

Theorem 8948.4 (Finite image stream-window RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of a* $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ *packet must first read the finite source window* S, J*, the certified map row* F*, and the finite image ledger* I*. The consumer may then read* Q *as the regular rational handoff for those displayed image entries, but it cannot infer a stream quotient or a selected limit from the handoff alone.*

Proof. The carrier places Q after the source window, certified map row, and finite image ledger. A regular rational consumer therefore receives Q only together with the displayed finite image rows that generated it.

Since Q records readback for the entries of I, it does not identify all future stream entries, select a convergent image, or quotient two streams. Those conclusions would require rows absent from the carrier. □

Theorem 8948.5 (Finite image stream-window Real seal nonescape)**.** *Every Real-facing read of an accepted* $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ *packet factors through*

$$S, J, F, I, Q, R, H, C, P, N$$

before the $\mathsf{Real}^{\uparrow}$ *seal row is visible. The seal exports only the finite image-window readback already certified by the packet; it does not broaden the packet to ambient Real completeness or selected image limits.*

Proof. By Definition 8948.2, the Real-facing row R is part of one packet with the source window, image ledger, and regular rational readback rows.

A Real-facing consumer must therefore read S, J, F, I, Q before R can be interpreted. The classifier stability theorem prevents replacing I by a host image set during transport, so the seal remains attached to the displayed finite image window. □

Falsifiable prediction: An accepted $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ packet is refuted by any Real or Completion consumer that reads the $\mathsf{Real}^{\uparrow}$ seal while skipping the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, or finite image ledger.

Independence witness: streamname, regseqrat, continuousmap, real: StreamNameUp supplies source observations but not image ledgers; RegSeqRatUp supplies rational readback but not a finite image surface; ContinuousMapUp supplies certified map rows but not the inspected window; RealUp supplies a terminal seal without recording the finite source-to-image ledger. None is bijective to the full $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ carrier under NameCert-row preservation.

The Lean follow-up file `lean4/BEDC/Derived/FiniteImageStreamWindowUp.lean` must provide the concrete carrier, the NameCert obligation theorem, classifier stability, the regular-rational handoff, and the Real-seal nonescape theorem without adding any host image-set primitive.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteImageStreamWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteImageStreamWindow}^{\uparrow}$ packet is generated from a StreamName source row, a finite window-index row, a certified ContinuousMap or readback row, a finite image ledger, RegSeqRat readback, a terminal Real seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteImageStreamWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8948.1, Definition 8948.2, Definition 8948.3, Definition 8948.4, and Definition 8948.5 seed the finite image-window carrier and its Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteImageStreamWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host image sets, selected image limits, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, classifier stability, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteImageStreamWindowUp.

Chapter 8949

A Concrete Naming Certificate for $\mathsf{MeasurementResidueLedger}^{\uparrow}$

The $\mathsf{MeasurementResidueLedger}^{\uparrow}$ packet realizes the measurement-residue grammar of section 9414.12 as finite NameCert data. It sits beside the quantum-state source of chapter 217, the observable readout surface of chapter 218, and the mixed-state ledger of chapter 220. Those siblings name state, observable, and density rows separately; this packet binds a source row, measurement-interface projection row, visible outcome, residual fiber, section-update certificate, and audit ledger into one finite boundary object.

Definition 8949.1 (Measurement residue ledger carrier)**.** *A* $\mathsf{MeasurementResidueLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, P, O, F, U, A, H, C, G, N)$$

with the following displayed rows:

1. *T is the source row, such as a* $\mathsf{QuantumState}^{\uparrow}$ *or* $\mathsf{DensityMatrix}^{\uparrow}$ *input surface;*
2. *M is the measurement-context row, recording preparation, observable or probe family, instrument data, and public ledger scope;*
3. *P is the* MeasurementInterface *projection row from the source surface to the outcome carrier;*
4. *O is the visible outcome row produced by the projection;*
5. *F is the residual-fiber row, recording the source distinctions not separated by the chosen interface;*
6. *U is the section-update certificate row, recording when a post-outcome section into the residual fiber is licensed;*
7. *A is the audit ledger, recording replay, acceptance, refusal, and public-export boundaries;*
8. *H stores componentwise* hsame *transport for the source, context, projection, outcome, residue, update, and audit rows;*
9. *C stores displayed* Cont *continuation through the measurement projection, residual fiber, and section-update rows;*
10. *G is the* Pkg *provenance over* $\mathsf{QuantumState}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, $\mathsf{DensityMatrix}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MeasurementResidueLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden classical assignment, a global common refinement of all measurements, a Born-weight law, or an unlicensed collapse operation.

Theorem 8949.2 (Measurement residue ledger NameCert obligations)**.** *For the carrier of Definition 8949.1, the local* $\mathsf{NameCert}_{\mathsf{MeasurementResidueLedger}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$L = (T, M, P, O, F, U, A, H, C, G, N),$$

source exposure through T, measurement-context exposure through M, interface projection through P, visible outcome exposure through O, residual-fiber exposure through F, section-update certification through U, audit-ledger exposure through A, componentwise hsame *transport through H, displayed* Cont *replay through C, provenance through G, and local naming through N.*

Proof. Project the finite packet. The rows T, M, P, O expose the source, measurement context, projection, and visible outcome before the residue or section rows are interpreted.

The residual row F records source distinctions retained by that chosen interface, while U records only licensed section-style updates into the displayed residual fiber. The audit row A records replay and refusal boundaries for that same packet.

The structural rows H, C, G, N transport, replay, source, and name the displayed ledger. Since no row supplies a hidden classical assignment or global common refinement, the NameCert obligations are exactly the listed finite rows. □

Theorem 8949.3 (Measurement residue ledger interface projection)**.** *Every public outcome exported by an accepted* $\mathsf{MeasurementResidueLedger}^{\uparrow}$ *packet factors through the displayed* MeasurementInterface *projection row P. A ledger row that carries a visible outcome O without the matching source row T, context row M, and projection row P is not an accepted public outcome ledger for this packet.*

Proof. The carrier places O after the source, context, and projection rows. Therefore a public read of O must preserve the source row T, the measurement context M, and the interface projection P.

If an alleged ledger row keeps O while dropping P, it no longer displays the map from the source surface to the outcome carrier. The NameCert obligations then fail at the projection row, so the row cannot export a public outcome ledger as $\mathsf{MeasurementResidueLedger}^{\uparrow}$. □

Theorem 8949.4 (Measurement residue ledger non-hidden-variable boundary)**.** *The residual-fiber row F of an accepted* $\mathsf{MeasurementResidueLedger}^{\uparrow}$ *packet is not a hidden-variable assignment. It records exactly the source distinctions not separated by the displayed projection row P, and it cannot be read as a simultaneous visible outcome for measurement contexts not displayed by M.*

Proof. Read F together with T, M, P, O. The fiber is attached to one source surface, one measurement context, one projection row, and one visible outcome.

A hidden-variable assignment would have to provide visible outcomes across contexts not named by M or a common classicalizing refinement not present in the carrier. Since neither row exists in L, the residual fiber remains a ledger of unseparated source structure for the chosen projection only. □

Theorem 8949.5 (Measurement residue ledger section-update handoff)**.** *Every section-style update exported by an accepted* $\mathsf{MeasurementResidueLedger}^{\uparrow}$ *packet must read the visible outcome O, the residual-fiber row F, the section-update certificate U, and the audit ledger A. Without those rows, a post-outcome state selection is not licensed by this NameCert packet.*

Proof. The section row U is displayed only after the source, context, projection, outcome, and residual-fiber rows have been fixed. It therefore licenses a section into the residual fiber of the accepted outcome rather than a primitive state change.

The audit row A records replay and public-export boundaries for that section. If O, F, U, A are not all present, the packet lacks either the outcome, the target fiber, the section license, or the audit replay needed for the handoff. □

Falsifiable prediction: An accepted $\mathsf{MeasurementResidueLedger}^{\uparrow}$ row without a matching $\mathsf{MeasurementInterface}$ projection row cannot export a public outcome ledger.

Independence witness: quantumstate, observable, densitymatrix, quantumchannel: QuantumStateUp supplies source-state rows without measurement residue sections; ObservableUp supplies readout operators without residual-fiber ledgers; DensityMatrixUp supplies mixed-state trace ledgers without projection-specific residue rows; QuantumChannelUp supplies state-transform rows without the measurement-interface projection and section-update certificate. The non-bijection is witnessed by the missing residue-section rows.

The Lean follow-up file `lean4/BEDC/Derived/MeasurementResidueLedgerUp.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{MeasurementResidueLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MeasurementResidueLedger}^{\uparrow}$ instances, together with the projection, non-hidden-variable boundary, and section-update handoff theorems.

Closure status for $\mathsf{NameCert}_{\mathsf{MeasurementResidueLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MeasurementResidueLedger}^{\uparrow}$ packet is generated from a quantum source row, a measurement-context row, a MeasurementInterface projection row, a visible outcome row, a residual-fiber row, a section-update certificate, an audit ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeasurementResidueLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8949.1, Definition 8949.2, Definition 8949.3, Definition 8949.4, and Definition 8949.5 seed the finite measurement-residue ledger and its projection, residue, section-update, and audit boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeasurementResidueLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the Born rule, Hilbert-space dynamics, hidden-variable assignments, global common refinements, unlicensed collapse operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, projection exactness, non-hidden-variable boundary, section-update handoff, transport, replay, provenance, and local naming for BEDC.Derived.MeasurementResidue

Chapter 8950

A Concrete Naming Certificate for $\mathsf{CompactCauchyRealizer}^{\uparrow}$

Following Bishop–Bridges constructive compact metric spaces, the $\mathsf{CompactCauchyRealizer}^{\uparrow}$ packet records the finite route by which compact finite-net data and displayed Cauchy windows are realized as a completion-facing regular rational name before any terminal Real seal is read. It sits after the compact metric finite-net surface of chapter 129, the completion-filter route of chapter 8944, the compact Cauchy filter packet of chapter 8097, the compact Cauchy net packet of chapter 7735, the dyadic tolerance ledger of chapter 291, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet realizes only displayed compact-net and Cauchy-window data; it does not quotient streams, select limits, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8950.1 (Compact Cauchy realizer carrier)**.** *A* $\mathsf{CompactCauchyRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (K, Q, M, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row.*
2. *Q is the compact Cauchy row, read through the* $\mathsf{CompactCauchyFilter}^{\uparrow}$ *or* $\mathsf{CompactCauchyNet}^{\uparrow}$ *sibling surface.*
3. *M is the* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *completion-filter row attached to the same compact source and Cauchy request.*
4. *W records the local Cauchy window whose compatibility is inspected before completion handoff.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for mesh, Cauchy-window, and readback comparisons.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *request row used to inspect the displayed windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from K, Q, M, W, D, S.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after the regular readback row is displayed.*
9. *H records componentwise* hsame *transport for compact source, Cauchy row, completion-filter row, window, tolerance, StreamName, readback, seal, replay, provenance, and naming rows.*
10. *C records displayed* Cont *replay of the same compact Cauchy realization route.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactCauchyRealizer}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricCompletionFilter}^{\uparrow}$, $\mathsf{CompactCauchyFilter}^{\uparrow}$, $\mathsf{CompactCauchyNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient stream equality, selected Cauchy representatives, countable-choice schedules, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8950.2 (Compact Cauchy realizer NameCert obligations)**.** *For every accepted* $\mathsf{CompactCauchyRealizer}^{\uparrow}$ *carrier*

$$Z = (K, Q, M, W, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactCauchyRealizer}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, compact Cauchy row exposure, completion-filter routing through* $\mathsf{MetricCompletionFilter}^{\uparrow}$*, local Cauchy-window display, dyadic tolerance exposure, StreamName finite-window exposure, RegSeqRat readback, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8950.1. The rows K, Q, M, W, D, S, R, E are the only analytic rows, and H, C, P, N are the only structural rows.

Read compactness through the sibling finite-net data. The compact metric row K and compact Cauchy row Q keep the source finite, while the completion-filter row M routes the Cauchy request through chapter 8944.

The local window, tolerance, StreamName, and RegSeqRat rows then produce the displayed readback. The terminal seal E is downstream of that readback, and the structural rows transport, replay, source, and name the same route.

Since the carrier has no additional coordinate, quotient streams, selected limits, countable-choice schedules, open-cover compactness, and ambient Real completeness are not obligations of the local certificate. □

Theorem 8950.3 (Compact Cauchy realizer completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactCauchyRealizer}^{\uparrow}$ *carrier must pass through the compact finite-net row K, compact Cauchy row Q,* $\mathsf{MetricCompletionFilter}^{\uparrow}$ *row M, local Cauchy window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, finite* $\mathsf{StreamName}^{\uparrow}$ *row S, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is available. The handoff realizes compact Cauchy data as a displayed regular rational route, not as a quotient stream, selected limit, countable-choice schedule, or ambient completeness theorem.*

Proof. Apply the NameCert obligations of Definition 8950.2. They expose the compact finite-net, compact Cauchy, completion-filter, local window, tolerance, StreamName, RegSeqRat, and Real-seal rows in the carrier order.

Use the completion-filter dependency of chapter 8944. The row M forces the completion read to remain a finite filter-facing route before any Real-facing consumer may inspect the seal.

The dyadic tolerance, StreamName, and RegSeqRat rows then convert the displayed windows into a regular rational readback. The terminal Real seal is therefore reached only after K, Q, M, W, D, S, R have been displayed and transported through H, C, P, N. □

Theorem 8950.4 (Compact Cauchy realizer L10 scope)**.** *The* $\mathsf{CompactCauchyRealizer}^{\uparrow}$ *packet supplies an L10 Real-and-Completion scope consisting of compact finite-net data, compact Cauchy rows, completion-filter routing, dyadic tolerance rows, finite StreamName windows, RegSeqRat*

readback, Real sealing, transport, replay, provenance, and local naming. It may feed completion, density-net, or compact-realization consumers, but it does not close quotient stream equality, selected representatives, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Start with the carrier and obligation surface, Definition 8950.1 and Definition 8950.2. They bind the visible L10 rows to $K, Q, M, W, D, S, R, E, H, C, P, N$.

Apply the completion handoff of Definition 8950.3. Any completion-facing read must traverse the compact finite-net row, compact Cauchy row, completion-filter row, dyadic tolerance row, StreamName row, and RegSeqRat row before the Real seal.

Finally compare with the sibling compact Cauchy and density-net routes of chapter 8097, chapter 7735, and chapter 6003. Those consumers may read the displayed packet as L10 completion data, but none of them adds the excluded quotient, choice, open-cover, ambient-completeness, Lean-verification, or bridge rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchyRealizer}\uparrow}$.

Constructive story: A $\mathsf{CompactCauchyRealizer}^{\uparrow}$ packet is generated from compact finite-net rows, compact Cauchy rows, MetricCompletionFilter rows, local Cauchy windows, DyadicRatCore tolerance data, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCauchyRealizer}\uparrow}, \mathsf{seed})$

Scope closed: Definition 8950.1, Definition 8950.2, Definition 8950.3, and Definition 8950.4 seed the compact finite-net Cauchy realization route for L10 Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCauchyRealizer}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, compact Cauchy routing, completion-filter handoff, dyadic tolerance exactness, StreamName coverage, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactCauchyRealizerUp.

Chapter 8951

A Concrete Naming Certificate for $\mathsf{CauchyTailTriangle}^{\uparrow}$

Following Bishop and Bridges' use of explicit Cauchy tail estimates, the $\mathsf{CauchyTailTriangle}^{\uparrow}$ packet records a finite triangle ledger for composing two tail-diameter controls into a third. It consumes the tail-diameter surface of chapter 1746, the regular-tail composition route of chapter 1747, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38. The packet is a NameCert for displayed tail-diameter composition; it is not ambient metric completeness, quotient equality of real names, a selected limit, countable choice, or a host triangle theorem outside the finite ledger.

Definition 8951.1 (CauchyTailTriangle carrier)**.** *A* $\mathsf{CauchyTailTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, C, T_{AB}, T_{BC}, T_{AC}, D, W, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A, B, C are the three tail-name rows whose finite windows are read through the accepted* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *surfaces.*
2. *T_{AB} is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *row bounding the tail distance from A to B after the displayed threshold.*
3. *T_{BC} is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *row bounding the tail distance from B to C after the displayed threshold.*
4. *T_{AC} is the target tail-diameter row for the composed A to C comparison.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that stores the two input tolerances, their displayed sum, and the target tolerance budget.*
6. *W is the common finite tail window on which the three names and all three tail-diameter rows are simultaneously readable.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational bounds extracted from D and W.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the dyadic and regular-rational rows have been displayed.*
9. *H stores componentwise* hsame *transport for the three names, three tail-diameter rows, tolerance ledger, common window, regular readback, real seal, provenance, and naming row.*

10. *K stores the displayed* Cont *replay routes for projecting either input tail-diameter witness and for feeding them through the* $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *row.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyTailTriangle}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{RegularCauchyTailComposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, C, T_{AB}, T_{BC}, T_{AC}, D, W, R, E, H, K, P, N$. It exports no ambient metric space, completed limit, quotient real equality, chosen modulus, countable choice, or host triangle inequality.

Theorem 8951.2 (CauchyTailTriangle NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailTriangle}^{\uparrow}$ *carrier*

$$Q = (A, B, C, T_{AB}, T_{BC}, T_{AC}, D, W, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailTriangle}^{\uparrow}}$ *surface consists of carrier admission for the three tail-name rows, projection of the two input* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *witnesses, display of the composed T_{AC} row, dyadic tolerance accounting through D, common-window readability through W, regular-rational readback through R, terminal real sealing through E, componentwise transport through H, continuation replay through K, package provenance through P, and local naming through N.*

Proof. Carrier admission is the tuple of Definition 8951.1. The rows A, B, C bind the three names, while T_{AB}, T_{BC}, T_{AC} bind the two input diameter witnesses and the target diameter row.

The tolerance and window obligations are the rows D, W, R, E. The dyadic row D stores the two input budgets and their displayed sum, the window W is the common tail domain where all three estimates are visible, and the rows R, E route the regular rational readback into the terminal real seal.

The structural obligations are H, K, P, N. Transport preserves each displayed coordinate, K replays the two projections and the tail-composition row, and provenance plus naming repack the same finite packet. Since no excluded host object is a coordinate, these rows exhaust the local NameCert surface. □

Theorem 8951.3 (CauchyTailTriangle composition)**.** *Every accepted* $\mathsf{CauchyTailTriangle}^{\uparrow}$ *packet exposes a finite composition route*

$$T_{AB}, T_{BC}, D, W, R, E, K \longrightarrow T_{AC}.$$

The route first projects the two tail-diameter witnesses, then reads the shared dyadic budget and common tail window, and only then feeds the two witnesses through the displayed $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *row to expose the A to C tail-diameter witness.*

Proof. Project the rows T_{AB} and T_{BC} from the carrier. These are the only two input tail-diameter witnesses available to the packet.

Next read the shared ledger. The rows D and W display the finite tolerance addition and the common window on which both input estimates and the target estimate are readable; R and E keep the readback within regular-rational and real-seal rows.

Finally use K. Its continuation replay row is precisely the route from the two projected witnesses through $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ to the displayed target row T_{AC}, so the composition remains inside the finite packet. □

Theorem 8951.4 (CauchyTailTriangle RegSeqRat handoff)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{CauchyTailTriangle}^{\uparrow}$ *packet must pass through*

$$D, W, R, E$$

after the two input tail-diameter rows have been displayed. Thus the triangle ledger hands rational tail bounds to completion-facing consumers through the existing regular-rational and real-name surfaces, not through an untracked ambient completeness principle.

Proof. Begin with a consumer request for a rational or real bound. The carrier gives no direct bound coordinate except the dyadic ledger D, the common tail window W, the regular-rational readback R, and the terminal seal E.

The two input diameter rows must already be visible, because D records the displayed sum of their budgets and W is the common window on which those estimates are read.

Once R and E are consumed, the downstream route has only the accepted $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ surfaces. No selected limit, quotient real equality, or ambient completion theorem is introduced by the handoff. □

Theorem 8951.5 (CauchyTailTriangle non-escape)**.** *The public information exported by a* $\mathsf{CauchyTailTriangle}^{\uparrow}$ *carrier is exactly the displayed packet*

$$A, B, C, T_{AB}, T_{BC}, T_{AC}, D, W, R, E, H, K, P, N.$$

In particular, a consumer cannot extract ambient metric completeness, a chosen limit, quotient equality of real names, countable choice, or a host triangle inequality from the certificate.

Proof. Read the carrier coordinates. The three tail-name rows, the three tail-diameter rows, and the dyadic, window, regular-rational, and real-seal rows are all explicitly listed in Definition 8951.1.

The structural rows do not add new mathematical objects. The transport row H preserves displayed coordinates, the continuation row K replays only the two projections and the regular-tail composition route, and P, N provide provenance and local naming for the same packet.

Therefore every public read factors through the displayed rows. None of the excluded host objects is present as a coordinate or produced by the replay route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailTriangle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailTriangle}^{\uparrow}$ packet is generated from three tail-name rows, two input tail-diameter witnesses, a target tail-diameter row, dyadic tolerance accounting, a common finite window, regular-rational readback, a terminal RealUp seal, componentwise hsame transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8951.1, Definition 8951.2, Definition 8951.3, Definition 8951.4, and Definition 8951.5 seed the finite triangle ledger for composing Cauchy tail-diameter rows through dyadic, regular-rational, and real-seal handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient metric completeness, selected limits, quotient real equality, countable choice, or a host triangle inequality.

Upgrade path: Obligation closure requires checked carrier admission, projection of both input tail-diameter rows, target tail-diameter exposure, dyadic tolerance accounting, common-window readability, regular-rational handoff, real sealing, transport, continuation replay, provenance, local naming, and ledger non-escape for BEDC.Derived.CauchyTailTriangleUp.

Chapter 8952

A Concrete Naming Certificate for $\mathsf{AbelIdentity}^{\uparrow}$

Following Coddington–Levinson and the Sturm–Liouville tradition, the $\mathsf{AbelIdentity}^{\uparrow}$ packet records the finite boundary at which the Wronskian route may read an Abel identity without importing a host ODE theorem. It sits beside the Wronskian certificate of chapter 6877, the derivative rows of chapter 122, the regular rational readback of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the terminal real-name seal of chapter 38. The packet names the coefficient, derivative-admission, Wronskian, and integrating-factor ledgers as BEDC rows; it does not identify them with host solutions of a differential equation.

Definition 8952.1 (AbelIdentity carrier)**.** *An* $\mathsf{AbelIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, D, W, I, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the coefficient row for the displayed first-order trace term in the Abel identity route;*
2. *D is the* $\mathsf{Derivative}^{\uparrow}$ *derivative-admission row for the coefficient and Wronskian reads;*
3. *W is the* $\mathsf{Wronskian}^{\uparrow}$ *row supplied by the sibling packet of chapter 6877;*
4. *I is the integrating-factor ledger, including the exponential readback route used to compare adjacent Wronskian windows;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected coefficient and Wronskian requests;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coefficient, integral-ledger, and Wronskian values;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed Abel-identity readback;*
8. *H records componentwise* hsame *transport of coefficient, derivative, Wronskian, integrating-factor, window, readback, and seal rows;*
9. *C records displayed* Cont *replay from coefficient requests through derivative admission, Wronskian rows, integrating-factor ledger, StreamName windows, RegSeqRat readback, and the Real seal;*
10. *P is the* Pkg *provenance over* $\mathsf{AbelIdentity}^{\uparrow}$, $\mathsf{Wronskian}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelIdentity}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for host ODE equality, selected fundamental systems, classical exponential uniqueness, quotient real equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.AbelIdentityUp.TasteGate.AbelIdentityTasteGate single carrier alignment`

Theorem 8952.2 (AbelIdentity NameCert obligations)**.** *For every accepted* $\mathsf{AbelIdentity}^\uparrow$ *carrier*

$$A = (Q, D, W, I, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelIdentity}^\uparrow}$ *surface consists exactly of coefficient exposure through Q, derivative admission through D, Wronskian exposure through W, integrating-factor and exponential-ledger exposure through I, finite-window exposure through S, regular rational readback through R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8952.1. The coefficient row, derivative-admission row, Wronskian row, integrating-factor ledger, finite windows, readback row, and Real seal are visible before any structural transport or replay is read.

Read the dependency route in order. The coefficient row Q and derivative-admission row D supply the finite ODE-facing data, while W imports only the already displayed Wronskian packet surface from chapter 6877. The ledger I records the integrating-factor and exponential comparison as finite data, not as an ambient uniqueness theorem.

The value-facing route then passes through S, R, E. The $\mathsf{StreamName}^\uparrow$ row fixes inspected requests, $\mathsf{RegSeqRat}^\uparrow$ reads coefficient and Wronskian values with $\mathsf{DyadicRatCore}^\uparrow$ tolerance arithmetic, and $\mathsf{Real}^\uparrow$ seals only that displayed readback.

The rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for host ODE equality, selected bases, quotient Real equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.AbelIdentityUp.AbelIdentityCarrier namecert obligations`

Falsifiable prediction: An accepted Abel-identity read that uses host solution equality, a selected fundamental system, or an unledgered exponential uniqueness theorem refutes the $\mathsf{AbelIdentity}^\uparrow$ packet.

Independence witness: wronskian, derivative, continuousmap, streamname, regseqrat, dyadicratcore, real: $\mathsf{AbelIdentity}^\uparrow$ is not bijective to any listed sibling because it jointly carries coefficient rows, derivative admission, Wronskian rows, integrating-factor and exponential ledgers, rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelIdentity}^\uparrow}$**.**

Constructive story: An $\mathsf{AbelIdentity}^\uparrow$ packet is generated from coefficient rows, derivative-admission rows, a Wronskian row, an integrating-factor and exponential ledger, finite StreamName requests, RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AbelIdentity}^\uparrow}$, seed)

Scope closed: Definition 8952.1, Definition 8952.2, the falsifiable prediction, and the independence witness seed the Abel identity boundary packet for Wronskian and ODE-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AbelIdentity}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host ODE equality, selected fundamental systems, classical exponential uniqueness, quotient real equality, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient exposure, derivative-route admission, Wronskian handoff, integrating-factor ledger exactness, StreamName coverage, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbelIdentityUp.

Chapter 8953

A Concrete Naming Certificate for $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$

The $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ packet realizes the hash-like apophatic fixed-point distinction of Definition 9402.13: the visible digest value is a local token, while the fiber far end is an apophatic boundary role behind the token. It sits beside the digest-fiber ledger of chapter 1067, the apophatic fixed-point packet of chapter 1068, the hash-apophatic seal of chapter 1079, and the hash digest-fiber boundary of chapter 1087. Its distinguishing role is narrower: it records the refusal boundary that prevents a consumer from identifying the visible digest value with the apophatic far end of the fiber.

Definition 8953.1 (FiberFarEndDigestBoundary carrier)**.** *A* $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, T, G, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest-value row;*
2. *F is the source-fiber row behind D;*
3. *T is the apophatic fiber far-end row, read as boundary role rather than digest value;*
4. *G is the* GAP *provenance row keeping the source side and far-end boundary auditable;*
5. *R is the refusal row blocking any accepted read from equating D with T;*
6. *H records componentwise* hsame *transport for digest, fiber, far-end, GAP, refusal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay through the same digest, fiber, far-end, GAP, and refusal route;*
8. *P is the* Pkg *provenance over* $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, $\mathsf{HashApophaticSeal}^{\uparrow}$, $\mathsf{HashDigestFiberBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiberFarEndDigestBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global universe digest, source equality from digest equality, selected preimage, internal T *object, or quotient collapse of fiber membership.*

Theorem 8953.2 (FiberFarEndDigestBoundary NameCert obligations)**.** *For every accepted* FiberFarEndDigestBou *carrier*

$$B = (D, F, T, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiberFarEndDigestBoundary}^{\uparrow}}$ *surface consists exactly of visible digest exposure through* D*, source-fiber exposure through* F*, apophatic far-end boundary exposure through* T*, GAP provenance through* G*, digest/far-end equality refusal through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 8953.1. The visible row D and the source-fiber row F are distinct coordinates, and the far-end row T is exposed only as an apophatic boundary role.

Read the refusal route through G and R. The GAP provenance row keeps the fiber and far-end boundary attached to the digest, while the refusal row blocks the digest value from being read as the far-end identity.

The transport, replay, provenance, and local naming rows preserve the same boundary. Since no carrier coordinate stores source equality from digest equality, a selected preimage, an internal far-end object, a global digest, or quotient collapse, those readings are absent from the local NameCert surface. □

Theorem 8953.3 (Fiber far-end digest nonescape)**.** *Every accepted* $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ *consumer that reads the visible digest row* D *must keep the source-fiber row* F*, apophatic far-end row* T*, GAP provenance row* G*, refusal row* R*, componentwise transport* H*, continuation replay* C*, package provenance* P*, and local naming* N *in the same packet. Therefore digest equality cannot be used as fiber far-end identity, source equality, selected preimage, internal* T *object, global universe digest, or quotient fiber collapse.*

Proof. Start from the obligation surface of Definition 8953.2. It makes $D, F, T, G, R, H, C, P, N$ the complete inventory for any accepted read.

The nonescape claim follows by inspecting that inventory. The digest row is only the visible value; the far-end row is only the apophatic boundary role; and the refusal row keeps those readings apart for every transported or replayed consumer.

No additional coordinate remains from which a consumer could construct source equality, a selected preimage, an internal far-end object, a global digest, or a quotient collapse. The packet therefore preserves the digest/fiber/far-end distinction of Definition 9402.13. □

Falsifiable prediction: Any accepted read equating the visible digest value with fiber far-end identity refutes the $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ packet.

Independence witness: apophaticfixedpoint, apophaticfixedpointfiber, hashapophaticseal, gap, namecert: $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries visible digest value, source fiber, apophatic far-end boundary, GAP provenance, digest/far-end refusal, transport, replay, provenance, and local naming.

The Lean TasteGate boundary is `lean4/BEDC/Derived/FiberFarEndDigestBoundaryUp/TasteGate.lean`. It must provide `Nontrivial FiberFarEndDigestBoundaryUp` and `FieldFaithful FiberFarEndDigestBoundaryU` for the displayed digest, fiber, far-end, GAP, refusal, transport, replay, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiberFarEndDigestBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiberFarEndDigestBoundary}^{\uparrow}$ packet is generated from a visible digest value, source-fiber row, apophatic far-end boundary row, GAP provenance, digest/far-end refusal, componentwise transport, displayed replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiberFarEndDigestBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8953.1, Definition 8953.2, Definition 8953.3, the falsifiable prediction, the independence witness, and the TasteGate boundary line seed the visible-digest versus fiber-far-end distinction.

Formal verification: FormalStatus(NameCert$_{\text{FiberFarEndDigestBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close source equality from digest equality, selected preimages, internal far-end objects, global universe digests, quotient fiber collapse, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, digest exposure, fiber exposure, apophatic far-end boundary exposure, GAP provenance, digest/far-end refusal, transport, replay, provenance, local naming, and TasteGate instances for BEDC.Derived.FiberFarEndDigestBoundaryUp.

Chapter 8954

A Concrete Naming Certificate for $\mathsf{GromovProduct}^{\uparrow}$

Following Gromov, the $\mathsf{GromovProduct}^{\uparrow}$ packet records the basepointed metric comparison row

$$(x, y)_b$$

as finite BEDC NameCert data. It sits beside the metric source of chapter 57, the real-distance seal of chapter 38, the dyadic bound carrier of chapter 1562, the hyperbolic consumer of chapter 6073, and the branch-comparison route of chapter 7303. The packet names only displayed distance reads, basepoint membership, dyadic comparison budgets, transport, replay, provenance, and local naming; it is not an ambient completion, selected geodesic, quotient boundary, or host hyperbolic space.

Definition 8954.1 (Gromov product carrier)**.** *A* $\mathsf{GromovProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, b, x, y, D, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the distance observations available to the packet.*
2. *b, x, y are the displayed basepoint and endpoint rows over the same metric source.*
3. *D is the* $\mathsf{Real}^{\uparrow}$ *distance-readback row for $d(x, b)$, $d(y, b)$, and $d(x, y)$.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row, recording the finite tolerances used to read the half-sum expression and any downstream inequality window.*
5. *H records componentwise* hsame *transport for the metric source, point rows, distance reads, dyadic budgets, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from metric admission through the three distance reads and the dyadic comparison row.*
7. *P is the* Pkg *provenance over* $\mathsf{GromovProduct}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GromovProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, b, x, y, D, T, H, C, P, N$. It has no coordinate for ambient completeness, selected geodesics, quotient boundary points, host hyperbolicity, or external Real equality.

Theorem 8954.2 (Gromov product NameCert obligations)**.** *For every accepted* $\mathsf{GromovProduct}^{\uparrow}$ *carrier*

$$G = (M, b, x, y, D, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GromovProduct}^{\uparrow}}$ *surface consists exactly of metric-source admission through* M*, basepoint and endpoint exposure through* b, x, y*,* $\mathsf{Real}^{\uparrow}$ *distance readback through* D*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8954.1. The metric row M is the only source of distance observations, and the point rows b, x, y are displayed before any product comparison is read.

Read the distance and budget rows next. The row D exposes exactly the three $\mathsf{Real}^{\uparrow}$ distance reads needed for the basepointed comparison, and T records the finite dyadic tolerances used by downstream inequality consumers.

The structural rows H, C, P, N preserve the same finite packet. Since no row supplies ambient completeness, selected geodesics, quotient boundary points, host hyperbolicity, or external Real equality, the listed rows exhaust the local NameCert obligations. □

Theorem 8954.3 (Gromov product metric-completion handoff)**.** *Every* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$*,* $\mathsf{RealTree}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{GromovProduct}^{\uparrow}$ *packet receives only the finite route*

$$M, b, x, y, D, T, H, C, P, N.$$

The consumer may inspect displayed metric distances, basepointed comparison data, and dyadic tolerance rows, but it cannot read ambient completeness, selected geodesic representatives, quotient boundary equality, or a host hyperbolic structure from the packet.

Proof. Start with the NameCert obligations of Definition 8954.2. They force every read to pass through the metric source, the three point rows, the Real distance row, and the dyadic comparison row before structural replay is available.

The hyperbolic sibling in chapter 6073 may consume the displayed basepointed comparison as one row in a boundary-shadow packet. The real-tree sibling in chapter 7303 may consume the same distance comparison as a branch or tripod budget. A completion-facing consumer may carry the row forward only through the displayed metric and Real readbacks.

Transport, replay, provenance, and local naming remain attached to M, b, x, y, D, T. Thus the handoff is a metric-completion neighbour rather than a completed metric object, a chosen geodesic model, or a quotient-boundary construction. □

Falsifiable prediction: An accepted $\mathsf{GromovProduct}^{\uparrow}$ read is refuted whenever it uses a basepointed product without displaying the common $\mathsf{Met}^{\uparrow}$ source, the three point rows, the three $\mathsf{Real}^{\uparrow}$ distance reads, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row.

Independence witness: metric, real, dyadic-rat-core, hyperbolic-gromov-product, real-tree: no listed sibling is bijective to $\mathsf{GromovProduct}^{\uparrow}$ because this packet jointly binds the metric source, basepoint, endpoints, distance readbacks, dyadic comparison budget, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GromovProduct}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GromovProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8954.1, Definition 8954.2, and Definition 8954.3 seed the basepointed metric product packet over displayed metric, point, Real-distance, dyadic-tolerance, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GromovProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient completions, selected geodesics, quotient boundaries, host hyperbolic structures, external Real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric-source admission, point exposure, Real distance readback, dyadic tolerance exposure, transport, replay, provenance, and local naming for BEDC.Derived.GromovProductUp.

Constructive story: A GromovProduct$^{\uparrow}$ packet is generated from a MetricUp source row, displayed basepoint and endpoint rows, RealUp distance readbacks, a DyadicRatCoreUp tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8955

A Concrete Naming Certificate for $\mathsf{SolvableGroup}^{\uparrow}$

Following Rotman and Hall on finite group theory, the $\mathsf{SolvableGroup}^{\uparrow}$ packet records solvability by a finite normal derived-series certificate rather than by an ambient classification theorem. It sits over the $\mathsf{Group}^{\uparrow}$ carrier of chapter 41, the subgroup and normal-subgroup rows of chapter 80, the quotient rows of chapter 82, and the abelian quotient surface of chapter 42. The downstream consumers are the Galois obstruction in chapter 285 and the radical-tower interface in chapter 286.

Definition 8955.1 (Solvable group carrier)**.** *A* $\mathsf{SolvableGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, D, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *G is a* $\mathsf{Group}^{\uparrow}$ *source row: carrier, multiplication, unit, inverse, classifier, and local group NameCert data.*
2. *D is a finite derived-series row*

$$G = D_0 \rhd D_1 \rhd \cdots \rhd D_m = \{e\},$$

   where each D_{i+1} is the commutator subgroup row of D_i and each inclusion is recorded as a $\mathsf{NormalSubgroup}^{\uparrow}$ *witness over the same* $\mathsf{Group}^{\uparrow}$ *carrier.*
3. *Q lists the quotient-group rows D_i/D_{i+1} with their* $\mathsf{QuotientGroup}^{\uparrow}$ *source, classifier, transport, and ledger coordinates.*
4. *A lists the* $\mathsf{AbGroup}^{\uparrow}$ *witnesses showing that every quotient row D_i/D_{i+1} is abelian.*
5. *H records componentwise* hsame *transport for the group row, derived-series rows, normal-subgroup inclusions, quotient rows, and abelian quotient witnesses.*
6. *C records displayed* Cont *replay from the group source row through derived-series construction, normality checking, quotient formation, abelian quotient witnessing, and terminal trivial-subgroup exposure.*
7. *P is the* Pkg *provenance over* $\mathsf{SolvableGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{NormalSubgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, *derived-series rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SolvableGroup}^{\uparrow}}$ *row.*

The classifier compares solvable-group packets only through G, D, Q, A, H, C, P, N. It has no coordinate for a classification of all finite groups, Galois correspondence, radical solvability, polynomial roots, or an unstored simplicity theorem.

Theorem 8955.2 (Solvable group NameCert obligations)**.** *For every* $\mathsf{SolvableGroup}^{\uparrow}$ *packet*

$$S = (G, D, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SolvableGroup}^{\uparrow}}$ *surface consists exactly of group-source exposure, finite derived-series display, normal-subgroup inclusion rows, quotient-group rows, abelian quotient witnesses, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. A downstream BEDC consumer may cite solvability only through those finite rows, and may not replace them by a general classification of finite groups, a Galois correspondence theorem, radical-tower adequacy, or a polynomial solution claim.*

Proof. Project the coordinates of Definition 8955.1. The group row G, derived-series row D, quotient row list Q, and abelian witness list A are displayed coordinates of the same finite packet.

The certificate obligations are inherited rowwise. The $\mathsf{Group}^{\uparrow}$ source supplies the ambient group carrier; the $\mathsf{Subgroup}^{\uparrow}$ and $\mathsf{NormalSubgroup}^{\uparrow}$ rows expose the series inclusions; the $\mathsf{QuotientGroup}^{\uparrow}$ rows expose each quotient; and the $\mathsf{AbGroup}^{\uparrow}$ rows witness the abelian quotient condition.

Transport, replay, provenance, and naming are supplied by H, C, P, N. Since no coordinate stores a finite-group classification, a Galois correspondence, radical-tower adequacy, or a polynomial solution claim, the local NameCert obligations stop at the displayed solvable-group packet. □

Theorem 8955.3 (Solvable group derived-series handoff)**.** *The derived-series row D of a* $\mathsf{SolvableGroup}^{\uparrow}$ *packet supplies the exact finite handoff needed by* $\mathsf{AbelRuffini}^{\uparrow}$ *and* $\mathsf{SolvableRadicals}^{\uparrow}$*: it exposes a normal series, the quotient rows D_i/D_{i+1}, abelian quotient witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. The handoff does not assert that a Galois group has been computed, that a radical tower exists, or that a polynomial is solvable by radicals.*

Proof. Read the finite series from D and the quotient list from Q. For each adjacent pair, the packet records normality of D_{i+1} in D_i and a quotient row D_i/D_{i+1}.

Read the abelian witness list A. These rows are the only solvability witnesses exported by the packet: they say that the displayed quotients are abelian in the sense of the local $\mathsf{AbGroup}^{\uparrow}$ surface.

The handoff to $\mathsf{AbelRuffini}^{\uparrow}$ and $\mathsf{SolvableRadicals}^{\uparrow}$ is therefore a finite carrier route, not a semantic theorem about polynomials or field towers. The transport, replay, provenance, and naming rows H, C, P, N keep the downstream read confined to the displayed derived-series data. □

Closure status for $\mathsf{NameCert}_{\mathsf{SolvableGroup}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SolvableGroup}^{\uparrow}}$, obligation)

Scope closed: Definition 8955.1, Definition 8955.2, and Definition 8955.3 bind the finite derived-series carrier, normal quotient rows, abelian quotient witnesses, transport, replay, provenance, and local naming surface for solvable groups.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SolvableGroup}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a classification of finite groups, Galois correspondence, radical-tower adequacy, polynomial root extraction, or Lean verification.

Upgrade path: Formal closure requires carrier admission, quotient-row exactness, abelian quotient witness replay, and downstream handoff proofs for BEDC.Derived.SolvableGroupUp.

Constructive story: A $\mathsf{SolvableGroup}^{\uparrow}$ packet is generated from a $\mathsf{Group}^{\uparrow}$ source row, a finite normal derived series, quotient rows, abelian quotient witnesses, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 8956

A Concrete Naming Certificate for $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$

Following Mac Lane on comparison functors and Bishop–Bridges/Kelley on constructive completion, the $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ packet records the finite BEDC route from the Kleisli presentation of Cauchy completion to the Eilenberg–Moore algebra presentation. It sits after the completion monad of chapter 378, the Kleisli category of chapter 3826, the Eilenberg–Moore category of chapter 7845, the categorical source rows of chapter 61, chapter 62, and chapter 63, and the L10 completion readback route of chapter 126, chapter 301, chapter 291, and chapter 38. The packet compares the two category-level completion interfaces through displayed finite rows; it is not a quotient completion, a selected representative construction, host equality of completed objects, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8956.1 (Cauchy completion comparison functor carrier)**.** *A* $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, K, E, B, F, A, T, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row, exposing the unit, bind, finite windows, regular-rational readback, dyadic tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route.*
2. *K is the* $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ *source row, displaying completion objects, unit arrows, Kleisli arrows, bind composition, and its real-completion handoff.*
3. *E is the* $\mathsf{EilenbergMooreCategory}^{\uparrow}$ *target row, displaying algebra objects, structure maps, law rows, homomorphisms, and composition ledgers.*
4. *B is the base* $\mathsf{Cat}^{\uparrow}$ *row shared by the Kleisli source and the Eilenberg–Moore target.*
5. *F is the displayed* $\mathsf{Functor}^{\uparrow}$ *comparison row from K to E.*
6. *A is the* $\mathsf{NatTrans}^{\uparrow}$ *action row, recording the unit and multiplication compatibility of the comparison functor with the completion monad.*
7. *T is the L10 readback row through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any completion-facing consumer reads the comparison.*
8. *L is the comparison ledger, recording that objects, arrows, unit routes, bind routes, and algebra law routes are compared only through M, K, E, B, F, A, T.*

9. *H records componentwise* hsame *transport for monad, Kleisli, Eilenberg–Moore, base-category, functor, naturality, readback, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay for each comparison read from the monad route through the source and target category rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$, $\mathsf{EilenbergMooreCategory}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, E, B, F, A, T, L, H, C, P, N$. It has no coordinate for quotient completion, host equality of completed objects, choice-selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8956.2 (Cauchy completion comparison functor NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ *carrier*

$$J = (M, K, E, B, F, A, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}$ *surface consists exactly of carrier exposure, completion-monad exposure, Kleisli source exposure, Eilenberg–Moore target exposure, base-category exposure, comparison-functor exposure, naturality and monad-compatibility exposure, L10 readback exposure, comparison-ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8956.1. The monad row M, the Kleisli row K, the Eilenberg–Moore row E, and the base row B are displayed before any comparison can be consumed.

The functor row F and naturality row A then record the comparison itself. They read the unit and bind schedules from M, the source composition route from K, and the algebra-law route from E.

The L10 readback row T keeps every completion-facing request on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. The ledger L records that object, arrow, unit, bind, and algebra-law comparisons are all made through those visible rows.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no quotient-completion, host-equality, representative-choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 8956.3 (Cauchy completion comparison functor handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ *packet receives the comparison functor only through the route*

$$\mathsf{CauchyCompletionMonad}^{\uparrow},\ \mathsf{CauchyCompletionKleisliCategory}^{\uparrow},\ \mathsf{EilenbergMooreCategory}^{\uparrow},\ \mathsf{Cat}^{\uparrow},\ \mathsf{Functor}^{\uparrow},\ \mathsf{NatTrans}^{\uparrow}$$

The consumer may inspect the displayed unit, bind, Kleisli composition, algebra structure, functorial action, and naturality rows, but it cannot read a quotient completion, host equality, choice-selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem from the packet.*

Proof. Begin with the completion monad row M. It supplies the unit and bind schedules together with finite stream windows, regular-rational readback, dyadic tolerance rows, and the terminal Real seal route.

Read the source and target category rows next. The Kleisli row K exposes objects, arrows, and bind composition, while the Eilenberg–Moore row E exposes algebra objects, structure maps, law rows, homomorphisms, and composition ledgers over the same base row B.

The comparison rows F and A are then available as displayed functor and naturality data. Their consumer route is recorded by L and must pass through the L10 readback row T before any Real-facing seal is consumed.

The structural rows H, C, P, N keep the handoff inside the displayed packet. Because the excluded objects are absent from the carrier inventory, no consumer can obtain them from the comparison functor route. □

Theorem 8956.4 (Cauchy completion comparison functor nonescape)**.** *No consumer of an accepted* $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ *carrier may export quotient completion equality, host equality of completed objects, a choice-selected representative, an ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or an external comparison between the Kleisli and Eilenberg–Moore presentations. The consumer may export only the displayed comparison surface on* $M, K, E, B, F, A, T, L, H, C, P, N$ *and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}$ *evidence attached to those rows.*

Proof. Inspect the carrier inventory in Definition 8956.1. The tuple contains only the completion monad, Kleisli source, Eilenberg–Moore target, base category, functor, naturality, L10 readback, comparison ledger, and structural naming rows.

Definition 8956.2 exhausts the local obligation surface. Every allowed read is tied to one of the displayed rows, and no excluded coordinate is introduced by the obligation theorem.

Definition 8956.3 fixes the completion-facing route through the monad, categorical, functorial, naturality, stream, regular-rational, dyadic, and Real rows. Hence any accepted export remains inside the finite comparison packet. □

Falsifiable prediction: An accepted $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ read is refuted whenever it compares Kleisli and Eilenberg–Moore completion presentations without displaying the completion monad, both category rows, the comparison functor row, the naturality row, the L10 stream, regular-rational, dyadic, and Real readback rows, and the local transport, replay, provenance, and naming rows.

Independence witness: cauchycompletionmonad, cauchy-completion-kleisli-category, eilenberg-moore-category, category, functor, nattrans, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ because this packet jointly binds the completion monad, Kleisli source, Eilenberg–Moore target, base category, functor action, naturality, L10 readback, comparison ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}$ packet is generated from a CauchyCompletionMonad row, a CauchyCompletionKleisliCategory source row, an EilenbergMooreCategory target row, a Category base row, Functor and NatTrans comparison rows, StreamName, RegSeqRat, DyadicRatCore, and Real readback rows, a comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8956.1, Definition 8956.2, Definition 8956.3, and Definition 8956.4 seed the finite L10 comparison-functor packet from the Kleisli completion presentation to the Eilenberg–Moore algebra presentation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host equality of completed objects, choice-selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monad exposure, Kleisli source exposure, Eilenberg–Moore target exposure, functorial action, naturality, L10 readback, comparison-ledger exactness, transport stability, replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyCompletionComparisonFunctorUp.

Chapter 8957

A Concrete Naming Certificate for $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$

Following Bishop and Bridges' diagonal selection practice for constructive Cauchy arguments, the $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ packet records the finite route by which a regular tail selector, a cofinal stream selector, and a tail-meet ledger are fused before a Real and Completion consumer reads a terminal seal. It sits over the regular tail-selection surface of $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, the cofinal stream route of $\mathsf{CofinalStreamTailSelector}^{\uparrow}$, the finite meet row of chapter 560, the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names Bishop-style diagonal tail selection as finite data; it is not a selected completed limit, quotient stream equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 8957.1 (CauchyTailSelectorFusion carrier)**.** *A* $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, C, M, D, W, Q, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *row recording the regular tail threshold selected for the displayed tolerance request.*
2. *C is the* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *row recording the cofinal stream window through which the selected tail is inspected.*
3. *M is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row binding the regular selector and the cofinal stream selector to one common tail window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that stores the requested precision, the selector budget, and the meet budget.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row on which the source stream, selected tail, and common meet are simultaneously readable.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational data extracted from D, W, S, C, M.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the selector, cofinal, meet, dyadic, stream, and regular-rational rows have been displayed.*
8. *H records componentwise* hsame *transport for every displayed object row, the provenance row, and the local naming row.*

9. *K records displayed* Cont *replay from the regular selector and cofinal selector through the meet row and into the regular-rational and real-seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{CofinalStreamTailSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailSelectorFusion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, C, M, D, W, Q, R, H, K, P, N$. It exports no selected limit, quotient stream equality, trichotomy of real names, countable-choice schedule, hidden tail oracle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8957.2 (CauchyTailSelectorFusion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ *carrier*

$$F = (S, C, M, D, W, Q, R, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailSelectorFusion}^{\uparrow}}$ *surface consists exactly of regular selector admission through S, cofinal stream selector admission through C, common-tail meet exposure through M, dyadic tolerance accounting through D, finite stream-window readability through W, regular rational readback through Q, terminal real sealing through R, componentwise transport through H, continuation replay through K, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8957.1. The rows S, C, M are the selector rows: S chooses the regular tail threshold for the displayed tolerance request, C chooses the cofinal stream window, and M records their finite meet.

The numerical and completion-facing rows are D, W, Q, R. The dyadic ledger D stores the precision budgets, the stream row W exposes the finite inspection window, and Q, R hand the displayed rational data to the terminal real seal.

The structural rows are H, K, P, N. Transport preserves each displayed coordinate, replay factors the diagonal tail route through the selector and meet rows, and provenance plus naming repack the same finite packet. Since none of the excluded host objects is a carrier coordinate, these rows exhaust the local NameCert surface. □

Theorem 8957.3 (CauchyTailSelectorFusion selector coherence)**.** *Every accepted* $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ *packet exposes a coherent diagonal tail route*

$$S, C, M, D, W, Q, R, K$$

in which the regular tail selector and the cofinal stream selector are both consumed before the common meet row is read, and the regular-rational and real-seal handoff is consumed only after that meet row has fixed the finite tail window.

Proof. Begin with the two selector rows. The regular selector S answers the tolerance request, while the cofinal selector C supplies the stream window on which that answer can be inspected.

Next read the meet row M. It is the only row that binds the two selector outputs to one common finite tail window, so no downstream regular-rational read can bypass the fusion point.

Finally consume D, W, Q, R, K. The dyadic ledger and stream window display the data read at the common tail, the regular-rational row translates it into the accepted rational readback surface, and the terminal real seal is reached only along the replay row K. Thus the diagonal selector route remains finite and coherent inside the displayed packet. □

Theorem 8957.4 (CauchyTailSelectorFusion non-escape)**.** *The public information exported by a* $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ *carrier is exactly*

$$S, C, M, D, W, Q, R, H, K, P, N.$$

In particular, no consumer can extract a selected completed limit, quotient stream equality, trichotomy of real names, countable-choice schedule, hidden tail oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem from the packet.*

Proof. Inspect the carrier coordinates. The only object rows are the regular selector, cofinal selector, meet, dyadic tolerance, stream-window, regular-rational readback, and terminal real-seal rows.

The structural rows add no hidden object. The transport row preserves the displayed coordinates, the replay row factors the same selector-fusion route, and provenance plus local naming identify the same packet.

Therefore any request for a selected limit, quotient stream equality, trichotomy, countable choice, hidden tail oracle, or ambient completeness theorem reaches no carrier coordinate and no replay coordinate. The packet exports only the finite selector-fusion surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailSelectorFusion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailSelectorFusion}^{\uparrow}$ packet is generated from regular tail selector, cofinal stream selector, regular tail meet, dyadic tolerance, stream-window, regular-rational readback, terminal Real seal, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailSelectorFusion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8957.1, Definition 8957.2, Definition 8957.3, and Definition 8957.4 seed the finite Bishop-style diagonal tail selector fusion route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailSelectorFusion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, trichotomy, countable choice, hidden tail oracles, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked selector admission, cofinal stream admission, meet exposure, dyadic tolerance accounting, finite-window readability, regular-rational handoff, terminal Real sealing, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchyTailSelectorFusionUp.

Chapter 8958

A Concrete Naming Certificate for $\mathsf{KreinMilman}^{\uparrow}$

Following Krein and Milman, the $\mathsf{KreinMilman}^{\uparrow}$ packet records the theorem-level NameCert wrapper that reads the compact-convex sibling of chapter 7460 together with the finite-extreme sibling of chapter 3610. The packet names the public theorem boundary as finite BEDC rows; it does not import host choice, selected supporting functionals, quotient compactness, or ambient locally convex completeness.

Definition 8958.1 (KreinMilman carrier)**.** *A* $\mathsf{KreinMilman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, E, F, W, H, R, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$ *sibling row, reading the compact, locally convex, convex-set, weak-window, face, scalar, transport, replay, provenance, and naming surface of chapter 7460;*
2. *E is the* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *sibling row, reading the exposed-face, candidate extreme-point, separating-functional, scalar, metric, transport, replay, provenance, and naming surface of chapter 3610;*
3. *F is the finite theorem-facing hull ledger tying the compact-convex row to the finite extreme witness row;*
4. *W is the weak-window compatibility row through which theorem-level consumers read the compact and extreme-point siblings in one packet;*
5. *H records componentwise* hsame *transport for the compact-convex, finite-extreme, hull-ledger, weak-window, replay, provenance, and naming rows;*
6. *R records displayed* Cont *replay through the same finite theorem-facing route;*
7. *P is the* Pkg *provenance over* $\mathsf{KreinMilman}^{\uparrow}$, $\mathsf{KreinMilmanCompactConvex}^{\uparrow}$, $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KreinMilman}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host Hahn–Banach theorem, selected extreme point, quotient convex body, global choice principle, or ambient compactness theorem.

Lean4 (statement) `BEDC.Derived.KreinMilmanUp`

Theorem 8958.2 (KreinMilman theorem-level NameCert obligations)**.** *For every accepted* $\mathsf{KreinMilman}^{\uparrow}$ *carrier*

$$K = (C, E, F, W, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KreinMilman}^{\uparrow}}$ *surface consists exactly of compact-convex sibling exposure through* C*, finite-extreme sibling exposure through* E*, finite hull-ledger visibility through* F*, weak-window compatibility through* W*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* R*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 8958.1. The compact-convex row and finite-extreme row are the only theorem-level mathematical inputs, and they are read from the two displayed sibling chapters.

The hull ledger and weak-window row then bind those siblings into one theorem-facing packet. They record finite BEDC dependency data rather than a selected extreme point or an ambient host proof.

The structural rows H, R, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for choice, host Hahn–Banach, quotient convex bodies, selected representatives, or ambient compactness, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreinMilman}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KreinMilman}^{\uparrow}$ packet is generated from compact-convex sibling rows, finite-extreme sibling rows, a theorem-facing hull ledger, weak-window compatibility, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KreinMilman}^{\uparrow}}$, seed)

Scope closed: Definition 8958.1 and Definition 8958.2 seed the theorem-level wrapper over the compact-convex and finite-extreme Krein–Milman siblings.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KreinMilman}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Hahn–Banach, selected extreme points, quotient convex bodies, global choice principles, ambient compactness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-convex sibling exposure, finite-extreme sibling exposure, hull-ledger visibility, weak-window compatibility, transport, replay, provenance, and local naming for BEDC.Derived.KreinMilmanUp.

Chapter 8959

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$

The $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ packet realizes the reversible observer-phase transport branch of chapter 9463 as finite NameCert data. It sits over the disk-metric readout of chapter 6209, the visual-boundary surface of chapter 4181, the observer-expansion route of chapter 4167, the observer-phase packet of chapter 7112, the phase-transport budget of chapter 7134, the boundary-shadow packet of chapter 3813, and the complex phase-fiber row of chapter 6044. The packet names reversible disk-phase, boundary-shadow, visual-boundary, provenance, and refusal rows; it is not a completed hyperbolic disk, a physical observer, a quotient boundary, a chosen global phase, or a cosmological transport law.

Definition 8959.1 (HyperbolicObserverPhaseTransport carrier)**.** *A* $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, M, V, O, G, A, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the Poincare disk-coordinate row read through* $\mathsf{PoincareDiskMetric}^{\uparrow}$*.*
2. *M is the hyperbolic metric comparison row over the same displayed disk data.*
3. *V is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row displaying the visible boundary face.*
4. *O is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row exposing the observer-expansion window.*
5. *G is the* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *row recording reversible composition of observer-phase arrows.*
6. *A is the* $\mathsf{HyperbolicObserverPhase}^{\uparrow}$ *phase-read row consumed by the transport.*
7. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row binding the transport to a finite boundary shadow.*
8. *F is the* $\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}$ *provenance-fiber row that prevents the phase read from becoming a hidden source equality.*
9. *E is the* $\mathsf{HyperbolicPhaseTransportBudget}^{\uparrow}$ *energy and curvature-budget row for the accepted transport.*
10. *H records componentwise* hsame *transport for every displayed object row, provenance, and local naming.*

11. *C records displayed* Cont *replay through disk metric, visual boundary, observer expansion, groupoid composition, phase read, boundary shadow, fiber provenance, and transport budget.*

12. *P is the* Pkg *provenance over* HyperbolicObserverPhaseTransport$^{\uparrow}$, PoincareDiskMetric$^{\uparrow}$, HyperbolicVisualBoundary HyperbolicObserverExpansion$^{\uparrow}$, ObserverPhaseGroupoid$^{\uparrow}$, HyperbolicObserverPhase$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$ ComplexPhaseSurfaceFiber$^{\uparrow}$, HyperbolicPhaseTransportBudget$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, V, O, G, A, B, F, E, H, C, P, N$. It exports no completed disk, quotient visual boundary, selected global phase, physical observer, cosmological transport law, or hidden fiber equality.

Theorem 8959.2 (HyperbolicObserverPhaseTransport NameCert obligations)**.** *For every accepted* HyperbolicObserverPhaseTransport$^{\uparrow}$ *carrier*

$$T = (D, M, V, O, G, A, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission through D, metric comparison through M, visual-boundary exposure through V, observer-expansion exposure through O, reversible observer-phase groupoid composition through G, phase readback through A, boundary-shadow binding through B, provenance-fiber exposure through F, phase-transport budget accounting through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8959.1. The rows D, M, V, O are the geometric source rows: disk coordinates, metric comparison, visual boundary, and observer expansion must all be displayed before transport is consumed.

The phase rows are G, A, B, F, E. The groupoid row records reversible observer-phase composition, the phase-read row is the local phase surface, the boundary-shadow row ties it to the far boundary, the fiber row records provenance, and the budget row accounts for the accepted transport.

The structural rows are H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed rows. Since no excluded host object appears as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8959.3 (HyperbolicObserverPhaseTransport boundary coherence)**.** *Every accepted* HyperbolicObserverP *read exposes boundary-shadow and provenance rows before phase transport is consumed. In particular, the route*

$$D, M, V, O, G, A \longmapsto B, F, E$$

is coherent only when B records the finite boundary shadow, F records the provenance fiber, and E records the transport budget before any downstream consumer reads the transported observer phase.

Proof. Start with the disk and visual rows. The disk coordinate, metric comparison, visual-boundary, and observer-expansion rows determine the visible surface on which the phase read is meaningful.

Next consume the reversible phase rows. The observer-phase groupoid row and phase-read row give a transport candidate, but they do not yet license a boundary-level consumer.

Finally read B, F, E. The boundary-shadow row binds the candidate to the visual boundary, the fiber row displays provenance, and the budget row records the accepted transport cost. A consumer

that skips one of these rows has no displayed boundary-shadow or provenance coordinate, so it cannot consume the transported phase as an accepted packet. □

Theorem 8959.4 (HyperbolicObserverPhaseTransport non-escape)**.** *The public information exported by a* $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ *carrier is exactly*

$$D, M, V, O, G, A, B, F, E, H, C, P, N.$$

Thus a consumer cannot extract a completed hyperbolic disk, quotient visual boundary, selected global phase, physical observer, cosmological transport law, or hidden fiber equality from the packet.

Proof. Inspect the carrier rows. The object rows are exactly the disk, metric, visual-boundary, observer-expansion, groupoid, phase-read, boundary-shadow, fiber, and budget rows.

The structural rows preserve the same inventory. Componentwise transport, displayed replay, provenance, and local naming do not add a completed disk, quotient boundary, selected phase, physical observer, cosmological law, or hidden fiber equality.

Therefore every public read factors through the displayed packet and the excluded objects remain outside the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ read must expose boundary-shadow and provenance rows before phase transport is consumed.

Independence witness: poincare-disk-metric, hyperbolic-visual-boundary, observer-phase-groupoid, hyperbolic-observer-expansion, complex-phase-surface-fiber: no listed sibling is bijective to HyperbolicObserverPhase because this packet jointly carries disk metric, visual-boundary, observer-expansion, reversible observer-phase groupoid, phase read, boundary-shadow, provenance-fiber, budget, transport, replay, provenance, and local NameCert rows.

The Lean TasteGate for this packet will require explicit Nontrivial $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ instances before a checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}$ packet is generated from disk-coordinate, metric, visual-boundary, observer-expansion, observer-phase groupoid, phase-read, boundary-shadow, provenance-fiber, transport-budget, hsame, Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8959.1, Definition 8959.2, Definition 8959.3, Definition 8959.4, the falsifiable prediction, and the independence witness seed the finite observer-phase transport surface over disk, boundary, phase, provenance, and refusal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic disks, quotient visual boundaries, selected global phases, physical observers, cosmological transport laws, hidden fiber equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk and metric visibility, visual-boundary exposure, observer-expansion exposure, groupoid composition, phase readback, boundary-shadow binding, fiber provenance, transport-budget exactness, TasteGate nontriviality, field faithfulness, replay stability, provenance exhaustion, and local naming for BEDC.Derived.HyperbolicObs

Chapter 8960

A Concrete Naming Certificate for BishopDiagonalProductCauchy$^{\uparrow}$

Following Bishop and Bridges, the BishopDiagonalProductCauchy$^{\uparrow}$ packet records the finite product-diagonal route by which two regular Cauchy reads are synchronized before a Real and Completion consumer asks for a product observation. It sits next to the dyadic tolerance core of chapter 291, the finite stream scheduler of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the tail comparison surface of chapter 387. The packet is a finite-means product certificate: it displays the common window, the two readback rows, the dyadic product budget, and the Real seal, without quotient sequence equality, selected representatives, countable choice, trichotomy, or ambient completeness of Real$^{\uparrow}$.

Definition 8960.1 (Bishop diagonal product Cauchy carrier)**.** *A* BishopDiagonalProductCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (W_X, W_Y, D_X, D_Y, R_X, R_Y, T, M, S, H, C, G, N)$$

with the following displayed rows:

1. W_X *and* W_Y *are finite* StreamName$^{\uparrow}$ *window rows for the two product coordinates.*
2. D_X *and* D_Y *are* DyadicRatCore$^{\uparrow}$ *tolerance rows attached to the same requested product scale.*
3. R_X *and* R_Y *are* RegSeqRat$^{\uparrow}$ *readback rows served from* W_X, D_X *and* W_Y, D_Y*.*
4. T *is the* CauchyTailComparison$^{\uparrow}$ *row witnessing that the two coordinates are compared only after a common finite tail threshold has been displayed.*
5. M *is the diagonal-product row. It records the finite rule that reads both coordinates at the shared threshold and forms the product observation from the displayed rational data.*
6. S *is the downstream* Real$^{\uparrow}$ *seal row for the accepted product observation.*
7. H *records componentwise* hsame *transport for the two windows, two tolerance rows, two readbacks, tail comparison, product row, Real seal, replay, provenance, and local naming.*
8. C *records displayed* Cont *replay from the requested product scale through the common window, dyadic tolerances, regular readbacks, tail comparison, product row, and Real seal.*
9. G *is the* Pkg *provenance over* BishopDiagonalProductCauchy$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyTailComparison$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalProductCauchy}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient sequence equality, selected representatives, hidden tail choice, countable-choice schedules, trichotomy, host real multiplication, host equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 8960.2 (Bishop diagonal product Cauchy carrier exactness)**.** *Every accepted* BishopDiagonalProductCauc *carrier exposes exactly the rows*

$$W_X, W_Y, D_X, D_Y, R_X, R_Y, T, M, S, H, C, G, N$$

before a product observation is served. In particular, the product route is determined by finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *common-tail row, the diagonal-product row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the carrier of Definition 8960.1. The analytic rows are the two stream windows, two dyadic tolerance rows, two regular readbacks, the tail comparison row, the product row, and the Real seal.

The structural rows are separate. Row H transports the displayed rows componentwise, row C replays the finite route, row G records provenance, and row N names the packet.

No other coordinate occurs in the carrier. Therefore an accepted product observation can be served only after the listed rows have been displayed. □

Theorem 8960.3 (Bishop diagonal product Cauchy classifier)**.** *Let* $P = (W_X, W_Y, D_X, D_Y, R_X, R_Y, T, M, S, H, C$ *be an accepted* $\mathsf{BishopDiagonalProductCauchy}^{\uparrow}$ *carrier. Any classifier read of P first consumes the common finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyTailComparison}^{\uparrow}$$

on both coordinates, then reads M and S. Thus the classifier may use the displayed product observation and its $\mathsf{Real}^{\uparrow}$ *seal, but it cannot branch through quotient Cauchy classes, host sequence equality, selected representatives, trichotomy, hidden tail choice, countable choice, or ambient Real completeness.*

Proof. Start with the two coordinate routes. The classifier must read W_X, D_X, R_X for the first coordinate and W_Y, D_Y, R_Y for the second coordinate before the common-tail row T is meaningful.

The tail comparison row T supplies the shared finite threshold. Only downstream of that row can M form the product observation and S serve the $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and naming rows preserve the same order of consumption. Since none of the carrier coordinates is a quotient Cauchy class, host sequence equality, selected representative, trichotomy branch, hidden tail choice, choice schedule, or ambient-completeness theorem, no accepted classifier read can branch through those objects. □

Theorem 8960.4 (Bishop diagonal product Cauchy ledger)**.** *The local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalProductCauchy}^{\uparrow}}$ *ledger for an accepted packet consists exactly of the following five rows:*

1. *two finite* $\mathsf{StreamName}^{\uparrow}$ *windows and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows;*
2. *two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks served from the displayed windows and tolerances;*
3. *one* $\mathsf{CauchyTailComparison}^{\uparrow}$ *common-tail row and one diagonal-product row;*
4. *the downstream* $\mathsf{Real}^{\uparrow}$ *seal with componentwise* hsame *transport and displayed* Cont *replay;*

5. Pkg *provenance and local naming.*

These rows exhaust the finite product-diagonal obligation surface.

Proof. The first two ledger rows are the coordinate data. They are exactly the windows, tolerances, and readbacks named by $W_X, W_Y, D_X, D_Y, R_X, R_Y$.

The third row is the product synchronization surface. The common-tail row T fixes the shared finite threshold, and the row M records the product observation at that threshold.

The fourth and fifth rows are structural and terminal. The Real seal S, transports H, replay C, provenance G, and local name N account for every remaining coordinate. No additional obligation row is present in the carrier, so the ledger is exhausted. □

Falsifiable prediction: Every accepted $\mathsf{BishopDiagonalProductCauchy}^{\uparrow}$ product read displays two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{CauchyTailComparison}^{\uparrow}$ common-tail row, a diagonal-product row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-tail-comparison: $\mathsf{BishopDiagonalProductCauchy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two synchronized coordinate routes, a common-tail comparison row, a diagonal-product row, a Real seal, and structural NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalProductCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDiagonalProductCauchy}^{\uparrow}$ packet is generated from two StreamName windows, two DyadicRatCore tolerance rows, two RegSeqRat readbacks, a CauchyTailComparison common-tail row, a diagonal-product row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDiagonalProductCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8960.1, Definition 8960.2, Definition 8960.3, and Definition 8960.4 seed the finite product-diagonal Cauchy route over StreamName, DyadicRatCore, RegSeqRat, CauchyTailComparison, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDiagonalProductCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient sequence equality, selected representatives, hidden tail choice, countable choice, trichotomy, host real multiplication, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier determinacy, ledger exactness, product-window stability, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.BishopDiagonalProductCauchy

Chapter 8961

A Concrete Naming Certificate for $\mathsf{PrueferTransformation}^{\uparrow}$

Following the Sturm–Liouville and Pruefer phase-transform tradition, the $\mathsf{PrueferTransformation}^{\uparrow}$ packet records the finite phase-angle read by which a displayed second-order real differential system is routed from amplitude and derivative rows into a Sturm-facing oscillation surface. It sits after the Wronskian determinant corridor of chapter 6877, the separation consumer of chapter 8378, the root-counting consumer of chapter 3612, the derivative-admission route of chapter 122, the continuous-map admission route of chapter 130, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a displayed finite phase-angle carrier; it does not import full spectral theory, selected eigenfunctions, host phase equality, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8961.1 (Pruefer transformation carrier)**.** *A* $\mathsf{PrueferTransformation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, D, A, \Theta, W, S, Z, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source function-family row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *before any phase request is inspected.*
2. *D is the derivative-admission row, exposing the first derivative and second-order source data only through* $\mathsf{Derivative}^{\uparrow}$*.*
3. *A is the amplitude ledger, recording the displayed radius-squared and nonzero-window requests used before a phase angle is named.*
4. *Θ is the phase-angle row, listing the finite angle windows produced from the displayed amplitude and derivative rows.*
5. *W is the* $\mathsf{Wronskian}^{\uparrow}$ *determinant handoff row, keeping zero and nonzero separation reads attached to the finite determinant ledger.*
6. *S is the Sturm consumer row, available only through the* $\mathsf{SturmSeparation}^{\uparrow}$ *and* $\mathsf{SturmRootCount}^{\uparrow}$ *sibling routes.*
7. *Z is the zero-window discipline row, recording which displayed windows may be treated as phase-crossing candidates.*

8. *R is the regular rational readback row through* $\mathsf{RegSeqRat}^{\uparrow}$.
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and R have been displayed.*
10. *H records componentwise* hsame *transport for the source, derivative, amplitude, phase, Wronskian, Sturm, zero-window, readback, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a phase request through source admission, derivative admission, amplitude readback, Wronskian handoff, Sturm consumer routing, dyadic tolerance, regular rational readback, Real sealing, transport, provenance, and local naming.*
12. *P is the* Pkg *provenance over* $\mathsf{PrueferTransformation}^{\uparrow}$, $\mathsf{Wronskian}^{\uparrow}$, $\mathsf{SturmSeparation}^{\uparrow}$, $\mathsf{SturmRootCount}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PrueferTransformation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host phase function, a selected eigenfunction, quotient real equality, countable choice, full spectral decomposition, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8961.2 (Pruefer transformation NameCert obligations)**.** *For every accepted* $\mathsf{PrueferTransformation}^{\uparrow}$ *carrier*

$$T = (F, D, A, \Theta, W, S, Z, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrueferTransformation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, continuous source admission through F, derivative admission through D, amplitude and phase-window exposure through A, Θ, Z, Wronskian determinant handoff through W, Sturm separation and root-count consumer routing through S, dyadic tolerance and regular rational readback through R, E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, local naming through N, and nonescape from full spectral theory, selected eigenfunctions, host phase equality, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8961.1. The tuple displays source functions, derivative rows, amplitude windows, phase windows, Wronskian handoff, Sturm consumer routing, zero-window discipline, regular rational readback, Real sealing, transport, replay, provenance, and local naming as separate rows.

The source and differentiability data are read first. The rows F and D force every phase request through $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{Derivative}^{\uparrow}$, so the carrier cannot begin from an unnamed host solution or an external differentiability witness.

Next inspect A, Θ, Z, W, S. These rows expose only finite amplitude and phase windows, route determinant separation through $\mathsf{Wronskian}^{\uparrow}$, and send Sturm-facing reads through the displayed $\mathsf{SturmSeparation}^{\uparrow}$ and $\mathsf{SturmRootCount}^{\uparrow}$ sibling surfaces.

Finally pass through R, E, H, C, P, N. The regular rational readback, dyadic tolerance, terminal Real seal, transport, replay, provenance, and naming rows preserve the same finite route. Since no row supplies spectral decomposition, selected eigenfunctions, host phase equality, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8961.3 (Pruefer transformation Wronskian–Sturm handoff)**.** *Every* $\mathsf{SturmSeparation}^{\uparrow}$ *or* $\mathsf{SturmRootCount}^{\uparrow}$ *consumer that reads a* $\mathsf{PrueferTransformation}^{\uparrow}$ *phase window must first expose the*

Wronskian row W, the source and derivative rows F, D, the amplitude and phase rows A, Θ, Z, and the Real-facing readback rows R, E. The consumer receives the handoff only as a displayed route through Wronskian$^{\uparrow}$, Derivative$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *with no host zero theorem, selected eigenfunction, host phase equality, quotient real equality, countable choice, or ambient completeness.*

Proof. Start with a Sturm-facing phase request. The carrier makes the phase row Θ downstream of the source row F, derivative row D, and amplitude ledger A, so no phase read is available before the continuous and derivative admissions have been displayed.

Use the Wronskian row W to route zero and nonzero separation through the determinant corridor. The consumer therefore reads the same finite Wronskian surface before it can treat a phase crossing as a separation or root-count input.

Now inspect the Sturm row S. It sends the read to the SturmSeparation$^{\uparrow}$ and SturmRootCount$^{\uparrow}$ sibling chapters as consumers of the displayed finite route, not as new coordinates inside the transformation packet.

Finally apply the readback and seal rows R, E together with H, C, P, N. They preserve the handoff under regular rational readback, dyadic tolerance, Real sealing, componentwise transport, replay, provenance, and local naming. The refused endpoints are absent because none of these rows names a host zero theorem, selected eigenfunction, host phase equality, quotient real equality, countable-choice schedule, or ambient completeness principle. □

Falsifiable prediction: Every accepted PrueferTransformation$^{\uparrow}$ read displays continuous source rows, derivative rows, amplitude and phase windows, Wronskian determinant handoff, Sturm separation and root-count consumer routing, dyadic tolerance, regular rational readback, terminal Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: wronskian, sturm-separation, sturm-root-count, derivative, continuousmap, dyadicratcore, regseqrat, real: PrueferTransformation$^{\uparrow}$ is not bijective to any listed sibling because it binds amplitude and phase windows, Wronskian determinant handoff, zero-window discipline, Sturm consumer routing, and Real-facing readback into one finite NameCert packet.

Closure status for NameCert$_{\text{PrueferTransformation}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{PrueferTransformation}^{\uparrow}}$, seed)

Scope closed: Definition 8961.1, Definition 8961.2, and Definition 8961.3 seed the finite source, derivative, amplitude, phase-window, Wronskian, Sturm-consumer, dyadic, readback, Real-seal, transport, replay, provenance, and local naming route.

Formal verification: FormalStatus(NameCert$_{\text{PrueferTransformation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full spectral theory, selected eigenfunctions, host phase equality, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous and derivative admission, amplitude and phase-window exactness, Wronskian determinant handoff, Sturm consumer routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PrueferTransformationUp.

Constructive story: A PrueferTransformation$^{\uparrow}$ packet is generated from displayed source rows, derivative rows, amplitude and phase windows, a Wronskian determinant handoff, Sturm separation and root-count consumer routing, dyadic tolerance, regular rational readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 8962

A Concrete Naming Certificate for BishopLocatedCauchyCriterion$^{\uparrow}$

Following Bishop and Bridges, the BishopLocatedCauchyCriterion$^{\uparrow}$ packet records the finite located Cauchy test that lets the L10 Real and Completion chain read a rational stream as a real-name candidate without quotient streams, ambient completeness, trichotomy, or a selected limit. It sits after the dyadic arithmetic of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the modulus surface of chapter 8512, and the terminal real-name seal of chapter 38. Its role is to name the located comparison criterion that later RealSequenceLimit$^{\uparrow}$ and Bishop sequence-limit packets can consume before any completion theorem is available.

Definition 8962.1 (Bishop located Cauchy criterion carrier)**.** *A* BishopLocatedCauchyCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, L, D, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence-source row, recording the rational stream only through a named* StreamName$^{\uparrow}$ *observation route.*
2. *L is the located-comparison row, recording the finite comparison by which two observed rational values are witnessed to be within or apart from a requested dyadic tolerance.*
3. *D is the dyadic tolerance-request row, read through* DyadicRatCore$^{\uparrow}$ *and aligned with the modulus scale of* CauchySequenceModulus$^{\uparrow}$*.*
4. *W is the finite* StreamName$^{\uparrow}$ *window row requested by D.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row produced from the displayed windows and dyadic tolerance request.*
6. *R is the* Real$^{\uparrow}$ *seal row reached only after S, L, D, W, Q have been displayed together.*
7. *H records componentwise* hsame *transport for the source, located comparison, tolerance request, windows, readback, real seal, provenance, and naming rows.*
8. *C records displayed* Cont *replay through source observation, located comparison, dyadic tolerance, stream windows, regular readback, real sealing, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* BishopLocatedCauchyCriterion$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchySequenceModulus$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}}$ *row.*
The carrier contains no quotient-stream equality, no countable choice schedule, no hidden tail selector, no ambient completeness principle, no trichotomy on reals, and no selected host limit.

Theorem 8962.2 (Bishop located Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}$ *carrier*

$$B = (S, L, D, W, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}}$ *surface consists exactly of source admission through S, located comparison through L, dyadic tolerance request through D, finite window observation through W, regular rational readback through Q, real sealing through R, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

These rows are the complete finite criterion surface. A consumer may read the Cauchy test only by displaying the tolerance request, the windows answering it, the regular rational readback, and the located comparison that keeps the test apart from quotient-stream equality.

Proof. Project the carrier of Definition 8962.1. Its object rows are S, L, D, W, Q, R, and its structural rows are H, C, P, N.

The dyadic and stream rows form the finite request boundary. Row D names the tolerance scale, row W names the finite windows requested at that scale, and row Q is the only rational readback exported from those observations.

The located comparison row L is the criterion row. It relates the two readback values to the dyadic request without adding quotient-stream equality, a selected tail, or a total order on real names.

Finally R, H, C, P, N seal, transport, replay, source, and name the same finite packet. Since no coordinate names quotient equality, countable choice, hidden tail selection, ambient completeness, real trichotomy, or a host limit, the listed rows exhaust the local NameCert obligations. □

Theorem 8962.3 (Bishop located Cauchy criterion real handoff)**.** *Every accepted Real and Completion consumer of a* $\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchySequenceModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff keeps the located comparison row L attached to the tolerance request D, window row W, readback row Q, and support rows H, C, P, N.

Proof. Start with the finite tolerance request. The row D supplies the dyadic scale, and the row W records the finite stream windows requested at that scale.

Read back the rational data next. The row Q is served through $\mathsf{RegSeqRat}^{\uparrow}$, so the criterion sees regular rational values rather than a quotient stream or a host sequence.

Attach the located comparison. The row L compares the readback values at the requested dyadic tolerance, while the modulus sibling $\mathsf{CauchySequenceModulus}^{\uparrow}$ records why that comparison is the Cauchy criterion visible to the packet.

Only after those rows are present may the consumer read the real seal R. The structural rows H, C, P, N carry the same handoff under transport, replay, provenance, and local naming, so no ambient completeness theorem or selected host limit enters the route. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}$ consumer displays source, located comparison, dyadic tolerance, finite window, regular readback, Real seal, transport, replay, provenance, and local naming rows before treating the rational stream as a real-name candidate.

Independence witness: dyadicratcore, streamname, regseqrat, cauchysequencemodulus, real: $\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}$ is not bijective to any listed sibling because it carries the located comparison row and the Real handoff row together with the tolerance, window, readback, transport, replay, provenance, and local naming rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}$ packet is generated from sequence source, located comparison, dyadic tolerance request, StreamName window, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8962.1, Definition 8962.2, Definition 8962.3, the falsifiable prediction, and the independence witness seed the finite located Cauchy criterion route for the L10 Real and Completion chain.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, countable choice, hidden tail selection, ambient completeness, real trichotomy, selected host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-comparison exactness, dyadic tolerance request stability, StreamName window coverage, RegSeqRat readback stability, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCauchyCriterionUp.

Chapter 8963

A Concrete Naming Certificate for $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$

Following Bishop and Bridges on located real numbers and regular rational approximations, the $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ packet records the finite cofinal-bracket surface by which a located real read is inspected through nested dyadic windows. It sits after the located-real carrier of **??**, the located-interval carrier of chapter 394, the regular rational readback of chapter 301, the stream-window surface of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the finite bracket windows and their cofinal refinement ledger; it does not assert quotient real equality, trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8963.1 (LocatedReal cofinal bracket carrier)**.** *A* $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, I, S, Q, D, W, E, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *source row whose locatedness requests are being bracketed.*
2. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row recording the current rational interval face around the inspected read.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observed prefixes used by the bracket.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint-comparison row.*
6. *W is the cofinal bracket-window ledger, recording displayed pairs $q_k^- \leq q_k^+$ and the finite refinement relation between consecutive windows.*
7. *E is the endpoint-exposure row, recording that each displayed bracket endpoint is read through I, S, Q, D before any real-facing seal is consumed.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after L, I, S, Q, D, W, E have been displayed.*
9. *H records componentwise* hsame *transport for located-real, located-interval, stream, regular-rational, dyadic, bracket, endpoint, real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay for each bracket read from locatedness requests through stream windows, rational readback, dyadic comparison, and real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCofinalBracket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, I, S, Q, D, W, E, R, H, C, P, N$. It has no coordinate for quotient real equality, trichotomy, a selected limiting real, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8963.2 (LocatedReal cofinal bracket NameCert obligations)**.** *For every accepted* LocatedRealCofinalBrac *carrier*

$$B = (L, I, S, Q, D, W, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealCofinalBracket}^{\uparrow}}$ *surface consists exactly of located-real source exposure through L, located-interval exposure through I, finite stream-window exposure through S, regular rational readback through Q, dyadic tolerance and endpoint comparison through D, cofinal bracket-window accounting through W, endpoint exposure through E, terminal real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8963.1. The located source row L supplies the displayed locatedness request, while I supplies the interval face in which the current bracket is read.

The finite readback rows are S, Q, D, W, E. Stream windows expose inspected prefixes, regular rational readback turns those windows into rational data, the dyadic row records endpoint comparison and tolerance, the bracket ledger records cofinal refinement, and the endpoint row records that endpoints are consumed only through those displayed rows.

The real seal R is downstream of the bracket data, and H, C, P, N preserve, replay, source, and name the same finite packet. Since the carrier has no coordinate for quotient equality, trichotomy, selected limits, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8963.3 (LocatedReal cofinal bracket nested readback)**.** *Every accepted* $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ *read factors through a nested finite route*

$$\mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto W \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes each requested bracket window and its refinement relation before the terminal real seal may be used. It exports no quotient real equality, trichotomy, selected limiting representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with L and I. A located-real request is visible only as a displayed source row together with a located interval face, so no real-facing consumer can skip directly to the terminal seal.

Next read S, Q, D. The stream row supplies finite observations, the regular rational row supplies the corresponding rational readback, and the dyadic row supplies tolerance and endpoint comparison data.

Then read W and E. The bracket ledger records the nested cofinal windows, and the endpoint row records that the endpoints used by the bracket were produced by the displayed finite route.

Finally consume R, H, C, P, N. These rows seal, transport, replay, source, and name the same packet without adding any quotient, trichotomy, selection, choice, or completeness coordinate. □

Theorem 8963.4 (LocatedReal cofinal bracket nonescape)**.** *The public information exported by an accepted* $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ *carrier is exactly*

$$L, I, S, Q, D, W, E, R, H, C, P, N.$$

Thus a consumer may inspect finite cofinal bracket windows and their nested readback route, but cannot extract quotient real equality, trichotomy, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Inspect the carrier inventory. The rows L, I, S, Q, D, W, E, R are the only mathematical data rows, and the rows H, C, P, N are the structural transport, replay, provenance, and naming rows.

Definition 8963.2 fixes the NameCert obligations to exactly those rows. Definition 8963.3 fixes the order in which real-facing reads may consume them.

Any quotient equality, trichotomy decision, selected limiting representative, countable-choice schedule, or completeness theorem would require a coordinate outside the displayed packet. Since no such coordinate is present, accepted exports remain inside the finite cofinal-bracket surface. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ read is refuted whenever it reaches the terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying the located-real source, located-interval face, stream window, regular rational readback, dyadic tolerance row, cofinal bracket-window ledger, endpoint row, transport, replay, provenance, and local naming rows.

Independence witness: locatedreal, locatedinterval, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective to $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ because this packet jointly binds located reads, interval faces, finite stream windows, regular rational readback, dyadic endpoint comparison, cofinal bracket refinement, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealCofinalBracket}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealCofinalBracket}^{\uparrow}$ packet is generated from LocatedReal source rows, LocatedInterval faces, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance and endpoint comparisons, cofinal bracket-window ledgers, endpoint-exposure rows, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealCofinalBracket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8963.1, Definition 8963.2, Definition 8963.3, and Definition 8963.4 seed the finite cofinal-bracket packet for located real reads through nested dyadic windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealCofinalBracket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, interval-face exactness, stream-window readback, regular rational readback, dyadic endpoint comparison, cofinal bracket refinement, endpoint exposure, real-seal nonescape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.LocatedRealCofinalBracketUp.

Chapter 8964

A Concrete Naming Certificate for $\mathsf{NewtonForwardSeries}^{\uparrow}$

Following Boole and Whittaker–Robinson, the $\mathsf{NewtonForwardSeries}^{\uparrow}$ packet records the finite forward-difference series route by which a displayed table is repacked as a Newton coefficient row and then handed to polynomial and regular-Cauchy consumers. It sits after the finite-difference surface of chapter 5135, the finite-difference table of chapter 7065, the polynomial carrier of chapter 50, and the regular rational readback of chapter 301. The packet supplies a finite-calculus interface for L10 Real and Completion estimates; it does not claim analytic convergence, quotient real equality, selected limits, host function extensionality, or ambient completeness.

Definition 8964.1 (Newton forward series carrier)**.** *A* $\mathsf{NewtonForwardSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, D, C, P, R, W, H, T, G, N)$$

with the following displayed rows:

1. *X records the basepoint row, step row, and finite source-window observations used by the forward table.*
2. *D is the forward-difference table row, read through the finite-difference and finite-difference-table siblings before any coefficient is exposed.*
3. *C is the Newton coefficient row, storing the finite binomial-basis coefficients determined by D.*
4. *P is the polynomial reconstruction row, reading C as a finite polynomial or finite Newton-series section through* $\mathsf{Poly}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for window estimates derived from the displayed finite section.*
6. *W is the regular-Cauchy window compatibility row used by L10 Real and Completion consumers.*
7. *H records componentwise* hsame *transport for the basepoint, step, table, coefficient, reconstruction, readback, window, provenance, and local naming rows.*
8. *T is the displayed* Cont *route*

$$X, D \longmapsto C \longmapsto P \longmapsto R \longmapsto W.$$

9. *G is the* Pkg *provenance over* $\mathsf{NewtonForwardSeries}^{\uparrow}$*,* $\mathsf{FiniteDifference}^{\uparrow}$*,* $\mathsf{FiniteDifferenceTable}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, regular-Cauchy window rows,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{NewtonForwardSeries}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite analytic series, a host function graph, quotient real equality, selected convergence, Taylor remainder completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8964.2 (Newton forward series NameCert obligations)**.** *For the carrier of Definition 8964.1, the local* $\mathsf{NameCert}_{\mathsf{NewtonForwardSeries}^{\uparrow}}$ *surface consists exactly of the finite obligation rows for basepoint and step admission through X, forward-difference table admission through D, Newton coefficient exposure through C, polynomial or finite-series reconstruction through P, regular readback and window compatibility through R, W, componentwise transport through H, displayed continuation through T, provenance through G, and local naming through N.*

Lean4 (statement) `BEDC.Derived.NewtonForwardSeriesUp`

Proof. Project the carrier rows of Definition 8964.1. The basepoint, step, and source-window data are displayed in X, and the finite forward-difference table is displayed in D before any coefficient row can be read.

The coefficient row C is then the only route to the polynomial or finite-series reconstruction row P. A polynomial consumer therefore receives a finite Newton section rather than an ambient function graph or an infinite analytic series.

The regular readback row R and the window row W are downstream of the same finite table and coefficient rows. Componentwise transport, continuation replay, provenance, and local naming preserve this displayed route and do not add quotient real equality, selected convergence, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonForwardSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{NewtonForwardSeries}^{\uparrow}$ packet is generated from a basepoint and step row, a finite forward-difference table, a Newton coefficient row, a polynomial or finite-series reconstruction row, RegSeqRat readback, regular-Cauchy window compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NewtonForwardSeries}^{\uparrow}}$, seed)

Scope closed: Definition 8964.1 and Definition 8964.2 seed the finite Newton forward-series packet over finite-difference, polynomial, regular readback, regular-Cauchy window, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NewtonForwardSeries}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic convergence, infinite series completeness, quotient real equality, selected limits, host function extensionality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked basepoint and step admission, forward-difference table admission, Newton coefficient exposure, polynomial reconstruction, regular readback, regular-Cauchy window compatibility, transport, provenance, and local naming for BEDC.Derived.NewtonForwardSeriesUp.

Chapter 8965

A Concrete Naming Certificate for MetaCICGeneratorClosureAlgebra$^{\uparrow}$

The MetaCICGeneratorClosureAlgebra$^{\uparrow}$ packet realizes the generator-closure classifier algebra of Definition 9389.8 and Definition 9389.9 as finite MetaCIC NameCert rows. It sits beside the generator-closure certificate of chapter 486, the MetaCIC closure-preservation packet of chapter 788, the MetaCIC audit matrix of chapter 904, and the MetaCIC identity packet of chapter 860. The packet names identity, composition, alternation, restriction, finite iteration, product, and sum closure rows for classifier-preserving generators; it does not assert unrestricted induction, arbitrary higher-order closure, proof irrelevance, quotient syntax equality, or host evaluator soundness.

Definition 8965.1 (MetaCIC generator-closure algebra carrier)**.** *A* MetaCICGeneratorClosureAlgebra$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, K, R, C, O, W, T, P, S, L, H, Q, G, N)$$

with the following displayed rows:

1. *U is the carrier row naming the fixed MetaCIC object family on which classifiers are read.*
2. *K is the classifier row, recording the displayed predicate preserved by the generator rows.*
3. *R is the identity generator row corresponding to* `GeneratorClosureClassifier.refl`*.*
4. *C is the composition row corresponding to* `GeneratorClosureClassifier.comp`*.*
5. *O is the alternation row corresponding to* `GeneratorClosureClassifier.union`*.*
6. *W is the restriction row corresponding to* `GeneratorClosureClassifier.weaken`*.*
7. *T is the finite-iteration row corresponding to* `GeneratorClosureClassifier.iterate`*.*
8. *P is the product row corresponding to* `GeneratorClosureClassifier.product`*.*
9. *S is the sum row corresponding to* `GeneratorClosureClassifier.sum`*.*
10. *L is the algebra ledger, recording that the seven closure rows are consumed only as classifier-preserving generator operations.*
11. *H records componentwise* hsame *transport for carrier, classifier, algebra-operation, ledger, replay, provenance, and local naming rows.*

12. *Q records displayed* Cont *replay through the identity, composition, alternation, restriction, finite-iteration, product, and sum rows.*
13. *G is the* Pkg *provenance over* MetaCICGeneratorClosureAlgebra$^{\uparrow}$, GeneratorClosure$^{\uparrow}$, MetaCICClosurePreservation$^{\uparrow}$ MetaCICAuditMatrix$^{\uparrow}$, MetaCICIdentity$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* NameCert$_{\text{MetaCICGeneratorClosureAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, K, R, C, O, W, T, P, S, L, H, Q, G, N$. It has no coordinate for unrestricted induction, arbitrary higher-order generator closure, proof irrelevance, quotient syntax equality, or host evaluator soundness.

Theorem 8965.2 (MetaCIC generator-closure algebra NameCert obligations)**.** *For every accepted* MetaCICGeneratorClosureAlgebra$^{\uparrow}$ *carrier*

$$A = (U, K, R, C, O, W, T, P, S, L, H, Q, G, N),$$

the local NameCert$_{\text{MetaCICGeneratorClosureAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission through U, classifier exposure through K, identity closure through R, composition closure through C, alternation closure through O, restriction closure through W, finite-iteration closure through T, product closure through P, sum closure through S, algebra-ledger exactness through L, componentwise transport through H, displayed replay through Q, package provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 8965.1. The rows U and K fix the carrier and classifier whose preservation is being certified.

Read the seven algebra rows next. Identity, composition, alternation, restriction, finite iteration, product, and sum are each displayed as a classifier-preserving generator operation, and none is read as a free-standing induction principle.

The ledger L records that all seven rows are consumed only through that generator-closure interpretation. Finally H, Q, G, N transport, replay, source, and name the same packet. Since the carrier contains no unrestricted induction, higher-order closure, proof-irrelevance, quotient-syntax, or host-evaluator coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 8965.3 (MetaCIC generator-closure algebra operation readback)**.** *Every accepted* MetaCICGeneratorClos *read of a compound generator must factor through the finite operation route*

$$U, K \longmapsto R, C, O, W, T, P, S \longmapsto L, H, Q, G, N.$$

The route may certify identity, composition, alternation, restriction, finite iteration, product, and sum preservation for displayed classifiers, but it cannot be read as unrestricted induction or as closure under arbitrary generator formation.

Proof. Begin with U, K. A generator-closure operation has no public meaning until the carrier and classifier rows are fixed.

Next read the operation rows R, C, O, W, T, P, S. Each row records one finite algebra operation from the classifier principle and preserves the displayed classifier by the corresponding generator rule.

Then read the ledger and structural rows. The ledger keeps the operation inside the seven displayed cases, while H, Q, G, N transport, replay, source, and name the packet. No step adds a row for arbitrary generator formation or unrestricted induction. □

Theorem 8965.4 (MetaCIC generator-closure algebra nonescape)**.** *The public information exported by an accepted* MetaCICGeneratorClosureAlgebra$^{\uparrow}$ *carrier is exactly*

$$U, K, R, C, O, W, T, P, S, L, H, Q, G, N.$$

Thus downstream MetaCIC consumers may cite the displayed generator-closure algebra rows, but must separately supply any typing, conversion, confluence, normalization, quotient-syntax, or host-evaluator evidence they consume.

Proof. Inspect the carrier inventory. The only operation rows are identity, composition, alternation, restriction, finite iteration, product, and sum, all tied to the displayed carrier and classifier.

Definition 8965.2 fixes the NameCert surface to those rows. Definition 8965.3 fixes the route by which compound generators may be consumed.

Typing, conversion, confluence, normalization, quotient syntax, and host evaluator soundness are not coordinates of A. Any downstream use of those facts must cite a separate packet, so this algebra surface does not escape beyond classifier-preserving generator closure. □

Falsifiable prediction: Any accepted MetaCICGeneratorClosureAlgebra$^{\uparrow}$ read that treats identity, composition, alternation, restriction, finite iteration, product, or sum closure as unrestricted induction or arbitrary generator formation violates the algebra ledger.

Independence witness: generatorclosure, metacicclosurepreservation, metacic-audit-matrix, metacicidentity: no listed sibling is bijective to MetaCICGeneratorClosureAlgebra$^{\uparrow}$ because this packet jointly carries carrier and classifier rows, seven algebra-operation rows, ledger exactness, transport, replay, provenance, and local naming.

The Lean TasteGate expectation for this packet is explicit: a checked realization must provide a non-smoke Nontrivial MetaCICGeneratorClosureAlgebra$^{\uparrow}$ witness, a FieldFaithful MetaCICGeneratorClosureAlgebra$^{\uparrow}$ witness for the displayed rows $U, K, R, C, O, W, T, P, S, L, H, Q, G, N$, and a carrier-alignment theorem for the same fields before public TasteGate use.

Closure status for NameCert$_{\text{MetaCICGeneratorClosureAlgebra}^{\uparrow}}$.

Constructive story: A MetaCICGeneratorClosureAlgebra$^{\uparrow}$ packet is generated from a MetaCIC carrier row, a classifier row, identity, composition, alternation, restriction, finite-iteration, product, and sum generator-closure rows, an algebra ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICGeneratorClosureAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 8965.1, Definition 8965.2, Definition 8965.3, Definition 8965.4, the falsifiable prediction, the independence witness, and the TasteGate expectation seed the finite MetaCIC generator-closure algebra packet.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICGeneratorClosureAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted induction, arbitrary higher-order generator closure, proof irrelevance, quotient syntax equality, host evaluator soundness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, identity closure, composition closure, alternation closure, restriction closure, finite-iteration closure, product closure, sum closure, algebra-ledger exactness, TasteGate nontriviality, field faithfulness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.MetaCICGeneratorClosureAlgebraUp.

Chapter 8966

A Concrete Naming Certificate for $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$

Following Schauder, Brouwer, and Bishop–Bridges finite-net fixed-point arguments, the SchauderFiniteNetFixedPoin packet records the finite BEDC route by which a compact convex continuous self-map exposes an approximate fixed-point net before any endpoint consumer may read a fixed-point boundary. It sits over the Schauder packet of chapter 1923, the finite approximate fixed-point handoff of chapter 6087, the finite compact-map residual route of chapter 8184, the compact metric surface of chapter 129, the convex carrier of chapter 208, and the continuous-map row of chapter 130. The packet names finite nets, displacement ledgers, residual rows, transport, replay, provenance, and local naming; it is not a host fixed-point selector, open-cover compactness theorem, quotient equality of approximants, countable-choice schedule, or ambient completeness principle.

Definition 8966.1 (Schauder finite-net fixed-point carrier)**.** *A* $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, Q, F, M, A, E, D, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the finite compact-metric inspection windows used by the net.*
2. *Q is the* $\mathsf{ConvexSet}^{\uparrow}$ *row, recording membership and finite affine-combination stability for the compact convex carrier.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row, carrying graph access, source membership, target membership, and displayed continuity data.*
4. *M is the* $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$ *residual route, binding compact metric, convex, continuous-map, modulus, finite-net, displacement, and residual reads into one compact-map packet.*
5. *A is the* $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$ *handoff row, listing the accepted finite mesh, coverage, displacement tolerance, and approximate fixed-point read.*
6. *E is the finite-net exposure row, naming the centers, cells, tolerance requests, and projection reads actually inspected by the packet.*
7. *D is the displacement ledger comparing each inspected net point with its image through F at the displayed tolerance.*

8. *R is the fixed-point residual row exported toward* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *after* K, Q, F, M, A, E, D *have been displayed.*
9. *H records componentwise* hsame *transport for the compact source, convex carrier, self-map, compact-map route, approximate handoff, net exposure, displacement ledger, residual row, replay, provenance, and local naming row.*
10. *C records displayed* Cont *replay from a fixed-point request through* $K, Q, F, M, A, E, D, R, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$, $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, Q, F, M, A, E, D, R, H, C, P, N$. *It has no coordinate for a selected fixed point, host compactness, open-cover compactness, quotient equality of approximants, countable choice, or ambient completeness.*

Theorem 8966.2 (Schauder finite-net fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$ *carrier*

$$T = (K, Q, F, M, A, E, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}}$ *surface consists exactly of compact metric admission through K, convex carrier admission through Q, continuous self-map admission through F, compact-map residual routing through M, finite approximate fixed-point handoff through A, finite-net exposure through E, displacement-ledger comparison through D, fixed-point residual export through R, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and non-escape from host fixed-point selection, open-cover compactness, quotient approximants, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 8966.1. The accepted packet displays exactly the rows $K, Q, F, M, A, E, D, R, H, C, P, N$, so no consumer can read compactness, convexity, continuity, finite-net data, displacement, residual evidence, transport, replay, provenance, or naming outside this list.

Read the geometric source rows first. The compact metric row K supplies the inspected finite windows, the convex row Q supplies membership and affine-combination stability, and the continuous-map row F supplies the self-map data over that same displayed carrier.

The approximation route then passes through M and A. The compact-map row M packages the residual route of chapter 8184, while the approximate fixed-point row A supplies the finite mesh and tolerance handoff of chapter 6087.

Finally E, D, R, H, C, P, N expose the inspected net, compare displacements, export the residual row, transport the same components, replay the displayed request, record provenance, and register the local name. Since none of those rows names a host fixed-point selector, open-cover compactness theorem, quotient approximant, countable-choice schedule, or ambient completeness object, the listed obligations exhaust the local certificate. □

Theorem 8966.3 (Schauder finite-net approximate route)**.** *Every* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *consumer of an accepted* $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$ *carrier factors through the finite route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SchauderFiniteCompactMap}^{\uparrow}$, $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$, BHist

The consumer may inspect the residual row R only after the compact metric source, convex carrier, continuous self-map, compact-map residual route, finite approximate fixed-point handoff, finite-net exposure, displacement ledger, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start with a fixed-point consumer. The continuation row C is the only replay route, and it displays K, Q, F, M, A, E, D before R can be consumed.

The source side is finite and carried. Rows K and Q bind the compact convex carrier, while F gives the displayed self-map over that carrier. These rows are the only route to the compact metric, convex, and continuous-map siblings cited by the packet.

The fixed-point side is approximate. Row M carries the finite compact-map residual route, row A carries the finite approximate fixed-point handoff, row E exposes the net actually inspected, and row D records the displacement comparisons that feed R.

Thus the residual read exported to $\mathsf{SchauderFixedPoint}^{\uparrow}$ is the finite route displayed in the theorem. The transport, replay, provenance, and local naming rows preserve that route and add no selected fixed point, quotient equality, open-cover compactness, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}$ packet is generated from CompactMetric source rows, ConvexSet carrier rows, ContinuousMap self-map rows, SchauderFiniteCompactMap residual rows, FiniteApproximateFixedPoint handoff rows, finite-net exposure rows, displacement ledgers, residual rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8966.1, Definition 8966.2, and Definition 8966.3 seed the finite-net fixed-point packet over compact metric, convex, continuous-map, compact-map residual, finite approximate fixed-point, displacement, residual, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderFiniteNetFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close exact fixed-point selection, a host Schauder theorem, open-cover compactness, quotient equality of approximants, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric exposure, convex carrier exposure, continuous-map replay, compact-map residual routing, approximate fixed-point handoff, finite-net exposure, displacement-ledger exactness, residual export, transport, replay, provenance, and local naming for BEDC.Derived.SchauderFiniteNetFixedPoin

Chapter 8967

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequence arithmetic, the $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$ packet records the finite window reassociation route used when a regular rational product or paired completion read is regrouped before reaching the terminal Real seal. It sits over the finite schedule surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the product rows of chapter 464 and chapter 461, and the value-facing boundary of chapter 38. The packet is a finite reassociation certificate for displayed windows; it does not assert associativity of completed real multiplication, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8967.1 (Regular Cauchy window associator carrier)**.** *A* $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (W_L, W_M, W_R, D_L, D_M, D_R, Q_L, Q_M, Q_R, P_L, P_R, S, H, C, G, N)$$

with the following displayed rows:

1. W_L, W_M, W_R *are the three finite* $\mathsf{StreamName}^{\uparrow}$ *windows being reassociated.*
2. D_L, D_M, D_R *are the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows at the shared requested scale.*
3. Q_L, Q_M, Q_R *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows served from those windows and tolerances.*
4. P_L *and* P_R *are the left- and right-parenthesized product-window rows read through* $\mathsf{CauchyProduct}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$.
5. S *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the three windows, three tolerance rows, three readbacks, and both parenthesized product rows remain visible.*
6. H *records componentwise* hsame *transport for the windows, tolerances, readbacks, product rows, seal, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from a reassociation request through the same finite rows.*
8. G *is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowAssociator}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient sequence equality, selected representatives, hidden infinite reassociation, countable choice, host multiplication, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8967.2 (Regular Cauchy window associator NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$ *carrier*

$$A = (W_L, W_M, W_R, D_L, D_M, D_R, Q_L, Q_M, Q_R, P_L, P_R, S, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowAssociator}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, three finite* $\mathsf{StreamName}^{\uparrow}$ *windows, three* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, three* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the two parenthesized product-window rows* P_L, P_R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and nonescape from quotient stream equality, selected representatives, hidden infinite reassociation, countable choice, host multiplication, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8967.1. The tuple displays the three windows, three dyadic tolerance rows, three regular rational readbacks, two parenthesized product rows, terminal seal, transport, replay, provenance, and local naming as separate coordinates.

The product rows are downstream of the finite windows. Row P_L reads the left-parenthesized window route, and row P_R reads the right-parenthesized route, both through the existing CauchyProduct and RegularCauchyProduct sibling surfaces.

Finally inspect S, H, C, G, N. The Real seal, componentwise transport, replay, provenance, and local naming preserve the same displayed rows. Since no carrier coordinate stores quotient stream equality, selected representatives, hidden infinite reassociation, countable choice, host multiplication, trichotomy, or ambient completeness, the listed rows exhaust the NameCert surface. □

Theorem 8967.3 (Regular Cauchy window associator RegSeqRat handoff)**.** *Every RegSeqRat or Real-facing consumer of an accepted* $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$ *carrier must expose the three* $\mathsf{StreamName}^{\uparrow}$ *windows, the three* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the three* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and both parenthesized product-window rows before it may inspect the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The reassociation handoff is therefore a finite displayed route through StreamName, RegSeqRat, DyadicRatCore, CauchyProduct, RegularCauchyProduct, and Real rows, not an ambient completeness or quotient-stream principle.*

Proof. Start with a RegSeqRat-facing reassociation request. The readback rows Q_L, Q_M, Q_R are meaningful only after the corresponding windows W_L, W_M, W_R and tolerance rows D_L, D_M, D_R have been exposed.

Use the product rows P_L and P_R to compare the two parenthesized routes. Both rows are product-window data attached to the same three displayed windows and dyadic scales, so neither can introduce a hidden product representative or a host multiplication coordinate.

The terminal seal S is then read only through the preserved finite route. Transport, replay, provenance, and local naming keep that route attached to the accepted packet, leaving no coordinate for quotient stream equality, selected representatives, countable choice, trichotomy, host multiplication, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyWindowAssociator}^{\uparrow}$ read displays three $\mathsf{StreamName}^{\uparrow}$ windows, three $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, three $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two parenthesized product-window rows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchyproduct, regularcauchyproduct, real: RegularCauchyWindowAssociator$^{\uparrow}$ is not bijective to any listed sibling because it carries both parenthesized product-window routes over one three-window regular Cauchy readback surface.

Closure status for NameCert$_{\text{RegularCauchyWindowAssociator}^{\uparrow}}$.

Constructive story: A RegularCauchyWindowAssociator$^{\uparrow}$ packet is generated from three StreamName windows, three DyadicRatCore tolerance rows, three RegSeqRat readbacks, two parenthesized product-window rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyWindowAssociator}^{\uparrow}}$, seed)

Scope closed: Definition 8967.1, Definition 8967.2, and Definition 8967.3 seed the finite reassociation route over StreamName, RegSeqRat, DyadicRatCore, product, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyWindowAssociator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, associativity of completed real multiplication, quotient stream equality, selected representatives, hidden infinite reassociation, countable choice, trichotomy, host multiplication, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, product-window exactness, RegSeqRat handoff determinacy, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.RegularCauchyWindowAssociatorU

Chapter 8968

A Concrete Naming Certificate for $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$

The $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ packet realizes the transported stationary diagonal route as finite NameCert data rather than as a completeness principle. It reads a displayed observation window through chapter 126, the regular rational boundary of chapter 301, the dyadic tolerance ledger of chapter 291, the diagonal seal of chapter 4241, and the terminal real-name boundary of chapter 38. The router records only the finite path by which a diagonal row is inspected; it does not select limits, quotient streams, host real representatives, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8968.1 (Finite observation diagonal router carrier)**.** *A* $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, Q, D, \Delta, E, S, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window for the transported diagonal row.*
2. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface consumed at that window.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same observation request.*
4. Δ *is the diagonal-routing row, recording the finite transport from the displayed window to the selected diagonal read.*
5. E *is the* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *row that seals the diagonal read before the Real-facing boundary is public.*
6. S *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read only after* W, Q, D, Δ, E *remain visible.*
7. H *records componentwise* hsame *transport for the window, readback, dyadic ledger, diagonal route, seal, replay, provenance, and local naming.*
8. C *records displayed* Cont *replay from the observation request through the same finite rows.*
9. P *is the* Pkg *provenance over* $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite diagonal selector, quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 8968.2 (Finite observation diagonal router NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ *carrier*

$$R = (W, Q, D, \Delta, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}}$ *surface consists exactly of StreamName window admission through* W*, RegSeqRat readback through* Q*, dyadic tolerance exposure through* D*, finite diagonal routing through* Δ*, regular Cauchy diagonal sealing through* E*, terminal Real sealing through* S*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, local naming through* N*, and nonescape from infinite diagonal selection, quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8968.1. The analytic rows are exactly the finite observation window, regular rational readback, dyadic tolerance ledger, diagonal route, diagonal seal, and Real-facing seal.

The diagonal route is downstream of the displayed window. Row Δ can be inspected only after W, Q, D have been exposed, and row E seals that finite route through the regular Cauchy diagonal seal rather than through an infinite selector.

Finally read S, H, C, P, N. The terminal seal, transport, replay, provenance, and local naming preserve the same finite path. Because no carrier coordinate stores quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8968.3 (Finite observation diagonal router Real handoff)**.** *Every Real-facing read from an accepted* $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ *carrier must expose the* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, finite diagonal route* Δ*, and* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *row* E *before it may consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S*. The handoff is therefore a finite observation route through StreamName, RegSeqRat, DyadicRatCore, RegularCauchyDiagonalSeal, and Real rows, not a proof of Real completeness.*

Proof. Start with the observation window. The regular rational readback Q and dyadic ledger D are attached to the same finite request W, so no Real-facing row is public before those three rows are visible.

Use the diagonal route Δ and seal E. They route the displayed finite window through the regular Cauchy diagonal seal, keeping the diagonal read as packet data rather than as an ambient limit choice.

The terminal seal S is then consumed with H, C, P, N still attached. Transport, replay, provenance, and local naming preserve the finite route and add no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ read displays a $\mathsf{StreamName}^{\uparrow}$ finite window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite diagonal-route row, a $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-diagonal-seal: $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite observation window, diagonal-routing row, diagonal seal, terminal Real seal, and structural NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}$ packet is generated from a StreamName finite observation window, RegSeqRat readback, DyadicRatCore tolerance ledger, finite diagonal-routing row, RegularCauchyDiagonalSeal row, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8968.1, Definition 8968.2, and Definition 8968.3 seed the finite diagonal observation route over StreamName, RegSeqRat, DyadicRatCore, RegularCauchyDiagonalSeal, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationDiagonalRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, infinite diagonal selection, quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite observation routing, diagonal seal exactness, Real handoff non-escape, transport stability, replay stability, provenance non-escape, and local naming for BEDC.Derived.FiniteObservationDiagonalRouterUp.

Chapter 8969

A Concrete Naming Certificate for $\mathsf{KunnethFormula}^{\uparrow}$

Following Eilenberg and Mac Lane, the $\mathsf{KunnethFormula}^{\uparrow}$ packet records the finite BEDC surface at which homology of a tensor product is read through displayed source complexes, tensor rows, homology projections, and the retained Tor obstruction. It sits beside the homology certificate of chapter 98, the cohomology certificate of chapter 99, the tensor product certificate of chapter 87, and the abelian-group source of chapter 42. The packet names a finite algebraic-topology route; it does not import a host exact-sequence theorem, selected splitting, quotient homology equality, universal coefficient theorem, spectral sequence, or ambient category of chain complexes.

Definition 8969.1 (KunnethFormula carrier)**.** *A* $\mathsf{KunnethFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, D, T, H, Q, O, S, R, P, N)$$

with the following displayed rows:

1. *C is the first chain-complex source row, exposing only the inspected differential, cycle, boundary, and degree-window data through* $\mathsf{Homology}^{\uparrow}$*.*
2. *D is the second chain-complex source row, with the same visible homology-facing data for the paired source.*
3. *T is the* $\mathsf{TensorProduct}^{\uparrow}$ *row, displaying the tensor product of the two source windows and the induced differential ledger for the inspected degree.*
4. *H is the homology projection row, reading the visible homology outputs of C, D, and T through* $\mathsf{Homology}^{\uparrow}$ *rather than through host quotient groups.*
5. *Q is the cohomology-adjacent comparison row, available only as a sibling reference to* $\mathsf{Cohomology}^{\uparrow}$ *when a consumer asks for dual bookkeeping of the same finite degree window.*
6. *O is the retained* Tor*-boundary obstruction row over the displayed* $\mathsf{AbGroup}^{\uparrow}$ *source data.*
7. *S records componentwise* hsame *transport for the source, tensor, homology, cohomology-adjacent, obstruction, replay, provenance, and naming rows.*
8. *R records displayed* Cont *replay from source admission through tensor formation, homology projection, obstruction retention, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{KunnethFormula}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* $\mathsf{Cohomology}^{\uparrow}$*,* $\mathsf{TensorProduct}^{\uparrow}$*,* $\mathsf{AbGroup}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{KunnethFormula}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host exact sequence, selected splitting, quotient homology equality, a spectral sequence, countable choice, or an ambient abelian category.

Theorem 8969.2 (KunnethFormula NameCert obligations)**.** *For every accepted* $\mathsf{KunnethFormula}^{\uparrow}$ *carrier*

$$K = (C, D, T, H, Q, O, S, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KunnethFormula}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, source-chain exposure through C, D, tensor compatibility through T, homology projection through H, cohomology-adjacent bookkeeping through Q, retained* Tor *obstruction through O, classifier stability through componentwise* hsame *transport S, displayed* Cont *replay through R,* Pkg *provenance through P, local naming through N, and nonescape from host exact sequences, selected splittings, quotient homology equality, spectral sequences, countable choice, or ambient category structure.*

Proof. Project the carrier of Definition 8969.1. The rows $C, D, T, H, Q, O, S, R, P, N$ display source admission, tensor formation, homology projection, cohomology-adjacent bookkeeping, obstruction retention, transport, replay, provenance, and local naming as separate finite coordinates.

First inspect the source rows. The rows C and D expose only the chain-complex windows read through $\mathsf{Homology}^{\uparrow}$, while T records the $\mathsf{TensorProduct}^{\uparrow}$ route that combines those windows before any product homology row is inspected.

Next read the homology-facing rows H, Q, O. The homology projection is tied to the displayed tensor row; the cohomology-adjacent row is only sibling bookkeeping; and the Tor row remains an obstruction rather than a selected splitting or an exact-sequence discharge.

Finally pass through S, R, P, N. The componentwise transport, replay, provenance, and local name preserve the same finite packet. Since no row contains a host exact sequence, selected splitting, quotient homology equality, spectral sequence, countable-choice schedule, or ambient abelian category, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{KunnethFormula}^{\uparrow}$ read displays source homology rows, a tensor-product row, a homology projection row, a retained Tor obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Kunneth-facing consumer may inspect the packet.

Independence witness: homology, cohomology, tensorproduct, abgroup: $\mathsf{KunnethFormula}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source chain windows, tensor compatibility, homology projection, cohomology-adjacent bookkeeping, a retained Tor obstruction, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KunnethFormula}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KunnethFormula}^{\uparrow}}$, seed)

Scope closed: Definition 8969.1 and Definition 8969.2 seed the source-chain, tensor-product, homology-projection, cohomology-adjacent, Tor-obstruction, transport, replay, provenance, and local naming route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KunnethFormula}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host exact sequence, selected splitting, quotient homology equality, universal coefficient theorem, spectral sequence, countable choice, ambient abelian category, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-chain exposure, tensor compatibility, homology projection, Tor-obstruction retention, classifier stability, replay, provenance, and local naming for BEDC.Derived.KunnethFormulaUp.

Constructive story: A KunnethFormula$^{\uparrow}$ packet is generated from two displayed homology source windows, a tensor-product row, a homology projection row, a cohomology-adjacent bookkeeping row, a retained Tor-obstruction row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 8970

A Concrete Naming Certificate for $\mathsf{PoincareDuality}^{\uparrow}$

Following Hatcher and Spanier, the $\mathsf{PoincareDuality}^{\uparrow}$ packet records the finite BEDC route by which an oriented compact manifold fundamental-class row pairs cohomology and homology through the cap-product surface. It sits beside the manifold source of chapter 96, the homology carrier of chapter 98, the cohomology carrier of chapter 99, and the chain-cochain action route of chapter 7629. The packet names only displayed orientation, degree-window, cap-product, homology, cohomology, transport, replay, provenance, and local naming rows; it does not import host triangulation choice, quotient chain equality, excision, universal coefficients, or an ambient duality theorem.

Definition 8970.1 (Poincare duality carrier)**.** *A* $\mathsf{PoincareDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, O, H, G, C, Q, T, R, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *source row, exposing the compact oriented manifold data only through the registered manifold NameCert surface.*
2. *O is the orientation and fundamental-class row, recording the displayed top-dimensional homology class used by the packet.*
3. *H is the* $\mathsf{Homology}^{\uparrow}$ *row that carries the target homology degree and the fundamental-class endpoint.*
4. *G is the* $\mathsf{Cohomology}^{\uparrow}$ *row that carries the source cohomology degree inspected by the duality read.*
5. *C is the* $\mathsf{CapProduct}^{\uparrow}$ *action row, tying the cohomology read and the fundamental-class row to the resulting homology read.*
6. *Q is the degree-window row, naming the manifold dimension, cohomology degree, and complementary homology degree used by the displayed action.*
7. *T records componentwise* hsame *transport for manifold, orientation, homology, cohomology, cap-product, degree-window, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a Poincare-duality request through $M, O, H, G, C, Q, T, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{PoincareDuality}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{CapProduct}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{PoincareDuality}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, O, H, G, C, Q, T, R, P, N$. It has no coordinate for a selected triangulation, quotient chain equality, a host manifold theorem, universal coefficient data, excision, or bridge checking.

Theorem 8970.2 (Poincare duality NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDuality}^{\uparrow}$ *carrier*

$$D = (M, O, H, G, C, Q, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareDuality}^{\uparrow}}$ *surface consists exactly of manifold-source admission through M, orientation and fundamental-class exposure through O, homology exposure through H, cohomology exposure through G, cap-product action exposure through C, degree-window exposure through Q, componentwise* hsame *transport through T, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8970.1. The displayed packet contains the manifold source, orientation row, homology row, cohomology row, cap-product row, degree window, and structural rows.

Read the algebraic-topology route in order. The manifold row M supplies only the registered finite manifold surface, while O displays the orientation and fundamental class. The rows G and H expose the cohomology and homology sides to be paired.

The cap-product row C is the only action coordinate. It consumes the cohomology and fundamental-class rows at the degree window Q before any homology-facing output is read.

Finally T, R, P, N transport, replay, source, and name the same finite route. Since the carrier has no selected triangulation, quotient chain equality, universal coefficient data, excision, host theorem, or bridge row, the listed rows exhaust the NameCert surface. □

Theorem 8970.3 (Poincare duality scope boundary)**.** *Every duality read exported by an accepted* $\mathsf{PoincareDuality}^{\uparrow}$ *packet factors through*

$$M, O, H, G, C, Q, T, R, P, N.$$

A consumer may read the displayed manifold source, orientation and fundamental-class row, cohomology row, cap-product action, complementary homology row, degree window, transport, replay, provenance, and local name. The consumer may not read a selected triangulation, chain-level quotient equality, universal coefficient theorem, excision theorem, ambient manifold theorem, or bridge-checked standard schema from the packet.

Proof. Start with the obligation theorem. It identifies the complete visible row inventory for the local certificate and gives no coordinate outside $M, O, H, G, C, Q, T, R, P, N$.

Use the sibling route. The manifold chapter supplies M, homology supplies H and the fundamental-class endpoint named by O, cohomology supplies G, and cap product supplies the displayed action row C at the degree window Q.

The structural rows preserve this route under hsame transport and Cont replay. Because the excluded host principles are absent from the carrier and from the local classifier, any consumer claiming them has left the $\mathsf{PoincareDuality}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted $\mathsf{PoincareDuality}^{\uparrow}$ read exposes ManifoldUp, HomologyUp, CohomologyUp, CapProductUp, orientation, degree-window, transport, replay, provenance, and local NameCert rows before a duality-facing consumer reads a complementary homology output.

Independence witness: manifold, homology, cohomology, cap-product: none is bijective to $\mathsf{PoincareDuality}^{\uparrow}$ because this packet alone binds an oriented manifold fundamental-class row to the cohomology-to-homology cap-product route in one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDuality}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareDuality}^{\uparrow}$ packet is generated from a ManifoldUp source row, orientation and fundamental-class rows, HomologyUp and CohomologyUp rows, a CapProductUp action row, a complementary degree-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8970.1, Definition 8970.2, and Definition 8970.3 seed the finite oriented-manifold cap-product route from displayed cohomology rows to complementary homology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDuality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected triangulations, quotient chain equality, universal coefficient data, excision, ambient manifold duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, orientation and fundamental-class exposure, homology and cohomology row exposure, cap-product action exactness, degree-window exactness, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDualityUp.

Chapter 8971

A Concrete Naming Certificate for $\mathsf{IntervalBisectionChain}^{\uparrow}$

Following Bishop and Bridges constructive interval bisection, the $\mathsf{IntervalBisectionChain}^{\uparrow}$ packet records a finite nested row of dyadic intervals before any infinite branch, Dedekind cut, selected limit, or ambient completeness principle is available. It sits after the single-step routes of chapter 7168 and chapter 7273, reads dyadic endpoint and width arithmetic through chapter 291, exports finite windows to chapter 126, hands regular rational readback to chapter 301, and reaches a terminal real-facing seal only through chapter 38. The finite chain is the carrier-level source for the nested route of chapter 7290; it does not supply interval choice, Heine–Borel compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Falsifiable prediction: A consumer that reads an interval bisection chain without endpoint rows, containment rows, width-halving rows, finite StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, structural transport, continuation replay, provenance, and local NameCert rows has left the $\mathsf{IntervalBisectionChain}^{\uparrow}$ packet.

Independence witness: bishop interval bisection, dyadic interval bisection, nested interval principle, stream name, regseqrat, dyadicratcore, real: $\mathsf{IntervalBisectionChain}^{\uparrow}$ is not bijective to any one sibling because it jointly carries the whole finite nested row, per-step dyadic arithmetic, window generation, regular readback, terminal sealing, transport, replay, provenance, and local naming.

Definition 8971.1 (Interval bisection chain carrier)**.** *An* $\mathsf{IntervalBisectionChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (I_0, \ldots, I_n, E, K, W, S, R, Q, H, T, P, N)$$

where each $I_i = [a_i, b_i]$ *is a displayed dyadic interval row and the packet exposes the following data:*

1. *E is the endpoint ledger, listing the dyadic endpoints a_i, b_i and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *admission rows.*
2. *K is the containment ledger, recording $I_{i+1} \subseteq I_i$ for every displayed successor step.*
3. *W is the width-halving ledger, recording $b_{i+1} - a_{i+1} \leq (b_i - a_i)/2$ through dyadic arithmetic rows.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule generated from the endpoint rows and successor depth n.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the midpoint or endpoint selector read from the finite windows.*

6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after E, K, W, S, R have all been displayed.*
7. *H records componentwise* hsame *transport for the interval rows, endpoint ledger, containment ledger, width ledger, window schedule, regular readback, real seal, replay, provenance, and local name.*
8. *T records displayed* Cont *replay from an interval-chain request through $I_0, \ldots, I_n, E, K, W, S, R, Q, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{IntervalBisectionChain}^{\uparrow}$, $\mathsf{BishopIntervalBisection}^{\uparrow}$, $\mathsf{DyadicIntervalBisection}^{\uparrow}$, $\mathsf{NestedIntervalPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalBisectionChain}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a chosen infinite branch, Dedekind cut, interval choice principle, Heine–Borel cover argument, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8971.2 (Interval bisection chain NameCert obligations)**.** *For every accepted* $\mathsf{IntervalBisectionChain}^{\uparrow}$ *carrier*

$$C = (I_0, \ldots, I_n, E, K, W, S, R, Q, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalBisectionChain}^{\uparrow}}$ *surface consists exactly of finite interval-row admission for $I_0, \ldots, I_n$, endpoint admission through E, successor containment through K, width-halving exactness through W, finite window generation through S, regular rational readback through R, terminal real sealing through Q, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8971.1. The analytic rows are the displayed intervals, endpoint ledger, containment ledger, width-halving ledger, finite window schedule, regular rational readback, and terminal real seal.

The successor rows are read in order. Endpoint admission supplies the dyadic arithmetic inputs, containment keeps each retained child inside its predecessor, and the width ledger records the displayed halving budget for the same successor step.

The value-facing rows are downstream. The StreamName schedule is generated from the finite endpoint rows, the RegSeqRat readback reads the finite windows, and the Real seal is available only after those rows have been displayed.

Finally H, T, P, N transport, replay, source, and name the same packet. Since no coordinate carries a chosen infinite branch, Dedekind cut, interval choice, Heine–Borel cover argument, quotient real equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8971.3 (Interval bisection chain regular Cauchy window)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntervalBisectionChain}^{\uparrow}$ *packet must first read the finite route*

$$I_0, \ldots, I_n, E, K, W, S, R, Q, H, T, P, N.$$

The route supplies a regular Cauchy window from nested dyadic intervals by finite endpoint arithmetic, successor containment, and width-halving rows. It exports no selected infinite branch, no Dedekind cut, no interval-choice rule, no Heine–Borel subcover, no quotient real equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the obligation surface of Definition 8971.2. Any accepted consumer must expose the interval rows, endpoint ledger, containment ledger, and width-halving ledger before it can read a window or a value-facing seal.

The regular Cauchy window is finite. The displayed depth n bounds the row inspection, the StreamName schedule S records exactly the inspected windows, and the RegSeqRat row R reads only those finite windows together with the dyadic tolerance budget supplied by E and W.

The terminal Real read is therefore downstream of the finite chain. The seal Q is not a selected limit or quotient real equality; it is only the real-facing boundary reached after the chain, window, and readback rows have been displayed.

The structural rows H, T, P, N keep the same packet attached to the read. Since the excluded principles are absent from the carrier and from the local classifier, no consumer can obtain them from the interval bisection chain route. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalBisectionChain}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalBisectionChain}^{\uparrow}$ packet is generated from finite dyadic interval rows, endpoint ledgers, successor containment rows, width-halving rows, finite StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalBisectionChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8971.1, Definition 8971.2, and Definition 8971.3 seed the finite nested dyadic bisection-chain route from Bishop and dyadic bisection rows to regular Cauchy windows and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalBisectionChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, Dedekind cuts, interval choice, Heine–Borel compactness, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, successor containment, width-halving exactness, window generation, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.IntervalBisectionChainUp.

Chapter 8972

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ packet records the explicit dyadic modulus budget carried by a regular Cauchy rational name before the L10 Real and Completion route reads it as a value-facing object. It sits over the finite scheduling surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the sequence-space route of chapter 5965, and the terminal real-name seal of chapter 38. The packet names only the displayed modulus rows; it does not add quotient stream equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8972.1 (Bishop regular Cauchy modulus carrier)**.** *A* $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, D, M, S, H, C, P, L)$$

with the following displayed rows:

1. *R is the regular Cauchy row, read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface.*
2. *D is the dyadic budget row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and attached to each requested precision scale.*
3. *M is the modulus row, assigning to every displayed dyadic request the finite observation window that witnesses the regular Cauchy bound.*
4. *S is the shift row, recording the* $\mathsf{StreamName}^{\uparrow}$ *reindexing by which a tail or translated name is read with the same displayed budget.*
5. *H records componentwise* hsame *transport for the regular Cauchy row, dyadic budget row, modulus row, shift row, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay through regular readback, dyadic request, modulus lookup, StreamName reindexing, transport, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyModulus}^{\uparrow}}$ *ledger row.*

The carrier contains no coordinate for a host choice function, quotient real equality, selected representative stream, hidden infinite tail choice, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8972.2 (Bishop regular Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ *carrier*

$$B = (R, D, M, S, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet;*
2. *regular Cauchy readback through R and dyadic budget admission through D;*
3. *modulus lookup through M and StreamName reindexing through S;*
4. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
5. Pkg *provenance and local NameCert ledgering through P and L.*

No obligation row may replace this surface by quotient stream equality, selected representatives, countable choice, trichotomy, hidden tail choice, host completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8972.1. The rows R, D, M, S are the object-facing data, and the rows H, C, P, L are the structural data.

The regular Cauchy and dyadic rows form the budget boundary. Row R supplies the regular rational readback, while row D names the dyadic request scale against which the Cauchy estimate is checked.

The modulus and shift rows keep the boundary finite. Row M records the requested observation window, and row S records how the same request is read after a StreamName reindexing. These rows do not store a selected infinite tail.

Finally H, C, P, L transport, replay, source, and name the same displayed packet. Since no carrier coordinate names quotient equality, selected representatives, choice, trichotomy, hidden tail choice, host completion, or ambient completeness, the five rows exhaust the local NameCert surface. □

Theorem 8972.3 (Bishop regular Cauchy modulus shift stability)**.** *Let*

$$B = (R, D, M, S, H, C, P, L)$$

be an accepted $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ *carrier. If a consumer shifts the regular Cauchy name by the displayed* $\mathsf{StreamName}^{\uparrow}$ *reindexing row S, then the dyadic budget displayed by D and the modulus lookup displayed by M remain the budget read by the shifted name. Equivalently, the shifted read factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the same dyadic budget row D, and the same modulus row M, with only the StreamName indices readdressed by S.*

Proof. Start with the regular Cauchy row R. Its consumer-facing observations are read only through the finite StreamName windows named in the packet, not through a host sequence or a quotient representative.

Apply the shift row S. The row reindexes the displayed StreamName addresses, so the same observation request is read at the shifted addresses while preserving the packet's dyadic request row D.

Use the modulus row M after reindexing. Because M is attached to the dyadic request row and not to an external choice of representative, the shifted consumer obtains the same displayed budget for the shifted regular name.

The transport and replay rows H and C then carry the reindexed read through provenance P and local ledger L. Thus shifting changes only the visible StreamName addresses; it does not introduce a new modulus, a hidden tail selector, quotient equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ shifted read displays the same dyadic budget and modulus rows after applying its StreamName reindexing row.

Independence witness: regseqrat, streamname, dyadicratcore, regularsequencespace, real: BishopRegularCauchy is not bijective to any listed sibling because it carries the regular Cauchy row, dyadic budget row, explicit modulus row, StreamName shift row, transport, replay, provenance, and NameCert ledger in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyModulus}^{\uparrow}$ packet is generated from regular Cauchy readback, dyadic budget, explicit modulus, StreamName shift, hsame transport, Cont replay, Pkg provenance, and local NameCert ledger rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8972.1, Definition 8972.2, Definition 8972.3, the falsifiable prediction, and the independence witness seed the finite explicit-modulus route for regular Cauchy rational names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected representatives, countable choice, trichotomy, hidden tail choice, host completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular Cauchy row exactness, dyadic budget stability, modulus lookup determinacy, shift stability, transport, replay, provenance, and local ledgering for BEDC.Derived.BishopRegularCauchy

Chapter 8973

A Concrete Naming Certificate for $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$

The $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ packet realizes the mirror-slot faithfulness branch of section 9413.32 as finite NameCert data. It sits over the critical-strip source of chapter 72, the zero-source rows of chapter 73, the critical-line witness boundary of chapter 548, the complex-coordinate carrier of chapter 39, and the real-coordinate seal of chapter 38. The packet records a classifier that reads only displayed zero signatures and mirror slots; it does not assert RH, analytic continuation completeness, zero simplicity, Euler product closure, or an unledgered zero observable.

Definition 8973.1 (Zero signature faithfulness carrier)**.** *A* $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (Z, J, \Sigma, F, H, C, P, L)$$

with the following displayed rows:

1. *Z is the accepted zero-source row, read through* $\mathsf{ZetaZeros}^{\uparrow}$ *inside the* $\mathsf{CritStrip}^{\uparrow}$ *boundary.*
2. *J is the mirror-slot row, recording the functional-equation companion slot visible to the zero source.*
3. *Σ is the displayed zero signature row, consisting of zero atom type, spectral time, and mirror slot as in Definition 9413.420.*
4. *F is the packet classifier row: two accepted zero-source rows with equal displayed Σ rows are assigned to the same mirror slot.*
5. *H records componentwise* hsame *transport for zero source, mirror slot, signature, classifier, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay through zero-source admission, mirror-slot readback, signature readback, classifier comparison, transport, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CriticalLineWitness}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{ZeroSignatureFaithfulness}^{\uparrow}}$ *ledger row.*

The carrier contains no coordinate for an unledgered fourth zero observable, an RH proof object, a hidden analytic continuation theorem, zero simplicity, Euler product closure, or host-level equality of complex zeros.

Theorem 8973.2 (Zero signature faithfulness NameCert obligations)**.** *For every accepted* ZeroSignatureFaithfulness *carrier*

$$Q = (Z, J, \Sigma, F, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{ZeroSignatureFaithfulness}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *zero-source admission through the* $\mathsf{CritStrip}^{\uparrow}$ *and* $\mathsf{ZetaZeros}^{\uparrow}$ *sibling rows;*
2. *mirror-slot readback through* J *and displayed signature readback through* Σ*;*
3. *packet classifier faithfulness through* F*;*
4. *componentwise* hsame *transport and displayed* Cont *replay through* H *and* C*;*
5. Pkg *provenance and local NameCert ledgering through* P *and* L*.*

These rows expose the mirror-slot faithfulness packet only as finite NameCert data. They do not supply RH, analytic continuation completeness, zero simplicity, Euler product closure, hidden zero observables, or host-level equality of complex zeros.

Proof. Project the carrier of Definition 8973.1. The rows Z, J, Σ, F are the zero-source, mirror-slot, signature, and classifier data; the rows H, C, P, L are the structural data.

The zero-source row is bounded by the concrete sibling chapters. It is read through the critical-strip and zeta-zero surfaces before the mirror slot or signature can be inspected.

The classifier row is the faithfulness surface. It compares displayed zero signatures and returns equality of mirror slots inside the packet classifier, without adding a fourth zero observable.

Finally H, C, P, L transport, replay, source, and name the same finite packet. Since no coordinate stores RH, analytic continuation completeness, zero simplicity, Euler product closure, hidden observables, or host-level equality of complex zeros, the listed rows exhaust the local NameCert surface. □

Theorem 8973.3 (Zero signature faithfulness mirror-slot rigidity)**.** *Let two accepted zero-source rows be read by a* $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ *carrier. If their displayed zero signatures agree, then the packet classifier row* F *assigns them to the same mirror slot. Thus equal displayed zero signatures force the same mirror slot under the finite classifier surface of the packet.*

Proof. Read the two zero-source rows through Z. Each row first passes through the critical-strip and zeta-zero sibling surfaces, so the comparison is internal to the displayed zero-source boundary.

Compare the signature rows through Σ. Equality of the displayed signatures means equality of the zero atom type, spectral time, and mirror-slot component recorded by the carrier.

Apply the classifier row F. By the definition of the packet classifier, equal displayed Σ rows are classified into the same mirror slot.

The transport and replay rows H and C preserve this comparison under accepted readback, while P and L keep the result attached to package provenance and local naming. The conclusion is finite mirror-slot rigidity, not an RH theorem or a host equality principle for complex zeros. □

Falsifiable prediction: If two accepted zero-source rows share the displayed signature but occupy distinct mirror slots, then $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ fails the mirror-slot rigidity theorem.

Independence witness: criticalstripzetazerowitness, criticallinewitness, zetazeros, nontrivialzeroclassifier: $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ is not bijective to any listed sibling because it carries the zero-source row, mirror-slot row, displayed zero signature, packet classifier, transport, replay, provenance, and local NameCert ledger in one packet.

The Lean follow-up must provide `Nontrivial ZeroSignatureFaithfulnessUp` and `FieldFaithful ZeroSignatureFaithfulnessUp` instances for the displayed carrier. The existing RH-route context at `BEDC.Derived.RHRoute.ZeroSignatureFaithfulness` supplies the zero-signature carrier-map boundary but does not yet define the concrete chapter carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroSignatureFaithfulness}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ packet is generated from zero-source, mirror-slot, displayed zero-signature, classifier, hsame transport, Cont replay, Pkg provenance, and local NameCert ledger rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeroSignatureFaithfulness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8973.1, Definition 8973.2, Definition 8973.3, the falsifiable prediction, and the independence witness seed the finite mirror-slot faithfulness route for zero-source signatures.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeroSignatureFaithfulness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, zero simplicity, analytic continuation completeness, Euler product closure, host-level equality of complex zeros, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-source row exactness, mirror-slot readback, signature comparison, classifier faithfulness, transport, replay, provenance, local ledgering, and TasteGate instances for BEDC.Derived.ZeroSignatureFaithfulnessUp.

Chapter 8974

A Concrete Naming Certificate for $\mathsf{ChoquetBoundary}^{\uparrow}$

Following Bishop-de Leeuw and Choquet, the $\mathsf{ChoquetBoundary}^{\uparrow}$ packet records the finite boundary surface by which a compact convex function-system read exposes peak-boundary and representing-measure evidence before any boundary representation is consumed. It sits over the compact carrier of chapter 60, the convex-set rows of chapter 208, the continuous-map surface of chapter 130, the scalar seal of chapter 38, and the order-vector context of chapter 3620 and chapter 3386. The packet names a boundary certificate for displayed function-system data; it is not a host Shilov boundary, a selected representing measure, a quotient state space, or an appeal to ambient compactness beyond the sibling rows.

Definition 8974.1 (ChoquetBoundary carrier)**.** *A* $\mathsf{ChoquetBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (K, A, P, E, R, H, T, G, N)$$

with the following displayed rows:

1. *K is the compact source row for the inspected state space, read through the* $\mathsf{Compact}^{\uparrow}$ *certificate.*
2. *A is the convex function-system row over K, read through* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and the* $\mathsf{Real}^{\uparrow}$ *scalar seal.*
3. *P is the peak-boundary observation row: it records the finite peak tests and exposed boundary witnesses inspected by the packet.*
4. *E is the extremal handoff row that keeps exposed-point or extremal-state evidence visible before a boundary representation row is consumed.*
5. *R is the representing-measure handoff row; it records only the displayed measure or barycentric route presented to the packet.*
6. *H records componentwise* hsame *transport for compact source rows, function-system rows, peak observations, extremal or representing-measure handoffs, replay rows, provenance, and the local naming row.*
7. *T records the displayed* Cont *replay from compact source data to function-system data, then to peak observations, then to extremal or representing-measure handoff rows.*
8. *G is the* Pkg *provenance over* $\mathsf{ChoquetBoundary}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RieszSpace}^{\uparrow}$*,* $\mathsf{LocallyConvexSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ChoquetBoundary}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host-selected boundary, Shilov minimality, quotient state equality, countable choice, a selected representing measure, or an ambient compactness principle detached from the displayed sibling certificates.

Theorem 8974.2 (ChoquetBoundary NameCert obligations)**.** *For the carrier of Definition 8974.1, the local* $\mathsf{NameCert}_{\mathsf{ChoquetBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* $C = (K, A, P, E, R, H, T, G, N)$*, compact source exposure through* K*, convex function-system exposure through* A*, peak-boundary observation exposure through* P*, extremal handoff exposure through* E*, representing-measure handoff exposure through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* T*,* Pkg *provenance through* G*, and local naming through* N*.*

Proof. Carrier admission is tuple projection from the displayed packet C. The analytic rows expose the compact source, convex function-system data, peak observations, extremal evidence, and representing-measure handoff.

The structural rows are exactly H, T, G, N. They transport the same displayed rows, replay the same finite dependency route, record provenance for the named packets, and bind the local certificate.

No additional boundary object is available. Since the excluded principles are not carrier coordinates, the local NameCert cannot export them. □

Theorem 8974.3 (ChoquetBoundary peak route)**.** *Every accepted* $\mathsf{ChoquetBoundary}^{\uparrow}$ *carrier routes a boundary-facing read through the compact source row* K*, the convex function-system row* A*, and the peak-boundary observation row* P *before any extremal handoff* E *or representing-measure handoff* R *is consumed. The route preserves the* $\mathsf{Real}^{\uparrow}$ *scalar seal and the* hsame *transport rows, and it exports no host-selected boundary, Shilov minimality claim, quotient state equality, countable-choice schedule, selected representing measure, or detached compactness principle.*

Proof. Project the carrier of Definition 8974.1. Its only source rows for boundary analysis are the compact row K, the function-system row A, and the peak-observation row P.

Apply the NameCert obligations of Definition 8974.2. They place the extremal and representing-measure handoffs after those rows and keep the replay, transport, provenance, and local name attached to the same packet.

It remains to check non-escape. The carrier has no coordinate for the excluded host boundary, minimality, quotient, choice, selected-measure, or detached compactness principles, so a consumer can only read the displayed peak route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoquetBoundary}^{\uparrow}$ packet is generated from compact source rows, convex function-system rows, peak-boundary observations, extremal or representing-measure handoff rows, Real scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoquetBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8974.1, Definition 8974.2, and Definition 8974.3 seed the finite carrier and peak-route obligation surface for later representation packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoquetBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Shilov minimality, a selected representing measure, quotient state equality, countable choice, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a public route theorem that packages the peak-boundary and representing-measure handoff rows into a downstream comparison boundary.

Chapter 8975

A Concrete Naming Certificate for $\mathsf{IncidenceAlgebra}^{\uparrow}$

Following Rota and Stanley on incidence algebras of finite posets, the $\mathsf{IncidenceAlgebra}^{\uparrow}$ packet records the convolution carrier that makes zeta and Mobius rows inspectable before any inversion consumer reads them. It sits after the order source of chapter 53, the scalar operation rows of chapter 43, the finite tabular carrier of chapter 49, and the inversion consumer route of chapter 8007. The packet names interval-supported functions and their finite convolution ledger; it does not import an infinite algebra, a host function ring, quotient equality of functions, unbounded sums, or selected inverse matrices.

Definition 8975.1 (Incidence algebra carrier)**.** *An* $\mathsf{IncidenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, R, F, S, M, Z, U, H, C, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poset}^{\uparrow}$ *source row, including the displayed finite comparable-pair ledger.*
2. *R is the* $\mathsf{Ring}^{\uparrow}$ *scalar row used for addition, multiplication, zero, and one on interval coefficients.*
3. *F is the interval-supported function table, assigning a scalar coefficient only to displayed pairs $x \leq y$.*
4. *S is the finite support ledger, refusing every non-comparable pair and every pair outside the displayed finite source.*
5. *M is the* $\mathsf{Mat}^{\uparrow}$ *table view of the same interval-supported data, indexed by the finite poset enumeration.*
6. *Z is the zeta row, with unit coefficients on accepted comparable pairs and zero outside the interval support.*
7. *U is the finite convolution row, listing exactly the intermediate z with $x \leq z \leq y$ for every product read.*
8. *H records componentwise* hsame *transport for the poset source, scalar row, function table, support ledger, matrix view, zeta row, convolution row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a convolution request through $P, R, F, S, M, Z, U, H, G, N$.*

10. *G is the* Pkg *provenance over* $\mathsf{IncidenceAlgebra}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{MobiusInversion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{IncidenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, R, F, S, M, Z, U, H, C, G, N$. It has no coordinate for an infinite incidence algebra, host function equality, unbounded summation, selected inverse matrices, or ambient algebraic completion.

Theorem 8975.2 (Incidence algebra NameCert obligations)**.** *For every accepted* $\mathsf{IncidenceAlgebra}^{\uparrow}$ *carrier*

$$I = (P, R, F, S, M, Z, U, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{IncidenceAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite poset source, scalar ring row, interval-supported function table, support ledger, matrix view, zeta row, finite convolution row, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8975.1. The rows P, R, F, S, M, Z, U are the displayed mathematical data, and H, C, G, N are the structural rows that transport, replay, source, and name the same packet.

The support row S and convolution row U keep every product read inside a finite interval of the displayed poset. The matrix row M is only a finite table view of the same interval-supported data, so it cannot add host matrix inversion or quotient function equality.

Since no accepted coordinate stores infinite support, unbounded summation, selected inverses, or ambient completion, the listed rows exhaust the NameCert obligations. □

Theorem 8975.3 (Incidence algebra convolution stability)**.** *Every convolution read of two accepted* $\mathsf{IncidenceAlgebra}^{\uparrow}$ *packets over the same P and R remains an* $\mathsf{IncidenceAlgebra}^{\uparrow}$ *read over the displayed route*

$$P, R, F, S, M, Z, U, H, C, G, N.$$

The product coefficient at $x \leq y$ may use only the finite intermediates recorded by U for that interval.

Proof. Start with the common poset and scalar rows P, R. The support rows of the two packets restrict both coefficient tables to comparable pairs in the same finite source.

For a displayed interval $x \leq y$, read the convolution row U. It lists the only intermediate elements available for the product coefficient, and the scalar multiplication and addition are the operations exposed by R.

The resulting table has the same finite support boundary and the same matrix view over the finite poset enumeration. Transport, replay, provenance, and local naming are inherited through H, C, G, N, so the product remains inside the stated carrier surface. □

Theorem 8975.4 (Incidence algebra Mobius consumer route)**.** *Every* $\mathsf{MobiusInversion}^{\uparrow}$ *consumer that reads zeta or Mobius rows from an* $\mathsf{IncidenceAlgebra}^{\uparrow}$ *packet must first expose the finite poset source, scalar row, interval-supported table, support ledger, matrix view, zeta row, and convolution row of that packet. The consumer receives a finite incidence carrier; it does not receive an ambient function algebra or a selected inverse object.*

Proof. Use the provenance row G to identify the Mobius inversion consumer and the incidence carrier it reads. The consumer starts from the same finite $\mathsf{Poset}^{\uparrow}$ source and $\mathsf{Ring}^{\uparrow}$ scalar row as the incidence packet.

The zeta row Z is visible only after the interval support and matrix table have been displayed. The Mobius-facing inverse check can then consume the finite convolution row U, matching the finite interval ledger used by chapter 8007.

Because the route is through $P, R, F, S, M, Z, U, H, C, G, N$, no downstream consumer can replace it by host function equality, an infinite incidence algebra, unbounded summation, or a selected matrix inverse. □

Falsifiable prediction: Every accepted $\mathsf{IncidenceAlgebra}^{\uparrow}$ convolution read displays PosetUp source rows, RingUp scalar rows, interval-supported function tables, support ledgers, MatrixUp table views, zeta rows, finite convolution windows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any MobiusInversionUp consumer can read it.

Independence witness: poset, ring, matrix, mobius-inversion: none of the listed siblings jointly carries interval-supported function rows, finite support refusal, matrix table view, zeta rows, finite convolution windows, transport, replay, provenance, and local NameCert data as one incidence-algebra packet.

Closure status for $\mathsf{NameCert}_{\mathsf{IncidenceAlgebra}^{\uparrow}}$.

Constructive story: An $\mathsf{IncidenceAlgebra}^{\uparrow}$ packet is generated from a PosetUp source row, a RingUp scalar row, interval-supported function rows, finite support refusal, a MatrixUp table view, zeta rows, finite convolution windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IncidenceAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8975.1, Definition 8975.2, Definition 8975.3, and Definition 8975.4 seed the finite incidence-algebra route consumed by Mobius inversion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IncidenceAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim infinite incidence algebras, host function rings, quotient equality of functions, unbounded summation, selected inverse matrices, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-support exactness, convolution stability, Mobius consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.IncidenceAlgebraUp.

Chapter 8976

A Concrete Naming Certificate for $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$

The $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ packet realizes the boundary-measure branch of chapter 9463 as a finite NameCert surface. It sits after the visual-boundary rows of chapter 4181, the boundary-shadow window of chapter 3813, the metric comparison surface of chapter 3474, and the terminal scalar seal of chapter 38. The packet names visual weights only as displayed finite shadow-window data; it does not construct a host sigma-algebra, a completed measure space, a quotient boundary, a global Patterson–Sullivan density, or ambient analytic regularity.

Definition 8976.1 (Hyperbolic visual measure carrier)**.** *A* $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, S, M, R, W, T, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *row, exposing the boundary face visible to the observer.*
2. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, listing the finite shadow windows on which visual weights are read.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row attached to the same displayed disk or metric source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-readback row for finite nonnegative weights and total-window comparisons.*
5. *W is the visual-weight table, assigning a displayed scalar only to shadow windows listed by S.*
6. *T is the finite-total ledger, recording the finite sum of weights over the current visual window and refusing unlisted boundary regions.*
7. *F is the TasteGate-facing field-faithfulness row, requiring the same packet to support Nontrivial and FieldFaithful readings before a scalar consumer treats R as a faithful weight surface.*
8. *H records componentwise* hsame *transport for boundary, shadow, metric, scalar, weight, total, TasteGate-facing, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a visual-measure request through $B, S, M, R, W, T, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, M, R, W, T, F, H, C, P, N$. It has no coordinate for completed measures, host measurable subsets, quotient ideal boundary equality, selected densities, or ambient analytic regularity.

Theorem 8976.2 (Hyperbolic visual measure NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ *carrier*

$$V = (B, S, M, R, W, T, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the visual-boundary row, boundary-shadow window, hyperbolic metric comparison, Real scalar readback, visual-weight table, finite-total ledger, TasteGate-facing field-faithfulness row, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8976.1. The boundary, shadow, metric, scalar, weight, total, and TasteGate-facing rows are B, S, M, R, W, T, F, while H, C, P, N transport, replay, source, and name the same packet.

The shadow row S bounds the visual windows available to the measure read. The scalar row R and finite-total row T then expose only finite weight comparisons on those displayed windows.

Since the packet has no coordinate for completed measures, host measurable subsets, quotient boundary equality, selected densities, or ambient regularity, the listed rows exhaust the NameCert obligations. □

Theorem 8976.3 (Hyperbolic visual measure shadow support)**.** *Every visual-measure read from an accepted* $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ *packet is supported on the finite boundary-shadow windows listed by S. A consumer may sum weights through T only after displaying the visual-boundary row B, the shadow row S, the metric row M, and the scalar readback row R.*

Proof. Begin with the displayed boundary and shadow rows. The visual-boundary row names the observer-visible boundary face, and the boundary-shadow row lists the finite windows that may receive weights.

The metric row supplies the comparison data used to index the shadow windows, while the Real row supplies the scalar readback. The visual-weight table W assigns values only to those listed windows.

The finite-total ledger T can therefore sum only the displayed shadow support. Transport and replay preserve this route, so a consumer cannot turn the packet into a completed measure on an unlisted boundary region. □

Theorem 8976.4 (Hyperbolic visual measure TasteGate route)**.** *The TasteGate-facing obligations for* $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ *are bound to the same finite packet*

$$B, S, M, R, W, T, F, H, C, P, N.$$

In particular, `lean4/BEDC/Derived/HyperbolicVisualMeasureUp/TasteGate.lean` *must provide Nontrivial and FieldFaithful instances for this carrier before a downstream scalar consumer treats the visual-weight table as a faithful measurement row.*

Proof. Read the field-faithfulness row F from the carrier. It is displayed beside the boundary, shadow, metric, scalar, weight, and total rows, so it cannot be supplied by a different hyperbolic sibling.

The provenance row P identifies the consumed visual-boundary, boundary-shadow, hyperbolic-metric, and Real rows. The replay row C preserves their order before a scalar consumer can inspect W or T.

Thus the Nontrivial and FieldFaithful requirements are obligations on the local $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ packet. They do not assert a global measure theory, a completed boundary, or an ambient analytic density. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ read displays HyperbolicVisualBoundaryUp, HyperbolicBoundaryShadowUp, HyperbolicMetricUp, RealUp scalar readback, finite visual weights, finite-total ledgers, TasteGate-facing field-faithfulness rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data before any visual measure consumer may read the packet.

Independence witness: hyperbolic-visual-boundary, hyperbolic-boundary-shadow, hyperbolic-metric, hyperbolic-observer-expansion, hyperbolic-visual-metric: none of these siblings jointly carries finite visual-weight rows, finite-total ledgers, TasteGate-facing field-faithfulness rows, transport, replay, provenance, and local NameCert data as one visual-measure packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualMeasure}^{\uparrow}$ packet is generated from HyperbolicVisualBoundaryUp boundary rows, HyperbolicBoundaryShadowUp finite shadow windows, HyperbolicMetricUp comparison rows, RealUp scalar readbacks, finite visual-weight rows, finite-total ledgers, TasteGate-facing Nontrivial and FieldFaithful obligations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8976.1, Definition 8976.2, Definition 8976.3, and Definition 8976.4 seed the finite visual-measure route over boundary, shadow, metric, scalar, field-faithfulness, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sigma-algebras, completed measure spaces, quotient boundary equality, selected densities, global Patterson–Sullivan theory, ambient analytic regularity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shadow support exactness, finite-total exactness, TasteGate Nontrivial and FieldFaithful rows, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicVisualMeasureUp.

Chapter 8977

A Concrete Naming Certificate for $\mathsf{RealClosedField}^{\uparrow}$

Following Artin and van der Waerden on real closed fields, the $\mathsf{RealClosedField}^{\uparrow}$ packet records the finite ordered-algebra surface by which a displayed field is read as real-closed without importing a host algebraic closure, selected roots, or classical trichotomy. It sits over the real-name boundary of chapter 38, the field carrier of chapter 45, the polynomial carrier of chapter 50, the root-isolation surface of chapter 2269, and the Sturm-chain route of chapter 2079. The packet names only the displayed order, polynomial, odd-degree root, positive-square, isolation, transport, replay, provenance, and local certificate rows.

Definition 8977.1 (Real closed field carrier)**.** *A* $\mathsf{RealClosedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, K, O, P, S, I, A, Q, H, C, G, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row naming the real-facing carrier boundary.*
2. *K is the* $\mathsf{Field}^{\uparrow}$ *row naming addition, multiplication, units, inverse-on-witnesses, and field classifier transport.*
3. *O is the order row attached to the same displayed carrier as K, with positivity and comparison data read only through finite BEDC rows.*
4. *P is the* $\mathsf{Poly}^{\uparrow}$ *row for coefficient lists, degree ledgers, formal derivatives, and evaluation requests over the displayed field.*
5. *S is the* $\mathsf{SturmSequence}^{\uparrow}$ *row carrying finite sign-variation and endpoint ledger data for the polynomial request.*
6. *I is the* $\mathsf{RealRootIsolation}^{\uparrow}$ *row exporting retained isolating windows and no selected host root.*
7. *A is the odd-degree root route: for each displayed odd-degree polynomial row admitted by P, it records the Sturm/isolation handoff through S and I to a retained* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *Q is the positive-square route: every displayed positive field element is paired with a square witness row, and every square witness remains tied to the same order classifier.*
9. *H records componentwise* hsame *transport for the real, field, order, polynomial, Sturm, isolation, root, square, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from any real-closed-field read through $R, K, O, P, S, I, A, Q, H$.*
11. *G is the* Pkg *provenance over* $\mathsf{RealClosedField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, $\mathsf{RealRootIsolation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host algebraic closure, a selected algebraic root outside the isolating window, quotient polynomial equality, classical trichotomy, choice of signs, or ambient completeness.

Theorem 8977.2 (Real closed field NameCert obligations)**.** *For every accepted* $\mathsf{RealClosedField}^{\uparrow}$ *carrier $F = (R, K, O, P, S, I, A, Q, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for the real, field, order, polynomial, Sturm, root-isolation, odd-degree root, positive-square, transport, replay, provenance, and local naming rows;*
2. *field/order compatibility through K and O on the same displayed carrier;*
3. *polynomial admission through the finite coefficient, degree, derivative, and evaluation rows of P;*
4. *odd-degree root exposure through the route P, S, I, A before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
5. *positive-square exposure through O, K, Q, with the square witness tied to the displayed order classifier;*
6. *componentwise transport, continuation replay, provenance, and local naming through H, C, G, N.*

No other real-closed-field obligation is part of the packet.

Proof. Project the carrier of Definition 8977.1. Its rows already list the complete finite surface: real boundary, field operations, order data, polynomial requests, Sturm variation, root isolation, root and square routes, and BEDC structural rows.

The odd-degree clause is read through the polynomial row P, then through the Sturm row S, then through the isolating-window row I, and only then through the root route A. Thus an accepted root read cannot be replaced by an ambient algebraic closure or selected host element.

The positive-square clause is separate. It uses the order classifier O, the field row K, and the displayed witness row Q, so positivity and squareness remain rows of the same packet.

Finally H, C, G, N transport, replay, source, and name the same rows. Since no coordinate stores host algebraic closure, quotient polynomial equality, classical trichotomy, or a choice schedule, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}$ surface. □

Theorem 8977.3 (Real closed field order-root route)**.** *Every real-closed-field read from an accepted* $\mathsf{RealClosedField}^{\uparrow}$ *packet factors through*

$$R, K, O, P \longmapsto S, I, A, Q \longmapsto H, C, G, N.$$

Odd-degree root reads must pass through the $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, *and* $\mathsf{RealRootIsolation}^{\uparrow}$ *rows before exposing a retained* $\mathsf{Real}^{\uparrow}$ *seal; positive-square reads must pass through the displayed order and field rows before exposing the square witness. No read may bypass these routes by appealing to host algebraic closure, a selected root, quotient real equality, trichotomy, choice, or ambient completeness.*

Proof. Start with an odd-degree root request. The polynomial source is the row P, so coefficients, degree, derivative, and evaluation requests are displayed before any root data is considered.

Next the Sturm and isolation rows are read. The row S supplies the sign-variation ledger, and the row I supplies the retained isolating window. The root route A can therefore expose only the retained $\mathsf{Real}^{\uparrow}$ seal attached to those rows.

For a positive-square request, the order row O supplies positivity and the field row K supplies multiplication. The witness row Q records the square witness over that same displayed carrier and classifier.

The structural rows H, C, G, N then preserve and replay the same packet. Because no listed row contains host algebraic closure, quotient equality, trichotomy, choice, or ambient completeness data, the displayed route is the whole order-root interface. □

Falsifiable prediction: Every accepted $\mathsf{RealClosedField}^{\uparrow}$ consumer that reads an odd-degree root without the $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, and $\mathsf{RealRootIsolation}^{\uparrow}$ rows, or reads a positive-square witness without the displayed field and order rows, is outside the carrier surface.

Independence witness: real, field, polynomial, sturm-sequence, real-root-isolation: no listed sibling is bijective with $\mathsf{RealClosedField}^{\uparrow}$ because this packet binds one real-facing field carrier to order data, polynomial requests, Sturm variation, isolating windows, odd-degree root exposure, positive-square witnesses, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}$.

Constructive story: A $\mathsf{RealClosedField}^{\uparrow}$ packet is generated from RealUp, FieldUp, order, PolynomialUp, SturmSequenceUp, RealRootIsolationUp, odd-degree root, positive-square, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8977.1, Definition 8977.2, and Definition 8977.3 seed the finite ordered-algebra packet for real-closed-field reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealClosedField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host algebraic closure, selected roots outside isolating windows, quotient polynomial equality, classical trichotomy, choice of signs, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, field/order compatibility, polynomial admission, odd-degree root exposure, positive-square exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealClosedFieldUp.

Chapter 8978

A Concrete Naming Certificate for $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$

Following Rockafellar convex analysis and Boyd–Vandenberghe, the $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$ packet names the biconjugation boundary by finite primal functional rows, conjugate rows, biconjugate comparison rows, lower-semicontinuity witnesses, convex epigraph rows, real-order seals, transport, replay, provenance, and local naming. It consumes the conjugacy surface of chapter 3563, the primal-dual route of chapter 321, the local subgradient boundary of chapter 3712, the convex-set carrier of chapter 208, the vector-space source of chapter 47, and the terminal real-name boundary of chapter 38. The packet records only displayed BEDC rows; it does not import host suprema, quotient function spaces, ambient locally convex completions, or an external Fenchel–Moreau theorem.

Definition 8978.1 (Fenchel–Moreau biconjugation carrier)**.** *A* $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, V, K, D, E, L, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the primal proper convex functional row read through* $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
2. *V is the vector-space source row naming the primal and dual-facing carrier interfaces;*
3. *K is the consumed* $\mathsf{ConvexConjugate}^{\uparrow}$ *row, including the pairing and displayed supremum-budget coordinates;*
4. *D is the* $\mathsf{FenchelDuality}^{\uparrow}$ *dependency row that records the primal-dual handoff through the same finite conjugacy packet;*
5. *E is the biconjugate comparison row, exposing only finite real-order and epigraph comparisons between the original functional and its displayed biconjugate read;*
6. *L is the lower-semicontinuity and closed-epigraph ledger row used to decide when the comparison row is public;*
7. *S is the* $\mathsf{Subdifferential}^{\uparrow}$ *handoff row, available to consumers that read supporting-functional evidence from the same conjugacy surface;*
8. *H records componentwise* hsame *transport for the primal functional, vector-space source, conjugate row, duality row, comparison row, closed-epigraph ledger, subdifferential handoff, replay, provenance, and local naming rows;*

9. *C records displayed* Cont *replay from primal reads to conjugate reads, from conjugate reads to biconjugate comparisons, and from those comparisons to the closed-epigraph ledger;*
10. *P is the* Pkg *provenance over* $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{Subdifferential}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FenchelMoreauBiconjugation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, K, D, E, L, S, H, C, P, N$. It has no coordinate for a host supremum operator, quotient function equality, selected support functional, ambient dual completion, or external biconjugation theorem.

Lean4 (statement) `BEDC.Derived.FenchelMoreauBiconjugationUp`

Theorem 8978.2 (Fenchel–Moreau biconjugation NameCert obligations)**.** *For every accepted* $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$ *carrier $B = (F, V, K, D, E, L, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FenchelMoreauBiconjugation}}$ *surface consists exactly of carrier admission for the displayed rows, conjugate-row exposure through K, primal-dual handoff through D, biconjugate comparison through E, closed-epigraph ledger visibility through L, optional subdifferential handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 8978.1. The visible rows are exactly $F, V, K, D, E, L, S, H, C, P, N$, so every local admission and classifier read is componentwise on those rows.

The conjugacy and duality rows are consumed before the biconjugate comparison is public. The row K supplies the displayed $\mathsf{ConvexConjugate}^{\uparrow}$ pairing and budget data, while D records that the same finite packet is available to the $\mathsf{FenchelDuality}^{\uparrow}$ route.

The comparison row E is then guarded by the lower-semicontinuity ledger L and by the real-order and convex-epigraph dependencies named in P. Transport and replay preserve only that displayed route.

Since host suprema, quotient function equality, selected support functionals, ambient completions, and external biconjugation theorems are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8978.3 (Fenchel–Moreau conjugate handoff)**.** *Every* $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, *or* $\mathsf{Subdifferential}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$ *packet receives the displayed route*

$$F, V, K, D, E, L, S \longmapsto H, C, P, N.$$

The consumer may inspect primal functional rows, vector-space source rows, conjugate rows, duality rows, biconjugate comparison rows, lower-semicontinuity ledger rows, subdifferential handoff rows, transport, replay, provenance, and local naming. It receives no host supremum, quotient function space, selected subgradient, ambient dual completion, or external Fenchel–Moreau theorem.

Proof. Use Definition 8978.2. It forces every biconjugation-facing read through the displayed rows before any structural replay or provenance row can source the answer.

For a conjugate consumer, the row K is the only path to $\mathsf{ConvexConjugate}^{\uparrow}$. For a duality consumer, the row D records the compatible $\mathsf{FenchelDuality}^{\uparrow}$ handoff. For a subdifferential consumer, the row S exposes supporting-functional evidence only after the same conjugacy and comparison rows have been displayed.

The structural rows H, C, P, N preserve, replay, source, and name that route. Because the excluded host objects are not coordinates of the carrier, the handoff supplies exactly the finite BEDC packet and no ambient analytic theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FenchelMoreauBiconjugation}^{\uparrow}}$.

Constructive story: A $\mathsf{FenchelMoreauBiconjugation}^{\uparrow}$ packet is generated from primal functional rows, vector-space source rows, conjugate and duality rows, biconjugate comparison rows, lower-semicontinuity ledger rows, subdifferential handoff rows, transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FenchelMoreauBiconjugation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8978.1, Definition 8978.2, and Definition 8978.3 seed the finite biconjugation NameCert surface over the existing convex-analysis siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FenchelMoreauBiconjugation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Fenchel–Moreau equality, selected suprema, quotient function equality, ambient dual completion, global separation theorems, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, conjugate-row exposure, duality handoff, biconjugate comparison, closed-epigraph ledger visibility, subdifferential handoff, transport, replay, provenance, and local naming for BEDC.Derived.FenchelMoreauBiconjugationUp.

Chapter 8979

A Concrete Naming Certificate for $\mathsf{AskSigGapRoute}^{\uparrow}$

The $\mathsf{AskSigGapRoute}^{\uparrow}$ packet realizes the L3–L5 kernel spine of chapter 9352 as finite route data. It is a structural reconstruction of the generated asking, signature, package, gap, exact-globalization, and naming-certificate interface: no mature analytic, algebraic, geometric, or computational object is added by the carrier. The packet gives later finite-data classifiers one concrete NameCert landing surface for the route from Ask events to visible Pkg tokens, Gap ledgers, ExactGlobalizeBase fields, and local NameCert rows.

The same generated Ask/Sig/Pkg/Gap route is read by downstream finite-name consumers before Real and Completion arguments inspect generated classifier tokens. In particular, the finite $\mathsf{StreamName}^{\uparrow}$ scheduling surface of chapter 126 and the regular rational readback surface of chapter 301 consume this route as a visible token boundary, not as a semantic oracle or hidden quotient classifier.

Definition 8979.1 (Ask–Sig–Pkg–Gap route carrier)**.** *An* $\mathsf{AskSigGapRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, D, A, S, E, P, G, X, N, H, C, L)$$

with the following displayed rows:

1. *B is the probe-bundle row, naming the finite* Bundle *of probes whose answers generate the route.*
2. *D is the admitted-domain row over source histories.*
3. *A is the* Ask *row, recording each probe request, emitted mark, and event evidence for admitted histories.*
4. *S is the* Sig *row, recording the generated answer history and evidence ledger assembled from A.*
5. *E is the* sameSig *row, comparing generated answer histories only by internal* hsame *witnesses.*
6. *P is the* Pkg *row, naming visible package tokens for generated answer histories.*
7. *G is the* Gap *row, recording package-token membership for admitted source histories.*
8. *X is the* ExactGlobalizeBase *row, carrying coverage, base soundness, and base completeness for the displayed gap-signature route.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}$ *row, with source, pattern, classifier, stability, and ledger fields read from B, D, A, S, E, P, G, X.*

10. *H records componentwise* hsame *transport for bundle, domain, ask, signature, signature-sameness, package, gap, exactness, naming, replay, and ledger rows.*
11. *C records displayed* Cont *replay from a route request through $B, D, A, S, E, P, G, X, N, H$.*
12. *L is the* Pkg *provenance row for* $\mathsf{AskSigGapRoute}^{\uparrow}$, Ask, Sig, Pkg, Gap, ExactGlobalizeBase, NameCert, BHist, hsame, *and* Cont.

The classifier compares accepted route packets only through these rows. It has no coordinate for a semantic oracle, hidden source classifier, quotient package equality, external global object, or mature downstream carrier.

Theorem 8979.2 (Ask–Sig–Pkg–Gap route NameCert obligations)**.** *For every accepted* $\mathsf{AskSigGapRoute}^{\uparrow}$ *carrier $R = (B, D, A, S, E, P, G, X, N, H, C, L)$, the local* $\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for bundle, domain, ask, signature, signature-sameness, package, gap, exactness, local naming, transport, replay, and provenance rows;*
2. *ask totality and determinacy only for admitted histories in D and probes in B;*
3. *signature construction and determinacy through the generated* Sig *row S and the* sameSig *row E;*
4. *package-token visibility and reflection through P;*
5. *gap coverage and same-source separation through G;*
6. *exact local classification through the coverage, base-soundness, and base-completeness fields of X;*
7. *source, pattern, classifier, stability, ledger, transport, replay, and provenance discipline through N, H, C, L.*

No NameCert obligation for a downstream mature object is stored in this route packet.

Proof. Project the carrier of Definition 8979.1. The rows B, D, A display exactly the finite probe bundle, admitted source domain, and ask-event ledger, so an ask read has no access to an external oracle.

The signature and package rows are downstream. The row S assembles answer histories from ask events, E compares those answer histories by hsame, and P exposes only visible package tokens backed by generated signatures.

The gap and exactness rows complete the route. The row G records package-token membership for admitted histories, while X supplies precisely the coverage, base-soundness, and base-completeness fields of ExactGlobalizeBase.

Finally N, H, C, L name, transport, replay, and source the same finite data. Since the carrier has no semantic oracle, hidden quotient, external classifier, or downstream mature-object coordinate, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}$ surface. □

Theorem 8979.3 (Ask–Sig–Pkg–Gap route exactness)**.** *Every exact route read from an accepted* $\mathsf{AskSigGapRoute}^{\uparrow}$ *packet factors through*

$$B, D, A \longmapsto S, E \longmapsto P, G, X \longmapsto N, H, C, L.$$

Thus two visible package tokens are classified together only after the corresponding ask-generated signatures, gap memberships, and ExactGlobalizeBase *fields have been displayed. The route exports no quotient package equality, semantic source equality, external global classifier, or mature downstream object.*

Proof. Start with the source side. The bundle row B and domain row D fix which probes may be asked, and A records the emitted marks and evidence rows for those admitted asks.

The signature side is then forced through S and E. The generated answer histories are assembled by the signature row and compared only by the internal hsame witness carried by sameSig.

The visible-token side is forced through P and G. Package tokens are introduced from generated answer histories, and gap memberships tie those tokens back to admitted source histories.

The exactness row X supplies coverage, base soundness, and base completeness for exactly those displayed memberships. Therefore local classification reaches package-token sameness only through the route already named, and the structural rows N, H, C, L cannot add a quotient, oracle, external classifier, or downstream carrier. □

Falsifiable prediction: Every accepted $\mathsf{AskSigGapRoute}^{\uparrow}$ consumer that reads package-token exactness without first displaying the probe bundle, admitted domain, ask events, generated signatures, signature-sameness row, package row, gap row, and ExactGlobalizeBase row is outside the carrier surface.

Independence witness: ask, sig, pkg, gap, exact-globalize-base, namecert: no single listed kernel component is bijective with $\mathsf{AskSigGapRoute}^{\uparrow}$ because this packet binds them into one finite route from probe requests to visible exact package classification.

Closure status for $\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AskSigGapRoute}^{\uparrow}$ packet is generated from a probe-bundle row, admitted-domain row, Ask event rows, Sig generation rows, sameSig comparison rows, Pkg token rows, Gap membership rows, ExactGlobalizeBase fields, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8979.1, Definition 8979.2, and Definition 8979.3 seed the finite L3–L5 route from ask events to exact package classification.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AskSigGapRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a mature downstream object, quotient package equality, semantic source equality, an external global classifier, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ask determinacy, signature determinacy, package visibility, gap coverage, exact base classification, transport, replay, provenance, and local naming for BEDC.Derived.AskSigGapRouteUp.

Chapter 8980

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$

Following Bourbaki and Kelley on completion and Hausdorff reflection, the $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ packet records the finite unit-counit surface by which a displayed completion source is routed through separated reflection before any regular-rational or Real-facing seal is consumed. It sits over the completion carrier of chapter 95, the metric-completion carrier of chapter 1506, the separated-completion boundary of chapter 1612, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the adjointness route as displayed BEDC data; it does not identify quotient completion classes, choose representatives, import countable choice, compare host completed objects, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8980.1 (Hausdorff completion adjointness carrier)**.** *A* $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (C, H, U, V, S, R, E, T, K, P, N)$$

with the following displayed rows:

1. *C is the completion source row, recording the* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *request, its inspected source carrier, and the finite completion-facing branch exposed before any separated reflection is read.*
2. *H is the Hausdorff or separated reflection row, tying the completion source to the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary used to compare completed names.*
3. *U is the unit row, sending each displayed source observation into the completion-reflection packet through C and H.*
4. *V is the counit row, returning any completion-facing separated consumer to the same displayed completion-reflection surface.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite observation windows that must be open before the unit or counit rows can be consumed by downstream readers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing regular rational data only after C, H, U, V, S have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only downstream of the completion source, separated reflection, unit row, counit row, StreamName windows, and RegSeqRat readback.*

8. *T records componentwise* hsame *transport for the completion source, Hausdorff reflection, unit, counit, StreamName windows, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay from the completion source through reflection, unit, counit, finite windows, rational readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionAdjointness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, H, U, V, S, R, E, T, K, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, host equality of completed objects, detached Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.HausdorffCompletionAdjointnessUp`

Theorem 8980.2 (Hausdorff completion adjointness NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ *carrier $A = (C, H, U, V, S, R, E, T, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionAdj}}$ *surface consists exactly of the following five obligation rows:*

1. *completion-source admission, requiring the displayed* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *row C before any adjointness data are read;*

2. *Hausdorff-reflection admission, requiring the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row H before completed names may be compared;*

3. *unit-counit visibility, requiring the unit row U and counit row V to be exposed as finite rows over the same C, H route;*

4. *finite readback routing, requiring* $\mathsf{StreamName}^{\uparrow}$ *windows through S and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R before the* $\mathsf{Real}^{\uparrow}$ *seal E is available;*

5. *nonescape under structure, requiring T, K, P, N to transport, replay, source, and name only the displayed route, with no quotient completion equality, selected representatives, countable choice, host equality, detached Real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8980.1. The completion source is C, the separated-reflection boundary is H, and the unit-counit rows are exactly U, V.

The readback rows are downstream of that route. The finite StreamName windows are stored in S, the regular rational readback is stored in R, and the Real-facing seal is stored in E.

The structural rows cannot add new analytic content. The row T transports the displayed components by hsame, K replays the same route by Cont, P records provenance, and N names the local packet.

Since the carrier has no coordinate for the excluded quotient, selector, choice, host-equality, detached-Real, or ambient-completeness objects, the five listed rows exhaust the NameCert surface. □

Theorem 8980.3 (Hausdorff completion adjointness unit-counit route)**.** *Every completion-facing consumer of an accepted* $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ *carrier must expose the unit row U and counit row V over the finite route*

$$C \longmapsto H \longmapsto U, V \longmapsto S \longmapsto R \longmapsto E \longmapsto T, K, P, N$$

before any $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is read. In particular, the packet permits downstream completion, regular-rational, and Real consumers to see the adjointness boundary only as displayed unit-counit data over the same completion and Hausdorff-reflection rows.*

Proof. Start with a completion-facing consumer. Carrier admission exposes no rational readback or Real seal until the completion source C and Hausdorff-reflection row H have been read.

The unit and counit rows are the next public rows on the route. The row U sends source observations into the completion-reflection packet, while V returns separated completion consumers to that same packet.

Only after U, V are visible can the consumer inspect finite observations and rational readback. These are the rows S and R, and the terminal Real seal E is downstream of both.

Finally T, K, P, N transport, replay, source, and name the displayed route. They cannot introduce quotient classes, selected representatives, countable choice, host equality, detached Real equality, or ambient completeness because no such coordinate is carried. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ completion-facing read exposes a completion source row, a Hausdorff or separated reflection row, unit and counit rows, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data before any downstream consumer reads the terminal seal.

Independence witness: completion, metriccompletion, separatedcompletion, streamname, regseqrat, real: $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries completion source admission, Hausdorff reflection, unit and counit visibility, finite-window routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionAdjointness}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionAdjointness}^{\uparrow}$ packet is generated from a completion source row, a Hausdorff or separated reflection row, a unit row, a counit row, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionAdjointness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8980.1, Definition 8980.2, and Definition 8980.3 seed the completion adjointness boundary over completion source, Hausdorff reflection, unit, counit, finite-window, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionAdjointness}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.HausdorffCompletionAdjointnessUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality, detached Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Hausdorff reflection exposure, unit and counit visibility, StreamName window routing, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffCompletionAdjointnessUp.

Chapter 8981

A Concrete Naming Certificate for $\mathsf{ApollonianGasket}^{\uparrow}$

Following Descartes circle theorem and the classical Apollonian gasket construction, the $\mathsf{ApollonianGasket}^{\uparrow}$ packet records one finite replacement window of mutually tangent circles before any infinite packing, limiting set, or measure-theoretic boundary is read. It sits beside the circle carrier of chapter 4010, the Ford-circle tangency ledger of chapter 8445, the metric source of chapter 57, the dyadic interval-net route of chapter 5966, and the terminal real-name seal of chapter 38. The packet names only finite Descartes replacement data and its local sibling lattice.

Definition 8981.1 (Apollonian gasket carrier)**.** *An* $\mathsf{ApollonianGasket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (C, T, K, D, M, R, L, H, Q, P, N)$$

with the following displayed rows:

1. *C is the ordered Descartes cluster row, listing four displayed* $\mathsf{Circle}^{\uparrow}$ *entries together with their tangency incidence.*
2. *T is the tangent-pair ledger, recording exactly the six pairwise tangencies inspected in the current window.*
3. *K is the curvature row, carrying the four signed curvature reads used by the Descartes relation.*
4. *D is the replacement row, naming the alternate circle obtained by replacing one member of the cluster while keeping the other three tangent circles fixed.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *contact row used to read tangency and separation in the ambient displayed metric surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for curvature, center-coordinate, and contact-radius comparisons.*
7. *L is the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite localization row for the inspected centers and radii.*
8. *H records componentwise* hsame *transport for the cluster, tangency, curvature, replacement, metric, real, localization, replay, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from a Descartes replacement request through $C, T, K, D, M, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{ApollonianGasket}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, $\mathsf{FordCircle}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{ApollonianGasket}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $C, T, K, D, M, R, L, H, Q, P, N$. It has no coordinate for an infinite gasket, a limiting dust set, a selected infinite replacement branch, a completed curvature spectrum, area measure, or host Euclidean plane.

Theorem 8981.2 (Apollonian gasket NameCert obligations)**.** *For every accepted* $\mathsf{ApollonianGasket}^{\uparrow}$ *carrier*

$$G = (C, T, K, D, M, R, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApollonianGasket}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Descartes cluster, tangent-pair ledger, curvature row, one-circle replacement row, metric contact row, Real readback row, dyadic localization row, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 8981.1. The rows C, T, K, D, M, R, L are the mathematical data of the inspected replacement window, and H, Q, P, N are the structural rows that transport, replay, source, and name the same packet.

The Descartes cluster and tangent-pair ledger expose only four circles and six tangencies. The curvature row and replacement row then name one finite replacement step, not an infinite branch of replacements.

The metric, Real, and dyadic localization rows keep contact and curvature reads inside displayed finite data. Since no carrier coordinate stores a limiting set, selected infinite path, completed curvature spectrum, or measure surface, the listed rows exhaust the NameCert obligations. □

Theorem 8981.3 (Apollonian gasket Descartes replacement stability)**.** *Let $G = (C, T, K, D, M, R, L, H, Q, P, N)$ be an accepted* $\mathsf{ApollonianGasket}^{\uparrow}$ *carrier. Replacing one circle of the displayed Descartes cluster by the alternate circle named in D preserves the public route*

$$C, T, K, D, M, R, L, H, Q, P, N$$

for the resulting finite window. The replacement can be consumed through the same circle, metric, dyadic localization, and Real readback rows, and it exports no infinite gasket, selected replacement branch, limiting dust, area measure, or ambient-plane completeness principle.

Proof. Begin with the tangent-pair ledger T. The replacement row D keeps the three fixed tangent circles visible and records the alternate circle only as a row of the same finite carrier.

Read the curvature row K through the Descartes relation. The Real readback R supplies the displayed curvature and contact-radius comparisons, while the metric row M supplies the tangency and separation vocabulary used by the local replacement.

Use the dyadic localization row L to keep centers and radii inside the inspected finite window. Componentwise transport H and replay Q then carry the replacement packet forward without changing its public rows.

Finally P and N source and name the resulting window as an $\mathsf{ApollonianGasket}^{\uparrow}$ packet. Because the carrier has no infinite-branch, measure, limiting-set, or ambient-plane coordinate, the replacement stability statement is confined to the displayed Descartes window. □

Theorem 8981.4 (Apollonian gasket sibling lattice)**.** *Every* $\mathsf{ApollonianGasket}^{\uparrow}$ *read factors through the sibling lattice*

$$\mathsf{Circle}^{\uparrow},\ \mathsf{FordCircle}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{DyadicIntervalNet}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before a consumer may inspect the Descartes replacement row. The circle and Ford-circle rows supply tangency vocabulary, the metric row supplies contact comparison, the dyadic interval-net row supplies finite localization, and the Real row supplies curvature and radius readback.

Proof. Project the provenance row P of Definition 8981.1. It names the circle, Ford-circle, metric, dyadic localization, and Real siblings as the only external packet surfaces available to an Apollonian replacement read.

The Descartes cluster row C and tangent ledger T are read through the circle and Ford-circle surfaces. The metric row M then fixes the contact relation before curvature and radius data are read by R.

The dyadic interval-net row L supplies the finite localization boundary for centers and radii. Transport, replay, provenance, and local naming preserve the same route, so downstream consumers receive a finite sibling lattice rather than an ambient plane, a completed packing, or a hidden choice of replacement branch. □

Falsifiable prediction: Every accepted $\mathsf{ApollonianGasket}^{\uparrow}$ replacement read displays four circle rows, six tangent-pair rows, curvature readback, one replacement row, metric contact, dyadic localization, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert data before any gasket consumer may read it.

Independence witness: circle, ford-circle, metric, dyadic-interval-net, real: none of the listed siblings jointly carries a Descartes cluster, tangent-pair ledger, curvature row, one-circle replacement row, localization row, transport, replay, provenance, and local NameCert data as one finite gasket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ApollonianGasket}^{\uparrow}}$.

Constructive story: An $\mathsf{ApollonianGasket}^{\uparrow}$ packet is generated from a finite Descartes circle cluster, tangent-pair ledger, curvature row, one-circle replacement row, metric contact row, Real readback row, dyadic localization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApollonianGasket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8981.1, Definition 8981.2, Definition 8981.3, and Definition 8981.4 seed the finite Descartes replacement route for one Apollonian gasket window.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApollonianGasket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an infinite gasket, a limiting dust set, a selected infinite replacement branch, a completed curvature spectrum, area measure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Descartes tangency exactness, replacement stability, sibling-lattice routing, transport, replay, provenance, and local naming for BEDC.Derived.ApollonianGasketUp.

Chapter 8982

A Concrete Naming Certificate for $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$

Following Bishop and Bridges on interval bisection and regular Cauchy construction, the FiniteIntervalBisectionSelect packet records the finite selector used when an L10 Real or Completion route must choose a displayed half of a finite interval cell. It sits after the endpoint-order lattice of chapter 6006, the dyadic interval cells of chapter 306, the stream schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The selector supplies a finite bisection handoff; it is not an open-cover compactness principle, a quotient real equality row, an ambient completeness theorem, or a choice function over all intervals.

Definition 8982.1 (Finite interval bisection selector carrier)**.** *A* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, I, B, S, Q, R, H, C, P, N)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order row naming the lower endpoint, midpoint, and upper endpoint comparisons used by the bisection.*
2. *I is the* $\mathsf{DyadicInterval}^{\uparrow}$ *cell row naming the displayed parent interval and its two finite half-cells.*
3. *B is the selected-half row, recording whether the accepted continuation consumes the left half, the right half, or a refusal row when the half predicate is not displayed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite endpoint and midpoint observations used by the selector.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations attached to the selected half.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, I, B, S, Q have remained visible.*
7. *H records componentwise* hsame *transport for endpoint order, interval cells, selected-half data, stream windows, readback, Real sealing, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a parent interval request to the selected half and then to the regular-rational and Real-facing rows.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$, $\mathsf{FiniteIntervalLattice}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}}$ *row.*
The carrier has no coordinate for a global interval choice function, a selected limit point, quotient equality of real names, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8982.2 (Finite interval selected-half exactness)**.** *Let $F = (D, I, B, S, Q, R, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ *carrier. Every selected-half read from B must expose the parent interval, midpoint split, and chosen half through the dyadic endpoint-order row D and the displayed cell row I before it can be consumed by a StreamName, RegSeqRat, or Real-facing continuation. The packet therefore selects only one of the two listed half-cells of I and exports no hidden representative, quotient real equality, open-cover compactness, or ambient completeness principle.*

Proof. Project the carrier of Definition 8982.1. The rows D and I display the lower endpoint, midpoint, upper endpoint, parent interval, and the two finite half-cells before the selector row is read.

Use the endpoint-order and Real-completion route of chapter 6006. That route already forces interval consumers to keep dyadic endpoint evidence and finite cell membership visible before analytic handoff.

The selected-half row B is then interpreted only relative to those displayed endpoint and cell rows. If the requested half is absent, the refusal row remains inside B; if it is present, the chosen half is one of the two cells listed by I.

Finally S, Q, R, H, C, P, N preserve the same displayed route by stream scheduling, regular rational readback, Real sealing, transport, replay, provenance, and local naming. None of those rows introduces a hidden representative, quotient real equality, open-cover compactness, or ambient completeness coordinate. □

Theorem 8982.3 (Finite interval bisection refinement induction)**.** *Let $F = (D, I, B, S, Q, R, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ *carrier. For every displayed finite bisection depth, iterating the selector rebuilds a finite chain of dyadic half-cells whose endpoint comparisons remain in D, whose interval cells remain in I, whose selected-half rows remain in B, and whose analytic handoff remains the route*

$$S, Q, R, H, C, P, N.$$

Thus refinement induction supplies only a finite chain of displayed bisections and cannot introduce an infinite nested interval object, selected limit point, countable choice schedule, quotient real equality, or ambient completion theorem.

Proof. Induct on the displayed bisection depth. At depth zero the chain is the parent cell named by I, with endpoint order and analytic handoff already recorded by the carrier.

For the successor step, apply selected-half exactness, Definition 8982.2. The next half-cell is one of the two displayed halves of the current cell, and its lower endpoint, midpoint, and upper endpoint comparisons are still read through D.

The finite interval lattice refinement induction of Definition 6006.5 supplies the table-level refinement discipline for the surrounding cell lattice. The selector adds only the row recording which displayed half is consumed at the current step.

Finally the stream schedule, regular rational readback, Real seal, transport, replay, provenance, and local naming rows remain S, Q, R, H, C, P, N at every finite stage. The induction therefore builds a finite displayed chain and never supplies an infinite nested object, selected limit point, countable choice schedule, quotient real equality, or ambient completion theorem. □

Theorem 8982.4 (Finite interval bisection RegSeqRat handoff)**.** *Every RegSeqRat-facing consumer of an accepted* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{FiniteIntervalLattice}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}.$$

The selected-half row B may narrow the visible interval cell, but it does not create a rational sequence outside the displayed stream schedule or bypass the endpoint-order evidence.

Proof. Begin with the carrier of Definition 8982.1. The dyadic endpoint row D, interval-cell row I, selected-half row B, and stream schedule row S are upstream of the regular rational readback row Q.

Apply selected-half exactness. It forces B to choose only a displayed half-cell whose endpoint order is already visible through D and I.

Apply refinement induction. Every finite tail of repeated bisection remains a finite chain of displayed cells, so a RegSeqRat-facing readback sees the same endpoint and stream-window route at each finite stage.

Thus Q is reached only through the displayed finite interval lattice, dyadic interval, dyadic endpoint, and stream schedule rows. No quotient stream, selected representative, countable-choice subsequence, or ambient Real-completion route is imported by the handoff. □

Theorem 8982.5 (Finite interval bisection Real-seal nonescape)**.** *Let $F = (D, I, B, S, Q, R, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ *carrier. A* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer reaches the terminal seal R only after the dyadic endpoint row D, displayed interval-cell row I, selected-half row B, stream schedule S, and RegSeqRat readback Q have been exposed. The Real-facing seal therefore records the same finite bisection route and not a chosen limit, quotient real equality, open-cover compactness principle, or ambient completeness witness.*

Proof. Use the RegSeqRat handoff of Definition 8982.4. It places every regular rational readback downstream of the finite interval lattice, dyadic interval, endpoint-order, selected-half, and stream schedule rows.

The Real-completion route of chapter 6006 keeps analytic consumers inside the displayed finite lattice route before a Real-facing seal is read.

Finally replay the structural rows H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same packet, so the terminal row R cannot add a chosen limit, quotient real equality, open-cover compactness principle, or ambient completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}$ packet is generated from dyadic endpoint-order rows, displayed interval cells, a selected-half row, finite stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}}$, seed)

Scope closed: Definition 8982.1, selected-half exactness, refinement induction, RegSeqRat handoff, and Real-seal nonescape seed the finite bisection selector route for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteIntervalBisectionSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global interval choice, infinite nested interval convergence, selected limits, quotient real equality, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-half exactness, finite refinement induction, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalBisectionSelecto

Chapter 8983

A Concrete Naming Certificate for $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$

The $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ chapter realizes the omega generation structure interface of chapter 9361 as one finite NameCert carrier. The packet binds licensed interface objects, licensed morphisms, generation diagrams, legal stream displays, pairing rows, time rows, transport, replay, provenance, and local naming before any tower coordinate is read. It is an omega-boundary certificate, not a global tower object, not a quotient of generated bodies, and not an unlicensed selector for hidden coordinates.

Omega boundary reads consume the kernel acceptance boundary of chapter 6007 and the machine export boundary of chapter 5205 before exposing coordinate-facing rows. The acceptance boundary supplies the licensed public row, while the export boundary supplies the machine-facing read discipline for legal displays, pairing rows, and time rows.

Definition 8983.1 (Omega generation boundary packet carrier)**.** *An* $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, M, D, L, P, T, H, C, K, N)$$

with the following rows:

1. *A is the licensed interface-object row of Definition 9361.1, including carrier, public classifier, ledger, and displayed NameCert fields.*
2. *M is the licensed morphism row of Definition 9361.2, recording classifier, ledger, and continuation preservation.*
3. *D is the generation-diagram row of Definition 9361.4, including displayed readback morphisms and compatibility witnesses.*
4. *L is the legal stream display row of Definition 9361.5, including public stream classifier, stream ledger, and decode relation.*
5. *P is the pairing row, recording only typed involutive layers whose source interface licenses pairing behavior.*
6. *T is the time-row, recording generated one-step extension and continuation direction rather than an involution.*
7. *H records componentwise* hsame *transport for interface, morphism, generation diagram, legal display, pairing, time, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a boundary read through the licensed interface, morphism, diagram, display, pairing, and time rows.*
9. *K is the* Pkg *provenance over* $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$*, licensed interface objects, licensed morphisms, generation diagrams, legal stream displays, pairing rows, time rows,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unlicensed tower element, hidden body equality, raw code stream, global inverse to time, host quotient of generated bodies, or external omega-completion principle.

Theorem 8983.2 (Omega generation boundary packet row exposure)**.** *Every accepted* OmegaGenerationBoundaryP *consumer that reads a tower coordinate must first display the licensed interface object row A, licensed morphism row M, generation-diagram row D, legal stream display row L, pairing row P, time row T, transport row H, replay row C, provenance row K, and local naming row N. Thus a coordinate read is accepted only through the finite boundary carrier and not through raw body equality, unlicensed code, hidden tower selection, or an ambient omega object.*

Proof. Project the carrier of Definition 8983.1. Its first six rows display the licensed object, morphism, diagram, legal stream, pairing, and time surfaces before any coordinate-facing conclusion is public.

Apply the no-unlicensed-coordinate principle of Definition 9361.3. It requires every visible omega coordinate to be represented by carrier, classifier, stability, ledger, and NameCert rows before tower language is accepted.

Use the omega generation structure package of Definition 9361.20. Encoded reflection, typed pairing, and generated time direction all live over the licensed carrier, classifier, ledger, and NameCert rows already exposed by A, M, D, L, P, T.

Finally replay H, C, K, N. Componentwise transport, displayed continuation replay, package provenance, and local naming preserve the same finite boundary packet and add no raw body equality, unlicensed code, hidden tower selector, or ambient omega object. □

Theorem 8983.3 (Omega generation boundary packet nonescape)**.** *An accepted* $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ *read cannot replace the displayed legal stream row by a raw stream, cannot replace typed pairing by time direction, and cannot replace the time row by an involutive mirror. The only public route is the carrier*

$$A, M, D, L, P, T, H, C, K, N.$$

Proof. Use row exposure, Definition 8983.2. It puts the legal stream display, pairing row, and time row on the same finite boundary route.

The legal stream display of Definition 9361.5 is a certified public display with ledger-visible source rows, not a raw bit string. The pairing source discipline of Definition 9361.12 permits pairing only where an involution is licensed.

The time separation theorem Definition 9361.18 and the principle Definition 9361.19 keep generated extension direction separate from pairing. Therefore no accepted read can trade raw code, untyped pairing, or a global inverse-time map for the displayed boundary route. □

Falsifiable prediction: Every accepted $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ consumer that reads a tower coordinate without first displaying licensed interface, morphism, generation-diagram, legal-stream, pairing, and time rows is outside the carrier surface.

Independence witness: finite-interface-spine, ask-sig-gap-route, kernel-category, finite-data-assembly, kernel-acceptance-boundary: none is bijective with the omega boundary packet because the packet binds all licensed-interface, display, pairing, and time rows in one carrier.

The Lean follow-up for this packet is the $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ taste gate, whose public surface is expected to provide a nontriviality instance and a field-faithfulness instance for the boundary carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}}$.

Constructive story: An $\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}$ packet is generated from licensed interface-object rows, licensed morphism rows, generation-diagram rows, legal stream display rows, pairing rows, time rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8983.1, row exposure, nonescape, falsifiability, and independence rows seed the finite boundary carrier for omega generation coordinate reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OmegaGenerationBoundaryPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global omega tower, quotient body equality, raw stream coding, inverse-time symmetry, RH rigidity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, licensed-interface exposure, legal-stream display, pairing-time separation, transport, replay, provenance, local naming, and taste-gate instances for BEDC.Derived.OmegaGenerationBoundaryPa

Chapter 8984

A Concrete Naming Certificate for $\mathsf{NetAdherence}^{\uparrow}$

Following Kelley and Bourbaki on nets and filters, the $\mathsf{NetAdherence}^{\uparrow}$ packet records the finite boundary at which a displayed net, a generated filterbase, and neighbourhood requests meet an adherence or cluster-point test. It sits downstream of the convergence interface of chapter 1667, the Cauchy-net window discipline of chapter 589, the filterbase carrier of chapter 1601, the neighbourhood language of chapter 88, and the completion-facing seal of chapter 95. The packet names finite hit evidence; it is not a selected subnet, a choice principle, a quotient completion, or ambient real completeness.

Definition 8984.1 (Net adherence carrier)**.** *A* $\mathsf{NetAdherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, N, F, U, A, H, C, P, L)$$

with the following displayed rows:

1. *D is the directed-index row, naming the finite index window and the tail requests available to the net-side reader.*
2. *N is the net-sample row, recording the inspected values of the net on the displayed directed window.*
3. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row generated from the displayed tail family.*
4. *U is the neighbourhood-request row, read through the local* $\mathsf{Topology}^{\uparrow}$ *surface and only at the displayed candidate boundary.*
5. *A is the adherence-hit row, recording the finite evidence that a tail or filterbase member meets the requested neighbourhood.*
6. *H records componentwise* hsame *transport for directed indices, samples, filterbase rows, neighbourhood requests, hits, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay for any consumer that asks for the same hit evidence.*
8. *P is the* Pkg *provenance over* $\mathsf{NetAdherence}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}$ *row, whose source, pattern, classifier, stability, and ledger fields are read from D, N, F, U, A, H, C, P.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected subnet, cofinal choice over the directed set, quotient equality of filters, a completed ambient point, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8984.2 (Net adherence FilterBase obligations)**.** *For every accepted* $\mathsf{NetAdherence}^{\uparrow}$ *carrier* $A = (D, N, F, U, A, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed-index row, net-sample row, generated filterbase row, neighbourhood-request row, adherence-hit row, transport row, replay row, provenance row, and local naming row;*
2. *directed-tail discipline, requiring every hit read to expose the finite tail request in* D *before it inspects a sample in* N*;*
3. *filterbase generation, requiring the* $\mathsf{FilterBase}^{\uparrow}$ *row* F *to be read from the same displayed tail family and not from a host filter equality;*
4. *neighbourhood discipline, requiring the request row* U *to be interpreted through chapter 88;*
5. *hit stability, requiring the adherence-hit row to transport along the componentwise* hsame *row* H *and replay through* C*;*
6. *provenance and local naming through* P *and* L*.*

No obligation for selected subnet construction, choice over tails, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is stored in this packet.*

Proof. Project the accepted carrier of Definition 8984.1. The source-side rows are D, N, F, U, A, and the structural rows are H, C, P, L.

The directed-tail evidence is displayed first. A hit read has to name the finite index window in D and the inspected net samples in N before any filterbase or neighbourhood row can be consumed.

The filterbase and topology rows then fix the analytic boundary. The row F is generated from the same displayed tails, while U records only the finite neighbourhood requests supplied by the topology-side surface. Thus the adherence hit is a finite meet between a displayed tail family and a displayed neighbourhood request.

Finally H, C, P, L transport, replay, source, and name the same packet. Since no coordinate stores a selected subnet, a choice operator, quotient filter equality, a completed ambient point, or real completeness, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}$ obligations. □

Theorem 8984.3 (Net adherence Cauchy-net cluster route)**.** *Every completion-facing or cluster-point-facing consumer of an accepted* $\mathsf{NetAdherence}^{\uparrow}$ *carrier factors through*

$$D, N \longmapsto F, U \longmapsto A \longmapsto H, C, P, L.$$

The route may be read after the $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *rows of chapter 1667 and chapter 589, may hand its filterbase row to chapter 1601, and may be consumed by completion arguments in chapter 95. It exports finite adherence evidence only; it does not select a subnet, choose a cluster point, identify quotient filters, or import ambient completeness.*

Proof. Begin with a completion-facing request. The carrier first exposes the directed-index and sample rows D, N, so the request sees a finite net window rather than a host net predicate.

The next boundary is the filterbase and neighbourhood interface. The generated row F gives the filterbase side, and U records the topology-side neighbourhood request. Their common display is exactly the finite surface needed by adherence and cluster arguments.

The hit row A is then read as evidence that the displayed tail or filterbase member meets the displayed neighbourhood request. The structural rows H, C, P, L preserve that evidence under transport, replay it for downstream consumers, record provenance, and bind the local name.

Thus a Cauchy-net or completion consumer receives only the finite route through $D, N, F, U, A, H, C, P, L$. The packet cannot add subnet selection, cluster-point choice, quotient-filter equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness because none of those data appear among its rows. □

Falsifiable prediction: Every accepted $\mathsf{NetAdherence}^{\uparrow}$ consumer that reads adherence or cluster evidence without displaying the directed-index row, net-sample row, filterbase row, neighbourhood-request row, adherence-hit row, transport, replay, provenance, and local NameCert rows is outside the carrier surface.

Independence witness: net, filterbase, topology, completion: no single listed sibling is bijective with $\mathsf{NetAdherence}^{\uparrow}$ because this packet binds directed net samples, generated filterbase rows, neighbourhood requests, and finite hit evidence into one local NameCert boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}$.

Constructive story: A $\mathsf{NetAdherence}^{\uparrow}$ packet is generated from a directed-index row, net-sample row, generated FilterBase row, topology-side neighbourhood-request row, adherence-hit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8984.1, Definition 8984.2, and Definition 8984.3 seed the finite net/filter adherence boundary used by completion and cluster-point arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NetAdherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subnets, choice over directed tails, quotient filter equality, selected cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail discipline, filterbase generation, neighbourhood discipline, hit stability, transport, replay, provenance, and local naming for BEDC.Derived.NetAdherenceUp.

Chapter 8985

A Concrete Naming Certificate for $\mathsf{KakeyaSet}^{\uparrow}$

Following Besicovitch and Kakeya on direction-rich plane sets, the $\mathsf{KakeyaSet}^{\uparrow}$ packet records only the finite BEDC geometry needed to name a displayed direction-window and segment-incidence carrier. It sits over the terminal real-name surface of chapter 38, the metric certificate of chapter 57, the dyadic interval rows of chapter 8927, and the incidence bookkeeping of chapter 8975. The packet is a finite naming certificate for direction-window geometry; it does not claim measure zero, the Kakeya conjecture, Besicovitch set existence, ambient planar completeness, or a host set-theoretic construction.

Definition 8985.1 (KakeyaSet carrier)**.** *A* $\mathsf{KakeyaSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, S, I, M, R, H, C, P, N)$$

with the following displayed rows:

1. *D is a finite direction-window row, listing named rational or dyadic direction samples and the tolerance cells that cover the inspected direction interval.*
2. *W is the window ledger, recording the finite observation windows in which the direction samples and segment endpoints are read.*
3. *S is the segment row, assigning to each displayed direction sample one finite segment name with endpoint rows over* $\mathsf{Real}^{\uparrow}$*.*
4. *I is the segment-incidence row, recording which sampled direction, endpoint pair, and window row belong to the same displayed segment.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used to read endpoint distance and segment length only through displayed metric data.*
6. *R is the real-name handoff row, tying every endpoint coordinate and tolerance cell to the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DyadicRealInterval}^{\uparrow}$ *sibling surfaces.*
7. *H records componentwise* hsame *transport for direction windows, segment rows, incidence rows, metric comparisons, real-name handoff, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a direction request through the finite window, segment-incidence row, metric row, and real-name handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{KakeyaSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KakeyaSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, S, I, M, R, H, C, P, N$. It has no coordinate for outer measure, null sets, arbitrary direction choice, infinite unions, host subsets of the plane, or any assertion that a Besicovitch set exists.

Theorem 8985.2 (KakeyaSet NameCert obligations)**.** *For every accepted* $\mathsf{KakeyaSet}^{\uparrow}$ *carrier*

$$K = (D, W, S, I, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KakeyaSet}^{\uparrow}}$ *surface consists exactly of finite direction-window admission for D, finite observation-window display for W, segment-row admission for S, segment-incidence exactness for I, metric comparison display for M, real-name and dyadic-interval handoff for R, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No KakeyaSet obligation is stored outside these rows.

Proof. Project the accepted carrier. The finite direction-window row D and the observation ledger W are the only rows from which directions may be read, so the packet never introduces an arbitrary direction choice.

The segment data are then read through S and I. These rows bind each sampled direction to a displayed segment endpoint pair and to the window in which that pair is observed.

Finally the metric and real-name handoff rows M and R keep all endpoint and length comparisons inside the sibling Real, dyadic-interval, and metric surfaces. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the local certificate contains no measure-theoretic or host set-theoretic coordinate. □

Theorem 8985.3 (KakeyaSet direction-window route)**.** *Every consumer that reads a* $\mathsf{KakeyaSet}^{\uparrow}$ *packet as direction-rich geometry must first expose the finite route*

$$D, W, S, I, M, R, H, C, P, N.$$

The direction-window row and segment-incidence row are public before any geometric consumer may compare directions or endpoints. The route exports finite incidence geometry only; it does not assert measure zero, the Kakeya conjecture, Besicovitch set existence, arbitrary direction selection, or a completed host subset of the plane.

Proof. Start from the carrier of Definition 8985.1. The only direction data are the finite direction samples and tolerance cells stored in D, with their read windows stored in W.

Apply the NameCert obligations of Definition 8985.2. They require the segment row S and incidence row I before any consumer can attach a segment to a direction window.

The metric row M and real-name handoff row R then route endpoint and length reads through chapter 57, chapter 38, and chapter 8927. Transport, replay, provenance, and local naming preserve that same finite path, leaving no coordinate from which a measure-theoretic or completed-plane claim could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{KakeyaSet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KakeyaSet}^{\uparrow}$ packet is generated from finite direction-window rows, finite observation windows, segment rows, segment-incidence rows, metric comparison rows, Real and dyadic-interval handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{KakeyaSet}^{\uparrow}}$, seed)

Scope closed: Definition 8985.1, Definition 8985.2, and Definition 8985.3 seed the finite direction-window and segment-incidence carrier for KakeyaSet$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{KakeyaSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure zero, the Kakeya conjecture, Besicovitch set existence, arbitrary direction choice, infinite unions, host subsets of the plane, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked direction-window admission, segment-incidence exactness, metric comparison display, Real and dyadic-interval handoff, transport stability, replay stability, and non-escape for BEDC.Derived.KakeyaSetUp.

Chapter 8986

A Concrete Naming Certificate for $\mathsf{StreamRegularLimitSeal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{StreamRegularLimitSeal}^{\uparrow}$ packet records the finite cofinal diagonal representative used when a regular stream family is handed to Real and Completion consumers. It sits after the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the tail-control route of chapter 2220. The packet names one displayed diagonal-limit handoff; it does not select a host limit, quotient stream, choice schedule, trichotomy decision, or ambient completeness theorem.

Definition 8986.1 (Stream regular limit seal carrier)**.** *A* $\mathsf{StreamRegularLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, W, R, D, I, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row, naming the inspected precision and the completion-facing consumer request.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the requested member stream and the diagonal observation are read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational data obtained from the same finite windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic budget used to compare the requested tail with the diagonal representative.*
5. *I is the cofinal diagonal index row, naming the finite index chosen by the displayed modulus route for this request.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, W, R, D, I have been displayed.*
7. *H records componentwise* hsame *transport for the request, window, readback, tolerance, diagonal index, Real seal, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from the finite request through the stream window, regular readback, dyadic tolerance, cofinal diagonal index, and terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamRegularLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{StreamRegularLimitSeal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, W, R, D, I, E, H, C, P, N$. It has no coordinate for selected limits, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 8986.2 (Stream regular limit seal NameCert obligations)**.** *For every accepted* $\mathsf{StreamRegularLimitSeal}^{\uparrow}$ *carrier*

$$S = (Q, W, R, D, I, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamRegularLimitSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite request,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure, cofinal diagonal index exposure,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8986.1. The packet first exposes the finite request Q, the stream window W, the regular rational readback R, the dyadic tolerance ledger D, and the cofinal diagonal index I.

Read the dependency route in order. The stream window confines the inspected rows to finite observations, the regular readback turns those observations into the regular rational surface, and the dyadic tolerance ledger supplies the displayed comparison budget used by the tail-control route of chapter 2220.

Only after those rows are visible may the packet expose the terminal Real seal E. The transport, replay, provenance, and local naming rows H, C, P, N preserve that same finite route, so the accepted surface is the displayed handoff through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Since the carrier has no row for a selected host limit, quotient stream equality, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof, none of those objects can be exported by the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamRegularLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamRegularLimitSeal}^{\uparrow}$ packet is generated from a finite request row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, cofinal diagonal index, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamRegularLimitSeal}^{\uparrow}}$, seed)

Scope closed: Definition 8986.1 and Definition 8986.2 seed the finite diagonal-limit seal route over request, window, readback, tolerance, cofinal index, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamRegularLimitSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite request routing, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance exposure, cofinal diagonal index routing, Real seal nonselection, transport, replay, provenance, and local naming for BEDC.Derived.StreamRegularLimitSealUp.

Chapter 8987

A Concrete Naming Certificate for EpsilonDeltaModulusSelector$^{\uparrow}$

Following Bishop and Bridges constructive analysis, an epsilon-delta proof of continuity is useful to later real analysis only when its local witness can be read as explicit finite modulus data. The EpsilonDeltaModulusSelector$^{\uparrow}$ packet records that read as a NameCert carrier between the displayed graph and local continuity rows of chapter 130, the modulus consumer interface of chapter 369, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. It names the finite route from a requested output tolerance to a displayed input-radius row, not a host choice of deltas, quotient equality of functions, or an ambient completeness principle.

Definition 8987.1 (Epsilon-delta modulus selector carrier)**.** *An* EpsilonDeltaModulusSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, E, D, W, Q, U, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* ContinuousMap$^{\uparrow}$ *graph and local epsilon-delta continuity row whose source and target observations are already named;*
2. *E is the* DyadicRatCore$^{\uparrow}$ *output-tolerance request row;*
3. *D is the selected* DyadicRatCore$^{\uparrow}$ *input-radius row displayed for the same request;*
4. *W is the finite observation-window row naming the source and target samples inspected by the local proof;*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed source and target rational approximations;*
6. *U is the* UniformModulus$^{\uparrow}$ *consumer row that can be exported only after F, E, D, W, Q are visible;*
7. *R is the* Real$^{\uparrow}$ *seal row reached by the readback and modulus route, with no independent completeness coordinate;*
8. *H records componentwise* hsame *transport for the continuous-map, tolerance, radius, window, readback, uniform-modulus, Real seal, replay, provenance, and naming rows;*
9. *C records displayed* Cont *replay from an output tolerance through the local epsilon-delta row to the selected input radius and then to the uniform-modulus consumer;*

10. *P is the* Pkg *provenance over* $\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice function over all tolerances, quotient equality of maps, a hidden global modulus, trichotomy over reals, or ambient completeness.

Theorem 8987.2 (Epsilon-delta modulus selector NameCert obligations)**.** *For every accepted* $\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}$ *carrier* $S = (F, E, D, W, Q, U, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaModulusSelecto}}$ *surface consists exactly of carrier admission for the displayed continuous-map row, output-tolerance row, selected input-radius row, finite window, regular-rational readback, uniform-modulus row, Real seal, transport, replay, provenance, and local naming rows. The selected radius D must be read from the same local epsilon-delta row F and the same tolerance request E before U is visible. Every consumer read of U or R preserves $F, E, D, W, Q, H, C, P, N$ and exports no host choice of radii, quotient map equality, hidden global modulus, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 8987.1. The accepted packet exposes exactly the rows $F, E, D, W, Q, U, R, H, C, P, N$, so the naming surface is exhausted by those finite coordinates.

Read the analytic rows next. The tolerance request E is displayed as dyadic data, and the selected input radius D is accepted only as the row supplied by the same local epsilon-delta continuity proof F. The finite window W and readback Q record the observations on which the comparison is checked.

The structural rows then preserve the route. Transport H, replay C, provenance P, and the local name N cannot add a choice function, quotient equality, hidden modulus, trichotomy, or completeness coordinate because none is present in the carrier. Thus the NameCert obligations are exactly the displayed rows and their consumer order. □

Theorem 8987.3 (Epsilon-delta selector ContinuousMap route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In carrier notation, the consumer must read the local continuous-map row F, the requested tolerance E, the selected radius D, the finite window W, and the readback row Q before it can consume the uniform-modulus row U or the Real seal R. The route is a displayed epsilon-delta modulus selection surface, not a host modulus function, quotient equality of functions, selected global delta table, trichotomy, or ambient real completeness.

Proof. Begin with the $\mathsf{ContinuousMap}^{\uparrow}$ row. It supplies the graph and local epsilon-delta continuity data as the displayed row F, while the dyadic request E names the target tolerance that the consumer wants to spend.

The selector row D is then read only through that same F, E pair. The finite window W and regular rational readback Q expose the source and target observations needed to check the local estimate before any modulus row is exported.

Finally read U and R through the structural rows. The uniform-modulus consumer receives the route preserved by H, C, P, N, and the Real seal is reached only after the readback and modulus rows are displayed. Since the packet has no coordinate for host modulus functions, quotient equality, global delta tables, trichotomy, or ambient completeness, none of those readings escapes the displayed route. □

Falsifiable prediction: An accepted $\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}$ packet is refuted if a UniformModulus consumer reads the exported modulus without displaying the ContinuousMap row, dyadic tolerance request, selected dyadic radius, finite observation window, RegSeqRat readback, transport, replay, provenance, and local NameCert rows.

Independence witness: continuousmap, uniformmodulus, dyadicratcore, regseqrat, real: EpsilonDeltaModulusSel is not bijective to any listed sibling because it jointly stores the local epsilon-delta row, requested tolerance, selected input radius, finite readback window, uniform-modulus handoff, Real seal, and structural support in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}}$.

Constructive story: An $\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}$ packet is generated from a ContinuousMap local epsilon-delta row, dyadic output-tolerance requests, selected dyadic input-radius rows, finite observation windows, RegSeqRat readback rows, a UniformModulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8987.1, Definition 8987.2, and Definition 8987.3 seed the explicit epsilon-delta modulus selector over ContinuousMap, UniformModulus, DyadicRatCore, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice of radii, quotient equality of maps, selected global delta tables, trichotomy, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.EpsilonDeltaModulusSelec

Chapter 8988

A Concrete Naming Certificate for $\mathsf{CarlesonBox}^{\uparrow}$

Following Carleson and the Hardy-space boundary geometry tradition, the $\mathsf{CarlesonBox}^{\uparrow}$ packet records the finite boundary arc and radial window data needed before a Hardy-boundary consumer can inspect a Carleson box. It sits after the Hardy-space carrier of chapter 5108, the dyadic tent-function window surface of chapter 8351, the finite content budget of chapter 3765, and the measure-facing carrier of chapter 92. The packet names only displayed arcs, dyadic radial intervals, tent windows, budget rows, Hardy handoff data, transport, replay, provenance, and local naming; it does not assert the Carleson embedding theorem, selected boundary limits, completed measure theory, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8988.1 (Carleson box carrier)**.** *A* $\mathsf{CarlesonBox}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, R, T, M, H, E, S, C, P, N)$$

with the following displayed rows:

1. *A is the boundary-arc row. It stores the named center, dyadic angular endpoints, orientation, and finite endpoint inspections of the boundary circle arc.*
2. *R is the radial dyadic-window row. It records the inner and outer radial cutoffs attached to the same arc, with all radial tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *T is the* $\mathsf{DyadicTentFunction}^{\uparrow}$ *window row whose tent footprint contains the displayed arc and radial window.*
4. *M is the finite measure or* $\mathsf{HausdorffContent}^{\uparrow}$ *budget row for the same window. It records only the displayed cover, mass, or content upper-bound data used by the boundary request.*
5. *H is the* $\mathsf{HardySpace}^{\uparrow}$ *handoff row. It stores the Hardy-boundary source whose local read is allowed to inspect the box after A, R, T, M are visible.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the radial, content, and Hardy-boundary scalar reads.*
7. *S records componentwise* hsame *transport for A, R, T, M, H, E.*
8. *C records displayed* Cont *replay from a boundary request through the arc, radial window, tent window, budget, Hardy handoff, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CarlesonBox}^{\uparrow}$, $\mathsf{HardySpace}^{\uparrow}$, $\mathsf{DyadicTentFunction}^{\uparrow}$, $\mathsf{HausdorffContent}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CarlesonBox}^\uparrow}$ *row.*
The classifier compares carriers only through $A, R, T, M, H, E, S, C, P, N$. It has no coordinate for a selected nontangential boundary limit, a host Carleson measure, a completed measure algebra, quotient function equality, a global embedding constant, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 8988.2 (Carleson box NameCert obligations)**.** *For every accepted* $\mathsf{CarlesonBox}^\uparrow$ *carrier $B = (A, R, T, M, H, E, S, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CarlesonBox}^\uparrow}$ *surface consists exactly of boundary-arc admission, radial dyadic-window admission, dyadic tent-window visibility, finite measure or Hausdorff-content budget visibility, Hardy-space handoff visibility, Real sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. A Hardy-boundary consumer may read the box only after A, R, T, M, H are all public in the same packet.*

Proof. Project the carrier of Definition 8988.1. The displayed tuple exposes exactly the boundary arc, radial window, tent window, budget row, Hardy handoff, Real seal, transport, replay, provenance, and local naming row.

Read the geometric rows first. The arc A and radial window R determine the finite boundary box, while T records the dyadic tent window that contains the same footprint.

The budget and Hardy rows are downstream of that footprint. The row M supplies only the displayed finite measure or content budget, and H supplies only the Hardy-space boundary source attached to that same window.

Finally the structural rows preserve the route. Since S, C, P, N add only transport, replay, provenance, and naming, the packet has no coordinate for selected boundary limits, host Carleson measures, completed measure algebras, quotient function equality, embedding constants, countable choice, or ambient completeness. □

Theorem 8988.3 (Carleson box Hardy-boundary route)**.** *Every* $\mathsf{HardySpace}^\uparrow$ *boundary read that consumes an accepted* $\mathsf{CarlesonBox}^\uparrow$ *packet factors through the finite route*

$$A, R \longmapsto \mathsf{DyadicTentFunction}^\uparrow \longmapsto \mathsf{HausdorffContent}^\uparrow/\mathsf{Measure}^\uparrow \longmapsto \mathsf{HardySpace}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

In carrier notation, the consumer reads the boundary arc A, radial dyadic window R, tent window T, budget row M, Hardy handoff H, Real seal E, and the structural rows S, C, P, N. The route is a finite Hardy-boundary geometry handoff, not a Carleson embedding theorem, selected boundary limit, completed measure theory, quotient function equality, or ambient completeness claim.

Proof. Start with the geometric footprint. The boundary arc row A and radial window row R name the box as finite dyadic data, so the packet never begins with an ambient boundary set or completed measure object.

Next read the tent and budget rows. The $\mathsf{DyadicTentFunction}^\uparrow$ row T attaches the window to a finite tent footprint, and the row M supplies only the displayed $\mathsf{HausdorffContent}^\uparrow$ or $\mathsf{Measure}^\uparrow$ budget for that footprint.

The Hardy-facing read then passes through H and E. The Hardy handoff is visible only after the footprint and budget rows have been displayed, and the Real seal records the scalar endpoint of that same route.

Transport, replay, provenance, and local naming preserve the displayed packet. Because no carrier row stores a Carleson embedding theorem, selected boundary limit, completed measure theory, quotient equality, or ambient completeness principle, no Hardy-boundary consumer can read those endpoints from the box. □

Falsifiable prediction: An accepted $\mathsf{CarlesonBox}^{\uparrow}$ packet is refuted if a Hardy-boundary consumer reads a box without displaying the boundary arc, radial dyadic window, tent-function window, finite budget, Hardy handoff, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: hardy-space, dyadic-tent-function, hausdorff-content, measure: $\mathsf{CarlesonBox}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores the boundary arc, radial dyadic window, tent footprint, finite budget, Hardy handoff, Real seal, and structural support in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CarlesonBox}^{\uparrow}}$.

Constructive story: A $\mathsf{CarlesonBox}^{\uparrow}$ packet is generated from a boundary arc, radial dyadic window, dyadic tent-function window, finite Hausdorff-content or measure budget, HardySpace handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CarlesonBox}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8988.1, Definition 8988.2, and Definition 8988.3 seed the finite Hardy-boundary geometry packet over boundary arcs, radial windows, tent windows, content or measure budgets, Hardy handoff, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CarlesonBox}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the Carleson embedding theorem, selected boundary limits, completed measure theory, quotient function equality, global embedding constants, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.CarlesonBoxUp.

Chapter 8989

A Concrete Naming Certificate for $\mathsf{LocatedUpperLowerPair}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ packet records a located real by paired lower and upper rational enclosures whose gap is visible at each requested tolerance. It sits between the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located real interface of chapter 1933. The packet supplies a finite L10 Real and Completion interface only; it does not assert quotient-cut equality, a chosen completed representative, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 8989.1 (Located upper-lower pair carrier)**.** *A* $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, E, H, C, P, N)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure row, carrying the displayed lower endpoint, upper endpoint, dyadic tolerance, and gap comparison for the requested precision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the consumer requests and observes the enclosure stage.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the same finite window into regular rational lower and upper observations.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after D, S, R have exposed the finite enclosure data.*
5. *H records componentwise* hsame *transport for enclosure, window, readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a requested tolerance to the matching enclosure window and then to the terminal seal.*
7. *P is the* Pkg *provenance over* $\mathsf{LocatedUpperLowerPair}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperLowerPair}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient cut, selected host real, omniscient comparison oracle, countable choice schedule, or ambient completeness principle.

Lean4 (statement) `BEDC.Derived.LocatedUpperLowerPairUp`

Theorem 8989.2 (Located upper-lower pair NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ *carrier* $L = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperLowerPair}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic enclosure admission, StreamName window visibility, RegSeqRat lower-upper readback, Real seal nonescape, componentwise transport, displayed continuation replay, package provenance, and local naming. No NameCert obligation may replace these rows by quotient-cut equality, selected representative equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 8989.1. The rows D, S, R, E expose the dyadic enclosure, finite window, regular rational readback, and terminal seal in that order, while H, C, P, N expose the transport, replay, provenance, and local naming rows.

The classifier is stable because every accepted comparison reads the same tolerance window through D and the same finite observation window through S. Componentwise hsame transport in H preserves the displayed lower endpoint, upper endpoint, tolerance, and gap comparison before the readback reaches R.

Finally, the terminal seal E is downstream of D, S, R. Thus the NameCert surface exports only the finite located enclosure route and cannot supply quotient-cut equality, selected host representatives, trichotomy, countable choice, or ambient completeness. □

Theorem 8989.3 (Located upper-lower pair completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ *carrier must read*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the displayed rows D, S, R, E *before treating the pair as a Real-facing located name.*

Proof. Start with the dyadic row D. It is the only row that displays the lower endpoint, upper endpoint, tolerance, and gap comparison for the request.

Open the finite observation row S and the regular rational readback row R. These rows bind the same request window to the lower-upper rational observations available to the consumer.

The Real-facing read then reaches E only after D, S, R have been displayed. Componentwise transport, continuation replay, package provenance, and local naming keep the handoff inside the accepted packet, so no quotient cut, selected representative, trichotomy, countable choice, or ambient completeness is introduced. □

Falsifiable prediction: An accepted $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading a DyadicRatCore enclosure row, a StreamName finite window, and a RegSeqRat lower-upper readback row refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-real: no listed sibling is bijective with $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ because this packet jointly binds lower and upper enclosure rows, tolerance windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUpperLowerPair}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedUpperLowerPair}^{\uparrow}$ packet is generated from a DyadicRatCore lower-upper enclosure row, a StreamName finite-window row, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedUpperLowerPair}^{\uparrow}}$, seed)

Scope closed: Definition 8989.1, the NameCert obligation sketch, completion handoff, falsifiability, and independence rows seed the finite Bishop-style located upper-lower pair route over DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedUpperLowerPair}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-cut equality, selected representatives, trichotomy, countable choice, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, enclosure exactness, window visibility, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUpperLowerPairUp.

Chapter 8990

A Concrete Naming Certificate for $\mathsf{BoundedCauchyFilter}^{\uparrow}$

Following Kelley and Bishop–Bridges on Cauchy filters and bounded filters, the $\mathsf{BoundedCauchyFilter}^{\uparrow}$ packet records the bounded source surface used by the L10 completion corridor. It sits between the Cauchy-filter carrier of chapter 333, the bounded base face of chapter 1601, the dyadic tolerance ledger of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion handoff of chapter 4054. The packet exposes bounded Cauchy-filter source data only as finite NameCert rows; it is not a selected limit, quotient filter equality, countable-choice selector, or ambient completeness principle.

Definition 8990.1 (Bounded Cauchy filter carrier)**.** *A* $\mathsf{BoundedCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (F, B, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose requests and meets are read as displayed Cauchy-filter data.*
2. *B is the bounded* $\mathsf{FilterBase}^{\uparrow}$ *face naming the finite base requests and the bound certificate used by every source read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to bounded filter requests.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened from the bounded Cauchy-filter source.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows and dyadic tolerances.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after F, B_0, D, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for the Cauchy-filter source, bounded base face, dyadic tolerance, stream window, regular rational readback, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay for the same bounded source route.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilter}^{\uparrow}}$ *row.*

The carrier exports no coordinate for selected limits, quotient filter equality, countable choice, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.BoundedCauchyFilterUp.TasteGate.BoundedCauchyFilterUp`

Theorem 8990.2 (Bounded Cauchy filter NameCert obligations)**.** *For an accepted carrier* $X = (F, B, D, S, Q, R, H, C, P, N)$ *of Definition 8990.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{CauchyFilter}^{\uparrow}$ *source admission, bounded* $\mathsf{FilterBase}^{\uparrow}$ *face admission, dyadic tolerance accounting, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BoundedCauchyFilterUp.BoundedCauchyFilterCarrier namecert obligations`

Proof. Project the accepted carrier. The Cauchy-filter source row is F, the bounded base face is B, and the dyadic tolerance ledger is D.

The route can open a finite window only after those source rows have been displayed. The window row S then determines the regular rational readback row Q available to Real-facing source consumers.

The seal row R is downstream of F, B_0, D, S, Q. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so the listed rows are precisely the local NameCert obligations. □

Theorem 8990.3 (Bounded Cauchy filter L10 handoff)**.** *Every L10 completion consumer of an accepted* $\mathsf{BoundedCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BoundedCauch}$$

The handoff exports only the displayed source, bounded-base, tolerance, window, readback, seal, transport, replay, provenance, and naming rows.

Proof. Begin with an L10 completion consumer of X. By Definition 8990.2, the consumer cannot reach the completion-facing source route until it has read the Cauchy-filter source F, the bounded base face B, and the dyadic tolerance ledger D.

The finite readback is then forced through S, Q, R. These rows are the displayed stream schedule, regular rational readback, and Real seal, so the bounded completion packet receives only the bounded Cauchy-filter source route.

The structural rows H, C, P, N replay and name that same route. Since the carrier contains no selected limit, quotient-filter equality, host real equality, countable-choice selector, or ambient completeness witness, no other L10 handoff is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchyFilter}^{\uparrow}$ packet is generated from a CauchyFilter source row, bounded FilterBase face, DyadicRatCore tolerances, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 8990.1, Definition 8990.2, and Definition 8990.3 seed the bounded Cauchy-filter source corridor for L10 completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient filter equality, host real equality, countable choice, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source compatibility, dyadic tolerance accounting, finite-window exposure, regular readback, Real seal nonescape, package provenance, and local naming for the displayed packet.

Chapter 8991

A Concrete Naming Certificate for $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ packet records the finite located intermediate-value route for a continuous map on a located real interval whose endpoint reads display a sign change. It sits after the dyadic tolerance rows of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the map carrier of chapter 130, the located interval boundary of chapter 1709, and the retained sign-change route of chapter 7008. The chapter names the Real and Completion boundary consumed by constructive IVT arguments; it does not assert classical completeness, trichotomy, quotient equality of real names, selected roots, countable choice, or an ambient host intermediate-value theorem.

Definition 8991.1 (ConstructiveLocatedIVT carrier)**.** *A* $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, S, B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, carrying the endpoint names, radius data, locator reads, locatedness moduli, and bracket boundary consumed by the interval argument.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row, recording the finite continuity and modulus surface through which endpoint and midpoint image reads are requested.*
3. *S is the sign-change ledger, recording only displayed endpoint and retained-window sign reads for the map row F over the interval row I.*
4. *B is the* $\mathsf{SignChangeBisection}^{\uparrow}$ *row, read through chapter 7008, which supplies the retained branch route from the displayed sign-change data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic modulus and tolerance row for midpoint, endpoint, and image-window requests.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row through which the bisection and image reads are inspected.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed windows and dyadic tolerance ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, F, S, B, D, W, R have been displayed.*

9. *H records componentwise* hsame *transport for the interval, map, sign ledger, bisection, dyadic, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a constructive located IVT request through the same rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SignChangeBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedIVT}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, S, B, D, W, R, E, H, C, P, N$. It has no coordinate for host real trichotomy, quotient equality of real names, countable choice, selected zeroes, classical completeness, or an ambient intermediate-value theorem.

Theorem 8991.2 (ConstructiveLocatedIVT NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ *carrier $K = (I, F, S, B, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedIVT}^{\uparrow}}$ *surface consists exactly of located-interval admission, continuous-map admission, displayed sign-change ledger admission, sign-change bisection routing, dyadic tolerance and modulus admission, finite StreamName window admission, RegSeqRat readback, Real seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the displayed rows.*

Proof. Read the carrier of Definition 8991.1. Each accepted packet exposes exactly the rows $I, F, S, B, D, W, R, E, H, C, P, N$, and the classifier is declared over those rows only.

The cited sibling chapters supply the row meanings. Located intervals provide endpoint and bracket data, continuous maps provide the image-read surface, sign-change bisection provides the retained branch route, and the dyadic, StreamName, RegSeqRat, and Real chapters supply the finite tolerance, observation, rational readback, and terminal seal layers.

The structural rows then preserve the same surface under transport, replay, provenance, and local naming. Since the carrier excludes trichotomy, quotient real equality, countable choice, selected zeroes, classical completeness, and ambient IVT data, the listed rows exhaust the local NameCert obligations. □

Theorem 8991.3 (ConstructiveLocatedIVT sign-change route)**.** *Every accepted* $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ *packet routes an IVT-facing sign-change read through*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto S \longmapsto \mathsf{SignChangeBisection}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \mapsto$$

before any Real or Completion consumer may inspect the terminal seal. The route exports the finite displayed boundary of the constructive IVT argument, not a classical completeness principle or a selected root.

Proof. Start with the source rows I and F. The interval row fixes the located endpoint boundary, and the continuous-map row fixes the finite image-read surface over that boundary.

The sign ledger S is then consumed by the sign-change bisection row B. By chapter 7008, the retained branch is a displayed bisection route, not a selected root or a host theorem.

Finally the branch route reaches Real and Completion readers only through D, W, R, E. The dyadic tolerance ledger supplies the finite budgets, the StreamName row supplies visible windows, the RegSeqRat row supplies regular rational readback, and the Real row supplies the terminal seal. Transport, replay, provenance, and local naming do not add another path. □

Theorem 8991.4 (ConstructiveLocatedIVT Real-seal nonescape)**.** *Let $K = (I, F, S, B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ *carrier. A consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal E must have displayed the located interval I, continuous map F, sign ledger S, sign-change bisection row B, dyadic tolerance row D, StreamName window row W, RegSeqRat readback row R, and structural rows H, C, P, N. Hence the seal cannot be used as a shortcut to quotient real equality, trichotomy, countable choice, a selected zero, classical completeness, or an ambient IVT theorem.*

Proof. Project the Real seal coordinate from the carrier. The row E is declared terminal and is available only after the preceding interval, map, sign, bisection, dyadic, window, and readback rows have been displayed.

Use Definition 8991.3. It orders the only IVT-facing path to the seal through the located interval, continuous map, sign-change bisection, dyadic tolerance, finite window, and regular rational readback rows.

The transport, replay, provenance, and local naming rows preserve that route under accepted presentation changes. Because the classifier has no coordinate for quotient real equality, trichotomy, countable choice, selected zeroes, classical completeness, or ambient IVT data, the Real seal cannot escape the displayed carrier boundary. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ Real-facing read displays a located interval, continuous map, sign-change ledger, sign-change bisection route, dyadic tolerance row, StreamName window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: located-real-interval, continuousmap, sign-change-bisection, dyadicratcore, streamname, regseqrat, real: none of these siblings contains the full located interval, continuousmap, sign-change, bisection, dyadic-modulus, stream-window, regular-readback, Real-seal, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveLocatedIVT}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveLocatedIVT}^{\uparrow}$ packet is generated from a $\mathsf{LocatedRealInterval}^{\uparrow}$ source row, a $\mathsf{ContinuousMap}^{\uparrow}$ row, a sign-change ledger, a $\mathsf{SignChangeBisection}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic modulus row, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveLocatedIVT}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8991.1, Definition 8991.2, Definition 8991.3, and Definition 8991.4 seed the constructive located IVT boundary over located intervals, continuous maps, sign-change bisection, dyadic tolerances, StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveLocatedIVT}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical completeness, trichotomy, quotient real equality, selected zeroes, countable choice, ambient IVT, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, sign-change routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveLocatedIVTUp.

Chapter 8992

A Concrete Naming Certificate for $\mathsf{UniformContractionRealCompletion}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive completion, the $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ packet records the finite route by which a uniformly contractive iteration over a displayed complete metric source is read as a regular rational completion handoff. It sits beside the Picard orbit of chapter 2359, the contraction source of chapter 1418, the Banach contraction sequence of chapter 8253, the interval contraction route of chapter 5867, the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ completion rows of chapter 126 and chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite completion-facing iteration surface; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, a selected fixed point, quotient stream equality, host contraction equality, or countable choice.

Definition 8992.1 (Uniform contraction real-completion carrier)**.** *A* $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, L, P, W, D, R, S, E, H, C, Q, N)$$

with the following displayed rows:

1. *M is the complete metric source row, exposing the source carrier, metric comparison, and endpoint-admission boundary.*
2. *L is the contraction modulus row, recording one uniform shrinkage bound for the displayed self-map.*
3. *P is the Picard window schedule row, naming the finite iterates inspected by the consumer.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window read of those iterates.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare adjacent windows and requested error bounds.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback generated from P, W, D.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the regular rational readback is visible.*
8. *E is the endpoint handoff row for fixed-point consumers; it records the completion-facing route without selecting a fixed point.*
9. *H records componentwise* hsame *transport for M, L, P, W, D, R, S, E and the local certificate rows.*

10. *C records displayed* Cont *replay from the contraction source through the Picard schedule, dyadic tolerance ledger, stream window, regular readback, Real seal, and endpoint handoff.*
11. *Q is the* Pkg *provenance over* $\mathsf{UniformContractionRealCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{BanachContractionSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContractionRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient real-completeness principle, selected limits, quotient stream equality, host fixed-point equality, function extensionality, or countable choice.

Lean4 (definition) `BEDC.Derived.UniformContractionRealCompletionUp.UniformContractionRealCompletionUp`

Theorem 8992.2 (Uniform contraction real-completion NameCert obligations)**.** *For every accepted* $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ *carrier* $U = (M, L, P, W, D, R, S, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformContractionRealCompletion}^{\uparrow}}$ *surface consists exactly of complete metric source visibility, contraction modulus visibility, Picard window schedule visibility, dyadic tolerance exactness,* $\mathsf{StreamName}^{\uparrow}$ *window visibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback visibility,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, endpoint handoff nonselection, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 8992.1.*

Proof. Project the carrier. The source and iteration rows are M, L, P, the completion-facing rows are W, D, R, S, E, and the structural rows are H, C, Q, N.

Read M and L first. The complete metric source supplies the admissible endpoint boundary, while the contraction modulus row supplies the uniform shrinkage budget for every displayed Picard window.

Read P, W, D, R, S next. The schedule names only finite windows, the dyadic ledger supplies the tolerances, the stream row records the inspected iterates, the regular rational row reads those iterates as Cauchy data, and the Real seal is downstream of that readback.

Finally read E, H, C, Q, N. The endpoint handoff is a consumer boundary, and the structural rows preserve the same packet. Since none of the excluded ambient or choice objects is present in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 8992.3 (Uniform contraction real-completion Cauchy induction)**.** *Every Cauchy read of an accepted* $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ *packet is obtained by induction on the displayed Picard window schedule P. The base window reads the complete metric source row M, the successor step spends the contraction modulus row L and dyadic tolerance row D, and the resulting finite windows pass through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The induction exports regular rational completion data only; it does not export selected limits, quotient stream equality, countable choice, host fixed-point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use induction over the finite schedule row P. In the base case the source row M supplies the initial admitted metric point and the completion-facing boundary.

In the successor step, apply the contraction modulus row L to the displayed self-map read and spend the tolerance ledger D for the next window. This keeps every residual comparison attached to the same finite schedule rather than to a hidden infinite orbit.

The rows W and R then repack the finite windows as a stream name and regular rational readback. The Real seal S is reached only through that readback, so the induction supplies Cauchy data without adding any selected endpoint or ambient completeness coordinate. □

Theorem 8992.4 (Uniform contraction real-completion fixed-point handoff)**.** *Every fixed-point consumer reading an accepted* $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ *packet must first expose the route*

$$M, L, P, W, D, R, S, H, C, Q, N.$$

Only after those rows are visible may the consumer use the endpoint handoff row E as input to a separate fixed-point certificate. The handoff supplies a finite contraction-iteration completion route; it is not itself a fixed-point theorem, a uniqueness theorem, a host function equality, a selected limit, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface. It makes the complete metric source, contraction modulus, Picard schedule, dyadic tolerance ledger, stream windows, regular readback, and Real seal visible before any endpoint-facing consumer can inspect the packet.

Apply the Cauchy induction theorem. It shows that R and S are downstream of the finite schedule and tolerance rows, so the endpoint handoff cannot be detached from the displayed contraction route.

The structural rows H, C, Q, N preserve the same route for consumers. Thus E can be passed to a later fixed-point certificate, but it carries no fixed-point existence, uniqueness, selected-limit, quotient, choice, or ambient-completeness claim by itself. □

Falsifiable prediction: An accepted $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ consumer is refuted if it reads a RegSeqRat handoff, Real seal, or fixed-point endpoint handoff without displaying the complete metric source, contraction modulus, Picard window schedule, StreamName windows, DyadicRatCore tolerance ledger, transport, replay, provenance, and local NameCert rows.

Independence witness: contractionmapping, picard-iteration, banach-contraction-sequence, bishop-interval-contraction, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the joint packet containing complete metric source, uniform contraction modulus, Picard window schedule, dyadic tolerance ledger, StreamName readback, RegSeqRat readback, Real seal, endpoint handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContractionRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformContractionRealCompletion}^{\uparrow}$ packet is generated from a complete metric source row, a uniform contraction modulus row, a Picard window schedule, StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, endpoint handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformContractionRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8992.1, Definition 8992.2, Definition 8992.3, and Definition 8992.4 seed the finite contraction-iteration completion route without asserting a completed fixed point.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformContractionRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient Real completeness, selected limits, quotient stream equality, host fixed-point equality, uniqueness, function extensionality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-modulus visibility, Picard-window induction, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, endpoint handoff nonselection, transport, replay, provenance, and local naming for BEDC.Derived.UniformContractionRealCompletionUp.

Chapter 8993

A Concrete Naming Certificate for $\mathsf{ArtinianModule}^{\uparrow}$

Following Bourbaki and Atiyah–Macdonald on Artinian modules, the $\mathsf{ArtinianModule}^{\uparrow}$ packet records descending submodule-chain stabilization over a displayed module source. It sits beside the module carrier of chapter 46, the scalar ring source of chapter 43, and the ascending-chain sibling $\mathsf{NoetherianModule}^{\uparrow}$ of chapter 3648. The packet supplies finite descending-chain evidence only; it does not assert a composition series, a length theorem, quotient-module classification, selected complements, or a host Artinian-module principle.

8993.1 Carrier definition

Definition 8993.1 (Artinian module carrier)**.** *An* $\mathsf{ArtinianModule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{ArtMod}(M, R, D, S, H, C, P, N).$$

The row M is the $\mathsf{Module}^{\uparrow}$ *source surface. The row R is the scalar* $\mathsf{Ring}^{\uparrow}$ *source read from the module certificate. The row D is the descending-submodule-chain ledger: it lists the displayed submodule rows, their inclusion comparisons, and the finite read order in which a consumer may inspect them. The row S is the stabilization witness, assigning to every displayed descending chain in D a finite stage after which all later submodule endpoints classify with that stage. The row H is componentwise* hsame *transport for the module, scalar, descending-chain, and stabilization rows. The row C is displayed* Cont *replay for the descending-chain read and stabilization handoff. The row P is* Pkg *provenance over* $\mathsf{ArtinianModule}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows. The row N is the local* $\mathsf{NameCert}_{\mathsf{ArtinianModule}^{\uparrow}}$ *row.*

Theorem 8993.2 (Artinian module NameCert obligations)**.** *For every accepted* $\mathsf{ArtinianModule}^{\uparrow}$ *carrier* $\mathsf{ArtMod}(M, R, D, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArtinianModule}^{\uparrow}}$ *surface consists exactly of carrier admission for the eight displayed rows, source compatibility with the* $\mathsf{Module}^{\uparrow}$ *and* $\mathsf{Ring}^{\uparrow}$ *rows, descending-chain ledger admission through D, finite stabilization access through S, componentwise classifier transport through H, displayed continuation replay through C, provenance non-escape through P, and local naming through N.*

Proof. Project the eight rows of Definition 8993.1. The module row M fixes the additive carrier, scalar action, and module classifier before any descending submodule chain is inspected.

The scalar row R binds the chain to the same ring action already visible in the $\mathsf{Module}^{\uparrow}$ certificate. The descending ledger D then supplies only displayed submodule rows and inclusion comparisons, so the chain is not replaced by a quotient-module classifier or a selected complement.

Read the stabilization witness S over the displayed chain in D. Componentwise transport H moves the module, scalar, chain, and stabilization rows together through hsame, while C replays only the listed continuation data.

Finally, P and N repack the same finite inventory as the local naming certificate. Since no composition-series selector, length theorem, quotient classification, selected complement, or host Artinian principle occurs in the carrier, none can occur among the local obligations. □

Theorem 8993.3 (Artinian descending-chain route)**.** *Let* $\mathsf{ArtMod}(M, R, D, S, H, C, P, N)$ *be an accepted* $\mathsf{ArtinianModule}^{\uparrow}$ *carrier. Every downstream module consumer that reads descending-chain stabilization must first expose the* $\mathsf{Module}^{\uparrow}$ *source* M*, the scalar* $\mathsf{Ring}^{\uparrow}$ *row* R*, and the descending-submodule-chain ledger* D*, and only then may read the finite stabilization witness* S*. The route is the descending-chain analogue of the* $\mathsf{NoetherianModule}^{\uparrow}$ *ascending-chain packet, but it exports no ascending-chain theorem, composition-series theorem, selected quotient, or host Artinian-module result.*

Proof. Begin with the source rows M and R. They determine the module classifier and scalar action, so the later descending-chain read is attached to the same algebraic object used by chapter 46.

The ledger D is read next. It contains the finite list of displayed submodule rows and inclusion comparisons, and it is the only row from which a descending chain can be inspected.

Apply S to that displayed ledger. The resulting stage is finite and lives inside the same module classifier. Transport through H and replay through C preserve that ordered route, while P and N expose only the provenance and naming rows of the same packet.

Thus a consumer receives descending-chain stabilization as finite BHist data. The sibling comparison with chapter 3648 is structural only: the present packet uses descending chains, not the ascending-chain or finite-generator rows of $\mathsf{NoetherianModule}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArtinianModule}^{\uparrow}}$.

Constructive story: An $\mathsf{ArtinianModule}^{\uparrow}$ packet is generated from a ModuleUp source, a scalar RingUp source, a descending-submodule-chain ledger, a finite stabilization witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArtinianModule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8993.1, Definition 8993.2, and Definition 8993.3 seed the finite descending-submodule-chain route through ModuleUp, RingUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArtinianModule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close composition series, length theory, quotient-module classification, selected complements, host Artinian-module theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, descending-chain ledger admission, finite stabilization access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.ArtinianModuleUp.

Chapter 8994

A Concrete Naming Certificate for $\mathsf{FinitePrefixCompactness}^{\uparrow}$

Following Bishop and Troelstra on fan-style compactness, the $\mathsf{FinitePrefixCompactness}^{\uparrow}$ packet records the finite prefix evidence used by compact-uniform-continuity consumers before any real-valued modulus is exported. It sits after the finite bar packet of chapter 5863, the binary tree surface of chapter 1475, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names finite prefix compactness only; it does not assert open-cover compactness, choice-selected branches, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

8994.1 Carrier definition

Definition 8994.1 (Finite prefix compactness carrier)**.** *A* $\mathsf{FinitePrefixCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{FinPref}(F, K, S, Q, R, E, H, C, P, N).$$

The row F is the $\mathsf{FiniteFanTheorem}^{\uparrow}$ *bounded-bar source. The row K is the* $\mathsf{CantorSpace}^{\uparrow}$ *binary prefix tree read by the same fan window. The row S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule. The row Q is the finite prefix-cover ledger: it lists the retained prefixes, their depths, and the cover decision read at each retained prefix. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the prefix packet is consumed by rational-modulus or regular-Cauchy routes. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for downstream compact-uniform-continuity consumers. The row H is componentwise* hsame *transport for the fan, Cantor, stream, prefix-cover, regular-rational, and real-seal rows. The row C is displayed* Cont *replay for finite prefix traversal, regular-rational readback, and real-seal handoff. The row P is* Pkg *provenance over* $\mathsf{FinitePrefixCompactness}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows. The row N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixCompactness}^{\uparrow}}$ *row.*

Theorem 8994.2 (Finite prefix compactness NameCert obligations)**.** *For every accepted* $\mathsf{FinitePrefixCompactness}^{\uparrow}$ *carrier* $\mathsf{FinPref}(F, K, S, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, source compatibility with* $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$*, finite prefix-cover admission through Q, regular-rational readback through R, terminal real sealing through E, componentwise classifier transport through H, displayed continuation replay through C, provenance non-escape through P, and local naming through N.*

Proof. Project the ten rows of Definition 8994.1. The fan row F, Cantor row K, and stream row S determine the branch-facing finite prefix surface before any rational or real read is available.

The prefix-cover ledger Q is then read over those displayed prefixes. It stores only retained prefix names, their depths, and local cover decisions, so it cannot supply an arbitrary open cover or a choice-selected branch.

Rows R and E expose the analytic handoff. The regular-rational row R is read from the finite prefix packet, and the terminal $\mathsf{Real}^{\uparrow}$ row E is available only after that readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no row stores open-cover compactness, a selected branch, quotient stream equality, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 8994.3 (Finite fan handoff)**.** *Let* $\mathsf{FinPref}(F, K, S, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FinitePrefixCompactness}^{\uparrow}$ *carrier. Every compact-uniform finite-prefix consumer must read the* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *row* F*, the* $\mathsf{CantorSpace}^{\uparrow}$ *row* K*, and the* $\mathsf{StreamName}^{\uparrow}$ *row* S *before inspecting the finite prefix-cover ledger* Q*; it must then pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is exported. Thus the compact-uniform-continuity branch receives a finite prefix packet rather than an open-cover compactness theorem or a choice-selected branch.*

Proof. Start with the fan source F. It supplies bounded-bar evidence in the form accepted by chapter 5863, and it fixes the finite prefix discipline for the packet.

The Cantor and stream rows K and S make the actual prefix surface visible. The prefix-cover ledger Q is indexed by those displayed prefixes, so a consumer cannot inspect a cover decision outside the finite branch window.

The analytic handoff is ordered by R and E. Regular-rational readback in R records the finite rational data consumed by modulus routes, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of that readback.

Transport and replay through H and C preserve this order, while P and N expose only provenance and local naming. The handoff therefore factors through finite fan, Cantor, stream, RegSeqRat, and Real rows, with no open-cover or choice-selected branch coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePrefixCompactness}^{\uparrow}$ packet is generated from FiniteFanTheorem, CantorSpace, StreamName, finite prefix-cover, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8994.1, Definition 8994.2, and Definition 8994.3 seed the finite prefix compactness route through FiniteFanTheoremUp, CantorSpaceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix-cover exactness, fan handoff order, RegSeqRat readback, RealUp seal non-escape, componentwise transport, provenance, and local NameCert exactness for BEDC.Derived.FinitePrefixCompactnessUp.

Chapter 8995

A Concrete Naming Certificate for $\mathsf{PrimeParityPhase}^{\uparrow}$

The $\mathsf{PrimeParityPhase}^{\uparrow}$ packet realizes the projection dictionary of section 9413.37 as one finite NameCert surface. It binds prime-charge rows, parity and unit-boundary rows, phase-routing rows, height-ledger rows, triadic closure-defect rows, transport, replay, provenance, and local naming before any phase or energy consumer reads the packet. The chapter names the finite dictionary surface; it does not prove RH, analytic continuation, Lee–Yang zeros, host complex equality, or an ambient spectral classification.

Definition 8995.1 (Prime parity phase carrier)**.** *A* $\mathsf{PrimeParityPhase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Q, U, F, H, D, T, C, V, N)$$

with the following displayed rows:

1. *Q is the prime-charge row, displaying the finite prime register, height values, and charge ledger read from the source dictionary.*
2. *U is the parity and unit-boundary row, recording the sign boundary, unit-circle boundary, and the refusal of hidden positive real scale inside a compact phase fibre.*
3. *F is the phase-routing row, displaying the finite phase successor, winding, and solenoid-facing route used by the phase consumer.*
4. *H is the height-ledger row, tying the prime-charge heights to the routed phase and energy-facing reads.*
5. *D is the triadic closure-defect row, exposing only the finite defect data visible through the prime, parity, phase, and height rows.*
6. *T records componentwise* hsame *transport for prime charge, parity, unit boundary, phase route, height ledger, defect, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from the dictionary request through Q, U, F, H, D, T, V, N.*
8. *V is the* Pkg *provenance over* $\mathsf{PrimeParityPhase}^{\uparrow}$*, the prime-charge dictionary of section 9413.37, chapter 291, chapter 126,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimeParityPhase}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $Q, U, F, H, D, T, C, V, N$. It has no coordinate for a completed zeta continuation, RH, Lee–Yang classification, host complex equality, noncompact hidden scale, selected infinite prime product, or global spectral theorem.

Theorem 8995.2 (PrimeParityPhase NameCert obligations)**.** *For an accepted* $\mathsf{PrimeParityPhase}^{\uparrow}$ *carrier*

$$P = (Q, U, F, H, D, T, C, V, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimeParityPhase}^{\uparrow}}$ *surface consists exactly of prime-charge exposure, parity and unit-boundary exposure, phase-routing exposure, height-ledger exposure, triadic closure-defect exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted phase or energy read must pass through*

$$Q, U, F, H, D \longmapsto T, C, V, N$$

before the read is exported.

Proof. Project the carrier of Definition 8995.1. Its visible dictionary rows are Q, U, F, H, D, and its structural rows are T, C, V, N.

The source dictionary of section 9413.37 separates prime charge, parity, phase routing, height ledger, and triadic closure defect. The carrier keeps those rows separate until the local NameCert read asks for a phase or energy-facing export.

Transport and replay are componentwise through T and C, while provenance and local naming are exhausted by V and N. Since no row records RH, analytic continuation, Lee–Yang classification, host complex equality, hidden noncompact scale, or a selected infinite prime product, the NameCert obligations are exactly the displayed finite dictionary rows. □

Theorem 8995.3 (PrimeParityPhase window readback)**.** *Every finite* $\mathsf{PrimeParityPhase}^{\uparrow}$ *phase window readback is anchored in the prime-charge row, parity and unit-boundary row, phase-routing row, height-ledger row, triadic closure-defect row, transport, replay, provenance, and local naming of the same accepted carrier.*

Proof. Let

$$P = (Q, U, F, H, D, T, C, V, N)$$

be an accepted carrier. By Definition 8995.2, a phase or energy consumer reads Q, U, F, H, D before any export.

The finite window is therefore not a host complex phase, a hidden real scale, or a selected infinite Euler product. It is the readback of the displayed route Q, U, F, H, D under the componentwise transport T, replay C, provenance V, and local naming N. □

Falsifiable prediction: Any accepted read of phase or energy without first displaying the prime-charge row, parity and unit-boundary row, phase-routing row, height-ledger row, transport row, replay row, provenance row, and local naming row refutes the $\mathsf{PrimeParityPhase}^{\uparrow}$ carrier.

Independence witness: prime-charge dictionary, zeta-continuation-socket, solenoid-lee-yang, dyadicratcore, streamname: the $\mathsf{PrimeParityPhase}^{\uparrow}$ packet is not bijective to any sibling because it jointly carries prime charge, parity, unit boundary, phase routing, height ledger, closure defect, transport, replay, provenance, and local naming. The dictionary supplies the source rows, the zeta and solenoid sockets remain outside the finite NameCert claim, and the dyadic and stream siblings supply readback discipline without absorbing the prime-parity-phase carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeParityPhase}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeParityPhase}^{\uparrow}$ packet is generated from finite prime-charge, parity, unit-boundary, phase-routing, height-ledger, triadic closure-defect, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeParityPhase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8995.1, Definition 8995.2, and Definition 8995.3 seed the finite prime-charge, parity, phase, height-ledger, closure-defect, transport, replay, provenance, and naming dictionary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeParityPhase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, analytic continuation, Lee–Yang classification, host complex equality, noncompact hidden scale, selected infinite prime products, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Nontrivial and FieldFaithful TasteGate instances together with carrier, readback, transport, replay, provenance, and local naming rows for BEDC.Derived.PrimeParityPhaseUp.

Chapter 8996

A Concrete Naming Certificate for $\mathsf{LagrangeSpectrum}^{\uparrow}$

Following Perron and Cassels on the classical Lagrange spectrum of Diophantine approximation constants, the $\mathsf{LagrangeSpectrum}^{\uparrow}$ packet records a finite spectrum-window certificate over displayed real names and rational approximation data. It sits after the Hurwitz approximation packet of chapter 2111, the continued-fraction real packet of chapter 6738, the Farey and Stern–Brocot rows of chapter 1759 and chapter 1704, and the terminal StreamName, RegSeqRat, DyadicRatCore, and Real seals of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names finite approximation windows; it does not classify the full Lagrange spectrum or select infinite continued fractions.

Definition 8996.1 (Lagrange spectrum finite-window carrier)**.** *A* $\mathsf{LagrangeSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, K, H, F, S, D, W, Q, R, T, C, P, N)$$

with the following displayed rows:

1. *X is the displayed Real input row, naming the real source only through the available finite seal.*
2. *A is the finite continued-fraction prefix row read through* $\mathsf{ContinuedFractionReal}^{\uparrow}$*.*
3. *K is the approximation-constant row, listing the rational approximants and finite values used in the local spectrum window.*
4. *H is the Hurwitz comparison row, recording the finite Hurwitz-style threshold comparison visible in the window.*
5. *F is the Farey comparison row, recording the neighbouring rational interval data used to bound the approximants.*
6. *S is the Stern–Brocot comparison row, recording the finite tree route to the same rational window.*
7. *D is the DyadicRatCore tolerance row, displaying the dyadic tolerance and common-scale budget.*
8. *W is the StreamName window row, listing the finite observation schedule for the real source.*
9. *Q is the RegSeqRat readback row, giving the regular rational sequence read of the same window.*
10. *R is the Real seal row, recording the finite seal through which the spectrum-window data are exposed.*

11. *T records componentwise* hsame *transport for all displayed rows.*

12. *C records displayed* Cont *replay from the approximation-window request through $X, A, K, H, F, S, D, W, Q, R, T, P$*

13. *P is the* Pkg *provenance over* $\mathsf{LagrangeSpectrum}^{\uparrow}$*, the classical approximation siblings listed above,* BHist*,* hsame*,* Cont*, and* NameCert*.*

14. *N is the local* $\mathsf{NameCert}_{\mathsf{LagrangeSpectrum}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $X, A, K, H, F, S, D, W, Q, R, T, C, P, N$. It has no coordinate for the completed Lagrange spectrum, an infinite continued-fraction choice, quotient real equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 8996.2 (LagrangeSpectrum NameCert obligations)**.** *For an accepted* $\mathsf{LagrangeSpectrum}^{\uparrow}$ *carrier*

$$L = (X, A, K, H, F, S, D, W, Q, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LagrangeSpectrum}^{\uparrow}}$ *surface consists exactly of displayed Real input exposure, finite continued-fraction prefix exposure, approximation-constant row exposure, Hurwitz comparison, Farey comparison, Stern–Brocot comparison, dyadic tolerance exposure, StreamName window exposure, RegSeqRat readback, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 8996.1. The rows $X, A, K, H, F, S, D, W, Q, R$ are the visible approximation-window data, and T, C, P, N are the transport, replay, provenance, and local naming rows.

The classical approximation rows remain finite. Hurwitz data are read through chapter 2111, Farey data through chapter 1759, and Stern–Brocot data through chapter 1704. The continued-fraction row is only the prefix A, not a selected infinite expansion.

The dyadic, StreamName, RegSeqRat, and Real rows expose the same finite request through D, W, Q, R. Since the carrier has no row for quotient real equality, trichotomy, countable choice, ambient Real completeness, or completed spectrum classification, the NameCert obligations are exhausted by the displayed finite-window route. □

Theorem 8996.3 (LagrangeSpectrum finite approximation route)**.** *Every accepted* $\mathsf{LagrangeSpectrum}^{\uparrow}$ *approximation-window read factors through the finite continued-fraction prefix, Hurwitz comparison, Farey comparison, Stern–Brocot comparison, dyadic tolerance, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows of the same carrier.*

Proof. Let

$$L = (X, A, K, H, F, S, D, W, Q, R, T, C, P, N)$$

be an accepted carrier. By Definition 8996.2, the approximation-constant row K is read together with A, H, F, S, D, W, Q, R before export.

The route is therefore a finite comparison surface. The continued-fraction prefix supplies the displayed approximants, Hurwitz supplies the threshold comparison, Farey and Stern–Brocot supply rational-window checks, and the dyadic and stream rows bind the finite tolerance to the Real seal.

Transport, replay, provenance, and local naming are supplied by T, C, P, N. Thus the read has a finite BEDC route and does not require a completed spectrum, selected infinite continued fraction, quotient equality of reals, trichotomy, countable choice, or ambient completeness. □

Theorem 8996.4 (LagrangeSpectrum Real seal handoff)**.** *The* $\mathsf{LagrangeSpectrum}^{\uparrow}$ *finite-window route hands its displayed approximation data to* $\mathsf{Real}^{\uparrow}$ *only through the StreamName window, RegSeqRat readback, DyadicRatCore tolerance, Real seal row, transport, replay, provenance, and local naming rows already present in the carrier.*

Proof. Start with the accepted carrier of Definition 8996.1. The Real-facing rows are $X, D, W, Q, R, T, C, P, N$.

Apply Definition 8996.2. The Real seal R is downstream of the displayed real input, dyadic tolerance, StreamName window, and RegSeqRat readback; it is not an ambient completeness principle.

Therefore the handoff to $\mathsf{Real}^{\uparrow}$ is local to the finite packet. Any consumer that asks for a spectrum-window value must read the same tolerance, stream, regular-sequence, seal, transport, replay, provenance, and naming rows, and cannot replace them by quotient real equality or a global real-completeness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{LagrangeSpectrum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LagrangeSpectrum}^{\uparrow}$ packet is generated from displayed Real input, finite continued-fraction prefix, approximation-constant, Hurwitz, Farey, Stern–Brocot, DyadicRatCore tolerance, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LagrangeSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 8996.1, Definition 8996.2, Definition 8996.3, and Definition 8996.4 seed the finite Diophantine-approximation spectrum-window route over real names and rational comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LagrangeSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Lagrange-spectrum classification, selected infinite continued fractions, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier, finite approximation route, Real seal handoff, transport, replay, provenance, and local naming rows for BEDC.Derived.LagrangeSpectrumUp.

Chapter 8997

A Concrete Naming Certificate for $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$

Following Bishop and Bridges on diagonal arguments for regular Cauchy data, the $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$ packet records the finite L10 route by which a located diagonal sequence is read before any completion-facing consumer reaches the real seal. It sits between the regular-sequence carrier of chapter 5965, the schedule surface of chapter 126, the rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, the diagonal selector of chapter 372, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The carrier stores the diagonal route as finite NameCert data, not as quotient streams, choice-selected representatives, or ambient completeness.

Definition 8997.1 (BishopLocatedDiagonalSequence carrier)**.** *A* $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, D, L, R, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the inspected finite windows of the source family;*
2. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values visible in those windows;*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, exposing the dyadic budget used by the located diagonal read;*
4. L *is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row whose accepted carrier supplies the family-level regular-sequence route;*
5. R *is the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *diagonal row, read only after* S, Q, D, L *have been exposed;*
6. H *records componentwise* hsame *transport for* S, Q, D, L, R, C, P, N*;*
7. C *records displayed* Cont *replay from the regular-sequence source to the stream window, regular rational readback, dyadic tolerance row, diagonal row, and real-facing seal route;*
8. P *is the* Pkg *provenance over* $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularSequenceSpace}^{\uparrow}$*,* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $S, Q, D, L, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected diagonal representatives, countable-choice schedules, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 8997.2 (BishopLocatedDiagonalSequence NameCert obligations)**.** *For the carrier of Definition 8997.1, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for $B = (S, Q, D, L, R, H, C, P, N)$;*
2. *analytic exposure of the* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row L, and* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *diagonal row R;*
3. *structural stability through componentwise* hsame *transport H and displayed* Cont *replay C;*
4. *provenance and local naming through P and N;*
5. *nonescape for quotient streams, selected diagonal representatives, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 8997.1. The analytic data are exactly S, Q, D, L, R, and the structural data are exactly H, C, P, N.

The schedule, readback, and tolerance rows expose the finite observation surface. The regular-sequence source row L keeps the family-level read inside $\mathsf{RegularSequenceSpace}^{\uparrow}$, while the diagonal row R is unavailable until those finite rows have been displayed.

Transport and replay preserve this route componentwise. Since the carrier has no row for quotient equality, choice of representatives, countable-choice schedules, or ambient completeness, the listed obligations exhaust the NameCert surface. □

Theorem 8997.3 (BishopLocatedDiagonalSequence L10 route)**.** *Every completion-facing read of an accepted* $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$ *carrier passes through chapter 5965, chapter 126, chapter 301, chapter 291, and chapter 38 before the diagonal read is public. The route may cite the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *selector of chapter 372, but only as a consumer of the displayed regular-sequence, stream-window, rational-readback, and dyadic-tolerance rows.*

Proof. Start with the regular-sequence source row L. It binds the diagonal read to the accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier before any completion-facing endpoint is visible.

Read the finite observation rows next. The $\mathsf{StreamName}^{\uparrow}$ schedule S fixes the inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row Q supplies the regular rational readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the displayed tolerance budget.

Only then consume the diagonal row R. The $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ selector receives the finite window, readback, and tolerance data already exposed by the source route, so it cannot introduce quotient streams or selected representatives.

Finally replay H, C, P, N into the $\mathsf{Real}^{\uparrow}$ seal. The terminal real-facing read is therefore public only after the displayed L10 route has passed through the regular-sequence source, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance surface, and Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance row, RegularSequenceSpace source row, RegularCauchyDiagonal selector row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}}$, seed)

Scope closed: Definition 8997.1, Definition 8997.2, and Definition 8997.3 seed the finite located diagonal-sequence route from regular sequence spaces to the Real seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedDiagonalSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove Lean verification, quotient stream equality, selected diagonal representatives, countable-choice schedules, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Formal closure requires a Lean carrier for BEDC.Derived.BishopLocatedDiagonalSequenceUp and checked NameCert obligations for the displayed L10 route.

Chapter 8998

A Concrete Naming Certificate for $\mathsf{UniformCauchyNetBridge}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Cauchy nets, the $\mathsf{UniformCauchyNetBridge}^{\uparrow}$ packet records the finite handoff by which directed Cauchy-net data, entourage-filter data, and regular-Cauchy readback are inspected before any completion-facing consumer may read a terminal seal. It sits between the regular Cauchy net surface of chapter 6338, the Cauchy entourage-filter route of chapter 1791, the uniform-space completion packet of chapter 3560, the regular rational readback of chapter 301, the finite stream windows of chapter 126, and the real-name seal of chapter 38. The packet names a constructive bridge object, not a quotient of nets, a selected limit, a countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a checked Lean bridge.

Definition 8998.1 (Uniform Cauchy net bridge carrier)**.** *A* $\mathsf{UniformCauchyNetBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (N, E, F, R, S, Z, H, C, P, L)$$

with the following displayed rows:

1. *N is the directed Cauchy-net source row, naming the directed index data, net observations, and displayed Cauchy-entourage requests.*
2. *E is the entourage row, recording finite uniform entourages and their monotone refinement route.*
3. *F is the filter row, recording the Cauchy filter data read from the entourage route and the compatibility witness with the net source.*
4. *R is the regular-Cauchy readback row, exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name obtained from the displayed net/filter windows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observations by which the directed net and regular readback are inspected.*
6. *Z is the completion-seal row, recording the* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *handoff and the terminal* $\mathsf{Real}^{\uparrow}$ *zero-distance or classifier row available to completion consumers.*
7. *H records componentwise* hsame *transport for $N, E, F, R, S, Z, C, P, L$.*
8. *C records displayed* Cont *replay for directed-net inspection, entourage refinement, filter compatibility, regular-Cauchy readback, stream-window inspection, completion sealing, and zero-distance or classifier consumption.*

9. *P is the* Pkg *provenance over* UniformCauchyNetBridge$^{\uparrow}$, RegularCauchyNet$^{\uparrow}$, CauchyEntourageFilter$^{\uparrow}$, UniformSpaceCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *L is the local* NameCert$_{\text{UniformCauchyNetBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting nets, selecting a limit point, invoking countable choice, proving ambient completeness of Real$^{\uparrow}$*, or replacing the displayed bridge with a host uniform-completion theorem.*

Theorem 8998.2 (Uniform Cauchy net bridge NameCert obligations)**.** *For the carrier of Definition 8998.1, the local* NameCert$_{\text{UniformCauchyNetBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (N, E, F, R, S, Z, H, C, P, L)$, directed-entourage stability through N, E, filter/net compatibility through F, regular readback through R, stream-window inspection through S, completion sealing and zero-distance or classifier exposure through Z, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier $K = (N, E, F, R, S, Z, H, C, P, L)$. Carrier admission supplies the directed net source before any entourage or filter row is consumed.

Read E and F together. The entourage row displays finite uniform refinements, while the filter row records that the same displayed route is compatible with the directed Cauchy-net source.

Next read R, S, Z. The regular-Cauchy readback is inspected through finite stream windows before the completion seal and the Real$^{\uparrow}$ zero-distance or classifier row can be consumed.

Finally use H, C, P, L. Transport, replay, provenance, and local naming preserve only the displayed packet, so the listed rows exhaust the NameCert surface and add no quotient net, selected limit, countable-choice schedule, ambient Real completeness theorem, or checked bridge. □

Falsifiable prediction: A completion consumer that reads a selected limit or quotient net from UniformCauchyNetBridge$^{\uparrow}$ without displaying $N, E, F, R, S, Z, H, C, P, L$ refutes this packet.

Independence witness: Nearest siblings: regular-cauchy-net, cauchy-entourage-filter, uniform-space-completion, regseqrat, streamname, real. None carries the combined bridge packet because each sibling exposes one route, while UniformCauchyNetBridge$^{\uparrow}$ stores directed-net source data, entourage/filter compatibility, regular-Cauchy readback, stream windows, completion seal, zero-distance or classifier exposure, transport, replay, provenance, and local naming in one finite carrier.

Closure status for NameCert$_{\text{UniformCauchyNetBridge}^{\uparrow}}$**.**

Constructive story: A UniformCauchyNetBridge$^{\uparrow}$ packet is generated by displaying a directed Cauchy-net source, finite entourage refinements, a compatible Cauchy-filter row, regular-Cauchy readback, stream windows, a completion seal, zero-distance or classifier exposure, componentwise transport, replay, provenance, and local naming in one finite BHist carrier.

Theory closure: TheoryGate(NameCert$_{\text{UniformCauchyNetBridge}^{\uparrow}}$, seed)

Scope closed: Definition 8998.1 and Definition 8998.2 bind the seed carrier and local NameCert obligations for the constructive net/filter/regular-Cauchy completion handoff.

Formal verification: FormalStatus(NameCert$_{\text{UniformCauchyNetBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim quotient nets, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: A scoped closure would require stable downstream theorems for directed-entourage compatibility, regular readback, completion sealing, and the displayed refusal boundary.

Chapter 8999

A Concrete Naming Certificate for $\mathsf{StrongChoquetSpace}^{\uparrow}$

Following Oxtoby and Kechris on Choquet-game topology, the $\mathsf{StrongChoquetSpace}^{\uparrow}$ packet records the finite rows by which a complete-metric or Baire-style space exposes a player-two response strategy without importing an ambient powerset of open sets. It sits beside the Baire-space carrier of chapter 334, the complete-metric surface of chapter 3687, the Polish-space carrier of chapter 1490, the stream-window discipline of chapter 126, and the topology source of chapter 88. The chapter names the finite Choquet-game certificate only; it does not assert choice-selected infinite plays, host open-set powersets, or ambient completeness.

8999.1 Strong Choquet Carrier

Definition 8999.1 (Strong Choquet space carrier)**.** *A* $\mathsf{StrongChoquetSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{Chq}(T, B, O, R, S, W, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, naming the finite open-neighborhood interface used by the game.*
2. *B is the Baire and Polish comparison row, recording the visible handoff to* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *and* $\mathsf{PolishSpace}^{\uparrow}$ *without asserting equality with any sibling carrier.*
3. *O is the open-move ledger, listing the finite initial-point, open-set, and refinement requests that the packet actually observes.*
4. *R is the response-strategy ledger, giving the displayed player-two response row for each listed open move.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite play prefix and the observation windows through which nested moves are read.*
6. *W is the winning-boundary row, naming the finite nonempty-intersection and legality checks exported by the packet.*
7. *H records componentwise* hsame *transport for topology source, comparison, open moves, responses, stream windows, winning boundary, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a game request through $T, B, O, R, S, W, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{StrongChoquetSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StrongChoquetSpace}^{\uparrow}}$ *row.*

The carrier contains no host collection of all open sets, no chosen infinite play, no ambient complete metric, and no rule that identifies the Choquet strategy with a Baire or Polish carrier.

Theorem 8999.2 (Strong Choquet NameCert obligations)**.** *For every accepted* $\mathsf{StrongChoquetSpace}^{\uparrow}$ *carrier* $\mathsf{Chq}(T, B, O, R, S, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StrongChoquetSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission, Baire and Polish comparison, finite open-move admission, response-strategy exposure, stream-window exposure, winning-boundary exposure, componentwise transport, displayed replay, provenance, and local naming. Any accepted Choquet-game consumer must read*

$$T, B, O, R, S, W \longmapsto H, C, P, N$$

before the strategy row is exported.

Proof. Project the carrier of Definition 8999.1. The game-facing surface is exactly T, B, O, R, S, W, and the structural surface is exactly H, C, P, N.

The open-move ledger O and response ledger R are finite rows. They expose only the moves and replies already listed in the packet, while the stream row S records the play prefix used by a consumer.

The winning-boundary row W is therefore read from displayed finite legality and intersection checks. Since the carrier contains no host open-set powerset, selected infinite play, or ambient complete metric, the listed rows exhaust the local NameCert obligations. □

Theorem 8999.3 (Strong Choquet winning strategy handoff)**.** *Every winning-strategy read of an accepted* $\mathsf{StrongChoquetSpace}^{\uparrow}$ *carrier factors through the topology source T, Baire and Polish comparison row B, open-move ledger O, response ledger R,* $\mathsf{StreamName}^{\uparrow}$ *window row S, winning boundary W, transport H, replay C, provenance P, and local naming N. The handoff exports a finite strategy surface and does not choose an infinite play, a host powerset of open sets, or a complete metric outside the displayed packet.*

Proof. Start with a consumer asking for a strategy. The topology source T supplies only the displayed open-neighborhood interface, and O lists the open moves that the packet accepts.

The response row R then gives the corresponding player-two replies, while S records the finite play windows and W records the boundary checks needed for a winning read. The comparison row B keeps the Baire, complete-metric, and Polish siblings available only as named dependency surfaces.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Without a coordinate for choice-selected infinite plays, host open-set powersets, or ambient completeness, the exported handoff is exactly the displayed finite strategy surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{StrongChoquetSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StrongChoquetSpace}^{\uparrow}$ packet is generated from topology-source, Baire and Polish comparison, finite open-move, response-strategy, StreamName window, winning-boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StrongChoquetSpace}^{\uparrow}}$, seed)

Scope closed: Definition 8999.1, Definition 8999.2, and Definition 8999.3 seed the finite Choquet-game strategy surface over topology, Baire, complete-metric, Polish, StreamName, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StrongChoquetSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected infinite plays, host open-set powersets, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-move exactness, response-strategy handoff, StreamName window stability, winning-boundary coverage, componentwise transport, provenance, and local naming for BEDC.Derived.StrongChoquetSpaceUp.

Chapter 9000

A Concrete Naming Certificate for $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$

The $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ packet realizes the prime-fingerprint separation row of chapter 9248 as finite NameCert data. It binds prime-probe rows from chapter 64, zeta-channel projection rows from chapter 71, rational and dyadic response ledgers from chapter 37 and chapter 291, transport, replay, provenance, and local naming before any RH-facing fingerprint consumer reads a spectral separation witness. The chapter names the finite prime-local response packet only; it does not prove RH, analytic continuation, host complex equality, or an infinite Euler product.

9000.1 Prime Spectral Fingerprint Carrier

Definition 9000.1 (Prime spectral fingerprint carrier)**.** *A* $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{PSF}(P, Z, D, R, H, C, G, N)$$

with the following displayed rows:

1. *P is the prime-probe row, listing the finite prime names and local probe windows inspected by the packet.*
2. *Z is the zeta projection row, recording the visible prime-local channel response consumed from the zeta packet.*
3. *D is the dyadic response ledger, storing the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximants used by the response comparison.*
4. *R is the rational response ledger, storing the finite* $\mathsf{Rat}^{\uparrow}$ *readback rows used to compare probe amplitudes and phases.*
5. *H records componentwise* hsame *transport for prime probes, zeta projection, dyadic responses, rational responses, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a fingerprint request through P, Z, D, R, H, G, N.*
7. *G is the* Pkg *provenance over* $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, *zeta projection rows,* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimeSpectralFingerprint}^{\uparrow}}$ *row.*

The carrier contains no host complex equality, no completed zeta continuation, no selected infinite prime product, no RH proof object, and no ambient spectral theorem.

Theorem 9000.2 (PrimeSpectralFingerprint NameCert obligations)**.** *For every accepted* $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ *carrier* $\mathsf{PSF}(P, Z, D, R, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PrimeSpectralFingerprint}^{\uparrow}}$ *surface consists exactly of prime-probe exposure, zeta-projection exposure, dyadic response-ledger exposure, rational response-ledger exposure, componentwise transport, displayed replay, provenance, and local naming. Any accepted prime-local zeta response read must pass through*

$$P, Z, D, R \longmapsto H, C, G, N$$

before a separation consumer may inspect the fingerprint.

Proof. Project the carrier of Definition 9000.1. The fingerprint-facing rows are P, Z, D, R, and the structural rows are H, C, G, N.

The prime-probe row P supplies the finite probe names, while the zeta row Z supplies the corresponding visible channel responses. The dyadic and rational ledgers D and R record only finite approximants and readback rows used by that response comparison.

Transport, replay, provenance, and naming are exhausted by H, C, G, N. Since no row records host complex equality, a completed continuation, an infinite Euler product, or an RH proof object, the NameCert obligations are exactly the finite rows listed above. □

Theorem 9000.3 (PrimeSpectralFingerprint separation witness)**.** *Every separation witness exported by an accepted* $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ *carrier factors through the prime-probe row* P*, zeta projection row* Z*, dyadic ledger* D*, rational ledger* R*, transport* H*, replay* C*, provenance* G*, and local naming* N*. The witness may record that two displayed spectral histories differ at a prime-local response row, but it does not identify histories by host complex equality or by an unlisted infinite family of probes.*

Proof. Start with a consumer asking whether two displayed histories are separated by a prime-local response. The packet can inspect only the prime names in P and the zeta-channel response rows in Z.

The dyadic and rational ledgers D and R then provide the finite comparison surface for the response readback. They do not add a completed analytic continuation or a global Euler product; they only expose the approximants and rational rows already present in the packet.

Finally H, C, G, N transport, replay, source, and name the same finite witness. Therefore the exported separation row is prime-local and finite, with no host equality or unlisted infinite probe family. □

Falsifiable prediction: Any accepted prime-local zeta response read that exports a $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ separation without displaying the prime-probe row, zeta projection row, dyadic response ledger, rational response ledger, transport row, replay row, provenance row, and local naming row refutes the carrier.

Independence witness: prime, zeta, rat, dyadicratcore, namecert: the $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ packet is not bijective to any sibling because it jointly carries prime probes, zeta-channel projections, dyadic response approximants, rational readback rows, transport, replay, provenance, and local naming. Prime and zeta siblings supply source channels, rational and dyadic siblings supply response ledgers, and NameCert supplies the local naming discipline without absorbing the fingerprint carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeSpectralFingerprint}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ packet is generated from prime-probe, zeta-projection, dyadic response-ledger, rational response-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeSpectralFingerprint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9000.1, Definition 9000.2, and Definition 9000.3 seed the finite prime-local zeta response surface over PrimeUp, zeta projection, RatUp, DyadicRatCoreUp, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeSpectralFingerprint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RH, analytic continuation, host complex equality, selected infinite prime products, ambient spectral classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prime-probe exactness, zeta-projection response coverage, dyadic and rational ledger stability, componentwise transport, provenance, and local naming for BEDC.Derived.PrimeSpectralFingerpr

Chapter 9001

A Concrete Naming Certificate for $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$

Following Darboux, Stieltjes, and Bishop-style constructive analysis, the $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$ packet records the finite upper-lower integration route for a bounded integrand read against a bounded-variation integrator. It sits beside the Darboux finite-partition surface of chapter 1460, the tagged Riemann–Stieltjes route of chapter 1637, the variation ledger of chapter 398, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the displayed finite Stieltjes mesh and gap route only; it does not import measure completion, choice-selected tags, quotient integrands, completed variation objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9001.1 (Darboux–Stieltjes integral carrier)**.** *A* $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, F, A, U, L, SU, SL, G, R, E, H, C, Pkg, N)$$

with the following displayed rows:

1. *P is the finite partition row, carrying the displayed interval endpoints, cell order, and common-refinement ledger used by all later rows.*
2. *F is the bounded integrand row, exposing bounded rational or real-value windows on the cells named by P.*
3. *A is the* $\mathsf{BoundedVariation}^{\uparrow}$ *integrator row, carrying endpoint changes, transported jump or oscillation rows, and the finite variation budget attached to the same partition.*
4. *U and L are upper and lower Darboux–Stieltjes envelope rows over a displayed common refinement of P.*
5. *SU and SL are the finite upper and lower Darboux–Stieltjes sum rows computed from U, L and the integrator changes in A.*
6. *G is the upper-lower gap and modulus row, binding the visible precision request to the finite difference between SU and SL.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the gap/modulus ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after $P, F, A, U, L, SU, SL, G, R$ have been displayed.*
9. *H records componentwise* hsame *transport for all carrier rows.*

10. *C records displayed* Cont *replay of refinements, variation reads, sum computations, gap comparisons, regular readback, and real sealing.*
11. *Pkg is the* Pkg *provenance over* DarbouxStieltjesIntegral$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, RiemannStieltjes$^{\uparrow}$, BoundedVariation$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{DarbouxStieltjesIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, F, A, U, L, SU, SL, G, R, E, H, C, Pkg, N$. It has no coordinate for measure completion, choice-selected partition tags, host integrand equality, quotient functions, selected limiting partitions, completed variation objects, or ambient real completeness.

Theorem 9001.2 (Darboux–Stieltjes integral NameCert obligations)**.** *For every accepted* DarbouxStieltjesIntegral$^{\uparrow}$ *carrier $S = (P, F, A, U, L, SU, SL, G, R, E, H, C, Pkg, N)$, the local* NameCert$_{\text{DarbouxStieltjesIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, partition-refinement stability for P, U, L, bounded-integrand exposure through F, bounded-variation integrator control through A, finite upper/lower Darboux–Stieltjes sum exactness for SU, SL, gap/modulus exposure through G,* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *sealing through R, E, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and ledger refusal for measure completion, selected tags, quotient functions, selected limiting partitions, completed variation objects, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 9001.1. The packet first displays the partition, integrand bounds, bounded-variation integrator row, upper and lower envelope rows, and finite sum rows before any gap or seal can be consumed.

Refinement stability is inherited from the Darboux and Riemann–Stieltjes sibling routes. The partition and envelope rows follow the finite Darboux common-refinement discipline, while the integrator changes are read only through the bounded-variation ledger attached to the same refined cells.

The sum and gap obligations are then finite ledger checks. The upper and lower Darboux–Stieltjes sums compute from the displayed envelopes and integrator changes, the gap row records the visible precision comparison, and R, E seal that comparison as a regular rational and real-name handoff. Since the excluded host objects are absent from the carrier coordinates, no continuation or transport row can export them. □

Theorem 9001.3 (Darboux–Stieltjes variation handoff)**.** *Every* DarbouxIntegral$^{\uparrow}$ *or* RiemannStieltjes$^{\uparrow}$ *consumer of an accepted* DarbouxStieltjesIntegral$^{\uparrow}$ *packet must read the finite partition row P, bounded integrand row F,* BoundedVariation$^{\uparrow}$ *integrator row A, upper and lower envelope rows U, L, finite sum rows SU, SL, gap/modulus row G,* RegSeqRat$^{\uparrow}$ *readback R,* Real$^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance Pkg, and local* NameCert *row N before treating the packet as an integral handoff. The consumer receives a finite variation-controlled Darboux route; it receives no measure completion, selected tags, quotient integrand equality, selected limiting partition, completed variation object, or ambient real-completeness proof.*

Proof. Start from the NameCert obligations of Definition 9001.2. They expose the carrier, partition refinement, bounded integrand, bounded-variation integrator, finite sum, gap, readback, seal, transport, replay, provenance, and local naming rows as the complete public surface.

For a DarbouxIntegral$^{\uparrow}$ consumer, forget only the Stieltjes-specific reading of A as an integrator ledger while keeping the partition, upper and lower envelope rows, finite sum rows, gap row, regular readback, and real seal visible. This is still a finite Darboux route because the sums and gap are computed from displayed cells.

For a $\mathsf{RiemannStieltjes}^{\uparrow}$ consumer, read A as the bounded-variation handoff and use the same finite mesh, sum, and seal order. The tagged route is not supplied by choice-selected tags; it is constrained by the displayed Stieltjes partition and variation ledger already present in the carrier.

The structural rows H, C, Pkg, N preserve this order and add no analytic coordinate. Thus every downstream integration handoff remains inside the finite variation-controlled packet and cannot export the refused host data. □

Falsifiable prediction: Every accepted $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$ read displays one finite partition row, one bounded integrand row, one BoundedVariationUp integrator row, upper and lower Darboux–Stieltjes envelope rows, upper and lower finite sum rows, one gap/modulus row, one RegSeqRatUp readback row, one RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before any integral consumer reads the seal.

Independence witness: darboux-integral, riemann-stieltjes, bounded-variation, regseqrat, real: $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite upper/lower Darboux rows to bounded-variation integrator changes before the regular-rational and real-name seal, while refusing measure completion, selected tags, quotient function equality, selected limiting partitions, completed variation objects, and ambient real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxStieltjesIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxStieltjesIntegral}^{\uparrow}$ packet is generated from a finite partition row, bounded integrand row, BoundedVariationUp integrator row, upper and lower envelope rows, finite upper and lower Darboux–Stieltjes sum rows, a gap/modulus row, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxStieltjesIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9001.1, Definition 9001.2, and Definition 9001.3 seed the finite variation-controlled Darboux–Stieltjes handoff through $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxStieltjesIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, selected tags, quotient integrand equality, selected limiting partitions, completed variation objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite sum exactness, bounded-variation handoff, gap/modulus nonescape, RegSeqRatUp readback, and RealUp seal ordering for BEDC.Derived.DarbouxStieltjesIntegralUp.

Chapter 9002

A Concrete Naming Certificate for $\mathsf{FiniteEnergyTower}^{\uparrow}$

The $\mathsf{FiniteEnergyTower}^{\uparrow}$ packet realizes the finite-to-infinite phase-energy tower of section 9413.55 as concrete NameCert data. It reads finite prime windows through the prime-window gluing surface of chapter 8765, prime rows through chapter 64, zeta-facing boundary data through chapter 71, complex coordinates through chapter 39, real readback through chapter 38, and dyadic tolerance rows through chapter 291. The packet names finite positive normal-energy evidence and monotone window extension; it does not claim RH, zero exclusion, an infinite spectral theorem, Euler-product completion, or analytic continuation beyond the displayed rows.

Definition 9002.1 (Finite energy tower carrier)**.** *A* $\mathsf{FiniteEnergyTower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, \Phi, E, M, Z, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite prime-window row, carrying a nonempty displayed prime list, the finite inclusion order used for window extension, and the local prime provenance.*
2. *Φ is the phase-profile row, recording the finite phase rotations attached to the primes in W at the displayed complex coordinate.*
3. *E is the positive normal-energy row, computing the finite sum of nonnegative normal square terms and recording positivity when the displayed normal displacement is nonzero.*
4. *M is the monotone window-extension row, showing that passing from one displayed finite prime window to a larger displayed window only adds nonnegative normal-energy terms.*
5. *Z is the zeta-facing boundary row, recording the local zero-boundary request without asserting zero exclusion or a completed infinite tower.*
6. *R is the* $\mathsf{Real}^{\uparrow}/\mathsf{Complex}^{\uparrow}$ *readback row, exposing the real normal-energy value and the complex coordinate only after W, Φ, E, M, Z have been displayed.*
7. *H records componentwise* hsame *transport for finite windows, phase profiles, energy rows, monotone extension, zeta boundary, readback, provenance, and local naming.*
8. *C records displayed* Cont *replay from prime-window inspection through phase readback, positive energy, monotone extension, boundary request, and Real/Complex sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteEnergyTower}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$, $\mathsf{Zeta}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEnergyTower}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, \Phi, E, M, Z, R, H, C, P, N$. It has no coordinate for a completed infinite prime tower, a proof of RH*, global zeta recovery, infinite spectral completeness, Euler-product completion, selected limiting windows, or ambient analytic continuation.*

Theorem 9002.2 (Finite energy tower NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEnergyTower}^{\uparrow}$ *carrier $T = (W, \Phi, E, M, Z, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteEnergyTower}^{\uparrow}}$ *surface consists exactly of finite prime-window admission, phase-profile exposure, positive normal-energy exposure, monotone finite-window extension, zeta-boundary request routing,* $\mathsf{Real}^{\uparrow}/\mathsf{Complex}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for completed infinite towers, RH proof claims, zero-exclusion claims, infinite spectral theorems, Euler-product completion, selected limiting windows, and ambient analytic continuation.*

Proof. Carrier admission is projection from Definition 9002.1. The packet first displays the finite prime window W, the phase profile Φ, the positive normal-energy row E, and the monotone extension row M.

The zeta-facing row Z is only a boundary request. It records where a zero-relevance consumer would inspect the finite energy data, while leaving the global zero law outside this packet.

The readback row R is downstream of those finite rows. It exposes Real and Complex data only after the prime window, phase profile, normal-energy, monotonicity, and boundary request are visible.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no carrier coordinate stores an infinite tower, RH proof, zero exclusion theorem, spectral completion, Euler-product completion, limiting-window selector, or ambient analytic continuation principle, the listed rows exhaust the NameCert surface. □

Falsifiable prediction: A $\mathsf{FiniteEnergyTower}^{\uparrow}$ consumer reaching the energy obstruction without first displaying the finite prime window row W and positive normal-energy row E refutes the packet.

Independence witness: prime, zeta, complex, real, dyadicratcore: $\mathsf{FiniteEnergyTower}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite prime-window rows, phase-profile rows, positive normal-energy rows, monotone window extension, zeta-boundary request, Real/Complex readback, dyadic tolerance context, transport, replay, provenance, and local naming while refusing RH proof claims and infinite spectral closure.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEnergyTower}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteEnergyTower}^{\uparrow}$ packet is generated from a finite prime-window row, phase-profile row, positive normal-energy row, monotone window-extension row, zeta-boundary request, RealUp and ComplexUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteEnergyTower}^{\uparrow}}$, seed)

Scope closed: Definition 9002.1 and Definition 9002.2 seed finite prime-window phase rows, positive normal-energy rows, monotone extension, Real/Complex readback, and nonescape ledgers as NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteEnergyTower}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim RH, nontrivial zero exclusion, global zeta recovery, infinite spectral completeness, Euler-product completion, selected limiting windows, analytic continuation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prime-window exposure, phase-profile exposure, positive normal-energy exposure, monotone extension, Real/Complex readback, zeta-boundary nonescape, and local naming for BEDC.Derived.FiniteEnergyTowerUp.

Chapter 9003

A Concrete Naming Certificate for $\mathsf{PartialMetric}^{\uparrow}$

Following Matthews on partial metric spaces, the $\mathsf{PartialMetric}^{\uparrow}$ packet records the constructive distance surface in which self-distance is visible data rather than a suppressed diagonal law. It sits between the pre-distance carrier of chapter 5565, the zero-distance boundary of chapter 1614, the directed-distance surface of chapter 2158, the metric reader of chapter 57, and the terminal distance seal of chapter 38. The packet names only the finite NameCert rows needed to expose self-distance, partial triangle evidence, and completion-facing handoff; it does not identify points, quotient carriers, select limits, or import ambient completeness.

Definition 9003.1 (Partial metric carrier)**.** *A* $\mathsf{PartialMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, X, E, D, L, T, M, K, H, C, Q, N)$$

with the following displayed rows:

1. *S is the source carrier row. It records the traditional partial-metric source and the finite observation context in which point and distance requests are admitted.*
2. *X is the point row. It records admitted point names and their classifier without quotienting them by distance-zero evidence or by equality-like self-distance comparisons.*
3. *E is the self-distance row. For each admitted point row x, it displays the value $p(x, x)$ as a* $\mathsf{Real}^{\uparrow}$*-sealed distance witness.*
4. *D is the directed distance row. It records requested values $p(x, y)$ and keeps the reversed request $p(y, x)$ visible as a separate displayed read unless a later row supplies symmetry evidence.*
5. *L is the small-self-distance comparison row. It displays the comparisons $p(x, x) \leq p(x, y)$ and $p(y, y) \leq p(x, y)$ as finite ledger data, not as host-order laws.*
6. *T is the partial triangle ledger. It records the comparison of $p(x, z)$ with the finite route through $p(x, y)$, $p(y, z)$, and the displayed self-distance $p(y, y)$.*
7. *M is the metric-reflection handoff row. It names the boundary through which a later* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Pseudometric}^{\uparrow}$ *reader may consume the displayed self-distance and triangle data.*
8. *K is the completion-facing row. It binds every completion-oriented request to S, X, E, D, L, T, M before any completed point, Cauchy limit, or quotient representative can be named.*

9. *H records componentwise* hsame *transport for source, point, self-distance, directed distance, small-self-distance, partial triangle, metric-reflection, completion, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a consumer request through point admission, self-distance exposure, directed distance readback, small-self-distance comparison, partial triangle evidence, metric-reflection handoff, completion boundary, transport, provenance, and local naming.*
11. *Q is the* Pkg *provenance over* $\mathsf{PartialMetric}^{\uparrow}$, $\mathsf{Premetric}^{\uparrow}$, $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PartialMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, X, E, D, L, T, M, K, H, C, Q, N$. It has no coordinate for separatedness, metric completeness, open-cover compactness, quotient completion, selected representatives, selected limits, host metric equality, Lean verification, or bridge checking.

Theorem 9003.2 (Partial metric NameCert obligations)**.** *For every accepted* $\mathsf{PartialMetric}^{\uparrow}$ *carrier*

$$P = (S, X, E, D, L, T, M, K, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartialMetric}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the source row, point row, self-distance row, directed distance row, small-self-distance comparison row, partial triangle ledger, metric-reflection handoff, completion-facing row, componentwise transport, displayed replay, provenance, and local naming;*
2. *self-distance exposure, requiring every consumer that reads a point row to see the corresponding E row before treating diagonal distance as unavailable or zero;*
3. *partial triangle ledger, requiring each triangle comparison to read T through the displayed self-distance correction rather than through a host metric triangle axiom;*
4. *metric-reflection boundary, requiring every* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Pseudometric}^{\uparrow}$ *consumer to pass through M after E, D, L, T have been displayed;*
5. *classifier transport, requiring accepted packets to move only by the componentwise* hsame *row H and the replay row C;*
6. *ledger nonescape, excluding separatedness, metric completeness, open-cover compactness, quotient completion, selected representatives, selected limits, host metric equality, Lean verification, and bridge checking.*

No partial-metric NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 9003.1. The tuple lists exactly the source, point, self-distance, directed distance, small-self-distance comparison, partial triangle, metric-reflection, completion, transport, replay, provenance, and local naming rows.

Self-distance exposure is the row E. A consumer cannot erase diagonal distance before reading E, and the comparisons in L keep the small-self-distance boundary visible whenever an off-diagonal distance is inspected.

The partial triangle obligation is the row T. It routes each comparison through the displayed self-distance correction, so the triangle evidence is a finite ledger read rather than a detached metric axiom.

Metric and pseudometric consumers reach the packet only through M, while completion-facing consumers are stopped at K until the displayed source, distance, comparison, and triangle rows

have been consumed. The rows H, C, Q, N then transport, replay, source, and name the same finite surface.

The remaining exclusions follow from the carrier inventory. Since separatedness, metric completeness, open-cover compactness, quotient completion, selected representatives, selected limits, host metric equality, Lean verification, and bridge checking do not occur among the coordinates, none of those data can be exported by the local certificate. □

Theorem 9003.3 (Partial metric self-distance boundary)**.** *Let*

$$P = (S, X, E, D, L, T, M, K, H, C, Q, N)$$

be an accepted $\mathsf{PartialMetric}^{\uparrow}$ *carrier. Every* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Pseudometric}^{\uparrow}$ *consumer may read equality-like separation only through the displayed self-distance boundary*

$$X, E, D, L, T \longmapsto M \longmapsto H, C, Q, N.$$

The packet itself never exports quotient point equality, selected representatives, host metric equality, chosen limits, ambient completeness, or a completed quotient carrier.

Proof. Start with an accepted carrier and read the point row X. The self-distance row E and directed distance row D remain visible before any equality-like separation request can be interpreted.

The comparison row L and triangle row T bind the self-distance data to the off-diagonal distance ledger. Thus a consumer sees the partial-metric boundary as displayed finite evidence, not as a quotient equality or a host metric equality.

Only after those rows are displayed does the metric-reflection handoff M become available. The support rows H, C, Q, N transport, replay, source, and name that same boundary for metric and pseudometric readers.

The completion row K does not add selected representatives, chosen limits, ambient completeness, or a completed quotient carrier. Those objects are absent from the coordinate list, so equality-like separation remains scoped to the displayed self-distance boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartialMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PartialMetric}^{\uparrow}$ packet is generated from a source row, a point row, a self-distance row, a directed distance row, a small-self-distance comparison row, a partial triangle ledger, a metric-reflection handoff, a completion-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartialMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9003.1, Definition 9003.2, and Definition 9003.3 seed the constructive partial-metric boundary between premetric, pseudometric, quasi-metric, metric, Real-facing, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartialMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim separatedness, metric completeness, open-cover compactness, quotient completion, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, self-distance exposure, partial triangle ledger, metric-reflection boundary, classifier transport, and ledger nonescape for BEDC.Derived.PartialMetricUp.

Chapter 9004

A Concrete Naming Certificate for RegularCauchyLocatedIntervalModulus$^{\uparrow}$

Following Bishop–Bridges regular real practice, the RegularCauchyLocatedIntervalModulus$^{\uparrow}$ packet records the finite route by which located interval endpoint data is read together with a regular Cauchy modulus before any Real-facing seal is consumed. It sits beside the locatedness source of chapter 2216, the modulus source of chapter 1423, the stability source of chapter 1927, the interval source of chapter 1709, the stream window surface of chapter 126, the regular rational readback of chapter 301, the dyadic radius core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only displayed located interval and modulus rows; it does not assert quotient equality of real names, selected representatives, trichotomy, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 9004.1 (RegularCauchyLocatedIntervalModulus carrier)**.** *A* RegularCauchyLocatedIntervalModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, A, B, M, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* LocatedRealInterval$^{\uparrow}$ *interval row, recording the located lower-endpoint and upper-endpoint incidence used by the interval consumer.*
2. *A and B are the endpoint rows, each read through* CauchyLocatedness$^{\uparrow}$ *and tied to the same located interval request.*
3. *M is the* RealCauchyModulus$^{\uparrow}$ *row, recording the finite regular Cauchy modulus budget consumed by the endpoint reads.*
4. *W is the* StreamName$^{\uparrow}$ *window row through which the endpoint names and modulus request are inspected.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed endpoint windows and modulus budget.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *radius row, carrying the dyadic tolerance and interval-radius comparisons used by the locatedness test.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after I, A, B, M, W, Q, D have been displayed.*

8. *H records componentwise* hsame *transport for interval rows, endpoint rows, modulus rows, windows, readback, dyadic radii, Real sealing, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a located-interval modulus request through $I, A, B, M, W, Q, D, E, H$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}$, $\mathsf{CauchyLocatedness}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyModulusStability}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, A, B, M, W, Q, D, E, H, C, P, N$. It has no coordinate for host endpoint equality, quotient equality of regular names, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9004.2 (RegularCauchyLocatedIntervalModulus obligations)**.** *For every accepted* RegularCauchyLocated *carrier*

$$L = (I, A, B, M, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row, the two* $\mathsf{CauchyLocatedness}^{\uparrow}$ *endpoint rows, the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *regular modulus row, the* $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ *stability dependency, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, displayed replay, package provenance, and local naming. Any interval-modulus consumer must read those rows before citing the packet as a located regular Cauchy modulus witness.*

Proof. Project the carrier of Definition 9004.1. The interval and endpoint rows are I, A, B, and they supply the located interval surface through $\mathsf{LocatedRealInterval}^{\uparrow}$ and $\mathsf{CauchyLocatedness}^{\uparrow}$.

Read the modulus and radius rows next. Row M supplies the regular Cauchy budget, the stability dependency keeps that budget attached to the same regular read, and row D records the dyadic radius comparisons used by the locatedness test.

The window and readback rows W, Q turn the endpoint and modulus data into a finite regular rational read. The terminal seal E is downstream of that read and is not an ambient completeness theorem.

Finally replay H, C, P, N. Componentwise transport, displayed continuation, provenance, and local naming preserve the same finite packet. Since no row stores host endpoint equality, quotient equality, selected representatives, trichotomy, choice, or ambient Real completeness, the listed rows exhaust the local obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}$ read exposes located interval endpoint rows, a regular Cauchy modulus row, StreamName windows, RegSeqRat readback, DyadicRatCore radii, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data before any interval-modulus consumer can cite it.

Independence witness: cauchy-locatedness, realcauchymodulus, regular-cauchy-modulus-stability, located-real-interval, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}$ is not bijective to any listed sibling because it binds located interval endpoints, regular Cauchy modulus budget, stability dependency, finite windows, regular readback, dyadic radius comparisons, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedIntervalModulus}^{\uparrow}}$.

Constructive story: A RegularCauchyLocatedIntervalModulus$^{\uparrow}$ packet is generated from located interval endpoint rows, a regular Cauchy modulus row, a stability dependency, StreamName windows, RegSeqRat readback, DyadicRatCore radius comparisons, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyLocatedIntervalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 9004.1, Definition 9004.2, the falsifiable prediction, and the independence witness seed the finite located-interval regular-modulus route over endpoint, modulus, window, readback, radius, seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyLocatedIntervalModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient equality of regular names, selected representatives, trichotomy, countable choice, host endpoint equality, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint locatedness, regular Cauchy modulus stability, StreamName window admission, RegSeqRat readback, DyadicRatCore radius exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedIntervalModulusUp.

Chapter 9005

A Concrete Naming Certificate for $\mathsf{BishopCauchyReblocking}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{BishopCauchyReblocking}^{\uparrow}$ packet records the finite L10 handoff by which a displayed regular rational Cauchy name is reblocked before a real-name consumer may inspect its endpoint. It sits beside the general reblocking carrier of chapter 7649, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the finite displayed rows of the Bishop-style reblocking route; it does not assert quotient equality of real names, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9005.1 (BishopCauchyReblocking carrier)**.** *A* $\mathsf{BishopCauchyReblocking}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, W, K, T, D, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regular Cauchy readings are being reblocked.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the source observations consumed by the reblocking schedule.*
3. *K is the finite block schedule row, listing the displayed block endpoints and their monotone order.*
4. *T is the Bishop tail-control row, assigning each L10 tolerance request to a displayed block after which all later reads remain inside the required dyadic bound.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, comparisons, and arithmetic cells used by T.*
6. *L is the L10 handoff row, stating that a Real-facing consumer may read the reblocked name only after R, W, K, T, D are visible.*
7. *H records componentwise* hsame *transport for the source, window, schedule, tail-control, tolerance, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through $R, W, K, T, D, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyReblocking}^{\uparrow}$, $\mathsf{CauchyReblocking}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyReblocking}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, K, T, D, L, H, C, P, N$. No coordinate stores a completed real, a quotient representative, a selected limit, a countable-choice schedule, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 9005.2 (BishopCauchyReblocking NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyReblocking}^{\uparrow}$ *carrier $B = (R, W, K, T, D, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyReblocking}^{\uparrow}}$ *surface consists exactly of carrier admission, regular-source preservation, finite-window exposure, block-schedule exactness, Bishop tail-control exactness, dyadic tolerance readback, L10 handoff gating, transport, replay, provenance, and local naming. A downstream* $\mathsf{Real}^{\uparrow}$ *consumer may inspect the reblocked name only through those rows and receives no quotient equality, selected endpoint, countable choice, trichotomy branch, or ambient real completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyReblockingUp.BishopCauchyReblockingCarrier namecert obligations`

Proof. Project the carrier of Definition 9005.1. The rows $R, W, K, T, D, L, H, C, P, N$ display the source regular sequence, finite stream window, block schedule, tail control, dyadic tolerance ledger, L10 handoff, transport, replay, provenance, and local naming.

Regular-source preservation and finite-window exposure are forced by R and W. The reblocking schedule cannot be read until the same source observations have been displayed through the stream-name row.

Block-schedule exactness and Bishop tail-control exactness are supplied by K and T. The dyadic row D records the tolerance cells used by the tail control, and the handoff row L requires those displayed rows before any Real-facing read.

The remaining rows transport, replay, cite, and name the same route. Since the carrier has no coordinate for a completed endpoint, quotient representative, selected limit, choice schedule, trichotomy branch, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 9005.3 (BishopCauchyReblocking L10 handoff)**.** *Every L10-facing read exported by an accepted* $\mathsf{BishopCauchyReblocking}^{\uparrow}$ *carrier factors through*

$$R, W, K, T, D, L, H, C, P, N.$$

Thus the handoff to $\mathsf{Real}^{\uparrow}$ *is a finite Bishop-style reblocking of the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row through a displayed* $\mathsf{StreamName}^{\uparrow}$ *window, finite block schedule, dyadic tolerance ledger, componentwise transport, replay, provenance, and local naming. It is not a separate completion, selected host limit, quotient equality of Cauchy names, or countable-choice principle.*

Proof. Start with an accepted carrier $B = (R, W, K, T, D, L, H, C, P, N)$. The only source of rational Cauchy data is R, and the only displayed observations of that source are exposed by W.

The block schedule K and tail-control row T choose the finite reblocking route for each L10 tolerance request. The dyadic tolerance row D records the comparisons consumed by that route, and the handoff row L blocks a Real-facing read until the route is visible.

Transport and replay through H and C preserve the same rows, while P and N record provenance and local naming. Therefore every L10-facing read factors through the stated finite packet and cannot acquire an independent completion or selected endpoint. □

Falsifiable prediction: An accepted $\mathsf{BishopCauchyReblocking}^{\uparrow}$ read is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ finite window, finite block schedule, Bishop tail-control row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, L10 handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: cauchy-reblocking, streamname, regseqrat, dyadicratcore, real: BishopCauchyReblocking is not bijective to any listed sibling because it binds a regular rational source, finite stream window, block schedule, Bishop tail-control row, dyadic tolerance ledger, and L10 handoff gate into one Real-facing NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyReblocking}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyReblocking}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite window, finite block schedule, Bishop tail-control row, DyadicRatCore tolerance ledger, L10 handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyReblocking}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9005.1, Definition 9005.2, and Definition 9005.3 seed the finite Bishop-style Cauchy reblocking handoff before any Real seal is consumed.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyReblocking}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of Cauchy names, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-source preservation, finite-window exposure, block-schedule exactness, Bishop tail-control exactness, dyadic tolerance readback, L10 handoff gating, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyReblockingUp.

Chapter 9006

A Concrete Naming Certificate for $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$

The $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ packet realizes the prime-skew boundary-defect route of chapter 9249 as finite NameCert data. It records an accepted off-line zero packet, a displayed prime-visible skew row, boundary-defect readback, zero-defect incompatibility, refusal, transport, replay, provenance, and local naming. The packet consumes the prime-spectral fingerprint carrier of chapter 9000 through row Q before exposing the boundary-defect ledger. The packet is a falsifiable carrier for the route; it does not prove RH, construct zeta zeros, import complex analysis beyond displayed rows, or identify a global prime-spectral object.

The ledger consumes the prime-skew boundary packet of chapter 8848, the prime spectral fingerprint of chapter 9000, and the prime carrier of chapter 64 before its off-line-zero refusal row is public.

Definition 9006.1 (PrimeSkewDefectLedger carrier)**.** *A* $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Z, P, B, I, F, H, C, Q, R, N)$$

with the following displayed rows:

1. *Z is the accepted off-line zero packet row, carrying the displayed* $\mathsf{OffLineZero}$ *source data.*
2. *P is the prime-visible skew row, recording the displayed* $\mathsf{PrimeSkew}$ *evidence attached to the zero packet.*
3. *B is the boundary-defect ledger row, listing the local boundary layers on which the prime-visible skew is read.*
4. *I is the zero-defect incompatibility row, recording that an accepted non-trivial zero atom cannot coexist with the displayed boundary defect.*
5. *F is the falsifiable refusal row, declaring that an accepted off-line zero packet with no displayed prime-skew row refutes this carrier.*
6. *H records componentwise* hsame *transport for the zero, skew, boundary-defect, incompatibility, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the off-line zero packet through the prime-skew row, boundary-defect row, incompatibility row, refusal row, transport row, provenance row, and local naming row.*

8. *Q is the prime-spectral fingerprint row, naming the displayed fingerprint consumed by the ledger without claiming a complete spectral reconstruction.*
9. *R is the* Pkg *provenance over* $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$, $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimeSkewDefectLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, P, B, I, F, H, C, Q, R, N$. It has no coordinate for an unshown zero, a hidden prime-skew witness, a chosen complex-analytic model, a critical-line oracle, or an ambient proof of RH.

Lean4 (statement) `BEDC.Derived.PrimeSkewDefectLedgerUp`

Theorem 9006.2 (PrimeSkewDefectLedger NameCert obligations)**.** *For every accepted* $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ *carrier $D = (Z, P, B, I, F, H, C, Q, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{PrimeSkewDefectLedger}^{\uparrow}}$ *surface consists exactly of off-line zero admission, prime-skew display, boundary-defect ledger exposure, zero-defect incompatibility, falsifiable refusal, componentwise transport, displayed replay, prime-spectral fingerprint citation, provenance, and local naming. The chapter therefore packages the off-line zero, prime-visible skew, and local-global defect incompatibility rows as one finite ledger without claiming the Riemann Hypothesis.*

Proof. Project the carrier of Definition 9006.1. Its rows display the off-line zero packet, prime-visible skew, boundary-defect ledger, zero-defect incompatibility, falsifiable refusal, transport, replay, fingerprint, provenance, and local naming.

The zero row Z cannot be accepted alone. The prime-skew row P and boundary-defect row B expose the local-global defect route, while I records the incompatibility used by the refutation interface.

The refusal row F makes the carrier falsifiable: if an accepted off-line zero packet omits a displayed prime-skew row, the packet is outside the ledger. The rows H, C, Q, R, N transport, replay, cite, source, and name the same finite evidence.

Since no coordinate stores a hidden zero, chosen analytic model, critical-line oracle, or proof of RH, the listed rows exhaust the local NameCert obligations. □

Theorem 9006.3 (PrimeSkewDefectLedger falsifiable boundary)**.** *Any accepted off-line zero packet that lacks a displayed prime-skew row refutes the* $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ *carrier. Equivalently, every accepted ledger route from Z to the boundary-defect row must pass through*

$$Z, P, B, I, F, H, C, Q, R, N$$

before the local NameCert surface may be read.

Proof. Read an accepted carrier $D = (Z, P, B, I, F, H, C, Q, R, N)$. The source zero packet is Z, but the carrier admits it only as part of the ten-row ledger.

The prime-skew row P is the required displayed witness connecting the off-line zero packet to the boundary-defect ledger B. The incompatibility row I and refusal row F then state the boundary at which the ledger can be falsified.

Transport, replay, fingerprint citation, provenance, and local naming pass through H, C, Q, R, N. Therefore a zero packet without P cannot be an accepted $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ packet; it refutes the carrier surface rather than supplying a hidden route. □

Falsifiable prediction: Any accepted off-line zero packet with no displayed PrimeSkew row refutes the $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ chapter.

Independence witness: prime, zeta, complex, critical-line-witness, prime-spectral-fingerprint: $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries off-line zero admission, prime-skew display, boundary-defect ledger exposure, zero-defect incompatibility, falsifiable refusal, transport, replay, prime-spectral fingerprint citation, provenance, and local naming as one finite packet.

The Lean follow-up for this chapter must provide `Nontrivial` and `FieldFaithful` instances in `lean4/BEDC/Derived/PrimeSkewDefectLedgerUp/TasteGate.lean`; the present marker records the carrier name that the TasteGate file will inhabit.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeSkewDefectLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeSkewDefectLedger}^{\uparrow}$ packet is generated from off-line zero admission, prime-skew display, boundary-defect ledger exposure, zero-defect incompatibility, falsifiable refusal, componentwise hsame transport, displayed Cont replay, prime-spectral fingerprint citation, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeSkewDefectLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9006.1, Definition 9006.2, and Definition 9006.3 seed the finite carrier for the off-line zero, prime-visible skew, and defect-incompatibility route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeSkewDefectLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove RH, construct zeta zeros, close complex analysis, provide critical-line witnesses, complete the prime-spectral fingerprint, claim Lean TasteGate coverage, or bridge check the route.

Upgrade path: Obligation closure requires checked off-line zero admission, prime-skew display, boundary-defect ledger exposure, zero-defect incompatibility, falsifiable refusal, transport, replay, fingerprint citation, provenance, local naming, and the required Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.PrimeSkewDefectLedgerUp.

Chapter 9007

A Concrete Naming Certificate for $\mathsf{RegularSequenceSubsequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularSequenceSubsequence}^{\uparrow}$ packet records the finite route by which a regular sequence remains regular after a displayed increasing subsequence schedule is read. It sits over the regular-sequence source of chapter 5965, the finite observation windows of chapter 126, the rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names only the subsequence-stability route; it does not choose a host subsequence, identify quotient sequence names, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9007.1 (RegularSequenceSubsequence carrier)**.** *A* $\mathsf{RegularSequenceSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, K, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row, retaining the source regular-sequence classifier and its admitted observation windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the source observations from which the subsequence is read.*
3. *K is the increasing subsequence schedule row. It records finite monotone index windows and does not select an infinite host function outside the packet.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic budget used to compare source and subsequence windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations of the selected windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed by Real-facing or Completion-facing readers after S, W, K, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for source, stream-window, subsequence schedule, dyadic tolerance, rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a subsequence request through S, W, K, D, Q, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceSubsequence}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, K, D, Q, E, H, C, P, N$. It has no coordinate for a host subsequence selector, quotient sequence equality, selected limit, countable choice, or ambient completeness.

Theorem 9007.2 (RegularSequenceSubsequence NameCert obligations)**.** *For every accepted* RegularSequenceSubse *carrier*

$$R = (S, W, K, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularSequenceSubsequence}^{\uparrow}}$ *surface consists exactly of source regular-sequence admission, finite StreamName window exposure, increasing subsequence schedule display, dyadic tolerance readback, RegSeqRat rational readback, terminal Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Each obligation is read through the finite route*

$$\mathsf{RegularSequenceSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Proof. Project the carrier of Definition 9007.1. The source row S and finite-window row W are the only rows from which a subsequence request may begin.

The schedule row K records monotone finite index windows, while D supplies the dyadic comparison budget for those windows. Thus a subsequence consumer receives a displayed finite route rather than a host-selected infinite map.

The rational readback row Q is downstream of S, W, K, D, and the Real seal row E is downstream of Q. Componentwise transport through H and replay through C preserve that order, while P, N record provenance and local naming.

Since the tuple contains no coordinate for quotient sequence equality, selected limits, countable choice, host subsequence selectors, or ambient completeness, no NameCert obligation can be discharged outside the displayed packet rows. □

Theorem 9007.3 (RegularSequenceSubsequence regularity preservation)**.** *Let $R = (S, W, K, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSubsequence}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumer that reads the subsequence as regular must first expose the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source row S, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the increasing schedule row K, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q. Therefore regularity preservation is a finite readback route from the source regular sequence to the subsequence seal, not a host theorem about arbitrary subsequences.*

Proof. Start with the obligation surface of Definition 9007.2. It places S, W, K, D, Q before every Real-facing or completion-facing read of the subsequence.

The source regularity is read from $\mathsf{RegularSequenceSpace}^{\uparrow}$ through S, and the observed subsequence windows are read through $\mathsf{StreamName}^{\uparrow}$ and K. The dyadic row D supplies the tolerance budget that makes the selected windows compatible with the regular rational readback.

Finally the $\mathsf{RegSeqRat}^{\uparrow}$ row Q exposes the rational regularity data before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be consumed. The support rows H, C, P, N transport, replay, package, and name that same finite route, so they cannot add a host subsequence selector, quotient equality, selected limit, countable choice, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceSubsequence}^{\uparrow}$ packet is generated from a RegularSequenceSpace source row, StreamName finite windows, an increasing subsequence schedule, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9007.1, Definition 9007.2, and Definition 9007.3 seed the finite subsequence-stability route over RegularSequenceSpaceUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host subsequence selector, quotient sequence equality, selected limits, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked source admission, finite-window exposure, subsequence-schedule display, dyadic tolerance readback, RegSeqRat rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceSubsequenceUp.

Chapter 9008

A Concrete Naming Certificate for $\mathsf{EventualMap}^{\uparrow}$

Following Kelley and Bourbaki on eventual maps of directed sets and filters, the $\mathsf{EventualMap}^{\uparrow}$ packet records the finite route by which a tail-indexed source surface is transported into a target tail surface. It sits over the directed-index substrate of chapter 1607, the Moore–Smith subnet route of chapter 5098, the Frechet filter boundary of chapter 1564, the filter-base surface of chapter 1601, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. The chapter names eventuality only as displayed cofinal-tail data; it does not import host functions, quotient filters, selected representatives, or countable choice.

Definition 9008.1 (EventualMapUp carrier)**.** *An* $\mathsf{EventualMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, T, M, C, W, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the source tail row, naming the inspected directed-set, filter-base, or stream-index tail requests on the domain side.*
2. *T is the target tail row, naming the inspected directed-set, filter-base, or stream-index tail requests on the codomain side.*
3. *M is the monotone eventual-map row, recording only the displayed comparisons showing that target-tail refinement pulls source-tail requests forward.*
4. *C is the cofinal ledger, assigning to each displayed target tail a source tail whose image is eventually inside it.*
5. *W is the finite-window pullback row, recording the pulled-back StreamName and RegSeqRat windows used by Real and Completion consumers.*
6. *R is the readback row, exposing the target-side observation only after S, T, M, C, W have been displayed.*
7. *H records componentwise* hsame *transport for source tails, target tails, monotone-eventual comparisons, cofinal ledgers, finite-window pullbacks, readbacks, replay, provenance, and local naming.*
8. *K records displayed* Cont *replay from every eventual-map request through $S, T, M, C, W, R, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{EventualMap}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithSubnet}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EventualMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for equality of host functions, selected representatives of filters or nets, quotient filters, choice-indexed cofinal maps, or an ambient eventual-equality relation outside the displayed BHist *packet.*

Theorem 9008.2 (EventualMapUp NameCert obligations)**.** *For every accepted* $\mathsf{EventualMap}^{\uparrow}$ *carrier*

$$E = (S, T, M, C, W, R, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EventualMap}^{\uparrow}}$ *surface consists exactly of these finite obligations:*

1. *carrier admission for source-tail data, target-tail data, monotone eventual-map comparisons, the cofinal ledger, finite-window pullbacks, readback, transport, replay, provenance, and local naming;*
2. *monotone exactness, requiring every exported refinement comparison to be a displayed row of M between addresses already named by S and T;*
3. *cofinal exactness, requiring each target-tail request to read its source-tail witness through the ledger C before any downstream consumer can use it;*
4. *tail-window pullback stability, requiring the window row W to preserve the same StreamName and RegSeqRat request under the displayed tail pullback;*
5. *readback handoff, requiring R to expose the target-side observation only after S, T, M, C, W are visible;*
6. *transport and replay stability under H and K, with no hidden row added by structural replay;*
7. *provenance exhaustion through P, N, so the packet is sourced and locally named exactly as an* $\mathsf{EventualMap}^{\uparrow}$ *carrier;*
8. *refusal of host function equality, selected representatives, quotient filters, countable choice, and ambient eventual-equality predicates.*

Proof. Project the carrier of Definition 9008.1. The source and target tail rows are S and T, while the monotone eventual-map comparisons and cofinal witnesses are M and C.

The window and readback rows are downstream. The row W records finite pullbacks of the displayed StreamName and RegSeqRat windows, and R reads the target observation only after the pulled-back tail data have been exposed.

The structural rows H, K, P, N preserve this route componentwise. They transport, replay, source, and name the same finite packet, so they cannot add host function equality, selected representatives, quotient filters, a choice-indexed cofinal map, or an ambient eventual-equality predicate. □

Theorem 9008.3 (EventualMapUp composition handoff)**.** *Let*

$$E_{01} = (S_0, S_1, M_{01}, C_{01}, W_{01}, R_{01}, H_{01}, K_{01}, P_{01}, N_{01})$$

and

$$E_{12} = (S_1', S_2, M_{12}, C_{12}, W_{12}, R_{12}, H_{12}, K_{12}, P_{12}, N_{12})$$

be accepted EventualMap$^{\uparrow}$ *packets whose middle tail rows agree through a displayed* hsame *transport. Their composite is exported only by reading the shared finite cofinal ledgers* C_{01}, C_{12} *and the visible tail-window pullbacks* W_{01}, W_{12}*. No composition step may replace those rows by host function composition, quotient-filter equality, a selected subnet representative, or countable choice.*

Proof. First expose the middle-tail agreement. The only bridge between the target row of E_{01} and the source row of E_{12} is the displayed hsame transport on the middle tail surface.

Next read the cofinal data in order. A final target-tail request is pulled back through C_{12} to a middle-tail request and then through C_{01} to a source-tail request. The monotone rows M_{12} and M_{01} supply only the displayed refinement comparisons needed for this two-step pullback.

Finally compose the finite windows. The rows W_{12} and W_{01} carry the visible StreamName and RegSeqRat pullbacks, and the readback rows R_{12} and R_{01} remain downstream of those windows. Transport, replay, provenance, and local naming preserve the same finite ledgers, so no host-level composition or choice principle is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{EventualMap}^{\uparrow}}$**.**

Constructive story: An EventualMap$^{\uparrow}$ packet is generated from source and target tail rows, monotone eventual-map comparisons, a cofinal ledger, finite-window pullbacks, readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EventualMap}^{\uparrow}}$, seed)

Scope closed: Definition 9008.1, Definition 9008.2, and Definition 9008.3 seed the finite eventual-map interface for directed tails, filters, StreamName windows, and RegSeqRat readbacks.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EventualMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, selected subnet or filter representatives, quotient filters, countable choice, ambient eventual equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone exactness, cofinal exactness, tail-window pullback stability, readback handoff, transport, replay, provenance, and local naming for BEDC.Derived.EventualMapUp.

Chapter 9009

A Concrete Naming Certificate for $\mathsf{OrliczSpace}^{\uparrow}$

Following Orlicz and the classical presentations of Krasnoselskii–Rutickii and Rao–Ren, the $\mathsf{OrliczSpace}^{\uparrow}$ packet records the Luxemburg-gauge construction of an Orlicz function space as displayed BEDC NameCert data. It sits after the functional-analysis carrier of chapter 94, the norm ledger of chapter 89, the measure carrier of chapter 92, the terminal real-name seal of chapter 38, and the vector-space surface of chapter 47. The packet names only the finite rows by which a Young-function ledger, measurable-function row, modular readback, Luxemburg gauge, vector operations, transport, replay, provenance, and local NameCert are exposed; it does not import a host Banach-space object, quotient representatives, selected almost-everywhere classes, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9009.1 (Orlicz space carrier)**.** *An* $\mathsf{OrliczSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, \Phi, M, F, I, L, V, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the measure-domain row. It records the measurable source and its* $\mathsf{Measure}^{\uparrow}$ *NameCert route.*
2. *Φ is the Young-function row. It displays the convex growth ledger, null-at-zero read, and finite real-valued comparison data used by the modular.*
3. *M is the measurable-function admission row. It records the function names inspected by the packet and keeps their measurable status visible before any norm read.*
4. *F is the finite real-readback row. It routes every scalar comparison through* $\mathsf{Real}^{\uparrow}$ *and the local functional-analysis surface.*
5. *I is the modular row. It records the displayed integral-read of $\Phi(|f|)$ as a measure-facing ledger, not as a host integral over quotient classes.*
6. *L is the Luxemburg-gauge row. It records the finite scale test for the value*

$$\inf\{\lambda > 0 : I(f/\lambda) \leq 1\}$$

only through displayed positive-scale and modular rows.

7. *V is the vector-space row. It records addition and scalar multiplication requests as* $\mathsf{VecSpace}^{\uparrow}$ *reads that preserve* A, Φ, M, F, I, L.
8. *R is the norm-readback row. It names the* $\mathsf{Norm}^{\uparrow}$ *consumer route through the Luxemburg gauge and exposes no external Banach-space completion.*
9. *H records componentwise* hsame *transport for domain, Young-function, measurable-function, real-readback, modular, Luxemburg-gauge, vector-space, norm-readback, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a consumer request through* $A, \Phi, M, F, I, L, V, R, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{OrliczSpace}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{OrliczSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, \Phi, M, F, I, L, V, R, H, C, P, N$. *It has no coordinate for quotient measurable-function classes, selected representatives, countable choice, host almost-everywhere equality, Banach-space completeness, Lean verification, or bridge checking.*

Theorem 9009.2 (Orlicz space NameCert obligations)**.** *For every accepted* $\mathsf{OrliczSpace}^{\uparrow}$ *carrier*

$$O = (A, \Phi, M, F, I, L, V, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OrliczSpace}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the measure-domain row, Young-function row, measurable-function row, finite real-readback row, modular row, Luxemburg-gauge row, vector-space row, norm-readback row, componentwise transport, displayed replay, provenance, and local naming;*
2. *modular exposure, requiring each consumer of a Luxemburg value to first read* A, Φ, M, F, I;
3. *positive-scale visibility, requiring the row L to display the finite positive-scale tests used by the Luxemburg gauge before any* $\mathsf{Norm}^{\uparrow}$ *readback is exported;*
4. *vector-operation stability, requiring V to keep addition and scalar multiplication inside the same* A, Φ, M, F, I, L *route;*
5. *norm-reader boundary, requiring every* $\mathsf{Norm}^{\uparrow}$ *or functional-analysis consumer to pass through R after the modular and vector rows have been displayed;*
6. *ledger nonescape, excluding quotient measurable-function classes, selected representatives, countable choice, host almost-everywhere equality, Banach-space completeness, Lean verification, and bridge checking.*

No Orlicz-space NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 9009.1. The tuple lists exactly the domain, Young-function, measurable-function, real-readback, modular, Luxemburg-gauge, vector-space, norm-readback, transport, replay, provenance, and local naming rows.

The modular obligation is the row I. It can be read only after A, Φ, M, F are displayed, so the integral expression remains a measure-facing ledger over named functions and real comparisons.

The Luxemburg obligation is the row L. Its positive-scale tests are displayed finite reads of the modular row, hence a norm consumer cannot replace them by a host infimum over quotient representatives.

The vector and norm-reader obligations are the rows V and R. They keep addition, scalar multiplication, and $\mathsf{Norm}^{\uparrow}$ readback attached to the same modular route, while H, C, P, N transport, replay, source, and name that route.

The remaining exclusions follow from the carrier inventory. Since quotient measurable-function classes, selected representatives, countable choice, host almost-everywhere equality, Banach-space completeness, Lean verification, and bridge checking do not occur among the coordinates, none of those data can be exported by the local certificate. □

Theorem 9009.3 (Orlicz space Luxemburg route)**.** *Every* $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, *or vector-space consumer of an accepted* $\mathsf{OrliczSpace}^{\uparrow}$ *packet must read the Luxemburg gauge through the displayed route*

$$A, \Phi, M, F, I \longmapsto L \longmapsto V, R, H, C, P, N.$$

Thus the norm-facing value is a local NameCert route through measure data, Young-function data, finite real readback, modular exposure, and positive-scale tests. It is not a host quotient-space norm, selected almost-everywhere representative, countable-choice family of scales, ambient Banach-space completion, or external $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier and expose A, Φ, M, F, I. These rows display the measure source, Young-function ledger, admitted measurable functions, real comparisons, and modular read before a Luxemburg request is available.

Read the Luxemburg row L. It records the positive-scale test surface over the modular row, so the gauge remains a finite route through the displayed carrier rather than an external norm formula.

The vector row V and norm-readback row R are downstream of L. They let vector-space and norm consumers inspect the same route, while H, C, P, N preserve componentwise transport, replay, provenance, and local naming.

The refused endpoints are absent from the carrier. Therefore a consumer can receive only the displayed Luxemburg route and cannot extract quotient representatives, selected almost-everywhere classes, countable-choice scales, Banach-space completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{OrliczSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{OrliczSpace}^{\uparrow}$ packet is generated from a measure-domain row, a Young-function row, a measurable-function row, finite real readback, a modular row, a Luxemburg-gauge row, a vector-space row, a norm-readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrliczSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9009.1, Definition 9009.2, and Definition 9009.3 seed the Orlicz-space route from measure and Young-function data to Luxemburg-gauge norm readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrliczSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient measurable-function classes, selected representatives, countable choice, host almost-everywhere equality, Banach-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modular exposure, positive-scale visibility, vector-operation stability, norm-reader boundary, transport, replay, provenance, and local naming for BEDC.Derived.OrliczSpaceUp.

Chapter 9010

A Concrete Naming Certificate for DyadicMeshRefinement$^{\uparrow}$

Following Bishop and Bridges on constructive interval refinements, the DyadicMeshRefinement$^{\uparrow}$ packet records the finite route by which one dyadic mesh is refined before any regular-Cauchy or Real-facing consumer reads the resulting windows. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite mesh-refinement data; it does not assert quotient interval equality, select a limiting point, prove Real completeness, supply open-cover compactness, or exhibit a standard bridge.

Definition 9010.1 (Dyadic mesh refinement carrier)**.** *A* DyadicMeshRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, R, E, K, B, D, H, C, P, N)$$

with the following displayed rows:

1. *G is the parent dyadic grid row, read through* DyadicRatCore$^{\uparrow}$ *endpoints and mesh widths.*
2. *R is the refinement grid row, whose cells are again dyadic and whose indexing is exposed by finite* StreamName$^{\uparrow}$ *windows.*
3. *E is the endpoint-preservation row, recording that the first and last endpoints of the refinement are the displayed parent endpoints.*
4. *K is the cell-containment row, assigning each refinement cell to a displayed parent cell without quotienting intervals by host equality.*
5. *B is the mesh-bound comparison row, comparing the refinement mesh width with the parent bound through dyadic arithmetic.*
6. *D is the dyadic tolerance readback row, carrying the requested tolerance to the* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumers that inspect refined windows.*
7. *H records componentwise* hsame *transport for the parent grid, refinement grid, endpoint, containment, mesh-bound, tolerance, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a consumer request through parent-grid inspection, refinement-grid inspection, endpoint preservation, cell containment, mesh-bound comparison, and dyadic tolerance readback.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshRefinement}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, R, E, K, B, D, H, C, P, N$. It has no coordinate for quotient interval equality, selected limiting data, ambient real completeness, open-cover compactness, or bridge checking.

Theorem 9010.2 (Dyadic mesh refinement NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshRefinement}^{\uparrow}$ *carrier $M = (G, R, E, K, B, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, parent dyadic-grid exposure, refinement-grid exposure, endpoint preservation, refinement-cell containment in parent cells, mesh-bound comparison, dyadic tolerance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every regular-Cauchy or Real-facing consumer must read the finite route from $\mathsf{DyadicRatCore}^{\uparrow}$ *through* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before it reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient interval equality, selected limit, Real completeness, open-cover compactness, or standard bridge.*

Proof. Project the accepted carrier. The tuple displays exactly the parent grid G, refinement grid R, endpoint row E, containment row K, mesh-bound row B, dyadic tolerance row D, transport row H, replay row C, provenance row P, and local naming row N.

Endpoint preservation and cell containment are read from E and K. These rows compare displayed dyadic endpoints and finite cells, so the refinement is witnessed by $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic rather than by host interval equality.

The mesh-bound and tolerance rows B, D are the only route to a regular rational or Real-facing read. A consumer therefore passes through the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before using the $\mathsf{Real}^{\uparrow}$ seal.

The support rows H, C, P, N transport, replay, package, and name that same finite route. Since no row carries quotient intervals, selected limiting data, Real completeness, open-cover compactness, or bridge evidence, none of those data are exported by the local certificate. □

Theorem 9010.3 (Dyadic mesh refinement sibling route)**.** *Every accepted* $\mathsf{DyadicMeshRefinement}^{\uparrow}$ *read used by interval, regular-Cauchy, compact-net, or Real-facing arguments factors through the sibling chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The parent and refinement grids are dyadic rows, the inspected windows are $\mathsf{StreamName}^{\uparrow}$ *rows, the rational observations are* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, and the terminal seal is a* $\mathsf{Real}^{\uparrow}$ *row. No consumer may bypass this chain by importing quotient interval equality, a selected limit, ambient real completeness, open-cover compactness, or a standard bridge.*

Proof. Start with the carrier of Definition 9010.1. The parent grid, refinement grid, endpoint preservation, containment, and mesh-bound comparison are all dyadic rows, so the first sibling dependency is chapter 291.

The refinement is read through finite windows before regular rational observations are consumed. Thus the next two dependencies are the $\mathsf{StreamName}^{\uparrow}$ surface of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301.

Finally the Real-facing consumer reads only the terminal seal named by chapter 38. The obligation surface of Definition 9010.2 keeps all support rows inside H, C, P, N, so the sibling route

cannot export quotient intervals, selected limits, Real completeness, open-cover compactness, or bridge evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshRefinement}^{\uparrow}$ packet is generated from a parent dyadic grid, a refinement grid, endpoint preservation, refinement-cell containment, mesh-bound comparison, dyadic tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9010.1, Definition 9010.2, and Definition 9010.3 seed the finite dyadic refinement route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient interval equality, selected limits, Real completeness, open-cover compactness, Lean verification, bridge checking, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint preservation, cell containment, mesh-bound comparison, dyadic tolerance readback, transport/replay/provenance stability, and sibling-route non-escape for BEDC.Derived.DyadicMeshRefinementUp.

Chapter 9011

A Concrete Naming Certificate for $\mathsf{TraceClassOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{TraceClassOperator}^{\uparrow}$ packet records the standard trace-class operator surface as finite BEDC naming data. It sits over the Hilbert-space carrier of chapter 91, the bounded-linear-operator row of chapter 1644, the operator-ideal route of chapter 213, the spectral-measure window of chapter 298, the terminal Real row of chapter 38, and the regular rational readback of chapter 301. The packet names only the displayed trace-class route through Hilbert, bounded-operator, ideal, spectral, Real, and rational rows; it does not assert compact-operator completeness, Schatten duality, arbitrary spectral integration, choice of an eigenbasis, host Hilbert-space separability, or ambient operator-algebra completeness.

Definition 9011.1 (TraceClassOperator carrier)**.** *A* $\mathsf{TraceClassOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H, B, I, S, R, Q, A, X, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, retaining the inner-product, norm, completion, and local naming data visible to the packet.*
2. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row, exposing a bounded operator over the same Hilbert source.*
3. *I is the* $\mathsf{OperatorIdeal}^{\uparrow}$ *row, recording the ideal-membership route through which the bounded operator is inspected.*
4. *S is the* $\mathsf{SpectralMeasure}^{\uparrow}$ *window row, available only as a finite spectral readback surface and not as unrestricted functional calculus.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *trace-value row, carrying the terminal real-number seal for trace reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational approximation row used to display finite trace windows before the Real seal is consumed.*
7. *A records the finite trace-window admission row tying B, I, S, Q, R to one inspected trace-class request.*
8. *X records componentwise* hsame *transport for $H, B, I, S, R, Q, A, C, P, N$.*
9. *C records displayed* Cont *replay from a trace-class query through the Hilbert, bounded-operator, ideal, spectral, rational, Real, and admission rows.*

10. *P is the* Pkg *provenance over* TraceClassOperator$^\uparrow$, Hilbert$^\uparrow$, BoundedLinearOperator$^\uparrow$, OperatorIdeal$^\uparrow$, SpectralMeasure$^\uparrow$, Real$^\uparrow$, RegSeqRat$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{TraceClassOperator}^\uparrow}$ *row.*

The classifier compares accepted packets only through $H, B, I, S, R, Q, A, X, C, P, N$. It has no coordinate for arbitrary eigenbasis choice, full compact-operator theory, unrestricted spectral integration, Schatten duality, host trace-norm completion, or ambient operator-algebra completeness.

Lean4 (statement) `BEDC.Derived.TraceClassOperatorUp`

Theorem 9011.2 (TraceClassOperator NameCert obligations)**.** *For every accepted* TraceClassOperator$^\uparrow$ *carrier*

$$T = (H, B, I, S, R, Q, A, X, C, P, N),$$

the local NameCert$_{\text{TraceClassOperator}^\uparrow}$ *surface consists exactly of Hilbert-source admission, bounded-operator admission, operator-ideal membership, finite spectral-window readback, rational trace approximation, terminal Real seal, finite trace-window admission, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 9011.1. The Hilbert row H is the source for the inner-product and norm data, while B is the only bounded-operator row exposed by the packet.

The ideal row I and spectral row S are read before any trace value is consumed. They place the inspected operator inside the finite operator-ideal and spectral-window routes rather than inside an ambient compact-operator algebra.

The rational trace row Q supplies the finite approximation surface, and the Real row R is visible only after Q and the trace-window admission row A have been displayed. Thus the trace read is a finite route through H, B, I, S, Q, R, A.

Finally X, C, P, N transport, replay, source, and name the same route. Since the tuple contains no coordinate for eigenbasis choice, unrestricted spectral integration, Schatten duality, trace-norm completion, or ambient operator-algebra completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 9011.3 (TraceClassOperator finite trace window)**.** *Every Real-valued trace read exported by an accepted* TraceClassOperator$^\uparrow$ *packet first exposes the* Hilbert$^\uparrow$ *source row, the* BoundedLinearOperator$^\uparrow$ *boundedness row, the* OperatorIdeal$^\uparrow$ *membership row, the* SpectralMeasure$^\uparrow$ *finite-window row, and the* RegSeqRat$^\uparrow$ *rational trace approximation row. The exported trace is therefore a finite NameCert window into a trace-class request, not a host theorem about arbitrary compact operators, spectral decompositions, or operator-algebra completions.*

Proof. Start with the NameCert obligations of Definition 9011.2. They place the Hilbert, bounded-operator, ideal, spectral, rational, Real, and admission rows before any public trace read.

The finite trace-window row A binds the bounded operator B to the ideal route I and the spectral readback S. The rational row Q then exposes the finite trace approximation consumed by the Real row R.

Componentwise transport and replay preserve that same row order. A downstream spectral, quantum-state, or operator-ideal consumer therefore receives only $H, B, I, S, Q, R, A, X, C, P, N$ and cannot read arbitrary eigenbasis choice, unrestricted spectral integration, Schatten duality, trace-norm completion, or compact-operator completeness from the packet. □

Closure status for NameCert$_{\text{TraceClassOperator}^\uparrow}$.

Constructive story: A TraceClassOperator$^{\uparrow}$ packet is generated from Hilbert, bounded-linear-operator, operator-ideal, spectral-measure, Real, RegSeqRat, finite trace-window, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{TraceClassOperator}\uparrow}$, seed)

Scope closed: Definition 9011.1, Definition 9011.2, and Definition 9011.3 seed the finite trace-class operator route over HilbertUp, BoundedLinearOperatorUp, OperatorIdealUp, SpectralMeasureUp, RealUp, RegSeqRatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{TraceClassOperator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim compact-operator completeness, Schatten duality, arbitrary spectral integration, eigenbasis choice, host Hilbert-space separability, Lean verification, bridge checking, or ambient operator-algebra completeness.

Upgrade path: Obligation closure requires checked Hilbert-source admission, bounded-operator admission, operator-ideal membership, finite spectral-window readback, rational trace approximation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TraceClassOperatorUp.

Chapter 9012

A Concrete Naming Certificate for $\mathsf{RegularVariation}^{\uparrow}$

Following Karamata and Bingham–Goldie–Teugels, the $\mathsf{RegularVariation}^{\uparrow}$ packet records regular variation as finite asymptotic-comparison data rather than as a completed function space. It sits after the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the slowly varying sibling row of chapter 6724, and the dyadic tolerance discipline of chapter 291. The packet gives later asymptotic comparison chapters a named BEDC carrier while refusing ambient real completeness, quotient equality of real-valued streams, and host-selected limits.

Definition 9012.1 (Regular variation carrier)**.** *A* $\mathsf{RegularVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, L, R, M, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the positive* $\mathsf{Real}^{\uparrow}$ *scale stream whose inspected windows are exposed before any ratio comparison is read;*
2. *L is the slowly varying sibling row inherited from* $\mathsf{NameCert}_{\mathsf{SlowlyVaryingSequence}^{\uparrow}}$*;*
3. *R is the ratio-window readback row comparing the displayed values of S against the slowly varying component over the same finite window;*
4. *M records multiplication and division provenance through the visible* $\mathsf{Real}^{\uparrow}$ *algebra rows and the regular-rational readback supplied by* $\mathsf{RegSeqRat}^{\uparrow}$*;*
5. *O is the eventual order and tolerance comparison row, measured through dyadic and rational windows rather than by an ambient order oracle;*
6. *H records componentwise* hsame *transport for scale, slowly varying, ratio, algebra, order, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from the scale stream through the ratio and order windows;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularVariation}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{SlowlyVaryingSequence}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularVariation}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $S, L, R, M, O, H, C, P, N$.

Theorem 9012.2 (Regular variation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularVariation}^{\uparrow}}$ *surface consists exactly of carrier admission for* $Q = (S, L, R, M, O, H, C, P, N)$*, positive* $\mathsf{Real}^{\uparrow}$ *scale exposure through* S*, slowly varying sibling exposure through* L*, ratio-window readback through* R*, visible real algebra provenance through* M*, eventual order and tolerance comparison through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every Real or regular-sequence consumer must read the sibling routes of chapter 6724, chapter 301, and chapter 38 before it treats the packet as a regular-variation comparison. The certificate exports no host function space, no quotient stream equality, no selected real limit, no countable-choice selection of eventual bounds, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The scale stream, slowly varying row, ratio readback, algebra provenance, order comparison, transport, replay, provenance, and local name are the only visible coordinates.

Read the dependency rows in order. The slowly varying component is supplied by the sibling certificate, the regular rational windows provide finite readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after those displayed rows and the algebra provenance have been exposed.

Since the carrier has no coordinate for host functions, quotient stream equality, selected limits, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert obligation surface. □

Falsifiable prediction: A purported $\mathsf{RegularVariation}^{\uparrow}$ consumer is refuted if it reads a regular-variation ratio without displaying the positive RealUp scale stream, slowly varying row, ratio-window readback, real-algebra provenance, eventual order comparison, transport, replay, provenance, and local NameCert rows.

Independence witness: slowly-varying-sequence, regseqrat, real, dyadicratcore: the slowly varying sibling lacks the positive scale and ratio-window rows, RegSeqRatUp supplies regular rational readback without the asymptotic classifier, RealUp supplies the terminal seal without eventual comparison data, and DyadicRatCoreUp supplies tolerance rows without the regular-variation carrier. None is bijective to $\mathsf{RegularVariation}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularVariation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularVariation}^{\uparrow}$ packet is generated from a positive RealUp scale stream, a slowly varying sibling row, ratio-window readback, real-algebra provenance, eventual order and tolerance comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularVariation}^{\uparrow}}$, seed)

Scope closed: Definition 9012.1 and Definition 9012.2 seed the finite regular-variation NameCert surface over RealUp, RegSeqRatUp, SlowlyVaryingSequenceUp, RatUp, OrderUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, ambient RealUp completeness, quotient equality of real-valued streams, selected host limits, countable-choice eventual bounds, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ratio-window exactness, real-algebra provenance, eventual order comparison, transport, replay, provenance, and local naming for BEDC.Derived.RegularVariationUp.

Chapter 9013

A Concrete Naming Certificate for HyperbolicSectorOverlapLedger$^{\uparrow}$

The HyperbolicSectorOverlapLedger$^{\uparrow}$ packet realizes the sector-overlap branch of chapter 9463 as finite observer-expansion data. It sits beside the sector envelope of chapter 8546, the sector expansion packet of chapter 8685, the boundary-shadow transport route of chapter 6678, and the four-point metric discipline of chapter 6862. The carrier names overlap only through displayed sector, shadow, radius, observer-window, transport, replay, provenance, and local NameCert rows.

Definition 9013.1 (Hyperbolic sector-overlap ledger carrier)**.** *A* HyperbolicSectorOverlapLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, V, W, O, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the sector aperture row inherited from the sector-envelope and sector-expansion siblings;*
2. *B is the boundary-shadow row that records the boundary face inspected by the overlap comparison;*
3. *V is the visual-radius row measured through the existing hyperbolic visual and Gromov-metric readback routes;*
4. *W is the observer-window provenance row, naming the finite observer-expansion window in which the sector is read;*
5. *O is the overlap witness row, displaying the two sector windows and the finite boundary-shadow intersection data;*
6. *U is the TasteGate-facing row requiring Lean* Nontrivial HyperbolicSectorOverlapLedger$^{\uparrow}$ *and* FieldFaithful HyperbolicSectorOverlapLedger$^{\uparrow}$ *instances for the same displayed carrier fields;*
7. *H records componentwise* hsame *transport for aperture, boundary shadow, visual radius, overlap witness, observer window, TasteGate-facing evidence, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from sector aperture through boundary shadow, visual radius, observer window, and overlap witness;*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}$, $\mathsf{HyperbolicSectorEnvelope}^{\uparrow}$, $\mathsf{HyperbolicSectorExpar}$ $\mathsf{HyperbolicBoundaryShadowTransport}^{\uparrow}$, $\mathsf{GromovHyperbolicMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $A, B, V, W, O, U, H, C, P, N$.

Theorem 9013.2 (Hyperbolic sector-overlap ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorOverlap}}$ *surface consists exactly of carrier admission for $Q = (A, B, V, W, O, U, H, C, P, N)$, sector-aperture exposure through A, boundary-shadow exposure through B, visual-radius readback through V, observer-window provenance through W, overlap-witness exposure through O, TasteGate-facing nontriviality and field-faithfulness obligations through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every observer-expansion consumer must read the sibling routes of chapter 8546, chapter 8685, chapter 6678, and chapter 6862 before it treats an overlap row as accepted. The packet exports no completed hyperbolic boundary, no selected global sector, no host geometric intersection theorem, no physical observer law, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The aperture, boundary shadow, visual radius, observer window, overlap witness, TasteGate-facing row, transport, replay, provenance, and local name are exactly the visible coordinates.

Read the sibling routes in order. The sector envelope supplies the aperture surface, sector expansion supplies the expansion comparison, boundary-shadow transport supplies the transported shadow row, and the Gromov hyperbolic metric supplies the finite metric discipline for visual-radius readback.

The TasteGate-facing row records the Lean obligation to provide Nontrivial and FieldFaithful instances preserving the same fields. Since no carrier coordinate stores a completed boundary, selected global sector, host intersection theorem, physical observer law, or ambient Real completeness principle, the displayed rows exhaust the NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}$ read is refuted if an observer-expansion consumer obtains a sector-overlap witness without displaying sector aperture, boundary shadow, visual radius, observer-window provenance, overlap witness, TasteGate-facing nontriviality and field-faithfulness rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: hyperbolic-sector-envelope, hyperbolic-sector-expansion, hyperbolic-boundary-shadow-transport, gromov-hyperbolic-metric: the envelope sibling lacks an overlap witness, the expansion sibling lacks the two-sector ledger, the boundary-shadow transport sibling lacks sector aperture comparison, and the Gromov metric sibling supplies metric discipline without observer-sector overlap provenance. None is bijective to $\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}$ packet is generated from sector aperture, boundary-shadow, visual-radius, observer-window provenance, overlap-witness, TasteGate-facing nontriviality and field-faithfulness, hsame transport, Cont replay, Pkg provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}}$, seed)

Scope closed: Definition 9013.1 and Definition 9013.2 seed the finite sector-overlap ledger over the hyperbolic sector envelope, sector expansion, boundary-shadow transport, Gromov metric, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicSectorOverlapLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, a completed hyperbolic boundary, selected global sectors, host geometric intersection, physical observer dynamics, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sector-aperture exposure, boundary-shadow and visual-radius readback, overlap-witness exactness, observer-window provenance, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicSectorOverlapLedgerUp.

Chapter 9014

A Concrete Naming Certificate for $\mathsf{BishopSequenceSpace}^{\uparrow}$

Following Bishop and Bridges on sequence spaces over constructive real names, the $\mathsf{BishopSequenceSpace}^{\uparrow}$ packet records the finite L10 route by which a displayed stream-indexed family of regular rational names is read as a sequence-space object before any completion or Real-facing consumer inspects the terminal seal. It sits over the finite schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the source route of chapter 5965, and the terminal real-name seal of chapter 38. The packet names only displayed sequence-space rows; it does not assert quotient equality of real sequences, selected pointwise representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9014.1 (BishopSequenceSpace carrier)**.** *A* $\mathsf{BishopSequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, listing the finite sequence indices, precision requests, and observation windows opened by the consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, assigning each displayed window the regular rational data read from the scheduled sequence entry.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, comparisons, and finite arithmetic cells used to compare the scheduled reads.*
4. *Q is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source route, exposing the sequence-space admission row consumed by the Bishop packet.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, R, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, readback, tolerance, source-route, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a sequence-space request through S, R, D, Q, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient equality of real sequences, selected pointwise representatives, countable-choice schedules, trichotomy branches, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9014.2 (BishopSequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{BishopSequenceSpace}^{\uparrow}$ *carrier*

$$B = (S, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSequenceSpace}^{\uparrow}}$ *surface consists exactly of finite StreamName scheduling through S, RegSeqRat readback through R, DyadicRatCore tolerance accounting through D, RegularSequenceSpace source admission through Q, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9014.1. The schedule row S, readback row R, tolerance row D, source-route row Q, and seal row E are visible before the structural rows are consumed.

Read the sequence-space route in order. The $\mathsf{StreamName}^{\uparrow}$ row opens only the finite windows requested by the consumer, $\mathsf{RegSeqRat}^{\uparrow}$ reads those windows as regular rational data, and $\mathsf{DyadicRatCore}^{\uparrow}$ records the displayed tolerance comparisons.

The $\mathsf{RegularSequenceSpace}^{\uparrow}$ row supplies the source sequence-space admission, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of that finite route. The structural rows H, C, P, N transport, replay, source, and name the same packet.

Since the carrier contains no quotient sequence equality, selected representative, choice schedule, trichotomy branch, or ambient completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 9014.3 (BishopSequenceSpace Stream-RegSeq-Real scope)**.** *Every completion-facing or Real-facing read of an accepted* $\mathsf{BishopSequenceSpace}^{\uparrow}$ *carrier must pass through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, RegularSequenceSpace source route, terminal Real seal, componentwise transport, replay, provenance, and local naming. It receives no quotient stream, selected pointwise representative, countable choice, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $B = (S, R, D, Q, E, H, C, P, N)$. A completion-facing read can inspect only the displayed schedule, readback, tolerance, source-route, seal, transport, replay, provenance, and local naming rows.

Apply the NameCert obligation theorem of Definition 9014.2. It forces the read through S, R, D, Q, E before the structural rows can export the packet to a downstream consumer.

The structural rows do not add new analytic content. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming preserve the same finite route from StreamName windows to the Real seal.

Therefore every public read is scoped to the StreamName, RegSeqRat, DyadicRatCore, RegularSequenceSpace, and Real rows named in the statement, and the excluded quotient, representative, choice, trichotomy, and completeness surfaces are unavailable. □

Falsifiable prediction: Every accepted $\mathsf{BishopSequenceSpace}^{\uparrow}$ completion-facing read exposes a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, RegularSequenceSpace source route, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data before a downstream consumer reads the terminal seal.

Independence witness: streamname, regseqrat, dyadicratcore, regular-sequence-space, real: $\mathsf{BishopSequenceSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds schedule windows, regular rational readback, tolerance arithmetic, sequence-space source admission, Real sealing, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSequenceSpace}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, RegularSequenceSpace source route, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9014.1, Definition 9014.2, and Definition 9014.3 seed the finite Bishop sequence-space route from displayed stream windows through regular rational readback and dyadic tolerance accounting to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real sequences, selected pointwise representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName scheduling, RegSeqRat readback, DyadicRatCore tolerance exactness, RegularSequenceSpace source routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopSequenceSpaceUp.

Chapter 9015

A Concrete Naming Certificate for $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ packet records the finite observer surface by which a regular Cauchy rational name is read through a displayed modulus, dyadic tolerance, sequence-space row, and terminal Real seal. It sits over the finite window discipline of chapter 126, the rational readback of chapter 301, the dyadic tolerance core of chapter 291, the sequence-space route of chapter 5965, and the value-facing boundary of chapter 38. The observer names only displayed regular-completion rows; it does not add quotient stream equality, selected representatives, countable choice, trichotomy, host completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9015.1 (Bishop regular completion observer carrier)**.** *A* $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, R, D, M, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row, listing the finite windows opened by the observer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row served from those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the requested precision scale.*
4. *M is the regular-modulus row, recording the displayed window bound that witnesses the regular Cauchy condition at tolerance D.*
5. *W is the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *membership row tying the observed readback to the sequence-space carrier.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D, M, W remain visible together.*
7. *H records componentwise* hsame *transport for observation windows, regular readback, dyadic tolerance, modulus, sequence-space membership, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an observer request through the same finite rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionObserver}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected representative streams, countable choice, hidden tail selection, trichotomy, host completion equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9015.2 (Bishop regular completion observer NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ *carrier*

$$O = (S, R, D, M, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionObserver}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, regular-modulus evidence,* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *membership, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation row may replace this observer surface by quotient stream equality, selected representatives, countable choice, trichotomy, host completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9015.1. The rows S, R, D, M, W, E are the observer-facing data, while H, C, P, N preserve transport, replay, provenance, and local naming.

The finite observation row S opens the windows that make R visible as regular rational readback. The dyadic row D fixes the tolerance request, and the modulus row M records the displayed regular Cauchy bound for that request.

The sequence-space row W and terminal seal E keep the readback route attached to $\mathsf{RegularSequenceSpace}^{\uparrow}$ before any Real-facing consumer reads it. The structural rows H, C, P, N can only transport, replay, source, and name this same packet.

Since no carrier coordinate stores quotient equality, selected representatives, countable choice, hidden tail selection, trichotomy, host completion equality, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 9015.3 (Bishop regular completion observer regularity window)**.** *Every Real-facing consumer of an accepted* $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ *carrier must expose the observation row* S*, regular rational readback* R*, dyadic tolerance row* D*, regular-modulus row* M*, sequence-space row* W*, and Real seal* E *before reading the value-facing boundary. Thus the observer's completion read is a finite regularity-window route through StreamName, RegSeqRat, DyadicRatCore, RegularSequenceSpace, and Real rows, not a quotient or host-completion principle.*

Proof. Start with an accepted observer packet. The Real-facing boundary is E, but E is downstream of the finite rows S, R, D, M, W in the carrier.

The window row S and regular readback row R display the inspected rational name. The dyadic tolerance row D and modulus row M display the regularity budget at the requested precision scale.

The sequence-space row W attaches this budgeted readback to $\mathsf{RegularSequenceSpace}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. Transport, replay, provenance, and local naming preserve that same finite route through H, C, P, N.

Therefore any consumer that skips S, R, D, M, W is outside the accepted carrier. The observer exports a regularity-window certificate and no quotient equality, representative selector, choice schedule, hidden tail, trichotomy, host completion, or ambient completeness. □

Falsifiable prediction: Any accepted $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ Real-facing read that reaches the terminal seal without displaying a StreamName window, RegSeqRat readback, DyadicRatCore tolerance, regular-modulus row, and RegularSequenceSpace membership refutes the carrier.

Independence witness: streamname, regseqrat, dyadicratcore, regularsequencespace, real: BishopRegularComple is not bijective to any listed sibling because it jointly carries observation windows, regular rational readback, dyadic tolerance, modulus evidence, sequence-space membership, Real sealing, transport, replay, provenance, and local naming as one observer packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCompletionObserver}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCompletionObserver}^{\uparrow}$ packet is generated from finite StreamName observation windows, RegSeqRat readback, DyadicRatCore tolerance, regular-modulus evidence, RegularSequenceSpace membership, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionObserver}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9015.1, Definition 9015.2, Definition 9015.3, the falsifiable prediction, and the independence witness seed the finite regular-completion observer surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCompletionObserver}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected representatives, countable choice, hidden tail choice, trichotomy, host completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observation-window exactness, regular rational readback, dyadic tolerance stability, modulus determinacy, sequence-space membership, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularCompletionObserverUp.

Chapter 9016

A Concrete Naming Certificate for $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$

The $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$ packet realizes the boundary side of section 9413.67 as finite NameCert data. It records only displayed q-split Dedekind lift rows, local split-prime energy, tower provenance, zeta and number-field sources, refusal, transport, replay, and local naming. It sits beside the zeta carrier of chapter 71, the prime carrier of chapter 64, the number-field carrier of chapter 168, the Dedekind carrier of chapter 172, and the prime-spectral fingerprint of chapter 9000. The packet is a boundary certificate for visible lift rows; it does not construct Dedekind cuts, form quotient completions, prove RH, assert analytic continuation, or internalize an infinite tower.

Definition 9016.1 (Dedekind split-tower lift boundary carrier)**.** *A* $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Z, P, K, D, E, F, Q, H, C, G, N)$$

with the following displayed rows:

1. *Z is the* $\mathsf{ZetaBasic}^{\uparrow}$ *source row naming the visible Riemann zeta read being routed into the lift boundary.*
2. *P is the* $\mathsf{Prime}^{\uparrow}$ *split-prime row naming the displayed rational prime q and the finite split-prime windows inspected by the packet.*
3. *K is the* $\mathsf{NumField}^{\uparrow}$ *tower-source row naming the displayed number-field levels and Galois-route data consumed by the boundary.*
4. *D is the* $\mathsf{Dedekind}^{\uparrow}$ *read row naming the displayed Dedekind zeta read at the inspected tower levels.*
5. *E is the split-prime energy row, recording only displayed normal-energy terms for the listed prime-ideal rows.*
6. *F is the refusal row: an off-route Dedekind lift row with no displayed split-prime energy is rejected by the carrier.*
7. *Q is the* $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$ *handoff row, naming the prime-local response fingerprint consumed by the boundary without claiming a spectral classification.*

8. *H records componentwise* hsame *transport for zeta, prime, number-field, Dedekind, energy, refusal, fingerprint, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same finite route.*
10. *G is the* Pkg *provenance over* $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$, $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{NumField}^{\uparrow}$, $\mathsf{Dedekind}^{\uparrow}$, $\mathsf{PrimeSpectralFingerprint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a Dedekind-cut real, quotient completion, selected tower limit, analytic-continuation proof, RH proof object, hidden prime-ideal family, or ambient spectral theorem.

Theorem 9016.2 (Dedekind split-tower lift boundary NameCert obligations)**.** *For every accepted* $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$ *carrier*

$$B = (Z, P, K, D, E, F, Q, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}}$ *surface consists exactly of zeta-source admission, split-prime admission, number-field tower-source admission, Dedekind-read exposure, split-prime energy exposure, off-route refusal, prime-spectral fingerprint handoff, componentwise transport, displayed replay, provenance, and local naming. The packet therefore records a finite boundary route for q-split Dedekind lift rows without constructing Dedekind cuts, quotient completions, hidden tower limits, analytic-continuation proofs, RH proof objects, or ambient spectral classifications.*

Proof. Project the carrier of Definition 9016.1. The rows Z, P, K, D, E, F, Q are the boundary-facing data, and H, C, G, N are the structural rows.

The source route starts with the zeta row Z, split-prime row P, number-field tower row K, and Dedekind read row D. These rows display the finite tower and split-prime data that the boundary is allowed to inspect.

The energy row E and refusal row F make the boundary falsifiable. A lift row that is not accompanied by displayed split-prime energy cannot enter the accepted packet, and the fingerprint row Q supplies only a finite prime-local handoff.

Transport, replay, provenance, and local naming preserve the same displayed rows through H, C, G, N. Since the carrier contains no Dedekind cut, quotient completion, selected tower limit, analytic-continuation proof, RH proof object, hidden prime-ideal family, or ambient spectral theorem, the listed rows exhaust the NameCert obligations. □

Theorem 9016.3 (Dedekind split-tower lift boundary nonescape)**.** *Every accepted* DedekindSplitTowerLiftBoundary *read must pass through*

$$Z, P, K, D, E, F, Q, H, C, G, N$$

before any boundary consumer may inspect the lift route. In particular, an off-route Dedekind lift row with no displayed split-prime energy row E refutes the carrier rather than supplying a hidden tower escape.

Proof. Start with an accepted carrier $B = (Z, P, K, D, E, F, Q, H, C, G, N)$. The zeta, prime, number-field, and Dedekind rows display the only source route that the boundary recognizes.

The split-prime energy row E is required before the lift is read as boundary data. The refusal row F declares that a Dedekind lift row lacking such displayed energy is outside the accepted carrier.

The fingerprint row Q and structural rows H, C, G, N then carry only finite prime-local response, transport, replay, provenance, and local naming. They cannot introduce a hidden tower limit or RH-facing proof object because no such coordinate appears in the carrier.

Thus any boundary consumer must traverse the displayed rows in the stated order, and an off-route lift row without E refutes the carrier surface. □

Falsifiable prediction: Any off-route Dedekind lift row accepted without displayed split-prime energy refutes the $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$ carrier.

Independence witness: zeta-basic, prime, numfield, dedekind, prime-spectral-fingerprint: DedekindSplitTowerLif is not bijective to any listed sibling because it jointly carries zeta-source admission, split-prime display, number-field tower source, Dedekind read, split-prime energy, off-route refusal, prime-spectral fingerprint handoff, transport, replay, provenance, and local naming as one finite boundary packet.

The Lean follow-up for this chapter must provide `Nontrivial` and `FieldFaithful` TasteGate instances for the displayed zeta, prime, number-field, Dedekind, split-energy, refusal, fingerprint, transport, replay, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}$ packet is generated from zeta-source, split-prime, number-field tower, Dedekind read, split-prime energy, refusal, prime-spectral fingerprint, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9016.1, Definition 9016.2, Definition 9016.3, the falsifiable prediction, and the independence witness seed the finite boundary packet for displayed q-split Dedekind lift rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindSplitTowerLiftBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut construction, quotient completion, selected tower limits, analytic continuation, RH, ambient spectral classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zeta-source admission, split-prime display, number-field tower-source admission, Dedekind-read exposure, split-prime energy exactness, off-route refusal, fingerprint handoff, transport, replay, provenance, local naming, and the required Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.DedekindSplitTowerLiftBoundaryUp.

Chapter 9017

A Concrete Naming Certificate for $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$

Following Bishop and Bridges on constructive compactness of closed intervals, the $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$ packet records a finite open-cover certificate by displaying endpoints, cover rows, and dyadic refinements rather than importing compactness as a primitive. It sits beside the real-name seal of chapter 38, the dyadic arithmetic core of chapter 291, the finite dyadic interval-net route of chapter 5966, and the dyadic mesh refinement route of chapter 9010. The packet names only finite interval-cover data; it does not assert arbitrary Heine–Borel compactness, choose a Lebesgue number, select a limiting point, or provide ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9017.1 (Finite open-cover interval carrier)**.** *A* $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (E, U, M, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the interval endpoint row, recording the left and right endpoint reads through* $\mathsf{Real}^{\uparrow}$ *together with their dyadic boundary approximations through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *U is the finite* ProbeBundle *open-cover row, listing the displayed open probes, their interval-hit predicates, and the finite cover ledger inspected by the packet.*
3. *M is the dyadic mesh refinement row, read through* $\mathsf{DyadicMeshRefinement}^{\uparrow}$ *and the finite dyadic interval-net route of chapter 5966.*
4. *W is the membership-witness row assigning every displayed mesh cell to a cover probe whose openness witness contains that cell.*
5. *H records componentwise* hsame *transport for endpoint rows, cover probes, mesh refinements, membership witnesses, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an interval-cover request through endpoint inspection, finite cover inspection, dyadic mesh refinement, and membership witnessing.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicMeshRefinement}^{\uparrow}$*,* ProbeBundle*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through E, U, M, W, H, C, P, N. It has no coordinate for an arbitrary open cover over an unbounded domain, a chosen Lebesgue number, a selected interval point, a quotient real representative, or ambient compactness.

Theorem 9017.2 (FiniteOpenCoverInterval NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$ *carrier* $O = (E, U, M, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverInterval}^{\uparrow}}$ *surface consists exactly of five certificate rows:*

1. *carrier admission for the displayed endpoint and finite-cover packet;*
2. *endpoint exposure through E along the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling routes;*
3. *finite open-cover exposure through the* ProbeBundle *row U;*
4. *dyadic mesh refinement and membership witnessing through M and W;*
5. *structural stability through componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming N.*

Every downstream compact-uniform, density-net, or interval consumer must read those five rows before treating the packet as a finite interval-cover certificate. The route exports no arbitrary compactness principle, no selected Lebesgue number, no point selector, no quotient real representative, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted packet of Definition 9017.1. The endpoint row E, finite cover row U, mesh row M, membership row W, transport row H, replay row C, provenance row P, and local name row N are exactly the displayed coordinates.

Read the sibling routes in order. The real-name seal supplies the endpoint read, the dyadic arithmetic core supplies finite boundary approximations, the dyadic interval-net route supplies the finite cell surface, and the dyadic mesh refinement route supplies the refinement ledger.

The cover row is finite because it is a displayed ProbeBundle row, and the membership row assigns each displayed mesh cell to one displayed open probe. Thus the certificate is consumed through finite rows rather than through an unrecorded compactness theorem.

Finally replay the structural rows. Componentwise hsame transport, Cont replay, Pkg provenance, and local naming preserve the same route and add no coordinate for an arbitrary compactness principle, a selected Lebesgue number, a point selector, quotient real representatives, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$ read is refuted if a compact-uniform, density-net, or interval consumer obtains a finite interval-cover certificate without displaying endpoint rows, finite ProbeBundle cover rows, dyadic mesh refinement rows, membership-witness rows, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: real, dyadicratcore, dyadic-interval-net, dyadic-mesh-refinement: the real-name sibling supplies endpoint seals without finite cover data, the dyadic arithmetic core supplies endpoint arithmetic without open probes, the dyadic interval-net sibling supplies cells without cover membership, and the mesh-refinement sibling supplies refinement rows without the finite ProbeBundle cover. None is bijective to $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOpenCoverInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOpenCoverInterval}^{\uparrow}$ packet is generated from interval endpoint rows, finite ProbeBundle cover rows, dyadic mesh refinement rows, membership-witness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOpenCoverInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9017.1 and Definition 9017.2 seed the finite open-cover interval certificate over RealUp, DyadicRatCoreUp, DyadicMeshRefinementUp, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteOpenCoverInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary Heine–Borel compactness, a selected Lebesgue number, point selection, quotient real representatives, RealUp completeness, Lean verification, bridge checking, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, finite ProbeBundle cover exposure, dyadic mesh refinement, membership-witness exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOpenCoverIntervalUp.

Chapter 9018

A Concrete Naming Certificate for $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$

The $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$ packet realizes the sector-refinement branch of chapter 9463 as finite NameCert data. It sits after the sector envelope of chapter 8546, the sector expansion packet of chapter 8685, the sector overlap ledger of chapter 9013, and the observer-expansion packet of chapter 4167. The packet names a finite refinement of an observer-facing sector; it is not a completed hyperbolic boundary, a selected global sector, a host geometric subdivision theorem, or a physical observer law.

Definition 9018.1 (Hyperbolic sector-refinement carrier)**.** *A* $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, B, V, O, L, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the aperture row inherited from the sector-expansion packet.*
2. *B is the boundary-shadow row inherited from the sector-envelope and boundary-facing sector packets.*
3. *V is the visual-radius row that keeps the refined sector visible at finite observer scale.*
4. *O is the observer-window row, naming the finite focus through which the refinement is read.*
5. *L is the overlap ledger row, exposing the sector-overlap constraints before any refined sector is accepted.*
6. *T is the transport row for moving aperture, boundary-shadow, radius, observer-window, and overlap evidence together.*
7. *H records componentwise* hsame *transport for $A, B, V, O, L, T, C, P, N$.*
8. *C records displayed* Cont *replay through aperture, boundary shadow, visual radius, observer window, overlap, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSectorEnvelope}^{\uparrow}$*,* $\mathsf{HyperbolicSectorExpansion}^{\uparrow}$*,* $\mathsf{HyperbolicSectorOverlapLe}$ $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSectorRefinement}^{\uparrow}}$ *row.*

The classifier compares two refinement packets only through these rows. It has no coordinate for ambient hyperbolic-sector equality, selected ideal endpoints, completed boundary regions, or a host subdivision algorithm.

Theorem 9018.2 (Hyperbolic sector-refinement NameCert obligations)**.** *For every accepted* HyperbolicSectorRefine *carrier*

$$R = (A, B, V, O, L, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicSectorRefinement}^{\uparrow}}$ *surface consists exactly of aperture admission through* A*, boundary-shadow admission through* B*, visual-radius admission through* V*, observer-window admission through* O*, overlap-ledger admission through* L*, transport through* T, H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Every observer-expansion consumer must read the sector envelope, sector expansion, and overlap-ledger sibling routes before treating the refinement as accepted. The packet exports no completed hyperbolic boundary, selected global sector, host geometric subdivision theorem, physical observer law, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 9018.1. The aperture row, boundary-shadow row, visual-radius row, observer-window row, and overlap ledger are the only rows from which a refinement can be read.

Read the sibling anchors in order. The sector envelope supplies the boundary-facing shell, the sector expansion packet supplies aperture and growth comparison, and the sector overlap ledger supplies the finite compatibility rows.

The transport and replay rows keep these dependencies together for each consumer request. Provenance and local naming then record the same finite route as a NameCert surface.

Since the carrier has no coordinate for completed boundary regions, selected endpoints, host subdivision laws, or physical observer dynamics, the listed rows exhaust the accepted refinement packet. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$ read fails if the overlap ledger is removed while the aperture, boundary-shadow, visual-radius, and observer-window rows remain.

Independence witness: hyperbolic-sector-envelope, hyperbolic-sector-expansion, hyperbolic-sector-overlap-ledger, hyperbolic-observer-expansion: no listed sibling is bijective to $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$ because this packet jointly carries aperture, boundary-shadow, visual-radius, observer-window, overlap, transport, replay, provenance, and local naming rows as one refinement surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSectorRefinement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicSectorRefinement}^{\uparrow}$ packet is generated from aperture, boundary-shadow, visual-radius, observer-window, overlap, transport, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9018.1 and Definition 9018.2 seed the finite sector-refinement packet over HyperbolicSectorEnvelopeUp, HyperbolicSectorExpansionUp, HyperbolicSectorOverlapLedgerUp, HyperbolicObserverExpansionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicSectorRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed hyperbolic boundary sectors, selected ideal endpoints, host subdivision theorems, physical observer laws, or ambient Real completeness.

Upgrade path: Obligation closure requires checked aperture admission, boundary-shadow admission, visual-radius admission, observer-window admission, overlap-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicSectorRefinementUp.

Chapter 9019

A Concrete Naming Certificate for $\mathsf{ChomskyHierarchy}^{\uparrow}$

Following Chomsky and Hopcroft–Ullman, the $\mathsf{ChomskyHierarchy}^{\uparrow}$ packet records the standard formal-language strata as one finite BEDC carrier. It links the regular-language surface of chapter 351, the context-free language surface of chapter 7733, the context-sensitive language surface of chapter 7749, the linear-bounded automaton surface of chapter 8100, and the Turing-machine surface of chapter 280. The packet names the hierarchy classifier before any consumer may treat the strata as host powersets of languages or as an unrestricted semantic inclusion theorem.

Definition 9019.1 (Chomsky hierarchy carrier)**.** *A* $\mathsf{ChomskyHierarchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, F, S, L, T, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular-language stratum, anchored by the finite automaton carrier and accepted-word classifier of chapter 351.*
2. *F is the context-free language stratum, anchored by the grammar, derivation, yield, and pushdown-handoff rows of chapter 7733.*
3. *S is the context-sensitive language stratum, anchored by the noncontracting grammar and linear-bounded acceptance rows of chapter 7749.*
4. *L is the linear-bounded automaton stratum, anchored by the bounded-tape computation rows of chapter 8100.*
5. *T is the unrestricted Turing-machine stratum, anchored by the finite trace, source-classifier, ledger-exactness, and public certificate rows of chapter 280.*
6. *H records componentwise* hsame *transport for the five strata and the local hierarchy name.*
7. *C records displayed* Cont *replay from a hierarchy query through R, F, S, L, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{ContextFreeLanguage}^{\uparrow}$, $\mathsf{ContextSensitiveLanguage}^{\uparrow}$, $\mathsf{LinearBoundedAutomaton}^{\uparrow}$, $\mathsf{TuringMachine}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ChomskyHierarchy}^{\uparrow}}$ *row.*

The classifier reads the hierarchy only through the ordered stratum packet R, F, S, L, T. It has no coordinate for a host language powerset, quotient grammar equality, selected universal machine, complement closure, strictness theorem, or ambient decidability principle.

Theorem 9019.2 (Chomsky hierarchy NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyHierarchy}^{\uparrow}$ *carrier*

$$K = (R, F, S, L, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyHierarchy}^{\uparrow}}$ *surface consists exactly of regular-stratum admission through* R*, context-free admission through* F*, context-sensitive admission through* S*, linear-bounded admission through* L*, Turing-machine admission through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

The sibling anchors are the regular-language obligation surface of Definition 351.6, the context-free language obligation surface of Definition 7733.2, the context-sensitive language obligation surface of Definition 7749.2, the linear-bounded automaton obligation surface of Definition 8100.2, and the Turing-machine obligation surface of Definition 280.12. The hierarchy exports this finite ordered carrier, not a host-language inclusion theorem or a strictness classification.

Proof. Project the carrier of Definition 9019.1. The rows R, F, S, L, T are the only stratum rows, and each row points to an existing finite NameCert surface.

Read the sibling obligations. The regular row is grounded by deterministic finite-run data, the context-free row by grammar and derivation data, the context-sensitive row by noncontracting grammar and bounded acceptance data, the linear-bounded row by bounded-tape computation data, and the Turing-machine row by finite trace and public certificate data.

The transport and continuation rows keep hierarchy queries inside the same ordered packet. Provenance and local naming then record exactly this dependency surface.

Since no row stores host powersets, quotient grammar equality, a selected universal machine, complement closure, strictness, or global decidability, the local NameCert obligations close only the finite hierarchy carrier. □

Falsifiable prediction: Removing any one of the five stratum rows R, F, S, L, T makes the $\mathsf{ChomskyHierarchy}^{\uparrow}$ carrier unable to answer the corresponding hierarchy query without importing an external language semantics.

Independence witness: regularlanguage, context-free-language, context-sensitive-language, linear-bounded-automaton, turingmachine: none is bijective to $\mathsf{ChomskyHierarchy}^{\uparrow}$ because the hierarchy packet jointly carries the five ordered stratum anchors, transport, replay, provenance, and local naming as one finite classifier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyHierarchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChomskyHierarchy}^{\uparrow}$ packet is generated from regular, context-free, context-sensitive, linear-bounded automaton, and Turing-machine stratum rows together with componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChomskyHierarchy}^{\uparrow}}$, seed)

Scope closed: Definition 9019.1 and Definition 9019.2 seed the finite hierarchy carrier over RegularLanguageUp, ContextFreeLanguageUp, ContextSensitiveLanguageUp, LinearBoundedAutomatonUp, TuringMachineUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChomskyHierarchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, strictness of the hierarchy, host language-set inclusion, complement closure, universal-machine selection, grammar quotient equality, or global decidability.

Upgrade path: Obligation closure requires checked stratum admission, ordered-classifier stability, route exactness, package provenance, local naming, and non-escape from the finite hierarchy packet for BEDC.Derived.ChomskyHierarchyUp.

Chapter 9020

A Concrete Naming Certificate for $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$

Following Kuratowski's closure axioms and the finite-set practice used in elementary topology, the $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$ packet records the displayed finite ledger by which a closure operator proves preservation of the empty set, extensivity, idempotence, and binary-union stability without importing a host powerset closure as primitive. It sits beside the closure-operator surface of chapter 6154, the topology row of chapter 88, the membership classifier of chapter 3668, and the finite-set packet of chapter 7157.

Definition 9020.1 (Kuratowski finite axioms carrier)**.** *A* $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, T, S, E, A, I, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying displayed-object row whose subset requests are read through* $\mathsf{SetLike}^{\uparrow}$*;*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row exposing the finite open and closed tests available to the packet;*
3. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership classifier row for every subset name mentioned by the ledger;*
4. *E is the empty-set closure row;*
5. *A is the extensivity row, recording that each displayed member of a named subset remains visible after closure;*
6. *I is the idempotence row, recording that a second closure read adds no new displayed subset request;*
7. *U is the binary-union row, recording that closure of a displayed union is read by the two component closure rows and the union classifier;*
8. *H records componentwise* hsame *transport for $X, T, S, E, A, I, U, C, P, N$;*
9. *C records displayed* Cont *replay for the empty, inclusion, idempotence, and binary-union reads;*
10. *P is the* Pkg *provenance over* $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$*,* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteAxioms}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host powerset, arbitrary indexed union, quotient equality of subsets, selected closed hull, classical separation principle, compactness theorem, or ambient set-theoretic closure operator.

Theorem 9020.2 (Kuratowski finite axiom NameCert obligations)**.** *For an accepted* $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$ *carrier* $K = (X, T, S, E, A, I, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteAxioms}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object, topology row, SetLike membership classifier, empty-set closure row, extensivity row, idempotence row, binary-union row, componentwise transport, continuation replay, provenance, and local naming. Every topology or closure-operator consumer must read the sibling* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*, and* $\mathsf{KuratowskiFiniteSet}^{\uparrow}$ *rows before it treats the finite closure ledger as a Kuratowski axiom packet.*

Proof. Project the carrier of Definition 9020.1. The rows X, T, S supply the displayed object, topology surface, and membership classifier before any closure axiom is inspected.

Read the finite axiom ledger next. The empty row E, extensivity row A, idempotence row I, and binary-union row U are the only closure-axiom rows in the packet, and each is tied to the SetLike classifier rather than to an ambient powerset operation.

Finally apply transport, replay, provenance, and local naming through H, C, P, N. These rows keep the four finite axiom checks attached to the Kuratowski closure, topology, SetLike, and finite-set sibling sources, so the certificate exports no arbitrary-union closure, quotient subset equality, selected hull, separation theorem, compactness principle, or host closure operator. □

Theorem 9020.3 (Kuratowski finite axiom topology route)**.** *Every accepted* $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$ *read consumed by a topology-facing chapter factors through*

$$\mathsf{KuratowskiClosureAxiom}^{\uparrow}, \quad \mathsf{Topology}^{\uparrow}, \quad \mathsf{SetLike}^{\uparrow}, \quad \mathsf{KuratowskiFiniteSet}^{\uparrow}.$$

The route exposes only the finite empty, extensivity, idempotence, and binary-union checks displayed in E, A, I, U*. It cannot be used as an arbitrary indexed-union closure principle, a selected closed-hull construction, a quotient equality of subsets, or a compactness argument.*

Proof. Begin with the sibling dependency route named in the carrier. The Kuratowski closure row supplies the closure-operator context, the topology row supplies open and closed tests, the SetLike row classifies displayed subset membership, and the finite-set row bounds the inspected ledger.

The consumer then reads E, A, I, U as four finite checks. Since these rows are displayed entries of the same packet, a topology-facing consumer cannot replace them by an unbounded family of subset claims or by an ambient closure operation.

The structural rows H, C, P, N preserve the same route under transport and replay. Thus the exported surface remains a finite NameCert ledger over the four sibling chapters and does not acquire host powerset, selected-hull, quotient, or compactness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiFiniteAxioms}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KuratowskiFiniteAxioms}^{\uparrow}$ packet is generated from a displayed object row, topology row, SetLike membership classifier, finite empty-set closure row, extensivity row, idempotence row, binary-union row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KuratowskiFiniteAxioms}^{\uparrow}}$, seed)

Scope closed: Definition 9020.1, Definition 9020.2, and Definition 9020.3 seed the finite Kuratowski closure-axiom ledger over topology, SetLike, and finite-set rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KuratowskiFiniteAxioms}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary indexed-union closure, quotient subset equality, selected closed hulls, compactness, separation axioms, or an ambient host powerset closure operator.

Upgrade path: Obligation closure requires checked carrier admission, empty-set closure, extensivity, idempotence, binary-union stability, topology routing, SetLike classifier routing, finite-set bounding, transport, replay, provenance, and local naming for BEDC.Derived.KuratowskiFiniteAxiomsUp.

Chapter 9021

A Concrete Naming Certificate for $\mathsf{CompletionInjectiveExtension}^{\uparrow}$

Following Bishop and Bridges on constructive completions and Kelley on separated completions, the $\mathsf{CompletionInjectiveExtension}^{\uparrow}$ packet records the finite surface by which a densely defined map into a separated target has at most one completion extension. It sits after the dense-extension surface of chapter 6382, the extension-uniqueness surface of **??**, the metric-completion dense embedding of chapter 3351, and the dense embedding carrier of chapter 3617. The packet names injective extension determinacy only as finite NameCert data; it does not identify quotient completions, host functions, selected representatives, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9021.1 (CompletionInjectiveExtension carrier)**.** *A* $\mathsf{CompletionInjectiveExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (D, E, I, U, S, R, H, C, P, N)$$

with the following visible rows:

1. *D is the dense source row, reading the* $\mathsf{DenseEmbedding}^{\uparrow}$ *and* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *data used to place the source inside its completion.*
2. *E is the completion dense-extension row supplied by* $\mathsf{CompletionDenseExtension}^{\uparrow}$*.*
3. *I is the injective image-reflection row, recording that two extension reads with the same dense-source image have the same target-facing image row.*
4. *U is the completion-extension uniqueness row supplied by* $\mathsf{CompletionExtensionUniqueness}^{\uparrow}$ *and the universal-property boundary of* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*.*
5. *S is the separated target row, read through* $\mathsf{CompleteMetric}^{\uparrow}$ *only as the classifier needed to turn zero-distance target reads into a unique target-facing comparison.*
6. *R is the StreamName, RegSeqRat, DyadicRatCore, and Real readback row used to expose the finite completion-facing comparison before the separated classifier consumes it.*
7. *H records componentwise* hsame *transport for dense-source, dense-extension, injective-reflection, uniqueness, separated-target, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from dense-source reads through the completion extension, image reflection, uniqueness route, separated classifier, and readback rows.*

9. *P records* Pkg *provenance over* CompletionInjectiveExtension$^{\uparrow}$, CompletionDenseExtension$^{\uparrow}$, CompletionExtensionU MetricCompletionDenseEmbedding$^{\uparrow}$, DenseEmbedding$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, CompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CompletionInjectiveExtension}^{\uparrow}}$ *row.*

The carrier accepts exactly those finite packets whose injective-extension read can be replayed through $D, E, I, U, S, R, H, C, P, N$.

Theorem 9021.2 (CompletionInjectiveExtension NameCert obligations)**.** *For every accepted* CompletionInjectiveE *carrier*

$$J = (D, E, I, U, S, R, H, C, P, N),$$

the local NameCert$_{\text{CompletionInjectiveExtension}^{\uparrow}}$ *surface consists exactly of dense-source admission for D, completion dense-extension admission for E, injective image-reflection exactness for I, completion-extension uniqueness for U, separated target classification for S, StreamName/RegSeqRat/DyadicRatCore/Real readback for R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The dense source row and dense-extension row give the only input surface from which an extension read can start.

Read the image-reflection and uniqueness rows next. The reflection row I keeps image comparison internal to the displayed dense source, while U supplies the completion-extension uniqueness boundary already named by the sibling chapter.

The target-facing comparison then passes through R and S. The readback row exposes the finite completion comparison, and the separated target row is the only classifier allowed to turn it into a uniqueness read.

Finally H, C, P, N transport, replay, source, and name the same packet. No quotient completion equality, host function equality, selected representative, countable-choice schedule, or ambient Real completeness row occurs among these coordinates. □

Theorem 9021.3 (CompletionInjectiveExtension uniqueness route)**.** *Every accepted injective-extension comparison exported by a* CompletionInjectiveExtension$^{\uparrow}$ *packet factors through the dense source row D, the completion dense-extension row E, the injective image-reflection row I, the completion-extension uniqueness row U, the StreamName/RegSeqRat/DyadicRatCore/Real readback row R, the separated target row S, and the structural rows H, C, P, N. The route is therefore a finite completion-facing determinacy path, not an equality of host functions or quotient completion representatives.*

Proof. Start with the carrier of Definition 9021.1. It lists $D, E, I, U, S, R, H, C, P, N$ as the only accepted coordinates.

Apply the NameCert obligations of Definition 9021.2. They force the comparison through the dense source, dense extension, image reflection, completion-extension uniqueness, readback, and separated-target rows before a consumer can read injective-extension determinacy.

The structural rows preserve that route. Since no host function row, quotient equality row, representative selector, choice schedule, or ambient completeness principle is present in the carrier, no downstream read can obtain one from the packet. □

Closure status for NameCert$_{\text{CompletionInjectiveExtension}^{\uparrow}}$**.**

Constructive story: A CompletionInjectiveExtension$^{\uparrow}$ packet is generated from a dense source row, a completion dense-extension row, an injective image-reflection row, a completion-extension uniqueness row, a separated target classifier,

StreamName/RegSeqRat/DyadicRatCore/Real readback, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompletionInjectiveExtension}^{\uparrow}}$, seed)

Scope closed: Definition 9021.1, Definition 9021.2, and Definition 9021.3 bind the finite injective-extension carrier, its local NameCert obligations, and its uniqueness route.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompletionInjectiveExtension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, host function equality, countable choice, selected representatives, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, injective image-reflection exactness, completion-extension uniqueness routing, separated target classification, readback provenance, transport, replay, and local naming for the same finite packet.

Chapter 9022

A Concrete Naming Certificate for $\mathsf{RegularCauchySelectorStability}^{\uparrow}$

Bishop–Bridges constructive analysis treats a regular Cauchy real by displayed moduli and finite rational windows, not by a hidden completed endpoint. The $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ packet records the finite stability route by which a selected modulus window remains the same Real-facing row after displayed transport. It sits after the selector surface of chapter 1576, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names selector stability only as finite BEDC data; it does not assert quotient equality of real names, a chosen limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9022.1 (Regular Cauchy selector stability carrier)**.** *A* $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, D, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *selector row, naming the accepted modulus threshold used by the regular Cauchy route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the selected source observations are inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the finite dyadic budget attached to the selected modulus threshold.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed selector window and tolerance budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, S, D, R have been displayed.*
6. *U is the selector-stability row, recording that displayed componentwise transport preserves the same selector, window, tolerance, readback, and seal route.*
7. *H records componentwise* hsame *transport for selector rows, stream windows, tolerance rows, readback rows, Real seals, the stability row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a regular Cauchy selector request through M, S, D, R, E, U, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySelectorStability}^{\uparrow}$, $\mathsf{RealCauchyModulusSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, S, D, R, E, U, H, C, P, N$. It has no coordinate for quotient equality of real names, selected representatives, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9022.2 (Regular Cauchy selector stability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ *carrier*

$$T = (M, S, D, R, E, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *selector row,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, selector-stability routing through U, componentwise transport, displayed replay, package provenance, and local naming. A Real or Completion consumer may cite selector stability only after those rows have been displayed.*

Proof. Project the carrier of Definition 9022.1. The selector-facing route first exposes M, S, D, R, E, so the selected modulus threshold, finite window, tolerance budget, regular rational readback, and Real seal are visible before any stability conclusion is read.

Read the stability row U next. It records preservation of the same displayed selector route under componentwise transport; it is not an independent selector, quotient representative, or completed endpoint.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores quotient real equality, selected representatives, countable choice, trichotomy, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ read is refuted if it cites selector stability without displaying the $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ selector row, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: realcauchymodulusselector, streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds selector admission, finite windows, dyadic tolerance, regular readback, Real sealing, selector-stability transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySelectorStability}^{\uparrow}$ packet is generated from a RealCauchyModulusSelector row, a StreamName finite window, a DyadicRatCore tolerance row, RegSeqRat readback, a RealUp seal, selector-stability routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySelectorStability}^{\uparrow}}$, seed)

Scope closed: Definition 9022.1, Definition 9022.2, the falsifiable prediction, and the independence witness seed the finite regular Cauchy selector-stability route over selector, window, tolerance, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySelectorStability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector preservation, StreamName window exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySelectorStabilityUp.

Chapter 9023

A Concrete Naming Certificate for $\mathsf{SpectralSpace}^{\uparrow}$

Following Hochster and the scheme-topology tradition, the $\mathsf{SpectralSpace}^{\uparrow}$ packet records the finite BEDC surface of quasi-compact open bases, finite intersections, and sober prime-facing observation rows. It sits between the topology carrier of chapter 88, the point-free open ledger of chapter 5153, the clopen compact boundary of chapter 7424, and the affine-site consumer surface of chapter 153. The packet names the compact-open certificate data available to later scheme, locale, and Stone routes; it does not import host prime spectra, selected prime ideals, quotient topology equality, spatiality for every locale, or an open-cover compactness theorem outside the displayed rows.

Definition 9023.1 (Spectral space carrier)**.** *A* $\mathsf{SpectralSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, B, K, I, L, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *indexed-open row.*
2. *B is the compact-open basis row, recording displayed basis indices and membership reads.*
3. *K is the quasi-compactness witness row for each displayed basis open, read only as finite cover-refinement data.*
4. *I is the finite-intersection row, requiring the intersection of two displayed compact-open basis rows to be routed through another displayed basis row.*
5. *L is the* $\mathsf{LocaleSpace}^{\uparrow}$ *handoff row, exposing the same compact-open data as a point-free finite meet and cover-refinement ledger.*
6. *A is the* $\mathsf{Scheme}^{\uparrow}$ *affine-site row, used only when scheme consumers read distinguished-open or affine-cover data through the compact-open basis surface.*
7. *H records componentwise* hsame *transport for the topology, basis, compactness, intersection, locale, scheme, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for the route*

$$T, B, K, I \longmapsto L, A, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{SpectralSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{LocaleSpace}^{\uparrow}$*,* $\mathsf{StoneSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SpectralSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, B, K, I, L, A, H, C, P, N$. It has no coordinate for a host set of points, selected prime ideals, arbitrary open-cover compactness, quotient topology equality, or a global spatiality theorem.

Theorem 9023.2 (Spectral space NameCert obligations)**.** *For every accepted* $\mathsf{SpectralSpace}^{\uparrow}$ *carrier $S = (T, B, K, I, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpectralSpace}^{\uparrow}}$ *surface consists exactly of topology-source exposure through T, compact-open basis exposure through B, finite cover-refinement compactness through K, compact-open finite-intersection stability through I, locale handoff through L, affine-site scheme handoff through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. A* $\mathsf{StoneSpace}^{\uparrow}$*,* $\mathsf{LocaleSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or* $\mathsf{Topology}^{\uparrow}$ *consumer may read only those rows from the packet.*

Proof. Carrier admission is projection from Definition 9023.1. The accepted packet displays exactly the topology row, compact-open basis row, compactness row, finite-intersection row, locale handoff, scheme handoff, transport, replay, provenance, and local naming row.

The basis row B is read before compactness or intersection data. The row K then records only finite cover-refinement witnesses for displayed basis opens, while I records only finite intersections that remain inside the displayed compact-open basis.

The locale handoff L and scheme handoff A consume that same basis route. The locale consumer receives finite meet and cover-refinement data, and the scheme consumer receives affine-site compact-open reads, but neither receives a detached host point set or a chosen family of prime ideals.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no coordinate for arbitrary open covers, quotient topology equality, selected prime ideals, or a global spatiality theorem, the listed rows exhaust the local NameCert obligation surface. □

Theorem 9023.3 (Spectral space compact-open stability)**.** *Every compact-open read exported by an accepted* $\mathsf{SpectralSpace}^{\uparrow}$ *packet factors through*

$$\mathsf{Topology}^{\uparrow} \longmapsto B, K, I \longmapsto \mathsf{LocaleSpace}^{\uparrow}, \mathsf{Scheme}^{\uparrow}, \mathsf{StoneSpace}^{\uparrow} \longmapsto H, C, P, N.$$

Thus a downstream scheme, locale, Stone, or topology consumer sees a compact-open basis row together with finite cover-refinement and finite-intersection stability. The read exports no selected prime point, host powerset of opens, quotient topology equality, arbitrary open-cover compactness, or bridge-checked spatial representation.

Proof. Start with the topology row T and compact-open basis row B. A compact-open read is admitted only after the basis index and membership observation have been displayed.

The compactness and stability data are then K and I. The compactness row is finite cover-refinement data for the displayed basis open, and the intersection row keeps binary meets inside the same compact-open basis ledger.

The locale, scheme, and Stone consumers can only inspect those displayed rows. Locale reads use the point-free meet and cover route, scheme reads use the affine-site compact-open handoff, and Stone reads compare the clopen compact boundary as a sibling compact-open fragment.

Transport, replay, provenance, and local naming preserve the same finite route. Since no accepted carrier stores selected prime points, host powersets, quotient topologies, arbitrary open-cover compactness, or a checked bridge, those objects cannot be exported by the compact-open read. □

Falsifiable prediction: Every accepted $\mathsf{SpectralSpace}^{\uparrow}$ read exposes a TopologyUp indexed-open row, compact-open basis row, finite cover-refinement compactness row, finite-intersection row, LocaleSpaceUp handoff, SchemeUp affine-site handoff, StoneSpaceUp sibling comparison, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row before a compact-open consumer may use the packet.

Independence witness: topology, locale-space, stone-space, scheme: $\mathsf{SpectralSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact-open basis data, finite cover-refinement compactness, finite-intersection stability, locale handoff, scheme affine-site handoff, Stone sibling comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralSpace}^{\uparrow}$ packet is generated from a TopologyUp indexed-open row, compact-open basis row, finite cover-refinement compactness row, finite-intersection row, LocaleSpaceUp handoff, SchemeUp affine-site handoff, StoneSpaceUp sibling comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9023.1, Definition 9023.2, and Definition 9023.3 seed the compact-open spectral-space surface over topology, locale, Stone, scheme, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected prime points, host powersets of opens, quotient topology equality, arbitrary open-cover compactness, spatiality for every locale, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open basis exposure, finite cover-refinement compactness, finite-intersection stability, locale and scheme handoffs, Stone sibling comparison, transport, replay, provenance, and local naming for BEDC.Derived.SpectralSpaceUp.

Chapter 9024

A Concrete Naming Certificate for FiniteModulusRegularCauchyCompletion$^{\uparrow}$

Following Bishop and Bridges, the FiniteModulusRegularCauchyCompletion$^{\uparrow}$ packet records the finite modulus surface by which a regular rational Cauchy name is read into the Real and Completion route without claiming ambient completeness. It sits after the dyadic arithmetic core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet packages the displayed modulus ledger only; it does not quotient regular streams, select a completed limit, assert host real equality, import countable choice, or promote Real$^{\uparrow}$ to an ambient completeness theorem.

Definition 9024.1 (Finite modulus regular Cauchy completion carrier)**.** *A* FiniteModulusRegularCauchyCompletion *carrier is a finite* BHist *packet*

$$F = (D, S, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *window row, carrying the dyadic precision scale, comparison budget, and tolerance ledger used by the completion-facing read.*
2. *S is the* StreamName$^{\uparrow}$ *schedule row, listing the finite observations inspected before a modulus claim is exposed.*
3. *R is the* RegSeqRat$^{\uparrow}$ *regularity row, recording the rational-name regularity predicate on the displayed schedule.*
4. *M is the tail-modulus row, binding each requested dyadic precision to a displayed tail window in S and a regularity witness in R.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after D, S, R, M have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic window, stream schedule, regularity, tail modulus, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the dyadic precision request through the stream schedule, regular rational readback, tail modulus, real seal, transport, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* FiniteModulusRegularCauchyCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, hidden convergence witnesses, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9024.2 (Finite modulus regular Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}$ *carrier $F = (D, S, R, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}}$ *surface consists exactly of dyadic-window admission, finite stream scheduling, regular rational readback, tail-modulus exposure, terminal Real seal, componentwise transport, displayed replay, package provenance, and local naming. Every completion-facing consumer must read the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling surfaces through those displayed rows before citing the finite modulus route.*

Proof. Project the carrier of Definition 9024.1. The rows D, S, R, M expose the dyadic precision window, the inspected stream schedule, the regular rational readback, and the tail-modulus ledger.

The terminal seal E is downstream from those rows. It can be read only after the displayed regular Cauchy route has opened the dyadic budget and finite schedule, so the seal is not a selected host limit or a quotient of streams.

The structural rows H, C, P, N preserve, replay, source, and name the same finite evidence. Since the carrier has no coordinate for hidden convergence, countable choice, quotient equality, host real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 9024.3 (Finite modulus completion route boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}$ *packet factors through the dyadic window row D, stream schedule row S, regularity row R, tail-modulus row M, terminal seal row E, transport row H, replay row C, provenance row P, and local naming row N. A consumer that reaches a real seal without the displayed finite modulus leaves the packet.*

Proof. Apply the obligations theorem. It lists $D, S, R, M, E, H, C, P, N$ as the complete local surface, with the $\mathsf{Real}^{\uparrow}$ seal downstream from dyadic, stream, regularity, and modulus rows.

If a consumer omits M, then the requested precision has no displayed tail window. If it omits D, S, or R, then the modulus is detached from the dyadic budget, finite observations, or regular rational readback.

Transport, replay, provenance, and local naming do not create missing analytic data. Therefore every accepted completion read must pass through the full finite route, and a detached seal is outside the carrier. □

Falsifiable prediction: A Real or Completion consumer that cites a terminal RealUp seal without a displayed dyadic window, stream schedule, regular rational readback, and tail-modulus row refutes the FiniteModulusRegularCauchyCompletionUp carrier.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}$ is not bijective to any listed sibling because it binds dyadic tolerance, finite observation schedule, regular rational readback, tail modulus, terminal seal, transport, replay, provenance, and local naming as one completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore precision window, StreamName observation schedule, RegSeqRat regularity row, tail-modulus ledger, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusRegularCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 9024.1, Definition 9024.2, and Definition 9024.3 seed the finite regular-Cauchy completion route over DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteModulusRegularCauchyCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic-window admission, stream scheduling, regularity readback, tail-modulus exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRegularCauchyCo

Chapter 9025

A Concrete Naming Certificate for DyadicTruncationLPRefutationWindow$^{\uparrow}$

The DyadicTruncationLPRefutationWindow$^{\uparrow}$ packet realizes the sharp dyadic truncation route obstruction of chapter 9338 as finite NameCert data. It sits beside the zeta-zero surface of chapter 73, the critical-strip boundary of chapter 72, the zero-count ledger of chapter 573, the contour-operation route of chapter 521, and the finite ordered-field comparison discipline of chapter 235. The packet names the obstruction to the all-cutoff Laguerre–Polya premise; it does not prove or disprove RH, classify all truncations, assert a smooth-deformation route, or import analytic continuation beyond displayed rows.

Definition 9025.1 (Dyadic truncation LP refutation window carrier)**.** *A* DyadicTruncationLPRefutationWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, T, Z, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *K is the dyadic cutoff row, recording the sharp finite cutoff and the dyadic precision budget used to inspect the truncated integral.*
2. *T is the truncation-evaluation row, recording the displayed finite approximation of the sharp cutoff function on the inspected contour windows.*
3. *Z is the zero-count row, carrying the argument-principle or equivalent winding-count evidence that the inspected truncation has a non-real zero pair.*
4. *Q is the Laguerre–Polya premise row, naming the all-real-zero condition that the sharp cutoff route would require at the inspected cutoff.*
5. *F is the refusal row, recording that the displayed non-real zero-count row refutes the all-cutoff premise at this finite window.*
6. *H records componentwise* hsame *transport for cutoff, truncation, zero-count, premise, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the cutoff and truncation rows through zero-count evidence, premise refusal, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* DyadicTruncationLPRefutationWindow$^{\uparrow}$*,* ZetaZeros$^{\uparrow}$*,* CritStrip$^{\uparrow}$*,* LPDuality$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, T, Z, Q, F, H, C, P, N$. It has no coordinate for a proof of RH*, a disproof of* RH*, a global zero classifier, a hidden analytic-continuation theorem, a smoothed heat-flow replacement, or an all-cutoff real-rootedness oracle.*

Theorem 9025.2 (Dyadic truncation LP refutation window NameCert obligations)**.** *For every accepted* $\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}$ *carrier $W = (K, T, Z, Q, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}}$ *surface consists exactly of cutoff admission, truncation-evaluation exposure, zero-count evidence, Laguerre–Polya premise naming, finite refusal, componentwise transport, displayed replay, package provenance, and local naming. The packet makes the sharp-cutoff route obstruction inspectable as one finite window while leaving all global RH readings outside the carrier.*

Proof. Project the carrier of Definition 9025.1. The cutoff row K and truncation row T identify the inspected sharp finite window, while Z supplies the displayed zero-count evidence.

The premise row Q names the all-real-zero condition required by the route. The refusal row F is then finite: the displayed non-real zero-count evidence at the same window blocks that premise for the sharp-cutoff family.

The rows H, C, P, N transport, replay, source, and name the same evidence. Since no coordinate stores an RH proof, an RH refutation, a global zero classifier, or a smoothed deformation route, the listed rows exhaust the local NameCert obligations. □

Theorem 9025.3 (Dyadic truncation LP route nonclaim boundary)**.** *Every accepted* DyadicTruncationLPRefutation *read exposes only the finite route*

$$K, T, Z, Q, F, H, C, P, N$$

before a consumer may cite the Laguerre–Polya premise obstruction. The read cannot be reinterpreted as a proof of RH*, a disproof of* RH*, a statement about every cutoff family, or a smooth de Bruijn–Newman deformation certificate.*

Proof. Use the obligations theorem. The packet's positive content is the finite cutoff, truncation-evaluation, zero-count, premise, and refusal surface.

The obstruction is local to the displayed sharp cutoff row. Transport and replay preserve that local evidence; they do not quantify over all cutoffs and do not replace the sharp cutoff by a smooth heat-flow deformation.

Thus the consumer receives a finite route refutation window only. Any RH conclusion, global zero classification, or smooth-deformation certificate would require coordinates absent from the carrier. □

Falsifiable prediction: An accepted sharp dyadic truncation window with displayed non-real zero-count evidence refutes the all-cutoff Laguerre–Polya premise for that sharp-cutoff route.

Independence witness: zeta-zeros, critical-strip, riemann-zero-count-ledger, contourintegral-operation, lpduality: $\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries cutoff, truncation, zero-count, premise, refusal, transport, replay, provenance, and local naming rows while refusing all global RH readings.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTruncationLPRefutationWindow}^{\uparrow}$ packet is generated from a dyadic cutoff row, truncation-evaluation row, zero-count evidence row, Laguerre–Polya premise row, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicTruncationLPRefutationWindow}\uparrow}$, seed)

Scope closed: Definition 9025.1, Definition 9025.2, and Definition 9025.3 seed the finite sharp-cutoff route obstruction over zeta-zero, critical-strip, zero-count, contour, LP-duality, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicTruncationLPRefutationWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove RH, disprove RH, classify all truncations, close analytic continuation, replace sharp cutoffs by a smooth deformation, provide Lean verification, or bridge check the route.

Upgrade path: Obligation closure requires checked cutoff admission, truncation evaluation, zero-count evidence, Laguerre–Polya premise naming, finite refusal, transport, replay, provenance, and local naming for BEDC.Derived.DyadicTruncationLPRefutatio

Chapter 9026

Bishop Separated Real Completion NameCert Construction

Following Bishop–Bridges constructive-completion practice, the $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ packet records the finite route by which regular rational stream data is read as a separated real completion object. It refines the regular Cauchy route of chapter 1639, the metric completion carrier of chapter 1506, and the separation reflector of chapter 1790. The packet exposes completion data through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{SeparatedMetricReflection}^{\uparrow}$; it does not add quotient real equality, countable choice, a host limit selector, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9026.1 (Bishop separated real completion carrier)**.** *A* $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row, exposing only the stream windows requested by the completion consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational row read from those stream windows.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, recording the completion-facing metric handoff before separation is applied.*
4. *Q is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *separation row, restricting comparison to displayed indistinguishability data instead of quotient equality.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, R, M, Q have been displayed.*
6. *H records componentwise* hsame *transport for the stream, regularity, metric-completion, separation, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion request through S, R, M, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{SeparatedMetricReflection}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSeparatedRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, M, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopSeparatedRealCompletionUp`

Theorem 9026.2 (Bishop separated real completion classifier surface)**.** *For every accepted* BishopSeparatedRealCor *carrier $B = (S, R, M, Q, E, H, C, P, N)$, the local classifier is exhausted by the stream row S, regular rational row R, metric-completion row M, separated-reflection row Q, Real seal E, componentwise transport H, replay C, provenance P, and local naming row N. Every comparison between accepted carriers must first expose the same finite stream window and regular rational readback before it may inspect the separated metric reflection row. No classifier branch may introduce quotient equality of streams, host real equality, a selected limit, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier in Definition 9026.1. The only data available to the classifier are the displayed rows $S, R, M, Q, E, H, C, P, N$.

The completion-facing part of the route reads finite observations through S, regular rational data through R, and the metric completion handoff through M. The separated reflection row Q is downstream of those rows, so it can compare only the displayed metric-completion data already present in the packet.

Transport and replay are row-local. Componentwise hsame in H preserves the same rows, and Cont replay in C repeats the route S, R, M, Q, E for each request.

Since neither the carrier nor its structural rows contain an excluded coordinate, the classifier surface cannot supply quotient stream equality, host real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 9026.3 (Bishop separated real completion stability route)**.** *Let $B = (S, R, M, Q, E, H, C, P, N)$ and $B' = (S', R', M', Q', E', H', C', P', N')$ be accepted* $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ *carriers. If the displayed transport rows identify corresponding components by* hsame*, then any Real-facing consumer reads B and B' through the same ordered route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{SeparatedMetricReflection}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The stability route preserves the finite carrier surface and exports no quotient equality, host-selected representative, countable-choice schedule, detached real equality, or ambient Real completeness.

Proof. Start with the classifier surface of Definition 9026.2. It restricts both accepted carriers to the rows $S, R, M, Q, E, H, C, P, N$ and their transported companions.

The hsame rows identify corresponding displayed components only. Thus a transported read of B' follows the same finite stream window, the same regular rational handoff shape, the same metric-completion exposure, and the same separated-reflection boundary as the read of B.

Finally, the Cont rows replay the ordered route before the terminal $\mathsf{Real}^{\uparrow}$ seal can be inspected. The replay keeps the separated metric reflection downstream of the metric completion row and upstream of the Real seal, so no external quotient or selected representative is introduced. □

Theorem 9026.4 (Bishop separated real completion embedding handoff)**.** *Every standard real-embedding read from an accepted* $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ *packet factors through the displayed carrier rows*

$$S, R, M, Q, E, H, C, P, N.$$

The embedding may use the regular rational stream data as a completion-facing name and may pass through the separated metric reflection boundary, but it receives the $\mathsf{Real}^{\uparrow}$ *seal only as the terminal displayed row. It cannot claim ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, countable choice, host real equality, or a selected representative outside the packet.*

Proof. Use the carrier definition to expose the finite rows. A read that asks for an embedding first consumes the $\mathsf{StreamName}^{\uparrow}$ row S and the $\mathsf{RegSeqRat}^{\uparrow}$ row R, because these rows are the only source of regular rational stream data in the packet.

The metric-completion row M then records the completion-facing handoff, while Q records the separated reflection boundary. The terminal Real seal E is downstream of both rows and cannot be read independently of them.

The structural rows H, C, P, N transport, replay, record provenance, and name the same carrier surface. They add no coordinate for an external representative, quotient equality, countable choice, host equality, or ambient completeness. □

Theorem 9026.5 (Bishop separated real completion uniqueness boundary)**.** *If two accepted* BishopSeparatedRealCo *packets have the same displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the same* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *comparison row, then every completion-facing comparison between them is determined inside the shared finite carrier surface. The comparison is a separated-reflection boundary over displayed rows, not a quotient of streams, a host real equality, a selected limit, a countable-choice witness, or an ambient completeness theorem.*

Proof. Read the two packets through the stability route of Definition 9026.3. Both completion-facing comparisons must expose stream windows and regular rational readbacks before either metric-completion or Real-seal data is available.

The shared separated reflection row is the only comparison boundary admitted by the carrier. Once that row is fixed, the remaining Real-facing read is replayed through the same displayed metric completion handoff and terminal seal.

Thus uniqueness is scoped to the finite carrier surface. The packet supplies no further equality principle and no way to choose representatives outside the displayed rows. □

Theorem 9026.6 (Bishop separated real completion ledger rows)**.** *The local* $\mathsf{NameCert}_{\mathsf{BishopSeparatedRealCompletion}^{\uparrow}}$ *ledger contains exactly these rows: carrier admission for* $S, R, M, Q, E, H, C, P, N$*; classifier exposure for the same rows; stability under componentwise* hsame *transport; embedding handoff through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{SeparatedMetricReflection}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; uniqueness boundary inside the separated reflection row; provenance through* Pkg*; and local naming through* N*. It records no quotient equality, countable choice, host-selected limit, detached real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Carrier admission is Definition 9026.1. Classifier exposure is Definition 9026.2.

Stability is supplied by Definition 9026.3, which keeps every transported read inside the same ordered row surface. The embedding handoff is Definition 9026.4.

The uniqueness boundary is Definition 9026.5. Provenance and local naming are the displayed rows P and N, and neither row adds data beyond the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSeparatedRealCompletion}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopSeparatedRealCompletion}^{\uparrow}}$, seed)

Scope closed: The chapter records the seed carrier, classifier, stability, embedding, uniqueness, and ledger rows for the Bishop separated real completion packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopSeparatedRealCompletion}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.BishopSeparatedRealCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of streams, host real equality, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: To reach scoped closure, formalize the listed carrier rows and prove classifier, stability, embedding, uniqueness, and ledger exactness in Lean.

Constructive story: A BishopSeparatedRealCompletion$^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, MetricCompletion handoff, SeparatedMetricReflection comparison, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9027

A Concrete Naming Certificate for $\mathsf{LoewnerOrder}^{\uparrow}$

Following Loewner's order on self-adjoint matrices and the Boyd–Vandenberghe semidefinite-programming presentation, the $\mathsf{LoewnerOrder}^{\uparrow}$ packet records the finite BEDC surface for the comparison $A \preceq B$ as semidefinite-cone membership of the displayed difference $B - A$. It sits after the Hermitian matrix carrier of chapter 6403, the semidefinite-cone packet of chapter 310, and the affine feasible-region route of chapter 5960. The packet names an order interface over finite Hermitian matrix rows; it does not assert spectral decomposition, eigenvalue monotonicity, analytic closure, selected diagonalization, or a host positive-semidefinite predicate.

Definition 9027.1 (Loewner order carrier)**.** *A* $\mathsf{LoewnerOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(A, B, H, D, K, V, O, T, C, P, N)$$

with the following displayed rows:

1. *A is the left* $\mathsf{HermitianMatrix}^{\uparrow}$ *row.*
2. *B is the right* $\mathsf{HermitianMatrix}^{\uparrow}$ *row over the same square dimension, scalar source, and vector-space boundary as A.*
3. *H is the shared Hermitian comparison window, recording the common dimension, complex scalar rows, vector-space source, and self-adjointness readbacks used by A and B.*
4. *D is the displayed difference-matrix row for $B - A$, read through the same* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *provenance as the two endpoint rows.*
5. *K is the* $\mathsf{SemidefiniteCone}^{\uparrow}$ *membership row for the difference matrix $B - A$.*
6. *V is the joint* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *provenance row tying A, B, H, and D to one finite coordinate surface.*
7. *O is the* $\mathsf{Order}^{\uparrow}$ *nonnegativity readback exported by K for displayed vector tests against $B - A$.*
8. *T records componentwise* hsame *transport for the endpoint, Hermitian, difference, cone-membership, order, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from endpoint inspection through difference formation and semidefinite-cone readback.*

10. *P is the* Pkg *provenance over* $\mathsf{LoewnerOrder}^{\uparrow}$, $\mathsf{HermitianMatrix}^{\uparrow}$, $\mathsf{SemidefiniteCone}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LoewnerOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the eleven displayed rows. It has no coordinate for spectral theorem data, eigenvalue comparison, quotient matrix equality, analytic closure, selected basis data, or host-level positive semidefiniteness.

Theorem 9027.2 (Loewner order NameCert obligations)**.** *For every accepted* $\mathsf{LoewnerOrder}^{\uparrow}$ *carrier*

$$L = (A, B, H, D, K, V, O, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LoewnerOrder}^{\uparrow}}$ *surface consists exactly of carrier admission, left and right Hermitian endpoint exposure, shared Hermitian comparison-window exposure, displayed difference-matrix formation, semidefinite-cone membership of $B - A$, vector-space and matrix provenance,* $\mathsf{Order}^{\uparrow}$ *nonnegativity readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission is projection from Definition 9027.1. The endpoint rows A and B are visible only through the Hermitian matrix carrier, and the shared window H keeps their dimension, scalar source, vector-space source, and self-adjointness reads aligned.

The difference obligation is finite. The row D is the displayed matrix row for $B - A$, formed from the same matrix and vector-space provenance row V that types the two endpoint packets.

The positivity obligation is the semidefinite-cone row K. Its order-facing output is the nonnegativity readback O, so a consumer sees the comparison only after passing through the sibling semidefinite-cone certificate.

Finally T, C, P, N close the transport, replay, provenance, and local naming surface. Since no listed row stores spectral data, eigenvalue comparison, selected bases, quotient matrix equality, analytic closure, or a host positivity predicate, the displayed rows exhaust the local NameCert obligations. □

Theorem 9027.3 (Loewner order semidefinite route)**.** *Every optimization or convex-feasibility consumer that reads an accepted* $\mathsf{LoewnerOrder}^{\uparrow}$ *packet as $A \preceq B$ must factor through the displayed route*

$$(A, B, H, V) \longmapsto D = B - A \longmapsto K : \mathsf{SemidefiniteCone}^{\uparrow}(D) \longmapsto O : \mathsf{Order}^{\uparrow}.$$

The route may be cited by spectrahedral consumers, but it exports only the finite difference row, semidefinite-cone membership row, order readback, transport, replay, and provenance rows already present in the packet.

Proof. Read the accepted packet through Definition 9027.1. The left and right endpoints are Hermitian rows over the shared comparison window, so a matrix inequality read has no source other than A, B, H, and V.

Next form the displayed difference row D. This is the only row that turns endpoint data into a cone-facing matrix, and it remains tied to the same $\mathsf{Mat}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ provenance.

Then read the semidefinite-cone membership row K and its order output O. By chapter 310, the cone read is a finite BEDC membership surface rather than an imported positive-semidefinite predicate.

A spectrahedral consumer may use the resulting route as an affine-matrix inequality boundary, as in chapter 5960. The consumer still receives only D, K, O, T, C, P, N, so the Loewner comparison remains a finite semidefinite route. □

Theorem 9027.4 (Loewner order Hermitian nonescape)**.** *Every accepted* $\mathsf{LoewnerOrder}^{\uparrow}$ *packet keeps both endpoints and the displayed difference inside the finite Hermitian matrix surface: the rows* A *and* B *are* $\mathsf{HermitianMatrix}^{\uparrow}$ *carriers, the shared row* H *supplies the common self-adjoint comparison window, and the difference row* D *is admitted only as the matrix row consumed by the semidefinite-cone membership* K *over that same Hermitian boundary.*

Thus a Loewner comparison cannot escape to non-Hermitian matrices, quotient matrix equality, host equality of arrays, selected eigenbasis data, or spectral positivity.

Proof. Project the endpoint rows A and B from the carrier. Both are explicit $\mathsf{HermitianMatrix}^{\uparrow}$ rows, so their matrix entries, conjugate-transpose reads, and self-adjointness comparisons are governed by chapter 6403.

The row H keeps the two endpoint packets over one comparison window. Transport through T is componentwise over that same window, and replay through C inspects only the displayed endpoint, difference, cone, and order rows.

The difference row D is not a separate ambient matrix object. It is the finite matrix row consumed by K, and K is the semidefinite-cone membership row whose order readback is O.

Since the carrier has no additional coordinate, a consumer cannot read non-Hermitian matrix data, quotient equality, host array equality, selected eigenbasis data, or spectral positivity from an accepted $\mathsf{LoewnerOrder}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LoewnerOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LoewnerOrder}^{\uparrow}$ packet is generated from two $\mathsf{HermitianMatrix}^{\uparrow}$ endpoint rows over a shared comparison window, a displayed difference-matrix row, a $\mathsf{SemidefiniteCone}^{\uparrow}$ membership row for $B - A$, an $\mathsf{Order}^{\uparrow}$ nonnegativity readback, hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LoewnerOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9027.1, Definition 9027.2, Definition 9027.3, and Definition 9027.4 seed the finite Loewner-order packet over displayed $\mathsf{HermitianMatrix}^{\uparrow}$, $\mathsf{SemidefiniteCone}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LoewnerOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close spectral decomposition, eigenvalue monotonicity, analytic closure, selected diagonalization, host positive-semidefinite predicates, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint Hermitian visibility, difference-row exactness, semidefinite-cone membership, order readback, transport, replay, provenance, and local naming for the same finite packet.

Chapter 9028

A Concrete Naming Certificate for $\mathsf{ConstructiveMetricSpace}^{\uparrow}$

Following Bishop and Bridges on constructive metric spaces, the $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ packet records the finite interface needed before an L10 completion consumer may ask for points, located distances, separatedness evidence, or completion ledgers. It is read downstream of the metric naming surface in chapter 57, the finite window service of chapter 126, the regular rational-name route of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 128. The packet supplies countable constructive naming data; it does not import host equality, quotients, choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9028.1 (Constructive metric-space carrier)**.** *A* $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, E, d, L, S, R, C, Q, P, N)$$

with the following displayed rows:

1. *X is the point-name carrier, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *E is the point classifier, transported by displayed* hsame *rows and never by host equality of points.*
3. *d is the located-distance service, returning finite* $\mathsf{RegSeqRat}^{\uparrow}$ *evidence for requested point pairs and requested tolerances.*
4. *L is the separatedness-reflection row: a positive located distance is the only exported witness that two point names are apart.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger used by point and distance reads.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for completed distance names, available only after the located-distance and regular-rational rows have been displayed.*
7. *C is the completion ledger tying the point, classifier, located-distance, separatedness, stream, regular-rational, and real-seal rows to downstream* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers.*
8. *Q stores componentwise* hsame *transport for the displayed rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, E, d, L, S, R, C, Q, P, N$. It has no coordinate for quotient completion, countable choice, selected representatives, host point equality, or ambient completeness.

Theorem 9028.2 (Constructive metric-space NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ *carrier* $M = (X, E, d, L, S, R, C, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveMetricSpace}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *point carrier: X admits only finitely requested point-name windows through S;*
2. *classifier: E compares point names through displayed* hsame *and* Cont *routes, not through host equality;*
3. *located distance: d supplies* $\mathsf{RegSeqRat}^{\uparrow}$ *evidence for each requested pair and tolerance before the* $\mathsf{Real}^{\uparrow}$ *seal R is available;*
4. *separatedness reflection: L exports apartness only from a displayed positive located-distance row;*
5. *completion ledger: C records how $X, E, d, L, S, R, Q, P, N$ may be consumed by a later* $\mathsf{CompleteMetric}^{\uparrow}$ *packet.*

No obligation row exports quotient points, selected Cauchy representatives, choice functions, host equality of names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9028.1. The point carrier, classifier, located-distance row, separatedness-reflection row, stream ledger, real seal, completion ledger, transports, provenance, and local naming row are all visible coordinates of the same finite packet.

The located-distance obligation reads a requested point pair through X, E, S before it reaches the regular-rational evidence in d. The real-facing row R is a seal for that displayed distance name, not a completion principle.

The completion ledger C then records the downstream handoff to $\mathsf{CompleteMetric}^{\uparrow}$ consumers. Since the carrier has no quotient row, selected representative row, choice schedule, host equality row, or ambient completeness row, the NameCert surface is exhausted by the five displayed obligations. □

Theorem 9028.3 (Constructive metric-space separated completion route)**.** *Every downstream completion read of an accepted* $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ *carrier must pass through the displayed route*

$$X, E, d, L, S, R, C, Q, P, N.$$

The route may hand a $\mathsf{CompleteMetric}^{\uparrow}$ *consumer point names, classifier evidence, located distances, separatedness reflection, finite StreamName windows, RegSeqRat distance names, Real seals, transport rows, package provenance, and local naming. It may not hand over quotient-completion equality, a selected limit, countable choice, host point equality, or a theorem that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with the completion ledger C. Its only inputs are the visible rows of the carrier, so every completion-facing read is downstream of the point carrier, classifier, located-distance service, separatedness row, window ledger, real seal, transports, provenance, and local NameCert row.

Separatedness is supplied by L only after a positive located-distance row has been displayed. Thus a completion consumer may cite separatedness as a finite reflection of the located metric data, not as host equality or quotient equality.

The terminal handoff to $\mathsf{CompleteMetric}^{\uparrow}$ is therefore a route through countable constructive naming data. It supplies the interface needed by completion work while leaving quotient construction, selected limits, choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside this chapter. □

Falsifiable prediction: Every $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ completion-facing read factors through one point carrier, one point classifier, one located-distance service, one separatedness-reflection row, one finite StreamName ledger, one RegSeqRat distance-name route, one RealUp seal, one completion ledger, componentwise hsame transport, package provenance, and local naming.

Independence witness: metric, streamname, regseqrat, real, completemetric: $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries point naming, classifier stability, located-distance service, separatedness reflection, finite windows, regular rational distance names, Real seals, completion ledger rows, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveMetricSpace}^{\uparrow}$ packet is generated from point names, a displayed point classifier, located-distance rows, separatedness reflection, StreamName windows, RegSeqRat distance names, a RealUp seal, a CompleteMetricUp ledger, componentwise hsame transport, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9028.1, Definition 9028.2, and Definition 9028.3 seed the finite constructive metric-space interface consumed by separated completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked point-carrier admission, classifier stability, located-distance exactness, separatedness reflection, completion-ledger replay, transport, provenance, and local naming for BEDC.Derived.ConstructiveMetricSpaceUp

Chapter 9029

A Concrete Naming Certificate for $\mathsf{CantorSpaceCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CantorSpaceCompletion}^{\uparrow}$ packet records the completion of Cantor space by finite prefix Cauchy names. It sits after the binary-prefix carrier of chapter 1475, the finite prefix stream certificate of chapter 7253, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the Cantor-intersection route of chapter 1421, and the interval coding surface of chapter 7588. The packet names constructive prefix completion as displayed BEDC rows; it is not quotient stream equality, not open-cover compactness, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9029.1 (CantorSpaceCompletion carrier)**.** *A* $\mathsf{CantorSpaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, D, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row, naming the inspected binary cylinders and their extension order.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, displaying the prefix observations available at the requested depths.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic diameter bound attached to each displayed cylinder.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same prefix windows and dyadic bounds.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after A, W, D, R have been displayed.*
6. *L is the prefix Cauchy ledger, recording that later prefix windows extend the earlier cylinders and shrink their displayed dyadic diameters.*
7. *H records componentwise* hsame *transport for prefix rows, windows, dyadic ledgers, regular readbacks, terminal seals, the prefix Cauchy ledger, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the same finite prefix Cauchy route.*
9. *P is the* Pkg *provenance over* $\mathsf{CantorSpaceCompletion}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{BishopIntervalCantor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CantorSpaceCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, W, D, R, E, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected infinite representatives, trichotomy, open-cover compactness, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9029.2 (CantorSpaceCompletion NameCert obligations)**.** *For every accepted* $\mathsf{CantorSpaceCompletion}^{\uparrow}$ *carrier*

$$K = (A, W, D, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorSpaceCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, prefix Cauchy ledger exposure, componentwise transport, displayed replay, package provenance, and local naming. A Real or Completion consumer may cite completed Cantor-space data only after those rows have been displayed.*

Proof. Project the carrier of Definition 9029.1. The visible completion-facing rows are exactly the binary prefix row A, finite windows W, dyadic diameter ledger D, regular readback R, terminal seal E, and prefix Cauchy ledger L.

Read the prefix Cauchy ledger L after the source rows. It says only that later displayed windows extend earlier binary cylinders and tighten the dyadic diameter bounds already recorded in D.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the rows stores quotient stream equality, a selected representative stream, trichotomy, open-cover compactness, countable choice, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 9029.3 (CantorSpaceCompletion prefix limit route)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{CantorSpaceCompletion}^{\uparrow}$ *carrier must factor its read through the displayed route*

$$\mathsf{CantorSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the prefix Cauchy ledger L and the structural rows H, C, P, N. The route may be compared with $\mathsf{CantorIntersection}^{\uparrow}$ *and* $\mathsf{BishopIntervalCantor}^{\uparrow}$ *sibling packets as a finite prefix handoff, but it cannot be replaced by quotient stream equality, a selected infinite branch, trichotomy, open-cover compactness, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 9029.2. They force any accepted consumer to read A, W, D, R, E, L before using the completion-facing surface.

The Cantor-space source row A supplies only displayed binary cylinders. The StreamName row W exposes finite observations of those cylinders, the DyadicRatCore row D records the displayed diameter bounds, and the RegSeqRat row R reads the same finite data as regular rational completion input before the terminal Real seal E is available.

Now use the sibling routes. $\mathsf{CantorIntersection}^{\uparrow}$ and $\mathsf{BishopIntervalCantor}^{\uparrow}$ may consume the same displayed prefix Cauchy ledger as a finite handoff, but neither sibling adds a quotient stream row, selected infinite branch, trichotomy row, open-cover compactness witness, countable-choice schedule, host real equality, or ambient completeness theorem.

Finally, H, C, P, N preserve only the displayed route under transport, replay, provenance, and local naming. Thus every prefix-limit read remains inside the finite Cantor-space completion packet named in the statement. □

Falsifiable prediction: An accepted $\mathsf{CantorSpaceCompletion}^{\uparrow}$ read is refuted if it reaches Real or Completion data without displaying the CantorSpace prefix row, StreamName finite windows, DyadicRatCore diameter ledger, RegSeqRat readback, RealUp seal, prefix Cauchy ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cantorspace, streamname, regseqrat, dyadicratcore, real, cantorintersection, bishopintervalcantor: $\mathsf{CantorSpaceCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds binary prefix cylinders, finite windows, dyadic diameter bounds, regular readback, terminal sealing, prefix Cauchy extension, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpaceCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorSpaceCompletion}^{\uparrow}$ packet is generated from a CantorSpace prefix row, StreamName finite windows, a DyadicRatCore diameter ledger, RegSeqRat readback, a RealUp seal, a prefix Cauchy ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9029.1, Definition 9029.2, Definition 9029.3, the falsifiable prediction, and the independence witness seed the finite prefix Cauchy completion route over CantorSpace, StreamName, DyadicRatCore, RegSeqRat, Real, CantorIntersection, BishopIntervalCantor, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite representatives, trichotomy, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix Cauchy ledger exactness, StreamName window exposure, DyadicRatCore diameter accounting, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorSpaceCompletionUp.

Chapter 9030

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusWitness}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy treatment of constructive real numbers, the RegularCauchyModulusW packet isolates the finite tail-modulus witness needed before a regular rational stream is sealed as a real object. It sits on the dyadic tolerance rows of chapter 291, the finite stream-window interface of chapter 126, the regular rational sequence route of chapter 301, and the terminal real-name seal of chapter 38. The packet records the witnessed tail window as finite NameCert data; it is not a selected limit, a quotient stream, a countable-choice principle, or an ambient completeness theorem.

Definition 9030.1 (Regular Cauchy modulus-witness carrier)**.** *A* $\mathsf{RegularCauchyModulusWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, R, T, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, naming the finite observations of the rational stream that are available to the witness.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording that the source stream is read through the regular rational sequence carrier rather than through a host sequence.*
3. *T is the tail-window row, naming a finite index budget after which the displayed comparison is served.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the tail-window request.*
5. *Q is the tail readback row, recording that every displayed comparison inside the window reads the same S, R, T, D route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *consumer-facing seal row, recording that a later real-name read may consume the witnessed route without receiving a selected limit or quotient equality.*
7. *H records componentwise* hsame *transport for S, R, T, D, Q, E and the local naming row.*
8. *C records displayed* Cont *replay through stream readback, regularity readback, tail-window service, dyadic tolerance lookup, seal handoff, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitness}^{\uparrow}}$ *row.*
The carrier exports only the displayed witness route. It contains no hidden selector, no second tolerance ledger, no quotient stream, no host real equality, no selected limit, no countable-choice row, and no ambient completeness coordinate.

Theorem 9030.2 (Regular Cauchy modulus-witness NameCert obligations)**.** *For the carrier of Definition 9030.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$W = (S, R, T, D, Q, E, H, C, P, N),$$

finite stream-window exposure through S, regular rational sequence exposure through R, tail-window exposure through T, dyadic tolerance exposure through D, readback exactness through Q, consumer-facing $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. Its displayed rows are exactly the stream window, regular rational sequence row, tail-window budget, dyadic tolerance, readback route, real-seal handoff, transport, replay, provenance, and local naming.

The four sibling dependencies are consumed in order. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular-sequence discipline, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance being served, and the $\mathsf{Real}^{\uparrow}$ row is only a downstream seal consumer.

Finally read H, C, P, N. These rows transport, replay, package, and name the same finite packet, and no carrier coordinate can be projected as a selected limit, quotient stream, countable choice, host real equality, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusWitness}^{\uparrow}$ packet is generated from a finite stream window, a regular rational sequence row, a displayed tail-window budget, a dyadic tolerance row, a readback route, a Real-facing seal handoff, componentwise transports, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitness}^{\uparrow}}$, obligation)

Scope closed: Definition 9030.1 and Definition 9030.2 bind the finite tail-modulus witness surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected limits, quotient streams, countable choice, host real equality, or ambient completeness for $\mathsf{RegularCauchyModulusWitness}^{\uparrow}$.

Upgrade path: Lean closure requires an encoding of the carrier, tail-window readback exactness, classifier transport, displayed replay, provenance, and local naming for the same finite packet.

Chapter 9031

A Concrete Naming Certificate for $\mathsf{MatroidGreedy}^{\uparrow}$

Following Edmonds' matroid greedy theorem, the $\mathsf{MatroidGreedy}^{\uparrow}$ packet records the finite exchange route by which a displayed weight-ordered scan produces a maximum-weight independent set in a finite matroid. It reads the independence and exchange rows of chapter 202, the augmenting-incidence discipline of chapter 234, the finite residual comparison surface of chapter 233, and the ordered-field weak-duality rows of chapter 235. The packet names the greedy certificate as finite BEDC data; it does not import a host optimizer, quotient equality of bases, selected maximal bases outside the ledger, or an ambient linear-programming oracle.

Definition 9031.1 (Matroid greedy carrier)**.** *A* $\mathsf{MatroidGreedy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, M, W, O, A, X, V, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite ground row shared with the source* $\mathsf{Matroid}^{\uparrow}$ *packet.*
2. *M is the source* $\mathsf{Matroid}^{\uparrow}$ *row, including independent-set admission, hereditary restriction, and exchange data.*
3. *W is the finite weight row, assigning an ordered value to every displayed ground element.*
4. *O is the weight-order scan row, recording the finite descending inspection order and tie ledger.*
5. *A is the accepted greedy prefix row, recording each admitted independent prefix.*
6. *X is the exchange witness row, pairing any competing independent row with the first displayed exchange point at which the greedy prefix dominates.*
7. *V is the value comparison row, read through finite sums and the* $\mathsf{LPDuality}^{\uparrow}$ *weak-duality discipline.*
8. *H records componentwise* hsame *transport for ground, source matroid, weights, order, accepted prefixes, exchange witnesses, value comparison, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for every consumer that reopens the scan or exchange route.*
10. *P is the* Pkg *provenance over* $\mathsf{MatroidGreedy}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MatroidGreedy}^\uparrow}$ *row.*
The carrier has no coordinate for a host greedy algorithm, quotient ground equality, selected bases outside the displayed scan, an unrestricted optimizer, or ambient linear-programming strong duality.

Theorem 9031.2 (Matroid greedy NameCert obligations)**.** *For every accepted* $\mathsf{MatroidGreedy}^\uparrow$ *carrier*

$$G = (E, M, W, O, A, X, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MatroidGreedy}^\uparrow}$ *surface consists exactly of carrier admission for G, source* $\mathsf{Matroid}^\uparrow$ *exposure through M, weight and scan exposure through W, O, greedy-prefix exposure through A, exchange-witness exposure through X, value-comparison exposure through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9031.1. The rows E, M, W, O, A, X, V are the complete finite optimization inventory: source ground and matroid, ordered weights, accepted scan prefixes, exchange witnesses, and value comparison.

Read M through chapter 202. Hereditary restriction licenses rejected scan steps, while the exchange row is the only source for replacing a competing independent element by the earliest greedy element.

The matching and network-flow siblings supply finite incidence and residual-comparison discipline for the exchange ledger, and the $\mathsf{LPDuality}^\uparrow$ sibling supplies the ordered value-comparison surface. The rows H, C, P, N transport, replay, source, and name exactly the same data.

Since no carrier coordinate stores a host optimizer, quotient ground equality, selected basis family, unrestricted search, or ambient strong-duality theorem, none of those objects can discharge a local $\mathsf{NameCert}_{\mathsf{MatroidGreedy}^\uparrow}$ obligation. □

Theorem 9031.3 (Matroid greedy exchange certificate)**.** *Every accepted* $\mathsf{MatroidGreedy}^\uparrow$ *packet exports a finite exchange certificate: for any displayed competing independent row over the same* $\mathsf{Matroid}^\uparrow$ *source, the row X supplies a finite sequence of exchange steps that replaces the competitor by the greedy prefix without lowering the displayed value row V. The exchange route factors through the same ground row E, source matroid row M, weight row W, scan row O, accepted-prefix row A, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

The certificate is finite and public. It is not a host optimality theorem, not quotient equality of bases, not a selected external maximum, not an unbounded augmenting search, and not an ambient linear-programming oracle.

Proof. Start with a displayed competitor over the same source matroid. The carrier in Definition 9031.1 makes the source ground E, independence row M, weights W, and scan order O visible before any exchange step is read.

Use the source exchange row from $\mathsf{Matroid}^\uparrow$. At the first scan position where the competitor differs from the greedy prefix, the exchange witness row X names a displayed replacement that preserves independence and does not move to a lower weight position.

Iterating this finite exchange ledger rewrites the competitor into the greedy prefix. The matching and network-flow sibling rows keep the incidence and residual bookkeeping finite, while the $\mathsf{LPDuality}^\uparrow$ value row V records that the displayed value comparison is monotone along the ledger.

The support rows H, C, P, N transport, replay, source, and name the same exchange certificate. Because the packet contains no host optimizer, quotient-basis equality, selected maximum,

unbounded search, or ambient LP oracle, the exported certificate is exactly the finite exchange route. □

Falsifiable prediction: A consumer that cites $\mathsf{MatroidGreedy}^{\uparrow}$ without displaying the source matroid, weight-order scan, accepted prefixes, exchange witness ledger, value comparison, hsame, Cont, Pkg, and local NameCert rows is not reading the carrier.

Independence witness: matroid, matching, networkflow, lpduality: $\mathsf{Matroid}^{\uparrow}$ supplies the exchange source, $\mathsf{Matching}^{\uparrow}$ supplies incidence pairing, $\mathsf{NetworkFlow}^{\uparrow}$ supplies residual comparison discipline, and $\mathsf{LPDuality}^{\uparrow}$ supplies finite ordered value comparison, but none carries the greedy scan plus exchange-certificate packet as a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MatroidGreedy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MatroidGreedy}^{\uparrow}$ packet is generated from a finite ground row, source MatroidUp row, finite weight row, weight-order scan row, accepted greedy prefix row, exchange witness row, value comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MatroidGreedy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9031.1, Definition 9031.2, and Definition 9031.3 seed the finite greedy exchange certificate over MatroidUp, MatchingUp, NetworkFlowUp, LPDualityUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MatroidGreedy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host matroid optimization, quotient ground equality, selected global bases, unrestricted search, ambient linear-programming strong duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scan-prefix exactness, exchange-witness exactness, value comparison, transport, replay, provenance, and local naming for BEDC.Derived.MatroidGreedyUp.

Chapter 9032

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$ packet records the finite-window criterion by which two regular Cauchy rational names exhibit positive apartness before any quotient real equality, trichotomy principle, or host real order is consulted. It sits between the regular rational names of chapter 301, the observation schedule of chapter 126, the positive dyadic gap evidence of chapter 291, the real-name seal of chapter 38, and the finite separation interface of chapter 304.

Definition 9032.1 (Regular Cauchy apartness criterion carrier)**.** *A* $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, R, W, D, A, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, including their displayed rational-value reads and regular Cauchy modulus rows.*
2. *W is a common finite* $\mathsf{StreamName}^{\uparrow}$ *window that names exactly the finitely many indices inspected on both rows.*
3. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap evidence read from the windowed endpoint approximations.*
4. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row produced after the* $\mathsf{Real}^{\uparrow}$ *seal reads the two regular Cauchy rows as real names.*
5. *H records componentwise* hsame *transport for L, R, W, D, A, C, P, N and for the two windowed endpoint readbacks.*
6. *C records displayed* Cont *replay from an apartness request through the two regular Cauchy rows, the common window, the positive dyadic gap, the real seal, and the separation row.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ApartnessReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, W, D, A, H, C, P, N$. It has no coordinate for quotient equality of reals, a total order, trichotomy, excluded middle, a selected limit, a host metric, or a completion theorem.

Theorem 9032.2 (Regular Cauchy apartness criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$ *carrier*

$$Q = (L, R, W, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}}$ *surface consists exactly of left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, common finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap evidence,* $\mathsf{Real}^{\uparrow}$ *seal compatibility,* $\mathsf{ApartnessReal}^{\uparrow}$ *separation-row handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 9032.1. The rows L and R supply the two regular rational names, while W restricts the inspection to a common finite window.

Read the dyadic gap row D. Because D is positive evidence over the displayed windowed endpoint readbacks, the criterion uses a finite witness rather than a total comparison of completed reals.

The row A is then obtained by replaying the $\mathsf{Real}^{\uparrow}$ seal and the $\mathsf{ApartnessReal}^{\uparrow}$ finite-window separation interface on the same displayed data. The structural rows H, C, P, N transport, replay, source, and name only these finite rows, so the listed obligations exhaust the local NameCert surface. □

Theorem 9032.3 (Regular Cauchy apartness criterion Real seal handoff)**.** *Let*

$$Q = (L, R, W, D, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{ApartnessReal}^{\uparrow}$ *consumer that reads the positive apartness of the two regular Cauchy names must first expose* L, R, W, D *and the structural rows* H, C, P, N*. The handoff is the finite route*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{ApartnessReal}^{\uparrow},$$

not quotient equality of completed reals, a host real order, trichotomy, a selected limit, or a choice principle over tails.

Proof. Start from the two $\mathsf{RegSeqRat}^{\uparrow}$ rows and the common window. The finite window W identifies which rational approximants are inspected, so the handoff has an explicit observation schedule before the real-name seal is consumed.

Replay the positive dyadic gap D through the displayed endpoint readbacks. This produces the separation data required by the $\mathsf{ApartnessReal}^{\uparrow}$ finite-window row after the $\mathsf{Real}^{\uparrow}$ seal reads the same regular Cauchy names.

Finally use H, C, P, N to transport, replay, source, and locally name the handoff. Since every consumer must pass through these rows, the public output is positive apartness evidence only, with no quotient equality, trichotomy, host order, selected limit, or tail-choice principle added. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9032.1, Definition 9032.2, and Definition 9032.3 seed the finite-window apartness criterion over regular Cauchy rational rows, common stream windows, positive dyadic gaps, real seals, apartness handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of reals, trichotomy, total real order, host metric apartness, selected limits, completion, or choice over tails.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, positive dyadic-gap stability, real-seal handoff, apartness-row handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyApartnessCriterionU

Constructive story: A RegularCauchyApartnessCriterion$^{\uparrow}$ packet is generated from two RegSeqRatUp rows, a common finite StreamNameUp window, positive DyadicRatCoreUp gap evidence, RealUp seal compatibility, an ApartnessRealUp separation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Chapter 9033

A Concrete Naming Certificate for $\mathsf{BishopRealIntervalHalving}^{\uparrow}$

Following Bishop and Bridges, interval halving is a finite located-interval operation used to keep a constructive real search inside explicit rational bounds. The $\mathsf{BishopRealIntervalHalving}^{\uparrow}$ packet isolates that operation as L10 NameCert data after the dyadic carrier of chapter 291, the stream naming surface of chapter 126, the regular rational sequence packet of chapter 301, the constructive real packet of chapter 38, and the located-real surface of **??**.

Definition 9033.1 (BishopRealIntervalHalving carrier)**.** *A* $\mathsf{BishopRealIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic endpoint row;*
2. *U is the upper dyadic endpoint row;*
3. *M is the dyadic midpoint row, read as the exact average of L and U inside the dyadic carrier of chapter 291;*
4. *S is the selected-half row, choosing either the lower half $[L, M]$ or the upper half $[M, U]$ from displayed located evidence;*
5. *D is the dyadic tolerance row recording the radius before and after one halving step;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected endpoint sequence;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, asserting that the selected half remains inside the same constructive-real envelope;*
8. *H records componentwise* hsame *transport for endpoint, midpoint, selected-half, tolerance, readback, seal, replay, provenance, and local certificate rows;*
9. *C records displayed* Cont *routes for endpoint inspection, midpoint replay, selected-half handoff, tolerance decrease, regular-sequence readback, real-seal replay, and consumer use;*
10. *P is the provenance row over the cited dyadic, stream-name, regular-sequence, real, and located-real siblings;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealIntervalHalving}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, Heine–Borel compactness, quotient-real equality, global choice of nested intervals, or countable choice. It records one finite bisection step with all endpoints, tolerance, readback, seal, transport, replay, provenance, and naming rows visible.

Theorem 9033.2 (BishopRealIntervalHalving NameCert obligations)**.** *For the carrier of Definition 9033.1, the local* $\mathsf{NameCert}_{\mathsf{BishopRealIntervalHalving}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *midpoint exactness: M is displayed as the dyadic midpoint of the endpoint rows L and U;*
2. *selected-half containment: the selected row S names either $[L, M]$ or $[M, U]$ and remains contained in the original located interval;*
3. *dyadic radius decrease: the tolerance row D records the halved radius of the selected interval;*
4. *Real seal nonescape: the seal row E keeps the selected half inside the same* $\mathsf{Real}^{\uparrow}$ *envelope read through R;*
5. *provenance exhaustion: the rows H, C, P, N replay only the dyadic endpoint, StreamName, RegSeqRat, Real, LocatedReal, transport, continuation, package, and local NameCert sources displayed in K.*

The obligations produce a finite bisection certificate, not an interval-completeness theorem, trichotomy principle, Heine–Borel theorem, quotient-real construction, uniform search over arbitrary predicates, or countable-choice principle.

Proof. The midpoint obligation is read directly from the M row together with the dyadic endpoint rows L and U. Since all three rows are part of the finite packet, no external real-number quotient or hidden completion operation is required to name the midpoint.

The selected-half and radius obligations are the rows S and D. The selected half is one of the two displayed dyadic subintervals, and the tolerance row records that this subinterval has the halved radius.

The readback and seal obligations are the rows R and E. The regular rational sequence readback supplies the finite endpoint data consumed by the constructive-real seal, while the seal records nonescape inside the same real envelope.

Finally, H, C, P, N exhaust the transport, replay, provenance, and naming rows. They cite only the dyadic, stream-name, regular-sequence, real, located-real, BHist, hsame, Cont, Pkg, and NameCert surfaces already displayed in the packet, so the certificate exports a single finite halving step and none of the refused global principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealIntervalHalving}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealIntervalHalving}^{\uparrow}$ packet is generated from dyadic endpoints, an exact midpoint, a selected half, a dyadic tolerance row, RegSeqRat readback, a Real seal, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealIntervalHalving}^{\uparrow}}$, seed)

Scope closed: Definition 9033.1 and Definition 9033.2 bind the finite interval-halving carrier and its midpoint, containment, radius, seal, and provenance obligations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealIntervalHalving}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close interval completeness, trichotomy, Heine–Borel compactness, quotient-real equality, uniform predicate search, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Formalization requires a future Lean carrier and theorem under BEDC.Derived.BishopRealIntervalHalvingUp for midpoint exactness, selected-half containment, dyadic radius decrease, Real seal nonescape, and provenance exhaustion.

Chapter 9034

A Concrete Naming Certificate for $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$

Following Bishop and Bridges on completion by Cauchy data, the $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$ packet names the finite universal-factorization surface for a Cauchy-filter completion. It sits over the Cauchy-filter carrier of chapter 333, the completion boundary of chapter 1569, the abstract completion interface of chapter 95, the metric source of chapter 57, the continuous-map handoff of chapter 130, and the terminal real-name seal of chapter 38. The packet records the universal property as displayed rows only; it does not quotient filters, select limits, invoke choice, use host function extensionality, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9034.1 (Cauchy-filter completion universal carrier)**.** *A* $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, D, Y, u, f, g, U_0, H, C, P, N)$$

with the following rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing only displayed filter bases and finite Cauchy-window requests.*
2. *D is the dense embedding row from the metric source into the completion boundary.*
3. *Y is the completion target row, read through* $\mathsf{Completion}^{\uparrow}$ *and the Cauchy-filter completion surface.*
4. *u is the uniformly continuous source map row, routed through the* $\mathsf{ContinuousMap}^{\uparrow}$ *sibling.*
5. *f is the factor map row from the completion target to the receiving metric object.*
6. *g is the uniqueness ledger row comparing two displayed factor rows only after F, D, Y, u, f are visible.*
7. *U_0 is the universal-factorization transport row, recording that u is replayed through D and then through f.*
8. *H records componentwise* hsame *transport for the source Cauchy filters, dense embedding, completion target, source map, factor map, uniqueness ledger, universal row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the factorization route.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, Y, u, f, g, U_0, H, C, P, N$. It has no coordinate for quotient filters, selected completion points, countable choice, host function extensionality, or ambient completeness.

Lean4 (statement) `BEDC.Derived.CauchyFilterCompletionUniversalUp`

Theorem 9034.2 (Cauchy-filter completion universal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$ *carrier $U = (F, D, Y, u, f, g, U_0, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}}$ *surface consists exactly of carrier admission for the source Cauchy-filter row, dense embedding row, completion target row, uniformly continuous source-map row, factor-map row, uniqueness ledger, universal-factorization transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may replace these rows by quotient classes of Cauchy filters, selected limits, countable choice, host-level function extensionality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9034.1. The only available coordinates are $F, D, Y, u, f, g, U_0, H, C, P, N$.

The source side is finite. The Cauchy-filter row F supplies displayed filter-base and Cauchy-window data, while the dense embedding row D and target row Y place that data inside the completion boundary named by the sibling completion chapters.

The mapping side is also row-local. The source map u is consumed through the continuous-map handoff, the factor row f is visible only after F, D, Y, u have been displayed, and the uniqueness ledger g compares factor rows inside that same packet.

Transport, replay, provenance, and local naming are the rows H, C, P, N. Since no carrier coordinate records quotient filters, selected limits, choice schedules, host function extensionality, or ambient completeness, no NameCert obligation can read such data. □

Theorem 9034.3 (Cauchy-filter completion universal factorization)**.** *Every accepted* CauchyFilterCompletionUniver *packet factors a displayed uniformly continuous source map through the displayed completion target by the finite route*

$$F, D, Y, u \longmapsto f \longmapsto g \longmapsto U_0$$

and then through H, C, P, N. The route is universal only over the displayed Cauchy-filter, dense-embedding, completion-target, continuous-map, factor-map, uniqueness, transport, replay, provenance, and local naming rows. It does not assert quotient-filter completion, selected completed points, countable choice, host-level function extensionality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Read the source Cauchy filters through F and the dense embedding through D. The completion target Y is available only as the displayed completion boundary supplied by the sibling completion surfaces.

Use the source-map row u as the only uniformly continuous input. The factor row f is attached after F, D, Y, u have been exposed, and the uniqueness ledger g compares factor rows inside that same displayed route.

Finally read the universal-factorization row U_0 and the structural rows H, C, P, N. Componentwise transport and replay preserve the same route, while provenance and local naming certify only this finite packet. The excluded host-side objects have no row in the carrier, so no consumer can extract them from the factorization. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}}, \mathsf{seed})$

Scope closed: The chapter records the seed carrier, NameCert obligations, and displayed universal-factorization route for Cauchy-filter completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.CauchyFilterCompletionUniversalUp`

Bridge status: none

Not claimed: This chapter does not close quotient-filter completion, selected completed points, countable choice, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: To reach scoped closure, formalize the carrier rows and prove the NameCert-obligation and factorization routes in Lean.

Constructive story: A $\mathsf{CauchyFilterCompletionUniversal}^{\uparrow}$ packet is generated from source Cauchy filters, dense embedding, completion target, uniformly continuous source map, factor map, uniqueness ledger, universal transport, replay, provenance, and a local NameCert row.

Chapter 9035

A Concrete Naming Certificate for SeparatedCompletionUniversalArrow$^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley's separated uniform-space language, the SeparatedCompletionUniversalArrow$^{\uparrow}$ packet records the finite universal-arrow surface for a separated completion without quotienting Cauchy names or selecting representatives. It sits after the separated-reflection route of chapter 4151, the metric-completion carrier of chapter 1506, the Cauchy-completion universal arrow of chapter 3997, the finite StreamName windows of chapter 126, the RegSeqRat readback of chapter 301, and the terminal Real seal of chapter 38. The packet exposes only the displayed separated universal-arrow route consumed by Real and Completion readers.

Definition 9035.1 (Separated completion universal-arrow carrier)**.** *A* SeparatedCompletionUniversalArrow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, A, W, Q, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the* SeparatedReflection$^{\uparrow}$ *row exposing the separated classifier and zero-distance identity boundary.*
2. *M is the* MetricCompletion$^{\uparrow}$ *completion row admitted only after the separated reflection has been displayed.*
3. *A is the* CauchyCompletionUniversalArrow$^{\uparrow}$ *row carrying the unit and universal-arrow handoff.*
4. *W is the* StreamName$^{\uparrow}$ *finite-window row read by the completion consumer.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the same windows.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row.*
7. *F records the factorization route from a separated source map through S, M, A, W, Q, E.*
8. *H records componentwise* hsame *transport for the separated reflection, completion, universal arrow, finite windows, readback, seal, factorization, replay, provenance, and local name.*
9. *C records displayed* Cont *replay of the same route.*
10. *P is the* Pkg *provenance over* SeparatedCompletionUniversalArrow$^{\uparrow}$, SeparatedReflection$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CauchyCompletionUniversalArrow$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected Cauchy representatives, countable choice, host function equality, trichotomy, or ambient Real completeness.

Theorem 9035.2 (Separated completion universal-arrow NameCert obligations)**.** *For an accepted* $\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}$ *carrier* $U = (S, M, A, W, Q, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}}$ *surface consists exactly of separated-reflection exposure through S, metric-completion exposure through M, Cauchy universal-arrow exposure through A, finite StreamName window exposure through W, RegSeqRat readback through Q, Real sealing through E, factorization through F, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9035.1. Its rows display the separated reflection, completion object, universal arrow, finite window, regular rational readback, Real seal, factorization, transport, replay, provenance, and local name.

The separatedness obligation is held in S before completion is consumed. The completion row M and universal-arrow row A are downstream of that separated boundary, so a consumer cannot read a non-separated quotient representative as the source of the factorization.

The Real-facing part is held in W, Q, E. These rows expose finite windows, regular rational readback, and the terminal seal before any completion-facing consumer cites the packet.

Finally F, H, C, P, N factor, transport, replay, source, and name only those displayed rows. Since no excluded principle appears in the carrier, the local NameCert surface is exhausted by the finite route above. □

Theorem 9035.3 (Separated completion universal-arrow factorization route)**.** *Every accepted* $\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}$ *universal-arrow read factors through*

$$S \longmapsto M \longmapsto A \longmapsto W \longmapsto Q \longmapsto E \longmapsto (F, H, C, P, N).$$

In particular, the completion-facing consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{SeparatedReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have all been displayed.*

Proof. Apply the NameCert obligations of Definition 9035.2. They force the separated reflection row S, the metric-completion row M, and the Cauchy universal-arrow row A to be visible before the factorization row F can be consumed.

Next read the finite observation segment W, Q, E. The StreamName and RegSeqRat rows expose the inspected finite windows and rational readback before the terminal Real seal is named.

The rows H, C, P, N preserve the same route under componentwise transport, continuation replay, provenance, and local naming. Because the carrier stores no quotient completion equality, selected representative, countable-choice schedule, host function equality, trichotomy, or ambient Real completeness row, the displayed factorization route is the only accepted route to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}$ packet is generated from a separated-reflection row, metric-completion row, Cauchy universal-arrow row, finite StreamName windows, RegSeqRat readback, a RealUp seal, factorization row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionUniversalArrow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9035.1, Definition 9035.2, and Definition 9035.3 seed the separated completion universal-arrow route over SeparatedReflection, MetricCompletion, CauchyCompletionUniversalArrow, StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparatedCompletionUniversalArrow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host function equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separatedness exactness, completion-object exposure, universal-arrow factorization, finite-window readback, Real seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionUniversalArrowUp.

Chapter 9036

A Concrete Naming Certificate for $\mathsf{SeparatedCauchyFilter}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ packet records the finite route by which a Cauchy-filter completion read is first reflected through separated metric data before any Real-facing consumer may inspect the terminal seal. It sits between the filter surface of chapter 333, the completion route of chapter 1506, the separated reflection surface of chapter 1790, the finite stream observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet names only displayed filter-base, separated-reflection, finite-window, regular-readback, and Real-seal rows; it does not identify quotient filters, choose representatives, invoke countable choice, import host limits, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9036.1 (Separated Cauchy filter carrier)**.** *A* $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, M, Q, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. F *is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, exposing only a finite filter-base request and its displayed refinement data.*
2. M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row, naming the completion-facing carrier through which the filter request is read.*
3. Q *is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *row, recording the separated reflection boundary before any equality-facing completion read is available.*
4. T *is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row used to inspect the reflected filter route.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed observation windows.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after* F, M, Q, T, R *remain visible together.*
7. H *records componentwise* hsame *transport for the filter source, completion route, separated reflection, stream observation, regular readback, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a completion request through the same finite rows.*
9. P *is the* Pkg *provenance over* $\mathsf{SeparatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetricReflectio}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SeparatedCauchyFilter}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient filter equality, selected representatives, countable choice, hidden limit selection, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9036.2 (Separated Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ *carrier*

$$S = (F, M, Q, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedCauchyFilter}^{\uparrow}}$ *surface consists exactly of Cauchy-filter source admission, metric-completion route admission, separated-reflection exposure, finite stream observation, regular rational readback, terminal Real sealing, componentwise transport, displayed continuation replay, provenance, and local naming. No obligation row may replace these displayed rows by quotient filter equality, a selected representative, countable choice, host limit selection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9036.1. The analytic route is carried by F, M, Q, T, R, E, and the structural route is carried by H, C, P, N.

The source row F exposes a finite Cauchy-filter base request. The completion row M records the metric-completion route for that request, while Q inserts the separated reflection boundary before any equality-facing read can occur.

The finite observation and readback rows then make the route inspectable. The stream row T displays the observation windows, and R records the regular rational readback extracted from those windows before the terminal Real seal E is consumed.

Transport, replay, provenance, and naming preserve the same rows through H, C, P, N. Since no carrier coordinate stores quotient filter equality, selected representatives, countable choice, host limit selection, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 9036.3 (Separated Cauchy filter reflection soundness)**.** *Every Real-facing read of an accepted* $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{SeparatedMetricReflection}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently the packet exports a separated completion-facing filter read without exporting quotient filter equality, selected representatives, countable choice, host limits, or ambient Real completeness.

Proof. Start with an accepted packet $S = (F, M, Q, T, R, E, H, C, P, N)$. The only source row is F, and the only completion route row is M.

The separated reflection row Q is placed between the completion route and the observation rows. Thus a consumer cannot read the terminal seal as an unreflected filter quotient or as an equality of representative filters.

The finite observation row T and regular readback row R supply the displayed data used by the terminal seal E. The structural rows H, C, P, N transport, replay, source, and name this same route rather than adding a host limit operation or choice schedule.

Therefore every accepted Real-facing read follows the displayed route, and any consumer that skips the separated-reflection, finite-window, or regular-readback rows is outside the carrier. □

Falsifiable prediction: An accepted $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ Real-facing read that reaches the terminal seal without displaying a CauchyFilter source row, MetricCompletion route row, SeparatedMetricReflection boundary row, StreamName observation row, and RegSeqRat readback row refutes the carrier.

Independence witness: cauchyfilter, metriccompletion, separatedmetricreflection, streamname, regseqrat, real: $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds filter source, completion route, separated reflection, finite observation, regular readback, Real sealing, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCauchyFilter}^{\uparrow}$ packet is generated from CauchyFilter source, MetricCompletion route, SeparatedMetricReflection boundary, StreamName observation, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9036.1, Definition 9036.2, Definition 9036.3, the falsifiable prediction, and the independence witness seed the separated Cauchy-filter completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient filter equality, selected representatives, countable choice, host limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base exposure, completion-route admission, separated-reflection soundness, finite observation, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCauchyFilterUp.

Chapter 9037

A Concrete Naming Certificate for $\mathsf{OreLocalization}^{\uparrow}$

Following Ore localization of noncommutative rings, the $\mathsf{OreLocalization}^{\uparrow}$ packet records the finite denominator calculus needed to read fractions $s^{-1}r$ or rs^{-1} without treating localization as generic parameter transport. It sits after the multiplicative monoid surface of chapter 40, the ring operation rows of chapter 43, and the ideal-facing absorption surface of chapter 81. The packet names an admissible denominator set, left and right Ore squares, cancellation witnesses, fraction rows, transport, replay, provenance, and local naming; it does not import a host quotient ring, a selected common denominator outside the ledger, commutativity, or a classical total ring of fractions.

Definition 9037.1 (OreLocalization carrier)**.** *An* $\mathsf{OreLocalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, S, O, C, F, D, H, T, P, N)$$

with the following displayed rows:

1. *R is the ambient* $\mathsf{Ring}^{\uparrow}$ *row, including the additive, multiplicative, and distributive operation data read through chapter 43;*
2. *S is the denominator* $\mathsf{Monoid}^{\uparrow}$ *row, naming the multiplicatively closed set used by the localization route;*
3. *O is the Ore-square ledger, listing the displayed left and right common-multiple squares used to move a numerator past a denominator;*
4. *C is the cancellation ledger, recording the denominator-side cancellation witnesses allowed by the same Ore surface;*
5. *F is the finite fraction row, whose entries are displayed numerator-denominator pairs and not host quotient classes;*
6. *D is the denominator ledger, recording which denominator row is consumed by each fraction, Ore square, cancellation step, and multiplication handoff;*
7. *H records componentwise* hsame *transport for the ring, denominator, Ore-square, cancellation, fraction, denominator-ledger, replay, provenance, and local naming rows;*
8. *T records displayed* Cont *replay for denominator admission, Ore-square formation, fraction multiplication, cancellation, and consumer readback;*

9. *P is the* Pkg *provenance over* OreLocalization$^{\uparrow}$, Ring$^{\uparrow}$, Monoid$^{\uparrow}$, Ideal$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{OreLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, O, C, F, D, H, T, P, N$.

Lean4 (definition) `BEDC.Derived.OreLocalizationUp.OreLocalizationUp`
Lean4 ✓ `BEDC.Derived.OreLocalizationUp.OreLocalizationTasteGate single carrier alignment`

Theorem 9037.2 (OreLocalization NameCert obligations)**.** *For the carrier of Definition 9037.1, the local* NameCert$_{\text{OreLocalization}^{\uparrow}}$ *surface consists exactly of ambient ring admission, denominator-monoid admission, left and right Ore-square exposure, denominator-side cancellation exposure, finite fraction-row admission, denominator-ledger readback, componentwise transport, displayed continuation replay, provenance, and local naming. Every accepted localization read must pass through these rows before a fraction multiplication or denominator handoff is public.*

Proof. Project the carrier $L = (R, S, O, C, F, D, H, T, P, N)$. The ring and denominator rows are visible first, so every numerator and denominator read is grounded in the displayed Ring$^{\uparrow}$ and Monoid$^{\uparrow}$ certificates.

The Ore-square and cancellation rows are O and C. They provide the finite common-multiple and cancellation witnesses used to compare displayed fraction rows; no commutative shortcut or host quotient equality is available.

The fraction and denominator ledgers are F and D. They record which denominator is consumed at each multiplication, comparison, and handoff step.

Finally, H, T, P, N transport, replay, source, and name the same finite data. Since the packet has no coordinate for a host localization object, selected representatives outside the ledger, total quotient rings, or ambient ideal equality, these rows exhaust the local NameCert obligations. □

Theorem 9037.3 (OreLocalization denominator ledger)**.** *In an accepted* OreLocalization$^{\uparrow}$ *carrier $L = (R, S, O, C, F, D, H, T, P, N)$, every public fraction row in F has its denominator named in D, every Ore-square witness in O consumes denominator rows from D, and every cancellation witness in C returns to the same denominator ledger. A consumer cannot cite a localized product, comparison, or handoff unless the relevant denominator route is displayed in D.*

Proof. Read the fraction row F together with the denominator ledger D. The carrier admits fractions only as displayed numerator-denominator pairs, so each fraction coordinate has a corresponding denominator entry.

Use the Ore-square ledger O. A common-multiple square is accepted only when the denominator rows it moves across are already listed in D.

Use the cancellation ledger C for the final handoff. Cancellation may shorten or compare displayed fraction rows, but it does not create a denominator outside the ledger. The transport, replay, provenance, and naming rows H, T, P, N preserve the same denominator route for downstream consumers. □

Closure status for NameCert$_{\text{OreLocalization}^{\uparrow}}$.

Constructive story: An OreLocalization$^{\uparrow}$ packet is generated from an ambient RingUp row, a denominator MonoidUp row, Ore-square witnesses, cancellation witnesses, finite fraction rows, a denominator ledger, transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{OreLocalization}^{\uparrow}}$, seed)

Scope closed: Definition 9037.1, Definition 9037.2, and Definition 9037.3 seed the finite denominator-localization carrier and its NameCert and denominator-ledger obligations.

Formal verification: FormalStatus(NameCert$_{\mathsf{OreLocalization}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not construct a host quotient ring, prove universal localization, assume commutativity, select common denominators outside the ledger, close ideal equality, provide Lean verification, or bridge check the route.
Upgrade path: Obligation closure requires checked ring admission, denominator-monoid admission, Ore-square exposure, cancellation exposure, fraction-row admission, denominator-ledger readback, transport, replay, provenance, and local naming for BEDC.Derived.OreLocalizationUp.

Chapter 9038

A Concrete Naming Certificate for $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$

The $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$ packet realizes the right-block flux-reservoir coefficient of Definition 9346.49 as finite NameCert data. It records the right-block boundary count, the cube-degree normalization, the resulting local ratio, the static cut source, transport, replay, provenance, and local naming rows before any impedance or coupling reader can cite the coefficient. The packet sits beside the Zeckendorf phase shell of chapter 8811, the static Green boundary-flux row of chapter 8769, the flux-operator geometry surface of chapter 8821, and the beta-sign selection packet of chapter 8846.

Definition 9038.1 (RightBlockFluxReservoirCoefficient carrier)**.** *A* $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, E, G, R, S, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the right-block row, naming the six-element right edge-flux block inside the Window-six cube;*
2. *E is the boundary-count row, recording the displayed boundary size* 32*;*
3. *G is the cube-degree row, recording the degree* 6 *used by the local directed-edge normalization;*
4. *R is the normalized-ratio row, displaying the coefficient* $32/(6 \cdot 6) = 8/9$*;*
5. *S is the static-cut row, tying the coefficient to the right-boundary cut-flux source rather than to the dynamic Green-relaxation readout;*
6. *T is the transport row from the coefficient packet to impedance or coupling consumers;*
7. *H records componentwise* hsame *transport for block, boundary, degree, ratio, static-cut, consumer, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay for block admission, boundary counting, degree normalization, ratio readback, static-cut handoff, and consumer use;*
9. *P is the* Pkg *provenance over* $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$*,* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$*,* StaticGreenBoundaryFl FluxOperatorGeometry$^{\uparrow}$*,* $\mathsf{BetaSignSelection}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, E, G, R, S, T, H, C, P, N$.

Theorem 9038.2 (RightBlockFluxReservoirCoefficient NameCert obligations)**.** *For the carrier of Definition 9038.1, the local* $\mathsf{NameCert}_{\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}}$ *surface consists exactly of right-block admission, boundary-count exposure, cube-degree exposure, normalized-ratio readback, static-cut source exposure, consumer transport, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. The packet therefore turns the coefficient* $8/9$ *into a finite boundary-count and normalization certificate, not a free-floating numerical parameter.*

Proof. Project the carrier $K = (B, E, G, R, S, T, H, C, P, N)$. The block, boundary, degree, ratio, static-cut, and consumer rows are all visible coordinates of one finite packet.

The boundary-count and degree rows determine the normalized ratio row. The coefficient is read as $32/(6 \cdot 6) = 8/9$ through E, G, R, so it cannot be replaced by an untracked scalar or by the global cut-flux density.

The static-cut and transport rows are S and T. They route the coefficient to impedance or coupling consumers while keeping it separate from the dynamic Green-relaxation layer.

Finally, H, C, P, N preserve the same finite rows under transport, replay, provenance, and local naming. No additional coordinate supplies a physical continuum model, a dynamic relaxation coefficient, or a hidden normalization convention. □

Theorem 9038.3 (RightBlockFluxReservoirCoefficient boundary ratio)**.** *In an accepted* RightBlockFluxReservoirCo *carrier $K = (B, E, G, R, S, T, H, C, P, N)$, the public coefficient row R is available only after the right-block row B, boundary-count row E, cube-degree row G, and static-cut row S have been displayed. The ratio readback is the local boundary escape value* $8/9$*, and it does not claim the global cut-flux density, the one-way mismatch mean, or a dynamic Green-relaxation coefficient.*

Proof. Read the displayed block row B and boundary-count row E. They identify the finite right edge-flux block and its boundary size before any ratio is public.

Use the cube-degree row G to normalize the boundary count locally. The ratio row R is therefore tied to the directed-edge budget of the block, not to the full cube budget.

The static-cut row S fixes the readout mode. It keeps the coefficient in the static right-boundary source layer, while T, H, C, P, N transport, replay, source, and name the same finite readback for downstream consumers. □

Falsifiable prediction: If an accepted RightBlockFluxReservoirCoefficientUp packet exposes the coefficient without the right-block row, boundary-count row, cube-degree row, and static-cut row, then the packet is refuted as a NameCert carrier for the right-block flux-reservoir coefficient.

Independence witness: zeckendorf-phase-shell, static-green-boundary-flux, flux-operator-geometry, beta-sign-selection: none is bijective to $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$ because those siblings respectively bind phase-shell, Green-exact cut-source, operator-geometry, or sign-selection rows, while this packet binds the right-block boundary count, cube-degree normalization, local ratio, static-cut source, transport, replay, provenance, and local NameCert rows.

The Lean follow-up file `lean4/BEDC/Derived/RightBlockFluxReservoirCoefficientUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}$ and FieldFaithful RightBlockFluxReservoirCoeffici instances. The nontriviality witness must distinguish an accepted coefficient packet from a packet with a missing boundary-count or cube-degree row, and the field-faithfulness witness must preserve $B, E, G, R, S, T, H, C, P, N$ as separate rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RightBlockFluxReservoirCoefficient}^{\uparrow}}$.

Constructive story: A RightBlockFluxReservoirCoefficient$^{\uparrow}$ packet is generated from a right-block row, boundary-count row, cube-degree row, normalized-ratio row, static-cut row, consumer transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RightBlockFluxReservoirCoefficient}^{\uparrow}}$, seed)

Scope closed: Definition 9038.1, Definition 9038.2, and Definition 9038.3 seed the finite boundary-count, cube-degree, ratio, static-cut, transport, replay, provenance, and naming surface.

Formal verification: FormalStatus(NameCert$_{\text{RightBlockFluxReservoirCoefficient}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim a physical continuum model, dynamic Green-relaxation coefficient, global cut-flux density, one-way mismatch theorem, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked right-block admission, boundary-count exposure, cube-degree exposure, normalized-ratio readback, static-cut exposure, transport, replay, provenance, local naming, and the required Nontrivial and FieldFaithful instances for BEDC.Derived.RightBlockFluxReservoirCoefficientUp.

Chapter 9039

A Concrete Naming Certificate for $\mathsf{ConvexFunction}^{\uparrow}$

Following Rockafellar's convex analysis and Boyd–Vandenberghe's convex optimization presentation, the $\mathsf{ConvexFunction}^{\uparrow}$ packet records the finite route by which a displayed convex domain, vector source, real codomain, rational affine weights, and comparison rows name a convex-function surface. It sits over the convex carrier of chapter 208, the real comparison boundary of chapter 38, the vector source of chapter 47, and the rational coefficient source of chapter 37. The packet supplies the source row consumed by chapter 2235 and chapter 321; it is a finite NameCert surface, not a host function space or an ambient completeness principle.

Definition 9039.1 (ConvexFunction carrier)**.** *A* $\mathsf{ConvexFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, V, R, A, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the convex-domain row, recording the accepted* $\mathsf{ConvexSet}^{\uparrow}$ *carrier whose elements may be queried.*
2. *V is the vector-source row, recording the displayed* $\mathsf{VecSpace}^{\uparrow}$ *operations used to form affine expressions.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *codomain and readback row, recording the real-valued output names and comparison interface.*
4. *A is the affine-weight route, recording finite rational weights through* $\mathsf{Rat}^{\uparrow}$*, nonnegative checks, and the displayed finite-sum-to-one row.*
5. *M is the midpoint or finite affine-combination ledger, recording the source points, weighted vector expression, and the proof row that the result remains inside D.*
6. *E is the epigraph and comparison row, recording the displayed inequality between the function value at the affine point and the corresponding affine combination of output reads.*
7. *H records componentwise* hsame *transport for D, V, R, A, M, E and the local naming row.*
8. *C records displayed* Cont *replay from the domain, vector, rational-weight, real-comparison, transport, provenance, and local name rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ConvexFunction}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvexFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, V, R, A, M, E, H, C, P, N$. It has no coordinate for a host convex function, quotient equality of functions or real names, selected extrema, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9039.2 (ConvexFunction NameCert obligations)**.** *For every accepted* $\mathsf{ConvexFunction}^{\uparrow}$ *carrier $F = (D, V, R, A, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConvexFunction}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the displayed convex domain, vector source, real codomain and readback row, affine-weight route, midpoint or finite affine-combination ledger, epigraph and comparison row, transport, replay, provenance, and local naming;*
2. *affine-combination accountability, requiring every weight and finite affine expression read by a consumer to pass through A and M;*
3. *convexity comparison exactness, requiring every displayed convexity inequality to be the row recorded in E over the* $\mathsf{Real}^{\uparrow}$ *readback interface R;*
4. *transport and replay stability, requiring componentwise* hsame *transport and* Cont *replay to preserve the same domain, vector, rational-weight, real-comparison, provenance, and local naming route;*
5. *ledger nonescape from host functions, quotient real equality, selected extrema, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet $F = (D, V, R, A, M, E, H, C, P, N)$. The carrier definition exposes exactly the convex domain, vector source, real codomain, affine-weight, affine-combination, comparison, transport, replay, provenance, and naming rows.

The affine data are read before the comparison row. Thus A supplies the rational weights and finite-sum row, while M supplies the displayed midpoint or finite affine expression and its return to the convex domain.

The convexity comparison is then read only through E and R. Transport and replay are componentwise through H and C, and provenance and naming are carried by P and N. Since none of the excluded host objects is a carrier coordinate, the five listed clauses exhaust the local NameCert surface. □

Theorem 9039.3 (ConvexFunction midpoint route)**.** *Every midpoint or finite affine read of an accepted* $\mathsf{ConvexFunction}^{\uparrow}$ *carrier routes through*

$$D, V, A, M \longmapsto R, E \longmapsto H, C, P, N.$$

Consequently Jensen and Fenchel consumers may inspect a convex-function source only after the displayed domain, rational weights, $\mathsf{VecSpace}^{\uparrow}$ *affine route,* $\mathsf{Real}^{\uparrow}$ *comparison row, componentwise transport, replay, provenance, and local name have all been read.*

Proof. Start with a consumer read of a midpoint or finite affine expression. By Definition 9039.1, the only source rows for such a read are the domain row D, vector row V, affine-weight row A, and affine-combination ledger M.

The real-valued convexity statement is downstream of those rows. The codomain row R supplies the displayed real reads, and E supplies the comparison row tying the value at the affine point to the affine combination of output reads.

Finally H, C, P, N transport, replay, source, and name the same route. Since the packet contains no separate host-function or quotient-equality coordinate, Jensen and Fenchel consumers cannot bypass this displayed path. □

Falsifiable prediction: Every $\mathsf{ConvexFunction}^{\uparrow}$ consumer reads a ConvexSet domain row, VecSpace affine route, Rat weight row, Real codomain and comparison row, midpoint or finite affine-combination ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the convex-function source is public.

Independence witness: convexset, vecspace, rat, real: $\mathsf{ConvexFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a convex-domain row, vector-source row, rational affine-weight row, real codomain and comparison row, epigraph ledger, transport, replay, provenance, and local NameCert data as one finite function packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConvexFunction}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{ConvexFunction}^{\uparrow}$ packet is generated from a displayed ConvexSet domain row, VecSpace source row, Real codomain and readback row, Rat affine-weight route, midpoint or finite affine-combination ledger, epigraph and comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvexFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9039.1, Definition 9039.2, and Definition 9039.3 seed the finite convex-function route over ConvexSet, VecSpace, Rat, Real, affine-combination, epigraph, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvexFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex functions, quotient equality of functions or real names, selected extrema, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-combination accountability, convexity comparison exactness, transport and replay stability, ledger nonescape, provenance, and local naming for BEDC.Derived.ConvexFunctionUp.

Chapter 9040

A Concrete Naming Certificate for $\mathsf{FaulhaberFormula}^{\uparrow}$

Following Faulhaber and the finite-difference treatment of sums of powers, the $\mathsf{FaulhaberFormula}^{\uparrow}$ packet records the finite BEDC route by which a unary exponent row, a polynomial coefficient ledger, and a finite-difference telescoping row name the closed-form power-sum formula. It sits over the natural-number carrier of chapter 28, the additive and multiplicative arithmetic rows of chapter 29 and chapter 30, the polynomial coefficient surface of chapter 50, and the finite-difference row of chapter 5135. The packet is a finite summation consumer: it does not import an ambient analytic series, a host polynomial identity library, quotient real equality, countable choice, or a selected infinite limit.

Definition 9040.1 (Faulhaber formula carrier)**.** *A* $\mathsf{FaulhaberFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (N, A, M, P, D, S, H, C, Q, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *exponent and summation-length row.*
2. *A is the* $\mathsf{Add}^{\uparrow}$ *accumulation row for finite partial sums.*
3. *M is the* $\mathsf{Mul}^{\uparrow}$ *multiplication row used to form powers and polynomial products.*
4. *P is the* $\mathsf{Poly}^{\uparrow}$ *coefficient ledger whose displayed polynomial is the closed power-sum candidate.*
5. *D is the* $\mathsf{FiniteDifference}^{\uparrow}$ *row asserting that the forward difference of the displayed polynomial is the inspected power row.*
6. *S is the finite summation row comparing the accumulated power values with the polynomial endpoint difference.*
7. *H records componentwise* hsame *transport for exponent, arithmetic, polynomial, finite-difference, summation, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a unary bound through N, A, M, P, D, S, H.*
9. *Q is the* Pkg *provenance over* $\mathsf{FaulhaberFormula}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Mul}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{FiniteDifference}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{FaulhaberFormula}^{\uparrow}}$ *row.*

The classifier accepts a Faulhaber read only through these rows. It has no coordinate for an analytic generating function, Bernoulli-number completion outside the displayed coefficient ledger, quotient equality of polynomials, ambient ordered-field automation, or an infinite summation principle.

Theorem 9040.2 (Faulhaber formula NameCert obligations)**.** *For every accepted* $\mathsf{FaulhaberFormula}^{\uparrow}$ *carrier* $F = (N, A, M, P, D, S, H, C, Q, L)$*, the local* $\mathsf{NameCert}_{\mathsf{FaulhaberFormula}^{\uparrow}}$ *surface consists exactly of carrier admission for the natural-number exponent and bound, additive accumulation, multiplicative power formation, polynomial coefficient ledger, finite-difference certificate, finite summation comparison, componentwise transport, displayed replay, provenance, and local naming rows. No accepted consumer may read the closed power-sum formula before those rows are displayed.*

Proof. Project the carrier of Definition 9040.1. The rows $N, A, M, P, D, S, H, C, Q, L$ are the only displayed data available to a local certificate.

The arithmetic rows A and M expose finite addition and multiplication before any power row is summed. The polynomial row P supplies the closed candidate as a finite coefficient ledger, not as an ambient host polynomial identity.

The finite-difference row D is then the exactness row: it relates the displayed polynomial ledger to the inspected power row. The summation row S consumes that exactness by finite telescoping over the unary bound named by N.

Finally H, C, Q, L transport, replay, source, and name the same finite route. Since no excluded object is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Theorem 9040.3 (Faulhaber finite-difference route)**.** *Every accepted* $\mathsf{FaulhaberFormula}^{\uparrow}$ *consumer factors the closed power-sum read through the finite route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{Add}^{\uparrow}, \mathsf{Mul}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{FiniteDifference}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The route reads the polynomial coefficient ledger and its displayed finite difference before it exports the summation comparison, and it cannot be replaced by an analytic series, a host Bernoulli-coefficient table, quotient equality of polynomials, countable choice, or an infinite limit.

Proof. Use the NameCert obligations of Definition 9040.2. They force every accepted read to expose the natural-number, additive, multiplicative, polynomial, finite-difference, and summation rows before provenance and local naming are consumed.

The finite-difference row is the bridge from the coefficient ledger to the power row. Once the displayed forward difference is read, the finite summation row performs only the bounded telescoping comparison over the unary length from N.

The non-escape clause follows from the carrier definition. None of analytic series data, host Bernoulli tables outside P, quotient polynomial equality, countable choice, or infinite limits is present in F, so none can serve as an intermediate factor in an accepted Faulhaber read. □

Falsifiable prediction: A $\mathsf{FaulhaberFormula}^{\uparrow}$ consumer that exports a power-sum formula while omitting the polynomial coefficient ledger P or the finite-difference row D is not reading an accepted packet of this chapter.

Independence witness: nat, add, mul, polynomial, finite-difference: $\mathsf{FaulhaberFormula}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries arithmetic power formation, a polynomial coefficient ledger, a finite-difference exactness row, a summation comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FaulhaberFormula}^{\uparrow}}$**.**

Constructive story: A FaulhaberFormula$^{\uparrow}$ packet is generated from NatUp exponent and bound rows, AddUp accumulation, MulUp power formation, a PolynomialUp coefficient ledger, a FiniteDifferenceUp exactness row, a finite summation comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FaulhaberFormula}^{\uparrow}}$, seed)

Scope closed: Definition 9040.1, Definition 9040.2, and Definition 9040.3 seed the finite power-sum certificate route over arithmetic, polynomial, finite-difference, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FaulhaberFormula}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, analytic generating functions, Bernoulli-number completion outside the displayed coefficient ledger, quotient polynomial equality, countable choice, or infinite summation.

Upgrade path: Obligation closure requires checked carrier admission, arithmetic power formation, polynomial coefficient readiness, finite-difference exactness, finite summation telescoping, transport, replay, provenance, and local naming for BEDC.Derived.FaulhaberFormulaUp.

Chapter 9041

A Concrete Naming Certificate for $\mathsf{SeparatedCauchySequence}^{\uparrow}$

Following Bishop–Bridges style separated metric and Cauchy-sequence grounding, the $\mathsf{SeparatedCauchySequence}^{\uparrow}$ packet records the finite route by which a displayed regular rational sequence is read as a separated completion-facing name before a Real consumer may inspect the terminal seal. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the cofinal restriction route of chapter 1592, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names separated Cauchy-sequence data as finite NameCert rows, not as quotient sequence equality, a selected limit, host real equality, countable choice, or ambient Real completeness.

Definition 9041.1 (Separated Cauchy sequence carrier)**.** *A* $\mathsf{SeparatedCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, K, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the observed prefix and tail windows of the sequence.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the separated tail comparisons used by the Cauchy test.*
4. *K is the* $\mathsf{CauchySubnet}^{\uparrow}$ *cofinal-tail route used when the consumer restricts the read to a displayed subnet of tail requests.*
5. *Q is the separated-comparison row, recording that zero-distance comparison is read only through displayed tail and tolerance rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after W, R, D, K, Q remain visible together.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, cofinal route, separated-comparison, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through W, R, D, K, Q, E in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{SeparatedCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient sequence equality, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.

Theorem 9041.2 (Separated Cauchy sequence NameCert obligations)**.** *For every accepted* SeparatedCauchySequen *carrier $S = (W, R, D, K, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}$ *surface consists exactly of finite-window admission, regular rational readback, dyadic tolerance exposure, cofinal-tail route exposure, separated-comparison exposure, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Every obligation is anchored in the displayed BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No obligation row may replace the finite packet by quotient sequence equality, a selected limit, countable choice, host real equality, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 9041.1. The analytic data are W, R, D, K, Q, E, and the structural support is H, C, P, N.

Read the route in order. The StreamName window W exposes the inspected finite tail data, the RegSeqRat row R records the regular rational readback, and the DyadicRatCore row D supplies the tolerance ledger for separated tail comparison.

The cofinal-tail row K is the only subnet restriction route, and the separated-comparison row Q is downstream of W, R, D, K. The terminal Real seal E is downstream of all five displayed data rows.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since no carrier coordinate stores quotient sequence equality, selected limits, countable choice, host real equality, trichotomy, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 9041.3 (Separated Cauchy sequence real-completion handoff)**.** *Every Real or completion-facing consumer of an accepted* $\mathsf{SeparatedCauchySequence}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchySubnet}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the separated-comparison, BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}$ *rows. The consumer receives a displayed separated Cauchy-sequence handoff and not quotient sequence equality, a selected limit, host real equality, countable choice, trichotomy, or ambient Real completeness.*

Proof. Start with an accepted carrier $S = (W, R, D, K, Q, E, H, C, P, N)$. The terminal seal E is not a free Real value; it is downstream of the window, readback, tolerance, cofinal route, and separated-comparison rows.

Use Definition 9041.2 to exhaust the local surface. A consumer must expose $W, R, D, K, Q, E, H, C, P, N$ before the packet can be consumed by a Real or completion-facing route.

The cited sibling chapters supply the allowed components: StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, CauchySubnet cofinal-tail routing, and the terminal Real seal. Since the carrier refuses the excluded coordinates, the handoff is a finite NameCert route rather than an ambient completeness theorem or selected host limit. □

Falsifiable prediction: An accepted $\mathsf{SeparatedCauchySequence}^{\uparrow}$ read is refuted by any Real-facing consumer that reaches the terminal seal without first displaying the StreamName finite

window, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchySubnet cofinal route, and separated-comparison row.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchysubnet: $\mathsf{SeparatedCauchySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite windows, regular rational readback, dyadic separated-tail tolerance, cofinal-tail routing, terminal Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCauchySequence}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readback, a DyadicRatCore separated-tail tolerance ledger, a CauchySubnet cofinal route, a separated-comparison row, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9041.1, Definition 9041.2, and Definition 9041.3 seed the separated Cauchy-sequence route through StreamName, RegSeqRat, DyadicRatCore, CauchySubnet, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-tail tolerance exposure, cofinal-tail routing, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCauchySequenceUp.

Chapter 9042

A Concrete Naming Certificate for $\mathsf{RealPrefixDiagonal}^{\uparrow}$

The $\mathsf{RealPrefixDiagonal}^{\uparrow}$ packet realizes the finite prefix-window diagonal route of chapter 9238 as NameCert data. It consumes the finite observation discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the diagonal-limit route of chapter 6002, and the terminal seal of chapter 38. The packet is paper-side TasteGate aware: a future gate must preserve nontrivial prefix variation and field-faithful visibility for the displayed rows before any positive discovery claim can be made.

Falsifiable prediction: An accepted $\mathsf{RealPrefixDiagonal}^{\uparrow}$ read is refuted by any diagonal-limit consumer that reaches the terminal RealUp seal without displaying a finite prefix window, RegSeqRat readback, DyadicRatCore tolerance ledger, diagonal selector row, transport row, replay row, provenance row, and local NameCert row.

Independence witness: streamname, regseqrat, dyadicratcore, regular-cauchy-diagonal-limit, real: $\mathsf{RealPrefixDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the finite prefix-window diagonal selector, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming as one packet.

Definition 9042.1 (Real prefix diagonal carrier)**.** *A* $\mathsf{RealPrefixDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (B, W, R, D, G, E, H, C, K, N)$$

with the following displayed rows:

1. *B is the finite prefix-budget row naming the admitted observation depth for the diagonal read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed under that prefix budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the diagonal error budget.*
5. *G is the diagonal selector row, recording which displayed prefix coordinate is read at the current precision.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after B, W, R, D, G have been displayed together.*
7. *H records componentwise* hsame *transport for the budget, window, readback, tolerance, selector, seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a precision request through B, W, R, D, G, E.*
9. *K is the* Pkg *provenance over* $\mathsf{RealPrefixDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealPrefixDiagonal}^{\uparrow}}$ *row.*

The carrier exports only finite prefix-window diagonal data. It has no coordinate for a host-selected subsequence, quotient stream equality, ambient Real completeness, countable choice, or a bridge schema.

Theorem 9042.2 (Real prefix diagonal NameCert obligations)**.** *For every accepted* $\mathsf{RealPrefixDiagonal}^{\uparrow}$ *carrier $P = (B, W, R, D, G, E, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealPrefixDiagonal}^{\uparrow}}$ *surface consists exactly of prefix-budget admission, finite StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance exposure, diagonal selector exposure, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming.*

The paper-side TasteGate boundary is the same row list: any future gate must preserve visible variation in B, W, R, D, G, E and field-faithful access to H, C, K, N. Until such a gate exists, the chapter claims only the finite NameCert packet and not a positive discovery result.

Proof. Project the carrier of Definition 9042.1. The finite data rows are B, W, R, D, G, E, and the structural rows are H, C, K, N.

Read the diagonal route in order. The prefix budget B admits a bounded observation, the StreamName row W exposes the corresponding window, the RegSeqRat row R supplies regular rational readback, and the DyadicRatCore row D records the tolerance attached to the readback.

The selector row G chooses only among displayed prefix coordinates, and the terminal Real seal E is downstream of that selector. Transport, replay, provenance, and naming preserve this route without adding host-selected subsequences, quotient stream equality, countable choice, ambient Real completeness, or a bridge schema. □

Theorem 9042.3 (Real prefix diagonal limit route)**.** *Every Real-completeness diagonal consumer of an accepted* $\mathsf{RealPrefixDiagonal}^{\uparrow}$ *packet factors through the finite route*

$$B, W, R, D, G, E, H, C, K, N.$$

Equivalently, the consumer must expose a prefix budget, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, diagonal selector row, terminal Real seal, componentwise transport, displayed replay, package provenance, and local NameCert row before it can read the diagonal-limit handoff named by chapter 9238.

Proof. Begin with Definition 9042.1. The terminal seal E is not accessible until the budget, window, readback, tolerance, and selector rows are present in the same packet.

Use Definition 9042.2 to exhaust the local ledger. The listed rows account for every data-bearing and structural coordinate of the packet.

The vision route of chapter 9238 can therefore consume this chapter only as a finite prefix-window handoff. Because the carrier has no coordinate for host-selected subsequences, quotient stream equality, countable choice, ambient Real completeness, or a bridge schema, none of those objects can be projected from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealPrefixDiagonal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealPrefixDiagonal}^{\uparrow}$ packet is generated from a finite prefix budget, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, diagonal selector row, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealPrefixDiagonal}\uparrow}$, seed)

Scope closed: Definition 9042.1, Definition 9042.2, Definition 9042.3, the falsifiable prediction, and the independence witness seed the finite prefix-window diagonal route of the Real-completeness vision.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealPrefixDiagonal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full Real completeness, quotient stream equality, host-selected subsequences, countable choice, positive discovery status, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window exposure, diagonal selector exactness, tolerance-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealPrefixDiagonalUp.

Chapter 9043

A Concrete Naming Certificate for $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$

The $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ packet realizes the normal zero-sheet geometry of section 9413.30 as finite BEDC NameCert data. It sits after the complex-coordinate carrier of chapter 39, the continuation and zero-consumer ledgers of chapter 580 and chapter 73, the critical-strip window of chapter 8868, the zero-type factorization of chapter 8772, the finite prime-window gluing route of chapter 8765, and the RH boundary separation of chapter 8864. The packet names finite normal-sheet support evidence; it does not assert RH, zero exclusion, analytic continuation completeness, zero simplicity, or an infinite spectral theorem.

Definition 9043.1 (Zero-fibre normal sheet carrier)**.** *A* $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (C, V, W, O, M, P, R, H, K, Q, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Complex}^{\uparrow}$ *mirror-coordinate row, displaying a critical-strip point as* $\frac{1}{2} + \delta + i\tau$*.*
2. *V is the completed-zeta value row read through the existing zeta-continuation and zero-consumer ledgers.*
3. *W is the* $\mathsf{RHCriticalStripWindowCertificate}^{\uparrow}$ *row that keeps the zero read inside a displayed finite critical-strip window.*
4. *O is the* $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$ *row separating zero-fibre, trivial-zero, generated-zero, and RH-facing support reads.*
5. *M is the* $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$ *row recording the mirror companion of an off-fixed coordinate without identifying it with the fixed line.*
6. *P is the* $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$ *support row used when a normal coordinate is compared with finite prime-window defect evidence.*
7. *R is the support route row from a normal-coordinate read and completed-zero read to the finite no-extra-zero-sheet boundary.*
8. *H records componentwise* hsame *transport for coordinate, completed-zeta, critical-window, zero-factorization, mirror, prime-window, support-route, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a zero-fibre query through C, V, W, O, M, P, R, H.*

10. *Q is the* Pkg *provenance over* $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, RHCriticalStripWindowC ZetaZeroTypeFactorization$^{\uparrow}$, $\mathsf{RHMirrorOrbitTarget}^{\uparrow}$, $\mathsf{FinitePrimeWindowGluing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ZeroFibreNormalSheet}^{\uparrow}}$ *row.*

The classifier accepts a normal-sheet support read only through these rows. It has no coordinate for an RH proof, global zero exclusion, analytic continuation completeness, zero simplicity, host complex equality, an infinite prime window, or a spectral trace formula.

Theorem 9043.2 (Zero-fibre normal sheet NameCert obligations)**.** *For every accepted* $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ *carrier* $Z = (C, V, W, O, M, P, R, H, K, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ZeroFibreNormalSheet}^{\uparrow}}$ *surface consists exactly of carrier admission for the mirror-coordinate row, completed-zeta value row, finite critical-strip window, zero-type factorization, mirror-orbit row, finite prime-window support row, normal-sheet support route, componentwise transport, displayed replay, provenance, and local naming. No accepted consumer may use the packet as an RH proof or zero-exclusion theorem before these finite rows are displayed.*

Proof. Project the carrier of Definition 9043.1. The rows $C, V, W, O, M, P, R, H, K, Q, N$ are the only coordinates available to the local certificate.

The coordinate row C and value row V expose the zero-fibre read, while W keeps the read inside a finite critical-strip window. The factorization row O prevents this zero read from being confused with trivial-zero, generated-zero, or global RH surfaces.

The mirror row M records pairing under the functional-equation symmetry without collapsing the normal coordinate. The finite prime-window row P supplies the support evidence consumed by R when the packet is read as a no-extra-zero-sheet boundary.

Finally H, K, Q, N transport, replay, source, and name the same finite route. Since the carrier contains no RH proof, global zero-exclusion coordinate, analytic-completion coordinate, zero-simplicity coordinate, host equality coordinate, infinite prime window, or spectral trace formula, none of those objects is exported by the local NameCert surface. □

Theorem 9043.3 (Zero-fibre normal sheet RH support route)**.** *Every RH-facing support read from an accepted* $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{Complex}^{\uparrow}, \mathsf{ZetaCont}^{\uparrow}, \mathsf{ZetaZeros}^{\uparrow} \longmapsto \mathsf{RHCriticalStripWindowCertificate}^{\uparrow} \longmapsto \mathsf{ZetaZeroTypeFactorization}^{\uparrow}, \mathsf{RHMirrorO}$$

The route names a finite support certificate for the normal zero-sheet boundary; it does not decide RH, exclude all zeros off the fixed line, prove analytic continuation completeness, or replace finite prime-window evidence by an infinite spectral object.

Proof. Use the obligation surface of Definition 9043.2. It forces every accepted support read to expose the coordinate, completed-zeta, critical-window, zero-factorization, mirror, finite prime-window, and support-route rows before provenance and local naming are consumed.

The support row R is downstream of both the mirror-orbit row and the finite prime-window row. Thus the packet may record normal-sheet support evidence only after the zero-fibre read has been separated from global RH claims and after the finite prime-window comparison has been displayed.

The non-escape statement is carrier inspection. The tuple contains no coordinate for an RH proof, global zero exclusion, analytic continuation completeness, or infinite spectral trace, so the accepted route remains a finite support certificate. □

Falsifiable prediction: A $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ consumer that claims no-extra-zero-sheet support while omitting the critical-strip window W, zero-factorization row O, mirror-orbit row M, or finite prime-window support row P is not reading an accepted packet of this chapter.

Independence witness: zeta-continuation-witness, zeta-zeros, rh-critical-strip-window, zeta-zero-type-factorization, finite-prime-window-gluing: $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries mirror coordinates, completed-zeta zero reads, finite critical-window support, zero-factorization, mirror-orbit, finite prime-window support, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide explicit $\mathsf{Nontrivial\ ZeroFibreNormalSheet}^{\uparrow}$ and $\mathsf{FieldFaithful\ ZeroFibreNormalSheet}^{\uparrow}$ instances. Those instances must keep $C, V, W, O, M, P, R, H, K, Q, N$ as separate displayed fields before any checked support-route theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroFibreNormalSheet}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeroFibreNormalSheet}^{\uparrow}$ packet is generated from ComplexUp mirror coordinates, completed-zeta value rows, a finite critical-strip window, zero-type factorization, mirror-orbit rows, finite prime-window support, a normal-sheet support route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeroFibreNormalSheet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9043.1, Definition 9043.2, and Definition 9043.3 seed the finite normal-sheet support route over zeta, critical-window, mirror, prime-window, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeroFibreNormalSheet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, RH, global zero exclusion, analytic continuation completeness, zero simplicity, host complex equality, infinite prime windows, or a spectral trace formula.

Upgrade path: Obligation closure requires checked carrier admission, completed-zero visibility, critical-window support, zero-factorization separation, mirror-orbit preservation, finite prime-window support, Nontrivial ZeroFibreNormalSheetUp, and FieldFaithful ZeroFibreNormalSheetUp for BEDC.Derived.ZeroFibreNormalSheetUp.

Chapter 9044

A Concrete Naming Certificate for $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$

Following Ambrosio–Gigli–Savare and Burago–Burago–Ivanov metric-curve practice, the $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$ packet records absolute continuity of a displayed metric curve by finite interval windows, Real-valued distance estimates, and a partition-sum modulus. It sits between the L10 metric-completion route of $\mathsf{Met}^{\uparrow}$ in chapter 4230, the path graph surface of $\mathsf{ContinuousMap}^{\uparrow}$ in chapter 130, the Real tolerance ledger of $\mathsf{Real}^{\uparrow}$ in chapter 38, the partition-control discipline of $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ in chapter 3361, the finite length handoff of $\mathsf{RectifiableCurve}^{\uparrow}$ in chapter 3443, and the scale row of $\mathsf{ArcLengthParameter}^{\uparrow}$ in chapter 6944. The certificate names the curve-level metric object only through displayed rows; it is not a host path space, a selected representative curve, a measure-completed derivative theorem, or quotient equality of curves.

Definition 9044.1 (Absolute continuous curve carrier)**.** *An* $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, I, W, G, D, E, U, R, L, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, containing the accepted metric carrier and the distance read used for curve samples.*
2. *I is the interval row, naming the ordered source interval and its finite endpoint windows.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *parameter-window row recording exactly the sampled subintervals and endpoint requests consumed by the packet.*
4. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *curve graph row from the displayed interval windows into the metric carrier.*
5. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance-estimate row for adjacent sampled curve values, read only through the metric distance in M and the graph row G.*
6. *E is the partition-sum absolute-continuity modulus row: for each displayed Real error budget it records a finite interval-length budget and the corresponding sum of distance estimates over disjoint sampled subintervals.*
7. *U is the* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *handoff row, exposing the same partition-budget discipline as a function-level finite ledger when a scalar Real-valued consumer asks for it.*
8. *R is the* $\mathsf{RectifiableCurve}^{\uparrow}$ *handoff row, available only after M, I, W, G, D, E have been displayed.*

9. *L is the* $\mathsf{ArcLengthParameter}^{\uparrow}$ *compatibility row, pairing accepted length-parameter requests with the same distance-estimate and partition-sum rows.*
10. *S records componentwise* hsame *transport for metric, interval, window, graph, distance-estimate, modulus, rectifiable, arc-length, replay, provenance, and local naming rows.*
11. *H records displayed* Cont *replay from a curve absolute-continuity request through $M, I, W, G, D, E, U, R, L$.*
12. *C is the* Pkg *provenance row over* $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$, $\mathsf{RectifiableCurve}^{\uparrow}$, $\mathsf{ArcLengthParameter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *P is the downstream handoff ledger, recording which of the function-level, rectifiable-curve, and arc-length consumers has read the displayed rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{AbsoluteContinuousCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, I, W, G, D, E, U, R, L, S, H, C, P, N$. It has no coordinate for host function equality, selected curve representatives, quotient paths, measure-completed derivatives, an ambient path-space completion, countable choice, or bridge evidence.

Theorem 9044.2 (Absolute continuous curve NameCert obligations)**.** *For the carrier of Definition 9044.1, the local* $\mathsf{NameCert}_{\mathsf{AbsoluteContinuousCurve}^{\uparrow}}$ *surface consists of exactly five obligation rows:*

1. *carrier admission for the finite packet $A = (M, I, W, G, D, E, U, R, L, S, H, C, P, N)$;*
2. *metric-curve exposure through M, I, W, G, D, so every distance estimate is read from displayed parameter windows and the accepted metric distance;*
3. *partition-sum absolute-continuity control through E and the function-level handoff U;*
4. *rectifiable-curve and arc-length compatibility through R, L, P after the same distance-estimate rows have been displayed;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and exclusion of host path spaces, selected representatives, quotient curves, measure-completed derivatives, countable choice, and bridge evidence.*

Proof. Project the carrier in Definition 9044.1. The rows $M, I, W, G, D, E, U, R, L$ are the whole metric and analytic surface before transport, replay, provenance, handoff bookkeeping, and naming are read.

The metric-curve rows are M, I, W, G, D. They expose the accepted metric carrier, the interval and StreamName parameter windows, the continuous curve graph, and the Real-valued distance estimates for adjacent sampled values.

The absolute-continuity row is E, with U giving the function-level handoff. A consumer that asks for partition-sum control must read the same finite interval-length budget and the same finite sum of distance estimates before it can use the scalar ledger.

The curve-length rows are R, L, P. Rectifiable-curve and arc-length consumers receive only the already displayed distance-estimate packet, together with the handoff ledger recording which consumer route is being used.

Finally S, H, C, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a host path space, a selected representative curve, quotient equality of curves, a measure-completed derivative, countable choice, or bridge evidence, the five listed obligations exhaust the local NameCert surface. □

Theorem 9044.3 (Absolute continuous curve metric-length handoff)**.** *Every metric-completion, rectifiable-curve, or arc-length consumer of an accepted* $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$ *packet reads the displayed route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{AbsolutelyContinuousFunction}^{\uparrow},\ \mathsf{RectifiableCurve}^{\uparrow},\ \mathsf{ArcLengthParameter}^{\uparrow}$$

through the finite rows $M, I, W, G, D, E, U, R, L, S, H, C, P, N$*. The handoff exports curve-level absolute continuity as a finite metric partition certificate, not as a host path, quotient curve, selected representative, completed path space, measure derivative, or bridge datum.*

Proof. Start with the metric and graph rows. The accepted metric carrier M and the continuous graph row G can be read only at the displayed interval and StreamName windows I, W.

Next read the numerical control. The row D gives the Real-valued distance estimates for adjacent sampled curve values, while E records the partition-sum modulus that bounds those estimates over finite disjoint subintervals.

Now pass to the sibling consumers. The function-level row U reuses the same partition-control surface, the rectifiable row R reuses the same finite distance-sum ledger, and the arc-length row L reuses the same length-compatible parameter requests.

The structural rows S, H, C, P, N preserve this route by transport, replay, provenance, consumer bookkeeping, and local naming. No step introduces a host path object, quotient equality of curves, selected representative, completed path space, measure-completed derivative, countable-choice schedule, or bridge evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteContinuousCurve}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbsoluteContinuousCurve}^{\uparrow}$ packet is generated from a MetricSpace source row, interval and StreamName parameter windows, a ContinuousMap curve graph, Real-valued distance estimates, a partition-sum absolute-continuity modulus, AbsolutelyContinuousFunction handoff, RectifiableCurve handoff, ArcLengthParameter compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, downstream handoff bookkeeping, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuousCurve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9044.1, Definition 9044.2, and Definition 9044.3 seed the finite metric-curve absolute-continuity route over $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$, $\mathsf{RectifiableCurve}^{\uparrow}$, and $\mathsf{ArcLengthParameter}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuousCurve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, selected curve representatives, quotient curve equality, measure-completed derivatives, completed path-space structure, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-curve exposure, Real-valued distance-estimate exactness, partition-sum absolute-continuity control, rectifiable-curve handoff, arc-length compatibility, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteContinuousCurveUp.

Chapter 9045

A Concrete Naming Certificate for $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$ packet records the finite L10 route by which one displayed dyadic rational source is embedded as a constant regular rational sequence before a $\mathsf{Real}^{\uparrow}$ consumer may inspect the terminal seal. It sits beside the dyadic carrier of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite embedding handoff; it does not assert quotient equality of real names, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9045.1 (DyadicEmbeddingRegularSequence carrier)**.** *A* $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, naming the dyadic rational being embedded and the dyadic arithmetic cells available to later tolerance reads;*
2. *W is the constant* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing finite observations whose entries all read the same source dyadic row D;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the regular rational Cauchy schedule for the constant stream and its dyadic tolerance comparison;*
4. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, W, R have been displayed;*
5. *H records componentwise* hsame *transport for dyadic source, constant window, regularity, Real seal, replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay from a Real-facing request through dyadic admission, constant-window inspection, regular rational readback, Real seal consumption, provenance, and local naming;*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, W, R, L, H, C, P, N. No coordinate stores a host real, quotient representative, selected limit, countable-choice schedule, trichotomy branch, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 9045.2 (DyadicEmbeddingRegularSequence obligation surface)**.** *For every accepted* $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$ *carrier*

$$E = (D, W, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}}$ *surface consists exactly of dyadic source admission, constant* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback for the constant stream, terminal* $\mathsf{Real}^{\uparrow}$ *seal gating, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A Real-facing consumer may inspect the embedded dyadic name only through these rows.*

Proof. Project the carrier $E = (D, W, R, L, H, C, P, N)$. The dyadic row D is exposed before any stream or regularity read, so the source rational is a visible BEDC row rather than a hidden scalar.

The window row W repeats the same source dyadic row at every displayed observation. The regularity row R reads that constant window together with the dyadic tolerance budget, so the Cauchy schedule is finite row data.

Only after D, W, R have been displayed can the Real seal L be consumed. The support rows H, C, P, N transport, replay, source, and name the same route. Since the packet has no coordinate for quotient equality, selected limits, countable choice, trichotomy, or ambient Real completeness, those data are not exported by the obligation surface. □

Theorem 9045.3 (DyadicEmbeddingRegularSequence Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$ *carrier first exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row D, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row R. Therefore the Real seal is a finite embedding route from a dyadic rational to a regular sequence name, not a selected host real or an ambient completeness theorem.*

Proof. Read the source row D and the constant window row W. They make the dyadic rational and its repeated finite observations visible before any Real-facing consumer acts.

Use the regularity row R. It supplies the regular rational Cauchy readback and the dyadic tolerance comparison for the constant stream.

The terminal row L is then reached through the displayed embedding route. Componentwise transport, replay, provenance, and local naming preserve that same route and add no host real, quotient representative, selected limit, choice schedule, trichotomy branch, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}$ packet is generated from a DyadicRatCoreUp source row, a constant StreamNameUp window row, a RegSeqRatUp regularity row, a terminal RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}}$, seed)

Scope closed: Definition 9045.1, Definition 9045.2, and Definition 9045.3 seed the finite dyadic-to-regular-sequence embedding surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicEmbeddingRegularSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of real names, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic source admission, constant stream-window exposure, regular rational readback, Real seal gating, transport, replay, provenance, and local naming for BEDC.Derived.DyadicEmbeddingRegularSequenceUp

Chapter 9046

A Concrete Naming Certificate for $\mathsf{FixedHalfZeroWitness}^{\uparrow}$

The $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ packet names only the typed witness-function package of Definition 9333.1. It is not a proof of RH, not a zero-location theorem, and not an analytic exclusion of off-line zeros. The packet sits beside the critical-line witness surface of chapter 548, the basic zeta source of chapter 71, the zeta-zero carrier of chapter 73, and the zero-type factorization packet of chapter 8772.

Definition 9046.1 (FixedHalfZeroWitness carrier)**.** *A* $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Z, S, W, T, B, H, C, P, N)$$

with the following displayed rows:

1. *Z is the non-trivial zeta-zero source row, read through the zero and critical-strip typing data of the RH proof-obligation package;*
2. *S is the source-classifier row, recording the projection back to the original zero source before any fixed-half witness is cited;*
3. *W is the fixed-half witness-function row, displaying the typed function from the non-trivial-zero package to the fixed-half-zero package;*
4. *T is the transport row witnessing that the projected source of W is classifier-equal to the source row named by S;*
5. *B is the refusal boundary row, recording that the packet exports no proof of* RH*, no analytic zero exclusion, no global zero classifier, and no host truth value;*
6. *H records componentwise* hsame *transport for source, classifier, witness-function, source-transport, refusal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a witness-function request through zero-source admission, classifier projection, fixed-half witness readback, refusal-boundary inspection, provenance, and local naming;*
8. *P is the* Pkg *provenance over* $\mathsf{FixedHalfZeroWitness}^{\uparrow}$*,* $\mathsf{CriticalLineWitness}^{\uparrow}$*,* $\mathsf{ZetaBasic}^{\uparrow}$*,* $\mathsf{ZetaZeros}^{\uparrow}$*,* $\mathsf{ZetaZeroTypeFactorization}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FixedHalfZeroWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, S, W, T, B, H, C, P, N$. It has no coordinate for an RH proof, a zero-free strip theorem, a completed analytic continuation argument, an ambient truth predicate, Classical choice, quotient equality, or a standard bridge.

Theorem 9046.2 (FixedHalfZeroWitness obligation surface)**.** *For the carrier of Definition 9046.1, the local* $\mathsf{NameCert}_{\mathsf{FixedHalfZeroWitness}^{\uparrow}}$ *surface consists exactly of non-trivial-zero source admission, source-classifier projection, fixed-half witness-function readback, source-transport evidence, refusal-boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the finite packet $F = (Z, S, W, T, B, H, C, P, N)$. The source row Z and source-classifier row S are visible before the witness-function row is read, so the package starts from typed zero data rather than an untyped analytic assertion.

The witness-function row W supplies only the displayed function package named by Definition 9333.1. The transport row T records the projection back to the same source classifier.

The refusal row B is part of the same carrier. It prevents the read from being cited as an RH proof, an off-line zero exclusion, a global zero classifier, or a host truth value. The support rows H, C, P, N preserve, replay, source, and name that finite package, and no further obligation is stored outside the displayed rows. □

Theorem 9046.3 (FixedHalfZeroWitness RH witness route)**.** *Every accepted* $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ *read routes the typed witness function of Definition 9333.1 through $Z, S, W, T, B, H, C, P, N$. The route may be used as a concrete NameCert landing for the RH witness-function target, but it does not prove* RH*, does not supply analytic zero exclusion, and does not turn fixed-half witness data into a bridge-checked theorem.*

Proof. Read the source row Z and classifier projection row S. They identify the typed non-trivial-zero input and its source before a fixed-half target can be named.

Next read W and T. The witness-function row displays the fixed-half output package, and the transport row records that its projection returns to the source classifier.

Finally inspect B, H, C, P, N. The refusal boundary keeps the route inside the typed package and the structural rows transport, replay, source, and name the same data. Thus the route is a witness-function NameCert surface, not an RH proof or a bridge-checked analytic theorem. □

Falsifiable prediction: If an accepted FixedHalfZeroWitnessUp packet exports a fixed-half witness without displaying the non-trivial-zero source row, source-classifier projection, witness-function row, source-transport row, and refusal boundary, then it is refuted as a NameCert carrier for the RH witness-function package.

Independence witness: criticallinewitness, zeta-basic, zeta-zeros, zeta-zero-type-factorization: none is bijective to $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ because those siblings respectively bind critical-line witness rows, basic zeta source rows, zero carrier rows, or zero-type factorization rows, while this packet binds the typed witness-function route with source projection, refusal boundary, transport, replay, provenance, and local NameCert rows.

The Lean follow-up file `lean4/BEDC/Derived/FixedHalfZeroWitnessUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{FixedHalfZeroWitness}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{FixedHalfZeroWitness}^{\uparrow}$ instances. The nontriviality witness must distinguish an accepted witness-function packet from a packet missing the source-transport or refusal-boundary row, and the field-faithfulness witness must preserve $Z, S, W, T, B, H, C, P, N$ as separate rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FixedHalfZeroWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ packet is generated from a non-trivial zero source row, source-classifier row, typed fixed-half witness-function row, source-transport row, refusal-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FixedHalfZeroWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9046.1, Definition 9046.2, and Definition 9046.3 seed the typed fixed-half witness-function package without proving RH.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FixedHalfZeroWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove RH, exclude off-line zeros, provide analytic continuation evidence, supply a global zero classifier, close Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked zero-source admission, source projection, witness-function read-back, source transport, refusal-boundary exposure, provenance, local naming, and the required Nontrivial and FieldFaithful instances for BEDC.Derived.FixedHalfZeroWitnessUp.

Chapter 9047

Bishop Separated Cauchy Completion Idempotence NameCert Construction

Following Bishop and Bridges, the $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ packet records the finite certificate by which a separated regular Cauchy completion read remains on the same displayed route when completion is requested again. It sits after the separated completion idempotence row of chapter 6970, the regular Cauchy Hausdorff reflection of chapter 4372, and the Bishop separated real completion packet of chapter 9026. The packet names idempotence of the displayed separated regular Cauchy completion route; it is not quotient equality of completed objects, not a selected completed representative, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9047.1 (Bishop separated Cauchy completion idempotence carrier)**.** *A* BishopSeparatedCauchyCompleti *carrier is a finite* BHist *packet*

$$J = (C, S, R, Q, B, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Completion}^{\uparrow}$ *source row for the repeated completion request.*
2. *S is the* $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ *row, exposing the separated classifier before any repeated readback.*
3. *R is the* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *row, recording the Hausdorff-separated regular Cauchy comparison surface.*
4. *Q is the* $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ *row, giving the Bishop separated real completion route consumed by Real-facing readers.*
5. *B is the regular Cauchy readback row, carrying the displayed finite windows and regular rational data for the completion request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after C, S, R, Q, B have been displayed.*
7. *H records componentwise* hsame *transport for the completion, separated-idempotence, Hausdorff-reflection, Bishop separated completion, readback, seal, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from a second separated regular Cauchy completion request through the same rows.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, SeparatedCompletion RegularCauchyHausdorffReflection$^{\uparrow}$, $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}}$ *row.*

The classifier for accepted packets compares only $C, S, R, Q, B, E, H, T, P, N$. It has no coordinate for quotient equality of completions, selected representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9047.2 (Bishop separated Cauchy completion idempotence NameCert obligations)**.** *For every accepted* $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ *carrier $J = (C, S, R, Q, B, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}}$ *surface consists exactly of carrier admission for the repeated completion source, separated completion idempotence row, regular Cauchy Hausdorff reflection row, Bishop separated real completion row, regular Cauchy readback, terminal Real seal, componentwise transport, displayed replay, package provenance, and local naming. A repeated separated regular Cauchy completion read must expose C, S, R, Q, B before any* $\mathsf{Real}^{\uparrow}$*-facing consumer may inspect E.*

Proof. Project the carrier of Definition 9047.1. The rows $C, S, R, Q, B, E, H, T, P, N$ are the complete accepted inventory.

The separated idempotence row S is read before the Hausdorff-reflection row R and the Bishop separated real completion row Q. Thus the second completion request returns to the displayed separated classifier and regular Cauchy comparison surface before any Real-facing boundary is reached.

The regular Cauchy readback B and terminal seal E are downstream of those rows. Transport, replay, provenance, and local naming preserve the same finite route and add no coordinate for quotient equality, selected representatives, countable choice, host equality, or ambient completeness. □

Theorem 9047.3 (Bishop separated Cauchy completion idempotent seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ *packet factors through*

$$\mathsf{Completion}^{\uparrow} \longmapsto \mathsf{SeparatedCompletionIdempotence}^{\uparrow} \longmapsto \mathsf{RegularCauchyHausdorffReflection}^{\uparrow} \longmapsto \mathsf{BishopSeparatedRea}$$

Consequently, applying separated regular Cauchy completion again returns to the same finite displayed route before the terminal seal is visible. The route is an idempotence certificate for repeated separated regular Cauchy completion reads, not quotient equality, not a representative selector, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the NameCert obligations of Definition 9047.2. They force every seal-facing read through C, S, R, Q, B before E can be consumed.

Use the sibling separated idempotence surface of chapter 6970. It supplies the repeated-completion boundary as finite row data, so the second request is not interpreted as equality of completion objects.

Now read the regular Cauchy Hausdorff reflection and Bishop separated real completion siblings, chapter 4372 and chapter 9026. These keep the Hausdorff-separated comparison and Bishop separated completion route visible before the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use H, T, P, N only as transport, replay, provenance, and local naming for the same displayed rows. Since the carrier has no selected representative, quotient equality, countable-choice

schedule, host equality, or ambient completeness coordinate, none of those endpoints is exported by the seal route. □

Falsifiable prediction: An accepted $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ read is refuted if a repeated separated regular Cauchy completion request reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first exposing the $\mathsf{Completion}^{\uparrow}$ source row, $\mathsf{SeparatedCompletionIdempotence}^{\uparrow}$ row, RegularCauchyHausdorffReflec row, $\mathsf{BishopSeparatedRealCompletion}^{\uparrow}$ row, regular Cauchy readback row, transport, replay, provenance, and local naming rows.

Independence witness: separated-completion-idempotence, regular-cauchy-hausdorff-reflection, bishop-separated-real-completion, regseqrat, real: $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries the repeated completion source, separated idempotence boundary, regular Cauchy Hausdorff reflection, Bishop separated real completion route, regular readback, terminal seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9047.1, Definition 9047.2, and Definition 9047.3 seed the repeated separated regular Cauchy completion route through separated idempotence, Hausdorff reflection, Bishop separated completion, regular readback, and terminal Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of completion objects, selected representatives, countable choice, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: To reach formal target status, add the Lean declaration for BEDC.Derived.BishopSeparatedCauchyCompletionIdempotence and then prove the carrier, obligation, seal-route, transport, replay, provenance, and local naming clauses.

Constructive story: A $\mathsf{BishopSeparatedCauchyCompletionIdempotence}^{\uparrow}$ packet is generated from a Completion source row, separated completion idempotence row, regular Cauchy Hausdorff reflection row, Bishop separated real completion row, regular Cauchy readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9048

A Concrete Naming Certificate for $\mathsf{BurnsideLemma}^{\uparrow}$

Following Burnside's finite group-action counting theorem, the $\mathsf{BurnsideLemma}^{\uparrow}$ packet records the orbit-count formula as finite BEDC NameCert data. It sits over the finite group source of chapter 41 and consumes the stabilizer and fixed-orbit accounting route of chapter 6559. The certificate names fixed-point tables and orbit counts as displayed finite ledgers, not as quotient orbit sets, selected representatives, or an ambient enumeration theorem.

Definition 9048.1 (Burnside lemma carrier)**.** *A* $\mathsf{BurnsideLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, X, A, L, F, O, K, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Group}^{\uparrow}$ *action source row, exposing the group operation, unit, inverse, and finite multiplication table used by the action.*
2. *X is the acted finite set/list carrier row, exposing a duplicate-controlled finite list of acted points together with its local classifier.*
3. *A is the finite action row, displaying the unit-action and multiplication-action replay laws over G and X.*
4. *L is the finite group-list ledger, listing the group elements whose fixed-point rows are inspected.*
5. *F is the fixed-point table, assigning to each displayed element of L the finite sublist of points of X fixed by that element.*
6. *O is the orbit-count row, recording the displayed count of action orbits as the averaged fixed-point ledger named by F and L.*
7. *K is the Nat/List count ledger, exposing finite lengths, duplicate control, summation, divisibility, and the final natural-number orbit count.*
8. *H records componentwise* hsame *transport for the group, acted list, action, group-list ledger, fixed-point table, orbit-count row, Nat/List count ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from* $\mathsf{Group}^{\uparrow}$ *and X through A, L, F, O, K, H, P, N into* $\mathsf{BurnsideLemma}^{\uparrow}$.

10. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, *finite action rows, fixed-point rows, Nat/List ledgers,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BurnsideLemma}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. It has no coordinate for quotient orbit sets, selected orbit representatives, external cardinal arithmetic, an infinite group action, cycle-index polynomials, Polya enumeration, or host equality of quotient types.

Theorem 9048.2 (Burnside lemma NameCert obligations)**.** *For an accepted* $\mathsf{BurnsideLemma}^{\uparrow}$ *carrier*

$$T = (G, X, A, L, F, O, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BurnsideLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite* $\mathsf{Group}^{\uparrow}$ *source, finite set/list carrier, action row, group-list ledger, fixed-point table, orbit-count row, Nat/List count ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Proof. Project the accepted packet onto its eleven displayed rows. The group source, acted finite list, action law, group-list ledger, fixed-point table, orbit-count row, and Nat/List count ledger are present before the local name can be read.

The fixed-point table is indexed by the displayed group-list ledger L, so it cannot be replaced by an external class function. The orbit-count row is read only after the fixed-point table and Nat/List ledger have exposed the finite average, so it cannot be replaced by a quotient orbit object or selected representative list.

Finally, H, C, P, N transport, replay, source, and name only the displayed rows. Since the carrier has no coordinate for quotient orbit sets, external cardinal arithmetic, infinite actions, cycle-index polynomials, Polya enumeration, or host equality of quotient types, no local NameCert obligation can depend on those objects. □

Theorem 9048.3 (Burnside lemma orbit-count route)**.** *Let $T = (G, X, A, L, F, O, K, H, C, P, N)$ be an accepted* $\mathsf{BurnsideLemma}^{\uparrow}$ *carrier. Every orbit-count read exported by T factors through the finite route*

$$\mathsf{Group}^{\uparrow} \longmapsto X \longmapsto A \longmapsto L \longmapsto F \longmapsto K \longmapsto O \longmapsto (H, C, P, N).$$

The stabilizer and fixed-orbit accounting consumed from chapter 6559 remains a finite ledger over the same action row from chapter 41; it is not a quotient orbit carrier, selected representative family, cycle-index polynomial, Polya enumeration surface, or ambient finite-action category.

Proof. Read the group and acted-list rows G and X, then read the action row A. The group-list ledger L enumerates only the displayed elements whose fixed points are inspected, and the table F records those fixed points as finite sublists of X.

The Nat/List ledger K is the only row where lengths, summation, divisibility, and the averaged fixed-point count are exported. The orbit-count row O is therefore downstream of the fixed-point table and the finite count ledger.

The dependency on chapter 6559 supplies stabilizer and fixed-orbit accounting for the same action surface. Transport through H and replay through C preserve that row order before P and N package the local certificate. Thus every consumer sees the finite Burnside ledger, while quotient orbit carriers, selected representatives, cycle-index polynomials, Polya enumeration, and ambient action categories remain outside the packet. □

Lean4 (statement) `BEDC.Derived.BurnsideLemmaUp`

Closure status for $\mathsf{NameCert}_{\mathsf{BurnsideLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{BurnsideLemma}^{\uparrow}$ packet is generated from a displayed finite GroupUp action source, a finite acted list, an action row, a group-list ledger, a fixed-point table, an orbit-count row, a Nat/List count ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BurnsideLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9048.1, Definition 9048.2, and Definition 9048.3 seed a finite group-action counting certificate over fixed-point and orbit-count ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BurnsideLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient orbit sets, selected orbit representatives, external cardinal arithmetic, infinite group actions, cycle-index polynomials, Polya enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fixed-point table exactness, Nat/List ledger soundness, hsame transport, Cont replay, provenance, and orbit-count nonescape for BEDC.Derived.BurnsideLemmaUp.

Chapter 9049

A Concrete Naming Certificate for $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ packet records a certified change of regular-Cauchy rate between two displayed regular-rational reading disciplines. It sits after the rate-realization route of chapter 1673, the regular rational surface of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the source-modulus rate transform of chapter 8294. The carrier changes the displayed rate schedule before downstream $\mathsf{Real}^{\uparrow}$ consumers inspect the seal, without quotienting sequences or choosing a limit.

Definition 9049.1 (Regular Cauchy rate transformer carrier)**.** *A* $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, D, A, B, R, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which both regular-Cauchy rate schedules are inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested target tolerances and the dyadic comparison budget for translating source-rate requests into target-rate requests.*
3. *A is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *rate row.*
4. *B is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *rate row.*
5. *R is the rate-change replay row. For every displayed target request it names the source request, the finite window read, and the dyadic tolerance comparison used to justify the transformed rate.*
6. *H stores componentwise* hsame *transport for S, D, A, B, R and the structural rows below.*
7. *C stores displayed* Cont *replay from the source rate through the window and dyadic ledger into the transformed target rate.*
8. *P records* Pkg *provenance over* $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$, $\mathsf{CauchyRateRealization}^{\uparrow}$, $\mathsf{RegularCauchyRateTran}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransformer}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, or a detached $\mathsf{Real}^{\uparrow}$ *seal.*

Theorem 9049.2 (Regular Cauchy rate transformer NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ *carrier* $T = (S, D, A, B, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransformer}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the finite window, dyadic tolerance ledger, source rate row, target rate row, rate-change replay, componentwise transport, continuation replay, provenance, and local naming;*
2. *source-rate exposure, requiring each transformed request to cite a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source-rate row* A*;*
3. *target-rate exposure, requiring the transformed request to land in the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *target-rate row* B *through the same* $\mathsf{StreamName}^{\uparrow}$ *window* S*;*
4. *dyadic comparison and replay, requiring* D *and* R *to record the tolerance budget and the finite route from target request back to source request;*
5. *structural stability and nonescape, requiring* H, C, P, N *to transport, replay, source, and name the same packet without adding quotient streams, selected limits, countable choice, host real equality, ambient completeness, or a detached real seal.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 9049.1. Its analytic rows are the window S, dyadic ledger D, source rate A, target rate B, and replay row R.

The source-rate and target-rate obligations are read from A and B. The finite window S keeps both schedules on the same displayed observation surface, so a transformed request cannot bypass the named StreamName row.

The dyadic comparison is exactly the row D, and the finite transformation route is exactly the row R. Together they record how a target tolerance is justified by a source request and a displayed comparison budget.

Finally H, C, P, N provide componentwise transport, continuation replay, provenance, and local naming for the same rows. Since the carrier omits quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, and detached real seals, the local certificate cannot export any of those principles. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ read is refuted if a target $\mathsf{RegSeqRat}^{\uparrow}$ rate request reaches a $\mathsf{Real}^{\uparrow}$ consumer without displaying the source-rate row A, the target-rate row B, the finite $\mathsf{StreamName}^{\uparrow}$ window S, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and the replay row R.

Independence witness: cauchy-rate-realization, regular-cauchy-rate-transform, regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular-Cauchy rate rows, a shared finite window, a dyadic comparison ledger, a rate-change replay row, transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransformer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyRateTransformer}^{\uparrow}$ packet is generated from a finite StreamName window, a DyadicRatCore tolerance ledger, source and target RegSeqRat rate rows, a rate-change replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateTransformer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9049.1 and Definition 9049.2 seed the finite rate-change certificate consumed by regular-rational and Real-facing readers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyRateTransformer}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-rate exposure, target-rate exposure, dyadic comparison, replay stability, transport, provenance, and local naming for the same finite packet.

Chapter 9050

A Concrete Naming Certificate for CylindricalAlgebraicDecomposition$^{\uparrow}$

Following Collins and the real-closed-field quantifier-elimination tradition, the CylindricalAlgebraicDecomposition$^{\uparrow}$ packet records a finite projection-and-lifting surface for a displayed polynomial family over a real closed field. It sits after the real-closed-field source of chapter 8977, the polynomial rows of chapter 50, the finite reduction ledgers of chapter 2197, the Sturm-chain route of chapter 2079, and the root-isolation carrier of chapter 2269. The packet names the cells and sign rows inspected by a cylindrical decomposition; it is not a host semialgebraic set, a selected root object, classical trichotomy, quotient equality, or choice.

Definition 9050.1 (Cylindrical algebraic decomposition carrier)**.** *A* CylindricalAlgebraicDecomposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, P, G, S, I, C, L, H, A, Q, N)$$

with the following displayed rows:

1. *R is the* RealClosedField$^{\uparrow}$ *source row, carrying the displayed ordered-field, polynomial, Sturm, and root-isolation reads available to the packet.*
2. *P is the polynomial-family row, listing the admitted coefficient arrays, variable levels, formal derivatives, and specialization requests.*
3. *G is the Groebner/projection ledger, recording resultants, discriminants, leading-coefficient rows, and the finite reduction data used to move from one cylinder level to the next.*
4. *S is the Sturm/root-count row for each projected one-variable request over an admitted base cell.*
5. *I is the root-isolation row, retaining isolating windows for the roots selected by S without exposing a host root element.*
6. *C is the sign-table row, recording the sign of every admitted polynomial on each retained section or sector.*
7. *L is the cell-stack and lifting row, listing base cells, lifted sections, lifted sectors, sample-window requests, and parent-child cylinder links.*
8. *H records componentwise* hsame *transport for the real-closed-field, polynomial, projection, Sturm, isolation, sign-table, lifting, replay, provenance, and local naming rows.*

9. *A records displayed* Cont *replay from any CAD read through the projection ledger, root-isolation rows, sign table, and cell stack.*
10. *Q is the* Pkg *provenance over* CylindricalAlgebraicDecomposition$^{\uparrow}$, RealClosedField$^{\uparrow}$, Poly$^{\uparrow}$, GroebnerBasis$^{\uparrow}$, SturmSequence$^{\uparrow}$, RealRootIsolation$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{CylindricalAlgebraicDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host semialgebraic sets, selected roots, classical trichotomy, quotient equality, global algebraic closure, or choice.

Theorem 9050.2 (Cylindrical algebraic decomposition NameCert obligations)**.** *For the carrier of Definition 9050.1, the local* NameCert$_{\text{CylindricalAlgebraicDecomposition}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $D = (R, P, G, S, I, C, L, H, A, Q, N)$ over the displayed* RealClosedField$^{\uparrow}$ *and* Poly$^{\uparrow}$ *sources;*
2. *projection coverage through G, requiring every discriminant, resultant, leading-coefficient, derivative, and specialization row used by the decomposition to appear in the finite projection ledger;*
3. *sign-invariance through C, requiring every admitted polynomial row from P to have one displayed sign row on each retained section or sector;*
4. *lifting and sample readback through S, I, L, requiring root counts, isolating windows, cell-stack links, and sample windows to be read before any higher cylinder level is accepted;*
5. *nonescape from host semialgebraic sets, selected roots, classical trichotomy, quotient equality, global algebraic closure, and choice, enforced by the transport, replay, provenance, and local naming rows H, A, Q, N.*

Proof. Project the carrier of Definition 9050.1. The rows R and P give the only ordered-field and polynomial sources, while G, S, I, C, L are the finite algebraic rows that can be inspected before the structural rows are read.

Projection coverage is finite because G lists exactly the resultants, discriminants, leading coefficients, derivatives, specialization rows, and reduction traces used by this packet. A consumer cannot appeal to an unlisted projection polynomial.

Sign-invariance is read from C only after the Sturm and isolation rows have split each base cell into the displayed sections and sectors. Thus the sign row is tied to a retained cell, not to a host semialgebraic object.

The lifting read is the sequence S, I, L: first count roots, then retain isolating windows, then attach the lifted section or sector to its parent cell and sample window. The structural rows H, A, Q, N transport, replay, source, and name that same finite packet. Since no coordinate stores host semialgebraic sets, selected roots, trichotomy, quotient equality, global algebraic closure, or choice, the five rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted CylindricalAlgebraicDecomposition$^{\uparrow}$ consumer that reads sign-invariance without the projection ledger, Sturm/root-isolation rows, and cell-stack lifting row is outside the carrier surface.

Independence witness: real-closed-field, polynomial, Groebner-basis, Sturm-sequence, and real-root-isolation: no listed sibling is bijective with CylindricalAlgebraicDecomposition$^{\uparrow}$ because this

packet binds one polynomial family to projection coverage, root isolation, sign tables, cylindrical lifting, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CylindricalAlgebraicDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CylindricalAlgebraicDecomposition}^{\uparrow}$ packet is generated from a RealClosedField source row, a polynomial-family row, a Groebner/projection ledger, Sturm/root-count rows, retained isolation windows, a sign table, a cylindrical cell stack, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CylindricalAlgebraicDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9050.1 and Definition 9050.2 seed the finite projection-cell packet over $\mathsf{RealClosedField}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{GroebnerBasis}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, and $\mathsf{RealRootIsolation}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CylindricalAlgebraicDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host semialgebraic sets, selected roots, classical trichotomy, quotient equality, global algebraic closure, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection coverage, sign-invariance, lifting and sample readback, transport, replay, provenance, and local naming for BEDC.Derived.CylindricalAlgebraicDecompositionUp.

Chapter 9051

A Concrete Naming Certificate for $\mathsf{WindowSixBetaSeed}^{\uparrow}$

The $\mathsf{WindowSixBetaSeed}^{\uparrow}$ packet realizes the internal beta seed of Definition 9346.60 and the infrared-stability row of Definition 9346.61 as finite NameCert data. It sits beside the finite golden-mean window of chapter 8802, the Zeckendorf phase shell of chapter 8811, the prime-edge classifier source of chapter 8890, and the beta-sign selection packet of chapter 8846. The packet names the local inverse-coupling derivative and its no-pole interval certificate; it is not a high-energy running law, a continuum renormalization group, or a physical coupling measurement.

Definition 9051.1 (Window-six beta seed carrier)**.** *A* $\mathsf{WindowSixBetaSeed}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W, G, I, S, R, H, C, P, N)$$

with the following displayed coordinates:

1. *W is the window-six finite source row read through* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$*.*
2. *G is the Green-excess spectral-response row read through the Zeckendorf phase-shell and prime-edge sibling surfaces.*
3. *I is the inverse-coupling derivative row. It records the local derivative $\frac{d}{dz}\alpha_{Z6}^{-1}(z)$ and its negative sign on the displayed activation interval.*
4. *S is the static beta-seed row, recording the displayed value at $z = 0$ and the finite small-z expansion used by the vision theorem.*
5. *R is the infrared-stability row. It records the denominator-root separation needed to keep the spectral response pole-free on the displayed interval.*
6. *H stores componentwise* hsame *transport for W, G, I, S, R and the structural rows below.*
7. *C stores displayed* Cont *replay from the window source through the derivative row and infrared-stability row.*
8. *P records* Pkg *provenance over* $\mathsf{WindowSixBetaSeed}^{\uparrow}$*,* $\mathsf{GoldenMeanShiftWindowSix}^{\uparrow}$*,* $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$*,* $\mathsf{ZeckendorfPrimeEdge}^{\uparrow}$*,* $\mathsf{BetaSignSelection}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WindowSixBetaSeed}^{\uparrow}}$ *row.*

The carrier has no coordinate for continuum running, a host analytic continuation, an unbounded energy-scale law, empirical coupling data, quotient spectra, or a detached real-completion theorem.

Theorem 9051.2 (Window-six beta seed NameCert obligations)**.** *For every accepted* $\mathsf{WindowSixBetaSeed}^{\uparrow}$ *carrier* $B = (W, G, I, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WindowSixBetaSeed}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the window source, spectral-response row, inverse-coupling derivative, static beta seed, infrared-stability row, transport, replay, provenance, and local naming;*
2. *inverse-coupling derivative exposure, requiring* I *to display the negative derivative row before any beta-seed consumer may read* S*;*
3. *static seed exposure, requiring* S *to store the finite small-*z *expansion and the displayed static value rather than a host analytic function;*
4. *infrared-stability exposure, requiring* R *to display the denominator-root separation that excludes poles on the inspected interval;*
5. *structural nonescape, requiring* H, C, P, N *to transport, replay, source, and name only the displayed packet rows.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 9051.1. The analytic rows visible to a consumer are W, G, I, S, R, and the structural rows are H, C, P, N.

The inverse-coupling derivative is read from I. It supplies the sign and derivative route before the static seed row is consumed, so the beta seed is not a detached number.

The static row S records the finite small-z expansion, while R records the no-pole denominator check on the inspected interval. These two rows realize the derivative and infrared-stability claims as finite packet data.

Finally H, C, P, N transport, replay, source, and name the same rows through the cited window-six and Zeckendorf siblings. Since the carrier omits continuum running, host analytic continuation, empirical coupling data, quotient spectra, and real-completion theorems, none of those readings can be exported by the local certificate. □

Falsifiable prediction: An accepted $\mathsf{WindowSixBetaSeed}^{\uparrow}$ read is refuted if a beta-seed consumer receives the static value without first displaying the window-six source W, the spectral-response row G, the inverse-coupling derivative row I, the infrared-stability row R, and the structural rows H, C, P, N.

Independence witness: golden-mean-shift-window-six, zeckendorf-phase-shell, zeckendorf-prime-edge, beta-sign-selection: $\mathsf{WindowSixBetaSeed}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the window source, spectral response, inverse-coupling derivative, static seed value, infrared-stability denominator row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowSixBetaSeed}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WindowSixBetaSeed}^{\uparrow}$ packet is generated from a window-six finite source row, a Green-excess spectral-response row, an inverse-coupling derivative row, a static beta-seed row, an infrared-stability row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WindowSixBetaSeed}^{\uparrow}}$, seed)

Scope closed: Definition 9051.1 and Definition 9051.2 seed the finite beta-seed packet over inverse-coupling derivative, static value, infrared-stability, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WindowSixBetaSeed}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a continuum running law, host analytic continuation, empirical coupling data, quotient spectrum, real-completion theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative exposure, static seed exposure, infrared-stability exposure, transport, replay, provenance, and local naming for the same finite packet.

Chapter 9052

A Concrete Naming Certificate for CountableCompletionDiagonalRepresentative$^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completions, the CountableCompletionDiagonalRepresentative$^{\uparrow}$ packet realizes the diagonal representative route of Definition 9357.20 in chapter 9357 as finite NameCert data. It sits over the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the modulus selector surface of chapter 1576. The packet names one diagonal readback route; it is not quotient stream equality, a selected completed limit, countable choice, host real equality, or ambient completeness.

Definition 9052.1 (Countable completion diagonal representative carrier)**.** *A* CountableCompletionDiagonalRepres*carrier is a finite* BHist *packet*

$$D = (F, M, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the regular-Cauchy family row, naming the displayed sequence of* RegSeqRat$^{\uparrow}$ *representatives inspected by the completion request.*
2. *M is the modulus-selector row, read through* RealCauchyModulusSelector$^{\uparrow}$*, which turns each dyadic tolerance request into a displayed family index.*
3. *T is the dyadic tolerance row over* DyadicRatCore$^{\uparrow}$*, including the requests corresponding to $1/(n+1)$ and the internal observation budget.*
4. *W is the* StreamName$^{\uparrow}$ *window row recording the finite family index, the selected regular name, and the internal observation index for each diagonal read.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row returning the dyadic value of the selected regular name at the selected observation index.*
6. *E is the* Real$^{\uparrow}$ *seal row available only after the regularity and diagonal tolerance rows have been displayed.*
7. *H records componentwise* hsame *transport for family, selector, tolerance, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a diagonal-representative request through F, M, T, W, R, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyModulusSelector}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected completed limits, countable choice, host real equality, ambient completeness, or bridge checking.

Theorem 9052.2 (Countable completion diagonal representative NameCert obligations)**.** *For the carrier of Definition 9052.1, the local* $\mathsf{NameCert}_{\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $D = (F, M, T, W, R, E, H, C, P, N)$;*
2. *modulus-index exposure through M and T, requiring every diagonal family index and observation index to be tied to a displayed dyadic tolerance request;*
3. *finite window readback through W and R, requiring the selected regular-Cauchy member and its internal* $\mathsf{RegSeqRat}^{\uparrow}$ *observation to be visible before a diagonal value is read;*
4. *Real-seal handoff through E, requiring regularity and diagonal tolerance rows before the completion-facing* $\mathsf{Real}^{\uparrow}$ *name is accepted;*
5. *transport, replay, provenance, local naming, and nonescape from quotient streams, selected limits, countable choice, host real equality, ambient completeness, and bridge checking through H, C, P, N.*

Proof. Project the carrier of Definition 9052.1. The rows F, M, T, W, R, E are the whole diagonal readback surface before transport, replay, provenance, and local naming are applied.

The selector row M and dyadic row T display the tolerance request used to choose the family index. The window row W then records that chosen family index together with the internal observation index, so no countable-choice schedule is hidden behind the diagonal notation.

The readback row R exposes the selected $\mathsf{RegSeqRat}^{\uparrow}$ value at the displayed observation index. The Real-seal row E is downstream of that readback and of the regularity/tolerance rows, so the packet names only the finite diagonal representative route.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for quotient stream equality, selected completed limits, countable choice, host real equality, ambient completeness, or bridge checking, the five rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}$ consumer that reads a Real seal without the modulus selector, dyadic tolerance request, StreamName window, and RegSeqRat readback rows is outside the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, and realcauchymodulusselector: no listed sibling is bijective with $\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}$ because this packet binds one regular-Cauchy family, one tolerance-index selector, one finite diagonal observation window, one RegSeqRat readback, one Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CountableCompletionDiagonalRepresentative}^{\uparrow}}$ **·**

Constructive story: A CountableCompletionDiagonalRepresentative$^{\uparrow}$ packet is generated from a regular-Cauchy family row, a RealCauchyModulusSelector row, dyadic tolerance requests, StreamName window rows, RegSeqRat readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CountableCompletionDiagonalRepresentative}^{\uparrow}}$, seed)

Scope closed: Definition 9052.1 and Definition 9052.2 seed the finite diagonal-representative packet over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and RealCauchyModulusSelector$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{CountableCompletionDiagonalRepresentative}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected completed limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-index exposure, finite window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CountableCompletionDiagonalRepresentativeUp.

Chapter 9053

A Concrete Naming Certificate for FiniteWindowUniformCompletion$^{\uparrow}$

Following Bishop and Bridges on constructive completion, and Bourbaki and Kelley on uniform completion, the FiniteWindowUniformCompletion$^{\uparrow}$ packet records the finite-window surface by which a uniform Cauchy completion request is inspected through StreamName windows, regular rational readback, and dyadic tolerance ledgers before any Real-facing seal is visible. It sits after the reflector of chapter 6282, the finite stream surface of chapter 126, the regular rational surface of chapter 301, and the dyadic core of chapter 291; it also reads the terminal Real$^{\uparrow}$ seal as a named row rather than as ambient completeness.

Definition 9053.1 (Finite-window uniform completion carrier)**.** *A* FiniteWindowUniformCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, U, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* UniformCompletionReflector$^{\uparrow}$ *row, naming the source uniform object, completion target, dense route, and reflector-facing route visible to the packet.*
2. *U is the* UniformCauchyCompletion$^{\uparrow}$ *row, recording the uniform Cauchy family and the completion-facing request inspected by the finite window.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row; it exposes only the stream windows used by the uniform Cauchy request.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, budgeting every displayed comparison made by the finite windows.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row on those same windows and dyadic tolerances.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after R, U, S, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the reflector, uniform Cauchy, stream-window, dyadic-tolerance, regular-readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform completion request through $R, U, S, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* FiniteWindowUniformCompletion$^{\uparrow}$, UniformCompletionReflector$^{\uparrow}$, UniformCauchyCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowUniformCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, U, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient completion objects, selected limit representatives, countable choice, host equality of completed points, detached Real completeness, or an ambient completion theorem.

Theorem 9053.2 (Finite-window uniform completion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowUniformCompletion}^{\uparrow}$ *carrier $W = (R, U, S, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowUniformCom}}$ *surface consists exactly of the following finite obligation rows:*

1. *reflector admission through the* $\mathsf{UniformCompletionReflector}^{\uparrow}$ *row R;*
2. *uniform Cauchy admission through the* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *row U;*
3. *finite observation through the* $\mathsf{StreamName}^{\uparrow}$ *row S;*
4. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
5. *regular rational readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q;*
6. *terminal sealing through the* $\mathsf{Real}^{\uparrow}$ *row E;*
7. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
8. Pkg *provenance and local naming through P and N.*

These rows exhaust the certificate surface and add no quotient completion, selected limit, countable-choice, host-equality, detached-completeness, or ambient-completion coordinate.

Proof. Project the carrier of Definition 9053.1. Its visible coordinates are $R, U, S, D, Q, E, H, C, P, N$, so every local obligation must be read from one of those rows.

The completion request enters through the reflector row R and the uniform Cauchy row U. The only observations available to that request are the StreamName windows S, the dyadic tolerance row D, and the RegSeqRat readback row Q.

The terminal row is E. Since E is downstream of the finite observation stack, a Real-facing consumer cannot read a completed point, selected representative, or ambient completeness theorem before the displayed rows have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. No refused host-side principle appears as a carrier coordinate, so the listed rows exhaust the local NameCert obligations. □

Theorem 9053.3 (Finite-window uniform completion nonescape)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{FiniteWindowUniformCompletion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{UniformCompletionReflector}^{\uparrow}, \mathsf{UniformCauchyCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The route is transported by hsame*, replayed by* Cont*, sourced by* Pkg*, and locally named by* $\mathsf{NameCert}$ *through the rows H, C, P, N. It exports no quotient completion object, selected limit representative, countable-choice schedule, host equality of completed points, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient uniform-completion theorem.*

Proof. Start with an accepted packet $W = (R, U, S, D, Q, E, H, C, P, N)$. The reflector and uniform Cauchy rows R, U are the only completion-request coordinates.

The finite observational stack is S, D, Q. These rows expose the stream windows, tolerance ledger, and regular rational readback used by the request, and no consumer can bypass them because no other observation coordinate is present.

The only terminal seal is E. The structural rows H, C, P, N preserve and name the same displayed route, while the carrier has no quotient, selection, choice, host-equality, detached-completeness, or ambient-completion coordinate. □

Falsifiable prediction: Every accepted $\mathsf{FiniteWindowUniformCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing boundary while omitting the reflector row, uniform Cauchy row, StreamName windows, DyadicRatCore tolerance row, RegSeqRat readback row, Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: uniform-completion-reflector, uniform-cauchy-completion, streamname, dyadicratcore, regseqrat, real: $\mathsf{FiniteWindowUniformCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the reflector route, uniform Cauchy request, finite windows, dyadic tolerances, regular readback, Real seal, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowUniformCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowUniformCompletion}^{\uparrow}$ packet is generated from UniformCompletionReflector, UniformCauchyCompletion, StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9053.1, Definition 9053.2, Definition 9053.3, the falsifiable prediction, and the independence witness seed the finite-window uniform completion route from uniform Cauchy data to Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion objects, selected limit representatives, countable choice, host equality of completed points, detached $\mathsf{Real}^{\uparrow}$ completeness, ambient uniform completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector admission, uniform Cauchy admission, finite-window readback, dyadic tolerance exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowUniformCompletionUp.

Chapter 9054

A Concrete Naming Certificate for $\mathsf{BanachFixedPointContraction}^{\uparrow}$

Following Banach, the $\mathsf{BanachFixedPointContraction}^{\uparrow}$ packet records the contraction-mapping principle as finite BEDC NameCert data over a complete metric source, a displayed Lipschitz self-map, a Picard iterate stream, a regular rational Cauchy handoff, and a Real-facing endpoint boundary. It sits beside the complete metric source of chapter 128, the metric comparison surface of chapter 57, the Lipschitz map packet of chapter 3762, the stream-name windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The chapter names a standard contraction fixed-point neighbour for completion and continuous-map consumers; it does not assert host fixed-point equality, quotient stream equality, selected limits, or ambient completeness outside the displayed $\mathsf{CompleteMetric}^{\uparrow}$ row.

Definition 9054.1 (Banach fixed-point contraction carrier)**.** *A* $\mathsf{BanachFixedPointContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, d, L, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, exposing the admitted carrier, the completion boundary, and the finite metric-comparison service.*
2. *d is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare adjacent Picard iterates and later endpoint windows.*
3. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *self-map row with a displayed rational contraction ledger $0 \leq q < 1$.*
4. *S is the finite Picard schedule row, recording the inspected iterates $x_0, \ldots, x_n$ and the application rows $x_{k+1} = L(x_k)$.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the scheduled iterates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the contraction ledger and the finite stream windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal boundary reached only after R and the carried complete metric route are displayed.*
8. *H records componentwise* hsame *transport for M, d, L, S, W, R, E and the local certificate rows.*
9. *C records displayed* Cont *replay for metric reads, Lipschitz-map application, Picard-window extension, regular rational readback, and endpoint sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BanachFixedPointContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointContraction}^{\uparrow}}$ *row.*

The classifier accepts a contraction fixed-point read only through these rows. It has no coordinate for an externally selected fixed point, host equality of endpoints, a quotient of Cauchy streams, a non-displayed completeness theorem, or countable choice.

Theorem 9054.2 (Banach fixed-point contraction obligation surface)**.** *For every accepted* BanachFixedPointContrac *carrier $K = (M, d, L, S, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointContraction}^{\uparrow}}$ *surface consists exactly of complete metric source visibility, metric comparison visibility, Lipschitz self-map visibility, rational contraction ledger visibility, finite Picard schedule visibility, StreamName window visibility, RegSeqRat readback visibility, Real endpoint seal nonescape, componentwise transport, displayed replay, package provenance, and local naming for the packet of Definition 9054.1.*

Proof. Project the carrier. The analytic source rows are M, d, L, the iteration rows are S, W, the completion-facing rows are R, E, and the structural rows are H, C, P, N.

The contraction ledger belongs to L, so a consumer may spend a shrinkage bound only after the self-map row and metric row are visible. The finite Picard schedule S records each inspected application step, while W repacks those steps as stream windows.

The readback row R and endpoint seal E are downstream of that finite route. Since no excluded object is a coordinate of K, the listed rows exhaust the local naming surface. □

Theorem 9054.3 (Banach fixed-point contraction iterate Cauchy route)**.** *Every Cauchy read of an accepted* $\mathsf{BanachFixedPointContraction}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{LipschitzMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The finite Picard schedule first reads adjacent iterate distances through d and the contraction ledger in L, then packages the inspected windows through W, then exposes the regular rational readback R, and only then reaches the endpoint seal E. The route does not export host convergence, a selected limit, quotient stream equality, or ambient Real completeness.

Proof. Start with the finite schedule S. Each adjacent row is an application of the displayed Lipschitz self-map L, and every distance comparison is read through the metric row d over the complete metric source M.

The rational contraction ledger in L supplies the tail estimate for the inspected Picard windows. Those windows are recorded by W as a $\mathsf{StreamName}^{\uparrow}$ row rather than as a completed infinite orbit.

The regular rational readback R is therefore generated from displayed finite data, and the endpoint boundary E is reached only after that readback and the carried complete metric route are visible. The absent host objects cannot be recovered from this route because the carrier has no coordinate for them. □

Theorem 9054.4 (Banach fixed-point contraction unique fixed-point boundary)**.** *If two accepted* $\mathsf{BanachFixedPointContraction}^{\uparrow}$ *carriers have the same displayed complete metric row, metric row, Lipschitz self-map row, rational contraction ledger, Picard-window readback, and structural transports, then their* $\mathsf{Real}^{\uparrow}$ *endpoint seal rows are classifier-determinate. This boundary is the BEDC fixed-point uniqueness surface; it is not a host equality statement about completed real points.*

Proof. Compare the two carriers componentwise through their H and C rows. The shared rows M, d, L, S, W, R force both endpoint routes to spend the same contraction ledger and the same regular rational readback.

The usual contraction estimate is used here only as a finite classifier pattern: a rational bound below one identifies two displayed endpoint routes after both are routed through the same Picard windows and completion handoff.

Thus the endpoint rows agree at the $\mathsf{Real}^{\uparrow}$ seal boundary seen by the BEDC classifier. Because the packet contains no host endpoint object, the conclusion cannot be read as external equality. □

Falsifiable prediction: An accepted $\mathsf{BanachFixedPointContraction}^{\uparrow}$ consumer is refuted if it reads a fixed-point endpoint without displaying the complete metric source, metric comparison row, Lipschitz contraction ledger, Picard schedule, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local NameCert rows.

Independence witness: complete-metric, metric, lipschitz-map, streamname, regseqrat, real: $\mathsf{BanachFixedPointContraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a complete metric source, a contraction self-map, a Picard schedule, finite stream windows, a regular rational readback, a Real endpoint seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointContraction}^{\uparrow}$ packet is generated from a complete metric source row, metric comparison row, Lipschitz self-map row with rational contraction ledger, finite Picard schedule, StreamName windows, RegSeqRat readback, Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9054.1, Definition 9054.2, Definition 9054.3, and Definition 9054.4 seed the finite contraction fixed-point carrier and its completion-facing endpoint boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host fixed-point equality, selected limits, quotient stream equality, countable choice, or ambient completeness beyond the carried $\mathsf{CompleteMetric}^{\uparrow}$ row.

Upgrade path: Obligation closure requires checked carrier admission, contraction-ledger visibility, Picard-window Cauchy readback, endpoint seal nonescape, uniqueness-boundary determinacy, transport, replay, provenance, local naming, and TasteGate support for BEDC.Derived.BanachFixedPointContractionUp.

Chapter 9055

A Concrete Naming Certificate for $\mathsf{Transportability}^{\uparrow}$

Following Pearl and Bareinboim on causal transportability, the $\mathsf{Transportability}^{\uparrow}$ packet records the finite source-to-target selection surface by which causal effects can be transported across displayed domains. It sits after the intervention calculus of chapter 453, the mediator-facing front-door packet of chapter 7600, and the paired-world outcome surface of chapter 3989. The carrier names the source-domain distribution, target selection row, intervention ledger, adjustment ledger, transported distribution row, justification ledger, transport, replay, provenance, and local naming rows; it does not assert a hidden selection oracle or host-level external validity.

Definition 9055.1 (Transportability carrier)**.** *A* $\mathsf{Transportability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, Tg, I, A, D, J, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-domain distribution row whose observational or interventional source probabilities are exposed before any transport read.*
2. *Tg is the target-domain selection row, recording which structural mechanisms or domain coordinates are allowed to differ.*
3. *I is the intervention ledger shared with the* $\mathsf{DoCalculus}^{\uparrow}$ *surface.*
4. *A is the adjustment ledger, including front-door or other displayed admissible adjustment rows when they are consumed.*
5. *D is the transported distribution row produced only after S, Tg, I, A have been replayed.*
6. *J is the justification ledger naming the do-calculus and causal-identification handoff used by the transport step.*
7. *H records componentwise* hsame *transport for source distribution, target selection, intervention, adjustment, transported distribution, justification, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source-domain request through the selection row, intervention ledger, adjustment ledger, transported distribution, and justification rows.*
9. *P is the* Pkg *provenance over* $\mathsf{Transportability}^{\uparrow}$, $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{FrontdoorCriterion}^{\uparrow}$, $\mathsf{CounterfactualOutcome}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden selection oracle, host probability equality, unlicensed counterfactual selector, or ambient causal completeness.

Theorem 9055.2 (Transportability NameCert obligations)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *surface is exhausted by the following obligation rows:*

1. *carrier admission for the source-domain distribution, target-domain selection, intervention, adjustment, transported distribution, justification, transport, replay, provenance, and local naming rows;*
2. *selection-row exposure, requiring the target-domain selection row Tg to be displayed before any source-to-target distribution read is served;*
3. *do-calculus handoff, requiring the intervention ledger I and justification ledger J to cite the displayed do-calculus route before a transported distribution row is accepted;*
4. *source-to-target distribution transport, requiring S, Tg, I, A, J to replay through C before D and the local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *row are exposed;*
5. *refusal of hidden selection oracle, host probability equality, unlicensed counterfactual selector, and ambient causal completeness.*

No transportability obligation is stored outside these rows.

Proof. Project the carrier of Definition 9055.1. The rows $S, Tg, I, A, D, J, H, C, P, N$ are the only rows available to a consumer.

The selection row Tg is read before the transported distribution row D. The intervention ledger I and adjustment ledger A are then justified by J, which points to the do-calculus and front-door handoff surface rather than to an unrecorded host causal model.

Replay through C fixes the consumption order from source distribution and target selection to intervention, adjustment, justification, and transported distribution. Componentwise transport through H and provenance through P, N preserve that same finite packet, so the excluded hidden-oracle and host-completeness data cannot be exported. □

Lean4 ✓ `BEDC.Derived.TransportabilityUp.TransportabilityCarrier namecert obligations`

Theorem 9055.3 (Transportability carrier admission)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, the admitted carrier rows are exactly the source-domain distribution row S, target-domain selection row Tg, intervention ledger I, adjustment ledger A, transported distribution row D, justification ledger J, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N.*

Proof. Project the carrier of Definition 9055.1. Its displayed tuple has precisely the ten coordinates $S, Tg, I, A, D, J, H, C, P, N$.

The obligation surface of Definition 9055.2 names carrier admission only for those coordinates. Since the carrier has no hidden selection, host probability, unlicensed counterfactual, or ambient completeness row, no additional row can be admitted. □

Theorem 9055.4 (Transportability selection-ledger obligation)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, every transported-effect consumer must expose the source-domain distribution row S, target selection row Tg, intervention ledger I, adjustment ledger A,*

transported distribution row D, justification ledger J, componentwise transport H, replay C, provenance P, and local naming N before reading the transported result. The same packet therefore blocks hidden selection oracles, host probability equality, and ambient causal completeness from entering the public transportability surface.

Proof. Project the carrier of Definition 9055.1. The only rows available to a transported-effect consumer are $S, Tg, I, A, D, J, H, C, P, N$, and the carrier explicitly has no hidden selection-oracle coordinate.

Apply Definition 9055.2. Its source-to-target transport clause requires S, Tg, I, A, J to replay through C before D and the local naming row are exposed.

The sibling routes through do-calculus, front-door adjustment, and counterfactual-outcome bookkeeping are read only by the displayed ledgers I, A, J. Componentwise transport through H and provenance through P, N preserve the same finite packet, so no host probability equality or ambient causal-completeness assumption can be consumed outside the ledger. □

Theorem 9055.5 (Transportability selection row exposure)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier $T = (S, Tg, I, A, D, J, H, C, P, N)$, the target-domain selection row Tg is displayed before any transported distribution row D is public. A consumer that reads D without first reading S and Tg has left the finite carrier of Definition 9055.1 and the obligation surface of Definition 9055.2.*

Proof. Project the carrier rows $S, Tg, I, A, D, J, H, C, P, N$ from Definition 9055.1. The target selection row Tg is a named coordinate of the same finite packet as the transported distribution row D.

Apply Definition 9055.2. Its selection-row exposure clause requires Tg to be displayed before any source-to-target distribution read is served, and its transport clause requires S, Tg, I, A, J to replay before D and the local naming row are exposed.

Thus the only public route to D passes through S and Tg. Since the carrier has no hidden selection oracle or host external-validity coordinate, a route that omits Tg is outside the accepted $\mathsf{Transportability}^{\uparrow}$ packet. □

Theorem 9055.6 (Transportability distribution transport)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier $T = (S, Tg, I, A, D, J, H, C, P, N)$, the rows S, Tg, I, A, J replay through C before the transported distribution row D and the local naming row N are exposed.*

Proof. The carrier of Definition 9055.1 separates the source distribution, selection row, intervention ledger, adjustment ledger, justification ledger, replay row, transported distribution, and local naming rows.

Use the source-to-target distribution transport clause in Definition 9055.2. It places S, Tg, I, A, J upstream of D through the displayed replay C, and it names the local $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ row only after that replay route is available. □

Theorem 9055.7 (Transportability source-target replay)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier $T = (S, Tg, I, A, D, J, H, C, P, N)$, the only public route to the transported distribution row D is the displayed replay from source distribution S through target selection Tg, intervention ledger I, adjustment ledger A, and justification ledger J, followed by componentwise transport H, replay C, provenance P, and local naming N.*

Proof. Project the rows of Definition 9055.1. The source distribution, target selection, intervention, adjustment, transported distribution, justification, transport, replay, provenance, and local naming rows are exactly $S, Tg, I, A, D, J, H, C, P, N$.

Use Definition 9055.2 to read the source-to-target distribution transport clause. It requires S, Tg, I, A, J to replay through C before D and the local $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ row can be exposed.

The structural rows close the route. Componentwise transport through H preserves the same displayed coordinates, provenance through P records the allowed sibling surfaces, and N names the local packet. No carrier coordinate stores an alternate transported distribution, hidden host probability equality, or unlicensed counterfactual selector, so the stated replay is the unique public route. □

Theorem 9055.8 (Transportability domain-selection coverage)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, a read of the transported distribution row* D *must expose source distribution* S*, target selection* Tg*, intervention ledger* I*, adjustment ledger* A*, justification ledger* J*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *row* N*. The selection coverage surface is therefore the finite carrier route from the source-domain distribution through the target-domain selection row before any transported distribution is public.*

Proof. Project the carrier of Definition 9055.1. Its displayed rows are exactly $S, Tg, I, A, D, J, H, C, P, N$.

Apply source-target replay, Definition 9055.7. It identifies the only public route to D as the replay from source distribution through target selection, intervention, adjustment, and justification, followed by transport, replay, provenance, and local naming.

The selection row exposure theorem Definition 9055.5 prevents D from being read before S and Tg are visible. Since no alternate transported distribution or hidden selection row exists in the carrier, domain-selection coverage is exhausted by the stated rows. □

Theorem 9055.9 (Transportability intervention-adjustment ledger)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, each transported-distribution read through* D *must retain both the intervention ledger* I *and adjustment ledger* A *together with source distribution* S*, target selection* Tg*, justification ledger* J*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* N*. The intervention and adjustment rows are not optional annotations; they are part of the carrier route through which* D *is admitted.*

Proof. Project the carrier rows in Definition 9055.1. The intervention ledger I and adjustment ledger A are coordinates of the same finite packet as source distribution S, target selection Tg, transported distribution D, and justification ledger J.

Use the NameCert obligations of Definition 9055.2. The do-calculus handoff requires I and J to cite the displayed intervention route, while the source-to-target transport clause requires S, Tg, I, A, J to replay through C before D and N are exposed.

Componentwise transport through H and provenance through P preserve that same ledger. Therefore no transported-distribution read can bypass the intervention or adjustment rows without leaving the accepted $\mathsf{Transportability}^{\uparrow}$ carrier. □

Theorem 9055.10 (Transportability target-distribution nonescape)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, the target-side transported distribution* D *is readable only through source distribution* S*, target selection* Tg*, intervention ledger* I*, adjustment ledger* A*, justification ledger* J*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *row* N*. Thus* D *cannot escape into a hidden selection oracle, host probability equality, unlicensed counterfactual selector, or ambient causal-completeness principle.*

Proof. Start from the carrier admission theorem Definition 9055.3. It lists $S, Tg, I, A, D, J, H, C, P, N$ as the full admitted row surface and records no hidden host-level target distribution coordinate.

Apply domain-selection coverage and the intervention-adjustment ledger, Definition 9055.8 and Definition 9055.9. Together they force every read of D to pass through source distribution, target selection, intervention, adjustment, justification, transport, replay, provenance, and local naming rows.

Finally use the refusal clause in Definition 9055.2. Since hidden selection oracles, host probability equality, unlicensed counterfactual selectors, and ambient causal completeness are absent from the finite carrier, the target distribution has no route outside the stated ledger. □

Theorem 9055.11 (Transportability domain-selection obligation)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, a transported-effect read is admissible only after the source-domain distribution row* S *and target-domain selection row* Tg *have both been exposed. The selection obligation is componentwise:* S *supplies the source distribution being transported,* Tg *supplies the target-domain mechanism boundary, and the later rows* I, A, D, J, H, C, P, N *may be consumed only downstream of that pair.*

Proof. Project the carrier of Definition 9055.1. The source row S and target row Tg are separate coordinates, so the target side cannot be recovered by reading the source row alone.

Apply Definition 9055.5. It places Tg before the transported distribution row D, and Definition 9055.8 keeps S and Tg together on every route to D.

The structural rows do not change the source-target order. Intervention and adjustment are later ledger rows, while H, C, P, N transport, replay, source, and name the same displayed route. Therefore the domain-selection obligation is exactly the displayed S, Tg boundary followed by the remaining carrier rows. □

Theorem 9055.12 (Transportability adjustment-ledger exactness)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, the intervention-adjustment portion of a transported read is exactly the pair of ledger rows* I, A *together with the justification ledger* J*. The row* I *records the displayed* $\mathsf{DoCalculus}^{\uparrow}$ *intervention route,* A *records the displayed adjustment route, including any* $\mathsf{FrontdoorCriterion}^{\uparrow}$ *row actually consumed, and* J *records the cited handoff; no hidden adjustment oracle or host probability equality is added.*

Proof. Start from Definition 9055.1. The only intervention and adjustment coordinates are I and A, and the only justification coordinate is J.

Use the do-calculus handoff clause of Definition 9055.2. It requires I and J to cite the displayed do-calculus route before a transported distribution is accepted. Use Definition 9055.9 to keep A in the same route whenever D is read.

The provenance row P records the allowed $\mathsf{DoCalculus}^{\uparrow}$ and $\mathsf{FrontdoorCriterion}^{\uparrow}$ sources, while C replays the displayed source-to-target route. Since hidden adjustment data and host probability equality are absent from the carrier, ledger exactness is precisely the finite I, A, J package. □

Theorem 9055.13 (Transportability target replay coverage)**.** *For every accepted* $\mathsf{Transportability}^{\uparrow}$ *carrier* $T = (S, Tg, I, A, D, J, H, C, P, N)$*, every exported transported-distribution read factors through source distribution* S*, target selection* Tg*, intervention ledger* I*, adjustment ledger* A*, transported row* D*, justification ledger* J*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *row* N*. No exported target read bypasses one of these coordinates.*

Proof. Project the full carrier in Definition 9055.1. Its displayed coordinates are exactly $S, Tg, I, A, D, J, H, C, P, N$.

Apply the source-target replay theorem Definition 9055.7. It identifies the public route to D as source distribution, target selection, intervention, adjustment, and justification followed by structural transport, replay, provenance, and naming.

Now apply Definition 9055.10. Since the transported distribution cannot escape into hidden selection, host probability equality, unlicensed counterfactual selection, or ambient causal completeness, every exported target read is covered by the displayed replay route and no extra coordinate remains. □

Theorem 9055.14 (Transportability scope exhaustion)**.** *Every accepted* $\mathsf{Transportability}^{\uparrow}$ *read factors through the displayed* BHist *rows*

$$S, Tg, I, A, D, J, H, C, P, N,$$

componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$ *naming, and the sibling* $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{FrontdoorCriterion}^{\uparrow}$, *and* $\mathsf{CounterfactualOutcome}^{\uparrow}$ *routes. The packet exports no hidden selection oracle, host probability equality, unlicensed counterfactual selector, or ambient causal completeness.*

Proof. Project the carrier of Definition 9055.1. The displayed rows $S, Tg, I, A, D, J, H, C, P, N$ are the complete local packet, and Definition 9055.13 requires every exported target read to pass through them.

Read the structural rows next. Componentwise hsame transport is stored in H, displayed Cont replay is stored in C, Pkg provenance is stored in P, and local naming is stored in N.

Finally apply Definition 9055.12, Definition 9055.9, and Definition 9055.10. These keep the do-calculus, front-door adjustment, and counterfactual-outcome routes inside the displayed ledgers while refusing hidden selection, host probability equality, unlicensed counterfactual selection, and ambient causal completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$.

Constructive story: A $\mathsf{Transportability}^{\uparrow}$ packet is generated from source-domain distribution, target-domain selection, intervention, adjustment, transported distribution, justification, componentwise transport, displayed replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 9055.1, Definition 9055.2, and Definition 9055.14 bind the finite transportability carrier to its BEDC primitive rows and sibling do-calculus, front-door, and counterfactual routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Transportability}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.TransportabilityUp.TransportabilityCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close hidden selection oracles, host probability equality, unlicensed counterfactual selectors, ambient causal completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public transportability export theorem over the displayed source-target rows and sibling causal routes.

Chapter 9056

A Concrete Naming Certificate for $\mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}$

The $\mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}$ packet realizes the proof target of chapter 9246 as finite NameCert data. It records a critical-strip row, a fixed-half comparison row, an apartness witness, a zero-ledger request, a zeta-continuation application row, a prime-window recursive tower row, a compatibility ledger, transport, replay, provenance, and local naming before any consumer may exclude a zero away from the fixed surface. The packet is a certificate route for the local-global zero exclusion target; it is not a host proof of the Riemann hypothesis or an ambient completeness assertion for zeta zeros.

The fixed-half comparison row is consumed from chapter 9046, whose fixed-surface witness supplies the displayed comparison boundary before the local-global zero-ledger row may be read.

The zeta-continuation application row K is consumed from chapter 588: the certificate reads the displayed eta, functional-equation, pole-zero ledger, Gamma-horizon, and application-route rows before the local-global compatibility ledger G can be exposed.

Definition 9056.1 (Local-global zero exclusion certificate carrier)**.** *A* $\mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, F, A, Z, K, T, G, H, R, P, N)$$

with the following displayed rows:

1. *C is the critical-strip carrier row being tested.*
2. *F is the fixed-half comparison row naming the mirror-fixed surface.*
3. *A is the apartness witness row separating C from F by displayed positive rational data.*
4. *Z is the zero-ledger request row whose consumption is blocked until A has been exposed.*
5. *K is the zeta-continuation application row used to read the completed zeta carrier.*
6. *T is the prime-window recursive tower row supplying local source compatibility.*
7. *G is the global compatibility ledger tying the local window rows to the completed carrier.*
8. *H records componentwise* hsame *transport for critical-strip, fixed-half, apartness, zero-ledger, zeta-continuation, tower, compatibility, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from the critical-strip request through the apartness witness, zero-ledger gate, zeta-continuation application, tower row, and compatibility ledger.*

10. *P is the* Pkg *provenance over* LocalGlobalZeroExclusionCertificate$^{\uparrow}$, FixedHalfZeroWitness$^{\uparrow}$, *the zeta-continuation application route, the prime-window recursive tower,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{LocalGlobalZeroExclusionCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a zero row consumed before apartness, a selected off-fixed zero, a hidden completed-carrier equality, host analytic continuation, or ambient RH completeness.

Theorem 9056.2 (Local-global zero exclusion carrier certificate)**.** *For every accepted* LocalGlobalZeroExclusionCerti *carrier $E = (C, F, A, Z, K, T, G, H, R, P, N)$, a consumer may consume the zero-ledger request row Z only after the apartness witness row A is displayed against the fixed-half comparison row F. The accepted route then reads the zeta-continuation application row K, prime-window recursive tower row T, compatibility ledger G, componentwise transport H, replay R, provenance P, and local* NameCert$_{\text{LocalGlobalZeroExclusionCertificate}^{\uparrow}}$ *row N before any local-global zero exclusion claim is public. The packet exports no selected off-fixed zero, hidden carrier equality, host analytic continuation theorem, or ambient RH completeness.*

Proof. Project the carrier of Definition 9056.1. The rows $C, F, A, Z, K, T, G, H, R, P, N$ are the entire public surface of the packet.

The apartness row A is the gate for the zero-ledger request Z. Thus a consumer that attempts to read Z without first displaying A and the fixed-half comparison row F has left the accepted carrier.

After that gate, replay through R fixes the remaining order: the zeta-continuation application row K, prime-window recursive tower row T, and compatibility ledger G are read before transport, provenance, and local naming. Since no carrier coordinate stores a selected off-fixed zero, hidden completed-carrier equality, host analytic continuation theorem, or ambient RH completeness, the certificate is exactly the finite non-escape route stated above. □

Falsifiable prediction: Any accepted off-fixed zero row must expose the apartness witness before the zero-ledger row can be consumed.

Independence witness: fixed-half-zero-witness, zeta-continuation-application, prime-window-recursive-tower, real: none is bijective to LocalGlobalZeroExclusionCertificate$^{\uparrow}$ because the packet jointly carries critical-strip input, fixed-half comparison, apartness, zero-ledger gate, zeta-continuation application, prime-window tower, compatibility, transport, replay, provenance, and local naming rows.

Closure status for NameCert$_{\text{LocalGlobalZeroExclusionCertificate}^{\uparrow}}$.

Constructive story: A LocalGlobalZeroExclusionCertificate$^{\uparrow}$ packet is generated from critical-strip input, fixed-half comparison, apartness witness, zero-ledger gate, zeta-continuation application, prime-window tower, compatibility, componentwise transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{LocalGlobalZeroExclusionCertificate}^{\uparrow}}$, seed)

Scope closed: Definition 9056.1, Definition 9056.2, the falsifiable prediction, and the independence witness seed the finite NameCert carrier for the local-global zero exclusion target over critical-strip, fixed-half, apartness, zero-ledger, zeta-continuation, tower, compatibility, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{LocalGlobalZeroExclusionCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host proof of RH, selected off-fixed zeros, host analytic continuation, hidden completed-carrier equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-before-zero-ledger gating, zeta-continuation application coverage, prime-window tower compatibility, replay non-escape, and the Nontrivial and Field-Faithful rows for the same packet.

Chapter 9057

A Concrete Naming Certificate for $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$ packet records the finite boundary where sequential completion reads, apartness evidence, and Real-facing seals meet without being allowed to collapse into quotient completions or host-selected limits. It sits after the apartness source surface of chapter 1514, the generic apartness completion route of chapter 1787, the located apartness completion route of chapter 3565, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real seal discipline of chapter 38. The packet is a Real and Completion seed boundary; it does not assert quotient equality of completion names, selected limits, trichotomy, countable choice, or ambient completeness.

Definition 9057.1 (Sequential apartness completion boundary carrier)**.** *A* $\mathsf{SequentialApartnessCompletionBoundary}$ *carrier is a finite* BHist *packet*

$$B = (A, Q, L, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ApartnessSpace}^{\uparrow}$ *source row exposing positive separation before any completion-facing read is consumed.*
2. *Q is the* $\mathsf{ApartnessCompletion}^{\uparrow}$ *boundary row recording the separated completion route attached to the same apartness source.*
3. *L is the* $\mathsf{LocatedApartnessCompletion}^{\uparrow}$ *row displaying located completion windows and apartness evidence before Real-facing inspection.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window schedule for the sequential observations seen by the completion boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback extracted from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording positive gaps, dyadic radii, and comparison cells used by the readback.*
7. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal row, available only downstream of A, Q, L, S, R, D.*
8. *H records componentwise* hsame *transport for the apartness-space row, apartness-completion row, located-apartness-completion row, stream windows, regular readback, dyadic ledger, optional seal, replay row, provenance row, and local NameCert row.*

9. *C records displayed* Cont *replay from apartness inspection through apartness completion, located apartness completion, stream windows, regular readback, dyadic tolerance, and the optional Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$, $\mathsf{ApartnessSpace}^{\uparrow}$, $\mathsf{ApartnessCompletion}^{\uparrow}$, $\mathsf{LocatedApartnessCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion classes, host equality of completed points, selected limits, countable-choice schedules, trichotomy rows, or ambient Real completeness.

Theorem 9057.2 (Sequential apartness completion boundary NameCert obligations)**.** *For every accepted* $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$ *carrier* $B = (A, Q, L, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed apartness-space source, apartness-completion boundary, located-apartness-completion route, stream windows, regular readback, dyadic tolerance ledger, optional Real seal, transport, replay, provenance, and local naming rows;*

2. *source-first exactness, requiring A to be displayed before the completion rows Q and L are consumed;*

3. *sequential readback discipline, requiring every completion-facing observation to pass through S, R, and D before the optional seal row E is available;*

4. *componentwise stability, requiring H and C to transport and replay only the displayed rows $A, Q, L, S, R, D, E, P, N$;*

5. *provenance exhaustion, requiring P and N to export no quotient completion class, host equality of completed names, selected limit, countable-choice schedule, trichotomy row, or ambient-completeness principle.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 9057.1. The tuple displays the apartness source, apartness-completion boundary, located-apartness-completion route, finite windows, regular readback, dyadic tolerance ledger, optional Real seal, transport, replay, provenance, and local name.

Source-first exactness follows from the row order. A completion-facing read can inspect Q or L only after the positive separation row A has been named, so the completion boundary cannot hide its apartness source.

The sequential readback discipline is carried by S, R, D, E. Each observation first opens a finite stream window, then reads the regular rational row, then consults the dyadic tolerance ledger; the optional Real seal is downstream of that finite route.

Componentwise stability and provenance exhaustion are the roles of H, C, P, N. These rows transport, replay, package, and name only the displayed packet, and therefore cannot add quotient completion classes, host equality, selected limits, countable choice, trichotomy, or ambient completeness. □

Theorem 9057.3 (Sequential apartness completion boundary Real seal nonescape)**.** *Let* $B = (A, Q, L, S, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$ *packet. Every Real-facing or completion-facing read of B, when the optional seal row E is present, factors*

through the finite route

$$A, Q, L, S, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal receives only the displayed apartness-space source, apartness-completion boundary, located-apartness-completion route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, transport rows, continuation replay, provenance, and local NameCert row. It receives no quotient completion, host equality of completed points, selected limit, countable-choice schedule, trichotomy row, or ambient-completeness coordinate.

Proof. Start with a downstream completion or Real-facing read. Projection of the carrier gives a single ordered route from the apartness-space source row A through the apartness-completion row Q and the located-apartness-completion row L.

The finite observation rows then exhaust the readback path. The window schedule S, regular readback R, and dyadic tolerance ledger D must be displayed before the Real seal row E can be inspected.

Finally, the transport, replay, provenance, and local name rows H, C, P, N carry the same route without adding coordinates. Row exhaustion therefore blocks quotient completions, host equality, selected limits, countable choice, trichotomy, and ambient completeness from entering the Real seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9057.1, Definition 9057.2, and Definition 9057.3 seed the sequential apartness completion boundary over ApartnessSpace, ApartnessCompletion, LocatedApartnessCompletion, StreamName, RegSeqRat, DyadicRatCore, optional Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host equality of completed points, selected limits, countable choice, trichotomy, ambient completeness, bridge checking, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, source-first exactness, sequential readback discipline, componentwise stability, provenance exhaustion, and Real seal nonescape proofs for BEDC.Derived.SequentialApartnessCompletionBoun

Constructive story: A $\mathsf{SequentialApartnessCompletionBoundary}^{\uparrow}$ packet is generated from an ApartnessSpace source row, an ApartnessCompletion boundary row, a LocatedApartnessCompletion row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, an optional RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lean4 (statement) `BEDC.Derived.SequentialApartnessCompletionBoundaryUp`

Chapter 9058

A Concrete Naming Certificate for $\mathsf{RiemannSampler}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RiemannSampler}^{\uparrow}$ packet records the finite tagged sampling data used by constructive Riemann-sum interfaces. It sits beside the dyadic arithmetic core of chapter 291, the terminal real-name seal of chapter 38, the interval-subdivision carrier of chapter 7313, the tagged-subinterval surface of chapter 7449, and the tagged-gauge sum route of chapter 4115. The packet names one finite sample row over a displayed partition; it does not assert integrability, convergence of sums, measure, open-cover compactness, host real functions, or choice.

Definition 9058.1 (Riemann sampler carrier)**.** *A* $\mathsf{RiemannSampler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, R, P, T, F, V, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the dyadic scale, cell-width budget, and endpoint comparison data used by the sample.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *interval row, carrying the named left and right endpoint seals for the interval being sampled.*
3. *P is the* $\mathsf{RiemannPartition}^{\uparrow}$ *partition row, exposing the ordered finite subdivision and its mesh-refinement data.*
4. *T is the* $\mathsf{TaggedPartition}^{\uparrow}$ *tag row, assigning one displayed tag to each visible cell of P together with its tag-in-cell witness.*
5. *F is the function-readback row, recording the displayed evaluation interface opened at the tags rather than a host function space.*
6. *V is the value row, listing the finite real-facing sample values read from F at the tags in T.*
7. *H records componentwise* hsame *transport for mesh, interval, partition, tags, function readback, values, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$D \longrightarrow R \longrightarrow P \longrightarrow T \longrightarrow F \longrightarrow V.$$

9. *G is the* Pkg *provenance over* $\mathsf{RiemannSampler}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannPartition}^{\uparrow}$, $\mathsf{TaggedPartition}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RiemannSampler}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, R, P, T, F, V, H, C, G, N$. It has no coordinate for integrability, convergence of Riemann sums, measure, open-cover compactness, host real functions, quotient equality of partitions, selected tags outside the displayed ledger, or choice.

Lean4 ✓ `BEDC.Derived.RiemannSamplerUp.RiemannSamplerTasteGate single carrier alignment`

Theorem 9058.2 (Riemann sampler NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSampler}^{\uparrow}$ *carrier*

$$S = (D, R, P, T, F, V, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannSampler}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic mesh, Real interval, RiemannPartition row, TaggedPartition row, function-readback row, value row, transport, replay, provenance, and local name; tag-in-mesh exactness requiring every tag in T to be read through the same dyadic mesh row D and partition row P; sample readback stability requiring F and V to move only by the componentwise transport row H; mesh-refinement replay requiring any refinement consumer to reopen D, R, P, T before reading F, V; and provenance exhaustion requiring G, N to name only the displayed finite sampling route.*

Proof. Project the carrier of Definition 9058.1. The accepted packet exposes exactly the rows $D, R, P, T, F, V, H, C, G, N$, so carrier admission is the displayed finite list.

The tag-in-mesh obligation is supplied by D, P, T. The dyadic mesh row records the comparison scale, the partition row exposes the finite cells, and the tagged-partition row attaches each displayed tag to one of those same cells. A tag that is not read through this route has no coordinate in the carrier.

The sample-readback obligation is supplied by F, V, H. The function-readback row is opened only at the displayed tags, the value row lists the resulting finite sample values, and componentwise hsame transport is the only available stability mechanism for moving them.

Mesh-refinement replay is supplied by C. A refinement or Riemann-sum consumer replays the route from D through T before reading F, V, so refinement cannot introduce a detached tag selector or a host mesh limit.

Finally, G and N exhaust provenance and local naming. Since the carrier contains no rows for integrability, convergence of sums, measure, open-cover compactness, host real functions, quotient partition equality, or choice, the listed obligations are exactly the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSampler}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannSampler}^{\uparrow}$ packet is generated from a DyadicRatCore mesh row, Real interval row, RiemannPartition row, TaggedPartition tag row, function-readback row, finite value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannSampler}^{\uparrow}}$, seed)

Scope closed: Definition 9058.1 and Definition 9058.2 seed the finite tagged-sample route over dyadic partitions for downstream Riemann, Darboux, and Henstock-style sum consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannSampler}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close integrability, convergence of sums, measure, open-cover compactness, host real functions, quotient equality of partitions, selected tag choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tag-in-mesh exactness, sample readback stability, mesh-refinement replay, provenance exhaustion, and local naming for BEDC.Derived.RiemannSamplerUp.

Chapter 9059

A Concrete Naming Certificate for $\mathsf{CantorRealCompletionKernel}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ packet records the finite nested-dyadic completion route by which a decreasing interval chain becomes a regular rational name and then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic endpoint ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested dyadic interval carrier of chapter 328. The packet names the completion kernel as displayed BEDC rows; it does not quotient streams, select a host limit, invoke countable choice, or assert ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 9059.1 (Cantor real completion kernel carrier)**.** *A* $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *chain row, listing the retained dyadic intervals and the successor containment ledger.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and diameter row, recording lower endpoints, upper endpoints, midpoint reads, and dyadic diameter bounds for the retained windows.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row through which a consumer requests inspected stages of the chain.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and diameter bounds.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, D, W, Q have been displayed.*
6. *H records componentwise* hsame *transport for interval rows, endpoint ledgers, observation windows, regular readbacks, terminal seals, replay, provenance, and local naming.*
7. *C records the displayed* Cont *replay of the finite route from nested intervals to regular rational readback.*
8. *P records*

$$\mathsf{Pkg}(K) = \mathsf{Pkg}(\mathsf{CantorRealCompletionKernel}^{\uparrow}, \mathsf{NestedDyadicInterval}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow},$$

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorRealCompletionKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, W, Q, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected completed point, host interval intersection, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9059.2 (Cantor kernel nested-containment obligations)**.** *For every accepted* $\mathsf{CantorRealCompletionKernel}$ *carrier*

$$K = (I, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorRealCompletionKernel}^{\uparrow}}$ *surface exposes successor containment only through the* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *row I and the endpoint comparisons recorded in the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D. Every displayed adjacent pair in the retained chain must be read as a finite containment certificate before the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{Real}^{\uparrow}$ *seal can be consumed.*

Proof. Project the carrier of Definition 9059.1. The rows available before any regular readback are the nested interval row I, the dyadic endpoint row D, and the structural transport rows.

Read the successor containment ledger in I. Each adjacent retained pair is paired with lower-endpoint and upper-endpoint comparisons in D, so containment is witnessed by finite dyadic data rather than by a completed intersection point.

Only after those rows are displayed may the consumer inspect W, Q, E. Since the carrier has no quotient, selected limit, host intersection, or ambient-completeness coordinate, the containment obligation is exactly the finite route stated above. □

Theorem 9059.3 (Cantor kernel shrinking-diameter obligations)**.** *For every accepted* $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ *carrier*

$$K = (I, D, W, Q, E, H, C, P, N),$$

each precision request read by the $\mathsf{StreamName}^{\uparrow}$ *row W must be matched by a displayed dyadic diameter bound in D for a retained interval of I. The* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q may read a Cauchy modulus only through those bounds, and no consumer can replace the displayed shrinking ledger by an ambient compactness or completeness principle.*

Proof. Begin with the finite observation row W. It names only the inspected stages of the nested dyadic chain, so a precision request must point back to one retained interval row of I.

The dyadic row D supplies the corresponding diameter bound. That bound is a displayed dyadic comparison attached to the same retained interval, not a host statement about all unseen stages.

The readback row Q consumes the pair (W, D) as regular rational completion data. The packet therefore exposes shrinking diameter through finite windows and dyadic bounds, while the excluded coordinates prevent compactness, selected limits, quotient equality, or ambient completeness from entering the route. □

Theorem 9059.4 (Cantor kernel RegSeqRat-to-Real sealing)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ *carrier must pass through the displayed route*

$$\mathsf{NestedDyadicInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the rows $I, D, W, Q, E, H, C, P, N$. The terminal seal E is not readable until the nested-containment and shrinking-diameter obligations have been displayed through I, D, W, Q.

Proof. Apply Definition 9059.2. It forces the consumer to read the nested interval row I and the dyadic endpoint comparisons in D before treating the chain as a valid decreasing dyadic packet.

Apply Definition 9059.3. It forces every precision request in W to be witnessed by a displayed dyadic diameter bound in D, and it routes the resulting Cauchy data through Q.

Finally read the terminal seal row E through H, C, P, N. These structural rows transport, replay, source, and name only the displayed route. Since the packet contains no selected point, quotient real equality, host limit, countable-choice schedule, or ambient completeness coordinate, the $\mathsf{Real}^{\uparrow}$ seal is reached only by the stated $\mathsf{RegSeqRat}^{\uparrow}$ handoff. □

Falsifiable prediction: An accepted $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ consumer is refuted if it reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the nested dyadic interval row, dyadic endpoint and diameter ledger, finite StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorRealCompletionKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorRealCompletionKernel}^{\uparrow}$ packet is generated from a NestedDyadicInterval chain row, a DyadicRatCore endpoint and diameter ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorRealCompletionKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9059.1, Definition 9059.2, Definition 9059.3, and Definition 9059.4 seed the finite Cantor completion route from nested dyadic intervals through dyadic endpoint and diameter rows, finite windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorRealCompletionKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested containment, shrinking diameter, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorRealCompletionKernelUp.

Chapter 9060

A Concrete Naming Certificate for $\mathsf{PhaseObserverShellRealization}^{\uparrow}$

The $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ packet realizes the shell-like observer-expansion branch of chapter 9463 as finite NameCert data. It sits after the boundary-shadow row of chapter 3813, the visual-boundary row of chapter 4181, the phase-distance observer-expansion route of chapter 6017, the reversible disk-boundary transport of chapter 6267, the reversible phase route of chapter 6920, and the shell packet of chapter 6319. The packet is a concrete realization surface for finite observer-phase shells; it is not a physical observer law, a completed visual boundary, a selected far-end point, or an ambient hyperbolic manifold.

Definition 9060.1 (Phase observer shell realization carrier)**.** *A* $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, V, D, T, R, O, H, C, P, N)$$

with the following displayed rows:

1. B *is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *boundary-shadow row.*
2. V *is the* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *visual-boundary readback row.*
3. D *is the* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *phase-distance comparison row.*
4. T *is the reversible transport row, read through* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *and* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$.
5. R *is the* $\mathsf{HyperbolicObserverPhaseRoute}^{\uparrow}$ *route row binding boundary-shadow, visual-boundary, phase-distance, and reversible-transport data.*
6. O *is the* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ *shell row that keeps the observer-facing packet boundary-facing.*
7. H *records componentwise* hsame *transport for* $B, V, D, T, R, O, C, P, N$.
8. C *records displayed* Cont *replay through boundary shadow, visual boundary, phase distance, transport, route, shell, provenance, and local naming.*
9. P *records*

   $\mathsf{Pkg}(S) = \mathsf{Pkg}(\mathsf{PhaseObserverShellRealization}^{\uparrow}, \mathsf{HyperbolicBoundaryShadow}^{\uparrow}, \mathsf{HyperbolicVisualBoundary}^{\uparrow}, \mathsf{PhaseDis}$

10. N *is the local* $\mathsf{NameCert}_{\mathsf{PhaseObserverShellRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, V, D, T, R, O, H, C, P, N$. It has no coordinate for global hyperbolic space, completed visual boundary, selected observer center, physical observer dynamics, host phase equality, or unlogged transport.

Theorem 9060.2 (Phase observer shell realization NameCert obligations)**.** *For every accepted* $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ *carrier*

$$S = (B, V, D, T, R, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PhaseObserverShellRealization}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission through B, visual-boundary readback through V, phase-distance comparison through D, reversible-transport admission through T, route admission through R, shell admission through O, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9060.1. Its observable shell data are precisely the boundary-shadow, visual-boundary, phase-distance, reversible-transport, route, and shell rows.

Read the rows in dependency order. Boundary shadow and visual boundary make the shell boundary-facing, phase distance supplies the comparison row, reversible transport records how the surface moves, and the observer phase route binds those data before the shell row is accepted.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a completed boundary, selected center, physical dynamics, host phase equality, or unlogged transport, the listed rows exhaust the NameCert surface. □

Theorem 9060.3 (Phase observer shell realization nonescape)**.** *Every observer-expansion read from an accepted* $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ *carrier must factor through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, $\mathsf{PoincareDiskBou}$

It cannot replace those rows by a global hyperbolic space, a completed visual boundary, a selected far-end point, a physical observer law, host phase equality, or an unlogged reversible transport.

Proof. Apply the NameCert obligations of Definition 9060.2. Any accepted read must consume B, V, D, T, R, O before the structural rows are available.

The boundary-shadow and visual-boundary rows make the reading boundary-facing. The phase-distance and reversible-transport rows make the reading phase-facing. The route and shell rows bind those data as observer-expansion evidence.

Finally, H, C, P, N preserve only that finite route. Because the excluded objects are not carrier coordinates, no observer-expansion read can escape to them through the realization packet. □

Falsifiable prediction: An accepted $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ read is refuted if it exports an observer-phase shell without displaying boundary-shadow, visual-boundary, phase-distance, reversible-transport, route, shell, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-visual-boundary, phase-distance-observer-expansion, poincare-disk-boundary-transport, hyperbolic-observer-phase-route: no listed sibling is bijective to $\mathsf{PhaseObserverShellRealization}^{\uparrow}$ because this packet jointly carries boundary-shadow, visual-boundary, phase-distance, reversible-transport, route, shell, replay, provenance, and local naming rows as one realization surface.

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseObserverShellRealization}^{\uparrow}}$ **·**

Constructive story: A PhaseObserverShellRealization$^{\uparrow}$ packet is generated from boundary-shadow, visual-boundary, phase-distance, reversible-transport, observer-phase route, observer-phase shell, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{PhaseObserverShellRealization}^{\uparrow}}$, seed)

Scope closed: Definition 9060.1, Definition 9060.2, Definition 9060.3, the falsifiable prediction, and the independence witness seed the finite observer-phase shell realization over boundary-shadow, visual-boundary, phase-distance, reversible-transport, route, shell, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{PhaseObserverShellRealization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global hyperbolic spaces, completed visual boundaries, selected far-end points, physical observer laws, host phase equality, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow admission, visual-boundary readback, phase-distance comparison, reversible-transport admission, route admission, shell admission, transport, replay, provenance, and local naming for BEDC.Derived.PhaseObserverShellRealizationUp.

Chapter 9061

A Concrete Naming Certificate for FiniteHerbrandDisjunction$^{\uparrow}$

Following Herbrand's theorem as a proof-search interface for first-order logic, the FiniteHerbrandDisjunction$^{\uparrow}$ packet records only the finite disjunction certificate surfaced by a displayed syntactic proof-search route. It sits over the first-order syntax rows of chapter 197, the model-theory boundary of chapter 198, the typed MetaCIC example gallery of chapter 775, the GroundCompiler audit packet of chapter 827, and the local NameCert discipline used by the proof-tree siblings chapter 3756 and chapter 5907. The packet names a finite certificate surface; it does not claim host completeness, infinite proof search, semantic truth in all models, choice, or quotient proof equality.

Definition 9061.1 (Finite Herbrand disjunction carrier)**.** *A* FiniteHerbrandDisjunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (F, D, W, S, Q, R, T, C, P, N)$$

with the following displayed rows:

1. *F is the* FirstOrder$^{\uparrow}$ *formula row containing the prenex source formula, free-variable ledger, quantifier-prefix ledger, and finite term-source alphabet;*
2. *D is the finite disjunct inventory, listing the ground instances admitted into the Herbrand disjunction;*
3. *W is the witness-term and substitution row, attaching each disjunct in D to a displayed tuple of terms and a substitution readback from F;*
4. *S is the syntactic checker row for the displayed disjunction, read through natural-deduction, sequent-calculus, or MetaCIC-facing proof rows only as finite proof-search data;*
5. *Q is the proof-search route row, recording the finite branch order, branch closures, and checker calls that produced D, W, S;*
6. *R records componentwise* hsame *transport for the source formula, disjunct inventory, witness-term rows, checker route, proof-search route, replay row, provenance row, and local naming row;*
7. *T records displayed* Cont *replay from the source formula through Q and then through D, W, S, R, P, N;*
8. *C is the consumer handoff row exposing the certificate to* MetaCICTypedExamplesGallery$^{\uparrow}$ *and* GroundCompilerAuditPacket$^{\uparrow}$ *only after the finite disjunct inventory and checker route are visible;*

9. *P is the* Pkg *provenance over* FiniteHerbrandDisjunction$^{\uparrow}$, FirstOrder$^{\uparrow}$, ModelTheory$^{\uparrow}$, MetaCICTypedExamplesGal GroundCompilerAuditPacket$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{FiniteHerbrandDisjunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, W, S, Q, R, T, C, P, N$. It has no coordinate for semantic completeness, truth in every structure, infinite proof search, selected model construction, choice, quotient proof equality, or a host proof checker outside the displayed rows.

Theorem 9061.2 (Finite Herbrand disjunction NameCert obligations)**.** *For every accepted* FiniteHerbrandDisjuncti *carrier $H = (F, D, W, S, Q, R, T, C, P, N)$, the local* NameCert$_{\text{FiniteHerbrandDisjunction}^{\uparrow}}$ *surface consists exactly of first-order formula exposure through F, finite disjunct inventory exposure through D, witness-term and substitution exposure through W, checker exposure through S, proof-search route exposure through Q, componentwise transport through R, displayed replay through T, downstream certificate handoff through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9061.1. The source syntax is F, the finite Herbrand inventory is D, and the witness-term data are exactly the substitution rows in W.

The checker and search data are also displayed. The row S records the finite checker calls, while Q records the branch order and branch closures that led to the disjunction.

Transport, replay, and consumer exposure are then forced through R, T, C. The package row P records the cited FirstOrder, ModelTheory, MetaCIC gallery, GroundCompiler, and NameCert dependencies, and N names only this displayed packet.

Since the carrier has no coordinate for semantic completeness, truth in every structure, infinite proof search, selected model construction, choice, quotient proof equality, or a host proof checker outside the displayed rows, no local NameCert obligation can export those principles. □

Theorem 9061.3 (Finite Herbrand disjunction non-escape)**.** *Every accepted* FiniteHerbrandDisjunction$^{\uparrow}$ *consumer reads the Herbrand certificate only through the finite route*

$$F \longmapsto Q \longmapsto (D, W, S) \longmapsto (R, T, C, P, N).$$

The consumer may cite the route as a displayed finite disjunction certificate for a first-order proof-search branch, but it may not read the packet as host completeness, infinite proof search, model-theoretic truth, choice, quotient proof equality, or an ambient proof checker.

Proof. Start with the NameCert obligations of Definition 9061.2. They force every accepted read to expose the source formula, the finite disjunct inventory, the witness-term substitutions, the checker row, and the proof-search route before any downstream consumer sees the packet.

The consumer handoff row C can then pass the certificate to MetaCIC or GroundCompiler-facing rows only after the finite inventory and checker route have been replayed through T and transported through R.

The non-escape clauses are the non-coordinates of Definition 9061.1. Because none of host completeness, infinite proof search, model-theoretic truth, choice, quotient proof equality, or an ambient proof checker appears in H, no accepted consumer can project such data from the packet. □

Closure status for NameCert$_{\text{FiniteHerbrandDisjunction}^{\uparrow}}$.

Constructive story: A FiniteHerbrandDisjunction$^{\uparrow}$ packet is generated from a first-order formula row, a finite Herbrand disjunct inventory, witness-term and substitution rows, a checker row, a proof-search route, componentwise hsame transport, displayed Cont replay, downstream certificate handoff, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteHerbrandDisjunction}^{\uparrow}}$, seed)

Scope closed: Definition 9061.1, Definition 9061.2, and Definition 9061.3 seed the finite proof-search certificate over $F, D, W, S, Q, R, T, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteHerbrandDisjunction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host completeness, infinite proof search, model-theoretic truth, choice, quotient proof equality, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked first-order formula exposure, finite disjunct inventory exposure, witness-term substitution exposure, checker exposure, proof-search route exposure, transport, replay, downstream handoff, provenance, and local naming for BEDC.Derived.FiniteHerbrandDisjunctionUp.

Chapter 9062

A Concrete Naming Certificate for $\mathsf{BarInductionCauchyModulus}^{\uparrow}$

Following Brouwer's bar induction practice, Troelstra–van Dalen's fan arguments, and the Bishop–Bridges use of explicit Cauchy moduli, the $\mathsf{BarInductionCauchyModulus}^{\uparrow}$ packet records a finite route from bar evidence to a modulus row for regular real names. The route sits beside chapter 126, chapter 301, chapter 291, chapter 38, chapter 292, chapter 334, and chapter 400; it exposes only finite BHist rows, transport, replay, provenance, and a local $\mathsf{NameCert}$ seal.

Definition 9062.1 (Bar induction Cauchy modulus carrier)**.** *A* $\mathsf{BarInductionCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(B, F, W, S, R, D, E, H, C, P, N)$$

whose rows have the following roles:

1. *B is a decidable bar row for finite prefixes of the displayed branch space.*
2. *F is the fan or finite-cover row certifying that the bar is read through a bounded prefix cover.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite observations used by the bar test.*
4. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose precision request is discharged by the bar.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row produced after the bounded readback.*
7. *H, C, P, N are respectively the* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BarInductionCauchyM}}$ *naming rows for the same finite packet.*

The accepted carrier may be consumed as a $\mathsf{CauchyModulus}^{\uparrow}$ *source only through the displayed bar and fan rows; it contains no countable-choice selector, host infinite branch, quotient stream representative, or ambient Real-completeness witness.*

Theorem 9062.2 (Bar induction Cauchy modulus NameCert obligations)**.** *An accepted* $\mathsf{BarInductionCauchyModulu}$ *carrier satisfies the following five NameCert obligations:*

1. *carrier habitation by the finite rows $B, F, W, S, D, E, H, C, P, N$ of Definition 9062.1;*
2. *bar admissibility, saying that every tolerance request reaches the decidable bar row through a finite prefix accepted by B;*
3. *fan boundedness, saying that the usable prefix is selected only from the finite cover row F;*

4. *readback exactness, saying that the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows read the same bounded prefix and tolerance request;*
5. *transport, replay, provenance, and local naming stability along* H, C, P, N.

Proof. Project the carrier of Definition 9062.1. Its visible rows already name the decidable bar, finite fan cover, StreamName window, RegSeqRat readback, dyadic tolerance, Real seal, and structural naming data.

The bar and fan obligations are read before the analytic rows. The row B supplies the decidable stopping test for the tolerance request, and F supplies the bounded prefix cover through which the stopping test is admitted.

The readback rows then consume the same prefix. The $\mathsf{StreamName}^{\uparrow}$ row gives the finite observation window, $\mathsf{RegSeqRat}^{\uparrow}$ reads the rational data on that window, $\mathsf{DyadicRatCore}^{\uparrow}$ records the tolerance, and $\mathsf{Real}^{\uparrow}$ seals only the resulting bounded readback.

Finally H, C, P, N carry the same packet through transport, replay, provenance, and local naming. No obligation contains a selected infinite branch, a countable-choice function, quotient stream equality, or ambient completeness. □

Theorem 9062.3 (Bar induction Cauchy modulus fan handoff)**.** *Every* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BarInductionCauchyModulus}^{\uparrow}$ *packet reads the modulus only through the finite route*

$$\mathsf{FanTheorem}^{\uparrow}, \quad \mathsf{BaireSpace}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The consumer must first inspect the decidable bar row B *and the fan cover row* F*, then the StreamName window* W*, RegSeqRat readback* S*, dyadic tolerance* D*, and terminal Real seal* E*. The route exports no countable choice, host infinite branch selection, quotient streams, or ambient Real completeness.*

Proof. Start with the obligation surface of Definition 9062.2. It places the decidable bar and finite fan cover before the analytic readback rows.

The $\mathsf{CauchyModulus}^{\uparrow}$ read is therefore a bounded-prefix read. A tolerance request reaches D only after B admits a finite prefix and F bounds the usable cover, so the modulus row cannot be read as a host function selected from infinitely many branches.

The $\mathsf{Real}^{\uparrow}$ read is also bounded. The terminal seal E is exposed only after the $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback have been matched to the same dyadic tolerance row, so quotient stream representatives and ambient Real completeness are not available to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarInductionCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BarInductionCauchyModulus}^{\uparrow}$ packet is generated by reading a decidable bar through a finite fan cover, then routing the bounded StreamName, RegSeqRat, DyadicRatCore, and Real rows into a Cauchy-modulus handoff.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BarInductionCauchyModulus}^{\uparrow}}$, seed)

Scope closed: The seed surface records the finite bar-to-fan-to-Cauchy-modulus route for regular Real names, with only the carrier, obligation, and handoff rows displayed.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BarInductionCauchyModulus}^{\uparrow}}$, unformalized)

Lean4 (statement) `BEDC.Derived.BarInductionCauchyModulusUp`

Bridge status: none

Not claimed: This chapter does not claim a checked Lean NameCert, public Real completeness, countable choice, selected infinite branches, quotient-stream equality, or a standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, bar admissibility, fan boundedness, readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.BarInductionCauchyModulusUp.

Chapter 9063

A Concrete Naming Certificate for $\mathsf{CofinalStreamReindexing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CofinalStreamReindexing}^{\uparrow}$ packet records the finite route by which a StreamName schedule may be read through a cofinal reindexing ledger before regular rational readback, dyadic tolerance comparison, or Real-facing sealing occurs. It sits after the $\mathsf{StreamName}^{\uparrow}$ naming surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance arithmetic of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal discipline of chapter 38, and the cofinal subnet route of chapter 1592. The packet names cofinal reindexing as finite NameCert data, not as quotient stream equality, a host-selected subsequence, countable choice, host real equality, or ambient completeness.

Definition 9063.1 (Cofinal StreamName reindexing carrier)**.** *A* $\mathsf{CofinalStreamReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, J, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite observation windows are to be reindexed.*
2. *J is the cofinal reindex ledger. It records finite dominance checks from each accepted target window back to a displayed source window of S.*
3. *W is the finite-window row obtained after replaying the source row and the cofinal ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed target windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the source and reindexed readbacks.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, J, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the source stream, cofinal ledger, target windows, readback, tolerance ledger, Real seal, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a target precision request through S, J, W, R, D, E, H in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{CofinalStreamReindexing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalStreamReindexing}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host-selected subsequences, countable-choice schedules, host real equality, or ambient completeness.

Theorem 9063.2 (Cofinal StreamName reindexing NameCert obligations)**.** *For every accepted* $\mathsf{CofinalStreamReindexing}^{\uparrow}$ *carrier* $Q = (S, J, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalStreamReindexing}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source StreamName row, cofinal reindex ledger, finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *cofinal coverage, requiring every accepted target window to pass through* J *and to name a displayed source window in* S*;*
3. *finite-window stability, requiring the row* W *to preserve the window requested after the cofinal ledger has been replayed;*
4. *readback preservation, requiring* R *and* D *to be the only rows through which the reindexed window reaches a regular rational comparison;*
5. *seal nonescape, requiring every Real-facing consumer to inspect* S, J, W, R, D *before the seal row* E *is available.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 9063.1. The projection exposes exactly the source StreamName row, cofinal ledger, finite-window row, regular rational readback, dyadic tolerance ledger, Real seal, and structural rows.

Cofinal coverage is the role of J. Each accepted target window must replay a finite dominance check back to the source row S before the target row W can be consumed.

Finite-window stability and readback preservation are then forced by W, R, D. The target window is displayed first, its regular rational readback is exposed next, and the dyadic tolerance ledger is the only comparison row available before the seal.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no quotient stream, host-selected subsequence, countable-choice schedule, host real equality, or ambient-completeness coordinate appears in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 9063.3 (Cofinal StreamName reindexing Real seal nonescape)**.** *Let* $Q = (S, J, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{CofinalStreamReindexing}^{\uparrow}$ *carrier. Every Real-facing consumer of the reindexed stream factors through the finite route*

$$S, J, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus any Real-facing read first consumes the source $\mathsf{StreamName}^{\uparrow}$ *row, the cofinal reindex ledger, the finite target window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The* $\mathsf{Real}^{\uparrow}$ *seal receives no quotient stream, host-selected subsequence, countable-choice schedule, host real equality, or ambient-completeness coordinate.*

Proof. Begin with a Real-facing consumer. Carrier admission gives a seal row E, but it is downstream of the source stream, cofinal ledger, target window, readback, and tolerance rows.

Replay the continuation row C. The request first consumes S and J, then inspects W, reads R, and checks the dyadic tolerance ledger D.

Only after this finite route has been displayed can the packet expose E. Componentwise transport and provenance through H, P, N preserve the same dependency order, so the Real seal cannot be reached by quotient stream equality, host-selected subsequence choice, countable choice, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalStreamReindexing}^{\uparrow}}$ **.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalStreamReindexing}^{\uparrow}}$, seed)

Scope closed: Definition 9063.1, Definition 9063.2, and Definition 9063.3 seed the cofinal StreamName reindexing surface over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofinalStreamReindexing}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, host-selected subsequences, countable choice, host real equality, ambient completeness, bridge checking, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, cofinal coverage, finite-window stability, readback preservation, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CofinalStreamReindexingUp.

Constructive story: A $\mathsf{CofinalStreamReindexing}^{\uparrow}$ packet is generated from a StreamName source row, a cofinal reindex ledger, finite target windows, RegSeqRat readback, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9064

A Concrete Naming Certificate for $\mathsf{FanCauchyModulus}^{\uparrow}$

Following Brouwer and Bishop–Bridges, the $\mathsf{FanCauchyModulus}^{\uparrow}$ packet records the finite fan-bar extraction route by which a decidable bar on binary choice prefixes yields one displayed Cauchy-modulus request before any regular rational readback or real seal is consumed. It sits after the fan-bar surface of chapter 400, the Cauchy-modulus interface of chapter 292, the finite observation window of chapter 126, the regular rational sequence handoff of chapter 301, and the terminal real-name seal of chapter 38. The certificate is a finite NameCert reconstruction of the fan-to-Cauchy-modulus handoff; it does not claim open-cover compactness, choice, quotient streams, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9064.1 (Fan-Cauchy modulus carrier)**.** *A* $\mathsf{FanCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite decidable* $\mathsf{FanTheorem}^{\uparrow}$ *bar row, listing the inspected binary prefixes and the local decision ledger for bar membership.*
2. *K is the prefix-bound row extracted from B, recording one finite depth that meets every displayed branch through the bar.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *request row, translating the prefix bound into a precision-to-window demand for a Cauchy-facing consumer.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the consumer inspects exactly the prefixes and tail windows licensed by K and M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, carrying the regular rational readback obtained after the window and modulus rows have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal route, readable only after B, K, M, W, R have been opened.*
7. *H records componentwise* hsame *transport for bar, bound, modulus, stream window, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$\mathsf{FanTheorem}^{\uparrow} \longrightarrow \mathsf{CauchyModulus}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

9. *P records provenance over* $\mathsf{FanCauchyModulus}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FanCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, M, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, a choice functional, quotient stream equality, a selected infinite branch, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9064.2 (Fan-Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{FanCauchyModulus}^{\uparrow}$ *carrier*

$$F = (B, K, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FanCauchyModulus}^{\uparrow}}$ *surface consists exactly of finite bar admission through B, prefix-bound extraction through K, Cauchy-modulus request formation through M, StreamName window exposure through W, RegSeqRat handoff through R, Real seal routing through E, componentwise transport through H, displayed replay through C, provenance exhaustion through P, and local naming through N.*

Proof. Project the carrier of Definition 9064.1. The accepted packet exposes exactly the rows $B, K, M, W, R, E, H, C, P, N$, so carrier admission is the displayed finite list.

The fan-facing obligation is supplied by B and K. The decidable bar row lists the inspected prefixes, and the bound row records the finite depth used by all downstream requests. No separate branch selector is present.

The Cauchy-facing obligation is supplied by M, W, R. The modulus request names the demanded precision window, the StreamName row opens only that finite observation surface, and the RegSeqRat row reads regular rational data from the same window.

Finally, E, H, C, P, N seal, transport, replay, source, and name the displayed route. Since the carrier contains no row for open-cover compactness, choice, quotient streams, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 9064.3 (Fan-Cauchy finite-bar extraction)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FanCauchyModulus}^{\uparrow}$ *carrier must pass through the displayed finite route*

$$\mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the rows $B, K, M, W, R, E, H, C, P, N$. The prefix bound K is the only source of the Cauchy-modulus request M, and the Real seal row E is not readable until the bar, bound, modulus, window, and regular-readback rows have all been displayed.

Proof. Begin with the finite bar row B. It supplies the decidable prefix ledger read from the fan-theorem sibling, while K records the finite bound extracted from that same ledger.

Read the Cauchy-modulus request M through K. This forces any precision demand to be a displayed finite-window demand rather than a choice functional or a branchwise host argument.

Now open W and R. The StreamName window exposes the inspected prefixes and tail windows, and the RegSeqRat handoff reads regular rational data only from those windows.

The Real seal E is then available through H, C, P, N. These structural rows move, replay, source, and name the same finite route, so no open-cover compactness, quotient stream, selected branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim can enter the extraction. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FanCauchyModulus}^{\uparrow}$ packet is generated from a finite decidable FanTheorem bar row, prefix-bound row, CauchyModulus request row, StreamName window row, RegSeqRat handoff, Real seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9064.1, Definition 9064.2, and Definition 9064.3 seed the finite fan-bar to Cauchy-modulus route consumed by later regular-Cauchy and Real-seal arguments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice, quotient streams, selected infinite branches, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked finite-bar admission, prefix-bound extraction, Cauchy-modulus request formation, StreamName window exposure, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FanCauchyModulusUp.

Chapter 9065

A Concrete Naming Certificate for $\mathsf{LocatedIntervalSampling}^{\uparrow}$

Following Bishop and Bridges on located intervals and constructive real approximation, the $\mathsf{LocatedIntervalSampling}$ packet records the finite sampling route by which a located closed interval is inspected through density-net cells before any completion-facing Real seal is read. It sits beside the finite Cauchy density-net route of chapter 6003, the dyadic interval-cell route of chapter 5966, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with displayed finite interval samples; it does not assert classical compactness, selected limits, quotient representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9065.1 (Located interval sampling carrier)**.** *A* $\mathsf{LocatedIntervalSampling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, N, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the located closed interval row, exposing named endpoint seals and a located endpoint comparison before any sample is read.*
2. *N is the* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *row naming the finite density-net handoff for the same interval source.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the observed sample windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the windows in W.*
5. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after I, N, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval, density-net, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through the route*

$$I \longrightarrow N \longrightarrow W \longrightarrow R \longrightarrow E.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalSampling}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSampling}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, N, W, R, E, H, C, P, L$. It has no coordinate for a quotient representative of the interval, a selected completed sample, trichotomy on endpoints, countable choice over windows, classical compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9065.2 (Located interval sampling NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalSampling}^{\uparrow}$ *carrier*

$$S = (I, N, W, R, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalSampling}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed located interval, finite Cauchy density-net, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-cell exposure, requiring the sample cells to be read through the* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *surface before a density-net row is consumed;*
3. *density-net handoff, requiring N to expose the finite Cauchy density-net route of chapter 6003 for the same located interval row I;*
4. *readback-before-seal discipline, requiring W and R to be displayed before the* $\mathsf{Real}^{\uparrow}$ *seal row E is available;*
5. *refusal of quotient representatives, selected limits, trichotomy, countable choice, and ambient Real completeness as exportable rows of the packet.*

No local obligation is stored outside $I, N, W, R, E, H, C, P, L$.

Proof. Carrier admission is projection from Definition 9065.1. The accepted packet exposes exactly the located interval, density-net, window, readback, Real seal, transport, replay, provenance, and local naming rows.

The interval-cell obligation is supplied by the dyadic interval-net sibling. A sample cell may be consumed only after the displayed interval source has been matched to a finite $\mathsf{DyadicIntervalNet}^{\uparrow}$ cell route, so the packet never treats an interval as an untracked host subset.

The density-net handoff is supplied by N. It names the finite Cauchy density-net route over the same interval row and does not select a completed sample or assert compactness outside the displayed finite rows.

Finally read W, R, E, H, C, P, L. The StreamName windows and RegSeqRat readback must precede the Real seal, while transport, replay, provenance, and local naming preserve only the displayed route. Hence quotient representatives, selected limits, trichotomy, countable choice, and ambient Real completeness have no carrier coordinate from which to escape. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalSampling}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalSampling}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9065.1 and Definition 9065.2 seed the located interval sampling route over located interval, DyadicIntervalNet, FiniteCauchyDensityNet, StreamName, RegSeqRat, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalSampling}^{\uparrow}}, \mathsf{encodedDef})$

Bridge status: none

Not claimed: This chapter does not close classical compactness, quotient interval representatives, selected limits, trichotomy, countable choice, ambient RealUp completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-cell exposure, density-net handoff, StreamName and RegSeqRat readback before Real sealing, provenance exhaustion, and local naming for BEDC.Derived.LocatedIntervalSamplin

Constructive story: A $\mathsf{LocatedIntervalSampling}^{\uparrow}$ packet is generated from a located closed interval row, a finite Cauchy density-net row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lean4 (definition) `BEDC.Derived.LocatedIntervalSamplingUp`

Chapter 9066

A Concrete Naming Certificate for $\mathsf{InfoGenerationTrace}^{\uparrow}$

The $\mathsf{InfoGenerationTrace}^{\uparrow}$ packet realizes the constructive information generation vision of section 9414.11 as finite NameCert data over visible record rows, residue rows, compatible refinement rows, replay, and provenance. It sits beside the finite tower packet of chapter 1128, the reflection-gap audit route of chapter 7705, and the classifier boundary surface of chapter 595. Those siblings supply tower source rows, audit and gap rows, and visible classifier-boundary discipline; this packet binds them into one generated-information trace before any consumer treats a visible record as the whole source.

Definition 9066.1 (InfoGenerationTrace carrier)**.** *An* $\mathsf{InfoGenerationTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, V, L, Q, R, J, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-carrier row, naming the displayed $\mathcal{I}_n$ layer or tower source from which the trace is read.*
2. *V is the visible-record row, recording the generated record exposed to the current consumer.*
3. *L is the ledger row, carrying provenance, gap, audit, and classifier-boundary entries that license V.*
4. *Q is the residue row, recording the remaining source fiber over V rather than collapsing it into the visible record.*
5. *R is the compatible refinement row, naming any readback from a later visible surface to the present one.*
6. *J is the visible-refinement join row, displaying how V, Q, R are consumed together by a downstream generated-information read.*
7. *H records componentwise* hsame *transport for source, visible record, ledger, residue, refinement, join, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$S \longrightarrow V \longrightarrow L \longrightarrow Q \longrightarrow R \longrightarrow J.$$

9. *P records provenance over* $\mathsf{InfoGenerationTrace}^{\uparrow}$, $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$, $\mathsf{ReflectionGapAudit}^{\uparrow}$, $\mathsf{ClassifierBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{InfoGenerationTrace}^{\uparrow}}$ *row.*

The classifier compares accepted traces only through $S, V, L, Q, R, J, H, C, P, N$. It has no coordinate for a bare information object, hidden source equality, final physical truth, tower-external information, host semantic completeness, or an unrecorded generation step.

Theorem 9066.2 (InfoGenerationTrace NameCert obligations)**.** *For every accepted* $\mathsf{InfoGenerationTrace}^{\uparrow}$ *carrier*

$$T = (S, V, L, Q, R, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InfoGenerationTrace}^{\uparrow}}$ *surface consists exactly of source admission through S, visible-record exposure through V, audit and gap licensing through L, residue display through Q, compatible refinement through R, visible-refinement joining through J, componentwise transport through H, displayed replay through C, provenance exhaustion through P, and local naming through N.*

Proof. Project the carrier of Definition 9066.1. The accepted packet exposes exactly the rows $S, V, L, Q, R, J, H, C, P, N$, so source admission and visible-record exposure are finite displayed data.

The licensing obligation is supplied by L. It records the provenance, gap, audit, and classifier-boundary entries through which the visible row is admitted, so no bare information object can enter the trace.

The non-exhaustion obligation is supplied by Q and R. The residue row keeps the source fiber visible, while the refinement row records any compatible readback from a later surface to the current visible record.

Finally, J, H, C, P, N join, transport, replay, source, and name the same route. Since the carrier contains no row for hidden source equality, final physical truth, tower-external information, host semantic completeness, or unrecorded generation, the listed obligations exhaust the local NameCert surface. □

Theorem 9066.3 (InfoGenerationTrace visible residue)**.** *Every accepted generated-information read through an* $\mathsf{InfoGenerationTrace}^{\uparrow}$ *carrier must expose the visible row V, residue row Q, refinement row R, replay row C, provenance row P, and local NameCert row N before it exports the generated record. A consumer that reads V while omitting Q or R has read only a visible record, not the full generated-information trace.*

Proof. Start with the source and visible rows S, V. They identify the layer source and the record exposed to the consumer, but they do not contain the residual source family by themselves.

Read the ledger and residue rows L, Q. The ledger licenses the visible row, and the residue row records what remains over that visible record. This is the finite trace counterpart of the residue family in Definition 9414.220.

Read the refinement row R and the join row J. They display how later compatible layers may return to the present visible record without erasing the residue data.

The replay, provenance, and local naming rows C, P, N then export the generated-information trace. Since the carrier has no coordinate for bare objects, hidden source equality, tower-external information, or final truth, omitting Q or R leaves the consumer with an incomplete visible read. □

Falsifiable prediction: Any accepted generated-information read through $\mathsf{InfoGenerationTrace}^{\uparrow}$ must expose visible record, residue, compatible refinement, replay, provenance, and local NameCert rows before exporting the generated record.

Independence witness: constructive-information-generation, belief, generation-trace-adjacent, realityconstrainedtowerpacket, reflection-gap-audit, classifierboundary: $\mathsf{InfoGenerationTrace}^{\uparrow}$ is not bijective to any listed sibling because it binds one visible record to residue and compatible-refinement rows together with audit, replay, provenance, and local NameCert data.

The Lean follow-up file `lean4/BEDC/Derived/InfoGenerationTraceUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{InfoGenerationTrace}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{InfoGenerationTrace}^{\uparrow}$ instances. These instances must preserve the rows $S, V, L, Q, R, J, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{InfoGenerationTrace}^{\uparrow}}$.

Constructive story: An $\mathsf{InfoGenerationTrace}^{\uparrow}$ packet is generated from a source-carrier row, visible-record row, audit and gap ledger row, residue row, compatible-refinement row, visible-refinement join row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InfoGenerationTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9066.1, Definition 9066.2, and Definition 9066.3 seed the finite trace packet realizing constructive information generation as visible rows plus residue and refinement data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InfoGenerationTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close final physical truth, hidden source equality, tower-external information, host semantic completeness, unrecorded generation steps, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, visible-record exposure, audit and gap licensing, residue display, compatible refinement, visible-refinement joining, transport, replay, provenance, and local naming for BEDC.Derived.InfoGenerationTraceUp.

Chapter 9067

A Concrete Naming Certificate for $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, the $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$ packet records the finite product-completion route by which two regular rational Cauchy names expose a product row before any $\mathsf{Real}^{\uparrow}$ multiplication consumer reads a completed seal. It sits after the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Cauchy product source of chapter 464, the regular product route of chapter 461, and the product-window route of chapter 2238. The packet names the completion-facing product seal as finite BEDC rows; it does not quotient streams, select product representatives, assert host real equality, use countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9067.1 (CauchyProductCompletionSeal carrier)**.** *A* $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (L, R, P, B, Q, T, S, H, C, K, N)$$

with the following displayed rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows admitted for product formation.*
2. *P is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product-source row.*
3. *B is the finite product-budget row exposing which tail windows and dyadic tolerances are consumed.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-tolerance row.*
5. *T is the dyadic tail-stability ledger for the product row.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *completion-seal handoff row.*
7. *H records componentwise* hsame *transport for the source, product, budget, tolerance, stability, seal, provenance, replay, and naming rows.*
8. *C records displayed* Cont *replay from the source rows through product budget, dyadic tolerance, tail stability, and seal handoff.*
9. *K is the* Pkg *provenance over* $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionSeal}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient streams, selected representatives, host real equality, countable choice, or ambient completion.

Theorem 9067.2 (CauchyProductCompletionSeal classifier)**.** *Two accepted* $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$ *carriers are classified by the displayed rows $L, R, P, B, Q, T, S, H, C, K, N$. A classifier comparison may identify the product-completion seal only when the source regular names, product-source row, product budget, dyadic tolerance, tail-stability ledger, seal handoff, transport, replay, provenance, and local naming rows are all compared through the same finite packet.*

Proof. Project the carrier of Definition 9067.1. The classifier-visible data are exactly the listed rows.

The structural rows H, C, K, N preserve and name the same source, product, budget, tolerance, stability, and seal coordinates. Since the carrier has no quotient-stream or host-real-equality coordinate, no classifier comparison can use one. □

Theorem 9067.3 (CauchyProductCompletionSeal source admission)**.** *Every accepted* CauchyProductCompletionSe *carrier admits its product source only through the two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rows L, R and the* $\mathsf{CauchyProduct}^{\uparrow}$ *row P.*

Proof. Read L, R, P from Definition 9067.1. These are the only rows that can serve as product-source data before the budget row is exposed.

The classifier theorem Definition 9067.2 keeps all source comparisons inside the same finite packet, so no unrecorded product source can enter the seal route. □

Theorem 9067.4 (CauchyProductCompletionSeal product-budget exposure)**.** *Every accepted* CauchyProductComp *carrier exposes the product-budget row B before a completion-facing consumer can read the dyadic tolerance row Q, tail-stability ledger T, or seal handoff S.*

Proof. The carrier definition places B between the product-source row P and the downstream tolerance, stability, and seal rows. The replay row C records that same order as displayed Cont data.

By Definition 9067.2, downstream comparisons preserve B as part of the finite packet. Thus a completion consumer cannot read Q, T, S while bypassing the budget exposure row. □

Theorem 9067.5 (CauchyProductCompletionSeal dyadic tail stability)**.** *Every accepted* CauchyProductCompletion *carrier reads dyadic tail stability through the tolerance row Q and the stability ledger T after the product budget has been exposed.*

Proof. Use Definition 9067.1. The dyadic tolerance is the row Q, and the product-tail stability evidence is the ledger T.

Apply product-budget exposure from Definition 9067.4. It places B before Q and T, so the dyadic tail-stability read is a budgeted finite read rather than a host-level convergence claim. □

Theorem 9067.6 (CauchyProductCompletionSeal completion-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$ *carrier reaches the seal row S only after reading $L, R, P, B, Q, T, H, C, K, N$.*

Proof. Start with source admission and product-budget exposure. They place L, R, P before B, and B before the dyadic tolerance and tail-stability rows.

Use dyadic tail stability above. It fixes Q, T as the finite stability route. The structural rows H, C, K, N transport, replay, source, and name that same route before the seal row S is consumed. □

Theorem 9067.7 (CauchyProductCompletionSeal quotient and choice refusal)**.** *No accepted* CauchyProductCompl *carrier exports quotient streams, selected representatives, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 9067.1. Its rows are exactly source, product, budget, dyadic tolerance, tail stability, seal, transport, replay, provenance, and local naming rows.

The completion-seal handoff of Definition 9067.6 routes every completion-facing read through those rows. The refused objects are not rows of the carrier and are not introduced by H, C, K, N. □

Lean4 (definition) `BEDC.Derived.CauchyProductCompletionSealUp`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionSeal}^{\uparrow}$ packet is generated from left and right RegSeqRat source rows, a CauchyProduct source row, product-budget exposure, DyadicRatCore tolerance, tail-stability ledger, Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9067.1, Definition 9067.2, and the five obligation theorems seed the finite product-completion seal route over regular rational sources, dyadic product budgets, tail stability, and Real-facing handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionSeal}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.CauchyProductCompletionSealUp`

Bridge status: none

Not claimed: This chapter does not close Real multiplication, quotient stream equality, selected product representatives, countable choice, host real equality, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, product-budget exposure, dyadic tail stability, completion-seal handoff, refusal, and local naming for BEDC.Derived.CauchyProductCompletionSealUp.

Chapter 9068

A Concrete Naming Certificate for $\mathsf{CarlesonMeasure}^{\uparrow}$

Following Carleson and the Hardy-space boundary embedding tradition, the $\mathsf{CarlesonMeasure}^{\uparrow}$ packet records the finite measure-window surface needed before a Hardy-boundary consumer treats a displayed measure row as Carleson-facing data. It sits after the Hardy-space carrier of chapter 5108, the box geometry of chapter 8988, the event-value substrate of chapter 92, the dyadic tolerance core of chapter 291, and the terminal scalar seal of chapter 38. The packet names only displayed boxes, local measure budgets, Hardy boundary rows, dyadic tolerances, Real scalar seals, transport, replay, provenance, and local naming; it does not assert selected boundary limits, completed measure theory, quotient function equality, global embedding constants, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9068.1 (Carleson measure carrier)**.** *A* $\mathsf{CarlesonMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, H, U, D, E, S, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CarlesonBox}^{\uparrow}$ *window row from chapter 8988. It exposes the boundary arc, radial dyadic window, tent footprint, and finite box budget inspected by the measure request.*
2. *H is the* $\mathsf{HardySpace}^{\uparrow}$ *handoff row from chapter 5108. It names the Hardy-boundary source that may consume the same box window.*
3. *U is the* $\mathsf{Measure}^{\uparrow}$ *local mass row from chapter 92. It records only the displayed mass assigned to the box window and the finite event-value route supporting that mass.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row from chapter 291. It stores the dyadic scale, arc length, radial depth, and comparison tolerance used by the window estimate.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row from chapter 38. It seals the displayed mass, dyadic window size, and local comparison value after B, H, U, D are visible.*
6. *S records componentwise* hsame *transport for B, H, U, D, E.*
7. *C records displayed* Cont *replay from a boundary embedding request through the box, Hardy handoff, measure mass, dyadic tolerance, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CarlesonMeasure}^{\uparrow}$, $\mathsf{CarlesonBox}^{\uparrow}$, $\mathsf{HardySpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CarlesonMeasure}^{\uparrow}}$ *row.*
The classifier compares carriers only through $B, H, U, D, E, S, C, P, N$. It has no coordinate for a selected boundary limit, a completed measure algebra, quotient equality of Hardy functions, a global embedding theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9068.2 (Carleson measure NameCert obligations)**.** *For every accepted* $\mathsf{CarlesonMeasure}^{\uparrow}$ *carrier $M = (B, H, U, D, E, S, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CarlesonMeasure}^{\uparrow}}$ *surface consists exactly of Carleson-box visibility, Hardy-space handoff visibility, displayed measure-mass visibility, dyadic tolerance visibility, Real scalar sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. A boundary embedding consumer may read the measure row only after B, H, U, D, E are all public in the same packet.*

Proof. Project the carrier of Definition 9068.1. The displayed tuple exposes exactly the box, Hardy handoff, measure mass, dyadic tolerance, Real seal, transport, replay, provenance, and local naming row.

Read the window rows first. The $\mathsf{CarlesonBox}^{\uparrow}$ row B names the boundary arc and radial window, while the $\mathsf{HardySpace}^{\uparrow}$ row H names the Hardy-boundary source allowed to inspect that same window.

The measure and scalar rows are downstream of the window. The row U supplies only the displayed $\mathsf{Measure}^{\uparrow}$ mass for the box, D supplies the dyadic scale and comparison tolerance, and E seals the resulting scalar reads through $\mathsf{Real}^{\uparrow}$.

Finally the structural rows preserve the displayed route. Since S, C, P, N add only transport, replay, provenance, and naming, the packet has no coordinate for selected boundary limits, completed measure algebras, quotient function equality, global embedding theorems, countable choice, or ambient completeness. □

Theorem 9068.3 (Carleson measure window-embedding handoff)**.** *Every Hardy-boundary embedding request that consumes an accepted* $\mathsf{CarlesonMeasure}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CarlesonBox}^{\uparrow} \longmapsto \mathsf{Measure}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{HardySpace}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In carrier notation, the consumer reads the box window B, local mass row U, dyadic tolerance row D, Hardy handoff H, Real seal E, and the structural rows S, C, P, N. The route is a finite Carleson-window measure handoff, not a proof of the host Carleson embedding theorem, a selected boundary-limit principle, a completed measure construction, a quotient-function equality claim, or an ambient completeness claim.

Proof. Start with the box window. The row B is already a finite $\mathsf{CarlesonBox}^{\uparrow}$ packet, so the measure request begins with displayed boundary geometry rather than an ambient measurable subset.

Next read the local mass and tolerance rows. The $\mathsf{Measure}^{\uparrow}$ row U provides only the mass assigned to that displayed box, and D records the dyadic scale and tolerance used by the comparison.

The Hardy-facing read then passes through H and E. The Hardy handoff is visible only after the box, mass, and tolerance rows have been displayed, and the Real seal records the scalar endpoint of the same route.

Transport, replay, provenance, and local naming preserve the packet. Because no carrier row stores a selected boundary limit, completed measure theory, quotient equality of Hardy functions, global embedding theorem, or ambient completeness principle, no boundary embedding consumer can read those endpoints from the packet. □

Falsifiable prediction: An accepted $\mathsf{CarlesonMeasure}^{\uparrow}$ packet is refuted if a Hardy-boundary embedding consumer reads a measure mass without displaying the Carleson box, Hardy handoff, local MeasureUp mass, dyadic tolerance, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: hardy-space, carleson-box, measure, dyadicratcore, real: $\mathsf{CarlesonMeasure}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores box geometry, Hardy handoff, local measure mass, dyadic tolerance, Real scalar sealing, and structural support in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CarlesonMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{CarlesonMeasure}^{\uparrow}$ packet is generated from a CarlesonBox window, HardySpace handoff, MeasureUp local mass row, DyadicRatCore tolerance row, RealUp scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CarlesonMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9068.1, Definition 9068.2, and Definition 9068.3 seed the finite Hardy-boundary measure packet over box windows, local masses, dyadic tolerances, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CarlesonMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected boundary limits, completed measure theory, quotient function equality, the host Carleson embedding theorem, global embedding constants, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification requires a Lean carrier and checked NameCert obligations for BEDC.Derived.CarlesonMeasureUp.

Chapter 9069

Cauchy Uniform Continuation NameCert Construction

Bishop-style constructive analysis treats the uniformly continuous extension theorem for completions as a modulus theorem: a map defined on dense rational or dyadic Cauchy data extends only when the same explicit modulus transports regular tails to regular tails. The $\mathsf{CauchyUniformContinuation}^{\uparrow}$ packet records that theorem as finite BEDC NameCert data. It sits after the regular rational names of chapter 301, the terminal real-name seal of chapter 38, and the uniformly continuous completion-extension surface of chapter 3699. The packet names continuation across regular Cauchy completion without quotient completion classes, selected limits, host function extensionality, countable choice, or ambient completeness outside the displayed $\mathsf{Real}^{\uparrow}$ seal.

Definition 9069.1 (Cauchy uniform continuation carrier)**.** *A* $\mathsf{CauchyUniformContinuation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, R, S, E, M, G, T, C, P, N)$$

with the following displayed rows:

1. *D is the dense dyadic or rational source row, recording the finite source observations on which the map is already defined.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy name row produced from the source observations and the requested target precision.*
3. *S is the finite stream-window row that records which source and target Cauchy-name prefixes are inspected.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *completion-name seal, exposed only after D, R, S have been displayed.*
5. *M is the explicit uniform-continuity modulus row, translating each target precision request into a source-window request.*
6. *G is the graph row for the source map on dense dyadic or regular-Cauchy representatives, scoped by D, R, S, M.*
7. *T records componentwise* hsame *transport for source observations, regular-Cauchy names, stream windows, Real seals, moduli, graph rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a continuation request through D, R, S, E, M, G, T.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyUniformContinuation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformContinuation}^{\uparrow}}$ *row.*

The carrier stores no quotient class of Cauchy names, selected completed value, host function equality, countable-choice selector, trichotomy branch, or ambient completeness principle.

Theorem 9069.2 (Cauchy uniform continuation NameCert obligations)**.** *For an accepted* CauchyUniformContinuati *carrier $U = (D, R, S, E, M, G, T, C, P, N)$, the local NameCert surface consists of exactly the following finite obligations:*

1. *carrier admission, requiring every continuation read to display $D, R, S, E, M, G, T, C, P, N$ before the local name row is public;*
2. *classifier locality, comparing accepted packets only through their displayed dense-source, regular-Cauchy, window, Real-seal, modulus, graph, transport, replay, provenance, and naming rows;*
3. *stability, requiring* hsame *transport to preserve the modulus-controlled tail request and the graph row on the displayed regular-Cauchy windows;*
4. *exactness, requiring the target* $\mathsf{Real}^{\uparrow}$ *seal E to be reached only through the source graph G and the modulus row M applied to D, R, S;*
5. *ledger nonescape, requiring P, N to export no quotient completion class, selected limit, host function equality, countable choice, trichotomy, or ambient completeness principle.*

Proof. Project the carrier of Definition 9069.1. Carrier admission and classifier locality are the visible tuple inventory: a continuation request has no analytic row other than $D, R, S, E, M, G, T, C, P, N$.

For stability, use the displayed hsame row T. It transports dense-source observations, regular-Cauchy names, windows, Real seals, moduli, and graph rows componentwise, so the same target precision request is replayed through the same source-window bound.

For exactness, the modulus row M is the only route from target precision to a source-window request, and the graph row G is scoped by that request. The terminal seal E is therefore downstream of D, R, S, M, G and cannot be read as a detached completed value.

Finally, replay and provenance are exhausted by C, P, N. Since the carrier contains no row for quotient completion classes, selected limits, host function equality, countable choice, trichotomy, or ambient completeness, the local certificate cannot export those objects. □

Theorem 9069.3 (Modulus composition for Cauchy uniform continuations)**.** *Let $U = (D, R, S, E, M, G, T, C, P, N$ and $V = (D', R', S', E', M', G', T', C', P', N')$ be accepted* $\mathsf{CauchyUniformContinuation}^{\uparrow}$ *packets whose target seal of U is the displayed dense-source row consumed by V. Then the composite continuation is controlled by the finite modulus route M followed by M': a target precision request for V first opens the M' source window for V, then opens the M source window for U, and the resulting continuation reaches the final* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed rows of U and V.*

Proof. Start with the target precision request for the second packet V. Its NameCert obligations route that request through the modulus row M' and the displayed source-window, regular-Cauchy, graph, and Real-seal rows of V.

The source row consumed by V is the target seal exported by U. Apply the obligations for U to that intermediate request. The row M opens the required source window for U, and G supplies the dense-source graph only on that window.

Compose the two displayed replay rows C and C'. The combined replay first follows M', then M, and then reads the two graph rows in order before the final seal is public.

The provenance rows P, P' and local names N, N' record only this finite route. Because neither carrier contains quotient completion classes, selected limits, host function equality, countable

choice, trichotomy, or ambient completeness, the composite continuation exports none of those surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyUniformContinuation}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyUniformContinuation}\uparrow}, \mathsf{seed})$

Scope closed: Definition 9069.1, Definition 9069.2, and Definition 9069.3 seed the finite modulus-controlled continuation route across regular Cauchy completion names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyUniformContinuation}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completed values, host function extensionality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier locality, modulus stability, continuation exactness, composition exactness, replay, provenance, and local naming for BEDC.Derived.CauchyUniformContinuationUp.

Constructive story: A $\mathsf{CauchyUniformContinuation}^{\uparrow}$ packet is generated from dense dyadic or rational source rows, RegSeqRat regular-Cauchy names, finite stream windows, a RealUp seal, explicit uniform-continuity moduli, source graph rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9070

A Concrete Naming Certificate for $\mathsf{KKMLemma}^{\uparrow}$

Following Knaster, Kuratowski, and Mazurkiewicz, the $\mathsf{KKMLemma}^{\uparrow}$ packet records the finite simplex-cover intersection datum behind the KKM lemma as a NameCert carrier. It sits downstream of the finite-cover surface of chapter 6671, the compact metric route of chapter 129, the convex-set carrier of chapter 208, the polytope face ledger of chapter 209, and the finite NameCert cover surfaces used by chapter 8591 and chapter 1472. The packet names only finite face-cover rows, convex-hull membership rows, compact-metric support, intersection-window data, transport, replay, provenance, and local naming; it does not select a host intersection point or import arbitrary open-cover compactness.

Definition 9070.1 (KKM lemma carrier)**.** *A* $\mathsf{KKMLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, V, C, P_0, M, Q, I, H, R, G, N)$$

with the following displayed rows:

1. *S is the finite simplex inventory row.*
2. *F is the face ledger naming every inspected face and its spanning vertex list.*
3. *V is the vertex-indexed cover row inherited from the* $\mathsf{KKMFiniteCover}^{\uparrow}$ *finite-cover surface.*
4. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *row certifying convex hull membership for the simplex and each displayed face.*
5. *P_0 is the* $\mathsf{Polytope}^{\uparrow}$ *face-incidence row.*
6. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite compactness support row for the inspected simplex carrier.*
7. *Q is the finite cover-qualification row, asserting that each displayed face is covered by the cells indexed by its own vertices.*
8. *I is the intersection-window row naming the finite subfamily whose common intersection is forced by the face-cover inspection.*
9. *H records componentwise* hsame *transport for the simplex, face, cover, convexity, compactness, qualification, and intersection-window rows.*
10. *R records displayed* Cont *replay through the same finite KKM route.*

11. *G is the* Pkg *provenance over* $\mathsf{KKMLemma}^{\uparrow}$, $\mathsf{KKMFiniteCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KKMLemma}^{\uparrow}}$ *row.*

The carrier exports a finite intersection certificate. It contains no coordinate for an externally selected intersection point, quotient equality of closed sets, arbitrary open-cover compactness, countable choice, or ambient completeness.

Theorem 9070.2 (KKM lemma NameCert obligations)**.** *For every accepted* $\mathsf{KKMLemma}^{\uparrow}$ *carrier* $K = (S, F, V, C, P_0, M, Q, I, H, R, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KKMLemma}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *simplex and face admission through S, F, P_0, binding every inspected face to a displayed finite vertex list;*
2. *cover-row admission through V, Q, requiring each face to be covered only by cells indexed by vertices of that face;*
3. *convex and compact support through C, M, exposing convex-hull membership and compact-metric finite support before any intersection-window read;*
4. *intersection-window handoff through I, naming the finite subfamily whose common meet is forced by the face-cover inspection;*
5. *transport, replay, provenance, and local naming through H, R, G, N, with no host point selector, quotient cover equality, arbitrary open-cover compactness, countable choice, or ambient completeness row.*

No local KKM lemma obligation is stored outside these rows.

Proof. Project the carrier row by row. The simplex inventory S, face ledger F, and polytope row P_0 determine the only faces that can be inspected.

Next read the finite-cover route. The cover row V and qualification row Q say that each displayed face is covered by the cells indexed by its own vertices, matching the finite KKM cover surface of chapter 6671.

The convex and compact support rows then bind the geometric read. The $\mathsf{ConvexSet}^{\uparrow}$ row C supplies face-hull membership, and the $\mathsf{CompactMetric}^{\uparrow}$ row M supplies only the displayed finite compactness support.

Finally the intersection-window row I is read after those face-cover checks. Componentwise transport, replay, provenance, and the local NameCert row repack the same finite data, so the obligation surface cannot export a selected host point, quotient cover equality, arbitrary open-cover compactness, countable choice, or ambient completeness. □

Theorem 9070.3 (KKM lemma finite simplex-cover handoff)**.** *Every downstream constructive fixed-point, finite intersection, or compact-convex cover consumer reading an accepted* $\mathsf{KKMLemma}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{KKMFiniteCover}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ConvexSet}^{\uparrow},\quad \mathsf{Polytope}^{\uparrow} \longmapsto S, F, V, C, P_0, M, Q, I, H, R, G, N.$$

The consumer receives a displayed finite simplex-cover intersection window, not a host-selected intersection point, arbitrary compactness theorem, quotient cover equality, countable-choice schedule, or ambient completeness assertion.

Proof. Start from the obligation surface of Definition 9070.2. It orders simplex and face admission before cover qualification, and it keeps the convex and compact support rows visible before I can be inspected.

The finite simplex-face inspection is the only handoff mechanism. For each displayed face, the packet checks the listed vertices, reads the cover cells indexed by those vertices, and records the corresponding intersection-window contribution inside the same BHist packet.

Structural rows preserve that order. The hsame transport row H, Cont replay row R, package row G, and local NameCert row N keep the handoff attached to $\mathsf{KKMFiniteCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, and $\mathsf{Polytope}^{\uparrow}$ rather than to any external compactness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KKMLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{KKMLemma}^{\uparrow}$ packet is generated from a finite simplex inventory, face ledger, vertex-indexed KKM cover row, convex-set support, polytope face incidence, compact-metric support, finite cover qualification, intersection-window data, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KKMLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9070.1, Definition 9070.2, and Definition 9070.3 seed the finite simplex-cover intersection route over KKMFiniteCoverUp, CompactMetricUp, ConvexSetUp, PolytopeUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KKMLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host-selected intersection point, arbitrary open-cover compactness, quotient equality of closed sets, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, face-cover exactness, convex and compact support, intersection-window nonescape, structural replay, provenance exhaustion, and local naming for BEDC.Derived.KKMLemmaUp.

Chapter 9071

A Concrete Naming Certificate for $\mathsf{HenkinModel}^{\uparrow}$

Following Henkin and the standard completeness proof tradition, the $\mathsf{HenkinModel}^{\uparrow}$ packet reconstructs the witness construction as finite BEDC NameCert data. It is a local consumer of the first-order syntax surface of chapter 197 and the satisfaction surface of chapter 198; it does not assert a host compactness theorem or a global completeness theorem.

9071.1 Finite Henkin packet

A carried $\mathsf{HenkinModel}^{\uparrow}$ row is a displayed BHist tuple $(S, \Theta, C, W, T, M, \mathsf{Sat}, \rho, \lambda)$. The row S is the $\mathsf{FirstOrder}^{\uparrow}$ signature and formula endpoint source. The row Θ is a finite theory window, consisting only of the named formula endpoints currently admitted by the packet. The row C lists the Henkin constants exposed inside that window, and the row W records the witness obligations tying existential formula endpoints to their displayed witness constants.

The row T is the finite term-model table. Its entries are raw term endpoints read through S, together with the congruence comparisons already visible in the local packet. The row M packages the term-model interpretation rows for relation and function symbols over those endpoints. The row Sat is the satisfaction ledger: each entry names a formula endpoint, an assignment or closed-term environment endpoint, and the truth row read from the finite Henkin window.

Transport is part of the carrier. The comparison ledger λ records the hsame and Cont rows that move the finite theory window, Henkin constants, witness obligations, term-model rows, and satisfaction entries between classifier-related packets. The provenance row ρ records the Pkg source of each displayed row, so a downstream consumer can see whether a formula endpoint came from the first-order syntax certificate, a witness extension row, or a term-model readback.

9071.2 Local certificate surface

The local $\mathsf{NameCert}_{\mathsf{HenkinModel}^{\uparrow}}$ accepts exactly the rows exposed by the finite packet. A witness obligation is accepted when its existential formula endpoint lies in Θ, its witness constant lies in C, and its instantiated formula endpoint is recorded in the same finite window. A term-model row is accepted when every term endpoint and interpretation entry is read through S, T, and M, with provenance still attached to ρ. A satisfaction entry is accepted when it is one of the displayed Sat rows and its formula endpoint is carried by the same $\mathsf{FirstOrder}^{\uparrow}$ source.

Classifier stability is rowwise. If two $\mathsf{HenkinModel}^{\uparrow}$ packets are related by the displayed comparison ledger, then the finite theory windows, constants, witness rows, term-model entries, interpretation rows, satisfaction entries, and provenance rows transport together. No comparison appeals to ambient syntax equality, quotient equality on terms, or a chosen infinite model outside the packet.

The certificate is deliberately local. It records how a finite Henkin window can be named, transported, and consumed as a term-model witness surface. It does not choose a completed infinite theory, take a quotient by a host equivalence relation, invoke countable choice to select witnesses, or derive the global model-existence theorem.

Theorem 9071.1 (Henkin model witness-extension obligations)**.** *Let*

$$H = (S, \Theta, C, W, T, M, \mathsf{Sat}, \rho, \lambda)$$

be an accepted $\mathsf{HenkinModel}^{\uparrow}$ *packet. Every witness-extension row in* W *extends only displayed formula histories already carried by* Θ *through the* $\mathsf{FirstOrder}^{\uparrow}$ *source* S*. If an existential endpoint is not present as a* BHist *formula row in the finite theory window, the local* $\mathsf{NameCert}_{\mathsf{HenkinModel}^{\uparrow}}$ *has no coordinate that can attach a Henkin constant, instantiated formula endpoint, or downstream witness read to it.*

Proof. Project the accepted packet. The finite theory window Θ and the witness ledger W are carried as displayed BHist rows, and both are read through the same first-order syntax source S.

The local certificate admits a witness obligation only when the existential endpoint, its witness constant, and the instantiated formula endpoint are visible in the same finite window. Thus the witness-extension read is a rowwise extension of displayed formula histories.

The structural rows ρ and λ preserve this restriction under provenance and transport. They replay already displayed witness rows, but they do not create constants for formula endpoints absent from Θ. □

Theorem 9071.2 (Henkin model term-model satisfaction scope)**.** *For every accepted* $\mathsf{HenkinModel}^{\uparrow}$ *packet*

$$H = (S, \Theta, C, W, T, M, \mathsf{Sat}, \rho, \lambda),$$

term-model satisfaction is readable only through the displayed $\mathsf{FirstOrder}^{\uparrow}$ *syntax row* S*, the term table* T*, the interpretation table* M*, and the satisfaction ledger* Sat*, with the model-theoretic use scoped by the* $\mathsf{ModelTheory}^{\uparrow}$ *rows named in the packet provenance. The read exports no ambient syntax quotient, no host equality of terms, and no satisfaction relation outside the finite ledger.*

Proof. Begin with the term-model rows. The table T lists only raw term endpoints read through S, and M interprets only the relation and function endpoints visible in the same packet.

Next inspect the satisfaction ledger. Each entry of Sat names a formula endpoint and an environment endpoint, so the model-theoretic read is a finite ledger row rather than a host satisfaction relation.

The provenance row ρ records the $\mathsf{FirstOrder}^{\uparrow}$ and $\mathsf{ModelTheory}^{\uparrow}$ sources, while λ transports the same entries under accepted comparisons. Since no carrier coordinate stores quotient syntax equality or an external model, the satisfaction scope is exactly the displayed packet surface. □

Theorem 9071.3 (Henkin model completeness-facing non-escape)**.** *Every completeness-facing consumer of an accepted* $\mathsf{HenkinModel}^{\uparrow}$ *packet must factor through*

$$S, \Theta, C, W, T, M, \mathsf{Sat}, \rho, \lambda.$$

It may read displayed formula histories, witness constants, term-model rows, satisfaction entries, provenance, and transport. It cannot export a selected infinite model, quotient syntax equality, countable choice, host compactness, or a global completeness theorem from the local packet.

Proof. The consumer first exposes the finite theory window and witness rows. These rows show which formula endpoints are present and which witness constants are attached inside the packet.

It then reads the term-model and satisfaction rows through T, M, Sat. Those rows are finite BHist data, so the consumer receives only the displayed term and formula entries.

Finally ρ and λ supply provenance and transport for the same finite route. Because the carrier contains no coordinate for selected infinite models, quotient syntax equality, countable choice, host compactness, or global completeness, none of those objects can escape from the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{HenkinModel}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenkinModel}\uparrow}, \mathsf{seed})$

Scope closed: The chapter seeds the finite Henkin witness packet, its term-model rows, satisfaction ledger, transport rows, provenance rows, and local NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HenkinModel}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Constructive story: The construction starts from a finite first-order window, adjoins displayed witness constants for accepted existential endpoints, reads closed terms into the term-model table, and records satisfaction only through the resulting finite ledger.

Not claimed: The chapter does not close host compactness, quotient syntax equality, a selected infinite model, countable choice, or global completeness.

Upgrade path: A scoped certificate requires Lean carriers for the finite theory window, Henkin constants, witness obligations, term-model rows, and satisfaction transport.

Chapter 9072

A Concrete Naming Certificate for FiniteIntervalMeshRefinement$^{\uparrow}$

Following Bishop and Bridges, finite interval arguments are read through displayed meshes, endpoint comparisons, and rational tolerances rather than through a completed continuum of hidden subdivision choices. The FiniteIntervalMeshRefinement$^{\uparrow}$ packet sits between the dyadic arithmetic of chapter 291, the compact interval surfaces of chapter 5937, the compact metric finite-net route of chapter 129, the total-boundedness surface of chapter 127, and the terminal real-name seal of chapter 38. It names the finite refinement discipline used by compact interval and dyadic-net consumers without asserting open-cover compactness, selected partitions, or ambient Real$^{\uparrow}$ completeness.

Definition 9072.1 (Finite interval mesh refinement packet)**.** *A* FiniteIntervalMeshRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, D, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the* CompactInterval$^{\uparrow}$ *endpoint row, displaying the interval source and the endpoint order data used by the mesh read.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *mesh ledger, listing the dyadic breakpoints, dyadic cell radii, endpoint incidence, and finite containment comparisons.*
3. *Q is the refinement map from a coarse finite mesh to the displayed finer mesh, including the finite parent-cell table.*
4. *R is the stability row saying that each refined dyadic cell inherits its endpoint, radius, and incidence rows from the parent cell through the same dyadic comparisons.*
5. *U is the coverage row, recording that the refined mesh covers the same compact interval source and that a downstream* TotallyBounded$^{\uparrow}$ *or* CompactMetric$^{\uparrow}$ *consumer reads only the displayed finite cells.*
6. *H records componentwise* hsame *transport for interval endpoints, dyadic mesh rows, refinement maps, stability rows, coverage rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from an interval-mesh request through I, D, Q, R, U, H.*
8. *P is the* Pkg *provenance over* FiniteIntervalMeshRefinement$^{\uparrow}$, CompactInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite subdivision, a selected limiting mesh, an open-cover choice principle, quotient equality of intervals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9072.2 (Finite interval mesh refinement NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ *carrier*

$$M = (I, D, Q, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the compact interval source I, dyadic mesh ledger D, refinement table Q, stability row R, coverage row U, transport row H, replay row C, provenance row P, and local name N;*
2. *mesh refinement stability, requiring every refined cell in Q to inherit its dyadic endpoint, radius, and incidence rows from a displayed parent cell through R;*
3. *coverage monotonicity, requiring the refined mesh to cover the same interval source as the coarse mesh and to expose no smaller domain when read by a* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{TotallyBounded}^{\uparrow}$ *consumer;*
4. hsame *and* Cont *transport, requiring H and C to preserve the same interval source, dyadic rows, refinement table, stability row, coverage row, provenance row, and local name under accepted renamings and replay;*
5. *ledger refusal, requiring the packet to export no infinite subdivision, selected limiting mesh, open-cover choice principle, quotient interval equality, hidden Real selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No finite interval mesh-refinement obligation is stored outside these rows.

Proof. Project the carrier of Definition 9072.1. The interval row I, dyadic ledger D, refinement table Q, stability row R, coverage row U, and structural rows H, C, P, N are the entire displayed packet.

Carrier admission is therefore admission of those rows. Mesh refinement stability is read from Q together with R: each refined cell points to a displayed parent cell, and the endpoint, radius, and incidence data are transported through dyadic comparisons already present in D.

Coverage monotonicity is read from U. The refined mesh may subdivide cells, but it keeps the same compact interval source and the same finite coverage target before any compact metric or totally bounded consumer reads it.

Finally H, C, P, N transport, replay, source, and locally name the same finite route. Since the carrier contains no coordinate for infinite subdivision, selected limiting meshes, open-cover choice, quotient interval equality, hidden Real selectors, or ambient Real completeness, the five listed rows exhaust the local NameCert surface. □

Theorem 9072.3 (Finite interval mesh refinement compact handoff)**.** *Every compact interval, compact metric, totally bounded, dyadic-net, or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ *packet must read the displayed route*

$$I, D, Q, R, U, H, C, P, N$$

before treating a refined mesh as usable finite data. The handoff is a finite BEDC mesh route, not a selected subdivision limit, not an open-cover compactness theorem, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 9072.2. They force the refined mesh to expose carrier admission, refinement stability, coverage monotonicity, transport, replay, provenance, local naming, and ledger refusal.

A compact interval or dyadic-net consumer reads I, D, Q, R, U first. These rows display the source interval, dyadic cells, parent-cell table, inherited endpoint data, and coverage ledger.

A compact metric, totally bounded, or Real-facing consumer is downstream of the same displayed route. The rows H, C, P, N preserve and name it without creating a selected limiting partition or a hidden completeness principle. Thus every accepted handoff remains finite and local to the packet. □

Falsifiable prediction: Every accepted $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ read exposes a compact interval source row, dyadic mesh ledger, finite refinement table, stability row, coverage row, hsame transport, Cont replay, provenance, and local NameCert before a compact interval or Real-facing consumer reads the refined mesh.

Independence witness: dyadicratcore, compact-interval, compactmetric, totallybounded, real: no listed sibling simultaneously carries a finite parent-cell refinement table, inherited dyadic endpoint rows, coverage monotonicity, transport, replay, provenance, and local NameCert rows for compact interval mesh refinement.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ packet is generated from a compact interval source, a DyadicRatCore mesh ledger, a finite parent-cell refinement table, stability and coverage rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9072.1, Definition 9072.2, and Definition 9072.3 seed the finite compact-interval mesh refinement route from dyadic cell rows to compact and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, infinite subdivisions, selected limiting meshes, quotient interval equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh refinement stability, coverage monotonicity, transport, replay, provenance, and ledger refusal for BEDC.Derived.FiniteIntervalMeshRefinementUp.

Chapter 9073

A Concrete Naming Certificate for $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$

Following Bishop and Bridges, Lipschitz control on a compact interval is read through finite interval meshes, displayed bounds, and regular rational readback rather than through an ambient supremum over an uninspected continuum. The $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ packet sits after the compact metric surface of chapter 129, the continuous-map route of chapter 130, the Lipschitz-map carrier of chapter 410, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. It realizes the compact interval finite-envelope route without claiming open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a selected global extremum.

Definition 9073.1 (Compact interval Lipschitz-envelope packet)**.** *A* $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, L, M, B, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact interval source, including the finite mesh rows accepted by the interval carrier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{LipschitzMap}^{\uparrow}$ *function row whose values are being bounded.*
3. *L is the finite Lipschitz-bound row, displaying the rational or dyadic bound used to compare values across mesh cells.*
4. *M is the finite mesh row, read through* $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ *or an existing compact interval finite-net sibling when the consumer supplies one.*
5. *B is the envelope row assigning displayed lower and upper rational bounds to each mesh cell.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed envelope bounds.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for envelope values exposed after K, F, L, M, B, R have all been displayed.*
8. *H records componentwise* hsame *transport for compact source rows, map rows, Lipschitz bounds, mesh rows, envelope rows, readback rows, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from an envelope request through K, F, L, M, B, R, S, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for an uninspected supremum, a selected maximizing point, an open-cover choice principle, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9073.2 (Compact interval Lipschitz-envelope NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ *carrier*

$$E = (K, F, L, M, B, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}}$ *surface requires exactly the displayed compact interval source K, function row F, finite Lipschitz-bound row L, finite mesh row M, envelope bound row B, regular readback row R, Real seal S, transport H, replay C, provenance P, and local name N. A consumer may read a compact interval envelope only after these rows expose the finite mesh, Lipschitz bound, uniform-modulus handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Project the carrier of Definition 9073.1. The source, map, Lipschitz bound, mesh, envelope, readback, seal, and structural rows are the whole packet.

The finite mesh row M is the only place where the interval is inspected. The Lipschitz row L then transports pointwise control across cells, and B records the resulting lower and upper rational envelope bounds.

The readback and seal rows R, S make the envelope usable by regular rational and Real-facing consumers only after the finite rows have been displayed. Rows H, C, P, N preserve, replay, source, and name the same route. Since no additional coordinate is present, the certificate exports no uninspected supremum, selected maximizer, open-cover choice, quotient function equality, or ambient Real completeness. □

Theorem 9073.3 (Compact interval Lipschitz-envelope uniform route)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ *packet factors through*

$$K, F, L, M, B, R, S, H, C, P, N.$$

The packet supplies finite mesh rows, a Lipschitz bound, envelope bounds, uniform-modulus handoff data, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal. It does not supply open-cover compactness, selected extrema, quotient function representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 9073.2. It identifies the carrier rows that must be exposed before an envelope is public.

The compact-uniform-continuity consumer reads K, M as the finite compact interval mesh route and reads F, L as the map and Lipschitz-control route. It then reads B as the finite envelope ledger.

The rows R, S turn that ledger into regular rational readback and a Real-facing seal. The structural rows H, C, P, N keep the route stable under transport and replay. Thus the consumer receives exactly the displayed finite envelope and none of the refused ambient principles. □

Falsifiable prediction: Every accepted $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ read exposes compact interval finite mesh rows, a ContinuousMap or LipschitzMap row, a finite Lipschitz bound, envelope bounds, RegSeqRat readback, a RealUp seal, transport, replay, provenance, and local NameCert before a compact-uniform-continuity consumer reads the envelope.

Independence witness: compactmetric, continuousmap, lipschitzmap, regseqrat, real: no listed sibling simultaneously carries compact interval mesh rows, Lipschitz envelope bounds, regular rational readback, Real sealing, transport, replay, provenance, and local NameCert rows for a compact interval envelope.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}$ packet is generated from a compact interval finite-mesh source, a ContinuousMap or LipschitzMap row, a finite Lipschitz-bound row, envelope bound rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9073.1, Definition 9073.2, and Definition 9073.3 seed the compact interval Lipschitz-envelope route from finite mesh and Lipschitz rows to RegSeqRat and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalLipschitzEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected extrema, quotient function representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, map admission, Lipschitz-bound stability, finite mesh envelope bounds, RegSeqRat readback, Real sealing, transport, replay, provenance, and ledger refusal for BEDC.Derived.CompactIntervalLipschitzEnvelopeUp.

Chapter 9074

A Concrete Naming Certificate for $\mathsf{BorelMeasure}^{\uparrow}$

Following Halmos and Bourbaki on Borel measures, the $\mathsf{BorelMeasure}^{\uparrow}$ packet records the finite route from a displayed measurable space and its generated Borel event row to a measure value surface. It sits after the measurable-space carrier of chapter 3537, the cover-valued outer-measure boundary of chapter 3385, and the regular-measure consumer surface of chapter 2263. The packet names a Borel-event measure route for downstream Radon and probability consumers; it is not a completed measure algebra, a host sigma-finiteness theorem, a quotient equality of measures, or an ambient standard Borel oracle.

Definition 9074.1 (BorelMeasure carrier)**.** *A* $\mathsf{BorelMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, A, E, O, M, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying measurable-space carrier row read through* $\mathsf{MeasurableSpace}^{\uparrow}$*.*
2. *A is the displayed Borel event-family row over X, read through the generated-event surface already exposed by the measurable-space packet.*
3. *E is the event-admission classifier, recording that every measured event is read through A before any value row is exposed.*
4. *O is the outer-measure boundary row, carrying empty-set, monotonicity, finite-cover, and Caratheodory test data from* $\mathsf{OuterMeasure}^{\uparrow}$*.*
5. *M is the Borel measure value row, assigning displayed real values only to admitted Borel events.*
6. *R is the regular-measure handoff row for consumers that later read the same values through* $\mathsf{RadonMeasure}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for measurable-space, Borel-event, outer-measure, value, Radon-handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the Borel event request through event admission, outer-measure boundary, measure value, Radon handoff, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{BorelMeasure}^{\uparrow}$*,* $\mathsf{MeasurableSpace}^{\uparrow}$*,* $\mathsf{OuterMeasure}^{\uparrow}$*,* $\mathsf{RadonMeasure}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BorelMeasure}^{\uparrow}}$ *row.*
The carrier has no coordinate for a completed measure algebra, host sigma-finiteness, quotient equality of measures, choice-selected Borel codes, or an ambient standard-model probability space.

Theorem 9074.2 (BorelMeasure NameCert obligations)**.** *For every accepted* $\mathsf{BorelMeasure}^{\uparrow}$ *carrier $B = (X, A, E, O, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BorelMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission through the measurable-space row X, Borel-event admission through A and E, outer-measure boundary exposure through O, Borel measure valuation through M, Radon-facing regularity handoff through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Every downstream measure, probability, or Radon consumer must read the Borel event row before the value row. The packet exports no completed measure algebra, host sigma-finiteness theorem, quotient measure equality, choice-selected code enumeration, or ambient standard Borel oracle.

Proof. Project the carrier of Definition 9074.1. The rows $X, A, E, O, M, R, H, C, P, N$ are the entire public surface of the packet.

The measurable-space row X and Borel event rows A, E supply the domain on which value reads are legal. The outer-measure boundary row O is the only route by which finite cover and Caratheodory test data enter the carrier.

The measure value row M is therefore exposed only after Borel event admission and outer-measure boundary data. The Radon handoff row R records a downstream regularity request, but it does not add a host regularity theorem or a completed measure object.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since no further coordinate is present, the listed rows exhaust the local NameCert obligation surface. □

Theorem 9074.3 (BorelMeasure measurable-space handoff)**.** *Every accepted* $\mathsf{BorelMeasure}^{\uparrow}$ *packet hands a measure-facing consumer the route*

$$X, A, E, O \longmapsto M \longmapsto R, H, C, P, N$$

where X, A, E are read through chapter 3537, O is read through chapter 3385, and R is the regularity-facing row consumed by chapter 2263. Thus Borel measure values are public only after the measurable-event and outer-measure boundary rows have been displayed.

Proof. Start with the measurable-space carrier. It supplies the underlying point row and generated Borel event family, so the Borel measure packet cannot expose M before X, A, E have been read.

Next use the outer-measure boundary. The row O supplies the finite cover and Caratheodory test data that make the Borel valuation a displayed measure route rather than a host assignment of values to arbitrary subsets.

The Radon handoff row R is downstream from M. Transport, replay, provenance, and local naming through H, C, P, N keep the same order, so a Radon or probability consumer receives only the displayed Borel measure route and no hidden completed measure algebra. □

Falsifiable prediction: A $\mathsf{BorelMeasure}^{\uparrow}$ consumer that reads a measure value M without first exposing the Borel event rows A, E and the outer-measure boundary row O refutes the carrier surface.

Independence witness: measurable-space, outer-measure, radon-measure, measure, probspace: no listed sibling supplies the full ten-row packet $B = (X, A, E, O, M, R, H, C, P, N)$, because each

sibling contributes only event admission, cover boundary, regularity, value, or normalization data while this chapter binds the Borel measure route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{BorelMeasure}^{\uparrow}$ packet is generated from measurable-space carrier rows, Borel event rows, event admission, outer-measure boundary data, Borel measure values, Radon handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9074.1, Definition 9074.2, and Definition 9074.3 seed the finite Borel measure route over measurable-space, Borel-event, outer-measure, value, Radon-handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, sigma-finiteness, quotient measure equality, selected Borel code enumeration, probability normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Borel-event exposure, outer-measure boundary coverage, Borel value exactness, Radon handoff, transport, replay, provenance, and local naming for BEDC.Derived.BorelMeasureUp.

Chapter 9075

A Concrete Naming Certificate for $\mathsf{SolenoidPhase}^{\uparrow}$

Following Pontryagin duality and the classical theory of solenoids, the $\mathsf{SolenoidPhase}^{\uparrow}$ packet records the finite phase carrier associated to an S-localized integer row and its compact inverse-limit circle presentation. It realizes the solenoid carrier described in section 9413.63 as concrete NameCert data, reading prime rows from chapter 64, visible circle phase rows from chapter 4010 and chapter 58, and compact group operation rows from chapter 204. The packet names only displayed localization, circle projection, compact fibre, resonance phase, transport, replay, provenance, and local naming rows; it does not assert a zeta zero theorem, a hidden Pontryagin-dual object, quotient equality of characters, or an ambient compactness principle.

Definition 9075.1 (SolenoidPhase carrier)**.** *A* $\mathsf{SolenoidPhase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, L, Q, V, F, R, H, C, G, N)$$

with the following displayed rows:

1. *P is the displayed prime-family row, read through* $\mathsf{Prime}^{\uparrow}$*, naming the finite prime set or prime-window source whose elements are made invertible.*
2. *L is the S-localization row for the integer carrier, recording the denominators whose prime divisors are constrained by P.*
3. *Q is the compact solenoid phase row, read as the dual or inverse-limit phase carrier generated by L.*
4. *V is the visible circle projection row, routed through the* $\mathsf{Circle}^{\uparrow}$ *and* $\mathsf{S}^{1\uparrow}$ *phase surfaces.*
5. *F is the compact fibre row, recording the kernel of the visible projection without naming a hidden quotient space.*
6. *R is the resonance phase row, carrying the displayed prime-local phase coordinates consumed by the solenoid Lee–Yang route.*
7. *H records componentwise* hsame *transport for prime-family, localization, compact phase, circle projection, fibre, resonance, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from prime-family selection through localization, compact solenoid phase, visible circle projection, fibre, resonance row, provenance, and local naming.*

9. *G is the* Pkg *provenance over* $\mathsf{SolenoidPhase}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{TopGroup}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SolenoidPhase}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host Pontryagin-dual construction, an undisplayed character group, quotient equality of phase points, a zeta-zero locator, or a noncompact positive scale.

Theorem 9075.2 (SolenoidPhase NameCert obligations)**.** *For every accepted* $\mathsf{SolenoidPhase}^{\uparrow}$ *carrier* $S = (P, L, Q, V, F, R, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SolenoidPhase}^{\uparrow}}$ *surface consists exactly of prime-family admission through P, S-localization exposure through L, compact solenoid phase admission through Q, visible circle projection through V, compact fibre exposure through F, resonance phase exposure through R, componentwise transport through H, displayed replay through C, provenance through G, and local naming through N.*

Every solenoid-facing resonance consumer must read the localization and visible circle projection before the resonance phase row is public. The packet exports no hidden Pontryagin-dual carrier, quotient character equality, zeta-zero theorem, ambient compactness principle, or noncompact scale coordinate.

Proof. Project the carrier of Definition 9075.1. The rows $P, L, Q, V, F, R, H, C, G, N$ are the only rows available to a consumer.

The prime-family row P and localization row L supply the arithmetic source. The compact solenoid phase row Q, visible projection row V, and compact fibre row F are then displayed before the resonance row R can be consumed.

The structural rows H, C, G, N transport, replay, source, and name the same finite packet. Since no coordinate stores a hidden character group, quotient phase equality, zeta-zero locator, or noncompact scale, the listed rows exhaust the local NameCert obligation surface. □

Theorem 9075.3 (SolenoidPhase resonance handoff)**.** *Every accepted* $\mathsf{SolenoidPhase}^{\uparrow}$ *packet hands a resonance consumer the route*

$$P, L \longmapsto Q, V, F \longmapsto R, H, C, G, N$$

where P is read through chapter 64, V is read through the circle phase rows of chapter 4010 and chapter 58, and Q, V, F are compatible with the topological-group operation surface of chapter 204. Thus the resonance phase row is public only after the displayed localization, compact phase, visible circle, and compact fibre rows have been exposed.

Proof. Start with the arithmetic source. The prime-family row P fixes the admitted prime directions, and the localization row L records exactly which denominator primes can appear.

Next expose the compact phase carrier. The solenoid row Q is read together with its visible circle projection V and compact fibre F, so a resonance consumer cannot treat the phase as a noncompact positive scale or as an undisplayed character group.

Finally read R, H, C, G, N. The resonance row R is downstream from the localization and compact phase rows, while transport, replay, provenance, and local naming preserve that order. Hence the handoff is a finite phase certificate, not a proof of the zeta-specific dominance theorem. □

Falsifiable prediction: A $\mathsf{SolenoidPhase}^{\uparrow}$ consumer that reads a resonance phase row R without first exposing the prime-family row P, localization row L, visible circle projection V, and compact fibre row F refutes the carrier surface.

Independence witness: prime, circle, s1, topgroup: no listed sibling supplies the full ten-row packet $S = (P, L, Q, V, F, R, H, C, G, N)$, because each sibling contributes only prime admission, visible phase, group operation, or compactness-facing data while this chapter binds the solenoid phase route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SolenoidPhase}\uparrow}$.

Constructive story: A $\mathsf{SolenoidPhase}^{\uparrow}$ packet is generated from prime-family rows, localization rows, compact solenoid phase rows, visible circle projection rows, compact fibre rows, resonance phase rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SolenoidPhase}\uparrow}, \mathsf{seed})$

Scope closed: Definition 9075.1, Definition 9075.2, and Definition 9075.3 seed the finite solenoid phase route over prime-family, localization, compact phase, visible circle, fibre, resonance, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SolenoidPhase}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the zeta solenoid Lee–Yang row, zeta-zero classification, host Pontryagin duality, quotient character equality, noncompact positive scale exclusion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, localization exposure, compact phase admission, visible circle projection, compact fibre exposure, resonance handoff, transport, replay, provenance, and local naming for BEDC.Derived.SolenoidPhaseUp.

Chapter 9076

A Concrete Naming Certificate for $\mathsf{FrobeniusCharacterFormula}^{\uparrow}$

Following Serre and Fulton–Harris on finite-group characters, the $\mathsf{FrobeniusCharacterFormula}^{\uparrow}$ packet records the finite value formula for an induced character as NameCert data. It consumes the finite group and subgroup sources of chapter 41 and chapter 80, the finite representation row of chapter 3662, the induced-character packet of chapter 7757, the character-table consumer of chapter 3574, and the reciprocity comparison surface of chapter 5806. The packet names only the displayed coset-filtered summation rule, its conjugacy stability, its finite table handoff, and its reciprocity-facing provenance.

Definition 9076.1 (FrobeniusCharacterFormula carrier)**.** *A* $\mathsf{FrobeniusCharacterFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{F} = (G, H, Q, V, \theta, \chi, \sigma, \eta, \tau, \rho, \lambda, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row naming the finite group source.*
2. *H is the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *row naming the subgroup inclusion inside G.*
3. *Q is the finite coset ledger whose accepted rows are used as the only summation indices.*
4. *V is the consumed* $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *row whose restricted trace supplies the source character on H.*
5. *θ is the source character value row on H.*
6. *χ is the consumed* $\mathsf{NameCert}_{\mathsf{InducedCharacter}^{\uparrow}}$ *row naming the induced character to be evaluated.*
7. *σ is the displayed finite formula ledger: for each accepted group element row g, it lists exactly those accepted coset rows q whose transported element row lies in H, and it assigns the finite sum of the corresponding θ values to $\chi(g)$.*
8. *η is the representative-independence row, recording that* hsame *replacement of accepted coset rows in Q preserves the displayed summand ledger.*
9. *τ is the conjugacy-stability row, recording that the value row is transported through conjugation in G as a class-function value.*
10. *ρ is the* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *provenance row that lets reciprocity consumers read the same induction–restriction boundary.*

11. *λ is the downstream consumer ledger for* $\mathsf{CharacterTable}^{\uparrow}$, $\mathsf{InducedCharacter}^{\uparrow}$, *and* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *reads.*
12. *P is the* Pkg *provenance over the consumed rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusCharacterFormula}^{\uparrow}}$ *row.*

The carrier contains no irreducible decomposition, no quotient-character row, no host representation category, and no selected coset representatives outside the displayed ledger Q.

Theorem 9076.2 (FrobeniusCharacterFormula NameCert obligations)**.** *For every accepted* FrobeniusCharacterForm *packet*

$$\mathcal{F} = (G, H, Q, V, \theta, \chi, \sigma, \eta, \tau, \rho, \lambda, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FrobeniusCharacterFormula}^{\uparrow}}$ *surface consists exactly of the following six finite obligations:*

1. *carrier admission exposes precisely the rows listed in Definition 9076.1;*
2. *subgroup admissibility checks every summand of σ against the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *row H inside the same* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source G;*
3. *source-character routing requires every summand value to be read through θ and the finite representation row V;*
4. *representative independence is the row η, so changing an accepted coset representative by* hsame *changes only the displayed index, not the named value row;*
5. *conjugacy stability is the row τ, so the formula exports a class-function value row to character-table consumers;*
6. *reciprocity provenance requires every comparison read to pass through ρ and the Frobenius reciprocity boundary named by chapter 5806.*

Proof. Project the carrier of Definition 9076.1. The tuple $\mathcal{F}$ already displays the group, subgroup, coset, representation, source-character, induced-character, formula, stability, reciprocity, consumer, provenance, and local naming rows, giving carrier admission.

Subgroup admissibility and source-character routing are checked on each listed summand of σ. The admissibility test reads only the row H inside G, and the summand value reads only θ through the finite representation row V.

Representative independence is exactly the row η. It compares two accepted rows of the same finite coset ledger Q by hsame transport and leaves the subgroup test and source-character value row unchanged.

Conjugacy stability is the row τ. It transports the displayed group element row through conjugation in G and repacks the same formula value as the class-function row consumed by $\mathsf{CharacterTable}^{\uparrow}$.

Finally, ρ supplies the reciprocity-facing provenance. Repacking these projected rows with λ, P, and N gives exactly the six listed NameCert obligations and no additional host character theory. □

Theorem 9076.3 (FrobeniusCharacterFormula induced value route)**.** *Any* $\mathsf{InducedCharacter}^{\uparrow}$ *or* $\mathsf{CharacterTable}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrobeniusCharacterFormula}^{\uparrow}$ *packet may read the induced value at a displayed group row only through*

$$G,\ H,\ Q,\ V,\ \theta,\ \chi,\ \sigma,\ \eta,\ \tau,\ \lambda.$$

It may not import an irreducible decomposition, a host representation category, a quotient-character row, or a coset representative outside Q.

Proof. Route the consumer request through the local packet. The group and subgroup boundary is supplied by G and H, and the only indexing surface is the finite coset ledger Q.

The source values are read through V and θ. The induced-character name is χ, and the value itself is the finite formula row σ.

The independence and table-facing stability checks are η and τ. The downstream read is recorded by λ, so no other row of $\mathcal{F}$ can supply an irreducible decomposition, host representation category, quotient character, or external representative selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusCharacterFormula}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusCharacterFormula}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9076.1, Definition 9076.2, and Definition 9076.3 seed the finite induced-character value formula over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{InducedCharacter}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, and $\mathsf{FrobeniusReciprocity}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusCharacterFormula}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close irreducible decomposition, host representation categories, quotient characters, selected coset representatives outside the displayed ledger, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subgroup admissibility, source-character routing, representative independence, conjugacy stability, and reciprocity provenance for the same finite packet.

Constructive story: A $\mathsf{FrobeniusCharacterFormula}^{\uparrow}$ packet is generated from finite group and subgroup sources, a finite coset ledger, a represented source character, an induced-character row, a displayed finite formula ledger, representative-independence and conjugacy-stability rows, reciprocity provenance, downstream consumer rows, package provenance, and a local NameCert row.

Chapter 9077

A Concrete Naming Certificate for $\mathsf{FloydWarshallClosure}^{\uparrow}$

Following Floyd and Warshall, finite all-pairs shortest-path closure is read here as a stage-indexed matrix certificate rather than as an ambient graph-theoretic oracle. The $\mathsf{FloydWarshallClosure}^{\uparrow}$ packet sits after the finite graph rows of chapter 118, the single-path ledger of chapter 236, the matrix surface of chapter 49, the finite order and natural-stage rows of $\mathsf{Order}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$, and the nonnegative weight seal of $\mathsf{Real}^{\uparrow}$. It records the all-pairs closure route consumed by finite algorithm certificates, not negative-cycle detection, choice of a canonical path representative, quotient equality of paths, or ambient completeness of weights.

Definition 9077.1 (FloydWarshallClosure certificate)**.** *A* $\mathsf{FloydWarshallClosure}^{\uparrow}$ *certificate is a finite* BHist *packet*

$$F = (G, W, D, K, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Graph}^{\uparrow}$ *row listing vertices and directed edge incidence.*
2. *W is the nonnegative weight row, read through* $\mathsf{Real}^{\uparrow}$ *and the finite order surface of* $\mathsf{Order}^{\uparrow}$*.*
3. *D is the distance-matrix row, read as a finite* $\mathsf{Mat}^{\uparrow}$ *table indexed by ordered vertex pairs.*
4. *K is the* $\mathsf{Nat}^{\uparrow}$ *intermediate-vertex stage index, displaying the finite order in which vertices are admitted as possible internal nodes.*
5. *R is the relaxation table; its k, i, j entry records the displayed comparison between the old distance D_{ij} and the routed candidate $D_{ik} + D_{kj}$.*
6. *H records componentwise* hsame *transports for the graph, weight, matrix, stage, and relaxation rows.*
7. *C records displayed* Cont *replays of the finite relaxation passes.*
8. *P is the* Pkg *provenance over* $\mathsf{FloydWarshallClosure}^{\uparrow}$*,* $\mathsf{Graph}^{\uparrow}$*,* $\mathsf{ShortestPath}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FloydWarshallClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, W, D, K, R, H, C, P, N$. It has no coordinate for an infinite graph, a selected global path representative, a negative-cycle oracle, quotient path equality, or an ambient real-completeness principle.

Theorem 9077.2 (FloydWarshallClosure NameCert obligations)**.** *For every accepted* $\mathsf{FloydWarshallClosure}^{\uparrow}$ *certificate* $F = (G, W, D, K, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FloydWarshallClosure}^{\uparrow}}$ *surface consists exactly of finite graph admission, nonnegative weight exposure, matrix-entry exposure, intermediate-stage admission, relaxation-table exactness, componentwise transport, continuation replay, package provenance, and local naming. Every all-pairs shortest-path consumer must read those rows before interpreting the matrix* D *as a closed distance table.*

Proof. Project the certificate of Definition 9077.1. The rows G, W, D, K, R are the only graph, weight, distance-matrix, stage, and relaxation data available to a consumer.

The relaxation obligation is finite because K lists the displayed intermediate vertices and D is a finite $\mathsf{Mat}^{\uparrow}$ table over ordered vertex pairs. For each k, i, j, the row R stores the comparison between the previous entry and the route through k, so no hidden shortest-path oracle is needed.

Transport and replay preserve the same displayed rows. Componentwise hsame through H cannot add a new graph, weight, or path representative, and the Cont replay through C only replays the listed relaxation passes.

Finally P and N source and name the same finite packet. Since the carrier contains no infinite graph, negative-cycle oracle, quotient path equality, selected representative, or ambient completeness row, the local NameCert surface is exactly the displayed all-pairs closure certificate. □

Theorem 9077.3 (FloydWarshallClosure induction invariant)**.** *Let* $F = (G, W, D, K, R, H, C, P, N)$ *be an accepted* $\mathsf{FloydWarshallClosure}^{\uparrow}$ *certificate. Inducting over the displayed intermediate-vertex row* K*, after the stage prefix* k *every matrix entry in* D *records the least displayed route whose internal vertices are drawn from that prefix, as witnessed by the relaxation entries of* R *and the nonnegative weight comparisons in* W*. The invariant is transported only through* H *and replayed only through* C*.*

Proof. At the empty stage prefix, the matrix row D reads only the direct graph edges and diagonal entries exposed by G and W. The order and nonnegativity rows make each base comparison visible without selecting an external path representative.

For the successor stage, expose the next intermediate vertex from K. For each ordered pair i, j, the relaxation row R compares the previous entry with the displayed routed candidate through that vertex. The finite order row decides which displayed entry is retained in the certificate table.

The induction step is componentwise over the matrix indices. Because the matrix is finite, the proof rebuilds the invariant entry-by-entry, using only rows already present in G, W, D, K, R.

Transport through H and replay through C preserve that finite inventory. They cannot introduce a hidden path oracle, quotient path equality, or ambient real-completeness principle, so the induction invariant remains the displayed Floyd-Warshall stage invariant. □

Theorem 9077.4 (FloydWarshallClosure shortest-path handoff)**.** *Every* $\mathsf{ShortestPath}^{\uparrow}$ *or finite algorithm consumer of an accepted* $\mathsf{FloydWarshallClosure}^{\uparrow}$ *certificate must read the route*

$$G, W, D, K, R, H, C, P, N$$

before using any all-pairs distance entry. The handoff interprets each selected source-target entry of D *through the finite graph row, nonnegative weight row, stage induction row, and relaxation table, not through a detached Dijkstra ledger, Bellman–Ford ledger, quotient path equality, or selected global path representative.*

Proof. Begin with the NameCert obligations of Definition 9077.2. They require every consumer to expose the graph, weight, matrix, stage, relaxation, transport, replay, provenance, and naming rows before the matrix can be read as closed.

Apply the induction invariant of Definition 9077.3. It identifies each matrix entry after the final stage as the result of the finite relaxation pass over the displayed intermediate-vertex row K.

A $\mathsf{ShortestPath}^{\uparrow}$ consumer can therefore project a single source-target entry only after the all-pairs closure route has been read. The projected entry remains tied to $G, W, D, K, R, H, C, P, N$.

The packet has no coordinate for a detached single-source ledger, a selected global path representative, quotient path equality, negative-cycle detection, or ambient completeness of weights. Thus the shortest-path handoff is the displayed finite closure certificate itself. □

Falsifiable prediction: Every accepted $\mathsf{FloydWarshallClosure}^{\uparrow}$ all-pairs read exposes one finite Graph row, one nonnegative weight row, one Matrix distance table, one Nat stage row, one relaxation table, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a shortest-path consumer reads a distance entry.

Independence witness: graph, shortestpath, matrix, order, nat, real: $\mathsf{FloydWarshallClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite all-pairs distance matrix, the displayed intermediate-stage row, and the relaxation table in one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FloydWarshallClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{FloydWarshallClosure}^{\uparrow}$ packet is generated from a finite graph row, nonnegative weight row, finite distance matrix, intermediate-vertex stage index, relaxation table, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FloydWarshallClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9077.1, Definition 9077.2, Definition 9077.3, and Definition 9077.4 seed the finite all-pairs shortest-path closure certificate and its handoff to shortest-path consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FloydWarshallClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close negative-cycle detection, selected global path representatives, quotient path equality, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, relaxation-table exactness, stage induction, transport stability, replay stability, provenance, and local naming for BEDC.Derived.FloydWarshallClosureUp.

Chapter 9078

A Concrete Naming Certificate for FiniteNetSelectionPrinciple$^{\uparrow}$

Following Bishop and Bridges on constructive compactness and total boundedness, the FiniteNetSelectionPrinciple$^{\uparrow}$ packet records finite net selection as displayed interval or metric data rather than as an open-cover compactness principle. It sits after the finite interval mesh discipline of chapter 9072, the modulus-controlled completion route of chapter 9069, and the density window surface of chapter 1809. The packet supplies the Real and Completion corridor with an L10 selector object for finite dyadic nets, without claiming a host choice function, selected Cauchy limits, Heine–Borel compactness, or ambient Real$^{\uparrow}$ completeness.

Definition 9078.1 (Finite net selection principle carrier)**.** *A* FiniteNetSelectionPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, L, M, Q, W, R, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the certified located interval or located metric source row whose points may be inspected only through displayed finite probes.*
2. *L is the locatedness row, giving the interval endpoint or metric-ball witness used to decide finite covering incidence.*
3. *M is the finite dyadic mesh or finite metric-net row, read through* FiniteIntervalMeshRefinement$^{\uparrow}$ *when the source is an interval and through the accepted total-bounded or compact-metric net row when the source is metric.*
4. *Q is the finite selector table assigning each requested dyadic or metric tolerance to a displayed finite list of net centers.*
5. *W is the mesh monotonicity row, saying that a sharper requested tolerance refines the previous finite selector table through displayed parent-cell or parent-ball incidence.*
6. *R is the cover soundness row, recording that every admitted source probe is covered by one displayed net center at the requested tolerance.*
7. *T is the selector stability row, requiring* hsame *transport to preserve source rows, located witnesses, mesh rows, selector tables, monotonicity rows, cover rows, replay, provenance, and local naming.*
8. *H records componentwise* hsame *transport for the displayed selector route.*

9. *C records displayed* Cont *replay from a finite-net request through X, L, M, Q, W, R, T, H.*
10. *P is the* Pkg *provenance over* FiniteNetSelectionPrinciple$^{\uparrow}$, FiniteIntervalMeshRefinement$^{\uparrow}$, CauchyUniformContinu RealSeparability$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite open cover, a global choice function, a selected limit point, quotient completion equality, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.FiniteNetSelectionPrincipleUp`

Theorem 9078.2 (Finite net selection principle obligations)**.** *For every accepted* FiniteNetSelectionPrinciple$^{\uparrow}$ *carrier*

$$S = (X, L, M, Q, W, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source X, locatedness witness L, mesh or net row M, selector table Q, monotonicity row W, cover row R, selector stability row T, transport row H, replay row C, provenance row P, and local name N;*
2. *classifier locality, comparing accepted selectors only through the displayed located source, finite mesh or net, selector table, monotonicity, cover, stability, transport, replay, provenance, and naming rows;*
3. *mesh monotonicity, requiring sharper tolerances to refine the selected finite net through the parent-cell or parent-ball incidence recorded in W;*
4. *cover soundness, requiring every admitted located source probe to be covered by a displayed net center before a Real or Completion consumer reads the selector;*
5. *selector stability, requiring T, H, C, P, N to preserve, replay, source, and locally name the same finite selector route under accepted renamings.*

No finite net selector obligation is stored outside these rows.

Proof. Project the carrier of Definition 9078.1. The source row, locatedness row, mesh or net row, selector table, monotonicity row, cover row, and structural rows are the whole displayed packet.

Carrier admission and classifier locality are therefore row inventory facts. A consumer can compare two accepted selectors only by reading $X, L, M, Q, W, R, T, H, C, P, N$, so no hidden open-cover or choice coordinate can enter the classifier.

Mesh monotonicity is read from W. A sharper tolerance may subdivide interval cells or metric balls, but the refinement must point back to displayed parent incidence in the same finite route.

Cover soundness is read from R. The selected net is usable only after every admitted located probe has a displayed center at the requested tolerance.

Finally T, H, C, P, N transport, replay, source, and locally name the same finite route. Since the carrier contains no row for infinite covers, global choice functions, selected limits, quotient completion equality, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 9078.3 (Finite net selection principle ledger)**.** *Every Real or Completion consumer of an accepted* FiniteNetSelectionPrinciple$^{\uparrow}$ *packet must read the finite route*

FiniteIntervalMeshRefinement$^{\uparrow}$ $\longmapsto$ FiniteNetSelectionPrinciple$^{\uparrow}$ $\longmapsto$ CauchyUniformContinuation$^{\uparrow}$ $\longmapsto$ RealSeparabilit

through the displayed rows $X, L, M, Q, W, R, T, H, C, P, N$ before it treats a finite net as available selector data. The ledger exports a finite dyadic or metric net handoff; it exports no classical open-cover compactness theorem, no choice-based Heine–Borel principle, no selected Cauchy limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 9078.2. They force every selector read to expose the located source, finite mesh or net, selector table, monotonicity, cover, stability, transport, replay, provenance, and local naming rows.

The interval-facing branch reads M through chapter 9072. Thus dyadic refinement data remain finite parent-cell data before any selector table is public.

The completion-facing branch then reads continuation requests through chapter 9069. The selector table can be used only as a finite window for a modulus-controlled Cauchy route, not as a detached completed value.

The Real-facing density branch reads chapter 1809 only after the selector has displayed its net centers and cover row. Hence the consumer receives finite density windows and not a host dense subset, a countable-choice enumeration, or an ambient compactness theorem. □

Falsifiable prediction: Every accepted $\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}$ read exposes a located interval or metric source row, locatedness witness, finite dyadic mesh or metric-net row, selector table, mesh monotonicity row, cover soundness row, selector stability row, hsame transport, Cont replay, provenance, and local NameCert before a Real or Completion consumer reads finite-net selector data.

Independence witness: finite-interval-mesh-refinement, cauchy-uniform-continuation, real-separability: no listed sibling simultaneously carries located source witnesses, finite selector tables, mesh monotonicity, cover soundness, selector stability, completion continuation handoff, density-window handoff, transport, replay, provenance, and local NameCert rows for constructive finite-net selection.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}$ packet is generated from a certified located interval or metric source, a locatedness witness, a finite dyadic mesh or metric-net row, a finite selector table, mesh monotonicity, cover soundness, selector stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9078.1, Definition 9078.2, and Definition 9078.3 seed the finite selector route for located interval and metric finite-net reads in the Real and Completion corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.FiniteNetSelectionPrincipleUp`

Bridge status: none

Not claimed: This chapter does not claim classical open-cover compactness, choice-based Heine–Borel selection, global choice functions, selected Cauchy limits, quotient completion equality, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier locality, mesh monotonicity, cover soundness, selector stability, transport, replay, provenance, and ledger refusal for BEDC.Derived.FiniteNetSelectionPrincipleUp.

Chapter 9079

A Concrete Naming Certificate for $\mathsf{DyadicReciprocalNewton}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ packet records the finite Newton reciprocal route used when a positive dyadic source supplies an approximate inverse to Real and Completion consumers. It sits over the dyadic arithmetic rows of chapter 291, the nonnegative tolerance route of chapter 293, and the interval-enclosure surface of chapter 306. The packet exposes a positive source, a displayed reciprocal iterate, an interval enclosure, and an exact error budget before any real-facing inverse language is consumed; it does not assert host division, quotient equality, selected inverse, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9079.1 (Dyadic reciprocal Newton carrier)**.** *A* $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, P, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, carrying the dyadic input, its displayed positivity witness, and the arithmetic rows used by the Newton step.*
2. *P is the reciprocal-iterate row, recording the current dyadic approximation and the displayed update*
$$p \longmapsto p(2 - dp)$$
inside the dyadic arithmetic ledger.
3. *I is the* $\mathsf{DyadicInterval}^{\uparrow}$ *enclosure row, recording the interval in which the accepted reciprocal iterate remains confined.*
4. *E is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *error-budget row, recording the displayed residual endpoint and tolerance ledger for the expression $1 - dp$.*
5. *H records componentwise* hsame *transport for the source, iterate, interval, error budget, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay for the finite route from the source row through the reciprocal step, interval enclosure, and error-budget readback.*
7. *Q records* Pkg *provenance over* $\mathsf{DyadicReciprocalNewton}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicReciprocalNewton}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, P, I, E, H, C, Q, N. It has no coordinate for host division, quotient rational equality, a selected inverse, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9079.2 (Dyadic reciprocal Newton NameCert obligations)**.** *For every accepted* $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ *carrier*

$$R = (D, P, I, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicReciprocalNewton}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row D;*
2. *positive-source visibility before the reciprocal iterate row P is read;*
3. *reciprocal-step exposure for the displayed update $p \mapsto p(2 - dp)$ inside the dyadic arithmetic ledger;*
4. *enclosure preservation through the* $\mathsf{DyadicInterval}^{\uparrow}$ *row I;*
5. *error-budget exactness through the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *row E for the displayed residual;*
6. *componentwise* hsame *transport through H;*
7. *displayed* Cont *replay through C;*
8. Pkg *provenance through Q;*
9. *local naming through N.*

Thus the packet supplies a finite dyadic reciprocal-iteration certificate, not a host inverse operation, quotient rational equality, selected inverse, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9079.1. Its rows display the positive source, reciprocal iterate, interval enclosure, error budget, transport, replay, provenance, and local naming data.

Read the rows in dependency order. The source row D exposes positivity and dyadic arithmetic before the reciprocal iterate row P is consumed. The interval row I then records enclosure preservation, and the error row E records the residual budget through the existing absolute-value route.

Finally apply H, C, Q, N. These rows transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for host division, quotient equality, selected inverse, trichotomy, countable choice, or ambient completeness, none can be exported by the local certificate. □

Theorem 9079.3 (Dyadic reciprocal Newton quadratic-error route)**.** *Every Real-facing inverse consumer reading an accepted* $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ *packet must first pass through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{DyadicAbsoluteValue}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ D,\ P,\ I,\ E,\ H,\ C,\ Q,\ N.$$

In particular, each accepted Newton reciprocal read factors through the dyadic source arithmetic, residual absolute-value budget, and interval enclosure rows before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *inverse consumer can use the resulting approximation.*

Proof. Begin with the source row D. It is the only positive dyadic input row in the carrier, so the reciprocal step cannot be read before the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic and positivity witness are visible.

Next read the error and enclosure rows. The residual budget is exposed through E, the $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ row attached to the displayed residual, and the accepted iterate remains bounded by the $\mathsf{DyadicInterval}^{\uparrow}$ enclosure row I.

Only after those rows are displayed may a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer attach inverse terminology to the approximation. The transport, replay, provenance, and local naming rows H, C, Q, N preserve the same route and add no hidden host division, quotient equality, selected inverse, trichotomy, countable choice, or ambient completeness data. □

Falsifiable prediction: An accepted $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ read is refuted by any Real-facing inverse consumer that consumes the reciprocal iterate before displaying the positive $\mathsf{DyadicRatCore}^{\uparrow}$ source row, the $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ residual budget, the $\mathsf{DyadicInterval}^{\uparrow}$ enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, dyadicabsolutevalue, dyadicinterval, regseqrat, real: none is bijective to $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ because this packet jointly carries a positive dyadic source, a reciprocal iterate, an interval enclosure, a residual error budget, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicReciprocalNewton}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicReciprocalNewton}^{\uparrow}$ packet is generated from a positive DyadicRatCore source row, a reciprocal-iterate row, a DyadicInterval enclosure row, a DyadicAbsoluteValue residual-budget row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicReciprocalNewton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9079.1, Definition 9079.2, and Definition 9079.3 seed the finite constructive reciprocal-iteration route over source, iterate, enclosure, error budget, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicReciprocalNewton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host division, quotient equality, selected inverse, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-source visibility, reciprocal-step exposure, enclosure preservation, error-budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.DyadicReciprocalNewto

Chapter 9080

A Concrete Naming Certificate for $\mathsf{SerreSwanFiniteProjective}^{\uparrow}$

Following Serre and Swan, the $\mathsf{SerreSwanFiniteProjective}^{\uparrow}$ packet records the finite bridge by which a vector-bundle row over a compact base is read as a finite-projective module row, and conversely by which displayed finite-projective module data supplies the local vector-bundle surface consumed by BEDC readers. It sits between the vector-bundle certificate of chapter 210, the module carrier of chapter 46, and the scalar ring source of chapter 43. The packet names only displayed section rows, idempotent matrix rows, local trivialization rows, transport, replay, provenance, and local naming; it does not import a host equivalence of categories, selected global bases, quotient module equality, or ambient compactness outside the carried rows.

Definition 9080.1 (Serre–Swan finite-projective carrier)**.** *A* $\mathsf{SerreSwanFiniteProjective}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, R, M, E, V, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact-base naming row over which the section and vector-bundle reads are scoped.*
2. *R is the scalar* $\mathsf{Ring}^{\uparrow}$ *row for continuous scalar functions on the displayed base.*
3. *M is the* $\mathsf{Module}^{\uparrow}$ *row carrying the finite-projective section module with its displayed scalar action.*
4. *E is the idempotent or finite-summand ledger witnessing finite projectivity through matrix rows over R.*
5. *V is the* $\mathsf{VectorBundle}^{\uparrow}$ *row carrying fibres, local trivializations, transition maps, and overlap checks.*
6. *L is the local section ledger, displaying the finite windows in which module generators read as bundle sections and bundle sections read back as module elements.*
7. *Q is the comparison row tying the projective-module idempotent surface to the vector-bundle transition surface.*
8. *H records componentwise* hsame *transport for the base, scalar ring, module, idempotent, vector-bundle, section, comparison, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Serre–Swan request through $X, R, M, E, V, L, Q, H, P, N$.*

10. *P is the* Pkg *provenance over* SerreSwanFiniteProjective$^{\uparrow}$, VectorBundle$^{\uparrow}$, Module$^{\uparrow}$, Ring$^{\uparrow}$, Mat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{SerreSwanFiniteProjective}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, R, M, E, V, L, Q, H, C, P, N$. It has no coordinate for a host category equivalence, a selected basis, quotient equality of modules, an ambient sheaf of sections, or external compactness data.

Theorem 9080.2 (Serre–Swan finite-projective NameCert obligations)**.** *For every accepted* SerreSwanFiniteProjecti *carrier*

$$S = (X, R, M, E, V, L, Q, H, C, P, N),$$

the local NameCert$_{\text{SerreSwanFiniteProjective}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed base row, scalar ring row, finite-projective module row, idempotent ledger, vector-bundle row, local section ledger, comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the Serre–Swan handoff must expose $X, R, M, E, V, L, Q, H, C, P, N$ before reading either the finite-projective module side or the vector-bundle side as public.*

Proof. Project the carrier of Definition 9080.1. The finite packet contains only the base, ring, module, idempotent, vector-bundle, section, comparison, transport, replay, provenance, and local naming rows.

The module-facing read is routed through R, M, E. The scalar ring row supplies the displayed coefficients, the module row supplies the carrier and scalar action, and the idempotent ledger supplies the finite-projective witness rather than a selected basis or quotient-module equality.

The bundle-facing read is routed through V, L, Q. The vector-bundle row exposes fibres and transition data, the local section ledger records the finite section windows, and the comparison row is the only bridge between those windows and the finite-projective module surface.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since no host category equivalence, ambient section sheaf, external compactness assertion, or selected global basis is present in the carrier, the listed rows exhaust the public NameCert obligations. □

Closure status for NameCert$_{\text{SerreSwanFiniteProjective}^{\uparrow}}$.

Constructive story: A SerreSwanFiniteProjective$^{\uparrow}$ packet is generated from a displayed base row, scalar Ring row, finite-projective Module row, idempotent ledger, VectorBundle row, local section ledger, comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SerreSwanFiniteProjective}^{\uparrow}}$, seed)

Scope closed: Definition 9080.1 and Definition 9080.2 seed the finite Serre–Swan handoff between finite-projective module rows and vector-bundle rows.

Formal verification: FormalStatus(NameCert$_{\text{SerreSwanFiniteProjective}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host equivalence of categories, selected global bases, quotient module equality, ambient sheaf semantics, external compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, idempotent-ledger exposure, vector-bundle section handoff, comparison-row exactness, transport, replay, provenance, and local naming for BEDC.Derived.SerreSwanFiniteProjectiveUp.

Chapter 9081

A Concrete Naming Certificate for $\mathsf{LocatedApartnessSplit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedApartnessSplit}^{\uparrow}$ packet records the finite branch evidence by which a located real and a displayed apartness witness are routed to Real and Completion consumers without importing constructive trichotomy. It sits over the located-real carrier of **??**, the positive-separation surface of chapter 304, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a guarded positive or negative branch split for already displayed apartness data; it is not a selected sign decision, not a quotient equality of real names, not countable choice, not ambient $\mathsf{Real}^{\uparrow}$ completeness, and not Lean verification.

Definition 9081.1 (Located apartness split carrier)**.** *A* $\mathsf{LocatedApartnessSplit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, A, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *source row whose located interval data fixes the endpoint presentation under inspection.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *branch-evidence row, displaying either the positive-side or negative-side separation witness from the named comparison boundary.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the branch radius, endpoint comparison arithmetic, and displayed tolerance budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window on which the branch evidence is inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite window and dyadic ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, A, D, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the located-real source, apartness branch evidence, dyadic ledger, window, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the split request through the located source, branch evidence, dyadic tolerance ledger, finite window, regular rational readback, Real seal, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedApartnessSplit}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedApartnessSplit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $L, A, D, W, Q, E, H, C, P, N$. It has no coordinate for trichotomy, total sign on $\mathsf{Real}^{\uparrow}$*, quotient equality, countable choice, a selected branch independent of the apartness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9081.2 (Located apartness split NameCert obligations)**.** *For every accepted* $\mathsf{LocatedApartnessSplit}^{\uparrow}$ *carrier*

$$S = (L, A, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessSplit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located-real source, apartness branch evidence, dyadic tolerance ledger, finite window, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *located source exposure through L, requiring the branch request to begin from the displayed* $\mathsf{LocatedReal}^{\uparrow}$ *interval data rather than from a host real value;*
3. *branch evidence exposure through A, requiring the positive or negative side to be read as already displayed* $\mathsf{ApartnessReal}^{\uparrow}$ *evidence;*
4. *dyadic tolerance exactness through D, requiring the branch radius and comparison arithmetic to be the rows used by the split;*
5. *window and readback routing through W, Q, E, requiring the Real-facing seal to be downstream of the finite stream window and regular rational readback;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N for the same rows;*
7. *refusal of trichotomy, selected sign decisions, quotient real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as certificate data.*

No located apartness split obligation is stored outside $L, A, D, W, Q, E, H, C, P, N$.

Proof. Project the carrier of Definition 9081.1. The accepted packet exposes exactly the located source row L, the apartness evidence row A, the dyadic tolerance ledger D, the finite observation window W, the readback row Q, the Real seal E, and the structural rows H, C, P, N.

Read the mathematical rows in dependency order. The located source L fixes the displayed endpoint presentation, and the apartness row A supplies one already evidenced positive or negative branch. The dyadic ledger D then fixes the radius and comparison arithmetic attached to that evidence.

The completion-facing route is finite. The stream window W and regular rational readback Q are the only observation rows available before the Real seal E is reached.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since no row contains trichotomy, a selected sign decision, quotient equality, countable choice, or completeness, those principles cannot become local NameCert obligations. □

Theorem 9081.3 (Located apartness split branch extraction)**.** *Every branch extracted from an accepted* $\mathsf{LocatedApartnessSplit}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{LocatedReal}^{\uparrow}, \quad \mathsf{ApartnessReal}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}.$$

The extracted branch is positive or negative only relative to the apartness row A and the tolerance ledger D. It does not decide a sign for arbitrary $\mathsf{Real}^{\uparrow}$ *names and does not choose between equality and apartness.*

Proof. Start with the located source row L. It supplies the named endpoint presentation and interval data, so any branch read is tied to the displayed located-real packet.

Next read the apartness row A. The branch side is already present as positive or negative separation evidence; the split packet does not compute a side by trichotomy.

The dyadic and finite-window rows then determine the scope of the branch. The ledger D supplies the tolerance radius and comparison arithmetic, while W and Q supply the finite observation and regular rational readback used by the same evidence.

Thus branch extraction is row projection from L, A, D, W, Q followed by the structural nonescape rows. Without those rows, the packet has no branch to export; with them, it exports only the guarded branch stated above. □

Theorem 9081.4 (Located apartness split Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedApartnessSplit}^{\uparrow}$ *packet must read the terminal seal E only after the located-real source L, apartness branch evidence A, dyadic tolerance ledger D, finite stream window W, regular rational readback Q, componentwise transport H, displayed replay C, provenance P, and local naming row N have been displayed. The handoff supplies a guarded branch certificate over finite located-apartness data, not trichotomy, a selected sign decision, quotient real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Apply Definition 9081.2. The Real-facing seal is one row of the same finite packet, and the certificate obligations require the located source, apartness evidence, dyadic ledger, window, readback, transport, replay, provenance, and local naming rows to be visible.

Apply Definition 9081.3. The only branch available to the consumer is the one that has already passed through L, A, D, W, Q.

The structural rows enforce nonescape. Componentwise transport H, displayed replay C, package provenance P, and local naming N preserve the same finite route at the Real seal boundary. Since the carrier has no coordinate for trichotomy, selected sign decisions, quotient equality, countable choice, or completeness, none of those objects can be exported by the handoff. □

Falsifiable prediction: A consumer refutes a $\mathsf{LocatedApartnessSplit}^{\uparrow}$ packet if it exports a positive or negative branch without displaying the located-real source, apartness branch evidence, dyadic tolerance ledger, finite stream window, regular rational readback, Real seal, componentwise transport, replay, provenance, and local NameCert rows.

Independence witness: locatedreal, apartnessreal, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{LocatedApartnessSplit}^{\uparrow}$ because only this packet binds the located source, apartness branch evidence, dyadic tolerance ledger, finite observation window, regular rational readback, Real seal, transport, replay, provenance, and local naming into one guarded branch carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessSplit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedApartnessSplit}^{\uparrow}$ packet is generated from a LocatedReal source row, an ApartnessReal positive or negative branch-evidence row, a DyadicRatCore tolerance ledger, a finite StreamName window, a RegSeqRat readback row, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartnessSplit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9081.1, Definition 9081.2, Definition 9081.3, and Definition 9081.4 seed the finite located-apartness branch route from displayed source and apartness evidence to Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedApartnessSplit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, selected sign decisions, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, apartness branch exposure, dyadic tolerance exactness, finite-window readback routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedApartnessSplitUp.

Chapter 9082

A Concrete Naming Certificate for $\mathsf{SequentialCompactInterval}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of compact intervals by finite approximation and subsequence extraction, the $\mathsf{SequentialCompactInterval}^{\uparrow}$ packet records the sequential compactness surface for a displayed closed real interval. It consumes the finite-net route of chapter 1808, the terminal real-name surface of chapter 38, the regular rational readback of chapter 301, and the finite observation windows of chapter 126. The packet supplies explicit finite-net and subsequence data for interval sequences; it does not export open-cover compactness, selected subcovers, countable choice, quotient real equality, host subsequence selectors, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9082.1 (Sequential compact interval carrier)**.** *A* $\mathsf{SequentialCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, Q, W, R, A, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located closed interval source row, displaying the endpoint names and interval membership boundary consumed by sequence readers.*
2. *F is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *finite-net row, listing the mesh scale, finite interval net, and coverage checks used to classify sequence tails.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the sequence observations and endpoint comparisons attached to the same interval boundary.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the sequence prefixes, endpoint names, and finite-net membership tests already displayed by I, F, Q.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the interval endpoints and the retained subsequence candidate, available only after the finite windows and regular rational readbacks have been displayed.*
6. *A is the accumulation-cell row, naming a descending finite-net cell chain and a retained infinite-tail witness as finite BEDC data.*
7. *D is the subsequence-decision ledger, recording the finite choices of cells, tail indices, and comparison checks used to produce the displayed subsequence route.*
8. *E is the extraction row, exposing the selected finite prefixes and regular rational convergence budgets for the retained subsequence.*

9. *H records componentwise* hsame *transport for the interval source, finite net, sequence readback, stream windows, real seal, accumulation-cell row, subsequence ledger, extraction row, replay row, provenance row, and local naming row.*
10. *C records displayed* Cont *replay from a sequence request through the interval source, finite-net row, finite observation windows, regular rational readback, real seal, accumulation-cell ledger, and extraction row.*
11. *P is the* Pkg *provenance over* $\mathsf{SequentialCompactInterval}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompactInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, Q, W, R, A, D, E, H, C, P, N$. It has no coordinate for open-cover compactness, unrestricted Heine–Borel compactness, countable choice, host real equality, quotient interval representatives, an external subsequence selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9082.2 (Sequential compact interval NameCert obligations)**.** *For every accepted* SequentialCompactInter *carrier*

$$S = (I, F, Q, W, R, A, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompactInterval}^{\uparrow}}$ *surface consists exactly of located interval admission, finite-net exposure through* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, *sequence readback through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$, *terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, accumulation-cell formation, subsequence-decision recording, extraction-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The* $\mathsf{Compact}^{\uparrow}$ *dependency in P is readable only as a finite-net compatibility row and not as open-cover compactness.*

Proof. Project the accepted carrier. The interval source is I, and the only compactness-facing row is the displayed finite-net row F; therefore every compactness read starts with finite interval coverage rather than with an open cover.

The sequence data are finite at each stage. The rows W and Q expose only the inspected stream windows and regular rational readbacks, while R seals those observations as $\mathsf{Real}^{\uparrow}$ data after the finite windows have been displayed.

The extraction rows are also internal to the packet. The accumulation row A records a finite chain of net cells with retained tail witnesses, D records the finite subsequence-decision ledger, and E exposes the resulting subsequence prefixes and convergence budgets.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores an arbitrary open cover, selected subcover, countable-choice functional, quotient representative, host subsequence selector, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Theorem 9082.3 (Sequential compact interval finite-net subsequence extraction)**.** *Every accepted* $\mathsf{SequentialCompactInterval}^{\uparrow}$ *packet supplies subsequence data only through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Concretely, the packet first reads the finite interval net F, then opens the finite sequence windows W, then checks the regular rational readback Q, then reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal R, and only then exposes the accumulation-cell row A, subsequence ledger D, and extraction row E. Thus the route supplies explicit finite-net and subsequence data rather than open-cover compactness, a selected subcover principle, countable choice, or an external subsequence selector.*

Proof. Start with the finite-net row F. Its coverage data come from chapter 1808, so the sequence is inspected against displayed interval cells and mesh ledgers before any extraction row is available.

Next read W and Q. These rows expose finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks for the sequence values and endpoint comparisons attached to the same interval source.

The terminal seal R is downstream of those finite observations. It provides the $\mathsf{Real}^{\uparrow}$ boundary for the retained subsequence candidate but does not create a host real representative or an ambient completeness argument.

Finally inspect A, D, E. The accumulation-cell chain and subsequence ledger are finite BEDC rows, and the extraction row reports the retained prefixes and convergence budgets produced by those rows. No row in the carrier exports open-cover compactness, selected subcovers, countable choice, or an external selector, so the finite-net subsequence route is the only available extraction surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompactInterval}^{\uparrow}$ packet is generated from a located closed interval source, a LocatedClosedIntervalFiniteNet row, finite StreamName windows, RegSeqRat readback, RealUp seals, accumulation-cell data, subsequence-decision rows, extraction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9082.1, Definition 9082.2, and Definition 9082.3 seed the explicit finite-net subsequence route for closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, unrestricted Heine–Borel compactness, selected subcovers, countable choice, quotient real equality, host subsequence selectors, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, finite-net exposure, sequence readback, RealUp seal admission, accumulation-cell formation, subsequence-decision recording, extraction-row exposure, transport, replay, provenance, and local naming for BEDC.Derived.SequentialCompactIntervalUp.

Chapter 9083

A Concrete Naming Certificate for $\mathsf{RegularCauchySumCriterion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ packet records the finite addition criterion that lets two displayed regular rational Cauchy sources produce a summed readback before any $\mathsf{Real}^{\uparrow}$ consumer receives the seal. It sits beside the addition route of chapter 1695, the finite product sufficiency of chapter 3672, and the difference criterion of chapter 1874. The packet names only synchronized windows, dyadic endpoint-sum budgets, regular readback, and seal routing; it does not import host real addition, quotient equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9083.1 (Regular Cauchy sum criterion carrier)**.** *A* $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W_A, W_B, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations are being added.*
2. *W_A and W_B are synchronized* $\mathsf{StreamName}^{\uparrow}$ *window rows for the same precision request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-sum and tolerance-budget row.*
4. *L is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row, read only after A, B, W_A, W_B, D have been displayed.*
5. *R is the summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the finite addition route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R has been displayed.*
7. *H records componentwise* hsame *transport for the source, window, dyadic, route, readback, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from the two source rows through windows, dyadic sum budget, regular addition route, readback, and seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySumCriterion}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySumCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W_A, W_B, D, L, R, E, H, C, P, N$. It has no coordinate for completed-real addition, quotient stream equality, trichotomy, selected representatives, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySumCriterionUp.RegularCauchySumCriterionTasteGate single carrier alignment`

Theorem 9083.2 (Regular Cauchy sum criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ *carrier* $S = (A, B, W_A, W_B, D, L, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySumC}}$ *surface consists exactly of source admission for* A, B*, synchronized finite-window exposure through* W_A, W_B*, dyadic endpoint-sum budgeting through* D*, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route* L*, summed regular readback through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 9083.1. The only analytic rows are the two source rows, the paired finite windows, the dyadic sum budget, the addition route, the regular readback, and the terminal Real seal.

Read the source and window rows first. The rows A, B, W_A, W_B display exactly the finite regular-Cauchy observations available to the sum request.

Next read the dyadic and addition rows. The row D records the endpoint-sum tolerance budget, and the route row L consumes that budget before the summed regular readback R is exposed.

Finally read E, H, C, P, N. These rows seal, transport, replay, source, and name the same finite route. Since no carrier coordinate names host addition, quotient stream equality, trichotomy, choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 9083.3 (Regular Cauchy sum criterion window-budget exactness)**.** *For every accepted* $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ *carrier, the synchronized windows* W_A, W_B *and dyadic row* D *determine the finite sum budget consumed by the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route* L*. Every summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factors through*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto L \longmapsto R.$$

No budget rule is read from host real addition, quotient stream equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The windows W_A, W_B are the only displayed observation windows for the two sources. They synchronize the source rows before any dyadic endpoint-sum row is available.

The dyadic row D is the only tolerance and endpoint-sum ledger in the carrier. The addition route L is downstream of D, and the readback row R is downstream of L.

Transport, replay, provenance, and local naming preserve that same row order. Therefore every summed regular readback follows the displayed finite chain, and no excluded external budget can be consumed. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ read is refuted if a summed RegSeqRatUp or RealUp output is exposed without first displaying two RegSeqRatUp sources, paired StreamNameUp windows, a DyadicRatCoreUp endpoint-sum budget, the RegularCauchyAdditionUp route, transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-addition, regular-cauchy-product-criterion, cauchy-product-criterion, cauchy-difference-criterion: $\mathsf{RegularCauchySumCriterion}$

is not bijective to any listed sibling because it carries the two-source addition criterion, synchronized finite windows, dyadic endpoint-sum budget, summed regular readback, Real seal, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySumCriterion}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchySumCriterion}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore endpoint-sum budget, a RegularCauchyAddition route, a summed RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySumCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 9083.1, Definition 9083.2, and Definition 9083.3 seed the finite addition criterion for regular Cauchy real names through StreamName, RegSeqRat, DyadicRatCore, RegularCauchyAddition, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySumCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real addition, quotient stream equality, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic endpoint-sum budgeting, RegularCauchyAddition routing, RegSeqRat readback, Real seal handoff, transport stability, replay stability, provenance nonescape, and local naming for BEDC.Derived.RegularCauchySumCriterionUp.

Chapter 9084

A Concrete Naming Certificate for $\mathsf{DyadicCauchySandwich}^{\uparrow}$

Bishop and Bridges constructive analysis treats a real presentation as inspectable Cauchy data with explicit finite tolerances rather than as a completed quotient object. The $\mathsf{DyadicCauchySandwich}^{\uparrow}$ packet records the L10 finite-means sandwich in which lower, middle, and upper regular rational rows share one dyadic tolerance ledger before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits beside the squeeze route of chapter 1772, the dyadic enclosure packet of chapter 3627, the tolerance source of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names only displayed finite rows; it does not assert quotient real equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9084.1 (Dyadic Cauchy sandwich carrier)**.** *A* $\mathsf{DyadicCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, M, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower regular Cauchy row, exposed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
2. *M is the middle regular Cauchy row whose observations are squeezed between the lower and upper rows at the displayed dyadic precisions.*
3. *U is the upper regular Cauchy row, exposed through the same finite-window and regular-readback discipline.*
4. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested radii, refinements, and lower-middle-upper containment inequalities.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by every tolerance request for L, M, U.*
6. *R records the three* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks reached from those windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, readable only after the rows L, M, U, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the three Cauchy rows, shared tolerance ledger, windows, readbacks, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay for tolerance admission, window inspection, regular readback, squeeze handoff, real sealing, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchySandwich}^{\uparrow}$, $\mathsf{RegularCauchySqueeze}^{\uparrow}$, $\mathsf{DyadicSandwich}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchySandwich}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, M, U, D, W, R, E, H, C, P, N$. It has no coordinate for quotient real equality, a selected limit, detached trichotomy, countable choice, or ambient real completeness.

Theorem 9084.2 (Dyadic Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchySandwich}^{\uparrow}$ *carrier*

$$T = (L, M, U, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCauchySandwich}^{\uparrow}}$ *surface consists exactly of six finite obligation rows: carrier admission for the displayed triple of regular Cauchy rows; shared dyadic tolerance admission through D; finite window inspection through W; threefold* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R; squeeze-compatible* $\mathsf{Real}^{\uparrow}$ *sealing through E; and transport, replay, provenance, and local ledgering through H, C, P, N. No obligation row exports quotient real equality, selected limits, trichotomy, countable choice, hidden tail selection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The object-facing data are the three regular Cauchy rows L, M, U, the shared tolerance ledger D, the finite windows W, and the regular readback row R.

The squeeze boundary is finite because the middle row M is checked only against the lower and upper rows through the dyadic ledger D and the displayed windows W. This is the shared row consumed by the $\mathsf{RegularCauchySqueeze}^{\uparrow}$ and $\mathsf{DyadicSandwich}^{\uparrow}$ sibling routes.

The terminal row E seals the same displayed packet for $\mathsf{Real}^{\uparrow}$ consumers after the regular readbacks have been exposed. It is not a quotient, a chosen representative, or an ambient completion principle.

Finally H, C, P, N transport, replay, source, and name the packet. Since the carrier lists no coordinate for quotient equality, selected limits, trichotomy, choice, hidden tails, or ambient completeness, the six obligation rows exhaust the local NameCert surface. □

Theorem 9084.3 (Dyadic Cauchy sandwich squeeze route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{DyadicCauchySandwich}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegularCauchySqueeze}^{\uparrow},\ \mathsf{DyadicSandwich}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, a consumer may use the middle row as trapped between lower and upper regular Cauchy rows only after the shared dyadic tolerance ledger, finite windows, regular rational readbacks, and terminal real seal have all been projected.

Proof. Start with the carrier $T = (L, M, U, D, W, R, E, H, C, P, N)$. The rows L, M, U provide the lower, middle, and upper regular Cauchy observations, but the consumer cannot inspect them outside the finite windows W.

Use D to read the shared dyadic tolerance ledger. This is the row that makes the lower-middle and middle-upper containment checks visible to the dyadic sandwich route.

Use R to pass the three windowed observations through $\mathsf{RegSeqRat}^{\uparrow}$. The squeeze route is then the regular Cauchy route named by $\mathsf{RegularCauchySqueeze}^{\uparrow}$, with the dyadic enclosure discipline named by $\mathsf{DyadicSandwich}^{\uparrow}$.

The real-facing read reaches E only after those rows have been projected, and H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Hence the consumer receives a finite sandwich certificate, not quotient real equality, a selected limit, trichotomy, countable choice, hidden tail selection, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCauchySandwich}^{\uparrow}$ read exposes lower, middle, upper Cauchy rows, a shared DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert data before a Real or Completion consumer receives the packet.

Independence witness: regularcauchysqueeze, dyadicsandwich, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicCauchySandwich}^{\uparrow}$ is not bijective to any listed sibling because it binds a three-row regular Cauchy sandwich to one shared dyadic ledger, finite windows, readbacks, seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchySandwich}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchySandwich}^{\uparrow}$ packet is generated from lower, middle, and upper regular Cauchy rows, a shared DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchySandwich}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9084.1, Definition 9084.2, Definition 9084.3, the falsifiable prediction, and the independence witness seed the finite lower-middle-upper Cauchy sandwich route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchySandwich}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, countable choice, hidden tail selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared dyadic tolerance exactness, finite-window readback, squeeze-route stability, real-seal nonescape, transport, replay, provenance, and local ledgering for BEDC.Derived.DyadicCauchySandwichUp.

Chapter 9085

A Concrete Naming Certificate for $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ packet realizes the finite comparison-modulus boundary of section 9238.4. It records two located-real endpoint presentations, one displayed dyadic comparison modulus, finite inspection windows, rational readbacks, and real seals before any comparison-facing consumer reads the packet. The chapter sits beside the located-real comparison packet of chapter 1730, the comparison-modulus sibling of chapter 5813, the tolerance ledger of chapter 291, the finite-window source of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is constructive: it does not export exact trichotomy, LPO, quotient real equality, or selected representative limits.

Definition 9085.1 (Dyadic modulus comparison boundary carrier)**.** *A* $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (A, B, D, M, W, R, E, H, K, P, N)$$

with the following displayed rows:

1. *A is the left* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row, exposed through finite stream windows and regular rational readback.*
2. *B is the right* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row, exposed through the same finite-window discipline.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the comparison request.*
4. *M is the displayed dyadic comparison modulus assigning each requested tolerance to a finite common observation boundary.*
5. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window family selected by M for the two endpoint rows.*
6. *R records the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks reached from those windows.*
7. *E records the two* $\mathsf{Real}^{\uparrow}$ *seals, readable only after A, B, D, M, W, R have been projected.*
8. *H records componentwise* hsame *transport for endpoint rows, tolerance ledger, modulus, windows, readbacks, real seals, replay, comparison boundary, provenance, and local name.*
9. *K records the displayed comparison-boundary row: finite tolerance closeness, finite apartness window, or common-refinement handoff, but not exact trichotomy.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, M, W, R, E, H, K, P, N$. It has no coordinate for arbitrary exact order trichotomy, LPO*, quotient real equality, countable choice, or selected completion limits.*

Theorem 9085.2 (Dyadic modulus comparison boundary NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ *carrier*

$$C = (A, B, D, M, W, R, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}}$ *surface consists exactly of six finite obligation rows: endpoint admission for A and B; dyadic tolerance admission through D; comparison-modulus determinacy through M; paired finite-window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through W and R; Real-seal nonescape through E; and transport, comparison-boundary replay, provenance, and local ledgering through H, K, P, N. No obligation row exports exact trichotomy,* LPO*, quotient real equality, countable choice, or a selected real representative.*

Proof. Project the accepted carrier. The endpoint data are A and B, while D supplies the shared dyadic request against which comparison is read.

The modulus row M turns a requested tolerance into a finite common observation boundary. The paired windows W and readbacks R then expose only the rational observations selected by that displayed modulus.

The comparison row K records the constructive boundary: finite tolerance closeness, finite apartness, or a common-refinement handoff. It does not return a total exact order branch for arbitrary located-real presentations.

The real-seal row E and structural rows H, P, N preserve the same finite packet under sealing, transport, provenance, and local naming. Since no carrier coordinate names exact trichotomy, LPO, quotient equality, choice, or selected representatives, the six obligation rows exhaust the local NameCert surface. □

Theorem 9085.3 (Dyadic modulus comparison common refinement)**.** *Let $C = (A, B, D, M, W, R, E, H, K, P, N)$ be an accepted* $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ *carrier. Every comparison read for the two endpoint rows factors through the common refinement selected by the dyadic modulus row M and the tolerance ledger D. Thus the comparison route may compose finite moduli and read two* $\mathsf{Real}^{\uparrow}$ *seals, but it cannot produce exact trichotomy,* LPO*, quotient real equality, or a selected limit.*

Proof. Read the requested tolerance through D. The dyadic ledger fixes the scale at which the two endpoints may be compared.

Apply the modulus row M. It selects a common finite observation boundary for both endpoints, and the paired StreamName row W exposes exactly those windows.

Use R to read the rational observations and K to record the comparison boundary. When two comparison steps are composed, the common refinement is obtained by taking the displayed finite boundary that serves both modulus rows, as in the finite comparison composition of section 9238.4.

The real seals E are read only after the finite comparison rows have been projected, and H, P, N preserve the same packet under transport, provenance, and local naming. Therefore the route supplies a constructive common refinement, not exact trichotomy, omniscience, quotient equality, choice, or selected limits. □

Falsifiable prediction: Every accepted $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ read exposes two located-real endpoint rows, a dyadic tolerance ledger, displayed comparison modulus, paired StreamName windows, paired RegSeqRat readbacks, two RealUp seals, comparison-boundary row, hsame transport, Pkg provenance, and local NameCert data before a comparison consumer receives the packet.

Independence witness: locatedreal, located-real-comparison, located-real-comparison-modulus, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ is not bijective to any listed sibling because it binds two endpoint rows to one displayed dyadic comparison modulus, paired finite windows, paired readbacks, Real seals, comparison-boundary replay, transport, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$ packet is generated from two located-real endpoint rows, a DyadicRatCore tolerance ledger, a displayed comparison modulus, paired StreamName windows, paired RegSeqRat readbacks, two RealUp seals, hsame transport, a comparison-boundary row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9085.1, Definition 9085.2, Definition 9085.3, the falsifiable prediction, and the independence witness seed the finite dyadic comparison-modulus boundary for located-real reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close exact trichotomy, LPO, quotient real equality, countable choice, selected real representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic tolerance admission, comparison-modulus determinacy, finite-window readback, common-refinement stability, real-seal nonescape, transport, provenance, and local ledgering for BEDC.Derived.DyadicModulusComparisonBoundaryUp.

Chapter 9086

A Concrete Naming Certificate for $\mathsf{GaussianElimination}^{\uparrow}$

Following Gaussian and standard finite-dimensional linear algebra, the $\mathsf{GaussianElimination}^{\uparrow}$ packet records field-based row reduction as a finite pivot, normalization, and elimination ledger over displayed $\mathsf{Mat}^{\uparrow}$ and $\mathsf{Field}^{\uparrow}$ rows. It sits beside the fraction-free determinant route of chapter 7715: Bareiss elimination records determinant-preserving updates over ring and polynomial rows, while this packet names the finite field-division surface used by row-reduction consumers. The packet exports no ambient quotient matrix equality, no selected solution set, no rank-nullity theorem, no Smith normal form, and no host proof that every matrix reaches a canonical echelon form.

Definition 9086.1 (GaussianElimination carrier)**.** *A* $\mathsf{GaussianElimination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, M, I, P, N, L, Z, R, H, C, K, Q)$$

with the following displayed rows:

1. *F is the consumed* $\mathsf{Field}^{\uparrow}$ *scalar row, including the displayed zero, one, addition, multiplication, negation, and inverse reads used by the ledger.*
2. *M is the consumed* $\mathsf{Mat}^{\uparrow}$ *row, including the finite row and column index sets and the initial entry readbacks.*
3. *I lists the pivot-search windows inspected by the packet and the finite row-swap rows used before each pivot is normalized.*
4. *P records the chosen pivot entries and their nonzero-field witnesses.*
5. *N records the pivot-normalization rows, including the displayed inverse read through F and the row-scaling operation on M.*
6. *L records the elimination rows that cancel each displayed nonpivot entry using finite field operations.*
7. *Z records the zero-row and terminal echelon-window checks obtained from the finite ledger.*
8. *R is the row-reduction readback row exported to downstream finite linear-algebra consumers.*
9. *H records componentwise* hsame *transport for F, M, I, P, N, L, Z, R and the local naming rows.*
10. *C records displayed* Cont *replay from a row-reduction request through the scalar, matrix, pivot, normalization, elimination, terminal-window, and readback rows.*

11. *K is the* Pkg *provenance over* $\mathsf{GaussianElimination}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *Q is the local* $\mathsf{NameCert}_{\mathsf{GaussianElimination}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an unlogged pivot inverse, quotient equality of matrices, arbitrary solution-space selection, rank theory, or a host canonical-form theorem.

Theorem 9086.2 (GaussianElimination NameCert obligations)**.** *For every accepted* $\mathsf{GaussianElimination}^{\uparrow}$ *carrier* $G = (F, M, I, P, N, L, Z, R, H, C, K, Q)$*, the local* $\mathsf{NameCert}_{\mathsf{GaussianElimination}^{\uparrow}}$ *surface consists exactly of carrier admission, field-source exposure, matrix-source exposure, finite pivot-window exposure, pivot nonzero evidence, pivot normalization, elimination-row exposure, terminal echelon-window readback, componentwise transport, replay, provenance, local naming, and refusal of unlogged inverses, quotient matrix equality, selected solution sets, rank theory, Smith normal form, and host canonical-form theorems.*

Proof. Project the carrier of Definition 9086.1. The source rows F and M must be read before any pivot operation is available.

Inspect the finite ledger rows I, P, N, L, Z, R. Pivot windows and swaps are displayed in I, nonzero pivot evidence is stored in P, normalization reads field inverses through N, elimination rows are listed in L, and terminal echelon-window checks are recorded in Z before the row-reduction readback R is exported.

The rows H, C, K, Q transport, replay, source, and name the same finite route. Since no additional coordinate is present, the packet cannot export an unlogged inverse, quotient matrix equality, selected solution set, rank theorem, Smith normal form, or host canonical-form theorem. □

Theorem 9086.3 (GaussianElimination Bareiss row-reduction handoff)**.** *The* $\mathsf{GaussianElimination}^{\uparrow}$ *packet and the* $\mathsf{BareissFractionFreeElimination}^{\uparrow}$ *packet share the displayed matrix-ledger boundary at* $\mathsf{Mat}^{\uparrow}$*, but they expose different finite routes. A Gaussian row-reduction consumer must read*

$$\mathsf{Field}^{\uparrow},\ \mathsf{Mat}^{\uparrow} \longmapsto I, P, N, L, Z, R \longmapsto H, C, K, Q,$$

whereas the Bareiss determinant route of chapter 7715 reads the fraction-free pivot, divisibility, branch, and determinant handoff rows. Thus the Gaussian packet may hand a finite row-reduction read to matrix consumers without claiming the fraction-free determinant exactness of Bareiss, and the Bareiss packet may hand determinant data to $\mathsf{Determinant}^{\uparrow}$ *without claiming arbitrary field-based Gaussian elimination.*

Proof. Start with the shared matrix boundary. Both packets consume a displayed $\mathsf{Mat}^{\uparrow}$ row, so downstream consumers can compare that source row through ordinary hsame transport and Pkg provenance.

Separate the scalar routes. The Gaussian packet reads field inverses through F and pivot normalization through N, while the Bareiss packet reads ring rows, previous-pivot denominators, exact-divisibility witnesses, and determinant exactness rows. These rows are not interchangeable.

The handoff is therefore a dependency comparison, not an identification. The Gaussian route exports finite row-reduction readback through R, and the Bareiss route exports determinant-facing fraction-free data through its determinant handoff. Neither packet imports the other's refused claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaussianElimination}^{\uparrow}}$.

Constructive story: A $\mathsf{GaussianElimination}^{\uparrow}$ packet is generated from FieldUp scalar rows, MatrixUp entry rows, finite pivot-window rows, pivot nonzero witnesses, pivot-normalization rows, elimination rows, terminal echelon-window rows, row-reduction readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaussianElimination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9086.1, Definition 9086.2, and Definition 9086.3 seed the finite field-based row-reduction route over displayed matrix, field, pivot, normalization, elimination, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaussianElimination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient matrix equality, selected solution sets, rank-nullity, Smith normal form, Bareiss determinant exactness, host canonical echelon form, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pivot-window exposure, pivot nonzero evidence, normalization exactness, elimination replay, terminal-window readback, structural transport, provenance, and consumer nonescape for BEDC.Derived.GaussianEliminationUp.

Chapter 9087

A Concrete Naming Certificate for $\mathsf{SternBrocotRealCompletion}^{\uparrow}$

Following Stern and Brocot's mediant enumeration and the Hardy–Wright continued-fraction treatment of rational bracketing, the $\mathsf{SternBrocotRealCompletion}^{\uparrow}$ packet reconstructs a real-completion route from finite rational intervals. It sits beside the tree carrier of chapter 1704 and the expansion carrier of chapter 6738, then hands its regular refinement stream to chapter 291, chapter 301, and chapter 38. The packet records only finite bracketing, mediant refinement, dyadic tolerance, stream handoff, transport, replay, provenance, and local naming rows; it does not import Dedekind cuts, quotient real equality, selected limits, completeness, or choice.

Definition 9087.1 (Stern-Brocot real-completion carrier)**.** *A* $\mathsf{SternBrocotRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is a finite* $\mathsf{SternBrocot}^{\uparrow}$ *bracket row, naming adjacent rational endpoints and the finite path that generated them.*
2. *M is the mediant-refinement row, recording the next finite bracket rows obtained by the Stern–Brocot mediant rule.*
3. *T is the dyadic tolerance row, bounding each active bracket width by a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational-core readback used to compare bracket widths and tolerances.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *stream row obtained by reading the bracket representatives through the dyadic tolerance schedule.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row receiving the regular rational stream as a completion candidate.*
7. *H records componentwise* hsame *transport for brackets, mediant refinements, tolerances, dyadic rational rows, regular rational streams, Real seals, provenance, and local names.*
8. *C records displayed* Cont *replay from finite bracket data through mediant refinement, dyadic tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *stream handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SternBrocotRealCompletion}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{ContinuedFractionReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SternBrocotRealCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $B, M, T, D, R, E, H, C, P, N$. It contains no coordinate for Dedekind cuts, quotient real equality, selected limiting points, completeness of $\mathsf{Real}^{\uparrow}$*, host real arithmetic, or choice.*

Theorem 9087.2 (Stern-Brocot real-completion NameCert obligations)**.** *For every accepted* SternBrocotRealComp *carrier*

$$S = (B, M, T, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SternBrocotRealCompletion}^{\uparrow}}$ *surface consists exactly of Stern–Brocot bracket exposure through B, mediant refinement exposure through M, dyadic tolerance exposure through T,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *stream handoff through R,* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9087.1. Rows B, M, T, D, R, E are the completion-facing data, while H, C, P, N are the structural certificate rows.

The rational bracketing route begins with B and M. The Stern–Brocot row supplies adjacent rational endpoints, and the mediant row records the finite refinement step without reading an infinite path as a completed object.

The tolerance route then runs through T and D. The dyadic row bounds bracket widths by finite rational data, so the regular stream handoff has a displayed tolerance source rather than an ambient real limit.

Finally the handoff through R and E presents the regular rational stream to the $\mathsf{Real}^{\uparrow}$ seal. The transport, replay, provenance, and local naming rows preserve this finite route, so no Dedekind cut, quotient equality, selected limit, completeness principle, or choice coordinate is available. □

Theorem 9087.3 (Stern-Brocot real-completion sibling route)**.** *Every public read of an accepted* $\mathsf{SternBrocotRealCompletion}^{\uparrow}$ *packet factors through the sibling rows named by the carrier: the* $\mathsf{SternBrocot}^{\uparrow}$ *bracket row, the continued-fraction bracketing discipline of* $\mathsf{ContinuedFractionReal}^{\uparrow}$*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular stream row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. A consumer cannot read the packet as an isolated completeness theorem or as quotient real equality.*

Proof. Begin with the carrier. Its first two rows, B and M, are finite Stern–Brocot bracket data; they match the rational bracketing route also exposed by the continued-fraction real packet.

Next read the dyadic tolerance and rational-core rows T, D. These rows keep the finite bracket widths explicit before any regular stream can be named.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R and $\mathsf{Real}^{\uparrow}$ seal E then provide the completion-facing handoff. Since the structural rows H, C, P, N replay and source only this route, the consumer receives a rational-refinement certificate rather than a Dedekind cut, quotient equality class, selected limit, or completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SternBrocotRealCompletion}^{\uparrow}$ packet is generated from finite Stern–Brocot bracket rows, mediant refinement rows, dyadic tolerance rows, $\mathsf{DyadicRatCore}^{\uparrow}$ rational-core readback, a $\mathsf{RegSeqRat}^{\uparrow}$ stream handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9087.1, Definition 9087.2, and Definition 9087.3 seed the finite rational-bracketing route from Stern–Brocot refinement to a regular rational stream and a Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SternBrocotRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient real equality, selected limits, completeness, host real arithmetic, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mediant refinement, dyadic tolerance exactness, RegSeqRat stream handoff, Real seal exposure, hsame transport, Cont replay, Pkg provenance, and local NameCert coverage for BEDC.Derived.SternBrocotRealCompletionUp.

Chapter 9088

A Concrete Naming Certificate for $\mathsf{GelfandNaimark}^{\uparrow}$

Following Gelfand and Naimark's representation theorem for C^*-algebras, the $\mathsf{GelfandNaimark}^{\uparrow}$ packet records the finite NameCert surface by which a displayed C^*-algebra source is handed to a Hilbert-space operator-action reader. It sits after the C^*-algebra carrier of chapter 211, the Hilbert carrier of chapter 91, the bounded-linear-operator interface of chapter 1644, the real seal of chapter 38, and the complex scalar carrier of chapter 39. The chapter names only the finite representation packet; it does not import the full standard representation theorem, a selected faithful representation, quotient host equality, spectral completeness, or ambient choice.

Definition 9088.1 (Gelfand–Naimark representation carrier)**.** *A* $\mathsf{GelfandNaimark}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, H, O, R, C, M, T, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CStarAlg}^{\uparrow}$ *source row, exposing the algebra, involution, norm, and C^*-identity surface that the representation reader may inspect.*
2. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *representation-space row, naming the target Hilbert carrier without selecting a host completion.*
3. *O is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *operator-action row, recording the displayed action of source elements on the Hilbert row.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row for norm comparisons, positivity readbacks, and displayed operator bounds.*
5. *C is the* $\mathsf{Complex}^{\uparrow}$ *scalar seal row for complex scalar multiplication, involution compatibility, and inner-product scalar reads.*
6. *M stores the multiplicative and involutive compatibility ledger tying A to the operator-action row O.*
7. *T records componentwise* hsame *transport for $A, H, O, R, C, M, K, P, N$.*
8. *K records displayed* Cont *replay from source-algebra reads through Hilbert-space action, scalar seals, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{GelfandNaimark}^{\uparrow}$, $\mathsf{CStarAlg}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for a host faithful representation, universal representation, quotient operator algebra, selected state space, unrestricted spectrum, ambient Hilbert completion, or choice principle.

Theorem 9088.2 (Gelfand–Naimark NameCert obligations)**.** *For every accepted* $\mathsf{GelfandNaimark}^{\uparrow}$ *carrier* $G = (A, H, O, R, C, M, T, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}$ *surface consists exactly of* C^**-algebra source admission, Hilbert representation-space admission, bounded-operator action exposure, real and complex scalar seal rows, multiplicative and involutive compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Any representation-facing consumer must first display the sibling routes through chapter 211, chapter 91, chapter 1644, chapter 38, and chapter 39. Replacing one of those rows by a host representation theorem, selected state space, quotient operator algebra, unrestricted spectrum, or ambient choice leaves the accepted carrier.

Proof. Project the accepted carrier $G = (A, H, O, R, C, M, T, K, P, N)$. The source row A is the only C^*-algebra input, while H and O display the Hilbert carrier and bounded-operator action consumed by the representation reader.

The scalar rows R and C keep norm, positivity, complex scalar, and inner-product reads on displayed Real and Complex seals. The compatibility row M attaches multiplicative and involutive checks to the same operator-action route.

The structural rows T, K, P, N transport, replay, source, and name this finite inventory. Since no row stores a host faithful representation, universal representation, quotient operator algebra, selected state space, unrestricted spectrum, ambient Hilbert completion, or choice principle, the listed rows exhaust the NameCert obligations. □

Theorem 9088.3 (Gelfand–Naimark representation handoff)**.** *Every representation-facing read from an accepted* $\mathsf{GelfandNaimark}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CStarAlg}^{\uparrow} \longmapsto \mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},\ \mathsf{Complex}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}.$$

For $G = (A, H, O, R, C, M, T, K, P, N)$*, the consumer must expose the* C^**-algebra source row, Hilbert representation-space row, bounded-operator action row, real and complex scalar seal rows, compatibility ledger, transport, replay, provenance, and local naming before any standard-facing representation domain can be compared. The handoff exports no full standard representation theorem, selected faithful representation, quotient host equality, unrestricted spectrum, ambient Hilbert completion, or choice schedule.*

Proof. Begin with a representation-facing request. The carrier gives access to the source only through A, so the consumer first reads the displayed C^*-algebra surface.

Next expose the representation target. The Hilbert row H and bounded-operator row O provide the finite action surface, while M records the multiplicative and involutive compatibility that the action may claim.

The scalar checks then pass through R and C. These rows keep norm, positivity, complex scalar, and inner-product reads on the displayed scalar seals before any standard-facing comparison is attempted.

Finally T, K, P, N preserve the same route under transport, continuation replay, provenance, and local naming. The absent host representation theorem, selected faithful representation, quotient equality, unrestricted spectrum, ambient completion, and choice data cannot be recovered from the packet. □

Falsifiable prediction: An accepted $\mathsf{GelfandNaimark}^{\uparrow}$ read is refuted if a representation-facing consumer obtains a standard-facing representation comparison without displaying the CStarAlg source row, Hilbert representation row, BoundedLinearOperator action row, Real and Complex scalar seals, compatibility ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cstaralgebra, hilbert, boundedlinearoperator, real, and complex: none is bijective to $\mathsf{GelfandNaimark}^{\uparrow}$ because only this packet jointly records a C^*-algebra source, Hilbert representation carrier, bounded-operator action, scalar seal rows, compatibility ledger, transport, replay, provenance, and local NameCert naming as one representation surface.

Closure status for $\mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}$.

Constructive story: A $\mathsf{GelfandNaimark}^{\uparrow}$ packet is generated from a CStarAlg source row, Hilbert representation-space row, BoundedLinearOperator action row, Real and Complex scalar seals, multiplicative and involutive compatibility ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9088.1, Definition 9088.2, and Definition 9088.3 seed the finite representation handoff from CStarAlg rows through Hilbert and BoundedLinearOperator rows to Real and Complex scalar seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GelfandNaimark}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full standard representation theorem, selected faithful representation, quotient host equality, unrestricted spectrum, ambient Hilbert completion, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Hilbert and bounded-operator action exposure, scalar seal compatibility, multiplicative and involutive exactness, transport, replay, provenance, and local naming for BEDC.Derived.GelfandNaimarkUp.

Chapter 9089

A Concrete Naming Certificate for $\mathsf{UnscentedKalmanFilter}^{\uparrow}$

Following Julier and Uhlmann on the unscented transform, the $\mathsf{UnscentedKalmanFilter}^{\uparrow}$ packet records a nonlinear filtering certificate whose sigma-point generation, propagation, covariance ledger, and update row are all displayed as finite BEDC data before a downstream estimator reads the filter. It sits beside the continuous-time $\mathsf{KalmanBucyFilter}^{\uparrow}$ packet of chapter 6360 and the finite Gramian ledger of $\mathsf{ObservabilityGramian}^{\uparrow}$ in chapter 5787. Its carrier is read through the $\mathsf{Mat}^{\uparrow}$ rows of chapter 49, the $\mathsf{VecSpace}^{\uparrow}$ rows of chapter 47, the $\mathsf{ProbSpace}^{\uparrow}$ rows of chapter 184, and the $\mathsf{RandomVar}^{\uparrow}$ rows of chapter 185. The packet names only the finite unscented-filter ledger; it does not claim an ambient stochastic process, analytic convergence theorem, selected nonlinear model class, or optimality theorem.

Definition 9089.1 (UnscentedKalmanFilter carrier)**.** *An* $\mathsf{UnscentedKalmanFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, \mu, \Sigma, S, W, F, G, \widehat{\mu}, \widehat{\Sigma}, Z, K, R, E, H, C, P, N)$$

with the following visible rows:

1. *X and Y are the state and observation rows, each displayed through* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *coordinates;*
2. *μ and Σ are the prior mean and covariance rows read through* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{RandomVar}^{\uparrow}$ *provenance;*
3. *S is the finite sigma-point row and W is the corresponding weight row;*
4. *F is the propagated state-sigma row and G is the propagated observation-sigma row;*
5. *$\widehat{\mu}$ and $\widehat{\Sigma}$ are the predicted mean and covariance rows, and Z, K, R are the innovation, gain, and residual-ledger rows;*
6. *E records sibling readback through the* $\mathsf{KalmanBucyFilter}^{\uparrow}$ *and* $\mathsf{ObservabilityGramian}^{\uparrow}$ *surfaces without identifying the nonlinear sigma-point packet with either sibling;*
7. *H records componentwise* hsame *transport of state, observation, covariance, sigma-point, weight, propagation, innovation, gain, and residual rows;*
8. *C records displayed* Cont *replay for accepted prediction and update routes;*

9. *P is the* Pkg *provenance naming* $\mathsf{UnscentedKalmanFilter}^{\uparrow}$, $\mathsf{KalmanBucyFilter}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UnscentedKalmanFilter}^{\uparrow}}$ *row.*

No external nonlinear dynamics oracle, host probability space, selected noise process, analytic approximation bound, or optimality certificate is part of the carrier.

Theorem 9089.2 (UnscentedKalmanFilter NameCert obligations)**.** *For the carrier of Definition 9089.1, the local* $\mathsf{NameCert}_{\mathsf{UnscentedKalmanFilter}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed packet* $U = (X, Y, \mu, \Sigma, S, W, F, G, \widehat{\mu}, \widehat{\Sigma}, Z, K, R, E, H, C, P, N)$*;*
2. *sigma-point classifier stability, saying that the accepted classifier depends only on the displayed* S, W, F, G *rows and their stated state, observation, mean, and covariance sources;*
3. *covariance and update ledger exactness, saying that* $\widehat{\mu}$*,* $\widehat{\Sigma}$*,* Z*,* K*, and* R *are read only through the displayed sigma-point, weight, propagated-observation, innovation, and residual rows;*
4. *componentwise* hsame *transport through* H *for state, observation, covariance, sigma-point, weight, propagation, innovation, gain, residual, provenance, and local naming rows;*
5. *displayed* Cont *replay through* C *for every prediction, update, and sibling-consumer route exported by the packet.*

Proof. Project the accepted carrier of Definition 9089.1. Carrier admission is coordinate inspection: every row used by the certificate is a displayed BHist coordinate of U.

The sigma-point classifier is stable because it reads the finite sigma-point row S, weight row W, and propagated rows F, G together with their displayed state, observation, mean, and covariance sources. Componentwise replacement by hsame-matched rows preserves that finite shape.

The covariance and update ledger is exact by the same projection. The predicted mean, predicted covariance, innovation, gain, and residual rows have no route except through $S, W, F, G, \widehat{\mu}, \widehat{\Sigma}, Z, K, R$.

Finally, H and C are the only transport and replay rows. A downstream Kalman, Gramian, matrix, vector-space, probability, or random-variable consumer must therefore read the same finite packet through the package provenance P and the local name row N, not through an undisplayed stochastic or analytic object. □

Falsifiable prediction: An accepted $\mathsf{UnscentedKalmanFilter}^{\uparrow}$ packet is refuted by any estimator consumer that reads a covariance, gain, innovation, residual, or propagated sigma-point row while bypassing the displayed sigma-point row S, weight row W, covariance ledger $\widehat{\Sigma}$, update rows Z, K, R, transport row H, replay row C, or package provenance P.

Independence witness: kalman-bucy-filter, observability-gramian, matrix, vecspace, probspace: none is bijective to $\mathsf{UnscentedKalmanFilter}^{\uparrow}$ because the packet jointly carries sigma-point generation, weights, nonlinear propagation, covariance prediction, innovation update, sibling readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UnscentedKalmanFilter}^{\uparrow}}$**.**

Constructive story: An $\mathsf{UnscentedKalmanFilter}^{\uparrow}$ packet is generated from finite state, observation, mean, covariance, sigma-point, weight, propagation, innovation, gain, residual, sibling readback, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnscentedKalmanFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9089.1, Definition 9089.2, the displayed falsifiable prediction, and the displayed independence witness seed the finite nonlinear filtering NameCert surface over KalmanBucy, Gramian, matrix, vector-space, probability-space, and random-variable rows.

Formal verification: FormalStatus(NameCert$_{\text{UnscentedKalmanFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host stochastic-process semantics, analytic convergence, nonlinear filtering optimality, or a selected external probability model.

Upgrade path: Obligation closure requires checked carrier admission, sigma-point classifier stability, covariance and update ledger exactness, componentwise transport, replay, package provenance, and local NameCert exhaustion for BEDC.Derived.UnscentedKalmanFilterUp.

Chapter 9090

A Concrete Naming Certificate for $\mathsf{FiniteHellySelection}^{\uparrow}$

Following Helly and Bishop–Bridges, the $\mathsf{FiniteHellySelection}^{\uparrow}$ packet records the finite interval-selection surface needed in the L10 real-completion route. It sits between the dyadic cell machinery of chapter 291, the finite window naming discipline of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the subsequence-facing compactness packet of chapter 1254. The chapter names only finite closed-interval rows and located handoff data; it does not invoke open-cover compactness, arbitrary choice, selected infinite subsequences, quotient real equality, or ambient real completeness.

Definition 9090.1 (FiniteHellySelection carrier)**.** *A* $\mathsf{FiniteHellySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite closed-interval family row. It lists the inspected interval names and their rational endpoint witnesses.*
2. *D is the dyadic cell row. It records the finite tolerance grid read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and keeps each cell boundary as rational data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the bounded observation windows used by the selection request.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It translates the finite window and dyadic cell data into regular rational approximants.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the selected closed-interval witness, read only after I, D, W, Q have been exposed.*
6. *S is the finite selection ledger. It records which inspected interval row supplies the local witness and which finite compatibility checks were used.*
7. *H records componentwise* hsame *transport for interval, dyadic, window, readback, real-seal, selection-ledger, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a consumer request through the interval family, dyadic cell, finite window, regular readback, real seal, selection ledger, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteHellySelection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteHellySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, W, Q, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, selected infinite subsequences, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9090.2 (FiniteHellySelection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteHellySelection}^{\uparrow}$ *carrier*

$$K = (I, D, W, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteHellySelection}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite closed-interval family, dyadic cell grid,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, finite selection ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every compatibility read must expose the finite interval rows I, dyadic tolerance row D, window row W, and regular readback row Q before the selected real seal R is handed to an L10 consumer.*

Proof. Project the carrier of Definition 9090.1. The tuple lists exactly the interval, dyadic, window, readback, real-seal, selection-ledger, transport, replay, provenance, and naming rows.

The selection ledger S records only finite compatibility checks among the displayed interval rows. Thus a consumer cannot replace the ledger by an open-cover compactness argument, a choice function over all intervals, or a selected infinite subsequence.

The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and naming. Since quotient real equality and ambient completeness are not coordinates of the carrier, the NameCert surface exports only the displayed L10 interval-selection packet. □

Theorem 9090.3 (FiniteHellySelection interval handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer reading a selected interval witness from an accepted* $\mathsf{FiniteHellySelection}^{\uparrow}$ *packet must pass through*

$$I, D, W, Q \longmapsto R, S \longmapsto H, C, P, N.$$

The handoff uses the dyadic tolerance source of chapter 291, the finite-window route of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38 before it can feed the compactness-facing consumer of chapter 1254. It supplies a finite closed-interval selection row, not open-cover compactness, arbitrary choice, selected infinite subsequences, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 9090.2. They force every selection read through $I, D, W, Q, R, S, H, C, P, N$.

The dyadic and finite-window rows are read before the regular rational row. This keeps the L10 route finite: the consumer sees only inspected cells and windows, then the regular readback that feeds the terminal real seal.

Finally the handoff to $\mathsf{Real}^{\uparrow}$ or $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumes R and S through the same transport, replay, provenance, and naming rows. No coordinate in the carrier names open-cover compactness, choice over arbitrary families, an infinite subsequence selector, quotient real equality, or ambient completeness, so none can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteHellySelection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteHellySelection}^{\uparrow}$ packet is generated from finite closed-interval rows, dyadic cell rows, StreamName finite windows, RegSeqRat readback rows, terminal RealUp seals, a finite selection ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteHellySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9090.1, Definition 9090.2, and Definition 9090.3 define the finite interval-selection packet and its L10 real-completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteHellySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary choice, selected infinite subsequences, quotient real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-cell compatibility, finite-window readback, selection-ledger stability, RealUp seal handoff, provenance, and local naming for BEDC.Derived.FiniteHellySelectionUp.

Chapter 9091

Buckingham Pi NameCert Construction

Following Buckingham's dimensional-analysis theorem, the $\mathsf{BuckinghamPi}^{\uparrow}$ packet records the finite unit-exponent route by which a displayed physical law is read through dimensionless monomial coordinates. It sits beside the finite vector-space carrier of chapter 47, the linear-map row of chapter 48, the matrix ledger of chapter 49, and the terminal scalar seal of chapter 38. The packet names the traditional Pi-theorem surface as BEDC data; it does not import host units, choose a basis outside the displayed exponent matrix, quotient physical quantities, or assert analytic smoothness of the modeled law.

Definition 9091.1 (Buckingham Pi carrier)**.** *A* $\mathsf{BuckinghamPi}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, M, K, T, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite unit-exponent carrier, listing the base-unit coordinates of every inspected quantity.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *row for the displayed dimension matrix over the same finite coordinate window.*
3. *K is the dimensionless monomial classifier, exposing exactly the nullspace rows whose exponent vectors cancel all base-unit coordinates.*
4. *T is the exponent-transport stability row, recording componentwise* hsame *transport for unit rows, exponent vectors, matrix reads, and monomial classifiers.*
5. *E is the nullspace and exactness ledger, showing that each exported Pi row is read through M and K before any scalar readback is consumed.*
6. *H records the no-host-units non-escape boundary: every accepted read remains inside the displayed unit-exponent packet.*
7. *C is the displayed* Cont *replay row for finite unit windows, matrix rows, nullspace rows, and dimensionless monomial reads.*
8. *P is the* Pkg *provenance over* $\mathsf{BuckinghamPi}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BuckinghamPi}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for host units, quotient dimensions, selected physical constants, analytic regularity, or an ambient real-valued law outside the displayed finite packet.

Theorem 9091.2 (Buckingham Pi NameCert surface)**.** *For every accepted* $\mathsf{BuckinghamPi}^{\uparrow}$ *carrier* $B = (U, M, K, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BuckinghamPi}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite unit-exponent row, matrix readback for the dimension map, dimensionless monomial classification, exponent-transport stability, nullspace and exactness accounting, no-host-units non-escape, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 9091.1. The rows U and M display the finite unit coordinates and the dimension matrix before any monomial read is available.

Read the classifier row K through the nullspace ledger E. A dimensionless monomial is therefore a finite exponent vector whose matrix read is zero in the displayed unit window, not an equality of host physical quantities.

Transport and replay through T and C. These rows preserve the same unit-exponent coordinates and the same classifier row, while H prevents escape to unlisted host units.

Finally package the route through P and N. Since the carrier has no coordinate for host units, quotient dimensions, selected physical constants, or analytic regularity, the listed rows exhaust the NameCert surface. □

Theorem 9091.3 (Buckingham Pi non-escape)**.** *Every consumer of an accepted* $\mathsf{BuckinghamPi}^{\uparrow}$ *packet that reads a dimensionless Pi coordinate must first expose the route*

$$\mathsf{VecSpace}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{Mat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the carrier rows $U, M, K, T, E, H, C, P, N$*. The exported read is a finite nullspace and monomial-classifier packet; it is not a host-unit equation, a quotient-dimension object, a selected physical constant, or an analytic theorem about the modeled system.*

Proof. Begin with the requested Pi coordinate. The finite unit-exponent row U fixes the displayed basis of dimensions, and the matrix row M records the corresponding dimension map.

Use the classifier K and exactness ledger E to read only exponent vectors annihilated by that displayed map. The read is stable under T because transport acts componentwise on the same exponent and unit rows.

The no-host-units row H then blocks any appeal to unlisted dimensions or ambient physical quantities. Replay, provenance, and naming through C, P, N preserve the finite route, so no excluded host object is exported. □

Falsifiable prediction: Any accepted $\mathsf{BuckinghamPi}^{\uparrow}$ read must display the unit-exponent carrier, dimension matrix, dimensionless monomial classifier, exponent-transport row, nullspace exactness ledger, no-host-units boundary, replay, provenance, and local NameCert row before a Pi coordinate is public.

Independence witness: vecspace, linearmap, matrix, real: $\mathsf{BuckinghamPi}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite unit-exponent data, a dimension matrix, nullspace classification, exponent transport, exactness, no-host-units non-escape, replay, provenance, and local naming in one physical-law-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BuckinghamPi}^{\uparrow}}$**.**

Constructive story: A BuckinghamPi$^{\uparrow}$ packet is generated from finite unit-exponent rows, a displayed dimension matrix, dimensionless monomial classifier rows, exponent-transport stability, nullspace exactness, no-host-units non-escape, displayed replay, package provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{BuckinghamPi}\uparrow}$, seed)

Scope closed: Definition 9091.1, Definition 9091.2, and Definition 9091.3 seed the finite Buckingham Pi NameCert surface.

Formal verification: FormalStatus(NameCert$_{\text{BuckinghamPi}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host physical-unit equality, quotient dimensions, selected physical constants, analytic regularity of a modeled law, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dimension-matrix readback, monomial classification, exponent transport, nullspace exactness, no-host-units non-escape, replay, provenance, and local naming for BEDC.Derived.BuckinghamPiUp.

Chapter 9092

Axis-Zeckendorf Running Coordinate NameCert Construction

The $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ packet realizes the phase-shell running-coordinate branch of Definition 9346.62 as finite BEDC data. It sits beside the Zeckendorf phase-shell surface of chapter 8811, the carry-classifier boundary of chapter 567, the hyperbolic observer phase-shell packet of chapter 6319, and the golden projection surface used by phase readouts. The packet separates shell-exponent motion from denominator response: a consumer may read a running coordinate only after the shell exponent, denominator response, readout route, transport, provenance, and local NameCert rows remain visible.

Definition 9092.1 (Axis-Zeckendorf running-coordinate carrier)**.** *An* $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, D, R, T, V, P, N)$$

with the following displayed rows:

1. *S is the shell-exponent row, naming the Zeckendorf phase shell and the finite golden-radius exponent being inspected.*
2. *D is the denominator-response row, recording the fixed-window denominator coordinate without changing the shell exponent.*
3. *R is the readout route row, displaying how the shell exponent and denominator response are consumed by the running-coordinate read.*
4. *T is the transport row, recording componentwise* hsame *stability for shell exponents, denominator responses, readout rows, replay, provenance, and local naming.*
5. *V is the displayed* Cont *replay row for fixed-denominator transport and shell-exponent readback.*
6. *P is the* Pkg *provenance over* $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$, $\mathsf{ZeckendorfPhaseShell}^{\uparrow}$, $\mathsf{ZeckendorfCarryClassifie}$ $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$, $\mathsf{GoldenPhaseProjection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}}$ *row.*

The classifier reads only these seven rows. It has no coordinate for a hidden electroweak running law, an untracked denominator drift, a host real equality of readouts, or a shell change not exposed by S.

Theorem 9092.2 (Axis-Zeckendorf running-coordinate surface)**.** *For every accepted* AxisZeckendorfRunningCoordin *carrier* $Z = (S, D, R, T, V, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}}$ *surface consists exactly of shell-exponent exposure, denominator-response exposure, readout-route exposure, componentwise transport, displayed replay, provenance, and local naming. Fixed denominator transport preserves the shell-exponent row: any read that changes* D *without exposing a new* S *remains in the same shell.*

Proof. Project the carrier of Definition 9092.1. The shell exponent S and denominator response D are separate rows, so a read cannot identify denominator motion with shell motion.

Read the route row R. It consumes S and D in the displayed order and exposes the running coordinate as finite readout data rather than as a host real law.

Transport through T and replay through V. Because transport is componentwise, fixed denominator transport acts on D and the readout route while preserving the shell-exponent row S unless a new shell row is explicitly supplied.

Finally package the route through P and N. The packet therefore exports only the displayed shell-exponent, denominator-response, readout, transport, provenance, and naming surface. □

Theorem 9092.3 (Axis-Zeckendorf fixed-denominator non-escape)**.** *Every fixed-denominator read of an accepted* $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ *packet factors through the same shell-exponent row* S *before it reaches the readout route* R*. The read may alter denominator response data inside* D*, but it cannot change shell exponent, phase-shell provenance, or local NameCert identity without exposing a new accepted carrier row.*

Proof. Start with a fixed-denominator read. The row S names the shell exponent, while D names the denominator response inspected inside that shell.

Use the readout route R and transport row T. Fixed-denominator transport has no coordinate that rewrites S, so the transported read remains attached to the same shell-exponent row.

Replay and source the read through V and P, then name it through N. Since no hidden shell-change coordinate exists in the carrier, every fixed-denominator read remains in the original shell until a new accepted shell-exponent row is displayed. □

Falsifiable prediction: For an accepted $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ packet, fixed denominator transport cannot change the shell-exponent row; any observed shell change must expose a new S row and a new local NameCert read.

Independence witness: zeckendorf-phase-shell, zeckendorf-carry-classifier, hyperbolic-observer-phase-shell, golden-phase-projection: $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries shell exponent, denominator response, readout route, transport, provenance, replay, and local naming as one running-coordinate packet.

Lean's TasteGate surface for $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ must provide Nontrivial and FieldFaithful instances for the seven-row carrier, so downstream consumers can distinguish shell-exponent rows from denominator-response rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}$ packet is generated from shell-exponent rows, denominator-response rows, readout-route rows, componentwise transport, displayed replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9092.1, Definition 9092.2, and Definition 9092.3 seed the finite Axis-Zeckendorf running-coordinate packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxisZeckendorfRunningCoordinate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a hidden electroweak running law, host real equality of readouts, shell changes without displayed shell-exponent rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shell-exponent exposure, denominator-response exposure, readout routing, fixed-denominator non-escape, transport, replay, provenance, local naming, Nontrivial evidence, and FieldFaithful evidence for BEDC.Derived.AxisZeckendorfRunningCoordinateUp.

Chapter 9093

A Concrete Naming Certificate for $\mathsf{SingletonMetric}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{SingletonMetric}^{\uparrow}$ packet records the terminal one-point metric space as finite BEDC NameCert data. It sits at the boundary shared by chapter 4230, chapter 128, chapter 129, and chapter 38: every displayed distance is the zero $\mathsf{Real}^{\uparrow}$ distance, every point-read is the unique point-read, and every classifier comparison is total because there is no second point from which a distance, cover, or Cauchy tail could diverge.

Definition 9093.1 (Singleton metric carrier)**.** *A* $\mathsf{SingletonMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, D, T, H, C, G, N)$$

with the following displayed rows:

1. *P is the unique point row.*
2. *D is the zero-distance row, recording that every displayed distance read has value $0_{\mathsf{Real}^{\uparrow}}$.*
3. *T is the total classifier row: any two accepted point reads are classified as the same singleton point.*
4. *H records componentwise* hsame *transport for the point row, zero-distance row, total classifier, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay for the unique-point metric query.*
6. *G records*

$$\mathsf{Pkg}(S) = \mathsf{Pkg}(\mathsf{SingletonMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{NameCert}).$$

7. *N is the local* $\mathsf{NameCert}_{\mathsf{SingletonMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through P, D, T, H, C, G, N. It has no coordinate for a nonzero distance, a second point, a selected Cauchy limit, an open-cover choice, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9093.2 (Singleton metric NameCert obligations)**.** *For every accepted* $\mathsf{SingletonMetric}^{\uparrow}$ *carrier*

$$S = (P, D, T, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{SingletonMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the unique point row, zero* $\mathsf{Real}^{\uparrow}$ *distance row, total classifier row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The packet therefore presents the terminal one-point* $\mathsf{Met}^{\uparrow}$ *boundary case, with immediate* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *readings only through the same displayed zero-distance and total-classifier rows.*

Proof. Project the carrier of Definition 9093.1. The only point data are the row P, the zero-distance row D, and the total classifier row T.

The metric, completeness, and compactness readings are terminal readings of the same finite surface. Since every point read is P and every distance read is the zero $\mathsf{Real}^{\uparrow}$ row D, a Cauchy request has no independent limit choice and a finite-cover request has no independent cover-selection coordinate.

The structural rows H, C, G, N transport, replay, source, and name exactly this terminal surface. Because the carrier contains no nonzero distance, second point, selected limit, open-cover choice, quotient equality, or ambient-completeness coordinate, none can be exported by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SingletonMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{SingletonMetric}^{\uparrow}$ packet is generated from one point row, zero RealUp distance row, total classifier row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SingletonMetric}^{\uparrow}}$, seed)

Scope closed: Definition 9093.1 and Definition 9093.2 seed the terminal one-point metric packet over the displayed rows P, D, T, H, C, G, N.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SingletonMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close nontrivial metric spaces, nonzero distances, selected Cauchy limits, open-cover choice, quotient equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance exactness, total-classifier stability, hsame transport, Cont replay, Pkg provenance, and local naming for BEDC.Derived.SingletonMetricUp.

Chapter 9094

A Concrete Naming Certificate for $\mathsf{UniformOscillationModulus}^{\uparrow}$

Bishop–Bridges constructive analysis treats uniform control as finite tolerance data rather than as an ambient compactness theorem. The $\mathsf{UniformOscillationModulus}^{\uparrow}$ packet records the finite route by which displayed oscillation windows become a uniform oscillation-control row before any compact-uniform, regular-real, or continuity-modulus consumer reads a seal. It sits beside the oscillation-modulus carrier of chapter 2224, the finite partition surface of chapter 3800, the finite stream names of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only finite oscillation rows, partition cells, stream windows, dyadic budgets, regular readback, transport, replay, provenance, and local naming data.

Definition 9094.1 (Uniform oscillation modulus carrier)**.** *A* $\mathsf{UniformOscillationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (O, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *source row, recording tolerance requests and displayed oscillation bounds.*
2. *F is the* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, recording the finite cells on which the oscillation request is checked.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row for the cell endpoints and sample names.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational bounds produced by the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row, recording dyadic precision requests, local comparison margins, and finite budget transfers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the displayed oscillation, partition, stream, regular-rational, and dyadic rows have been read.*
7. *H records componentwise* hsame *transport for the oscillation, partition, stream, readback, dyadic, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each uniform oscillation request through*

$$O, F, S, R, D, E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{UniformOscillationModulus}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformOscillationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, F, S, R, D, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected limits, quotient streams, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9094.2 (Uniform oscillation modulus NameCert obligations)**.** *For every accepted* UniformOscillationModu *carrier*

$$U = (O, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformOscillationModulus}^{\uparrow}}$ *surface consists exactly of oscillation-modulus admission through O, finite partition exposure through F, stream-window observation through S, regular rational readback through R, dyadic tolerance budgeting through D, terminal real-name sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9094.1. The analytic rows are exactly the oscillation source, finite partition, stream observations, regular rational readback, dyadic budget, and terminal Real seal.

Read O and F first. The source oscillation row supplies the tolerance request, while the finite partition row supplies the cells on which the request is witnessed.

Next read S, R, D, E. Stream windows expose the finite observations, regular rational readback records the numeric bound, the dyadic row stores the budget used by the comparison, and the terminal Real seal is available only after those displayed rows.

Finally read H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for open-cover compactness, selected limits, quotient streams, host real equality, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 9094.3 (Uniform oscillation modulus finite-partition readback)**.** *Every compact-uniform, regular-real, or continuity-modulus consumer read accepted by a* $\mathsf{UniformOscillationModulus}^{\uparrow}$ *packet must factor through*

$\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame,

Equivalently, the consumer may inspect $O, F, S, R, D, E, H, C, P, N$, but it cannot bypass the finite partition and dyadic readback by invoking open-cover compactness, selected limits, quotient streams, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 9094.2. It forces every accepted consumer read to open the oscillation source O, finite partition F, stream observations S, regular rational readback R, dyadic budget D, and terminal seal E before the structural rows are available.

The finite-partition step is carried by F and S. These rows display the cells and observations used by the uniform request, so no consumer can replace them by an open-cover compactness argument or a selected external limit.

The numeric readback is carried by R, D, E. Regular rational bounds, dyadic budgets, and the terminal Real seal provide the only exported route to downstream real-facing packets.

Transport, replay, provenance, and naming through H, C, P, N preserve exactly that displayed route. Therefore compact-uniform, regular-real, and continuity-modulus consumers receive a finite partition readback and no quotient-stream, choice, host-equality, or ambient-completeness principle. □

Falsifiable prediction: An accepted $\mathsf{UniformOscillationModulus}^{\uparrow}$ read is refuted if a compact-uniform, regular-real, or continuity-modulus consumer receives an oscillation-control output without first displaying an OscillationModulus row, a FiniteOscillationPartition row, StreamName observations, RegSeqRat readback, a DyadicRatCore budget, a Real seal, transport, replay, provenance, and local naming rows.

Independence witness: oscillation-modulus, finite-oscillation-partition, streamname, regseqrat, dyadicratcore, real: $\mathsf{UniformOscillationModulus}^{\uparrow}$ is not bijective to any listed sibling because it carries the finite partition readback, dyadic tolerance budget, terminal seal, and uniform consumer handoff in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformOscillationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformOscillationModulus}^{\uparrow}$ packet is generated from an OscillationModulus source row, a FiniteOscillationPartition row, StreamName finite observations, RegSeqRat readback, a DyadicRatCore tolerance budget, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformOscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9094.1, Definition 9094.2, and Definition 9094.3 seed the finite uniform oscillation-control packet for compact-uniform, regular-real, and continuity-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformOscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient streams, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, oscillation-modulus admission, finite partition exposure, stream-window observation, regular rational readback, dyadic budgeting, Real seal handoff, transport stability, replay stability, provenance nonescape, and local naming for BEDC.Derived.UniformOscillationModulusUp.

Chapter 9095

A Concrete Naming Certificate for $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$

Following Bishop and Bridges, continuity on regular Cauchy names is witnessed by an explicit finite modulus that turns a target tolerance into a source-window and dyadic-tolerance budget. The $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$ packet sits between the dyadic arithmetic core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite transformer surface of chapter 1401. It names the regular-Cauchy continuity budget consumed by later completion arguments without asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9095.1 (Regular Cauchy continuity-modulus carrier)**.** *A* $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, E, D, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer row whose source and target regular-name reads are being controlled.*
2. *E is the target-error row, a finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request presented before any source-window read is served.*
3. *D is the dyadic control ledger, recording the source tolerance, target tolerance, and finite comparison rows used by the modulus.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened on the source and target Cauchy names.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic ledger.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after F, E, D, W, R have all been replayed.*
7. *H records componentwise* hsame *transport for transformer, tolerance, dyadic-control, stream-window, regular-readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a target tolerance request through the transformer, dyadic control, finite windows, regular readback, and terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyContinuityModulus}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host function on reals, quotient stream equality, countable choice over source windows, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9095.2 (Regular Cauchy continuity-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$ *carrier* $M = (F, E, D, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyC}}$ *surface is exhausted by carrier admission for the displayed transformer, target-error, dyadic-control, stream-window, regular-readback, real-seal, transport, replay, provenance, and local naming rows; target tolerance exposure through E; dyadic control through D; finite-window accountability through W; regular readback through R; and refusal of host real functions, quotient stream equality, selected representatives, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 9095.1. The tuple $F, E, D, W, R, S, H, C, P, N$ is the only packet available to the local naming row.

The target tolerance first appears as the row E. The dyadic control row D then records the source-side tolerance and comparison data, and the finite stream window row W records exactly which Cauchy-name prefixes are opened before the transformer row F is read.

Regular readback through R and terminal sealing through S are downstream of the same finite route. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve that packet without adding a host function, quotient equality, representative selector, choice schedule, trichotomy, or Real-completeness coordinate. □

Theorem 9095.3 (Regular Cauchy continuity dyadic control)**.** *Every accepted* $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$ *read that is consumed by a* $\mathsf{CauchyContinuousMap}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *route must expose the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *target tolerance E, the dyadic control ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal S is available. The packet therefore supplies finite regular-Cauchy continuity control, not an ambient continuity theorem for completed reals.*

Proof. Start with Definition 9095.2. It places target tolerance exposure, dyadic control, finite-window accountability, and regular readback inside the same local NameCert surface.

The consumer route through $\mathsf{CauchyContinuousMap}^{\uparrow}$ reads the transformer row F only together with the tolerance rows E, D and the finite stream windows W. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is then the visible regular-name readback produced from those displayed rows.

Only after that readback may the terminal seal S be consumed by a $\mathsf{Real}^{\uparrow}$-facing route. Since the carrier of Definition 9095.1 has no row for completed-real continuity, quotient streams, selected representatives, countable choice, trichotomy, or host real equality, the dyadic control route is exactly the finite packet described above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyContinuityModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyContinuityModulus}^{\uparrow}$ packet is generated from a CauchyContinuousMap transformer row, target dyadic tolerance, dyadic control ledger, finite StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9095.1, Definition 9095.2, and Definition 9095.3 seed the finite regular-Cauchy continuity-modulus carrier and its dyadic control route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed-real continuity, quotient stream equality, selected representatives, countable choice over windows, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires binding the transformer, target-error, dyadic-control, finite-window, regular-readback, seal, transport, replay, provenance, and local naming rows to BEDC primitives.

Chapter 9096

A Concrete Naming Certificate for $\mathsf{BirkhoffPolytope}^{\uparrow}$

Following Birkhoff and von Neumann, the $\mathsf{BirkhoffPolytope}^{\uparrow}$ packet records the finite convex-combinatorial object whose points are doubly stochastic matrices and whose displayed vertices are permutation matrices. It sits after the matrix carrier of chapter 49, the polytope carrier of chapter 209, the convex-set surface of chapter 208, the rational and real scalar readbacks of chapter 37 and chapter 38, the finite permutation rows of chapter 116, the finite list carrier of chapter 36, and the permutation-flow decomposition route of chapter 8080. The packet names the finite convex hull and stochastic-row boundary; it is not a host theorem about arbitrary matrix spaces, quotient matrix equality, selected decompositions, linear-programming duality, or compactness by choice.

Definition 9096.1 (Birkhoff polytope carrier)**.** *A* $\mathsf{BirkhoffPolytope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, R, C, V, L, A, D, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *square-matrix row, including the finite row and column index lists on which every entry is read.*
2. *R is the row-stochastic ledger, displaying nonnegative entries and unit row sums through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *scalar rows.*
3. *C is the column-stochastic ledger, displaying the matching unit column sums over the same finite index spine.*
4. *V is the finite permutation-vertex row, listing the* $\mathsf{Permutation}^{\uparrow}$ *matrix rows admitted as vertices of the polytope.*
5. *L is the finite* $\mathsf{List}^{\uparrow}$ *convex-combination ledger, carrying displayed scalar weights, total-one readback, and support membership for the listed vertices.*
6. *A is the* $\mathsf{ConvexSet}^{\uparrow}$ *affine-membership row for the convex hull generated by V and L.*
7. *D is the decomposition handoff row, available only after the stochastic ledgers and the finite vertex list have been displayed; it may cite the route of chapter 8080.*
8. *H records componentwise* hsame *transport for matrix, stochastic, vertex-list, convex-combination, affine-membership, decomposition, provenance, and local naming rows.*

9. *K records displayed* Cont *replay through the same finite convex-combinatorial rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BirkhoffPolytope}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{BirkhoffVonNeumannFinite}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BirkhoffPolytope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, C, V, L, A, D, H, K, P, N$. It has no coordinate for host matrix equality, arbitrary ambient matrix spaces, selected convex decompositions, linear-programming duality, open-cover compactness, Lean verification, or bridge checking.

Theorem 9096.2 (Birkhoff polytope NameCert obligations)**.** *For every accepted* $\mathsf{BirkhoffPolytope}^{\uparrow}$ *carrier*

$$B = (M, R, C, V, L, A, D, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BirkhoffPolytope}^{\uparrow}}$ *surface consists exactly of finite matrix admission, row-stochastic exposure, column-stochastic exposure, permutation-vertex exposure, finite-list convex-combination exposure, convex-set affine membership, decomposition handoff, componentwise transport, displayed replay, package provenance, and local naming. Every consumer that treats a doubly stochastic matrix as a point of the Birkhoff polytope must read M, R, C before V, L, A, D; every consumer that treats the point as a convex combination of permutation matrices must read the finite list and decomposition rows before transport or local naming is public.*

Proof. Project the carrier of Definition 9096.1. The stochastic matrix surface is exactly M, R, C: the finite square matrix row, nonnegative entry reads, and unit row and column sum ledgers.

The vertex and convex-combination surface is V, L, A. The permutation vertices are displayed as finite $\mathsf{Permutation}^{\uparrow}$ matrix rows, the convex weights are displayed as a $\mathsf{List}^{\uparrow}$ ledger with scalar readbacks through $\mathsf{Rat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, and the affine-membership row reads those two surfaces through $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$.

The decomposition handoff D is downstream from both surfaces. When a decomposition-facing consumer needs the Birkhoff–von Neumann finite route, it cites chapter 8080 only after the stochastic rows and vertex-list rows are already visible.

Finally H, K, P, N transport, replay, source, and name the same finite packet. Since the carrier has no row for host matrix equality, arbitrary ambient matrix spaces, selected decompositions, linear-programming duality, compactness by choice, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 9096.3 (Birkhoff polytope finite vertex route)**.** *Every accepted* $\mathsf{BirkhoffPolytope}^{\uparrow}$ *read that exports a convex-combinatorial conclusion factors through the finite route*

$$\mathsf{Mat}^{\uparrow} \longmapsto \mathsf{Permutation}^{\uparrow} \longmapsto \mathsf{List}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{Polytope}^{\uparrow}.$$

The route exposes the doubly stochastic matrix rows, the finite permutation-matrix vertex list, the convex weight ledger, affine membership, and the polytope naming row before any downstream matrix, optimization, or decomposition consumer may inspect the result.

Proof. Start with Definition 9096.2. A convex-combinatorial read cannot bypass the matrix row M, the stochastic ledgers R, C, or the permutation-vertex row V.

Next read the finite list row L. It supplies the displayed weights and vertex support, so the affine row A receives an actual finite convex-combination ledger rather than an external selector.

The $\mathsf{Polytope}^{\uparrow}$ source is present through provenance and the affine-membership row: it names the convex hull surface generated by those vertices, while chapter 8080 may supply the finite decomposition route when a stochastic matrix is consumed as such a convex combination.

The structural rows then preserve exactly that route. No host equality, selected decomposition oracle, linear-programming theorem, or compactness principle is available inside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BirkhoffPolytope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BirkhoffPolytope}^{\uparrow}$ packet is generated from finite MatrixUp rows, row-stochastic and column-stochastic ledgers, finite PermutationUp vertex rows, ListUp convex-combination weights, ConvexSetUp affine membership, PolytopeUp provenance, BirkhoffVonNeumannFiniteUp decomposition handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BirkhoffPolytope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9096.1, Definition 9096.2, and Definition 9096.3 seed the finite convex-combinatorial carrier over matrix, stochastic, permutation-vertex, finite-list, convex-set, polytope, rational, and real scalar rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BirkhoffPolytope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host matrix equality, arbitrary ambient matrix spaces, selected decompositions, linear-programming duality, compactness by choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stochastic-row exposure, permutation-vertex exposure, finite-list convex-combination exactness, affine-membership routing, decomposition handoff, transport, replay, provenance, and local naming for BEDC.Derived.BirkhoffPolytopeUp.

Chapter 9097

A Concrete Naming Certificate for $\mathsf{SchauderProjection}^{\uparrow}$

Following Schauder's projection-theorem tradition in constructive functional analysis, the $\mathsf{SchauderProjection}^{\uparrow}$ packet records a displayed finite-rank approximation route for projection-facing consumers without asserting a global nearest-point selector. It sits after the Hilbert projection boundary of chapter 2356, the Hilbert carrier of chapter 91, the convex subset surface of chapter 208, the complete-metric limit service of chapter 128, and the terminal real-name seal of chapter 38. The packet names a finite Schauder approximation certificate, not a choice-heavy nearest-point map, quotient equality of points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9097.1 (Schauder projection carrier)**.** *A* $\mathsf{SchauderProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, H, V, K, F, E, R, T, C, P, N)$$

with the following displayed rows:

1. *M is the parent* $\mathsf{MetricProjection}^{\uparrow}$ *boundary row, read only as the existing projection-facing endpoint discipline.*
2. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row containing the inner-product, norm, and induced metric surfaces.*
3. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *source row naming the closed convex or affine target surface to which the projection request is confined.*
4. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *convergence-service row used by the approximation window.*
5. *F is the finite-rank approximation ledger: displayed finite subspaces, endpoint witnesses, and approximation tolerances.*
6. *E is the endpoint-candidate row produced by the displayed approximation ledger.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the distance or norm bound read after M, H, V, K, F, E have been displayed.*
8. *T records componentwise* hsame *transport for the projection, Hilbert, convex, complete-metric, finite-rank, endpoint, real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a projection request through finite-rank approximation, endpoint admission, and real-seal readback.*

10. *P is the* Pkg *provenance over* $\mathsf{SchauderProjection}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchauderProjection}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, H, V, K, F, E, R, T, C, P, N$. It has no coordinate for a host projection operator, quotient point equality, selected representatives, countable choice, or ambient real completeness.

Theorem 9097.2 (Schauder projection NameCert obligations)**.** *For every accepted* $\mathsf{SchauderProjection}^{\uparrow}$ *carrier*

$$S = (M, H, V, K, F, E, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchauderProjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the parent metric-projection boundary, Hilbert source, convex target row, complete-metric convergence service, finite-rank approximation ledger, endpoint candidate, terminal real seal, componentwise transport, displayed replay, package provenance, and local naming. Every projection-facing consumer must expose those rows before reading E or R.*

Proof. Project the carrier of Definition 9097.1. The displayed tuple lists every row available to the local certificate, so carrier admission can only expose $M, H, V, K, F, E, R, T, C, P, N$.

Read the analytic rows in order. The parent $\mathsf{MetricProjection}^{\uparrow}$ boundary supplies the nonescape discipline for projection endpoints, while H and V supply the Hilbert and convex source data. The complete-metric row K is then consumed only by the finite-rank approximation ledger F.

The endpoint candidate E and real seal R are downstream of that finite ledger. The structural rows T, C, P, N transport, replay, source, and name the same route, and they add no hidden projection operator or selected representative. □

Theorem 9097.3 (Schauder projection endpoint nonescape)**.** *Every endpoint read exported by an accepted* $\mathsf{SchauderProjection}^{\uparrow}$ *packet factors through*

$$\mathsf{MetricProjection}^{\uparrow},\ \mathsf{Hilbert}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

by way of the displayed rows $M, H, V, K, F, E, R, T, C, P, N$. The exported endpoint is a finite Schauder-approximation candidate only; it cannot be read as a host nearest-point selector, quotient representative, countable-choice selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 9097.2. They force every endpoint consumer to expose the parent projection boundary, Hilbert source, convex target, complete-metric service, finite-rank ledger, endpoint row, and real seal before the local name can be used.

Use the parent nonescape boundary from Definition 2356.3. It permits a projection endpoint only as a displayed BEDC projection-facing row, not as a host projection map or selected representative.

Finally inspect the Schauder carrier. The only approximation data are the finite rows in F, and the only numerical boundary is the real seal R. Since no coordinate stores quotient point equality, countable choice, or ambient completeness, none of those objects can escape through E. □

Theorem 9097.4 (Schauder projection finite approximation route)**.** *Any consumer that uses a* $\mathsf{SchauderProjection}^{\uparrow}$ *packet for a projection-facing approximation must first display the finite-rank ledger F over the Hilbert row H, convex row V, complete-metric service K, and parent metric-projection boundary M. Only after that route is replayed may the consumer inspect the endpoint candidate E and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R.*

Proof. Apply Definition 9097.2. The finite-rank ledger is one of the mandatory certificate rows, and the endpoint and real seal occur only after the source and convergence-service rows have been exposed.

Next apply Definition 9097.3. It keeps the endpoint as a finite approximation candidate, so the consumer cannot replace F by a hidden nearest-point operator or a selected sequence of representatives.

The structural rows T, C, P, N then preserve the same displayed route. Thus the only accepted projection-facing use is the finite approximation route through M, H, V, K, F, E, R. □

Falsifiable prediction: An accepted $\mathsf{SchauderProjection}^{\uparrow}$ consumer that reads an endpoint without displaying the finite-rank ledger F and the parent $\mathsf{MetricProjection}^{\uparrow}$ boundary refutes the carrier surface.

Independence witness: metricprojection, hilbert, convexset, completemetric, real: $\mathsf{SchauderProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite-rank approximation ledger, a parent projection boundary, Hilbert and convex source rows, a complete-metric convergence service, and a terminal real seal while refusing host projection operators and choice-heavy nearest-point selection.

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderProjection}^{\uparrow}$ packet is generated from a MetricProjection boundary row, Hilbert source row, ConvexSet target row, CompleteMetric convergence-service row, finite-rank approximation ledger, endpoint candidate, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9097.1, Definition 9097.2, and Definition 9097.3 seed the finite Schauder projection route through displayed approximation data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host projection operator, nearest-point selection, quotient point equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-rank ledger exactness, endpoint nonescape, real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.SchauderProjectionUp.

Chapter 9098

A Concrete Naming Certificate for $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ packet records the finite merge route by which two displayed regular Cauchy real names are read through one interleaved window and one regular-real tail-equivalence boundary before a $\mathsf{Real}^{\uparrow}$ consumer receives the seal. It sits after the finite stream schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name surface of chapter 38, the even/odd interlacing packet of chapter 5870, and the same-real tail route of chapter 5940. The packet specializes interlacing to the regular-Cauchy merge boundary: it names the finite source rows, shared window, tolerance ledger, tail-equivalence route, and Real seal together, without turning them into quotient streams, selected representatives, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9098.1 (Regular Cauchy interleaving merge carrier)**.** *A* $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, D, I, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular Cauchy tails are merged.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It records the requests at which both sources are inspected before a merge read is served.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails, the interleaved tail, and the terminal seal request.*
4. *I is the* $\mathsf{CauchyInterlacing}^{\uparrow}$ *schedule row, read only after A, B, W, D have been displayed.*
5. *T is the* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *handoff row witnessing that the interleaved regular read and the displayed source route reach the same regular-real tail boundary.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after A, B, W, D, I, T have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, dyadic, interlacing, tail-equivalence, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D \longmapsto I \longmapsto T \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$, $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegularRealTailEquivalence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, D, I, T, R, H, C, P, N$. It has no coordinate for quotient streams, selected representatives, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9098.2 (Regular Cauchy interleaving merge NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ *carrier*

$$M = (A, B, W, D, I, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}}$ *surface consists exactly of source admission for A, B, shared finite-window exposure through W, dyadic tolerance exposure through D, interlacing-schedule exposure through I, regular-real tail-equivalence exposure through T, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through R, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9098.1. The first two rows expose the regular rational source names, while the shared window row W records the finite requests through which both tails are inspected.

Next read the merge rows. The dyadic ledger D supplies the tolerance accounting, the interlacing row I supplies the finite even/odd schedule already certified by chapter 5870, and the tail-equivalence row T places the merged read inside the regular-real same-tail route of chapter 5940.

Finally read R, H, C, P, N. The seal row is downstream of the source, window, tolerance, interlacing, and tail-equivalence rows, and the structural rows only transport, replay, source, and name that same displayed packet. Since no excluded host object has a carrier coordinate, replay edge, provenance entry, or local naming row, the displayed rows exhaust the local NameCert surface. □

Theorem 9098.3 (Regular Cauchy interleaving merge Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ *carrier reaches the seal row R only through the finite route*

$$A, B, W, D, I, T, H, C, P, N.$$

Thus the merge read is a carrier-local certificate over two displayed $\mathsf{RegSeqRat}^{\uparrow}$ *sources, one* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one* $\mathsf{CauchyInterlacing}^{\uparrow}$ *schedule row, and one* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *handoff row. It is not a quotient stream, selected representative, countable-choice schedule, host equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. The seal row R is listed after the two source rows, shared window, dyadic ledger, interlacing schedule, and regular-real tail-equivalence handoff, so it is not available as an independent endpoint.

Apply the NameCert obligations of Definition 9098.2. They identify the complete route that must be displayed before the local certificate can expose the Real-facing seal.

Use the sibling routes only at their displayed boundaries. The interlacing row I supplies the finite merge schedule from chapter 5870, and the handoff row T supplies the same-real tail boundary from chapter 5940. The remaining rows H, C, P, N transport, replay, source, and name that route, so the seal cannot leak quotient streams, selected representatives, countable choice, host equality, trichotomy, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ Real-facing read displays two RegSeqRatUp source rows, one shared StreamNameUp window, one DyadicRatCoreUp tolerance ledger, one CauchyInterlacingUp schedule row, one RegularRealTailEquivalenceUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-interlacing, regular-real-tail-equivalence: $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, a shared finite window, a dyadic tolerance ledger, an interlacing schedule, a regular-real tail-equivalence handoff, a terminal Real seal, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one shared StreamName window, one DyadicRatCore tolerance ledger, one CauchyInterlacing schedule row, one RegularRealTailEquivalence handoff row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9098.1, Definition 9098.2, and Definition 9098.3 seed the finite regular-Cauchy merge packet over StreamName, RegSeqRat, DyadicRatCore, CauchyInterlacing, RegularRealTailEquivalence, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleavingMerge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires binding the source, shared-window, dyadic-tolerance, interlacing, regular-real tail-equivalence, seal, transport, replay, provenance, and local naming rows to BEDC.Derived.RegularCauchyInterleavingMergeUp.

Chapter 9099

A Concrete Naming Certificate for $\mathsf{StreamShuffle}^{\uparrow}$

Following Bishop and Bridges on sequences in constructive analysis, the $\mathsf{StreamShuffle}^{\uparrow}$ packet records the finite recombination route by which two displayed stream-name windows are read through parity-indexed windows before a regular rational readback and Real-seal handoff are available. It sits beside the stream carrier of chapter 126, the regular rational route of chapter 301, the even/odd Cauchy shuffle of chapter 1721, and the finite merge boundary of chapter 1944. The packet is a finite NameCert object; it is not a host stream, quotient stream equality, selected representative, countable-choice schedule, or ambient completeness principle.

Definition 9099.1 (Stream shuffle carrier)**.** *A* $\mathsf{StreamShuffle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, E, O, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source* $\mathsf{StreamName}^{\uparrow}$ *rows.*
2. *E is the even-index window row reading source A.*
3. *O is the odd-index window row reading source B.*
4. *W is the combined finite window exposed to downstream stream consumers.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the parity-indexed windows when the shuffled stream is read as a regular rational sequence.*
6. *R is the Real-seal handoff row available only after A, B, E, O, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source, parity, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$A, B, E, O \longmapsto W \longmapsto Q \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{StreamShuffle}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{StreamMerge}^{\uparrow}$, $\mathsf{RegularCauchyShuffle}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamShuffle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, E, O, W, Q, R, H, C, P, N$.

Theorem 9099.2 (Stream shuffle NameCert obligations)**.** *For every accepted* $\mathsf{StreamShuffle}^{\uparrow}$ *carrier*

$$S = (A, B, E, O, W, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamShuffle}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *source admission for the two* $\mathsf{StreamName}^{\uparrow}$ *rows A, B;*
2. *parity-window exposure through E, O, W;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q;*
4. *Real-seal handoff through R;*
5. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 9099.1. The source rows A, B are displayed before either parity window is read, so the shuffled stream cannot be introduced as an external host stream.

Next read the parity-indexed windows. The even row E and odd row O expose finite windows of the two source streams, and the combined row W is the only window made public to a downstream $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{StreamMerge}^{\uparrow}$ consumer.

Now attach the readback and seal rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q reads the same parity-indexed window, and the Real-seal handoff R is downstream of Q. This is the point at which the packet can meet the $\mathsf{RegularCauchyShuffle}^{\uparrow}$ route, but it still exposes only the displayed finite rows.

Finally use H, C, P, N. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming preserve the same carrier surface. Since no coordinate names quotient stream equality, a selected representative, countable choice, host stream equality, or ambient completeness, the five listed obligation rows exhaust the local certificate. □

Theorem 9099.3 (Stream shuffle parity-window nonescape)**.** *Every accepted* $\mathsf{StreamShuffle}^{\uparrow}$ *consumer reads the shuffled window only through*

$$A, B, E, O, W, H, C, P, N.$$

If the consumer asks for regular rational or Real data, the same route continues through Q and R; no consumer receives a separate host stream, quotient stream equality, selected representative, or countable-choice schedule.

Proof. Start with the source admission and parity rows of Definition 9099.2. They force each finite window read to enter through A, B, E, O before W is public.

For regular rational and Real-facing reads, use the readback and seal rows Q, R. These rows are downstream of the same parity window and cannot select a second source schedule.

The structural rows H, C, P, N only transport, replay, source, and name the displayed packet. Therefore the consumer remains inside the finite shuffle surface and cannot escape to the excluded host or quotient objects. □

Falsifiable prediction: Every accepted $\mathsf{StreamShuffle}^{\uparrow}$ read displays two StreamNameUp source rows, even and odd parity windows, one combined window, one RegSeqRatUp readback row when rational data is requested, one RealUp seal row when Real data is requested, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, streammerge, regular-cauchy-shuffle, regseqrat: $\mathsf{StreamShuffle}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source StreamName rows, parity-indexed windows, a combined downstream window, optional regular rational readback, a Real-seal handoff, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamShuffle}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamShuffle}^{\uparrow}$ packet is generated from two StreamName source rows, even and odd parity windows, one combined window, a RegSeqRat readback row, a Real-seal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamShuffle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9099.1, Definition 9099.2, and Definition 9099.3 seed the finite stream recombination packet over StreamName, StreamMerge, RegularCauchyShuffle, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamShuffle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable-choice schedules, host stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires binding the source, parity-window, combined-window, readback, Real-seal, transport, replay, provenance, and local naming rows to BEDC.Derived.StreamShuffleUp.

Chapter 9100

A Concrete Naming Certificate for $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$

Following the common-carrier programme of chapter 9413 and the completion-boundary analysis of section 9413.33, the $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ packet records the finite route by which shared carrier rows are allowed to reach completion-facing readback. It reads the dyadic scale surface of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a completion boundary for a common carrier; it does not assert a universal object, quotient completion, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9100.1 (Common-carrier completion-boundary carrier)**.** *A* $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (G, O, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the common-carrier source row, exposing only the shared generated carrier and its public classifier boundary.*
2. *O is the object-specific observable ledger, recording which public rows of the carrier are being tested for completion-facing faithfulness.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to measure finite completion windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row generated from G, O, D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or completion-seal row reached only after D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, observable ledger, dyadic tolerance, stream window, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from common-carrier observation through completion-facing readback and back to the same carrier rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $G, O, D, S, R, E, H, C, P, N$. It has no coordinate for an unlicensed fourth observable, a universal completed carrier, quotient representatives, countable choice, or ambient real completeness.

Theorem 9100.2 (Common-carrier boundary NameCert obligations)**.** *For every accepted* $\mathsf{CommonCarrierCompletio}$ *carrier*

$$Q = (G, O, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the common source G, observable-ledger admission for O, dyadic tolerance admission for D, stream-window generation through S, regular rational readback through R, terminal sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9100.1. The displayed tuple gives the only rows available to the local certificate, so the NameCert surface can expose only $G, O, D, S, R, E, H, C, P, N$.

Read the completion-facing rows in order. The observable ledger O chooses which public rows of the common source are under test, the dyadic row D fixes the finite tolerance scale, the stream row S supplies the windows, and the regular readback R is attached to those windows before the terminal seal E can be inspected.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since none of the excluded objects appears in the carrier, the obligations cannot export a universal completed carrier, a quotient representative, countable choice, or ambient real completeness. □

Theorem 9100.3 (Common-carrier completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ *packet must read the finite route*

$$G, O, D, S, R, E, H, C, P, N.$$

In particular, a consumer moving from a shared carrier row toward $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback cannot bypass the observable ledger O, the dyadic tolerance row D, or the stream window S, and cannot replace them by an unledgered fourth observable, quotient completion, selected representative, or ambient completeness theorem.*

Proof. Start with the NameCert obligations of Definition 9100.2. They force every completion-facing read to expose the common source, observable ledger, dyadic tolerance, stream windows, regular readback, terminal seal, transport, replay, provenance, and local name.

Use the core-completeness boundary of Definition 9413.451. Any row that is not a core readout must be carried as a displayed extension, bridge, boundary, or refusal. Here the boundary row is exactly O, D, S, R, E over the common source G.

Finally inspect the carrier. Since no coordinate stores quotient completion, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be read from the accepted packet. □

Theorem 9100.4 (Common-carrier regular readback handoff)**.** *Let $Q = (G, O, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read from Q is tied to the same common source row G and observable ledger O through the dyadic tolerance row D and stream window S. The handoff is therefore a displayed completion boundary, not a claim that the common carrier itself is a completed universal object.*

Proof. Read D as the finite dyadic tolerance row admitted over the common source and observable ledger. It supplies the scale for the stream windows, but it does not choose a completed object.

The stream windows S are displayed before the regular readback R and terminal seal E can be consumed. Thus every completion-facing read remains attached to the same G, O, D, S route.

Transport, replay, provenance, and local naming preserve those rows. A readback that omits O, D, or S would use a coordinate absent from Definition 9100.1, so it is not a read of the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ completion-facing data without displaying the observable ledger O, dyadic tolerance row D, and $\mathsf{StreamName}^{\uparrow}$ window row S refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole ten-row packet $Q = (G, O, D, S, R, E, H, C, P, N)$, because each sibling contributes only tolerance, window, regular readback, or terminal seal data while this chapter binds those rows to a common-carrier source and an observable completion-boundary ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$ packet is generated from a common-carrier source row, an observable ledger, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9100.1, Definition 9100.2, Definition 9100.3, and Definition 9100.4 seed the common-carrier completion-boundary route over source, observable, dyadic, stream, regular-readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a universal completed carrier, quotient completion, selected representative, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CommonCarrierCompletionBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, observable-ledger exactness, dyadic-window admission, stream generation, regular readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CommonCarrierCompletionBoundaryUp.

Chapter 9101

A Concrete Naming Certificate for FiniteFourierTailProjector$^{\uparrow}$

Following Fourier and Fejer, the FiniteFourierTailProjector$^{\uparrow}$ packet records the finite tail-projection surface used when a displayed Fourier series is inspected only beyond a named cutoff. It sits after the coefficient-window packet of chapter 6529, the symmetric partial-sum kernel of chapter 2311, the positive averaging sibling of chapter 1939, the real-series corridor of chapter 1581, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a constructive finite projector: it does not assert Fourier convergence, host function equality, quotient coefficient streams, countable choice, or a selected infinite tail limit.

Definition 9101.1 (Finite Fourier tail projector carrier)**.** *A* FiniteFourierTailProjector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, A, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the cutoff row, naming the first frequency excluded from the head window and the finite right endpoint of the inspected tail window.*
2. *A is the coefficient-prefix row inherited from the* FourierSeries$^{\uparrow}$ *packet, including the finite coefficient names that determine the inspected partial sums.*
3. *W is the finite tail-window row, listing exactly the frequencies whose coefficients are projected into the tail read.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *frequency ledger, storing the dyadic tolerance, frequency bounds, and finite comparison scale used by the projector.*
5. *S is the* RealSeries$^{\uparrow}$ *partial-sum row for the displayed tail block.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row reached from S and D.*
7. *E is the terminal* Real$^{\uparrow}$ *seal for the projected tail block, readable only after K, A, W, D, S, R have been exposed.*
8. *H records componentwise* hsame *transport for cutoff, coefficient prefix, tail window, dyadic ledger, series row, readback, real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a Fourier-series request through the tail projector and into the same finite tail-window read.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteFourierTailProjector}^{\uparrow}$, $\mathsf{FourierSeries}^{\uparrow}$, $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteFourierTailProjector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, A, W, D, S, R, E, H, C, P, N$. It has no coordinate for host function equality, a quotient coefficient stream, an unbounded tail predicate, countable choice, Fourier convergence, or a selected infinite sum.

Lean4 (definition) `BEDC.Derived.FiniteFourierTailProjectorUp.FiniteFourierTailProjectorUp`

Theorem 9101.2 (Finite Fourier tail projector NameCert obligations)**.** *For every accepted* FiniteFourierTailProjecto *carrier*

$$T = (K, A, W, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFourierTailProjector}^{\uparrow}}$ *surface consists exactly of cutoff admission through K, coefficient-prefix exposure through A, finite tail-window admission through W, dyadic frequency-ledger exactness through D,* $\mathsf{RealSeries}^{\uparrow}$ *tail-block readback through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The cutoff row K fixes the boundary between the head coefficients and the tail block, while A supplies only the finite coefficient prefix already visible to the $\mathsf{FourierSeries}^{\uparrow}$ packet.

Read the projector window next. The row W lists the displayed frequencies, and D records the dyadic scale at which those frequencies and their coefficient reads are inspected. Thus the projector is a finite window operation before any real-facing seal is available.

The tail read then passes through S, R, E. The $\mathsf{RealSeries}^{\uparrow}$ row S records the displayed tail-block partial sum, R gives the regular rational readback, and E seals that endpoint as a $\mathsf{Real}^{\uparrow}$ row only after the preceding data have been projected.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no row stores host function equality, a quotient coefficient stream, Fourier convergence, countable choice, or a selected infinite tail limit, the listed rows exhaust the NameCert obligation surface. □

Theorem 9101.3 (Finite Fourier tail projector Fourier-series handoff)**.** *Every* $\mathsf{FourierSeries}^{\uparrow}$, $\mathsf{DirichletKernel}^{\uparrow}$, *or* $\mathsf{FejerKernel}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteFourierTailProjector}^{\uparrow}$ *packet reaches the projected tail only through the finite route*

$$\mathsf{FourierSeries}^{\uparrow} \longmapsto \mathsf{FiniteFourierTailProjector}^{\uparrow} \longmapsto \mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may compare the projected block with Dirichlet or Fejer kernel windows, but it cannot replace the carrier rows by host function equality, quotient coefficient streams, pointwise convergence, countable choice, or a selected infinite tail.

Proof. Begin with the Fourier-series source. The coefficient-prefix row A is read as finite $\mathsf{FourierSeries}^{\uparrow}$ data, not as a completed coefficient stream.

Apply the tail projector. The cutoff K and tail window W identify the finite block that may be passed to a kernel-facing or series-facing consumer, and the dyadic ledger D fixes the comparison scale for that block.

Read the real-facing rows. The tail block enters $\mathsf{RealSeries}^{\uparrow}$ through S, then $\mathsf{RegSeqRat}^{\uparrow}$ through R, and finally the terminal $\mathsf{Real}^{\uparrow}$ seal through E. Dirichlet and Fejer consumers may inspect the same displayed window only through their named kernel rows.

The structural rows H, C, P, N keep the handoff inside the local certificate. Therefore the route exports a finite Fourier tail projection certificate rather than convergence, quotient equality, choice, or a selected limit. □

Falsifiable prediction: Every accepted $\mathsf{FiniteFourierTailProjector}^{\uparrow}$ read exposes a cutoff row, coefficient-prefix row, finite tail window, dyadic frequency ledger, RealSeriesUp tail-block row, RegSeqRatUp readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert data before a Fourier-series or kernel consumer receives the projected tail.

Independence witness: fourier-series, dirichlet-kernel, fejer-kernel, realseries, regseqrat, dyadicratcore, real: $\mathsf{FiniteFourierTailProjector}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a cutoff, coefficient prefix, finite tail window, dyadic frequency ledger, tail-block series row, regular rational readback, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFourierTailProjector}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{FiniteFourierTailProjector}^{\uparrow}$ packet is generated from a cutoff row, coefficient-prefix row, finite tail window, DyadicRatCore frequency ledger, RealSeries tail-block row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFourierTailProjector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9101.1, Definition 9101.2, Definition 9101.3, the falsifiable prediction, and the independence witness seed the finite Fourier tail-projection packet over displayed coefficient windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFourierTailProjector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Fourier convergence, pointwise convergence, host function equality, quotient coefficient streams, countable choice, selected infinite tails, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cutoff exactness, coefficient-prefix exposure, finite tail-window admission, dyadic frequency-ledger stability, RealSeries readback, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local ledgering for BEDC.Derived.FiniteFourierTailProjectorUp.

Chapter 9102

A Concrete Naming Certificate for $\mathsf{FiniteObservationBoundary}^{\uparrow}$

Following Bishop-style located-real comparison, the $\mathsf{FiniteObservationBoundary}^{\uparrow}$ packet realizes the exact-comparison boundary of section 9238.4 as finite BEDC data. It records two located-real endpoint moduli, the requested exact comparison surface, a finite apartness or equality window, and a refusal ledger for total trichotomy before any comparison-facing consumer reads the packet. It sits beside the located-real comparison packet of chapter 1730, the comparison-modulus packet of chapter 5813, the dyadic comparison boundary of chapter 9085, the finite separation packet of chapter 1202, and the located-order packet of chapter 3640. The packet names a finite observation boundary, not arbitrary exact trichotomy, quotient real equality, LPO, countable choice, or ambient real completeness.

Definition 9102.1 (Finite observation boundary carrier)**.** *A* $\mathsf{FiniteObservationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, R, Q, W, A, E, F, H, C, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint modulus row, exposing only the finite observations selected by its displayed modulus.*
2. *R is the right* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint modulus row, exposed by the same finite-window discipline.*
3. *Q is the requested exact comparison surface, recording that a consumer asked for total left/equal/right classification.*
4. *W is the finite observation window selected from the two endpoint moduli for the request.*
5. *A is the finite apartness or equality row visible at W, using* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback data.*
6. *E is the refusal ledger for total trichotomy: it records that Q may be answered only by finite apartness or finite equality evidence, not by arbitrary exact comparison.*
7. *F records the sibling comparison handoff through located comparison, comparison-modulus, dyadic comparison boundary, finite separation, or located order rows when one of those rows is displayed.*

8. *H records componentwise* hsame *transport for endpoint moduli, request surface, finite window, apartness or equality evidence, refusal ledger, sibling handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from the exact-comparison request back through the finite observation window.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationBoundary}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicModulusComparisonBoundary}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$, $\mathsf{RealLocatedOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, R, Q, W, A, E, F, H, C, P, N$. It has no coordinate for total trichotomy of arbitrary located reals, quotient-real equality, LPO*, countable choice, a selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9102.2 (Finite observation boundary NameCert obligations)**.** *For every accepted* FiniteObservationBound *carrier*

$$B = (L, R, Q, W, A, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteObservationBoundary}^{\uparrow}}$ *surface consists exactly of left and right endpoint-modulus admission through L, R, exact-comparison request exposure through Q, finite observation-window admission through W, apartness or equality evidence through A, trichotomy refusal through E, sibling comparison handoff through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The endpoint rows L and R expose only the observations selected by the two located-real moduli.

Read the request row Q. It records the exact-comparison demand as a surface to be refused or finitely witnessed, not as an available trichotomy operator.

Use the finite window W and the evidence row A. These rows can display a positive apartness witness or a finite equality approximation through dyadic, regular-rational, and Real readbacks, but they do not classify every arbitrary pair of located reals.

The refusal ledger E and sibling handoff F preserve the boundary: downstream comparison consumers must pass through located comparison, comparison-modulus, dyadic boundary, finite separation, or located-order rows when those rows are present. The structural rows H, C, P, N add only transport, replay, provenance, and local naming, so no row can export total trichotomy, quotient equality, omniscience, choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteObservationBoundary}^{\uparrow}$ read is refuted if a located-real comparison consumer returns total left/equal/right trichotomy without first exposing two endpoint modulus rows, the requested exact-comparison surface, a finite observation window, finite apartness or equality evidence, the trichotomy refusal ledger, sibling comparison handoff, transport, replay, provenance, and local NameCert data.

Independence witness: located-real-comparison, located-real-comparison-modulus, dyadic-modulus-comparison-boundary, finite-window-real-separation, real-located-order: $\mathsf{FiniteObservationBoundary}^{\uparrow}$ is not bijective to any listed sibling because it stores the exact-comparison request surface and refusal ledger together with two endpoint moduli, one finite observation window, finite apartness or equality evidence, sibling handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationBoundary}^{\uparrow}}$**.**

Constructive story: A FiniteObservationBoundary$^{\uparrow}$ packet is generated from two LocatedReal endpoint modulus rows, a requested exact-comparison surface, a finite observation window, finite apartness or equality evidence, a total-trichotomy refusal ledger, sibling comparison handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteObservationBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 9102.1, Definition 9102.2, the falsifiable prediction, and the independence witness seed the finite observation boundary for exact located-real comparison requests.

Formal verification: FormalStatus(NameCert$_{\text{FiniteObservationBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary exact trichotomy, quotient-real equality, LPO, countable choice, selected representatives, ambient real completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint-modulus admission, exact-request exposure, finite-window admission, apartness-or-equality evidence, trichotomy refusal, sibling handoff, transport, replay, provenance, and local ledgering for BEDC.Derived.FiniteObservationBoundaryUp.

Chapter 9103

A Concrete Naming Certificate for $\mathsf{MorseHomology}^{\uparrow}$

Following Morse and Milnor, the $\mathsf{MorseHomology}^{\uparrow}$ packet records the finite BEDC surface on which a Morse function, its critical-point rows, oriented unstable-cell data, boundary-count ledgers, and homology readback are consumed as one NameCert. It sits beside the manifold carrier of chapter 96, the homology carrier of chapter 98, the differential-form surface of chapter 136, and the Morse-theory packet of chapter 3533. The certificate names a finite Morse chain and boundary route; it does not import a host transversality theorem, selected gradient-flow compactification, quotient chain equality, or ambient smooth-function space.

Definition 9103.1 ($\mathsf{MorseHomology}^{\uparrow}$ finite carrier)**.** *A* $\mathsf{MorseHomology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, C, G, B, D, H, T, R, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *source row, exposing the charted space on which the Morse data are read.*
2. *F is the finite Morse-function row, read through* $\mathsf{MorseTheory}^{\uparrow}$ *and carrying the displayed critical values and index windows.*
3. *C is the critical-point row family, with each point tied to a chart row of X and an index row of F.*
4. *G is the gradient-incidence row, recording only finite oriented flow-line counts between adjacent index rows.*
5. *B is the boundary ledger for the Morse chain rows; it stores the signed incidence data used to form the differential.*
6. *D is the differential-squared boundary row, recording the finite cancellation surface that later homology consumers may inspect.*
7. *H is the* $\mathsf{Homology}^{\uparrow}$ *readback row, exposing cycles, boundaries, and degree windows through the displayed Morse chain ledger.*
8. *T records componentwise* hsame *transport for the manifold, Morse-function, critical-point, incidence, boundary, cancellation, homology, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from a Morse-homology request through X, F, C, G, B, D, H, T.*

10. *P is the* Pkg *provenance over* $\mathsf{MorseHomology}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{MorseTheory}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient moduli space, a host flow-category quotient, chosen perturbations, selected triangulations, or a completed smooth-function universe.

Theorem 9103.2 ($\mathsf{MorseHomology}^{\uparrow}$ NameCert obligations)**.** *For every accepted* $\mathsf{MorseHomology}^{\uparrow}$ *carrier*

$$M = (X, F, C, G, B, D, H, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}$ *surface consists exactly of carrier admission for the manifold row, Morse-function row, critical-point rows, gradient-incidence row, boundary ledger, differential-squared boundary row, homology readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{Homology}^{\uparrow}$ *consumer reads the cycle and boundary data through H only after X, F, C, G, B, D are displayed, and every differential-form-facing consumer reads orientation or index bookkeeping through the* $\mathsf{DiffForm}^{\uparrow}$ *and* $\mathsf{MorseTheory}^{\uparrow}$ *provenance rows rather than through a separate host complex.*

Proof. Project the carrier of Definition 9103.1. The rows $X, F, C, G, B, D, H, T, R, P, N$ are the only coordinates available to a consumer of the packet.

The Morse chain stage is X, F, C, G, B. The manifold row supplies the charted source, the Morse-function row supplies critical values and index windows, and the critical-point and gradient-incidence rows supply the finite signed boundary counts.

The homology stage is D, H. The cancellation row D records the finite boundary-of-boundary surface, and H exposes only the resulting cycle, boundary, and degree-window readbacks to $\mathsf{Homology}^{\uparrow}$.

Finally T, R, P, N transport, replay, source, and name the same packet. Since the carrier contains no ambient moduli coordinate, quotient chain equality, selected perturbation, or host flow-category object, the displayed rows exhaust the local NameCert obligations. □

Theorem 9103.3 ($\mathsf{MorseHomology}^{\uparrow}$ sibling handoff)**.** *Every* $\mathsf{Manifold}^{\uparrow}$*-,* $\mathsf{Homology}^{\uparrow}$*-, or* $\mathsf{DiffForm}^{\uparrow}$*-facing read of an accepted* $\mathsf{MorseHomology}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Manifold}^{\uparrow},\ \mathsf{MorseTheory}^{\uparrow},\ \mathsf{DiffForm}^{\uparrow},\ \mathsf{Homology}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}.$$

The handoff supplies displayed chart, critical-point, incidence, boundary, cancellation, and homology rows; it does not supply a host transversality theorem, completed moduli space, smooth-function quotient, or selected chain representative outside the packet.

Proof. Start with the obligation surface of Definition 9103.2. A consumer must first read the manifold source X, Morse-function row F, critical-point rows C, and gradient-incidence row G before it can inspect the boundary ledger B.

The differential-form and homology sides are then separated. Orientation and index bookkeeping are carried only through the displayed $\mathsf{DiffForm}^{\uparrow}$ and $\mathsf{MorseTheory}^{\uparrow}$ provenance rows, while homology readback is carried only through D and H.

Transport, replay, provenance, and local naming preserve the same rows. Thus the sibling handoff reaches manifold, homology, and differential-form consumers without adding any of the excluded host-side structures. □

Closure status for $\mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}$.

Constructive story: A $\mathsf{MorseHomology}^{\uparrow}$ packet is generated from a manifold source row, Morse-function row, critical-point rows, gradient-incidence row, boundary ledger, differential-squared boundary row, homology readback, component-wise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9103.1, Definition 9103.2, and Definition 9103.3 seed the finite Morse chain, boundary, cancellation, and homology readback route over manifold, homology, and differential-form siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MorseHomology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close transversality, compactified moduli spaces, continuation invariance, selected perturbations, quotient chain equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-point exposure, incidence boundary exactness, cancellation rows, homology readback, transport, replay, provenance, and local naming for BEDC.Derived.MorseHomologyUp.

Chapter 9104

A Concrete Naming Certificate for $\mathsf{NuclearSpace}^{\uparrow}$

Following Grothendieck and Treves on nuclear locally convex spaces, the $\mathsf{NuclearSpace}^{\uparrow}$ packet records the finite BEDC surface by which a locally convex source is inspected through seminorm windows, bounded-set ledgers, nuclear factorization data, and tensor-test rows before a functional-analysis consumer may cite nuclearity. It sits after the locally convex surface of chapter 3386, the topological-vector source of chapter 205, the tensor-product carrier of chapter 87, the functional-analysis consumer frontier of chapter 94, and the scalar readback of chapter 38. The packet names a displayed nuclearity test route, not a completed locally convex space, a strong duality theorem, Grothendieck's theorem, choice-selected bases, or an ambient function-space quotient.

Definition 9104.1 (Nuclear space carrier)**.** *A* $\mathsf{NuclearSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, S, B, F, T, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *source row, read through its topological-vector, finite-seminorm, and convex-neighbourhood data.*
2. *S is the finite seminorm-family row selected for the nuclearity test window.*
3. *B is the bounded-set ledger, recording the displayed bounded probes against which the seminorm window is read.*
4. *F is the nuclear-map factorization row: it records the finite source seminorm, target seminorm, scalar coefficient readback, and intermediate finite-rank witness used by the local test.*
5. *T is the* $\mathsf{TensorProduct}^{\uparrow}$ *tensor-test row, available only after L, S, B, F have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar-readback row for coefficient bounds, summability budgets, and seminorm estimates.*
7. *Q is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer row that may replay nuclearity only through L, S, B, F, T, R.*
8. *H records componentwise* hsame *transport for locally convex source, seminorm, bounded-set, factorization, tensor-test, scalar-readback, consumer, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a seminorm or bounded-set query through the factorization and tensor-test rows.*

10. *P is the* Pkg *provenance over* $\mathsf{NuclearSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{NuclearSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, S, B, F, T, R, Q, H, C, P, N$. It has no coordinate for topological-vector-space completion, strong duality, Grothendieck theorem, choice-selected bases, ambient function-space quotients, Lean verification, or bridge checking.

Theorem 9104.2 (Nuclear space NameCert obligations)**.** *For every accepted* $\mathsf{NuclearSpace}^{\uparrow}$ *carrier*

$$U = (L, S, B, F, T, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NuclearSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the locally convex source, finite seminorm-family window, bounded-set ledger, nuclear-map factorization row, tensor-test row, real scalar readback, functional-analysis consumer row, componentwise transport, displayed replay, package provenance, and local naming. Every consumer that treats U as a nuclear-space certificate must expose L, S, B, F, T, R before Q, H, C, P, N are public.*

Proof. Project the carrier of Definition 9104.1. The locally convex source is the row L, and the only local probes available to it are the finite seminorm-family row S and bounded-set ledger B.

Read the factorization surface next. The row F stores the finite nuclear-map factorization witness for the displayed test window, while the scalar budgets used by that witness are read through the $\mathsf{Real}^{\uparrow}$ row R.

The tensor-test row T is downstream of the same data. It receives the locally convex source, the seminorm and bounded-set windows, the factorization row, and the scalar readback before a functional-analysis consumer may inspect Q.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no completion object, strong-duality row, Grothendieck theorem, choice-selected basis, or quotient function-space carrier, the listed rows exhaust the local NameCert surface. □

Theorem 9104.3 (Nuclear space finite tensor-test handoff)**.** *Every accepted* $\mathsf{NuclearSpace}^{\uparrow}$ *read that is consumed by* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *must factor through*

$$\mathsf{LocallyConvexSpace}^{\uparrow},\ \mathsf{TopVecSpace}^{\uparrow},\ \mathsf{TensorProduct}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

by way of the displayed rows $L, S, B, F, T, R, Q, H, C, P, N$. The handoff exposes finite seminorm probes, bounded-set tests, nuclear factorization data, tensor-test evidence, and scalar readback; it does not export completed locally convex spaces, strong duals, Grothendieck theorem, selected bases, or quotient function spaces.

Proof. Start from the NameCert obligations of Definition 9104.2. They force a nuclearity consumer to read the locally convex source row, finite seminorm window, bounded-set ledger, factorization row, tensor-test row, and real scalar readback before the local certificate can be used.

The $\mathsf{TopVecSpace}^{\uparrow}$ dependency is already inside the $\mathsf{LocallyConvexSpace}^{\uparrow}$ source route, so a consumer cannot detach the seminorm window from the underlying topological-vector operations and continuity data.

The $\mathsf{TensorProduct}^{\uparrow}$ row then receives only the displayed factorization data. Its tensor test is a finite BEDC handoff from F and R, not a completed projective tensor product or a host theorem about all continuous linear maps.

The structural rows H, C, P, N preserve that finite route into $\mathsf{FunctionalAnalysis}^{\uparrow}$. Because no coordinate stores strong duality, a Grothendieck theorem, a selected basis, or an ambient quotient function space, none of those endpoints can be exported by the packet. □

Falsifiable prediction: An accepted $\mathsf{NuclearSpace}^{\uparrow}$ consumer that reads nuclearity without first displaying a $\mathsf{LocallyConvexSpace}^{\uparrow}$ source, finite seminorm-family window, bounded-set ledger, nuclear-map factorization row, $\mathsf{TensorProduct}^{\uparrow}$ tensor test, $\mathsf{Real}^{\uparrow}$ scalar readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: locally-convex-space, functional-analysis, topvecspace, tensor-product, real: $\mathsf{NuclearSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a locally convex source, finite seminorm windows, bounded-set ledgers, nuclear factorization rows, tensor-test rows, scalar readback, transport, replay, provenance, and local NameCert data while refusing completion, strong duality, selected bases, and quotient function-space endpoints.

Closure status for $\mathsf{NameCert}_{\mathsf{NuclearSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{NuclearSpace}^{\uparrow}$ packet is generated from a LocallyConvexSpace source row, finite seminorm-family window, bounded-set ledger, nuclear-map factorization row, TensorProduct test row, Real scalar readback, FunctionalAnalysis consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NuclearSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9104.1, Definition 9104.2, and Definition 9104.3 seed the finite nuclear-space route through locally convex, topological-vector, tensor-test, scalar-readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NuclearSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full topological-vector-space completion, strong duality, Grothendieck theorem, choice-selected bases, ambient function-space quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite seminorm-window exactness, bounded-set ledger routing, nuclear factorization routing, tensor-test handoff, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.NuclearSpaceUp.

Chapter 9105

A Concrete Naming Certificate for $\mathsf{CauchySequenceMorphism}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, a $\mathsf{CauchySequenceMorphism}^{\uparrow}$ packet names a finite modulus-preserving route between accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ packets. It is not a host function between streams and not a quotient map between completion classes. The route consumes the source and target spaces of chapter 406, the scheduled windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, the modulus rows of chapter 292, and the completion boundary of chapter 360. Its Real-facing read remains compatible with the seal discipline of chapter 38.

Definition 9105.1 (Cauchy sequence morphism carrier)**.** *A* $\mathsf{CauchySequenceMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, M, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted source* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet.*
2. *T is the accepted target* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet.*
3. *M is the modulus-preserving row, carrying each source finite-window request to a target window request with the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *bound.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records how the source tolerance is carried to the target tolerance.*
5. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after S, T, M, D have been displayed.*
6. *H records componentwise* hsame *transport for the source, target, modulus-preserving row, tolerance ledger, Real-seal handoff, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceMorphism}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceMorphism}^{\uparrow}}$ *row.*

The classifier compares accepted morphism packets only through $S, T, M, D, E, H, C, P, N$. It has no coordinate for a host function, quotient stream map, selected limit, countable choice schedule, ambient completeness principle, or full category of completions.

Theorem 9105.2 (Cauchy sequence morphism NameCert obligations)**.** *For every accepted* CauchySequenceMorphis *carrier*

$$F = (S, T, M, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceMorphism}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *source and target admission for the accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets S and T;*
2. *modulus preservation, saying that every source window request consumed by the morphism is routed through M to a target window request with the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *bound;*
3. *dyadic tolerance exactness for D, tying the transported source tolerance to the target tolerance ledger;*
4. *Real-seal handoff nonescape for E, forcing every completion-facing read through S, T, M, D before any* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is consumed;*
5. *structural stability for H, C, P, N, transporting, replaying, sourcing, and locally naming only the same displayed route.*

No morphism obligation is stored outside these rows.

Proof. Project the carrier of Definition 9105.1. The source and target sequence-space packets are the only endpoints available, and M is the only row that transports window requests between them.

Read the dyadic ledger D. It is stored beside the same source, target, and modulus rows, so tolerance transfer cannot be supplied by an external real equality or by a quotient stream map.

Finally read the handoff row E and the structural rows H, C, P, N. They preserve the displayed route through transport, replay, provenance, and local naming, and they add no coordinate for host functions, selected limits, countable choice, ambient completeness, or a category-of-completions theorem. □

Theorem 9105.3 (Cauchy sequence morphism Real-seal nonescape)**.** *Let*

$$F = (S, T, M, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceMorphism}^{\uparrow}$ *carrier. Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$*- or* $\mathsf{Real}^{\uparrow}$*-facing read exported by F factors first through the source packet S, the target packet T, the modulus-preserving row M, the dyadic tolerance ledger D, the transport row H, the replay row C, the provenance row P, and the local naming row N before the seal row E is consumed.*

Consequently the morphism exports no host function, quotient stream map, selected limit, countable-choice schedule, ambient completeness principle, or full category-of-completions structure.

Proof. Start with the accepted carrier. The only Real-facing coordinate is E, and it is stored in the same packet as S, T, M, D, H, C, P, N.

Apply the NameCert obligation surface of Definition 9105.2. The source-target admission row and modulus-preservation row place S, T, M before any completion-facing read, while dyadic exactness places D on the same route.

The structural stability row then routes the read through H, C, P, N. Since these rows only transport, replay, source, and locally name the displayed packet, the seal row cannot be consumed through a host function, quotient stream map, selected limit, choice schedule, ambient completeness principle, or category-of-completions structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceMorphism}^{\uparrow}$ packet is generated from source and target $\mathsf{CauchySequenceSpace}^{\uparrow}$ rows, a modulus-preserving row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RealCauchyCompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceMorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9105.1, Definition 9105.2, and Definition 9105.3 seed the finite modulus-preserving morphism route between accepted Cauchy-sequence-space packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceMorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host functions, quotient stream maps, selected limits, countable choice, ambient completeness, a full category of completions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-target admission, modulus preservation, dyadic tolerance exactness, Real-seal nonescape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.CauchySequenceMorphism

Chapter 9106

A Concrete Naming Certificate for $\mathsf{BishopRealNameRealizer}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRealNameRealizer}^{\uparrow}$ packet records the finite route by which a rational source window is realized as a constructive real name before any host real equality, quotient class, countable-choice selector, or ambient completeness theorem is available. It sits beside the finite observation discipline of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the located-real witness surface of chapter 6180, and the terminal real-name seal of chapter 38. The packet names a finite real-name realization layer for later Real and RegSeqRat consumers, not a completed host real.

Definition 9106.1 (Bishop real-name realizer carrier)**.** *A* $\mathsf{BishopRealNameRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, W, D, G, L, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational source-window row, listing the finite rational observations and requested precision bounds before any real-facing read is allowed.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window opened for exactly the positions named by Q.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, slack bounds, and comparison budgets used by the window.*
4. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the same window as regular rational data.*
5. *L is the* $\mathsf{BishopLocatedReal}^{\uparrow}$ *witness row, recording located lower and upper tests only through the displayed rational and dyadic rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after Q, W, D, G, L have been replayed.*
7. *H records componentwise* hsame *transport for the source window, stream window, dyadic ledger, regular readback, located witness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-name request through $Q, W, D, G, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRealNameRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealNameRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by $Q, W, D, G, L, E, H, C, P, N$. It has no coordinate for host real equality, quotient classes of names, countable choice, trichotomy, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopRealNameRealizerUp`

Theorem 9106.2 (Bishop real-name realizer NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealNameRealizer}^{\uparrow}$ *carrier $R = (Q, W, D, G, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealNameRealizer}^{\uparrow}}$ *surface consists exactly of rational source-window exposure through Q, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through G,* $\mathsf{BishopLocatedReal}^{\uparrow}$ *witness exposure through L, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9106.1. The finite analytic rows are exactly Q, W, D, G, L, E, and the structural rows are exactly H, C, P, N.

The source window Q fixes the rational data being realized. The finite observation row W supplies only the named stream positions, while the dyadic ledger D supplies the comparison budget for those observations.

The real-facing rows remain downstream. The regular rational row G reads the displayed observations, the located witness L records only the located tests supported by the same rational and dyadic budget, and the terminal seal E is reached only after those rows have been replayed.

Finally H, C, P, N preserve, replay, source, and name the same packet. Since no excluded host equality, quotient, choice, trichotomy, selected-limit, or ambient-completeness object is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 9106.3 (Bishop real-name realizer finite-window readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRealNameRealizer}^{\uparrow}$ *carrier reads the real name through the finite route*

$$Q \longmapsto W \longmapsto D \longmapsto G \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

The exported read is a finite real-name realization from displayed rational windows; it is not an equality class of streams, a choice-selected Cauchy limit, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 9106.2. It requires every real-name read to expose the rational source window, StreamName observation window, dyadic tolerance ledger, regular rational readback, located witness, Real seal, transport, replay, provenance, and local naming rows.

The readback route is finite because Q and W name only displayed windows. The dyadic row D fixes the tolerance ledger for the window, and G is the only regular-rational readback row available to the consumer.

The located witness L and real seal E are downstream of that finite route. Componentwise transport, replay, provenance, and naming through H, C, P, N preserve the same order and do not add a detached tail or completed host object. □

Theorem 9106.4 (Bishop real-name realizer Real-seal nonescape)**.** *For every accepted* $\mathsf{BishopRealNameRealizer}^{\uparrow}$ *carrier $R = (Q, W, D, G, L, E, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is readable only after the rational source window Q, finite observation window W, dyadic tolerance ledger D, regular rational readback G, located-real witness L, componentwise transport H, displayed replay C, provenance P, and local naming N have been exposed. Thus E cannot be consumed as host real equality, quotient-name identity, countable choice, a selected limit, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 9106.1. The only Real-facing coordinate is E, and it appears in the same finite packet as $Q, W, D, G, L, H, C, P, N$.

Apply finite-window readback, Definition 9106.3. It places E after the rational source, StreamName window, dyadic tolerance ledger, RegSeqRat readback, and Bishop located witness.

The refusal follows from the absence of the excluded objects in the carrier. Since no row stores host real equality, quotient-name identity, countable choice, selected limits, trichotomy, or ambient completeness, the Real seal has no escape route outside the displayed NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{BishopRealNameRealizer}^{\uparrow}$ read that reaches a Real seal factors through one rational source window, one StreamName finite observation window, one DyadicRatCore tolerance ledger, one RegSeqRat readback, one BishopLocatedReal witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: streamname, dyadicratcore, regseqrat, bishoplocatedreal, real: $\mathsf{BishopRealNameRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a rational source window, finite observation window, dyadic tolerance ledger, regular readback, located witness, Real seal, componentwise transport, replay, provenance, and local naming as one realization packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealNameRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealNameRealizer}^{\uparrow}$ packet is generated from a rational source window, StreamName finite observation window, DyadicRatCore tolerance ledger, RegSeqRat readback, BishopLocatedReal witness, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealNameRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9106.1, Definition 9106.2, Definition 9106.3, and Definition 9106.4 seed the finite constructive real-name realization route from rational windows to a downstream RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealNameRealizer}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.BishopRealNameRealizerUp`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient-name identity, countable choice, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, located-witness exposure, Real-seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.BishopRealNameRealizerUp.

Chapter 9107

A Concrete Naming Certificate for $\mathsf{ParacompactSpace}^{\uparrow}$

Following Bourbaki and Engelking on paracompact Hausdorff spaces, the $\mathsf{ParacompactSpace}^{\uparrow}$ packet records the space-level certificate that binds a topology source, Hausdorff separation, normal-space separation, locally finite refinement, and partition-refinement ledgers into one finite BEDC NameCert surface. It sits over the open-cover refinement route of chapter 1963, the point-separation rows of chapter 1925, and the closed-pair separation route of chapter 1866. The packet names only displayed topological, refinement, and partition-refinement rows; it does not assert arbitrary subcover selection, host powersets of open sets, unrestricted partitions of unity, or ambient choice principles.

Definition 9107.1 (Paracompact space carrier)**.** *A* $\mathsf{ParacompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, H, N, A, R, L, P, W, E, K, C, G, Z)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the ambient open-row operations on the displayed space surface.*
2. *H is the* $\mathsf{HausdorffSpace}^{\uparrow}$ *point-separation row, read only as the sibling neighbourhood-separation certificate.*
3. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *closed-pair separation row over the same topology source.*
4. *A is the displayed open-cover request row.*
5. *R is the locally finite open-refinement ledger inherited from the* $\mathsf{Paracompact}^{\uparrow}$ *refinement surface.*
6. *L is the coverage and containment ledger, recording that refinement cells cover the displayed region and sit inside their assigned cover cells.*
7. *P is the partition-refinement ledger, recording displayed support rows, local finiteness of the support family, and assignment of each support row to the refinement cell that licenses it.*
8. *W is the exactness row tying the partition-refinement support ledger to the same cover and refinement windows A, R, L.*
9. *E records componentwise* hsame *transport for the topology, separation, refinement, partition-refinement, exactness, replay, provenance, and local naming rows.*

10. *K records displayed* Cont *replay from a paracompact-space request through* $T, H, N, A, R, L, P, W, E, G, Z$.
11. *C is the classifier row comparing accepted packets only through the displayed coordinates* $T, H, N, A, R, L, P, W, E$
12. *G is the* Pkg *provenance over* $\mathsf{ParacompactSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *Z is the local* $\mathsf{NameCert}_{\mathsf{ParacompactSpace}^{\uparrow}}$ *row.*

The classifier has no coordinate for a selected subcover, an arbitrary host-indexed open family, unrestricted partition-of-unity weights, host complement selection, Lean verification, or bridge checking.

Theorem 9107.2 (Paracompact space NameCert obligations)**.** *For every accepted* $\mathsf{ParacompactSpace}^{\uparrow}$ *carrier*

$$S = (T, H, N, A, R, L, P, W, E, K, C, G, Z),$$

the local $\mathsf{NameCert}_{\mathsf{ParacompactSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, topology-source exposure, Hausdorff and normal-space separation exposure, open-cover exposure, locally finite refinement exactness, partition-refinement ledger exposure, support-to-refinement exactness, componentwise transport, displayed replay, classifier comparison, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 9107.1. The accepted tuple displays exactly the topology source, the Hausdorff and normal separation rows, the cover request, the locally finite refinement ledger, the coverage and containment ledger, the partition-refinement ledger, exactness, transport, replay, classifier, provenance, and local naming.

The topological and separation obligations are read from T, H, N. Point-separation information must pass through the $\mathsf{HausdorffSpace}^{\uparrow}$ row, while closed-pair separation information must pass through the $\mathsf{NormalSpace}^{\uparrow}$ row over the same topology source.

The refinement obligations are read from A, R, L. The cover request is displayed before the refinement is consumed, and the ledger records both local finiteness and containment in the assigned cover rows.

The partition-refinement obligations are read from P, W. These rows expose only support rows tied to the displayed locally finite refinement; they do not create a host partition-of-unity object or a selected subcover.

Finally, E, K, C, G, Z exhaust transport, replay, classifier comparison, provenance, and local naming. Since the packet has no further coordinate, no NameCert obligation can export arbitrary cover choice, host powersets, unrestricted weights, ambient normality, Lean verification, or bridge checking. □

Theorem 9107.3 (Paracompact space locally finite refinement route)**.** *Every accepted* $\mathsf{ParacompactSpace}^{\uparrow}$ *refinement read factors through the displayed route*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{HausdorffSpace}^{\uparrow} \longrightarrow \mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{Paracompact}^{\uparrow} \longrightarrow \mathsf{ParacompactSpace}^{\uparrow}.$$

The exported refinement data are exactly the cover request, locally finite refinement ledger, coverage and containment ledger, separation rows, transport, replay, provenance, and local naming rows of the carrier.

Proof. Start with an accepted carrier $S = (T, H, N, A, R, L, P, W, E, K, C, G, Z)$. The topology source T is displayed before any separation or refinement row is read.

The separation segment is H, N. The Hausdorff row supplies point-neighbourhood separation as a sibling certificate, and the normal-space row supplies closed-pair separation over the same displayed topology.

The refinement segment is A, R, L. It is the only route from the open-cover request to a locally finite refinement read and to the corresponding coverage and containment exactness rows.

The structural rows E, K, C, G, Z preserve this order under transport, replay, classifier comparison, provenance, and naming. Hence a refinement consumer reaches $\mathsf{ParacompactSpace}^{\uparrow}$ only through the displayed topology, separation, and locally finite refinement rows. □

Theorem 9107.4 (Paracompact space partition-refinement ledger)**.** *Every partition-refinement read exported by an accepted* $\mathsf{ParacompactSpace}^{\uparrow}$ *packet factors through the locally finite refinement ledger before any support row can be consumed. The partition-refinement ledger is the finite pair* (P, W)*:* P *displays support rows and their assignment to refinement cells, while* W *records exactness against the same cover and refinement windows* A, R, L*.*

Proof. Project the carrier of Definition 9107.1. The rows P and W are the only partition-refinement coordinates in the packet.

Before P can be read, the cover and refinement rows A, R, L have already displayed the local finiteness, coverage, and containment data. Thus every support row named by P is tied to a displayed refinement cell rather than to an arbitrary host open family.

The exactness row W then checks the support assignment against the same cover and refinement windows. This prevents the partition-refinement read from escaping to unrestricted partition-of-unity weights, host complement selection, or a separate ambient normality theorem.

Transport, replay, classifier comparison, provenance, and local naming are exhausted by E, K, C, G, Z, so the partition-refinement ledger remains a finite NameCert row over the displayed paracompact-space carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ParacompactSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ParacompactSpace}^{\uparrow}$ packet is generated from a $\mathsf{Topology}^{\uparrow}$ source row, $\mathsf{HausdorffSpace}^{\uparrow}$ point-separation row, $\mathsf{NormalSpace}^{\uparrow}$ closed-pair separation row, displayed open-cover request, locally finite refinement ledger, coverage and containment ledger, partition-refinement support ledger, exactness row, componentwise hsame transport, displayed Cont replay, classifier row, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ParacompactSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9107.1, Definition 9107.2, Definition 9107.3, and Definition 9107.4 seed the finite space-level route from topology and separation rows through locally finite refinement into partition-refinement support ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ParacompactSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subcovers, unrestricted partitions of unity, host powersets of opens, ambient choice principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Hausdorff and normal separation exposure, locally finite refinement exactness, partition-refinement ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.ParacompactSpaceUp.

Chapter 9108

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$

Following Bishop and Bridges on regulated functions and regular Cauchy real names, the RegulatedCauchyStepEnvel packet records the finite route by which a regulated step approximant supplies regular rational readback before a $\mathsf{Real}^{\uparrow}$ completion-facing seal is consumed. It sits after the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational name surface of chapter 301, the regulated-function packet of chapter 1906, and the terminal real-name seal of chapter 38. The packet names the finite step-envelope handoff without importing quotient functions, selected limits, measure completion, host step functions, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9108.1 (Regulated Cauchy step-envelope carrier)**.** *A* $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, S, W, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row whose finite step approximants are being read.*
2. *S is the step-envelope row, listing the inspected partition cells, rational step values, and the finite oscillation budget attached to the request.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened on the approximant schedule before any regular readback is served.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the partition mesh, the step-value comparisons, and the terminal seal request.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from F, S, W, D.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regulated source, step envelope, finite windows, dyadic ledger, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the source, envelope, window, dyadic, regular-readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$F, S, W, D \longmapsto Q \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, W, D, Q, R, H, C, P, N$. It has no coordinate for a quotient function, selected limit, measure completion, host step function, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 9108.2 (Regulated Cauchy step-envelope NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$ *carrier*

$$E = (F, S, W, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed regulated-function source, step-envelope schedule, finite-window row, dyadic-tolerance ledger, regular rational readback, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows. A consumer may read the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q only through F, S, W, D, and may read the* $\mathsf{Real}^{\uparrow}$ *seal R only through $F, S, W, D, Q, H, C, P, N$.*

Proof. Project the carrier of Definition 9108.1. The regulated source row F and step-envelope row S are the only rows that expose approximant data, while the window row W records the finite stream requests through which the approximant schedule is inspected.

Read the dyadic ledger D. It stores the mesh and step-value tolerance comparisons attached to the same finite request, so no external real comparison or hidden selected representative can supply the error budget.

Finally read Q, R, H, C, P, N. The regular readback Q is downstream of F, S, W, D, and the Real seal R is downstream of that readback. The structural rows transport, replay, source, and locally name only the displayed packet, so the local certificate surface has no further coordinate from which a quotient function, selected limit, measure completion, host step function, trichotomy, or ambient completeness principle could be read. □

Theorem 9108.3 (Regulated Cauchy step-envelope Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read of an accepted* $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$ *carrier factors first through the finite route*

$$F, S, W, D, Q, H, C, P, N.$$

Thus the handoff from regulated step approximants to a real-name seal is a carrier-local certificate over $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. It exports no quotient function space, selected limiting function, measure completion, hidden step-function representative, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the accepted carrier. The only terminal real-name coordinate is R, and it is stored in the same finite packet as the regulated source row, step envelope, stream window, dyadic ledger, regular readback, transport, replay, provenance, and local naming rows.

Apply the obligation surface of Definition 9108.2. It places F, S, W, D before the regular rational readback Q, and places Q before any $\mathsf{Real}^{\uparrow}$-facing seal can be consumed.

The remaining rows H, C, P, N preserve the displayed route by componentwise transport, replay, provenance, and local naming. Since the carrier of Definition 9108.1 contains no row for quotient function equality, selected limits, measure completion, host step functions, trichotomy, or ambient Real completeness, every completion-facing read is forced through the finite envelope route above. □

Falsifiable prediction: Every accepted $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$ completion-facing read displays one RegulatedFunction source row, one finite step-envelope row, one StreamName window row, one DyadicRatCore tolerance ledger, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal before the seal is consumed.

Independence witness: regulated-function, streamname, dyadicratcore, regseqrat, real: RegulatedCauchyStepEnv is not bijective to any listed sibling because it jointly carries a regulated-function source, a finite step-envelope schedule, finite stream windows, a dyadic tolerance ledger, regular rational readback, a terminal Real seal, transport, replay, provenance, and local NameCert in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}$ packet is generated from a regulated-function source row, finite step-envelope schedule, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9108.1, Definition 9108.2, and Definition 9108.3 seed the finite route from regulated step approximants to regular rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyStepEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient functions, selected limits, measure completion, host step functions, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, step-envelope schedule admission, finite-window exposure, dyadic tolerance exactness, RegSeqRat readback, Real-seal nonescape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.RegulatedCauchyStepEnvelopeUp.

Chapter 9109

A Concrete Naming Certificate for $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$

Following Bishop and Bridges constructive compactness criteria, the $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$ packet records the finite route by which Cauchy real data are admitted to a compactness read through explicit net evidence. It sits beside the finite-window naming surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the compact-metric surface of chapter 129, the interval finite-net row of chapter 1379, the finite-net selection principle of chapter 9078, and the regular Cauchy criterion of chapter 8437. The packet names the handoff from regular Cauchy windows to finite nets; it does not import open-cover compactness, sequential compactness without explicit data, quotient-stream equality, countable choice, or ambient Real completeness.

Definition 9109.1 (Cauchy compactness criterion carrier)**.** *A* $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, D, R, M, I, F, G, H, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records the inspected name windows.*
2. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data exposed by those windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling mesh size, Cauchy radius, and comparison budget.*
4. R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S, Q, D *have been displayed.*
5. M *is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-metric row read by the finite-net consumer.*
6. I *is the* $\mathsf{HeineBorelInterval}^{\uparrow}$ *interval-net row supplying bounded interval net cells.*
7. F *is the* $\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}$ *row selecting the displayed finite net from the available compact-metric and interval data.*
8. G *is the* $\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}$ *handoff row that records compatibility with regular Cauchy compactness data.*
9. H *records componentwise* hsame *transport for* $S, Q, D, R, M, I, F, G, P, N$.

10. *P is the displayed* Cont *replay and* Pkg *provenance over* $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelInterval}^{\uparrow}$, $\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}$, $\mathsf{RegularCauchyCompactnessCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompactnessCriterion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that compactness is read through these finite rows. It has no coordinate for open-cover compactness, sequential compactness without explicit finite-window data, quotient equality of streams, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9109.2 (Cauchy compactness criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$ *carrier* $K = (S, Q, D, R, M, I, F, G, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompactnessCriterio}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{CompactMetric}^{\uparrow}$ *compact-metric row,* $\mathsf{HeineBorelInterval}^{\uparrow}$ *interval-net row,* $\mathsf{FiniteNetSelectionPrinciple}^{\uparrow}$ *finite-net selection row,* RegularCauchyCompactness *handoff row, componentwise* hsame *transport, displayed* Cont *replay with* Pkg *provenance, and local naming. No obligation row may be discharged by host open-cover compactness, sequential compactness without displayed data, quotient-stream equality, countable choice, selected representatives, or ambient Real completeness.*

Proof. Project the carrier of Definition 9109.1. Its displayed coordinates are exactly $S, Q, D, R, M, I, F, G, H, P, N$, so carrier admission and local naming can inspect only those rows.

Read the Cauchy side first. The finite windows S expose the regular rational readback Q, the dyadic tolerance row D controls the Cauchy and mesh budgets, and the terminal seal R is downstream of those displayed rows.

Now read the compactness side. The compact-metric row M, interval-net row I, finite-net selection row F, and regular Cauchy handoff G provide the complete finite-net route consumed by the criterion. Transport, replay, provenance, and local naming preserve the same finite packet, so the excluded compactness and choice principles have no carrier coordinate. □

Theorem 9109.3 (Cauchy compactness criterion finite-net handoff)**.** *Every compactness read of an accepted* $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$ *carrier factors through the finite-net route*

$$S, Q, D, R, M, I, F, G, H, P, N.$$

In particular, the read consumes StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, the Real seal, CompactMetric finite data, Heine–Borel interval nets, FiniteNetSelectionPrinciple selection data, the RegularCauchyCompactnessCriterion handoff, transport, replay, provenance, and local naming before it can expose compactness data. It is therefore a finite-net handoff, not open-cover compactness, not sequential compactness without explicit data, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 9109.2. They identify the complete local surface of an accepted carrier, so a compactness read cannot leave the rows $S, Q, D, R, M, I, F, G, H, P, N$.

Use the Real-facing route next. Rows S, Q, D provide finite observation, regular rational readback, and tolerance accounting before the terminal seal R is available.

Finally apply the finite-net rows M, I, F, G. The compact-metric and interval rows provide the net cells, the selection-principle row records the finite choice already displayed in the packet, and the regular Cauchy handoff keeps the read compatible with regular Cauchy compactness data. Since none of these rows contains an open-cover subcover choice, a sequence-only compactness

principle, quotient stream equality, countable choice, or an ambient completeness theorem, the compactness read is exactly the stated finite-net handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompactnessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompactnessCriterion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal RealUp seal, CompactMetric finite data, Heine–Borel interval nets, FiniteNetSelectionPrinciple selection data, RegularCauchyCompactnessCriterion handoff, componentwise hsame transport, displayed Cont replay with Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompactnessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9109.1, Definition 9109.2, and Definition 9109.3 seed the finite-net compactness criterion route over StreamName, RegSeqRat, DyadicRatCore, Real, CompactMetric, Heine–Borel interval, finite-net selection, and regular Cauchy compactness data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompactnessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness without explicit data, quotient-stream equality, countable choice, selected representatives, Lean verification, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exactness, regular rational readback, dyadic tolerance accounting, Real seal routing, CompactMetric handoff, Heine–Borel interval handoff, finite-net selection handoff, regular Cauchy compactness handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompactnessCriterion

Chapter 9110

A Concrete Naming Certificate for $\mathsf{NearestPointProjection}^{\uparrow}$

Following Bishop–Bridges metric constructions and Kelley-style metric neighbourhoods, the $\mathsf{NearestPointProjection}^{\uparrow}$ packet records a displayed nearest-point projection witness only when the metric source, located candidate surface, finite distance comparisons, endpoint row, dyadic comparison ledger, and $\mathsf{Real}^{\uparrow}$ distance seal are all present. It sits after the metric source of chapter 57, the terminal real-distance naming discipline of chapter 38, the finite projection table of chapter 7348, and the finite selector boundary of chapter 7761. The packet is a finite displayed projection certificate; it does not assert a global nearest-point choice principle, hidden minimizers, quotient subset equality, compactness, or ambient completion.

Definition 9110.1 (Nearest-point projection carrier)**.** *A* $\mathsf{NearestPointProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, S, T, Q, E, R, H, C, G, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *source row naming the metric space in which the projection question is posed.*
2. *S is the located candidate-surface row naming the finite visible candidate family and its membership ledger.*
3. *T is the consumed* $\mathsf{FiniteMetricProjection}^{\uparrow}$ *table row, recording the displayed distances from the query point to the visible candidates.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger used to compare all distance entries in T.*
5. *E is the endpoint row naming the candidate whose distance row is certified minimal by Q.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *distance seal attached to the endpoint row after the finite comparisons have been replayed.*
7. *H records componentwise* hsame *transport for the metric source, candidate surface, finite table, comparison ledger, endpoint, distance seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a projection request through the finite table, comparison ledger, endpoint row, and terminal distance seal.*
9. *G is the* Pkg *provenance over* $\mathsf{NearestPointProjection}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteMetricProjection}^{\uparrow}$, $\mathsf{FiniteMetricProjectionSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{NearestPointProjection}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host projection function, selected minimizer outside the displayed table, quotient subset equality, compactness theorem, countable selection schedule, or ambient completion.

Theorem 9110.2 (Nearest-point projection NameCert obligations)**.** *For every accepted* $\mathsf{NearestPointProjection}^{\uparrow}$ *carrier* $P = (M, S, T, Q, E, R, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NearestPointProjection}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, located candidate-surface exposure, finite projection-table exposure, dyadic comparison admission, endpoint handoff, terminal distance sealing, componentwise stability transport, displayed replay, provenance, local naming, and refusal of global nearest-point choice, hidden minimizers, quotient subset equality, compactness, and ambient completion.*

Proof. Project the carrier of Definition 9110.1. Its public rows are exactly $M, S, T, Q, E, R, H, C, G, N$, so admission exposes only the displayed metric source, candidate surface, finite distance table, comparison ledger, endpoint row, distance seal, and structural rows.

The finite projection-table obligation is inherited from chapter 7348. The row T supplies displayed candidate-distance entries, while Q supplies the dyadic comparison route that can certify one endpoint row against the visible table.

The endpoint and refusal obligations follow from the absence of any additional coordinate. Since the packet stores no host projection map, no quotient subset object, no compactness theorem, and no completion row, the local NameCert surface can export only the displayed endpoint and terminal $\mathsf{Real}^{\uparrow}$ distance seal. □

Theorem 9110.3 (Nearest-point projection distance minimality)**.** *Let* $P = (M, S, T, Q, E, R, H, C, G, N)$ *be an accepted* $\mathsf{NearestPointProjection}^{\uparrow}$ *packet. Every visible candidate row s in the located surface S is compared with the endpoint row E by the finite route*

$$M, S, T, Q \longmapsto E, R \longmapsto H, C, G, N.$$

The terminal read therefore certifies the displayed inequality

$$d_M(x, E) \leq d_M(x, s)$$

only for candidates whose distance rows occur in T.

Proof. Start with the visible candidate surface S and the finite table T. The table contains exactly the distance rows available to the packet, so a candidate s can be compared with E only when its distance entry is displayed in T.

Read the dyadic ledger Q. It performs the finite comparison between the endpoint entry and each displayed candidate entry, and the endpoint row E is accepted only after those comparisons have been replayed.

Finally read the terminal seal R and structural rows H, C, G, N. They preserve the comparison route as BEDC data and expose the inequality as a finite displayed fact, not as an ambient minimization principle over an unbounded subset. □

Theorem 9110.4 (Nearest-point projection completion handoff)**.** *Every completion-facing consumer that reads an accepted* $\mathsf{NearestPointProjection}^{\uparrow}$ *packet must first expose the metric row M, candidate surface S, finite projection table T, dyadic comparison ledger Q, endpoint row E, terminal* $\mathsf{Real}^{\uparrow}$ *seal R, and structural rows H, C, G, N. The handoff supplies a finite endpoint witness and its displayed distance; it does not import ambient completion, quotient subset equality, compactness, or a global projection selector.*

Proof. Begin with the metric and real-distance dependencies. The rows M and R place the packet inside the metric and $\mathsf{Real}^{\uparrow}$ naming surfaces referenced by chapter 57 and chapter 38.

Next read the finite projection table and selector boundary. The row T is the finite projection table admitted by chapter 7348, and the endpoint read is constrained by the finite selector discipline of chapter 7761.

It remains to pass the endpoint to the consumer. The continuation row C replays M, S, T, Q, E, R, while H, G, N preserve transport, provenance, and local naming. Because no completion carrier or quotient subset coordinate is present, the consumer receives only the displayed endpoint witness and distance seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{NearestPointProjection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NearestPointProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9110.1, Definition 9110.2, Definition 9110.3, and Definition 9110.4 seed the finite displayed nearest-point projection certificate over MetricUp, RealUp, FiniteMetricProjectionUp, FiniteMetricProjectionSelectorUp, DyadicRatCoreUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NearestPointProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global nearest-point choice, hidden minimizers, quotient subset equality, compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located candidate-surface exposure, finite comparison exactness, endpoint handoff, Real distance sealing, transport, replay, provenance, and local naming for BEDC.Derived.NearestPointProjectionUp.

Constructive story: A $\mathsf{NearestPointProjection}^{\uparrow}$ packet is generated from a Metric row, a located candidate-surface row, a FiniteMetricProjection table row, a DyadicRatCore comparison ledger, an endpoint row, a Real distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9111

A Concrete Naming Certificate for $\mathsf{MetricApartnessCompletion}^{\uparrow}$

Following Bishop–Bridges constructive metric and apartness completion, the $\mathsf{MetricApartnessCompletion}^{\uparrow}$ packet records the finite route by which a separated metric completion keeps positive apartness data visible while it accepts Cauchy names. It sits after the metric carrier of chapter 57, the abstract completion packet of chapter 95, the apartness-space surface of chapter 1514, the regular Cauchy criterion of chapter 1611, and the Cauchy completion criterion of chapter 1476. The packet makes the separated completion consume displayed apartness evidence directly; it does not treat apartness as a side condition stored outside the completion carrier.

Definition 9111.1 (Metric apartness completion carrier)**.** *A* $\mathsf{MetricApartnessCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, C, A, R, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Met}^{\uparrow}$ *source row whose distance reads supply the Cauchy and separation data.*
2. *C is the* $\mathsf{Completion}^{\uparrow}$ *row naming the completion request and its finite target surface.*
3. *A is the* $\mathsf{ApartnessSpace}^{\uparrow}$ *row that displays positive apartness before separated completion reads are accepted.*
4. *R is the* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *row recording the regular Cauchy schedule consumed by the completion request.*
5. *Q is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *row that admits the Cauchy-name handoff into the completion carrier.*
6. *E is the separation ledger, recording that apartness transport and zero-distance refusal are replayed along the Cauchy-name route.*
7. *H records componentwise* hsame *transport for the metric source, completion target, apartness row, Cauchy criteria, separation ledger, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from the metric source and Cauchy criteria through the apartness row and separation ledger before the completion name is exposed.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricApartnessCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{ApartnessSpace}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, C, A, R, Q, E, H, T, P, N$. It has no coordinate for host equality of completion classes, selected Cauchy representatives, untracked apartness, real trichotomy, or an ambient completeness theorem.

Theorem 9111.2 (MetricApartnessCompletion NameCert obligations)**.** *For every accepted* MetricApartnessComple *carrier $M = (S, C, A, R, Q, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, completion target, apartness row, regular Cauchy criterion, Cauchy completion criterion, separation ledger, componentwise transport, displayed replay, provenance, and local naming rows. A completion-facing consumer may read the completed name only after the route*

$$S, R, Q \longmapsto A, E \longmapsto C, H, T, P, N$$

has been displayed.

Proof. Project the carrier of Definition 9111.1. Its only rows are $S, C, A, R, Q, E, H, T, P, N$, so every accepted read must use those coordinates.

The Cauchy side is exposed by R and Q over the metric source S. The apartness side is exposed by A, and the separation ledger E records the replay that keeps zero-distance refusal and positive apartness attached to the Cauchy-name route.

The structural rows then close the local certificate. Componentwise transport H, displayed replay T, provenance P, and local naming N preserve the same finite packet, while C is available only downstream of the displayed criteria and apartness ledger. Since no hidden coordinate stores representatives or host equality, the listed rows exhaust the NameCert surface. □

Theorem 9111.3 (MetricApartnessCompletion apartness transport)**.** *Every accepted* $\mathsf{MetricApartnessCompletion}^{\uparrow}$ *read transports apartness through the displayed* $\mathsf{ApartnessSpace}^{\uparrow}$ *row A and separation ledger E. If a consumer reads the completed name without exposing A, E, H, T, P, N, then the read is outside the carrier of Definition 9111.1.*

Proof. The carrier contains apartness information only in the displayed row A and its separation ledger E. The transport row H and replay row T preserve those rows componentwise along the completion route.

Apply Definition 9111.2. It places A, E between the Cauchy criteria and the exposed completion name, with H, T, P, N stabilizing the same packet.

Thus an apartness-free read cannot be a read of the accepted carrier. It would require a coordinate for hidden apartness or host equality, and the carrier has neither. □

Theorem 9111.4 (MetricApartnessCompletion Cauchy separation)**.** *For every accepted* MetricApartnessCompletio *carrier $M = (S, C, A, R, Q, E, H, T, P, N)$, the regular Cauchy row R and Cauchy completion row Q are separated by the displayed apartness row A and separation ledger E before the completion row C becomes public. The packet therefore refuses a completion route that identifies Cauchy names by hidden representative equality.*

Proof. Project R, Q, A, E, C from the carrier. The rows R, Q provide the Cauchy schedule and admission criterion, while A, E carry the positive apartness and separation replay attached to that schedule.

Use the obligation theorem above. Its route displays S, R, Q before A, E, and only then exposes the completion-facing rows C, H, T, P, N.

Because representative equality is not a carrier coordinate, no consumer can replace the displayed apartness separation with hidden equality of Cauchy names. The only public route to C is the separated route listed in the carrier. □

Theorem 9111.5 (MetricApartnessCompletion public boundary)**.** *The public boundary of an accepted* $\mathsf{MetricApartnessCompletion}^{\uparrow}$ *packet is exactly the metric source S, completion row C, apartness row A, regular Cauchy row R, Cauchy completion row Q, separation ledger E, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$ *row N. It exports no host equality of completion classes, selected representative, untracked apartness predicate, trichotomy, or ambient completeness principle.*

Proof. The carrier definition lists the public packet as $S, C, A, R, Q, E, H, T, P, N$. No other coordinate is available to the classifier.

The NameCert obligations force a read of the completion row through the metric source, Cauchy criteria, apartness row, and separation ledger. The apartness-transport and Cauchy-separation theorems keep that route visible for every completion-facing consumer.

Therefore the public boundary is exactly the displayed packet. The excluded host equality, selected representative, untracked apartness predicate, trichotomy, and ambient completeness data have no row from which they could be exported. □

Falsifiable prediction: Every accepted $\mathsf{MetricApartnessCompletion}^{\uparrow}$ read displays a MetricUp source, Completion row, ApartnessSpace row, RegularCauchyCriterion row, CauchyCompletion-Criterion row, separation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the completion name is exposed.

Independence witness: metric, completion, apartnessspace, regularcauchycriterion, cauchy-completioncriterion: $\mathsf{MetricApartnessCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric source data, a completion target, apartness evidence, regular Cauchy scheduling, Cauchy completion admission, separation replay, transport, continuation, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricApartnessCompletion}^{\uparrow}$ packet is generated from a MetricUp source row, CompletionUp target row, ApartnessSpace row, RegularCauchyCriterion row, CauchyCompletionCriterion row, separation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 9111.1, Definition 9111.2, Definition 9111.3, Definition 9111.4, and Definition 9111.5 seed the finite separated metric completion surface that keeps apartness data on the completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricApartnessCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host equality of completion classes, selected Cauchy representatives, quotient completion semantics, real trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness transport, Cauchy separation, completion-boundary nonescape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.MetricApartnessCompletio

Chapter 9112

A Concrete Naming Certificate for $\mathsf{ApartnessCompletionReflector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ packet records the finite route by which a regular Cauchy source carrying displayed apartness evidence is reflected into the separated comparison surface of the Real completion route. It sits after the tail-diameter witness of chapter 1746, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a reflector from apartness-stable regular Cauchy data to a separated Real-facing comparison row; it does not select a completed limit, quotient a Cauchy name, assert countable choice, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9112.1 (Apartness completion reflector carrier)**.** *An* $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, W, D, R, X, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular Cauchy source row, read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *discipline and carrying the source observations used by the reflector.*
2. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *row witnessing that the displayed tail window has the requested dyadic diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite observations used by S and T.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic request, tail budget, and comparison scale.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and tolerance data.*
6. *X is the displayed apartness witness row, recording the separated comparison branch that survives the regular Cauchy reflection.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, T, W, D, R, X have been displayed.*
8. *H records componentwise* hsame *transport for the source, tail-diameter, window, tolerance, readback, apartness, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same reflector route.*

10. *P is the* Pkg *provenance over* $\mathsf{ApartnessCompletionReflector}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessCompletionReflector}^{\uparrow}}$ *row.*

The classifier accepts only packets whose separated completion read is exhausted by $S, T, W, D, R, X, E, H, C, P, N$. It has no coordinate for quotient Cauchy equality, selected representatives, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9112.2 (Apartness completion carrier obligation)**.** *For every accepted* $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ *carrier*

$$A = (S, T, W, D, R, X, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApartnessCompletionReflector}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; regular Cauchy source exposure through S; tail-diameter exposure through T; finite StreamName window exposure through W; DyadicRatCore tolerance exposure through D; RegSeqRat readback through R; displayed apartness witness exposure through X; terminal Real sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 9112.1. The analytic rows are the regular source, tail-diameter row, finite window, dyadic tolerance, regular readback, apartness witness, and Real seal.

The structural rows H, C, P, N transport, replay, source, and name exactly those analytic rows. They do not add an external completion endpoint or a quotient equality.

Thus the local certificate surface is exhausted by $S, T, W, D, R, X, E, H, C, P, N$. The excluded objects are absent from the carrier inventory, so they cannot become local obligations of the reflector. □

Theorem 9112.3 (Apartness completion separated readback)**.** *Every separated Real-facing read of an accepted* $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ *carrier factors through*

$$S \longmapsto T, W, D \longmapsto R, X \longmapsto E \longmapsto H, C, P, N.$$

The displayed apartness witness X is read after the same tail-diameter, StreamName, DyadicRatCore, and RegSeqRat rows that serve the regular Cauchy source. Hence the separated comparison is a reflected finite route, not a selected completed real or a quotient of Cauchy names.

Proof. Start from the source row S. It supplies only the displayed regular Cauchy observations and must be inspected through the finite windows W.

Read the control rows next. The tail-diameter row T and dyadic ledger D fix the requested tolerance before the regular readback row R is consumed.

The apartness witness X is then attached to the same finite route. Since X is downstream of T, W, D, R, it cannot be replaced by a detached trichotomy decision or host real inequality.

Finally the terminal Real seal E and structural rows H, C, P, N preserve the same route. The separated readback therefore remains finite and displayed. □

Theorem 9112.4 (Apartness completion reflector nonescape)**.** *No accepted* $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ *packet exports quotient Cauchy equality, a selected representative, a selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every consumer that reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal or the separated apartness witness must first read the rows $S, T, W, D, R, X, E, H, C, P, N$.*

Proof. Use the obligation surface of Definition 9112.2. It lists all rows that a local NameCert read may inspect.

Apply the separated readback theorem Definition 9112.3. It places the apartness witness and Real seal after the displayed tail-diameter, window, tolerance, and regular readback rows.

The excluded principles are not rows of the carrier. Since transport, replay, provenance, and local naming preserve only the displayed packet, no consumer can obtain those principles from an accepted reflector read. □

Falsifiable prediction: Every accepted ApartnessCompletionReflectorUp read that reaches a separated Real-facing comparison displays the regular Cauchy source, tail-diameter row, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, apartness witness, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cauchy-tail-diameter, regseqrat, streamname, dyadicratcore, real: no listed sibling alone binds a regular Cauchy source, tail-diameter control, finite stream window, dyadic tolerance, regular readback, displayed apartness witness, Real seal, transport, replay, provenance, and local naming into one completion reflector packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessCompletionReflector}^{\uparrow}}$.

Constructive story: An $\mathsf{ApartnessCompletionReflector}^{\uparrow}$ packet is generated from a regular Cauchy source, CauchyTailDiameter row, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, displayed apartness witness, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9112.1, Definition 9112.2, Definition 9112.3, and Definition 9112.4 seed the L10 route from apartness-stable regular Cauchy data to a separated Real-facing comparison surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected representatives, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-diameter exactness, finite-window readback, displayed apartness preservation, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ApartnessCompletionReflector

Chapter 9113

A Concrete Naming Certificate for $\mathsf{StreamNameRealizer}^{\uparrow}$

Following Bishop and Bridges and Troelstra–van Dalen on lawlike choice sequences, the $\mathsf{StreamNameRealizer}^{\uparrow}$ packet records finite realization data for observed stream names without converting the stream into a quotient object or a countable-choice schedule. It sits over the finite stream-name surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a lawlike finite admission route for Real and Completion consumers: it tells the consumer which finite observations are available, how they are replayed, and where regular-rational and dyadic checks enter before any $\mathsf{Real}^{\uparrow}$ seal can be read.

Definition 9113.1 (StreamNameRealizer carrier)**.** *A* $\mathsf{StreamNameRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, naming the observed stream and the finite request discipline by which windows are opened.*
2. *W is the lawlike window-realization row, recording a finite request list, its returned observations, and the monotone extension discipline for later requests.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those realized windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii, comparison budgets, and endpoint arithmetic for the realized observations.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after S, W, Q, D have been displayed.*
6. *H records componentwise* hsame *transport for source streams, realized windows, regular readback rows, dyadic tolerance rows, real seals, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from a finite observation request through S, W, Q, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamNameRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamNameRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, W, Q, D, E, H, C, P, N$. It has no coordinate for a completed infinite stream, quotient stream equality, countable choice, a host modulus selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external oracle for future observations.*

Theorem 9113.2 (StreamNameRealizer window admission)**.** *Every finite observation request admitted by an accepted* $\mathsf{StreamNameRealizer}^{\uparrow}$ *carrier*

$$R = (S, W, Q, D, E, H, C, P, N)$$

must first expose the $\mathsf{StreamName}^{\uparrow}$ *source row S and the lawlike window-realization row W. Any downstream regular-rational or real-facing read must then pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N. The admission route exports only finite realized windows and refuses quotient stream equality, countable choice, host-side future observation, and ambient Real completeness.*

Proof. Project the carrier of Definition 9113.1. The rows S and W are the only source and window-realization entrances, so an admitted finite request cannot be read before those rows are displayed.

The completion-facing part of the route is ordered by the carrier. Regular-rational readback is Q, dyadic tolerance and endpoint arithmetic are D, and the real-name seal is E. Thus a consumer cannot skip directly from a stream observation to a $\mathsf{Real}^{\uparrow}$ row.

Finally use the structural rows. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route. Since no carrier coordinate stores a completed stream, quotient equality, countable choice, or a host oracle, those claims are outside the admission theorem. □

Theorem 9113.3 (StreamNameRealizer NameCert obligations)**.** *For every accepted* $\mathsf{StreamNameRealizer}^{\uparrow}$ *carrier*

$$R = (S, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamNameRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite* BHist *packet,* $\mathsf{StreamName}^{\uparrow}$ *source exposure through S, lawlike finite-window realization through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness through D,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Read the accepted carrier of Definition 9113.1. Its displayed rows are exactly $S, W, Q, D, E, H, C, P, N$.

Apply the window-admission theorem Definition 9113.2. It fixes the order in which source, finite window realization, regular readback, dyadic tolerance, and real seal rows may be consumed.

The remaining rows are structural. Componentwise hsame transports the same finite realization data, Cont replays the displayed request, Pkg records the provenance, and the local NameCert row names the packet. No other obligation can be projected from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamNameRealizer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StreamNameRealizer}^{\uparrow}$ packet is generated from a StreamName source row, a lawlike finite-window realization row, RegSeqRat readback, DyadicRatCore tolerance data, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamNameRealizer}^{\uparrow}}$, obligation)

Scope closed: Definition 9113.1, Definition 9113.2, and Definition 9113.3 bind the finite lawlike realization route from StreamName observations to RegSeqRat, DyadicRatCore, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamNameRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed streams, quotient stream equality, countable choice, host-side future observation, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, window admission, readback exactness, tolerance exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.StreamNameRealizerUp.

Chapter 9114

A Concrete Naming Certificate for $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$

The $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$ packet realizes the observer-centered constructor certificate of section 9421.6 as finite NameCert data. It packages the OCLSD constructor packet of Definition 9421.21 together with the orthogonal section handoff of Definition 9421.23, so the projection residual is a displayed ledger row rather than an implicit Hilbert-space afterthought.

The carrier consumes the Hilbert source of chapter 91, the displayed projection route of chapter 2356, and the local-trivialization rows of chapter 97. These sibling rows make the projection residual a visible ledger entry before the orthogonal handoff, instead of treating the Hilbert bundle and projection data as ambient structure.

Falsifiable prediction: Removing the projection residual row prevents the NameCert obligations from deriving the orthogonal section handoff.

Independence witness: hilbert, projection, bundle, observer-centered-dynamics, ledger: no listed sibling is bijective to $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$ because this packet jointly carries frame selection, Hilbert bundle data, reshaping rows, closed ledger rows, projection residual rows, orthogonal section handoff, transport, replay, provenance, and local naming.

Definition 9114.1 (Observer-centered constructor certificate carrier)**.** *An* $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, B, S, L, P, R, E, H, C, G, N)$$

with the following displayed rows:

1. *F is the adjacent frame-selection row naming the displayed frame edge and successor frame.*
2. *B is the Hilbert-bundle row naming the source and successor Hilbert spaces used by the observer-centered step.*
3. *S is the isometric reshaping row carrying the section transport between the two frames.*
4. *L is the closed ledger row naming the prior ledger subspace and its rewritten closed image.*
5. *P is the orthogonal projection row against the rewritten ledger.*
6. *R is the projection residual row naming the vector subtracted from the projection row.*
7. *E is the orthogonal section handoff row, recording the residual line and accepted ledger sum.*

8. *H records componentwise* hsame *transport for frame, bundle, reshaping, ledger, projection, residual, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the constructor packet and the orthogonal handoff.*
10. *G is the* TasteGate *readiness row for the later* Nontrivial *and* FieldFaithful *instances.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, B, S, L, P, R, E, H, C, G, N$. It has no coordinate for a hidden observer, an untyped Hilbert-space completion, a projection theorem detached from the displayed ledger, or a residual-free section handoff.

Theorem 9114.2 (Observer-centered constructor certificate obligations)**.** *For every accepted* ObserverCenteredCons *carrier*

$$O = (F, B, S, L, P, R, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}}$ *surface consists exactly of adjacent frame selection through F, Hilbert-bundle exposure through B, isometric reshaping through S, closed ledger retention through L, projection against the rewritten ledger through P, projection residual exposure through R, orthogonal section handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* TasteGate *readiness through G, and local naming through N.*

Proof. Project the carrier of Definition 9114.1. The rows F, B, S, L reproduce the typed inputs of the OCLSD constructor packet, and P, R, E are the projection, residual, and accepted orthogonal section rows.

Apply Definition 9421.23. The theorem forces the residual row to be orthogonal to the rewritten ledger, so the handoff row E cannot be derived if R is absent.

The structural rows H, C, G, N preserve, replay, reserve TasteGate readiness for, and name the same packet. Since the carrier has no hidden observer, detached projection theorem, or residual-free handoff coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 9114.3 (Observer-centered projection handoff nonescape)**.** *Every consumer of an accepted* $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$ *packet that reads the orthogonal section handoff must factor through*

$$F, B, S, L, P, R, E, H, C, G, N.$$

In particular, the consumer must read the projection residual row R before using the handoff row E. The route exports a finite observer-centered constructor certificate, not an ambient Hilbert projection oracle, a hidden observer state, or an unledgered dynamics theorem.

Proof. Start from the carrier of Definition 9114.1. The handoff row E is listed after frame, bundle, reshaping, ledger, projection, and residual rows.

Use the NameCert obligations of Definition 9114.2. They identify the residual row as part of the local certificate surface before the orthogonal handoff can be exported.

The replay and transport rows keep the same finite order under consumer reads. Therefore any handoff consumer factors through the displayed route and receives no detached projection oracle or unledgered dynamics theorem. □

The Lean follow-up file `lean4/BEDC/Derived/ObserverCenteredConstructorCertificateUp/TasteGate.le` must supply `Nontrivial ObserverCenteredConstructorCertificateUp` and `FieldFaithful ObserverCentere` These instances preserve the rows $F, B, S, L, P, R, E, H, C, G, N$ and keep the projection residual row visible before the orthogonal section handoff is exported.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$ packet is generated from adjacent frame selection, Hilbert bundle data, isometric reshaping, closed ledger rows, projection rows, residual rows, orthogonal handoff rows, componentwise hsame transport, displayed Cont replay, TasteGate readiness, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 9114.1, Definition 9114.2, Definition 9114.3, the falsifiable prediction, and the independence witness expose the finite constructor certificate for observer-centered orthogonal ledger steps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a hidden observer state, ambient Hilbert projection oracle, untyped dynamics theorem, Lean verification, or bridge checking.

Upgrade path: Lean closure requires Nontrivial ObserverCenteredConstructorCertificateUp and FieldFaithful ObserverCenteredConstructorCertificateUp instances in `lean4/BEDC/Derived/ObserverCenteredConstructorCertificateUp/TasteGate.le`

Chapter 9115

A Concrete Naming Certificate for $\mathsf{FiniteCoverMesh}^{\uparrow}$

Following Bishop and Bridges on constructive compactness and finite covers, the $\mathsf{FiniteCoverMesh}^{\uparrow}$ packet records the finite mesh carrier consumed by compact-completion handoffs. It sits after the compact-metric source of chapter 129, the total-bounded finite-net row of chapter 127, the metric-space source of chapter 4230, the Real seal of chapter 38, the regular rational readback of chapter 301, and the dyadic tolerance core of chapter 291. The mesh ledger is the finite-cover consumer of the dyadic interval net and compact finite-net surfaces, so its carrier should be read together with chapter 5966 and chapter 129. The packet names the finite mesh data that compact and completion routes can consume; it is not the mesh-chain uniform-modulus theorem and it does not assert open-cover compactness.

Definition 9115.1 (Finite cover mesh carrier)**.** *A* $\mathsf{FiniteCoverMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, T, M, R, Q, D, L, U, H, E, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-metric source row whose finite-cover data are being meshed.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row supplying the finite mesh cells.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row used for radius comparison, triangle transport, and mesh containment checks.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed by completion-facing readers.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for the inspected mesh radii.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling mesh size, cover radius, and completion budget.*
7. *L is the finite mesh ledger, recording which displayed mesh cells refine which displayed cover rows.*
8. *U is the compact-completion handoff row exposed only after C, T, M, R, Q, D, L have been read.*
9. *H records componentwise* hsame *transport for the compact, total-bounded, metric, real, rational, dyadic, mesh-ledger, and handoff rows.*
10. *E records displayed* Cont *replay from compact finite-cover reads through dyadic mesh bounds into the handoff row.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverMesh}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverMesh}^{\uparrow}}$ *registration row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for an open-cover compactness principle, a least Lebesgue number, hidden centre choice, quotient equality of Real names, countable choice, or ambient completion.

Theorem 9115.2 (Finite cover mesh NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverMesh}^{\uparrow}$ *carrier* $F = (C, T, M, R, Q, D, L, U, H, E, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverMesh}^{\uparrow}}$ *obligations are exactly compact-metric source admission, total-bounded finite-net admission, metric-space comparison admission, Real seal exposure, regular rational readback, dyadic tolerance accounting, finite mesh-ledger exactness, compact-completion handoff exposure, componentwise stability transport, displayed replay, provenance, and local naming. The obligations do not include open-cover compactness, a host Lebesgue-number theorem, hidden mesh-centre selection, quotient Real equality, countable choice, or ambient completion.*

Proof. Project the carrier of Definition 9115.1. Its displayed compact data are C, T, M, so compact-metric, total-bounded, and metric comparison reads must enter through those rows.

Read the tolerance side next. The Real seal R, regular rational readback Q, and dyadic tolerance row D provide the numerical budget consumed by the finite mesh ledger L.

The handoff row U is downstream of the mesh ledger. Componentwise transport H, replay E, provenance P, and local naming N preserve the same finite route, so none of the excluded compactness, hidden-choice, quotient, choice, or ambient-completion principles has a carrier coordinate. □

Theorem 9115.3 (Finite cover mesh compact handoff)**.** *Every compact or completion consumer of an accepted* $\mathsf{FiniteCoverMesh}^{\uparrow}$ *packet must read the route*

$$C, T, M, R, Q, D, L, U, H, E, P, N$$

before using the exported mesh handoff. Thus the packet supplies concrete finite mesh NameCert data between compact finite-net sources and Real, RegSeqRat, and DyadicRatCore tolerance consumers; it does not duplicate the finite-cover mesh-chain uniform-modulus route.

Proof. Start with Definition 9115.2. It identifies the complete local surface of an accepted packet, so a compact or completion consumer cannot bypass $C, T, M, R, Q, D, L, U, H, E, P, N$.

The compact side is read through C, T, M. These rows expose compact-metric data, total-bounded finite cells, and the metric comparisons needed to relate finite mesh cells to cover rows.

The tolerance side is read through R, Q, D, L. The Real seal, regular rational readback, and dyadic tolerance row feed the finite mesh ledger, and the handoff row U becomes available only after that ledger is displayed.

Finally H, E, P, N transport, replay, package, and name the same route. Since the packet exports only this finite path, downstream compact and completion readers receive a mesh carrier rather than a host compactness theorem or the separate mesh-chain uniform-modulus certificate. □

Falsifiable prediction: A $\mathsf{FiniteCoverMesh}^{\uparrow}$ consumer is refuted whenever a compact or completion read bypasses the compact-metric source, total-bounded finite-net row, metric-space comparison row, Real seal, regular rational readback, dyadic tolerance row, finite mesh ledger, handoff row, transport, replay, provenance, or local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverMesh}^{\uparrow}$ packet is generated from CompactMetric finite-cover data, TotallyBounded finite-net cells, MetricSpace comparison rows, a Real seal, RegSeqRat readback, DyadicRatCore tolerance accounting, a finite mesh ledger, compact-completion handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9115.1, Definition 9115.2, and Definition 9115.3 seed the finite mesh carrier consumed by compact finite-net and completion-tolerance routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverMesh}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.FiniteCoverMeshUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host Lebesgue-number existence, hidden mesh-centre selection, quotient Real equality, countable choice, ambient completion, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite mesh-ledger exactness, compact-completion handoff nonescape, tolerance-route compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverMeshUp.

Chapter 9116

Menger Sponge NameCert Construction

Following Menger and classical fractal topology, the $\mathsf{MengerSponge}^{\uparrow}$ packet records the standard ternary-cube sponge as finite BEDC naming data. It sits beside the square deletion surface of chapter 3960, the Cantor-address source of chapter 1475, and the finite contraction-family row of chapter 7342. The packet names only inspected cube stages, deleted middle slabs, boundary incidence, coordinate readback, transport, replay, provenance, and local naming; it does not import a completed compact subset of the cube, a selected infinite branch, Hausdorff dimension, measure computation, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9116.1 (Menger sponge carrier)**.** *A* $\mathsf{MengerSponge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (C, D, B, A, W, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the ternary cube-stage row. It records an inspected finite word over the* 27 *subcube positions, the displayed depth, and the retained cube cell at that depth.*
2. *D is the deleted middle-slab ledger. For each inspected cube address it records the three coordinate directions and refuses exactly those subcubes whose address lies in at least two middle-coordinate slabs.*
3. *B is the boundary-incidence row. It records which retained faces, edges, and corner incidences remain visible after the deleted-slab ledger has been applied.*
4. *A is the finite Cantor-address row read from chapter 1475; it supplies only inspected prefixes and no selected infinite branch.*
5. *W is the coordinate* $\mathsf{StreamName}^{\uparrow}$ *window row for the three coordinate schedules attached to the retained cube.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those coordinate schedules, with dyadic tolerance comparisons routed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any terminal seal is exposed.*
7. *E is the triple of* $\mathsf{Real}^{\uparrow}$ *coordinate seals obtained from the same cube-stage, deleted-slab, boundary, and coordinate-window rows.*
8. *H records componentwise* hsame *transport for $C, D, B, A, W, Q, E, T, P, N$.*

9. *T records displayed* Cont *replay from a sponge request through retained cube stages, deleted slabs, boundary incidence, coordinate windows, rational readbacks, and Real coordinate seals.*
10. *P is the* Pkg *provenance over* $\mathsf{MengerSponge}^{\uparrow}$, $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{IteratedFunctionSystem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerSponge}^{\uparrow}}$ *row.*

The classifier accepts a packet only through $C, D, B, A, W, Q, E, H, T, P, N$. It has no coordinate for a completed subset of $[0,1]^3$, hyperspace equality, selected infinite address, dimension or measure data, quotient equality of coordinate triples, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MengerSpongeUp.MengerSpongeTasteGate single carrier alignment`

Theorem 9116.2 (Menger sponge NameCert obligations)**.** *For every accepted* $\mathsf{MengerSponge}^{\uparrow}$ *carrier*

$$M = (C, D, B, A, W, Q, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MengerSponge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; ternary cube-stage exposure through C; deleted middle-slab exclusion through D; boundary incidence through B; finite Cantor-address exposure through A; coordinate window exposure through W; regular rational readback through Q; Real coordinate sealing through E; componentwise transport through H; displayed replay through T; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 9116.1. The accepted packet displays exactly the ternary cube-stage, deleted middle-slab, boundary-incidence, finite address, coordinate-window, rational-readback, Real-seal, and structural rows.

The sponge-specific obligations are exhausted by C, D, B. A consumer first reads the inspected ternary cube address, then the deleted-slab ledger for that address, and only then the boundary incidences that remain available.

The coordinate obligations are exhausted by A, W, Q, E. The Cantor-address row supplies finite prefixes, the StreamName row opens the finite coordinate windows, the RegSeqRat row reads the scheduled rational data through the dyadic tolerance ledger, and the terminal Real row seals exactly those three coordinates.

Finally H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no completed subset, hyperspace equality, selected branch, dimension row, measure row, quotient coordinate equality, or ambient-completeness coordinate, none of those objects can become a local NameCert obligation. □

Theorem 9116.3 (Menger sponge Cantor-cube handoff)**.** *Every accepted* $\mathsf{MengerSponge}^{\uparrow}$ *coordinate read factors through the finite route*

$$\mathsf{CantorSpace}^{\uparrow}, \mathsf{IteratedFunctionSystem}^{\uparrow} \longmapsto C, D, B, A \longmapsto W, Q, E \longmapsto H, T, P, N.$$

The route first displays the Cantor-prefix and finite contraction-family sources, then reads retained ternary cube stages, deleted middle slabs, boundary incidence, coordinate windows, regular rational readback, and terminal Real coordinate seals. It exports no completed fractal subset, selected infinite branch, hyperspace fixed point, quotient equality of coordinate triples, Hausdorff-dimension computation, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted sponge consumer. By Definition 9116.1, the consumer enters through the displayed cube-stage row C and the source rows named by $\mathsf{CantorSpace}^{\uparrow}$ and $\mathsf{IteratedFunctionSystem}^{\uparrow}$.

Apply Definition 9116.2. It forces every retained-cell read through D and B, every finite address read through A, and every coordinate read through W, Q, E before the structural rows are inspected.

The sibling chapters supply only their own finite routes: chapter 1475 supplies finite prefix data, chapter 7342 supplies a finite contraction-family interface, and chapter 3960 supplies the two-dimensional deletion sibling. None adds a carrier coordinate for a completed sponge subset, selected branch, hyperspace fixed point, dimension, measure, quotient coordinate equality, or ambient completeness. Hence every accepted coordinate consumer remains on the displayed Cantor-cube handoff. □

Falsifiable prediction: An accepted $\mathsf{MengerSponge}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ coordinate seal without first displaying the retained ternary cube stage, deleted middle-slab ledger, boundary-incidence row, finite Cantor-address prefix, StreamName coordinate windows, and RegSeqRat readback refutes the carrier surface.

Independence witness: sierpinski-carpet, cantorspace, iterated-function-system, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole $\mathsf{MengerSponge}^{\uparrow}$ packet, because the sponge certificate binds three-dimensional ternary cube stages, deleted middle slabs, boundary incidence, coordinate readback, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerSponge}^{\uparrow}}$.

Constructive story: A $\mathsf{MengerSponge}^{\uparrow}$ packet is generated from retained ternary cube-stage rows, deleted middle-slab ledgers, boundary-incidence rows, finite Cantor-address prefixes, StreamName coordinate windows, RegSeqRat readbacks, DyadicRatCore tolerance checks, RealUp coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MengerSponge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9116.1, Definition 9116.2, and Definition 9116.3 seed the finite ternary-cube sponge carrier and its Cantor-cube coordinate handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MengerSponge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MengerSpongeUp.MengerSpongeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close completed sponge subsets, hyperspace fixed points, selected infinite branches, Hausdorff dimension, measure computation, quotient equality of coordinate triples, or bridge checking.

Upgrade path: Obligation closure requires a Lean-side MengerSpongeUp carrier with checked cube-stage exposure, deleted middle-slab exclusion, boundary incidence, coordinate readback, Real coordinate sealing, transport, replay, provenance, and local naming.

Chapter 9117

A Concrete Naming Certificate for $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$

Following Kalman and Kailath on finite-dimensional realization theory, the $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$ packet records the finite BEDC route from displayed input-output transfer readback rows to a state-space realization surface. It sits after the state-space carrier of chapter 329, the minimal-realization packet of chapter 6055, the observability and controllability certificates of chapter 294 and chapter 297, and the finite $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ rows of chapter 49 and chapter 48. The packet names a realization certificate, not an analytic equality between host transfer functions.

Definition 9117.1 (TransferFunctionStateSpaceRealization carrier)**.** *A* $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (U, Y, S, G, A, B, C, D, Q, O, E, H, K, P, N)$$

with the following displayed rows:

1. *U is the input row, read through a finite vector or matrix-facing source.*
2. *Y is the output row, read through the same finite input-output discipline.*
3. *S is the consumed* $\mathsf{StateSpaceModel}^{\uparrow}$ *state-space realization row.*
4. *G is the transfer readback row naming the finite input-output response surface.*
5. *A, B, C, D are the transition, input, output, and feedthrough rows consumed from the state-space realization.*
6. *Q is the* $\mathsf{ControlControllability}^{\uparrow}$ *row used by realization-facing consumers.*
7. *O is the* $\mathsf{ControlObservability}^{\uparrow}$ *row used by realization-facing consumers.*
8. *E is the realization compatibility row saying that G is read only through $U, Y, S, A, B, C, D, Q, O$.*
9. *H records componentwise* hsame *transport for input, output, state-space, transfer readback, controllability, observability, compatibility, provenance, and local naming rows.*
10. *K records displayed* Cont *replay from input-output requests through the state-space realization and transfer readback rows.*
11. *P is the* Pkg *provenance over* $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{MinimalRealization}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}}$ *row.*
The carrier has no coordinate for analytic transfer-function equality, selected canonical realization, host quotient state space, infinite trajectory semantics, or external minimality theorem.

Theorem 9117.2 (TransferFunctionStateSpaceRealization NameCert obligations)**.** *For the carrier of Definition 9117.1, the local* $\mathsf{NameCert}_{\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}}$ *surface consists exactly of input admission through U, output admission through Y, state-space realization admission through S, A, B, C, D, transfer readback through G, controllability through Q, observability through O, realization compatibility through E, componentwise transport through H, displayed replay through K, package provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 9117.1. The accepted packet displays exactly $U, Y, S, G, A, B, C, D, Q, O$ so every public read is one of these rows or a replay route through them.

The state-space side is finite. The realization row S supplies the transition, input, output, and feedthrough rows A, B, C, D, while the controllability and observability rows Q, O record the finite reachability and observation surfaces consumed by realization-facing chapters.

The transfer side is bounded by compatibility. The row E permits G to be read only after the displayed input-output and state-space rows have been exposed, so the packet never needs an analytic transfer-function equality outside the carrier.

The structural rows close the certificate. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row repack the same finite data, while host quotient state spaces, selected canonical realizations, infinite trajectories, and external minimality theorems remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$ packet is generated from input and output rows, a StateSpaceModel realization row, transfer readback, transition, input, output, feedthrough, controllability, observability, compatibility, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}}$, seed)

Scope closed: Definition 9117.1 and Definition 9117.2 seed the finite transfer-function/state-space realization surface over StateSpaceModel, MinimalRealization, ControlObservability, ControlControllability, Matrix, and LinearMap rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim analytic transfer-function equality, selected canonical realization, host quotient state space, infinite trajectory semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, state-space realization readback, transfer compatibility, controllability and observability consumption, transport, replay, provenance, and local naming for the same finite packet.

Chapter 9118

A Concrete Naming Certificate for $\mathsf{CauchyIdeal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyIdeal}^{\uparrow}$ packet records rounded Cauchy ideal data as a finite family of observation rows before any quotient completion, selected limit, or ambient completeness theorem is available. It sits after the rounded filter route of chapter 333, the finite stream naming surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the ideal carrier and its local certificate obligations; it does not identify Cauchy representatives or choose a completed real.

Definition 9118.1 (Cauchy ideal carrier)**.** *A* $\mathsf{CauchyIdeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, R, D, S, Q, E, O, H, T, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose neighborhoods supply the admitted Cauchy observations.*
2. *R is the rounded directed ideal row, listing finitely many observation rows closed under displayed coarsening and carrying a finite common-refinement witness for any two listed rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii and comparison budgets used by the displayed observations.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation-window row through which the ideal is inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the same stream windows and dyadic tolerance data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after F, R, D, S, Q have been displayed.*
7. *O is the local rounded-obligation ledger, recording inhabitedness, upward closure under displayed weakening, directed refinement, and Cauchy compatibility inside the finite packet.*
8. *H records componentwise* hsame *transport for the filter source, rounded ideal row, dyadic tolerance row, stream windows, regular readback, real seal, obligation ledger, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from an ideal read through $F, R, D, S, Q, E, O, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyIdeal}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyIdeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, D, S, Q, E, O, H, T, P, N$. It has no coordinate for a quotient of filters, equality of completed reals, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9118.2 (CauchyIdeal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIdeal}^{\uparrow}$ *carrier $I = (F, R, D, S, Q, E, O, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIdeal}^{\uparrow}}$ *surface consists exactly of CauchyFilter source exposure through F, rounded directed ideal exposure through R, DyadicRatCore tolerance accounting through D, StreamName window exposure through S, RegSeqRat readback through Q, terminal Real sealing through E, rounded-obligation coverage through O, componentwise transport through H, displayed replay through T, provenance through P, and local naming through N. A Real or Completion consumer may read the ideal only after the route*

$$F, R, D \longmapsto S, Q \longmapsto E, O, H, T, P, N$$

has been displayed.

Proof. Project the carrier of Definition 9118.1. Its analytic rows are F, R, D, S, Q, E, O, and its structural rows are H, T, P, N.

The rounded ideal row R is read together with the CauchyFilter source F and dyadic tolerance row D. Thus inhabitedness, displayed weakening, directed refinement, and Cauchy compatibility are obligations of the same finite packet rather than properties imported from a quotient completion.

The readback rows remain downstream. The finite StreamName window S and RegSeqRat row Q expose only the listed observations, while the Real seal E is available only after those observations and tolerances have been replayed.

Finally O, H, T, P, N cover the local obligations, transport them componentwise, replay them, source them, and name them. Since no hidden coordinate stores quotient equality, representative choice, countable choice, or ambient completeness, the displayed rows exhaust the NameCert surface. □

Theorem 9118.3 (CauchyIdeal rounded readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyIdeal}^{\uparrow}$ *carrier reads through a rounded directed family of finite Cauchy observation rows over* $\mathsf{CauchyFilter}^{\uparrow}$. *The consumer may inspect the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming, but receives no quotient completion or selected limit.*

Proof. Use Definition 9118.2. It places the finite CauchyFilter source, rounded ideal row, and dyadic tolerance data before the stream and regular-rational readback rows.

The rounded-obligation ledger O records exactly the finite ideal laws visible to the consumer: inhabitedness, weakening, directed refinement, and Cauchy compatibility. These laws are replayed through T and transported through H without adding a completed object.

Therefore a downstream read can be a read of the rounded Cauchy ideal packet only through the displayed rows. A quotient completion, selected limit, or host equality claim would require a coordinate not present in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{CauchyIdeal}^{\uparrow}$ consumer read displays a CauchyFilter source, rounded directed ideal row, DyadicRatCore tolerance ledger, StreamName finite observation window, RegSeqRat readback, RealUp seal, rounded-obligation ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert row before the ideal can be consumed.

Independence witness: cauchyfilter, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyIdeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rounded directed Cauchy observation rows, finite stream windows, regular rational readback, dyadic tolerance accounting, a Real seal, obligation coverage, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIdeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIdeal}^{\uparrow}$ packet is generated from a CauchyFilter source, rounded directed ideal row, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, rounded-obligation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIdeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9118.1, Definition 9118.2, and Definition 9118.3 seed the finite rounded Cauchy ideal route for Bishop-style completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIdeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, equality of completed reals, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rounded-directed ideal coverage, Cauchy compatibility, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyIdealUp.

Chapter 9119

Analytic Tail Budget NameCert Construction

Following Bishop–Bridges Cauchy tail estimates, the $\mathsf{AnalyticTailBudget}^{\uparrow}$ packet records the finite tolerance budget by which analysis-facing arguments hand a tail claim to Real and Completion consumers. It realizes the analysis-interface route of chapter 9358 as concrete NameCert data and sits over the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Cauchy modulus route of chapter 292, and the completion boundary of chapter 1639. The packet names a finite tail-control row only; it does not assert ambient completeness, choose a limit, quotient streams, decide trichotomy, or identify host real equality.

Definition 9119.1 (Analytic tail-budget carrier)**.** *An* $\mathsf{AnalyticTailBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (\varepsilon, D, M, S, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. ε *is the requested finite tolerance row, presented as dyadic precision data and not as a detached host-real parameter.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger. It records the dyadic comparison cells, tail radius, and tolerance arithmetic used by the request.*
3. M *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that turns the tolerance request into a displayed finite tail index.*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row opened at and beyond that index.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected windows and their regular rational tail estimates.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* D, M, S, R *have been displayed.*
7. B *is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *completion-facing boundary row; it records that the same tail data may be consumed by completion arguments without selecting a completed limit.*
8. H *records componentwise* hsame *transport for* $\varepsilon, D, M, S, R, E, B, C, P, N$.
9. C *records displayed* Cont *replay from a tail request through dyadic budget, Cauchy modulus, finite windows, regular readback, Real seal, and completion boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{AnalyticTailBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AnalyticTailBudget}^{\uparrow}}$ *row.*

The classifier compares packets only through $\varepsilon, D, M, S, R, E, B, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, ambient completeness of $\mathsf{Real}^{\uparrow}$*, trichotomy, countable choice, or host real equality.*

Theorem 9119.2 (Analytic tail-budget NameCert obligations)**.** *For every accepted* $\mathsf{AnalyticTailBudget}^{\uparrow}$ *carrier*

$$A = (\varepsilon, D, M, S, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AnalyticTailBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; dyadic tolerance exposure through ε and D; tail-index extraction through M; finite observation through S; regular rational readback through R; terminal Real sealing through E; completion-facing boundary exposure through B; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 9119.1. The accepted packet exposes precisely the tolerance, dyadic budget, modulus, finite-window, readback, Real-seal, completion-boundary, and structural rows.

The tolerance obligation is exhausted by ε, D, M. The request is read as dyadic data, the ledger fixes the comparison cells and radius, and the Cauchy modulus row turns that budget into the displayed tail index.

The analytic readback obligation is exhausted by S, R, E, B. The StreamName row opens the finite windows, the RegSeqRat row reads the regular rational estimates, the Real seal remains downstream of that readback, and the Bishop regular completion row records the completion-facing boundary without selecting a limit.

Finally H, C, P, N transport, replay, source, and name only the displayed rows. Since no carrier coordinate stores quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient completeness, none of those objects can become a local NameCert obligation. □

Theorem 9119.3 (Analytic tail-budget Real completion route)**.** *Every accepted Real-facing or Completion-facing read of an* $\mathsf{AnalyticTailBudget}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{BishopRegularCauchyCompletion}$$

The route makes a tail request public only after the dyadic budget, modulus index, finite observation windows, and regular rational readback have been displayed. It exports no selected limit, quotient stream equality, countable-choice schedule, trichotomy decision, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted Real-facing or Completion-facing consumer. By Definition 9119.1, the consumer must read the dyadic budget row D and the Cauchy modulus row M before any tail window is available.

Apply Definition 9119.2. It places S and R between the modulus index and the terminal rows E and B, so neither the Real seal nor the completion-facing boundary has an independent entry point.

The sibling chapters supply the listed interfaces only as finite rows: dyadic tolerances, finite stream windows, regular rational readback, Cauchy modulus indices, Real seals, and Bishop regular

completion boundaries. None adds a quotient stream, selected limit, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness coordinate. Therefore the accepted consumer remains on the displayed Real completion route. □

Falsifiable prediction: An accepted $\mathsf{AnalyticTailBudget}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ seal or the $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ boundary without first displaying the dyadic budget, Cauchy modulus index, StreamName windows, and RegSeqRat readback refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchymodulus, bishop-regular-cauchy-completion: no listed sibling supplies the whole $\mathsf{AnalyticTailBudget}^{\uparrow}$ packet, because the tail-budget certificate binds tolerance request, dyadic budget, modulus index, finite observation, regular readback, terminal seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AnalyticTailBudget}^{\uparrow}}$.

Constructive story: An $\mathsf{AnalyticTailBudget}^{\uparrow}$ packet is generated from a finite dyadic tolerance request, DyadicRatCore budget ledger, CauchyModulus tail index, StreamName observation windows, RegSeqRat readback, RealUp seal, BishopRegularCauchyCompletionUp boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AnalyticTailBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9119.1, Definition 9119.2, and Definition 9119.3 seed the finite analysis-interface tail-control route into Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AnalyticTailBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires a Lean-side AnalyticTailBudgetUp carrier with checked tolerance exposure, dyadic budget routing, Cauchy modulus extraction, finite observation, regular readback, Real and completion boundary sealing, transport, replay, provenance, and local naming.

Chapter 9120

A Concrete Naming Certificate for ZeckendorfReadoutCoefficientStrata$^{\uparrow}$

The ZeckendorfReadoutCoefficientStrata$^{\uparrow}$ packet realizes the coefficient-strata readout route as finite Zeckendorf carry, phase-shell, running-coordinate, and boundary-flux rows before any physical scalar fit is accepted. It sits after the phase shell of chapter 8811, the carry classifier of chapter 567, the running coordinate route of chapter 9092, and the static boundary flux of chapter 8769. The chapter names the finite carrier that keeps the static impedance coefficient, beta-shell coefficient, and curvature-residual coefficient in separate rows.

Definition 9120.1 (Zeckendorf readout coefficient strata carrier)**.** *A* ZeckendorfReadoutCoefficientStrata$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (P, K, A, G, S, B, C, L, H, T, R, N)$$

with the following displayed rows:

1. *P is the* ZeckendorfPhaseShell$^{\uparrow}$ *phase-shell row.*
2. *K is the* ZeckendorfCarryClassifier$^{\uparrow}$ *carry-classifier row.*
3. *A is the* AxisZeckendorfRunningCoordinate$^{\uparrow}$ *running-coordinate row.*
4. *G is the* StaticGreenBoundaryFlux$^{\uparrow}$ *boundary-flux row.*
5. *S is the static impedance stratum row, recording the displayed $8/9$ coefficient as a boundary-flux readout.*
6. *B is the beta-shell stratum row, recording the displayed $2/3$ coefficient as a phase-image readout.*
7. *C is the curvature-residual stratum row, recording the displayed $4/9$ coefficient as a one-way fold-mismatch readout.*
8. *L is the local layer-separation ledger, refusing any read that identifies S, B, C as competing estimates of a single scalar.*
9. *H records componentwise* hsame *transport for the phase-shell, carry-classifier, running-coordinate, boundary-flux, three coefficient strata, ledger, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay from a coefficient read through $P, K, A, G, S, B, C, L, H, R, N$.*
11. *R is the* Pkg *provenance over* ZeckendorfReadoutCoefficientStrata$^{\uparrow}$*,* ZeckendorfPhaseShell$^{\uparrow}$*,* ZeckendorfCarryClassif AxisZeckendorfRunningCoordinate$^{\uparrow}$*,* StaticGreenBoundaryFlux$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $P, K, A, G, S, B, C, L, H, T, R, N$. It has no coordinate for an unstratified scalar fit, floating coefficient tuning, hidden physical calibration, quotient phase equality, or host-field semantics.

Theorem 9120.2 (Zeckendorf readout coefficient strata NameCert obligations)**.** *For every accepted* $\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}$ *carrier $Z = (P, K, A, G, S, B, C, L, H, T, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}}$ *surface consists exactly of phase-shell exposure through P, carry-classifier exposure through K, running-coordinate exposure through A, static boundary-flux exposure through G, coefficient-row exposure through S, B, C, layer-separation refusal through L, componentwise transport through H, displayed replay through T, provenance through R, and local naming through N.*

Proof. Project the carrier of Definition 9120.1. The source rows are P, K, A, G, the coefficient rows are S, B, C, and the structural rows are L, H, T, R, N.

The three coefficient rows have distinct readout sources. The static row S is bound to the boundary-flux row G, the beta-shell row B is bound to phase-image and running-coordinate data through P, K, A, and the curvature-residual row C is bound to the one-way fold-mismatch boundary.

The ledger L refuses a collapse of these strata into a single fit parameter. Transport and replay through H, T preserve the same row separation, while R, N source and name the packet.

Since no carrier coordinate stores an unstratified scalar, floating calibration, quotient phase equality, or host-field interpretation, the listed rows exhaust the NameCert obligations. □

Theorem 9120.3 (Zeckendorf coefficient readout separation)**.** *Every accepted* $\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}$ *read keeps the displayed coefficients $8/9$, $2/3$, and $4/9$ in three separate strata. The read may transport and replay the strata through H, T, R, N, but it cannot replace the row separation by a single low-energy scalar fit.*

Proof. Apply Definition 9120.2. It exposes S, B, C as three rows rather than as one row with three candidate values.

The source rows separate the meaning of the coefficients. Static impedance reads through G, beta-shell reads through P, K, A, and curvature residual reads through the one-way mismatch row C together with the ledger L.

Thus a consumer that collapses the coefficients into one scalar has left the carrier. The packet has transport, replay, provenance, and naming rows, but no coordinate that can erase the displayed stratum separation. □

The Lean TasteGate follow-up for $\mathsf{BEDC.Derived.ZeckendorfReadoutCoefficientStrataUp}$ must provide explicit $\mathsf{Nontrivial}$ and $\mathsf{FieldFaithful}$ instances for the same carrier rows $P, K, A, G, S, B, C, L, H, T, R, N$ before any checked taste-gate target is recorded.

Falsifiable prediction: Every accepted $\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}$ consumer read displays Zeckendorf phase-shell, carry-classifier, running-coordinate, static boundary-flux, static impedance, beta-shell, curvature-residual, layer-separation, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a coefficient stratum can be consumed.

Independence witness: zeckendorf-phase-shell, zeckendorf-carry-classifier, axis-zeckendorf-running-coordinate, static-green-boundary-flux: $\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}$ is not bijective to any

listed sibling because it jointly carries phase-shell, carry-classifier, running-coordinate, boundary-flux, three coefficient strata, layer-separation refusal, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}$ packet is generated from Zeckendorf phase-shell, carry-classifier, running-coordinate, static boundary-flux, static impedance, beta-shell, curvature-residual, layer-separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9120.1, Definition 9120.2, and Definition 9120.3 seed the finite coefficient-strata readout route for Zeckendorf consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeckendorfReadoutCoefficientStrata}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical scalar fitting, host-field semantics, quotient phase equality, checked TasteGate evidence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-row separation, layer-refusal coverage, Nontrivial TasteGate evidence, FieldFaithful TasteGate evidence, transport, replay, provenance, and local naming for BEDC.Derived.ZeckendorfReadoutCoefficientStrataUp.

Chapter 9121

A Concrete Naming Certificate for $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's compactness discipline for uniform spaces, the $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ packet records the finite BEDC surface by which a uniformly continuous map on compact metric data extends across displayed completion names. It sits over the compact-metric row of chapter 129, the completion row of chapter 95, the uniform extension ledger of chapter 769, the Cauchy completion functor of chapter 423, and the finite Real and Completion corridor through chapter 126, chapter 301, chapter 291, and chapter 38. The packet exposes compact finite-net evidence and completion-extension uniqueness as displayed rows; it does not replace them by open-cover compactness, quotient completions, or selected limits.

Definition 9121.1 (Bishop compact completion-extension carrier)**.** *A* $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, C, U, F, S, R, D, Q, L, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row whose compact data are read only through displayed finite-net or total-boundedness witnesses.*
2. *C is the* $\mathsf{Completion}^{\uparrow}$ *row naming the source and target completion surfaces used by the extension read.*
3. *U is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row carrying the source graph, finite uniform modulus, target convergence handoff, and separated uniqueness ledger.*
4. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row recording the functorial source-to-completion and completion-consumer routes for the same Cauchy data.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which compact source names and completion names are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling compact-net mesh size, Cauchy radius, and uniform-modulus budget.*
8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D and the completion-extension rows have been displayed.*

9. *L is the finite compact-net ledger: for each displayed dyadic tolerance it lists net centers, incidence rows, image-window rows, and the extension-uniqueness comparison row served by K, U, F.*
10. *H records componentwise* hsame *transport for $K, C, U, F, S, R, D, Q, L, P, N$.*
11. *T records displayed* Cont *replay from a compact-net request through the completion-extension route.*
12. *P is the* Pkg *provenance over* $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactCompletionExtension}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected limits, quotient completion equality, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9121.2 (Bishop compact completion-extension NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ *carrier*

$$E = (K, C, U, F, S, R, D, Q, L, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompactCompletionExtension}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-metric row, completion row, uniform-extension row, Cauchy-completion-functor row, finite stream windows, regular-rational readback, dyadic tolerance ledger, terminal Real seal, finite compact-net ledger, componentwise transport, displayed replay, provenance, and local naming. Every completion-extension read must display the finite-net row L and the uniqueness route supplied by U and F before it exposes a completed value.*

Proof. Project the carrier of Definition 9121.1. Its coordinates are exactly $K, C, U, F, S, R, D, Q, L, H, T, P, N$, so carrier admission and local naming can inspect only those rows.

Read the compact side next. The row K supplies compact metric data only through displayed finite-net evidence, while L records the tolerance-indexed net centers, incidence rows, and image windows used by the extension read.

Read the completion side through C, U, F. The completion row names the source and target completion surfaces, the uniform-extension row carries the finite modulus and separated uniqueness ledger, and the Cauchy-completion-functor row keeps the source-to-completion and completion-consumer routes attached to the same Cauchy data.

Finally read the numerical corridor S, R, D, Q and the structural rows H, T, P, N. Finite windows, regular-rational readback, and dyadic budgets precede the terminal Real seal, while transport, replay, provenance, and local naming preserve the same packet. No excluded compactness, choice, quotient, host equality, or ambient completeness principle is a carrier coordinate. □

Theorem 9121.3 (Bishop compact completion-extension finite-net handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Completion}^{\uparrow},\ \mathsf{UniformContinuousExtension}^{\uparrow},\ \mathsf{CauchyCompletionFunctor}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRa}$$

as displayed by $K, C, U, F, S, R, D, Q, L, H, T, P, N$. In particular, compactness enters only through the finite compact-net ledger L, and completion-extension uniqueness enters only through the rows U and F before the terminal $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the NameCert obligations of Definition 9121.2. They force every accepted read to expose the compact row, completion row, uniform-extension row, Cauchy functor row, finite windows, rational readback, dyadic ledger, compact-net ledger, and structural rows.

The compact handoff is finite. For each requested dyadic tolerance, L lists the net centers and incidence rows read from K, then records the image windows that the extension row may inspect. Thus the consumer receives compact-net rows, not an open-cover subcover principle.

The extension handoff is unique only inside the displayed completion route. Rows U and F say that two extension reads agreeing on the same source completion windows and uniform-modulus data have the same separated target read. This is a finite uniqueness ledger, not quotient completion equality and not host function equality.

The Real-facing corridor then reads S, R, D, Q. The terminal seal Q is reached only after the stream windows, regular-rational readback, dyadic budgets, compact-net ledger, and completion-extension rows have all been displayed, so the handoff exports no selected limit, countable choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ consumer must display CompactMetric, Completion, UniformContinuousExtension, CauchyCompletionFunctor, StreamName, RegSeqRat, DyadicRatCore, Real, finite compact-net, hsame, Cont, Pkg, and local NameCert rows before reading a completion-extension value.

Independence witness: compactmetric, completion, uniform-continuous-extension, cauchycompletionfunctor, streamname, regseqrat, dyadicratcore, real: no listed sibling carries both the finite compact-net ledger and the completion-extension uniqueness route in one local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactCompletionExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactCompletionExtension}^{\uparrow}$ packet is generated from CompactMetric finite-net data, Completion surfaces, a UniformContinuousExtension ledger, a CauchyCompletionFunctor route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactCompletionExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9121.1, Definition 9121.2, and Definition 9121.3 seed the finite compact completion-extension route over CompactMetric, Completion, UniformContinuousExtension, CauchyCompletionFunctor, StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactCompletionExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient completion equality, countable choice, host function equality, ambient RealUp completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exactness, completion-surface routing, uniform-extension uniqueness, Cauchy-functor handoff, StreamName and RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactCompletionExten

Chapter 9122

A Concrete Naming Certificate for CompactUniformNetSelection$^{\uparrow}$

Following Bishop and Bridges on compact finite nets and uniform continuity, the CompactUniformNetSelection$^{\uparrow}$ packet records the finite selected-net handoff by which compact metric source data and continuous-map local moduli are turned into a displayed uniform-modulus consumer row. It sits beside the finite cover mesh carrier of chapter 9115, reads compact source rows through chapter 129, continuous-map rows through chapter 130, terminal seals through chapter 38, regular rational readbacks through chapter 301, and dyadic tolerance arithmetic through chapter 291. The packet supplies a finite selected-net ledger for compact-uniform-continuity consumers; it does not assert open-cover compactness, an external point selector, quotient Real equality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 9122.1 (Compact uniform net selection carrier)**.** *A* CompactUniformNetSelection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, M, Q, R, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* CompactMetric$^{\uparrow}$ *source row whose finite compact-net data are inspected.*
2. *F is the* ContinuousMap$^{\uparrow}$ *row carrying the pointwise modulus data for the displayed graph.*
3. *M is the finite selected-net row, listing the compact probes and their local-radius witnesses.*
4. *Q is the finite lower-bound fold row over the selected probe radii.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the folded rational tolerance data.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the requested precision with the selected radius.*
7. *L is the local coverage ledger saying which displayed probe covers each inspected source point.*
8. *U is the compact-uniform handoff row exposed to* UniformModulus$^{\uparrow}$ *and compact-uniform-continuity consumers.*
9. *H records componentwise* hsame *transport for the compact, continuous, net, fold, readback, dyadic, coverage, and handoff rows.*

10. *C records displayed* Cont *replay from compact finite-net reads and continuous-map local moduli into the uniform handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformNetSelection}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteCoverMesh}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformNetSelection}^{\uparrow}}$ *registration row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an open-cover compactness principle, an external center selector, hidden Lebesgue-number existence, quotient Real equality, countable choice, or ambient completion.

Theorem 9122.2 (Compact uniform net selection NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformNetSelection}^{\uparrow}$ *carrier* $S = (K, F, M, Q, R, D, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformNetSe}}$ *obligations are exactly compact-metric source admission, continuous-map local-modulus admission, selected finite-net exposure, finite lower-bound fold exposure, regular rational readback, dyadic tolerance comparison, coverage-ledger exactness, compact-uniform handoff exposure, componentwise stability transport, displayed replay, provenance, and local naming. The obligations do not include open-cover compactness, hidden center choice, quotient Real equality, countable choice, an ambient completion theorem, or a host Lebesgue-number principle.*

Proof. Project the carrier of Definition 9122.1. The compact and continuity source rows are K and F, so selected probes and local moduli must be read before any uniform handoff is available.

Read the finite selection route next. The selected-net row M supplies only displayed probes, the fold row Q records the finite lower-bound comparison, and the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows R, D expose the rational tolerance data used by the coverage ledger L.

The handoff row U is downstream of that ledger. Componentwise transport H, replay C, provenance P, and local naming N preserve the same finite route, so none of the excluded compactness, choice, quotient, completion, or host Lebesgue-number principles has a carrier coordinate. □

Theorem 9122.3 (Compact uniform net selection vision handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformNetSelection}^{\uparrow}$ *packet must read the route*

$$K, F, M, Q, R, D, L, U, H, C, P, N$$

before using the exported uniform-modulus handoff. Thus the compact finite-net source, continuous-map local-modulus row, regular rational readback, dyadic tolerance comparison, and finite cover mesh relation are exposed as concrete NameCert data; the consumer receives no open-cover compactness theorem, external selector, quotient Real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 9122.2. It identifies the local surface of an accepted selected-net packet and forces every consumer through the compact source, continuous-map source, selected-net, fold, readback, dyadic, coverage, and handoff rows.

The compact side is read through K, M, L. These rows display the compact finite-net probes and the local coverage ledger, matching the finite mesh discipline of chapter 9115.

The analytic side is read through F, Q, R, D, U. Continuous-map local moduli feed the finite lower-bound fold, the regular rational and dyadic rows expose the tolerance comparison, and the uniform handoff is available only after those rows have been displayed.

Finally H, C, P, N transport, replay, package, and name the same route. Since the packet exports only this finite selected-net path, the compact-uniform-continuity vision receives a concrete NameCert handoff and no hidden compactness or choice principle. □

Falsifiable prediction: A $\mathsf{CompactUniformNetSelection}^{\uparrow}$ consumer is refuted whenever a compact-uniform handoff bypasses the compact-metric source, continuous-map local-modulus row, selected finite-net row, lower-bound fold, regular rational readback, dyadic tolerance comparison, coverage ledger, transport, replay, provenance, or local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformNetSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformNetSelection}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap local-modulus rows, selected probe data, a finite lower-bound fold, RegSeqRat readback, DyadicRatCore tolerance comparison, a coverage ledger, compact-uniform handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformNetSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9122.1, Definition 9122.2, and Definition 9122.3 seed the finite selected-net handoff from compact and continuous rows into compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformNetSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, hidden center choice, host Lebesgue-number existence, quotient Real equality, countable choice, ambient completion, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-net exactness, finite lower-bound fold compatibility, compact-uniform handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformNetSele

Chapter 9123

A Concrete Naming Certificate for $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$

Following Bourbaki, Artin, and Bishop–Bridges on absolute values and separated completions, the $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ packet records the finite route by which an absolute-value completion boundary can be compared through constructive separated uniqueness. It sits downstream of the absolute-value branch packet of chapter 6893, the separated uniqueness packet of chapter 1463, and the regular real-name spine of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names only displayed source-field, absolute-value, branch, regular-name, separated-ledger, transport, replay, provenance, and local naming rows; it does not quotient completions, select limits, invoke choice, or identify a host completed field.

Definition 9123.1 (Ostrowski completion uniqueness carrier)**.** *An* $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, A, V, D, S, R, E, U, T, C, P, N)$$

with the following displayed rows:

1. *F is the source field row, exposing the finite arithmetic operations and equality classifier visible to the completion request.*
2. *A is the absolute-value row, displaying positivity, multiplicativity, and triangle-control data as packet history.*
3. *V is the visible Archimedean or non-Archimedean branch row inherited from the* $\mathsf{OstrowskiCompletion}^{\uparrow}$ *boundary.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read by the regular completion names.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the compared names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational source row routed through the schedule and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the displayed D, S, R spine.*
8. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *separated ledger comparing completion-facing presentations over the same F, A, V, D, S, R, E packet.*
9. *T records componentwise* hsame *transport for the source, absolute-value, branch, dyadic, stream, regular, Real, separated-ledger, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an absolute-value completion comparison through $F, A, V, D, S, R, E, U, T, P$*
11. *P is the* Pkg *provenance over* $\mathsf{OstrowskiCompletion}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion classes, selected Cauchy representatives, ambient completed fields, global branch trichotomy, countable choice, or host equality of completed points.

Theorem 9123.2 (Ostrowski completion uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ *carrier*

$$K = (F, A, V, D, S, R, E, U, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}}$ *surface consists exactly of source-field exposure, absolute-value exposure, visible branch exposure, regular-name spine exposure, separated uniqueness ledger admission, componentwise transport, displayed continuation replay, package provenance, and local naming. A consumer may read the separated ledger U only after the absolute-value branch rows F, A, V and the regular real-name rows D, S, R, E have both been displayed.*

Proof. Project the carrier of Definition 9123.1. The source-field, absolute-value, branch, dyadic, stream, regular-rational, Real-seal, separated-ledger, transport, replay, provenance, and naming rows are precisely the coordinates of the finite packet.

The branch-facing obligations are supplied by F, A, V. These rows identify the source field visible to the completion request, the absolute-value data that controls Cauchy admission, and the Archimedean or non-Archimedean branch visible to the packet.

The uniqueness-facing obligations are supplied by D, S, R, E, U. The rows D, S, R, E give the regular real-name spine, and U compares completion-facing presentations only over that displayed spine together with the same absolute-value branch data.

Finally T, C, P, N preserve, replay, source, and name the same finite packet. Since no quotient completion, selected representative, host completed field, hidden branch choice, or host equality coordinate occurs in the carrier, the stated rows exhaust the local NameCert obligation surface. □

Theorem 9123.3 (Ostrowski completion uniqueness real seal)**.** *Let*

$$K = (F, A, V, D, S, R, E, U, T, C, P, N)$$

be an accepted $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ *carrier. Every comparison exported by the separated ledger U reaches the route*

$$F, \quad A, \quad V, \quad D, \quad S, \quad R, \quad E$$

before any completion-facing uniqueness read is made. Thus the comparison is a Real-seal and separated-ledger read over displayed absolute-value completion data, not a quotient completion identity or a selected host-field limit.

Proof. Start from the carrier decomposition. The rows F, A, V expose the absolute-value completion boundary, and the rows D, S, R, E expose the regular real-name spine used by the comparison.

The separated ledger U is downstream of those displayed rows. It may compare completion-facing presentations only after the visible branch and Real-seal data have been projected from the same packet.

The replay row C can reproduce the comparison only through the listed route, while P records no host completed field or quotient-completion object. Therefore the uniqueness read is a separated Real-seal route over finite packet data. □

Falsifiable prediction: An accepted $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ replay must display the same source field row, absolute-value row, visible branch row, dyadic tolerance ledger, stream schedule, regular-rational row, Real seal, separated uniqueness ledger, transport, replay, provenance, and local naming before any completion-facing comparison is accepted. A replay that reaches a quotient completion, selected representative, hidden branch choice, host completed field, or unrecorded limit rejects this NameCert instance.

Independence witness: The packet is independent of quotient completion carriers, selected Cauchy representatives, ambient field equality, and host completed fields because the comparison route is mediated only by displayed BHist, hsame, Cont, Pkg, NameCert, $\mathsf{OstrowskiCompletion}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Closure status for $\mathsf{NameCert}_{\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}}$.

Constructive story: An $\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}$ packet is generated from source field data, absolute-value data, a visible Archimedean or non-Archimedean branch row, DyadicRatCore, StreamName, RegSeqRat, and Real spine rows, a CompletionUniqueness separated ledger, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9123.1, Definition 9123.2, and Definition 9123.3 seed the absolute-value completion uniqueness route over displayed regular real-name and separated-ledger data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OstrowskiCompletionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Ostrowski classification, quotient completions, selected Cauchy limits, host completed fields, branch trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, absolute-value branch exposure, regular-name spine exposure, separated-ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.OstrowskiCompletionUniqueness

Chapter 9124

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on adjunction units, the $\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}$ packet records the finite unit component by which a displayed dense source enters the Cauchy-completion reflector and then feeds the functorial completion surface. It sits between the functorial route of chapter 1937, the unit packet of chapter 1527, and the reflector packet of chapter 1585. The chapter names only the displayed NameCert surface of the reflection unit; it does not assert quotient completion equality, selected representatives, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9124.1 (CauchyCompletionReflectionUnit carrier)**.** *A* $\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, K, \eta, R, T, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source dense object row, carrying the displayed source histories before any completion-facing read is available.*
2. *K is the completion seal row for the Cauchy-completion reflector over X.*
3. *η is the unit map row sending the displayed dense source into the completion seal.*
4. *R is the reflection handoff row consumed by the local* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *surface.*
5. *T is the triangle identity row that compares the unit handoff with the reflector replay inside the same finite packet.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by the unit and reflection routes.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after X, K, η, R, T, S, Q have been displayed.*
9. *H records componentwise* hsame *transport for the dense source, completion seal, unit map, reflection handoff, triangle identity, window, readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$X, K, \eta \longmapsto R, T \longmapsto S, Q, E.$$

11. *P is the* Pkg *provenance over* CauchyCompletionReflectionUnit$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionReflectionUnit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected stream representatives, chosen limits, countable choice, host equality, or ambient completeness.

Theorem 9124.2 (CauchyCompletionReflectionUnit NameCert obligations)**.** *For every accepted* CauchyCompletionReflectionUnit$^{\uparrow}$ *carrier*

$$U = (X, K, \eta, R, T, S, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionReflectionUnit}^{\uparrow}}$ *surface consists exactly of source dense-object admission for X, completion-seal admission for K, unit-map exposure for η, reflection handoff through R, triangle identity exposure through T, finite StreamName window exposure through S, RegSeqRat readback through Q, Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9124.1. The displayed rows are exactly the dense source, completion seal, unit map, reflection handoff, triangle identity, finite-window, readback, Real-seal, and structural rows.

The unit and reflection obligations are read in that order. The dense source row X first enters the completion seal K through η, and only then may the reflection handoff row R be consumed by a completion-facing reader.

The analytic readback rows remain attached to the same packet. The finite window S, regular-rational readback Q, and terminal Real seal E are downstream of the unit and reflection rows, so no consumer can detach a Real endpoint or a selected representative from the displayed route.

Finally T, H, C, P, N record the triangle comparison, transport, replay, provenance, and local naming for the same carrier. Since the excluded principles are not coordinates of the packet, the listed rows exhaust the local NameCert surface. □

Theorem 9124.3 (CauchyCompletionReflectionUnit triangle identity route)**.** *Every triangle-identity read from an accepted* CauchyCompletionReflectionUnit$^{\uparrow}$ *carrier factors through the finite route*

$$X, K, \eta \longmapsto R \longmapsto T \longmapsto S, Q, E \longmapsto H, C, P, N.$$

Thus the functorial and separated-completion consumers receive the unit component only after the source dense object, completion seal, unit map, reflection handoff, triangle identity row, finite StreamName window, RegSeqRat readback, terminal Real seal, transport, replay, provenance, and local NameCert rows are visible.

Proof. Start with a consumer asking for the triangle identity. Projection gives the dense source row X, the completion seal K, and the unit map row η.

The read next passes through the reflection handoff R. This is the row visible to the CauchyCompletionReflector$^{\uparrow}$ packet and to the functorial completion route of chapter 1937.

The triangle identity is then the displayed row T, not an ambient equation of completions. The StreamName window S, RegSeqRat readback Q, and Real seal E keep the analytic route finite before any completion-facing consumer reads the unit component.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Therefore the triangle route exports no quotient completion equality, selected representative, countable-choice schedule, host equality, or ambient Real$^{\uparrow}$ completeness. □

Theorem 9124.4 (CauchyCompletionReflectionUnit finite-window nonescape)**.** *For every accepted* $\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}$ *carrier*

$$U = (X, K, \eta, R, T, S, Q, E, H, C, P, N),$$

every completion-facing consumer that reads the reflection-unit route and reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal factors through*

$$X, K, \eta, R, T, S, Q, E, H, C, P, N.$$

In particular, the finite $\mathsf{StreamName}^{\uparrow}$ *window row* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *are visible before* E*, and the route exports no quotient completion equality, selected representatives, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9124.1. The NameCert obligations of Definition 9124.2 identify the displayed rows as the source dense object, completion seal, unit map, reflection handoff, triangle identity, finite window, regular-rational readback, Real seal, transport, replay, provenance, and local naming rows.

The route theorem Definition 9124.3 supplies the prefix $X, K, \eta \longmapsto R \longmapsto T$ for every consumer that reads the reflection-unit triangle. The rows S, Q, E are downstream coordinates of the same carrier, so the finite StreamName window and RegSeqRat readback must be displayed before the terminal Real seal is reached.

Finally H, C, P, N keep the transport, replay, provenance, and local naming attached to this finite packet. Quotient completion equality, selected representatives, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not carrier coordinates, so no such principle can escape through the finite-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}$ packet is generated from a dense source object, a completion seal, a unit map row, a reflection handoff, a triangle identity row, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9124.1, Definition 9124.2, Definition 9124.3, and Definition 9124.4 seed the finite reflection-unit surface for Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dense-source admission, completion-seal admission, unit-map exposure, reflection handoff, triangle identity exposure, finite readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionReflectionUnitUp.

Chapter 9125

A Concrete Naming Certificate for $\mathsf{BishopCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion, the $\mathsf{BishopCompletionFunctor}^{\uparrow}$ packet records the functorial completion route as finite NameCert data. It sits over the metric completion carrier of chapter 1506, the operator surface of chapter 3379, the uniform-completion handoff of chapter 1473, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the complete separated landing object of chapter 2355, and the terminal real-name seal of chapter 38. The packet exports a constructive completion functor surface for downstream $\mathsf{Real}^{\uparrow}$ and completion-facing chapters, while refusing quotient completion equality, selected representatives, selected limits, countable choice, a host completion operator, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9125.1 (Bishop completion functor carrier)**.** *A* $\mathsf{BishopCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, O, U, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, naming the metric carrier and completion request.*
2. *O is the* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *row that turns displayed Cauchy data into a completion-facing handoff.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row carrying the uniform completion request attached to the same metric source.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by the Cauchy names.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the window and readback rows.*
7. *Q is the* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *landing row for the separated completion target.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row used by Real-facing consumers.*
9. *H records componentwise* hsame *transport for metric completion, operator, uniform completion, stream, regular-rational, dyadic, complete-separated, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a completion request through*

$$M, O, U, S, R, D, Q, E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionOperator}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompleteSeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, O, U, S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, selected limits, countable-choice schedules, a host completion operator, or ambient Real completeness.

Theorem 9125.2 (Bishop completion functor NameCert obligations)**.** *For every accepted* BishopCompletionFunctor *carrier*

$$B = (M, O, U, S, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionFunctor}^{\uparrow}}$ *surface consists exactly of metric-completion source exposure through M, Cauchy-completion operator exposure through O, uniform-completion handoff through U, StreamName window exposure through S, RegSeqRat readback through R, DyadicRatCore tolerance exposure through D, complete-separated landing exposure through Q, terminal Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9125.1. The rows M, O, U give the completion-facing source, operator, and uniform handoff, while S, R, D give the finite L10 readback route.

The landing rows Q, E are downstream of that finite route. They expose the complete separated metric object and the terminal Real seal only after the StreamName, RegSeqRat, and DyadicRatCore rows have been displayed.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. Since no quotient completion equality, selected representative, selected limit, countable-choice schedule, host completion operator, or ambient Real-completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligation surface. □

Theorem 9125.3 (Bishop completion functor L10 handoff)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCompletionFunctor}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CauchyCompletionOperator}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCo}$$

The handoff supplies a constructive completion functor row over displayed Cauchy and regular-name data. It supplies no quotient completion equality, selected representative, selected limit, countable choice, host completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 9125.2. It places every completion-facing read behind M, O, U and every L10 readback behind S, R, D.

The separated landing row Q and the Real seal row E are reached only after those displayed rows. Therefore a consumer cannot replace the route by a quotient completion class, a selected representative, or a selected limit.

The replay and provenance rows keep the route inside the finite packet. Thus downstream Real and completion-facing chapters consume a displayed Bishop completion functor surface, not a host completion operator or an ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompletionFunctor}^{\uparrow}$ read must display MetricCompletion, CauchyCompletionOperator, UniformCompletion, StreamName, RegSeqRat, DyadicRatCore, CompleteSeparatedMetric, Real, hsame, Cont, Pkg, and NameCert rows before any completion-facing or Real-facing output is available.

Independence witness: metriccompletion, cauchycompletionoperator, uniformcompletion, streamname, regseqrat, dyadicratcore, complete-separated-metric, real: $\mathsf{BishopCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds the completion operator, uniform handoff, L10 finite-window route, separated landing row, Real seal, transport, replay, provenance, and local naming as one functorial completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionFunctor}^{\uparrow}$ packet is generated from MetricCompletion, CauchyCompletionOperator, UniformCompletion, StreamName, RegSeqRat, DyadicRatCore, CompleteSeparatedMetric, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9125.1, Definition 9125.2, and Definition 9125.3 seed the constructive completion functor route over the L10 Cauchy, StreamName, RegSeqRat, DyadicRatCore, CompleteSeparatedMetric, and Real spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion operator exposure, uniform handoff, finite-window readback, separated landing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionFu

Chapter 9126

A Concrete Naming Certificate for $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ packet realizes the finite partition-refinement route described in chapter 9237. It sits over the compact source of chapter 60, the metric carrier of chapter 57, the continuous-map pointwise rows of chapter 130, the rational tolerance rows of chapter 37, the $\mathsf{ProbeBundle}^{\uparrow}$ finite partition spine, and the uniform-modulus handoff of chapter 369. The packet names a finite ladder from compact nets and pointwise moduli to uniform-modulus selection; it does not import open-cover compactness, a selected Lebesgue number, countable choice, quotient point equality, or an external modulus oracle.

Definition 9126.1 (Compact modulus partition ladder carrier)**.** *A* $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, F, R, B, A, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row exposing the compact finite-net or finite-cover data.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used by the partition comparisons.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the compact source.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *tolerance ledger for positive local radii, lower-bound comparisons, and refinement thresholds.*
5. *B is the* $\mathsf{ProbeBundle}^{\uparrow}$ *partition spine listing the finite probes and membership rows.*
6. *A is the ladder row, recording refinement steps from coarse probes to the subpartition carrying the local modulus selected at each probe.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced after the finite ladder has been folded.*
8. *H records componentwise* hsame *transport for compact, metric, continuous-map, rational, probe-bundle, ladder, uniform-modulus, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through*

$$K, M, F, R, B, A, U, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusPartitionLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, F, R, B, A, U, H, C, P, N$. It has no coordinate for open-cover compactness, a selected Lebesgue number, an infinite refinement tree, countable choice, quotient point equality, or an external uniform-modulus selector.

Theorem 9126.2 (Compact modulus partition ladder NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ *carrier*

$$L = (K, M, F, R, B, A, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusPartitionLadder}^{\uparrow}}$ *surface consists exactly of compact source exposure through K, metric distance exposure through M, continuous-map pointwise-modulus exposure through F, rational tolerance exposure through R, finite* $\mathsf{ProbeBundle}^{\uparrow}$ *partition exposure through B, partition-refinement ladder exposure through A, uniform-modulus handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9126.1. The compact, metric, continuous-map, rational, probe-bundle, ladder, uniform-modulus, transport, replay, provenance, and naming rows are exactly the displayed coordinates.

The compact and metric rows provide the finite partition context. The continuous-map and rational rows attach local moduli and positive tolerance comparisons to each finite probe.

The $\mathsf{ProbeBundle}^{\uparrow}$ row and ladder row record the finite refinement route before the uniform-modulus handoff is available. Transport, replay, provenance, and local naming preserve that same finite route.

No carrier coordinate names open-cover compactness, a selected Lebesgue number, an infinite refinement tree, countable choice, quotient point equality, or an external selector, so none of those objects is part of the local NameCert obligation surface. □

Theorem 9126.3 (Compact modulus partition ladder handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ *packet must factor its uniform-modulus request through the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{ProbeBundle}^{\uparrow} \longmapsto A \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto H, C, P, N.$$

The ladder supplies finite partition-refinement data from compact nets and pointwise moduli to a uniform-modulus handoff. It supplies no open-cover compactness theorem, selected Lebesgue number, infinite refinement tree, countable choice, quotient point equality, or external modulus selector.

Proof. Apply Definition 9126.2. It places compact, metric, continuous-map, rational, and finite ProbeBundle rows before the ladder row A.

The ladder row records the finite refinement steps that feed the uniform-modulus handoff U. Because U is downstream of A, the consumer receives a tolerance-only result only after the finite partition route has been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the excluded host objects are absent from the carrier, they cannot be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ read displays Compact, Metric, ContinuousMap, Rat, ProbeBundle, partition ladder, UniformModulus, hsame, Cont, Pkg, and NameCert rows before any uniform-modulus handoff is accepted.

Independence witness: compact, metric, continuousmap, rat, probebundle, uniformmodulus: $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ is not bijective to any listed sibling because it binds finite compact partitions, metric distance comparisons, pointwise continuous-map moduli, rational tolerance ledgers, ProbeBundle membership, partition refinement, uniform-modulus handoff, transport, replay, provenance, and local naming as one ladder packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusPartitionLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusPartitionLadder}^{\uparrow}$ packet is generated from Compact, Metric, ContinuousMap, Rat, ProbeBundle, partition-refinement ladder, UniformModulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusPartitionLadder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9126.1, Definition 9126.2, and Definition 9126.3 seed the finite partition ladder from compact nets and pointwise moduli to uniform-modulus handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusPartitionLadder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, infinite refinement trees, countable choice, quotient point equality, external modulus selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, metric-distance exposure, pointwise ContinuousMap exposure, rational tolerance exactness, ProbeBundle partition coverage, refinement ladder stability, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactModulusPartitionLadderUp.

Chapter 9127

A Concrete Naming Certificate for $\mathsf{CauchyKovalevskaya}^{\uparrow}$

Following Cauchy and Kovalevskaya on analytic initial-value problems, the $\mathsf{CauchyKovalevskaya}^{\uparrow}$ packet records the finite analytic-jet surface by which a displayed PDE coefficient ledger, initial hypersurface jet, derivative ledger, stream schedule, rational coefficient readback, dyadic coefficient core, and real seal are named before any local solution claim is public. It sits next to the PDE carrier of chapter 194, the derivative rows of chapter 122, the continuous-map boundary of chapter 130, the stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic coefficient core of chapter 291, and the real-name seal of chapter 38. The packet names finite analytic data for a constructive CK read; it is not analytic continuation, a selected solution germ, quotient equality of functions, countable choice over coefficients, or ambient real-completeness closure.

Definition 9127.1 (Cauchy–Kovalevskaya carrier)**.** *A* $\mathsf{CauchyKovalevskaya}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, D, C, I, J, S, Q, Y, R, H, T, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Pde}^{\uparrow}$ *row naming the inspected analytic PDE operator and the distinguished evolution variable.*
2. *D is the* $\mathsf{Derivative}^{\uparrow}$ *ledger for the partial derivative rows requested by the operator.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *coefficient-boundary row for the displayed analytic coefficient maps.*
4. *I is the initial hypersurface row, including the finite analytic initial data read by the packet.*
5. *J is the finite jet recursion row, recording the coefficient-by-coefficient recursion available from P, D, C, I.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row scheduling the inspected coefficient windows and finite jet depths.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational coefficient bounds and convergence radii visible at those windows.*
8. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row storing the dyadic coefficient ledger used by the rational readback.*
9. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the displayed real coefficient and radius reads.*
10. *H records componentwise* hsame *transport for PDE, derivative, continuous-map, initial-data, jet, stream, rational, dyadic, real, replay, provenance, and local-name rows.*

11. *T records displayed* Cont *replay from a finite analytic-jet request through $P, D, C, I, J, S, Q, Y, R$.*
12. *G is the* Pkg *provenance over* $\mathsf{CauchyKovalevskaya}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyKovalevskaya}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, D, C, I, J, S, Q, Y, R, H, T, G, N$. It has no coordinate for a host analytic solution, analytic continuation past the displayed radius, selected germ, quotient equality of functions, countable coefficient choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyKovalevskayaUp.CauchyKovalevskayaCarrier`

Theorem 9127.2 (Cauchy–Kovalevskaya NameCert obligations)**.** *For every accepted* $\mathsf{CauchyKovalevskaya}^{\uparrow}$ *carrier $K = (P, D, C, I, J, S, Q, Y, R, H, T, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyKovalevskaya}^{\uparrow}}$ *surface consists exactly of carrier admission for the PDE row, derivative ledger, continuous coefficient boundary, initial hypersurface data, finite jet recursion, stream schedule, regular rational readback, dyadic coefficient ledger, real seal, componentwise transport, displayed replay, provenance, and local naming. No obligation may replace those rows by an unlogged analytic solution, analytic continuation principle, selected germ, quotient equality of functions, countable-choice coefficient stream, or ambient real-completeness theorem.*

Proof. Project the carrier of Definition 9127.1. The analytic rows are $P, D, C, I, J, S, Q, Y, R$, and the structural rows are H, T, G, N.

The PDE row, derivative ledger, coefficient boundary, and initial hypersurface data are read before the jet recursion. Thus the row J records only the finite coefficient recursion determined by displayed inputs.

The coefficient windows are finite. The stream row S fixes the inspected depths, the regular rational row Q records the rational bounds, the dyadic row Y stores the coefficient ledger, and the real row R seals only those displayed reads.

Finally H, T, G, N transport, replay, source, and name the same packet. Since the carrier has no solution-germ, analytic-continuation, quotient-function, choice-stream, or ambient-completeness coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 9127.3 (Cauchy–Kovalevskaya analytic-jet scope)**.** *Every constructive CK consumer of an accepted* $\mathsf{CauchyKovalevskaya}^{\uparrow}$ *carrier must read the finite route*

$\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg,

The route exposes the displayed PDE operator, partial-derivative ledger, analytic coefficient boundary, initial hypersurface data, finite jet recursion, stream window, rational coefficient readback, dyadic coefficient ledger, real seal, transport, replay, provenance, and local naming before any analytic-jet read is public. It exports no global solution theorem, uniqueness theorem, analytic continuation theorem, quotient equality of functions, or selected solution germ.

Proof. Start with a CK read request. It first opens the PDE row P, the derivative ledger D, the continuous coefficient row C, and the initial data row I.

Only then can the consumer inspect J. That row is a finite jet recursion over the displayed rows, so it cannot supply a host solution or a global existence theorem.

Next read the stream, rational, dyadic, and real rows S, Q, Y, R. These rows bind coefficient depth, rational bounds, dyadic ledgers, and real seals to the same finite analytic window.

The structural rows H, T, G, N preserve that order by componentwise transport, displayed replay, provenance, and local naming. Hence the public scope is the analytic-jet NameCert surface and not analytic continuation or solution selection. □

Theorem 9127.4 (Cauchy–Kovalevskaya finite coefficient ledger)**.** *The coefficient ledger of an accepted* $\mathsf{CauchyKovalevskaya}^{\uparrow}$ *packet is finite at every public read: each requested jet depth is scheduled by S, bounded through Q, stored through Y, and sealed through R before the local* $\mathsf{NameCert}$ *row is consumed. A downstream PDE consumer cannot bypass this ledger by appealing to an infinite formal power series, a selected analytic germ, quotient equality of coefficient streams, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the stream schedule S. It names the finite coefficient windows and jet depths available to the public request.

Use Q and Y next. The regular rational row records the visible coefficient bounds, while the dyadic core row stores the displayed coefficient ledger for the same windows.

The real seal R is downstream of those finite rows. It seals the displayed coefficient and radius reads only after the rational and dyadic ledgers are present.

Therefore the local naming row N can certify only the finite coefficient ledger transported and replayed by H, T, G. It cannot certify an unbounded series, a selected germ, quotient equality, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{CauchyKovalevskaya}^{\uparrow}$ packet that exports a CK analytic-jet read while omitting the PDE row, derivative ledger, coefficient boundary, finite jet recursion, stream schedule, rational readback, dyadic ledger, real seal, transport, replay, provenance, or local NameCert row refutes the carrier.

Independence witness: pde, derivative, continuousmap, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyKovalevskaya}^{\uparrow}$ is not bijective to any listed sibling because it binds PDE operator data, derivative rows, analytic coefficient boundaries, initial hypersurface data, finite jet recursion, scheduled coefficient windows, rational and dyadic coefficient ledgers, real seals, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyKovalevskaya}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyKovalevskaya}^{\uparrow}$ packet is generated from a PDE row, derivative ledger, continuous coefficient boundary, initial hypersurface row, finite jet recursion, StreamName schedule, RegSeqRat readback, DyadicRatCore coefficient ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyKovalevskaya}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9127.1, Definition 9127.2, Definition 9127.3, and Definition 9127.4 seed the finite analytic-jet NameCert surface for constructive PDE consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyKovalevskaya}^{\uparrow}}, \mathsf{encodedDef})$

Bridge status: none

Not claimed: This chapter does not close analytic continuation, global existence, uniqueness of analytic solutions, quotient equality of functions, selected solution germs, countable choice over coefficients, RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, PDE-row exposure, derivative-ledger exactness, coefficient-boundary visibility, jet-recursion exactness, finite coefficient ledger, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyKovalevskayaUp.

Chapter 9128

A Concrete Naming Certificate for $\mathsf{RegularCauchyBoundedness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ packet records the finite boundedness information that a regular Cauchy rational name must expose before a $\mathsf{Real}^{\uparrow}$ consumer may read the terminal seal. The packet sits over the stream window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the modulus threshold surface of chapter 292, the real-name seal of chapter 38, and the map-facing consumer of chapter 1401. It names boundedness as displayed finite induction data, not as a supremum, selected limit, quotient stream, or ambient completeness theorem.

9128.1 Regular Cauchy boundedness carrier

Definition 9128.1 (Regular Cauchy boundedness carrier)**.** *A* $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, D, M, I, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation window naming the inspected rational values and the finite prefix whose maximum is used by the bound;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, exposing the regular Cauchy inequality used after the prefix window;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the dyadic budgets and rational comparisons appearing in the prefix and tail estimates;*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row telling which depth places every later value within the displayed dyadic tolerance of the chosen prefix value;*
5. *I is the finite induction budget row, recording the prefix-length induction and the displayed bound assembled from the prefix maximum and the tail tolerance ledger;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached only after S, R, D, M, I have been read;*
7. *H stores componentwise* hsame *transport for the stream window, regular readback, dyadic ledger, modulus threshold, induction budget, seal, replay, provenance, and local naming rows;*

8. *C stores displayed* Cont *replay from a boundedness request through the stream window, regular readback, dyadic ledger, modulus threshold, induction budget, and real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBoundedness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedness}^{\uparrow}}$ *row.*

The carrier exports no host supremum, no quotient stream equality, no selected Cauchy limit, no global real completeness principle, and no hidden bound beyond the displayed finite induction budget.

Theorem 9128.2 (Regular Cauchy boundedness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ *carrier* $B = (S, R, D, M, I, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedness}^{\uparrow}}$ *surface consists exactly of carrier admission for these ten rows, stream-window exposure through S, regular readback through R, dyadic tolerance exactness through D, threshold exposure through M, finite induction-budget exactness through I, real-seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9128.1. The rows S, R, D, M, I are the whole boundedness surface: the prefix is visible in S, the tail comparison is visible in R, dyadic arithmetic is visible in D, the threshold is visible in M, and the finite induction budget is visible in I.

The remaining rows do not add analytic content. The seal E receives the displayed route, while H, C, P, N transport, replay, source, and name the same packet.

Since no coordinate stores a supremum, quotient stream equality, selected limit, or ambient completeness theorem, the stated rows exhaust the local NameCert obligations. □

Theorem 9128.3 (Regular Cauchy boundedness finite induction bound)**.** *Every accepted* RegularCauchyBoundedne *carrier* $B = (S, R, D, M, I, E, H, C, P, N)$ *exposes a boundedness witness by the finite route*

$$S, R, D, M \longmapsto I.$$

The induction budget I first compares the finitely many values in the $\mathsf{StreamName}^{\uparrow}$ *prefix named by S, then uses the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold M and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D to cover every later displayed value. Thus the bound is a finite induction certificate, not a selected supremum, quotient-stream representative, host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Read the prefix window in S. The finitely many inspected values admit an induction over the displayed prefix length, and the budget row I records the resulting prefix bound.

Next read the tail route. The modulus threshold M tells where the regular Cauchy comparison begins, R supplies the regular rational readback after that depth, and D supplies the dyadic tolerance arithmetic used to compare later values with the prefix witness.

Combine the prefix induction with the tail comparison in I. Because each step names only rows already present in B, the exported bound is the displayed finite induction certificate and not an off-packet supremum or completeness argument. □

Theorem 9128.4 (Regular Cauchy boundedness Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ *packet must read the route*

$$S, R, D, M, I, E, H, C, P, N$$

before using the bounded regular Cauchy name. The handoff exposes the stream window, regular readback, dyadic tolerance ledger, modulus threshold, finite induction budget, real seal, transport, replay, provenance, and local naming rows, and it refuses host functions, quotient streams, selected limits, detached suprema, and ambient real completeness.

Proof. Begin with the NameCert obligations of Definition 9128.2. They require the stream, regular readback, dyadic, threshold, and induction rows to be present before the seal row can be consumed.

Use the finite induction bound of Definition 9128.3. It identifies the boundedness witness as the displayed S, R, D, M, I route, so a Real-facing or Cauchy-continuous-map-facing consumer has no separate bound to inspect.

Finally pass through E, H, C, P, N. These rows seal, transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host functions, quotient streams, selected limits, detached suprema, or completeness, the consumer receives only the displayed boundedness certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedness}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ packet is generated from a StreamName window, RegSeqRat regular readback, DyadicRatCore tolerance ledger, CauchyModulus threshold row, finite induction budget, RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedness}\uparrow}, \mathsf{obligation})$

Scope closed: Definition 9128.1, Definition 9128.2, Definition 9128.3, and Definition 9128.4 bind the finite boundedness route from regular Cauchy rational names to the RealUp and CauchyContinuousMap consumer surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBoundedness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host suprema, quotient stream equality, selected Cauchy limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, classifier stability, consumer coverage for the boundedness handoff, and Lean verification for BEDC.Derived.RegularCauchyBoundednessUp.

Chapter 9129

A Concrete Naming Certificate for $\mathsf{CompletionReflector}^{\uparrow}$

Following Bourbaki and Mac Lane, the $\mathsf{CompletionReflector}^{\uparrow}$ packet records the reflector surface by which a displayed source carrier is routed to a completed carrier through unit, extension, universal comparison, separatedness, transport, replay, provenance, and NameCert rows. It abstracts the metric route of chapter 2149, the regular Cauchy route of chapter 3398, and the uniform route of chapter 6282 into one L10 boundary. The packet names only finite BEDC data; it does not import quotient completion equality, a selected limit, ambient completeness, or an external reflector theorem.

Definition 9129.1 (Completion-reflector carrier)**.** *A* $\mathsf{CompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, U, E, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-carrier row, recording the object whose completion request is visible to downstream consumers.*
2. *T is the completed-carrier row, recording the displayed completion target without identifying it with any host quotient.*
3. *U is the unit row from the source carrier into the completed carrier.*
4. *E is the extension row, recording how an accepted compatible map is replayed across the completed target.*
5. *Q is the universal-comparison row, exposing the reflector comparison as finite request-and-response data.*
6. *A is the separatedness row, recording the boundary at which the completed target may be read by separated or Real-facing consumers.*
7. *H records componentwise* hsame *transport for source, completed target, unit, extension, universal comparison, separatedness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-reflector request through*

$$S, T, U, E, Q, A, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionReflector}^{\uparrow}$*, the cited completion-reflector siblings,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, U, E, Q, A, H, C, P, N$. It has no coordinate for quotient equality of completions, selected Cauchy representatives, chosen limits, ambient completeness, or host-level reflector equality.

Theorem 9129.2 (Completion-reflector NameCert obligations)**.** *For every accepted* $\mathsf{CompletionReflector}^{\uparrow}$ *carrier*

$$R = (S, T, U, E, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionReflector}^{\uparrow}}$ *surface consists exactly of source-carrier exposure through S, completed-carrier exposure through T, unit exposure through U, extension replay through E, universal comparison through Q, separatedness through A, componentwise transport through H, displayed continuation through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9129.1. The displayed tuple lists the source, completed target, unit, extension, universal comparison, separatedness, transport, replay, provenance, and naming rows, so every public read of the packet first exposes those finite coordinates.

The unit row U and extension row E give the constructive arrow data. The comparison row Q is downstream of those rows, and the separatedness row A records the boundary required by separated or Real-facing reflector consumers.

The structural rows H, C, P, N preserve the same route under hsame, replay it through Cont, source it through Pkg, and name it by NameCert. Since no quotient equality, selected representative, selected limit, ambient completeness theorem, or host reflector equality appears as a coordinate, no such object belongs to the local obligation surface. □

Theorem 9129.3 (Completion-reflector L10 route)**.** *Every metric, regular Cauchy, uniform, separated, or Real-facing consumer of an accepted* $\mathsf{CompletionReflector}^{\uparrow}$ *packet must factor its reflector read through the finite route*

$$S \longmapsto U \longmapsto T \longmapsto E \longmapsto Q \longmapsto A \longmapsto H, C, P, N.$$

The route supplies the common L10 reflector surface consumed by completion-reflector specializations. It supplies no quotient completion, selected limit, ambient completeness theorem, or host equality between completed objects.

Proof. Apply Definition 9129.2. It places the source row S, unit row U, completed target row T, extension row E, universal-comparison row Q, and separatedness row A before any structural read is available.

The sibling reflector chapters instantiate this same order with metric, regular Cauchy, or uniform source data. Their consumers may refine the rows, but they still read the common unit, extension, comparison, separatedness, transport, replay, provenance, and local NameCert boundary named here.

Finally, the absence clauses in the carrier prevent a consumer from extracting quotient completion equality, selected representatives, chosen limits, ambient completeness, or host reflector equality from the packet. □

Falsifiable prediction: Every accepted $\mathsf{CompletionReflector}^{\uparrow}$ read displays source, completed-target, unit, extension, universal-comparison, separatedness, hsame, Cont, Pkg, and NameCert rows before any completion-facing output is accepted.

Independence witness: metric-completion-reflector, regular-cauchy-completion-reflector, uniform-completion-reflector: $\mathsf{CompletionReflector}^{\uparrow}$ is not bijective to any listed sibling because it binds the common source-to-completion reflector route across unit, extension, comparison, separatedness, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionReflector}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CompletionReflector}^{\uparrow}$ packet is generated from source carrier, completed target, unit, extension, universal comparison, separatedness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9129.1, Definition 9129.2, and Definition 9129.3 seed the common L10 reflector route consumed by metric, regular Cauchy, uniform, separated, and Real-facing completion chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, ambient completeness, host reflector equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and completed-target exposure, unit exactness, extension replay, universal comparison, separatedness, transport, replay, provenance, and local naming for BEDC.Derived.CompletionReflectorUp.

Chapter 9130

A Concrete Naming Certificate for $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$

Following Brouwer and Bishop on bars and compact uniform continuity, the $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ packet realizes the finite prefix branch of chapter 9237. It sits after the finite prefix bar carrier of chapter 7071, the compact source of chapter 60, and the tolerance-only uniform-modulus handoff of chapter 369. The packet records a finite route from prefix-depth evidence to a modulus window; it does not choose an infinite branch, import open-cover compactness, or assert an ambient Heine–Cantor theorem.

Definition 9130.1 (Finite prefix bar-modulus window carrier)**.** *A* $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, K, T, D, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite prefix bar row, recording the accepted bar prefixes and their prefix-order boundary.*
2. *K is the compact-source row, exposing the compact data consumed by the finite prefix argument.*
3. *T is the tolerance-request row, recording the requested output precision as visible finite data.*
4. *D is the prefix-depth modulus row, assigning the finite depth at which every accepted prefix supplies the requested tolerance.*
5. *U is the uniform-modulus handoff row consumed by* $\mathsf{UniformModulus}^{\uparrow}$ *readers.*
6. *H records componentwise* hsame *transport for bar, compact-source, tolerance, depth-modulus, uniform-modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through*

$$B, K, T, D, U, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$, $\mathsf{FinitePrefixBar}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, T, D, U, H, C, P, N$. It has no coordinate for an infinite branch selector, open-cover compactness, countable choice, quotient stream equality, or an external uniform-modulus oracle.

Theorem 9130.2 (Finite prefix bar-modulus window NameCert obligations)**.** *For every accepted* $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ *carrier*

$$W = (B, K, T, D, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}}$ *surface consists exactly of finite prefix bar exposure through* B*, compact-source exposure through* K*, tolerance request exposure through* T*, prefix-depth modulus exposure through* D*, uniform-modulus handoff through* U*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 9130.1. The rows $B, K, T, D, U, H, C, P, N$ are exactly the accepted packet coordinates.

The finite prefix bar row B and compact-source row K provide the source data. The tolerance request T is then routed to the prefix-depth modulus row D, which is the only place where a finite depth is exposed.

The uniform-modulus handoff U is downstream of D. Transport, replay, provenance, and local naming preserve the same finite route, and the absent infinite-branch, open-cover, countable-choice, quotient-stream, and external-oracle objects cannot be exported by the local certificate. □

Theorem 9130.3 (Finite prefix bar-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ *packet must factor its tolerance-only request through the finite route*

$$\mathsf{FinitePrefixBar}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ T \longmapsto D \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the handoff from finite prefix depth to uniform-modulus output is a displayed BEDC route, not an informal compactness-to-modulus jump.

Proof. Apply Definition 9130.2. It places the finite prefix bar, compact-source, and tolerance rows before the depth-modulus row D.

The row D supplies the finite prefix depth consumed by the uniform-modulus handoff U. A downstream consumer therefore receives a tolerance-only handoff only after the prefix-depth row has been displayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no infinite branch selector, open-cover compactness theorem, quotient stream equality, countable choice, or external modulus oracle, none of those objects can mediate the handoff. □

Falsifiable prediction: Every accepted $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ read displays FinitePrefixBar, compact-source, tolerance, prefix-depth modulus, UniformModulus, hsame, Cont, Pkg, and NameCert rows before any tolerance-only uniform-modulus handoff is accepted.

Independence witness: finite-prefix-bar, compact, uniformmodulus: $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite prefix bars, compact-source rows, tolerance requests, prefix-depth modulus rows, uniform-modulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}$ packet is generated from finite prefix bar, compact-source, tolerance-request, prefix-depth modulus, UniformModulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixBarModulusWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9130.1, Definition 9130.2, and Definition 9130.3 seed the compact-uniform-continuity route from finite prefix depth to tolerance-only uniform-modulus handoff rows.

Formal verification: FormalStatus(NameCert$_{\text{FinitePrefixBarModulusWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, open-cover compactness, countable choice, quotient stream equality, external modulus oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix bar exposure, compact-source exposure, tolerance request exactness, prefix-depth modulus stability, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FinitePrefixBarModulusWindowUp.

Chapter 9131

A Concrete Naming Certificate for $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$

Following Bishop and Bridges and Kelley on completion and uniform stabilization, the $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$ packet records the finite route by which repeated completion reads are stabilized before a Real-facing consumer inspects the same regular name. It sits over the idempotence route of **??**, the functorial route of chapter 423, the finite window source of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a repeated-read stabilizer only; it does not assert quotient completion, selected limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9131.1 (CauchyCompletionStabilizer carrier)**.** *A* $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, I, F, W, D, R, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the source completion row, recording the first completion-facing read before any repeated-read stabilization is available.*
2. *I is the* $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$ *row, exposing the finite idempotence comparison between the first completion read and the repeated completion read.*
3. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row, carrying the functorial route that transports the same displayed completion request.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the repeated read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the finite window and comparison budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window and dyadic tolerance rows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after C, I, F, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for source completion, idempotence comparison, functorial route, finite window, dyadic tolerance, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay of the route*

$$C \longmapsto I, F \longmapsto W, D, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionFu}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStabilizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, I, F, W, D, R, E, H, T, P, N$. It has no coordinate for quotient completion equality, selected limits, countable choice, host equality between completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9131.2 (CauchyCompletionStabilizer NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionStab}$ *carrier*

$$S = (C, I, F, W, D, R, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionStabilizer}^{\uparrow}}$ *surface consists exactly of source completion exposure through C, repeated-read stabilization through I, functorial completion transport through F, finite StreamName window exposure through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real sealing through E, componentwise transport through H, displayed replay through T, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9131.1. The displayed rows are the source completion, idempotence, functorial route, finite window, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and local naming rows.

The completion rows are read first. Row C supplies the source completion read, row I compares it with the repeated read, and row F keeps the comparison inside the functorial completion route.

The analytic rows remain downstream of that route. The finite window W, dyadic tolerance D, regular readback R, and terminal seal E are part of the same packet, so a consumer cannot detach a selected limit or a quotient completion object from the repeated-read comparison.

Finally H, T, P, N preserve, replay, source, and name the same carrier. Since quotient completion, selected limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the packet, the listed rows exhaust the local NameCert surface. □

Theorem 9131.3 (CauchyCompletionStabilizer seal route)**.** *Every Real-facing consumer of an accepted* $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$ *packet must factor its repeated completion read through the finite route*

$$\mathsf{CauchyCompletionIdempotence}^{\uparrow},\ \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

The route stabilizes the displayed repeated completion read before the terminal Real seal is consumed. It supplies no quotient completion, selected limit, countable-choice schedule, host equality of completions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 9131.2. It places the source completion row C before the idempotence and functorial rows I, F.

Next read the finite observation route. The StreamName window W, DyadicRatCore tolerance D, and RegSeqRat readback R are displayed before the Real seal E, so the consumer receives a finite repeated-read packet rather than a chosen completed object.

Finally use the structural rows H, T, P, N. They transport, replay, source, and name the same route, which prevents a Real-facing read from bypassing the stabilizer through quotient equality, a selected limit, countable choice, host completion equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionStabilizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionStabilizer}^{\uparrow}$ packet is generated from a source completion row, CauchyCompletionIdempotence comparison, CauchyCompletionFunctor route, StreamName finite windows, DyadicRatCore tolerance, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStabilizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9131.1, Definition 9131.2, and Definition 9131.3 seed the repeated completion-read stabilizer route over the CauchyCompletionIdempotence, CauchyCompletionFunctor, StreamName, DyadicRatCore, RegSeqRat, and Real spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStabilizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source completion admission, repeated-read stabilization, functorial transport, finite-window readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionStabil

Chapter 9132

A Concrete Naming Certificate for $\mathsf{DarbouxGapRefinement}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxGapRefinement}^{\uparrow}$ packet records the finite partition refinement surface that turns an upper-minus-lower Darboux gap into a displayed refinement certificate before a Real-facing consumer reads a regular rational window. It sits after the Darboux-sum refinement route of chapter 8838, the tagged or dyadic partition surfaces of chapter 7449 and chapter 1863, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Darboux and Riemann-facing consumers a finite gap-refinement certificate without importing selected partition sequences, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9132.1 (DarbouxGapRefinement carrier)**.** *A* $\mathsf{DarbouxGapRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, L, U, T, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{DarbouxSumRefinement}^{\uparrow}$ *source row, carrying the source partition, refined partition, finite cell-incidence map, and Darboux handoff.*
2. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row on the displayed refined partition.*
3. *U is the* ***??*** *row on the same displayed refined partition.*
4. *T is either the* $\mathsf{TaggedPartition}^{\uparrow}$ *row or the* $\mathsf{DyadicPartition}^{\uparrow}$ *row that supplies the finite refinement cells read by R, L, U.*
5. *D is the finite upper-minus-lower gap row comparing U and L over exactly the displayed cells of T.*
6. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed gap budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, L, U, T, D, W have been displayed.*
8. *H records componentwise* hsame *transport for the refinement, lower row, upper row, partition row, gap row, readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from Darboux, Riemann, Real, or Completion requests through the same finite packet.*

10. *P is the* Pkg *provenance over* $\mathsf{DarbouxGapRefinement}^{\uparrow}$, $\mathsf{DarbouxSumRefinement}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{TaggedPartition}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxGapRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, L, U, T, D, W, E, H, C, P, N$. It has no coordinate for a selected sequence of partitions, quotient equality of functions, a host supremum or infimum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9132.2 (DarbouxGapRefinement NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxGapRefinement}^{\uparrow}$ *carrier $G = (R, L, U, T, D, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxGapRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the Darboux-sum refinement row, lower and upper Darboux-sum rows, tagged or dyadic partition row, finite gap row, regular rational readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is read through the same finite partition-gap packet.*

Proof. Carrier admission is projection from Definition 9132.1. The admitted packet displays the Darboux-sum refinement source, the lower and upper rows, the tagged or dyadic partition row, the gap comparison, the regular rational readback, and the Real seal.

The partition-gap obligations are finite. The row T supplies the displayed cells, R supplies the refinement route, and D compares the lower and upper Darboux rows only over those cells.

The readback and seal rows are downstream. The row W turns the displayed gap into a regular rational window, and E is reached only after R, L, U, T, D, W have been opened. The structural rows H, C, P, N transport, replay, source, and name exactly the same packet, so the listed rows exhaust the local NameCert surface. □

Theorem 9132.3 (DarbouxGapRefinement partition route)**.** *Every Darboux, Riemann, Real, or Completion consumer of an accepted* $\mathsf{DarbouxGapRefinement}^{\uparrow}$ *carrier reaches the gap certificate only through the finite route*

$$\mathsf{DarbouxSumRefinement}^{\uparrow} \longmapsto (\mathsf{TaggedPartition}^{\uparrow} \text{ or } \mathsf{DyadicPartition}^{\uparrow}) \longmapsto (\mathsf{LowerDarbouxSum}^{\uparrow}, ??) \longmapsto \mathsf{RegSeqRat}^{\uparrow} \mapsto$$

The consumer first reads the refinement row R, the displayed partition row T, the lower and upper rows L, U, and the gap row D; only then may it consume the regular rational readback W and terminal Real seal E. It receives no selected partition sequence, quotient-function equality, host integral value, countable-choice schedule, or ambient completeness theorem.

Proof. Start with an accepted carrier $G = (R, L, U, T, D, W, E, H, C, P, N)$. The carrier definition makes R the Darboux-sum refinement source and T the only displayed partition row available to the consumer.

Read L and U over the same cells supplied by T. The gap row D then compares the upper and lower rows over exactly that displayed finite partition route.

Only after this comparison is displayed can the consumer reach W and E. The regular rational readback and Real seal are therefore downstream of the finite partition-gap route, while H, C, P, N merely transport, replay, source, and name the same packet. Since the carrier has no coordinate for a selected partition sequence, quotient-function equality, host integral value, countable-choice schedule, or ambient completeness theorem, none of those objects is exported. □

Falsifiable prediction: Every accepted $\mathsf{DarbouxGapRefinement}^{\uparrow}$ read used by a Darboux, Riemann, Real, or Completion consumer displays one DarbouxSumRefinementUp source row, one

tagged or dyadic partition row, lower and upper Darboux-sum rows, one finite upper-minus-lower gap row, one RegSeqRat readback row, one RealUp seal, and the structural rows H, C, P, N before it is treated as a partition-gap refinement certificate.

Independence witness: darboux-sum-refinement, tagged-partition, dyadic-partition, regseqrat, real: $\mathsf{DarbouxGapRefinement}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries the refinement source, lower and upper finite Darboux rows, the displayed partition cells, the finite gap comparison, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxGapRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxGapRefinement}^{\uparrow}$ packet is generated from a DarbouxSumRefinement source row, lower and upper Darboux-sum rows, a tagged or dyadic partition row, a finite gap row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxGapRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9132.1, Definition 9132.2, and Definition 9132.3 seed the finite partition-gap refinement route from Darboux-sum refinement to Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxGapRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected partition sequences, quotient equality of functions, host suprema or infima, host integral values, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper sum coherence, partition-route exactness, finite gap readback, Real-seal nonescape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.DarbouxGapRefinementUp.

Chapter 9133

A Concrete Naming Certificate for $\mathsf{RationalIntervalHull}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalIntervalHull}^{\uparrow}$ packet records the finite rational lower-upper hull used before enclosure and completion consumers inspect a regular rational or real-facing interval route. It sits between the finite bracket surface of chapter 1802, the dyadic tolerance arithmetic of chapter 291, the regular rational source rows of chapter 301, the stream-window boundary of chapter 126, the terminal real-name seal of chapter 38, and the located hull surface of chapter 2327. The packet names a finite rational hull family; it does not assert quotient real equality, a least ambient real interval, selected endpoints, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9133.1 (Rational interval hull carrier)**.** *A* $\mathsf{RationalIntervalHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (L, U, O, D, S, R, E, A, T, C, P, N)$$

with the following displayed rows:

1. *L is the finite family of rational lower endpoint rows, each read through* $\mathsf{Rat}^{\uparrow}$*.*
2. *U is the finite family of rational upper endpoint rows, each read through* $\mathsf{Rat}^{\uparrow}$*.*
3. *O is the rational order ledger, recording $L_i \leq U_i$ and the pairwise lower-upper comparisons requested by the hull consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare rational endpoints against dyadic radii and finite observation scales.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite requests at which the source interval family is inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational names served by the windows in S.*
7. *E is the* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ *handoff row, exposing each rational bracket as an enclosure packet.*
8. *A is the* $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ *admission row, reached only after the rational endpoints, dyadic tolerance ledger, stream windows, regular readbacks, and enclosure handoff have been displayed.*
9. *T records componentwise* hsame *transport for endpoints, order ledgers, tolerances, windows, readbacks, enclosures, located hull rows, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a hull request through $L, U, O, D, S, R, E, A, T, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalHull}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RationalIntervalEnclosure}^{\uparrow}$, $\mathsf{LocatedRealIntervalHull}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalHull}^{\uparrow}}$ *row.*

The classifier compares accepted rational hulls only through $L, U, O, D, S, R, E, A, T, C, P, N$. It has no coordinate for quotient real equality, selected representatives, a host least interval, total trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9133.2 (Rational interval hull NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalHull}^{\uparrow}$ *carrier $H = (L, U, O, D, S, R, E, A, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalHull}^{\uparrow}}$ *surface consists exactly of carrier admission, rational endpoint ordering, dyadic tolerance compatibility, stream-window exposure, regular rational readback, enclosure handoff, located-hull admission, componentwise transport, displayed replay, provenance, and local naming. Every obligation is read from the displayed rows over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RationalIntervalEnclosure}^{\uparrow}$, $\mathsf{LocatedRealIntervalHull}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Proof. Project the accepted carrier. The rational endpoint rows L, U and the order ledger O are the only endpoint and comparison coordinates available to the local classifier.

The dyadic and regular-source stages are then read through D, S, R. The dyadic ledger fixes finite tolerance comparisons, the stream row fixes the inspected requests, and the regular rational row records the readback served at those requests.

The enclosure and located-hull stages are downstream. The row E repackages the finite rational brackets as $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ data, while A admits the same displayed brackets to the $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ surface without adding a host least interval.

Finally T, C, P, N transport, replay, package, and name the same finite route. Since no excluded quotient, selected endpoint, total comparison, choice, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 9133.3 (Rational interval hull finite-family exactness)**.** *Every accepted* $\mathsf{RationalIntervalHull}^{\uparrow}$ *packet exposes its hull as exactly the finite family route*

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$
$$\longmapsto \mathsf{RationalIntervalEnclosure}^{\uparrow} \longmapsto \mathsf{LocatedRealIntervalHull}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

A Real or Completion consumer receives the displayed rational endpoint family, rational order ledger, dyadic tolerance rows, stream windows, regular readbacks, enclosure handoff, located-hull admission, transport, replay, provenance, and local naming. It receives no quotient real equality, host least interval, selected endpoint pair, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or hidden infinite family.*

Proof. Start with the NameCert obligation theorem. It identifies $L, U, O, D, S, R, E, A, T, C, P, N$ as the whole local surface, so a consumer has no access to any row outside the finite packet.

Read the family route in order. The rational endpoint rows and order ledger precede the dyadic tolerance row; the stream windows and regular rational readback precede the enclosure handoff; and the located-hull row precedes the terminal Real-facing seal.

Transport and replay preserve the same route. Componentwise hsame transport and displayed Cont replay can only move the named rows T and C, while P and N record provenance and local naming for the same finite family.

Thus finite-family exactness is a carrier inventory claim: all public hull reads factor through the displayed rows, and the excluded quotient, host, selected, completeness, and hidden-family principles are absent. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalHull}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalHull}^{\uparrow}$ packet is generated from finite rational lower and upper endpoint rows, a rational order ledger, a DyadicRatCoreUp tolerance ledger, StreamNameUp windows, RegSeqRatUp readbacks, RationalIntervalEnclosureUp handoff rows, LocatedRealIntervalHullUp admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalHull}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9133.1, Definition 9133.2, and Definition 9133.3 seed the finite rational hull route from rational endpoint families to enclosure, located-hull, Real, and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalHull}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, a host least interval, selected endpoints, total trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational endpoint ordering, dyadic tolerance compatibility, stream-window exposure, regular readback, enclosure handoff, located-hull admission, transport, replay, provenance, and local naming for BEDC.Derived.RationalIntervalHullUp.

Chapter 9134

A Concrete Naming Certificate for $\mathsf{FiniteComparisonBoundary}^{\uparrow}$

Following Bishop and Bridges on located real comparison, the $\mathsf{FiniteComparisonBoundary}^{\uparrow}$ packet realizes the process-level comparison surface of section 9238.4 as finite rows. It sits after the located-real comparison packet of chapter 1730, the comparison-modulus route of chapter 5813, the finite observation boundary of chapter 9102, the finite-window separation packet of chapter 1202, and the dyadic tolerance arithmetic of chapter 291. The packet names finite close and apartness windows; it does not export total trichotomy, host real equality, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9134.1 (Finite comparison boundary carrier)**.** *A* $\mathsf{FiniteComparisonBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, M, W, Q, A, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the first located-real endpoint row, read through* $\mathsf{Real}^{\uparrow}$ *and its displayed regular rational presentation.*
2. *Y is the second located-real endpoint row, read through* $\mathsf{Real}^{\uparrow}$ *and its displayed regular rational presentation.*
3. *M is the finite comparison-modulus row selecting the common observation thresholds for the requested tolerance.*
4. *W is the finite apartness-window row selecting the precision at which a positive separation may be read.*
5. *Q is the rational close row, recording the displayed* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *close comparison at the selected window.*
6. *A is the rational apart row, recording the displayed positive separation when the apartness window is present.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for both endpoint presentations at the selected thresholds.*
8. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request row fixing the inspected finite windows.*
9. *H records componentwise* hsame *transport for endpoints, moduli, windows, rational close and apart rows, readback, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a comparison request through* $X, Y, M, W, Q, A, R, S, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{FiniteComparisonBoundary}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{FiniteObservationBoundary}^{\uparrow}$, $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteComparisonBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted boundary packets only through $X, Y, M, W, Q, A, R, S, H, C, P, N$. It has no coordinate for total trichotomy, host real equality, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9134.2 (Finite comparison boundary process-level obligations)**.** *For every accepted* $\mathsf{FiniteComparisonBoundary}^{\uparrow}$ *carrier $B = (X, Y, M, W, Q, A, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteComparisonBoundar}}$ *surface consists exactly of endpoint exposure, comparison-modulus exposure, apartness-window exposure, rational close and apart rows, regular rational readback, finite stream-window exposure, componentwise transport, displayed replay, provenance, and local naming. Thus every finite tolerance comparison or finite apartness witness is a process-level route through displayed moduli and finite rational observations.*

Proof. Project the accepted carrier. The endpoint rows X, Y are the only real-facing coordinates, and they are not compared as completed host points.

Read the modulus and window rows next. The comparison modulus M selects the finite thresholds for a close request, while W selects the finite precision at which an apartness witness may be inspected.

The rational evidence is then confined to Q, A, R, S. The close row Q and apart row A are read only at the windows fixed by S and through the regular rational readback row R.

Finally H, C, P, N transport, replay, source, and name the same finite comparison route. Because the excluded trichotomy, host equality, quotient equality, selected representative, choice, and completeness objects are absent from the carrier, the listed rows exhaust the local process-level obligations. □

Theorem 9134.3 (Finite comparison boundary apartness nonescape)**.** *Every accepted* $\mathsf{FiniteComparisonBoundary}^{\uparrow}$ *apartness read factors through the finite route*

$$X, Y \longmapsto M, W \longmapsto S, R \longmapsto Q, A \longmapsto H, C, P, N.$$

The apartness witness therefore remains a finite window over rational readbacks and dyadic tolerance rows. It cannot be used to obtain total located-real trichotomy, host real equality, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the process-level obligation theorem. It makes $X, Y, M, W, Q, A, R, S, H, C, P, N$ the entire accepted comparison surface.

For an apartness read, the row W fixes the inspected precision before the rational apart row A is available. The stream and regular rational rows S, R determine the finite endpoint observations at that precision.

The close row Q and apart row A remain rational evidence at the displayed window; neither row decides a global trichotomy branch for arbitrary located-real presentations.

Transport, replay, provenance, and local naming preserve only this displayed route. Since no host equality, quotient equality, selected representative, choice schedule, or ambient completeness theorem appears in the packet, apartness cannot escape the finite window. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteComparisonBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteComparisonBoundary}^{\uparrow}$ packet is generated from two located-real endpoint rows, one finite comparison-modulus row, one finite apartness-window row, rational close and apart rows, RegSeqRatUp readbacks, StreamNameUp finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteComparisonBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9134.1, Definition 9134.2, and Definition 9134.3 seed the finite process-level comparison boundary for close and apartness reads over located-real presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteComparisonBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total located-real trichotomy, host real equality, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, comparison-modulus exposure, apartness-window exposure, rational close and apart rows, regular readback, finite stream-window exposure, transport, replay, provenance, and local naming for BEDC.Derived.FiniteComparisonBoundaryUp.

Chapter 9135

A Concrete Naming Certificate for $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$

The $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$ packet realizes the observer-centered frame selector of chapter 9421 as finite NameCert data. It sits after the observer-centered constructor route of chapter 9114, the frame-facing observer row of chapter 1047, the displayed Hilbert carrier of chapter 91, the solenoid-visible phase row of chapter 9075, and the focused ledger surface of chapter 1247. The packet names a records-side section dynamics only; it does not claim a host Hilbert-space primitive, a completed cosmos, a subject substance, or an ambient physical evolution operator.

Definition 9135.1 (LogarithmicSectionDynamics carrier)**.** *A* $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, \chi, W, \Phi, \Delta, \Lambda, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the reference-frame graph row, exposing only finitely recorded frame vertices and adjacency witnesses.*
2. *χ is the observer-centered selector row, recording the displayed adjacent frame choices read by the focused history.*
3. *W is the world-section bundle row, naming the frame-relative section read through* $\mathsf{Hilbert}^{\uparrow}$ *data without importing a host Hilbert-space primitive.*
4. *Φ is the frame reshaping row, recording the displayed transport from one selected frame row to the next.*
5. *Δ is the residual ledger row, storing orthogonal section residues as packet data rather than as unrecorded world content.*
6. *Λ is the logarithmic clock row, recording the finite log-coordinate read of multiplicative accumulation.*
7. *S is the solenoid-visible phase row, exposing the public phase read through* $\mathsf{SolenoidPhase}^{\uparrow}$ *while leaving prime-register fibre data inside the displayed ledger.*
8. *H records componentwise* hsame *transport for $R, \chi, W, \Phi, \Delta, \Lambda, S, C, P, N$.*
9. *C records displayed* Cont *replay from frame selection through section readback, reshaping, residual recording, logarithmic readback, and solenoid-visible phase.*

10. *P is the* Pkg *provenance over* $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$, $\mathsf{ObserverCenteredConstructorCertificate}^{\uparrow}$, $\mathsf{ObserverDirection}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{SolenoidPhase}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LogarithmicSectionDynamics}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, \chi, W, \Phi, \Delta, \Lambda, S, H, C, P, N$. It has no coordinate for a global observer, completed observer-independent world, host Hilbert-space primitive, hidden physical time flow, unlogged residual, or fibre choice outside the displayed solenoid row.

Theorem 9135.2 (LogarithmicSectionDynamics NameCert obligations)**.** *For every accepted* LogarithmicSectionDy *carrier*

$$L = (R, \chi, W, \Phi, \Delta, \Lambda, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LogarithmicSectionDynamics}^{\uparrow}}$ *surface consists exactly of frame-graph exposure through R, observer-centered selection through χ, frame-relative section readback through W, reshaping transport through Φ, residual-ledger exposure through Δ, logarithmic clock readback through Λ, solenoid-visible phase exposure through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9135.1. The displayed rows are the reference-frame graph, selector, section bundle, reshaping map, residual ledger, logarithmic clock, solenoid-visible phase, transport, replay, provenance, and local naming rows.

The frame rows are read first. Row R supplies only displayed adjacency data, row χ selects along those adjacencies, and row W exposes a section relative to the selected frame rather than a completed host world.

Next read the dynamics rows. The reshaping row Φ, residual ledger Δ, logarithmic row Λ, and solenoid-visible row S keep every section change, residual, accumulated clock value, and phase read inside the same finite packet.

Finally H, C, P, N transport, replay, source, and name that route. Since no coordinate stores a global observer, host Hilbert-space primitive, hidden physical time flow, unlogged residual, or fibre choice outside S, the displayed rows exhaust the local NameCert obligations. □

Theorem 9135.3 (LogarithmicSectionDynamics no host-Hilbert escape)**.** *Every* $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$ *consumer that reads a frame-relative section or residual ledger must first expose the selected frame row, the* $\mathsf{Hilbert}^{\uparrow}$*-named section row, the reshaping row, and the residual ledger row of Definition 9135.1. The read supplies no host Hilbert-space primitive and no observer-independent completed world.*

Proof. Apply Definition 9135.2. It places the selector row χ and the frame-relative section row W before any reshaping or residual read.

The reshaping and residual rows Φ, Δ are then consumed as displayed packet rows. Because the carrier names the section through the $\mathsf{Hilbert}^{\uparrow}$ certificate boundary and records residuals in Δ, no consumer can detach a host Hilbert space or unlogged world content from the accepted packet.

The structural rows H, C, P, N preserve that dependency route under transport, replay, provenance, and local naming. □

Falsifiable prediction: An accepted $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$ consumer that reads a section update, residual, logarithmic clock value, or solenoid-visible phase without first exposing the selected frame row and displayed packet ledger refutes the carrier surface.

Independence witness: observer-centered-constructor-certificate, observer-direction, hilbert, solenoid-phase, focused-ledger: these siblings supply constructor, frame-facing observer, Hilbert-named section, solenoid-visible phase, and focused ledger surfaces, but none combines frame selection, section reshaping, residual exposure, logarithmic readback, solenoid visibility, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LogarithmicSectionDynamics}^{\uparrow}}$.

Constructive story: A $\mathsf{LogarithmicSectionDynamics}^{\uparrow}$ packet is generated from a reference-frame graph, observer-centered selector, frame-relative section row, frame reshaping row, residual ledger, logarithmic clock row, solenoid-visible phase row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LogarithmicSectionDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9135.1, Definition 9135.2, and Definition 9135.3 seed the finite observer-centered section dynamics surface over frame selection, section reshaping, residual ledger, logarithmic readback, solenoid visibility, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LogarithmicSectionDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert-space primitives, a completed observer-independent cosmos, subject-substance semantics, global physical time flow, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked frame-graph admission, selector adjacency, section readback, reshaping transport, residual-ledger exposure, logarithmic clock readback, solenoid visibility, transport, replay, provenance, and local naming for BEDC.Derived.LogarithmicSectionDynamicsUp.

Chapter 9136

A Concrete Naming Certificate for $\mathsf{LocatedSetoid}^{\uparrow}$

Following Bishop and Bridges on constructive equality, setoids, and located apartness, the $\mathsf{LocatedSetoid}^{\uparrow}$ packet records the finite BEDC naming surface for equality that is witnessed by displayed equivalence data and separated by displayed apartness data. It sits after the membership classifier of chapter 3668, the displayed-family carrier of chapter 3600, the located-subset route of chapter 1541, the metric route of chapter 57, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a constructive equality object for Real and RegSeqRat consumers; it does not quotient representatives, identify values by host equality, use trichotomy, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9136.1 (LocatedSetoid carrier)**.** *A* $\mathsf{LocatedSetoid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, M, Q, A, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership-classifier row for the underlying displayed collection;*
2. *T is the* $\mathsf{TypeLike}^{\uparrow}$ *displayed-family row carrying endpoints, fibers, and transport requests;*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row through which distance-shaped equality tests are read;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance-readback row attached to the displayed comparison schedule;*
5. *A is the located apartness row, recording the finite positive-separation witnesses and the refusal branch for equality reads;*
6. *D is the equality-decision ledger for the inspected finite window, storing reflexivity, symmetry, transitivity, and apartness incompatibility rows;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, T, M, Q, A, D have been displayed;*
8. *H records componentwise* hsame *transport for the membership, family, metric, rational, apartness, equality-ledger, and seal rows;*
9. *C records displayed* Cont *replay from membership inspection through located apartness, rational readback, equality-ledger exposure, and Real sealing;*
10. *P records* Pkg *provenance over* $\mathsf{LocatedSetoid}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSetoid}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Quot*, host equality, trichotomy, selected representatives, or ambient Real completeness.*

Theorem 9136.2 (LocatedSetoid NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSetoid}^\uparrow$ *carrier*

$$L = (S, T, M, Q, A, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSetoid}^\uparrow}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{SetLike}^\uparrow$ *membership classifier,* $\mathsf{TypeLike}^\uparrow$ *family endpoint route,* $\mathsf{LocatedSet}^\uparrow$ *locatedness context,* $\mathsf{Met}^\uparrow$ *comparison row,* $\mathsf{RegSeqRat}^\uparrow$ *tolerance readback, located apartness row, finite equality ledger, terminal* $\mathsf{Real}^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Equality may be consumed only as this displayed finite route, and apartness may be consumed only through the positive-separation row A.*

Proof. Project the carrier of Definition 9136.1. The rows S, T, M, Q, A, D, R are the complete equality and apartness sources, while H, C, P, N are the structural rows that preserve, replay, source, and name the same packet.

The equality route is not independent of located apartness. Membership and displayed-family endpoints are read first through S, T, metric comparison enters through M, and rational tolerance readback enters through Q. The row A records the positive-separation alternative, and the ledger D records exactly the inspected equivalence and incompatibility rows.

Finally the Real-facing seal R is reached only after those finite rows have been displayed. Since no row stores quotient representatives, host equality, trichotomy, or ambient completeness, the local NameCert obligations are exhausted by the carrier rows. □

Theorem 9136.3 (LocatedSetoid Real classifier handoff)**.** *Every* $\mathsf{Real}^\uparrow$ *or* $\mathsf{RegSeqRat}^\uparrow$ *consumer that reads equality or apartness from an accepted* $\mathsf{LocatedSetoid}^\uparrow$ *packet must factor through the displayed sequence*

$$\mathsf{SetLike}^\uparrow,\ \mathsf{TypeLike}^\uparrow,\ \mathsf{LocatedSet}^\uparrow,\ \mathsf{Met}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow.$$

Concretely, the consumer first exposes the membership and displayed-family rows S, T, then the metric comparison row M, then the rational tolerance row Q, then the located apartness and finite equality ledger rows A, D, and only then reaches the Real seal R. The handoff supplies no Quot *primitive, no host equality of representatives, no trichotomy split, and no ambient Real completeness theorem.*

Proof. Start with the carrier rows of Definition 9136.1. A Real-facing consumer has only one seal row, namely R, and that row is packaged after the membership, family, metric, rational, apartness, and equality-ledger rows.

Apply the obligation surface of Definition 9136.2. It forces equality and apartness reads to remain inside $S, T, M, Q, A, D, H, C, P, N$ before R is exposed.

Thus the classifier handoff is a finite route through the named sibling chapters. The excluded equality sources have no carrier coordinate, no replay edge, no provenance row, and no local NameCert row. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSetoid}^\uparrow}$**.**

Constructive story: A $\mathsf{LocatedSetoid}^\uparrow$ packet is generated from SetLike membership, TypeLike displayed-family data, LocatedSet context, Metric comparison, RegSeqRat tolerance readback, located apartness, finite equality-ledger rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocatedSetoid}^{\uparrow}}$, seed)

Scope closed: Definition 9136.1, Definition 9136.2, and Definition 9136.3 seed the constructive setoid equality and located-apartness carrier over SetLike$^{\uparrow}$, TypeLike$^{\uparrow}$, LocatedSet$^{\uparrow}$, Met$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedSetoid}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Quot equality, host equality of representatives, trichotomy, selected representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, equality-ledger stability, located-apartness incompatibility, Real classifier handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSetoidUp.

Chapter 9137

A Concrete Naming Certificate for $\mathsf{BishopCauchyLocatedLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyLocatedLimit}^{\uparrow}$ packet records the finite route by which a located regular Cauchy source exposes a completion-facing limit window before any quotient or selected representative is available. It sits over the regular rational readback of chapter 301, the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names located Cauchy limit extraction as displayed BEDC data; it does not assert quotient completion, choose a representative, invoke countable choice, or prove ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9137.1 (BishopCauchyLocatedLimit carrier)**.** *A* $\mathsf{BishopCauchyLocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, W, D, O, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular Cauchy source row, read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational sequence surface.*
2. *W is the located window row, recording the finite interval and modulus request under which the source is inspected.*
3. *D is the dyadic tolerance ledger supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *O is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row used to read the source only through named windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed after R, W, D, O have been displayed.*
6. *H records componentwise* hsame *transport for the regular source, located window, dyadic tolerance, finite observation, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$R, W, D, O \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyLocatedLimit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyLocatedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, D, O, E, H, C, P, N$. It has no coordinate for quotient completion, a selected representative, countable choice, a host limit object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9137.2 (BishopCauchyLocatedLimit NameCert obligations)**.** *For every accepted* BishopCauchyLocatedLi *carrier*

$$L = (R, W, D, O, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyLocatedLimit}^{\uparrow}}$ *surface consists exactly of regular Cauchy source exposure through R, located-window exposure through W, dyadic tolerance exposure through D, StreamName finite observation through O, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9137.1. The rows R, W, D, O expose the regular source, located window, dyadic tolerance, and finite observation data before any limit-facing read can occur.

The Real seal row E is downstream of those displayed rows. A consumer therefore receives a named finite extraction route rather than a quotient completion object, a selected representative, or a countable-choice schedule.

Finally H, C, P, N transport, replay, source, and name the same route. Since no host limit object or ambient completeness theorem is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyLocatedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyLocatedLimit}^{\uparrow}$ packet is generated from a RegSeqRat regular Cauchy source, a located window, a DyadicRatCore tolerance ledger, StreamName finite observations, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyLocatedLimit}^{\uparrow}}$, seed)

Scope closed: Definition 9137.1 and Definition 9137.2 seed the located Cauchy limit extraction route over the RegSeqRat, StreamName, DyadicRatCore, and Real spine.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyLocatedLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked regular source admission, located-window readback, dyadic tolerance, finite observation, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyLocatedLimitUp.

Chapter 9138

A Concrete Naming Certificate for $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$

The $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ packet realizes the Cook cyclic-tag production branch of section 9390.5 as finite NameCert data. It sits beside the cyclic-tag carrier of chapter 3764, the Rule 110 local-stencil observation of chapter 6004, the Cook frontier budget of chapter 6849, and the cellular pattern catalog of chapter 995. The packet records production rows, tape-symbol rows, queue-head rows, step-budget rows, replay rows, provenance rows, and local naming before any consumer may read the Cook branch as a cellular production surface.

Definition 9138.1 (Rule 110 production-queue carrier)**.** *A* $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (C, L, B, T, Hd, S, R, V, E, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CyclicTagSystem}^{\uparrow}$ *source row, exposing the accepted cyclic-tag production family and the inspected word row;*
2. *L is the* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ *row, recording the local Rule 110 stencil window used to read each simulated production step;*
3. *B is the* $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$ *row, fixing the displayed production count, tape-symbol count, and step-budget boundary;*
4. *T is the tape-symbol queue row, listing the finite queue segment, head symbol, and tail segment inspected by the branch;*
5. *Hd is the queue-head row, naming the active production, head position, and local rotation state;*
6. *S is the step-budget row, tying each production dequeue and enqueue event to the finite Cook frontier budget;*
7. *R is the replay row, displaying the cyclic-tag update and the corresponding Rule 110 local-stencil observations in lockstep;*
8. *V is the cellular pattern-catalog row, recording which glider or ether-pattern tags are read from the replayed stencil window;*
9. *E is the obstruction and refusal row, recording when a requested production read exceeds the displayed queue, tape, or budget rows;*

10. *K records componentwise* hsame *transport for source, stencil, budget, tape, head, replay, catalog, obstruction, provenance, and local naming rows;*
11. *P records* Pkg *provenance over* $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$, $\mathsf{CyclicTagSystem}^{\uparrow}$, $\mathsf{RuleOneTenLocalStencilObserva}$ $\mathsf{RuleOneTenCookFrontierBudget}^{\uparrow}$, $\mathsf{CellularPatternCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenProductionQueue}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unbounded cyclic-tag run, hidden Rule 110 evaluator, unstated universality theorem, external tape oracle, or detached complexity estimate.

Theorem 9138.2 (Rule 110 production-queue NameCert obligations)**.** *For every accepted* RuleOneTenProductionQ *carrier*

$$Q = (C, L, B, T, Hd, S, R, V, E, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenProductionQueue}^{\uparrow}}$ *surface consists exactly of carrier admission for the cyclic-tag source, Rule 110 local-stencil observation, Cook frontier budget, tape-symbol queue, queue-head state, step-budget ledger, lockstep replay row, cellular pattern-catalog row, obstruction row, componentwise* hsame *transport, displayed replay, package provenance, and local naming. A consumer may inspect a production read only after all of those displayed rows are available.*

Proof. Project the carrier of Definition 9138.1. The source and cellular rows are C, L, B, V, while the queue-specific rows are T, Hd, S, R, E.

Read the branch in dependency order. The cyclic-tag row supplies the production family, the local-stencil row supplies the Rule 110 observation window, and the Cook frontier budget supplies the finite count and step boundary. Only then can the tape-symbol queue, queue head, and lockstep replay rows be inspected.

The structural rows preserve the same packet. Transport is read through K, provenance through P, and local naming through N. Since the carrier has no hidden evaluator, unbounded tape oracle, unstated universality theorem, or detached complexity estimate, the listed rows exhaust the NameCert obligations. □

Theorem 9138.3 (Rule 110 production-queue Cook branch handoff)**.** *Every Cook-branch consumer of an accepted* $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ *packet factors through*

$$\mathsf{CyclicTagSystem}^{\uparrow},\ \mathsf{RuleOneTenLocalStencilObservation}^{\uparrow},\ \mathsf{RuleOneTenCookFrontierBudget}^{\uparrow},\ \mathsf{CellularPatternCatalog}^{\uparrow}.$$

Concretely, the consumer first displays the cyclic-tag production row C, then the Rule 110 local-stencil row L, then the budget row B, then the finite tape, head, step, replay, catalog, obstruction, transport, provenance, and naming rows $T, Hd, S, R, V, E, K, P, N$. The handoff is a finite production queue, not an unbounded universality proof or a hidden tape oracle.

Proof. Start with the carrier. A Cook-branch read has no public coordinate except the displayed rows $C, L, B, T, Hd, S, R, V, E, K, P, N$.

Apply the NameCert obligations of Definition 9138.2. They force a production read to pass through the cyclic-tag source, local stencil observation, budget, queue, head, replay, catalog, obstruction, transport, provenance, and naming rows in the same finite packet.

Thus the handoff is exactly the listed finite route. Any claim requiring an unbounded run, hidden evaluator, unstated universality theorem, external tape oracle, or detached complexity estimate lacks a carrier row and cannot be projected from the packet. □

Falsifiable prediction: Every accepted $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ Cook-branch read displays cyclic-tag source rows, Rule 110 local-stencil observations, Cook frontier budget rows, tape-symbol queue rows, queue-head rows, step-budget rows, replay rows, cellular pattern-catalog rows, obstruction rows, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Independence witness: cyclic-tag-system, rule110-local-stencil-observation, rule110-cook-frontier-budget, cellularpatterncatalog: none is bijective to $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ because the queue packet jointly carries the cyclic-tag source, local Rule 110 stencil observations, finite Cook budget, tape-symbol queue, queue-head state, step-budget ledger, lockstep replay, catalog tags, obstruction rows, transport, provenance, and local naming.

Theorem 9138.4 (Rule 110 production-queue TasteGate obligation)**.** *The follow-on TasteGate surface for* $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ *must provide* Nontrivial $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ *and* FieldFaithful $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ *instances for the same displayed carrier* $Q = (C, L, B, T, Hd, S, R, V, E,$ *Nontriviality must exhibit an accepted finite queue packet, and field faithfulness must keep the cyclic-tag source, stencil observation, budget, tape-symbol queue, queue-head state, replay, catalog, obstruction, transport, provenance, and local naming rows distinct.*

Proof. The obligation is read from Definition 9138.1. The packet has separate rows for cyclic-tag source data, Rule 110 stencil observations, finite budget data, queue state, replay, catalog classification, obstruction, transport, provenance, and local naming.

Nontriviality must therefore inhabit that finite packet rather than a bare tag. Field faithfulness must prevent the source row from replacing the stencil row, the budget row from replacing the replay row, or the catalog row from replacing obstruction and provenance.

The excluded hidden evaluator and unbounded tape oracle are absent from the carrier and from the handoff theorem. The TasteGate instances therefore have to target the displayed production queue itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenProductionQueue}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenProductionQueue}^{\uparrow}$ packet is generated from CyclicTagSystem source rows, Rule 110 local-stencil observations, Cook frontier budget rows, tape-symbol queue rows, queue-head state, step-budget ledger, lockstep replay, cellular pattern-catalog tags, obstruction rows, componentwise hsame transport, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenProductionQueue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9138.1, Definition 9138.2, Definition 9138.3, and Definition 9138.4 seed the finite production-queue packet for the Rule 110 Cook cyclic-tag branch.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenProductionQueue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Cook universality, an unbounded cyclic-tag run, a hidden Rule 110 evaluator, an external tape oracle, a detached complexity estimate, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, queue replay, local-stencil alignment, budget exactness, catalog readback, obstruction nonescape, Nontrivial and FieldFaithful TasteGate instances, transport, provenance, and local naming for BEDC.Derived.RuleOneTenProductionQueueUp.

Chapter 9139

A Concrete Naming Certificate for $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$

The $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$ packet realizes the route of chapter 9285 as finite normal-mode, prime-ledger, and unit-boundary NameCert data. It reads prime frequency registers through chapter 64, finite exponent and support ledgers through chapter 28, rational coefficient rows through chapter 37, and the terminal real comparison surface through chapter 38. The packet names a BEDC certificate surface for the arithmetic string route; it does not prove the Riemann hypothesis, construct all zeta-zero certificates, import a physical string model, or assert a host spectral theorem.

Definition 9139.1 (ArithmeticSpectralStringCertificate carrier)**.** *An* $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, Q, L, U, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite normal-mode packet, listing the active logarithmic modes inspected by the certificate.*
2. *Q is the rational coefficient row, read through* $\mathsf{Rat}^{\uparrow}$*, for the finite amplitude data attached to the active modes.*
3. *L is the prime ledger row, read through* $\mathsf{Prime}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$*, recording the factorization data behind the displayed logarithmic heights.*
4. *U is the unit-boundary row saying that the active centered channels are inspected only through unit phase data.*
5. *Z is the zero-readout row, exposing the finite cancellation certificate before any global zero-coverage claim is available.*
6. *H records componentwise* hsame *transport for modes, coefficients, prime ledgers, unit-boundary rows, zero readouts, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M, Q, L, U \longmapsto Z.$$

8. *P is the* Pkg *provenance over* $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, L, U, Z, H, C, P, N$. It has no coordinate for a physical string model, host spectral operator, all-zero coverage theorem, radial scale reservoir, or completed zeta proof.

Theorem 9139.2 (ArithmeticSpectralStringCertificate NameCert obligations)**.** *For every accepted* $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$ *carrier*

$$S = (M, Q, L, U, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}}$ *surface consists exactly of finite normal-mode exposure through M, rational coefficient exposure through Q, prime-ledger exposure through L, unit-boundary exposure through U, zero-readout exposure through Z, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9139.1. The rows M, Q, L, U expose the finite mode packet, coefficient data, prime ledger, and unit-boundary discipline before the zero-readout row can be inspected.

The zero-readout row Z is therefore a finite cancellation certificate, not a theorem that every accepted zeta zero has already been covered. The packet also has no carrier coordinate for a physical string model, host spectral operator, or radial scale reservoir.

Finally H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert obligations and leave global spectral coverage to separate source rows. □

Falsifiable prediction: A completed formal route for $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$ must expose a finite normal-mode row, prime-ledger row, unit-boundary row, and zero-readout row before any RH-facing consumer can inspect the certificate; a purported proof that bypasses one of these rows is outside this NameCert surface.

Independence witness: The packet is independent from the sibling surfaces prime, nat, rat, real, and zeta-zero-certificate-tower in the following sense: those chapters may supply source rows, but none of them alone supplies the combined finite normal-mode, prime-ledger, unit-boundary, and local NameCert packet carried by $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$.

A Lean follow-up must provide Nontrivial and FieldFaithful instances for the carrier before the chapter can be promoted beyond statement-level paper evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}}$**:**

Constructive story: An $\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}$ packet is generated from finite normal modes, rational coefficient rows, prime and Nat ledgers, unit-boundary rows, a zero-readout row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}}$, seed)

Scope closed: Definition 9139.1 and Definition 9139.2 seed the arithmetic spectral string certificate route over finite modes, prime ledgers, unit-boundary data, and zero-readout rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ArithmeticSpectralStringCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove the Riemann hypothesis, construct all zeta-zero certificates, import a physical string model, supply a host spectral theorem, provide Lean verification, or bridge the packet to a public schema.

Upgrade path: Obligation closure requires checked finite-mode admission, prime-ledger admission, unit-boundary exposure, zero-readout exposure, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.ArithmeticSpectralStringCertificateUp.

Chapter 9140

A Concrete Naming Certificate for $\mathsf{CompactZeroDimensionalMetric}^{\uparrow}$

Following Bishop and Bridges on compact metric data and Kelley's zero-dimensional compact spaces, the $\mathsf{CompactZeroDimensionalMetric}^{\uparrow}$ packet records the finite BEDC surface by which a compact metric source also exposes a clopen-basis schedule. It sits over the compact metric source of chapter 129, the metric carrier of chapter 57, the topology carrier of chapter 88, the total-bounded finite-net ledger of chapter 127, and the real-valued diameter and tolerance readback rows supplied by chapter 38. The packet is a finite naming certificate for compact zero-dimensional metric presentations; it is not an open-cover compactness theorem, not a selected arbitrary clopen refinement, and not a quotient compactum.

Definition 9140.1 (CompactZeroDimensionalMetric carrier)**.** *A* $\mathsf{CompactZeroDimensionalMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (K, M, T, B, F, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact-metric finite-net and completion-facing boundary inherited from chapter 129;*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row, keeping the distance read, triangle ledger, and metric observation surface visible through chapter 57;*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *row, recording the open-set and finite-observation topology carrier used by the zero-dimensional read;*
4. *B is a finite clopen-basis schedule: for each displayed tolerance window and observed point row, it gives a finite list of clopen neighborhoods whose union covers the relevant finite-net window and whose members refine the requested open row;*
5. *F is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger, binding each clopen-basis window to a finite net, mesh row, and covering schedule before compact-metric data is exported;*
6. *D is the* $\mathsf{Real}^{\uparrow}$*-valued diameter and tolerance readback row, recording the displayed bounds used to compare clopen windows with metric balls;*
7. *H stores componentwise* hsame *transport for the compact source, metric row, topology row, clopen-basis schedule, finite-net ledger, diameter readback, provenance, and local naming row;*

8. *C stores displayed* Cont *replay for reading the compact metric source, the metric and topology rows, the clopen-basis schedule, the finite-net ledger, the diameter row, and the local certificate row;*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactZeroDimensionalMetric}^{\uparrow}}$ *row.*

The carrier names zero-dimensional compact-metric data only through the displayed finite rows. It does not add an ambient open-cover compactness proof, quotient equality of compact spaces, arbitrary clopen choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 9140.2 (CompactZeroDimensionalMetric NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactZeroDimension}}$ *surface for the carrier of Definition 9140.1 consists of seven finite obligations:*

1. *carrier admission: the accepted object is the displayed packet $Z = (K, M, T, B, F, D, H, C, P, N)$;*
2. *compact-source visibility: the* $\mathsf{CompactMetric}^{\uparrow}$ *row K is read before any compact-metric conclusion is exported;*
3. *metric-topology compatibility: the* $\mathsf{Met}^{\uparrow}$ *row M and* $\mathsf{Topology}^{\uparrow}$ *row T remain visible when a clopen window is inspected;*
4. *finite clopen coverage: each row of B supplies only the finite clopen windows displayed for the current finite-net and tolerance request;*
5. *total-bounded net binding: the ledger F ties every clopen-basis read to a finite-net schedule and never to an unbounded cover search;*
6. *real-valued readback: the diameter and tolerance comparisons pass through the* $\mathsf{Real}^{\uparrow}$ *row D before any metric-ball refinement is exposed;*
7. *non-escape: every downstream read factors through $K, M, T, B, F, D, H, C, P, N$ and has no route to open-cover compactness, arbitrary clopen choice, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the projection of the displayed packet in Definition 9140.1. The compact source, metric carrier, topology carrier, clopen-basis schedule, finite-net ledger, real-valued readback, transport, replay, provenance, and local name are exactly the coordinates of that packet.

The compact-source and metric-topology obligations are componentwise reads. A consumer must first expose the $\mathsf{CompactMetric}^{\uparrow}$ source row and then keep the $\mathsf{Met}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ rows visible when a clopen window is inspected.

The finite coverage and net-binding obligations use the rows B and F. The clopen-basis schedule is indexed by displayed finite-net and tolerance requests, so it is a finite schedule for the current packet rather than a selected family of all clopen refinements.

The real-valued comparison obligation is the row D. Diameter and tolerance readback pass through the displayed $\mathsf{Real}^{\uparrow}$ seal before any metric-ball refinement is consumed.

The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming. Since no coordinate stores open-cover compactness, arbitrary clopen choice, quotient equality, or ambient Real completeness, no accepted read can export those endpoints. □

Theorem 9140.3 (CompactZeroDimensionalMetric clopen-basis route)**.** *Every clopen-basis read exported by an accepted* $\mathsf{CompactZeroDimensionalMetric}^{\uparrow}$ *carrier factors through*

$$K, M, T, B, F, D, H, C, P, N.$$

It must expose the compact metric source, metric row, topology row, finite clopen-basis schedule, total-bounded finite-net ledger, real-valued diameter and tolerance readback, componentwise hsame *transport, displayed* Cont *replay, package provenance, and local* NameCert *row before any zero-dimensional compact-metric consumer can inspect the basis. The route exports finite clopen windows only, not open-cover compactness, unrestricted clopen choice, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a clopen-basis request. By the carrier definition, the request can inspect only the displayed rows $K, M, T, B, F, D, H, C, P, N$.

Read the source rows first. The compact-metric row K supplies the compact source, while M and T keep the metric and topology carriers visible before any clopen window is read.

Next consume the finite schedule. The clopen-basis row B is indexed by the total-bounded finite-net ledger F, and the diameter or tolerance comparison is read through the real-valued row D.

Finally replay the structural rows H, C, P, N. These rows transport, replay, source, and name the same finite packet, so the exported basis remains a displayed zero-dimensional compact-metric schedule and cannot be reread as the refused compactness, choice, quotient, or completeness endpoints. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactZeroDimensionalMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactZeroDimensionalMetric}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source, metric and topology rows, a finite clopen-basis schedule, a total-bounded finite-net ledger, real-valued diameter readback, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactZeroDimensionalMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite compact-metric carrier, metric-topology route, clopen-basis schedule, total-bounded finite-net binding, real-valued readback, and downstream non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactZeroDimensionalMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove open-cover compactness, choose arbitrary clopen refinements, identify quotient compact spaces, or close ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, metric-topology compatibility, finite clopen coverage, total-bounded net binding, real-valued readback exactness, and non-escape for the displayed packet.

Chapter 9141

A Concrete Naming Certificate for $\mathsf{AronszajnAscoli}^{\uparrow}$

Following Aronszajn and Ascoli, the $\mathsf{AronszajnAscoli}^{\uparrow}$ packet records the finite NameCert surface for compactness of a displayed equicontinuous family only after the compact source, continuous-map rows, finite-net modulus data, and terminal real-name handoff are visible. It sits beside the Arzela–Ascoli finite-net packet of chapter 7576, the equicontinuity family surface of chapter 413, the compact metric source of chapter 129, and the continuous-map carrier of chapter 130. The packet names only the BEDC finite route; it is not a host function-space compactness theorem, a selected subnet, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9141.1 (Aronszajn–Ascoli carrier)**.** *An* $\mathsf{AronszajnAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, E, N_f, W, R, H, C, P, L)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row and F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows over that source.*
2. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, and N_f is the finite-net handoff row consumed from the Arzela–Ascoli finite-net sibling.*
3. *W records the finite observation windows through* $\mathsf{StreamName}^{\uparrow}$*, and R records the regular rational readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal used by each inspected map value.*
4. *H, C, P, L are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{AronszajnAscoli}^{\uparrow}}$ *row.*

Theorem 9141.2 (Aronszajn–Ascoli NameCert obligations)**.** *For every accepted* $\mathsf{AronszajnAscoli}^{\uparrow}$ *carrier $A = (K, F, E, N_f, W, R, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{AronszajnAscoli}^{\uparrow}}$ *surface consists exactly of compact-source admission, continuous-map family admission, equicontinuity visibility, finite-net handoff, finite observation windows, regular rational readback, terminal real-name sealing, componentwise transport, displayed replay, provenance, and local naming. Every compactness-facing consumer must read those rows before citing* $\mathsf{AronszajnAscoli}^{\uparrow}$*.*

Proof. Project the carrier of Definition 9141.1. The rows K, F, E, N_f, W, R are the only mathematical rows available before structural transport and local naming are inspected.

The compact and continuous-map dependencies are supplied by chapter 129 and chapter 130. The family-wide modulus information is supplied by chapter 413, while the finite-net handoff is supplied by chapter 7576.

Finally the readback and sealing rows pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the local name L can be exported. No carrier coordinate remains for a host function-space compactness theorem, selected subnet, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 9141.3 (Aronszajn–Ascoli finite equicontinuity route)**.** *Every accepted* $\mathsf{AronszajnAscoli}^{\uparrow}$ *compactness read factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{EquicontinuityFamily}^{\uparrow}, \mathsf{ArzelaAscoliFiniteNet}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{hs}$$

Thus the equicontinuity witness and finite-net witness are displayed before any compactness-facing handoff is public.

Proof. Start with the compact source row K and displayed continuous family row F. They bind every inspected map value to the same compact-metric source and the same finite family ledger.

Next read the equicontinuity row E together with the finite-net handoff row N_f. These rows provide the shared modulus and the finite source-net boundary that an Arzela–Ascoli style consumer may inspect.

Finally read W, R, H, C, P, L. The finite observation windows and real-name seals make each inspected value visible before local naming, so the compactness handoff remains a finite BEDC route and cannot import selected subnets or host function-space compactness. □

Theorem 9141.4 (Aronszajn–Ascoli compact handoff)**.** *The* $\mathsf{AronszajnAscoli}^{\uparrow}$ *packet may hand a compactness-facing consumer to the Arzela–Ascoli and compact-metric siblings only through the carrier rows of Definition 9141.1. In particular, the handoff cites chapter 412, chapter 129, chapter 413, and chapter 130 as displayed NameCert dependencies, not as hidden ambient theorems.*

Proof. Use Definition 9141.2 to identify the complete local obligation surface. The only handoff rows are the displayed compact source, continuous family, equicontinuity, finite-net, finite-window, readback, real-seal, and structural rows.

Apply Definition 9141.3. It orders the compact and equicontinuity dependencies before the real-name handoff, so the consumer cannot bypass them by citing an ambient compactness principle.

Therefore every compact handoff is a sibling-lattice read through the named concrete-instance chapters. The packet exports no selected subnet, quotient map equality, countable-choice schedule, host function-space compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{AronszajnAscoli}^{\uparrow}}$.

Constructive story: An $\mathsf{AronszajnAscoli}^{\uparrow}$ packet is generated from CompactMetric, ContinuousMap, EquicontinuityFamily, ArzelaAscoliFiniteNet, StreamName, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AronszajnAscoli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9141.1, Definition 9141.2, Definition 9141.3, and Definition 9141.4 seed the finite compactness-facing NameCert surface over compact-source, continuous-family, equicontinuity, finite-net, window, readback, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AronszajnAscoli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space compactness, selected subnets, quotient equality of maps, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, continuous-family admission, equicontinuity visibility, finite-net handoff, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.AronszajnAscoliUp.

Chapter 9142

A Concrete Naming Certificate for $\mathsf{ZeroLedgerRigidity}^{\uparrow}$

The $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ packet realizes the zero-ledger rigidity route of section 9414.6 as finite visible-route, replay, provenance, and no-hidden-ledger NameCert data. It sits beside the fixed-half witness of chapter 9046, the local-global zero exclusion certificate of chapter 9056, the reality-constrained open-fit boundary of chapter 1132, the finite observation boundary of chapter 9102, and the finite comparison boundary of chapter 9134. The packet names a zero-ledger rigidity surface only; it is not a host proof of RH, a selected zero list, an analytic-continuation theorem, a hidden completed-carrier equality, or an ambient real-completeness principle.

Definition 9142.1 (ZeroLedgerRigidity carrier)**.** *A* $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (V, Q, F, E, O, R, G, H, C, P, N)$$

with the following displayed rows:

1. *V is the visible zero-ledger route, exposing the same tower point, zeta-coordinate row, zero atom, and information-band row before any rigidity read is public.*
2. *Q is the fixed-half comparison row consumed from* $\mathsf{FixedHalfZeroWitness}^{\uparrow}$*.*
3. *F is the off-fixed exclusion row consumed from* $\mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}$*.*
4. *E is the finite evidence row tying the zero-ledger atom to open-fit, observation-boundary, and finite-comparison witnesses.*
5. *O is the no-hidden-ledger row refusing zero data not displayed in V, E.*
6. *R is the replay row recording the order in which visible route, fixed-half comparison, exclusion, and evidence rows are consumed.*
7. *G is the provenance row over the vision route, the fixed-half sibling, the zero-exclusion sibling, open fit, observation boundary, and comparison boundary.*
8. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *packaging, and local* $\mathsf{NameCert}_{\mathsf{ZeroLedgerRigidity}^{\uparrow}}$ *naming.*

The carrier has no coordinate for a hidden zero ledger, selected off-fixed zero, host analytic continuation, quotient completed-carrier equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9142.2 (ZeroLedgerRigidity NameCert obligations)**.** *For every accepted* $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ *carrier* $Z = (V, Q, F, E, O, R, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ZeroLedgerRigidity}^{\uparrow}}$ *surface consists exactly of visible-route admission, fixed-half comparison, off-fixed exclusion, finite evidence exposure, no-hidden-ledger refusal, replay ordering, provenance, transport, displayed continuation, packaging, and local naming. Every rigidity-facing consumer must display those rows before reading the packet.*

Proof. Project the carrier of Definition 9142.1. The rows V, Q, F, E, O are the mathematical surface: visible zero route, fixed-half comparison, exclusion, finite evidence, and no-hidden-ledger refusal.

The route rows are consumed from the siblings. $\mathsf{FixedHalfZeroWitness}^{\uparrow}$ supplies the fixed surface, $\mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}$ supplies the off-fixed exclusion gate, and the open-fit, observation-boundary, and finite-comparison siblings supply the visible evidence boundary.

Finally read R, G, H, C, P, N. Replay fixes the order, provenance records the sibling sources, and transport, continuation, package, and local naming keep the export inside the displayed packet. Since no carrier coordinate remains, a hidden ledger or selected off-fixed zero cannot be read by the accepted NameCert. □

Theorem 9142.3 (ZeroLedgerRigidity route nonescape)**.** *Every accepted* $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ *rigidity read factors through*

$$\mathsf{FixedHalfZeroWitness}^{\uparrow}, \mathsf{LocalGlobalZeroExclusionCertificate}^{\uparrow}, \mathsf{RealityConstrainedOpenFit}^{\uparrow}, \mathsf{FiniteObservationBoundary}^{\uparrow}$$

It cannot obtain a zero atom from a hidden ledger, cannot bypass the fixed-half comparison row, and cannot replace finite evidence with host analytic continuation or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the visible route row V. It binds the tower point, zeta-coordinate row, zero atom, and information-band row before rigidity is available.

Next consume Q, F, E, O. The fixed-half comparison and off-fixed exclusion rows place the zero-ledger atom against the fixed surface, while the finite evidence and no-hidden-ledger rows force all observation and comparison data to be public.

Finally use R, G, H, C, P, N to replay and name the same route. Since every export is already listed in the carrier, a consumer cannot add hidden zero data, selected off-fixed witnesses, host analytic continuation, quotient completed-carrier equality, countable choice, or ambient real completeness. □

Falsifiable prediction: An accepted $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ rigidity read is refuted if it obtains a zero atom, fixed-half comparison, off-fixed exclusion, or evidence row without first exposing the visible route row V and the no-hidden-ledger row O.

Independence witness: fixed-half-zero-witness, local-global-zero-exclusion-certificate, reality-constrained-open-fit, finite-observation-boundary, finite-comparison-boundary: no listed sibling is bijective to $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ because this packet jointly carries visible zero-route data, fixed-half comparison, off-fixed exclusion, finite evidence, no-hidden-ledger refusal, replay, provenance, transport, packaging, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{ZeroLedgerRigidity}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ZeroLedgerRigidity}^{\uparrow}$ instances before a checked carrier-alignment theorem can serve as a public TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroLedgerRigidity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZeroLedgerRigidity}^{\uparrow}$ packet is generated from a visible zero-ledger route, fixed-half comparison, off-fixed exclusion, finite evidence, no-hidden-ledger refusal, replay ordering, provenance, componentwise hsame transport, displayed Cont replay, Pkg packaging, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{ZeroLedgerRigidity}\uparrow}$, seed)

Scope closed: Definition 9142.1, Definition 9142.2, Definition 9142.3, the falsifiable prediction, the independence witness, and the Lean follow-up statement seed the finite visible-route rigidity packet over fixed-half comparison, off-fixed exclusion, evidence, no-hidden-ledger, replay, provenance, transport, package, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZeroLedgerRigidity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host proof of RH, selected zero lists, host analytic continuation, hidden completed-carrier equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked visible-route admission, fixed-half comparison, off-fixed exclusion, finite evidence exposure, no-hidden-ledger refusal, replay ordering, provenance, transport, packaging, local naming, and the Nontrivial and FieldFaithful rows for BEDC.Derived.ZeroLedgerRigidityUp.

Chapter 9143

A Concrete Naming Certificate for $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ packet records the finite-net extraction route by which a displayed continuous map on a compact metric source yields a uniform-continuity budget only through inspected compact, metric, rational, and real-name rows. It sits after the compact carrier of chapter 60, the metric carrier of chapter 57, the continuous-map carrier of chapter 130, the rational carrier of chapter 37, and the terminal real-name seal of chapter 38. The packet names a constructive finite-net extractor; it is not a host Heine–Cantor theorem, a selected infinite cover, quotient function equality, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 9143.1 (Compact uniform-continuity extractor carrier)**.** *A* $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, M, F, Q, R, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row, carrying the finite cover or finite-net admission surface available to the extractor.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row on the same displayed carrier as K.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose local continuity reads are inspected only over the compact-metric source.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *radius and tolerance ledger used to compare finite net cells.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *distance and value-readback seal reached from the rational ledger.*
6. *W is the finite-net witness row: it records the chosen inspected net cells, the local continuity budgets attached to them, and the common modulus bound exported to consumers.*
7. *H records componentwise* hsame *transport for $K, M, F, Q, R, W, C, P, N$.*
8. *C records displayed* Cont *replay from compact-metric admission through local continuity, finite-net comparison, real readback, and exported uniform budget.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient compactness oracle, a host function space, an infinite cover selector, quotient equality of maps, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9143.2 (Compact uniform-continuity extractor NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ *carrier*

$$E = (K, M, F, Q, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityExtractor}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, metric-source admission through M, continuous-map local reads through F, rational radius and tolerance comparison through Q, real-name distance and value seals through R, finite-net extraction through W, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9143.1. The source rows K, M, F make compactness, metric distance, and local continuity visible before any uniform budget can be read.

The extraction row is W. It is reached only after the rational ledger Q and the real-name readback row R have displayed the finite radii, tolerance comparisons, and value-distance seals needed by the finite-net argument. Thus a consumer cannot replace the extractor by an ambient compactness theorem or by a selected infinite cover.

The structural rows preserve the same route. Transport through H, replay through C, provenance through P, and local naming through N export the finite packet named above. Since the carrier has no coordinate for host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ read exposes $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite-net extraction, componentwise transport, displayed replay, provenance, and local naming before any uniform-continuity budget is public.

Independence witness: compact, metric, continuousmap, rat, real: $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact finite-net admission, metric distance, continuous-map local reads, rational tolerance comparison, Real readback, finite-net extraction, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityExtractor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformContinuityExtractor}^{\uparrow}$ packet is generated from a compact source row, a metric source row, a continuous-map row, rational radius and tolerance rows, real-name distance and value seals, finite-net extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformContinuityExtractor}^{\uparrow}}$, seed)

Scope closed: Definition 9143.1 and Definition 9143.2 bind the finite-net extractor surface over compact, metric, continuous-map, rational, real-name, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformContinuityExtractor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, a host Heine–Cantor theorem, quotient equality of maps, selected infinite covers, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-source visibility, metric-source visibility, continuous-map local-read routing, rational tolerance comparison, real-name readback, finite-net extraction, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformContinuityExtractorUp.

Chapter 9144

A Concrete Naming Certificate for HyperbolicBusemannBoundaryBudget$^{\uparrow}$

The HyperbolicBusemannBoundaryBudget$^{\uparrow}$ packet realizes the boundary-budget branch of chapter 9463 as finite BEDC NameCert data. It sits after the hyperbolic metric row of chapter 3474, the Busemann boundary row of chapter 3732, the boundary-fixed disk transport row of chapter 6267, the horocycle row of chapter 3534, and the horosphere row of chapter 8521. The packet names the finite disk-map, metric-read, boundary, classifier-stability, provenance, and refusal rows required by observer-expansion boundary budgets; it is not a global hyperbolic model, a quotient ideal boundary, a selected geodesic ray, or a physical observer.

Definition 9144.1 (Hyperbolic Busemann-boundary budget carrier)**.** *A* HyperbolicBusemannBoundaryBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, M, U, Y, O, S, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-map readout row, keeping the inspected Poincare-disk coordinate and boundary-admission window finite.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row attached to that disk window.*
3. *U is the* HyperbolicBusemannBoundary$^{\uparrow}$ *boundary row, exposing Busemann-level readback only through displayed metric and boundary data.*
4. *Y is the boundary-fixed transport row supplied by the Poincare disk boundary-transport sibling.*
5. *O is the horocycle and horosphere budget row, combining the displayed* HyperbolicHorocycle$^{\uparrow}$ *and* HyperbolicHorosphere$^{\uparrow}$ *boundary-level reads.*
6. *S is the classifier-stability row, requiring the same disk, metric, Busemann-boundary, transport, and budget rows before a boundary verdict is public.*
7. *R is the refusal row excluding interior far-end fixed points, quotient ideal-boundary equality, selected rays, and detached observer states.*
8. *H records componentwise* hsame *transport for $D, M, U, Y, O, S, R, C, P, N$.*
9. *C records displayed* Cont *replay from disk readout through metric read, Busemann boundary, boundary-fixed transport, budget, classifier stability, refusal, provenance, and local naming.*

10. *P is the* Pkg *provenance over* HyperbolicBusemannBoundaryBudget$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBou HyperbolicHorocycle$^{\uparrow}$, HyperbolicHorosphere$^{\uparrow}$, *boundary-fixed transport,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicBusemannBoundaryBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host hyperbolic surface, quotient ideal-boundary equality, selected geodesic rays, a physical observer, or ambient completeness.

Theorem 9144.2 (Hyperbolic Busemann-boundary budget NameCert obligations)**.** *For every accepted* HyperbolicBusemannBoundaryBudget$^{\uparrow}$ *carrier*

$$B = (D, M, U, Y, O, S, R, H, C, P, N),$$

the local NameCert$_{\text{HyperbolicBusemannBoundaryBudget}^{\uparrow}}$ *surface consists exactly of disk-map readout through D, hyperbolic metric readback through M, Busemann-boundary exposure through U, boundary-fixed transport through Y, horocycle and horosphere budget exposure through O, classifier stability through S, refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9144.1. The metric and boundary rows M, U are not public until the disk-map row D is displayed, and the budget row O is not public until the Busemann-boundary and boundary-fixed transport rows are visible.

The classifier-stability row S reads the same disk, metric, boundary, transport, and budget rows before it exports a boundary verdict. The refusal row R then records the nonescape constraints: no interior far-end fixed point, quotient ideal-boundary equality, selected ray, or detached observer state can be projected from the packet.

The structural rows preserve the same route. Transport through H, replay through C, provenance through P, and local naming through N expose the finite NameCert surface and no additional hyperbolic host object. □

Theorem 9144.3 (Hyperbolic Busemann-boundary budget TasteGate requirements)**.** *The Lean follow-up for* HyperbolicBusemannBoundaryBudget$^{\uparrow}$ *must provide* Nontrivial HyperbolicBusemannBoundaryBudget$^{\uparrow}$ *and* FieldFaithful HyperbolicBusemannBoundaryBudget$^{\uparrow}$ *instances for the same carrier*

$$B = (D, M, U, Y, O, S, R, H, C, P, N).$$

Nontriviality must distinguish a packet with the displayed boundary budget from a packet with a missing disk, metric, Busemann-boundary, transport, budget, classifier-stability, or refusal row. Field faithfulness must preserve the row order $D, M, U, Y, O, S, R, H, C, P, N$ without replacing it by a host hyperbolic model or quotient boundary.

Proof. Read the requirement from Definition 9144.1. The packet is nontrivial only when the disk-map, metric, Busemann-boundary, boundary-fixed transport, horocycle and horosphere budget, classifier-stability, refusal, transport, replay, provenance, and local naming rows are all present.

Field faithfulness is the same row-preservation demand. Any classifier that identifies two packets while changing one of $D, M, U, Y, O, S, R, H, C, P, N$ would erase the finite boundary-budget route and would no longer be a faithful TasteGate for the chapter.

The excluded host objects are absent from the carrier and from the NameCert obligations of Definition 9144.2. The Lean follow-up therefore has to certify the finite carrier, not a global hyperbolic surface or quotient ideal boundary. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}$ boundary-budget read exposes disk-map, hyperbolic metric, Busemann-boundary, boundary-fixed transport, horocycle and horosphere budget, classifier-stability, refusal, transport, replay, provenance, and local naming rows before any observer-expansion boundary verdict is public.

Independence witness: hyperbolic-metric, hyperbolic-busemann-boundary, poincare-disk-boundary-transport, hyperbolic-horocycle, hyperbolic-horosphere: $\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds disk readout, metric readback, Busemann-boundary exposure, boundary-fixed transport, horocycle and horosphere budgets, classifier stability, refusal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}$ packet is generated from disk-map readout, hyperbolic metric readback, Busemann-boundary exposure, boundary-fixed transport, horocycle and horosphere budget rows, classifier stability, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9144.1, Definition 9144.2, and Definition 9144.3 bind the finite boundary-budget carrier, NameCert surface, and TasteGate follow-up requirements.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundaryBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, a global hyperbolic model, quotient ideal-boundary equality, selected geodesic rays, physical observers, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, disk-map readout, metric readback, Busemann-boundary exposure, boundary-fixed transport, horocycle and horosphere budget exactness, classifier stability, refusal, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.HyperbolicBusemannBoundaryBudgetUp.

Chapter 9145

A Concrete Naming Certificate for McNaughtonPiecewiseLinear$^{\uparrow}$

Following McNaughton and standard finite MV-algebra representation theory, the McNaughtonPiecewiseLinear$^{\uparrow}$ packet records a finite rational polyhedral presentation of an integer-slope piecewise-linear map before any real-valued or algebraic consumer reads it. It sits beside the finite interval and partition routes of chapter 9133 and chapter 9115, the rational ledger of chapter 291, the finite stream window of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names finite-cell McNaughton data only; it does not assert an ambient real-function extensionality theorem, an infinite subdivision search, a hidden fan, or quotient equality of completed functions.

Definition 9145.1 (McNaughton piecewise-linear carrier)**.** *A* McNaughtonPiecewiseLinear$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (C, A, R, D, S, Q, E, H, T, P, N)$$

with the following displayed rows:

1. *C is a finite rational polyhedral cell list, with each cell described by finitely many rational half-space inequalities and with overlaps recorded only through the displayed shared-refinement ledger.*
2. *A assigns to each cell an integer-slope affine expression with rational intercept data.*
3. *R is the shared refinement ledger, recording the common subdivision on which adjacent cell expressions agree on the displayed boundary faces.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *and* Rat$^{\uparrow}$ *coefficient readback row for rational vertices, half-space constants, slopes, intercepts, and boundary comparisons.*
5. *S is the* StreamName$^{\uparrow}$ *finite inspection row for requested cells, faces, and coefficient windows.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the same windows.*
7. *E is the* Real$^{\uparrow}$ *seal row reached only after C, A, R, D, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for cell lists, affine pieces, shared-refinement rows, rational readbacks, stream windows, regular rational tails, and the real seal.*
9. *T records displayed* Cont *replay from the finite cell table through rational readback and regular-tail rows to the real-facing seal.*

10. *P is the* Pkg *provenance over* McNaughtonPiecewiseLinear$^\uparrow$, DyadicRatCore$^\uparrow$, Rat$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{McNaughtonPiecewiseLinear}^\uparrow}$ *row.*

The carrier has no coordinate for an infinite partition, hidden real-function equality, externally chosen refinement, quotient completed-function equality, non-rational coefficient oracle, or ambient Real$^\uparrow$ *completeness.*

Theorem 9145.2 (McNaughton piecewise-linear NameCert obligations)**.** *For every accepted* McNaughtonPiecewise *carrier $M = (C, A, R, D, S, Q, E, H, T, P, N)$, the local* NameCert$_{\text{McNaughtonPiecewiseLinear}^\uparrow}$ *surface consists exactly of finite cell admission, integer-slope affine-piece admission, shared-refinement agreement, rational coefficient readback, finite-window exposure, regular rational tail readback, real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 9145.1. The rows C, A, R are the finite McNaughton table: cells, affine expressions, and the common refinement on which boundary agreement is witnessed.

Next read D, S, Q, E. Rational coefficient data is exposed through DyadicRatCore$^\uparrow$ and Rat$^\uparrow$, finite requests pass through StreamName$^\uparrow$, regular rational readback is supplied by RegSeqRat$^\uparrow$, and the Real$^\uparrow$ seal is available only after those rows have been displayed.

Finally inspect H, T, P, N. Transport, replay, provenance, and local naming preserve the same finite row inventory. Since no coordinate remains, an accepted certificate cannot add an infinite subdivision, a hidden real-function equality, or an ambient completeness claim. □

Theorem 9145.3 (McNaughton finite-cell handoff)**.** *Every* Real$^\uparrow$*-facing or* Rat$^\uparrow$*-facing consumer of an accepted* McNaughtonPiecewiseLinear$^\uparrow$ *carrier must factor through the finite route*

$$C, A, R \longmapsto \mathsf{DyadicRatCore}^\uparrow, \mathsf{Rat}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

using the displayed rows D, S, Q, E, H, T, P, N. The handoff exports the finite-cell piecewise-linear table, not host extensional equality of real functions, an infinite refinement procedure, quotient completed-function equality, or a hidden coefficient oracle.

Proof. Begin with the finite cell and affine-piece rows. The shared-refinement ledger R is part of the same packet as C and A, so a consumer cannot read an affine expression independently of the cell subdivision on which it is certified.

The rational readback row D then exposes vertices, inequalities, slopes, intercepts, and boundary comparisons through DyadicRatCore$^\uparrow$ and Rat$^\uparrow$. The window row S and regular readback row Q make the same finite data available to stream and regular-rational consumers before the terminal real seal E is reached.

The structural rows H, T, P, N replay and name exactly this route. Therefore every accepted handoff is finite-cell data transported through rational and regular readback rows, with no additional real-function extensionality, infinite search, or hidden refinement surface. □

The Lean follow-up is expected under `BEDC.Derived.McNaughtonPiecewiseLinearUp`; it must provide explicit finite-cell carrier admission, shared-refinement agreement, and nontrivial field-faithful TasteGate rows before any checked marker is cited.

Closure status for NameCert$_{\text{McNaughtonPiecewiseLinear}^\uparrow}$.

Constructive story: A McNaughtonPiecewiseLinear$^\uparrow$ packet is generated from finite rational polyhedral cells, integer-slope affine pieces, shared-refinement agreement, DyadicRatCore and Rat readback, StreamName windows, RegSeqRat tails, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{McNaughtonPiecewiseLinear}^\uparrow}$, seed)

Scope closed: Definition 9145.1, Definition 9145.2, and Definition 9145.3 seed the finite-cell McNaughton packet over rational polyhedral cells, integer-slope affine pieces, shared refinement, rational readback, stream windows, regular rational tails, real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{McNaughtonPiecewiseLinear}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient real-function extensionality, infinite subdivision search, hidden fan construction, quotient completed-function equality, non-rational coefficient oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-cell admission, integer-slope affine-piece admission, shared-refinement agreement, rational coefficient readback, finite-window exposure, regular rational tail readback, real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 9146

A Concrete Naming Certificate for LocatedCauchyModulusExtraction$^{\uparrow}$

Following Bishop and Bridges on located real numbers and constructive Cauchy data, the LocatedCauchyModulusExt packet records the finite route by which a displayed located regular Cauchy name exposes an explicit modulus row before any Real or Completion consumer reads the terminal seal. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located-real comparison surface of chapter 1933. The packet names modulus extraction as displayed NameCert data; it does not choose a quotient representative, invoke countable choice, or appeal to ambient Real$^{\uparrow}$ completeness.

Definition 9146.1 (Located Cauchy modulus extraction carrier)**.** *A* LocatedCauchyModulusExtraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, W, D, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the located Cauchy source row, recording the located interval and comparison data visible to the extractor.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row through which the source is inspected.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger controlling the requested precision.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same finite windows and dyadic tolerances.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after L, W, D, R have been displayed.*
6. *Q is the extracted modulus row, recording the finite stage bound certified for the displayed tolerance request.*
7. *H records componentwise* hsame *transport for the located source, finite window, dyadic tolerance, regular readback, Real seal, modulus row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$L, W, D, R \longmapsto Q \longmapsto E.$$

9. *P is the* Pkg *provenance over* LocatedCauchyModulusExtraction$^{\uparrow}$*,* LocatedReal$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, W, D, R, E, Q, H, C, P, N$. It has no coordinate for a quotient representative, a countable-choice selector, a hidden limit object, host equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9146.2 (Located Cauchy modulus extraction NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (L, W, D, R, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}}$ *surface consists exactly of located-source exposure through L, StreamName finite-window exposure through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real sealing through E, extracted-modulus exposure through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9146.1. The rows L, W, D, R expose the located source, finite stream window, dyadic tolerance, and regular rational readback before the modulus row can be inspected.

The modulus row Q is therefore tied to the displayed tolerance request. It is not a selected global schedule and it is not a quotient representative; it is the finite stage bound named by this packet for the visible located Cauchy source.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since no countable-choice selector, hidden host limit, host equality of real names, or ambient completeness theorem appears as a coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 9146.3 (Located Cauchy modulus soundness)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}$ *packet reads the extracted modulus only through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto Q \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the exported modulus is sound only relative to the displayed located source, stream window, dyadic tolerance, regular readback, and terminal seal. It supplies no representative choice, quotient equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 9146.2. The located source row L, window row W, tolerance row D, and readback row R are exposed before the extracted modulus row Q.

The terminal seal E is downstream of that same modulus row. A consumer may therefore use the modulus only as part of the displayed L, W, D, R, Q, E packet.

The structural rows H, C, P, N preserve the route under transport, replay it as a displayed continuation, record its provenance, and name it locally. No coordinate remains from which a consumer could recover a detached representative choice, quotient equality, countable-choice schedule, or ambient completeness assertion. □

The sibling route is explicit. $\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}$ reads dyadic tolerances through $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291, finite windows through $\mathsf{StreamName}^{\uparrow}$ in chapter 126, regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, terminal sealing through $\mathsf{Real}^{\uparrow}$ in chapter 38, and located comparison through $\mathsf{LocatedReal}^{\uparrow}$ in chapter 1933.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A LocatedCauchyModulusExtraction$^{\uparrow}$ packet is generated from a located Cauchy source, StreamName finite windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, an extracted modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCauchyModulusExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 9146.1, Definition 9146.2, and Definition 9146.3 seed the located Cauchy modulus extraction route over the DyadicRatCore, StreamName, RegSeqRat, Real, and LocatedReal spine.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyModulusExtraction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient representatives, countable choice, selected limits, host equality of real names, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked located-source admission, finite-window readback, dyadic tolerance, regular readback, Real sealing, modulus exposure, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyMod

Chapter 9147

A Concrete Naming Certificate for CauchyNameModulusExtraction$^{\uparrow}$

Following Bishop and Bridges on regular Cauchy representatives and diagonal limits, the CauchyNameModulusExtrac packet records the finite modulus-extraction route behind the regular-Cauchy diagonal-limit construction of chapter 9238. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite separation boundary of chapter 1202. The packet names the finite extraction object used by a Cauchy name; it does not select a limit, quotient Cauchy names, invoke countable choice, or assert ambient Real$^{\uparrow}$ completeness.

Theorem 9147.1 (Cauchy name modulus extraction carrier)**.** *A* CauchyNameModulusExtraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, W, D, R, E, F, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular-Cauchy name source row, recording the sequence of regular rational names whose diagonal route is being inspected.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row selecting the visible indices used by the diagonal readback.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested precision and comparison budget.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same finite window and dyadic tolerance data.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row that receives the extracted Cauchy-name data.*
6. *F is the* FiniteWindowRealSeparation$^{\uparrow}$ *row that records the finite separation boundary used before a Real-facing equality or apartness read.*
7. *M is the extracted modulus row, naming the diagonal stage bound visible to the Cauchy-name consumer.*
8. *H records componentwise* hsame *transport for source, window, tolerance, readback, Real seal, finite separation, modulus, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S, W, D, R, F \longmapsto M \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyNameModulusExtraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNameModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, R, E, F, M, H, C, P, N$. It has no coordinate for a selected limit, quotient equality of Cauchy names, countable choice, host equality of Real representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9147.2 (Cauchy name modulus extraction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNameModulusExtraction}^{\uparrow}$ *carrier*

$$X = (S, W, D, R, E, F, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyNameModulusExtraction}^{\uparrow}}$ *surface consists exactly of regular-Cauchy source exposure through S, StreamName finite-window exposure through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real sealing through E, finite separation exposure through F, modulus exposure through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9147.1. The rows S, W, D, R, F expose the regular-Cauchy source, finite window, dyadic tolerance, regular readback, and finite separation boundary before any modulus row is exported.

The extracted modulus row M is tied to that displayed diagonal window. It is not a chosen limit, a quotient representative, or a countable-choice schedule; it is the finite stage bound named for the inspected Cauchy-name route.

The terminal seal E and structural rows H, C, P, N preserve the same route for Real-facing consumers. Since no selected limit, quotient Cauchy-name equality, host real equality, or ambient completeness principle appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 9147.3 (Cauchy name diagonal modulus route)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{CauchyNameModulusExtraction}^{\uparrow}$ *packet factors its read through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{FiniteWindowRealSeparation}^{\uparrow} \longmapsto M \longmapsto \mathsf{Real}^{\uparrow}.$$

The route realizes the regular-Cauchy diagonal-limit modulus extraction as finite NameCert data, without quotient representatives, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 9147.2. It places the source, window, tolerance, readback, and finite separation rows before the extracted modulus row M.

The Real seal E is downstream of the same row. A diagonal-limit consumer therefore receives a finite Cauchy-name extraction packet, not a detached limit object.

Finally H, C, P, N transport, replay, source, and name the route. These rows keep the exported modulus inside the displayed BEDC surface and leave no coordinate for quotient equality, countable choice, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNameModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNameModulusExtraction}^{\uparrow}$ packet is generated from a regular-Cauchy name source, StreamName finite windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a finite separation boundary, an extracted modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNameModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9147.1, Definition 9147.2, and Definition 9147.3 seed the regular-Cauchy diagonal-limit modulus route over the DyadicRatCore, StreamName, RegSeqRat, Real, and FiniteWindowRealSeparation spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNameModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy names, countable choice, host equality of Real representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, finite-window readback, dyadic tolerance, regular readback, finite separation, Real sealing, modulus exposure, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNameModulusExtractionUp.

Chapter 9148

A Concrete Naming Certificate for $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$ packet records the L10 finite-window compactness route by which a bounded regular Cauchy source is inspected through finite nets before a Real or Completion consumer asks for a cluster-facing read. It sits beside the bounded source of chapter 1697, the bounded-subsequence route of chapter 1698, the Cauchy sequence space of chapter 406, the stream-window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a sequential finite-window total-boundedness carrier; it is not open-cover compactness, a selected subsequence, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9148.1 (SequentiallyTotallyBounded carrier)**.** *A* $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, Q, X, W, D, E, F, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *source row;*
2. *Q is the* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *downstream route available only as displayed bounded-subsequence provenance;*
3. *X is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *ambient sequence-space row;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule inspected at the requested precision;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite window;*
6. *E is the finite net row covering the inspected window at the displayed dyadic tolerance;*
7. *F is the sequential finite-window classifier saying that every inspected prefix element is assigned to one displayed net cell;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row reached only after B, X, W, D, E, F have been displayed;*
9. *H records componentwise* hsame *transport over the source, sequence-space, finite-window, dyadic, net, classifier, readback, provenance, and local naming rows;*
10. *C records displayed* Cont *replay for the same finite-window route;*

11. *P is the* Pkg *provenance over* $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$, $\mathsf{BoundedRegularCauchy}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubs}$ $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentiallyTotallyBounded}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open-cover compactness proof, a selected infinite subsequence, countable choice, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9148.2 (SequentiallyTotallyBounded NameCert obligations)**.** *For every accepted* SequentiallyTotallyBou *carrier $S = (B, Q, X, W, D, E, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentiallyTotallyBounded}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded regular Cauchy source exposure, bounded-subsequence provenance, Cauchy sequence-space exposure, finite stream-window scheduling, dyadic tolerance exactness, finite-net assignment, sequential finite-window classifier stability, regular rational and Real readback non-escape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 9148.1. The accepted packet exposes only the rows $B, Q, X, W, D, E, F, R, H, C, P, N$.

The finite-window route reads the bounded source B, the sequence-space row X, the stream schedule W, and the dyadic ledger D before the finite net E and classifier F are available. Thus total boundedness is inspected through displayed finite windows rather than through an open-cover or host compactness principle.

Finally R, H, C, P, N preserve the same route for regular rational readback, Real sealing, transport, replay, provenance, and local naming. Since selected subsequences, countable choice, quotient stream equality, host real equality, and ambient completeness are absent from the carrier, no accepted consumer can project them from the certificate surface. □

Theorem 9148.3 (SequentiallyTotallyBounded L10 finite-window compactness route)**.** *Every L10 compactness-facing consumer of an accepted* $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$ *packet factors through*

$\mathsf{BoundedRegularCauchy}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, DyadicRatC

The consumer first reads bounded regular Cauchy data, bounded-subsequence provenance, and the Cauchy sequence-space row; then it inspects a finite stream window and dyadic tolerance ledger; then it reads the finite net classifier; only then may it reach the regular rational and Real seal rows. The route does not claim open-cover compactness, selected subsequences, countable choice, quotient stream equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 9148.2. They require the bounded source, bounded-subsequence provenance, sequence-space row, stream schedule, dyadic ledger, finite net, and classifier to be displayed before readback.

Use the carrier ordering from Definition 9148.1. The readback row R is available only after B, X, W, D, E, F, and the bounded-subsequence row Q is provenance rather than a selected infinite subsequence.

The structural rows H, C, P, N transport, replay, source, and name the same finite-window packet. Because the forbidden compactness and choice coordinates are absent, the exported L10 route is a finite-window total-boundedness certificate, not a classical compactness or completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{Sequentially TotallyBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyTotallyBounded}^{\uparrow}$ packet is generated from a BoundedRegularCauchy source, bounded-subsequence provenance, a CauchySequenceSpace row, finite StreamName windows, a DyadicRatCore tolerance ledger, finite net rows, a sequential finite-window classifier, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentiallyTotallyBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9148.1, Definition 9148.2, and Definition 9148.3 seed the L10 finite-window compactness route over bounded regular Cauchy, bounded-subsequence, CauchySequenceSpace, StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentiallyTotallyBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subsequences, countable choice, quotient stream equality, host real equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window scheduling, dyadic tolerance exactness, finite-net classifier stability, bounded-subsequence provenance, readback non-escape, transport, replay, and local naming for BEDC.Derived.SequentiallyTotallyBoundedUp.

Chapter 9149

A Concrete Naming Certificate for InnovationSolderingMetricInterface$^{\uparrow}$

The InnovationSolderingMetricInterface$^{\uparrow}$ packet concretizes the soldering target of chapter 9423 as finite BEDC data. It names the interface where a ledger innovation row determines a soldering one-form, and the soldering one-form determines the metric row. The packet sits beside LogarithmicSectionDynamics$^{\uparrow}$ in chapter 9135, SolenoidPhase$^{\uparrow}$ in chapter 9075, ObserverCenteredConstructorCertificat in chapter 9114, and FiniteObservationBoundary$^{\uparrow}$ in chapter 9102. Those siblings provide section dynamics, phase transport, observer-centered construction, or finite observation boundaries; this packet binds the metric-facing soldering interface itself.

Definition 9149.1 (InnovationSolderingMetricInterface carrier)**.** *An* InnovationSolderingMetricInterface$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, O, D, B, Q, \Theta, G, g, F, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the ledger-field row;*
2. *O is the displayed orthogonal innovation row;*
3. *D is the frame-reshaping connection row;*
4. *B is the finite section-bundle boundary row named by the source vision;*
5. *Q is the innovation projector row;*
6. *Θ is the soldering one-form row, read as QDB in finite carrier notation;*
7. *G is the displayed signature row on the effective innovation bundle;*
8. *g is the metric row read from $G(\Theta, \Theta)$;*
9. *F is the field-faithfulness row requiring changes in L, O, D, Q, Θ, G to be visible at the metric interface when the carrier admits such a change;*
10. *A is the audit row refusing product-only coexistence, independent metric insertion, Hilbert-norm-as-Lorentz-metric replacement, hidden rank assumptions, hidden signature assumptions, and unledgered curvature dynamics;*
11. *H records componentwise* hsame *transport over the ledger, innovation, connection, boundary, projector, soldering, signature, metric, faithfulness, audit, provenance, and local naming rows;*

12. *C records displayed* Cont *replay for the same interface;*
13. *P is the* Pkg *provenance over* InnovationSolderingMetricInterface$^{\uparrow}$, LogarithmicSectionDynamics$^{\uparrow}$, SolenoidPhase$^{\uparrow}$, ObserverCenteredConstructorCertificate$^{\uparrow}$, FiniteObservationBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
14. *N is the local* NameCert$_{\text{InnovationSolderingMetricInterface}^{\uparrow}}$ *row.*

The carrier requires Lean-side Nontrivial InnovationSolderingMetricInterface$^{\uparrow}$ *and* FieldFaithful InnovationSolderingMe *instances before any checked discovery or bridge claim may use the interface as a non-smoke metric row.*

Theorem 9149.2 (InnovationSolderingMetricInterface NameCert obligations)**.** *For every accepted* InnovationSolderingMetricInterface$^{\uparrow}$ *carrier $M = (L, O, D, B, Q, \Theta, G, g, F, A, H, C, P, N)$, the local* NameCert$_{\text{InnovationSolderingMetricInterface}^{\uparrow}}$ *surface consists exactly of carrier admission, ledger-field exposure, orthogonal innovation exposure, connection exposure, section-bundle boundary exposure, innovation-projector exactness, soldering one-form exactness, signature-row exposure, metric-row exactness, field-faithfulness visibility, audit-row exactness, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Carrier admission is projection from Definition 9149.1. The rows $L, O, D, B, Q, \Theta, G, g, F, A, H, C, P, N$ are the whole accepted interface.

The soldering route reads L, O, D, B, Q before Θ, and reads Θ and G before g. Thus the metric row is not inserted independently and the positive Hilbert norm is not treated as a Lorentz metric.

Finally the field-faithfulness row and audit row constrain the export. The structural rows H, C, P, N transport, replay, source, and name the same finite interface, while the required Lean Nontrivial and FieldFaithful instances are the formal hooks that prevent the carrier from becoming a vacuous metric label. □

Theorem 9149.3 (InnovationSolderingMetricInterface metric dependency)**.** *Every metric-facing consumer of an accepted* InnovationSolderingMetricInterface$^{\uparrow}$ *packet must read the metric row through the displayed dependency chain*

$$L, O, D, B, Q, \Theta, G, g.$$

The carrier may be consumed as an interface only when the Nontrivial and FieldFaithful instance surface for InnovationSolderingMetricInterface$^{\uparrow}$ *is available on the Lean side; the paper route itself does not claim Einstein dynamics, empirical matter coupling, quantum gravity, unledgered curvature equations, or a metric inserted independently of the ledger innovation.*

Proof. Project the carrier and the NameCert obligations. They order the ledger, innovation, connection, boundary, projector, soldering, signature, and metric rows before any public metric read.

Use the audit row A. It refuses the product-only route, independent metric insertion, Hilbert-norm replacement, hidden rank data, hidden signature data, and unledgered curvature dynamics.

The sibling provenance row P locates the section-dynamics, solenoid-phase, observer-centered construction, and finite-observation boundary packets without merging their carriers into the metric interface. Thus the exported metric dependency is exactly the displayed soldering route, with Nontrivial and FieldFaithful Lean instances required for checked non-vacuity and field sensitivity. □

Falsifiable prediction: Every accepted InnovationSolderingMetricInterface$^{\uparrow}$ consumer that reaches a metric row while omitting one of L, O, D, B, Q, Θ, G refutes the packet.

Independence witness: logarithmicsectiondynamics, solenoidphase, observercenteredconstructorcertificate, finiteobservationboundary: none is bijective with $\mathsf{InnovationSolderingMetricInterface}^{\uparrow}$ because each omits the full ledger-innovation-projector-soldering-signature-metric route or the field-faithfulness audit row.

Closure status for $\mathsf{NameCert}_{\mathsf{InnovationSolderingMetricInterface}^{\uparrow}}$.

Constructive story: An $\mathsf{InnovationSolderingMetricInterface}^{\uparrow}$ packet is generated from a ledger-field row, orthogonal innovation row, frame-reshaping connection row, section-bundle boundary row, innovation projector, soldering one-form, signature row, metric row, field-faithfulness row, audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InnovationSolderingMetricInterface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9149.1, Definition 9149.2, and Definition 9149.3 seed the finite soldering metric interface over ledger innovation, projector, soldering, signature, metric, audit, and field-faithfulness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InnovationSolderingMetricInterface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Einstein dynamics, empirical matter coupling, quantum gravity, unledgered curvature equations, checked Nontrivial or FieldFaithful Lean instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, soldering one-form exactness, signature-row exactness, metric-row dependency, Nontrivial and FieldFaithful instances, audit exactness, transport, replay, and local naming for BEDC.Derived.InnovationSolderingMetricInterfaceUp.

Chapter 9150

A Concrete Naming Certificate for Window6 Forward Deliverable Shapes

The Window6 forward-math package realizes six Fibonacci-cube proof-object shapes as finite NameCert data: a closed-zero count identity, a seam-gluing law, a sharp saturation-deficit inequality, a live-tail congruence with exact finite-state memory, a signed involution with a fixed block, and a zero-mark Galois closure. Each shape is recorded here as a bounded Window6 surface over Boolean words, local six-letter windows, and the no-adjacent-1 Fibonacci-cube constraint.

9150.1 Closed-Zero Flattening Count Identity

Theorem 9150.1 (Closed-zero interior flattening)**.** *For Window6 Fibonacci-cube words, a closed-zero endpoint window is insulated from the no-adjacent interior between its two displayed zero guards. Replacing that interior by any other no-adjacent interior preserves the surrounding no-adjacent read. In the concrete enumerated windows this yields the checked count identity*

$$C_m = 8Z_m$$

for the displayed finite lengths $m = 6, \ldots, 12$.

Lean4 ✓ `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten bijection`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m6`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m7`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m8`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m9`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m10`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m11`

↪ variant `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6 closed zero flatten count identity m12`

Closure status for NameCert$_{Window6ClosedZeroFlattenBijection}$.

Theory closure: TheoryGate(NameCert$_{Window6ClosedZeroFlattenBijection}$, scopedClosed)

Scope closed: Definition 9150.1 closes the closed-zero insulation statement and records concrete count identities for lengths 6 through 12.

Formal verification: FormalStatus(NameCert$_{Window6ClosedZeroFlattenBijection}$, theoremChecked)

Lean target: `BEDC.Derived.Window6ClosedZeroFlattenBijection.window6\_closed\_zero\_flatten\_bijection`

Bridge status: none

Not claimed: This anchor does not claim a symbolic all-length formula beyond the checked bijection surface and the displayed finite count identities.

Upgrade path: A broader closure would package the finite count witnesses into a single symbolic recurrence or all-length enumeration theorem.

9150.2 Seam-Gluing Parity

Theorem 9150.2 (Window6 seam scanner gluing)**.** *Let x and y be Boolean words of length at least 5. The even-two-hit Window6 scanner is compositional under append: the scan of $x++y$ is the scan of x, plus the scan of y, plus the bounded seam contribution determined only by the suffix of x of length 5 and the prefix of y of length 5:*

$$\nu_2(x++y) = \nu_2(x) + \nu_2(y) + \mathrm{straddle}(\mathrm{suf}_5(x), \mathrm{pref}_5(y)).$$

The parity reading is the corresponding Boolean xor of the three summands.

Lean4 ✓ `BEDC.Derived.Window6SeamGluingParity.window6 seam scanner append`

↪ variant `BEDC.Derived.Window6SeamGluingParity.window6 seam scanner append parity`

Closure status for NameCert$_{Window6SeamGluingParity}$.

Theory closure: TheoryGate(NameCert$_{Window6SeamGluingParity}$, scopedClosed)

Scope closed: Definition 9150.2 closes append composition for the Window6 seam scanner and its parity readback under the length-at-least-5 hypotheses.

Formal verification: FormalStatus(NameCert$_{Window6SeamGluingParity}$, theoremChecked)

Lean target: `BEDC.Derived.Window6SeamGluingParity.window6\_seam\_scanner\_append`

Bridge status: none

Not claimed: This anchor does not claim an arbitrary stream transducer theorem, a cyclic seam theorem, or a scanner over non-Boolean alphabets.

Upgrade path: A broader closure would expose the seam scanner as a reusable finite transducer interface with checked associativity across multiple appends.

9150.3 Saturation-Deficit Extremality

Theorem 9150.3 (Sharp Window6 saturation-deficit bound)**.** *For every Boolean word w, the Window6 saturation deficit is bounded on both sides by*

$$0 \le \mathrm{deficit}_6(w) \le 3N_6(|w|).$$

The upper bound is attained by the all-zero witnesses, while alternating witnesses attain the lower bound with deficit 0.

Lean4 ✓ `BEDC.Derived.Window6SaturationDeficitExtremal.window6 saturation deficit upper`

↪ variant `BEDC.Derived.Window6SaturationDeficitExtremal.window6 saturation deficit lower`

↪ variant `BEDC.Derived.Window6SaturationDeficitExtremal.window6 deficit allzeros`

↪ variant `BEDC.Derived.Window6SaturationDeficitExtremal.window6 deficit alt`

Closure status for NameCert$_{Window6SaturationDeficitExtremal}$.

Theory closure: TheoryGate(NameCert$_{Window6SaturationDeficitExtremal}$, scopedClosed)

Scope closed: Definition 9150.3 closes the two-sided Window6 deficit inequality and the all-zero and alternating extremal witnesses.

Formal verification: FormalStatus(NameCert$_{Window6SaturationDeficitExtremal}$, theoremChecked)

Lean target: `BEDC.Derived.Window6SaturationDeficitExtremal.window6\_saturation\_deficit\_upper`

Bridge status: none

Not claimed: This anchor does not claim a classification of every extremizing word or a higher-window analogue.

Upgrade path: A broader closure would add a checked extremizer classification and a parameterized window-size family.

9150.4 Live-Tail Congruence

Theorem 9150.4 (Five-tail right congruence and exact memory)**.** *For a prefix p of length at least 5, every real suffix-window read after appending a continuation s factors through the last five bits of p:*

$$\mathrm{realSuffixWindows}(p, s) = \mathrm{windows}_6(\mathrm{tail}_5(p) + +s).$$

Consequently two such prefixes with the same five-tail are right-congruent for all continuations. The checked finite memory surface lists the 13 no-adjacent five-tails and proves length-five minimality by separating unequal tails with a one-bit continuation.

Lean4 ✓ `BEDC.Derived.Window6LiveTailCongruence.window6 live tail congruence`

↪ variant `BEDC.Derived.Window6LiveTailCongruence.window6 live tail factor`

↪ variant `BEDC.Derived.Window6LiveTailCongruence.T5 length`

↪ variant `BEDC.Derived.Window6LiveTailCongruence.window6 live tail minimality length five`

Closure status for NameCert$_{Window6LiveTailCongruence}$.

Theory closure: TheoryGate(NameCert$_{Window6LiveTailCongruence}$, scopedClosed)

Scope closed: Definition 9150.4 closes the five-tail factorization, right congruence, 13-state no-adjacent memory count, and length-five separation witness.

Formal verification: FormalStatus(NameCert$_{Window6LiveTailCongruence}$, theoremChecked)

Lean target: `BEDC.Derived.Window6LiveTailCongruence.window6\_live\_tail\_congruence`

Bridge status: none

Not claimed: This anchor does not claim a minimized automaton object with transition table equivalence beyond the checked tail congruence and separation surface.

Upgrade path: A broader closure would package the thirteen tails as an explicit automaton with checked transition totality and minimization.

9150.5 Signed Involution Fixed Point

Theorem 9150.5 (Window6 signed involution with fixed block)**.** *On admissible Window6 blocks, the checked map flip_6 is an involution, preserves the Fibonacci no-adjacent constraint, reverses parity away from its fixed block, and has the unique fixed block 010010. The induced guarded counts satisfy the finite Franklin-type identities*

$$|G_{\mathrm{ev}}(m)| = |G_{\mathrm{odd}}(m)| + |H(m)|$$

for the checked lengths $m = 6, \ldots, 12$.

Lean4 ✓ `BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6 involutive`

↪ variant `BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6 parity flip`

↪ variant `BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6 fixed 010010`

↪ variant `BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6 fixed unique`

↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6 preserves fib
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 6
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 7
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 8
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 9
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 10
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 11
↪ variant BEDC.Derived.Window6SignedInvolutionFixedPoint.guarded count identity 12

Closure status for NameCert$_{Window6SignedInvolutionFixedPoint}$.

Theory closure: TheoryGate(NameCert$_{Window6SignedInvolutionFixedPoint}$, scopedClosed)

Scope closed: Definition 9150.5 closes involutivity, Fibonacci preservation, parity reversal away from 010010, fixed-point uniqueness, and finite guarded count identities for lengths 6 through 12.

Formal verification: FormalStatus(NameCert$_{Window6SignedInvolutionFixedPoint}$, theoremChecked)

Lean target: `BEDC.Derived.Window6SignedInvolutionFixedPoint.flip6\_involutive`

Bridge status: none

Not claimed: This anchor does not claim a global infinite Franklin involution or symbolic all-length guarded count formula.

Upgrade path: A broader closure would bind the local involution to a single all-length signed enumeration theorem.

9150.6 Zero-Mark Galois Closure

Theorem 9150.6 (Window6 zero-mark Galois closure)**.** *The Window6 zero-mark operators form a finite Galois connection: dilation is left adjoint to erosion under the displayed implication order, with the required five-cell padding equation*

$$\mathrm{dilate}_6(S) \le P \quad \Longleftrightarrow \quad S \le \mathrm{erode}_6(P).$$

The induced operator Cl *is extensive, monotone, and idempotent. Its finite closed reads are generated by unions of marked intervals whose separating gaps are at least six positions.*

Lean4 ✓ `BEDC.Derived.Window6ZeroMarkGaloisClosure.window6 galois adjunction`
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.window6 galois adjunction forward
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.window6 galois adjunction backward
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.cl extensive
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.cl monotone
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.cl idempotent
↪ variant BEDC.Derived.Window6ZeroMarkGaloisClosure.erode6 image closed window

Closure status for NameCert$_{Window6ZeroMarkGaloisClosure}$.

Theory closure: TheoryGate(NameCert$_{Window6ZeroMarkGaloisClosure}$, scopedClosed)

Scope closed: Definition 9150.6 closes the Window6 dilation–erosion adjunction, closure extensivity, monotonicity, idempotence, and the checked finite closed-window image surface.

Formal verification: FormalStatus(NameCert$_{Window6ZeroMarkGaloisClosure}$, theoremChecked)

Lean target: `BEDC.Derived.Window6ZeroMarkGaloisClosure.window6\_galois\_adjunction`

Bridge status: none

Not claimed: This anchor does not claim a topological closure space over an infinite carrier or a complete symbolic classification of every closed finite set outside the checked Window6 mark interface.

Upgrade path: A broader closure would expose fixed-point interval-union normal forms as a single checked classifier theorem.

Chapter 9151

A Finite Window6 Skew/Neutral Certificate for the RH No-Neutral-Leak Obligation

The zeta-to-Zeckendorf residual route sends an off-mirror zeta zero event into a Zeckendorf residual tower over the Fibonacci-cube projections $X_m = \{0,1\}^m_{\text{no}\,11}$, with residual carrier $V_m = \mathbb{Q}[X_m]$. One of the load-bearing obligations of that route is a no-neutral-leak condition: the residual must live in the skew sector of a return operator R_m, with $1 \notin \operatorname{spec}(R_m|_{\text{skew}})$, since the attractive computation $R_6 r = -\frac{1}{2} r$ eliminates a residual only after its neutral 1-eigenmode has been excluded. This chapter records a finite Window6-level certificate for the shape of that obligation: an explicit mirror involution whose fixed set is the neutral support and whose free orbits are the skew sector, so that the skew/neutral split of X_6 is exhibited as a checked ± 1-eigenspace decomposition. This is a finite residual-surface certificate, not a proof of RH.

9151.1 The Window6 Mirror Involution

Let X_6 be the Fibonacci-cube surface of six-letter Boolean words with no adjacent 1. The mirror is the letter reversal

$$\mu(b_0 b_1 b_2 b_3 b_4 b_5) = b_5 b_4 b_3 b_2 b_1 b_0.$$

Theorem 9151.1 (Mirror is a no-adjacent involution)**.** *The reversal μ is an involution, $\mu \circ \mu = \mathrm{id}$, and it preserves the no-adjacent-1 constraint: if a six-letter word has no adjacent 1, then so does its reversal. Hence μ acts on X_6.*

```
Lean4 ✓  BEDC.Derived.Window6MirrorSkewNeutralSplit.mirror6 involutive
  ↪ variant  BEDC.Derived.Window6MirrorSkewNeutralSplit.mirror6 preserves noAdj
```

9151.2 Fixed Points Are the Neutral Support

Theorem 9151.2 (Fixed iff palindrome)**.** *A word of X_6 is fixed by μ if and only if it is a palindrome. On the no-adjacent surface the palindromes are exactly*

$$000000, \qquad 010010, \qquad 100001,$$

so the mirror-fixed set has size 3*. These fixed words are the finite neutral support: they are the words the mirror cannot move, the Window6 analogue of the* 1*-eigenmode that the no-neutral-leak obligation must exclude from the skew sector.*

Lean4 ✓ `BEDC.Derived.Window6MirrorSkewNeutralSplit.mirror6 fixed iff palindrome`

9151.3 The Skew/Neutral Split Certificate

Every non-palindrome pairs with its distinct reversal, so the non-fixed words of X_6 partition into free two-element mirror orbits, the finite skew sector. Choosing the lexicographically smaller representative of each orbit counts those orbits.

Theorem 9151.3 (Finite skew/neutral decomposition)**.** *The Fibonacci-cube surface X_6 decomposes under the mirror into the neutral fixed set and the free skew orbits:*

$$|X_6| = |X_6^{\text{neutral}}| + 2\,|X_6^{\text{skew}}|, \qquad 21 = 3 + 2 \cdot 9.$$

The neutral part has dimension 3*; the skew part has* 9 *free mirror orbits, spanning the remaining* 18 *words. This is the checked* ± 1*-eigenspace split of the mirror on the Window6 residual surface.*

Lean4 ✓ `BEDC.Derived.Window6MirrorSkewNeutralSplit.window6 mirror skew neutral split certificate`
↪ variant `BEDC.Derived.Window6MirrorSkewNeutralSplit.neutral count`
↪ variant `BEDC.Derived.Window6MirrorSkewNeutralSplit.skew pair count`

9151.4 Relation to the No-Neutral-Leak Obligation

The zeta-to-Zeckendorf route asks, at each layer, that a certified residual carry no component in the 1-eigenmode of the return operator, so that the $-\frac{1}{2}$ contraction can drive it to zero. Definition 9151.3 realizes the finite instance of that split explicitly: the mirror supplies a concrete involution on X_6 whose $+1$-eigenspace is spanned by the three palindromic neutral words and whose -1-eigenspace is spanned by the nine skew orbit differences. A residual placed in the skew span is, at this Window6 layer, provably free of the neutral mode, because the neutral mode is exactly the fixed-point span, which the skew span meets only in zero.

This is a finite residual-surface certificate. It does not prove RH, and it does not by itself discharge the all-layer spectral obligation $1 \notin \operatorname{spec}(R_m|_{\text{skew}})$ for every m; it exhibits the combinatorial shape of that obligation on the sixth Fibonacci-cube projection and gives an axiom-free finite witness of the skew/neutral separation there.

Closure status for NameCert$_{Window6MirrorSkewNeutralSplit}$.

Theory closure: TheoryGate(NameCert$_{Window6MirrorSkewNeutralSplit}$, scopedClosed)

Scope closed: Definition 9151.3 closes the finite Window6 mirror skew/neutral decomposition $21 = 3 + 2 \cdot 9$ with the mirror-fixed palindromes as the neutral support.

Formal verification: FormalStatus(NameCert$_{Window6MirrorSkewNeutralSplit}$, theoremChecked)

Lean target: `BEDC.Derived.Window6MirrorSkewNeutralSplit.window6\_mirror\_skew\_neutral\_split\_certificate`

Bridge status: none

Not claimed: This does not prove RH; it is a finite Window6-layer skew/neutral separation certificate only, and it does not discharge the all-layer spectral no-neutral-leak obligation.

Upgrade path: An RH-route upgrade would extend the finite mirror split to the all-layer Zeckendorf tower and connect the mirror-fixed neutral support to the 1-eigenmode of the return operator R_m on V_m.

Chapter 9152

The Fibonacci–Polya Jensen Kernel

Following Polya and Griffin–Ono–Rolen–Zagier, this chapter records a finite Fibonacci smoothing of the actual Riemann ξ Jensen criterion. Let

$$J^{d,n}(X) = \sum_{j=0}^{d} \binom{d}{j} \gamma_{n+j} X^j$$

be the Jensen polynomial of the Taylor coefficients of ξ. Polya's criterion places the Riemann Hypothesis inside the assertion that all such Jensen polynomials are hyperbolic, while Griffin–Ono–Rolen–Zagier prove unconditionally that, for each fixed d, $J^{d,n}$ is hyperbolic for all sufficiently large n.

For a Fibonacci cube of length m, set

$$W_m(T) = \sum_k \binom{m-k+1}{k} T^k.$$

Its coefficients count binary words of length m with no adjacent 1s and exactly k entries equal to 1. The real-rootedness of W_m places $W_m(D)$, with $D = d/dX$, among the finite Polya–Schur hyperbolicity preservers. Applied to the reciprocal Jensen polynomial

$$\widehat{J}^{d,n}(X) = X^d J^{d,n}(1/X) = \sum_{j=0}^{d} \binom{d}{j} \gamma_{n+j} X^{d-j},$$

the packet studies the finite window

$$F_{m,d,n}(X) = W_m(D) \widehat{J}^{d,n}(X).$$

Thus hyperbolicity of $J^{d,n}$ implies hyperbolicity of $F_{m,d,n}$, and the GORZ theorem gives an unconditional large-n hyperbolicity shadow for every fixed m, d.

9152.1 Degree-Two Discriminant Window

In degree two write

$$\widehat{J}^{2,n}(X) = aX^2 + 2bX + c,$$

where $a = \gamma_n$, $b = \gamma_{n+1}$, and $c = \gamma_{n+2}$. Since

$$W_m(D) = 1 + mD + \frac{(m-1)(m-2)}{2} D^2$$

on degree-two polynomials, the smoothed polynomial is

$$F_{m,2,n}(X) = aX^2 + (2am + 2b)X + (am^2 - 3am + 2a + 2bm + c).$$

Its discriminant is the finite identity

$$\operatorname{disc} F_{m,2,n} = 4\left((b^2 - ac) + (3m - 2)a^2\right).$$

Equivalently,

$$\operatorname{disc} F_{m,2,n} - 4(b^2 - ac) = 4(3m - 2)a^2.$$

For $m \geq 1$, the added Fibonacci-kernel term is nonnegative. The Fibonacci window therefore preserves and strengthens the degree-two Turan margin $b^2 - ac$; it does not replace the xi coefficient problem by a Fibonacci counting problem.

Theorem 9152.1 (Fibonacci–Polya degree-two Jensen certificate)**.** *The degree-two Fibonacci–Polya smoothing has the symbolic discriminant certificate*

$$\operatorname{disc} F_{m,2,n} = 4\left((\gamma_{n+1}^2 - \gamma_n\gamma_{n+2}) + (3m - 2)\gamma_n^2\right),$$

and the concrete $m = 5$ differential kernel satisfies

$$W_5(D)X^0 = 1, \quad W_5(D)X = X+5, \quad W_5(D)X^2 = X^2+10X+12, \quad W_5(D)X^3 = X^3+15X^2+36X+6.$$

Lean4 ✓ `BEDC.Derived.FibonacciPolyaJensenKernel.fibonacci polya jensen kernel certificate`

9152.2 Honest Scope

This is a hyperbolicity-preserving Fibonacci smoothing of the Jensen criterion, a genuine finite RH-shadow, not a sharper attack on RH, not a claim that ξ coefficients are Zeckendorf or Fibonacci counts, and not a proof of the Riemann Hypothesis. The Lean certificate treats $\gamma_n, \gamma_{n+1}, \gamma_{n+2}$ as symbolic coefficients. It proves the finite algebraic normal-form identity behind the displayed degree-two discriminant and checks the $W_5(D)$ monomial action; it does not assume numerical values of the Riemann ξ coefficients and does not discharge any analytic zero-location theorem.

The analytic bridge is external and classical: Polya identifies RH with total hyperbolicity of the Jensen family, and GORZ gives unconditional eventual hyperbolicity in each fixed degree. The BEDC content here is the finite kernel interface placed inside that classical criterion: applying a Fibonacci-cube Laguerre–Polya kernel to the actual ξ Jensen window adds a visible nonnegative degree-two margin term while preserving the honest distinction between finite hyperbolicity shadows and RH itself.

Falsifiable prediction: A recomputation of the symbolic degree-two smoothing that changes the coefficient tuple $(a, 2am + 2b, am^2 - 3am + 2a + 2bm + c)$, the discriminant term $4((b^2 - ac) + (3m - 2)a^2)$, or the $W_5(D)$ monomial actions listed in Definition 9152.1, refutes this packet.

Independence witness: riemann-hypothesis, zeckendorf-coefficient-models: this packet is a finite hyperbolicity-preserving smoothing of the actual Riemann ξ Jensen criterion, not a proof of RH and not a model identifying ξ coefficients with Fibonacci or Zeckendorf counts.

Closure status for $\mathsf{NameCert}_{FibonacciPolyaJensenKernel}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{FibonacciPolyaJensenKernel}$, scopedClosed)

Scope closed: degree-2 discriminant identity and Fibonacci-kernel margin preservation for symbolic γ coefficients.

Formal verification: FormalStatus(NameCert$_{FibonacciPolyaJensenKernel}$, theoremChecked)

Lean target: `BEDC.Derived.FibonacciPolyaJensenKernel.fibonacci\_polya\_jensen\_kernel\_certificate`

Bridge status: none

Not claimed: not a proof of RH; a hyperbolicity-preserving Fibonacci Polya-Schur smoothing of the actual xi Jensen criterion, degree-2 discriminant identity only; not a Zeckendorf model of xi coefficients.

Upgrade path: lift to general degree d via the full Polya-Schur preservation, and add the deformation parameter λ in $W_m(\lambda D)$ to certify a Jensen stability margin over rational λ-intervals.

Chapter 9153

The Fibonacci–Polya Jensen Deformation

This chapter records a finite lambda-deformation of the Fibonacci–Polya Jensen kernel. Let

$$\widehat{J}^{d,n}(X) = X^d J^{d,n}(1/X)$$

be the reciprocal Jensen polynomial attached to the Taylor coefficients γ_n of the Riemann ξ function. For a Fibonacci cube of length m, set

$$W_m(T) = \sum_k \binom{m-k+1}{k} T^k.$$

The deformation is

$$F^{(\lambda)}_{m,d,n}(X) = W_m(\lambda D)\widehat{J}^{d,n}(X),$$

where $D = d/dX$. At $\lambda = 0$ this is the actual reciprocal xi Jensen polynomial. At $\lambda = 1$ this is the Fibonacci-smoothed window. The parameter therefore embeds the actual degree window into a finite Polya–Schur smoothing family without replacing the xi coefficients by a Fibonacci counting model.

9153.1 Degree-Two Lambda Discriminant

In degree two write

$$\widehat{J}^{2,n}(X) = aX^2 + 2bX + c,$$

where $a = \gamma_n$, $b = \gamma_{n+1}$, and $c = \gamma_{n+2}$. Since

$$W_m(\lambda D) = 1 + m\lambda D + \frac{(m-1)(m-2)}{2}\lambda^2 D^2$$

on degree-two polynomials, the deformed window is

$$F^{(\lambda)}_{m,2,n}(X) = aX^2 + (2am\lambda + 2b)X + \left(a\lambda^2 m^2 - 3a\lambda^2 m + 2a\lambda^2 + 2bm\lambda + c\right).$$

Its discriminant is the finite identity

$$\operatorname{disc} F^{(\lambda)}_{m,2,n} = 4\left((b^2 - ac) + (3m-2)\lambda^2 a^2\right).$$

Equivalently,

$$\operatorname{disc} F^{(\lambda)}_{m,2,n} - 4(b^2 - ac) = 4(3m-2)\lambda^2 a^2.$$

For $m \geq 1$, the Fibonacci term is nonnegative after the usual shift $m \mapsto m+1$. Thus the deformation certifies a continuous hyperbolicity stability margin from the actual Jensen discriminant at $\lambda = 0$ to the Fibonacci-smoothed margin at $\lambda = 1$.

Theorem 9153.1 (Fibonacci–Polya lambda-deformation Jensen certificate)**.** *The degree-two lambda-deformed Fibonacci–Polya Jensen window has the symbolic discriminant certificate*

$$\operatorname{disc} F^{(\lambda)}_{m,2,n} = 4\left((\gamma_{n+1}^2 - \gamma_n\gamma_{n+2}) + (3m-2)\lambda^2\gamma_n^2\right).$$

The specialization $\lambda = 0$ gives the actual reciprocal Jensen discriminant $4(\gamma_{n+1}^2 - \gamma_n\gamma_{n+2})$, the specialization $\lambda = 1$ gives the Fibonacci-smoothed degree-two window, and the difference from the actual Jensen discriminant is the explicit lambda-square margin.

Lean4 ✓ `BEDC.Derived.FibonacciPolyaJensenDeformation.fibonacci polya jensen deformation certificate`

9153.2 Honest Scope

This is a finite Jensen stability-margin certificate under Fibonacci Polya–Schur deformation, interpolating the actual reciprocal xi Jensen polynomial at $\lambda = 0$ and the Fibonacci-smoothed window at $\lambda = 1$. It is a genuine finite RH-shadow, not a sharper attack on RH, not a Zeckendorf model of xi, and not a proof of the Riemann Hypothesis. The coefficients $\gamma_n, \gamma_{n+1}, \gamma_{n+2}$ remain symbolic throughout.

The Lean certificate proves the algebraic normal-form identity for the lambda discriminant, the $\lambda = 0$ and $\lambda = 1$ specializations, and the lambda-square margin identity. It does not evaluate xi coefficients, does not assert global hyperbolicity for all Jensen degrees, and does not discharge any analytic zero-location theorem. Its role is to place the actual degree-two Jensen discriminant inside a finite hyperbolicity-preserving deformation whose extra Fibonacci contribution is visible and nonnegative in the certified margin.

Falsifiable prediction: A recomputation of the symbolic degree-two deformation that changes the coefficient tuple $(a, 2am\lambda + 2b, a\lambda^2m^2 - 3a\lambda^2m + 2a\lambda^2 + 2bm\lambda + c)$, the lambda discriminant term $4((b^2 - ac) + (3m-2)\lambda^2a^2)$, the $\lambda = 0$ Turan specialization, or the $\lambda = 1$ Fibonacci-window specialization refutes this packet.

Independence witness: riemann-hypothesis, zeckendorf-coefficient-models: this packet is a finite Jensen stability-margin certificate under Fibonacci Polya–Schur deformation, interpolating the actual reciprocal xi Jensen polynomial at $\lambda = 0$ and the Fibonacci-smoothed window at $\lambda = 1$; it is not a proof of RH and not a model identifying xi coefficients with Fibonacci or Zeckendorf counts.

Closure status for NameCert$_{FibonacciPolyaJensenDeformation}$.

Theory closure: TheoryGate(NameCert$_{FibonacciPolyaJensenDeformation}$, scopedClosed)

Scope closed: degree-2 λ-discriminant identity plus $\lambda = 0$ and $\lambda = 1$ specializations and margin, with symbolic γ coefficients.

Formal verification: FormalStatus(NameCert$_{FibonacciPolyaJensenDeformation}$, theoremChecked)

Lean target: `BEDC.Derived.FibonacciPolyaJensenDeformation.fibonacci\_polya\_jensen\_deformation\_certificate`

Bridge status: none

Not claimed: not a proof of RH; a finite Jensen stability-margin certificate under Fibonacci deformation, degree-2 discriminant identity interpolating lambda=0 (actual Jensen) and lambda=1 (Fibonacci window); not a Zeckendorf model of xi.

Upgrade path: certify hyperbolicity and root separation over rational lambda-intervals for higher degree d.

Chapter 9154

The Zeckendorf Mobius Sieve Certificate

This chapter records a finite Mobius and squarefree arithmetic packet over Fibonacci–Zeckendorf windows. Let $F_0 = 0$, $F_1 = 1$, and $F_{n+2} = F_{n+1} + F_n$. For each $0 \le N < F_{m+2}$, the Zeckendorf representation writes N uniquely as a sum of nonadjacent Fibonacci numbers, and $s_Z(N)$ denotes the number of summands in that representation.

9154.1 Squarefree Zeckendorf-Hamming Polynomials

For each window length m, define the squarefree Zeckendorf-Hamming polynomial

$$SQ_m(u) = \sum_{1 \le N < F_{m+2}} \mu(N)^2 u^{s_Z(N)}$$

and its signed companion

$$M_m(u) = \sum_{1 \le N < F_{m+2}} \mu(N) u^{s_Z(N)}.$$

The certificate computes these polynomials as finite integer coefficient lists, with coefficient degree equal to Zeckendorf Hamming weight. The Lean carrier uses only finite recursion over natural numbers and integer lists: Fibonacci values, greedy Zeckendorf weights, squarefree testing, the Mobius value, and polynomial accumulation are all closed computations.

The squarefree arithmetic core is the finite sieve identity

$$\mu(n)^2 = \sum_{d^2 \mid n} \mu(d).$$

Inside the certificate this is checked for $1 \le n \le 20$ by the same Mobius and divisibility routines used by the window polynomials. The squarefree polynomial is therefore not merely a count of selected examples; it is tied to the classical squarefree sieve in the bounded arithmetic range recorded by the certificate.

9154.2 Finite Mertens Cancellation

The signed companion has evaluation

$$M_m(1) = \sum_{1 \le N < F_{m+2}} \mu(N).$$

The certificate records, for $1 \le m \le 12$, an explicit nonnegative slack witnessing the finite Mertens-cancellation inequality

$$\left(\sum_{1\le N<F_{m+2}} \mu(N)\right)^2 \le F_{m+2}.$$

In Lean this bound is carried as a natural-number slack equality

$$|M_m(1)|^2 + \mathrm{slack}_m = F_{m+2},$$

which avoids importing any analytic argument or any classical inequality principle. The theorem target also records the concrete polynomial window $m \le 8$, the squarefree sieve checks through $n = 20$, and the representative terminal Mertens slack certificate at $m = 12$.

Theorem 9154.1 (Zeckendorf Mobius sieve certificate)**.** *The finite Zeckendorf-window packet computes $SQ_m(u)$ and $M_m(u)$ for $1 \le m \le 8$, checks the squarefree sieve identity $\mu(n)^2 = \sum_{d^2|n} \mu(d)$ for $1 \le n \le 20$, and records explicit Mertens-cancellation slack certificates $|M_m(1)|^2 + \mathrm{slack}_m = F_{m+2}$ for $1 \le m \le 12$.*

Lean4 ✓ `BEDC.Derived.ZeckendorfMobiusSieve.zeckendorf mobius sieve certificate`

9154.3 Honest Scope

This is a finite Mobius/squarefree and Mertens-cancellation certificate over Zeckendorf windows. It places classical Mobius arithmetic inside finite Fibonacci windows and mirrors the RH-equivalent Mertens bound $M(x) = O(x^{1/2})$, but it is finite computed evidence, not a proof of that bound for all x, not an analytic zeta statement, and not a proof of the Riemann Hypothesis.

The squarefree polynomial is classically adjacent to the squarefree summatory structure behind $1/\zeta(2s)$, and the signed window is adjacent to the Mertens summatory function. The BEDC content here is the finite arithmetic interface: Zeckendorf weight supplies the Fibonacci window coordinate, Mobius values supply the signed arithmetic, the squarefree sieve supplies the bounded divisor check, and the slack equations supply finite cancellation witnesses. No Euler product, analytic continuation, zero-free region, or asymptotic estimate is claimed.

Falsifiable prediction: A recomputation of the finite Lean carrier that changes the greedy Zeckendorf weights for 6, 7, or 12, the squarefree status of 4 or 6, the Mobius values of 4, 6, or 30, the SQ_m and M_m coefficient lists through $m = 8$, the squarefree sieve checks through $n = 20$, or the Mertens slack certificates through $m = 12$, refutes this packet.

Independence witness: riemann-hypothesis, global-mertens-bound, analytic-zeta-claims: this packet is a finite Mobius/squarefree and Mertens-cancellation certificate over Zeckendorf windows only; it does not prove RH, does not prove $M(x) = O(x^{1/2})$ for all x, and does not assert an analytic zeta theorem.

Closure status for NameCert$_{ZeckendorfMobiusSieve}$.

Theory closure: TheoryGate(NameCert$_{ZeckendorfMobiusSieve}$, scopedClosed)

Scope closed: SQ_m/M_m concrete $m \le 8$, squarefree sieve $n \le 20$, Mertens bound $m \le 12$.

Formal verification: FormalStatus(NameCert$_{ZeckendorfMobiusSieve}$, theoremChecked)

Lean target: `BEDC.Derived.ZeckendorfMobiusSieve.zeckendorf\_mobius\_sieve\_certificate`

Bridge status: none

Not claimed: not a proof of RH or of the Mertens bound for all x; a finite Mobius/squarefree and Mertens-cancellation certificate over Zeckendorf windows only; the analytic zeta connection is heuristic.

Upgrade path: prove the squarefree residue-transfer sieve identity $SQ_m = \sum_d \mu(d)(C_{m,d^2,0} - 1)$ universally via the Zeckendorf residue recurrence, and extend the Mertens bound beyond the finite table.

Chapter 9155

Fibonacci-Cube Ihara Zeta

This chapter records a finite Ihara zeta computation for the Fibonacci-cube graphs. For $m \geq 1$, let Γ_m be the graph whose vertices are the binary words of length m with no adjacent 1s, with an edge between two words exactly when their Hamming distance is 1. Thus $|V(\Gamma_m)| = F_{m+2}$, and the first non-tree case is Γ_3.

Let A_m be the adjacency matrix of Γ_m, and let D_m be the diagonal degree matrix. Bass' determinant formula gives the reciprocal Ihara zeta expression

$$Z_{\Gamma_m}(u)^{-1} = (1-u^2)^{|E|-|V|} \det\bigl(I - A_m u + (D_m - I)u^2\bigr).$$

The Lean certificate works at the polynomial determinant layer:

$$\det\bigl(I - A_m u + (D_m - I)u^2\bigr).$$

It avoids encoding negative powers of $(1-u^2)$ as rational functions. For Γ_1 and Γ_2, the graphs are trees; the determinant is $1-u^2$, and the Bass prefactor cancels it in the reciprocal zeta. For Γ_3, the exponent $|E|-|V|$ is 0, so the reciprocal zeta is exactly the determinant.

9155.1 Finite Bass Determinants

The verified vertex orders are the lexicographic no-adjacent-1 orders

$$\Gamma_1 : \ 0, 1,$$

$$\Gamma_2 : \ 00, 01, 10,$$

and

$$\Gamma_3 : \ 000, 001, 010, 100, 101.$$

For these orders, the certificate evaluates the concrete Bass matrices with entries

$$\bigl(I - A_m u + (D_m - I)u^2\bigr)_{ij} = \delta_{ij} - A_{ij}u + (\deg(i) - 1)\delta_{ij}u^2.$$

The determinant identities are

$$\det\bigl(I - A_1 u + (D_1 - I)u^2\bigr) = 1 - u^2,$$

$$\det\bigl(I - A_2 u + (D_2 - I)u^2\bigr) = 1 - u^2,$$

and

$$\det(I - A_3u + (D_3 - I)u^2) = 1 - 2u^4 + u^8.$$

Equivalently, the first non-tree case factors as

$$1 - 2u^4 + u^8 = (u-1)^2(u+1)^2(u^2+1)^2.$$

Theorem 9155.1 (Fibonacci-cube Ihara determinant certificate)**.** *For the Fibonacci-cube graphs $\Gamma_1, \Gamma_2, \Gamma_3$, the polynomial Bass determinants are*

$$\det(I - A_1u + (D_1 - I)u^2) = 1 - u^2,$$

$$\det(I - A_2u + (D_2 - I)u^2) = 1 - u^2,$$

and

$$\det(I - A_3u + (D_3 - I)u^2) = 1 - 2u^4 + u^8 = (u-1)^2(u+1)^2(u^2+1)^2.$$

For Γ_3, since $|E| - |V| = 0$, this is also $Z_{\Gamma_3}(u)^{-1}$.

Lean4 ✓ `BEDC.Derived.FibonacciCubeIharaZeta.fibonacci cube ihara zeta certificate`

9155.2 Honest Scope

This packet is a finite Ihara (graph) zeta of the Fibonacci-cube graphs computed by the Bass determinant; it belongs to graph and dynamical zeta theory and to the graph-RH (Ramanujan) analogy, not to the actual Riemann xi or its Jensen criterion, and it is not a proof of the Riemann Hypothesis. The Lean object is a concrete polynomial determinant calculation over integer coefficient lists. It does not encode analytic continuation, the critical strip, the nontrivial zeros of $\zeta(s)$, the Riemann ξ function, or any Jensen-polynomial criterion for RH.

The finite analogy is still mathematically legitimate. Ihara zeta functions attach Euler-product-style zeta data to finite graphs, and their pole geometry is governed by graph spectra. In regular graph families, the Ramanujan condition is commonly expressed as a graph-theoretic RH analogue. The Fibonacci-cube calculation here stays inside that finite graph-zeta world: it verifies the Bass determinant data through Γ_3 and marks the first non-tree determinant explicitly, while leaving any uniform spectral theorem for all Γ_m outside the closed scope.

Falsifiable prediction: A recomputation of the listed Bass matrices for $\Gamma_1, \Gamma_2, \Gamma_3$ that changes the determinant coefficient lists $[1, 0, -1]$, $[1, 0, -1]$, or $[1, 0, 0, 0, -2, 0, 0, 0, 1]$ refutes this packet.

Independence witness: riemann-hypothesis, riemann-xi-jensen-criterion: this packet is a finite Ihara graph zeta computation for Fibonacci-cube graphs, not a statement about the actual Riemann ξ function, not a Jensen criterion certificate, and not a proof of RH.

Closure status for NameCert$_{FibonacciCubeIharaZeta}$.

Theory closure: TheoryGate(NameCert$_{FibonacciCubeIharaZeta}$, scopedClosed)

Scope closed: Bass det identity for Γ_1 through Γ_3.

Formal verification: FormalStatus(NameCert$_{FibonacciCubeIharaZeta}$, theoremChecked)

Lean target: `BEDC.Derived.FibonacciCubeIharaZeta.fibonacci\_cube\_ihara\_zeta\_certificate`

Bridge status: none

Not claimed: not a proof of RH; a finite Ihara graph zeta of the Fibonacci cube (graph-RH/Ramanujan analogy) only, not the actual Riemann xi.

Upgrade path: extend the Bass determinant to all Γ_m and connect the graph-RH (Ramanujan) property to the Fibonacci-cube spectral gap.

Chapter 9156

The Finite-Window Fibonacci–RH Program

The Riemann Hypothesis has several equivalent finite-window formulations: hyperbolicity of the Jensen polynomials of ξ, positivity of Li's coefficients, positivity of Weil's explicit-formula quadratic form, and cancellation bounds such as $M(x) = O(x^{1/2+\varepsilon})$ for the Mertens function. This chapter is the organizing summary of a finite, axiom-free program that places Fibonacci and Zeckendorf structure inside these finite-window criteria. It proves no new theorem; it records which finite certificates were built, how each touches a genuine RH criterion, and where the exact-arithmetic frontier honestly ends.

9156.1 The Certificates

Each certificate below is a finite, 0-axiom object verified in the Lean development. None is a proof of the Riemann Hypothesis; each is a finite shadow of, or a finite carrier for, a genuine RH-equivalent criterion.

Certificate	**Finite content**	**Genuine RH**
Fibonacci–Pólya Jensen kernel	$\operatorname{disc} F_{m,2,n} = 4[(\gamma_{n+1}^2 - \gamma_n\gamma_{n+2}) + (3m-2)\gamma_n^2]$	the actual ξ Je
Jensen kernel deformation	$\operatorname{disc} F_{m,2,n}^{(\lambda)}$, $\lambda = 0$ the exact ξ Jensen window	Jensen stability
Zeckendorf–Möbius sieve	$\left(\sum_{N<F_{m+2}} \mu(N)\right)^2 \le F_{m+2}$	the Mertens bo
Fibonacci-cube Ihara zeta	Bass $\det(I - Au + (D-I)u^2)$, $\Gamma_3 = (u-1)^2(u+1)^2(u^2+1)^2$	the graph-theor

The first two are the hyperbolicity flavor: the Fibonacci-cube weight polynomial W_m is real-rooted, so $W_m(D)$ is a finite Pólya–Schur hyperbolicity preserver, and the Griffin–Ono–Rolen–Zagier theorem gives unconditional large-n hyperbolicity of the smoothed windows. The third is the arithmetic flavor: it places Möbius and squarefree arithmetic inside Fibonacci windows and records a finite Mertens cancellation certificate. The fourth is the graph-zeta flavor: it computes the Ihara zeta of the Fibonacci-cube graphs through the Bass determinant and marks the first non-tree case.

Theorem 9156.1 (Program summary)**.** *The four finite certificates above are verified in Lean as axiom-free, propext-free objects. Together they realize the hyperbolicity, arithmetic, and graph-zeta flavors of the finite-window Fibonacci–RH program on exact integer and polynomial data.*

↪ variant `BEDC.Derived.FibonacciPolyaJensenKernel.fibonacci polya jensen kernel certificate`
↪ variant `BEDC.Derived.ZeckendorfMobiusSieve.zeckendorf mobius sieve certificate`
↪ variant `BEDC.Derived.FibonacciCubeIharaZeta.fibonacci cube ihara zeta certificate`

9156.2 The Program Boundary

This program is complete as a finite, exact-arithmetic certificate program. The three flavors above exhaust the finite-window RH criteria that reduce to exact integer, polynomial, or $\mathbb{Z}[\varphi]$ arithmetic: Jensen hyperbolicity through symbolic ξ coefficients, Mertens cancellation through exact Möbius sums, and the graph zeta through an exact Bass determinant.

The one genuinely distinct criterion not reduced here is Weil's explicit-formula positivity: the finite Galerkin quadratic form of the Weil functional on Fibonacci-windowed test functions. That object is real and central, but its matrix entries are transcendental, involving the Euler–Mascheroni constant, $\log 4\pi$, archimedean integrals, and von Mangoldt evaluations at logarithms of integers. It is therefore not an exact-arithmetic 0-axiom object; a certificate would require certified interval arithmetic together with a rational positive-semidefinite (Cholesky) certificate, a different and substantially harder formalization. It is recorded here as the honest open direction, not as a completed certificate.

Falsifiable prediction: A recomputation of any listed finite certificate that contradicts its displayed identity or inequality at the enumerated parameters refutes the corresponding packet and this summary's placement of it.

Independence witness: finite-window-fib-rh-program: these are finite axiom-free shadows and carriers of genuine RH-equivalent criteria (Jensen hyperbolicity, the Mertens bound, and the graph-RH analogy); none is a proof of the Riemann Hypothesis, none evaluates the analytic ξ or its zeros, and the Weil explicit-formula positivity criterion is left as an honest open direction outside the exact-arithmetic scope.

Closure status for NameCert$_{FiniteWindowFibRHProgramHub}$.

Theory closure: TheoryGate(NameCert$_{FiniteWindowFibRHProgramHub}$, scopedClosed)

Scope closed: Definition 9156.1 records the four finite axiom-free certificates and section 9156.2 states the exact-arithmetic boundary; the program is complete as a finite exact-arithmetic certificate program.

Formal verification: FormalStatus(NameCert$_{FiniteWindowFibRHProgramHub}$, theoremChecked)

Lean target: `BEDC.Derived.FibonacciPolyaJensenKernel.fibonacci\_polya\_jensen\_kernel\_certificate`

Bridge status: none

Not claimed: not a proof of RH; a finite exact-arithmetic certificate program of shadows/carriers of RH-equivalent criteria; the Weil explicit-formula positivity criterion is an honest open direction requiring certified interval arithmetic, not a completed certificate.

Upgrade path: formalize the Fibonacci-windowed Weil positivity matrix with certified interval-arithmetic entries and a rational Cholesky certificate, or prove a density/fundamentality theorem for a Fibonacci/Zeckendorf restriction of a finite-window RH criterion.

Chapter 9157

A Concrete Naming Certificate for Euler's Pentagonal Number Theorem

Following Euler and Franklin, the Euler pentagonal number theorem packet records a finite BEDC reconstruction of the classical partition-theory identity

$$\prod_{n\geq 1}(1-q^n) = \sum_{k=-\infty}^{\infty}(-1)^k q^{k(3k-1)/2}.$$

The generalized pentagonal numbers are

$$g_k = \frac{k(3k-1)}{2}$$

for $k \in \mathbb{Z}$, and the attached sign is $(-1)^k$. The BEDC carrier does not assert analytic convergence of the infinite product or a completed infinite formal power series. It records an exact finite reconstruction of Euler's pentagonal number theorem and Franklin's sign-reversing involution as finite NameCert data; the generating-function identity is truncated to a finite polynomial coefficient identity.

Definition 9157.1 (Euler–Franklin finite carrier)**.** *An Euler–Franklin finite carrier consists of the following displayed rows:*

1. *a low-degree q-polynomial row for the product $\prod_{n=1}^{30}(1-q^n)$;*
2. *a coefficient window row exposing the coefficients of q^d for $0 \leq d \leq 26$;*
3. *a generalized pentagonal sign row, computed by the finite searches $d = k(3k-1)/2$ and $d = k(3k+1)/2$ for $0 \leq k \leq d$;*
4. *a distinct-partition enumerator row for strict decreasing positive parts summing to n;*
5. *a signed partition-count row computing $\sum_{\lambda\in D(n)}(-1)^{\#\lambda}$ for $0 \leq n \leq 12$;*
6. *a local NameCert row naming the finite Euler coefficient window and the finite Franklin cancellation count as the same pentagonal-sign packet.*

The classifier has no coordinate for convergence, infinite products, infinite sums over all integers, quotient power series, or a host-side partition theorem outside the displayed finite data.

Theorem 9157.2 (Finite Euler coefficient certificate)**.** *For the finite product*

$$P_{30}(q) = \prod_{n=1}^{30}(1-q^n),$$

the coefficient row through degree 26 *is*

$$1, -1, -1, 0, 0, 1, 0, 1, 0, 0, 0, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1.$$

Equivalently, the nonzero entries in this window occur at

$$(0, +1), (1, -1), (2, -1), (5, +1), (7, +1), (12, -1), (15, -1), (22, +1), (26, +1),$$

matching the generalized pentagonal data

$$(0, 0, +), (1, 1, -), (-1, 2, -), (2, 5, +), (-2, 7, +), (3, 12, -), (-3, 15, -), (4, 22, +), (-4, 26, +).$$

Proof. The Lean certificate computes the coefficient window by a finite coefficient recurrence for multiplication by $1-q^n$. The resulting row is compared by Boolean list equality with the displayed coefficient table, and the pentagonal-sign row is computed independently by finite search over generalized pentagonal shapes. □

Theorem 9157.3 (Finite Franklin signed-count certificate)**.** *Let $D(n)$ be the finite set of partitions of n into distinct positive parts, represented as strict decreasing lists. Then the finite signed count*

$$\sum_{\lambda \in D(n)} (-1)^{\#\lambda}$$

agrees with the pentagonal sign for every $0 \le n \le 12$. In particular the checked rows include

$$n = 0 : +1, \quad n = 1 : -1, \quad n = 2 : -1, \quad n = 5 : +1, \quad n = 7 : +1, \quad n = 12 : -1.$$

Proof. The Lean certificate enumerates distinct-part partitions by bounded recursion over the largest permitted part. Each listed partition contributes $+1$ or -1 according to the parity of its length. The signed count is then compared by kernel reduction with the same finite pentagonal-sign row used in the coefficient certificate. □

Theorem 9157.4 (Euler–Franklin finite NameCert reconstruction)**.** *The Euler–Franklin packet is a finite NameCert reconstruction of the classical theorem: the Euler side verifies the product $\prod_{n=1}^{30}(1-q^n)$ through degree* 26*, while the Franklin side verifies the distinct-partition signed count through $n = 12$. Franklin's sign-reversing involution is represented at this scope by its finite cancellation consequence: all non-pentagonal signed rows in the checked range cancel, and only the pentagonal fixed rows remain.*

`Lean4 ✓ BEDC.Derived.EulerPentagonalNumberTheorem.euler pentagonal number theorem certificate`

Proof. Combine the finite Euler coefficient table, the independent pentagonal-sign computation, and the Franklin signed-count rows. The Lean target records these rows as a single conjunction. It is a finite certificate of the displayed reconstruction, not an analytic assertion about the infinite product. □

Falsifiable prediction: Every accepted Euler–Franklin finite carrier exposes the finite product row, coefficient window through degree 26, generalized pentagonal sign search, distinct-partition enumeration, signed count through $n = 12$, and local NameCert row before exporting an Euler pentagonal-number theorem conclusion.

Independence witness: q-series, distinct partitions, pentagonal sign: this packet is not bijective to a generic polynomial row or a generic partition row because it jointly binds the truncated Euler product, the generalized pentagonal sign search, and the finite Franklin signed cancellation count.

Closure status for NameCert$_{EulerPentagonalNumberTheoremUp}$.

Constructive story: The packet is generated from a finite coefficient window for $\prod_{n=1}^{30}(1-q^n)$, a bounded generalized-pentagonal sign search, a distinct-partition enumerator, a signed-count row through $n = 12$, and a local NameCert row tying the Euler and Franklin surfaces together.

Theory closure: TheoryGate(NameCert$_{EulerPentagonalNumberTheoremUp}$, scopedClosed)

Scope closed: finite truncation $\prod_{n=1}^{30}(1 - q^n)$ coefficient identity through degree 26 + Franklin signed distinct-partition count through $n = 12$

Formal verification: FormalStatus(NameCert$_{EulerPentagonalNumberTheoremUp}$, theoremChecked)

Lean target: `BEDC.Derived.EulerPentagonalNumberTheorem.euler\_pentagonal\_number\_theorem\_certificate`

Bridge status: none

Not claimed: a finite reconstruction of Euler's pentagonal number theorem and Franklin's involution; the generating-function identity is verified as a finite truncated polynomial coefficient identity, not as an infinite product convergence statement

Upgrade path: extend to the Jacobi triple product and the Rogers–Ramanujan identities as further finite q-series NameCert reconstructions

Chapter 9158

A Concrete Naming Certificate for Euler's Distinct–Odd Partition Identity

Following Euler and Glaisher, the Euler distinct–odd partition packet records a finite BEDC reconstruction of the classical identity

$$\prod_{n\geq 1}(1+q^n) = \prod_{n\geq 1}\frac{1}{1-q^{2n-1}}.$$

Equivalently, the number of partitions of n into distinct positive parts equals the number of partitions of n into odd positive parts. The BEDC carrier does not assert analytic convergence of either infinite product, nor does it close an all-n formal-power-series proof. It records an exact finite reconstruction of Euler's distinct-equals-odd partition identity and Glaisher's bijection as finite NameCert data.

Definition 9158.1 (Euler–Glaisher finite carrier)**.** *An Euler–Glaisher finite carrier consists of the following displayed rows:*

1. *a distinct-partition row, enumerating strict decreasing positive-part lists summing to n;*
2. *an odd-partition row, enumerating positive odd-part lists summing to n;*
3. *a count-comparison row verifying the equality of the two finite lists for $1 \leq n \leq 12$;*
4. *a low-degree q-polynomial row for the product $\prod_{n=1}^{12}(1+q^n)$;*
5. *a coefficient-window row exposing the coefficients of q^d for $0 \leq d \leq 12$;*
6. *a Glaisher row sending an odd partition to a distinct partition by binary expansion of the multiplicity of each odd part;*
7. *a reverse Glaisher row sending a distinct part $2^a u$ with u odd to 2^a copies of u;*
8. *a local NameCert row tying the finite count table, the generating-function coefficient window, and the displayed bijection witnesses into one named packet.*

The classifier has no coordinate for infinite products, completed power series, convergence, quotient partition equality, or a host-side all-n partition theorem outside the displayed finite data.

Theorem 9158.2 (Finite Euler distinct–odd count certificate)**.** *Let $D(n)$ be the finite list of partitions of n into distinct positive parts, represented as strict decreasing lists, and let $O(n)$ be the finite list of partitions of n into odd positive parts. For $1 \le n \le 12$ the checked count row is*

$$1, 1, 2, 2, 3, 4, 5, 6, 8, 10, 12, 15.$$

Thus $|D(n)| = |O(n)|$ throughout the displayed window. For example, at $n = 6$ the distinct side is

$$[6],\ [5, 1],\ [4, 2],\ [3, 2, 1],$$

and the odd side is

$$[5, 1],\ [3, 3],\ [3, 1, 1, 1],\ [1, 1, 1, 1, 1, 1].$$

Proof. The Lean certificate enumerates distinct partitions by bounded recursion over the largest permitted part. It enumerates odd partitions by bounded recursion over the largest odd head allowed at the current step. Each displayed equality is then checked by kernel reduction of the two list lengths. □

Theorem 9158.3 (Finite generating-function coefficient certificate)**.** *For the finite product*

$$P_{12}(q) = \prod_{n=1}^{12} (1 + q^n),$$

the coefficient row through degree 12 is

$$1, 1, 1, 2, 2, 3, 4, 5, 6, 8, 10, 12, 15.$$

For each $0 \le d \le 12$, this coefficient agrees with the finite distinct-partition count $|D(d)|$.

Proof. The Lean certificate computes $P_{12}(q)$ by finite polynomial multiplication over integer coefficient lists. The coefficient row is compared by Boolean list equality with the displayed table, and each coefficient in the window is separately identified with the corresponding distinct-partition count by reduction. □

Theorem 9158.4 (Finite Glaisher bijection witnesses)**.** *Glaisher's map from odd partitions to distinct partitions is encoded by carrying equal odd parts in base 2: if an odd part u has multiplicity $m = \sum_a \epsilon_a 2^a$, then every nonzero bit contributes the distinct part $2^a u$. The reverse map writes each distinct part as $2^a u$ with u odd and expands it into 2^a copies of u.*

In the checked finite carrier, the Lean rows include the round-trip witnesses

$$[5, 1] \mapsto [5, 1] \mapsto [5, 1],$$

$$[3, 3] \mapsto [6] \mapsto [3, 3],$$

$$[3, 1, 1, 1] \mapsto [3, 2, 1] \mapsto [3, 1, 1, 1],$$

and

$$[1, 1, 1, 1, 1, 1] \mapsto [4, 2] \mapsto [1, 1, 1, 1, 1, 1].$$

The same finite rows record preservation of the total sum for representative odd-to-distinct and distinct-to-odd expansions.

Proof. The Lean certificate implements the carry operation as an insertion into a finite distinct list, and implements the reverse operation by repeated halving until an odd kernel is reached. The displayed rows are concrete reductions of these functions. They are explicit finite Glaisher witnesses, not an all-n proof of bijectivity. □

Theorem 9158.5 (Euler–Glaisher finite NameCert reconstruction)**.** *The Euler–Glaisher packet is a finite NameCert reconstruction of the classical identity: the partition-count side verifies* $|D(n)| = |O(n)|$ *through* $n = 12$*, the generating-function side verifies the coefficients of* $\prod_{n=1}^{12}(1+q^n)$ *through degree* 12*, and the Glaisher side records concrete binary-carry round-trip witnesses.*

Lean4 ✓ `BEDC.Derived.EulerDistinctOddPartition.euler distinct odd partition certificate`

Proof. Combine the finite distinct-partition count row, the finite odd-partition count row, the coefficient-window row for $P_{12}(q)$, and the displayed Glaisher witnesses. The Lean target records these rows as a single conjunction. It is a finite certificate of the displayed reconstruction, not an analytic assertion about infinite products. □

Falsifiable prediction: Every accepted Euler–Glaisher finite carrier exposes the distinct-partition enumerator, odd-partition enumerator, count row through $n = 12$, coefficient row for $\prod_{n=1}^{12}(1+q^n)$ through degree 12, Glaisher binary-carry map, reverse odd-kernel expansion, and local NameCert row before exporting an Euler distinct–odd partition conclusion.

Independence witness: distinct partitions, odd partitions, q-series, Glaisher binary carry: this packet is not bijective to a generic polynomial row or a generic partition row because it jointly binds the finite distinct-count table, the finite odd-count table, the truncated Euler product coefficients, and the displayed Glaisher maps.

Closure status for NameCert$_{EulerDistinctOddPartitionUp}$.

Constructive story: The packet is generated from a distinct-partition enumerator, an odd-partition enumerator, a count-comparison row through $n = 12$, a coefficient window for $\prod_{n=1}^{12}(1+q^n)$ through degree 12, finite Glaisher binary-carry witnesses, reverse odd-kernel expansion witnesses, and a local NameCert row tying the Euler and Glaisher surfaces together.

Theory closure: TheoryGate(NameCert$_{EulerDistinctOddPartitionUp}$, scopedClosed)

Scope closed: distinct-part = odd-part partition counts through $n = 12$ + generating-function coefficients + Glaisher bijection witnesses

Formal verification: FormalStatus(NameCert$_{EulerDistinctOddPartitionUp}$, theoremChecked)

Lean target: `BEDC.Derived.EulerDistinctOddPartition.euler\_distinct\_odd\_partition\_certificate`

Bridge status: none

Not claimed: a finite reconstruction of Euler's distinct-equals-odd identity and Glaisher's bijection; counts verified through $n = 12$, not an all-n generating-function proof

Upgrade path: extend to the general Glaisher d-distinct=d-fold identity and to further q-series partition bijections

Chapter 9159

A Concrete Naming Certificate for Euler's Partition Pentagonal Recurrence

Following Euler, the partition pentagonal recurrence packet records a finite BEDC reconstruction of the classical partition function $p(n)$, where $p(n)$ counts unordered sums of positive integers with total n. The recurrence is

$$p(n) = \sum_{k\geq 1}(-1)^{k-1}\left(p\left(n - \frac{k(3k-1)}{2}\right) + p\left(n - \frac{k(3k+1)}{2}\right)\right),$$

with $p(0) = 1$ and every negative-index term omitted. This is the reciprocal consequence of Euler's pentagonal number theorem:

$$\left(\prod_{n\geq 1}(1-q^n)\right)\left(\sum_{n\geq 0} p(n)q^n\right) = 1.$$

The BEDC carrier below records an exact finite reconstruction of Euler's pentagonal recurrence for the partition function, verified through $n = 20$ together with the reciprocal identity against the pentagonal product; it is not an all-n generating-function proof.

Definition 9159.1 (Partition pentagonal recurrence finite carrier)**.** *A partition pentagonal recurrence finite carrier consists of the following displayed rows:*

1. *a generalized pentagonal row*

$$g_k = \frac{k(3k-1)}{2}, \qquad g'_k = \frac{k(3k+1)}{2}$$

   for positive k;

2. *a sign row $(-1)^{k-1}$ for the paired contribution at g_k and g'_k;*
3. *a prefix table row for $p(0), p(1), \ldots, p(20)$ generated by the pentagonal recurrence, not by enumerating partitions;*
4. *a recurrence-check row exposing the finite alternating sums at representative degrees $5, 10, 15, 20$;*
5. *a q-polynomial row for the truncated partition generating function $\sum_{n=0}^{20} p(n)q^n$;*

6. *a reciprocal row checking the coefficients of*

$$\left(\prod_{n=1}^{20}(1-q^n)\right)\left(\sum_{n=0}^{20}p(n)q^n\right)$$

through degree 20*;*

7. *a local NameCert row naming the finite recurrence table and the finite reciprocal coefficient window as the same Euler partition packet.*

The classifier has no coordinate for convergence of the infinite product, an infinite formal-power-series inverse, a Hardy–Ramanujan analytic estimate, or a host-side all-n partition theorem outside the displayed finite data.

Theorem 9159.2 (Finite partition value certificate)**.** *The checked partition row through* $n = 20$ *is*

$$1, 1, 2, 3, 5, 7, 11, 15, 22, 30, 42, 56, 77, 101, 135, 176, 231, 297, 385, 490, 627.$$

Equivalently,

$$p(0) = 1, \quad p(6) = 11, \quad p(10) = 42, \quad p(20) = 627.$$

Proof. The Lean certificate generates the table by the finite pentagonal recurrence itself. For each new n, it reads only the already generated prefix values at offsets $n - g_k$ and $n - g'_k$ whose indices are nonnegative, sums them with the sign $(-1)^{k-1}$, and appends the resulting integer value to the table. □

Theorem 9159.3 (Finite Euler recurrence certificate)**.** *The finite recurrence row includes the representative checks*

$$p(5) = p(4) + p(3) - p(0),$$
$$p(10) = p(9) + p(8) - p(5) - p(3),$$
$$p(15) = p(14) + p(13) - p(10) - p(8) + p(3) + p(0),$$

and

$$p(20) = p(19) + p(18) - p(15) - p(13) + p(8) + p(5).$$

These are the displayed finite instances of Euler's pentagonal recurrence.

Proof. Each row is a kernel reduction of the same prefix-table recurrence used to compute the values. The stopping condition is finite: once $k(3k-1)/2 > n$, all later positive generalized pentagonal offsets are outside the prefix window. □

Theorem 9159.4 (Finite reciprocal identity certificate)**.** *Let*

$$P_{20}(q) = \prod_{n=1}^{20}(1-q^n)$$

and

$$F_{20}(q) = \sum_{n=0}^{20}p(n)q^n.$$

The checked coefficient row of $P_{20}(q)F_{20}(q)$ *through degree* 20 *is*

$$1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.$$

Thus the finite carrier verifies the reciprocal identity

$$P_{20}(q)F_{20}(q) \equiv 1 \pmod{q^{21}}.$$

Proof. The Lean certificate reuses the finite q-polynomial multiplication and pentagonal product row from the Euler pentagonal-number certificate. It multiplies the product row by the partition prefix row and checks each coefficient from degree 0 through degree 20 by reduction. □

Theorem 9159.5 (Partition pentagonal recurrence finite NameCert reconstruction)**.** *The partition pentagonal recurrence packet is an exact finite reconstruction of Euler's pentagonal recurrence for $p(n)$: the recurrence-generated values are checked through $n = 20$, the representative pentagonal recurrence rows are checked at $n = 5, 10, 15, 20$, and the reciprocal identity against $\prod(1 - q^n)$ is checked through degree 20.*

Lean4 ✓ `BEDC.Derived.PartitionPentagonalRecurrence.partition pentagonal recurrence certificate`

Proof. Combine the finite prefix table, the representative recurrence rows, and the reciprocal coefficient window. The Lean target records these rows as one conjunction. It is a finite NameCert reconstruction of Euler's partition recurrence, not an all-n generating-function proof. □

Falsifiable prediction: Every accepted partition pentagonal recurrence carrier exposes the generalized pentagonal offsets, alternating sign row, recurrence-generated prefix table through $n = 20$, representative recurrence checks, reciprocal polynomial product row, coefficient window through degree 20, and local NameCert row before exporting an Euler partition-recurrence conclusion.

Independence witness: partition values, pentagonal recurrence, reciprocal q-series: this packet is not bijective to a generic partition table or a generic polynomial row because it jointly binds the recurrence-generated prefix values, the generalized pentagonal offsets, and the finite inverse relation against $\prod(1 - q^n)$.

Closure status for NameCert$_{PartitionPentagonalRecurrenceUp}$**.**

Constructive story: The packet is generated from generalized pentagonal offsets, alternating signs, a recurrence-generated partition prefix row through $n = 20$, representative recurrence checks, a finite pentagonal product, a partition generating-function prefix, a reciprocal coefficient window through degree 20, and a local NameCert row tying the recurrence and product surfaces together.

Theory closure: TheoryGate(NameCert$_{PartitionPentagonalRecurrenceUp}$, scopedClosed)

Scope closed: partition values $p(0)$ through $p(20)$, the pentagonal recurrence, and the reciprocal identity against $\prod(1 - q^n)$ through degree 20

Formal verification: FormalStatus(NameCert$_{PartitionPentagonalRecurrenceUp}$, theoremChecked)

Lean target: `BEDC.Derived.PartitionPentagonalRecurrence.partition\_pentagonal\_recurrence\_certificate`

Bridge status: none

Not claimed: a finite reconstruction of Euler's pentagonal recurrence for $p(n)$; verified through $n = 20$, not an all-n generating-function proof

Upgrade path: extend the reciprocal identity to arbitrary truncation degree and connect to the Hardy–Ramanujan asymptotics as an external analytic layer

Chapter 9160

A Concrete Naming Certificate for $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$ packet records the finite inspection route by which an Ishihara-style argument may expose boundedness information for a real-name sequence without importing an omniscience principle. It sits beside the inspection boundary of chapter 1665, the finite-test boundary of chapter 8578, the stream window surface $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular rational readback $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, the dyadic tolerance ledger $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291, the located real route $\mathsf{LocatedReal}^{\uparrow}$ in **??**, and the terminal real-name seal $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet supplies a finite boundedness test for L10 Real and Completion consumers; it does not supply WLPO, LLPO, LPO, trichotomy, a selected supremum, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9160.1 (Ishihara boundedness finite-test carrier)**.** *An* $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, L, U, K, R, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row whose boundedness-facing read is being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for the current boundedness request.*
3. *L is the lower finite-test row, naming the inspected evidence that the displayed window stays within the lower side of the bound.*
4. *U is the upper finite-test row, naming the inspected evidence that the displayed window stays within the upper side of the bound.*
5. *K is the* $\mathsf{LocatedReal}^{\uparrow}$ *split row through which the boundedness request is read as a located comparison rather than as a global order decision.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget used by the lower and upper tests.*
8. *D is the finite boundedness ledger saying which inspected coordinates, tolerances, and comparison margins have been exposed.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, L, U, K, R, Q, D have been displayed.*

10. *H records componentwise* hsame *transport for the source, window, tests, located split, readback, tolerance, boundedness ledger, and seal rows.*
11. *C records displayed* Cont *replay of the same finite boundedness inspection route.*
12. *P records* Pkg *provenance over* $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$, $\mathsf{IshiharaTrick}^{\uparrow}$, $\mathsf{IshiharaFiniteTestBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, L, U, K, R, Q, D, E, H, C, P, N$. It has no coordinate for WLPO, LLPO, LPO, trichotomy of real names, an externally selected bound, quotient stream equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.IshiharaBoundednessFiniteTestUp`

Theorem 9160.2 (Ishihara boundedness NameCert obligations)**.** *For every accepted* IshiharaBoundednessFiniteTest *carrier $F = (S, W, L, U, K, R, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}}$ *surface consists exactly of source-sequence admission, finite-window admission, lower-test admission, upper-test admission, located comparison split, regular rational readback, dyadic tolerance budgeting, boundedness-ledger exposure, terminal Real-seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 9160.1. The visible boundedness data are $S, W, L, U, K, R, Q, D, E, H, C, P,$

The boundedness read begins with the finite window W over the source row S. The lower and upper test rows L, U are then interpreted through the located split K, the regular readback R, and the dyadic budget Q before the finite boundedness ledger D is exposed.

The terminal Real seal E is downstream of those rows. Transport, replay, provenance, and local naming preserve the same finite route, so no WLPO, LLPO, LPO, trichotomy, selected bound, quotient equality, choice schedule, or ambient Real-completeness coordinate can enter the local NameCert surface. □

Theorem 9160.3 (Ishihara boundedness located case split)**.** *Every boundedness-facing read of an accepted* $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$ *packet passes through the displayed located split row K before it may consume the lower or upper finite-test rows. The split exposes only the finite comparison route W, K, R, Q, D and does not decide a global order on real names.*

Proof. Project the rows W, L, U, K, R, Q, D from the carrier. The two test rows are meaningful only against the displayed window W and the located comparison row K.

The regular readback R and dyadic budget Q attach the split to a finite rational request. The ledger D then records the boundedness-facing comparison margins that were actually inspected.

Since the carrier has no global order row, no trichotomy row, and no quotient-real equality row, a consumer can read the case split only as located finite evidence. It cannot turn the lower and upper test rows into an omniscience principle. □

Theorem 9160.4 (Ishihara boundedness L10 route)**.** *Any L10 Real or Completion consumer using an accepted* $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$ *packet must read the* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget Q,* $\mathsf{LocatedReal}^{\uparrow}$ *split K, finite boundedness ledger D, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N before naming a boundedness conclusion.*

Proof. Apply Definition 9160.2. It exposes the finite source, window, test, located split, readback, tolerance, ledger, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 9160.3. The boundedness conclusion reaches the two test rows only through the located comparison route W, K, R, Q, D.

The terminal $\mathsf{Real}^{\uparrow}$ seal E is therefore a handoff boundary for L10 Real and Completion consumers, not an ambient completeness theorem. The structural rows H, C, P, N replay and name the same route, excluding WLPO, LLPO, LPO, trichotomy, selected suprema, quotient stream equality, and countable choice from the exported boundedness surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}}$.

Constructive story: An $\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}$ packet is generated from a source sequence, a finite StreamName window, lower and upper finite-test rows, a LocatedReal split, RegSeqRat readback, DyadicRatCore tolerance budget, boundedness ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9160.1, Definition 9160.2, Definition 9160.3, and Definition 9160.4 seed the finite boundedness test route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IshiharaBoundednessFiniteTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close WLPO, LLPO, LPO, trichotomy, selected suprema, quotient stream equality, countable choice, ambient Real completeness, Lean-checked obligations, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper test exactness, located split stability, StreamName and RegSeqRat readback compatibility, DyadicRatCore tolerance preservation, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IshiharaBoundednessFiniteTestUp.

Chapter 9161

A Concrete Naming Certificate for $\mathsf{FiniteMetricChainBound}^{\uparrow}$

Following Bishop and Bridges on constructive metric estimates, the $\mathsf{FiniteMetricChainBound}^{\uparrow}$ packet records the finite triangle-chain ledger by which metric distances, dyadic rational tolerances, and regular Cauchy readbacks are compared before compact finite-net or complete-metric consumers may use the estimate. It sits over the metric source of chapter 57, the rational and dyadic ledgers of chapter 37 and chapter 291, the finite-net source of chapter 2048, the compact metric source of chapter 129, the observation windows of chapter 126, and the regular rational readback of chapter 301. The packet names a finite ledger for chained triangle estimates only; it does not choose a path, assert a host geodesic, prove completeness, or import countable choice.

Definition 9161.1 (Finite metric chain-bound carrier)**.** *A* $\mathsf{FiniteMetricChainBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, R, D, T, C, W, Q, E, H, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the metric distance observations used by the chain.*
2. *R is the* $\mathsf{Rat}^{\uparrow}$ *rational comparison row for the displayed intermediate bounds.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic endpoint and error budgets for the chain.*
4. *T is the finite* $\mathsf{TotallyBounded}^{\uparrow}$ *or totally-bounded-metric net row consumed by compact finite-net readers.*
5. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, present only when the chain is read as a compact finite-net estimate.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which endpoints and intermediate observations are inspected.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from W and D.*
8. *E is the terminal estimate row, making the accumulated bound public only after M, R, D, T, C, W, Q have been displayed.*
9. *H records componentwise* hsame *transport for the metric, rational, dyadic, finite-net, compact, window, readback, estimate, provenance, and local naming rows.*

10. *P records* Pkg *provenance over* $\mathsf{FiniteMetricChainBound}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricChainBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, D, T, C, W, Q, E, H, P, N$. It has no coordinate for a selected shortest path, a host geodesic, quotient equality of metric names, countable choice, or ambient completeness.

Theorem 9161.2 (Finite metric chain-bound triangle obligation)**.** *For every accepted* $\mathsf{FiniteMetricChainBound}^{\uparrow}$ *carrier*

$$F = (M, R, D, T, C, W, Q, E, H, P, N),$$

the public triangle-chain estimate is read only after the metric source M, rational comparison row R, dyadic tolerance ledger D, finite-net row T, compact metric row C, finite window row W, and regular rational readback row Q are visible. The terminal estimate row E may therefore be consumed by compact finite-net and complete-metric Cauchy readers only as the displayed finite ledger

$$M, R, D, T, C, W, Q, E, H, P, N.$$

It exports no selected path, geodesic representative, quotient metric equality, countable-choice schedule, host completion, or ambient compactness theorem.

Proof. Project the carrier of Definition 9161.1. The metric row M, rational row R, and dyadic row D provide the only distance and tolerance data available to the chain.

Read the finite-net and compact rows next. The rows T and C make the estimate available to totally bounded and compact metric consumers without turning it into an open-cover compactness or selected-subsequence theorem.

Finally read the finite observation window W and regular rational readback Q before the estimate row E is public. Transport, provenance, and local naming preserve exactly this row sequence, so no excluded host object can be exported by the triangle-chain obligation. □

Theorem 9161.3 (Finite metric chain-bound NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteMetricChainBound}^{\uparrow}}$ *surface consists exactly of carrier admission for $M, R, D, T, C, W, Q, E, H, P, N$; metric-distance admission through M; rational comparison through R; dyadic tolerance exactness through D; compact finite-net exposure through T and C; finite-window inspection through W; regular rational readback through Q; terminal estimate nonescape through E; componentwise transport through H; provenance through P; and local naming through N.*

Proof. Start from the carrier rows. The analytic part of the NameCert surface is exactly the metric source, rational comparison, dyadic tolerance ledger, finite-net row, compact source, window row, regular readback, and terminal estimate.

The structural rows then preserve that analytic surface. Componentwise hsame transport keeps the same fields visible, package provenance records the sibling sources, and the local NameCert row names the packet.

Since the carrier contains no row for selected paths, geodesics, quotient metric names, choice schedules, host completions, or ambient compactness, the local certificate cannot export any of those readings. □

Falsifiable prediction: Every accepted $\mathsf{FiniteMetricChainBound}^{\uparrow}$ consumer that reads a compact finite-net or complete-metric Cauchy estimate must display Metric, Rat, DyadicRatCore, finite-net,

compact metric, StreamName, RegSeqRat, estimate, transport, provenance, and local NameCert rows before the bound is public.

Independence witness: metric, rat, dyadicratcore, totally-bounded-metric, compactmetric, streamname, regseqrat: $\mathsf{FiniteMetricChainBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds metric distances, rational comparisons, dyadic tolerances, finite-net routing, compact source rows, finite windows, regular rational readback, terminal estimate, transport, provenance, and local naming as one chain-bound ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricChainBound}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetricChainBound}^{\uparrow}$ packet is generated from Metric distance observations, Rat comparison rows, DyadicRatCore tolerance ledgers, totally bounded finite-net rows, CompactMetric sources, StreamName windows, RegSeqRat readbacks, a terminal estimate row, componentwise hsame transport, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricChainBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9161.1, Definition 9161.2, and Definition 9161.3 seed the finite triangle-chain ledger over metric, rational, dyadic, finite-net, compact, window, readback, estimate, transport, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetricChainBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected paths, host geodesics, quotient metric equality, countable choice, host completion, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, triangle-chain row ordering, dyadic tolerance exactness, compact finite-net routing, StreamName window coverage, RegSeqRat readback, terminal estimate nonescape, provenance, and local naming for BEDC.Derived.FiniteMetricChainBoundUp.

Chapter 9162

A Concrete Naming Certificate for $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$

The $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ packet realizes the boundary-fixed flow-window branch of chapter 9463 as finite BEDC NameCert data. It sits after the hyperbolic metric source of chapter 3474, the Busemann boundary packet of chapter 3732, the horosphere packet of chapter 8521, the visual metric readback of chapter 3866, and the terminal Real seal of chapter 38. The packet names boundary-fixed flow windows, Busemann levels, horosphere rows, visual-metric readback, Real-seal provenance, and local naming; it does not assert a host hyperbolic flow, quotient boundary equality, selected geodesic rays, physical observer dynamics, or ambient compactification.

Definition 9162.1 (Horosphere Busemann flow-budget carrier)**.** *A* $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, B, L, S, V, E, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row for the finite disk or hyperbolic source window.*
2. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-fixed Busemann row.*
3. *L is the Busemann-level budget row, recording the finite level interval in which the flow window is read.*
4. *S is the* $\mathsf{HyperbolicHorosphere}^{\uparrow}$ *horosphere row determined by the same boundary row B and level budget L.*
5. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual-metric readback row for the boundary-facing flow window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal and provenance row, available only after M, B, L, S, V have been displayed.*
7. *R is the refusal row excluding quotient ideal-boundary equality, selected geodesic representatives, host hyperbolic flows, physical observer dynamics, and ambient compactification.*
8. *H records componentwise* hsame *transport for $M, B, L, S, V, E, R, C, P, N$.*
9. *C records displayed* Cont *replay from metric and boundary rows through the Busemann level, horosphere row, visual metric, Real seal, refusal, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundar}$ $\mathsf{HyperbolicHorosphere}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, L, S, V, E, R, H, C, P, N$. It has no coordinate for a host flow parameter space, quotient boundary equality, selected rays, physical observer trajectories, global horosphere foliations, or ambient compactification.

Theorem 9162.2 (Horosphere Busemann flow-budget NameCert surface)**.** *For every accepted* $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ *carrier*

$$F = (M, B, L, S, V, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}}$ *surface consists exactly of metric-read admission through M, Busemann-boundary admission through B, level-budget exposure through L, horosphere-row exposure through S, visual-metric readback through V, Real-seal provenance through E, refusal through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9162.1. The metric row M and Busemann boundary row B provide the only geometric source data available to the flow-budget packet.

Read the level, horosphere, and visual-metric rows next. The row L fixes the finite Busemann-level budget, S records the horosphere row over that same boundary and level data, and V supplies the boundary-facing visual-metric readback.

The Real seal E is downstream of those rows, while R, H, C, P, N refuse, transport, replay, source, and name the same finite packet. Since no row names a host flow parameter space, quotient boundary equality, selected ray, physical observer trajectory, global horosphere foliation, or compactification, the NameCert surface cannot export any such object. □

Theorem 9162.3 (Horosphere Busemann flow-budget TasteGate requirements)**.** *The Lean TasteGate for* $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ *must include* Nontrivial $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ *and* FieldFaithful $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ *instances for the same displayed carrier*

$$F = (M, B, L, S, V, E, R, H, C, P, N).$$

Nontriviality must distinguish a packet with all metric, boundary, level, horosphere, visual-metric, Real-seal, refusal, transport, replay, provenance, and local naming rows from any packet with a missing row. Field faithfulness must preserve the row order $M, B, L, S, V, E, R, H, C, P, N$ without replacing the finite flow budget by a host hyperbolic flow or quotient boundary object.

Proof. Read the requirement from the carrier. The packet is nontrivial only when all eleven rows are displayed together, because omitting any of $M, B, L, S, V, E, R, H, C, P, N$ removes a named face of the flow-budget surface.

Field faithfulness is the row-preservation demand. A classifier that changes the boundary row, level row, horosphere row, visual-metric readback, or Real-seal provenance while preserving the name would no longer classify the same finite packet.

The excluded host objects are absent from the carrier and from the NameCert surface of Definition 9162.2. The Lean TasteGate therefore certifies the finite budget rows, not a host flow theorem. □

Falsifiable prediction: Every accepted $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ flow-window read displays a HyperbolicMetric row, HyperbolicBusemannBoundary row, Busemann-level budget, HyperbolicHorosphere row, HyperbolicVisualMetric readback, RealUp seal, refusal, transport, replay, provenance, and local NameCert row before any boundary-fixed flow budget is public.

Independence witness: hyperbolic-metric, hyperbolic-busemann-boundary, hyperbolic-horosphere, hyperbolic-visual-metric, real: $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds boundary-fixed flow windows, Busemann levels, horosphere rows, visual-metric readback, Real-seal provenance, refusal, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}$ packet is generated from HyperbolicMetric source rows, HyperbolicBusemannBoundary rows, Busemann-level budgets, HyperbolicHorosphere rows, HyperbolicVisualMetric readback, RealUp seal provenance, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9162.1, Definition 9162.2, and Definition 9162.3 seed the finite boundary-fixed flow-budget packet over metric, Busemann boundary, level, horosphere, visual metric, Real seal, refusal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HorosphereBusemannFlowBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host hyperbolic flows, quotient ideal-boundary equality, selected geodesic rays, physical observer dynamics, global horosphere foliations, or ambient compactification.

Upgrade path: Obligation closure requires checked carrier admission, metric readback, Busemann-boundary admission, level-budget exactness, horosphere-row exposure, visual-metric readback, Real-seal provenance, refusal, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.HorosphereBusemannFlowBudgetUp.

Chapter 9163

A Concrete Naming Certificate for $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ packet records the finite route by which a displayed real-name convergence request is answered from explicit modulus data before any completed real is consumed. It sits over the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the diagonal Cauchy telescope of chapter 5923. The packet names a constructive convergence-rate extractor for Real and Completion consumers; it is not a selected limit, quotient representative, countable-choice modulus, host real equality, or ambient completeness theorem.

Definition 9163.1 (Real convergence-rate extractor carrier)**.** *A* $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, D, W, R, G, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the tolerance-request row, recording the requested output precision and the consumer that asks for a Real-facing convergence read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, translating Q into the dyadic radius and comparison slack used by the extractor.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the bounded prefix and tail windows inspected for that dyadic request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational observations obtained from W under the budget D.*
5. *G is the convergence-rate row, returning a displayed index after which all accepted readbacks meet the requested tolerance.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after Q, D, W, R, G have been displayed.*
7. *H records componentwise* hsame *transport for tolerance requests, dyadic budgets, finite windows, regular rational readbacks, convergence-rate rows, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the request row through D, W, R, G to the Real seal row S.*

9. *P is the* Pkg *provenance over* $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealConvergenceRate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, W, R, G, S, H, C, P, N$. It has no coordinate for a hidden limit selector, quotient representative, unbounded stream scan, countable choice, host equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9163.2 (Real convergence-rate extractor NameCert obligations)**.** *For every accepted* $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ *carrier $E = (Q, D, W, R, G, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtract}}$ *surface consists exactly of tolerance-request admission, dyadic budget exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational observation, convergence-rate row extraction, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer that uses the extracted rate must therefore read the finite route*

$$Q,\ D,\ W,\ R,\ G \longmapsto S$$

before treating the requested convergence as Real-facing evidence.

Proof. Project the carrier of Definition 9163.1. The rows Q, D, W, R, G are the only pre-seal data available to the extractor.

The tolerance request Q is first translated into the dyadic budget D. The finite window row W then displays exactly which stream observations are inspected, and the regular rational readback row R records the observations seen on those windows.

The convergence-rate row G is downstream of those displayed rows. It returns an index for the request already budgeted by D and read through W and R; it cannot be replaced by a selected limit or an unbounded scan.

Finally the Real seal row S and structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no further coordinate, the listed obligation groups exhaust the local $\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtractor}^{\uparrow}}$ surface. □

Theorem 9163.3 (Real convergence-rate extractor Cauchy handoff)**.** *Every accepted* RealConvergenceRateExtracto *consumer read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal or Completion-facing handoff factors through the same dyadic-window-readback route used by the* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *packet: a tolerance request Q, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget D, a* $\mathsf{StreamName}^{\uparrow}$ *finite window W, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and a convergence-rate row G precede the terminal Real seal S. Thus convergence evidence for Real and Completion consumers depends on displayed finite modulus data, not on selected limits or quotient representatives.*

Proof. Start with an accepted carrier $E = (Q, D, W, R, G, S, H, C, P, N)$. The terminal seal S is not public until the request row, dyadic budget, finite windows, regular rational readback, and convergence-rate row have all been displayed.

Compare this route with chapter 5923. Both packets force a Real-facing consumer through dyadic tolerance data, finite stream windows, and regular rational readback before any Real seal is available.

The extractor's row G is the finite modulus output attached to that route. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed dependency, so a downstream Real or Completion consumer has no coordinate from which to import

a selected limit, quotient representative, countable-choice modulus, host real equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or Completion-facing handoff without displaying a tolerance request, DyadicRatCore budget, StreamName finite window, RegSeqRat readback, and convergence-rate row refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishopdiagonalcauchytelescope: $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ is not bijective to any listed sibling because it binds a tolerance request to a finite extracted convergence-rate row before the Real seal, while the siblings separately supply budgets, windows, readbacks, seals, and diagonal-tail routes.

Closure status for $\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{RealConvergenceRateExtractor}^{\uparrow}$ packet is generated from a tolerance request, DyadicRatCore budget, StreamName finite windows, RegSeqRat readback, convergence-rate row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9163.1, Definition 9163.2, and Definition 9163.3 seed the finite convergence-rate route through dyadic budgets, stream windows, regular rational readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealConvergenceRateExtractor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient real equality, countable-choice moduli, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic budget exposure, finite-window readback, regular rational handoff, convergence-rate exactness, Real seal handoff, structural transport, and ledger non-escape for BEDC.Derived.RealConvergenceRateExtractorUp.

Chapter 9164

A Concrete Naming Certificate for $\mathsf{PrimaryDecomposition}^{\uparrow}$

Following Atiyah and Macdonald, Introduction to Commutative Algebra, the $\mathsf{PrimaryDecomposition}^{\uparrow}$ packet records the finite row data by which a noetherian algebra source displays an ideal or submodule as an intersection of primary components. It sits after the chain-stabilization surfaces of $\mathsf{NoetherianModule}^{\uparrow}$ and $\mathsf{NoetherianRing}^{\uparrow}$ in chapter 3648 and chapter 3637, the ideal membership surface of $\mathsf{Ideal}^{\uparrow}$ in chapter 81, the finite reduction witness $\mathsf{GroebnerBasis}^{\uparrow}$ of chapter 2197, the finite basis route $\mathsf{HilbertBasis}^{\uparrow}$ of chapter 3472, and the residue recombination sibling $\mathsf{ChineseRemainder}^{\uparrow}$ of chapter 8224. The packet exports only the displayed decomposition route; it is not a host quotient-ideal type, a uniqueness theorem, an ambient radical oracle, or a selected associated-prime principle.

Definition 9164.1 (Primary decomposition carrier)**.** *A* $\mathsf{PrimaryDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{PrimDec}(R, M, I, B, G, L, E, H, C, Q, N)$$

with the following rows.

- *R is the* $\mathsf{NoetherianRing}^{\uparrow}$ *source row for the ambient commutative ring operations and ascending-chain control.*
- *M is the optional* $\mathsf{NoetherianModule}^{\uparrow}$ *row used when the decomposed object is a submodule rather than an ideal.*
- *I is the* $\mathsf{Ideal}^{\uparrow}$ *or submodule-membership row naming the object whose decomposition is being read.*
- *B is the* $\mathsf{HilbertBasis}^{\uparrow}$ *finite generator row for the displayed ideal or submodule presentation.*
- *G is the* $\mathsf{GroebnerBasis}^{\uparrow}$ *finite reduction row used only to replay membership in the listed components.*
- *L is the finite component list. Each entry records a component carrier, a primaryness witness row, and the radical-name row used by the classifier below.*
- *E is the exactness ledger: it stores the two finite membership directions between the original object I and the intersection of the component rows in L.*
- *H, C, Q, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over the listed sibling packets, and local* $\mathsf{NameCert}_{\mathsf{PrimaryDecomposition}^{\uparrow}}$ *row.*

The classifier compares two carriers only when their ambient $\mathsf{NoetherianRing}^{\uparrow}$ *and optional* $\mathsf{NoetherianModule}^{\uparrow}$ *rows classify together, their original object rows classify together, their component lists match by displayed component rows, and their exactness ledgers replay the same two membership directions. It has no coordinate for uniqueness of minimal decompositions, equality of radicals except the radical-name rows stored in* L*, host quotient ideals, associated-prime selection, or a decomposition theorem outside the finite packet.*

Lean4 (definition) `BEDC.Derived.PrimaryDecompositionUp.PrimaryDecompositionUp`

Theorem 9164.2 (Primary decomposition NameCert obligations)**.** *For every accepted* $\mathsf{PrimaryDecomposition}^{\uparrow}$ *carrier* $\mathsf{PrimDec}(R, M, I, B, G, L, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PrimaryDecomposition}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, noetherian ring compatibility through* R*, optional noetherian module compatibility through* M*, ideal or submodule source exposure through* I*, finite generator access through* B*, finite reduction replay through* G*, primary component-list exposure through* L*, intersection exactness through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance non-escape through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 9164.1. The source rows R and M determine the algebraic classifier before any component can be read.

The object row I is then inspected through the finite generator and reduction surfaces B and G. These rows expose the same membership problem to every component of L, but they do not introduce quotient ideals or a host decomposition primitive.

The component list L stores only displayed primaryness and radical-name rows, while E stores the two finite membership directions between I and the displayed intersection. Transport, continuation, provenance, and local naming repack those rows. Since no other coordinate appears in the carrier, the local certificate cannot claim uniqueness, associated-prime selection, or radical equality beyond the listed classifier rows. □

Theorem 9164.3 (Primary decomposition localization route)**.** *Let* $\mathsf{PrimDec}(R, M, I, B, G, L, E, H, C, Q, N)$ *be an accepted* $\mathsf{PrimaryDecomposition}^{\uparrow}$ *carrier. Any downstream local computation that reads a component of the decomposition must first expose the ambient* $\mathsf{NoetherianRing}^{\uparrow}$ *row* R*, the optional* $\mathsf{NoetherianModule}^{\uparrow}$ *row* M *when present, the original* $\mathsf{Ideal}^{\uparrow}$ *or submodule row* I*, and the finite component-list row* L*. The localized read may use only the selected component row, its stored primaryness witness, its radical-name row, the membership replay supplied by* G*, and the exactness ledger* E*.*

Proof. Start from R and M. They bind the algebraic source and chain-control surface for every later component read.

Next expose I, B, and G. The finite generator and reduction rows are the only route by which membership in the original object or in a listed component is replayed.

Finally choose a displayed component position in L and use the corresponding rows together with E. This yields a local component computation inside the same finite decomposition packet. The route does not pass through host quotient ideals, a selected localization theorem, or a hidden associated-prime construction. □

Theorem 9164.4 (Primary decomposition NameCert route)**.** *Every public consumer of an accepted* $\mathsf{PrimaryDecomposition}^{\uparrow}$ *packet that claims a primary decomposition must read the route*

$$\mathsf{NoetherianRing}^{\uparrow},\ \mathsf{NoetherianModule}^{\uparrow},\ \mathsf{Ideal}^{\uparrow},\ \mathsf{HilbertBasis}^{\uparrow},\ \mathsf{GroebnerBasis}^{\uparrow} \longrightarrow \mathsf{NameCert}_{\mathsf{PrimaryDecomposition}^{\uparrow}}.$$

The exported object is exactly the finite component list L, the exactness ledger E, and the structural rows H, C, Q, N. The route may be compared with ChineseRemainder$^{\uparrow}$ *only as a finite recombination sibling:* PrimaryDecomposition$^{\uparrow}$ *stores intersection components and primaryness witnesses, while* ChineseRemainder$^{\uparrow}$ *stores residue rows and coprime recombination data.*

Proof. Read the noetherian source row R, and read M when the carrier decomposes a submodule. These rows provide chain-control context but do not by themselves export a decomposition.

Open the object row I, the finite basis row B, and the reduction row G. They supply the membership and generator surface over which component membership can be checked.

The public decomposition is then exactly L together with E. The remaining rows H, C, Q, N transport, replay, record provenance, and name that same finite packet. The comparison with ChineseRemainder$^{\uparrow}$ is therefore only a sibling boundary: recombination data and component-intersection data remain different carrier coordinates. □

Falsifiable prediction: Every PrimaryDecomposition$^{\uparrow}$ consumer that claims a primary decomposition must exhibit the noetherian ring row, optional noetherian module row, ideal or submodule source row, Hilbert-basis row, Groebner reduction row, finite component list, exactness ledger, transport, continuation, provenance, and local naming row.

Independence witness: noetherian-module, noetherian-ring, ideal, groebner-basis, hilbert-basis, chinese-remainder: PrimaryDecomposition$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a noetherian source, a decomposed object row, finite generator and reduction witnesses, primary component rows, radical-name classifier rows, and a two-sided intersection exactness ledger.

Closure status for NameCert$_{\text{PrimaryDecomposition}^{\uparrow}}$.

Constructive story: A PrimaryDecomposition$^{\uparrow}$ packet is generated from NoetherianRingUp and optional NoetherianModuleUp source rows, an IdealUp or submodule source row, HilbertBasisUp finite generator data, GroebnerBasisUp finite reduction traces, a finite primary component list with radical-name rows, a two-sided intersection exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PrimaryDecomposition}^{\uparrow}}$, seed)

Scope closed: Definition 9164.1, Definition 9164.2, Definition 9164.3, and Definition 9164.4 seed the finite primary-decomposition route over NoetherianRingUp, NoetherianModuleUp, IdealUp, HilbertBasisUp, GroebnerBasisUp, ChineseRemainderUp-adjacent recombination boundaries, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{PrimaryDecomposition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close uniqueness of primary decompositions, radical equality beyond the displayed classifier rows, Lean verification, bridge checking, host quotient ideals, associated-prime selection, or an ambient decomposition theorem.

Upgrade path: Obligation closure requires checked carrier admission, noetherian source compatibility, component-list primaryness, radical-name classifier stability, exactness-ledger coverage, localized read non-escape, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.PrimaryDecompositionUp.

Chapter 9165

A Concrete Naming Certificate for $\mathsf{CompactOscillationBudget}^{\uparrow}$

Following Bishop and Bridges on compact metric spaces and constructive uniform continuity, the $\mathsf{CompactOscillationBudget}^{\uparrow}$ packet records the finite-net oscillation rows used before a continuous map on a compact source may expose a uniform-modulus or Real-facing readback. It sits beside the compact metric source of chapter 129, the continuous-map row of chapter 130, the uniform-modulus handoff of chapter 369, the dyadic tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite compact-net oscillation budget; it does not assert host suprema, open-cover compactness, selected infinite subcovers, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9165.1 (Compact oscillation budget carrier)**.** *A* $\mathsf{CompactOscillationBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, N, O, D, Q, R, U, H, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the compact finite-net cells inspected by the packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose local graph data are being tested on those cells.*
3. *N is the finite compact-net row, listing the displayed centers, radii, and cell-membership witnesses.*
4. *O is the finite oscillation row, assigning to each displayed cell the pointwise oscillation bound read from F over K.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested precision with the cellwise oscillation rows.*
6. *Q is the finite budget ledger folding the cellwise oscillation rows into one displayed compact budget.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the folded rational budget.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after K, F, N, O, D, Q, R have been displayed.*
9. *H records componentwise* hsame *transport for the compact source, map, net, oscillation, dyadic, budget, readback, handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from compact-net cells and continuous-map oscillation rows through dyadic comparison, regular readback, and the terminal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationBudget}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, N, O, D, Q, R, U, H, C, P, L$. It has no coordinate for host suprema of oscillation, open-cover compactness, selected infinite subcovers, quotient function equality, countable choice, or ambient real completeness.

Theorem 9165.2 (Compact oscillation budget NameCert obligations)**.** *For every accepted* CompactOscillationBudg *carrier*

$$B = (K, F, N, O, D, Q, R, U, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationBudget}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, continuous-map admission through F, finite compact-net exposure through N, pointwise cell oscillation rows through O, dyadic tolerance comparison through D, folded budget exposure through Q, regular rational readback through R, terminal uniform-modulus or Real handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 9165.1. Its analytic rows are exactly the compact source K, the continuous-map row F, the finite net N, the cellwise oscillation row O, the dyadic tolerance D, the folded budget Q, the regular readback R, and the terminal handoff U.

Read the finite-net route in order. The row N lists only displayed cells over K, and O records only the pointwise oscillation rows supplied by F on those cells. The dyadic row D and budget ledger Q compare those finitely many rows without taking a host supremum.

The readback and handoff rows R, U are downstream of that finite ledger. The structural rows H, C, P, L transport, replay, source, and locally name the same packet, so the excluded open-cover, selected-subcover, quotient, choice, and ambient-completeness data are not part of the NameCert surface. □

Theorem 9165.3 (Compact oscillation budget finite-net route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactOscillationBudget}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, N, O, \mathsf{DyadicRatCore}^{\uparrow}, Q, \mathsf{RegSeqRat}^{\uparrow}, U, H, C, P, L$$

before using the exported compact oscillation budget. Thus the route is determined by compact-net cells, pointwise oscillation rows, dyadic tolerance comparison, regular rational readback, and the terminal handoff, rather than by host suprema or an ambient compactness theorem.

Proof. Start from the NameCert obligations of Definition 9165.2. They force every consumer through the compact source, continuous-map row, finite net, oscillation ledger, dyadic comparison, budget fold, and regular readback before U is public.

The compact finite-net cells are the only coverage data in the carrier. A consumer that asks for a different cover, a selected infinite subcover, or a host supremum of oscillation would need a coordinate absent from B.

Finally, transport and replay preserve the same finite dependency route through H, C, P, L. Therefore the exported budget is a BEDC finite NameCert route to $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, not an external compactness or supremum principle. □

Falsifiable prediction: An accepted $\mathsf{CompactOscillationBudget}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ without displaying the compact metric source, continuous-map row, finite compact-net cells, pointwise oscillation rows, dyadic tolerance row, folded budget ledger, regular rational readback, transport, replay, provenance, and local NameCert row.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{CompactOscillationBudget}^{\uparrow}$ because this packet jointly binds compact finite-net cells, pointwise oscillation rows, dyadic tolerance comparison, a folded finite budget, regular rational readback, terminal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationBudget}^{\uparrow}$ packet is generated from CompactMetric finite-net cells, a ContinuousMap row, pointwise oscillation rows, DyadicRatCore tolerance comparison, a folded finite budget ledger, RegSeqRat readback, UniformModulus or RealUp handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOscillationBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9165.1, Definition 9165.2, and Definition 9165.3 seed the finite compact-net oscillation route from compact and continuous rows into uniform-modulus and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOscillationBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host suprema of oscillation, open-cover compactness, selected infinite subcovers, quotient function equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-map exposure, finite-net exposure, oscillation-row exactness, dyadic budget comparison, regular readback, terminal handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactOscillationBudgetUp.

Chapter 9166

A Concrete Naming Certificate for $\mathsf{CompactConvergenceTheorem}^{\uparrow}$

Following Kelley and Bishop–Bridges on compact convergence, the $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ packet defines a finite BHist route over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$. It sits after the compact metric source of chapter 129, the continuous-map surface of chapter 130, the uniform-limit sibling of chapter 463, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real seal of chapter 38. The packet records the compact-source uniform-convergence theorem as finite BEDC data; it does not import open-cover compactness, host function-space completion, selected suprema, or ambient Real completeness.

Definition 9166.1 (Compact convergence theorem carrier)**.** *A* $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{CompConv}(K, F, U, W, R, D, S, H, C, P, N)$$

with the following rows.

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric carrier and the finite compactness evidence used by the theorem packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous maps on the compact source.*
3. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *row, carrying the uniform Cauchy or uniform convergence evidence for the family F over K.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect only bounded prefixes of the convergence data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected windows into regular rational observations.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius and comparison slack used by the uniform-limit route.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after K, F, U, W, R, D have been displayed.*

8. *H records componentwise* hsame *transport for compact sources, continuous maps, uniform-limit evidence, stream windows, regular rational readbacks, dyadic budgets, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from the compact source and map-family rows through U, W, R, D to the Real seal row S.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactConvergenceTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactConvergenceTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, W, R, D, S, H, C, P, N$. It has no coordinate for an open-cover compactness proof, a host function-space completion, a selected supremum norm, a quotient of maps, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 9166.2 (Compact convergence theorem NameCert obligations)**.** *For every accepted* $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ *carrier* $\mathsf{CompConv}(K, F, U, W, R, D, S, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{CompactConverge}}$ *surface consists exactly of compact-source admission through K, continuous-map family exposure through F, uniform-limit evidence through U, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgeting through D, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9166.1. The rows $K, F, U, W, R, D, S, H, C, P, N$ are the only fields available to the local classifier.

The compact source K and continuous-map family F determine the theorem's domain and maps. The uniform-limit row U supplies the displayed convergence evidence, but it must still pass through the finite window row W, the regular rational readback row R, and the dyadic tolerance row D before any Real-facing seal is visible.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier contains no open-cover compactness proof, host function-space completion, selected supremum, quotient-map equality, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 9166.3 (Compact convergence uniform-limit route)**.** *Every accepted* $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ *consumer that reads a compact-source uniform convergence theorem must factor through the route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the public read sees a compact metric source, a continuous-map family, uniform-limit evidence, finite observation windows, regular rational readback, and dyadic tolerance data before the terminal Real seal.

Proof. Begin with an accepted carrier $\mathsf{CompConv}(K, F, U, W, R, D, S, H, C, P, N)$. The compact source and continuous-map rows are the only entries that name the domain and map family.

The uniform-limit row U is downstream of those rows. It is not public until the finite $\mathsf{StreamName}^{\uparrow}$ windows W, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D have displayed the inspected convergence data.

Finally the $\mathsf{Real}^{\uparrow}$ seal S and structural rows H, C, P, N preserve the same route. Thus the theorem packet exports a finite uniform-limit path, not a hidden compactness theorem, selected supremum, or host completion object. □

Theorem 9166.4 (Compact convergence Real-seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *boundary of an accepted* $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ *packet is exactly the seal row* S *reached from* K, F, U, W, R, D*. A downstream Real or Completion consumer may use* S *only together with the displayed compact source, continuous-map family, uniform-limit row, finite stream windows, regular rational readback, dyadic tolerance budget, transport, replay, provenance, and local naming rows stored in the same packet.*

Proof. Read the carrier rows in order. The seal row S has no independent source: it is reached only after the compact metric row, continuous-map family row, uniform-limit row, finite window row, regular rational readback row, and dyadic tolerance row have been projected.

Componentwise transport H and replay C keep that dependency visible, while P and N record provenance and local naming for the same packet. Since there is no further coordinate for open covers, ambient completeness, or selected suprema, any Real or Completion consumer must remain inside this displayed Real-seal boundary. □

Falsifiable prediction: An accepted $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying the compact metric source, continuous-map family, uniform-limit row, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance data refutes the carrier surface.

Independence witness: compactmetric, continuousmap, uniformlimit, streamname, regseqrat, dyadicratcore, real: $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the compact source, map family, uniform-limit evidence, finite observation windows, regular rational readback, dyadic tolerance budget, Real seal, transport, replay, provenance, and local naming as one theorem packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactConvergenceTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactConvergenceTheorem}^{\uparrow}$ packet is generated from CompactMetricUp, ContinuousMapUp, UniformLimitUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactConvergenceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9166.1, Definition 9166.2, Definition 9166.3, and Definition 9166.4 seed the compact-source uniform-convergence route through finite windows, regular rational readback, dyadic tolerance data, and a Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactConvergenceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host function-space completion, selected suprema, quotient-map equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source compatibility, continuous-map family exposure, uniform-limit route exactness, finite-window readback, dyadic tolerance preservation, Real-seal nonescape, componentwise transport, provenance, and local naming for BEDC.Derived.CompactConvergenceTheoremUp.

Chapter 9167

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ packet records the finite route by which the diagonal representative of Definition 9238.3 becomes a Real-facing seal. It sits after the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real packet of chapter 38. The packet names the diagonal representative, its window choice, its regularity and tail comparison data, and its Real seal; it is not a selected limit, quotient-stream equality, countable-choice modulus, or ambient completeness theorem.

Definition 9167.1 (Regular Cauchy diagonal-limit seal carrier)**.** *A* $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{DiagSeal}(Q, M, W, R, D, L, S, H, C, P, N)$$

with the following rows.

1. *Q is the precision-request row naming the Real-completion consumer and the requested tolerance.*
2. *M is the displayed regular Cauchy modulus row used to choose the diagonal stage for that request.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected diagonal and tail windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations obtained from W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic budget for diagonal regularity and tail comparison.*
6. *L is the diagonal representative row, storing the finite readback of the representative described in Definition 9238.3.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after Q, M, W, R, D, L have been displayed.*
8. *H records componentwise* hsame *transport for precision requests, moduli, stream windows, regular rational readbacks, dyadic budgets, diagonal representatives, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from the precision request through M, W, R, D, L to the Real seal row S.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, W, R, D, L, S, H, C, P, N$. It has no coordinate for a hidden limit selection, quotient equality of streams, countable-choice modulus, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 9167.2 (Regular Cauchy diagonal-limit seal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ *carrier* $\mathsf{DiagSeal}(Q, M, W, R, D, L, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}}$ *surface consists exactly of precision-request admission through Q, regular Cauchy modulus exposure through M, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgeting through D, diagonal representative exposure through L, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9167.1. The rows $Q, M, W, R, D, L, S, H, C, P, N$ are the complete local surface.

The request Q determines the tolerance being answered. The modulus row M chooses the diagonal stage, the finite window row W displays what is inspected, and the readback and dyadic rows R, D record the regular rational and tolerance data used in the diagonal estimate.

The diagonal representative row L is downstream of those visible rows, and the Real seal S is downstream of L. Transport, replay, provenance, and local naming preserve the same dependency. Since no carrier coordinate stores a selected limit, quotient stream, countable choice, host real equality, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 9167.3 (Regular Cauchy diagonal-limit seal route)**.** *Every accepted* $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ *consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must pass through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

after displaying the precision request, the regular Cauchy modulus row, and the diagonal representative row. Thus the diagonal-limit construction is consumed as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, *and* NameCert *data.*

Proof. Start with an accepted carrier $\mathsf{DiagSeal}(Q, M, W, R, D, L, S, H, C, P, N)$. The precision request and modulus rows identify the diagonal stage used for the read.

The finite stream window W and regular rational readback R expose the actual observations, while D supplies the dyadic tolerance budget for the diagonal regularity and tail comparison checks. The representative row L is therefore a finite readback row, not a hidden limit.

Finally S is the only Real-facing seal in the carrier, and H, C, P, N keep the same route visible to downstream consumers. The route cannot be shortened through quotient-stream equality, countable choice, or ambient Real completeness because none of those objects appears in the carrier. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying a precision request, regular Cauchy modulus, StreamName window, RegSeqRat readback, DyadicRatCore budget, and diagonal representative row refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a precision request, regular Cauchy modulus, finite diagonal window, regular rational readback, dyadic tolerance budget, diagonal representative, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}$ packet is generated from a precision request, regular Cauchy modulus row, StreamNameUp finite windows, RegSeqRatUp readback, DyadicRatCoreUp tolerance data, diagonal representative row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9167.1, Definition 9167.2, and Definition 9167.3 seed the diagonal representative route from finite StreamName windows through RegSeqRat readback and DyadicRatCore tolerance data into a Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimitSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable-choice moduli, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, precision-request exposure, regular Cauchy modulus compatibility, finite-window readback, diagonal representative exactness, dyadic tolerance preservation, Real-seal nonescape, componentwise transport, provenance, and local naming for BEDC.Derived.RegularCauchyDiagonalLimitSealUp.

Chapter 9168

A Concrete Naming Certificate for $\mathsf{CauchyFilterTelescope}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki–Kelley language of filters and filter bases, the $\mathsf{CauchyFilterTelescope}^{\uparrow}$ packet records the finite telescope that reads a Cauchy-filter presentation back as regular Cauchy sequence data before a Real or Completion consumer inspects it. It sits between the Cauchy-filter source of chapter 333, the finite basis generator of chapter 1501, the regular-filter handoff of chapter 1445, the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet is a finite bridge from displayed filter data to regular sequence readback; it is not a quotient of filters, not a selected limit, not host equality of reals, not countable choice, and not ambient completeness.

Definition 9168.1 (Cauchy filter telescope carrier)**.** *A* $\mathsf{CauchyFilterTelescope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, B, G, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing only displayed finite filterbase requests and Cauchy-window evidence.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row that lists the finite generator windows read from F.*
3. *G is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *handoff row, retaining the directed tail-meet data that regularizes the basis.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used by the readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback windows and radii.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after F, B, G, S, R, D have been replayed.*
8. *H records componentwise* hsame *transport for all displayed source, basis, handoff, stream, readback, dyadic, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the filter source through basis generation, regular-filter handoff, stream windows, regular rational readback, dyadic tolerances, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterTelescope}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable-choice schedules, host real equality, or ambient Real completeness.

Theorem 9168.2 (Cauchy filter telescope NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterTelescope}^{\uparrow}$ *carrier $T = (F, B, G, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterTelescope}^{\uparrow}}$ *surface consists exactly of source Cauchy-filter admission, basis exposure, regular-filter handoff, stream-window exposure, regular rational readback, dyadic tolerance exposure, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 9168.1. The displayed tuple contains precisely the CauchyFilter source row, CauchyFilterBasis row, RegularCauchyFilter handoff, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, and structural rows.

The replay row C orders those rows. A consumer must first inspect the filter source F, then the basis row B, then the regular-filter handoff G before it can read stream windows, rational readback, dyadic tolerances, or the Real seal.

Transport, provenance, and local naming add no further mathematical coordinates. Since the carrier contains no quotient-filter, selected-limit, choice, host-equality, or ambient-completeness row, the listed rows exhaust the NameCert obligations. □

Theorem 9168.3 (Cauchy filter telescope regular-sequence handoff)**.** *Every accepted* $\mathsf{CauchyFilterTelescope}^{\uparrow}$ *read from a Cauchy-filter presentation to a regular Cauchy sequence factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCo}$$

The handoff exposes the same rows $F, B, G, S, R, D, E, H, C, P, N$ to downstream Real and Completion consumers and exports no selected limit, quotient-filter equality, countable-choice selector, host equality, or ambient completeness principle.

Proof. Start with the NameCert obligations of Definition 9168.2. They force the source filter row, basis row, regular-filter handoff, stream schedule, rational readback, and dyadic tolerance ledger to remain visible before the Real seal can be consumed.

The basis row B supplies only displayed generator windows from F. The regular-filter row G then keeps the directed tail-meet data attached to those windows, so the readback cannot skip directly from a filter presentation to a Real endpoint.

Finally S, R, D, E provide the regular sequence readback and seal. The structural rows H, C, P, N preserve the same order, and no carrier coordinate provides any of the refused quotient, choice, equality, or completeness endpoints. □

Theorem 9168.4 (Cauchy filter telescope completion boundary)**.** *A downstream Completion or Real consumer of an accepted* $\mathsf{CauchyFilterTelescope}^{\uparrow}$ *packet may read only the displayed telescope rows $F, B, G, S, R, D, E, H, C, P, N$. The packet therefore supplies a finite boundary between Cauchy-filter presentations and regular Cauchy sequence readback; it does not assert that every filter has a selected limit, that filters are quotiented by host equality, that countable choices of basis elements exist, or that the ambient Real object is complete.*

Proof. Apply the regular-sequence handoff. It places every Real or Completion read behind the finite telescope route from F through E.

The completion-facing consumer then receives only the rows named by the carrier. The source filter, basis, regular handoff, stream windows, regular rational readback, dyadic ledger, and Real seal are all displayed before the structural rows transport and name them.

Because the carrier inventory has no selected-limit, quotient-filter, countable-choice, host-equality, or ambient-completeness coordinate, the boundary is a finite readback interface and not a completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterTelescope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterTelescope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9168.1, Definition 9168.2, Definition 9168.3, and Definition 9168.4 seed the finite telescope from CauchyFilter and CauchyFilterBasis rows through RegularCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterTelescope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis exposure, regular-filter handoff, stream-window readback, dyadic tolerance exactness, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterTelescopeUp.

Constructive story: A $\mathsf{CauchyFilterTelescope}^{\uparrow}$ packet is generated from a CauchyFilter source row, a CauchyFilterBasis row, a RegularCauchyFilter handoff, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9169

A Concrete Naming Certificate for $\mathsf{TotallyBoundedInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{TotallyBoundedInterval}^{\uparrow}$ packet records the constructive fact that a located real interval admits finite meshes at each displayed positive radius. It sits after the real-name carrier of chapter 38, the regular rational readback of chapter 301, the dyadic mesh arithmetic of chapter 291, the located interval membership surface of chapter 394, and the compact metric finite-net interface of chapter 129. The packet names interval total boundedness as finite NameCert data; it is not open-cover compactness, a selected subcover, quotient equality of real names, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9169.1 (Totally bounded interval carrier)**.** *A* $\mathsf{TotallyBoundedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, L, D, Q, M, F, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper endpoint histories, each sealed through* $\mathsf{Real}^{\uparrow}$ *only after its regular rational readback has been displayed.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *membership row, exposing interval entry and endpoint comparison only through displayed locatedness data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-radius row naming the positive dyadic radius requested by the total-boundedness read.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint and mesh histories.*
5. *M is the finite mesh schedule, recording the displayed dyadic centers between A and B at radius D.*
6. *F is the finite* $\mathsf{ProbeBundle}$ *net row whose probes name exactly those displayed centers.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint, center, and radius histories after Q and M are exposed.*
8. *H records componentwise* hsame *transport for endpoints, located membership, dyadic radius, regular readback, mesh schedule, finite net, real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from endpoint histories to located membership, from located membership to dyadic mesh extraction, from mesh extraction to regular readback, and from readback to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows.

Lean4 (statement) `BEDC.Derived.TotallyBoundedIntervalUp`

Theorem 9169.2 (Totally bounded interval NameCert obligations)**.** *For every accepted* $\mathsf{TotallyBoundedInterval}^{\uparrow}$ *carrier $T = (A, B, L, D, Q, M, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedInterval}^{\uparrow}}$ *surface consists exactly of endpoint admission through A, B, located-interval membership through L, dyadic mesh-radius admission through D, regular rational readback through Q, finite mesh scheduling through M, finite* ProbeBundle *net exposure through F, Real seal nonescape through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9169.1. The endpoint histories A, B and the located membership row L are visible before a mesh row can be consumed, so interval membership is not read from a host interval predicate.

The dyadic and regular rows determine the mesh. Row D supplies the displayed radius, Q supplies the regular rational readback of endpoint and radius histories, and M records the finite dyadic centers admitted at that radius. The finite net row F then packages those centers as a ProbeBundle.

The terminal rows preserve the same route. The $\mathsf{Real}^{\uparrow}$ seal R is reached only after the mesh and readback rows are displayed, while H, C, P, N transport, replay, source, and name the same packet. No excluded compactness, subcover, quotient, trichotomy, choice, or ambient completeness principle has a carrier coordinate. □

Theorem 9169.3 (Totally bounded interval net extraction)**.** *Every completion-facing or compact-metric consumer of an accepted* $\mathsf{TotallyBoundedInterval}^{\uparrow}$ *packet receives a finite interval net only through*

$$A, B, L, D, Q, M, F, R, H, C, P, N.$$

Thus the exported net is the finite ProbeBundle *of dyadic centers read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows; it is not a host Heine–Borel theorem, a selected finite subcover, or an ambient compactness assertion.*

Proof. Use Definition 9169.2. It identifies the full local surface of an accepted packet, so a consumer cannot reach the finite net before the endpoint, locatedness, dyadic radius, readback, and mesh rows have been displayed.

The extraction is finite. The mesh schedule M lists the dyadic centers admitted at radius D, and F repackages precisely those centers as a ProbeBundle net. The endpoint and membership rows A, B, L keep the centers inside the located interval route.

Finally the readback and seal rows constrain downstream use. Rows Q and R route the mesh through regular rational and real-name data, while H, C, P, N preserve the same displayed path for compact-metric and completion consumers. □

Falsifiable prediction: A $\mathsf{TotallyBoundedInterval}^{\uparrow}$ consumer is refuted whenever it reads an interval net without exposing the RealUp endpoint histories, LocatedIntervalUp membership row, DyadicRatCore mesh radius, RegSeqRat readback, finite ProbeBundle centers, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{TotallyBoundedInterval}^{\uparrow}$ packet is generated from RealUp endpoint histories, a LocatedIntervalUp membership row, a DyadicRatCore mesh-radius row, RegSeqRat readback, a finite dyadic mesh schedule, finite ProbeBundle nets, RealUp seal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9169.1, Definition 9169.2, and Definition 9169.3 seed the constructive finite-net route for located real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedInterval}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.TotallyBoundedIntervalUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, quotient equality of real names, trichotomy, countable choice, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, located-interval membership, dyadic mesh extraction, regular readback, finite ProbeBundle coverage, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TotallyBoundedIntervalUp.

Chapter 9170

A Concrete Naming Certificate for LebesgueNumberMesh$^{\uparrow}$

The LebesgueNumberMesh$^{\uparrow}$ packet is grounded in the Lebesgue number lemma for compact metric spaces: a finite cover of a compact metric source yields a positive mesh radius whose balls refine the cover. In BEDC form the packet records only finite cover mesh, compact-net, local-radius, rational lower-bound, and uniform-modulus handoff data. It realizes the compact-uniform-continuity route of chapter 9237 after the compact metric source of chapter 129, the continuous-map local-radius source of chapter 130, the uniform-modulus consumer of chapter 369, and the rational lower-bound fold surface of chapter 37. The packet is not a host compactness theorem, a chosen open-cover subcover, a least radius, quotient map equality, countable choice, or ambient metric completion.

Definition 9170.1 (Lebesgue number mesh carrier)**.** *A* LebesgueNumberMesh$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, V, G, O, B, Q, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose finite net is being inspected.*
2. *V is the finite cover mesh row, recording the cover cells and the mesh-to-cover incidence data.*
3. *G is the compact-net* ProbeBundle *row whose probes name the finite centers used to test the cover.*
4. *O is the local radius ledger, assigning to each displayed probe the cover radius admitted by the local continuity or cover-membership read.*
5. *B is the* Rat$^{\uparrow}$ *lower-bound fold over the finitely many positive local radii.*
6. *Q is the mesh-refinement witness saying that the folded rational bound refines the cover mesh.*
7. *U is the* UniformModulus$^{\uparrow}$ *handoff row exposed only after K, V, G, O, B, Q have been displayed.*
8. *H records componentwise* hsame *transport for compact source, cover mesh, compact net, local radii, rational fold, refinement, uniform handoff, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from compact-net exposure through local-radius reads, rational lower-bound folding, mesh refinement, and uniform-modulus handoff.*
10. *P is the* Pkg *provenance over* LebesgueNumberMesh$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberMesh}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through this finite mesh and lower-bound surface.

Lean4 (statement) `BEDC.Derived.LebesgueNumberMeshUp`

Theorem 9170.2 (Lebesgue number mesh NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueNumberMesh}^{\uparrow}$ *carrier* $L = (K, V, G, O, B, Q, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberMesh}^{\uparrow}}$ *surface consists exactly of compact-metric source admission through* K*, finite cover mesh exposure through* V*, compact-net* ProbeBundle *exposure through* G*, local radius ledger exposure through* O*,* $\mathsf{Rat}^{\uparrow}$ *lower-bound folding through* B*, mesh-refinement exposure through* Q*,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through* U*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 9170.1. The compact source K, finite cover mesh V, and compact-net row G are displayed before any lower-bound or uniform-modulus row is available.

Read the radius data next. The local radius ledger O attaches one positive rational radius to each displayed probe, and the $\mathsf{Rat}^{\uparrow}$ fold B produces the common lower bound from that finite family. The mesh-refinement row Q records that this bound makes each inspected mesh ball land inside a cover cell.

The handoff row U is therefore downstream of finite cover, compact-net, local-radius, and rational-fold data. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route, so the packet exports no host compactness theorem, selected subcover, least-radius principle, quotient map equality, choice functional, or ambient completion. □

Theorem 9170.3 (Lebesgue number mesh uniform-modulus extraction)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{LebesgueNumberMesh}^{\uparrow}$ *packet must read the finite route*

$$K, V, G, O, B, Q, U, H, C, P, N$$

before consuming the exported $\mathsf{UniformModulus}^{\uparrow}$ *row. The extracted modulus is the tolerance-only handoff obtained from finite cover mesh data, compact-net probes, local radii, and a rational lower-bound fold.*

Proof. Apply Definition 9170.2. It places U after the compact source, finite cover mesh, compact net, local radii, rational fold, and mesh-refinement witness.

The extraction is a finite fold. Each probe in G contributes one radius row in O, and the $\mathsf{Rat}^{\uparrow}$ row B folds these finitely many positive radii into a common lower bound. Row Q then records that this bound refines the finite cover mesh.

Only after those rows are exposed may a compact-uniform-continuity consumer read U as a $\mathsf{UniformModulus}^{\uparrow}$ handoff. Since the carrier contains no host compactness, cover selector, least-radius operator, quotient equality, choice, or ambient completion coordinate, the exported modulus is exactly the finite mesh route described above. □

Falsifiable prediction: A $\mathsf{LebesgueNumberMesh}^{\uparrow}$ consumer is refuted whenever a UniformModulusUp handoff is read before the compact-metric source, finite cover mesh, compact-net ProbeBundle, local radius ledger, RatUp lower-bound fold, mesh-refinement witness, transport, replay, provenance, and local NameCert row are displayed.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberMesh}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LebesgueNumberMesh}^{\uparrow}$ packet is generated from CompactMetric source rows, finite cover mesh rows, compact-net ProbeBundle data, local radius ledgers, RatUp lower-bound folds, mesh-refinement witnesses,

UniformModulus handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LebesgueNumberMesh}\uparrow}$, seed)

Scope closed: Definition 9170.1, Definition 9170.2, and Definition 9170.3 seed the finite Lebesgue-number mesh route for compact-uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{LebesgueNumberMesh}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.LebesgueNumberMeshUp`

Bridge status: none

Not claimed: This chapter does not close host compactness, selected finite subcovers, least Lebesgue numbers, quotient equality of maps, countable choice, ambient completion, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source admission, finite cover mesh exposure, compact-net exposure, local radius ledger exactness, rational lower-bound folding, mesh-refinement nonescape, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueNumberMeshUp.

Chapter 9171

A Concrete Naming Certificate for $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$

Following Moore and Smith on directed nets and Bishop–Bridges constructive completion, the $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ packet records the finite tail-window route by which a displayed directed index row reaches the Real and Completion corridor. It sits after the directed-net source of chapter 3403, the finite StreamName observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal Real seal of chapter 38, and the completion boundary of chapter 1639. The packet exposes a cofinal tail-window handoff without quotient nets, selected cofinal maps, countable choice, host net equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9171.1 (Directed-net tail-window completion carrier)**.** *A* $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, U, W, R, D, E, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the directed-index row, carrying finitely many index names, preorder comparison rows, and displayed common-upper witnesses.*
2. *U is the cofinal tail-window row. For each requested precision it records a displayed late-index threshold and the finite tail slice inspected above that threshold.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row attached to the retained tail slice.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and rational observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic precision, comparison, and tail-radius cells used by the request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, U, W, R, D have been displayed.*
7. *B is the* $\mathsf{Completion}^{\uparrow}$ *handoff row, read through the Bishop regular Cauchy completion boundary and never as an ambient completed object.*
8. *H records componentwise* hsame *transport for the directed-index, cofinal tail-window, StreamName, RegSeqRat, DyadicRatCore, Real, Completion, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through I, U, W, R, D, E, B.*
10. *P is the* Pkg *provenance over* $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, U, W, R, D, E, B, H, C, P, N$. It has no coordinate for quotient nets, selected cofinal maps, countable choice, host net equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9171.2 (Directed-net tail-window completion NameCert obligations)**.** *For every accepted* $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ *carrier $T = (I, U, W, R, D, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}}$ *surface consists exactly of directed-index admission through I, cofinal tail-window exposure through U, StreamName observation through W, RegSeqRat readback through R, DyadicRatCore tolerance exposure through D, Real sealing through E, Completion handoff through B, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9171.1. Its object rows are exactly the directed-index row, cofinal tail-window row, observation window, regular rational readback, dyadic tolerance ledger, Real seal, and Completion handoff.

The directed portion is exhausted by I and U. A completion request first displays the finite index comparisons and common-upper witnesses, then records the late tail slice selected for the requested precision.

The analytic portion is exhausted by W, R, D, E, B. The finite tail slice is observed as StreamName windows, read back as regular rational data, checked against the dyadic tolerance ledger, and only then served to the Real seal and Completion handoff.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no row stores quotient nets, selected cofinal maps, countable choice, host equality, selected limits, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 9171.3 (Directed-net tail-window Real and Completion handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MooreSmithNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{Completion}^{\uparrow}.$$

The handoff uses the displayed cofinal tail-window row as its source and exports no quotient net, selected cofinal map, countable-choice schedule, host net equality, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted packet $T = (I, U, W, R, D, E, B, H, C, P, N)$. The directed-index row I and cofinal tail-window row U are the only source rows available to a Real or Completion consumer.

Read the observation and readback rows next. The StreamName window W is opened on the displayed tail slice, the RegSeqRat row R reads back the same finite observations, and the DyadicRatCore row D records the tolerance arithmetic used by the tail request.

Finally E and B are terminal handoff rows for Real and Completion consumers. The transport, replay, provenance, and local naming rows preserve the same source route, so the consumer never obtains a quotient-net representative, selected cofinal function, countable-choice schedule, host equality, selected limit, or ambient completeness theorem. □

Falsifiable prediction: A Real or Completion consumer that reads a $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ seal while omitting the directed-index row, cofinal tail-window row, StreamName window, RegSeqRat readback, or DyadicRatCore tolerance ledger is outside the accepted packet surface.

Independence witness: moore-smith-net, streamname, regseqrat, dyadicratcore, real, bishop-regular-cauchy-completion: no listed sibling supplies the whole $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ packet because this packet binds directed indices, cofinal tail windows, finite observation, regular rational readback, dyadic tolerance, terminal Real sealing, Completion handoff, transport, replay, provenance, and local naming in one route.

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}$ packet is generated from a directed-index row, cofinal tail-window row, $\mathsf{StreamName}^{\uparrow}$ finite observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{Completion}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9171.1, Definition 9171.2, and Definition 9171.3 seed the finite directed-tail route from Moore–Smith net data into Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirectedNetTailWindowCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected cofinal maps, countable choice, host net equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked directed-index admission, cofinal tail-window exposure, StreamName observation, RegSeqRat readback, DyadicRatCore tolerance exactness, Real and Completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.DirectedNetTailWindowCompletionUp.

Chapter 9172

A Concrete Naming Certificate for $\mathsf{RealTailBudgetHandoff}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ packet turns the tail-budget branch of chapter 9238 into finite NameCert data. It sits after the explicit-modulus and diagonal routes named in that vision, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet exposes a concrete Real-completeness tail-budget handoff without quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9172.1 (Real tail-budget handoff carrier)**.** *A* $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the precision-request row received from the Real-completeness route.*
2. *M is the explicit Cauchy-modulus row that turns Q into a displayed tail threshold.*
3. *T is the tail-budget row, recording the threshold, the finite budget for inspected indices, and the bounded window request answered by the packet.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by that tail budget.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the window values and tail radius.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, M, T, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for precision requests, moduli, tail budgets, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from the Real-completeness precision request through M, T, W, D, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{RealTailBudgetHandoff}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailBudgetHandoff}^{\uparrow}}$ *row.*

The classifier compares packets only through $Q, M, T, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable-choice schedule, host real equality, a detached completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9172.2 (Real tail-budget handoff NameCert obligations)**.** *For every accepted* $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ *carrier $B = (Q, M, T, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTailBudgetHandoff}^{\uparrow}}$ *surface consists exactly of precision-request admission through Q, explicit-modulus exposure through M, tail-budget exposure through T, StreamName window inspection through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9172.1. The listed rows are exactly the precision request, modulus, tail-budget, window, tolerance, readback, Real seal, and structural support of the packet.

The request side is exhausted by Q, M, T. The precision request is converted by the explicit modulus into a displayed threshold, and the tail-budget row records the finite index window that will answer that request.

The readback side is exhausted by W, D, R, E. The StreamName window is inspected under the DyadicRatCore tolerance ledger, the RegSeqRat row reads back the regular rational data, and the Real seal is downstream of that readback.

The structural rows H, C, P, N preserve only this displayed route. Since no row stores quotient streams, selected limits, countable choice, host equality, detached completion objects, or ambient completeness, the local obligation surface contains none of those objects. □

Theorem 9172.3 (Real tail-budget concretization route)**.** *Every Real-completeness consumer that uses an accepted* $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route is triggered by the displayed precision request, explicit modulus, and tail-budget rows Q, M, T; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, detached completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the precision request Q. The explicit modulus row M turns that request into a concrete tail threshold, and T records the finite budget of indices that will be inspected.

The consumer then reads the displayed observation route. The StreamName row W supplies the inspected window, the DyadicRatCore row D supplies the tolerance arithmetic, and the RegSeqRat row R supplies the regular rational readback.

Finally the terminal row E seals the read as a Real-facing handoff. The support rows H, C, P, N transport, replay, source, and name only those rows, so none of the excluded completion principles can enter the concretized route. □

Falsifiable prediction: A Real-completeness route that reads a $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ seal while omitting the precision request, explicit modulus, tail budget, StreamName window, DyadicRatCore tolerance ledger, or RegSeqRat readback is outside the accepted packet surface.

Independence witness: streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole $\mathsf{RealTailBudgetHandoff}^{\uparrow}$ packet because this packet binds a precision request, explicit modulus, finite tail budget, window observation, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and local naming in one concrete Real-completeness route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailBudgetHandoff}^{\uparrow}}$.

Constructive story: A RealTailBudgetHandoff$^{\uparrow}$ packet is generated from a precision request, explicit Cauchy modulus, finite tail-budget row, StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ tolerance ledger, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealTailBudgetHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 9172.1, Definition 9172.2, and Definition 9172.3 seed the finite tail-budget handoff from the Real-completeness vision into StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailBudgetHandoff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, detached completion objects, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked precision-request admission, explicit-modulus exposure, tail-budget exposure, StreamName inspection, DyadicRatCore tolerance exactness, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealTailBudgetHandoffUp.

Chapter 9173

A Concrete Naming Certificate for $\mathsf{BendixsonDulac}^{\uparrow}$

Following Bendixson and Dulac, the $\mathsf{BendixsonDulac}^{\uparrow}$ packet records the finite divergence certificate used to exclude closed trajectories in a planar ODE window before any periodic-orbit conclusion is read. It sits beside the trapping-region certificate of chapter 6911, the displayed ODE source of chapter 193, the dynamical-state surface of chapter 195, the differential-form route of chapter 136, the terminal real-name seal of chapter 38, and the regular-rational readback surface of chapter 301. The packet names only the finite sign, divergence, and boundary-read rows needed for a cycle-exclusion handoff.

Definition 9173.1 (Bendixson–Dulac carrier)**.** *A* $\mathsf{BendixsonDulac}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, D, F, G, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *planar vector-field row, carrying the local state window and the right-hand-side read;*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *local-flow row, recording the finite time-window dynamics consumed by the criterion;*
3. *F is the* $\mathsf{DiffForm}^{\uparrow}$ *divergence-form row for the displayed Dulac multiplier and vector field;*
4. *G is the finite sign ledger asserting that* $\operatorname{div}(gX)$ *has one displayed sign on the inspected region;*
5. *S is the boundary-separation row recording where the sign ledger is read relative to the planar window boundary;*
6. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback row for the coefficient and divergence estimates used by G;*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for those estimates;*
8. *H records componentwise* hsame *transport for the ODE, dynamics, form, sign, boundary, readback, seal, provenance, and local naming rows;*
9. *C records displayed* Cont *replay from the ODE and dynamics rows through the divergence sign ledger and Real-facing readback;*
10. *P is the* Pkg *provenance over* $\mathsf{BendixsonDulac}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

11. N is the local $\mathsf{NameCert}_{\mathsf{BendixsonDulac}^{\uparrow}}$ *row.*
The carrier has no coordinate for a selected periodic orbit, global flow completion, quotient path equality, measure-theoretic area, or ambient Real completeness.

Theorem 9173.2 (Bendixson–Dulac NameCert obligations)**.** *For every accepted* $\mathsf{BendixsonDulac}^{\uparrow}$ *carrier* $B = (O, D, F, G, S, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BendixsonDulac}^{\uparrow}}$ *surface consists exactly of planar ODE admission through* O*, local dynamics admission through* D*, differential-form divergence readback through* F*, finite sign-ledger visibility through* G*, boundary-separation visibility through* S*, regular-rational readback through* W*, Real-seal readback through* R*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 9173.1. The rows O, D, F, G, S, W, R are the analytic certificate rows, and H, C, P, N are the structural rows that preserve, replay, source, and name the same finite packet.

The ODE and dynamics rows display the planar local-flow window. The differential-form row supplies the divergence read, while the sign ledger and boundary-separation row record how the Dulac sign is inspected on the finite region.

The regular-rational and Real-seal rows expose only the displayed coefficient and divergence estimates. Since no further coordinate is present, the local NameCert obligations are exhausted by the listed rows. □

Theorem 9173.3 (Bendixson–Dulac divergence handoff)**.** *Every cycle-exclusion consumer of an accepted* $\mathsf{BendixsonDulac}^{\uparrow}$ *packet must first read the route*

$$\mathsf{Ode}^{\uparrow},\ \mathsf{DynSystem}^{\uparrow},\ \mathsf{DiffForm}^{\uparrow},\ G,\ S,\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ H,\ C,\ P,\ N.$$

The divergence handoff may be used as a finite sign certificate beside the Poincare–Bendixson trap route of chapter 6911, but it exports no selected periodic orbit, no global flow completion, no quotient path equality, no measure-theoretic area principle, and no ambient Real completeness.

Proof. Apply the NameCert obligations theorem Definition 9173.2. It forces every accepted read through the displayed ODE, dynamics, differential-form, sign-ledger, boundary, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.

The sign ledger is the only row that records the divergence condition. Therefore a consumer cannot obtain a cycle-exclusion handoff without preserving the displayed divergence-form read and the finite Real-facing estimates that support its sign.

The comparison with chapter 6911 is a consumer route only. That sibling supplies a trapping-region packet, while this packet supplies a divergence sign packet; neither row set creates a selected orbit, a completed global flow, quotient path equality, measure theory, or ambient Real completeness. □

Falsifiable prediction: If the finite sign ledger G or the differential-form row F is absent, an accepted $\mathsf{BendixsonDulac}^{\uparrow}$ packet cannot expose a divergence-based cycle-exclusion handoff.

Independence witness: ode, dynsystem, diffform, real, regseqrat, poincarebendixsontrap: none is bijective with $\mathsf{BendixsonDulac}^{\uparrow}$ because this packet jointly stores the planar ODE row, local dynamics row, divergence-form row, finite sign ledger, boundary-separation row, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BendixsonDulac}^{\uparrow}}$**.**

Constructive story: A BendixsonDulac$^{\uparrow}$ packet is generated from a planar OdeUp row, a DynSystemUp local-flow row, a DiffFormUp divergence-form row, a finite sign ledger, a boundary-separation row, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BendixsonDulac}^{\uparrow}}$, seed)

Scope closed: Definition 9173.1, Definition 9173.2, and Definition 9173.3 seed the finite divergence criterion over displayed ODE, dynamics, differential-form, sign, boundary, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{BendixsonDulac}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected periodic orbits, global flow completion, measure theory, quotient path equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Further closure requires checked carrier admission, divergence-sign stability, cycle-exclusion handoff, and bridge evidence for the same finite Bendixson–Dulac packet.

Chapter 9174

A Concrete Naming Certificate for $\mathsf{RegularRealMorphism}^{\uparrow}$

Following the Bishop–Bridges treatment of constructive real functions by regular Cauchy names and explicit moduli, the $\mathsf{RegularRealMorphism}^{\uparrow}$ packet records a finite BEDC surface for a morphism between regular real names. It sits after the dyadic rational core of chapter 291, the stream-name interface of chapter 126, the regular rational sequence carrier of chapter 301, the real-name certificate of chapter 38, and the continuity certificate of chapter 130. The packet names only the displayed source name, target name, dyadic modulus, transport, continuity replay, provenance, and local certificate; it is not a host function, quotient-real equality principle, completeness theorem, or choice operator.

Definition 9174.1 (Regular real morphism carrier)**.** *A* $\mathsf{RegularRealMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *row, exposed as a regular Cauchy real name through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *sibling surfaces.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *row, exposed as a regular Cauchy real name with the same sibling interfaces available for inspection.*
3. *M is the dyadic modulus row over* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording how a requested target precision is converted into the source precision that the morphism reads.*
4. *H records componentwise* hsame *transport for source name, target name, dyadic modulus, continuity replay, provenance, and local name.*
5. *C records displayed* Cont *replay from the source regular name, through the dyadic modulus read, to the target regular name.*
6. *P is the* Pkg *provenance over* $\mathsf{RegularRealMorphism}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularRealMorphism}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through this finite morphism surface.

Theorem 9174.2 (Regular real morphism NameCert surface)**.** *For every accepted* $\mathsf{RegularRealMorphism}^{\uparrow}$ *carrier* $R = (S, T, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularRealMorphism}^{\uparrow}}$ *surface consists exactly*

of source regular-name admission through S, target regular-name admission through T, dyadic-modulus exposure through M, componentwise transport through H, displayed continuity replay through C, provenance through P, and local naming through N.

Proof. Project the carrier of Definition 9174.1. Rows S and T display the source and target real names before any morphism certificate is consumed.

The dyadic modulus row M is the only precision transformer in the packet. It records the source precision required for each displayed target precision, using the dyadic rational core rather than a host real function or quotient-real representative.

Transport, continuity replay, provenance, and local naming are then read through H, C, P, N. These rows preserve the same finite route, so the exported certificate contains no host function graph, extensional quotient equality of reals, completeness construction, or choice functional. □

Theorem 9174.3 (Regular real morphism sibling anchors)**.** *Every consumer of an accepted* $\mathsf{RegularRealMorphism}^{\uparrow}$ *packet must read the sibling route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow}$$

through the displayed rows S, T, M, H, C, P, N. The morphism surface is determined by regular source and target names, a dyadic modulus, hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. The source and target rows S, T anchor the packet in the regular real-name surface. Their available inspections pass through the stream-name and regular rational sequence siblings, so a consumer cannot read a target-name claim without the corresponding regular-name witnesses.

The modulus row M anchors the precision handoff in $\mathsf{DyadicRatCore}^{\uparrow}$. The replay row C then exposes the continuous-map route, while H transports the displayed data and P, N package the provenance and local name.

Thus the sibling anchors are exactly the five named surfaces together with the morphism rows S, T, M, H, C, P, N. Nothing in the packet adds host-function extensionality, quotient-real equality, a completion theorem, or a choice principle. □

Falsifiable prediction: A $\mathsf{RegularRealMorphism}^{\uparrow}$ consumer is refuted whenever it reads a morphism between regular real names before the source row, target row, dyadic modulus, hsame transport, Cont replay, Pkg provenance, and local NameCert row are displayed.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealMorphism}^{\uparrow}$ packet is generated from source and target regular Cauchy real names, a DyadicRatCore modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealMorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9174.1, Definition 9174.2, and Definition 9174.3 seed the finite morphism route between regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealMorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions, quotient reals, completeness, choice, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, target admission, dyadic modulus exposure, transport, continuity replay, provenance, and local naming for BEDC.Derived.RegularRealMorphismUp.

Chapter 9175

A Concrete Naming Certificate for $\mathsf{LowerVietorisCompletion}^{\uparrow}$

Following Vietoris hyperspace practice and the domain-theoretic lower powerdomain reading of closed-set completion, the $\mathsf{LowerVietorisCompletion}^{\uparrow}$ packet records completion through finite lower-hit observations. It sits after the topology carrier of chapter 88, the Vietoris topology surface of chapter 2375, the lower topology route of chapter 8141, the Cauchy ideal route of chapter 9118, the finite probe cover fold of chapter 7452, and the completion reflector of chapter 9129. The packet names only the finite lower-hit, directed-ideal, and completion-handoff rows needed by downstream hyperspace completion consumers.

Definition 9175.1 (Lower Vietoris completion carrier)**.** *A* $\mathsf{LowerVietorisCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (T, L, H, I, F, D, R, S, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the ambient open observations used by the lower-hit classifier.*
2. *L is the* $\mathsf{LowerTopology}^{\uparrow}$ *row, recording that observations are read by hit data and not by upper miss data.*
3. *H is the* $\mathsf{VietorisTopology}^{\uparrow}$ *hyperspace row, restricted to the lower subbasis generated by displayed opens.*
4. *I is the* $\mathsf{CauchyIdeal}^{\uparrow}$ *directed-ideal row, storing the finite tail of lower-hit obligations that is sent to completion.*
5. *F is the finite* $\mathsf{ProbeBundleCoverFold}^{\uparrow}$ *row, listing the displayed opens whose nonempty intersection with the approximating closed set is observed.*
6. *D is the directed lower-hit stability row, asserting that the finite hit list is monotone along the ideal tail.*
7. *R is the* $\mathsf{CompletionReflector}^{\uparrow}$ *handoff row, carrying the lower-hit ideal into the completion corridor.*
8. *S records componentwise* hsame *transport for the topology, lower topology, Vietoris, ideal, finite probe, stability, reflector, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from ambient opens to lower-hit probes, from probes to directed ideals, and from directed ideals to the completion handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{LowerVietorisCompletion}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LowerTopology}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{CauchyIdeal}^{\uparrow}$, $\mathsf{ProbeBundleCoverFold}^{\uparrow}$, $\mathsf{CompletionReflector}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerVietorisCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for upper Vietoris miss tests, Hausdorff distance, compactness, selected limits, quotient equality of closed sets, powerset extensionality, or ambient metric completeness.

Lean4 (statement) `BEDC.Derived.LowerVietorisCompletionUp`

Theorem 9175.2 (Lower Vietoris completion NameCert obligations)**.** *For every accepted* $\mathsf{LowerVietorisCompletion}^{\uparrow}$ *carrier $V = (T, L, H, I, F, D, R, S, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LowerVietorisCompletion}^{\uparrow}}$ *surface consists exactly of ambient topology admission through T, lower topology admission through L, lower Vietoris hyperspace readback through H, directed Cauchy ideal visibility through I, finite lower-hit probe exposure through F, directed lower-hit stability through D, completion handoff through R, componentwise transport through S, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9175.1. Rows T, L, H identify the ambient-open, lower-topology, and Vietoris hyperspace sources before any completion read is consumed.

The lower-hit data is finite at each read. Row F lists the displayed opens, and D records monotone stability of those hits along the directed ideal row I. Thus a consumer cannot replace the lower-hit route by an upper miss test, a metric Hausdorff estimate, or an extensional powerset equality.

The terminal rows preserve the same route. Row R is the only completion-facing handoff, while S, C, P, N transport, replay, source, and name the packet without adding compactness, selected limits, quotient equality, or ambient completeness. □

Theorem 9175.3 (Lower-hit completion consumer route)**.** *Every hyperspace completion consumer of an accepted* $\mathsf{LowerVietorisCompletion}^{\uparrow}$ *packet must first read the finite route*

$$T, L, H, I, F, D, R, S, C, P, N.$$

The exported object is therefore a lower-hit completion handoff from directed finite observations; it is not a Fell topology packet, an upper Vietoris classifier, a Hausdorff metric completion, or a selected completed closed set.

Proof. Apply Definition 9175.2. It identifies the entire accepted local surface, so every consumer receives the topology, lower topology, lower Vietoris, directed ideal, finite probe, stability, completion, transport, replay, provenance, and naming rows.

The classifier is lower-hit only. The finite probe row records observed intersection with displayed opens, and the directed stability row keeps those observations coherent along the Cauchy ideal tail.

Consequently the completion reflector row can export only this finite lower-hit handoff. Since no row stores upper miss tests, Fell convergence, Hausdorff metric estimates, a selected closed-set limit, or quotient equality, none of those structures is available to the consumer. □

Falsifiable prediction: If the finite lower-hit row F or the directed stability row D is absent, an accepted $\mathsf{LowerVietorisCompletion}^{\uparrow}$ packet cannot expose a hyperspace completion handoff.

Independence witness: topology, lower-topology, vietoris-topology, cauchy-ideal, probe-bundle-cover-fold, completion-reflector: none is bijective with $\mathsf{LowerVietorisCompletion}^{\uparrow}$ because this packet jointly stores ambient opens, lower-hit classifier data, finite probe observations, directed ideal stability, completion handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LowerVietorisCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerVietorisCompletion}^{\uparrow}$ packet is generated from a TopologyUp source row, a LowerTopologyUp lower-hit row, a VietorisTopologyUp hyperspace row, a CauchyIdealUp directed-ideal row, finite ProbeBundleCoverFoldUp observations, directed lower-hit stability, a CompletionReflectorUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerVietorisCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9175.1, Definition 9175.2, and Definition 9175.3 seed the finite lower-hit completion route over displayed topology, ideal, probe, reflector, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerVietorisCompletion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.LowerVietorisCompletionUp`

Bridge status: none

Not claimed: This chapter does not close upper Vietoris miss tests, Fell topology, Hausdorff metric completion, compactness, selected closed-set limits, quotient equality, powerset extensionality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked lower-hit carrier admission, directed-ideal stability, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.LowerVietorisCompletionUp.

Chapter 9176

A Concrete Naming Certificate for $\mathsf{GaussMap}^{\uparrow}$

Following Gauss and Khinchin on continued fractions, the $\mathsf{GaussMap}^{\uparrow}$ packet records the finite BEDC surface for the map that sends a positive real input in the unit interval to the fractional part of its reciprocal. It consumes the real-name row of chapter 38, the reciprocal route of chapter 6251, the floor row of chapter 6900, the residual row of chapter 1648, the finite continued-fraction prefix row of chapter 342, the finite window and regular-rational readback rows of chapter 126 and chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet names a constructive dynamics step; it is not a host real function, an infinite continued-fraction shift, quotient equality of real names, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9176.1 (Gauss map carrier)**.** *A* $\mathsf{GaussMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, Z, V, F, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row for the input real name.*
2. *Z is the nonzero positive unit-interval guard row attached to the same real source.*
3. *V is the* $\mathsf{RealReciprocal}^{\uparrow}$ *row displaying the reciprocal read permitted by Z.*
4. *F is the* $\mathsf{FloorFunction}^{\uparrow}$ *row for the integer part of the reciprocal read.*
5. *Q is the* $\mathsf{FractionalPart}^{\uparrow}$ *residual row bound to the same reciprocal and floor rows.*
6. *A is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *finite-prefix handoff row, recording the partial-quotient prefix visible after the reciprocal and floor read.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the fractional residual.*
8. *H records componentwise* hsame *transport for the source, guard, reciprocal, floor, residual, prefix, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the source real and guard through reciprocal, floor, fractional residual, continued-fraction prefix, and terminal seal.*
10. *P is the* Pkg *provenance over* $\mathsf{GaussMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{FloorFunction}^{\uparrow}$, $\mathsf{FractionalPart}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaussMap}^\uparrow}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host Gauss-map function, quotient real equality, an infinite continued-fraction stream, a selected representative, trichotomy, countable choice, or detached $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 9176.2 (Gauss map NameCert obligations)**.** *For every accepted* $\mathsf{GaussMap}^\uparrow$ *carrier* $G = (R, Z, V, F, Q, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GaussMap}^\uparrow}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^\uparrow$ *row, the nonzero positive unit-interval guard, the* $\mathsf{RealReciprocal}^\uparrow$ *row, the* $\mathsf{FloorFunction}^\uparrow$ *row, the* $\mathsf{FractionalPart}^\uparrow$ *residual row, the* $\mathsf{ContinuedFraction}^\uparrow$ *prefix handoff, the terminal* $\mathsf{Real}^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 9176.1. The source row R and guard row Z are the only admission data for the real input; the reciprocal row V is public only after those rows have been displayed.

Read the arithmetic route next. The floor row F and fractional residual row Q are tied to the same reciprocal row, and the continued-fraction prefix row A records only the finite partial-quotient handoff visible from that route.

Finally inspect the seal and structural rows. The row E is a $\mathsf{Real}^\uparrow$ seal for the displayed residual, while H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 9176.3 (Gauss map fractional-part route)**.** *Every dynamics-facing read exported by an accepted* $\mathsf{GaussMap}^\uparrow$ *carrier factors through*

$$R \longmapsto Z \longmapsto V \longmapsto F \longmapsto Q \longmapsto A \longmapsto E$$

together with H, C, P, N. Consequently the public read is a finite reciprocal, floor, fractional-part, continued-fraction-prefix, and real-seal route, not a host function evaluation, an infinite shift on continued fractions, quotient equality of real names, a selected representative, or an ambient completeness principle.

Lean4 (statement) `BEDC.Derived.GaussMapUp`

Proof. Start from the accepted carrier. The guard row Z keeps the reciprocal request attached to the displayed positive nonzero interval data of the source row R, so the reciprocal row V is not detached from the input name.

Apply the floor and fractional-part sibling routes. The floor row F supplies the integer part of the reciprocal read, and Q records the residual on the same reciprocal and floor data before any continued-fraction prefix is exposed.

The finite-prefix handoff A and seal row E are downstream of that residual. Transport, replay, provenance, and local naming preserve the same route, and the excluded host-function, infinite-stream, quotient, representative, choice, and completeness coordinates are absent from the carrier. □

Falsifiable prediction: Every $\mathsf{GaussMap}^\uparrow$ dynamics read factors through $\mathsf{Real}^\uparrow$, $\mathsf{RealReciprocal}^\uparrow$, $\mathsf{FloorFunction}^\uparrow$, $\mathsf{FractionalPart}^\uparrow$, $\mathsf{ContinuedFraction}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and a terminal $\mathsf{Real}^\uparrow$ seal before the local $\mathsf{NameCert}_{\mathsf{GaussMap}^\uparrow}$ row is consumed.

Independence witness: real, real-reciprocal, floor-function, fractional-part, continuedfraction, streamname, regseqrat: $\mathsf{GaussMap}^\uparrow$ is not bijective to any listed sibling because it jointly carries a

source Real row, positive nonzero interval guard, reciprocal row, floor row, fractional-part residual, continued-fraction prefix handoff, Real seal, transport, replay, provenance, and local naming while refusing host function evaluation and infinite continued-fraction shifts.

Closure status for $\mathsf{NameCert}_{\mathsf{GaussMap}\uparrow}$**.**

Constructive story: A $\mathsf{GaussMap}^{\uparrow}$ packet is generated from a source RealUp row, a nonzero positive interval guard, a RealReciprocal row, a FloorFunction row, a FractionalPart residual row, a ContinuedFraction prefix handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaussMap}\uparrow}, \mathsf{seed})$

Scope closed: Definition 9176.1, Definition 9176.2, and Definition 9176.3 seed the finite reciprocal, floor, fractional-part, continued-fraction-prefix, and Real-seal route for the Gauss map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaussMap}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real functions, infinite continued-fraction shifts, quotient real equality, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked Lean obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, guard exactness, reciprocal routing, floor and fractional-part handoff, continued-fraction prefix handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.GaussMapUp.

Chapter 9177

A Concrete Naming Certificate for $\mathsf{BishopNestedInterval}^{\uparrow}$

Following Bishop and Bridges on constructive nested intervals, the $\mathsf{BishopNestedInterval}^{\uparrow}$ packet records the finite endpoint-window surface by which a decreasing sequence of inhabited closed rational intervals is read into regular Cauchy and real-name consumers. It sits after the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested-interval principle packet of chapter 7290. The packet names only displayed endpoint monotonicity, shrinking-width budgets, finite windows, regular readback, transport, replay, provenance, and local naming; it does not assert host completeness, Heine–Borel compactness, selected limit points, quotient real equality, trichotomy, or countable choice.

Definition 9177.1 (Bishop nested interval carrier)**.** *A* $\mathsf{BishopNestedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-endpoint stream row, exposing each finite lower endpoint only through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *data.*
2. *U is the upper-endpoint stream row, exposing each finite upper endpoint through the same dyadic and stream rows.*
3. *D is the dyadic width-budget row, recording the displayed shrinking tolerance assigned to each inspected interval window.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *request row selecting the endpoint windows consumed by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the paired endpoint streams and their shrinking-width budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S and R have been displayed.*
7. *H records componentwise* hsame *transport for endpoint streams, width budgets, windows, regular readback, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from endpoint-window requests through monotonicity, shrinking width, regular readback, and the Real seal.*

9. *P is the* Pkg *provenance over* BishopNestedInterval$^{\uparrow}$, NestedIntervalPrinciple$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopNestedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, S, R, E, H, C, P, N$. It has no coordinate for a host completed point, open-cover compactness, a selected intersection element, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 9177.2 (Bishop nested interval monotone endpoint obligation)**.** *For every accepted* BishopNestedInterval$^{\uparrow}$ *carrier*

$$I = (L, U, D, S, R, E, H, C, P, N),$$

the monotone endpoint obligation consists exactly of lower-endpoint stream exposure through L, upper-endpoint stream exposure through U, dyadic width-budget exposure through D, finite window exposure through S, regular rational readback through R, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.

Proof. Project the carrier of Definition 9177.1. The endpoint rows L and U are the only rows that expose interval endpoints, and the budget row D is the only row that records shrinking-width data.

Read the monotonicity obligation through finite windows. The row S selects the inspected endpoint windows, so endpoint order and interval inclusion are checked only on displayed StreamName$^{\uparrow}$ requests and dyadic comparisons.

The regular and Real-facing rows are downstream. Row R reads the paired endpoints and their width budget as regular rational data, while E seals only that displayed readback as Real$^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate supplies host completed points, open-cover compactness, selected intersections, quotient equality, trichotomy, choice, or ambient completeness, the listed rows exhaust the monotone endpoint obligation. □

Theorem 9177.3 (Bishop nested interval regular Cauchy handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* BishopNestedInterval$^{\uparrow}$ *packet receives a regular Cauchy handoff only through the route*

$$L, U, D, S, R, E, H, C, P, N.$$

The handoff factors through DyadicRatCore$^{\uparrow}$ *endpoint and width arithmetic,* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *regular readback,* Real$^{\uparrow}$ *sealing, and the nested-interval principle of chapter 7290; it exports no host completeness theorem, Heine–Borel compactness, selected point, quotient real equality, trichotomy, or countable-choice row.*

Proof. Use the monotone endpoint obligation of Definition 9177.2. It places the lower and upper endpoint streams, dyadic width budget, finite windows, and regular readback before any Real-facing row is public.

Apply the nested-interval principle route through chapter 7290. That route consumes finite window and shrinking-budget data before it hands a readback to Real-facing consumers.

The terminal seal E is therefore downstream of S and R. A completion-facing consumer can read a regular Cauchy name only after the displayed StreamName windows and RegSeqRat readback have been exposed.

Because the carrier lacks rows for host completeness, Heine–Borel compactness, selected points, quotient real equality, trichotomy, and countable choice, the handoff is a finite NameCert route rather than an ambient intersection theorem. □

Theorem 9177.4 (Bishop nested interval NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{BishopNestedInterval}^{\uparrow}}$ *surface for an accepted carrier is exactly the finite endpoint-window route named by Definition 9177.1, Definition 9177.2, and Definition 9177.3. Every accepted read must expose the endpoint streams, dyadic width budget, finite StreamName windows, RegSeqRat readback, Real seal, componentwise transport, continuation replay, provenance, and local naming before a Real or Completion consumer may read the packet.*

Proof. Project the accepted carrier. Its analytic rows are L, U, D, S, R, E, and its structural rows are H, C, P, N.

Use the monotone endpoint obligation to identify the endpoint and shrinking-width part of the certificate. This keeps interval inclusion and endpoint comparison inside the displayed dyadic and stream-window rows.

Use the regular Cauchy handoff to identify the Real-facing part of the certificate. The handoff remains downstream of StreamName windows, RegSeqRat readback, the nested-interval principle route, and the Real seal.

The structural rows preserve only that finite route under transport and replay. Thus the local NameCert surface contains no hidden host completion, Heine–Borel compactness, selected intersection point, quotient real equality, trichotomy, or countable-choice principle. □

Falsifiable prediction: A $\mathsf{BishopNestedInterval}^{\uparrow}$ consumer is refuted whenever it reads a Real-facing nested-interval handoff without exposing endpoint streams, a DyadicRatCore width budget, StreamName windows, RegSeqRat readback, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopNestedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopNestedInterval}^{\uparrow}$ packet is generated from lower and upper endpoint streams, dyadic width budgets, finite StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopNestedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9177.1, Definition 9177.2, Definition 9177.3, and Definition 9177.4 seed the endpoint-window route from constructive nested intervals to regular Cauchy and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopNestedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, Heine–Borel compactness, selected intersection points, quotient real equality, trichotomy, countable choice, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint-stream admission, dyadic width-budget exactness, finite window generation, regular readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopNestedIntervalUp.

Chapter 9178

A Concrete Naming Certificate for CompactRealIntervalModulusWindow$^{\uparrow}$

Following Bishop and Bridges on compact intervals and uniform continuity, the CompactRealIntervalModulusWindow packet realizes the countable-completion spine of chapter 9357 as finite interval-window and modulus data. It sits after the dyadic tolerance arithmetic of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the compact metric finite-net route of chapter 129, and the continuous-map graph route of chapter 130. The packet names a finite compact interval modulus window for Real and completion consumers; it does not assert host Heine–Borel compactness, selected Lebesgue numbers, quotient real equality, countable choice, external modulus oracles, or ambient Real$^{\uparrow}$ completeness.

Definition 9178.1 (Compact Real interval modulus window carrier)**.** *A* CompactRealIntervalModulusWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, D, S, R, K, F, W, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper* Real$^{\uparrow}$ *endpoint rows for the inspected compact interval.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row naming the requested source-window and target-window budgets.*
3. *S is the* StreamName$^{\uparrow}$ *finite window row selecting endpoint, mesh, and function-value observations.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for endpoint streams, mesh centers, and displayed value windows.*
5. *K is the* CompactMetric$^{\uparrow}$ *finite-net row for the compact interval window.*
6. *F is the* ContinuousMap$^{\uparrow}$ *graph row whose values are read only on the displayed compact interval windows.*
7. *W is the modulus-window ledger, assigning the displayed source tolerance to the target tolerance over the finite compact interval net.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row for endpoint, mesh, and value histories after S, R, K, F, W are displayed.*

9. *H records componentwise* hsame *transport for endpoints, tolerance rows, finite windows, regular readback, compact-net data, continuous-map graph rows, modulus windows, Real seals, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a tolerance request through finite interval windows, compact finite nets, regular readback, continuous-map graph rows, modulus-window assignment, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, S, R, K, F, W, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected Lebesgue numbers, quotient real equality, countable choice, external modulus oracles, or ambient real completeness.

Theorem 9178.2 (Compact Real interval modulus window obligation)**.** *For every accepted* CompactRealIntervalMo *carrier*

$$M = (A, B, D, S, R, K, F, W, E, H, C, P, N),$$

the local obligation surface consists exactly of endpoint admission through A, B, dyadic tolerance admission through D, finite window exposure through S, regular rational readback through R, compact finite-net exposure through K, continuous-map graph exposure through F, modulus-window assignment through W, Real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.

Proof. Project the carrier of Definition 9178.1. The endpoint rows, dyadic tolerance row, finite window row, regular readback row, compact finite-net row, continuous-map graph row, modulus-window ledger, and Real seal are all explicit coordinates.

The countable-name route is read before the compact-modulus route. Rows S and R expose finite StreamName windows and RegSeqRat readback, while D supplies the dyadic tolerance budget used by both source and target windows.

The compact interval route is then read through K, F, W. The compact finite-net row fixes the interval windows, the continuous-map graph row supplies displayed values on those windows, and the modulus-window ledger assigns source tolerances to target tolerances over that finite route.

Finally E, H, C, P, N seal, transport, replay, source, and name the same packet. Since no excluded host compactness, selected Lebesgue number, quotient equality, choice, external oracle, or ambient completeness coordinate is present, the listed rows exhaust the local obligation surface. □

Theorem 9178.3 (Compact Real interval modulus window NameCert surface)**.** *Every compact-uniform-continuity, Real-completion, or countable-completion consumer of an accepted* CompactRealIntervalModulus *packet must read the finite route*

$$A, B, D, S, R, K, F, W, E, H, C, P, N.$$

This route realizes the countable-completion spine through finite interval endpoints, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows, modulus-window assignment, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no open-cover compactness theorem, selected Lebesgue number, quotient Real equality, countable-choice row, external modulus oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply the obligation theorem of Definition 9178.2. It identifies all endpoint, tolerance, finite-window, readback, compact-net, continuous-map, modulus-window, seal, transport, replay, provenance, and naming rows visible to a consumer.

Read the countable-completion side through D, S, R, E. These rows provide dyadic tolerances, finite StreamName windows, RegSeqRat readback, and a terminal Real seal before any completion-facing consumer can use the packet.

Read the compact-uniform side through K, F, W. The compact finite-net row and continuous-map graph row provide the displayed interval windows, and the modulus-window ledger is downstream of those finite rows.

The structural rows H, C, P, N preserve the same route. Since the classifier has no coordinates for the excluded host principles, the NameCert surface is exactly the finite route displayed in the statement. □

Falsifiable prediction: A $\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}$ consumer is refuted whenever it reads a compact interval modulus without exposing Real endpoint rows, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, CompactMetric finite-net rows, ContinuousMap graph rows, a modulus-window ledger, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}$ packet is generated from Real endpoint rows, DyadicRatCore tolerance budgets, finite StreamName windows, RegSeqRat readback, CompactMetric finite-net rows, ContinuousMap graph rows, a modulus-window ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9178.1, Definition 9178.2, and Definition 9178.3 seed the finite compact interval modulus-window route for countable-completion and compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalModulusWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, quotient Real equality, countable choice, external modulus oracles, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic tolerance exactness, finite window generation, regular readback, compact finite-net exposure, continuous-map graph exposure, modulus-window assignment, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealIntervalModulusWindowUp.

Chapter 9179

A Concrete Naming Certificate for $\mathsf{ParallelogramLaw}^{\uparrow}$

Following Jordan and von Neumann's characterization of normed spaces whose norm is induced by an inner product, the $\mathsf{ParallelogramLaw}^{\uparrow}$ packet records the finite BEDC surface on which a vector-space row, norm row, and inner-product candidate are inspected through the parallelogram identity. It sits after the vector carrier of chapter 47, the norm carrier of chapter 89, the inner-product surface of chapter 86, and the Hilbert-space handoff of chapter 91. The packet names only displayed norm and pairing rows; it is not a host vector equality proof, a completed analytic structure, a quotient-norm theorem, or a selected polarization oracle.

Definition 9179.1 (Parallelogram-law carrier)**.** *A* $\mathsf{ParallelogramLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, N, I, P, H, C, R, L)$$

with the following rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, recording the displayed vector carrier, zero, addition, inverse, scalar action, and finite linear-combination windows used in the law.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *row over V, recording the norm endpoints for x, y, $x+y$, and $x-y$ together with the finite comparison ledger used by the packet.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *candidate row, recording the pairing table and symmetry, additivity, scalar, and positivity witnesses that may induce the displayed norm.*
4. *P is the parallelogram identity ledger. For every displayed pair (x, y) in the finite window it stores the two norm-square sides*

   $$N(x+y) + N(x-y) \quad \textit{and} \quad 2N(x) + 2N(y)$$

   as $\mathsf{Real}^{\uparrow}$*-sealed rows, together with the equality readback for that pair.*
5. *H records componentwise* hsame *transport for the vector, norm, inner-product, parallelogram, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from the vector row through the norm and inner-product rows to the parallelogram ledger.*
7. *R is the* Pkg *provenance over* $\mathsf{ParallelogramLaw}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

8. L is the local $\mathsf{NameCert}_{\mathsf{ParallelogramLaw}^{\uparrow}}$ *row.*
The local classifier compares accepted packets only by these displayed rows. It has no coordinate for host vector equality, completed Hilbert structure, quotient norm equality, unbounded analytic closure, or an inner-product reconstruction not recorded in I and P.

Theorem 9179.2 (Parallelogram-law NameCert obligations)**.** *For every accepted* $\mathsf{ParallelogramLaw}^{\uparrow}$ *carrier $Q = (V, N, I, P, H, C, R, L)$, the local* $\mathsf{NameCert}_{\mathsf{ParallelogramLaw}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, vector-source exposure through V, norm-readback exposure through N, inner-product candidate exposure through I, parallelogram identity accountability through P, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through R, and local naming through L.*

Proof. Project the carrier of Definition 9179.1. The vector source row V is the only place where the vector operations and finite pair window are stored.

The norm row N exposes the four norm endpoints for each displayed pair, while I exposes the inner-product candidate whose induced norm is being tested. Neither row supplies a host equality principle or a completed Hilbert object.

The ledger P records the two parallelogram sides and their $\mathsf{Real}^{\uparrow}$ readback for the same pair window. Transport and replay through H and C preserve that row order, and provenance and naming through R and L package the same finite data.

Since no other coordinate appears in Q, the listed rows exhaust the local NameCert obligation surface. □

Theorem 9179.3 (Parallelogram-law norm-inner-product handoff)**.** *Every Hilbert-facing consumer of an accepted* $\mathsf{ParallelogramLaw}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{Norm}^{\uparrow} \longmapsto \mathsf{InnerProduct}^{\uparrow} \longmapsto \mathsf{ParallelogramLaw}^{\uparrow} \longmapsto \mathsf{Hilbert}^{\uparrow}.$$

The route exports only the displayed vector operations, norm endpoints, inner-product candidate, parallelogram ledger, transport, replay, provenance, and local naming rows. It does not export completed analytic structure, quotient vector equality, a selected polarization formula outside I, or an ambient Hilbert-space theorem.

Proof. Begin with V. A consumer must first expose the finite vector-space window from which the two vectors, their sum, and their difference are read.

Next read N and I. The norm row supplies the observable norm endpoints, and the inner-product row supplies the candidate pairing surface that may explain those norm endpoints.

Then read the parallelogram ledger P. This row is the handoff point: it records that the norm data and pairing data agree on the displayed finite identity rows before a Hilbert-facing consumer may cite the packet.

Finally use H, C, R, L to transport, replay, package, and name the same rows. The Hilbert sibling may consume this route only as finite NameCert data, so the packet never imports host equality, completion, quotienting, or an external polarization oracle. □

Falsifiable prediction: Every accepted $\mathsf{ParallelogramLaw}^{\uparrow}$ Hilbert-facing read must display $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, parallelogram ledger, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before the $\mathsf{Hilbert}^{\uparrow}$ handoff is visible.

Independence witness: vecspace, norm, innerproduct, hilbert: $\mathsf{ParallelogramLaw}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the vector operation row, norm readback row, inner-product candidate row, parallelogram equality ledger, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ParallelogramLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{ParallelogramLaw}^{\uparrow}$ packet is generated from a VecSpaceUp source row, NormUp readback row, InnerProductUp candidate row, finite parallelogram ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ParallelogramLaw}^{\uparrow}}$, seed)

Scope closed: Definition 9179.1, Definition 9179.2, and Definition 9179.3 seed the finite parallelogram-law route over $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ParallelogramLaw}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient norm equality, completed Hilbert structure, host vector equality, selected polarization outside the displayed rows, or ambient analytic completeness.

Upgrade path: Obligation closure requires checked carrier admission, vector-source exposure, norm-readback exposure, inner-product candidate exposure, parallelogram ledger accountability, transport and replay stability, provenance non-escape, and Hilbert handoff exactness for BEDC.Derived.ParallelogramLawUp.

Chapter 9180

A Concrete Naming Certificate for $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$

Following Besicovitch covering theory and Lebesgue differentiation, the $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ packet records the finite window where bounded-overlap cover data, local density estimates, measure rows, and Real seals are read together before a differentiation consumer sees a verdict. It sits after the Vitali selection surface of chapter 2184, the bounded-overlap covering packet of chapter 6339, the differentiation certificate of chapter 4153, the measure interface of chapter 92, and the terminal real-name seal of chapter 38. The packet names one finite density window only; it is not a host differentiation theorem, a measure-completion principle, a selected almost-everywhere representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 9180.1 (Besicovitch differentiation window carrier)**.** *A* $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, F, O, D, M, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite cover family row, exposing displayed balls or intervals together with their centers, radii, and request cells.*
2. *F is the* $\mathsf{BesicovitchCovering}^{\uparrow}$ *bounded-overlap ledger for the same finite family.*
3. *O is the density-window row, recording the inspected center, radius window, local-average row, exceptional-budget row, and finite nonescape condition.*
4. *D is the* $\mathsf{LebesgueDifferentiation}^{\uparrow}$ *handoff row whose local differentiation read is allowed only after B, F, O are displayed.*
5. *M is the* $\mathsf{Measure}^{\uparrow}$ *row for local measure, measurable-cell admission, and finite density comparison.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for density values, local averages, overlap budgets, and error bounds.*
7. *H records componentwise* hsame *transport for cover family, bounded-overlap ledger, density window, differentiation handoff, measure rows, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from finite cover family through bounded overlap, density-window comparison, measure read, differentiation handoff, and Real sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$, $\mathsf{VitaliCover}^{\uparrow}$, $\mathsf{BesicovitchCovering}^{\uparrow}$, $\mathsf{LebesgueDifferentiation}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, F, O, D, M, R, H, C, P, N$. It has no coordinate for a completed measure algebra, selected countable cover, host differentiation theorem, almost-everywhere quotient, chosen function representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9180.2 (Besicovitch differentiation window NameCert obligations)**.** *For every accepted* $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ *carrier*

$$W = (B, F, O, D, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}}$ *surface consists exactly of finite cover-family admission through B, bounded-overlap exposure through F, density-window exposure through O, differentiation handoff through D, local measure admission through M, Real sealing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9180.1. The rows $B, F, O, D, M, R, H, C, P, N$ are the whole displayed packet.

The covering rows are read first. The finite cover family B exposes only displayed balls or intervals, and the bounded-overlap ledger F routes the same family through the $\mathsf{BesicovitchCovering}^{\uparrow}$ surface named by chapter 6339.

The density rows are then attached to the same finite family. The window row O records the inspected center, radius, average, and exceptional budget, while M supplies the local measure row and R seals the resulting real-valued comparisons.

The differentiation handoff D is downstream of those displayed rows, and H, C, P, N transport, replay, source, and name the same route. Since no excluded host theorem, measure completion, countable selector, quotient, selected representative, or ambient-completeness coordinate is present, the listed rows exhaust the local obligation surface. □

Theorem 9180.3 (Besicovitch differentiation window density handoff)**.** *Every density read accepted by a* $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{VitaliCover}^{\uparrow},\ \mathsf{BesicovitchCovering}^{\uparrow},\ B,\ F,\ O,\ \mathsf{Measure}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{LebesgueDifferentiation}^{\uparrow}.$$

The handoff exports finite cover cells, bounded overlap, one density window, local measure comparisons, Real seals, transport, replay, provenance, and local naming only.

Proof. Start with an accepted carrier $W = (B, F, O, D, M, R, H, C, P, N)$. A density-facing consumer can inspect cover cells only through the displayed finite family B.

The covering comparison is forced through F. This row cites the bounded-overlap discipline of $\mathsf{BesicovitchCovering}^{\uparrow}$ and keeps every overlap count attached to the same finite family.

The density comparison is forced through O, M, R. These rows expose the inspected window, local measure values, local averages, exceptional budget, and Real seal without choosing null-set representatives or completing the measure algebra.

Only after those rows are displayed may the consumer read D as a $\mathsf{LebesgueDifferentiation}^{\uparrow}$ handoff. The structural rows H, C, P, N preserve the route, so the packet exports a finite density-window handoff rather than an ambient differentiation theorem. □

Falsifiable prediction: A $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ consumer is refuted whenever a density verdict is read without displaying a finite cover family, bounded-overlap ledger, density-window row, $\mathsf{Measure}^{\uparrow}$ local measure row, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: vitali-cover, besicovitch-covering, lebesgue-differentiation, measure, real: $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite cover cells, bounded overlap, one density window, local measure comparison, Real sealing, differentiation handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}$ packet is generated from finite cover family rows, BesicovitchCovering bounded-overlap ledgers, density-window rows, LebesgueDifferentiation handoff rows, Measure rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9180.1, Definition 9180.2, and Definition 9180.3 seed the finite density-window route for Besicovitch covering and Lebesgue differentiation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BesicovitchDifferentiationWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full Besicovitch differentiation theorem, exact Euclidean multiplicity constants, measure completion, almost-everywhere quotient equality, countable-choice radius selection, selected function representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite cover-family admission, bounded-overlap exactness, density-window exposure, Measure row admission, Real seal nonescape, differentiation handoff, transport, replay, provenance, and local naming for BEDC.Derived.BesicovitchDifferentiationWindowUp.

Chapter 9181

A Compact Real Interval Choice-Guard Certificate for $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$ packet records the finite guard that lets a compact real interval consumer read only bounded window data before a terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the real-name carrier of chapter 38, the interval enclosure of chapter 1625, the compact-metric finite packet of chapter 129, the StreamName and RegSeqRat readback rows of chapter 126 and chapter 301, and the dyadic arithmetic core of chapter 291. The guard exposes finite cover-window reads for interval consumers; it is not Heine–Borel compactness, an open-cover theorem, a selected finite subcover, quotient equality of real names, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 9181.1 (Compact real interval choice-guard carrier)**.** *A* $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, C, M, S, R, H, T, P, N)$$

with the following displayed rows:

1. *I is the compact real interval source row, naming the displayed endpoint histories and the interval enclosure data read through* $\mathsf{Real}^{\uparrow}$*.*
2. *C is the finite cover-window request row, recording only the cover probes and interval window requested by the consumer.*
3. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row attached to that same finite window.*
4. *S is the combined* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint, window, mesh, and probe histories.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, C, M, S have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, cover-window request, mesh, readback, Real seal, replay, provenance, and local name.*
7. *T records displayed* Cont *replay from interval admission through cover-window request, dyadic mesh, StreamName and RegSeqRat readback, and terminal Real sealing.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for open-cover compactness, a selected finite subcover, quotient real equality, countable choice, detached endpoint representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9181.2 (Compact real interval choice-guard NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$ *carrier*

$$G = (I, C, M, S, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact real interval source, finite cover-window exposure, dyadic mesh compatibility, StreamName and RegSeqRat readback, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. The packet refuses open-cover compactness, countable choice, selected subcovers, quotient real equality, detached endpoint representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 9181.1. The row I is the only compact interval source, and the row C is the only finite cover-window request accepted by the guard.

Read the finite route next. The dyadic mesh row M is attached to the same window as C, and the combined readback row S exposes the StreamName and RegSeqRat data available from the interval, window, mesh, and probe histories.

The Real seal handoff is downstream of those rows. The row R can be consumed only after I, C, M, S have been displayed, while H, T, P, N transport, replay, source, and name the same finite packet.

The refusal clauses follow from absence of carrier coordinates. No row stores an open-cover compactness theorem, a countable-choice selector, a selected finite subcover, quotient real equality, detached endpoint representatives, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}$ packet is generated from a compact real interval source row, a finite cover-window request, a dyadic mesh row, StreamName and RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}}$, seed)

Scope closed: Definition 9181.1 and Definition 9181.2 seed the finite cover-window, mesh, readback, Real-seal, transport, replay, provenance, and local naming route for compact real interval consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactRealIntervalChoiceGuard}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, open-cover compactness, selected finite subcovers, countable choice, quotient real equality, detached endpoint representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite cover-window exposure, dyadic mesh compatibility, StreamName and RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealIntervalChoiceGuardUp.

Chapter 9182

A Concrete Naming Certificate for RegularCauchyTailProductBound$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RegularCauchyTailProductBound$^{\uparrow}$ packet records the finite tail-window product estimate used when two regular Cauchy rational names are multiplied before a Real$^{\uparrow}$ seal is consumed. It sits after the tail-sum comparison route of chapter 2137, the regular product route of chapter 461, the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a tail-product bound as finite BEDC rows; it does not assert host multiplication of completed reals, quotient stream equality, selected product representatives, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 9182.1 (RegularCauchyTailProductBound carrier)**.** *A* RegularCauchyTailProductBound$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, D, P, A, B, E, H, C, Q, N)$$

with the following displayed rows:

1. *W is the tail-window row naming the common cutoff and precision window inherited from the regular Cauchy tail route.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *product-budget row recording source tolerances, endpoint majorants, and the cross-error allowance for the tail product.*
3. *P is the* RegularCauchyProduct$^{\uparrow}$ *row exposed only after the tail window and dyadic product budget are displayed.*
4. *A and B are the two* RegSeqRat$^{\uparrow}$ *source rows whose tails are compared inside the product window.*
5. *E is the* Real$^{\uparrow}$ *seal row consumed only after P has been displayed.*
6. *H records componentwise* hsame *transport for the tail window, product budget, product row, source rows, real seal, provenance, replay, and local naming.*
7. *C records displayed* Cont *replay from the tail-window request through the dyadic budget, regular product row, regular rational sources, and real seal.*
8. *Q is the* Pkg *provenance over* RegularCauchyTailProductBound$^{\uparrow}$, RegularCauchyTailSum$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$ DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailProductBound}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $W, D, P, A, B, E, H, C, Q, N$. It has no coordinate for host real multiplication, quotient streams, selected representatives, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.RegularCauchyTailProductBoundUp`

Theorem 9182.2 (RegularCauchyTailProductBound obligation surface)**.** *For every accepted* RegularCauchyTailPro *carrier $T = (W, D, P, A, B, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailProductBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the tail window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *handoff, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 9182.1. The only tail-facing row is W, and the only arithmetic row available before product handoff is the dyadic budget D.

The rows P, A, B, E then expose the product route. The product row is read through $\mathsf{RegularCauchyProduct}^{\uparrow}$, the source rows through $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal seal through $\mathsf{Real}^{\uparrow}$.

Finally H, C, Q, N transport, replay, source, and name exactly those rows. Since the carrier contains no host multiplication, quotient-stream, selected-representative, choice, trichotomy, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 9182.3 (RegularCauchyTailProductBound product-tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailProductBound}^{\uparrow}$ *carrier reaches the product seal only through the displayed route*

$$W \longmapsto D \longmapsto P \longmapsto A, B \longmapsto E.$$

Thus a downstream product estimate first reads the shared tail window, the dyadic product budget, the regular product row, and the two regular rational source rows before the real seal can be consumed.

Proof. Start with the carrier projection. The row W fixes the finite tail window and D records the product tolerance budget attached to that window.

The handoff row P is downstream of W and D. It repacks the tail-product estimate as a $\mathsf{RegularCauchyProduct}^{\uparrow}$ row, while A and B keep the two source tails visible as $\mathsf{RegSeqRat}^{\uparrow}$ data.

The seal row E is last in the displayed route. Transport, replay, provenance, and naming preserve the same route through H, C, Q, N, so no consumer can replace it by completed-real multiplication, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{RegularCauchyTailProductBound}^{\uparrow}$ read factors through a tail window, a DyadicRatCore product budget, a RegularCauchyProduct row, two RegSeqRat source rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-tail-sum, regularcauchyproduct, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyTailProductBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the product-tail window, dyadic product-budget ledger, regular product handoff, two regular rational source rows, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailProductBound}^{\uparrow}}$.

Constructive story: A RegularCauchyTailProductBound$^{\uparrow}$ packet is generated from a regular Cauchy tail window, a DyadicRatCore product-budget row, a RegularCauchyProduct handoff, two RegSeqRat source rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailProductBound}^{\uparrow}}$, seed)

Scope closed: Definition 9182.1, Definition 9182.2, and Definition 9182.3 seed the finite tail-product bound route for regular Cauchy product and Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailProductBound}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.RegularCauchyTailProductBoundUp`

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient stream equality, selected product representatives, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exposure, dyadic product-budget exactness, RegularCauchyProduct handoff, RegSeqRat source preservation, Real seal nonescape, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.RegularCauchyTailProductBoundUp.

Chapter 9183

A Concrete Naming Certificate for $\mathsf{ChromaticPolynomial}^{\uparrow}$

Following Birkhoff and Whitney, the $\mathsf{ChromaticPolynomial}^{\uparrow}$ packet records the finite graph-colouring data used to compute the chromatic polynomial through deletion and contraction. It sits over the finite graph carrier of chapter 118, the finite-set rows of chapter 112, the spanning-tree witness surface of chapter 237, the deletion-contraction neighbour chapter 7713, and the matroid boundary of chapter 202. The packet names only finite colour palettes, proper-colouring ledgers, and deletion-contraction rows; it does not import an infinite colouring principle, a host graph isomorphism oracle, an unrestricted polynomial ring, or a Tutte-polynomial theorem outside the displayed packet.

Definition 9183.1 (ChromaticPolynomial carrier)**.** *A* $\mathsf{ChromaticPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (G, Q, C, D, H, R, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Graph}^{\uparrow}$ *row naming the vertex list, edge list, incidence readback, loop refusal, and simple-graph boundary used by the packet.*
2. *Q is the* $\mathsf{FinSet}^{\uparrow}$ *colour-palette row, including the finite colour list and membership ledger visible to the colouring read.*
3. *C is the proper-colouring ledger, listing the accepted vertex-to-colour rows and the edgewise inequality checks for adjacent vertices.*
4. *D is the deletion-contraction row for a displayed edge, exposing the deletion graph, contraction graph, and the boundary cases for loops and empty edge lists.*
5. *H records componentwise* hsame *transport for the graph, colour palette, colouring ledger, deletion-contraction row, replay, provenance, and local name.*
6. *R records displayed* Cont *replay from graph admission through palette lookup, proper-colouring verification, deletion-contraction comparison, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{ChromaticPolynomial}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{SpanningTree}^{\uparrow}$, $\mathsf{TuttePolynomial}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ChromaticPolynomial}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through G, Q, C, D, H, R, P, N. It has no coordinate for an infinite colour set, host graph equality, unrestricted polynomial algebra, a hidden deletion-contraction oracle, or a matroid/Tutte theorem not displayed as finite sibling data.

Theorem 9183.2 (ChromaticPolynomial NameCert obligations)**.** *For every accepted* $\mathsf{ChromaticPolynomial}^{\uparrow}$ *carrier* $X = (G, Q, C, D, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChromaticPolynomial}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite graph row, finite colour-palette exposure, proper-colouring ledger exactness, deletion-contraction boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 9183.1. The row G is the only graph source, and Q is the only colour-palette source.

Read the colouring data through C. Properness is checked edge by edge against the finite graph row, so the ledger records only displayed vertex-colour assignments and displayed adjacency refusals.

Read deletion and contraction through D. The row names the selected edge, the deletion graph, the contraction graph, and the loop or empty-edge boundary cases before any recurrence-style comparison is exposed.

Finally use H, R, P, N. These rows transport, replay, source, and name the same finite packet. Since the carrier contains no infinite palette, host graph equality, unrestricted polynomial algebra, or hidden Tutte/matroid theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 9183.3 (ChromaticPolynomial deletion-contraction boundary)**.** *Every deletion-contraction read accepted by a* $\mathsf{ChromaticPolynomial}^{\uparrow}$ *carrier factors through the displayed route*

$$G \longmapsto D \longmapsto C \longmapsto Q \longmapsto H, R, P, N.$$

The deletion graph and contraction graph are read as finite graph rows tied to the same colour-palette and proper-colouring ledger. Boundary cases for loops, empty edge lists, and palette exhaustion remain ledger rows, not external polynomial identities.

Proof. Start with the graph row G. It supplies the finite vertices, finite edges, and incidence readback used by the selected edge in D.

The deletion-contraction row D keeps both derived graphs inside the same packet. Their colouring reads are not independent host graphs; each is compared through the palette row Q and the proper-colouring ledger C.

The structural rows H, R, P, N preserve the same finite route and record its provenance through the graph, finite-set, spanning-tree, Tutte-polynomial, and matroid neighbours. They do not introduce an unrestricted polynomial identity or a hidden recurrence theorem.

Thus deletion-contraction is a boundary of the displayed NameCert packet: a downstream consumer may read the finite comparison route, but not an ambient chromatic-polynomial algebra beyond it. □

Falsifiable prediction: Every accepted $\mathsf{ChromaticPolynomial}^{\uparrow}$ read exposes a finite graph row, finite colour-palette row, proper-colouring ledger, deletion-contraction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: graph, finset, spanningtree, tutte-polynomial, matroid: none is bijective to $\mathsf{ChromaticPolynomial}^{\uparrow}$ because none jointly carries the finite colour palette, proper-colouring ledger, deletion-contraction row, transport, replay, provenance, and local naming rows for one chromatic-polynomial packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ChromaticPolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{ChromaticPolynomial}^{\uparrow}$ packet is generated from a finite graph row, a finite colour-palette row, a proper-colouring ledger, a deletion-contraction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChromaticPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9183.1, Definition 9183.2, and Definition 9183.3 seed the finite graph-colouring carrier, NameCert obligation surface, and deletion-contraction boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChromaticPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted polynomial algebra, infinite colour palettes, host graph equality, a Tutte-polynomial theorem, or a matroid theorem for $\mathsf{ChromaticPolynomial}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, palette exposure, proper-colouring ledger exactness, deletion-contraction boundary exactness, transport stability, replay, provenance, and local naming for BEDC.Derived.ChromaticPolynomia

Chapter 9184

A Concrete Naming Certificate for $\mathsf{QuantumGRSectionBundle}^{\uparrow}$

The $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ packet concretizes the section-bundle boundary of chapter 9422 as finite NameCert data. It names a base-frame ledger, vertical fibre rows, quantum-state and observable rows, horizontal comparison rows, connection rows, transport, replay, provenance, and local naming; it does not claim a physical quantum-gravity theory, a Hilbert-space completion theorem, a smooth spacetime manifold, a curvature equation, or an empirical law outside the packet. Rows B, V, Q, O, H, K consume the sibling carriers of chapter 97, chapter 134, chapter 217, and chapter 218 before the local section-bundle packet is read.

Falsifiable prediction: If a claimed $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ read can expose a quantum state, observable, or horizontal frame comparison without the displayed bundle, connection, vertical fibre, and horizontal comparison rows, then the local carrier admission row is false.

Independence witness: bundle, connection, quantumstate, observable, hilbert: none is bijective to $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ because none jointly carries the finite base-frame ledger, vertical fibre rows, quantum-state row, observable row, horizontal comparison row, connection row, transport, replay, provenance, and local NameCert row.

Definition 9184.1 (QuantumGRSectionBundle carrier)**.** *A* $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (B, V, Q, O, H, K, T, R, P, N)$$

with the following displayed rows:

1. *B is the finite base-frame ledger naming the observed frame labels, overlap rows, and section domains used by the packet.*
2. *V is the vertical fibre row, recording the finite fibre labels, state slots, projection slots, and residual ledgers visible inside each displayed frame.*
3. *Q is the quantum-state row, exposing only the listed amplitudes, phase tags, normalization checks, and measurement-update entries available in V.*
4. *O is the observable row, naming the finite observable probes, outcome rows, and Born-weight readback attached to Q.*
5. *H is the horizontal comparison row, recording neighbouring frame comparisons, metric readout slots, and curvature-facing boundary entries.*

6. *K is the connection row that ties vertical and horizontal reads through displayed compatibility checks rather than a detached smooth connection.*
7. *T records componentwise* hsame *transport for the bundle, fibre, quantum-state, observable, horizontal comparison, connection, provenance, replay, and local naming rows.*
8. *R records displayed* Cont *replay from base-frame admission through vertical fibre readback, quantum-state and observable exposure, horizontal comparison, connection compatibility, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{QuantumGRSectionBundle}^{\uparrow}$*, chapter 9422,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{QuantumGRSectionBundle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, V, Q, O, H, K, T, R, P, N$. It has no coordinate for a completed Hilbert space, a smooth manifold, an unrestricted connection, a curvature field equation, a hidden measurement oracle, or a physical unification theorem.

Theorem 9184.2 (QuantumGRSectionBundle NameCert obligations)**.** *For every accepted* QuantumGRSectionBun *carrier $Y = (B, V, Q, O, H, K, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuantumGRSectionBundle}^{\uparrow}}$ *surface consists exactly of carrier admission for the base-frame ledger, vertical fibre rows, quantum-state row, observable row, horizontal comparison row, connection row, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 9184.1. The base data are B and V: a finite frame ledger and a finite vertical fibre ledger.

Read the quantum face through Q and O. State rows, phase tags, normalization checks, observables, outcomes, and Born-weight readback are exposed only as finite rows inside the displayed fibre.

Read the relativistic face through H and K. Horizontal frame comparisons and connection compatibility rows are visible only after the base and fibre rows have been named.

The structural rows T, R, P, N transport, replay, source, and name the same packet. Since no completed Hilbert space, smooth manifold, unrestricted connection, curvature equation, hidden measurement oracle, or unification theorem is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

The Lean follow-up for $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ must provide `Nontrivial` and `FieldFaithful` instances for the same finite carrier $Y = (B, V, Q, O, H, K, T, R, P, N)$, with witnesses for the bundle, connection, quantum-state, observable, and horizontal comparison rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuantumGRSectionBundle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuantumGRSectionBundle}^{\uparrow}$ packet is generated from a finite base-frame ledger, vertical fibre rows, quantum-state rows, observable rows, horizontal comparison rows, connection rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuantumGRSectionBundle}^{\uparrow}}$, seed)

Scope closed: Definition 9184.1 and Definition 9184.2 seed the finite section-bundle carrier and NameCert obligation surface for vertical fibre rows and horizontal comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{QuantumGRSectionBundle}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, completed Hilbert spaces, smooth spacetime manifolds, unrestricted connections, curvature equations, measurement postulates, or a physical quantum-gravity theory.

Upgrade path: Obligation closure requires checked carrier admission, vertical fibre exposure, quantum-state and observable row exactness, horizontal comparison exactness, connection compatibility, transport stability, replay, provenance, local naming, and TasteGate instances for BEDC.Derived.QuantumGRSectionBundleUp.

Chapter 9185

A Concrete Naming Certificate for $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, a bounded Cauchy subsequence is usable only when the subsequence schedule and the Cauchy modulus are displayed as finite data before any Real-facing read occurs. The $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$ packet records this extraction surface over the finite density rows of chapter 6003, the compact-real uniform-modulus sibling of chapter 6375, the dyadic arithmetic of chapter 291, the finite source windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It supplies displayed subsequence-modulus data and explicitly refuses any general sequential compactness principle without such data.

Definition 9185.1 (Bounded Cauchy subsequence-modulus carrier)**.** *A* $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, D, W, R, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded Cauchy source row, naming the dyadic bound and the Cauchy tail request that the extraction is allowed to inspect.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of dyadic centers, radii, and tolerance comparisons used by the subsequence schedule.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing exactly which source windows are opened before a subsequence index is admitted.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance rows.*
5. *Q is the finite subsequence schedule row, recording the displayed increasing index data rather than a host-level selector.*
6. *K is the compact-real subsequence-modulus row, read through the compact-real uniform-modulus sibling of chapter 6375 and through the density-net surface of chapter 6003.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after B, D, W, R, Q, K have all been displayed.*
8. *H records componentwise* hsame *transport for source, dyadic, window, readback, schedule, compact-modulus, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a requested precision through $B, D, W, R, Q, K, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{CompactRealUniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, Q, K, E, H, C, P, N$. It has no coordinate for ambient sequential compactness, countable choice, a hidden infinite subsequence selector, quotient stream equality, selected limits, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 9185.2 (Bounded Cauchy subsequence-modulus NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$ *carrier*

$$S = (B, D, W, R, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}}$ *surface consists exactly of bounded Cauchy source admission for B, dyadic tolerance visibility for D, finite StreamName window admission for W, RegSeqRat readback for R, displayed subsequence schedule admission for Q, compact-real modulus handoff for K, terminal Real seal admission for E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9185.1. The tuple $B, D, W, R, Q, K, E, H, C, P, N$ is the whole accepted packet, so the local certificate can read only those rows.

The analytic route first opens the bounded Cauchy source row B, the dyadic tolerance ledger D, and the finite window row W. The regular rational readback R is downstream of those displayed windows.

The subsequence schedule row Q and compact-real modulus row K are then admitted as finite data. They are read together with the density-net and compact-real uniform-modulus sibling surfaces, so the packet exposes a displayed extraction route rather than a hidden choice schedule.

Rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for ambient sequential compactness, countable choice, quotient streams, selected limits, host equality, or ambient Real completeness, no additional NameCert obligation can be exported. □

Theorem 9185.3 (Bounded Cauchy subsequence-modulus regular window)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$ *packet receives subsequence evidence only through the displayed finite route*

$$\mathsf{FiniteCauchyDensityNet}^{\uparrow},\ \mathsf{CompactRealUniformModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer may inspect the bounded source row B, dyadic tolerance row D, finite window row W, regular readback row R, displayed subsequence schedule Q, compact-real modulus handoff K, terminal seal E, transport, replay, provenance, and local naming. It receives no general sequential compactness theorem, no countable-choice schedule, no hidden infinite subsequence selector, no quotient stream equality, no selected limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the NameCert obligations of Definition 9185.2. They force every accepted consumer read to expose $B, D, W, R, Q, K, E, H, C, P, N$.

The density-net and compact-real modulus rows supply the finite extraction context: D, W, R give the visible dyadic net rows and regular rational readback, while Q, K display the subsequence schedule and its modulus data.

The terminal Real seal E is reachable only after those rows are present. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Because the carrier has no coordinate for a host subsequence selector, countable choice, general sequential compactness, quotient equality, selected limits, or ambient Real completeness, the displayed regular-window route exhausts what a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can receive. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}$ packet is generated from a bounded Cauchy source row, DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, displayed subsequence schedule data, compact-real modulus data, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9185.1, Definition 9185.2, and Definition 9185.3 seed the finite bounded Cauchy subsequence-modulus route through density nets, compact-real modulus data, dyadic windows, RegSeqRat readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCauchySubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close ambient sequential compactness, countable choice, hidden infinite subsequence selection, quotient stream equality, selected limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bounded-source admission, density-net row visibility, compact-real modulus handoff, subsequence schedule exactness, regular-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BoundedCauchySubsequenceModulusUp.

Chapter 9186

A Concrete Naming Certificate for $\mathsf{BishopCauchySpace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchySpace}^{\uparrow}$ packet presents the completion-facing Cauchy-space substrate as finite NameCert data before quotient completion or countable selection is invoked. It reads the dyadic tolerance arithmetic of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal Real name of chapter 38, the finite density handoff of chapter 6003, and the general completion interface of chapter 95. The packet names the displayed Cauchy-space infrastructure; it is not a quotient construction, a selected limit, countable choice, host equality of reals, or ambient completeness.

Definition 9186.1 (BishopCauchySpace carrier)**.** *A* $\mathsf{BishopCauchySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to that tolerance;*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational Cauchy data observed through W;*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row reached only through D, W, S;*
5. *E is the* $\mathsf{Completion}^{\uparrow}$ *exposure row, with the finite dense handoff constrained by* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$*;*
6. *H records componentwise* hsame *transport for tolerance, window, regular readback, Real readback, completion exposure, and local packet rows;*
7. *C records displayed* Cont *replay from dyadic tolerance through stream windows, regular rational reads, Real reads, finite density handoff, and completion exposure;*
8. *P records* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchySpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for quotient stream equality, an unshown representative, countable choice, host real equality, or an ambient completeness oracle.

Theorem 9186.2 (BishopCauchySpace NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchySpace}^{\uparrow}$ *carrier*

$$B = (D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchySpace}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, dyadic tolerance readback through* D*, finite stream-window readback through* W*, regular rational Cauchy readback through* S*, Real readback through* R*, completion exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 9186.1. The rows D, W, S, R, E are the completion-facing Cauchy-space route, while H, C, P, N preserve, replay, source, and name that route.

Read the dependencies in order. The dyadic row fixes the tolerance, the stream row fixes the finite observation window, the regular-rational row supplies the Cauchy readback, and the Real row is exposed only after those rows have been displayed.

The completion row E is therefore a finite exposure surface rather than an ambient completion principle. The density handoff is constrained by $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, and the general $\mathsf{Completion}^{\uparrow}$ interface is read only through the displayed rows.

No further NameCert obligation can be projected, because the carrier has no quotient, representative-selection, host-equality, countable-choice, or ambient-completeness coordinate. □

Theorem 9186.3 (BishopCauchySpace finite-density handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCauchySpace}^{\uparrow}$ *carrier*

$$B = (D, W, S, R, E, H, C, P, N)$$

must factor its finite density read through the same $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* S*,* $\mathsf{Real}^{\uparrow}$ *readback row* R*, and* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *handoff recorded by* E*. The handoff may expose the displayed* $\mathsf{Completion}^{\uparrow}$ *route, but it may not select quotient representatives, import countable choice, compare host reals, or assert ambient completeness outside the finite packet.*

Proof. Start with the NameCert obligations of Definition 9186.2. They identify the completion-facing surface as the rows $D, W, S, R, E, H, C, P, N$ and no others.

The finite density handoff is carried by E and sourced by the provenance row P. Thus a consumer reaches the completion interface only after preserving the dyadic tolerance, stream window, regular rational readback, and Real readback that E exposes.

The structural rows H, C, N keep that route componentwise, replay it as displayed continuation data, and name it locally. Since quotient representatives, countable choice, host real equality, and ambient completeness are absent from the carrier, no downstream completion consumer can recover them from the finite density handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchySpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchySpace}^{\uparrow}$ packet is generated from dyadic tolerance, stream-window, regular-rational readback, Real readback, finite density handoff, completion exposure, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchySpace}^{\uparrow}}$, seed)

Scope closed: Definition 9186.1, Definition 9186.2, and Definition 9186.3 seed the finite Bishop Cauchy-space packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchySpace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient completion, selected representatives, countable choice, host Real equality, or ambient completeness.

Upgrade path: The next paper step is an obligation theorem separating the finite Cauchy-space rows from quotient completion and representative selection.

Chapter 9187

A Concrete Naming Certificate for $\mathsf{TukeyDepth}^{\uparrow}$

Following Tukey's halfspace-depth construction, the $\mathsf{TukeyDepth}^{\uparrow}$ packet records the finite robust-statistics surface by which a displayed point is tested against inspected affine halfspace windows before any data-depth consumer reads a centrality score. It sits after the real-name seal of chapter 38, the convex-set carrier of chapter 208, and the bounded linear functional surface of chapter 1644. The packet names halfspace depth as finite NameCert data only; it does not import host Euclidean measure, arbitrary infima over all halfspaces, selected minimizers, quotient equality of samples, or bridge checking.

Definition 9187.1 (Tukey depth carrier)**.** *A* $\mathsf{TukeyDepth}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, X, H, S, M, W, T, C, P, N)$$

with the following visible rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal used for displayed depth values and threshold reads.*
2. *X is the inspected finite sample row, carrying only the displayed data window and its local membership reads.*
3. *H is the affine halfspace-window row, represented by a displayed functional threshold and the induced side test.*
4. *S is the convex support row, tying each inspected halfspace to the* $\mathsf{ConvexSet}^{\uparrow}$ *boundary read used by the depth comparison.*
5. *M is the mass or count ledger for the inspected sample points lying in the displayed halfspace window.*
6. *W is the finite window row listing the halfspace probes actually read by the packet.*
7. *T records the transported comparison between the displayed point, the halfspace side test, and the scalar depth read.*
8. *C records componentwise* hsame *transport for $R, X, H, S, M, W, T, P, N$.*
9. *P is the* Pkg *provenance over the real, convex-set, linear-functional, sample-window, and local depth rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TukeyDepth}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, X, H, S, M, W, T, C, P, N$. It has no coordinate for a host measure space, an infinite halfspace family, an attained infimum, a selected centerpoint, quotient sample equality, or bridge credentials.

Lean4 (statement) `BEDC.Derived.TukeyDepthUp`

Theorem 9187.2 (Tukey depth NameCert obligations)**.** *For every accepted* $\mathsf{TukeyDepth}^{\uparrow}$ *carrier*

$$D = (R, X, H, S, M, W, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TukeyDepth}^{\uparrow}}$ *surface consists exactly of real-scalar admission through R, sample-window exposure through X, halfspace-window exposure through H, convex-support exposure through S, displayed count or mass comparison through M, finite probe listing through W, depth-read transport through T, componentwise* hsame *transport through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9187.1. The packet displays exactly the real scalar seal, sample window, halfspace window, convex support, count ledger, finite probe list, transported comparison, provenance, and local naming rows.

The halfspace read is not detached from the convex and scalar rows. The row H supplies the inspected side test, S ties that test to the convex-set boundary, and R, M, T expose the scalar comparison read used as a depth value.

Finally C, P, N transport, source, and name the same finite route. Since no coordinate ranges over all host halfspaces or all subsets of the sample, the listed rows exhaust the local NameCert obligations. □

Theorem 9187.3 (Tukey depth halfspace-window route)**.** *Every Tukey-depth read exported by an accepted* $\mathsf{TukeyDepth}^{\uparrow}$ *packet must expose the displayed* $\mathsf{Real}^{\uparrow}$ *scalar seal of chapter 38, the* $\mathsf{ConvexSet}^{\uparrow}$ *halfspace boundary of chapter 208, and the linear-functional comparison route of chapter 1644 before a depth consumer reads the scalar centrality value. The exported route is the finite sequence $R, X, H, S, M, W, T, C, P, N$, not an ambient infimum over all halfspaces, a host probability measure, or a selected deepest point.*

Proof. Start with the displayed halfspace window. The row H gives only the inspected functional-threshold side test, and W lists the finite probes available to the packet.

Read the geometric and scalar support next. The convex-set row S keeps the halfspace boundary visible, while the real row R and count ledger M expose the scalar comparison used by the depth read.

The transport row T and structural rows C, P, N replay that same finite window to downstream consumers. Because the carrier lacks coordinates for host measure, arbitrary halfspace families, selected minimizers, or quotient sample equality, the depth read remains a finite BEDC window route. □

Falsifiable prediction: Every accepted $\mathsf{TukeyDepth}^{\uparrow}$ read exposes a $\mathsf{Real}^{\uparrow}$ scalar seal, finite sample row, displayed halfspace-window row, $\mathsf{ConvexSet}^{\uparrow}$ support row, count or mass ledger, finite probe list, transport, provenance, and local naming before any data-depth consumer reads centrality.

Independence witness: real, convexset, bounded-linear-operator: $\mathsf{TukeyDepth}^{\uparrow}$ is not bijective to any listed sibling because it binds sample-window counts and halfspace probe windows into one robust-statistics NameCert carrier rather than merely naming scalars, convex sets, or functionals.

Closure status for $\mathsf{NameCert}_{\mathsf{TukeyDepth}^{\uparrow}}$.

Constructive story: A $\mathsf{TukeyDepth}^{\uparrow}$ packet is generated from a Real scalar seal, a finite sample row, an affine halfspace-window row, a ConvexSet support row, a count or mass ledger, finite probe listing, scalar-depth transport, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TukeyDepth}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9187.1, Definition 9187.2, and Definition 9187.3 seed the finite halfspace-depth carrier over real scalar, sample-window, convex halfspace, count-ledger, transport, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TukeyDepth}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host probability measure, arbitrary infima over all halfspaces, selected deepest points, quotient sample equality, Euclidean compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, halfspace-window exactness, count-ledger exactness, scalar-depth transport, provenance, local naming, and non-escape for BEDC.Derived.TukeyDepthUp.

Chapter 9188

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseLens}^{\uparrow}$

The $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ packet realizes the observer-expansion window of chapter 9463 as finite boundary-shadow, Busemann-displacement, Real-seal, transport, replay, provenance, and local naming rows. It sits beside the boundary-shadow packet of chapter 3813, the Busemann boundary surface of chapter 3732, the visual-boundary readout of chapter 4181, the observer-expansion packet of chapter 4167, and the observer-shadow metric route of chapter 4154. The packet names a finite observer-window lens; it does not construct a global hyperbolic model, quotient an ideal boundary, choose geodesic rays, identify physical spacetime, or bridge-check the route.

Definition 9188.1 (Hyperbolic phase lens carrier)**.** *A* $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, U, V, D, R, W, H, C, P, N)$$

with the following visible rows:

1. *B is the boundary-shadow source row, exposing the displayed shadow window read from a finite observer boundary packet.*
2. *U is the Busemann-displacement row, recording the signed displacement comparison used by the lens.*
3. *V is the visual-boundary row, keeping the boundary endpoint readback visible before any phase interpretation.*
4. *D is the observer-distance row, tying the phase lens to the observer-shadow metric route.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal for the displayed scalar comparison attached to U and D.*
6. *W is the finite observer-window row listing the shadows, displacements, and scalar reads inspected by the packet.*
7. *H records componentwise* hsame *transport for $B, U, V, D, R, W, C, P, N$.*
8. *C records displayed* Cont *replay from boundary shadow through Busemann displacement, visual-boundary readback, observer-distance comparison, Real sealing, and local naming.*
9. *P is the* Pkg *provenance over the hyperbolic shadow, Busemann, visual-boundary, observer-expansion, Real-seal, transport, and replay rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLens}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, U, V, D, R, W, H, C, P, N$. It has no coordinate for a global hyperbolic space, quotient ideal boundary, chosen geodesic ray family, physical observer law, ambient compactification, or bridge credential.

Lean4 (statement) `BEDC.Derived.HyperbolicPhaseLensUp`

Theorem 9188.2 (Hyperbolic phase lens NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ *carrier*

$$L = (B, U, V, D, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLens}^{\uparrow}}$ *surface consists exactly of boundary-shadow exposure through B, Busemann-displacement exposure through U, visual-boundary readback through V, observer-distance comparison through D, Real sealing through R, finite observer-window listing through W, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9188.1. The packet displays the shadow, displacement, boundary, observer-distance, Real-seal, window, transport, replay, provenance, and naming rows as its whole public surface.

The observer-window read is chained rather than detached. Boundary-shadow data enters through B, Busemann displacement through U, visual-boundary readback through V, and observer-distance comparison through D before the Real-sealed scalar row R can be consumed.

Finally W, H, C, P, N list, transport, replay, source, and name the same finite route. Since no coordinate supplies a global boundary quotient or selected geodesic family, the listed rows exhaust the NameCert obligations. □

Theorem 9188.3 (Hyperbolic phase lens observer-window route)**.** *Every observer-expansion read exported by an accepted* $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ *packet must first pass through chapter 3813, chapter 3732, chapter 4181, and chapter 4154. The exported lens is the finite route $B, U, V, D, R, W, H, C, P, N$, not a global hyperbolic compactification, a quotient ideal-boundary object, a physical observer law, or a selected geodesic-ray representative.*

Proof. Read the hyperbolic siblings in dependency order. The boundary-shadow row B supplies the finite shadow window, while U and V keep the Busemann and visual-boundary rows visible before a phase-lens read is public.

The observer-distance and Real rows then bind the scalar comparison. Rows D and R make the observer-shadow metric and Real seal explicit, and W records the finite observer windows actually inspected by the packet.

The structural rows H, C, P, N transport, replay, source, and name that same route. Because the packet has no coordinates for global compactification, quotient endpoints, geodesic-ray choice, or physical spacetime semantics, the exported lens remains a finite observer-window route. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ read exposes boundary-shadow, Busemann-displacement, visual-boundary, observer-distance, Real-seal, finite observer-window, transport, replay, provenance, and local naming rows before any observer-expansion consumer reads the lens.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-observer-expansion, hyperbolic-observer-shadow-metric: $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ is not bijective to any listed sibling because it binds shadow, displacement, visual-boundary, observer-distance, and Real-seal rows into one finite lens packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLens}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPhaseLens}^{\uparrow}$ packet is generated from a boundary-shadow source row, a Busemann-displacement row, a visual-boundary row, an observer-distance row, a Real seal, a finite observer-window listing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLens}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9188.1, Definition 9188.2, and Definition 9188.3 seed the finite hyperbolic observer-window lens over boundary-shadow, Busemann-displacement, visual-boundary, observer-distance, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseLens}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global hyperbolic compactification, quotient ideal-boundary equality, selected geodesic rays, physical observer laws, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, Busemann-displacement exactness, observer-window listing, Real-seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.HyperbolicPhaseLensUp.

Chapter 9189

A Concrete Naming Certificate for $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ packet records the finite guard that must be displayed before a regular Cauchy reciprocal or inverse read is admitted. It sits after the regular rational source of chapter 301, the dyadic lower-bound ledger of chapter 291, the reciprocal budget route of chapter 703, the terminal real-name seal of chapter 38, the located apartness route of chapter 6730, and the guarded reciprocal surface of chapter 6251. The packet is a denominator-guard certificate for inspected finite windows; it is not trichotomy, quotient real equality, total inversion on all $\mathsf{Real}^{\uparrow}$ names, countable choice, or ambient completion.

Definition 9189.1 (Regular Cauchy inverse guard carrier)**.** *A* $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, A, D, L, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose denominator-facing tail is being inspected.*
2. *A is the located apartness row, exposing a positive separation from zero for the inspected tail.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the finite dyadic lower-bound, tolerance, and reciprocal-radius data read from the same inspected window.*
4. *L is the* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *handoff row tying Q, A, D to the located-apartness surface.*
5. *B is the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row consumed by the guarded reciprocal or inverse read.*
6. *R is the* $\mathsf{RealReciprocal}^{\uparrow}$ *row, available only after Q, A, D, L, B have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the guarded reciprocal candidate.*
8. *H records componentwise* hsame *transport for the source, apartness, dyadic ledger, located-apartness handoff, reciprocal budget, reciprocal row, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from source inspection through located apartness, dyadic lower-bound reading, reciprocal-budget selection, reciprocal handoff, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, A, D, L, B, R, E, H, C, P, N$. It has no coordinate for a total reciprocal operation, quotient equality of regular names, trichotomy, selected denominators, countable choice, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 9189.2 (Regular Cauchy inverse guard NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ *carrier $G = (Q, A, D, L, B, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, located apartness row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound ledger,* $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$ *handoff,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row,* $\mathsf{RealReciprocal}^{\uparrow}$ *guarded reciprocal row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 9189.1. The analytic rows are Q, A, D, L, B, R, E, and the structural rows are H, C, P, N.

Read the guard rows first. The regular rational source Q is inspected only through the displayed apartness row A and dyadic ledger D, and the located-apartness handoff L records that this lower-bound data is visible before any reciprocal budget is consumed.

Next read the reciprocal side. The budget row B can feed the guarded reciprocal row R only after Q, A, D, L are already present, and the terminal seal E receives only that guarded read.

Finally the transport, replay, provenance, and naming rows preserve the same finite route. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 9189.3 (Regular Cauchy inverse guard finite denominator bound)**.** *Every accepted* $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ *carrier $G = (Q, A, D, L, B, R, E, H, C, P, N)$ exposes the finite denominator guard*

$$Q, A, D, L \longmapsto B \longmapsto R \longmapsto E$$

before any reciprocal or inverse read is public. Thus a consumer that reaches R or E without the $\mathsf{RegSeqRat}^{\uparrow}$ *source row, located apartness row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound ledger, and located-apartness handoff remains outside the local classifier.*

Proof. Start from Definition 9189.2. It places Q, A, D, L before the reciprocal budget row B in the displayed packet.

The dyadic row D is the finite denominator-bound ledger. It records the lower-bound and tolerance data read from the inspected regular Cauchy window, while L ties that ledger to the located-apartness surface.

The reciprocal rows are downstream. A consumer must pass from Q, A, D, L to B, then to the guarded reciprocal row R, and only then to the terminal Real seal E. The structural rows H, C, P, N transport, replay, source, and name this route without adding total inversion, quotient equality, trichotomy, countable choice, or ambient completion data. □

Theorem 9189.4 (Regular Cauchy inverse guard reciprocal-read boundary)**.** *Every reciprocal-facing or inverse-facing consumer of an accepted* $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ *packet must read the sibling surfaces of chapter 301, chapter 291, chapter 703, chapter 6730, chapter 6251, and chapter 38 through the displayed rows $Q, A, D, L, B, R, E, H, C, P, N$. The handoff exports a finite denominator guard and no total reciprocal, no quotient real equality, no selected inverse, no trichotomy, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use the finite denominator-bound theorem to fix the route $Q, A, D, L \longmapsto B \longmapsto R \longmapsto E$.

The sibling chapters supply the meanings of those rows. The regular rational source is read through $\mathsf{RegSeqRat}^{\uparrow}$, the lower-bound ledger through $\mathsf{DyadicRatCore}^{\uparrow}$, the budget through $\mathsf{CauchyInverseBudget}^{\uparrow}$, the apartness handoff through $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$, the guarded reciprocal row through $\mathsf{RealReciprocal}^{\uparrow}$, and the terminal seal through $\mathsf{Real}^{\uparrow}$.

Since every consumer route is carried by the displayed rows and their H, C, P, N structure, no hidden reciprocal operator or completion principle can be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ reciprocal read exposes a $\mathsf{RegSeqRat}^{\uparrow}$ source, located apartness row, $\mathsf{DyadicRatCore}^{\uparrow}$ lower-bound ledger, $\mathsf{CauchyInverseBudget}^{\uparrow}$ row, $\mathsf{RealReciprocal}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}$ row is consumed.

Independence witness: regseqrat, dyadicratcore, cauchyinversebudget, regular-cauchy-located-apartness, real-reciprocal, real: $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ is not bijective to any listed sibling because it binds a regular Cauchy source, located apartness row, dyadic lower-bound ledger, reciprocal budget, guarded reciprocal row, terminal Real seal, transport, replay, provenance, and local naming as one finite inverse-guard packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyInverseGuard}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source, located apartness evidence, a $\mathsf{DyadicRatCore}^{\uparrow}$ lower-bound ledger, a located-apartness handoff, a $\mathsf{CauchyInverseBudget}^{\uparrow}$ row, a $\mathsf{RealReciprocal}^{\uparrow}$ guarded reciprocal row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9189.1, Definition 9189.2, Definition 9189.3, and Definition 9189.4 seed the finite denominator-guard surface for regular Cauchy reciprocal reads over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegularCauchyLocatedApartness}^{\uparrow}$, $\mathsf{RealReciprocal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInverseGuard}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total reciprocal construction, quotient real equality, selected inverses, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located apartness exposure, finite denominator-bound exactness, reciprocal-budget routing, guarded reciprocal handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyInverseGuardUp.

Chapter 9190

A Concrete Naming Certificate for $\mathsf{LocaleFrame}^{\uparrow}$

Following Johnstone and constructive locale theory, the $\mathsf{LocaleFrame}^{\uparrow}$ packet records the point-free topology interface as finite frame operations and finite cover-refinement data. It sits after the open-index topology of chapter 88, the meet–join order surface of chapter 55, the implication boundary of chapter 8536, and the locale-space handoff of chapter 5153; it also supplies a point-free input surface for compact-open consumers such as chapter 4138. The packet names finite generators, finite meets and joins, cover refinement, topology handoff, Heyting/frame boundary, transport, replay, provenance, and local naming without importing host powersets, selected points, quotient spaces, choice-indexed covers, or spatiality.

Definition 9190.1 (Locale frame carrier)**.** *A* $\mathsf{LocaleFrame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, M, J, C, T, B, H, R, P, N)$$

with the following displayed rows:

1. *G is the finite open-generator row, naming the basic point-free observations available to the frame.*
2. *M is the finite-meet row, recording displayed finite intersections of generators and generated opens.*
3. *J is the finite-join row, recording displayed finite unions of generators and generated opens.*
4. *C is the cover-refinement row, recording finite cover requests and finite refinement witnesses between generated-open families.*
5. *T is the topology handoff row, exposing generated-open reads to* $\mathsf{Topology}^{\uparrow}$ *without replacing the frame by a host open-set powerset.*
6. *B is the Heyting/frame boundary row, exposing implication-shaped or complemented finite fragments through* $\mathsf{HeytingAlgebra}^{\uparrow}$ *only when those fragments are displayed.*
7. *H records componentwise* hsame *transport for generator, meet, join, cover-refinement, topology handoff, Heyting/frame boundary, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a frame request through generators, finite meets, finite joins, cover refinement, topology handoff, Heyting/frame boundary, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{LocaleFrame}^{\uparrow}$, $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{HeytingAlgebra}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocaleFrame}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, M, J, C, T, B, H, R, P, N$. It has no coordinate for host powersets of opens, selected points, quotient spatial presentations, arbitrary joins, choice-indexed covers, ambient completeness, or a global Boolean algebra of all opens.

Theorem 9190.2 (Locale frame NameCert obligations)**.** *For every accepted* $\mathsf{LocaleFrame}^{\uparrow}$ *carrier*

$$F = (G, M, J, C, T, B, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocaleFrame}^{\uparrow}}$ *surface consists exactly of finite open-generator admission through G, finite-meet exposure through M, finite-join exposure through J, cover-refinement exactness through C, topology handoff through T, Heyting/frame boundary exposure through B, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 9190.1. The rows G, M, J, C display the point-free frame data before any topology-facing or implication-facing consumer is allowed to read the packet.

The topology handoff is exactly the row T. It sends generated-open reads to $\mathsf{Topology}^{\uparrow}$ but does not create a host powerset of all opens, a selected point carrier, or a quotient spatial presentation.

The Heyting/frame boundary is exactly the row B. It may expose implication-shaped or complemented finite fragments through $\mathsf{HeytingAlgebra}^{\uparrow}$, but only for fragments already displayed by the generator, meet, join, and cover-refinement rows.

Finally H, R, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for arbitrary joins, choice-indexed covers, ambient completeness, or global Boolean openness, the listed rows exhaust the local NameCert surface. □

Theorem 9190.3 (Locale frame finite-meet cover route)**.** *Every accepted* $\mathsf{LocaleFrame}^{\uparrow}$ *read that compares a finite meet against a cover-refinement request factors through the finite route*

$$G \longmapsto M \longmapsto J \longmapsto C \longmapsto T \longmapsto B \longmapsto H \longmapsto R \longmapsto P \longmapsto N.$$

A $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{HeytingAlgebra}^{\uparrow}$, *or* $\mathsf{CompactOpenTopology}^{\uparrow}$ *consumer may inherit only the displayed generator, finite-meet, finite-join, cover-refinement, topology handoff, Heyting/frame boundary, transport, replay, provenance, and local naming rows. The route exports no host powerset, selected point, quotient spatial carrier, arbitrary cover choice, ambient compactness theorem, or global Boolean algebra of opens.*

Proof. Start with the generator row G. A finite-meet comparison has no accepted source until the basic open observations are displayed and the finite-meet row M records the intersection being inspected.

Add the finite-join and cover-refinement rows. The row J keeps generated finite unions in the same point-free ledger, and C records the finite cover family and its refinement witness before any topology-facing consumer may use the comparison.

Only after those rows are fixed may the handoff rows be read. The topology handoff T exposes generated-open data to $\mathsf{Topology}^{\uparrow}$, while the Heyting/frame boundary B exposes implication-shaped or complemented finite fragments without turning the whole packet into a Boolean algebra of opens.

The structural rows H, R, P, N preserve the same route by transport, replay, provenance, and local naming. Thus every finite-meet cover read is point-free and finite before any sibling consumer receives it. □

Falsifiable prediction: Every accepted $\mathsf{LocaleFrame}^{\uparrow}$ read exposes finite open generators, finite meets, finite joins, cover-refinement, topology handoff, Heyting/frame boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a topology, lattice, Heyting, locale-space, or compact-open consumer may use the frame.

Independence witness: topology, lattice, heytingalgebra, localespace, compactopentopology: $\mathsf{LocaleFrame}^{\uparrow}$ is not bijective to any listed sibling because it binds generators, finite frame operations, cover-refinement, topology handoff, Heyting/frame boundary, transport, replay, provenance, and local NameCert rows in one point-free packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocaleFrame}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocaleFrame}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 9190.1, Definition 9190.2, and Definition 9190.3 seed the point-free finite-frame packet over generator, meet, join, cover-refinement, topology handoff, Heyting/frame boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocaleFrame}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets of opens, selected points, quotient spatial presentations, arbitrary joins, choice-indexed covers, spatiality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite meet and join exactness, cover-refinement stability, topology handoff, Heyting/frame boundary restriction, transport, replay, provenance, and local naming for BEDC.Derived.LocaleFrameUp.

Constructive story: A $\mathsf{LocaleFrame}^{\uparrow}$ packet is generated from finite open generators, finite meet rows, finite join rows, cover-refinement rows, topology handoff rows, Heyting/frame boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 9191

A Concrete Naming Certificate for the Fibonacci-Window Weil Positivity Matrix

Following Weil and Bombieri, this packet records a finite positive-semidefiniteness shadow of the Weil explicit-formula criterion. The carrier is the displayed degree-3 Fibonacci-hat test window together with a concrete rational approximant of the Weil quadratic form on that window. It is a finite rational matrix certificate, not a global positivity theorem for all test functions.

The matrix is the scaled integer form

$$W = \begin{pmatrix} 303318 & -12152 & -193136 \\ -12152 & 303318 & -12152 \\ -193136 & -12152 & 303318 \end{pmatrix}.$$

For $x = (x_0, x_1, x_2)$ the represented quadratic form is

$$x^{\mathsf{T}} W x = 303318(x_0^2 + x_1^2 + x_2^2) - 24304(x_0x_1 + x_1x_2) - 386272x_0x_2.$$

The corner entry is genuinely negative, so positivity is not termwise positivity.

Definition 9191.1 (Fibonacci-window Weil matrix packet)**.** *The finite packet consists of the rational matrix W, the coordinate row x_0, x_1, x_2, the symmetric coordinates*

$$u = x_0 + x_2, \qquad v = x_0 - x_2, \qquad w = x_1,$$

and the two positive-semidefiniteness subcertificates

$$B = \begin{pmatrix} 110182 & -24304 \\ -24304 & 606636 \end{pmatrix}, \qquad d = 496454.$$

The displayed determinant comparison is

$$(-24304)^2 = 590684416 \le 66840367752 = 110182 \cdot 606636.$$

The minimum-eigenvalue subcertificate uses the shifted integer matrix $W - 100000I$:

$$B_{\mathrm{sh}} = \begin{pmatrix} 10182 & -24304 \\ -24304 & 406636 \end{pmatrix}, \qquad d_{\mathrm{sh}} = 396454,$$

with

$$(-24304)^2 = 590684416 \le 4140367752 = 10182 \cdot 406636.$$

Theorem 9191.2 (Fibonacci-window Weil matrix positive definiteness)**.** *For all rational x_0, x_1, x_2,*

$$x^{\mathsf{T}} W x \geq 0.$$

More precisely, the exact rational identity

$$\begin{aligned} 2(x^{\mathsf{T}} W x) = (110182)u^2 + 2(-24304)uw + (606636)w^2 \\ + (496454)v^2 \end{aligned}$$

holds for $u = x_0 + x_2$, $v = x_0 - x_2$, and $w = x_1$. The first line is nonnegative by the 2×2 positive-semidefiniteness criterion for B, and the second line is nonnegative by the 1×1 criterion for d.

The same coordinates certify the minimum-eigenvalue lower bound for the scaled integer form:

$$x^{\mathsf{T}} W x \geq 100000(x_0^2 + x_1^2 + x_2^2).$$

Equivalently, the inspected rational Weil matrix $10^{-9}W$ satisfies

$$10^{-9} W \succeq 10^{-4} I.$$

Indeed,

$$\begin{aligned} 2x^{\mathsf{T}}(W - 100000I)x = (10182)u^2 + 2(-24304)uw + (406636)w^2 \\ + (396454)v^2, \end{aligned}$$

and the shifted 2×2 block and shifted scalar block are nonnegative by the displayed determinant and scalar comparisons.

Proof. The displayed change of variables separates the symmetric and antisymmetric coordinates for the endpoints x_0 and x_2. Expanding u^2, v^2, and uw gives

$$\begin{aligned} &(110182)(x_0 + x_2)^2 + 2(-24304)(x_0 + x_2)x_1 + (606636)x_1^2 \\ &\quad + (496454)(x_0 - x_2)^2 \\ &= 606636(x_0^2 + x_1^2 + x_2^2) - 48608(x_0x_1 + x_1x_2) - 772544x_0x_2 \\ &= 2(x^{\mathsf{T}} W x). \end{aligned}$$

The diagonal entries 110182 and 606636 are nonnegative, and the determinant comparison in Definition 9191.1 verifies the 2×2 block. The scalar coefficient 496454 is nonnegative. Therefore the right-hand side is a sum of a nonnegative 2×2 quadratic block and a nonnegative 1×1 block. Since $2 > 0$, the original quadratic form is nonnegative.

For the lower bound, replace the diagonal coefficient 303318 by $203318 = 303318 - 100000$ and keep the off-diagonal coefficients unchanged. The same expansion gives

$$\begin{aligned} &(10182)(x_0 + x_2)^2 + 2(-24304)(x_0 + x_2)x_1 + (406636)x_1^2 \\ &\quad + (396454)(x_0 - x_2)^2 \\ &= 406636(x_0^2 + x_1^2 + x_2^2) - 48608(x_0x_1 + x_1x_2) - 772544x_0x_2 \\ &= 2x^{\mathsf{T}}(W - 100000I)x. \end{aligned}$$

The shifted diagonal entries 10182 and 406636 are nonnegative, the determinant comparison for B_{sh} verifies the shifted 2×2 block, and 396454 verifies the shifted 1×1 block. Hence $x^{\mathsf{T}}(W - 100000I)x \geq 0$, which is the displayed lower bound. □

Lean4 ✓ `BEDC.Derived.RHRoute.FibonacciWindowWeilMatrixPositivity.fibonacci window weil matrix positivity certificate`

Theorem 9191.3 (Finite Weil-positivity shadow boundary)**.** *The packet is a finite Weil-positivity shadow: a concrete rational approximant of the degree-3 Fibonacci-windowed Weil quadratic form is positive semidefinite despite genuine negative off-diagonal entries, so positivity is a nontrivial cancellation certified by symmetric diagonalization onto the* 2×2 *and* 1×1 *positive-semidefiniteness criteria.*

Proof. The rational matrix $W/10^9$ is the inspected approximant to the Weil functional $T(\psi_r * \psi_s^\sharp)$ on the three-dimensional Fibonacci-hat test space. Its entries are computed from the von Mangoldt prime contribution for $n \le F_5^2 = 25$ together with the archimedean Γ-factor logarithmic-derivative density. This construction is non-circular: it does not input zeta zero locations.

The Lean theorem is exact only after the rational entries are fixed. It proves the rational positive-semidefiniteness statement, the eigenvalue lower bound $W \succeq 10^{-4}I$ for the normalized matrix, and the displayed change of variables. The eigenvalue lower bound $W \succeq 10^{-4}I$, together with the external entry-certification error of order $10^{-9} \ll 10^{-4}$, shows the exact degree-3 Fibonacci-windowed Weil matrix is positive definite, not merely that the rational approximant is positive semidefinite. □

Falsifiable prediction: Recomputing the displayed rational matrix, the determinant comparisons $590684416 \le 66840367752$ and $590684416 \le 4140367752$, the diagonalization identities for $u = x_0 + x_2$, $v = x_0 - x_2$, $w = x_1$, the shifted lower-bound identity, or the exact Lean certificate refutes this packet if any listed equality or inequality fails.

Independence witness: fibonacci-window-weil-matrix-positivity: this finite packet is not a global Weil-positivity theorem, not a zeta-zero computation, and not a proof of RH; it is the exact rational PSD and minimum-eigenvalue certificate for one degree-3 Fibonacci-hat approximant.

Closure status for $\mathsf{NameCert}_{FibonacciWindowWeilMatrixPositivity}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{FibonacciWindowWeilMatrixPositivity}$, scopedClosed)

Scope closed: positive semidefiniteness and the lower bound $10^{-9}W \succeq 10^{-4}I$ for a concrete 3×3 rational approximant of the degree-3 Fibonacci-windowed Weil quadratic form, via symmetric diagonalization onto the 2×2 and 1×1 criteria

Formal verification: FormalStatus($\mathsf{NameCert}_{FibonacciWindowWeilMatrixPositivity}$, theoremChecked)

Lean target: `BEDC.Derived.RHRoute.FibonacciWindowWeilMatrixPositivity.fibonacci\_window\_weil\_matrix\_positivity`

Bridge status: none

Not claimed: a finite Weil-positivity shadow on a 3-dimensional Fibonacci-hat test subspace; not a proof of RH; entry values are external interval numerics, the Lean theorem is exact rational positive semidefiniteness and a minimum-eigenvalue lower bound

Upgrade path: raise the window degree m (minimum eigenvalue shrinks $\sim F_m^{-2}$, needing higher-precision certified rational enclosures) and discharge the WeilPositivityRoute reduction obligations

Part VI

Concrete Hardening

Chapter 9192

Internal Domain Judgments and Signature Token Reflection

This chapter replaces two external-looking concrete notations by internal judgments:

$$h \in D, \qquad \mathsf{Tok}_{\Pi}(s).$$

They are readable, but both carry too much external mathematical flavor. This chapter replaces them with internal judgments.

Definition 9192.1 (Internal domain judgment)**.** *A concrete source domain is not treated as a set. It is a source name D equipped with a judgment*

$$\mathsf{InDom}(D, h),$$

read as: the history h is admitted by the source domain name D.

Lean4 ✓ `BEDC.FKernel.Gap.InDom iff setup field`

Remark 9192.2 (Layer discipline)**.** *The notation $\mathsf{InDom}(D, h)$ is still written in the metalanguage, but it is the public BEDC judgment to be used in all concrete gap statements. The notation $h \in D$ is not used in the active concrete presentation.*

Definition 9192.3 (Signature token introduction)**.** *For a probe bundle Π and an answer history s, a visible package token is introduced by a judgment*

$$\mathsf{TokIntro}(\Pi, s, p).$$

It means that p is a package token introduced from the signature history s under bundle Π.

Lean4 ✓ `BEDC.FKernel.Package.TokIntro iff setup field`
Lean4 (definition) `BEDC.FKernel.Package.SignatureTokenIntro`
↪ variant `BEDC.FKernel.Package.SignatureTokenIntro iff TokIntro`

Definition 9192.4 (Concrete signature package)**.** *The concrete signature package relation is defined by token introduction:*

$$\mathsf{Pkg}^{\mathrm{sig}}(\Pi, s, p) \quad \Longleftrightarrow \quad \mathsf{TokIntro}(\Pi, s, p).$$

Lean4 (definition) `BEDC.FKernel.Package.PkgSig`

Lean4 ✓ `BEDC.FKernel.Package.PkgSig iff TokIntro`

↪ variant `BEDC.FKernel.Package.TokIntro iff PkgSig`

↪ variant `BEDC.FKernel.Package.PkgSig intro from token`

↪ variant `BEDC.FKernel.Package.TokIntro nonempty pkg`

Definition 9192.5 (Concrete package-sameness policy)**.** *For the signature-token instance, package-sameness is generated only from history-sameness of source signatures:*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t) \quad \Rightarrow \quad \mathsf{psame}(p, q).$$

No additional package-sameness constructors are admitted for the concrete signature-token instance.

Theorem 9192.6 (Package reflection for signature tokens)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$ *for the concrete signature-token package policy. If*

$$\mathsf{TokIntro}(\Pi, s, p), \qquad \mathsf{TokIntro}(\Pi, t, q), \qquad \mathsf{psame}(p, q),$$

then

$$\mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.FKernel.Package.package reflection`

↪ variant `BEDC.FKernel.Package.package reflection forward`

↪ variant `BEDC.FKernel.Package.package reflection roundtrip pair`

↪ variant `BEDC.FKernel.Package.package reflection iff hsame`

↪ variant `BEDC.FKernel.Package.psame reflection witness chain under tok unique`

Proof. Apply the reflection field of $\mathsf{PackageTokenPolicy}(\Pi)$ to the displayed token-introduction judgments and the package-sameness judgment. In the base-relation presentation, inversion on the constructor exposes introducing signatures for p and q; the policy reflection field, or a token-uniqueness theorem supplying that field, transports those constructor witnesses to the displayed introductions and yields $\mathsf{hsame}(s, t)$. □

Corollary 9192.7 (Concrete package equivalence is signature-grounded)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$ *for the concrete signature-token package policy. If*

$$\mathsf{TokIntro}(\Pi, s, p) \qquad \textit{and} \qquad \mathsf{TokIntro}(\Pi, t, q),$$

then

$$\mathsf{psame}(p, q) \quad \Longleftrightarrow \quad \mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.FKernel.Package.package reflection iff hsame`

Proof. Apply the two fields of $\mathsf{PackageTokenPolicy}(\Pi)$ to the displayed token-introduction judgments. The reflection field sends $\mathsf{psame}(p, q)$ to $\mathsf{hsame}(s, t)$, and the soundness field sends $\mathsf{hsame}(s, t)$ back to $\mathsf{psame}(p, q)$. □

Chapter 9193

Internalized Concrete Gaps and Globalize Soundness

This chapter restates the concrete gap policy using InDom and $\mathsf{TokIntro}$. This removes the remaining set-membership notation from the concrete instance.

Definition 9193.1 (Internalized signature gap)**.** *Given a source domain name D, a bundle Π, and a package token p, define*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$$

to mean that there exist an answer history s and an evidence ledger Δ such that

$$\mathsf{InDom}(D, h), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

Lean4 ✓ `BEDC.FKernel.Gap.internalized signature gap iff`
↪ variant `BEDC.FKernel.Gap.policySupportedSignatureGap token witness`

Definition 9193.2 (Internalized concrete Globalize instance)**.** *The internalized concrete signature globalization is the triple*

$$\mathsf{Glob}^{\mathrm{sig}}(\Pi, D) := (\mathsf{Pkg}^{\mathrm{sig}}, \mathsf{psame}, \mathsf{InGap}^{\mathrm{sig}}).$$

It is the active concrete finite-kernel Globalize instance.

Lean4 (definition) `BEDC.FKernel.Gap.InternalizedConcreteGlobalizeInstance`
Lean4 ✓ `BEDC.FKernel.Gap.internalized concrete globalize instance`

Theorem 9193.3 (Internalized gap coverage)**.** *Assume $\mathsf{AskPolicy}(\Pi, D)$ and the dependent generated-signature token condition: whenever $\mathsf{InDom}(D, h)$ and $\mathsf{Sig}(\Pi, h, s, \Delta)$, there exists a token p with $\mathsf{TokIntro}(\Pi, s, p)$. If $\mathsf{InDom}(D, h)$, then there exists a token p such that*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Lean4 ✓ `BEDC.FKernel.Gap.internalized gap coverage`

Proof. From $\mathsf{AskPolicy}(\Pi, D)$ and $\mathsf{InDom}(D, h)$, obtain a generated signature event

$$\mathsf{Sig}(\Pi, h, s, \Delta).$$

By the dependent token condition, obtain p with $\mathsf{TokIntro}(\Pi, s, p)$. These data witness $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. □

Theorem 9193.4 (Internalized gap separation)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then

$$\mathsf{psame}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Gap.internalized gap separation`
↪ variant `BEDC.FKernel.Gap.concrete gap separation witnesses`
↪ variant `BEDC.FKernel.Gap.internalized gap separation witnesses`
↪ variant `BEDC.FKernel.Gap.internalized gap separation packed`
↪ variant `BEDC.FKernel.Gap.gap separation concrete`

Proof. Unpack the two gap memberships. We obtain signatures s, t and evidence ledgers Δ, Θ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p),$$
$$\mathsf{Sig}(\Pi, h, t, \Theta), \qquad \mathsf{TokIntro}(\Pi, t, q).$$

By signature determinacy under $\mathsf{AskPolicy}(\Pi, D)$, $\mathsf{hsame}(s, t)$. The concrete package-sameness introduction rule then yields $\mathsf{psame}(p, q)$. □

Theorem 9193.5 (Same-source gap memberships share signatures)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then there exist generated signatures s, t *introducing* p, q *such that* $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.inGapSig same source package witnesses`
↪ variant `BEDC.FKernel.Gap.inGapSig same source full witnesses`
↪ variant `BEDC.FKernel.Gap.inGapSig same source signature events same`
↪ variant `BEDC.FKernel.Gap.inGapSig same source sigRel witnesses`
↪ variant `BEDC.FKernel.Gap.gap memberships share signature`

Proof. Unpack the two gap memberships. Signature determinacy for their shared source history gives $\mathsf{hsame}(s, t)$, while the unpacked memberships provide the two token-introduction witnesses. □

Theorem 9193.6 (Concrete Globalize soundness)**.** *If* $\mathsf{Sig}(\Pi, h, s, \Delta)$*,* $\mathsf{Sig}(\Pi, k, t, \Theta)$*,* $\mathsf{TokIntro}(\Pi, s, p)$*,* $\mathsf{TokIntro}(\Pi, t, q)$*, and* $\mathsf{hsame}(s, t)$*, then* $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.concrete globalize soundness intro`

Proof. Apply the concrete package-sameness constructor to the two token-introduction premises and the internal signature-sameness premise. □

Theorem 9193.7 (Concrete Globalize completeness)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k), \qquad \mathsf{psame}(p, q),$$

then there exist answer histories s, t *and evidence ledgers* Δ, Θ *such that*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta), \qquad \mathsf{hsame}(s, t).$$

Proof. The two gap premises provide generated signatures and token-introduction witnesses for p and q. The package-token policy reflects $\mathsf{psame}(p, q)$ back to internal sameness of those generated signatures, giving the displayed witnesses. □

Lean4 ✓ `BEDC.FKernel.Gap.concrete globalize completeness`

↪ variant `BEDC.FKernel.Gap.globalize completeness primary`
↪ variant `BEDC.FKernel.Gap.concrete globalization completeness samesig`
↪ variant `BEDC.FKernel.Gap.concrete globalize completeness sameSig`
↪ variant `BEDC.FKernel.Gap.concrete globalize completeness with tokens`
↪ variant `BEDC.FKernel.Gap.internalized globalize completeness concrete`
↪ variant `BEDC.FKernel.Gap.internalized globalize completeness with tokens`
↪ variant `BEDC.FKernel.Gap.internalized globalize completeness hsame only`
↪ variant `BEDC.FKernel.Gap.internalized globalize completeness`
↪ variant `BEDC.FKernel.Gap.internalized globalize completeness projection`
↪ variant `BEDC.FKernel.Gap.concrete globalize completeness from package same`
↪ variant `BEDC.FKernel.Gap.globalize completeness from package reflection`
↪ variant `BEDC.FKernel.Gap.globalize completeness`

Corollary 9193.8 (Concrete Globalize classifies exactly by signatures)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$, $\mathsf{PackageTokenPolicy}(\Pi)$,

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k).$$

Then visible package sameness and internal sameness of generated answer histories agree in both orientations:

$$\mathsf{psame}(p, q) \iff \exists s, t, \Delta, \Theta.\ \mathsf{Sig}(\Pi, h, s, \Delta) \wedge \mathsf{Sig}(\Pi, k, t, \Theta) \wedge \mathsf{hsame}(s, t).$$

Symmetrically,

$$\mathsf{psame}(q, p) \iff \exists t, s, \Theta, \Delta.\ \mathsf{Sig}(\Pi, k, t, \Theta) \wedge \mathsf{Sig}(\Pi, h, s, \Delta) \wedge \mathsf{hsame}(t, s).$$

Proof. The forward directions are Theorem 9193.7 applied to the two ordered pairs of gap memberships. The reverse direction for $\mathsf{psame}(p, q)$ is Theorem 9199.5 applied to the ordered memberships for p and q. The reverse direction for $\mathsf{psame}(q, p)$ applies the same local classification theorem to the ordered memberships for q and p. In both cases, the local theorem performs the transport from the displayed existential signatures to the token-introducing signatures carried by the two gap witnesses. □

Lean4 ✓ `BEDC.FKernel.Gap.concrete globalize classifies symmetric package pair`

↪ variant `BEDC.FKernel.Gap.internalized globalize classifies iff`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies sameSig mp`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies sameSig mpr`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies by signatures`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies by signatures primary`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies sameSig`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies with token witnesses iff`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies token witness directions`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies sameSig directions`
↪ variant `BEDC.FKernel.Gap.concrete globalize classifies by signatures directions`

Theorem 9193.9 (Bundle-local internalized gap separation)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then

$$\mathsf{psame}(p, q).$$

Proof. Unpack the first gap membership using Definition 9193.1. It supplies the admitted-source judgment $\mathsf{InDom}(D, h)$, an answer history s, and an evidence ledger Δ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

Unpack the second gap membership in the same way, obtaining an answer history t and an evidence ledger Θ such that

$$\mathsf{Sig}(\Pi, h, t, \Theta), \qquad \mathsf{TokIntro}(\Pi, t, q).$$

The shared admitted-source judgment and the two generated signatures are exactly the hypotheses of Definition 6.63 under $\mathsf{BundleAskPolicy}(\Pi, D)$, so $\mathsf{hsame}(s, t)$. The concrete package-sameness introduction rule, equivalently Definition 9193.6 applied to the two generated signatures, the two token-introduction witnesses, and $\mathsf{hsame}(s, t)$, yields $\mathsf{psame}(p, q)$. □

Theorem 9193.10 (Bundle-local internalized gap coverage)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$ *and the dependent generated-signature token condition: whenever* $\mathsf{InDom}(D, u)$ *and* $\mathsf{Sig}(\Pi, u, s, \Delta)$*, there exists a token* p *with* $\mathsf{TokIntro}(\Pi, s, p)$*. If* $\mathsf{InDom}(D, h)$*, then there exists a token* p *such that*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Proof. By bundle-local signature existence, Definition 6.62, the bundle-local asking policy and the admitted-source premise give an answer history s and an evidence ledger Δ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta).$$

Apply the dependent token condition to $\mathsf{InDom}(D, h)$ and this generated signature. It supplies a token p with

$$\mathsf{TokIntro}(\Pi, s, p).$$

The admitted-source premise, the generated signature, and this token-introduction judgment are exactly the witnesses required by Definition 9193.1. Therefore

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h),$$

and the displayed token p gives the required coverage. □

Theorem 9193.11 (Bundle-local same-source gap memberships share signatures)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$ *If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then there exist answer histories s, t *and evidence ledgers* Δ, Θ *such that*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p),$$

$$\mathsf{Sig}(\Pi, h, t, \Theta), \qquad \mathsf{TokIntro}(\Pi, t, q), \qquad \mathsf{hsame}(s, t).$$

Proof. Unpack the first gap membership by Definition 9193.1. It supplies the admitted-source judgment $\mathsf{InDom}(D, h)$, an answer history s, and an evidence ledger Δ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

Unpack the second gap membership in the same way. It supplies an answer history t and an evidence ledger Θ such that

$$\mathsf{Sig}(\Pi, h, t, \Theta), \qquad \mathsf{TokIntro}(\Pi, t, q).$$

The two generated signatures have the same bundle and the same admitted source history. Applying Definition 6.63 under $\mathsf{BundleAskPolicy}(\Pi, D)$ gives $\mathsf{hsame}(s, t)$. The displayed signatures, token-introduction witnesses, and this history-sameness witness are the required data. □

Chapter 9194

Unary Continuation Commutativity and the Additive Name

The $\mathsf{Add}^{\uparrow}$ certificate establishes a unary continuation interface with unit, associativity, determinacy, and closure. To justify the additive name rather than merely a monoid-like continuation name, one more paper-level obligation is needed: unary continuation commutativity.

Definition 9194.1 (Unary histories)**.** *The unary judgment* $\mathsf{Unary}(h)$ *is generated by*

$$\mathsf{Unary}(\mathsf{e}) \qquad \textit{and} \qquad \mathsf{Unary}(h) \Rightarrow \mathsf{Unary}(\mathsf{E1}(h)).$$

No rule generates $\mathsf{Unary}(\mathsf{E0}(h))$*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary histories iff empty or eone tail`

Definition 9194.2 (Unary continuation result)**.** $\mathsf{UnaryCont}(h, k, r)$ *means*

$$\mathsf{Unary}(h), \qquad \mathsf{Unary}(k), \qquad \mathsf{Cont}(h, k, r).$$

Lean4 (definition) `BEDC.FKernel.Unary.UnaryCont`
Lean4 ✓ `BEDC.FKernel.Unary.unaryCont empty units same`
↪ variant `BEDC.FKernel.Unary.UnaryCont iff fields`

Definition 9194.3 (Unary continuation commutativity obligation)**.** *The unary continuation commutativity obligation is the statement that two unary histories continued in either order have internally same results.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryContinuationCommutativityObligation`
Lean4 ✓ `BEDC.FKernel.Unary.unary commutativity obligation induction`
↪ variant `BEDC.FKernel.Unary.unary commutativity obligation iff`
↪ variant `BEDC.FKernel.Unary.UnaryContinuationCommutativityObligation iff fields`
↪ variant `BEDC.FKernel.Unary.unary continuation commutativity obligation witnesses`

Lemma 9194.4 (Unary continuation closure and factors)**.** *If* $\mathsf{UnaryCont}(h, k, r)$*, then* $\mathsf{Unary}(r)$*. If* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{Unary}(r)$*, then the unary left and right factors are recovered as* $\mathsf{Unary}(h)$ *and* $\mathsf{Unary}(k)$*.*

Lean4 ✓ BEDC.FKernel.Unary.unary append closed
↪ variant BEDC.FKernel.Unary.unary append closed right induction
↪ variant BEDC.FKernel.Unary.unary cont closed
↪ variant BEDC.FKernel.Unary.unary cont closed left induction
↪ variant BEDC.FKernel.Unary.unary cont result closed
↪ variant BEDC.FKernel.Unary.unary continuation closure
↪ variant BEDC.FKernel.Unary.unary continuation closure spine
↪ variant BEDC.FKernel.Unary.unary repetition continuation closed
↪ variant BEDC.FKernel.Unary.unary cont right factor
↪ variant BEDC.FKernel.Unary.unary cont left factor
↪ variant BEDC.FKernel.Unary.unary append left factor
↪ variant BEDC.FKernel.Unary.unary cont factors from result
↪ variant BEDC.FKernel.Unary.unary cont result iff factors
↪ variant BEDC.FKernel.Unary.unary cont factors iff result
↪ variant BEDC.FKernel.Unary.unary cont iff result unary
↪ variant BEDC.FKernel.Unary.unary append factors iff result
↪ variant BEDC.FKernel.Unary.unary continuation closed and factors
↪ variant BEDC.FKernel.Unary.unary repetition continuation closed and factors

Proof. Induct on the derivation of $\mathsf{Cont}(h, k, r)$. The empty case gives r internally same as k, hence unary. The $\mathsf{E1}$-case preserves unary by the constructor rule. The $\mathsf{E0}$-case is incompatible with $\mathsf{Unary}(h)$ by unary no-zero-extension. □

Theorem 9194.5 (Unary append context middle equivalence)**.** *Let l, m, r : BHist. If l and r are unary histories, then unary membership of the two-sided append context is equivalent to unary membership of the middle history:*

$$\mathsf{Unary}(\mathsf{append}(l, \mathsf{append}(m, r))) \iff \mathsf{Unary}(m).$$

Proof. The forward direction applies unary append factor recovery twice: first recover the right factor of the outer append, then recover the left factor of the inner append. The reverse direction closes unary histories under append, first for the middle with the right context and then with the left context. □

Lean4 ✓ BEDC.FKernel.Unary.unary append context middle iff

Theorem 9194.6 (Unary continuation commutativity)**.** *If*

$$\mathsf{Unary}(h), \quad \mathsf{Unary}(k), \quad \mathsf{Cont}(h, k, r), \quad \mathsf{Cont}(k, h, r'),$$

then

$$\mathsf{hsame}(r, r').$$

Lean4 ✓ BEDC.FKernel.Unary.unary cont comm
↪ variant BEDC.FKernel.Unary.unary cont comm from append comm
↪ variant BEDC.FKernel.Unary.unary cont comm obligation holds
↪ variant BEDC.FKernel.Unary.unary append comm hsame
↪ variant BEDC.FKernel.Unary.unary append comm with closed results
↪ variant BEDC.FKernel.Unary.unary append comm three
↪ variant BEDC.FKernel.Unary.unaryCont comm and results closed

```
↪ variant  BEDC.FKernel.Unary.unaryCont comm closed symmetric pair
↪ variant  BEDC.FKernel.Unary.unaryCont comm sameness pair
↪ variant  BEDC.FKernel.Unary.unary cont comm symmetric pair
↪ variant  BEDC.FKernel.Unary.unary append comm right induction
↪ variant  BEDC.FKernel.Unary.unary append comm hsame left induction closed pair
```

Proof. Proceed by induction on the unary derivation of h.

If $h = \mathsf{e}$, then $\mathsf{Cont}(\mathsf{e}, k, r)$ gives $\mathsf{hsame}(r, k)$, while $\mathsf{Cont}(k, \mathsf{e}, r')$ gives $\mathsf{hsame}(r', k)$ by the right-unit theorem for continuation. Thus $\mathsf{hsame}(r, r')$.

For the step case, write $h = \mathsf{E1}(h_0)$ with $\mathsf{Unary}(h_0)$. From $\mathsf{Cont}(\mathsf{E1}(h_0), k, r)$, inversion yields a result u such that $\mathsf{Cont}(h_0, k, u)$ and $\mathsf{hsame}(r, \mathsf{E1}(u))$. The other branch $\mathsf{Cont}(k, \mathsf{E1}(h_0), r')$ is shown, by induction on the unary derivation of k, to be internally same as $\mathsf{E1}(v)$, where $\mathsf{Cont}(k, h_0, v)$. Applying the outer induction hypothesis to h_0, k gives $\mathsf{hsame}(u, v)$, hence $\mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(v))$, and therefore $\mathsf{hsame}(r, r')$.

This proof uses only unary constructors, continuation inversion, and history-sameness congruence under $\mathsf{E1}$. □

Corollary 9194.7 (Upgrade of the additive naming certificate)**.** *The certificate for* $\mathsf{Add}^{\uparrow}$ *is upgraded from a unary continuation monoid-like certificate to a commutative unary additive certificate once Theorem 9194.6 is included among its stability fields.*

```
Lean4 ✓  BEDC.FKernel.Unary.add up commutative license
↪ variant  BEDC.FKernel.Unary.add up commutative certificate upgrade
↪ variant  BEDC.FKernel.Unary.upgrade additive naming certificate
↪ variant  BEDC.FKernel.Unary.additive certificate upgrade
```

Proof. The certificate already contains unit, associativity, determinacy, closure, and composite ledger policy. Theorem 9194.6 supplies commutativity up to internal history-sameness. □

Remark 9194.8 (Naming boundary)**.** *Without Theorem 9194.6, the conservative public name is* $\mathsf{UnaryContMonoid}^{\uparrow}$*. With it, the name* $\mathsf{Add}^{\uparrow}$ *is licensed as a commutative unary addition-like interface.*

Part VII

Proof Obligations

Chapter 9195

Domain Policy and Unary Instance

9195.1 Domain Policy and Transport Obligations

This chapter records the internal domain judgment $\mathsf{InDom}(D, h)$ and the transport obligation needed for domain behavior.

Definition 9195.1 (Domain policy)**.** *A source-domain name D is equipped with a domain policy, written*

$$\mathsf{DomainPolicy}(D),$$

when the following BEDC-level transport behavior is supplied: if $\mathsf{InDom}(D, h)$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{InDom}(D, k)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.DomainPolicy transport field`

↪ variant `BEDC.FKernel.Gap.DomainPolicy iff transport and invariance`

The carrier D and judgment $\mathsf{InDom}(D, h)$ are supplied by the domain setup. Concrete gap statements use $\mathsf{InDom}(D, h)$ as their source-admission premise rather than imported set-membership notation.

Theorem 9195.2 (Domain transport)**.** *Assume* $\mathsf{DomainPolicy}(D)$*. If*

$$\mathsf{InDom}(D, h) \quad \textit{and} \quad \mathsf{hsame}(h, k),$$

then

$$\mathsf{InDom}(D, k).$$

Lean4 ✓ `BEDC.FKernel.Gap.domain transport`

↪ variant `BEDC.FKernel.Gap.domain transport refl`

↪ variant `BEDC.FKernel.Gap.domain policy transport witnesses`

↪ variant `BEDC.FKernel.Gap.DomainPolicy iff transport`

↪ variant `BEDC.FKernel.Gap.domain transport symmetric`

↪ variant `BEDC.FKernel.Gap.domain transport transitive`

↪ variant `BEDC.FKernel.Gap.domain transport three step chain`

↪ variant `BEDC.FKernel.Gap.domain transport four step chain`

↪ variant `BEDC.FKernel.Gap.domain transport backward chain`

Proof. This is the transport field of $\mathsf{DomainPolicy}(D)$. The theorem is recorded separately because later gap arguments must cite it explicitly rather than silently treating D as a set. □

Corollary 9195.3 (Domain invariance of concrete source admission)**.** *Under* $\mathsf{DomainPolicy}(D)$*, if* $\mathsf{hsame}(h, k)$*, then*

$$\mathsf{InDom}(D, h) \iff \mathsf{InDom}(D, k).$$

Lean4 ✓ `BEDC.FKernel.Gap.domain invariance of concrete source admission`

↪ variant `BEDC.FKernel.Gap.domain invariance`
↪ variant `BEDC.FKernel.Gap.DomainPolicy transport and invariance`
↪ variant `BEDC.FKernel.Gap.inGapSig domain transport with signature`
↪ variant `BEDC.FKernel.Gap.inGapSig domain transport source`
↪ variant `BEDC.FKernel.Gap.inGapSig transport existing signature witness`
↪ variant `BEDC.FKernel.Gap.inGapSig transport iff with signature witnesses`
↪ variant `BEDC.FKernel.Gap.inGapSig domain transport equiv with witnesses`
↪ variant `BEDC.FKernel.Gap.domain transport two way chain`

Proof. Immediate from Theorem 9195.2 in both directions, using symmetry of hsame for the reverse direction. The other parts of a concrete gap, namely signature generation and token introduction, require their own policies and are not bundled into domain transport. □

Corollary 9195.4 (Three-history domain transport equivalence)**.** *Under* $\mathsf{DomainPolicy}(D)$*, if* $\mathsf{hsame}(h, k)$ *and* $\mathsf{hsame}(k, l)$*, then*

$$\mathsf{InDom}(D, h) \iff \mathsf{InDom}(D, l).$$

Lean4 ✓ `BEDC.FKernel.Gap.domain transport three way equiv`

Proof. Apply Theorem 9195.2 along $\mathsf{hsame}(h, k)$ and then along $\mathsf{hsame}(k, l)$ for the forward direction. For the reverse direction, use symmetry of the same two history-sameness premises in the opposite order. □

Remark 9195.5 (Active reading)**.** *Active concrete gap statements use* $\mathsf{InDom}(D, h)$*, not* $h \in D$*. What is not automatic is any claim that an arbitrary domain name satisfies* $\mathsf{DomainPolicy}(D)$*. That must be supplied for each concrete domain.*

9195.2 Concrete Domain Policy for the Unary Source

section 9195.1 introduces $\mathsf{DomainPolicy}(D)$ as the condition allowing $\mathsf{InDom}(D, h)$ to be transported along internal history-sameness. This section records the unary source instance.

Definition 9195.6 (Unary source domain)**.** *Let* $\mathsf{UnarySource}$ *be the source-domain name whose internal domain judgment is*

$$\mathsf{InDom}(\mathsf{UnarySource}, h) \quad \textit{exactly when} \quad \mathsf{Unary}(h).$$

This is not a set-membership statement. It is a domain judgment licensed by the unary-generation rules.

Lean4 ✓ `BEDC.FKernel.Unary.unaryDomain inDom iff unaryHistory`

Lemma 9195.7 (Unary is transported by history-sameness)**.** *If*

$$\mathsf{Unary}(h), \qquad \mathsf{hsame}(h, k),$$

then

$$\mathsf{Unary}(k).$$

Lean4 ✓ `BEDC.FKernel.Unary.unary transport`
↪ variant `BEDC.FKernel.Unary.unary transport symm`
↪ variant `BEDC.FKernel.Unary.unary history hsame iff`

Proof. Proceed by induction on the derivation of $\mathsf{Unary}(h)$.

For the empty case, h is internally the empty history. Since $\mathsf{hsame}(h, k)$, closed generation and no-confusion for hsame force k to be internally the empty history. Hence $\mathsf{Unary}(k)$.

For the step case, the unary derivation has the form $\mathsf{Unary}(h_0) \Rightarrow \mathsf{Unary}(\mathsf{E1}(h_0))$. From $\mathsf{hsame}(\mathsf{E1}(h_0), k)$, constructor inversion for hsame yields a history k_0 such that k is internally $\mathsf{E1}(k_0)$ and $\mathsf{hsame}(h_0, k_0)$. By the induction hypothesis, $\mathsf{Unary}(k_0)$. Therefore $\mathsf{Unary}(\mathsf{E1}(k_0))$, and by internal sameness this is the required unary judgment for k. □

Theorem 9195.8 (Unary source satisfies DomainPolicy)**.**

$$\mathsf{DomainPolicy}(\mathsf{UnarySource}).$$

Lean4 ✓ `BEDC.FKernel.Unary.unary domain policy`

Proof. Assume $\mathsf{InDom}(\mathsf{UnarySource}, h)$ and $\mathsf{hsame}(h, k)$. By definition of $\mathsf{UnarySource}$, the first premise is $\mathsf{Unary}(h)$. Lemma 9195.7 gives $\mathsf{Unary}(k)$, hence $\mathsf{InDom}(\mathsf{UnarySource}, k)$. □

Corollary 9195.9 (Unary-domain gaps may transport sources)**.** *Any concrete gap theorem whose source domain is* $\mathsf{UnarySource}$ *may use domain transport along* hsame*, provided no additional source-domain restriction is imposed.*

Lean4 ✓ `BEDC.FKernel.Unary.unary domain gap transport`

Proof. Immediate from Theorem 9195.8 and Lemma 9195.2. □

Definition 9195.10 (Binary Boolean compound domains)**.** *Given source-domain names D_0 and D_1, a conjunctive compound domain $D_\wedge$ and a disjunctive compound domain $D_\vee$ are BEDC domain names whose internal admission judgments are defined by*

$$\mathsf{InDom}(D_\wedge, h) \iff \big(\mathsf{InDom}(D_0, h) \text{ and } \mathsf{InDom}(D_1, h)\big)$$

and

$$\mathsf{InDom}(D_\vee, h) \iff \big(\mathsf{InDom}(D_0, h) \text{ or } \mathsf{InDom}(D_1, h)\big).$$

These are compound source-domain names, not external set operations on histories.

Theorem 9195.11 (DomainPolicy is closed under binary Boolean compounds)**.** *Assume* $\mathsf{DomainPolicy}(D_0)$ *and* $\mathsf{DomainPolicy}(D_1)$*. Let $D_\wedge$ and $D_\vee$ be compound domains as in Definition 9195.10. Then*

$$\mathsf{DomainPolicy}(D_\wedge) \quad \text{and} \quad \mathsf{DomainPolicy}(D_\vee).$$

Proof. It is enough, by Definition 9195.1, to prove transport of the displayed admission judgments along hsame.

For $D_\wedge$, assume $\mathsf{InDom}(D_\wedge, h)$ and $\mathsf{hsame}(h, k)$. By the definition of $D_\wedge$, the admission premise gives both $\mathsf{InDom}(D_0, h)$ and $\mathsf{InDom}(D_1, h)$. Applying Definition 9195.2 to $\mathsf{DomainPolicy}(D_0)$ gives $\mathsf{InDom}(D_0, k)$, and applying Definition 9195.2 to $\mathsf{DomainPolicy}(D_1)$ gives $\mathsf{InDom}(D_1, k)$. The defining equivalence for $D_\wedge$ therefore gives $\mathsf{InDom}(D_\wedge, k)$.

For $D_\vee$, assume $\mathsf{InDom}(D_\vee, h)$ and $\mathsf{hsame}(h, k)$. By the definition of $D_\vee$, either $\mathsf{InDom}(D_0, h)$ or $\mathsf{InDom}(D_1, h)$. In the first case, Definition 9195.2 and $\mathsf{DomainPolicy}(D_0)$ give $\mathsf{InDom}(D_0, k)$, hence $\mathsf{InDom}(D_\vee, k)$. In the second case, Definition 9195.2 and $\mathsf{DomainPolicy}(D_1)$ give $\mathsf{InDom}(D_1, k)$, hence $\mathsf{InDom}(D_\vee, k)$. □

Corollary 9195.12 (Finite Boolean domain expressions preserve DomainPolicy)**.** *Every finite source-domain expression built from domain names satisfying* DomainPolicy *by repeated use of the conjunctive and disjunctive compound constructors of Definition 9195.10 also satisfies* DomainPolicy.

Proof. Induct on the construction of the finite expression. A leaf is one of the supplied domain names, so its policy is an assumption. At a conjunctive or disjunctive constructor, the induction hypotheses give DomainPolicy for the two immediate subexpressions, and Definition 9195.11 gives the policy for the compound expression. □

Chapter 9196

Package Token Policy, Uniqueness, and Reflection

9196.1 Package Token Policy and Reflection

This chapter replaces the function-like token notation $\mathsf{Tok}_\Pi(s)$ by the judgment $\mathsf{TokIntro}(\Pi, s, p)$ and records the proof interface governing introduced tokens.

Definition 9196.1 (Package token policy)**.** *For a probe bundle* Π*, a package token policy, written*

$$\mathsf{PackageTokenPolicy}(\Pi),$$

consists of the following clauses for the concrete signature-token instance.

1. ***Soundness.*** *If* $\mathsf{TokIntro}(\Pi, s, p)$*,* $\mathsf{TokIntro}(\Pi, t, q)$*, and* $\mathsf{hsame}(s, t)$*, then* $\mathsf{psame}(p, q)$*.*
2. ***Reflection.*** *If* $\mathsf{TokIntro}(\Pi, s, p)$*,* $\mathsf{TokIntro}(\Pi, t, q)$*, and* $\mathsf{psame}(p, q)$*, then* $\mathsf{hsame}(s, t)$*.*

```
Lean4 ✓  BEDC.FKernel.Package.PackageTokenPolicy iff fields
  ↪ variant  BEDC.FKernel.Package.PackageTokenPolicy iff soundness reflection
  ↪ variant  BEDC.FKernel.Package.PackageTokenPolicy fields
  ↪ variant  BEDC.FKernel.Package.PackageTokenPolicy classification fields on introduced
  ↪ variant  BEDC.FKernel.Package.PackageTokenPolicy soundness reflection pair
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy soundness field
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy reflection field
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy compares introduced by psame
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy classification equiv on introduced pair
  ↪ variant  BEDC.FKernel.Package.PackageTokenPolicy soundness reflection and closure
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy closure fields on introduced
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy psame trans on introduced
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy psame three step on introduced
  ↪ variant  BEDC.FKernel.Package.packageTokenPolicy reflect sound reflect chain
```

Package-sameness closure on introduced tokens is a consequence of soundness and reflection, recorded by the checked closure theorem above. Token existence is a separate premise, written

$$\forall s,\ \exists p,\ \mathsf{TokIntro}(\Pi, s, p).$$

The rule that visible tokens are compared by psame rather than by imported equality is a token-comparison principle for theorem statements, not a field of $\mathsf{PackageTokenPolicy}(\Pi)$.

Theorem 9196.2 (Package tokens are signature-facing)**.** *Assume* PackageTokenPolicy(Π)*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t),$$

then

$$\mathsf{psame}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Package.packageTokenPolicy signature facing`

Proof. This is the soundness field of the package token policy applied to the two token-introduction premises and the history-sameness premise. □

Theorem 9196.3 (Package-sameness is equivalent to signature-sameness)**.** *Assume* PackageTokenPolicy(Π)*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \qquad \mathsf{TokIntro}(\Pi, t, q),$$

then

$$\mathsf{psame}(p, q) \iff \mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.FKernel.Package.psame iff hsame`

↪ variant `BEDC.FKernel.Package.PackageTokenPolicy iff classifies introduced`
↪ variant `BEDC.FKernel.Package.PackageTokenPolicy sound complete`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy classifies introduced signatures`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy classification roundtrip`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy classification roundtrip on introduced`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy classification directions pair`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy psame five step on introduced`
↪ variant `BEDC.FKernel.Package.packages classify introduced signatures`
↪ variant `BEDC.FKernel.Package.packages classify introduced signatures directional`
↪ variant `BEDC.FKernel.Package.packagePolicy classifies signatures directions`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy psame three step reflect on introduced`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy psame hsame equivalence fields`
↪ variant `BEDC.FKernel.Package.packageTokenPolicy classification iff symm on introduced`

Proof. The implication from right to left is the soundness clause of PackageTokenPolicy(Π). The implication from left to right is the reflection clause. No external token equality is used. □

Corollary 9196.4 (Concrete package completeness is policy-grounded)**.** *The completeness direction of the concrete signature Globalize instance holds only under* PackageTokenPolicy(Π)*. In particular, it is not a theorem for arbitrary package-sameness relations.*

Lean4 ✓ `BEDC.FKernel.Package.concrete package completeness policy grounded`

↪ variant `BEDC.FKernel.Package.package completeness reflection field`

Proof. Completeness requires recovering $\mathsf{hsame}(s, t)$ from $\mathsf{psame}(p, q)$. This is exactly the reflection field of PackageTokenPolicy(Π). □

Remark 9196.5 (Proof-obligation boundary)**.** *The token policy is deliberately narrow. It applies to the concrete signature-token package instance. Other package constructions must provide their own soundness and reflection certificates before they may inherit the same completeness theorem.*

9196.2 Inductive Package-Token Policy for Signature Packages

section 9196.1 introduces $\mathsf{PackageTokenPolicy}(\Pi)$. This section removes one remaining ambiguity: package-sameness for the concrete signature-token instance should be generated by one rule only. The one-constructor relation supplies soundness directly; reflection is supplied either by the active package-token policy or by a token-uniqueness theorem that derives the policy field.

Definition 9196.6 (Concrete signature-token introduction)**.** *For the concrete signature-token instance, the judgment*

$$\mathsf{TokIntro}(\Pi, s, p)$$

means that token p is introduced as a visible token for answer history s under probe bundle Π.

Lean4 ✓ `BEDC.FKernel.Package.SignatureTokenIntro nonempty pkg`

Definition 9196.7 (Inductive package-sameness for signature tokens)**.** *For the concrete signature-token instance, $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$ is generated only by the following rule:*

$$\frac{\mathsf{TokIntro}(\Pi, s, p) \qquad \mathsf{TokIntro}(\Pi, t, q) \qquad \mathsf{hsame}(s, t)}{\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)}.$$

No other package-sameness constructors are admitted for this concrete instance.

Lean4 (definition) `BEDC.FKernel.Package.psame`
Lean4 (definition) `BEDC.FKernel.Package.ConcretePackageSamenessPolicy`
Lean4 ✓ `BEDC.FKernel.Package.PkgSig psame intro`
↪ variant `BEDC.FKernel.Package.ConcretePackageSamenessPolicy holds`
↪ variant `BEDC.FKernel.Package.concrete package sameness policy`
↪ variant `BEDC.FKernel.Package.psame left token witness`
↪ variant `BEDC.FKernel.Package.psame right token witness`
↪ variant `BEDC.FKernel.Package.psame token witness pair`
↪ variant `BEDC.FKernel.Package.psame token witnesses`
↪ variant `BEDC.FKernel.Package.psame source history pair`
↪ variant `BEDC.FKernel.Package.psame constructor witnesses with hsame`
↪ variant `BEDC.FKernel.Package.psame constructor symm witnesses`
↪ variant `BEDC.FKernel.Package.psame source hsame under tok unique`

Lemma 9196.8 (Package-sameness soundness)**.** *If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t),$$

then

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Package.signature package extensionality`
↪ variant `BEDC.FKernel.Package.signature package extensionality hsame`

Proof. This is the single constructor of $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$. □

Lemma 9196.9 (Package-sameness reflection)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q),$$

then

$$\mathsf{hsame}(s, t).$$

Lean4 ✓ BEDC.FKernel.Package.psame reflect

Proof. Apply the reflection field of $\mathsf{PackageTokenPolicy}(\Pi)$ to the two token-introduction premises and the package-sameness premise. □

Theorem 9196.10 (Concrete PackageTokenPolicy)**.** *The concrete signature-token instance satisfies* $\mathsf{PackageTokenPolicy}(\Pi)$ *from the introduced-token reflection direction*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

Lean4 ✓ BEDC.FKernel.Package.concrete package token policy

Proof. The soundness field is Lemma 9196.8. The reflection field is the displayed introduced-token reflection direction. □

Corollary 9196.11 (Exact signature package)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. For the concrete signature-token instance,* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$ *between introduced tokens is equivalent to internal sameness of the answer histories that introduced them.*

Lean4 ✓ BEDC.FKernel.Package.psame iff hsame under tok unique
↪ variant BEDC.FKernel.Package.psame hsame directions under tok unique
↪ variant BEDC.FKernel.Package.psame reflect under tok unique
↪ variant BEDC.FKernel.Package.exact signature package

Proof. The forward direction is package reflection derived from token-introduction uniqueness. The reverse direction is Lemma 9196.8. □

Remark 9196.12 (Implementation note)**.** *In a machine implementation, this chapter should be realized by defining package-sameness as an inductive relation with exactly one constructor and by supplying either an explicit reflection field or token-introduction uniqueness for the visible tokens.*

9196.3 Token-Introduction Uniqueness and Package Reflection

section 9196.2 makes concrete package-sameness a one-constructor relation. That is necessary but not sufficient for the strongest reflection theorem. If the same visible token can be introduced by two unrelated signatures, then package-sameness of tokens does not determine the signatures uniquely. This section isolates the missing requirement.

Definition 9196.13 (Token-introduction uniqueness)**.** *For a fixed generated probe bundle* Π*, define the token-introduction uniqueness obligation* $\mathsf{TokUnique}(\Pi)$ *as the following principle:*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

This is an internal uniqueness condition up to history-sameness. It is not host-level equality of tokens or signatures.

Lean4 ✓ BEDC.BaseReflection.TokUnique intro
↪ variant BEDC.BaseReflection.TokUnique iff tokenReplacement
Lean4 (definition) BEDC.FKernel.Package.TokUnique
Lean4 ✓ BEDC.FKernel.Package.TokUnique replacement
↪ variant BEDC.FKernel.Package.tokUnique replacement

Remark 9196.14 (Why this is needed)**.** *From a derivation of* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$*, inversion yields some generated signatures* s_0, t_0 *introducing* p, q*, together with* $\mathsf{hsame}(s_0, t_0)$*. If the displayed premises use* s, t *instead, one still needs to compare* s *with* s_0 *and* t *with* t_0*. This is exactly the role of* $\mathsf{TokUnique}(\Pi)$*.*

Lemma 9196.15 (Token replacement)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, s_0, p),$$

then

$$\mathsf{hsame}(s, s_0).$$

Lean4 ✓ `BEDC.BaseReflection.TokUnique replacement self`
↪ variant `BEDC.BaseReflection.TokUnique replacement congr`

Proof. This is Definition 9196.13 applied to the two token-introduction judgments for the same token p. □

Lemma 9196.16 (Symmetric token replacement)**.** *Assume* $\mathsf{TokUnique}(\Pi)$ *and that* hsame *is symmetric. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p),$$

then

$$\mathsf{hsame}(t, s).$$

Lean4 ✓ `BEDC.BaseReflection.TokUnique replacement symm`
↪ variant `BEDC.BaseReflection.TokUnique replacement trans`
↪ variant `BEDC.FKernel.Package.tokUnique replacement symm`

Proof. Token-introduction uniqueness gives $\mathsf{hsame}(s, t)$; symmetry of hsame gives $\mathsf{hsame}(t, s)$. □

Theorem 9196.17 (Concrete package reflection with token uniqueness)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q),$$

then

$$\mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.FKernel.Package.package reflection token unique`
↪ variant `BEDC.BaseReflection.PackageReflection token unique from inversion`
↪ variant `BEDC.FKernel.Package.psame two step reflect under tok unique`

Proof. Invert the derivation of $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$. Since the concrete signature-token package-sameness relation has only the constructor displayed in section 9196.2, the derivation supplies signatures s_0, t_0 such that

$$\mathsf{TokIntro}(\Pi, s_0, p), \quad \mathsf{TokIntro}(\Pi, t_0, q), \quad \mathsf{hsame}(s_0, t_0).$$

By Lemma 9196.15, the first token-introduction premise gives $\mathsf{hsame}(s, s_0)$, and the second gives $\mathsf{hsame}(t, t_0)$. Symmetry and transitivity of hsame now yield $\mathsf{hsame}(s, t)$. □

Corollary 9196.18 (Exact package-sameness under token uniqueness)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. For introduced tokens,*

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$$

if and only if their introducing signatures are internally history-same.

Lean4 ✓ BEDC.BaseReflection.exact package sameness
↪ variant BEDC.BaseReflection.PsameBase equivalence under tok unique
↪ variant BEDC.FKernel.Package.psame trans under tok unique
↪ variant BEDC.FKernel.Package.psame three step reflect under tok unique

Proof. The forward direction is Theorem 9196.17. The reverse direction is the single constructor of $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$. □

Corollary 9196.19 (Introduced-token closure under package token policy)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*. For package tokens introduced by* Π*, the following three clauses hold:*

$$\mathsf{TokIntro}(\Pi, s, p) \Rightarrow \mathsf{psame}(p, p),$$

$$\mathsf{TokIntro}(\Pi, s, p),\ \mathsf{TokIntro}(\Pi, t, q),\ \mathsf{psame}(p, q) \Rightarrow \mathsf{psame}(q, p),$$

and

$$\mathsf{TokIntro}(\Pi, a, p),\ \mathsf{TokIntro}(\Pi, b, q),\ \mathsf{TokIntro}(\Pi, c, r),\ \mathsf{psame}(p, q),\ \mathsf{psame}(q, r) \Rightarrow \mathsf{psame}(p, r).$$

Lean4 ✓ BEDC.FKernel.Package.packageTokenPolicy psame equivalence on introduced

Proof. For reflexivity, apply package-token soundness to the same token-introduction premise on both sides, together with reflexivity of hsame. For symmetry, reflect $\mathsf{psame}(p, q)$ across the two token-introduction premises to obtain $\mathsf{hsame}(s, t)$, reverse it by symmetry of hsame, and apply package-token soundness to the reversed pair. For transitivity, reflect $\mathsf{psame}(p, q)$ along the first two token-introduction premises and $\mathsf{psame}(q, r)$ along the last two; compose the resulting $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$ judgments, then apply package-token soundness to the first and last token-introduction premises. □

Theorem 9196.20 (PackageTokenPolicy from token uniqueness)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. For the concrete signature-token relation* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$*, the package token policy* $\mathsf{PackageTokenPolicy}(\Pi)$ *holds.*

Lean4 ✓ BEDC.FKernel.Package.packageTokenPolicy from tokUnique

Proof. The soundness field is the single constructor of $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$. The reflection field is Theorem 9196.17. □

9196.4 Token-Uniqueness Modes

This section makes $\mathsf{TokUnique}(\Pi)$ explicit as an active policy field and separates two implementation modes for the same reflection contract.

Definition 9196.21 (Policy mode)**.** *In policy mode,* $\mathsf{TokUnique}(\Pi)$ *is a field of the concrete package-token policy:*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

Concrete Globalize exactness theorems may use this field as an assumption.

Lean4 (definition) BEDC.BaseReflection.PolicyTokenMode
Lean4 ✓ BEDC.BaseReflection.PolicyTokenMode iff TokUnique
↪ variant BEDC.BaseReflection.PolicyTokenMode tokenReplacement
↪ variant BEDC.BaseReflection.PolicyTokenMode replacement pair

Definition 9196.22 (Canonical-constructor mode)**.** *In canonical-constructor mode, token introduction is implemented by a canonical token constructor, written schematically as*

$$\mathsf{TokCan}(\Pi, s, p).$$

The implementation must provide the no-confusion rule

$$\mathsf{TokCan}(\Pi, s, p), \quad \mathsf{TokCan}(\Pi, t, p) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

If TokIntro *is defined by* TokCan*, then* TokUnique(Π) *follows.*

Lean4 (definition) `BEDC.BaseReflection.CanonicalTokenMode`

Lemma 9196.23 (Canonical mode implies token uniqueness)**.** *Assume* $\mathsf{TokIntro}(\Pi, s, p)$ *is definitionally implemented by* $\mathsf{TokCan}(\Pi, s, p)$*, and assume the no-confusion rule of Definition 9196.22. Then* TokUnique(Π) *holds.*

Proof. Given $\mathsf{TokIntro}(\Pi, s, p)$ and $\mathsf{TokIntro}(\Pi, t, p)$, unfold the implementation to obtain $\mathsf{TokCan}(\Pi, s, p)$ and $\mathsf{TokCan}(\Pi, t, p)$. The no-confusion rule gives $\mathsf{hsame}(s, t)$, which is exactly TokUnique(Π). □

Lean4 ✓ `BEDC.BaseReflection.CanonicalTokenMode implies TokUnique`
- ↪ variant `BEDC.BaseReflection.canonical implies unique`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode to PolicyTokenMode`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode tokenReplacement`
- ↪ variant `BEDC.BaseReflection.canonical mode token unique`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode replacement and unique`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode witnesses TokUnique`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode replacement pair`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode witness pair`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode witness pair symm`
- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode unique and policy pair`

Principle 9196.24 (Active mode)**.** *Paper theorems express the assumption as* TokUnique(Π) *(policy mode). Canonical-constructor mode is an alternative carrier that derives* TokUnique(Π) *from a constructor no-confusion rule.*

- ↪ variant `BEDC.BaseReflection.CanonicalTokenMode implies TokUnique`

Corollary 9196.25 (Stable package reflection contract)**.** *The package reflection theorem may be used later only in one of two forms:*

1. *base-relation reflection under* TokUnique(Π)*;*
2. *base-relation reflection under a checked canonical-constructor theorem implying* TokUnique(Π)*.*

It may be used for $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ *only through the checked closure-reflection theorem of Proposition 9198.15; the exact Globalize export remains the base relation.*

Lean4 ✓ `BEDC.BaseReflection.StableReflectionContract from canonical`

Proof. The first form is Theorem 9198.5. The second form follows from Lemma 9196.23. The final exclusion is Corollary 9198.6. □

Lean4 ✓ `BEDC.BaseReflection.PackageReflection base from canonical`
- ↪ variant `BEDC.BaseReflection.StableReflectionContract`

Theorem 9196.26 (Concrete token reflection duality)**.** *Assume canonical-constructor mode for* Π*: for every answer history* u *and package token* r*, the term-level predicate* $\mathsf{TokCan}(\Pi, u, r)$ *implements* $\mathsf{TokIntro}(\Pi, u, r)$*, and the no-confusion rule of Definition 9196.22 holds. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q),$$

then

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q) \quad \Longleftrightarrow \quad \mathsf{hsame}(s, t).$$

The same equivalence holds with $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ *in place of* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$ *under the identification in Definition 9198.1.*

Proof. By Definition 9196.23, the concrete canonical implementation and its no-confusion rule give $\mathsf{TokUnique}(\Pi)$. For the forward implication, apply Definition 9196.17 to this token-uniqueness witness, the two displayed token-introduction premises, and $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$. For the reverse implication, apply Definition 9196.8 to the same token-introduction premises and the assumed $\mathsf{hsame}(s, t)$. The formulation with $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ follows from its identification with $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$ in Definition 9198.1. No use is made of $\mathsf{PackageTokenPolicy}(\Pi)$. □

Theorem 9196.27 (Signature append residual exactness under known token component)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*,* $\mathsf{BundleAskPolicy}(\Theta, D)$*,* $\mathsf{PackagePolicy}(\mathsf{BAppend}(\Pi, \Theta))$*,* $\mathsf{PackageTokenPolicy}(\mathsf{BAppend}(\Pi, \Theta$ *and* $\mathsf{PackageTokenPolicy}(\Pi)$*. Let* h *and* k *be source histories with*

$$\mathsf{InDom}(D, h), \qquad \mathsf{InDom}(D, k).$$

Suppose the appended interface has generated signatures and introduced package tokens

$$\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, a, \Lambda_h), \qquad \mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), k, b, \Lambda_k),$$

$$\mathsf{TokIntro}(\mathsf{BAppend}(\Pi, \Theta), a, P_h), \qquad \mathsf{TokIntro}(\mathsf{BAppend}(\Pi, \Theta), b, P_k), \qquad \mathsf{psame}_{\mathsf{BAppend}(\Pi,\Theta)}(P_h, P_k).$$

Suppose also that the known Π *component has generated signatures and introduced package tokens*

$$\mathsf{Sig}(\Pi, h, s, \Delta_h), \qquad \mathsf{Sig}(\Pi, k, t, \Delta_k),$$

$$\mathsf{TokIntro}(\Pi, s, p_h), \qquad \mathsf{TokIntro}(\Pi, t, p_k), \qquad \mathsf{psame}_{\Pi}(p_h, p_k).$$

Then

$$\mathsf{sameSig}_{\Theta}(h, k).$$

Proof. The public package policy supplies the appended package interface in which the displayed package tokens live. The comparison facts used below are the introduced-token reflection clauses of $\mathsf{PackageTokenPolicy}$ from Definition 9196.1.

By Definition 9196.3 for $\mathsf{BAppend}(\Pi, \Theta)$, the appended token-introduction premises and the appended package-sameness premise give $\mathsf{hsame}(a, b)$. The two appended signature judgments and this result sameness form $\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$ by Definition 6.75.

Applying Definition 9196.3 again for Π to the component token-introduction premises and the Π package-sameness premise gives $\mathsf{hsame}(s, t)$, hence $\mathsf{sameSig}_{\Pi}(h, k)$.

The first residual equivalence in Definition 6.57 applies with the bundle-local policy on Π, the two domain witnesses, and the known Π signature-sameness. Its forward direction sends $\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$ to $\mathsf{sameSig}_{\Theta}(h, k)$. □

Theorem 9196.28 (Signature append residual exactness under known right token component)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*,* $\mathsf{BundleAskPolicy}(\Theta, D)$*,* $\mathsf{PackagePolicy}(\mathsf{BAppend}(\Pi, \Theta))$*,* $\mathsf{PackageTokenPolicy}(\mathsf{BApp}$ *and* $\mathsf{PackageTokenPolicy}(\Theta)$*. Let* h *and* k *be source histories with*

$$\mathsf{InDom}(D, h), \qquad \mathsf{InDom}(D, k).$$

Suppose the appended interface has generated signatures and introduced package tokens

$$\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, a, \Lambda_h), \qquad \mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), k, b, \Lambda_k),$$

$$\mathsf{TokIntro}(\mathsf{BAppend}(\Pi, \Theta), a, P_h), \qquad \mathsf{TokIntro}(\mathsf{BAppend}(\Pi, \Theta), b, P_k), \qquad \mathsf{psame}_{\mathsf{BAppend}(\Pi,\Theta)}(P_h, P_k).$$

Suppose also that the known Θ *component has generated signatures and introduced package tokens*

$$\mathsf{Sig}(\Theta, h, u, \Gamma_h), \qquad \mathsf{Sig}(\Theta, k, v, \Gamma_k),$$

$$\mathsf{TokIntro}(\Theta, u, q_h), \qquad \mathsf{TokIntro}(\Theta, v, q_k), \qquad \mathsf{psame}_{\Theta}(q_h, q_k).$$

Then

$$\mathsf{sameSig}_{\Pi}(h, k).$$

Proof. The displayed token-introduction judgments place the appended tokens in the package-token interface for $\mathsf{BAppend}(\Pi, \Theta)$ and the component tokens in the package-token interface for Θ. The comparison step uses the introduced-token reflection direction of $\mathsf{PackageTokenPolicy}$ from Definition 9196.1.

By Definition 9196.3 for $\mathsf{BAppend}(\Pi, \Theta)$, the appended token-introduction premises and the appended package-sameness premise give $\mathsf{hsame}(a, b)$. Together with the two appended signature judgments, this is exactly a witness for $\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$ by Definition 6.75.

Applying Definition 9196.3 for Θ to the component token-introduction premises and the Θ package-sameness premise gives $\mathsf{hsame}(u, v)$. The two Θ signature judgments therefore witness $\mathsf{sameSig}_{\Theta}(h, k)$.

The second residual equivalence in Definition 6.57 applies with the bundle-local policy on Θ, the two domain witnesses, and the known Θ signature-sameness. Its forward direction sends $\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$ to $\mathsf{sameSig}_{\Pi}(h, k)$. □

Theorem 9196.29 (PackageTokenPolicy reflects token uniqueness)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*. For the concrete signature-token relation* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$*, token-introduction uniqueness holds:*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

Equivalently, $\mathsf{TokUnique}(\Pi)$ *holds for the visible tokens introduced by* Π*.*

Proof. Let $\mathsf{TokIntro}(\Pi, s, p)$ and $\mathsf{TokIntro}(\Pi, t, p)$ be given. Since hsame is reflexive, the first token-introduction premise can be compared with itself:

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{hsame}(s, s).$$

The soundness clause of $\mathsf{PackageTokenPolicy}(\Pi)$ from Definition 9196.1, applied in the concrete signature-token instance of Definition 9196.7, gives

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, p).$$

Now apply the reflection clause of $\mathsf{PackageTokenPolicy}(\Pi)$ to the two original token-introduction premises

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p)$$

and to this package-sameness fact $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, p)$. The conclusion is $\mathsf{hsame}(s, t)$, which is exactly the obligation named $\mathsf{TokUnique}(\Pi)$ in Definition 9196.13. □

Corollary 9196.30 (Concrete PackageTokenPolicy and token uniqueness are equivalent)**.** *For the concrete signature-token relation* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$*,*

$$\mathsf{PackageTokenPolicy}(\Pi) \iff \mathsf{TokUnique}(\Pi).$$

Proof. The forward implication is Definition 9196.29. The reverse implication is Definition 9196.20: token-introduction uniqueness supplies the reflection field, and the single constructor of the concrete signature-token relation supplies the soundness field. □

Proposition 9196.31 (Empty introduction separates package-token policy from token existence)**.** *Assume that no package token is introduced by* Π*:*

$$\forall s, p,\ \neg\, \mathsf{TokIntro}(\Pi, s, p).$$

Then $\mathsf{PackageTokenPolicy}(\Pi)$ *holds, but package-wide token totality fails:*

$$\neg\Big(\forall s,\ \exists p,\ \mathsf{TokIntro}(\Pi, s, p)\Big).$$

Thus the soundness and reflection clauses of $\mathsf{PackageTokenPolicy}(\Pi)$ *have no token-existence consequence.*

Proof. Let E denote the displayed no-introduction hypothesis. We first verify the two fields in Definition 9196.1. For soundness, suppose

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t).$$

The first token-introduction premise contradicts E at s, p, so the required conclusion $\mathsf{psame}(p, q)$ follows by contradiction elimination. For reflection, suppose

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}(p, q).$$

Again the first token-introduction premise contradicts E at s, p, so the required conclusion $\mathsf{hsame}(s, t)$ follows by contradiction elimination. These are exactly the soundness and reflection clauses of $\mathsf{PackageTokenPolicy}(\Pi)$.

It remains to show that token totality fails. If

$$\forall s,\ \exists p,\ \mathsf{TokIntro}(\Pi, s, p)$$

held, then instantiating it at the empty history e would give a token p with $\mathsf{TokIntro}(\Pi, \mathsf{e}, p)$. This contradicts E at e, p. Hence

$$\neg\Big(\forall s,\ \exists p,\ \mathsf{TokIntro}(\Pi, s, p)\Big).$$

The policy fields therefore classify pairs of already introduced tokens only; they do not manufacture an introduced token for any history. □

Chapter 9197

Gap Policy

9197.1 Consolidated Signature Gap Policy

This chapter restates the concrete signature gap and records the exact assumptions needed for coverage, separation, soundness, and completeness.

Definition 9197.1 (Policy-supported signature gap)**.** *For a probe bundle* Π*, source domain* D*, package* p*, and history* h*, define*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$$

when there exist an answer history s *and an evidence ledger* Δ *such that*

$$\mathsf{InDom}(D, h), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

Lean4 (definition) `BEDC.FKernel.Gap.InGapSig`
Lean4 (definition) `BEDC.FKernel.Gap.PolicySupportedSignatureGap`
Lean4 ✓ `BEDC.FKernel.Gap.inGapSig witness pair`
↪ variant `BEDC.FKernel.Gap.PolicySupportedSignatureGap iff InGapSig`
↪ variant `BEDC.FKernel.Gap.PolicySupportedSignatureGap domain witness`
↪ variant `BEDC.FKernel.Gap.PolicySupportedSignatureGap witnesses`
↪ variant `BEDC.FKernel.Gap.PolicySupportedSignatureGap domain transport with signature`
↪ variant `BEDC.FKernel.Gap.inGapSig witness pair iff`
↪ variant `BEDC.FKernel.Gap.policy supported signature gap iff witness pair`
↪ variant `BEDC.FKernel.Gap.inGapSig intro from witness pair`
↪ variant `BEDC.FKernel.Gap.inGapSig domain token witness`

Theorem 9197.2 (Policy-supported gap coverage)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and*

$$\forall s,\ \exists p,\ \mathsf{TokIntro}(\Pi, s, p).$$

If

$$\mathsf{InDom}(D, h),$$

then there exists a token p *such that*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Lean4 ✓ `BEDC.FKernel.Gap.policy gap coverage`
↪ variant `BEDC.FKernel.Gap.gap coverage policy`

Proof. From $\mathsf{AskPolicy}(\Pi, D)$ and $\mathsf{InDom}(D, h)$, obtain a generated signature event $\mathsf{Sig}(\Pi, h, s, \Delta)$. From the explicit token-existence premise, obtain p with $\mathsf{TokIntro}(\Pi, s, p)$. These data witness $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. □

Theorem 9197.3 (Policy-supported gap separation)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and* $\mathsf{PackageTokenPolicy}(\Pi)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then

$$\mathsf{psame}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Gap.policy gap separation`

↪ variant `BEDC.FKernel.Gap.policy gap separation witnesses`
↪ variant `BEDC.FKernel.Gap.policy gap separation with domain witnesses`
↪ variant `BEDC.FKernel.Gap.policy gap separation signature hsame`
↪ variant `BEDC.FKernel.Gap.policy gap separation pair`
↪ variant `BEDC.FKernel.Gap.gap separation policy`
↪ variant `BEDC.FKernel.Gap.gapPolicy separation field`
↪ variant `BEDC.FKernel.Gap.gapPolicy separation packed`
↪ variant `BEDC.FKernel.Gap.policy gap separation signature token witnesses`

Proof. Unpack both gap memberships. We obtain the admitted-source witness $\mathsf{InDom}(D, h)$, signatures s, t, and evidence ledgers Δ, Θ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p),$$

$$\mathsf{Sig}(\Pi, h, t, \Theta), \qquad \mathsf{TokIntro}(\Pi, t, q).$$

Theorem 6.67, applied to $\mathsf{AskPolicy}(\Pi, D)$, the admitted-source witness, and the two generated signatures, gives $\mathsf{hsame}(s, t)$. Package soundness in $\mathsf{PackageTokenPolicy}(\Pi)$ then gives $\mathsf{psame}(p, q)$. □

Theorem 9197.4 (Token-level Globalize exactness)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*,* $\mathsf{TokIntro}(\Pi, s, p)$*, and* $\mathsf{TokIntro}(\Pi, t, q)$*. Then* $\mathsf{psame}(p, q) \iff \mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.policy globalize exact`

Proof. This is Theorem 9196.3. □

Corollary 9197.5 (Active reading)**.** *The concrete signature Globalize instance is presented only as a policy-supported construction. Its exactness depends on* $\mathsf{DomainPolicy}(D)$*,* $\mathsf{AskPolicy}(\Pi, D)$*, and* $\mathsf{PackageTokenPolicy}(\Pi)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.active reading policy dependencies`

↪ variant `BEDC.FKernel.Gap.active reading policy dependency fields`

Definition 9197.6 (Classifier-preserving pullback ledger)**.** *Let* D' *and* D *be source-domain names and let* τ *be a BEDC history operation written* $\tau : D' \to D$*. A classifier-preserving pullback ledger for* τ *over* Π *consists of a pulled-back membership predicate* $\mathsf{InGap}^{\mathrm{sig}^{\tau}}(\Pi, D', D, p, h')$ *with the following clauses:*

1. *source preservation: for every* h'*,*

$$\mathsf{InDom}(D', h') \Longrightarrow \mathsf{InDom}(D, \tau(h'));$$

2. ledger classifier preservation: for every visible package token p and source h',

$$\mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h') \iff \mathsf{InDom}(D', h') \text{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau(h')).$$

The classifier in the second clause is the package-gap ledger classifier in the sense of Definition 9.6: the visible token p remains a token of the original bundle Π, and the pullback changes the source coordinate.

Theorem 9197.7 (Package-token pullback separation)**.** *Assume* $\mathsf{PackagePolicy}(\Pi)$ *and* $\mathsf{GapPolicy}(\Pi, D)$*. Let $\tau : D' \to D$ be equipped with a classifier-preserving pullback ledger over Π. If two pulled-back gap ledgers meet on the same D'-source,*

$$\mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h') \quad \text{and} \quad \mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, q, h'),$$

then the visible package tokens are package-same in the original package interface of Π:

$$\mathsf{psame}(p, q).$$

Proof. Unpack the two pulled-back memberships by the ledger classifier preservation clause of Definition 9197.6. They give original ledger memberships

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau(h')) \quad \text{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, \tau(h')).$$

These two memberships lie over the same target-domain source $\tau(h')$. Applying Definition 8.5 to $\mathsf{GapPolicy}(\Pi, D)$ and the displayed memberships gives $\mathsf{psame}(p, q)$. The $\mathsf{PackagePolicy}(\Pi)$ premise fixes the package-token interface whose visible tokens are compared; the pullback introduces no separate comparison relation. □

Theorem 9197.8 (Pulled-back gap policy preserves coverage and separation)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. Let $\tau : D' \to D$ be equipped with the source preservation and ledger classifier preservation clauses of Definition 9197.6. Then the pulled-back membership predicate* $\mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h')$ *satisfies the coverage and separation obligations over D':*

1. for every source history h',

$$\mathsf{InDom}(D', h') \Longrightarrow \exists p,\ \mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h');$$

2. for every source history h' and visible package tokens p, q,

$$\mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h') \Longrightarrow \mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, q, h') \Longrightarrow \mathsf{psame}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Gap.PulledBackGapPolicy coverage separation`

Proof. For coverage, fix h' and assume $\mathsf{InDom}(D', h')$. The source preservation clause of Definition 9197.6 gives $\mathsf{InDom}(D, \tau(h'))$. The coverage field of $\mathsf{GapPolicy}(\Pi, D)$ in Definition 8.2 supplies a visible package token p with

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau(h')).$$

Using the reverse direction of the ledger classifier preservation clause with $\mathsf{InDom}(D', h')$ and this original membership gives

$$\mathsf{InGap}^{\mathrm{sig}^\tau}(\Pi, D', D, p, h'),$$

which proves pulled-back coverage.

For separation, fix h', p, and q, and assume

$$\mathsf{InGap}^{\mathrm{sig}^{\tau}}(\Pi, D', D, p, h') \quad \text{and} \quad \mathsf{InGap}^{\mathrm{sig}^{\tau}}(\Pi, D', D, q, h').$$

The forward direction of the ledger classifier preservation clause gives original gap memberships

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau(h')) \quad \text{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, \tau(h')).$$

Applying Definition 8.5 to $\mathsf{GapPolicy}(\Pi, D)$ and these two memberships gives $\mathsf{psame}(p, q)$. □

Theorem 9197.9 (Composite pullback gap policy)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. Let* $\tau_1 : D' \to D$ *and* $\tau_2 : D'' \to D'$ *be BEDC history operations, each equipped with a classifier-preserving pullback ledger over* Π *in the sense of Definition 9197.6. Let the composite pulled-back membership predicate be the pullback of the* τ_1*-ledger along* τ_2*:*

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'') \iff \mathsf{InDom}(D'', h'') \text{ and } \mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(h'')).$$

Then the operation $\tau_1 \circ \tau_2 : D'' \to D$ *carries a classifier-preserving pullback ledger over* Π*. Its composite membership predicate satisfies the coverage and separation obligations over* D''*:*

1. for every source history h''*,*

$$\mathsf{InDom}(D'', h'') \Longrightarrow \exists p,\ \mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'');$$

2. for every source history h'' *and visible package tokens* p, q*,*

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'') \Longrightarrow \mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, q, h'') \Longrightarrow \mathsf{psame}(p, q).$$

Proof. First verify the two clauses of Definition 9197.6 for the composite operation. If $\mathsf{InDom}(D'', h'')$, the source-preservation clause for the τ_2-ledger gives $\mathsf{InDom}(D', \tau_2(h''))$. Applying the source-preservation clause for the τ_1-ledger gives

$$\mathsf{InDom}(D, \tau_1(\tau_2(h''))).$$

Thus $\tau_1 \circ \tau_2$ preserves admitted sources from D'' to D.

For classifier preservation, fix p and h''. Suppose first that

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'')$$

holds. By the displayed definition, $\mathsf{InDom}(D'', h'')$ and

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(h''))$$

hold. The classifier-preservation clause for the τ_1-ledger gives

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau_1(\tau_2(h''))).$$

Hence the forward direction of the composite classifier clause holds.

Conversely, assume $\mathsf{InDom}(D'', h'')$ and

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau_1(\tau_2(h''))).$$

Source preservation for the τ_2-ledger gives $\mathsf{InDom}(D', \tau_2(h''))$. The reverse direction of the classifier-preservation clause for the τ_1-ledger then gives

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(h'')),$$

and the displayed definition gives

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'').$$

Therefore the composite predicate satisfies

$$\mathsf{InGap}^{\mathrm{sig}^{\tau_1 \circ \tau_2}}(\Pi, D'', D, p, h'') \iff \mathsf{InDom}(D'', h'') \text{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau_1(\tau_2(h''))),$$

which is the classifier-preservation clause for $\tau_1 \circ \tau_2$.

The composite operation is therefore equipped with the source-preservation and classifier-preservation clauses required by Definition 9197.6. Applying Definition 9197.8 to $\mathsf{GapPolicy}(\Pi, D)$ and this composite ledger gives the displayed coverage and separation obligations on D''. □

Theorem 9197.10 (Identity pullback gap policy unit law)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. Let* $\mathrm{id}_D : D \to D$ *be the identity BEDC history operation,* $\mathrm{id}_D(h) = h$*, and define its pulled-back package-gap membership predicate by*

$$\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h) \iff \mathsf{InDom}(D, h) \textit{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Then id_D *carries a classifier-preserving pullback ledger over* Π *in the sense of Definition 9197.6. Moreover, on every admitted history* h*,*

$$\mathsf{InDom}(D, h) \implies \big(\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h) \iff \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)\big),$$

and the identity pulled-back membership predicate satisfies the coverage and separation obligations over D*:*

1. for every source history h*,*

$$\mathsf{InDom}(D, h) \implies \exists p,\ \mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h);$$

2. for every source history h *and visible package tokens* p, q*,*

$$\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h) \implies \mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, q, h) \implies \mathsf{psame}(p, q).$$

Proof. First verify the two clauses of Definition 9197.6. Source preservation is immediate: from $\mathsf{InDom}(D, h)$ and $\mathrm{id}_D(h) = h$ we obtain

$$\mathsf{InDom}(D, \mathrm{id}_D(h)).$$

For classifier preservation, fix p and h. The displayed definition gives

$$\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h) \iff \mathsf{InDom}(D, h) \text{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Since $\mathrm{id}_D(h) = h$, this is exactly

$$\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h) \iff \mathsf{InDom}(D, h) \text{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \mathrm{id}_D(h)),$$

which is the classifier-preservation clause for the identity operation.

Now assume $\mathsf{InDom}(D, h)$. If $\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h)$ holds, the defining equivalence gives $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. Conversely, if $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ holds, pairing it with the admitted-source witness $\mathsf{InDom}(D, h)$ and applying the reverse direction of the displayed definition gives $\mathsf{InGap}^{\mathrm{sig}^{\mathrm{id}_D}}(\Pi, D, D, p, h)$. This proves the exact admitted-history equivalence.

The identity operation has therefore been equipped with the source-preservation and classifier-preservation clauses required by Definition 9197.6. Applying Definition 9197.8 to $\mathsf{GapPolicy}(\Pi, D)$ and this identity ledger gives the displayed coverage and separation obligations over D. □

Theorem 9197.11 (Pullback gap-policy composition associativity)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. Let* $\tau_1 : D' \to D$*,* $\tau_2 : D'' \to D'$*, and* $\tau_3 : D''' \to D''$ *be BEDC history operations, each equipped with a classifier-preserving pullback ledger over* Π *in the sense of Definition 9197.6. For a visible package token* p *and a source* h'''*, define the left-associated composite pulled-back predicate by*

$$L_{\tau_1,\tau_2,\tau_3}(p, h''') \iff \mathsf{InDom}(D''', h''') \text{ and } \big(\mathsf{InDom}(D'', \tau_3(h''')) \text{ and } \mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(\tau_3(h'''))))\big),$$

which is the pullback along τ_3 *of the* $\tau_1 \circ \tau_2$ *composite predicate from Definition 9197.9. Define the right-associated composite pulled-back predicate by*

$$R_{\tau_1,\tau_2,\tau_3}(p, h''') \iff \mathsf{InDom}(D''', h''') \text{ and } \mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(\tau_3(h'''))),$$

which is the pullback of the τ_1*-ledger along the composed source operation* $\tau_2 \circ \tau_3 : D''' \to D'$*. Then the two associated composite gap predicates are pointwise equivalent:*

$$L_{\tau_1,\tau_2,\tau_3}(p, h''') \iff R_{\tau_1,\tau_2,\tau_3}(p, h''').$$

Consequently the two composite presentations determine the same coverage and separation obligations over D'''*:*

$$\big(\mathsf{InDom}(D''', h''') \implies \exists p,\ L_{\tau_1,\tau_2,\tau_3}(p, h''')\big) \iff \big(\mathsf{InDom}(D''', h''') \implies \exists p,\ R_{\tau_1,\tau_2,\tau_3}(p, h''')\big),$$

and

$$\big(L_{\tau_1,\tau_2,\tau_3}(p, h''') \implies L_{\tau_1,\tau_2,\tau_3}(q, h''') \implies \mathsf{psame}(p, q)\big) \iff \big(R_{\tau_1,\tau_2,\tau_3}(p, h''') \implies R_{\tau_1,\tau_2,\tau_3}(q, h''') \implies \mathsf{psame}(p, q)\big).$$

Proof. Fix p and h'''. The defining equivalence for the left-associated predicate gives

$$L_{\tau_1,\tau_2,\tau_3}(p, h''') \iff \mathsf{InDom}(D''', h''') \text{ and } \big(\mathsf{InDom}(D'', \tau_3(h''')) \text{ and } \mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(\tau_3(h'''))))\big).$$

The source-preservation clause for the τ_3-ledger in Definition 9197.6 gives

$$\mathsf{InDom}(D''', h''') \implies \mathsf{InDom}(D'', \tau_3(h''')).$$

Hence the middle admitted-source conjunct is forced whenever the outer admitted-source conjunct holds, and the displayed formula is equivalent to

$$\mathsf{InDom}(D''', h''') \text{ and } \mathsf{InGap}^{\mathrm{sig}^{\tau_1}}(\Pi, D', D, p, \tau_2(\tau_3(h'''))).$$

By the defining equivalence for the right-associated predicate, this is exactly $R_{\tau_1,\tau_2,\tau_3}(p, h''')$. This proves the pointwise equivalence.

Equivalently, unfolding the τ_1-ledger classifier clause of Definition 9197.6 shows that both sides reduce to the same original target classifier. For the right-associated side one obtains

$$R_{\tau_1,\tau_2,\tau_3}(p, h''') \iff \mathsf{InDom}(D''', h''') \text{ and } \big(\mathsf{InDom}(D', \tau_2(\tau_3(h'''))) \text{ and } \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, \tau_1(\tau_2(\tau_3(h'''))))\big).$$

For the left-associated side the same unfolding gives the same target classifier with the extra conjunct $\mathsf{InDom}(D'', \tau_3(h'''))$. The source-preservation clauses for τ_3 and τ_2 make the D'' and D' admitted-source conjuncts consequences of $\mathsf{InDom}(D''', h''')$. Thus both associated routes read the original gap ledger at the identical source coordinate $\tau_1(\tau_2(\tau_3(h'''))$.

For coverage, apply the pointwise equivalence inside the existential over p: a witness for the left-associated predicate is the same token read through the right-associated predicate, and conversely. Placing the common implication premise $\mathsf{InDom}(D''', h''')$ in front of both statements preserves the equivalence of the two coverage obligations.

For separation, assume the left-associated implication. Given $R_{\tau_1,\tau_2,\tau_3}(p, h''')$ and $R_{\tau_1,\tau_2,\tau_3}(q, h''')$, convert both hypotheses to the corresponding left-associated memberships by the pointwise equivalence and obtain $\mathsf{psame}(p, q)$. The converse implication is identical with left and right interchanged. Therefore the two composite predicates determine the same separation obligation over D'''. □

Theorem 9197.12 (Gap transport has a package-same representative)**.** *Assume* $\mathsf{DomainPolicy}(D)$*,* $\mathsf{BundleAskPolicy}(\Pi, D)$*,* $\mathsf{PackageTokenPolicy}(\Pi)$*, and token totality for the bundle:*

$$\forall u,\ \exists r,\ \mathsf{TokIntro}(\Pi, u, r).$$

If

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{hsame}(h, k),$$

then there exists a package token q such that

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \quad \textit{and} \quad \mathsf{psame}(p, q).$$

Proof. Unpack the gap membership by Definition 9197.1. It supplies an admitted-source judgment, an answer history s, and an evidence ledger Δ such that

$$\mathsf{InDom}(D, h), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

By Definition 9195.2, the premises $\mathsf{DomainPolicy}(D)$, $\mathsf{InDom}(D, h)$, and $\mathsf{hsame}(h, k)$ give

$$\mathsf{InDom}(D, k).$$

Apply bundle-local signature existence from Definition 6.62 to $\mathsf{BundleAskPolicy}(\Pi, D)$ and this admitted source. This gives an answer history t and an evidence ledger Θ with

$$\mathsf{Sig}(\Pi, k, t, \Theta).$$

Token totality supplies a package token q such that

$$\mathsf{TokIntro}(\Pi, t, q).$$

The three judgments $\mathsf{InDom}(D, k)$, $\mathsf{Sig}(\Pi, k, t, \Theta)$, and $\mathsf{TokIntro}(\Pi, t, q)$ witness

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k).$$

It remains to compare p and q. Bundle-local source respect, Definition 6.64, applied to $\mathsf{BundleAskPolicy}(\Pi, D)$, the premise $\mathsf{hsame}(h, k)$, and the two generated signatures gives

$$\mathsf{hsame}(s, t).$$

The package-token soundness direction in Definition 9196.2, applied to $\mathsf{PackageTokenPolicy}(\Pi)$, the two token-introduction judgments, and $\mathsf{hsame}(s, t)$, yields

$$\mathsf{psame}(p, q).$$

Thus the displayed q is both a gap representative over k and package-same to the original token p. □

Theorem 9197.13 (Second-layer separation obstruction for composite gaps)**.** *The second-layer separation premise in Definition 8.9 is not implied by first-layer coverage, first-layer separation, and second-layer coverage. More concretely, there is a finite composite-gap setup with a source x, an intermediate token y, and final tokens* $\mathsf{b0}, \mathsf{b1}$ *such that:*

1. *the first gap covers x and separates all first-layer witnesses over x;*
2. *the second gap covers y;*
3. *x lies in the composite gaps of both* $\mathsf{b0}$ *and* $\mathsf{b1}$*;*
4. *the final comparison is* $\mathsf{finalSame} = \mathsf{msame}$*, and* $\mathsf{finalSame}(\mathsf{b0}, \mathsf{b1})$ *fails.*

Thus second-layer coverage alone cannot yield composite separation.

Proof. Take the finite source carrier $\{x\}$, the finite intermediate carrier $\{y\}$, and the final carrier given by the two BEDC raw marks $\{\mathsf{b0}, \mathsf{b1}\}$. Let the admitted-source and intermediate-admissibility predicates hold on their unique elements:

$$\mathsf{SourceOk}(x), \qquad \mathsf{InterOk}(y).$$

Define the first-layer gap by the single membership

$$\mathsf{firstGap}(y, x),$$

and no other first-layer memberships. Define the intermediate sameness relation on the one-point intermediate carrier by

$$\mathsf{interSame}(y, y).$$

Then first-layer coverage holds because y covers x. First-layer separation also holds: if u and v are first-layer witnesses over x, the displayed definition of $\mathsf{firstGap}$ forces $u = y$ and $v = y$, and hence $\mathsf{interSame}(u, v)$ is exactly $\mathsf{interSame}(y, y)$.

Define the second-layer gap by

$$\mathsf{secondGap}(\mathsf{b0}, y) \qquad \text{and} \qquad \mathsf{secondGap}(\mathsf{b1}, y),$$

and no other second-layer memberships. Second-layer coverage holds for the only intermediate token y, for instance by the witness $\mathsf{b0}$.

Let the final comparison relation be raw mark-sameness:

$$\mathsf{finalSame}(z, w) \iff \mathsf{msame}(z, w).$$

By the composite ledger definition in Definition 8.7, the witness y gives

$$\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, \mathsf{b0}, x)$$

from $\mathsf{firstGap}(y, x)$ and $\mathsf{secondGap}(\mathsf{b0}, y)$, and the same witness gives

$$\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, \mathsf{b1}, x)$$

from $\mathsf{firstGap}(y, x)$ and $\mathsf{secondGap}(\mathsf{b1}, y)$.

However, raw mark no-confusion in Definition 2.9 gives

$$\neg\, \mathsf{msame}(\mathsf{b0}, \mathsf{b1}).$$

Since $\mathsf{finalSame}$ is msame, this is

$$\neg\, \mathsf{finalSame}(\mathsf{b0}, \mathsf{b1}).$$

Therefore the same source x belongs to composite gaps of two final tokens that are not final-same. The only failed hypothesis from the layered separation theorem is the second-layer separation condition, because that condition would have to turn

$$\mathsf{interSame}(y, y), \qquad \mathsf{secondGap}(\mathsf{b0}, y), \qquad \mathsf{secondGap}(\mathsf{b1}, y)$$

into $\mathsf{finalSame}(\mathsf{b0}, \mathsf{b1})$, contradicting raw mark no-confusion. Hence first-layer coverage, first-layer separation, and second-layer coverage do not suffice for composite separation. □

Chapter 9198

Package-Sameness Design and Base Reflection

9198.1 Concrete Package-Sameness Design Contract

The reflection theorem is correct for the one-constructor concrete package-sameness relation. It must not be silently transferred to an equivalence closure. This chapter fixes the active design.

Definition 9198.1 (Base concrete package-sameness)**.** *For a fixed generated probe bundle* Π*, the active concrete signature-token package-sameness relation is written*

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

It is generated by exactly one constructor:

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t) \quad \Longrightarrow \quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

The active notation $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$ *is identified with this base relation:*

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi} := \mathsf{psame}^{\mathrm{base}}_{\Pi}.$$

Lean4 (definition) `BEDC.BaseReflection.PsameSig`

Lean4 ✓ `BEDC.BaseReflection.PsameSig iff PsameBase`

↪ *variant* `BEDC.BaseReflection.PsameSig constructor`

↪ *variant* `BEDC.BaseReflection.PsameSig constructor inversion`

↪ *variant* `BEDC.BaseReflection.PsameSig iff constructor witnesses`

↪ *variant* `BEDC.BaseReflection.PsameBase iff constructor witnesses`

↪ *variant* `BEDC.BaseReflection.PsameBase symm under equiv`

↪ *variant* `BEDC.BaseReflection.PsameBase refl from token`

↪ *variant* `BEDC.BaseReflection.PsameBase swapped constructor witnesses`

Principle 9198.2 (No implicit equivalence closure)**.** *No theorem in the concrete signature Globalize instance may replace* $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ *by an equivalence closure without explicitly adding a closure-reflection theorem.*

Lean4 ✓ `BEDC.BaseReflection.no implicit closure exports base`

Definition 9198.3 (Optional equivalence closure)**.** *If a future package-algebra layer needs an equivalence relation, it must introduce a separate relation*

$$\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p,q) := \mathsf{EqCl}(\mathsf{psame}^{\mathrm{base}}_{\Pi})(p,q).$$

Lean4 ✓ `BEDC.BaseReflection.PsameEqClosure base inclusion`
↪ variant `BEDC.BaseReflection.PsameEqClosure base symm inclusion`
↪ variant `BEDC.BaseReflection.PsameEqClosure refl`
↪ variant `BEDC.BaseReflection.PsameEqClosure equivalence`
↪ variant `BEDC.BaseReflection.PsameEqClosure symm`
↪ variant `BEDC.BaseReflection.PsameEqClosure trans`

This relation is not the active relation for concrete Globalize exactness.

Lean4 ✓ `BEDC.BaseReflection.PsameEqClosure base intro`

Definition 9198.4 (Closure-reflection obligation)**.** *The optional equivalence-closure relation is reflection-safe only if the following additional theorem is proved:*

$$\mathsf{TokIntro}(\Pi,s,p),\quad \mathsf{TokIntro}(\Pi,t,q),\quad \mathsf{psame}^{\mathrm{eq}}_{\Pi}(p,q) \implies \mathsf{hsame}(s,t).$$

This obligation is named $\mathsf{ClosureReflect}(\Pi)$.
Lean4 (definition) `BEDC.BaseReflection.ClosureReflect`
Lean4 ✓ `BEDC.BaseReflection.ClosureReflect base edge`
↪ variant `BEDC.BaseReflection.ClosureReflect apply`

Theorem 9198.5 (Base reflection remains the active theorem)**.** *Assume* $\mathsf{TokUnique}(\Pi)$. *If*

$$\mathsf{TokIntro}(\Pi,s,p),\quad \mathsf{TokIntro}(\Pi,t,q),\quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(p,q),$$

then

$$\mathsf{hsame}(s,t).$$

↪ variant `BEDC.BaseReflection.PackageReflection base`

Proof. This is Theorem 9196.17 with the notation of Definition 9198.1. The only constructor of $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ exposes introducing signatures s_0, t_0 with $\mathsf{hsame}(s_0,t_0)$. Token-introduction uniqueness compares s with s_0 and t with t_0; symmetry and transitivity of hsame finish the proof. □

Corollary 9198.6 (Equivalence closure is not exported as exact)**.** *The exact concrete Globalize theorem exports* $\mathsf{psame}^{\mathrm{base}}_{\Pi}$*, not* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$*. Reflection through* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ *requires the explicit hypothesis* $\mathsf{ClosureReflect}(\Pi)$.

Lean4 ✓ `BEDC.BaseReflection.eqClosure export requires closure reflect`
↪ variant `BEDC.BaseReflection.NotExported GeneratedSameSig to base`

Proof. By Principle 9198.2, equivalence closure is a separate construction. Definition 9198.4 records the extra theorem required for reflection through that closure. The checked export theorem applies that predicate directly to the two token introductions and the closure proof, so the base relation remains the exact Globalize export. □

9198.2 Base Reflection Schema

This chapter isolates the smallest constructor-inversion chain needed for base package reflection. It is the first proof-check target in the present base-reflection contract.

Definition 9198.7 (Base package witness data)**.** *For fixed* Π, p, q*, define the witness predicate*

$$\mathsf{PBaseData}(\Pi, p, q) :\Longleftrightarrow \exists s_0, t_0,\ \mathsf{TokIntro}(\Pi, s_0, p) \wedge \mathsf{TokIntro}(\Pi, t_0, q) \wedge \mathsf{hsame}(s_0, t_0).$$

Lean4 ✓ `BEDC.BaseReflection.PBaseData witnesses`
↪ variant `BEDC.BaseReflection.PBaseData nonempty from witnesses`
↪ variant `BEDC.BaseReflection.PsameBase iff PBaseData`

Lemma 9198.8 (Base constructor)**.** *If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{hsame}(s, t),$$

then

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

Lean4 ✓ `BEDC.BaseReflection.PsameBase constructor`

Proof. This is the single constructor of Definition 9198.1. The important checker rule is that no second public constructor may be added without changing the public contract. □

Lemma 9198.9 (Base inversion)**.** *If* $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q)$*, then* $\mathsf{PBaseData}(\Pi, p, q)$*.*

Lean4 ✓ `BEDC.BaseReflection.PsameBase inversion`
↪ variant `BEDC.BaseReflection.psame base inversion`
↪ variant `BEDC.BaseReflection.PsameBase witnesses from base`
↪ variant `BEDC.BaseReflection.PsameBase inversion exists`
↪ variant `BEDC.BaseReflection.PsameBase single constructor witnesses`
↪ variant `BEDC.BaseReflection.PsameBase token intro witnesses`
↪ variant `BEDC.BaseReflection.PsameBase left token witness`
↪ variant `BEDC.BaseReflection.PsameBase right token witness`
↪ variant `BEDC.BaseReflection.PsameBase inversion data and witnesses`
↪ variant `BEDC.BaseReflection.PsameBase inversion witness object`

Proof. Invert the one-constructor inductive proof of $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q)$. The inversion exposes introducing signatures s_0, t_0 and a proof $\mathsf{hsame}(s_0, t_0)$. This is exactly $\mathsf{PBaseData}(\Pi, p, q)$. □

Lemma 9198.10 (Token replacement)**.** *Assume* $\mathsf{TokUnique}(\Pi)$*. If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, p),$$

then $\mathsf{hsame}(s, t)$*.*

Proof. This is the token-uniqueness field applied at the same visible package token p. In a canonical-constructor implementation it may instead be obtained by the no-confusion theorem described in Definition 9196.22. □

Lean4 ✓ `BEDC.BaseReflection.TokUnique replacement`

↪ variant `BEDC.BaseReflection.token replacement base`
↪ variant `BEDC.BaseReflection.tok unique replacement base`
↪ variant `BEDC.BaseReflection.TokUnique replacement pair`
↪ variant `BEDC.BaseReflection.TokUnique replacement from pair`
↪ variant `BEDC.BaseReflection.TokUnique replacement from shared token`
↪ variant `BEDC.BaseReflection.TokUnique replacement shared token triangle`

Theorem 9198.11 (Base package reflection)**.** *Assume* $\mathsf{TokUnique}(\Pi)$. *If*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q),$$

then

$$\mathsf{hsame}(s, t).$$

↪ variant `BEDC.BaseReflection.PackageReflection base`
↪ variant `BEDC.BaseReflection.PackageReflection base from data`

Proof. By Lemma 9198.9, choose witnesses s_0, t_0 such that

$$\mathsf{TokIntro}(\Pi, s_0, p), \quad \mathsf{TokIntro}(\Pi, t_0, q), \quad \mathsf{hsame}(s_0, t_0).$$

Token replacement at p gives $\mathsf{hsame}(s, s_0)$. Token replacement at q gives $\mathsf{hsame}(t, t_0)$, hence by symmetry $\mathsf{hsame}(t_0, t)$. Transitivity of hsame gives

$$\mathsf{hsame}(s, s_0), \quad \mathsf{hsame}(s_0, t_0), \quad \mathsf{hsame}(t_0, t) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

□

Protocol 9198.12 (Public theorem names)**.** *The checker-facing theorem names for this spine are:*

Paper object	***Suggested checker name***
Base constructor	`PsameBase.intro`
Base inversion	`PsameBase.inversion`
Token replacement	`TokUnique.tokenReplacement`
Base package reflection	`PackageReflection.base`

Lean4 ✓ `BEDC.BaseReflection.base reflection public name contract`

9198.3 Optional Closure Reflection Target

The equivalence closure of the base package relation is useful for package algebra, but it must not be confused with the active exact relation. This chapter records the closure relation and its conditional reflection theorem without exporting it as exactness.

Definition 9198.13 (Equivalence closure target)**.** *The optional relation remains*

$$\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q) := \mathsf{EqCl}(\mathsf{psame}^{\mathrm{base}}_{\Pi})(p, q).$$

Lean4 (definition) `BEDC.BaseReflection.PsameEqClosure`

It is not part of $\mathsf{ExactGlobalizeBase}(\Pi, D)$.

Protocol 9198.14 (Closure-reflection proof route)**.** *The theorem* `PackageReflection.eqClosure` *proves* $\mathsf{ClosureReflect}(\Pi)$ *by induction over* $\mathsf{EqCl}(\mathsf{psame}^{\mathrm{base}}_{\Pi})$*. The proof cases are:*

1. *reflexive case: use two introductions of the same token and* $\mathsf{TokUnique}(\Pi)$*;*
2. *base-edge case: use Theorem 9198.11;*
3. *symmetric case: use symmetry of* hsame*;*
4. *transitive case: use transitivity of* hsame*.*

Lean4 ✓ `BEDC.BaseReflection.closure reflection route cases`

Proposition 9198.15 (Conditional closure-reflection schema)**.** *Assume* hsame *is an equivalence relation, assume* $\mathsf{TokUnique}(\Pi)$*, assume every package has an introducing signature witness, and assume* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ *is implemented as the ordinary reflexive-symmetric-transitive closure of* $\mathsf{psame}^{\mathrm{base}}_{\Pi}$*. Then the proof route of Protocol 9198.14 proves*

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q) \Longrightarrow \mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.BaseReflection.PackageReflection eqClosure`

↪ variant `BEDC.BaseReflection.PsameEqClosure iff hsame under tok unique`

↪ variant `BEDC.BaseReflection.ClosureReflect from eqClosure`

↪ variant `BEDC.BaseReflection.conditional closure reflection schema`

↪ variant `BEDC.BaseReflection.PsameEqClosure endpoint reflection pair`

↪ variant `BEDC.BaseReflection.PsameEqClosure two base reflection`

↪ variant `BEDC.BaseReflection.PsameEqClosure three base reflection`

↪ variant `BEDC.BaseReflection.ClosureReflect from canonical`

Proof. Induct on the derivation of $\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q)$. The reflexive case compares two introductions of one token by token uniqueness. The base-edge case is Theorem 9198.11. The symmetric case reverses the induction result by symmetry of hsame. The transitive case chooses an introducing signature witness for the middle package and composes the two induction results by transitivity of hsame. □

Corollary 9198.16 (Four base-edge reflection)**.** *Assume* hsame *is an equivalence relation and* $\mathsf{TokUnique}(\Pi)$*. Let* p, q, r, u, w *be package tokens, with endpoint introductions* $\mathsf{TokIntro}(\Pi, x, p)$ *and* $\mathsf{TokIntro}(\Pi, v, w)$*, and with introducing signature witnesses for* q*,* r*, and* u*. If*

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q), \quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(q, r), \quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(r, u), \quad \mathsf{psame}^{\mathrm{base}}_{\Pi}(u, w),$$

then $\mathsf{hsame}(x, v)$*.*

Lean4 ✓ `BEDC.BaseReflection.PsameEqClosure four base reflection`

Proof. Apply base-package reflection to each adjacent base edge, using the middle introducing witnesses for the internal package tokens. Transitivity of hsame composes the four reflected comparisons into the endpoint comparison. □

Corollary 9198.17 (Closure reflection yields generated signature sameness)**.** *When a package pair is related by* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ *and both endpoints have token introductions, a closure-reflection witness produces the corresponding generated signature sameness comparison.*

Lean4 ✓ `BEDC.BaseReflection.ClosureReflect to GeneratedSameSig`

Proof. Unfold the generated signature sameness target for the introduced endpoints. The closure-reflection witness applies directly to the two introductions and the equivalence-closure premise, yielding the required hsame comparison. □

Corollary 9198.18 (Closure reflection preserves base export)**.** *Adding a closure-reflection witness does not change the exported package relation of* $\mathsf{ExactGlobalizeBase}(\Pi, D)$*: exactness continues to export* $\mathsf{psame}^{\mathrm{base}}_{\Pi}$*.*

Lean4 ✓ `BEDC.BaseReflection.ClosureReflect preserves base export`

Proof. The export discipline is Protocol 9199.9 together with Corollary 9199.18. Closure reflection is consumed only as a local theorem for introduced endpoints, while the exactness predicate continues to name $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ as its package relation. □

Theorem 9198.19 (Finite equivalence-closure reflection obstruction)**.** *Let* Π *be a generated probe bundle. Suppose there are signatures* a, b, c, d *and package tokens* p, r, q *with token introductions*

$$\mathsf{TokIntro}(\Pi, a, p), \quad \mathsf{TokIntro}(\Pi, c, r), \quad \mathsf{TokIntro}(\Pi, d, r), \quad \mathsf{TokIntro}(\Pi, b, q),$$

and history-sameness data

$$\mathsf{hsame}(a, c), \quad \mathsf{hsame}(d, b), \quad \neg\mathsf{hsame}(c, d), \quad \neg\mathsf{hsame}(a, b).$$

Then the finite configuration satisfies

$$\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q), \qquad \neg\,\mathsf{TokUnique}(\Pi), \qquad \neg\,\mathsf{ClosureReflect}(\Pi).$$

In particular, the endpoints a *and* b *introduce the closure-related tokens* p *and* q *but fail to reflect to* $\mathsf{hsame}(a, b)$*.*

Proof. The first and second token-introduction judgments, together with $\mathsf{hsame}(a, c)$, give

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, r)$$

by Definition 9198.8. The third and fourth token-introduction judgments, together with $\mathsf{hsame}(d, b)$, give

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(r, q)$$

by the same constructor. Since $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ is the equivalence closure of $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ in Definition 9198.13, and Definition 9198.14 records the base-edge and transitive closure cases, the two displayed base edges compose to $\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q)$.

If $\mathsf{TokUnique}(\Pi)$ held, Definition 9196.13 applied to $\mathsf{TokIntro}(\Pi, c, r)$ and $\mathsf{TokIntro}(\Pi, d, r)$ would give $\mathsf{hsame}(c, d)$, contradicting the displayed separation. Hence $\neg\,\mathsf{TokUnique}(\Pi)$.

If $\mathsf{ClosureReflect}(\Pi)$ held, unfolding Definition 9198.4 and applying it to $\mathsf{TokIntro}(\Pi, a, p)$, $\mathsf{TokIntro}(\Pi, b, q)$, and the closure proof $\mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q)$ would give $\mathsf{hsame}(a, b)$, contradicting the displayed endpoint separation. Hence $\neg\,\mathsf{ClosureReflect}(\Pi)$. □

Corollary 9198.20 (Equivalence-closure exactness needs reflection)**.** *The base exact Globalize contract of Definition 9199.11 cannot be converted into an exactness statement for* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ *by replacing the exported relation name alone. Any such conversion must add* $\mathsf{ClosureReflect}(\Pi)$*, or hypotheses such as those in Definition 9198.15 that prove it.*

Proof. An exactness statement at $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$ needs the endpoint reflection step

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q), \quad \mathsf{psame}^{\mathrm{eq}}_{\Pi}(p, q) \quad \Longrightarrow \quad \mathsf{hsame}(s, t).$$

The finite configuration in Definition 9198.19 supplies the antecedents for $s = a$ and $t = b$ and also supplies $\neg\mathsf{hsame}(a, b)$. Thus the reflection step is not derivable from the base constructor and equivalence closure alone. The export relation in Definition 9199.11 is $\mathsf{psame}^{\mathrm{base}}_{\Pi}$, and Definition 9199.18 records that the closure relation is not exported as exactness. □

Chapter 9199

Exact Globalize Base Contract

9199.1 Exact Concrete Signature Globalize Under Explicit Policies

The concrete signature Globalize instance becomes exact only when the checked concrete inputs are visible:

$$\mathsf{AskPolicy}(\Pi, D) + \mathsf{PackageTokenPolicy}(\Pi) + \forall h, s, \Delta,\ \mathsf{InDom}(D, h) \wedge \mathsf{Sig}(\Pi, h, s, \Delta) \Longrightarrow \exists p,\ \mathsf{TokIntro}(\Pi, s, p).$$

This chapter records the exactness theorem in its non-overclaiming form. Domain transport is the separate policy recorded in Theorem 9195.2.

The predicate $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ is Definition 9193.1.

Theorem 9199.1 (Concrete gap coverage)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and the dependent generated-signature token condition: whenever* $\mathsf{InDom}(D, h)$ *and* $\mathsf{Sig}(\Pi, h, s, \Delta)$*, there exists a token* p *with* $\mathsf{TokIntro}(\Pi, s, p)$*. If* $\mathsf{InDom}(D, h)$*, then there exists a token* p *such that*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Lean4 ✓ `BEDC.FKernel.Gap.gap coverage globalize`

Proof. By the totality field of $\mathsf{AskPolicy}(\Pi, D)$, the admitted history h generates some signature s with evidence Δ. The dependent token condition applied to $\mathsf{InDom}(D, h)$ and $\mathsf{Sig}(\Pi, h, s, \Delta)$ supplies a token p with $\mathsf{TokIntro}(\Pi, s, p)$. The three displayed judgments witness $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. □

Theorem 9199.2 (Concrete gap separation)**.** *Let* $\mathsf{psame}^{\mathrm{sig}}_{\Pi}$ *be the concrete package-sameness relation generated by its signature-sameness constructor. Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h),$$

then

$$\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q).$$

Lean4 ✓ `BEDC.FKernel.Gap.gap separation globalize`

↪ variant `BEDC.FKernel.Gap.gap separation with domain pair`

↪ variant `BEDC.FKernel.Gap.signature gap separation from witnesses`

↪ variant `BEDC.FKernel.Gap.gap separation same signature witness pair`

Proof. The two gap premises yield the admitted-source witness $\mathsf{InDom}(D, h)$ and signatures s, t with

$$\mathsf{Sig}(\Pi, h, s, \Delta), \quad \mathsf{Sig}(\Pi, h, t, \Theta),$$

and token-introduction judgments

$$\mathsf{TokIntro}(\Pi, s, p), \quad \mathsf{TokIntro}(\Pi, t, q).$$

Theorem 6.67, applied to $\mathsf{AskPolicy}(\Pi, D)$, the admitted-source witness, and the two generated signatures, gives $\mathsf{hsame}(s, t)$. The package-sameness constructor then gives $\mathsf{psame}^{\mathrm{sig}}_{\Pi}(p, q)$. □

Theorem 9199.3 (Concrete Globalize soundness)**.** *Given a witness $G : \mathsf{GeneratedSameSig}(\Pi, h, k)$, if the left generated signature object of G introduces token p and the right generated signature object of G introduces token q, then $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q)$.*

Lean4 ✓ `BEDC.BaseReflection.GeneratedSameSig psameBase`

Proof. The witness G carries the history-sameness component between its generated signature objects. The base package-sameness constructor applied to the two token-introduction premises and that component gives $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q)$. □

Theorem 9199.4 (Concrete Globalize completeness)**.** *Assume $\mathsf{TokUnique}(\Pi)$. Given generation judgments $\mathsf{SigGen}(\Pi, h, x, \Delta)$ and $\mathsf{SigGen}(\Pi, k, y, \Theta)$, token-introduction judgments $\mathsf{TokIntro}(\Pi, x, p)$ and $\mathsf{TokIntro}(\Pi, y, q)$, and $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q)$, there is an element of $\mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k))$.*

Lean4 ✓ `BEDC.BaseReflection.PsameBase to GeneratedSameSig under tok unique`

Proof. Token-uniqueness reflection converts the base package-sameness premise, together with the two token-introduction judgments, into history-sameness of x and y. Combining that reflected sameness with the two generation judgments fills the fields of a $\mathsf{GeneratedSameSig}(\Pi, h, k)$ witness. □

9199.2 Exactness Export Contract

This chapter records the three-clause gap export used for active development. It is intentionally narrower than a full package algebra theorem.

Theorem 9199.5 (Local signature classification)**.** *Assume $\mathsf{AskPolicy}(\Pi, D)$, $\mathsf{PackageTokenPolicy}(\Pi)$,*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k).$$

Then package-sameness is classified by generated signatures:

$$\mathsf{psame}(p, q) \iff \exists s, t, \Delta, \Theta,\ \mathsf{Sig}(\Pi, h, s, \Delta) \textit{ and } \mathsf{Sig}(\Pi, k, t, \Theta) \textit{ and } \mathsf{hsame}(s, t).$$

Lean4 ✓ `BEDC.FKernel.Gap.internalized globalize classifies by signatures`

Proof. The forward direction is the concrete completeness clause applied to the two gap memberships and the package-sameness premise. For the reverse direction, unpack the two $\mathsf{InGap}^{\mathrm{sig}}$ premises. The p-membership supplies a domain witness for h, a generated signature u, and the token-introduction witness for p; the q-membership supplies a domain witness for k, a generated signature v, and the token-introduction witness for q. Separately unpack the displayed existential

into signatures s from h and t from k, together with $\mathsf{hsame}(s,t)$. Asking-policy determinacy at h gives $\mathsf{hsame}(s,u)$, and determinacy at k gives $\mathsf{hsame}(t,v)$. By symmetry and transitivity, these three history-sameness facts compose to $\mathsf{hsame}(u,v)$. The package-token policy soundness field then consumes the two token-introduction witnesses and $\mathsf{hsame}(u,v)$ to conclude $\mathsf{psame}(p,q)$. □

Theorem 9199.6 (Exact Globalize base soundness export)**.** *Assume* $\mathsf{ExactGlobalizeBase}(\Pi, D)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k), \qquad \mathsf{GeneratedSameSig}(\Pi, h, k),$$

then $\mathsf{psame}^{\mathrm{base}}_{\Pi}(p,q)$*.*

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase soundness exports base`

Proof. This is the soundness projection of $\mathsf{ExactGlobalizeBase}(\Pi, D)$ applied to the two gap memberships and the generated-sameness witness. □

Definition 9199.7 (Not-exported boundary predicate)**.** *The predicate* $\mathsf{NotExported}(\Pi, D, E)$ *records the boundary attached to an exact base Globalize witness. It holds exactly when, for all histories* h, k *and packages* p, q*, the two gap memberships*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k)$$

bound the exported statement to

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p,q) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

Lean4 (definition) `BEDC.BaseReflection.NotExported`

Proposition 9199.8 (What is intentionally not exported)**.** *The following statements are not exported:*

1. $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$*-exactness;*
2. *package-sameness transitivity without token-uniqueness hypotheses as a constructor-level theorem for* $\mathsf{psame}^{\mathrm{base}}_{\Pi}$*;*
3. *equivalence-closure exactness as a Globalize export;*
4. $\mathsf{Add}^{\uparrow}$ *certificate data as a clause of the exact Globalize export;*
5. *a full natural-number or addition construction beyond the licensed unary-continuation interface.*

Proof. Each item requires an exactness predicate whose exported relation is not $\mathsf{psame}^{\mathrm{base}}_{\Pi}$, a derived-interface certificate carried as separate naming-certificate data, or a derived-object construction that is not a field of the exact Globalize predicate. □

Lean4 ✓ `BEDC.BaseReflection.NotExported scope boundary`

Protocol 9199.9 (Export discipline)**.** *Modules cite Definition 9199.11 for the exactness predicate and Theorem 9199.16 for its field constructor. If a module needs* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$*-exactness, it must state an exactness predicate whose exported relation is* $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$*, or explicitly downgrade its result from exact reflection to closure-level compatibility.*

9199.3 Checker-Friendly Exactness Predicate

Exact concrete Globalize is the base-relation exactness predicate whose data are coverage, base soundness, and base completeness between $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ and $\mathsf{GeneratedSameSig}(\Pi, h, k)$, with the local iff derived from the two directional fields.

Definition 9199.10 (Generated signature sameness)**.** *For a generated probe bundle* Π*,* $\mathsf{GeneratedSameSig}(\Pi, h, k)$ *is a witness object. Its fields are signatures* s, t*, evidence objects* Δ, Θ*, generation judgments* $\mathsf{Sig}(\Pi, h, s, \Delta)$ *and* $\mathsf{Sig}(\Pi, k, t, \Theta)$*, and a sameness witness* $\mathsf{hsame}(s, t)$*.*

The existential reading is the inhabitation proposition for that object:

$$\mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)) \iff \exists s, t, \Delta, \Theta,\ \mathsf{Sig}(\Pi, h, s, \Delta) \wedge \mathsf{Sig}(\Pi, k, t, \Theta) \wedge \mathsf{hsame}(s, t).$$

Thus the checker-facing data corresponding to "the two admitted histories generate the same visible signature" is the witness object itself, while classification statements use its inhabitation.

```
Lean4 (definition)  BEDC.BaseReflection.GeneratedSameSig
Lean4 ✓  BEDC.BaseReflection.GeneratedSameSig hsame
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig from witnesses
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig refl nonempty from sig
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig left witness
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig right witness
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig witnesses
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig nonempty witnesses
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig iff witnesses
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig swap witnesses
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig symm
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig symm nonempty
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig trans nonempty under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig trans under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig trans nonempty direct under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig chain witnesses under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig trans nonempty witnesses under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig trans witness object under determinacy
  ↪ variant  BEDC.BaseReflection.GeneratedSameSig equivalence nonempty under determinacy
```

Definition 9199.11 (Base exact Globalize structure)**.** *For a bundle* Π *and a concrete domain policy* D*, the proposition*

$$\mathsf{ExactGlobalizeBase}(\Pi, D)$$

is the conjunction of the following three structural fields.

1. ***Coverage.***

$$\forall h,\ \mathsf{InDom}(D, h) \Rightarrow \exists p,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

2. ***Base soundness.***

$$\forall h, k, p, q,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \Rightarrow \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \Rightarrow \mathsf{GeneratedSameSig}(\Pi, h, k) \Rightarrow \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

3. ***Base completeness.***

$$\forall h, k, p, q,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \Rightarrow \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \Rightarrow \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \Rightarrow \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)$$

The local classification equivalence is derived from soundness and completeness.

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase iff fields`

↪ variant `BEDC.BaseReflection.ExactGlobalizeBase soundness`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase soundness nonempty`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase completeness`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase sound complete`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase constructor`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase public export shape`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase four field export`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase field projection pair`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase coverage pair`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase sound complete and classify`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase local sound complete classification`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase coverage sound complete classify`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase coverage and classification`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase classification four fields`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase classification field`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase symmetric classify iff`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase admitted pair export shape`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase from fields covered classification`

Proposition 9199.12 (Classification is derivable from the two directions)**.** *If base soundness and base completeness hold, then for all h, k, p, q satisfying*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \textit{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k),$$

one has

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase classify iff from directions`

↪ variant `BEDC.BaseReflection.ExactGlobalizeBase classify directions`

Proof. The forward implication is base completeness. For the reverse implication, unpack the inhabited witness proposition and apply base soundness to the resulting $\mathsf{GeneratedSameSig}(\Pi, h, k)$ object. Coverage is not needed for this local equivalence, but it remains part of the global exactness structure because exact Globalize must also classify every admitted history, not only pairs already carrying visible packages. □

Corollary 9199.13 (Exact Globalize projections and classification)**.** *From* $\mathsf{ExactGlobalizeBase}(\Pi, D)$ *one obtains coverage, base soundness, base completeness, and the derived local classification equivalence*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \wedge \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \quad \Rightarrow \quad (\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k))).$$

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase coverage soundness completeness and classify`

Proof. The first three clauses are the projections of $\mathsf{ExactGlobalizeBase}(\Pi, D)$. The displayed equivalence is Proposition 9199.12. □

Theorem 9199.14 (Exact Globalize self-covered classification)**.** *Let* $\mathsf{ExactGlobalizeBase}(\Pi, D)$ *hold. For every admitted history* h*, there is a package* p *covering* h *such that the local classification relates* h *to itself through the package pair* (p, p)*:*

$$\mathsf{InDom}(D, h) \quad \Rightarrow \quad \exists p,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \wedge \big(\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, p) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, h))\big).$$

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase self covered classification`

Proof. Use the coverage component isolated in Corollary 9199.13 for the admitted history h. It supplies a package p with $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$.

Apply the same corollary's local classification equivalence to the two identical displayed witnesses. With both sides instantiated by p and h, the pairwise equivalence becomes exactly the self-covered classification statement. □

Protocol 9199.15 (Exactness export discipline)**.** *Global existence cites Definition 9199.11. Pairwise classification cites Proposition 9199.12 or Corollary 9199.13 under displayed* $\mathsf{InGap}^{\mathrm{sig}}$ *witnesses. Coverage is used only when the statement must produce visible package tokens for admitted histories.*

9199.4 Exact Globalize Base Contract

The theorem packages coverage, base soundness, and base completeness at the base package relation into the concrete signature Globalize exactness predicate.

Theorem 9199.16 (Exact Globalize base target)**.** *Assume the following three field hypotheses:*

1. *coverage:*
$$\forall h,\ \mathsf{InDom}(D, h) \Rightarrow \exists p,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h);$$
2. *soundness:*
$$\forall h, k, p, q,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \Rightarrow \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \Rightarrow \mathsf{GeneratedSameSig}(\Pi, h, k) \Rightarrow \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q);$$
3. *completeness:*
$$\forall h, k, p, q,\ \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \Rightarrow \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k) \Rightarrow \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \Rightarrow \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)$$

Then
$$\mathsf{ExactGlobalizeBase}(\Pi, D).$$

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase from fields`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase fields`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase coverage sound complete`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase from fields classify iff`
↪ variant `BEDC.BaseReflection.globalize exact base`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase target`
↪ variant `BEDC.BaseReflection.exact globalize base target`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase export from fields`
↪ variant `BEDC.BaseReflection.ExactGlobalizeBase coverage projection`
↪ variant `BEDC.BaseReflection.no scaffold laundering fields and classifier`

```
↪ variant BEDC.BaseReflection.ExactGlobalizeBase covered classification
↪ variant BEDC.BaseReflection.ExactGlobalizeBase covered sound complete
↪ variant BEDC.BaseReflection.ExactGlobalizeBase covered soundness direction
↪ variant BEDC.BaseReflection.ExactGlobalizeBase covered completeness direction
↪ variant BEDC.BaseReflection.ExactGlobalizeBase covered classification directions
```

Proof. The theorem is the constructor for $\mathsf{ExactGlobalizeBase}(\Pi, D)$ from its coverage, soundness, and completeness fields. The soundness field consumes an actual $\mathsf{GeneratedSameSig}(\Pi, h, k)$ witness object; the completeness field returns the inhabited-witness proposition. □

Theorem 9199.17 (Exact Globalize admitted-pair export from fields)**.** *Assume the three field hypotheses of Theorem 9199.16. For admitted histories h and k with $\mathsf{InDom}(D, h)$ and $\mathsf{InDom}(D, k)$, there exist packages p and q such that*

$$\mathsf{ExactGlobalizeBase}(\Pi, D) \wedge \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \wedge \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k)$$
$$\wedge\ \left(\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k))\right).$$

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase from fields admitted pair export`

Proof. First build the exactness witness from the three fields by Theorem 9199.16. This gives $\mathsf{ExactGlobalizeBase}(\Pi, D)$ with the same coverage, soundness, and completeness data.

Apply coverage to the displayed admission $\mathsf{InDom}(D, h)$ to obtain a package p with $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. Apply coverage again to $\mathsf{InDom}(D, k)$ to obtain a package q with $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k)$.

The local classification projection for $\mathsf{ExactGlobalizeBase}(\Pi, D)$ applies to these two displayed gap-signature witnesses. Its equivalence is precisely

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \iff \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

Together with the constructed exactness witness and the two coverage witnesses, this gives the required package pair. □

Corollary 9199.18 (No closure export)**.** *Theorem 9199.16 exports exactness only at $\mathsf{psame}^{\mathrm{base}}_{\Pi}$. It does not export exactness at $\mathsf{psame}^{\mathrm{eq}}_{\Pi}$.*

Proof. The statement concludes $\mathsf{ExactGlobalizeBase}(\Pi, D)$, whose relation field is $\mathsf{psame}^{\mathrm{base}}_{\Pi}$ by Definition 9199.11. The optional equivalence closure has the local reflection theorem of Proposition 9198.15; that theorem does not replace the relation field of $\mathsf{ExactGlobalizeBase}(\Pi, D)$. □

Lean4 ✓ `BEDC.BaseReflection.ExactGlobalizeBase relation is base`

```
↪ variant BEDC.BaseReflection.ExactGlobalizeBase exports base relation
↪ variant BEDC.BaseReflection.ExactGlobalizeBase no closure export
↪ variant BEDC.BaseReflection.ExactGlobalizeBase no closure export directions
↪ variant BEDC.BaseReflection.ExactGlobalizeBase no closure export sound complete pair
↪ variant BEDC.BaseReflection.ClosureReflect preserves base sound complete pair
↪ variant BEDC.BaseReflection.ExactGlobalizeBase no closure export classification iff
↪ variant BEDC.BaseReflection.ExactGlobalizeBase no eqClosure export
```

Protocol 9199.19 (Acceptance gate for a checked report)**.** *This target may be upgraded from paper-level to checked only if the report includes all of the following:*

1. *a proof-assistant build command and version;*

2. *the public theorem name* `Globalize.exactBase`*;*
3. *no* `sorry`*,* `admit`*, or unchecked postulate in the proof of* `Globalize.exactBase`*;*
4. *no public theorem that replaces* hsame *by host equality of histories;*
5. *no public theorem that replaces* TokIntro *by a host function without a relational wrapper;*
6. *a list of every remaining axiom, postulate, or imported theorem used by the checked artifact.*

`Lean4✓ BEDC.BaseReflection.checked acceptance gate exactBase exports public shape`

Theorem 9199.20 (Generated signature witness inhabitation)**.** *For every generated probe bundle* Π *and histories* h, k*, the following are equivalent:*

1. *there is an element* $G : \mathsf{GeneratedSameSig}(\Pi, h, k)$*;*
2. $\mathsf{sameSig}_{\Pi}(h, k)$*.*

Proof. Unpack Definition Definition 9199.10. An element $G : \mathsf{GeneratedSameSig}(\Pi, h, k)$ consists of histories s, t, evidence objects Δ, Θ, generation judgments

$$\mathsf{Sig}(\Pi, h, s, \Delta) \quad \text{and} \quad \mathsf{Sig}(\Pi, k, t, \Theta),$$

and a sameness witness $\mathsf{hsame}(s, t)$. These are exactly the witnesses required by Definition Definition 6.75 for $\mathsf{sameSig}_{\Pi}(h, k)$. This proves the forward implication.

Conversely, unpack $\mathsf{sameSig}_{\Pi}(h, k)$ using Definition Definition 6.75. It supplies histories s, t, evidence objects Δ, Θ, the two displayed generation judgments, and $\mathsf{hsame}(s, t)$. By Definition Definition 9199.10, these fields assemble an element of $\mathsf{GeneratedSameSig}(\Pi, h, k)$. □

Theorem 9199.21 (Generated signature sameness closure under bundle append)**.** *For probe bundles* Π, Θ *and histories* h, k*, if* $G : \mathsf{GeneratedSameSig}(\Pi, h, k)$ *and* $H : \mathsf{GeneratedSameSig}(\Theta, h, k)$*, then*

$$\mathsf{GeneratedSameSig}(\mathsf{BAppend}(\Pi, \Theta), h, k).$$

Proof. Let Ω be the appended bundle with $\Omega = \mathsf{BAppend}(\Pi, \Theta)$. Unpack G as signatures s_h, s_k with evidence Δ_h, Δ_k,

$$\mathsf{Sig}(\Pi, h, s_h, \Delta_h), \qquad \mathsf{Sig}(\Pi, k, s_k, \Delta_k), \qquad \mathsf{hsame}(s_h, s_k),$$

and unpack H as signatures t_h, t_k with evidence Γ_h, Γ_k,

$$\mathsf{Sig}(\Theta, h, t_h, \Gamma_h), \qquad \mathsf{Sig}(\Theta, k, t_k, \Gamma_k), \qquad \mathsf{hsame}(t_h, t_k).$$

Apply Definition 6.48 to the two h-side signatures, with appended-bundle witness Ω. This gives a history u_h and evidence Λ_h such that

$$\mathsf{Cont}(t_h, s_h, u_h) \quad \text{and} \quad \mathsf{Sig}(\Omega, h, u_h, \Lambda_h).$$

Apply the same theorem to the two k-side signatures, with the same appended-bundle witness Ω. This gives a history u_k and evidence Λ_k such that

$$\mathsf{Cont}(t_k, s_k, u_k) \quad \text{and} \quad \mathsf{Sig}(\Omega, k, u_k, \Lambda_k).$$

History-sameness reflexivity gives $\mathsf{hsame}(u_h, u_h)$. Using $\mathsf{hsame}(t_h, t_k)$, $\mathsf{hsame}(s_h, s_k)$, $\mathsf{hsame}(u_h, u_h)$, and the two displayed continuation witnesses, Definition 4.42 yields $\mathsf{hsame}(u_h, u_k)$. The generated-sameness witness for Ω is therefore filled by the two appended signature judgments and this final history-sameness component, exactly as in Definition 9199.10. Since $\Omega = \mathsf{BAppend}(\Pi, \Theta)$, this is the required generated-sameness witness for $\mathsf{BAppend}(\Pi, \Theta)$. □

Definition 9199.22 (Concrete source subdomain)**.** *A concrete source domain D' is a subdomain of D, written*

$$\mathsf{SubDom}(D', D),$$

when every D'-admitted source is D-admitted:

$$\forall h,\ \mathsf{InDom}(D', h) \Longrightarrow \mathsf{InDom}(D, h).$$

The condition changes only the source-admission judgment. It does not change the bundle Π, generated-signature judgments, token-introduction judgments, or the base package relation.

Lemma 9199.23 (Subdomain gap base-change)**.** *Assume $\mathsf{SubDom}(D', D)$. For every bundle Π, token p, and source history h, if $\mathsf{InDom}(D', h)$, then*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h) \Longleftrightarrow \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Proof. Unpack $\mathsf{InGap}^{\mathrm{sig}}$ by Definition 9193.1. A D'-gap membership consists of $\mathsf{InDom}(D', h)$, a generated signature $\mathsf{Sig}(\Pi, h, s, \Delta)$, and a token-introduction judgment $\mathsf{TokIntro}(\Pi, s, p)$. The subdomain premise sends the admitted-source part to $\mathsf{InDom}(D, h)$, while the generated signature and token-introduction witnesses are unchanged. These three data repack as $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$.

Conversely, assume $\mathsf{InDom}(D', h)$ and unpack $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. The unpacked membership supplies a generated signature $\mathsf{Sig}(\Pi, h, s, \Delta)$ and a token-introduction judgment $\mathsf{TokIntro}(\Pi, s, p)$. Replacing only the admitted-source certificate by the given $\mathsf{InDom}(D', h)$ repacks the same signature and token witnesses as $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h)$. □

Theorem 9199.24 (Exact Globalize base subdomain base-change)**.** *Assume $\mathsf{ExactGlobalizeBase}(\Pi, D)$ and $\mathsf{SubDom}(D', D)$. Then the concrete signature Globalize structure restricted through $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h)$ satisfies*

$$\mathsf{ExactGlobalizeBase}(\Pi, D').$$

Proof. By Definition 9199.11 and Definition 9199.16, it is enough to provide coverage, base soundness, and base completeness over D'.

For coverage, fix h and assume $\mathsf{InDom}(D', h)$. The subdomain premise gives $\mathsf{InDom}(D, h)$. The coverage field of $\mathsf{ExactGlobalizeBase}(\Pi, D)$ supplies a token p with

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

The reverse direction of Definition 9199.23, applied with the original D'-admission premise, converts this membership into

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h),$$

which proves D'-coverage.

For soundness, assume

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D', q, k), \qquad \mathsf{GeneratedSameSig}(\Pi, h, k).$$

Each D'-gap membership exposes its own D'-admitted-source certificate by Definition 9193.1. The forward direction of Definition 9199.23 converts the two memberships to

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \quad \text{and} \quad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k).$$

Applying the base soundness field of $\mathsf{ExactGlobalizeBase}(\Pi, D)$ to these two converted memberships and the displayed generated-sameness witness gives

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

For completeness, assume

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D', p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D', q, k), \qquad \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q).$$

As in the soundness argument, the two D'-gap memberships transport through Definition 9199.23 to D-gap memberships with the same histories and visible tokens. The base completeness field of $\mathsf{ExactGlobalizeBase}(\Pi, D)$ then yields

$$\mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

These three clauses are precisely the fields required for $\mathsf{ExactGlobalizeBase}(\Pi, D')$. □

Chapter 9200

Unary Shift and Commutativity

9200.1 Unary Commutativity Proof Obligations

The additive boundary correctly states that $\mathsf{Add}^{\uparrow}$ requires unary continuation commutativity. This chapter decomposes that commutativity proof into smaller obligations. These obligations are more suitable for machine verification than a single opaque paper proof.

Definition 9200.1 (Unary right-shift obligation)**.** *The unary right-shift obligation is the statement that if*

$$\mathsf{Unary}(k), \qquad \mathsf{Cont}(k, \mathsf{E1}(h), r),$$

then there exists a history v such that

$$\mathsf{Cont}(k, h, v) \quad \textit{and} \quad \mathsf{hsame}(r, \mathsf{E1}(v)).$$

Lean4 (definition) `BEDC.FKernel.Unary.UnaryRightShiftObligation`
↪ variant `BEDC.FKernel.Unary.UnaryRightShiftObligation iff fields`
Lean4 ✓ `BEDC.FKernel.Unary.unary right shift obligation holds`

Definition 9200.2 (E1-congruence obligation)**.** *The* E1*-congruence obligation is the statement that*

$$\mathsf{hsame}(u, v) \quad \Rightarrow \quad \mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(v)).$$

Lean4 (definition) `BEDC.FKernel.Unary.EOneCongruenceObligation`
Lean4 ✓ `BEDC.FKernel.Unary.EOneCongruenceObligation holds`
↪ variant `BEDC.FKernel.Unary.eone congruence obligation holds`

Definition 9200.3 (Unary continuation commutativity obligation)**.** *The unary continuation commutativity obligation is the statement that if*

$$\mathsf{Unary}(h), \quad \mathsf{Unary}(k), \quad \mathsf{Cont}(h, k, r), \quad \mathsf{Cont}(k, h, r'),$$

then

$$\mathsf{hsame}(r, r').$$

Theorem 9200.4 (Commutativity from right-shift and E1-congruence)**.** *Assume a proof of the unary right-shift obligation and a proof of the* E1*-congruence obligation. Then for all histories h, k, r, r', if* $\mathsf{Unary}(h)$*,* $\mathsf{Unary}(k)$*,* $\mathsf{Cont}(h, k, r)$*, and* $\mathsf{Cont}(k, h, r')$*, then* $\mathsf{hsame}(r, r')$*.*

Lean4 ✓ BEDC.FKernel.Unary.comm from obligations

Proof. Induct on the unary derivation of the left input h. The empty case follows from Theorem 4.38. In the step case, write $h = \mathsf{E1}(h_0)$. Invert $\mathsf{Cont}(\mathsf{E1}(h_0), k, r)$ to obtain u with $\mathsf{Cont}(h_0, k, u)$ and $\mathsf{hsame}(r, \mathsf{E1}(u))$. Apply unary right-shift to $\mathsf{Cont}(k, \mathsf{E1}(h_0), r')$ to obtain v with $\mathsf{Cont}(k, h_0, v)$ and $\mathsf{hsame}(r', \mathsf{E1}(v))$. By the induction hypothesis, $\mathsf{hsame}(u, v)$. By $\mathsf{E1}$-congruence, $\mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(v))$. Transitivity of hsame yields $\mathsf{hsame}(r, r')$. □

↪ variant BEDC.FKernel.Unary.unary commutativity obligation from named obligations

Corollary 9200.5 (Additive naming boundary)**.** *The name* $\mathsf{Add}^{\uparrow}$ *is licensed only after the unary right-shift and* $\mathsf{E1}$*-congruence obligations have been supplied. Before that point, the safe name is* $\mathsf{UnaryContMonoid}^{\uparrow}$*.*

Lean4 ✓ BEDC.FKernel.Unary.additive naming boundary from obligations

Proof. The monoid-like name requires unit and associativity. The additive name requires the additional commutativity field. Theorem 9200.4 reduces that field to the two named obligations. □

9200.2 Unary Shift and the Additive License

section 9200.1 decomposes unary continuation commutativity into auxiliary obligations. This section states the concrete instances of those obligations whose proofs license the additive name $\mathsf{Add}^{\uparrow}$.

Lemma 9200.6 ($\mathsf{E1}$-congruence)**.** *If* $\mathsf{hsame}(h, k)$*, then*

$$\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k)).$$

Lean4 ✓ BEDC.FKernel.Hist.eone congruence
↪ variant BEDC.FKernel.Hist.hsame e1 congruence

Proof. This is the $\mathsf{E1}$-constructor rule for hsame. □

Lemma 9200.7 (Unary right-shift, witness form)**.** *Assume* $\mathsf{Unary}(k)$ *and* $\mathsf{Cont}(k, \mathsf{E1}(h), r')$*. Then there exists a history* v *such that*

$$\mathsf{Cont}(k, h, v) \qquad \textit{and} \qquad \mathsf{hsame}(r', \mathsf{E1}(v)).$$

Proof. This is the proof obligation named $\mathsf{UnaryShift}$. The proof proceeds by induction on the unary derivation of k. The empty case uses the left-unit law for Cont. The step case uses inversion on the continuation derivation, the induction hypothesis for the shorter unary history, $\mathsf{E1}$-congruence, and transitivity of hsame. □

Lean4 ✓ BEDC.FKernel.Unary.unary shift witness closed symmetric result pair
↪ variant BEDC.FKernel.Unary.unary shift witness primary
↪ variant BEDC.FKernel.Unary.unary shift witness with factor
↪ variant BEDC.FKernel.Unary.unary shift witness exact append
↪ variant BEDC.FKernel.Unary.unary shift witness result unique pair
↪ variant BEDC.FKernel.Unary.unary shift witness two results unique

↪ variant BEDC.FKernel.Unary.unary shift witness transport result
↪ variant BEDC.FKernel.Unary.unary shift witness closed result pair
↪ variant BEDC.FKernel.Unary.unary shift witness closed unique result
↪ variant BEDC.FKernel.Unary.unary shift witness induction closed unique
↪ variant BEDC.FKernel.Unary.unary shift step result unique pair
↪ variant BEDC.FKernel.Unary.unary shift witness by induction
↪ variant BEDC.FKernel.Unary.unary shift witness by induction result closed
↪ variant BEDC.FKernel.Unary.unary shift witness obligation pair
↪ variant BEDC.FKernel.Unary.unary shift witness tail unique congruence

Lemma 9200.8 (Unary right-shift result closure)**.** *If* $\mathsf{Unary}(k)$*,* $\mathsf{Unary}(h)$*, and* $\mathsf{Cont}(k, \mathsf{E1}(h), r')$*, then* $\mathsf{Unary}(r')$*.*

Lean4 ✓ BEDC.FKernel.Unary.unary shift result closed
↪ variant BEDC.FKernel.Unary.unary shift closed witness and result

Lemma 9200.9 (Unary append commutativity, left induction)**.** *If* h *and* k *are unary histories, then appending them commutes:*

$$\mathsf{append}(h, k) = \mathsf{append}(k, h).$$

Lean4 ✓ BEDC.FKernel.Unary.unary append comm left induction
↪ variant BEDC.FKernel.Unary.unary append comm hsame left induction
↪ variant BEDC.FKernel.Unary.unary append comm right induction closed pair

Proof. Induct on the unary derivation of h. The empty case is the left-unit law for append. The $\mathsf{E0}$-case is impossible for unary histories. The $\mathsf{E1}$-case follows by moving the leading $\mathsf{E1}$ through append and applying the induction hypothesis. □

9200.3 Unary Right-Shift: Micro-Lemmas for the Additive License

The theorem $\mathsf{UnaryShift}$ supplies the commutativity field of the additive naming certificate that licenses $\mathsf{Add}^{\uparrow}$ over the conservative name $\mathsf{UnaryContMonoid}^{\uparrow}$. This section decomposes its proof into smaller lemmas.

Lemma 9200.10 ($\mathsf{E1}$-congruence)**.** *If* $\mathsf{hsame}(h, k)$*, then*

$$\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k)).$$

Proof. This is the $\mathsf{E1}$-constructor rule for hsame. □

Lean4 ✓ BEDC.FKernel.Hist.hsame e1 congr
↪ variant BEDC.FKernel.Hist.eone cong

Lemma 9200.11 (Unary right-shift base)**.** *If* $\mathsf{Cont}(\mathsf{e}, \mathsf{E1}(h), r)$*, then there exists a history* v *such that*

$$\mathsf{Cont}(\mathsf{e}, h, v) \quad \textit{and} \quad \mathsf{hsame}(r, \mathsf{E1}(v)).$$

Lean4 ✓ BEDC.FKernel.Unary.unary shift base

Proof. By the left-unit law for Cont, the first premise yields $\mathsf{hsame}(r, \mathsf{E1}(h))$. Again by the left-unit law, $\mathsf{Cont}(\mathsf{e}, h, h)$. Take $v = h$. □

Lemma 9200.12 (Unary right-shift step)**.** *Assume the right-shift statement holds for a unary history k_0. If*

$$\mathsf{Unary}(k_0), \qquad \mathsf{Cont}(\mathsf{E1}(k_0), \mathsf{E1}(h), r'),$$

then there exists a history v such that

$$\mathsf{Cont}(\mathsf{E1}(k_0), h, v) \quad \textit{and} \quad \mathsf{hsame}(r', \mathsf{E1}(v)).$$

Lean4 ✓ `BEDC.FKernel.Unary.unary shift step`

↪ variant `BEDC.FKernel.Unary.unary shift step with unary result`

↪ variant `BEDC.FKernel.Unary.unary shift step with unique witness`

↪ variant `BEDC.FKernel.Unary.unary shift step closed result pair`

↪ variant `BEDC.FKernel.Unary.unary shift step closed unique witness`

Proof. Invert the continuation premise. The only active constructor for a left source headed by $\mathsf{E1}$ exposes a history u with

$$\mathsf{Cont}(k_0, \mathsf{E1}(h), u) \quad \text{and} \quad \mathsf{hsame}(r', \mathsf{E1}(u)).$$

By the induction hypothesis applied to k_0, there exists v_0 such that

$$\mathsf{Cont}(k_0, h, v_0) \quad \text{and} \quad \mathsf{hsame}(u, \mathsf{E1}(v_0)).$$

The continuation step gives

$$\mathsf{Cont}(\mathsf{E1}(k_0), h, \mathsf{E1}(v_0)).$$

Let $v = \mathsf{E1}(v_0)$. By Lemma 9200.10, $\mathsf{hsame}(u, v)$ gives $\mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(v))$. Together with $\mathsf{hsame}(r', \mathsf{E1}(u))$ and transitivity, this yields $\mathsf{hsame}(r', \mathsf{E1}(v))$. □

Theorem 9200.13 (Unary right-shift)**.** *Assume* $\mathsf{Unary}(k)$ *and* $\mathsf{Cont}(k, \mathsf{E1}(h), r')$*. Then there exists a history v such that*

$$\mathsf{Cont}(k, h, v) \quad \textit{and} \quad \mathsf{hsame}(r', \mathsf{E1}(v)).$$

Lean4 ✓ `BEDC.FKernel.Unary.unary shift`

↪ variant `BEDC.FKernel.Unary.unary shift result shape`

↪ variant `BEDC.FKernel.Unary.unary shift witness with result shape`

↪ variant `BEDC.FKernel.Unary.unary shift induction spine`

↪ variant `BEDC.FKernel.Unary.unary shift unique witness`

↪ variant `BEDC.FKernel.Unary.unary shift unique closed witness`

↪ variant `BEDC.FKernel.Unary.unary shift witness closed unique result`

Proof. Induct on the derivation of $\mathsf{Unary}(k)$. The empty case is Lemma 9200.11. The step case is Lemma 9200.12 applied to the induction hypothesis. □

Theorem 9200.14 (Unary continuation commutativity)**.** *If*

$$\mathsf{Unary}(h), \quad \mathsf{Unary}(k), \quad \mathsf{Cont}(h, k, r), \quad \mathsf{Cont}(k, h, r'),$$

then

$$\mathsf{hsame}(r, r').$$

Lean4 ✓ `BEDC.FKernel.Unary.unary commutativity refined`
↪ variant `BEDC.FKernel.Unary.unary continuation commutativity`
↪ variant `BEDC.FKernel.Unary.unary commutativity refined with result closure`

Proof. Induct on the derivation of $\mathsf{Unary}(h)$. The empty case follows from the continuation unit laws. In the step case, write the left source as $\mathsf{E1}(h_0)$. Invert $\mathsf{Cont}(\mathsf{E1}(h_0), k, r)$, obtaining a history u such that $\mathsf{Cont}(h_0, k, u)$ and $\mathsf{hsame}(r, \mathsf{E1}(u))$. Apply Theorem 9200.13 to $\mathsf{Cont}(k, \mathsf{E1}(h_0), r')$, obtaining v with $\mathsf{Cont}(k, h_0, v)$ and $\mathsf{hsame}(r', \mathsf{E1}(v))$. The induction hypothesis gives $\mathsf{hsame}(u, v)$. Lemma 9200.10 and transitivity of hsame yield $\mathsf{hsame}(r, r')$. □

Corollary 9200.15 (Activation of the additive name)**.** *Theorem 9200.14 is included in the stability field of the additive naming certificate, so the unary continuation interface is licensed to use the name* $\mathsf{Add}^{\uparrow}$.

Lean4 ✓ `BEDC.FKernel.Unary.add up name certificate`
↪ variant `BEDC.FKernel.Unary.unary add activation`

Proof. The monoid-like fields are unit, associativity, and determinacy of unary continuation. Theorem 9200.14 supplies the commutativity field required by the additive name. □

Theorem 9200.16 (Two-element unital continuations are swapped-commutative)**.** *Let C be an equality-classified carrier, let $e, u \in C$, and let $\Gamma(a, b, r)$ be a deterministic total continuation relation on C. Assume e is a two-sided unit:*

$$\Gamma(e, x, x) \qquad \textit{and} \qquad \Gamma(x, e, x)$$

for every $x \in C$. Assume also that every $x \in C$ is equal to e or to u. If $a, b, r, s \in C$, $\Gamma(a, b, r)$, and $\Gamma(b, a, s)$, then $r = s$. Consequently a finite equality-classified deterministic total unital continuation that fails the swapped-source commutativity field has at least three carrier elements.

Proof. The field under test is the order-swap result comparison isolated by Definition 9200.5; Definition 29.9 records that additive comparison is made at the result-classifier level. In the equality-classified case, this classifier is ordinary equality. The available unital and deterministic roles are the abstract counterparts of the roles recorded for continuation in Definition 4.38 and Definition 21.7, and packaged among the stability data in Definition 29.10.

Use the two-element cover for a and b. If $a = e$, then $\Gamma(a, b, r)$ is $\Gamma(e, b, r)$. The left-unit instance $\Gamma(e, b, b)$ and determinacy at the ordered pair (e, b) give $r = b$. The swapped witness is $\Gamma(b, e, s)$; the right-unit instance $\Gamma(b, e, b)$ and determinacy at (b, e) give $s = b$. Hence $r = s$. The case $b = e$ is symmetric, using the right unit on $\Gamma(a, e, r)$ and the left unit on $\Gamma(e, a, s)$.

It remains to consider the case $a = u$ and $b = u$. Then both displayed continuation witnesses have the same ordered source (u, u):

$$\Gamma(u, u, r) \qquad \text{and} \qquad \Gamma(u, u, s).$$

Determinacy gives $r = s$. Totality supplies such result witnesses for every ordered pair, so the comparison holds for the full swapped-source field. If a finite equality-classified unital continuation were a countermodel on at most two carrier elements, the preceding case split would supply the missing swapped-source equality, a contradiction. Hence a finite equality countermodel must have at least three carrier elements. □

Theorem 9200.17 (Three-point minimality of swapped-source additive obstruction)**.** *Let C be a finite equality-classified carrier equipped with a total deterministic continuation relation Γ and a two-sided unit $e \in C$. Assume the additive result classifier is equality, as in Definition 29.9. If the swapped-source additive commutativity field fails, so that there exist $a, b, r, s \in C$ with*

$$\Gamma(a, b, r), \qquad \Gamma(b, a, s), \qquad r \neq s,$$

then C has at least three equality classes, hence at least three carrier elements. The bound is sharp: the three-element equality-classified carrier of Definition 29.17 satisfies the carrier NameCert*, closure, total deterministic graph behavior, two-sided unit, associativity, same-source determinacy, and the source-to-pattern and source-to-ledger fields of the additive naming template, while failing the swapped-source additive commutativity field.*

Proof. Assume, toward contradiction, that C has at most two equality classes. Since e is the unit, choose an element $u \in C$ such that every carrier element is equal either to e or to u; in the one-class case take $u = e$. The hypotheses are exactly the equality-classified deterministic total unital setup covered by Definition 9200.16. Applying that theorem to the witnesses

$$\Gamma(a, b, r) \qquad \text{and} \qquad \Gamma(b, a, s)$$

gives $r = s$, contradicting the assumed failure $r \neq s$. Therefore any equality-classified finite countermodel to the swapped-source additive commutativity field must have at least three equality classes.

It remains to check sharpness. In Definition 29.17, the carrier is $\{e, p, q\}$, the classifier is equality, and the continuation relation is the graph of the displayed operation table. That theorem verifies the carrier certificate, closure, unit, associativity, same-source determinacy, and the source-to-pattern and source-to-ledger fields. The same table gives

$$\Gamma(p, q, p) \qquad \text{and} \qquad \Gamma(q, p, q),$$

and equality classification does not identify p and q. Hence the swapped-source additive commutativity field fails on a three-element carrier. Combining the lower bound with this witness proves that three points are minimal for this obstruction. □

Chapter 9201

Lean Scaffold Contract

9201.1 Lean-Style Scaffold Contract

This chapter records the base-reflection scaffold file in the Lean repository. The file is a checked theorem-shape scaffold: abstract carriers and relations are represented as fields of a setup structure, the base relation is an inductive definition, and the local reflection theorems are proved in a form that future implementations can instantiate with concrete objects.

9201.2 Scaffold excerpt

```
structure ExactGlobalizeBase (s : BaseReflectionSetup) (P : s.Pi) (D : s.Domain) : Prop where
  coverage : forall h, s.InDom D h -> exists p, s.InGapSig P D p h
  soundness : forall h k p q,
    s.InGapSig P D p h -> s.InGapSig P D q k ->
    GeneratedSameSig s P h k -> PsameBase s P p q
  completeness : forall h k p q,
    s.InGapSig P D p h -> s.InGapSig P D q k ->
    PsameBase s P p q -> Nonempty (GeneratedSameSig s P h k)

theorem ExactGlobalizeBase.classify_iff
  (ex : ExactGlobalizeBase s P D)
  (hp : s.InGapSig P D p h) (hq : s.InGapSig P D q k) :
  PsameBase s P p q <-> Nonempty (GeneratedSameSig s P h k) := by
  constructor
  - intro hbase
    exact ex.completeness h k p q hp hq hbase
  - intro hgen
    cases hgen with
    | intro hsig =>
        exact ex.soundness h k p q hp hq hsig
```

9201.3 Interpretation

The scaffold separates three layers:

1. **abstract carriers**: histories, signatures, packages, domains, bundles, and evidence, packaged by `BaseReflectionSetup`;
2. **relational object layer**: setup fields `TokIntro`, `hsame`, `SigGen`, and `InGapSig`;
3. **checked-shape target**: `PsameBase`, `PackageReflection.base`, and `ExactGlobalizeBase.classify_iff`.

Principle 9201.1 (No scaffold laundering)**.** *A scaffold theorem may be cited only as a theorem-shape target. It may be cited as a verified BEDC theorem only when a build log records a successful checker run and discloses all axioms or postulates.*

Lean4 ✓ `BEDC.BaseReflection.no scaffold laundering coverage soundness`

9201.4 Lean Verification Status

The five base-reflection targets are rendered here for at-a-glance reading. Each line is generated by the `\leanchecked` family of macros (see `preamble.tex`); the underscore in Lean identifier names is rendered as a thin space rather than a literal underscore. The audit `lean4/scripts/bedc_ci.py audit` enforces that each target name resolves to a real declaration in `lean4/BEDC/`.

Lean4 ✓ `BEDC.BaseReflection.PackageReflection base`

↪ variant `BEDC.BaseReflection.PsameBase inversion`

↪ variant `BEDC.BaseReflection.ExactGlobalizeBase classify iff`

Lean4 (definition) `BEDC.BaseReflection.GeneratedSameSig`

Part VIII

Implementation-Facing Contract

Chapter 9202

Formalization Implementation Contract

9202.1 Layer separation

The implementation has two layers:

1. Host layer: the proof assistant's type theory, inductive types, equality, and functions used to encode BEDC.
2. BEDC layer: the internal relations exposed by the library: `MSame`, `HSame`, `Ext`, `Cont`, `Ask`, `SigRel`, `SameSig`, `InGap`, and certificate records.

Host equality may be used to implement constructors and proofs, but it must not be advertised as BEDC internal equality.

9202.2 Module targets

Recommended module layout:

```
BEDC/FKernel/Mark.lean
BEDC/FKernel/Hist.lean
BEDC/FKernel/Ext.lean
BEDC/FKernel/Cont.lean
BEDC/FKernel/Ask.lean
BEDC/FKernel/Bundle.lean
BEDC/FKernel/Sig.lean
BEDC/FKernel/Package.lean
BEDC/FKernel/Gap.lean
BEDC/FKernel/NameCert.lean
BEDC/FKernel/Unary.lean
```

9202.3 Public theorem names

First target theorem names:

```
msame_refl
```

```
msame_symm
msame_trans
not_msame_b0_b1
not_msame_b1_b0
hsame_refl
hsame_symm
hsame_trans
not_hsame_emp_e0
not_hsame_emp_e1
not_hsame_e0_e1
ext_deterministic
ext_respects_sameness
cont_deterministic
cont_left_unit
cont_right_unit
cont_assoc_exists
sig_deterministic
sameSig_refl
sameSig_symm
sameSig_trans
gap_coverage
gap_separation
compGap_coverage
```

9202.4 Implementation choices

The first sprint should implement a finite evidence-erased signature layer. That is, use

```
SigRel Ask bundle history answerHistory
```

without an explicit evidence ledger. Once the finite kernel is checked, add evidence aggregation as a refinement. This avoids blocking signature determinacy on unresolved ledger aggregation details.

Probe bundles should be internally generated by constructors, not exposed publicly as host lists.

Probe behavior should be represented as a relation:

```
Ask :  ProbeName -> BHist -> BMark -> Evidence -> Prop.
```

A deterministic ask policy should be a structure containing the theorem that two ask events for the same probe and history emit internally same marks.

9202.5 Proof-sprint order

The recommended order:

1. prove all `MSame` theorems;
2. prove all `HSame` theorems and no-confusion lemmas;
3. prove `Ext` determinacy and sameness-respect;

4. prove `Cont` determinacy, units, and associativity witness;
5. define bundles and signature generation;
6. prove signature determinacy under deterministic ask;
7. prove `SameSig` equivalence;
8. define package and gap policies;
9. prove gap coverage and separation;
10. prove composite gap coverage.

9202.6 Forbidden public shortcuts

Do not expose host `Nat`, host `List`, host equality, or host function spaces as BEDC primitives. They may be used internally in an implementation, but the public BEDC API must state the internal relations explicitly.

Part IX

Acceptance Discipline

Chapter 9203

Closure and Verification Discipline

This chapter records the discipline by which a mature mathematical object's claim is read. The discipline reports three independent axes. The *theoretical closure axis* records what the paper has proved about the object's NameCert, scope, exactness, ledger, stability, and bridge; it is read entirely from the manuscript, with no input from any proof assistant. The *external correspondence axis* records how the package is compared with a declared outside reference theory and scope. The *formal verification axis* records the independent audit state of the corresponding Lean targets; it strengthens reliability and reproducibility but assigns no theoretical content. A claim and its audit are separate artefacts that compose only via VerifiedGate; external correspondence is reported alongside them and does not alter the theory or verification grade.

Read against Definition 9387.10, this chapter is the positive face of the project's discipline: positive content admitted only if its closure is exhibited by paper-side witnesses, while machine verification is reported as an external audit layer that never substitutes for closure.

9203.1 Theoretical closure axis

Definition 9203.1 (Theoretical closure grades)**.** *The* theoretical closure axis *is the six-element set*

$$\mathsf{TheoryClosure} = \{\,\mathsf{seed},\ \mathsf{obligation},\ \mathsf{scopedClosed},\ \mathsf{publicClosed},\ \mathsf{bridgedClosed},\ \mathsf{maturePackageClosed}\,\}$$

ordered by $\mathsf{seed} < \mathsf{obligation} < \mathsf{scopedClosed} < \mathsf{publicClosed} < \mathsf{bridgedClosed} < \mathsf{maturePackageClosed}$.

Definition 9203.2 (Closure grade interpretations)**.** *The grades are read as follows.*

seed: *a naming seed has been declared; carrier and classifier are sketched; no exactness theorem has been settled.*

obligation: *every proof obligation needed to close* NameCert(N) *has been written down precisely; the scope of the eventual closure is fixed; the central proofs are not yet completed.*

scopedClosed: *within an explicit scope, every* NameCert *field (carrier, classifier, exactness, ledger, stability) is supplied with paper-level proof. The scope is recorded in the chapter's closurestatus block and is the manuscript's binding commitment about what has actually been reconstructed.*

publicClosed: *in addition to* scopedClosed*, the chapter exhibits the public-object surface of the standard reading: public constructors, readback, endpoint exactness, normal form, and a no-hidden-leakage statement.*

bridgedClosed: *in addition to* publicClosed*, the chapter exhibits a paper-level standard bridge per chapter 9204, including a paper-level no-host-leak argument.*

maturePackageClosed: *in addition to* bridgedClosed*, the chapter records the mature-package theorem inventory whose conclusions a host reader expects (e.g. Peano induction for* $\mathsf{Nat}^{\uparrow}$*, ordered-field laws for* $\mathsf{Rat}^{\uparrow}$*).*

Definition 9203.3 (Theory gate)**.** *For an object N and grade $c \in$* TheoryClosure*, the* theory gate at grade c *is*

$$\mathsf{TheoryGate}(N, c) \;:=\; \mathsf{NameCert}(N) \wedge \mathsf{ClosureCert}(N, c).$$

A claim that BEDC closes N at grade c is admissible only when $\mathsf{TheoryGate}(N, c)$ *holds.*

Definition 9203.4 (Closure certificate fields)**.** *The closure certificate* $\mathsf{ClosureCert}(N, c)$ *packages: a* source field *naming the BEDC layer that carries N (*Mark*,* Hist*,* Bundle*,* Pkg*, or* NameCert*); a* classifier field *witnessing the equivalence relation; an* exactness field *citing the classifier-respecting transport; a* ledger field*; a* stability field*; an explicit* scope field *naming exactly what is closed; and the chosen grade c. At grades* publicClosed *and above, the certificate additionally cites a public-object surface; at* bridgedClosed *and above, the certificate cites the chapter's* StandardBridge*.*

9203.2 External correspondence axis

Definition 9203.5 (External correspondence grades)**.** *The* external correspondence axis *is the four-element set*

$$\mathsf{ExternalCorrespondenceStatus} \;=\; \{\,\mathsf{notApplicable},\ \mathsf{referencedExternal},\ \mathsf{bridgedExternal},\ \mathsf{calibratedReconstruction}\,\}$$

ordered by notApplicable $<$ referencedExternal $<$ bridgedExternal $<$ calibratedReconstruction*.*

Definition 9203.6 (External correspondence interpretations)**.** *The grades are read as follows.*

notApplicable: *the package is stated internally, with no declared outside reference theory for comparison.*

referencedExternal: *an outside reference theory R is named, and the comparison scope S is stated.*

bridgedExternal: *the chapter states the object and main-theorem correspondences between R and the BEDC construction over S.*

calibratedReconstruction: *relative to a declared and commit-pinned outside reference theory R and a declared scope S, the package closes the theory-narrative question of constructive content over S: it supplies an internal constructive theory C over S, states the $R \leftrightarrow C$ object and main-theorem correspondences, declares coverage and exclusions, and exhausts S as the disjoint union of the BEDC reconstructed constructive kernel and the measured or refused classical boundary, with no hidden third region. This grade does not assert kernel checking, axiom purity, or footprint cleanliness; those are reported only by the formal verification and bridge-status axes.*

Principle 9203.7 (Calibrated reconstruction requires boundary exhaustion)**.** *A closurestatus block may report* `\externalcorrespondence{\calibratedReconstruction}` *only when it also supplies* `\boundaryexhaustive{<label>}` *pointing to a finite decomposition of the declared scope. Each row of that decomposition must be marked either as BEDC reconstructed constructive kernel, with local Lean targets or definitions, or as measured/refused classical boundary, with the external footprint explicitly named.*

Remark 9203.8 (External correspondence is not bridged closure)**.** *The external correspondence axis is additive. It does not rewrite* bridgedClosed *and it does not promote a package along the theoretical closure ladder. A package can have* calibratedReconstruction *for an external corpus while its theory closure and formal verification grades remain exactly the values recorded by* theoryclosure *and* formalstatus*.*

9203.3 Formal verification axis

Definition 9203.9 (Formal verification grades)**.** *The* formal verification axis *is the eight-element set*

$$\mathsf{FormalStatus} = \{\mathsf{unformalized}, \mathsf{formalTarget}, \mathsf{encodedDef}, \mathsf{scaffoldChecked}, \mathsf{theoremChecked}, \mathsf{auditClean}, \mathsf{axiomCle}$$

ordered by inclusion of audit guarantees.

Definition 9203.10 (Verification grade interpretations)**.** unformalized**:** *no Lean target.*

formalTarget**:** *a Lean theorem-shape target exists (statement only); proof is empty, sorry, or scaffold.*

encodedDef**:** *the relevant carriers and classifiers are encoded as Lean* `defs` *or* `inductives`*.*

scaffoldChecked**:** *a scaffold theorem in the sense of chapter 9201 type-checks.*

theoremChecked**:** *the canonical Lean theorem corresponding to the closure certificate's distinguished proof has been checked by* `lake build`*.*

auditClean**:** *in addition, every* `\leanchecked`*,* `\leanvariant`*,* `\leanstmt`*,* `\leandef` *marker on the certificate's targets resolves to a real declaration (* `lean4/scripts/bedc_ci.py audit` *clean).*

axiomClean**:** *in addition, the transitive axiom audit reports no* `Classical.choice`*,* `Quot.sound`*, or* `propext` *dependency for the cited targets (* `tools/check-axioms.py` *and* `bedc_ci.py axiom-purity -strict` *clean).*

bridgeChecked**:** *when the certificate is at* bridgedClosed*, the standard-bridge no-host-leak field has been machine-checked against the chosen host language.*

Principle 9203.11 (Verification is not a closure formation rule)**.** *Machine verification is an audit layer over a stated theoretical closure claim. It strengthens reliability and reproducibility; it does not assign any closure grade by itself. A theory may be closed without being machine-checked, and a checked target does not by itself close any object.*

9203.4 The verified gate

Definition 9203.12 (Verified gate)**.** *For an object N, a closure grade $c \in$* TheoryClosure*, and a verification grade $v \in$* FormalStatus*, the* verified gate *is*

$$\mathsf{VerifiedGate}(N, c, v) \;:=\; \mathsf{TheoryGate}(N, c) \wedge \mathsf{FormalStatus}(N, v).$$

A claim that BEDC exports *N is admissible only when* TheoryGate(N, c) *holds for some c. The export's* verification *is reported separately by* FormalStatus(N, v)*.*

Theorem 9203.13 (Theory gate is monotone in c)**.** *For grades $c, c' \in \mathsf{TheoryClosure}$ with $c' \leq c$, $\mathsf{TheoryGate}(N, c) \to \mathsf{TheoryGate}(N, c')$.*

Sketch. The $\mathsf{NameCert}$ conjunct is independent of the grade. Each higher-grade $\mathsf{ClosureCert}(N, c)$ exhibits the lower-grade certificate as substructure (Definition 9203.2); deleting the additional fields yields $\mathsf{ClosureCert}(N, c')$. □

Theorem 9203.14 (Formal status is monotone in v)**.** *For grades $v, v' \in \mathsf{FormalStatus}$ with $v' \leq v$, $\mathsf{FormalStatus}(N, v) \to \mathsf{FormalStatus}(N, v')$.*

Sketch. Each higher verification grade extends the audit guarantees of the lower one (Definition 9203.10); the audit witnesses for the higher grade contain those of the lower grade. □

Theorem 9203.15 (Theory and verification axes are independent)**.** *Neither axis determines the other. A pair $(c, v) \in \mathsf{TheoryClosure} \times \mathsf{FormalStatus}$ is admissible whenever $\mathsf{TheoryGate}(N, c)$ and $\mathsf{FormalStatus}(N, v)$ both hold; no c implies any v, and no v implies any c beyond* seed.

Sketch. c depends only on paper-side witnesses; v depends only on Lean-side audit. The $\mathsf{NameCert}(N)$ predicate inside $\mathsf{TheoryGate}$ is met by the manuscript's textual certificate, never by a Lean target alone; conversely, a checked Lean target whose paper site exhibits no scope, no exactness, and no ledger only witnesses seed on the theory axis. Consequently the two axes are independent up to the trivial bound that any $v \geq \mathsf{unformalized}$ requires $c \geq \mathsf{seed}$ for the marker discipline of chapter 9203 to bind. □

9203.5 Marker-to-verification correspondence

Definition 9203.16 (Marker assignment)**.** *The Lean status macros map to the formal verification axis only. None assigns or strengthens a closure grade.*

1. `\leandef` *on X:* $\mathsf{FormalStatus}(\,\cdot\,, \mathsf{encodedDef})$.
2. `\leanstmt` *on X:* $\mathsf{FormalStatus}(\,\cdot\,, \mathsf{formalTarget})$.
3. `\leansorryd` *on X with rationale r:* $\mathsf{FormalStatus}(\,\cdot\,, \mathsf{formalTarget})$ *regardless of paper coverage; the rationale is recorded for later upgrade.*
4. `\leanchecked` *on X at the canonical proof site, with no* `\leansorryd` *siblings on the same claim:* $\mathsf{FormalStatus}(\,\cdot\,, \mathsf{theoremChecked})$.
5. `\leanvariant` *on X: does not change the verification grade of the canonical site; it labels a wrapper or projection at the grade of its primary.*
6. *A passed* `lean4/scripts/bedc_ci.py audit` *run lifts* $\mathsf{theoremChecked}$ *to* $\mathsf{auditClean}$ *on the cited targets.*
7. *A passed* `lean4/scripts/bedc_ci.py axiom-purity -strict` *run lifts* $\mathsf{auditClean}$ *to* $\mathsf{axiomClean}$.
8. *A machine-checked no-host-leak field on* $\mathsf{StandardBridge}(N, T)$ *lifts the bridge claim to* $\mathsf{bridgeChecked}$.

Remark 9203.17 (The marker map is not a closure rule)**.** *Definition 9203.16 assigns only a verification grade. The closure grade is supplied by the chapter's closurestatus block (section 9203.6 below); the audit script never reads the closure grade and the closure registry never reads the marker family.*

Lemma 9203.18 (The audit script reflects verification, not closure)**.** *The script* `lean4/scripts/bedc_ci.py audit` *verifies that every Lean marker resolves to a real declaration. The script reports* verification grade*; closure grade is read out of the chapter's closurestatus block by* `critical_path.py` *and by reviewers, never out of the audit JSON.*

9203.6 Per-chapter closurestatus block

Every chapter that asserts an object's closure ends with one block of the form

```
\begin{closurestatus}{\<X>Up}
  \constructivestory{A closure claim is generated bottom-up by first naming the finite source
  \theoryclosure{\scopedClosure}
  \scopeclosed{<text describing exactly what the chapter closes>}
  \externalcorrespondence{\notApplicable | \referencedExternal
    | \bridgedExternal | \calibratedReconstruction}
  \reference{<external reference theory and commit, when applicable>}
  \reconstructionscope{<external comparison scope>}
  \boundaryexhaustive{<label of finite decomposition table>}
  \formalstatus{\theoremCheckedV}
  \leantarget
    {BEDC.<...>.<theorem>}
  \bridgestatus{none | paperBridge | bridgeChecked | bridgePending}
  \notclaimed{<text listing what the chapter does NOT close>}
  \upgradepath{<text naming the next closure grade and what it requires>}
\end{closurestatus}
```

The block is the manuscript's binding commitment about N. `lean4/scripts/bedc_ci.py audit` parses every block and refuses unresolved `\leantarget` targets; `lean4/scripts/critical_path.py` reads theoryclosure and formalstatus to decide whether the chapter is retired from the work queue.

Remark 9203.19 (Both axes are mandatory)**.** *A closurestatus block lacking either* theoryclosure *or* formalstatus *is rejected at audit time. A block with* formalstatus = theoremChecked *but no* leantarget *is similarly rejected. The intent is to make the two axes inseparable in the artefact even though they are independent in content.*

9203.7 Upgrade obligations

Definition 9203.20 (Closure upgrade seed → obligation)**.** *A seed-grade object upgrades to* obligation *when the paper exhibits, in writing, the precise list of proof obligations whose discharge would close* NameCert(N) *at the next grade and the explicit scope at which the closure will be claimed.*

Definition 9203.21 (Closure upgrade obligation → scopedClosed)**.** *An* obligation *object upgrades to* scopedClosed *when paper-level proofs are exhibited within the recorded scope for source membership; classifier transitivity and reflexivity; exactness against* msame/hsame/psame*; absence or explicit form of a gap-policy ledger; and stability of the constructed certificate.*

Definition 9203.22 (Closure upgrade scopedClosed → publicClosed)**.** *A* scopedClosed *certificate upgrades to* publicClosed *when the chapter exhibits the public-object surface: public constructors compatible with the BEDC carrier, a readback function, endpoint exactness, a normal-form theorem, and a no-hidden-leakage statement bracketing the public surface.*

Definition 9203.23 (Closure upgrade publicClosed → bridgedClosed)**.** *A* publicClosed *certificate upgrades to* bridgedClosed *when the chapter exhibits a paper-level standard bridge per chapter 9204, with each of its six fields supplied and the no-host-leak field argued at paper level. Machine verification of the no-host-leak field is the separate* bridgeChecked *obligation on the verification axis.*

Definition 9203.24 (Closure upgrade bridgedClosed → maturePackageClosed)**.** *A* bridgedClosed *certificate upgrades to* maturePackageClosed *when the chapter records the full mature-package theorem inventory characterising the object in the standard reference (e.g. Peano arithmetic, ordered-field axioms, vector-space theory, full determinant theory) and supplies certificates that those theorems hold in the BEDC reading.*

Definition 9203.25 (Verification upgrades)**.** *The verification axis admits its own upgrades, each entirely independent of the closure axis:*

- unformalized → formalTarget*: a Lean theorem statement is added.*
- formalTarget → encodedDef*: the carriers and classifiers are encoded as Lean* `defs`*.*
- encodedDef → scaffoldChecked*: a scaffold theorem type-checks.*
- scaffoldChecked → theoremChecked*: the canonical theorem is proved against the scaffold.*
- theoremChecked → auditClean*:* `bedc_ci.py audit` *reports zero unresolved markers for the cited targets.*
- auditClean → axiomClean*: the transitive axiom audit is clean.*
- *to* bridgeChecked*: the standard bridge's no-host-leak field is machine-checked.*

9203.8 Reuse, composition, and retroactive promotion

Theorem 9203.26 (Theory closure reuse)**.** *If $N_1, \ldots, N_k$ have* TheoryGate(N_i, c_i) *and M is constructed from the N_i via* ReuseCert *instances, then M admits a theory gate at grade $\min_i c_i$ conditional on M's own certificate fields being supplied.*

Theorem 9203.27 (Verification status reuse)**.** *If $N_1, \ldots, N_k$ have* FormalStatus(N_i, v_i) *and M's Lean target is constructed from theirs, then M's verification grade is $\min_i v_i$ on the cited carriers, conditional on M's own canonical theorem being checked.*

Remark 9203.28 (No retroactive promotion)**.** *A construction admitted at (c, v) does not become a construction at (c', v') with $c' > c$ or $v' > v$ merely because the inputs are later upgraded. Promoting either axis requires re-discharging the corresponding upgrade obligation on the construction itself.*

9203.9 The discipline as research-plan language

Remark 9203.29 (What the discipline is for)**.** *This chapter is not a new theorem; it is a vocabulary for stating the present closure and verification status of every claim in the manuscript. Its role is to replace the implicit reading—that an object "exists" in BEDC if it appears anywhere—with the explicit reading that an object* is closed at grade c *only when* TheoryGate(N, c) *holds, and* is verified at grade v *only when* FormalStatus(N, v) *holds. The chapters that follow use this vocabulary to state the bridge protocol (chapter 9204), the cannot-claim registry (chapter 9205), and the closure reading convention (chapter 9206).*

Remark 9203.30 (The discipline is not a kernel proof obligation)**.** *The proof obligations of chapter 9195 and its siblings are obligations on the kernel: they constrain what theorems must hold for the kernel to be coherent. The closure-and-verification discipline is an obligation on the manuscript: it constrains what claims may be made about objects exported from the kernel, and how those claims relate to their independent audit. The two disciplines compose: a kernel theorem that fails its proof obligation cannot support any object's* scopedClosed *or above, while a closurestatus block that misreports either axis is refused at audit time even if every kernel theorem stands.*

Chapter 9204

Standard Bridge Protocol

The closure-and-verification discipline of chapter 9203 ends, on the closure axis, at publicClosed unless a standard bridge to a host mathematical formalism is exhibited at the paper level. This chapter records the shape such a bridge must take, the discipline that prevents it from leaking host-language structure into the BEDC kernel, and the quotient and closure obligations several downstream objects ($\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, real-valued completions, set-like and type-like reflections) must discharge before they can present themselves at bridgedClosed. The bridge is a conservative extension, not a fusion: it lets a host reader recognise the BEDC object, and it lets BEDC accept host theorems via a typed translation, without admitting host primitives as BEDC inputs.

The bridge is a paper-level construction: exhibiting all six fields advances the closure axis from publicClosed to bridgedClosed. Machine verification of the bridge's no-host-leak field is the independent bridgeChecked point on the verification axis; a paper bridge can stand at bridgedClosed before its no-host-leak field is machine-checked, and a machine-checked no-host-leak field cannot retroactively close a chapter that does not exhibit the bridge in writing.

9204.1 The six fields of a standard bridge

Definition 9204.1 (Standard bridge to a host theory)**.** *A* standard bridge *for an accepted BEDC object N to a host formal theory T is a tuple*

$$\mathsf{StandardBridge}(N,T) \;=\; (\mathsf{encode}_{N,T},\ \mathsf{decode}_{N,T},\ \mathsf{decodeEncode}_{N,T},\ \mathsf{encodeDecode}_{N,T},\ \mathsf{theoremPreservation}_{N,T},\ \mathsf{no}$$

The six fields are described in Definition 9204.2, Definition 9204.2, Definition 9204.3, Definition 9204.4, and Definition 9204.5 below.

Definition 9204.2 (Encode and decode functions)**.** *The* encode field *is a function* $\mathsf{encode}_{N,T}\colon T(N) \to \mathrm{Carrier}(N)$ *from the host's representative type to the BEDC carrier of N. The* decode field *is a function* $\mathsf{decode}_{N,T}\colon \mathrm{Carrier}(N) \to T(N)$ *in the reverse direction. Both functions are required to be classifier-respecting on the BEDC side: identifications by N's classifier translate to identifications recognised by T.*

Definition 9204.3 (Round-trip witness fields)**.** *The* decode-encode field *is a witness* $\mathsf{decodeEncode}_{N,T}\colon \forall x \in T(N).\ \mathsf{decode}_{N,T}(\mathsf{encode}_{N,T}(x)) \sim_T x$*, where $\sim_T$ is the host's recognised equality on $T(N)$. The* encode-decode field *is a witness* $\mathsf{encodeDecode}_{N,T}\colon \forall y \in \mathrm{Carrier}(N).\ \mathsf{encode}_{N,T}(\mathsf{decode}_{N,T}(y)) \sim_N y$*, where $\sim_N$ is the classifier of N on its carrier.*

Definition 9204.4 (Theorem-preservation field)**.** *The* theorem-preservation field *is a finite list of host-side basic theorems together with their BEDC-side counterparts and a translation: each host theorem in the list is asserted to follow, under encode and decode, from a corresponding BEDC theorem already accepted at* $\mathsf{TheoryGate}(N, c)$ *for some* $c \geq \mathsf{scopedClosed}$*. The list is finite and explicit; absence of a host theorem from the list means it is not claimed to be preserved.*

Definition 9204.5 (No-host-leak field)**.** *The* no-host-leak field *is the assertion that* $\mathsf{encode}_{N,T}$*,* $\mathsf{decode}_{N,T}$*, the round-trip witnesses, and the theorem-preservation translation refer to host primitives only inside the bridge module, never inside the BEDC kernel's public surface for* N*. Specifically, the host's* Nat*,* Quot*,* propext*, and* Classical.choice *are forbidden to appear inside any function or theorem statement that names* N *as a BEDC export.*

9204.2 The no-host-leak field as object-level axiom-purity

Theorem 9204.6 (The no-host-leak field is the object-level reading of axiom-purity)**.** *The discipline that the no-host-leak field of Definition 9204.5 imposes on* $\mathsf{StandardBridge}(N, T)$ *coincides, on the BEDC public surface for* N*, with the project-wide refusal of* Classical.choice*,* Quot.sound*, and* propext *that Definition 9387.10 reads as the project's refusal of meta-closure.*

Sketch. The no-host-leak field forbids the four host primitives from appearing in functions and theorems naming N as a BEDC export. Axiom-purity refuses the three meta-closure axioms in the transitive dependencies of any kernel theorem. Restricted to theorems that name N as a BEDC export, the two refusals quantify over the same syntactic surface: encode and decode functions, and theorems whose statement mentions the BEDC carrier of N. The host-Nat refusal is the additional clause that distinguishes the bridge discipline; it is the operational analogue of the term-stratum refusal in Definition 9387.5. □

Remark 9204.7 (Why the host's Nat is named explicitly)**.** *Host* Nat *is included in Definition 9204.5 alongside the three meta-closure axioms because* Nat *is the carrier most often re-used implicitly inside type-theoretic host languages: a function whose body mentions* Nat *for indexing has, in effect, smuggled host-level inductive structure into the BEDC public surface. The bridge field forbids this smuggling at the object level.*

Theorem 9204.8 (Bridge with no-host-leak is conservative over BEDC)**.** *If* $\mathsf{StandardBridge}(N, T)$ *is provided with the no-host-leak field discharged, then no theorem about* N*'s BEDC export becomes provable through the bridge that was not already provable in BEDC at* $\mathsf{TheoryGate}(N, c)$ *for some* $c \geq \mathsf{scopedClosed}$*.*

Sketch. The no-host-leak field excludes the four host primitives from the public surface for N. The remaining bridge content factors through the encode and decode functions, whose round-trip witnesses identify $T(N)$ and $\mathrm{Carrier}(N)$ up to the chosen classifier. A new theorem about N that the bridge supports must be the encode-decode image of a host theorem; the theorem-preservation field of Definition 9204.4 requires such an image to be paired with an existing BEDC-side counterpart. Hence no truly new BEDC theorem about N is added. □

9204.3 Quotient and closure discipline as bridge prerequisites

Definition 9204.9 (Base relation versus closure relation)**.** *A* base relation *on a BEDC carrier is one defined by a finite generating list of clauses, each clause a single-step relation between carrier*

elements. A closure relation *is the smallest relation that contains a given base relation and is closed under reflexivity, symmetry, transitivity, congruence with carrier operations, or any combination thereof named by a closure operator.*

Definition 9204.10 (Closure-level reflection obligation)**.** *For a closure relation $\widehat{R}$ over a base relation R, the* closure-level reflection obligation *is a paper-level proof that, for every theorem φ stated against $\widehat{R}$-equivalence classes, φ holds whenever its base-relation projection holds on representatives. The obligation is discharged by exhibiting an induction principle over the closure operator that lifts a base-level proof to a closure-level proof.*

Remark 9204.11 (Closure-level exactness gating)**.** *The exactness field of* $\mathsf{ClosureCert}(N, c)$ *in Definition 9203.4 demands a classifier-respecting transport at the layer the certificate is stated on. When the classifier is a closure relation, the transport must be supplied at the closure layer: a base-level transport does not suffice. Until the closure-level reflection obligation of Definition 9204.10 is discharged, the certificate is admitted at most at* $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ *on the base layer; closure-grade exactness above* $\mathsf{scopedClosed}$ *is not exported.*

Theorem 9204.12 (Closure transport composes with base transport)**.** *A closure-level transport for $\widehat{R}$ over R, together with a base-level transport for R, yields a transport on the closure quotient that respects the classifier of $\widehat{R}$ on every closure operator clause.*

Sketch. The closure-level induction of Definition 9204.10 drives the lift of the base transport to the closure layer; each closure operator step is handled by the corresponding clause of the induction. Composition of the two transports is associative because both factor through the same carrier. □

Remark 9204.13 (Bridge prerequisites for downstream objects)**.** *The integers, the rationals, real-valued completions, set-like reflections, and type-like reflections all rest on closure relations: integer equivalence over $\mathbb{N}^2$, rational equivalence over $\mathbb{Z} \times \mathbb{Z}_{\neq 0}$, Cauchy equivalence over rational sequences, set extensionality at the type level, and type identifications respectively. Each of these requires the closure-level reflection obligation of Definition 9204.10 discharged before a standard bridge to its host counterpart can be admitted. Until then the bridge is presented at* $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ *on the base relation, with closure-grade upgrade above* $\mathsf{scopedClosed}$ *deferred.*

9204.4 Conservativity in the presence of theorem preservation

Theorem 9204.14 (Theorem-preservation does not extend BEDC)**.** *A finite theorem-preservation list as in Definition 9204.4 preserves Definition 9204.8: every host theorem in the list translates, under encode and decode, to an existing BEDC theorem about N.*

Sketch. The list's defining condition is that each host entry is paired with an accepted BEDC counterpart at $\mathsf{TheoryGate}(N, c)$ for $c \geq \mathsf{scopedClosed}$. The translation under encode and decode is the round-trip witness. A bridge user who reads off a host theorem from the list and translates back to BEDC obtains a theorem already in BEDC. □

Remark 9204.15 (The list is finite by design)**.** *The theorem-preservation list of Definition 9204.4 is finite because BEDC's discipline is to admit only what has been derived. An infinite list would silently re-introduce schematic host content; a finite list with explicit entries records exactly the host theorems whose BEDC counterparts have been accepted.*

9204.5 Worked sketch: integers

Remark 9204.16 (Integers as the canonical bridge example)**.** *The integer object* $\mathsf{Int}^{\uparrow}$ *illustrates the six fields. The host* T *is the type-theoretic integer type with its standard interface. The encode field maps a host integer to its* $\mathsf{Int}^{\uparrow}$ *carrier representative built from a pair of unary histories; the decode field reads a* $\mathsf{Int}^{\uparrow}$ *carrier representative back to the host integer by collapsing the pair via host subtraction. The decode-encode round trip is host-side equality of integers, which holds by the host's own reflection theorem; the encode-decode round trip is* $\mathsf{Int}^{\uparrow}$*'s classifier on* $\mathbb{N}^2$ *pairs, which holds because* $\mathsf{Int}^{\uparrow}$*'s classifier identifies pairs differing by an additive cancellation. Theorem preservation lists the host's basic identities of integer addition and multiplication paired with their* $\mathsf{Int}^{\uparrow}$ *counterparts already at* $\mathsf{TheoryGate}(\mathsf{Int}^{\uparrow}, \mathsf{scopedClosed})$*. The no-host-leak field forbids host* Quot *from appearing inside any* $\mathsf{Int}^{\uparrow}$*-naming function: the integer object's classifier is a closure relation, but its closure-level reflection is supplied by Definition 9204.10's induction, not by host* $\mathsf{Quot.sound}$*.*

Remark 9204.17 (The sketch is not a proof)**.** *Definition 9204.16 is a paper-level sketch that records the shape of* $\mathsf{StandardBridge}(\mathsf{Int}^{\uparrow}, T_{\mathbb{Z}})$ *for orientation. Its admission as a* $\mathsf{bridgedClosed}$ *export of* $\mathsf{Int}^{\uparrow}$ *requires the closure-level reflection obligation discharged at the paper level; subsequent machine verification of the bridge fields lifts the verification axis to* $\mathsf{bridgeChecked}$ *but is independent of the* $\mathsf{bridgedClosed}$ *closure grade. None of these clauses is asserted to be discharged by this chapter.*

9204.6 Bridge prerequisites table

Remark 9204.18 (Common prerequisites for bridge-strength export)**.** *Five downstream object families share the prerequisites recorded in this chapter. For each family, the bridge-strength export requires: the closure gate at* $\mathsf{TheoryGate}(N, \mathsf{publicClosed})$*, the closure-level reflection obligation of Definition 9204.10, the six fields of Definition 9204.1, and (on the independent verification axis) the no-host-leak field at* $\mathsf{bridgeChecked}$ *if the bridge's host translation is to be machine-audited.*

- *Integer family:* $\mathsf{Int}^{\uparrow}$ *with* $T_{\mathbb{Z}}$ *over the host's integer type.*
- *Rational family:* $\mathsf{Rat}^{\uparrow}$ *with* $T_{\mathbb{Q}}$ *over the host's rational type.*
- *Real-valued completions: a real object with* $T_{\mathbb{R}}$ *over the host's real-number type, conditional on a Seal/Completion certificate.*
- *Set-like reflections: a set object with the host's* Set *type, conditional on a closure-level reflection over set extensionality.*
- *Type-like reflections: a type object with the host's universe, conditional on a closure-level reflection over type identifications.*

The current manuscript admits none of these at $\mathsf{bridgedClosed}$*; the prerequisites are recorded here to make the upgrade obligation on the closure axis explicit, separate from any verification-axis upgrade to* $\mathsf{bridgeChecked}$*.*

Remark 9204.19 (Standard bridge is per host, not absolute)**.** *The bridge is parameterised by the host theory* T*. A bridge to one host (for instance, the type-theoretic host of* `lean4/`*) does not automatically produce a bridge to a different host (a set-theoretic foundation). Each host requires its own discharge of the six fields. The protocol is uniform; the obligations are per-host.*

Chapter 9205

Cannot-Claim Registry and Failure Witnesses

The closure-and-verification discipline of chapter 9203 states what the manuscript may export on the closure axis and what its Lean targets audit on the verification axis; the standard bridge protocol of chapter 9204 states the additional discipline that bridgedClosed export requires. Both chapters describe the positive face of the project's discipline. This chapter records the negative face: the typed failure witnesses that block a closure or verification claim and the explicit cannot-claim registry that records which export claims the manuscript refuses on principle. Read together with Definition 9387.10, the cannot-claim registry plays at the export level the same role that axiom-purity plays at the kernel level: a refusal to admit an unwitnessed positive claim.

9205.1 The discipline of negative claims

Remark 9205.1 (Negative claims as part of the export surface)**.** *A research manuscript that exhibits only its positive theorems implicitly invites readers to infer the rest of mathematics is on its way. The BEDC discipline is the opposite: every claim outside the certified surface is presumed unsupported until a certificate is exhibited, and the manuscript states explicitly which downstream claims it refuses to make at the present closure grade. The cannot-claim registry of Definition 9205.6 is the explicit form of that refusal.*

Remark 9205.2 (Failure witnesses as negative duals of certificates)**.** *A certificate exhibits the structure that licenses an export. A failure witness exhibits the structure that blocks one. Both are typed: the certificate populates a fixed schema of fields, and the failure witness selects one of a fixed list of failure modes. The pairing is total: at any given moment an object either satisfies* $\mathsf{TheoryGate}(N, c)$ *and* $\mathsf{FormalStatus}(N, v)$ *at the grades it claims, in which case its paper witnesses and audit witnesses are exhibited; or it does not, in which case a failure witness names the missing field.*

9205.2 Failure witness type

Definition 9205.3 (Failure certificate)**.** *The* failure certificate $\mathsf{FailureCert}(N, c, v)$ *at the candidate closure grade* c *and verification grade* v *for an object* N *is one of the thirteen typed branches:*

1. `missing-source`*: no source-layer assignment in the sense of Definition 9203.4;*

2. `missing-classifier`: *no classifier supplied for the cited source;*
3. `classifier-not-transitive`: *classifier supplied but transitivity fails or is unproved;*
4. `exactness-fails`: *classifier does not respect* msame/hsame/psame *on the cited source;*
5. `operation-does-not-descend`: *a kernel operation fails to descend through the classifier;*
6. `gap-ledger-absent`: *the ledger field is unspecified and no explicit absence statement is supplied;*
7. `completion-not-stable`: *a completion construction fails the stability field;*
8. `scope-overstated`: *the closure block claims a grade whose scope exceeds what the chapter actually closes (e.g.* publicClosed *asserted for a singleton certificate);*
9. `standard-bridge-fails`: StandardBridge(N, T) *is incomplete in at least one of its six fields when* bridgedClosed *is claimed;*
10. `host-primitive-leakage`: *the no-host-leak field is violated, that is, host* Nat, Quot, propext, *or* Classical.choice *appears in the BEDC public surface for N;*
11. `theorem-checker-failure`: *a Lean target cited under* `\leanchecked` *is missing or fails to build (refutes* theoremChecked*);*
12. `audit-not-clean`: `lean4/scripts/bedc_ci.py audit` *reports unresolved markers on the cited targets (refutes* auditClean*);*
13. `axis-confusion`: *the chapter's prose, table, or marker discipline assigns a closure grade as a function of a verification grade (or vice versa), in violation of Definition 9203.11.*

A claim that TheoryGate(N, c) *holds is refuted by exhibiting any one of branches 1–10. A claim that* FormalStatus(N, v) *holds is refuted by branches 11–12. A claim that the discipline of chapter 9203 is honoured is refuted by branch 13.*

Theorem 9205.4 (Failure witness blocks the gate)**.** *If* FailureCert(N, c, v) *is exhibited against a claimed gate, then that claimed gate does not hold: branches 1–10 block* TheoryGate(N, c)*, branches 11–12 block* FormalStatus(N, v)*, and branch 13 blocks the two-axis discipline that would license* VerifiedGate(N, c, v)*.*

Sketch. Each branch of Definition 9205.3 negates a clause of Definition 9203.4, Definition 9204.1, Definition 9203.16, or Definition 9203.11. A field that the claimed gate requires is absent, invalid, or assigned on the wrong axis by exhibiting the branch; therefore the corresponding conjunction fails. □

Remark 9205.5 (Why thirteen branches and not fewer)**.** *The thirteen branches of Definition 9205.3 are not a partition into incomparable failure modes; some branches are special cases of others read at different closure or verification grades. The list is finite and explicit so that a given audit pass can refer to a single branch when refuting an export claim, and so that grade decisions can be local: a failure cited under one branch is the obligation that needs discharging before a stronger gate is claimed.*

9205.3 Cannot-claim registry

Definition 9205.6 (Cannot-claim entry)**.** *A* cannot-claim entry *is a quadruple*

$$(\text{ClaimText},\ \text{Status},\ \text{WhyNotClaimable},\ \text{RequiredUpgrade})$$

where ClaimText *is a sentence the manuscript refuses to assert at the present time;* Status *is one of* {`forbidden`, `deferred`, `partial`}*;* WhyNotClaimable *cites either an open obligation or a structural reason; and* RequiredUpgrade *names the certificate or bridge whose discharge would license the claim.*

Definition 9205.7 (Cannot-claim registry)**.** *The* cannot-claim registry *is the finite list of cannot-claim entries that the manuscript currently records. The registry is part of the manuscript's export surface in the sense of Definition 9205.1: removing an entry requires the corresponding upgrade discharged, and adding an entry records a new refusal.*

Remark 9205.8 (Anchor entries of the registry)**.** *The four anchor entries below name claims that the manuscript explicitly refuses at the present closure and verification grades. They are the registry: principled refusals that this chapter commits to fixed phrasing. A claim's removal from the registry, or its addition, is a manuscript edit at the cadence of editorial review, not an automated state report.*

- ***Anchor 1.*** ClaimText*: "BEDC derives all mathematics."* Status*:* ***`forbidden`****.* WhyNotClaimable*: the kernel admits only finite generative content; downstream theories enter through naming certificates and standard bridges, not as primitives.* RequiredUpgrade*: there is no upgrade. The claim is refused on principle, not deferred.*
- ***Anchor 2.*** ClaimText*: "*$\mathsf{Nat}^{\uparrow}$ *is the full Peano natural numbers."* Status*:* ***`partial`****.* WhyNotClaimable*:* $\mathsf{Nat}^{\uparrow}$ *records unary repetition histories with* $\mathsf{TheoryGate}(\mathsf{Nat}^{\uparrow}, \mathsf{scopedClosed})$ *on the canonical NameCert scope; full Peano content (induction principle as a closed theorem schema, primitive recursion with computation rules, totality of arithmetic operations) requires the upgrade* $\mathsf{TheoryGate}(\mathsf{Nat}^{\uparrow}, \mathsf{maturePackageClosed})$*.* RequiredUpgrade*:* $\mathsf{Nat}^{\uparrow}$*'s arithmetic test suite (induction, recursion, addition associativity and commutativity, multiplication distributivity, order totality) closed at* maturePackageClosed *on the closure axis. The verification axis grade does not by itself close any further Peano content.*
- ***Anchor 3.*** ClaimText*: "A real-valued completion is exported."* Status*:* ***`deferred`****.* WhyNotClaimable*: the seal/completion module of chapter 11 is a thin interface; Cauchy equivalence, completion existence, uniqueness of limits, and the field-operation descent are not discharged.* RequiredUpgrade*: a Seal/Completion certificate satisfying Definition 9204.10 on Cauchy equivalence, with field operations descending to the completion.*
- ***Anchor 4.*** ClaimText*: "The Lean scaffold is verified end-to-end."* Status*:* ***`forbidden`****.* WhyNotClaimable*: the scaffold of chapter 9201 is a statement-level surface; verification means each scaffold target stands at* theoremChecked *in* ***`lean4/BEDC/`*** *with a successful build report and at* auditClean *on* ***`bedc_ci.py audit`****.* RequiredUpgrade*: a complete checker report from* ***`lake build`*** *together with the audit of* ***`lean4/scripts/bedc_ci.py`*** *and the axiom-purity gate, all exiting clean. This upgrade strengthens the verification axis only; it does not by itself close any object's theory.*

Remark 9205.9 (The registry is finite by design)**.** *The cannot-claim registry of Definition 9205.7 is a finite list because the chapter's discipline is to commit only to refusals it can phrase precisely. A status report enumerating every object that does not currently meet* $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ *would mistake a derivation queue for a refusal: an object that is presently at* seed *for lack of a classifier is not a refused claim but an unfinished one. The registry therefore lists principled refusals (Anchor 1, Anchor 4) and bounded deferred claims with explicit upgrade obligations (Anchor 2, Anchor 3); per-object derivation status is read from chapter closurestatus blocks and the audit script of Definition 9203.18, not from the registry.*

9205.4 Closure reading convention

Remark 9205.10 (Reading a closure claim)**.** *A theoretical closure claim and a formal verification claim are distinct. A line of the form* $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ *means that the paper supplies the source, classifier, exactness, ledger, and stability witnesses needed to close* N *within the stated scope. A line of the form* $\mathsf{FormalStatus}(N, \mathsf{theoremChecked})$ *means that the cited Lean target has been machine-checked. The second claim audits the first; it does not create the first. A theory may be closed without being machine-checked. Conversely, a checked scaffold or checked definition does not by itself close the corresponding mathematical object.*

9205.5 Anti-confusion register

Remark 9205.11 (Confusions explicitly refused)**.** *The following readings are refused. Citing any one of them is a discipline violation that the audit catches via the* `axis-confusion` *branch of Definition 9205.3.*

1. *"BEDC has re-derived all of mature mathematics."*
2. "$\mathsf{Real}^{\uparrow}$ *is complete."*
3. "$\mathsf{S}^{1\uparrow}$ *is complete."*
4. "$\mathsf{Fold}^{\uparrow}$ *is complete."*
5. "$\mathsf{Nat}^{\uparrow}$ *at* $\mathsf{scopedClosed}$ *equals full natural-number theory."*
6. "$\mathsf{Add}^{\uparrow}$ *at* $\mathsf{scopedClosed}$ *equals full addition."*
7. *"Paper certificates equal machine-checked certificates."*
8. *"Lean scaffolds equal verified implementations."*
9. *"External standard models replace BEDC internal proof obligations."*
10. *"Host equality, host* Nat*, or host functions are part of the BEDC public surface."*
11. *"Machine verification is required for theoretical closure."*
12. *"Without* $\mathsf{theoremChecked}$ *no closure grade above* seed *may be claimed."*
13. *"A checked theorem-shape scaffold closes the corresponding object theory."*
14. *"A scoped certificate closure is a full traditional-object reconstruction."*
15. *"A bridge is closed merely because the BEDC-side certificate is checked."*

9205.6 Failure witness as audit input

Remark 9205.12 (Audit policy when a failure is exhibited)**.** *When a failure certificate is exhibited for an object* N *presently at* $\mathsf{TheoryGate}(N, c)$ *with* $c \geq \mathsf{scopedClosed}$*, or at* $\mathsf{FormalStatus}(N, v)$ *with* $v \geq \mathsf{theoremChecked}$*, the closurestatus block of the chapter is downgraded by the discipline of Definition 9203.16: a* `\leanchecked` *becomes a* `\leanstmt` *if the source still survives but exactness no longer holds, or a* `\leansorryd` *if the proof has been retracted; the closure grade is lowered to the highest grade still licensed by the surviving witnesses. The audit script of Definition 9203.18 reflects only the verification axis; the closure grade is reflected by the chapter's closurestatus block.*

Remark 9205.13 (The cannot-claim registry is not a roadmap)**.** *A roadmap states what will be claimed when the present blockers are cleared. The cannot-claim registry of Definition 9205.7 states what the manuscript currently refuses. The two artefacts are not the same: a refusal that is principled (Anchor 1) does not become a future claim regardless of progress; a refusal that is deferred (Anchor 3) becomes a future claim only when the named upgrade is discharged. The registry is the negative face of the manuscript, not the schedule of what the manuscript will eventually admit.*

Remark 9205.14 (Composition with the axiom-purity gate)**.** *The axiom-purity gate of Definition 9387.10 refuses kernel theorems whose transitive dependencies include the three meta-closure axioms. The cannot-claim registry refuses export claims whose certificates or bridges are absent. The two refusals operate at different layers and compose: a kernel theorem permitted by axiom-purity may still fail to license an export claim if its standard bridge is incomplete, and an export claim refused by the registry does not become licit by the kernel passing its purity gate.*

Chapter 9206

Closure Reading Convention

This short chapter records the convention by which a reader should read every closurestatus block in the manuscript.

9206.1 Read both axes, separately

A line of the form $\mathsf{TheoryGate}(N, c)$ records what the paper has closed about N at grade c. A line of the form $\mathsf{FormalStatus}(N, v)$ records the audit state of the cited Lean target at grade v. The two are independent: neither line implies the other, and the conjunction $\mathsf{VerifiedGate}(N, c, v)$ records both at once without merging them.

9206.2 Read closure as a paper claim

The closure axis is read entirely from the manuscript. A reader interrogating a closure claim should ask whether the chapter exhibits, within the stated scope, a source, a classifier, an exactness theorem, a ledger or absence statement, and a stability witness. None of these inquiries asks anything about a Lean file.

9206.3 Read verification as an audit claim

The verification axis is read entirely from the audit. A reader interrogating a verification claim should ask whether the cited Lean target exists, whether `lake build` accepts it, whether `bedc_ci.py audit` resolves it, and whether `check-axioms.py` reports the transitive dependencies clean. None of these inquiries asks anything about whether the paper's mathematical reading is closed.

9206.4 Per-object orientation table

The following table orients a reader who lands on a chapter and wants to know its present axes at a glance. The table is non-normative: each chapter's closurestatus block is the binding source of truth.

Object	Theory closure	Scope	Formal status
$\mathsf{Nat}^{\uparrow}$	scopedClosed	unary history certificate	theoremChecked

Object	Theory closure	Scope	Formal status
Add$^{\uparrow}$	scopedClosed	unary continuation certificate	theoremChecked
Int$^{\uparrow}$	scopedClosed	history pair certificate	theoremChecked
Rat$^{\uparrow}$	scopedClosed	history-pair semantic certificate	theoremChecked
Bool$^{\uparrow}$	scopedClosed	boolean history semantic certificate	theoremChecked
Option$^{\uparrow}$	scopedClosed	tagged-option semantic certificate	theoremChecked
Prod$^{\uparrow}$	scopedClosed	product history semantic certificate	theoremChecked
Sum$^{\uparrow}$	scopedClosed	sum history semantic certificate	theoremChecked
List$^{\uparrow}$	scopedClosed	list spine bridge classifier certificate	theoremChecked
Complex$^{\uparrow}$	scopedClosed	history pair semantic certificate	theoremChecked
Monoid$^{\uparrow}$	scopedClosed	monoid history semantic certificate	theoremChecked
Group$^{\uparrow}$	scopedClosed	group stability certificate	theoremChecked
AbGroup$^{\uparrow}$	scopedClosed	singleton empty-history certificate	theoremChecked
CommRing$^{\uparrow}$	scopedClosed	singleton empty-history certificate	theoremChecked
Field$^{\uparrow}$	scopedClosed	FieldApartZero / phase-exit certificate	theoremChecked
Module$^{\uparrow}$	scopedClosed	singleton smul image certificate	theoremChecked
VecSpace$^{\uparrow}$	scopedClosed	singleton empty-history vector-space certificate	theoremChecked
LinMap$^{\uparrow}$	scopedClosed	singleton linear-map laws	theoremChecked
Met$^{\uparrow}$	scopedClosed	empty-history distance certificate	theoremChecked
Cat$^{\uparrow}$	scopedClosed	hom-carrier composition certificate	theoremChecked
Functor$^{\uparrow}$	scopedClosed	prefix functor carrier certificate	theoremChecked
NatTrans$^{\uparrow}$	scopedClosed	prefix-component classifier certificate	theoremChecked
Yoneda$^{\uparrow}$	scopedClosed	empty-component-family certificate	theoremChecked
Subgroup$^{\uparrow}$	scopedClosed	subgroup centralizer certificate	theoremChecked
QuotientGroup$^{\uparrow}$	scopedClosed	centralizer-normalizer certificate	theoremChecked

Object	Theory closure	Scope	Formal status
Determinant$^{\uparrow}$	scopedClosed	singleton determinant certificate	theoremChecked
Homology$^{\uparrow}$	scopedClosed	singleton cycle certificate	theoremChecked
Cohomology$^{\uparrow}$	scopedClosed	singleton cocycle certificate	theoremChecked
Prime$^{\uparrow}$	scopedClosed	prime semantic certificate	theoremChecked
DyadicRatCore	maturePackageClosed	dyadic obligation faces and finite ledger	axiomClean
StreamName$^{\uparrow}$	maturePackageClosed	rational stream-name public certificate	bridgeChecked
RegSeqRat	maturePackageClosed	rational-stream certificate spine	axiomClean
Real$^{\uparrow}$	maturePackageClosed	public rational-stream export	bridgeChecked

Remark 9206.1 (How to read the closure entries)**.** *Most present BEDC entries close a precise, paper-recorded scope, while a chapter that exhibits a public-object surface may carry* publicClosed*. The table above is only a derived orientation aid; the binding closure grade is read from the object's own closurestatus block. Grades above* scopedClosed *require the corresponding public or bridge surface described in chapter 9204, and each closurestatus block names its own scope and upgrade path explicitly.*

Remark 9206.2 (How to read the verification entries)**.** *The verification entries are likewise a derived reading of the current closurestatus blocks. A* theoremChecked *entry means the block names a Lean target that resolves under* `lean4/BEDC/` *and that* `lake build` *accepts; an* unformalized *entry records that the paper boundary has not yet reached that checked Lean grade. The audit script of Definition 9203.18 is responsible for marker resolution, while the closure axis grade remains independent.*

Part X

Ground Compiler Discipline

Chapter 9207

Ground Event Flow

This chapter fixes the only formal input the BEDC compilation discipline admits. The input is not a Pkg, a NameCert, a ClosureCert, an AST, a YAML manifest, an opcode list, an arity table, or a theorem identifier; it is a finite event flow displayed from the kernel's two raw marks b0 and b1. The chapter records the discipline by which structural readings of an event flow—package roles, certificate fields, theorem fields, compiler fields, and object acceptance—arrive only through generated recognizers, never as primitive compiler inputs.

The chapter sits between chapter 9203, which fixes how a mature object is read through TheoryGate and FormalStatus, and chapter 9371, which records the abstract algebraic view of compilation as a NameCert morphism. The present chapter supplies the concrete substrate: the input language to which any such morphism must be applied is the language of event flows, and that language admits no hidden structural input.

9207.1 Display alphabet and raw events

Definition 9207.1 (Display alphabet). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.DisplayAlphabet`

The display alphabet *is*

$$\Sigma := \{0, 1\}.$$

The chapter adopts the external display convention

$$\mathsf{b0} \mapsto 0, \qquad \mathsf{b1} \mapsto 1.$$

The displayed symbols 0 *and* 1 *are presentation glyphs only; they are not natural numbers, truth values, binary numerals, or members of an external set.*

Remark 9207.2 (Display is not interpretation). *The display convention sends each raw mark to a glyph for the purpose of writing event flows on the page. It does not interpret* b0 *as the integer zero,* b1 *as the integer one, or finite glyph strings as numerals, completion data, dimensions, types, or theorems. Any such reading must be supplied by a downstream classifier together with a* NameCert*; the licensing discipline is recorded in chapter 9 and chapter 9203.*

Definition 9207.3 (Raw event). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RawEvent`

A raw event *is a finite display string* $w \in \Sigma^*$*. The set of raw events is*

$$\mathsf{RawEvent} := \Sigma^*.$$

The empty raw event is permitted and is denoted ϵ*.*

Remark 9207.4 (A raw event is a finite trace, not a theorem)**.** *A raw event is a finite displayed string of glyphs. It is not a theorem, not a proof, not a* NameCert*, not a* ClosureCert*, not a packaged object, and not a kernel judgment. Visible patterns inside a raw event—such as* 011 *or* 100*—do not by themselves license any classifier reading; in particular they do not assert the Zeckendorf-normal-form classifier of chapter 278. Such a reading is licensed only when supported by a generated source flow, pattern flow, classifier flow, stability flow, and ledger flow under the certificate discipline of Definition 9203.4.*

9207.2 Event flows and boundary information

Definition 9207.5 (Event flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.EventFlow`

An event flow *is a finite ordered sequence of raw events*

$$S \;=\; (w_0, w_1, \ldots, w_{n-1}), \qquad w_i \in \mathsf{RawEvent}.$$

The set of event flows is

$$\mathsf{EventFlow} \;:=\; (\Sigma^*)^*.$$

A representative event flow is written with a vertical bar between successive raw events, e.g.

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

which denotes $(0, 00, 000, 01, 011, 100)$*.*

Definition 9207.6 (Event boundary)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.EventBoundary`

An event boundary *is the separation between two consecutive raw events in an event flow. The event boundary is part of the generated event-flow datum and is not recoverable from the concatenation of glyphs alone. Two event flows* $(0, 00, 000)$ *and* $(00, 0000)$ *are distinct elements of* EventFlow *even though both display the concatenated glyph string* 000000 *if their boundaries are erased.*

Proposition 9207.7 (Bare bitstreams do not determine event flows)**.** *Lean4 ✓* `BEDC.GroundCompiler.EventFlow.e` `not injective`

The boundary-erasure map

$$\mathrm{erase}\colon \mathsf{EventFlow} \longrightarrow \Sigma^*,$$

which sends an event flow to the concatenation of its raw events, is not injective.

Proof. Take

$$S_1 \;=\; (0, 00, 000), \qquad S_2 \;=\; (00, 0000).$$

Both event flows erase to the same bare string 000000. As event flows they differ: their event decompositions are not equal as sequences. Hence erase identifies S_1 and S_2, contradicting injectivity. □

Corollary 9207.8 (A compiler input cannot be a bare bitstream)**.** *Lean4 ✓* `BEDC.GroundCompiler.EventFlow.inpu` `not bare bitstream`

If the formal input to the compiler were a bare bitstream $b \in \Sigma^*$*, then the event segmentation that re-establishes the underlying event flow would have to be supplied externally, contradicting the no-hidden-input discipline of Definition 9207.9. Therefore the first formal input layer is* EventFlow*, not* Σ^**.*

9207.3 No-hidden-input discipline

Principle 9207.9 (No hidden structural input). *Lean4✓ BEDC.GroundCompiler.EventFlow.structural hidden not formal*

The compiler admits no formal input other than event flows. In particular, none of the following is a formal compiler input:

$$\mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ClosureCert},\quad \text{TheoremPkg},\quad \text{ChapterPkg},\quad \text{ProofTree},$$

$$\text{AST},\quad \text{YAML},\quad \text{manifest},\quad \text{external theorem identifier},\quad \text{external object name},$$

$$\text{external type tag},\quad \text{external mode tag},\quad \text{external opcode},\quad \text{external arity table}.$$

If any such structure is to participate in compilation, it must first appear as an event-flow pattern and be recognized by a generated recognizer flow.

Definition 9207.10 (Generated recognizer). *Lean4 (definition) BEDC.GroundCompiler.EventFlow.GeneratedRecogni*

A generated recognizer *is itself an event flow:*

$$R \in \mathsf{EventFlow}.$$

A generated recognizer is not a host parser, host function, schema, or AST constructor. Its only admissible public form is a generated event flow. A generated recognizer may carry a recognition relation $\mathsf{Recognizes}(R, S)$ *for a target event flow* S*; the relation is a kernel-level judgment about a pair of event flows, not a host predicate.*

Definition 9207.11 (Recognized package flow). *Lean4 (definition) BEDC.GroundCompiler.EventFlow.RecognizedPac*

A recognized package flow *is an event flow* $S \in \mathsf{EventFlow}$ *for which there exists a generated recognizer* R *such that*

$$\mathsf{RecognizesPkg}(R, S)$$

holds. The relation $\mathsf{RecognizesPkg}$ *is a kernel-level judgment between two event flows; the package structure is a structure recognized in* S *by* R*, not an external datum supplied to the compiler.*

Proposition 9207.12 (Package structure is not a primitive compiler input). *Lean4✓ BEDC.GroundCompiler.Event not primitive*

There is no formal compiler map of the form $\mathsf{Pkg} \to \mathsf{Code}$*. The only admissible form is*

$$\mathsf{EventFlow} \longrightarrow \mathsf{Code},$$

optionally accompanied by recognition judgments such as $\mathsf{RecognizesPkg}(R, S)$*.*

Proof. Suppose to the contrary that the compiler accepts a Pkg object P as primitive input. Then P's constituent fields, package boundaries, and structural roles are supplied to the compiler before any event-flow generation has taken place. These fields constitute an external structural datum and therefore a structural hidden input in the sense of Definition 9207.21. This contradicts Definition 9207.9. Hence package structure must be recognized from a previously generated event flow rather than supplied as input. □

Remark 9207.13 (Relation to typed naming certificates). *The discipline above is a strict instance of the typed naming certificate admission discipline of chapter 9. A derived name* N *is admitted only when the source, pattern, classifier, stability, and ledger fields of* $\mathsf{NameCert}(N)$ *are supplied; an upward-arrow derived interface is licensed only by such a certificate. The compiler may later recognize a* $\mathsf{NameCert}$ *flow within an event flow, but it may not begin by accepting a* $\mathsf{NameCert}$ *object as input.*

9207.4 Recognized derived flows

Definition 9207.14 (Recognized NameCert flow). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RecognizedN`

A recognized NameCert flow *is an event flow S for which a generated recognizer R witnesses*

$$\mathsf{RecognizesNameCert}(R, S).$$

A recognized NameCert *flow is later expected to contain, in recognized order, event subflows that play the certificate roles of Definition 9203.4: a source-spec subflow, a pattern-spec subflow, a classifier-spec subflow, a stability-cert subflow, a ledger-policy subflow, and a closing seal subflow. These role assignments are not external tags supplied at the ground input layer; they are roles recognized inside S by R.*

Definition 9207.15 (Recognized theorem flow). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RecognizedThe`

A recognized theorem flow *is an event flow S for which a generated recognizer R witnesses*

$$\mathsf{RecognizesTheorem}(R, S).$$

A recognized theorem flow may be further recognized as containing event subflows for statement, dependency, proof, certificate, ledger, status, canonical site, and closing seal. None of these is a first-layer compiler input; each is a role recognized in S.

Definition 9207.16 (Recognized compiler flow). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RecognizedCo`

A recognized compiler flow *is an event flow C that satisfies*

$$\mathsf{RecognizesCompiler}(C).$$

The compiler is therefore not a host-level object in the formal surface; it is eventually represented as an event flow whose recognized roles are precisely those needed to recognize and compile other event flows.

Principle 9207.17 (Self-hosting target). *Lean4 ✓* `BEDC.GroundCompiler.EventFlow.self hosting target recognized compiler`

The self-hosting target of the compilation discipline is represented by an event flow C equipped with a compiler-behavior relation B on event flows such that C is behaviorally equivalent to itself under B. This target datum necessarily includes the judgment that C is a recognized compiler flow. Behavioral equivalence is read through the compiler-behavior classifier of the recognized NameCert *flow attached to C, never as host-level kernel equality. The self-hosting target is consistent with the abstract algebraic reading of Definition 9371.3: a self-hosting compiler is a* NameCert *endomorphism whose* hsame*-preservation witness is itself a recognized event-flow datum.*

9207.5 Accepted object flows and the ground theorem

Definition 9207.18 (Accepted object flow). *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.AcceptedObjectFl`

An accepted object flow *is not merely an event flow that the channel layer can encode and decode. It is an event flow whose recognized structure contains a recognized* NameCert *flow and a recognized* ClosureCert *flow that together witness* $\mathsf{TheoryGate}(N, c)$ *at some grade* $c \in \mathsf{TheoryClosure}$*, accompanied by an audited verification record that witnesses* $\mathsf{FormalStatus}(N, v)$ *at some grade v. The composite reading is the verification gate*

$$\mathsf{VerifiedGate}(N, c, v) \;:=\; \mathsf{TheoryGate}(N, c) \wedge \mathsf{FormalStatus}(N, v),$$

in the sense of Definition 9203.3 and the verification axis of chapter 9203. The recognized ClosureCert *flow records the source, classifier, exactness, ledger, stability, and grade fields of* ClosureCert(N, c)*; these fields are citations of previously closed* NameCert *data, not new mathematical construction supplied at the ground input layer.*

Proposition 9207.19 (Code existence does not imply acceptance). Lean4✓ `BEDC.GroundCompiler.EventFlow.code not acceptance`

The existence of an event flow $S \in$ EventFlow *does not, by itself, imply that S is an accepted object flow.*

Proof. Membership $S \in$ EventFlow asserts only that S is a finite ordered sequence of raw events; channel encoding adds only that this sequence can be displayed and recovered through a self-delimiting bit-level encoder/decoder. Neither datum supplies a recognized NameCert flow, a recognized ClosureCert flow, or an audited verification record. By Definition 9207.18, acceptance requires both the recognized certificate flows and the verification record, and the verification gate VerifiedGate(N, c, v) depends on data not present in S as a bare event flow. Therefore code existence does not imply acceptance. □

Corollary 9207.20 (Accepted export requires recognized certificates). Lean4✓ `BEDC.GroundCompiler.EventFlow.export requires cert`

If a compiled event flow is claimed to export a mature object N at grades (c, v), its recognized structure must include NameCert *and* ClosureCert *flows sufficient for* VerifiedGate(N, c, v)*. Without such recognition the compiled flow is only a raw event-flow display.*

Definition 9207.21 (Structural hidden input). Lean4 (definition) `BEDC.GroundCompiler.EventFlow.StructuralHidd`

A structural hidden input *is any datum used by the compiler, or by a downstream analysis governed by the compilation discipline, that is not present as a raw event in some event flow and not recognized from such a flow by a generated recognizer. Examples include:*

1. *a host AST that supplies node, edge, arity, and category data before event-flow recognition;*
2. *a host list of* NameCert *fields supplied directly to the compiler;*
3. *a YAML or manifest file that supplies package boundaries and field roles;*
4. *an external arity table, object name, theorem identifier, type tag, mode tag, or opcode;*
5. *a host parser state that records a package boundary or proof tree without a corresponding event-flow witness.*

Theorem 9207.22 (No-hidden-input ground theorem). Lean4✓ `BEDC.GroundCompiler.EventFlow.no hidden input`

Every formal datum used by the BEDC compiler occurs in exactly one of the following ways:

1. *as a raw event $w \in$* RawEvent*;*
2. *as an event flow $S \in$* EventFlow*;*
3. *as a structure recognized in an event flow S by a generated recognizer R, witnessed by a kernel-level recognition relation;*
4. *as a certificate-recognized export governed by the verification gate* VerifiedGate(N, c, v)*.*

No other input form is admissible.

Proof. By Definition 9207.3 and Definition 9207.5, the ground data are raw events and event flows. By Definition 9207.10, generated recognizers are themselves event flows; recognized structures are therefore event flows equipped with kernel-level recognition witnesses, not new input forms. By Definition 9207.18, accepted object flows are characterized by recognized NameCert and ClosureCert flows together with an audited verification record, all expressed via the recognition discipline of class 3. Any datum outside classes 1–4 would either be a host-level structure or an unrecognized mathematical primitive, both of which are excluded by Definition 9207.9 and by the anti-primitive discipline of Definition 1.2. □

Corollary 9207.23 (YAML is not a formal compiler input)**.** `Lean4✓` `BEDC.GroundCompiler.EventFlow.yaml not input`

A YAML or manifest file may be generated as a human-readable report of a decoded event flow, but it cannot serve as formal input to the compiler.

Proof. A YAML file supplies package boundaries, field names, and structural roles externally. By Definition 9207.21, these are structural hidden inputs. By Definition 9207.22, structural hidden inputs are not admissible compiler inputs. □

Corollary 9207.24 (AST is not a formal compiler input)**.** `Lean4✓` `BEDC.GroundCompiler.EventFlow.ast not input`

An AST may be reconstructed from a decoded event flow by a generated recognizer, but an AST may not be supplied as initial compiler input.

Proof. An AST supplies node, edge, arity, and category data before event-flow recognition. By Definition 9207.21, these are structural hidden inputs. By Definition 9207.22, they are not admissible as first-layer input. □

9207.6 Worked event-flow examples

Example 9207.25 (Completion skeleton as event flow)**.** *The event flow*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

is a valid element of EventFlow*. A downstream recognizer may later read it as a completion skeleton with role assignments*

0	*initial unit*
00	*one finite repetition*
000	*two finite repetitions*
01	*repeat-forever or boundary witness*
011	*repeat-to-completion or overflow witness*
100	*normal address.*

At the ground input layer, however, the datum is only the sequence $(0, 00, 000, 01, 011, 100)$*. Any claim that the flow licenses a derived completion interface, full-axis interface, or dimension-lift interface must be supported by a recognized* NameCert *flow whose source, pattern, classifier, stability, and ledger subflows are exhibited; this is the same discipline by which* $\mathsf{AxisNat}^{\uparrow}$ *and the Zeckendorf normal-form classifier are licensed in chapter 276 and chapter 278.*

Example 9207.26 (Additive / fold skeleton as event flow)**.** *The event flow*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100$$

is a valid element of EventFlow*. A downstream recognizer may later read it as an additive or fold-shaped skeleton with role assignments*

0	*initial unit*
00	*binary repetition witness*
001	*additive recursion witness*
0011	*iterated-addition completion witness*
0100	*normalized fold seed.*

At the ground input layer the datum remains a sequence of raw events. A claim that the flow exports a derived additive, multiplicative, or fold interface requires a recognized NameCert *flow and, for export, the verification gate of Definition 9207.18. The unary repetition and continuation seeds of chapter 10 are licensed by the same discipline.*

Example 9207.27 (Carry ledger as event flow, not equality)**.** *The event flow*

$$011 \mid 100$$

is a valid element of EventFlow*. A downstream recognizer may later read it as a source-level carry ledger, with role assignments*

011	*pre-normal event*
100	*normal event.*

The flow does not assert equality of the underlying histories, nor does it assert hsame *between* $\mathsf{b1}(\mathsf{b1}(\mathsf{b0}(\mathsf{e})))$ *and* $\mathsf{b0}(\mathsf{b0}(\mathsf{b1}(\mathsf{e})))$*. The relevant downstream relation is the Zeckendorf carry classifier of chapter 9345, which is a structural classifier between recognized normal-form histories and is not equality.*

9207.7 Conservativity and closing observations

Theorem 9207.28 (Event-flow conservativity)**.** *Lean4✓ BEDC.GroundCompiler.EventFlow.event flow conservativity*

Adding the ground event-flow input layer does not add new BEDC kernel objects, new mark-sameness rules, new history-sameness rules, new Ext *or* Cont *rules, or new derived interfaces.*

Proof. The event-flow layer records only finite displayed occurrences of 0 and 1, i.e. displays of b0 and b1 under the convention of Definition 9207.1. It does not add mark constructors beyond b0 and b1, and it does not add any rule generating $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ or $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$, since the kernel rules of chapter 2 are unchanged. It does not license a derived interface either, because derived-interface admission requires the certificate fields of Definition 9203.4; the event-flow layer supplies only a readback carrier for displayed traces. Therefore all kernel and derived-interface formation rules remain unchanged, and the event-flow layer is conservative over the finite kernel. □

Corollary 9207.29 (Event-flow readability is weaker than theoremhood)**.** *Lean4✓ BEDC.GroundCompiler.EventFl weaker than theoremhood*

If a finite sequence is an event flow, it is readable by the compiler. Readability does not imply that the sequence is a theorem, a proof, a recognized package, a recognized NameCert*, or an accepted object flow.*

Proof. Readability asserts only $S \in \mathsf{EventFlow}$. Theoremhood requires a recognized theorem flow per Definition 9207.15. Acceptance requires a recognized $\mathsf{NameCert}$ flow, a recognized $\mathsf{ClosureCert}$ flow, and an audited verification record per Definition 9207.18. Each of these is strictly stronger than $\mathsf{EventFlow}$ membership. □

Remark 9207.30 (Outlook)**.** *The next chapter fixes the channel-level encoding* $\mathsf{Compile}\colon \mathsf{EventFlow} \to \Sigma^*$ *that displays an event flow as a single self-delimiting bit string under the Zeckendorf-safe convention* $\mathsf{b0} \mapsto 0$, $\mathsf{b1} \mapsto 10$, *and event boundary* $\mapsto 11$, *and proves that the resulting language is in bijection with* $\mathsf{EventFlow}$. *The source-level rewrite* $011 \to 100$ *of chapter 278 is a classifier on recognized history flows, not a rewrite of channel-level bit strings; the next chapter records the layer separation explicitly.*

Chapter 9208

Zeckendorf-safe Channel Encoding

chapter 9207 fixes the only formal input the BEDC compiler admits: an element of $\mathsf{EventFlow} = (\Sigma^*)^*$, presented as a finite ordered sequence of raw events with explicit boundary information. Definition 9207.7 records that boundary erasure is non-injective, and Definition 9207.8 records that the formal input therefore cannot be a bare bitstream.

This chapter fixes a channel-level compilation

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \Sigma^*$$

that displays an event flow as a single self-delimiting glyph string. The encoding is required to satisfy four conditions:

1. the output language is over $\Sigma = \{0, 1\}$ alone;
2. encoding is streaming on the input event flow;
3. decoding is streaming on the output glyph string;
4. event boundaries are recovered by the decoder without access to a host parser, schema, manifest, length table, or any structural hidden input in the sense of Definition 9207.21.

The body encoding is the Zeckendorf-safe substitution

$$\zeta(0) = 0, \qquad \zeta(1) = 10,$$

and the event terminator is the glyph pair

$$\mathrm{End} \;=\; 11.$$

The combined effect is that no body of an encoded event contains the adjacent pair 11; consequently the first 11 encountered by the decoder is unambiguously a terminator. The chapter records the encoder, the decoder, the legal-stream language, the round-trip theorems, the bijection $\mathsf{EventFlow} \cong \mathsf{LegalZStream}$, the streaming property of both directions with one-glyph lookahead, and conservativity over the BEDC kernel and over chapter 9207.

The chapter does not perform any source-level rewrite. The Zeckendorf normal-form classifier of chapter 278 and the Zeckendorf carry classifier of chapter 9345 act on recognized history flows; the channel-level encoding presented here acts on event flows. The two layers are independent and are not allowed to overwrite one another (Definition 9208.24).

9208.1 Body encoding and event terminator

Definition 9208.1 (Zeckendorf-safe body encoding)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.Bo`

The body encoding *is the function*

$$\zeta\colon \Sigma^* \longrightarrow \Sigma^*$$

defined by recursion on the input glyph string:

$$\begin{aligned} \zeta(\epsilon) &= \epsilon, \\ \zeta(0\,w) &= 0\,\zeta(w), \\ \zeta(1\,w) &= 10\,\zeta(w). \end{aligned}$$

On individual glyphs, ζ acts by the substitution $0 \mapsto 0$ and $1 \mapsto 10$, applied left-to-right.

Definition 9208.2 (Event terminator)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.EventTerminato`

The event terminator *is the glyph pair*

$$\mathrm{End} \;:=\; 11.$$

The event terminator is a channel-level marker. It signals only that the encoding of the current raw event is complete. It does not assert source-level Zeckendorf carry, completion, hsame*, or any classifier judgment; those readings are licensed only by recognized* NameCert *flows in the sense of Definition 9207.14.*

Definition 9208.3 (Event encoding)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.EventEncoding`

The event encoding *of a raw event $w \in \mathsf{RawEvent}$ is*

$$\mathsf{Ev}(w) \;:=\; \zeta(w)\,11 \;\in\; \Sigma^*.$$

Example 9208.4 (Sample event encodings)**.** *The following table records the body encoding ζ on small inputs:*

$$\begin{aligned} \zeta(\epsilon) &= \epsilon \\ \zeta(0) &= 0 \\ \zeta(1) &= 10 \\ \zeta(00) &= 00 \\ \zeta(01) &= 010 \\ \zeta(10) &= 100 \\ \zeta(11) &= 1010 \\ \zeta(011) &= 01010 \\ \zeta(100) &= 1000 \\ \zeta(0011) &= 001010. \end{aligned}$$

For each row, the corresponding event encoding $\mathsf{Ev}(w)$ is obtained by appending the terminator 11. For instance,

$$\mathsf{Ev}(\epsilon) = 11, \quad \mathsf{Ev}(0) = 011, \quad \mathsf{Ev}(1) = 1011, \quad \mathsf{Ev}(011) = 0101011.$$

The raw event $w = 1$ encodes as 1011, not as 0111. The reason for that choice is recorded in Definition 9208.5.

Remark 9208.5 (Why a naive encoding fails)**.** *A naive scheme might try*

$$0 \mapsto 00, \qquad 1 \mapsto 01, \qquad \mathrm{End} \mapsto 11,$$

producing $\mathsf{Ev}(1) = 0111$*. Read from left to right, the decoder would parse* 0111 *as the body* 01 *followed by* 11 *as terminator and consume the leading* 0 *as a glyph; but the body* 01 *is itself a length-*2 *chunk, leaving an isolated trailing* 1 *that is neither a glyph encoding nor a terminator. The stream would be rejected as malformed even though it was generated as* $\mathsf{Ev}(1)$*. The Zeckendorf-safe substitution* $0 \mapsto 0$*,* $1 \mapsto 10$ *avoids this collision because no encoded body contains adjacent* 11 *(Definition 9208.6).*

Proposition 9208.6 (Body encoding contains no adjacent terminator pattern)**.** *Lean4✓ BEDC.GroundCompiler.Ch encoding no adjacent 11*

For every raw event $w \in \mathsf{RawEvent}$*, the encoded body* $\zeta(w)$ *contains no adjacent pair* 11 *as a substring.*

Proof. By Definition 9208.1, $\zeta(w)$ is a finite concatenation of fragments drawn from $\{0, 10\}$, i.e. $\zeta(w) \in (0 \mid 10)^*$. Neither fragment 0 nor fragment 10 contains the substring 11. The local junctions between two consecutive fragments take exactly four shapes,

$$0\,|\,0, \qquad 0\,|\,10, \qquad 10\,|\,0, \qquad 10\,|\,10,$$

producing the local strings 00, 010, 100, and 1010 respectively, none of which contains 11. Hence no two adjacent positions in $\zeta(w)$ both carry the glyph 1. □

Corollary 9208.7 (The first 11 is the event terminator)**.** *Lean4✓ BEDC.GroundCompiler.ChannelEncoding.first 11 is terminator*

For every raw event w*, the leftmost adjacent pair* 11 *in* $\mathsf{Ev}(w) = \zeta(w)\,11$ *is the appended terminator.*

Proof. By Definition 9208.6, the body $\zeta(w)$ contains no adjacent 11. Therefore the only position in $\mathsf{Ev}(w)$ at which 11 can occur is the appended trailing pair. □

9208.2 Legal event code and channel stream

Definition 9208.8 (Legal event code language)**.** *Lean4 (definition) BEDC.GroundCompiler.ChannelEncoding.LegalEv*

The legal event code language *is*

$$\mathsf{LegalEvent} \;:=\; (0 \mid 10)^*\,11 \;\subseteq\; \Sigma^*.$$

Equivalently, a glyph string $c \in \Sigma^*$ *is a legal event code iff there is a raw event* $w \in \mathsf{RawEvent}$ *with* $c = \mathsf{Ev}(w)$*.*

Definition 9208.9 (Flow encoding)**.** *Lean4 (definition) BEDC.GroundCompiler.ChannelEncoding.FlowEncoding*

The flow encoding *is the function*

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \Sigma^*$$

defined by

$$\mathsf{Compile}(w_0, w_1, \ldots, w_{n-1}) \;:=\; \mathsf{Ev}(w_0)\,\mathsf{Ev}(w_1)\cdots\mathsf{Ev}(w_{n-1}).$$

On the empty event flow, $\mathsf{Compile}(\,) = \epsilon$*.*

Definition 9208.10 (Legal channel-stream language)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.L`

The legal channel-stream language *is*

$$\mathsf{LegalZStream} \;:=\; \big((0 \mid 10)^* \, 11\big)^* \;\subseteq\; \Sigma^*.$$

A glyph string $c \in \Sigma^*$ belongs to $\mathsf{LegalZStream}$ *iff it admits a finite decomposition into legal event codes.*

Example 9208.11 (Completion skeleton encoded)**.** *The completion skeleton event flow of Definition 9207.25,*

$$S \;=\; (0,\, 00,\, 000,\, 01,\, 011,\, 100),$$

encodes event-by-event as

$$\begin{aligned}
\mathsf{Ev}(0) &= 011\\
\mathsf{Ev}(00) &= 0011\\
\mathsf{Ev}(000) &= 00011\\
\mathsf{Ev}(01) &= 01011\\
\mathsf{Ev}(011) &= 0101011\\
\mathsf{Ev}(100) &= 100011.
\end{aligned}$$

The flow encoding $\mathsf{Compile}(S)$ *is therefore*

$$011 \mid 0011 \mid 00011 \mid 01011 \mid 0101011 \mid 100011,$$

where each 11 *is the terminator of the corresponding event code.*

Example 9208.12 (Additive / fold skeleton encoded)**.** *The additive / fold skeleton event flow of Definition 9207.26,*

$$S \;=\; (0,\, 00,\, 001,\, 0011,\, 0100),$$

encodes event-by-event as

$$\begin{aligned}
\mathsf{Ev}(0) &= 011\\
\mathsf{Ev}(00) &= 0011\\
\mathsf{Ev}(001) &= 001011\\
\mathsf{Ev}(0011) &= 00101011\\
\mathsf{Ev}(0100) &= 0100011.
\end{aligned}$$

The flow encoding $\mathsf{Compile}(S)$ *is the concatenation*

$$011 \mid 0011 \mid 001011 \mid 00101011 \mid 0100011,$$

again with one terminator per event code. As recorded in Definition 9207.26, recognizing the events 001, 0011, *and* 0100 *as additive, fold, or normalization witnesses requires recognizer flows and certificate flows that are not part of channel encoding.*

9208.3 Decoders and round-trip theorems

Definition 9208.13 (Channel-level decoder for one event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncodi`

The single-event decoder

$$\mathsf{DecEv}\colon \Sigma^* \longrightarrow \mathsf{Option}(\mathsf{RawEvent} \times \Sigma^*)$$

is defined by left-to-right scan. On success it returns the decoded raw event together with the unconsumed suffix immediately after the terminator; on malformed input it rejects. Starting with empty raw output,

1. *if the next glyph is* 0*, emit raw* 0 *and advance one glyph;*
2. *if the next glyph is* 1*, inspect the following glyph. If it is* 0*, emit raw* 1 *and advance two glyphs; if it is* 1*, the event is complete; if no further glyph exists, the input is rejected.*

The procedure terminates with success on every generated event code $\mathsf{Ev}(w)$ *and consumes exactly that event code.*

Definition 9208.14 (Channel-level decoder for streams)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding`

The stream decoder

$$\mathsf{Decode}\colon \Sigma^* \longrightarrow \mathsf{Option}(\mathsf{EventFlow})$$

applies DecEv *repeatedly: on input* c*, scan glyphs left-to-right and run* DecEv *until its terminator is reached, recording the resulting raw event; then begin a fresh single-event scan from the position immediately after the terminator. The output is the sequence of raw events produced by successive single-event scans, in order. If the scan produces an isolated* 1 *with no following glyph, or any local prefix outside* $(0 \mid 10)^* 11$*, the input is rejected.*

Proposition 9208.15 (The string 0111 is illegal)**.** *Lean4* ✓ `BEDC.GroundCompiler.ChannelEncoding.channel encoding 0111 illegal`

The glyph string 0111 *is not a legal channel stream.*

Proof. Scanning left-to-right, the leading 0 is consumed as raw 0. The remaining glyphs are 111. The first two consume as a terminator 11, ending an event whose raw output is 0. The remaining glyph is an isolated 1, which is neither a body fragment (0 or 10) nor a terminator. By Definition 9208.14, the scan rejects. □

Theorem 9208.16 (Event-level round trip)**.** *Lean4* ✓ `BEDC.GroundCompiler.ChannelEncoding.event level round trip`

For every raw event $w \in \mathsf{RawEvent}$*,*

$$\mathsf{DecEv}(\mathsf{Ev}(w)) \;=\; \mathsf{some}(w, \epsilon).$$

Proof. By induction on the length of w.

Base. If $w = \epsilon$, then $\mathsf{Ev}(\epsilon) = 11$. The decoder reads the terminator immediately and emits the empty raw event $\epsilon = w$.

Step, leading 0*.* If $w = 0\,u$, then $\mathsf{Ev}(w) = 0\,\zeta(u)\,11$. The decoder reads the leading 0, emits raw 0, and advances. The remaining glyphs are $\zeta(u)\,11 = \mathsf{Ev}(u)$. By the induction hypothesis, the remainder decodes to u. The total output is $0\,u = w$.

Step, leading 1*.* If $w = 1\,u$, then $\mathsf{Ev}(w) = 10\,\zeta(u)\,11$. The decoder reads the pair 10, emits raw 1, and advances by two. The remaining glyphs are $\zeta(u)\,11 = \mathsf{Ev}(u)$. By the induction hypothesis, the remainder decodes to u. The total output is $1\,u = w$. □

Theorem 9208.17 (Flow-level round trip)**.** *Lean4* ✓ `BEDC.GroundCompiler.ChannelEncoding.flow level round trip`

For every event flow $S \in \mathsf{EventFlow}$*,*

$$\mathsf{Decode}(\mathsf{Compile}(S)) \;=\; \mathsf{some}(S).$$

Proof. Let $S = (w_0, \ldots, w_{n-1})$. By Definition 9208.9,

$$\mathsf{Compile}(S) \;=\; \mathsf{Ev}(w_0)\,\mathsf{Ev}(w_1)\cdots\mathsf{Ev}(w_{n-1}).$$

By Definition 9208.7, the leftmost 11 in each $\mathsf{Ev}(w_i)$ is the trailing terminator and no 11 appears earlier inside the body $\zeta(w_i)$. Therefore the stream decoder of Definition 9208.14 segments the input exactly at the boundaries between successive $\mathsf{Ev}(w_i)$ codes. By Definition 9208.16, each segment decodes to its source raw event w_i. The output sequence is therefore $(w_0, \ldots, w_{n-1}) = S$. □

Theorem 9208.18 (Legal-stream completeness). *Lean4✓ BEDC.GroundCompiler.ChannelEncoding.legal stream completeness*

For every legal channel stream $c \in \mathsf{LegalZStream}$, there exists an event flow $S \in \mathsf{EventFlow}$ such that

$$\mathsf{Decode}(c) \;=\; \mathsf{some}(S) \qquad \textit{and} \qquad \mathsf{Compile}(S) \;=\; c.$$

Proof. By Definition 9208.10, c admits a decomposition $c = e_0\, e_1 \cdots e_{n-1}$ with $e_i \in \mathsf{LegalEvent}$. The decomposition is unique, since each e_i ends at the leftmost 11 available after the end of e_{i-1} and that leftmost 11 is the terminator by Definition 9208.7.

For each i, by Definition 9208.8 there exists a unique raw event w_i with $e_i = \mathsf{Ev}(w_i)$, and Definition 9208.16 gives $\mathsf{DecEv}(e_i) = w_i$. Hence $\mathsf{Decode}(c) = (w_0, \ldots, w_{n-1})$, and

$$\mathsf{Compile}(\mathsf{Decode}(c)) \;=\; \mathsf{Ev}(w_0) \cdots \mathsf{Ev}(w_{n-1}) \;=\; e_0 \cdots e_{n-1} \;=\; c.$$

□

Corollary 9208.19 (Bijection between event flows and legal channel streams). *Lean4✓ BEDC.GroundCompiler.Cha encoding bijection*

The map $\mathsf{Compile}\colon \mathsf{EventFlow} \to \mathsf{LegalZStream}$ *is a bijection with inverse* Decode*:*

$$\mathsf{EventFlow} \;\cong\; \mathsf{LegalZStream} \;=\; \big(\,(0 \mid 10)^*\, 11\,\big)^*.$$

Proof. Definition 9208.17 gives $\mathsf{Decode} \circ \mathsf{Compile} = \mathrm{id}_{\mathsf{EventFlow}}$. Definition 9208.18 gives $\mathsf{Compile} \circ \mathsf{Decode} = \mathrm{id}_{\mathsf{LegalZStream}}$. Hence $\mathsf{Compile}$ and Decode are mutually inverse, making both bijective. □

9208.4 Streaming behavior and host-structure independence

Proposition 9208.20 (Channel encoder is streaming). *Lean4✓ BEDC.GroundCompiler.ChannelEncoding.encoder streaming*

The map $\mathsf{Compile}$ *is streaming on its input: while reading raw events left-to-right, the encoder emits the corresponding glyphs immediately upon seeing each raw glyph and emits the terminator* 11 *immediately upon seeing each event boundary.*

Proof. The local rules of Definition 9208.1 are $0 \mapsto 0$ and $1 \mapsto 10$, both depending on the current raw glyph alone. The terminator rule of Definition 9208.3 depends on the current event boundary alone. Neither rule consults future raw events, future raw glyphs, an event-flow length table, an AST, a Pkg object, or any structural hidden input in the sense of Definition 9207.21. Therefore the encoder may be implemented as a single left-to-right pass. □

Proposition 9208.21 (Channel decoder is streaming with one-glyph lookahead). *Lean4✓ BEDC.GroundCompiler.C streaming one glyph lookahead*

The map Decode *is streaming on its input with at most one-glyph lookahead.*

Proof. By Definition 9208.13, the decoder advances on the current glyph except when the current glyph is 1, in which case it inspects the next glyph to decide between raw 1 (next glyph 0), terminator (next glyph 1), and rejection (no next glyph). In every case the decision uses at most one glyph beyond the current position. No further lookahead, no buffer of previous events, and no consultation of an external table is required. □

Corollary 9208.22 (No tree, no manifest, no table). `Lean4✓ BEDC.GroundCompiler.ChannelEncoding.no tree no manifest no table`

Channel-level encoding and decoding can be carried out without an AST, a Pkg *object, a recognized* NameCert *flow, a recognized theorem flow, a YAML manifest, an external arity table, or any other structural hidden input.*

Proof. By Definition 9208.20, the encoder needs only the current raw glyph and a flag for the current event boundary. By Definition 9208.21, the decoder needs only the current glyph plus one-glyph lookahead. Neither function consults any datum forbidden by Definition 9207.21. □

Remark 9208.23 (Empty events). *The empty raw event* $\epsilon \in \mathsf{RawEvent}$ *encodes as* $\mathsf{Ev}(\epsilon) = 11$*, and the decoder of Definition 9208.13 accepts the input* 11 *as a single empty event. Whether a particular generated event flow is allowed to contain empty events is a question for the generators of chapter 9207, not for channel encoding; the channel layer accepts and emits them losslessly.*

Remark 9208.24 (Layer separation: source rewrite versus channel rewrite). *The Zeckendorf normal-form classifier of chapter 278 acts on recognized history flows:* ZNormal *is generated structurally on* b0*- and* b1*-extensions of histories, and the carry classifier of chapter 9345 relates two such histories. None of these is a rewrite of bit strings.*

The channel encoding of this chapter, by contrast, acts on event flows: ζ *substitutes raw glyphs to channel glyphs and* $\mathrm{End} = 11$ *marks event boundaries. The channel encoder is not allowed to rewrite a body fragment* 11 *to* 1010 *or to apply the source-level rewrite* $011 \to 100$ *to a channel stream; doing so would break the bijection of Definition 9208.19 and conflate the two layers.*

In short: source-level relations live on recognized history flows; channel-level relations live on event-flow / legal-stream displays. The two layers commute through their respective recognizers but are never composed in either direction without a bridge.

9208.5 Conservativity and closing observations

Principle 9208.25 (Channel code does not imply theoremhood). `Lean4✓ BEDC.GroundCompiler.ChannelEncoding stream not theoremhood`

Membership $c \in \mathsf{LegalZStream}$ *asserts only that* c *is a self-delimiting display of an event flow. It does not assert that* c *encodes a theorem, a proof, a recognized* NameCert*, a recognized* ClosureCert*, an accepted object flow, or a derived interface. Mature object export is governed by the verification gate* $\mathsf{VerifiedGate}(N, c, v)$ *of Definition 9207.18, not by channel encodability.*

In particular, the empty channel stream is legal because it is the encoding of the empty event flow, but it does not carry a recognized theorem flow.

Theorem 9208.26 (Channel conservativity). `Lean4✓ BEDC.GroundCompiler.ChannelEncoding.channel conservativity`

Every displayed mark occurring in $\mathsf{Compile}(S)$ *for an event flow* S *is one of the two existing kernel marks* b0 *and* b1*. Adjoining the channel-level encoder* Compile*, the decoder* Decode*, the legal*

event language LegalEvent*, and the legal-stream language* LegalZStream *to the BEDC compiler discipline therefore introduces no third channel glyph and does not add new BEDC kernel objects, new mark-sameness rules, new history-sameness rules, new* Ext *or* Cont *rules, new derived interfaces, or new recognizer relations.*

Proof. The channel-level objects of this chapter operate exclusively on the external display of event flows. ζ rewrites raw glyphs to channel glyphs; End adds a terminator at each event boundary; Compile concatenates event encodings; Decode inverts Compile on legal channel streams. None of these maps generates new raw marks beyond b0 and b1, new msame or hsame judgments, new Ext or Cont rules, or new derived interfaces, since interface admission requires the certificate fields of Definition 9203.4. The recognizer relations RecognizesPkg, RecognizesNameCert, RecognizesTheorem, and RecognizesCompiler are already fixed in chapter 9207; this chapter neither adds nor modifies them. By the same argument as Definition 9207.28, the channel layer is conservative over the BEDC finite kernel and over chapter 9207. □

Remark 9208.27 (Outlook)**.** *The next chapter strengthens the layer-separation discipline of Definition 9208.24. It records that source-level relations on recognized history flows—in particular the carry classifier of chapter 9345 and any future fold or completion classifier—may not be applied to channel streams, and that channel-level rewrites may not be transported to histories without a bridge whose source is a recognized* NameCert *flow and whose target is a recognized channel-encoded event flow.*

Chapter 9209

Source-Level Events and Channel-Level Bits

chapter 9207 fixes the source-layer input language $\mathsf{EventFlow} = (\Sigma^*)^*$, and chapter 9208 fixes the channel-layer encoder

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \mathsf{LegalZStream} \;=\; \big((0 \mid 10)^* \, 11\big)^*,$$

together with its decoder Decode and the bijection $\mathsf{EventFlow} \cong \mathsf{LegalZStream}$ of Definition 9208.19.

The two layers share the glyph set $\Sigma = \{0, 1\}$, but the role of each glyph differs by layer:

- in the source layer, 0 and 1 display the raw marks $\mathsf{b0}$ and $\mathsf{b1}$, and the pair 11 is a permissible substring of a raw event;
- in the channel layer, 0 and 1 are encoded glyphs, and 11 is the unique event terminator.

This chapter records the discipline that follows from this difference. The central principle is that source-level normalization, classification, and ledger-recording act on decoded event flows in $\mathsf{EventFlow}$, never on compiled glyph strings in $\mathsf{LegalZStream}$. The Zeckendorf carry classifier of chapter 9345 and the Zeckendorf normal-form classifier of chapter 278 are source-layer relations on recognized history flows; they may not be applied to channel glyph strings as substring rewrites. The chapter records the mistake explicitly, supplies the correct decode-normalize-compile pipeline, and proves a layer-separation theorem with seven independent components.

9209.1 Source layer and channel layer

Definition 9209.1 (Source layer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceChannel.SourceLayer`

The source layer *consists of raw events and event flows of chapter 9207:*

$$\mathsf{RawEvent} \;=\; \Sigma^*, \qquad \mathsf{EventFlow} \;=\; (\Sigma^*)^*.$$

A glyph string in the source layer, e.g. 011*, denotes the raw event* $011 \in \mathsf{RawEvent}$*. The pair* 11 *is a permissible substring of a source raw event; the raw events* 11*,* 011*, and* 0011 *are all elements of* $\mathsf{RawEvent}$*.*

Definition 9209.2 (Channel layer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceChannel.ChannelLayer`

The channel layer *consists of compiled glyph strings:*

$$\mathsf{LegalZStream} \;=\; \big((0 \mid 10)^* \, 11\big)^*.$$

In the channel layer, the pair 11 *has a single fixed role: it marks the end of the encoding of the current event (Definition 9208.2). A channel-layer* 11 *is therefore not a source-layer raw substring, not a Zeckendorf carry source, not a completion witness, not a theorem seal, and not an* hsame *judgment.*

Remark 9209.3 (Same display, different layers)**.** *A glyph string written on the page may be read at either layer, but never at both simultaneously. The source-layer reading places the string in* RawEvent *or* EventFlow*; the channel-layer reading places the string in* LegalZStream*. The reading is fixed by the surrounding context—typically by which side of* Compile *or* Decode *the string is generated by—and the chapter's discipline insists that the reading not silently change.*

Definition 9209.4 (Source-level relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceChannel.SourceLevelRel`

A source-level relation *is a binary relation*

$$(\rightsquigarrow) \subseteq \mathsf{RawEvent} \times \mathsf{RawEvent}$$

on raw events. By extension, a source-level relation may be lifted pointwise or via context to a relation on event flows. A relation $w \rightsquigarrow v$ *asserts only a source-layer fact between two raw events; it asserts neither* hsame *between underlying histories nor channel-layer glyph-string rewriting.*

A representative source-level relation is the Zeckendorf carry $\rightsquigarrow_Z$*, whose canonical instance is*

$$011 \rightsquigarrow_Z 100,$$

*read as: "*011 *is a pre-normal source event whose Zeckendorf-normal source event is* 100*." The classifier-level reading of this relation is the Zeckendorf carry classifier of chapter 9345, which is a structural relation between recognized normal-form histories, not glyph-string rewriting.*

Definition 9209.5 (Source carry ledger)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceChannel.SourceCarryLedg`

Given a source-level relation $w \rightsquigarrow v$*, the corresponding* ledger flow *is the event flow*

$$(w, v) \in \mathsf{EventFlow}.$$

Displayed, the flow is written $w \mid v$ *and read as "pre-normal event* w *followed by normal event* v*."*

For the canonical Zeckendorf instance $011 \mid 100$*, the channel encoding is computed event-by-event using Definition 9208.3:* $\mathsf{Ev}(011) = 0101011$ *and* $\mathsf{Ev}(100) = 100011$*, so*

$$\mathsf{Compile}(011, 100) = 0101011\,100011.$$

The compiled stream records both events; it does not erase the pre-normal event 011*.*

9209.2 No channel-level source rewrite

Principle 9209.6 (No channel-level source rewrite)**.** *Lean4✓* `BEDC.GroundCompiler.SourceChannel.no channel level source rewrite`

For every source-level relation $w \rightsquigarrow v$*, it is not admissible to rewrite the displayed substring* w *to* v *inside the channel string* $\mathsf{Ev}(w)$ *or inside the compiled stream* $\mathsf{Compile}(S)$*. In particular, the source-level pair* $011 \rightsquigarrow_Z 100$ *is not a channel-layer string-rewriting rule.*

Example 9209.7 (Channel-level $011 \to 100$ rewrite breaks legality)**.** *The source raw event* 011 *has channel encoding*

$$\mathsf{Ev}(011) = \zeta(0)\,\zeta(1)\,\zeta(1)\,11 = 0\,10\,10\,11 = 0101011.$$

Suppose the displayed substring 011 *at the right end of* 0101011 *is mistakenly rewritten to* 100*:*

$$0101011 \longrightarrow 0101100.$$

Decoding the result of this rewrite glyph-by-glyph, per Definition 9208.13: 0 *emits raw* 0*;* 10 *emits raw* 1*;* 11 *marks event end; the remaining glyphs are* 00*, which contains no subsequent terminator. By Definition 9208.14, the stream is rejected. Thus the channel-level rewrite has destroyed legality.*

Proposition 9209.8 (Channel-level source rewrite may destroy legality)**.** *Lean4✓ BEDC.GroundCompiler.SourceCh level source rewrite may destroy legality*

There exists a legal channel stream $c \in \mathsf{LegalZStream}$ *and a displayed source-level rewrite* $w \rightsquigarrow v$ *such that the result of replacing the displayed substring* w *in* c *by* v *is not a legal channel stream.*

Proof. Take $c = \mathsf{Ev}(011) = 0101011$, which lies in $\mathsf{LegalZStream}$ by Definition 9208.8. Apply the source-level rewrite $011 \rightsquigarrow_Z 100$ to the displayed substring 011 at the right end of c. The result is $c' = 0101100$. By Definition 9209.7, the stream-decoder of Definition 9208.14 rejects c', so $c' \notin \mathsf{LegalZStream}$. Hence channel-level application of a source-level rewrite is not legality-preserving in general. □

Corollary 9209.9 (A source normalizer is not a channel transducer)**.** *Lean4✓ BEDC.GroundCompiler.SourceChanne output factors through decoding*

A source-level normalizer

$$N : \mathsf{EventFlow} \longrightarrow \mathsf{EventFlow}$$

does not lift to a glyph-level transducer $\Sigma^* \to \Sigma^*$ *obtained by rewriting displayed substrings of channel streams. To apply* N *to an already-compiled channel stream, the admissible procedure is to decode first, normalize on the resulting event flow, and recompile: whenever the channel normalizer returns an output* o *on a channel string* c*, there is an event flow* S *such that* $\mathsf{Decode}(c) = \mathsf{some}(S)$ *and* $o = \mathsf{Compile}(N(S))$*.*

9209.3 Decode-normalize-compile pipeline

Definition 9209.10 (Decode-normalize-compile pipeline)**.** *Lean4 (definition) BEDC.GroundCompiler.SourceChannel*

Given a source-level normalizer $N : \mathsf{EventFlow} \to \mathsf{EventFlow}$*, the* decode-normalize-compile pipeline *is the composition*

$$\mathsf{NormalizeChannel}_N \;:=\; \mathsf{Compile} \circ N \circ \mathsf{Decode} \;:\; \mathsf{LegalZStream} \longrightarrow \mathsf{LegalZStream},$$

read as

$$\begin{aligned} &\textit{channel stream} \\ &\xrightarrow{\mathsf{Decode}} \textit{source event flow} \\ &\xrightarrow{N} \textit{normalized source event flow} \\ &\xrightarrow{\mathsf{Compile}} \textit{channel stream.} \end{aligned}$$

On an arbitrary channel glyph string the same operation is partial: it rejects malformed input when Decode *rejects, and otherwise returns* $\mathsf{Compile}(N(S))$ *for the decoded event flow* S*. Restricted to* $\mathsf{LegalZStream}$*, it is the displayed map above. By construction the pipeline forbids any direct rewrite on channel glyphs.*

Theorem 9209.11 (The pipeline preserves channel legality). *Lean4✓ BEDC.GroundCompiler.SourceChannel.norma channel preserves legality*

For every channel stream $c \in$ LegalZStream and every total source-level normalizer N: EventFlow $\to$ EventFlow,

$$\mathsf{NormalizeChannel}_N(c) \in \mathsf{LegalZStream}.$$

Proof. By Definition 9208.19, $\mathsf{Decode}(c) \in \mathsf{EventFlow}$. By the totality of N, $N(\mathsf{Decode}(c)) \in \mathsf{EventFlow}$. By Definition 9208.9, Compile sends every event flow into LegalZStream. The composite output is therefore in LegalZStream. □

9209.4 Ledger preservation and classifier quotient

Definition 9209.12 (Ledger-preserving source relation). *Lean4 (definition) BEDC.GroundCompiler.SourceChannel.L*

Given a source-level relation $w \rightsquigarrow v$, a recording operation

$$L\colon \mathsf{RawEvent} \times \mathsf{RawEvent} \longrightarrow \mathsf{EventFlow}$$

is ledger-preserving *when every witnessed pair is recorded as the ordered event flow*

$$L(w, v) \;=\; w \mid v.$$

The ledger-preserving form of $w \rightsquigarrow v$ is the event flow $w \mid v$ of Definition 9209.5, not the singleton flow v alone.

Principle 9209.13 (Normal address does not erase the raw event). *Lean4✓ BEDC.GroundCompiler.SourceChannel. address does not erase raw event*

For any source-level relation $w \rightsquigarrow v$, the right-hand event v may be displayed as a normal address, but the left-hand event w must remain present in the recorded ledger flow. The flow $w \mid v$ is admissible; deleting w in favor of the singleton flow v is not. This discipline is the source-layer specialization of Definition 1.4: any compression of generated material requires a ledger that tracks the hidden material; composite ledgers must keep the source map rather than only the visible interface.

Proposition 9209.14 (Ledger records distinguish ordered pairs). *Lean4✓ BEDC.GroundCompiler.SourceChannel.l records distinguish ordered pairs*

The ledger-record map $w, v \mapsto (w, v)$ is injective on its arguments: if $(w, v) = (w', v')$, then $w = w'$ and $v = v'$.

Proof. Equality of ordered pairs is componentwise. The conclusion is immediate. □

Corollary 9209.15 (Quotient may be many-to-one; ledger flow is lossless). *Lean4✓ BEDC.GroundCompiler.Source may be many to one ledger lossless*

A source-level classifier may identify multiple raw events as one classified object, but the ledger flow recording the carry events remains lossless. In particular, even when a classifier asserts $w \sim v$, the ledger flow $w \mid v$ continues to distinguish the pre-normal event w from the normal event v.

Definition 9209.16 (Source-level classified object). *Lean4 (definition) BEDC.GroundCompiler.SourceChannel.Source*

A source-level classifier *consists of a carrier of classified objects and a classification map*

$$q\colon \mathsf{RawEvent} \longrightarrow \mathsf{ClassifiedObject}.$$

A classified object is an element of this carrier. For the Zeckendorf carry reading of chapter 9345, the pre-normal event 011 *and the normal event* 100 *may be sent to the same classified object, while at the raw-event layer the two raw events remain distinct.*

Proposition 9209.17 (Classified-object map is generally not injective)**.** Lean4✓ BEDC.GroundCompiler.SourceCha object map not injective

The natural map

$$q\colon \mathsf{RawEvent} \longrightarrow \mathsf{ClassifiedObject}$$

is generally not injective.

Proof. It suffices to exhibit one source-level classifier that identifies two distinct raw events. Let the classifier carrier have a carry class, and send both 011 and 100 to that class. The raw events 011 and 100 are different finite sequences of raw marks, but their classified images coincide. Hence a classified-object map need not be injective. □

Corollary 9209.18 (Bijection is unconditional only at the raw-flow layer)**.** Lean4✓ BEDC.GroundCompiler.SourceC only at raw layer

The unconditional bijection supplied by channel encoding is between raw event flows and legal channel streams,

$$\mathsf{EventFlow} \;\cong\; \mathsf{LegalZStream} \quad \textit{(Definition 9208.19).}$$

The composite map from raw event data to classified objects is generally not injective. Therefore, when a downstream argument speaks of a bijection between "theory" and "code," the theory side must be the canonical raw-event flow, not an abstract classified proposition.

9209.5 Encoding and decoding examples for ledger flows

Example 9209.19 (Carry ledger encoding)**.** *The carry-ledger event flow* $011 \mid 100$ *encodes event-by-event as*

$$\mathsf{Ev}(011) \;=\; 0101011, \qquad \mathsf{Ev}(100) \;=\; 100011,$$

hence

$$\mathsf{Compile}(011, 100) \;=\; 0101011\,100011.$$

The stream decoder of Definition 9208.14 segments the result at the leftmost terminator, recovering $0101011 \mid 100011$ *and decoding to* $(011, 100)$*. The pre-normal event* 011 *is preserved by the channel round trip.*

Example 9209.20 (Completion skeleton with carry ledger)**.** *The event flow* $0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$ *of Definition 9207.25 encodes event-by-event as*

$$\begin{aligned}
\mathsf{Ev}(0) &= 011\\
\mathsf{Ev}(00) &= 0011\\
\mathsf{Ev}(000) &= 00011\\
\mathsf{Ev}(01) &= 01011\\
\mathsf{Ev}(011) &= 0101011\\
\mathsf{Ev}(100) &= 100011.
\end{aligned}$$

The complete channel stream is therefore the concatenation

$$011\,0011\,00011\,01011\,0101011\,100011,$$

which is in LegalZStream *by Definition 9208.19. If a downstream classifier asserts* $011 \sim_Z 100$*, the assertion is read against the decoded source events, not against any displayed substring of the channel stream.*

Theorem 9209.21 (Source-level normalization is conservative over channel decoding). *Lean4✓ BEDC.GroundCompiler.SourceChannel.source normalization conservative over decoding*

For every source-level normalizer $N : \mathsf{EventFlow} \to \mathsf{EventFlow}$*, the diagram*

$$\mathsf{LegalZStream} \xrightarrow{\mathsf{Decode}} \mathsf{EventFlow} \xrightarrow{N} \mathsf{EventFlow} \xrightarrow{\mathsf{Compile}} \mathsf{LegalZStream}$$

is well-defined. Source-level normalization therefore acts on channel streams indirectly through the decode-normalize-compile pipeline of Definition 9209.10; it does not act on channel streams as a direct glyph-level rewrite.

Proof. The first arrow is well-defined by Definition 9208.19; the second arrow is well-defined by the totality of N; the third arrow is well-defined by Definition 9208.9. The composite is therefore a well-defined function from $\mathsf{LegalZStream}$ to itself. Direct glyph-level channel rewriting is excluded from this diagram and is forbidden by Definition 9209.6. □

9209.6 Channel substrings versus source events

Proposition 9209.22 (Channel terminator is not the source event 11). *Lean4✓ BEDC.GroundCompiler.SourceChan terminator not source event 11*

The channel-layer terminator 11 *of Definition 9208.2 is not the source raw event* $11 \in \mathsf{RawEvent}$.

Proof. The source raw event 11 has channel encoding

$$\mathsf{Ev}(11) \;=\; \zeta(1)\,\zeta(1)\,11 \;=\; 10\,10\,11 \;=\; 101011,$$

whereas the channel terminator is the literal glyph pair 11. If the channel terminator were identified with the source raw event 11, the encoding of the source raw event 11 would be 11 rather than 101011. The two strings are distinct, so the identification fails. □

Corollary 9209.23 (The source raw event 11 encodes unambiguously). *Lean4✓ BEDC.GroundCompiler.SourceChan event 11 unambiguous*

The source raw event 11 *encodes as the channel string* $\mathsf{Ev}(11) = 101011$*. The decoder of Definition 9208.13 reads this as* $10 \to \text{raw } 1$, $10 \to \text{raw } 1$, $11 \to \text{event end}$*, recovering* $11 \in \mathsf{RawEvent}$ *without ambiguity.*

Principle 9209.24 (No channel proof by substring). *Lean4✓ BEDC.GroundCompiler.SourceChannel.no channel proof by substring*

The appearance of a displayed substring u *inside a channel stream* $c \in \mathsf{LegalZStream}$ *does not entail that the source event* u *appears in* $\mathsf{Decode}(c)$*. Source-event analysis is admissible only on the output of* Decode.

Proposition 9209.25 (Channel substrings are not source events). *Lean4✓ BEDC.GroundCompiler.SourceChannel. substring not source event*

There exists a channel stream $c \in \mathsf{LegalZStream}$ *and a displayed substring* u *of* c *such that* u *is not any source event of* $\mathsf{Decode}(c)$.

Proof. Take $c = \mathsf{Ev}(1) = 1011$. The displayed substring $u = 011$ occurs at positions 2–4 of c. By Definition 9208.16, $\mathsf{Decode}(c) = (1)$. The only source event of $\mathsf{Decode}(c)$ is 1, not 011. Hence the displayed substring 011 is not a source event of the decoded flow. □

Corollary 9209.26 (Source analysis must begin after decoding)**.** *Lean4✓ BEDC.GroundCompiler.SourceChannel.s analysis after decoding*

Any source-level analysis—finite repetition, completion witness, Zeckendorf carry, NameCert *source pattern, theorem event, ledger event—is admissible only after* Decode *has been applied to the channel stream. Source-level analyses applied directly to channel glyphs are not admissible.*

9209.7 Layer separation theorem and closing observations

Theorem 9209.27 (Layer separation theorem)**.** *Lean4✓ BEDC.GroundCompiler.SourceChannel.layer separation*

The source layer and the channel layer satisfy the following seven separation conditions:

1. *source events are elements of* RawEvent*;*
2. *channel streams are elements of* LegalZStream*;*
3. Compile *encodes source event flows into channel streams;*
4. Decode *recovers source event flows from channel streams;*
5. *source-level normalization acts on* EventFlow *only;*
6. *the channel-layer event terminator* 11 *is not the source raw event* 11*;*
7. *displayed channel substrings are not, in general, source events of the decoded flow.*

Proof. Items (1) and (2) are Definition 9209.1 and Definition 9209.2. Items (3) and (4) are Definition 9208.9, Definition 9208.14, and Definition 9208.19. Item (5) is Definition 9209.4 and Definition 9209.21. Item (6) is Definition 9209.22. Item (7) is Definition 9209.25. □

Remark 9209.28 (Outlook)**.** *The next chapter records how recognized package,* NameCert, ClosureCert*, theorem, and ledger structures are read inside a decoded event flow. By Definition 9209.27, every such recognition begins from* Decode(c)*, never from* c *as a glyph string. The recognizer flows of Definition 9207.10–Definition 9207.16 are therefore source-layer relations on event flows, in line with Definition 9207.9; their composition with* Compile *and* Decode *is the discipline by which structural readings reach the channel layer without appearing as primitive compiler inputs.*

Chapter 9210

Package Structure Is Generated, Not Input

chapter 9207 and chapter 9208 establish the raw bijection

$$\mathsf{EventFlow} \;\cong\; \mathsf{LegalZStream} \;=\; \big((0 \mid 10)^* \, 11\big)^*.$$

chapter 9209 then separates source-level events from channel-level bits: source-level normalization, classification, and ledger recognition act on decoded event flows, never on channel substrings.

This chapter records the next no-hidden-input constraint. Package structure is not a formal compiler input. If the compiler directly accepted a datum such as

$$\mathrm{Package}(\mathrm{source}, \mathrm{pattern}, \mathrm{classifier}, \mathrm{stability}, \mathrm{ledger})$$

or a host object such as TheoryPkg, NameCert, or TheoremPkg, then the package fields, boundaries, roles, order, and hierarchy would have been supplied from outside the generated event flow. That is exactly the structural hidden input excluded by Definition 9207.21.

The chapter's central rule is therefore:

$$\boxed{\text{packages are recognized structures in event flows, not compiler inputs.}}$$

This is the source-layer form of the package and gap discipline already used by BEDC: package tokens are generated by answer histories, package interfaces expose only the token and its ledger, and nontrivial package visibility requires a gap ledger recording which source histories enter the package token.

9210.1 Package recognition on event flows

Definition 9210.1 (Package candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.PackageC`

A package candidate flow *is an event flow*

$$S \in \mathsf{EventFlow}.$$

The phrase does not assert that S is a package. It asserts only that S is available as a candidate source flow on which package recognition may be attempted.

Definition 9210.2 (Generated package recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.Ge`

A generated package recognizer *is an event flow*

$$R \in \mathsf{EventFlow}.$$

It is not a host-level parser, YAML schema, AST constructor, or external grammar. Its public formal carrier is itself generated from the two raw marks through the event-flow layer.

Definition 9210.3 (Package recognition relation). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.Pack`

The package recognition relation is written

$$\mathsf{RecognizesPkg}(R, S), \qquad R, S \in \mathsf{EventFlow}.$$

It is read as: the generated recognizer flow R recognizes the event flow S as a package flow. If such an R exists, then S is a recognized package flow.

Remark 9210.4 (Recognition is not input). *The judgment* $\mathsf{RecognizesPkg}(R, S)$ *does not mean that S has been externally tagged as a package. It means that an already generated recognizer flow R recognizes package-shaped structure in the source event flow S. The direction is therefore*

$$\mathsf{EventFlow} \longrightarrow \mathsf{RecognizedPackageFlow},$$

not

$$\mathsf{Pkg} \longrightarrow \mathsf{EventFlow}.$$

Definition 9210.5 (Recognized package flow). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.PkgFlow`

For an event flow S, define

$$\mathsf{PkgFlow}(S) \;:\Longleftrightarrow\; \exists R \in \mathsf{EventFlow},\ \mathsf{RecognizesPkg}(R, S).$$

Thus S is a recognized package flow exactly when a generated recognizer flow recognizes it as package-shaped.

Principle 9210.6 (No external package input). *Lean4 ✓* `BEDC.GroundCompiler.PackageGenerated.no external package input`

At the formal datum boundary, a host package object is not an admissible compiler input:

$$\neg\, \mathsf{FormalCompilerInput}(\mathrm{hostPkg}).$$

There is no formal compiler entry point

$$\mathsf{CompilePkg}\colon \mathsf{Pkg} \longrightarrow \mathsf{LegalZStream}.$$

The only formal channel compiler remains

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \mathsf{LegalZStream}.$$

If an event flow S is recognized as a package flow, its package code is just $\mathsf{Compile}(S)$.

Proposition 9210.7 (Package recognition does not change channel code). *Lean4 ✓* `BEDC.GroundCompiler.Package recognition preserves code`

If $\mathsf{RecognizesPkg}(R, S)$*, then the channel encoding of S is still* $\mathsf{Compile}(S)$*. Package recognition does not add, delete, or rewrite channel bits.*

Proof. The judgment $\mathsf{RecognizesPkg}(R, S)$ is a source-level recognition relation. It says only that S is recognized by the generated recognizer R as package-shaped. The channel encoding is given by $\mathsf{Compile}$ from Definition 9208.9, and depends only on the event flow S. Since the recognition judgment does not alter S, it does not alter $\mathsf{Compile}(S)$. □

Corollary 9210.8 (Recognized package code is not a separate code)**.** *Lean4 ✓* `BEDC.GroundCompiler.PackageGener` `code not separate`

If S is a recognized package flow, then

$$\mathsf{PackageCode}(S) \;:=\; \mathsf{Compile}(S)$$

is the ordinary channel code of the same event flow, not the output of a separate package encoder.

9210.2 Roles, seals, and ledgers

Definition 9210.9 (Package role emergence)**.** *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.PackageR`

Let ρ range over the finite package-role list

source flow,
pattern flow,
classifier flow,
stability flow,
ledger flow,
package-seal flow.

For a generated package recognizer R, a package candidate flow S, and a source-level event subflow T, define

$$\mathsf{PkgRole}_R(S, T, \rho) \;:\Longleftrightarrow\; \mathsf{RecognizesPkg}(R, S) \wedge T \in \mathsf{EventFlow}_{\neq\emptyset}.$$

The role name ρ is not a ground input. It is a recognition result attached to an event-flow subflow after $\mathsf{RecognizesPkg}(R, S)$ *has already been witnessed.*

Remark 9210.10 (Relation to the five fields of $\mathsf{NameCert}$)**.** *The five fields of a* $\mathsf{NameCert}$*,*

$$\mathsf{SourceSpec}, \qquad \mathsf{PatternSpec}, \qquad \mathsf{ClassifierSpec}, \qquad \mathsf{StabilityCert}, \qquad \mathsf{LedgerPolicy},$$

are derived-interface certificate roles. $\mathsf{PatternSpec}$ *records stable behavior on the source,* $\mathsf{ClassifierSpec}$ *records how source material is grouped,* $\mathsf{StabilityCert}$ *records closure, reuse, descent, composition, and seal stability, and* $\mathsf{LedgerPolicy}$ *records what a name hides and how the hidden material is tracked. This chapter does not treat those five fields as external input. It says only that, in a recognized package flow, a generated recognizer may identify subflows corresponding to those roles.*

Definition 9210.11 (Package-seal subflow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.PackageSe`

If R recognizes a source-level event subflow σ of S as expressing package completion, then σ is a package-seal subflow *of S, written*

$$\mathsf{PkgSeal}_R(S, \sigma) \;:\Longleftrightarrow\; \mathsf{PkgRole}_R(S, \sigma, \mathrm{seal}).$$

The package-seal subflow is not the channel terminator 11*. It is source-level event-flow structure, recovered through* Decode *and then recognized by R.*

Proposition 9210.12 (Package seal is not the channel terminator). *Lean4✓ BEDC.GroundCompiler.PackageGenera seal is source subflow*

If $\mathsf{PkgSeal}_R(S, \sigma)$*, then* σ *is a source-level event subflow. In particular, its formal carrier is an element of* $\mathsf{EventFlow}_{\neq\emptyset}$*, whereas the channel-level terminator* 11 *is a channel-code delimiter.*

Proof. The channel-level terminator 11 belongs to the encoding mechanism of LegalZStream, as fixed in Definition 9208.2. A package seal is a subflow of EventFlow recognized by R. These live on different layers. By the layer-separation theorem Definition 9209.27, a channel substring cannot be used directly as a source event or source subflow. Therefore the package seal is not the channel terminator. □

Definition 9210.13 (Package ledger subflow). *Lean4 (definition) BEDC.GroundCompiler.PackageGenerated.Package*

The package ledger subflow *predicate is*

$$\mathsf{PkgLedger}_R(S, L) \;:\Longleftrightarrow\; \mathsf{PkgRole}_R(S, L, \text{ledger}).$$

Thus if $\mathsf{RecognizesPkg}(R, S)$ *and* L *is recognized as the ledger-role subflow of* S*, then* L *records the source material hidden or compressed by* S*. The ledger subflow is part of the decoded source event flow, not an external provenance table.*

Principle 9210.14 (No package without ledger). *Lean4✓ BEDC.GroundCompiler.PackageGenerated.no package without ledger*

A nontrivial recognized package flow must carry a recognized ledger subflow:

$$\mathsf{PkgFlow}(S) \implies \exists R, L,\ \mathsf{RecognizesPkg}(R, S) \wedge \mathsf{PkgLedger}_R(S, L),$$

where the ledger witness is an event subflow identified by the same generated recognizer that recognizes S *as package-shaped. This is the package-level form of the gap policy: visible package tokens are not admissible for compression unless their ledgers record the histories entering the token.*

Theorem 9210.15 (Visible package code alone is insufficient). *Lean4✓ BEDC.GroundCompiler.PackageGenerated. code insufficient*

If there are no R *and* L *such that*

$$\mathsf{RecognizesPkg}(R, S) \wedge \mathsf{PkgLedger}_R(S, L),$$

then S *is not a recognized package flow:*

$$\neg\, \mathsf{PkgFlow}(S).$$

Proof. By Definition 9210.14, any recognized package flow supplies a generated recognizer R and a ledger subflow L with

$$\mathsf{RecognizesPkg}(R, S) \wedge \mathsf{PkgLedger}_R(S, L).$$

If no such pair exists, the assumption $\mathsf{PkgFlow}(S)$ is impossible. □

Corollary 9210.16 (A package code is not merely a visible token). *Lean4✓ BEDC.GroundCompiler.PackageGenerat not merely visible token*

A package code cannot consist only of a visible package-token event. It must include enough recognized ledger flow to recover provenance, unless the recognizer identifies the package as trivial and non-compressing.

9210.3 Compile-decode invariance

Definition 9210.17 (Recognition-preserving compilation). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenera`

Compilation is recognition-preserving for packages *when, for every* $\mathsf{RecognizesPkg}(R, S)$,

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S$$

and therefore

$$\mathsf{RecognizesPkg}(R, \mathsf{Decode}(\mathsf{Compile}(S))).$$

Theorem 9210.18 (Channel compilation preserves package recognition). *Lean4✓* `BEDC.GroundCompiler.Package(` `compilation preserves recognition`

If $\mathsf{RecognizesPkg}(R, S)$, *then*

$$\mathsf{RecognizesPkg}(R, \mathsf{Decode}(\mathsf{Compile}(S))).$$

Proof. By the flow-level round-trip theorem Definition 9208.17,

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S.$$

Substituting this equality into the recognition judgment gives

$$\mathsf{RecognizesPkg}(R, \mathsf{Decode}(\mathsf{Compile}(S))).$$

□

Corollary 9210.19 (Package recognition is invariant under compile-decode). *Lean4✓* `BEDC.GroundCompiler.Pack` `recognition invariant`

For every recognized package flow S, channel compilation followed by decoding preserves its package-recognition status.

Definition 9210.20 (Package-code equivalence). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.Packa`

For recognized package flows S and T, define

$$S \equiv_{\text{code}} T \;:\Longleftrightarrow\; \mathsf{Compile}(S) = \mathsf{Compile}(T).$$

Proposition 9210.21 (Package-code equivalence is raw-flow equality). *Lean4✓* `BEDC.GroundCompiler.PackageGe` `code equivalence is equality`

For recognized package flows S and T,

$$S \equiv_{\text{code}} T \quad\Longleftrightarrow\quad S = T.$$

Proof. If $S = T$, then $\mathsf{Compile}(S) = \mathsf{Compile}(T)$.

Conversely, suppose $\mathsf{Compile}(S) = \mathsf{Compile}(T)$. Applying Decode to both sides gives

$$\mathsf{Decode}(\mathsf{Compile}(S)) = \mathsf{Decode}(\mathsf{Compile}(T)).$$

By Definition 9208.17, the left side is S and the right side is T. Hence $S = T$. □

Corollary 9210.22 (Raw package flows are code-bijective). *Lean4✓* `BEDC.GroundCompiler.PackageGenerated.rau` `flows code bijective`

Recognized package flows, considered as raw event flows, are in bijection with their legal channel codes. This is not the same as saying classified package objects are bijective with codes: a package classifier may identify two raw package flows.

9210.4 Classified packages as quotients

Definition 9210.23 (Package classifier quotient). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGenerated.Pack`

If a package recognizer supplies a classifier relation $\sim_{\mathrm{pkg}}$ on recognized package flows, then the package classifier quotient relation records when two recognized package flows are assigned to the same classified package object:

$$\mathsf{PackageClassifierQuotient}(\sim_{\mathrm{pkg}}, S, T) \;:\Longleftrightarrow\; S \sim_{\mathrm{pkg}} T.$$

The projection

$$S \longmapsto \textit{the classified package object represented by } S$$

may therefore identify distinct raw package flows whenever $S \sim_{\mathrm{pkg}} T$ holds with $S \neq T$.

Proposition 9210.24 (Classified packages need not be code-bijective). *Lean4✓* `BEDC.GroundCompiler.PackageGen` `not code bijective`

If there exist recognized package flows $S \neq T$ with

$$S \sim_{\mathrm{pkg}} T,$$

then there exist two distinct recognized package codes whose decoded event flows are identified by the package classifier quotient. Thus the code-to-classified-package reading is not injective at the classified-object layer.

Proof. By Definition 9210.21, $S \neq T$ implies

$$\mathsf{Compile}(S) \neq \mathsf{Compile}(T).$$

But $S \sim_{\mathrm{pkg}} T$ implies

$$\mathsf{PackageClassifierQuotient}(\sim_{\mathrm{pkg}}, S, T).$$

Thus two distinct codes decode to raw flows identified by the classifier quotient. The code-to-classified-package reading is therefore not injective. □

Corollary 9210.25 (Raw package code and classified package object are different layers). *Lean4✓* `BEDC.GroundCompiler.PackageGenerated.raw code vs classified object layers`

Package codes are bijective with recognized raw package flows. Classified package objects are quotients of those flows:

$$\begin{array}{lcl} \textit{raw package flow} & \longleftrightarrow & \textit{package code,} \\ \textit{raw package flow} & \longrightarrow & \textit{classified package object.} \end{array}$$

The first relation is bijective; the second is generally many-to-one.

9210.5 Recognition from channel code

Definition 9210.26 (Package recognition from channel code). *Lean4 (definition)* `BEDC.GroundCompiler.PackageGen`

Given $c \in \mathsf{LegalZStream}$, define package recognition from code by the existence of a decoded event flow:

$$\mathsf{RecognizesPkgCode}(R, c) \;:\Longleftrightarrow\; \exists S \in \mathsf{EventFlow}, \mathsf{Decode}(c) = \mathsf{some}(S) \wedge \mathsf{RecognizesPkg}(R, S).$$

Thus package recognition from channel code always proceeds through decoding.

Proposition 9210.27 (No package recognition by channel substring)**.** `Lean4✓` `BEDC.GroundCompiler.PackageGene` `code recognition has decoded flow`

Every valid package-recognition judgment from channel code first supplies a decoded event flow:

$$\mathsf{RecognizesPkgCode}(R, c) \implies \exists S,\ \mathsf{Decode}(c) = \mathsf{some}(S) \wedge \mathsf{RecognizesPkg}(R, S).$$

Thus there is no valid package recognition rule that inspects arbitrary channel substrings without first decoding the channel stream into an event flow.

Proof. By Definition 9210.26, $\mathsf{RecognizesPkgCode}(R, c)$ is defined by the existence of a decoded event flow S with $\mathsf{Decode}(c) = \mathsf{some}(S)$ and $\mathsf{RecognizesPkg}(R, S)$. By Definition 9209.27, channel substrings are not source events. Package recognition is defined on event flows by Definition 9210.3; therefore valid package recognition from channel code must proceed through Decode. □

Example 9210.28 (Package recognition over a completion skeleton)**.** *Consider the event flow*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Its channel code is

$$011001100011010110101011100011.$$

A generated recognizer R may recognize this event flow as a completion-like package candidate, identifying

$0 \mid 00 \mid 000$	*finite source-prefix flow,*
01	*repeat-boundary witness,*
$011 \mid 100$	*raw/normal carry ledger,*
σ	*package-seal subflow.*

None of this occurs at the channel-substring layer. The channel code must first decode to

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

and only then can R recognize package roles.

Example 9210.29 (Package recognition over an additive / fold skeleton)**.** *Consider the event flow*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

Its channel code is

$$011001100101100101011010 0011.$$

A recognizer may identify

0	*unit-source event,*
00	*binary-repetition witness,*
001	*additive-recursion witness,*
$0011 \mid 0100$	*raw/normal fold ledger.*

The compiler itself does not call this flow addition, multiplication, or fold. Those names require a recognized package flow and, if exported as derived interfaces, the corresponding NameCert*, derived certificate, and ledger support.*

9210.6 Conservativity and derived-interface boundary

Theorem 9210.30 (Package-recognition conservativity)**.** Lean4✓ BEDC.GroundCompiler.PackageGenerated.packa recognition conservativity

Adding package recognition relations of the form

$$\mathsf{RecognizesPkg}(R, S)$$

does not add new finite-kernel marks, histories, mark-sameness rules, history-sameness rules, continuation rules, or derived-interface formation rules.

Proof. The relation $\mathsf{RecognizesPkg}(R, S)$ is defined over event flows $R, S \in \mathsf{EventFlow}$. It does not introduce raw marks beyond b0 and b1, construct new histories in the finite kernel, assert new msame, hsame, or psame judgments, or license derived interfaces by itself. A derived interface still requires the relevant NameCert fields, and an exported object still requires the acceptance discipline of Definition 9207.18. Therefore package recognition is conservative over the finite kernel. □

Corollary 9210.31 (Recognized package flow is weaker than derived interface)**.** Lean4✓ BEDC.GroundCompiler.Pa package weaker than derived interface

A derived-interface license for a package flow consists of package recognition together with recognized NameCert *flow, recognized* ClosureCert *flow, and accepted-object support. Such a license projects to package recognition:*

$$\mathsf{DerivedInterfaceLicense}(S) \Longrightarrow \mathsf{PkgFlow}(S).$$

Package recognition alone is only this projected component; it is not the whole derived-interface license.

Proof. By definition, $\mathsf{DerivedInterfaceLicense}(S)$ contains a generated package recognizer R with $\mathsf{RecognizesPkg}(R, S)$, together with the certificate and acceptance witnesses. Projecting the package-recognition component gives $\mathsf{PkgFlow}(S)$. □

Remark 9210.32 (Outlook)**.** *This chapter establishes that package structure is a recognition result inside an event flow, not a compiler input. The core relation is* $\mathsf{RecognizesPkg}(R, S)$ *with* $R, S \in \mathsf{EventFlow}$*: it preserves channel code, keeps package seals separate from channel terminators, requires ledger subflows for nontrivial package visibility, makes raw package flows code-bijective, treats classified packages as quotients, forces channel-code recognition through* Decode*, and remains conservative over the finite kernel. The same no-hidden-input rule applies to* NameCert *recognition: certificate fields are recognized as event-flow substructures rather than received as a host-level five-field object.*

Chapter 9211

NameCert Flow Recognition

chapter 9210 records that package structure is not a compiler input: it is recognized inside an event flow by a generated recognizer. This chapter applies the same no-hidden-input rule to NameCert.

The central rule is

$$\boxed{\text{NameCert is a recognized five-field certificate flow, not an input object.}}$$

The paper-side $\mathsf{NameCert}(N)$ template of Definition 9.6 is already a five-field certificate:

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The proposed name N is licensed only when all required fields are supplied, upward-arrow derived interfaces are not licensed without a NameCert, and limit-like interfaces carry ledger policy witnesses.

This chapter does not change the mathematical content of the NameCert discipline. It changes the input reading: the compiler does not receive a host object of the form

$$\mathsf{NameCert}(\mathsf{SourceSpec}, \mathsf{PatternSpec}, \mathsf{ClassifierSpec}, \mathsf{StabilityCert}, \mathsf{LedgerPolicy}).$$

It receives an event flow. A generated recognizer then recognizes source, pattern, classifier, stability, and ledger subflows inside the decoded source layer.

9211.1 NameCert recognition on event flows

Definition 9211.1 (Name candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenerated.NameCandi`

A name candidate flow *is an event flow*

$$N \in \mathsf{EventFlow}.$$

It represents the visible generated trace of a proposed name. It is not an external string label, host-level object name, theorem label, or registry entry.

Definition 9211.2 (NameCert candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenerated.NameC`

A NameCert candidate flow *is an event flow*

$$S \in \mathsf{EventFlow}.$$

It has not yet been recognized as a NameCert*. Thus* $S \in \mathsf{EventFlow}$ *does not imply* $S = \mathsf{NameCert}(N)$*.*

Definition 9211.3 (Generated NameCert recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenerated`

A generated NameCert recognizer *is an event flow*

$$R \in \mathsf{EventFlow}.$$

It is not a host parser, YAML schema, external grammar, Lean structure input, or human field table. Its public formal carrier is generated from the two raw marks and works on the source event layer.

Definition 9211.4 (NameCert recognition relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenerated.N`

The name-indexed NameCert *recognition relation is written*

$$\mathsf{RecognizesNameCert}(R, S, N), \qquad R, S, N \in \mathsf{EventFlow}.$$

It is read as: the generated recognizer flow R recognizes the event flow S as a NameCert *flow for the candidate name flow N.*

Remark 9211.5 (Relation to the unindexed recognition judgment)**.** *Definition 9207.14 used the unindexed judgment* $\mathsf{RecognizesNameCert}(R, S)$ *to record that S is recognized as a* NameCert *flow. The ternary notation of this chapter is the same source-layer recognition with the associated candidate name flow displayed explicitly. When the name parameter is irrelevant, the unindexed notation remains a harmless abbreviation.*

Remark 9211.6 (NameCert recognition is not NameCert input)**.** *The judgment* $\mathsf{RecognizesNameCert}(R, S, N)$ *does not mean that an external* NameCert *object has been supplied. It means that, in the decoded source event layer, R recognizes S as having the five-field certificate structure for the candidate name N. The direction is*

$$\mathsf{EventFlow} \longrightarrow \mathsf{RecognizedNameCertFlow},$$

not

$$\mathsf{NameCertObject} \longrightarrow \mathsf{EventFlow}.$$

Definition 9211.7 (Recognized NameCert flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenerated.Name`

Define

$$\mathsf{NameCertFlow}(S, N) \;:\Longleftrightarrow\; \exists R,\ \mathsf{RecognizesNameCert}(R, S, N).$$

Thus S is a recognized NameCert *flow for the candidate name N exactly when some generated recognizer flow recognizes it as the certificate flow for N.*

Principle 9211.8 (No external NameCert input)**.** *Lean4✓* `BEDC.GroundCompiler.NameCertGenerated.no external namecert input`

There is no formal compiler entry point

$$\mathsf{CompileNameCert}\colon \mathsf{NameCertObject} \longrightarrow \mathsf{LegalZStream}.$$

The only formal channel compiler remains

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \mathsf{LegalZStream}.$$

If S is a recognized NameCert *flow, its* NameCert *code is just* $\mathsf{Compile}(S)$.

Proposition 9211.9 (NameCert recognition does not change channel code)**.** Lean4 ✓ BEDC.GroundCompiler.NameC recognition preserves code

If $\mathsf{RecognizesNameCert}(R, S, N)$*, then the channel encoding of* S *is still* $\mathsf{Compile}(S)$*.* NameCert *recognition does not add, delete, or rewrite channel bits.*

Proof. The judgment $\mathsf{RecognizesNameCert}(R, S, N)$ is a source-level recognition relation. It says only that S is recognized by R as the NameCert flow for N. The channel encoding is supplied by Compile and depends only on the event flow S. Since the recognition relation does not alter S, it does not alter $\mathsf{Compile}(S)$. □

Corollary 9211.10 (Recognized NameCert code is not a separate code)**.** Lean4 ✓ BEDC.GroundCompiler.NameCert code not separate

If S *is a recognized* NameCert *flow, then*

$$\mathsf{NameCertCode}(S) := \mathsf{Compile}(S)$$

is the ordinary channel code of the same event flow, not the output of a separate NameCert *encoder.*

9211.2 Field subflows and seals

Definition 9211.11 (Recognized NameCert field subflows)**.** Lean4 (definition) BEDC.GroundCompiler.NameCertGene

If $\mathsf{RecognizesNameCert}(R, S, N)$*, then* R *may recognize subflows of* S *as the five certificate roles:*

source flow,
pattern flow,
classifier flow,
stability flow,
ledger flow.

Formally, this chapter uses the relations

$$\mathsf{NameCertSource}_R(S, S_0), \qquad \mathsf{NameCertPattern}_R(S, S_1), \qquad \mathsf{NameCertClassifier}_R(S, S_2),$$

$$\mathsf{NameCertStability}_R(S, S_3), \qquad \mathsf{NameCertLedger}_R(S, S_4).$$

These are not external field labels. They are subflow roles recognized by R *inside* S*.*

Remark 9211.12 (Field names are explanatory metalanguage)**.** *The names* SourceSpec*,* PatternSpec*,* ClassifierSpec*,* StabilityCert*, and* LedgerPolicy *are explanatory labels for the five paper-side roles. Formally, the roles must be recognized by the relations*

$$\mathsf{NameCertSource}_R, \quad \mathsf{NameCertPattern}_R, \quad \mathsf{NameCertClassifier}_R, \quad \mathsf{NameCertStability}_R, \quad \mathsf{NameCertLedger}_R$$

on event flows. Field names are not first-layer compiler inputs, and the five-field structure is not received as an external record.

Definition 9211.13 (NameCert-seal subflow)**.** Lean4 (definition) BEDC.GroundCompiler.NameCertGenerated.NameCe

If $\mathsf{RecognizesNameCert}(R, S, N)$ *and* R *recognizes a terminal subflow* σ *of* S *as expressing* NameCert *completion, then* σ *is a* NameCert-seal subflow*, written*

$$\mathsf{NameCertSeal}_R(S, \sigma).$$

The NameCert *seal is a source-level subflow, not the channel-level terminator* 11*.*

Proposition 9211.14 (NameCert seal is not the channel terminator). *Lean4✓ BEDC.GroundCompiler.NameCertGen seal is source subflow*

If $\mathsf{NameCertSeal}_R(S, \sigma)$*, then* σ *is a source-level subflow, not the channel-level terminator* 11*.*

Proof. The channel-level 11 is the event terminator of Definition 9208.2; it belongs to the LegalZStream encoding mechanism. A NameCert seal is a source-level subflow of an event flow recognized by R. By the layer-separation theorem Definition 9209.27, a channel substring cannot be used directly as a source event or source subflow. Hence the NameCert seal is not the channel terminator. □

Definition 9211.15 (Complete five-field recognition). *Lean4 (definition) BEDC.GroundCompiler.NameCertGenerated*

The judgment $\mathsf{RecognizesNameCert}(R, S, N)$ *has* complete five-field recognition *when there exist subflows* S_0, S_1, S_2, S_3, S_4 *and* σ *such that*

$$\mathsf{NameCertSource}_R(S, S_0), \qquad \mathsf{NameCertPattern}_R(S, S_1), \qquad \mathsf{NameCertClassifier}_R(S, S_2),$$

$$\mathsf{NameCertStability}_R(S, S_3), \qquad \mathsf{NameCertLedger}_R(S, S_4), \qquad \mathsf{NameCertSeal}_R(S, \sigma).$$

The displayed roles are recognized by R*, not supplied by external field tags.*

Principle 9211.16 (No NameCert without five recognized fields). *Lean4✓ BEDC.GroundCompiler.NameCertGenera namecert without five fields*

The name-indexed recognition relation

$$\mathsf{RecognizesNameCert}(R, S, N)$$

includes complete five-field recognition. In particular, a recognized certificate for a proposed name supplies source, pattern, classifier, stability, ledger, and seal subflows recognized inside the event flow. This is the event-flow form of the paper template of Definition 9.6: a proposed name is licensed only after all required fields are supplied.

Theorem 9211.17 (Incomplete NameCert flow does not license a name). *Lean4✓ BEDC.GroundCompiler.NameCert namecert does not license*

If every generated recognizer relation for (S, N) *fails complete five-field recognition, then* S *is not a* NameCert *flow licensing* N*.*

Proof. By Definition 9211.16, a recognized NameCert flow must have complete five-field recognition. If any required field subflow is missing, complete five-field recognition fails. Therefore S cannot serve as an admissible NameCert flow licensing N. □

Corollary 9211.18 (Visible name event is insufficient). *Lean4✓ BEDC.GroundCompiler.NameCertGenerated.visibl name event insufficient*

A visible candidate name event N *does not by itself license a derived interface. A recognized* NameCert *flow must supply the five certificate fields for* N*.*

9211.3 Name licensing

Definition 9211.19 (Name licensing by recognized flow). *Lean4 (definition) BEDC.GroundCompiler.NameCertGenera*

Define

$$\mathsf{LicensedName}(N) \;:\Longleftrightarrow\; \exists S,\ \mathsf{NameCertFlow}(S, N).$$

Thus the candidate name N *is licensed exactly when some event flow is recognized as its* NameCert *flow.*

Proposition 9211.20 (Licensing is event-flow based). *Lean4✓ BEDC.GroundCompiler.NameCertGenerated.licensi event flow based*

If $\mathsf{LicensedName}(N)$*, then there exists* $S \in \mathsf{EventFlow}$ *such that* $\mathsf{NameCertFlow}(S, N)$*.*

Proof. This is the content of Definition 9211.19, with the ambient event-flow condition supplied by Definition 9211.7. □

Theorem 9211.21 (Derived interfaces require recognized NameCert flow). *Lean4✓ BEDC.GroundCompiler.NameCe interfaces require flow*

If an upward-arrow derived interface $N^{\uparrow}$ *is licensed, then there exists a recognized* NameCert *flow* S *for* N*:*

$$N^{\uparrow} \text{ licensed} \implies \exists S,\ \mathsf{NameCertFlow}(S, N).$$

Proof. The formation rule of Definition 9.9 says that upward-arrow interfaces are not licensed without a NameCert. In the present compiler discipline, a NameCert object is not an external input; it must appear as a recognized event-flow certificate. Hence a licensed $N^{\uparrow}$ requires some S with $\mathsf{NameCertFlow}(S, N)$. □

Corollary 9211.22 (No upward-arrow interface from raw code alone). *Lean4✓ BEDC.GroundCompiler.NameCertGe upward from raw code alone*

A legal channel stream $c \in \mathsf{LegalZStream}$*, even though it decodes to an event flow, does not by itself license any* $N^{\uparrow}$*. Licensing requires a recognized* NameCert *flow for* N*, either on* $\mathsf{Decode}(c)$ *itself or on a recognized subflow of it.*

9211.4 Ledger completeness

Definition 9211.23 (Ledger-complete NameCert flow). *Lean4 (definition) BEDC.GroundCompiler.NameCertGenerate*

A recognized NameCert *flow is* ledger-complete *when it has a recognized ledger subflow*

$$\mathsf{NameCertLedger}_R(S, L)$$

and L *records the source material hidden, compressed, named, or stabilized by that* NameCert*.*

Principle 9211.24 (Certified derived interfaces carry ledger). *Lean4✓ BEDC.GroundCompiler.NameCertGenerated. derived carry ledger*

Every licensed derived interface is supported by a ledger-complete NameCert *flow. This is the event-flow reading of Definition 9.10: every certified derived interface carries a ledger policy witness.*

Theorem 9211.25 (NameCert without ledger is not admissible). *Lean4✓ BEDC.GroundCompiler.NameCertGenerat without ledger not admissible*

If no generated recognizer relation for (S, N) *admits a ledger subflow* L *such that*

$$\mathsf{NameCertLedger}_R(S, L),$$

then S *is not an admissible* NameCert *flow licensing* N*.*

Proof. The five-field NameCert template includes LedgerPolicy as a required field. By Definition 9211.24, a licensed derived interface must be supported by a ledger-complete NameCert flow. If no recognized ledger subflow exists, the recognition is incomplete and cannot license N. □

Corollary 9211.26 (Four fields without ledger are not enough). *Lean4✓ BEDC.GroundCompiler.NameCertGenerate fields without ledger not enough*

For a fixed generated recognizer R, even if source, pattern, classifier, and stability subflows are recognized inside S, the recognition is not complete five-field recognition when no ledger subflow is recognized by R.

9211.5 NameCert code and channel recognition

Definition 9211.27 (NameCert code). *Lean4 (definition) BEDC.GroundCompiler.NameCertGenerated.NameCertCode*

If S is a recognized NameCert *flow for N, define*

$$\mathsf{NameCertCode}(S, N) := \mathsf{Compile}(S).$$

This is not a new channel compiler. It is the existing Compile *applied to S.*

Proposition 9211.28 (NameCert code is injective on raw NameCert flows). *Lean4✓ BEDC.GroundCompiler.NameC code injective*

If S and T are recognized NameCert *flows and*

$$\mathsf{NameCertCode}(S, N) = \mathsf{NameCertCode}(T, M),$$

then $S = T$.

Proof. By Definition 9211.27,

$$\mathsf{NameCertCode}(S, N) = \mathsf{Compile}(S), \qquad \mathsf{NameCertCode}(T, M) = \mathsf{Compile}(T).$$

Equality of the displayed codes therefore gives $\mathsf{Compile}(S) = \mathsf{Compile}(T)$. By the channel bijection Definition 9208.19, Compile is injective on EventFlow. Hence $S = T$. □

Remark 9211.29 (Names may still classify together). *Definition 9211.28 says only that raw* NameCert *flows and their codes correspond one-to-one. A later classifier may still identify different candidate names or different certificate flows as equivalent; that is a classifier quotient and does not affect the raw-flow bijection.*

Definition 9211.30 (NameCert recognition from channel code). *Lean4 (definition) BEDC.GroundCompiler.NameCert*

For $c \in$ LegalZStream*, define*

$$\mathsf{RecognizesNameCertCode}(R, c, N) \;:\Longleftrightarrow\; \mathsf{RecognizesNameCert}(R, \mathsf{Decode}(c), N).$$

Thus NameCert *recognition from channel code first decodes to an event flow.*

Proposition 9211.31 (No NameCert recognition by channel substring). *Lean4✓ BEDC.GroundCompiler.NameCert namecert recognition by channel substring*

There is no valid rule that recognizes a NameCert *by inspecting channel substrings without first applying*

$$\mathsf{Decode}\colon \mathsf{LegalZStream} \to \mathsf{EventFlow}.$$

Proof. Definition 9209.27 proves that channel substrings are not source events. NameCert recognition is a source-level relation over event flows. Direct NameCert recognition on channel substrings would therefore operate on the wrong layer. Recognition must first decode. □

Theorem 9211.32 (Channel compilation preserves NameCert recognition)**.** *Lean4 ✓* `BEDC.GroundCompiler.NameC` `compilation preserves namecert recognition`

If $\mathsf{RecognizesNameCert}(R, S, N)$*, then*

$$\exists S', \quad \mathsf{Decode}(\mathsf{Compile}(S)) = \mathsf{some}(S') \quad \textit{and} \quad \mathsf{RecognizesNameCert}(R, S', N).$$

Proof. By the flow-level round trip Definition 9208.17,

$$\mathsf{Decode}(\mathsf{Compile}(S)) = \mathsf{some}(S).$$

Taking $S' = S$ preserves the same source-layer recognition judgment. □

Corollary 9211.33 (NameCert recognition is invariant under compile-decode)**.** *Lean4 ✓* `BEDC.GroundCompiler.Na` `recognition invariant`

For every recognized NameCert *flow, compiling to a channel stream and decoding back to an event flow preserves its* NameCert *recognition status.*

9211.6 Field soundness events

Definition 9211.34 (NameCert-source soundness event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenera`

If $\mathsf{RecognizesNameCert}(R, S, N)$ *and* R *recognizes source subflow* S_0*, then a* NameCert-source soundness event *is a subflow of* S *witnessing that* S_0 *supplies the generated material from which* N *is named. The relation is written*

$$\mathsf{NameCertSourceSound}_R(S, S_0).$$

Definition 9211.35 (NameCert-pattern soundness event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGener`

If R *recognizes pattern subflow* S_1*, then a* NameCert-pattern soundness event *is a subflow of* S *witnessing that* S_1 *is stable behavior observed in the source:*

$$\mathsf{NameCertPatternSound}_R(S, S_1).$$

Definition 9211.36 (NameCert-classifier soundness event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGene`

If R *recognizes classifier subflow* S_2*, then a* NameCert-classifier soundness event *is a subflow of* S *witnessing that* S_2 *organizes source material into the intended groups:*

$$\mathsf{NameCertClassifierSound}_R(S, S_2).$$

This is the event-flow reading of the role played by ClassifierSpec *in the paper template.*

Definition 9211.37 (NameCert-stability soundness event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGene`

If R *recognizes stability subflow* S_3*, then a* NameCert-stability soundness event *is a subflow of* S *witnessing that the source satisfies the relevant closure, reuse, descent, composition, or seal stability under the named behavior:*

$$\mathsf{NameCertStabilitySound}_R(S, S_3).$$

Definition 9211.38 (NameCert-ledger soundness event)**.** *Lean4 (definition)* `BEDC.GroundCompiler.NameCertGenera`

If R *recognizes ledger subflow* S_4*, then a* NameCert-ledger soundness event *is a subflow of* S *witnessing that* S_4 *records what the name hides and how the hidden material is tracked:*

$$\mathsf{NameCertLedgerSound}_R(S, S_4).$$

Definition 9211.39 (Sound recognized NameCert flow)**.** *Lean4 (definition)* *BEDC.GroundCompiler.NameCertGenerat*

A recognized NameCert *judgment* $\mathsf{RecognizesNameCert}(R, S, N)$ *is* sound *when there exist subflows* S_0, S_1, S_2, S_3, S_4 *such that*

$$\mathsf{NameCertSource}_R(S, S_0) \wedge \mathsf{NameCertSourceSound}_R(S, S_0),$$

$$\mathsf{NameCertPattern}_R(S, S_1) \wedge \mathsf{NameCertPatternSound}_R(S, S_1),$$

$$\mathsf{NameCertClassifier}_R(S, S_2) \wedge \mathsf{NameCertClassifierSound}_R(S, S_2),$$

$$\mathsf{NameCertStability}_R(S, S_3) \wedge \mathsf{NameCertStabilitySound}_R(S, S_3),$$

$$\mathsf{NameCertLedger}_R(S, S_4) \wedge \mathsf{NameCertLedgerSound}_R(S, S_4).$$

Theorem 9211.40 (Sound NameCert flow licenses candidate name)**.** *Lean4✓* *BEDC.GroundCompiler.NameCertGene namecert flow licenses name*

If $\mathsf{RecognizesNameCert}(R, S, N)$ *and this recognition is sound, then*

$$\mathsf{LicensedName}(N).$$

Proof. By Definition 9211.39, S contains recognized and sound source, pattern, classifier, stability, and ledger subflows. These are the event-flow presentations of the five required fields in Definition 9.6. Therefore the candidate name N is supported by a recognized NameCert flow, so $\mathsf{LicensedName}(N)$ by Definition 9211.19. □

Corollary 9211.41 (Derived interface licensing is event-flow mediated)**.** *Lean4✓* *BEDC.GroundCompiler.NameCertG licensing mediated by sound flow*

If $N^{\uparrow}$ *is licensed, the licensing can be traced to a sound recognized* NameCert *flow. It is not produced by an external* NameCert *object.*

9211.7 Conservativity and examples

Theorem 9211.42 (NameCert flow recognition is conservative over the finite kernel)**.** *Lean4✓* *BEDC.GroundCompiler.NameCertGenerated.namecert flow recognition conservativity*

Adding relations of the form

$$\mathsf{RecognizesNameCert}(R, S, N)$$

and the associated field-recognition relations does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.

Proof. These recognition relations assert structure over event flows. They do not generate new b0 or b1 marks, add msame or hsame rules, change Ext or Cont, or turn a candidate name into a kernel object by itself. Derived-interface licensing still requires a recognized and sound NameCert flow. Hence this recognition layer is conservative over the finite kernel. □

Corollary 9211.43 (Recognized NameCert flow does not change source events)**.** *Lean4✓* *BEDC.GroundCompiler.Na does not change source events*

If S *is recognized as a* NameCert *flow, then some generated recognizer* R *witnesses the* NameCert *recognition relation over that same event flow* S*. Recognition is a relation over an existing event flow, not a rewrite operation.*

Example 9211.44 ($\mathsf{AxisNat}^{\uparrow}$-style NameCert flow)**.** *An axis-style* NameCert *flow may have recognized subflows*

source flow	b0-*emission events,*
pattern flow	*repeated* b0-*extension events,*
classifier flow	hsame *restricted to the* b0-*spine,*
stability flow	*closure under* E0 *and continuation events,*
ledger flow	*each visible axis token records source-history events.*

These are not external fields. They must be recognized inside an event flow.

Example 9211.45 ($\mathsf{FullAxis}^{\uparrow}$ NameCert flow)**.** *A* $\mathsf{FullAxis}^{\uparrow}$ NameCert *flow may have recognized subflows*

source flow	*finite* b0-*spine prefixes,*
pattern flow	*every prefix continues to a later prefix,*
classifier flow	*prefix-compatible and tail-compatible classification,*
stability flow	*directedness, monotonicity, and prefix closure,*
ledger flow	*all finite prefixes retained as ledger.*

*The reading is not "*b0b1 *is an infinite object." The reading is that a full-axis certificate over generated finite prefixes licenses* $\mathsf{FullAxis}^{\uparrow}$ *while retaining the finite-prefix ledger.*

Example 9211.46 ($\mathsf{ZNormal}^{\uparrow}$ NameCert flow)**.** *A* $\mathsf{ZNormal}^{\uparrow}$ NameCert *flow may have recognized subflows*

source flow	*observation histories with boundary marks,*
pattern flow	*no adjacent full-boundary marks,*
classifier flow	*histories related by Zeckendorf carry classify together,*
stability flow	*carry preserves represented observation content,*
ledger flow	*the carry ledger records the pre-normal form.*

The source-level display $011 \rightsquigarrow 100$ *is therefore a carry classifier with ledger support, not ordinary equality and not a channel substring rewrite.*

Example 9211.47 ($\mathsf{DimLift}^{\uparrow}$ NameCert flow)**.** *A* $\mathsf{DimLift}^{\uparrow}$ NameCert *flow may have recognized subflows*

source flow	*overflow traces, including the event* 011*,*
pattern flow	*Zeckendorf overflow and carry behavior,*
classifier flow	*overflow trace classified with the lifted normal form,*
stability flow	*lift preserves finite-prefix observations,*
ledger flow	*old axis, full-axis seal, and overflow witness.*

The lifted name is licensed by the recognized certificate flow and its ledger, not by the overflow event alone.

Remark 9211.48 (Outlook)**.** *This chapter establishes the no-hidden-input reading of* NameCert*: a* NameCert *is not received as a five-field object, but recognized as a five-field certificate flow inside* EventFlow*. The core judgment is* $\mathsf{RecognizesNameCert}(R, S, N)$*, where* R *is a generated recognizer flow,* S *is the candidate certificate event flow, and* N *is the candidate name flow. The judgment preserves channel code, separates certificate seals from channel terminators, requires five recognized fields and a ledger, keeps raw* NameCert *flows code-injective, forces channel-code recognition*

through Decode*, licenses names only through sound recognized certificate structure, and remains conservative over the finite kernel. The same discipline applies next to derived certificates and acceptance gates: mature object code is accepted only through recognized certificate and gate flows, not as a primitive input object.*

Chapter 9212

DerivCert and AcceptanceGate Flow Recognition

chapter 9211 records the no-hidden-input reading of NameCert: a naming certificate is not supplied as an external five-field object, but recognized as a five-field certificate flow inside EventFlow. This chapter applies the same rule to derivation certificates, acceptance gates, strength readings, and accepted export.

The governing rules are

$$\boxed{\text{DerivCert is a recognized six-field export certificate flow, not an input object,}}$$

and

$$\boxed{\text{AcceptGate is the simultaneous recognition of NameCert and derivation-certificate flows.}}$$

The paper-side acceptance discipline says that a mature object is not exported merely because it appears in the manuscript or in a codebase. It must pass a gate whose ingredients include a naming certificate and an export certificate at a stated strength. In the ground-compiler layer, those ingredients appear only as recognized event-flow structure.

9212.1 Strength and DerivCert recognition

Definition 9212.1 (Strength event flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.StrengthEventFlow`

A strength event flow *is an event flow*

$$s \in \mathsf{EventFlow}$$

recognized by a generated recognizer as representing a derivation strength.

For exposition, this chapter speaks of four strength readings

$$\text{seed}, \qquad \text{paperCert}, \qquad \text{checkedCert}, \qquad \text{bridgeCert},$$

ordered by

$$\text{seed} < \text{paperCert} < \text{checkedCert} < \text{bridgeCert}.$$

These names are metalanguage. They are not host enum inputs. The corresponding strength roles must be recognized from event flows.

Definition 9212.2 (Generated strength recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.Generated`
A generated strength recognizer *is an event flow*

$$R_s \in \mathsf{EventFlow}.$$

It supports source-layer judgments of the form

$$\mathsf{RecognizesStrength}(R_s, s, \sigma),$$

where s is the strength event flow and σ is the recognized strength role. The displayed labels seed*,* paperCert*,* checkedCert*, and* bridgeCert *are explanatory readings of such roles, not primitive compiler inputs.*

Definition 9212.3 (DerivCert candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.DerivCertCand`
A DerivCert candidate flow *is an event flow*

$$D \in \mathsf{EventFlow}.$$

It has not yet been recognized as a derivation certificate. Thus $D \in \mathsf{EventFlow}$ does not imply $D = \mathrm{DerivCert}(N, s)$.

Definition 9212.4 (Generated DerivCert recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerate`
A generated DerivCert recognizer *is an event flow*

$$R_D \in \mathsf{EventFlow}.$$

It is not a host parser, Lean structure, YAML schema, or external record constructor. Its public formal carrier is generated from the two raw marks and acts on event flows.

Definition 9212.5 (DerivCert recognition relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.`
The derivation-certificate recognition relation is written

$$\mathsf{RecognizesDerivCert}(R_D, D, N, s), \qquad R_D, D, N, s \in \mathsf{EventFlow}.$$

It is read as: the generated recognizer flow R_D recognizes the event flow D as a DerivCert *flow for candidate name flow N at strength flow s.*

Remark 9212.6 (DerivCert recognition is not DerivCert input)**.** *The judgment* $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ *does not mean that an external* DerivCert *object has been supplied. It says that, in the decoded source event layer, R_D recognizes D as carrying the six export-certificate roles for N at strength s. The direction is*

$$\mathsf{EventFlow} \longrightarrow \mathsf{RecognizedDerivCertFlow},$$

not

$$\mathsf{DerivCertObject} \longrightarrow \mathsf{EventFlow}.$$

Definition 9212.7 (Recognized DerivCert flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Der`
Define

$$\mathsf{DerivCertFlow}(D, N, s) \;:\Longleftrightarrow\; \exists R_D,\ \mathsf{RecognizesDerivCert}(R_D, D, N, s).$$

Thus D is a recognized derivation-certificate flow for N at strength s exactly when some generated recognizer recognizes it in that role.

Principle 9212.8 (No external DerivCert input). *Lean4✓ BEDC.GroundCompiler.DerivCertGenerated.no external derivcert input*

There is no formal compiler entry point

$$\mathsf{CompileDerivCert}\colon \mathsf{DerivCertObject} \longrightarrow \mathsf{LegalZStream}.$$

The only formal channel compiler remains

$$\mathsf{Compile}\colon \mathsf{EventFlow} \longrightarrow \mathsf{LegalZStream}.$$

If D is a recognized derivation-certificate flow, its code is just $\mathsf{Compile}(D)$.

Proposition 9212.9 (DerivCert recognition does not change channel code). *Lean4✓ BEDC.GroundCompiler.Deriv recognition preserves code*

If $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$*, then the channel encoding of D is still* $\mathsf{Compile}(D)$*. Derivation-certificate recognition does not add, delete, or rewrite channel bits.*

Proof. The judgment $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ is a source-level recognition relation. It says only that D is recognized by R_D as the derivation-certificate flow for N at s. The channel encoding is supplied by $\mathsf{Compile}$ and depends only on the event flow D. Since recognition does not alter D, it does not alter $\mathsf{Compile}(D)$. □

Corollary 9212.10 (Recognized DerivCert code is not a separate code). *Lean4✓ BEDC.GroundCompiler.DerivCert code not separate*

If D is a recognized derivation-certificate flow, then

$$\mathsf{DerivCertCode}(D, N, s) := \mathsf{Compile}(D)$$

is the ordinary channel code of D, not the output of a separate derivation-certificate encoder.

9212.2 Six fields and field soundness

Definition 9212.11 (Recognized DerivCert field subflows). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGene*

If $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$*, then R_D may recognize subflows of D as the six export-certificate roles:*

source flow,
classifier flow,
exactness flow,
ledger flow,
stability flow,
strength flow.

Formally, this chapter uses

$$\mathsf{DerivSource}_{R_D}(D, D_0), \quad \mathsf{DerivClassifier}_{R_D}(D, D_1), \quad \mathsf{DerivExactness}_{R_D}(D, D_2),$$

$$\mathsf{DerivLedger}_{R_D}(D, D_3), \quad \mathsf{DerivStability}_{R_D}(D, D_4), \quad \mathsf{DerivStrength}_{R_D}(D, D_5).$$

These roles are recognized in D by R_D, not supplied as external field labels.

Remark 9212.12 (DerivCert field names are explanatory metalanguage)**.** *The field names source, classifier, exactness, ledger, stability, and strength are explanatory metalanguage. Formally, the roles are recognized by* $\mathsf{DerivSource}_{R_D}$, $\mathsf{DerivClassifier}_{R_D}$, $\mathsf{DerivExactness}_{R_D}$, $\mathsf{DerivLedger}_{R_D}$, $\mathsf{DerivStability}_{R_D}$, *and* $\mathsf{DerivStrength}_{R_D}$ *on event flows. The six-field structure is not received as an external record.*

Definition 9212.13 (DerivCert-seal subflow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Deriv`

If $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ *and* R_D *recognizes a terminal subflow* σ_D *of* D *as expressing derivation-certificate completion, then* σ_D *is a* DerivCert-seal subflow*, written*

$$\mathsf{DerivCertSeal}_{R_D}(D, \sigma_D).$$

The seal is a source-level subflow, not the channel-level terminator 11*.*

Definition 9212.14 (Complete six-field DerivCert recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.DerivCer`

The judgment $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ *has* complete six-field recognition *when there exist subflows* $D_0, D_1, D_2, D_3, D_4, D_5$ *and* σ_D *such that*

$$\mathsf{DerivSource}_{R_D}(D, D_0), \quad \mathsf{DerivClassifier}_{R_D}(D, D_1), \quad \mathsf{DerivExactness}_{R_D}(D, D_2),$$

$$\mathsf{DerivLedger}_{R_D}(D, D_3), \quad \mathsf{DerivStability}_{R_D}(D, D_4), \quad \mathsf{DerivStrength}_{R_D}(D, D_5), \quad \mathsf{DerivCertSeal}_{R_D}(D, \sigma_D).$$

Proposition 9212.15 (DerivCert seal is not the channel terminator)**.** *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGe` `seal is source subflow`

If $\mathsf{DerivCertSeal}_{R_D}(D, \sigma_D)$*, then* σ_D *is a source-level subflow and not the channel-level terminator* 11*.*

Proof. The channel-level 11 is the event terminator of Definition 9208.2; it belongs to the LegalZStream encoding mechanism. A derivation-certificate seal is a source-level subflow recognized by R_D. By the layer-separation theorem Definition 9209.27, a channel substring cannot be used directly as a source event or source subflow. Hence the derivation-certificate seal is not the channel terminator. □

Principle 9212.16 (No DerivCert without six recognized fields)**.** *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenera` `derivcert without six fields`

A recognized derivation-certificate flow must have complete six-field recognition: source, classifier, exactness, ledger, stability, strength, and seal subflows are all required. This is the event-flow form of the six export-certificate roles.

Theorem 9212.17 (Incomplete DerivCert flow does not support export)**.** *Lean4 ✓* `BEDC.GroundCompiler.DerivCe` `derivcert does not support export`

If an event flow D *lacks any one of the recognized source, classifier, exactness, ledger, stability, or strength field subflows, then* D *cannot serve as a recognized derivation-certificate flow for* N *at* s*.*

Proof. By Definition 9212.16, a recognized derivation-certificate flow must have complete six-field recognition. Missing any required field prevents complete recognition. Therefore $\mathsf{DerivCertFlow}(D, N, s)$ is not established. □

Corollary 9212.18 (Strength flow alone is insufficient)**.** *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenerated.stre` `flow alone insufficient`

Even if an event flow is recognized as a checkedCert *or* bridgeCert *strength flow, it does not form a derivation-certificate flow without recognized source, classifier, exactness, ledger, and stability fields.*

Definition 9212.19 (DerivCert field soundness). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Der`

The field soundness relations for a recognized derivation-certificate flow are:

$$\mathsf{DerivSourceSound}_{R_D}(D, D_0, N), \qquad \mathsf{DerivClassifierSound}_{R_D}(D, D_1, N),$$

$$\mathsf{DerivExactnessSound}_{R_D}(D, D_2, N), \qquad \mathsf{DerivLedgerSound}_{R_D}(D, D_3, N),$$

$$\mathsf{DerivStabilitySound}_{R_D}(D, D_4, N), \qquad \mathsf{DerivStrengthSound}_{R_D}(D, D_5, s).$$

They witness, respectively, that the source field names the BEDC layer carrying N, the classifier organizes the intended carriers, exactness respects the relevant msame, hsame, *or* psame *relation on the cited layer, the ledger cites the required gap policy or an explicit absence statement, the stability field cites the relevant stability certificate, and the strength field matches the claimed strength flow s.*

Definition 9212.20 (Sound recognized DerivCert flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenera`

A recognized judgment $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ *is* sound *when there exist subflows $D_0, D_1, D_2, D_3, D_4, D_5$ such that*

$$\mathsf{DerivSource}_{R_D}(D, D_0) \wedge \mathsf{DerivSourceSound}_{R_D}(D, D_0, N),$$

$$\mathsf{DerivClassifier}_{R_D}(D, D_1) \wedge \mathsf{DerivClassifierSound}_{R_D}(D, D_1, N),$$

$$\mathsf{DerivExactness}_{R_D}(D, D_2) \wedge \mathsf{DerivExactnessSound}_{R_D}(D, D_2, N),$$

$$\mathsf{DerivLedger}_{R_D}(D, D_3) \wedge \mathsf{DerivLedgerSound}_{R_D}(D, D_3, N),$$

$$\mathsf{DerivStability}_{R_D}(D, D_4) \wedge \mathsf{DerivStabilitySound}_{R_D}(D, D_4, N),$$

$$\mathsf{DerivStrength}_{R_D}(D, D_5) \wedge \mathsf{DerivStrengthSound}_{R_D}(D, D_5, s).$$

Theorem 9212.21 (Sound DerivCert recognition establishes DerivCertFlow). *Lean4 ✓* `BEDC.GroundCompiler.Der` `derivcert recognition establishes flow`

If $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$ *and this recognition is sound, then*

$$\mathsf{DerivCertFlow}(D, N, s).$$

Proof. By Definition 9212.20, D contains recognized and sound source, classifier, exactness, ledger, stability, and strength subflows. These are exactly the event-flow presentations of the six derivation-certificate roles. Hence D is a recognized derivation-certificate flow for N at s. □

9212.3 Acceptance flows

Definition 9212.22 (Accepted flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.AcceptedFlow`

For $N, s \in$ EventFlow*, an event flow A is an* accepted flow *for N at s when there exist event flows C, D and recognizers R_C, R_D such that*

$$\mathsf{RecognizesNameCert}(R_C, C, N), \qquad \mathsf{RecognizesDerivCert}(R_D, D, N, s),$$

the NameCert *recognition is sound in the sense of Definition 9211.39, the derivation-certificate recognition is sound in the sense of Definition 9212.20, and A is recognized as containing the* NameCert *flow C, the derivation-certificate flow D, and an acceptance-seal subflow. This is written*

$$\mathsf{AcceptFlow}(A, N, s).$$

The contained NameCert *flow, derivation-certificate flow, and acceptance-seal subflow are nonempty event flows.*

Definition 9212.23 (Flow-level acceptance gate). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Ac`

Define the flow-level acceptance gate by

$$\mathsf{AcceptGateFlow}(N, s) \;:\Longleftrightarrow\; \exists A,\ \mathsf{AcceptFlow}(A, N, s).$$

This is the event-flow version of the gate reading

$$\mathrm{AcceptGate}(N, s) := \mathsf{NameCert}(N) \wedge \mathrm{DerivCert}(N, s).$$

Theorem 9212.24 (Accepted flow requires NameCert-flow and DerivCert-flow). *Lean4✓* `BEDC.GroundCompiler.D` `requires namecert derivcert`

If $\mathsf{AcceptGateFlow}(N, s)$*, then there exist* C, D *such that*

$$\mathsf{NameCertFlow}(C, N) \qquad \textit{and} \qquad \mathsf{DerivCertFlow}(D, N, s).$$

Proof. By Definition 9212.23, there is an A with $\mathsf{AcceptFlow}(A, N, s)$. By Definition 9212.22, that accepted flow contains a sound recognized $\mathsf{NameCert}$ flow C and a sound recognized derivation-certificate flow D. Hence $\mathsf{NameCertFlow}(C, N)$ and $\mathsf{DerivCertFlow}(D, N, s)$. □

Corollary 9212.25 (Code existence does not imply acceptance). *Lean4✓* `BEDC.GroundCompiler.DerivCertGenera` `existence not acceptance`

A legal channel stream $c \in \mathsf{LegalZStream}$ *does not by itself imply* $\mathsf{RecognizesAcceptanceCode}(c, N, s)$*. Acceptance from code requires decoding and then recognizing nonempty sound* $\mathsf{NameCert}$*, nonempty sound derivation-certificate, and nonempty acceptance-seal structure in the decoded event flow or its recognized subflows.*

Proof. Membership $c \in \mathsf{LegalZStream}$ says only that c decodes to an event flow. Acceptance-code recognition requires additional source-layer recognition judgments on the decoded flow. The empty channel stream is legal because it decodes to the empty event flow, but the empty event flow cannot contain a nonempty $\mathsf{NameCert}$ flow, derivation-certificate flow, and acceptance-seal subflow. Hence channel legality alone does not entail acceptance-code recognition. □

Definition 9212.26 (Accepted object code). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Accepte`

If $\mathsf{AcceptFlow}(A, N, s)$*, define*

$$\mathsf{AcceptedObjectCode}(A, N, s) := \mathsf{Compile}(A).$$

The code is accepted because its decoded event flow is recognized as an acceptance flow, not because it has a special bit pattern.

Proposition 9212.27 (Accepted object code is injective on acceptance flows). *Lean4✓* `BEDC.GroundCompiler.Der` `object code injective`

If

$$\mathsf{AcceptedObjectCode}(A, N, s) = \mathsf{AcceptedObjectCode}(B, M, t),$$

then $A = B$*.*

Proof. By Definition 9212.26, the equality is $\mathsf{Compile}(A) = \mathsf{Compile}(B)$. By the channel bijection Definition 9208.19, $\mathsf{Compile}$ is injective on $\mathsf{EventFlow}$. Hence $A = B$. □

Remark 9212.28 (Accepted flows and accepted objects). *Definition 9212.27 concerns raw acceptance flows. It does not say that classified mature objects are code-bijective. If two accepted flows are identified by a classifier or bridge, object-level sameness is a quotient and may be many-to-one.*

9212.4 Recognition from code and round trips

Definition 9212.29 (Acceptance recognition from channel code). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCe`

For $c \in \mathsf{LegalZStream}$*, define*

$$\mathsf{RecognizesAcceptanceCode}(c, N, s) \;:\Longleftrightarrow\; \exists A,\ \mathsf{Decode}(c) = \mathsf{some}(A) \;\wedge\; \mathsf{AcceptFlow}(A, N, s).$$

Thus accepted-object recognition from channel code first decodes to an event flow.

Proposition 9212.30 (No acceptance recognition by channel substring). *Lean4✓* `BEDC.GroundCompiler.DerivCer` `acceptance recognition by channel substring`

Any valid recognition of an accepted object from channel code must first exhibit an event flow decoded by

$$\mathsf{Decode}\colon \mathsf{LegalZStream} \to \mathsf{EventFlow}.$$

Equivalently, a judgment $\mathsf{RecognizesAcceptanceCode}(c, N, s)$ *factors through some* A *such that* $\mathsf{Decode}(c) = \mathsf{some}(A)$ *and* $\mathsf{AcceptFlow}(A, N, s)$*.*

Proof. Definition 9209.27 proves that channel substrings are not source events. Acceptance-flow recognition is a source-level relation over event flows. Direct accepted-object recognition on channel substrings would therefore operate on the wrong layer. □

Theorem 9212.31 (Channel compilation preserves acceptance recognition). *Lean4✓* `BEDC.GroundCompiler.Deriv` `compilation preserves acceptance`

If $\mathsf{AcceptFlow}(A, N, s)$*, then the channel code* $\mathsf{Compile}(A)$ *is recognized as acceptance code:*

$$\mathsf{RecognizesAcceptanceCode}(\mathsf{Compile}(A), N, s).$$

Proof. By the flow-level round trip Definition 9208.17,

$$\mathsf{Decode}(\mathsf{Compile}(A)) = A.$$

Substitution in $\mathsf{AcceptFlow}(A, N, s)$ gives the result. □

Corollary 9212.32 (Acceptance recognition is invariant under compile-decode). *Lean4✓* `BEDC.GroundCompiler.D` `recognition invariant`

For every accepted flow, compiling to a channel stream and decoding back to an event flow preserves its acceptance status.

9212.5 Strength monotonicity and reuse

Definition 9212.33 (Strength monotonicity at flow level). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGene`

The flow-level acceptance gate is monotone in strength *when, for strength flows* $s' \le s$*,*

$$\mathsf{AcceptGateFlow}(N, s) \Longrightarrow \mathsf{AcceptGateFlow}(N, s').$$

At event-flow level, the comparison $s' \le s$ *is witnessed by a recognized derivation-certificate flow for* N *at* s' *that occurs as a subflow of the same accepted flow that witnesses acceptance at* s*.*

Theorem 9212.34 (Flow-level gate is monotone in strength). *Lean4✓ BEDC.GroundCompiler.DerivCertGenerated monotone strength*

Assume recognized strength flows are ordered by

$$\text{seed} < \text{paperCert} < \text{checkedCert} < \text{bridgeCert},$$

and each higher-strength derivation-certificate flow contains the corresponding lower-strength derivation-certificate flow as a recognized subflow. Then

$$\mathsf{AcceptFlow}(A, N, s) \wedge \exists D' \left(\mathsf{DerivCertFlow}(D', N, s') \wedge D' \neq \epsilon \wedge \mathsf{DerivCertSourceSubflow}(D', A)\right) \Longrightarrow \mathsf{AcceptGateFlow}(N$$

Proof. From $\mathsf{AcceptFlow}(A, N, s)$, take the recognized $\mathsf{NameCert}$ flow, derivation-certificate flow, and acceptance-seal subflow contained in A. The naming certificate and seal are independent of the strength comparison. By the hypothesis on strength nesting, A also contains a nonempty recognized derivation-certificate flow D' for N at s'. Replacing the strength-s derivation-certificate component by D' gives an accepted flow for N at s', hence $\mathsf{AcceptGateFlow}(N, s')$. □

Definition 9212.35 (Reuse flow). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.ReuseFlow*

A reuse flow *is an event flow U recognized as saying that accepted flows for $N_1, \ldots, N_k$ are reused to construct M, that transport respects the relevant classifiers, and that M supplies its own certificate fields. It is written*

$$\mathsf{ReuseFlow}(U; N_1, \ldots, N_k; M).$$

Theorem 9212.36 (Reuse closure at flow level). *Lean4✓ BEDC.GroundCompiler.DerivCertGenerated.reuse closure at flow level*

Let $I = ((N_1, s_1), \ldots, (N_k, s_k))$ be a finite list of accepted input flows. If there is a reuse flow

$$\mathsf{ReuseFlow}(U; I; M, s)$$

whose recognized transport is classifier-respecting and whose target component supplies an accepted flow for M at strength s, then

$$\mathsf{AcceptGateFlow}(M, s).$$

Proof. Each pair (N_i, s_i) in I has an accepted flow by the input component of $\mathsf{ReuseFlow}$. The reuse flow supplies classifier-respecting transport from the accepted inputs to the target construction and includes a target acceptance flow for M at strength s. Projecting that target component gives $\mathsf{AcceptGateFlow}(M, s)$. □

Corollary 9212.37 (Reuse does not erase target certificate obligations). *Lean4✓ BEDC.GroundCompiler.DerivCert does not erase obligations*

Even if all inputs N_i are accepted, any accepted target M must still supply its own recognized $\mathsf{NameCert}$ *flow and its own recognized derivation-certificate flow:*

$$\mathsf{AcceptGateFlow}(M, s) \Longrightarrow \exists C, D \left(\mathsf{NameCertFlow}(C, M) \wedge \mathsf{DerivCertFlow}(D, M, s)\right).$$

Reuse cannot replace M's derivation-certificate flow.

Definition 9212.38 (Retroactive promotion). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.Retroa*

A target M is retroactively promoted *when its accepted strength is claimed to increase merely because its dependencies are later accepted at higher strength, while M itself has no recognized higher-strength flow.*

Principle 9212.39 (No retroactive promotion). Lean4✓ BEDC.GroundCompiler.DerivCertGenerated.no retroactive promotion

A flow accepted at strength s does not automatically become accepted at a higher strength $s' > s$ merely because its dependencies are later accepted at higher strength. Promoting M requires a recognized higher-strength flow for M itself.

9212.6 Conservativity and examples

Theorem 9212.40 (Accepted flow recognition is conservative over the finite kernel). Lean4✓ BEDC.GroundCompiler.DerivCertGenerated.accepted flow recognition conservativity

Adding RecognizesDerivCert, AcceptFlow, AcceptGateFlow, *and the associated field-recognition relations does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.*

Proof. These relations assert certificate recognition, export recognition, and bookkeeping discipline over event flows. They do not generate new b0 or b1 marks, add msame, hsame, or psame rules, change Ext or Cont, or turn a mature object into a kernel primitive. Export still requires recognized NameCert and derivation-certificate flows. Hence the recognition layer is conservative over the finite kernel. □

Corollary 9212.41 (Accepted object code is weaker than object equality). Lean4✓ BEDC.GroundCompiler.DerivCe object code weaker than object equality

If two accepted object codes are identified by a classifier or bridge as the same mature object, that does not imply equality of their raw acceptance flows. Equivalently, a classifier may identify distinct accepted flows by a coarser event-flow reading. Raw-flow equality is controlled by channel code equality; mature-object sameness is controlled by classifier and bridge structure.

Example 9212.42 ($\mathsf{AxisNat}^{\uparrow}$ acceptance flow). *An* $\mathsf{AxisNat}^{\uparrow}$ *acceptance flow contains a* NameCert *flow with* b0*-emission source histories, repeated* b0*-extension pattern,* hsame *restricted to the* b0*-spine, closure under* E0 *and continuation, and a source-history ledger. It also contains a derivation-certificate flow naming the layer carrying* $\mathsf{AxisNat}^{\uparrow}$*, the axis classifier, exactness for the relevant sameness relation, the axis ledger, the stability certificate, and the recognized strength flow. The channel code is* Compile *of that whole accepted flow, not a local event such as* 0*,* 00*, or* 01*.*

Example 9212.43 ($\mathsf{FullAxis}^{\uparrow}$ acceptance flow). *A* $\mathsf{FullAxis}^{\uparrow}$ *acceptance flow recognizes finite* b0*-spine prefixes as source, later-prefix continuation as pattern, prefix-compatible and tail-compatible classification, directedness and prefix closure as stability, and all finite prefixes as ledger. Its derivation-certificate flow still supplies source layer, classifier exactness, ledger citation, stability citation, and strength. The visible overflow skeleton alone does not export* $\mathsf{FullAxis}^{\uparrow}$*.*

Example 9212.44 ($\mathsf{DimLift}^{\uparrow}$ acceptance flow). *A* $\mathsf{DimLift}^{\uparrow}$ *acceptance flow recognizes overflow traces such as* 011*, Zeckendorf overflow and carry behavior, the lifted normal-form classifier, finite-prefix preservation, and the ledger consisting of old axis, full-axis seal, and overflow witness. Its derivation-certificate flow then recognizes the six export roles. The event* 011 *alone is not the lifted object.*

Remark 9212.45 (Outlook). *This chapter establishes that derivation certificates and acceptance gates are not compiler inputs. The core judgments are* $\mathsf{RecognizesDerivCert}(R_D, D, N, s)$*,* $\mathsf{AcceptFlow}(A, N, s)$*, and* $\mathsf{AcceptGateFlow}(N, s)$*. Derivation-certificate recognition preserves channel code, requires six recognized fields, rejects strength-only export, and is conservative over the*

finite kernel. Acceptance requires both recognized NameCert *and derivation-certificate flows, code existence alone does not imply acceptance, accepted object code is injective only on raw acceptance flows, recognition from code must pass through* Decode*, strength monotonicity and reuse are flow-level disciplines, and no higher-strength export appears without a recognized higher-strength flow for the target itself. The next ground-compiler chapter applies the same rule to theorem flow recognition: theorems are not external theorem labels or proof objects, but recognized statement, dependency, proof, certificate, ledger, status, and seal flows.*

Chapter 9213

Theorem Flow Recognition

chapter 9212 records the no-hidden-input reading of derivation certificates and acceptance gates: a DerivCert is recognized as a six-field export-certificate flow, and an acceptance gate is the simultaneous recognition of naming and derivation-certificate flows. This chapter applies the same rule to theorems themselves.

The governing rule is

$$\boxed{\text{a theorem is recognized event-flow structure, not an external label.}}$$

The compiler therefore has no formal entry point for theorem objects, theorem labels, proposition objects, proof trees, Lean theorem names, external theorem identifiers, external statement strings, or host proof checkers. A theorem enters this layer only as an event flow recognized by a generated theorem recognizer.

9213.1 Theorem recognition

Definition 9213.1 (Theorem candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.Theorem`

A theorem candidate flow *is an event flow*

$$T \in \mathsf{EventFlow}.$$

It has not yet been recognized as a theorem. Thus $T \in \mathsf{EventFlow}$ *does not imply that* T *is a theorem, has a proof, carries a status, or satisfies any certificate or acceptance condition.*

Definition 9213.2 (Generated theorem recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.Ge`

A generated theorem recognizer *is an event flow*

$$R_T \in \mathsf{EventFlow}.$$

It is not a host-level proof checker, theorem table, external parser, proof-object constructor, or theorem manifest. Its public formal carrier is itself generated from the two raw marks and acts on decoded event flows.

Definition 9213.3 (Theorem recognition relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.The`

The theorem recognition relation is written

$$\mathsf{RecognizesTheorem}(R_T, T), \qquad R_T, T \in \mathsf{EventFlow}.$$

It is read as: the generated theorem recognizer flow R_T *recognizes the event flow* T *as a theorem flow.*

Remark 9213.4 (Recognition is not theorem input)**.** *The judgment* $\mathsf{RecognizesTheorem}(R_T, T)$ *does not say that an external theorem object has been supplied. It says that, in the decoded source event layer,* R_T *recognizes* T *as having theorem-flow structure. The direction is*

$$\mathsf{EventFlow} \longrightarrow \mathsf{RecognizedTheoremFlow},$$

not

$$\mathsf{TheoremObject} \longrightarrow \mathsf{EventFlow}.$$

Definition 9213.5 (Recognized theorem flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.Theorem`

Define

$$\mathsf{TheoremFlow}(T) \;:\Longleftrightarrow\; \exists R_T, \mathsf{RecognizesTheorem}(R_T, T) \wedge \mathsf{CompleteTheoremFlowRecognition}(R_T, T).$$

Thus T *is a recognized theorem flow exactly when some generated recognizer recognizes it in that role and recognizes the complete theorem-role structure inside it.*

Principle 9213.6 (No external theorem input)**.** *Lean4* ✓ `BEDC.GroundCompiler.TheoremGenerated.no external theorem input`

There is no formal compiler entry point

$$\mathsf{CompileTheorem} : \mathsf{TheoremObject} \to \mathsf{LegalZStream}.$$

The only formal compiler entry point remains

$$\mathsf{Compile} : \mathsf{EventFlow} \to \mathsf{LegalZStream}.$$

If T *is recognized as a theorem flow, its theorem code is just* $\mathsf{Compile}(T)$.

Proposition 9213.7 (Recognition preserves channel code)**.** *Lean4* ✓ `BEDC.GroundCompiler.TheoremGenerated.the recognition preserves code`

If $\mathsf{RecognizesTheorem}(R_T, T)$*, then the channel encoding of* T *is still*

$$\mathsf{Compile}(T).$$

Theorem recognition does not add, delete, or rewrite channel bits.

Proof. The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ is a source-level recognition relation. It says only that T is recognized by R_T as a theorem flow. The channel encoding is supplied by $\mathsf{Compile}$ and depends only on the event flow T. Since recognition does not alter T, it does not alter $\mathsf{Compile}(T)$. □

Corollary 9213.8 (Recognized theorem code is not a new code)**.** *Lean4* ✓ `BEDC.GroundCompiler.TheoremGenerated code not separate`

If T *is a recognized theorem flow, then*

$$\mathsf{TheoremCode}(T) := \mathsf{Compile}(T)$$

is the channel code of the same event flow T*, not the output of a separate theorem compiler.*

9213.2 Theorem roles

Definition 9213.9 (Recognized theorem subflows). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.The`

If $\mathsf{RecognizesTheorem}(R_T, T)$*, then* R_T *may recognize subflows of* T *as theorem roles:*

$$\begin{array}{ll} \mathsf{ThmStatement}_{R_T}(T, T_0), & \mathsf{ThmDependencies}_{R_T}(T, T_1), \\ \mathsf{ThmProof}_{R_T}(T, T_2), & \mathsf{ThmCertificates}_{R_T}(T, T_3), \\ \mathsf{ThmLedger}_{R_T}(T, T_4), & \mathsf{ThmStatus}_{R_T}(T, T_5), \\ \mathsf{ThmCanonicalSite}_{R_T}(T, T_6), & \mathsf{ThmSeal}_{R_T}(T, \sigma_T). \end{array}$$

These are not external field labels. They are source-level subflow roles recognized inside T *by* R_T*.*

Remark 9213.10 (Theorem field names are explanatory metalanguage). *The words statement, dependency, proof, certificate, ledger, status, canonical site, and seal are explanatory metalanguage. Formally, the roles must be recognized in event flows by relations such as*

$$\mathsf{ThmStatement}_{R_T}, \quad \mathsf{ThmDependencies}_{R_T}, \quad \mathsf{ThmProof}_{R_T}, \quad \mathsf{ThmCertificates}_{R_T},$$
$$\mathsf{ThmLedger}_{R_T}, \quad \mathsf{ThmStatus}_{R_T}, \quad \mathsf{ThmCanonicalSite}_{R_T}, \quad \mathsf{ThmSeal}_{R_T}.$$

This preserves the no-hidden-input rule: theorem fields are not first-layer inputs, and theorem structure is not an external record supplied to the compiler.

Definition 9213.11 (Complete theorem-flow recognition). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenera`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *has* complete theorem-flow recognition *when there exist subflows*

$$T_0, T_1, T_2, T_3, T_4, T_5, T_6, \sigma_T$$

such that

$$\mathsf{ThmStatement}_{R_T}(T, T_0), \quad \mathsf{ThmDependencies}_{R_T}(T, T_1), \quad \mathsf{ThmProof}_{R_T}(T, T_2),$$
$$\mathsf{ThmCertificates}_{R_T}(T, T_3), \quad \mathsf{ThmLedger}_{R_T}(T, T_4), \quad \mathsf{ThmStatus}_{R_T}(T, T_5),$$
$$\mathsf{ThmCanonicalSite}_{R_T}(T, T_6), \quad \mathsf{ThmSeal}_{R_T}(T, \sigma_T).$$

Intuitively, T_0 *is the statement flow,* T_1 *the dependency flow,* T_2 *the proof flow,* T_3 *the certificate flow,* T_4 *the ledger flow,* T_5 *the status flow,* T_6 *the canonical-site flow, and* σ_T *the theorem-seal flow. These roles are recognized by* R_T*, not supplied by an external theorem package.*

Principle 9213.12 (No theorem without complete recognized flow). *Lean4 ✓* `BEDC.GroundCompiler.TheoremGener` `theorem without complete recognition`

A recognized theorem flow must have complete theorem-flow recognition. Thus

$$\mathsf{TheoremFlow}(T) \Longrightarrow \exists R_T, \mathsf{RecognizesTheorem}(R_T, T) \wedge \mathsf{CompleteTheoremFlowRecognition}(R_T, T),$$

with recognized statement, dependency, proof, certificate, ledger, status, canonical-site, and seal subflows.

Theorem 9213.13 (Incomplete theorem flow is not a theorem flow). *Lean4 ✓* `BEDC.GroundCompiler.TheoremGener` `theorem flow not theorem`

If, for every generated theorem recognizer R_T *that recognizes* T*, complete theorem-flow recognition fails, then* T *is not a theorem flow.*

Proof. By Definition 9213.12, a recognized theorem flow must have complete theorem-flow recognition. If any required subflow is absent, complete recognition fails. Therefore $\mathsf{TheoremFlow}(T)$ cannot be used for that incomplete flow. □

Corollary 9213.14 (Statement alone is not a theorem)**.** Lean4 ✓ `BEDC.GroundCompiler.TheoremGenerated.statem` `alone not theorem`

If an event flow carries a recognized statement subflow but no generated theorem recognizer recognizes a proof subflow for it, then the event flow is not a theorem flow.

Corollary 9213.15 (Proof alone is not a theorem)**.** Lean4 ✓ `BEDC.GroundCompiler.TheoremGenerated.proof` `alone not theorem`

A recognized proof flow does not by itself constitute a theorem flow. It must be recognized as proving a statement flow under dependency, certificate, ledger, status, and canonical-site structure.

9213.3 Proof, certificate, ledger, status, and site soundness

Definition 9213.16 (Generated proof checker flow)**.** Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.Ge`

A generated proof checker flow *is an event flow*

$$R_P \in \mathsf{EventFlow}.$$

It is not a host proof checker, Lean kernel, or external verifier. It is itself generated as an event flow.

Definition 9213.17 (Proof checking relation)**.** Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.ChecksP`

The proof checking relation is written

$$\mathsf{ChecksProof}(R_P, T_0, T_1, T_2).$$

It is read as: the generated proof checker flow R_P checks that proof-flow T_2 proves statement-flow T_0 using dependency-flow T_1. This is a source-level recognition or verification judgment, not host-level proof checking supplied as compiler input.

Definition 9213.18 (Proof-sound theorem recognition)**.** Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerate`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *is* proof-sound *when there exist T_0, T_1, T_2 and a generated proof checker flow R_P such that*

$$\mathsf{ThmStatement}_{R_T}(T, T_0), \qquad \mathsf{ThmDependencies}_{R_T}(T, T_1), \qquad \mathsf{ThmProof}_{R_T}(T, T_2),$$
$$\mathsf{ChecksProof}(R_P, T_0, T_1, T_2).$$

Definition 9213.19 (Certificate soundness)**.** Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.Certifica`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *is* certificate-sound *when there is a certificate subflow T_3 recognized inside T by R_T and a generated certificate recognizer R_C such that*

$$\mathsf{ThmCertificates}_{R_T}(T, T_3) \quad \textit{and} \quad \mathsf{Recognizes}(R_C, T_3, T_3)$$

hold. The certificate subflow may itself be recognized as containing required NameCert *flows, derivation-certificate flows, proof-certificate flows, or other certificate flows; the certificate datum is recognized from the theorem event flow, not supplied externally.*

Definition 9213.20 (Ledger-sound theorem recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerat`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *is* ledger-sound *when*

$$\mathsf{ThmLedger}_{R_T}(T, T_4)$$

and T_4 *records the source material hidden or cited by the statement, dependencies, proof, certificates, status, and canonical-site subflows.*

Definition 9213.21 (Status-sound theorem recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerate`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *is* status-sound *when*

$$\mathsf{ThmStatus}_{R_T}(T, T_5)$$

and T_5 *is recognized by a generated status recognizer as the theorem's derivation status, strength, or proof status. The status is not a host enum; it is an event-flow role.*

Definition 9213.22 (Site-sound theorem recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.`

The judgment $\mathsf{RecognizesTheorem}(R_T, T)$ *is* site-sound *when*

$$\mathsf{ThmCanonicalSite}_{R_T}(T, T_6)$$

and T_6 *records the theorem flow's canonical site, namely the recognized place where the theorem package occurs in the theory-dependency flow. This prevents the same statement, wrapper, or proof from being confused with the same raw theorem flow merely because their statement readings coincide.*

Definition 9213.23 (Sound theorem flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.SoundTheor`

A recognized theorem judgment

$$\mathsf{RecognizesTheorem}(R_T, T)$$

is sound *when it has complete theorem-flow recognition and is proof-sound, certificate-sound, ledger-sound, status-sound, and site-sound.*

Theorem 9213.24 (Sound flow establishes theoremhood)**.** *Lean4 ✓* `BEDC.GroundCompiler.TheoremGenerated.soun` `theorem flow establishes theoremhood`

If $\mathsf{RecognizesTheorem}(R_T, T)$ *and this recognition is sound, then*

$$\mathsf{TheoremFlow}(T),$$

and T *is available as a source-level event flow for theorem package code.*

Proof. Complete theorem-flow recognition supplies statement, dependency, proof, certificate, ledger, status, canonical-site, and theorem-seal subflows. Proof-soundness recognizes the proof flow as proving the statement flow using the dependency flow. Certificate-soundness recognizes the required certificate flows and their own soundness. Ledger-soundness records hidden and cited source material. Status-soundness and site-soundness record the theorem's strength and canonical occurrence. Hence T satisfies the flow-level conditions for theoremhood, and its theorem package code is obtained from the ordinary channel compiler. □

9213.4 Theorem code and quotients

Definition 9213.25 (Theorem code)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremGenerated.TheoremCode`

If T is a sound recognized theorem flow, define

$$\mathsf{TheoremCode}(T) := \mathsf{Compile}(T).$$

This code is the channel compilation of T. It is not a proof, theorem object, theorem label, or separate theorem compiler output.

Proposition 9213.26 (Theorem code is injective on raw theorem flows)**.** *Lean4✓* `BEDC.GroundCompiler.TheoremG` `code injective`

If T and U are sound recognized theorem flows and

$$\mathsf{TheoremCode}(T) = \mathsf{TheoremCode}(U),$$

then $T = U$.

Proof. By definition, $\mathsf{TheoremCode}(T) = \mathsf{Compile}(T)$ and $\mathsf{TheoremCode}(U) = \mathsf{Compile}(U)$. Equality of theorem codes gives

$$\mathsf{Compile}(T) = \mathsf{Compile}(U).$$

By the channel bijection of Definition 9208.19, $\mathsf{Compile}$ is injective on event flows. Hence $T = U$. □

Corollary 9213.27 (Raw theorem flows and theorem codes are bijective)**.** *Lean4✓* `BEDC.GroundCompiler.Theorem` `code bijective`

The theorem-flow carrier, considered at the raw event-flow layer, is in bijection with its theorem codes:

$$\mathsf{TheoremCandidateFlow} \cong \mathsf{LegalTheoremCode}_{01}.$$

Sound recognition is an additional source-level predicate on this carrier; it does not change the underlying channel code map.

Remark 9213.28 (Abstract propositions are not theorem flows)**.** *An abstract proposition is not the same thing as a theorem flow. The same proposition can have multiple statement presentations, proofs, dependency unfoldings, certificate strengths, ledgers, canonical sites, bridge statuses, and wrapper theorem flows. Thus*

$$\textit{proposition} \ncong \textit{raw theorem code}$$

in general. The bijection is between canonical raw theorem flows and their theorem codes.

Proposition 9213.29 (The same statement can have multiple theorem flows)**.** *Lean4✓* `BEDC.GroundCompiler.The` `statement multiple flows`

There may be theorem flows T and U whose statement subflows are identified by the same statement classifier, while $T \neq U$.

Proof. Take two theorem flows with classifier-equivalent statement flows but different proof, dependency, ledger, status, or canonical-site subflows. As complete event flows, they are distinct. Their statements may nevertheless be equivalent under the statement classifier. Hence one classified statement can be carried by multiple raw theorem flows. □

Corollary 9213.30 (Code equality refines statement equivalence). *Lean4 ✓ BEDC.GroundCompiler.TheoremGenerat equality refines statement equivalence*

If

$$\mathsf{TheoremCode}(T) = \mathsf{TheoremCode}(U),$$

then $T = U$, so the full statement, proof, dependencies, certificates, ledger, status, and canonical-site structure agree at the raw-flow layer. Statement-flow equivalence alone does not imply theorem-code equality.

Definition 9213.31 (Theorem classifier quotient). *Lean4 (definition) BEDC.GroundCompiler.TheoremGenerated.Theo*

If a theorem classifier $\sim_{\mathrm{thm}}$ identifies theorem flows, the classified theorem object is an equivalence class

$$[T]_{\sim_{\mathrm{thm}}}.$$

The projection

$$T \longmapsto [T]_{\sim_{\mathrm{thm}}}$$

need not be injective.

Proposition 9213.32 (Classified theorem objects need not be code-bijective). *Lean4 ✓ BEDC.GroundCompiler.Theo theorem objects not code bijective*

If there are sound theorem flows $T \neq U$ with

$$T \sim_{\mathrm{thm}} U,$$

then the map from theorem codes to classified theorem objects is not injective.

Proof. By Definition 9213.26, $T \neq U$ implies

$$\mathsf{TheoremCode}(T) \neq \mathsf{TheoremCode}(U).$$

But $T \sim_{\mathrm{thm}} U$ implies

$$[T]_{\sim_{\mathrm{thm}}} = [U]_{\sim_{\mathrm{thm}}}.$$

Thus two distinct theorem codes map to one classified theorem object. □

Corollary 9213.33 (The theorem-flow layer and theorem-object layer differ). *Lean4 ✓ BEDC.GroundCompiler.Theo flow object layer differ*

The raw-flow layer and classified-object layer have different maps:

$$\textit{theorem flow} \leftrightarrow \textit{theorem code}, \qquad \textit{theorem flow} \longrightarrow \textit{classified theorem object}.$$

The first is bijective; the second is generally a quotient.

9213.5 Recognition from code

Definition 9213.34 (Theorem recognition from channel code). *Lean4 (definition) BEDC.GroundCompiler.TheoremGe*

For a legal channel stream $c \in \mathsf{LegalZStream}$, define

$$\mathsf{RecognizesTheoremCode}(R_T, c) \;:\Longleftrightarrow\; \exists T,\ \mathsf{Decode}(c) = T \wedge \mathsf{RecognizesTheorem}(R_T, T).$$

Recognition from channel code therefore proceeds through decoding.

Proposition 9213.35 (No theorem recognition by channel substring). *Lean4✓ BEDC.GroundCompiler.TheoremGen theorem recognition by channel substring*

There is no valid rule that recognizes a theorem by inspecting arbitrary channel substrings without first decoding the channel stream into an event flow. A channel-code recognition judgment factors through a decoded event flow T and a source-level judgment $\mathsf{RecognizesTheorem}(R_T, T)$.

Proof. Definition 9209.27 proves that channel substrings are not source events. Theorem recognition is a source-level relation over event flows. Any theorem recognition rule acting directly on channel substrings would therefore operate on the wrong layer. Valid recognition must first apply Decode. □

Theorem 9213.36 (Compile-decode preserves recognition). *Lean4✓ BEDC.GroundCompiler.TheoremGenerated.co decode preserves recognition*

If $\mathsf{RecognizesTheorem}(R_T, T)$, *then*

$$\mathsf{RecognizesTheoremCode}(R_T, \mathsf{Compile}(T)).$$

Proof. By the flow-level round trip Definition 9208.17,

$$\mathsf{Decode}(\mathsf{Compile}(T)) = T.$$

Together with $\mathsf{RecognizesTheorem}(R_T, T)$, this supplies the decoded witness required by $\mathsf{RecognizesTheoremCode}(R$ □

Corollary 9213.37 (Theorem recognition is invariant under compile-decode). *Lean4✓ BEDC.GroundCompiler.Theo invariant under compile decode*

For every recognized theorem flow, its theorem code is recognized after the compile-decode round trip.

9213.6 Acceptance and seals

Definition 9213.38 (Accepted theorem flow). *Lean4 (definition) BEDC.GroundCompiler.TheoremGenerated.Accepte*

A theorem flow T is an accepted theorem flow *when:*

1. *T is a sound recognized theorem flow;*
2. *the required* NameCert, *derivation-certificate, or proof-certificate flows in its certificate subflow are soundly recognized;*
3. *whenever the theorem claims mature-object export, the corresponding acceptance-gate flow of Definition 9212.23 is recognized.*

This is written $\mathsf{AcceptedTheoremFlow}(T)$.

Proposition 9213.39 (Theorem code existence does not imply accepted theorem). *Lean4✓ BEDC.GroundCompiler code existence not accepted*

There exists a legal theorem code $c \in \mathsf{LegalTheoremCode}_{01}$ whose decoded event flow is not an accepted theorem flow. Thus legal channel-code existence alone does not imply

$$\mathsf{AcceptedTheoremFlow}(T)$$

for the decoded theorem-candidate flow T.

Proof. Legal channel membership says only that c decodes to an event flow. Accepted theoremhood requires theorem recognition, proof-soundness, certificate-soundness, ledger-soundness, status-soundness, and, where export is claimed, an acceptance-gate flow. Channel legality supplies none of those recognition judgments. □

Corollary 9213.40 (Code is not proof). Lean4✓ `BEDC.GroundCompiler.TheoremGenerated.theorem code is not proof`

Theorem code records a theorem flow. It does not replace the proof-flow, generated proof checker, or proof-soundness judgment.

Definition 9213.41 (Theorem seal subflow). Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremSe`

If $\mathsf{RecognizesTheorem}(R_T, T)$ *and* R_T *recognizes a terminal subflow* σ_T *of* T *as expressing theorem-package completion, then* σ_T *is a* theorem-seal subflow*, written*

$$\mathsf{ThmSeal}_{R_T}(T, \sigma_T).$$

The theorem seal is a source-level subflow, not the channel-level terminator 11*.*

Proposition 9213.42 (Theorem seal is not the channel terminator). Lean4✓ `BEDC.GroundCompiler.TheoremGener seal is source subflow`

If $\mathsf{ThmSeal}_{R_T}(T, \sigma_T)$*, then* σ_T *is a source-level subflow of* T*, not a channel-level terminator datum.*

Proof. The channel-level 11 is the event terminator of Definition 9208.2; it belongs to the LegalZStream encoding mechanism. A theorem seal is recognized inside a decoded event flow. By the layer-separation theorem Definition 9209.27, a channel substring cannot be used directly as a source subflow. Hence the theorem seal is not the channel terminator. □

9213.7 Conservativity and examples

Theorem 9213.43 (Theorem-flow recognition is conservative over the finite kernel). Lean4✓ `BEDC.GroundCompiler.TheoremGenerated.theorem flow recognition conservativity`

Adding RecognizesTheorem*,* ChecksProof*,* AcceptedTheoremFlow*, and the associated role-recognition relations does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.*

Proof. These relations assert theorem recognition, proof checking, certificate recognition, ledger discipline, status recognition, site recognition, and acceptance bookkeeping over event flows. They do not generate new b0 or b1 marks, add msame, hsame, or psame rules, change Ext or Cont, or turn theorem objects into kernel primitives. Theorem acceptance still depends on recognized proof, certificate, ledger, status, and acceptance flows. Hence the recognition layer is conservative over the finite kernel. □

Example 9213.44 (Channel round-trip theorem as theorem flow). *The channel round-trip theorem can be represented by a theorem flow whose statement subflow reads* $\mathsf{Decode}(\mathsf{Compile}(S)) = S$*, whose dependencies include event encoding, decoding, the no-internal-*11 *lemma, and event-level round trip, and whose proof flow is an induction over the event flow. Its certificate subflow records the compiler definitions and decoder-correctness evidence; its ledger records both* S *and* $\mathsf{Compile}(S)$*; its status subflow records the recognized strength; and its canonical site points to the channel-encoding theorem site. The theorem code is* Compile *of that whole theorem flow, not the channel code of the displayed statement alone.*

Example 9213.45 (Zeckendorf carry theorem as theorem flow)**.** *A source-level carry theorem may have a statement subflow saying that* 011 *and* 100 *are related by a Zeckendorf carry relation. Its dependencies include source-event definitions, a normal-form recognizer, the no-adjacent-boundary pattern, and the carry classifier. Its proof flow supplies the classifier witness, and its ledger records both the pre-normal event* 011 *and the normal event* 100*. This theorem flow does not assert* $011 = 100$ *or* $\mathsf{hsame}(011, 100)$*; it asserts a classifier-level carry relation with ledger support.*

Example 9213.46 ($\mathsf{FullAxis}^{\uparrow}$ theorem as theorem flow)**.** *A* $\mathsf{FullAxis}^{\uparrow}$ *theorem flow can state that* $\mathsf{FullAxis}^{\uparrow}$ *is licensed by finite-prefix source, compatible-thread pattern, prefix-compatible classifier, seal stability, and tail ledger. Its dependencies include* $\mathsf{AxisNat}^{\uparrow}$ *flow, finite-prefix flow, thread flow, seal-certificate flow, and* $\mathsf{NameCert}$ *flow. Its proof flow verifies the five-field certificate structure, while its ledger retains* 0, 00, 000*, the repeat-boundary witness, the seal witness, and all finite prefixes. Such a theorem flow is not the bare event* 011*; that event can only be a local witness inside the flow.*

Remark 9213.47 (Outlook)**.** *This chapter establishes that a theorem is not an external label or proof object, but a recognized theorem flow. The core judgments are* $\mathsf{RecognizesTheorem}(R_T, T)$ *and* $\mathsf{ChecksProof}(R_P, T_0, T_1, T_2)$*. Theorem recognition preserves channel code, theorem code is just* $\mathsf{Compile}(T)$*, complete theorem flow requires statement, dependencies, proof, certificates, ledger, status, canonical site, and seal, statement alone and proof alone are insufficient, sound theorem flow establishes theoremhood at the flow level, theorem code is injective only on raw theorem flows, classified theorem objects are quotients, recognition from code must pass through* Decode*, code existence does not imply accepted theoremhood, code is not proof, theorem seals are not channel terminators, and theorem-flow recognition is conservative over the finite kernel. The next ground-compiler chapter applies the same rule to chapter flow recognition.*

Chapter 9214

Chapter Flow Recognition

chapter 9213 records the no-hidden-input reading of theoremhood: a theorem is not an external theorem label or proof object, but a recognized statement, dependency, proof, certificate, ledger, status, canonical-site, and seal structure inside an event flow. This chapter lifts the same rule to chapter structure.

The governing rule is

a chapter is recognized event-flow structure, not a manifest, outline, or external package.

The formal compiler therefore does not accept a chapter package, manifest, YAML file, Markdown heading tree, LaTeX section tree, external chapter identifier, chapter title string, definition list, theorem list, proof-tree list, certificate table, cannot-claim table, or status table. A chapter enters this layer only as an event flow recognized by a generated chapter recognizer.

9214.1 Chapter recognition

Definition 9214.1 (Chapter candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapterCandid`

A chapter candidate flow *is an event flow*

$$C \in \mathsf{EventFlow}.$$

It has not yet been recognized as a chapter. Thus $C \in \mathsf{EventFlow}$ *does not imply that* C *is a chapter, contains definitions, lemmas, theorems, proofs, certificates, ledgers, status data, or carries a chapter seal.*

Definition 9214.2 (Generated chapter recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Generate`

A generated chapter recognizer *is an event flow*

$$R_C \in \mathsf{EventFlow}.$$

It is not a YAML parser, Markdown parser, LaTeX section parser, host-level AST constructor, or external chapter manifest. Its public formal carrier is itself generated from the two raw marks and acts on decoded event flows.

Definition 9214.3 (Chapter recognition relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Recognize`

The chapter recognition relation is written

$$\mathsf{RecognizesChapter}(R_C, C), \qquad R_C, C \in \mathsf{EventFlow}.$$

It is read as: the generated chapter recognizer flow R_C *recognizes the event flow* C *as a chapter flow.*

Remark 9214.4 (Recognition is not chapter input)**.** *The judgment* $\mathsf{RecognizesChapter}(R_C, C)$ *does not say that an external chapter object has been supplied. It says that, in the decoded source event layer,* R_C *recognizes* C *as having chapter-flow structure. The direction is*

$$\mathsf{EventFlow} \longrightarrow \mathsf{RecognizedChapterFlow},$$

not

$$\mathsf{ChapterPkg} \longrightarrow \mathsf{EventFlow}.$$

Definition 9214.5 (Recognized chapter flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapterFlow`

Define

$$\mathsf{ChapterFlow}(C) \;:\Longleftrightarrow\; \exists R_C,\ \mathsf{CompleteChapterRecognition}(R_C, C).$$

Thus C *is a recognized chapter flow exactly when some generated chapter recognizer recognizes it with the complete chapter-role structure recorded below.*

Principle 9214.6 (No external chapter input)**.** *Lean4 ✓* `BEDC.GroundCompiler.ChapterFlow.no external chapter input`

There is no formal compiler entry point

$$\mathsf{CompileChapter} : \mathsf{ChapterPkg} \to \mathsf{LegalZStream}.$$

The only formal compiler entry point remains

$$\mathsf{Compile} : \mathsf{EventFlow} \to \mathsf{LegalZStream}.$$

If C *is recognized as a chapter flow, its chapter code is just* $\mathsf{Compile}(C)$*.*

Proposition 9214.7 (Recognition preserves channel code)**.** *Lean4 ✓* `BEDC.GroundCompiler.ChapterFlow.chapter recognition preserves code`

If $\mathsf{RecognizesChapter}(R_C, C)$*, then the channel encoding of* C *is still*

$$\mathsf{Compile}(C).$$

Chapter recognition does not add, delete, or rewrite channel bits.

Proof. The judgment $\mathsf{RecognizesChapter}(R_C, C)$ is a source-level recognition relation. It says only that C is recognized by R_C as a chapter flow. The channel encoding is supplied by $\mathsf{Compile}$ and depends only on the event flow C. Since recognition does not alter C, it does not alter $\mathsf{Compile}(C)$. □

Corollary 9214.8 (Recognized chapter code is not a new code)**.** *Lean4 ✓* `BEDC.GroundCompiler.ChapterFlow.chap code not new code`

If C *is a recognized chapter flow, then*

$$\mathsf{ChapterCode}(C) := \mathsf{Compile}(C)$$

is the channel code of the same event flow C*, not the output of a separate chapter compiler.*

9214.2 Chapter roles

Definition 9214.9 (Recognized chapter subflows). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapterSt`

If $\mathsf{RecognizesChapter}(R_C, C)$*, then* R_C *may recognize subflows of* C *as chapter roles:*

$$\begin{array}{ll} \mathsf{ChapStart}_{R_C}(C, C_0), & \mathsf{ChapDependencies}_{R_C}(C, C_1), \\ \mathsf{ChapDefinitions}_{R_C}(C, C_2), & \mathsf{ChapLemmas}_{R_C}(C, C_3), \\ \mathsf{ChapTheorems}_{R_C}(C, C_4), & \mathsf{ChapProofs}_{R_C}(C, C_5), \\ \mathsf{ChapCertificates}_{R_C}(C, C_6), & \mathsf{ChapLedger}_{R_C}(C, C_7), \\ \mathsf{ChapCannotClaims}_{R_C}(C, C_8), & \mathsf{ChapStatus}_{R_C}(C, C_9), \\ \mathsf{ChapSeal}_{R_C}(C, \sigma_C). & \end{array}$$

These are not external field labels. They are source-level subflow roles recognized inside C *by* R_C*.*

Remark 9214.10 (Chapter field names are explanatory metalanguage). *The words chapter-start, dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and chapter-seal are explanatory metalanguage. Formally, the roles must be recognized in event flows by the corresponding* $\mathsf{Chap}\cdots_{R_C}$ *relations. This preserves the no-hidden-input rule: chapter fields are not first-layer inputs, and chapter structure is not an external record supplied to the compiler.*

Definition 9214.11 (Complete chapter-flow recognition). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Co`

The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *has* complete chapter-flow recognition *when there exist subflows*

$$C_0, C_1, \ldots, C_9, \sigma_C$$

such that

$$\begin{array}{ccc} \mathsf{ChapStart}_{R_C}(C, C_0), & \mathsf{ChapDependencies}_{R_C}(C, C_1), & \mathsf{ChapDefinitions}_{R_C}(C, C_2), \\ \mathsf{ChapLemmas}_{R_C}(C, C_3), & \mathsf{ChapTheorems}_{R_C}(C, C_4), & \mathsf{ChapProofs}_{R_C}(C, C_5), \\ \mathsf{ChapCertificates}_{R_C}(C, C_6), & \mathsf{ChapLedger}_{R_C}(C, C_7), & \mathsf{ChapCannotClaims}_{R_C}(C, C_8), \\ \multicolumn{3}{c}{\mathsf{ChapStatus}_{R_C}(C, C_9), \qquad \mathsf{ChapSeal}_{R_C}(C, \sigma_C).} \end{array}$$

Intuitively, C_0 *is the chapter-start flow,* C_1 *the dependency flow,* C_2 *the definition flow,* C_3 *the lemma flow,* C_4 *the theorem flow,* C_5 *the proof flow,* C_6 *the certificate flow,* C_7 *the ledger flow,* C_8 *the cannot-claim flow,* C_9 *the status flow, and* σ_C *the chapter-seal flow. These roles are recognized by* R_C*, not supplied by an external chapter package.*

Principle 9214.12 (No chapter without complete recognized flow). *Lean4 ✓* `BEDC.GroundCompiler.ChapterFlow.n` `chapter without complete recognition`

A recognized chapter flow must have complete chapter-flow recognition. Thus

$$\mathsf{ChapterFlow}(C) \Longrightarrow \exists R_C,\ \mathsf{CompleteChapterRecognition}(R_C, C),$$

with recognized chapter-start, dependency, definition, lemma, theorem, proof, certificate, ledger, cannot-claim, status, and seal subflows.

Theorem 9214.13 (Incomplete chapter flow is not a chapter flow). *Lean4 ✓* `BEDC.GroundCompiler.ChapterFlow.in` `chapter flow not chapter`

If, for every generated chapter recognizer R_C*, the complete recognition judgment*

$$\mathsf{CompleteChapterRecognition}(R_C, C)$$

fails, then C cannot be used as a recognized chapter flow. In particular, an event flow that lacks any one of the recognized chapter-start, dependency, definition, lemma, theorem, proof, certificate, ledger, cannot-claim, status, or chapter-seal subflows for every candidate recognizer is not a chapter flow.

Proof. By definition, $\mathsf{ChapterFlow}(C)$ means that some generated recognizer R_C supplies CompleteChapterRecognitio If every such complete recognition judgment fails, the existential witness required for $\mathsf{ChapterFlow}(C)$ cannot exist. □

Corollary 9214.14 (Title alone is not a chapter)**.** Lean4 ✓ BEDC.GroundCompiler.ChapterFlow.title alone not chapter

A chapter-start flow or title-like event does not by itself constitute a chapter flow. If no generated chapter recognizer supplies complete chapter-flow recognition for the same candidate flow, then the candidate flow is not a chapter flow. It still needs dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and chapter seal.

Corollary 9214.15 (Theorem list alone is not a chapter)**.** Lean4 ✓ BEDC.GroundCompiler.ChapterFlow.theorem list alone not chapter

Even if an event flow contains recognized theorem flows, that alone does not constitute a chapter flow. If no generated chapter recognizer supplies complete chapter-flow recognition for the same candidate flow, then the candidate flow is not a chapter flow. The chapter still needs dependency, definition, proof, certificate, ledger, cannot-claim, status, and chapter-seal structure.

9214.3 Soundness of chapter structure

Definition 9214.16 (Dependency soundness)**.** Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.DependencySo

The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* dependency-sound *when*

$$\mathsf{ChapDependencies}_{R_C}(C, C_1)$$

for some recognized dependency subflow C_1. The ordering, deferred-claim marks, cannot-claim marks, and open-obligation marks are carried by this event subflow, not by an external dependency manifest.

Definition 9214.17 (Definition soundness)**.** Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.DefinitionSoun

The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* definition-sound *when*

$$\mathsf{ChapDefinitions}_{R_C}(C, C_2)$$

and each definition flow inside C_2 is generated from prior dependencies, raw events, recognized package flows, recognized NameCert *flows, or earlier definitions. If a definition introduces a derived name, the corresponding* NameCert *flow must be recognized, unless the definition is explicitly marked as candidate-only.*

Definition 9214.18 (Theorem soundness)**.** Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.TheoremSound

The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* theorem-sound *when*

$$\mathsf{ChapTheorems}_{R_C}(C, C_4)$$

and each theorem subflow inside C_4 is recognized by a generated theorem recognizer as a sound theorem flow in the sense of Definition 9213.23.

Definition 9214.19 (Proof soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ProofSound`
The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* proof-sound *when*

$$\mathsf{ChapProofs}_{R_C}(C, C_5)$$

and each proof subflow inside C_5 *is checked by a generated proof checker flow, or else is explicitly marked as a proof obligation, paper-level proof, or unchecked proof status.*

Definition 9214.20 (Certificate soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.CertificateSou`
The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* certificate-sound *when*

$$\mathsf{ChapCertificates}_{R_C}(C, C_6)$$

and each certificate subflow inside C_6 *is recognized by its appropriate recognizer, for example as a* NameCert *flow, derivation-certificate flow, acceptance-gate flow, seal-certificate flow, reuse-certificate flow, or bridge-certificate flow, or else is explicitly marked as an open certificate obligation.*

Definition 9214.21 (Ledger soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.LedgerSound`
The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* ledger-sound *when*

$$\mathsf{ChapLedger}_{R_C}(C, C_7)$$

and C_7 *records the source material hidden or cited by the chapter's definitions, theorems, proofs, certificates, cannot-claims, status flow, and dependency flow. If the chapter uses nontrivial packaging, compression, classifier quotients, normalization, completion, or bridge structure, the ledger flow records the corresponding provenance.*

Definition 9214.22 (Cannot-claim soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.CannotClaim`
The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* cannot-claim-sound *when*

$$\mathsf{ChapCannotClaims}_{R_C}(C, C_8)$$

and C_8 *records the claims the chapter is not allowed to make. Typical examples are bare code equals theorem, bare event equals mature object, source-level carry equals* hsame*, channel-level* 11 *participates in source-level carry, derived interface without certificate, completion without ledger, and export without acceptance gate. These cannot-claim items must be consistent with the chapter's definitions, theorems, certificates, and status flow.*

Definition 9214.23 (Status soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.StatusSound`
The judgment $\mathsf{RecognizesChapter}(R_C, C)$ *is* status-sound *when*

$$\mathsf{ChapStatus}_{R_C}(C, C_9)$$

and C_9 *records the status of each main definition, theorem, certificate, bridge, and implementation target in the chapter. Explanatory status readings include* seed, paperCert, checkedCert, bridgeCert*, open obligation, and cannot-claim. These status readings are not automatically upgraded by channel code; they must be supplied by recognized status flows.*

Definition 9214.24 (Sound recognized chapter flow). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.SoundR`
A recognized chapter judgment

$$\mathsf{RecognizesChapter}(R_C, C)$$

is sound *when it has complete chapter-flow recognition and is dependency-sound, definition-sound, theorem-sound, proof-sound, certificate-sound, ledger-sound, cannot-claim-sound, and status-sound.*

Theorem 9214.25 (Sound flow establishes chapterhood). *Lean4✓ BEDC.GroundCompiler.ChapterFlow.sound establishes chapterhood*

If $\mathsf{RecognizesChapter}(R_C, C)$ *and this recognition is sound, then*

$$\mathsf{ChapterFlow}(C),$$

and C *is available as a source-level event flow for chapter code.*

Proof. Complete chapter-flow recognition supplies chapter-start, dependency, definition, lemma, theorem, proof, certificate, ledger, cannot-claim, status, and chapter-seal subflows. Dependency-soundness records the well-founded dependency order. Definition-soundness prevents definitions from adding unlicensed external primitives. Theorem-soundness and proof-soundness recognize theorem and proof subflows in the required sense. Certificate-soundness recognizes the supporting certificate structure. Ledger-soundness tracks hidden and cited material. Cannot-claim-soundness records the claims not licensed by the chapter. Status-soundness records the chapter's strength and implementation status. Hence C satisfies the flow-level conditions for chapterhood, and its chapter code is obtained from the ordinary channel compiler. □

9214.4 Chapter code and quotients

Definition 9214.26 (Chapter code). *Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.ChapterCode*

If C *is a sound recognized chapter flow, define*

$$\mathsf{ChapterCode}(C) := \mathsf{Compile}(C).$$

This code is the channel compilation of C*. It is not a chapter object, YAML file, manifest, or human-readable chapter tree.*

Proposition 9214.27 (Chapter code is injective on raw chapter flows). *Lean4✓ BEDC.GroundCompiler.ChapterFlo code injective*

If C *and* D *are sound recognized chapter flows and*

$$\mathsf{ChapterCode}(C) = \mathsf{ChapterCode}(D),$$

then $C = D$*.*

Proof. By definition, $\mathsf{ChapterCode}(C) = \mathsf{Compile}(C)$ and $\mathsf{ChapterCode}(D) = \mathsf{Compile}(D)$. Equality of chapter codes gives

$$\mathsf{Compile}(C) = \mathsf{Compile}(D).$$

By the channel bijection of Definition 9208.19, $\mathsf{Compile}$ is injective on event flows. Hence $C = D$. □

Corollary 9214.28 (Raw chapter flows and chapter codes are bijective). *Lean4✓ BEDC.GroundCompiler.ChapterFl code bijective*

Sound recognized chapter flows, considered as raw event flows, are in bijection with their chapter codes:

$$\mathsf{SoundChapterFlow} \cong \mathsf{LegalChapterCode}_{01}.$$

Remark 9214.29 (Chapter topic is not chapter flow)**.** *An abstract chapter topic is not the same thing as a chapter flow. A chapter about* NameCert *may have multiple proof orders, lemma organizations, cannot-claim presentations, ledger unfoldings, status reports, and bridge readings. Thus*

$$\textit{chapter topic} \ncong \textit{raw chapter code.}$$

The bijection is between canonical raw chapter flows and their chapter codes.

Definition 9214.30 (Chapter classifier quotient)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapterCl`

A chapter classifier relation $\sim_{\text{chap}}$ is a source-level relation on chapter candidate flows. The classified chapter object relation is represented by

$$\mathsf{ChapterClassifierQuotient}_{\sim_{\text{chap}}}(C, D) \ :\Longleftrightarrow\ C \sim_{\text{chap}} D.$$

The classified projection

$$C \longmapsto [C]_{\sim_{\text{chap}}}$$

need not be injective.

Proposition 9214.31 (Classified chapter objects need not be code-bijective)**.** *Lean4✓* `BEDC.GroundCompiler.Chap` `chapter objects not code bijective`

If there are sound chapter flows $C \neq D$ with

$$C \sim_{\text{chap}} D,$$

then the map from chapter codes to classified chapter objects is not injective.

Proof. By Definition 9214.27, $C \neq D$ implies

$$\mathsf{ChapterCode}(C) \neq \mathsf{ChapterCode}(D).$$

But $C \sim_{\text{chap}} D$ implies

$$[C]_{\sim_{\text{chap}}} = [D]_{\sim_{\text{chap}}}.$$

Thus two distinct chapter codes map to one classified chapter object. □

Corollary 9214.32 (The chapter-flow layer and chapter-object layer differ)**.** *Lean4✓* `BEDC.GroundCompiler.Chapt` `flow object layer differ`

The raw-flow layer and classified-object layer have different maps:

$$\textit{chapter flow} \leftrightarrow \textit{chapter code}, \qquad \textit{chapter flow} \longrightarrow \textit{classified chapter object.}$$

The first is bijective; the second is generally a quotient.

9214.5 Recognition from code and readable views

Definition 9214.33 (Chapter recognition from channel code)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlo`

For a legal channel stream $c \in \mathsf{LegalZStream}$, define

$$\mathsf{RecognizesChapterCode}(R_C, c) \ :\Longleftrightarrow\ \exists C,\ \mathsf{Decode}(c) = C \ \wedge\ \mathsf{RecognizesChapter}(R_C, C).$$

Recognition from channel code therefore proceeds through decoding.

Proposition 9214.34 (No chapter recognition by channel substring). Lean4✓ BEDC.GroundCompiler.ChapterFlow chapter recognition by channel substring

There is no valid rule that recognizes a chapter by inspecting arbitrary channel substrings without first decoding the channel stream into an event flow.

Proof. Definition 9209.27 proves that channel substrings are not source events. Chapter recognition is a source-level relation over event flows. Any chapter recognition rule acting directly on channel substrings would therefore operate on the wrong layer. Valid recognition must first apply Decode. □

Theorem 9214.35 (Compile-decode preserves chapter recognition). Lean4✓ BEDC.GroundCompiler.ChapterFlow. decode preserves chapter recognition

If $\mathsf{RecognizesChapter}(R_C, C)$, *then*

$$\mathsf{RecognizesChapterCode}(R_C, \mathsf{ChapterCode}(C)).$$

Proof. By the flow-level round trip Definition 9208.17,

$$\mathsf{Decode}(\mathsf{ChapterCode}(C)) = C.$$

Together with $\mathsf{RecognizesChapter}(R_C, C)$, this supplies the decoded witness required by RecognizesChapterCode. □

Corollary 9214.36 (Chapter recognition is invariant under compile-decode). Lean4✓ BEDC.GroundCompiler.Chap recognition invariant

For every recognized chapter flow, compiling to a channel stream and decoding back to an event flow preserves its chapter-recognition status.

Definition 9214.37 (Human-readable chapter view). Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.HumanR

A human-readable chapter view *is a display generated from a decoded and recognized chapter flow, such as Markdown, LaTeX, YAML, JSON, an outline, a theorem-status table, or a dependency graph. These views are output presentations, not input structure.*

Principle 9214.38 (YAML is output, not input). Lean4✓ BEDC.GroundCompiler.ChapterFlow.chapter yaml output not input

YAML, manifests, and outlines may be generated from a decoded and recognized chapter flow, but they cannot serve as formal compiler input.

Proposition 9214.39 (YAML input would be hidden structure). Lean4✓ BEDC.GroundCompiler.ChapterFlow.cha yaml input hidden structure

If YAML were accepted as formal input, then chapter boundaries, field names, theorem order, certificate slots, ledger slots, and status labels would be supplied externally. Such input is structural hidden input.

Proof. YAML explicitly supplies chapter structure: field names, nesting, lists, order, labels, and status readings. These are not recognized from a generated event flow over b0 and b1; they are passed in as external structure. This violates the no-hidden-input rule. □

Corollary 9214.40 (YAML may be a report). Lean4✓ BEDC.GroundCompiler.ChapterFlow.yaml may report

If a decoded and recognized chapter flow C *is already available, YAML is a human-readable chapter view generated as a report*

$$C \longrightarrow \mathrm{YAMLView}(C).$$

The reverse direction is not a formal compiler input.

9214.6 Dependencies and seals

Definition 9214.41 (Chapter-level dependency order). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Chap`

The dependency order of a chapter flow is the recognized dependency subflow

$$\mathsf{ChapDependencies}_{R_C}(C, C_1).$$

Each referenced flow is carried with one of the admissible dependency markers: earlier generated, earlier recognized, deferred obligation, cannot-claim item, open obligation, or future bridge. The unmarked-forward case is not a valid dependency-order entry.

Theorem 9214.42 (No forward dependency without marker). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.no forward dependency without marker`

If a chapter flow C refers to a flow X through the dependency subflow C_1, and the reference is assigned the unmarked-forward marker, then it is not a valid chapter-level dependency-order entry.

Proof. By Definition 9214.41, a valid dependency-order entry requires a marker different from the unmarked-forward marker. The displayed reference has exactly that excluded marker, so it cannot satisfy the dependency-order predicate. □

Corollary 9214.43 (Chapter flow must record open obligations). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.ope obligations recorded`

If a chapter flow has complete chapter-flow recognition, then its recognized structure includes a cannot-claim flow, a status flow, and a ledger flow. Any unfinished certificate, proof, bridge, or recognizer that the chapter depends on must be recorded through this recognized obligation structure.

Definition 9214.44 (Chapter seal subflow). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapSeal`

If $\mathsf{RecognizesChapter}(R_C, C)$ *and* R_C *recognizes an event flow* σ_C *as the chapter-completion role for* C*, then* σ_C *is a* chapter-seal subflow*, written*

$$\mathsf{ChapSeal}_{R_C}(C, \sigma_C).$$

The chapter seal is a source-level event flow carrying a recognized role, not the channel-level terminator 11*.*

Proposition 9214.45 (Chapter seal is not the channel terminator). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.c seal is event flow`

If $\mathsf{ChapSeal}_{R_C}(C, \sigma_C)$*, then* σ_C *is a source-level event flow recognized as nonempty formal event-flow input. The seal judgment is therefore not a claim about the channel-level terminator* 11*.*

Proof. The channel-level 11 is the event terminator of Definition 9208.2; it belongs to the LegalZStream encoding mechanism. A chapter seal is recognized inside a decoded event flow. By the layer-separation theorem Definition 9209.27, a channel substring cannot be used directly as a source subflow. Hence the chapter seal is not the channel terminator. □

9214.7 Conservativity and examples

Theorem 9214.46 (Chapter-flow recognition is conservative over the finite kernel). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.chapter flow conservativity`

Adding RecognizesChapter*,* ChapterFlow*,* SoundChapterFlow*, and the associated role-recognition relations does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.*

Proof. These relations assert chapter recognition, theorem inclusion, certificate recognition, ledger tracking, cannot-claim tracking, and status bookkeeping over event flows. They do not generate new b0 or b1 marks, add msame, hsame, or psame rules, change Ext or Cont, or turn chapter objects into kernel primitives. Hence the recognition layer is conservative over the finite kernel. □

Example 9214.47 (NameCert chapter as chapter flow)**.** *A* NameCert *chapter flow may contain a chapter-start flow for the chapter opening, dependency flows for raw events, package and gap flows, classifier flows, and ledger flows, definition flows for candidate certificates and five-field recognition, lemma flows saying that incomplete certificates do not license names and that ledgers are required, theorem flows saying that sound* NameCert *flows license candidate names and that recognition is conservative, certificate flows for supporting recognizers, ledger flows for field provenance and hidden source material, cannot-claim flows excluding code-alone licensing and channel-substring recognition, status flows for theorem targets, and a chapter-seal flow. Its channel code is* $\mathsf{Compile}(C_{\mathrm{NameCertChapter}})$*, not a YAML outline or chapter manifest.*

Example 9214.48 (Channel compiler chapter as chapter flow)**.** *A channel compiler chapter flow may contain a chapter-start flow, dependency flows for event-flow definitions, definition flows for* ζ*, event encoding,* Compile*,* LegalZStream*, and* Decode*, lemma flows for no internal* 11*, first-*11 *termination, and illegality of* 0111*, theorem flows for event-level and flow-level round trips and the bijection* $\mathsf{EventFlow} \cong \mathsf{LegalZStream}$*, proof flows by induction on raw events and event flows, certificate flows for conservativity and source-channel separation, ledgers retaining both raw event and compiled stream, cannot-claim flows excluding code-as-theorem and channel-substring-as-source, status flows for proof and Lean target status, and a chapter seal.*

Remark 9214.49 (Outlook)**.** *This chapter establishes that a chapter is not YAML, a manifest, or a chapter package, but a recognized chapter flow. The core judgment is* $\mathsf{RecognizesChapter}(R_C, C)$*. Chapter recognition preserves channel code, chapter code is just* $\mathsf{Compile}(C)$*, complete chapter flow requires chapter-start, dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and seal, title alone and theorem list alone are insufficient, sound chapter flow establishes chapterhood at the flow level, chapter code is injective only on raw chapter flows, chapter topics and classified chapter objects are quotients, recognition from code must pass through* Decode*, YAML is an output view rather than formal input, forward dependencies require explicit markers, chapter seals are not channel terminators, and chapter-flow recognition is conservative over the finite kernel. The next ground-compiler chapter applies the same rule to recognizer flows themselves.*

Chapter 9215

Recognizer Flows

The preceding chapters established that packages, naming certificates, derivation certificates, acceptance gates, theorems, and chapters are not compiler inputs. They enter the system only as structures recognized inside event flows. This creates the next no-hidden-input question: are the recognizers themselves hidden inputs?

If package, certificate, theorem, or chapter recognition were supplied by a host parser, host function, AST constructor, YAML schema, or Lean meta-level object, the hidden structure would merely move from the object being recognized to the recognizer. The governing rule of this chapter is therefore

$$\boxed{\text{a recognizer is not an external parser; a recognizer is itself an event flow.}}$$

Package recognizers, NameCert recognizers, derivation-certificate recognizers, theorem recognizers, chapter recognizers, proof checkers, status recognizers, ledger recognizers, and compiler recognizers must be generated recognizer flows.

9215.1 Recognizer candidates and roles

Definition 9215.1 (Recognizer candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RecognizerCan`

A recognizer candidate flow *is an event flow*

$$R \in \mathsf{EventFlow}.$$

It has not yet been accepted as a recognizer. Thus $R \in \mathsf{EventFlow}$ *does not imply that* R *recognizes packages, recognizes theorems, checks proofs, or compiles itself. Such capabilities must be supported by recognizer-certificate flows.*

Definition 9215.2 (Recognizer role)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.RecognitionRole`

A recognizer role *is a source-level description of a recognition task. Explanatory examples include package recognition,* NameCert *recognition, derivation-certificate recognition, theorem recognition, chapter recognition, compiler recognition, proof checking, status recognition, and ledger recognition.*

These role names are metalanguage. Formally, a role is not a host enum input; it must be represented and recognized inside the relevant recognizer-certificate flow.

Definition 9215.3 (Recognition relation schema)**.** *Lean4 (definition)* `BEDC.GroundCompiler.EventFlow.Recognizes`

Given a recognizer candidate flow R, a target event flow S, and a recognition role ρ represented by event-flow structure, the generic recognition judgment is written

$$\mathsf{Recognizes}(R, \rho, S).$$

It is read as: the generated flow R recognizes the event flow S under recognition role ρ.

The concrete judgments used earlier are instances or refinements of this schema:

$$\mathsf{RecognizesPkg}(R, S), \qquad \mathsf{RecognizesNameCert}(R, S, N), \qquad \mathsf{RecognizesDerivCert}(R, D, N, s),$$
$$\mathsf{RecognizesTheorem}(R, T), \qquad \mathsf{RecognizesChapter}(R, C), \qquad \mathsf{RecognizesCompiler}(R).$$

These are source-level event-flow judgments, not external parser calls.

Remark 9215.4 (Recognition is not host parsing)**.** *The judgment* $\mathsf{Recognizes}(R, \rho, S)$ *does not mean that a host parser has parsed S, nor that a Lean meta function has accepted it. It means that the event flow R acts in the BEDC source layer as a recognizer witness for the target event flow S under role ρ. Thus the formal system does not allow* RecognizesPkg *to abbreviate an external parser result, or* RecognizesTheorem *to abbreviate host proof-checker output.*

9215.2 Recognizer certificates

Definition 9215.5 (Recognizer certificate candidate)**.** *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Re`

A recognizer certificate candidate *is an event flow*

$$C_R \in \mathsf{EventFlow}.$$

It claims to supply recognition-capability evidence for a recognizer candidate R, but has not yet been recognized as a sound recognizer certificate.

Definition 9215.6 (Recognizer certificate recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.`

The recognizer-certificate recognition relation is written

$$\mathsf{RecognizesRecognizerCert}(Q, C_R, R, \rho), \qquad Q, C_R, R, \rho \in \mathsf{EventFlow}.$$

It is read as: the generated meta-recognizer flow Q recognizes C_R as a recognizer certificate for recognizer flow R under recognition role ρ. The meta-recognizer Q is also an event flow.

Remark 9215.7 (Layered bootstrapping, not external regress)**.** *Definition 9215.6 raises an apparent regress: who recognizes the recognizer certificate, and who recognizes that recognizer? The answer is not an external parser. The discipline uses a layered bootstrap:*

$$C_0,\ C_1,\ C_2,\ C_3,\ C_4,\ C_5,\ C_\infty,$$

where the lower layers cover channel decoding and basic event-flow recognition, later layers cover package, certificate, theorem, chapter, and compiler recognition, and the limiting target is a self-hosted compiler flow. Lower recognizers recognize finite certificate structure for higher recognizers, retaining ledgers and failure records at each step.

Definition 9215.8 (Certified recognizer flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Certifie`

A recognizer candidate flow R is a certified recognizer flow *for role ρ, written*

$$\mathsf{CertifiedRecognizer}(R, \rho),$$

when there exist a recognizer certificate flow C_R and meta-recognizer flow Q such that

$$\mathsf{RecognizesRecognizerCert}(Q, C_R, R, \rho)$$

and C_R is a sound recognizer-certificate flow.

Definition 9215.9 (Recognizer certificate fields). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Recogn`

If

$$\mathsf{RecognizesRecognizerCert}(Q, C_R, R, \rho),$$

then Q may recognize subflows of C_R as the certificate roles

$$\begin{array}{lll} \mathsf{RecDom}_Q(C_R, D), & \mathsf{RecTarget}_Q(C_R, T), & \mathsf{RecRole}_Q(C_R, \rho), \\ \mathsf{RecSoundness}_Q(C_R, \mathrm{Snd}), & \mathsf{RecCompleteness}_Q(C_R, \mathrm{Cmp}), & \mathsf{RecLedger}_Q(C_R, L), \\ \mathsf{RecFailure}_Q(C_R, F), & \mathsf{RecStability}_Q(C_R, \mathrm{St}), & \mathsf{RecSeal}_Q(C_R, \sigma_R). \end{array}$$

These are not external field labels. They are subflow roles recognized by Q inside C_R.

Definition 9215.10 (Recognizer soundness). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.RecSound`

A recognizer R is sound *for role ρ when every flow it recognizes under that role satisfies the structure required by the role:*

$$\mathsf{RecSound}(R, \rho) \;:\Longleftrightarrow\; \forall S,\ \mathsf{Recognizes}(R, \rho, S) \Rightarrow \mathsf{RoleOK}(\rho, S).$$

For example, a sound NameCert *recognizer may recognize only flows that contain the required source, pattern, classifier, stability, and ledger structure and satisfy the associated soundness checks.*

Definition 9215.11 (Recognizer completeness). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.RecComp`

A recognizer R is complete relative to a domain D *for role ρ when*

$$\mathsf{RoleOK}(\rho, S) \wedge S \in D \Longrightarrow \mathsf{Recognizes}(R, \rho, S).$$

Completeness is always relative to a recognized domain flow. No finite recognizer is required to recognize every possible structure without a specified domain.

Definition 9215.12 (Recognizer failure ledger). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.RecFail`

A recognizer-certificate flow must contain a failure flow

$$\mathsf{RecFailure}_Q(C_R, F)$$

recording why candidate flows are rejected: format failure, missing field, missing ledger, soundness failure, status mismatch, or ill-founded dependency. This is the recognizer-level form of cannot-claim and audit discipline.

Definition 9215.13 (Recognizer ledger). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.RecLedger`

A recognizer-certificate flow must contain a ledger flow

$$\mathsf{RecLedger}_Q(C_R, L)$$

recording the source events used by R, the subflows assigned to roles, the source material compressed by recognition, the raw and normal forms related by the recognizer, and the failure branches excluded. If a recognizer hides source structure without such a ledger, it is not admissible.

Principle 9215.14 (No recognizer without certificate). *Lean4✓ BEDC.GroundCompiler.RecognizerFlows.formal recognition evidence requires certified*

A recognizer flow may be used as a formal recognizer only when it is certified for the relevant role. Formal recognition evidence is the paired datum consisting of the recognition judgment

$$\mathsf{Recognizes}(R, \rho, S)$$

and the certificate judgment

$$\mathsf{CertifiedRecognizer}(R, \rho).$$

Hence any formal recognition evidence for R, ρ, and S projects to $\mathsf{CertifiedRecognizer}(R, \rho)$.

Theorem 9215.15 (Uncertified recognizer cannot license recognition). *Lean4✓ BEDC.GroundCompiler.Recognizer cannot license*

If $\neg\mathsf{CertifiedRecognizer}(R, \rho)$*, then there is no formal recognition evidence for R, ρ, and S. Equivalently, the system cannot use the bare judgment*

$$\mathsf{Recognizes}(R, \rho, S)$$

as formal evidence that S has role ρ.

Proof. By Definition 9215.14, formal recognition requires a certified recognizer. If R is not certified for ρ, its output has no recognized soundness, completeness, ledger, or failure-flow support. Hence that output cannot be used as formal evidence for the role. □

Corollary 9215.16 (Host parser output is inadmissible). *Lean4✓ BEDC.GroundCompiler.RecognizerFlows.host parser output inadmissible*

Recognition supplied only by a host parser, YAML schema, external grammar, Lean meta-level elaborator, or file manifest is not admissible as a BEDC recognition judgment unless it is backed by a certified recognizer flow.

9215.3 Conservativity and hierarchy

Definition 9215.17 (Recognizer conservativity). *Lean4 (definition) BEDC.GroundCompiler.RecognizerFlows.Recogn*

A certified recognizer R is kernel-conservative *for role ρ when* $\mathsf{Recognizes}(R, \rho, S)$ *does not add new raw marks, history constructors, sameness rules, continuation rules, or derived-interface formation rules.*

Theorem 9215.18 (Certified recognizers are recognition-only). *Lean4✓ BEDC.GroundCompiler.RecognizerFlows. recognition only*

If $\mathsf{CertifiedRecognizer}(R, \rho)$ *and its certificate contains a kernel-conservativity subflow, then the recognition judgments of R do not change the finite kernel.*

Proof. The kernel-conservativity subflow records that R recognizes structure inside source event flows rather than generating new kernel rules. It does not add b0 or b1, does not add msame, hsame, or psame rules, does not change Ext or Cont, and does not license derived interfaces. Derived interfaces still require NameCert flow; accepted objects still require acceptance-gate flow. Hence R is conservative over the finite kernel. □

Definition 9215.19 (Recognizer hierarchy). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Recognizer`

A recognizer hierarchy *is a finite list of event flows*

$$R_0, R_1, \ldots, R_n$$

with corresponding roles

$$\rho_0, \rho_1, \ldots, \rho_n$$

such that R_0 handles channel decoding and event-flow recognition, each recognizer certificate for R_{i+1} is recognized by R_i or earlier recognizers, each R_i recognizes only the finite structure required by its role, and every level retains ledger and failure-flow data.

Definition 9215.20 (Bootstrap base recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Bootst`

The bootstrap base recognizer *R_0 recognizes legal channel streams and event flows:*

$$\mathsf{RecognizesLegalZStream}(R_0, c), \qquad \mathsf{RecognizesEventFlow}(R_0, S).$$

It corresponds to the channel compiler and decoder bijection

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}.$$

Its role covers event boundaries, raw 0, raw 1, legal event codes, and legal event flows.

Proposition 9215.21 (Bootstrap base does not recognize mathematical structure). *Lean4 ✓* `BEDC.GroundCompiler` `base no math structure`

The base recognizer R_0 recognizes only channel legality and event-flow structure. It does not recognize packages, NameCert *flows, theorem flows, chapter flows, proof flows, accepted objects, or mature interfaces.*

Proof. The role of R_0 is channel grammar and event-flow decoding. Packages, certificates, theorems, chapters, and accepted objects require higher recognizers with field recognition, ledger recognition, soundness checks, and failure flows. Those roles are not part of the base recognizer. □

Definition 9215.22 (Package recognizer level). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Package`

A package recognizer level *is a level R_i in a recognizer hierarchy whose role is package recognition. It supports judgments of the form*

$$\mathsf{RecognizesPkg}(R_i, S),$$

but only after $\mathsf{CertifiedRecognizer}(R_i, \rho_{\mathrm{pkg}})$ *has been established.*

Definition 9215.23 (Certificate recognizer level). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Certi`

A certificate recognizer level *is a level R_i in a recognizer hierarchy that recognizes* NameCert *flows, derivation-certificate flows, or acceptance flows. It supports judgments such as*

$$\mathsf{RecognizesNameCert}(R_i, S, N),$$
$$\mathsf{RecognizesDerivCert}(R_i, D, N, s),$$
$$\mathsf{RecognizesAcceptance}(R_i, A, N, s).$$

Definition 9215.24 (Theorem recognizer level). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Theorem`

A theorem recognizer level *is a level R_i in a recognizer hierarchy that recognizes theorem flows:*

$$\mathsf{RecognizesTheorem}(R_i, T).$$

Its certificate records soundness for statement, proof, dependency, certificate, ledger, status, canonical-site, and theorem-seal recognition.

Definition 9215.25 (Chapter recognizer level). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Chapter`

A chapter recognizer level *is a level R_i in a recognizer hierarchy that recognizes chapter flows:*

$$\mathsf{RecognizesChapter}(R_i, C).$$

Its certificate records soundness for definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and chapter-seal recognition.

Theorem 9215.26 (Recognizer hierarchy removes structural hidden input). *Lean4 ✓* `BEDC.GroundCompiler.Recog` `hierarchy removes hidden input`

If the recognition judgments for package, NameCert*, derivation-certificate, theorem, and chapter structures are supplied by certified recognizers in a recognizer hierarchy, then these structures are not hidden structural inputs.*

Proof. Hidden structural input is structure that enters the system without being generated as event flow and without being recognized by generated recognizers. In a certified recognizer hierarchy, every such structural use is supported by some event flow R_i and its recognizer certificate. The structure is therefore recognized from event-flow data rather than supplied as external input. □

Corollary 9215.27 (YAML, ASTs, and theorem labels remain inadmissible). *Lean4 ✓* `BEDC.GroundCompiler.Reco` `views inadmissible`

Even in the presence of a recognizer hierarchy, YAML files, ASTs, external theorem labels, and external package records remain inadmissible as formal inputs. They may at most be human-readable output views generated from decoded event flows.

9215.4 Self-description and self-hosting

Definition 9215.28 (Recognizer self-description flow). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.R`

A self-description flow *for a recognizer R is an event flow*

$$D_R \in \mathsf{EventFlow}$$

recording which role R recognizes, which domain it requires, which recognition judgments it returns, which ledger and failure branches it records, which earlier recognizer certifies it, and which kernel constraints it preserves. If D_R is recognized by R or by a later compiler recognizer, then R has self-description structure.

Definition 9215.29 (Compiler recognizer flow). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Compile`

A compiler recognizer flow *is an event flow C that recognizes channel compiler definitions, event encoder and decoder structure, the recognizer hierarchy, recognizer certificates, theorem and chapter code construction, and self-description flows. This is written*

$$\mathsf{RecognizesCompiler}(C).$$

Definition 9215.30 (Compiler behavior classifier). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.Comp`

Define the compiler behavior classifier by

$$C \equiv_{\text{comp}} C'$$

when C and C' produce the same recognized outputs on all recognized event-flow compilation tasks and preserve the same ledger and failure-flow behavior. This is not raw event-flow equality and not hsame*; it is a classifier for compiler behavior.*

Definition 9215.31 (Self-hosting compiler flow). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.SelfHo`

A compiler recognizer flow C is self-hosting *when* $\mathsf{RecognizesCompiler}(C)$*, C compiles its own compiler flow to some C', the result is behaviorally equivalent to C, and the self-compilation ledger records raw C, compiled code, decoded C', and the behavior-classifier witness. Formally,* $\mathsf{SelfHosting}(C)$ *holds when there exists C' such that*

$$\mathsf{Compiles}_C(C, C') \qquad \textit{and} \qquad C' \equiv_{\text{comp}} C.$$

Proposition 9215.32 (Self-hosting is not kernel equality). *Lean4✓* `BEDC.GroundCompiler.RecognizerFlows.self hosting projects behavior classifier`

If $\mathsf{SelfHosting}(C)$*, then there exists a compiled flow C' together with a formal self-compilation judgment* $\mathsf{Compiles}_C(C, C')$ *and a compiler-behavior classifier witness*

$$C' \equiv_{\text{comp}} C.$$

Thus self-hosting records compiler-behavior stability through a classifier witness; it does not record raw event-flow equality or kernel-level sameness as the self-hosting datum.

Proof. Definition 9215.31 requires $C' \equiv_{\text{comp}} C$. By Definition 9215.30, $\equiv_{\text{comp}}$ is a behavior classifier, not raw equality or kernel sameness. Hence self-hosting does not imply $C = C'$ as raw event flows. □

Theorem 9215.33 (Self-hosting removes compiler-as-hidden-input). *Lean4✓* `BEDC.GroundCompiler.RecognizerFl hosting removes compiler as hidden input`

If there is a self-hosting compiler flow C, and its recognizer hierarchy recognizes package, NameCert*, derivation-certificate, theorem, chapter, and compiler flows, then the compiler is not a host-level hidden input.*

Proof. Under self-hosting, the compiler C is itself an event flow and its compiler status is recognized by $\mathsf{RecognizesCompiler}(C)$. It compiles itself to a behavior-equivalent flow C' and records the self-compilation ledger. Thus its structure, behavior, and reuse discipline are supported by event-flow recognition and behavior classification, not by a host-level parser or external compiler assumption. □

Corollary 9215.34 (Full no-hidden-input needs self-hosting or a bootstrap boundary). *Lean4✓* `BEDC.GroundCompiler.RecognizerFlows.full no hidden input needs self hosting`

Before a self-hosting compiler flow is available, the system cannot assert a full no-hidden-input self-hosting claim. A full no-hidden-input claim must include a self-hosting compiler flow; any remaining external compiler dependence belongs in the bootstrap-obligation layer.

Definition 9215.35 (Bootstrap obligation). *Lean4 (definition)* `BEDC.GroundCompiler.RecognizerFlows.BootstrapO`

A bootstrap obligation *is a cannot-claim style record saying that a recognizer or compiler component still depends on an external implementation or on structure not yet discharged by self-hosting event flow. It records claim text, status, why the obligation is not discharged, and the required upgrade.*

Theorem 9215.36 (Bootstrap obligation blocks full self-hosting claim). *Lean4✓* `BEDC.GroundCompiler.Recognize obligation blocks full self hosting claim`

If any bootstrap obligation remains undischarged, the system cannot claim that the compiler is fully no-hidden-input self-hosting.

Proof. A full self-hosting claim requires compiler and recognizer structure to be generated, recognized, and self-hosted as event-flow structure. An undischarged bootstrap obligation records that some recognizer or compiler component still depends on an external implementation or uncertified structure. Therefore the full no-hidden-input self-hosting claim is not available. □

9215.5 Examples and conservativity

Example 9215.37 (Channel compiler as base recognizer)**.** *The channel compiler and decoder can serve as R_0. They recognize raw* 0 *as channel* 0*, raw* 1 *as channel* 10*, event end as channel* 11*, and the bijection between legal streams and event flows. Their soundness theorem is* $\mathsf{Decode}(\mathsf{Compile}(S)) = S$*, and their completeness theorem is that legal* c *satisfies* $\mathsf{Compile}(\mathsf{Decode}(c)) = c$*. This base does not recognize packages, theorems, or* NameCert *flows.*

Example 9215.38 (NameCert recognizer certificate)**.** *A* NameCert *recognizer certificate records a domain of certificate-candidate flows, target* NameCert *flows, the role of* NameCert *recognition, soundness saying that recognized flows have source, pattern, classifier, stability, and ledger structure, domain-relative completeness, ledgers for the field and seal subflows, failure branches for missing fields and malformed seals, stability under compile-decode round trip, and a recognizer-certificate seal.*

Example 9215.39 (Theorem recognizer certificate)**.** *A theorem recognizer certificate records theorem-candidate flows, theorem-flow targets, theorem-recognition role, soundness for statement, dependencies, proof, certificates, ledger, status, canonical site, and theorem seal, domain-relative completeness, ledgers for statement, proof, dependency, and certificate subflows, failure branches for missing statement, missing proof, bad proof checker, missing ledger, status mismatch, and missing canonical site, plus compile-decode stability and a recognizer-certificate seal.*

Example 9215.40 (Chapter recognizer certificate)**.** *A chapter recognizer certificate records chapter-candidate flows, recognized chapter-flow targets, chapter-recognition role, soundness for start, dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and seal, domain-relative completeness, ledgers for dependency order and recognized lists, failure branches for forward dependency without marker, theorem without proof, missing certificate, missing ledger, and missing cannot-claim entry, plus compile-decode stability and a recognizer-certificate seal.*

Theorem 9215.41 (Recognizer flows are conservative over the finite kernel)**.** `Lean4✓ BEDC.GroundCompiler.Recog flows conservative over kernel`

Adding certified recognizer flows, recognizer-certificate flows, and self-hosting compiler flows does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.

Proof. Recognizer flows are event flows. Recognition judgments are relations over event flows and certificates of such relations. They do not generate new b0 or b1, do not add msame, hsame, or psame rules, do not alter Ext or Cont, and do not license derived interfaces by themselves. Derived interfaces still require recognized NameCert flows; exports still require acceptance-gate flows. Self-hosting establishes behavior classification of compiler flows, not kernel equality. Hence recognizer flows are conservative over the finite kernel. □

Corollary 9215.42 (Recognizer existence does not imply recognized object existence)**.** *The existence of a recognizer flow R does not imply the existence of an event flow S such that* $\mathsf{Recognizes}(R, \rho, S)$*. A recognizer flow is a possible recognizer; actual recognized structures still require target event flows and recognition judgments.*

Corollary 9215.43 (Recognition does not imply acceptance)**.** `Lean4✓` `BEDC.GroundCompiler.RecognizerFlows.re` `not acceptance`

Even if R recognizes S as a package, NameCert*, theorem, or chapter flow, acceptance of mature objects still requires recognized* NameCert *flow, recognized derivation-certificate flow, and acceptance-gate flow where applicable.*

Remark 9215.44 (Outlook)**.** *This chapter establishes that recognizers are not external parsers, but event flows. The core judgments are* $\mathsf{Recognizes}(R, \rho, S)$ *and* $\mathsf{CertifiedRecognizer}(R, \rho)$*. A recognizer candidate flow is not automatically a recognizer; formal use requires a recognizer certificate with domain, target, role, soundness, completeness, ledger, failure, stability, and seal subflows. Uncertified recognizers and host-parser outputs are inadmissible. A certified recognizer hierarchy removes hidden structural input for package, certificate, theorem, and chapter recognition. Self-hosting compiler flow removes compiler-as-hidden-input, but self-hosting is behavior classification, not kernel equality. Undischarged bootstrap obligations block full self-hosting claims. Recognizer flows remain conservative over the finite kernel, recognizer existence does not imply recognized-object existence, and recognition does not imply acceptance. The next ground-compiler chapter gives the detailed self-hosting construction path from the channel base recognizer to a compiler flow that can compile and recognize itself.*

Lemma 9215.45 (Certified recognizer certificates expose ledger and failure)**.** *Let R, ρ, S, C_R, and Q be event flows. Suppose there is formal recognition evidence for R, ρ, and S, and suppose its certificate side is backed by*

$$\mathsf{RecognizesRecognizerCert}(Q, C_R, R, \rho).$$

Then there are event flows L and F such that

$$\mathsf{RecLedger}_Q(C_R, L),$$
$$\mathsf{RecFailure}_Q(C_R, F).$$

In particular, the evidence cannot use R as a formal recognizer while hiding either the recognizer ledger or the recognizer failure record.

Proof. By Definition 9215.14, formal recognition evidence for R, ρ, and S includes the certificate judgment $\mathsf{CertifiedRecognizer}(R, \rho)$. By Definition 9215.8, such a certification is witnessed by a recognizer-certificate flow and a meta-recognizer flow, and the assumed judgment

$$\mathsf{RecognizesRecognizerCert}(Q, C_R, R, \rho)$$

identifies C_R as the certificate flow recognized by Q for this role. The recognized certificate field surface in Definition 9215.9 includes the ledger and failure roles. The ledger requirement in Definition 9215.13 supplies an event flow L with $\mathsf{RecLedger}_Q(C_R, L)$, and the failure-ledger requirement in Definition 9215.12 supplies an event flow F with $\mathsf{RecFailure}_Q(C_R, F)$. These are finite subflows of the same recognized recognizer certificate C_R, so the formal recognition evidence exposes both fields. □

Chapter 9216

Self-Hosting Compiler Flow

chapter 9215 records the no-hidden-input reading of recognizers: a recognizer is not an external parser, but an event flow. This raises the next question. If package, NameCert, derivation-certificate, theorem, and chapter recognizers are themselves generated event flows, who recognizes those recognizers?

The answer is layered bootstrapping, with an explicit boundary until a self-hosting compiler flow is available. The governing rule is

the compiler is not an external program; ultimately it must be a recognizable, compilable, reproducible event flo

The intended ladder is

channel compiler $\longrightarrow$ event-flow recognizer $\longrightarrow$ package recognizer $\longrightarrow$ certificate recognizers $\longrightarrow$ theorem and chapter recognizers $\longrightarrow$ compiler recognizer $\longrightarrow$ self-hosting compiler flow.

This chapter does not claim that the base is already fully self-hosted. It distinguishes a visible bootstrap boundary from self-hosting discharge. Any compiler component not yet discharged by event-flow self-hosting must be recorded as a bootstrap obligation, not hidden in the implementation layer.

9216.1 Compiler candidates and behavior

Definition 9216.1 (Compiler candidate flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.`

A compiler candidate flow *is an event flow*

$$C \in \mathsf{EventFlow}.$$

It has not yet been accepted as a compiler. Thus $C \in \mathsf{EventFlow}$ does not imply that C compiles event flows, recognizes packages, recognizes theorems, or recognizes itself. Such capabilities must be supported by compiler recognizer and compiler certificate flows.

Definition 9216.2 (Compiler recognition relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerF`

The compiler recognition relation is written

$$\mathsf{RecognizesCompiler}(R_C, C), \qquad R_C, C \in \mathsf{EventFlow}.$$

It is read as: the generated compiler recognizer flow R_C recognizes the event flow C as a compiler flow. If some R_C satisfies this relation, then C is a recognized compiler flow.

Definition 9216.3 (Compiler behavior relation). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlo`
The compiler behavior relation is written

$$\mathsf{Compiles}_C(S, T), \qquad C, S, T \in \mathsf{EventFlow}.$$

It is read as: compiler flow C compiles input event flow S into output event flow T. The compiler, input, and output are all event flows, not host programs, ASTs, or package objects.

Remark 9216.4 (Compiler behavior is a source-level relation). *The judgment* $\mathsf{Compiles}_C(S, T)$ *is a source-level relation over event flows. It is not a channel-level bit rewrite and not a host-level function call. To obtain channel presentations of S and T, one applies* $\mathsf{Compile}(S)$ *and* $\mathsf{Compile}(T)$ *separately.*

Definition 9216.5 (Compiler task domain). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.Co`
The task domain *of a compiler flow C is a recognized event-flow class*

$$\mathsf{TaskDom}(C, D),$$

where $D \in$ EventFlow *represents the input-flow range that C is licensed to handle. A channel compiler has only the event-flow and channel-flow task domain; it does not thereby handle theorem recognition or* NameCert *recognition.*

Definition 9216.6 (Compiler target domain). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.C`
The target domain *of a compiler flow C is a recognized event-flow class*

$$\mathsf{TargetDom}(C, T),$$

recording the kind of output flow C is meant to produce, such as channel-code flow, package-recognizer flow, NameCert*-recognizer flow, theorem-recognizer flow, or compiler flow.*

9216.2 Compiler certificates

Definition 9216.7 (Compiler certificate candidate flow). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingComp`
A compiler certificate candidate flow *is an event flow*

$$K_C \in \mathsf{EventFlow}.$$

It claims to supply compilation-capability evidence for a compiler candidate C, but has not yet been recognized as a sound compiler certificate.

Definition 9216.8 (Compiler certificate recognition). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompile`
The compiler-certificate recognition relation is written

$$\mathsf{RecognizesCompilerCert}(R_K, K_C, C), \qquad R_K, K_C, C \in \mathsf{EventFlow}.$$

It is read as: the generated recognizer flow R_K recognizes the event flow K_C as a compiler certificate for compiler flow C.

Definition 9216.9 (Compiler certificate fields). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow`
If

$$\mathsf{RecognizesCompilerCert}(R_K, K_C, C),$$

then R_K may recognize subflows of K_C as compiler-certificate roles:

$$\begin{aligned}
&\mathsf{CompInputDom}_{R_K}(K_C, D_{\text{in}}), && \mathsf{CompOutputDom}_{R_K}(K_C, D_{\text{out}}),\\
&\mathsf{CompEncodingRule}_{R_K}(K_C, E), && \mathsf{CompDecodingRule}_{R_K}(K_C, D),\\
&\mathsf{CompRoundTrip}_{R_K}(K_C, P), && \mathsf{CompRecognizerHierarchy}_{R_K}(K_C, H),\\
&\mathsf{CompLedger}_{R_K}(K_C, L), && \mathsf{CompFailure}_{R_K}(K_C, F),\\
&\mathsf{CompKernelConservativity}_{R_K}(K_C, K), && \mathsf{CompSelfDescription}_{R_K}(K_C, \mathrm{SD}),\\
&\mathsf{CompSeal}_{R_K}(K_C, \sigma_C).
\end{aligned}$$

These are not external field labels. They are subflow roles recognized by R_K inside K_C.

Definition 9216.10 (Certified compiler flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.C`

A compiler candidate C is a certified compiler flow*, written*

$$\mathsf{CertifiedCompiler}(C),$$

when there exist a compiler certificate flow K_C and recognizer R_K such that

$$\mathsf{RecognizesCompilerCert}(R_K, K_C, C)$$

and the recognized certificate flow is sound.

Principle 9216.11 (No compiler without compiler certificate)**.** *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompil` `compilation requires certificate`

A compiler flow may be used as a formal compiler only when it is certified. Before the judgment

$$\mathsf{Compiles}_C(S, T)$$

is used formally, the system must have

$$\mathsf{CertifiedCompiler}(C).$$

Otherwise C remains only a compiler candidate flow.

Theorem 9216.12 (Uncertified compiler cannot be used as compiler)**.** *Lean4 ✓* `BEDC.GroundCompiler.SelfHosting` `cannot compile`

If $\neg\mathsf{CertifiedCompiler}(C)$, then for every compiler behavior relation

$$\mathsf{Compiles}_C(S, T)$$

the pair

$$\mathsf{CertifiedCompiler}(C) \;\wedge\; \mathsf{Compiles}_C(S, T)$$

cannot hold. Thus the system cannot use $\mathsf{Compiles}_C(S, T)$ as formal evidence that C compiles S into T unless the certificate component of the formal compiler judgment is present.

Proof. By Definition 9216.11, a formal compiler judgment contains the certified-compiler component together with the behavior component. If $\neg\mathsf{CertifiedCompiler}(C)$, then the certified-compiler component is impossible, so the conjunction $\mathsf{CertifiedCompiler}(C) \wedge \mathsf{Compiles}_C(S, T)$ is impossible. Hence $\mathsf{Compiles}_C(S, T)$ cannot be used as a formal compilation judgment by itself. □

Corollary 9216.13 (Host compiler output is inadmissible without event-flow certificate)**.** *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompilerFlow.host output inadmissible without certificate`

If a compilation result comes only from a host compiler, script, external parser, build tool, YAML manifest, or meta-level program, and has no certified compiler flow, then it is not admissible as a formal BEDC compiler judgment.

9216.3 Bootstrap boundary

Definition 9216.14 (Bootstrap compiler). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.Boo`

A bootstrap compiler *is a compiler component explicitly recorded as part of the bootstrap boundary before self-hosting is discharged. It is written*

$$\mathsf{Bootstrap}(C, \rho),$$

where ρ records the bootstrap role, such as channel encoder and decoder, first event-flow recognizer, or first recognizer-certificate checker.

Definition 9216.15 (Bootstrap obligation). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.Bo`

A bootstrap obligation *is an event flow B whose recognized content records the bootstrap component, its role, why it is assumed, which part is not yet self-hosted, the required discharge condition, status, and ledger. It is written*

$$\mathsf{BootstrapObligation}(B, C, \rho).$$

Principle 9216.16 (Bootstrap must be explicit). *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompilerFlow.boots` `must be explicit`

Before full self-hosting is established, every compiler component that depends on external implementation must be recorded by a bootstrap obligation. The component cannot be hidden in host functions, YAML, Lean meta code, proof scripts, or implementation details.

Theorem 9216.17 (Hidden bootstrap violates no-hidden-input discipline). *Lean4 ✓* `BEDC.GroundCompiler.SelfHo` `bootstrap violates`

If a compiler component C is used in a formal compilation judgment, but is neither a certified compiler flow nor recorded by a bootstrap obligation, then the system violates the no-hidden-input discipline.

Proof. Formal compilation judgments are supported by certified compiler flows, except at an explicit bootstrap boundary. If C is neither certified nor recorded as a bootstrap obligation, then the source of its behavior is outside the event-flow system and has no ledger. It is therefore a hidden input. □

9216.4 Self-compilation and self-hosting

Definition 9216.18 (Compiler behavior classifier). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerF`

The compiler behavior classifier is written

$$C \equiv_{\mathrm{comp}} C'$$

when C and C' have the same recognized behavior on the same recognized task domain and preserve the same ledger and failure-flow behavior. The classifier records input-domain agreement, output-domain agreement, encoding behavior, decoding behavior, round-trip theorem behavior, failure-flow behavior, ledger behavior, and kernel-conservativity behavior. It is not raw event-flow equality, kernel equality, or hsame.

Definition 9216.19 (Self-compilation). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.SelfCo`

A compiler flow C self-compiles *when there exists an event flow C' such that*

$$\mathsf{Compiles}_C(C, C') \qquad \textit{and} \qquad C' \equiv_{\mathrm{comp}} C.$$

This is written $\mathsf{SelfCompiles}(C)$.

Definition 9216.20 (Self-hosting compiler flow). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlo`

A compiler flow C is self-hosting, *written* $\mathsf{SelfHosting}(C)$, *when:*

1. $\mathsf{CertifiedCompiler}(C)$;
2. $\mathsf{RecognizesCompiler}(R_C, C)$ *for some generated compiler recognizer* R_C;
3. $\mathsf{SelfCompiles}(C)$;
4. *the self-compilation ledger records raw C, compiled code, decoded output C', and the behavior-classifier witness $C' \equiv_{\text{comp}} C$.*

Proposition 9216.21 (Self-hosting is not raw equality). *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompilerFlou` *`hosting yields behavior classifier`*

If $\mathsf{SelfHosting}(C)$, *then for some event flow C' one obtains*

$$C' \equiv_{\text{comp}} C,$$

as a compiler-behavior classifier witness. The self-hosting judgment therefore exposes behavior classification as its equality-like datum, rather than raw event-flow equality.

Proof. Definition 9216.20 contains a self-compilation output C' together with the behavior-classifier witness $C' \equiv_{\text{comp}} C$. By Definition 9216.18, this classifier is a relation on compiler behavior, not raw event-flow equality. □

Corollary 9216.22 (Self-hosting does not imply kernel equality). *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingComp` *`hosting not kernel equality`*

Since $C' \equiv_{\text{comp}} C$ is a compiler-behavior classifier, self-hosting asserts no new kernel-level equality, hsame, *or* msame *judgment.*

Definition 9216.23 (Self-hosting ledger). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.Selfl`

A self-hosting ledger *for C is an event flow L_C recording raw compiler flow C, compiled channel code* $\mathsf{Compile}(C)$, *decoded output flow C', the compiler-behavior classifier witness $C' \equiv_{\text{comp}} C$, failure branches, discharged or remaining bootstrap obligations, and kernel-conservativity witness. This is written*

$$\mathsf{SelfHostingLedger}(C, L_C).$$

Principle 9216.24 (No self-hosting without ledger). *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompilerFlow.se` *`hosting requires ledger`*

A self-hosting claim requires a self-hosting ledger:

$$\mathsf{SelfHosting}(C) \Longrightarrow \exists L_C,\ \mathsf{SelfHostingLedger}(C, L_C).$$

This is the compiler-level form of the BEDC ledger discipline: recognition, compression, and behavior equivalence must retain provenance.

Theorem 9216.25 (Self-hosting removes compiler-as-hidden-input). *Lean4 ✓* `BEDC.GroundCompiler.SelfHosting(` *`hosting removes hidden input`*

If $\mathsf{SelfHosting}(C)$ *and the recognizer hierarchy of C recognizes package,* NameCert, *derivation-certificate, theorem, chapter, and compiler flows, then the compiler is no longer a host-level hidden input.*

Proof. Self-hosting gives a certified compiler flow C and a self-compilation output C' with $C' \equiv_{\text{comp}} C$. The self-hosting ledger records the raw flow, compiled code, decoded output, behavior equivalence, and failure behavior. The recognizer hierarchy recognizes package, certificate, theorem, chapter, and compiler structures by generated recognizer flows. The system therefore no longer needs a host compiler, parser, AST, or manifest as formal input; compiler behavior is supported by event-flow certificate and self-compilation ledger. □

Corollary 9216.26 (Full no-hidden-input claim requires self-hosting or declared bootstrap)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.full claim requires boundary`

If SelfHosting(C) *has not been established, then the system cannot claim a full no-hidden-input compiler unless every non-self-hosted component is recorded as a bootstrap obligation.*

9216.5 Bootstrap ladder

Definition 9216.27 (Channel compiler C_0)**.** Lean4 (definition) `BEDC.GroundCompiler.SelfHostingCompilerFlow.Ch`

The lowest channel compiler C_0 is the event-flow table

$$C_0 := (0,\ 10,\ 11).$$

Its three raw events record the channel codewords

$$\textit{raw } 0 \mapsto 0, \qquad \textit{raw } 1 \mapsto 10, \qquad \textit{event end} \mapsto 11,$$

and its channel-only behavior sends an input event flow S to the one-event flow whose raw event is Compile(S)*. This behavior supports the bijection*

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}.$$

It recognizes legal event codes, event boundaries, raw event reconstruction, and flow round trips. It does not recognize packages, NameCert *flows, derivation-certificate flows, theorem flows, chapter flows, or compiler flows.*

Proposition 9216.28 (C_0 is not a full compiler)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.channe` `compiler not full`

The channel compiler C_0 is not a full self-hosting compiler under the channel-only behavior above.

Proof. The behavior of C_0 is limited to event-channel encoding and decoding. When applied to C_0 itself, it returns the channel code flow Compile(C_0) rather than a compiler flow behaviorally classifying back to C_0. Thus the self-hosting classifier required for a full compiler cannot be obtained from the channel-only behavior. □

Definition 9216.29 (Compiler bootstrap ladder)**.** Lean4 (definition) `BEDC.GroundCompiler.SelfHostingCompilerFl`

A compiler bootstrap ladder *is a sequence*

$$C_0, C_1, C_2, C_3, C_4, C_5$$

with roles:

C_0 : *channel compiler,*
C_1 : *event-flow and basic recognizer compiler,*
C_2 : *package recognizer compiler,*
C_3 : NameCert *and derivation-certificate recognizer compiler,*
C_4 : *theorem and chapter recognizer compiler,*
C_5 : *compiler recognizer and self-hosting compiler.*

Each transition to C_{i+1} must be compiled or recognized using prior stages and must carry a certificate flow documenting the transition.

Definition 9216.30 (Ladder soundness). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlow.Ladde

A compiler bootstrap ladder is sound *when every transition*

$$C_i \to C_{i+1}$$

has an event-flow certificate containing input domain, output domain, compiler behavior, round-trip or recognition soundness, ledger, failure-flow, kernel-conservativity, status, and seal.

Theorem 9216.31 (Sound ladder discharges staged hidden inputs). Lean4✓ BEDC.GroundCompiler.SelfHostingC ladder discharges

If $C_0, \ldots, C_5$ is a sound compiler bootstrap ladder and C_5 is a self-hosting compiler flow, then the lower recognizer and compiler structures are generated, recognized, compiled, or self-hosted by event-flow structure rather than supplied as external hidden inputs.

Proof. By ladder soundness, each stage C_{i+1} is supported by prior stages and a transition certificate flow. By self-hosting of C_5, the final compiler compiles and behaviorally reproduces itself. Thus the recognizer hierarchy and compiler behavior are supported by event flows and ledgers rather than by an external parser, AST, manifest, or host compiler as formal input. □

Definition 9216.32 (Remaining bootstrap boundary). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompil

Before C_5 is self-hosting, the remaining bootstrap boundary *is the recorded collection of components not yet discharged by the sound ladder:*

$$\mathsf{BootBoundary} = \{\, C_i \mid C_i \text{ is not yet certified or not yet self-hosted, and has a bootstrap-obligation flow} \,\}.$$

Principle 9216.33 (Bootstrap boundary must be visible). Lean4✓ BEDC.GroundCompiler.SelfHostingCompilerFl bootstrap boundary visible

Every element of BootBoundary *must have a bootstrap-obligation event flow. Hidden bootstrap is not allowed.*

Theorem 9216.34 (Invisible bootstrap blocks compiler certification). Lean4✓ BEDC.GroundCompiler.SelfHosting bootstrap blocks certification

If some $C_i \in$ BootBoundary lacks a bootstrap-obligation flow, then the compiler part cannot be certified as no-hidden-input.

Proof. A component in the bootstrap boundary depends on external implementation or unproven recognition behavior. If it lacks a bootstrap-obligation flow, that dependence is not recorded in the event-flow ledger. Hidden input remains, so no-hidden-input certification fails. □

9216.6 Compiler theorem packages

Definition 9216.35 (Compiler theorem package). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFl

A compiler theorem package *is a recognized theorem flow whose statement is witnessed as belonging to a compiler-theorem topic. The topic may be channel round trip, decode-encode behavior, recognizer soundness, recognizer completeness, compiler self-hosting, or bootstrap discharge. Each such package must be a recognized theorem flow in the sense of chapter 9213.*

Proposition 9216.36 (Compiler theorem code is theorem-flow code). Lean4✓ `BEDC.GroundCompiler.SelfHosting theorem code is theorem code`

The code of a compiler theorem is

$$\mathsf{Compile}(T),$$

where T is the recognized compiler theorem flow. It is not a host theorem name, Lean theorem identifier, or theorem label.

Proof. By Definition 9213.25, theorem code is the channel compilation of theorem flow. Compiler theorems are theorem flows whose statements concern compiler behavior. Therefore their code is $\mathsf{Compile}(T)$. □

Example 9216.37 (Channel round-trip compiler theorem). *A channel round-trip theorem flow may contain statement flow* $\mathsf{Decode}(\mathsf{Compile}(S)) = S$*, dependency flows for ζ, event encoding,* Compile*, and* Decode*, a proof flow by induction on event flow, a ledger containing raw S, compiled stream, and decoded stream, a status flow, a canonical-site flow in the channel compiler chapter, and a theorem-seal flow. Its code is* $\mathsf{Compile}(T_{\mathrm{roundtrip}})$.

Example 9216.38 (Recognizer soundness compiler theorem). *A recognizer soundness theorem flow may state that if R recognizes S under role ρ, then S satisfies* $\mathsf{RoleOK}(\rho)$*. Its dependencies include recognizer-certificate definitions, the role definition, and the ledger definition. Its proof flow projects the soundness subflow from the recognizer certificate, and its ledger records R, S, and the certificate flow.*

Example 9216.39 (Self-hosting theorem flow). *A self-hosting theorem flow may state* $\mathsf{SelfHosting}(C)$*. Its dependencies include* $\mathsf{CertifiedCompiler}(C)$*,* $\mathsf{RecognizesCompiler}(R_C, C)$*,* $\mathsf{Compiles}_C(C, C')$*,* $C' \equiv_{\mathrm{comp}} C$*, and* $\mathsf{SelfHostingLedger}(C, L_C)$*. Its ledger records raw C,* $\mathsf{Compile}(C)$*, decoded C', the behavior-classifier witness, and remaining bootstrap obligations.*

Theorem 9216.40 (Self-hosting theorem is not required for channel bijection). Lean4✓ `BEDC.GroundCompiler.Self hosting channel bijection independent`

The channel bijection

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}$$

does not require a self-hosting compiler theorem. It requires only the channel encoder and decoder definitions and their round-trip proofs.

Proof. The channel bijection is established at the C_0 layer by the definitions of ζ, event encoding, Compile, and Decode, plus round-trip proofs. Self-hosting concerns the compiler flow itself as a generated object capable of reproducing its behavior. These are logically distinct. □

Corollary 9216.41 (Channel correctness and no-hidden-input completeness are separate). Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.channel correctness separate`

The system may have a correct channel compiler while still carrying bootstrap obligations for higher recognizers. Full no-hidden-input status requires both channel correctness and discharge of recognizer and compiler bootstrap obligations.

9216.7 Conservativity

Theorem 9216.42 (Compiler-flow recognition is conservative over the finite kernel)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.compiler flow recognition conservativity`

Adding compiler candidate flows, compiler certificate flows, compiler behavior relations, self-hosting relations, and bootstrap obligations does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.

Proof. All compiler-related objects in this chapter are event flows or relations over event flows. They do not generate new b0 or b1, do not add msame, hsame, or psame rules, do not change Ext or Cont, and do not license derived interfaces by themselves. Compiler behavior equivalence is a classifier over compiler flows, not kernel equality. Therefore compiler-flow recognition is conservative over the finite kernel. □

Corollary 9216.43 (Self-hosting compiler is not a BEDC primitive)**.** *Even if* $\mathsf{SelfHosting}(C)$ *holds, C is not a new primitive. It is a certified event flow with behavior classifier and ledger.*

Remark 9216.44 (Outlook)**.** *This chapter establishes that the compiler is not an external program; ultimately it must be an event flow. The core relations are* $\mathsf{CertifiedCompiler}(C)$, $\mathsf{Compiles}_C(S,T)$, $C' \equiv_{\mathrm{comp}} C$, *and* $\mathsf{SelfHosting}(C)$. *A compiler candidate flow is not automatically a compiler; formal use requires a compiler certificate. Hidden bootstrap violates the no-hidden-input discipline. Self-hosting is behavior classification, not raw equality or kernel equality, and it requires a self-hosting ledger. Self-hosting removes compiler-as-hidden-input, while C_0 channel correctness remains separate from full no-hidden-input completeness. A sound bootstrap ladder discharges staged hidden inputs; invisible bootstrap blocks certification. Compiler theorem packages are theorem flows, and compiler-flow recognition is conservative over the finite kernel. The next chapter turns from compiler foundations to semantic motif analysis of event flows and channel codes.*

Chapter 9217

Semantic Motif Analysis

The preceding chapters establish a no-hidden-input compiler discipline:

$$\mathsf{b0}, \mathsf{b1} \text{ event flows} \longrightarrow \text{Zeckendorf-safe channel codes} \longrightarrow \text{generated recognizers}$$
$$\longrightarrow \text{package, certificate, theorem, chapter, and compiler recognition} \longrightarrow \text{self-hosting compiler flow.}$$

This chapter begins to use that discipline to compare mathematical theories at the event-flow layer.

The guiding observation is

traditionally distinct theories may share the same generating motifs in their underlying event flows.

Objects such as $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, categorical composition, completion, real-number structure, circle or phase structure, complex-number structure, and topological structure can have different names and certificates while sharing finite repetition, continuation, seal, carry, classifier quotient, ledger compression, or reuse motifs.

The no-hidden-input rule remains unchanged:

a motif is not an external pattern; it is a stable subflow recognized by a generated motif recognizer.

Thus finite repetition, completion, carry, reuse, classifier, and ledger motifs may not be attached to code by external tags or regular expressions. They must be recognized inside decoded event flows by generated recognizer flows.

9217.1 Motif recognition

Definition 9217.1 (Subflow). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Subflow`

Let

$$S = (w_0, w_1, \ldots, w_{n-1}) \in \mathsf{EventFlow}.$$

A contiguous subflow *is a segment*

$$S[i : j] = (w_i, w_{i+1}, \ldots, w_{j-1}), \qquad 0 \le i \le j \le n.$$

An indexed subflow *is an order-preserving selection*

$$(w_{i_0}, w_{i_1}, \ldots, w_{i_{k-1}}), \qquad i_0 < i_1 < \cdots < i_{k-1}.$$

Contiguous subflows support local motif recognition. Indexed subflows support ledgers, dependencies, reuse, and other cross-position structure.

Definition 9217.2 (Motif candidate flow). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifCandidate`

A motif candidate flow *is a subflow*

$$M \subseteq S$$

of an event flow. It has not yet been recognized as a motif. Thus $M \subseteq S$ does not imply that M is a repetition motif, carry motif, completion motif, classifier motif, or any other semantic motif.

Definition 9217.3 (Generated motif recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Generate`

A generated motif recognizer *is an event flow*

$$R_M \in \mathsf{EventFlow}.$$

It is not an external pattern matcher, regular expression, host-level algorithm, or YAML-defined motif table. It is itself generated from the two raw marks and supported by recognizer-certificate structure.

Definition 9217.4 (Motif recognition relation). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Recognize`

The motif recognition relation is written

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu), \qquad R_M, S, M, \mu \in \mathsf{EventFlow}.$$

It is read as: the generated motif recognizer flow R_M recognizes subflow M of S as motif role μ. Explanatory roles include finite repetition, continuation, seal, carry, classifier quotient, ledger compression, reuse, and bridge. Formally, μ is not a host enum; it is role-flow structure represented in event-flow form.

Remark 9217.5 (Motif recognition is not regex matching). *The judgment* $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$ *is not a search for a bit substring or host pattern. It takes place on decoded source event flow and is witnessed by generated recognizer flow R_M. Its certificate discipline must retain domain, ledger, failure branches, and stability conditions.*

Definition 9217.6 (Motif occurrence). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifOccurrence`

If

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu),$$

then M is a motif occurrence *in S, written*

$$M \preceq_{\mu, R_M} S.$$

This means that, under recognizer R_M, the subflow M occurs in S as motif role μ.

Principle 9217.7 (No external motif input). *Lean4 ✓* `BEDC.GroundCompiler.SemanticMotif.no external motif input`

There is no formal input motif table, pattern list, or regular-expression rule that defines motifs. Motifs are recognized from event flows by generated motif recognizers.

Definition 9217.8 (Motif ledger). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifLedger`

If $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$*, a* motif ledger *is an event subflow L_M recording the raw events of M, its positions in S, why it is recognized as μ, whether normalization or classifier quotienting is involved, whether source material is hidden, and why adjacent candidate subflows fail. It is written*

$$\mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

The formal core of the relation requires L_M to be an event flow attached to the recognition witness and carried as a subflow of the same source flow.

Principle 9217.9 (No motif without ledger). *Lean4✓ BEDC.GroundCompiler.SemanticMotif.no motif without ledger*

Every motif recognition carries a motif ledger. The support flow M supplies the minimal ledger witness, and richer analyses may recognize a larger ledger subflow:

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu) \Longrightarrow \exists L_M,\ \mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

Theorem 9217.10 (Motif recognition is source-level). *Lean4✓ BEDC.GroundCompiler.SemanticMotif.motif recognition source level*

If $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$*, then S, M, and μ are source-level event-flow objects, not channel-level bit substrings.*

Proof. By Definition 9217.4, the arguments of RecognizesMotif are event flows. Event flows are source-level objects. By the source-channel separation theorem Definition 9209.27, channel substrings cannot be used directly as source events or source subflows. Hence motif recognition is a source-level relation. □

Corollary 9217.11 (Motif analysis must decode first). *Lean4✓ BEDC.GroundCompiler.SemanticMotif.motif analysis decodes first*

For a legal channel stream $c \in \mathsf{LegalZStream}$, motif analysis must first form

$$\mathsf{Decode}(c) = S$$

and then recognize motifs in S. It cannot search channel bits directly.

9217.2 Basic motif forms

Definition 9217.12 (Finite repetition motif). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.FiniteRepet*

A finite repetition motif *is a recognized event subflow of the explanatory form*

$$M = (x, xx, xxx, \ldots, x^k),$$

where x is a raw event and each term is recognized as a source-level extension of the previous term. For example, $0 \mid 00 \mid 000$ *can be a finite repetition candidate, but it becomes a motif only through a judgment*

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu_{\mathrm{rep}}).$$

Definition 9217.13 (Continuation motif). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.ContinuationM*

A continuation motif *is a recognized event subflow expressing continuation, composition, append, or combination behavior. Explanatory shapes include*

$$x, \quad y, \quad xy$$

or a larger input-flow, continuation-result-flow, and continuation-witness-flow. A candidate such as $00 \mid 000 \mid 00000$ *becomes a continuation motif only when the recognizer supplies the continuation witness and ledger.*

Definition 9217.14 (Seal motif). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.SealMotif*

A seal motif *is a recognized event subflow expressing finite stages, a repeat-boundary or completion witness, and tail ledger. A candidate such as*

$$0 \mid 00 \mid 000 \mid 01 \mid 011$$

may be recognized as completion-like, but the local event 011 *is not by itself a completed object. Completion requires seal, certificate, and ledger structure.*

Definition 9217.15 (Carry motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.CarryMotif`

A carry motif *is a recognized subflow*

$$w \mid v$$

where w is a pre-normal event, v is a normal event, and a source-level classifier or normalization relation relates w to v. The candidate $011 \mid 100$ *can be a Zeckendorf carry motif. It is not equality; it must retain a ledger for both raw pre-normal and normal events.*

Definition 9217.16 (Classifier quotient motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Classifie`

A classifier quotient motif *is a recognized event structure in which multiple raw event flows are placed in one class by a classifier:*

$$S_1 \sim S_2.$$

It contains raw-flow ledgers for S_1 and S_2, classifier witness, exactness witness, and failure cases.

Definition 9217.17 (Ledger compression motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.LedgerCo`

A ledger compression motif *is a recognized structure in which a visible event or package token hides source material while a ledger subflow preserves provenance. It contains visible token flow, hidden source material flow, ledger flow, pullback or source-map flow, and compression seal flow. This is the motif-level form of the package and gap ledger discipline.*

Definition 9217.18 (Reuse motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.ReuseMotif`

A reuse motif *is a recognized structure in which a new flow references accepted or recognized flows $N_1, \ldots, N_k$ and uses a reuse witness while still supplying its own certificate obligations. It contains input accepted-flow references, transport witness, new source-flow, classifier-flow, ledger-flow, certificate-flow, and strength bound. Reuse does not erase the target object's own certificate obligations.*

Definition 9217.19 (Bridge motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.BridgeMotif`

A bridge motif *is a recognized structure establishing conservative correspondence between a BEDC flow and a standard-side mathematical description. It contains BEDC source flow, standard-side target description flow, translation flow, soundness flow, reflection or completeness flow where available, no-host-leak ledger, and bridge seal. The external object is not a primitive input; it appears only through certified correspondence flow.*

9217.3 Code behavior

Proposition 9217.20 (Motif recognition does not change channel code). *Lean4 ✓* `BEDC.GroundCompiler.Semantic` `recognition preserves code`

If $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$, *then the channel code of S is still*

$$\mathsf{Compile}(S).$$

Proof. Motif recognition is a source-level relation over an existing event flow S. It does not change the events of S. The channel code is defined by $\mathsf{Compile}(S)$ and depends only on S. Therefore motif recognition does not change channel code. □

Corollary 9217.21 (Motif code is not a separate code). *Lean4 ✓* `BEDC.GroundCompiler.SemanticMotif.motif` `code not separate`

The code of a motif occurrence M is the ordinary channel compilation $\mathsf{Compile}(M)$ *of its raw subflow. It is not produced by a separate motif compiler.*

Theorem 9217.22 (Motif recognition is invariant under compile-decode). *Lean4 ✓ BEDC.GroundCompiler.Semanti compile decode invariant*

If $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$*, then*

$$\mathsf{RecognizesMotif}(R_M, \mathsf{Decode}(\mathsf{Compile}(S)), M, \mu).$$

Proof. By the channel round-trip theorem Definition 9208.17,

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S.$$

Substitution into the recognition judgment gives the claim. □

9217.4 Profiles, overlap, and prefixes

Definition 9217.23 (Motif profile). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.MotifProfile*

Given an event flow S *and a family of motif recognizers* $\mathcal{R}$*, define the* motif profile *of* S *by*

$$\mathsf{Profile}_{\mathcal{R}}(S) = \{\, (\mu, M, L_M) \mid \exists R_M \in \mathcal{R},\ \mathsf{RecognizesMotif}(R_M, S, M, \mu) \wedge \mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M) \,\}.$$

Thus the profile records recognized motif occurrences together with their ledgers.

Remark 9217.24 (Motif profile depends on recognizer family). *The profile* $\mathsf{Profile}_{\mathcal{R}}(S)$ *depends on* $\mathcal{R}$*. If* $\mathcal{R}$ *contains no carry recognizer, carry motifs will not be reported; if it contains no seal recognizer, completion-like motifs will not be reported. A motif profile is analysis relative to a generated and certified recognizer family, not every truth about a bare event flow.*

Definition 9217.25 (Motif support). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.MotifSupport*

If

$$(\mu, M, L_M) \in \mathsf{Profile}_{\mathcal{R}}(S),$$

then M *is the* support subflow *of the motif occurrence and* L_M *is its* support ledger*.*

Definition 9217.26 (Motif overlap). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.MotifOverlap*

For event flows S *and* T*, their motif overlap is*

$$\mathsf{Overlap}_{\mathcal{R}}(S, T) = \mathsf{Profile}_{\mathcal{R}}(S) \cap \mathsf{Profile}_{\mathcal{R}}(T),$$

where the comparison is by motif role, support shape, and ledger-compatible equivalence, not by bare channel substring.

Proposition 9217.27 (Channel substring overlap is insufficient). *Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.c substring overlap insufficient*

If two channel codes share a bit substring, that alone does not imply that their source flows share a motif.

Proof. Channel substrings are not source events. For example, a channel code may contain a substring shaped like 011 even when its decoded event is the raw event 1. Therefore channel substring overlap cannot serve as source motif overlap. Motif overlap must proceed by decoding and recognition. □

Definition 9217.28 (Event-level common prefix). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.EventCo`

For event flows

$$S = (s_0, \ldots, s_{m-1}), \qquad T = (t_0, \ldots, t_{n-1}),$$

an event-level common prefix of length k *is the assertion that* $k \le m$, $k \le n$, *and*

$$s_i = t_i \quad \text{for all } 0 \le i < k.$$

It is written $\mathsf{PrefixLen}(S, T, k)$.

Definition 9217.29 (Recognized semantic prefix). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Recogni`

A recognized semantic prefix *of* S *and* T *is a common event-level prefix* P *such that a generated motif recognizer recognizes* P *inside both flows under the same motif role. Raw prefix equality alone is not semantic prefix equality, because the semantic datum includes the recognized role.*

Proposition 9217.30 (Same raw prefix need not imply same semantic prefix). *Lean4 ✓* `BEDC.GroundCompiler.Sem` `raw prefix not unique semantic prefix`

Two event flows can share the same raw prefix while that prefix is recognized under different motif roles. Thus raw prefix equality does not determine a unique semantic prefix role.

Proof. The one-event prefix 0 is a common event-level prefix of the two displayed one-event flows 0 and 0. A generated motif recognizer may recognize that prefix under the finite-repetition role in one occurrence and under the continuation role in another occurrence. These roles are distinct event-flow roles, so the shared raw prefix does not determine a unique semantic prefix role. □

9217.5 Motif metrics

Definition 9217.31 (Seal depth). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.SealDepth`

For a recognizer family $\mathcal{R}$, *the* seal depth $\mathsf{SealDepth}_{\mathcal{R}}(S)$ *is the number of ledger-distinct motif occurrences in* S *recognized as seal motifs.*

Definition 9217.32 (Carry index). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.CarryIndex`

The carry index $\mathsf{CarryIndex}_{\mathcal{R}}(S)$ *is the number of ledger-distinct motif occurrences in* S *recognized as carry motifs. A recognized occurrence* $011 \mid 100$ *contributes at least one carry occurrence.*

Definition 9217.33 (Ledger depth). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.LedgerDepth`

The ledger depth $\mathsf{LedgerDepth}_{\mathcal{R}}(S)$ *is the maximum recognized dependency nesting depth of motif ledgers, package ledgers,* $\mathsf{NameCert}$ *ledgers, derivation-certificate ledgers, and theorem ledgers inside* S.

Definition 9217.34 (Classifier compression ratio). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Classi`

If $\mathcal{R}$ *recognizes a classifier quotient motif in* S *placing raw flows* $M_1, \ldots, M_k$ *into* r *recognized classes with* $r > 0$, *define*

$$\mathsf{CompressionRatio}_{\mathcal{R}}(S) = \frac{k}{r}.$$

This metric is defined only when classifier recognition and exactness ledger are both present.

Definition 9217.35 (Reuse depth). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.ReuseDepth`

The reuse depth $\mathsf{ReuseDepth}_{\mathcal{R}}(S)$ *is the length of the longest dependency chain of reuse motifs in* S. *It measures how many layers of accepted or recognized objects the flow depends on.*

Definition 9217.36 (Motif distance). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifDistance`

For event flows S and T, a motif distance $\mathsf{Dist}_{\mathcal{R}}(S,T)$ *is any distance function based on their motif profiles. A simple Jaccard-style choice is*

$$\mathsf{Dist}_{\mathcal{R}}(S,T) = 1 - \frac{|\mathsf{Profile}_{\mathcal{R}}(S) \cap \mathsf{Profile}_{\mathcal{R}}(T)|}{|\mathsf{Profile}_{\mathcal{R}}(S) \cup \mathsf{Profile}_{\mathcal{R}}(T)|}.$$

Applications may choose distances better suited to ledger, role, and support-shape comparisons.

Theorem 9217.37 (Motif metrics are code-invariant under round trip). *Lean4 ✓* `BEDC.GroundCompiler.SemanticM` `metrics roundtrip invariant`

If $c = \mathsf{Compile}(S)$, then for every recognizer family $\mathcal{R}$,

$$\mathsf{Profile}_{\mathcal{R}}(S) = \mathsf{Profile}_{\mathcal{R}}(\mathsf{Decode}(c)).$$

Consequently seal depth, carry index, ledger depth, compression ratio, reuse depth, and motif-distance values defined from profiles are invariant under compile-decode round trip.

Proof. By channel round trip, $\mathsf{Decode}(\mathsf{Compile}(S)) = S$. Hence S and $\mathsf{Decode}(c)$ are the same source event flow. Motif recognizers act on source event flows, so their profiles agree. Metrics defined from those profiles agree as well. □

Corollary 9217.38 (Motif analysis can be performed on decoded code). *Lean4 ✓* `BEDC.GroundCompiler.SemanticM` `analysis on decoded code`

Given $c \in \mathsf{LegalZStream}$, one can decode $S = \mathsf{Decode}(c)$ and compute $\mathsf{Profile}_{\mathcal{R}}(S)$. This is the same source-level analysis as applying the recognizers to the original event flow.

9217.6 Examples

Example 9217.39 (Completion skeleton motif profile). *The event flow*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

may be recognized as containing finite repetition $0 \mid 00 \mid 000$*, a repeat-boundary motif* 01*, a seal motif* 011*, a carry motif* $011 \mid 100$*, and the larger completion skeleton. Its channel code is*

$$011001100011010110101011100011.$$

Motif recognition must decode back to source event flow before identifying 011 *or* 100 *as source-level events.*

Example 9217.40 (Add and fold skeleton motif profile). *The event flow*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100$$

may be recognized as unit source 0*, binary repetition* 00*, additive recursion* 001*, iterated-addition raw completion* 0011*, fold normal address* 0100*, and carry or normalization ledger* $0011 \mid 0100$*. Compared with the completion skeleton, it shares the finite-source prefix* $0 \mid 00$ *but then branches toward binary combination and fold rather than seal-style completion.*

Example 9217.41 ($\mathsf{List}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ motif relation)**.** *A* $\mathsf{List}^{\uparrow}$ *flow can contain a finite spine motif for nil, cons, cons-cons, and so on. Under bottom readback this overlaps with the finite repetition and finite-length motifs of* $\mathsf{Nat}^{\uparrow}$*. The list flow also carries payload classifier, constructor-shape ledger, length ledger, and source-carrier reuse. Thus* $\mathsf{List}^{\uparrow}$ *is not merely* $\mathsf{Nat}^{\uparrow}$*; it is finite repetition plus payload and constructor ledger structure.*

Example 9217.42 ($\mathsf{Add}^{\uparrow}$ and categorical composition)**.** *An* $\mathsf{Add}^{\uparrow}$ *event flow may recognize binary continuation, unit behavior, and associativity behavior. A category-composition flow may recognize arrow composition, identity behavior, and associativity behavior. Their motif profiles can overlap in continuation, unit, and associativity ledgers, while category structure also carries object and arrow sorting, domain and codomain compatibility, and identity-arrow ledger.*

Example 9217.43 ($\mathsf{Real}^{\uparrow}$ and completion)**.** *Completion flows and* $\mathsf{Real}^{\uparrow}$ *flows can both contain finite approximation stages, compatible thread, tail classifier, seal motif, and ledger retaining finite stages. The* $\mathsf{Real}^{\uparrow}$ *flow also carries arithmetic descent, order or metric compatibility, field-operation ledger, and Cauchy or regularity classifier. Thus* $\mathsf{Real}^{\uparrow}$ *is not a bare completion motif; it is completion plus algebra, order, and metric certificate stack.*

Example 9217.44 ($\mathsf{DimLift}^{\uparrow}$ and circle or phase structure)**.** *A* $\mathsf{DimLift}^{\uparrow}$ *flow may contain completion witness, overflow witness, carry motif, lifted normal address, and old-axis ledger. Circle or phase structure may contain periodic closure, phase carry, lifted coordinate, unit-constraint ledger, and topological classifier. Shared carry shapes such as* $011 \mid 100$ *or higher carry motifs can reveal common carry and lift structure without implying that the mature objects are the same.*

9217.7 Limits of motif analysis

Theorem 9217.45 (Motif similarity does not imply object equality)**.** *Lean4✓ BEDC.GroundCompiler.SemanticMoti similarity not object equality*

If two event flows S *and* T *have nonempty motif overlap,*

$$\mathsf{Overlap}_{\mathcal{R}}(S,T) \neq \emptyset,$$

one cannot conclude that they are the same source flow, represent the same accepted object, or are equivalent under any classifier.

Proof. Motif overlap says only that the event flows contain recognized motif occurrences with compatible roles and ledgers. Two distinct source flows can share such an occurrence. Object equality or accepted-object equivalence further requires the appropriate classifier, bridge, exactness, and acceptance-gate recognition. Motif overlap does not supply those judgments. □

Corollary 9217.46 (Motif analysis is explanatory, not proof of identity)**.** *Motif analysis can explain generating similarity between theories, but it does not replace proof, classifier exactness,* NameCert*, derivation-certificate, or acceptance-gate structure.*

Theorem 9217.47 (Motif recognition is conservative over the finite kernel)**.** *Lean4✓ BEDC.GroundCompiler.Seman generated conservativity*

Adding RecognizesMotif*,* Profile*,* SealDepth*,* CarryIndex*,* LedgerDepth*, and* ReuseDepth *does not add new raw marks, history constructors, sameness rules, continuation rules, or kernel-level equalities.*

Proof. All these relations act on event flows or recognized subflows. They record, recognize, and compare source-level event structures already present. They do not generate new b0 or b1, add msame, hsame, or psame rules, change Ext or Cont, or license derived interfaces. Derived interfaces still require NameCert flow; accepted objects still require acceptance-gate flow. Hence motif analysis is conservative over the finite kernel. □

Corollary 9217.48 (Motif analysis cannot license mature objects). Lean4 ✓ `BEDC.GroundCompiler.SemanticMotif.analysis cannot license mature objects`

A flow showing completion, carry, phase, or quotient motifs is not thereby licensed as $\mathsf{Real}^{\uparrow}$*, circle structure,* $\mathsf{Complex}^{\uparrow}$*, topology, or any other mature object. Mature objects still require recognized* NameCert*, derivation-certificate, and acceptance-gate flows.*

Definition 9217.49 (Motif report). Lean4 (definition) `BEDC.GroundCompiler.SemanticMotif.MotifReport`

A motif report *is a human-readable output view generated from an event flow S and recognizer family $\mathcal{R}$. It may list recognized motifs, support subflows, motif ledgers, metrics, overlaps with other flows, and failure cases. It is not formal input, proof, or certificate.*

Principle 9217.50 (Motif reports are output, not input). Lean4 ✓ `BEDC.GroundCompiler.SemanticMotif.motif reports output not input`

Motif reports, similarity tables, distance matrices, diagrams, and YAML or JSON reports cannot serve as formal motif-analysis input or formal compiler input. They are outputs of event-flow analysis.

Remark 9217.51 (Outlook). *This chapter establishes that motifs are not external patterns, but stable subflows recognized by generated motif recognizers. The core judgment is* $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$*. Motif recognition is source-level, requires decoding before analysis, preserves channel code, and is invariant under compile-decode round trip. Channel substring overlap does not imply motif overlap, and raw prefix equality does not imply semantic prefix equality. Motif profiles support seal depth, carry index, ledger depth, compression ratio, reuse depth, and motif distance, all relative to a recognizer family. Motif similarity explains generative relation but does not prove object equality or license mature objects. Motif analysis is conservative over the finite kernel, and motif reports are outputs rather than inputs. The next chapter applies this framework to concrete flow comparisons among* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*, completion, and Zeckendorf axis skeletons.*

Chapter 9218

First Case Studies: Finite Repetition, Add/Fold, List, and Completion

chapter 9217 establishes the no-hidden-input form of semantic motif analysis: motifs are not external patterns, but stable subflows recognized by generated motif recognizers. This chapter begins the first concrete comparisons. The goal is not to prove that any mature object has already been accepted. The goal is to show how already-generated event flows can be compared at the underlying $\mathsf{b0}, \mathsf{b1}$ layer.

The case studies are finite repetition skeletons, Nat-like skeletons, Add and fold skeletons, List-like skeletons, and Completion or $\mathsf{FullAxis}^{\uparrow}$-like skeletons. The guiding rule is

$$\boxed{\text{different traditional theories can share event motifs without becoming the same object.}}$$

For example, $\mathsf{Nat}^{\uparrow}$ and $\mathsf{List}^{\uparrow}$ can share finite-spine motifs; $\mathsf{Add}^{\uparrow}$ and fold or composition can share continuation motifs; completion and $\mathsf{FullAxis}^{\uparrow}$ can share seal and tail-ledger motifs; Zeckendorf carry and $\mathsf{DimLift}^{\uparrow}$ can share carry and normal-address motifs. These are motif-level relations. Mature export still requires recognized $\mathsf{NameCert}$ flow, derivation-certificate flow, ledger flow, and acceptance-gate flow.

9218.1 Case-study flows

Definition 9218.1 (Case-study flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.CaseStudyFlow`

A case-study flow *is an event flow*

$$S \in \mathsf{EventFlow}$$

equipped with an explanatory target such as finite repetition skeleton, Add and fold skeleton, Completion skeleton, or List skeleton. The explanatory target is not input structure. Formally the flow is still just

$$S = (w_0, \ldots, w_{n-1}),$$

and its theoretical reading must be recognized by generated recognizer flow.

Definition 9218.2 (Skeleton code)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.SkeletonCode`

For a case-study flow $S = (w_0, \ldots, w_{n-1})$, define its skeleton code *by*

$$\mathsf{SkCode}(S) := \mathsf{Compile}(S).$$

This is not object code and not accepted theorem code. It is the channel code of the event flow.

Remark 9218.3 (Skeletons are weaker than certificates)**.** *A skeleton flow can display a motif such as finite repetition, carry, or seal structure, but it is not thereby a* NameCert *flow, derivation-certificate flow, or accepted-object flow. For example,*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

can be a Completion or $\mathsf{FullAxis}^{\uparrow}$ *skeleton, but a full* $\mathsf{FullAxis}^{\uparrow}$ *certificate still requires recognized source, pattern, classifier, stability, and ledger fields.*

9218.2 Finite repetition skeleton

Definition 9218.4 (Finite repetition skeleton)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.FiniteRepet`

Let

$$S_{\mathrm{rep}}^{(k)} = (0, 00, 000, \ldots, 0^k)$$

be the event flow consisting of the first k finite b0*-repetition events. For example,*

$$S_{\mathrm{rep}}^{(3)} = (0, 00, 000).$$

Its explanatory reading is starting unit, one repetition, two repetitions, and so on. Formally it is only an event flow.

Example 9218.5 (Finite repetition skeleton code)**.** *For*

$$0 \mid 00 \mid 000,$$

the event encodings are

$$0 \mapsto 011, \qquad 00 \mapsto 0011, \qquad 000 \mapsto 00011.$$

Hence

$$\mathsf{Compile}(0 \mid 00 \mid 000) = 011001100011,$$

displayed by event as

$$011 \mid 0011 \mid 00011.$$

Definition 9218.6 (Finite repetition motif recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.Fin`

Given a motif recognizer R_{rep}, if

$$\mathsf{RecognizesMotif}(R_{\mathrm{rep}}, S, M, \mu_{\mathrm{rep}})$$

and M has event shape

$$(x, xx, xxx, \ldots, x^k),$$

then M is recognized in S as a finite repetition motif. When $x = 0$, this is the finite b0*-axis repetition motif.*

Proposition 9218.7 (Finite repetition skeleton contains repetition motif)**.** *Lean4 ✓* `BEDC.GroundCompiler.CaseSt` `skeleton has motif`

If R_{rep} is certified as a finite-repetition recognizer and

$$S_{\mathrm{rep}}^{(k)} = (0, 00, \ldots, 0^k),$$

then

$$\mathsf{RecognizesMotif}(R_{\mathrm{rep}}, S_{\mathrm{rep}}^{(k)}, S_{\mathrm{rep}}^{(k)}, \mu_{\mathrm{rep}})$$

holds.

Proof. By definition, each successor event in $S_{\mathrm{rep}}^{(k)}$ is obtained from the preceding event by source-level appending of 0:

$$0^i \mapsto 0^{i+1}.$$

This is exactly the source-level pattern certified for the finite-repetition recognizer. Hence R_{rep} recognizes the skeleton as a finite repetition motif. □

Corollary 9218.8 (Finite repetition motif does not imply $\mathsf{Nat}^{\uparrow}$)**.** *Lean4✓ BEDC.GroundCompiler.CaseStudies.repe skeleton not repeat rule*

The finite repetition skeleton

$$0 \mid 00 \mid 000$$

does not by itself contain the Nat-like repeat-rule witness

$$01.$$

Thus finite repetition motif data alone cannot be read as $\mathsf{Nat}^{\uparrow}$*. A derived interface such as* $\mathsf{Nat}^{\uparrow}$ *still requires recognized* NameCert *flow with source, pattern, classifier, stability, and ledger fields.*

9218.3 Nat-like skeleton

Definition 9218.9 (Nat-like skeleton)**.** *Lean4 (definition) BEDC.GroundCompiler.CaseStudies.NatLikeSkeleton*

A Nat-like skeleton *is an event flow*

$$S_{\mathrm{NatSkel}} = (0, 00, 000, \ldots, 0^k, 01),$$

whose initial segment is a finite repetition skeleton and whose final event 01 *is a candidate repeat-rule witness. Formally it remains just an event flow.*

Example 9218.10 (Nat-like skeleton code)**.** *For*

$$0 \mid 00 \mid 000 \mid 01,$$

the event encodings are

$$0 \mapsto 011, \quad 00 \mapsto 0011, \quad 000 \mapsto 00011, \quad 01 \mapsto 01011.$$

Thus the skeleton code is

$$01100110001101011.$$

Definition 9218.11 (Repeat-rule motif)**.** *Lean4 (definition) BEDC.GroundCompiler.CaseStudies.RepeatRuleMotif*

A repeat-rule motif *is a recognized structure consisting of finite repetition source plus repeat-rule witness. An explanatory shape is*

$$0 \mid 00 \mid 000 \mid 01,$$

where $0 \mid 00 \mid 000$ *is the finite repetition source and* 01 *is the repeat-rule witness candidate.*

Proposition 9218.12 (Nat-like skeleton extends finite repetition skeleton)**.** *Lean4✓ BEDC.GroundCompiler.CaseSt like extends repetition*

The Nat-like skeleton S_{NatSkel} *contains* $S_{\mathrm{rep}}^{(k)}$ *as a prefix subflow.*

Proof. By definition,

$$S_{\mathrm{NatSkel}} = (0, 00, \ldots, 0^k, 01).$$

Its first k repetition events are exactly

$$S_{\mathrm{rep}}^{(k)} = (0, 00, \ldots, 0^k).$$

□

Corollary 9218.13 (Nat-like and repetition skeletons share a prefix motif)**.** Lean4 ✓ BEDC.GroundCompiler.CaseSt repetition share prefix

If the finite repetition recognizer is certified, then the Nat-like skeleton and the finite repetition skeleton share the finite repetition motif.

Remark 9218.14 (Existing $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$)**.** *In existing BEDC development,* $\mathsf{Nat}^{\uparrow}$ *can arise from unary repetition direction, while the Zeckendorf-axis reading can separately consider the* $\mathsf{b0}$ *direction as* $\mathsf{AxisNat}^{\uparrow}$*. This chapter does not identify them as the same object. It only records that, when both use finite repetition as source, their motif profiles share finite repetition or spine structure.*

9218.4 Add and fold skeletons

Definition 9218.15 (Add skeleton)**.** Lean4 (definition) BEDC.GroundCompiler.CaseStudies.AddSkeleton

The Add-like skeleton is

$$S_{\mathrm{AddSkel}} = (0, 00, 001).$$

The explanatory reading is starting unit, binary repetition or binary combination witness, and binary recursion witness. The event 001 *is not bare addition; it is an Add-style local witness candidate.*

Example 9218.16 (Add skeleton code)**.** *The event encodings are*

$$0 \mapsto 011, \qquad 00 \mapsto 0011, \qquad 001 \mapsto 001011.$$

Therefore

$$\mathsf{SkCode}(S_{\mathrm{AddSkel}}) = 0110011001011.$$

Definition 9218.17 (Fold or MulSeed skeleton)**.** Lean4 (definition) BEDC.GroundCompiler.CaseStudies.FoldSkelet

The fold-like or MulSeed-like skeleton is

$$S_{\mathrm{FoldSkel}} = (0, 00, 001, 0011, 0100).$$

The explanatory reading is starting unit, binary repetition, Add-style recursion witness, iterated-addition completion witness, and normal-address candidate.

Example 9218.18 (Fold or MulSeed skeleton code)**.** *The event encodings are*

$$0 \mapsto 011, \qquad 00 \mapsto 0011, \qquad 001 \mapsto 001011,$$
$$0011 \mapsto 00101011, \qquad 0100 \mapsto 0100011.$$

Thus

$$\mathsf{SkCode}(S_{\mathrm{FoldSkel}}) = 011001100101100101011010 0011.$$

Definition 9218.19 (Additive recursion motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.AdditiveRec`

An additive recursion motif *is a recognized event structure containing unit source, binary repetition witness, and binary recursion witness. An explanatory shape is*

$$0 \mid 00 \mid 001.$$

Definition 9218.20 (Fold completion motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.FoldCompletio`

A fold completion motif *is a recognized event structure containing additive recursion witness, raw iterated-addition completion witness, and normal fold address. An explanatory shape is*

$$001 \mid 0011 \mid 0100,$$

where $0011 \mid 0100$ *may be recognized as source-level carry or normalization ledger.*

Proposition 9218.21 (Fold skeleton extends Add skeleton). *Lean4 ✓* `BEDC.GroundCompiler.CaseStudies.fold skeleton extends add`

The fold skeleton S_{FoldSkel} *contains* S_{AddSkel} *as a prefix subflow.*

Proof. By definition,

$$S_{\mathrm{FoldSkel}} = (0, 00, 001, 0011, 0100).$$

Its first three events are $(0, 00, 001) = S_{\mathrm{AddSkel}}$. □

Corollary 9218.22 (Fold skeleton reuses Add skeleton). *Lean4 ✓* `BEDC.GroundCompiler.CaseStudies.fold reuses add`

At the motif-profile layer, the fold or MulSeed skeleton reuses the Add skeleton's additive recursion motif and adds fold completion or normalization motifs.

Theorem 9218.23 (Add and fold skeletons do not license $\mathsf{Add}^{\uparrow}$ or $\mathsf{Mul}^{\uparrow}$). *Lean4 ✓* `BEDC.GroundCompiler.CaseStud fold not license`

There exist Add-like and fold-like case-study readings whose flows are

$$S_{\mathrm{AddSkel}} \quad \textit{and} \quad S_{\mathrm{FoldSkel}},$$

with S_{AddSkel} *a prefix subflow of* S_{FoldSkel}*, while their case-study targets are distinct. Thus recognizing additive or fold motifs does not by itself license* $\mathsf{Add}^{\uparrow}$ *or* $\mathsf{Mul}^{\uparrow}$*.*

Proof. Add and fold skeletons are source-level motifs. The formal witness is the pair of case-study flows tagged by distinct Add-like and fold-like target readings, together with the prefix inclusion of S_{AddSkel} in S_{FoldSkel}. Derived interfaces such as $\mathsf{Add}^{\uparrow}$ and $\mathsf{Mul}^{\uparrow}$ require recognized $\mathsf{NameCert}$ flows; mature export further requires derivation-certificate and acceptance-gate flows. Motif recognition supplies none of these certificates. □

9218.5 List-like skeleton

Definition 9218.24 (List spine skeleton). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.ListSpineSkeleto`

Given a source event a*, define the list-spine skeleton*

$$S_{\mathrm{ListSkel}}(a, k) = (a, aa, aaa, \ldots, a^k).$$

Its explanatory reading is one-element spine, two-element spine, three-element spine, and so on. Formally this is a finite repetition motif over a*. If* $a = 0$*, then* $0 \mid 00 \mid 000$ *is also the minimal finite repetition skeleton.*

Definition 9218.25 (List constructor ledger motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.ListCon`

A list constructor ledger motif *is ledger structure over a finite spine motif recording nil or cons shape, length, payload source, payload classifier, and constructor decomposition. This motif is not provided by bare finite repetition; it must be recognized by a list recognizer flow.*

Proposition 9218.26 (List spine skeleton shares finite repetition with Nat-like skeleton). *Lean4✓* `BEDC.GroundCompiler.CaseStudies.list spine shares repetition with nat`

If $a = 0$, then

$$S_{\mathrm{ListSkel}}(0, k) = S^{(k)}_{\mathrm{rep}}.$$

Hence a List-like finite spine skeleton and a Nat-like finite repetition skeleton have the same raw finite repetition motif.

Proof. By definition,

$$S_{\mathrm{ListSkel}}(0, k) = (0, 00, \ldots, 0^k),$$

which is exactly $S^{(k)}_{\mathrm{rep}}$. □

Corollary 9218.27 ($\mathsf{List}^{\uparrow}$ is finite repetition plus constructor ledger). *Lean4✓* `BEDC.GroundCompiler.CaseStudies finite plus ledger`

If a list constructor ledger motif is recognized, then it contains a recognized reuse-spine motif and a nonempty ledger flow. When $\mathsf{List}^{\uparrow}$ *uses finite spine as source, its motif profile may share finite repetition structure with Nat-like skeletons, but it still needs constructor ledger, payload classifier, length ledger, and source-carrier certificate. Thus it is not* $\mathsf{Nat}^{\uparrow}$*; it is finite-spine motif plus list-specific ledger and classifier structure.*

Theorem 9218.28 (Same finite spine does not imply same object). *Lean4✓* `BEDC.GroundCompiler.CaseStudies.s finite spine not same object`

There exist two case-study readings whose event flows share the same finite spine motif, and even the same underlying finite-spine flow, while their case-study targets are distinct. Hence finite-spine sharing alone does not determine object identity.

Proof. The same finite spine motif can serve as Nat-like source, List-like spine, finite index carrier, matrix index carrier, or polynomial coefficient list. The formal witness is a pair of case-study flows with identical finite-spine event flow and distinct target tags. Object identity is determined by classifier, ledger, $\mathsf{NameCert}$, and derivation-certificate structure. Sharing finite spine records source-level motif overlap only. □

9218.6 Completion and FullAxis skeleton

Definition 9218.29 (Completion skeleton). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.CompletionSkele`

The Completion or $\mathsf{FullAxis}^{\uparrow}$*-like skeleton is*

$$S_{\mathrm{CompSkel}} = (0, 00, 000, 01, 011, 100).$$

The explanatory reading is starting unit, one repetition, two repetitions, repeat-boundary witness, repeat-to-completion or overflow witness, and normal-address candidate.

Example 9218.30 (Completion skeleton code). *The event encodings are*

$$0 \mapsto 011, \qquad 00 \mapsto 0011, \qquad 000 \mapsto 00011, \qquad 01 \mapsto 01011,$$
$$011 \mapsto 0101011, \qquad 100 \mapsto 100011.$$

Therefore

$$\mathsf{SkCode}(S_{\mathrm{CompSkel}}) = 011001100011010110101011100011.$$

Definition 9218.31 (Completion motif)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.CompletionMotif`

A completion motif *is a recognized structure containing finite stages, repeat-boundary witness, seal or completion witness, normal-address witness, and tail ledger. The explanatory shape is*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Proposition 9218.32 (Completion skeleton contains finite repetition motif)**.** *Lean4✓* `BEDC.GroundCompiler.Case` `skeleton contains repetition`

The completion skeleton contains $(0, 00, 000)$ *as a prefix subflow, and hence contains a finite repetition motif candidate.*

Proof. By definition,

$$S_{\mathrm{CompSkel}} = (0, 00, 000, 01, 011, 100).$$

Its first three events are $(0, 00, 000)$. □

Proposition 9218.33 (Completion skeleton contains carry ledger candidate)**.** *Lean4✓* `BEDC.GroundCompiler.Case` `skeleton contains carry`

The completion skeleton contains

$$011 \mid 100$$

as a source-level carry ledger candidate.

Proof. By definition, the last two events of S_{CompSkel} are $(011, 100)$. Hence the carry ledger candidate appears as a subflow. □

Corollary 9218.34 (Completion skeleton splits from Add and fold skeleton)**.** *Lean4✓* `BEDC.GroundCompiler.CaseS` `completion split`

The Add and fold skeleton

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100$$

and the Completion skeleton

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

share the prefix $0 \mid 00$ *and then branch. The former turns toward additive recursion and fold; the latter turns toward repeat-boundary, seal, and carry.*

Theorem 9218.35 (Completion skeleton does not license $\mathsf{FullAxis}^{\uparrow}$)**.** *Even if* S_{CompSkel} *is recognized as a completion motif, one cannot infer* $\mathsf{FullAxis}^{\uparrow}$*.*

Proof. A completion motif supplies source-level event structure. The derived interface $\mathsf{FullAxis}^{\uparrow}$ requires recognized $\mathsf{NameCert}$ flow: finite-prefix source, compatible-thread pattern, prefix-compatible classifier, seal stability, and tail ledger. Motif recognition is not $\mathsf{NameCert}$ recognition. □

9218.7 Cross-theory comparison

Definition 9218.36 (Skeleton overlap). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.SkeletonOverlap`

For skeleton flows S and T, their raw prefix overlap is $\mathsf{PrefixLen}(S, T)$, *and their recognized motif overlap is*

$$\mathsf{Overlap}_{\mathcal{R}}(S, T),$$

the set of motif occurrences shared under recognizer family $\mathcal{R}$.

Proposition 9218.37 (Add and fold skeleton and Completion skeleton share finite source). *Lean4✓* `BEDC.GroundCompiler.CaseStudies.fold completion prefix`

Let

$$S_{\text{FoldSkel}} = (0, 00, 001, 0011, 0100)$$

and

$$S_{\text{CompSkel}} = (0, 00, 000, 01, 011, 100).$$

Then

$$\mathsf{PrefixLen}(S_{\text{FoldSkel}}, S_{\text{CompSkel}}) = 2.$$

Proof. The first two events in both flows are $0 \mid 00$. The third events are 001 and 000, respectively, so the longest common prefix has length 2. □

Corollary 9218.38 (Shared prefix indicates shared generative source). *Lean4✓* `BEDC.GroundCompiler.CaseStudie completion share starting prefix`

At the raw event-flow layer, the Add and fold skeleton and the Completion skeleton share the starting unit and one finite repetition. Their paths branch at the third event.

Proposition 9218.39 (Nat-like skeleton is a prefix of Completion skeleton). *Lean4✓* `BEDC.GroundCompiler.CaseS prefix completion`

Let

$$S_{\text{NatSkel}} = (0, 00, 000, 01)$$

and

$$S_{\text{CompSkel}} = (0, 00, 000, 01, 011, 100).$$

Then S_{NatSkel} is a prefix subflow of S_{CompSkel}.

Proof. The first four events of S_{CompSkel} are

$$(0, 00, 000, 01) = S_{\text{NatSkel}}.$$

□

Corollary 9218.40 (Completion skeleton extends Nat-like skeleton). *Lean4✓* `BEDC.GroundCompiler.CaseStudies extends nat like`

The Completion skeleton extends the Nat-like finite repetition and repeat-rule skeleton by adding

$$011 \mid 100,$$

the completion witness and normal-address candidate.

Theorem 9218.41 (Skeleton extension does not imply object extension). *Lean4✓ BEDC.GroundCompiler.CaseStud extension not object extension*

There exist two case-study skeleton readings whose event flows are related by prefix inclusion while their case-study targets are distinct. Hence raw skeleton extension alone does not determine inclusion, subobject, subtype, extension, or morphism between corresponding mature objects.

Proof. The Nat-like skeleton is a prefix of the Completion skeleton, but their target readings are distinct. The prefix relation is a raw event-flow relation. Mature-object relations require classifier, NameCert, derivation-certificate, bridge, or morphism certificates. Raw prefix structure supplies none of these. □

9218.8 Motif reports

Definition 9218.42 (Skeleton motif report). *Lean4 (definition) BEDC.GroundCompiler.CaseStudies.SkeletonMotifR*

For a skeleton flow S and recognizer family $\mathcal{R}$, define

$$\mathsf{SkReport}_{\mathcal{R}}(S)$$

to be the human-readable output view of its source flow, motif profile flow, support subflows, ledger subflows, and basic metrics. It is not formal input and not proof.

Example 9218.43 (Completion skeleton motif report). *For*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

a report may list finite repetition motif $0 \mid 00 \mid 000$*, repeat-rule motif* 01*, seal motif* 011*, carry motif* $011 \mid 100$*, raw prefix length of six events, seal depth one, carry index one, and a ledger warning that* $\mathsf{FullAxis}^{\uparrow}$ *is not licensed without* NameCert *flow.*

Example 9218.44 (Add and fold skeleton motif report). *For*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100,$$

a report may list finite source $0 \mid 00$*, additive recursion motif* 001*, fold completion motif* $0011 \mid 0100$*, seal-depth candidate one, carry-index candidate one, and a ledger warning that* $\mathsf{Add}^{\uparrow}$ *or* $\mathsf{Mul}^{\uparrow}$ *is not licensed without* NameCert *and derivation-certificate flows.*

Example 9218.45 (Nat-like versus List-like motif report). *If the Nat-like skeleton is* $0 \mid 00 \mid 000 \mid 01$ *and the List-like skeleton is* $0 \mid 00 \mid 000$*, a report may show shared finite repetition motif* $0 \mid 00 \mid 000$*, the Nat-like extra event* 01 *as repeat-rule witness, missing List-specific payload classifier, constructor-shape ledger, and length ledger, and the warning that the same finite spine does not imply the same object.*

Theorem 9218.46 (Skeleton motif reports are explanatory, not certificatory). *Lean4✓ BEDC.GroundCompiler.Cas reports not certificates*

There exists a skeleton motif report on a readable event-flow source whose reported source flow is not an accepted object flow. Hence the existence of a skeleton motif report alone does not license derived interfaces, theoremhood, accepted objects, or object equality.

Proof. Take the empty source-flow report. Its source flow is readable as an event flow, but it is not an accepted object flow because acceptance requires a nonempty accepted export together with recognized certificate structure. Motif reports are human-readable outputs generated from recognized motif profiles. They do not supply NameCert flow, derivation-certificate flow, proof flow, or acceptance-gate flow by themselves. They are explanatory tools, not certificates. □

Corollary 9218.47 (Skeleton comparison cannot replace proof)**.** `Lean4✓ BEDC.GroundCompiler.CaseStudies.comp not proof`

There exist two skeleton flows with shared raw prefix length 2 *that are not equal as event flows. Thus raw skeleton comparison alone cannot replace a proof of theorem equivalence, object equivalence, bridge, morphism, or certified reuse relation.*

9218.9 Conservativity

Theorem 9218.48 (Case-study analysis is conservative over the finite kernel)**.** `Lean4✓ BEDC.GroundCompiler.Case studies conservativity`

The concrete Add, fold, and Completion skeletons in this chapter use only display marks from the finite alphabet $\{\mathsf{b0}, \mathsf{b1}\}$*. Adding these case-study skeletons, motif reports, and cross-skeleton comparisons therefore does not add new raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, or accepted-object export rules.*

Proof. All case-study skeletons are event flows over b0 and b1. Motif reports and comparisons are source-level analyses of event flows. They do not generate new b0 or b1, add msame, hsame, or psame rules, change Ext or Cont, license derived interfaces, or export accepted objects. Hence this analysis layer is conservative over the finite kernel. □

Remark 9218.49 (Outlook)**.** *This chapter gives the first concrete skeleton comparisons: finite repetition* 0 | 00 | 000*, Nat-like* 0 | 00 | 000 | 01*, Add* 0 | 00 | 001*, fold or MulSeed* 0 | 00 | 001 | 0011 | 0100*, and Completion or* $\mathsf{FullAxis}^{\uparrow}$ *skeleton* 0 | 00 | 000 | 01 | 011 | 100*. Their typical channel codes are* 011001100011*,* 01100110001101011*,* 0110011001011*,* 011001100101100101011010 0011*, and* 01100110001101011010101110001 1*. The analysis shows shared finite repetition prefixes, Add-to-fold extension, Nat-like-to-completion extension, and the split between fold and completion paths. It also records the limits: same finite spine does not imply same object, skeleton extension does not imply mature-object extension, motif reports are explanatory rather than certificates, and the case-study layer is conservative over the finite kernel. The next chapter defines theory-distance invariants such as prefix profile, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, and normal-address distance.*

Chapter 9219

Theory Metrics and Flow Invariants

chapter 9218 compares finite repetition, Nat-like, Add, fold, and Completion skeletons. It shows that skeletons can share low-level motifs without implying object equality, theorem equivalence, or mature export. This chapter defines a group of theory metrics for comparing event flows by their recognized structures.

The metrics include prefix profile, motif profile, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, normal-address distance, and theory-flow distance. These are not proof, NameCert, derivation certificate, or acceptance gate. They are statistics and comparisons over decoded source event flows, recognized motifs, recognized ledgers, recognized classifiers, normal-address records, and reuse structures.

The governing rule is

metrics are computed only from recognized event-flow structure, never from channel substrings or external label

9219.1 Admissible metric data

Definition 9219.1 (Metric recognizer family)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.MetricRecogn`

A metric recognizer family *is a finite family of generated recognizer flows*

$$\mathcal{R} = (R_1, \ldots, R_k),$$

where each R_i is certified for some structure used by the metric, such as finite repetition, continuation, seal, carry, ledger, classifier, reuse, bridge, theorem-flow, or chapter-flow recognition. The family $\mathcal{R}$ is not an external metric table. It is represented by event-flow structure and supported by recognizer certificates.

Definition 9219.2 (Metric admissibility)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.MetricAdmissible`

A metric specification records a finite list of data sources. A metric M is admissible *for a recognizer family $\mathcal{R}$ when every recorded source for $M(S)$ is one of:*

1. *the decoded source event flow S;*
2. *motif occurrences recognized by $\mathcal{R}$;*
3. *ledger flows recognized by $\mathcal{R}$;*
4. *classifier flows recognized by $\mathcal{R}$;*
5. *reuse flows recognized by $\mathcal{R}$;*

6. *normal-address flows recognized by $\mathcal{R}$;*

7. *the associated failure-flow and cannot-claim-flow data.*

It may not depend on channel-substring search, external object names, YAML fields, host ASTs, theorem labels, human chapter titles, external semantic categories, or unrecorded classifier judgments.

Principle 9219.3 (No external metric input)**.** Lean4 ✓ `BEDC.GroundCompiler.MetricsFlow.no external metric input`

Every theory metric must be event-flow-derived. A metric depending on external object names, host tags, YAML fields, manual theorem grouping, or unrecognized semantic labels is not a BEDC-admissible metric.

9219.2 Prefix profiles

Definition 9219.4 (Raw prefix profile)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.RawPrefixProfile`

For an event flow

$$S = (w_0, \ldots, w_{n-1}),$$

define its raw prefix profile *by*

$$\mathsf{RawPrefix}(S) = \{\, (w_0, \ldots, w_i) \mid 0 \le i < n \,\}.$$

For the empty event flow, set $\mathsf{RawPrefix}(S) = \{\epsilon\}$*. The raw prefix profile records source-level event prefixes but no semantic role.*

Definition 9219.5 (Recognized prefix profile)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.RecognizedPr`

Given a recognizer family $\mathcal{R}$, define $\mathsf{Prefix}_{\mathcal{R}}(S)$ *to be the set of all triples (P, μ, L) such that P is a raw prefix of S, P is recognized by some $R \in \mathcal{R}$ as motif role μ, and L is its motif ledger:*

$$(P, \mu, L) \in \mathsf{Prefix}_{\mathcal{R}}(S)$$

iff there exists $R \in \mathcal{R}$ with

$$\mathsf{RecognizesMotif}(R, S, P, \mu) \quad \textit{and} \quad \mathsf{MotifLedger}_R(S, P, \mu, L).$$

Remark 9219.6 (Raw prefix and semantic prefix differ)**.** *Two event flows can share a raw prefix without sharing a recognized prefix profile. For instance,* 01 *may be recognized in one flow as a repeat-boundary witness and in another as a branch separator. Different recognized roles or ledgers mean different semantic prefixes.*

Definition 9219.7 (Prefix-overlap length)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.RawPrefixOverla`

For event flows S and T, define $\mathsf{RawPrefixLen}(S, T)$ *as the largest k such that*

$$S[0:k] = T[0:k].$$

Define recognized prefix overlap by

$$\mathsf{PrefixOverlap}_{\mathcal{R}}(S, T) = \mathsf{Prefix}_{\mathcal{R}}(S) \cap \mathsf{Prefix}_{\mathcal{R}}(T),$$

where intersection is computed by recognized role, support subflow shape, and ledger-compatible equivalence, not by bare event strings alone.

Proposition 9219.8 (Raw prefix overlap is weaker than recognized prefix overlap). *Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.raw prefix weaker*

If $\mathsf{PrefixOverlap}_{\mathcal{R}}(S,T) \neq \emptyset$*, then* S *and* T *share a raw prefix support. Conversely, shared raw prefix support need not imply recognized prefix overlap.*

Proof. A recognized prefix overlap includes a recognized motif support, so its support is a raw prefix of both flows. The converse fails because a shared raw prefix may be recognized under different roles, have incompatible ledgers, or fail to be recognized in one of the two flows. □

9219.3 Motif profiles and counts

Definition 9219.9 (Motif profile). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.MotifProfile*

For a recognizer family $\mathcal{R}$*, define the motif profile of an event flow* S *as a finite list of recognized occurrence records satisfying*

$$(\mu, M, L, R) \in \mathsf{MotifProfile}_{\mathcal{R}}(S) \Longrightarrow R \in \mathcal{R},\ M \subseteq S,\ \mathsf{RecognizesMotif}(R, S, M, \mu) \wedge \mathsf{MotifLedger}_R(S, M, \mu, L).$$

Here μ *is the motif role,* M *the support subflow,* L *the motif ledger, and* R *the generated recognizer supplying the occurrence.*

Definition 9219.10 (Motif multiplicity). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.MotifMultiplicity*

For a finite motif profile list P*, the multiplicity of motif role* μ *is*

$$\mathsf{Mult}(P, \mu) = \#\{\, (\mu', M, L, R) \in P \mid \mu' = \mu \,\}.$$

This chapter considers finite chapter, theorem, and package event flows.

Definition 9219.11 (Seal depth). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.SealDepth*

The seal depth *of* S *is*

$$\mathsf{SealDepth}_{\mathcal{R}}(S) = \mathsf{Mult}_{\mathcal{R}}(S, \mu_{\text{seal}}),$$

where μ_{seal} *is the recognized event-flow role for seal motifs.*

Definition 9219.12 (Carry index). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.CarryIndex*

The carry index *of* S *is*

$$\mathsf{CarryIndex}_{\mathcal{R}}(S) = \mathsf{Mult}_{\mathcal{R}}(S, \mu_{\text{carry}}),$$

where μ_{carry} *is the recognized carry motif role. A recognized occurrence* $011 \mid 100$ *contributes at least one carry occurrence.*

Definition 9219.13 (Ledger depth). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.LedgerDepth*

The ledger depth $\mathsf{LedgerDepth}_{\mathcal{R}}(S)$ *is the maximum dependency nesting depth among recognized ledger flows in* S*. A ledger that references no other ledger has depth* 1*. If a ledger* L *references ledgers* $L_1, \ldots, L_k$*, then*

$$\operatorname{depth}(L) = 1 + \max_i \operatorname{depth}(L_i).$$

If no ledger flow is recognized in S*, the ledger depth is* 0*.*

Definition 9219.14 (Classifier compression ratio). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.Compress`

If $\mathcal{R}$ recognizes a classifier quotient in S that sends k raw support flows $M_1, \ldots, M_k$ into r recognized classifier classes with $r > 0$, define

$$\mathsf{CompressionRatio}_{\mathcal{R}}(S) = \frac{k}{r}.$$

This metric is defined only when classifier recognition, ledger recognition, and exactness recognition are all supplied.

Remark 9219.15 (Compression ratio is not equality strength). *The compression ratio measures how strongly raw flows are grouped by a classifier. It does not express kernel equality or theorem equivalence. If several raw flows are compressed into one classifier class, the ledger must retain the raw supports.*

Definition 9219.16 (Reuse depth). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.ReuseDepth`

The reuse depth $\mathsf{ReuseDepth}_{\mathcal{R}}(S)$ *is the length of the longest dependency chain of recognized reuse motifs in S. If no reuse motif is recognized, the reuse depth is* 0.

Definition 9219.17 (Bridge depth). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.BridgeDepth`

If S contains recognized bridge motifs, its bridge depth $\mathsf{BridgeDepth}_{\mathcal{R}}(S)$ *is the maximum bridge-chain length from BEDC source flow to standard-side target flow. If no bridge motif is recognized, the bridge depth is* 0.

9219.4 Normal addresses and distances

Definition 9219.18 (Normal-address map). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.NormalAddressM`

Given a source-level normalizer recognizer in $\mathcal{R}$, if S contains a ledger-preserving normalization

$$w \rightsquigarrow v,$$

then the normal-address record is

$$(w, v, L_{w \to v}),$$

where $L_{w \to v}$ is the normalization ledger. The normal-address map $\mathsf{NormAddr}_{\mathcal{R}}(S)$ is the set of all such records.

Definition 9219.19 (Normal-address distance). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.NormalAddre`

For event flows S and T, define a normal-address distance by

$$\mathsf{NormDist}_{\mathcal{R}}(S, T) = 1 - \frac{|\mathsf{NormAddr}_{\mathcal{R}}(S) \cap \mathsf{NormAddr}_{\mathcal{R}}(T)|}{|\mathsf{NormAddr}_{\mathcal{R}}(S) \cup \mathsf{NormAddr}_{\mathcal{R}}(T)|}.$$

If the denominator is zero, the distance is either undefined or set to 0, *according to the chosen analysis protocol.*

Definition 9219.20 (Motif Jaccard distance). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MotifJaccardI`

The motif Jaccard distance is

$$\mathsf{MotifDist}_{\mathcal{R}}(S, T) = 1 - \frac{|\mathsf{MotifProfile}_{\mathcal{R}}(S) \cap \mathsf{MotifProfile}_{\mathcal{R}}(T)|}{|\mathsf{MotifProfile}_{\mathcal{R}}(S) \cup \mathsf{MotifProfile}_{\mathcal{R}}(T)|}.$$

Intersections and unions are computed by motif role, support shape, and ledger-compatible equivalence, not by raw strings.

Definition 9219.21 (Flow signature vector). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.FlowSignature`

Given $\mathcal{R}$, define the flow signature vector *of S by*

$$\mathsf{Sig}_{\mathcal{R}}(S) = (\mathsf{SealDepth}, \mathsf{CarryIndex}, \mathsf{LedgerDepth}, \mathsf{CompressionRatio}, \mathsf{ReuseDepth}, \mathsf{BridgeDepth})_S,$$

with all components computed relative to $\mathcal{R}$.

Remark 9219.22 (Signature vector is not object identity). *Equal signature vectors do not imply that two event flows represent the same theory, theorem, object, or classifier class. A signature vector is an analysis summary, not an identity certificate.*

Theorem 9219.23 (Metrics are channel-roundtrip invariant). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.metric` `channel roundtrip invariant`

If $c = \mathsf{Compile}(S)$, then for every admissible metric M,

$$M(S) = M(\mathsf{Decode}(c)).$$

Proof. By channel round trip, $\mathsf{Decode}(\mathsf{Compile}(S)) = S$, so $\mathsf{Decode}(c) = S$. An admissible metric depends only on decoded event flow and recognized structure. Therefore it has the same value on S and on $\mathsf{Decode}(c)$. □

Corollary 9219.24 (Metrics may be computed from code after decoding). *Lean4 ✓* `BEDC.GroundCompiler.Metrics` `computed from decoded code`

If $c \in \mathsf{LegalZStream}$, one may first compute $S = \mathsf{Decode}(c)$ and then compute any admissible metric $M(S)$. This agrees with computing the metric on the original event flow.

Proposition 9219.25 (Channel substring metrics are inadmissible). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.` `substring metrics inadmissible`

Any metric that directly counts channel substrings such as 011, 100, *or* 11 *is inadmissible unless it first decodes the channel stream and turns the result into source-level event-flow recognition.*

Proof. Definition 9209.27 proves that channel substrings are not source events. A metric that counts channel substrings can confuse event terminators, raw-event encodings, and source-level event shapes. Such a metric depends on the wrong layer and violates Definition 9219.2. □

9219.5 Theory-flow distance and reports

Definition 9219.26 (Theory-flow distance). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.TheoryFlowDist`

For recognized theory flows S and T, an optional theory-flow distance *is a weighted combination*

$$\begin{aligned}\mathsf{TheoryDist}_{\mathcal{R}}(S,T) = {} & \alpha_1 \mathsf{MotifDist}_{\mathcal{R}}(S,T) + \alpha_2 \mathsf{NormDist}_{\mathcal{R}}(S,T) \\ & + \alpha_3 d_{\text{prefix}}(S,T) + \alpha_4 d_{\text{ledger}}(S,T) + \alpha_5 d_{\text{reuse}}(S,T).\end{aligned}$$

The weights $\alpha_i \geq 0$ must be supplied by an analysis protocol flow, not hidden as external parameters.

Remark 9219.27 (Metric weights must be explicit). *If a weighted distance is used, the weights must be recorded in an analysis-protocol event flow. Otherwise the metric result contains hidden input.*

Definition 9219.28 (Analysis protocol flow). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.AnalysisProto`

An analysis protocol flow *is an event flow P recording the chosen recognizer family, chosen metrics, metric weights, comparison domain, failure cases, ledger requirements, normalization policy, and report format. Without such a protocol, no distance or similarity is canonical.*

Theorem 9219.29 (Theory-flow distance is protocol-relative). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.theor distance protocol relative

The distance

$$\mathsf{TheoryDist}_{\mathcal{R},P}(S,T)$$

depends on recognizer family $\mathcal{R}$ and analysis protocol flow P. There is no unconditional absolute theory distance unless canonical recognizers and a canonical analysis protocol are supplied.

Proof. Theory-flow distance uses recognizer family, metric selection, weights, normalization policy, and ledger policy. These are specified by $\mathcal{R}$ and P. Changing either can change which motifs, normal addresses, ledger depths, and weights are available. Hence the distance is protocol-relative. □

Corollary 9219.30 (Similarity claims require analysis protocol). *A claim that two theories are similar, closer, or share more structure must supply an analysis protocol flow P and recognizer family $\mathcal{R}$. Without them the claim is informal intuition, not a BEDC-admissible metric claim.*

Definition 9219.31 (Metric report). Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.MetricReport

A metric report *is an output view generated from an analysis protocol flow P, recognizer family $\mathcal{R}$, and one or more event flows. It records computed metrics, recognized motifs, normal-address records, ledger depth, classifier compression, reuse depth, bridge depth, warnings, and cannot-claim entries. It is not input, proof, or certificate.*

Principle 9219.32 (Metric reports are not evidence of theoremhood). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow reports not theoremhood

A metric report does not prove theoremhood, object identity, mature export, bridge correctness, or classifier exactness. It records recognized analysis results only.

Theorem 9219.33 (Metrics do not imply object equality). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.metrics do not imply object equality

Even if

$$\mathsf{Sig}_{\mathcal{R}}(S) = \mathsf{Sig}_{\mathcal{R}}(T)$$

or

$$\mathsf{TheoryDist}_{\mathcal{R},P}(S,T) = 0,$$

one cannot conclude that S and T represent the same accepted object, theorem, chapter, or classified object unless an additional classifier, exactness, or bridge certificate is supplied. In particular, the empty admissible analysis assigns the same zero signature to distinct event flows, so equality of metric signatures does not even determine equality of the underlying event-flow datum.

Proof. Metrics are summaries or distances over recognized motifs and ledgers. Object equality and theorem equivalence require classifier or proof-level exactness. Equal metrics can occur for distinct flows with the same structural counts. Even distance zero relative to a protocol means only indistinguishability by that protocol, not equality in all recognized structures. Equality or equivalence therefore requires a separate certificate. □

Corollary 9219.34 (Metrics are explanatory, not identificatory). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.met explanatory not identificatory

Metric analysis can explain structural similarity, but it cannot identify objects without additional classifier or bridge flow.

9219.6 Examples and cannot-claims

Example 9219.35 (Metrics for Completion skeleton)**.** *Let*

$$S_{\mathrm{Comp}} = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

If the recognizer family identifies finite repetition $0 \mid 00 \mid 000$*, seal* 011*, and carry* $011 \mid 100$*, then the report may record*

$$\mathsf{SealDepth}(S_{\mathrm{Comp}}) = 1, \qquad \mathsf{CarryIndex}(S_{\mathrm{Comp}}) = 1, \qquad \mathsf{LedgerDepth}(S_{\mathrm{Comp}}) \geq 1.$$

If no recognized ledger flow is present, the carry index should not be treated as admissible, because carry motifs require ledger support.

Example 9219.36 (Metrics for Add and fold skeleton)**.** *Let*

$$S_{\mathrm{Fold}} = 0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

If additive recursion 001 *and fold completion* $0011 \mid 0100$ *are recognized, then a report may record a seal-depth candidate, carry-index candidate, and reuse depth* 1 *if* 001 *is recognized as reused by* 0011*. Whether* $\mathsf{Add}^{\uparrow}$ *or* $\mathsf{Mul}^{\uparrow}$ *is licensed still depends on* $\mathsf{NameCert}$ *and derivation-certificate flows.*

Example 9219.37 (Prefix comparison: Fold versus Completion)**.** *For*

$$S_{\mathrm{Fold}} = (0, 00, 001, 0011, 0100), \qquad S_{\mathrm{Comp}} = (0, 00, 000, 01, 011, 100),$$

the raw prefix overlap is $0 \mid 00$*, so*

$$\mathsf{RawPrefixLen}(S_{\mathrm{Fold}}, S_{\mathrm{Comp}}) = 2.$$

The recognized interpretation is that both share finite-source prefix, then the fold skeleton branches into additive recursion while the completion skeleton branches into repeat-boundary, seal, and carry. This is motif-level explanation, not a theorem of object relation.

Example 9219.38 (Nat-like versus Completion)**.** *For*

$$S_{\mathrm{Nat}} = (0, 00, 000, 01), \qquad S_{\mathrm{Comp}} = (0, 00, 000, 01, 011, 100),$$

S_{Nat} *is a raw prefix of* S_{Comp}*. A metric report may say that the Completion skeleton extends the Nat-like skeleton by* $011 \mid 100$*, but it may not say that the Completion object extends the Nat object without a bridge or object-level certificate.*

Definition 9219.39 (Metric cannot-claim entries)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricCa`

A metric report must include cannot-claim entries for claims not supported by the metrics. Examples include:

- *same motif profile implies same theorem;*
- *same signature vector implies same object;*
- *shared prefix implies subobject;*
- *positive carry index implies dimension exists;*
- *positive seal depth implies real completion exists;*

- *compression ratio greater than one implies quotient exactness.*

These entries prevent metric analysis from being misread as proof.

Proposition 9219.40 (Cannot-claim entries are required for nontrivial reports)**.** Lean4✓ BEDC.GroundCompiler.M
claims required for nontrivial reports

A BEDC metric report satisfying the cannot-claim guard and involving seal, carry, classifier, bridge, or reuse motifs includes cannot-claim entries specifying what is not established by the report.

Proof. Seal, carry, classifier, bridge, and reuse motifs are easily confused with mature-object claims such as completion, equality, quotient exactness, standard bridge, or object construction. Metrics do not supply NameCert, derivation certificate, proof, or exactness by themselves. A report without cannot-claim entries can therefore overstate its results. Nontrivial reports must record these limits. □

9219.7 Conservativity

Theorem 9219.41 (Metric analysis is conservative over the finite kernel)**.** Lean4✓ BEDC.GroundCompiler.MetricsI
conservativity

Adding admissible metrics, metric reports, flow distances, signature vectors, and motif profiles does not add new raw marks, history constructors, sameness rules, continuation rules, or derived-interface formation rules.

Proof. All metrics are computed from decoded event flows and recognized motifs. They do not generate new b0 or b1, add msame, hsame, or psame rules, change Ext or Cont, or license derived interfaces. They summarize recognized structures only. Therefore metric analysis is conservative over the finite kernel. □

Corollary 9219.42 (Metrics cannot replace certificates or gates)**.** Lean4✓ BEDC.GroundCompiler.MetricsFlow.met
cannot replace certificates

No metric, distance, profile, or report can replace recognized NameCert *flow, derivation-certificate flow, proof flow, or acceptance-gate flow.*

Remark 9219.43 (Outlook)**.** *This chapter establishes the no-hidden-input version of theory metrics: metrics are computed only from decoded event flows and generated recognizers. It defines metric recognizer families, admissibility, raw and recognized prefix profiles, prefix overlap, motif profiles, motif multiplicity, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, bridge depth, normal-address maps and distances, motif Jaccard distance, flow signature vectors, theory-flow distance, analysis protocol flows, metric reports, and metric cannot-claim entries. It proves that recognized prefix overlap is stronger than raw prefix overlap, metrics are invariant under compile-decode round trip, channel substring metrics are inadmissible, theory-flow distance is protocol-relative, similarity claims require protocol data, metrics do not imply object equality, metric reports are explanatory rather than proof, nontrivial reports require cannot-claim entries, metric analysis is conservative over the finite kernel, and metrics cannot replace certificates or acceptance gates. The next chapter applies these indicators to higher-level objects such as* $\mathsf{Real}^{\uparrow}$*, topology, circle and phase structure,* $\mathsf{Complex}^{\uparrow}$*, functors, and homology.*

Chapter 9220

Higher-Layer Case Studies: Completion, Real, Topology, Geometry, Category, and Homology

chapter 9219 defines theory metrics on event flows: prefix profile, motif profile, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, bridge depth, normal-address distance, and theory-flow distance. Those metrics are explanatory analysis tools, not proof, NameCert, derivation certificate, or acceptance gate.

This chapter applies those indicators to higher-layer object routes: completion, real-number structure, metric structure, topology, circle or phase structure, complex-number structure, functors, and homology. The purpose is not to prove that these mature objects have been accepted. It is to show how traditionally different theories can share low-level event motifs such as finite approximation, compatible thread, seal, tail ledger, classifier quotient, reuse, carry or lift, bridge, composition, and boundary-square-zero.

The governing rule is

high-layer objects are not event codes; they are recognized structures after certificates, classifiers, ledgers, and ga

All case studies in this chapter say only that certain high-layer theories exhibit certain event-flow motifs. They do not imply mature export, object equality, completed bridge, or high-level theoremhood.

9220.1 Higher-layer case-study flows

Definition 9220.1 (Higher-layer case-study flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.HigherLay`

A higher-layer case-study flow *is an event flow*

$$S \in \mathsf{EventFlow}$$

whose motif profile, under some recognizer family $\mathcal{R}$, contains at least one high-layer motif such as completion, metric, topology, phase, circle, complex, category, functor, homology, bridge, or analysis. The high-layer reading is not an external label; it is determined by motif roles and ledger structures recognized in S.

Definition 9220.2 (Higher-layer motif bundle)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.HighMotifBu`

For an event flow S and recognizer family $\mathcal{R}$, define

$$\mathsf{HighMotif}_{\mathcal{R}}(S)$$

as the subcollection of $\mathsf{MotifProfile}_{\mathcal{R}}(S)$ *whose records are recognized as higher-layer roles: completion, metric, topology, phase, circle, complex, category, functor, homology, bridge, or analysis. These role names are explanatory; formally they are recognized motif role flows.*

Remark 9220.3 (Higher-layer motif bundles are not mature export)**.** *The bundle* $\mathsf{HighMotif}_{\mathcal{R}}(S)$ *is a collection of recognized motifs. It is not* $\mathrm{AcceptGate}(N, s)$ *and cannot automatically promote* S *to* $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, *circle structure,* $\mathsf{Complex}^{\uparrow}$, *or homology.*

9220.2 Completion and Real-like flows

Definition 9220.4 (Completion skeleton flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.CompletionSk`

A completion skeleton flow *contains recognized finite-stage flow, compatible-thread flow, tail-classifier flow, seal witness flow, tail-ledger flow, and normal-address flow. An explanatory shape is*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

where the initial segment records finite stages, 01 *is a thread witness,* 011 *is a seal or completion witness, and* 100 *is a normal address. This shape does not by itself constitute* $\mathsf{Completion}^{\uparrow}$.

Definition 9220.5 (Completion motif record)**.** *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.CompletionMo`

A completion motif record *is a tuple*

$$(M_{\text{stage}}, M_{\text{thread}}, M_{\text{classifier}}, M_{\text{seal}}, M_{\text{ledger}})$$

where the entries are finite-stage support flow, compatible-thread support flow, tail or equivalence classifier flow, seal witness flow, and tail ledger flow. The tuple must be recognized by a generated completion recognizer.

Proposition 9220.6 (Completion skeleton is seal-heavy)**.** *Lean4 ✓* `BEDC.GroundCompiler.CaseStudies.completion skeleton seal heavy`

If an event flow S contains a recognized completion motif record

$$(M_{\text{stage}}, M_{\text{thread}}, M_{\text{classifier}}, M_{\text{seal}}, M_{\text{ledger}}),$$

then M_{seal} is a nonempty recognized seal subflow of S. In the metric language of chapter 9219, this witnesses

$$\mathsf{SealDepth}_{\mathcal{R}}(S) \geq 1.$$

Proof. A completion motif record contains a seal witness flow M_{seal}. If that witness is recognized as a seal motif by $\mathcal{R}$, then the seal depth counts at least that occurrence. □

Proposition 9220.7 (Completion skeleton is ledger-dependent)**.** *Lean4 ✓* `BEDC.GroundCompiler.CaseStudies.comp skeleton ledger dependent`

If a candidate completion motif record in S has no nonempty tail ledger flow M_{ledger}, then it is not admissible as a recognized completion motif record.

Proof. Completion hides or compresses an infinite-looking compatible thread into a named or sealed interface. Without a ledger, the finite stages and their provenance cannot be recovered. BEDC ledger discipline requires hidden material to remain traceable. Hence a completion motif without tail ledger is inadmissible. □

Corollary 9220.8 (Seal event alone is not completion). Lean4 ✓ `BEDC.GroundCompiler.CaseStudies.seal event alone not completion`

A single event such as 011*, even when recognized as a seal witness, does not constitute completion. It must appear with finite-stage source, compatible-thread pattern, tail classifier, seal stability, and tail ledger.*

Definition 9220.9 (Real-like event flow). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.RealLikeEventFlou`

A Real-like event flow *is a completion skeleton enriched by rational or dyadic approximation source motif, Cauchy or regularity classifier motif, operation descent motif, order or metric compatibility motif, completion seal motif, and real-ledger motif. Thus it is finite approximation plus classifier, completion, operation descent, and order or metric ledger.*

Definition 9220.10 (Real motif bundle). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.RealMotifBundle`

Define $\mathsf{RealMotif}_{\mathcal{R}}(S)$ *as the collection of recognized approximation-source, Cauchy-classifier, completion-seal, operation-descent, order-compatibility, metric-compatibility, and real-ledger records in* S*. If these records are not all present,* S *is only a partial Real-like flow, not a complete* $\mathsf{Real}^{\uparrow}$ *certificate flow.*

Proposition 9220.11 (Real-like flow extends completion flow). Lean4 ✓ `BEDC.GroundCompiler.CaseStudies.real like extends completion`

If S *is a complete Real-like flow, then* S *contains a recognized completion motif.*

Proof. By Definition 9220.9, a Real-like flow contains completion seal and completion classifier structure. These are components of a recognized completion motif. □

Corollary 9220.12 (Completion is necessary but not sufficient for $\mathsf{Real}^{\uparrow}$). Lean4 ✓ `BEDC.GroundCompiler.CaseStu necessary not sufficient real`

Completion motif is a necessary part of Real-like structure, but it is not sufficient. $\mathsf{Real}^{\uparrow}$ *also requires approximation source, Cauchy or regularity classifier, operation descent, order and metric compatibility, and ledger.*

Theorem 9220.13 (Real-like motif profile does not export $\mathsf{Real}^{\uparrow}$). *If the motif profile of* S *contains Real-like motifs, one still cannot infer that* $\mathsf{Real}^{\uparrow}$ *has been exported.*

Proof. Motif profile records structural motifs only. Exporting $\mathsf{Real}^{\uparrow}$ requires recognized NameCert and derivation-certificate flows, including exactness, classifier, ledger, stability, and strength. The motif profile does not supply those certificates. □

9220.3 Metric and topology flows

Definition 9220.14 (Metric motif). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.MetricMotif`

A metric motif *is a recognized event structure containing carrier flow, distance-value flow, nonnegativity witness flow, symmetry witness flow, triangle witness flow, zero-exactness classifier flow, and metric ledger flow. It is only a source-level component of a possible* $\mathsf{Met}^{\uparrow}$ *certificate.*

Definition 9220.15 (Topology motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.TopologyMotif`

A topology motif *is a recognized event structure containing open or neighborhood source flow, finite-intersection stability flow, arbitrary-union or generated-union schema flow, continuity classifier flow, local ledger flow, and topology seal flow. If it contains only finite intersections and lacks union, basis, or generated-topology ledger, it is a partial topology motif.*

Definition 9220.16 (Neighborhood-basis motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.Neighborho`

A neighborhood-basis motif *is a weaker topology motif containing point flow, neighborhood candidate flow, refinement witness flow, intersection witness flow, and local ledger flow. It can serve as source for a topology construction, but it does not by itself give a full topology.*

Proposition 9220.17 (Metric motif generates topology-like motif only with certificate). *Lean4* ✓ `BEDC.GroundCompiler.CaseStudies.metric to topology needs cert`

A metric motif in S does not by itself imply a topology motif. An additional recognized flow is needed to prove that metric balls or neighborhoods generate a topology basis and to carry the basis ledger.

Proof. Metric motif supplies distance structure. Topology motif requires open or neighborhood closure behavior, finite intersection, union or generated-basis schema, and continuity classifier. These are not supplied by distance structure alone. They require a recognized construction certificate from metric balls to topology basis. □

Corollary 9220.18 (Metric does not automatically imply topology at raw motif layer). *Lean4* ✓ `BEDC.GroundCompiler.CaseStudies.metric not automatic topology`

Metric-like and topology-like event structures are related but not identical at the raw motif layer. The passage from one to the other requires bridge or certificate flow.

Definition 9220.19 (Topological classifier compression ratio). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudie`

If a topology flow uses an open-set or neighborhood classifier to group raw local flows into equivalent local neighborhoods, define

$$\mathsf{TopoCompression}_{\mathcal{R}}(S) = \mathsf{CompressionRatio}_{\mathcal{R}}(S)$$

restricted to topology classifier motifs.

Theorem 9220.20 (Topology motif does not prove continuity theorems). *Lean4* ✓ `BEDC.GroundCompiler.CaseStud` `motif not continuity`

There exists an event flow S that contains the topology source, finite-intersection, generated-union, continuity-classifier, basis-ledger, and topology-seal subflows of a topology motif, but contains no continuity proof-flow. Hence a topology motif alone does not prove any continuity theorem.

Proof. Take the canonical topology motif skeleton consisting exactly of the six topology-support subflows. It has the topology motif components, but no event subflow carrying a continuity proof. Continuity requires a separate relation between source and target topology flows and a proof that inverse image or neighborhood transport preserves that structure. □

9220.4 Phase, circle, and complex flows

Definition 9220.21 (Phase motif). *Lean4 (definition)* `BEDC.GroundCompiler.CaseStudies.PhaseMotif`

A phase motif *is a recognized event structure containing cyclic or periodic source flow, completion or seal witness flow, carry or wrap-around witness flow, normal-address flow, and phase ledger*

flow. Explanatory carry shapes include 011 | 100 *and* 10011 | 10100*, but these are source-level carry candidates, not equalities.*

Definition 9220.22 (Circle motif)**.** Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.CircleMotif`

A circle motif *is phase motif plus unit-constraint or closed-loop classifier structure. It contains phase source flow, periodic closure flow, unit constraint flow, equation classifier flow, topological compatibility flow, and circle ledger flow. It does not automatically export circle structure.*

Proposition 9220.23 (Circle motif extends phase motif)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudies.circle extends phase`

If S contains a recognized circle motif, then S contains a recognized phase motif.

Proof. By Definition 9220.22, circle motif contains phase source flow, periodic closure flow, and ledger structure whose base part is precisely phase motif structure. □

Corollary 9220.24 (Phase motif is necessary but not sufficient for circle)**.** Lean4✓ `BEDC.GroundCompiler.CaseStu necessary not sufficient circle`

Phase motif is part of circle-like construction, but circle structure still requires unit constraint, equation classifier, topological compatibility, and ledger.

Theorem 9220.25 (Carry motif does not imply geometric dimension)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudie motif not dimension`

There exists a carry-only event flow, for example the source-level carry candidate 011 | 100*, that contains a carry motif and contains no dimension-certificate subflow. Therefore a carry motif does not imply geometric dimension, topological dimension, linear dimension, or manifold structure.*

Proof. The flow 011 | 100 records a carry candidate and no additional certificate subflow. Dimension requires carrier, topology, chart, vector-space, manifold, or homological certificates. Carry motif alone supplies none of these. □

Definition 9220.26 (Complex motif)**.** Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.ComplexMotif`

A complex motif *is a recognized event structure containing ordered-pair or product source flow, real-component source flow, addition and multiplication descent flow, rotation or phase motif, conjugation or norm motif, field or ring classifier flow, and complex ledger flow. It typically reuses Real-like, product-like, ring-like, and phase-like flows, so its reuse depth and ledger depth are high.*

Definition 9220.27 (Complex analytic motif)**.** Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.ComplexAnaly`

A complex analytic motif *is complex motif plus limit or derivative source flow, difference quotient flow, Cauchy-Riemann or analytic compatibility flow, local power-series flow, modulus ledger flow, and analytic seal flow. It is not automatic from the complex motif.*

Proposition 9220.28 (Complex motif inherits reuse depth from real and product components)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudies.complex reuse depth`

If a complex motif in S is recognized as reusing Real-like and product-like flows, then

$$\mathsf{ReuseDepth}_{\mathcal{R}}(S)$$

is at least the maximum of the reuse depths of those components plus one.

Proof. The complex motif has reuse edges from the Real-like and product-like components into the complex construction. This adds one reuse layer above the maximum reuse depth of the reused components. □

Theorem 9220.29 (Complex motif does not imply complex analysis). Lean4 ✓ BEDC.GroundCompiler.CaseStudies not analysis

There exists a complex-motif-only event flow whose complex support is present but whose analysis-support flow is absent. Hence a complex motif alone does not imply complex differentiability, analyticity, Cauchy integral theorem, or power-series expansion.

Proof. Complex motif supplies algebraic or coordinate support over paired components. Complex analysis requires limit, derivative, analytic compatibility, local expansion, and proof-flow support. The exhibited motif-only flow contains the former support and omits the latter support. □

9220.5 Category, functor, and natural transformation motifs

Definition 9220.30 (Category composition motif). Lean4 (definition) BEDC.GroundCompiler.CaseStudies.Category

A category composition motif *is an event structure containing object-source flow, morphism-source flow, domain and codomain classifier flow, identity witness flow, composition witness flow, associativity proof-flow, unit proof-flow, and category ledger flow. It generalizes continuation or composition motif by adding object and morphism sorting and composability conditions.*

Definition 9220.31 (Functor motif). Lean4 (definition) BEDC.GroundCompiler.CaseStudies.FunctorMotif

A functor motif *is an event structure containing source-category flow, target-category flow, object-map flow, morphism-map flow, identity-preservation flow, composition-preservation flow, and functor ledger flow. It reuses two category motifs.*

Definition 9220.32 (Natural transformation motif). Lean4 (definition) BEDC.GroundCompiler.CaseStudies.Natural

A natural transformation motif *is an event structure containing source functor flow, target functor flow, component morphism family flow, naturality-square proof-flow, and component ledger flow. It reuses functor motifs and category composition motifs.*

Proposition 9220.33 (Functor motif extends category motif). Lean4 ✓ BEDC.GroundCompiler.CaseStudies.funct extends category

If S contains a recognized functor motif, then S contains or references recognized category motifs for source and target categories.

Proof. By Definition 9220.31, functor motif contains source-category and target-category flows. These must be recognized category motifs or references to recognized category flows. □

Proposition 9220.34 (Natural transformation motif extends functor motif). Lean4 ✓ BEDC.GroundCompiler.Case transformation extends functor

If S contains a recognized natural transformation motif, then S contains or references source and target functor motifs.

Proof. By Definition 9220.32, natural transformation motif contains source functor flow and target functor flow. □

Theorem 9220.35 (Composition motif alone does not yield category). Lean4 ✓ BEDC.GroundCompiler.CaseStudie not category

There exists a composition-motif-only event flow whose composition support is present but whose category-sorting support is absent. Hence a continuation or composition motif does not by itself yield a category.

Proof. Composition motif records operation-like event support. A category additionally requires object and morphism sorting, domain and codomain compatibility, identity witnesses, associativity proof-flow, unit proof-flow, and ledger. The exhibited composition-only flow omits the category-sorting support. □

9220.6 Homology as boundary-quotient flow

Definition 9220.36 (Chain-complex motif). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.ChainComplexMo`

A chain-complex motif *is an event structure containing graded object flow, boundary-map flow, boundary-composition flow, zero-composition proof-flow, and chain ledger flow. The key proof-flow states* $d \circ d = 0$.

Definition 9220.37 (Cycle-boundary classifier motif). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.Cycle`

A cycle-boundary classifier motif *is an event structure containing cycle flow, boundary flow, equivalence modulo boundaries, quotient classifier flow, and exactness ledger flow. It is classifier-heavy and ledger-dependent.*

Definition 9220.38 (Homology motif). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.HomologyMotif`

A homology motif *is chain-complex motif plus cycle-boundary classifier motif plus quotient ledger: chain complex, cycle classifier, boundary classifier, quotient flow, exactness proof-flow, and homology ledger.*

Proposition 9220.39 (Homology motif has classifier compression). Lean4 ✓ `BEDC.GroundCompiler.CaseStudies.h` `has classifier compression`

If S *contains a recognized homology motif, then* S *contains a classifier quotient motif.*

Proof. Homology motif includes equivalence modulo boundaries. This identifies raw cycles that differ by boundaries, so it contains classifier quotient structure. □

Theorem 9220.40 (Chain-complex motif does not imply homology). Lean4 ✓ `BEDC.GroundCompiler.CaseStudies` `complex not homology`

There exists a chain-complex-motif-only event flow whose chain support is present but whose homology quotient support is absent. Hence a chain-complex motif alone does not yield homology.

Proof. Chain-complex motif supplies graded objects and a boundary map with $d^2 = 0$. Homology requires forming cycles modulo boundaries. That formation needs classifier quotient and quotient ledger; the exhibited chain-complex-only flow omits that quotient support. □

9220.7 Cross-domain comparison

Definition 9220.41 (Cross-domain motif comparison). Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.Cros`

For event flows S *and* T*, define their cross-domain comparison by*

$$\mathsf{CrossCompare}_{\mathcal{R}}(S,T) = (\mathsf{MotifProfile}_{\mathcal{R}}(S), \mathsf{MotifProfile}_{\mathcal{R}}(T), \mathsf{Overlap}_{\mathcal{R}}(S,T), \mathsf{TheoryDist}_{\mathcal{R},P}(S,T)).$$

This comparison is valid only relative to recognizer family $\mathcal{R}$ *and analysis protocol* P.

Example 9220.42 ($\mathsf{Completion}^{\uparrow}$ versus $\mathsf{Real}^{\uparrow}$). *Expected shared motifs include finite approximation, compatible thread, seal, and tail ledger. Real-like structure adds Cauchy or regularity classifier, operation descent, order or metric compatibility, and field ledger. The report may say that* $\mathsf{Real}^{\uparrow}$ *is completion motif plus arithmetic, metric, and classifier stack, but it may not claim* $\mathsf{Completion}^{\uparrow} = \mathsf{Real}^{\uparrow}$.

Example 9220.43 (Phase structure versus circle)**.** *Expected shared motifs include periodic or cyclic source, seal, carry or wrap-around, and normal address. Circle-like structure adds unit constraint, equation classifier, topology compatibility, and circle ledger. Carry motif is not the circle object.*

Example 9220.44 ($\mathsf{Add}^{\uparrow}$ versus category composition)**.** *Expected shared motifs include composition or continuation, identity or unit, and associativity. Category structure adds object and morphism sorting, domain and codomain classifier, identity morphism ledger, and composability ledger. This supports the claim that they share a continuation-stability motif, not that* $\mathsf{Add}^{\uparrow}$ *is category composition.*

Example 9220.45 (Homology versus rational or Cauchy quotient)**.** *Expected shared motifs include classifier quotient, exactness or equivalence witness, and ledger preserving raw representatives. Homology adds chain complex,* $d^2 = 0$*, and cycle-boundary quotient. Cauchy quotient adds approximation sequence, tail-control classifier, and completion seal. Both are quotient-heavy, but their source and exactness motifs differ.*

Theorem 9220.46 (Cross-domain motif overlap does not imply bridge)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudi` `not bridge`

If two flows share motifs across domains, this does not imply that there is a bridge between the corresponding mature theories.

Proof. Motif overlap records structural similarity at event-flow level. A bridge requires translation flow, soundness flow, reflection or completeness where available, and no-host-leak ledger. Motif overlap supplies none of these. □

Corollary 9220.47 (Bridge requires bridge motif plus certificate)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudies.br` `needs cert`

Even when two theories have high motif similarity, a conservative bridge must be supplied as a separate recognized bridge flow with certificate support.

9220.8 Reports, limits, and conservativity

Definition 9220.48 (High-level motif report)**.** Lean4 (definition) `BEDC.GroundCompiler.CaseStudies.HighLevelMot`

A high-level motif report *for an event flow* S *includes its higher-layer motif bundle, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, bridge depth, normal-address map, warnings, and cannot-claim entries. It is an output view, not proof.*

Principle 9220.49 (High-level reports require cannot-claim entries)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudies.` `level reports require cannot claims`

Any report involving Real-like, Topology-like, Circle-like, Complex-like, Category-like, or Homology-like motifs must include cannot-claim entries preventing overinterpretation. Required examples include: Real-like motifs do not export $\mathsf{Real}^{\uparrow}$*; topology motif does not prove continuity; carry motif does not imply dimension; phase motif does not imply circle structure; complex motif does not imply complex analysis; composition motif does not imply category; chain-complex motif does not imply homology; motif overlap does not imply bridge.*

Theorem 9220.50 (High-level motif analysis is conservative over the finite kernel)**.** Lean4✓ `BEDC.GroundCompiler.CaseStudies.higher layer conservativity`

Adding high-layer motif analysis for Completion, Real, Topology, Circle, Complex, Category, Functor, Natural Transformation, and Homology does not add new raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, or accepted-object export rules.

Proof. High-layer motif analysis operates on decoded event flows and recognized motif profiles. It does not create new b0 or b1, add kernel sameness, change Ext or Cont, or license any derived interface. Object export still requires NameCert flow, derivation-certificate flow, and acceptance-gate flow. Therefore the analysis is conservative over the finite kernel. □

Corollary 9220.51 (High-layer motifs cannot replace certificates)**.** `Lean4✓ BEDC.GroundCompiler.CaseStudies.h` `layer motifs cannot replace certificates`

No high-layer motif bundle, metric report, or similarity score can replace NameCert *flow, derivation-certificate flow, proof flow, acceptance-gate flow, bridge certificate, or ledger completeness.*

Remark 9220.52 (Outlook)**.** *This chapter applies event-flow metrics to higher-layer routes. It defines and analyzes completion motif, Real motif bundle, metric motif, topology motif, neighborhood-basis motif, phase motif, circle motif, complex motif, complex analytic motif, category composition motif, functor motif, natural transformation motif, chain-complex motif, cycle-boundary classifier motif, homology motif, cross-domain motif comparison, and high-level motif reports. It proves that completion is seal-heavy and ledger-dependent, completion is necessary but not sufficient for Real-like structure, Real-like motifs do not export* $\mathsf{Real}^{\uparrow}$*, metric motifs do not automatically generate topology, topology motifs do not prove continuity, circle motifs extend phase motifs, carry motifs do not imply dimension, complex motifs do not imply complex analysis, functor and natural transformation motifs extend category and functor motifs, composition does not yield category, homology includes classifier quotient while chain-complex alone does not yield homology, cross-domain overlap does not imply bridge, high-level reports require cannot-claim entries, high-level motif analysis is conservative, and high-layer motifs cannot replace certificates. The next chapter organizes motifs, metrics, and reports into a compiler analysis pipeline from channel code decoding through recognizer selection, protocol flow, reports, distance matrices, cannot-claim audit, and bridge-obligation discovery.*

Chapter 9221

Compiler Analysis Pipeline

chapter 9220 applies event-flow metrics to Completion, Real, Topology, Circle, Complex, Category, and Homology routes. It shows that these theories can share finite approximation, seal, carry, classifier quotient, ledger compression, reuse, bridge, and composition motifs without thereby proving mature export, equality, or bridge correctness.

This chapter packages those ingredients into a compiler analysis pipeline:

$$\text{channel code} \longrightarrow \text{legality check} \longrightarrow \text{decode} \longrightarrow \text{protocol recognition} \longrightarrow \text{recognizer verification}$$
$$\longrightarrow \text{motif extraction} \longrightarrow \text{metric computation} \longrightarrow \text{report generation} \longrightarrow \text{cannot-claim audit} \longrightarrow \text{bridge-obligation}$$

The pipeline remains subject to the same no-hidden-input rule:

$$\boxed{\text{analysis is driven by protocol flows and recognizer flows, not by external configuration.}}$$

Recognizer families, analysis protocols, metric weights, report formats, cannot-claim audits, and bridge-obligation discovery must appear as recognized event-flow structure.

9221.1 Inputs and protocol recognition

Definition 9221.1 (Analysis input)**.** *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.AnalysisInput`

An analysis input *is either a legal channel stream*

$$c \in \mathsf{LegalZStream}$$

or equivalently an event flow

$$S \in \mathsf{EventFlow}.$$

The two presentations are related by the channel bijection:

$$S = \mathsf{Decode}(c), \qquad c = \mathsf{Compile}(S).$$

If the input is channel code, it must be decoded before analysis. If the input is already a source event flow, it may be analyzed directly.

Principle 9221.2 (Decode-first analysis)**.** *Lean4 ✓* `BEDC.GroundCompiler.AnalysisPipeline.decode first analysis`

Every analysis acts on source-level event flow. For $c \in \mathsf{LegalZStream}$*, the pipeline first forms*

$$S = \mathsf{Decode}(c).$$

All recognizer, motif, metric, ledger, cannot-claim, and bridge-obligation analysis then acts on S*, not on channel bit substrings.*

Definition 9221.3 (Analysis protocol candidate flow). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.`

An analysis protocol candidate flow *is an event flow*

$$P \in \mathsf{EventFlow}.$$

It has not yet been recognized as an analysis protocol. Thus $P \in \mathsf{EventFlow}$ *does not imply that* P *specifies a recognizer family, metrics, weights, report format, cannot-claim audit, or bridge discovery policy.*

Definition 9221.4 (Generated analysis-protocol recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPip`

A generated analysis-protocol recognizer *is an event flow*

$$R_P \in \mathsf{EventFlow}.$$

It is not an external configuration parser, YAML schema, host analysis script, or handwritten metric table. It is generated from the two raw marks and acts on event flows.

Definition 9221.5 (Analysis-protocol recognition relation). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipe`

The analysis-protocol recognition relation is written

$$\mathsf{RecognizesAnalysisProtocol}(R_P, P), \qquad R_P, P \in \mathsf{EventFlow}.$$

It is read as: the generated protocol recognizer flow R_P *recognizes event flow* P *as an analysis protocol flow. The relation records source-level support by requiring the generated recognizer flow and the protocol candidate flow to be present as nonempty event flows. If such an* R_P *exists,* P *is a recognized analysis protocol flow.*

Definition 9221.6 (Analysis protocol subflows). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.Analy`

If $\mathsf{RecognizesAnalysisProtocol}(R_P, P)$*, then* R_P *may recognize subflows of* P *as protocol roles:*

$$\begin{aligned}
&\mathsf{ProtInputDom}_{R_P}(P, P_0), && \mathsf{ProtRecognizerFamily}_{R_P}(P, P_1),\\
&\mathsf{ProtMotifSelection}_{R_P}(P, P_2), && \mathsf{ProtMetricSelection}_{R_P}(P, P_3),\\
&\mathsf{ProtMetricWeights}_{R_P}(P, P_4), && \mathsf{ProtNormalizationPolicy}_{R_P}(P, P_5),\\
&\mathsf{ProtLedgerPolicy}_{R_P}(P, P_6), && \mathsf{ProtCannotClaimPolicy}_{R_P}(P, P_7),\\
&\mathsf{ProtBridgeObligationPolicy}_{R_P}(P, P_8), && \mathsf{ProtReportPolicy}_{R_P}(P, P_9),\\
&\mathsf{ProtSeal}_{R_P}(P, \sigma_P).
\end{aligned}$$

These are not external field labels. They are source-level subflow roles recognized inside P*.*

Principle 9221.7 (No external analysis configuration). *Lean4✓* `BEDC.GroundCompiler.AnalysisPipeline.no external analysis config`

There is no formal input such as analysis YAML, metric JSON, motif table, weight vector, report template, manual object grouping, human theory category list, or external bridge candidate list. If such information is used, it must appear as a recognized subflow of an analysis protocol flow.

Theorem 9221.8 (Unrecognized protocol cannot drive analysis). *Lean4✓* `BEDC.GroundCompiler.AnalysisPipelin protocol rejected`

If there is no R_P *such that*

$$\mathsf{RecognizesAnalysisProtocol}(R_P, P),$$

then P *cannot be used as a formal analysis protocol.*

Proof. An analysis protocol determines recognizer family, metric selection, weights, normalization policy, ledger policy, cannot-claim policy, and report policy. If P is not recognized by a generated protocol recognizer, these choices have no event-flow support. Using P would introduce external analysis configuration, contrary to the no-hidden-input discipline. □

Corollary 9221.9 (Metric weights require protocol flow)**.** Lean4 ✓ `BEDC.GroundCompiler.AnalysisPipeline.metri` `weights require protocol flow`

Any weighted distance or similarity metric whose weights are not recorded by a recognized protocol flow is not a BEDC-admissible metric.

9221.2 Pipeline stages

Definition 9221.10 (Analysis pipeline)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.Analyze`

Given $c \in \mathsf{LegalZStream}$ and a recognized analysis protocol flow P, define

$$\mathsf{Analyze}(c, P)$$

as the following sequence of stages:

1. *legality check;*
2. *decode;*
3. *protocol recognition;*
4. *recognizer-family extraction;*
5. *motif extraction;*
6. *metric computation;*
7. *normal-address analysis;*
8. *ledger audit;*
9. *cannot-claim audit;*
10. *bridge-obligation discovery;*
11. *report generation.*

Every stage acts on event flows or recognized event-flow structures, not on channel substrings or external labels.

Definition 9221.11 (Stage 0: legality check)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.StageLeg`

Stage 0 checks $c \in \mathsf{LegalZStream}$. If this fails, the pipeline rejects the input.

Definition 9221.12 (Stage 1: decode)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.StageDecode`

Stage 1 forms

$$S := \mathsf{Decode}(c).$$

All later stages act on S and recognized subflows of S.

Definition 9221.13 (Stage 2: protocol recognition)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.St`

Stage 2 checks whether

$$\mathsf{RecognizesAnalysisProtocol}(R_P, P)$$

holds. If the protocol is not recognized, the pipeline rejects the analysis run.

Definition 9221.14 (Stage 3: recognizer-family extraction). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipe`

Stage 3 identifies the recognizer-family subflow

$$\mathsf{ProtRecognizerFamily}_{R_P}(P, P_1).$$

This subflow specifies generated recognizers

$$\mathcal{R}_P = (R_1, \ldots, R_k).$$

Each R_i must be certified. If an R_i is not certified, the pipeline rejects or records a bootstrap obligation.

Definition 9221.15 (Stage 4: motif extraction). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.Stagel`

Stage 4 computes

$$\mathsf{MotifProfile}_{\mathcal{R}_P}(S),$$

namely all triples (μ, M, L) such that some $R \in \mathcal{R}_P$ satisfies

$$\mathsf{RecognizesMotif}(R, S, M, \mu) \quad \textit{and} \quad \mathsf{MotifLedger}_R(S, M, \mu, L).$$

Motif recognition without a required ledger is recorded as a failure or cannot-claim item, not silently accepted.

Definition 9221.16 (Stage 5: metric computation). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.St`

Stage 5 computes the admissible metrics selected by the protocol: prefix profile, motif profile, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, bridge depth, normal-address distance, or theory-flow distance. If a metric requires a recognizer or ledger that is missing, the metric is undefined or blocked; the pipeline may not output a misleading numerical value.

Definition 9221.17 (Stage 6: normal-address analysis). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipelin`

Stage 6 uses the protocol's normalization-policy flow to recognize source-level normal-address records

$$(w, v, L_{w \to v}).$$

For example, $011 \mid 100$ *may be recognized as raw-to-normal carry ledger. This analysis acts on decoded event flow, never on channel bits.*

Definition 9221.18 (Stage 7: ledger audit). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.StageLedg`

Stage 7 checks whether recognized motifs, packages, certificates, theorem flows, and bridge candidates carry their required ledgers. It emits ledger-audit failure items for motif without ledger, package without ledger, NameCert *without ledger, completion without tail ledger, carry without pre-normal ledger, and bridge without no-host-leak ledger.*

Definition 9221.19 (Stage 8: cannot-claim audit). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.Sta`

Stage 8 uses the cannot-claim policy flow to attach required cannot-claim entries. Typical entries include: motif overlap does not imply object equality, same signature vector does not imply theorem equivalence, carry motif does not imply dimension, seal motif does not imply completion object, completion motif does not imply $\mathsf{Real}^{\uparrow}$*, topology motif does not prove continuity, complex motif does not imply complex analysis, bridge candidate does not imply bridge certificate, and code existence does not imply acceptance gate.*

Definition 9221.20 (Stage 9: bridge-obligation discovery). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipel`

Stage 9 identifies bridge-obligation candidates, such as strong motif overlap without recognized bridge flow, reuse-like source relation without reuse-certificate flow, or normal-address relation without classifier exactness flow. Discovery creates obligation candidates; it does not establish bridges.

Definition 9221.21 (Stage 10: report generation). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.Sta`

Stage 10 generates a human-readable report: motif report, metric report, ledger audit report, cannot-claim report, bridge-obligation report, distance matrix, or summary visualization. These are output views, not formal inputs and not proofs.

9221.3 Pipeline correctness

Definition 9221.22 (Well-formed analysis run). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.WellFo`

A well-formed analysis run *is a triple*

$$(c, P, O)$$

such that $c \in \mathsf{LegalZStream}$*,* P *is a recognized analysis protocol flow,* O *is the output or report flow generated by the stages of Definition 9221.10, every used recognizer is certified or recorded by bootstrap obligation, every nontrivial motif recognition has a ledger, and every high-risk claim has a cannot-claim entry.*

Theorem 9221.23 (Well-formed analysis run is source-determined). *Lean4 ✓* `BEDC.GroundCompiler.AnalysisPipe` `formed analysis run source determined`

If (c, P, O) *is a well-formed analysis run and* $S = \mathsf{Decode}(c)$*, then the formal output descriptor of the run is determined by* S*,* P*, and the recognizer family recognized inside* P*.*

Proof. Stages 0 and 1 uniquely decode c into S. Stage 2 fixes the recognized protocol P. Stage 3 extracts the recognizer family from P. Stages 4 through 10 are specified by S, P, and that recognizer family. No external metric table, YAML file, object label, or manual grouping is used. □

Corollary 9221.24 (Same decoded flow and protocol give same output). *Lean4 ✓* `BEDC.GroundCompiler.Analysis` `decoded same protocol same output`

If $\mathsf{Decode}(c) = \mathsf{Decode}(c')$ *and both runs use the same recognized protocol flow* P *and recognizer family, then the formal analysis output descriptors agree.*

Theorem 9221.25 (Pipeline is invariant under compile-decode round trip). *Lean4 ✓* `BEDC.GroundCompiler.Analys` `roundtrip invariant`

If $c = \mathsf{Compile}(S)$*, then for any recognized protocol flow* P *and recognizer family, the descriptor obtained from the code is the descriptor obtained from the decoded source:*

$$\mathsf{DescriptorFromCode}(\mathsf{Compile}(S), P, \mathcal{R}_P) = \mathsf{CanonicalDescriptor}(S, P, \mathcal{R}_P).$$

Proof. By the channel round trips, $\mathsf{Decode}(\mathsf{Compile}(S)) = S$ and, for legal c, $\mathsf{Compile}(\mathsf{Decode}(c)) = c$. Both sides decode to the same source event flow and use the same protocol. By Definition 9221.23, their outputs agree. □

Theorem 9221.26 (Pipeline is conservative over the finite kernel). *Lean4✓ BEDC.GroundCompiler.AnalysisPipeli pipeline conservativity*

Adding the analysis pipeline, protocol recognition, motif reports, metric reports, ledger audits, cannot-claim audits, and bridge-obligation discovery does not add new raw marks, history constructors, sameness rules, continuation rules, or derived-interface formation rules.

Proof. Pipeline stages act only on decoded event flows and recognized structures. They do not generate new b0 or b1, add msame, hsame, or psame rules, change Ext or Cont, or license derived interfaces. Object export still requires NameCert, derivation-certificate, and acceptance-gate flows. Bridge-obligation discovery is not bridge certification. Hence the pipeline is conservative over the finite kernel. □

9221.4 Report discipline

Definition 9221.27 (Report flow). *Lean4 (definition) BEDC.GroundCompiler.AnalysisPipeline.ReportFlow*

A report flow *is an event flow O recognized by a generated report recognizer as an output of the analysis pipeline. It may contain recognized motif lists, metric tables, normal-address tables, ledger audit results, cannot-claim entries, bridge-obligation candidates, failure cases, and summary remarks. It is output, not input.*

Principle 9221.28 (Reports are not evidence). *Lean4✓ BEDC.GroundCompiler.AnalysisPipeline.reports not evidence*

A report flow is not a proof-flow, certificate-flow, or accepted-object-flow. It can serve only as explanatory output unless separately recognized in some stronger role.

Proposition 9221.29 (Report cannot discharge certificate obligations). *Lean4✓ BEDC.GroundCompiler.AnalysisP cannot discharge certificate obligation*

If an object lacks NameCert *flow, derivation-certificate flow, or acceptance-gate flow, no report flow can supply the missing certificate obligation.*

Proof. NameCert, derivation-certificate, and acceptance-gate flows are recognized certificate structures required for object export. A report flow is analysis output and does not have the certificate-recognition role. Unless the same event flow is separately recognized as a certificate, it cannot discharge certificate obligations. □

Corollary 9221.30 (Motif report cannot license mature object). *Lean4✓ BEDC.GroundCompiler.AnalysisPipeline report cannot license mature object*

Even if a report shows completion, Real-like, Circle-like, or Homology-like motifs, it does not license the corresponding mature object.

Definition 9221.31 (Bridge-obligation candidate). *Lean4 (definition) BEDC.GroundCompiler.AnalysisPipeline.Bri*

A bridge-obligation candidate *is a recognized item B_{obl} in a report flow attached to a source event flow S and a target event flow T. It records support and evidence subflows saying that motif-level similarity suggests a possible bridge, together with the explicit condition that no recognized bridge flow from S to T has been supplied.*

Proposition 9221.32 (Bridge obligation is not bridge). *Lean4✓ BEDC.GroundCompiler.AnalysisPipeline.bridge obligation not bridge*

If B_{obl} is a bridge-obligation candidate attached to source event flow S and target event flow T, then B_{obl} does not establish a recognized bridge flow from S to T.

Proof. A bridge obligation records a missing bridge certificate. It explicitly reports that a bridge flow has not yet been supplied. It therefore cannot imply the bridge it asks for. □

Corollary 9221.33 (Bridge discovery remains noncommittal)**.** Lean4 ✓ `BEDC.GroundCompiler.AnalysisPipeline.discovery noncommittal`

Bridge-obligation discovery may guide later work, but each bridge-obligation candidate it records remains noncommittal: it does not establish correspondence, soundness, reflection, or conservative bridge between its source and target flows.

9221.5 Failure handling

Definition 9221.34 (Analysis failure item)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.AnalysisFa`

An analysis failure item *is an event subflow in a report flow recording failure at some pipeline stage, together with a failure kind. Common failure kinds include illegal channel stream, unrecognized protocol, uncertified recognizer, missing ledger, missing cannot-claim entry, undefined metric, missing classifier exactness, missing bridge certificate, normalization without raw ledger, and source/channel layer confusion.*

Definition 9221.35 (Failure-complete report)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.Failure`

For a list F_{req} of required analysis failure items and a list F_{rec} of recorded analysis failure items, a report flow is failure-complete *when every item of F_{req} occurs in F_{rec}.*

Theorem 9221.36 (Non-failure-complete report is inadmissible)**.** Lean4 ✓ `BEDC.GroundCompiler.AnalysisPipelin failure complete inadmissible`

If an analysis run requires a failure item $f \in F_{\mathrm{req}}$ and its report omits f from F_{rec}, then the report is not failure-complete and is not admissible as a failure-complete report.

Proof. A report that omits failures hides analysis obligations. It may make a metric appear defined when its recognizer is missing, or make a motif appear valid when its ledger is absent. The no-hidden-input discipline requires these conditions to be explicit. Failure omission therefore makes the report inadmissible. □

Corollary 9221.37 (Missing data cannot be silently treated as zero)**.** Lean4 ✓ `BEDC.GroundCompiler.AnalysisPipe data not zero`

If a metric is undefined because a recognizer or ledger is missing, the pipeline may not set it to zero by default. It must record a failure or undefined-metric item.

9221.6 Theorem and chapter code analysis

Definition 9221.38 (Theorem-code analysis)**.** Lean4 (definition) `BEDC.GroundCompiler.AnalysisPipeline.Theorem`

Given theorem code

$$c_T = \mathsf{Compile}(T),$$

where T is a sound theorem flow, define theorem-code analysis by

$$\mathsf{Analyze}(c_T, P).$$

Its report may contain proof motif profile, dependency motif profile, certificate motif profile, ledger depth, status flow, cannot-claim audit, and bridge-obligation candidates.

Definition 9221.39 (Chapter-code analysis). *Lean4 (definition)* `BEDC.GroundCompiler.AnalysisPipeline.ChapterC`

Given chapter code

$$c_C = \mathsf{Compile}(C),$$

where C is a sound chapter flow, define chapter-code analysis by

$$\mathsf{Analyze}(c_C, P).$$

Its report may contain definition, theorem, proof, and certificate motif profiles, certificate coverage, cannot-claim coverage, status distribution, ledger-depth distribution, reuse graph, and bridge-obligation list.

Proposition 9221.40 (Theorem-code analysis does not reprove theorem). *Lean4✓* `BEDC.GroundCompiler.Analysi` `code analysis not reprove`

The analysis $\mathsf{Analyze}(c_T, P)$ *does not reprove the theorem. It analyzes the structure of the theorem flow.*

Proof. The proof is contained in the proof flow and checked by proof recognizer or proof checker flow. The analysis pipeline extracts motifs and metrics; it does not perform proof checking unless proof checking is explicitly part of the protocol and recognized in that role. □

Proposition 9221.41 (Chapter-code analysis does not validate all chapter claims). *Lean4✓* `BEDC.GroundCompiler` `code analysis not validation`

The analysis $\mathsf{Analyze}(c_C, P)$ *does not automatically validate every claim in the chapter.*

Proof. Chapter validation requires chapter recognition, theorem recognition, proof checking, certificate recognition, ledger audit, and status flow. The analysis protocol may include some of these stages, but motif or metric analysis alone does not validate all claims. The report therefore records only the structures, metrics, failures, and obligations recognized by the selected protocol. □

9221.7 Examples and determinacy

Example 9221.42 (Analyze completion skeleton code). *For input code*

$$011001100011010110101011100011,$$

Stage 1 decodes

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Possible recognized motifs are finite repetition $0 \mid 00 \mid 000$*, repeat boundary* 01*, seal* 011*, and carry* $011 \mid 100$*. A report may record seal depth* 1*, carry index* 1*, ledger depth at least* 1 *when carry ledger is recognized, and cannot-claim entries saying that the completion skeleton does not license* $\mathsf{FullAxis}^{\uparrow}$*, carry motif does not imply dimension, and motif report does not imply object equality.*

Example 9221.43 (Analyze fold skeleton code). *For input code*

$$011001100101100101011010 0011,$$

Stage 1 decodes

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

Possible recognized motifs are finite source $0 \mid 00$*, additive recursion* 001*, and fold completion* $0011 \mid 0100$*. A report may record carry-index candidate and reuse-depth candidate, while retaining cannot-claim entries that additive recursion does not license* $\mathsf{Add}^{\uparrow}$ *and fold motif does not license* $\mathsf{Mul}^{\uparrow}$*.*

Example 9221.44 (Compare completion and fold skeletons)**.** *For*

$$S_{\mathrm{Comp}} = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

and

$$S_{\mathrm{Fold}} = 0 \mid 00 \mid 001 \mid 0011 \mid 0100,$$

the raw prefix overlap is $0 \mid 00$*. A report may say that both share finite-source motif, while the completion skeleton branches into repeat-boundary, seal, and carry, and the fold skeleton branches into additive recursion and fold completion. It may not claim that shared prefix implies subobject, that shared finite-source motif implies the same theory, or that fold and completion are bridge-equivalent.*

Theorem 9221.45 (Analysis is deterministic relative to protocol)**.** `Lean4✓` `BEDC.GroundCompiler.AnalysisPipeli` `deterministic relative protocol`

If $c \in \mathsf{LegalZStream}$*,* P *is a recognized analysis protocol flow, and all recognizers selected by* P *are deterministic on the decoded input domain, then* $\mathsf{Analyze}(c, P)$ *is uniquely determined.*

Proof. The code c uniquely decodes to S. The protocol P fixes recognizer family, metric selection, weights, normalization policy, ledger policy, cannot-claim policy, and report policy. If each selected recognizer is deterministic on the domain, then motif extraction, metric computation, ledger audit, and report generation are uniquely determined. □

Corollary 9221.46 (Nondeterministic recognizer requires nondeterminism ledger)**.** *If the recognizer family contains a nondeterministic recognizer, the analysis output must record choice branches, failure branches, or all possible outputs. Otherwise the report is not admissible.*

Remark 9221.47 (Outlook)**.** *This chapter establishes the compiler analysis pipeline: legality check, decode, protocol recognition, recognizer-family extraction, motif extraction, metric computation, normal-address analysis, ledger audit, cannot-claim audit, bridge-obligation discovery, and report generation. It defines analysis inputs, protocol candidate flows, protocol recognizers, protocol subflows, well-formed runs, report flows, bridge-obligation candidates, analysis failure items, failure-complete reports, theorem-code analysis, chapter-code analysis, and protocol-relative determinacy. It proves that unrecognized protocols cannot drive analysis, metric weights require protocol flow, well-formed runs are source-determined, equal decoded flow and protocol give equal output, the pipeline is invariant under compile-decode round trip and conservative over the finite kernel, reports cannot discharge certificates, bridge obligation is not bridge, incomplete failure reporting is inadmissible, missing data cannot silently become zero, theorem-code analysis does not reprove theorem, chapter-code analysis does not validate all claims, deterministic protocols determine analysis output, and nondeterminism requires ledger. The next chapter collects the main theorem of the no-hidden-input compiler discipline.*

Chapter 9222

Main Theorems and Cannot-Claim Registry

The preceding chapters establish the components of the no-hidden-input compiler discipline: ground event flows, Zeckendorf-safe channel encoding, source/channel separation, package recognition, certificate recognition, theorem and chapter recognition, recognizer flows, self-hosting compiler flows, motif analysis, theory metrics, higher-layer case studies, and the compiler analysis pipeline.

This chapter collects the main theorems and records the cannot-claim registry for this part. The governing assertion is:

$$\boxed{\text{the only formal ground input is a finite event flow generated from } \mathsf{b0}, \mathsf{b1}.}$$

All higher structures are recognized structures inside event flows:

package, certificate, acceptance gate, theorem, chapter, recognizer, compiler, motif, metric, and report.

None of these may enter as an external object.

9222.1 Compiler state and hidden input

Definition 9222.1 (No-hidden-input compiler state)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.NoHid`

A no-hidden-input compiler state *is a tuple*

$$\mathcal{C} = (\mathsf{EventFlow}, \mathsf{LegalZStream}, \mathsf{Compile}, \mathsf{Decode}, \mathsf{RecognizerHierarchy}, \mathsf{CertificateFlows}, \mathsf{LedgerFlows}, \mathsf{BootstrapObligat}$$

satisfying:

1. $\mathsf{EventFlow} = (\{0,1\}^*)^*$*;*
2. $\mathsf{LegalZStream} = ((0 \mid 10)^*11)^*$*;*
3. $\mathsf{Compile}$ *is the Zeckendorf-safe channel compiler;*
4. Decode *is its streaming inverse decoder;*
5. *package, certificate, theorem, chapter, and compiler structures are recognized by the recognizer hierarchy;*
6. *every nontrivial recognition carries ledger data;*

7. *every not-yet-self-hosted component is recorded as a bootstrap obligation.*

Definition 9222.2 (Hidden input). *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.HiddenInput`

A datum X is a hidden input *when it is used for compilation, recognition, proof, analysis, or reporting, but X is not any of:*

1. *a raw event;*
2. *an event flow;*
3. *a recognized event-flow structure;*
4. *a recognized certificate flow;*
5. *a recognized ledger flow;*
6. *a recognized bootstrap obligation;*
7. *a recognized analysis protocol flow;*
8. *a self-hosted compiler flow, or an explicitly recorded bootstrap boundary.*

Typical hidden inputs include external atom tables, external enums, YAML, manifests, host ASTs, external package structures, theorem identifiers, external proof objects, host parsers, host proof checkers, external metric tables, motif tables, type tags, mode tags, unrecorded package boundaries, and unrecorded classifier oracles.

Principle 9222.3 (No hidden input). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.no hidden input`

The formal compiler discipline may not use hidden input. If a component is not yet generated and recognized as event-flow structure, it must be explicitly recorded as a bootstrap obligation. Otherwise it may not support a formal claim.

9222.2 Channel and layer separation

Theorem 9222.4 (Channel bijection theorem). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.channel bijection`

The channel compiler gives a bijection

$$\boxed{\mathsf{EventFlow} \cong \mathsf{LegalZStream}}$$

where

$$\mathsf{EventFlow} = (\{0,1\}^*)^*, \qquad \mathsf{LegalZStream} = ((0 \mid 10)^* 11)^*.$$

The event encoding is

$$\zeta(0) = 0, \qquad \zeta(1) = 10, \qquad \mathsf{Ev}(w) = \zeta(w)11,$$

and

$$\mathsf{Compile}(w_0, \ldots, w_{n-1}) = \mathsf{Ev}(w_0) \cdots \mathsf{Ev}(w_{n-1}).$$

Proof. The channel-encoding chapter proves that event bodies lie in $(0 \mid 10)^*$ and contain no internal adjacent 11. Hence the first 11 after an event body is the event terminator. The streaming decoder therefore uniquely recovers each raw event and then the whole event flow. Thus

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S$$

and, for every legal stream c,

$$\mathsf{Compile}(\mathsf{Decode}(c)) = c.$$

So Compile and Decode are inverse maps. □

Corollary 9222.5 (Channel code is lossless). *Lean4✓ BEDC.GroundCompiler.MainTheorems.channel code lossless*

If

$$\mathsf{Compile}(S) = \mathsf{Compile}(T),$$

then $S = T$.

Proof. Apply Decode to both sides and use the round-trip law. □

Corollary 9222.6 (Channel code is not proof). *Lean4✓ BEDC.GroundCompiler.MainTheorems.code not proof*

The judgment $c \in \mathsf{LegalZStream}$ *only says that* c *decodes to an event flow. It does not say that the flow is a theorem, proof, name certificate, derivation certificate, accepted object, or mature interface.*

Theorem 9222.7 (Source/channel layer separation theorem). *Lean4✓ BEDC.GroundCompiler.MainTheorems.sourc channel separation*

Source events and channel streams are strictly separated:

1. *source events belong to* $\mathsf{RawEvent} = \{0,1\}^*$*;*
2. *channel streams belong to* LegalZStream*;*
3. *channel-level* 11 *is an event terminator;*
4. *source-level* 11 *may be part of a raw event;*
5. *source-level normalization acts only on decoded event flows;*
6. *channel substrings cannot be used directly as source events.*

Proof. The first two clauses are definitions. The third follows from $\mathsf{Ev}(w) = \zeta(w)11$. The fourth follows because raw events are arbitrary finite binary words; the source event 11 has channel code $\mathsf{Ev}(11) = 101011$, not 11. The fifth follows because source-level normalization is defined over EventFlow. For the sixth, the channel code 1011 contains the substring 011, but decodes to the source event 1, not the source event 011. □

Corollary 9222.8 (Source carry is not channel rewrite). *Lean4✓ BEDC.GroundCompiler.MainTheorems.carry not channel rewrite*

A source-level relation such as $011 \rightsquigarrow 100$ *cannot be applied as a string rewrite on compiled channel bits. The valid order is decode, source-level normalization, and compile.*

9222.3 Recognized structures and exports

Definition 9222.9 (Generated structure). *Lean4 (definition) BEDC.GroundCompiler.MainTheorems.GeneratedStruct*

A structure K *is a* generated structure *when there are an event flow* S *and a generated recognizer flow* R *such that*

$$\mathsf{Recognizes}_K(R, S).$$

Here $\mathsf{Recognizes}_K$ *may be package recognition,* $\mathsf{NameCert}$ *recognition, derivation-certificate recognition, theorem recognition, chapter recognition, compiler recognition, motif recognition, or analysis-protocol recognition.*

Theorem 9222.10 (Structure emergence theorem)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.structure emergence

Packages, name certificates, derivation certificates, acceptance gates, theorems, chapters, recognizers, compilers, motifs, metrics, and reports are not first-layer inputs. They may appear only as generated structures.

Proof. If any of these structures were first-layer inputs, their fields, boundaries, roles, order, nesting, or intended reading would be externally supplied. That is hidden input. By Definition 9222.3, such input is not admissible. Therefore these structures must be supplied by event flows and generated recognizer flows. □

Corollary 9222.11 (YAML and AST are output views only)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.yaml ast output only

YAML, ASTs, Markdown outlines, JSON reports, theorem tables, and dependency graphs may be human-readable output views of decoded event flows. They are not formal inputs.

Corollary 9222.12 (Recognizer generatedness)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.recognizer generatedness

A recognizer cannot be a host parser or external function. A recognizer used in a formal recognition judgment must itself be an event flow and must carry a recognizer certificate flow.

Definition 9222.13 (Certificate-mediated export)**.** Lean4 (definition) BEDC.GroundCompiler.MainTheorems.Certifi

An object candidate N *is a* certificate-mediated export *at strength* s *when there are recognized name-certificate, derivation-certificate, and acceptance-gate flows such that*

$$\mathsf{AcceptGateFlow}(N, s)$$

holds.

Theorem 9222.14 (Accepted export theorem)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.accepted export

A mature object N *may be exported at strength* s *exactly when its decoded event flow supplies the recognized name-certificate, derivation-certificate, and acceptance-gate structure:*

$$\boxed{\mathsf{AcceptedExport}(N, s) \iff \mathsf{AcceptGateFlow}(N, s).}$$

Proof. By definition, $\mathsf{AcceptGateFlow}(N, s)$ holds exactly when there is an acceptance flow containing a sound name-certificate flow and a sound derivation-certificate flow. The original acceptance discipline requires $\mathrm{NameCert}(N) \wedge \mathrm{DerivCert}(N, s)$. The compiler discipline reads those two components as recognized event-flow certificates. If either certificate is missing, export is not licensed; if both are soundly recognized and accepted, export is licensed. □

Corollary 9222.15 (Code existence does not imply export)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.code existence not export

From $c \in \mathsf{LegalZStream}$ *alone one cannot infer that* c *recognizes* $\mathsf{AcceptedExport}(N, s)$*. The stream must first be decoded, and the required certificate and acceptance flows must be recognized.*

Corollary 9222.16 (Motif existence does not imply export)**.** Lean4✓ BEDC.GroundCompiler.MainTheorems.motif existence not export

Completion, carry, circle, complex, or homology motifs do not export mature objects. Mature export remains certificate-mediated.

9222.4 Theorem, chapter, and quotient layers

Definition 9222.17 (Canonical theorem flow). *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.CanonicalTh`

A canonical theorem flow *is a sound recognized theorem flow with a recognized canonical-site subflow. Write*

$$\mathsf{CanonicalTheoremFlow}(T).$$

Definition 9222.18 (Theorem code). *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.TheoremCode`

If $\mathsf{CanonicalTheoremFlow}(T)$*, define*

$$\mathsf{ThmCode}(T) := \mathsf{Compile}(T).$$

Theorem 9222.19 (Theorem flow / theorem code bijection). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.theore` `code bijection`

There is a bijection

$$\boxed{\mathsf{CanonicalTheoremFlow} \cong \mathsf{LegalTheoremCode}_{01}.}$$

Proof. If T is a canonical theorem flow, then $\mathsf{Compile}(T)$ is a legal channel code. If canonical theorem flows T and U have the same code, then

$$\mathsf{Compile}(T) = \mathsf{Compile}(U)$$

implies $T = U$ by the channel bijection. Conversely, a legal theorem code is by definition the compiled code of a canonical theorem flow, and decoding recovers that flow. □

Corollary 9222.20 (Proposition is not theorem code). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.proposition` `not theorem code`

An abstract proposition is generally not in bijection with theorem code. One proposition may have multiple theorem flows, with different proofs, dependency expansions, certificate strengths, ledgers, canonical sites, and bridge statuses.

Corollary 9222.21 (Theorem code is not proof). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.theorem` `code not proof`

Theorem code records the full theorem flow. Proof is a recognized subflow of that theorem flow and requires generated proof-checker flow. Code does not replace proof checking.

Definition 9222.22 (Canonical chapter flow). *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.CanonicalCh`

A canonical chapter flow *is a sound recognized chapter flow with recognized chapter seal, status, ledger, and cannot-claim subflows. Write*

$$\mathsf{CanonicalChapterFlow}(C).$$

Definition 9222.23 (Chapter code). *Lean4 (definition)* `BEDC.GroundCompiler.MainTheorems.ChapterCode`

If $\mathsf{CanonicalChapterFlow}(C)$*, define*

$$\mathsf{ChapCode}(C) := \mathsf{Compile}(C).$$

Theorem 9222.24 (Chapter flow / chapter code bijection). *Lean4✓* `BEDC.GroundCompiler.MainTheorems.chapter` `code bijection`

There is a bijection

$$\boxed{\mathsf{CanonicalChapterFlow} \cong \mathsf{LegalChapterCode}_{01}.}$$

Proof. The proof is the same as for theorem codes. The channel compiler is injective on event flows, and every legal chapter code is the compiled code of a canonical chapter flow. □

Corollary 9222.25 (Chapter topic is not chapter code). Lean4✓ `BEDC.GroundCompiler.MainTheorems.topic not chapter code`

An abstract chapter topic is not in bijection with chapter code. Chapter code corresponds to a complete canonical chapter flow, including definitions, lemmas, theorems, proofs, certificates, ledgers, cannot-claims, status, and chapter seal.

Definition 9222.26 (Raw-flow code layer). Lean4 (definition) `BEDC.GroundCompiler.MainTheorems.RawFlowCodeLay`

The raw-flow code layer *is the graph of the channel coding relation:*

$$\mathsf{RawFlowCodeLayer}(S, c) \iff c = \mathsf{Compile}(S).$$

The channel bijection theorem identifies this graph with the lossless correspondence $\mathsf{EventFlow} \cong \mathsf{LegalZStream}$.

Definition 9222.27 (Classified object layer). Lean4 (definition) `BEDC.GroundCompiler.MainTheorems.ClassifiedOb`

Given a classifier $\sim$ on event flows, the classified-object layer records the co-classification relation between event flows:

$$\mathsf{ClassifiedObjectLayer}(\sim, S, T) \iff S \sim T.$$

This is the constructive representative of the class notation $[S]_\sim$: membership in the class is expressed by a classifier witness rather than by a primitive quotient object.

Theorem 9222.28 (Classifier quotient is generally many-to-one). Lean4✓ `BEDC.GroundCompiler.MainTheorems.cl quotient many to one`

The projection

$$S \longmapsto [S]_\sim$$

is generally not injective. Hence the classified object layer is generally not in bijection with the raw code layer.

Proof. Take the classifier that identifies event flows with the same boundary-erased display string. The distinct flows

$$(0, 00, 000) \qquad \text{and} \qquad (00, 0000)$$

both erase to 000000, so they are co-classified by this classifier while remaining distinct raw event flows. Thus distinct raw flows may map to the same classified object. A source-level Zeckendorf carry can likewise classify 011 and 100 together while keeping them distinct as raw events. □

Corollary 9222.29 (Normal address requires ledger). Lean4✓ `BEDC.GroundCompiler.MainTheorems.normal address requires ledger`

If a normal representative is used as an object address, the raw ledger must be retained; otherwise the pre-normal flow is lost.

9222.5 Recognizers, compilers, motifs, and metrics

Theorem 9222.30 (Recognizer generatedness theorem). Lean4 ✓ BEDC.GroundCompiler.MainTheorems.recognizer generatedness theorem

Every formal recognizer must be a certified event flow. Outputs of host parsers, external grammars, YAML schemas, AST constructors, or meta-level elaborators are not formal recognition judgments unless supported by event-flow recognizer certificates.

Proof. If a recognizer is not an event flow, its recognition power comes from outside the system and is hidden input. If it is an event flow but has no certificate, then soundness, completeness, ledger behavior, failure behavior, and conservativity are not established. Therefore a formal recognizer must be a certified event flow. □

Definition 9222.31 (Self-hosted compiler). Lean4 (definition) BEDC.GroundCompiler.MainTheorems.SelfHostedComp

A compiler flow C is self-hosted *when:*

1. *C is a certified compiler flow;*
2. *C is recognized by a compiler recognizer;*
3. *C compiles itself to a flow C';*
4. *$C' \equiv_{\text{comp}} C$;*
5. *a self-hosting ledger records raw C, compiled code, decoded C', a behavior-classifier witness, and remaining bootstrap obligations.*

Theorem 9222.32 (Self-hosting removes compiler-as-hidden-input). Lean4 ✓ BEDC.GroundCompiler.MainTheorems hosting removes hidden compiler

If there is a self-hosted compiler flow C, and if the package, name-certificate, derivation-certificate, theorem, chapter, and analysis recognizers are supported by the recognizer hierarchy of C, then the compiler is no longer a host-level hidden input.

Proof. Self-hosting states that the compiler itself is an event flow and that compiling it yields a behavior-equivalent flow. The recognizer hierarchy supplies structure recognition without host parsers. The self-hosting ledger records the raw compiler, compiled code, decoded output, behavior equivalence, and bootstrap obligations. Thus formal compiler behavior is supplied by event-flow structure rather than by an external program. □

Corollary 9222.33 (Declared bootstrap boundary). Lean4 ✓ BEDC.GroundCompiler.MainTheorems.declared bootstrap boundary

Without self-hosting, every component not discharged by the event-flow compiler must be recorded as a bootstrap obligation. Otherwise full no-hidden-input status cannot be claimed.

Theorem 9222.34 (Motif analysis theorem). Lean4 ✓ BEDC.GroundCompiler.MainTheorems.motif analysis

Motif analysis is admissible only when motifs are recognized by generated motif recognizers and every nontrivial motif carries ledger data.

Proof. A motif is a source-level structure. If it is obtained from an external pattern table or a channel substring search, then it is hidden input or wrong-layer analysis. If it has no ledger, its raw support, normalization, classifier use, or compression is not traceable. Therefore admissible motif analysis requires generated recognizer flow and ledger flow. □

Theorem 9222.35 (Metric conservativity theorem). Lean4✓ BEDC.GroundCompiler.MainTheorems.metric conservativity

Admissible metrics, motif reports, distance matrices, analysis reports, and bridge-obligation reports add no finite-kernel rules and license no mature objects.

Proof. These structures analyze decoded event flows and recognized motifs. They generate no raw marks, add no sameness or continuation rules, provide no name certificate or derivation certificate, and do not trigger an acceptance gate. Hence they are conservative analysis structures. □

Corollary 9222.36 (Similarity is not identity). Lean4✓ BEDC.GroundCompiler.MainTheorems.similarity not identity

Motif similarity, equal signatures, zero distance, and shared prefixes do not imply theorem equivalence, object equality, bridge correctness, or accepted export without separate classifier, bridge, proof, or certificate flow.

9222.6 Cannot-claim registry

Definition 9222.37 (Compiler cannot-claim entry). Lean4 (definition) BEDC.GroundCompiler.MainTheorems.Compil

A compiler cannot-claim entry *contains at least:*

ClaimText	*the blocked claim;*
Status	*the present recognition status;*
WhyNotDischarged	*the missing certificate or ledger;*
RequiredUpgrade	*the flow needed to discharge the block;*
Ledger	*the event-flow record of the block.*

Definition 9222.38 (Required cannot-claim entries). Lean4 (definition) BEDC.GroundCompiler.MainTheorems.Requi

The no-hidden-input compiler part requires at least the following cannot-claim entries.

C1. Arbitrary binary strings have BEDC meaning. *One may not claim that every binary string has BEDC meaning. Only legal channel streams decode to event flows, and only recognized event flows carry structures.*

C2. A channel substring is a source event. *One may not read substrings of channel code as source events. Source events are obtained only after decoding.*

C3. Source-level carry is channel rewrite. *One may not apply* $011 \rightsquigarrow 100$ *directly to compiled channel bits. The valid order is decode, normalize source events, and compile.*

C4. Code existence implies proof. *One may not infer proof from the existence of code. Theorem code records theorem flow; proof-flow and proof-checker flow are required.*

C5. Code existence implies accepted object. *One may not infer accepted export from code existence. Accepted export requires name-certificate flow, derivation-certificate flow, and acceptance-gate flow.*

C6. Package, certificate, theorem, and chapter structures are inputs. *One may not use package objects, certificate objects, theorem packages, or chapter packages as compiler inputs. They are recognized structures inside event flows.*

C7. A recognizer may be a host parser. *An external parser or YAML schema does not count as a formal recognizer. A recognizer must be a certified event flow or an explicit bootstrap obligation.*

C8. Classifier quotient preserves raw-code bijection. *Raw flows biject with codes. Classified objects are quotients and are generally many-to-one over raw flows.*

C9. Motif similarity implies object equality. *Motifs explain structural similarity. Equality requires classifier, proof, bridge, or certificate flow.*

C10. Carry motif implies dimension. *Carry motif is a normalization or lift witness. Geometric, topological, linear, or manifold dimension requires additional certificate flow.*

C11. Seal motif implies completion object. *A seal-like event does not automatically produce a completion object. Completion requires compatible-thread source, classifier, seal certificate, and ledger.*

C12. Report is proof. *Metric and motif reports are output views. Proof and certificate flows are separate.*

C13. Self-hosting implies raw compiler equality. *Self-hosting gives behavior-classifier equivalence $C' \equiv_{\mathrm{comp}} C$, not raw event-flow equality $C' = C$.*

C14. Full no-hidden-input claim without self-hosting. *One may not claim a fully no-hidden-input compiler unless a self-hosting compiler flow exists, or all remaining bootstrap obligations are explicit.*

Theorem 9222.39 (Cannot-claim registry is mandatory)**.** Lean4✓ `BEDC.GroundCompiler.MainTheorems.cannot claim registry mandatory`

A cannot-claim registry is adequate only if each required entry C1–C14 has a registry entry with a nonempty ledger flow. Consequently, every adequate registry contains ledger-backed entries for C1–C14.

Proof. Each entry blocks a known failure mode: invalid channel analysis, hidden structure input, code-as-proof confusion, classifier/equality collapse, motif-as-object confusion, or hidden bootstrap. Adequacy of the registry is the finite condition that every required cannot-claim code has an entry whose ledger is present as an event flow. Applying that condition to C1–C14 gives the required ledger-backed entries. □

9222.7 Global conservativity and final theorem

Theorem 9222.40 (Global conservativity theorem)**.** Lean4✓ `BEDC.GroundCompiler.MainTheorems.global conservativity`

The no-hidden-input streaming compiler layer is conservative over the BEDC finite kernel. It adds no raw marks, history constructors, msame *rules,* hsame *rules,* psame *rules,* Ext/Cont *rules, kernel equalities, derived-interface formation rules, or mature-object primitives.*

Proof. Every construction in this part is one of: an event flow over the existing display alphabet, a channel encoding or decoding relation, a recognition relation over event flows, a certificate recognition relation, an analysis relation over decoded event flows, a report output view, a bootstrap obligation, or a behavior classifier over compiler flows. None introduces BEDC kernel constructors

or equality rules. Derived interfaces still require recognized name-certificate flows. Mature exports still require recognized derivation-certificate flows and acceptance gates. Classifier relations remain classifier-level, not kernel equality. □

Corollary 9222.41 (Compiler layer is an address-and-analysis layer)**.** *Lean4✓ BEDC.GroundCompiler.MainTheorem layer address analysis layer*

The compiler layer provides lossless event-flow coding, recognizer-mediated structure extraction, certificate-flow tracking, theorem and chapter coding, motif and metric analysis, cannot-claim auditing, and bootstrap accounting. It provides no new mathematical primitives.

Theorem 9222.42 (No-Hidden-Input Streaming Compiler Theorem)**.** *Lean4✓ BEDC.GroundCompiler.MainTheorem hidden input streaming compiler*

There is a formal compiler architecture satisfying all of the following:

1. *The only ground input is a finite* $\mathsf{b0}, \mathsf{b1}$*-generated event flow.*
2. *The channel compiler gives a bijection*

$$\mathsf{EventFlow} \cong ((0 \mid 10)^* 11)^*.$$

3. *Package, name-certificate, derivation-certificate, acceptance-gate, theorem, chapter, recognizer, compiler, motif, metric, and report structures are not inputs; they are recognized event-flow structures.*
4. *Recognizers themselves are certified event flows or explicitly recorded bootstrap obligations.*
5. *Mature-object export is mediated only by recognized name-certificate, derivation-certificate, and acceptance-gate flows.*
6. *Theorem codes are bijective with canonical theorem flows, not with abstract propositions.*
7. *Chapter codes are bijective with canonical chapter flows, not with abstract chapter topics.*
8. *Classifier quotient generally destroys raw-code bijection at the object layer; raw ledger must be retained.*
9. *Source-level normalization never acts directly on channel bits.*
10. *Motif and metric analyses are explanatory and conservative; they do not replace proof or certificate.*
11. *Self-hosting compiler flow, if achieved, removes compiler-as-hidden-input up to compiler-behavior classification.*
12. *Until self-hosting is achieved, bootstrap boundaries must be explicitly recorded.*
13. *The compiler layer is conservative over the BEDC finite kernel.*

Proof. The first clause follows from the ground event-flow definition. The second follows from Definition 9222.4. The third follows from Definition 9222.10 and the recognition chapters. The fourth follows from Definition 9222.30. The fifth follows from Definition 9222.14. The sixth and seventh follow from the theorem-code and chapter-code bijections. The eighth follows from Definition 9222.28. The ninth follows from Definition 9222.7. The tenth follows from Definition 9222.34 and Definition 9222.35. The eleventh follows from Definition 9222.32. The twelfth follows from Definition 9222.33. The final clause follows from Definition 9222.40. □

Remark 9222.43 (Summary)**.** *The only ground input is a* `b0, b1` *event flow. Event flows and legal channel streams are in bijection. Higher structures are recognized event-flow structures, not inputs. Theorem codes biject with canonical theorem flows; chapter codes biject with canonical chapter flows. Mature export still requires name-certificate, derivation-certificate, and acceptance-gate flows. Classifier quotients, motif similarity, and metric reports do not replace proof or certificate. The compiler layer is conservative over the BEDC finite kernel.*

Chapter 9223

Implementation Interface and Verification Targets

chapter 9222 establishes the main theorem of the no-hidden-input streaming compiler:

$$\mathsf{EventFlow} \cong ((0 \mid 10)^* 11)^*.$$

It also fixes the rule that packages, certificates, theorems, chapters, recognizers, and compilers are not first-layer inputs. They must appear as structures recognized inside event flows.

This chapter moves to the implementation interface while preserving the same discipline:

implementations may use host data structures, but the public formal interface may not expose them as BEDC p

Host representations such as lists, arrays, strings, parser states, Lean structures, Rust structs, or Python dataclasses may be engineering devices. The formal surface exposes only event-flow relations such as IsEventFlow, Compiles, Decodes, and the recognition judgments.

9223.1 Implementation layer and formal surface

Definition 9223.1 (Implementation representation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInte`

An implementation representation *is a host-language data structure used temporarily to represent event flows, bit streams, recognition results, or reports. Examples include host lists of booleans, arrays of bits, strings, vectors, parser states, JSON objects, Lean structures, Rust structs, and Python dataclasses.*

Such representations may be used in engineering implementations, but they are not BEDC public mathematical objects.

Definition 9223.2 (Public formal interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Pu`

A public formal interface *is an interface stated only in BEDC event-flow relations. Allowed public forms include*

$$\mathsf{IsRawEvent}(x), \quad \mathsf{IsEventFlow}(S), \quad \mathsf{IsLegalZStream}(c),$$

$$\mathsf{Compiles}(S, c), \quad \mathsf{Decodes}(c, S), \quad \mathsf{RecognizesPkg}(R, S),$$

$$\mathsf{RecognizesNameCert}(R, S, N), \quad \mathsf{RecognizesDerivCert}(R, D, N, s),$$

$$\mathsf{RecognizesTheorem}(R, T), \quad \mathsf{RecognizesChapter}(R, C), \quad \mathsf{RecognizesCompiler}(R, C),$$

and

$$\mathsf{AcceptFlow}(A, N, s).$$

Disallowed public forms include maps from host package structures to strings, parsers from YAML to theory packages, recognizers from host ASTs to theorems, and proof checkers from host proof objects to booleans.

Principle 9223.3 (Host representation is not BEDC structure). Lean4 ✓ BEDC.GroundCompiler.ImplementationIn representation not structure

Host representation may implement BEDC relations, but it must not become BEDC public structure. A host list may implement an event-flow representation, but EventFlow *is not a mathematical primitive defined by host lists. A parser state may implement decoding, but a parser state is not a source-level theorem witness.*

Definition 9223.4 (Implementation soundness). Lean4 (definition) BEDC.GroundCompiler.ImplementationInterfac

Let an implementation function or program f_{impl} claim to implement a formal relation $\mathcal{R}$. The implementation is sound *for $\mathcal{R}$ when every success returned by f_{impl} implies the corresponding formal relation.*

For example, if an implementation decoder returns success with output S on input c, then one must have

$$\mathsf{Decodes}(c, S).$$

Definition 9223.5 (Implementation completeness). Lean4 (definition) BEDC.GroundCompiler.ImplementationInter

An implementation f_{impl} is complete relative to a domain *D for a formal relation $\mathcal{R}$ when every instance of $\mathcal{R}$ on input in D is returned by the implementation.*

For example, if $c \in \mathsf{LegalZStream}$ and $\mathsf{Decodes}(c, S)$, then a complete decoder over the legal-stream domain returns success with output S. Completeness is always domain-relative.

Definition 9223.6 (No-host-leak condition). Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface.No

An implementation satisfies the no-host-leak condition *when its public theorem statements, certificate statements, and compiler claims do not expose host-level primitives as BEDC objects. In particular, the public surface may not treat host natural numbers, booleans, lists, strings, equality, quotients, function spaces, parser states, ASTs, theorem identifiers, file paths, YAML, JSON, choice, or classical oracles as BEDC objects.*

These devices may appear inside implementation proofs as tools, but not as the BEDC public surface.

Theorem 9223.7 (Host leak invalidates no-hidden-input status). Lean4 ✓ BEDC.GroundCompiler.Implementationl leak invalidates

If an implementation's public interface contains a datum that is both public input and an implementation representation, then the implementation cannot claim no-hidden-input compiler status.

Proof. The no-hidden-input status includes the no-host-leak condition for the public surface. A public datum that is also an implementation representation is exactly a witness that the no-host-leak condition fails. Hence the no-hidden-input claim fails. □

9223.2 Core implementation modules

Definition 9223.8 (Channel core module). Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface.Cha

The channel core module *implements the channel-layer definitions*

$$\zeta(0) = 0, \qquad \zeta(1) = 10, \qquad \mathsf{Ev}(w) = \zeta(w)11.$$

Its public relations are

$$\mathsf{EncodesEvent}(w, c), \quad \mathsf{Compiles}(S, c), \quad \mathsf{IsLegalZStream}(c).$$

A suitable module name is

BEDC.Compiler.Channel.Core.

Executable functions may exist internally, but theorem statements should align with relation form.

Definition 9223.9 (Decoder core module)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Dec`

The decoder core module *implements the scan rules:*

$$0 \mapsto \textit{raw } 0, \qquad 10 \mapsto \textit{raw } 1, \qquad 11 \mapsto \textit{event end}.$$

Its public relations are

$$\mathsf{Decodes}(c, S), \qquad \mathsf{Rejects}(c, r),$$

where the rejection reason r must itself eventually be event-flow data, not a permanent host enum. A suitable module name is

BEDC.Compiler.Channel.Decode.

Definition 9223.10 (Round-trip module)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Roun`

The round-trip module *proves*

$$\mathsf{Decodes}(\mathsf{Compile}(S), S)$$

and, for legal channel streams,

$$c \in \mathsf{LegalZStream} \Rightarrow \mathsf{Compiles}(\mathsf{Decode}(c), c).$$

A suitable module name is

BEDC.Compiler.Channel.RoundTrip.

Definition 9223.11 (Layer-separation module)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface`

The layer-separation module *proves that source-level events are not channel substrings, that the channel terminator* 11 *is not the source event* 11*, and that source normalization acts only after decoding. A suitable module name is*

BEDC.Compiler.LayerSeparation.

Definition 9223.12 (Recognizer core module)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.`

The recognizer core module *supplies the general recognition relation*

$$\mathsf{Recognizes}(R, \rho, S)$$

and recognizer certification relation

$$\mathsf{CertifiedRecognizer}(R, \rho).$$

A suitable module name is

BEDC.Compiler.Recognizer.Core.

Definition 9223.13 (Certificate recognizer modules). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInte`

The certificate and structure recognizer modules expose only event-flow recognition relations. Suitable module names include

BEDC.Compiler.Recognizer.Package, BEDC.Compiler.Recognizer.NameCert,
BEDC.Compiler.Recognizer.DerivCert, BEDC.Compiler.Recognizer.AcceptGate,
BEDC.Compiler.Recognizer.Theorem, BEDC.Compiler.Recognizer.Chapter,
BEDC.Compiler.Recognizer.Compiler.

Definition 9223.14 (Analysis modules). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Analy`

Analysis modules include motif, metric, report, cannot-claim, and bridge-obligation modules. They must maintain the rules that reports are output views, metrics are protocol-relative, motif recognition requires generated recognizers, and analysis is conservative.

9223.3 Channel compiler specification

Definition 9223.15 (Event encoder specification). *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.Enco`

The event encoding specification is the relation

$$\mathsf{EncodesEvent}(w, c)$$

defined by

$$c = \zeta(w)11.$$

Definition 9223.16 (Flow compiler specification). *Lean4 (definition)* `BEDC.GroundCompiler.ChannelEncoding.Comp`

The flow encoding specification is the relation

$$\mathsf{Compiles}(S, c)$$

defined as follows. If

$$S = (w_0, \ldots, w_{n-1}),$$

then

$$c = \mathsf{Ev}(w_0) \cdots \mathsf{Ev}(w_{n-1}).$$

Definition 9223.17 (Event decoder specification). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterf`

The event decoding specification is the relation

$$\mathsf{DecodesEvent}(c, w)$$

defined by

$$\mathsf{EncodesEvent}(w, c).$$

This is a specification-level relation. An implementation may scan the input, but its proof must connect the scan to this relation.

Definition 9223.18 (Stream decoder specification). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInter`

The stream decoding specification is the relation

$$\mathsf{Decodes}(c, S)$$

defined by

$$\mathsf{Compiles}(S, c).$$

Thus the executable decoder must prove soundness and completeness with respect to the specification-level bijection.

Proposition 9223.19 (Specification-level decoder is functional)**.** *Lean4✓ BEDC.GroundCompiler.ImplementationI functional*

If

$$\mathsf{Decodes}(c, S) \qquad \textit{and} \qquad \mathsf{Decodes}(c, T),$$

then $S = T$.

Proof. By the definition of Decodes, both S and T compile to c. The channel compiler is injective on event flows, so $S = T$. □

Proposition 9223.20 (Specification-level compiler is functional)**.** *Lean4✓ BEDC.GroundCompiler.ChannelEncodin functional*

If

$$\mathsf{Compiles}(S, c) \qquad \textit{and} \qquad \mathsf{Compiles}(S, d),$$

then $c = d$.

Proof. By the definition of Compiles, both c and d are the concatenated event encoding of the same event flow S. Hence $c = d$. □

9223.4 Executable encoder and decoder obligations

Definition 9223.21 (Executable decoder)**.** *Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface.Exe*

An executable decoder *is an implementation-layer procedure that consumes a bit stream and returns either an event flow or a rejection reason. The formal public statements do not expose host result types or host rejection enums as BEDC primitives.*

Its obligations are decoder soundness, decoder completeness on the legal-stream domain, and rejection soundness.

Theorem 9223.22 (Decoder soundness obligation)**.** *Lean4✓ BEDC.GroundCompiler.ImplementationInterface.dec soundness obligation*

If the implementation decoder returns success with event flow S on input c, then

$$\mathsf{Decodes}(c, S)$$

must hold.

Proof. The implementation proof must establish scanner invariants: the already-read prefix corresponds to completed events, the current event body lies in $(0 \mid 10)^*$, reading 11 completes an event, a dangling 1 rejects, and input termination leaves no unclosed event. These invariants imply agreement with the specification-level Decodes relation. □

Theorem 9223.23 (Decoder completeness obligation)**.** *Lean4✓ BEDC.GroundCompiler.ImplementationInterface. completeness obligation*

If $c \in \mathsf{LegalZStream}$ *and* $\mathsf{Decodes}(c, S)$*, then the implementation decoder must return success with event flow* S.

Proof. A legal stream lies in $((0 \mid 10)^*11)^*$. The scanner therefore encounters no dangling 1, no unclosed event, and no illegal token. Each event body decodes uniquely to a raw event. Hence the implementation returns the specified S. □

Theorem 9223.24 (Reject soundness obligation). *Lean4✓ BEDC.GroundCompiler.ImplementationInterface.reje soundness obligation*

If the reference executable decoder rejects input c with event-flow reason r, then

$$c \notin \mathsf{LegalZStream}.$$

Any implementation-domain rejection policy must be expressed as its own event-flow relation before it can appear in the public formal interface.

Proof. If $c \in \mathsf{LegalZStream}$, then $c = \mathsf{Compile}(S)$ for some event flow S. By the channel round-trip theorem, decoding c returns success with S. This contradicts rejection by the reference executable decoder. □

Definition 9223.25 (Executable encoder). *Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface.Exe*

An executable encoder *is an implementation-layer procedure from event-flow representation to bit-stream representation. Its formal obligations are encoder soundness, encoder totality over event flows, and absence of internal event terminators inside encoded event bodies.*

Theorem 9223.26 (Encoder soundness obligation). *Lean4✓ BEDC.GroundCompiler.ImplementationInterface.enc soundness obligation*

If the implementation encoder returns c on event flow S, then

$$\mathsf{Compiles}(S, c)$$

must hold.

Proof. The implementation must show that each raw event is encoded by ζ and followed by the event terminator 11, and that event encodings are concatenated in the order of S. □

Theorem 9223.27 (Encoder totality obligation). *Lean4✓ BEDC.GroundCompiler.ImplementationInterface.encod totality obligation*

For every $S \in \mathsf{EventFlow}$, the implementation encoder returns some c such that

$$\mathsf{Compiles}(S, c).$$

Proof. Each raw event is finite, so applying ζ to its finite body and appending 11 produces a finite event code. A finite event flow has finitely many such event codes, so their concatenation exists. □

Theorem 9223.28 (Encoder no-internal-terminator obligation). *Lean4✓ BEDC.GroundCompiler.ImplementationI no internal terminator obligation*

For every raw event w, the event body $\zeta(w)$ contains no adjacent 11.

Proof. Each source 0 contributes channel 0, and each source 1 contributes channel 10. Hence every channel 1 inside the body is followed by 0, never by another 1. Therefore 11 can occur only as the appended terminator. □

9223.5 Public recognition interfaces

Definition 9223.29 (Package recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Implementatio`
The package recognition public interface is

$$\mathsf{RecognizesPkg}(R, S).$$

Host AST package parsers and YAML package parsers are not public formal inputs.

Definition 9223.30 (Name-certificate recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Imple`
The name-certificate recognition public interface is

$$\mathsf{RecognizesNameCert}(R, S, N).$$

Host name-certificate records are not formal inputs.

Definition 9223.31 (Derivation-certificate recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.I`
The derivation-certificate recognition public interface is

$$\mathsf{RecognizesDerivCert}(R, D, N, s).$$

Host derivation-certificate structures and host strength enums are not formal inputs.

Definition 9223.32 (Theorem recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Implementati`
The theorem recognition public interface is

$$\mathsf{RecognizesTheorem}(R, T).$$

Theorem names as strings, Lean theorem identifiers, proof ASTs, and host proof objects are not formal theorem inputs.

Definition 9223.33 (Chapter recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Implementatio`
The chapter recognition public interface is

$$\mathsf{RecognizesChapter}(R, C).$$

Chapter YAML, chapter manifests, Markdown ASTs, and LaTeX section trees are not formal chapter inputs.

Definition 9223.34 (Compiler recognition public interface)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Implementati`
The compiler recognition public interface consists of relations such as

$$\mathsf{RecognizesCompiler}(R, C), \qquad \mathsf{CertifiedCompiler}(C), \qquad \mathsf{Compiles}_C(S, T), \qquad \mathsf{SelfHosting}(C).$$

Host executable paths, binary hashes, and build scripts may be implementation-ledger audit data, but they do not replace compiler event flow.

9223.6 No-host-leak audit

Definition 9223.35 (Host-leak audit item). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Hos`

A host-leak audit item *is an event flow recording the occurrence of host-level input in an implementation statement. It contains at least a location, the host construct, why it is a leak, the replacement relation, status, and ledger.*

Examples include a host list exposed as BEDC event flow, a host natural number exposed as raw length, YAML used as package input, or a string theorem label used as theorem identity.

Definition 9223.36 (No-host-leak audit). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.NoHo`

An implementation module passes no-host-leak audit *when:*

1. *all public statements use only event-flow relations;*
2. *all host structures remain inside implementation proof internals;*
3. *all remaining host dependencies are recorded as bootstrap obligations;*
4. *all reports, YAML views, and JSON views are marked as output views.*

Theorem 9223.37 (No-host-leak audit is required for compiler certification). *Lean4 ✓* `BEDC.GroundCompiler.Impl` `host leak required`

If an implementation module fails the no-host-leak audit, then it cannot be a certified no-hidden-input compiler component.

Proof. Failure of the audit means that host-level structure has entered the public interface, or that a host dependency is unrecorded. This is hidden input. A no-hidden-input compiler component cannot depend on hidden input. □

9223.7 Lean targets and reference executable

Definition 9223.38 (Core Lean target set). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterface.Co`

The first Lean target set should cover only the channel and layer-separation layers:

BitRep, RawEvent, EventFlow, ζ, Ev, Compile,
LegalZStream, Decode, decodeEncodeEvent, decodeEncodeFlow,
encodeDecodeLegal, noInternalTerminator, channelTerminatorUnique, channelSubstringNotSourceEvent.

Definition 9223.39 (Recognizer Lean target set). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterfa`

The second Lean target set covers the recognizer abstraction layer:

RecognizerFlow, CertifiedRecognizer, RecognizesPkg, RecognizesNameCert,
RecognizesDerivCert, RecognizesTheorem, RecognizesChapter, RecognizesCompiler,
recognitionCompileDecodeInvariant, recognitionConservative.

Definition 9223.40 (Certificate Lean target set). *Lean4 (definition)* `BEDC.GroundCompiler.ImplementationInterfa`

The third Lean target set covers complete certificate recognition and acceptance:

NameCertFlow, DerivCertFlow, AcceptGateFlow,
soundNameCertFlowLicensesName, soundDerivCertFlowEstablishesDerivCert,
acceptedFlowRequiresNameCertAndDerivCert, codeExistenceNotAcceptance.

Definition 9223.41 (Analysis Lean target set)**.** Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface

The fourth Lean target set covers motif and metric analysis:

$$\mathsf{MotifProfile},\quad \mathsf{SealDepth},\quad \mathsf{CarryIndex},\quad \mathsf{LedgerDepth},$$
$$\mathsf{MetricAdmissible},\quad \mathsf{metricRoundTripInvariant},\quad \mathsf{channelSubstringMetricInadmissible},$$
$$\mathsf{motifSimilarityNotIdentity},\quad \mathsf{metricsConservative}.$$

Remark 9223.42 (Do not formalize high-level objects first)**.** *The first implementation targets should not be definitions of infinity, dimension, real numbers, or circles as primitives. They should be event flows, legal channel streams, recognizer flows, name-certificate flows, derivation-certificate flows, acceptance-gate flows, and motif profiles. Mature objects enter only by certificate-mediated export.*

Definition 9223.43 (Reference executable)**.** Lean4 (definition) BEDC.GroundCompiler.ImplementationInterface.Re

A reference executable *is a runnable program satisfying the channel encoder and decoder specifications. Its tasks are to encode event flows, decode channel streams, check legal streams, and generate human-readable reports from decoded flows. It is not a proof checker or certificate recognizer unless the corresponding recognizer flows and certificates are supplied.*

Proposition 9223.44 (Reference executable alone is not a formal compiler)**.** Lean4 ✓ BEDC.GroundCompiler.Imple
executable not formal compiler

Even if a reference executable passes all encode/decode tests, it is not a full no-hidden-input compiler unless the recognizer hierarchy, certificate flows, and bootstrap obligations are also handled.

Proof. Encode/decode tests establish channel behavior. A full compiler also needs package, certificate, theorem, chapter, and compiler recognition, plus self-hosting or an explicit bootstrap boundary. A reference executable alone does not provide those structures. □

Definition 9223.45 (Reference executable test suite)**.** Lean4 (definition) BEDC.GroundCompiler.ImplementationInt

The minimal reference test suite contains event encoding cases

$$\begin{aligned}
\epsilon &\mapsto 11,\\
0 &\mapsto 011,\\
1 &\mapsto 1011,\\
00 &\mapsto 0011,\\
01 &\mapsto 01011,\\
011 &\mapsto 0101011,\\
100 &\mapsto 100011,
\end{aligned}$$

decoding cases $011 \mapsto 0$ *and* $1011 \mapsto 1$*, rejection of* 0111*, and round trips for*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

and

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

Example 9223.46 (Completion skeleton round trip)**.** *The event flow*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

has expected encoded stream

$$011001100011010110101011100011.$$

Decoding that stream should return the original event flow.

Example 9223.47 (Add/fold skeleton round trip)**.** *The event flow*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100$$

has expected encoded stream

$$011001100101100101011010 0011.$$

Decoding that stream should return the original event flow.

9223.8 Implementation reports and adequacy

Definition 9223.48 (Implementation report)**.** Lean4 (definition) `BEDC.GroundCompiler.ImplementationInterface..`

An implementation report *is an output view generated by a reference executable or formal module. It records module name, implemented relations, test cases, round-trip results, rejection cases, host-leak audit items, bootstrap obligations, proof status, and Lean target status.*

An implementation report is output, not formal input.

Principle 9223.49 (Implementation reports are not certificates)**.** Lean4 ✓ `BEDC.GroundCompiler.Implementationl` `reports not certificates`

An implementation report is not a name-certificate flow, derivation-certificate flow, acceptance-gate flow, or theorem proof-flow. Unless the report flow is separately recognized as a certificate flow, it cannot discharge certificate obligations.

Theorem 9223.50 (Implementation adequacy theorem)**.** Lean4 ✓ `BEDC.GroundCompiler.ImplementationAdequacy` `adequacy`

An implementation I may serve as an implementation of a no-hidden-input compiler component exactly when:

1. *every public relation implemented by I has a formal specification;*
2. *I is sound for those specifications;*
3. *I is complete on the specified domains;*
4. *the public interface has no host leak;*
5. *every host dependency is proof-internal or recorded as a bootstrap obligation;*
6. *every report is marked as an output view;*
7. *every claimed recognizer, certificate, theorem, or chapter function is supported by the corresponding recognizer flow and certificate flow;*
8. *every claimed full-compiler function has self-hosting or an explicitly recorded bootstrap boundary.*

Proof. Necessity: if any condition fails, then there is unspecified behavior, unsoundness, incompleteness, host leak, hidden dependency, report-as-input confusion, uncertified recognition, or hidden compiler input. Each violates the no-hidden-input discipline.

Sufficiency: if all eight conditions hold, every formal output is supported by event-flow relations and recognized certificates; host artifacts remain proof-internal or are recorded as bootstrap obligations; reports are not inputs; recognition claims have certificates; and compiler claims have self-hosting or a declared boundary. Thus I may serve as a no-hidden-input compiler component. □

Corollary 9223.51 (Passing tests is weaker than implementation adequacy)**.** `Lean4✓ BEDC.GroundCompiler.Impl`
`tests weaker than adequacy`

Passing reference executable tests is weaker than implementation adequacy. Tests can support sanity checks and partial soundness evidence, but they cannot replace formal specifications, no-host-leak audit, recognizer certificates, or self-hosting ledger.

Remark 9223.52 (Summary)**.** *This chapter separates implementation representation from public formal interface. Implementations may use host data structures internally, but the public BEDC surface must expose event-flow relations. Channel encoder and decoder implementations require soundness, completeness, rejection, and no-internal-terminator obligations. Recognizers expose relation-form interfaces, not host parsers. No-host-leak audit is required for certification. Reference executables are useful channel tools, but implementation adequacy additionally requires formal specifications, no-host-leak status, bootstrap accounting, recognizer certificates, and self-hosting or a declared boundary.*

Chapter 9224

Minimal Executable Prototype Specification

chapter 9223 fixes the implementation interface: host data structures may be used internally, but the public formal interface must not leak host structure. This chapter specifies the minimal executable prototype. Its purpose is not to recognize all mathematical structures, prove theorems, export mature objects, or replace name certificates, derivation certificates, or acceptance gates. It implements only the lossless streaming channel:

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}.$$

The prototype encodes event flows, decodes legal channel streams, rejects illegal channel streams, emits human-readable reports, runs round-trip tests, and records rejection, warning, and cannot-claim information.

9224.1 Scope and display representations

Definition 9224.1 (Reference prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.ReferencePr`

A reference prototype *is an executable implementation P_0 whose formal capability is limited to*

$$\mathsf{Compile} : \mathsf{EventFlow} \to \mathsf{LegalZStream}, \qquad \mathsf{Decode} : \mathsf{LegalZStream} \to \mathsf{EventFlow},$$

together with illegal-input rejection. Write

$$\mathsf{ReferencePrototype}(P_0).$$

It is a channel-level prototype, not a full compiler.

Principle 9224.2 (Prototype is not full compiler)**.** *Lean4 ✓* `BEDC.GroundCompiler.MinimalPrototype.reference prototype not full compiler`

The reference prototype P_0 must not claim package recognition, name-certificate recognition, derivation-certificate recognition, acceptance-gate recognition, theorem recognition, chapter recognition, compiler self-hosting, motif analysis, metric analysis, or bridge discovery unless these functions are provided at a later prototype level with the required recognizer flows, certificate flows, or bootstrap obligations.

Definition 9224.3 (Prototype level)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.PrototypeLevel`

Prototype levels are ordered as follows:

P_0 *channel prototype;*
P_1 *source-event reporting prototype;*
P_2 *source-level normalizer-ledger prototype;*
P_3 *motif-report prototype;*
P_4 *recognizer-flow prototype;*
P_5 *certificate-flow prototype;*
P_6 *theorem/chapter-flow prototype;*
P_7 *self-hosting prototype.*

This chapter specifies P_0 and optional P_1 output views.

Definition 9224.4 (Prototype input representations)**.** Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.`

At implementation level, P_0 may accept two display representations:

1. *an event-flow display, such as*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

which displays the event flow $(0, 00, 000, 01, 011, 100)$*;*

2. *a channel-stream display, such as*

$$011001100011010110101011100011,$$

which displays a candidate channel stream.

These displays are implementation representations, not BEDC formal objects.

Remark 9224.5 (Display separator is not formal structure)**.** *The symbol* $\mid$ *belongs to display notation for human-readable event boundaries. It is not a raw mark, and it does not occur in formal channel code. Thus*

$$0 \mid 00 \mid 000$$

is only a display of the event flow $(0, 00, 000)$*.*

Definition 9224.6 (Prototype encoder)**.** Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.PrototypeEnc`

The prototype encoder *is an implementation function from event-flow display to channel-stream display. If the input display parses as event flow S and the output display is c, then it must implement the formal relation*

$$\mathsf{Compiles}(S, c),$$

equivalently $c = \mathsf{Compile}(S)$*.*

Definition 9224.7 (Prototype decoder)**.** Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.PrototypeDec`

The prototype decoder *is an implementation function from channel-stream display to either event-flow display or a rejection report. If it returns success with event flow S on channel stream c, then it must implement*

$$\mathsf{Decodes}(c, S).$$

If it rejects, it must return a rejection report.

Definition 9224.8 (Reject report)**.** Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.RejectReport`

A reject report *is an implementation-level output view recording why a channel stream is not accepted by the prototype. Typical rejection cases include dangling* 1*, unfinished event, non-binary character, empty-input policy violation, resource-bound excess, and noncanonical display.*

The case "dangling 1*" means that the input ends with an isolated* 1 *that cannot form* 10 *or* 11*. The case "unfinished event" means that an event body has not reached a terminator* 11*. The case "non-binary character" means that the display contains a character other than* 0 *or* 1*.*

Principle 9224.9 (Reject reports are output)**.** *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.reject reports output not formal input*

Reject reports are output views. They are not formal proofs, name certificates, derivation certificates, theorems, chapters, or accepted objects.

9224.2 Encoding and decoding specification

Definition 9224.10 (Prototype event encoder)**.** *Lean4 (definition) BEDC.GroundCompiler.ChannelEncoding.Prototy*

For a raw event $w = w_1 \cdots w_n$, the prototype event encoder outputs

$$\mathsf{Ev}(w) = \zeta(w)11,$$

where

$$\zeta(0) = 0, \qquad \zeta(1) = 10.$$

In implementation terms, each displayed 0 *contributes* 0*, each displayed* 1 *contributes* 10*, and the event terminator* 11 *is appended at the end.*

Definition 9224.11 (Prototype flow encoder)**.** *Lean4 (definition) BEDC.GroundCompiler.ChannelEncoding.Prototyp*

For an event flow

$$S = (w_0, \ldots, w_{n-1}),$$

the prototype outputs

$$\mathsf{Compile}(S) = \mathsf{Ev}(w_0) \cdots \mathsf{Ev}(w_{n-1}).$$

Example 9224.12 (Completion skeleton encoding)**.** *The display*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

has event encodings

$$\begin{aligned} 0 &\mapsto 011, \\ 00 &\mapsto 0011, \\ 000 &\mapsto 00011, \\ 01 &\mapsto 01011, \\ 011 &\mapsto 0101011, \\ 100 &\mapsto 100011. \end{aligned}$$

The compiled output is

$$011001100011010110101011100011.$$

Example 9224.13 (Add/fold skeleton encoding)**.** *The display*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100$$

has event encodings

$$\begin{aligned} 0 &\mapsto 011, \\ 00 &\mapsto 0011, \\ 001 &\mapsto 001011, \\ 0011 &\mapsto 00101011, \\ 0100 &\mapsto 0100011. \end{aligned}$$

The compiled output is

$$011001100101100101011010 0011.$$

Definition 9224.14 (Prototype stream decoder). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.Proto`

The prototype decoder scans channel bits left to right using:

$$0 \mapsto \textit{append raw } 0, \qquad 10 \mapsto \textit{append raw } 1, \qquad 11 \mapsto \textit{close current event.}$$

It rejects dangling 1*, unclosed event body under the chosen display policy, and non-binary characters.*

Definition 9224.15 (Decoder state). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.DecoderState`

An implementation may use host-level decoder state containing current event bits, completed events, position, and rejection reason. These are implementation internals. Formal behavior is captured by the source-level relation $\mathsf{Decodes}(c, S)$*.*

Example 9224.16 (Decode completion skeleton code). *The channel stream*

$$011001100011010110101011100011$$

segments as

$$011, \quad 0011, \quad 00011, \quad 01011, \quad 0101011, \quad 100011$$

and decodes to

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Example 9224.17 (Rejecting 0111). *The input* 0111 *scans as* 0 *followed by* 11*, closing the first event as raw* 0*, and then leaves a dangling* 1*. The prototype rejects with a dangling-*1 *report.*

Example 9224.18 (Decoding raw source event 11). *The source event* 11 *encodes as*

$$101011.$$

The decoder reads 10 *as raw* 1*, then* 10 *as raw* 1*, then* 11 *as event terminator. The output is the source event* 11*. Thus channel terminator* 11 *is not the source raw event* 11*.*

9224.3 Prototype test suite

Definition 9224.19 (Required unit tests). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.RequiredUni`

The reference prototype must pass the following event-level encoding tests:

$$\begin{array}{rcl}
\epsilon & \mapsto & 11, \\
0 & \mapsto & 011, \\
1 & \mapsto & 1011, \\
00 & \mapsto & 0011, \\
01 & \mapsto & 01011, \\
10 & \mapsto & 10011, \\
11 & \mapsto & 101011, \\
011 & \mapsto & 0101011, \\
100 & \mapsto & 100011, \\
0011 & \mapsto & 00101011.
\end{array}$$

Definition 9224.20 (Required round-trip tests). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.Requi`

The reference prototype must pass event round trips for

$$\epsilon, \quad 0, \quad 1, \quad 00, \quad 01, \quad 10, \quad 11, \quad 011, \quad 100, \quad 001, \quad 0011, \quad 0100,$$

and flow round trips for

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100,$$

$$011 \mid 100, \qquad 1 \mid 11 \mid 011 \mid 100.$$

Definition 9224.21 (Required reject tests). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.RequiredR`

The reference prototype must reject, or explicitly classify by implementation policy, the displays

$$1, \qquad 0111, \qquad 010, \qquad 101, \qquad 000.$$

Here 1 *is dangling,* 0111 *contains a valid first event followed by dangling* 1*,* 010 *decodes an event body but lacks terminator,* 101 *decodes raw* 1 *followed by dangling* 1*, and* 000 *is an event body without a terminator.*

Remark 9224.22 (Empty stream policy). *The empty channel stream may be accepted as the empty event flow or rejected by implementation policy when nonempty input is required. The formal channel language allows empty concatenation:*

$$\epsilon \in ((0 \mid 10)^* 11)^*.$$

Rejecting the empty stream is therefore implementation-domain policy, not a channel-language theorem.

9224.4 Prototype outputs

Definition 9224.23 (Decode report). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.DecodeReport`

A decode report *is human-readable output containing the input channel stream, legality status, decoded event flow when legal, event segmentation, rejection reason when illegal, and warnings. It is an output view, not a formal proof.*

Definition 9224.24 (Encode report). *Lean4 (definition)* `BEDC.GroundCompiler.MinimalPrototype.EncodeReport`

An encode report *is human-readable output containing the input event-flow display, per-event encoding, final channel stream, round-trip check result, and warnings. It is an output view, not a formal proof.*

Example 9224.25 (Encode report for completion skeleton). *For input*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

an encode report may record the per-event codes 011*,* 0011*,* 00011*,* 01011*,* 0101011*, and* 100011*, the compiled stream*

$$011001100011010110101011100011,$$

and a successful round trip.

Example 9224.26 (Decode report for carry-looking pair)**.** *The input*

$$0101011100011$$

segments as

$$0101011, \quad 100011$$

and decodes to

$$011 \mid 100.$$

The report must not assert $011 = 100$*. If this pair is read as a carry, a source-level carry recognizer and ledger are required.*

Principle 9224.27 (Reports must include warnings)**.** *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.reports must include warnings*

When reports contain carry-looking pairs such as $011 \mid 100$ *or* $0011 \mid 0100$*, they must warn that these are raw events, not equality claims. Any carry interpretation requires source-level recognizer and ledger.*

9224.5 Prototype correctness

Theorem 9224.28 (Prototype encoder soundness)**.** *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.prototype encoder soundness*

If the reference prototype encodes event flow S *and returns channel stream* c*, then*

$$\mathsf{Compiles}(S, c).$$

Proof. By specification, the encoder applies Ev to each event and concatenates the results. This is exactly the definition of $\mathsf{Compile}(S)$. □

Theorem 9224.29 (Prototype decoder soundness)**.** *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.prototype decoder soundness*

If the reference prototype decodes channel stream c *and returns event flow* S*, then*

$$\mathsf{Decodes}(c, S).$$

Proof. The decoder scans c by the grammar 0 for raw 0, 10 for raw 1, and 11 for event end. If it returns S, then c has been parsed as a concatenation of legal event codes for the events of S. Hence $c = \mathsf{Compile}(S)$, so $\mathsf{Decodes}(c, S)$. □

Theorem 9224.30 (Prototype decoder completeness on legal streams)**.** *Lean4✓ BEDC.GroundCompiler.MinimalPr decoder completeness on legal streams*

If $c \in \mathsf{LegalZStream}$*, then the reference decoder returns some event flow* S *such that*

$$\mathsf{Decodes}(c, S).$$

Proof. A legal stream lies in $((0 \mid 10)^*11)^*$, hence is a concatenation of legal event codes. Event bodies contain no internal 11, so the first 11 after each body uniquely terminates an event. The scanner returns the corresponding event flow. □

Theorem 9224.31 (Prototype reject soundness). *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.prototype reject soundness*

If the reference decoder rejects c with a syntactic reject reason, formally

$$\mathsf{PrototypeSyntacticReject}(c, r),$$

then

$$c \notin \mathsf{LegalZStream},$$

where implementation-domain policy reasons are not syntactic reject reasons.

Proof. Syntactic rejection records violation of the grammar $((0 \mid 10)^*11)^*$: dangling 1, unclosed event body, or non-binary display character. Such failures cannot occur in a legal stream. Policy rejections, such as empty-input disallowance or resource bounds, must be recorded separately and do not assert syntactic illegality. □

Corollary 9224.32 (Prototype round-trip correctness). *Lean4✓ BEDC.GroundCompiler.ChannelEncoding.prototy roundtrip correctness*

For every event flow S, decoding the prototype encoding of S returns S. For every legal channel stream c, encoding the prototype decoding of c returns c.

Proof. This follows from encoder soundness, decoder soundness, decoder completeness on legal streams, and the channel bijection theorem. □

9224.6 Prototype limitations and CLI

Theorem 9224.33 (Reference prototype does not recognize mathematical structures). *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.reference prototype no math structure recognition*

The reference prototype P_0 does not recognize package, name-certificate, derivation-certificate, theorem, chapter, motif, metric, or accepted-object structures.

Proof. By definition, P_0 implements only channel encoding, decoding, and legality checking. It has no generated recognizer flow for any higher structure. Therefore it cannot formally recognize those structures. □

Corollary 9224.34 (Prototype output is not a name certificate). *Lean4✓ BEDC.GroundCompiler.MinimalPrototyp output not namecert*

Even if a decoded event flow visually resembles a name-certificate flow, P_0 alone cannot recognize it as one. A name-certificate recognizer flow is required.

Corollary 9224.35 (Prototype output is not a theorem). *Lean4✓ BEDC.GroundCompiler.MinimalPrototype.proto output not theorem*

Even if a decoded event flow visually resembles a theorem flow, P_0 alone cannot recognize it as one. A theorem recognizer flow and proof-checker flow are required.

Corollary 9224.36 (Prototype output is not an accepted object). *Lean4✓ BEDC.GroundCompiler.MinimalPrototyp output not accepted*

Even if a decoded event flow contains apparent certificate-like segments, P_0 alone cannot establish an acceptance gate.

Definition 9224.37 (Allowed CLI commands). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.Allowe`

A reference implementation may provide commands named encode, decode, check, roundtrip, and report. Their meanings are implementation conveniences:

encode	*event-flow display to channel stream;*
decode	*channel stream to event-flow display or reject report;*
check	*channel stream to legal or illegal status;*
roundtrip	*event-flow display through encode and decode;*
report	*human-readable encode/decode report.*

These commands are not BEDC formal operators.

Definition 9224.38 (Forbidden CLI meanings at P_0). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.F`

At P_0, the prototype must not provide commands implying higher recognition: name-certificate recognition, theorem recognition, object acceptance, proof, bridge, dimension inference, real-object inference, circle-object inference, or channel-level carry normalization. If such commands exist as stubs, they must report that they require higher recognizer flows.

9224.7 Output formats and audit

Definition 9224.39 (Event-flow display format). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.Event`

An event-flow display may use 0, 1, |, *and whitespace, with* | *separating raw events and whitespace ignored. Empty events are supported only by explicit display syntax such as ϵ, which is display-only and compiles as the empty raw event.*

Definition 9224.40 (Channel-stream display format). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.`

A channel-stream display uses only 0 *and* 1, *with no separators. Successful decoding requires legal channel syntax.*

Definition 9224.41 (Report format). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.PrototypeReport`

A report may contain input, output, events, event codes, rejection reason, round-trip status, and warnings. It must not be accepted as formal input.

Principle 9224.42 (Reports are output views). Lean4 ✓ `BEDC.GroundCompiler.MinimalPrototype.prototype reports output not formal input`

Reports may be exported as text, Markdown, JSON, YAML, or HTML, but all are output views. None is a formal compiler input.

Definition 9224.43 (Prototype audit checklist). Lean4 (definition) `BEDC.GroundCompiler.MinimalPrototype.Proto`

A P_0 prototype must satisfy:

1. *event encoding implements $\zeta(w)11$;*
2. *flow encoding concatenates event encodings;*
3. *stream decoding implements the* 0/10/11 *scanner;*
4. *dangling* 1 *is rejected;*
5. *unclosed event body is rejected;*
6. *non-binary characters are rejected;*
7. *required unit tests pass;*

8. *required round-trip tests pass;*
9. *carry-looking pairs produce warnings;*
10. *no structure recognition is claimed;*
11. *YAML, AST, and package structures are not accepted as input;*
12. *no theorem, object, or proof claim is exposed.*

Theorem 9224.44 (Prototype audit suffices for P_0 adequacy)**.** `Lean4✓` `BEDC.GroundCompiler.MinimalPrototype.` `audit suffices`

If a reference prototype satisfies the audit checklist and its encoder and decoder satisfy soundness and completeness obligations, then it is adequate as a P_0 channel prototype.

Proof. The checklist covers exactly the duties of P_0: channel encoding, decoding, legality checking, round-trip testing, rejection reporting, and non-claim discipline. Soundness and completeness establish correctness against the formal channel relations. Therefore the prototype is adequate at level P_0. □

Corollary 9224.45 (P_0 adequacy does not imply higher adequacy)**.** `Lean4✓` `BEDC.GroundCompiler.MinimalProtot` `adequacy not higher`

Adequacy at P_0 does not imply adequacy for package recognition, theorem recognition, certificate recognition, motif analysis, or self-hosting.

9224.8 Example sessions and conservativity

Example 9224.46 (Encoding completion skeleton)**.** *Encoding the display*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

returns

$$011001100011010110101011100011.$$

The report may say that round trip succeeds and that the occurrence of 011 *followed by* 100 *may be read as carry only after a source-level carry recognizer is supplied.*

Example 9224.47 (Decoding completion skeleton)**.** *Decoding*

$$011001100011010110101011100011$$

returns

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

with legal status and six events.

Example 9224.48 (Rejecting invalid stream)**.** *Decoding* 0111 *rejects with dangling* 1*: the prefix* 011 *decodes to raw event* 0*, and the remaining* 1 *is not a legal token.*

Example 9224.49 (Encoding raw source event 11)**.** *Encoding the single raw source event* 11 *returns*

$$101011.$$

The report must warn that source raw event 11 *is not channel terminator* 11*.*

Theorem 9224.50 (Reference prototype is conservative over the finite kernel)**.** Lean4✓ BEDC.GroundCompiler.Min
prototype conservative over kernel

Implementing P_0 adds no raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, or accepted-object export rules.

Proof. P_0 implements only channel encoding and decoding of event flows. It does not introduce source events beyond the existing display alphabet, assert msame, hsame, or psame, alter Ext or Cont, recognize name certificates or acceptance gates, or license mature objects. Therefore it is conservative over the finite kernel. □

Corollary 9224.51 (Reference prototype is an address layer)**.** Lean4✓ BEDC.GroundCompiler.MinimalPrototype.r
prototype address layer

P_0 provides a lossless address layer for event flows. It is not a mathematical object generator.

Remark 9224.52 (Summary)**.** *This chapter specifies the minimal executable prototype P_0. It implements event and flow encoding, channel decoding, legal-stream checking, round trips, and reports. It must satisfy encoder soundness, encoder totality, decoder soundness, decoder completeness, rejection soundness, warning discipline, and the P_0 audit checklist. It must not claim package recognition, certificate recognition, theorem recognition, object export, motif analysis, metric analysis, bridge discovery, or compiler self-hosting. Thus P_0 is a channel-level reference prototype, not a full compiler.*

Chapter 9225

Source-Event Reporting Prototype

chapter 9224 defines the minimal executable prototype P_0, which encodes event flows, decodes channel streams, checks legality, rejects illegal inputs, and runs round-trip tests. This chapter defines the next prototype level:

$$\boxed{P_1 = P_0 + \text{decoded source-event reports.}}$$

The P_1 layer is a source-event reporting prototype. It does not recognize packages, name certificates, derivation certificates, theorems, chapters, or accepted objects. It decodes a legal channel stream into an event flow and emits a structured human-readable report.

The guiding rule is:

$$\boxed{\text{a report is an output view, not an input, not a recognizer, and not a proof.}}$$

9225.1 Source-event reports

Definition 9225.1 (Source-event reporting prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.Sou`

A source-event reporting prototype *is an executable prototype P_1 satisfying:*

1. *P_1 includes the encode, decode, check, and round-trip functions of P_0;*
2. *if the input is a legal channel stream c, then P_1 decodes*
$$S = \mathsf{Decode}(c);$$
3. *P_1 emits a source-event report $\mathsf{Report}_{\mathrm{src}}(c, S)$;*
4. *the report describes source events, channel segments, round-trip status, warnings, and cannot-claim annotations;*
5. *the report does not recognize packages, name certificates, theorems, chapters, or accepted objects.*

Write

$$\mathsf{SourceReportPrototype}(P_1).$$

Principle 9225.2 (Report is output, not input)**.** *Lean4 ✓* `BEDC.GroundCompiler.SourceReport.source report output not input`

Reports generated by P_1 are output views. They cannot be used as formal compiler input structures. In particular, there is no formal inference step from report to package, name certificate, theorem, or accepted object. The allowed direction is:

$$\textit{channel stream} \longrightarrow \mathsf{Decode} \longrightarrow \textit{event flow} \longrightarrow \textit{source-event report.}$$

Definition 9225.3 (Source-event report). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.SourceEventRepo`

Given a legal channel stream

$$c \in \mathsf{LegalZStream}$$

and decoded event flow

$$S = \mathsf{Decode}(c) = (w_0, \ldots, w_{n-1}),$$

a source-event report *is an output view containing at least the input channel stream, legality status, decoded event count, event index, raw event, channel segment for each event, body segment for each event, terminator segment for each event, round-trip check, source/channel separation warning, and cannot-claim annotations.*

Write

$$\mathsf{SrcReport}(c, S).$$

Definition 9225.4 (Event segment). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.EventSegment`

Let $S = (w_0, \ldots, w_{n-1})$ *and* $c = \mathsf{Compile}(S)$. *The* event segment *of the* i*-th event is*

$$\mathsf{Seg}_i(c, S) := \mathsf{Ev}(w_i).$$

This is a channel-level substring corresponding to a complete event encoding, not an arbitrary channel substring.

Definition 9225.5 (Body and terminator segment). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.BodyTe`

For an event w_i*, its event segment is*

$$\mathsf{Seg}_i = \zeta(w_i)11.$$

Define

$$\mathsf{Body}_i := \zeta(w_i), \qquad \mathsf{Term}_i := 11.$$

Then

$$\mathsf{Seg}_i = \mathsf{Body}_i \mathsf{Term}_i.$$

Proposition 9225.6 (Event segments partition the channel stream). *Lean4✓* `BEDC.GroundCompiler.SourceReport segments partition`

If

$$c = \mathsf{Compile}(w_0, \ldots, w_{n-1}),$$

then

$$c = \mathsf{Seg}_0 \mathsf{Seg}_1 \cdots \mathsf{Seg}_{n-1}.$$

Proof. By the flow compiler definition,

$$\mathsf{Compile}(w_0, \ldots, w_{n-1}) = \mathsf{Ev}(w_0) \cdots \mathsf{Ev}(w_{n-1}).$$

By definition, $\mathsf{Seg}_i = \mathsf{Ev}(w_i)$. Substitution gives the displayed partition. □

Corollary 9225.7 (Reports may display event segmentation). *Lean4✓* `BEDC.GroundCompiler.SourceReport.sourc report may display segments`

Since event segments partition the channel stream, P_1 *may display segment* 0 *through segment* $n-1$ *without introducing an external tree or AST. The segmentation is induced by channel decoding.*

Definition 9225.8 (Event index). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.EventIndex`

For a decoded event flow $S = (w_0, \ldots, w_{n-1})$, the event index *of w_i is i. In a report, event index is implementation-level display. It is not a BEDC source event and not part of the formal event flow.*

Remark 9225.9 (Event index is not source structure). *The index i helps a human inspect the decoded flow. It is not a raw event, theorem label, package label, or object identity. A report may print $w_i = 011$, but the formal source event is just 011.*

9225.2 Warnings and cannot-claims

Definition 9225.10 (Literal source warning). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.LiteralSourc`

A literal source warning *is a report entry generated from raw source events without claiming motif recognition. Examples include: an event 011 followed by 100, an event containing adjacent source 11, an event resembling a raw completion witness, or an event resembling a normal-address candidate.*

Every literal warning must state that it is candidate-only and requires recognizer flow for interpretation.

Principle 9225.11 (Literal warning is not motif recognition). *Lean4 ✓* `BEDC.GroundCompiler.SourceReport.liter` `warning not recognition`

A literal source warning is not a motif recognition judgment. Thus a warning that 011 is followed by 100 does not imply a Zeckendorf-carry relation and does not imply a classifier equivalence. It only states that the decoded event flow contains the literal adjacent events $011 \mid 100$.

Definition 9225.12 (Cannot-claim annotation). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.CannotCla`

A cannot-claim annotation *in a source-event report records what the report does not establish. Required examples include:*

- *decoded event flow is not theoremhood;*
- *decoded event flow is not a name certificate;*
- *decoded event flow is not a derivation certificate;*
- *decoded event flow is not an acceptance gate;*
- *literal $011 \mid 100$ is not carry without recognizer;*
- *channel terminator 11 is not source event 11;*
- *report is not proof;*
- *report is not package recognition.*

Principle 9225.13 (Nontrivial reports include cannot-claims). *Lean4 ✓* `BEDC.GroundCompiler.SourceReport.sour` `report nontrivial cannot claims`

If a report displays events commonly used as mathematical witnesses, such as 01, 011, 100, 001, 0011, or 0100, then it must include cannot-claim annotations preventing overinterpretation.

Definition 9225.14 (Source report for illegal input). *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.Illega`

If channel input c is illegal, P_1 does not produce a source-event report. The formal core of an illegal-input report is:

$$c \notin \mathsf{LegalZStream} \quad \textit{and} \quad \forall S \in \mathsf{EventFlow},\ \neg\, \mathsf{SrcReport}(c, S).$$

The rendered rejection report may contain the input, failure position, rejection reason, optionally a partial decoded prefix if policy allows it, and a cannot-claim annotation saying that an illegal channel stream has no decoded event flow.

Proposition 9225.15 (Illegal channel stream has no source-event report). Lean4 ✓ BEDC.GroundCompiler.SourceR channel no source report

If $c \notin \mathsf{LegalZStream}$, then P_1 cannot output $\mathsf{SrcReport}(c, S)$ for any S.

Proof. A source-event report requires $S = \mathsf{Decode}(c)$. But Decode is defined on legal channel streams. If c is not legal, no decoded event flow S is available. □

Definition 9225.16 (Round-trip report). Lean4 (definition) BEDC.GroundCompiler.SourceReport.RoundTripReport

A round-trip report *for input event flow S records S, $\mathsf{Compile}(S)$, $\mathsf{Decode}(\mathsf{Compile}(S))$, and the comparison result. It succeeds when*

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S.$$

For input channel stream c, it records c, $\mathsf{Decode}(c)$, $\mathsf{Compile}(\mathsf{Decode}(c))$, and the comparison result. It succeeds when

$$\mathsf{Compile}(\mathsf{Decode}(c)) = c.$$

Proposition 9225.17 (Round-trip report is determined by channel bijection). Lean4 ✓ BEDC.GroundCompiler.Sou report determined

If $S \in \mathsf{EventFlow}$, then its round-trip report succeeds. If $c \in \mathsf{LegalZStream}$, then its round-trip report succeeds.

Proof. This is exactly the channel bijection theorem:

$$\mathsf{Decode}(\mathsf{Compile}(S)) = S, \qquad \mathsf{Compile}(\mathsf{Decode}(c)) = c.$$

□

9225.3 Report format and examples

Definition 9225.18 (Minimal source-event report format). Lean4 (definition) BEDC.GroundCompiler.SourceReport.

A minimal report contains input channel stream, legal status, decoded event-flow display when legal, event index, raw event, body code, terminator code, full event code, round-trip status, literal source warnings, and cannot-claim annotations.

Example 9225.19 (Report for completion skeleton). *For input*

$$011001100011010110101011100011,$$

the decoded event flow is

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

The event table records:

index	*raw*	*body*	*terminator*	*code*
0	0	0	11	011
1	00	00	11	0011
2	000	000	11	00011
3	01	010	11	01011
4	011	01010	11	0101011
5	100	1000	11	100011.

Warnings include that the literal pair 011 | 100 *appears, but this is not carry recognition and requires source-level carry recognizer flow. Cannot-claims include that the decoded flow does not license* $\mathsf{FullAxis}^{\uparrow}$*, Completion, or dimension-lift structure, that* 011 *is not equal to* 100*, that channel terminator* 11 *is not source event* 11*, and that the report is not proof.*

Example 9225.20 (Report for Add/fold skeleton)**.** *For input*

$$011001100101100101011010 0011,$$

the decoded event flow is

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

The event table records raw events 0*,* 00*,* 001*,* 0011*, and* 0100 *with codes* 011*,* 0011*,* 001011*,* 00101011*, and* 0100011*. The report may warn that the literal pair* 0011 | 0100 *appears, but this is not fold recognition without recognizer flow. Cannot-claims include that* 001 *is not* $\mathsf{Add}^{\uparrow}$*,* 0011 *is not* $\mathsf{Mul}^{\uparrow}$*,* 0100 *is not an accepted fold object, and the report supplies no name-certificate or derivation-certificate flow.*

Example 9225.21 (Report for raw source event 11)**.** *For input*

$$101011,$$

the decoded event flow is the single source event 11*. The event table records raw* 11*, body* 1010*, terminator* 11*, and code* 101011*. Warnings state that source raw event* 11 *is represented in channel as* 101011*, that the final* 11 *is the channel terminator, and that the source* 11 *is encoded as* 10 10*. Cannot-claims state that channel terminator is not source* 11*, source* 11 *is not automatically a seal, and source* 11 *is not automatically completion.*

9225.4 Soundness and display policy

Definition 9225.22 (Sound source report)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceReport.SoundSourceRep`

A source-event report $\mathsf{SrcReport}(c, S)$ *is* sound *when:*

1. $c \in \mathsf{LegalZStream}$*;*
2. $\mathsf{Decode}(c) = S$*;*
3. *every event-table row corresponds to an actual event* w_i *of* S*;*
4. *every displayed event code equals* $\mathsf{Ev}(w_i)$*;*
5. *every warning is marked candidate-only unless backed by recognizer flow;*
6. *every cannot-claim annotation is consistent with the limited capability of* P_1*.*

Theorem 9225.23 (Sound report theorem)**.** *Lean4 ✓* `BEDC.GroundCompiler.SourceReport.source report sound report`

If P_1 *produces a sound source-event report* $\mathsf{SrcReport}(c, S)$*, then*

$$\mathsf{Decodes}(c, S).$$

Proof. By soundness of the report, $c \in \mathsf{LegalZStream}$ and $\mathsf{Decode}(c) = S$. Hence $\mathsf{Decodes}(c, S)$. □

Corollary 9225.24 (Source report preserves raw event flow). Lean4 ✓ BEDC.GroundCompiler.SourceReport.source report preserves raw flow

A sound source-event report records exactly the decoded source event flow. It does not add or remove source events.

Theorem 9225.25 (Source report does not recognize structures). Lean4 ✓ BEDC.GroundCompiler.SourceReport.so report does not recognize structures

A sound P_1 source-event report does not establish package, name-certificate, derivation-certificate, theorem, chapter, motif, or accepted-object recognition.

Proof. The P_1 layer decodes channel streams and reports source events. It has no generated recognizer flow for package, certificate, theorem, chapter, motif, or accepted-object roles. Therefore it cannot establish those recognitions. □

Corollary 9225.26 (Warnings are weaker than recognitions). Lean4 ✓ BEDC.GroundCompiler.SourceReport.warni weaker than recognitions

A warning such as "literal pair $011 \mid 100$ *appears" is not carry motif recognition. It only reports a literal pattern in decoded source events.*

Theorem 9225.27 (Cannot-claims prevent overinterpretation). Lean4 ✓ BEDC.GroundCompiler.SourceReport.can claims prevent overinterpretation

If a P_1 source report contains raw events commonly used as higher-layer witnesses, then cannot-claim annotations are required to prevent the report from being read as proof or certificate.

Proof. At P_1, no higher recognizers are present. Any interpretation of raw events as completion, carry, theorem, package, or accepted object would exceed P_1 capability. Cannot-claim annotations record this limitation explicitly. Without them, the report could be mistaken for a stronger claim. □

Definition 9225.28 (Display policy). Lean4 (definition) BEDC.GroundCompiler.SourceReport.DisplayPolicy

A display policy *is an implementation-level convention determining how event flows and reports are printed. It may specify the raw event separator, empty-event marker, event-index format, indentation, warning format, and cannot-claim format. Display policy is output formatting, not formal input.*

Principle 9225.29 (Display policy is not semantics). Lean4 ✓ BEDC.GroundCompiler.SourceReport.display policy not semantics

Changing display policy must not change $\mathsf{Decode}(c)$ *or* $\mathsf{Compile}(S)$*. Display policy may change how reports look, but not what event flow they report.*

Proposition 9225.30 (Display policy does not affect round trip). Lean4 ✓ BEDC.GroundCompiler.SourceReport.di policy roundtrip

If two reports differ only by display policy, they represent the same decoded event flow and the same channel stream.

Proof. Display policy affects only output formatting. It does not alter Compile or Decode. □

Definition 9225.31 (Canonical display policy). Lean4 (definition) BEDC.GroundCompiler.SourceReport.CanonicalD

A canonical display policy for P_1 may use separator "$\mid$", empty event marker ϵ, event index form i, explicit body, terminator, and code fields, warnings after the event table, and cannot-claims last. This policy is recommended for reproducibility but remains output policy, not source semantics.

9225.5 No-host-leak and adequacy

Definition 9225.32 (Host leak in report). Lean4 (definition) BEDC.GroundCompiler.SourceReport.HostLeakInRepor

A source report contains a host leak *if it presents host-level structures as formal BEDC structures. Examples include a Python list interpreted as an event-flow object, JSON object interpreted as a package, YAML object interpreted as a chapter, string label interpreted as theorem identity, or host enum interpreted as status flow.*

Theorem 9225.33 (Host-leaking report is inadmissible). Lean4✓ BEDC.GroundCompiler.SourceReport.host leak inadmissible

If a P_1 report contains a host leak, then it is not an admissible source-event report.

Proof. A report is an output view of decoded event flow. Presenting host structures as formal BEDC structures introduces hidden input or misrepresents implementation artifacts as source-level objects. This violates the no-host-leak condition. □

Corollary 9225.34 (JSON and YAML reports are views only). Lean4✓ BEDC.GroundCompiler.SourceReport.json yaml reports view only

JSON or YAML output is allowed only when clearly marked as a human-readable report. It cannot be re-imported as formal compiler input.

Definition 9225.35 (P_1 audit checklist). Lean4 (definition) BEDC.GroundCompiler.SourceReport.P1AuditChecklist

A P_1 prototype must satisfy:

1. *all P_0 audit requirements;*
2. *legal channel streams decode to event flows;*
3. *reports display exact event segmentation;*
4. *reports display body and terminator separately;*
5. *reports flag source/channel distinctions;*
6. *reports include cannot-claim annotations for high-risk events;*
7. *reports do not claim package, theorem, certificate, or accepted-object recognition;*
8. *reports do not use host structures as formal objects;*
9. *report round trips agree with* Compile *and* Decode*;*
10. *illegal input generates rejection report, not source report.*

Theorem 9225.36 (P_1 adequacy theorem). Lean4✓ BEDC.GroundCompiler.SourceReport.p1 adequacy

If a prototype satisfies the P_1 audit checklist and its reports are sound, then it is adequate as a source-event reporting prototype.

Proof. The checklist ensures that the prototype implements all P_0 functions and faithfully reports decoded source events. Sound reports guarantee that displayed event flows match $\mathsf{Decode}(c)$. Cannot-claim annotations prevent overinterpretation, and no-host-leak restrictions prevent reports from becoming hidden input. Therefore the prototype is adequate at P_1. □

Corollary 9225.37 (P_1 adequacy does not imply higher adequacy). Lean4✓ BEDC.GroundCompiler.SourceReport. adequacy not higher

A P_1 prototype is not a source-level normalizer, package recognizer, name-certificate recognizer, theorem recognizer, motif analyzer, or self-hosting compiler.

Theorem 9225.38 (Source-event reporting is conservative)**.** `Lean4✓ BEDC.GroundCompiler.SourceReport.source report conservative`

Adding P_1 source-event reports adds no raw marks and creates no source-report recognition judgment. Every raw mark visible in a source-event report remains one of the display-alphabet constructors b0 *or* b1 *in the decoded event flow, and the report itself licenses no sameness rule, continuation rule, derived-interface formation rule, theorem rule, or accepted-object export rule.*

Proof. P_1 only decodes channel streams into event flows and prints reports. It introduces no new source events, asserts no msame, hsame, or psame, modifies neither Ext nor Cont, and recognizes or licenses no higher structure. Hence it is conservative over the finite kernel. □

Corollary 9225.39 (P_1 remains an address/report layer)**.** `Lean4✓ BEDC.GroundCompiler.SourceReport.p1 address report layer`

P_1 remains an address and reporting layer. It is not a mathematical object generator.

9225.6 Full example report

Example 9225.40 (Full P_1 report for completion skeleton)**.** *For input*

$$011001100011010110101011100011,$$

the report has legal status true and decoded event flow

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Its event table is

index	*raw*	*body*	*terminator*	*code*
0	0	0	11	011
1	00	00	11	0011
2	000	000	11	00011
3	01	010	11	01011
4	011	01010	11	0101011
5	100	1000	11	100011.

The round trip records

$$\mathsf{Decode}(c) = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

and

$$\mathsf{Compile}(\mathsf{Decode}(c)) = c.$$

Warnings record that the literal event pair $011 \mid 100$ *appears, with candidate interpretation only: possible source-level carry ledger, requiring generated carry recognizer flow, carry ledger, and classifier exactness. Cannot-claims record that this report does not prove* $011 = 100$*, does not prove* $\mathsf{hsame}(011, 100)$*, does not license* $\mathsf{FullAxis}^{\uparrow}$*, Completion, or dimension-lift structure, and is not a name-certificate flow, theorem flow, or acceptance-gate flow.*

Remark 9225.41 (Summary)**.** *This chapter defines P_1, the source-event reporting prototype. It extends P_0 with decoded event tables, event segmentation reports, body/terminator separation, round-trip reports, literal source warnings, cannot-claim annotations, display policy, and host-leak restrictions. It proves that illegal channel streams have no source-event report, round-trip reports are determined by the channel bijection, sound reports imply decoding, reports preserve*

raw event flows, reports do not recognize higher structures, warnings are weaker than recognitions, cannot-claims prevent overinterpretation, display policy is not semantics, host-leaking reports are inadmissible, P_1 *adequacy follows from audit and report soundness, and source-event reporting is conservative.*

Chapter 9226

Source-Level Normalizer Ledger Prototype

chapter 9225 defines P_1, the source-event reporting prototype. It decodes channel streams, displays source events, shows event segmentation, emits literal warnings, and attaches cannot-claim annotations. It does not recognize packages, certificates, theorems, motifs, or source-level relations.

This chapter defines the next prototype level:

$$P_2,$$

the *source-level normalizer ledger prototype*. Its task is to record candidate normalization ledgers on decoded source event flows, such as

$$011 \mid 100, \qquad 0011 \mid 0100, \qquad 10011 \mid 10100.$$

The essential rule is:

P_2 records candidate normalization ledgers; it does not automatically prove normalization relations.

Thus $011 \mid 100$ is not $011 = 100$, not $\mathsf{hsame}(011, 100)$, and not a Zeckendorf-carry judgment unless a later generated normalizer recognizer flow recognizes and certifies that relation.

9226.1 Adjacent pairs and candidates

Definition 9226.1 (Source-level adjacent pair)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.AdjPai`

Given an event flow

$$S = (w_0, w_1, \ldots, w_{n-1}),$$

a source-level adjacent pair *is a pair of consecutive events*

$$(w_i, w_{i+1})$$

with $0 \le i < n - 1$*. Write*

$$\mathsf{AdjPair}(S, i) = (w_i, w_{i+1}).$$

For

$$S = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

the adjacent pairs are $0 \mid 00$*,* $00 \mid 000$*,* $000 \mid 01$*,* $01 \mid 011$*, and* $011 \mid 100$*.*

Definition 9226.2 (Candidate normalizer pair)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.CandN`

A candidate normalizer pair *is a source-level adjacent pair* (w, v) *reported as a possible pre-normal event followed by a possible normal-address event. Write*

$$\mathsf{CandNormPair}(w, v).$$

This is a reporting-layer candidate, not a source-level normalization relation. In particular,

$$\mathsf{CandNormPair}(011, 100)$$

does not imply $011 \rightsquigarrow 100$.

Definition 9226.3 (Normalizer ledger candidate)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.Norm`

Given an event flow S *and a candidate-reporting predicate*

$$P_{\text{cand}}(w, v),$$

a normalizer ledger candidate *is a tuple*

$$(i, w, v)$$

such that

$$\mathsf{AdjPair}(S, i) = (w, v)$$

and the pair is reported by P_2 *as a candidate normalizer pair according to* P_{cand}*. Write*

$$\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v).$$

Remark 9226.4 (Candidate reporting is not recognition)**.** *The judgment*

$$\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v)$$

is not the relation $w \rightsquigarrow v$*. It says only that* $w \mid v$ *occurs at positions* $i, i+1$ *in* S *and is recorded as a candidate normalization ledger. A genuine source-level normalization relation requires a generated normalizer recognizer flow* R_N *and a judgment*

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v).$$

Definition 9226.5 (Generated normalizer recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.`

A generated normalizer recognizer *is an event flow*

$$R_N \in \mathsf{EventFlow}$$

supporting a relation

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v).$$

It reads as: R_N *recognizes the adjacent pair* $w \mid v$ *at position* i *in* S *as a source-level normalizer ledger. It is not an external rewrite table, regex, hardcoded rule, or host-level normalizer.*

Definition 9226.6 (Recognized normalizer ledger)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.Nor`

If

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v),$$

then (i, w, v) *is a* recognized normalizer ledger *in* S*. Write*

$$\mathsf{NormLedger}_{R_N}(S, i, w, v).$$

Definition 9226.7 (Recognizer-induced normalization relation). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNo`

If

$$\mathsf{NormLedger}_{R_N}(S, i, w, v)$$

and the certificate of R_N contains a soundness flow proving that w and v satisfy a source-level normalization relation, define

$$w \rightsquigarrow_{R_N} v.$$

The relation $\rightsquigarrow_{R_N}$ is induced by the recognizer R_N.

Principle 9226.8 (No normalization without recognizer). *Lean4✓* `BEDC.GroundCompiler.SourceNormalizer.no normalization without recognizer`

Candidate pairs do not produce normalization relations. A recognizer-induced normalization relation contains a recognized normalizer ledger as its ground witness. Only recognized normalizer ledgers, together with recognizer soundness, can support

$$w \rightsquigarrow v.$$

Thus

$$\mathsf{CandNormPair}(w, v) \not\Rightarrow w \rightsquigarrow v.$$

Proposition 9226.9 (Candidate pair preserves raw events). *Lean4✓* `BEDC.GroundCompiler.SourceNormalizer.can preserves raw`

If

$$\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v),$$

then w and v are both retained as source events in S.

Proof. By definition, $\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v)$ means $\mathsf{AdjPair}(S, i) = (w, v)$ together with candidate reporting. Hence $w = w_i$ and $v = w_{i+1}$, so both are source events in S. □

Corollary 9226.10 (Candidate normalizer does not erase pre-normal event). *Lean4✓* `BEDC.GroundCompiler.Sourc does not erase pre normal`

If P_2 reports $011 \mid 100$*, then both* 011 *and* 100 *remain in the decoded event flow. The event* 011 *is not deleted.*

Definition 9226.11 (Pre-normal and normal-address events). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNorm`

If $\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v)$ *or* $\mathsf{NormLedger}_{R_N}(S, i, w, v)$*, then w is the* pre-normal event *of the pair and v is the* normal-address event*. The normal-address role is a ledger role, not object identity, equality target, or mature object.*

Definition 9226.12 (P_2 normalizer ledger report). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.P2R`

Given a legal channel stream c with

$$S = \mathsf{Decode}(c),$$

a P_2 normalizer ledger report *contains at least the input channel stream, decoded event flow, adjacent pairs, candidate normalizer pairs, pre-normal events, normal-address events, candidate-only warnings, recognized normalizer ledgers if recognizers are supplied, cannot-claim annotations, and round-trip status.*

Principle 9226.13 (P_2 separates candidate and recognized status). Lean4✓ BEDC.GroundCompiler.SourceNormali recognized separated

A P_2 report must distinguish candidate normalizer pairs from recognized normalizer ledgers. It must not display a candidate pair as a proved relation.

Theorem 9226.14 (Candidate normalizer report does not establish source relation). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.candidate not relation

If P_2 reports only

$$\mathsf{NormCand}(P_{\text{cand}}, S, i, w, v)$$

and no judgment

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v)$$

is supplied, then one cannot infer

$$w \rightsquigarrow v.$$

Proof. By Definition 9226.8, a normalization relation requires generated normalizer recognition and soundness flow. A candidate report records adjacent events and candidate role only; it contains no recognizer soundness. □

Corollary 9226.15 (011 | 100 candidate does not prove carry). Lean4✓ BEDC.GroundCompiler.SourceNormalizer. 100 not carry

If P_2 reports 011 | 100 *but no generated carry recognizer flow is supplied, one cannot infer a Zeckendorf-carry relation or a classifier equivalence between* 011 *and* 100.

9226.2 Candidate policies

Definition 9226.16 (Candidate policy flow). Lean4 (definition) BEDC.GroundCompiler.SourceNormalizer.Candidat

A candidate policy flow *is an event flow*

$$P_{\text{cand}} \in \mathsf{EventFlow}$$

specifying which adjacent pairs P_2 may report as candidate normalizer pairs. It is not an external configuration file, YAML, regex, or hardcoded table.

Definition 9226.17 (Generated candidate-policy recognizer). Lean4 (definition) BEDC.GroundCompiler.SourceNorm

A generated candidate-policy recognizer *is an event flow*

$$R_{\text{cand}} \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesCandidatePolicy}(R_{\text{cand}}, P_{\text{cand}}).$$

Without a recognized candidate policy, P_2 may list literal adjacent pairs but may not mark any pair as a normalizer candidate.

Principle 9226.18 (No hardcoded normalizer table). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.no hardcoded table

The pairs 011 | 100, 0011 | 0100, *and* 10011 | 10100 *may be examples, but they must not be hardcoded as formal normalizer candidates. Formal candidate marking requires recognized candidate policy flow.*

Definition 9226.19 (Literal adjacent-pair listing). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.Lit`

Without recognized candidate policy, P_2 may still output every adjacent pair

$$(w_i, w_{i+1})$$

as a literal adjacent-pair listing. *This is not normalizer candidate recognition; it is only an extension of the source-event report.*

Proposition 9226.20 (Literal listing is policy-free). *Lean4✓* `BEDC.GroundCompiler.SourceNormalizer.literal listing policy free`

Literal adjacent-pair listing does not require candidate policy.

Proof. For $S = (w_0, \ldots, w_{n-1})$, the adjacent pairs (w_i, w_{i+1}) are determined by the sequence structure of S. No external rule selects their meaning. Listing them is therefore policy-free. □

Corollary 9226.21 (Candidate marking requires policy). *Lean4✓* `BEDC.GroundCompiler.SourceNormalizer.candi marking requires policy`

P_2 may list adjacent pairs without policy, but marking a pair as a candidate normalizer pair requires recognized candidate policy flow.

9226.3 Recognized normalizer ledgers

Definition 9226.22 (Normalizer recognizer certificate). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer`

A normalizer recognizer certificate is a record whose public fields are a certificate flow K_N, a generated normalizer recognizer R_N, a domain flow, a candidate-policy flow, a pre-normal criterion flow, a normal-address criterion flow, a relation-soundness flow, a ledger flow, a failure flow, a stability flow, and a seal flow. Write

$$\mathsf{NormalizerRecognizerCert}.$$

Definition 9226.23 (Normalizer soundness). *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.Normaliz`

A normalizer recognizer R_N is sound *when*

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v)$$

implies:

1. *w and v are adjacent in S at position i;*
2. *w is recognized as pre-normal event;*
3. *v is recognized as normal-address event;*
4. *relation-soundness flow supports $w \rightsquigarrow_{R_N} v$;*
5. *ledger flow retains S, i, w, v, failure branches, and classifier context.*

Theorem 9226.24 (Sound normalizer recognizer establishes source relation). *Lean4✓* `BEDC.GroundCompiler.Sour recognizer establishes relation`

If R_N is a sound normalizer recognizer and

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v),$$

then

$$w \rightsquigarrow_{R_N} v.$$

Proof. By soundness, the relation-soundness flow supports $w \rightsquigarrow_{R_N} v$. □

Corollary 9226.25 (Recognized carry is stronger than candidate carry)**.** Lean4✓ BEDC.GroundCompiler.SourceNo stronger than candidate

A recognized normalizer ledger with a sound normalizer recognizer can establish a source-level relation. A candidate pair only reports shape.

Definition 9226.26 (Ledger-preserving normalizer)**.** Lean4 (definition) BEDC.GroundCompiler.SourceNormalizer.Le

A normalizer recognizer R_N is ledger-preserving *when every judgment*

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v)$$

has ledger flow recording source flow S, position i, pre-normal event w, normal-address event v, normalizer role, classifier context, and failure cases.

Theorem 9226.27 (Ledger-preserving normalizer does not collapse raw events)**.** Lean4✓ BEDC.GroundCompiler.S preserving no collapse

If R_N is ledger-preserving and

$$\mathsf{RecognizesNormalizer}(R_N, S, i, w, v),$$

then w and v remain recorded as raw events even if a later classifier identifies them.

Proof. Ledger preservation records w and v as the two components of the pair and records their position in S. A later classifier may place them in one class, but the ledger retains the raw pre-normal and normal-address events. □

9226.4 Examples and reports

Example 9226.28 (Literal listing for completion skeleton)**.** *The channel stream*

$$011001100011010110101011100011$$

decodes to

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

The literal adjacent pairs are

$$0 \mid 00, \quad 00 \mid 000, \quad 000 \mid 01, \quad 01 \mid 011, \quad 011 \mid 100.$$

If no candidate policy is supplied, the report may only list these literal adjacent pairs. It may not mark $011 \mid 100$ *as a carry candidate.*

Example 9226.29 (Candidate policy for carry-looking pairs)**.** *Suppose a recognized candidate policy flow permits a class of source-normal-address-looking pairs. Then in*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

the pair $011 \mid 100$ *may be marked as a candidate normalizer pair with pre-normal event* 011 *and normal-address event* 100*. Without normalizer recognizer soundness, it is still not a proved relation.*

Example 9226.30 (Add/fold candidate normalizer)**.** *For input event flow*

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100,$$

the literal adjacent pairs are

$$0 \mid 00, \quad 00 \mid 001, \quad 001 \mid 0011, \quad 0011 \mid 0100.$$

If a candidate policy recognizes fold-style completion pairs, then $0011 \mid 0100$ *may be marked as a candidate normalizer pair. The report must still say that* 001 *is not* $\mathsf{Add}^{\uparrow}$, 0011 *is not* $\mathsf{Mul}^{\uparrow}$, 0100 *is not an accepted normal object, and the candidate pair does not prove a fold relation.*

Example 9226.31 (Higher carry candidate)**.** *For input*

$$100 \mid 10011 \mid 10100,$$

the literal adjacent pairs are $100 \mid 10011$ *and* $10011 \mid 10100$*. A recognized candidate policy may mark* $10011 \mid 10100$ *as a candidate normalizer pair. Without recognizer soundness, no relation*

$$10011 \rightsquigarrow 10100$$

is established.

Definition 9226.32 (P_2 report)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SourceNormalizer.P2ReportFormat`

A P_2 *report extends a* P_1 *report with an adjacent-pair table, candidate-policy status, candidate normalizer pairs, recognized normalizer ledgers if supplied, pre-normal and normal-address display, candidate-only warnings, and normalizer cannot-claims.*

Example 9226.33 (P_2 report for completion skeleton)**.** *For the completion skeleton channel stream, the decoded flow is*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

The adjacent-pair table is

index	*adjacent pair*
0	$0 \mid 00$
1	$00 \mid 000$
2	$000 \mid 01$
3	$01 \mid 011$
4	$011 \mid 100.$

If a carry-looking-pair policy is recognized, the report may mark position 4 *as candidate-only: pre-normal* 011*, normal-address* 100*. Warnings record that no normalizer recognizer, classifier exactness, or Zeckendorf-carry judgment is supplied. Cannot-claims record that* 011 *is not equal to* 100*, not* hsame *with* 100*, and does not license dimension-lift or accepted-object structure.*

Example 9226.34 (P_2 report with recognized normalizer)**.** *If a generated normalizer recognizer* R_N *is supplied and soundly recognizes*

$$\mathsf{RecognizesNormalizer}(R_N, S, 4, 011, 100),$$

the report may include a recognized normalizer ledger at position 4 *and the relation*

$$011 \rightsquigarrow_{R_N} 100.$$

It still may not claim $011 = 100$ *as raw equality or* $\mathsf{hsame}(011, 100)$ *unless a separate classifier or sameness certificate is supplied.*

9226.5 Soundness, conservativity, and adequacy

Theorem 9226.35 (P_2 candidate report is conservative). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.p2 candidate report conservative

Adding candidate normalizer reporting does not add source-level normalization relations: a candidate adjacent pair, without any generated normalizer recognizer recognizing the same source position, cannot produce an induced normalization relation.

Proof. Candidate reporting records adjacent source events and marks them according to candidate policy. It does not supply normalizer recognizer soundness. By Definition 9226.14, candidate reporting alone cannot establish $w \rightsquigarrow v$. □

Theorem 9226.36 (Recognized normalizer relation is source-level). Lean4✓ BEDC.GroundCompiler.SourceNormali normalizer relation source level

If R_N establishes

$$w \rightsquigarrow_{R_N} v,$$

then this relation is source-level and does not act directly on channel bits.

Proof. The relation $\rightsquigarrow_{R_N}$ is defined through $\mathsf{RecognizesNormalizer}(R_N, S, i, w, v)$, whose arguments are source-level event data. Channel bits are not arguments of the relation. To apply it to channel code, one must decode, apply the source relation, and compile. □

Corollary 9226.37 (P_2 does not permit channel rewrite). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.p2 no channel rewrite

Every recognized normalizer relation still returns an adjacent pair in the decoded event flow. It is therefore a source-level ledger relation, not a direct rewrite rule on channel strings. The correct path is decode, source normalizer ledger, and compile.

Theorem 9226.38 (P_2 is conservative over the finite kernel). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.p2 conservative over finite kernel

Adding P_2 candidate normalizer reports, candidate policies, normalizer recognizer relations, and normalizer ledgers adds no raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, theorem rules, or accepted-object export rules.

Proof. P_2 operates on decoded event flows. Candidate reporting records adjacent events. Recognized normalizer relations are source-level relations, not kernel equality. None of these generates new $\mathsf{b0}, \mathsf{b1}$ marks, modifies msame, hsame, or psame, changes Ext or Cont, or licenses derived interfaces. □

Corollary 9226.39 (P_2 remains below package recognition). Lean4✓ BEDC.GroundCompiler.SourceNormalizer.p2 below higher recognition

Even with recognized normalizer ledgers, P_2 does not turn host package, name-certificate, derivation-certificate, theorem-identifier, or closure-certificate data into primitive formal compiler inputs. Package, certificate, theorem, chapter, motif, metric, and accepted-object readings require additional generated recognizers.

Definition 9226.40 (P_2 audit checklist). Lean4 (definition) BEDC.GroundCompiler.SourceNormalizer.P2AuditCheck

A P_2 prototype must satisfy:

1. *all P_1 audit requirements;*
2. *all adjacent pairs in the decoded source flow are listable;*

3. *literal adjacent pairs are distinguished from candidate normalizer pairs;*
4. *candidate marking requires recognized candidate policy;*
5. *candidate normalizers are distinguished from recognized normalizer ledgers;*
6. *asserted $w \rightsquigarrow v$ relations require sound generated normalizer recognizers;*
7. *pre-normal and normal-address events are preserved in ledger;*
8. *channel-level rewriting is rejected;*
9. *every candidate pair has cannot-claim annotations;*
10. *no equality,* hsame*, theoremhood, name-certificate, or acceptance claim is made without the relevant certificate.*

Theorem 9226.41 (P_2 adequacy theorem)**.** `Lean4✓` `BEDC.GroundCompiler.SourceNormalizer.p2adequacy`

If a prototype satisfies the P_2 audit checklist, has sound P_1 reports, and every recognized normalizer relation is backed by a sound generated normalizer recognizer, then it is adequate as a source-level normalizer ledger prototype.

Proof. The checklist ensures that the prototype reports adjacent source events, candidate pairs, recognized normalizer ledgers, and warnings without collapsing levels. Sound P_1 reports provide the decoded source event flow. Sound normalizer recognizers justify any asserted $w \rightsquigarrow v$ relation. Cannot-claim annotations prevent overinterpretation. □

Corollary 9226.42 (P_2 adequacy does not imply higher adequacy)**.** `Lean4✓` `BEDC.GroundCompiler.SourceNormali adequacy not higher`

A P_2 prototype is not a motif analyzer, package recognizer, theorem recognizer, name-certificate recognizer, or accepted-object compiler.

9226.6 Full reports

Example 9226.43 (Full P_2 report for completion skeleton)**.** *For input*

$$011001100011010110101011100011,$$

the decoded flow is

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

The adjacent pairs are indexed as in Definition 9226.33. If a source-normal-address candidate policy is recognized, the pair at index 4 *is reported as candidate-only with pre-normal event* 011 *and normal-address event* 100*. Recognized normalizer ledgers are none. Warnings state that the candidate pair is not a proved carry relation, no generated normalizer recognizer is supplied, and no classifier exactness is supplied. Cannot-claims prohibit claiming equality,* hsame*, Zeckendorf-carry, dimension-lift, full-axis, or accepted-object status. Round trip succeeds by re-encoding the decoded flow to the original input.*

Example 9226.44 (Full P_2 report for fold skeleton)**.** *For input*

$$011001100101100101011010 0011,$$

the decoded flow is

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

The adjacent pairs are $0 \mid 00$, $00 \mid 001$, $001 \mid 0011$, *and* $0011 \mid 0100$. *If candidate policy marks the last pair, it is reported candidate-only with pre-normal event* 0011 *and normal-address event* 0100. *Warnings say that the pair is not a proved fold relation and no generated fold normalizer recognizer is supplied. Cannot-claims prohibit reading* 001 *as* $\mathsf{Add}^{\uparrow}$, 0011 *as* $\mathsf{Mul}^{\uparrow}$, *or* 0100 *as an accepted normal object.*

Remark 9226.45 (Summary)**.** *This chapter defines* P_2, *the source-level normalizer ledger prototype. It extends* P_1 *with adjacent-pair listing, candidate normalizer reporting, candidate policy flows, generated normalizer recognizers, recognized normalizer ledgers, pre-normal and normal-address roles, normalizer reports, and normalizer cannot-claims. It proves that candidate reporting is not recognition, candidate normalizer reports do not establish source relations, candidate marking requires recognized policy, sound normalizer recognizers establish recognizer-indexed source relations, ledger preservation does not collapse raw events,* P_2 *reports remain source-level and do not permit channel rewrite,* P_2 *is conservative over the finite kernel, and* P_2 *adequacy requires candidate/recognized separation plus cannot-claim annotations.*

Chapter 9227

Motif-Report Prototype

chapter 9226 defines P_2, the source-level normalizer ledger prototype. It adds adjacent-pair listings, candidate normalizer pairs, recognized normalizer ledgers, pre-normal and normal-address roles, candidate-only warnings, and normalizer cannot-claims to the P_1 source-event report.

This chapter defines the next prototype level:

$$\boxed{P_3 = P_2 + \text{generated motif recognizers} + \text{motif reports.}}$$

The task of P_3 is to recognize and report stable generative motifs on decoded source event flows: finite repetition, continuation, seal, carry, ledger compression, classifier quotient, reuse, and bridge-candidate motifs. It still does not export mature objects. A motif report cannot turn a support flow into $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, topology, or homology without later name-certificate, derivation-certificate, acceptance-gate, and proof flows.

The governing rule is:

a motif is not an external pattern; it is a stable subflow recognized by a generated recognizer.

9227.1 Motif candidates and recognition

Definition 9227.1 (Motif-report prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifReport`

A motif-report prototype *is an executable prototype P_3 satisfying:*

1. *P_3 includes all P_2 functions;*
2. *P_3 accepts a decoded event flow $S = \mathsf{Decode}(c)$;*
3. *P_3 may use generated motif recognizer flows R_M;*
4. *P_3 may recognize motif occurrences*

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu);$$

5. *P_3 may output motif reports;*
6. *P_3 does not treat motif reports as proofs, certificates, name-certificate flows, derivation-certificate flows, or acceptance gates.*

Write

$$\mathsf{MotifReportPrototype}(P_3).$$

Definition 9227.2 (Motif candidate). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifCandidate`
Given an event flow

$$S = (w_0, \ldots, w_{n-1}),$$

a motif candidate *is a source-level subflow $M \subseteq S$. The subflow may be contiguous or indexed. For*

$$S = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

the subflow $0 \mid 00 \mid 000$ *is a finite-repetition candidate,* $011 \mid 100$ *is a carry candidate, and the whole flow is a completion-skeleton candidate. The inclusion $M \subseteq S$ does not imply recognized motif status.*

Definition 9227.3 (Generated motif recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Generate`
A generated motif recognizer *is an event flow*

$$R_M \in \mathsf{EventFlow}.$$

It is not a host-level pattern matcher, regular expression, YAML motif table, hand-written tag table, or external semantic classifier. It supports a relation

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu),$$

where S is the ambient source event flow, M is the support subflow, and μ is a motif role flow.

Definition 9227.4 (Motif role flow). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifRole`
A motif role flow *is an event flow*

$$\mu \in \mathsf{EventFlow}$$

representing a motif role, such as finite repetition, continuation, seal, carry, classifier quotient, ledger compression, reuse, or bridge candidate. These names are explanatory; formally μ is not an external enum and must be represented in event-flow form.

Definition 9227.5 (Recognized motif occurrence). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Recogn`
If

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu),$$

then M is a recognized motif occurrence *in S under R_M and role μ. Write*

$$M \preceq_{\mu, R_M} S.$$

Remark 9227.6 (Candidate motif is weaker than recognized motif). *The inclusion $M \subseteq S$ only supplies a candidate subflow. The judgment*

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu)$$

is the recognized motif judgment. Thus $0 \mid 00 \mid 000$ *may look like finite repetition and* $011 \mid 100$ *may look like carry, but neither becomes recognized motif without generated recognizer flow.*

Definition 9227.7 (Motif ledger). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifLedger`
If

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu),$$

a motif ledger *is an event subflow L_M recording the ambient flow S, support subflow M, motif role μ, recognizer R_M, support positions, raw events, normal forms if any, classifier context if any, failure cases, and cannot-claim annotations. Write*

$$\mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

Principle 9227.8 (No recognized motif without ledger). *Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.recognize motif occurrence has ledger*

Every nontrivial motif recognition carries a motif ledger:

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu) \Rightarrow \exists L_M,\ \mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

Without such a ledger, the subflow may be reported only as a candidate motif.

Theorem 9227.9 (Motif recognition requires generated recognizer). *Lean4 ✓ BEDC.GroundCompiler.SemanticMoti recognition requires generated recognizer*

If M is a recognized motif occurrence in S, then the recognizing datum R_M is itself present as a formal generated event-flow input. Without such a generated motif recognizer flow, one cannot formally assert

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu).$$

Proof. Motif recognition is a source-level recognition judgment. By the no-hidden-input discipline, the recognizer cannot be supplied by an external pattern table, host regular expression, or human semantic tag. If no generated recognizer flow R_M exists, the source of the recognition is not present inside the event-flow system. □

Corollary 9227.10 (Host regular expressions are not motif recognizers). *Lean4 ✓ BEDC.GroundCompiler.Semantic regex not motif recognizer*

A host regular expression may locate a displayed substring or subflow for implementation convenience, but it is not formal motif recognition. Formal motif recognition requires generated motif recognizer flow and ledger.

9227.2 Basic motifs

Definition 9227.11 (Finite repetition motif). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.FiniteRepet*

A finite repetition motif *is a recognized event structure*

$$(x, xx, xxx, \ldots, x^k),$$

where $x \in \mathsf{RawEvent}$ and each successor event is obtained by source-level homogeneous extension of the preceding event. The displayed subflow $0 \mid 00 \mid 000$ is a candidate for $x = 0$. It is recognized only when a judgment

$$\mathsf{RecognizesMotif}(R_{\mathrm{rep}}, S, M, \mu_{\mathrm{rep}})$$

is supplied.

Definition 9227.12 (Continuation motif). *Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.ContinuationM*

A continuation motif *is a recognized event structure indicating that source events combine into a continuation result. Explanatory shapes include*

$$x \mid y \mid xy$$

or an input-flow, input-flow, continuation-result-flow, continuation-witness-flow pattern. A subflow such as $00 \mid 000 \mid 00000$ is only a candidate until recognized.

Definition 9227.13 (Seal motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.SealMotif`

A seal motif *is a recognized event structure indicating completion or closure of a repeat, thread, or process. Explanatory shapes include*

$$x \mid xx \mid xxx \mid x1 \mid x11$$

or finite stages, repeat-boundary witness, seal witness, and tail ledger. The subflow $0 \mid 00 \mid 000 \mid 01 \mid 011$ *is only a completion-like candidate until recognized.*

Definition 9227.14 (Carry motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.CarryMotif`

A carry motif *is a recognized source-level normalizer ledger consisting of a pre-normal event and a normal-address event. Explanatory shapes include*

$$011 \mid 100 \qquad \textit{and} \qquad 0011 \mid 0100.$$

The motif must retain both pre-normal and normal-address events.

Definition 9227.15 (Ledger compression motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.LedgerCo`

A ledger compression motif *is a recognized structure in which a visible event or package-like token hides source material while a ledger subflow retains provenance. It contains visible flow, hidden source flow, ledger flow, source-map or pullback flow, and compression seal flow.*

Definition 9227.16 (Classifier quotient motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.Classifie`

A classifier quotient motif *is a recognized structure in which distinct raw flows are assigned to one classifier class. It contains raw support flows, classifier witness, exactness witness, and a ledger preserving all raw supports. The raw events* 011 *and* 100 *together with a carry-classifier witness form only a candidate until recognized.*

Definition 9227.17 (Reuse motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.ReuseMotif`

A reuse motif *is a recognized structure in which a flow reuses recognized or accepted input flows and creates its own source, classifier, ledger, or certificate obligations. It contains input recognized flows, transport or reuse witness, new local source, new classifier, new ledger, strength bound, and reuse seal.*

Definition 9227.18 (Bridge-candidate motif). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.BridgeCand`

A bridge-candidate motif *is a recognized structure suggesting that two flows may support a bridge, without supplying the bridge certificate. It contains source flow, target-like flow, motif overlap, translation candidate, missing soundness flow, missing reflection or completeness flow, and missing no-host-leak ledger. A bridge-candidate motif is not a bridge.*

9227.3 Motif reports

Definition 9227.19 (Motif report). *Lean4 (definition)* `BEDC.GroundCompiler.SemanticMotif.MotifReport`

Given a decoded event flow S *and a family of generated motif recognizers* $\mathcal{R}$*, a* motif report *is an output view containing the ambient event flow, recognized motif occurrences, candidate motifs, motif roles, support subflows, motif ledgers, failure cases, warnings, and cannot-claim annotations. Write*

$$\mathsf{MotifReport}_{\mathcal{R}}(S).$$

Principle 9227.20 (Motif report is output, not input). Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.motif report output not input

A motif report is not formal input. It cannot replace recognizer flow, certificate flow, proof flow, name-certificate flow, derivation-certificate flow, or acceptance-gate flow.

Definition 9227.21 (Candidate motif section). Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.Candidatel

A motif report may contain a candidate motif section listing candidate finite repetition, continuation, seal, carry, reuse, and bridge structures. Every entry in that section must be marked candidate-only and requiring recognizer flow for recognition.

Definition 9227.22 (Recognized motif section). Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.Recognize

The recognized motif section may list only motif occurrences satisfying both

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu)$$

and

$$\mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

Principle 9227.23 (Candidate and recognized sections are separate). Lean4 ✓ BEDC.GroundCompiler.MotifReport recognized sections separate

A motif report must clearly separate candidate motifs from recognized motifs. It must not display a candidate motif as a recognized motif.

Theorem 9227.24 (Motif report soundness). Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.motif report soundness

If a motif report claims that M is a recognized motif in S, then there exist R_M, μ, and L_M such that

$$\mathsf{RecognizesMotif}(R_M, S, M, \mu)$$

and

$$\mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M).$$

Proof. By definition, the recognized motif section may list only entries with a recognizer judgment and ledger. Therefore every reported recognized motif has both. □

Corollary 9227.25 (Motif report cannot invent motifs). Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.motif report cannot invent

If recognizer or ledger is missing, a report may not promote a candidate motif to recognized motif.

9227.4 P_3 pipeline

Definition 9227.26 (P_3 pipeline). Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.P3Pipeline

Given channel code c, the P_3 pipeline has stages:

1. *legality check;*
2. *decode to source event flow S;*
3. *source-event report;*
4. *adjacent-pair listing;*

5. *candidate normalizer ledger report;*
6. *motif recognizer selection;*
7. *motif recognition;*
8. *motif ledger extraction;*
9. *motif report generation;*
10. *cannot-claim annotation.*

Recognizer selection cannot come from an external table. It must come from a recognized recognizer-family flow or be recorded as bootstrap obligation.

Definition 9227.27 (Recognizer family flow)**.** Lean4 (definition) BEDC.GroundCompiler.SemanticMotif.RecognizerF

A recognizer family flow *is an event flow*

$$F_R \in \mathsf{EventFlow}$$

recognized as a family of motif recognizer flows $R_1, \ldots, R_k$*. Write*

$$\mathsf{RecognizesRecognizerFamily}(R_F, F_R, (R_1, \ldots, R_k)).$$

Without a recognized recognizer-family flow, P_3 *can generate only candidate motif reports, not recognized motif reports.*

Principle 9227.28 (No hardcoded motif family)**.** Lean4 ✓ BEDC.GroundCompiler.SemanticMotif.no hardcoded motif family

The finite-repetition, carry, seal, and reuse recognizers may not be hardcoded as a formal recognizer family. They must come from recognized recognizer-family flow or be recorded as bootstrap obligations.

Theorem 9227.29 (Without recognizer family, P_3 is candidate-only)**.** Lean4 ✓ BEDC.GroundCompiler.SemanticMot family candidate only

If P_3 *has no recognized recognizer-family flow, then it can output only a candidate motif section, not a recognized motif section.*

Proof. Recognized motifs require a judgment $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$ and a motif ledger. Without a recognizer family, there is no available R_M. Thus no recognized motif judgment can be produced. □

9227.5 Examples

Example 9227.30 (Candidate motif report for completion skeleton)**.** *For input*

$$011001100011010110101011100011,$$

the decoded event flow is

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

A candidate motif report may list finite repetition candidate $0 \mid 00 \mid 000$*, repeat-boundary candidate* 01*, seal candidate* 011*, carry candidate* $011 \mid 100$*, and completion-skeleton candidate equal to the whole flow. Warnings state that the report is candidate-only, no motif recognizer family is supplied, no name-certificate flow is supplied, and no acceptance-gate flow is supplied. Cannot-claims state that finite repetition does not imply* $\mathsf{Nat}^{\uparrow}$*, seal does not imply completion, carry does not imply Zeckendorf-carry, and completion skeleton does not imply* $\mathsf{FullAxis}^{\uparrow}$*.*

Example 9227.31 (Recognized motif report for completion skeleton)**.** *Assume a recognizer family contains certified finite-repetition, seal, and carry recognizers. A report may then list recognized finite repetition with support* $0 \mid 00 \mid 000$*, recognized seal with support* 011*, and recognized carry with support* $011 \mid 100$*, pre-normal event* 011*, normal-address event* 100*, and retained ledger. It still cannot claim that* $\mathsf{FullAxis}^{\uparrow}$ *is licensed, completion is exported, dimension-lift is established, or* $011 = 100$ *as* hsame *without further certificate flows.*

Example 9227.32 (Candidate motif report for fold skeleton)**.** *For input*

$$01100110010110010101101000 11,$$

the decoded flow is

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

A candidate report may list finite source candidate $0 \mid 00$*, additive-recursion candidate* 001*, fold-completion candidate* $0011 \mid 0100$*, and the whole fold skeleton candidate. Cannot-claims state that* 001 *does not imply* $\mathsf{Add}^{\uparrow}$*,* $0011 \mid 0100$ *does not imply* $\mathsf{Mul}^{\uparrow}$*, and* 0100 *is not an accepted fold object.*

Example 9227.33 (Nat-like and List-like motif comparison)**.** *For flows*

$$0 \mid 00 \mid 000 \mid 01 \qquad \textit{and} \qquad 0 \mid 00 \mid 000,$$

a candidate report may show the shared finite spine $0 \mid 00 \mid 000$*, the extra event* 01 *in the Nat-like flow, and the missing payload classifier, constructor-shape ledger, and length ledger for a List-like reading. Cannot-claims state that shared spine does not imply same object, and neither flow licenses* $\mathsf{Nat}^{\uparrow}$ *or list structure without the corresponding name-certificate flow.*

9227.6 Motif counts

Definition 9227.34 (P_3 motif count)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MotifReportCounts.MotifCountP3`

Given a recognized motif report, define

$$\mathsf{MotifCount}_{P_3}(S, \mu)$$

as the count status for role μ *in the report: it is* $\mathsf{count}(n)$ *when the recognized section supplies the list of* n *recognized motif occurrences of role* μ*, and it is* $\mathsf{undefined}$ *when the report has only a candidate section and no recognized motif list for* μ*. Undefined count status is not the same datum as* $\mathsf{count}(0)$*.*

Definition 9227.35 (P_3 carry count)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MotifReportPrototype.P3CarryCoun`

The P_3 *carry count is the number of recognized carry motif occurrences. If only candidate carry pairs are present, the recognized carry count is undefined; a separate candidate-carry count may be reported.*

Definition 9227.36 (P_3 seal count)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MotifReportPrototype.P3SealCount`

The P_3 *seal count is the number of recognized seal motif occurrences. Candidate seal events appear only in the candidate section.*

Proposition 9227.37 (Candidate count and recognized count differ)**.** *Lean4 ✓* `BEDC.GroundCompiler.MotifReport` `recognized count differ`

Candidate motif count cannot replace recognized motif count: there is a report state with one candidate motif occurrence and no recognized motif list for the same role, so the recognized motif count is undefined rather than $\mathsf{count}(1)$*.*

Proof. Candidate count depends on shape or candidate policy. Recognized count depends on generated recognizers and motif ledgers. These are different evidence levels. Substituting candidate count for recognized count treats unrecognized patterns as recognized motifs. □

Corollary 9227.38 (Missing recognizer does not imply zero motif). Lean4✓ `BEDC.GroundCompiler.MotifReportCo` `recognizer not zero`

If a motif recognizer is missing, the recognized count for that motif is undefined or recognizer-missing, not zero. Otherwise absence of recognition would be mistaken for absence of structure.

9227.7 Soundness, adequacy, and conservativity

Definition 9227.39 (Sound motif report). Lean4 (definition) `BEDC.GroundCompiler.MotifReportPrototype.SoundMo`

A motif report is sound *when:*

1. *the candidate section lists only candidates;*
2. *the recognized section lists only motifs with recognizer and ledger;*
3. *every recognized motif has support subflow in the decoded event flow;*
4. *every motif ledger records support and role;*
5. *cannot-claim annotations cover all high-risk motifs;*
6. *missing recognizers are reported as missing, not as negative evidence.*

Theorem 9227.40 (Sound motif report theorem). Lean4✓ `BEDC.GroundCompiler.MotifReportPrototype.sound` `motif report has recognizer ledger`

If P_3 outputs a sound motif report, then every recognized motif occurrence in the report has a source-level recognition judgment and motif ledger.

Proof. By soundness, the recognized section contains only entries with recognizer and ledger. Hence each recognized motif has $\mathsf{RecognizesMotif}(R_M, S, M, \mu)$ and $\mathsf{MotifLedger}_{R_M}(S, M, \mu, L_M)$. □

Theorem 9227.41 (Motif report does not license objects). Lean4✓ `BEDC.GroundCompiler.MotifReportPrototype.` `report does not license objects`

There exists a sound motif report whose underlying event flow is not an accepted object flow. Thus sound motif reporting does not, by itself, imply accepted-object status or license any derived interface.

Proof. Motif reports supply motif recognition and ledger information. The empty recognized section is sound when it contains no unsupported recognized motifs, but the empty event flow is not an accepted object flow because acceptance requires nonempty certified export structure. Derived interfaces and accepted objects require certificate and acceptance-gate data not supplied by a sound motif report alone. □

Corollary 9227.42 (Recognized carry motif does not imply dimension-lift). Lean4✓ `BEDC.GroundCompiler.Motifl` `carry motif not dimension lift`

There is a recognized carry-motif witness whose ambient event flow does not contain the dimension-lift certificate subflow. Thus recognized carry evidence such as $011 \mid 100$ *does not by itself imply dimension-lift structure; that structure requires its own certificate subflow and ledger.*

Corollary 9227.43 (Recognized seal motif does not imply completion). Lean4✓ BEDC.GroundCompiler.MotifRepo seal motif not completion

There is a recognized seal-motif witness whose ambient event flow does not contain the completion certificate subflow. Thus recognized seal evidence does not by itself imply completion; completion requires compatible-thread source, classifier, seal certificate, and tail ledger.

Definition 9227.44 (P_3 audit checklist). Lean4 (definition) BEDC.GroundCompiler.MotifReportPrototype.P3AuditC

A P_3 prototype must satisfy:

1. *all P_2 audit requirements;*
2. *candidate motifs are distinguished from recognized motifs;*
3. *recognized motifs require generated motif recognizers;*
4. *nontrivial recognized motifs require motif ledgers;*
5. *missing recognizers are reported as undefined, not zero;*
6. *source-level motifs are separated from channel substrings;*
7. *every high-risk motif has cannot-claim annotations;*
8. *no derived interface is licensed;*
9. *no object equality is asserted;*
10. *no theoremhood is asserted;*
11. *no hardcoded motif table is used as formal input;*
12. *implementation-level recognizers have bootstrap obligations unless certified.*

Theorem 9227.45 (P_3 adequacy theorem). Lean4✓ BEDC.GroundCompiler.MotifReportPrototype.p3 adequacy

If a prototype satisfies the P_3 audit checklist, has sound P_2 reports, and every recognized motif is backed by generated motif recognizer and motif ledger, then it is adequate as a motif-report prototype.

Proof. The audit checklist separates candidates, recognized motifs, recognizers, ledgers, missing recognizers, and cannot-claims. Sound P_2 reports supply the decoded source event flow. Generated recognizers and ledgers prevent host-level or hardcoded pattern matches from becoming formal motif recognition. □

Corollary 9227.46 (P_3 adequacy does not imply higher adequacy). Lean4✓ BEDC.GroundCompiler.MotifReportPr adequacy not higher

There exists a sound P_3 motif report whose underlying event flow has no recognized package-flow evidence, no recognized theorem-flow evidence, and no accepted-object-flow evidence. Thus P_3 motif-report adequacy alone is not higher-layer adequacy.

Theorem 9227.47 (P_3 is conservative over the finite kernel). Lean4✓ BEDC.GroundCompiler.MotifReportPrototyp conservative over finite kernel

Adding motif candidate reports, recognized motif reports, motif ledgers, motif counts, and P_3 cannot-claim annotations adds no raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, theorem rules, or accepted-object export rules.

Proof. P_3 operates on decoded event flows and recognized subflows. It generates no new $\mathsf{b0}, \mathsf{b1}$, adds no msame, hsame, or psame rules, does not alter Ext or Cont, and does not license derived interfaces or accepted objects. □

Corollary 9227.48 (P_3 remains an analysis layer)**.** `Lean4✓ BEDC.GroundCompiler.MotifReportPrototype.p3 analysis layer not object generator`

There exists a sound P_3 motif report with an empty recognized section, undefined recognized carry count, and no accepted-object-flow evidence. Thus P_3 can provide motif-level analysis and reporting without generating a mathematical object.

9227.8 Full report examples

Example 9227.49 (P_3 report for completion skeleton)**.** *For input*

$$011001100011010110101011100011,$$

the decoded flow is

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

The candidate section lists finite repetition $0 \mid 00 \mid 000$*, repeat-boundary* 01*, seal* 011*, carry* $011 \mid 100$*, and completion skeleton equal to the whole flow. Suppose the recognizer family has recognized finite-repetition and carry recognizers, but missing seal and completion recognizers. The recognized section lists finite repetition with retained ledger and carry with pre-normal* 011*, normal-address* 100*, and retained ledger. Seal and completion counts are undefined because their recognizers are missing. Cannot-claims say that there is no completion export, no* $\mathsf{FullAxis}^{\uparrow}$*, no dimension-lift, no object equality, no* $\mathsf{hsame}(011, 100)$*, and no theoremhood.*

Example 9227.50 (P_3 report for fold skeleton)**.** *For input*

$$011001100101100101011010 0011,$$

the decoded flow is

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

If additive and fold recognizers are missing, the report is candidate-only: finite source $0 \mid 00$*, additive-recursion candidate* 001*, and fold-completion candidate* $0011 \mid 0100$*. Recognized motifs are none. Cannot-claims state that* 001 *is not* $\mathsf{Add}^{\uparrow}$*,* 0011 *is not* $\mathsf{Mul}^{\uparrow}$*,* 0100 *is not an accepted fold object, and no fold relation is present without recognizer.*

Remark 9227.51 (Summary)**.** *This chapter defines P_3, the motif-report prototype. It extends P_2 with motif candidates, generated motif recognizers, motif role flows, recognized motif occurrences, motif ledgers, candidate and recognized sections, motif counts, and P_3 audit rules. It proves that motif recognition requires generated recognizers, host regular expressions are not motif recognizers, nontrivial recognized motifs require ledger, candidate and recognized report sections must be separated, reports cannot invent motifs, missing recognizer families make P_3 candidate-only, candidate and recognized counts differ, missing recognizers do not mean zero motif, sound motif reports provide recognition judgments and ledgers, motif reports do not license objects, and P_3 is conservative over the finite kernel.*

Chapter 9228

Metric-Report Prototype

chapter 9227 defines P_3, the motif-report prototype. It separates candidate motifs, recognized motifs, motif ledgers, recognized motif counts, missing recognizers, and cannot-claim annotations. This chapter defines the next prototype level:

$$\boxed{P_4 = P_3 + \text{admissible metric computation} + \text{metric report.}}$$

The task of P_4 is to compute structural indicators from recognized motif reports: prefix overlap, motif counts, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, bridge depth, normal-address distance, motif distance, and flow signature vectors.

The governing rule is:

$$\boxed{\text{metrics are explanatory summaries, not proofs, certificates, or object identities.}}$$

Thus P_4 cannot infer that equal metric profiles give equal theories, equal signature vectors give equal objects, distance zero gives theorem equivalence, high overlap gives a bridge, positive carry index gives dimension, or positive seal depth gives a completion object.

9228.1 Metric protocols and admissibility

Definition 9228.1 (Metric-report prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricReport`

A metric-report prototype *is an executable prototype P_4 satisfying:*

1. *P_4 includes all P_3 functions;*
2. *P_4 receives a decoded event flow S and a recognized motif report;*
3. *P_4 uses a recognized metric protocol flow;*
4. *P_4 computes admissible metrics;*
5. *P_4 emits a metric report;*
6. *P_4 does not treat metric reports as proofs, certificates, bridges, name-certificate flows, derivation-certificate flows, or acceptance gates.*

Write

$$\mathsf{MetricReportPrototype}(P_4).$$

Definition 9228.2 (Metric candidate). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricCandidate`

A metric candidate *is a proposed numerical value, sequence, set, or summary to compute on an event flow. Examples include seal depth, carry index, ledger depth, reuse depth, bridge depth, motif count, motif distance, prefix overlap, and compression ratio. A metric candidate is not yet an admissible metric.*

Definition 9228.3 (Generated metric recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.Generated`

A generated metric recognizer *is an event flow*

$$R_{\mathrm{met}} \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesMetric}(R_{\mathrm{met}}, m),$$

where m is a metric specification flow. The specification is not an external function name, Python function, JSON field, or hand-written metric table.

Definition 9228.4 (Metric protocol flow). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricProtocolFl`

A metric protocol flow *is an event flow*

$$P_{\mathrm{met}} \in \mathsf{EventFlow}$$

recognized as a metric-computation protocol. It contains metric-selection flow, metric-domain flow, recognizer-family flow, normalization-policy flow, ledger-policy flow, undefined-value-policy flow, comparison-policy flow, report-policy flow, cannot-claim-policy flow, and protocol seal flow. Write

$$\mathsf{MetricProtocolFlow}(P_{\mathrm{met}}).$$

Principle 9228.5 (No metric without protocol). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.no metric without protocol`

External metric configuration, weight files, manual metric tables, host metric functions, and external similarity functions are not formal inputs. A metric is BEDC-admissible only when specified by recognized metric protocol flow.

Definition 9228.6 (Admissible metric). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.AdmissibleMetric`

A metric M for an event flow S is admissible *relative to protocol P_{met} when:*

1. *M is given by a recognized metric specification flow;*
2. *its input is the decoded event flow S;*
3. *all motifs used by M come from a recognized motif report;*
4. *all ledgers used by M are recognized ledger flows;*
5. *undefined cases are governed by the protocol flow;*
6. *cannot-claim entries are generated by the report policy;*
7. *M does not depend on channel substring search, external object names, host tags, YAML fields, manual grouping, or unrecognized semantic labels.*

Write

$$\mathsf{AdmissibleMetric}(M, S, P_{\mathrm{met}}).$$

Theorem 9228.7 (Unrecognized metric is inadmissible). *Lean4✓ BEDC.GroundCompiler.MetricsFlow.unrecognized metric inadmissible*

If there is no recognized metric specification flow and no recognized metric protocol flow, then a value or summary cannot be used as a BEDC-admissible metric.

Proof. Admissibility requires recognized specification and protocol flow. Without them, the domain, computation rule, undefined cases, recognizer family, ledger policy, and cannot-claim policy are externally supplied. That is hidden input. □

Corollary 9228.8 (Host similarity score is not admissible). *Lean4✓ BEDC.GroundCompiler.MetricsFlow.host similarity score not admissible*

A similarity score produced only by a host program, embedding model, external statistic, or manual choice is not a BEDC-admissible metric without recognized metric protocol flow.

9228.2 Basic metrics

Definition 9228.9 (Recognized motif count). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.RecognizedMot*

Given an event flow S, a recognized motif report, and motif role μ, define

$$\mathsf{MotifCount}(S, \mu)$$

as the number of recognized μ-role occurrences listed in the recognized motif report. If the report contains only candidate motifs and no recognized section, then $\mathsf{MotifCount}(S, \mu)$ *is undefined, not zero.*

Definition 9228.10 (Candidate motif count). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.CandidateMoti*

The candidate motif count *for role μ is the number of μ-like candidates listed in the candidate section. It is distinct from recognized motif count.*

Proposition 9228.11 (Candidate count is weaker than recognized count). *Lean4✓ BEDC.GroundCompiler.Metrics count weaker*

Candidate motif count cannot replace recognized motif count.

Proof. Candidate count records shape or candidate policy. Recognized count requires generated recognizer and motif ledger. Treating candidate count as recognized count promotes unverified patterns to recognized motifs. □

Corollary 9228.12 (Missing recognizer does not imply zero). *Lean4✓ BEDC.GroundCompiler.MetricsFlow.missing recognizer not zero*

If a motif recognizer is missing, the corresponding recognized motif count is undefined or recognizer-missing. It is not zero.

Definition 9228.13 (Seal depth). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.SealDepth*

The seal depth *of S is the number of recognized seal motif occurrences in S. If the seal recognizer is missing, seal depth is undefined. Candidate seal events may be counted separately as candidate seal count.*

Definition 9228.14 (Carry index). *Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.CarryIndex*

The carry index *of S is the number of recognized carry motif occurrences in S. If only candidate pairs such as* $011 \mid 100$ *or* $0011 \mid 0100$ *are present, but no recognized carry motif is supplied, carry index is undefined.*

Definition 9228.15 (Ledger depth). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.LedgerDepth`

The ledger depth *of S is the maximum nesting depth among recognized ledger flows in S. If there is no recognized ledger flow, ledger depth is* 0*. If there are suspected ledger candidates without recognizer, the report must separately mark ledger-recognizer missing; it must not treat the value* 0 *as a structural conclusion.*

Definition 9228.16 (Reuse depth). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.ReuseDepth`

The reuse depth *of S is the length of the longest dependency chain of recognized reuse motifs. If the reuse recognizer is missing, reuse depth is undefined.*

Definition 9228.17 (Bridge depth). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.BridgeDepth`

The bridge depth *of S is the maximum bridge-chain length among recognized bridge motifs or bridge flows. If only bridge-candidate motifs are present, bridge depth is undefined, while bridge-candidate count may be reported separately.*

Definition 9228.18 (Classifier compression ratio). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.Compress`

If a recognized classifier quotient in S sends k raw support flows into r recognized classes with $r > 0$, define

$$\mathsf{CompressionRatio}(S) = \frac{k}{r}.$$

This metric requires classifier recognizer, exactness recognizer, and ledger recognizer.

Proposition 9228.19 (Compression ratio requires exactness). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.compr ratio requires exactness`

Without classifier exactness flow, admissible compression ratio cannot be computed.

Proof. Compression ratio counts how raw supports are classified. If classifier exactness is not recognized, the classes may not be sound, may not respect the intended sameness, and may lack ledger completeness. The metric is therefore inadmissible. □

9228.3 Comparison metrics

Definition 9228.20 (Raw prefix overlap). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.RawPrefixOverlap`

For event flows

$$S = (s_0, \ldots, s_{m-1}), \qquad T = (t_0, \ldots, t_{n-1}),$$

the raw prefix overlap length

$$\mathsf{RawPrefixLen}(S, T)$$

is the largest k such that $s_i = t_i$ for every $0 \le i < k$.

Definition 9228.21 (Recognized prefix overlap). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.Recognized`

The recognized prefix overlap *of S and T is the set of shared recognized prefix motif records. Sharing requires the same support shape, same motif role, ledger-compatible evidence, and either the same recognizer family or recognized compatible recognizers.*

Proposition 9228.22 (Raw prefix overlap does not imply recognized prefix overlap). ↪ *variant* `BEDC.GroundCompiler.MetricsFlow.raw prefix weaker`

Two event flows can share raw prefix without sharing recognized semantic prefix.

Proof. The same raw prefix may be recognized as different motif roles in different contexts, or may be unrecognized in one flow. Therefore raw prefix overlap alone does not imply recognized prefix overlap. □

Definition 9228.23 (Motif overlap). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MotifOverlap`

The motif overlap *of two flows is the intersection of their recognized motif profiles:*

$$\mathsf{MotifOverlap}(S,T) = \mathsf{MotifProfile}(S) \cap \mathsf{MotifProfile}(T),$$

where intersection is computed by motif role, support shape, and ledger-compatible equivalence.

Definition 9228.24 (Motif distance). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MotifJaccardDistance`

When the protocol defines the denominator, the motif distance *is*

$$\mathsf{MotifDist}(S,T) = 1 - \frac{|\mathsf{MotifProfile}(S) \cap \mathsf{MotifProfile}(T)|}{|\mathsf{MotifProfile}(S) \cup \mathsf{MotifProfile}(T)|}.$$

If the denominator is 0*, the undefined-value policy in the metric protocol determines the report entry.*

Definition 9228.25 (Normal-address overlap and distance). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.`

Given normalizer recognizers, the normal-address profile *of S is the set of recognized records $(w, v, L_{w\to v})$. The overlap of S and T is the intersection of these profiles. When defined, the normal-address distance is*

$$\mathsf{NormDist}(S,T) = 1 - \frac{|\mathsf{NormOverlap}(S,T)|}{|\mathsf{NormalAddressProfile}(S) \cup \mathsf{NormalAddressProfile}(T)|}.$$

Zero denominator behavior is protocol-defined.

Definition 9228.26 (Flow signature vector). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.FlowSignature`

The flow signature vector *of S is*

$$\mathsf{Sig}(S) = (\mathsf{SealDepth}, \mathsf{CarryIndex}, \mathsf{LedgerDepth}, \mathsf{CompressionRatio}, \mathsf{ReuseDepth}, \mathsf{BridgeDepth})_S.$$

It is defined only when each component is admissible or when the protocol explicitly handles undefined components.

Theorem 9228.27 (Signature equality does not imply object equality). *↪ variant* `BEDC.GroundCompiler.MetricsFlo` `do not imply object equality`

If

$$\mathsf{Sig}(S) = \mathsf{Sig}(T),$$

one cannot infer that S and T represent the same theorem, chapter, object, classifier class, or bridge target.

Proof. A signature vector is a finite summary. Distinct event flows may have equal carry count, seal count, ledger depth, or other component values while having different supports, ledgers, classifiers, proofs, and certificates. Object equivalence or theorem equivalence requires classifier, proof, bridge, or certificate flow. □

Corollary 9228.28 (Signature vector is explanatory summary). *↪ variant* `BEDC.GroundCompiler.MetricsFlow.metri` `explanatory not identificatory`

Signature vectors may support comparison and clustering, but they are not identity criteria.

9228.4 Metric reports

Definition 9228.29 (Metric report). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricReport`

Given event flow S, metric protocol flow P_{met}, and a recognized motif report, a metric report *is an output view containing the input flow, recognized motif summary, candidate motif summary, metric protocol, computed admissible metrics, undefined metrics, missing recognizers, ledger requirements, warnings, and cannot-claim annotations. Write*

$$\mathsf{MetricReport}(S, P_{\text{met}}).$$

Definition 9228.30 (Undefined metric item). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.UndefinedMetr`

An undefined metric item *records why a metric is undefined: recognizer missing, ledger missing, classifier exactness missing, normalizer missing, division by zero unresolved by protocol, candidate-only evidence, or unresolved bootstrap obligation.*

Principle 9228.31 (Undefined metric is explicit). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.undefined metric explicit`

If a metric is inadmissible or undefined, the report must record an undefined metric item. It must not silently omit the metric or treat missing data as 0.

Theorem 9228.32 (Silent zero is unsound). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.silent zero unsound`

If a metric is undefined because a recognizer is missing, but the report records it as 0*, then the metric report is unsound.*

Proof. Missing recognizer means the system cannot determine whether the motif exists. Recording 0 asserts absence of the motif. That turns epistemic absence into structural absence, which is unsound. □

Definition 9228.33 (Metric report soundness). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricRepor`

A metric report is sound *when every listed source flow is witnessed as a formal event-flow input, every listed undefined metric has a corresponding undefined metric item in the report, and the report satisfies the cannot-claim guard for nontrivial metric signatures.*

Theorem 9228.34 (Sound metric report theorem). *Lean4 ✓* `BEDC.GroundCompiler.MetricsFlow.sound metric report`

If P_4 outputs a sound metric report, then every source flow listed by the report is a formal event-flow input, and every undefined metric item listed by the report is explicitly recorded in the report.

Proof. Both claims are the defining witnesses carried by metric-report soundness. □

9228.5 Metric claims and cannot-claims

Definition 9228.35 (Metric claim). *Lean4 (definition)* `BEDC.GroundCompiler.MetricsFlow.MetricClaim`

A metric claim *is a statement based on a metric report, such as: S and T share a finite-repetition motif, S has carry index* 1*, S has seal depth* 2*, S and T have motif distance* 0.25*, or S has classifier compression ratio* 3*. Such a claim is admissible only when the relevant metric report is sound and the metric is admissible.*

Definition 9228.36 (Forbidden metric claim). Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.ForbiddenMet

The following metric claims are forbidden unless supported by additional certificate flows:

- *same metric profile implies same object;*
- *same signature vector implies same theorem;*
- *motif distance zero implies proof equivalence;*
- *high motif overlap implies bridge;*
- *positive carry index implies dimension;*
- *positive seal depth implies completion object;*
- *compression ratio greater than one implies quotient exactness;*
- *positive reuse depth implies a reuse certificate.*

Theorem 9228.37 (Forbidden metric claims need certificates). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.forbi metric claims need certificates

Every forbidden metric claim is invalid without additional classifier, proof, bridge, name-certificate, derivation-certificate, or acceptance-gate flow.

Proof. Metric reports summarize event-flow structure. Forbidden claims assert object equivalence, proof equivalence, bridge existence, dimension, completion, quotient exactness, or reuse certification. These require independent certificate flows. Metrics alone do not supply them. □

Corollary 9228.38 (Metric reports require cannot-claim sections). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow.m report requires cannot claim sections

Every nontrivial metric report must contain a cannot-claim section listing the stronger claims that the metrics do not support.

9228.6 P_4 pipeline and examples

Definition 9228.39 (P_4 pipeline). Lean4 (definition) BEDC.GroundCompiler.MetricsFlow.P4Pipeline

Given channel code c, the P_4 pipeline consists of P_0 legality check, P_1 source-event report, P_2 normalizer ledger report, P_3 motif report, metric protocol recognition, admissible metric computation, undefined metric audit, cannot-claim audit, and metric report generation.

Theorem 9228.40 (P_4 pipeline is source-determined relative to protocol). Lean4 ✓ BEDC.GroundCompiler.Metrics source determined relative to protocol

If $c \in \mathsf{LegalZStream}$, P_{met} is a recognized metric protocol flow, and all selected recognizers are deterministic, then the P_4 output is uniquely determined.

Proof. The channel bijection decodes c to a unique event flow S. The recognized protocol fixes metric selection, recognizer family, undefined-value policy, and report policy. Deterministic recognizers determine motif reports and ledgers. Metrics are computed from those data. □

Corollary 9228.41 (Different protocols may yield different reports). Lean4 ✓ BEDC.GroundCompiler.MetricsFlow. protocols may yield different reports

Different metric protocol flows may choose different recognizer families, metric selections, weights, undefined-value policies, or report policies. Different reports under different protocols are therefore not contradictory.

Example 9228.42 (Completion skeleton metric report)**.** *For input*

$$011001100011010110101011100011,$$

the decoded flow is

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100.$$

Assume the recognized motif report includes finite repetition $0 \mid 00 \mid 000$*, carry* $011 \mid 100$*, and seal* 011*. The metric report may record motif counts for finite repetition, carry, and seal as* 1*, seal depth* 1*, carry index* 1*, and ledger depth at least* 1 *depending on the carry ledger. Completion-object status and full-axis export remain undefined. Cannot-claims say that seal depth* 1 *does not imply completion, carry index* 1 *does not imply dimension-lift, and the report does not establish* $011 = 100$*.*

Example 9228.43 (Fold skeleton metric report)**.** *For input*

$$011001100101100101011010 0011,$$

the decoded flow is

$$0 \mid 00 \mid 001 \mid 0011 \mid 0100.$$

If recognized motifs include additive recursion 001 *and fold normalizer* $0011 \mid 0100$*, the report may count each as* 1*. Carry index is* 1 *only if the fold normalizer is recognized under carry role. Seal depth is undefined unless a seal recognizer recognizes* 0011*, and reuse depth is undefined unless a reuse recognizer is supplied.*

Example 9228.44 (Completion/fold comparison)**.** *For*

$$S_{\mathrm{Comp}} = 0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

and

$$S_{\mathrm{Fold}} = 0 \mid 00 \mid 001 \mid 0011 \mid 0100,$$

the raw prefix overlap is $0 \mid 00$*. A recognized shared finite-source motif exists only if recognizer evidence is supplied. Motif distance is protocol-relative. Cannot-claims record that shared prefix does not imply bridge and distance values do not imply object identity.*

9228.7 Adequacy and conservativity

Definition 9228.45 (P_4 audit checklist)**.** Lean4 (definition) `BEDC.GroundCompiler.MetricsFlow.P4AuditChecklist`
A P_4 *prototype must satisfy:*

1. *all* P_3 *audit requirements;*
2. *recognized metric protocol flow is required;*
3. *candidate counts are separated from recognized counts;*
4. *missing recognizers are marked undefined, not zero;*
5. *metric dependencies are recorded;*
6. *ledger requirements are recorded;*
7. *undefined metric items are recorded;*

8. *cannot-claim sections are included;*
9. *object equality is not inferred from metrics;*
10. *theorem equivalence is not inferred from metrics;*
11. *bridge is not inferred from motif overlap;*
12. *external metric tables are not formal inputs.*

Theorem 9228.46 (P_4 adequacy theorem)**.** Lean4✓ BEDC.GroundCompiler.MetricsFlow.p4 adequacy

If a prototype satisfies the P_4 audit checklist, uses sound P_3 reports, and computes only admissible metrics under recognized metric protocol flow, then it is adequate as a metric-report prototype.

Proof. The checklist ensures that metrics are computed only from recognized event-flow structures, that candidate and recognized evidence are separated, undefined cases are explicit, and cannot-claims are included. Sound P_3 reports validate motif data. Recognized metric protocol prevents metric selection and policies from becoming hidden inputs. □

Corollary 9228.47 (P_4 adequacy does not imply higher adequacy)**.** Lean4✓ BEDC.GroundCompiler.MetricsFlow.p4 adequacy not higher

There exists an adequate P_4 metric-report prototype whose source flow is not an accepted object flow. Thus P_4 adequacy alone is not higher-layer adequacy; package, name-certificate, derivation-certificate, theorem, chapter, and accepted-object status require separate recognition or acceptance evidence.

Theorem 9228.48 (P_4 is conservative over the finite kernel)**.** Lean4✓ BEDC.GroundCompiler.MetricsFlow.p4 conservative over finite kernel

Adding metric reports, signature vectors, distances, counts, compression ratios, and undefined metric items adds no raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, theorem rules, or accepted-object export rules.

Proof. P_4 computes summaries over decoded event flows and recognized motif reports. It does not generate new $\mathsf{b0}, \mathsf{b1}$, add msame, hsame, or psame, alter Ext or Cont, or supply name certificates, derivation certificates, proof flows, or acceptance gates. □

Corollary 9228.49 (P_4 remains an analysis layer)**.** *P_4 provides metric analysis. It is not a mathematical object generator, proof checker, or certificate generator.*

9228.8 Full reports

Example 9228.50 (P_4 report for completion skeleton)**.** *For*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

assume recognized finite repetition, seal, and carry motifs with supports $0 \mid 00 \mid 000$, 011, *and* $011 \mid 100$. *The report records*

$$\mathsf{MotifCount}(\textit{finite repetition}) = 1, \quad \mathsf{MotifCount}(\textit{seal}) = 1, \quad \mathsf{MotifCount}(\textit{carry}) = 1,$$

and seal depth 1, *carry index* 1, *and ledger depth* 1 *when the ledger is one level. Compression ratio, reuse depth, and bridge depth are undefined if the corresponding recognizers are missing. Cannot-claims say that seal depth does not imply completion, carry index does not imply dimension-lift, classifier exactness is not supplied, no name-certificate flow is supplied, no derivation-certificate flow is supplied, no acceptance-gate flow is supplied, and the report is not proof.*

Example 9228.51 (P_4 comparison report)**.** *For the completion and fold skeletons, the report may record raw prefix* $0 \mid 00$*, recognized shared finite-source motif when recognizer evidence exists, completion-specific seal and carry motifs, and fold-specific additive recursion and fold normalizer motifs. Raw prefix length is* 2*, while shared recognized motif count and motif distance are protocol-relative. Cannot-claims say that shared prefix does not imply subobject, motif distance does not imply theorem equivalence, and no bridge certificate is supplied.*

Remark 9228.52 (Summary)**.** *This chapter defines* P_4*, the metric-report prototype. It adds metric candidates, generated metric recognizers, metric protocol flows, admissible metrics, recognized and candidate motif counts, seal depth, carry index, ledger depth, reuse depth, bridge depth, classifier compression ratio, prefix overlap, motif overlap, motif distance, normal-address distance, signature vectors, undefined metric items, metric claims, forbidden metric claims, the* P_4 *pipeline, and the* P_4 *audit checklist. It proves that unrecognized metrics are inadmissible, host similarity scores are not admissible metrics, candidate counts are weaker than recognized counts, missing recognizers do not imply zero, compression ratio requires exactness, raw prefix overlap does not imply recognized prefix overlap, signature equality does not imply object equality, silent zero is unsound, sound metric reports are backed by recognized structures, forbidden metric claims require certificates, the pipeline is source-determined relative to protocol, and* P_4 *is conservative over the finite kernel.*

Chapter 9229

Package and NameCert Recognition Prototype

chapter 9228 defines P_4, the metric-report prototype. It can produce motif reports, metric reports, prefix overlap, seal depth, carry index, ledger depth, classifier compression ratio, reuse depth, and cannot-claim audits. It remains an analysis layer. It does not recognize packages, name certificates, derivation certificates, theorems, chapters, or accepted objects.

This chapter defines the next prototype level:

$$\boxed{P_5 = P_4 + \text{package-flow recognition} + \mathsf{NameCert}\text{-flow recognition.}}$$

This is the first layer that moves from analysis reports to certificate-structure recognition. The governing rule remains:

packages and name certificates are not inputs; they are structures recognized inside decoded event flows.

Thus P_5 does not accept host package structures, host name-certificate records, YAML packages, JSON certificates, external field tables, host ASTs, or manually supplied source, pattern, classifier, stability, or ledger records. The only formal input is a channel stream or decoded event flow.

9229.1 P_5 prototype interface

Definition 9229.1 (Package and NameCert recognition prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Packa`

A package and NameCert recognition prototype *is an executable prototype P_5 satisfying:*

1. *P_5 includes all P_4 functions;*
2. *P_5 can run generated package recognizers on decoded event flows;*
3. *P_5 can recognize package flows;*
4. *P_5 can run generated* NameCert *recognizers inside recognized package flows or decoded event flows;*
5. *P_5 can recognize* NameCert *flows;*
6. *P_5 can generate package and* NameCert *recognition reports;*
7. *P_5 does not recognize derivation certificates, acceptance gates, theorems, chapters, or accepted objects unless later prototype levels supply those recognizers.*

Write

$$\mathsf{PkgNameCertPrototype}(P_5).$$

Principle 9229.2 (P_5 does not accept package objects). *Lean4 ✓ BEDC.GroundCompiler.PackageNameCertPrototyp no package object input*

There is no formal input of the form

$$\mathsf{PackageStruct} \to \mathsf{Code}$$

or

$$\mathsf{NameCertStruct} \to \mathsf{Code}.$$

The formal input remains EventFlow *or its channel code in* LegalZStream*. If the input is channel code c, P_5 first forms*

$$S = \mathsf{Decode}(c)$$

and then recognizes package flows and NameCert *flows in S.*

Definition 9229.3 (P_5 input). *Lean4 (definition) BEDC.GroundCompiler.PackageNameCertPrototype.P5FormalInput*

The input to P_5 may be displayed as a channel stream c or as an event flow S. If the input is c, then decoding is performed first:

$$S = \mathsf{Decode}(c).$$

If $c \notin$ LegalZStream*, then P_5 rejects and produces neither package report nor* NameCert *report.*

Definition 9229.4 (P_5 output). *Lean4 (definition) BEDC.GroundCompiler.PackageNameCertPrototype.P5Output*

The output of P_5 is a report flow or human-readable report containing decoded event flow, package candidates, recognized package flows, package ledgers, NameCert *candidates, recognized* NameCert *flows, recognized source-specification, pattern-specification, classifier-specification, stability-certificate, and ledger-policy flows, missing fields, failure cases, warnings, and cannot-claim annotations.*

This output is report data. It is not input, proof, or accepted object.

9229.2 Package-flow recognition

Definition 9229.5 (Package candidate). *Lean4 (definition) BEDC.GroundCompiler.PackageNameCertPrototype.Packa*

Given a decoded event flow S, a package candidate *is a source-level subflow*

$$P \subseteq S.$$

It has not yet been recognized as a package flow. Write

$$\mathsf{PkgCand}(S, P).$$

Definition 9229.6 (Generated package recognizer). *Lean4 (definition) BEDC.GroundCompiler.PackageNameCertProt*

A generated package recognizer *is an event flow*

$$R_P \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesPkg}(R_P, S, P).$$

This says that R_P recognizes subflow P of ambient event flow S as a package flow. The recognizer is not a host parser, YAML schema, or external package grammar.

Definition 9229.7 (Recognized package flow). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype`

If

$$\mathsf{RecognizesPkg}(R_P, S, P),$$

then P is a recognized package flow *in S. Write*

$$\mathsf{PkgFlow}_{R_P}(S, P).$$

Definition 9229.8 (Package role subflows). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype.Pa`

If

$$\mathsf{RecognizesPkg}(R_P, S, P),$$

then R_P may recognize subflows of P as package roles:

$$\mathsf{PkgSource}_{R_P}(P, P_0), \quad \mathsf{PkgVisible}_{R_P}(P, P_1), \quad \mathsf{PkgPattern}_{R_P}(P, P_2),$$

$$\mathsf{PkgClassifier}_{R_P}(P, P_3), \quad \mathsf{PkgLedger}_{R_P}(P, P_4), \quad \mathsf{PkgSeal}_{R_P}(P, \sigma_P).$$

These roles are not external field labels; they are source-level subflow roles recognized by R_P.

Definition 9229.9 (Complete package recognition). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertProt`

The judgment

$$\mathsf{RecognizesPkg}(R_P, S, P)$$

is complete package recognition *when there exist $P_0, P_1, P_2, P_3, P_4, \sigma_P$ satisfying the source, visible, pattern, classifier, ledger, and seal role judgments of Definition 9229.8. Intuitively, these are the source material, visible package interface, package pattern, package classifier, package ledger, and package seal.*

Principle 9229.10 (No nontrivial package without ledger). *Lean4 ✓* `BEDC.GroundCompiler.PackageNameCertProto` `package recognition has ledger`

Every nontrivial package flow must have package-ledger subflow

$$\mathsf{PkgLedger}_{R_P}(P, P_4).$$

Without ledger, it may be reported only as a package candidate, not as a recognized nontrivial package flow.

Theorem 9229.11 (Incomplete package cannot be used as package flow). *Lean4 ✓* `BEDC.GroundCompiler.Package` `package not package`

If P lacks any recognized source, visible, pattern, classifier, ledger, or seal subflow, then P cannot be used as a complete recognized package flow.

Proof. Complete package recognition requires all six recognized role subflows. If any is missing, complete recognition fails. In the nontrivial case, missing ledger also violates Definition 9229.10. □

Corollary 9229.12 (Visible token alone is not a package). *Lean4 ✓* `BEDC.GroundCompiler.PackageNameCertProto` `token alone not package`

A visible token or short event such as 01, 011, 100, *or* 001 *is not by itself a complete package flow. A visible-only witness has a visible subflow but no recognized source, pattern, classifier, ledger, or seal subflow. Complete package recognition requires those role subflows, so visible-only evidence cannot be used as a complete recognized package flow.*

Definition 9229.13 (Package report). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype.Packag`

A package report *is a P_5 output view containing package candidates, recognized package flows, package source subflows, visible subflows, pattern subflows, classifier subflows, ledger subflows, seal subflows, missing package fields, package failure cases, and package cannot-claim annotations.*

Principle 9229.14 (Package report is output). *Lean4✓* `BEDC.GroundCompiler.PackageNameCertPrototype.packag` `report output not input`

A package report is an output view, not an input structure. Later stages must continue from decoded event flows and recognizer judgments, not from a report treated as a package object.

9229.3 NameCert-flow recognition

Definition 9229.15 (Name candidate flow). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype.N`

A name candidate flow *is an event subflow*

$$N \subseteq S$$

representing a proposed source-level visible name candidate. It is not an external object name, string label, or theorem identifier. Write

$$\mathsf{NameCand}(S, N).$$

Definition 9229.16 (NameCert candidate). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype.N`

A NameCert candidate *is an event subflow*

$$C_N \subseteq S$$

not yet recognized as a NameCert *flow. Write*

$$\mathsf{NameCertCand}(S, C_N, N).$$

Definition 9229.17 (Generated NameCert recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertP`

A generated NameCert recognizer *is an event flow*

$$R_N \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesNameCert}(R_N, S, C_N, N).$$

It recognizes subflow C_N of ambient event flow S as a NameCert *flow for name candidate N.*

Definition 9229.18 (Recognized NameCert flow). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertProto`

If

$$\mathsf{RecognizesNameCert}(R_N, S, C_N, N),$$

then C_N is a recognized NameCert *flow for N in S. Write*

$$\mathsf{NameCertFlow}_{R_N}(S, C_N, N).$$

Definition 9229.19 (NameCert field subflows). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototyp`

If

$$\mathsf{RecognizesNameCert}(R_N, S, C_N, N),$$

then R_N may identify five field subflows and a seal:

$$\mathsf{NCSource}_{R_N}(C_N, C_0), \quad \mathsf{NCPattern}_{R_N}(C_N, C_1), \quad \mathsf{NCClassifier}_{R_N}(C_N, C_2),$$

$$\mathsf{NCStability}_{R_N}(C_N, C_3), \quad \mathsf{NCLedger}_{R_N}(C_N, C_4), \quad \mathsf{NCSeal}_{R_N}(C_N, \sigma_N).$$

Definition 9229.20 (Complete NameCert recognition). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertP`

The judgment

$$\mathsf{RecognizesNameCert}(R_N, S, C_N, N)$$

is complete NameCert recognition *when there exist $C_0, C_1, C_2, C_3, C_4, \sigma_N$ satisfying all six judgments in Definition 9229.19.*

Principle 9229.21 (No NameCert without five fields). *Lean4✓* `BEDC.GroundCompiler.PackageNameCertPrototype` `namecert without five fields`

A NameCert *flow must have recognized source, pattern, classifier, stability, ledger, and seal. If any is missing, the flow remains a* NameCert *candidate, not a recognized* NameCert *flow.*

Theorem 9229.22 (Incomplete NameCert candidate cannot license name). *Lean4✓* `BEDC.GroundCompiler.Packag` `namecert candidate cannot license`

If C_N lacks any recognized source, pattern, classifier, stability, ledger, or seal subflow for every generated NameCert *recognizer, then C_N cannot serve as a P_5 licensing witness for the name candidate N.*

Proof. Name licensing requires complete recognized NameCert flow. If any required field or seal is missing, complete recognition fails, so N is not licensed. □

Corollary 9229.23 (Name event alone does not license interface). *Lean4✓* `BEDC.GroundCompiler.PackageNameCer` `event alone not license`

A name candidate event N cannot by itself license a derived interface. A recognized NameCert *flow is required.*

Definition 9229.24 (Licensed name at P_5). *Lean4 (definition)* `BEDC.GroundCompiler.PackageNameCertPrototype.L`

At the P_5 layer, define

$$\mathsf{LicensedName}_{P_5}(S, N)$$

to hold when there exist C_N and R_N such that

$$\mathsf{RecognizesNameCert}(R_N, S, C_N, N)$$

and that recognition is complete and sound. This licenses a name, not an accepted mature object.

Theorem 9229.25 (P_5 licensing is weaker than accepted export). *Lean4✓* `BEDC.GroundCompiler.PackageNameCert` `license weaker than export`

There exist event flows S and N such that

$$\mathsf{LicensedName}_{P_5}(S, N)$$

holds while S is not an accepted object flow. Thus P_5 licensing does not by itself imply accepted export at any strength.

Proof. Take the empty decoded flow as the carrier of a complete NameCert recognition witness for a name candidate under the P_5 recognition predicates. This supplies a P_5 license. The same empty flow is not an accepted object flow, because accepted object flow requires the nonempty exported flow together with recognized certificate support. Hence P_5 licensing is weaker than accepted export. □

9229.4 Package-sourced NameCert flows

Definition 9229.26 (NameCert over package). Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertPrototype`

A NameCert *flow C_N is* over package flow *P when:*

1. *P is a recognized package flow;*
2. *the source field of C_N references or contains the source, visible, or ledger subflows of P;*
3. *the ledger field of C_N retains the package ledger of P.*

Write

$$\mathsf{NameCertOverPkg}(C_N, N, P).$$

Proposition 9229.27 (NameCert over package requires package ledger). Lean4 ✓ `BEDC.GroundCompiler.PackageN` `over package requires ledger`

If

$$\mathsf{NameCertOverPkg}(C_N, N, P),$$

then P has a recognized package-ledger subflow.

Proof. A NameCert over a package requires the NameCert ledger to retain package provenance. Without package ledger, that source material cannot be traced. □

Corollary 9229.28 (Package without ledger cannot support NameCert). Lean4 ✓ `BEDC.GroundCompiler.PackageN` `without ledger no namecert over package`

A nontrivial package flow without recognized ledger cannot support a derived name through NameCert *recognition.*

Definition 9229.29 (Package-sourced NameCert report). Lean4 (definition) `BEDC.GroundCompiler.PackageNameCer`

A package-sourced NameCert report *contains recognized package flow P, recognized* NameCert *flow C_N, name candidate N, the source-field relation between C_N and P, the ledger-pullback relation, missing-field report, and cannot-claim annotations.*

Principle 9229.30 (Package-sourced NameCert is still not accepted export). Lean4 ✓ `BEDC.GroundCompiler.Packa` `sourced namecert licenses name not export`

If a recognized package flow with ledger supports a recognized NameCert *flow, then the package-sourced record supplies a P_5 licensed-name witness. It still does not supply accepted-object export unless the accepted-object witness is separately present; mature export requires derivation-certificate and acceptance-gate flows.*

9229.5 P_5 reports and soundness

Definition 9229.31 (P_5 report). Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertPrototype.P5Report`

A P_5 report *is an output view containing input channel stream, decoded event flow, P_4 motif and metric summaries if available, package candidates, recognized package flows, package field*

status, NameCert *candidates, recognized* NameCert *flows,* NameCert *field status, licensed names at P_5, missing fields, ledger audit, warnings, and cannot-claim annotations.*

Definition 9229.32 (Package field status)**.** Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertPrototype.Pa`

For each package candidate P, the report records source, visible, pattern, classifier, ledger, and seal status as present, missing, or candidate-only.

Definition 9229.33 (NameCert field status)**.** Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertPrototype.`

For each NameCert *candidate C_N, the report records source, pattern, classifier, stability, ledger, and seal status as present, missing, or candidate-only.*

Principle 9229.34 (Missing field is not negative theorem)**.** Lean4✓ `BEDC.GroundCompiler.PackageNameCertProto` `field report not negative theorem`

If a field is reported missing, the report only says that the field is missing from the recognized flow. It does not assert that the theory contains no such field, unless recognizer completeness and domain coverage have been established.

Definition 9229.35 (P_5 cannot-claim annotations)**.** Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertProt`

A P_5 report must include cannot-claim annotations saying that recognized package flow does not imply NameCert *flow, recognized* NameCert *flow does not imply derivation-certificate flow, licensed name does not imply accepted object, package token alone is insufficient, name event alone is insufficient, missing fields block recognition, channel code is not proof, and report is not certificate.*

Definition 9229.36 (Sound P_5 report)**.** Lean4 (definition) `BEDC.GroundCompiler.PackageNameCertPrototype.Sound`

A P_5 report is sound *when:*

1. *every recognized package flow is backed by generated package recognizer;*
2. *every nontrivial recognized package flow has required ledger;*
3. *every recognized* NameCert *flow is backed by generated* NameCert *recognizer;*
4. *every recognized* NameCert *flow has five fields and seal;*
5. *every licensed name is backed by sound recognized* NameCert *flow;*
6. *all missing fields are explicitly reported;*
7. *candidate-only entries are not reported as recognized;*
8. *cannot-claim annotations are complete.*

Theorem 9229.37 (Sound P_5 report theorem)**.** Lean4✓ `BEDC.GroundCompiler.PackageNameCertPrototype.sound` `p5 report`

If P_5 outputs a sound report, then every recognized package flow and recognized NameCert *flow in the report has the corresponding recognition judgment and ledger support.*

Proof. Soundness requires each recognized package flow to be supported by generated package recognition and required ledger, and each recognized NameCert flow to be supported by generated NameCert recognition with five fields and seal. Hence every recognized item has recognition and ledger support. □

Theorem 9229.38 (Sound P_5 licensed name theorem)**.** Lean4✓ `BEDC.GroundCompiler.PackageNameCertPrototype` `p5 licensed name`

If a sound P_5 report asserts

$$\mathsf{LicensedName}_{P_5}(S, N),$$

then there exists a complete and sound recognized NameCert *flow C_N for N.*

Proof. By definition, a licensed name at P_5 requires complete and sound recognized NameCert flow. Sound reporting ensures that licensed-name entries are not supported by candidate-only flows. □

Corollary 9229.39 (P_5 licensed name is not accepted export)**.** Lean4✓ BEDC.GroundCompiler.PackageNameCertPr licensed not accepted export

Even when P_5 soundly licenses N, it does not follow that N is accepted at any strength. Accepted export requires recognized derivation-certificate and acceptance-gate flows.

9229.6 Adequacy

Definition 9229.40 (P_5 audit checklist)**.** Lean4 (definition) BEDC.GroundCompiler.PackageNameCertPrototype.P5Au

A P_5 prototype must satisfy:

1. *all P_4 audit requirements;*
2. *package candidates are distinguished from recognized package flows;*
3. *recognized package flows require generated package recognizers;*
4. *nontrivial packages require package ledger;*
5. NameCert *candidates are distinguished from recognized* NameCert *flows;*
6. *recognized* NameCert *flows require generated* NameCert *recognizers;*
7. *recognized* NameCert *flows require five fields and seal;*
8. *missing fields are reported explicitly;*
9. *licensed names are distinguished from accepted objects;*
10. *derivation-certificate and acceptance-gate recognition is not claimed unless later recognizers are supplied;*
11. *host package and* NameCert *structures are not formal inputs;*
12. *cannot-claim annotations cover package and* NameCert *overinterpretation risks.*

Theorem 9229.41 (P_5 adequacy theorem)**.** Lean4✓ BEDC.GroundCompiler.PackageNameCertPrototype.p5 adequacy

If a prototype satisfies the P_5 audit checklist, uses sound P_4 reports, and every recognized package flow and NameCert *flow is backed by generated recognizers and required ledger, then it is adequate as a package and* NameCert *recognition prototype.*

Proof. The checklist enforces separation between candidates and recognized package or NameCert structures, requires generated recognizers, requires ledgers and the five NameCert fields, and prevents overinterpretation as derivation certificate or acceptance gate. Sound P_4 reports ensure that lower-level event, motif, and metric data are reliable. □

Corollary 9229.42 (P_5 adequacy does not imply higher adequacy)**.** Lean4✓ BEDC.GroundCompiler.PackageNameCe adequacy not higher

A P_5 prototype does not yet support derivation-certificate flow, acceptance-gate flow, theorem flow, chapter flow, accepted-object export, or self-hosting compiler recognition.

9229.7 Examples

Example 9229.43 (Package candidate without ledger)**.** *Suppose a decoded event flow contains subflow* 011*. A report may list it as a visible package candidate. If there is no package source, pattern, classifier, ledger, or seal, the status is candidate-only with missing source, pattern, classifier, ledger, and seal. Cannot-claims say that* 011 *is not a package, not* $\mathsf{FullAxis}^{\uparrow}$*, and not completion.*

Example 9229.44 (Recognized package flow over completion skeleton)**.** *Suppose the decoded flow is*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100$$

and a generated package recognizer identifies source flow $0 \mid 00 \mid 000$*, visible flow* $011 \mid 100$*, repeat-boundary or seal-like pattern flow, source-level normal-address classifier candidate, ledger retaining finite prefixes and both* 011 *and* 100*, and a package seal. The report may state that a package flow is recognized. It still cannot claim that a* NameCert *exists, that* $\mathsf{FullAxis}^{\uparrow}$ *is licensed, or that completion is accepted.*

Example 9229.45 (NameCert candidate with missing ledger)**.** *Suppose a* NameCert *candidate contains source, pattern, classifier, and stability flows, plus seal, but no ledger. The report lists those fields as present and ledger as missing, with status incomplete. Cannot-claims say that the candidate does not license a name and that missing ledger blocks* NameCert *recognition.*

Example 9229.46 (Complete Axis-like NameCert flow)**.** *A complete Axis-like* NameCert *flow may recognize a name candidate Axis, source given by a* b0*-emission flow, pattern given by repeated* b0 *extension, classifier given by* hsame *restricted to the* b0 *spine, stability given by closure under extension or continuation, ledger recording source history for each visible axis token, and a* NameCert *seal. If all fields are recognized and sound,* P_5 *may report* $\mathsf{LicensedName}_{P_5}(S, \mathrm{Axis})$*. It must also report that accepted Axis export still requires derivation-certificate and acceptance-gate flows.*

Example 9229.47 (Package-sourced NameCert)**.** *A report may contain recognized package flow* P *with source finite-prefix flow, visible token flow, and ledger flow, and recognized* NameCert *flow* C_N *whose source references* P*, whose pattern and classifier extend those of* P*, whose stability is recognized, and whose ledger pulls back the package ledger. The report may state* $\mathsf{NameCertOverPkg}(C_N, N, P)$*. It still cannot infer* $\mathsf{AcceptedExport}(N, s)$*.*

9229.8 Conservativity

Theorem 9229.48 (P_5 is conservative over the finite kernel)**.** `Lean4✓` `BEDC.GroundCompiler.PackageNameCertProt` `conservative over finite kernel`

Adding package-flow recognition, NameCert*-flow recognition, package reports,* NameCert *reports,* P_5 *licensed-name reports, and package-sourced* NameCert *reports adds no raw marks, history constructors, sameness rules, continuation rules, theorem rules, derivation-certificate rules, acceptance-gate rules, or mature-object export rules.*

Proof. P_5 operates on decoded event flows and generated recognizer relations. It generates no new b0, b1, adds no msame, hsame, or psame, does not alter Ext or Cont, and does not create derivation certificates or acceptance gates. It can license names only at the NameCert-flow level; mature export still requires higher certificate flows. □

Corollary 9229.49 (P_5 remains below accepted-object compiler)**.** P_5 *can recognize package flows and* NameCert *flows, and can license names at the* P_5 *layer, but it cannot export mature objects.*

Remark 9229.50 (Summary)**.** *This chapter defines P_5, the package and* NameCert *recognition prototype. It adds package candidates, generated package recognizers, recognized package flows, package source, visible, pattern, classifier, ledger, and seal subflows, package reports, name candidate flows,* NameCert *candidates, generated* NameCert *recognizers, recognized* NameCert *flows, source, pattern, classifier, stability, ledger, and seal subflows, complete* NameCert *recognition, P_5 licensed names,* NameCert *over package, field-status reports, and the P_5 audit checklist. It proves that incomplete packages and incomplete* NameCert *candidates cannot license their intended structures, visible tokens and name events alone are insufficient, P_5 licensing is weaker than accepted export,* NameCert *over package requires package ledger, sound reports provide recognition and ledger support, P_5 adequacy requires generated recognizers and cannot-claims, and P_5 is conservative over the finite kernel.*

Chapter 9230

DerivCert and AcceptGate Recognition Prototype

chapter 9229 defines P_5, the package and NameCert recognition prototype. It can recognize package flows, NameCert flows, source-specification flows, pattern-specification flows, classifier-specification flows, stability-certificate flows, ledger-policy flows, and P_5-level licensed names. It still cannot export mature objects.

This chapter defines the next prototype level:

$$\boxed{P_6 = P_5 + \text{derivation-certificate recognition} + \text{strength recognition} + \text{acceptance-gate recognition}.}$$

This is the first layer where accepted object code may be reported. The rule is strict:

$$\boxed{\text{accepted object code is not bare code; it is code of a recognized acceptance flow.}}$$

The prototype does not accept derivation-certificate structures, acceptance-gate structures, strength enums, accepted objects, object names, theorem identifiers, proof-status tables, or bridge-status tables as formal inputs. It accepts only channel streams or decoded event flows and then recognizes derivation-certificate, strength, and acceptance-gate flows inside them.

9230.1 P_6 prototype interface

Definition 9230.1 (DerivCert and AcceptGate recognition prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.De`

A DerivCert and AcceptGate recognition prototype *is an executable prototype P_6 satisfying:*

1. *P_6 includes all P_5 functions;*
2. *P_6 recognizes strength flows;*
3. *P_6 recognizes derivation-certificate flows;*
4. *P_6 recognizes acceptance-gate flows;*
5. *when matching recognized* NameCert *and derivation-certificate flows are present, P_6 can report accepted object code;*
6. *P_6 does not accept accepted objects as input;*
7. *P_6 does not treat code existence as acceptance;*
8. *P_6 does not treat acceptance as object equality;*

9. P_6 does not treat accepted object code as proof of all later properties.

Write

$$\mathsf{DerivAcceptPrototype}(P_6).$$

Principle 9230.2 (Accepted object is not input). *Lean4 ✓ BEDC.GroundCompiler.DerivCertGenerated.p6 external inputs not formal*

There is no formal input

$$\mathsf{AcceptedObject} \to \mathsf{Code}, \qquad \mathsf{DerivCertStruct} \to \mathsf{Code}, \qquad \mathsf{AcceptGateStruct} \to \mathsf{Code}.$$

The only formal inputs remain event flows and legal channel streams. If the input is channel stream c, P_6 first forms $S = \mathsf{Decode}(c)$ and then recognizes derivation-certificate and acceptance-gate flows in S.

Definition 9230.3 (P_6 input). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.P6FormalInput*

The input to P_6 is displayed as a channel stream c or event flow S. If the input is c, then $S = \mathsf{Decode}(c)$. If $c \notin \mathsf{LegalZStream}$, P_6 rejects and produces neither derivation-certificate report nor acceptance-gate report.

Definition 9230.4 (P_6 output). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.P6Output*

The output of P_6 is a report flow or human-readable report containing the decoded event flow, recognized NameCert *flows inherited from P_5, derivation-certificate candidates, recognized derivation-certificate flows, strength candidates, recognized strength flows, acceptance-gate candidates, recognized acceptance-gate flows, accepted object codes, missing fields, missing exactness, missing ledger, missing strength, failure cases, warnings, and cannot-claim annotations.*

9230.2 Strength-flow recognition

Definition 9230.5 (Strength candidate flow). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.Streng*

A strength candidate flow *is an event subflow*

$$s \subseteq S.$$

It is not yet recognized as a strength flow. Explanatory strength levels include seed, paper certificate, checked certificate, and bridge certificate, but these names are not host enums. They must be recognized in event-flow form.

Definition 9230.6 (Generated strength recognizer). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.*

A generated strength recognizer *is an event flow*

$$R_s \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesStrength}(R_s, S, s, \sigma),$$

where s is a subflow of S and σ is a strength role flow.

Definition 9230.7 (Recognized strength flow). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.Recog*

If

$$\mathsf{RecognizesStrength}(R_s, S, s, \sigma),$$

then s is a recognized strength flow *in S. Write*

$$\mathsf{StrengthFlow}_{R_s}(S, s, \sigma).$$

Definition 9230.8 (Strength order witness). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Strengt`

A strength order witness *is an event subflow o recognizing the hierarchy*

$$\text{seed} < \text{paperCert} < \text{checkedCert} < \text{bridgeCert}.$$

Write

$$\mathsf{StrengthOrder}_{R_s}(S, o).$$

The witness is not an external comparison function.

Principle 9230.9 (No strength enum input). *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenerated.no strength enum input`

Strength labels such as seed, paper certificate, checked certificate, and bridge certificate are not host enum inputs. They appear only as recognized strength flows.

Theorem 9230.10 (Strength candidate alone is insufficient). *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenerated. candidate without recognition insufficient`

If s is only a strength candidate and no judgment

$$\mathsf{RecognizesStrength}(R_s, S, s, \sigma)$$

is supplied, then s cannot be used as the strength field of a derivation-certificate flow.

Proof. The strength field of a derivation certificate must be recognized. If s is not recognized, its strength role has no event-flow support. Using it would introduce hidden input. □

9230.3 DerivCert-flow recognition

Definition 9230.11 (DerivCert candidate). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6Deriv`

A DerivCert candidate *is an event subflow*

$$D \subseteq S$$

with a formal event-flow name carrier N and a strength candidate subflow $s \subseteq S$, not yet recognized as a derivation-certificate flow. Write

$$\mathsf{DerivCertCand}(S, D, N, s),$$

where N is a name candidate or licensed name flow and s is a strength candidate or strength flow.

Definition 9230.12 (Generated DerivCert recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerat`

A generated DerivCert recognizer *is an event flow*

$$R_D \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesDerivCert}(R_D, S, D, N, s).$$

This says that R_D recognizes subflow D of S as a derivation-certificate flow for N at strength s.

Definition 9230.13 (Recognized DerivCert flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6`

If

$$\mathsf{DerivCertCand}(S, D, N, s) \quad \textit{and} \quad \mathsf{RecognizesDerivCert}(R_D, D, N, s),$$

then D is a recognized derivation-certificate flow for N and s in S. Write

$$\mathsf{DerivCertFlow}_{R_D}(S, D, N, s).$$

Definition 9230.14 (DerivCert field subflows). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6De`

If

$$\mathsf{RecognizesDerivCert}(R_D, S, D, N, s),$$

then R_D may recognize six fields and a seal:

$$\mathsf{DerivSource}_{R_D}(D, D_0), \quad \mathsf{DerivClassifier}_{R_D}(D, D_1), \quad \mathsf{DerivExactness}_{R_D}(D, D_2),$$

$$\mathsf{DerivLedger}_{R_D}(D, D_3), \quad \mathsf{DerivStability}_{R_D}(D, D_4), \quad \mathsf{DerivStrength}_{R_D}(D, D_5), \quad \mathsf{DerivSeal}_{R_D}(D, \sigma_D).$$

Definition 9230.15 (Complete DerivCert recognition). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerat`

The judgment

$$\mathsf{RecognizesDerivCert}(R_D, S, D, N, s)$$

is complete DerivCert recognition *when there exist*

$$D_0, D_1, D_2, D_3, D_4, D_5, \sigma_D$$

satisfying all field and seal judgments in Definition 9230.14.

Principle 9230.16 (No DerivCert without six fields). *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenerated.p6` `recognized derivcert has six fields`

A derivation-certificate flow must have recognized source, classifier, exactness, ledger, stability, strength, and seal. If any is missing, the flow remains a candidate.

Theorem 9230.17 (Incomplete DerivCert candidate cannot support export). *Lean4 ✓* `BEDC.GroundCompiler.Deri` `incomplete derivcert candidate no export`

If D lacks any recognized source, classifier, exactness, ledger, stability, strength, or seal subflow, then D cannot support export of N at s.

Proof. Derivation-certificate recognition requires complete six-field recognition. Export requires recognized derivation-certificate flow. If any required field is missing, complete recognition fails. □

Corollary 9230.18 (Strength alone cannot support export). *Lean4 ✓* `BEDC.GroundCompiler.DerivCertGenerated.` `strength alone no export`

Even a recognized high-strength flow cannot form a derivation certificate if source, classifier, exactness, ledger, or stability fields are missing.

9230.4 Field soundness

Definition 9230.19 (DerivCert field soundness). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6D`

The six field soundness judgments are:

$$\mathsf{DerivSourceSound}_{R_D}(D, D_0, N), \quad \mathsf{DerivClassifierSound}_{R_D}(D, D_1, N),$$

$$\mathsf{DerivExactnessSound}_{R_D}(D, D_2, N), \quad \mathsf{DerivLedgerSound}_{R_D}(D, D_3, N),$$

$$\mathsf{DerivStabilitySound}_{R_D}(D, D_4, N), \quad \mathsf{DerivStrengthSound}_{R_D}(D, D_5, s).$$

They require the source field to carry N, the classifier field to organize carriers or representatives of N, the exactness field to show that the classifier respects the relevant sameness, the ledger field to retain hidden material and citations, the stability field to cite or supply the relevant stability certificate, and the strength field to match the claimed recognized strength s.

Definition 9230.20 (Sound DerivCert flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6Sound`

A recognized derivation-certificate flow

$$\mathsf{RecognizesDerivCert}(R_D, S, D, N, s)$$

is sound *when it is complete and all six field soundness judgments of Definition 9230.19 hold.*

Theorem 9230.21 (Sound DerivCert flow establishes derivation certificate). *Lean4 ✓* `BEDC.GroundCompiler.Deriv` `sound derivcert flow establishes derivcert`

If D is a sound recognized derivation-certificate flow for N at strength s, then

$$\mathsf{DerivCertFlow}_{R_D}(S, D, N, s)$$

may serve as the derivation-certificate flow for N at s.

Proof. Soundness supplies complete six-field recognition and the soundness condition for each field. This is precisely the flow-level structure required of a derivation certificate. □

9230.5 AcceptGate-flow recognition

Definition 9230.22 (AcceptGate candidate). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Accept`

An AcceptGate candidate *is an event subflow*

$$A \subseteq S$$

claiming to combine a NameCert *flow and a derivation-certificate flow into an acceptance gate, but not yet recognized. Write*

$$\mathsf{AcceptGateCand}(S, A, N, s).$$

Definition 9230.23 (Generated AcceptGate recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenera`

A generated AcceptGate recognizer *is an event flow*

$$R_A \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesAcceptGate}(R_A, S, A, N, s).$$

Definition 9230.24 (Recognized AcceptGate flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.`

If

$$\mathsf{RecognizesAcceptGate}(R_A, S, A, N, s),$$

then A is a recognized acceptance-gate flow for N at s in S. Write

$$\mathsf{AcceptGateFlow}_{R_A}(S, A, N, s).$$

Definition 9230.25 (AcceptGate components). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.Acce`

If

$$\mathsf{RecognizesAcceptGate}(R_A, S, A, N, s),$$

then R_A identifies

$$\mathsf{GateNameCert}_{R_A}(A, C_N), \qquad \mathsf{GateDerivCert}_{R_A}(A, D_N),$$

$$\mathsf{GateStrength}_{R_A}(A, s), \qquad \mathsf{GateSeal}_{R_A}(A, \sigma_A).$$

Principle 9230.26 (No AcceptGate without NameCert and DerivCert). *Lean4✓* `BEDC.GroundCompiler.DerivCert` `acceptgate components from recognition`

An acceptance-gate flow must contain a recognized NameCert *flow and a sound recognized derivation-certificate flow. If either is missing, the flow is only an acceptance-gate candidate.*

Theorem 9230.27 (AcceptGate recognition requires NameCert and DerivCert). *Lean4✓* `BEDC.GroundCompiler.D` `acceptgate requires namecert derivcert`

If

$$\mathsf{AcceptGateFlow}_{R_A}(S, A, N, s),$$

then there exist C_N and D_N such that

$$\mathsf{NameCertFlow}(C_N, N) \qquad \textit{and} \qquad \mathsf{DerivCertFlow}(D_N, N, s).$$

Proof. By the component definition, recognized acceptance-gate flow identifies NameCert and derivation-certificate components. By the acceptance-gate principle, these components must be recognized flows. □

Corollary 9230.28 (NameCert alone does not imply AcceptGate). *Lean4✓* `BEDC.GroundCompiler.DerivCertGene` `alone no acceptgate`

There exists a recognized NameCert *flow whose same event-flow carrier is not an accepted flow. Licensing a name at P_5 therefore does not by itself supply the derivation-certificate flow, gate seal, and gate recognition required for an acceptance gate.*

Corollary 9230.29 (DerivCert alone does not imply AcceptGate). *Lean4✓* `BEDC.GroundCompiler.DerivCertGene` `alone no acceptgate`

There exists a recognized derivation-certificate flow whose same event-flow carrier is not an accepted flow. Derivation-certificate recognition alone therefore does not supply the matching NameCert *flow, gate seal, and gate recognition required for an acceptance gate.*

9230.6 Accepted object code

Definition 9230.30 (Accepted object flow). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6Accep`

If

$$\mathsf{RecognizesAcceptGate}(R_A, S, A, N, s)$$

and A is a sound recognized acceptance-gate flow, then A is an accepted object flow *for N at s. Write*

$$\mathsf{AcceptedObjectFlow}(A, N, s).$$

Definition 9230.31 (Accepted object code). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenerated.P6Accep`

If

$$\mathsf{AcceptedObjectFlow}(A, N, s),$$

define

$$\mathsf{AcceptedObjectCode}(A, N, s) := \mathsf{Compile}(A).$$

This is the channel code of the acceptance flow. It is not bare object code, object equality, theorem proof, or bridge certificate.

Theorem 9230.32 (Accepted object code is injective on raw acceptance flows). *Lean4✓* `BEDC.GroundCompiler.Der` `accepted object code injective raw flows`

If

$$\mathsf{AcceptedObjectCode}(A, N, s) = \mathsf{AcceptedObjectCode}(B, M, t),$$

then $A = B$.

Proof. By definition both sides are compiled channel codes:

$$\mathsf{Compile}(A) = \mathsf{Compile}(B).$$

The channel compiler is injective on event flows, so $A = B$. □

Remark 9230.33 (Accepted object code is not classified-object identity). *The injectivity theorem concerns raw acceptance flows and their codes. If a later classifier or bridge identifies two accepted flows as the same mature object, the mature-object layer is a quotient and may be many-to-one.*

Corollary 9230.34 (Accepted code equality is stronger than object equivalence). *Lean4✓* `BEDC.GroundCompiler.D` `accepted code equality stronger than equivalence`

Equal accepted object codes imply equal raw acceptance flows. Equivalent mature objects under classifier or bridge need not have equal accepted object codes.

9230.7 Strength monotonicity and promotion

Definition 9230.35 (Flow-level strength monotonicity). *Lean4 (definition)* `BEDC.GroundCompiler.DerivCertGenera`

Flow-level strength monotonicity is the rule that, when an accepted flow at strength s contains a recognized lower-strength derivation-certificate subflow supporting s', P_6 may report

$$\mathsf{AcceptGateFlow}(N, s) \Rightarrow \mathsf{AcceptGateFlow}(N, s')$$

provided that the higher-strength derivation-certificate flow contains a recognized lower-strength derivation-certificate subflow supporting s'.

Theorem 9230.36 (Strength monotonicity). *Lean4 ✓ BEDC.GroundCompiler.DerivCertGenerated.p6 strength monotonicity*

If AcceptGateFlow(N, s) *holds and the accepted flow contains a recognized lower-strength derivation-certificate subflow supporting* s'*, then* AcceptGateFlow(N, s') *holds.*

Proof. The acceptance gate at s contains a NameCert flow and a derivation-certificate flow at s. The lower-strength subflow supplies the derivation-certificate component at s'. Together with the same NameCert flow, it forms an acceptance gate at s'. □

Definition 9230.37 (Upgrade flow). *Lean4 (definition) BEDC.GroundCompiler.DerivCertGenerated.UpgradeFlow*

An upgrade flow *is an event flow* U *that upgrades object* N *from strength* s *to* $s' > s$*. It contains prior acceptance-gate flow at* s*, new obligations for* s'*, new proof, exactness, bridge, or checked-target flow, new ledger, and upgrade seal. Write*

$$\mathsf{UpgradeFlow}(U, N, s, s').$$

Principle 9230.38 (No retroactive promotion). *Lean4 ✓ BEDC.GroundCompiler.DerivCertGenerated.p6 no retroactive promotion*

An object accepted at s *does not become accepted at* $s' > s$ *merely because a dependency is later upgraded. A recognized upgrade flow is required.*

Theorem 9230.39 (Upgrade requires upgrade flow). *Lean4 ✓ BEDC.GroundCompiler.DerivCertGenerated.upgrade requires upgrade flow*

If UpgradeFlow(U, N, s, s') *holds, then the flow contains* AcceptGateFlow(N, s)*, a recognized strength flow for* s'*, and recognized derivation-certificate and seal subflows inside* U *supporting* N *at* s'*.*

Proof. This is the defining projection from the upgrade-flow predicate: the prior acceptance gate, new strength flow, and new derivation-certificate and seal subflows are conjunctive components of UpgradeFlow(U, N, s, s'). □

9230.8 Reports and soundness

Definition 9230.40 (P_6 report). *Lean4 (definition) BEDC.GroundCompiler.DerivCertReports.P6Report*

A P_6 report *is an output view containing decoded event flow, recognized* NameCert *flows, derivation-certificate candidates, recognized derivation-certificate flows, derivation-certificate field status, strength-flow status, acceptance-gate candidates, recognized acceptance-gate flows, accepted object flows, accepted object codes, missing fields, upgrade status, warnings, and cannot-claim annotations.*

Definition 9230.41 (DerivCert and AcceptGate field status). *Lean4 (definition) BEDC.GroundCompiler.DerivCertRe*

For each derivation-certificate candidate D*, the report records source, classifier, exactness, ledger, stability, strength, and seal status as present, missing, or candidate-only. For each acceptance-gate candidate* A*, it records* NameCert*-flow, derivation-certificate flow, strength flow, and gate-seal status as present, missing, or candidate-only.*

Definition 9230.42 (P_6 cannot-claim annotations). *Lean4 (definition) BEDC.GroundCompiler.DerivCertReports.Co*

A P_6 *report must state that* NameCert *flow alone does not imply accepted export, derivation-certificate flow alone does not imply accepted export, strength flow alone does not imply derivation*

certificate, accepted flow code is not object equality, accepted object code is not proof of later theorems, accepted object code is not bridge certificate, retroactive promotion is not allowed, classifier equivalence may identify distinct accepted flows, and report is not proof unless theorem-flow recognition is supplied separately.

Definition 9230.43 (Sound P_6 report)**.** Lean4 (definition) `BEDC.GroundCompiler.DerivCertReports.SoundP6Report`

A P_6 report is sound *when:*

1. *every recognized derivation-certificate flow is backed by generated derivation-certificate recognizer;*
2. *every recognized derivation-certificate flow has six fields and seal;*
3. *every recognized derivation-certificate flow satisfies field soundness;*
4. *every recognized strength flow is backed by generated strength recognizer;*
5. *every recognized acceptance-gate flow contains matching* NameCert *and derivation-certificate flows;*
6. *every accepted object code comes from recognized acceptance-gate flow;*
7. *missing fields are explicitly reported;*
8. *candidate-only entries are not reported as recognized;*
9. *cannot-claim annotations are complete.*

Theorem 9230.44 (Sound P_6 report theorem)**.** Lean4✓ `BEDC.GroundCompiler.DerivCertReports.sound p6 report`

If P_6 outputs a sound report, then every accepted object code in the report corresponds to a recognized acceptance-gate flow containing sound NameCert *and derivation-certificate flows.*

Proof. By report soundness, accepted object code can come only from recognized acceptance-gate flow. Such a gate contains matching NameCert and derivation-certificate flows, and the derivation-certificate flow has all six fields and field soundness. □

Corollary 9230.45 (Sound accepted code is certificate-mediated)**.** Lean4✓ `BEDC.GroundCompiler.DerivCertRepor accepted code certificate mediated`

Every sound P_6 accepted object code is certificate-mediated, not bare-code export.

Theorem 9230.46 (P_6 code existence still does not imply theoremhood)**.** Lean4✓ `BEDC.GroundCompiler.DerivCer code not theoremhood`

In a sound P_6 report, accepted object code is recorded together with the cannot-claim annotation that accepted object code is not proof of later theorems. Thus no theorem about the object follows unless a recognized theorem flow and proof flow are supplied.

Proof. Accepted object code records that an acceptance gate has been recognized for an object candidate at a strength. Theoremhood requires theorem-flow recognition with statement, proof, dependencies, certificates, ledger, status, and canonical site. The accepted object code does not provide those data. □

9230.9 Adequacy and examples

Definition 9230.47 (P_6 audit checklist)**.** Lean4 (definition) BEDC.GroundCompiler.DerivCertReports.P6AuditCheck

A P_6 prototype must satisfy:

1. *all P_5 audit requirements;*
2. *derivation-certificate candidates are distinguished from recognized derivation-certificate flows;*
3. *recognized derivation-certificate flows require generated recognizers;*
4. *six derivation-certificate fields and seal are required;*
5. *field soundness is required;*
6. *strength candidates are distinguished from recognized strength flows;*
7. *strength recognition requires generated recognizer;*
8. *acceptance-gate candidates are distinguished from recognized acceptance-gate flows;*
9. *matching* NameCert *and derivation-certificate flows are required for acceptance gates;*
10. *accepted object code comes only from recognized acceptance-gate flow;*
11. *missing fields are reported explicitly;*
12. *accepted code is distinguished from object equality;*
13. *no retroactive promotion is recorded;*
14. *theoremhood is not inferred from accepted object code;*
15. *cannot-claim annotations are included.*

Theorem 9230.48 (P_6 adequacy theorem)**.** Lean4 ✓ BEDC.GroundCompiler.DerivCertReports.p6 adequacy

If a prototype satisfies the P_6 audit checklist, uses sound P_5 reports, and every recognized derivation-certificate or acceptance-gate flow is backed by generated recognizers and field soundness, then it is adequate as a DerivCert and AcceptGate recognition prototype.

Proof. The checklist enforces separation between candidates and recognized flows, six-field derivation-certificate recognition, strength recognition, matching NameCert and derivation-certificate components for acceptance gates, and cannot-claim discipline. Sound P_5 reports supply NameCert correctness. Generated recognizers and field soundness supply the certificate support. □

Corollary 9230.49 (P_6 adequacy does not imply higher adequacy)**.** Lean4 ✓ BEDC.GroundCompiler.DerivCertRepo adequacy does not imply higher

A P_6 prototype can report accepted object codes, but it is not yet a theorem-flow recognizer, chapter-flow recognizer, bridge certifier, full proof checker, or self-hosting compiler.

Example 9230.50 (NameCert without DerivCert)**.** *If a report has a recognized and sound* NameCert *flow, but missing derivation-certificate and acceptance-gate flows, it may say that the name is licensed at P_5 but that there is no accepted object at P_6.*

Example 9230.51 (DerivCert candidate missing exactness)**.** *If a derivation-certificate candidate has source, classifier, ledger, stability, strength, and seal fields, but exactness is missing, the report says that derivation-certificate recognition fails, acceptance-gate recognition fails, and no accepted object code is produced.*

Example 9230.52 (Accepted Axis-like object at seed strength)**.** *Suppose the report recognizes a sound Axis-like* NameCert *flow, a sound derivation-certificate flow with source, classifier, exactness, ledger, stability, seed strength, and seal, and an acceptance-gate flow combining them. Then P_6 may report*

$$\mathsf{AcceptedObjectCode}(\mathrm{Axis}, \mathrm{seed}) = \mathsf{Compile}(\textit{acceptance-gate flow}).$$

Cannot-claims state that seed strength does not imply checked certificate or bridge certificate, and accepted code is not equality with another Axis presentation.

Example 9230.53 (FullAxis candidate still not accepted)**.** *For the completion skeleton* 0 | 00 | 000 | 01 | 011 | 100*, suppose the* NameCert *candidate has source and pattern present, classifier candidate, stability missing, and ledger present, while derivation certificate and acceptance gate are missing. The report must say that* $\mathsf{FullAxis}^{\uparrow}$ *is not accepted: missing stability blocks* NameCert *recognition, and missing derivation certificate blocks acceptance.*

Example 9230.54 (No retroactive promotion)**.** *If an object N is accepted at seed strength and a dependency is later upgraded to checked-certificate strength, P_6 must still report N as accepted only at seed unless there is a recognized upgrade flow for N from seed to checked-certificate strength.*

9230.10 Conservativity

Theorem 9230.55 (P_6 is conservative over the finite kernel)**.** *Lean4✓ BEDC.GroundCompiler.DerivCertReports.p6 conservative over finite kernel*

Adding derivation-certificate recognition, strength recognition, acceptance-gate recognition, accepted object code reports, and upgrade-flow reports adds no raw marks, history constructors, sameness rules, continuation rules, theorem rules, bridge rules, or object-equality rules.

Proof. P_6 operates on decoded event flows and generated recognition relations. It generates no new b0, b1, adds no msame, hsame, or psame, does not alter Ext or Cont, and does not create kernel equality. Accepted object code is mediated by NameCert and derivation-certificate flows; it does not introduce mature objects as primitives. □

Corollary 9230.56 (P_6 is accepted-object reporting, not proof engine)**.** *Lean4✓ BEDC.GroundCompiler.DerivCertI not proof engine*

P_6 can report accepted object codes, but it does not prove arbitrary theorems about those objects. Proofs require theorem-flow and proof-flow recognition.

Remark 9230.57 (Summary)**.** *This chapter defines P_6, the DerivCert and AcceptGate recognition prototype. It adds strength candidates, generated strength recognizers, recognized strength flows, derivation-certificate candidates, generated derivation-certificate recognizers, recognized derivation-certificate flows, six derivation-certificate fields, field soundness, acceptance-gate candidates, generated acceptance-gate recognizers, accepted object flows, accepted object codes, strength monotonicity, upgrade flows, no-retroactive-promotion discipline, P_6 reports, and the P_6 audit checklist. It proves that strength candidates and incomplete derivation certificates cannot support export, sound derivation certificates establish derivation-certificate flows, acceptance gates require* NameCert *and derivation-certificate flows, accepted object code is injective only on raw acceptance flows, accepted code is not object identity or theoremhood, upgrade requires upgrade flow, sound P_6 reports are certificate-mediated, and P_6 is conservative over the finite kernel.*

Chapter 9231

Theorem and Proof Recognition Prototype

chapter 9230 defines P_6, the derivation-certificate and acceptance-gate recognition prototype. It can report accepted object code, but it is still not a theorem recognizer. An accepted object does not automatically provide arbitrary theorems about that object.

This chapter defines the next prototype level:

$$\boxed{P_7 = P_6 + \text{statement-flow recognition} + \text{proof-flow recognition} + \text{theorem-flow recognition.}}$$

This is the first layer where theorem code may be reported. The distinctions among theorem code, proof-flow, accepted theorem, and abstract proposition are strict:

a theorem is not an external label; it is a recognized statement/proof/dependency/certificate/ledger/status/site

Thus P_7 does not accept theorem objects, theorem labels, Lean theorem names, host proof objects, proof ASTs, external proposition strings, or manual theorem tables. It accepts channel streams or decoded event flows and recognizes theorem and proof flows by generated recognizers and generated proof checkers.

9231.1 P_7 prototype interface

Definition 9231.1 (Theorem and proof recognition prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremPr`

A theorem and proof recognition prototype *is an executable prototype P_7 satisfying:*

1. *P_7 includes all P_6 functions;*
2. *P_7 recognizes statement flows;*
3. *P_7 recognizes dependency flows;*
4. *P_7 recognizes proof flows;*
5. *P_7 recognizes theorem certificate flows;*
6. *P_7 recognizes theorem ledger flows;*
7. *P_7 recognizes theorem status flows;*
8. *P_7 recognizes canonical-site flows;*

9. *P_7 recognizes theorem-seal flows;*
10. *when proof-flow is validated by generated proof checker, P_7 can report theorem code;*
11. *P_7 does not treat theorem code as mature-object export;*
12. *P_7 does not treat theorem code as abstract-proposition identity;*
13. *P_7 does not treat theorem code as bridge certificate.*

Write

$$\mathsf{TheoremProofPrototype}(P_7).$$

Principle 9231.2 (Theorem is not input). *Lean4✓ BEDC.GroundCompiler.TheoremProofPrototype.theorem proof objects not formal*

There is no formal input

$$\mathsf{TheoremObject} \to \mathsf{Code} \qquad \textit{or} \qquad \mathsf{ProofObject} \to \mathsf{Code}.$$

The only formal input remains EventFlow *or* LegalZStream*. If the input is channel stream c, then P_7 first checks $c \in \mathsf{LegalZStream}$ and forms $S = \mathsf{Decode}(c)$.*

Definition 9231.3 (P_7 input). *Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.P7FormalInput*

The input to P_7 is displayed as channel stream c or event flow S. If $c \notin \mathsf{LegalZStream}$, P_7 rejects and produces no theorem report.

Definition 9231.4 (P_7 output). *Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.P7Output*

The output of P_7 is a report flow or human-readable report containing decoded event flow, statement candidates, recognized statement flows, dependency candidates, recognized dependency flows, proof candidates, recognized proof flows, proof-checking results, theorem candidates, recognized theorem flows, theorem code, theorem field status, missing fields, proof failures, certificate failures, ledger failures, status failures, canonical-site failures, warnings, and cannot-claim annotations.

9231.2 Statement and dependency recognition

Definition 9231.5 (Statement candidate). *Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.Statem*

A statement candidate *is an event subflow*

$$\Phi \subseteq S$$

not yet recognized as a statement flow. Write

$$\mathsf{StmtCand}(S, \Phi).$$

Definition 9231.6 (Generated statement recognizer). *Lean4 (definition) BEDC.GroundCompiler.TheoremProofProto*

A generated statement recognizer *is an event flow*

$$R_\Phi \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesStatement}(R_\Phi, S, \Phi).$$

Definition 9231.7 (Recognized statement flow)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.`

If

$$\mathsf{RecognizesStatement}(R_\Phi, S, \Phi),$$

then Φ *is a recognized statement flow in* S*. Write*

$$\mathsf{StatementFlow}_{R_\Phi}(S, \Phi).$$

Remark 9231.8 (Statement flow is not abstract proposition)**.** *A statement flow is a statement structure inside an event flow. It is not an external proposition, host-level formula, Lean theorem statement, or LaTeX string. One abstract proposition may have multiple statement flows with different presentations, unfoldings, dependency exposure, notation, bridge expression, or canonical site.*

Definition 9231.9 (Dependency candidate and recognizer)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProof`

A dependency candidate *is an event subflow*

$$D_{\mathrm{dep}} \subseteq S$$

claiming to record theorem dependencies. A generated dependency recognizer *is an event flow* R_{dep} *supporting*

$$\mathsf{RecognizesDependencies}(R_{\mathrm{dep}}, S, D_{\mathrm{dep}}).$$

If this holds, D_{dep} *is a recognized dependency flow.*

Definition 9231.10 (Dependency soundness)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.De`

A dependency flow D_{dep} *is* sound *when every raw dependency entry it records has a dependency status among: generated in an earlier event flow, proved in an earlier theorem flow, recognized in an earlier certificate flow, local to the same theorem flow, explicitly marked as deferred obligation, explicitly listed as cannot-claim, or explicitly listed as open obligation. A dependency entry whose status is future-ungenerated is unsound.*

Theorem 9231.11 (Unsound dependency blocks theorem recognition)**.** *Lean4 ✓* `BEDC.GroundCompiler.TheoremPr` `dependency blocks theorem recognition`

If a theorem candidate has a recognized dependency subflow whose dependency soundness predicate fails, then that candidate cannot become a sound theorem flow with that dependency subflow.

Proof. The sound theorem-flow predicate includes dependency soundness for the recognized dependency subflow. If dependency soundness fails for that subflow, the conjunction required for sound theorem recognition cannot be constructed. □

9231.3 Proof-flow recognition and checking

Definition 9231.12 (Proof candidate)**.** *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.ProofCand`

A proof candidate *is an event subflow*

$$\Pi \subseteq S$$

not yet recognized as a proof flow. Write

$$\mathsf{ProofCand}(S, \Pi).$$

Definition 9231.13 (Generated proof-flow recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofProt`

A generated proof-flow recognizer *is an event flow*

$$R_\Pi \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesProofFlow}(R_\Pi, S, \Pi).$$

Definition 9231.14 (Recognized proof flow). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.Rec`

If

$$\mathsf{RecognizesProofFlow}(R_\Pi, S, \Pi),$$

then Π *is a recognized proof flow in* S*. Write*

$$\mathsf{ProofFlow}_{R_\Pi}(S, \Pi).$$

Definition 9231.15 (Generated proof checker). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.G`

A generated proof checker *is an event flow*

$$R_{\mathrm{chk}} \in \mathsf{EventFlow}$$

supporting

$$\mathsf{ChecksProof}(R_{\mathrm{chk}}, \Phi, D_{\mathrm{dep}}, \Pi).$$

This says that R_{chk} *checks proof-flow* Π *as a proof of statement-flow* Φ *using dependency-flow* D_{dep}*. It is not host proof checker output or external proof-engine result. Host proof checkers must be recorded by bootstrap obligation or bridge/certificate flow.*

Definition 9231.16 (Proof soundness). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.ProofSou`

A proof flow Π *is* sound *for statement flow* Φ *and dependency flow* D_{dep} *when there exists a generated proof checker* R_{chk} *such that*

$$\mathsf{ChecksProof}(R_{\mathrm{chk}}, \Phi, D_{\mathrm{dep}}, \Pi).$$

Theorem 9231.17 (Proof candidate alone is insufficient). *Lean4✓* `BEDC.GroundCompiler.TheoremProofPrototype.candidate alone insufficient`

If Π *is a proof candidate in an event flow* S*, no generated proof-flow recognizer recognizes* Π *in* S*, and no generated proof checker establishes*

$$\mathsf{ChecksProof}(R_{\mathrm{chk}}, \Phi, D_{\mathrm{dep}}, \Pi),$$

then Π *cannot be used as a theorem proof for statement flow* Φ *and dependency flow* D_{dep}*.*

Proof. A proof candidate is just an event subflow. To serve as theorem proof, it must first be recognized as proof flow and then checked by generated proof checker against the statement and dependencies. Without both, the proof role is not established. □

Corollary 9231.18 (Code is not proof flow). *Lean4✓* `BEDC.GroundCompiler.TheoremProofPrototype.code not proof flow`

A channel code may decode to an event flow, but it does not thereby become proof flow. Proof-flow recognition is required.

Corollary 9231.19 (Proof flow is not proof checking). *Lean4✓* `BEDC.GroundCompiler.TheoremProofPrototype.pro flow not checking`

Recognized proof flow does not by itself prove a statement. A ChecksProof *judgment is required.*

9231.4 Theorem-flow recognition

Definition 9231.20 (Theorem candidate). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.Theor`

A theorem candidate *is an event subflow*

$$T \subseteq S$$

not yet recognized as theorem flow. Write

$$\mathsf{TheoremCand}(S, T).$$

Definition 9231.21 (Generated theorem recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofProtot`

A generated theorem recognizer *is an event flow*

$$R_T \in \mathsf{EventFlow}$$

supporting

$$\mathsf{RecognizesTheorem}(R_T, S, T).$$

Definition 9231.22 (Theorem subflows). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.P7Theor`

If

$$\mathsf{RecognizesTheorem}(R_T, S, T),$$

then R_T may identify:

$$\mathsf{ThmStatement}_{R_T}(T, \Phi), \qquad \mathsf{ThmDependencies}_{R_T}(T, D_{\mathrm{dep}}),$$

$$\mathsf{ThmProof}_{R_T}(T, \Pi), \qquad \mathsf{ThmCertificates}_{R_T}(T, C_{\mathrm{thm}}),$$

$$\mathsf{ThmLedger}_{R_T}(T, L_{\mathrm{thm}}), \qquad \mathsf{ThmStatus}_{R_T}(T, s_{\mathrm{thm}}),$$

$$\mathsf{ThmCanonicalSite}_{R_T}(T, \kappa), \qquad \mathsf{ThmSeal}_{R_T}(T, \sigma_T).$$

Definition 9231.23 (Complete theorem recognition). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofProtot`

The judgment

$$\mathsf{P7CompleteTheoremRecognition}(R_T, T)$$

is complete theorem recognition *when statement, dependency, proof, certificate, ledger, status, canonical-site, and seal subflows are all recognized.*

Principle 9231.24 (No theorem without eight subflows). *Lean4✓* `BEDC.GroundCompiler.TheoremProofPrototype.theorem without eight subflows`

A theorem flow must have recognized statement, dependencies, proof, certificates, ledger, status, canonical site, and seal. If any is missing, the flow remains a theorem candidate.

Theorem 9231.25 (Incomplete theorem candidate cannot be theorem flow). *Lean4✓* `BEDC.GroundCompiler.Theor incomplete theorem candidate not theorem`

If a theorem candidate T lacks any required recognized subflow, then T cannot be used as recognized theorem flow.

Proof. Theorem-flow recognition requires complete theorem recognition. Missing any of the eight required subflows blocks completeness. □

9231.5 Theorem-flow soundness and code

Definition 9231.26 (Theorem-flow soundness components). Lean4 (definition) BEDC.GroundCompiler.TheoremProof

The soundness components for a theorem flow are:

1. *statement soundness: the statement is recognized and every symbol, dependency, and object reference has a source in the dependency or certificate flow;*
2. *dependency soundness: the dependency flow satisfies Definition 9231.10;*
3. *proof soundness: a generated proof checker validates* $\mathsf{ChecksProof}(R_{\mathrm{chk}}, \Phi, D_{\mathrm{dep}}, \Pi)$*;*
4. *certificate soundness: required certificate flows are recognized or explicitly marked as open obligations;*
5. *ledger soundness: the ledger records statement, dependency, proof, certificate, hidden abbreviation, classifier quotient, normal-address, bridge-obligation, and failure-branch material;*
6. *status soundness: status is recognized and matches proof, certificate, checked-flow, or bridge-flow status;*
7. *canonical-site soundness: the site records the theorem's canonical position, with competing sites marked as variant, wrapper, or projection.*

Definition 9231.27 (Sound theorem flow). Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.P7Sou

A recognized theorem flow T is sound *when complete theorem recognition holds and all soundness components of Definition 9231.26 hold.*

Theorem 9231.28 (Sound theorem flow establishes theorem code). Lean4✓ BEDC.GroundCompiler.TheoremProofP theorem flow establishes theorem code

If T is a sound theorem flow, define

$$\mathsf{TheoremCode}(T) := \mathsf{Compile}(T).$$

This code is the lossless channel encoding of T.

Proof. The theorem flow T is an event-flow subflow, so $\mathsf{Compile}(T)$ is defined. By the channel bijection, compilation is lossless. By soundness, T has the statement, dependencies, proof, certificates, ledger, status, site, and seal required for theoremhood. □

Theorem 9231.29 (Theorem code is injective on raw theorem flows). Lean4✓ BEDC.GroundCompiler.TheoremProof theorem code injective

If T and U are sound theorem flows and

$$\mathsf{TheoremCode}(T) = \mathsf{TheoremCode}(U),$$

then $T = U$.

Proof. Both theorem codes are compiled event-flow codes. Equality gives $\mathsf{Compile}(T) = \mathsf{Compile}(U)$, hence $T = U$ by injectivity of the channel compiler. □

Corollary 9231.30 (Theorem code equality is stronger than statement equivalence). Lean4✓ BEDC.GroundCompiler.TheoremProofPrototype.theorem code stronger than statement

Equal theorem codes imply equal complete theorem flows, including statement, dependencies, proof, certificates, ledger, status, and canonical site. Statement-flow equivalence alone does not imply theorem-code equality.

9231.6 Abstract propositions and accepted theorem flows

Definition 9231.31 (Statement classifier and abstract proposition class). *Lean4 (definition)* `BEDC.GroundCompiler.`

A statement classifier *is a source-level classifier* $\sim_{\mathrm{stmt}}$ *on statement flows. If* $\Phi \sim_{\mathrm{stmt}} \Psi$*, then* Φ *and* Ψ *may be read as statement-flow presentations of the same abstract proposition. The abstract proposition class is*

$$[\Phi]_{\sim_{\mathrm{stmt}}}.$$

Proposition 9231.32 (Abstract proposition may have many theorem flows). *Lean4 ✓* `BEDC.GroundCompiler.Theo` `proposition many flows`

There may be distinct theorem flows T *and* U *whose statement flows lie in the same abstract proposition class.*

Proof. Let the statement flows of T and U be Φ and Ψ, with $\Phi \sim_{\mathrm{stmt}} \Psi$. If proof flow, dependency flow, ledger flow, or canonical-site flow differs, then the complete event flows T and U differ. □

Corollary 9231.33 (Abstract proposition is not theorem code). *Lean4 ✓* `BEDC.GroundCompiler.TheoremProofProt` `proposition not theorem code`

Theorem code is bijective with theorem flow. Abstract proposition classes are quotients of statement flows and are generally not in bijection with theorem codes.

Definition 9231.34 (Accepted theorem flow). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.P7`

A sound theorem flow T *is an* accepted theorem flow *when its proof flow is checked by generated proof checker, all required certificates in its certificate flow are recognized, any claimed mature-object export has the relevant acceptance-gate flow, status matches proof and certificate state, and ledger is complete. Write*

$$\mathsf{AcceptedTheoremFlow}(T).$$

Definition 9231.35 (Accepted theorem code). *Lean4 (definition)* `BEDC.GroundCompiler.TheoremProofPrototype.Ac`

If

$$\mathsf{AcceptedTheoremFlow}(T),$$

define

$$\mathsf{AcceptedTheoremCode}(T) := \mathsf{Compile}(T).$$

Proposition 9231.36 (Accepted theorem code is theorem-flow code with acceptance status). *Lean4 ✓* `BEDC.GroundCompiler.TheoremProofPrototype.accepted theorem code same code`

Accepted theorem code is not a new encoding function. It is theorem-flow code for a theorem flow satisfying accepted-theorem conditions.

Proof. By definition, $\mathsf{AcceptedTheoremCode}(T) = \mathsf{Compile}(T)$. □

Theorem 9231.37 (Accepted theorem code does not imply accepted object). *Lean4 ✓* `BEDC.GroundCompiler.Theo` `theorem not object acceptance distinct`

There exists an accepted theorem flow T *and an object-flow mention* O *inside* T *such that* O *is not an accepted object flow. Thus accepted-theorem status is a status of the theorem flow and does not by itself provide the acceptance-gate flows required for every object mentioned by the theorem.*

Proof. Accepted theorem status concerns the theorem package: proof, certificates, ledger, status, and site. Object acceptance is separately mediated by acceptance-gate flows for the objects. A theorem flow may mention an empty object-flow position; the empty flow has no accepted-object witness, because accepted object flow requires nonempty accepted structure and recognized certificate flows. □

Corollary 9231.38 (Theorem acceptance and object acceptance are distinct). Lean4 ✓ BEDC.GroundCompiler.Theo acceptance and object acceptance distinct

Theorem-flow acceptance and object acceptance are different statuses requiring different recognized flows.

9231.7 Reports and soundness

Definition 9231.39 (P_7 report). Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.P7ReportDatum

A P_7 report *is an output view containing decoded event flow, statement candidates, recognized statement flows, dependency candidates, recognized dependency flows, proof candidates, recognized proof flows, proof-checking status, theorem candidates, recognized theorem flows, theorem field status, accepted theorem flows, theorem codes, missing fields, proof failures, dependency failures, certificate failures, ledger failures, status failures, canonical-site failures, and cannot-claim annotations.*

Definition 9231.40 (Theorem field and proof status). Lean4 (definition) BEDC.GroundCompiler.TheoremProofProto

For each theorem candidate T*, the report records statement, dependencies, proof, certificates, ledger, status, canonical site, and seal as present, missing, or candidate-only. For each proof candidate* Π*, it records whether proof-flow is recognized, statement and dependencies match, proof checker is supplied, proof check passes, failure reason, and ledger status.*

Definition 9231.41 (P_7 cannot-claim annotations). Lean4 (definition) BEDC.GroundCompiler.TheoremProototy

A P_7 *report must state that statement flow alone is not theorem, proof flow alone is not theorem, recognized proof flow is not checked proof unless proof checker validates it, theorem code is not abstract-proposition identity, accepted theorem does not imply all mentioned objects are accepted, theorem code is not bridge certificate, same statement classifier does not imply same theorem code, missing proof checker blocks accepted theorem, and missing canonical site blocks canonical theorem code.*

Definition 9231.42 (Sound P_7 report). Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.SoundP7R

A P_7 *report is* sound *when:*

1. *every recognized statement flow is backed by generated statement recognizer;*
2. *every recognized dependency flow is backed by generated dependency recognizer;*
3. *every recognized proof flow is backed by generated proof-flow recognizer;*
4. *every successful proof-checking claim is backed by generated proof checker;*
5. *every recognized theorem flow has eight subflows;*
6. *every accepted theorem flow satisfies accepted-theorem conditions;*
7. *every theorem code is* $\mathsf{Compile}(T)$*;*
8. *candidate-only entries are not reported as recognized;*
9. *missing fields and failure cases are explicit;*
10. *cannot-claim annotations are complete.*

Theorem 9231.43 (Sound P_7 report theorem). Lean4 ✓ BEDC.GroundCompiler.TheoremProofPrototype.sound p7 report theorem

If P_7 *outputs a sound report, then every theorem code in the report corresponds to a sound theorem flow, and every accepted theorem code corresponds to an accepted theorem flow.*

Proof. By soundness, theorem code comes only from recognized theorem flow with all required subflows and soundness data. Accepted theorem code further requires accepted-theorem conditions. □

Corollary 9231.44 (Sound theorem code is flow-mediated). Lean4✓ BEDC.GroundCompiler.TheoremProofPrototyp theorem code flow mediated

P_7 theorem code is not an external theorem label; it is the channel compilation of theorem flow.

Theorem 9231.45 (P_7 code existence still does not imply bridge). Lean4✓ BEDC.GroundCompiler.TheoremProofPr code existence still not bridge

There exists an accepted theorem flow whose accepted theorem code is legal theorem code and whose theorem flow contains no bridge-certificate witness flow.

Proof. Accepted theorem flow may have non-bridge status. Bridge certification requires additional bridge-certificate flow. A singleton accepted theorem flow has legal theorem code, but it cannot contain the designated bridge-certificate witness flow. □

9231.8 Adequacy and examples

Definition 9231.46 (P_7 audit checklist). Lean4 (definition) BEDC.GroundCompiler.TheoremProofPrototype.P7Audit

A P_7 prototype must satisfy:

1. *all P_6 audit requirements;*
2. *statement candidates are distinguished from recognized statement flows;*
3. *proof candidates are distinguished from recognized proof flows;*
4. *proof-checking claims require generated proof checker;*
5. *theorem candidates are distinguished from recognized theorem flows;*
6. *eight theorem subflows are required;*
7. *dependency soundness is required;*
8. *proof soundness is required;*
9. *certificate soundness is required;*
10. *ledger soundness is required;*
11. *status soundness is required;*
12. *canonical-site soundness is required;*
13. *theorem code is distinguished from abstract proposition;*
14. *theorem acceptance is distinguished from object acceptance;*
15. *missing fields and proof failures are recorded;*
16. *cannot-claim annotations are included.*

Theorem 9231.47 (P_7 adequacy theorem). Lean4✓ BEDC.GroundCompiler.TheoremProofPrototype.p7 adequacy

If a prototype satisfies the P_7 audit checklist, then it is adequate as a theorem and proof recognition prototype: recognized statement entries are backed by generated statement recognizers, recognized theorem entries are backed by generated theorem recognizers, and theorem-code entries are mediated by theorem flows rather than by external theorem labels.

Proof. The audit checklist includes the backing obligations for recognized statement entries, recognized theorem entries, and theorem-code entries. Projecting those obligations yields the adequacy clauses for the prototype. □

Corollary 9231.48 (P_7 adequacy does not imply higher adequacy)**.** Lean4✓ BEDC.GroundCompiler.TheoremProofP adequacy does not imply higher

P_7 recognizes theorem and proof flows, but does not yet recognize chapter flows, compiler self-hosting, full bridge protocols, or global manuscript status.

Example 9231.49 (Channel round-trip theorem flow)**.** *A theorem flow for channel round trip may contain a statement flow* $\mathsf{Decode}(\mathsf{Compile}(S)) = S$*, dependency flows for* ζ*, event encoding, compilation, and decoding, proof flow by induction on event flow, certificate flow for channel compiler definitions and decoder correctness, ledger retaining raw event flow, compiled stream, and decoded stream, status flow, canonical site in the channel compiler chapter, and theorem seal. Its theorem code is* $\mathsf{Compile}(T_{\text{roundtrip}})$*.*

Example 9231.50 (Carry theorem flow)**.** *A theorem flow for source-level carry may state that* 011 *and* 100 *are related by a carry classifier. It contains dependencies for source events, normalizer recognizer, carry classifier, and ledger-preservation theorem; proof flow from recognizer soundness; certificates for normalizer recognition and classifier exactness; ledger retaining pre-normal* 011 *and normal-address* 100*; status, canonical site, and seal. Cannot-claims state that this does not prove* $011 = 100$ *as* hsame*, does not imply dimension-lift without its own* NameCert *flow, and does not imply geometric dimension.*

Example 9231.51 (Accepted object theorem flow)**.** *A theorem flow may state* $\mathsf{AcceptGateFlow}(N, s)$*. It contains dependencies on the* NameCert *flow for* N*, derivation-certificate flow for* N *at* s*, and strength flow* s*; proof flow constructing the acceptance gate; certificate flow with* P_6 *accepted-object data; ledger referencing certificate ledgers; status, canonical site, and seal. This theorem code proves the acceptance-flow statement, but not arbitrary properties of* N*.*

Example 9231.52 (Incomplete theorem candidate)**.** *If a theorem candidate has statement, dependencies, certificates, ledger, status, canonical site, and seal, but proof is missing, the report says recognized theorem flow is absent and theorem code is not produced. The cannot-claim annotation says that statement alone is not theorem.*

9231.9 Conservativity

Theorem 9231.53 (P_7 is conservative over the finite kernel)**.** Lean4✓ BEDC.GroundCompiler.TheoremProofPrototy conservative over finite kernel

Adding statement-flow recognition, proof-flow recognition, proof-checking relations, theorem-flow recognition, theorem-code reports, and accepted-theorem reports adds no raw marks, history constructors, sameness rules, continuation rules, derived-interface formation rules, object-equality rules, or bridge rules.

Proof. P_7 operates on decoded event flows and generated recognition relations. It generates no new $\mathsf{b0}, \mathsf{b1}$, adds no msame, hsame, or psame, does not alter Ext or Cont, and does not license mature objects except through existing certificate-mediated acceptance flows. Theorem recognition records statement and proof structure; it does not change the finite kernel. □

Corollary 9231.54 (P_7 is theorem-reporting layer)**.** `Lean4✓ BEDC.GroundCompiler.TheoremProofPrototype.p7 theorem reporting layer`

P_7 can recognize theorem and proof flows, but without chapter-flow recognition and compiler self-hosting it is not a full manuscript compiler.

Remark 9231.55 (Summary)**.** *This chapter defines P_7, the theorem and proof recognition prototype. It adds statement candidates, generated statement recognizers, recognized statement flows, dependency candidates and flows, proof candidates, generated proof-flow recognizers, recognized proof flows, generated proof checkers, theorem candidates, generated theorem recognizers, recognized theorem flows, eight theorem subflows, proof, certificate, ledger, status, and canonical-site soundness, theorem code, abstract proposition classes, accepted theorem flows, P_7 reports, and the P_7 audit checklist. It proves that theorem is not input, statement flow is not abstract proposition, unsound dependencies block theorem recognition, proof candidates and proof flows are insufficient without checking, incomplete theorem candidates are not theorem flows, sound theorem flows establish theorem code, theorem code is injective on raw theorem flows, abstract proposition classes are quotients, accepted theorem code does not imply object acceptance or bridge status, P_7 adequacy requires generated recognizers and proof checkers, and P_7 is conservative over the finite kernel.*

Theorem 9231.56 (Accepted theorem export cites an AcceptGate witness)**.** *Let T be an accepted theorem flow. If the theorem ledger of T records a mature-object export claim for an object-flow name O at strength s, then there exist an event flow S, an acceptance-gate flow A, and a generated acceptance-gate recognizer R_A such that*

$$\mathsf{AcceptGateFlow}_{R_A}(S, A, O, s)$$

and the certificate or ledger subflow of T contains or cites A as the witness for that export claim.

Proof. By Definition 9231.34, accepted-theorem status is not only proof checking. It also requires that any claimed mature-object export has the relevant acceptance-gate flow, that all required certificates are recognized, and that the ledger is complete. Apply this defining condition to the ledger entry asserting export of O at strength s. The accepted-theorem witness for T therefore supplies a relevant acceptance-gate flow for that same object name and strength.

By Definition 9230.24, a relevant acceptance-gate flow is represented by some event flow S, subflow A, and generated recognizer R_A satisfying

$$\mathsf{AcceptGateFlow}_{R_A}(S, A, O, s).$$

The same accepted-theorem condition places this gate in the finite evidence carried by T: because the export is a theorem-ledger claim and the ledger is complete, the gate cannot be external unstated data; it must occur either as a certificate subflow of T or as a ledger citation retained by T. Thus the mature-object export claim projects to a concrete recognized AcceptGate witness for O at s. □

Chapter 9232

Chapter and Manuscript Recognition Prototype

chapter 9231 defines P_7, the theorem and proof recognition prototype. It recognizes statement flows, dependency flows, proof flows, theorem certificate flows, theorem ledgers, theorem status, canonical sites, and theorem seals. It can report theorem code and accepted theorem code, but it is still not a chapter compiler or manuscript compiler.

This chapter defines the next prototype level:

$$\boxed{P_8 = P_7 + \text{chapter-flow recognition} + \text{manuscript-flow recognition.}}$$

This is the first layer where chapter code, section code, and manuscript code may be reported. The boundary remains strict:

a chapter is not a heading, and a manuscript is not a file; both are recognized event-flow structures.

Thus P_8 does not accept chapter manifests, manuscript manifests, LaTeX section trees, Markdown heading trees, file paths, chapter numbers, external tables of contents, manual theorem lists, manual status tables, or manual dependency graphs. It accepts channel streams or decoded event flows, then recognizes chapter and manuscript structures by generated recognizers.

9232.1 P_8 prototype interface

Definition 9232.1 (Chapter and manuscript recognition prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.Chap`

A chapter and manuscript recognition prototype *is an executable prototype P_8 satisfying:*

1. *P_8 includes all P_7 functions;*
2. *P_8 recognizes definition flows;*
3. *P_8 recognizes lemma flows;*
4. *P_8 recognizes theorem-flow lists;*
5. *P_8 recognizes proof-flow lists;*
6. *P_8 recognizes certificate-flow lists;*
7. *P_8 recognizes chapter-level ledger flows;*
8. *P_8 recognizes chapter-level cannot-claim registries;*

9. *P_8 recognizes chapter status flows;*
10. *P_8 recognizes chapter-seal flows;*
11. *P_8 recognizes manuscript-level chapter sequences;*
12. *P_8 recognizes manuscript dependency flows;*
13. *P_8 recognizes manuscript global status flows;*
14. *P_8 recognizes manuscript-seal flows;*
15. *P_8 does not treat chapter titles, section numbers, file paths, YAML, LaTeX, or Markdown as formal input.*

Write

$$\mathsf{ChapterManuscriptPrototype}(P_8).$$

Principle 9232.2 (Chapter and manuscript are not inputs)**.** *Lean4✓ BEDC.GroundCompiler.ChapterManuscript.ch manuscript not input*

There is no formal input

$$\mathsf{ChapterPkg} \to \mathsf{Code} \qquad \textit{or} \qquad \mathsf{ManuscriptFile} \to \mathsf{Code}.$$

The only formal input remains EventFlow *or* LegalZStream*. If the input is channel stream c, then P_8 first forms $S = \mathsf{Decode}(c)$ and only then recognizes chapter or manuscript flows in S.*

Definition 9232.3 (P_8 input)**.** *Lean4 (definition) BEDC.GroundCompiler.ChapterManuscript.P8FormalInput*

The input to P_8 is displayed as channel stream c or event flow S. If the input is c, then P_8 first requires $c \in \mathsf{LegalZStream}$ and decodes it. If $c \notin \mathsf{LegalZStream}$, P_8 rejects and produces no chapter report or manuscript report.

Definition 9232.4 (P_8 output)**.** *Lean4 (definition) BEDC.GroundCompiler.ChapterManuscript.P8Output*

The output of P_8 is a report flow or human-readable report containing decoded event flow, chapter candidates, recognized chapter flows, chapter field status, definition-flow lists, lemma-flow lists, theorem-flow lists, proof-flow lists, certificate-flow lists, chapter ledger flow, chapter cannot-claim registry, chapter status flow, chapter seal flow, manuscript candidates, recognized manuscript flows, manuscript dependency order, global cannot-claim registry, global status table, manuscript seal flow, missing fields, warnings, and cannot-claim annotations.

9232.2 Definition and lemma recognition

Definition 9232.5 (Definition candidate and recognizer)**.** *Lean4 (definition) BEDC.GroundCompiler.ChapterFlow.De*

A definition candidate *is an event subflow*

$$D_{\mathrm{def}} \subseteq S$$

not yet recognized as a definition flow. Write $\mathsf{DefCand}(S, D_{\mathrm{def}})$.

A generated definition recognizer *is an event flow $R_{\mathrm{def}} \in \mathsf{EventFlow}$ supporting*

$$\mathsf{RecognizesDefinition}(R_{\mathrm{def}}, S, D_{\mathrm{def}}).$$

If this holds, D_{def} is a recognized definition flow. Write

$$\mathsf{DefinitionFlow}_{R_{\mathrm{def}}}(S, D_{\mathrm{def}}).$$

Definition 9232.6 (Definition soundness). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.DefinitionSoundn`

A definition flow is sound *when it only cites generated or deferred dependencies, does not introduce uncertified mature objects, includes or cites a NameCert-flow whenever it proposes a derived name, marks candidate-only definitions as candidate-only, records source events and hidden material in ledger flow, and does not treat host-level names as BEDC primitives.*

Theorem 9232.7 (Unsound definition cannot enter chapter-flow). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.un` `definition blocks`

Let $\mathsf{DefinitionSoundness}(R_{\mathrm{def}}, S, D_{\mathrm{def}})$ *record the dependency, certificate, ledger, candidate-only, and no-host-name clauses for a recognized definition flow. If a definition candidate* D_{def} *does not satisfy* $\mathsf{DefinitionSoundness}(R_{\mathrm{def}}, S, D_{\mathrm{def}})$*, then it cannot serve as a sound chapter definition subflow.*

Proof. A sound chapter-flow must not inject hidden input or uncertified mature objects through its definitions. If a definition candidate lacks dependency tracking, certificate support, ledger support, or a candidate-only marker, then the chapter would import unsupported structure. Hence the candidate cannot be admitted as a sound definition subflow. □

Definition 9232.8 (Lemma candidate and recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Lemma`

A lemma candidate *is an event subflow* $L_{\mathrm{lem}} \subseteq S$ *not yet recognized as a lemma flow. A* generated lemma recognizer *is an event flow* $R_{\mathrm{lem}} \in \mathsf{EventFlow}$ *supporting*

$$\mathsf{RecognizesLemma}(R_{\mathrm{lem}}, S, L_{\mathrm{lem}}).$$

If this holds, L_{lem} *is a recognized lemma flow.*

Proposition 9232.9 (Lemma-flow is theorem-like but site-local). *Lean4✓* `BEDC.GroundCompiler.ChapterFlow.lem` `site local`

If L_{lem} *is a recognized lemma-flow, then it has theorem-like subflow structure, while its canonical-site flow may be chapter-local rather than global.*

Proof. A lemma is a proof package used as a supporting result inside a chapter. It therefore requires statement, dependencies, proof, ledger, status, and seal data like a theorem-flow. Its canonical site records its chapter-local role, so it need not be a global theorem site. □

9232.3 Chapter-flow recognition

Definition 9232.10 (Chapter candidate and recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Cha` *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.GeneratedChapterRecognizer` *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.RecognizesChapter`

A chapter candidate *is an event subflow*

$$C \subseteq S$$

not yet recognized as a chapter flow. Write $\mathsf{ChapCand}(S, C)$.

A generated chapter recognizer *is an event flow* $R_C \in \mathsf{EventFlow}$ *supporting*

$$\mathsf{RecognizesChapter}(R_C, S, C).$$

Definition 9232.11 (Chapter subflows). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.ChapterSubflows`

If $\mathsf{RecognizesChapter}(R_C, S, C)$*, then* R_C *may identify:*

$$\begin{aligned}
&\mathsf{ChapStart}_{R_C}(C, C_0),\\
&\mathsf{ChapDependencies}_{R_C}(C, C_1),\\
&\mathsf{ChapDefinitions}_{R_C}(C, C_2),\\
&\mathsf{ChapLemmas}_{R_C}(C, C_3),\\
&\mathsf{ChapTheorems}_{R_C}(C, C_4),\\
&\mathsf{ChapProofs}_{R_C}(C, C_5),\\
&\mathsf{ChapCertificates}_{R_C}(C, C_6),\\
&\mathsf{ChapLedger}_{R_C}(C, C_7),\\
&\mathsf{ChapCannotClaims}_{R_C}(C, C_8),\\
&\mathsf{ChapStatus}_{R_C}(C, C_9),\\
&\mathsf{ChapSeal}_{R_C}(C, \sigma_C).
\end{aligned}$$

These are source-level subflow roles, not external fields from a table of contents.

Definition 9232.12 (Complete chapter recognition). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterFlow.Comple`

The recognition $\mathsf{RecognizesChapter}(R_C, S, C)$ *is* complete *when all chapter subflows in Definition 9232.11 are recognized.*

Principle 9232.13 (No chapter without complete chapter subflows). *Lean4 ✓* `BEDC.GroundCompiler.ChapterManus` `incomplete chapter`

A chapter-flow must contain recognized start, dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and seal. If any required subflow is missing, the flow remains a chapter candidate.

Theorem 9232.14 (Incomplete chapter candidate cannot be chapter-flow). *Lean4 ✓* `BEDC.GroundCompiler.Chapte` `chapter candidate not flow`

If a chapter candidate C *lacks any required chapter subflow, then* C *cannot become a recognized chapter-flow.*

Proof. Recognized chapter-flow requires complete chapter recognition. Missing any required subflow prevents complete recognition, so C cannot be used as a recognized chapter-flow. □

Corollary 9232.15 (Title alone is not chapter). *Lean4 ✓* `BEDC.GroundCompiler.ChapterManuscript.title` `alone not chapter`

A chapter-title-like event or chapter-start-flow alone is not a chapter-flow.

Corollary 9232.16 (Theorem list alone is not chapter). *Lean4 ✓* `BEDC.GroundCompiler.ChapterManuscript.theor` `list alone not chapter`

A recognized theorem-flow list alone is not a chapter-flow. Dependencies, definitions, proofs, certificates, ledger, cannot-claims, status, and seal are also required.

9232.4 Chapter soundness and code

Definition 9232.17 (Chapter soundness clauses). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManuscript.Cha`

For a recognized chapter-flow:

- *dependency soundness requires each dependency to be generated earlier, generated in an earlier chapter of the same manuscript-flow, generated earlier inside the same chapter, marked deferred, or listed as an open obligation;*
- *definition soundness requires every definition-flow to satisfy Definition 9232.6;*
- *theorem soundness requires every theorem-flow to be sound or explicitly marked as a theorem obligation;*
- *proof soundness requires every proof-flow to be checked by a generated proof checker or explicitly marked unchecked;*
- *certificate soundness requires every certificate-flow to be recognized by the appropriate package, NameCert, DerivCert, AcceptGate, normalizer, bridge, or reuse recognizer, or marked as a certificate obligation;*
- *ledger soundness records definition sources, theorem dependencies, proof dependencies, certificate dependencies, hidden abbreviations, classifier quotients, normal-address relations, bridge obligations, cannot-claim causes, and status justifications;*
- *cannot-claim soundness requires the chapter cannot-claim flow to match the chapter proof, certificate, and status state;*
- *status soundness requires each definition, lemma, theorem, certificate, bridge, and implementation target status to match its evidence.*

Definition 9232.18 (Sound chapter-flow)**.** Lean4 (definition) `BEDC.GroundCompiler.ChapterManuscript.SoundChapt`

A recognized chapter-flow C is sound *when complete chapter recognition holds and the dependency, definition, theorem, proof, certificate, ledger, cannot-claim, and status soundness clauses all hold.*

Theorem 9232.19 (Sound chapter-flow establishes chapter code)**.** Lean4✓ `BEDC.GroundCompiler.ChapterManuscr` `chapter code`

If C is a sound chapter-flow, then one may define

$$\mathsf{ChapterCode}(C) := \mathsf{Compile}(C),$$

and this code is a lossless channel encoding of C.

Proof. The flow C is an event subflow, so $\mathsf{Compile}(C)$ is defined. By the channel bijection, compilation is lossless on event flows. By sound chapter-flow, C contains the required chapter subflows and soundness evidence. Hence its compiled stream may serve as chapter code. □

Theorem 9232.20 (Chapter code is injective on raw chapter-flows)**.** Lean4✓ `BEDC.GroundCompiler.ChapterManus` `code injective`

If C and D are sound chapter-flows and

$$\mathsf{ChapterCode}(C) = \mathsf{ChapterCode}(D),$$

then $C = D$.

Proof. By definition, the equality is $\mathsf{Compile}(C) = \mathsf{Compile}(D)$. Injectivity of the channel compiler gives $C = D$. □

Corollary 9232.21 (Chapter code equality is stronger than topic equivalence)**.** Lean4✓ `BEDC.GroundCompiler.Cha` `code not topic`

Equal chapter codes imply equal raw chapter-flows. Two chapters with the same topic, such as NameCert, need not have equal chapter codes.

9232.5 Manuscript-flow recognition

Definition 9232.22 (Manuscript candidate and recognizer). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManus` *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManuscript.GeneratedManuscriptRecognizer` *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManuscript.RecognizesManuscript`

A manuscript candidate *is an event subflow $M \subseteq S$ not yet recognized as a manuscript-flow. A* generated manuscript recognizer *is an event flow $R_M \in \mathsf{EventFlow}$ supporting*

$$\mathsf{RecognizesManuscript}(R_M, S, M).$$

Definition 9232.23 (Manuscript subflows). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManuscript.Manuscri`

If $\mathsf{RecognizesManuscript}(R_M, S, M)$*, then R_M may identify:*

$$\begin{aligned}
&\mathsf{MsStart}_{R_M}(M, M_0),\\
&\mathsf{MsChapters}_{R_M}(M, M_1),\\
&\mathsf{MsDependencies}_{R_M}(M, M_2),\\
&\mathsf{MsNotation}_{R_M}(M, M_3),\\
&\mathsf{MsCertificates}_{R_M}(M, M_4),\\
&\mathsf{MsLedger}_{R_M}(M, M_5),\\
&\mathsf{MsCannotClaims}_{R_M}(M, M_6),\\
&\mathsf{MsStatus}_{R_M}(M, M_7),\\
&\mathsf{MsBridgeObligations}_{R_M}(M, M_8),\\
&\mathsf{MsImplementationTargets}_{R_M}(M, M_9),\\
&\mathsf{MsSeal}_{R_M}(M, \sigma_M).
\end{aligned}$$

Definition 9232.24 (Complete manuscript recognition). *Lean4 (definition)* `BEDC.GroundCompiler.ChapterManuscri`

The recognition $\mathsf{RecognizesManuscript}(R_M, S, M)$ *is* complete *when all manuscript subflows in Definition 9232.23 are recognized.*

Principle 9232.25 (No manuscript without complete manuscript subflows). *Lean4 ✓* `BEDC.GroundCompiler.Chapt` `incomplete manuscript`

A manuscript-flow must contain recognized start, chapter sequence, global dependencies, notation, certificates, ledger, cannot-claims, status, bridge obligations, implementation targets, and seal. If any required subflow is missing, the flow remains a manuscript candidate.

Theorem 9232.26 (Incomplete manuscript candidate cannot be manuscript-flow). *Lean4 ✓* `BEDC.GroundCompiler` `manuscript flow not manuscript`

If a manuscript candidate M lacks any required manuscript subflow, then M cannot become a recognized manuscript-flow.

Proof. Manuscript-flow recognition requires complete manuscript recognition. Missing any required subflow prevents complete recognition, so M cannot be used as a recognized manuscript-flow. □

Corollary 9232.27 (Table of contents alone is not manuscript). *Lean4 ✓* `BEDC.GroundCompiler.ChapterManuscrip` `of contents alone not manuscript`

A chapter sequence or table-of-contents-like flow alone is not a manuscript-flow.

9232.6 Manuscript soundness and code

Definition 9232.28 (Manuscript soundness clauses). Lean4 (definition) BEDC.GroundCompiler.ChapterManuscript..

For a recognized manuscript-flow:

- *chapter soundness requires every chapter-flow to be sound or explicitly marked as pending or cannot-claim chapter;*
- *dependency soundness requires chapter dependencies to form a well-founded dependency structure, unless a dependency is marked deferred, open, or future bridge obligation;*
- *notation soundness requires notation, abbreviations, and symbolic aliases to trace to event-flow definitions or recognized NameCert-flows;*
- *certificate soundness requires NameCert, DerivCert, AcceptGate, bridge, reuse, and normalizer flows used by chapters to be globally recorded and traceable;*
- *ledger soundness tracks chapter ledgers, cross-chapter dependencies, hidden abbreviations, classifier quotients, normal-address relations, bridge obligations, implementation targets, and bootstrap obligations;*
- *cannot-claim soundness requires all undischarged cannot-claim entries to appear in the global registry;*
- *status soundness requires global chapter, theorem, certificate, implementation target, and bridge-obligation status to agree with local chapter status;*
- *implementation soundness records Lean targets, reference executable targets, round-trip tests, no-host-leak audits, bootstrap obligations, and self-hosting status consistently with the relevant flows.*

Definition 9232.29 (Sound manuscript-flow). Lean4 (definition) BEDC.GroundCompiler.ChapterManuscript.SoundM

A recognized manuscript-flow M is sound *when complete manuscript recognition holds and the chapter, dependency, notation, certificate, ledger, cannot-claim, status, and implementation soundness clauses all hold.*

Theorem 9232.30 (Sound manuscript-flow establishes manuscript code). Lean4 ✓ BEDC.GroundCompiler.Chapter manuscript code

If M is a sound manuscript-flow, then one may define

$$\mathsf{ManuscriptCode}(M) := \mathsf{Compile}(M),$$

and this code is a lossless channel encoding of M.

Proof. The flow M is an event subflow, so $\mathsf{Compile}(M)$ is defined. The channel bijection makes this encoding lossless. Sound manuscript-flow provides the required chapter sequence, global dependencies, certificates, ledger, cannot-claims, status, implementation targets, and seal. Hence the compiled stream may serve as manuscript code. □

Theorem 9232.31 (Manuscript code is injective on raw manuscript-flows). Lean4 ✓ BEDC.GroundCompiler.Chapte code injective

If M and N are sound manuscript-flows and

$$\mathsf{ManuscriptCode}(M) = \mathsf{ManuscriptCode}(N),$$

then $M = N$.

Proof. By definition, the equality is $\mathsf{Compile}(M) = \mathsf{Compile}(N)$. Injectivity of the channel compiler gives $M = N$. □

Corollary 9232.32 (Manuscript code equality is stronger than topic equivalence). Lean4✓ `BEDC.GroundCompiler.code not topic`

Equal manuscript codes imply equal raw manuscript-flows. Two manuscripts that discuss the same compiler topic need not have equal manuscript codes.

9232.7 Reports and soundness

Definition 9232.33 (P_8 report). Lean4 (definition) `BEDC.GroundCompiler.ChapterManuscript.P8Report`

A P_8 report is an output view containing decoded event flow, chapter candidates, recognized chapter-flows, chapter field status, soundness status per chapter, chapter codes, manuscript candidates, recognized manuscript-flows, manuscript field status, manuscript soundness status, manuscript code, missing fields, dependency failures, ledger failures, cannot-claim gaps, status inconsistencies, implementation target gaps, warnings, and cannot-claim annotations.

Definition 9232.34 (Chapter and manuscript field status). Lean4 (definition) `BEDC.GroundCompiler.ChapterManusc`

For each chapter candidate, the report records status for start, dependencies, definitions, lemmas, theorems, proofs, certificates, ledger, cannot-claims, status, and seal as present, missing, or candidate-only. For each manuscript candidate, the report records status for start, chapters, dependencies, notation, certificates, ledger, cannot-claims, status, bridge obligations, implementation targets, and seal as present, missing, or candidate-only.

Definition 9232.35 (P_8 cannot-claim annotations). Lean4 (definition) `BEDC.GroundCompiler.ChapterManuscript.C`

A P_8 report must include cannot-claim annotations stating that chapter title alone is not chapter, theorem list alone is not chapter, table of contents alone is not manuscript, chapter code is not theorem proof, manuscript code does not imply all claims accepted, sound chapter does not imply all objects mentioned in the chapter accepted, sound manuscript does not imply all bridge obligations discharged, missing global cannot-claim registry blocks manuscript soundness, implementation target list is not checked implementation, and manuscript topic equality does not imply manuscript code equality.

Definition 9232.36 (Sound P_8 report). Lean4 (definition) `BEDC.GroundCompiler.ChapterManuscript.SoundP8Repor`

A P_8 report is sound *when every recognized chapter-flow is backed by a generated chapter recognizer, every recognized chapter-flow satisfies complete recognition, every sound chapter-flow claim satisfies chapter soundness, every chapter code is $\mathsf{Compile}(C)$ for the corresponding chapter-flow, every recognized manuscript-flow is backed by a generated manuscript recognizer, every recognized manuscript-flow satisfies complete recognition, every sound manuscript-flow claim satisfies manuscript soundness, manuscript code is $\mathsf{Compile}(M)$ for the corresponding manuscript-flow, missing fields and failures are explicit, and candidate-only entries are not reported as recognized.*

Theorem 9232.37 (Sound P_8 report theorem). Lean4✓ `BEDC.GroundCompiler.ChapterManuscript.sound p8 report`

If P_8 outputs a sound P_8 report, then every chapter code in the report corresponds to a sound chapter-flow, and every manuscript code in the report corresponds to a sound manuscript-flow.

Proof. By soundness of the report, chapter code may only be reported from a sound recognized chapter-flow, and manuscript code may only be reported from a sound recognized manuscript-flow. Hence every reported code has event-flow evidence of the stated kind. □

Corollary 9232.38 (Sound manuscript code is flow-mediated). Lean4✓ BEDC.GroundCompiler.ChapterManuscript code flow mediated

Manuscript code is not a file name, article title, or table of contents. It is the channel compilation of a sound manuscript-flow.

9232.8 Adequacy

Definition 9232.39 (P_8 audit checklist). Lean4 (definition) BEDC.GroundCompiler.ChapterManuscript.P8AuditChec

A P_8 prototype must satisfy:

1. *all P_7 audit requirements;*
2. *it separates chapter candidates from recognized chapter-flows;*
3. *recognized chapter-flow requires generated chapter recognizer;*
4. *recognized chapter-flow requires complete chapter subflows;*
5. *sound chapter claims require chapter soundness;*
6. *it separates manuscript candidates from recognized manuscript-flows;*
7. *recognized manuscript-flow requires generated manuscript recognizer;*
8. *recognized manuscript-flow requires complete manuscript subflows;*
9. *sound manuscript claims require manuscript soundness;*
10. *missing fields are explicit;*
11. *dependency failures are recorded;*
12. *ledger failures are recorded;*
13. *cannot-claim gaps are recorded;*
14. *status inconsistencies are recorded;*
15. *implementation target gaps are recorded;*
16. *YAML, LaTeX, Markdown, and file paths are not treated as formal input;*
17. *cannot-claim annotations are included.*

Theorem 9232.40 (P_8 adequacy theorem). Lean4✓ BEDC.GroundCompiler.ChapterManuscript.p8 adequacy

If a prototype satisfies the P_8 audit checklist, uses sound P_7 reports, and every chapter or manuscript recognition claim is backed by generated recognizers and soundness evidence, then it is adequate as a chapter and manuscript recognition prototype.

Proof. The audit checklist separates candidates from recognized flows, requires generated recognizers, enforces complete subflows and soundness, records failures, forbids YAML, LaTeX, Markdown, and file paths as formal inputs, and requires cannot-claim annotations. Sound P_7 reports provide reliable theorem and proof evidence when referenced. Thus the prototype is adequate at level P_8. □

Corollary 9232.41 (P_8 adequacy does not imply higher adequacy)**.** Lean4✓ BEDC.GroundCompiler.ChapterManusc
adequacy not higher

A P_8 prototype recognizes chapter and manuscript flows, but it is not yet a self-hosting compiler, full bridge-certifier, or global implementation verifier unless those recognizers and certificates are supplied.

9232.9 Examples

Example 9232.42 (Chapter candidate with theorem list only)**.** *A report may show theorem-flow list and proof-flow list present, while chapter ledger, cannot-claim registry, status-flow, and chapter seal are missing. The conclusion is that there is no recognized chapter-flow and no chapter code. The required cannot-claim is that a theorem list alone is not a chapter.*

Example 9232.43 (Sound compiler chapter)**.** *A sound chapter-flow for the channel compiler may contain chapter start, dependencies on raw event, event flow, and* LegalZStream*, definitions of ζ,* Ev*,* Compile*, and* Decode*, lemmas such as no internal* 11 *and first terminator uniqueness, theorems for both round trips, proofs by induction on raw events and event flows, channel conservativity certificates, source/channel separation certificates, ledger recording raw flow, channel stream, and decoded flow, cannot-claims for code-is-not-proof and channel-substring-is-not-source-event, status, and chapter seal. Then* $\mathsf{ChapterCode}(C_{\text{channel}}) = \mathsf{Compile}(C_{\text{channel}})$.

Example 9232.44 (Manuscript candidate missing global cannot-claim registry)**.** *A manuscript candidate may have chapters, dependencies, certificates, ledger, status, and seal, while global cannot-claims are missing. The report must say that sound manuscript-flow is absent, because the global cannot-claim registry is missing.*

Example 9232.45 (Sound manuscript-flow)**.** *A sound manuscript-flow may contain manuscript start, the chapter sequence for the no-hidden-input streaming compiler part, global dependency order, global notation for* EventFlow*,* LegalZStream*,* Compile*,* Decode*, and recognizer relations, global certificates for channel bijection, recognizer generatedness, certificate-mediated export, theorem-flow soundness, and chapter-flow soundness, global ledger for event-flow provenance, raw/channel distinction, bootstrap obligations, and recognizer hierarchy, cannot-claim entries, status, bridge obligations, implementation targets for prototype layers, and manuscript seal. Then* $\mathsf{ManuscriptCode}(M) = \mathsf{Compile}(M)$.

9232.10 Conservativity

Theorem 9232.46 (P_8 is conservative over the finite kernel)**.** Lean4✓ BEDC.GroundCompiler.ChapterManuscript.p
conservative over finite kernel

Adding chapter-flow recognition, manuscript-flow recognition, chapter codes, manuscript codes, chapter reports, manuscript reports, and global status, cannot-claim, and ledger reports does not add new raw marks, history constructors, sameness rules, continuation rules, object-equality rules, bridge rules, or mature-object export rules.

Proof. P_8 operates on decoded event flows and generated recognition relations. It does not generate new b_0, b_1, does not add msame, hsame, or psame rules, does not alter Ext/Cont, and does not license mature objects beyond already recognized certificate-mediated flows. Chapter and manuscript recognition organize event-flow structures; they do not add finite-kernel rules. □

Corollary 9232.47 (P_8 is not full self-hosting compiler)**.** `Lean4✓ BEDC.GroundCompiler.ChapterManuscript.p8 not full self hosting`

P_8 can recognize chapters and manuscripts, but without compiler self-hosting and global implementation verification it is not a fully self-hosted compiler.

9232.11 Summary

This chapter defines P_8, the chapter and manuscript recognition prototype. It adds definition recognizers, lemma recognizers, chapter candidates, generated chapter recognizers, chapter subflows, complete chapter recognition, chapter soundness, chapter code, manuscript candidates, generated manuscript recognizers, manuscript subflows, complete manuscript recognition, manuscript soundness, manuscript code, P_8 reports, and the P_8 audit checklist.

The main conclusions are:

1. chapter and manuscript are not inputs;
2. unsound definitions block chapter-flow soundness;
3. lemma-flow is theorem-like but site-local;
4. incomplete chapter candidates cannot become chapter-flows;
5. title alone and theorem list alone are not chapters;
6. sound chapter-flow establishes chapter code;
7. chapter code is injective on raw chapter-flows;
8. topic equivalence does not imply chapter-code equality;
9. incomplete manuscript candidates cannot become manuscript-flows;
10. table of contents alone is not manuscript;
11. sound manuscript-flow establishes manuscript code;
12. manuscript code is injective on raw manuscript-flows;
13. manuscript topic equivalence does not imply manuscript-code equality;
14. sound P_8 reports give chapter and manuscript flow evidence;
15. P_8 adequacy requires generated recognizers, complete subflows, soundness, missing-field reports, and cannot-claims;
16. P_8 adequacy does not imply higher adequacy;
17. P_8 is conservative over the finite kernel;
18. P_8 is a manuscript-reporting layer, not a full self-hosting compiler.

Theorem 9232.48 (Sound manuscript covers chapter cannot-claims globally)**.** *Let M be a sound manuscript-flow recognized by R_M, and let C be a chapter-flow in the manuscript chapter sequence $\mathsf{MsChapters}_{R_M}(M, M_1)$. If e is an undischarged cannot-claim entry in the chapter cannot-claim flow $\mathsf{ChapCannotClaims}_{R_C}(C, C_8)$, then e appears in the manuscript global cannot-claim registry $\mathsf{MsCannotClaims}_{R_M}(M, M_6)$.*

Proof. By Definition 9232.29, soundness of M includes the manuscript cannot-claim soundness clause. By Definition 9232.28, that clause requires all undischarged cannot-claim entries to appear in the global registry. Since C is one of the chapter-flows in the manuscript chapter sequence, the undischarged chapter entry e is one of the manuscript's undischarged cannot-claim entries. Applying the cannot-claim soundness clause places e in $\mathsf{MsCannotClaims}_{R_M}(M, M_6)$. □

Chapter 9233

Self-Hosting Compiler and Global Verification Prototype

chapter 9232 defines P_8, the chapter and manuscript recognition prototype. It recognizes chapter flows, manuscript flows, definition flows, lemma flows, theorem-flow lists, proof-flow lists, certificate-flow lists, chapter ledgers, global cannot-claim registries, global status flows, and manuscript seals.

It is still not a complete no-hidden-input compiler. Even if chapters, manuscripts, theorems, and certificates are recognized, the compiler that recognizes them could still be an external program, host parser, elaborator, utility program, reader, or manual convention.

This chapter defines the final prototype level:

$$\boxed{P_9 = P_8 + \text{compiler-flow recognition} + \text{compiler certificate-flow} + \text{bootstrap obligation-flow} + \text{self-compilation-flo}}$$

This is the closing layer of the no-hidden-input compiler architecture. The core boundary is:

$$\boxed{\text{the compiler is not input; the compiler itself must be an event flow.}}$$

Full no-hidden-input status may be claimed only under self-hosting with no remaining obligations, or under an explicitly declared bootstrap boundary.

9233.1 P_9 prototype interface

Definition 9233.1 (Self-hosting compiler prototype)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompile`

A self-hosting compiler prototype *is an executable prototype P_9 satisfying:*

1. *P_9 includes all P_8 functions;*
2. *P_9 recognizes compiler flows;*
3. *P_9 recognizes compiler certificate flows;*
4. *P_9 recognizes bootstrap-obligation flows;*
5. *P_9 recognizes compiler behavior flows;*
6. *P_9 recognizes self-compilation flows;*
7. *P_9 recognizes behavior-equivalence flows;*
8. *P_9 generates self-hosting ledgers;*

9. *P_9 generates global verification reports;*
10. *P_9 does not treat host executables, file paths, binary hashes, build procedures, data manifests, or host environments as compiler identity;*
11. *P_9 does not treat behavior equivalence as raw event-flow equality;*
12. *P_9 does not claim full no-hidden-input compiler status while bootstrap obligations remain undisclosed or undischarged.*

Write

$$\mathsf{SelfHostingPrototype}(P_9).$$

Principle 9233.2 (Compiler is not input)**.** *Lean4✓ BEDC.GroundCompiler.SelfHostingCompilerFlow.host compiler identity not input*

There is no formal input

$$\mathsf{HostCompiler} \to \mathsf{Code},$$

nor formal inputs

$$\mathsf{ExecutablePath} \to \mathsf{CompilerFlow}, \qquad \mathsf{BinaryHash} \to \mathsf{CompilerFlow}, \qquad \mathsf{BuildScript} \to \mathsf{CompilerFlow}.$$

The only formal input remains EventFlow *or* LegalZStream*. If a compiler enters the system, it must appear as a recognized compiler-flow inside a decoded event flow.*

Definition 9233.3 (Compiler candidate and recognizer)**.** *Lean4 (definition) BEDC.GroundCompiler.SelfHostingComp Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlow.P9RecognizesCompiler Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlow.P9RecognizedCompilerFlow*

A compiler candidate flow *is an event subflow $C \subseteq S$ not yet recognized as compiler-flow. Write* $\mathsf{CompilerCand}(S, C)$.

A generated compiler recognizer *is an event flow $R_C \in \mathsf{EventFlow}$ supporting*

$$\mathsf{RecognizesCompiler}(R_C, S, C).$$

If this holds, C is a recognized compiler-flow in S. Write

$$\mathsf{CompilerFlow}_{R_C}(S, C).$$

9233.2 Compiler certificate-flow

Definition 9233.4 (Compiler certificate candidate and recognizer)**.** *Lean4 (definition) BEDC.GroundCompiler.SelfHo Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlow.P9RecognizesCompilerCert*

A compiler certificate candidate *is an event subflow $K_C \subseteq S$ claiming to certify compilation capability for compiler-flow C. Write* $\mathsf{CompilerCertCand}(S, K_C, C)$.

A generated compiler-certificate recognizer *is an event flow $R_K \in \mathsf{EventFlow}$ supporting*

$$\mathsf{RecognizesCompilerCert}(R_K, S, K_C, C).$$

Definition 9233.5 (Compiler certificate subflows)**.** *Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerF*

If $\mathsf{RecognizesCompilerCert}(R_K, S, K_C, C)$*, then* R_K *may identify:*

$$\begin{aligned}
&\mathsf{CompInputDom}_{R_K}(K_C, K_0),\\
&\mathsf{CompOutputDom}_{R_K}(K_C, K_1),\\
&\mathsf{CompEncodingRule}_{R_K}(K_C, K_2),\\
&\mathsf{CompDecodingRule}_{R_K}(K_C, K_3),\\
&\mathsf{CompRoundTrip}_{R_K}(K_C, K_4),\\
&\mathsf{CompRecognizerHierarchy}_{R_K}(K_C, K_5),\\
&\mathsf{CompFailure}_{R_K}(K_C, K_6),\\
&\mathsf{CompLedger}_{R_K}(K_C, K_7),\\
&\mathsf{CompNoHostLeak}_{R_K}(K_C, K_8),\\
&\mathsf{CompKernelConservativity}_{R_K}(K_C, K_9),\\
&\mathsf{CompSelfDescription}_{R_K}(K_C, K_{10}),\\
&\mathsf{CompSeal}_{R_K}(K_C, \sigma_C).
\end{aligned}$$

Definition 9233.6 (Complete compiler certificate recognition)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHosti`

The recognition $\mathsf{RecognizesCompilerCert}(R_K, S, K_C, C)$ *is* complete *when all compiler certificate subflows in Definition 9233.5 are recognized.*

Principle 9233.7 (No compiler without certificate)**.** *Lean4✓* `BEDC.GroundCompiler.SelfHostingCompilerFlow.p9` `no compiler without complete certificate`

A compiler-flow cannot be used as a formal compiler unless it has complete compiler certificate-flow. Recognizing $\mathsf{CompilerFlow}_{R_C}(S, C)$ *alone is insufficient for formal compilation judgments.*

Theorem 9233.8 (Uncertified compiler-flow cannot compile formally)**.** *Lean4✓* `BEDC.GroundCompiler.SelfHostin` `uncertified cannot compile`

If C *is a recognized compiler-flow but no complete compiler certificate-flow* K_C *exists for* C*, then* $\mathsf{Compiles}_C(X, Y)$ *cannot be used as a formal judgment.*

Proof. Formal compilation requires event-flow evidence for input domain, output domain, encoding rules, decoding rules, round-trip behavior, failure behavior, ledger, no-host-leak status, and kernel conservativity. Without compiler certificate-flow, these behaviors are unsupported by internal evidence. Using C formally would introduce hidden compiler input. □

Corollary 9233.9 (Recognized compiler-flow is weaker than certified compiler)**.** *Lean4✓* `BEDC.GroundCompiler.Se` `certified compiler has recognized flow`

Certified compiler status includes recognized compiler-flow, but recognized compiler-flow by itself records only compiler-shaped event-flow structure. Formal compiler use requires compiler certificate-flow.

9233.3 Compiler behavior-flow

Definition 9233.10 (Compiler behavior relation)**.** *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFl`

The relation

$$\mathsf{Compiles}_C(X, Y)$$

has $C, X, Y \in \mathsf{EventFlow}$ *and reads: compiler-flow* C *compiles input event flow* X *into output event flow* Y*. It is a source-level relation over event flows, not a host function call.*

Definition 9233.11 (Behavior record). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.P9Behav`

A behavior record *is an event subflow B_C recording compiler-flow C, input flow X, output flow Y, the compilation relation $C : X \mapsto Y$, failure branches, and ledger. Write*

$$\mathsf{BehaviorRecord}(B_C, C, X, Y).$$

Definition 9233.12 (Behavior soundness). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.P9B`

A behavior record B_C is sound *when its recorded* $\mathsf{Compiles}_C(X, Y)$ *relation agrees with the encoding, decoding, and recognition rules in C's compiler certificate-flow, and when failure-flow and ledger-flow are complete.*

Theorem 9233.13 (Sound behavior record establishes compiler action). *Lean4✓* `BEDC.GroundCompiler.SelfHosti` `sound behavior record establishes action`

If B_C is a sound behavior record for (C, X, Y), then $\mathsf{Compiles}_C(X, Y)$ *may be used formally.*

Proof. Sound behavior records contain the compiler, input, output, compilation relation, failure behavior, and ledger, and verify consistency with the compiler certificate-flow. They therefore provide event-flow evidence for the compiler action. □

9233.4 Bootstrap obligations

Definition 9233.14 (Bootstrap component). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.B`

A bootstrap component *is an implementation component not yet fully generated or verified by self-hosting compiler-flow. Examples include initial channel encoders, initial decoders, initial recognizer-certificate checkers, proof-checker bridges, host executables, and kernel reliance.*

Definition 9233.15 (Bootstrap-obligation flow). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlo`

A bootstrap-obligation flow *is an event subflow B recording the bootstrap component, its role, why it is needed, what is not internally generated yet, what would discharge it, current status, ledger, and cannot-claim entries. Write*

$$\mathsf{BootstrapObligationFlow}(B).$$

Principle 9233.16 (No invisible bootstrap). *Lean4✓* `BEDC.GroundCompiler.SelfHostingCompilerFlow.no` `invisible bootstrap`

Every undisclosed or undischarged bootstrap component used by a formal judgment must have bootstrap-obligation flow. No bootstrap assumption may hide inside host implementation.

Theorem 9233.17 (Invisible bootstrap invalidates full no-hidden-input claim). *Lean4✓* `BEDC.GroundCompiler.Se` `bootstrap invalidates full claim`

If a formal judgment depends on a bootstrap component B but no corresponding bootstrap-obligation flow exists, then the system cannot claim full no-hidden-input compiler status.

Proof. Full no-hidden-input status requires every used structure to come from event flow, recognized structure, certificate-flow, or explicit bootstrap obligation. If B is used but not recorded, its source remains external and hidden. The full claim is therefore invalid. □

Corollary 9233.18 (Bootstrap boundary must be reportable). *Lean4✓* `BEDC.GroundCompiler.SelfHostingCompil` `boundary reportable`

Every P_9 report must list all remaining bootstrap obligations.

9233.5 Self-compilation

Definition 9233.19 (Self-compilation flow). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.Se`

A self-compilation flow *is a recognized event flow SC containing compiler-flow C, input flow C, compiled output flow C', behavior record $C : C \mapsto C'$, and self-compilation ledger. Write*

$$\mathsf{SelfCompilationFlow}(SC, C, C').$$

Definition 9233.20 (Self-compilation ledger). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompilerFlow.`

A self-compilation ledger *is an event subflow L_{SC} recording raw compiler-flow C, channel code* $\mathsf{Compile}(C)$*, decoded or output compiler-flow C', behavior record, failure branches, bootstrap obligations used, bootstrap obligations discharged, and remaining bootstrap obligations. Write*

$$\mathsf{SelfCompilationLedger}(L_{SC}, C, C').$$

Principle 9233.21 (No self-compilation without ledger). *Lean4 ✓* `BEDC.GroundCompiler.SelfHostingCompilerFlo` `self compilation without ledger`

A self-compilation claim requires self-compilation ledger. Thus $\mathsf{SelfCompilationFlow}(SC, C, C')$ *must include* $\mathsf{SelfCompilationLedger}(L_{SC}, C, C')$.

Theorem 9233.22 (Self-compilation flow preserves compiler provenance). *Lean4 ✓* `BEDC.GroundCompiler.SelfHos` `compilation preserves provenance`

If $\mathsf{SelfCompilationFlow}(SC, C, C')$ *holds and its ledger is sound, then the raw compiler C, compiled output C', and compilation provenance are traceable from the ledger.*

Proof. The self-compilation ledger records C, C', the behavior record, failure branches, bootstrap obligations, and discharge status. Ledger soundness makes these records complete. Hence compiler provenance is traceable. □

9233.6 Compiler behavior equivalence

Definition 9233.23 (Compiler behavior equivalence). *Lean4 (definition)* `BEDC.GroundCompiler.SelfHostingCompil`

Write

$$C \equiv_{\text{comp}} C'$$

when a compiler-behavior classifier witnesses both compiler certificates and the two directional behavior implications between C and C' for all event-flow inputs and outputs.

This is a compiler-behavior classifier. It is not raw event-flow equality, not hsame*, and not kernel equality.*

Proposition 9233.24 (Behavior equivalence is weaker than raw equality). *Lean4 ✓* `BEDC.GroundCompiler.SelfHos` `behavior equivalence not raw equality`

There exist compiler-flow candidates C and C' and a compiler-behavior classifier such that

$$C' \equiv_{\text{comp}} C \qquad \textit{and} \qquad C' \neq C.$$

Thus behavior equivalence is a behavior-level relation and is not raw event-flow equality.

Proof. Take two distinct certified event flows and a behavior relation that ignores raw compiler identity while preserving the same input-output behavior under both certified flows. The classifier records certificates and the two directional behavior implications. Since the two underlying event flows remain distinct, the behavior-equivalence judgment does not collapse to raw equality. □

Corollary 9233.25 (Self-hosting is behavior-level stability). Lean4 ✓ BEDC.GroundCompiler.SelfHostingCompiler
self hosting behavior level stability

Self-hosting proves $C' \equiv_{\text{comp}} C$, not raw equality $C' = C$.

9233.7 Self-hosting compiler-flow

Definition 9233.26 (Self-hosting compiler-flow). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlo

A compiler-flow C is self-hosting *when there exist C', a self-compilation flow SC, a behavior-equivalence flow, and a self-compilation ledger L_{SC} such that:*

1. *C is certified compiler-flow;*
2. $\mathsf{SelfCompilationFlow}(SC, C, C')$*;*
3. $\mathsf{SelfCompilationLedger}(L_{SC}, C, C')$*;*
4. *$C' \equiv_{\text{comp}} C$;*
5. *remaining bootstrap obligations are listed;*
6. *discharged bootstrap obligations are recorded;*
7. *no-host-leak audit passes or failed items are listed.*

Write $\mathsf{SelfHostingCompiler}(C)$.

Theorem 9233.27 (Self-hosting removes compiler-as-hidden-input up to obligations). Lean4 ✓
BEDC.GroundCompiler.SelfHostingCompilerFlow.self hosting removes hidden compiler

If $\mathsf{SelfHostingCompiler}(C)$ *holds and the recognizer hierarchy covers the compiler tower, then the compiler is no longer a hidden input: C is represented by self-hosting event-flow evidence, the compiler tower is recognized, and a self-hosting ledger exists. Every remaining external dependency must occur among the remaining bootstrap obligations.*

Proof. Self-hosting represents the compiler as event flow C and supplies compiler certificate-flow, self-compilation flow, behavior-equivalence flow, and self-compilation ledger. Its behavior is recorded by event-flow evidence. Any remaining external dependency must be listed as a bootstrap obligation by definition, so no compiler hidden input remains unrecorded. □

Corollary 9233.28 (Remaining obligations block full discharge). Lean4 ✓ BEDC.GroundCompiler.SelfHostingComp
obligations block full discharge

If $\mathsf{SelfHostingCompiler}(C)$ *holds but remaining bootstrap obligations are nonempty, then the system may claim self-hosting with declared obligations, but not fully discharged no-hidden-input compiler status.*

Definition 9233.29 (Fully discharged compiler-flow). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompile

A self-hosting compiler-flow C is fully discharged *when* $\mathsf{SelfHostingCompiler}(C)$ *holds and the remaining bootstrap-obligation list is empty. Write* $\mathsf{FullyDischargedCompiler}(C)$.

Theorem 9233.30 (Fully discharged compiler supports full no-hidden-input claim). Lean4 ✓
BEDC.GroundCompiler.SelfHostingCompilerFlow.fully discharged supports full claim

If $\mathsf{FullyDischargedCompiler}(C)$*, then C supports the full no-hidden-input compiler claim: the compiler is no longer a hidden input and there is no remaining recorded bootstrap boundary.*

Proof. Fully discharged compiler-flow is self-hosting and has no remaining bootstrap obligations. By Definition 9233.27, the compiler is not unrecorded hidden input. Since every external dependency has been discharged, no bootstrap boundary remains. □

9233.8 Global verification-flow

Definition 9233.31 (Global verification-flow). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompilerFlow.

A global verification-flow *is a recognized event flow G containing manuscript-flow, chapter-flow list, theorem-flow list, accepted-object-flow list, compiler-flow, self-hosting status, bootstrap-obligation registry, cannot-claim registry, implementation target status, no-host-leak audit, global ledger, and global status seal. Write*

$$\mathsf{GlobalVerificationFlow}(G).$$

Definition 9233.32 (Global verification soundness). Lean4 (definition) BEDC.GroundCompiler.SelfHostingCompiler

A global verification-flow G is sound *when the manuscript-flow is sound, all chapter-flow statuses are recorded, all theorem-flow statuses are recorded, all accepted-object-flow statuses are recorded, compiler-flow is recognized, self-hosting status is recorded, bootstrap obligations are complete, cannot-claim registry is complete, implementation targets are audited, no-host-leak audit is complete, global ledger is complete, and global seal is present.*

Theorem 9233.33 (Sound global verification-flow establishes system status). Lean4✓ BEDC.GroundCompiler.Self
global verification status

If G is a sound global verification-flow, then it may serve as the global status certificate-flow for the compiler and manuscript system.

Proof. Global verification-flow records manuscript, chapters, theorems, objects, compiler, bootstrap obligations, cannot-claims, implementation targets, ledger, and no-host-leak audit. Soundness ensures that these subsystem states are consistent and ledger-backed. Thus G can serve as global status certificate-flow. □

Corollary 9233.34 (Global verification adds no new mathematics). Lean4✓ BEDC.GroundCompiler.SelfHostingC
verification no new math

Global verification-flow records system status. It does not add new theorems, new objects, or new kernel rules.

9233.9 Reports and soundness

Definition 9233.35 (P_9 report). Lean4 (definition) BEDC.GroundCompiler.SelfHostingReports.P9Report

A P_9 report is an output view containing decoded event flow, compiler candidates, recognized compiler-flows, compiler certificate status, behavior records, self-compilation candidates, recognized self-compilation flows, behavior-equivalence status, self-hosting ledger, bootstrap obligations, fully discharged status, global verification-flow status, no-host-leak audit, implementation target audit, remaining obligations, warnings, and cannot-claim annotations.

Definition 9233.36 (Compiler field and self-hosting status). Lean4 (definition) BEDC.GroundCompiler.SelfHostingR
Lean4 (definition) BEDC.GroundCompiler.SelfHostingReports.P9SelfHostingStatusRecord

For each compiler candidate C, the report records recognized compiler-flow status, compiler certificate-flow status, input-domain flow, output-domain flow, encoding-rule flow, decoding-rule flow, round-trip proof flow, recognizer-hierarchy flow, ledger flow, failure flow, no-host-leak flow, kernel-conservativity flow, self-description flow, and seal flow.

Self-hosting status records compiler-flow C, self-compilation flow SC, output compiler-flow C', behavior-equivalence $C' \equiv_{\mathrm{comp}} C$, self-compilation ledger, discharged obligations, remaining obligations, no-host-leak audit result, and status as not self-hosting, self-hosting with obligations, or fully discharged.

Definition 9233.37 (P_9 cannot-claim annotations). Lean4 (definition) BEDC.GroundCompiler.SelfHostingReports..

A P_9 report must include cannot-claim annotations stating that recognized compiler-flow alone is not certified compiler, certified compiler alone is not self-hosting, self-compilation alone is not behavior equivalence, behavior equivalence is not raw equality, self-hosting with obligations is not fully discharged, host executable is not compiler identity, binary hash is not compiler-flow, global verification-flow is not a new theorem, implementation report is not proof, and full no-hidden-input claim requires no remaining obligations.

Definition 9233.38 (Sound P_9 report). Lean4 (definition) BEDC.GroundCompiler.SelfHostingReports.SoundP9Repo

A P_9 report is sound *when every recognized compiler-flow is backed by generated compiler recognizer, every certified compiler claim has complete compiler certificate-flow, every behavior record satisfies behavior soundness, every self-compilation claim has self-compilation ledger, every behavior-equivalence claim has recognized behavior-equivalence flow, all bootstrap obligations are listed, fully discharged status appears only when remaining obligations are empty, global verification-flow claims have soundness evidence, no-host-leak audit either passes or lists failed items, and cannot-claim annotations are complete.*

Theorem 9233.39 (Sound P_9 report theorem). Lean4 ✓ BEDC.GroundCompiler.SelfHostingReports.sound p9 report supports self hosting

↪ variant BEDC.GroundCompiler.SelfHostingReports.sound p9 report full claim requires empty obligations

If P_9 outputs a sound P_9 report, then every self-hosting claim is backed by certified compiler-flow, self-compilation-flow, behavior-equivalence-flow, and self-hosting ledger; every full no-hidden-input claim requires no remaining bootstrap obligations.

Proof. Sound P_9 report requires compiler certificate, self-compilation ledger, and behavior-equivalence flow for self-hosting claims. It also permits fully discharged status only when the remaining obligation list is empty. Hence the stated support is required. □

Corollary 9233.40 (Sound P_9 report distinguishes compiler statuses). Lean4 ✓ BEDC.GroundCompiler.SelfHostin statuses distinct

A sound P_9 report distinguishes compiler candidate, recognized compiler-flow, certified compiler-flow, self-compiling compiler-flow, self-hosting compiler-flow, self-hosting with obligations, and fully discharged compiler-flow.

9233.10 Adequacy

Definition 9233.41 (P_9 audit checklist). Lean4 (definition) BEDC.GroundCompiler.SelfHostingReports.P9AuditChe

A P_9 prototype must satisfy:

1. *all P_8 audit requirements;*
2. *it separates compiler candidates from recognized compiler-flows;*
3. *recognized compiler-flow requires generated compiler recognizer;*
4. *recognized compiler-flow is separated from certified compiler-flow;*
5. *certified compiler-flow requires complete compiler certificate-flow;*
6. *compiler behavior records are separated from host execution traces;*
7. *behavior records require behavior soundness;*

8. *self-compilation candidates are separated from self-compilation flows;*
9. *self-compilation claims require self-compilation ledger;*
10. *behavior equivalence is separated from raw equality;*
11. *bootstrap obligations are recorded;*
12. *self-hosting with obligations is separated from fully discharged status;*
13. *no-host-leak audit is required;*
14. *global status claims require global verification-flow;*
15. *executable paths, binary hashes, and build procedures are not compiler identity;*
16. *P_9 cannot-claim annotations are included.*

Theorem 9233.42 (P_9 adequacy theorem)**.** `Lean4✓ BEDC.GroundCompiler.SelfHostingReports.p9 adequacy`

If a prototype satisfies the P_9 audit checklist, uses sound P_8 reports, and every compiler, self-hosting, and global verification claim is backed by generated recognizers, certificates, ledgers, and bootstrap accounting, then it is adequate as a self-hosting compiler and global verification prototype.

Proof. The audit checklist enforces separation of candidate, recognized, certified, self-compiling, self-hosting, and fully discharged statuses. It requires compiler certificates, behavior records, self-compilation ledger, behavior equivalence, bootstrap accounting, no-host-leak audit, and global verification-flow. Sound P_8 reports provide manuscript, chapter, and theorem layer correctness. Therefore the prototype is adequate at level P_9. □

Corollary 9233.43 (P_9 adequacy is top prototype adequacy)**.** `Lean4✓ BEDC.GroundCompiler.SelfHostingReports top adequacy`

Within this prototype ladder, P_9 is the first level capable of supporting a full no-hidden-input compiler claim, provided remaining bootstrap obligations are empty.

9233.11 Examples

Example 9233.44 (Compiler candidate without certificate)**.** *A report may show compiler candidate present and recognized compiler-flow present, while compiler certificate-flow is missing. The conclusion is that there is no certified compiler and no self-hosting compiler. The required cannot-claim is that recognized compiler-flow alone is not certified compiler.*

Example 9233.45 (Certified compiler without self-compilation)**.** *A report may show recognized compiler-flow and complete compiler certificate-flow, while self-compilation-flow is missing. The conclusion is certified compiler yes, self-hosting no. Certified compiler does not imply self-hosting.*

Example 9233.46 (Self-compilation without behavior equivalence)**.** *A report may show compiler-flow C, self-compilation output C', and no behavior-equivalence flow. The conclusion is self-compilation candidate yes, self-hosting no. Self-compilation output does not imply $C' \equiv_{\text{comp}} C$.*

Example 9233.47 (Self-hosting with obligations)**.** *A report may show certified compiler C, self-compilation, behavior equivalence, self-hosting ledger, and remaining obligations such as an initial proof-checker bridge or no-host-leak audit failure. The conclusion is self-hosting with obligations, not fully discharged.*

Example 9233.48 (Fully discharged compiler)**.** *A report may show certified compiler C, self-compilation, behavior equivalence, self-hosting ledger, no bootstrap obligations, passed no-host-leak audit, and sound global verification-flow. The conclusion is fully discharged compiler and support for the full no-hidden-input claim.*

9233.12 Conservativity and ladder theorem

Theorem 9233.49 (P_9 is conservative over the finite kernel)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingReports.conservative over finite kernel`

Adding compiler-flow recognition, compiler certificate-flow, bootstrap-obligation-flow, self-compilation-flow, behavior-equivalence-flow, self-hosting ledger, global verification-flow, and P_9 reports does not add new raw marks, history constructors, sameness rules, continuation rules, object-equality rules, theorem rules, or bridge rules.

Proof. All P_9 structures are event flows or relations over event flows. Compiler behavior equivalence is a classifier over compiler flows, not kernel equality. Self-hosting records behavior-level stability, not new raw marks or sameness rules. Global verification records system status; it does not add mathematical objects or theorems. Hence P_9 is conservative over the finite kernel. □

Corollary 9233.50 (P_9 is verification layer)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingReports.p9 verification layer`

P_9 verifies compiler status and no-hidden-input conditions. It does not create new BEDC kernel objects or mature mathematical objects.

Theorem 9233.51 (Prototype ladder theorem)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.prototo ladder`

The prototype ladder

$$P_0 \to P_1 \to P_2 \to P_3 \to P_4 \to P_5 \to P_6 \to P_7 \to P_8 \to P_9$$

has the following interpretation: P_0 is channel encoding and decoding; P_1 is source-event reporting; P_2 is source-level normalizer ledger reporting; P_3 is motif reporting; P_4 is metric reporting; P_5 is package and NameCert recognition; P_6 is DerivCert and AcceptGate recognition; P_7 is theorem and proof recognition; P_8 is chapter and manuscript recognition; P_9 is self-hosting compiler and global verification. Each P_i extends the previous prototype by adding recognized event-flow structures and corresponding reports, without changing the channel compiler or finite kernel.

Proof. Each prototype level was defined as the previous level plus explicitly stated recognition or reporting capability. Each extension acts on decoded event flows, requires generated recognizers and ledgers where nontrivial, and remains conservative over the finite kernel. Therefore the ladder interpretation holds. □

Corollary 9233.52 (Full compiler claim is P_9-level)**.** Lean4✓ `BEDC.GroundCompiler.SelfHostingCompilerFlow.fu claim p9 level`

A system below P_9 cannot claim full self-hosting no-hidden-input compiler status. It may only claim the capabilities of its achieved prototype level.

9233.13 Summary

This chapter defines P_9, the self-hosting compiler and global verification prototype. It adds compiler candidates, generated compiler recognizers, recognized compiler-flows, compiler certificate-flow, compiler behavior relation, behavior records, bootstrap components, bootstrap-obligation flows, self-compilation flows, self-compilation ledgers, compiler behavior equivalence, self-hosting compiler-flow, fully discharged compiler-flow, global verification-flow, P_9 reports, the P_9 audit checklist, and the prototype ladder theorem.

The main conclusions are:

1. compiler is not input;
2. recognized compiler-flow alone is not certified compiler;
3. uncertified compiler cannot compile formally;
4. sound behavior record establishes compiler action;
5. invisible bootstrap invalidates full no-hidden-input status;
6. self-compilation requires ledger;
7. behavior equivalence is weaker than raw equality;
8. self-hosting is behavior-level stability;
9. self-hosting removes compiler-as-hidden-input up to declared obligations;
10. fully discharged compiler supports the full no-hidden-input claim;
11. sound global verification-flow establishes system status;
12. sound P_9 reports support self-hosting and full discharge claims only under required evidence;
13. P_9 adequacy requires compiler certificates, behavior records, self-compilation ledger, behavior equivalence, bootstrap accounting, no-host-leak audit, and global verification-flow;
14. P_9 adequacy is the top prototype adequacy;
15. P_9 is conservative over the finite kernel;
16. P_9 is a verification layer, not new mathematics;
17. the full prototype ladder P_0 through P_9 is established.

Chapter 9234

Programmatic Realization of the Prototype Ladder

chapter 9233 closes the prototype ladder at P_9. This chapter records the implementation reading of that ladder. The whole mechanism is programmatically realizable, but three claims must remain separate:

$$\boxed{\text{executable implementation} \neq \text{formally verified implementation} \neq \text{fully self-hosting implementation.}}$$

The executable layers can run as ordinary programs. The recognition and certificate layers require event-flow recognizers, checkers, ledgers, and failure records. The self-hosting layer requires compiler-flow evidence, behavior equivalence, and bootstrap-obligation accounting.

9234.1 Executable base

Definition 9234.1 (Executable channel core)**.** *Lean4 (definition)* `BEDC.GroundCompiler.ProgrammaticRealization.`

The executable channel core is the deterministic program layer implementing:

1. *source raw mark* 0 *encodes as channel word* 0*;*
2. *source raw mark* 1 *encodes as channel word* 10*;*
3. *event end encodes as channel word* 11*;*
4. *event flow compilation concatenates encoded events;*
5. *decoding rejects dangling or malformed channel words.*

Equivalently,

$$\mathsf{Ev}(w) = \zeta(w)11, \qquad \zeta(0) = 0, \qquad \zeta(1) = 10.$$

Proposition 9234.2 (The base language is stream-regular)**.** *Lean4✓* `BEDC.GroundCompiler.ProgrammaticRealiza` `language stream regular`

The legal channel stream language for the executable channel core is

$$((0 \mid 10)^*11)^*.$$

It is recognized by a finite streaming automaton.

Proof. Inside an event body the decoder accepts either 0 or the two-bit word 10. The word 11 terminates the current event. A single trailing 1 is rejected, and any non-binary mark is rejected before decoding. Thus each event has body $(0 \mid 10)^*$ followed by terminator 11, and legal streams are finite concatenations of such event words. This is a regular expression, hence a finite automaton recognizes it. □

Definition 9234.3 (Executable P_0 realization)**.** Lean4 (definition) `BEDC.GroundCompiler.ProgrammaticRealization`

An executable P_0 realization provides deterministic procedures for:

1. *encoding one raw event;*
2. *encoding a finite event flow;*
3. *checking legal channel streams;*
4. *decoding legal channel streams;*
5. *rejecting illegal channel streams;*
6. *running round-trip tests.*

This realization is executable code. It is not, by itself, a certificate that later recognizers, proofs, packages, or objects exist.

Example 9234.4 (Channel core trace)**.** *For the source event flow*

$$0 \mid 00 \mid 000 \mid 01 \mid 011 \mid 100,$$

the executable channel core computes

$$011001100011010110101011100011.$$

Decoding that channel stream recovers the same event flow.

9234.2 Report and metric layers

Definition 9234.5 (Executable report layer)**.** Lean4 (definition) `BEDC.GroundCompiler.ProgrammaticRealization.E`

The executable report layer consists of P_1 through P_4 report routines:

1. *P_1 source-event report generation;*
2. *P_2 adjacent-pair and normalizer-candidate ledger reporting;*
3. *P_3 motif candidate and recognized motif reporting;*
4. *P_4 admissible metric reporting.*

These routines may compute displays, candidate sections, missing-recognizer items, undefined metrics, and cannot-claim annotations.

Principle 9234.6 (Programs do not upgrade candidates)**.** Lean4 ✓ `BEDC.GroundCompiler.ProgrammaticRealizati` `do not upgrade candidates`

An executable report routine may discover candidate shapes, but it may not upgrade a candidate to a recognized structure without the corresponding recognizer-flow, certificate-flow, ledger-flow, or proof-checker-flow.

Example 9234.7 (Candidate pair boundary)**.** *The adjacent pair*

$$011 \mid 100$$

may be reported as a candidate normalizer pair by an executable routine. It may be reported as a recognized carry ledger only when a generated normalizer recognizer and its ledger support that judgment. The executable routine alone does not establish $\mathsf{ZeckendorfCarry}(011, 100)$*, equality, or* hsame*.*

Proposition 9234.8 (P_1 through P_4 are executable report layers)**.** Lean4✓ BEDC.GroundCompiler.ProgrammaticRe
p4 executable

P_1 through P_4 can be implemented as deterministic report and analysis procedures over decoded event flows, relative to any recognized recognizers and metric protocols supplied as event flows.

Proof. P_1 reads decoded events and their channel segments. P_2 enumerates adjacent source events and records candidate or recognized normalizer ledgers according to available policy and recognizer evidence. P_3 separates candidate motifs from recognized motifs. P_4 computes only admissible metrics and explicitly records undefined items. Each step is a finite computation over decoded event-flow data and recognized evidence already present in the input environment. Hence each step is executable. □

9234.3 Recognizer virtual machine

Definition 9234.9 (Recognizer virtual machine)**.** Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRealizatio

A recognizer virtual machine *is an executable interpreter whose formal inputs are:*

1. *an ambient decoded event flow S;*
2. *a recognizer-flow R;*
3. *an optional candidate support subflow $M \subseteq S$.*

Its output is a report containing recognition judgments, ledger records, missing-field records, failure records, and cannot-claim annotations.

Principle 9234.10 (Recognizer programs are not recognizer evidence)**.** Lean4✓ BEDC.GroundCompiler.Programmat
not evidence

The host interpreter for a recognizer virtual machine is not itself formal recognizer evidence. Formal recognition evidence is the event-flow recognizer, its certificate, its ledger, and its failure behavior. Host execution is an implementation vehicle until covered by compiler certificates or bootstrap obligations.

Definition 9234.11 (Recognizer VM discipline)**.** Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRealizatio

The recognizer virtual machine must enforce the following discipline:

1. *no external structured input replaces decoded event flow;*
2. *no manifest, schema, table, label, file path, or host object becomes a formal package, certificate, theorem, chapter, or compiler;*
3. *every nontrivial recognition result carries a ledger or a missing-ledger failure;*
4. *every failed recognition records the field or soundness condition that failed;*
5. *every overinterpretation risk receives a cannot-claim annotation.*

Theorem 9234.12 (Recognizer VM realizes the P_5 through P_8 interface)**.** Lean4✓ BEDC.GroundCompiler.Programm
realizes p5 p8

A recognizer virtual machine satisfying Definition 9234.11 can implement the executable interface of P_5 through P_8 without accepting external package objects, certificate objects, theorem labels, chapter trees, or manuscript files as formal input.

Proof. P_5 requires package-flow and NameCert-flow recognition; P_6 requires strength, DerivCert, and AcceptGate recognition; P_7 requires statement, proof, theorem, and proof-checking recognition; P_8 requires chapter and manuscript recognition. In each case the formal inputs are decoded event flows and generated recognizer flows. The recognizer virtual machine interprets those flows and emits judgments, ledgers, field-status records, and failure records. Since the machine never treats external structured data as formal input, it realizes the executable interface while preserving the no-hidden-input boundary. □

9234.4 Certificates, proofs, and acceptance

Definition 9234.13 (Certificate checker layer)**.** Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRealization

The certificate checker layer is the executable layer that checks recognized certificate flows and proof-checker flows. It covers:

1. *package and NameCert field completeness;*
2. *DerivCert field completeness and soundness;*
3. *AcceptGate composition from matching NameCert and DerivCert flows;*
4. *theorem proof checking against statement and dependency flows;*
5. *chapter and manuscript soundness checks;*
6. *compiler certificate and global verification checks.*

Principle 9234.14 (Programs check certificates; they do not replace them)**.** Lean4✓ BEDC.GroundCompiler.Progr
program not certificate evidence

An executable checker may verify a certificate-flow or proof-flow, but it may not replace the certificate-flow or proof-flow. In particular, successful host computation is not the same judgment as recognized proof-flow, recognized DerivCert-flow, recognized AcceptGate-flow, or recognized theorem-flow.

Proposition 9234.15 (Acceptance is certificate-mediated in programs)**.** Lean4✓ BEDC.GroundCompiler.Programm
certificate mediated

In a programmatic realization, accepted object code and accepted theorem code can be emitted only from recognized acceptance or theorem flows satisfying their certificate and proof obligations.

Proof. Accepted object code at P_6 is compiled from a sound AcceptGate-flow. The AcceptGate-flow must include matching NameCert-flow, DerivCert-flow, strength-flow, and gate seal. Accepted theorem code at P_7 is compiled from an accepted theorem-flow whose proof and certificates have been checked by generated flows. Thus the executable program emits acceptance codes only after the corresponding event-flow evidence is present. □

9234.5 Self-hosting route

Definition 9234.16 (Programmatic self-hosting route)**.** Lean4 (definition) BEDC.GroundCompiler.ProgrammaticReal

The programmatic self-hosting route consists of:

1. *a host reference runtime for the executable channel and report layers;*
2. *event-flow descriptions of the channel rules and recognizer rules;*
3. *a recognizer virtual machine for event-flow recognizers;*
4. *compiler-flow recognition for the compiler description;*
5. *compiler certificate-flow recognition for its behavior;*
6. *self-compilation-flow recording $C : C \mapsto C'$;*
7. *behavior-equivalence-flow recording $C' \equiv_{\text{comp}} C$;*
8. *bootstrap-obligation flows for every remaining host reliance.*

Principle 9234.17 (Self-hosting is not mere execution)**.** Lean4 ✓ BEDC.GroundCompiler.ProgrammaticRealization hosting not execution

Running a program that compiles a compiler description is not yet self-hosting. Self-hosting requires certified compiler-flow, self-compilation-flow, behavior-equivalence-flow, self-compilation ledger, no-host-leak audit, and complete bootstrap-obligation accounting.

Theorem 9234.18 (Programmatic realization reaches P_9 only with compiler evidence)**.** Lean4 ✓ BEDC.GroundCompiler.ProgrammaticRealization.programmatic realization reaches p9 with evidence

An executable implementation reaches the P_9 status only when its compiler identity, compiler behavior, self-compilation, behavior equivalence, bootstrap obligations, no-host-leak audit, and global verification status are represented by recognized event-flow evidence.

Proof. P_9 is defined by compiler-flow recognition, compiler certificate-flow, bootstrap-obligation-flow, self-compilation-flow, behavior-equivalence-flow, self-hosting ledger, and global verification-flow. Ordinary host execution supplies none of these as formal event-flow judgments. Therefore the executable implementation reaches P_9 only when these structures are represented and recognized inside the event-flow system. □

9234.6 Implementation architecture

Definition 9234.19 (Four implementation layers)**.** Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRealizat

A faithful implementation architecture has four layers:

1. *an executable runtime layer for encoding, decoding, legality checking, reporting, and tests;*
2. *a formal relation layer exposing event-flow relations such as legal stream, compile, decode, recognition, checking, and acceptance;*
3. *a recognizer and certificate layer interpreting recognizer flows, certificate flows, proof-checker flows, ledgers, failures, and seals;*
4. *a self-hosting and global verification layer tracking compiler-flow, compiler certificates, self-compilation, behavior equivalence, bootstrap obligations, and global status.*

Definition 9234.20 (Implementation module partition). Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRea

A concrete implementation may be partitioned into modules for:

1. *channel encoding, decoding, legality, and round-trip tests;*
2. *source reports, adjacent-pair reports, normalizer candidates, motif reports, and metric reports;*
3. *recognizer virtual machine execution;*
4. *package, NameCert, DerivCert, AcceptGate, theorem, chapter, manuscript, and compiler recognizers;*
5. *certificate soundness, ledger extraction, failure records, status records, and cannot-claim audits;*
6. *bootstrap obligations, self-hosting records, behavior equivalence, and global verification.*

This partition is engineering structure only. It is not a formal source of BEDC objects.

Proposition 9234.21 (Automatic computation has bounded authority). Lean4 ✓ BEDC.GroundCompiler.Programm computation bounded authority

The executable implementation may automatically compute channel round trips, source reports, candidate listings, field-status reports, missing-field reports, undefined metric items, cannot-claim annotations, and any recognized judgment whose recognizer and checker evidence is present. It may not automatically assert mathematical identity, object acceptance, theoremhood, bridge certification, or full self-hosting status without the required event-flow certificates.

Proof. The first group consists of finite computations over decoded event flows and already recognized evidence. The second group crosses into certificate-mediated or proof-mediated judgments. By the prototype ladder definitions, those judgments require generated recognizers, certificate flows, proof-checker flows, ledgers, gates, or compiler verification flows. Hence automatic computation has bounded authority. □

9234.7 Deliverable ladder

Definition 9234.22 (Executable deliverable ladder). Lean4 (definition) BEDC.GroundCompiler.ProgrammaticRealiz

The executable deliverable ladder is:

1. *channel encode, decode, legality, and round-trip;*
2. *source-event report;*
3. *source-level adjacent-pair and normalizer-candidate ledger report;*
4. *motif candidate report with a small recognized motif example;*
5. *metric report under an explicit metric protocol;*
6. *package and NameCert recognizer for a small certificate example;*
7. *DerivCert and AcceptGate recognizer for a small accepted-object example;*
8. *theorem-flow recognizer and proof-checker-flow for a channel round-trip theorem;*
9. *chapter-flow recognizer for a compiler chapter;*
10. *compiler-flow and self-hosting scaffold with explicit bootstrap obligations.*

Principle 9234.23 (Executable completeness is not certificate completeness)**.** `Lean4✓` `BEDC.GroundCompiler.Prog` `not certificate complete`

Completing the executable deliverable ladder does not by itself discharge all certificate obligations. It supplies a running system in which obligations are explicit, checkable, and reportable.

9234.8 Summary

The prototype ladder is programmatically realizable. The channel layer is a small deterministic streaming compiler. The report layers are finite computations over decoded event flows. The recognition layers are implemented by a recognizer virtual machine whose formal inputs are event-flow recognizers. The certificate and proof layers are implemented by checkers that validate event-flow certificates and proof flows. The self-hosting layer is implemented only when compiler-flow, compiler certificate-flow, self-compilation-flow, behavior-equivalence-flow, bootstrap obligations, no-host-leak audit, and global verification-flow are all represented in the event-flow system.

The central implementation boundary is:

programs may implement the mechanism; programs do not replace certificates.

Chapter 9235

Self-Compilation Surface

BEDC's reflection capabilities live on a stratified surface. This chapter distinguishes what BEDC formally achieves (L1, L4-mid), what it explicitly does not claim (L4a, L4c), and the research boundaries of the MetaCIC reconstruction. Subsequent sections record each stratum's current scope and the meta-theoretic constraints (Godel-Tarski) that fix its ceiling.

9235.1 The L1-L4 Stratification

The phrase "BEDC compiles itself" has several inequivalent readings. The paper uses a stratified reading rather than a single slogan. Each layer states which object is visible, which checker acts on it, and which ceiling prevents a stronger reading.

L1 is static paper-Lean alignment. A paper site names a Lean target through '`\leanchecked`', '`\leanvariant`', '`\leanstmt`', '`\leandef`', or a closure-status '`\leantarget`'. The external audit script '`lean4/scripts/bedc_ci.py audit`' reads those names and checks that the referenced Lean declarations exist under '`lean4/BEDC`'. L1 therefore does not ask BEDC to prove its own metatheory. It asks the repository to expose a typed correspondence between paper claims and Lean declarations, with failed name resolution treated as a build-time fault.

L1 has three important limits. First, it is syntactic at the boundary between paper and Lean: a marker can name a declaration, but it does not itself explain the declaration's proof. Second, the audit is a Python-side repository check, not a Lean theorem about LaTeX. Third, the closure axis and the verification axis remain distinct: a Lean target can be checked while the paper-side theory claim is only scoped, and a paper claim can be theoretically delimited while its formal marker is only a target.

L2 is internal base reflection. At this layer BEDC represents selected parts of its own kernel vocabulary as Lean data and proves internal self-consistency facts about them. The base reflection surface covers package reflection, signature-object reflection, domain reflection, token uniqueness replacements, and witness-chain descriptions of generated sameness. Its job is not to certify the whole Lean kernel. Its job is to show that BEDC can place its own finite kernel objects under typed constructors and prove the advertised local coherence facts inside CIC.

In symbols, the intended shape of L2 is:

$$\text{finite BEDC object} \longmapsto \text{Lean-coded reflection datum},$$
$$\text{reflection datum} \longmapsto \text{local coherence theorem}.$$

The arrow is intentionally local. It does not assert that every paper sentence, every Lean declaration, or every semantic reading of BEDC is reflected. It asserts that the finite kernel

interfaces identified by the base-reflection surface admit typed internal witnesses.

L3 is the NameCert obligation surface. Here BEDC records that concrete mathematical interfaces expose obligation packets rather than finished external mathematics. Holonomy, KKT, and SpectralMeasure are representative cases: their Lean declarations register the shape of the name-certificate obligation, not a public closure theorem for the surrounding domain. The L3 surface is therefore a disciplined boundary between a typed internal packet and the mathematical work required to discharge a domain-specific certificate.

L3 matters for self-compilation because it prevents reflection from becoming a claim-laundering device. A reflected name-certificate packet can say, in Lean, which obligations belong to the interface. It cannot, merely by being named, turn those obligations into domain theorems. The packet is a typed surface for inspection and later discharge; it is not a bridge from syntax to truth.

L4 is not a single level. It splits into four readings, only one of which is claimed here.

L4a is an axiom-free completeness theorem for BEDC inside CIC. This would say that BEDC internally proves the complete soundness or completeness of the host logic sufficient to certify all of its own proof checking. BEDC does not claim L4a. Since BEDC is formalized inside the host Calculus of Inductive Constructions, Tarski undefinability and Godel's second incompleteness theorem block the relevant global truth and consistency assertions from being obtained as axiom-free internal theorems of the same strength.

L4-mid is the finite type-checked Manifest of paper-anchored theorems. This is the achieved self-compilation surface. BEDC contains a Lean-side manifest whose entries pair paper labels, Lean names, propositions, and proof terms. The dependent field '`proof : prop`' makes type correctness a kernel obligation when the manifest elaborates. The observable theorem '`BEDC.Manifest.manifest_wellformed` then exposes, for every manifest entry, a witness of the registered proposition.

L4b is an elaboration-time Lean audit that replaces the external Python name audit. Such a layer would move more of L1 from repository tooling into Lean's own elaboration and reflection facilities. The present chapter does not claim L4b. It isolates the target shape: Lean-visible declarations, Lean-visible paper anchors or generated manifest entries, and an elaboration-time failure mode when the surface does not match.

L4c is a bijection between the theory-closure axis and the verification-status axis. BEDC rejects L4c as a matter of governance. Paper-side closure and Lean-side verification are intentionally independent coordinates. Their joint reporting is a product condition, not a fixed point:

$$\mathrm{VerifiedGate}(N, c, v) := \mathrm{TheoryGate}(N, c) \wedge \mathrm{FormalStatus}(N, v).$$

This product form is part of the self-compilation surface. It lets BEDC say exactly which fragment is internally checked, which fragment is theoretically closed, and which bridge remains absent, without collapsing those statements into a single certification number.

9235.2 Internal Manifest

The internal Manifest is the L4-mid surface in its most concrete form. It is a finite Lean object that records selected paper-anchored theorem claims as kernel-checked entries. Each entry carries a paper label, a fully qualified Lean name, the proposition being recorded, and a proof of that proposition.

The core shape is the Lean structure '`BEDC.Manifest.ClaimEntry`':

$$\begin{aligned}\text{ClaimEntry} = \{\, &\text{paper_label} : \text{String},\\ &\text{lean_name} : \text{String},\\ &\text{prop} : \text{Prop},\\ &\text{proof} : \text{prop}\,\}.\end{aligned}$$

The last field is the essential one. The manifest does not store an untyped string saying that a proof exists. It stores a proof term whose type is the entry's own proposition. When Lean elaborates the list of entries, the kernel checks that each proof term inhabits the proposition supplied by that same entry. A malformed pair cannot be silently accepted as a well-typed entry.

This makes Manifest wellformedness kernel-decidable in the practical sense used by this paper: the Lean kernel either accepts the manifest term or rejects it during checking. The paper does not claim that BEDC internally decides arbitrary truth. It claims the finite manifest has a type-checking boundary. For each entry already present in the finite list, Lean checks the displayed proof against the displayed proposition.

The Manifest therefore separates three surfaces that are easy to confuse. The paper label is a human-facing anchor into the LaTeX document. The Lean name is a string that records the declaration intended by the entry. The proposition and proof field are the kernel-facing content. The manifest theorem below speaks about the kernel-facing content; repository audits continue to police paper marker names and file-level correspondence.

Lean4 ✓ `BEDC.Manifest.manifest wellformed`

The theorem '`BEDC.Manifest.manifest_wellformed`' is an observable corollary of this dependent representation. It says that every entry in '`BEDC.Manifest.manifest`' has a nonempty proposition:

$$\forall e,\ e \in \text{manifest} \longrightarrow \text{Nonempty}(e.\text{prop}).$$

Its proof is direct: the entry already contains '`e.proof`', so the witness is '`\langle e.proof\rangle`'. The content is not hidden in that short proof term. The work is performed by the typing of '`ClaimEntry`' and by the kernel's acceptance of each manifest entry.

This is why the Manifest is L4-mid rather than L4a. It is internal, finite, and type checked. It does not define Lean truth for Lean inside Lean. It does not prove that every possible BEDC paper claim has a corresponding proof. It does not certify arbitrary external mathematics. It certifies that the listed paper-anchored theorem entries have proof terms accepted by the host kernel.

The Manifest also clarifies the relation between paper governance and Lean verification. A Manifest entry can support a paper marker and can make the corresponding proof term visible as data. It does not erase the need for the paper-side closure fields: '`\theoryclosure`', '`\scopeclosed`', '`\formalstatus`', '`\bridgestatus`', '`\notclaimed`', and '`\upgradepath`'. Those fields state the theoretical scope around the claim. The Manifest states that selected Lean propositions have typed proof terms.

Consequently, the self-compilation claim at this layer is precise. BEDC can carry a finite manifest of its own paper-anchored Lean theorems as Lean data, and the kernel can reject ill-typed entries. That is a real internal reflection surface, but it is deliberately not a global self-certification theorem.

9235.3 MetaCIC Reconstruction

The MetaCIC reconstruction surface asks how much of the host calculus can be represented as BEDC data and predicates. It is a reconstruction analysis, not a replacement of the host. The relevant namespace is '`BEDC.MetaCIC`', whose role is to name the current surface on which CIC syntax, typing, and reduction can be discussed as first-class BEDC objects.

The structural reconstruction layer constructs syntax, contexts, terms, universe annotations as needed, beta-reduction, and typing judgments as inductive objects or inductive predicates over BEDC histories.

Lean4 (definition) `BEDC.MetaCIC.Term`
Lean4 (definition) `BEDC.MetaCIC.BetaStep`
Lean4 (definition) `BEDC.MetaCIC.HasType`

The result is not a claim that BEDC has reimplemented Lean. It is a claim that the grammar and judgment forms of a CIC fragment can be represented directly enough to support quantification and induction over them.

The difference between behavior and structure is central. A Turing-machine simulation of Lean execution can describe observed runs:

$$\exists t,\ \mathrm{MachineTrace}(\mathrm{Lean}, x, t) \wedge \mathrm{Accepts}(t).$$

Such a statement is behavioral and trace-based. It predicts or records a finite execution. A direct inductive predicate has a different form:

$$\Gamma \vdash_{\mathrm{MetaCIC}} t : T.$$

This is a first-class judgment. BEDC can quantify over contexts, terms, and derivations, and can prove results by induction on the derivation. TM simulation gives Sigma-one execution traces; direct reconstruction gives typed judgment objects. The base sort judgment is inhabited in the empty context.

Lean4 ✓ `BEDC.MetaCIC.sort in empty ctx`

The same fragment carries concrete checked examples. A polymorphic identity term over 'Sort' is well typed, and its Pi type is itself a represented typing judgment.

Lean4 ✓ `BEDC.MetaCIC.id sort well typed`
Lean4 ✓ `BEDC.MetaCIC.pi sort sort well typed`

Applying that identity to 'Sort' has the expected represented application typing, and the corresponding beta step contracts to the argument.

Lean4 ✓ `BEDC.MetaCIC.id sort applied`

The construction surface covers syntax and judgments. It supports statements of the form "for every represented derivation with this constructor shape, the following structural property holds." It also supports explicit boundary claims: which syntax is represented, which conversion relation is present, which typing rules are included, and which host-kernel features remain outside the fragment.

The metatheory layer covers structural claims about represented derivations. Subject reduction says that typing survives reduction:

$$\Gamma \vdash t : T \wedge t \to_\beta u \longrightarrow \Gamma \vdash u : T.$$

Confluence says that divergent reductions can be joined. These theorems are structural metatheorems about the represented calculus, and their checked fragments must be stated over the explicit syntax, reduction, and typing relations carried by the reconstruction.

The encoded confluence surface separates ordinary multi-step beta reduction from parallel reduction. BetaStarStep is the reflexive-transitive closure of one-step β; BetaParallel is a one-round parallel relation with identity clauses for variables and sort, componentwise clauses for application, lambda, and Pi, and a beta clause that contracts a lambda redex after reducing the domain, body, and argument in parallel. The checked diamond material is therefore shape-local. The sort and variable sources have genuine joins because their parallel reductions can only return the same atom, while application, lambda, and Pi constructors have join assemblers when their component joins are supplied. At the ordinary multi-step level, the atom cases are closed by showing that a beta-star path out of a sort or variable has the same endpoint.

Lean4 ✓ `BEDC.MetaCIC.betaStar var join`

Lean4 ✓ `BEDC.MetaCIC.betaStar sort join`

9235.3.1 Encoding Overview

The encoding in `BEDC.MetaCIC` is a direct Lean representation of the fragment. `Term` is an inductive syntax type with five constructors: `var`, `app`, `lam`, `pi`, and `sort`. Contexts are the abbreviation `Ctx := List Term`, and `Ctx.lookup` synchronizes lookup with de Bruijn shifting across binder slots.

The substitution and shift operations are the standard de Bruijn operations: descent below a binder raises the depth by one, and the substituted value is shifted by `shift 0 1`. The typing predicate `HasType` is inductive and uses a shift-aware binder slot, namely shift 0 1 dom, when checking lambda and Pi bodies.

Reduction is split into three inductive relations. `BetaStep` gives single-step β contraction with congruence rules; `BetaStarStep` gives the multi-step closure; and `BetaParallel` gives the Tait–Martin-Löf parallel relation. Closure is represented by the indexed inductive judgment `ClosedAt`. All of these objects are ordinary Lean 4 inductive types or definitions, use no axioms, and have exactly the strength of the host CIC that checks them.

9235.3.2 Syntax Constructor Table

Constructor	Description
`Term.var i`	de Bruijn variable, $i \in \mathbb{N}$
`Term.app f a`	function application $f\ a$
`Term.lam d b`	abstraction $\lambda x : d.\ b$ with domain d and body b
`Term.pi d c`	dependent product $\Pi x : d.\ c$
`Term.sort`	single universe sort literal

Each constructor is a pure inductive constructor, with no host metaprogramming.

9235.3.3 Paper vs Mechanization

The paper and the mechanized Lean module describe the same reconstruction surface at different levels of notation. The paper uses LaTeX judgments such as $\Gamma \vdash t : T$ to present typing, while the mechanization uses an inductive proposition such as `HasType Gamma t T` to make the judgment a host-checked object.

Paper-side markers record the correspondence explicitly. A marker written as `\leanchecked`, `\leandef`, or `\leanstmt` names the Lean target that carries the corresponding theorem, definition, or statement-only surface.

Thus each paper-claimed theorem has a visible formal status: direct theorem via `\leanchecked`, inductive or definitional encoding via `\leandef`, or statement-only target via `\leanstmt`. The paper therefore does not silently identify prose, notation, and kernel objects.

`bedc_ci.py audit` continuously checks those marker names against `lean4/BEDC/`. A missing or renamed Lean target is an audit failure, so the paper text and mechanized module cannot drift silently.

9235.3.4 Host Fragment Coverage

The represented host fragment is the dependent-function core of CIC, with explicit de Bruijn variables, one sort constructor, application, lambda, Pi, beta reduction, and a shift-aware typing judgment. The coverage boundary is:

Host feature	BEDC.MetaCIC	Status
`Term.app` / `lam` / `pi` / `var` / `sort`	`Term` inductive	covered
β-reduction	`BetaStep` + `BetaStarStep` + `BetaParallel`	covered
Typing judgment	`HasType`	covered, shift-aware substrate
Universe polymorphism	none	not covered
η-conversion	none	not covered
Inductive types beyond Pi/sort	none	not covered
Equality types (`Eq`)	none	not covered
Definitional unfolding	`substitute` composition	partial coverage
Reduction strategies (CBV/CBN)	non-deterministic `BetaStep`	abstract

Thus `BEDC.MetaCIC` is not a model of the full host kernel. It is the host-checked representation of the small CIC fragment needed for the local syntax, reduction, typing, and binder-mechanics claims.

9235.3.5 Correctness Claims

The correctness meaning of the `BEDC.MetaCIC` mechanization is internal to the host calculus. `BEDC.MetaCIC.HasType` is an inductive proposition. An inhabitant of this proposition is therefore a derivation object accepted by the host kernel. If the host is consistent, such inhabitants are real typing derivations for the represented CIC fragment whose rules are given by the constructors of `HasType`.

`BEDC.MetaCIC.BetaStep` has the same status for one-step reduction. It is a host `inductive Prop`, and each inhabitant is a kernel-checked witness of exactly one constructor-generated single-step beta reduction in the represented syntax.

The operations `substitute` and `shift` are host `def` declarations. Their closed-form values are computed by the host kernel through small-step simplification of those definitions, rather than by an external evaluator or a meta-level convention.

Consequently, theorems such as `sort_in_empty_ctx`, `id_sort_well_typed`, and the related MetaCIC examples are host-verified claims about the represented CIC fragment. Host consistency implies the soundness of those BEDC.MetaCIC derivations and reductions within that fragment.

9235.3.6 Representation vs Mockup

A mockup form places a host program in front of a separate syntax. For example, one may write a Lean tactic that runs a BNF parser, builds an external abstract syntax tree, and then reports whether the parsed expression satisfies a hand-coded relation. Such a program may be useful as tooling, but its behavior is not thereby formally tied to the host's actual type checker. The host checks the tactic implementation and any theorem it finally emits; it does not make the parser's accepted syntax into the host's own typing judgment.

`BEDC.MetaCIC` uses a representation form instead. `Term` is a host data type, and `HasType` is a host inductive proposition. When the host kernel elaborates BEDC theorems about `HasType`, it is at the same time verifying the represented typing claims and their proof consequences.

Thus `BEDC.MetaCIC` is not about mimicking the host from outside. It directly specifies the grammar and typing rules of a host-CIC fragment, and lets the host kernel check the consequences of those specifications.

9235.3.7 Positioning Within BEDC

The `BEDC.MetaCIC` namespace occupies a calculus-reconstruction position within the BEDC codebase. `BEDC.FKernel` is the foundational kernel layer, carrying the `BMark`, `BHist`, and probe primitives. `BEDC.Derived` builds derived mathematical structures such as real objects, groups, modules, and related packets on top of that base.

`BEDC.Manifest` is the paper-anchored theorem registry: it records named theorem entries as kernel-typed artifacts. `BEDC.Reflection` handles closure and certificate forms, including package-facing identification principles and name-certificate surfaces. These modules give the project its object layer, registry layer, and certificate layer.

`BEDC.MetaCIC` is complementary to those layers rather than downstream of them. It gives an inductive description of a CIC fragment itself: terms, contexts, reduction, typing, and closure as represented syntax and judgments. It therefore formalizes a calculus fragment in isolation from the derived structure, manifest, and reflection APIs, while sharing the same axiom-free Lean checking discipline as the rest of BEDC.

MetaCIC does not replace the host CIC or serve as a second kernel beside Lean. It represents a fragment inside the host, so theorems about represented derivations remain host-checked BEDC theorems. The foundational guarantee is the same project-wide one: the surface is built from inductive definitions and checked proofs, with no additional axioms.

9235.3.8 Stratification Recap

Within `BEDC.MetaCIC`, L1 is the registration level. BEDC records paper-claimed theorems as host-checked Lean data through `BEDC.Manifest`, and the MetaCIC reconstruction is the syntactic surface on which the registered calculus-facing claims can be named.

L4-mid is the internal representation level. BEDC expresses the inductive structure of a CIC fragment, together with selected metatheory, as Lean objects and predicates. MetaCIC therefore supplies representation and derivation forms; it does not attempt to prove the host's consistency.

L4a is blocked by the Godel boundary. Since BEDC is checked inside the host, BEDC cannot prove the host's own consistency from within that same host. MetaCIC keeps this boundary intact and does not turn reconstruction into a consistency proof for Lean.

L4c is rejected. BEDC does not replace the host CIC, and MetaCIC is not an alternative foundation. The reconstruction is an inside-host formalization of syntax, reduction, typing, and selected metatheory, all under the checking authority of the host calculus.

9235.3.9 Fragments and Limitations

`BEDC.MetaCIC` represents a minimal CIC syntactic fragment. The included term grammar has variables, a single sort, dependent function types, lambda abstraction, and application. Binders use de Bruijn indices rather than names, and the substrate is shift-aware: context lookup, binder descent, and substitution all account for the index movement caused by crossing binders.

Several host-CIC features are outside this represented fragment. There is no universe hierarchy or universe polymorphism: the syntax has one sort and no `Type i` annotation. There are no standalone inductive types beyond Pi and sort, so objects such as natural numbers or lists are not represented as primitive inductive families in this layer. There is also no represented equality type, no inference hint or coercion layer, and no eta-conversion: `BetaStep` contains beta contraction and beta congruence only.

The design intent is correspondingly narrow. The fragment is large enough to demonstrate the self-compilation idea as syntax, typing, reduction, and binder mechanics internalized as BEDC data and predicates. It is not a full host-CIC model and does not aim to cover Lean's complete kernel language.

9235.3.10 Typing Rules

The represented typing judgment HasType $\Gamma\ t\ A$ is generated by five rules. They are the whole typing surface for the local MetaCIC fragment:

$$
\begin{aligned}
\text{sortRule}: &\quad \Gamma \vdash \text{sort} : \text{sort},\\
\text{varRule}: &\quad \Gamma.\text{lookup}\ i = \text{some}\ A \Rightarrow \Gamma \vdash \text{var}\ i : A,\\
\text{piRule}: &\quad \Gamma \vdash \text{dom} : \text{sort},\ (\text{shift}\ 0\ 1\ \text{dom}) :: \Gamma \vdash \text{cod} : \text{sort} \Rightarrow \Gamma \vdash \pi\,\text{dom}\,\text{cod} : \text{sort},\\
\text{lamRule}: &\quad \Gamma \vdash \text{dom} : \text{sort},\ (\text{shift}\ 0\ 1\ \text{dom}) :: \Gamma \vdash \text{body} : \text{cod} \Rightarrow \Gamma \vdash \lambda\,\text{dom}\,\text{body} : \pi\,\text{dom}\,\text{cod},\\
\text{appRule}: &\quad \Gamma \vdash f : \pi\,\text{dom}\,\text{cod},\ \Gamma \vdash a : \text{dom} \Rightarrow \Gamma \vdash f\,a : \text{substitute}\ 0\ a\ \text{cod}.
\end{aligned}
$$

9235.3.11 Typing Inference Rules Table

Rule	Conclusion
sortRule	$\Gamma \vdash \text{sort} : \text{sort}$
varRule	$\Gamma.\text{lookup}\ i = \text{some}\ A \Rightarrow \Gamma \vdash \text{var}\ i : A$
piRule	(shift-aware) $\Gamma \vdash \text{pi}\ d\ c : \text{sort}$ from $\Gamma \vdash d : \text{sort}$, $(\text{shift}\ 0\ 1\ d) :: \Gamma \vdash c : \text{sort}$
lamRule	(shift-aware) $\Gamma \vdash \text{lam}\ d\ b : \text{pi}\ d\ \text{cod}$ from the same Pi-rule binder slot and body type cod
appRule	$\Gamma \vdash f\ a : \text{substitute}\ 0\ a\ \text{cod}$ from $\Gamma \vdash f : \text{pi}\ d\ \text{cod}$, $\Gamma \vdash a : d$

The shift-aware binder slot appears in the Pi and lambda rules as $(\text{shift}\ 0\ 1\ d) :: \Gamma$, rather than as $d :: \Gamma$.

9235.3.12 Typing Relation vs Host CIC

`BEDC.MetaCIC.HasType` is an inductive proposition inside the host calculus. It represents a typing relation for the MetaCIC fragment: an inhabitant of HasType $\Gamma\ t\ A$ is a derivation object built from the five constructors listed above, not the algorithm that checks Lean terms.

The checking algorithm belongs to the host CIC kernel. That kernel elaborates and type-checks Lean terms inside the implementation, with executable kernel code running in the OCaml/Rust compiler/runtime environment. BEDC does not replace that kernel; every theorem about `HasType` is accepted only when the host kernel checks its proof term.

The represented relation is intentionally narrower than host typing. It describes the fragment's Pi, lambda, application, variable, and sort rules, together with explicit beta reduction. It does not simulate the full host reduction and elaboration stack: eta-conversion, universe polymorphism, and inductive types beyond the represented Pi/Sort fragment are outside this local relation.

The main sanity condition is proof-theoretic. `HasType` is axiom-free, and each displayed `HasType` derivation is a Lean term checked by the host kernel. The strict axiom-purity audit also verifies that the checked MetaCIC theorem dependencies do not use `propext`, `Classical.choice`, or `Quot.sound`.

9235.3.13 Context Mechanics

A context is the Lean stdlib list Ctx = List Term. Its entries are represented types, stored as a de Bruijn stack: the head is the type of variable 0, the next entry is the type of variable 1, and so on down the list. The operation Γ.lookup i returns the ith entry of Γ, counted from this top position.

The important invariant is that lookup and binder crossing perform the same index accounting. When a fresh binder contributes a top entry B, the top variable is read directly, while every deeper lookup is raised by shift 0 1:

$$(B :: \Gamma).\text{lookup } 0 = \text{some } B,$$

$$(B :: \Gamma).\text{lookup } (i+1) = \begin{cases} \text{some (shift } 0\ 1\ T), & \Gamma.\text{lookup } i = \text{some } T, \\ \text{none}, & \Gamma.\text{lookup } i = \text{none}. \end{cases}$$

Lean4 ✓ `BEDC.MetaCIC.lookup cons zero`

Lean4 ✓ `BEDC.MetaCIC.lookup cons succ`

Thus the type of a deeper variable automatically accumulates exactly the number of binders crossed. This is the context-side companion to the shift-aware design of substitute under binders.

9235.3.14 Reduction Layer Summary

The reduction layer has three relations. BetaStep is single-step β reduction: root beta plus congruence under application heads and arguments, lambda bodies and domains, and Pi codomains and domains.

Lean4 (definition) `BEDC.MetaCIC.BetaStep`

BetaStarStep is the reflexive-transitive closure, generated by reflexive and one-step extension constructors, with checked transitivity for finite path composition.

Lean4 (definition) `BEDC.MetaCIC.BetaStarStep`

Lean4 ✓ `BEDC.MetaCIC.betaStar trans`

BetaParallel is the Tait–Martin-Löf one-round parallel relation: variables and sort reduce to themselves; Pi, lambda, and application reduce componentwise; beta reduces domain, body, and argument before contraction.

Lean4 (definition) `BEDC.MetaCIC.BetaParallel`

The checked containment spine records reflexivity, BetaStep $\subseteq$ BetaParallel, and BetaParallel $\subseteq$ BetaStarStep.

Lean4 ✓ `BEDC.MetaCIC.betaParallel refl`

Lean4 ✓ `BEDC.MetaCIC.betaStep to parallel`
Lean4 ✓ `BEDC.MetaCIC.betaParallel to betaStar`

9235.3.15 Reduction Relations

Relation	Description
`BetaStep t t'`	single-step β plus congruence in `MetaCIC.Beta`
`BetaStarStep t t'`	reflexive-transitive closure of `BetaStep`
`BetaParallel t t'`	Tait–Martin-Löf parallel reduction

`BetaStarStep` and `BetaParallel` are separate inductive propositions, connected by atom-diamond facts and congruence lemmas. There is no implicit conversion between them.

9235.3.16 No Eta-Conversion

`BEDC.MetaCIC.Beta` contains no η-conversion, either as η-expansion or as η-reduction. Its one-step relation is generated only by root β contraction and the congruence constructors for application, lambda, and Pi terms.

This has concrete consequences. The represented terms `lam d (app f (var 0))` and `f` do not reduce to each other through `BetaStep`, even when `f` is closed. The multi-step relation `BetaStarStep` and the parallel relation `BetaParallel` are built from the same β surface, so they do not blur this boundary into η-equality.

Accordingly, subject reduction and confluence in this reconstruction are statements about β-equivalence only. A calculus containing η-conversion has a different reduction signature: it requires an additional `BetaStep` constructor, such as an `etaStep`, and separate compatibility proofs between that constructor and the existing β metatheory.

The normalization facts are deliberately atomic. sort_no_beta and closed_var_no_beta say that the sort constructor and a variable constructor are not beta-redex heads; there is no one-step reduction whose source has either shape. The typed closed-atom theorem packages this as a normal-form statement: if a closed represented term typed at sort is either the sort atom or a variable atom, then no beta successor exists. In this bounded sense, closed atoms are already normal forms in the represented calculus.

Lean4 ✓ `BEDC.MetaCIC.closed atom normal form`

Strong normalization has a different status. A global strong-normalization theorem strong enough to recover consistency of the represented host calculus meets the Godel-II boundary when the represented calculus is comparable in strength to the host in which the proof is carried out. BEDC can study bounded or fragment-specific termination claims, and it can state which hypotheses make those claims available. It does not obtain an axiom-free global consistency certificate for its own host by internal reconstruction.

The replacement reading would make BEDC the foundation instead of CIC. This is not a claim of the paper. BEDC is formalized inside the host CIC. All Lean theorems in the project are checked by that host. The MetaCIC surface may represent syntax and prove fragment metatheory, but it does not move BEDC outside the host or give BEDC a stronger foundational status than the kernel that checks it.

This yields the governing inclusion:

$$\text{BEDC formalization} \subseteq \text{host CIC}.$$

The inclusion is methodological, not merely notational. A theorem about a represented calculus is a theorem inside the host. It can be useful, precise, and internally quantified, but it cannot serve as a host-independent foundation for the host without crossing the Godel-Tarski boundary.

The MetaCIC reconstruction surface thus complements the Manifest. The Manifest registers finite paper-anchored theorem entries as checked Lean data. MetaCIC reconstruction studies the syntax and judgment structure of the calculus in which such checking is meaningful. Together they explain why "BEDC compiles itself" has a legitimate L4-mid reading and no legitimate L4a or replacement reading.

9235.3.17 Self-Compilation Completeness

In this chapter, self-compilation has a precise BEDC meaning. It does not mean that BEDC compiles BEDC into an executable binary, nor that BEDC replaces the host kernel. It means that BEDC represents enough of a CIC fragment for the paper's quantified judgments about syntax, typing, reduction, closure, and substitution to be stated as Lean-checkable inductive objects and theorems.

The completed mechanized surface covers the paper claims for syntax, typing rules, atom-shape preservation, atom normalization, and closed-domain substitution preservation. These claims are expressed over the represented MetaCIC terms, contexts, beta relations, closure predicate, and typing judgment, so Lean checks them as ordinary theorems about inductively generated data.

The surface does not close substitute preservation for arbitrary binder domains, strong normalization, or a decidable type checker for arbitrary represented terms. Those targets require binder-domain transport, global termination structure, or complete inference coverage beyond the represented fragment used here.

Thus the self-compilation status is L1 plus L4-mid. It is a syntax-and-judgment reconstruction layer together with finite Lean-checked paper claims. It is not L4a, because BEDC is not closed over the host that checks it, and it is not L4c, because BEDC does not replace the host kernel or identify paper closure with kernel verification.

9235.3.18 Host CIC Relation

MetaCIC is represented inside the host Lean calculus, not beside it. The objects named `BEDC.MetaCIC.Term`, `BEDC.MetaCIC.HasType`, and `BEDC.MetaCIC.BetaStep` are implemented as inductive types or inductive predicates in the host CIC. They are therefore first-order objects for the host: Lean can quantify over represented terms, represented typing derivations, and represented beta steps exactly as it quantifies over any other inductively generated data.

The important boundary is axiomatic. The host Lean environment has available constants such as `propext`, `Quot.sound`, and `Classical.choice` when developments choose to depend on them. `BEDC.MetaCIC` does not use those constants. Its syntax, reduction, typing, and checked metatheory are built in the pure CIC fragment by inductive definitions and structural proofs.

The verification relation is one-way. The host CIC kernel type-checks each BEDC theorem. Within that checked development, a MetaCIC typing derivation is an inhabitant of the inductive predicate `HasType`, and reasoning about typing inference is ordinary induction over that predicate. There is no circular dependence: the host checks BEDC, while BEDC encodes a CIC fragment as data and judgments.

This also fixes the Godel-Tarski boundary. `BEDC.MetaCIC` is not outside the host and cannot prove the consistency of the host that checks it. It can state and prove fragment metatheory for the represented calculus, but it does not create a host-independent consistency proof.

9235.3.19 Godel-Tarski Boundary

`BEDC.MetaCIC` does not try to cross the Godel-II or Tarski undefinability boundaries. It is a representation of a CIC fragment inside the host CIC, not an external standpoint from which the host can certify itself.

The typing judgment `BEDC.MetaCIC.HasType` is an inductive proposition in the host. Its inhabitants are host-checked derivation objects for the represented calculus, but the existence of such a predicate does not supply a proof of the host's own consistency without an external consistency assumption for that host.

Likewise, a strong-normalization theorem for the represented calculus, if proved, would be a theorem using the host's full proof strength. In Lean this includes the kernel's accepted structural and well-founded recursion principles and the definitional computation of dependent-function reduction. Such a theorem would establish normalization for the represented syntax under host checking; it would not be a host-independent normalization proof for Lean itself.

Finally, `BEDC.MetaCIC.Term` ranges over represented terms only. It can state and prove claims about this syntax and its generated judgments, but it cannot quantify over every host term and assert that each host term has a normal form without moving beyond the strength of the same host being represented.

9235.3.20 Host Internalization

`BEDC.MetaCIC` is an internalization of the host CIC, not a mock interpreter. `BEDC.MetaCIC.Term` is the host's `inductive Term` type, so represented syntax is genuine host data. `BEDC.MetaCIC.HasType` is the host's inductive proposition for typing derivations, so represented typing is genuine host proposition structure.

The operations `BEDC.MetaCIC.substitute` and `BEDC.MetaCIC.shift` are host `def` declarations. They are therefore genuine host computations on the represented syntax, not external notations or parser artifacts. When the Lean kernel type-checks `BEDC.MetaCIC` theorems, the same kernel is also checking the BEDC `HasType` propositions that occur inside those theorem statements.

This differs from a mock interpreter. BEDC does not stub out a fake parser and then reason about traces of that parser. It directly represents the inductive structure of a CIC fragment as host inductive data, host inductive propositions, and host recursive definitions. Consequently, an inconsistency in the host kernel would imply an inconsistency in BEDC, because BEDC lives inside that host. The converse implication does not follow: a contradiction in the represented BEDC surface would not by itself establish host-kernel inconsistency.

9235.3.21 Axiom Purity Guarantee

The axiom-purity guarantee for `BEDC.MetaCIC` is the same guarantee applied to the rest of the BEDC formalization. At the source level, the project has zero tolerance for the raw `axiom` keyword in `lean4/BEDC/`: `tools/check-axioms.py` scans the BEDC Lean tree for any `axiom` declaration and exits with status 1 when one is found.

At the theorem-dependency level, `BEDC.MetaCIC` is audited together with the other BEDC modules by `python3 lean4/scripts/bedc_ci.py axiom-purity -strict`. The audit uses Lean's `#print axioms` output to inspect the transitive dependencies of each checked theorem. The forbidden dependency set is `Classical.choice`, `Quot.sound`, and `propext`: excluded respectively as excluded-middle or choice dependence, quotient-type axiom dependence, and propositional-extensionality dependence.

More than 9000 BEDC theorem declarations pass this strict audit with zero forbidden dependencies. The formal content used by the MetaCIC reconstruction therefore remains inside the pure CIC fragment checked by Lean's kernel, rather than importing additional Lean stdlib axioms through either declarations or transitive proof dependencies.

9235.3.22 Pure CIC Fragment

`BEDC.MetaCIC` represents a pure CIC fragment. It does not depend on any axiom introduced by the Lean standard library, including `Classical.choice`, `Quot.sound`, and `propext`. This premise is audited by `bedc_ci.py axiom-purity --strict`, which checks each theorem's transitive dependencies against the forbidden axiom set.

The PCIC fragment has the following concrete features:

- no quotient types and no propositional-extensionality strengthening of propositional equality;
- no law of excluded middle and no dependent choice;
- no universe polymorphism;
- `pi`, `lam`, `app`, `sort`, and `var` as the only represented term constructors.

9235.3.23 Encoding Choice

The reconstruction uses a de Bruijn, shift-aware substrate for binders. Named binders are convenient for informal prose, but a checked development must then carry freshness side conditions through weakening, substitution, and binder descent. Those side conditions are not part of the mathematical point of the fragment, so the substrate uses numeric binder positions instead.

Naive de Bruijn encoding is still not sufficient. If a binder simply extends a context by placing dom at the head, as dom :: Γ, then the body sees var 0 both as the binder slot and as an index sharing the outer context convention. Under substitution or descent below another binder, this creates exactly the capture phenomenon the encoding is meant to avoid.

The shift-aware substrate stores the binder slot type as shift 0 1 dom. Inside the body, var 0 therefore points strictly to the binder, while the old context has been moved under that new slot. This matches the host CIC's internal de Bruijn discipline: crossing a binder changes the reference depth, and the represented context records that change explicitly.

The price is that some attractive unconditional preservation statements are not true at this level of representation. In particular, shift adjustment and substitution do not commute freely under an open binder. Substitution preservation therefore carries closure hypotheses in the binder cases; those hypotheses make the shifted domain stable and let the represented typing context line up without capture.

9235.3.24 Closure Discipline

The main substitute-preservation theorems are closure-conditioned because the binder domain participates directly in the represented context. In the shift-aware typing rule, a lambda or Pi body is checked under (shift 0 1 dom) :: Γ, so preservation under substitution must keep that domain aligned with the context obtained after the binder is crossed.

The closed-domain hypothesis supplies exactly that alignment. When ClosedAt 0 dom holds, the closed-shift lemma gives shift 0 1 dom = dom. The lift/lower path inside substitute then has a stable binder entry: descending under the binder raises the substitution depth and shifts the incoming term, but the domain stored in the context is unchanged.

For open binder domains, the same preservation shape requires a general shift-substitute commutation theorem strong enough to transport the open domain through both operations. That commutation theorem is not part of the available substrate. The explicit condition keeps the theorem statement at the proved boundary rather than hiding a missing transport principle.

ClosedAt is the inductive predicate defined in `BEDC.MetaCIC.ClosedTerm`: var i is closed at n when $i < n$, and compound terms are closed exactly when their immediate subterms are closed at the appropriate binder depth.

9235.3.25 Shift Mechanism Detail

The operation shift cutoff amount t takes a cutoff threshold, an addition amount, and a represented term t. On a variable var i, it compares i with the cutoff. If $i \geq$ cutoff, the variable is mapped to var $(i + \text{amount})$; if $i <$ cutoff, it is left unchanged.

The operation is structural on applications and sorts. Under a lambda or Pi binder, the domain is shifted at the same cutoff, while the body is shifted at cutoff $+ 1$. The extra unit records the binder slot that now occupies one de Bruijn index position before the surrounding variables.

This is the mechanism behind capture-aware substitution. When substitute descends below a binder, it raises the substitution depth and shifts the incoming term by shift 0 1, so the incoming term passes around the binder slot instead of being captured by it.

The core lemmas state the bookkeeping equalities used downstream: `shift_closed` says shifting at a closed boundary leaves the term unchanged; `substitute_shift_at_eq` says substitution at depth n undoes a same-depth one-step shift; and `substitute_closed` says substitution at a closed boundary leaves the term unchanged.

9235.3.26 Substitution Operation

The operation substitute depth v t takes a de Bruijn replacement depth, an incoming replacement value v, and an original represented term t. It is the capture-aware operation used by beta contraction and by the substitution-preservation surface.

On a variable var i, the operation compares i with the replacement depth. If $i =$ depth, the variable is replaced by v. If $i >$ depth, the variable crosses the removed binder slot and becomes var $(i - 1)$. If $i <$ depth, the variable is below the active replacement boundary and is left unchanged.

Lean4 ✓ `BEDC.MetaCIC.substitute var zero`

Lean4 ✓ `BEDC.MetaCIC.substitute var succ zero`

The operation is structural on application, sort, lambda, and Pi terms. Under a binder, the body is processed at depth $+1$, and the incoming value is simultaneously raised to shift 0 1 v. This synchronized shift makes v pass the new binder slot without capture.

The syntax-level lemmas summarize the arithmetic used downstream: `substitute_var_zero` gives replacement at depth zero; `substitute_var_succ_zero` gives the positive-variable decrement; `substitute_closed` says closed terms are unaffected by substitution; and `substitute_shift_at_eq` says substitution at depth n cancels a same-depth one-step shift.

Lean4 ✓ `BEDC.MetaCIC.substitute closed`

Lean4 ✓ `BEDC.MetaCIC.substitute shift at eq`

9235.3.27 Glossary

The following terms name the local reconstruction conventions used in this section:

- A shift-aware binder context is a lambda or Pi binder slot whose stored domain is shift 0 1 dom rather than dom; this choice prevents capture by var 0.
- A closed binder domain is the domain of that lambda or Pi satisfying ClosedAt 0.
- BetaStarStep is the reflexive-transitive closure of one-step BetaStep.
- BetaParallel is Tait–Martin-Löf style parallel reduction.
- An atom shape is a term whose shape is either sort or var i.

9235.3.28 Construction Style

The MetaCIC layer is built in the first instance from pure inductive objects. Term is an inductive syntax type, while HasType, BetaStep, BetaStarStep, BetaParallel, and ClosedAt are inductive predicates. No constructor in this surface depends on an additional axiom.

Binders are represented by de Bruijn indices. Variables have the form Term.var i for an index i, and substitution is the explicit operation substitute depth v t. Lambda and Pi binders are handled by the same shift-aware discipline used by the typing judgment: crossing a binder raises the depth and shifts the incoming term so that capture is not introduced by representation.

The main substitution-preservation statements are conditioned by closure. They work under hypotheses such as ClosedAt 0 dom, because a closed binder domain is stable under the binder shift: shift 0 1 dom = dom. This is the local fact that lets the represented binder context and the substituted context line up.

Many checked lemmas are intentionally statement-shape lemmas. Names such as `betaSubstitutionPreservat` `betaSubstitutionPreservation_pi_body`, `betaSubstitutionPreservation_lam_body`, and `betaSubstitutior` isolate the term shape under which substitution preservation is proved. The full picture is assembled from these shape-local facts rather than from an opaque global argument.

9235.3.29 Derivation Style

The MetaCIC proof layer follows a derivation style tied closely to the inductive term and judgment constructors. Most local lemmas proceed by case analysis on the represented term or typing witness. In the atom cases of subject reduction and substitution preservation, a proof often reduces to term-level inversion by `cases ht`: when the source term is sort or var i, the impossible typing constructors are eliminated and the surviving constructor exposes exactly the required premises.

Many theorems are therefore shape-restricted. A case-specific preservation result may assume a syntactic side condition such as $t = \text{sort} \vee \exists i,\ t = \text{var } i$, rather than claiming the unrestricted preservation theorem directly. The global proof surface is assembled by chaining such case lemmas, with separate entries for sort/variable atoms, Pi bodies, lambda bodies, and application bodies.

The proofs are also closure-conditioned. Hypotheses such as ClosedAt 0 dom keep binder-domain shifts from changing the represented type, so the shifted context used by the typing rule agrees with the context obtained after substitution. This is the point at which the de Bruijn bookkeeping becomes a local equality instead of a global transport problem.

Finally, the formalization separates statements from proof instances. Clean targets such as BetaSubstitutionPreservation name the proposition needed downstream, while the checked lemmas prove that target under specific term shapes and closure hypotheses. The development does not hide the remaining unrestricted theorem inside one monolithic proof; it exposes the proof-relevant shapes that make each derivation go through.

9235.3.30 Induction Style

The induction style in `BEDC.MetaCIC` follows the object being analyzed. Term induction is used for syntactic properties of represented terms. A proof may proceed by `induction t with | sort => ... | var => ... | app f a ihf iha => ...`, with the recursive hypotheses supplied exactly for the immediate subterms. This is the natural shape for results such as `substitute_closed`, where the claim is driven by the constructors of `Term`.

Judgment induction is used for structural properties of derivations. Proofs about typing and reduction inspect the evidence itself, for example by `induction ht` over a `HasType` witness or by induction over a `BetaStep` witness. This is the appropriate style for subject-reduction case analysis, where each typing or beta constructor exposes the premises that the corresponding preservation case may use.

Most case lemmas are therefore local in shape: for an atom source, a concrete congruence constructor, or a fixed binder form, the required property holds. The main theorem surface is assembled by combining these lemmas with ordinary `cases` and induction over the relevant term or derivation. No proof relies on well-founded recursion beyond Lean's built-in recursion principles for `Nat`, `List`, and the inductive objects defined in the MetaCIC surface.

9235.3.31 Namespace Rationale

The `BEDC.MetaCIC` namespace is divided by proof responsibility rather than by presentation order. `BEDC.MetaCIC.Syntax` contains the syntactic core: Term, shift, substitution, and contexts. `BEDC.MetaCIC.Beta` contains one-step β reduction, while `BEDC.MetaCIC.Typing` contains the typing relation over that syntax. `BEDC.MetaCIC.SubjectReduction` names the subject-reduction theorem suite, so that preservation claims remain downstream of both reduction and typing. `BEDC.MetaCIC.Confluence` contains multi-step reduction and parallel reduction, because join arguments use a different induction shape from one-step contraction. `BEDC.MetaCIC.Normalization` records the atom normal-form facts. `BEDC.MetaCIC.AtomInfer` and `BEDC.MetaCIC.TypedExamples` are demonstration and inference surfaces, kept outside the metatheory namespaces so that examples do not determine the core proof interfaces. Each sub-namespace is axiom-free under the strict `bedc_ci.py axiom-purity -strict` audit.

9235.3.32 Lean Module Topology

The Lean dependency topology for `BEDC.MetaCIC` is a directed acyclic graph. `BEDC.MetaCIC.Syntax` is the foundation: it imports `BEDC.FKernel.Mark` and defines Term, Ctx, shift, and substitute. On that base, `BEDC.MetaCIC.ClosedTerm` and `BEDC.MetaCIC.Beta` depend only on syntax. The substitution family `BEDC.MetaCIC.Substitution.NatShift`, `BEDC.MetaCIC.Substitution.Basic`, `BEDC.MetaCIC.Substitution.Core`, and `BEDC.MetaCIC.Substitution.Statements` depends on syntax and closed terms, while `BEDC.MetaCIC.Substitution` re-exports that surface. `BEDC.MetaCIC.Typing.Core` `BEDC.MetaCIC.Typing.SubstitutionSurface`, and `BEDC.MetaCIC.Typing.ClosedBinderDomains` depend on syntax and the substitution surface. `BEDC.MetaCIC.SubjectReduction` depends on typing and beta reduction; `BEDC.MetaCIC.Confluence` depends on beta reduction; `BEDC.MetaCIC.Normalization` depends on beta reduction and closed terms; and `BEDC.MetaCIC.AtomInfer` and `BEDC.MetaCIC.TypedExamples` depend on typing. No module in this topology imports back into one of its prerequisites.

9235.3.33 Tactic Discipline

The MetaCIC Lean modules are kept under a restricted proof discipline. Tactics such as `simp`, `decide`, and `omega` are not used in this layer, because they can introduce transitive dependencies on `propext`, decision procedures, or classical reasoning.

The permitted proof surface is the structural fragment needed by the reconstruction: `match`, `cases`, `rfl`, `exact`, `apply`, `intro`, `contradiction`, `rw`, `unfold`, `induction`, and `change`. These commands keep proofs close to the generated inductive structure of the syntax, typing, reduction, and substrate definitions.

This discipline is a necessary condition for preserving the axiom-free status of the MetaCIC reconstruction. The strict purity audit `python3 lean4/scripts/bedc_ci.py axiom-purity -strict` checks each theorem's transitive dependencies against the forbidden constants `Classical.choice`, `Quot.sound`, and `propext`; the gate passes only when the forbidden set is empty for every audited theorem.

9235.3.34 Build and Audit Process

The complete local verification gate for `BEDC.MetaCIC` is the Lean build followed by the paper and axiom audits:

```
cd lean4 && lake build
python3 tools/check-axioms.py
python3 lean4/scripts/bedc_ci.py audit
python3 lean4/scripts/bedc_ci.py axiom-purity --strict
python3 lean4/scripts/bedc_ci.py conservativity-audit
```

The first command checks that the Lean modules build with no accepted `sorry` terms and no project axioms. The axiom audit enforces the source-level ban on `axiom`. The drift audit resolves the paper's Lean markers against the actual `lean4/BEDC/` namespace. The strict axiom-purity audit checks transitive theorem dependencies, excluding `propext`, `Classical.choice`, and `Quot.sound`. The conservativity audit is the metalogical gate for the reconstruction surface. Each command must exit with status 0. The chain passes only when the paper invariant and the mechanized state agree, and that agreement is the local shipping condition for the MetaCIC reconstruction.

9235.3.35 Axiom-Free Claim

The axiom-free claim for `BEDC.MetaCIC` has two distinct parts. First, no `BEDC.MetaCIC` module declares a host axiom. This is a project-wide hard invariant: `tools/check-axioms.py` exits with failure if an axiom declaration appears in the BEDC Lean sources.

Second, the checked `BEDC.MetaCIC` theorems do not depend on Lean stdlib axioms such as `Classical.choice`, `Quot.sound`, or `propext`. This dependency claim is audited by `bedc_ci.py axiom-purity -strict`, which queries Lean through `#print axioms` and rejects those extra axiom dependencies.

Thus the commitment is stronger than the absence of local axiom declarations: the reconstructed MetaCIC layer is a pure constructive fragment inside CIC, with no reliance on Lean's additional axioms beyond CIC. The audit currently covers more than 9,000 checked theorems, and the exact count is allowed to change with the theorem set.

9235.3.36 Internal Manifest Coverage

`BEDC.Manifest` is the Lean-side registry for paper-facing theorem claims. Its core record is `BEDC.Manifest.ClaimEntry`, with fields for the paper label, fully qualified Lean name, proposition, and proof witness. The field `proof : prop` is dependent: a manifest entry elaborates only when Lean's kernel accepts the supplied proof term at the recorded proposition. Thus manifest well-formedness is not a string audit; it is kernel type checking of the `List ClaimEntry`.

The registry has 65 Lean-side entries in the unfiltered manifest-coverage report. Its breadth includes paper Lean-checked theorems from base reflection, ground compiler results, F-kernel gap transport, capstone fable-machine claims, reflection claims, derived structures for holonomy, KKT, and spectral measure packets, and the six primary MetaCIC examples in this section: `BEDC.MetaCIC.sort_in_empty_ctx`, `BEDC.MetaCIC.id_sort_well_typed`, `BEDC.MetaCIC.pi_sort_sort_well_`
`BEDC.MetaCIC.id_sort_applied`, `BEDC.MetaCIC.id_sort_beta`, and `BEDC.MetaCIC.var_zero_in_sort_ctx`.

The audit command `bedc_ci.py manifest-coverage` compares two sets: paper targets named by `\leanchecked`$\{X\}$ and Lean targets registered in `BEDC.Manifest.manifest`. It reports the marker count, manifest-entry count, coverage ratio, and a sample of paper markers that are not represented by manifest entries. The unfiltered report gives 7831 paper markers and 65 manifest entries, for `65/7831 = 0.83%`. With the literal namespace filter `-scope all`, the same command reports zero markers and zero entries, because no Lean target has prefix `all`.

The gap is therefore marker-heavy rather than manifest-heavy. The manifest contains a selected kernel-typed spine, while many paper markers remain outside the internal registry. In this section, the six primary MetaCIC markers above are manifest entries; the substitution, subject-reduction, and binder-transport markers below are Lean-checked paper markers without manifest entries. The audit makes this distinction mechanical: every paper-claimed theorem must be visible either through the ordinary `\leanchecked` marker stream or through the stronger internal manifest entry stream.

9235.3.37 Shift-Aware Metatheory Progress

The shift-aware substrate is represented by `BEDC.MetaCIC.Typing`. Its sanity layer covers the sort, Pi, identity, application, and dependent identity examples, and its atomic substitution layer is no longer a schematic placeholder: sort and variable atoms are checked directly.

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing sort var`

The captured-substitution demonstration also has a positive source typing derivation and a rejected captured target.

Lean4 ✓ `BEDC.MetaCIC.r11 substitute preserves typing`

The Pi and lambda cases expose the shift-aware shift discipline directly. Their substitution-preservation lemmas are checked under closed-domain and atomic-shape hypotheses, with weakening across the shifted binder premise.

↪ variant `BEDC.MetaCIC.substitute preserves typing pi`

↪ variant `BEDC.MetaCIC.substitute preserves typing lam`

Lean4 ✓ `BEDC.MetaCIC.hasType shift weaken sort var`

The beta case of subject reduction is therefore isolated in `BEDC.MetaCIC.SubjectReduction`. It is provable from a clean substitution-preservation lemma:

Lean4 (definition) `BEDC.MetaCIC.BetaSubstitutionPreservation`

Lean4 (definition) `BEDC.MetaCIC.AppArgTypeStable`

Lean4 (definition) `BEDC.MetaCIC.LamDomainSubjectReduction`

Lean4 (definition) `BEDC.MetaCIC.PiDomainSubjectReduction`

Lean4 ✓ `BEDC.MetaCIC.subject reduction beta`

↪ variant `BEDC.MetaCIC.subject reduction`

This puts the beta subject-reduction burden exactly on BetaSubstitutionPreservation, without adding an extra operational premise to beta reduction itself. Application congruence is checked separately at the preserved-type boundary.

↪ variant `BEDC.MetaCIC.subject reduction congApp1`

The binder-transport lemmas identify the semantic footprint of the shift-aware binder context. The shift inserted under the binder is judgmentally absorbed by closedness on closed binder domains.

9235.3.38 Substrate Stack

The MetaCIC reconstruction is a layered stack of Lean modules. Each module contributes a specific fragment of the represented type theory's substrate. The stack is finite, ordered by formal dependency, and audited by the standing paper and Lean checks.

Syntax layer. `BEDC.MetaCIC.Syntax` provides Term (an inductive with five constructors: var, app, lam, pi, sort), shift (de Bruijn lifting at cutoff with explicit amount, dispatching on Boolean Nat.ble), and substitute (capture-aware substitution at arbitrary depth, dispatching on Boolean Nat.beq and Nat.blt; the Bool-level branching avoids the propext exposure that Prop-level decidable-if would introduce for generalized substitute lemmas).

Typing layer. `BEDC.MetaCIC.Typing` provides Ctx as a list of represented types, Ctx.lookup with depth-aware shift on tail-looked-up types, and the inductive predicate

$$\text{HasType} : \text{Ctx} \to \text{Term} \to \text{Term} \to \text{Prop}.$$

Its formation rules are sortRule, varRule, piRule, lamRule, and appRule.

Reduction layer. `BEDC.MetaCIC.Beta` provides BetaStep with one base beta rule and congruence rules for application heads, application arguments, lambda bodies, lambda domains, Pi codomains, and Pi domains.

Well-formedness layer. `BEDC.MetaCIC.ContextWF` provides WellFormedCtx with empty and cons formation rules. The cons rule requires the head type to be derivable at Sort in the predecessor context. This premise is required for substitution preservation: without it, ill-typed contexts admit refutations of preservation.

Substitution-preservation layer. `BEDC.MetaCIC.Substitution` hosts the substitution-preservation target with the WellFormedCtx premise. Its proof body is the active research target.

Subject-reduction layer. `BEDC.MetaCIC.SubjectReduction` depends on substitution preservation and is the next downstream target.

Operational layer. `BEDC.MetaCIC.Examples` carries concrete HasType witnesses for representative terms: the base sort judgment, the polymorphic identity, and variable lookup. `BEDC.MetaCIC.LeanBridge` carries one worked encoding of a host CIC term into MetaCIC with its HasType derivation.

Decidable layer. `BEDC.MetaCIC.Checker` provides Term.beq, structural equality on terms, with soundness Term.beq_eq; infer, type inference returning Option Term, with soundness infer_sound; and check, decidable typing check, with soundness check_sound.

Display and elaboration layer. `BEDC.MetaCIC.PrettyPrint` provides a display function Term → String. `BEDC.MetaCIC.Elaborator` provides ElabFailure, ElabResult, normalizeOneStep for single-step beta reduction, and

$$\text{elaborate} : \text{Ctx} \to \text{Term} \to \text{ElabResult}.$$

Reflection bridge implementation status. `BEDC.MetaCIC.HostBridge`.reflectExpr covers eight host Lean Expr constructors: sort, bvar, app, lam, forallE, const, fvar returning none, and mdata as a passthrough. Two elaboration commands surface the encoding to BEDC automation: #reflect_metacic e prints the Term form, and #metacic_decide e runs infer on the reflected Term and prints the inferred MetaCIC type. The latter is the end-to-end demonstration of the host-MetaCIC pipeline.

Atomic inference and concrete typing gallery. `BEDC.MetaCIC.AtomInfer` carries a minimal type-inference procedure for closed atoms: a sort literal infers to itself, a de Bruijn variable infers to its context lookup.

Lean4 (definition) `BEDC.MetaCIC.inferAtom`

Lean4 ✓ `BEDC.MetaCIC.inferAtom sort sound`

Lean4 ✓ `BEDC.MetaCIC.inferAtom var sound`

`BEDC.MetaCIC.TypedExamples` carries a gallery of concrete typed terms over the shift-aware substrate: sort and Pi types in singleton and binary contexts, the dependent identity that exhibits shift-aware binder tracking, and a beta-reducible identity application.

Lean4 ✓ `BEDC.MetaCIC.var zero in sort ctx`

Lean4 ✓ `BEDC.MetaCIC.var one in double sort ctx`

Lean4 ✓ `BEDC.MetaCIC.pi var sort in sort ctx`

Lean4 ✓ `BEDC.MetaCIC.lam dependent identity`

Lean4 ✓ `BEDC.MetaCIC.pi sort sort in empty ctx`

Lean4 ✓ `BEDC.MetaCIC.app id sort in empty ctx`

`BEDC.MetaCIC.Normalization` records that sort and variable atoms are reduction-stable: no β-step starts from a sort or variable constructor, so closed atoms typed at sort are normal forms.

Lean4 ✓ `BEDC.MetaCIC.sort no beta`

Lean4 ✓ `BEDC.MetaCIC.closed var no beta`

Audit gating. Every module in the stack is gated by `check-axioms.py`, which rejects raw `axiom` declarations; `bedc_ci.py axiom-purity -strict`, which rejects transitive `propext`, `Classical.choice`, and `Quot.sound` dependencies; `bedc_ci.py audit`, which rejects paper-Lean marker drift; and `bedc_ci.py marker-existence-audit`, which rejects paper references to non-existent Lean names. A module that fails any of these audits is rejected at PR merge time.

Stack closure. The substrate stack is open at the substitution-preservation layer and closed elsewhere: the remaining layers carry definitions and theorems that are axiom-free under `#print axioms`. Subject reduction depends on substitution preservation; closing the substitution layer is the prerequisite for opening the subject-reduction layer. No layer is closed by importing unverified content from outside the stack.

9235.3.39 Substrate Components Table

Component	Lean Module
Term + Ctx + shift + substitute	`BEDC.MetaCIC.Syntax`
ClosedAt predicate	`BEDC.MetaCIC.ClosedTerm`
BetaStep + congruence	`BEDC.MetaCIC.Beta`
Substitute syntactic lemmas	`BEDC.MetaCIC.Substitution.{NatShift,Basic,Core,Statements}`
Typing relation	`BEDC.MetaCIC.Typing.{Core,SubstitutionSurface,ClosedBinderDomai`
Subject reduction	`BEDC.MetaCIC.SubjectReduction`
Confluence relations	`BEDC.MetaCIC.Confluence.{Core,AtomShape,AtomJoin}`
Normalization atoms	`BEDC.MetaCIC.Normalization`
Atom inference	`BEDC.MetaCIC.AtomInfer`
Typed examples gallery	`BEDC.MetaCIC.TypedExamples`

9235.3.40 Current State of the Reconstruction

This subsection records the state of the MetaCIC reconstruction as encoded in BEDC's Lean library.

Definitions in place. The represented type theory exposes Term, Ctx, HasType, BetaStep, shift, substitute, WellFormedCtx, infer, check, Term.beq, pp, normalizeOneStep, and elaborate. Each definition is axiom-free under `#print axioms`.

Theorems with full proofs. The library carries axiom-free proofs for:

- Sanity instances of HasType: sort in the empty context, polymorphic identity at the Sort level, Pi types, and variable lookup in singleton contexts.
- A worked encoding of host single-sort polymorphic identity as a MetaCIC Term with its HasType derivation (section 9235.3).
- Soundness of Term.beq as structural equality.
- Soundness of infer against HasType.
- Soundness of check against HasType.

Theorems with statements pending proof. The library carries the unrestricted target, substitute_preserves_typing, guarded by a WellFormedCtx premise. Its proof requires weakening of HasType, substitution-shift composition, and binder substitution composition. The same substrate also supports closed_term_substitute_preserves_typing as a more tractable closed-term variant. The unrestricted target depends on the WellFormedCtx requirement, binder-shift propagation, and the induction-based assembly of the available helper lemmas.

Closed-term substitute preservation. The library carries a closed-term restriction of substitute preservation:

$$\begin{aligned}\text{closed_term_substitute_preserves_typing} : \text{WellFormedCtx } (B :: \Gamma) \to \text{ClosedAt } 0\ B \to \text{ClosedAt } 0\ s \to \\ \text{HasType } (B :: \Gamma)\ t\ A \to \text{HasType } \Gamma\ s\ B \to \\ \text{HasType } \Gamma\ (\text{substitute } 0\ s\ t)\ (\text{substitute } 0\ s\ A).\end{aligned}$$

The statement is registered as ClosedTermSubstitutePreservesTypingStatement. The infrastructure—ClosedAt inductive, shift_closed, substitute_closed, substitute_shift_at_eq—is in place with axiom-free proofs.

Closed-term substitute preservation: case status. The closed-term restriction has axiom-free case lemmas covering all five inductive cases of the typing derivation. The variable case is split by depth, and the application helper carries a type-equality premise:

- substitute_sort_preserves_typing_closed_anchor (sort case);
- substitute_var_zero_preserves_typing_closed_anchor (variable, depth 0);
- substitute_var_nonzero_preserves_typing_closed_anchor (variable, depth > 0);
- substitute_pi_preserves_typing_closed_anchor (Pi);
- substitute_lam_preserves_typing_closed_anchor (lambda, proved via term induction; the corresponding HasType-induction path depends on binder-shift propagation);
- substitute_app_preserves_typing_closed_anchor_with_type_eq (application, conditional on a substitute composition equation).

The main theorem depends on two proof obligations:

1. Prove the substitute composition equation substitute 0 s (substitute 0 a cod) = substitute 0 (substitute 0 s a) (su discharging the application conditional.
2. Run the outer induction on the typing derivation, supplying each case lemma's recursive premises and assembling them into the main theorem.

The closed-term restriction avoids the binder-shift propagation obstruction by ensuring the substitution targets and source terms have no free variables that would expose it.

Substitute composition. The closure-restricted form of the substitute composition equation

$$\text{substitute } 0\ s\ (\text{substitute } 0\ a\ \text{cod}) = \text{substitute } 0\ (\text{substitute } 0\ s\ a)\ (\text{substitute } 1\ s\ \text{cod})$$

is established for five of five inductive cases of `cod`: sort, var, app, lam, and pi. The sort case is unconditional, the var case assumes ClosedAt 0 s, the app case uses two recursive premises, and the lam and pi cases are conditional on a shift-substitute commutation lemma

$$\text{shift } 0\ 1\ (\text{substitute } 0\ s\ a) = \text{substitute } 1\ (\text{shift } 0\ 1\ s)\ (\text{shift } 0\ 1\ a).$$

The commutation lemma is the required premise for full unconditional composition closure.

Host reflection bridge. `BEDC.MetaCIC.HostBridge`.reflectExpr covers eight host Lean Expr constructors: sort, bvar, app, lam, forallE, const, fvar returning none, and mdata as a passthrough. The elaboration commands #reflect_metacic e and #metacic_decide e expose the bridge: the former prints the reflected Term, and the latter runs infer on that reflected term and prints the inferred MetaCIC type.

Out of scope. The reconstruction does not currently implement strong normalization, confluence, full universe hierarchy, inductive declarations within MetaCIC, or a verified embedding into the host Lean kernel. These remain research targets as outlined in section 9235.6.

Audit guarantees. The reconstruction's status is maintained by four standing audits at PR merge time:

- `check-axioms.py` rejects any `axiom` declaration.
- `bedc_ci.py axiom-purity -strict` rejects theorems whose transitive axiom dependency includes `Classical.choice`, `Quot.sound`, or `propext`.
- `bedc_ci.py audit` rejects \`leanchecked{$X$} markers whose $X$ does not exist in `lean4/BEDC/`.
- `bedc_ci.py marker-existence-audit` reports the same information as a standing first-class audit.

The current state is coherent. Formal targets without full proof terms are reported as such, and the audit guarantees make this distinction enforceable.

9235.3.41 Evolution Trajectory

The MetaCIC surface admits several coherent directions while retaining its inside-host reconstruction role. One direction is a richer conversion relation: eta-conversion can be added to `BetaStep` and `BetaParallel`, with eta-reduction for functions represented beside beta contraction. Another direction is a universe hierarchy, replacing the single sort by indexed Type i levels and typing rules that track universe movement. A third direction is syntax for inductive families beyond Pi and Sort, so that families with constructors and eliminators become represented objects.

Deeper metatheory would strengthen the current shape-local surface. A full substitution-preservation theorem would transport arbitrary binder domains through shift and substitution, rather than relying only on closed-domain cases. Strong normalization for closed sort-typed terms would naturally follow a Tait reducibility route, with reducibility predicates indexed by represented types. A complete Church–Rosser theorem would turn the local parallel diamond components into confluence for the full represented reduction system.

A host bridge would add reflection in the other direction: selected `Lean.Expr` values could be translated into BEDC `Term` values under an explicit reconstruction certificate. On that basis, a command such as `#metacic_decide` could ask the host to quote a term, reconstruct the corresponding MetaCIC object, and discharge represented typing or conversion goals against the internal BEDC surface.

9235.3.42 Open Targets

The reconstruction does not yet close the following metatheoretic targets:

- Substitute preservation for open binder domains. The checked theorem covers closed binder-domain hypotheses; open domains still require a general binder-shift and substitution transport argument.
- Subject reduction without body-shape restrictions. The closed body cases are represented, while arbitrary lambda and Pi bodies still require binder-drop handling that commutes with typing projections.
- Confluence as a full diamond property for parallel reduction across all term shapes. The atom cases are closed; Pi, lambda, and application cases still depend on general join assemblers for their components.
- Strong normalization for closed sort-typed terms.

- Decidable type checking for arbitrary Pi, lambda, and application terms. The checked AtomInfer surface covers only sort and variable atoms.

9235.3.43 Results Overview

The formalized MetaCIC result set covers the main closed reconstruction claims needed by this chapter. Closed binder-domain substitute preservation is proved by substitute_preserves_typing_closed_binder_d giving the complete closed theorem over the shift-aware typing substrate. Subject reduction is represented by the main theorem subject_reduction together with the beta case, the application congruence cases such as subject_reduction_congApp1, and the closed body cases for lambda and Pi constructors.

The confluence layer supplies the parallel reduction framework BetaParallel, beta-star transport lemmas betaStarStep_∗, and atom-shape diamond components for variables and sorts. The normalization layer proves that atoms do not reduce: sort_no_beta, closed_var_no_beta, and closed_atom_normal_form close the represented normal-form facts for the atomic fragment.

The executable demonstration layer contains the closed-atom inference theorems inferAtom_sort_sound and inferAtom_var_sound, plus a concrete typed-term gallery with roughly fifteen examples spanning variables, Pi terms, dependent identity terms, and applications. All listed results are axiom-free under `#print axioms`; the audited dependency sets contain no `propext`, `Classical.choice`, or `Quot.sound`.

9235.3.44 Reconstruction Summary

The MetaCIC reconstruction has three foundation layers in Lean: `BEDC.MetaCIC.Syntax` defines the de Bruijn term grammar, contexts, lookup, shift, and substitution; `BEDC.MetaCIC.Beta` defines one-step beta reduction with congruence constructors; and `BEDC.MetaCIC.ClosedTerm` defines closure at a de Bruijn boundary together with the closure lemmas for shift and substitution. On this base, `BEDC.MetaCIC.Typing` exposes the shift-aware typing relation and its substitution surface.

Substitution preservation is represented inside `BEDC.MetaCIC.Typing`. The closed binder-domain form, substitute_preserves_typing_closed_binder_domains, gives the binder-domain complete closed variant: when the substituted domain, source term, and binder-domain typing surface are closed at the relevant boundary, substitution preserves the resulting typing judgment. The same module family records the closedness, prefix-removal, and HasType projection lemmas used by that theorem.

Subject reduction is represented by `BEDC.MetaCIC.SubjectReduction`. It contains the beta case, the main subject-reduction theorem parameterized by the substitution-preservation and stability premises, and the congruence cases for application heads, application arguments, lambda bodies, and Pi codomains. The confluence surface in `BEDC.MetaCIC.Confluence` supplies beta-star and parallel beta relations, the embedding of one-step beta into parallel beta, and diamond components for the variable, sort, application, lambda, and Pi cases. The normalization surface in `BEDC.MetaCIC.Normalization` proves the atomic no-step cases for sorts and closed variables, including the closed atomic normal-form theorem.

The demonstration layer consists of `BEDC.MetaCIC.AtomInfer`, which infers and proves soundness for sort and variable atoms, and `BEDC.MetaCIC.TypedExamples`, which gives concrete typing derivations for variables, Pi terms, lambda identity terms, and applications. The whole MetaCIC reconstruction is maintained as axiom-free under `#print axioms`: the audited theorem dependencies do not include `propext`, `Classical.choice`, or `Quot.sound`.

Shift-Aware Metatheory: Case-Level Closure

The shift-aware MetaCIC layer isolates subject reduction into local closure contracts for substitution, beta contraction, and congruence. The scope is deliberately case-level: each theorem states a structurally checkable condition on the represented syntax and typing derivation, and the final assembly records the remaining interface hypotheses.

Atom-shaped binder substitution. Substitution preserves typing for the Pi and lambda constructors under closedness and atom-shape hypotheses. For Pi terms, the domain and codomain premises reduce to the sort-or-variable fragment after binder-head transport.

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing pi`

For lambda terms, the same pattern applies to the body under the shifted binder context, yielding a typed substituted lambda against the substituted Pi type.

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing lam`

The shared binder transport step is represented explicitly as the equality which turns substitution under a shifted closed head into the shifted substituted head.

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing binder head shape`

The lower-level constructor assemblers expose the same decomposition without the atom-shape discharge:

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing pi if subderivations`

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing lam if subderivations`

The closed binder-domain surface removes the atom-shape restriction when every binder domain in the source derivation is closed at its local depth. In that surface, removing the substituted hypothesis preserves the represented derivation, and closedness makes substitution definitionally inert on both the term and its type. This gives a direct closed-domain substitution-preservation theorem for the represented typing judgment.

Lean4 (definition) `BEDC.MetaCIC.ClosedBinderDomainTyping`

Lean4 (definition) `BEDC.MetaCIC.ClosedBinderDomainsSubstitutionSurface`

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing closed binder domains`

Beta-substitution cases. The clean beta-substitution contract is BetaSubstitutionPreservation: a typed body under the shifted domain context remains typed after substituting a typed argument into both body and codomain. The sort and variable cases are fully closed by direct inversion on the body derivation.

Lean4 ✓ `BEDC.MetaCIC.betaSubstitutionPreservation sort var`

The depth-one form supplies the same atom-shaped argument inside an additional binder frame, which is the common recursive premise used by binder bodies.

Lean4 ✓ `BEDC.MetaCIC.betaSubstitutionPreservation depth one sort var`

Pi bodies, lambda bodies, and application bodies are then separated into constructor-specific clean lemmas. The Pi case reconstructs a typed Pi after cleaning the substituted domain and proving the codomain in the shifted context.

Lean4 ✓ `BEDC.MetaCIC.betaSubstitutionPreservation pi body`

The lambda case follows the same domain-cleaning pattern and reconstructs the lambda rule with the body under the shifted substituted domain.

Lean4 ✓ `BEDC.MetaCIC.betaSubstitutionPreservation lam body`

The application case factors through typed substituted function and argument premises, then closes the result type by a shape-bounded substitute composition equation for the application

codomain.

Lean4 ✓ BEDC.MetaCIC.betaSubstitutionPreservation app body

Parallel-reduction confluence frame. The confluence layer separates ordinary beta steps from a parallel reduction relation. One-step beta embeds into the parallel relation, while reflexivity is available for every represented term. The diamond goal is recorded as the closed proposition BetaParallelDiamond, and the constructor-level join assemblers close the variable, sort, application, lambda, and Pi component cases.

Lean4 (definition) BEDC.MetaCIC.BetaStarStep
Lean4 (definition) BEDC.MetaCIC.BetaStarConfluent
Lean4 (definition) BEDC.MetaCIC.BetaParallel
Lean4 (definition) BEDC.MetaCIC.BetaParallelDiamond
↪ variant BEDC.MetaCIC.betaParallel refl
↪ variant BEDC.MetaCIC.betaStep to parallel
Lean4 ✓ BEDC.MetaCIC.betaParallel var diamond
Lean4 ✓ BEDC.MetaCIC.betaParallel sort diamond
Lean4 ✓ BEDC.MetaCIC.betaParallel app diamond of components
Lean4 ✓ BEDC.MetaCIC.betaParallel lam diamond of components
Lean4 ✓ BEDC.MetaCIC.betaParallel pi diamond of components

Subject-reduction assembly. Subject reduction is split across beta contraction and congruence rules. The beta contraction step is the abstract interface from BetaSubstitutionPreservation to the application redex typing judgment. The congruence steps preserve typing when reduction happens in the function, argument, lambda body, or Pi codomain position:

Lean4 ✓ BEDC.MetaCIC.subject reduction congApp1
Lean4 ✓ BEDC.MetaCIC.subject reduction congApp2
Lean4 ✓ BEDC.MetaCIC.subject reduction congLam
Lean4 ✓ BEDC.MetaCIC.subject reduction congPi

The main theorem assembles these cases by induction on the represented beta step. It takes beta-substitution preservation and application-argument type stability as explicit hypotheses, then returns preservation of the same represented type across a single beta step.

Lean4 ✓ BEDC.MetaCIC.subject reduction

The closed-term boundary is also represented explicitly. A closed lambda can contain a beta redex whose local argument is the bound variable of the enclosing lambda; the whole term is closed, the inner beta contraction steps under the binder, but the contracted argument is not closed at depth zero. This records the precise local obstruction that a closed-term subject-reduction theorem must discharge: closedness of the outer term alone does not provide zero-depth closedness for every local redex argument.

Lean4 (definition) BEDC.MetaCIC.closedBinderRedexTerm
Lean4 ✓ BEDC.MetaCIC.closedBinderRedexTerm closed
Lean4 ✓ BEDC.MetaCIC.closedBinderRedexTerm steps
Lean4 ✓ BEDC.MetaCIC.closedBinderRedexTerm argument not closed
Lean4 ✓ BEDC.MetaCIC.closed term step has unclosed local argument

9235.4 Representational Adequacy for the MetaCIC Fragment

The adequacy question for `BEDC.MetaCIC` is not whether BEDC replaces the host calculus. It is whether the represented fragment has enough syntax, typing structure, and reduction structure to stand for the corresponding host CIC fragment without changing the derivational content of the examples under discussion. The relevant fragment is the closed dependent-function fragment with variables, application, lambda abstraction, Pi formation, one base sort, and β-reduction.

9235.4.1 Adequacy Criterion

Let t be a closed term of the host CIC fragment generated by one base sort, dependent products, lambda abstraction, application, and bound variables. A representational adequacy statement for `BEDC.MetaCIC` has three parts. First, there is a representation map

$$\ulcorner - \urcorner : \mathrm{HostTerm}_{0,\Pi,\lambda,\mathrm{app}} \longrightarrow \texttt{BEDC.MetaCIC.Term}.$$

Second, host typing derivations for that fragment correspond to represented typing derivations:

$$\varnothing \vdash_{\mathrm{host}} t : T \quad \Longleftrightarrow \quad \texttt{BEDC.MetaCIC.HasType}\ []\ \ulcorner t \urcorner\ \ulcorner T \urcorner.$$

Third, the correspondence is derivational rather than merely extensional: constructor uses in the host fragment are matched by constructor uses in `HasType`. Sort formation corresponds to `sortRule`, variable lookup corresponds to `varRule`, Pi formation corresponds to `piRule`, lambda introduction corresponds to `lamRule`, and application elimination corresponds to `appRule`. Thus the represented object is not only a pretty-printed term. It is a term together with a checked inhabitant of the represented typing judgment.

This criterion is intentionally fragment-relative. It does not assert adequacy for the whole host kernel, and it does not assert that the host elaborator can be recovered from the represented syntax. It says that every closed derivation in the specified fragment has a represented term and a corresponding `HasType` derivation, and that every represented derivation in the fragment reads back to a host derivation of the same dependent-function form.

9235.4.2 Mechanized Witnesses

The Lean file `BEDC.MetaCIC.TypedExamples` gives concrete witnesses for the representation criterion. These examples are not a general translation theorem. They are checked anchors showing that the rules used by familiar host CIC identity and projection terms are present in the represented calculus.

Host fragment form	MetaCIC representative	Checked witness
Identity in a sort-indexed context, $\lambda x : A.\ x$	`Term.lam (Term.var 0) (Term.var 0)` with type `Term.pi (Term.var 0) (Term.var 1)`	`carrier_identity_in_sort_ctx`
First projection from two inhabitants, $\lambda A.\lambda x : A.\lambda y : A.\ x$	Three nested lambdas returning the de Bruijn variable for the first argument	`first_of_two`
Second projection from two inhabitants, $\lambda A.\lambda x : A.\lambda y : A.\ y$	Three nested lambdas returning the de Bruijn variable for the second argument	`second_of_two`
Identity on a Pi value, $\lambda f : (\Pi x : \mathrm{Sort}.\mathrm{Sort}).\ f$	`Term.lam (Term.pi Term.sort Term.sort) (Term.var 0)`	`id_on_pi_sort_sort`
Application of a function argument, $\lambda A.\lambda x : A.\lambda f : (A \to \mathrm{Sort}).\ f\ x$	Nested lambdas ending in `Term.app (Term.var 0) (Term.var 1)`	`apply_function_arg`

Lean4 ✓ `BEDC.MetaCIC.carrier identity in sort ctx`
Lean4 ✓ `BEDC.MetaCIC.first of two`
Lean4 ✓ `BEDC.MetaCIC.second of two`
Lean4 ✓ `BEDC.MetaCIC.id on pi sort sort`
Lean4 ✓ `BEDC.MetaCIC.apply function arg`

The witnesses cover the core constructor pattern. The identity examples use lambda introduction and variable lookup under a binder. The projection examples test multiple nested binders and the index discipline needed to distinguish earlier and later arguments. The Pi-value identity shows that the fragment can quantify over represented function types, not merely over the base sort. The function-argument example combines Pi typing, binder shifting, variable lookup, and application elimination in one derivation.

The table should be read as a finite mechanized adequacy gallery. It supports the claim that the represented rules are aligned with the host fragment on these canonical terms. A full adequacy theorem would quantify over all closed terms in the fragment and prove the two readback directions by induction on host and represented derivations.

9235.4.3 Substrate Differences

The representation is not textually identical to the host surface. Three substrate differences are essential.

De Bruijn binders and named binders. The host presentation usually uses named binders, while `BEDC.MetaCIC.Term` uses de Bruijn indices. This is an encoding choice, not a change in binding structure. A closed named term has a de Bruijn translation determined by the binder stack, and alpha-equivalent named terms translate to the same de Bruijn term. Conversely, a de Bruijn term can be read back to a named term by choosing fresh binder names. The correspondence is therefore isomorphic up to alpha-renaming for the closed fragment.

Single universe and a universe hierarchy. The represented syntax has one constructor `Term.sort`. It therefore covers only the universe-zero presentation of the fragment used here. Host terms whose derivations rely on explicit universe levels, cumulative constraints, or polymorphic

universe parameters are outside this section's adequacy criterion. Within the one-sort fragment, `sortRule` supplies the represented sort judgment used by the witness derivations.

β-only reduction and $\beta + \eta$ conversion. `BEDC.MetaCIC.BetaStep` contains beta contraction and congruence rules. It does not contain eta-conversion. Adequacy for this section is therefore adequacy for a beta-conversion fragment. Host arguments that rely on eta-equality are not represented as definitional equality in this layer.

Substrate point	Host-side surface	MetaCIC surface
Binders	Named binders modulo alpha-renaming	De Bruijn indices with shift-aware substitution
Universes	Universe hierarchy and polymorphic levels	One represented `Term.sort`
Conversion	Host definitional equality may include more than beta for specific fragments	Explicit beta-step relation without eta

9235.4.4 Coverage Boundary

The adequacy claim is bounded by the represented grammar and judgment rules. The following host-kernel features are not covered by the section's criterion:

Feature	Status in `BEDC.MetaCIC`
Universe polymorphism	Not represented; there are no universe variables, levels, constraints, or cumulative universe rules.
Inductive families	Not represented; the grammar has Pi, lambda, application, variables, and sort, but no family declarations or eliminators.
Propositions versus sets	Not separated; the one-sort fragment has no distinct `Prop`/`Type` split.
Eta conversion	Not represented; beta is the explicit conversion relation.
Host elaboration	Not represented; elaborator behavior, coercions, typeclass search, and notation expansion are outside the term grammar.

These boundaries are part of the adequacy statement. The represented calculus is adequate only for the stated closed dependent-function fragment. It is not an adequate representation of Lean's complete kernel language, nor of the surface language accepted by the elaborator. Within its boundary, however, the mechanized examples demonstrate that the core identity, projection, Pi, and application patterns have BEDC.MetaCIC terms whose `HasType` derivations mirror the corresponding host CIC derivations.

9235.5 Bounds: What This Surface Does Not Claim

The self-compilation surface is bounded by explicit negative claims. These bounds are part of the theory, because without them a reflection result can be misread as a stronger foundational theorem.

First, BEDC does not claim L4a. Since the formalization sits inside the host Calculus of Inductive Constructions, an internal axiom-free theorem asserting the relevant complete truth, soundness, or consistency of that same host-scale calculus is blocked by the Godel-Tarski boundary. Tarski undefinability blocks an internal truth predicate with the required global behavior. Godel's second incompleteness theorem blocks a sufficiently strong, consistent system from proving its own consistency in the corresponding internal form.

Second, BEDC rejects L4c. The paper's governance discipline keeps theory closure and verification status as independent axes. The policy is recorded in chapter 9466. A scoped paper theorem can be paired with a Lean target, a checked Lean theorem can sit inside a deliberately narrow paper scope, and a bridge can be absent even when both coordinates are clear. This is not a defect in the surface. It is the mechanism that prevents a proof marker from becoming a claim of public mathematical closure.

Third, TM simulation does not subsume CIC reconstruction. A simulator can make behavioral statements about execution traces. Such statements have the shape of finite runs, accept states, and observed outputs. CIC reconstruction gives structural statements about syntax, contexts, typing, and reduction. These objects can occur under universal quantifiers and induction principles:

$$\forall d,\ \mathrm{TypingDerivation}(d) \longrightarrow \mathrm{Property}(d).$$

That judgmental form is not the same as predicting that a machine accepts a file. Both surfaces are useful, but they support different theorems.

Fourth, the reflection bridge between MetaCIC and the host kernel is outside the present scope. A bridge would have to relate represented judgments to actual host elaboration or kernel checking with enough precision to transfer meaningful proof obligations. That is a metatheoretic-program target. The present surface records the construction and boundary of represented syntax, represented typing, finite checked manifest entries, and external paper-Lean audits.

These bounds fix the intended reading. BEDC can internally carry finite checked theorem entries, represent parts of its own kernel vocabulary, and expose obligation packets for concrete name certificates. It cannot, from inside the same host logic and without additional axioms, certify the global truth or consistency of the host that checks it.

MetaCIC Mechanization Status Dashboard

The `BEDC.MetaCIC` formalization is a mathlib-free, zero-axiom, zero-sorry Lean 4 development over a first-principles represented calculus. The dashboard below reads that development against a standard CIC metatheory checklist. It distinguishes proved Lean targets, statement-level interfaces, finite galleries, and open textbook theorems without upgrading any open entry into a closure claim.

Metatheory property	**BEDC status**	**Lean target**	**Notes**
Object syntax grammar	encoded	`BEDC.MetaCIC.Term`	de Bruijn terms with sort, variable, application, lambda, and Pi shapes
Context lookup	encoded	`BEDC.MetaCIC.Ctx.lookup`	structural list lookup for represented contexts

Metatheory property	BEDC status	Lean target	Notes
Shift operation	encoded	`BEDC.MetaCIC.shift`	native structural recursion over represented terms
Substitution operation	encoded	`BEDC.MetaCIC.substitute`	capture-aware de Bruijn substitution with binder shift
Typing judgment	encoded	`BEDC.MetaCIC.HasType`	inductive represented typing relation
One-step beta reduction	encoded	`BEDC.MetaCIC.BetaStep`	contraction plus app, lambda, and Pi congruence shapes
Closure predicate	encoded	`BEDC.MetaCIC.ClosedAt`	boundary-indexed local closedness predicate
Closed shift stability	proved	`BEDC.MetaCIC.shift_closed`	shift is inert at a closed boundary
Closed substitution stability	proved	`BEDC.MetaCIC.substitute_closed`	substitution is inert at a closed boundary
Same-depth shift/substitute cancellation	proved	`BEDC.MetaCIC.substitute_shift_at_zero`	de Bruijn bookkeeping equality
Substitution preservation on sort/variable atoms	proved	`BEDC.MetaCIC.substitute_preserves_typing_sort_var`	atom-shape preservation surface
Substitution preservation for Pi	proved	`BEDC.MetaCIC.substitute_preserves_typing_pi`	closed typing and atom-shape hypotheses
Substitution preservation for lambda	proved	`BEDC.MetaCIC.substitute_preserves_typing_lam`	closed typing and atom-shape hypotheses
Substitution preservation for closed binder domains	proved	`BEDC.MetaCIC.substitute_preserves_typing_lam_closed_binder_d`	closed domain surface
Naive substitution composition	refuted	`BEDC.MetaCIC.substitute_substitute_zero_zero_unclosed_co`	fails without closure discipline
Closed substitution composition, sort case	proved	`BEDC.MetaCIC.substitute_substitute_zero_zero_closed_sort`	closed sort case
Closed substitution composition, variable case	proved	`BEDC.MetaCIC.substitute_substitute_zero_zero_closed_var`	closed variable case
Closed substitution composition, app case	proved	`BEDC.MetaCIC.substitute_substitute_zero_zero_closed_app`	closed app case
Closed substitution composition, lambda anchor	proved	`BEDC.MetaCIC.substitute_substitute_zero_zero_lam_closed_`	closed lambda anchor case

Metatheory property	BEDC status	Lean target	Notes
Closed substitution composition, Pi anchor	proved	`BEDC.MetaCIC.substitute_substitute_anchor_zero_zero_pi_closed_a`	substitution composition case
Beta substitution preservation interface	statement	`BEDC.MetaCIC.BetaSubstitutionPreservation`	general preservation used by subject reduction
Beta substitution preservation, sort/variable cases	proved	`BEDC.MetaCIC.betaSubstitutionPreservation_sort_var`	beta preservation over sort/variable cases
Beta substitution preservation, depth-one atoms	proved	`BEDC.MetaCIC.betaSubstitutionPreservation_depth_one_sort`	beta preservation depth-one atom case
Beta substitution preservation, Pi body	proved	`BEDC.MetaCIC.betaSubstitutionPreservation_pi_body`	beta preservation specific Pi body case lemma
Beta substitution preservation, lambda body	proved	`BEDC.MetaCIC.betaSubstitutionPreservation_lam_body`	beta preservation specific lambda body case lemma
Beta substitution preservation, app body	proved	`BEDC.MetaCIC.betaSubstitutionPreservation_app_body`	beta preservation specific application body case lemma
Closure preservation under beta	proved	`BEDC.MetaCIC.betaStep_preserves_ClosurePreservation`	beta step preserves closure preservation module
Closure preservation under beta-star	proved	`BEDC.MetaCIC.betaStarStep_preserves_closure`	beta-star preserves closure over beta-star
Beta-star relation	encoded	`BEDC.MetaCIC.BetaStarStep`	reflexive transitive closure over one-step beta
Beta-star transitivity	proved	`BEDC.MetaCIC.betaStar_trans`	composition of beta-star evidence
Beta-star app congruence	proved	`BEDC.MetaCIC.betaStarStep_app_double`	componentwise multi-step application congruence
Beta-star lambda congruence	proved	`BEDC.MetaCIC.betaStarStep_lam_double`	domain and body multi-step congruence
Beta-star Pi congruence	proved	`BEDC.MetaCIC.betaStarStep_pi_double`	domain and codomain multi-step congruence
Confluence, beta-star sort atom	proved	`BEDC.MetaCIC.betaStarStep_diamond_sort`	sort diamond component
Confluence, beta-star variable atom	proved	`BEDC.MetaCIC.betaStarStep_diamond_var`	variable diamond component
Parallel beta relation	encoded	`BEDC.MetaCIC.BetaParallel`	parallel reduction substrate for diamond assembly
Parallel beta reflexivity	proved	`BEDC.MetaCIC.betaParallel_refl`	reflexive parallel reduction
One-step beta embeds in parallel beta	proved	`BEDC.MetaCIC.betaStep_to_parallel`	ordinary beta step to parallel step

Metatheory property	BEDC status	Lean target	Notes
Parallel beta to beta-star	proved	`BEDC.MetaCIC.betaParallel_to_betaStar`	parallel step embeds in beta-star
Parallel confluence, variable atom	proved	`BEDC.MetaCIC.betaParallel_var_diamond`	variable component of parallel diamond
Parallel confluence, sort atom	proved	`BEDC.MetaCIC.betaParallel_sort_diamond`	sort component of parallel diamond
Parallel confluence, application component	proved	`BEDC.MetaCIC.betaParallel_app_diamond_of_components`	component diamonds assembler
Parallel confluence, lambda component	proved	`BEDC.MetaCIC.betaParallel_lam_diamond_of_components`	component diamonds assembler
Parallel confluence, Pi component	proved	`BEDC.MetaCIC.betaParallel_pi_diamond_of_components`	component diamonds assembler
Full beta confluence	open	*none*	extension from component diamonds to the full textbook theorem
Subject reduction, beta contraction	hypothesis-based	`BEDC.MetaCIC.subject_reduction_beta`	uses the beta-substitution preservation interface
Subject reduction, app function congruence	proved	`BEDC.MetaCIC.subject_reduction_congApp1`	preserves represented type under function-step congruence
Subject reduction, app argument congruence	proved	`BEDC.MetaCIC.subject_reduction_congApp2`	application-argument type stability
Subject reduction, lambda body congruence	proved	`BEDC.MetaCIC.subject_reduction_congLam`	body-step congruence case
Subject reduction, Pi codomain congruence	proved	`BEDC.MetaCIC.subject_reduction_congPi`	codomain-step congruence case
Subject reduction, closed terms	hypothesis-based	`BEDC.MetaCIC.subject_reduction`	requires beta substitution, app-argument stability, and binder-domain stability hypotheses
Subject reduction, retyping wrapper	hypothesis-based	`BEDC.MetaCIC.subject_reduction_retype`	packages type preservation as existence of a post-step typing witness
Lambda-domain stability interface	statement	`BEDC.MetaCIC.LamDomainSubjectReduction`	domain-stability hypothesis for subject reduction
Pi-domain stability interface	statement	`BEDC.MetaCIC.PiDomainSubjectReduction`	domain-stability hypothesis for subject reduction

Metatheory property	BEDC status	Lean target	Notes
Application argument stability interface	statement	`BEDC.MetaCIC.AppArgTypeStable`	argument-step type-stability hypothesis for subject reduction
Closed local-redex obstruction	proved	`BEDC.MetaCIC.closed_term_step_has_unclosed_local_argumen`	closed terms can contain locally open redex arguments
Strong normalization	open	*none*	no global termination theorem for beta reduction
Bounded normalizer	operational	`parts/self_compilation/08_metatheory_normalization`	fuel-bounded normalization reading, with closure preservation carried by finite beta traces
Atomic normal forms	proved	`BEDC.MetaCIC.closed_atom_normal_form`	closed atom-shaped typed terms cannot beta-step
Sort has no beta target	proved	`BEDC.MetaCIC.sort_no_beta`	normal-form component
Variable has no beta target	proved	`BEDC.MetaCIC.closed_var_no_beta`	normal-form component
Atom inference, sort	proved	`BEDC.MetaCIC.inferAtom_sort_sound`	minimal inference procedure sound on sort
Atom inference, variable	proved	`BEDC.MetaCIC.inferAtom_var_sound`	minimal inference procedure sound on context lookup
Representational adequacy	partial	`parts/self_compilation/04_metatheory_representational_ad`	finite typed representation witnesses, not a full adequacy theorem
Host-kernel reflection bridge	open	*none*	no checked bridge from represented typing to the host kernel

```
Lean4 ✓  BEDC.MetaCIC.betaStep preserves closed
Lean4 ✓  BEDC.MetaCIC.betaStarStep diamond sort
  ↪ variant  BEDC.MetaCIC.betaStarStep diamond var
  ↪ variant  BEDC.MetaCIC.subject reduction retype
Lean4 ✓  BEDC.MetaCIC.not appArgTypeStable
Lean4 ✓  BEDC.MetaCIC.not lamDomainSubjectReduction
Lean4 ✓  BEDC.MetaCIC.not piDomainSubjectReduction
```

What is not claimed

The dashboard does not claim universe polymorphism, cumulative universes, inductive families, large eliminations, proof irrelevance, eta conversion, quotients, classical choice, or a bridge from represented typing into Lean's kernel typing judgment. It also does not claim full beta confluence, strong normalization, decidable type checking for the represented calculus, or a global representational adequacy theorem. The checked entries are exactly the Lean targets named above and the already marked MetaCIC targets cited elsewhere in this chapter.

9235.6 Reflection Bridge to the Host Kernel

The structural reconstruction in section 9235.3 represents CIC syntax and typing inside BEDC as inductive predicates. A natural question is whether this represented system can be connected back to the host Lean kernel that elaborates BEDC itself. This section outlines what such a reflection bridge would entail and why the bridge sits above the present scope.

9235.6.1 Bridge Form

A bridge between '`BEDC.MetaCIC.HasType`' and the host kernel would take one of two shapes.

Embedding bridge. A function $\mathsf{embed} : \mathrm{Term} \to \mathrm{Expr}_{\mathsf{Lean}}$ takes a MetaCIC term to a Lean expression, paired with a soundness theorem

$$\mathsf{embed\text{-}sound} : \mathrm{HasType}\ \Gamma\ t\ A \to \texttt{Lean.elab}\ (\mathsf{embed}\ t)\ (\mathsf{embed}\ A).$$

The function embed would be defined by structural recursion on Term. The soundness theorem would require the host's elaboration semantics to be exposed as a first-class object inside BEDC, which it currently is not.

Correspondence bridge. An inductive relation $\mathsf{Corresponds} : \mathrm{Term} \to \mathrm{Expr}_{\mathsf{Lean}} \to \mathrm{Prop}$ relates a MetaCIC term to a Lean expression structurally, without claiming elaboration soundness. This is weaker than the embedding bridge but more tractable: only the syntactic shape needs to align, not the kernel's behavior.

9235.6.2 Formal Obstacles

Three obstacles prevent immediate implementation.

First, Lean expressions are not first-class BEDC objects. The host kernel's Expr type lives in '`Lean`' meta machinery, including the elaboration monad. Importing Expr as a BEDC inductive would require either using '`Lean.Expr`' from Lean's own module hierarchy, which would couple BEDC to non-mathlib Lean meta, or duplicating Expr as a fresh inductive inside BEDC, which pays for syntactic divergence.

Second, elaboration is not a closed-form function. '`Lean.elab`' involves unification, universe polymorphism, instance search, and reduction steps that are not expressible as a pure CIC inductive predicate. The soundness theorem above is not a target the present MetaCIC framework can state, let alone prove.

Third, soundness would require reflection inside CIC. Even if embed and Corresponds are defined, a soundness theorem connecting represented derivations to host elaboration is itself a metatheoretic statement of the kind blocked by Tarski, as described in section 9235.5.

9235.6.3 Tractable Subgoals

Several subgoals are tractable in isolation.

Subgoal A: pretty-printing. A pure function Term → String renders MetaCIC terms in a human-readable syntax. This requires no host coupling; it is an exercise in pure BEDC.

Subgoal B: a checker. A decidable function check : Ctx → Term → Term → Bool returns `true` exactly when HasType Γ t A is derivable, paired with soundness and completeness theorems against the inductive judgment. This is entirely internal to MetaCIC and a natural metatheory target.

Subgoal C: a reified elaborator. A pure BEDC function returns a result with explicit failure modes when given an ill-formed MetaCIC term, plus an evaluator returning normal forms when typing succeeds. This still does not couple to the host kernel, but it produces an artifact that resembles one.

9235.6.4 Position

The reflection bridge is research-stage and outside the present self-compilation surface. The chapter records its form so that later work can address the subgoals above without re-deriving the question. The bridge is not currently a delivery commitment of BEDC.

9235.7 Shift-Aware Substrate Design

The substrate for `BEDC.MetaCIC` is part of the formal claim, not only a notation choice. The represented calculus needs binders, substitution, beta contraction, typing under contexts, and induction over syntax while preserving the project-wide axiom-free discipline. The chosen substrate is therefore a pure de Bruijn representation with explicit shift and substitution. It is small enough to be defined from first principles, but explicit enough to expose the binder-index invariant needed by beta reduction and typing.

9235.7.1 Substrate Alternatives Surveyed

Several binder substrates are mathematically standard. They do not have the same fit for an axiom-free, self-contained Lean reconstruction.

Literal binders. A literal-binder encoding stores a variable name at each binder and each occurrence. It is close to ordinary mathematical notation: lambda terms carry names, and alpha-renaming is treated as irrelevant. The formal cost is that the syntax quotient by alpha-equivalence becomes part of the substrate. Equality of terms is no longer the constructor equality of an inductive type; it is a quotient-respecting equality. In Lean this pulls the representation toward quotient reasoning or proposition extensionality for the equality of alpha-equivalence classes. That

cost conflicts with the BEDC invariant that the MetaCIC layer remains axiom-free and uses no hidden quotient axiom.

Nominal sets. A nominal substrate treats names, freshness, and finite supports as primitive structure. It can express alpha-equivalence elegantly, but it requires a fresh-name mechanism and finite-support reasoning. Those mechanisms introduce additional infrastructure around finite sets, name permutation, and support transport before the calculus itself appears. For BEDC this is the wrong dependency direction: the calculus substrate should be a closed inductive object, not an object whose well-formedness rests on a separate freshness algebra.

HOAS. A higher-order abstract syntax substrate represents object binders by host binders. This removes many manual substitution lemmas, but it moves binder behavior into the host. BEDC.MetaCIC needs the opposite direction: object terms, object binders, and object reductions must be ordinary data and relations over which BEDC can quantify. A HOAS body has host function type, so its substitution behavior is supplied by the host binder rather than internalized as a represented operation. That makes full self-reflection of the object binder mechanics unavailable at the MetaCIC layer.

Shift-aware de Bruijn. The chosen substrate uses natural-number indices and explicit operations: `shift` raises free indices at or above a cutoff, and `substitute` replaces the variable at a given depth while shifting the substituted value under binders.

Lean4 (definition) `BEDC.MetaCIC.shift`

Lean4 (definition) `BEDC.MetaCIC.substitute`

The term grammar remains an ordinary inductive type, and all binder behavior is visible as recursive functions on that type. No quotient, freshness oracle, or host-level binder semantics is needed.

This representation is less pleasant for informal notation because every binder movement must account for indices. That inconvenience is also its advantage: the object-language bookkeeping becomes part of the checked substrate rather than an implicit convention outside the formal surface.

9235.7.2 Why De Bruijn Alone Is Insufficient

Bare de Bruijn indices solve alpha-equivalence by construction, but they do not by themselves solve binder transport. A variable index is depth-relative: the same numeral denotes a different surrounding binder after descent below a lambda or Pi. Any operation that crosses a binder must record this change in depth.

Substitution without shift loses that invariant. Consider a body under a binder and an argument inserted below another binder. If the argument is copied under the inner binder without raising its free indices, its free variable zero is captured by the inner binder. Conversely, when a substituted variable is removed, variables above the removed depth must decrease by one. The operation therefore has two distinct duties: insert the replacement at the selected depth, and maintain the de Bruijn coordinate system around the removed binder.

In `BEDC.MetaCIC.Syntax`, these duties are encoded directly. For a variable, `substitute` replaces exactly the matching depth, lowers indices above that depth, and leaves lower indices unchanged. For binders, it recurses at depth $d + 1$ and replaces the inserted value by `shift` 0 1 v. This is the explicit form of the depth-relative invariant: when the operation enters a binder, the

object-language environment has one more bound variable, so free variables of the inserted value must be raised away from capture.

The typing rules use the same invariant. The lambda and Pi body premises are checked in a context headed by `shift` 0 1 dom rather than by the unshifted domain. The domain was defined outside the binder body; when it is placed into the body context, its free indices must be read one binder deeper. Thus shift is not an auxiliary convenience. It is the operation that makes the context-extension rule match the de Bruijn meaning of binder descent.

9235.7.3 Mechanized Properties

The substrate is represented by three layers of checked facts.

Closed terms are stable at their boundary. `ClosedAt n t` says that every free variable of t is below n. Shifting at the closure boundary does not change the syntax, and substituting at the closure boundary is inert:

Lean4 ✓ `BEDC.MetaCIC.shift closed`

`BEDC.MetaCIC.substitute_closed` is marked at the reconstruction substrate site. These facts are the base sanity checks for the substrate. They say that operations aimed outside the free-variable support of a term have no effect.

Shift and substitution commute in the closed anchor cases. The one-step cancellation law `substitute_shift_at_eq` records that substituting at the same depth undoes a single shift at that depth; its primary marker is also at the reconstruction substrate site. The closed-source commuting law `shift_substitute_zero_zero_closed` records the binder-head transport shape used by typing preservation:

Lean4 ✓ `BEDC.MetaCIC.shift substitute zero zero closed`

This is the algebraic reason the substituted binder domain can be moved into a shifted context without changing its represented meaning.

Closedness is preserved by substitution and beta reduction. Substitution preserves closure when the replacement is closed at the target boundary and the source term is closed one boundary higher:

Lean4 ✓ `BEDC.MetaCIC.substitute preserves closed at`

Since beta contraction is implemented by substitution, this feeds directly into the closure preservation theorem for one beta step and then for beta-star:

↪ variant `BEDC.MetaCIC.betaStep preserves closed`

↪ variant `BEDC.MetaCIC.betaStarStep preserves closed`

Typing preservation exposes the same binder transport. The typing surface contains constructor-local substitution results for Pi, lambda, and application terms, together with the main case analysis over the represented syntax. The broad term-induction surface states that, under the closedness and shape hypotheses required by the fragment, typing is preserved after substitution:

Lean4 ✓ `BEDC.MetaCIC.substitute preserves typing via term induction`

The subject-reduction assembly then uses substitution preservation as an explicit hypothesis and proves that a represented beta step preserves the same represented type. The main subject-reduction theorem is marked at the case-level metatheory site in this chapter.

9235.7.4 Open Algebraic Gap

The substrate deliberately does not assume that every familiar textbook substitution law transfers in its most direct form. A naive composition law for zero-depth substitution is false for the represented operations.

Take $d = 0$, $v = \text{sort}$, $u = \text{sort}$, and $t = \text{var } 1$ in the naive law

$$\text{substitute } d\ v\,(\text{substitute } d\ u\ t) = \text{substitute } d\,(\text{substitute } d\ v\ u)\ t.$$

The left side first lowers var 1 to var 0 and then replaces it by sort. The right side simplifies substitute 0 sort sort to sort and then substitutes at depth zero in var 1, which lowers to var 0. Thus the left side is sort and the right side is var 0.

The checked counterexample in the Lean substrate records the same obstruction using an unclosed zero-depth anchor. With

$$\begin{aligned} s &= \text{var } 0, \\ a &= \text{sort}, \\ t &= \text{var } 1, \end{aligned}$$

the checked counterexample is

$$\text{substitute } 0\ s\,(\text{substitute } 0\ a\ t) \neq \text{substitute } 0\,(\text{substitute } 0\ s\ a)\,(\text{substitute } 1\ s\ t).$$

Lean4 ✓ `BEDC.MetaCIC.substitute substitute zero zero unclosed counterexample`

Here the left side is var 0, while the right side is sort. The point is not that substitution composition is unavailable in all forms. The point is that the exact law must account for closure, shifting, and binder depth. BEDC therefore derives the composition shapes it uses, instead of importing a nominal-looking lemma whose hypotheses do not match this substrate.

9235.7.5 Comparison to the Host CIC

Lean 4's kernel and elaborator expose a richer implementation environment than the MetaCIC fragment. The host surface accepts named binders in user syntax, elaborates them through local constants and internal binder representations, and manages levels, metavariables, definitional unfolding, and kernel conversion. BEDC.MetaCIC does not mirror that implementation.

The MetaCIC substrate is intentionally smaller. It has one sort constructor, variables, application, lambda, Pi, beta contraction, and a shift-aware typing relation over explicit contexts. Names are absent from the object syntax, and there is no attempt to reproduce Lean's elaboration pipeline or the exact locally nameless machinery used by host internals.

This separation keeps the claim precise. The host kernel checks the Lean files that define the MetaCIC syntax and prove its lemmas. The object calculus inside `BEDC.MetaCIC` is a represented fragment with its own explicit binder mechanics. Its substrate is not a reimplementation of the host kernel; it is a small inductive calculus whose shift and substitution rules can be inspected, proved about, and audited inside the project.

9235.8 MetaCIC Confluence Status

The confluence surface for `BEDC.MetaCIC` is intentionally split into three layers. The first layer is the inductive reduction interface: one-step beta reduction, its reflexive-transitive closure, and

parallel beta reduction. The second layer contains shape and join lemmas for terms whose source shape prevents a real beta fork. The third layer records the component-wise diamond principles available for application, lambda, and Pi terms, together with the case distinctions that remain outside the checked surface.

This section records exactly what is represented in Lean and what is still a mathematical obligation for a full Church-Rosser theorem over the represented beta system. The statement is deliberately local to the object calculus in `BEDC.MetaCIC`: it does not appeal to Lean kernel conversion and does not import a confluence theorem from the host calculus.

9235.8.1 Setting

The primitive one-step relation is `BetaStep`, defined in `BEDC.MetaCIC.Beta`. Its constructors are the represented beta contraction and the congruence rules under application, lambda body, lambda domain, Pi codomain, and Pi domain:

$$(\lambda d.\, b)\ a \longrightarrow_\beta b[0 := a].$$

`BEDC.MetaCIC.BetaStep` is the checked name for this interface.

The reflexive-transitive closure is `BetaStarStep`. It has only reflexivity and left-consing by one `BetaStep`. The confluence predicate `BetaStarConfluent` is the ordinary joinability condition for two `BetaStarStep` paths with the same source:

$$\text{BetaStarConfluent} \quad \equiv \quad \forall t, u_1, u_2.\ t \longrightarrow_\beta^* u_1 \to t \longrightarrow_\beta^* u_2 \to \exists v.\ u_1 \longrightarrow_\beta^* v \wedge u_2 \longrightarrow_\beta^* v.$$

`BEDC.MetaCIC.BetaStarStep` and `BEDC.MetaCIC.BetaStarConfluent` are the checked names for these two interfaces.

The parallel relation `BetaParallel` is the standard proof vehicle for diamond arguments. It reduces independent subterms in one represented parallel step and also contracts an application whose function side is already a lambda after a parallel step. The associated predicate `BetaParallelDiamond` asks for a one-step parallel join:

$$\text{BetaParallelDiamond} \quad \equiv \quad \forall t, u_1, u_2.\ t \Rightarrow u_1 \to t \Rightarrow u_2 \to \exists v.\ u_1 \Rightarrow v \wedge u_2 \Rightarrow v.$$

`BEDC.MetaCIC.BetaParallel` and `BEDC.MetaCIC.BetaParallelDiamond` are the checked names for these two interfaces.

The bridge between the two relations is already mechanized in the direction needed for a Tait-Martin-Lof style proof: one-step beta embeds into parallel beta, and a parallel step is simulated by beta-star. `BEDC.MetaCIC.betaStep_to_parallel` and `BEDC.MetaCIC.betaParallel_to_betaStar` are the checked bridge lemmas. Thus, a checked full parallel diamond would be enough substrate for a beta-star Church-Rosser theorem, provided the standard multi-step lifting is also assembled inside the same object calculus.

9235.8.2 Atom-Shape Diamond

The atom fragment consists of `Term.sort` and variables. These terms have no beta redex at the root and no subterm position in which a congruence constructor can act. For one-step beta, this is recorded by impossibility lemmas:

```
↪ variant  BEDC.MetaCIC.betaStep sort absurd
↪ variant  BEDC.MetaCIC.betaStep var absurd
```

Consequently, beta-star paths out of atoms are forced to be reflexive. The target uniqueness lemmas state that a beta-star target of a sort is the same sort, and a beta-star target of a variable is the same variable:

```
↪ variant BEDC.MetaCIC.betaStarStep sort unique target
↪ variant BEDC.MetaCIC.betaStarStep var unique target
```

The join lemmas are therefore immediate. Two beta-star paths out of `Term.sort` join at `Term.sort`; two beta-star paths out of `Term.var i` join at `Term.var i`. `BEDC.MetaCIC.betaStarStep_diamond_s` and `BEDC.MetaCIC.betaStarStep_diamond_var` are the checked join theorems.

The same fact is also available on the parallel side. Parallel reduction from an atom has a unique target, so the two branches of a parallel fork have the source atom itself as their common reduct: `BEDC.MetaCIC.betaParallel_sort_diamond` and `BEDC.MetaCIC.betaParallel_var_diamond` are the corresponding parallel join theorems.

This is the precise sense in which atom confluence is closed in `BEDC.MetaCIC`. There is no redex that can make the two branches choose different residual structure. The proof is shape analysis over the constructors of the represented reduction relation, not an appeal to normalization or typing.

9235.8.3 Application, Lambda, and Pi Shapes

For compound terms, BEDC has mechanized important shape and component lemmas, but not the full diamond theorem over all application, lambda, and Pi forks. The checked component lemmas say that, if corresponding subcomponents already have parallel joins, then the compound terms have a parallel join: `BEDC.MetaCIC.betaParallel_app_diamond_of_components`, `BEDC.MetaCIC.betaParallel_lam_diamond_of_components`, and `BEDC.MetaCIC.betaParallel_pi_diamond_of`

For lambda and Pi sources, the checked shape diamond lemmas reduce the compound fork to diamond obligations for the domain and body or codomain:

```
Lean4 ✓ BEDC.MetaCIC.betaParallel lam shape diamond
Lean4 ✓ BEDC.MetaCIC.betaParallel pi shape diamond
```

These lemmas do not prove the recursive premises. They are local assembly principles: once the domain and body joins are supplied, the outer constructor is joined by congruence.

Applications are where the complete proof becomes more delicate. The checked surface contains a full shape inversion for parallel reduction from an application:

```
Lean4 ✓ BEDC.MetaCIC.betaParallel app full inv
```

It separates two target forms. Either the target is another application whose function and argument are parallel reducts of the source components, or the target is a beta contractum obtained after the function side reduces in parallel to a lambda.

The full application diamond must account for four fork shapes.

Congruence against congruence. Both branches keep the application shape:

$$f\ a \Rightarrow f_1\ a_1,$$
$$f\ a \Rightarrow f_2\ a_2.$$

This case is exactly the component-wise argument: join f_1 with f_2, join a_1 with a_2, and rebuild the application.

Congruence against beta. One branch keeps application shape while the other branch contracts after the function side exposes a lambda:

$$f\ a \Rightarrow f_1\ a_1,$$
$$f\ a \Rightarrow b_2[0 := a_2].$$

The missing obligation is a residual statement: the application branch must be shown to reduce toward a shape compatible with the contractum branch, and the contracted body must commute with the residual reductions of the argument.

Beta against congruence. This is the symmetric case. The source branch that contracts first produces a substitution instance, while the other branch keeps the application wrapper. The join must show that the wrapper branch can expose a compatible lambda and then contract to the same residual substitution target.

Beta against beta. Both branches expose lambda structure and contract:

$$f\ a \Rightarrow b_1[0 := a_1],$$
$$f\ a \Rightarrow b_2[0 := a_2].$$

The join depends on a simultaneous residual argument for the two lambda bodies and the two arguments, plus the substitution-commutation law that connects parallel residuals before and after substituting the argument into the body.

Lambda and Pi terms avoid root beta contraction, but their complete diamond proof still requires recursive joins for all binder subcomponents. For lambda, the cases are:

Domain against domain. Both branches reduce the domain and keep the body branch compatible. The join is rebuilt by the lambda congruence constructor.

Body against body. Both branches reduce the body. The join is rebuilt under the binder after the body residuals are joined.

Cross branch. One branch reduces the domain and the other reduces the body. The diamond must commute these independent moves and rebuild the lambda over the joined domain and body. The checked `betaParallel_lam_shape_diamond` lemma packages this once the recursive component joins are supplied.

Pi terms have the same pattern with domain and codomain in place of lambda domain and body. The checked `betaParallel_pi_shape_diamond` lemma again performs the outer constructor assembly, but it does not supply the recursive induction principle for the subcomponents.

9235.8.4 Strategy Notes

The standard route to Church-Rosser for beta reduction is the Tait-Martin-Lof parallel-reduction method. One proves a diamond property for parallel reduction, shows that ordinary beta steps embed into parallel reduction, shows that parallel steps are included in beta-star, and then lifts the parallel diamond to confluence of beta-star.

BEDC has the first-order interface and many constructor-local lemmas needed by that route. What is absent is the complete induction that covers every constructor of `BetaParallel` with the

correct residual term in the application beta cases. In informal notation the residual target is natural: parallel-reduce the body and argument, then substitute the residual argument into the residual body. In the BEDC substrate, that sentence expands into shift-aware de Bruijn algebra.

The hard point is the congruence-beta family of application cases. A branch may first reduce the function position to a lambda while the other branch contracts after a different exposure of lambda structure. Joining these branches requires a theorem saying that substitution commutes with the parallel residuals in exactly the represented form:

$$b \Rightarrow b' \wedge a \Rightarrow a' \quad \Longrightarrow \quad b[0 := a] \Rightarrow^* b'[0 := a'].$$

The displayed formula is schematic. In the actual substrate, descent under a binder changes the cutoff and shifts the inserted term. The open algebraic gap therefore is not merely the absence of a textbook substitution lemma; it is the absence of the exact shift-aware commutation theorem that matches `BEDC.MetaCIC.substitute`. subsection 9235.7.4 shows why a naive zero-depth composition law is false for unclosed anchors, and section 9235.7 explains the binder transport invariant that any usable law must respect.

This is also why the atom fragment is not representative of the general case. Atoms need no residual calculus, no binder descent, and no substitution transport. Application beta cases need all three.

9235.8.5 BEDC Scope

Within the closed-atom fragment, BEDC already has the confluence behavior it uses. Sorts and closed variables are normal forms for represented beta reduction, and forks out of those sources are forced to join at the source itself. This covers the atomic normal-form statements used in the self-compilation discussion: `BEDC.MetaCIC.closed_atom_normal_form`.

The scope changes once reducible closed terms are admitted. A closed term can contain an application redex, and then a fork may choose between reducing inside subcomponents and contracting the exposed lambda. BEDC has checked closure preservation for beta steps and beta-star, and it has checked subject-reduction under the stated substitution-preservation interface, but those facts are not a full diamond theorem: `BEDC.MetaCIC.betaStep_preserves_closed`, `BEDC.MetaCIC.betaStarStep_preserves_closed`, and `BEDC.MetaCIC.subject_reduction`.

The resulting boundary is sharp. Atom sources are closed under the diamond argument because their reduction relation is trivial. Reducible closed terms remain within the represented closed world under reduction, but full joinability for all their beta forks requires the application residual proof and the shift-aware substitution commutation lemmas described above.

9235.9 Normalization Status for the MetaCIC Fragment

The normalization surface of `BEDC.MetaCIC` has two distinct layers. The first layer is operational: reduction is represented by a first-principles syntax, a beta-step relation, and finite reduction chains. This layer supports bounded computation and checked closure preservation. The second layer is the textbook strong-normalization theorem for the represented dependent-function calculus. BEDC does not prove that theorem. The difference matters because the bounded layer is an effective, fuel-indexed operational account, while strong normalization is a global theorem over all well-typed reduction sequences.

9235.9.1 Bounded Normalizer

A bounded normalizer for the represented calculus has the shape

$$\texttt{BEDC.MetaCIC.normalizeBounded} : \mathrm{Nat} \to \texttt{BEDC.MetaCIC.Term} \to \texttt{BEDC.MetaCIC.Term}.$$

Its intended reading is fuel driven. At fuel 0 it returns the input term. At fuel $k+1$ it attempts one head reduction step and then continues with fuel k on the reduct; if no head step is available, it returns the current term. Termination of this procedure is inherited from structural recursion on the natural-number fuel, not from any theorem that the represented calculus is strongly normalizing.

This bounded interface is therefore not a strong-normalization certificate. It says that every chosen run of the evaluator has a finite host computation because the host recursion argument decreases. It does not say that every well-typed represented term admits no infinite beta-reduction path, and it does not identify the returned term as a unique normal form for arbitrary fuel. A run may stop because a normal form has been reached, or because the fuel budget has been exhausted.

The available checked metatheory around this operational layer is closure preservation. If a fuel-driven run is decomposed into zero or more `BetaStep` transitions, each transition preserves the closedness predicate, and induction on the finite step trace preserves closedness at the end of the run. The proof is not a termination argument. It is an invariant argument over the finite trace that the bounded computation actually takes.

9235.9.2 Closure Preservation Under Reduction

The mechanized closure theorem for one-step beta reduction is

$$\texttt{betaStep_preserves_closed} : \texttt{ClosedAt}\ n\ t \to \texttt{BetaStep}\ t\ t' \to \texttt{ClosedAt}\ n\ t'.$$

The proof lives in `lean4/BEDC/MetaCIC/ClosurePreservation.lean`. Its beta-contraction case uses the closure-preserving substitution theorem. Its congruence cases peel the corresponding `ClosedAt` constructor and apply the induction hypothesis to the reduced component. Thus closedness is not treated as a syntactic side condition on whole terms only; it is propagated through application, lambda, and Pi constructors.

The finite-chain theorem is

$$\texttt{betaStarStep_preserves_closed} : \texttt{ClosedAt}\ n\ t \to \texttt{BetaStarStep}\ t\ t' \to \texttt{ClosedAt}\ n\ t'.$$

It is induction on the represented reflexive-transitive closure. The reflexive case returns the original closedness proof. The step case applies `betaStep_preserves_closed` to the first transition and then invokes the induction hypothesis on the remaining chain.

```
↪ variant  BEDC.MetaCIC.betaStep preserves closed
↪ variant  BEDC.MetaCIC.betaStarStep preserves closed
```

These two checked targets are the normalizer-relevant safety results available inside BEDC. They justify the following operational statement: for any finite reduction trace produced by a bounded evaluator, closed input remains closed at every subsequent represented term. They do not assert progress, uniqueness of normal forms, confluence for all represented terms, or strong normalization.

Theorem 9235.1 (Bounded normalization trace closure)**.** *Let t be a represented MetaCIC term closed at index n. If a bounded evaluator run displays a finite beta trace*

$$t \to^* t'$$

from t to t', interpreted as a `BEDC.MetaCIC.BetaStarStep` *witness, then t' is closed at the same index n. The statement does not assert progress, fuel sufficiency, unique normal form, confluence, or strong normalization.*

Proof. Read the evaluator output only through the displayed finite `BetaStarStep` trace. The initial hypothesis supplies `ClosedAt` n t, and the trace supplies the finite chain from t to t'.

Apply `betaStarStep_preserves_closed`. Its reflexive case returns the original closedness proof, and its step case uses `betaStep_preserves_closed` before applying the induction hypothesis to the remaining finite trace.

No claim about the evaluator's fuel budget or global reduction behavior enters the argument. The theorem records only preservation of closedness along the finite trace that is actually displayed. □

9235.9.3 Strong Normalization

Strong normalization for the represented calculus would state that every well-typed term in the intended admissible fragment has no infinite `BetaStep` chain. The standard proof route for a dependent-function calculus is a logical-relations or reducibility-candidates argument. Such a proof requires a predicate of admissible represented types and a semantic interpretation that is stable under the type formers of the calculus.

BEDC does not currently carry that infrastructure. In particular, the current substrate lacks an inductive predicate of admissible types strong enough to support reducibility candidates. It also lacks preservation hypotheses in the form needed under Pi, lambda, and application constructors. The available subject-reduction surface is intentionally local: it records beta and congruence preservation facts, plus hypothesis-based interfaces for domain and argument stability. Those interfaces are useful for finite local metatheory, but they are not a logical-relations model.

A strong-normalization proof would therefore need at least the following ingredients:

- an inductive definition of the admissible type fragment;
- a reducibility interpretation for sort, Pi, lambda, and application shapes;
- closure of the interpretation under beta expansion and beta contraction;
- preservation lemmas whose hypotheses are stable under the relevant binder constructors;
- an induction principle linking represented typing derivations to the reducibility interpretation.

None of these is a consequence of having a fuel-indexed evaluator. Fuel gives an executable finite approximation. Strong normalization is a proof that no unbounded reduction path exists for well-typed represented terms in the chosen fragment.

9235.9.4 Practical Impact

The practical BEDC examples sit far below the strong-normalization boundary. The closed atom results prove that sort and closed variable atoms have no beta successor, and the typed closed-atom theorem packages this as a normal-form fact for the atomic cases.

↪variant BEDC.MetaCIC.closed atom normal form

The `TypedExamples` gallery is also operationally tame. Its witnesses are built from sort, Pi, lambda, variable, and application constructors in small closed shapes. The identity and projection examples have lambda or Pi heads until they are applied; the sort-headed and variable-headed components are already beta-inert; and the application examples expose the ordinary single beta contraction at the represented head when the function is a lambda. For these examples, a bounded evaluator gives an effective semantics: with enough fuel, the finite computation reaches the expected beta-normal representative and stops because there is no next reduction step along that chosen strategy.

This is the operational content needed by the paper's examples. The examples do not require a theorem that all well-typed represented terms strongly normalize. They require a checked syntax, checked typing witnesses, checked beta-step rules, and checked preservation of closedness along the finite reduction traces under discussion.

9235.9.5 Comparison with the Host Calculus

Lean 4's host kernel is trusted as the checker for this development. Its metatheory includes the normalization and consistency facts needed for the host's own dependent type theory at the meta level of the project. BEDC does not import those facts as axioms about `BEDC.MetaCIC`. The represented calculus is a first-principles object language built with inductive syntax, inductive typing, and inductive reduction relations. Its metatheory must be proved over those represented objects.

This is the same boundary recorded in section 9235.4. Representational adequacy for the finite gallery says that selected host-fragment terms have corresponding represented terms and typing derivations. It does not transfer the host kernel's global strong-normalization theorem into a checked theorem about `BEDC.MetaCIC`. Such a transfer would require a formal bridge whose statement and proof are not present here.

The normalization status is consequently precise. BEDC has a finite operational reduction layer and mechanized closure preservation for one-step and finite-chain beta reduction. It has atomic normal-form facts and concrete closed examples whose reductions are finite and explicit. It does not claim a strong-normalization theorem for the represented calculus.

9235.9.6 Status Summary

Topic	BEDC status	Exact reading
Fuel-bounded normalization	operational	finite host computation controlled by a natural-number fuel argument
Closure under one beta step	checked	closedness is preserved by each represented `BetaStep` transition
Closure under finite beta chains	checked	closedness is preserved by induction over `BetaStarStep` traces
Atomic normal forms	checked	sort and closed variable atoms have no beta successor
Typed examples	finite gallery	closed witnesses reduce only through explicit, finite beta traces
Strong normalization	open	no theorem rules out all infinite beta paths for all admissible typed terms
Host-kernel transfer	open	host normalization is not imported as a theorem about represented MetaCIC

The table is deliberately two-axis. A checked closure theorem is a Lean-side verification fact about a represented relation. An open strong-normalization entry is a theory-side boundary about the represented calculus as a whole. The fuel-bounded operational layer is useful precisely because it gives finite computation without pretending to close the global theorem.

9235.10 MetaCIC Metatheory Roadmap

This section gives a consolidated reading map for the MetaCIC metatheory surface. The substrate is `BEDC.MetaCIC`: a first-principles represented dependent-function fragment used by the self-compilation chapter as its formal metatheory base. The Lean development for this surface is mathlib-free, carries no project axioms, and contains no `sorry` placeholders. The paper side reports only the properties that are represented or proved for this object calculus; it does not import host-kernel metatheory as a theorem about the represented terms.

9235.10.1 Setting

`BEDC.MetaCIC` contains the syntax, typing judgment, substitution operation, beta relation, beta-star closure, parallel beta relation, closure predicate, and finite galleries used by the self-compilation discussion. The objects under discussion are represented terms, not Lean elaborator syntax. Their binders are de Bruijn indices, and binder transport is handled by the explicit `shift` and `substitute` functions.

The surrounding siblings have different roles. The representational-adequacy section explains how selected host-fragment terms are mirrored by checked MetaCIC witnesses. The mechanization dashboard enumerates proved targets, statement-level interfaces, and open textbook theorems. The substrate-design section explains why the de Bruijn shift discipline is part of the claim. The confluence and normalization sections isolate the checked fragments and the remaining global metatheory obligations.

9235.10.2 Mechanized Surface

The following items are available as checked Lean targets or checked finite galleries for the represented calculus.

- Closed atom diamond behavior for beta-star and parallel beta. Atom sources have forced targets, so forks out of `Term.sort` and `Term.var i` join at the source shape.

  ↪ variant `BEDC.MetaCIC.betaStarStep diamond sort`

  ↪ variant `BEDC.MetaCIC.betaStarStep diamond var`

  ↪ variant `BEDC.MetaCIC.betaParallel sort diamond`

  ↪ variant `BEDC.MetaCIC.betaParallel var diamond`

- Closure preservation under one-step beta and beta-star. If a term is closed at a boundary, each represented beta step and each finite beta-star trace keeps it closed at the same boundary.

  ↪ variant `BEDC.MetaCIC.betaStep preserves closed`

  ↪ variant `BEDC.MetaCIC.betaStarStep preserves closed`

- Bounded normalizer safety is a finite-trace safety statement: any concrete beta trace taken by a fuel-indexed run inherits the preceding closure-preservation theorem. It is an operational invariant, not a strong-normalization theorem.

- Representational adequacy is witnessed by checked typed examples for identity, projections, Pi-shaped identity, and function application in the represented dependent-function fragment.

  ↪ variant `BEDC.MetaCIC.carrier identity in sort ctx`

  ↪ variant `BEDC.MetaCIC.first of two`

  ↪ variant `BEDC.MetaCIC.second of two`

  ↪ variant `BEDC.MetaCIC.id on pi sort sort`

  ↪ variant `BEDC.MetaCIC.apply function arg`

- The `BHist` substrate embedding has checked closedness and injectivity witnesses. The map packages a history object as a MetaCIC term while preserving enough structure to recover the original history from equality of encoded terms.

  ↪ variant `BEDC.MetaCIC.BHistSubstrate.bhistToTerm closed`

  ↪ variant `BEDC.MetaCIC.BHistSubstrate.bhistToTerm injective`

9235.10.3 Open Statements

The following entries are mathematical targets for the represented calculus, but they are not checked theorems in the MetaCIC surface.

Statement	Exact boundary
Full `BetaParallel` diamond	Atom cases, component assemblers, lambda and Pi shape assemblers, and application inversion are checked; the complete app, lambda, and Pi induction over all fork shapes is not packaged as a theorem.
Strong normalization	The bounded evaluator is fuel-indexed and total as a host computation, but no theorem rules out all infinite represented beta paths for every admissible well-typed term.
Subject reduction without hypotheses	The assembled theorem is hypothesis-based. Its beta, application-argument, lambda-domain, and Pi-domain stability assumptions isolate the substitution and domain transport facts needed by the full statement.
Naive substitution composition	The direct zero-depth law is false for unclosed anchors. The substrate needs closure- and shift-aware composition statements rather than a nominal-looking formula.

9235.10.4 Why These Gaps Are Specific

The missing confluence cases are concentrated in the interaction between application beta and congruence. A branch may keep an application wrapper while another branch exposes a lambda and contracts. Joining those residuals requires a statement saying that represented substitution commutes with parallel residuals after binder descent. In the chosen substrate, binder descent changes the cutoff and shifts the inserted term, so the needed theorem is not the ordinary informal substitution slogan.

The same shift-aware issue explains the subject-reduction boundary. The three stable interfaces used by the subject-reduction theorem are substitution-level and domain-transport properties. They are stated explicitly because the project has not collapsed them into a global substitution lemma for the whole represented calculus.

Strong normalization has a different shape. Its standard proof route uses logical relations or reducibility candidates. That requires an inductive predicate of admissible represented types, a semantic interpretation of those types, and closure of the interpretation under the represented constructors. The MetaCIC layer has syntax, typing, reduction, and finite examples; it does not yet carry that reducibility framework.

The false substitution-composition law in the substrate-design section is the sanity check for this roadmap. It shows that the open algebra is not a missing line of routine rewriting. The correct statement must mention the closure and shift hypotheses that make the de Bruijn coordinates stable.

9235.10.5 Reading Guide

Reader question	Next section	What it supplies
What is represented?	section 9235.4	Finite typed witnesses for the host-fragment identity, projection, Pi, and application patterns.
What is checked?	section 9235.5	A dashboard separating encoded definitions, checked theorems, statement-level interfaces, and open textbook targets.
Why this binder substrate?	section 9235.7	The de Bruijn shift discipline, closure lemmas, and the counterexample to a naive substitution-composition law.
How far does confluence go?	section 9235.8	Atom diamonds, component and shape assemblers, application inversion, and the residual cases needed for full parallel confluence.
How far does normalization go?	section 9235.9	Fuel-bounded computation, closure preservation along finite traces, atomic normal forms, and the boundary of strong normalization.

The dependency order is therefore:

- adequacy witnesses depend on the represented syntax and typing rules;
- closure preservation depends on the shift-aware substitution substrate;
- bounded normalizer safety depends on closure preservation for finite beta traces;
- full subject reduction depends on the stable substitution and domain transport interfaces;
- full confluence depends on residual substitution commutation for application beta forks;
- strong normalization depends on a reducibility framework for the admissible represented type fragment.

9235.11 BHist Substrate Bridge

The binary-history substrate is the smallest data layer that BEDC carries through its finite certificates. In the kernel layer it is not a derived numeric notation, a string encoding, or a host-side serialization format. It is the freely generated history type with one empty history and two visible extension constructors.

9235.11.1 What BHist Is

The core history type is defined in `lean4/BEDC/FKernel/Hist.lean`. Its constructor shape is

$$\mathsf{BHist} ::= \mathsf{Empty} \mid \mathsf{e0}(\mathsf{BHist}) \mid \mathsf{e1}(\mathsf{BHist}).$$

The manuscript writes the visible constructors as $b0$ and $b1$ when the emphasis is on binary digits, and as $\mathsf{E0}(-)$ and $\mathsf{E1}(-)$ when the emphasis is on generated history structure. The Lean constructors are `BHist.Empty`, `BHist.e0`, and `BHist.e1`. The important invariant is the same in every notation: a history is a finite binary digit sequence, built by closing the empty sequence under the two single-bit extensions.

This makes BHist a primitive carrier for BEDC's internal evidence. The relation hsame on histories is definitional equality in the kernel layer, and the visible no-confusion theorems for empty, $b0$, and $b1$ are proved by induction over that generated type. Thus the object used by package rows, continuations, signatures, and name certificates is not an external byte array. It is an inductive value whose constructors are visible to the proof language.

9235.11.2 Why Bridge BHist to Term

The MetaCIC reconstruction has a separate object language, `BEDC.MetaCIC.Term`. That language represents the syntax of a small dependent-function calculus: variables, applications, lambdas, Pi types, and sort. Self-compilation needs a way for the raw BEDC carrier data to appear inside that object language. Otherwise the substrate and the represented calculus would remain adjacent artefacts rather than a single typed internal surface.

The bridge supplied by `BEDC.MetaCIC.BHistSubstrate` serves exactly this role. It embeds host-side histories into represented terms. The input is a Lean value of the BEDC history type. The output is a MetaCIC term that can be placed under the ordinary syntactic and metatheoretic predicates of the represented calculus.

The point is not to claim that every MetaCIC term denotes a history. The point is more precise: every BEDC history has a canonical represented value. This turns the finite binary-history carrier into a Church-style data structure inside the MetaCIC substrate, so that raw BEDC rows can be mentioned by the same object calculus that carries the self-compilation surface.

9235.11.3 The Mechanized Embedding

The embedding is the function

$$\texttt{BEDC.MetaCIC.BHistSubstrate.bhistToTerm} : \mathsf{BHist} \to \texttt{BEDC.MetaCIC.Term}.$$

Lean4 (definition) `BEDC.MetaCIC.BHistSubstrate.bhistToTerm`

In Lean the local name of the source type is `BHistIdx`, an abbreviation for `BEDC.FKernel.Hist.BHist`. The abbreviation records that the substrate module treats histories as the index values being encoded into MetaCIC terms; it does not create a separate history type.

The encoding has the shape of a three-argument Church case analysis. The outer term binds three represented arguments. The payload uses variable 2 for the empty-history case, variable 1 for the $b0$ branch, and variable 0 for the $b1$ branch. A visible bit extension is represented by applying the corresponding branch variable to the recursively encoded tail, protected under the same three-binder interface.

For the empty history the payload is the represented variable for the empty case:

$$\mathsf{payload}(\mathsf{Empty}) = \mathrm{var}\ 2.$$

For a $b0$ extension the payload applies the $b0$ branch to the recursively encoded tail:

$$\mathsf{payload}(\mathsf{e0}(h)) = \text{app (var 1) } (\lambda_0.\lambda_1.\lambda_2.\ \mathsf{payload}(h)).$$

For a $b1$ extension the payload is analogous, using the other branch variable:

$$\mathsf{payload}(\mathsf{e1}(h)) = \text{app (var 0) } (\lambda_0.\lambda_1.\lambda_2.\ \mathsf{payload}(h)).$$

The full embedding wraps this payload in the three lambdas that bind the empty, $b0$, and $b1$ eliminator branches. In de Bruijn notation the branch variables are local to that wrapper, so the represented value is closed at depth 0.

The mechanized closure theorem is

$$\texttt{bhistToTerm_closed}\ (h : \mathsf{BHist}) : \texttt{ClosedAt}\ 0\ (\texttt{bhistToTerm}\ h).$$

Lean4 ✓ `BEDC.MetaCIC.BHistSubstrate.bhistToTerm closed`

This is the primary checked fact for the bridge. It is the property needed by the self-compilation surface: a raw finite history becomes a closed represented term, hence it can enter the closed-term side of the MetaCIC metatheory without free-variable debt.

The same module also proves injectivity of the encoding:

$$\texttt{bhistToTerm_injective} : \texttt{bhistToTerm}\ h = \texttt{bhistToTerm}\ k \to h = k.$$

↪ variant `BEDC.MetaCIC.BHistSubstrate.bhistToTerm injective`

Injectivity is a useful shape fact: distinct generated histories do not collapse under the represented encoding. It is not used here as the canonical closure site, but it records that the Church-style representation preserves the visible generated structure.

9235.11.4 What the Bridge Gives

The immediate result is a typed-substrate design point. BEDC's finite binary histories can appear as ordinary represented data in the MetaCIC syntax layer. The bridge does not need a host evaluator, a parser, or a binary-string library. It is a structural map from the inductive history carrier into the inductive term carrier.

Because the output is closed, all closed-term invariants apply to the image of the bridge. A represented history can be placed in contexts where the MetaCIC development reasons about closedness, beta-step preservation, finite reduction traces, confluence statements, and bounded normalization behavior. Those metatheorems remain theorems about `Term`; the bridge supplies a large family of concrete closed `Term` inputs coming from the BEDC history substrate.

This is the relevant self-compilation step. Host-side certificate rows are generated as finite histories. The MetaCIC substrate needs closed represented values for such rows. The embedding gives that interface without changing the definition of BHist and without adding a new primitive constructor to `Term`.

9235.11.5 Unproved Boundaries

Three boundaries remain outside the checked bridge.

First, no surjectivity theorem is claimed. The module does not characterize which closed normal forms of `Term` are exactly in the image of `bhistToTerm`. Such a theorem would need a normal-form predicate and a recognizer theorem connecting the Church-style shape to the source history.

Second, no inverse decoding theorem is used by the paper. The module contains partial decoding functions from represented terms back to optional histories, but this section does not claim a full inverse theorem for all closed represented terms, nor a completeness theorem over beta-equivalent normal forms.

Third, no commutation theorem is mechanized between host-side history operations and represented MetaCIC operations. The bridge preserves the generated shape and closedness of histories, but it does not prove that every host operation on histories has a corresponding represented program whose execution commutes with `bhistToTerm`.

These boundaries keep the statement exact. The bridge establishes a closed, injective, Church-style embedding from the BEDC history carrier into the represented calculus. It does not identify the whole represented-history fragment, does not certify total decoding, and does not internalize the history-operation algebra as MetaCIC programs.

Lemma 9235.2 (Represented BHist payload decoder left inverse)**.** *For every source history h : BHist, the payload-level partial decoder returns the same history after payload encoding:*

$$\texttt{bhistFromPayloadTerm?}(\texttt{bhistPayloadToTerm}(h)) = \mathrm{some}(h).$$

Proof. Proceed by induction on the generated history h. If h is empty, the payload encoding is the represented variable for the empty branch, namely var 2. The first decoder clause sends that variable to some(e), which is exactly some(h).

Suppose $h = \mathsf{E0}(r)$. The payload encoding is the represented $b0$ branch applied to the triple-lambda wrapper around the payload encoding of r. The matching decoder clause strips this visible $b0$ shape, recursively decodes the inner payload, and repacks the result with $\mathsf{E0}$. By the induction hypothesis, the recursive decode of the inner payload is some(r). Therefore the outer clause returns some($\mathsf{E0}(r)$), which is some(h).

Suppose $h = \mathsf{E1}(r)$. The proof is the same branch computation with the represented $b1$ variable. The decoder clause strips the visible $b1$ shape, recursively decodes the inner payload, and repacks the result with $\mathsf{E1}$. The induction hypothesis gives some(r) for the recursive call, hence the outer clause returns some($\mathsf{E1}(r)$) = some(h). □

Theorem 9235.3 (Represented BHist decoder left inverse on encoded histories)**.** *For every source history h : BHist, applying the represented-history partial decoder to the encoded term of h recovers h:*

$$\texttt{bhistFromTerm?}(\texttt{bhistToTerm}(h)) = \mathrm{some}(h).$$

Equivalently, the partial decoder is a left inverse of `bhistToTerm` *on the image of the BEDC history substrate.*

Proof. By the embedding description in subsection 9235.11.3, `bhistToTerm` wraps `bhistPayloadToTerm` in the three lambdas that bind the empty, $b0$, and $b1$ branches. The top-level decoder `bhistFromTerm?` has exactly the matching wrapper clause: on a three-lambda represented history it strips the wrapper and calls `bhistFromPayloadTerm?` on the body. The remaining computation is precisely Definition 9235.2, so the result is some(h).

This proves only the image-side inverse. It does not classify all represented terms that the partial decoder accepts and does not assert surjectivity of closed `Term` values onto BHist, matching the boundary stated in subsection 9235.11.5. □

9235.12 Substrate Stack Summary

`BEDC.MetaCIC` is a first-principles substrate for the object calculus used by the self-compilation surface. Its stack starts with a closed inductive syntax, adds de Bruijn mechanics, reduction, typing, and local metatheory, and then exposes concrete examples and BEDC-carrier embeddings. Each layer is a Lean module family with a bounded responsibility. No layer imports mathlib, adds an axiom, or treats a host-kernel theorem as a theorem about the represented object calculus.

9235.12.1 Layered View

Layer	Lean surface	Role in the stack
5	`TypedExamples.*`, `BHistSubstrate`	Concrete typed terms and embeddings from BEDC carrier data into closed represented terms.
4	`Confluence.*`, `SubjectReduction.*`, `Normalization`	Local metatheory: atom diamonds, hypothesis-based subject reduction, and fuel-bounded normalization facts.
3	`Typing.Core`, `Typing.SubstitutionSurface`, `Typing.ClosedBinderDomains`	The represented typing judgment and the guard surfaces needed by binder-aware substitution.
2	`Beta`, `ClosurePreservation`	One-step beta, finite beta chains, parallel beta, and closure preservation under represented reduction.
1	`Substitution`, `ClosedTerm`, `ClosedTerm.Helpers`	Shift, substitution, closedness at a de Bruijn boundary, and syntactic helper facts.
0	`Syntax`	The term grammar: sort, variable, lambda, Pi, and application.

The arrows in this table point upward. Syntax is the object over which every other layer recurses. Substrate mechanics define the admissible movement of de Bruijn indices. Reduction consumes those mechanics in beta contraction. Typing states the object judgment over the same terms and contexts. Metatheory proves local preservation and joining facts over the represented relations. Examples and embeddings give concrete closed inhabitants of the same substrate.

9235.12.2 Layer 0: Term Syntax

The base layer is `BEDC.MetaCIC.Syntax`. It defines `Term` as a closed inductive object with five constructors: `sort`, `var`, `lam`, `pi`, and `app`. Binders are de-named. Variable occurrence is represented by a natural-number de Bruijn index, so alpha-equivalent presentations have the same constructor shape once written in this substrate.

This layer deliberately contains no nominal freshness algebra, no quotient by alpha-equivalence, and no host-level binder function. The cost is explicit index arithmetic. The gain is that all binder movement is ordinary recursive data processing over `Term`, visible to the Lean kernel and to the paper-Lean audit.

9235.12.3 Layer 1: Substrate Mechanics

The mechanics layer is split across `BEDC.MetaCIC.Substitution`, `BEDC.MetaCIC.ClosedTerm`, and `BEDC.MetaCIC.ClosedTerm.Helpers`. The two central operations are `shift` and `substitute`. Shift raises free indices at or above a cutoff. Substitution replaces the variable at a selected depth, lowers indices above the removed variable, and shifts the inserted term when the operation crosses a binder.

`ClosedAt` records the boundary at which a term has no free variables. The helper layer supplies the constructor-local facts that make this boundary usable under application, lambda, and Pi shapes. These definitions are the substrate's bookkeeping contract: every higher result about beta reduction or typing inherits the same de Bruijn boundary discipline.

9235.12.4 Layer 2: Reduction

Reduction lives in `BEDC.MetaCIC.Beta`. The primitive relation `BetaStep` has a root beta-contraction rule and congruence rules for application heads, application arguments, lambda domains, lambda bodies, Pi domains, and Pi codomains. `BetaStarStep` is the finite reflexive-transitive closure used for finite traces. `BetaParallel` is the parallel relation used by the confluence surface.

`BEDC.MetaCIC.ClosurePreservation` connects reduction back to the mechanics layer. A beta contraction is implemented by substitution, so closure preservation for reduction depends on the closure-preserving substitution facts from Layer 1. This is the first place where the stack forces the algebra of binders to match the operational semantics.

9235.12.5 Layer 3: Typing

Typing is centered on `BEDC.MetaCIC.Typing.Core`. Its main predicate is `HasType`, an inductive relation between contexts, represented terms, and represented types. The formation rules cover sorts, variables, Pi types, lambdas, and applications over the explicit-context de Bruijn substrate.

The guard surfaces live in `BEDC.MetaCIC.Typing.SubstitutionSurface` and `BEDC.MetaCIC.Typing.ClosedB` They isolate the substitution and binder-domain hypotheses needed by typing preservation arguments. This matters because a domain introduced outside a binder must be read one level deeper inside the body. The typing layer therefore uses the same shift-aware interpretation as the syntax and substitution layers, rather than treating binder domains as inert annotations.

9235.12.6 Layer 4: Metatheory

The confluence surface is in `BEDC.MetaCIC.Confluence.*`. Atom diamonds are mechanized for sort and variable sources, where no real beta fork can occur. The full diamond over all compound beta forks is not asserted. Application forks need residual reasoning that commutes parallel reduction with shift-aware substitution, and that theorem is exactly where the represented substrate remains restrictive.

The subject-reduction surface is in `BEDC.MetaCIC.SubjectReduction.*`. It is hypothesis-based: the beta case and congruence cases are assembled under the substitution-preservation and domain-stability assumptions stated by the typing surface. Thus subject reduction is not obtained by importing a host theorem. It is a theorem over the represented `HasType` and `BetaStep` relations, at the strength supported by the current substrate hypotheses.

`BEDC.MetaCIC.Normalization` supplies a bounded normalizer and atomic normal-form facts. The normalizer terminates because its fuel decreases in the host recursion, not because the repre-

sented calculus has a checked strong normalization theorem. The normalization layer is therefore operational and finite-trace oriented.

9235.12.7 Layer 5: Examples and Embeddings

The example layer is the concrete gallery under `BEDC.MetaCIC.TypedExamples.*`. It contains sort, Pi, identity, application, projection, constant-form, and polymorphic witnesses over the same `HasType` relation used by the metatheory. These examples are not a separate evaluator surface. They are ordinary represented terms with ordinary typing derivations.

`BEDC.MetaCIC.BHistSubstrate` connects the finite BEDC history carrier to the MetaCIC term language. Its embedding sends each generated binary history to a closed represented term. This supplies a bridge from raw BEDC certificate data into the object calculus without adding a primitive history constructor to `Term`.

9235.12.8 Cross-Cutting Invariants

The whole stack is mathlib-free. Its Lean files are built from the Lean 4 standard environment and project-local inductive definitions, functions, and theorems. The project gate rejects `axiom` declarations and maintains the zero-`sorry` discipline for public theorem bodies.

The stricter purity audit also checks transitive dependencies. It rejects `Classical.choice`, `Quot.sound`, and `propext`. Thus the checked MetaCIC theorems are CIC-derived in the relevant project sense: constructive, quotient-free, and not backed by hidden classical or extensional host principles.

The paper markers are an independent audit axis. A Lean-side theorem marker must name an object that exists under `lean4/BEDC/`. Conversely, a marker does not close an informal theorem by itself. The stack is reported on two axes: what the paper claims about the represented calculus, and what Lean has checked for the corresponding definitions and theorems.

9235.12.9 Open Seam

The limiting algebraic point is substitution composition. In the naive zero-depth form, composition is false for the represented operations; the counterexample is recorded in subsection 9235.7.4. The failure is not a peripheral notation issue. It shows that the substrate must account for closure, shifting, and binder depth before a composition theorem can be used safely.

This same constraint reaches the upper layers. Full confluence needs residual substitution lemmas for application beta forks. Subject reduction depends on substitution preservation at the strength stated by the typing guard surface. Strong normalization would need an admissible-type and reducibility infrastructure stable under the same binder mechanics. The stack is therefore coherent from bottom to top: the highest open metatheory is bounded by the lowest shift-aware algebra that the substrate has made available.

9235.13 Typed Examples Gallery

The typed examples gallery is the reader-facing index for closed-term witnesses inside ‘`BEDC.MetaCIC.TypedExampl`
It complements the representational adequacy argument in section 9235.4: the adequacy claim explains why the first-principles substrate is the right target, while this gallery lists the concrete closed terms whose typing or reduction facts have been mechanized on that substrate.

9235.13.1 Why a gallery

'`BEDC.MetaCIC`' is a first-principles substrate. It does not inherit a host-kernel reflection API, a built-in quotation mechanism, or a host-level '`decide`' path for typing judgments. Each displayed typing fact is built by an explicit chain of constructors for the object syntax, object typing relation, and object reduction relation. The gallery is therefore not a convenience catalogue alone: it is the compact map of the mechanized closed-term witnesses that exercise the representational layer.

9235.13.2 Sibling index

Lean sibling	Role
'`Sorts.lean`'	Sort-level and variable lookup typing examples over small contexts.
'`Identity.lean`'	Identity-shaped $\lambda x : T.\ x$ witnesses, dependent identity forms, and simple projections.
'`ConstForms.lean`'	Constant-function forms and their staged applications.
'`AppForms.lean`'	Application-shaped witnesses, including reduced and unreduced forms over empty and small contexts.
'`PiForms.lean`'	Pi-type formation and lambda terms whose types are Pi forms.
'`BetaRedex.lean`'	Beta-redex witnesses and corresponding '`BetaStep`' or '`BetaStarStep`' instances.
'`ChurchGallery.lean`'	Church encodings for Booleans, naturals, pairs, and options.
'`Polymorphic.lean`'	Pi-polymorphic identity, constant, apply, flip, and compose forms.

9235.13.3 Mechanized witnesses

Example	Lean witness	Description
Sorts	↪variant `BEDC.MetaCIC.var zero in sort ctx`	Variable zero in a one-sort context.
Sorts	↪variant `BEDC.MetaCIC.var one in double sort ctx`	Variable one in a two-sort context.
Sorts	↪variant `BEDC.MetaCIC.middle var in triple sort`	Middle variable lookup in a three-sort context.
Sorts	↪variant `BEDC.MetaCIC.deep var in quad sort`	Deep variable lookup in a four-sort context.
Sorts	↪variant `BEDC.MetaCIC.mixed ctx vars`	Variable lookup in a mixed sort-and-Pi context.
Sorts	↪variant `BEDC.MetaCIC.mixed ctx var zero`	Front variable lookup in a mixed context.
Sorts	↪variant `BEDC.MetaCIC.mixed var in three ctx`	Interior variable lookup in a mixed three-entry context.
Sorts	↪variant `BEDC.MetaCIC.var in quint sort`	Deep variable lookup in a five-sort context.
Identity	↪variant `BEDC.MetaCIC.lam dependent identity`	Dependent identity lambda.
Identity	↪variant `BEDC.MetaCIC.carrier identity in sort ctx`	Identity over a carrier variable in a sort context.

Identity	↪ variant BEDC.MetaCIC.lam inner to outer in double sort	Lambda returning an outer variable.
Identity	↪ variant BEDC.MetaCIC.triple sort lam	Three-level sort lambda witness.
Identity	↪ variant BEDC.MetaCIC.polymorphic identity	Pi-polymorphic identity witness.
Identity	↪ variant BEDC.MetaCIC.nested poly id	Nested polymorphic identity witness.
Identity	↪ variant BEDC.MetaCIC.lam constructs dependent pi	Lambda forming a dependent Pi target.
Identity	↪ variant BEDC.MetaCIC.double lam pi construct	Two-lambda Pi construction.
Identity	↪ variant BEDC.MetaCIC.first of two	First projection over two arguments.
Identity	↪ variant BEDC.MetaCIC.second of two	Second projection over two arguments.
Identity	↪ variant BEDC.MetaCIC.ignore arg	Lambda that ignores an argument.
Identity	↪ variant BEDC.MetaCIC.double poly first	Polymorphic first projection.
Identity	↪ variant BEDC.MetaCIC.second proj	Second projection witness.
ConstForms	↪ variant BEDC.MetaCIC.app const sort in empty ctx	Constant sort function applied in the empty context.
ConstForms	↪ variant BEDC.MetaCIC.const sort app	Application of a constant sort function.
ConstForms	↪ variant BEDC.MetaCIC.double lam const	Two-argument constant lambda.
ConstForms	↪ variant BEDC.MetaCIC.double lam const first app	First application of a two-argument constant lambda.
ConstForms	↪ variant BEDC.MetaCIC.double lam const repeated app	Repeated application of a two-argument constant lambda.
ConstForms	↪ variant BEDC.MetaCIC.twice applied const	Fully applied constant function witness.
ConstForms	↪ variant BEDC.MetaCIC.first arg proj	First-argument projection as a constant form.
ConstForms	↪ variant BEDC.MetaCIC.second arg proj	Second-argument projection as a constant form.
ConstForms	↪ variant BEDC.MetaCIC.const sort 1 applied	One-step application of a sort constant.
ConstForms	↪ variant BEDC.MetaCIC.applied const sort	Applied constant-sort witness.
AppForms	↪ variant BEDC.MetaCIC.id applied to sort type	Type of identity applied to sort.
AppForms	↪ variant BEDC.MetaCIC.poly id at sort full	Fully instantiated polymorphic identity at sort.

AppForms	↪ variant BEDC.MetaCIC.poly id double app	Polymorphic identity with two applications.
AppForms	↪ variant BEDC.MetaCIC.poly id to pi type	Polymorphic identity applied to a Pi type.
AppForms	↪ variant BEDC.MetaCIC.app id sort in empty ctx	Identity application in the empty context.
AppForms	↪ variant BEDC.MetaCIC.id to id in empty	Identity function applied to identity in the empty context.
AppForms	↪ variant BEDC.MetaCIC.applied pi constructor in empty	Applied Pi constructor in the empty context.
AppForms	↪ variant BEDC.MetaCIC.applied pi constructor in empty result eq	Result equality for the applied Pi constructor.
AppForms	↪ variant BEDC.MetaCIC.id applied to pi	Identity applied to a Pi expression.
AppForms	↪ variant BEDC.MetaCIC.id applied pi value	Value witness for identity applied to Pi.
AppForms	↪ variant BEDC.MetaCIC.id applied to pi result eq	Result equality for identity applied to Pi.
AppForms	↪ variant BEDC.MetaCIC.id on pi sort sort	Identity on the sort-to-sort Pi type.
AppForms	↪ variant BEDC.MetaCIC.pi value id	Identity value for a Pi-shaped term.
AppForms	↪ variant BEDC.MetaCIC.proj for pi arg	Projection with a Pi argument.
AppForms	↪ variant BEDC.MetaCIC.identity on pi type	Identity over a Pi type.
AppForms	↪ variant BEDC.MetaCIC.id for pi applied	Applied identity for a Pi argument.
AppForms	↪ variant BEDC.MetaCIC.app id var in sort ctx	Identity applied to a variable in a sort context.
AppForms	↪ variant BEDC.MetaCIC.dep id applied to arg in sort ctx	Dependent identity applied to a parameter in a sort context.
AppForms	↪ variant BEDC.MetaCIC.dep id applied to arg result in sort ctx	Result typing for dependent identity application.
AppForms	↪ variant BEDC.MetaCIC.applied dep id in double sort ctx	Applied dependent identity in a two-sort context.
AppForms	↪ variant BEDC.MetaCIC.app with pi context	Application in a context containing a Pi type.
AppForms	↪ variant BEDC.MetaCIC.app pi var to sort	Pi variable applied to sort.
AppForms	↪ variant BEDC.MetaCIC.app pi function	Application of a Pi-typed function.
AppForms	↪ variant BEDC.MetaCIC.app pi ctx to sort	Pi-context application targeting sort.
AppForms	↪ variant BEDC.MetaCIC.app two pi vars	Application between two Pi-context variables.

AppForms	↪ variant BEDC.MetaCIC.app pi deep var	Application involving a deeper Pi variable.
AppForms	↪ variant BEDC.MetaCIC.apply function arg in empty	Function-argument application in the empty context.
AppForms	↪ variant BEDC.MetaCIC.app id pi var sort in sort ctx	Identity application over a Pi variable in a sort context.
AppForms	↪ variant BEDC.MetaCIC.pi var sort then sort id app in sort ctx	Pi variable followed by sort identity application.
AppForms	↪ variant BEDC.MetaCIC.apply function arg in value ctx	Function-argument application in a value context.
AppForms	↪ variant BEDC.MetaCIC.dep pi app	Dependent Pi application.
AppForms	↪ variant BEDC.MetaCIC.apply function arg	Function-argument application witness.
AppForms	↪ variant BEDC.MetaCIC.applied poly id	Applied polymorphic identity witness.
PiForms	↪ variant BEDC.MetaCIC.two sort vars	Pair of sort variables in context.
PiForms	↪ variant BEDC.MetaCIC.lam pi var0 var1	Lambda whose type is a Pi over variables zero and one.
PiForms	↪ variant BEDC.MetaCIC.lam pi var var	Lambda over variable-shaped Pi data.
PiForms	↪ variant BEDC.MetaCIC.pi var sort in sort ctx	Pi formation from a variable to sort.
PiForms	↪ variant BEDC.MetaCIC.pi using arrow in ctx	Arrow-like Pi formation in context.
PiForms	↪ variant BEDC.MetaCIC.pi sort sort in empty ctx	Sort-to-sort Pi in the empty context.
PiForms	↪ variant BEDC.MetaCIC.pi sort dependent	Dependent Pi over sort.
PiForms	↪ variant BEDC.MetaCIC.nested pi sort sort	Nested sort-to-sort Pi.
PiForms	↪ variant BEDC.MetaCIC.triple pi sort	Three-level Pi over sort.
PiForms	↪ variant BEDC.MetaCIC.nested pi dep in empty	Nested dependent Pi in the empty context.
PiForms	↪ variant BEDC.MetaCIC.nested pi var sort	Nested Pi using a variable and sort.
PiForms	↪ variant BEDC.MetaCIC.map pi in sort ctx	Map-like Pi in a sort context.
PiForms	↪ variant BEDC.MetaCIC.triple lam pi construction	Three-lambda Pi construction.
PiForms	↪ variant BEDC.MetaCIC.double poly pi	Double-polymorphic Pi witness.
PiForms	↪ variant BEDC.MetaCIC.double arrow in sort	Double-arrow Pi in sort.
PiForms	↪ variant BEDC.MetaCIC.pi var var in quad sort	Pi over variables in a four-sort context.

PiForms	↪variant BEDC.MetaCIC.nested pi chain	Chain of nested Pi forms.
PiForms	↪variant BEDC.MetaCIC.pi function to sort in empty	Function-to-sort Pi in the empty context.
PiForms	↪variant BEDC.MetaCIC.function of function	Function-of-function Pi witness.
PiForms	↪variant BEDC.MetaCIC.nested shifted pi in sort	Nested shifted Pi in sort.
BetaRedex	↪variant BEDC.MetaCIC.id sort redex	Beta step for identity at sort.
BetaRedex	↪variant BEDC.MetaCIC.id sort redex typed	Typing fact for identity-at-sort redex.
BetaRedex	↪variant BEDC.MetaCIC.fn sort to sort redex	Beta step for a sort-to-sort function.
BetaRedex	↪variant BEDC.MetaCIC.fn sort to sort redex typed	Typing fact for the sort-to-sort redex.
BetaRedex	↪variant BEDC.MetaCIC.const sort redex	Beta step for a constant sort function.
BetaRedex	↪variant BEDC.MetaCIC.const sort redex typed	Typing fact for the constant sort redex.
BetaRedex	↪variant BEDC.MetaCIC.inner id redex	Beta step inside an identity-shaped term.
BetaRedex	↪variant BEDC.MetaCIC.inner sort id redex	Inner sort identity beta step.
BetaRedex	↪variant BEDC.MetaCIC.inner sort id redex typed	Typing fact for the inner sort identity redex.
BetaRedex	↪variant BEDC.MetaCIC.app right id sort redex	Beta step in the right side of an application.
BetaRedex	↪variant BEDC.MetaCIC.app right id sort redex typed	Typing fact for the right-application redex.
BetaRedex	↪variant BEDC.MetaCIC.pi domain redex	Beta step in a Pi domain.
BetaRedex	↪variant BEDC.MetaCIC.pi domain redex typed	Typing fact for the Pi-domain redex.
BetaRedex	↪variant BEDC.MetaCIC.pi cod redex	Beta step in a Pi codomain.
BetaRedex	↪variant BEDC.MetaCIC.pi cod redex typed	Typing fact for the Pi-codomain redex.
BetaRedex	↪variant BEDC.MetaCIC.poly id two step redex	Two-step beta reduction for polymorphic identity.
BetaRedex	↪variant BEDC.MetaCIC.poly id two step redex typed	Typing fact for the two-step polymorphic redex.
BetaRedex	↪variant BEDC.MetaCIC.k sort partial redex	Partial beta step for a sort-level K combinator.
BetaRedex	↪variant BEDC.MetaCIC.k sort partial redex typed	Typing fact for the partial K redex.

BetaRedex	↪ variant BEDC.MetaCIC.k sort two step redex	Two-step beta reduction for the sort-level K combinator.
BetaRedex	↪ variant BEDC.MetaCIC.k sort two step redex typed	Typing fact for the two-step K redex.
BetaRedex	↪ variant BEDC.MetaCIC.app left projection redex	Beta step for a left projection application.
BetaRedex	↪ variant BEDC.MetaCIC.app left projection redex typed	Typing fact for the left projection redex.
ChurchGallery	↪ variant BEDC.MetaCIC.church bool type	Church Boolean type.
ChurchGallery	↪ variant BEDC.MetaCIC.church true	Church true term.
ChurchGallery	↪ variant BEDC.MetaCIC.church false	Church false term.
ChurchGallery	↪ variant BEDC.MetaCIC.church nat type	Church natural-number type.
ChurchGallery	↪ variant BEDC.MetaCIC.church zero	Church zero term.
ChurchGallery	↪ variant BEDC.MetaCIC.church one	Church one term.
ChurchGallery	↪ variant BEDC.MetaCIC.church nat type in ctx	Church natural-number type in context.
ChurchGallery	↪ variant BEDC.MetaCIC.church succ type	Church successor type.
ChurchGallery	↪ variant BEDC.MetaCIC.church succ	Church successor term.
ChurchGallery	↪ variant BEDC.MetaCIC.church identity type	Church identity type.
ChurchGallery	↪ variant BEDC.MetaCIC.church identity	Church identity term.
ChurchGallery	↪ variant BEDC.MetaCIC.church const type	Church constant-function type.
ChurchGallery	↪ variant BEDC.MetaCIC.church const	Church constant-function term.
ChurchGallery	↪ variant BEDC.MetaCIC.church bool elim sort	Boolean eliminator over sort.
ChurchGallery	↪ variant BEDC.MetaCIC.church pair type	Church pair type former.
ChurchGallery	↪ variant BEDC.MetaCIC.church pair ab type	Church pair type at two arguments.
ChurchGallery	↪ variant BEDC.MetaCIC.church mk pair type	Pair constructor type.
ChurchGallery	↪ variant BEDC.MetaCIC.church mk pair	Pair constructor term.
ChurchGallery	↪ variant BEDC.MetaCIC.church fst type	First projection type.
ChurchGallery	↪ variant BEDC.MetaCIC.church fst	First projection term.

ChurchGallery	↪ variant BEDC.MetaCIC.church snd type	Second projection type.
ChurchGallery	↪ variant BEDC.MetaCIC.church snc type	Constant-style helper type for the second projection.
ChurchGallery	↪ variant BEDC.MetaCIC.church snd	Second projection term.
ChurchGallery	↪ variant BEDC.MetaCIC.church pair former	Pair type former witness.
ChurchGallery	↪ variant BEDC.MetaCIC.church pair bool nat type	Boolean-natural Church pair type.
ChurchGallery	↪ variant BEDC.MetaCIC.church option at type	Church option type at an argument.
ChurchGallery	↪ variant BEDC.MetaCIC.church option type	Church option type former.
ChurchGallery	↪ variant BEDC.MetaCIC.church none type	None constructor type.
ChurchGallery	↪ variant BEDC.MetaCIC.church none	None constructor term.
ChurchGallery	↪ variant BEDC.MetaCIC.church some type	Some constructor type.
ChurchGallery	↪ variant BEDC.MetaCIC.church some	Some constructor term.
ChurchGallery	↪ variant BEDC.MetaCIC.church option in case ctx type	Option type in a case-analysis context.
ChurchGallery	↪ variant BEDC.MetaCIC.church case option type	Option case-analysis type.
ChurchGallery	↪ variant BEDC.MetaCIC.church case option	Option case-analysis term.
Polymorphic	↪ variant BEDC.MetaCIC.poly identity type	Pi-polymorphic identity type.
Polymorphic	↪ variant BEDC.MetaCIC.poly const type	Pi-polymorphic constant-function type.
Polymorphic	↪ variant BEDC.MetaCIC.poly apply type	Pi-polymorphic apply type.
Polymorphic	↪ variant BEDC.MetaCIC.poly flip type	Pi-polymorphic flip type.
Polymorphic	↪ variant BEDC.MetaCIC.poly compose type	Pi-polymorphic compose type.
Polymorphic	↪ variant BEDC.MetaCIC.poly identity at sort type	Polymorphic identity instantiated at sort.

9235.13.4 Boundary of the gallery

The gallery records closed-term witness patterns. It does not claim a catalogue of non-closed terms, although the object syntax and context machinery can state such judgments. It also does not present general typing in arbitrary non-empty contexts: several witnesses use small explicit contexts, but the dominant path for the self-compilation metatheory is the closed-term path. Finally, it does not claim reduction over open terms; the displayed reduction witnesses are closed

or are paired with closed typing facts where the file states them.

9235.13.5 Contributor discipline

A witness belongs in this gallery when it is a concrete '`theorem`' inside '`lean4/BEDC/MetaCIC/TypedExamples/`' and its target name is stable under '`BEDC.MetaCIC`'. Choose the sibling by term shape rather than by proof technique: sort and variable facts go in '`Sorts.lean`', identity-like lambda terms in '`Identity.lean`', constant functions in '`ConstForms.lean`', applications in '`AppForms.lean`', Pi formation in '`PiForms.lean`', beta witnesses in '`BetaRedex.lean`', Church encodings in '`ChurchGallery.lean`', and Pi-polymorphic combinators in '`Polymorphic.lean`'. The paper row should use '`\leanvariant`', not '`\leanchecked`', because the gallery is an index of sibling witnesses rather than the canonical primary site for a single paper theorem.

9235.14 Metatheory Milestone Consolidation

This section gives a single reader-facing account of the metatheory now available around `BEDC.MetaCIC`. The account is deliberately two-sided. On one side it records the checked Lean surface: represented syntax, closure, substitution, reduction, confluence components, subject reduction interfaces, typed examples, host-gallery transport, and the BHist embedding. On the other side it places that surface next to the standard textbook routes for the Calculus of Inductive Constructions and related pure type systems.

The point of the comparison is scope control. BEDC is not a replacement kernel and not a full formalization of CIC. It is a mathlib-free, zero-axiom, zero-sorry represented calculus inside Lean 4, with enough first-principles metatheory to support the self-compilation surface used in this paper. The checked pieces are meaningful because they are proved over the represented object language itself, not imported from host conversion, host normalization, or a library theorem about lambda calculus.

9235.14.1 What `BEDC.MetaCIC` Achieves

The represented substrate is a single-sort dependent-function core. Its term syntax has five constructors: sort, de Bruijn variable, application, lambda, and Pi. Context lookup is a first-principles list lookup. Shift and substitution are native recursive operations on the represented syntax, and the closure predicate `ClosedAt` records exactly which de Bruijn variables are in scope at a given boundary.

The substrate is shift-aware in the strict de Bruijn sense. Descent under a binder changes the relevant cutoff, and substitution below a binder shifts the inserted term. This is why closure preservation is a theorem rather than a notation convention. The checked closure layer says that closed terms remain closed under shift and substitution at the relevant boundary, and that beta reduction preserves the same closure predicate.

```
↪ variant  BEDC.MetaCIC.shift closed
↪ variant  BEDC.MetaCIC.substitute closed
↪ variant  BEDC.MetaCIC.substitute preserves closed at
↪ variant  BEDC.MetaCIC.betaStep preserves closed
↪ variant  BEDC.MetaCIC.betaStarStep preserves closed
```

These facts are not cosmetic. They are the algebraic guardrail that keeps the MetaCIC reduction relation inside the intended closed fragment. In particular, the one-step theorem covers beta contraction and all congruence constructors; the finite-chain theorem lifts it along the represented

reflexive-transitive closure `BetaStarStep`. Hence every finite represented beta trace starting from a closed term remains closed at every displayed point of the trace.

The confluence surface is split into atom results and compound-shape assemblers. For atoms, sort and variable sources cannot produce a genuine fork. Their beta-star and parallel-beta forks join at the source itself. The checked atom diamonds record the fully closed part of the confluence story:

```
↪variant BEDC.MetaCIC.betaStarStep diamond sort
↪variant BEDC.MetaCIC.betaStarStep diamond var
↪variant BEDC.MetaCIC.betaParallel sort diamond
↪variant BEDC.MetaCIC.betaParallel var diamond
```

For compound terms, the checked layer supplies the structural components needed by the standard parallel-reduction path. Application, lambda, and Pi diamonds can be assembled from diamonds of their components. Lambda and Pi shape forks reduce to domain-body and domain-codomain obligations. Application forks are exposed by a full inversion theorem, because an application branch may either remain an application or contract after the function side exposes a lambda.

```
↪variant BEDC.MetaCIC.betaParallel app diamond of components
↪variant BEDC.MetaCIC.betaParallel lam diamond of components
↪variant BEDC.MetaCIC.betaParallel pi diamond of components
↪variant BEDC.MetaCIC.betaParallel lam shape diamond
↪variant BEDC.MetaCIC.betaParallel pi shape diamond
↪variant BEDC.MetaCIC.betaParallel app full inv
↪variant BEDC.MetaCIC.betaParallel app shape diamond from substitution
↪variant BEDC.MetaCIC.betaParallel diamond from substitution
```

The last two targets make the boundary explicit. A full parallel diamond is available from a substitution-compatibility hypothesis. The Lean theorem does not hide that hypothesis; it states exactly where the remaining algebraic content must enter. In textbook prose the missing law is often described as "substitution commutes with residual reduction." In this substrate it must be a concrete law about `substitute 0` with the same shifts and cutoffs used by `BEDC.MetaCIC.substitute`.

Subject reduction has a similarly explicit shape. The central theorem is a hypothesis-based theorem over the represented typing and beta-step relations. Its beta case uses the beta-substitution-preservation interface; its congruence cases use the application-argument and binder-domain stability interfaces. This is stronger than a mere statement of intent, because the ordinary beta, app, lambda, and Pi cases are checked as Lean theorems over the object calculus.

```
↪variant BEDC.MetaCIC.subject reduction beta
↪variant BEDC.MetaCIC.subject reduction congApp1
↪variant BEDC.MetaCIC.subject reduction congApp2
↪variant BEDC.MetaCIC.subject reduction congLam
↪variant BEDC.MetaCIC.subject reduction congPi
↪variant BEDC.MetaCIC.subject reduction
↪variant BEDC.MetaCIC.subject reduction retype
Lean4✓ BEDC.MetaCIC.not subjectReductionDischargeBundle
```

The subject-reduction theorem should be read as a precise local theorem, not as a disguised claim that all standard preservation obligations have been discharged. The beta-substitution interface remains a carrier obligation, while the application-argument, lambda-domain, and Pi-domain interfaces are blocked by checked obstruction facts. The closed-term path is also constrained by the

checked obstruction showing that a closed outer term may contain a locally open redex argument when the redex is examined under binders.

↪ variant BEDC.MetaCIC.closedBinderRedexTerm closed
↪ variant BEDC.MetaCIC.closedBinderRedexTerm steps
↪ variant BEDC.MetaCIC.closedBinderRedexTerm argument not closed
↪ variant BEDC.MetaCIC.closed term step has unclosed local argument

The normalization surface is intentionally modest. Operationally, the paper uses a bounded normalizer reading: a fuel-indexed host computation follows a chosen finite beta path, and closure preservation is carried by the checked finite-trace theorems. The checked atom-level termination content is the normal-form behavior of sort and closed variable atoms: no beta successor can start from such a source, and a typed closed atom is therefore beta-inert.

↪ variant BEDC.MetaCIC.sort no beta
↪ variant BEDC.MetaCIC.closed var no beta
↪ variant BEDC.MetaCIC.closed atom normal form

This is the atom-level strong-normalization reading used here: for atom sources there is no first step, so every beta path is already finished. It is not a global strong-normalization theorem for all well-typed represented terms. The reducibility-candidates route is therefore a frontier for the represented calculus, not a completed adequacy theorem in the Lean surface.

The typed examples gallery gives concrete pressure on the syntax, typing, and reduction definitions. It contains Church-encoded Booleans, naturals, pairs, and options; polymorphic identity, constant, flip, and compose forms; and a family of beta-redex witnesses. The examples are not produced by quotation from Lean expressions. They are represented terms whose typing and reduction evidence is constructed inside `BEDC.MetaCIC`.

↪ variant BEDC.MetaCIC.church bool type
↪ variant BEDC.MetaCIC.church true
↪ variant BEDC.MetaCIC.church false
↪ variant BEDC.MetaCIC.church nat type
↪ variant BEDC.MetaCIC.church zero
↪ variant BEDC.MetaCIC.church one
↪ variant BEDC.MetaCIC.church pair type
↪ variant BEDC.MetaCIC.church mk pair
↪ variant BEDC.MetaCIC.church option type
↪ variant BEDC.MetaCIC.church none
↪ variant BEDC.MetaCIC.church some
↪ variant BEDC.MetaCIC.poly identity type
↪ variant BEDC.MetaCIC.poly const type
↪ variant BEDC.MetaCIC.poly flip type
↪ variant BEDC.MetaCIC.poly compose type
↪ variant BEDC.MetaCIC.id sort redex
↪ variant BEDC.MetaCIC.id sort redex typed
↪ variant BEDC.MetaCIC.poly id two step redex
↪ variant BEDC.MetaCIC.poly id two step redex typed
↪ variant BEDC.MetaCIC.k sort two step redex
↪ variant BEDC.MetaCIC.k sort two step redex typed

The host bridge gives a second view of the same gallery. Host-side Lean definitions record the familiar computational behavior of identity, constant, flip, compose, Church constructors, and selected redexes by reflexive equalities. The transport side then points to the represented MetaCIC

witnesses for corresponding typing and reduction facts. This is not a kernel soundness theorem from host elaboration to MetaCIC. It is a checked alignment gallery: named host examples and named represented examples are paired at the level of concrete witnesses.

```
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host identity beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host const beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host flip beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host compose beta
↪ variant BEDC.HostBridge.MetaCICTransport.transport identity well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport const well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport flip well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport compose well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church true well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church false well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church zero well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church one well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport id sort redex step
↪ variant BEDC.HostBridge.MetaCICTransport.transport id sort redex well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport poly id two step redex star
↪ variant BEDC.HostBridge.MetaCICTransport.transport poly id two step redex well typed
```

The BHist bridge connects the self-compilation substrate back to the finite-history world of BEDC. The embedding `bhistToTerm` turns a `BHistIdx` into a represented MetaCIC term, and the closure theorem says that the result is closed at boundary 0. This is exactly the kind of bridge needed by a closed-atom self-compilation surface: histories can be represented as terms without opening a de Bruijn variable.

```
↪ variant BEDC.MetaCIC.BHistSubstrate.bhistToTerm closed
↪ variant BEDC.MetaCIC.BHistSubstrate.bhistToTerm injective
```

9235.14.2 Comparison with Textbook CIC Metatheory

The textbook metatheory of CIC is usually organized around a rich kernel with universes, products, conversion, inductive objects, and a semantic or syntactic normalization argument. Coquand–Werner style presentations are commonly read through normalization by evaluation or closely related semantic normalization methods. Those routes exploit a carefully chosen semantic model of terms, types, and conversion. BEDC takes a more direct first-principles path: it defines the object syntax, object typing, object beta relation, and object metatheory inside Lean 4, then proves local facts about those inductive objects.

The difference is not merely aesthetic. Normalization by evaluation is a semantic proof architecture; the evaluator and reifier are part of the proof that well-typed terms normalize. BEDC's bounded normalizer is not such a semantic normalization proof. It is an operational finite approximation whose termination comes from host fuel. The checked theorem around it is closure preservation along represented beta traces, not a theorem that all well-typed represented terms normalize.

Pure type systems give another useful comparison point. In the Barendregt style, a calculus is presented by a set of sorts, axioms, and product rules. The BEDC fragment can be read as a single-sort sub-PTS: it has one represented sort, Pi formation, lambda introduction, application elimination, and explicit beta reduction, but it does not carry a universe hierarchy, cumulativity, or the full set of conversion principles used by CIC kernels. The single-sort choice keeps the object calculus small enough for mathlib-free proof from first principles.

Reducibility candidates provide the standard Girard/Tait route to strong normalization. That route interprets types as sets of strongly normalizing terms stable under reduction, then proves adequacy by induction on typing derivations. BEDC has the syntactic substrate needed to state this route: represented typing, represented beta, closure, finite reduction chains, and atom-level normal-form facts. What is not present is the full adequacy proof: an interpretation of every admissible represented type as a candidate and an induction showing that every typing derivation lands in the corresponding candidate.

The comparison therefore has a precise shape:

Textbook route	Standard role	BEDC reading
Coquand–Werner CIC	semantic normalization and kernel-scale conversion metatheory	direct object-language proofs over a narrower represented calculus
Pure type systems	sorts, axioms, and product rules as a parametric calculus schema	single-sort dependent-function sub-PTS with explicit beta only
Tait–Martin-Lof parallel reduction	diamond proof route for beta confluence	atom diamonds and compound-shape assemblers, with application residuals still isolated as substitution compatibility
Girard/Tait reducibility candidates	strong-normalization adequacy by semantic interpretation of types	frontier over the represented calculus, with atom-level normal-form facts already checked

This reading keeps the standard references conceptually visible without pretending that their full theorems have been imported. The BEDC theorem names are theorem names about the represented calculus. They do not mention the host kernel's own conversion relation, and they do not rely on host metatheory as a black box.

9235.14.3 Frontier

The first frontier is the full `BetaParallel` diamond without a closed-term precondition. The checked theorem `betaParallel_diamond_from_substitution` identifies the exact remaining input: parallel substitution compatibility. Proving that input in the present de Bruijn substrate requires a binder-aware commutation theorem for parallel residuals and substitution.

The second frontier is the complete discharge of the three subject-reduction hypotheses in the general form. The theorem `subject_reduction` already packages the main induction once beta substitution preservation, application-argument stability, lambda-domain stability, and Pi-domain stability are available. The closed-term path is natural, but the local-redex obstruction shows why closedness alone does not erase all binder bookkeeping.

The third frontier is a strong-normalization theorem for the represented calculus. The current checked facts prove that atoms are beta-inert and that finite traces preserve closedness. A full theorem would require an admissible type grammar, reducibility candidates for that grammar, and an adequacy proof over represented typing derivations.

The fourth frontier is the weak closed-term form of substitution composition. The unrestricted zero-depth equation is refuted by a checked counterexample. Positive shape-local facts already exist for closed sort, variable, application, lambda-anchor, and Pi-anchor cases, but the theorem needed by the application residual proof must be stated in the exact closed form that the shift-aware substitution operation satisfies.

```
↪ variant BEDC.MetaCIC.substitute substitute zero zero unclosed counterexample
↪ variant BEDC.MetaCIC.substitute substitute zero zero closed sort
↪ variant BEDC.MetaCIC.substitute substitute zero zero closed var
↪ variant BEDC.MetaCIC.substitute substitute zero zero closed app
↪ variant BEDC.MetaCIC.substitute substitute zero zero lam closed anchor
↪ variant BEDC.MetaCIC.substitute substitute zero zero pi closed anchor
```

9235.14.4 Boundary Statement

BEDC does not pursue a universe hierarchy or cumulativity in `BEDC.MetaCIC`. The represented calculus has a single sort and the paper reads it as a universe-zero dependent-function core. This excludes the kernel-scale universe constraints and cumulativity rules that a full CIC formalization would need.

BEDC does not pursue inductive families inside the MetaCIC substrate. The object language has no represented inductive declarations, constructors, recursors, motive elaboration, or large eliminators. Church encodings in the typed gallery are represented terms in the dependent-function fragment, not host inductive datatypes transported into the object calculus.

BEDC does not pursue propositional irrelevance. The object calculus has no separate proof-irrelevant proposition sort and no quotient by proof identity. All checked theorems are Lean theorems about the represented syntax and relations.

BEDC does not pursue eta-conversion. The reduction relation is beta-only: lambda eta and Pi eta are not reduction constructors, and the confluence and subject-reduction surfaces are stated for the beta relation that is actually encoded.

BEDC does not pursue pattern matching from inductive types beyond the closed atoms and Church-encoded examples displayed in the gallery. The object language has no case tree language and no pattern compiler. This keeps the self-compilation surface tied to the finite closed-term and closed-history bridge it needs.

9235.14.5 Why This Boundary

The first reason is axiom purity. The development is mathlib-free and audited against extra axioms. In particular, the project excludes reliance on `Classical.choice`, `Quot.sound`, and `propext` in the checked theorem dependency chain. That restriction rules out many convenient shortcuts for quotienting syntax, using classical well-founded arguments, or collapsing propositions by extensionality. The substrate must be built from inductive syntax, recursive functions, and explicit constructors.

The second reason is methodological. The self-compilation chapter values substrate coherence over maximal expressivity. A small represented calculus with explicit closure, substitution, beta, typing, and finite examples gives a clear audit trail. A larger calculus with universes, inductives, conversion, and pattern matching would be closer to a production kernel, but it would also move the paper away from the first-principles invariant that makes every claim traceable to a named Lean object.

The third reason is sufficiency for the BEDC goal. The self-compilation surface needs closed-atom and finite-history meta-reasoning, not a full mechanization of CIC. Histories must embed as closed represented terms. Gallery witnesses must exercise the dependent-function core. Reduction traces must preserve closedness. Atom sources must be beta-inert. These are exactly the checked components listed above. The larger textbook theorems remain the right comparison class, but the paper's claim is intentionally narrower.

The resulting boundary is neither a defect nor a hidden theorem. It is the chosen contract for this represented calculus: first-principles syntax and local metatheory, checked without additional axioms, supporting the self-compilation layer while leaving full CIC metatheory outside the claimed surface.

9235.15 Closed-Term Metatheory Completion

The closed-term surface of `BEDC.MetaCIC` is the part of the self-compilation substrate in which represented terms are read in the empty context and de Bruijn closure is an explicit hypothesis. The object calculus is the universe-0, beta-only, single-sort dependent-function fragment: represented syntax has sort, variable, application, lambda, and Pi; typing is an inductive judgment over represented contexts; beta reduction is an inductive relation over represented terms; and closure is the predicate `ClosedAt`. The proof objects live inside Lean 4 without mathlib, without project axioms, and without sorry placeholders.

The scope claim is deliberately narrow and binding. BEDC does not assert a full host-kernel theorem about Lean's own CIC, and it does not assert the full open-term metatheory of a production proof assistant. It asserts that the closed fragment needed by the self-compilation surface is represented from first principles, that the reduction and closure invariants needed by that surface are checked, and that the remaining theorem boundaries are exposed as named Lean statements rather than hidden behind prose.

9235.15.1 The Closed-Term Scope Claim

The represented calculus has a closed empty-context reading:

$$\mathrm{ClosedTermScope}(t, A) \quad \equiv \quad \texttt{ClosedAt}\ 0\ t \wedge \texttt{HasType}\ []\ t\ A.$$

Inside this reading, the three metatheoretic invariants used by BEDC's self-compilation account have checked representatives.

First, subject reduction is available as a hypothesis-explicit theorem over the represented typing and beta-step relations. The theorem states that beta substitution preservation, application-argument type stability, lambda-domain subject reduction, and Pi-domain subject reduction are exactly the interfaces needed to preserve the displayed type along one represented beta step. The ordinary beta case and the congruence cases are mechanized; the closed-scope discharge layer records which of the remaining interfaces is valid under the closed hypotheses and where the calculus rejects a stronger reading.

```
↪ variant BEDC.MetaCIC.subject reduction
↪ variant BEDC.MetaCIC.subject reduction beta
↪ variant BEDC.MetaCIC.subject reduction congApp1
↪ variant BEDC.MetaCIC.subject reduction congApp2
↪ variant BEDC.MetaCIC.subject reduction congLam
↪ variant BEDC.MetaCIC.subject reduction congPi
↪ variant BEDC.MetaCIC.subject reduction closed app independent codomain
```

Second, closure preservation is fully checked for one-step and finite-chain beta reduction. If a represented term is closed at boundary n, one `BetaStep` preserves closure at the same boundary. The finite-chain theorem lifts this to `BetaStarStep`. This is the core invariant that keeps the object-language evaluator inside the intended de Bruijn boundary: beta contraction, application

congruence, lambda congruence, and Pi congruence cannot create a free represented variable when the source term is closed.

```
↪variant BEDC.MetaCIC.betaStep preserves closed
↪variant BEDC.MetaCIC.betaStarStep preserves closed
↪variant BEDC.MetaCIC.substitute preserves closed at
```

Third, strong-normalization witnesses exist for the closed normal-form sub-scope used by the gallery and by the self-compilation reading. The normalization layer defines `StrongNormalizable`, beta-inert sources, candidates, closed candidates, Pi candidates, neutral normal forms, and closure of those witnesses under the relevant one-step reductions. It proves that sort and variables are strongly normalizable, that neutral applications are strongly normalizable when their parts are, that lambda and Pi preserve strong-normalization witnesses componentwise, and that the closed Church and polymorphic gallery terms carry concrete witnesses. This is not a theorem that every well-typed represented term normalizes; it is the checked normal-form and candidate sub-scope actually consumed by BEDC.

```
↪variant BEDC.MetaCIC.StrongNormalizable
↪variant BEDC.MetaCIC.sort is sn
↪variant BEDC.MetaCIC.var is sn
↪variant BEDC.MetaCIC.closed atom normal form
↪variant BEDC.MetaCIC.closed sort candidate
↪variant BEDC.MetaCIC.closedCandidate implies sn
↪variant BEDC.MetaCIC.PiAdequacy.piCandidate implies sn
↪variant BEDC.MetaCIC.churchTrueTm sn
↪variant BEDC.MetaCIC.churchFalseTm sn
↪variant BEDC.MetaCIC.churchZeroTm sn
↪variant BEDC.MetaCIC.churchOneTm sn
```

The bounded normalizer is the operational layer sitting next to these invariants. It is a fuel-indexed function on represented terms. Its equational laws connect concrete represented reductions with host CIC computation for the finite gallery. Its termination is structural recursion on fuel, while its metatheoretic safety is supplied by the closure-preservation theorems over represented beta traces.

```
↪variant BEDC.MetaCIC.normalizeBounded
↪variant BEDC.HostBridge.normalize id app sort
↪variant BEDC.HostBridge.normalize poly identity apply
↪variant BEDC.HostBridge.normalize church true apply
↪variant BEDC.HostBridge.normalize church zero apply
↪variant BEDC.HostBridge.id app sort beta star
↪variant BEDC.HostBridge.poly identity chain
```

9235.15.2 What This Scope Buys

The first payoff is a first-principles self-compilation boundary. The represented typed witnesses used by BEDC are not informal quotations of host Lean terms. They are object-language inhabitants built from the MetaCIC syntax constructors, checked by the MetaCIC typing relation, and reduced by the MetaCIC beta relation. When such a witness is closed and lies in the normal-form sub-scope, the project has both a termination witness and a type-preservation path appropriate to the exact local interface it uses.

The phrase "appropriate to the exact local interface" is important. The calculus validates broad subject reduction only under its named hypotheses. For applications with a closed independent codomain, the application-argument stability hypothesis is discharged directly. For reducible binder domains, Lean records counterexamples showing that the stronger lambda-domain and Pi-domain closed statements are not valid as stated. This is a positive metatheoretic result for BEDC's architecture: the paper does not need to guess where the closed scope ends, because the endpoint is itself mechanized.

```
↪ variant BEDC.MetaCIC.appArgTypeStable closed
↪ variant BEDC.MetaCIC.appArgTypeStable closed independent codomain
↪ variant BEDC.MetaCIC.lamDomainSR closed counterexample
↪ variant BEDC.MetaCIC.piDomainSR closed counterexample
↪ variant BEDC.MetaCIC.not subject reduction closed
```

The second payoff is representational adequacy for the finite gallery. The host bridge pairs ordinary host-level functions and Church encodings with represented MetaCIC terms, represented typing witnesses, and represented beta chains. The bridge is not a theorem that every host term has a represented image. It is a checked transport gallery for the exact functions used in the self-compilation account: identity, constant, flip, compose, Church Booleans, Church numerals, pairs, options, and selected beta-redex shapes.

```
↪ variant BEDC.HostBridge.MetaCICTransport.transport identity well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport const well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport flip well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport compose well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church true well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church false well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church zero well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church one well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport poly id two step redex star
```

The third payoff is executable normal-form operation without importing standard-library axioms. The bounded normalizer is a native recursive operation over the represented syntax. It decides the normal form of a finite run by host fuel and definitional equations, not by an excluded-middle search over arbitrary reduction paths. For the gallery, the equational laws state that enough fuel reaches the expected representative. For the paper's closed-term self-compilation surface, this is exactly the executable content needed: a finite evaluator, named beta traces, closure preservation for those traces, and concrete normal-form witnesses where termination is claimed.

9235.15.3 The Three-Hypothesis Chain

The subject-reduction theorem exposes three non-beta interfaces after beta substitution preservation. In the closed empty-context scope they have different resolutions. The application-argument interface is discharged for closed independent codomains; the reducible lambda-domain and Pi-domain interfaces are resolved by checked counterexamples. The following table is the binding reading of that chain.

Interface	Closed-scope resolution	Lean targets
`AppArgTypeStable`	Valid when the Pi codomain is closed at boundary 0, so substituting either the source argument or its reduct into that codomain returns the codomain itself. This is the application case BEDC uses for closed independent codomain witnesses.	↪variant `BEDC.MetaCIC.appArgTypeStable closed` ↪variant `BEDC.MetaCIC.subject reduction closed app independent codomain`
`LamDomainSubjectReduction`	Not valid as a blanket closed-term theorem. A closed lambda whose domain itself reduces can lose the displayed Pi type when the domain is rewritten without simultaneously transporting the codomain. The counterexample fixes the precise boundary.	↪variant `BEDC.MetaCIC.lamDomainSR closed counterexample` ↪variant `BEDC.MetaCIC.not lamDomainSR closed` ↪variant `BEDC.MetaCIC.not subject reduction closed lam`
`PiDomainSubjectReduction`	Not valid as a blanket closed-term theorem. A closed Pi source may have a codomain typed in the old shifted-domain context; reducing the domain changes the binder context, and the codomain need not retype under the new one.	↪variant `BEDC.MetaCIC.piDomainSR closed counterexample` ↪variant `BEDC.MetaCIC.not piDomainSR closed` ↪variant `BEDC.MetaCIC.not subject reduction closed pi`

This table is why the closed-term claim is stated as a scoped metatheory completion rather than as a hidden full-kernel preservation theorem. The application row supplies the positive discharge needed by the closed independent-codomain surface. The two domain rows prevent BEDC from treating closedness alone as a substitute for binder-context transport. The resulting architecture is stricter than a prose-only closure claim: every consumed interface is named, and every rejected strengthening is named as well.

9235.15.4 Closed Normalization and Candidate Witnesses

The normal-form sub-scope has two layers. The first layer is beta-strong inertia. A term is `BetaStrong` when it has no beta successor. Sort and variables satisfy this immediately by constructor inversion on `BetaStep`; `BetaStrong` implies `StrongNormalizable`; hence sort and variables are strongly normalizable. The closed-atom theorem packages the same fact for closed terms typed at sort when their shape is sort or variable.

↪variant `BEDC.MetaCIC.BetaStrong`
↪variant `BEDC.MetaCIC.betaStrong implies sn`
↪variant `BEDC.MetaCIC.sort betaStrong`
↪variant `BEDC.MetaCIC.var betaStrong`
↪variant `BEDC.MetaCIC.sort no betaStrong successor`
↪variant `BEDC.MetaCIC.var no betaStrong successor`

The second layer is candidate-based adequacy for the Pi and neutral-normal shapes used by the gallery. `Candidate` combines a `StrongNormalizable` witness with a type-head witness.

`PiCandidate` adds the application clause required for dependent functions. `ClosedCandidate` adds closure of both the type and the term. These objects are closed under one beta step when the target closure is available, and they imply strong normalization of the represented term.

```
↪ variant BEDC.MetaCIC.Candidate
↪ variant BEDC.MetaCIC.ClosedCandidate
↪ variant BEDC.MetaCIC.PiCandidate
↪ variant BEDC.MetaCIC.candidate closed under beta
↪ variant BEDC.MetaCIC.closedCandidate under beta with closed target
↪ variant BEDC.MetaCIC.pi candidate app
↪ variant BEDC.MetaCIC.piCandidate head sn
```

Neutral normal forms give the gallery its closed application witnesses. Variables are neutral. An application of a neutral function to a strongly normalizable argument is neutral. Neutral normal forms are stable under beta steps, and every neutral normal form is strongly normalizable. Lambda and Pi constructors preserve strong-normalization witnesses componentwise. The Church gallery then instantiates this small calculus of witnesses for the particular closed shapes used in the paper.

```
↪ variant BEDC.MetaCIC.NeutralNormal
↪ variant BEDC.MetaCIC.neutralNormal step
↪ variant BEDC.MetaCIC.neutralNormal implies sn
↪ variant BEDC.MetaCIC.neutralApp sn
↪ variant BEDC.MetaCIC.sn lam
↪ variant BEDC.MetaCIC.sn pi
↪ variant BEDC.MetaCIC.churchBoolTy sn
↪ variant BEDC.MetaCIC.churchNatTy sn
↪ variant BEDC.MetaCIC.churchSuccTm sn
```

This normalization layer is enough for the self-reflective use of the MetaCIC substrate. BEDC's self-compilation examples require finite typed witnesses, explicit beta traces, closedness preservation, and strong-normal witnesses for the closed gallery terms. They do not require a theorem that all admissible represented terms normalize under every possible reduction strategy.

9235.15.5 Bounded Normalizer as Operation

The bounded normalizer is executable because it is structurally recursive on fuel:

$$\begin{aligned}\texttt{normalizeBounded}\ 0\ t &= t,\\ \texttt{normalizeBounded}\ (k+1)\ \texttt{sort} &= \texttt{sort},\\ \texttt{normalizeBounded}\ (k+1)\ (\texttt{var}\ i) &= \texttt{var}\ i.\end{aligned}$$

For lambda and Pi it recursively normalizes components. For applications it normalizes the function and argument and contracts when the normalized function is a lambda. The operation is therefore deterministic as host computation, while the object-language safety statement remains the represented one: whenever the run is read as a finite `BetaStep` trace, `betaStarStep_preserves_closed` preserves closure from source to target.

The host bridge records that the same finite computations match host CIC equalities for the gallery. The left side is represented computation through `normalizeBounded`; the right side is a host definitional equality. The beta-chain lemmas give the object-language reduction witnesses for the same examples. Together they supply an operational bridge from host examples to represented examples without importing host normalization as an axiom about the represented calculus.

```
↪ variant BEDC.HostBridge.host id app sort
↪ variant BEDC.HostBridge.host poly identity apply
↪ variant BEDC.HostBridge.host church true apply
↪ variant BEDC.HostBridge.host church zero apply
↪ variant BEDC.HostBridge.church true apply chain
↪ variant BEDC.HostBridge.church false apply chain
↪ variant BEDC.HostBridge.church true apply step two
↪ variant BEDC.HostBridge.church false apply step two
```

9235.15.6 What Remains Outside the Closed-Term Scope

The frontier above the closed-term scope is explicit.

- Open-term subject reduction remains outside the checked claim. The obstruction is not merely bookkeeping: substitution composition under binders must account for shifting, local variables, and binder-context transport simultaneously.
- Full beta-parallel diamond over open terms remains outside the checked claim. Atom diamonds and compound-shape assemblers are mechanized, but the open application residual proof still depends on substitution compatibility across parallel residuals.
- Universe hierarchy remains outside the represented calculus. The object language has one sort, so it does not claim cumulativity, universe polymorphism, or universe-level consistency for a hierarchy.
- Inductive families remain outside the represented calculus. There are no represented constructors, eliminators, positivity conditions, recursion principles, or computation rules for inductive families.
- Proposition irrelevance remains outside the represented calculus. The fragment has typing, beta, Pi, lambda, application, variables, and sort; it does not include a proposition universe or proof-irrelevance principle.

These boundaries are architectural commitments. The closed-term MetaCIC surface is a substrate for BEDC's self-compilation account, not a replacement kernel for all of Lean and not a paper claim about every feature of CIC. The frontier is stated so that downstream BEDC arguments do not accidentally use universe hierarchy, inductive-family computation, proposition irrelevance, or open-term substitution algebra as if those facts were already part of the first-principles substrate.

9235.15.7 Meaning for the BEDC Architecture

`BEDC.MetaCIC` is the self-compilation substrate of the manuscript. It gives BEDC a represented language in which typed witnesses, reduction traces, normal-form evidence, and finite host-gallery transport can be discussed inside the project rather than only in metalinguistic prose. The closed-term scope covers the surface the self-reflective architecture truly consumes: closed represented terms, empty-context typing witnesses, beta-only finite reduction traces, candidate and neutral-normal witnesses for the gallery, and bounded normalizer equations for executable examples.

The finite-history bridge fits this reading. A `BHistIdx` can be embedded as a represented MetaCIC term, and the image is closed at boundary 0. This gives the self-compilation substrate a direct route back to BEDC's finite event-history world without opening the de Bruijn boundary.

```
↪ variant BEDC.MetaCIC.BHistSubstrate.bhistToTerm closed
```

↪ variant BEDC.MetaCIC.BHistSubstrate.bhistToTerm injective

Outside the closed-term scope, BEDC makes no first-principles claim. An argument that needs open-term subject reduction, a full parallel diamond, universe hierarchy, inductive families, or proposition irrelevance must carry that extra structure explicitly. The closed surface is therefore strong because it is exact: it proves the substrate BEDC actually uses, records the normal-form and operation facts needed by that substrate, and leaves the larger CIC frontier outside the claim.

9235.16 Session-Level Metatheory Claim

This section states the claim surface carried by `BEDC.MetaCIC` as a single formal object of the manuscript. The surface is intentionally narrow: it concerns the universe-0, beta-only, single-sort dependent-function fragment, read at closed represented terms and empty object contexts. Within that scope, the manuscript can cite a mathlib-free, axiom-free, sorry-free Lean 4 mechanization of the represented syntax, typing, beta reduction, closure discipline, selected confluence components, closed normalization witnesses, executable checking, host-gallery transport, Church-encoded data laws, and beta-conversion facts.

Scope claim. In the universe-0, beta-only, single-sort fragment with represented constructors `sort`, `var`, `app`, `lam`, and `pi`, and under the reading

$$\mathrm{ClosedEmptyCtx}(t, A) \quad \equiv \quad \texttt{ClosedAt}\ 0\ t \wedge \texttt{HasType}\ []\ t\ A,$$

`BEDC.MetaCIC` supplies first-principles mechanized metatheory inside Lean 4, without mathlib imports, without project axioms, and without sorry placeholders. The claim is about this represented object calculus, not about the full host Lean kernel.

9235.16.1 Claim 1: Substrate Invariants

The substrate invariant is that de Bruijn indices are not treated as informal names. The calculus defines shift and substitution as native recursive operations on represented terms. A descent under a binder changes the cutoff; substitution below a binder shifts the inserted term; closure is the predicate that records which indices are admissible at a boundary. The following targets are the basic shift-aware substrate facts used throughout the metatheory:

↪ variant BEDC.MetaCIC.shift closed
↪ variant BEDC.MetaCIC.substitute closed
↪ variant BEDC.MetaCIC.closedAt shift
↪ variant BEDC.MetaCIC.closedAt substitute
↪ variant BEDC.MetaCIC.closedAt succ
↪ variant BEDC.MetaCIC.closedAt zero at
↪ variant BEDC.MetaCIC.shift substitute at closed zero
↪ variant BEDC.MetaCIC.shift substitute zero zero closed

The typing substrate is correspondingly local. Substitution preservation is not asserted by a global slogan; it is exposed through first-principles subderivation theorems for the relevant shapes. The Pi, lambda, and application theorems say exactly which represented premises are sufficient to preserve typing across the local substitution operation:

↪ variant BEDC.MetaCIC.substitute preserves typing pi if subderivations
↪ variant BEDC.MetaCIC.substitute preserves typing lam if subderivations
↪ variant BEDC.MetaCIC.substitute preserves typing app if subderivations

```
↪ variant BEDC.MetaCIC.substitute preserves typing pi
↪ variant BEDC.MetaCIC.substitute preserves typing lam
↪ variant BEDC.MetaCIC.substitute preserves typing app cod sort var commutes
↪ variant BEDC.MetaCIC.substitute preserves typing via term induction
```

Closure is stable under represented beta computation. A one-step beta reduction from a term closed at boundary n has a target closed at the same boundary, and the reflexive-transitive beta chain preserves the same boundary. This is the guardrail that permits the self-compilation surface to execute finite represented beta traces without leaving the closed fragment:

```
↪ variant BEDC.MetaCIC.substitute preserves closed at
↪ variant BEDC.MetaCIC.betaStep preserves closed
↪ variant BEDC.MetaCIC.betaStarStep preserves closed
↪ variant BEDC.MetaCIC.betaParallel preserves closed
↪ variant BEDC.MetaCIC.betaParallel preserves closed zero
```

Read together, these targets justify the substrate part of the claim: the syntax, closure, substitution, and local typing algebra are not external assumptions. They are object-language theorems about the represented calculus, and all later claims depend on that explicit de Bruijn discipline.

9235.16.2 Claim 2: Subject Reduction in the Closed Scope

The central subject-reduction theorem has an explicit interface shape. It assumes the beta-substitution interface and three non-beta stability interfaces, then proves preservation of the displayed type along one represented beta step. The beta case and the congruence cases are checked directly over the represented typing and beta-step relations:

```
↪ variant BEDC.MetaCIC.subject reduction beta
↪ variant BEDC.MetaCIC.subject reduction congApp1
↪ variant BEDC.MetaCIC.subject reduction congApp2
↪ variant BEDC.MetaCIC.subject reduction congLam
↪ variant BEDC.MetaCIC.subject reduction congPi
↪ variant BEDC.MetaCIC.subject reduction
↪ variant BEDC.MetaCIC.subject reduction retype
```

Inside the closed empty-context reading, the application-argument stability interface has a positive closed use case. When the Pi codomain is already closed at boundary 0, substituting either the source argument or its beta reduct into that codomain returns the same codomain. This gives the closed independent-codomain application preservation theorem consumed by the self-compilation examples:

```
↪ variant BEDC.MetaCIC.appArgTypeStable closed independent codomain
↪ variant BEDC.MetaCIC.appArgTypeStable closed
↪ variant BEDC.MetaCIC.subject reduction closed app independent codomain
```

The closed scope also records the exact negative boundary for stronger blanket readings. A closed lambda may reduce in its displayed domain while the target no longer has the displayed Pi type. A closed Pi may reduce in its domain while its codomain was typed in the old shifted-domain context. Therefore closedness alone does not discharge the lambda-domain and Pi-domain interfaces as global theorems:

```
↪ variant BEDC.MetaCIC.lamDomainSR closed counterexample
↪ variant BEDC.MetaCIC.not lamDomainSR closed
↪ variant BEDC.MetaCIC.not subject reduction closed lam
↪ variant BEDC.MetaCIC.piDomainSR closed counterexample
↪ variant BEDC.MetaCIC.not piDomainSR closed
```

```
↪ variant BEDC.MetaCIC.not subject reduction closed pi
↪ variant BEDC.MetaCIC.not subject reduction closed
```

Thus the subject-reduction claim is exact. BEDC has a checked preservation theorem with explicit hypotheses, a checked positive closed application sub-scope, and checked counterexamples to the stronger closed binder-domain reading. The manuscript must not cite a blanket `subject_reduction_closed` theorem, because the represented calculus contains formal witnesses refuting that statement.

9235.16.3 Claim 3: Confluence in the Closed Scope

The confluence surface is split into atom facts, shape assemblers, and a closed-normal sub-scope. Atoms have trivial forks: a sort source and a variable source cannot produce a genuine beta fork. The beta-star and parallel-beta diamonds for atom sources therefore join at the source itself:

```
↪ variant BEDC.MetaCIC.betaStarStep diamond sort
↪ variant BEDC.MetaCIC.betaStarStep diamond var
↪ variant BEDC.MetaCIC.betaParallel sort diamond
↪ variant BEDC.MetaCIC.betaParallel var diamond
↪ variant BEDC.MetaCIC.betaParallel diamond sort
↪ variant BEDC.MetaCIC.betaParallel diamond var
```

For compound terms, the parallel relation exposes the expected structural components. Application, lambda, and Pi diamonds can be assembled from component diamonds. Shape-specific lambda and Pi diamonds are checked. Application forks are handled by an inversion theorem and by a conditional application-shape theorem whose remaining premise is the explicit substitution-compatibility property:

```
↪ variant BEDC.MetaCIC.betaParallel app diamond of components
↪ variant BEDC.MetaCIC.betaParallel lam diamond of components
↪ variant BEDC.MetaCIC.betaParallel pi diamond of components
↪ variant BEDC.MetaCIC.betaParallel lam shape diamond
↪ variant BEDC.MetaCIC.betaParallel pi shape diamond
↪ variant BEDC.MetaCIC.betaParallel app full inv
↪ variant BEDC.MetaCIC.betaParallel app cong cong diamond
↪ variant BEDC.MetaCIC.betaParallel app shape diamond from substitution
↪ variant BEDC.MetaCIC.betaParallel diamond from substitution
```

The unconditional closed part is available for normal sources. If a closed source is beta-normal, any parallel target or beta-star target is forced back to the source, so both forks join immediately. Sort is a closed normal source; a closed variable source is impossible at boundary 0, and the corresponding impossible cases are also named:

```
↪ variant BEDC.MetaCIC.BetaNormal
↪ variant BEDC.MetaCIC.betaNormal sort
↪ variant BEDC.MetaCIC.betaNormal var
↪ variant BEDC.MetaCIC.betaParallel diamond closed normal
↪ variant BEDC.MetaCIC.betaStarStep confluence closed normal
↪ variant BEDC.MetaCIC.betaParallel diamond closed sort
↪ variant BEDC.MetaCIC.betaStarStep confluence closed sort
↪ variant BEDC.MetaCIC.betaParallel diamond closed var impossible
↪ variant BEDC.MetaCIC.betaStarStep confluence closed var impossible
```

The Church-Rosser theorem for the whole closed fragment is therefore not read as an unqualified theorem here. What is checked is the atom diamond surface, the compound-shape machinery, the conditional full parallel diamond from substitution compatibility, and the unconditional closed-normal confluence surface. The missing global claim would require a closed substitution compatibility theorem strong enough to cover application residuals.

9235.16.4 Claim 4: Strong Normalization in the Closed Scope

The normalization layer defines the reducibility-candidate vocabulary used by the closed fragment. `StrongNormalizable` is an inductive predicate closed under one-step tails. Candidate, closed candidate, Pi candidate, and candidate-set structures package strong-normalization witnesses together with the type-head and closure facts needed by the represented calculus:

```
↪ variant BEDC.MetaCIC.StrongNormalizable
↪ variant BEDC.MetaCIC.strongNormalizable step
↪ variant BEDC.MetaCIC.strongNormalizable reduces once
↪ variant BEDC.MetaCIC.Candidate
↪ variant BEDC.MetaCIC.ClosedCandidate
↪ variant BEDC.MetaCIC.PiCandidate
↪ variant BEDC.MetaCIC.CandidateSet
↪ variant BEDC.MetaCIC.candidate implies sn
↪ variant BEDC.MetaCIC.closedCandidate implies sn
↪ variant BEDC.MetaCIC.piCandidate implies sn
↪ variant BEDC.MetaCIC.candidate closed under beta
↪ variant BEDC.MetaCIC.closedCandidate under beta with closed target
```

The Pi adequacy component says that a Pi candidate gives the expected application candidate and application strong-normalization witness. Neutral normal forms supply the closed application witnesses used in the gallery, and lambda and Pi constructors preserve strong-normalization from their parts:

```
↪ variant BEDC.MetaCIC.pi candidate app
↪ variant BEDC.MetaCIC.piCandidate head sn
↪ variant BEDC.MetaCIC.PiAdequacy.piCandidate implies sn
↪ variant BEDC.MetaCIC.NeutralNormal
↪ variant BEDC.MetaCIC.neutralNormal step
↪ variant BEDC.MetaCIC.neutralNormal implies sn
↪ variant BEDC.MetaCIC.neutralApp sn
↪ variant BEDC.MetaCIC.sn lam
↪ variant BEDC.MetaCIC.sn pi
```

The closed-typed-term layer turns those witnesses into theorem statements about empty-context terms. It does not assert that every closed typed term has a candidate. It asserts the exact theorem available from each of the candidate structures: if the candidate evidence is present, then the closed typed term is strongly normalizable and admits no infinite beta-reduction chain:

```
↪ variant BEDC.MetaCIC.ClosedTypedTerm
↪ variant BEDC.MetaCIC.ClosedTypedCandidate
↪ variant BEDC.MetaCIC.ClosedTypedPiCandidate
↪ variant BEDC.MetaCIC.closedTypedCandidate implies sn
↪ variant BEDC.MetaCIC.closedTypedPiCandidate implies sn
↪ variant BEDC.MetaCIC.closedTypedPiCandidate app sn
↪ variant BEDC.MetaCIC.closed typed term sn from candidate
```

```
↪ variant BEDC.MetaCIC.closed typed term sn from closedCandidate
↪ variant BEDC.MetaCIC.closed typed term sn from candidateSet
↪ variant BEDC.MetaCIC.strongNormalizable no infinite reduction
↪ variant BEDC.MetaCIC.closedTypedCandidate no infinite reduction
```

The gallery supplies concrete closed witnesses for the sort, Church Boolean, Church natural, identity, constant, and successor shapes used by the self-compilation account:

```
↪ variant BEDC.MetaCIC.closed sort sn
↪ variant BEDC.MetaCIC.closed lam sn from parts
↪ variant BEDC.MetaCIC.closed pi sn from parts
↪ variant BEDC.MetaCIC.closed neutral sn
↪ variant BEDC.MetaCIC.churchBoolTy sn
↪ variant BEDC.MetaCIC.churchNatTy sn
↪ variant BEDC.MetaCIC.churchIdentityTm sn
↪ variant BEDC.MetaCIC.churchIdentityTy sn
↪ variant BEDC.MetaCIC.churchTrueTm sn
↪ variant BEDC.MetaCIC.churchFalseTm sn
↪ variant BEDC.MetaCIC.churchZeroTm sn
↪ variant BEDC.MetaCIC.churchOneTm sn
↪ variant BEDC.MetaCIC.churchConstTy sn
↪ variant BEDC.MetaCIC.churchConstTm sn
↪ variant BEDC.MetaCIC.churchSuccTy sn
↪ variant BEDC.MetaCIC.churchSuccTm sn
```

This is the strong-normalization claim carried by the manuscript. The theorems are genuine Lean theorems with no axioms and no placeholders, but their quantifiers are candidate-scoped or gallery-scoped. A global theorem that every closed typed represented term has a reducibility candidate is not part of this claim.

9235.16.5 Claim 5: Decidable Type Checker

The executable checker is the function `inferTypeCtx`. It computes an optional represented type from a represented context and term, using native recursive equality checks and the same syntax and context lookup as the typing relation. Its closed version is the empty-context specialization:

```
↪ variant BEDC.MetaCIC.inferTypeCtx
↪ variant BEDC.MetaCIC.inferTypeCtxClosed
↪ variant BEDC.MetaCIC.inferTypeCtx eq inferType
```

Soundness states that a successful checker result gives a `HasType` witness. Completeness states that a represented typing derivation is found by the checker. The shape-specific completeness theorems expose the sort, variable, Pi, lambda, and application cases separately:

```
↪ variant BEDC.MetaCIC.inferTypeCtx sound
↪ variant BEDC.MetaCIC.inferTypeCtx complete
↪ variant BEDC.MetaCIC.inferTypeCtxClosed sound
↪ variant BEDC.MetaCIC.inferTypeCtxClosed complete
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete raw
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete total
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete sort
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete var
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete pi
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete lam
```

```
↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete app
↪ variant BEDC.MetaCIC.inferTypeCtx complete strict
```

The lookup-based closed-atom layer records the boundary behavior at empty contexts. Sort checks as sort, variables are rejected in the empty context, and the inversion lemmas expose how a typed atom or compound term must have arisen:

```
↪ variant BEDC.MetaCIC.inferTypeCtx complete sort
↪ variant BEDC.MetaCIC.inferTypeCtxClosed var empty none
↪ variant BEDC.MetaCIC.hasType sort empty ctx
↪ variant BEDC.MetaCIC.hasType var ctx lookup
↪ variant BEDC.MetaCIC.hasType var empty ctx absurd
↪ variant BEDC.MetaCIC.hasType app empty ctx pi
↪ variant BEDC.MetaCIC.hasType lam empty ctx pi
↪ variant BEDC.MetaCIC.hasType pi empty ctx
```

The checker claim is therefore an executable metatheory claim. It is not a prose assertion that type checking is decidable in some external calculus; it is a concrete function together with soundness, completeness, and closed empty-context atom behavior over the represented calculus.

9235.16.6 Claim 6: Self-Representability via HostBridge

The host bridge relates ordinary host CIC examples to represented MetaCIC terms by a finite transport gallery. On the host side, identity, constant, flip, compose, Church Booleans, Church naturals, pairs, options, and selected redexes compute by reflexive equalities. These equalities are not imported as axioms about the represented calculus; they are paired with represented typing and reduction witnesses:

```
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host identity beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host const beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host flip beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host compose beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church true beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church false beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church zero beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church one beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host pair fst beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host pair snd beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host none beta
↪ variant BEDC.HostBridge.MetaCICTransport.Host.host some beta
```

The transport layer gives the represented typing side of the same gallery. Each transport theorem is an empty-context `HasType` witness over the MetaCIC syntax:

```
↪ variant BEDC.HostBridge.MetaCICTransport.transport identity well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport const well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport flip well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport compose well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church bool type well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church true well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church false well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church nat type well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church zero well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church one well typed
```

```
↪ variant BEDC.HostBridge.MetaCICTransport.transport church succ well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church pair type well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church mk pair well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church option type well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church none well typed
↪ variant BEDC.HostBridge.MetaCICTransport.transport church some well typed
```

The bounded normalizer gives an operational companion to the transport gallery. It is a fuel-indexed recursive function over represented terms. Concrete equations state that sufficient fuel reaches the expected normal forms for the finite examples, and separate beta-chain theorems give the object-language reduction traces:

```
↪ variant BEDC.MetaCIC.normalizeBounded
↪ variant BEDC.HostBridge.normalize id app sort
↪ variant BEDC.HostBridge.normalize poly identity apply
↪ variant BEDC.HostBridge.normalize church true apply
↪ variant BEDC.HostBridge.normalize church false apply
↪ variant BEDC.HostBridge.normalize church zero apply
↪ variant BEDC.HostBridge.normalize church one apply
↪ variant BEDC.HostBridge.host id app sort
↪ variant BEDC.HostBridge.host poly identity apply
↪ variant BEDC.HostBridge.host church true apply
↪ variant BEDC.HostBridge.host church zero apply
↪ variant BEDC.HostBridge.id app sort beta star
↪ variant BEDC.HostBridge.poly identity chain
↪ variant BEDC.HostBridge.church true apply chain
↪ variant BEDC.HostBridge.church false apply chain
```

This is a self-representability claim for a finite gallery, not a theorem that arbitrary host Lean terms elaborate into MetaCIC terms. The bridge is exact because each transported example is named on both sides.

9235.16.7 Claim 7: Church-Encoded Data Structure Laws

The Church gallery is a typed object-language gallery. Booleans, naturals, pairs, options, and polymorphic function forms are represented as MetaCIC terms and typed in the empty context:

```
↪ variant BEDC.MetaCIC.church bool type
↪ variant BEDC.MetaCIC.church true
↪ variant BEDC.MetaCIC.church false
↪ variant BEDC.MetaCIC.church nat type
↪ variant BEDC.MetaCIC.church zero
↪ variant BEDC.MetaCIC.church one
↪ variant BEDC.MetaCIC.church succ type
↪ variant BEDC.MetaCIC.church succ
↪ variant BEDC.MetaCIC.church pair type
↪ variant BEDC.MetaCIC.church mk pair
↪ variant BEDC.MetaCIC.church fst
↪ variant BEDC.MetaCIC.church snd
↪ variant BEDC.MetaCIC.church option type
↪ variant BEDC.MetaCIC.church none
↪ variant BEDC.MetaCIC.church some
```

```
↪ variant BEDC.MetaCIC.church case option
↪ variant BEDC.MetaCIC.poly identity type
↪ variant BEDC.MetaCIC.poly const type
↪ variant BEDC.MetaCIC.poly flip type
↪ variant BEDC.MetaCIC.poly compose type
```

The algebraic laws are represented beta-star equalities, not host-side rewrite folklore. Boolean projections select the expected branch, Church zero and one apply as expected, polymorphic identity returns its argument, and pair elimination sends a Church pair to the selector applied to its two payloads:

```
↪ variant BEDC.MetaCIC.church true proj
↪ variant BEDC.MetaCIC.church false proj
↪ variant BEDC.MetaCIC.church zero apply
↪ variant BEDC.MetaCIC.church one apply
↪ variant BEDC.MetaCIC.poly identity apply
↪ variant BEDC.MetaCIC.churchPairPayload normalize
↪ variant BEDC.MetaCIC.church pair value to selector
```

The redex gallery connects the same algebraic surface to explicit `BetaStep` and `BetaStarStep` witnesses. These targets are the closed concrete traces used by the bounded normalizer and by the transport bridge:

```
↪ variant BEDC.MetaCIC.id sort redex
↪ variant BEDC.MetaCIC.id sort redex typed
↪ variant BEDC.MetaCIC.fn sort to sort redex
↪ variant BEDC.MetaCIC.fn sort to sort redex typed
↪ variant BEDC.MetaCIC.const sort redex
↪ variant BEDC.MetaCIC.const sort redex typed
↪ variant BEDC.MetaCIC.poly id two step redex
↪ variant BEDC.MetaCIC.poly id two step redex typed
↪ variant BEDC.MetaCIC.k sort two step redex
↪ variant BEDC.MetaCIC.k sort two step redex typed
```

The Church-encoded data claim is therefore a mechanized family of typed examples and beta-star laws for the represented calculus. It covers the finite data structures used in the self-compilation surface; it does not assert native inductive families in the represented object language.

9235.16.8 Claim 8: BetaConv Relation Properties

`BetaConv` is the joinability relation induced by `BetaStarStep`. Reflexivity, symmetry, inclusion of beta-star and one-step beta on either side, and conditional transitivity under a confluence input are all checked:

```
↪ variant BEDC.MetaCIC.BetaConv
↪ variant BEDC.MetaCIC.BetaConv.refl
↪ variant BEDC.MetaCIC.BetaConv.symm
↪ variant BEDC.MetaCIC.BetaConv.of betaStar left
↪ variant BEDC.MetaCIC.BetaConv.of betaStar right
↪ variant BEDC.MetaCIC.BetaConv.of betaStep left
↪ variant BEDC.MetaCIC.BetaConv.of betaStep right
↪ variant BEDC.MetaCIC.BetaConv.trans of confluence
```

There is also a closed-normal transitivity route. If the middle term is beta-strong, both conversion witnesses must meet at that normal middle, so conversion composes through it. Closed normal forms related by `BetaConv` are equal:

↪ variant `BEDC.MetaCIC.betaNormal of betaStrong`

↪ variant `BEDC.MetaCIC.BetaConv.trans normal middle`

↪ variant `BEDC.MetaCIC.BetaConv.trans closed normal middle`

↪ variant `BEDC.MetaCIC.BetaConv normal forms equal closed`

The important negative fact is also checked: empty-context typing is not invariant under `BetaConv` in full generality. The obstruction is a lambda-domain type example. The source and target are closed and beta-convertible, the source has the displayed type, and the target does not:

↪ variant `BEDC.MetaCIC.HasTypeBetaConvInvariantClosedStatement`

↪ variant `BEDC.MetaCIC.closedLamDomainTarget closed`

↪ variant `BEDC.MetaCIC.BetaConv closed lam domain type obstruction`

↪ variant `BEDC.MetaCIC.not HasType BetaConv invariant closed`

Consequently, the manuscript can use `BetaConv` as a checked joinability relation with equivalence-like behavior in the confluence or closed-normal sub-scope. It cannot use `BetaConv` as a blanket typing conversion rule for `HasType` over the closed empty-context fragment.

9235.16.9 Boundary Statement

The scope boundary is part of the claim. Open-term metatheory is outside the closed empty-context surface. The obstruction is concrete: the naive substitution-composition shape fails at depth 0 with $v = \texttt{var}\ 0$, $u = \texttt{sort}$, and $t = \texttt{var}\ 1$. The corresponding Lean target records the inequality directly, and the confluence layer exposes the same obstruction as a substitution-commutation boundary:

↪ variant `BEDC.MetaCIC.substitute substitute zero zero unclosed counterexample`

↪ variant `BEDC.MetaCIC.substitute commutation unclosed obstruction`

↪ variant `BEDC.MetaCIC.BetaParallelSubstitutionCompatibility`

The represented calculus also excludes a universe hierarchy, cumulative universes, native inductive families, eliminators for those families, propositional irrelevance, quotient principles, and eta-conversion. The syntax has one sort and beta reduction only. Any argument requiring those features must introduce a separate bridge or a separate represented calculus; it is not licensed by the claim in this section.

9235.16.10 Delivered Invariants

The delivered invariants are project-level as well as theorem-level. The `lean4/` development imports Lean 4 core facilities and project files, not mathlib. The project axiom audit rejects `axiom` declarations in `lean4/BEDC/`. The build is kept at zero sorry. The axiom-purity audit checks theorem dependencies against forbidden host principles, including `Classical.choice`, `Quot.sound`, and `propext`.

These invariants determine how the preceding claims are read. The markers in this section are not literary citations; each target is a Lean object in the project. The substrate, subject-reduction, confluence, normalization, checker, host-bridge, Church-algebra, and beta-conversion surfaces are therefore reproducible as mechanized claims over the represented calculus. The boundary clauses are equally binding: where a stronger statement is refuted or only conditional, the paper reads the formal target as a refusal to overstate the metatheory.

9235.17 Session Closing Summary

This closing section gathers the self-compilation surface as a single reader-facing account. The chapter has reached a point where the closed-term metatheory is represented inside Lean, the Church-encoded gallery is broad enough to exercise the object calculus, and the host bridge carries both directions of the intended comparison: ordinary Lean examples on one side, represented MetaCIC terms and beta traces on the other. The result is not a full kernel story. It is a closed, first-principles story about the fragment that BEDC uses when it treats its finite self-description as object language data.

The self-compilation chapter began from the L1/L4 distinction and the Godel–Tarski boundary. It then built the MetaCIC reconstruction as an explicit object calculus: syntax, contexts, closure, substitution, typing, beta reduction, finite beta chains, typed examples, bounded normalization, and bridges from finite BEDC histories. The present position is sharper than a catalogue of definitions. The calculus has closed-term preservation facts, subject-reduction interfaces with checked positive and negative resolutions, atom and closed-normal confluence results, candidate-based strong-normalization witnesses for the represented gallery, a decidable checker with soundness and completeness facts, Church-encoded data with beta laws, and HostBridge transport facts aligning represented examples with host computation.

The common invariant behind these pieces is that BEDC does not outsource the mathematics to a quotation facility or to mathlib. The represented syntax is an inductive datatype. De Bruijn shifting and substitution are recursive functions. Closure, typing, beta steps, beta-star chains, parallel beta, and conversion are object-language relations. Every Lean target cited below lives in this first-principles surface. The paper therefore reads the self-compilation chapter as a closed-term metatheory for the represented fragment, not as a statement about Lean's host kernel.

9235.17.1 Mechanized Theorems Table

The table records the principal theorem families used by the chapter. Each entry uses `\leanvariant` because this section is an index of already documented targets rather than the primary site for a single canonical paper theorem.

Layer	Mechanized content	Lean targets
Substrate	Closed terms, native substitution, shift-aware closure, and closure preservation across beta traces.	↪ variant `BEDC.MetaCIC.shift closed` ↪ variant `BEDC.MetaCIC.substitute closed` ↪ variant `BEDC.MetaCIC.closedAt shift` ↪ variant `BEDC.MetaCIC.closedAt substitute` ↪ variant `BEDC.MetaCIC.substitute preserves closed at` ↪ variant `BEDC.MetaCIC.betaStep preserves closed` ↪ variant `BEDC.MetaCIC.betaStarStep preserves closed`

Layer	Mechanized content	Lean targets
Typing rules	The object typing judgment, the beta-substitution interface, and shape-wise substitution preservation for Pi, lambda, and application.	↪ variant BEDC.MetaCIC.HasType ↪ variant BEDC.MetaCIC.BetaSubstitutionPreservation ↪ variant BEDC.MetaCIC.substitute preserves typing pi ↪ variant BEDC.MetaCIC.substitute preserves typing lam ↪ variant BEDC.MetaCIC.substitute preserves typing app cod sort var commutes ↪ variant BEDC.MetaCIC.substitute preserves typing via term induction
Subject reduction	The hypothesis-explicit subject-reduction theorem, its constructor cases, the closed independent-codomain application discharge, and the checked closed binder-domain obstructions.	↪ variant BEDC.MetaCIC.subject reduction beta ↪ variant BEDC.MetaCIC.subject reduction congApp1 ↪ variant BEDC.MetaCIC.subject reduction congApp2 ↪ variant BEDC.MetaCIC.subject reduction congLam ↪ variant BEDC.MetaCIC.subject reduction congPi ↪ variant BEDC.MetaCIC.subject reduction ↪ variant BEDC.MetaCIC.appArgTypeStable closed ↪ variant BEDC.MetaCIC.subject reduction closed app independent codomain ↪ variant BEDC.MetaCIC.not subject reduction closed

Layer	Mechanized content	Lean targets
Confluence	Atom diamonds, shape assemblers for compound terms, the conditional full parallel diamond, and the unconditional closed-normal surface.	↪ variant BEDC.MetaCIC.betaStarStep diamond sort ↪ variant BEDC.MetaCIC.betaStarStep diamond var ↪ variant BEDC.MetaCIC.betaParallel diamond sort ↪ variant BEDC.MetaCIC.betaParallel diamond var ↪ variant BEDC.MetaCIC.betaParallel app diamond of components ↪ variant BEDC.MetaCIC.betaParallel lam diamond of components ↪ variant BEDC.MetaCIC.betaParallel pi diamond of components ↪ variant BEDC.MetaCIC.betaParallel app shape diamond from substitution ↪ variant BEDC.MetaCIC.betaParallel diamond from substitution ↪ variant BEDC.MetaCIC.betaParallel diamond closed normal ↪ variant BEDC.MetaCIC.betaStarStep confluence closed normal
Strong normalization	The reducibility vocabulary, Pi adequacy, closed candidate consequences, and the gallery witnesses used by the closed self-compilation surface.	↪ variant BEDC.MetaCIC.StrongNormalizable ↪ variant BEDC.MetaCIC.Candidate ↪ variant BEDC.MetaCIC.ClosedCandidate ↪ variant BEDC.MetaCIC.PiCandidate ↪ variant BEDC.MetaCIC.CandidateSet ↪ variant BEDC.MetaCIC.candidate implies sn ↪ variant BEDC.MetaCIC.closedCandidate implies sn ↪ variant BEDC.MetaCIC.piCandidate implies sn ↪ variant BEDC.MetaCIC.pi candidate app ↪ variant BEDC.MetaCIC.PiAdequacy.piCandidate implies sn ↪ variant BEDC.MetaCIC.closed typed term sn from candidate ↪ variant BEDC.MetaCIC.closed typed term sn from closedCandidate ↪ variant BEDC.MetaCIC.closed typed term sn from candidateSet

Layer	Mechanized content	Lean targets
Decidable checker	The context-sensitive executable type checker, its closed wrapper, and the soundness and completeness theorems that connect it to the inductive typing judgment.	↪ variant BEDC.MetaCIC.inferTypeCtx ↪ variant BEDC.MetaCIC.inferTypeCtxClosed ↪ variant BEDC.MetaCIC.inferTypeCtx sound ↪ variant BEDC.MetaCIC.inferTypeCtx complete ↪ variant BEDC.MetaCIC.inferTypeCtxClosed sound ↪ variant BEDC.MetaCIC.inferTypeCtxClosed complete ↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete raw ↪ variant BEDC.MetaCIC.inferTypeCtx complete strict
Church gallery	Closed represented encodings for Booleans, naturals, pairs, options, and polymorphic combinators.	↪ variant BEDC.MetaCIC.church bool type ↪ variant BEDC.MetaCIC.church true ↪ variant BEDC.MetaCIC.church false ↪ variant BEDC.MetaCIC.church nat type ↪ variant BEDC.MetaCIC.church zero ↪ variant BEDC.MetaCIC.church one ↪ variant BEDC.MetaCIC.church succ ↪ variant BEDC.MetaCIC.church pair type ↪ variant BEDC.MetaCIC.church mk pair ↪ variant BEDC.MetaCIC.church fst ↪ variant BEDC.MetaCIC.church snd ↪ variant BEDC.MetaCIC.church option type ↪ variant BEDC.MetaCIC.church none ↪ variant BEDC.MetaCIC.church some ↪ variant BEDC.MetaCIC.church case option ↪ variant BEDC.MetaCIC.poly identity type ↪ variant BEDC.MetaCIC.poly compose type
Algebraic laws	Beta-star laws for Church Boolean projections, Church numeral application, polymorphic identity, and pair selection.	↪ variant BEDC.MetaCIC.church true proj ↪ variant BEDC.MetaCIC.church false proj ↪ variant BEDC.MetaCIC.church zero apply ↪ variant BEDC.MetaCIC.church one apply ↪ variant BEDC.MetaCIC.poly identity apply ↪ variant BEDC.MetaCIC.church pair value to selector

Layer	Mechanized content	Lean targets
Church natural recursion	Closed successor and addition facts for represented Church naturals, including successor application, zero-left addition, addition application, and the successor-left addition law.	↪ variant BEDC.MetaCIC.ChurchNatRec.church succ closed ↪ variant BEDC.MetaCIC.ChurchNatRec.church add type ↪ variant BEDC.MetaCIC.ChurchNatRec.church succ iter ↪ variant BEDC.MetaCIC.ChurchNatRec.church succ zero conv ↪ variant BEDC.MetaCIC.ChurchNatRec.church succ zero to one ↪ variant BEDC.MetaCIC.ChurchNatRec.church add zero ↪ variant BEDC.MetaCIC.ChurchNatRec.church add apply ↪ variant BEDC.MetaCIC.ChurchNatRec.church add succ left
HostBridge	Host-side beta equalities, represented typing transport, represented beta transport, and bounded-normalizer equations for the same finite examples.	↪ variant BEDC.HostBridge.MetaCICTransport.Host.host identity beta ↪ variant BEDC.HostBridge.MetaCICTransport.Host.host const beta ↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church true beta ↪ variant BEDC.HostBridge.MetaCICTransport.Host.host church zero beta ↪ variant BEDC.HostBridge.MetaCICTransport.transport identity well typed ↪ variant BEDC.HostBridge.MetaCICTransport.transport church true well typed ↪ variant BEDC.HostBridge.MetaCICTransport.transport church zero well typed ↪ variant BEDC.HostBridge.MetaCICTransport.transport church pair type well typed ↪ variant BEDC.HostBridge.MetaCICTransport.transport church option type well typed ↪ variant BEDC.HostBridge.MetaCICTransport.transport poly id two step redex star ↪ variant BEDC.HostBridge.normalize church true apply ↪ variant BEDC.HostBridge.normalize church zero apply

Layer	Mechanized content	Lean targets
Beta conversion	Symmetric join-based conversion, constructors from beta-star and beta-step, conditional transitivity, closed-normal transitivity, equality of closed normal forms, and a closed lambda-domain obstruction to a blanket typing invariance statement.	↪ variant BEDC.MetaCIC.BetaConv ↪ variant BEDC.MetaCIC.BetaConv.refl ↪ variant BEDC.MetaCIC.BetaConv.symm ↪ variant BEDC.MetaCIC.BetaConv.of betaStar left ↪ variant BEDC.MetaCIC.BetaConv.of betaStep left ↪ variant BEDC.MetaCIC.BetaConv.trans of confluence ↪ variant BEDC.MetaCIC.BetaConv.trans closed normal middle ↪ variant BEDC.MetaCIC.BetaConv normal forms equal closed ↪ variant BEDC.MetaCIC.BetaConv closed lam domain type obstruction ↪ variant BEDC.MetaCIC.not HasType BetaConv invariant closed

9235.17.2 How to Read the Table

The table is deliberately organized by proof pressure rather than by file path. A reader who starts from the substrate rows sees the object calculus as a small de Bruijn machine: terms are built by constructors, contexts are lists, closure is a boundary predicate, and substitution is the operation that must respect the boundary. This is the layer that prevents later examples from becoming informal syntax diagrams. The object calculus has to preserve its own scope before any self-compilation claim can use it.

The typing and subject-reduction rows then show how much of the ordinary metatheory is available at the represented level. The theorem named `subject_reduction` is intentionally hypothesis-explicit. It is not a paper promise that all conversion-sensitive facts hold automatically. It is a Lean theorem that lists the required stability interfaces and proves the constructor cases once those interfaces are supplied. The closed independent-codomain theorem is the positive closed case used by the gallery. The negative closed binder-domain targets explain why the paper does not silently strengthen the theorem beyond what the represented calculus proves.

The confluence row has the same discipline. The atom and closed-normal targets are unconditional. The application and full-diamond targets expose a substitution-compatibility premise. This matters because the self-compilation chapter uses confluence where its sources are atom-shaped, closed-normal, or part of the finite gallery. It does not need to pretend that every application residual of the represented calculus has been globally joined. The row therefore records both usable theorem content and the exact algebraic frontier.

The strong-normalization row should also be read as a scoped result. The project has a robust vocabulary of candidates, closed candidates, Pi candidates, neutral normal forms, and closed typed candidate consequences. That vocabulary is sufficient for the gallery witnesses and for the closed-term termination claims made in the chapter. It is not a semantic normalization theorem for every well-typed represented term. This distinction keeps the paper aligned with its Lean surface: the file names and theorem names show candidate adequacy where it is actually proved, not where a textbook proof might eventually go.

The decidable-checker row is the executable counterpart to the inductive typing relation. It closes a practical loop: the chapter can present a represented term, cite an inductive typing witness, and also cite a recursive checker whose soundness and completeness connect computation back to `HasType`. This is one of the reasons the self-compilation surface is not merely philosophical. The represented fragment can be used as data, and the project has a checked path for recognizing its closed typed examples.

The Church gallery and HostBridge rows give the computational content of the chapter. Church Booleans, naturals, pairs, options, and polymorphic combinators are familiar enough to act as inspection points. If the represented calculus mishandled binders, substitution, application, or closedness, these examples would be where the defect becomes visible. The host bridge then places those represented examples next to ordinary host Lean functions. The bridge does not collapse the two layers; it gives a finite alignment of examples across them.

The BetaConv row is a useful final row because it shows how the chapter treats conversion. Conversion is represented as a common-join relation over `BetaStarStep`. Reflexivity and symmetry are immediate from that definition. Transitivity is available under named confluence conditions and under the closed-normal middle case. The closed lambda-domain obstruction prevents the paper from claiming a blanket typing-invariance theorem for closed conversion. Thus even the conversion layer follows the same pattern: state the relation, prove the stable facts, and name the obstruction where the stronger reading fails.

9235.17.3 Closed Surface

The closed surface consists of the parts of the chapter that can be read without an unproved paper-side metatheoretic promise. The first closed part is the de Bruijn substrate. The calculus does not identify closedness with a typographic convention about missing variable names. It proves that the recursive operations respect the boundary they are meant to respect. This is the algebra behind every later finite evaluator trace: once a term is closed at boundary 0, beta contraction and beta congruence cannot manufacture a free represented variable.

The second closed part is the subject-reduction interface. The central theorem states the exact hypotheses needed for displayed-type preservation along one beta step. The beta case and congruence cases are checked directly. Inside the closed empty-context surface, the application-argument stability interface is discharged for independent codomains. The two binder-domain interfaces are not swept into prose: Lean supplies closed counterexamples showing that lambda-domain and Pi-domain blanket readings are false as stated. This gives the chapter a stronger boundary than an informal preservation paragraph would give. It says what works, what does not work, and which object-language witnesses enforce the distinction.

The third closed part is confluence at the level needed by the current self-compilation reading. Atom sources have no genuine beta forks. Closed normal sources have immediate joins. Compound terms have shape assemblers, and the full parallel diamond is available from the named substitution compatibility condition. The confluence story is therefore not hidden behind a universal Church–Rosser slogan. It is a structured surface: atom results, shape results, closed-normal results, and one explicit algebraic frontier for application residuals.

The fourth closed part is normalization for the represented gallery. The chapter uses strong-normalization witnesses and candidate structures, not a global theorem that every closed well-typed represented term normalizes. Sorts, variables, neutral applications, lambdas, Pis, Church Booleans, Church naturals, identity-shaped terms, and selected closed terms carry the witnesses needed by the paper. The main practical fact is enough for the BEDC surface: the concrete

represented programs used as examples come with closed typing, finite beta traces, or candidate-style termination evidence at the exact places where termination is cited.

The fifth closed part is the executable checker. The checker is not an oracle. It is a recursive function whose result is connected back to `HasType`. Soundness says that a returned type is justified by the inductive typing judgment. Completeness says that a derivable term is accepted by the checker at the displayed type. The closed wrapper specializes that story to the empty-context surface used by the chapter. This makes the gallery simultaneously human-readable and executable: a represented term can be shown in the paper, typed by an inductive proof, and checked by the decidable surface.

The sixth closed part is the Church gallery. The gallery covers Boolean choice, natural zero and one, successor, pairs, options, and polymorphic combinators. Its laws are beta-star laws over represented terms, not host-level reductions pasted into prose. The natural recursion surface closes successor and addition, including the relation between successor of zero and one, addition by zero on the left, application of addition, and the successor-left law. Multiplication and exponentiation are not cited here as mechanized targets because no Lean target with those names is present in the audited surface.

The seventh closed part is HostBridge. The bridge has two readings. On the host side, ordinary Lean functions compute by reflexivity. On the represented side, the corresponding MetaCIC terms have typing witnesses and beta traces. The bridge does not claim that host Lean elaboration is reflected into MetaCIC. It provides a checked finite transport gallery, large enough to say that the same examples that compute in the host also exist as represented object-language terms inside BEDC.

9235.17.4 Open Frontier

The open frontier is equally important to the chapter's meaning. The first frontier is open-term metatheory across binders. The closed empty-context surface is robust, but closedness at the outer boundary does not erase the local obligations created below lambdas and Pis. The checked binder-domain counterexamples show exactly why. A stronger theorem would need a transport discipline for binder contexts and displayed codomains, not merely a stronger wording of the closed theorem.

The second frontier is full confluence for `BetaStarStep` beyond the atom and closed-normal shapes. The atom diamonds are complete for atom sources, and the compound machinery exposes the right structure, but a full unconditional Church–Rosser result needs the remaining substitution compatibility law for application residuals. The chapter therefore treats `betaParallel_diamond_from_substitut` as a conditional theorem, not as a completed global confluence theorem.

The third frontier is the rest of Church natural arithmetic. Successor and addition are represented and mechanized. The present paper surface does not cite multiplication or exponentiation targets because the audited Lean tree does not contain them. Those operations fit the same object-language style, but the closing account reports only the checked targets that the audit can find.

The fourth frontier is the full host-kernel feature set: universe hierarchy, inductive families, propositional irrelevance, eta-conversion, cumulativity, quotients, and proof-irrelevant conversion. None of these is part of the present `BEDC.MetaCIC` scope. The calculus is a universe-0, beta-only, single-sort dependent-function fragment. This restriction is not an omission to hide. It is what keeps the formalization mathlib-free and axiom-free while still covering the represented terms used by the self-compilation story.

9235.17.5 Why This Surface Closes Here

This is the correct closing point because the closed-term metatheory now covers the substance of BEDC's self-compilation surface. The paper needs a finite represented syntax, a checked closure discipline, a typing relation, beta traces, a bounded evaluator, examples that exercise dependent functions, Church-encoded data, and a bridge to ordinary host examples. Those pieces are present. Their Lean targets are visible. Their restrictions are visible as well.

The zero-axiom invariant is central to the interpretation. BEDC's MetaCIC surface is not merely constructive in prose. It avoids project axioms, mathlib imports, sorry placeholders, and standard-library axiom dependencies in the validated discipline of the project. In that sense the represented metatheory sits in the same strict constructive class that one expects from a Brouwer or Bishop reading: theorems are derived from inductive definitions, recursive functions, structures, and prior theorems, not from an external logical extension.

The practical consequence is that the chapter can stop treating self-compilation as a metaphor. A finite BEDC history can be embedded as a closed represented term. A represented term can be typed. Finite represented beta traces can be checked. Closedness is preserved along those traces. Representative host computations have corresponding represented transport witnesses. The remaining task line is therefore not to argue again that reflection is possible in principle. It is to extend the BHist substrate bridge from closed atoms toward self-application while preserving the same audit discipline.

9235.17.6 Forward-Looking Summary

The next self-compilation work line is not a broader claim about every CIC feature. It is a deeper bridge from BEDC's finite-history substrate into the represented closed-term calculus. The current `BHistSubstrate` embedding already shows that a history index can be turned into a closed MetaCIC term and that the embedding is injective. The natural continuation is to give that embedding more semantic load: histories should not only enter the represented syntax as atoms, but should participate in typed represented programs that describe their own admissible transformations.

That direction keeps the chapter's scale under control. Instead of expanding the object calculus toward all of Lean, the project can ask a narrower question: which additional finite BEDC structures must be represented so that self-application is no longer merely a host-side explanation? The answer is likely to involve typed encodings of package signatures, bridge certificates, and closure-preserving transformations between history-shaped terms. Each candidate should enter by the same route used here: define the represented object, prove closure, prove typing, give finite beta behavior where computation is claimed, and expose every remaining hypothesis as a named Lean target.

The closed-term metatheory is already enough to support that route. It gives the low-level guarantee that represented self-application does not lose the boundary condition. It gives a subject-reduction theorem whose hypotheses tell the project which typing interfaces must be supplied. It gives a confluence surface for the atom and closed-normal cases likely to appear in finite certificate encodings. It gives candidate and strong-normalization witnesses for the small functional programs used as examples. It gives an executable checker for recognizing the typed fragment. Those are the ingredients needed for a disciplined expansion of the BHist bridge, provided each additional represented object remains inside the same audit rules.

The Church gallery suggests a useful design style for that expansion. The gallery does not begin with a universal theorem about all data encodings. It starts with concrete encodings whose

laws are inspectable. Booleans choose, naturals apply successors, pairs feed a selector, options choose between a default and a branch, and polymorphic functions exercise Pi structure. A self-application bridge can follow the same method: begin with finite, closed, typed objects whose behavior can be stated as beta-star or BetaConv laws, then use the resulting library as the surface that higher-level BEDC claims cite.

HostBridge also points to the right discipline. The host layer is valuable because it gives ordinary Lean computations that readers can recognize. The represented layer is valuable because it proves that corresponding examples exist inside the object calculus. Future bridge work should preserve that separation. A host computation can guide the example, but the represented term must carry its own typing and reduction evidence. A host theorem can explain the expected behavior, but the paper-side claim should cite the MetaCIC target that realizes the behavior in the represented language.

The open-term frontier should be approached only when the closed bridge requires it. The binder-domain counterexamples show that open contexts are not a matter of adding a quantifier to closed theorems. They require a context-transport account that respects shifted domains and displayed codomains. If self-application can be expressed through closed certificate objects and closed transformations, the project should keep using that route. If a later certificate necessarily crosses binders, the theorem statement should expose the transport structure directly rather than hiding it behind a closedness hypothesis.

The same caution applies to full confluence. A complete Church–Rosser theorem would be valuable, but the chapter already identifies the exact missing algebraic law. The correct next theorem would be the substitution compatibility needed by application residuals, stated over the actual `substitute` and `shift` functions of the represented calculus. Once that law is available, the existing conditional diamond target provides the route to the stronger result. Until then, the paper should continue to cite atom, shape, closed-normal, and conditional confluence targets with their actual scopes.

The arithmetic frontier is similarly concrete. Successor and addition are mechanized for Church naturals. Multiplication and exponentiation should be added only when their represented terms, closure facts, typing facts, and beta laws exist in Lean. The present closing section therefore treats them as mathematical next objects, not as paper facts. This is not a weakness of the self-compilation chapter; it is the rule that keeps the chapter audit-clean.

In short, the forward path is to make self-application less metaphorical while keeping the calculus small. Extend the finite-history bridge. Add represented certificate programs only when their closed typing and beta behavior are checked. Preserve the distinction between host computation and represented computation. Treat open binders, full confluence, richer arithmetic, universes, and inductive families as separate theorem projects. The closing surface is strong because it is narrow, visible, and mechanically audited.

9235.17.7 What BEDC Does Not Claim

BEDC does not claim to replace the Lean 4 kernel. The represented calculus is an object-language fragment inside Lean, not an implementation of Lean's elaborator, environment, universe constraints, inductive compiler, recursor generation, quotient layer, or kernel conversion checker. Host Lean remains the metalanguage in which BEDC proves facts about the represented object calculus.

BEDC also does not claim a universal mechanization of CIC. The present scope is deliberately constrained by three commitments: first-principles definitions, mathlib-free Lean, and zero project axioms. Those commitments make the proofs auditable, but they also keep the object calculus

narrow. The correct reading is a closed-term sub-fragment with a rich gallery and explicit bridges, not a general theorem about all terms and all type formers of CIC.

Finally, BEDC does not claim that every frontier has been discharged by naming it. Conditional theorems remain conditional. Counterexamples remain counterexamples. Open-term binder transport, full application-residual confluence, universe hierarchy, inductive families, propositional irrelevance, eta-conversion, multiplication, and exponentiation stay outside the closed surface reported here unless corresponding Lean targets enter the audited tree. This is the discipline that makes the closing meaningful: the chapter closes exactly the surface it can verify, and no larger one.

9235.18 What BEDC.MetaCIC Does Not Claim

This section fixes the load-bearing absences around `BEDC.MetaCIC`. The closed-term, empty-context surface is intentionally strong enough for the self-compilation account and intentionally too small to be read as a replacement for a production CIC kernel. The object calculus is represented inside Lean as a first-principles, mathlib-free development: it has represented terms, represented contexts, de Bruijn closure, shift, substitution, beta reduction, typed examples, bounded normalization, and a set of named metatheoretic interfaces. It is not a mirror of every feature that appears in Lean 4, Coq-style CIC, or the pure type systems literature.

The boundary matters because the positive results are now dense enough that a reader can overgeneralize them. A theorem about closed represented terms may look like a theorem about arbitrary open host terms. A beta-only reduction relation may look like definitional equality. A single-sort universe constant may look like a universe hierarchy. A gallery of Church encodings may look like native inductive families. Those readings are not the claims made here. The claim is narrower and more useful for BEDC: the represented closed fragment used by the self-compilation surface is explicit, checked, and bounded.

9235.18.1 Why This Boundary Is Stated

The self-compilation argument consumes finite represented artifacts. A BEDC reflection packet talks about names, histories, represented terms, typed closed witnesses, finite beta traces, and object-language certificates that are already inside the manuscript's finite surface. This use case does not require the whole host kernel. It requires a precise subcalculus whose syntax and metatheory can be inspected without importing untracked logical strength.

That is the reason `BEDC.MetaCIC` is formulated as an object calculus. The Lean kernel remains the trusted host that checks the formalization. The MetaCIC layer is not another implementation of that kernel. It is a represented calculus whose constructors and relations are small enough to be audited by the project gates. When the paper cites the MetaCIC surface, the reader should read the claim as a theorem about the represented fragment, not as a theorem about every term, universe, recursor, quotient, and definitional equality feature accepted by Lean.

The most important narrowing is the closed empty-context reading. The object-level invariant is:

$$\mathrm{ClosedEmptyMetaCIC}(t, A) \quad \equiv \quad \texttt{ClosedAt}\ 0\ t \wedge \texttt{HasType}\ []\ t\ A.$$

This is the surface on which the self-compilation examples live. It is also the surface on which closure preservation, selected subject-reduction interfaces, closed-normal confluence facts, candidate witnesses, executable checking, and finite HostBridge examples are read. The calculus

exposes stronger-looking open-term statements only when they have explicit hypotheses or checked obstructions. Therefore this section states the negative space in the same voice as the positive theorems: not as cautionary prose, but as part of the mathematical specification.

9235.18.2 Out of Scope by Design

Feature	Boundary in `BEDC.MetaCIC`	Rationale
Universe hierarchy	The represented syntax has a single universe-0 sort, not `Sort 0`, `Sort 1`, cumulativity, or universe polymorphism.	The self-compilation surface needs closed represented witnesses in one object-level sort. Adding universe levels would change the syntax, typing rules, conversion relation, and metatheory obligations.
Inductive families	The represented object language has the atomic term grammar `sort`, `var`, `app`, `lam`, and `pi`.	Church encodings are represented; native families such as `Vec n` or `List A` are not object-language constructors with recursors and computation rules.
Proposition irrelevance	The calculus does not separate `Prop` from `Type`.	There is one object sort. No proof-irrelevance principle is assumed, encoded, or needed for the closed self-compilation witnesses.
η-conversion	The reduction relation is strictly beta-only.	The mechanized reduction surface is generated by beta contraction and congruence. Extensional equality principles are not silently folded into conversion.
Pattern matching beyond term constructors	The proofs inspect represented syntax by constructor analysis, for example `cases ... with` over `Term`.	The object language does not contain a general pattern-matching eliminator. Meta-level proof scripts may analyze inductive syntax, but that is not an object-language feature.
General fixpoints and well-founded recursion	The calculus uses inductive relations and structurally recursive host functions where they are explicit.	There is no represented general recursive definition principle. Termination evidence is carried by inductive witnesses, candidates, fuel, or structural recursion in the host.
Open-term metatheory across binders	The manuscript does not claim the full open-term substitution and subject- reduction theory of a production kernel.	The substrate has a documented obstruction: the naive zero-depth substitution composition law fails for unclosed anchors, as recorded by counterexample #108 and by the named MetaCIC counterexample.

Universe hierarchy. The represented grammar contains a single sort constructor. The paper may write "universe-0" as a mnemonic for the object-level scale, but that is not a commitment to a hierarchy of object universes. There is no represented `Sort 1`, no lifting relation between universes, no cumulativity rule, and no universe-polymorphic binder discipline. Consequently, any theorem whose proof would use universe constraints, universe maximums, predicative stratification, or cumulativity transport lies outside the claim surface.

This omission is intentional. Universe hierarchy is not a cosmetic annotation on an existing calculus. It changes the formation rules for Pi, the shape of type inference, the statement of conversion, and the induction principles used in metatheory. BEDC's self-compilation surface does not need those rules. It needs closed represented terms whose typing witnesses can be checked without importing a second layer of universe bookkeeping.

Inductive families. The object language has dependent functions and application, but it does not have native inductive families. A family such as `Vec A n` would require object-level constructors, parameters, indices, eliminators, recursors, computation rules, and positivity or guardedness conditions. A production kernel has to define and check those interfaces. `BEDC.MetaCIC` does not.

The Church gallery should therefore be read correctly. Church Booleans, Church numerals, pairs, options, and combinators are represented as terms in the dependent-function fragment. They demonstrate that the fragment can carry useful closed encodings and beta laws. They are not a proof that the object language contains native inductive declarations, pattern matching, or recursor computation for arbitrary families.

Proposition irrelevance. The represented calculus does not distinguish `Prop` from `Type`. It has no separate impredicative proposition universe and no rule that identifies all proofs of the same proposition. Thus proposition irrelevance is not available as an object-language theorem, a conversion principle, or an implicit metatheoretic assumption.

This is also aligned with the axiom-purity discipline. BEDC avoids moving from logical equivalence to definitional identity unless a first-principles object-level construction supplies the equality in the represented calculus. That discipline is why `propext` is prohibited by audit rather than treated as a harmless background feature.

η-conversion. The reduction relation is beta-only. Lambda terms reduce by applying a lambda to an argument and substituting the argument into the body. Congruence allows beta steps under applications, lambdas, and Pis. No η rule says that a function is convertible to a lambda that applies it to a fresh variable. No record, pair, or function extensionality rule is folded into conversion.

This matters for subject reduction and confluence statements. A beta-only calculus has different normal forms, different residual obligations, and a smaller conversion relation than a beta-eta calculus. Any result that needs η-expansion, η-contraction, or extensional comparison of functions is outside the MetaCIC claim unless it is restated as an explicit additional relation with its own proof obligations.

Pattern matching beyond term constructors. The Lean proofs may analyze the inductive syntax of represented terms. This is ordinary host-side reasoning: a theorem about `Term` can proceed by cases over `sort`, `var`, `app`, `lam`, and `pi`. Such proof analysis does not mean that the represented object language has a general match expression or a dependent eliminator for every future object datatype.

The distinction is important. Host-side induction over the encoding is how the metatheory is proved. Object-side pattern matching would be a feature of the represented calculus itself, with typing rules, reduction rules, and subject-reduction obligations. BEDC uses the former and does not claim the latter.

General fixpoints and well-founded recursion. The object language does not contain a general fixpoint constructor. It also does not contain an object-level well-founded recursion principle. Where the formalization needs recursion, it uses ordinary Lean structural recursion on the encoded data, a fuel-indexed bounded function, or an inductively generated relation. Those are host-side implementation techniques and object-level relations, not an exported recursive programming language inside MetaCIC.

This boundary keeps termination evidence visible. Strong-normalization witnesses are carried by inductive predicates and candidate structures. Bounded normalization is controlled by fuel. Beta-star traces are finite inductive chains. None of these witnesses should be read as a general termination checker for arbitrary represented recursive definitions.

Open-term metatheory. The closed empty-context surface is not a disguised proof of full open-term metatheory. Open contexts require binder transport, shift/substitution interaction across changing cutoffs, and substitution-composition laws that are sensitive to closure hypotheses. The substrate obstruction is concrete: the naive zero-depth composition law for unclosed anchors is false, and the counterexample is documented as part of the MetaCIC boundary.

The existing closed facts therefore have an exact reading. Closure preservation says that beta steps do not create free variables from closed sources. The positive application subject-reduction result works for closed independent codomains. The binder-domain counterexamples show why closedness alone is not enough for blanket lambda-domain or Pi-domain preservation. Those are not gaps in an otherwise implicit open theorem. They are the formal end of the present claim surface.

9235.18.3 Comparison With Standard CIC Literature

Reference point	What it studies	BEDC reading
Coquand–Werner CIC	A rich dependent type theory with universes, inductives, conversion, and kernel-scale metatheory.	`BEDC.MetaCIC` is a represented single-sort beta-only fragment. It does not mirror normalization by evaluation or universe cumulativity.
Girard reducibility candidates	A semantic normalization method organized around candidates and adequacy.	BEDC has candidate vocabulary and checked candidate consequences for the closed gallery, but not a full mechanized adequacy chain for all represented typed terms.
Pure type systems	A framework parameterized by sorts, axioms, and product rules.	BEDC is best read as a single-sort sub-PTS encoded by an inductive typing relation, not as a general PTS framework.

Reference point	What it studies	BEDC reading
Lean 4 kernel	The host implementation that elaborates, reduces, and checks Lean code.	BEDC is checked by Lean but does not reimplement Lean. Its theorems avoid the standard-library axioms forbidden by the project audit.

Coquand–Werner CIC. The name MetaCIC signals the object of comparison, not identity of scope. Classical CIC presentations support a richer theory than the fragment used here: universe hierarchies, cumulativity, inductive types, eliminators, and a conversion relation integrated with the kernel. BEDC does not replicate that whole package. In particular, it does not present a normalization-by- evaluation algorithm for full CIC, and it does not mechanize universe cumulativity.

The comparison should be read as follows. BEDC takes the kernel-facing lesson that syntax, typing, substitution, and reduction must be formal objects. It then applies that lesson to the closed self-compilation surface. The result is a small, auditable object calculus rather than a production assistant kernel.

Girard reducibility candidates. The normalization layer uses candidate terminology because candidates are the right vocabulary for strong-normalization evidence. The formalization has `Candidate`, `ClosedCandidate`, `PiCandidate`, candidate sets, closure under beta, Pi application consequences, neutral-normal witnesses, and concrete gallery instantiations.

That is not the same as a full Girard-style adequacy theorem for every well-typed represented term. The adequacy chain would have to assign candidate interpretations to all types in the fragment, prove closure under substitution and conversion at the required generality, and then derive strong normalization for arbitrary typed terms. BEDC has the framework and the consumed closed witnesses; it does not claim the entire adequacy development.

Pure type systems. Pure type systems classify calculi by specifying sorts, axioms, and product rules. `BEDC.MetaCIC` can be compared to a very small sub-PTS: one represented sort and the dependent-function rules needed by the object grammar. That comparison is useful for orientation, but the formalization is not a generic PTS engine parameterized by arbitrary rules.

This matters for reviewer expectations. A theorem proved for the BEDC fragment should not be lifted to all PTSs, to all single-sort systems, or to all extensions with additional products. The theorem is about the inductive relations actually present in `BEDC.MetaCIC`.

Lean 4 kernel. Lean is the host that checks the project. It is not the object calculus being replaced. The host has its own elaborator, reduction machinery, inductive declaration checker, quotient support, universe handling, and runtime behavior. `BEDC.MetaCIC` sits inside that host as a verified object language.

The project-level distinction is sharper than ordinary shallow embedding. BEDC explicitly audits its own theorems for forbidden dependencies. The formal content cited by this chapter is required not to depend on `Classical.choice`, `Quot.sound`, or `propext`. Thus the MetaCIC results are not imported from host classical reasoning or quotient reasoning. They are checked within the constructive, axiom-clean discipline that BEDC imposes on its own theorem surface.

9235.18.4 Zero-Axiom Invariants Delivered

The negative scope is paired with a positive purity claim. BEDC does not claim to cover all of CIC, but the part it does cover is not supported by hidden project axioms. The invariant is audited at two levels: source files must not declare project axioms, and theorem dependencies must not rely on the forbidden host axioms.

Invariant	Delivered form	Consequence
No `Classical.choice`	The formal development avoids nonconstructive choice in cited theorem dependencies.	Existence claims must carry constructive witnesses or be expressed as explicit interfaces rather than selected by an unwitnessed global choice principle.
No `Quot.sound`	The MetaCIC surface does not use quotient types to identify represented objects.	Equality and conversion are represented directly by inductive relations and explicit joins, not by quotienting syntax.
No `propext`	The formalization avoids proposition extensionality in theorem dependencies.	Logical equivalence is not silently promoted to proposition equality. Proofs are structured to avoid match shapes that create such dependencies.
Mathlib-free	The Lean package for BEDC does not import mathlib.	The object calculus is developed from Lean 4 core facilities and project definitions rather than a large external theorem base.
No sorry placeholders	Public theorem statements cited by the manuscript are checked proofs or explicit definitions and interfaces.	Unproved strength is not smuggled into the paper as a placeholder theorem.

Constructive dependency discipline. Avoiding `Classical.choice` is not merely a style rule. In this project, a theorem that depends on choice would close over an unwitnessed selection operation. That would be the wrong kind of evidence for a self-compilation surface whose artifacts are finite, named, and carried by explicit histories. The formal proofs therefore use constructors, induction, structural recursion, and displayed hypotheses rather than classical selection.

No quotient collapse. Quotient types are also outside the cited MetaCIC theorem dependencies. Represented terms remain syntactic data. Beta conversion is represented by relations and join evidence, not by collapsing syntax into equivalence classes. This is essential for the paper's auditability: a reader can see which beta paths, closure predicates, and typing hypotheses are being used.

No proposition extensionality. The project treats `propext` as a real logical dependency. It is not allowed to enter through a convenient proof pattern. This affects the shape of the Lean development: proofs are written so that proposition equality is not required when ordinary constructor

reasoning, explicit implications, or relation evidence is sufficient. The result is a theorem surface that can be read constructively.

No mathlib dependency. The mathlib-free condition keeps the dependency chain small. It does not mean the work rejects established mathematics. It means the specific MetaCIC surface is audited as a first-principles Lean 4 development rather than as a collection of imported high-level results. The relevant objects are the project's own inductives, definitions, and theorems.

No unproved placeholders. The no-sorry invariant keeps the paper and Lean states aligned. When the manuscript cites a checked theorem, there is a proof term. When the project does not have a proof at the required strength, the theorem is either stated as an explicit interface, constrained to a checked sub-scope, or rejected by a counterexample. That is why the boundary section can be part of the scientific claim rather than a disclaimer outside the mathematics.

9235.18.5 Why This Matters for Self-Compilation

BEDC self-compilation is not a demand that the project reconstruct every feature of the host kernel inside itself. The operational surface is smaller: the project reflects finite, typed, closed artifacts. The represented terms used in that surface are object-language data. Their typing witnesses, closure facts, beta traces, and normal-form witnesses are the objects that the self-compilation argument actually consumes.

For that reason, the excluded features above do not weaken the stated self-compilation closure. Universe hierarchy does not participate in the finite closed witnesses currently consumed by the reflection surface. Native inductive families do not participate because the gallery uses Church-encoded data inside the dependent-function fragment. Proposition irrelevance does not participate because the object language has no separate proposition layer. η-conversion does not participate because the beta traces are the reduction evidence being checked. General fixpoints do not participate because termination evidence is supplied by structural recursion, fuel, inductive traces, or candidate witnesses.

Open-term metatheory is the most subtle boundary. A production kernel must handle open contexts constantly. BEDC's self-compilation surface, however, uses closed represented artifacts at the reflection boundary. The substrate still has to understand binders, shifting, and substitution, because closed terms may contain binders internally. But the exported self-compilation claim does not need a blanket theorem for arbitrary open object contexts. It needs the checked closed facts and the named local interfaces already present in the MetaCIC layer.

Thus the boundary is not a retreat from the self-compilation goal. It is the shape of that goal. The positive surface and the negative surface meet at the same line:

$$\text{closed represented terms} + \text{empty object context} + \text{beta-only traces} + \text{explicit certificates}$$
$$\text{rather than full host kernel reflection.}$$

This line is what makes the claim auditable. A reviewer can ask whether a self-compilation step uses only the displayed closed surface. If it does, the MetaCIC mechanization is relevant. If it requires universe cumulativity, native inductives, proposition irrelevance, η-conversion, general recursion, or an open-context substitution theorem, then it is outside the present commitment and must be stated as a separate extension.

9235.18.6 What Would Extend the Scope

The omitted features are not impossible additions. They are substantial theory projects with their own syntax, typing, reduction, and audit burden. The present manuscript does not rely on them, so it does not claim them. A larger MetaCIC surface would have to expose each one as first-principles content rather than as an informal analogy with an external kernel.

Universe hierarchy extension. A universe hierarchy would require a layered represented syntax with at least `Sort 0` and `Sort 1`, and likely a general level language. The typing judgment would need formation rules for sorts and Pis that compute result levels. Conversion would need to account for cumulativity if cumulativity is included. The metatheory would then have to prove that substitution, subject reduction, confluence, normalization witnesses, and checking remain valid in the level-aware calculus.

Inductive-family extension. Native inductive families would require represented declarations, constructors, parameters, indices, eliminators, and computation rules. The project would also need a positivity or guardedness condition appropriate to the represented language. Only after those pieces exist could a theorem about `Vec`, `List`, or indexed eliminators be stated as a MetaCIC theorem rather than as a Church-encoded example.

Proposition-layer extension. A separate proposition layer would require syntax and typing rules that distinguish proof-relevant and proof-irrelevant regions, or at least a clear choice about whether proof irrelevance is object-level, meta-level, or absent. If proposition irrelevance were included, it would need its own constructive or explicitly axiomatized status. Under BEDC's present audit discipline, it could not enter as an untracked dependency on `propext`.

Open-term metatheory extension. The open-term extension would have to start from the known obstruction rather than ignore it. It would need closure-indexed substitution composition, binder-context transport, weakening, exchange if contexts are generalized, and subject reduction stated with the exact transported type. The existing counterexamples show where naive statements fail. A successful extension would replace those naive statements with the right shift-aware invariants.

Kernel-bridge extension. A bridge to the host kernel would require a relation between represented MetaCIC judgments and actual Lean elaboration or kernel checking. That relation would have to be partial, because the represented fragment is smaller than Lean. It would also have to specify quotation, erasure, normalization, universe behavior, and failure modes. Such a bridge would be a separate metatheoretic layer above the present object calculus.

These extensions are ambitious, and none is required for the present self-compilation boundary. The commitment here is narrower: maintain the closed represented fragment as an axiom-clean, mathlib-free, sorry-free surface; state every consumed interface explicitly; and keep every excluded kernel feature outside the claim until its own represented syntax and metatheory are supplied.

Part XI

Capstones

Chapter 9236

Capstone Results

Visions chapters collect exploratory bridge narratives that span multiple BEDC concrete instances. Each vision is inspiration-only: AI revision rounds consume these files as context for proposing new `concrete_instances/` chapters but never write back to `parts/visions/`. When a vision's content is fully realized by one or more concrete-instance NameCert constructions, the vision chapter carries `\concretizedIn` markers pointing to the realizations.

Concretized in §7381: Vision coverage index packets realize the top-level vision routing surface as finite source-label, concrete-target, marker-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §7484: The vision coverage ledger realizes the top-level capstone routing surface as finite source-label, concrete-target, marker-ledger, replay, and NameCert rows.

Concretized in §7573: Affine filter hull packets realize the top-level vision routing surface as finite affine-generator, membership-event, generated filter-base, routing, replay, provenance, and NameCert rows.

Concretized in §8820: Finite vision marker audits realize the source-label, concrete-target, marker-ledger, audit-readback, transport, replay, provenance, and NameCert route for the visions index.

Chapter 9237

Compactness Implies Uniform Continuity

A continuous function on a compact metric space is uniformly continuous, constructively. This chapter assembles the components already isolated in the kernel: the metric carrier with its located distance (chapter 57), the pointwise modulus chain encoded by Cont on continuous naming certificates (chapter 59), and the compactness witness packaged as subset + located + finite ε-net + intermediate + compact (chapter 60). The argument identifies the compactness data as a finite cover indexed by a $\mathsf{ProbeBundle}$, folds the finitely many $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ radius histories through a certified lower-bound ledger, and reads off a uniform-modulus relation depending only on the tolerance history. The main theorem states this in BEDC form: from a continuous naming certificate together with a compact naming certificate on the same metric carrier, the kernel produces a uniform-continuity ledger, with no appeal to $\mathsf{Classical.choice}$ and no excluded middle.

9237.1 Uniform modulus from located finite covers

Definition 9237.1 (Pointwise modulus on $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$)**.** *Let X, Y be metric naming certificates and let $\mathsf{NameCert}_f$ be a continuous-function naming certificate witnessing $f\colon X \to Y$. A* pointwise modulus ledger *for $\mathsf{NameCert}_f$ is the BEDC relation on histories*

$$(x, \varepsilon, \delta, \mathsf{cert})$$

whose accepted rows carry a source-point history x, a positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ tolerance history ε, a positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ precision history δ, and a certificate history cert. The certificate row is carried by $\mathsf{ContinuousModulusWitness}(x, \delta, \mathsf{cert})$ together with the surrounding Cont rows of $\mathsf{NameCert}_f$: whenever $\mathsf{MetricDistanceWitness}(x, x', d)$ certifies a source distance bounded by δ, the graph and target-metric ledgers certify a target distance bounded by ε. The public datum is this certified relation, not a map from external points and tolerances to external rationals.

Definition 9237.2 (Uniform modulus)**.** *A* uniform modulus ledger *for $f\colon X \to Y$ is a BHist relation*

$$\mathsf{UniformModulusLedger}(\varepsilon, \delta, \mathsf{u})$$

from positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ tolerance histories to positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ precision histories. An accepted row carries a ledger certificate u such that, for every pair of source-point histories x, x' and every distance history d with $\mathsf{MetricDistanceWitness}(x, x', d)$ bounded by δ, the graph ledger for

f and the target metric certificate produce a target-distance history bounded by ε. The precision is therefore selected by a BEDC relation over histories and classifiers, rather than by an external selector.

Definition 9237.3 (Located finite ε-net)**.** *A* located finite ε-net *for a metric carrier X is a* ProbeBundle*-indexed family of center histories. For a positive* $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *tolerance history ε, the compact certificate supplies a bundle $\mathcal{P}_\varepsilon$, a center history c_p for every probe name p with* $\mathsf{InBundle}(p, \mathcal{P}_\varepsilon)$*, and coverage rows saying that every source-point history x has some bundled probe p with a distance history* $\mathsf{MetricDistanceWitness}(x, c_p, d_p)$ *certified below ε. The finite carrier is the closed inductive* ProbeBundle *together with its coverage witnesses; no external enumeration is exposed.*

9237.2 The kernel construction

Lemma 9237.4 (Subordinate finite modulus cover)**.** *Let X be a compact metric carrier and let $f\colon X \to Y$ have the pointwise modulus ledger of Definition 9237.1. For every positive* $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *tolerance history ε, the compact certificate supplies a nonempty* ProbeBundle *$\mathcal{P}_\varepsilon$ and, for every bundled probe p, histories*

$$(c_p, \rho_p, \delta_p, \mathsf{cert}_p)$$

such that ρ_p and δ_p are positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *histories, $(c_p, \varepsilon/3, \delta_p, \mathsf{cert}_p)$ is accepted by the pointwise modulus ledger, ρ_p is certified below δ_p, and every source-point history x has a bundled probe p whose center satisfies the certified half-radius coverage row $d_X(x, c_p) < \rho_p/2$.*

Certificate reading. Apply compactness to the located positive-radius ledger whose member at x is supplied by an accepted pointwise-modulus row for tolerance $\varepsilon/3$. The finite-cover certificate returns a ProbeBundle, center histories, radius histories, and subordination rows already tied to those pointwise rows, so the comparisons $\rho_p \le \delta_p$ are part of the returned data.

The located half-radius clause is the cover witness used by the kernel construction below: for each input point x, it produces a bundled probe name p with $d_X(x, c_p) < \rho_p/2$. No scale depending on the eventual uniform-modulus ledger is used to obtain this bundle. □

Theorem 9237.5 (Capstone: finite BEDC fold supports compact uniform modulus)**.** *Let X and Y be metric BEDC carriers whose distance rows are supplied by* MetricDistanceWitness*, and let a BEDC history carry*

$$\mathsf{CompactWitnessCarrier}(\mathsf{subset}, \mathsf{located}, \mathsf{finite}, \mathsf{intermediate}, \mathsf{compact})$$

for X. Let another history carry

$$\mathsf{ContinuousFunctionCarrier}(\mathsf{source}, \mathsf{map}, \mathsf{target}, \mathsf{modulus}, \mathsf{cert})$$

for a graph $f\colon X \to Y$ over the same source metric. For each positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *tolerance history ε, suppose Definition 9237.4 returns a nonempty* ProbeBundle *spine*

$$\mathcal{P}_\varepsilon$$

whose bundled probes carry positive $\mathsf{Rat}^\uparrow$ *radius histories, center histories, pointwise-modulus rows, and subordinate half-radius cover rows. Then primitive recursion over $\mathcal{P}_\varepsilon$, using the* $\mathsf{Rat}^\uparrow$ *order readback of Definition 54.22 at each successor step, produces a unary precision history μ_ε and a fold certificate saying that μ_ε is classifier-stably below every bundled half-radius ledger. The row* $\mathsf{UniformModulusLedger}(\varepsilon, \mu_\varepsilon, \mathsf{u}_\varepsilon)$ *of Definition 9237.2 is then certified for the same continuous graph.*

Certificate assembly. Read the finite cover from the compact witness: the finite and located rows give a nonempty ProbeBundle of centers, positive $\mathsf{Rat}^{\uparrow}$ radius histories, and for every point of X a bundled probe whose center lies within the displayed half-radius.

Transport the radius histories through the $\mathsf{Rat}^{\uparrow}$ total-order readback. Since the bundle is nonempty and every entry is carried by the rational positive-radius certificate, primitive recursion over $\mathcal{P}_\varepsilon$ compares the accumulator with the next half-radius ledger and keeps a certified lower-bound history. The uniqueness row of Definition 54.24 makes this fold stable under the classifier, so the selected lower bound is a BEDC history rather than an external choice.

At each bundled center, the ContinuousFunctionCarrier modulus row reads back the local implication from the graph certificate: a distance bounded by the selected radius below δ_p gives output distance below $\varepsilon/3$. The subordinate-cover rows guarantee that the same center can be used for any input point, and the fold certificate guarantees that the proposed uniform precision is below the chosen local radius.

Finally apply the metric semantic certificate: symmetry converts $d_X(x, x')$ to the orientation needed at the center, and the triangle row combines $d_X(x, c_p)$ with $d_X(x, x')$ to put x' inside the same local ball. Applying the target-metric triangle row to $f(x)$, $f(c_p)$, and $f(x')$ yields the displayed uniform estimate. Thus $\mathsf{UniformModulusLedger}(\varepsilon, \mu_\varepsilon, \mathsf{u}_\varepsilon)$ is certified by the finite BEDC rational-fold ledger. □

Theorem 9237.6 (Continuous on compact yields uniform modulus)**.** *Let X be a metric carrier with a compact naming certificate $\mathsf{NameCert}_X$ exposing the* ProbeBundle *finite-net witness of Definition 9237.3, and let $f\colon X \to Y$ be witnessed by a continuous naming certificate $\mathsf{NameCert}_f$ over X exposing the pointwise modulus ledger of Definition 9237.1. For every positive $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ tolerance history ε, primitive recursion over the compact bundle and the certified $\mathsf{Rat}^{\uparrow}$ lower-bound fold produces histories δ_ε and u_ε such that*

$$\mathsf{UniformModulusLedger}(\varepsilon, \delta_\varepsilon, \mathsf{u}_\varepsilon)$$

holds for f. The conclusion is a certified uniform-modulus ledger from tolerance histories to precision histories, not an external modulus selector.

Sketch. Fix a positive tolerance history ε. Invoke Definition 9237.4 to obtain a nonempty ProbeBundle $\mathcal{P}_\varepsilon$ whose probes carry centers c_p, pointwise precision histories δ_p, subordinate radius histories ρ_p, and half-radius coverage rows. Then Definition 9237.5 supplies the classifier-stable rational-fold ledger μ_ε below every bundled half-radius ledger.

Let x, x' be source-point histories with source distance certified below μ_ε. The subordinate cover gives a bundled probe p with $d_X(x, c_p) < \rho_p/2$. Because the fold ledger certifies $\mu_\varepsilon \le \rho_p/2$, the triangle row gives

$$d_X(x', c_p) \le d_X(x', x) + d_X(x, c_p) < \rho_p.$$

Also $d_X(x, c_p) < \rho_p/2 < \rho_p$.

The bound $\rho_p \le \delta_p$ lets the pointwise modulus row at c_p apply to both x and x'. Hence $d_Y(f\,c_p, f\,x) < \varepsilon/3$ and $d_Y(f\,c_p, f\,x') < \varepsilon/3$. The triangle inequality on Y gives $d_Y(f\,x, f\,x') < \varepsilon$. The construction uses only the finite subordinate cover, primitive recursion over ProbeBundle, the certified finite-fold rational ledger over $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$, and the metric certificates. □

9237.3 Axiom-purity of the uniform-modulus construction

Remark 9237.7 (No appeal to non-constructive principles)**.** *The construction in Definition 9237.6 uses only primitive recursion on the finite net (a* Cont*-shaped ledger over the closed inductive*

ProbeBundle *of centers), arithmetic on* $\mathbb{Q}_{>0}$ *(rational addition, multiplication, and division by* 2*, all kernel-level operations on* $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *from chapter 37), and the located-distance comparison built into* $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$*. None of these operations passes through* Classical.choice*,* Quot.sound*, or* propext*. The chapter therefore falls under the same axiom-purity gate as the rest of the manuscript.*

Remark 9237.8 (Heine–Cantor as a kernel theorem)**.** *Definition 9237.6 is the BEDC form of the Heine–Cantor theorem of classical analysis. The classical proof uses sequential compactness and an extracted convergent subsequence; the BEDC proof uses the located finite* ε*-net and a certified lower-bound fold over its* ProbeBundle *radius ledger. The two routes deliver the same uniform modulus relation when the metric carrier is taken as* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ *restricted to a closed bounded interval (chapter 56), making the classical result a derived corollary of the kernel-level statement and not the other way around.*

Concretized in §619: The finite ProbeBundle rational lower-bound fold is realized as a concrete NameCert carrier for compact-uniform-modulus handoff.

Concretized in §8630: The finite subcover modulus budget is realized as compact-net, rational-radius, local-modulus, fold-minimum, and uniform-modulus NameCert data.

Concretized in §6453: Compact-net modulus stability is realized as finite compact-net, continuous-map, uniform-modulus, metric-triangle, and Real-seal NameCert data.

Concretized in §7827: Compact interval uniform continuity is realized as finite interval, compact-metric, continuous-map, uniform-modulus, and Real-seal NameCert data.

Concretized in §7937: Compact derivative bounds are realized as finite compact-interval, continuous-map, derivative-observation, uniform-modulus, compact-net radius, RegSeqRat, and Real-seal NameCert data.

Concretized in §6814: Finite Sobolev derivative windows, norm ledgers, compact handoff rows, and RegSeqRat readback realize a compact-uniform-continuity consumer route.

Concretized in §6997: The finite cover star-refinement modulus packet realizes a compact finite-cover route from star refinements and local moduli to a tolerance-only uniform-modulus handoff.

Concretized in §6125: The finite ProbeBundle subcover ledger realizes the compact finite-cover handoff used by the uniform-modulus construction.

Concretized in §412: $\mathsf{ArzelaAscoli}^\uparrow$ realizes the compact-uniform-continuity route through finite equicontinuity rows, compact-metric source data, bounded-family ledgers, and diagonal-selection NameCert handoff.

Concretized in §8711: Compact uniform modulus extraction is realized as finite compact-net, ContinuousMap local-radius, DyadicRatCore tolerance, RegSeqRat readback, and Real-seal NameCert data.

Concretized in §399: The finite majorant and uniform-tail rows realize the series-majorant branch of compact-uniform and uniform-convergence consumers without host infinite sums or selected limits.

Concretized in §8269: Finite epsilon-net rows realize the compact-uniform-modulus handoff.

Concretized in §7665: The positive-series comparison packet realizes the nonnegative majorant comparison branch through RealSeries, CauchySeriesCriterion, CauchyMajorantSeries, DyadicRatCore, and RealUp rows.

Concretized in §7980: Sequential continuity over RealUp is realized as finite stream, regular-Cauchy, continuity-modulus, and Real-seal NameCert data.

Concretized in §8088: Bishop located compact sets realize the compact-metric route through finite-net, metric readback, RegSeqRat, StreamName, and locator rows.

Concretized in §8483: Compact-uniform splitting ledgers realize the finite split from pointwise compact-cover moduli to one tolerance-only uniform-modulus handoff.

Concretized in §9126: The compact modulus partition ladder realizes the finite partition-refinement route from compact nets and pointwise moduli to uniform-modulus handoff rows.

Concretized in §8994: FinitePrefixCompactnessUp realizes the compact-uniform finite-prefix branch through FiniteFanTheorem, CantorSpace, StreamName, RegSeqRat, and RealUp rows.

Remark 9237.9 (Composition with the real-completeness capstone)**.** *The construction composes with the real-completeness capstone (chapter 9238): once* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ *is upgraded to a regular Cauchy carrier, every closed bounded interval* $[a, b] \subset \mathsf{Real}^\uparrow$ *inherits a compact naming certificate via the standard located* ε*-net of dyadic rationals at scale* $\lceil 1/\varepsilon \rceil$*, and Definition 9237.6 then gives uniform continuity of every continuous* $f\colon [a, b] \to \mathsf{Real}^\uparrow$ *with an explicit uniform-modulus ledger. The chapter is therefore the canonical bridge from the analytic certificate layer to the constructive analysis literature.*

Concretized in §391: Uniform continuity is realized as the finite ProbeBundle fold extracting a tolerance-only modulus from compact and continuous certificates.

Concretized in §369: Uniform modulus packets are realized as finite ProbeBundle rational-fold NameCert carriers for compact-continuous consumers.

Concretized in §7480: The dyadic interval subdivision packet realizes the compact-uniform-continuity finite-cover route through dyadic cells, common-refinement rows, mesh bounds, and UniformModulus handoff data.

Concretized in §6395: The compact radius-chain packet realizes the finite ProbeBundle lower-bound route before $\mathsf{UniformModulus}^\uparrow$ is consumed.

Concretized in §388: The compact finite-net and continuous-map modulus ledger is realized as a concrete NameCert chapter.

Concretized in §1493: Extreme-value packets realize compact finite-net and continuous-map uniform-modulus data as finite attainment NameCert rows.

Concretized in §6173: The shared Lipschitz compact-family packet realizes the compact-uniform-continuity route through finite compact-net, Lebesgue-number, and UniformModulus rows.

Concretized in §7502: LipschitzSpaceUp realizes the metric-domain real-valued Lipschitz packet consumed by uniform-modulus and compact-uniform-continuity routes.

Concretized in §935: The metric source-lock for compact-uniform-continuity is realized as a finite packet joining MetricSpace distance rows, compact finite-net rows, continuous pointwise-modulus rows, and RatUp fold ledgers.

Concretized in §596: The subordinate finite cover and pointwise-modulus packet is realized as the compact-uniform-continuity handoff before the rational lower-bound fold.

Concretized in §914: The compact finite-net selector is realized as the finite ProbeBundle lower-bound packet that turns pointwise compact-cover data into a tolerance-only uniform-modulus ledger.

Concretized in §648: Compact modulus covers realize the finite compact-net and pointwise-modulus source packet before the rational lower-bound fold and UniformModulusUp handoff.

Concretized in §926: Compact net modulus budgets realize the finite-net, pointwise-modulus, rational-fold, and uniform-modulus handoff as one NameCert packet.

Concretized in §1095: Compact cover shrinkage ledgers realize the finite cover-shrinkage boundary from compact nets and pointwise moduli to a uniform-modulus consumer packet.

Concretized in §1159: The final source-and-target triangle estimate packet is realized as a concrete NameCert carrier for the UniformModulusUp consumer handoff.

Concretized in §7260: The final source-and-target triangle estimate is realized as finite compact-net, pointwise-modulus, RatUp tolerance, MetricSpace, and UniformModulus NameCert data.

Concretized in §1257: The epsilon-third source and target triangle handoff is realized as a concrete NameCert carrier for compact-uniform-continuity.

Concretized in §7652: The real-completeness modulus ledger is realized as finite RegularCauchyCriterion, MetricCompletionFinite, RegSeqRat, StreamName, and RealUp NameCert data.

Concretized in §1228: Pointwise modulus ledgers are realized as finite center-radius ContinuousMap source packets consumed by compact finite-net selector chapters.

Concretized in §1253: Closed bounded interval uniform-modulus packets realize the Heine–Cantor interval route through compact dyadic covers and finite Lebesgue-number ledgers.

Concretized in §1302: The finite cover lower-bound ledger is realized as a NameCert packet joining CompactMetric finite nets, ContinuousMap pointwise radii, RatUp folds, and UniformModulus handoff rows.

Concretized in §1301: The finite Lebesgue-number radius ledger is realized as the compact-net selector handoff consumed by UniformModulusUp.

Concretized in §7821: The compact metric modulus realizer packages the finite-net, pointwise-modulus, rational lower-bound fold, metric-triangle, and Real tolerance rows as NameCert data.

Concretized in §2301: The compact-cover-to-positive-radius lemma route is realized as a theorem-facing packet between CompactMetric finite nets, ContinuousMap pointwise radii, FiniteLebesgueNumber ledgers, and UniformModulus handoff rows.

Concretized in §2188: The full finite-net, pointwise-modulus, rational-fold, and metric-triangle route is realized as one scheme-level NameCert packet.

Concretized in §8242: The source-and-target triangle chain is realized as finite CompactMetric, ContinuousMap, UniformModulus, Metric, DyadicRatCore, RegSeqRat, and RealUp rows.

Concretized in §6087: Finite approximate fixed-point packets realize the compact-uniform-continuity finite-modulus route as mesh, radius, displacement, compact-metric, finite-net, metric-ball, and Real-seal NameCert data.

Concretized in §5145: Finite window compactness realizes compact-uniform-continuity as explicit dyadic finite-net windows and uniform-modulus handoff.

Concretized in §6614: The compact finite-net and family pointwise-modulus route is realized as a finite UniformModulus router packet.

Concretized in §1580: Real sequence cluster-point packets realize the sequential compactness comparison route as finite bounded-source, subsequence-window, regular-readback, dyadic-tolerance, and Real-seal NameCert data.

Concretized in §2202: The compact finite-net and pointwise-modulus rational fold is realized as a concrete extractor NameCert packet.

Concretized in §2221: Compact-open function spaces realize the compact-uniform-continuity route as finite subbasic-window, continuous-map, uniform-limit, and RealUp seal NameCert data.

Concretized in §4235: Compact-open uniform convergence is realized as compact-open function-space, compact-metric probe, equicontinuity, uniform-modulus, ContinuousMap, Metric, Real, replay, provenance, and NameCert rows.

Concretized in §2253: Dyadic compactness extractors realize the finite net and dyadic radius packet behind the compact-uniform-continuity route.

Concretized in §2292: Compact uniform continuity modulus packets realize the finite compact-net, ContinuousMap graph, DyadicRatCore tolerance, and RealUp seal route as a concrete modulus object.

Concretized in §2299: Compact uniform modulus realizes the finite-net fold from compact metric and continuous-map rows into a uniform modulus ledger.

Concretized in §7102: Compact subspace modulus restriction realizes inherited compact-source, subspace-admission, continuous-map, and uniform-modulus rows as one finite consumer handoff.

Concretized in §4760: Finite ProbeBundle nets and rational lower-bound folds realize the compact-uniform-continuity construction as NameCert data.

Concretized in §6598: CompactMetricUniformContinuityUp realizes the finite-net, ContinuousMap, Metric, RatUp fold, DyadicRatCore, RegSeqRat, Real seal, and UniformModulus handoff route for the compact-uniform-continuity vision.

Concretized in §8394: The compact graph-modulus packet realizes the finite graph-evaluation, compact-net, pointwise-modulus, uniform-modulus, and Real-seal route.

Concretized in §3444: The Heine-Cantor packet realizes the compact-metric finite-net route from continuous local moduli to a uniform-modulus NameCert consumer.

Concretized in §1379: The Heine-Borel interval packet realizes closed bounded interval finite-net compactness through RealUp, StreamName, regular-Cauchy, and dyadic-net rows.

Concretized in §7351: The subordinate finite cover and pointwise-modulus handoff is realized as finite Compact, ContinuousMap, RatUp, and UniformModulus NameCert rows.

Concretized in §3444: The Heine-Cantor compact-modulus selector handoff realizes the finite CompactMetric/TotallyBounded/ContinuousMap/UniformModulus route as a human theorem packet over the compact-uniform-continuity vision.

Concretized in §2379: The compact fan-bar modulus boundary is realized as a finite NameCert packet for compact-uniform-continuity consumers.

Concretized in §3358: Equicontinuity finite-net selectors realize the compact finite-net and family-radius lower-bound handoff consumed by UniformModulusUp.

Concretized in §7330: The finite ProbeBundle rational-fold construction is realized as compact finite-net, continuous pointwise-modulus, rational lower-bound, dyadic tolerance, regular readback, and Real-seal NameCert data.

Concretized in §1822: The compact-metric product branch is realized through CompactMetric, Metric, finite-net product windows, and uniform-modulus routing before compact-uniform-continuity consumers read a product source.

Concretized in §3549: The compact-source triangle residual is realized as a finite NameCert packet joining CompactMetric, ContinuousMap, UniformModulus, and MetricTriangleUniformEstimate rows.

Concretized in §8513: Compact-uniform equioscillation is realized as compact finite-net, equicontinuity-family, Arzela–Ascoli, band-comparison, rational-readback, and UniformModulus NameCert data.

Concretized in §7344: Equicontinuous compact image packets realize compact-uniform-continuity through CompactMetric, ContinuousMap, EquicontinuityFamily, UniformModulus, StreamName, RegSeqRat, and RealUp rows.

Concretized in §8509: The finite lower-bound ledger realizes the ProbeBundle rational-radius fold used by the compact uniform-continuity construction.

Concretized in §1387: The finite compact-window monotone convergence route realizes the compact-uniform-continuity vision as a Dini-theorem NameCert packet.

Concretized in §6374: Finite Dini modulus packets realize compact finite-net, monotone pointwise descent, dyadic tolerance, ContinuousMap, and RealUp seal data as a concrete uniform-modulus NameCert route.

Concretized in §3630: Observable diameter decay realizes the compact finite-net and uniform-modulus diameter ledger as a concrete NameCert packet.

Concretized in §3698: The compact finite-net uniform-modulus route is realized as completion-extension consumer data.

Concretized in §3737: The compact finite-net and uniform-modulus route is realized as a finite oscillation-bound NameCert packet.

Concretized in §8170: Compact modulus composition is realized as finite compact-cover, pointwise-modulus, rational-fold, metric, and uniform-ledger NameCert data.

Concretized in §7416: The compact-uniform oscillation nerve packet realizes finite cover-nerve incidence, local oscillation control, rational lower-bound fold, and UniformModulus handoff rows.

Concretized in §5105: Finite interval covers realize the closed-bounded interval finite-cover route as BEDC NameCert data.

Concretized in §5124: Bishop interval covers realize the located finite interval-cover branch through dyadic cells, located intervals, StreamName, RegSeqRat, and RealUp rows.

Concretized in §3821: The radius-choice ledger realizes the compact finite-net and pointwise-modulus handoff as finite NameCert data.

Concretized in §6022: The compact finite-net triangle ledger realizes the metric-triangle handoff from pointwise moduli to a uniform-modulus row.

Concretized in §7474: The source-target triangle estimate is realized as finite compact-net, pointwise-modulus, source-triangle, target-triangle, and UniformModulus ledger rows.

Concretized in §8703: Compact uniform barycentric refinement is realized as dyadic subdivision, compact-metric source, interval-net alignment, StreamName, RegSeqRat, UniformModulus, and Real-seal NameCert data.

Concretized in §3853: Finite oscillation partitions realize the compact-uniform-continuity route from local oscillation control to a displayed uniform-modulus row.

Concretized in §4096: Finite ProbeBundle rational lower-bound folds realize the compact-uniform-modulus handoff.

Concretized in §6428: BaireMetric compact-window packets realize finite prefix-cylinder and dyadic-radius reads before compact uniform-continuity consumers inspect a modulus.

Concretized in §6250: The compact cover modulus packet realizes the finite-cover and lower-bound-fold route for compact uniform continuity.

Concretized in §8751: Finite open-cover number packets realize the compact-uniform-continuity finite-cover route through ProbeBundle cover rows and RatUp radius ledgers.

Concretized in §3857: Located sequence windows realize the StreamName, dyadic tolerance, RegSeqRat, LocatedReal, and Real seal route consumed by compact finite-net modulus arguments.

Concretized in §3884: finite endpoint-grid route for compact uniform-continuity interval consumers

Concretized in §3887: The compact finite-net and uniform-modulus route is realized as a finite diameter-decay packet for compact-continuous images.

Concretized in §3924: The compact continuous image total-bounded packet realizes the finite-net, continuous-map modulus, metric-distance, and RatUp fold route as a concrete NameCert carrier.

Concretized in §3957: The finite ProbeBundle realization route packages compact metric coverage, continuous pointwise moduli, RatUp lower-bound folds, and UniformModulusUp handoff rows as concrete NameCert data.

Concretized in §5866: The subordinate finite-cover carrier realizes the ProbeBundle modulus-fold route used by the compact-uniform-continuity theorem.

Concretized in §7456: The compact uniform-modulus certificate realizes the finite-net, pointwise-modulus, rational-fold, and metric-triangle handoff as NameCert data.

Concretized in §3937: Subordinate finite modulus covers are realized as finite CompactMetric, ContinuousMap, FiniteLebesgueNumber, RatUp fold, and UniformModulus NameCert packets.

Concretized in §4006: The compact-uniform radius-choice packet realizes the finite ProbeBundle lower-bound fold for tolerance-only uniform-modulus reads.

Concretized in §7295: The finite-subfamily Arzela–Ascoli packet realizes compact-metric finite nets, bounded-family rows, equicontinuity rows, and finite selector readback as a concrete compact-uniform-continuity consumer.

Concretized in §452: The finite Lebesgue-number packet realizes the positive-radius ledger used by the compact-uniform-continuity route.

Concretized in §4015: The compact finite Lebesgue-number packet realizes the finite cover-radius ledger used by the compact uniform-continuity proof.

Concretized in §9130: Finite prefix bar-modulus windows realize the compact-uniform-continuity route from finite prefix depth to tolerance-only uniform-modulus handoff rows.

Concretized in §5116: Uniform Lebesgue-number packets realize the compact finite-cover radius bound consumed by uniform-modulus arguments.

Concretized in §5203: Finite cover radius stability is realized as the compact finite-net radius transport packet consumed by the uniform-modulus fold.

Concretized in §7048: Finite product uniform-modulus packets realize the compact-product finite-net route from coordinate local moduli to a product tolerance-only UniformModulus handoff.

Concretized in §7021: The compact uniform-modulus envelope realizes the finite-net, local-modulus, rational-fold, and UniformModulus handoff route as NameCert data.

Concretized in §8933: The compact modulus mesh handoff is realized as finite compact-net, pointwise-modulus, dyadic tolerance, rational-fold, and uniform-modulus NameCert data.

Concretized in §4076: Equicontinuity is realized as the finite family-modulus NameCert surface consumed by compact-uniform and Arzela–Ascoli routes.

Concretized in §4118: ProbeBundle finite minimum folds realize the compact-uniform-continuity lower-bound fold over RatUp radius histories.

Concretized in §4297: The finite fan-bar modulus packet realizes the compact-uniform-continuity route from finite bars and local moduli to a uniform modulus ledger.

Concretized in §6744: The compact-uniform-continuity image packet realizes finite-net image transport through CompactMetricUp, TotallyBoundedUp, UniformContinuityUp, ContinuousMapUp, and RealUp rows.

Concretized in §6676: The compact grid modulus extraction packet realizes the finite compact-net, dyadic grid, rational lower-bound, and uniform-modulus route.

Concretized in §4371: Computable compact metric data supplies the effective finite-cover realizer surface for compact-uniform-continuity consumers.

Concretized in §6566: Finite ProbeBundle folds produce the compact-uniform modulus ledger.

Concretized in §4361: finite compact-net modulus transfer for uniform continuity

Concretized in §5872: Compact uniform modulus stability realizes the synchronized finite route from compact-net selectors, lower-bound folds, metric-triangle estimates, and compact-uniform rows to UniformModulusUp consumers.

Concretized in §7189: The compact uniform local-to-global modulus packet realizes the finite-cover route that turns pointwise compact-cover moduli into a tolerance-only uniform-modulus handoff.

Concretized in §7905: The compact uniform radius selector realizes the finite ProbeBundle lower-bound fold from compact metric and continuous-map radius rows into a UniformModulus handoff.

Concretized in §7604: The compact oscillation-modulus packet realizes finite compact-net oscillation bounds and uniform-modulus handoff rows as NameCert data.

Concretized in §5895: Compact uniform modulus composition is realized as a finite packet composing two tolerance-only modulus rows through continuous-map and metric-triangle ledgers.

Concretized in §8216: The finite-net source and target triangle replay route is realized as finite CompactMetric, ContinuousMap, UniformModulus, MetricSpace, RatUp, RegSeqRat, and RealUp NameCert rows.

Concretized in §9122: Finite selected-net data for compact uniform continuity.

Concretized in §7445: The finite dyadic radius ladder realizes compact finite-net, pointwise-radius, lower-bound fold, and UniformModulus rows.

Concretized in §8194: The compact modulus tail-fusion ledger realizes the finite compact-uniform-continuity route from compact modulus rows to shared Cauchy tail windows and Real-seal readback.

Concretized in §6383: The finite compact-cover modulus functional realizes the ProbeBundle cover, dyadic radius, local modulus, and uniform-modulus handoff route.

Concretized in §8173: The compact law failure-envelope packet realizes compact-uniform modulus data with displayed failure-surface rows.

Concretized in §6392: The compact dyadic grid-refinement packet realizes finite compact-net, dyadic-radius, and uniform-modulus handoff data.

Concretized in §3891: Uniform-continuity composition realizes the compact-uniform-continuity route's reusable composed-modulus handoff for downstream compact sources.

Concretized in §5558: The compact uniform-modulus pullback is realized as finite compact-net, pointwise-modulus, dyadic-tolerance, and Real-seal NameCert data.

Concretized in §5134: Regular Cauchy rounding projections realize the finite dyadic grid handoff used by compact-uniform-continuity routes.

Concretized in §7954: Compact uniform fan moduli realize the compact-uniform-continuity route as finite compact-net, pointwise-modulus, finite fan-bar, lower-bound, uniform-modulus, StreamName, RegSeqRat, and Real-seal NameCert data.

Concretized in §6293: The compact finite-net and finite oscillation-window route realizes the compact-uniform-continuity vision as one NameCert packet.

Concretized in §5189: The compact finite-net radius-chain route is realized as finite NameCert data for uniform-modulus consumers.

Concretized in §5115: Real-induced metric-ball entourage bases are realized as finite Real, metric, uniform, Cauchy-filter, StreamName, and RegSeqRat NameCert data.

Concretized in §7834: Real modulus normalization realizes the compact-uniform-continuity route through dyadic tolerance normalization, regular-rational readback, StreamName scheduling, and UniformModulus handoff rows.

Concretized in §6927: Finite cover bar moduli realize the compact-uniform-continuity route as finite bar-cover, local-modulus, dyadic-fold, and Real-seal NameCert data.

Concretized in §3503: The finite-subcover criterion packet realizes the compact finite-cover extraction route consumed by the uniform-modulus construction.

Concretized in §5954: The compact equicontinuity stability packet realizes the family-modulus refinement route through finite compact nets and uniform-modulus ledgers.

Concretized in §5964: The compact Lipschitz uniform handoff realizes the finite-net, Lipschitz graph, tolerance-fold, and UniformModulus consumer route as one NameCert packet.

Concretized in §5973: The finite modulus compact-uniform equicontinuity packet realizes the compact finite-net route from family moduli to a uniform equicontinuity ledger.

Concretized in §7648: The finite oscillation-envelope packet realizes compact-uniform-continuity through finite ProbeBundle radii, pointwise modulus rows, RatUp folds, and UniformModulus handoff data.

Concretized in §5986: The compact-uniform cover-chain packet realizes the subordinate finite-cover, radius-ledger, pointwise-modulus, and uniform-modulus handoff route as finite NameCert data.

Concretized in §6000: Finite subcover radius ledgers realize the compact finite-cover lower-bound route consumed by uniform-modulus arguments.

Concretized in §7369: The compact sequence-modulus packet realizes finite compact-window and pointwise-modulus data as a uniform-modulus NameCert route.

Concretized in §2299: Compact uniform modulus realizes the finite ProbeBundle lower-bound fold route through metric, compact, continuous, rational, uniform-modulus, and NameCert rows.

Concretized in §6113: Compact uniform radius-minimum stability realizes classifier transport for the finite ProbeBundle lower-bound fold before the uniform-modulus handoff is consumed.

Concretized in §6048: The compact finite-net and pointwise-modulus proof is realized as a finite NameCert carrier for uniform-modulus extraction.

Concretized in §7208: The compact equicontinuity envelope realizes finite compact-net, family modulus, rational fold, UniformModulus, Metric, and Real-seal rows as NameCert data.

Concretized in §6216: The diameter-decay packet realizes the finite compact-net route from pointwise moduli to a tolerance-only uniform-modulus ledger.

Concretized in §6079: Compact oscillation bar moduli realize the finite compact-net, pointwise oscillation, dyadic tolerance, and uniform-modulus handoff route.

Concretized in §6092: Gauge Lebesgue-number packets realize the Cousin-cover and finite-radius handoff used by compact-uniform and gauge-integration consumers.

Concretized in §6095: Equicontinuity radius choice realizes the finite family-radius handoff from compact nets and displayed equicontinuity rows to a uniform-modulus consumer.

Concretized in §6125: The finite subcover ledger realizes the ProbeBundle cover-selection handoff for compact-uniform-continuity consumers.

Concretized in §6399: Observable open-filter packets realize compact-uniform-continuity cover reads through SierpinskiSpace, Topology, StreamName, and UniformModulus rows.

Concretized in §7228: The finite compact-net and Cauchy-tail router realizes the vision's uniform-modulus route through StreamName, RegSeqRat, DyadicRatCore, and Real seal rows.

Concretized in §6143: The compact finite-net radius transport packet realizes the uniform-modulus handoff from pointwise continuity data.

Concretized in §6264: The compact uniform-Cauchy packet realizes the finite-net, uniform-tail, regular-readback, dyadic-tolerance, and Real-seal route as NameCert data.

Concretized in §7173: The compact oscillation selector realizes the finite net, pointwise-radius, rational lower-bound fold, and UniformModulus handoff as one NameCert packet.

Concretized in §6263: The finite cover nerve is realized as CompactMetric cover-member, overlap, rational-fold, and UniformModulus handoff rows.

Concretized in §7597: The finite cover nerve is realized as CompactMetric cover-member, overlap-incidence, nerve-face, rational-fold, and UniformModulus handoff rows.

Concretized in §8329: Finite cover multiplicity realizes overlap-depth and rational-fold rows for the compact-uniform-continuity route.

Concretized in §6274: The compact Cauchy equicontinuity packet realizes the finite-net, pointwise-modulus, RegSeqRat tolerance, and Real-seal route for compact-uniform-continuity consumers.

Concretized in §6466: Compact uniform tail selectors realize the finite ProbeBundle tail-radius handoff from compact nets and pointwise moduli to a tolerance-only uniform-modulus ledger.

Concretized in §1624: The real uniform structure packet realizes the entourage substrate consumed by compact-uniform-continuity finite-net moduli.

Concretized in §5129: Modulus-of-continuity packets realize the pointwise tolerance-to-precision ledger consumed before compact-uniform and completion routes read a uniform modulus.

Concretized in §6300: The finite oscillation-radius ledger realizes compact-cover radius, dyadic oscillation, and uniform-continuity handoff rows as NameCert data.

Concretized in §6326: The finite dyadic closed-interval subcover packet realizes the constructive Heine–Borel handoff used by compact-uniform-continuity interval consumers.

Concretized in §6409: Compact-uniform cover reads are realized as finite ProbeBundle, pointwise-modulus, rational lower-bound, uniform-modulus, transport, replay, provenance, and NameCert rows.

Concretized in §6733: Regular Cauchy product metrics realize the paired finite-window product-distance route used by compact-uniform-continuity consumers.

Concretized in §6499: The compact finite modulus gluing packet realizes the ProbeBundle finite-net, rational-fold, pointwise-continuity, and UniformModulus route as NameCert data.

Concretized in §6444: The compact uniform-modulus bridge realizes the finite ProbeBundle fold from CompactUp and ContinuousUp rows to a UniformModulusUp output.

Concretized in §6458: Pointwise equicontinuity transfer is realized as finite compact-net, family-modulus, RatUp fold, and UniformModulus handoff data.

Concretized in §6489: Compact tail selectors realize the finite tail-window handoff from CompactMetric, StreamName, RegSeqRat, UniformModulus, and RealUp rows.

Concretized in §7014: Finite-net mesh refinement realizes the compact-uniform-continuity route from compact finite nets to tolerance-only uniform-modulus data.

Concretized in §388: The finite-net ProbeBundle fold and rational lower-bound route realize the compact-uniform-continuity vision as a CompactUniformContinuityUp NameCert surface.

Concretized in §6522: The finite fan route realizes the compact-uniform-continuity vision as a concrete NameCert surface for uniform modulus extraction.

Concretized in §6528: The star-refinement modulus packet realizes the finite cover-refinement-to-uniform-modulus route.

Concretized in §6564: The Real-induced uniform Cauchy-filter completion packet realizes the finite RealUniformStructure, CauchyFilterCompletion, StreamName, RegSeqRat, DyadicRatCore, and RealUp handoff consumed by compact-uniform continuity routes.

Concretized in §6596: Effective compactness realizes the finite epsilon-net and tolerance-only modulus route used by compact uniform continuity.

Concretized in §6633: The compact finite-cover radius fold is realized as a finite ledger from CompactMetric and ContinuousMap rows to a UniformModulus handoff.

Concretized in §6641: The compact uniform-limit packet realizes compact-source uniform convergence as finite modulus, continuous-map, regular-rational readback, and Real-seal NameCert rows.

Concretized in §6716: The compact cover-chain uniform-modulus packet realizes the finite cover-refinement branch of the compact-uniform-continuity vision.

Concretized in §6750: The compact finite-net, pointwise-modulus, rational-fold, and UniformModulus handoff is realized as a finite NameCert packet.

Concretized in §8680: The finite ProbeBundle lattice, rational lower-bound fold, and UniformModulus handoff are realized as compact-modulus extraction NameCert data.

Concretized in §6824: The finite fan-bar selector realizes the compact pointwise-modulus and ProbeBundle fold route consumed by UniformModulusUp.

Concretized in §6807: The compact-uniform modulus ladder realizes the finite-cover-to-uniform-modulus route as NameCert data.

Concretized in §6846: The compact finite mesh and rational lower-bound fold realize the tolerance-only uniform-modulus handoff.

Concretized in §6874: Real-uniform separation is realized as finite uniform-entourage, Cauchy-filter, RegSeqRat, StreamName, and Real-seal NameCert data.

Concretized in §6769: The finite-net Heine-Cantor packet realizes the compact metric, continuous-map, finite lower-bound fold, and uniform-modulus route.

Concretized in §6912: Compact continuous image diameter is realized as finite compact-net, continuous-map, uniform-modulus, metric-distance, and Real-seal NameCert data.

Concretized in §6908: Compact function algebra packets realize the compact-uniform-continuity route as finite continuous-function, uniform-modulus, pointwise-operation, and sup-norm NameCert data.

Concretized in §6959: The compact finite lower-bound packet realizes the ProbeBundle fold used to extract a uniform modulus without an external selector.

Concretized in §3568: The finite equicontinuity modulus route realizes the compact-uniform-continuity finite-net and uniform-modulus handoff for Arzela–Ascoli consumers.

Concretized in §7004: The compact uniform Cauchy diagonal packet realizes compact finite-net, pointwise-modulus, diagonal-tail, RegSeqRat, and Real-seal rows as finite NameCert data.

Concretized in §7106: The compact uniform Cauchy family packet realizes the finite-net and shared-modulus family route as NameCert data.

Concretized in §7162: Finite fan compactness realizes the ProbeBundle branch of compact-uniform-continuity as finite cover and uniform-modulus NameCert data.

Concretized in §7176: Compact modulus oscillation routers realize the compact finite-net and pointwise oscillation route before a uniform-modulus consumer reads the tolerance-only ledger.

Concretized in §7194: The finite epsilon-third, compact-net, rational-fold, and metric-triangle budget is realized as UniformModulusErrorBudgetUp NameCert data.

Concretized in §7223: The compact finite-net oscillation envelope is realized as finite CompactUp, MetricUp, ContinuousUp, UniformModulusUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7238: The finite-net lower-bound fold normal form is realized as Metric, Compact, ContinuousMap, Rat, and UniformModulus NameCert data.

Concretized in §6352: ConstructiveDiniTheoremUp realizes the monotone compact-metric family route as finite Dini-modulus, UniformLimit, StreamName, RegSeqRat, and Real-seal NameCert data.

Concretized in §4213: DiniMonotoneCompactUniformUp realizes the monotone compact-metric family route as finite compact-source, continuous-map, monotone-ledger, finite-cover, uniform-modulus, UniformLimit, transport, replay, provenance, and NameCert data.

Concretized in §8011: The shared compact-net, pointwise-modulus, rational-fold, triangle-estimate, and UniformModulus route is realized as one finite coherence packet.

Concretized in §7429: The finite oscillation-mesh realization records compact finite-net rows, pointwise oscillation rows, rational mesh folding, dyadic tolerance conversion, and UniformModulus handoff data.

Concretized in §7452: The finite ProbeBundle cover fold realizes the compact-uniform-continuity route from compact finite covers to a tolerance-only uniform-modulus ledger.

Concretized in §7541: The compact uniform finite-cover fold realizes the finite ProbeBundle lower-bound fold of the compact-uniform-continuity theorem as concrete NameCert data.

Concretized in §7472: Uniform-continuity graph modulus packets realize the compact finite-net, pointwise-modulus, ProbeBundle fold, metric-triangle, and UniformModulus handoff as finite NameCert data.

Concretized in §5549: The compact uniform continuity ledger chapter realizes the finite compact-source, continuous-map, metric-triangle, and Real-seal route as concrete NameCert data.

Concretized in §7698: The finite weak-window maximal packet realizes a compact finite-cover route from local averages and dyadic thresholds to a tolerance-visible weak-level ledger.

Concretized in §7735: The compact Cauchy-net packet realizes the finite-net, Cauchy-tail, RegSeqRat, DyadicRatCore, and Real-seal route used by compact uniform continuity.

Concretized in §7769: Finite cover mesh chains realize the compact-uniform-continuity finite-cover route through ProbeBundle cover rows, dyadic mesh bounds, star-refinement transport, rational lower-bound folds, and UniformModulus handoff data.

Concretized in §7779: The located compactness modulus packet realizes the compact-uniform finite-net lower-bound branch as finite CompactMetric, MetricSpace, rational-radius, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and NameCert rows.

Concretized in §7781: The finite compact-net diagonal route realizes the equicontinuity-family branch through StreamName, RegSeqRat, and RealUp rows.

Concretized in §8073: The compact uniform diagonal-net packet realizes compact finite-net, pointwise-modulus, finite-radius, UniformModulus, StreamName, RegSeqRat, and Real-seal rows as concrete NameCert data.

Concretized in §7787: The finite oscillation-window packet realizes compact finite-net, continuous-map, RatUp fold, and RealUp tolerance rows as a uniform-modulus NameCert carrier.

Concretized in §7784: Finite dyadic Cantor prefix windows realize the compact-uniform-continuity finite-cover route as Bool/List prefix NameCert data.

Concretized in §7810: Finite modulus evaluation packets realize compact uniform continuity as compact metric, continuous-map, dyadic-ledger, finite-net, uniform-modulus, and Real-seal NameCert data.

Concretized in §7839: Finite modulus transfer packets realize the compact-uniform-continuity branch from finite compact-source probes and local modulus witnesses to a uniform-modulus handoff.

Concretized in §7869: The compact-uniform cover-membership packet realizes the finite bundled-center membership row required before cover-read radius folds are consumed.

Concretized in §7889: Compact-open Lipschitz modulus packets realize the compact-uniform-continuity branch through compact-open tests, Lipschitz bounds, ContinuousMap rows, MetricSpace rows, and finite modulus ledgers.

Concretized in §8038: The pointwise radius-chain packet realizes the compact finite-net and local-modulus route before the rational lower-bound fold.

Concretized in §6003: Finite Cauchy density nets realize the compact-uniform-continuity finite-net branch through dyadic cells, StreamName windows, RegSeqRat readback, and Real-seal rows.

Concretized in §8049: Finite epsilon-net chains realize the compact-uniform finite cover fold as displayed CompactMetric, TotallyBounded, DyadicRatCore, StreamName, RegSeqRat, Real, transport, provenance, and NameCert rows.

Concretized in §7942: The finite dyadic-net modulus surface realizes compact-uniform continuity through Arzela–Ascoli, compact source, continuous-map family, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Concretized in §8102: Stream diagonalization is realized as finite StreamName, RegSeqRat, DyadicRatCore, diagonal-window, and Real-seal NameCert data.

Concretized in §8118: The final source-and-target triangle closure is realized as finite compact-net, pointwise-modulus, dyadic tolerance, metric-triangle, UniformModulus, and Real-seal NameCert data.

Concretized in §3800: The finite oscillation partition realizes the compact finite-cover oscillation route through interval source, finite mesh, cell-bound readback, and UniformModulus handoff rows.

Concretized in §8167: The compact-uniform-modulus selector realizes the finite ProbeBundle lower-bound fold as a compact-net, pointwise-modulus, RatUp, MetricDistanceWitness, and UniformModulus NameCert packet.

Concretized in §5892: The effective compact-interval search packet realizes the compact-uniform-continuity finite-net and uniform-modulus route as bisection, value-comparison, Real-seal, and extremum-facing NameCert data.

Concretized in §5937: Compact interval modulus packets realize the interval-specialized finite-net and Real-seal route from compact interval data to uniform-modulus readback.

Concretized in §8353: The compact-uniform local-modulus nerve packet realizes the finite overlap-compatibility route from compact covers and pointwise moduli to a uniform-modulus handoff.

Concretized in §8356: Compact interval modulus extraction is realized as finite ProbeBundle cover rows, rational lower-bound fold data, and UniformModulus handoff for compact-uniform-continuity.

Concretized in §8374: The compact cover-order bound packet realizes bounded-overlap finite cover rows for compact uniform-modulus extraction.

Concretized in §8495: Compact oscillation trees realize finite cover-refinement and oscillation-window handoffs for compact uniform-modulus consumers.

Concretized in §8504: The finite modulus mesh realizes compact-uniform-continuity as compact-net cells, pointwise modulus rows, dyadic tolerances, and a UniformModulus handoff.

Concretized in §8526: The finite cover-nerve and rational lower-bound handoff realizes the compact-uniform-continuity route as concrete NameCert data.

Concretized in §8565: Stream diagonalization completion packets realize finite StreamName, RegSeqRat, DyadicRatCore, RealUp, and completion-operator rows for compact-uniform Cauchy consumers.

Concretized in §8584: Finite closed-interval nested-intersection packets realize the choice-free nested interval branch through NestedInterval, DyadicIntervalNet, RegSeqRat, StreamName, and RealUp rows.

Concretized in §5966: Dyadic interval nets realize the compact-uniform-continuity finite-subnet branch through dyadic cell rows, finite windows, and uniform-modulus handoff data.

Concretized in §8664: Finite compact-net, pointwise-modulus, RatUp lower-bound fold, and diagonal replay rows realize the compact uniform-modulus route.

Concretized in §8698: The Banach Picard orbit realizes the compact-uniform continuity route through finite contraction, Cauchy-tail, complete-metric, StreamName, RegSeqRat, and Real-seal rows.

Concretized in §8840: Finite bundled positive-radius meets realize the compact-uniform lower-bound fold.

Concretized in §8883: Uniform-tail Ascoli packets realize compact finite-net, equicontinuity, uniform-tail, and diagonal-subfamily NameCert data.

Concretized in §8902: The finite nerve packet realizes the compact finite-cover, local-modulus, RatUp lower-bound fold, and UniformModulus handoff route.

Concretized in §8929: The Bishop compact-image packet realizes the finite compact-source, continuous-map, uniform-modulus, image-witness, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and NameCert route.

Concretized in §8947: The Dirichlet-function packet realizes the rational-density versus irrational-boundary branch as finite RatUp, StreamName, RegSeqRat, RealUp, ContinuousMap, and NameCert data, marking a discontinuity boundary for compact-uniform-continuity consumers.

Concretized in §7576: Finite-net Arzela–Ascoli packets realize the compact-uniform-continuity branch through compact metric nets, equicontinuity rows, RegSeqRat readback, and finite diagonal ledger data.

Concretized in §9073: Compact interval Lipschitz envelopes realize finite mesh, Lipschitz bound, uniform-modulus, RegSeqRat, and Real-seal rows for the compact-uniform-continuity route.

Concretized in §9165: Compact oscillation budgets realize the compact-uniform-continuity route through finite compact-net cells, pointwise oscillation rows, dyadic tolerance, RegSeqRat readback, and RealUp seal data.

Concretized in §9170: Lebesgue number mesh extraction is realized as finite cover mesh, compact-net, local-radius, rational lower-bound, and uniform-modulus NameCert data.

Chapter 9238

Real Completeness via Regular Cauchy Sequences

This capstone records the conditional analytical target for the Phase Real and Completion milestone of section 9467.12. It treats the real naming certificate $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38) as complete in the Bishop sense only after the DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp certificate obligations are available at machine-checked strength. The chapter defines a regular Cauchy sequence over RegSeqRat as a sequence of regular rational sequences together with an explicit modulus of convergence, gives the intended limit modulus and RegSeqRat representative by diagonal extraction, and records the uniqueness criterion up to the standard psame equivalence on naming certificates. The construction is fully constructive as a proof target: every modulus is an explicit function on naturals and rationals, no real number is identified with an equivalence class, and the required Lean target is expected to use neither Classical.choice nor Quot.sound nor propext.

9238.1 Regular Cauchy sequences and their limits

Definition 9238.1 (Regular sequence over $\mathsf{Rat}^{\uparrow}$)**.** *A* regular sequence *over* $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ *is a function* $x\colon \mathbb{N} \to \mathbb{Q}$ *together with the explicit regularity witness*

$$\forall\, m, n \in \mathbb{N}, \quad |x_m - x_n| \;\le\; \tfrac{1}{m+1} + \tfrac{1}{n+1}.$$

The carrier RegSeqRat *of chapter 38 packages such pairs as a closed inductive over* $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ *data, with* RealClassifier *the* psame *closure that identifies two regular sequences whose pointwise difference shrinks at the regularity rate.*

Definition 9238.2 (Regular Cauchy sequence over RegSeqRat)**.** *A* regular Cauchy sequence *of regular rational sequences is a function* $X\colon \mathbb{N} \to \mathsf{RegSeqRat}$ *together with an explicit modulus* $\mu\colon \mathbb{Q}_{>0} \to \mathbb{N}$ *such that, for every* $\varepsilon \in \mathbb{Q}_{>0}$ *and every* $m, n \ge \mu(\varepsilon)$*, the regular rational sequence* $X_m - X_n$ *(computed pointwise on representatives) lies inside the* ε*-ball at* 0 *in the sense of* $\mathsf{NameCert}_{\mathsf{RegSeqRat}}$.

Definition 9238.3 (Diagonal limit representative)**.** *The* diagonal limit representative *of* (X, μ) *is the rational sequence* $L\colon \mathbb{N} \to \mathbb{Q}$ *given by*

$$L_n \;=\; \big(X_{\mu(1/(n+1))}\big)_{\mu(1/(n+1))+n+1},$$

that is, the value at index $\mu(1/(n+1)) + n + 1$ of the $\mu(1/(n+1))$-th member of the Cauchy sequence. The construction is primitive recursion on n together with $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *arithmetic on the indices.*

Theorem 9238.4 (Diagonal limit is a regular sequence)**.** *The diagonal limit representative L of Definition 9238.3 is a regular sequence in the sense of Definition 9238.1: for every $m, n \in \mathbb{N}$,*

$$|L_m - L_n| \;\leq\; \tfrac{1}{m+1} + \tfrac{1}{n+1}.$$

Sketch. Fix $m \leq n$. Bound $|L_m - L_n|$ by triangle inequality through three intermediate points: the value of $X_{\mu(1/(m+1))}$ at the diagonal index, the value of $X_{\mu(1/(n+1))}$ at the same index, and the value at L_n's diagonal index. The first and third differences are controlled by the regularity of the constituent regular sequences (Definition 9238.1); the middle difference is controlled by the Cauchy modulus μ (Definition 9238.2). Each bound is at most $1/(m+1)+1/(n+1)$ after a one-line rational arithmetic step. □

Concretized in §1205: The diagonal-limit representative is realized as a finite BHist packet selecting the modulus row, StreamName window, RegSeqRat readback, dyadic ledger, and RealUp seal without a hidden limit choice.

Concretized in §1204: Tail-compatible Cauchy sections realize the diagonal-limit finite-window route as a BEDC NameCert packet.

Concretized in §1211: Dyadic tolerance triangle ledgers realize the three-leg diagonal-regularity estimate as finite DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1158: Regular Cauchy regularity witnesses realize the diagonal-regularity proof row as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert data.

Concretized in §1215: Finite real sections realize a single precision request, StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, RealUp seal, and local transport/provenance NameCert packet.

Concretized in §1230: Regular Cauchy tail certificates realize the shared tail-window witness as finite DyadicRatCore, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §3918: Cauchy window stability bridges realize shared finite tail-window transport from StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §1242: The diagonal-regularity theorem row is realized as a finite seal packet joining DiagonalLimit, StreamName, RegSeqRat, DyadicRatCore, triangle-bound, and RealUp handoff data.

Concretized in §1518: Fast Cauchy subsequences realize the finite cofinal-tail readback route from inspected Cauchy moduli into RegSeqRat and RealUp seal rows.

Concretized in §1245: Diagonal regularity seals realize the three-leg diagonal-regularity theorem as finite representative, endpoint-leg, middle-modulus, dyadic-tolerance, window-factorization, and Real-seal NameCert rows.

Concretized in §1255: Cofinal tail equivalences realize paired cofinal tail routes as a finite StreamName, RegSeqRat, DyadicRatCore, RealTailAgreementSeal, hsame, Cont, Pkg, and NameCert packet.

Concretized in §1281: Eventual tail-agreement packets realize the shared-tail classifier route for Real-completion consumers without quotient streams or selected limits.

Concretized in §1335: The modulus-indexed diagonal readback is realized as a finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert packet.

Concretized in §1338: The four-face exit classifier realizes the StreamName, DyadicRatCore, RegSeqRat, and RealUp exit route as one finite classifier packet.

Concretized in §2361: The Isbell tight-span packet supplies a finite injective-envelope boundary for metric completion routes.

Concretized in §3383: Tail-modulus limit handoffs realize the diagonal regularity proof route as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §3858: Cauchy modulus extractors realize the explicit stage-selection route from EffectiveCauchySequence and FastCauchyModulus evidence through StreamName, DyadicRatCore, RegSeqRat, and RealUp seal rows.

Concretized in §4241: The regular Cauchy diagonal seal packages the finite-window handoff from local regular sections to the terminal RealUp seal.

Concretized in §4290: Real completeness witnesses realize the four-face StreamName, DyadicRatCore, RegSeqRat, and RealUp seal route as finite NameCert data.

Concretized in §5548: Finite nested interval data yields an explicit modulus route into regular Cauchy real names.

Concretized in §5997: Rational open interval bases realize the real-completeness finite observation route through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Concretized in §6183: Regular Cauchy subnet selectors realize the cofinal finite-window extraction route for diagonal-limit and real-completeness consumers.

Concretized in §6941: Regular Cauchy tail-window selectors realize the finite cofinal-window route for diagonal-limit and real-completion consumers.

Concretized in §6232: Regular Cauchy saturation records the finite tail-budget closure route for the Real-completion vision.

Concretized in §6389: Finite-window RegSeqRat handoffs realize the StreamName, DyadicRatCore, RegSeqRat, and RealUp seal route as one finite NameCert packet.

Concretized in §6690: Regular Cauchy diagonal exchanges realize paired diagonal-window choices as finite StreamName, DyadicRatCore, RegSeqRat, and RealUp seal data.

Concretized in §6713: Regular Cauchy filter seals realize the tail-filter route into finite StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §7119: Tail-window filters realize cofinal finite tail windows for the Real completeness route through StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp seal rows.

Concretized in §7201: Tail-window modulus synchronization realizes the regular-Cauchy tail request as a finite modulus, TailCofinalityBudget, StreamName, DyadicRatCore, RegSeqRat, and RealUp route.

Concretized in §7324: RegularCauchyTailLimitUp realizes the regular-Cauchy tail-window limit route as finite StreamName, RegSeqRat, DyadicRatCore, and Real-seal NameCert data.

Concretized in §7513: StreamName tail-window routers realize the regular-Cauchy diagonal-limit route as finite precision, tail, tolerance, readback, and Real-seal NameCert rows.

Concretized in §7740: Cauchy bicompletion is realized as finite paired Cauchy-window, RegSeqRat readback, RealUp route, transport, replay, provenance, and NameCert rows.

Concretized in §7848: Regular Cauchy tail pasting realizes shared-tail pasting before RealUp seal reads as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and NameCert rows.

Concretized in §7861: Cantor intersection windows realize nested closed interval finite-window data as a Real and Completion NameCert handoff.

Concretized in §7903: Real completion diagonal budgets realize the finite precision-to-window-to-RegSeqRat-to-RealUp seal route used by the diagonal-limit construction.

Concretized in §7948: Dyadic contraction moduli realize the three-leg diagonal regularity estimate through finite DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Concretized in §8145: Cauchy tail-comparison functors realize regular-Cauchy comparison transport and composition as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8687: Regular Cauchy eventual windows realize the finite tail-window route for diagonal-limit and Real-completion consumers.

Concretized in §8729: Regular Cauchy limit witnesses realize the diagonal-limit route as finite modulus, DyadicRatCore, StreamName, RegSeqRat, RealUp, BishopRegularCauchyCompletion, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8767: Regular Cauchy window pullbacks realize a Real-completion precision request as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8895: Dyadic cover moduli realize the finite mesh control used by the regular-Cauchy limit route.

Concretized in §9147: Finite modulus extraction for the regular-Cauchy diagonal-limit route.

Concretized in §9167: Regular Cauchy diagonal-limit seals realize the diagonal representative as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

9238.2 The limit theorem

Theorem 9238.5 (Real completeness proof target over $\mathsf{RegSeqRat}$)**.** *Assume that the DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp certificate obligations of Phase Real and Completion are supplied at machine-checked strength. For every regular Cauchy sequence (X, μ) over* $\mathsf{RegSeqRat}$*, the diagonal representative L of Definition 9238.3 is the Bishop-style limit candidate: it is a regular sequence by Definition 9238.4, and the assignment $(X, \mu) \mapsto [L]$ into the supplied* $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$ *satisfies*

$$\forall\, n \in \mathbb{N}, \quad |X_n - L| \;\leq\; \tfrac{2}{n+1}$$

in the metric of $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$*. The limit candidate is unique up to the* psame*-equivalence underlying* $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$*: if* L' *is another regular sequence with the same convergence bound, then* $L \,\mathsf{psame}\, L'$ *as elements of* $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$*.*

Sketch. Under the stated certificate obligations, the convergence bound is an unfolding of the supplied definitions. At scale n, the diagonal index is at least $\mu(1/(n+1))$, so the regular Cauchy modulus controls X_n against $X_{\mu(1/(n+1))}$, and the diagonal-index regularity (Definition 9238.4) controls $X_{\mu(1/(n+1))}$ against L.

The uniqueness assertion is the psame closure on the supplied $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$ applied to the difference $L - L'$. Its pointwise size is bounded by $4/(n+1)$ for every n, hence shrinks at the regularity rate, hence is identified with 0 by the supplied RealClassifier. □

Capstone.

Definition 9238.6 (Stationary rational regular Cauchy input)**.** *For a history* $d : \mathsf{Hist}$ *with* $\mathsf{RatHistoryCarrier}(d)$*, the stationary rational regular Cauchy input* X_d *is the pair consisting of the constant sequence*

$$X_d(n) = (\mathsf{RatConstStream}_d, \mathsf{E1}(d))$$

for every $n : \mathbb{N}$*, and the constant modulus* $\mu_d(\varepsilon) = 0$ *for every positive rational tolerance* ε*. Each member is the constant rational RealUp name represented by the StreamNameUp seed* $\mathsf{RatConstStream}_d$ *together with the e-one RealUp seal* $\mathsf{E1}(d)$*.*

Theorem 9238.7 (Stationary rational diagonal agrees with the rational embedding)**.** *If* $\mathsf{RatHistoryCarrier}(d)$*, then the input* X_d *of Definition 9238.6 satisfies the regular Cauchy condition of Definition 9238.2. Its diagonal representative* L_d *from Definition 9238.3 is classified by the RealUp classifier with the constant rational embedding of* d *at* $\mathsf{E1}(d)$*, namely the name represented by* $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$*.*

Sketch. The carrier assumption on d gives RatUp classifier reflexivity by the semantic certificate Definition 37.90. Applying the StreamNameUp constant witness Definition 126.48 packages the constant stream $\mathsf{RatConstStream}_d$ as a valid stream name and gives its self-classifier at every certified unary observation.

For the Cauchy condition, every pair of members $X_d(m)$ and $X_d(n)$ has the same hidden rational seed. The zero modulus is therefore admissible: after unfolding the regular Cauchy predicate, the required endpoint comparison is exactly the self-classifier just obtained, transported through StreamNameUp constant point exactness Definition 126.55.

For the diagonal representative, the extraction formula always selects a member of X_d and then reads a point of the same constant stream. Thus every diagonal value is the seed d, independently of the outer index and independently of the selected diagonal bound.

Finally, Definition 38.31 identifies the pair $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$ with the RealUp constant rational embedding. The previous paragraph shows that L_d has exactly the same constant seed at every diagonal read, so the RealUp classifier relates L_d to that embedding. □

Concretized in §364: Stationary rational regular Cauchy inputs are recorded as a concrete RealUp handoff certificate.

Concretized in §673: Limit selector packets realize the diagonal-limit selector as finite modulus, window, RegSeqRat, RealUp seal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6644: The diagonal-limit witness route is realized as finite modulus, window, RegSeqRat, DyadicRatCore, RealUp, uniqueness, hsame, Cont, Pkg, and NameCert rows.

Concretized in §701: Stationary embedding classifier packets realize the RatUp-to-RealUp classifier correspondence for stationary rational diagonals.

Concretized in §1351: The explicit diagonal convergence bound is realized as a finite NameCert packet linking the diagonal representative, dyadic tolerance ledger, RegSeqRat readback, and RealUp seal.

Concretized in §8227: Diagonal limit modulus certificates realize the finite threshold, StreamName, RegSeqRat, DyadicRatCore, RealUp, and CauchyLimitSeal route used by the limit theorem.

Theorem 9238.8 (Stationary rational embedding classifier correspondence)**.** *For RatUp histories $d, e : \mathsf{Hist}$ with $\mathsf{RatHistoryCarrier}(d)$ and $\mathsf{RatHistoryCarrier}(e)$, let X_d and X_e be the stationary rational regular Cauchy inputs of Definition 9238.6, and let L_d and L_e be their diagonal representatives from Definition 9238.3. Then the RealUp classifier relates the two stationary diagonal representatives exactly when the RatUp classifier relates their hidden rational seeds:*

$$\mathsf{RealClassifier}(L_d, L_e) \iff \mathsf{RatHistoryClassifier}(d, e).$$

Sketch. Apply Definition 9238.7 separately to d and to e. This identifies L_d with the constant rational RealUp name represented by $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$, and identifies L_e with the corresponding constant name represented by $(\mathsf{RatConstStream}_e, \mathsf{E1}(e))$.

For the reverse direction, assume $\mathsf{RatHistoryClassifier}(d, e)$. The constant point exactness theorem Definition 126.55 transports this classifier to the constant StreamNameUp row, and Definition 38.31 transports the same row to the e-one RealUp constant classifier. The two diagonal agreement rows just obtained then transport the e-one constant classifier to $\mathsf{RealClassifier}(L_d, L_e)$.

For the forward direction, assume $\mathsf{RealClassifier}(L_d, L_e)$. Transport this classifier across the two diagonal agreement rows to obtain the e-one constant classifier $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e))$. Applying Definition 38.11 reflects that classifier back to $\mathsf{RatHistoryClassifier}(d, e)$; equivalently, Definition 126.55 reads the intervening constant StreamNameUp classifier back as the same RatUp seed classifier. □

Concretized in §362: Stationary rational diagonal packets are realized as a concrete Real and Completion NameCert surface.

Concretized in §411: Regular Cauchy comparison packets realize the finite comparison layer used by the Real completeness proof target.

Concretized in §644: Regular Cauchy mesh packets realize finite mesh-indexed regular-rational observations for the Real completeness handoff.

Concretized in §637: Real approximation envelopes realize the finite observation packet from DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp into a reusable Real and Completion handoff.

Concretized in §703: Cauchy inverse budgets realize nonzero-apart regular-Cauchy reciprocal windows as finite DyadicRatCore, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6251: Real reciprocal packets realize nonzero-apart RealUp reciprocal reads through finite Cauchy inverse budgets, RegSeqRat handoff, and terminal Real seal rows.

Concretized in §886: Cauchy seal-budget synchronizers realize the Real completeness route as one finite selector, observation-budget, shared-threshold, and seal NameCert packet.

Concretized in §898: Cauchy tail-witness envelopes realize the Real-completeness diagonal tail route as a finite StreamName, finite-window, CauchyTailComparison, RealTailAgreementSeal, and RealUp NameCert packet.

Concretized in §909: Regular Cauchy diagonal meets realize the shared-threshold diagonal tail route as a finite TailMeet, CauchyModulusMeet, CauchyTailWitnessEnvelope, and RealDiagonalWindowCoverage packet.

Concretized in §900: Real completion seal classifiers realize finite completion-seal equivalence as displayed RealTailAgreementSealUp, CauchyLimitSealUp, CauchySealBudgetSynchronizerUp, RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §902: Cauchy witness ledgers realize the Real completeness route as finite diagonal-budget, observation-budget, seal-synchronizer, and completion-seal NameCert rows.

Concretized in §913: Budgeted real seal routes realize the Real-completion budget, TailMeet threshold, observation-budget, and CauchyLimitSeal handoff as one finite NameCert packet.

Concretized in §921: Finite tail-fiber schedules realize the Real-completeness tail route as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §929: Cauchy rate budgets realize finite convergence-rate control for StreamName, RegSeqRat, DyadicRatCore, RealUp, and observation-budget seal routes.

Concretized in §981: Dyadic window seals realize finite dyadic precision, StreamName window, RegSeqRat readback, observation-budget, synchronizer, and RealUp seal rows for the Real and Completion route.

Concretized in §1276: Finite limit-seal envelopes realize the Real completeness limit theorem as one finite StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, RealUp, hsame, Cont, Pkg, and NameCert handoff.

Concretized in §1278: Diagonal-index cofinality realizes the finite modulus-threshold row used by the Real completeness limit proof.

Concretized in §1342: Cauchy seal interfaces realize the common finite seal handoff joining CauchyLimitSeal, seal-classifier, budget-synchronizer, StreamName, RegSeqRat, DyadicRatCore, and RealUp rows.

Concretized in §3701: The diagonal limit candidate and uniqueness classifier are realized as a finite CauchyLimitSeal, RegSeqRat, StreamName, DyadicRatCore, and RealUp NameCert packet.

Concretized in §6809: The modulus envelope realizes the finite regular-Cauchy limit handoff without asserting ambient Real completeness.

The ten-field fit is stationary rational embedding classifier reflection: traditional target constant rational diagonals inside RealUp; BEDC source RatUp plus StreamNameUp plus RealUp; carrier stationary regular Cauchy inputs and their diagonal representatives; classifier the displayed RealUp/RatUp equivalence; stability supplied by diagonal agreement and constant endpoint transport; ledger the two hidden rational seeds and their e-one seals; core theorem Definition 9238.8; exactness the displayed if-and-only-if; standard bridge constant rational RealUp names; status ProofTarget.

Theorem 9238.9 (Stationary rational finite-window classifier bridge)**.** *For histories* $d, e, a, w : \mathsf{Hist}$ *with* $\mathsf{UnaryHistory}(a)$ *and* $\mathsf{UnaryHistory}(w)$*, the stationary rational point classifier, the constant*

StreamNameUp classifier, the RealUp finite-window classifier, and the e-one RealUp constant classifier are equivalent:

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e).$$

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \iff \mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConst}$$

$$\mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e, a, w) \iff \mathsf{RealConstantHistoryClassifier}($$

Sketch. The RatUp and StreamNameUp rows are the constant-stream point exactness row of Definition 126.55: a classifier between the hidden rational seeds is the same data as the pointwise classifier between the two constant observation maps.

For the forward finite-window direction, read the constant StreamNameUp classifier as a RealUp unary-observation classifier by Definition 38.27. Apply the Real unary finite-window coverage theorem Definition 38.155 with the displayed witnesses $\mathsf{UnaryHistory}(a)$ and $\mathsf{UnaryHistory}(w)$.

For the reverse finite-window direction, evaluate the window at offset e. The witness $\mathsf{UnaryOffsetLe}(\mathsf{e}, w)$ is supplied by unary emptiness, the displayed unary bound on w, and the continuation unit law. The constant stream definitions erase the anchor and this offset, so the selected endpoint classifier is $\mathsf{RatHistoryClassifier}(d, e)$; applying Definition 126.55 returns the StreamNameUp row.

Finally, Definition 38.11 identifies the rational point classifier with $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e))$, and Definition 38.31 packages that identification as the constant rational RealUp embedding certificate. □

The bridge records the ten-field fit: traditional target stationary rational finite windows; BEDC source RatUp plus StreamNameUp plus RealUp; carrier stationary rational observation maps; classifier the four equivalent rows; stability inherited from RatUp classifier fields and unary-window coverage; ledger hidden rational seed plus anchor and bound; core theorem Definition 9238.9; exactness the displayed equivalence; standard bridge constant rational real names; status $\mathsf{ProofTarget}$.

Theorem 9238.10 (Stationary rational diagonal finite-window classifier correspondence)**.** *For RatUp histories $d, e : \mathsf{Hist}$ with $\mathsf{RatHistoryCarrier}(d)$ and $\mathsf{RatHistoryCarrier}(e)$, and for unary window endpoints $a, w : \mathsf{Hist}$ with $\mathsf{UnaryHistory}(a)$ and $\mathsf{UnaryHistory}(w)$, let X_d and X_e be the stationary rational regular Cauchy inputs of Definition 9238.6, and let L_d and L_e be their diagonal representatives from Definition 9238.3. Then the stationary diagonal RealUp classifier, the finite-window classifier on the constant rational streams, and the RatUp seed classifier are equivalent:*

$$\mathsf{RealClassifier}(L_d, L_e) \iff \mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e, a, w) \iff$$

Sketch. First apply Definition 9238.8 to the displayed stationary inputs. This identifies $\mathsf{RealClassifier}(L_d, L_e)$ exactly with the seed classifier $\mathsf{RatHistoryClassifier}(d, e)$.

Next apply Definition 9238.9 with the same seeds and unary endpoints. Its RatUp-to-StreamNameUp row and StreamNameUp-to-window row identify $\mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStrea}$ with the same seed classifier.

Both displayed identifications land on $\mathsf{RatHistoryClassifier}(d, e)$. Composing the first identification with the inverse of the second gives the RealUp-to-window equivalence, while each classifier also remains equivalent to the common RatUp seed classifier. This assembles the stated three-way correspondence. □

Theorem 9238.11 (Stationary rational selected tail seed exactness)**.** *For histories $d, e, a, w, k, d_0, e_0 : \mathsf{Hist}$, assume*

$$\mathsf{RatHistoryClassifier}(d, e), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w),$$

$$\mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

Then the stationary constant streams form a RealUp finite-window classifier,

$$\mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e, a, w),$$

and the selected endpoint e-one tail class is exactly the original RatUp seed-tail class:

$$\begin{aligned}&\forall u, v : \mathsf{Hist},\ \big(\mathsf{hsame}(\mathsf{RatConstStream}_d(\mathsf{append}(a,k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(\mathsf{RatConstStream}_e(\mathsf{append}(a,k)), \mathsf{E1}(v))\big)\\ &\quad\Longleftrightarrow\\ &\qquad\big(\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(d_0, u) \wedge \mathsf{hsame}(e_0, v)\big).\end{aligned}$$

Sketch. The displayed RatUp classifier and the two e-one seed displays are first read through the RatUp e-one tail readback row. This recovers $\mathsf{UnaryHistory}(d_0)$, $\mathsf{UnaryHistory}(e_0)$, and the seed-tail comparison $\mathsf{hsame}(d_0, e_0)$ inside the original rational classifier class.

Apply StreamNameUp constant point exactness Definition 126.55 to pass from the RatUp classifier on d, e to the classifier between $\mathsf{RatConstStream}_d$ and $\mathsf{RatConstStream}_e$. Then use Definition 9238.9 with the displayed unary anchor and bound to obtain the RealUp finite-window classifier row.

For the forward direction of the displayed exactness, apply Definition 38.584 to the constant streams, the StreamNameUp classifier, the unary anchor and bound, and the selected offset. Its canonical selected tails are the displayed seed tails d_0, e_0, because the constant streams erase $\mathsf{append}(a, k)$ and the RatUp e-one readback identifies the selected endpoint displays with the seed displays.

For the reverse direction, assume $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, $\mathsf{hsame}(d_0, u)$, and $\mathsf{hsame}(e_0, v)$. Apply BEDC.FKernel.Hist.hsame_e1_congr to the two tail comparisons, then compose by BEDC.FKernel.Hist.hsame_ with $\mathsf{hsame}(d, \mathsf{E1}(d_0))$ and $\mathsf{hsame}(e, \mathsf{E1}(e_0))$. Since each constant stream returns its seed at $\mathsf{append}(a, k)$, the two composites are exactly the selected endpoint displays for u and v. □

Fit. Traditional target: stationary rational selected finite-window tail exactness. BEDC source: $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow}$ plus FKernel e-one congruence and transitivity. Carrier: selected constant-stream endpoint tails. Classifier: RealUnaryStreamWindowClassifier obtained from the stationary rational classifier chain. Stability: endpoint hsame transport from the seed displays to selected e-one tails. Ledger: RatUp classifier, seed e-one displays, unary anchor, unary bound, selected offset, and selected endpoint displays. Core theorem: Definition 9238.11. Exactness: the displayed if-and-only-if identifies selected finite-window tails with the original RatUp seed-tail class. Standard bridge: constant rational RealUp observation windows. Downstream use: RegSeqRat stationary observation canonicalization. Status: ProofTarget.

Theorem 9238.12 (Stationary rational finite-window concatenation)**.** *For histories* $d, e, a_L, w_L, a_R, w_R, d_0, e_0 : \mathsf{Hist}$*, indices* n, m, r*, and a unary observation schedule* $\eta : \mathbb{N} \to \mathsf{Hist}$*, define the stationary rational RealUp observation streams*

$$C_d(i) = \mathsf{RatConstStream}_d(\eta(i)), \quad C_e(i) = \mathsf{RatConstStream}_e(\eta(i)).$$

Assume

$$\mathsf{RatHistoryClassifier}(d, e), \quad \mathsf{UnaryHistory}(a_L), \quad \mathsf{UnaryHistory}(w_L), \quad \mathsf{UnaryHistory}(a_R), \quad \mathsf{UnaryHistory}(w_R),$$

and the seed-tail displays

$$\mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

If the left block $[n, n+m]$ is named by unary offsets below w_L from anchor a_L, and the right adjacent block $[n+(m+1), n+(m+1)+r]$ is named by unary offsets below w_R from anchor a_R, then the joined natural-indexed finite window is classified:

$$\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, m+1+r).$$

Moreover, for every selected offset $k \le m+1+r$, the constructive split $k \le m$ or $k = m+1+j$ with $j \le r$ selects the corresponding left or right unary source offset, and the selected endpoint in the joined window has exactly the original RatUp seed-tail class:

$$\begin{aligned}&\forall u, v : \mathsf{Hist},\ \big(\mathsf{hsame}(C_d(n+k), \mathsf{E1}(u)) \wedge \mathsf{hsame}(C_e(n+k), \mathsf{E1}(v))\big)\\ &\iff\\ &\qquad \big(\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(d_0, u) \wedge \mathsf{hsame}(e_0, v)\big).\end{aligned}$$

Sketch. Apply Definition 9238.9 with the common $\mathsf{RatHistoryClassifier}(d, e)$, first at (a_L, w_L) and then at (a_R, w_R). This gives the two stationary $\mathsf{RealUnaryStreamWindowClassifier}$ rows for the left and right unary source windows.

Unfold the scheduled constant streams C_d and C_e. On the left block, each endpoint $C_d(n+k), C_e(n+k)$ is the same pair of hidden seeds d, e read through $\mathsf{RatConstStream}$; the left unary window therefore supplies the natural window $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, m)$. The same argument on the right block gives $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n+(m+1), r)$.

Apply Definition 38.182 to these two adjacent natural-indexed windows. The result is the joined classifier $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, m+1+r)$.

For a selected offset $k \le m+1+r$, split constructively between $k \le m$ and $k = m+1+j$ with $j \le r$. In the left case, use the corresponding unary source offset below w_L and apply Definition 9238.11 at (a_L, w_L). In the right case, apply the same theorem at (a_R, w_R) with the right source offset.

In both cases the stationary streams erase the selected observation schedule, so the endpoint displays are the same seed displays $\mathsf{hsame}(d, \mathsf{E1}(d_0))$ and $\mathsf{hsame}(e, \mathsf{E1}(e_0))$. The two cases therefore return the same displayed tail-class equivalence, and the offset split covers the whole joined window. □

The ten-field fit is: traditional target finite stationary rational window pasting; BEDC source RatUp plus StreamNameUp plus RealUp plus Real stream window concatenation; carrier stationary constant streams; classifier joined RealUp finite-window classifier; stability by stationary bridge exactness, bounded-offset splitting, and adjacent window concatenation; ledger hidden rational seed, two bounds, join index, and selected offset; core theorem Definition 9238.12; exactness selected endpoint tails remain the RatUp seed class; standard bridge RegSeqRat finite observation window pasting; downstream use RegSeqRat stationary observation canonicalization and the Phase Real and Completion diagonal-limit capstone. Status: $\mathsf{ProofTarget}$.

Theorem 9238.13 (Stationary rational finite-partition fold)**.** *For histories $d, e, d_0, e_0 : \mathsf{Hist}$, an anchor n, a total bound w, and a unary observation schedule $\eta : \mathbb{N} \to \mathsf{Hist}$, define the stationary rational RealUp observation streams*

$$C_d(i) = \mathsf{RatConstStream}_d(\eta(i)), \quad C_e(i) = \mathsf{RatConstStream}_e(\eta(i)).$$

Let P be a certified finite partition spine over (n, w) with segment anchors and radii (n_i, r_i), and suppose that P records the unary source-anchor, source-bound, and selected-offset witnesses for every segment. Assume

$$\mathsf{RatHistoryClassifier}(d, e), \quad \mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

Then the total stationary window classifier is exactly the finite conjunction of the segment classifiers named by P:

$$\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w) \iff \bigwedge_{(n_i,r_i)\in P} \mathsf{RealStreamWindowClassifier}(C_d, C_e, n_i, r_i).$$

Moreover, for every segment (n_i, r_i) named by P and every selected offset $k \le r_i$ whose unary source witness is recorded by P, the selected endpoint has the original displayed seed-tail class:

$$\begin{aligned}&\forall u, v : \mathsf{Hist},\ \big(\mathsf{hsame}(C_d(n_i+k), \mathsf{E1}(u)) \wedge \mathsf{hsame}(C_e(n_i+k), \mathsf{E1}(v))\big)\\ &\iff\\ &\qquad \big(\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(d_0, u) \wedge \mathsf{hsame}(e_0, v)\big).\end{aligned}$$

Sketch. First treat each segment named by P as an ordinary stationary rational finite window. The spine ledger supplies the unary source anchor, source bound, and selected offsets for that segment, while the common $\mathsf{RatHistoryClassifier}(d, e)$ row is shared by every segment.

Apply Definition 9238.11 segmentwise. This gives, for every selected offset in every segment, the displayed endpoint-tail equivalence with the seed tails d_0, e_0.

Next apply Definition 9238.9 to each segment ledger. The stationary streams erase the observation schedule η, so the same RatUp classifier produces every segment classifier $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n$

It remains to fold the segment classifiers along the certified spine. Apply Definition 38.308 to the streams C_d, C_e and to P. In the singleton case this is the existing stationary finite-window bridge. In the successor case the induction step uses the spine continuation ledger to concatenate the predecessor window with the final segment, exactly as in the finite partition fold. □

The ten-field fit is: traditional target finite stationary rational partition coverage; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow}$ plus the Real stream finite partition spine; carrier stationary constant rational observation streams; classifier total $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w)$ and the segment classifiers named by P; stability by RatUp classifier transport, selected-tail seed exactness, and partition-spine induction; ledger hidden rational seed, seed e-one displays, total bound, segment radii, accumulated anchors, unary source anchors, source bounds, selected offsets, and continuation rows; core theorem Definition 9238.13; exactness the displayed fold equivalence and the displayed seed-tail equivalence; standard bridge finite RegSeqRat observation coverage by stationary rational streams; downstream use Phase Real and Completion diagonal-limit capstone ledger consumption. Status: ProofTarget.

Theorem 9238.14 (Stationary rational finite-partition diagonal embedding)**.** *For RatUp histories $d, e, d_0, e_0 : \mathsf{Hist}$, an observation anchor n, a total bound w, and a unary observation schedule $\eta : \mathbb{N} \to \mathsf{Hist}$, define the stationary rational RealUp observation streams*

$$C_d(i) = \mathsf{RatConstStream}_d(\eta(i)), \quad C_e(i) = \mathsf{RatConstStream}_e(\eta(i)).$$

Let P be a certified finite partition spine over (n, w) whose segment ledger records the unary source-anchor, source-bound, and selected-offset witnesses required by Definition 9238.13. Assume

$$\mathsf{RatHistoryCarrier}(d), \quad \mathsf{RatHistoryCarrier}(e), \quad \mathsf{RatHistoryClassifier}(d, e),$$

and the seed displays

$$\mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

Then the finite conjunction of segment classifiers supplied by P is a complete finite ledger for the stationary input X_d of Definition 9238.6. Its diagonal representative L_d is classified by the RealUp

classifier with the constant rational embedding represented by $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$*. Likewise, the stationary input* X_e *has diagonal representative* L_e *classified by the constant rational embedding represented by* $(\mathsf{RatConstStream}_e, \mathsf{E1}(e))$*, and the two embeddings are related through the displayed* $\mathsf{RatHistoryClassifier}(d, e)$*.*

Sketch. First use RatUp semantic reflexivity for the carrier rows on d and e. Together with Definition 126.48 and Definition 126.55, the stationary streams C_d and C_e are valid constant StreamNameUp witnesses at every observation named by η.

Next consume the finite partition spine. Apply Definition 9238.13 to C_d, C_e, the common RatUp classifier, the seed displays, and the certified spine P. This supplies the total $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w$ exactly as the finite conjunction of segment classifiers, and it preserves the selected endpoint seed-tail equivalence on every segment.

The selected-tail rows are the ledger needed by the constant regular Cauchy input X_d: every member of X_d is the same constant RealUp name $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$, and every finite observation selected from the partition reads the same hidden rational seed. The same argument applies to X_e.

Now apply Definition 9238.11 and Definition 9238.9 to read the segment ledgers as RealUp finite-window classifiers. The bridge identifies these classifiers with the e-one RealUp constant classifier, while the selected-tail theorem keeps the endpoint displays tied to d_0 and e_0.

Finally apply Definition 9238.7 separately to d and e. The diagonal extraction of X_d and X_e always reads a point of the same constant stream, so Definition 38.31 identifies the two diagonal representatives with their constant rational RealUp embeddings. The original $\mathsf{RatHistoryClassifier}(d, e)$, transported through Definition 126.55, relates those two constant embeddings. □

Theorem 9238.15 (Stationary rational partition refinement invariance)**.** *For RatUp histories* $d, e, d_0, e_0 : \mathsf{Hist}$*, an observation anchor* n*, a total bound* w*, and a unary observation schedule* $\eta : \mathbb{N} \to \mathsf{Hist}$*, define the stationary rational RealUp observation streams*

$$C_d(i) = \mathsf{RatConstStream}_d(\eta(i)), \quad C_e(i) = \mathsf{RatConstStream}_e(\eta(i)).$$

Let P *be a certified finite partition spine over* (n, w)*, and let* P' *be a certified refinement of* P *obtained by replacing each segment* (n_i, r_i) *of* P *by a finite adjacent chain* $(n_{i,j}, r_{i,j})$ *whose union is the same window. Suppose that the refinement map preserves the unary source-anchor, source-bound, selected-offset, and continuation ledgers: every subsegment of* P' *is recorded with the same stationary source anchor and source bound as its parent segment in* P*, selected offsets are the corresponding translated offsets inside the parent segment, and the continuation rows concatenate the subsegments back to the parent window. Assume*

$$\mathsf{RatHistoryCarrier}(d), \quad \mathsf{RatHistoryCarrier}(e), \quad \mathsf{RatHistoryClassifier}(d, e),$$

and the seed displays

$$\mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

Then the finite segment ledger named by P *holds iff the finite segment ledger named by* P' *holds:*

$$\bigwedge_{(n_i, r_i) \in P} \mathsf{RealStreamWindowClassifier}(C_d, C_e, n_i, r_i) \iff \bigwedge_{(n_{i,j}, r_{i,j}) \in P'} \mathsf{RealStreamWindowClassifier}(C_d, C_e, n_{i,j}, r_{i,j}).$$

Moreover, either ledger gives the same diagonal RealUp embedding conclusions of Definition 9238.14 for X_d *and* X_e*, and every selected endpoint in either ledger has the same seed-tail class supplied by Definition 9238.11.*

Sketch. First apply Definition 9238.13 to the original spine P. Its ledger is equivalent to the total classifier $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w)$, and its selected endpoint rows have the seed-tail class determined by d_0, e_0.

Apply the same partition fold to the refined spine P'. The refinement preserves the source-anchor and source-bound rows, so each subsegment is a stationary rational finite window with the same hidden RatUp seed classifier.

For a subsegment inside a parent segment, use adjacent-window restriction to read the parent classifier on the smaller window, and use adjacent-window concatenation along the refinement chain to assemble the parent window from the subsegments. The stationary streams erase the observation schedule η, so this restriction and concatenation carry no extra endpoint data beyond the translated selected offsets already recorded in the refinement ledger.

The forward direction transports the P ledger through the common total classifier and then through the fold for P'. The reverse direction uses the fold for P' to recover the same total classifier and then the fold for P to recover the original segment conjunction.

Finally apply Definition 9238.14 once with P and once with P'. Both applications consume the same total stationary window classifier and the same RatUp seed classifier. Apply Definition 9238.11 on the selected offsets in each spine to identify every endpoint tail with the same displayed seed-tail class. □

The ten-field fit is: traditional target finite stationary rational segmentation independence; BEDC source $\mathsf{Rat}^{\uparrow}+\mathsf{StreamName}^{\uparrow}+\mathsf{Real}^{\uparrow}$ plus Real stream finite partition spines; carrier stationary constant rational observation streams; classifier the total $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w)$ and the segment classifiers named by P and P'; stability by partition fold, adjacent-window restriction, adjacent-window concatenation, and selected-tail seed exactness; ledger hidden rational seed, seed e-one displays, total bound, parent segment radii, refined subsegment radii, translated selected offsets, source anchors, source bounds, and continuation rows; core theorem Definition 9238.15; exactness the common finite ledger equivalence, the common diagonal embedding certificate, and the common selected endpoint seed-tail class; standard bridge RegSeqRat stationary observation partitions with certified refined segmentation; downstream use Real and Completion diagonal-limit capstone ledger consumption. Status: ProofTarget.

Theorem 9238.16 (Stationary rational partition normal form)**.** *For RatUp seeds $d, e, d_0, e_0 : \mathsf{Hist}$, an observation anchor n, a bound w, and a unary observation schedule $\eta : \mathbb{N} \to \mathsf{Hist}$, define the stationary rational RealUp observation streams*

$$C_d(i) = \mathsf{RatConstStream}_d(\eta(i)), \quad C_e(i) = \mathsf{RatConstStream}_e(\eta(i)).$$

Let P be any certified finite partition spine refining the one-segment window (n, w). Suppose that the refinement certificate preserves the source-anchor, source-bound, selected-offset, and continuation rows: every segment of P is recorded as a translated subwindow of (n, w), selected offsets translate to selected offsets of the one-segment window, and the continuation rows concatenate all segments of P back to (n, w). Assume

$$\mathsf{RatHistoryCarrier}(d), \quad \mathsf{RatHistoryCarrier}(e), \quad \mathsf{RatHistoryClassifier}(d, e),$$

and the seed displays

$$\mathsf{hsame}(d, \mathsf{E1}(d_0)), \quad \mathsf{hsame}(e, \mathsf{E1}(e_0)).$$

Then the finite segment ledger named by P is equivalent to the normal-form one-window classifier:

$$\bigwedge_{(n_i, r_i) \in P} \mathsf{RealStreamWindowClassifier}(C_d, C_e, n_i, r_i) \iff \mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w).$$

Moreover, the diagonal RealUp embedding conclusions for the stationary inputs X_d and X_e are exactly the conclusions obtained from the one-segment ledger on (n, w): the representative L_d is classified by the constant rational embedding represented by $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$*, the representative L_e is classified by the constant rational embedding represented by* $(\mathsf{RatConstStream}_e, \mathsf{E1}(e))$*, and the relation between the two embeddings is the transported* $\mathsf{RatHistoryClassifier}(d, e)$*.*

Sketch. First regard the one-segment window (n, w) as a certified partition spine S with a single segment. The stationary finite-window bridge identifies the ledger of S with $\mathsf{RealStreamWindowClassifier}(C_d, C_e, n, w$ because both streams read constant RatUp seeds through the unary schedule η.

Next apply Definition 9238.13 to the given spine P. This folds the finite segment conjunction of P to the same total classifier and keeps each selected endpoint tied to the displayed seed tails d_0, e_0.

Use Definition 9238.15 with S and P. The preservation hypotheses are exactly the refinement rows required there: translated source anchors, translated source bounds, translated selected offsets, and continuation rows that reassemble the one-segment window. Hence the P ledger and the one-segment ledger determine the same finite observation content.

Finally apply Definition 9238.14 to S and to P. Both applications consume the same RatUp classifier, the same seed displays, and the same total stationary window classifier. The stationary finite-window bridge erases the partition choice, so the extracted diagonal representatives for X_d and X_e are the constant rational RealUp embeddings obtained from the one-segment ledger. □

Concretized in §422: Stationary finite-partition diagonal packets are realized as a concrete Real and Completion NameCert surface.

Concretized in §1207: Selected tail seed packets realize the stationary rational selected finite-window endpoint as a finite RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, hsame, Cont, Pkg, and NameCert handoff.

Concretized in §1226: Finite-prefix limit stability realizes the diagonal-limit finite observation route as a NameCert packet over budget, StreamName, RegSeqRat, DyadicRatCore, RealUp seal, hsame, Cont, and provenance rows.

Concretized in §1231: Tail cofinality budgets realize the shared finite tail threshold for StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp seal reads as a finite NameCert packet.

Concretized in §1264: Cofinal modulus normalization seals realize paired cofinal modulus schedules as one finite Real-and-Completion NameCert handoff over DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §1299: Cauchy tail threshold normalizers realize the shared finite tail threshold as a normal-form route for StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, CauchyTailModulusSealUp, RealTailAgreementSealUp, and RealUp consumers.

Concretized in §1352: Real cofinal modulus gates realize the finite cofinal-tail schedule and modulus-comparison rows consumed before a RealUp seal may serve the limit theorem.

Concretized in §1354: Canonical tail-choice packets realize the finite Real-completion diagonal selector as displayed modulus, tolerance, finite-index, tail-window, seal, refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1356: The canonical tail-choice audit packet realizes the Real-completion selector as a finite non-choice NameCert boundary with displayed tail-window, refusal, transport, route, provenance, and Real-seal rows.

Concretized in §5722: Finite limit seals realize the diagonal-limit finite-window route through StreamName, RegSeqRat, DyadicRatCore, and RealUp without selected limits or quotient streams.

Concretized in §5974: The Real completion boundary witness realizes the diagonal-limit convergence and uniqueness boundary as finite DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6604: Regular Cauchy lifted-limit packets realize the diagonal-limit handoff through DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp finite rows.

Concretized in §3942: Regular Cauchy limit uniqueness realizes the finite paired-tail, dyadic witness, RegSeqRat readback, and RealUp seal route used by the uniqueness half of the limit theorem.

Concretized in §6759: Dyadic window reflectors realize the Real-completeness finite-window classifier route through DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, and RealUp seal rows.

The ten-field fit is: traditional target finite stationary rational partition normal form; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow}$ plus Real stream finite partition spines; carrier stationary constant rational observation streams; classifier the normal-form window classifier $\mathsf{RealStreamWindowClassifier}(C_d, C_e, \eta$ and the finite segment classifiers named by P; stability by partition fold, refinement invariance, diagonal embedding, and the stationary finite-window bridge; ledger hidden rational seed, seed e-one displays, total bound, subwindow radii, translated source anchors, translated source bounds, selected offsets, and continuation rows; core theorem Definition 9238.16; exactness the common one-window classifier and the common diagonal embedding certificate; standard bridge RegSeqRat stationary observation partitions read through the one-window classifier; downstream use RegSeqRat and RealUp diagonal-limit arguments without carrying arbitrary finite segmentation data. Status: ProofTarget.

Theorem 9238.17 (Transported stationary rational diagonal)**.** *Let $d, e : \mathsf{Hist}$ satisfy $\mathsf{RatHistoryCarrier}(d)$, $\mathsf{RatHistoryCarrier}(e)$, and $\mathsf{RatHistoryClassifier}(d, e)$. Let X_d and X_e be the stationary regular Cauchy inputs of Definition 9238.6, and let L_d and L_e be their diagonal representatives from Definition 9238.3. Then the transported RatUp classifier induces the constant StreamNameUp classifier*

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e),$$

the e-one RealUp constant classifier

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)),$$

and the RealUp classifier

$$\mathsf{RealClassifier}(L_d, L_e).$$

Moreover these three rows are the same transported stationary rational diagonal row: every finite RegSeqRat or DyadicRatCore observation window read from L_d and L_e factors through the constant StreamNameUp classifier and then through the e-one RealUp constant classifier induced by $\mathsf{RatHistoryClassifier}(d, e)$.

Sketch. First apply Definition 9238.7 to d and to e. The two applications identify the diagonal representatives L_d and L_e with the constant rational RealUp names $(\mathsf{RatConstStream}_d, \mathsf{E1}(d))$ and $(\mathsf{RatConstStream}_e, \mathsf{E1}(e))$.

Next apply Definition 9238.8 to the supplied RatUp classifier. It transports $\mathsf{RatHistoryClassifier}(d, e)$ across the two diagonal agreement rows and produces $\mathsf{RealClassifier}(L_d, L_e)$.

The finite-window content is the same row read at the StreamNameUp boundary. Use Definition 9238.9. It identifies the RatUp seed classifier with the constant StreamNameUp classifier and with the e-one RealUp constant classifier for any displayed unary observation window.

Finally read the RegSeqRat and DyadicRatCore boundary rows through the RealUp certificate boundary. The RegSeqRat seal consumes the finite StreamNameUp window together with the dyadic source ledger, and the DyadicRatCore finite-window transport rows preserve that same ledger under selected subwindows. Hence every stationary finite observation of L_d and L_e is the transported constant-stream row induced by the original RatUp classifier, with no additional RealUp comparison datum. □

Theorem 9238.18 (Transported stationary finite-window seal exhaustion)**.** *Let d, e : Hist satisfy $\mathsf{RatHistoryCarrier}(d)$, $\mathsf{RatHistoryCarrier}(e)$, and $\mathsf{RatHistoryClassifier}(d, e)$. Let L_d and L_e be the diagonal representatives of the stationary regular Cauchy inputs associated to d and e in Definition 9238.17. For every certified finite observation window W of L_d and L_e, there is a unique seal row Σ_W whose four public readings agree with the same RatUp seed classifier: the DyadicRatCore observation ledger carried by W, the StreamNameUp constant-window classifier, the RegSeqRat finite-window seal consumption row, and the RealUp finite-window classifier all read exactly $\mathsf{RatHistoryClassifier}(d, e)$.*

Sketch. First apply Definition 9238.17. It identifies the diagonal representatives L_d and L_e with the transported stationary constant-stream row induced by $\mathsf{RatHistoryClassifier}(d, e)$.

Now restrict to the certified finite window W. By Definition 9238.9, the finite StreamNameUp window, the e-one RealUp constant classifier, and the RatUp seed classifier are the same stationary row after reading the displayed window.

Feed this finite-window row to the RegSeqRat boundary. The seal exactness theorem Definition 38.251 consumes precisely the StreamNameUp finite-window classifier together with the DyadicRatCore observation ledger, so the seal row is forced by the already displayed window data.

Finally pass through the RealUp phase boundary using Definition 38.314 and Definition 38.320. These results say that the RealUp finite-window classifier has no additional public coordinate beyond the StreamNameUp window, the dyadic ledger, and the RegSeqRat seal. Thus any two seal rows over W project to the same four readings and are the same BHist package by constructor inversion, giving the claimed uniqueness and exhaustion. □

Concretized in §475: Transported stationary diagonal finite-window seals are realized as a concrete NameCert handoff across RatUp, StreamNameUp, RegSeqRat, DyadicRatCore, and RealUp.

Concretized in §605: Transported stationary finite-window seal uniqueness is realized as a four-reading NameCert packet over RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §613: Stationary diagonal window routes realize the RatUp-to-RealUp finite observation path as a NameCert packet over StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, and seal non-escape rows.

Concretized in §636: Cauchy modulus comparisons are realized as finite shared-tail NameCert packets linking CauchyModulusUp schedules, StreamNameUp windows, DyadicRatCore tolerances, RegSeqRatUp reads, and RealUp seals.

Concretized in §641: Cauchy modulus extraction is realized as a finite observation-budget packet feeding the real-completeness diagonal route.

Concretized in §632: Diagonal cofinal tails realize the transported stationary diagonal window as a finite cofinal-tail NameCert packet across RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §693: Regular Cauchy window fusion realizes the shared finite observation-window route across StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp seals.

Concretized in §897: Stationary window naturality is realized as a finite NameCert packet identifying RatUp classifier transport with StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp finite-window seal readings.

Concretized in §996: Rat classifier transport seals realize the stationary rational diagonal row as a finite NameCert handoff across RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §1017: Shared tail normalizers realize two Real-completion tail schedules as one finite StreamName, RegSeqRat, DyadicRatCore, shared-fiber, and RealUp seal route.

Concretized in §1041: Cofinal diagonal seals realize the cofinal-tail-to-diagonal RealUp seal route as finite precision, cofinal-budget, diagonal-budget, RegSeqRat, StreamName, DyadicRatCore, RealUp, route, provenance, and NameCert rows.

Concretized in §1104: Stationary window seal exhaustion is realized as a finite RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp NameCert handoff.

Concretized in §2240: Finite window pullbacks realize the transported stationary diagonal seal route as a finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert packet.

Concretized in §3722: Real completeness diagonal selectors realize the transported stationary diagonal route as finite choice-free selector, diagonal-tail, StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8968: Finite observation diagonal routers realize StreamName, RegSeqRat, DyadicRatCore, RegularCauchyDiagonalSeal, and RealUp finite-window diagonal routing as NameCert data.

9238.3 Constructive content and axiom purity

Remark 9238.19 (Explicit moduli throughout)**.** *Every modulus appearing in Definition 9238.2, Definition 9238.3, and Definition 9238.5 is specified as a function on naturals and rationals computed by primitive recursion on the inductive carriers of chapter 37. The required proof obligations are exactly the certificate-level regularity, convergence, and* psame*-closure facts that avoid quotienting an equivalence relation by a host-level* Quot*-soundness step. These obligations are the axiom-pure targets for the later machine-checked real-completion layer.*

Remark 9238.20 (Reuse for the continuous certificate)**.** *The diagonal limit construction (Definition 9238.3) identifies the limit step expected by the continuous certificate* $\mathsf{NameCert}_{\mathsf{CFun}\uparrow}$ *(chapter 59). A continuous function on* $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$ *is, in BEDC, a function on* RegSeqRat *together with an explicit pointwise modulus; once the real-completion obligations are checked, downstream continuity certificates can consume the same diagonal regularity bound. The reuse is structural, not a separate construction.*

Remark 9238.21 (Composition with the compact uniform-continuity capstone)**.** *After* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *has the supplied completion certificate and the continuous-function certificate is available, the compact uniform-continuity capstone (chapter 9237) can be instantiated on closed bounded intervals* $[a, b] \subset \mathsf{Real}^{\uparrow}$*. The downstream consumer is a future kernel theorem that, for every continuous* $f\colon [a, b] \to \mathsf{Real}^{\uparrow}$*, derives a uniform modulus depending only on the tolerance from the checked compactness and continuity certificates.*

Concretized in §372: Stationary regular Cauchy diagonal packets are represented as finite NameCert carriers over RatUp, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §365: Finite observation windows are realized as a NameCert envelope linking DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp seals.

Concretized in §390: The diagonal-limit modulus is represented as a finite NameCert packet over RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, and RealUp.

Concretized in §443: Regular Cauchy limit classifiers are realized as finite NameCert packets linking RegSeqRatUp diagonal extraction, StreamNameUp windows, and RealUp seal handoff.

Concretized in §437: Transported stationary diagonal window-seal packets are realized as a finite NameCert carrier over RatUp, StreamNameUp, RegSeqRatUp, FiniteWindowEnvelopeUp, and RealUp.

Concretized in §450: The Real completeness diagonal-limit target is represented as a finite NameCert seed over DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §466: The three-leg diagonal regularity estimate is realized as a finite NameCert packet over DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §559: Finite-tail diagonal seals realize the selected regular-Cauchy tail window as a RealUp-facing NameCert packet without quotient streams or host real equality.

Concretized in §560: Regular Cauchy tail-meet packets realize the shared dyadic tail threshold needed by diagonal RealUp completion consumers.

Concretized in §293: Dyadic absolute-value rows are realized as finite DyadicRatCore BHist packets with sign-split, nonnegative-output, and tolerance-ledger surfaces.

Concretized in §561: Real diagonal-window coverage realizes the finite selected-window route from StreamName and RegSeqRat data through DyadicRatCore observations to the RealUp seal.

Concretized in §618: Finite diagonal mesh-budget packets realize the precision-to-window-to-dyadic-readback route used by Real completeness consumers before the RealUp seal.

Concretized in §600: Regular Cauchy limit transport packets realize classifier-stable diagonal-limit handoff across RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, and RealUp rows.

Concretized in §593: Regular Cauchy witness schedules realize the modulus-indexed finite observation route from RegSeqRat and StreamName data through DyadicRatCore readback to the RealUp seal.

Concretized in §612: Cauchy diagonal fusion realizes the finite route from observation budgets and witness schedules through schedule fusion to the RealUp diagonal seal.

Concretized in §624: Regular Cauchy zero packets realize the zero-source identity surface for tail-meet, diagonal, and RealUp seal consumers.

Concretized in §626: Stationary diagonal finite-window classifier correspondence is realized as a concrete NameCert packet linking RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §628: The modulus-indexed finite tail-window cover is realized as a concrete NameCert packet from unary precision requests through CauchyModulusUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp seal rows.

Concretized in §631: Product regular-Cauchy limit packets realize the product-limit handoff across CauchyProductUp, RegularCauchyLimitClassifierUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §645: Bounded Cauchy covers realize the finite budget-to-window-to-RegSeqRat-to-RealUp seal route used by the real-completion diagonal argument.

Concretized in §649: Stationary window lattices realize common finite-window refinement for stationary Real-completeness diagonal consumers.

Concretized in §650: Choice-free limit schedules are realized as finite precision-to-window-to-seal NameCert packets over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §757: Dyadic tail-radius ledgers realize finite tail-window radius accounting for RegSeqRat and RealUp seal consumers without quotient streams, host real equality, or selected limits.

Concretized in §884: Real tail-agreement transitivity is realized as a finite three-source tail-window NameCert packet over two RealTailAgreementSealUp rows, shared middle RegSeqRat data, DyadicRatCore readbacks, and RealUp handoff.

Concretized in §766: Real algebra and order packets realize constructive addition, multiplication, nonnegative order, and strict apartness as finite RealUp-facing NameCert rows.

Concretized in §1200: Real modulus-purity boundaries realize explicit modulus, finite-window, RegSeqRat readback, RealUp seal, and axiom-refusal rows as one finite NameCert packet.

Concretized in §1325: Real window classifier seals realize the explicit finite-window classifier and axiom-purity refusal row for Real-completion modulus consumers.

Concretized in §8426: Finite-window Real-completeness subcover packets realize the StreamName, RegSeqRat, DyadicRatCore, and RealUp seal route without ambient open-cover compactness.

9238.4 Modulus comparison boundary

The located-real comparison surface is not a comparison of two completed host points. It is a comparison of two presented processes through finite windows selected by explicit moduli. This point is already visible in the Lean-side shape of LReal: a located real carries a rational sequence, a monotone modulus, and a Cauchy witness. The pointwise relation IrLe is deliberately stronger and should not be read as the general comparison interface. The finite comparison interface is instead the pair formed by LRealClose and IrApart.

Definition 9238.22 (Modulated located real)**.** *Fix a rational metric kit K with rational closeness, apartness, and order rows. A modulated located real is a triple*

$$x = (x_{\mathrm{seq}}, x_{\mathrm{mod}}, x_{\mathrm{cau}})$$

where $x_{\mathrm{seq}} : \mathsf{Nat}^{\uparrow} \to \mathsf{Rat}^{\uparrow}$, $x_{\mathrm{mod}} : \mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$, *and*

$$x_{\mathrm{mod}}(k) \le m \wedge x_{\mathrm{mod}}(k) \le n \Longrightarrow K.\mathsf{close}(x_{\mathrm{seq}}(m), x_{\mathrm{seq}}(n), k).$$

The rational embedding is the degenerate case

$$\eta(q)_{\mathrm{seq}}(n) = q, \qquad \eta(q)_{\mathrm{mod}}(k) = 0.$$

Thus a rational comparison is a comparison between a stationary process and the other displayed process. A comparison between two non-rational presentations is a comparison between two non-trivial moduli.

Lean4 (definition) `BEDC.Derived.LocatedReal.LReal`
Lean4 (definition) `BEDC.Derived.LocatedReal.lrLe`

Definition 9238.23 (Finite tolerance comparison)**.** *For located reals a, b and a tolerance k, the finite tolerance relation is a comparison datum D with its own modulus $D_{\mathrm{mod}} : \mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$ such that*

$$D_{\mathrm{mod}}(k) \le m \wedge D_{\mathrm{mod}}(k) \le n \Longrightarrow K.\mathsf{close}(a_{\mathrm{seq}}(m), b_{\mathrm{seq}}(n), k).$$

This is the role of LRealEqData *and* LRealClose*. The comparison data may use a common schedule separate from the two endpoint moduli, but it still inspects only finite rational windows.*

Lean4 (definition) `BEDC.Derived.LocatedReal.LRealEqData`
Lean4 (definition) `BEDC.Derived.LocatedReal.LRealClose`

Definition 9238.24 (Finite apartness window)**.** *The apartness surface for two located reals is the existence of a finite precision k such that the two endpoint moduli already expose a rational separation:*

$$\exists k : \mathsf{Nat}^{\uparrow},\ K.\mathsf{apart}(a_{\mathrm{seq}}(a_{\mathrm{mod}}(k)), b_{\mathrm{seq}}(b_{\mathrm{mod}}(k)), k).$$

This is the role of lrApart*. The witness is a finite separation window, not a decision that every pair of located reals has already been classified as left, equal, or right.*

Lean4 (definition) `BEDC.Derived.LocatedReal.lrApart`

Theorem 9238.25 (Finite comparison is process-level)**.** *Every finite tolerance comparison or apartness witness between located reals factors through the two sequences, their moduli, a comparison modulus when present, and rational-level close or apartness rows. It does not require a host real point or quotient representative.*

Proof. For finite tolerance, unfold Definition 9238.23. A tolerance request k selects finite thresholds through D_{mod}, and the only numerical read is the rational comparison $K.\mathsf{close}(a_{\mathrm{seq}}(m), b_{\mathrm{seq}}(n), k)$.

For apartness, unfold Definition 9238.24. The witness selects one finite precision k and reads the two rational observations picked by $a_{\mathrm{mod}}(k)$ and $b_{\mathrm{mod}}(k)$. The only separation claim is the displayed rational apartness row at that precision. Neither construction asks for a completed host point, a quotient stream, or a selected representative outside the visible moduli. □

Theorem 9238.26 (Comparison moduli compose)**.** *If a is finitely close to b at tolerance $k+1$, and b is finitely close to c at tolerance $k+1$, then a is finitely close to c at tolerance k.*

Proof. Let D_{ab} and D_{bc} be the two comparison data. Use the common comparison modulus

$$j \longmapsto \max(D_{ab,\mathrm{mod}}(j+1), D_{bc,\mathrm{mod}}(j+1)).$$

At this shared finite threshold, the first datum gives $K.\mathsf{close}(a, b, j+1)$ and the second gives $K.\mathsf{close}(b, c, j+1)$ at the corresponding rational observations. The rational triangle row in K composes the two tolerance-$j+1$ comparisons into a tolerance-j comparison from a to c. This is the proof pattern implemented by LRealClose_triangle. □

Lean4 ✓ `BEDC.Derived.LocatedReal.LRealClose triangle`

Concretized in §9022: Regular Cauchy selector stability realizes finite tolerance comparison by preserving selected modulus rows, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readbacks, and RealUp seals under displayed transport.

Concretized in §9085: Dyadic modulus comparison boundary packets realize the finite common-refinement comparison route through two located-real endpoints, one dyadic comparison modulus, paired finite windows, RegSeqRat readbacks, RealUp seals, transport, provenance, and local NameCert data.

Definition 9238.27 (Exact comparison boundary)**.** *An exact comparison operator for arbitrary located reals is a total operation of the form*

$$\prod_{x,y} (x < y) + (x = y) + (y < x),$$

where equality is full located-real equality and the strict branches are point-level order branches, not finite tolerance or apartness windows.

Theorem 9238.28 (Exact comparison has omniscience strength)**.** *A total operator of Definition 9238.27, supplied for arbitrary located-real presentations, would decide the limited principle of omniscience for binary streams.*

Proof. Given a binary stream $\alpha : \mathsf{Nat}^{\uparrow} \to \{0, 1\}$, form the nonnegative located real presentation

$$x_\alpha = \sum_{n=0}^{\infty} \alpha(n) 2^{-(n+1)}$$

by its rational partial sums and the usual geometric tail modulus. Apply the exact comparison operator to x_α and 0.

The branch $x_\alpha < 0$ is incompatible with the nonnegative construction. If $x_\alpha = 0$, then no coefficient can be 1, since any displayed $\alpha(n) = 1$ supplies a positive dyadic lower bound. Hence $\forall n, \alpha(n) = 0$. If $0 < x_\alpha$, the strict branch supplies a positive separation threshold; inspecting the finite prefix needed by that threshold yields some n with $\alpha(n) = 1$. The exact comparison operator therefore returns

$$(\forall n, \alpha(n) = 0) + (\exists n, \alpha(n) = 1).$$

This is LPO for binary streams. □

Corollary 9238.29 (Located comparison is not trichotomy)**.** *The constructive comparison surface of* $\mathsf{LocatedReal}^{\uparrow}$ *consists of finite tolerance comparison, finite apartness witnesses, comparison-modulus composition, and Real-seal readback through displayed rows. It does not export arbitrary exact trichotomy.*

Proof. By Definition 9238.25, the constructive comparison surface remains at the level of finite observations and explicit moduli. By Definition 9238.28, a total exact trichotomy operator for arbitrary located-real presentations would decide LPO. Such an operator is therefore outside the axiom-pure finite-window comparison interface recorded in this chapter. □

Concretized in §1730: Located real comparison packets realize the finite shared-window comparison surface for two displayed located-real endpoints.

Concretized in §5813: Located real comparison-modulus packets realize finite tolerance schedules without exporting trichotomy.

Concretized in §3640: Real located-order packets route order-facing reads through two Real seals, paired regular readbacks, shared windows, dyadic tolerances, and apartness handoff rows.

Concretized in §1202: Finite-window Real separation packets realize positive separation through shared finite windows and dyadic tolerance readback.

Concretized in §9102: Finite observation boundary packets realize exact located-real comparison requests as two endpoint moduli, one finite observation window, finite apartness or equality evidence, and a refusal ledger for total trichotomy.

Concretized in §9134: Finite comparison boundary packets realize process-level located-real tolerance and apartness comparisons as finite modulus, window, rational close/apart, transport, replay, provenance, and NameCert rows.

Concretized in §372: Stationary regular Cauchy diagonal packets are represented as finite NameCert carriers over RatUp, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §365: Finite observation windows are realized as a NameCert envelope linking DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp seals.

Concretized in §390: The diagonal-limit modulus is represented as a finite NameCert packet over RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, and RealUp.

Concretized in §443: Regular Cauchy limit classifiers are realized as finite NameCert packets linking RegSeqRatUp diagonal extraction, StreamNameUp windows, and RealUp seal handoff.

Concretized in §437: Transported stationary diagonal window-seal packets are realized as a finite NameCert carrier over RatUp, StreamNameUp, RegSeqRatUp, FiniteWindowEnvelopeUp, and RealUp.

Concretized in §450: The Real completeness diagonal-limit target is represented as a finite NameCert seed over DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §466: The three-leg diagonal regularity estimate is realized as a finite NameCert packet over DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §559: Finite-tail diagonal seals realize the selected regular-Cauchy tail window as a RealUp-facing NameCert packet without quotient streams or host real equality.

Concretized in §560: Regular Cauchy tail-meet packets realize the shared dyadic tail threshold needed by diagonal RealUp completion consumers.

Concretized in §293: Dyadic absolute-value rows are realized as finite DyadicRatCore BHist packets with sign-split, nonnegative-output, and tolerance-ledger surfaces.

Concretized in §561: Real diagonal-window coverage realizes the finite selected-window route from StreamName and RegSeqRat data through DyadicRatCore observations to the RealUp seal.

Concretized in §618: Finite diagonal mesh-budget packets realize the precision-to-window-to-dyadic-readback route used by Real completeness consumers before the RealUp seal.

Concretized in §600: Regular Cauchy limit transport packets realize classifier-stable diagonal-limit handoff across RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, and RealUp rows.

Concretized in §593: Regular Cauchy witness schedules realize the modulus-indexed finite observation route from RegSeqRat and StreamName data through DyadicRatCore readback to the RealUp seal.

Concretized in §612: Cauchy diagonal fusion realizes the finite route from observation budgets and witness schedules through schedule fusion to the RealUp diagonal seal.

Concretized in §624: Regular Cauchy zero packets realize the zero-source identity surface for tail-meet, diagonal, and RealUp seal consumers.

Concretized in §626: Stationary diagonal finite-window classifier correspondence is realized as a concrete NameCert packet linking RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §628: The modulus-indexed finite tail-window cover is realized as a concrete NameCert packet from unary precision requests through CauchyModulusUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp seal rows.

Concretized in §631: Product regular-Cauchy limit packets realize the product-limit handoff across CauchyProductUp, RegularCauchyLimitClassifierUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp.

Concretized in §658: Regular Cauchy tail fusion realizes the cofinal tail-to-diagonal RealUp seal route as a finite NameCert packet.

Concretized in §697: Diagonal tail selectors realize the finite cofinal-tail route from a regular Cauchy modulus through StreamName windows, dyadic readback, RegSeqRat rows, and the RealUp seal without quotient streams or host-selected limits.

Concretized in §699: Real tail-agreement seals realize paired cofinal tail-window agreement as a finite NameCert packet over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §646: Diagonal-limit compatibility is realized as a finite NameCert packet tying RealCompleteness, RegularCauchyTriangleBound, CauchyLimitSeal, StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §660: Regular Cauchy sequence packets realize the finite family, modulus, selected-window, dyadic readback, RegSeqRat, and RealUp seal surface used by the diagonal-limit handoff.

Concretized in §653: Regular Cauchy swap bisimulations realize selector-swapped interleavings as finite RegSeqRat, StreamName, RealUp, hsame, Cont, Pkg, and NameCert packets.

Concretized in §662: Budgeted tail-meet packets realize finite observation budgets and shared-tail thresholds for the Real completeness seal route.

Concretized in §652: Real completion witness extractors realize the diagonal regular-Cauchy limit route as a finite NameCert packet over modulus schedules, selected windows, RegSeqRat witnesses, RealUp seals, hsame transport, Cont replay, and non-escape ledgers.

Concretized in §663: Regular Cauchy cofinal limits are realized as finite cofinal-window NameCert packets over DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp seal rows.

Concretized in §670: Cofinal regular-limit budgets realize the Real completeness diagonal target as finite precision, cofinal-window, RegSeqRat handoff, RealUp seal, route, provenance, and NameCert rows.

Concretized in §676: Regular Cauchy product budgets realize the finite product-window ledger from RegSeqRatUp sources through StreamNameUp, DyadicRatCoreUp, FiniteErrorBudgetUp, and RealUp seal rows.

Concretized in §2076: Regular Cauchy product projections realize component readbacks from displayed product packets through product-modulus budgets, StreamName windows, DyadicRatCore tolerances, RegSeqRat handoffs, and RealUp seals.

Concretized in §678: Regular Cauchy selector budgets realize the modulus-selected finite window, RegSeqRat handoff, RealUp seal route, and non-escape ledger used by the Real completeness diagonal construction.

Concretized in §695: Choice-free diagonal selectors realize the finite precision, cofinal-window, StreamName, RegSeqRat, RealUp seal, hsame, Cont, Pkg, and NameCert selector boundary used by the Real completeness handoff.

Concretized in §680: Dyadic cofinal tail selectors realize the finite cofinal-window route used by the Real and Completion diagonal-limit handoff.

Concretized in §681: Tail cofinality schedules realize the finite shared-threshold route from DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp seal rows used by the diagonal-limit handoff.

Concretized in §702: RegularCauchy merge packets realize the two-route cofinal selector agreement used by the Real completeness diagonal-limit handoff.

Concretized in §698: Cofinal tail budgets realize finite shared-tail threshold packets for the Real and Completion diagonal-limit handoff.

Concretized in §706: Observable Cauchy seals realize finite observation routes from StreamName windows and dyadic readbacks through RegSeqRat handoff rows into a RealUp seal.

Concretized in §708: Regular Cauchy telescoping budgets realize finite chained-window error-ledger composition for the Real completeness diagonal-limit handoff.

*Concretized in §**??**:* Real modulus fusion realizes the real-completeness vision as a finite shared-modulus NameCert over Cauchy modulus rows, tail-meet thresholds, StreamName windows, RegSeqRat handoff, and RealUp seal rows.

Concretized in §1171: Cauchy tail-modulus seals realize the shared-tail modulus, fusion, witness-extractor, and RealUp seal route as one finite NameCert packet.

Concretized in §778: Regular Cauchy modulus normalizers realize the shared finite modulus-window route used by the Real completeness diagonal-limit handoff.

Concretized in §782: Finite observation-budget selector packets realize the finite diagonal observation budget used by the Real completeness route.

Concretized in §838: Regular Cauchy tail schedules realize the finite precision-to-cofinal-tail route consumed by tail-meet, tail-fusion, diagonal-selector, and RealUp seal packets.

Concretized in §853: Cauchy diagonal budgets realize the Real completeness diagonal selector as finite precision, mesh-budget, comparison, seal-handoff, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1198: Cofinal modulus seals realize the Real-completeness cofinal-tail modulus route as a finite completion-seal NameCert surface.

Concretized in §870: Diagonal-limit stability packets realize the finite stability comparison between a diagonal representative and the sealed RealUp limit route.

Concretized in §874: Regular-Cauchy finite observation routes are realized as finite StreamName, DyadicRatCore, RegSeqRat, RealObservationBudget, and RealUp seal packets.

Concretized in §885: Cauchy modulus refinements realize the shared-meet and selector-budget route used by the Real completeness diagonal handoff.

Concretized in §891: Real seal congruence packets realize classifier-preserving replacement of RegSeqRat and StreamName readbacks under the RealUp seal.

Concretized in §887: Real seal comparison packets realize equality-facing comparison of two sealed RealUp routes as finite CauchyLimitSeal, RegularLimitUniqueness, DiagonalLimitStability, hsame, Cont, Pkg, and NameCert rows.

Concretized in §893: Regular Cauchy tail fibers realize the shared finite tail-fiber classifier consumed by RealUp diagonal completion and tail-agreement routes.

Concretized in §896: Diagonal limit budget packets realize the finite modulus, window, dyadic-tolerance, and RealUp seal budget consumed by the Real completeness diagonal route.

Concretized in §1081: Real completion selector-seal packets realize the finite selector-budget, selected-window, RegSeqRat readback, CauchyLimitSeal, and RealUp seal route.

Concretized in §1113: Finite Cauchy gluing budgets realize the Real-completeness witness route as one finite budget, synchronizer, completion-seal, StreamName, RegSeqRat, DyadicRatCore, and RealUp NameCert surface.

Concretized in §5839: Regular Cauchy mesh cofinality realizes the Real-completeness route as a finite common-refinement mesh between CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §5975: Regular Cauchy mesh gluing realizes the finite common-refinement route from mesh cofinality and gluing budgets to the cofinal modulus seal.

Concretized in §7076: Real inverse-modulus packets realize the guarded reciprocal-modulus route as finite CauchyInverseBudget, RegSeqRat, RealReciprocal, RealUp, transport, replay, provenance, and NameCert rows.

Concretized in §7564: Tail-window filter refinements realize the finite common-filter route from StreamName windows through RegSeqRat, DyadicRatCore tolerance, and the RealUp seal.

Concretized in §8082: Cauchy uniform criterion packets realize the regular Cauchy uniform-tail route as finite StreamName, DyadicRatCore, RegSeqRat, RealUp, common-modulus, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8310: Regular Cauchy diagonal-limit moduli realize the diagonal representative's finite modulus route through RegSeqRat, StreamName, DyadicRatCore, CauchyLimitSeal, and RealUp rows.

Concretized in §416: Regular Cauchy interleave packets realize finite merge schedules for two RegSeqRat sources through StreamName windows, DyadicRatCore observations, and RealUp handoff rows.

Concretized in §646: Diagonal-limit compatibility is realized as a finite NameCert packet tying RealCompleteness, RegularCauchyTriangleBound, CauchyLimitSeal, StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §660: Regular Cauchy sequence packets realize the finite family, modulus, selected-window, dyadic readback, RegSeqRat, and RealUp seal surface used by the diagonal-limit handoff.

Concretized in §653: Regular Cauchy swap bisimulations realize selector-swapped interleavings as finite RegSeqRat, StreamName, RealUp, hsame, Cont, Pkg, and NameCert packets.

Concretized in §662: Budgeted tail-meet packets realize finite observation budgets and shared-tail thresholds for the Real completeness seal route.

Concretized in §652: Real completion witness extractors realize the diagonal regular-Cauchy limit route as a finite NameCert packet over modulus schedules, selected windows, RegSeqRat witnesses, RealUp seals, hsame transport, Cont replay, and non-escape ledgers.

Concretized in §663: Regular Cauchy cofinal limits are realized as finite cofinal-window NameCert packets over DyadicRatCore, StreamNameUp, RegSeqRat, and RealUp seal rows.

Concretized in §670: Cofinal regular-limit budgets realize the Real completeness diagonal target as finite precision, cofinal-window, RegSeqRat handoff, RealUp seal, route, provenance, and NameCert rows.

Concretized in §676: Regular Cauchy product budgets realize the finite product-window ledger from RegSeqRatUp sources through StreamNameUp, DyadicRatCoreUp, FiniteErrorBudgetUp, and RealUp seal rows.

Concretized in §678: Regular Cauchy selector budgets realize the modulus-selected finite window, RegSeqRat handoff, RealUp seal route, and non-escape ledger used by the Real completeness diagonal construction.

Concretized in §695: Choice-free diagonal selectors realize the finite precision, cofinal-window, StreamName, RegSeqRat, RealUp seal, hsame, Cont, Pkg, and NameCert selector boundary used by the Real completeness handoff.

Concretized in §680: Dyadic cofinal tail selectors realize the finite cofinal-window route used by the Real and Completion diagonal-limit handoff.

Concretized in §681: Tail cofinality schedules realize the finite shared-threshold route from DyadicRatCore, StreamNameUp, RegSeqRatUp, and RealUp seal rows used by the diagonal-limit handoff.

Concretized in §702: RegularCauchy merge packets realize the two-route cofinal selector agreement used by the Real completeness diagonal-limit handoff.

Concretized in §698: Cofinal tail budgets realize finite shared-tail threshold packets for the Real and Completion diagonal-limit handoff.

Concretized in §706: Observable Cauchy seals realize finite observation routes from StreamName windows and dyadic readbacks through RegSeqRat handoff rows into a RealUp seal.

Concretized in §708: Regular Cauchy telescoping budgets realize finite chained-window error-ledger composition for the Real completeness diagonal-limit handoff.

Concretized in §??: Real modulus fusion realizes the real-completeness vision as a finite shared-modulus NameCert over Cauchy modulus rows, tail-meet thresholds, StreamName windows, RegSeqRat handoff, and RealUp seal rows.

Concretized in §778: Regular Cauchy modulus normalizers realize the shared finite modulus-window route used by the Real completeness diagonal-limit handoff.

Concretized in §782: Finite observation-budget selector packets realize the finite diagonal observation budget used by the Real completeness route.

Concretized in §1361: Finite observation cofinality selectors realize finite budget rows whose selected StreamName windows are cofinal for the Real-completion diagonal route.

Concretized in §838: Regular Cauchy tail schedules realize the finite precision-to-cofinal-tail route consumed by tail-meet, tail-fusion, diagonal-selector, and RealUp seal packets.

Concretized in §853: Cauchy diagonal budgets realize the Real completeness diagonal selector as finite precision, mesh-budget, comparison, seal-handoff, hsame, Cont, Pkg, and NameCert rows.

Concretized in §870: Diagonal-limit stability packets realize the finite stability comparison between a diagonal representative and the sealed RealUp limit route.

Concretized in §874: Regular-Cauchy finite observation routes are realized as finite StreamName, DyadicRatCore, RegSeqRat, RealObservationBudget, and RealUp seal packets.

Concretized in §1007: Finite observation interfaces realize Real-completion finite-window reads as StreamName, RegSeqRat, DyadicRatCore, and RealUp NameCert handoff packets.

Concretized in §885: Cauchy modulus refinements realize the shared-meet and selector-budget route used by the Real completeness diagonal handoff.

Concretized in §891: Real seal congruence packets realize classifier-preserving replacement of RegSeqRat and StreamName readbacks under the RealUp seal.

Concretized in §887: Real seal comparison packets realize equality-facing comparison of two sealed RealUp routes as finite CauchyLimitSeal, RegularLimitUniqueness, DiagonalLimitStability, hsame, Cont, Pkg, and NameCert rows.

Concretized in §994: Real selector comparisons realize the finite two-selector diagonal route that forces shared StreamName, RegSeqRat, CauchyLimitSeal, RealTailAgreementSeal, and RealUp seal agreement.

Concretized in §1001: Real seal admission cuts realize the finite budget-window-readback boundary before a RealUp seal or CauchyLimitSeal handoff can be consumed.

Concretized in §893: Regular Cauchy tail fibers realize the shared finite tail-fiber classifier consumed by RealUp diagonal completion and tail-agreement routes.

Concretized in §1009: Cofinal real request reuse realizes repeated finite Real-completion precision bundles as one displayed StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert route.

Concretized in §1020: Cauchy window transducers realize finite StreamName window steps, dyadic tolerance rows, RegSeqRat readback, and RealUp seal routing.

Concretized in §1022: Regular Cauchy window intersections realize the common finite-window route used by the Real completeness diagonal handoff.

Concretized in §1034: Cauchy seal functors realize reusable finite transport of CauchyLimitSeal and RealCompletionSealClassifier rows through StreamName, RegSeqRat, DyadicRatCore, and RealUp handoff surfaces.

Concretized in §1035: Located Cauchy modulus packets realize the finite located tail-modulus route from StreamName and RegSeqRat windows to a RealUp seal.

Concretized in §1039: Real limit uniqueness seals realize the finite Real-completion uniqueness route through RegSeqRat, StreamName, DyadicRatCore, CauchyLimitSeal, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1119: Finite tail filters realize the shared cofinal-tail route for StreamName, DyadicRatCore, RegSeqRat, and RealUp seal reads.

Concretized in §1174: Cofinal-window Real seals realize the Real-completeness route as finite StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1175: Regular Cauchy witness selectors realize the diagonal-limit selector as finite modulus-request, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, RealUp seal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1180: Regular Cauchy completion frontier packets realize the Phase Real and Completion route as finite selector, cofinal-tail, RegSeqRat, StreamName, DyadicRatCore, RealUp seal, and NameCert rows.

Concretized in §1199: Finite-window Real seal audits realize the Real completeness route as displayed StreamName, DyadicRatCore, RegSeqRat, RealUp seal, audit-refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1210: Tail-modulus alignments realize the shared Real-completion tail-modulus route as one finite StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert packet.

Concretized in §2206: Real prefix diagnostics realize finite prefix-window, StreamName, RegSeqRat, dyadic-tolerance, and Real-seal readback rows for the Real completeness route.

Concretized in §3741: Uniform completion common-window packets realize the Real and Completion route as one UniformCompletionFunctor request, shared StreamName window, RegSeqRat readback, DyadicRatCore budget, and RealUp seal NameCert packet.

Concretized in §6229: Regular Cauchy tail balls realize the finite threshold, tail-window, dyadic tolerance, RegSeqRat readback, and RealUp seal route used by the Real-completion finite-tail direction.

Concretized in §7007: Cauchy subbasis tail factorization realizes the finite route from generated Cauchy subbases through cofinite tail filters to StreamName, RegSeqRat, and RealUp seal reads.

Concretized in §903: Observation-budget limiters realize finite request caps before StreamName, RegSeqRat, and RealUp seal reads.

Concretized in §905: Real Cauchy witness grids realize the Real completeness diagonal route as finite selector-budget, witness-ledger, completion-seal, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §906: Regular Cauchy tail budgets realize finite tail-threshold, observation-budget, window-service, dyadic-readback, RegSeqRat handoff, and RealUp seal rows for the Real completeness route.

Concretized in §917: Tail budget coherence packets realize the Real completeness route as one finite threshold, observation-budget, selector-budget, agreement-seal, and limit-seal NameCert surface.

Concretized in §920: CofinalTailAgreementUp realizes the paired cofinal-tail agreement route between regular Cauchy tail evidence and the Real completion seal.

Concretized in §919: Finite-prefix limit budgets realize the bounded observation surface used by the Real completeness diagonal-limit route.

Concretized in §918: Bounded monotone convergence seals realize the bounded monotone Cauchy witness as a finite RealUp convergence handoff.

Concretized in §930: Finite real route commitments realize the Real-completion route as finite precision, budget, StreamName, RegSeqRat, RealUp seal, and non-escape NameCert rows.

Concretized in §961: Diagonal-limit observers realize the Real-completeness diagonal route as finite StreamName window, RegSeqRat readback, Dyadic tolerance, CauchyLimitSeal, RealUp seal, and observer budget NameCert rows.

Concretized in §979: Real window synchronizers realize the StreamName, RegSeqRat, CauchyLimitSeal, RegularCauchyTailBudget, and RealUp seal route as one finite NameCert packet.

Concretized in §977: Regular Cauchy modulus witness ledgers realize the explicit-modulus route of the Real completeness proof target as finite StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §980: Regular Cauchy classifier normal forms realize the Real-completeness route as a finite DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert packet.

Concretized in §983: Cauchy shared-tail moduli realize the Real-completeness route as one shared dyadic tail threshold, StreamName window, RegSeqRat handoff, RealUp seal, hsame, Cont, Pkg, and NameCert packet.

Concretized in §990: Cauchy tail consensus realizes the Real-completeness route as one finite shared-tail, tail-agreement, CauchyLimitSeal, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert packet.

Concretized in §991: Regular Cauchy apartness budgets realize finite nonzero lower-bound windows for RegSeqRat, DyadicRatCore, StreamName, and RealUp consumers.

Concretized in §1250: Closed bounded interval nets realize finite dyadic mesh coverage for located RealUp intervals before compact-uniform-continuity consumers read the route.

Concretized in §1263: Cofinal StreamName tail selectors realize the finite cofinal-window route from precision requests through StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp seal rows without quotient streams or host-selected limits.

Concretized in §2313: Bishop Cauchy real packets realize the Real completeness route as finite RegSeqRat source, CauchyModulus schedule, StreamName window, DyadicRatCore tolerance, and RealUp seal rows without quotient streams or selected representatives.

Concretized in §3404: Bishop completion criterion packets realize the L10 completion route as explicit CauchyConvergenceModulus, StreamName, RegSeqRat, MetricCompletion, UniformCompletion, RealUp, hsame, Cont, Pkg, and NameCert rows without quotient completions or choice.

Concretized in §3670: Rational Cauchy completion packets realize the L10 completion route as finite Rat source, DyadicRatCore tolerance, StreamName window, RegSeqRat readback, explicit Cauchy modulus, RealUp seal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §3498: Proximinal set packets realize the located-set distance-attainment boundary as finite Metric, LocatedSet, LocatedMetric, RealUp, hsame, Cont, Pkg, and NameCert rows without nearest-point selection.

*Concretized in §**??**:* Metric embeddings give the certified distance-preserving route needed by completion and compact metric consumers.

Concretized in §3949: Regular Cauchy filter functors realize the filter-indexed finite-window route from CauchyFilter and RegularCauchyFilter data through StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp seal rows.

Concretized in §3953: Cauchy modulus normalizers realize explicit monotone threshold normalization before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal reads.

Concretized in §4024: Located compactness is realized as a finite neighbourhood-radius, compact-witness, closed-ball, local-support, and NameCert packet for completion consumers.

Concretized in §371: Regular Cauchy names realize the L10 source-row route from StreamName windows through RegSeqRat readback, DyadicRatCore tolerance ledgers, and RealUp seal rows.

Concretized in §5778: Finite-window functor packet for the Real and Completion spine.

Concretized in §5991: Pointwise finite-window extensionality realizes the Real and Completion StreamName equality route.

Concretized in §6015: Cofinal tail normal forms realize the Real-completeness tail route as finite CofinalTailAgreement, StreamName selector-window, RegSeqRat readback, DyadicRatCore tolerance, RealUp seal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1738: Finite tail-equivalence packets realize the Real completeness vision's regular-Cauchy equality route.

Concretized in §6002: Regular Cauchy diagonal limit packets realize the vision's diagonal extraction route as finite StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and NameCert rows.

Concretized in §6088: Real completeness gap routers separate allowed StreamName, RegSeqRat, DyadicRatCore, and RealUp reads from off-table completion claims.

Concretized in §6114: Finite selection sequences realize the choice-free finite-window selector route for the Real completeness diagonal.

Concretized in §6133: Real completion limit routers realize the finite observation-budget, StreamName, RegSeqRat, DyadicRatCore, and RealUp route without asserting full RealUp completeness.

Concretized in §6279: The finite StreamName–RegSeqRat–Real seal witness ledger realizes the real-completeness vision as NameCert data.

Concretized in §6949: Real completion budget routers realize finite observation-budget and selector-budget routing before StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, and RealUp seal reads.

Concretized in §6405: The finite diagonal scheduler realizes the real-completeness diagonal-window route as displayed modulus and tolerance data.

Concretized in §6440: The selection gate realizes the four-object Real-completion route as finite DyadicRatCore, StreamName, RegSeqRat, RealUp, transport, replay, provenance, and NameCert rows.

Concretized in §6762: Regular Cauchy ball bases realize the Real-completeness tail route as finite threshold, dyadic ball, StreamName, RegSeqRat, RealUp, transport, replay, provenance, and NameCert rows.

Concretized in §6766: Records the explicit modulus-bearing Cauchy completion packet used by the Real and Completion route.

Concretized in §7397: Rapidly Cauchy sequence packets realize the fixed dyadic-rate schedule used by the Real and Completion route through StreamName, DyadicRatCore, RegSeqRat, and RealUp rows.

Concretized in §7402: Regular Cauchy modulus joins realize the finite shared-tail route through paired StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, and the RealUp seal.

Concretized in §7612: Diagonal-limit uniqueness witnesses realize the paired diagonal-representative comparison route as finite StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7983: Located real spread packets realize the Real-completeness diagonal route as finite located-window, dyadic-tail, RegSeqRat readback, StreamName, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8218: Real completeness density windows realize explicit dyadic approximation windows before RegSeqRat and RealUp seal reads.

Concretized in §8426: Finite-window subcovers realize the Real-completeness route through explicit StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, and RealUp seal rows.

Concretized in §774: Finite Bairstow root-window packets realize polynomial coefficient windows, quadratic-factor updates, residual budgets, dyadic brackets, RegSeqRat readback, BisectionRootCauchySeal, and RealUp handoff rows.

Concretized in §8663: Real completion diagonal routers realize the finite precision-window route from StreamName and RegSeqRat data through DyadicRatCore readback to the RealUp seal.

Concretized in §9042: Real prefix diagonal packets realize the finite prefix-window diagonal route through StreamName, RegSeqRat, DyadicRatCore, and RealUp seal rows.

Concretized in §9172: Real tail-budget handoffs realize the Real-completeness route as finite precision-request, explicit-modulus, tail-budget, StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Chapter 9239

The Riemann Hypothesis as a BEDC Conjecture

This capstone formally states the Riemann Hypothesis inside BEDC's distinction-foundationalist framework, using only the naming certificates and operations licensed in the dependency chain spanning LEVEL 1 (the arithmetic certificates $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Int}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Prime}^\uparrow}$), LEVEL 2 (the analytic certificates $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$ together with the regular Cauchy completion of chapter 9238), LEVEL 3 (the complex-analytic limit and convergence-radius certificates $\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{ConvRad}^\uparrow}$), LEVEL 4 (the holomorphy package $\mathsf{NameCert}_{\mathsf{CplxDiff}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Holomorphic}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Contour}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{AnaCont}^\uparrow}$), and LEVEL 5 (the Dirichlet and zeta package $\mathsf{NameCert}_{\mathsf{CplxSer}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Dirichlet}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{ZetaBasic}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{ZetaCont}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{CritStrip}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{ZetaZeros}^\uparrow}$). The chapter does not prove the Riemann Hypothesis. Its purpose is fivefold. First, it makes the conjecture precise as a single BEDC statement-level Lean target. Second, it traces the LEVEL 1 through LEVEL 5 dependency chain explicitly so that the chain of constructive obligations is visible at a glance. Third, it catalogs the witness types that any constructive proof inside BEDC would have to produce. Fourth, it discusses axiom-purity considerations: any RH proof would have to remain inside the BEDC fragment that uses no Classical.choice, no Quot.sound, and no propext, or else expose explicitly the points at which it leaves the framework. Fifth, it connects the conjecture to existing partial BEDC results. A constructive proof of RH is, by Curry-Howard, a function that takes any complex history s in the critical strip with $\zeta(s) = 0$ and produces an explicit witness that $\Re(s) = 1/2$; equivalently, a constructive disproof would produce an explicit s realizing $\mathsf{ZetaZero}(s)$ and $\mathsf{InCritStrip}(s)$ and $\neg\mathsf{OnCritLine}(s)$ as distinct concrete histories. Whether RH is provable inside this fragment is itself an open question, and articulating that open question precisely is the only contribution this capstone makes.

9239.1 LEVEL 1–5 dependency chain

The dependency chain of the conjecture is laid out explicitly here, level by level, because the constructive content of the statement depends on every layer above it. Each level is given as a remark naming the supporting certificates, the responsible chapters, the current closure status (closed ✓, pending □, scaffold-only ∘), and the specific contribution of each chapter to the RH statement.

Remark 9239.1 (LEVEL 1: arithmetic foundation, status ✓/□)**.** *The naming certificates* $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$ *(chapter 28),* $\mathsf{NameCert}_{\mathsf{Int}^\uparrow}$*, and* $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ *(chapter 37) form the checked arithmetic base.*

They supply the generators on which every later layer is built: $\mathsf{Nat}^\uparrow$ *provides indexing for series,* $\mathsf{Int}^\uparrow$ *provides signed exponents, and* $\mathsf{Rat}^\uparrow$ *provides the dense subfield needed for the regular Cauchy completion of* $\mathsf{Real}^\uparrow$*. The certificate* $\mathsf{NameCert}_{\mathsf{Prime}^\uparrow}$ *(chapter 64) remains an open arithmetic certificate until its semantic naming certificate carries* $\mathsf{TheoryGate}(\mathsf{Prime}^\uparrow, \mathsf{scopedClosed})$ *at* $\mathsf{FormalStatus}(\mathsf{Prime}^\uparrow, \mathsf{theoremChecked})$*; once both axes hold, it supplies the index set of the Euler product* $\zeta(s) = \prod_p (1 - p^{-s})^{-1}$*.*

Remark 9239.2 (LEVEL 2: real analysis, status □)**.** *The certificate* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ *(chapter 38) is in place pointwise; its completion to a regular Cauchy carrier is the subject of chapter 9238 and remains a pending capstone. The certificates* $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$ *depend on* $\mathsf{Real}^\uparrow$ *for their distance and convergence structure. RH inherits the constructive content of the real completion: every* $\zeta(s)$ *value inside the critical strip is, ultimately, a regular Cauchy sequence over* $\mathsf{Rat}^\uparrow$ *with an explicit modulus.*

Remark 9239.3 (LEVEL 3: complex limit and radius, status ○)**.** *The certificates* $\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$ *(chapter 65) and* $\mathsf{NameCert}_{\mathsf{ConvRad}^\uparrow}$ *(chapter 66) are scaffolds awaiting elaboration. They lift the regular Cauchy machinery from* $\mathsf{Real}^\uparrow$ *to* $\mathsf{Complex}^\uparrow$*, and they fix what it means for a complex power series to converge with an explicit modulus. RH needs both: zeta-as-Dirichlet-series converges in* $\Re(s) > 1$*, and the analytic continuation past* $\Re(s) = 1$ *requires a complex-limit notion that is constructively well-behaved in the critical strip.*

Remark 9239.4 (LEVEL 4: holomorphy, status ○)**.** *The certificates* $\mathsf{NameCert}_{\mathsf{CplxDiff}^\uparrow}$ *(chapter 67, packaging the Cauchy-Riemann equations as a constructive predicate),* $\mathsf{NameCert}_{\mathsf{Holomorphic}^\uparrow}$ *(chapter 68, packaging holomorphy as a uniform-on-compacts version of complex differentiability),* $\mathsf{NameCert}_{\mathsf{Contour}^\uparrow}$ *(the contour-integral operation), and* $\mathsf{NameCert}_{\mathsf{AnaCont}^\uparrow}$ *(the analytic-continuation operation) are scaffolds. RH lives inside this layer:* ζ *is holomorphic on* $\mathsf{Complex}^\uparrow \setminus \{1\}$*, its zero set is invariant under contour deformation, and the critical strip is the natural domain of analytic continuation from* $\Re(s) > 1$*.*

Remark 9239.5 (LEVEL 5: Dirichlet series and zeta, status ○)**.** *The certificates* $\mathsf{NameCert}_{\mathsf{CplxSer}^\uparrow}$ *(chapter 69),* $\mathsf{NameCert}_{\mathsf{Dirichlet}^\uparrow}$ *(chapter 70),* $\mathsf{NameCert}_{\mathsf{ZetaBasic}^\uparrow}$ *(chapter 71), the* $\mathsf{NameCert}_{\mathsf{ZetaCont}^\uparrow}$ *application (chapter 9342),* $\mathsf{NameCert}_{\mathsf{CritStrip}^\uparrow}$ *(chapter 72), and* $\mathsf{NameCert}_{\mathsf{ZetaZeros}^\uparrow}$ *(chapter 73) are scaffolds. They specialize the LEVEL 4 holomorphy package to the specific function* ζ *and to the specific predicates* $\mathsf{InCritStrip}$, $\mathsf{ZetaZero}$, $\mathsf{OnCritLine}$ *that appear in the conjecture. The RH statement of Definition 9239.7 is a single quantification over the histories carried by* $\mathsf{NameCert}_{\mathsf{ZetaZeros}^\uparrow}$*.*

Concretized in §72: The critical-strip dependency in the RH formulation is realized as the finite $\mathsf{CritStrip}^\uparrow$ NameCert surface.

Remark 9239.6 (Reading the chain)**.** *Each level depends only on the levels strictly below it. The dependency is acyclic; the order in which a constructive RH proof must be assembled is exactly the order of Definition 9239.1 through Definition 9239.5. No level can be skipped. The closure status column makes visible the current obligation: LEVEL 1 has a closed* $\mathsf{Nat}^\uparrow/\mathsf{Int}^\uparrow/\mathsf{Rat}^\uparrow$ *base with* $\mathsf{Prime}^\uparrow$ *still pending, LEVEL 2 is one capstone away from closure, and LEVELs 3 through 5 are scaffolds whose elaboration is the bulk of the work between this capstone and a finished RH statement.*

9239.2 Formal statement of RH in BEDC

Definition 9239.7 (RH predicate)**.** *The Riemann Hypothesis* RH *is the proposition that for every complex history* s *carried by* $\mathsf{NameCert}_{\mathsf{Complex}^\uparrow}$ *such that* $\mathsf{InCritStrip}(s)$ *holds (chapter 72:* $0 <$

$\Re(s) < 1$, *presented as a constructive predicate on the regular Cauchy representative of* $\Re(s)$*) and such that* $\mathsf{ZetaZero}(s)$ *holds (chapter 73: the regular Cauchy representative of* $\zeta(s)$ *is* hsame *to the zero history of* $\mathsf{NameCert}_{\mathsf{Complex}^\uparrow}$*), the predicate* $\mathsf{OnCritLine}(s)$ *holds, where* $\mathsf{OnCritLine}(s)$ *is the constructive predicate stating that the regular Cauchy representative of* $\Re(s)$ *is* hsame *to the rational history* $1/2$*. The predicate* RH *is a* Π*-statement over the carrier of* $\mathsf{NameCert}_{\mathsf{ZetaZeros}^\uparrow}$ *restricted by* $\mathsf{NameCert}_{\mathsf{CritStrip}^\uparrow}$.

Proposition 9239.8 (Constructive form: zero-by-zero certificate)**.** *A constructive proof of* RH *is, by Curry-Howard, a total function* $\Phi : \{\, s \mid \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \,\} \to \{\, \textit{witness of } \mathsf{OnCritLine}(s) \,\}$ *that takes any zero of* ζ *in the critical strip and produces an explicit* hsame*-witness identifying the regular Cauchy representative of* $\Re(s)$ *with the rational history* $1/2$*. Equivalently, a constructive disproof of* RH *is an explicit complex history* s *together with three concrete witnesses* $\mathsf{ZetaZero}(s)$*,* $\mathsf{InCritStrip}(s)$*, and* $\neg\mathsf{OnCritLine}(s)$*, the last being a positive lower bound on the distance from the regular Cauchy representative of* $\Re(s)$ *to* $1/2$*.*

Remark 9239.9 (Why the constructive form is strictly stronger)**.** *A classical proof of* RH *that proceeds by contradiction (assume a zero off the critical line, derive a contradiction) does not, on its own, supply the function* Φ *of Definition 9239.8. Inside BEDC the constructive form is the only form available: by chapter 9458, the framework simulates classical reasoning by contradiction but does not entail it on existential conclusions. Any RH proof landing inside the BEDC fragment therefore has to produce* Φ *explicitly, or expose precisely where it appeals to a meta-closure move outside the fragment.*

9239.3 Carrier-stack reading

Definition 9239.10 (Top-down BEDC-RH stack)**.** *The top-down BEDC reading of the conjecture starts with the carrier containment*

$$\mathbb{Z}_\zeta \subseteq \mathbb{H}_\zeta$$

of Definition 9246.3. The ambient carrier is the pre-RH recovered zeta carrier $\mathbb{G}^0_\zeta$*, whose rows are the Dirichlet projection, Euler projection, analytic-continuation row, archimedean-completion row, functional-equation row, pole ledger, zero ledger, and branch-transport ledger of Definition 9254.13. The fixed-half subcarrier then admits the equivalent readouts*

$$\Re(s) = 1/2, \qquad J(s) = s, \qquad \forall p \in \mathsf{Prime}^\uparrow,\ |p^{1/2-s}|\mathsf{hsame1},$$

$$\delta = 0, \qquad \textit{no } e^{\delta x} \textit{ Laplace bias}, \qquad \mathsf{OnCritLine}(s),$$

as recorded in Definition 9254.10. Below these readouts sit the prime-coordinate and exponential-kernel rows, the prime-fingerprint separation row, the local finite prime-window defect rows, and the Axis-Zeckendorf finite-window laboratory used only as a bridge prototype.

Principle 9239.11 (Stack order is not proof order collapse)**.** *The stack of Definition 9239.10 is an ordering of interfaces, not a proof of* RH*. The upper containment*

$$\mathbb{Z}_\zeta \subseteq \mathbb{H}_\zeta$$

is the same target as Definition 9239.7 by Definition 9246.7. The mirror, prime-unitary, information-channel, and Laplace readings give equivalent fixed-half surfaces, but none of them proves that a zero ledger must factor through that surface. The finite Window-six and Zeckendorf layers provide auditable models of no-adjacent geometry and residual-return discipline; by Definition 9349.9, they do not construct the missing zeta-to-residual bridge by themselves.

Theorem 9239.12 (Defect incompatibility is the remaining stack hinge)**.** *Assume the located real-bias readout, the finite prime-window defect formula, source compatibility into the completed carrier, and the carrier-level row Definition 9246.13. Then the top-down stack of Definition 9239.10 supplies the BEDC-RH witness target.*

Proof. The finite prime-window defect formula turns a positive off-fixed witness into a positive finite prime-window radial defect. Source compatibility and completion place that defect in the same completed zeta carrier row as the zero ledger. Applying Definition 9246.13 yields $\mathsf{Apart}_0(\zeta(s))$. This is the local-global zero exclusion output assembled in Definition 9246.20. Its contradiction with a zero-ledger witness gives the fixed-half section target, hence the BEDC-RH witness target by Definition 9254.4. □

9239.4 Witness types a proof would produce

A constructive proof of RH exposes constructive content beyond the bare conjunction. The witness types below catalog this content as Lean statement-level targets that any RH proof would have to discharge or supply. Each is a refinement or corollary of Definition 9239.8.

Theorem 9239.13 (Modulus from zero-on-strip to half)**.** *A constructive proof of* RH *supplies, for every* s *with* $\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s)$*, an explicit modulus function* $\mu_s : \mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$ *such that for every* n*, the rational representative* q_n *of* $\Re(s)$ *at depth* $\mu_s(n)$ *satisfies* $|q_n - 1/2| < 2^{-n}$ *as a* $\mathsf{Rat}^{\uparrow}$*-level inequality. The function* μ_s *is the constructive content of the* hsame*-witness for* $\Re(s) = 1/2$*.*

Concretized in §548: Critical-line witness packets realize the RH constructive witness type as finite rows over $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Theorem 9239.14 (Approach-rate bound near the critical line)**.** *A compact-window strengthening of a constructive* RH *route may expose a height-local bound*

$$\beta : \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow} \to \mathsf{Rat}^{\uparrow}$$

with $\beta(n, T) > 0$ *such that every complex history* s *in the critical strip with* $|\Im(s)| \leq T$ *and* $|\Re(s) - 1/2| > 2^{-n}$ *has a constructive lower bound*

$$|\zeta(s)| \geq \beta(n, T).$$

The height parameter is part of the statement: compactness applies to the displayed window only. A height-uniform global $\beta(n)$ *would be an additional analytic barrier stronger than the bare RH witness and is not treated here as an ordinary output of a proof of* RH*.*

Concretized in §705: Critical-strip barrier packets realize the RH approach-rate witness as finite zero-localization, rational lower-bound, transport, route, package, and NameCert rows.

Theorem 9239.15 (Riemann-von Mangoldt zero count, prerequisite ledger)**.** *An unconditional analytic prerequisite ledger may provide an explicit zero-counting function* $N : \mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$ *together with a witness, for every imaginary height* $T \in \mathsf{Nat}^{\uparrow}$*, of a* Unary *upper bound on the cardinality of* $\{\, s \mid \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \wedge |\Im(s)| \leq T \,\}$ *that agrees, up to an explicit constructive error term, with the Riemann-von Mangoldt formula* $N(T) \approx (T/2\pi)\log(T/2\pi) - T/2\pi$*. This belongs to a complete zeta formalization, not to the Curry-Howard witness for* RH *itself.*

Concretized in §573: Riemann zero-count ledgers realize the unconditional height-window, zero-list, unary-count, error-ledger, and NameCert rows supporting a complete zeta formalization.

Theorem 9239.16 (Trivial-zero classifier, supporting chapter)**.** *The trivial-zero classifier is an unconditional support row: the negative even integers $-2, -4, -6, \ldots$ are classified through the functional equation and the arithmetic of negative even integer inputs, independently of* RH*. The classifier takes the form of a* psame*-respecting predicate on* $\mathsf{NameCert}_{\mathsf{ZetaZeros}^\uparrow}$ *such that the trivial-zero subset is exactly the image of the negative-even-integer carrier under the embedding* $\mathsf{Int}^\uparrow \hookrightarrow \mathsf{Complex}^\uparrow$*. The RH predicate then consumes the non-trivial zero interface after this separation is already in place.*

Concretized in §999: Trivial-zero classifiers are realized as unconditional finite negative-even-integer, zeta-zero, real-part, rational-comparison, hsame, Cont, Pkg, and NameCert rows.

Theorem 9239.17 (Critical-line restriction of the zero set)**.** *A constructive proof of* RH *supplies a* psame*-isomorphism between the carrier of* $\{\, s \mid \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \,\}$ *and the carrier of* $\{\, t \in \mathsf{Real}^\uparrow \mid \mathsf{ZetaZero}(1/2 + it) \,\}$*, exhibiting that every non-trivial zero is determined by its imaginary part alone. The isomorphism is the conjunction of* Φ *from Definition 9239.8 with the projection to imaginary part.*

9239.5 Axiom-purity considerations

The supporting chapters of LEVELs 1 through 5 are required to remain axiom-pure in the BEDC sense: no use of Classical.choice, no use of Quot.sound, no use of propext. Any RH statement-level target sits on top of this entire stack, so the same gate applies to it transitively. Three outcomes are mathematically meaningful, and the RH roadmap distinguishes them as the only three things a constructive engagement with the conjecture can land on.

Remark 9239.18 (Outcome (i): RH is provable in the fragment)**.** *A construction* Φ *of Definition 9239.8 is exhibited inside BEDC and the entire dependency chain runs under* check-axioms.py *and* bedc_ci.py axiom-purity --strict *without flagging any stdlib axiom. This is the strongest possible outcome and would constitute a genuine constructive proof of RH.*

Remark 9239.19 (Outcome (ii): independence inside the fragment)**.** *A meta-theorem is exhibited stating that neither* RH *nor* $\neg$RH *is provable in the BEDC fragment. This outcome would not require the supporting chapters to leave axiom-purity but would require a meta-level layer above the kernel. By chapter 9461, that meta layer is structurally available, and the independence statement would be a meta-level* `leanstmt` *record, not a closure of* RH *at the kernel level. This is the "schema-only horizon" outcome: BEDC exposes the conjecture as an interface that codifies the constructive content but admits no closed witness inside the framework.*

Remark 9239.20 (Outcome (iii): explicit constructive counterexample)**.** *A specific complex history* s_0 *is exhibited together with explicit witnesses* $\mathsf{ZetaZero}(s_0)$*,* $\mathsf{InCritStrip}(s_0)$*, and a positive* $\mathsf{Rat}^\uparrow$*-level lower bound on* $|\Re(s_0) - 1/2|$*. By Definition 9239.8, this is the constructive disproof. It would settle RH negatively, and unlike a classical disproof it would carry the actual offending zero as data.*

Remark 9239.21 (The fragment may not decide which outcome)**.** *The BEDC fragment is constructive and axiom-pure; it is not, in general, complete. By Definition 9461.18, the open meta loop of the framework guarantees that some statements over BEDC primitives are formally undecidable inside the fragment. Whether* RH *is one such statement is itself open. Articulating this open question is the purpose of the present capstone.*

9239.6 Connection to existing BEDC content

Remark 9239.22 (Reasoning by contradiction in RH)**.** *chapter 9458 establishes that BEDC reasoning by contradiction has a boundary: refutation of a false existence claim by exhibited counterexample stays inside the closed ground loop and is accepted, but proof of an existence statement by contradiction is not. The classical literature on RH is full of contradiction arguments. A BEDC translation of any such argument has to respect this boundary: refutational content stays, but any existence conclusion has to be replaced by an explicit witness or recategorized as a meta-closure move that exits the fragment.*

Remark 9239.23 (Reflection and limits)**.** *chapter 9461 fixes the level structure: the BEDC kernel lives at the ground, the meta layer describes the kernel from above, and Tarski undefinability prevents the meta layer from closing. The conjecture* RH *lives at the kernel level. Any proof of* RH *is a kernel-level theorem. Any meta-theorem about the provability of* RH *inside the fragment (Definition 9239.19) lives at the meta level. The two are different artifacts and do not collapse into one another; conflating them would be a Tarski-undefinability move and is forbidden by the framework.*

Remark 9239.24 (Real completeness dependency)**.** *chapter 9238 is the immediate prerequisite of LEVEL 2 closure. Every $\zeta(s)$ value inside the critical strip is, ultimately, a regular Cauchy sequence over* $\mathsf{Rat}^{\uparrow}$*, and the closure of chapter 38 into a complete carrier through that capstone is what licenses talking about ζ-values and zeros at all in the constructive sense. RH inherits the constructive content of the real completion: until the regular Cauchy completion of* $\mathsf{Real}^{\uparrow}$ *closes, the RH statement of Definition 9239.7 type-checks but is not yet inhabitable in any nontrivial way.*

Remark 9239.25 (Curry-Howard placement)**.** *The constructive form of Definition 9239.8 is what Curry-Howard buys at the BEDC fragment level:* RH *as a Π-type, a proof as a function, a disproof as a sigma-type witness. This is the same Curry-Howard discipline applied in every other concrete-instance certificate of LEVELs 1 through 5. RH is special only in that the inhabiting function or sigma-witness is conjectured but not yet supplied; the typing is identical to every other constructive statement in the manuscript.*

9239.7 Roadmap and scope

This capstone is a roadmap, not a proof. The conjecture RH has been stated as Definition 9239.7 and refined to its constructive form as Definition 9239.8; the witness types of Definition 9239.13 through Definition 9239.17 catalog the constructive content any proof would expose; the dependency chain of section 9239.1 fixes the order in which the supporting concrete-instance chapters must close before the RH statement can be inhabited; and section 9239.5 fixes the three mathematically meaningful outcomes inside the BEDC fragment. The supporting LEVEL 3 through LEVEL 5 chapters are scaffolds awaiting codex pipeline elaboration. A rough scale estimate: twelve concrete-instance chapters across LEVELs 3, 4, and 5, each carrying eight to twelve Lean targets, totaling on the order of one hundred and twenty new theorems plus the RH statement-level targets of the present capstone. Even when the entire scaffold is closed and the chain is fully formalized, the Riemann Hypothesis remains the foundation for a future research program rather than a finished result; the roadmap clarifies what such a program would have to produce, in what order, and under what axiom-purity constraints. The chapter uses no Classical.choice, no Quot.sound, and no propext; whether RH itself is provable inside this fragment is part of what this roadmap is intended to clarify.

Concretized in §550: Critical-strip zeta-zero witness packets realize the RH capstone as a finite NameCert obligation surface.

Concretized in §659: Non-trivial zero classifiers are realized as finite ZetaZerosUp/CritStripUp boundary packets separating critical-strip zero witnesses from trivial-zero rows without host zero-set quotients.

Concretized in §521: Contour-integral operation packets realize the RH holomorphy dependency as piecewise-linear contour rows, Riemann-sum ledgers, operation-law rows, hsame transport, Cont replay, Pkg provenance, and NameCert data.

Chapter 9240

Three Boundaries for RH

chapter 9239 states RH as a constructive function from certified critical-strip zero packets to OnCritLine witnesses. The nearby RH chapters also use the functional-equation mirror, prime-unitary channels, finite prime windows, and route-closure boundaries. The present chapter fixes the separation between three objects that should not be conflated: the geometric critical line, an exact zero decision operator, and the global rigidity statement called RH.

Definition 9240.1 (Mirror fixed surface)**.** *The functional-equation mirror is*

$$J(s) := 1 - \overline{s}.$$

The geometric boundary is the fixed surface

$$\mathsf{GeomBoundary}_\zeta := \mathrm{Fix}(J) := \sum_s J(s) \sim s.$$

Equivalently, the pointwise predicate $\mathsf{CritLine}(s)$ *is* $J(s) \sim s$*.*

Lean4 (definition) `BEDC.Derived.RHRoute.FunctionalEquationSymmetry.reflectJ`
Lean4 (definition) `BEDC.Derived.RHRoute.FunctionalEquationSymmetry.JFixed`
Lean4 (definition) `BEDC.Derived.RHRoute.FunctionalEquationSymmetry.CriticalLine`

Lemma 9240.2 (The mirror fixed surface is the critical line)**.** *For every complex history* s*,*

$$J(s) \sim s \quad \Longleftrightarrow \quad \Re(s) \sim_{\mathsf{Real}^\uparrow} 1/2.$$

Proof. Write $s = \sigma + it$. Then

$$J(s) = 1 - \overline{s} = 1 - \sigma + it.$$

Thus $J(s) \sim s$ is equivalent to equality of real parts

$$1 - \sigma \sim \sigma,$$

which is equivalent to $\sigma \sim 1/2$. The imaginary component is unchanged by the calculation. This is the same fixed-point computation used in Definition 9241.12. □

Lean4 ✓ `BEDC.Derived.RHRoute.FunctionalEquationSymmetry.J fixed iff re half`

Definition 9240.3 (Certified zero predicate and exact zero decider)**.** *The certified zeta-zero predicate is the packet*

$$\mathsf{ZetaZero}(s) := \sum_z \big(\mathsf{ZetaVal}(s,z) \times z \sim_{\mathsf{Complex}^\uparrow} 0\big),$$

read through the recovered zeta carrier and its continuation, branch, transport, and zero-ledger rows. The exact zero decision operator is the stronger object

$$\mathsf{ZeroDecider}_\zeta := \prod_s \mathsf{Decidable}\big(\mathsf{ZetaZero}(s)\big).$$

The first object is a certificate-bearing predicate; the second is a total decision procedure for that predicate on all complex histories.

Definition 9240.4 (Rigidity boundary)**.** *The rigidity boundary is the RH proposition*

$$\mathsf{RigidityBoundary}_\zeta := \prod_s \big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnCritLine}(s)\big).$$

Equivalently, it asserts that the support of the certified non-trivial zero predicate factors through $\mathrm{Fix}(J)$*.*

Theorem 9240.5 (Three-boundary separation)**.** *The geometric boundary* $\mathrm{Fix}(J)$*, the exact zero decision operator* $\mathsf{ZeroDecider}_\zeta$*, and the rigidity boundary* $\mathsf{RigidityBoundary}_\zeta$ *are distinct kinds of data.*

Proof. The geometric boundary is defined only from the mirror map J and the complex-history structure. By Definition 9240.2, it is the critical line, and it does not mention $\mathsf{ZetaZero}$.

The exact zero decision operator has type

$$\prod_s \mathsf{Decidable}\big(\mathsf{ZetaZero}(s)\big).$$

It must decide, for every complex history, whether a certified zero packet exists. This is stronger than the certified zero predicate itself, which only records evidence when a zero packet is supplied.

The rigidity boundary has type

$$\prod_s \big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnCritLine}(s)\big).$$

It does not decide whether arbitrary histories are zeros. It consumes an already certified critical-strip zero packet and returns a fixed-surface witness. Hence the three objects have different inputs, outputs, and proof obligations. □

Corollary 9240.6 (The critical line is the target fixed surface)**.** *The critical line is available before the RH proposition is stated. RH is the assertion that the certified non-trivial zero support lands in that fixed surface:*

$$\mathrm{Supp}(\mathsf{ZetaZero} \times \mathsf{InCritStrip}) \subseteq \mathrm{Fix}(J).$$

Proof. The fixed surface is defined in Definition 9240.1. The RH predicate of Definition 9239.7 is the map from a certified critical-strip zero packet to $\mathsf{OnCritLine}(s)$, and Definition 9240.2 identifies that output with membership in $\mathrm{Fix}(J)$. □

Definition 9240.7 (Prime-unitary readout boundary)**.** *For $a > 1$, define the centered channel*

$$U_a(s) := a^{1/2-s}.$$

For displayed prime rows, the prime-unitary readout is

$$\mathsf{PrimeUnitary}(s) := \prod_{p:\mathsf{Prime}^{\uparrow}} (|U_p(s)|\mathsf{hsame1}).$$

Theorem 9240.8 (Prime-unitary balance is a critical-line readout)**.** *For every displayed $a > 1$,*

$$|a^{1/2-s}| = 1 \quad \Longleftrightarrow \quad \Re(s) = 1/2.$$

Consequently, $\mathsf{PrimeUnitary}(s)$ *is equivalent to* $\mathsf{OnCritLine}(s)$ *on the displayed prime packet.*

Proof. Write $s = \sigma + it$. Then

$$a^{1/2-s} = a^{1/2-\sigma-it} = a^{1/2-\sigma}a^{-it}.$$

The factor a^{-it} has modulus one, so

$$|a^{1/2-s}| = a^{1/2-\sigma}.$$

For $a > 1$, this modulus is one exactly when $1/2 - \sigma = 0$, namely $\Re(s) = 1/2$. The prime-indexed version is Definition 9243.7. □

Principle 9240.9 (Prime-unitary balance is not the RH mechanism)**.** *Prime-unitary balance is an exact readout of the critical line, not by itself a proof mechanism for RH. The calculation of Definition 9240.8 works for every displayed $a > 1$ and therefore does not use zero ledgers, Euler-product support, finite-window residual control, or any incompatibility between an off-line zero packet and the recovered zeta carrier. It becomes RH-facing only when connected to certified non-trivial zero packets by additional bridge data.*

Definition 9240.10 (Finite prime imbalance bridge)**.** *A finite prime imbalance certificate*

$$\mathsf{FinitePrimeImbalanceCert}(s, W)$$

records that a finite prime window W constructively observes a nonzero deviation of s from the fixed-half surface. A tail bound

$$\mathsf{TailBound}(W, s)$$

records that the prime or zeta residual outside W is controlled by an explicit modulus and cannot hide a cancellation of the finite defect.

An RH bridge is the displayed equivalence

$$\mathsf{RHBridge} := \prod_s \left(\mathsf{OffLineZeroCert}(s) \simeq \sum_W (\mathsf{FinitePrimeImbalanceCert}(s, W) \times \mathsf{TailBound}(W, s)) \right),$$

where

$$\mathsf{OffLineZeroCert}(s) := \mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \neg\mathsf{OnCritLine}(s).$$

Theorem 9240.11 (Finite imbalance bridge gives the contradiction route)**.** *Assume* RHBridge*, a finite-window incompatibility row*

$$\prod_{s,W} (\mathsf{ZetaZero}(s) \times \mathsf{FinitePrimeImbalanceCert}(s, W) \times \mathsf{TailBound}(W, s) \to \bot),$$

and a located critical-line classifier

$$\prod_{s} (\mathsf{OnCritLine}(s) + \neg\mathsf{OnCritLine}(s)).$$

Then the BEDC RH predicate is inhabited.

Proof. Fix s with

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s).$$

Apply the located critical-line classifier. If it gives $\mathsf{OnCritLine}(s)$, the required witness is present. Otherwise the zero packet, critical-strip packet, and non-line witness form $\mathsf{OffLineZeroCert}(s)$. By RHBridge, this yields a finite window W with

$$\mathsf{FinitePrimeImbalanceCert}(s, W) \times \mathsf{TailBound}(W, s).$$

Together with $\mathsf{ZetaZero}(s)$, the finite-window incompatibility row gives $\bot$. Hence the non-line branch is impossible, and the surviving branch supplies $\mathsf{OnCritLine}(s)$. □

Principle 9240.12 (Axiom footprint is not a necessity lower bound)**.** *A Lean axiom printout for one implementation path records that path's dependency footprint. It does not quantify over all possible implementations and therefore does not show that zeta, zero predicates, or RH carry a mathematical need for* Classical.choice*,* Quot.sound*, or* propext*. In the BEDC discipline, any use of those constants is an exposed implementation dependency, while the RH target remains the axiom-pure construction of a uniform witness*

$$\Phi : \{\, s \mid \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \,\} \to \{\, \mathsf{OnCritLine}(s) \ \textit{witness} \,\}.$$

Concretized in §8864: The geometric critical-line surface, certified zero predicate, and RH rigidity boundary are separated as finite NameCert rows.

Concretized in §8868: finite critical-strip window certificate separating mirror geometry from exact zero decision and global RH rigidity

Chapter 9241

The RH Zero–Skew EventFlow Target

chapter 9239 states the Riemann Hypothesis as a constructive Π-statement: a proof consumes a critical-strip zeta-zero witness and returns an $\mathsf{OnCritLine}$ witness. The present chapter records the same obligation at the compiler-facing event-flow layer. It does not add a proof of RH. It identifies which rows are already recoverable from the zeta-continuation and functional-equation ledgers, which rows are supplied by the critical-strip zero witness packet, and which row is still the missing RH witness.

9241.1 Zero and skew packets

Definition 9241.1 (RH zero code)**.** *An* RH zero code *for a complex history s is a finite packet containing the displayed rows*

$$s, \quad \mathsf{ZetaCont}(s), \quad \mathsf{hsame}(\mathsf{ZetaCont}(s), 0_{\mathbb{C}}), \quad \mathsf{InCritStrip}(s),$$
$$H, \quad C, \quad P, \quad N.$$

The rows H, C, P, N record componentwise transport, continuation replay, package provenance, and local naming. The packet is not a host-side element of a zero set. It is a finite BEDC read of a zeta-zero premise through the $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, and $\mathsf{CritStrip}^{\uparrow}$ surfaces.

Definition 9241.2 (RH skew-zero code)**.** *For the same complex history s, the associated* RH skew-zero code *is a finite packet containing*

$$s, \quad \Re(s), \quad 1/2, \quad \delta(s) = \Re(s) - 1/2, \quad 0_{\mathbb{R}},$$
$$\mathsf{hsame}(\delta(s), 0_{\mathbb{R}}), \quad H, \quad C, \quad P, \quad N.$$

Thus the skew-zero packet is the critical-line witness written as a real offset from the line rather than as a direct equality $\Re(s) = 1/2$.

Proposition 9241.3 (Zero–skew form of RH)**.** *The BEDC statement RH of Definition 9239.7 is equivalent to the existence of a total transformer*

$$\Phi_{\mathsf{RH}} : \prod_{s} \mathsf{ZeroCode}_{\mathsf{RH}}(s) \to \mathsf{SkewZeroCode}_{\mathsf{RH}}(s).$$

Proof. By Definition 9239.7, the premise of RH is exactly the displayed conjunction $\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s)$, read through the zeta-zero and critical-strip packets. This is the semantic content of $\mathsf{ZeroCode}_{\mathsf{RH}}(s)$. The conclusion $\mathsf{OnCritLine}(s)$ says that the real component of s is hsame to

the rational half. Equivalently, the skew $\delta(s) = \Re(s) - 1/2$ is hsame to $0_{\mathbb{R}}$, which is precisely the additional row carried by $\mathsf{SkewZeroCode}_{\mathsf{RH}}(s)$. Under the Curry–Howard reading, an inhabitant of RH is therefore exactly a total function from the first packet type to the second packet type, uniformly in s. □

9241.2 EventFlow reading

Definition 9241.4 (RH zero event flow)**.** *An* RH zero event flow *for s is an event flow $S_0(s)$ together with a generated recognizer flow R_0 such that the judgment*

$$\mathsf{RecognizesZeroCode}(R_0, S_0(s), s)$$

holds. The recognized roles of $S_0(s)$ include a complex-history row for s, a recovered $\mathsf{ZetaCont}^{\uparrow}$ value row, an evaluation route for $\mathsf{ZetaCont}(s)$, a complex-zero row, an hsame row identifying the value with $0_{\mathbb{C}}$, an $\mathsf{InCritStrip}(s)$ row, and the H, C, P, N provenance rows.

Lean isolates the skew substrate with
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.SkewBalanceRow`
,
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.zero skew`
, and
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.zero skew eventflow`
. The finite-balance zero case is checked by
Lean4 ✓ `BEDC.Derived.RHRoute.EventflowCertificate.zero skew nil`
.

Definition 9241.5 (RH skew event flow)**.** *An* RH skew event flow *for s is an event flow $S_{\mathrm{skew}}(s)$ together with a generated recognizer flow R_{skew} such that*

$$\mathsf{RecognizesSkewZero}(R_{\mathrm{skew}}, S_{\mathrm{skew}}(s), s)$$

holds. The recognized roles include the real-part extraction row, the rational half row, the skew row $\delta(s) = \Re(s) - 1/2$, the real-zero row, the $\mathsf{hsame}(\delta(s), 0_{\mathbb{R}})$ row, and the same H, C, P, N support rows.

9241.3 Channel and recognition boundary

Proposition 9241.6 (Terminators are not zero records)**.** *The channel glyph pair* 11 *is only the event terminator of Definition 9208.2. It is not a complex zero, not a real zero, not a zeta zero, and not a proof row.*

Proof. By Definition 9208.3, a raw event w is encoded as

$$\mathsf{Ev}(w) = \zeta(w)\,11.$$

By Definition 9208.6, the body $\zeta(w)$ contains no adjacent 11, so the decoder reads the first adjacent 11 only as the end of the current raw event. The empty raw event already satisfies $\mathsf{Ev}(\epsilon) = 11$, which shows that the glyph pair alone cannot carry a zeta-zero reading. A zero reading requires a decoded event flow together with a generated recognizer judgment such as $\mathsf{RecognizesZeroCode}(R_0, S_0(s), s)$. □

Corollary 9241.7 (Terminator count is event count)**.** *If an RH zero event flow contains n raw events, then its legal channel encoding contains n event terminators. This count is a count of encoded event boundaries, not a count of zeta zeros.*

Proof. The flow encoding of Definition 9208.9 appends one terminator to each raw event and no terminator outside the event encodings. Thus the number of terminators is the number of raw events in the decoded flow. The zeta-zero reading is attached only to the whole recognized packet, not to each terminator. □

Proposition 9241.8 (Channel decodability is not self-reference)**.** *The bijection*

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}$$

of Definition 9208.19 is a unique-decodability result, not a self-reference principle.

Proof. The round-trip theorem

$$\mathsf{Decode}(\mathsf{Compile}(S)) = \mathsf{some}(S)$$

asserts that event boundaries can be recovered from a legal channel stream. It does not assert that the stream recognizes, compiles, proves, or interprets itself. Self-reference would require a semantic judgment at the recognizer or compiler layer, for example a recognized compiler flow whose behavior is related to its own event-flow code. The terminator 11 supplies only channel-level punctuation. □

Proposition 9241.9 (Display glyphs are not mathematical zeros)**.** *The displayed glyph* 0 *in the GroundCompiler alphabet is not* $0_{\mathbb{C}}$*, not* $0_{\mathbb{R}}$*, not a natural-number zero, and not a zeta-zero assertion.*

Proof. Definition 9207.1 fixes 0 and 1 as presentation glyphs for the two raw marks. The display convention does not interpret a glyph as a number, truth value, theorem, or analytic value. A complex-zero row or real-zero row appears only after a source-level recognizer assigns that role to an event flow and the corresponding certificate rows are accepted. Therefore the RH target is not a glyph-level implication $0 \to 0$; it is the recognized semantic implication from a complex-value zero row to a real-skew zero row. □

Definition 9241.10 (RH compiler-flow target)**.** *An* RH compiler-flow target *is a certified compiler flow C_{RH} such that, for every complex history s and every RH zero event flow $S_0(s)$, there exists an RH skew event flow $S_{\mathrm{skew}}(s)$ satisfying*

$$\mathsf{Compiles}_{C_{\mathsf{RH}}}(S_0(s), S_{\mathrm{skew}}(s)).$$

Theorem 9241.11 (Compiler-flow target implies BEDC-RH)**.** *If an RH compiler-flow target exists and its recognizer exports are accepted through the GroundCompiler certificate gate, then the BEDC statement* RH *holds.*

Proof. Let s be a complex history with $\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s)$. By the premise packet discipline of chapter 550, this hypothesis is represented by an accepted RH zero event flow $S_0(s)$, recognized by a generated recognizer R_0. Applying the compiler flow C_{RH} gives an output event flow $S_{\mathrm{skew}}(s)$ recognized as an RH skew event flow. The recognition judgment for the output supplies $\mathsf{hsame}(\Re(s) - 1/2, 0_{\mathbb{R}})$. This is the $\mathsf{OnCritLine}(s)$ witness of Definition 9239.7. Since s was arbitrary, the required Π-statement is inhabited. □

9241.4 Mirror closure and orbit collapse

Definition 9241.12 (Mirror map)**.** *For a complex history s, write*

$$J(s) = 1 - \overline{s}.$$

The fixed points of J are exactly the histories whose real component is $1/2$:

$$s = J(s) \quad \Longleftrightarrow \quad \Re(s) = 1/2.$$

Theorem 9241.13 (Functional-equation rows give mirror closure)**.** *Assume a zeta event-flow packet contains the $\mathsf{ZetaCont}^{\uparrow}$ row, the functional-equation row, the completed-zeta row, conjugation transport, pole non-escape, and Gamma-boundary ledgers. Then it can support a mirror compiler-flow target of the form*

$$\mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to \mathsf{ZeroEventFlow}_{\mathsf{RH}}(J(s)).$$

Proof. Recognition of the input zero event flow reads $\mathsf{ZetaCont}(s) = 0$ inside the critical strip. The completed-zeta row transports this zero read to the completed carrier, the functional-equation row reflects the completed carrier across $s \mapsto 1 - s$, and conjugation transport supplies the real-structure step. The pole and Gamma-boundary ledgers are exactly the non-escape rows needed to return from the completed carrier to the non-trivial zeta-zero surface. The resulting output packet is therefore a recognized zero event flow for $J(s)$. □

Corollary 9241.14 (Mirror closure is weaker than RH)**.** *The existence of the mirror compiler-flow target of Definition 9241.13 does not, by itself, prove* RH.

Proof. Mirror closure transports a zero at s to a zero at $J(s)$. It does not identify the two histories. Since $J^2(s) = s$, a mirror orbit has the form $\{s\}$ or $\{s, J(s)\}$; the first case is equivalent to $\Re(s) = 1/2$, while the second is a two-point orbit off the critical line. The Riemann Hypothesis asserts that every non-trivial zero orbit is of the first kind. That orbit-collapse row is not supplied by mirror closure alone. □

Definition 9241.15 (Zero-orbit collapse packet)**.** *A* zero-orbit collapse packet *for s is a finite packet containing an RH zero code for s, the mirror row $J(s)$, the mirror zero code for $J(s)$, and an* $\mathsf{hsame}(s, J(s))$ *row. By the fixed-point calculation in Definition 9241.12, this is equivalently a skew-zero packet for s.*

Corollary 9241.16 (GroundCompiler-normal form of RH)**.** *At the GroundCompiler-facing layer, the RH obligation is the production of a recognized zero-orbit collapse event flow:*

$$\mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to \mathsf{ZeroOrbitCollapseEventFlow}_{\mathsf{RH}}(s) \to \mathsf{SkewZeroEventFlow}_{\mathsf{RH}}(s).$$

Proof. The first arrow supplies the missing orbit-collapse witness $\mathsf{hsame}(s, J(s))$ from a recognized zero event flow. The second arrow is the algebraic readback of Definition 9241.12: the fixed-point condition for J is exactly the critical-line condition. Thus the composite is the zero–skew transformer of Definition 9241.3, and conversely any zero–skew transformer supplies the collapse row by reading $\Re(s) = 1/2$ as $s = J(s)$. □

9241.5 Relation to finite analysis routes

The event-flow formulation separates three kinds of RH-facing work. The channel layer is only the lossless display layer: by Definition 9208.17 and Definition 9208.19, it preserves event boundaries and recovers decoded event flows from legal channel strings. By Definition 9241.6 and Definition 9241.9, neither the terminator 11 nor the displayed glyph 0 carries a zero reading. The recognizer layer reads particular event flows as zeta-zero, critical-strip, mirror, or skew-zero packets, but a recognized input packet still contributes only the premise of Definition 9239.8. The proof layer is the additional compiler or proof flow that produces the missing collapse or skew-zero row.

This separation explains the status of the existing RH witness catalogue. The critical-line witness packet of chapter 548 is a possible output surface for the skew-zero row. The critical-strip barrier packet of chapter 705 is a possible route to that row by showing that every region bounded away from the critical line is bounded away from zero. The Riemann zero-count ledger of chapter 573 supplies finite height-window accounting, but it becomes a route to RH only when combined with exhaustion, multiplicity, boundary, and transport ledgers that cover all zero records in the requested window. In each case the deciding question is not whether a readable event flow exists, but whether the accepted flow exports the skew-zero or orbit-collapse row required by Definition 9241.3.

9241.6 Analytic continuation as projection recovery

The same boundary discipline explains why analytic continuation is the appropriate semantic model for the zeta side of the RH target. A convergent series expression is not the whole analytic carrier. It is a local projection of that carrier on a domain where the expression is readable. For zeta, the Dirichlet series is the projection of ZETA on the window $\Re(s) > 1$, the eta expression is another compatible projection on a larger window, and the functional-equation expression is a reflected projection with its own pole, Gamma, branch, and transport ledgers.

Principle 9241.17 (Projection recovery)**.** *Analytic continuation is not an instruction to assign values to a divergent local expression. It is the recovery of one invariant analytic carrier from local projection slices whose domains, overlaps, exceptional loci, and transport ledgers are displayed.*

Remark 9241.18 (EventFlow analogue)**.** *The GroundCompiler channel layer is the finite combinatorial analogue of the same rule. A bare glyph stream is not an event flow because event boundaries may have been erased; by Definition 9207.7, distinct event flows can have the same flattened display. The legal channel encoding restores the boundary ledger through* 11 *terminators, while recognizer and certificate gates restore semantic roles. In the analytic layer, a bare formula without its domain, overlap, branch, pole, and transport ledger likewise cannot be promoted to the global zeta carrier. In both settings, the object is the stable carrier recovered from compatible local records, not any single display slice.*

Concretized in §550: Critical-strip zeta-zero witness packets provide the recognized zero-event input surface for the RH zero–skew EventFlow target.

Concretized in §548: Critical-line witness packets provide the skew-zero output surface for any accepted zero–skew transformer.

Concretized in §581: The zeta functional-equation ledger supplies mirror-zero transport rows while leaving zero-orbit collapse as the RH-specific missing certificate.

Chapter 9242

EventFlow Certificate Flow for Primes, Zeta, and RH

chapter 9241 places the Riemann Hypothesis at the compiler-facing layer as a transformer from recognized zero-event packets to recognized critical-line packets. The present chapter records the certificate-flow substrate below that transformer. It shows how prime recognition, zeta recovery, and the RH fixed-point target are separated into finite EventFlow rows. This separation does not prove RH. It turns the conjecture into a fixed-output certificate target whose missing row is visible.

9242.1 Boundary before semantics

Definition 9242.1 (Channel fixed boundary)**.** *For an event flow $S = (w_0, \ldots, w_{n-1})$, the channel layer records a source event-flow row E, a legal channel-code row C, a decoder row D, and a round-trip witness row R:*

$$E \to C \to D \to R.$$

The round-trip row asserts only

$$\mathsf{Decode}(\mathsf{Compile}(S))\mathsf{hsame}S.$$

It recovers the event boundaries and the event sequence. It does not recover primehood, zeta values, zerohood, or proof rows.

Lean packages this finite substrate as
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.EventflowStep`
,
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.FlowLinked`
, and
Lean4 (definition) `BEDC.Derived.RHRoute.EventflowCertificate.CertificateFlow`
; the certificate transport is validated by
Lean4 ✓ `BEDC.Derived.RHRoute.EventflowCertificate.flow certificate sound`
.

Proposition 9242.2 (Channel reconstruction is a boundary fixed point)**.** *Every accepted legal-channel read makes the source event flow a fixed point of the compile-decode composite:*

$$(\mathsf{Decode} \circ \mathsf{Compile})(S)\mathsf{hsame}S.$$

Proof. The accepted read must pass through the displayed rows E, C, D, R. The row C stores the legal channel code of the source flow named by E, the row D stores the decoder read, and R identifies the decoded output with the source event flow up to hsame. Hence the compile-decode composite returns the same event sequence and no semantic classifier has yet been applied. □

Corollary 9242.3 (Terminator rows are not zeta-zero rows)**.** *If the channel representation uses* 11 *as event terminator, each occurrence of* 11 *records only the end of one raw event. A zeta-zero reading can appear only after a generated recognizer accepts a whole decoded packet as a* $\mathsf{ZeroCode}_{\mathsf{RH}}$ *or* $\mathsf{ZeroEventFlow}_{\mathsf{RH}}$ *surface.*

Proof. This is the channel-level specialization of Definition 9241.6. The terminator belongs to the legal-code row, while a zero row belongs to a later recognizer and certificate gate. □

9242.2 Prime flow as factorization fixed point

Definition 9242.4 (Arithmetic event packets)**.** *A natural-number event packet* $\mathsf{NatFlow}(n)$ *is a recognized finite EventFlow packet exposing*

$$n, \quad \mathsf{Nat}^{\uparrow}, \quad \mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}, \quad \mathsf{ContRoute}, \quad \mathsf{Pkg}.$$

A split event packet $\mathsf{SplitFlow}(n; a, b)$ *exposes natural-number rows for* a, b, n *together with a multiplication row*

$$a \cdot b\mathsf{hsame}n.$$

The split is non-trivial when both displayed factors are separated from the unit row.

Definition 9242.5 (Prime event flow)**.** *A prime event flow* $\mathsf{PrimeFlow}(p)$ *is a natural-number event packet for* p *together with a strict row* $p > 1$ *and a refusal row excluding every non-trivial split packet* $\mathsf{SplitFlow}(p; a, b)$.

Definition 9242.6 (Factorization normalizer)**.** *The factorization normalizer* $\mathsf{FactNorm}$ *sends a recognized natural-number packet to the finite multiset of prime packets generated by its displayed factorization route. A singleton output is written* $[p]$.

Proposition 9242.7 (Prime flow is the atomic factorization fixed point)**.** *For a recognized natural-number packet* p *with* $p > 1$,

$$\mathsf{PrimeFlow}(p) \quad \Longleftrightarrow \quad \mathsf{FactNorm}(p)\mathsf{hsame}[p].$$

Proof. If $\mathsf{PrimeFlow}(p)$ holds, the refusal row excludes all non-trivial split packets. The factorization normalizer therefore cannot expose a proper product decomposition and returns the singleton packet $[p]$. Conversely, if $\mathsf{FactNorm}(p)\mathsf{hsame}[p]$ while a non-trivial split $p\mathsf{hsame}a \cdot b$ existed with both factors separated from the unit row, the normalizer would have to continue through the factorization routes of a and b, yielding a non-singleton prime packet list. This contradicts the displayed singleton row. Hence no non-trivial split exists, and the packet is prime. □

Remark 9242.8 (No arithmetic self-assertion)**.** *The fixed-point reading of Definition 9242.7 is not a bare self-assertion that* p *is prime. It is a factorization readback: the object is atomic because the recognized multiplication-split flow returns the same prime packet and no proper split packet.*

9242.3 Zeta as socket recovery flow

Definition 9242.9 (Prime-local and Dirichlet event projections)**.** *For a recognized prime packet* $\mathsf{PrimeFlow}(p)$ *and a complex history* s*, the prime-local channel event is*

$$\mathsf{PrimeChan}_p(s) = p^{-s}.$$

On the right half-plane $\Re(s) > 1$*, the Euler event projection and Dirichlet event projection are*

$$\mathsf{EulerFlow}(s) = \prod_{p \in \mathsf{PrimeFlow}} (1 - p^{-s})^{-1}, \qquad \mathsf{DirichletFlow}(s) = \sum_{n \geq 1} n^{-s}.$$

Definition 9242.10 (Zeta event-flow socket)**.** *A zeta event-flow socket is a finite packet*

$$Z = (B, E, A, P, F, T, G, H, C, N)$$

whose rows are read as follows:

$$B : \mathsf{ZetaBasic}^{\uparrow} \textit{ source on } \Re(s) > 1, \quad E : \eta\textit{-half-plane handoff},$$
$$A : \mathsf{AnaCont}^{\uparrow} \textit{ handoff}, \quad P : \textit{pole ledger at } s = 1,$$
$$F : \textit{functional-equation factor ledger}, \quad T : \textit{trivial-zero ledger},$$
$$G : \mathsf{Gamma}^{\uparrow} \textit{ boundary row}, \quad H : \mathsf{hsame} \textit{ transports},$$
$$C : \textit{finite } \mathsf{Cont} \textit{ routes}, \quad N : \mathsf{NameCert} \textit{ row}.$$

This is the EventFlow reading of the $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *packet.*

Proposition 9242.11 (Zeta recovery from prime-local flow and socket rows)**.** *On* $\Re(s) > 1$*,*

$$\mathsf{DirichletFlow}(s)\mathsf{hsame}\mathsf{EulerFlow}(s).$$

Together with the rows of Definition 9242.10, this right-half-plane agreement recovers the global meromorphic carrier ZETA *as a compatible projection packet rather than as a single series expression.*

Proof. Absolute convergence on $\Re(s) > 1$ permits grouping by the displayed factorization routes. Each natural-number index decomposes uniquely as a finite product of prime packets:

$$n = \prod_p p^{k_p}.$$

Therefore

$$n^{-s} = \prod_p p^{-k_p s},$$

and the sum over positive indices becomes the product of local geometric series

$$\sum_{n \geq 1} n^{-s} = \prod_p \sum_{k \geq 0} p^{-ks} = \prod_p (1 - p^{-s})^{-1}.$$

This supplies the prime-local projection row. The zeta socket then adds the eta handoff, analytic-continuation handoff, pole ledger, functional-equation ledger, Gamma-boundary row, transports, and finite continuation routes. These rows form the compatible recovery surface for ZETA. □

Corollary 9242.12 (Zeta is not exhausted by its series projection)**.** *The event-flow reading of* ZETA *is not the bare row* $\sum_{n\geq 1} n^{-s}$*. It is the recovered carrier obtained from the Dirichlet row, Euler prime row, eta handoff, analytic-continuation handoff, pole row, functional-equation row, Gamma boundary, transport rows, and local NameCert row.*

Proof. The Dirichlet row is only one projection on $\Re(s) > 1$. The remaining rows of Definition 9242.10 are necessary to read the same carrier outside that window and to preserve its pole, mirror, trivial-zero, and transport ledgers. □

9242.4 RH as fixed-output target

Definition 9242.13 (Prime-unitary balance event flow)**.** *For a complex history* s*, a prime-unitary balance event flow is a recognized packet exposing, for every prime packet* $\mathsf{PrimeFlow}(p)$*, the centered channel row*

$$U_p(s) = p^{1/2}p^{-s} = p^{1/2-s}$$

and the unit-amplitude witness

$$|U_p(s)|\,\mathsf{hsame1}.$$

Definition 9242.14 (Prime-balanced RH compiler target)**.** *The prime-balanced RH compiler target is a certified compiler or proof flow* $C^{\mathrm{prime}}_{\mathsf{RH}}$ *with type*

$$C^{\mathrm{prime}}_{\mathsf{RH}} : \mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to \mathsf{PrimeUnitaryBalanceFlow}_{\mathsf{RH}}(s).$$

Expanded through the recognized rows, it is the target

$$(\mathsf{ZETA}(s)\mathsf{hsame0}_{\mathbb{C}} \wedge \mathsf{InCritStrip}(s)) \to \forall p \in \mathsf{PrimeFlow},\ \left|p^{1/2-s}\right|\,\mathsf{hsame1}.$$

Proposition 9242.15 (Prime-balanced compiler target is an RH witness)**.** *An accepted prime-balanced RH compiler target supplies the same mathematical content as the zero–skew transformer of Definition 9241.3.*

Proof. Definition 9243.7 identifies the prime-unitary output packet with the critical-line row $\Re(s)\mathsf{hsame1}/2$. Thus a compiler flow from a recognized zero event packet to a recognized prime-unitary balance packet yields the skew-zero output row required by Definition 9241.3. Conversely, a zero–skew transformer gives the prime-unitary output by reading $\Re(s)\mathsf{hsame1}/2$ through the normalized prime-channel amplitude formula. □

Definition 9242.16 (Zero-orbit collapse event flow)**.** *Let* $J(s) = 1 - \overline{s}$*. A zero-orbit collapse event flow for* s *is a finite packet containing*

$$\mathsf{ZeroEventFlow}_{\mathsf{RH}}(s), \quad \mathsf{ZeroEventFlow}_{\mathsf{RH}}(J(s)), \quad \mathsf{hsame}(s, J(s)),$$

together with the displayed transport, route, provenance, and local naming rows.

Proposition 9242.17 (RH is zero-orbit collapse)**.** *The BEDC statement* RH *is equivalent to the existence of a total target*

$$\mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to \mathsf{ZeroOrbitCollapseFlow}_{\mathsf{RH}}(s).$$

Proof. If RH holds, a recognized non-trivial zero packet has $\Re(s)\mathsf{hsame1}/2$. The mirror calculation of Definition 9241.12 then gives $\mathsf{hsame}(s, J(s))$, and the functional-equation mirror rows supply the companion zero packet for $J(s)$ when the zeta socket rows are displayed. This is the zero-orbit collapse packet. Conversely, if every zero event flow has an accepted collapse packet, its $\mathsf{hsame}(s, J(s))$ row implies $\Re(s)\mathsf{hsame1}/2$ by the same mirror calculation, hence supplies $\mathsf{OnCritLine}(s)$ for every critical-strip zero packet. □

9242.5 Fixed-point stratification

Principle 9242.18 (Certified fixed points only)**.** *The EventFlow reading admits fixed points only as displayed certificate rows: channel round-trip rows, factorization singleton rows, projection-recovery rows, mirror-collapse rows, and fixed-output witness rows. It does not admit bare self-reference, host-level fixed-point principles, or self-application oracles.*

Theorem 9242.19 (Three fixed-point layers)**.** *The EventFlow reconstruction of primes, zeta, and RH separates three fixed-point layers:*

$$\mathsf{PrimeFlow}(p) \iff \mathsf{FactNorm}(p)\mathsf{hsame}[p],$$

$$P_i(\mathrm{Rec}((D_i, f_i)_i))\mathsf{hsame} f_i \quad \textit{for each displayed zeta projection row,}$$

and

$$\mathsf{RH} \iff \forall s,\ \mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to \mathsf{ZeroOrbitCollapseFlow}_{\mathsf{RH}}(s).$$

Proof. The first equivalence is Definition 9242.7. The second is the projection-recovery discipline of Definition 9242.11: recovering the carrier from compatible local rows and projecting back to any displayed window returns that window's row. The third is Definition 9242.17. Each layer is a certified readback or fixed-output row, not a self-application principle. □

Remark 9242.20 (Generator fixed-point boundary)**.** *The same discipline is enforced by the* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet. A fixed-output witness is accepted only through the displayed rows* $G, L, Q, W, O, H, C, P, N$*; any diagonal or self-application attempt outside those rows remains a* GAP *boundary. The RH target therefore asks for a finite output witness, not for a host theorem that a proof generator must have a fixed point.*

Concretized in §829: GroundCompiler event-flow audit packets separate source events, channel rows, generated recognizers, certificate gates, transport, provenance, and local naming before any arithmetic or zeta meaning is read.

Concretized in §1094: Flow-level round-trip packets supply the compile-decode boundary fixed point for legal channel reads.

Concretized in §64: Prime packets supply the multiplication-split refusal rows used by the factorization fixed-point reading.

Concretized in §570: Zeta-continuation socket packets supply the finite zeta recovery rows $B, E, A, P, F, T, G, H, C,$

Concretized in §556: Generator fixed-point packets supply the fixed-output witness boundary and reject host-level self-application routes.

Concretized in §550: Critical-strip zeta-zero witness packets supply the recognized zero-event input surface consumed by the RH compiler targets.

Concretized in §548: Critical-line witness packets supply the recognized output surface equivalent to prime-unitary balance and zero-orbit collapse.

Chapter 9243

Prime-Local Balance and the RH Critical Surface

chapter 9239 states RH as the constructive implication from a critical-strip zeta-zero witness to an $\mathsf{OnCritLine}$ witness, and chapter 9241 rewrites the same obligation as a zero–skew transformer. The present chapter isolates a compatible prime-local reading of the critical line. It does not prove RH. It identifies $\Re(s) = 1/2$ as the common unit-modulus surface of the normalized prime channels that appear in the Euler-product projection of ζ.

9243.1 Zeta as a recovered carrier

Definition 9243.1 (Zeta prime-projection tower)**.** *Let $D_{>1}$ denote the half-plane $\Re(s) > 1$. The zeta prime-projection tower is the displayed packet*

$$\Pi_\zeta^{\mathrm{prime}} = \big(\zeta_D,\ (\mathsf{PrimeChan}_p)_{p\in\mathsf{Prime}^\uparrow},\ \mathsf{EulerProj},\ \mathsf{EtaRow},\ \mathsf{AnaCont}^\uparrow,\ \Gamma_\infty,\ \mathsf{FunctionalEquation},\ \mathsf{Pole}_{s=1}\big),$$

where

$$\zeta_D(s) = \sum_{n\geq 1} n^{-s}, \qquad \mathsf{PrimeChan}_p(s) = p^{-s}, \qquad \mathsf{EulerProj}(s) = \prod_{p\in\mathsf{Prime}^\uparrow} (1-p^{-s})^{-1}$$

on $D_{>1}$. The row Γ_∞ denotes the archimedean completion channel, and $\mathsf{Pole}_{s=1}$ denotes the simple-pole ledger consumed by the continuation packet.

Definition 9243.2 (Zeta recovery carrier)**.** *The BEDC-facing ZETA carrier is the analytic object recovered from the compatible projection rows*

$$\mathsf{DirichletSeries}_\zeta(s) = \sum_{n\geq 1} n^{-s} \quad \textit{on} \quad \Re(s) > 1,$$

$$\mathsf{Euler}_\zeta(s) = \prod_{p\in\mathsf{Prime}^\uparrow} (1-p^{-s})^{-1} \quad \textit{on} \quad \Re(s) > 1,$$

$$\mathsf{ZetaCont}^\uparrow, \quad \mathsf{Gamma}^\uparrow\textit{-completion rows}, \quad \textit{functional-equation mirror rows},$$
$$\textit{pole ledgers}, \quad \textit{zero ledgers}, \quad \textit{branch and transport ledgers.}$$

The Dirichlet series is therefore a readable projection of the carrier on its half-plane of absolute convergence, not the carrier by itself. Equivalently,

$$\mathsf{ZETA} = \mathrm{Rec}_{\mathsf{AnaCont}}(\Pi_\zeta^{\mathrm{prime}}),$$

where the recovery symbol records the compatible analytic-continuation, completion, pole, and transport rows rather than a primitive ambient function.

Principle 9243.3 (No projection exhausts zeta)**.** *No single projection row exhausts* ZETA*. A series expression, an Euler product, a completed functional-equation expression, and a zero ledger are compatible readings of the same recovered carrier only when their domains, overlaps, exceptional loci, and transport rows are displayed in the same certificate surface.*

Proposition 9243.4 (Euler product as prime-local gluing)**.** *On the half-plane $\Re(s) > 1$, the Dirichlet projection and the Euler prime-local projection agree:*

$$\zeta_D(s) = \mathsf{EulerProj}(s) = \prod_{p \in \mathsf{Prime}^{\uparrow}} (1 - p^{-s})^{-1}.$$

Thus the right-half-plane zeta projection is the multiplicative gluing of the prime-local channels $\mathsf{PrimeChan}_p(s) = p^{-s}$*.*

Proof. The absolute convergence of $\sum_{n \geq 1} n^{-s}$ on $\Re(s) > 1$ permits term grouping. The finite prime-factorization row supplied by $\mathsf{Prime}^{\uparrow}$ writes each positive index uniquely as a finite product

$$n = \prod_p p^{k_p}.$$

Hence

$$n^{-s} = \prod_p p^{-k_p s}.$$

Summing over positive indices is therefore equivalent, under the displayed absolute-convergence modulus, to multiplying the local geometric sums

$$\sum_{k \geq 0} p^{-ks} = (1 - p^{-s})^{-1}.$$

This is precisely the Euler-product theorem consumed by Definition 71.16. □

This principle is the analytic counterpart of Definition 9241.17. The Euler product displays the finite-prime channels. The $\mathsf{Gamma}^{\uparrow}$ row displays the archimedean channel. The functional-equation ledger displays the mirror symmetry. The critical line appears only after these rows have been assembled into the completed self-dual carrier.

9243.2 Prime-local channel normalization

Definition 9243.5 (Prime-local channels)**.** *For a displayed prime row $p \in \mathsf{Prime}^{\uparrow}$ and a complex history $s = \sigma + it$, the prime-local channel and its critical normalization are written*

$$\mathsf{PrimeChan}_p(s) = p^{-s}, \qquad \mathsf{UnitPrimeChan}_p(s) = p^{1/2} p^{-s}.$$

The centered amplitude row is

$$\mathsf{Amp}_p(s) = |\mathsf{UnitPrimeChan}_p(s)|\,.$$

Equivalently,

$$\mathsf{UnitPrimeChan}_p(s) = p^{1/2-\sigma} e^{-it \log p}.$$

Definition 9243.6 (Prime-unitary balance)**.** *The predicate* $\mathsf{PrimeUnitaryBalance}_\zeta(s)$ *holds when every displayed prime row has unit normalized amplitude:*

$$\mathsf{PrimeUnitaryBalance}_\zeta(s) :\Longleftrightarrow \forall p \in \mathsf{Prime}^\uparrow, \left|p^{1/2}p^{-s}\right| \mathsf{hsame1}.$$

This is a predicate on the prime-channel projection of the zeta carrier. It does not assert that a zero has been found, counted, or located.

Theorem 9243.7 (Prime-unitary balance is the critical line)**.** *For a complex history* $s = \sigma + it$ *and any displayed prime row* $p > 1$*,*

$$\left|p^{1/2}p^{-s}\right| = 1 \quad \Longleftrightarrow \quad \sigma = 1/2.$$

Consequently, on the $\mathsf{Prime}^\uparrow$ *packet,* $\mathsf{PrimeUnitaryBalance}_\zeta(s)$ *is equivalent to* $\mathsf{OnCritLine}(s)$*.*

Lean4 ✓ `BEDC.Derived.RHRoute.UnitaryBalance.singlePrimeUnitaryBalanceSurface norm one`

Proof. By the channel definition,

$$p^{1/2}p^{-s} = p^{1/2}p^{-(\sigma+it)} = p^{1/2-\sigma}p^{-it} = p^{1/2-\sigma}e^{-it\log p}.$$

The factor $e^{-it\log p}$ has modulus one, so

$$\left|p^{1/2}p^{-s}\right| = p^{1/2-\sigma}.$$

If $\sigma = 1/2$, this modulus is $p^0 = 1$ for every displayed prime. Conversely, if the modulus is one for a displayed prime $p > 1$, then $p^{1/2-\sigma} = 1$. Positivity and separation of the displayed prime row force $1/2 - \sigma = 0$, hence $\sigma = 1/2$. The BEDC predicate $\mathsf{OnCritLine}(s)$ is precisely the constructive read of $\Re(s)\mathsf{hsame}1/2$ from Definition 9239.7. □

Remark 9243.8 (Scope of the unit-modulus readout)**.** *The unit-modulus calculation*

$$|a^{1/2-s}| = 1 \quad \Longleftrightarrow \quad \Re(s) = 1/2$$

uses only a displayed real base $a > 1$*; primality is not needed for this geometric readout. Primes enter because the zeta carrier supplies prime-indexed Euler-product channels. The substantive prime-to-zero content is therefore not the unit-modulus equivalence itself but the boundary route that connects prime-channel skew to zero-ledger exclusion, represented by the prime-skew defect interface of Definition 9241.3.*

Corollary 9243.9 (Prime-channel amplitude tilt)**.** *Let*

$$\delta(s) = \Re(s) - 1/2.$$

Then for every displayed prime row p*,*

$$|\mathsf{UnitPrimeChan}_p(s)| = p^{-\delta(s)}.$$

Thus an off-line skew is read in the Euler-product projection as a coherent amplitude tilt across all prime-local channels.

Proof. Substitute $\sigma = \Re(s)$ into $\left|p^{1/2}p^{-s}\right| = p^{1/2-\sigma}$ and rewrite $1/2 - \sigma$ as $-\delta(s)$. □

Corollary 9243.10 (Prime-channel tilt detects off-line reads)**.** *For any displayed prime row $p > 1$,*

$$|\mathsf{UnitPrimeChan}_p(s)| \neq 1 \quad \Longleftrightarrow \quad \Re(s) \neq 1/2.$$

Equivalently, a critical-strip zero off the critical line is exactly a zero record carrying prime-local amplitude tilt.

Proof. This is the contrapositive reading of Definition 9243.7 at the displayed prime row $p > 1$, together with the amplitude formula of Definition 9243.9. □

9243.3 Prime-balanced form of RH

Definition 9243.11 (Prime-balanced RH)**.** *The prime-balanced form of the Riemann Hypothesis is the proposition*

$$\mathsf{PrimeBalancedRH}_\zeta :\Longleftrightarrow \forall s, \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{PrimeUnitaryBalance}_\zeta(s).$$

Equivalently, in the event-flow notation of Definition 9241.1, it asks for a total transformer

$$\mathsf{ZeroCode}_{\mathsf{RH}}(s) \to \mathsf{PrimeUnitaryBalance}_\zeta(s).$$

Proposition 9243.12 (Prime-balanced RH is equivalent to BEDC-RH)**.** *The proposition* $\mathsf{PrimeBalancedRH}_\zeta$ *is equivalent to the BEDC statement* RH *of Definition 9239.7. This is a scoped reparameterization of the same zero-ledger implication through the fixed-line prime-unitary readout, not an independent prime-side proof.*

Proof. The premise $\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s)$ is exactly the premise of Definition 9239.7 and the premise packet of Definition 9241.1. By Definition 9243.7, the conclusion $\mathsf{PrimeUnitaryBalance}_\zeta(s)$ is equivalent to $\mathsf{OnCritLine}(s)$. Substitution of equivalent conclusions in the same dependent implication gives the equivalence with RH. Under Definition 9241.3, the same conclusion is the skew-zero output row written through prime-channel amplitudes instead of through the real offset $\delta(s)$ alone. □

Corollary 9243.13 (No prime-amplitude tilt form)**.** *The BEDC statement* RH *is equivalent to the assertion that no displayed critical-strip zero packet carries a displayed prime-channel amplitude tilt:*

$$\neg\exists s,\ \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \wedge \exists p \in \mathsf{Prime}^{\uparrow},\ \left|p^{1/2}p^{-s}\right| \neq 1.$$

Proof. By Definition 9243.12, RH is equivalent to the implication from a critical-strip zero packet to $\mathsf{PrimeUnitaryBalance}_\zeta(s)$. Expanding Definition 9243.6 gives unit amplitude for every displayed prime row. A counter-witness to that universal row is exactly a displayed prime-channel amplitude tilt on a displayed critical-strip zero packet. □

Definition 9243.14 (Prime-channel RH witness)**.** *The prime-channel RH witness type is*

$$\mathsf{PrimeRHWitness}_\zeta = \prod_s (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)) \to \prod_{p \in \mathsf{Prime}^{\uparrow}} \mathsf{hsame}\left(\left|p^{1/2}p^{-s}\right|, 1\right).$$

The ordinary RH witness type is

$$\mathsf{RHWitness}_\zeta = \prod_s (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)) \to \mathsf{OnCritLine}(s).$$

Proposition 9243.15 (Prime and ordinary RH witnesses are equivalent)**.** *There is a canonical equivalence*

$$\mathsf{PrimeRHWitness}_\zeta \simeq \mathsf{RHWitness}_\zeta.$$

Proof. For each s, Definition 9243.7 identifies the prime-unitary output row with $\mathsf{OnCritLine}(s)$. Applying this pointwise to the common input packet $\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)$ converts a prime-channel witness into an ordinary witness and conversely. The two conversions are inverse because both read the same equality $\Re(s)\mathsf{hsame}1/2$ through different projection rows. □

Remark 9243.16 (Prime balance is not an Euler-factor proof)**.** *The implication in Definition 9243.11 is not a claim that a single Euler factor proves* RH*. A factor* $(1-p^{-s})^{-1}$ *is only one local prime projection. It does not carry the non-trivial zero locus of* ζ*, and it does not supply analytic continuation, a pole ledger, a* $\mathsf{Gamma}^{\uparrow}$ *boundary, or the functional-equation mirror row. The critical axis is a feature of the completed recovered carrier, while a single normalized prime channel only exhibits the amplitude test that any point on that axis must pass.*

9243.4 Archimedean completion and the central surface

Definition 9243.17 (Completed zeta surface)**.** *The completed zeta surface is the recovery packet whose displayed analytic core has the classical shape*

$$\xi(s) = \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s)$$

together with the functional-equation row

$$\xi(s) = \xi(1-s).$$

The finite-prime rows enter through the Euler-product projection of ζ*. The archimedean row enters through the* $\pi^{-s/2}\Gamma(s/2)$ *factor. The factor* $s(s-1)$ *records the pole-cancellation ledger at* $s=1$ *and* $s=0$*.*

Proposition 9243.18 (The critical surface is self-dual)**.** *Inside the completed zeta surface, the line* $\Re(s) = 1/2$ *is the fixed surface of the mirror* $s \mapsto 1-s$ *on the real component. The prime-unitary balance predicate of Definition 9243.6 is the finite-prime amplitude read of the same central normalization.*

Proof. The functional-equation row identifies the completed carrier at s with the completed carrier at $1-s$. The real components of s and $1-s$ agree exactly when $\Re(s) = 1/2$. By Definition 9243.7, this equality is also equivalent to unit normalized amplitude for every displayed prime-local channel. Therefore the same central surface has two compatible reads: a mirror fixed-surface read through the completed carrier and a unit-amplitude read through the Euler-product projection. □

Proposition 9243.19 (Prime mirror inversion)**.** *Let* $J(s) = 1-\overline{s}$ *be the mirror map of Definition 9241.12, and write*

$$U_p(s) = p^{1/2-s}.$$

For every displayed prime row p*,*

$$U_p(J(s)) = \frac{1}{\overline{U_p(s)}}.$$

Consequently,

$$|U_p(J(s))| = |U_p(s)|^{-1}.$$

Proof. Write $s = \sigma + it$. Then

$$U_p(s) = p^{1/2-\sigma} e^{-it\log p}, \qquad \overline{U_p(s)} = p^{1/2-\sigma} e^{it\log p},$$

so

$$\frac{1}{\overline{U_p(s)}} = p^{\sigma-1/2} e^{-it\log p}.$$

Since

$$J(s) = 1 - \overline{s} = 1 - \sigma + it,$$

one also has

$$U_p(J(s)) = p^{1/2-(1-\sigma+it)} = p^{\sigma-1/2} e^{-it\log p}.$$

The displayed equality follows, and taking moduli gives the amplitude inversion row. □

Corollary 9243.20 (Fixed mirror amplitude surface)**.** *The critical line is the unique surface on which every displayed normalized prime channel is fixed in amplitude by the mirror J:*

$$s = J(s) \quad \Longleftrightarrow \quad \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(s)| = 1.$$

Proof. If $s = J(s)$, Definition 9243.19 gives $|U_p(s)| = |U_p(s)|^{-1}$ for every displayed prime row. Since the channel amplitude is positive, $|U_p(s)| = 1$. Conversely, if every normalized prime-channel amplitude is one, then Definition 9243.7 gives $\Re(s) = 1/2$, which is equivalent to $s = J(s)$ by Definition 9241.12. □

Remark 9243.21 (Role of the archimedean row)**.** *The finite-prime channels explain the local amplitude test, but they do not by themselves determine the completed self-dual carrier. The $\pi^{-s/2}\Gamma(s/2)$ row supplies the archimedean channel that normalizes the functional equation into the $s \leftrightarrow 1-s$ mirror form. Without that row, the expression still has prime-local projections, but the central self-dual surface is not a closed recovery certificate.*

9243.5 Projection tower reading

Principle 9243.22 (Projection tower lifting)**.** *After ζ has been recovered as a meromorphic carrier on $\mathsf{Complex}^{\uparrow} \setminus \{1\}$, further structural work is not a larger complex-domain continuation. It is a lifting through projection towers:*

$$\zeta(s) \to \xi(s) \to \textit{prime-local channel packets} \to \textit{explicit-formula ledgers} \to \textit{adelic zeta-integral packets} \to \mathsf{LFunction}^{\uparrow}$$

Each arrow adds rows that explain how a displayed projection is recovered from or compared with a richer carrier. It does not license an unstated branch, zero oracle, or ambient analytic primitive.

Remark 9243.23 (Schematic L-function generalization)**.** *The same surface reading has a schematic counterpart for any displayed $\mathsf{LFunction}^{\uparrow}$ carrier whose packet contains a Dirichlet-series row, an Euler product row, prime-local factors, archimedean completion rows, analytic continuation, a functional equation, and pole or branch ledgers. Its central line is the fixed surface of its completed functional-equation mirror, and the analogue of prime-unitary balance is the assertion that all normalized finite-prime channels have unit amplitude on that surface. This is a schema-level reading of the generalized RH shape, not a closed theorem about all L-carriers.*

Proposition 9243.24 (Norm-local central unit surface)**.** *Let v be a displayed finite local channel of an* $\mathsf{LFunction}^{\uparrow}$ *carrier, and let $N(v) > 1$ be its local norm. For*

$$U_v(s) = N(v)^{1/2-s},$$

one has

$$|U_v(s)| = 1 \quad \Longleftrightarrow \quad \Re(s) = 1/2.$$

Proof. Writing $s = \sigma + it$ gives

$$U_v(s) = N(v)^{1/2-\sigma} e^{-it \log N(v)}.$$

The phase factor has modulus one, hence

$$|U_v(s)| = N(v)^{1/2-\sigma}.$$

Since $N(v) > 1$, this modulus is one exactly when $1/2 - \sigma = 0$. □

Theorem 9243.25 (Prime-local projection normal form)**.** *The RH projection reading is the conjunction of the following equivalent normal forms:*

$\mathsf{ZETA} = \mathrm{Rec}_{\mathsf{AnaCont}}(\textit{Dirichlet projection}, \textit{prime-local projection}, \textit{archimedean completion}, \textit{functional equation}, \textit{pole l}$

$$\mathsf{PrimeUnitaryBalance}_{\zeta}(s) \Longleftrightarrow \Re(s) = 1/2 \Longleftrightarrow s = J(s),$$

and

$$\mathsf{RH} \Longleftrightarrow \forall s,\ \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{PrimeUnitaryBalance}_{\zeta}(s).$$

Proof. The recovery line is Definition 9243.2 read through the prime-projection tower of Definition 9243.1. The second line combines Definition 9243.7 with the fixed-point characterization of Definition 9241.12. The third line is exactly Definition 9243.12. □

Concretized in §64: The prime-local channel rows are indexed by the finite $\mathsf{Prime}^{\uparrow}$ NameCert surface.

Concretized in §71: The Euler-product projection supplying p^{-s} channels is read through the $\mathsf{ZetaBasic}^{\uparrow}$ half-plane certificate.

Concretized in §580: The zeta-continuation witness packet supplies the continuation, pole, trivial-zero, and Gamma-boundary rows consumed before a zero-facing carrier can be read.

Concretized in §581: The zeta functional-equation ledger supplies the completed-carrier mirror row whose fixed surface is the critical line.

Concretized in §548: Critical-line witness packets provide the finite $\mathsf{OnCritLine}$ output surface equivalent to prime-unitary balance.

Concretized in §103: The $\mathsf{LFunction}^{\uparrow}$ interface provides the schematic family carrier for the corresponding central-line balance reading.

Chapter 9244

Finite Prime Windows and Local–Global RH Gluing

chapter 9243 identifies the critical line with unit amplitude on the normalized prime-local channels. This chapter records the proof-program boundary suggested by that reading. Finite prime windows give auditable local tests, but a finite window is not the recovered zeta carrier. A local proof route reaches RH only after the finite-prime rows, the archimedean completion row, the functional equation, and the zero ledger are glued into one carrier-level exclusion.

9244.1 Finite Prime Windows

Definition 9244.1 (Finite prime window)**.** *A finite prime window is a finite displayed set*

$$S \subseteq \mathsf{Prime}^{\uparrow}.$$

For s in the zeta carrier, its window amplitude readout is

$$\mathsf{Amp}_S(s) := \big(\mathsf{Amp}_p(s)\big)_{p \in S},$$

where Amp_p is the centered amplitude row of Definition 9243.5. The unit-window predicate is

$$\mathsf{UnitWindow}_S(s) := \prod_{p \in S} \big(\mathsf{Amp}_p(s) = 1\big).$$

Theorem 9244.2 (Nonempty finite windows detect the half surface)**.** *If S is nonempty, then*

$$\mathsf{UnitWindow}_S(s) \iff \mathsf{OnCritLine}(s).$$

Proof. If $\mathsf{UnitWindow}_S(s)$ holds, choose a displayed row $p \in S$. The window predicate gives $\mathsf{Amp}_p(s) = 1$, and Definition 9243.7 gives $\mathsf{OnCritLine}(s)$. Conversely, if $\mathsf{OnCritLine}(s)$ holds, then the same theorem gives $\mathsf{Amp}_p(s) = 1$ for every displayed prime row, hence for every $p \in S$. □

Definition 9244.3 (Finite-window RH target)**.** *For a finite prime window S, define*

$$\mathsf{RH}_S := \prod_s \big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{UnitWindow}_S(s)\big).$$

Theorem 9244.4 (Nonempty finite-window RH has full RH strength)**.** *If S is nonempty, then*

$$\mathsf{RH}_S \iff \mathsf{RH}.$$

Proof. Substitute the equivalence of Definition 9244.2 into the output predicate of Definition 9244.3. The result is exactly the BEDC statement of Definition 9239.7. □

Principle 9244.5 (Window localization changes the interface, not the strength)**.** *Proving the unit-window target for any nonempty finite prime window already has the strength of* RH*. The local route is useful only as a change of interface: it exposes how an off-line zero would appear as coherent amplitude defect in every prime-local channel. It does not make a single Euler factor into a zero carrier.*

9244.2 Local Defect Rows

Definition 9244.6 (Prime defect row)**.** *Let*

$$s = 1/2 + \delta(s) + i\tau(s).$$

For a displayed prime row p, define the local defect row

$$D_p(s) := \log \mathsf{Amp}_p(s).$$

By Definition 9243.9,

$$D_p(s) = -\delta(s) \log p.$$

Definition 9244.7 (Finite-window total defect)**.** *For a finite prime window S, define*

$$D_S(s) := \sum_{p \in S} D_p(s).$$

Equivalently,

$$D_S(s) = -\delta(s) \sum_{p \in S} \log p.$$

Theorem 9244.8 (Apart non-fixed rows create local defects)**.** *If $|\delta(s)| \geq \varepsilon$ for some $\varepsilon : \mathbb{Q}_{>0}$, then for every displayed prime row p,*

$$|D_p(s)| \geq \varepsilon \log p.$$

Proof. By Definition 9244.6,

$$|D_p(s)| = |\delta(s)| \log p.$$

Since $p > 1$, the displayed prime row has $\log p > 0$. The premise $|\delta(s)| \geq \varepsilon$ therefore gives the stated lower bound. □

Theorem 9244.9 (Finite-window defect accumulation)**.** *If $|\delta(s)| \geq \varepsilon$ for some $\varepsilon : \mathbb{Q}_{>0}$, then for every finite prime window S,*

$$|D_S(s)| \geq \varepsilon \sum_{p \in S} \log p.$$

Proof. The equality in Definition 9244.7 gives

$$|D_S(s)| = |\delta(s)| \sum_{p \in S} \log p.$$

The lower bound follows from $|\delta(s)| \geq \varepsilon$. □

9244.3 Carrier Gluing

Definition 9244.10 (Completed gluing surface)**.** *The completed gluing surface for a zeta row consists of the compatible packet*

$$\mathsf{FinitePrimeRows}+\mathsf{ArchimedeanRow}+\mathsf{FunctionalEquation}+\mathsf{PoleLedger}+\mathsf{ZeroLedger}+\mathsf{BranchTransport}.$$

Here $\mathsf{ArchimedeanRow}$ *denotes the* $\pi^{-s/2}\Gamma(s/2)$ *completion row. This surface is the carrier-level context of Definition 9243.2.*

Theorem 9244.11 (Pure finite windows are not zero carriers)**.** *A pure finite prime window can detect amplitude defect, but it cannot by itself decide whether the recovered zeta carrier has a non-trivial zero at the displayed row.*

Proof. By Definition 9244.8, finite prime windows read off local amplitude defects. By Definition 9243.3 and Definition 9243.2, the non-trivial zero ledger belongs to the recovered carrier assembled from Dirichlet, Euler, continuation, archimedean, functional-equation, pole, zero, branch, and transport rows. A pure finite prime window contains only a finite part of the Euler-product projection and omits the archimedean and zero-ledger rows. It therefore is not a carrier on which zeta-zero exclusion can be closed. □

Definition 9244.12 (Local-global zero exclusion)**.** *The local-global zero exclusion target is*

$$\mathsf{LocalGlobalZeroExclusion}_\zeta := \prod_{c:\mathcal{C}_\zeta} (\mathsf{ApartNonFix}_J(c) \to \mathsf{Apart}_0(\zeta(c))).$$

It is the assertion that a critical-strip row with positive distance from the mirror fixed surface cannot glue, through the completed zeta carrier, to a zero ledger.

Theorem 9244.13 (Local-global zero exclusion is the local proof target)**.** *Assume the bias readout is located and that* $\mathsf{Apart}_0(\zeta(c))$ *contradicts* $Z_\zeta(c)$*. Then* $\mathsf{LocalGlobalZeroExclusion}_\zeta$ *supplies the source fixed section* $\mathsf{ZeroFix}_\zeta$*, hence the BEDC-RH section target.*

Proof. The type of Definition 9244.12 is the apart orbit exclusion type of Definition 9334.3. Applying Definition 9334.7 yields $\mathsf{ZeroFix}_\zeta$. The equivalence between this fixed section and the BEDC-RH section target is Definition 9334.4. □

Principle 9244.14 (Local–global RH proof format)**.** *The local proof format has four displayed obligations: local defect rows for apart non-fixed inputs, finite-window accumulation of those defects, archimedean normalization into the completed carrier, and zero-ledger exclusion for completed rows carrying the defect. The composition of these obligations is exactly* $\mathsf{LocalGlobalZeroExclusion}_\zeta$*; without the final completed carrier step, the route remains a local projection analysis rather than an RH proof.*

Concretized in §8765: Finite prime-window gluing is realized as finite prime-row, defect-row, gluing-ledger, zero-boundary, transport, replay, provenance, and local NameCert data.

Chapter 9245

Prime Window Recursive Tower

chapter 9244 describes the local prime-window proof target for RH. The present chapter records the recursive tower form of the same target. Each layer is read through finite prime windows; layer transitions are allowed only through displayed readback, predicate inheritance, and certified recursive return; an exception must therefore become visible in some finite prime-window readout before it can be transported as a tower-internal object.

Concretized in §8765: FinitePrimeWindowGluingUp realizes the finite prime-window gluing route as displayed prime rows, defect rows, zero-boundary request, transport, replay, provenance, and local NameCert data.

9245.1 Prime-Window Layer Data

Definition 9245.1 (Prime-window tower layer)**.** *A prime-window tower layer is a concept layer*

$$\mathcal{L}_n = (R_n, \sim_n, P_n, \Lambda_n, \mathsf{Cert}_n, \mathsf{Aud}_n)$$

over the source carrier, together with a readout

$$P_n : T \to R_n$$

and, for each successor step, a readback map

$$\rho_{n+1} : R_{n+1} \to R_n$$

such that

$$\rho_{n+1}(P_{n+1}(t)) \sim_n P_n(t).$$

Definition 9245.2 (Layer prime coding)**.** *A layer prime coding for $\mathcal{L}_n$ is a code-readback pair*

$$\mathrm{code}_n : R_n \to \mathbb{N}, \qquad \mathrm{decode}_n : \mathrm{im}(\mathrm{code}_n) \to R_n$$

with certificate

$$\mathrm{decode}_n(\mathrm{code}_n(r)) \sim_n r.$$

Definition 9245.3 (Layer prime-window split)**.** *For a finite prime window $S \subseteq \mathsf{Prime}^{\uparrow}$, the S-window split of the layer R_n is the readout*

$$\Pi_{n,S} : R_n \to \mathbb{N}^S, \qquad \Pi_{n,S}(r) := \big(v_p(\mathrm{code}_n(r))\big)_{p \in S}.$$

Theorem 9245.4 (Finite windows are layer quotients)**.** *For any layer n and finite prime window S, equality of $\Pi_{n,S}(r)$ and $\Pi_{n,S}(r')$ means that r and r' are indistinguishable by every displayed prime-coordinate readout in S.*

Proof. By Definition 9245.3, equality of the two S-window values is precisely the family of equalities

$$v_p(\mathrm{code}_n(r)) = v_p(\mathrm{code}_n(r'))$$

for all $p \in S$. These are exactly the displayed prime-coordinate reads available inside the finite window. □

9245.2 Readback and Recursive Return

Definition 9245.5 (Prime-code compatible readback)**.** *The successor readback $\rho_{n+1} : R_{n+1} \to R_n$ is prime-code compatible when there is a displayed map*

$$\kappa_{n+1} : \mathbb{N} \to \mathbb{N}$$

such that

$$\mathrm{code}_n(\rho_{n+1}(r)) = \kappa_{n+1}(\mathrm{code}_{n+1}(r)).$$

Theorem 9245.6 (Prime-window inheritance)**.** *If ρ_{n+1} is prime-code compatible, then every finite S-window readout on layer n pulls back along ρ_{n+1} to the corresponding finite coordinate family on layer $n+1$:*

$$\Pi_{n,S}(\rho_{n+1}(r)) = \big(v_p(\kappa_{n+1}(\mathrm{code}_{n+1}(r)))\big)_{p\in S}.$$

Proof. Expand Definition 9245.3:

$$\Pi_{n,S}(\rho_{n+1}(r)) = \big(v_p(\mathrm{code}_n(\rho_{n+1}(r)))\big)_{p\in S}.$$

Substitute the compatibility equality of Definition 9245.5. The displayed formula follows. □

Definition 9245.7 (Prime recursive return)**.** *The prime recursive return at layer n is the composite*

$$R_n \xrightarrow{\mathrm{code}_n} \mathbb{N} \xrightarrow{p\mapsto v_p} \mathbb{N}^{(\mathsf{Prime}^{\uparrow})} \xrightarrow{\mathrm{rec}} \mathbb{N} \xrightarrow{\mathrm{decode}_n} R_n,$$

where

$$\mathrm{rec}(a) := \prod_{p\in\mathsf{Prime}^{\uparrow}} p^{a_p}$$

for finite-support exponent families.

Theorem 9245.8 (Recursive return closes on the layer)**.** *Assume unique factorization for $\mathrm{code}_n(r)$ and the decode certificate of Definition 9245.2. Then the prime recursive return sends r back to an equivalent row:*

$$\mathrm{decode}_n\big(\mathrm{rec}(p \mapsto v_p(\mathrm{code}_n(r)))\big) \sim_n r.$$

Proof. Unique factorization gives

$$\mathrm{rec}(p \mapsto v_p(\mathrm{code}_n(r))) = \mathrm{code}_n(r).$$

The decode certificate then gives $\mathrm{decode}_n(\mathrm{code}_n(r)) \sim_n r$. □

9245.3 No-Exception Invariant

Definition 9245.9 (Layer zero carrier and window no-exception)**.** *Let $\mathcal{Z}_n \subseteq R_n$ be the layer-visible critical-strip zeta-zero carrier. For a nonempty finite prime window S, define*

$$\mathsf{NoExc}_n(S) := \prod_{z \in \mathcal{Z}_n} \mathsf{UnitWindow}_S(z).$$

The corresponding exception type is

$$\mathsf{Exc}_n(S) := \sum_{z \in \mathcal{Z}_n} \neg\mathsf{UnitWindow}_S(z).$$

Theorem 9245.10 (Nonempty window no-exception is layer RH)**.** *For a nonempty finite prime window S, the predicate $\mathsf{NoExc}_n(S)$ is equivalent to the layer statement that every visible zero row lies on the half surface.*

Proof. For each $z \in \mathcal{Z}_n$, Definition 9244.2 identifies $\mathsf{UnitWindow}_S(z)$ with $\mathsf{OnCritLine}(z)$. Taking the dependent product over $\mathcal{Z}_n$ gives the stated equivalence. □

Theorem 9245.11 (No-exception inheritance)**.** *Suppose the successor readback preserves the layer zero carriers and the finite-window readouts. Then*

$$\mathsf{NoExc}_n(S) \to \mathsf{NoExc}_{n+1}(S).$$

Proof. Let $z_{n+1} \in \mathcal{Z}_{n+1}$. Zero-carrier preservation gives $\rho_{n+1}(z_{n+1}) \in \mathcal{Z}_n$. From $\mathsf{NoExc}_n(S)$ one gets $\mathsf{UnitWindow}_S(\rho_{n+1}(z_{n+1}))$. Window-readout preservation transports this row to $\mathsf{UnitWindow}_S(z_{n+1})$. Since z_{n+1} was arbitrary, $\mathsf{NoExc}_{n+1}(S)$ follows. □

Corollary 9245.12 (Recursive no-exception propagation)**.** *If $\mathsf{NoExc}_{n_0}(S)$ holds at one layer and every later successor step satisfies the hypotheses of Definition 9245.11, then $\mathsf{NoExc}_m(S)$ holds for every $m \geq n_0$.*

Proof. Iterate Definition 9245.11 along the finite interval from n_0 to m. □

9245.4 Prime-Recursive RH Target

Theorem 9245.13 (Exception visibility in finite windows)**.** *If a row s has a positive off-line bias witness $|\delta(s)| \geq \varepsilon$, then every nonempty finite prime window sees a failure of the unit-window predicate.*

Proof. By Definition 9244.8, every displayed prime row has a nonzero local defect. Hence for each p in a nonempty window, $\mathsf{Amp}_p(s) \neq 1$, so $\mathsf{UnitWindow}_S(s)$ fails. □

Theorem 9245.14 (Prime-recursive no-exception target)**.** *Assume the tower has layer prime codings, prime-code compatible readbacks, recursive returns closed by unique factorization, readback preservation of zero carriers and finite-window readouts, and the completed-carrier exclusion target*

$$\mathsf{LocalGlobalZeroExclusion}_\zeta.$$

Then the tower has no RH exception: every critical-strip zeta-zero row projects to the half surface.

Proof. The finite prime-window apparatus makes any positive off-line bias visible by Definition 9245.13. The recursive return theorem Definition 9245.8 keeps the finite-window analysis inside the same layer, and Definition 9245.11 transports no-exception rows through compatible readbacks. The only remaining global step is the completed-carrier exclusion target Definition 9244.12. By Definition 9244.13, that target supplies the source fixed section and therefore the BEDC-RH section target. □

Principle 9245.15 (Prime-recursive no-exception boundary)**.** *The formal content of the prime-window recursive tower is not that every local factor closes* RH*. It is that any noncentral radial defect must be visible in finite prime windows, must survive only through certified readback and recursive return, and must finally fail to glue to the completed zeta zero ledger. The named target remains* $\mathsf{LocalGlobalZeroExclusion}_{\zeta}$*.*

Chapter 9246

Local–Global Zero Exclusion Target

chapter 9244 and chapter 9245 reduce the local prime-window route to one remaining object:

$$\mathsf{LocalGlobalZeroExclusion}_\zeta.$$

The present chapter records that object as the proof target. It does not introduce another route layer. It separates the local finite-window algebra, the source-compatibility rows, the archimedean completion row, and the carrier-level incompatibility obligation.

9246.1 Target Statement

Definition 9246.1 (Local-global zero exclusion target)**.** *The target is the dependent product*

$$\mathsf{LocalGlobalZeroExclusion}_\zeta := \prod_{c:\mathcal{C}_\zeta} (\mathsf{ApartNonFix}_J(c) \to \mathsf{Apart}_0(\zeta(c))).$$

Equivalently, it says that a critical-strip row separated from the mirror fixed surface by a positive rational witness cannot carry a zeta-zero row inside the completed zeta carrier.

Definition 9246.2 (Zero-ledger fixed-half geometry)**.** *The zero-ledger fixed-half geometry is the inclusion-style statement*

$$\mathsf{ZeroLedger}_\zeta \subseteq \mathrm{Fix}(J),$$

read internally as

$$\prod_{c:\mathcal{C}_\zeta} (Z_\zeta(c) \to J_\zeta(c) = c).$$

The notation does not introduce a host-side point set of zeros. It is the source fixed-section target of Definition 9334.3, read inside the completed zeta carrier of Definition 9254.13.

Definition 9246.3 (Zero, fixed, and off-fixed subcarriers)**.** *Let $\mathbb{G}^0_\zeta$ denote the RH-neutral zeta carrier of Definition 9250.17. The zero-ledger, fixed-half, and off-fixed subcarriers are the pullback carriers*

$$\mathbb{Z}_\zeta := \sum_{g:\mathbb{G}^0_\zeta} \mathsf{ZeroLedger}(g),$$

$$\mathbb{H}_\zeta := \sum_{g:\mathbb{G}^0_\zeta} \mathsf{FixHalf}(g),$$

and

$$\mathbb{O}_\zeta := \sum_{g:\mathbb{G}^0_\zeta} \mathsf{ApartNonFix}_J(g).$$

They are not compared as unrelated types. They are subcarriers over the same ambient recovered carrier, and their projections to $\mathbb{G}^0_\zeta$ carry the SameSource or SameCarrier certificates required for any containment, intersection, or apartness claim.

Principle 9246.4 (Containment, not orthogonality)**.** *The RH geometry is not an orthogonality claim between $\mathbb{Z}_\zeta$ and $\mathbb{H}_\zeta$. It is the factorization of the zero-ledger projection through the fixed-half subcarrier, equivalently the containment*

$$\mathbb{Z}_\zeta \subseteq \mathbb{H}_\zeta.$$

The apartness target concerns $\mathbb{Z}_\zeta$ and the off-fixed subcarrier $\mathbb{O}_\zeta$:

$$\mathbb{Z}_\zeta \cap \mathbb{O}_\zeta = \varnothing$$

with a positive-apartness witness rather than a bare empty-intersection assertion.

Principle 9246.5 (Recovered-carrier geometry, not point-set geometry)**.** *The geometric ambient object for the RH target is $\mathbb{G}^0_\zeta$, not the complex plane regarded as a bare point set. The complex coordinate s is a readout of a recovered carrier row whose data include Dirichlet, Euler, continuation, archimedean-completion, functional-equation, pole-ledger, zero-ledger, branch-transport, and prime-fingerprint projections. Consequently the traditional statement*

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \Re(s) = 1/2$$

is the complex-plane projection of the carrier containment

$$\mathbb{Z}_\zeta \subseteq \mathbb{H}_\zeta.$$

Equivalently, it is the assertion that the zero-ledger projection from the recovered carrier factors through the fixed-half subcarrier before any external complex-plane reading is taken.

Principle 9246.6 (Fixed half as rational center with phase fiber)**.** *The fixed-half subcarrier is not an irrational-number subcarrier. Its center row is the rational half value $1/2$. The residual freedom on the fixed surface is the phase fiber*

$$s = \frac{1}{2} + i\tau,$$

read in centered prime channels as

$$U_p(s) = e^{-i\tau \log p}.$$

Thus irrational, transcendental, logarithmic, and phase data may occur in the fiber over the fixed center, while the fixed-half equation itself is the rational-center row $\Re(s) = 1/2$.

Theorem 9246.7 (Geometric compression of RH)**.** *The geometric statement*

$$\mathsf{ZeroLedger}_\zeta \subseteq \mathrm{Fix}(J)$$

is the same RH target as the section form $\mathsf{Sec}_{\mathsf{RH}}$.

Proof. By Definition 9246.2, the displayed inclusion is the product

$$\prod_{c:\mathcal{C}_\zeta} (Z_\zeta(c) \to J_\zeta(c) = c),$$

which is $\mathsf{ZeroFix}_\zeta$ in Definition 9334.3. The equivalence with $\mathsf{Sec}_{\mathsf{RH}}$ is Definition 9334.4. □

Theorem 9246.8 (Local-global zero exclusion gives the RH witness target)**.** *Assume the real bias readout is located and that* $\mathsf{Apart}_0(\zeta(c))$ *contradicts* $Z_\zeta(c)$*. Then Definition 9246.1 supplies the BEDC-RH witness target of Definition 9239.8.*

Proof. The target is the apart orbit exclusion used in Definition 9334.7. That theorem supplies $\mathsf{ZeroFix}_\zeta$. By Definition 9334.4, the source fixed section is equivalent to the BEDC-RH section form, which is the constructive witness target of Definition 9239.8. □

9246.2 Finite-Window Algebra

Definition 9246.9 (Prime-window packet)**.** *For a nonempty finite prime window* S *and a row* s*, the prime-window packet consists of the finite data*

$$\mathsf{PrimeWindow}_S(s) := (S,\ (p \in S),\ U_p(s),\ \mathsf{Amp}_p(s),\ D_S(s),\ \mathsf{UnitWindow}_S(s)),$$

together with the certificate rows of Definition 9244.1 and Definition 9244.7.

Theorem 9246.10 (Prime-window defect formula)**.** *If*

$$s = 1/2 + \delta(s) + i\tau(s),$$

then for every nonempty finite prime window S*,*

$$|D_S(s)| = |\delta(s)| \sum_{p \in S} \log p.$$

Consequently, $|D_S(s)| = 0$ *if and only if* $\mathsf{OnCritLine}(s)$*.*

Proof. The equality is Definition 9244.9 with the lower-bound premise specialized to equality through Definition 9244.7. Since each $\log p$ is positive and S is nonempty, the sum is positive. Thus $|D_S(s)| = 0$ exactly when $|\delta(s)| = 0$, equivalently when $\Re(s) = 1/2$, which is $\mathsf{OnCritLine}(s)$ by Definition 9239.7. □

9246.3 Carrier Compatibility Obligations

Definition 9246.11 (Zeta-carrier source compatibility)**.** *The zeta-carrier source compatibility obligation is the row*

$$\mathsf{ZetaCarrierCompatibility}_S(s)$$

asserting that the zero-ledger read of s *and the prime-window read* $\mathsf{PrimeWindow}_S(s)$ *are two projections of the same source row, with a displayed readback witness:*

$$\mathsf{SameSource}(\mathsf{ZeroLedger}(s), \mathsf{PrimeWindow}_S(s)).$$

Definition 9246.12 (Archimedean completion obligation)**.** *The archimedean completion obligation is the carrier row*

$$\mathsf{ArchimedeanCompletionRow}(s)$$

which supplies the $\pi^{-s/2}\Gamma(s/2)$ *completion term and the functional-equation mirror transport needed to enter the completed carrier:*

$$\mathsf{FinitePrimeRows} + \mathsf{ArchimedeanRow} + \mathsf{FunctionalEquation} \to \mathsf{CompletedCarrier}.$$

Definition 9246.13 (Defect-zero incompatibility)**.** *The central analytic obligation is*

$$\mathsf{DefectZeroIncompatibility}$$

with type

$$\prod_s (\mathsf{InCritStrip}(s) \times \mathsf{CompletedCarrier}(s) \times \sum_{S \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}} (S \neq \varnothing) \times (|D_S(s)| > 0) \to \mathsf{Apart}_0(\zeta(s))).$$

9246.4 Positive Defect Certificates

Definition 9246.14 (Finite radial energy)**.** *For a nonempty finite prime window* S *and positive weights* $(w_p)_{p\in S}$*, the finite radial energy of a row* s *is*

$$E_S(s) := \sum_{p\in S} w_p \big(\log |p^{1/2-s}|\big)^2.$$

Using the centered form

$$p^{1/2-s} = p^{-\delta(s)} e^{-i\tau(s)\log p},$$

this energy is

$$E_S(s) = \delta(s)^2 \sum_{p\in S} w_p (\log p)^2.$$

Proposition 9246.15 (Finite radial energy detects the fixed half)**.** *For every nonempty finite prime window* S *with positive weights,*

$$E_S(s) = 0 \quad \Longleftrightarrow \quad \mathsf{OnCritLine}(s).$$

Proof. The weight sum

$$\sum_{p\in S} w_p (\log p)^2$$

is positive because S is nonempty, every displayed weight is positive, and every prime has $\log p > 0$. Hence $E_S(s) = 0$ exactly when $\delta(s)^2 = 0$, equivalently $\delta(s) = 0$. This is the fixed-half condition $\Re(s) = 1/2$, namely $\mathsf{OnCritLine}(s)$. □

Definition 9246.16 (Constructive bad-zero packet)**.** *For a displayed positive rational* q*, a constructive bad-zero packet is*

$$\mathsf{BadZero}_q(s) := \mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \big(|\delta(s)| \geq q\big).$$

The third row is a positive radial-defect certificate. It is stronger than the bare negation $\delta(s) \neq 0$ *and is the form consumed by finite-window apartness arguments.*

Proposition 9246.17 (Bad zeros create finite radial energy)**.** *If* $\mathsf{BadZero}_q(s)$ *holds and* S *is a nonempty finite prime window with positive weights, then*

$$E_S(s) \geq q^2 \sum_{p \in S} w_p (\log p)^2 > 0.$$

Proof. The defect row of Definition 9246.16 gives $|\delta(s)| \geq q$. Squaring and substituting into Definition 9246.14 gives the displayed lower bound. The right-hand weight sum is positive by the proof of Definition 9246.15, so the energy is positive. □

Principle 9246.18 (No tail absorption row)**.** *Finite radial energy is a certificate interface, not a completed-carrier exclusion by itself. The missing carrier-level row is the assertion that a positive finite-window radial defect cannot be absorbed by the remaining Euler tail, the archimedean completion row, or the functional-equation transport while still carrying a zero ledger:*

$$\mathsf{CompletedCarrier}(s) \times \sum_{S \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}} (E_S(s) > 0) \Longrightarrow \mathsf{Apart}_0(\zeta(s)).$$

This is the radial-energy presentation of Definition 9246.13. It must be proved as a carrier-rigidity statement; it is not a consequence of finite enumeration alone.

Theorem 9246.19 (Bad-zero contradiction route)**.** *Assume source compatibility, archimedean completion, and the no-tail absorption row of Definition 9246.18. Assume also that* $\mathsf{Apart}_0(\zeta(s))$ *contradicts* $\mathsf{ZetaZero}(s)$*. Then no constructive bad-zero packet exists.*

Proof. Assume $\mathsf{BadZero}_q(s)$. Choose a nonempty finite prime window S with positive weights. By Definition 9246.17, this packet supplies $E_S(s) > 0$. Source compatibility identifies this finite-window defect with a projection of the same source row as the zero ledger, and the archimedean completion row places the data in the completed carrier. The no-tail absorption row yields $\mathsf{Apart}_0(\zeta(s))$, contradicting the $\mathsf{ZetaZero}(s)$ component of Definition 9246.16. □

9246.5 Obligation Assembly

Theorem 9246.20 (Four obligations assemble the target)**.** *Suppose the following rows are supplied for every relevant critical-strip row: prime-window packets with the defect formula, zeta-carrier source compatibility, archimedean completion into the completed carrier, and Definition 9246.13. Then* $\mathsf{LocalGlobalZeroExclusion}_\zeta$ *holds.*

Proof. Let $c : \mathcal{C}_\zeta$ and assume $a : \mathsf{ApartNonFix}_J(c)$. Choose a nonempty finite window S. By Definition 9246.10, the apart witness gives $|D_S(c)| > 0$. Source compatibility identifies the zero-ledger row and the prime-window row as projections of the same source item. The archimedean completion row places the finite-prime data in the completed carrier. Applying Definition 9246.13 yields $\mathsf{Apart}_0(\zeta(c))$, which is the required output of Definition 9246.1. □

Definition 9246.21 (Geometric rigidity contradiction schema)**.** *The geometric rigidity contradiction schema is the proof route*

$$\mathsf{ZeroLedger}(c) \times \mathsf{ApartNonFix}_J(c) \times \mathsf{CompletedCarrier}(c) \longrightarrow \bot$$

obtained by first strengthening the off-fixed premise to a positive finite-window defect and then applying the carrier-level incompatibility row

$$\mathsf{DefectZeroIncompatibility}.$$

Its constructive form is the stronger output

$$\mathsf{ApartNonFix}_J(c) \to \mathsf{Apart}_0(\zeta(c)),$$

which is exactly Definition 9246.1.

Theorem 9246.22 (Rigidity contradiction is constructive apartness)**.** *Assume zeta-carrier source compatibility, archimedean completion, the prime-window defect formula, and Definition 9246.13. Then every instance of the geometric rigidity contradiction schema is witnessed by positive apartness from zero.*

Proof. Let $c : \mathcal{C}_\zeta$ and assume $a : \mathsf{ApartNonFix}_J(c)$. By Definition 9246.10, a nonempty finite prime window supplies a positive defect $|D_S(c)| > 0$. Source compatibility identifies the zero-ledger, prime-window, archimedean, and functional-equation reads as projections of the same source row. The archimedean completion row places this data inside the completed carrier. Applying Definition 9246.13 gives $\mathsf{Apart}_0(\zeta(c))$. If a zero-ledger witness is also present, this apartness contradicts that zero witness; without the zero witness it is already the constructive output of Definition 9246.1. □

Principle 9246.23 (Geometric content of the remaining target)**.** *The fixed-half translations of Definition 9254.10 identify the same surface through real part, mirror fixedness, prime-unitary balance, information bias, Laplace bias, and* OnCritLine *witnesses. What remains is not another translation of that surface. It is the positive-apartness incompatibility*

$$(\mathcal{C}_\zeta \setminus \mathrm{Fix}(J)) \cap \mathsf{ZeroLedger}_\zeta = \varnothing$$

read constructively as

$$\mathsf{ApartNonFix}_J(c) \to \mathsf{Apart}_0(\zeta(c)).$$

Principle 9246.24 (No further route layer)**.** *The remaining mathematical task is not another projection layer, another route classifier, or another exception vocabulary. It is the carrier-level incompatibility row Definition 9246.13: a completed zeta carrier row with positive finite-window radial defect cannot also be a zero ledger.*

Concretized in §9056: finite NameCert carrier for the local-global zero exclusion target

Chapter 9247

Finite Hermitian Rouche Start for ξ

This chapter records the finite self-adjoint starting object for the RH route. It is deliberately phrased as a certificate interface. The construction of O_N uses only finite prime-power data through $\Lambda(n)$, a finite archimedean Γ-kernel quadrature, and finite Hilbert data. No zeta-zero list, zero-count table, or inverse spectral datum is an input. The chapter also separates two statements that must not be conflated: Rouche domination on the outer rectangle gives equality of zero counts in that rectangle, while zero localization on the critical line requires Rouche domination on the off-line components where the self-adjoint determinant is zero-free.

9247.1 Prime–Gamma finite packet

Definition 9247.1 (Prime–Gamma truncation packet)**.** *A prime–Gamma truncation packet at level N is the finite datum*

$$\mathsf{PGamma}_N := (\mathcal{P}_N, (w_{n,N})_{n\in\mathcal{P}_N}, \mathcal{Q}_N, (a_{q,N}, \omega_{q,N})_{q\in\mathcal{Q}_N}, \mathcal{H}_N, (e_1, \dots, e_{m_N})),$$

where:

1. *$\mathcal{P}_N \subset \{n \le N : \Lambda(n) \neq 0\}$ is a finite prime-power window;*
2. *$w_{n,N} \in \mathsf{Rat}^{\uparrow}$ are displayed finite weights attached to the $\Lambda(n)$ rows;*
3. *$\mathcal{Q}_N$ is a finite quadrature window for the archimedean Γ-kernel, with real weights $a_{q,N}$ and frequencies $\omega_{q,N}$;*
4. *$\mathcal{H}_N$ is a finite-dimensional Hermitian carrier with displayed basis $(e_1, \dots, e_{m_N})$.*

The packet is a source-side object. It contains primes, $\Lambda(n)$, and archimedean Γ data, but it contains no zero of ξ or ζ.

Definition 9247.2 (Completed finite explicit-formula form)**.** *Given PGamma_N, define the finite sesquilinear form $\mathcal{B}_N$ on $\mathcal{H}_N$ by*

$$\mathcal{B}_N(f,g) := b_N\langle f,g\rangle + \sum_{n\in\mathcal{P}_N} w_{n,N}\frac{\Lambda(n)}{\sqrt{n}}\widehat{f}(\log n)\overline{\widehat{g}(\log n)} + \sum_{q\in\mathcal{Q}_N} a_{q,N}\widehat{f}(\omega_{q,N})\overline{\widehat{g}(\omega_{q,N})}.$$

The scalar b_N is a displayed real counterterm. The last sum is the finite archimedean row; its quadrature target is the logarithmic derivative of the completion factor $\pi^{-s/2}\Gamma(s/2)$ on the chosen height window. The hats denote the finite readout functionals carried by the packet.

Lemma 9247.3 (Hermitian symmetry of the finite form)**.** *If all displayed coefficients b_N, $w_{n,N}$, and $a_{q,N}$ are real, then*

$$\mathcal{B}_N(g, f) = \overline{\mathcal{B}_N(f, g)}$$

for all $f, g \in \mathcal{H}_N$.

Proof. Each summand has the form $c\,\ell(f)\overline{\ell(g)}$ with $c \in \mathbb{R}$, or the Hermitian inner product term $b_N\langle f, g\rangle$. Interchanging f and g complex conjugates every summand. Finite summation preserves the identity. □

Definition 9247.4 (Finite Hermitian operator)**.** *The finite operator O_N is the unique matrix on $\mathcal{H}_N$ satisfying*

$$\langle O_N f, g\rangle = \mathcal{B}_N(f, g) \qquad (f, g \in \mathcal{H}_N).$$

Equivalently, in the displayed basis,

$$(O_N)_{ab} = \mathcal{B}_N(e_b, e_a).$$

Theorem 9247.5 (Self-adjointness is source-side)**.** *The operator O_N of Definition 9247.4 is Hermitian:*

$$O_N = O_N^*.$$

Consequently its eigenvalues $\lambda_1, \ldots, \lambda_{m_N}$ are real.

Proof. The matrix identity $(O_N)_{ab} = \overline{(O_N)_{ba}}$ is exactly Definition 9247.3 applied to e_a, e_b. A finite Hermitian matrix has real spectrum by the finite-dimensional spectral theorem. This proof uses only the finite prime–Gamma packet and does not mention any zero of ξ. □

9247.2 Determinant approximant

Definition 9247.6 (Zero-free amplitude)**.** *A zero-free amplitude attached to* PGamma_N *is an entire function*

$$A_N(s) = \exp(B_N(s)),$$

where B_N is a finite completed Euler–Gamma counterterm built from the same prime-power truncation, archimedean quadrature, and normalization rows. The important structural property is

$$A_N(s) \neq 0 \qquad (s \in \mathbb{C}).$$

Definition 9247.7 (Finite Hermitian determinant model)**.** *The determinant model associated to (O_N, A_N) is*

$$D_N(s) := \det\big((s - \tfrac{1}{2})I - iO_N\big)\, A_N(s).$$

Theorem 9247.8 (Zeros of the model lie on the critical line)**.** *Assume A_N is zero-free. Then every zero of D_N is of the form*

$$s = \frac{1}{2} + i\lambda_j$$

for a real eigenvalue λ_j of O_N.

Proof. By Definition 9247.5, the spectrum of O_N is real. Hence

$$\det\big((s - \tfrac{1}{2})I - iO_N\big) = \prod_{j=1}^{m_N}\big((s - \tfrac{1}{2}) - i\lambda_j\big).$$

The determinant factor therefore vanishes only at $s = 1/2 + i\lambda_j$. The amplitude contributes no zeros by Definition 9247.6. □

9247.3 Outer-rectangle Rouche certificate

Definition 9247.9 (Spectrally admissible height)**.** *For*

$$R_T := \{\, s \in \mathbb{C} : 0 \le \Re(s) \le 1,\ |\Im(s)| \le T \,\},$$

a height T is spectrally admissible for D_N when $\lambda_j \neq \pm T$ for every eigenvalue λ_j of O_N. Equivalently, the determinant factor has no zero on ∂R_T.

Definition 9247.10 (Finite boundary domination certificate)**.** *A finite boundary domination certificate for (N, T) consists of explicit positive rational bounds*

$$E^{\mathrm{cert}}_{N,T} \quad \text{and} \quad M^{\mathrm{cert}}_{N,T}$$

such that

$$\sup_{s \in \partial R_T} |\xi(s) - D_N(s)| \le E^{\mathrm{cert}}_{N,T},$$

$$0 < M^{\mathrm{cert}}_{N,T} \le \inf_{s \in \partial R_T} |D_N(s)|,$$

and

$$E^{\mathrm{cert}}_{N,T} < M^{\mathrm{cert}}_{N,T}.$$

The upper bound is decomposed into the prime tail, the $\Lambda(n)$ smoothing error, the archimedean Γ-quadrature error, and the zero-free amplitude normalization error:

$$E^{\mathrm{cert}}_{N,T} = E^{\mathrm{pr}}_{N,T} + E^{\Lambda}_{N,T} + E^{\Gamma}_{N,T} + E^{A}_{N,T}.$$

The lower bound may be certified from

$$|A_N(s)| \ge \alpha_{N,T} > 0$$

on ∂R_T and the spectral product

$$\inf_{s \in \partial R_T} \prod_{j=1}^{m_N} |(s - \tfrac{1}{2}) - i\lambda_j|.$$

Theorem 9247.11 (The boxed boundary inequality)**.** *If a finite boundary domination certificate for (N, T) is supplied, then*

$$\boxed{|\xi(s) - D_N(s)| < |D_N(s)| \quad (s \in \partial R_T).}$$

Proof. For $s \in \partial R_T$,

$$|\xi(s) - D_N(s)| \le E^{\mathrm{cert}}_{N,T} < M^{\mathrm{cert}}_{N,T} \le |D_N(s)|.$$

This is the displayed inequality. □

Theorem 9247.12 (Outer-rectangle Rouche count)**.** *Assume T is spectrally admissible and Definition 9247.10 holds. Then ξ and D_N have the same number of zeros in R_T, counted with multiplicity.*

Proof. The functions ξ and D_N are holomorphic on a neighbourhood of R_T and D_N has no zero on ∂R_T. The strict inequality of Definition 9247.11 is exactly the hypothesis of Rouche's theorem. Hence ξ and D_N have the same zero count inside R_T. □

Principle 9247.13 (The full rectangle gives count, not location)**.** *The implication*

$$|\xi - D_N| < |D_N| \text{ on } \partial R_T \quad \Longrightarrow \quad N_{R_T}(\xi) = N_{R_T}(D_N)$$

is a zero-count statement only. It does not, by itself, imply that the zeros of ξ occupy the same locations as the zeros of D_N, nor that the zeros of ξ lie on the critical line. The self-adjointness of O_N places the model zeros on the line; localization of the actual zeros requires Rouche control on zero-free off-line components.

9247.4 Off-line Rouche localization

Definition 9247.14 (Off-line rectangles)**.** *For $0 < \eta < 1/2$ define*

$$R^-_{T,\eta} := \{\, s : 0 \le \Re(s) \le 1/2 - \eta,\ |\Im(s)| \le T \,\},$$

$$R^+_{T,\eta} := \{\, s : 1/2 + \eta \le \Re(s) \le 1,\ |\Im(s)| \le T \,\}.$$

These rectangles are separated from the critical line by a positive rational margin η.

Definition 9247.15 (Off-line domination certificate)**.** *An off-line domination certificate for $(N, T, \eta, \pm)$ is the same finite prime–Λ–Γ error package as Definition 9247.10, but computed on $\partial R^{\pm}_{T,\eta}$:*

$$\sup_{s \in \partial R^{\pm}_{T,\eta}} |\xi(s) - D_N(s)| < \inf_{s \in \partial R^{\pm}_{T,\eta}} |D_N(s)|.$$

Because A_N is zero-free and the determinant zeros of D_N lie on $\Re(s) = 1/2$, the lower bound is a genuine zero-free lower bound on each off-line component.

Theorem 9247.16 (Off-line Rouche exclusion)**.** *Assume A_N is zero-free and $O_N = O_N^*$. If the off-line domination certificate holds for $R^-_{T,\eta}$, then ξ has no zero in $R^-_{T,\eta}$. If it holds for $R^+_{T,\eta}$, then ξ has no zero in $R^+_{T,\eta}$.*

Proof. By Definition 9247.8, all zeros of D_N lie on $\Re(s) = 1/2$. Thus D_N has no zero in either off-line rectangle. Applying Rouche's theorem on the chosen off-line rectangle gives equality of zero counts there. The model count is zero, so the ξ count is zero. □

Corollary 9247.17 (Finite-height critical-line conclusion)**.** *Fix T. Suppose that for every positive rational $\eta < 1/2$ there is some finite level $N = N(T, \eta)$ whose off-line domination certificates hold on both $R^-_{T,\eta}$ and $R^+_{T,\eta}$. Then every zero of ξ in R_T lies on $\Re(s) = 1/2$.*

Proof. Let ρ be a zero of ξ in R_T. If $\Re(\rho) < 1/2$, choose a positive rational $\eta < 1/2 - \Re(\rho)$; then $\rho \in R^-_{T,\eta}$, contradicting Definition 9247.16. The case $\Re(\rho) > 1/2$ is the same using $R^+_{T,\eta}$. Therefore $\Re(\rho) = 1/2$. □

9247.5 Non-circularity ledger

Principle 9247.18 (What self-adjointness proves)**.** *The self-adjoint step proves exactly this: the zeros of the finite model D_N contributed by the determinant factor are line-supported because $O_N = O_N^*$. This is non-circular because the spectrum of O_N is produced from the finite prime–Λ–Γ kernel, not from the zero set of ξ.*

Principle 9247.19 (What remains analytic)**.** *The remaining RH-facing obligation is not self-adjointness. It is the finite, checkable domination package*

$$\sup_{\partial R^{\pm}_{T,\eta}} |\xi - D_N| < \inf_{\partial R^{\pm}_{T,\eta}} |D_N|$$

with explicit prime tail, $\Lambda(n)$ smoothing, archimedean Γ-kernel, and amplitude lower-bound ledgers. Once these off-line inequalities are proved without zero input, Definition 9247.16 gives $r = 0$ on the corresponding height window.

Concretized in §9246: The finite Hermitian Rouche start supplies a spectral version of the local–global zero-exclusion hinge: line-supported model zeros come from self-adjointness, while the actual zeta-zero exclusion is carried by explicit off-line boundary-domination ledgers.

Chapter 9248

Prime Fingerprint Separation for RH

chapter 9243 identifies the critical line with the unit-amplitude surface of the centered prime channels. The present chapter records the separation row behind that identification. The point s is not treated as a bare geometric coordinate with prime data attached afterward. In the zeta recovery carrier, s is read by its responses to all prime-local probes.

9248.1 Prime fingerprint

Definition 9248.1 (Prime fingerprint)**.** *For a complex history s, its prime fingerprint is the family*

$$\mathsf{PF}(s) := \left(p^{-s}\right)_{p \in \mathsf{Prime}^{\uparrow}}.$$

Equivalently, for the multiplicative character

$$\chi_s(n) := n^{-s},$$

unique factorization gives

$$n = \prod_p p^{k_p} \qquad \Longrightarrow \qquad \chi_s(n) = \prod_p \chi_s(p)^{k_p}.$$

Thus the prime fingerprint is the local-probe presentation of the same Dirichlet projection row consumed by Definition 9243.1.

Definition 9248.2 (Prime-fingerprint equivalence)**.** *For complex histories s and t, write*

$$s \equiv_{\mathsf{PF}} t$$

when

$$\forall p \in \mathsf{Prime}^{\uparrow}, \quad p^{-s} = p^{-t}.$$

This is equality as observed by all prime-local zeta channels.

Proposition 9248.3 (Prime channels separate complex histories)**.** *On the displayed complex-history packet containing the prime rows* 2 *and* 3 *with their real logarithm rows,*

$$s \equiv_{\mathsf{PF}} t \quad \Longleftrightarrow \quad s = t.$$

Proof. The implication from $s = t$ to $s \equiv_{\mathsf{PF}} t$ is immediate. Conversely, assume $s \equiv_{\mathsf{PF}} t$ and put $a = s - t$. The prime rows 2 and 3 give

$$e^{-a \log 2} = 1, \qquad e^{-a \log 3} = 1.$$

Hence there are integers k and ℓ with

$$a \log 2 = 2\pi i k, \qquad a \log 3 = 2\pi i \ell.$$

If $a \neq 0$, then k and ℓ are not both zero and the displayed rows force

$$k \log 3 = \ell \log 2.$$

Exponentiating gives

$$3^k = 2^\ell,$$

which is impossible unless $k = \ell = 0$ by unique factorization. Therefore $a = 0$, so $s = t$. □

Principle 9248.4 (Primes are probes, not points)**.** *The prime rows are not points of the critical strip. They are separating local probes for the zeta information space. A complex history s is distinguished from t precisely when some prime-local response differs:*

$$s \neq t \quad \Longleftrightarrow \quad \exists p \in \mathsf{Prime}^{\uparrow}, \quad p^{-s} \neq p^{-t}.$$

9248.2 Mirror fingerprints

Definition 9248.5 (Mirror prime fingerprint)**.** *Let $J(s) = 1 - \overline{s}$ be the mirror map of Definition 9241.12. The mirror prime-fingerprint comparison for s is the pair*

$$(\mathsf{PF}(s), \mathsf{PF}(J(s))).$$

The comparison is collapsed when

$$\mathsf{PF}(s) = \mathsf{PF}(J(s)).$$

Proposition 9248.6 (Mirror changes amplitude bias, not phase clock)**.** *Write*

$$s = \frac{1}{2} + \delta + i\tau.$$

For the centered prime channel

$$U_p(s) = p^{1/2-s},$$

one has

$$U_p(s) = p^{-\delta} e^{-i\tau \log p}, \qquad U_p(J(s)) = p^{\delta} e^{-i\tau \log p}.$$

Thus the mirror preserves the prime phase clock and inverts the prime amplitude bias.

Proof. Since

$$J(s) = 1 - \overline{s} = \frac{1}{2} - \delta + i\tau,$$

direct substitution gives

$$U_p(s) = p^{1/2-(1/2+\delta+i\tau)} = p^{-\delta} e^{-i\tau \log p},$$

and

$$U_p(J(s)) = p^{1/2-(1/2-\delta+i\tau)} = p^{\delta} e^{-i\tau \log p}.$$

□

Corollary 9248.7 (Prime fingerprint collapses exactly on the mirror surface)**.** *For every complex history s,*

$$\mathsf{PF}(s) = \mathsf{PF}(J(s)) \quad \Longleftrightarrow \quad s = J(s) \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Proof. The first equivalence follows from Definition 9248.3 applied to s and $J(s)$. The second equivalence is the fixed-point calculation in Definition 9241.12. □

Corollary 9248.8 (Off-critical mirror pairs are prime-visible)**.** *If $s = 1/2 + \delta + i\tau$ with $\delta \neq 0$, then every displayed prime row p separates the centered channel amplitudes of s and $J(s)$:*

$$|U_p(s)| = p^{-\delta}, \qquad |U_p(J(s))| = p^{\delta} = |U_p(s)|^{-1},$$

and

$$|U_p(s)| \neq 1.$$

Proof. This is the modulus read of Definition 9248.6. Since $p > 1$ and $\delta \neq 0$, the positive scalar $p^{-\delta}$ is not one. □

9248.3 RH as prime-fingerprint indistinguishability

Theorem 9248.9 (Prime-fingerprint normal form of RH)**.** *The Riemann Hypothesis is equivalent to the assertion that every recognized non-trivial zeta-zero event is indistinguishable from its mirror by prime fingerprint. This is a scoped prime-channel reparameterization of the fixed-line output: the fingerprint row rewrites $s = J(s)$, while the zero-ledger exclusion remains the RH content:*

$$\mathsf{RH} \quad \Longleftrightarrow \quad \forall s,\ \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{PF}(s) = \mathsf{PF}(J(s)).$$

Proof. By Definition 9248.7, $\mathsf{PF}(s) = \mathsf{PF}(J(s))$ is equivalent to $\Re(s) = 1/2$. Substituting this equivalent output row into the BEDC statement of RH gives the displayed normal form. □

Corollary 9248.10 (No prime-visible mirror bias form)**.** *Equivalently,* RH *says that no recognized non-trivial zero event carries a prime-visible mirror amplitude bias:*

$$\forall s,\ \mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \forall p \in \mathsf{Prime}^{\uparrow}, \quad \left|p^{1/2-s}\right| = 1.$$

Proof. The output row is exactly the prime-unitary balance condition of Definition 9243.6. Its equivalence with the ordinary RH witness is Definition 9243.12; equivalently, it is the amplitude form of Definition 9248.9. □

Remark 9248.11 (Relation to mirror closure)**.** *The functional equation supplies mirror closure of the zero set, as in Definition 9241.13. The prime-fingerprint row asks for more: the two mirror-related zero packets must not be separated by any prime-local channel. By Definition 9241.14, this pointwise collapse is not supplied by mirror closure alone.*

Concretized in §64: Prime packets supply the local probe family used by the prime fingerprint.

Concretized in §71: The Dirichlet and Euler-product rows supply the half-plane channel on which $\mathsf{PF}(s)$ is read.

Concretized in §581: The zeta functional-equation ledger supplies the mirror transport row whose orbit collapse is tested by prime fingerprints.

Concretized in §548: Critical-line witness packets supply the fixed-surface row equivalent to prime-fingerprint collapse.

Concretized in §9000: Prime spectral fingerprint packets realize prime-local zeta response rows as finite NameCert data.

Chapter 9249

Prime-Skew Boundary Defect Refutation

chapter 9248 shows that an off-critical mirror pair is visible to centered prime channels, while chapter 9246 isolates the zero-defect incompatibility needed for the local-global route. The present chapter packages those two ideas as a conditional refutation interface. It does not prove the Riemann Hypothesis; it states that, once representation, prime-skew detection, and zero-defect incompatibility are all displayed, an off-line non-trivial zero cannot be inhabited.

Definition 9249.1 (Non-trivial zero and off-line zero packets)**.** *For a complex history s, write*

$$\mathcal{Z}_\zeta(s) := \mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s).$$

The fixed zero packet is

$$\mathsf{FixedZero}(s) := \mathcal{Z}_\zeta(s) \times \mathsf{OnCritLine}(s).$$

The off-line zero packet is

$$\mathsf{OffLineZero}(s) := \mathcal{Z}_\zeta(s) \times \mathsf{ApartCrit}(s),$$

where $\mathsf{ApartCrit}(s)$ *is a positive rational-level apartness witness between* $\Re(s)$ *and* $1/2$*. Thus* $\mathsf{FixedZero}(s)$ *is the mirror-fixed zero packet, and* $\mathsf{OffLineZero}(s)$ *is the mirror-split zero packet.*

Definition 9249.2 (Prime-visible mirror skew)**.** *For the mirror $J(s) = 1 - \overline{s}$ and centered prime channel*

$$U_p(s) := p^{1/2-s},$$

the prime-visible mirror skew is the family

$$\Delta_p(s) := \log |U_p(J(s))| - \log |U_p(s)|.$$

Writing

$$s = \frac{1}{2} + \delta + i\tau$$

gives

$$|U_p(s)| = p^{-\delta}, \qquad |U_p(J(s))| = p^{\delta},$$

and therefore

$$\Delta_p(s) = 2\delta \log p.$$

The existential skew row is

$$\mathsf{PrimeSkew}(s) := \sum_{p:\mathsf{Prime}^{\uparrow}} (\Delta_p(s)\#0).$$

Lemma 9249.3 (Off-line zeros produce prime-visible skew)**.** *For every complex history s,*

$$\mathsf{OffLineZero}(s) \to \mathsf{PrimeSkew}(s).$$

Proof. An off-line zero packet supplies $\mathsf{ApartCrit}(s)$, hence a nonzero real offset $\delta = \Re(s) - 1/2$. By Definition 9248.6, the centered prime channels read

$$|U_p(s)| = p^{-\delta}, \qquad |U_p(J(s))| = p^{\delta}.$$

Thus

$$\Delta_p(s) = 2\delta \log p.$$

For any displayed prime row p, the row $p > 1$ gives $\log p > 0$, so $\delta\#0$ implies $\Delta_p(s)\#0$. Choosing such a prime row gives the required element of $\mathsf{PrimeSkew}(s)$. □

Definition 9249.4 (Bulk zero atom and prime defect)**.** *Let T_Ω be a displayed bulk carrier with a zeta readout*

$$\pi_\zeta : T_\Omega \to R_\zeta.$$

The non-trivial zero atom over $x : T_\Omega$ is

$$\mathsf{NTZeroAtom}_\Omega(x) := \mathsf{ZetaZero}(\pi_\zeta(x)) \times \mathsf{InCritStrip}(\pi_\zeta(x)).$$

For boundary layers i with projections

$$\pi_i : T_\Omega \to R_i,$$

write $\mathsf{PrimeDefect}_i(x)$ for a boundary row saying that the prime-visible invariant of x is apart from zero at layer i. The global defect row is

$$\mathsf{PrimeDefect}_\Omega(x) := \sum_i \mathsf{PrimeDefect}_i(x).$$

Definition 9249.5 (Boundary faithfulness and zero-defect incompatibility)**.** *Prime-skew boundary faithfulness is the statement*

$$\mathsf{PrimeSkewDetect}_\Omega := \prod_{x:T_\Omega} (\mathsf{PrimeSkew}(\pi_\zeta(x)) \to \mathsf{PrimeDefect}_\Omega(x)).$$

Prime-defect zero incompatibility is the statement

$$\mathsf{PrimeDefectZeroIncompatibility}_\Omega := \prod_{x:T_\Omega} (\mathsf{NTZeroAtom}_\Omega(x) \times \mathsf{PrimeDefect}_\Omega(x) \to \bot).$$

Together they say that a prime-visible skew cannot be invisible to all boundary readouts, and that an accepted non-trivial zero atom cannot also carry such a boundary defect.

Definition 9249.6 (Bulk representation of zero packets)**.** *A bulk representation row for non-trivial zero packets is the statement*

$$\mathsf{BulkRep}_\Omega := \prod_s (\mathcal{Z}_\zeta(s) \to \sum_{x:T_\Omega} \pi_\zeta(x) \sim s).$$

The equivalence row is a classifier transport certificate; it is not a claim that s is a bare point of the bulk carrier.

Theorem 9249.7 (Boundary-defect refutation of off-line zeros)**.** *Assume* $\mathsf{BulkRep}_\Omega$*,* $\mathsf{PrimeSkewDetect}_\Omega$*, and* $\mathsf{PrimeDefectZeroIncompatibility}_\Omega$*. Then*

$$\prod_s (\mathsf{OffLineZero}(s) \to \bot).$$

Proof. Fix s and assume $H : \mathsf{OffLineZero}(s)$. The first component of H is $\mathcal{Z}_\zeta(s)$, so $\mathsf{BulkRep}_\Omega$ supplies $x : T_\Omega$ with $\pi_\zeta(x) \sim s$. Transporting the zero and critical-strip rows along this equivalence gives

$$\mathsf{NTZeroAtom}_\Omega(x).$$

By Definition 9249.3, H also supplies $\mathsf{PrimeSkew}(s)$. Transporting this skew row along $\pi_\zeta(x) \sim s$ gives $\mathsf{PrimeSkew}(\pi_\zeta(x))$. By $\mathsf{PrimeSkewDetect}_\Omega$, this yields

$$\mathsf{PrimeDefect}_\Omega(x).$$

Now x carries both $\mathsf{NTZeroAtom}_\Omega(x)$ and $\mathsf{PrimeDefect}_\Omega(x)$, contradicting $\mathsf{PrimeDefectZeroIncompatibility}_\Omega$. Hence $\mathsf{OffLineZero}(s) \to \bot$, and s was arbitrary. □

Corollary 9249.8 (Located critical-strip zeros give the RH predicate)**.** *Assume the hypotheses of Definition 9249.7 and a located critical-line classifier*

$$\prod_s (\mathcal{Z}_\zeta(s) \to \mathsf{OnCritLine}(s) + \mathsf{ApartCrit}(s)).$$

Then the BEDC RH predicate of Definition 9239.7 is inhabited.

Proof. Given $\mathcal{Z}_\zeta(s)$, apply the located classifier. In the $\mathsf{OnCritLine}(s)$ case, the desired witness is present. In the $\mathsf{ApartCrit}(s)$ case, the pair $\mathcal{Z}_\zeta(s) \times \mathsf{ApartCrit}(s)$ is an off-line zero packet, contradicted by Definition 9249.7. Thus the only surviving branch supplies $\mathsf{OnCritLine}(s)$. □

Corollary 9249.9 (Off-line zero type is empty)**.** *Under the hypotheses of Definition 9249.7, the sigma type*

$$\sum_s \mathsf{OffLineZero}(s)$$

maps to $\bot$.

Proof. Given $(s, H) : \sum_s \mathsf{OffLineZero}(s)$, apply Definition 9249.7 to s and H. □

Principle 9249.10 (Scope of the boundary-defect route)**.** *The route proves no unconditional analytic incompatibility. Its two substantive premises are boundary faithfulness and zero-defect incompatibility. If boundary faithfulness fails, prime-visible skew may be lost before reaching the observed boundary layers. If zero-defect incompatibility fails, a non-trivial zero atom can coexist with a nonzero prime defect. Either failure identifies a precise analytic obligation rather than a proof or disproof of* RH*.*

Concretized in §8848: Prime-skew boundary-defect packets realize the conditional route as finite zero-representation, prime-skew, boundary-defect, incompatibility, refusal, transport, replay, provenance, and NameCert rows without proving RH.

Concretized in §9006: Prime-skew defect ledgers realize the off-line zero, prime-visible skew, and local-global defect incompatibility rows as finite NameCert data.

Chapter 9250

Zeta as a Self-Similar Recovered Carrier

chapter 9243 defines ZETA as a recovered carrier rather than as one formula, and chapter 9248 reads complex histories through prime-local probes. The present chapter records the corresponding self-similar reading. It is a relative reading inside the BEDC recovery surface: zeta is not a geometric self-similar figure, but a carrier recovered from compatible multiplicative, prime-local, Laplace, continuation, completion, mirror, and zero-ledger rows.

Definition 9250.1 (Prime-local geometric fiber)**.** *For a displayed prime row $p \in \mathsf{Prime}^{\uparrow}$ and a complex history s in a convergence window where the local series is licensed, the prime-local geometric fiber is*

$$E_p(s) := \sum_{k \geq 0} p^{-ks}.$$

It carries the one-step recursion

$$p^{-ks} \mapsto p^{-(k+1)s}$$

with local ratio p^{-s}. Equivalently, on the Euler-product window,

$$E_p(s) = (1 - p^{-s})^{-1}.$$

The fiber is local to the prime row and to the declared convergence and branch ledgers.

Definition 9250.2 (Multiplicative self-similar zeta packet)**.** Lean4 (definition) BEDC.Derived.RHRoute.ZeroGenerati
Lean4✓ BEDC.Derived.RHRoute.ZeroGenerationInitiality.multiplicativeSelfSimilarZetaPacket
zeroLedger count

The multiplicative self-similar zeta packet is the compatible recovery surface

$$\mathcal{Z}_{\mathrm{ss}} := \mathrm{Rec}\begin{pmatrix} \mathsf{PrimeLocalFibers}, & \mathsf{DirichletRow}, & \mathsf{EulerRow}, & \mathsf{LaplaceRow}, \\ \mathsf{AnalyticContinuation}, & \mathsf{ArchimedeanCompletion}, & \mathsf{FunctionalEquation}, & \mathsf{ZeroLedger} \end{pmatrix}.$$

It is the self-similar reading of the recovered carrier ZETA *of Definition 9243.2 and Definition 9254.13. The recovery symbol records compatibility and transport among the listed rows; it does not introduce a primitive carrier outside those rows.*

Principle 9250.3 (Zeta self-similarity is carrier-level)**.** *The expressions*

$$\sum_{n \geq 1} n^{-s}$$

and

$$\prod_{p\in\mathsf{Prime}^{\uparrow}}(1-p^{-s})^{-1}$$

are projections of ZETA *on the right half-plane, not standalone definitions of the full carrier. The self-similar structure belongs to the recovered carrier only after the Dirichlet row, Euler row, analytic continuation row, archimedean completion row, functional equation row, pole ledger, zero ledger, and branch transport ledger are displayed together.*

Proposition 9250.4 (Euler gluing is multiplicative self-similarity)**.** *On the displayed Euler-product window, the zeta projection is obtained by gluing the prime-local geometric fibers:*

$$\zeta_D(s)=\prod_{p\in\mathsf{Prime}^{\uparrow}}E_p(s)=\prod_{p\in\mathsf{Prime}^{\uparrow}}(1-p^{-s})^{-1}.$$

This is a multiplicative self-similar projection indexed by ordinary prime rows.

Proof. By Definition 9243.4, the Dirichlet row and Euler row agree on $\Re(s)>1$. The proof of that proposition groups the Dirichlet terms by the unique factorization row

$$n=\prod_p p^{k_p}$$

and rewrites

$$n^{-s}=\prod_p p^{-k_p s}.$$

Thus the global right-half-plane projection is the product of the local geometric fibers $E_p(s)$. □

Definition 9250.5 (Prime-occupancy partition readout)**.** *Let $T_{\mathbb{P}}$ be the prime-occupancy carrier of Definition 9414.147. On a displayed right-half-plane convergence window, the prime-occupancy partition readout is*

$$Z_{\mathbb{P}}(s):=\sum_{k\in T_{\mathbb{P}}}\exp\left(-s\sum_{p\in\mathbb{P}}k_p\log p\right).$$

The sum ranges over finite-support prime occupancy rows. It is therefore the analytic readout of the multiplicative arithmetic presentation, not a separate unledgered source carrier.

Proposition 9250.6 (Zeta is the prime-occupancy partition readout)**.** *On the Euler-product window,*

$$Z_{\mathbb{P}}(s)=\zeta_D(s).$$

Equivalently,

$$Z_{\mathbb{P}}(s)=\prod_{p\in\mathsf{Prime}^{\uparrow}}\left(\sum_{k\geq 0}p^{-ks}\right)=\prod_{p\in\mathsf{Prime}^{\uparrow}}(1-p^{-s})^{-1}.$$

Proof. For a finite-support occupancy row k, its readback integer is

$$\Psi(k)=\prod_p p^{k_p}$$

by Definition 9414.148. Thus

$$\log\Psi(k)=\sum_p k_p\log p$$

and

$$\exp\left(-s\sum_p k_p \log p\right) = \prod_p p^{-sk_p}.$$

Summing over all finite-support occupancy rows is the same as summing over positive integers by unique factorization, hence gives the Dirichlet row. Splitting the independent prime coordinates gives the Euler product by Definition 9250.4. □

Principle 9250.7 (Zero packets are recovered-carrier cancellations)**.** *In the prime-occupancy presentation, ζ_D is the analytic partition readout of finite prime configurations on the right half-plane. A non-trivial zero packet is not a bare row of $T_{\mathbb{P}}$ and not a zero of one local geometric fiber. It belongs to the completed recovered carrier assembled from Dirichlet, Euler, continuation, archimedean completion, functional-equation, pole, zero, branch, and transport ledgers.*

Proposition 9250.8 (Dirichlet projection is a discrete Laplace readout)**.** *For the zeta coefficients $a_n = 1$, the Dirichlet projection is the discrete Laplace readout of the atomic source*

$$\mu_\zeta := \sum_{n\geq 1} \delta_{\log n}.$$

That is, on any displayed convergence window,

$$\sum_{n\geq 1} n^{-s} = \mathcal{L}[\mu_\zeta](s).$$

Proof. This is Definition 9254.12 with $a_n = 1$. The equality follows from

$$e^{-s\log n} = n^{-s}$$

on the declared branch and convergence window. □

Definition 9250.9 (Zeta recovered-geometry packet)**.** *The zeta recovered-geometry packet is the records-side synthesis*

$$\mathsf{ZetaGeom} := \mathrm{Rec}\begin{pmatrix} T_{\mathbb{P}}, & \mathsf{PrimeFingerprint}, & \mathsf{EulerPhaseChannels}, & \mathsf{LaplaceKernel}, \\ \mathsf{FixedHalfMirror}, & \mathsf{FibonacciHalfModel}, & \mathsf{ZetaRecoveryRows} & \end{pmatrix}.$$

Here $T_{\mathbb{P}}$ is the finite-support prime-occupancy carrier, PrimeFingerprint *is the family $(p^{-s})_{p\in\mathsf{Prime}^\uparrow}$,* EulerPhaseChannels *records the decompositions*

$$p^{-s} = p^{-\sigma}e^{-i\tau\log p}, \qquad U_p(s) = p^{1/2-s},$$

LaplaceKernel *records $e^{-s\log n} = n^{-s}$,* FixedHalfMirror *records $J(s) = 1 - \overline{s}$ and $s = 1/2 + \delta + i\tau$,* FibonacciHalfModel *records the classifier-relative comparison with the golden fixed-half pattern of Definition 9250.24, and* ZetaRecoveryRows *abbreviates the Dirichlet, Euler, continuation, archimedean-completion, functional-equation, pole, zero, branch, and transport ledgers of Definition 9254.13.*

Definition 9250.10 (Displayed zeta constraint diagram)**.** *A displayed zeta constraint diagram is a typed records-side diagram*

$$\mathcal{D}_\zeta := (I,\ (R_i, \sim_i, P_i, C_i, \Lambda_i, \mathsf{Cert}_i)_{i:I},\ (\rho_{ji})_{j\to i})$$

over one source carrier, where each P_i is a displayed readout, each $C_i : R_i \to \mathsf{Prop}$ is a displayed constraint row, and every edge $j \to i$ carries a readback map $\rho_{ji} : R_j \to R_i$ with the source compatibility row

$$\rho_{ji}(P_j(t)) \sim_i P_i(t).$$

For the zeta packet, the diagram contains at least the prime-axis, Dirichlet, Euler, Laplace, continuation, archimedean-completion, functional-equation, pole-ledger, zero-ledger, branch-transport, and provenance rows already listed in Definition 9250.9. The phrase "all zeta constraints" refers only to such a displayed diagram, not to an untyped totality of predicates.

Definition 9250.11 (Compatible zeta geometry carrier)**.** *For a displayed zeta constraint diagram $\mathcal{D}_\zeta$, the compatible zeta geometry carrier is the dependent carrier*

$$\mathbb{G}_{\mathcal{D}_\zeta} := \sum_{(r_i):\prod_{i:I} R_i} \left(\prod_{i:I} C_i(r_i)\right) \times \left(\prod_{j\to i} \rho_{ji}(r_j) \sim_i r_i\right) \times \Lambda_{\mathcal{D}_\zeta}.$$

The source-side version is

$$T_{\mathcal{D}_\zeta} := \sum_{t:T} \prod_{i:I} C_i(P_i(t)).$$

The canonical readout sends t to the compatible family $(P_i(t))_{i:I}$ together with the displayed readback and ledger rows.

Principle 9250.12 (Constraint totality is relative to a diagram)**.** *The infinite-dimensional geometry of zeta is a compatible limit of a displayed constraint diagram. It is not the complex plane, not the Euler product alone, and not the untyped collection of every sentence one could state about ζ. The carrier $\mathbb{G}_{\mathcal{D}_\zeta}$ records precisely the rows, constraints, readbacks, fibers, continuations, and ledgers named by $\mathcal{D}_\zeta$.*

Definition 9250.13 (Displayed zeta generation signature)**.** *A displayed zeta generation signature at layer ℓ is a BEDC-licensed multi-sorted signature*

$$\mathcal{F}_{\zeta,\ell} \quad \textit{on} \quad \mathsf{Sort}_\zeta := \{\mathsf{Obj}, \mathsf{Win}, \mathsf{Code}, \mathsf{Cert}, \mathsf{Ledger}\}.$$

The object sort carries the recovered carrier rows; the window sort carries inspection, refinement, pairing, summing, encoding, and sealing windows; the code sort carries public readbacks; the certificate sort carries stability and correctness witnesses; and the ledger sort carries source, pattern, classifier, stability, and replay rows. Typical window constructors have the forms

$$\mathsf{idWin}, \quad \mathsf{composeWin}, \quad \mathsf{refineWin}, \quad \mathsf{pairWin}, \quad \mathsf{sumWin}, \quad \mathsf{encodeWin}, \quad \mathsf{sealWin}.$$

Each constructor has displayed input sorts and a displayed ledger row. The signature ranges only over the rows named by the ambient zeta diagram; it is not a quantifier over every analytic predicate on the complex plane.

Definition 9250.14 (Generated zeta universe)**.** *When the ambient layer supplies the initiality certificate for Definition 9250.13, the generated zeta universe is the multi-sorted initial algebra*

$$\mathbb{U}_{\zeta,\ell} := \mu\mathcal{F}_{\zeta,\ell}.$$

Its components are written

$$U^{\mathsf{Obj}}_{\zeta,\ell}, \quad U^{\mathsf{Win}}_{\zeta,\ell}, \quad U^{\mathsf{Code}}_{\zeta,\ell}, \quad U^{\mathsf{Cert}}_{\zeta,\ell}, \quad U^{\mathsf{Ledger}}_{\zeta,\ell}.$$

The predicate

$$\mathsf{Generated}^{s}_{\zeta,\ell}(x)$$

means that $x : U^{s}_{\zeta,\ell}$ is obtained by the displayed constructors of $\mathcal{F}_{\zeta,\ell}$. A sealing or completion row is generated only when its compatible thread, modulus, branch ledger, and replay ledger are also displayed. If the initiality certificate is absent, the notation denotes an obligation row, not an additional axiom.

Principle 9250.15 (Internal origin of windows and certificates)**.** *In the generated zeta universe, object rows, detection windows, public codes, certificates, and ledgers have the same generative origin. For each sort $s \in \mathsf{Sort}_{\zeta}$, initial-algebra induction gives*

$$\prod_{x:U^{s}_{\zeta,\ell}} \mathsf{Generated}^{s}_{\zeta,\ell}(x).$$

Thus a detector for a zero packet, a branch window for continuation, or a certificate for a compatibility edge is not an external observer of $\mathbb{G}_{\mathcal{D}_\zeta}$. It is another generated component of the same displayed signature.

Principle 9250.16 (Fixed-half closure is not a generator)**.** *The fixed-half subcarrier may be read inside $\mathbb{G}_{\mathcal{D}_\zeta}$ only after the neutral zeta diagram, its window rows, and its certificate rows have already been generated. The RH claim is therefore not a constructor of $\mathcal{F}_{\zeta,\ell}$ and not a defining row of the neutral carrier. It is the section target asserting that the zero-ledger projection admitted by the generated neutral carrier factors through the fixed-half subcarrier.*

Definition 9250.17 (RH-neutral and fixed-half zeta carriers)**.** *Let $\mathcal{D}^{0}_{\zeta}$ be the displayed zeta constraint diagram containing the recovery rows of Definition 9254.13 and the zero-ledger row, but not the fixed-half conclusion as a defining constraint. Its compatible carrier is*

$$\mathbb{G}^{0}_{\zeta} := \mathbb{G}_{\mathcal{D}^{0}_{\zeta}}.$$

The fixed-half subcarrier is

$$\mathbb{G}^{1/2}_{\zeta} := \sum_{g:\mathbb{G}^{0}_{\zeta}} \mathsf{FixHalf}(g),$$

where $\mathsf{FixHalf}(g)$ may be read through any of the equivalent half-layer rows of Definition 9254.10.

Principle 9250.18 (RH as endogenous fixed-half constraint)**.** *RH is not the definition of $\mathbb{G}^{1/2}_{\zeta}$. It is the claim that the zero-ledger part of the RH-neutral carrier $\mathbb{G}^{0}_{\zeta}$ already lands in the fixed-half subcarrier. In constraint-diagram language, it says that the fixed-half row is not an extra defining row for zero packets, but a consequence of the other recovery, mirror, completion, and zero-ledger rows. The positive-apartness form of this consequence is the local-global obstruction isolated in Definition 9246.1.*

Proposition 9250.19 (Euler phase channels are prime-axis readouts)**.** *On the displayed right-half-plane window, the exponential kernel sends a finite-support prime occupancy row $k \in T_{\mathbb{P}}$ to the product of its prime-axis phase-scale channels:*

$$\exp\left(-s\sum_{p} k_p \log p\right) = \prod_{p}\left(p^{-\sigma}e^{-i\tau\log p}\right)^{k_p}$$

for $s = \sigma + i\tau$.

Proof. The prime-occupancy readback gives

$$\log \Psi(k) = \sum_p k_p \log p.$$

Substituting $s = \sigma + i\tau$ into the exponential kernel gives

$$e^{-s \log p} = e^{-\sigma \log p} e^{-i\tau \log p} = p^{-\sigma} e^{-i\tau \log p}.$$

Multiplying these local reads over the finite support of k yields the displayed product. □

Principle 9250.20 (Zeta geometry is recovered-carrier geometry)**.** *The geometric reading of zeta is not a point-set geometry in the complex plane. The complex coordinate s is a compressed history read through prime fingerprints, Euler phase-scale channels, the discrete Laplace kernel, the recovered-carrier continuation rows, and the fixed-half mirror. In this reading, RH is the zero-ledger section claim that recognized critical-strip zero packets of* ZetaGeom *land in the fixed-half surface, equivalently in the unit-amplitude surface of all centered prime channels.*

Principle 9250.21 (Prime rows are local probes of the recovered carrier)**.** *In the zeta self-similar packet, ordinary primes are the multiplicative-divisibility atoms, but their RH-facing role is not to be points of the critical strip. They index local probes*

$$p \mapsto p^{-s}$$

and centered channels

$$U_p(s) = p^{1/2-s}.$$

Thus zeta is an instance of the relative priming principle of Definition 9414.209: the classifier is multiplicative on the index side and prime-local on the information readout side.

Definition 9250.22 (Recovered-carrier self-dual mirror)**.** *The recovered-carrier mirror is the functional-equation readback displayed on complex histories as*

$$J(s) := 1 - \overline{s}.$$

Writing

$$s = \frac{1}{2} + \delta + i\tau$$

gives

$$J(s) = \frac{1}{2} - \delta + i\tau.$$

It preserves the prime phase clock $\tau \log p$ and reverses the prime-amplitude bias $\delta \log p$ in the centered channels, as in Definition 9248.6.

Theorem 9250.23 (RH is fixed-half zero ledger for the self-similar carrier)**.** *Under the recovered-carrier reading of Definition 9250.2, the Riemann Hypothesis is the assertion that every recognized non-trivial zero packet lies in the fixed-half surface of the mirror:*

$$\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow s = J(s).$$

Equivalently, every recognized non-trivial zero packet has no prime-visible mirror amplitude bias:

$$\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{PF}(s) = \mathsf{PF}(J(s)).$$

Proof. The first displayed implication is the fixed-surface reading of Definition 9254.22. The equivalence with prime-fingerprint collapse is Definition 9248.9. The mirror and centered prime-channel interpretation is supplied by Definition 9250.22 and Definition 9248.6. □

Principle 9250.24 (Self-similar comparison boundary)**.** *The zeta packet and the Fibonacci-Zeckendorf packet are both self-similar only after their classifiers are named. The Fibonacci route uses second-order recursive axis normalization*

$$e_k + e_{k+1} \rightsquigarrow e_{k+2}.$$

The zeta route uses prime-indexed local geometric fibers and multiplicative Euler gluing

$$E_p(s) = 1 + p^{-s} + p^{-2s} + \cdots .$$

The comparison is structural: it identifies classifier-relative self-similar recovery patterns, not a common ordinary geometric object and not a proof of the zero-ledger exclusion.

Concretized in §71: The basic zeta packet supplies the right-half-plane Dirichlet and Euler rows used by the self-similar projection.

Concretized in §580: The zeta-continuation witness packet supplies the continuation, pole, trivial-zero, and Gamma-boundary rows needed for recovered-carrier reading.

Concretized in §581: The zeta functional-equation ledger supplies the mirror transport row used by the recovered-carrier self-dual mirror.

Chapter 9251

Zeta–Euler Tick Algebra

chapter 9250 reads ZETA as a recovered carrier assembled from Dirichlet, Euler, Laplace, continuation, completion, mirror, and zero-ledger rows. The present chapter isolates the dynamical row inside that carrier. Integers are finite prime-occupancy sections, prime multiplication is a tick in the prime ledger, $\log n$ is the energy coordinate of the section, and the right-half-plane zeta projection is the partition trace of the corresponding logarithmic Hamiltonian. The chapter is an interface translation; it does not close the analytic continuation, zero coverage, or RH obligations.

9251.1 Prime Ledger and Logarithmic Hamiltonian

Definition 9251.1 (Prime-register lattice)**.** *The prime-register lattice is the finite-support abelian group*

$$G_{\mathbb{P}} := \bigoplus_{p\in\mathbb{P}} \mathbb{Z}e_p,$$

with positive cone

$$G_{\mathbb{P}}^{+} := \bigoplus_{p\in\mathbb{P}} \mathbb{N}e_p.$$

For a positive integer

$$n = \prod_p p^{v_p(n)},$$

the register row is

$$\nu(n) := \sum_p v_p(n)e_p \in G_{\mathbb{P}}^{+}.$$

The row is finite by unique prime factorization.

Definition 9251.2 (Zeta Fock carrier and Hamiltonian)**.** *The Zeta–Euler tick carrier is the Hilbert space*

$$\mathcal{H}_\zeta := \ell^2(\mathbb{N}_{\geq 1})$$

with standard basis $\{|n\rangle : n \geq 1\}$*. Equivalently, it has the finite-excitation prime-occupancy presentation*

$$\mathcal{H}_\zeta \cong \bigotimes_{p\in\mathbb{P}}^{0} \ell^2(\mathbb{N}_0), \qquad |n\rangle \longleftrightarrow \bigotimes_p |v_p(n)\rangle.$$

For each prime p, the number row is

$$N_p|n\rangle := v_p(n)|n\rangle.$$

The logarithmic Hamiltonian is

$$H_\zeta := \sum_{p\in\mathbb{P}} (\log p) N_p,$$

read on the finite-support basis. Thus

$$H_\zeta |n\rangle = (\log n)|n\rangle.$$

Theorem 9251.3 (Zeta phase flow is prime-ledger phase transport)**.** *Let*

$$U_t := e^{-itH_\zeta}.$$

Then, for every positive integer n,

$$U_t|n\rangle = n^{-it}|n\rangle = \prod_p e^{-itv_p(n)\log p}|n\rangle.$$

Proof. By Definition 9251.2,

$$H_\zeta|n\rangle = (\log n)|n\rangle.$$

Therefore

$$U_t|n\rangle = e^{-it\log n}|n\rangle = n^{-it}|n\rangle.$$

Since

$$\log n = \sum_p v_p(n)\log p,$$

the final product form follows. □

9251.2 Trace and Euler Decomposition

Definition 9251.4 (Zeta partition trace)**.** *For $s = \sigma + it$ in a displayed right-half-plane convergence window, the Zeta–Euler partition trace is*

$$Z_{\text{tick}}(s) := \operatorname{Tr}(e^{-sH_\zeta}),$$

when the trace is licensed by absolute convergence. For a finite prime window S, the finite carrier and Hamiltonian are

$$\mathcal{H}_S := \bigotimes_{p\in S} \ell^2(\mathbb{N}_0), \qquad H_S := \sum_{p\in S} (\log p) N_p,$$

with finite-window trace

$$Z_S(s) := \operatorname{Tr}_{\mathcal{H}_S}(e^{-sH_S}).$$

Theorem 9251.5 (Right-half-plane trace is the Dirichlet projection)**.** *On $\Re(s) > 1$,*

$$Z_{\text{tick}}(s) = \zeta_D(s).$$

Equivalently,

$$\operatorname{Tr}(e^{-sH_\zeta}) = \sum_{n\geq 1} e^{-s\log n} = \sum_{n\geq 1} n^{-s}.$$

Proof. The basis vector $|n\rangle$ is an eigenvector of H_ζ with eigenvalue $\log n$. Hence

$$e^{-sH_\zeta}|n\rangle = e^{-s\log n}|n\rangle = n^{-s}|n\rangle.$$

On $\Re(s) > 1$ the diagonal series is absolutely summable, so the trace is the displayed sum. This is the Dirichlet projection of Definition 9243.2. □

Theorem 9251.6 (Finite-window Euler trace)**.** *For a finite prime window S and $\Re(s) > 0$,*

$$Z_S(s) = \prod_{p \in S} \frac{1}{1 - p^{-s}}.$$

As the prime windows cofinally exhaust $\mathbb{P}$, this agrees with the Euler projection on $\Re(s) > 1$.

Proof. On the p-factor, the number row has eigenvalues $k \geq 0$, so

$$\mathrm{Tr}(e^{-s(\log p)N_p}) = \sum_{k \geq 0} p^{-ks} = \frac{1}{1 - p^{-s}}$$

for $\Re(s) > 0$. The finite tensor product trace is the product of these one-prime traces. The cofinal window statement is the Euler gluing row of Definition 9243.4. □

Corollary 9251.7 (Energy moments of the trace)**.** *On $\Re(s) > 1$,*

$$(-\partial_s)^k \zeta_D(s) = \sum_{n \geq 1} (\log n)^k n^{-s}.$$

In particular,

$$i\partial_t \zeta_D(\sigma + it) = -\zeta_D'(\sigma + it) = \mathrm{Tr}(H_\zeta e^{-(\sigma+it)H_\zeta}).$$

Proof. Absolute convergence on the displayed window licenses termwise differentiation. Each application of $-\partial_s$ multiplies the n-term by $\log n$. The t-derivative identity follows from $s = \sigma + it$. □

9251.3 Prime Tick and Solenoid Phase

Definition 9251.8 (Prime creation tick)**.** *For a prime p, the prime creation tick is the isometry*

$$C_p : \mathcal{H}_\zeta \to \mathcal{H}_\zeta, \qquad C_p|n\rangle := |pn\rangle.$$

More generally, $C_m|n\rangle := |mn\rangle$ for $m \geq 1$. On the register row,

$$\nu(pn) = \nu(n) + e_p.$$

Definition 9251.9 (Zeta tick ledger residual)**.** *Let $\mathcal{L} \subseteq \mathcal{H}_\zeta$ be a closed ledger subspace and let $B \in \mathcal{L}$. The residual of the prime tick p is*

$$\eta_p := C_p B - P_{\mathcal{L}} C_p B,$$

where $P_{\mathcal{L}}$ is the orthogonal projection onto $\mathcal{L}$. The tick-extended ledger is

$$\mathcal{L}^+ := \mathcal{L} \oplus \overline{\mathrm{span}}\{\eta_p\}.$$

Theorem 9251.10 (Prime ticks enter the ledger orthogonally)**.** *For every closed ledger $\mathcal{L}$ and every prime tick p,*

$$\eta_p \perp \mathcal{L}.$$

Proof. This is the Hilbert projection theorem applied to $C_p B$ and the closed subspace $\mathcal{L}$. The vector

$$C_p B - P_{\mathcal{L}} C_p B$$

is orthogonal to $\mathcal{L}$, and this vector is η_p by Definition 9251.9. □

Definition 9251.11 (Finite-window Zeta solenoid phase row)**.** *For a finite prime window S, the Zeta solenoid phase row is the one-parameter phase family on the S-prime torus with frequency vector*

$$\omega_p^\zeta := -\log p \qquad (p \in S).$$

For an integer supported in S,

$$n = \prod_{p \in S} p^{v_p(n)},$$

the associated character row is

$$\chi_n := \prod_{p \in S} \chi_p^{v_p(n)}.$$

Theorem 9251.12 (Solenoid phase reads the Zeta phase)**.** *Along the finite-window Zeta phase row,*

$$\chi_n(\tau_{\zeta,S}(t)) = n^{-it}.$$

Proof. By the frequency definition,

$$\chi_p(\tau_{\zeta,S}(t)) = e^{-it \log p}.$$

Multiplying over the finite support of n gives

$$\chi_n(\tau_{\zeta,S}(t)) = \prod_{p \in S} e^{-itv_p(n) \log p} = e^{-it \log n} = n^{-it}.$$

□

Principle 9251.13 (Zeta phase is not a bare circle readout)**.** *The visible row $n^{-it} \in \mathbb{T}$ is only the circle projection of the prime-ledger phase. The solenoid row of Definition 9289.4 permits hidden compact prime-register fibres. It does not permit an independent positive real radial coordinate to be hidden as phase fibre.*

9251.4 Zero Closure and Li Radial Residual

Definition 9251.14 (Li radial residual)**.** *For a nonzero zero-coordinate ρ, its Li coordinate is*

$$z_\rho := 1 - \frac{1}{\rho}.$$

Writing

$$z_\rho = e^{\ell_\rho + i\theta_\rho},$$

the real number

$$\ell_\rho := \log |z_\rho|$$

is the Li radial residual.

Theorem 9251.15 (Li radial residual is the fixed-half test)**.** *For $\rho = \beta + i\gamma$ with $\rho \neq 0$,*

$$\ell_\rho = 0 \quad \Longleftrightarrow \quad \beta = \frac{1}{2}.$$

Proof. The equality $\ell_\rho = 0$ is equivalent to $|z_\rho| = 1$. Since

$$z_\rho = \frac{\rho - 1}{\rho},$$

this is equivalent to

$$|\rho - 1| = |\rho|.$$

Writing $\rho = \beta + i\gamma$, the squared equality is

$$(\beta - 1)^2 + \gamma^2 = \beta^2 + \gamma^2,$$

which reduces to $\beta = 1/2$. □

Principle 9251.16 (Zeta–Euler tick algebra proof boundary)**.** *The Zeta–Euler tick algebra supplies the following translation: prime-register ticks generate finite-support integer sections, the logarithmic Hamiltonian reads their energy, the right-half-plane trace recovers the Dirichlet projection, the finite-window trace recovers the Euler product, and the Li radial residual measures departure from the fixed-half surface. The remaining RH-facing source statement is not any of these identities. It is the exclusion of a nonzero archimedean radial residual from every accepted completed zero packet, in the presence of the continuation, completion, solenoid, and zero-coverage ledgers.*

Chapter 9252

Information-Channel Normal Form of RH

chapter 9243 identifies the critical line with the unit-amplitude surface of the centered prime-local channels. The present chapter rewrites the same fact in information-channel coordinates. It does not introduce another conjecture and it does not prove RH. It replaces the geometric phrase "line $\Re(s) = 1/2$" by the coordinate-free certificate condition that every non-trivial prime-source cancellation event has zero amplitude bias.

9252.1 Information coordinates

Definition 9252.1 (Bias-clock coordinates)**.** *For a complex history s, write*

$$s = \frac{1}{2} + \delta + i\tau.$$

The real parameter δ is the amplitude-bias coordinate, and the real parameter τ is the phase-clock coordinate. The critical-line predicate is then

$$\mathsf{OnCritLine}(s) \quad \Longleftrightarrow \quad \delta\mathsf{hsame0}.$$

Definition 9252.2 (Prime information source)**.** *The prime information source is the displayed atomic source $\mathcal{P}$ whose atoms are the prime packets carried by $\mathsf{Prime}^{\uparrow}$. A positive integer packet is read as a finite product of these atoms:*

$$n = \prod_p p^{k_p}.$$

Thus the prime source is not an auxiliary interpretation of zeta; it is the local coordinate system already exposed by the Euler-product row.

Definition 9252.3 (Centered information channel)**.** *For $p \in \mathsf{Prime}^{\uparrow}$ and $s = 1/2 + \delta + i\tau$, define the centered prime information channel*

$$U_p(\delta, \tau) = p^{1/2-s} = p^{-\delta-i\tau}.$$

Equivalently,

$$U_p(\delta, \tau) = p^{-\delta} e^{-i\tau \log p}.$$

Its amplitude row is

$$\mathsf{Amp}_p(\delta, \tau) = |U_p(\delta, \tau)| = p^{-\delta}.$$

Proposition 9252.4 (Zero bias is unit channel amplitude)**.** *For any displayed prime row $p > 1$,*

$$|U_p(\delta, \tau)| = 1 \quad \Longleftrightarrow \quad \delta = 0.$$

Consequently,

$$\delta = 0 \quad \Longleftrightarrow \quad \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(\delta, \tau)| = 1.$$

Proof. By Definition 9252.3,

$$|U_p(\delta, \tau)| = p^{-\delta}.$$

Since $p > 1$, this amplitude is one exactly when $-\delta = 0$, equivalently $\delta = 0$. The universal statement follows because every displayed prime row has the same bias coordinate. □

Corollary 9252.5 (Amplitude bias is prime-source tilt)**.** *If $\delta > 0$, every sufficiently large prime channel is damped by the factor $p^{-\delta} < 1$. If $\delta < 0$, every sufficiently large prime channel is amplified by the factor $p^{-\delta} > 1$. Thus $\delta \neq 0$ is exactly the presence of a coherent prime-source amplitude tilt.*

Proof. This is the sign reading of $|U_p(\delta, \tau)| = p^{-\delta}$. Positive bias places the exponent below zero, while negative bias places the exponent above zero. □

9252.2 Prime carrier and cancellation events

Definition 9252.6 (Prime carrier)**.** *The prime carrier at information parameter (δ, τ) is the analytic recovery packet*

$$\mathsf{PrimeCarrier}(\delta, \tau) = \mathrm{AnaRec}\left(\prod_{p \in \mathsf{Prime}^{\uparrow}} (1 - p^{-1/2-\delta-i\tau})^{-1}\right).$$

Equivalently, on the Dirichlet projection window, it is recovered from

$$\sum_{n \geq 1} n^{-1/2-\delta-i\tau}.$$

This is the same ZETA *recovery carrier read through the information coordinates of Definition 9252.1.*

Definition 9252.7 (Prime-source cancellation)**.** *The prime-source cancellation predicate is*

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) :\Longleftrightarrow \mathsf{PrimeCarrier}(\delta, \tau)\mathsf{hsame0}_{\mathbb{C}}.$$

It is the information-channel reading of a zeta-zero row: the prime-source carrier has a global coherent cancellation at the displayed parameter.

Definition 9252.8 (Information band)**.** *The non-trivial information band is*

$$\mathsf{InfoBand}(\delta) :\Longleftrightarrow -\frac{1}{2} < \delta < \frac{1}{2}.$$

Under $s = 1/2 + \delta + i\tau$, this is exactly the critical-strip condition $0 < \Re(s) < 1$.

Definition 9252.9 (Information RH)**.** *The information-channel normal form of the Riemann Hypothesis is*

$$\mathsf{InfoRH}_{\mathcal{P}} :\Longleftrightarrow \forall(\delta,\tau),\ \mathsf{Cancel}_{\mathcal{P}}(\delta,\tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \delta = 0.$$

Equivalently,

$$\forall(\delta,\tau),\ \mathsf{Cancel}_{\mathcal{P}}(\delta,\tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(\delta,\tau)| = 1.$$

Proposition 9252.10 (Information RH is BEDC-RH)**.** *The proposition* $\mathsf{InfoRH}_{\mathcal{P}}$ *is equivalent to the BEDC statement* RH *of Definition 9239.7. This is a scoped coordinate readout through the substitution* $s = 1/2 + \delta + i\tau$ *and the same zero row, not a separate cancellation mechanism.*

Proof. Substitute $s = 1/2 + \delta + i\tau$. Then

$$\mathsf{InfoBand}(\delta) \quad\Longleftrightarrow\quad \mathsf{InCritStrip}(s),$$

and

$$\delta = 0 \quad\Longleftrightarrow\quad \mathsf{OnCritLine}(s).$$

The cancellation predicate is the same zero row as $\mathsf{ZetaZero}(s)$, read through the recovered carrier of Definition 9252.6. Therefore

$$\mathsf{Cancel}_{\mathcal{P}}(\delta,\tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \delta = 0$$

is exactly

$$\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{OnCritLine}(s),$$

which is the BEDC-RH predicate. □

9252.3 Dual return

Definition 9252.11 (Dual-return map)**.** *The information-channel dual return is*

$$\mathcal{J}(\delta,\tau) = (-\delta,\tau).$$

Under the coordinate relation $s = 1/2 + \delta + i\tau$*, this is the mirror map* $J(s) = 1 - \overline{s}$ *of Definition 9241.12.*

Proposition 9252.12 (Dual-return fixed packets are zero-bias packets)**.** *For every information parameter* (δ,τ)*,*

$$(\delta,\tau) = \mathcal{J}(\delta,\tau) \quad\Longleftrightarrow\quad \delta = 0.$$

Proof. The equality $(\delta,\tau) = (-\delta,\tau)$ is equivalent to $\delta = -\delta$, hence to $\delta = 0$. □

Corollary 9252.13 (Information fixed-point form of RH)**.** *The BEDC statement* RH *is equivalent to*

$$\forall(\delta,\tau),\ \mathsf{Cancel}_{\mathcal{P}}(\delta,\tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow (\delta,\tau) = \mathcal{J}(\delta,\tau).$$

Proof. Combine Definition 9252.10 with Definition 9252.12. The conclusion $\delta = 0$ is exactly the dual-return fixed-packet condition. □

Theorem 9252.14 (Information-channel normal form)**.** *The following are equivalent readings of the same RH target:*

$$\mathsf{ZetaZero}(s) \wedge \mathsf{InCritStrip}(s) \Longrightarrow \mathsf{OnCritLine}(s),$$

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \delta = 0,$$

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(\delta, \tau)| = 1,$$

and

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow (\delta, \tau) = \mathcal{J}(\delta, \tau).$$

Proof. The first and second forms are equivalent by the coordinate substitution $s = 1/2 + \delta + i\tau$ used in Definition 9252.10. The second and third forms are equivalent by Definition 9252.4. The second and fourth forms are equivalent by Definition 9252.12. □

Remark 9252.15 (Information reading)**.** *In this normal form, a non-trivial zero is read as a global cancellation event of the recovered prime-source carrier. The coordinate δ records the coherent amplitude bias of every centered prime channel, while τ records the phase clock. The functional-equation mirror changes the sign of δ and preserves τ. The RH target states that every non-trivial cancellation event is fixed by this dual return, equivalently that it occurs with zero amplitude bias.*

Concretized in §64: The prime source $\mathcal{P}$ is realized by the finite $\mathsf{Prime}^{\uparrow}$ NameCert surface and its factorization rows.

Concretized in §71: The right-half-plane prime carrier projection is realized by the $\mathsf{ZetaBasic}^{\uparrow}$ Dirichlet and Euler-product rows.

Concretized in §580: The analytic recovery of $\mathsf{PrimeCarrier}$ is routed through the zeta-continuation witness packet.

Concretized in §581: The dual-return map is the information-coordinate reading of the zeta functional-equation mirror row.

Concretized in §550: Prime-source cancellation in the information band is consumed through the critical-strip zeta-zero witness surface.

Concretized in §548: Zero-bias output is the information-channel reading of the critical-line witness packet.

Chapter 9253

Zeta Zero Type Factorization

chapter 9250 treats zeta as a recovered carrier assembled from compatible analytic, arithmetic, completion, mirror, and zero-ledger rows. The present chapter records the type-theoretic form of its zero surface. A zeta zero is not a bare point of the complex plane. It is an inhabitant of the zero fiber of the zeta-value relation, together with the ledger rows that identify the value with the zero complex history. This chapter does not prove RH; it identifies the pullback and factorization obligations consumed by the fixed-half section form of chapter 9254.

9253.1 Zeta Zero Fiber

Definition 9253.1 (Zeta domain and zero object)**.** *The zeta domain* $\mathsf{ZetaDom}^{\uparrow}$ *is the recovered continuation domain for* $\mathsf{ZetaCont}^{\uparrow}$*, with the pole at* 1 *excluded and with its continuation, branch, pole, and provenance ledgers displayed. The zero object of* $\mathsf{Complex}^{\uparrow}$ *is the terminal row*

$$0_{\mathbb{C}} : \mathbf{1} \to \mathsf{Complex}^{\uparrow}, \qquad 0_{\mathbb{C}}(*) = e_{\mathsf{Complex}^{\uparrow}}.$$

The zeta-value readout is represented internally by the relation

$$\mathsf{ZetaVal}(s, z),$$

with $s : \mathsf{ZetaDom}^{\uparrow}$ *and* $z : \mathsf{Complex}^{\uparrow}$*, rather than by an unledgered host function.*

Definition 9253.2 (Zeta zero type)**.** *The zeta zero type is the pullback-like carrier*

$$\mathsf{ZetaZeroType} := \sum_{s:\mathsf{ZetaDom}^{\uparrow}} \sum_{z:\mathsf{Complex}^{\uparrow}} (\mathsf{ZetaVal}(s, z) \times z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}).$$

It is the BEDC zero fiber

$$\mathsf{ZetaDom}^{\uparrow} \times_{\mathsf{Complex}^{\uparrow}} \mathbf{1}$$

read through the available $\mathsf{ZetaVal}$ *witness relation. Its classifier identifies two inhabitants when their source rows are classified together and their displayed value witnesses agree at the zero complex history.*

Definition 9253.3 (Zero projections)**.** *The source and value projections are*

$$\pi_s : \mathsf{ZetaZeroType} \to \mathsf{ZetaDom}^{\uparrow}, \qquad \pi_z : \mathsf{ZetaZeroType} \to \mathsf{Complex}^{\uparrow},$$

with

$$\pi_s(s, z, v, \epsilon) = s, \qquad \pi_z(s, z, v, \epsilon) = z.$$

They are projections from a witness-bearing fiber, not forgetful maps from a host-side zero set.

Theorem 9253.4 (Zeta zero type is the zero fiber)**.** *The carrier* $\mathsf{ZetaZeroType}$ *is equivalent, in the licensed classifier sense, to the zero fiber of the zeta-value readout.*

Proof. The zero fiber of the zeta-value readout is the dependent carrier of source rows s whose zeta value is classified with the zero complex history. In BEDC this assertion is represented by a displayed value row $z : \mathsf{Complex}^{\uparrow}$, a witness $\mathsf{ZetaVal}(s, z)$, and a classifier row $z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}$. Expanding the dependent carrier gives exactly Definition 9253.2. The classifier is the same source-and-value witness compatibility used by the zero ledger. □

9253.2 Non-Trivial and Fixed-Half Subtypes

Definition 9253.5 (Non-trivial zero type)**.** *The non-trivial zero type is the displayed subtype*

$$\mathsf{NTZeroType} := \sum_{u:\mathsf{ZetaZeroType}} \mathsf{InCritStrip}(\pi_s(u)).$$

It is the type-theoretic form of the RH source carrier $\mathcal{Z}_\zeta$ *in Definition 9254.1.*

Definition 9253.6 (Fixed-half zero type)**.** *The fixed-half zero type is*

$$\mathsf{FixedHalfZeroType} := \sum_{u:\mathsf{NTZeroType}} \mathsf{OnCritLine}(\pi_s(u)).$$

The projection

$$\iota : \mathsf{FixedHalfZeroType} \to \mathsf{NTZeroType}$$

forgets only the fixed-half witness and retains the zero and critical-strip ledgers.

Theorem 9253.7 (RH is fixed-half subtype coverage)**.** *The BEDC-RH predicate of Definition 9239.7 is equivalent to a section of*

$$\iota : \mathsf{FixedHalfZeroType} \to \mathsf{NTZeroType}.$$

Proof. An element of $\mathsf{NTZeroType}$ is a zeta-zero witness together with a critical-strip witness. A section of ι assigns to every such element an additional witness of $\mathsf{OnCritLine}$ for the same source row. This is exactly the dependent function required by Definition 9239.7. Conversely, any inhabitant of the BEDC-RH predicate sends an element of $\mathsf{NTZeroType}$ to the same element paired with its critical-line witness, producing a section. □

9253.3 Zero Generators

Definition 9253.8 (Zeta zero generator)**.** *For a licensed carrier* A*, a zeta zero generator is a licensed morphism*

$$G : A \to \mathsf{ZetaDom}^{\uparrow}$$

together with a zero-value witness family

$$\theta_G(a) : \sum_{z:\mathsf{Complex}^{\uparrow}} (\mathsf{ZetaVal}(G(a), z) \times z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}).$$

The generator is legal only when the carrier, classifier, stability, ledger, transport, and $\mathsf{NameCert}$ *rows for* A *and the displayed morphism are present.*

Definition 9253.9 (Induced zero map)**.** *The zero generator G induces*

$$\overline{G} : A \to \mathsf{ZetaZeroType}$$

by sending a to the tuple

$$(G(a), z_a, v_a, \epsilon_a)$$

read from $\theta_G(a) = (z_a, v_a, \epsilon_a)$.

Theorem 9253.10 (Zero generators factor through the zero type)**.** *Every legal zeta zero generator factors through* ZetaZeroType*, and the factorization is unique up to the zero-type classifier.*

Proof. The witness family θ_G gives exactly the data required by Definition 9253.2: a source row, a zeta-value row, and a zero classifier row. Hence it defines $\overline{G}$. If another map $H : A \to \mathsf{ZetaZeroType}$ has the same source projection and the same zero-value witness ledger, then the zero-type classifier identifies $H(a)$ with $\overline{G}(a)$ for every a. This is the classifier form of the pullback universal property. □

Corollary 9253.11 (Legal generators share the zero fiber)**.** *Any two legal zeta zero generators are maps into the same* ZetaZeroType *after applying their induced zero maps. They may present different subcarriers, but their zero status is typed through the same zero fiber.*

9253.4 Canonical Presentations and Compatibility

Definition 9253.12 (Canonical zero presentation)**.** *A canonical presentation of* ZetaZeroType *is a licensed map*

$$P : A \to \mathsf{ZetaZeroType}$$

such that:

$$a \sim_A a' \to P(a) \sim_{\mathsf{Zero}} P(a'),$$
$$\prod_{u:\mathsf{ZetaZeroType}} \sum_{a:A} P(a) \sim_{\mathsf{Zero}} u,$$

and

$$P(a) \sim_{\mathsf{Zero}} P(a') \to a \sim_A a'.$$

Thus P is an equivalence of licensed presentations, not merely a surjective listing of complex coordinates.

Theorem 9253.13 (Canonical zero presentations are unique up to equivalence)**.** *If $P : A \to$* ZetaZeroType *and $Q : B \to$* ZetaZeroType *are canonical presentations, then A and B are equivalent licensed presentations of the same zero type.*

Proof. By canonicity, P is full on zero-type rows and faithful to the source classifier, so it presents A as an equivalent carrier for ZetaZeroType. The same argument applies to Q and B. Composing one equivalence with the inverse of the other yields the licensed equivalence between A and B. □

Definition 9253.14 (Compatible zero generators)**.** *Two zero generators are compatible when their induced maps into* ZetaZeroType *factor through a common canonical presentation after licensed equivalence of their source carriers. They are incompatible when no such common presentation and equivalence data are supplied.*

Corollary 9253.15 (Incompatible canonical generators cannot present one zero type)**.** *Two incompatible generators cannot both be canonical presentations of the same* $\mathsf{ZetaZeroType}$.

Proof. If both were canonical presentations of the same zero type, then Definition 9253.13 would supply the common equivalence data required by Definition 9253.14. This contradicts incompatibility. □

9253.5 Zero and Apart-Zero Conflict

Definition 9253.16 (Same-source zero and apart-zero rows)**.** *A same-source zero and apart-zero conflict consists of a source row* $s : \mathsf{ZetaDom}^{\uparrow}$*, a zero witness*

$$(z, v, \epsilon) : \sum_{z:\mathsf{Complex}^{\uparrow}} (\mathsf{ZetaVal}(s, z) \times z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}),$$

and an apart-zero witness

$$\mathsf{Apart}_0(\zeta(s)).$$

Both witnesses must be read through the same zeta-value ledger for the conflict to be typed.

Theorem 9253.17 (Same-source zero and apart-zero conflict is impossible)**.** *A same-source zero and apart-zero conflict yields* $\bot$.

Proof. The zero witness supplies a value row $\mathsf{ZetaVal}(s, z)$ and a classifier row $z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}$. The apart-zero witness asserts that the same zeta value is positively separated from the zero complex history. Since the zeta-value ledger identifies fixed-source values up to the complex classifier, the same value cannot be both classified with zero and apart from zero. Hence the conflict yields $\bot$. □

Principle 9253.18 (Type-theoretic skeleton of defect-zero incompatibility)**.** *The local-global target of Definition 9246.13 has this type-theoretic skeleton: a prime-window or bulk-boundary defect route may contradict a zero packet only after it produces an apart-zero witness for the same source row and the same zeta-value ledger.*

9253.6 Omega Factorization

Definition 9253.19 (Omega non-trivial zero carrier)**.** *For an* Ω *tower with zeta projection*

$$\pi_\zeta : T_\Omega \to \mathsf{ZetaDom}^{\uparrow},$$

the Omega non-trivial zero carrier is

$$T_\Omega^{nt0} := \sum_{x:T_\Omega} (\mathsf{ZetaZero}(\pi_\zeta(x)) \times \mathsf{InCritStrip}(\pi_\zeta(x)))\,.$$

Theorem 9253.20 (Omega zero carrier factors through the non-trivial zero type)**.** *The projection* π_ζ *restricted to* T_Ω^{nt0} *factors through* $\mathsf{NTZeroType}$.

Proof. An element of T_Ω^{nt0} is a tower row x together with a zeta-zero witness and a critical-strip witness for $\pi_\zeta(x)$. Expanding Definition 9253.5, this is precisely an element of $\mathsf{NTZeroType}$ over the source $\pi_\zeta(x)$. The factorization is unique up to the displayed subtype classifier. □

Theorem 9253.21 (Fixed-half factorization is an RH sufficient interface)**.** *If the factorization of T_Ω^{nt0} through* NTZeroType *further factors through* FixedHalfZeroType*, and every legal non-trivial zeta-zero packet is represented by some row of T_Ω^{nt0}, then the BEDC-RH predicate holds.*

Proof. Let u : NTZeroType. By representation, choose a row $x : T_\Omega^{nt0}$ whose zeta projection is classified with the source of u and whose zero and critical-strip witnesses transport to those of u. Fixed-half factorization gives $\mathsf{OnCritLine}(\pi_\zeta(x))$. Classifier stability of OnCritLine transports this witness to $\pi_s(u)$. Since this holds for every u : NTZeroType, Definition 9253.7 yields the BEDC-RH predicate. □

Principle 9253.22 (Status of zero-type factorization)**.** *Zero-type factorization clarifies where uniqueness and incompatibility live: legal zero generators factor through the same zero fiber, canonical presentations are unique up to licensed equivalence, and a contradiction requires same-source zero and apart-zero witnesses. The remaining RH content is the construction of the fixed-half factorization, not the existence of the zero type itself.*

Concretized in §8772: Finite zero-fiber, generator-factorization, and fixed-half ledger rows are realized as a concrete NameCert packet.

Chapter 9254

Layered RH Translation

chapter 9239 states the Riemann Hypothesis as a constructive function from critical-strip zeta-zero witnesses to $\mathsf{OnCritLine}$ witnesses. The prime-local and information-channel chapters then rewrite the same target through centered prime amplitudes and zero-bias cancellation. The present chapter records the common layered translation principle: each formulation is a readout of the same fixed-half fiber over the source carrier of non-trivial zeta-zero packets. It does not prove RH and does not introduce an additional conjecture.

9254.1 Source Carrier and Half Fiber

Definition 9254.1 (RH source carrier)**.** *The RH source carrier is the displayed sigma carrier*

$$\mathcal{Z}_\zeta := \sum_s (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)).$$

It is the BEDC-facing zero packet carrier already used by Definition 9239.7; it is not a host-side zero set and not an ambient quotient of all complex roots.

Definition 9254.2 (Real half layer)**.** *The real half layer over $\mathcal{Z}_\zeta$ is*

$$\mathcal{H}_\mathbb{R} = (\mathsf{Real}^\uparrow,\ b_\mathbb{R},\ 1/2), \qquad b_\mathbb{R}(s) := \Re(s).$$

Its fixed half fiber is

$$\mathrm{Fix}_\mathbb{R} := \sum_{z:\mathcal{Z}_\zeta} (b_\mathbb{R}(z) = 1/2).$$

Definition 9254.3 (RH section form)**.** *The section form of RH is the type*

$$\mathsf{Sec}_\mathsf{RH} := \sum_{\sigma:\mathcal{Z}_\zeta \to \mathrm{Fix}_\mathbb{R}} (\pi_\mathbb{R} \circ \sigma = \mathrm{id}_{\mathcal{Z}_\zeta}),$$

where $\pi_\mathbb{R}$ is the projection from the fixed half fiber to the zero packet carrier.

Proposition 9254.4 (RH is a fixed-half section)**.** *The BEDC statement RH of Definition 9239.7 is equivalent to Sec_RH.*

Proof. By Definition 9239.7, RH is the dependent product assigning to every $z : \mathcal{Z}_\zeta$ a witness that $b_\mathbb{R}(z) = 1/2$. The section type records exactly the same data: from such a product one sends z to (z, p) in the fixed half fiber; from a section one reads the second component of $\sigma(z)$ and transports it along the section equation. This is the specialization of Definition 9414.159. □

9254.2 Equivalent Readout Layers

Definition 9254.5 (Complex mirror half layer)**.** *The complex mirror half layer reads a zero packet through the mirror fixed-surface predicate*

$$J(s) = s, \qquad J(s) := 1 - \overline{s}.$$

By Definition 9414.163, this predicate is equivalent to $b_{\mathbb{R}}(s) = 1/2$.

Definition 9254.6 (Prime-unitary half layer)**.** *The prime-unitary half layer reads a zero packet through the centered prime-channel predicate*

$$\forall p \in \mathsf{Prime}^{\uparrow},\ |p^{1/2}p^{-s}|\mathsf{hsame1}.$$

By Definition 9243.7, this predicate is equivalent to $\mathsf{OnCritLine}(s)$ *on the displayed prime packet.*

Definition 9254.7 (Information half layer)**.** *The information half layer writes*

$$s = 1/2 + \delta + i\tau$$

and reads the half condition as

$$\delta = 0.$$

By Definition 9252.14, this is the information-channel normal form of the same RH target.

Definition 9254.8 (Laplace half layer)**.** *The Laplace half layer reads a zero*

$$\rho = 1/2 + \delta + i\gamma$$

through the explicit-formula growth factor

$$e^{\rho x} = e^{x/2}e^{\delta x}e^{i\gamma x}.$$

The fixed half surface is the absence of the bias factor $e^{\delta x}$*, equivalently* $\delta = 0$.

Definition 9254.9 (BEDC witness half layer)**.** *The BEDC witness half layer reads a zero packet as the output type*

$$\mathsf{OnCritLine}(s).$$

This is the constructive witness surface of Definition 9239.8: an inhabitant is a finite witness identifying $\Re(s)$ *with* $1/2$ *inside the relevant* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *certificate rows.*

Theorem 9254.10 (Layered RH translations)**.** *The following readout forms are equivalent section targets over* $\mathcal{Z}_{\zeta}$*:*

$$\Re(s) = 1/2, \qquad J(s) = s, \qquad \forall p \in \mathsf{Prime}^{\uparrow},\ |p^{1/2}p^{-s}|\mathsf{hsame1},$$

$$\delta = 0, \qquad \textit{no } e^{\delta x} \textit{ Laplace bias}, \qquad \mathsf{OnCritLine}(s).$$

Proof. The first and second forms are equivalent by Definition 9414.163. The first and third forms are equivalent by Definition 9243.7. The first and fourth forms are the coordinate substitution of Definition 9252.1. The fourth and fifth forms are the same equality read through $e^{\rho x} = e^{x/2}e^{\delta x}e^{i\gamma x}$. The first and sixth forms are the definition of the constructive output in Definition 9239.7. All translations preserve the same source carrier $\mathcal{Z}_{\zeta}$ and therefore yield equivalent section targets. □

9254.3 Zeta Recovery and Laplace Readout

Definition 9254.11 (Discrete Laplace source)**.** *For a sequence (a_n), the discrete Laplace source is the atomic measure row*

$$\mu_a := \sum_{n\geq 1} a_n \delta_{\log n}.$$

Its Laplace readout is

$$\mathcal{L}[\mu_a](s) := \int_0^\infty e^{-st}\, d\mu_a(t).$$

Proposition 9254.12 (Dirichlet series as discrete Laplace readout)**.** *On any displayed convergence window,*

$$\sum_{n\geq 1} a_n n^{-s} = \mathcal{L}[\mu_a](s).$$

Proof. By the definition of μ_a,

$$\mathcal{L}[\mu_a](s) = \sum_{n\geq 1} a_n e^{-s\log n}.$$

Since $e^{-s\log n} = n^{-s}$ on the displayed branch and convergence window, this is the Dirichlet series. □

Definition 9254.13 (Zeta recovery layer)**.** *The zeta recovery layer is the carrier recovered from the compatible rows*

$$\mathsf{DirichletProjection},\quad \mathsf{EulerProjection},\quad \mathsf{ZetaCont}^{\uparrow},\quad \mathsf{Gamma}^{\uparrow}\text{-}\textit{completion},\quad \mathsf{FunctionalEquation},$$

$$\mathsf{PoleLedger},\quad \mathsf{ZeroLedger},\quad \mathsf{BranchTransport}.$$

This is the recovered ZETA *carrier of Definition 9243.2; no single projection row exhausts it.*

Principle 9254.14 (RH translation does not add analytic supply)**.** *The layered RH translation does not supply analytic continuation, Euler product convergence, Gamma factors, zero counting, or an RH proof. It only records that, once the relevant zeta-zero source packet is admitted, the real, complex-mirror, prime-unitary, information, Laplace, and BEDC witness surfaces are readouts of the same fixed half fiber.*

9254.4 Exception Transport and No-Exception Form

Definition 9254.15 (RH layer exception)**.** *For a half layer $\mathcal{H} = (B, b, h)$ over $\mathcal{Z}_\zeta$, a layer exception is the dependent sum*

$$\mathsf{Exc}_{\mathcal{H}} := \sum_{z:\mathcal{Z}_\zeta} \neg(b(z) = h).$$

It is a zero packet together with a failure of the half-value predicate at that readout layer.

Definition 9254.16 (Source fixed exception)**.** *Let $\tau : \mathcal{Z}_\zeta \to \mathcal{Z}_\zeta$ be the source mirror induced by the zeta functional-equation symmetry. A source fixed exception is*

$$\mathsf{Exc}_{\mathrm{src}} := \sum_{z:\mathcal{Z}_\zeta} \neg(\tau(z) = z).$$

This is the source-level non-fixed packet; it is not yet a failure at any particular real, prime, information, Laplace, or witness readout.

Definition 9254.17 (Half-preserving equivalence)**.** *A half layer $\mathcal{H} = (B, b, h)$ is half-preserving with respect to τ when there is a displayed family of equivalences*

$$\epsilon_{\mathcal{H}}(z) : (\tau(z) = z) \simeq (b(z) = h)$$

for every $z : \mathcal{Z}_\zeta$.

Theorem 9254.18 (Exception transport)**.** *If $\mathcal{H}$ is half-preserving with respect to τ, then*

$$\mathsf{Exc}_{\mathcal{H}} \simeq \mathsf{Exc}_{\mathrm{src}}.$$

Proof. For each z, the equivalence $\epsilon_{\mathcal{H}}(z)$ transports witnesses of $\tau(z) = z$ to witnesses of $b(z) = h$ and conversely. Contraposition along the two maps gives an equivalence between $\neg(b(z) = h)$ and $\neg(\tau(z) = z)$. Dependent-sum formation over the same source carrier $\mathcal{Z}_\zeta$ then yields the displayed equivalence of exception types. □

Theorem 9254.19 (No-exception transfer)**.** *If $\mathcal{H}$ is half-preserving with respect to τ, then*

$$\neg\mathsf{Exc}_{\mathrm{src}} \to \neg\mathsf{Exc}_{\mathcal{H}}.$$

Proof. Compose a hypothetical element of $\mathsf{Exc}_{\mathcal{H}}$ with the transport map of Definition 9254.18. The result is an element of $\mathsf{Exc}_{\mathrm{src}}$, contradicting the premise. □

Theorem 9254.20 (Symmetric closure is not no-exception)**.** *Closure of $\mathcal{Z}_\zeta$ under an involution τ does not by itself imply*

$$\neg\mathsf{Exc}_{\mathrm{src}}.$$

Proof. Consider a two-row carrier $\{a, b\}$ with $\tau(a) = b$ and $\tau(b) = a$. This carrier is closed under the involution and τ is an involution. Nevertheless neither row is fixed. Hence the sigma type $\sum_z \neg(\tau(z) = z)$ is inhabited. The example separates symmetry of the carrier from fixedness of every source row. □

Corollary 9254.21 (Precise role of contradiction)**.** *A contradiction argument can eliminate a layer exception only after that exception has been transported to a source fixed exception and the source fixed exception has been ruled out.*

Proof. The first requirement is Definition 9254.18. The second requirement is the premise of Definition 9254.19. Without the displayed half-preserving equivalence, a contradiction at one readout layer has no typed route to the source fixed predicate; without the source no-exception premise, the transported exception is not eliminated. □

9254.5 Synthesis

Theorem 9254.22 (Layered RH synthesis)**.** *Inside the RH vision stack, the Riemann Hypothesis has the unified section form*

$$\mathcal{Z}_\zeta \to \sum_{z:\mathcal{Z}_\zeta} (b(z) = h),$$

where (B, b, h) may be any of the equivalent half layers of Definition 9254.10. The only unshown content between two such layers is the corresponding fiber data over the same source carrier, not a separate object domain.

Proof. Definition 9254.4 identifies BEDC-RH with a section into the real fixed half fiber. Definition 9254.10 transports that section target across the complex-mirror, prime-unitary, information, Laplace, and witness readouts. The final statement about unshown content is the RH specialization of Definition 9414.142 and Definition 9414.45. □

Theorem 9254.23 (Layered RH no-exception form)**.** *For the real, complex-mirror, prime-unitary, information, Laplace, and BEDC witness layers of Definition 9254.10, the corresponding RH assertions are no-exception claims over the same source fixed-half fiber. When the half-preserving equivalence with the source mirror is displayed, the no-exception form is*

$$\neg\mathsf{Exc}_{\mathrm{src}} \to \prod_{\mathcal{H}} \neg\mathsf{Exc}_{\mathcal{H}}.$$

Carrier symmetry alone does not supply the premise $\neg\mathsf{Exc}_{\mathrm{src}}$*.*

Proof. Definition 9254.10 identifies the listed readout layers as equivalent half layers over $\mathcal{Z}_\zeta$. For each half-preserving layer, Definition 9254.19 transports source no-exception to layer no-exception. The final sentence is exactly Definition 9254.20. □

Concretized in §548: Critical-line witness packets realize the BEDC witness half layer as finite zero-packet, real-part, rational-half, hsame, transport, replay, and NameCert rows.

Concretized in §581: Functional-equation ledgers realize the mirror-readout row used by the layered RH translation.

Concretized in §580: Zeta-continuation witness packets realize the recovered zeta carrier consumed by the layered RH translation.

Chapter 9255

RH-Free Zero Generation by Initiality

chapter 9250 displays the recovered zeta carrier as a diagram of prime, Dirichlet, Euler, Laplace, continuation, completion, mirror, zero-ledger, window, code, certificate, and provenance rows. chapter 9254 identifies the Riemann Hypothesis with a section into the fixed-half fiber over the critical-strip zero-packet carrier. The present chapter isolates the initial-algebra pattern that would turn a zero-generation theorem into that section. It supplies no analytic continuation, no zero source, and no fixed-half witness by itself; it only separates the exact obligations.

Definition 9255.1 (RH-free zero signature). *Lean4 (definition)* `BEDC.Derived.RHRoute.ZeroGenerationInitiality.` *Lean4✓* `BEDC.Derived.RHRoute.ZeroGenerationInitiality.rhFreeZeroSignatureKinds no rh circ`

An RH-free zero signature is a displayed endofunctor F_Z on the zeta zero-presentation stratum together with a circularity audit

$$\mathsf{NoRHCirc}(F_Z).$$

The audit asserts that no constructor of F_Z consumes an $\mathsf{OnCritLine}$ *witness, a row asserting $\Re(s) = 1/2$, a section of the fixed-half inclusion, the proposition* RH*, or an arbitrary unfactored seed*

$$\mathcal{Z}_\zeta \to F_Z(X).$$

The admissible inputs are pre-RH rows such as prime-local spectral atoms, functional-equation transport, conjugation transport, classifier transport, analytic gluing, compatible limit seals, and ledger reconstruction. The word "pre-RH" means that the row is present in the neutral zeta carrier before the fixed-half consequence is imposed.

Definition 9255.2 (Generated zero carrier). *Lean4 (definition)* `BEDC.Derived.RHRoute.ZeroGenerationInitiality.` *Lean4✓* `BEDC.Derived.RHRoute.ZeroGenerationInitiality.generatedZero initiality`

For an RH-free zero signature F_Z, a generated zero carrier is an initial algebra

$$(Z, \mathsf{in}_Z) := \mu F_Z.$$

Its intended realization into the source carrier of Definition 9254.1 is a displayed map

$$\varepsilon : Z \to \mathcal{Z}_\zeta.$$

Zero-presentation exactness is the conjunction of soundness, faithfulness, and essential coverage:

$$\prod_{z:Z} \varepsilon(z) : \mathcal{Z}_\zeta,$$

$$\varepsilon(z) \sim \varepsilon(z') \to z \sim_Z z',$$

and

$$\prod_{u:\mathcal{Z}_\zeta} \sum_{z:Z} \varepsilon(z) \sim u.$$

When these rows are displayed, ε is written $Z \simeq \mathcal{Z}_\zeta$. This equivalence is not supplied by initiality; it is the completeness obligation saying that the displayed generators account for every accepted critical-strip zero packet.

Definition 9255.3 (Generated fixed-half subcarrier)**.** *Lean4 (definition)* `BEDC.Derived.RHRoute.ZeroGenerationIni`

Given $\varepsilon : Z \to \mathcal{Z}_\zeta$, define the generated fixed-half predicate

$$P_{1/2}(z) := \mathsf{OnCritLine}(\pi_s(\varepsilon(z))).$$

The fixed-half subcarrier of Z is

$$H_Z := \sum_{z:Z} P_{1/2}(z),$$

with projection $m : H_Z \to Z$.

Definition 9255.4 (Fixed-half closed zero signature)**.** *Lean4 (definition)* `BEDC.Derived.RHRoute.ZeroGenerationIni`

The RH-free zero signature F_Z is fixed-half closed relative to ε when H_Z carries an F_Z-algebra

$$h : F_Z(H_Z) \to H_Z$$

whose projection agrees with the original zero algebra:

$$m \circ h \sim \mathsf{in}_Z \circ F_Z(m).$$

Thus every displayed zero constructor, when fed fixed-half inputs, emits a fixed-half output. This is a constructor-by-constructor obligation, not a pointwise enumeration of zeros.

Theorem 9255.5 (Conditional initiality route to the RH section)**.** *Lean4 ✓* `BEDC.Derived.RHRoute.ZeroGeneratio` `conditional initiality route section`

Assume F_Z is RH-free, $(Z, \mathsf{in}_Z) = \mu F_Z$ is displayed, $\varepsilon : Z \simeq \mathcal{Z}_\zeta$ is zero-presentation exact, and F_Z is fixed-half closed relative to ε. Then the section form $\mathsf{Sec}_{\mathsf{RH}}$ of Definition 9254.3 is inhabited.

Proof. By Definition 9255.4, H_Z is an F_Z-algebra. Initiality of Z gives the unique algebra morphism

$$r : Z \to H_Z.$$

The composite $m \circ r : Z \to Z$ is an F_Z-algebra endomorphism of the initial algebra. The identity map on Z is also such an endomorphism, so initiality gives

$$m \circ r \sim \mathrm{id}_Z.$$

For each $z : Z$, the second component of $r(z) : H_Z$ is a witness of $P_{1/2}(z)$. Transporting this witness along $\varepsilon : Z \simeq \mathcal{Z}_\zeta$ yields a map from every source zero packet $u : \mathcal{Z}_\zeta$ to the fixed-half fiber over u. This is exactly a section of the projection $\mathrm{Fix}_{\mathbb{R}} \to \mathcal{Z}_\zeta$, hence an inhabitant of $\mathsf{Sec}_{\mathsf{RH}}$. □

Corollary 9255.6 (Conditional extraction of the BEDC RH witness)**.** *Under the hypotheses of Definition 9255.5, the constructive target of Definition 9239.8 is inhabited.*

Proof. Definition 9255.5 supplies $\mathsf{Sec}_{\mathsf{RH}}$. By Definition 9254.4, this is the scoped fixed-half section readback used by the BEDC statement RH of Definition 9239.7, whose proof content is the function described in Definition 9239.8. □

Definition 9255.7 (Weak eta capture packet)**.** *Let ρ be a non-trivial zeta zero in the open critical strip. A* weak eta capture packet *for ρ is a tuple*

$$(D, N, F_N, T_N, \gamma_D)$$

where D is a closed disk with ρ in its interior and $\overline{D} \subset \{0 < \Re(s) < 1\}$, the functions

$$F_N(s) := \sum_{n=1}^{N} (-1)^{n-1} n^{-s}, \qquad T_N(s) := \eta(s) - F_N(s)$$

satisfy

$$\eta(s) = F_N(s) + T_N(s),$$

and the boundary gap row is

$$\gamma_D : \prod_{s \in \partial D} \big(|T_N(s)| < |F_N(s)|\big).$$

The packet captures the local zero count of η by finite eta data and a tail ledger. It is not, by itself, a fixed-half or unit-boundary packet.

Theorem 9255.8 (Weak eta capture gives local zero-count capture)**.** *For every non-trivial zeta zero ρ in the open critical strip there exists a weak eta capture packet whose finite part F_N has the same number of zeros as η inside the disk D, counted with multiplicity.*

Proof. In the open critical strip,

$$\eta(s) = (1 - 2^{1-s})\zeta(s),$$

and the factor $1 - 2^{1-s}$ has no zero there. Hence ρ is a zero of η. Choose D so that its boundary has no η-zero and $\overline{D}$ stays in the open strip. Then

$$M := \min_{s \in \partial D} |\eta(s)|$$

is positive. The eta expansion converges uniformly on $\overline{D}$, so for some N the tail satisfies

$$\sup_{s \in \partial D} |T_N(s)| < M/2.$$

For $s \in \partial D$,

$$|F_N(s)| \geq |\eta(s)| - |T_N(s)| > M/2,$$

and therefore $|T_N(s)| < |F_N(s)|$ on ∂D. Rouché's theorem gives the same zero count for F_N and η inside D. □

Principle 9255.9 (Weak capture is not zero-presentation exactness)**.** *The weak eta capture packet is a local analytic capture row. It supplies finite eta data, a tail ledger, a disk, and a Rouché gap around a given zero. It does not supply the essential coverage row of Definition 9255.2. In particular, it does not build a canonical generated carrier Z, a global realization $\varepsilon : Z \to \mathcal{Z}_\zeta$, or the equivalence $Z \simeq \mathcal{Z}_\zeta$. Thus weak capture may be used as evidence for a local presentation mechanism, but it is not zero-presentation exactness.*

Principle 9255.10 (Boundary stability is the strong packet gap)**.** *The fixed-half closure hypothesis of Definition 9255.4 requires more than local zero-count capture. A strong zero packet must carry enough boundary data to show that the generated channel has no nontrivial real positive scale defect, equivalently that the active centered channels are unit-boundary rows:*

$$|e^{-(s-1/2)h_a}| = 1$$

for every active height $h_a > 0$ in the packet. Weak eta capture records a Rouché gap for a finite analytic approximation, but it does not prove this unit-boundary stability. The missing row is therefore a boundary-stability theorem, not a consequence of Rouché capture alone.

Principle 9255.11 (Conservative oscillator reading of the gap)**.** *The eta packet admits a physical reading as a many-oscillator phase sum. For*

$$s = \frac{1}{2} + r + it,$$

the n-th centered eta term is

$$(-1)^{n-1} n^{-1/2} e^{-r \log n} e^{-it \log n}.$$

The factor $e^{-it \log n}$ is a conservative phase rotation, while $e^{-r \log n}$ is a damping or gain factor. Thus a vanishing eta readout is not automatically a conservative phase closure. It may be a zero of a nonunitary amplitude-adjusted packet unless the packet carries unit-boundary stability.

Definition 9255.12 (Conservative normal-mode packet)**.** *A* conservative normal-mode packet *for a zeta zero is a strong generated packet whose finite oscillator rows display:*

canonical dominance or active capture,

unitary phase evolution,

no nontrivial real damping or gain factor,

far-end compatibility without hidden scale absorption,

and

transport to the accepted zeta-zero source row.

This is the physical reading of the strong packet required by Definition 9255.10: the zero is a normal phase cancellation of a closed conservative system, not a cancellation produced by adjusting amplitudes.

Definition 9255.13 (Galperin-style inertial unfolding)**.** *A* Galperin-style inertial unfolding *for a generated zeta-zero packet is a displayed carrier*

$$(\mathcal{M}, H, \Phi_t, \Pi)$$

with the following rows. The flow row is a single conservative phase flow

$$\Phi_t = e^{-itH}, \qquad \Phi_t^* \Phi_t = I.$$

The projection row Π reads the eta or zeta polygonal packet as a folded shadow of this unfolded orbit. The reflection row says that every folding, alternating-sign turn, functional-equation mirror, or prime-routing identification changes only phase direction inside the unfolded carrier. The unit row forbids any reflected step from changing radial scale. The crossing row reads a zero as a normalized cancellation of the projected orbit. The far-end row is a compatibility socket, not a reservoir for radial defect, and the coverage row transports the unfolded packet to the accepted zero source carrier.

Principle 9255.14 (Unique inertial time excludes radial time)**.** *In a Galperin-style inertial unfolding, the active centered channels are generated by one time parameter t through the unitary flow*

$$e^{-ith_a}.$$

For

$$s = \frac{1}{2} + r + it,$$

the corresponding analytic channel is

$$e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a}.$$

The second factor belongs to the inertial phase flow. The first factor is a real positive scale flow. If $r \neq 0$ and some active height satisfies $h_a > 0$, this supplies a second nonunitary time direction, not a reflection of the unitary unfolding. Thus the unique-inertial-time row can hold only when the radial parameter vanishes on every active channel.

Principle 9255.15 (ZetaGalperinInertialUnfolding target)**.** *The target*

$$\mathsf{ZetaGalperinInertialUnfolding}$$

is the combined strong form of the conservative-packet route. It requires canonical active capture, unit-boundary stability, no far-end radial absorption, and zero-presentation coverage. In this formulation, Definition 9255.13 supplies the inertial carrier, Definition 9255.14 supplies the exclusion of a scale-time component, and Definition 9255.2 supplies the coverage interface. The statement is a proof target: without these rows, a vanishing projected sum is still only a cancellation readout, not a certified conservative phase crossing.

Definition 9255.16 (Zeta cosmos closure row)**.** *The* zeta cosmos closure row *is the source-internal form of the strong route. It asserts that the zero, phase, tail, boundary, prime ledger, and generation-tower rows of the zeta sector are generated inside one BEDC triadic carrier:*

$$\mathsf{ZetaCosmosClosure}_\zeta.$$

This row has four visible components. First, every accepted non-trivial zero is covered by a generated zero:

$$\prod_{u:\mathcal{Z}_\zeta} \sum_{z:Z} \varepsilon(z) \sim u.$$

Second, every zero constructor preserves the triadic invariants

$$\mathsf{P}, \qquad \mathsf{U}, \qquad \mathsf{T},$$

read respectively as prime-ledger distinction, unit-boundary legality, and recursive time generation. Third, hidden coordinates are confined to unit phase or prime-register fibres:

$$\mathbb{T} \quad \textit{and} \quad \prod_{p \in S} \mathbb{Z}_p.$$

Fourth, no generated row may import an independent real positive scale ledger

$$\mathbb{R}_{>0}$$

as a closure coordinate. Thus the zeta sector is not treated as a projection embedded in a larger dissipative system; the row asks for the completed zero presentation itself to be source-internal.

Definition 9255.17 (Zeta Hist-flow closure row)**.** *The* zeta Hist-flow closure row *is the internal-time component of* $\mathsf{ZetaCosmosClosure}_\zeta$*. It requires the time data of a generated zeta-zero packet to factor through constructor-tail histories:*

$$B_0 = \mathsf{Empty}, \qquad B_{j+1} = E_{\epsilon_{j+1}}(B_j), \qquad \epsilon_{j+1} \in \{0,1\}.$$

The marks $\mathsf{b0}$ *and* $\mathsf{b1}$ *are read only as primitive distinction emissions; the state row is the resulting history* B_j*. Thus a later row contains its predecessor as constructor tail, and the time order is the generated history depth rather than an external clock. Write the closure target as*

$$\mathsf{ZetaHistFlowClosure}_\zeta.$$

For zeta, this row says that every accepted zero packet receives its phase-time readout from such internal histories and from the unitary phase component

$$e^{-ith_a},$$

not from an independent real positive scale flow.

Principle 9255.18 (Hist-flow closure rejects scale time)**.** *Assume a generated zero packet carries* $\mathsf{ZetaHistFlowClosure}_\zeta$ *and has active channel*

$$e^{-(s-1/2)h_a}$$

with

$$s = \frac{1}{2} + r + it, \qquad h_a > 0.$$

The internal Hist-flow supplies constructor extension and its phase readback. It supplies no independent coordinate whose readout is $e^{-rh_a} \in \mathbb{R}_{>0}$*. Hence a nonzero* r *would be a scale-time import rather than a Hist-generated phase row. The Hist-flow component of cosmos closure therefore refines the no-radial-time condition of Definition 9255.14.*

Principle 9255.19 (Cosmos closure absorbs the strong route obligations)**.** *The row* $\mathsf{ZetaCosmosClosure}_\zeta$ *packages the same analytic load as the conservative-packet route, but with the hidden-scale boundary made explicit. Its coverage component is zero-presentation exactness for Definition 9255.2. Its invariant-preservation component is the fixed-half constructor closure of Definition 9255.4. Its phase and prime-register restriction is the compact hidden-fibre side of Definition 9255.15. Its refusal of an* $\mathbb{R}_{>0}$ *closure coordinate is the no-radial-time row of Definition 9255.14.*

Corollary 9255.20 (Cosmos closure is conditionally sufficient)**.** *If* $\mathsf{ZetaCosmosClosure}_\zeta$ *holds for the RH-free zero signature, then the hypotheses of Definition 9255.23 hold.*

Proof. The coverage component supplies the essential coverage row $Z \simeq \mathcal{Z}_\zeta$. The triadic invariant-preservation component supplies constructor closure for the fixed-half subcarrier. The absence of an independent $\mathbb{R}_{>0}$ closure coordinate rules out a nontrivial factor e^{-rh_a} on every active unit-boundary channel, so boundary stability is available for the strong packet. These are exactly the two rows used by Definition 9255.23. □

Principle 9255.21 (Physical gap inventory)**.** *In the conservative-oscillator reading, the initiality route leaves five separate mathematical obligations. First, each zero must have a canonical dominant normal-mode packet, not merely an arbitrary finite truncation. Second, that packet must strongly capture the zeta zero, upgrading weak eta/Rouché capture to a source-compatible generated*

packet. Third, the packet must satisfy unitary boundary stability, excluding the damping or gain factor e^{-rh}. Fourth, the infinite tail must satisfy the no-absorption discipline of Definition 9413.143, so the far end does not act as a hidden reservoir for scale defect. Fifth, all accepted non-trivial zeta zeros must be covered by these generated packets, which is the zero-presentation exactness row of Definition 9255.2.

Corollary 9255.22 (Physical form of the conditional route)**.** *If every accepted non-trivial zeta zero is represented by a conservative normal-mode packet, then the only admissible centered-channel closure is pure phase closure. Hence the hypotheses of Definition 9255.23 are met, and the conditional initiality route supplies the RH section.*

Proof. Representation by conservative normal-mode packets supplies strong capture and zero-presentation coverage. The unitary phase and no damping-or-gain rows supply boundary stability. The far-end compatibility row prevents the infinite tail from absorbing a displayed real scale defect. These are exactly the two inputs isolated in Definition 9255.23: zero-presentation exactness and fixed-half closure. Applying that theorem gives the RH section. □

Theorem 9255.23 (Strong capture and boundary stability imply the initiality route)**.** *Assume an RH-free zero signature F_Z with initial algebra Z and a realization $\varepsilon : Z \to \mathcal{Z}_\zeta$. If every accepted non-trivial zeta-zero packet is represented by a strong generated packet and every constructor of that strong packet preserves the unit-boundary row of Definition 9255.10, then the hypotheses of Definition 9255.5 are exactly the zero-presentation exactness and fixed-half closure supplied by those two rows.*

Proof. Strong representation of every accepted non-trivial zero supplies the essential coverage component of Definition 9255.2; together with the displayed soundness and faithfulness rows, this is $\varepsilon : Z \simeq \mathcal{Z}_\zeta$. Boundary stability states that each constructor preserves the unit-boundary readout, hence preserves the predicate

$$P_{1/2}(z) = \mathsf{OnCritLine}(\pi_s(\varepsilon(z)))$$

appearing in Definition 9255.3. This is precisely the algebra structure on H_Z required by Definition 9255.4. Applying Definition 9255.5 then yields the RH section. □

Principle 9255.24 (Initiality does not give zero completeness)**.** *The initiality of $Z = \mu F_Z$ proves induction over the zeros generated by F_Z. It does not prove that F_Z has generated every accepted critical-strip zero packet. The essential coverage component of Definition 9255.2 is therefore one of the two central mathematical obligations of the route. Without it, an off-half zero outside the generated carrier would not be seen by initial-algebra induction.*

Principle 9255.25 (Constructor closure is the fixed-half hinge)**.** *The other central obligation is fixed-half closure of the constructors. Administrative constructors, such as classifier transport and ledger reconstruction, should preserve $P_{1/2}$ by stability. Symmetry constructors, such as conjugation and functional-equation transport, must preserve the mirror fixed surface. Limit constructors must preserve $P_{1/2}$ through real-part continuity and uniqueness of the completed limit. Primitive zero-source constructors are the analytic hinge: they must produce fixed-half zero packets without smuggling an* OnCritLine *witness into their input data.*

Principle 9255.26 (Window induction and limit windows)**.** *The same distinction applies to inspection windows. Finite generated windows are handled by structural induction over their window signature. Limit windows are handled as compatible inverse systems*

$$W_0 \leftarrow W_1 \leftarrow W_2 \leftarrow \cdots ,$$

with a limit-window predicate read as

$$P_\infty(w) := \prod_n P_n(\pi_n w).$$

This removes the need to inspect an untyped totality of windows while still recording the exact compatibility data required for a completed zero carrier.

Principle 9255.27 (Failure readings)**.** *The initiality route has three mathematically distinct failure readings. If* $\mathsf{NoRHCirc}(F_Z)$ *fails, the proposed generator has assumed the fixed-half conclusion. If zero-presentation exactness fails, the generator does not cover the accepted non-trivial zero carrier. If fixed-half closure fails for a constructor, the failing constructor identifies the analytic mechanism by which an off-half packet could enter the recovered carrier. None of these failures is a route-closure decider or an independence theorem for the fixed proposition* RH*.*

Theorem 9255.28 (Completeness bridge to constructive RH)**.** Lean4 ✓ `BEDC.Derived.RHRoute.GenerativeZeroCo` `of completion`

↪ variant `BEDC.Derived.RHRoute.GenerativeZeroCompletion.symmetryReadbackRH of completion`

Let $\varepsilon : Z \to \mathcal{Z}_\zeta$ *be a generated realization whose fixed-half closure is displayed, as in Definition 9255.2. Package the completeness datum as the record whose rows are* ε*, the fixed-half closure witness, and the coverage row*

$$\prod_{u:\mathcal{Z}_\zeta} \mathsf{NontrivZero}(u) \to \sum_{z:Z} \varepsilon(z) \sim u.$$

From this record, together with the located readback row

$$\mathsf{OnCritLineReadback}(u) \to \mathsf{OnCritLine}(u),$$

the fixed-half closure supplies $\mathsf{CritLine}(\varepsilon(z))$ *for the covering generator, and critical-line transport along* $\sim$ *yields* $\mathsf{OnCritLine}(u)$*. This is the constructive Riemann hypothesis of chapter 9254, read off the generation side. The theorem is conditional: it consumes the coverage record and the readback row as hypotheses and constructs neither. The coverage row is the completeness obligation of Definition 9255.2; it is not discharged here, and no inhabitant of the completeness record is exhibited.*

Principle 9255.29 (Zero-free forward dynamics and the obligation firewall)**.** Lean4 (definition) `BEDC.Derived.RHRoute.LocatedGenerationTower.ZetaBridgeObligations`
Lean4 (definition) `BEDC.Derived.RHRoute.LocatedGenerationTower.generated true zero twoWay`

The generation side is presented as a forward dynamics that never inspects a zeta value. A tick carries a radial log-scale row, an angular rotation row, and a fresh orthogonal residual, gated by a no-adjacent Zeckendorf certificate. A state stores only its generated syntax; its located point is the fold of that syntax, never a stored coordinate, so no tick can fit a zero backwards. The analytic content is quarantined behind an evaluator interface whose binding, completeness, and soundness rows are the only bridge to the true zero set, and the forward dynamics constructs none of them. Under the packaged bridge the generated orbit and the true zero set stand in a displayed two-way correspondence.

Remark 9255.30 (Non-vacuity of the forward dynamics)**.** Lean4 (definition) `BEDC.Derived.RHRoute.LocatedGenera` `generated`

A toy instance fills every abstract row of the forward dynamics with concrete data: a prime-local seed at $p = 2$*, a constant located-complex algebra, and a tick whose register advances. It exhibits an inhabitant of the generated-point type at tick depth one, so the dynamics and its membership predicate are not vacuous. The instance carries no zeta semantics; it certifies only that the generation types are inhabited and that the dynamics runs.*

9255.1 Spectral reformulation and the Hilbert–Pólya wall

The completeness obligation of Definition 9255.2 can be carried by a single spectral statement rather than a pointwise zeta binding. The generation side supplies, ζ-free, a formal phase register, a self-adjoint height operator, and the coupling that removes every internal bookkeeping obligation; the only surviving analytic content is that the generated spectrum is the true zero spectrum.

Principle 9255.31 (Prime-phase character factorization). Lean4 (definition) BEDC.Derived.RHRoute.OCLSDPhaseAlg
Lean4 ✓ BEDC.Derived.RHRoute.OCLSDPhaseAlgebra.readPZGPhase tick

The prime register reads back into a formal phase algebra by multiplying one atom phase per activated Zeckendorf bit,

$$\chi_t(n) = \prod_{(p,k)} \mathsf{primePhase}(p,k),$$

multiplicatively over register concatenation. The atom phase depends on the prime and Fibonacci index alone, so it is prime-locked: no free angle enters, and the phase cannot be tuned to a target height.

Definition 9255.32 (Spectral zero identity). Lean4 (definition) BEDC.Derived.RHRoute.OCLSDSpectral.SpectralZer
Lean4 (definition) BEDC.Derived.RHRoute.OCLSDSpectral.zetaBridge from spectral

Let H_{OCLSD} be the diagonal operator whose orthonormal basis is the forward orbits and whose eigenvalue on an orbit is its cumulative angular height. The spectral zero identity is the pair of readings

$$\gamma \in \mathrm{Spec}(H_{\mathrm{OCLSD}}) \iff \zeta(\tfrac{1}{2} + i\gamma) = 0.$$

It is strictly weaker than pointwise evaluator binding, and it suffices: with a binding stub and the located readback it assembles the full zero-bridge record.

Principle 9255.33 (Self-adjoint generator, prime-locked spectrum). Lean4 (definition) BEDC.Derived.RHRoute.OCLS
selfAdjoint
Lean4 ✓ BEDC.Derived.RHRoute.OCLSDPrimeLock.primeLocked angle independent

Because the orbit heights are located-real, H_{OCLSD} is self-adjoint and generates a unitary phase flow. Because the per-tick height factors through the prime register only, equal prime data forces equal height: the spectrum is determined by prime arithmetic, not by the free angle row.

Principle 9255.34 (Critical coupling closes the internal obligations). Lean4 (definition) BEDC.Derived.RHRoute.OCL
from stateHeight
Lean4 (definition) BEDC.Derived.RHRoute.OCLSDCriticalCoupling.zetaBridge from spectral coupled

The orbit-to-critical-line readback is not an analytic wall but a coupling gap between the syntactic point readback and the accumulated height. A state-level height that commutes with both discharges it by induction over the orbit recursor. After this the entire zero-bridge record follows from the spectral zero identity and a binding stub alone. The one remaining obligation is the spectral zero identity itself — that the prime-generated spectrum is the true zeta zero spectrum. That is the Hilbert–Pólya content; it is not discharged here, and nothing in this chapter constructs it.

Concretized in §8799: The RH-free zero-generation initiality route is represented as a finite NameCert packet with an explicit no-circularity audit.

Chapter 9256

Ledger Register and Tick Reflection Positivity

chapter 9255 builds the OCLSD zero-generation dynamics: a prime-driven tick advances a syntactic term, the accumulated angular heights are the eigenvalues of a self-adjoint operator H_{OCLSD}, and the sole remaining wall is the spectral zero identity (that the prime-generated spectrum is the true zeta zero spectrum). The present chapter records two finite rational sub-layers that sit strictly *below* that wall and are fully discharged: the arithmetic of the prime–Zeckendorf register (which grounds the tick's log-time bookkeeping), and the reflection positivity classification of the tick input (which separates the tick data that admit a unitary reconstruction from those that do not). Neither sub-layer touches the spectral zero identity; both are closed by finite located-rational data.

9256.1 Prime–Zeckendorf register arithmetic

The OCLSD register is a finite list of (prime, Fibonacci-index) atoms. Its per-prime weight and the integer it encodes are the arithmetic carrier of the "register norm squared equals log-time" bookkeeping.

Definition 9256.1 (Per-prime Fibonacci weight)**.** Lean4 (definition) `BEDC.Derived.RHRoute.OCLSDRegisterWeight.`

For a prime p the weight of a register R is the sum of F_k over the atoms of R whose prime equals p,

$$w_p(R) \;=\; \sum_{(p,k)\in R} F_k,$$

where F_k is the k-th Fibonacci number. For a Zeckendorf register this reads back the prime valuation v_p.

Theorem 9256.2 (Zeckendorf carry is an isometry of the register weight)**.** Lean4 ✓ `BEDC.Derived.RHRoute.OCLSDR` `carry`

Replacing two adjacent atoms $(p,k),(p,k+1)$ by $(p,k+2)$ preserves every per-prime weight, because $F_k + F_{k+1} = F_{k+2}$. The register's per-prime Fibonacci content is therefore carry-invariant: a well-defined valuation, not an artifact of the digit representation.

Definition 9256.3 (Encoded integer)**.** Lean4 (definition) `BEDC.Derived.RHRoute.OCLSDRegisterWeight.encodedNat`

The integer a register encodes is

$$n(R) \;=\; \prod_{(p,k)\in R} p^{F_k}.$$

Grouped by prime this is $n(R) = \prod_p p^{w_p(R)}$, so the register norm squared $\sum_{(p,k)} F_k \log p = \log n(R)$ is the log-time T.

Theorem 9256.4 (The ledger is a submonoid of the integers). Lean4 ✓ BEDC.Derived.RHRoute.OCLSDRegisterWeig carry

↪ variant BEDC.Derived.RHRoute.OCLSDRegisterWeight.encodedNat append

The Zeckendorf carry preserves $n(R)$, so the log-time is a carry-invariant of the register. Register concatenation multiplies the encoded integer, $n(R_1 +\!\!+ R_2) = n(R_1)\, n(R_2)$, so the encoding is a monoid homomorphism from the ledger onto the multiplicative monoid of S-smooth integers; the log-time is additive under concatenation.

9256.2 Finite reflection positivity as a PSD classification

A single reversible tick carries a located transition weight q. Its Osterwalder–Schrader reflection form, in the manifestly completed-square shape, is

$$\Phi_q(f_0, f_1) \;=\; q\,(f_0 + f_1)^2 \;+\; (1 - 2q)\,(f_0^2 + f_1^2).$$

Definition 9256.5 (Reflection form). Lean4 (definition) BEDC.Derived.RHRoute.OCLSDReflectionForm.osForm

Φ_q is the two-state reversible reflection quadratic form. On the antisymmetric mode $(1, -1)$ the first term vanishes and $\Phi_q(1, -1) = (1 - 2q) \cdot 2$: the reflection form on that mode is exactly the small transition eigenvalue.

Theorem 9256.6 (Reflection positivity classification). Lean4 ✓ BEDC.Derived.RHRoute.OCLSDReflectionForm.ref holds

↪ variant BEDC.Derived.RHRoute.OCLSDReflectionForm.reflection fails

↪ variant BEDC.Derived.RHRoute.OCLSDReflectionForm.ratSq nonneg

If $0 \le q$ and $2q \le 1$ then $\Phi_q(f_0, f_1) \ge 0$ for every f_0, f_1 — the transition operator is positive semidefinite and the reflection form is a sum of two nonnegative terms. If $2q > 1$ then $\Phi_q(1, -1) < 0$: an explicit indefinite counterexample. Reflection positivity is thus a classification condition *on the transition weight, discharged by the located square-nonnegativity fact $0 \le x^2$.*

9256.3 Reflection positivity of the OCLSD tick input

The classification is parameterized by the actual OCLSD tick.

Definition 9256.7 (Tick reflection kit). Lean4 (definition) BEDC.Derived.RHRoute.OCLSDTickReflection.TickRefle Lean4 (definition) BEDC.Derived.RHRoute.OCLSDTickReflection.tickReflectionForm

A tick reflection kit assigns to each raw generation step g a located transition weight $q(g)$ with $0 \le q(g)$; the tick's reflection form is $\Phi_{q(g)}$. The tick is reflection-positive *when $2q(g) \le 1$.*

Theorem 9256.8 (Antisymmetric mode decides reconstructability). Lean4 ✓ BEDC.Derived.RHRoute.OCLSDTickRe reflection holds

↪ variant BEDC.Derived.RHRoute.OCLSDTickReflection.tick reflection fails

↪ variant BEDC.Derived.RHRoute.OCLSDTickReflection.reflectionPositive of flip nonneg

A tick is reflection-positive if and only if its reflection form is nonnegative on the antisymmetric mode $(1, -1)$. A reflection-positive tick has $\Phi_{q(g)} \ge 0$ on every mode (a unitary Osterwalder–Schrader reconstruction is available); a reflection-anticorrelated tick is strictly negative on the antisymmetric mode (no such reconstruction). A single located-rational check on one mode therefore decides the tick's reconstructability.

9256.4 Carrier-relative Osterwalder–Schrader reconstruction

The reflection-positivity classification is the *hypothesis* of the reconstruction theorem: from a reflection-positive transition operator P one builds a positive self-adjoint generator $H = -\Delta T^{-1} \log P$ and the unitary flow e^{itH}. That passage is analytic — located operator logarithm, monotone spectral calculus, a GNS construction — none of which the finite located-rational layer builds. It is encoded honestly as a *carrier contract*: the analytic content is posited as an obligation, never faked.

Definition 9256.9 (Reconstruction carrier)**.** Lean4 (definition) BEDC.Derived.RHRoute.OSReconstructionContract.
Lean4 (definition) BEDC.Derived.RHRoute.OSReconstructionContract.OSReconstruction
Lean4 (definition) BEDC.Derived.RHRoute.OSReconstructionContract.OSReconstructionCarrier

A reconstruction kit posits the analytic carrier: a located time axis, transfer/generator/unitary operator types, the self-adjoint, nonnegative and unitary predicates, and the operations $P \mapsto \log P$, $H \mapsto e^{-tH}$, $H \mapsto e^{itH}$. A reconstruction of a tick g is a generator H with its self-adjointness, nonnegativity and unitary-flow certificates together with the heat-kernel identity $e^{-\Delta T \cdot H} = P_g$ recovering the tick's transfer operator. A carrier bundles the obligation as a map sending a reflection-positive tick to its reconstruction.

Theorem 9256.10 (The finite check feeds the reconstruction)**.** Lean4 (definition) BEDC.Derived.RHRoute.OSReconst
8 4 from finite RP
Lean4 (definition) BEDC.Derived.RHRoute.OSReconstructionContract.theorem 8 4 carrier relative

Against any reconstruction carrier, a tick whose reflection form is ≥ 0 on the antisymmetric mode — a single located-rational inequality — admits an Osterwalder–Schrader reconstruction. The finite classification discharged above is exactly the hypothesis the carrier consumes. No concrete carrier is supplied: the analytic construction remains an obligation, and the statement "reflection positivity yields a reconstruction carrier" is deliberately not *asserted, since a degenerate all-trivial carrier would fake it. The junction is honest, not the reconstruction.*

Principle 9256.11 (The classification closes; the reconstruction does not)**.** Lean4✓ BEDC.Derived.RHRoute.OCLSD
reflectionPositive

The reflection-positive tick class is inhabited over any dynamics type (the trivial $q \equiv 0$ kit is reflection-positive), so the classification is not vacuous. What the finite rational layer supplies is exactly the positivity gate: which tick inputs admit a unitary reconstruction. What it does not *supply is the reconstruction itself — the passage from a positive transition operator to the located generator $H = -\Delta T^{-1} \log P$ and its unitary group needs located operator logarithm and monotone spectral calculus, which this layer does not build. That analytic step, and the spectral zero identity of chapter 9255, remain the walls; nothing here constructs either.*

Chapter 9257

Pigeonhole Time Recurrence

chapter 9255 isolates the source-internal zero-generation route, and Definition 9255.17 records its Hist-flow time component. The present chapter records the finite-projection recurrence row attached to that component. Pigeonhole recurrence supplies compact phase-return candidates at finite resolution. It does not supply a radial scale coordinate, a zero-presentation equivalence, or a boundary-stability certificate.

Definition 9257.1 (Finite prime phase-time flow)**.** *Let*

$$S = \{p_1, \dots, p_m\}$$

be a finite displayed prime window. The finite prime phase-time flow *is*

$$\Phi_S(t) := \left(e^{-it\log p_1}, \dots, e^{-it\log p_m}\right) \in \mathbb{T}^S.$$

The defining row is that one parameter t drives every displayed prime frequency. A family of independent parameters $(t_p)_{p\in S}$ is not a single Hist-flow time readout.

Theorem 9257.2 (Exact finite prime phase time is unique)**.** *If S contains two distinct prime rows and*

$$\Phi_S(t) = \Phi_S(t'),$$

then $t = t'$.

Proof. Put $\Delta = t - t'$. Equality of the phase windows gives, for every $p \in S$,

$$e^{-i\Delta\log p} = 1.$$

Thus

$$\Delta \log p = 2\pi k_p$$

for some integer k_p. Choose distinct primes $p, q \in S$. If $\Delta \neq 0$, then k_p and k_q are nonzero, and

$$\frac{\log p}{\log q} = \frac{k_p}{k_q} \in \mathbb{Q}.$$

Hence there are nonzero integers a, b with

$$a \log p = b \log q.$$

Exponentiating gives

$$p^a = q^b,$$

contradicting unique factorization. Therefore $\Delta = 0$, so $t = t'$. □

Definition 9257.3 (Finite phase observer)**.** *A* finite phase observer *on the prime window S is a finite quotient readout*

$$q_n : \mathbb{T}^S \to C_n$$

where C_n is a finite carrier. It records the resolution available at the observation layer n. Equality after applying q_n is a finite projection equality, not equality of the full phase state.

Theorem 9257.4 (Pigeonhole phase recurrence candidate)**.** *Let $q_n : \mathbb{T}^S \to C_n$ be a finite phase observer. For any finite time list*

$$t_0, t_1, \ldots, t_M$$

with

$$M + 1 > |C_n|,$$

there are indices $i < j$ such that

$$q_n(\Phi_S(t_i)) = q_n(\Phi_S(t_j)).$$

The difference $t_j - t_i$ is a finite-resolution phase recurrence candidate.

Proof. The list

$$q_n(\Phi_S(t_0)), \ldots, q_n(\Phi_S(t_M))$$

has $M + 1$ entries in the finite carrier C_n. Since $M + 1 > |C_n|$, two entries are equal. This gives the displayed indices $i < j$. □

Principle 9257.5 (Projection recurrence is not time collapse)**.** *Pigeonhole recurrence occurs after applying the finite observer q_n. It does not contradict Definition 9257.2. The full ledger may still distinguish $\Phi_S(t_i)$ from $\Phi_S(t_j)$, while the finite projection cannot. Thus the recurrence row is legal only as a ledgered finite projection collision.*

Principle 9257.6 (Finite phase recurrence is not inverse dynamics)**.** *The named row*

$$\mathsf{PigeonholePhaseClosureNoScaleTime}_\zeta$$

records the boundary between finite phase recurrence and inverse dynamics. A finite observer can identify two projected phase readouts,

$$q_n(\Phi_n(t_i)) = q_n(\Phi_n(t_j)),$$

but this is not an inverse evolution from the later readout to the earlier ledger. Any source, tail, winding, parity, or prime-register data erased by the projection must have been retained as ledger data if it is to be recovered.

For a centered active channel

$$e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a},$$

the factor e^{-ith_a} belongs to the unit phase layer and may enter the finite recurrence row. The factor e^{-rh_a} belongs to $\mathbb{R}_{>0}$. Mirror transport sends r to $-r$, but the cancellation

$$e^{-rh_a} e^{rh_a} = 1$$

uses a radial ledger carrying the value of r and the active height h_a. It is therefore not a phase recurrence produced by the finite observer and not a legal replacement for unit-boundary closure.

Definition 9257.7 (Scale-time energy in a zero tower)**.** *Let a generated zero tower have active rows A_n with heights $h_a > 0$. For*

$$s = \frac{1}{2} + r + it,$$

the scale-time defect at $a \in A_n$ is

$$d_a(s) := -rh_a.$$

The finite scale-time energy is

$$E_n(s) := \sum_{a \in A_n} d_a(s)^2, \qquad W_n := \sum_{a \in A_n} h_a^2.$$

Equivalently,

$$E_n(s) = r^2 W_n.$$

Theorem 9257.8 (Pigeonhole recurrence with bounded energy excludes scale time)**.** *Assume a generated zero tower carries* $\mathsf{ZetaHistFlowClosure}_\zeta$*, finite phase observers with pigeonhole recurrence candidates, the scale-time energy row*

$$E_n(s) = r^2 W_n,$$

cofinal growth

$$W_n \to \infty,$$

and a zero-compatible bound

$$E_n(s) \le B$$

for all displayed tower levels. Then $r = 0$.

Proof. The pigeonhole row supplies recurrence candidates only in compact finite phase projections. By Definition 9257.6, these collisions are not inverse dynamics and do not license radial-scale closure. By Definition 9255.18, a nonzero radial parameter r would be a scale-time import rather than a Hist-generated phase row. If $r \neq 0$, then $r^2 > 0$, and cofinal growth gives

$$r^2 W_n \to \infty.$$

Using the energy identity,

$$E_n(s) = r^2 W_n,$$

we obtain $E_n(s) \to \infty$, contradicting the bound $E_n(s) \le B$. Therefore $r = 0$. □

Corollary 9257.9 (The pigeonhole row is a phase-candidate row)**.** *The pigeonhole row in the RH route supplies finite-resolution phase return candidates. The exclusion of off-half scale time still requires the Hist-flow closure, cofinal active growth, and bounded normal-energy rows.*

Proof. Definition 9257.4 gives only finite projection collisions. The full no-scale conclusion uses Definition 9257.8, whose hypotheses also include Hist-flow closure, the radial energy identity, cofinal growth, and a zero-compatible bound. □

Principle 9257.10 (Pigeonhole recurrence is route structure)**.** *The pigeonhole row is a structural component of the RH route, not an inhabitant of the RH witness type. It clarifies why finite-resolution phase projections of a single time flow must contain return candidates. It does not prove that such a candidate is a canonical zeta-zero packet, that the packet closes on the unit boundary, that the far end cannot absorb radial defect, or that every accepted non-trivial zero is covered by the generated carrier.*

Corollary 9257.11 (Source obligations after finite phase recurrence)**.** *After the finite phase-recurrence row is displayed, the source obligations for the route are:*

$$\mathsf{StrongActiveCapture},$$

$$\mathsf{BoundaryStability},$$

$$\mathsf{ZetaCosmosClosure}_{\zeta},$$

and

$$\mathsf{ZeroPresentationCompleteness}.$$

Proof. Strong active capture is the row turning finite phase-return candidates into canonical zero-relevant packets. Boundary stability is the row showing that those packets close only on the unit boundary. Zeta cosmos closure is the row excluding an external real positive scale ledger at the far end. Zero-presentation completeness is the row identifying all accepted non-trivial zeta-zero packets with generated zero packets. Together these rows are exactly the source content that Definition 9257.8 assumes but does not construct. □

Definition 9257.12 (Finite zero certificate)**.** *A* finite zero certificate *for the completed zeta packet is a finite analytic packet*

$$(D, F, T, \Lambda, \Gamma)$$

where D is a rationally described located-compact box or disk, the local decomposition row is

$$\Xi = F + T$$

on a neighbourhood of D, the boundary gap row is

$$|T(s)| < |F(s)| \qquad s \in \partial D,$$

the local-degree or winding row Λ gives the zero count of F in D, and the ledger row Γ records source, tail, boundary, interval, and completeness data. The certificate is a finite packet; it does not expose a bare exact zero coordinate.

Definition 9257.13 (Zero certificate tower)**.** *A* zero certificate tower *is a compatible sequence of finite zero certificates*

$$(D_n, F_n, T_n, \Lambda_n, \Gamma_n)_{n\geq 0}$$

such that

$$D_{n+1} \subseteq D_n, \qquad \operatorname{diam}(D_n) \to 0,$$

and the refinement maps preserve the Rouché gap, local-degree or winding row, source ledger, tail ledger, and completeness ledger. This is the RH-facing reading of the convergent active-capture zero tower of Definition 9413.633.

Theorem 9257.14 (Certificate towers generate completed zeros)**.** *If a zero certificate tower has nonempty located-compact boxes and each level certifies one completed-zero island of Ξ, then its nested boxes have a unique limit readout s_∞ and*

$$\Xi(s_\infty) = 0.$$

Proof. The nesting and diameter rows give a unique point in

$$\bigcap_n D_n$$

by Definition 9413.632. The Rouché and local-degree rows certify the same completed-zero island through the refinement maps. Therefore Definition 9413.634 applies and yields $\Xi(s_\infty) = 0$. □

Corollary 9257.15 (Legal certificate towers cannot converge off half)**.** *Let a zero certificate tower be boundary-stable and carry the Hist-flow and no-scale rows used in Definition 9257.8. If its limit readout is*

$$s_\infty = \frac{1}{2} + r + it,$$

then $r = 0$.

Proof. The tower supplies a completed zero by Definition 9257.14. Boundary stability supplies the zero-compatible bound, while the active rows give the scale-time energy identity and cofinal growth. Applying Definition 9257.8 gives $r = 0$. □

Principle 9257.16 (Zeros are certificate limits)**.** *The RH route should treat a completed zero readout as the limit of compatible finite certificates rather than as a naked point supplied outside the system. The unresolved global source row is not the existence of finite approximations; it is that every accepted non-trivial zeta zero is represented by a legal boundary-stable certificate tower whose generated limit is covered by the BEDC zero carrier.*

Principle 9257.17 (No naked zero, not no zero)**.** *The certificate-tower reading does not deny the existence of completed zeros. It denies only the admissibility of an ungenerated bare coordinate as the primitive datum of the proof route. A finite zero certificate is a located analytic assertion with a source ledger, a tail ledger, a boundary or winding row, and a refinement interface. A completed zero is then the exact readout selected by a compatible tower of such finite assertions. Thus the distinction is:*

no primitive naked zero *rather than* *no zero.*

Theorem 9257.18 (Finite certificates can determine exact zeros)**.** *Let*

$$(D_n)_{n\geq 0}$$

be nonempty located-compact zero boxes with

$$D_{n+1} \subseteq D_n, \qquad \operatorname{diam}(D_n) \to 0.$$

Assume each D_n *carries a certificate that* D_n *contains a completed zero of* Ξ*. Then the tower determines a unique point*

$$\rho \in \bigcap_n D_n$$

and

$$\Xi(\rho) = 0.$$

Proof. The nesting and diameter hypotheses give a unique point in the intersection by Definition 9413.632. Choose, for each n, a certified zero $\rho_n \in D_n$. Since the boxes are nested and their diameters tend to zero, the sequence ρ_n converges to the unique intersection point ρ. The completed zeta packet Ξ is continuous on the displayed analytic neighbourhoods, so

$$\Xi(\rho) = \lim_n \Xi(\rho_n) = 0.$$

□

Corollary 9257.19 (Approximation is not the proof object)**.** *A finite numerical or analytic box is not itself the completed zero. The proof object is the compatible certificate tower whose limit readout is exact. Consequently, the RH-facing task is not to replace exact zeros by approximations, but to prove that every accepted zero is represented by a legal boundary-stable tower.*

Proof. The first sentence is the distinction of Definition 9257.17. Exactness of the limit readout follows from Definition 9257.18. The final sentence is the global coverage row already isolated in Definition 9257.16. □

Definition 9257.20 (Certificate-tower triadic invariant)**.** *For a finite zero certificate*

$$\mathcal{C} = (D, F, T, \Lambda, \Gamma),$$

write

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}) := \mathcal{I}_{\mathsf{P}}(\mathcal{C}) \wedge \mathcal{I}_{\mathsf{U}}(\mathcal{C}) \wedge \mathcal{I}_{\mathsf{T}}(\mathcal{C}).$$

Here $\mathcal{I}_{\mathsf{P}}$ is the source and prime-ledger distinction row, $\mathcal{I}_{\mathsf{U}}$ is the unit-boundary and legal-gauge row, and $\mathcal{I}_{\mathsf{T}}$ is the Hist-flow single-time row. The invariant says that finite analytic capture is accompanied by a ledgered source, a unit-boundary gauge discipline, and an internal time readout. It is not merely the equation $\Xi = F + T$.

Theorem 9257.21 (Generated certificate towers preserve the triadic invariant)**.** *Assume a zero certificate tower is generated by RH-free certificate constructors, every constructor preserves $\mathcal{I}_{\mathrm{BEDC}}$, and every refinement morphism transports the invariant from the finer level to the displayed coarser level. Then every level of the tower satisfies*

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}_n).$$

Proof. The generated certificate carrier is read by structural induction over its constructors. The source constructors supply the prime-ledger row. Mirror, conjugation, classifier transport, and ledger replay transport an existing invariant without adding an unrecorded radial coordinate. Analytic gluing and finite-window closure use the displayed Rouché, tail, and boundary ledgers. Compatible-limit and recursive-tower readback use the refinement hypotheses. Thus each constructor preserves $\mathcal{I}_{\mathrm{BEDC}}$, and induction gives the invariant at every generated level. The refinement morphism hypothesis then makes the levelwise readouts compatible through the tower. □

Definition 9257.22 (Legal boundary-stable certificate tower)**.** *A zero certificate tower is* legal boundary-stable *when it is an active-capture tower, preserves the triadic invariant of Definition 9257.20, carries Hist-flow closure, carries no far-end absorption, and for its limit readout*

$$s_\infty = \frac{1}{2} + r + it$$

has defect extraction

$$E_n(s_\infty) = r^2 W_n + o(W_n),$$

cofinal growth

$$W_n \to \infty,$$

and a zero-compatible bound

$$E_n(s_\infty) \le B$$

for some finite B independent of n.

Theorem 9257.23 (No scale closure for legal certificate towers)**.** *The limit readout of a legal boundary-stable certificate tower satisfies*

$$r = 0.$$

Proof. Assume $r \neq 0$. Then $r^2 > 0$. Since

$$E_n(s_\infty) = r^2 W_n + o(W_n),$$

there is a cofinal tail of levels on which

$$|o(W_n)| \leq \frac{r^2}{2} W_n.$$

Hence, on that tail,

$$E_n(s_\infty) \geq \frac{r^2}{2} W_n.$$

By cofinal growth, $W_n \to \infty$, so $E_n(s_\infty)$ is unbounded. This contradicts the zero-compatible bound $E_n(s_\infty) \leq B$. Therefore $r = 0$. □

Corollary 9257.24 (Legal certificate towers have fixed-half limits)**.** *If a zero certificate tower is legal boundary-stable and generates a completed zero readout s_∞, then*

$$\Re(s_\infty) = \frac{1}{2}.$$

Proof. Write

$$s_\infty = \frac{1}{2} + r + it.$$

By Definition 9257.23, $r = 0$. Thus $\Re(s_\infty) = 1/2$. □

Definition 9257.25 (Strong zero-presentation completeness)**.** *The strong certificate coverage target is*

$$\mathsf{StrongZeroPresentationCompleteness}_\zeta.$$

It asserts that every accepted non-trivial completed zeta zero is the limit readout of a legal boundary-stable certificate tower. This target is stronger than weak eta/Rouché capture: weak capture supplies finite analytic boxes, while strong coverage also supplies canonical source, triadic invariant preservation, boundary stability, no far-end absorption, and generated-zero coverage.

Corollary 9257.26 (Certificate-tower proof load)**.** *If* $\mathsf{StrongZeroPresentationCompleteness}_\zeta$ *holds, then every accepted non-trivial completed zeta zero has fixed-half readout.*

Proof. Let u be an accepted non-trivial completed zeta zero. Strong zero-presentation completeness supplies a legal boundary-stable certificate tower whose limit readout is u. By Definition 9257.24, that readout has real part 1/2. □

Chapter 9258

PZG Finite-Window Self-Reference Recurrence

chapter 9257 records finite phase recurrence for a Hist-flow time row. The present chapter records the corresponding PZG register form. The point is the same boundary in a different coordinate system: full PZG coding is faithful, while every finite readout window creates fibres, recurrence candidates, and ledger obligations.

9258.1 Finite PZG Windows

Definition 9258.1 (Finite PZG readout window)**.** *A finite PZG readout window is a finite set*

$$W \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow} \times \mathbb{N}.$$

Its visible carrier B_W is the finite set of legal bit patterns on W compatible with the Zeckendorf no-adjacent condition on each displayed prime axis. The window projection is

$$\pi_W : \mathsf{Z}_{\mathsf{PZG}} \to B_W,$$

where $\mathsf{Z}_{\mathsf{PZG}}$ denotes the full finite-support PZG register carrier and π_W forgets every coordinate outside W.

Principle 9258.2 (Global faithfulness, finite fibres)**.** *The full PZG register is a faithful source code by Definition 9291.9. A finite readout window is not faithful to the full source. Equality*

$$\pi_W(z) = \pi_W(z')$$

means equality at the displayed resolution, not equality of full PZG registers. Any difference between z and z' is hidden from W and is therefore a fibre, refinement, or ledger issue.

Theorem 9258.3 (Every finite PZG window has infinite fibres)**.** *Let W be a finite PZG readout window. Every realizable window position $b \in B_W$ has infinitely many full PZG registers above it.*

Proof. Choose one full register z with $\pi_W(z) = b$. Since W uses only finitely many prime axes and depths, there are infinitely many primes q whose first coordinate $(q, 1)$ is not in W and whose axis is absent from the displayed window. For each such q, form the register obtained from z by activating the coordinate $(q, 1)$ if it is not already active on that fresh axis, leaving all coordinates in W unchanged. These registers remain legal PZG registers, are pairwise distinct, and all project to b under π_W. □

Theorem 9258.4 (Finite-window PZG recurrence)**.** *For any infinite PZG trajectory*

$$z_0, z_1, z_2, \ldots$$

and any finite PZG readout window W, there exist $i < j$ such that

$$\pi_W(z_i) = \pi_W(z_j).$$

Proof. The projected sequence

$$\pi_W(z_0), \pi_W(z_1), \pi_W(z_2), \ldots$$

takes values in the finite carrier B_W. The pigeonhole principle gives two indices $i < j$ with the same projected value. □

Principle 9258.5 (Finite return is not global repetition)**.** *The equality in Definition 9258.4 is a finite readout recurrence. It does not assert $z_i = z_j$. If the full registers differ, the difference is a window-hidden source row. A legal proof route must either refine the window until the difference is read or record the difference in the residual or tail ledger.*

Definition 9258.6 (Window-cycle residual)**.** *Given a finite-window recurrence*

$$\pi_W(z_i) = \pi_W(z_j) \qquad (i < j),$$

a window-cycle residual is the ledger row

$$\mathsf{CycleRes}_W(i, j) := (z_i, z_j, W, \pi_W(z_i) = \pi_W(z_j), \mathsf{status}).$$

Its status is closed only when the full registers classify together in the claimed scope. Otherwise it is refined to a larger window or kept as a tail residual.

Theorem 9258.7 (Window recurrence creates a ledger obligation)**.** *Every finite-window recurrence with unequal full registers creates a window-cycle residual unless a larger displayed readout separates the two registers.*

Proof. If $\pi_W(z_i) = \pi_W(z_j)$ but z_i and z_j are not classified together at the full claimed scope, then the window equality has hidden the source difference. By Definition 9258.2, that difference is not erased by the finite readout. It must either be exposed by a refinement window or recorded by the ledger row of Definition 9258.6. □

Principle 9258.8 (Zeta readout groups finite-window fibres)**.** *On a displayed convergence window, the zeta-facing PZG readout may be grouped through any finite PZG readout window:*

$$\sum_{z \in \mathsf{Z}_{\mathsf{PZG}}} e^{-sL(z)} = \sum_{b \in B_W} \sum_{\pi_W(z) = b} e^{-sL(z)}.$$

The outer sum ranges over visible finite positions, while the inner sums range over the hidden fibres of those positions. Pigeonhole recurrence supplies finite-window multiple occupancy; the phase weights supply interference; the completed carrier supplies the zero ledger.

9258.2 Zero Reads

Definition 9258.9 (Finite zero read)**.** *A finite zero read for a completed zeta-facing packet is a finite data row*

$$\mathfrak{r} = (D, \varepsilon, m, \mathcal{P}, \mathcal{E}).$$

Here D is a rationally described located-compact box or disk in the complex plane, ε is a displayed precision or tail tolerance, m is a finite zero-count readout, $\mathcal{P}$ is the finite protocol that justifies the readout, and $\mathcal{E}$ is the error, boundary, branch, source, and tail ledger. A finite zero read is a reading event; it is not the completed zero object itself.

Theorem 9258.10 (ZeroRead finite-read theorem)**.** *Every auditable zero read factors through a finite zero read.*

Proof. An auditable zero read must display a region in which the read is made, the precision or tolerance at which the read is made, the finite zero count or zero-type assertion being read, the finite protocol that justifies the assertion, and the ledger for error, boundary, branch, source, and tail data. These are exactly the fields of Definition 9258.9. Thus the read is a finite data row. □

Corollary 9258.11 (Zero reads have PZG addresses)**.** *Every finite zero read has a PZG address.*

Proof. The finite zero read $\mathfrak{r}$ is a finite data row, hence it has a finite syntactic description. By Definition 9292.6, that description has a unique PZG register address. □

Principle 9258.12 (Completed zeros are compatible readout closures)**.** *A completed zero is not a single finite zero read. It is the closed readout selected by a compatible system of finite zero reads with shrinking localisation, stable zero-count or winding data, analytic-continuation transport, and closed source, boundary, tail, and error ledgers. This is the same certificate-tower boundary as Definition 9257.13: finite reads supply the stages, while the completed zero is the globalized limit readout licensed by the compatible tower.*

Theorem 9258.13 (Finite zero reads create zero-ledger obligations)**.** *If a finite zero read is used as evidence for a completed zero, then its tail, boundary, branch, and error fields must be closed by a compatible refinement tower or retained in the zero ledger.*

Proof. By Definition 9258.10, the read supplies only finite data. By Definition 9258.12, a completed zero requires compatibility through refinement and analytic transport, together with closed source, boundary, tail, and error ledgers. Any field not closed by such a compatible tower remains outside the completed zero readout and must therefore be retained in the zero ledger. □

Chapter 9259

Resolution Refinement Induction

chapter 9257 organizes finite phase recurrence and zero certificates. This chapter records the refinement form of the same route: begin with a displayed resolution, refine the resolution through legal certificate steps, prove the triadic invariant at every level by induction, and only then take the limit readout. The statement is a conditional route. It identifies the source load required for a BEDC proof, rather than replacing that load by the existence of finite approximations.

Definition 9259.1 (Resolution scale)**.** *A* resolution scale *is a sequence*

$$\varepsilon_0 > \varepsilon_1 > \varepsilon_2 > \cdots > 0$$

with

$$\varepsilon_n \to 0.$$

At level n, a zero box D_n is displayed at this resolution when

$$\operatorname{diam}(D_n) \le \varepsilon_n.$$

The scale records observational precision; it is not an additional analytic coordinate of the zeta carrier.

Definition 9259.2 (Resolution zero certificate)**.** *A* resolution zero certificate *at level n is a finite zero certificate*

$$\mathcal{C}_n = (D_n, F_n, T_n, \Lambda_n, \Gamma_n)$$

in the sense of Definition 9257.12, together with the scale row

$$\operatorname{diam}(D_n) \le \varepsilon_n.$$

Its analytic capture component is the displayed decomposition

$$\Xi = F_n + T_n$$

and the boundary gap

$$|T_n(s)| < |F_n(s)| \qquad s \in \partial D_n.$$

Its BEDC component is the ledger row Γ_n, which must record the source, tail, phase, prime-register, boundary, defect, and energy data used by the refinement step.

Definition 9259.3 (Legal resolution refinement)**.** *A refinement*

$$\mathcal{C}_n \rightsquigarrow \mathcal{C}_{n+1}$$

is legal *when the following rows hold:*

$$D_{n+1} \subseteq D_n, \qquad \operatorname{diam}(D_{n+1}) \leq \varepsilon_{n+1},$$

the Rouché or winding certificate at level $n+1$ is valid, the source and tail ledgers refine the level-n ledgers, and the triadic invariant of Definition 9257.20 is transported from level n to level $n+1$. Equivalently, the refinement is not merely a smaller box; it is a smaller box whose analytic, source, boundary, and defect ledgers remain compatible.

Theorem 9259.4 (Resolution induction)**.** *Let*

$$(\mathcal{C}_n)_{n\geq 0}$$

be a sequence of resolution zero certificates. If

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}_0)$$

holds and every step

$$\mathcal{C}_n \rightsquigarrow \mathcal{C}_{n+1}$$

is a legal resolution refinement, then

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}_n)$$

holds for every n.

Proof. The proof is induction on n. The base case is the displayed hypothesis at level 0. For the induction step, assume

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}_n).$$

Since $\mathcal{C}_n \rightsquigarrow \mathcal{C}_{n+1}$ is legal, Definition 9259.3 transports the triadic invariant to level $n+1$. This gives

$$\mathcal{I}_{\mathrm{BEDC}}(\mathcal{C}_{n+1}).$$

Therefore the invariant holds at every finite resolution level. □

Definition 9259.5 (Resolution certificate tower)**.** *A* resolution certificate tower *is a sequence of resolution zero certificates*

$$(\mathcal{C}_n)_{n\geq 0}$$

such that every adjacent step is a legal resolution refinement. It is strong *when, in addition, it is a legal boundary-stable certificate tower in the sense of Definition 9257.22. Thus strong resolution towers include canonical source dominance, unit-boundary stability, no far-end absorption, cofinal active growth, and defect extraction.*

Theorem 9259.6 (Resolution towers have exact completed-zero limits)**.** *If a resolution certificate tower has nonempty located-compact boxes and each level certifies the same completed-zero island of Ξ, then it has a unique limit readout s_∞ and*

$$\Xi(s_\infty) = 0.$$

Proof. The scale row gives $\operatorname{diam}(D_n) \to 0$, while legal refinement gives $D_{n+1} \subseteq D_n$. The Rouché or winding row certifies the same completed-zero island through the compatible ledgers. Therefore Definition 9257.14 applies. □

Theorem 9259.7 (Strong resolution towers have fixed-half limits)**.** *If a resolution certificate tower is strong and has limit readout*

$$s_\infty = \frac{1}{2} + r + it,$$

then $r = 0$.

Proof. By Definition 9259.4, every finite level preserves the triadic invariant. Since the tower is strong, it is legal boundary-stable and carries defect extraction, cofinal growth, and a zero-compatible energy bound. Applying Definition 9257.23 gives $r = 0$. □

Corollary 9259.8 (Resolution induction separates weak and strong refinement)**.** *Weak eta/Rouché refinement can supply smaller analytic zero boxes and tail bounds. It does not by itself supply the strong resolution tower rows: canonical source dominance, boundary stability, no far-end absorption, and zero-presentation coverage.*

Proof. The weak eta row supplies finite analytic capture as in Definition 9255.9. The strong conclusion of Definition 9259.7 additionally uses the legal boundary-stable tower hypotheses in Definition 9259.5. These rows are not consequences of a Rouché gap alone. □

Principle 9259.9 (Resolution route proof load)**.** *The refinement-induction route reduces the RH-facing source load to the following assertion:*

$$\mathsf{StrongResolutionTowerCoverage}_\zeta.$$

This means that every accepted non-trivial completed zeta zero is represented by a strong resolution certificate tower. Once such a tower is supplied, Definition 9259.7 excludes the radial scale parameter. The unresolved content is the construction and boundary-stability of the strong tower, not the formal induction over its already legal refinement steps.

Definition 9259.10 (Golden resolution ruler)**.** *Let*

$$F_0 = 0, \qquad F_1 = 1, \qquad F_{n+2} = F_{n+1} + F_n,$$

and let φ *be the golden self-scale of Definition 9346.12. A* golden resolution ruler *is the resolution scale*

$$\delta_m := \varphi^{-m}$$

together with a Fibonacci truncation schedule

$$N_m := F_{Lm},$$

where L *is a fixed positive integer chosen to dominate the local tail estimate. The spatial resolution is* δ_m*, while the analytic packet resolution is* N_m*.*

Definition 9259.11 (Zeckendorf resolution ledger)**.** *For a positive integer* N*, a* Zeckendorf resolution ledger *is a finite digit row*

$$N = \sum_{k \geq 2} z_k F_k, \qquad z_k \in \{0, 1\}, \qquad z_k z_{k+1} = 0.$$

The no-adjacent condition is the stable-window grammar of Definition 9346.1. The ledger records the Fibonacci-block source of a truncation level; it is a source address and not a proof of boundary stability.

Theorem 9259.12 (Zeckendorf ledgers give unique truncation addresses)**.** *Every positive truncation level N has at most one Zeckendorf resolution ledger.*

Proof. This is the uniqueness part of the Zeckendorf normal form. If two digit rows with no adjacent 1 represent the same N, normal-form uniqueness identifies their digits coefficientwise. Thus the Fibonacci-block source of the truncation is unique. □

Definition 9259.13 (Golden eta certificate packet)**.** *Let ρ be an eta zero in a displayed compact subdomain of $\Re(s) > 0$. For a golden ruler and a fixed L, the level-m* golden eta certificate packet *is*

$$\mathcal{G}_m(\rho) := (D_m, P_m, R_m, \Gamma_m),$$

where

$$D_m := \{s : \ |s - \rho| \le \varphi^{-m}\},$$

$$P_m(s) := \sum_{n=1}^{F_{Lm}} (-1)^{n-1} n^{-s}, \qquad R_m(s) := \eta(s) - P_m(s),$$

and Γ_m contains the Zeckendorf ledger of F_{Lm}, the tail bound, the phase rows, and the defect-energy rows. The packet is weak unless Γ_m also supplies the strong boundary-stability and no-far-end rows.

Theorem 9259.14 (Golden eta weak active capture)**.** *Let ρ be an eta zero of multiplicity q in a compact local neighbourhood contained in $\Re(s) \ge \sigma_0 > 0$. If the golden ruler uses a positive integer L with*

$$L\sigma_0 > q,$$

then, for all sufficiently large m, the golden eta packet satisfies

$$|R_m(s)| < |P_m(s)| \qquad s \in \partial D_m.$$

Hence P_m and η have the same zero count in D_m.

Proof. Write locally

$$\eta(s) = (s - \rho)^q g(s)$$

with g analytic and $g(\rho) \ne 0$. After shrinking the displayed neighbourhood, there is $c > 0$ such that $|g(s)| \ge c$ on it. For $s \in \partial D_m$ this gives

$$|\eta(s)| \ge c\varphi^{-qm}.$$

Uniform convergence of the eta series on the displayed compact subdomain gives a constant C such that

$$|R_m(s)| \le C F_{Lm}^{-\sigma_0}.$$

Since F_{Lm} grows at the golden rate, there is $C' > 0$ with

$$|R_m(s)| \le C' \varphi^{-L\sigma_0 m}.$$

The inequality $L\sigma_0 > q$ implies that, for all sufficiently large m,

$$C' \varphi^{-L\sigma_0 m} < \frac{c}{2} \varphi^{-qm}.$$

Thus $|R_m(s)| < |\eta(s)|/2$ on ∂D_m. Since $P_m = \eta - R_m$,

$$|P_m(s)| \ge |\eta(s)| - |R_m(s)| > \frac{1}{2}|\eta(s)|,$$

and therefore $|R_m(s)| < |P_m(s)|$ on ∂D_m. Rouché's theorem then gives equality of zero counts in D_m. □

Corollary 9259.15 (Golden rulers supply weak resolution towers)**.** *Every accepted non-trivial zeta zero in the open critical strip admits, after the eta readout is licensed, a weak golden resolution certificate tower.*

Proof. In the open critical strip the eta and zeta zero rows agree after the standard nonvanishing factor between them is ledgered. Applying Definition 9259.14 on a nested sequence of golden boxes gives Rouché capture at all sufficiently fine levels. The boxes have diameter tending to zero, and the Zeckendorf ledgers give unique truncation addresses by Definition 9259.12. This is exactly a weak resolution certificate tower. □

Definition 9259.16 (Strong golden resolution tower)**.** *A weak golden resolution tower is* strong *when its ledgers also carry canonical packet dominance, unit-boundary stability, no far-end absorption, cofinal active-height growth, and defect extraction. The corresponding bridge target is*

$$\mathsf{ZetaGoldenLeeYangBridge}.$$

It asks for the Fibonacci or Zeckendorf packet to be transported to a Lee–Yang type unit-boundary certificate compatible with the zeta phase and tail ledgers.

Corollary 9259.17 (Strong golden coverage is conditionally sufficient)**.** *If every accepted non-trivial completed zeta zero is represented by a strong golden resolution tower, then every such zero has fixed-half readout.*

Proof. A strong golden resolution tower is a strong resolution certificate tower. Therefore Definition 9259.7 applies to its limit readout. □

Principle 9259.18 (Golden rulers do not supply boundary stability by themselves)**.** *The golden ruler fixes a clean resolution schedule and a unique Zeckendorf address ledger. It supplies weak active capture for eta packets once the local tail estimate is dominated by the golden shrinking rate. It does not prove that the finite packets are unit-boundary stable. The remaining strong row is the* ZetaGoldenLeeYangBridge *of Definition 9259.16, not another application of Rouché capture.*

Definition 9259.19 (Zeckendorf–Gödel path packet)**.** *Let X_m be the no-adjacent Zeckendorf window of Definition 9346.1, and let $(p_k)_{1\le k\le m}$ be a displayed prime register. For*

$$z = (z_1, \dots, z_m) \in X_m$$

define the Zeckendorf–Gödel integer

$$g(z) := \prod_{k=1}^{m} p_k^{z_k}.$$

For

$$u_k(s) := p_k^{1/2-s},$$

the corresponding path packet is

$$Q_m(s) := \sum_{z\in X_m} w_z \prod_{k=1}^{m} u_k(s)^{z_k}.$$

When the weights are the stable path weights, this is the finite independent-set packet attached to the no-adjacent grammar. Its advantage over the eta truncation is structural: the monomial support is already the Zeckendorf path carrier. Its missing row is zero-island capture for the completed zeta packet.

Definition 9259.20 (Golden stable capture bridge)**.** *A* golden stable capture bridge *for a zero island D_m consists of a weak analytic packet*

$$P_m(s) = \sum_{n=1}^{F_{Lm}} (-1)^{n-1} n^{-s},$$

a Zeckendorf–Gödel path packet Q_m, and bridge data satisfying:

$$|R_m(s)| < |P_m(s)| \qquad s \in \partial D_m,$$

the packet Q_m carries a Lee–Yang type unit-boundary stability certificate, and the two packets have the same local zero-island readout, for instance by a boundary comparison

$$|P_m(s) - Q_m(s)| < |P_m(s)| \qquad s \in \partial D_m,$$

or by an explicitly ledgered equality of winding numbers

$$\mathrm{wind}_{\partial D_m}(P_m, 0) = \mathrm{wind}_{\partial D_m}(Q_m, 0).$$

The bridge is canonical only when the construction of Q_m uses the Zeckendorf, prime-register, and path-stability ledgers rather than a post hoc interpolation to the zero island.

Principle 9259.21 (Golden weak capture and stable capture are separate rows)**.** *The golden eta packet and the Zeckendorf–Gödel path packet solve different parts of the route. The eta packet supplies analytic Rouché capture of a completed zero island, but it does not carry Lee–Yang stability. The path packet is organized by the no-adjacent Zeckendorf grammar and is the natural candidate for a Lee–Yang stable finite packet, but it does not automatically capture zeta zeros. The source theorem must identify these two rows through Definition 9259.20.*

Definition 9259.22 (Golden stable capture coverage)**.** *The coverage target*

$$\mathsf{GoldenStableCaptureBridge}_\zeta$$

asserts that every accepted non-trivial completed zeta zero admits, at some sufficiently fine golden resolution level, a golden stable capture bridge in the sense of Definition 9259.20.

Theorem 9259.23 (Golden stable capture coverage gives fixed-half readout)**.** *If*

$$\mathsf{GoldenStableCaptureBridge}_\zeta$$

holds, then every accepted non-trivial completed zeta zero has fixed-half readout.

Proof. Let u be an accepted non-trivial completed zeta zero with local coordinate

$$s = \frac{1}{2} + r + it.$$

The coverage target supplies a golden stable capture bridge. The weak analytic packet captures the zero island, and the bridge transfers the same winding row to the Zeckendorf–Gödel path packet Q_m. The Lee–Yang type certificate for Q_m excludes zero readouts with all active variables strictly inside the unit boundary and also excludes the corresponding outside region. Since

$$|u_k(s)| = |p_k^{1/2-s}| = p_k^{-r},$$

an $r > 0$ readout places every active variable strictly inside the unit circle, while an $r < 0$ readout places every active variable strictly outside it. Both cases contradict the stable packet's zero-island readout. Therefore $r = 0$. □

Principle 9259.24 (Golden blueprint)**.** *The BEDC–Golden route has the following compact form. The golden ruler and Zeckendorf ledger provide weak finite-resolution active capture; the only strong row is the bridge from that analytic capture packet to a Zeckendorf–Gödel Lee–Yang stable packet with the same zero-island winding. Thus the hard source theorem is*

$$\mathsf{GoldenStableCaptureBridge}_{\zeta},$$

not the existence of the golden ruler itself.

Chapter 9260

Golden Lee–Yang Bridge Obligations

chapter 9259 supplies the golden ruler, the Zeckendorf truncation ledger, and weak eta Rouché capture. The present chapter isolates the source rows needed to turn that weak capture into a Lee–Yang type boundary-stable packet. The bridge is a source theorem about the same zero island; it is not a restatement of fixed-half readout.

Definition 9260.1 (Fibonacci path carrier)**.** *For a width m, let X_m be the stable Zeckendorf window of Definition 9346.1. A* Fibonacci path carrier *is the finite hard-core packet*

$$I_m(t) := \sum_{z \in X_m} t^{|z|}, \qquad |z| := \sum_{k=1}^{m} z_k.$$

Equivalently, X_m is the independent-set carrier of the path graph on m vertices. The no-adjacent Zeckendorf grammar therefore supplies a finite path-model source row, but this row is only combinatorial until it is tied to a completed zeta zero island.

Definition 9260.2 (Lee–Yang lift certificate)**.** *Let $A(t)$ be a finite path-model polynomial of degree d. Its Joukowsky lift is the polynomial*

$$\mathcal{L}_A(u) := (1+u)^{2d} A\left(-\frac{u}{(1+u)^2}\right).$$

A Lee–Yang lift certificate *for A is a finite ledger proving that every nontrivial zero of $\mathcal{L}_A$ lies on the unit circle, or equivalently that the lifted zero-relevant packet has no zero branch in the strict inside or strict outside regions*

$$0 < |u| < 1, \qquad |u| > 1.$$

The certificate is a unit-boundary row for the lifted path packet. It does not by itself identify that packet with an eta or zeta packet.

Definition 9260.3 (Golden bridge witness)**.** *Let*

$$\mathcal{G}_m(\rho) = (D_m, P_m, R_m, \Gamma_m)$$

be a golden eta certificate packet in the sense of Definition 9259.13. A golden Lee–Yang bridge witness *for $\mathcal{G}_m(\rho)$ consists of a Fibonacci path carrier I_{k_m}, a lifted packet $\mathcal{L}_m$, and a bridge ledger with the following rows:*

$$|R_m(s)| < |P_m(s)| \qquad s \in \partial D_m,$$

a Lee–Yang lift certificate for $\mathcal{L}_m$, a same-island transfer from P_m to $\mathcal{L}_m$, radial purity, and dominance.

The same-island transfer may be given by a boundary comparison

$$|P_m(s) - \mathcal{L}_m(s)| < |P_m(s)| \qquad s \in \partial D_m,$$

or by a ledgered homotopy H_τ with

$$H_0 = P_m, \qquad H_1 = \mathcal{L}_m, \qquad 0 \notin H_\tau(\partial D_m) \quad 0 \le \tau \le 1.$$

Radial purity means that the comparison error and homotopy rows contain only phase, finite source, and p-adic or Zeckendorf ledger data, never an unrecorded factor e^{-rh}. Dominance means that the lifted path packet is selected by the source ledger as the zero-relevant branch, rather than chosen after the zero island is known.

Principle 9260.4 (Bridge rows are independent)**.** *The rows in Definition 9260.3 have separate proof roles. The eta Rouché inequality captures the completed zero island. The path carrier supplies the no-adjacent Fibonacci grammar. The Lee–Yang lift supplies unit-boundary stability for the lifted path packet. The same-island transfer proves that the eta packet and the lifted packet read the same local zero event. Radial purity prevents the transfer from hiding a real positive scale factor. Dominance makes the lifted packet canonical rather than interpolatory.*

Definition 9260.5 (Dominant branch packet)**.** *Let a finite analytic packet on a zero island Ω_m have a spectral expansion*

$$P_m(z) = \sum_{j=1}^{d} C_j(z)\lambda_j(z)^m.$$

A pair λ_+, λ_- is a dominant branch pair on Ω_m if there is $\kappa > 0$ such that every remaining branch satisfies

$$|\lambda_j(z)| \le (1-\kappa)\max\{|\lambda_+(z)|, |\lambda_-(z)|\}$$

on Ω_m. The dominant and subdominant parts are

$$P_m^{\mathrm{dom}}(z) := C_+(z)\lambda_+(z)^m + C_-(z)\lambda_-(z)^m$$

and

$$P_m^{\mathrm{sub}}(z) := \sum_{j \ne +,-} C_j(z)\lambda_j(z)^m.$$

Theorem 9260.6 (Dominant gap gives Rouché control)**.** *If a dominant branch packet satisfies*

$$|P_m^{\mathrm{sub}}(z)| < |P_m^{\mathrm{dom}}(z)| \qquad z \in \partial\Omega_m,$$

then P_m and P_m^{dom} have the same number of zeros in Ω_m, counted with multiplicity.

Proof. By definition,

$$P_m = P_m^{\mathrm{dom}} + P_m^{\mathrm{sub}}.$$

The displayed strict boundary inequality is precisely the Rouché hypothesis for the pair $(P_m^{\mathrm{dom}}, P_m^{\mathrm{sub}})$ on $\partial\Omega_m$. Rouché's theorem gives equality of the zero counts in Ω_m. □

Principle 9260.7 (Quartic endpoint finite model)**.** *section 9413.9 gives a finite-certificate model for the dominant-branch row. The fixed quartic carrier records recurrence algebra, companion-matrix spectrum, elliptic normalization, real projection, dominant branch, and endpoint-zero ledgers inside one packet. Its dominance row is separated from its ramification row by Definition 9413.99, and its local endpoint-zero row is supplied by Definition 9413.102. The scope boundary of Definition 9413.105 is essential: the model proves a fixed finite spectral pattern, not a zeta golden spectral expansion or a growing-window theorem.*

Definition 9260.8 (Stable golden Rouché homotopy)**.** *Let P_m be the zero-relevant golden packet on Ω_m, and let $\mathcal{L}_m$ be a Fibonacci Lee–Yang lift packet in the same local coordinate u. A stable golden Rouché homotopy from P_m to $\mathcal{L}_m$ is a ledgered family H_τ, $0 \le \tau \le 1$, such that*

$$H_0 = P_m, \qquad H_1 = \mathcal{L}_m,$$

the boundary of the zero island is never crossed,

$$0 \notin H_\tau(\partial\Omega_m) \qquad 0 \le \tau \le 1,$$

and the Lee–Yang stable regions remain zero-free:

$$H_\tau(u) \neq 0 \qquad 0 < |u| < 1,$$

$$H_\tau(u) \neq 0 \qquad |u| > 1.$$

The homotopy ledger may use phase, finite source, prime-register, and Zeckendorf rows, but it may not introduce an unrecorded $\mathbb{R}_{>0}$ scale coordinate.

Theorem 9260.9 (Stable Rouché transport preserves the unit boundary)**.** *If P_m admits a stable golden Rouché homotopy to a Fibonacci Lee–Yang lift packet, then every zero of P_m in the zero-relevant local coordinate lies on the unit boundary.*

Proof. The stable-region rows in Definition 9260.8 apply at $\tau = 0$. Hence

$$P_m(u) \neq 0 \qquad 0 < |u| < 1,$$

and

$$P_m(u) \neq 0 \qquad |u| > 1.$$

Thus a zero of P_m in the zero-relevant coordinate cannot lie in the strict inside or strict outside region. The remaining boundary is $|u| = 1$. □

Principle 9260.10 (Ordinary Rouché transport is insufficient)**.** *Boundary nonvanishing alone transports winding number and zero count; it does not transport the Lee–Yang unit-boundary property. The polynomial $u - 1$ has its zero on $|u| = 1$, while $u - (1 + \varepsilon)$ has the same zero count in any sufficiently large island and its zero lies outside the unit circle. Therefore the source row needed for zeta golden boundary stability is*

$$\mathsf{GoldenStableRoucheHomotopy},$$

not merely a zero-count homotopy. The stable homotopy row must include strict inside and strict outside zero-free ledgers together with radial purity.

Definition 9260.11 (Zeckendorf–Mellin path packet)**.** *Let $F_0 = 0$, $F_1 = 1$, and*

$$F_{j+2} = F_{j+1} + F_j.$$

For the path graph on $\{1, \ldots, m\}$, let

$$\mathrm{Ind}(P_m) := \{S \subseteq \{1, \ldots, m\} : i, j \in S \Rightarrow |i - j| \neq 1\}.$$

For $S \in \mathrm{Ind}(P_m)$ set

$$N(S) := \sum_{j \in S} F_{j+1}.$$

The finite Zeckendorf ledger identifies $\mathrm{Ind}(P_m)$ with the integers

$$0 \leq N < F_{m+2}.$$

For a vector $\mathbf{x} = (x_1, \ldots, x_m)$ define the multivariable path partition function

$$I_m(\mathbf{x}) := \sum_{S \in \mathrm{Ind}(P_m)} \prod_{j \in S} x_j.$$

For $u > 0$ the Mellin fibre is

$$x_j(u) := (-1)^{F_{j+1}} e^{-uF_{j+1}}.$$

Theorem 9260.12 (Zeckendorf–Mellin exact projection)**.** *For $\Re(s) > 0$,*

$$\sum_{n=1}^{F_{m+2}-1} (-1)^{n-1} n^{-s} = -\frac{1}{\Gamma(s)} \int_0^\infty u^{s-1} \left(I_m(\mathbf{x}(u)) - 1\right) du.$$

Proof. For $n > 0$ and $\Re(s) > 0$, the Mellin–Laplace identity gives

$$n^{-s} = \frac{1}{\Gamma(s)} \int_0^\infty u^{s-1} e^{-nu} \, du.$$

The Zeckendorf ledger in Definition 9260.11 gives a bijection from nonempty independent sets S to the integers $1 \leq N(S) \leq F_{m+2} - 1$. Thus

$$\sum_{n=1}^{F_{m+2}-1} (-1)^{n-1} n^{-s} = \sum_{\varnothing \neq S \in \mathrm{Ind}(P_m)} (-1)^{N(S)-1} N(S)^{-s}.$$

Substituting the Mellin–Laplace identity and using finiteness to interchange the sum and integral yields

$$\frac{1}{\Gamma(s)} \int_0^\infty u^{s-1} \sum_{\varnothing \neq S} (-1)^{N(S)-1} e^{-uN(S)} \, du.$$

For every independent set S,

$$(-1)^{N(S)} e^{-uN(S)} = \prod_{j \in S} \left((-1)^{F_{j+1}} e^{-uF_{j+1}}\right) = \prod_{j \in S} x_j(u).$$

Therefore

$$\sum_{\varnothing \neq S} (-1)^{N(S)-1} e^{-uN(S)} = -\sum_{\varnothing \neq S} \prod_{j \in S} x_j(u) = -\left(I_m(\mathbf{x}(u)) - 1\right),$$

which is the displayed identity. □

Definition 9260.13 (Zeckendorf transfer dominated splitting)**.** *The source row*

ZeckendorfTransferDominatedSplitting

asserts that the non-autonomous path transfer product

$$\begin{pmatrix} I_j(\mathbf{x}(u)) \\ I_{j-1}(\mathbf{x}(u)) \end{pmatrix} = \begin{pmatrix} 1 & x_j(u) \\ 1 & 0 \end{pmatrix} \begin{pmatrix} I_{j-1}(\mathbf{x}(u)) \\ I_{j-2}(\mathbf{x}(u)) \end{pmatrix}$$

has a ledgered dominated splitting on the zero island after Mellin projection. Concretely, the row selects two fibre branches whose Mellin-integrated contributions form P_m^{dom}, while all remaining transfer contributions form P_m^{sub} and satisfy

$$|P_m^{\mathrm{sub}}(s)| < |P_m^{\mathrm{dom}}(s)| \qquad s \in \partial D_m.$$

Definition 9260.14 (Self-dual Mellin Lee–Yang stability operator)**.** *The source row*

SelfDualMellinLeeYangOperator

asserts that the Zeckendorf–Mellin projection is used only after a completed self-dual packet has been supplied. The raw one-sided Mellin integral of Definition 9260.12 is not a stability-preserving operator by itself. The self-dual row includes the functional-equation or Poisson-duality ledger that pairs the u and u^{-1} directions. After the fibrewise path packet is lifted by the Joukowsky Lee–Yang ledger, this completed packet produces no zero branch in

$$0 < |u| < 1$$

and no zero branch in

$$|u| > 1,$$

and the operator introduces no unrecorded $\mathbb{R}_{>0}$ scale gauge.

Theorem 9260.15 (Mellin rows supply the stable bridge source package)**.** *Assume*

ZeckendorfTransferDominatedSplitting

and

SelfDualMellinLeeYangOperator

for the golden eta zero island. Then the three analytic source rows of Definition 9260.16 are supplied for that island.

Proof. Definition 9260.12 identifies the golden eta truncation with a Mellin projection of the Zeckendorf path partition function. The dominated-splitting row supplies the branch decomposition and boundary dominance estimate required for GoldenSpectralExpansion and GoldenSpectralGap. The self-dual Mellin Lee–Yang stability operator supplies the strict inside and strict outside zero-free ledgers, with no radial scale gauge. Those are exactly the stability rows required by GoldenStableRoucheHomotopy. □

Definition 9260.16 (Golden bridge analytic source rows)**.** *The analytic source content of*

ZetaGoldenLeeYangBridge

splits into three rows:

$$\mathsf{GoldenSpectralExpansion}, \qquad \mathsf{GoldenSpectralGap}, \qquad \mathsf{GoldenStableRoucheHomotopy}.$$

The first row supplies a spectral or exponential-sum expansion for the golden packet on the displayed island. The second row identifies the dominant pair and proves the uniform gap needed by Definition 9260.6. The third row transports the selected dominant packet to the Fibonacci Lee–Yang lift while preserving the strict inside and strict outside zero-free regions. The Zeckendorf–Mellin route of Definition 9260.15 offers a more structured way to supply these rows: first identify eta golden truncation as a Mellin projection of path independent-set data, then prove dominated transfer splitting and self-dual Mellin stability preservation. Together these rows upgrade weak eta capture to a boundary-stable golden packet.

Theorem 9260.17 (A golden bridge witness supplies boundary stability)**.** *Let a completed zero island be captured by a golden eta packet $\mathcal{G}_m(\rho)$ and let Definition 9260.3 supply a bridge witness whose active lifted variables have the centered form*

$$u_a(s) = e^{-(s-1/2)h_a}, \qquad h_a > 0.$$

If the island readout is

$$s = \frac{1}{2} + r + it,$$

then the bridge witness excludes $r \neq 0$ for that island.

Proof. The same-island row transfers the local zero event from the eta packet to the lifted Lee–Yang packet without changing the winding ledger on ∂D_m. Hence the lifted packet has a zero-relevant branch at the same readout. For each active variable,

$$|u_a(s)| = \left|e^{-(s-1/2)h_a}\right| = e^{-rh_a}.$$

If $r > 0$, then every active variable lies in the strict inside region $|u_a(s)| < 1$. If $r < 0$, then every active variable lies in the strict outside region $|u_a(s)| > 1$. Both alternatives are forbidden by the Lee–Yang lift certificate, and radial purity prevents the bridge from absorbing the contradiction into an unledgered scale factor. Therefore $r = 0$. □

Definition 9260.18 (Golden Lee–Yang bridge source row)**.** *The source row*

$$\mathsf{ZetaGoldenLeeYangBridge}$$

asserts that every accepted non-trivial completed zeta zero has a cofinal golden certificate tower whose sufficiently fine levels carry golden Lee–Yang bridge witnesses. The witnesses must be compatible under refinement: the zero island, Zeckendorf address, lifted path packet, same-island transfer, radial-purity row, and dominance row at level $m+1$ restrict to the corresponding level-m rows.

Theorem 9260.19 (The bridge source row gives golden stable capture)**.** *If* $\mathsf{ZetaGoldenLeeYangBridge}$ *holds, then*

$$\mathsf{GoldenStableCaptureBridge}_\zeta$$

holds.

Proof. Let an accepted non-trivial completed zeta zero be given. The bridge source row supplies a cofinal golden tower with bridge witnesses at sufficiently fine levels. Each witness contains the weak eta capture row, the Zeckendorf or Fibonacci path packet, the Lee–Yang unit-boundary certificate, and a same-island transfer. These are exactly the rows required by the golden stable capture bridge of Definition 9259.20, with the additional radial-purity and dominance ledgers recording canonicity of the source. Thus the zero is covered by $\mathsf{GoldenStableCaptureBridge}_{\zeta}$. □

Corollary 9260.20 (Golden Lee–Yang bridge suffices for fixed-half readout)**.** *If* ZetaGoldenLeeYangBridge *holds, then every accepted non-trivial completed zeta zero has fixed-half readout.*

Proof. By Definition 9260.19, the golden stable capture coverage target holds. Apply Definition 9259.23. □

Principle 9260.21 (Dominant branch source load)**.** *The dominant branch row is the analytic load of the bridge. It must show that the lifted Fibonacci path packet is the zero-relevant finite source for the displayed island, with tail and comparison budgets strong enough to preserve the boundary gap. Without that row, a Lee–Yang stable path packet and a weak eta capture packet remain two unrelated finite objects. With it, the same island is controlled by a unit-boundary packet, and the radial scale coordinate has no legal place to close.*

Principle 9260.22 (Stable bridge proof boundary)**.** *Definition 9260.6 is an abstract finite theorem, and Definition 9260.7 records a finite model where the corresponding dominance and endpoint-isolation rows are present. The zeta golden route still requires the three analytic source rows of Definition 9260.16. Without the spectral expansion, spectral gap, and stable Rouché homotopy rows, the bridge remains a conditional source package; with them, the Lee–Yang unit-boundary certificate can be transported to the same zero island without introducing a radial scale anomaly.*

Principle 9260.23 (Mellin factorization of the bridge load)**.** *The bridge need not be approached as a direct homotopy from an eta truncation to a Lee–Yang polynomial. The exact projection of Definition 9260.12 factors the eta truncation through the Zeckendorf path partition function, where the Fibonacci transfer and Lee–Yang lift are intrinsic. The remaining analytic load is therefore the pair of source rows*

$$\mathsf{ZeckendorfTransferDominatedSplitting}$$

and

$$\mathsf{SelfDualMellinLeeYangOperator}.$$

The first row prevents the Mellin projection from selecting a non-Fibonacci dominant branch; the second replaces the raw one-sided Mellin readout by a completed self-dual readout and prevents it from creating a zero in the strict inside or strict outside region. This factorization is still a proof obligation, but it is more local than an unstructured comparison between the eta packet and an unrelated Lee–Yang packet.

Chapter 9261

Golden Bridge Three-Level Closure

chapter 9260 isolates the analytic bridge rows that remain between weak eta capture and a Lee–Yang stable golden packet. The present chapter records the same load in three strata: an abstract finite theorem, a fixed quartic finite model, and the zeta-facing golden source package. The strata are kept separate so that a verified finite pattern is not read as a global zeta theorem.

Definition 9261.1 (Three-level golden bridge package)**.** *The* three-level golden bridge package *consists of the following rows.*

The abstract row is the finite dominant-branch control theorem Definition 9260.6. It requires a displayed spectral expansion, a dominant pair, and the boundary inequality

$$|P_m^{\mathrm{sub}}(z)| < |P_m^{\mathrm{dom}}(z)| \qquad z \in \partial\Omega_m.$$

The finite-model row is the quartic endpoint carrier of Definition 9260.7. It shows that the dominant-branch, Puiseux, two-branch interference, and Rouché isolation rows can be realized inside one finite spectral carrier.

The zeta-facing row is the source package of Definition 9260.16, namely

$$\mathsf{GoldenSpectralExpansion}, \qquad \mathsf{GoldenSpectralGap}, \qquad \mathsf{GoldenStableRoucheHomotopy}.$$

This row is the only stratum that can feed the golden zeta bridge.

Theorem 9261.2 (Abstract dominant control is finite)**.** *The abstract row of Definition 9261.1 is a finite analytic theorem: it transports zero count from P_m^{dom} to P_m under the displayed strict boundary inequality.*

Proof. This is exactly Definition 9260.6. The theorem uses only the decomposition

$$P_m = P_m^{\mathrm{dom}} + P_m^{\mathrm{sub}}$$

and the strict boundary estimate. No zeta-specific coverage row, Lee–Yang bridge row, or infinite tower row is consumed. □

Principle 9261.3 (Quartic endpoint as finite witness)**.** *The quartic endpoint carrier witnesses the shape of the argument, not its zeta coverage. It contains a fixed spectral polynomial, a normalization ledger, a real projection ledger, a dominance ledger, and an endpoint-zero ledger. Its scope boundary refuses a growing-window endpoint-zero theorem and refuses any zeta conclusion. Thus it licenses the methodology of finite dominance followed by local Rouché isolation, while leaving the zeta golden spectral expansion and gap as separate source obligations.*

Definition 9261.4 (Zeta-facing strong golden bridge package)**.** *The source row*

$$\mathsf{ZetaFacingStrongGoldenBridge}_{\zeta,\varphi}$$

asserts that every accepted non-trivial completed zeta zero has a golden certificate tower on which the following compatible rows are present: golden weak eta capture, the Zeckendorf–Mellin exact projection of Definition 9260.12, the analytic source rows of Definition 9260.16, cofinal active-height growth, and no unledgered radial scale gauge. The package does not assert the analytic rows by definition; it records the source surface that must be supplied before the fixed-half readout can be exported.

Theorem 9261.5 (Strong golden bridge gives the zeta Lee–Yang bridge)**.** *If*

$$\mathsf{ZetaFacingStrongGoldenBridge}_{\zeta,\varphi}$$

holds, then

$$\mathsf{ZetaGoldenLeeYangBridge}$$

holds.

Proof. Fix an accepted non-trivial completed zeta zero. The source row of Definition 9261.4 supplies a golden certificate tower for that zero and supplies the analytic source rows of Definition 9260.16. These rows provide the dominant branch selection, the stable Rouché transport, and the Lee–Yang stable zero-free regions on the same zero island. Together with the weak eta capture and Zeckendorf address ledgers, they are precisely the golden Lee–Yang bridge witnesses required by Definition 9260.18. □

Theorem 9261.6 (Strong golden bridge gives fixed-half readout)**.** *If*

$$\mathsf{ZetaFacingStrongGoldenBridge}_{\zeta,\varphi}$$

holds, then every accepted non-trivial completed zeta zero has real part $1/2$*.*

Proof. By Definition 9261.5, the source row $\mathsf{ZetaGoldenLeeYangBridge}$ holds. Apply Definition 9260.20. □

Principle 9261.7 (Three-level proof boundary)**.** *The abstract row and the quartic endpoint row are evidence about proof shape. They do not cover zeta zeros. The zeta-facing row must still provide the analytic bridge package*

$$\mathsf{GoldenSpectralExpansion} \quad \mathsf{GoldenSpectralGap} \quad \mathsf{GoldenStableRoucheHomotopy},$$

or, through the Mellin factorization of Definition 9260.23, the two lower rows

$$\mathsf{ZeckendorfTransferDominatedSplitting} \qquad \mathsf{SelfDualMellinLeeYangOperator}.$$

Only after those rows are attached to the same golden zero island does the no-scale argument apply to the readout

$$s = \frac{1}{2} + r + it.$$

Definition 9261.8 (Renormalized Zeckendorf endpoint transfer)**.** *The endpoint scale for the Zeckendorf–Mellin fibre is not a fixed $u > 0$. It is the normalized scale*

$$u = \frac{\alpha}{F_{m+2}}.$$

At this scale the Mellin fibre variables become

$$\widetilde{x}_{m,j}(\alpha) := (-1)^{F_{j+1}} \exp\left(-\alpha \frac{F_{j+1}}{F_{m+2}}\right).$$

The associated non-autonomous path transfer matrices are

$$T_{m,j}(\alpha) := \begin{pmatrix} 1 & \widetilde{x}_{m,j}(\alpha) \\ 1 & 0 \end{pmatrix},$$

with total endpoint transfer

$$\mathcal{T}_m(\alpha) := T_{m,m}(\alpha) T_{m,m-1}(\alpha) \cdots T_{m,1}(\alpha).$$

The source row

RenormalizedZeckendorfEndpointTransfer

asserts that the zero-relevant Mellin window is read through this normalized transfer product, not through a fixed-u fibre where the large Fibonacci weights freeze the tail variables.

Principle 9261.9 (Endpoint scaling is part of dominance)**.** *For fixed $u > 0$, the variables*

$$x_j(u) = (-1)^{F_{j+1}} e^{-uF_{j+1}}$$

decay rapidly with j, so the last variables of a long Zeckendorf window no longer participate in a balanced Fibonacci competition. The renormalized scale of Definition 9261.8 keeps the terminal ratios

$$\frac{F_{j+1}}{F_{m+2}}$$

visible. Hence the dominated-splitting row in Definition 9260.13 must be proved for the normalized endpoint transfer product $\mathcal{T}_m(\alpha)$.

Definition 9261.10 (Mellin stability attack surfaces)**.** *The source row*

SelfDualMellinLeeYangOperator

may be attacked through any of the following stability-preservation surfaces.

The total-positivity surface proves that the completed Mellin kernel produces a Pólya-frequency or Laguerre–Pólya type transform compatible with the Lee–Yang zero-free regions.

The Hermite–Biehler surface rewrites the completed Mellin packet as a function

$$E(z) = A(z) - iB(z)$$

whose half-plane or unit-circle stability is controlled by a de Branges-type ledger.

The Rouché-stable discretization surface approximates the completed Mellin integral by finite sums

$$\sum_{\ell} w_\ell(s) L_{m,u_\ell},$$

with a stability-preserving choice of weights and a boundary error ledger strong enough to pass to the integral without crossing zero.

Principle 9261.11 (Mellin stability as no-demon finite form)**.** *The self-dual Mellin stability row is the finite packet analogue of the no-demon constraint. An off-half readout*

$$s = \frac{1}{2} + r + it$$

introduces the real scale factor

$$e^{-rh_a}$$

and therefore tilts any active-mode distribution by real-positive weights. If the completed self-dual Mellin readout preserves the Lee–Yang inside and outside zero-free regions, that tilt cannot be hidden as a phase or prime-register gauge. The only remaining legal readout is the unit boundary readout, where

$$|e^{-(s-1/2)h_a}| = 1.$$

Principle 9261.12 (Mellin projection obstruction)**.** *Fibrewise Lee–Yang stability is not, by itself, a stability theorem for Dirichlet-polynomial readout. The Zeckendorf fibre identity of Definition 9260.12 places the finite geometric packet*

$$K_M(u) = \sum_{N=0}^{M-1} (-1)^N e^{-uN}$$

on the unit boundary in the variable $z = e^{-u}$. The Mellin-projected Dirichlet packet, however, reads the same finite data through powers n^{-s}. The smallest non-trivial Dirichlet packet

$$P_2(s) = 1 - 2^{-s}$$

has zero set

$$s = \frac{2\pi i k}{\log 2}, \qquad k \in \mathbb{Z},$$

and therefore lies on the line $\Re(s) = 0$. Thus the unrestricted implication from unit-boundary fibre zeros to fixed-half Mellin zeros is false. Any zeta-facing use of the Mellin row must carry the additional ledger that transports the additive Zeckendorf height into the multiplicative prime height without creating a real-positive scale gauge.

Definition 9261.13 (Self-dual Zeckendorf completed packet)**.** *Let*

$$K_m(u) := I_m(\mathbf{x}(u)) - 1 = \sum_{n=1}^{F_{m+2}-1} (-1)^n e^{-nu}$$

be the one-sided Zeckendorf–Mellin kernel. A self-dual Zeckendorf completed packet *is a completed kernel in the logarithmic variable $y = \log u$ of the form*

$$\Psi_m(y) := e^{y/2} K_m(e^y) + e^{-y/2} K_m(e^{-y}),$$

or a ledger-equivalent Poisson-completed kernel with the same symmetry row

$$\Psi_m(y) = \Psi_m(-y).$$

Its completed Mellin readout is

$$\mathcal{X}_m\left(\frac{1}{2} + z\right) := \int_{-\infty}^{\infty} \Psi_m(y) e^{zy}\, dy,$$

whenever the displayed integral is replaced, if necessary, by the corresponding regularized finite-packet ledger. The point of the row is not the particular displayed completion, but the presence of a functional-equation ledger pairing the u and u^{-1} directions before Lee–Yang stability is read out.

Definition 9261.14 (Self-dual Zeckendorf Hermite–Biehler target)**.** *The source row*

$$\mathsf{SelfDualZeckendorfHB}$$

asserts that the completed transform of Definition 9261.13 belongs to a Hermite–Biehler, de Branges, or Laguerre–Pólya stability class strong enough to force every zero of

$$\mathcal{X}_m(s)$$

in the zero-relevant island to satisfy

$$\Re(s) = \frac{1}{2}.$$

Equivalently, after writing $s = 1/2 + z$, the completed transform has no zero with $\Re(z) \neq 0$ on the island covered by the certificate.

Theorem 9261.15 (Self-dual Hermite–Biehler target gives fixed-half packet)**.** *If a zero-relevant golden packet is captured by a self-dual Zeckendorf completed packet satisfying* $\mathsf{SelfDualZeckendorfHB}$*, and if the capture homotopy preserves winding on the boundary of the same island, then all zeros captured on that island have real part* $1/2$*.*

Proof. Write $s = 1/2 + z$. The self-dual completed packet supplies a transform

$$\mathcal{X}_m\left(\frac{1}{2} + z\right) = \int_{-\infty}^{\infty} \Psi_m(y) e^{zy}\, dy$$

with the symmetry ledger $\Psi_m(y) = \Psi_m(-y)$. The $\mathsf{SelfDualZeckendorfHB}$ row excludes zeros with $\Re(z) \neq 0$ on the zero-relevant island. Thus any zero of the completed packet in that island has $z = it$ and hence real part $1/2$. Boundary winding preservation transfers the same zero count from the captured golden packet to the completed packet on the island, so the captured packet zeros have the same fixed-half readout. □

Principle 9261.16 (One-sided Mellin is a source identity, not a stability row)**.** *Definition 9260.12 remains the exact source identity connecting the eta golden truncation to the Zeckendorf path partition function. Its role is to expose the finite path ledger and the cyclotomic unit-boundary fibre. It does not license unit-boundary stability after Mellin projection. The stability row must instead pass through the completed self-dual packet of Definition 9261.13 and the Hermite–Biehler target of Definition 9261.14.*

Definition 9261.17 (Zeckendorf–Gödel prime code)**.** *Let*

$$N = \sum_j z_j F_j, \qquad z_j \in \{0, 1\}, \qquad z_j z_{j+1} = 0,$$

be a Zeckendorf address, and let p_j be the j-th prime. The Zeckendorf–Gödel prime code *of this address is*

$$g(N) := \prod_j p_j^{z_j}.$$

Its prime height is

$$H_P(N) := \log g(N) = \sum_j z_j \log p_j.$$

This converts the no-adjacent Zeckendorf ledger into a prime-register ledger while keeping the address bits finite and certificate-readable.

Definition 9261.18 (Prime–Zeckendorf stable transport)**.** *A* prime–Zeckendorf stable transport *is a family of finite packet transports*

$$\mathcal{T}_\tau, \qquad 0 \le \tau \le 1,$$

from the additive Zeckendorf height

$$H_Z(N) = N$$

to a prime height such as $H_P(N) = \log g(N)$*, or to a zeta-facing height* $H_\zeta(n) = \log n$*, satisfying the following certificate rows on the same zero island: ledger uniqueness, the no-adjacent address constraint, unit phase boundary, absence of an unledgered* $\mathbb{R}_{>0}$ *scale anomaly, Rouché winding invariance, and preservation of the active dominant branch. The row records precisely the structure missing from a bare Mellin projection.*

Definition 9261.19 (Prime-compatible golden tower)**.** *A* prime-compatible golden tower *is a tower of finite packets* $\mathcal{P}_m$ *such that each layer is generated by a path-independent-set or Zeckendorf address ledger, each active address bit is read as a prime-register channel, the finite packet satisfies a multivariate Lee–Yang container ledger, refinement is cofinal in the zero-relevant packet family, Rouché capture is available for the accepted zeta zero packet under discussion, and the transport contains no unledgered real-positive scale gauge.*

Theorem 9261.20 (Prime-compatible tower gives fixed-half readout)**.** *If every accepted non-trivial completed zeta zero is captured by a prime-compatible golden tower, then every such zero has real part* $1/2$*.*

Proof. Let

$$\rho = \frac{1}{2} + r + it$$

be an accepted non-trivial completed zeta zero, and choose a prime-compatible golden tower capturing its zero packet. The tower supplies a Lee–Yang unit-boundary ledger for each active channel, hence for each active height $h_a > 0$ the centered readout satisfies

$$\left| e^{-(\rho-1/2)h_a} \right| = 1.$$

Since

$$e^{-(\rho-1/2)h_a} = e^{-rh_a} e^{-ith_a},$$

the modulus condition is

$$e^{-rh_a} = 1.$$

Because $h_a > 0$, this gives $r = 0$. Therefore $\Re(\rho) = 1/2$. □

Principle 9261.21 (Bridge factorization through prime registers)**.** *The zeta-facing golden bridge is not licensed by a single unrestricted Mellin stability assertion. Its certificate surface factors through two source rows:*

$$\mathsf{PrimeZeckendorfLift} \qquad \mathsf{PrimePacketToZetaHomotopy}.$$

The first row transports Fibonacci or path-independent-set Lee–Yang packets into Zeckendorf–Gödel prime packets. The second row transports those prime packets into the eta or zeta golden packet without changing boundary winding and without introducing a radial scale ledger. This factorization is the prime-ledger refinement of

$$\mathsf{ZetaGoldenLeeYangBridge}.$$

Definition 9261.22 (Multivariate Lee–Yang container)**.** *Let A_m be a finite active-mode set with heights $h_a > 0$. For a readout s set*

$$z_a(s) := e^{-(s-1/2)h_a}, \qquad \mathbf{z}_m(s) := (z_a(s))_{a\in A_m}.$$

A multivariate Lee–Yang container *is a finite polynomial*

$$\mathcal{Z}_m(\mathbf{z})$$

equipped with a Lee–Yang nonvanishing ledger

$$\mathcal{Z}_m(\mathbf{z}) \neq 0 \qquad \text{if } |z_a| < 1 \text{ for all } a,$$

and

$$\mathcal{Z}_m(\mathbf{z}) \neq 0 \qquad \text{if } |z_a| > 1 \text{ for all } a.$$

The container may be supplied by a finite ferromagnetic Ising packet, an Asano-contraction packet, a matching-stability packet, or a Rouché-stable specialization of such a packet.

Theorem 9261.23 (Multivariate container gives no-scale readout)**.** *Let a zero-relevant golden packet be Rouché equivalent on a zero island to a multivariate Lee–Yang container evaluated at $\mathbf{z}_m(s)$. If all active heights satisfy $h_a > 0$, then every zero captured on that island has real part $1/2$.*

Proof. Let a captured zero have readout

$$s = \frac{1}{2} + r + it.$$

If $r > 0$, then

$$|z_a(s)| = e^{-rh_a} < 1$$

for every $a \in A_m$. If $r < 0$, then

$$|z_a(s)| = e^{-rh_a} > 1$$

for every $a \in A_m$. Both cases are excluded by the multivariate Lee–Yang nonvanishing ledger of Definition 9261.22. Since the Rouché equivalence preserves the zero count on the same island, the captured zero cannot have $r \neq 0$. Therefore $r = 0$. □

Definition 9261.24 (Multivariate path Joukowsky target)**.** *For the path independent-set polynomial $I_m(\mathbf{x})$, define the multivariate Joukowsky lift*

$$\mathcal{L}_m(\mathbf{u}) := \prod_{j=1}^{m} (1+u_j)^2 I_m\left(-\frac{u_1}{(1+u_1)^2}, \ldots, -\frac{u_m}{(1+u_m)^2}\right).$$

The displayed lift is a path-skeleton candidate, not itself a stability row. The source row

GoldenIsingContainerBridge

asserts that the zero-relevant zeta or eta golden packet is Rouché equivalent, on the same zero island, to an Ising, Asano, or matching container in which the path skeleton is auditable and the active variables are read as $u_a(s) = e^{-(s-1/2)h_a}$.

Principle 9261.25 (Path hard-core data is a skeleton)**.** *The Zeckendorf path independent-set packet is the finite combinatorial skeleton of the golden bridge, not the final Lee–Yang container. Raw hard-core specialization can produce Dirichlet packets such as* $1 - p^{-s}$*, whose zeros need not have fixed-half readout. The Joukowsky, Asano, Ising, or matching lift is the row that turns the path skeleton into a unit-boundary container on the zeta logarithmic ray. Thus the zeta-facing target is not a direct homotopy to a single variable Fibonacci polynomial, but a Rouché-stable realization inside a height-specialized Lee–Yang container.*

Chapter 9262

Zeta Ray Lee–Yang Container

chapter 9261 records the multivariate Lee–Yang container direction for the golden bridge. The present chapter sharpens that direction to the actual locus available to zeta packets. The variables in a zeta packet are not independent polydisk coordinates; they are synchronized by a single complex height parameter.

Definition 9262.1 (Zeta logarithmic ray)**.** *Let A_m be a finite active-mode set with heights $h_a > 0$. The* zeta logarithmic ray *is the real two-parameter surface*

$$\mathcal{R}_{A_m} := \left\{ \left(e^{-rh_a} e^{-ith_a} \right)_{a \in A_m} : r, t \in \mathbb{R} \right\}.$$

Its off-boundary part is

$$\mathcal{R}^{\mathrm{off}}_{A_m} := \left\{ \left(e^{-rh_a} e^{-ith_a} \right)_{a \in A_m} : r, t \in \mathbb{R},\ r \neq 0 \right\},$$

and its critical phase part is

$$\mathcal{R}^{\mathrm{crit}}_{A_m} := \left\{ \left(e^{-ith_a} \right)_{a \in A_m} : t \in \mathbb{R} \right\}.$$

Definition 9262.2 (Zeta-ray Lee–Yang stability)**.** *A finite packet $\mathcal{Z}_m(\mathbf{u})$ with active heights $(h_a)_{a \in A_m}$ is* zeta-ray Lee–Yang stable *if*

$$\mathcal{Z}_m \left(\left(e^{-rh_a} e^{-ith_a} \right)_{a \in A_m} \right) \neq 0$$

for every $r \neq 0$ and every $t \in \mathbb{R}$. This is weaker than nonvanishing on the whole polydisk and stronger than a one-variable diagonal Lee–Yang assertion.

Definition 9262.3 (Exponential Lee–Yang specialization)**.** *Given a multivariate packet $\mathcal{Z}_m(u_a)_{a \in A_m}$ and heights $h_a > 0$, its* exponential specialization *is the exponential polynomial*

$$\Phi_m(w) := \mathcal{Z}_m \left((e^{-wh_a})_{a \in A_m} \right).$$

The packet is exponential Lee–Yang stable *for the height vector if*

$$\Phi_m(w) = 0 \quad \Longrightarrow \quad \Re(w) = 0.$$

Theorem 9262.4 (Zeta-ray stability gives fixed-half readout)**.** *Let a zero-relevant golden packet be Rouché equivalent on a zero island to a packet $\mathcal{Z}_m$ that is zeta-ray Lee–Yang stable for the same active heights. Then every zero captured on that island has real part* $1/2$.

Proof. Let a captured zero have readout

$$\rho = \frac{1}{2} + r + it.$$

If $r \neq 0$, then its active variables form a point of $\mathcal{R}^{\mathrm{off}}_{A_m}$:

$$\left(e^{-(\rho-1/2)h_a}\right)_{a\in A_m} = \left(e^{-rh_a}e^{-ith_a}\right)_{a\in A_m}.$$

Zeta-ray Lee–Yang stability says that $\mathcal{Z}_m$ has no zero at that point. The Rouché equivalence gives the same zero count on the same island, so the captured zero cannot have $r \neq 0$. Hence $r = 0$. □

Definition 9262.5 (Prime-height Hermite–Biehler row)**.** *The source row*

$$\mathsf{PrimeHeightHB}$$

asserts that the exponential specialization

$$\Phi_m(w) = \mathcal{Z}_m\left((e^{-wh_a})_{a\in A_m}\right)$$

of the prime-compatible Zeckendorf container belongs to a Hermite–Biehler, de Branges, or equivalent exponential-polynomial stability class forcing all zeros of Φ_m in the zero-relevant island to satisfy $\Re(w) = 0$.

Principle 9262.6 (Ray stability is the required container surface)**.** *Full polydisk Lee–Yang non-vanishing is more than the zeta bridge requires. The zeta variables are tied by the same parameter $w = s - 1/2$ and can only move on the logarithmic ray of Definition 9262.1. The relevant replacement for a global multivariate stability claim is therefore*

$$\mathsf{ZetaRayStability} \qquad \textit{or structurally} \qquad \mathsf{PrimeHeightHB}.$$

Together with a same-island capture row

$$\mathsf{PrimePacketToZetaCapture},$$

these rows are sufficient for the no-scale readout of Definition 9262.4.

Principle 9262.7 (Path Joukowsky obstruction)**.** *The path Joukowsky lift is not, by itself, a multivariate Lee–Yang container. For $m = 2$ the path independent-set polynomial is*

$$I_2(x_1, x_2) = 1 + x_1 + x_2.$$

The coordinatewise Joukowsky lift has the form

$$\mathcal{L}_2(u_1, u_2) = (1 + u_1)^2(1 + u_2)^2 - u_1(1 + u_2)^2 - u_2(1 + u_1)^2.$$

After fixing $u_1 = i/2$, this becomes

$$\left(\frac{3}{4} + \frac{i}{2}\right)u_2^2 + \frac{3}{4}u_2 + \left(\frac{3}{4} + \frac{i}{2}\right) = 0.$$

Equivalently,

$$u_2^2 + \left(\frac{9}{13} - \frac{6}{13}i\right) u_2 + 1 = 0.$$

The product of the two roots is 1*, but their sum is not real. Hence the two roots cannot both lie on the unit circle. Since their product has modulus* 1*, one root lies inside the unit disk. Thus the coordinatewise path Joukowsky lift has a zero with both variables inside the unit disk, and the unrestricted multivariate stability claim is not available.*

Definition 9262.8 (Golden–Ising container bridge)**.** *The source row*

GoldenIsingContainerBridge

asserts the following data on a zero-relevant golden island. First, there is a finite ferromagnetic Ising, Asano-stable, or matching-stable container $Z_{G_m}(\mathbf{z})$ *with a Lee–Yang nonvanishing ledger on the required zeta logarithmic ray. Second, the Zeckendorf or Fibonacci path packet is present only as an auditable skeleton inside that container. Third, the zeta golden packet and* $Z_{G_m}(\mathbf{z}(s))$ *have the same island winding, either through a Rouché gap or through a stable homotopy. Fourth, the bridge has no unledgered* $\mathbb{R}_{>0}$ *scale coordinate.*

Definition 9262.9 (Dominant container and Ising homotopy rows)**.** *The Golden–Ising bridge separates into two source rows. The row*

GoldenDominantContainerTheorem

selects the zero-relevant Lee–Yang container branch and proves the boundary gap between that branch and every subdominant contribution. The row

IsingRoucheHomotopy

transports the zeta golden packet to the selected container without crossing zero on the boundary and without losing the Lee–Yang nonvanishing ledger on the zeta logarithmic ray.

Theorem 9262.10 (Golden–Ising bridge gives zeta-ray stability)**.** *If a zero-relevant golden packet has a Golden–Ising container bridge, then it satisfies the hypotheses of Definition 9262.4.*

Proof. The bridge supplies a Lee–Yang container Z_{G_m} with nonvanishing on the off-boundary part of the zeta logarithmic ray. It also supplies a same-island Rouché gap or stable homotopy between the zeta golden packet and $Z_{G_m}(\mathbf{z}(s))$. Thus the captured zero count is read through a zeta-ray Lee–Yang stable packet with the same active heights. These are precisely the hypotheses of Definition 9262.4. □

Principle 9262.11 (Skeleton and container separation)**.** *The Zeckendorf path data supplies ledger uniqueness, finite refinement, and the no-adjacent grammar. It does not supply the full thermodynamic stability row. The Lee–Yang container supplies the unit-boundary or ray-nonvanishing row. The bridge between them is therefore a stability-preserving transport, not a coordinatewise Joukowsky substitution. This separation is the working form of the golden bridge:*

Zeckendorf skeleton + *Lee–Yang container* + *same-island Rouché transport.*

Definition 9262.12 (Prime–Asano–Ising container)**.** *A* prime–Asano–Ising container *for an active golden packet is a finite polynomial $\mathcal{Z}_m(\mathbf{z})$ with the following rows. The finite-volume row says that $\mathcal{Z}_m$ is a ferromagnetic Ising partition function, or is built from Lee–Yang stable factors by Asano-contraction or matching-stability operations. The active-variable row assigns each variable to an audited active height*

$$z_a(s) = e^{-(s-1/2)h_a}, \qquad h_a > 0.$$

The stability row gives Lee–Yang nonvanishing on the required inside/outside regions, or the zeta-ray form of Definition 9262.2. The ledger row requires the heights and active modes to be supplied by the prime or Zeckendorf address ledger rather than by interpolation after the zero island has been selected. The no-scale row forbids any unledgered $\mathbb{R}_{>0}$ gauge in the construction.

Definition 9262.13 (Zeta-to-prime-Asano capture)**.** *An accepted non-trivial completed zeta zero satisfies*

$$\mathsf{ZetaToPrimeAsanoCapture}$$

when there are a golden analytic packet $P_m(s)$, a prime–Asano–Ising container $\mathcal{Z}_m(\mathbf{z})$, and a zero island D_m containing the zero such that either the Rouché gap

$$|P_m(s) - \mathcal{Z}_m(\mathbf{z}(s))| < |P_m(s)| \qquad s \in \partial D_m$$

holds, or a boundary-nonvanishing homotopy carries P_m to $\mathcal{Z}_m(\mathbf{z}(s))$ on the same island. The capture row also requires the Zeckendorf or Fibonacci data to appear only as the internal ruler and source skeleton of the container.

Theorem 9262.14 (Prime–Asano capture gives fixed-half readout)**.** *If an accepted non-trivial completed zeta zero satisfies* ZetaToPrimeAsanoCapture*, then its readout has real part* $1/2$*.*

Proof. The capture row supplies a prime–Asano–Ising container with the same zero-island winding as the golden analytic packet. By Definition 9262.12, this container is zeta-ray Lee–Yang stable for the active heights, with no unledgered real-positive scale gauge. Applying Definition 9262.4 gives real part $1/2$ for every zero captured on that island. □

Principle 9262.15 (Internal ruler is not the final container)**.** *The golden ruler and Zeckendorf ledger remain internal: they select finite modes, supply refinement, and make the source address auditable. They do not by themselves create the no-scale thermodynamic stability row. The final stability row is carried by a prime–Asano–Ising, Hermite–Biehler, or de Branges container, and the bridge obligation is the capture row of Definition 9262.13. Thus the active proof load is not the existence of an internal ruler alone, but the same-island transport from the internally ruled golden packet to a stable container.*

Definition 9262.16 (Polarized golden Lee–Yang container)**.** *Let $L_m(u)$ be the one-variable Fibonacci Lee–Yang lift supplied by the golden ruler, and let $D_m = \deg L_m$. If*

$$L_m(u) = \sum_{k=0}^{D_m} a_k u^k,$$

its polarized container is

$$\mathcal{Z}_m^{\mathrm{pol}}(z_1, \ldots, z_{D_m}) := \sum_{k=0}^{D_m} a_k \binom{D_m}{k}^{-1} e_k(z_1, \ldots, z_{D_m}),$$

where e_k is the k-th elementary symmetric polynomial. It satisfies the diagonal readback

$$\mathcal{Z}_m^{\mathrm{pol}}(u, \ldots, u) = L_m(u).$$

The source row

GoldenPolarizedLeeYangBridge

asserts that this polarized packet is the Lee–Yang container supplied by the golden skeleton, rather than the raw multivariate hard-core path polynomial.

Principle 9262.17 (Polarization supplies the container row)**.** *If the one-variable lift L_m has all nontrivial zeros on the unit circle, then its polarization has the Lee–Yang nonvanishing ledger:*

$$\mathcal{Z}_m^{\mathrm{pol}}(\mathbf{z}) \neq 0 \qquad \text{if } |z_j| < 1 \text{ for all } j,$$

and

$$\mathcal{Z}_m^{\mathrm{pol}}(\mathbf{z}) \neq 0 \qquad \text{if } |z_j| > 1 \text{ for all } j.$$

This is the Grace–Walsh–Szegő polarization route from the reliable one-variable Fibonacci circle theorem to a genuine multivariate Lee–Yang container. It is distinct from coordinatewise Joukowsky substitution into the hard-core path polynomial, which is excluded by Definition 9262.7.

Definition 9262.18 (Zeta-to-polarized golden container Rouché row)**.** *The source row*

ZetaToPolarizedGoldenContainerRouche

asserts that, on a zero-relevant island D_m, the zeta golden packet $P_m^{\zeta}(s)$ admits active heights $h_1, \ldots, h_{D_m} > 0$ and a polarized golden container such that

$$z_j(s) = e^{-(s-1/2)h_j}$$

and

$$\left| P_m^{\zeta}(s) - \mathcal{Z}_m^{\mathrm{pol}}(z_1(s), \ldots, z_{D_m}(s)) \right| < \left| \mathcal{Z}_m^{\mathrm{pol}}(z_1(s), \ldots, z_{D_m}(s)) \right|$$

for $s \in \partial D_m$. The row also requires the error term to be subdominant in the ordinary Rouché sense and not an unledgered real-positive scale reservoir.

Theorem 9262.19 (Polarized golden capture gives fixed-half readout)**.** *If an accepted non-trivial completed zeta zero satisfies* ZetaToPolarizedGoldenContainerRouche*, then its readout has real part* $1/2$*.*

Proof. The polarized container has the Lee–Yang nonvanishing ledger by Definition 9262.17. If a captured readout $s = 1/2 + r + it$ had $r > 0$, then every active variable $z_j(s) = e^{-rh_j}e^{-ith_j}$ would lie inside the unit disk. If $r < 0$, every active variable would lie outside the unit disk. Both cases are excluded by the polarized container. The Rouché row transports the zero count between P_m^{ζ} and the polarized container on the same island, so the captured zero cannot have $r \neq 0$. □

Definition 9262.20 (Polarized golden cofinality)**.** *The source row*

PolarizedGoldenCofinality

asserts that every accepted zero-relevant zeta packet is, after passing to a sufficiently fine golden level, covered by Definition 9262.18. This is the cofinal version of the bridge: the polarized golden containers need not equal the zeta packet, but they must dominate the same zero islands with strict Rouché margins.

Definition 9262.21 (Source-side golden Lee–Yang closure)**.** *The source row*

GoldenLeeYangClosureOfXi

asserts that the completed zeta function $\Xi(s)$ is obtained as a source-side limit of internally generated polarized golden Lee–Yang containers. Concretely, there are functions

$$C_m(s) = \mathcal{Z}_m^{\mathrm{pol}}\left(e^{-(s-1/2)h_{m,1}}, \dots, e^{-(s-1/2)h_{m,d_m}}\right)$$

such that each $h_{m,j} > 0$, each C_m is supplied before any zero island is selected, each C_m has no zero in

$$\Re(s) \neq \frac{1}{2},$$

and $C_m \to \Xi$ uniformly on compact sets with enough boundary Rouché control to prevent limit zeros from entering the off-line region.

Theorem 9262.22 (Golden Lee–Yang closure gives fixed-half readout)**.** *If* GoldenLeeYangClosureOfXi *holds, then Ξ has no zero with real part different from $1/2$.*

Proof. Assume that Ξ has a zero ρ with

$$\Re(\rho) \neq \frac{1}{2}.$$

Choose a small disk D around ρ whose closure does not meet the line $\Re(s) = 1/2$ and whose boundary has no zero of Ξ. Then

$$M := \min_{s \in \partial D} |\Xi(s)| > 0.$$

Uniform convergence gives, for sufficiently large m,

$$|C_m(s) - \Xi(s)| < \frac{M}{2} \qquad s \in \partial D.$$

Hence $|C_m(s)| > M/2$ on ∂D, and therefore

$$|\Xi(s) - C_m(s)| < |C_m(s)| \qquad s \in \partial D.$$

By Rouché's theorem, Ξ and C_m have the same number of zeros in D. But C_m has no zero in the off-line disk D, contradiction. Thus no such ρ exists. □

Principle 9262.23 (Local capture is not the final bridge)**.** *The local rows*

ZetaToPrimeAsanoCapture ZetaToPolarizedGoldenContainerRouche

are useful only as finite-island consequences of a source-side construction. If they are postulated separately around each candidate zero, the bridge risks encoding the desired exclusion into the local choice of container. The non-circular target is the prior container tower of Definition 9262.21: the container sequence is constructed from the internal ruler and polarization mechanism first, and zero exclusion is then read by Rouché.

Chapter 9263

Squarefree Prime Polarized Bridge

chapter 9262 isolates the polarized golden Lee–Yang container as the stable object in the golden route. The present chapter records the arithmetic form of that container after prime-height specialization. The point is not to choose a Lee–Yang container around a zero island, but to identify a source-side prime ledger through which a zeta packet could be compared with the polarized container.

Definition 9263.1 (Squarefree prime address set)**.** *Let $p_1, \dots, p_D$ be distinct primes. The squarefree prime address set generated by this window is*

$$\mathcal{N}_D^{\mathrm{sq}} := \left\{ \prod_{j \in S} p_j : S \subseteq \{1, \dots, D\} \right\}.$$

For $n \in \mathcal{N}_D^{\mathrm{sq}}$, let

$$\omega_D(n) := \#\{j : p_j \mid n\}.$$

Definition 9263.2 (Squarefree prime packet)**.** *A* squarefree prime packet *on the prime window $p_1, \dots, p_D$ is a Dirichlet packet of the form*

$$S_D(s) := \sum_{n \in \mathcal{N}_D^{\mathrm{sq}}} b(n) n^{1/2-s}.$$

The coefficients $b(n)$ are part of the source ledger. They are not chosen after a zero island is selected unless a separate capture row records the choice and its boundary margin.

Definition 9263.3 (Prime-height polarized readback)**.** *Let*

$$\mathcal{Z}^{\mathrm{pol}}(z_1, \dots, z_D) = \sum_{k=0}^{D} a_k \binom{D}{k}^{-1} e_k(z_1, \dots, z_D)$$

be the polarization of a one-variable Lee–Yang seed. Its prime-height readback *is obtained by setting*

$$z_j(s) = e^{-(s-1/2)\log p_j}.$$

Then

$$\mathcal{Z}^{\mathrm{pol}}(z_1(s), \dots, z_D(s)) = \sum_{S \subseteq \{1,\dots,D\}} a_{|S|} \binom{D}{|S|}^{-1} \left(\prod_{j \in S} p_j \right)^{1/2-s}.$$

Thus the prime-height readback is a squarefree prime packet whose coefficient depends only on the prime-support cardinality.

Principle 9263.4 (Polarization enters the prime ledger)**.** *The polarized golden container is not merely a formal multiaffine replacement for a one-variable seed. Under the specialization $h_j = \log p_j$, every elementary symmetric term becomes a squarefree prime monomial:*

$$e^{-(s-1/2)\sum_{j\in S}\log p_j} = \left(\prod_{j\in S} p_j\right)^{1/2-s}.$$

Consequently, the bridge from the zeta packet to the polarized Lee–Yang container factors through the arithmetic surface

$$\textit{zeta packet} \quad \rightsquigarrow \quad \textit{squarefree prime packet} \quad \rightsquigarrow \quad \textit{polarized Lee–Yang container.}$$

The first arrow is analytic and zero-island dependent; the second arrow is algebraic or boundary-small once the prime window and seed coefficients have been fixed.

Definition 9263.5 (Squarefree Rouché dominance)**.** *Let $P_m^\zeta(s)$ be a zero-relevant zeta golden packet on a zero island D_m. The packet satisfies*

$$\mathsf{SquarefreeRoucheDominance}$$

if there exist a prime window $p_1, \dots, p_D$ and a squarefree prime packet $S_D(s)$ such that

$$|P_m^\zeta(s) - S_D(s)| < |S_D(s)| \qquad s \in \partial D_m.$$

Equivalently, P_m^ζ and S_D have the same zero count in D_m by Rouché's theorem. The packet S_D must be supplied by a source ledger rather than by an unrecorded interpolation around the candidate zero.

Definition 9263.6 (Polarized coefficient matching)**.** *A squarefree prime packet S_D satisfies*

$$\mathsf{PolarizedCoefficientMatching}$$

if there is a one-variable Lee–Yang seed $L(u) = \sum_{k=0}^{D} a_k u^k$ such that either

$$b(n) = a_{\omega_D(n)} \binom{D}{\omega_D(n)}^{-1} \qquad n \in \mathcal{N}_D^{\mathrm{sq}},$$

or, for the corresponding polarized container $\mathcal{Z}^{\mathrm{pol}}$, the boundary estimate

$$|S_D(s) - \mathcal{Z}^{\mathrm{pol}}(z_1(s), \dots, z_D(s))| < |\mathcal{Z}^{\mathrm{pol}}(z_1(s), \dots, z_D(s))| \qquad s \in \partial D_m$$

holds on the zero island under the prime-height specialization.

Theorem 9263.7 (Squarefree bridge gives polarized transport)**.** *Let a zero-relevant zeta golden packet P_m^ζ on an island D_m satisfy* SquarefreeRoucheDominance *through S_D. If S_D satisfies* PolarizedCoefficientMatc *through a polarized golden Lee–Yang container, then P_m^ζ and the prime-height readback of that container have the same zero-island winding on D_m.*

Proof. The first row gives a strict Rouché inequality between P_m^ζ and S_D on ∂D_m, hence the same zero count on D_m. The second row either identifies S_D with the prime-height readback of a polarized container, or supplies a second strict Rouché inequality between S_D and that readback on the same boundary. Chaining the two boundary nonvanishing comparisons gives the same zero-island winding for P_m^ζ and the polarized Lee–Yang container. □

Corollary 9263.8 (Squarefree bridge gives fixed-half readout)**.** *Under the hypotheses of Definition 9263.7, every zero captured by the island D_m has readout with real part* $1/2$*.*

Proof. By Definition 9263.7, the zeta packet and the prime-height polarized container have the same zero-island winding on D_m. If a captured readout had real part different from $1/2$, then all prime-height variables would lie strictly inside the unit disk or strictly outside it. The polarized container has no zero on either region by Definition 9262.17. Hence the container has zero winding on that off-line island, and the same is true of P_m^ζ, contradicting capture. Therefore the readout has real part $1/2$. □

Principle 9263.9 (Analytic load of the squarefree bridge)**.** *The polarization and coefficient-matching rows are structural: after a one-variable Lee–Yang seed and a prime window are fixed, their prime-height readback is a squarefree prime packet by construction. The analytic burden is the dominance row*

SquarefreeRoucheDominance.

It asks whether the zero-relevant zeta packet can be compared, with a strict boundary gap, to a packet supported on squarefree products of a finite prime window. Prime powers and the non-squarefree tail must therefore enter only through a controlled remainder, a reciprocal Euler-product contour row, or an explicit-formula tail ledger. Without such a ledger, squarefree support is only a formal address surface and does not supply the local Rouché bridge.

Definition 9263.10 (Stable squarefree prime approximation of Xi)**.** *The source row*

StableSquarefreePrimeApproximationOfXi

asserts that there is a source-side sequence of functions

$$C_m(s) = \sum_{n \in \mathcal{N}_{D_m}^{\mathrm{sq}}} b_m(n) n^{1/2-s}$$

such that each C_m is the prime-height readback of a polarized Lee–Yang seed in the sense of Definition 9263.3, and

$$C_m \to \Xi$$

uniformly on compact subsets, with the ordinary contour margins obtained from compact-uniform convergence on each boundary curve. The construction of C_m is fixed by the source ledger before any zero island is selected.

Theorem 9263.11 (Stable squarefree approximation gives golden closure)**.** *If* StableSquarefreePrimeApproximationO *holds, then the source row* GoldenLeeYangClosureOfXi *of Definition 9262.21 holds.*

Proof. Each C_m is a prime-height readback of a polarized Lee–Yang seed. Hence, by Definition 9262.17, it has no zeros when all prime-height variables lie strictly inside the unit disk or strictly outside it. For

$$s = \frac{1}{2} + r + it$$

with $r \neq 0$, the variables

$$e^{-(s-1/2)\log p_j}$$

all have modulus $e^{-r \log p_j}$, hence are all strictly inside the unit disk when $r > 0$ and all strictly outside it when $r < 0$. Therefore each C_m has no zero off the critical line. The stated compact-uniform convergence supplies the boundary Rouché margins on every fixed zero-exclusion contour, and the construction is source-side. These are precisely the closure rows of Definition 9262.21. □

Corollary 9263.12 (Stable squarefree approximation gives fixed-half readout)**.** *If* StableSquarefreePrimeApproxima *holds, then* Ξ *has no zero with real part different from* $1/2$*.*

Proof. Apply Definition 9263.11 and then Definition 9262.22. □

Principle 9263.13 (Source-side squarefree obstruction)**.** *The row*

StableSquarefreePrimeApproximationOfXi

is stronger and less circular than local squarefree dominance. It does not compare a packet to a container chosen near a zero; it asks for a single source-side compact-uniform container tower. Its difficulty is arithmetic: squarefree prime products naturally occur in Euler factors for reciprocal or signed product packets, while Ξ *contains the completed zeta numerator. Thus prime powers, reciprocal contours, logarithmic derivatives, or explicit-formula tails must be accounted for by a displayed ledger before the source-side closure row can be claimed.*

Definition 9263.14 (Squarefree closure companion rows)**.** *The squarefree source-side route has three companion rows. The row*

$\mathsf{StableSquarefreePrimeApproximationOfXi}^{-1}$

uses stable squarefree packets to control the reciprocal completed function or an argument-principle contour, so that zeros of Ξ *are recorded as poles in a no-scale ledger. The row*

LeeYangWeilPositivity

translates Lee–Yang stability of the squarefree packets into a Weil positivity inequality through the explicit formula. The row

LeeYangToHermiteBiehlerSourceContainer

transports the polarized squarefree packets into a Hermite–Biehler or de Branges source container. These rows are alternatives to direct compact-uniform approximation of Ξ*; none is a consequence of local Rouché capture alone.*

Chapter 9264

Xi Jensen Internal Container

chapter 9262 and chapter 9263 formulate Lee–Yang containers as source-side packets to which zeta data may be transported. The Jensen route records a stricter internal alternative: the finite containers are obtained from the completed Xi function itself. The route is conditional. It does not discharge the hyperbolicity obligation, but it places that obligation on an intrinsic finite-resolution surface rather than on an externally selected thermodynamic model.

Definition 9264.1 (Xi Jensen coefficient row)**.** *Let $F(T)$ be the even entire readout of the completed Xi function in a real Taylor coordinate,*

$$F(T) = \sum_{k\geq 0} \gamma(k) \frac{T^k}{k!}.$$

The coefficient sequence γ is the source ledger for the Jensen container family. No zero island or auxiliary Lee–Yang packet is used to define these coefficients.

Definition 9264.2 (Jensen section)**.** *For integers $d, n \geq 0$, the Jensen section of degree d and shift n attached to γ is*

$$J_\gamma^{d,n}(X) := \sum_{j=0}^{d} \binom{d}{j} \gamma(n+j) X^j.$$

It is the finite observation at resolution d and source position n.

Definition 9264.3 (Jensen no-demon row)**.** *A Jensen section is* no-demon *when it is hyperbolic, i.e. when all of its zeros are real. The full Jensen no-demon row is*

$$\mathsf{JensenNoDemon}_\Xi := \forall d, n \geq 0, \quad J_\gamma^{d,n} \text{ is hyperbolic.}$$

This row is the finite-container form of the Laguerre–Pólya obligation for the Xi readout.

Principle 9264.4 (Jensen–Pólya finite-container criterion)**.** *The classical Jensen–Pólya criterion identifies* $\mathsf{JensenNoDemon}_\Xi$ *as the finite-container route to placing the Xi readout in the Laguerre–Pólya class. For the RH vision this means that the zero-location burden can be expressed as hyperbolicity of the internally generated Jensen sections. The criterion is used here only as a conditional bridge: the chapter does not assert the hyperbolicity row.*

Theorem 9264.5 (Jensen shift derivative identity)**.** *For all $d, n \geq 0$,*

$$\frac{d}{dX} J_\gamma^{d+1,n}(X) = (d+1) J_\gamma^{d,n+1}(X).$$

Proof. By definition,

$$J_\gamma^{d+1,n}(X) = \sum_{j=0}^{d+1} \binom{d+1}{j} \gamma(n+j) X^j.$$

Differentiating gives

$$\frac{d}{dX} J_\gamma^{d+1,n}(X) = \sum_{j=1}^{d+1} j \binom{d+1}{j} \gamma(n+j) X^{j-1}.$$

Set $k = j - 1$. Since

$$(k+1)\binom{d+1}{k+1} = (d+1)\binom{d}{k},$$

the derivative becomes

$$(d+1) \sum_{k=0}^{d} \binom{d}{k} \gamma(n+1+k) X^k = (d+1) J_\gamma^{d,n+1}(X).$$

□

Corollary 9264.6 (Derivative propagation of no-demon rows)**.** *If $J_\gamma^{d+1,n}$ is hyperbolic, then $J_\gamma^{d,n+1}$ is hyperbolic.*

Proof. The derivative of a real-rooted polynomial is real-rooted. By Definition 9264.5, $J_\gamma^{d,n+1}$ is a nonzero scalar multiple of such a derivative. □

Definition 9264.7 (Top-row Jensen no-demon)**.** *The source row*

$$\mathsf{TopRowJensenNoDemon}_\Xi$$

asserts that

$$J_\gamma^{D,0}$$

is hyperbolic for every $D \geq 0$.

Theorem 9264.8 (Top row supplies all Jensen sections)**.** *If* $\mathsf{TopRowJensenNoDemon}_\Xi$ *holds, then* $\mathsf{JensenNoDemon}_\Xi$ *holds.*

Proof. For fixed $d, n \geq 0$, repeated use of Definition 9264.5 gives

$$\frac{d^n}{dX^n} J_\gamma^{d+n,0}(X) = \frac{(d+n)!}{d!} J_\gamma^{d,n}(X).$$

The top-row hypothesis gives hyperbolicity of $J_\gamma^{d+n,0}$. Repeated differentiation preserves hyperbolicity, hence $J_\gamma^{d,n}$ is hyperbolic. □

Definition 9264.9 (Cayley circularization of a Jensen section)**.** *For a degree d Jensen section, define its Cayley circularization by*

$$\mathcal{L}_\gamma^{d,n}(u) := (1-u)^d J_\gamma^{d,n}\left(i\frac{1+u}{1-u}\right).$$

The map $u \mapsto i(1+u)/(1-u)$ sends the unit circle, away from $u = 1$, to the real line.

Theorem 9264.10 (Hyperbolic Jensen sections give Lee–Yang circles)**.** *If $J_\gamma^{d,n}$ is hyperbolic, then every zero of $\mathcal{L}_\gamma^{d,n}$ lies on the unit circle.*

Proof. Let $\mathcal{L}_\gamma^{d,n}(u) = 0$. If $u \neq 1$, then

$$X = i\frac{1+u}{1-u}$$

is a zero of $J_\gamma^{d,n}$, hence $X \in \mathbb{R}$. The inverse Cayley transform sends real X to $|u| = 1$. If $u = 1$ occurs, it is the point added by the circularizing factor and still lies on the unit circle. □

Definition 9264.11 (Polarized Jensen Lee–Yang container)**.** *Let*

$$\mathcal{L}_\gamma^{D,0}(u) = \sum_{k=0}^{D} a_k u^k.$$

The polarized Jensen Lee–Yang container is

$$\mathcal{C}_\gamma^{D}(z_1, \ldots, z_D) := \sum_{k=0}^{D} a_k \binom{D}{k}^{-1} e_k(z_1, \ldots, z_D).$$

It is internal because both the seed polynomial and the polarization degree are read from the Jensen section of Xi.

Principle 9264.12 (Jensen containers are internal no-scale containers)**.** *If* $\mathsf{TopRowJensenNoDemon}_\Xi$ *holds, then every top-row Jensen section yields, by Cayley circularization and polarization, a multivariate Lee–Yang container with no zeros when all variables lie strictly inside the unit disk or strictly outside it. Thus the container row is generated from Xi itself. The remaining source obligation is not the construction of an external Ising approximation, but the hyperbolicity of the top-row Jensen sections.*

Definition 9264.13 (Golden Jensen refinement operator)**.** *The source row*

$$\mathsf{GoldenJensenRefinementOperator}$$

asserts the existence of a Fibonacci-indexed family of stability preserving transformations

$$\mathcal{R}_m : J_\gamma^{F_m,0} \rightsquigarrow J_\gamma^{F_{m+1},0}$$

which are generated from the Xi coefficient ledger and the internal golden ruler, and which preserve hyperbolicity. Such a row would turn the top-row no-demon obligation into a golden refinement induction.

Principle 9264.14 (Top-row location of the Jensen obstruction)**.** *The derivative identity removes no-demon propagation from the shift direction: every shifted Jensen section is a normalized derivative of a top-row section. The irreducible obstruction in the Jensen route is therefore the source row*

$$\mathsf{TopRowJensenNoDemon}_\Xi,$$

or, more constructively, the refinement row

$$\mathsf{GoldenJensenRefinementOperator}.$$

These rows are not consequences of the golden ruler alone; they require a stability mechanism for the Xi coefficient ledger.

Chapter 9265

Global Weil Positive Container Tower

chapter 9263 records the squarefree prime surface for polarized Lee–Yang containers, and chapter 9264 records the Jensen hyperbolicity surface generated from Xi itself. The common global target behind these routes is a Weil-positive tower: finite positive containers whose limit is the Weil quadratic form supplied by the explicit formula. This chapter records that target as the non-local form of the no-scale route.

Definition 9265.1 (Multiplicative test row)**.** *Let $\mathcal{A}$ be a source test space on $\mathbb{R}_+^\times$ stable under the operations required by the explicit formula. For $f \in \mathcal{A}$, its Mellin readout is*

$$\widehat{f}(s) := \int_0^\infty f(x) x^s \frac{dx}{x},$$

whenever this integral is read directly, and otherwise by the regularized test-row convention of the explicit formula. In logarithmic coordinates $x = e^u$, with $\phi(u) = f(e^u)$, this is the Laplace–Fourier readout

$$\widehat{f}(s) = \int_{-\infty}^\infty \phi(u) e^{su}\, du.$$

Definition 9265.2 (Weil quadratic form row)**.** *The Weil quadratic form attached to the completed zeta zero ledger is the regularized pairing*

$$\mathcal{W}(f) := \sum_\rho \widehat{f}(\rho)\, \overline{\widehat{f}(1-\overline{\rho})},$$

where ρ ranges over the completed zeta zero ledger with the standard symmetry and multiplicity conventions of the explicit formula. The displayed formula records the zero-side contribution; the regularization and archimedean rows are part of the Weil-form source ledger.

Definition 9265.3 (Global Weil positivity)**.** *The source row*

$$\mathsf{GlobalWeilPositivity}$$

asserts that

$$\mathcal{W}(f) \geq 0 \qquad (f \in \mathcal{A}).$$

It is the global test-function version of the no-scale condition: no displayed test row may extract negative energy from the zero ledger.

Principle 9265.4 (Weil positivity as the no-scale global target)**.** *The local no-scale route reads an off-half parameter as a radial defect*

$$e^{-(s-1/2)h} = e^{-rh}e^{-ith}.$$

Finite packets express this as a scale-energy row. The global Weil route expresses the same obstruction as a test-function positivity row: an off-half mirror pair would allow a test readout that separates the two mirror points and contributes a negative term to Definition 9265.2. Thus the thermodynamic no-demon condition and the explicit-formula positivity condition have a common target, namely GlobalWeilPositivity*.*

Definition 9265.5 (Finite Weil-positive container)**.** *A* finite Weil-positive container *is a pair*

$$(\mathcal{A}_N, \mathcal{W}_N)$$

where $\mathcal{A}_N \subseteq \mathcal{A}$ is a finite displayed test space and

$$\mathcal{W}_N : \mathcal{A}_N \to \mathbb{R}$$

is a positive semidefinite quadratic row:

$$\mathcal{W}_N(f) \geq 0 \qquad (f \in \mathcal{A}_N).$$

The container is source-side when $\mathcal{A}_N$ and $\mathcal{W}_N$ are generated before any zero island is selected and carry no unledgered $\mathbb{R}_{>0}$ scale coordinate.

Definition 9265.6 (Cofinal Weil-positive container tower)**.** *The source row*

CofinalWeilPositiveContainerTower

asserts the existence of finite Weil-positive containers

$$(\mathcal{A}_N, \mathcal{W}_N)_{N\geq 1}$$

with the following rows.

The nesting row is

$$\mathcal{A}_N \subseteq \mathcal{A}_{N+1}.$$

The cofinality row says that

$$\bigcup_N \mathcal{A}_N$$

is dense in the test space used by Definition 9265.2. The compatibility row says that the restrictions of later quadratic forms to earlier test spaces agree with earlier forms up to a displayed error ledger tending to zero. The limit row says that for every test function f and every admissible cofinal approximation $f_N \in \mathcal{A}_N$ with $f_N \to f$,

$$\mathcal{W}_N(f_N) \to \mathcal{W}(f).$$

Theorem 9265.7 (Cofinal Weil-positive tower gives global positivity)**.** *If* CofinalWeilPositiveContainerTower *holds, then* GlobalWeilPositivity *holds.*

Proof. Let $f \in \mathcal{A}$. By cofinality choose admissible $f_N \in \mathcal{A}_N$ with $f_N \to f$. Finite positivity gives

$$\mathcal{W}_N(f_N) \geq 0$$

for every N. The limit row gives

$$\mathcal{W}_N(f_N) \to \mathcal{W}(f).$$

The limit of nonnegative real numbers is nonnegative, so

$$\mathcal{W}(f) \geq 0.$$

Since f was arbitrary, GlobalWeilPositivity holds. □

Definition 9265.8 (Weil criterion consumption row)**.** *The source row*

WeilCriterionConsumesGlobalPositivity

is the classical criterion that consumes GlobalWeilPositivity *and returns the fixed-half zero ledger for the completed zeta function. In this vision it is treated as a named bridge row, separate from the construction of the finite positive tower.*

Theorem 9265.9 (Finite Weil-positive tower gives fixed-half readout)**.** *Assume* CofinalWeilPositiveContainerTower *and* WeilCriterionConsumesGlobalPositivity*. Then the completed zeta zero ledger has fixed-half readout.*

Proof. By Definition 9265.7, the tower gives GlobalWeilPositivity. The consumption row then returns the fixed-half zero ledger. □

Principle 9265.10 (Finite container routes as Weil-tower candidates)**.** *The Lee–Yang, Jensen, golden-ruler, squarefree-prime, and quartic endpoint routes are best read as candidate constructions of the finite rows in Definition 9265.6. A Lee–Yang packet supplies finite nonvanishing and stability; a Jensen section supplies finite hyperbolicity; a golden ruler supplies source-side cofinal refinement; a squarefree prime packet supplies arithmetic ledgering; and a quartic endpoint model supplies a finite dominance and Rouché laboratory. None of these finite rows is by itself the global argument. The global argument is the cofinal passage from finite positivity to* GlobalWeilPositivity*.*

Definition 9265.11 (Scale-closure to Weil positivity row)**.** *The source row*

ZetaUnitaryScaleClosureToWeilPositivity

asserts that the bounded-energy, no-unledgered-scale, and cofinal-growth rows of the zeta unitary scale route assemble into finite Weil-positive containers compatible with Definition 9265.2. It is the bridge from local radial-defect exclusion to the global positivity row.

Principle 9265.12 (Final non-local proof obligation)**.** *The non-local proof obligation is not to build a stable packet around a candidate zero. It is to construct a source-side cofinal tower of finite positive quadratic rows. In symbols, the active target is*

$$\mathsf{CofinalWeilPositiveContainerTower} \Longrightarrow \mathsf{GlobalWeilPositivity} \Longrightarrow \textit{fixed-half zero readout.}$$

The first implication is the finite-container reduction proved in this chapter; the second is delegated to the Weil criterion consumption row.

Definition 9265.13 (Offline mirror pair)**.** *An* offline mirror pair *in the completed zeta zero ledger is a pair*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma,$$

with $r \neq 0$, related by

$$\rho_- = 1 - \overline{\rho_+}.$$

It is the zero-side expression of a radial scale split: the two points have the same phase height and opposite real scale directions.

Definition 9265.14 (Diagonal negative probe)**.** *A* diagonal negative probe *for an offline mirror pair is a test row $f \in \mathcal{A}$ whose Mellin readout separates the pair by opposite signs,*

$$\widehat{f}(\rho_+) = 1, \qquad \widehat{f}(\rho_-) = -1,$$

while all remaining zero-side contributions are controlled by the regularized tail ledger of Definition 9265.2. The row is diagonal because it reads precisely the mirror coordinate that a local scale selector would try to hide.

Principle 9265.15 (Offline mirror pairs create negative Weil directions)**.** *If an offline mirror pair admits a diagonal negative probe, then the pair contributes*

$$\widehat{f}(\rho_+)\overline{\widehat{f}(\rho_-)} + \widehat{f}(\rho_-)\overline{\widehat{f}(\rho_+)} = -2$$

to the zero-side Weil pairing. With the remaining zero contributions made smaller by the displayed tail ledger, this gives a negative direction for $\mathcal{W}$. Hence an off-half mirror pair is not merely a local radial defect; it is a global obstruction to GlobalWeilPositivity*.*

Definition 9265.16 (No diagonal negative probe)**.** *The row*

NoDiagonalNegativeProbe

asserts that the Weil quadratic form has no diagonal negative direction: there is no test row $f \in \mathcal{A}$ with

$$\mathcal{W}(f) < 0.$$

Thus NoDiagonalNegativeProbe *is the same positivity surface as* GlobalWeilPositivity*, but with emphasis on the diagonal obstruction created by an offline mirror pair.*

Principle 9265.17 (Halting boundary forbids universal local closure tests)**.** *Definition 9336.11 and Definition 9336.13 rule out a universal classifier deciding closure of all sufficiently expressive theorem routes. The RH zero route must therefore not be framed as a global local decider that inspects every zero box and declares it legal or illegal. Such a decider would be a theorem-route closure oracle at the certificate stratum. The diagonal-safe alternative is a source-side positive tower: each finite container is positive by construction, and the global no-negative-probe row is read only after cofinal passage.*

Definition 9265.18 (Diagonal-safe Weil tower)**.** *A cofinal Weil-positive container tower is* diagonal-safe *when each finite positivity row is supplied by internal structure, for example by a displayed factorization*

$$\mathcal{W}_N(f) = \|T_N f\|^2$$

or by a positive kernel row

$$\mathcal{W}_N(f) = \langle K_N f, f\rangle, \qquad K_N \geq 0,$$

and not by a procedure that searches for future negative probes and repairs the container after they are found. The corresponding source row is

$$\mathsf{DiagonalSafeWeilTower}.$$

Theorem 9265.19 (Diagonal-safe tower gives global positivity)**.** *If* DiagonalSafeWeilTower *holds, then* GlobalWeilPositivity *holds.*

Proof. A diagonal-safe tower is, in particular, a cofinal Weil-positive container tower. Apply Definition 9265.7. □

Principle 9265.20 (Diagonal-safe final target)**.** *The diagonal role of the halting boundary is negative and structural: it forbids a universal local legality decider. The diagonal role of the Weil form is positive and global: any off-half mirror pair would expose a negative probe. The compatible proof target is therefore*

$$\mathsf{DiagonalSafeWeilTower} \Longrightarrow \mathsf{GlobalWeilPositivity} \Longrightarrow \textit{fixed-half zero readout.}$$

This target is source-side and non-local. It asks for a cofinal family of internally positive finite containers rather than a decision procedure over all zero boxes.

Definition 9265.21 (Li–Weil coefficient row)**.** *Let $\xi(s)$ be the completed zeta function used by the zero ledger. The n-th Li–Weil coefficient is the source-side scalar*

$$\lambda_n := \frac{1}{(n-1)!}\,\frac{d^n}{ds^n}\left[s^{n-1}\log\xi(s)\right]\Big|_{s=1}.$$

Equivalently, when read through the regularized zero ledger,

$$\lambda_n = \sum_{\rho}\left[1-\left(1-\frac{1}{\rho}\right)^n\right].$$

Definition 9265.22 (Li–Weil finite negative witness)**.** *A* Li–Weil finite negative witness *is a positive integer n with*

$$\lambda_n < 0.$$

The row is finite: it is indexed by one displayed natural number rather than by an open-ended search through certificate towers.

Definition 9265.23 (Global Li–Weil positivity)**.** *The source row*

$$\mathsf{GlobalLiWeilPositivity}$$

asserts that

$$\lambda_n \geq 0 \qquad (n \geq 1).$$

Definition 9265.24 (Li criterion consumption row)**.** *The source row*

$$\mathsf{LiCriterionConsumesGlobalLiPositivity}$$

is the classical bridge consuming GlobalLiWeilPositivity *and returning fixed-half zero readout. It is used here as a criterion row, not as a construction of the coefficients' positivity.*

Principle 9265.25 (Li–Weil reduction gives finite bad witnesses)**.** *The Li–Weil row gives a concrete form of the diagonal negative-probe principle. If the fixed-half zero readout fails, then the Li criterion consumption row presents a finite negative witness*

$$\exists n \geq 1, \qquad \lambda_n < 0.$$

Thus an off-half defect is not allowed to remain an indefinitely delayed failure of an infinite tower: it must project to a finite Li–Weil coefficient. This is the finite-witness refinement of Definition 9265.16.

Definition 9265.26 (Source-side Li positivity certificate)**.** *A source-side positivity certificate for λ_n consists of internally generated data, such as a Hilbert space row and vector*

$$(\mathcal{H}_n, \Psi_n),$$

with

$$\lambda_n = \|\Psi_n\|^2_{\mathcal{H}_n},$$

or a finite positive-kernel decomposition whose displayed cancellation ledger leaves a nonnegative remainder. The certificate must be generated from the prime, phase, solenoid, golden, Lee–Yang, or explicit-formula source rows; it may not be obtained by searching for negative witnesses and repairing them afterward.

Definition 9265.27 (Source global Li positivity)**.** *The source row*

$$\mathsf{SourceGlobalLiPositivity}$$

asserts that every λ_n has a source-side positivity certificate in the sense of Definition 9265.26.

Theorem 9265.28 (Source Li positivity gives fixed-half readout)**.** *Assume* SourceGlobalLiPositivity *and* LiCriterionConsumesGlobalLiPositivity*. Then the completed zeta zero ledger has fixed-half readout.*

Proof. The source-side certificates give $\lambda_n \geq 0$ for every n, hence GlobalLiWeilPositivity. The Li criterion consumption row then returns the fixed-half zero ledger. □

Definition 9265.29 (Golden Li positivity limit)**.** *The source row*

$$\mathsf{GoldenLiPositivityLimit}$$

asserts that there are source-side golden Lee–Yang containers C_m with finite Li coefficients

$$\lambda_{n,m} := \frac{1}{(n-1)!} \frac{d^n}{ds^n} \left[s^{n-1} \log C_m(s) \right] \Big|_{s=1}$$

such that $\lambda_{n,m} \geq 0$ for all n, m, and for each fixed n,

$$\lambda_{n,m} \to \lambda_n.$$

Theorem 9265.30 (Golden Li positivity limit gives global Li positivity)**.** *If* GoldenLiPositivityLimit *holds, then* GlobalLiWeilPositivity *holds.*

Proof. Fix $n \geq 1$. The finite rows give $\lambda_{n,m} \geq 0$ for every m. Taking the displayed limit in m gives $\lambda_n \geq 0$. Since n was arbitrary, GlobalLiWeilPositivity holds. □

Principle 9265.31 (Li route separates stability from convergence)**.** *The Li–Weil refinement separates the remaining analytic load into two source rows. The finite stability row requires the finite containers C_m to have fixed-half zero readout, which yields $\lambda_{n,m} \geq 0$ by the usual Li coefficient computation for critical-line zero multisets. The convergence row requires enough log-derivative convergence to pass*

$$\lambda_{n,m} \to \lambda_n$$

for every fixed n. In named form the open source obligations are

$$\mathsf{FiniteGoldenLeeYangStability} \qquad \textit{and} \qquad \mathsf{LogDerivativeConvergence}.$$

Chapter 9266

Li Time Negative Witness

Definition 9265.22 records the finite Li–Weil witness produced by the Li criterion consumption row. This chapter gives the geometric readout of that witness. The Li coordinate turns the fixed-half line into the unit circle; off-half displacement becomes radial scale, and the discrete Li time n amplifies that radial component into a finite negative-energy row.

Definition 9266.1 (Li time coordinate)**.** *For a nontrivial zero row ρ of the completed zeta function, define*

$$z(\rho) := 1 - \frac{1}{\rho} = \frac{\rho - 1}{\rho}.$$

Theorem 9266.2 (Li coordinate reads the fixed-half circle)**.** *For $\rho = \sigma + i\gamma$,*

$$|z(\rho)| = 1$$

if and only if

$$\sigma = \frac{1}{2}.$$

Proof. By definition,

$$|z(\rho)| = \left|\frac{\rho - 1}{\rho}\right| = \frac{|\rho - 1|}{|\rho|}.$$

Thus $|z(\rho)| = 1$ exactly when $|\rho - 1| = |\rho|$. Squaring gives

$$(\sigma - 1)^2 + \gamma^2 = \sigma^2 + \gamma^2,$$

hence $\sigma = 1/2$. □

Definition 9266.3 (Li discrete time energy)**.** *For a finite reflected zero ledger Z, the nth Li time energy is*

$$\lambda_n^Z := \sum_{\rho \in Z} \left(1 - z(\rho)^n\right), \qquad n \geq 1.$$

The completed zeta Li–Weil row of Definition 9265.21 is the regularized infinite version of the same readout.

Theorem 9266.4 (Fixed-line pairs have nonnegative Li time energy)**.** *Let Z be a finite conjugation-stable zero multiset on the fixed line. Then*

$$\lambda_n^Z \geq 0 \qquad (n \geq 1).$$

Proof. By Definition 9266.2, each zero row has $z(\rho) = e^{i\theta_\rho}$. A conjugate pair contributes

$$\left(1 - e^{in\theta_\rho}\right) + \left(1 - e^{-in\theta_\rho}\right) = 2 - 2\cos(n\theta_\rho) \geq 0.$$

Summing over the finite conjugate pairs gives the claim. □

Definition 9266.5 (Radial Li pair)**.** *A radial Li pair is a reflected pair of zero rows whose Li coordinates are*

$$z, \qquad \frac{1}{\overline{z}},$$

with $|z| \neq 1$. Equivalently, one member of the pair lies inside the unit circle and the other lies outside it.

Principle 9266.6 (Radial Li time amplification)**.** *For a radial Li pair, some discrete time n detects the radial scale. Indeed, if the outside coordinate is $z_* = Re^{i\theta}$ with $R > 1$, then along a subsequence of n the real part of $e^{in\theta}$ stays positive. The term z_*^n then has real part of order R^n, so the contribution of $1 - z_*^n$ eventually gives a negative finite Li direction unless an equally large source row is displayed. The Li criterion consumption row packages this amplification as the finite negative witness of Definition 9265.22.*

Definition 9266.7 (Li time positive tower)**.** *The source row*

$$\mathsf{LiTimePositiveTower}$$

is GlobalLiWeilPositivity *read as a discrete-time energy tower:*

$$\lambda_n \geq 0 \qquad (n \geq 1).$$

Definition 9266.8 (Li coefficient square representation)**.** *The source row*

$$\mathsf{LiCoefficientSquareRepresentation}$$

asserts that every Li–Weil coefficient has a source-side nonnegative presentation, either as a finite Hilbert square

$$\lambda_n = \|T_n\|^2_{\mathcal{H}_n}$$

or as a displayed limit of such squares

$$\lambda_n = \lim_m \|T_{n,m}\|^2_{\mathcal{H}_{n,m}}.$$

Theorem 9266.9 (Li coefficient squares give the Li time tower)**.** *If* LiCoefficientSquareRepresentation *holds, then*

$$\mathsf{LiTimePositiveTower}$$

holds.

Proof. Each displayed square is nonnegative, and a displayed limit of nonnegative real rows is nonnegative. Hence $\lambda_n \geq 0$ for every $n \geq 1$, which is LiTimePositiveTower. □

Corollary 9266.10 (Li coefficient squares give fixed-half readout)**.** *Assume* LiCoefficientSquareRepresentation *and*

$$\mathsf{LiCriterionConsumesGlobalLiPositivity}.$$

Then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9266.9 gives GlobalLiWeilPositivity. The Li criterion consumption row then returns fixed-half readout. □

Principle 9266.11 (Li time no-scale target)**.** *The Li time coordinate is the finite witness version of the no-scale principle. Unit Li coordinates give phase iteration on S^1; radial Li pairs carry nonunit scale, and discrete time powers expose that scale as a finite negative Li witness. In named form the non-circular source target is*

$$\mathsf{LiCoefficientSquareRepresentation} \Longrightarrow \mathsf{LiTimePositiveTower} \Longrightarrow \textit{fixed-half zero readout.}$$

Chapter 9267

Balanced Golden Li Closure

chapter 9265 records the Li–Weil coefficient row and the golden Li positivity limit. The Lee–Yang input required for that limit is not merely a finite zero-location statement. A finite container may have all displayed zeros on the fixed line while its logarithmic derivative still carries an unledgered exponential drift. The Li–Weil route therefore requires a balanced source container: the critical-line divisor, the real structure, and the exponential normalization must be licensed by the same source data.

Definition 9267.1 (Raw critical-line exponential packet)**.** *For $h > 0$ set*

$$C_h(s) := 1 + e^{-h(s-1/2)}.$$

Its zeros satisfy

$$e^{-h(s-1/2)} = -1,$$

hence

$$s = \frac{1}{2} + \frac{(2k+1)\pi i}{h} \qquad (k \in \mathbb{Z}).$$

Thus the zero divisor of C_h lies on the fixed line.

Lemma 9267.2 (Critical-line zeros do not force raw Li positivity)**.** *The packet C_h of Definition 9267.1 has fixed-line zeros, but its raw first logarithmic derivative at $s = 1$ is negative:*

$$\frac{d}{ds} \log C_h(s)\Big|_{s=1} = \frac{-he^{-h/2}}{1 + e^{-h/2}} < 0.$$

Proof. The zero computation is part of Definition 9267.1. Differentiating gives

$$C_h'(s) = -he^{-h(s-1/2)}.$$

Therefore

$$\frac{d}{ds} \log C_h(s)\Big|_{s=1} = \frac{C_h'(1)}{C_h(1)} = \frac{-he^{-h/2}}{1 + e^{-h/2}},$$

which is strictly negative. □

Definition 9267.3 (Balanced elementary packet)**.** *The balanced normalization of C_h is*

$$B_h(s) := e^{h(s-1/2)/2} C_h(s) = e^{h(s-1/2)/2} + e^{-h(s-1/2)/2}.$$

Equivalently,

$$B_h(s) = 2\cosh\left(\frac{h(s-1/2)}{2}\right).$$

Writing $w = s - 1/2$, it satisfies the self-dual balance row

$$B_h\left(\frac{1}{2}+w\right) = B_h\left(\frac{1}{2}-w\right).$$

Definition 9267.4 (Balanced Lee–Yang source container)**.** *A finite entire container $C(s)$ is a balanced Lee–Yang source container when it satisfies the following rows.*

The real row is

$$C(\overline{s}) = \overline{C(s)}.$$

The fixed-line divisor row is

$$C(\rho) = 0 \Longrightarrow \mathrm{Re}(\rho) = \frac{1}{2}.$$

The balance row says that, after the source-determined exponential normalization has been applied,

$$C\left(\frac{1}{2}+w\right) = C\left(\frac{1}{2}-w\right)$$

as an identity in w. The no-free-exponential row says that the canonical product for C contains no independent factor

$$e^{a(s-1/2)}$$

whose coefficient a is chosen outside the source ledger. The source row says that all divisor, height, and normalization data are generated before any zero box is selected.

Definition 9267.5 (Finite balanced Li coefficient)**.** *Let F be a finite balanced container with $F(1) \neq 0$ and finite zero multiset Z_F. Its finite Li coefficient is read from the balanced canonical product by*

$$\lambda_n(F) := \sum_{\rho \in Z_F} \left[1 - \left(1 - \frac{1}{\rho}\right)^n\right] \qquad (n \geq 1).$$

The definition is tied to the no-free-exponential row of Definition 9267.4; adding an independent exponential factor would add logarithmic derivative terms not represented by the displayed divisor sum.

Theorem 9267.6 (Finite balanced critical-line divisors give Li positivity)**.** *Let F be a finite balanced container. If Z_F lies on the fixed line and is closed under complex conjugation, then*

$$\lambda_n(F) \geq 0 \qquad (n \geq 1).$$

Proof. Let

$$\rho = \frac{1}{2} + i\gamma \qquad \text{and} \qquad \overline{\rho} = \frac{1}{2} - i\gamma$$

be a conjugate pair in Z_F. Then

$$1 - \frac{1}{\rho} = \frac{\rho - 1}{\rho} = \frac{-1/2 + i\gamma}{1/2 + i\gamma}.$$

The numerator and denominator have the same absolute value, so there is some real θ with

$$1 - \frac{1}{\rho} = e^{i\theta}.$$

The conjugate zero gives

$$1 - \frac{1}{\bar{\rho}} = e^{-i\theta}.$$

The pair contribution to $\lambda_n(F)$ is therefore

$$\left[1 - e^{in\theta}\right] + \left[1 - e^{-in\theta}\right] = 2\left(1 - \cos(n\theta)\right) \geq 0.$$

Summing over the conjugate pairs gives the claim. □

Definition 9267.7 (Balanced golden normalization row)**.** *Let $L_m(u)$ be a finite Fibonacci or golden Lee–Yang seed and let*

$$\mathcal{Z}_m^{\mathrm{pol}}(z_1, \ldots, z_{D_m})$$

be its polarized container. Given source heights

$$h_{m,1}, \ldots, h_{m,D_m} > 0,$$

the raw zeta-mode substitution is

$$C_m^{\mathrm{raw}}(s) := \mathcal{Z}_m^{\mathrm{pol}}\left(e^{-(s-1/2)h_{m,1}}, \ldots, e^{-(s-1/2)h_{m,D_m}}\right).$$

A balanced golden normalization is a source-determined real number A_m such that

$$C_m(s) := e^{A_m(s-1/2)} C_m^{\mathrm{raw}}(s)$$

is a balanced Lee–Yang source container. The row is source-determined only when A_m is forced by the displayed degree, height, reciprocity, or self-duality ledger of the container.

Definition 9267.8 (Balanced golden construction)**.** *The source row*

BalancedGoldenConstruction

asserts that the golden polarized Lee–Yang containers of Definition 9267.7 admit balanced normalizations C_m whose divisor rows remain fixed-line, whose real rows remain conjugation-stable, and whose no-free-exponential rows are licensed by source data rather than by zero-specific choice.

Definition 9267.9 (Log-derivative convergence)**.** *The source row*

LogDerivativeConvergence

asserts that for each fixed $n \geq 1$ the balanced finite containers C_m satisfy

$$\frac{1}{(n-1)!} \frac{d^n}{ds^n} \left[s^{n-1} \log C_m(s)\right]\Big|_{s=1} \longrightarrow \frac{1}{(n-1)!} \frac{d^n}{ds^n} \left[s^{n-1} \log \xi(s)\right]\Big|_{s=1}.$$

In the notation of Definition 9265.21, this is the row

$$\lambda_n(C_m) \to \lambda_n.$$

Theorem 9267.10 (A local analytic condition gives log-derivative convergence)**.** *Let D be a closed disk centered at $s = 1$ with interior containing $s = 1$. Assume ξ is nonzero on D, the sequence C_m converges uniformly to ξ on D, the functions C_m are nonzero on D for all sufficiently large m, and the logarithm branches are normalized so that*

$$\log C_m(1) \to \log \xi(1).$$

Then LogDerivativeConvergence *holds for every fixed order whose Cauchy circle lies in the interior of D.*

Proof. Since ξ is nonzero on D, there is a holomorphic branch of $\log \xi$ in a neighborhood of D. Uniform convergence and the nonvanishing assumption give compatible holomorphic branches $\log C_m$ on the same neighborhood for all sufficiently large m. The normalization at $s = 1$ fixes the additive constants, so

$$\log C_m \to \log \xi$$

uniformly on smaller disks. Cauchy's integral formula then gives convergence of every fixed derivative of $\log C_m$ at $s = 1$. Since $\lambda_n(C_m)$ is a finite linear combination of derivatives of $\log C_m$ up to order n, the displayed Li–Weil coefficients converge. □

Definition 9267.11 (Balanced golden Lee–Yang Li closure)**.** *The source row*

BalancedGoldenLeeYangLiClosure

asserts the existence of balanced golden Lee–Yang containers C_m such that:

- *each C_m is supplied by* BalancedGoldenConstruction*;*
- *each finite divisor is fixed-line and conjugation-stable;*
- *the finite coefficients $\lambda_n(C_m)$ are read through the balanced product row of Definition 9267.5;*
- LogDerivativeConvergence *holds for the sequence (C_m).*

Theorem 9267.12 (Balanced golden closure gives global Li positivity)**.** *If* BalancedGoldenLeeYangLiClosure *holds, then*

GlobalLiWeilPositivity

holds.

Proof. Fix $n \geq 1$. By the fixed-line and conjugation-stable divisor rows, Definition 9267.6 gives

$$\lambda_n(C_m) \geq 0$$

for every m. By Definition 9267.9,

$$\lambda_n(C_m) \to \lambda_n.$$

The limit of nonnegative real numbers is nonnegative, hence

$$\lambda_n \geq 0.$$

Since n was arbitrary, GlobalLiWeilPositivity holds. □

Corollary 9267.13 (Balanced golden closure gives fixed-half readout)**.** *Assume* BalancedGoldenLeeYangLiClosure *and*

LiCriterionConsumesGlobalLiPositivity.

Then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9267.12 gives GlobalLiWeilPositivity. The Li criterion consumption row then returns the fixed-half zero ledger. □

Principle 9267.14 (Balanced Li closure target)**.** *The finite Lee–Yang condition required by the Li–Weil route is a balanced condition. Fixed-line zero location supplies the divisor part; the real row supplies conjugate pairing; the no-free-exponential row prevents logarithmic derivative drift; and local analytic convergence transfers finite positivity to the completed zeta coefficient row. In source-row form the target is*

BalancedGoldenConstruction *with* LogDerivativeConvergence,

which is the balanced realization of Definition 9265.29.

Chapter 9268

Balanced Polarized Lee–Yang CND

chapter 9262 supplies the polarized Lee–Yang container row, and chapter 9267 records the balanced normalization row. This chapter inserts the finite source implication between them and chapter 9269: balanced polarized Lee–Yang containers have fixed-line divisors and therefore supply Li conditional negative definiteness.

Definition 9268.1 (Self-inversive polarized packet)**.** *Let*

$$\mathcal{Z}(z_1, \dots, z_D)$$

be a multi-affine polarized Lee–Yang packet. It is self-inversive after phase normalization when

$$\mathcal{Z}(z_1, \dots, z_D) = \left(\prod_{j=1}^{D} z_j \right) \mathcal{Z}(z_1^{-1}, \dots, z_D^{-1}).$$

The phase normalization is part of the source ledger; it is not a zero-specific adjustment.

Definition 9268.2 (Balanced polarized readback)**.** *Let $h_1, \dots, h_D > 0$ be source heights and write*

$$H := \sum_{j=1}^{D} h_j.$$

For the logarithmic central variable $w = s - 1/2$, the raw readback is

$$C^{\mathrm{raw}}(w) := \mathcal{Z}(e^{-wh_1}, \dots, e^{-wh_D}).$$

The balanced readback is

$$C(w) := e^{Hw/2} C^{\mathrm{raw}}(w).$$

Theorem 9268.3 (Self-inversive packets have canonical balance)**.** *If $\mathcal{Z}$ is self-inversive in the sense of Definition 9268.1, then the balanced readback of Definition 9268.2 satisfies*

$$C(w) = C(-w).$$

Proof. Substitute $z_j = e^{-wh_j}$ into the self-inversive identity. Since

$$\prod_{j=1}^{D} e^{-wh_j} = e^{-wH},$$

one gets

$$\mathcal{Z}(e^{-wh_1}, \dots, e^{-wh_D}) = e^{-wH}\mathcal{Z}(e^{wh_1}, \dots, e^{wh_D}).$$

Multiplying by $e^{Hw/2}$ gives

$$C(w) = e^{-Hw/2}\mathcal{Z}(e^{wh_1}, \dots, e^{wh_D}).$$

The right side is exactly $C(-w)$. □

Theorem 9268.4 (Polarized Lee–Yang readback has fixed-line divisor)**.** *Assume $\mathcal{Z}$ has the Lee–Yang nonvanishing ledger of Definition 9262.17. Then every zero of the balanced readback $C(w)$ lies on*

$$\mathrm{Re}(w) = 0.$$

Proof. The exponential factor $e^{Hw/2}$ has no zeros, so $C(w) = 0$ is equivalent to

$$\mathcal{Z}(e^{-wh_1}, \dots, e^{-wh_D}) = 0.$$

If $\mathrm{Re}(w) > 0$, then

$$|e^{-wh_j}| = e^{-\mathrm{Re}(w)h_j} < 1$$

for every j, contradicting the inner Lee–Yang nonvanishing row. If $\mathrm{Re}(w) < 0$, then every $|e^{-wh_j}| > 1$, contradicting the outer Lee–Yang nonvanishing row. Thus every zero has $\mathrm{Re}(w) = 0$. □

Definition 9268.5 (Balanced polarized Lee–Yang CND packet)**.** *A balanced polarized Lee–Yang CND packet consists of:*

- *a polarized Lee–Yang packet $\mathcal{Z}$ with the nonvanishing ledger of Definition 9262.17;*
- *source heights $h_1, \dots, h_D > 0$;*
- *the self-inversive balance row of Definition 9268.1;*
- *the balanced readback $C(w)$ of Definition 9268.2;*
- *the real structure needed to pair conjugate fixed-line zeros.*

Theorem 9268.6 (Balanced polarized packets give finite Li CND)**.** *Every balanced polarized Lee–Yang CND packet supplies a finite Li kernel that is conditionally negative definite.*

Proof. By Definition 9268.4, the zeros of the balanced readback lie on $\mathrm{Re}(w) = 0$, hence the corresponding s-zeros lie on $\mathrm{Re}(s) = 1/2$. The real structure pairs them by conjugation. Therefore the zero multiset is a finite conjugation-stable fixed-line divisor. Apply Definition 9269.6. □

Definition 9268.7 (Source balanced golden approximation)**.** *The source row*

$$\mathsf{SourceBalancedGoldenApproximation}$$

asserts that the completed zeta function is approximated near $s = 1$ by balanced polarized golden Lee–Yang CND packets

$$C_m(s) = e^{A_m(s-1/2)}\mathcal{Z}_m^{\mathrm{pol}}\left(e^{-(s-1/2)h_{m,1}}, \dots, e^{-(s-1/2)h_{m,D_m}}\right),$$

where A_m is forced by the self-inversive height ledger and the approximation is source-side, not selected from the zero locations of ξ.

Theorem 9268.8 (Source approximation gives Li CND convergence under local analytic control)**.** *Assume* SourceBalancedGoldenApproximation *and the local analytic hypotheses of Definition 9267.10 for the approximating packets C_m. Then*

$$\mathsf{LiCNDConvergence}$$

holds.

Proof. The local analytic theorem gives convergence of every fixed Li coefficient:

$$\lambda_n(C_m) \to \lambda_n.$$

For each finite kernel matrix only finitely many differences $i - j$ are used, so the same fixed-coefficient convergence gives entrywise convergence of that matrix. This is precisely LiCNDConvergence. □

Corollary 9268.9 (Balanced polarized source approximation gives fixed-half readout)**.** *Assume* SourceBalancedGoldenApproximation*, the local analytic control row of Definition 9268.8, and*

$$\mathsf{LiCriterionConsumesGlobalLiPositivity}.$$

Then the completed zeta zero ledger has fixed-half readout.

Proof. Each balanced polarized packet gives a finite Li CND kernel by Definition 9268.6. The source approximation and local analytic control give LiCNDConvergence. Thus Definition 9269.9 gives LiCND, and Definition 9269.4 gives the fixed-half readout. □

Definition 9268.10 (Balanced Lee–Yang kernel closure)**.** *Let Φ denote the even Fourier–Laplace kernel whose transform reads the completed zeta function on the central line variable. The source row*

$$\mathsf{BalancedLeeYangKernelClosure}$$

asserts that Φ is the limit, in a topology strong enough to imply the local analytic control above, of discrete even source kernels whose Laplace transforms are balanced polarized Lee–Yang CND packets.

Principle 9268.11 (Kernel closure is the CND convergence load)**.** *The finite balanced polarized Lee–Yang construction supplies Li CND without further zero-side input. The analytic load is the convergence row: the source kernels must approximate the Riemann kernel strongly enough to pass logarithmic derivatives at $s = 1$. In named form the target is*

$$\mathsf{BalancedLeeYangKernelClosure} \Longrightarrow \mathsf{LiCNDConvergence} \Longrightarrow \mathsf{LiCND} \Longrightarrow \textit{fixed-half zero readout.}$$

Chapter 9269

Balanced Golden Li CND Closure

chapter 9267 records the balanced Lee–Yang route to pointwise Li positivity. The same source rows carry a stronger structure: the Li coefficients form a conditionally negative definite kernel on the integer ledger. This chapter records that kernel form. The advantage is that the finite input is a single quadratic negative-type statement rather than a list of separate coefficient inequalities.

Definition 9269.1 (Two-sided Li kernel row)**.** *Let $(\lambda_n)_{n\geq 1}$ be the Li–Weil coefficients of Definition 9265.21. Set*

$$\lambda_0 := 0$$

and extend the sequence to $\mathbb{Z}$ by

$$\lambda_{-n} := \lambda_n \qquad (n \geq 1).$$

The associated two-sided Li kernel is

$$K_\lambda(i,j) := \lambda_{i-j} \qquad (i,j \in \mathbb{Z}).$$

Definition 9269.2 (Li conditional negative definiteness)**.** *The source row*

$$\mathsf{LiCND}$$

asserts that the two-sided Li kernel of Definition 9269.1 is conditionally negative definite: for every finite coefficient family

$$c_1, \ldots, c_N \in \mathbb{C}$$

with

$$\sum_{i=1}^{N} c_i = 0,$$

one has

$$\sum_{i,j=1}^{N} c_i \overline{c_j} \lambda_{i-j} \leq 0.$$

Theorem 9269.3 (Li CND gives pointwise Li positivity)**.** *If* LiCND *holds, then*

$$\lambda_n \geq 0 \qquad (n \geq 1).$$

Proof. Fix $n \geq 1$ and use the two-point zero-sum family

$$c_0 = 1, \qquad c_n = -1.$$

The conditional negative definiteness inequality gives

$$\lambda_0 - \lambda_{-n} - \lambda_n + \lambda_0 \leq 0.$$

Since $\lambda_0 = 0$ and $\lambda_{-n} = \lambda_n$, this becomes

$$-2\lambda_n \leq 0.$$

Hence $\lambda_n \geq 0$. □

Corollary 9269.4 (Li CND gives fixed-half readout)**.** *Assume* LiCND *and*

LiCriterionConsumesGlobalLiPositivity.

Then the completed zeta zero ledger has fixed-half readout.

Proof. By Definition 9269.3, $\lambda_n \geq 0$ for every $n \geq 1$. This is GlobalLiWeilPositivity. The Li criterion consumption row then returns fixed-half readout. □

Definition 9269.5 (Finite balanced Li CND sequence)**.** *Let Z be a finite conjugation-stable zero multiset on the fixed line. For each $\rho \in Z$ define*

$$\alpha_\rho := 1 - \frac{1}{\rho}.$$

The finite balanced Li sequence attached to Z is

$$\lambda_n^Z := \sum_{\rho \in Z} \left(1 - \alpha_\rho^n\right) \qquad (n \geq 1),$$

with

$$\lambda_0^Z := 0, \qquad \lambda_{-n}^Z := \lambda_n^Z.$$

Theorem 9269.6 (Balanced critical divisors give Li CND)**.** *Let Z be a finite conjugation-stable zero multiset on the fixed line. Then the sequence $(\lambda_n^Z)_{n \in \mathbb{Z}}$ of Definition 9269.5 is conditionally negative definite.*

Proof. For

$$\rho = \frac{1}{2} + i\gamma$$

one has

$$\left|1 - \frac{1}{\rho}\right| = \left|\frac{\rho - 1}{\rho}\right| = 1.$$

Thus every conjugate pair contributes phases

$$\alpha = e^{i\theta}, \qquad \overline{\alpha} = e^{-i\theta}.$$

The pair contribution to λ_n^Z is

$$2 - e^{in\theta} - e^{-in\theta} = 2(1 - \cos(n\theta)).$$

Hence there is a positive finite measure μ_Z on the circle such that

$$\lambda_n^Z = \int_{\mathbb{T}} (1 - \cos(n\theta))\, d\mu_Z(\theta).$$

Let $c_1, \ldots, c_N$ be a finite zero-sum family. Then

$$\sum_{i,j} c_i \overline{c_j} \lambda_{i-j}^Z = \int_{\mathbb{T}} \sum_{i,j} c_i \overline{c_j}\, (1 - \cos((i-j)\theta))\, d\mu_Z(\theta).$$

The constant part vanishes because

$$\sum_{i,j} c_i \overline{c_j} = \left| \sum_i c_i \right|^2 = 0.$$

The remaining part is

$$-\int_{\mathbb{T}} \mathrm{Re}\left(\sum_{i,j} c_i \overline{c_j} e^{i(i-j)\theta} \right) d\mu_Z(\theta).$$

But

$$\sum_{i,j} c_i \overline{c_j} e^{i(i-j)\theta} = \left| \sum_i c_i e^{ii\theta} \right|^2 \geq 0.$$

Therefore the displayed quadratic form is nonpositive. This is conditional negative definiteness. □

Corollary 9269.7 (Balanced finite containers give Li CND)**.** *Let C be a balanced Lee–Yang source container whose finite zero multiset is fixed-line and conjugation-stable. Then the finite Li sequence of C is conditionally negative definite.*

Proof. The zero multiset of C is a finite conjugation-stable fixed-line divisor. Apply Definition 9269.6. □

Definition 9269.8 (Li CND convergence)**.** *Let C_m be balanced finite containers with finite Li sequences $(\lambda_n(C_m))_{n\in\mathbb{Z}}$. They satisfy*

$$\mathsf{LiCNDConvergence}$$

to the completed zeta Li kernel when for every fixed $n \in \mathbb{Z}$,

$$\lambda_n(C_m) \to \lambda_n,$$

or equivalently, for every finite matrix size, the kernel entries

$$\lambda_{i-j}(C_m)$$

converge entrywise to

$$\lambda_{i-j}.$$

Theorem 9269.9 (CND is closed under finite kernel limits)**.** *Assume each finite kernel $(\lambda_{i-j}(C_m))$ is conditionally negative definite and* LiCNDConvergence *holds. Then*

$$\mathsf{LiCND}$$

holds for the completed zeta Li kernel.

Proof. Let $c_1, \ldots, c_N$ be a finite zero-sum family. For each m, conditional negative definiteness gives

$$\sum_{i,j=1}^{N} c_i \overline{c_j} \lambda_{i-j}(C_m) \leq 0.$$

By entrywise convergence on this finite matrix,

$$\sum_{i,j=1}^{N} c_i \overline{c_j} \lambda_{i-j}(C_m) \longrightarrow \sum_{i,j=1}^{N} c_i \overline{c_j} \lambda_{i-j}.$$

The limit of nonpositive real numbers is nonpositive, so the completed zeta Li kernel is conditionally negative definite. □

Definition 9269.10 (Balanced golden Li CND closure)**.** *The source row*

BalancedGoldenLiCNDClosure

asserts the existence of balanced golden Lee–Yang containers C_m such that:

- *the containers are generated by* BalancedGoldenConstruction*;*
- *each finite zero multiset is fixed-line and conjugation-stable;*
- *each finite Li kernel is read through the balanced product row;*
- LiCNDConvergence *holds.*

Theorem 9269.11 (Balanced golden Li CND closure gives Li CND)**.** *If* BalancedGoldenLiCNDClosure *holds, then*

LiCND

holds.

Proof. By Definition 9269.7, each finite container supplies a conditionally negative definite Li kernel. The closure row supplies LiCNDConvergence. Apply Definition 9269.9. □

Corollary 9269.12 (Balanced golden Li CND closure gives fixed-half readout)**.** *Assume* BalancedGoldenLiCNDClos *and*

LiCriterionConsumesGlobalLiPositivity.

Then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9269.11 gives LiCND. Then Definition 9269.4 applies. □

Principle 9269.13 (CND closure as Li–Weil source positivity)**.** *The conditionally negative definite formulation strengthens pointwise Li positivity into a source-side kernel statement. Finite balanced Lee–Yang divisors give negative type on the integer ledger; finite kernel limits preserve negative type; and the two-point zero-sum test recovers the usual coefficient inequalities. Thus a source target is*

BalancedGoldenConstruction *with* LiCNDConvergence,

rather than a separate inspection of the individual coefficients.

Definition 9269.14 (Li second-difference kernel)**.** *For the two-sided Li kernel of Definition 9269.1, define*

$$\beta_n := \lambda_{n+1} + \lambda_{n-1} - 2\lambda_n \qquad (n \in \mathbb{Z}).$$

The source row

LiSecondDifferencePositiveDefinite

asserts that $(\beta_n)_{n\in\mathbb{Z}}$ *is a positive definite translation kernel on* $\mathbb{Z}$*: for every finite family*

$$a_1, \ldots, a_N \in \mathbb{C}, \qquad k_1, \ldots, k_N \in \mathbb{Z},$$

one has

$$\sum_{i,j=1}^{N} a_i \overline{a_j} \beta_{k_i - k_j} \geq 0.$$

Theorem 9269.15 (Second-difference positivity gives Li CND)**.** *If* LiSecondDifferencePositiveDefinite *holds, then*

LiCND

holds.

Proof. Let $(c_k)_{k\in\mathbb{Z}}$ be a finitely supported family with

$$\sum_k c_k = 0.$$

Choose a finitely supported antiderivative $(b_k)_{k\in\mathbb{Z}}$ with

$$c_k = b_k - b_{k-1}.$$

Such a b is obtained by cumulative summation across an interval containing the support of c; the zero-sum condition makes the cumulative row return to 0 outside that interval. Expanding gives

$$\sum_{r,s} c_r \overline{c_s} \lambda_{r-s} = \sum_{r,s} b_r \overline{b_s} \left(2\lambda_{r-s} - \lambda_{r-s+1} - \lambda_{r-s-1}\right).$$

By the definition of β,

$$2\lambda_{r-s} - \lambda_{r-s+1} - \lambda_{r-s-1} = -\beta_{r-s}.$$

Therefore

$$\sum_{r,s} c_r \overline{c_s} \lambda_{r-s} = -\sum_{r,s} b_r \overline{b_s} \beta_{r-s} \leq 0$$

by positive definiteness of the β kernel. This is precisely LiCND. □

Corollary 9269.16 (Second-difference positivity gives fixed-half readout)**.** *Assume* LiSecondDifferencePositiveDefin *and*

LiCriterionConsumesGlobalLiPositivity.

Then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9269.15 gives LiCND. Apply Definition 9269.4. □

Definition 9269.17 (Finite balanced beta kernel)**.** *For a finite balanced Li sequence* $(\lambda_n^Z)_{n\in\mathbb{Z}}$ *from Definition 9269.5, define*

$$\beta_n^Z := \lambda_{n+1}^Z + \lambda_{n-1}^Z - 2\lambda_n^Z.$$

Theorem 9269.18 (Balanced critical divisors give beta positivity)**.** *Let Z be a finite conjugation-stable zero multiset on the fixed line. Then $(\beta_n^Z)_{n\in\mathbb{Z}}$ is positive definite.*

Proof. The proof of Definition 9269.6 gives a positive finite measure μ_Z on the circle such that

$$\lambda_n^Z = \int_{\mathbb{T}} (1-\cos(n\theta))\, d\mu_Z(\theta).$$

Taking the second difference gives

$$\beta_n^Z = \int_{\mathbb{T}} 2(1-\cos\theta)\cos(n\theta)\, d\mu_Z(\theta).$$

Equivalently, β_n^Z is the nth Fourier coefficient of the positive finite measure

$$d\sigma_Z(\theta) := 2(1-\cos\theta)\, d\mu_Z(\theta).$$

Hence for every finite family (a_i, k_i),

$$\sum_{i,j} a_i\overline{a_j}\beta_{k_i-k_j}^Z = \int_{\mathbb{T}} \left|\sum_i a_i e^{ik_i\theta}\right|^2 d\sigma_Z(\theta) \geq 0.$$

□

Definition 9269.19 (Beta-kernel convergence)**.** *Balanced finite containers C_m satisfy*

$$\mathsf{BetaKernelConvergence}$$

when, for every fixed $n \in \mathbb{Z}$,

$$\beta_n(C_m) \to \beta_n.$$

Equivalently, every finite Toeplitz matrix

$$\Big(\beta_{k_i-k_j}(C_m)\Big)_{i,j}$$

converges entrywise to the corresponding completed zeta beta-kernel matrix.

Theorem 9269.20 (Positive beta kernels are closed under finite limits)**.** *Assume every finite beta kernel $\beta(C_m)$ is positive definite and that* BetaKernelConvergence *holds. Then*

$$\mathsf{LiSecondDifferencePositiveDefinite}$$

holds for the completed zeta Li sequence.

Proof. Fix a finite family (a_i, k_i). For each m,

$$\sum_{i,j} a_i\overline{a_j}\beta_{k_i-k_j}(C_m) \geq 0.$$

Entrywise convergence on this finite matrix gives

$$\sum_{i,j} a_i\overline{a_j}\beta_{k_i-k_j}(C_m) \longrightarrow \sum_{i,j} a_i\overline{a_j}\beta_{k_i-k_j}.$$

The limit of nonnegative real numbers is nonnegative, so the completed zeta beta kernel is positive definite. □

Definition 9269.21 (Balanced golden beta closure)**.** *The source row*

BalancedGoldenBetaClosure

asserts the existence of balanced golden Lee–Yang containers C_m such that:

- *the containers are generated by* BalancedGoldenConstruction*;*
- *each finite zero multiset is fixed-line and conjugation-stable;*
- *each finite beta kernel is read from the balanced product row;*
- BetaKernelConvergence *holds.*

Theorem 9269.22 (Balanced golden beta closure gives Li second-difference positivity)**.** *If* BalancedGoldenBetaClos *holds, then*

LiSecondDifferencePositiveDefinite

holds.

Proof. By Definition 9269.18, each finite container supplies a positive definite beta kernel. The closure row supplies BetaKernelConvergence. Apply Definition 9269.20. □

Corollary 9269.23 (Balanced golden beta closure gives fixed-half readout)**.** *Assume* BalancedGoldenBetaClosure *and*

LiCriterionConsumesGlobalLiPositivity.

Then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9269.22 gives LiSecondDifferencePositiveDefinite. Then Definition 9269.16 applies. □

Principle 9269.24 (Beta closure as Toeplitz Li–Weil positivity)**.** *The beta-kernel formulation replaces separate Li coefficient inspection by one Toeplitz positive-definiteness row. Finite balanced critical divisors supply this row as Fourier positivity of a positive circle measure; finite kernel limits preserve it; and the discrete antiderivative identity converts it into* LiCND*. Thus*

BalancedGoldenBetaClosure

is a sharper source-side target than entrywise Li positivity, while its fixed-half readout still passes through the displayed Li criterion consumption row.

Chapter 9270

Li–Weil Herglotz Target

chapter 9269 turns the Li–Weil route into the positive definiteness of the second-difference kernel β. This chapter records the same target as a positive-real log-derivative row. The purpose is not to claim the row; it is to name the Herglotz interface through which beta-kernel positivity, balanced Lee–Yang limits, and fixed-half readout can be compared.

Definition 9270.1 (Li beta Herglotz function)**.** *Let $(\beta_n)_{n\in\mathbb{Z}}$ be the Li second-difference kernel of Definition 9269.14. Its Herglotz readout is the holomorphic disk function*

$$H_\beta(z) := \beta_0 + 2\sum_{n\geq 1} \beta_n z^n$$

on every disk where the series is defined by the Li–Weil generating row.

Definition 9270.2 (Xi log-derivative Herglotz row)**.** *The source row*

$$\mathsf{XiLogDerivativeHerglotz}$$

asserts

$$\operatorname{Re}\frac{\xi'(s)}{\xi(s)} \geq 0 \qquad \left(\operatorname{Re}(s) > \frac{1}{2}\right).$$

It is a positive-real row for the logarithmic derivative of the completed zeta function on the open half-plane to the right of the fixed line.

Theorem 9270.3 (Li beta readout is the xi log-derivative readout)**.** *With*

$$s = \frac{1}{1-z},$$

one has the formal identity

$$H_\beta(z) = 2\frac{\xi'(s)}{\xi(s)}.$$

Proof. Let

$$L(z) := \sum_{n\geq 1} \lambda_n z^n.$$

The Li generating row gives

$$L(z) = z\frac{d}{dz}\log\xi\left(\frac{1}{1-z}\right).$$

Let

$$B_+(z) := \sum_{n\geq 0} \beta_n z^n.$$

Using $\lambda_0 = 0$ and $\beta_0 = 2\lambda_1$, the definition of β_n gives

$$B_+(z) = \lambda_1 + (z + z^{-1} - 2)L(z).$$

Since $H_\beta(z) = 2B_+(z) - \beta_0$, this becomes

$$H_\beta(z) = 2(z + z^{-1} - 2)L(z).$$

Now

$$z + z^{-1} - 2 = \frac{(1-z)^2}{z}.$$

Therefore

$$H_\beta(z) = 2(1-z)^2 \frac{d}{dz} \log \xi \left(\frac{1}{1-z} \right).$$

For $s = 1/(1-z)$,

$$\frac{ds}{dz} = \frac{1}{(1-z)^2}.$$

The displayed expression reduces to

$$2\frac{\xi'(s)}{\xi(s)}.$$

□

Theorem 9270.4 (Herglotz row gives beta positivity)**.** *If H_β is a Herglotz function on the unit disk, then*

LiSecondDifferencePositiveDefinite

holds.

Proof. The Herglotz representation theorem gives a positive finite circle measure σ such that

$$H_\beta(z) = \int_{\mathbb{T}} \frac{\zeta + z}{\zeta - z}\, d\sigma(\zeta)$$

after the standard normalization of the imaginary constant. Comparing Taylor coefficients gives

$$\beta_n = \int_{\mathbb{T}} \zeta^{-n}\, d\sigma(\zeta).$$

Thus for every finite family (a_i, k_i),

$$\sum_{i,j} a_i \overline{a_j} \beta_{k_i - k_j} = \int_{\mathbb{T}} \left| \sum_i a_i \zeta^{-k_i} \right|^2 d\sigma(\zeta) \geq 0.$$

□

Theorem 9270.5 (Xi Herglotz row gives fixed-half readout)**.** *Assume* XiLogDerivativeHerglotz *and the usual reflection consumption row for the completed zeta zero ledger. Then the completed zeta zero ledger has fixed-half readout.*

Proof. Suppose a zero ρ of multiplicity $m \geq 1$ satisfies

$$\mathrm{Re}(\rho) > \frac{1}{2}.$$

Near ρ,

$$\frac{\xi'(s)}{\xi(s)} = \frac{m}{s-\rho} + \text{holomorphic row}.$$

Taking $s = \rho - \varepsilon$ with $\varepsilon > 0$ real and sufficiently small keeps s in the half-plane $\mathrm{Re}(s) > 1/2$ while

$$\mathrm{Re}\,\frac{m}{s-\rho} = -\frac{m}{\varepsilon} \to -\infty.$$

The holomorphic part is bounded near ρ, so the real part of ξ'/ξ is negative at such points. This contradicts XiLogDerivativeHerglotz. The reflection consumption row excludes the opposite open side by reflecting any such zero into the side already excluded. Hence the zero ledger reads through the fixed-half surface. □

Definition 9270.6 (Positive spectral measure for xi)**.** *The source row*

PositiveSpectralMeasureForXi

asserts the existence of a positive measure ν on $\mathbb{R}$ such that, for $\mathrm{Re}(w) > 0$,

$$\frac{\xi'}{\xi}\left(\frac{1}{2} + w\right) = \int_{\mathbb{R}} \frac{2w}{w^2 + t^2}\, d\nu(t).$$

The measure must be constructed from source rows rather than from a post-read zero spectrum.

Theorem 9270.7 (Positive spectral measure gives xi Herglotz row)**.** *If* PositiveSpectralMeasureForXi *holds, then*

XiLogDerivativeHerglotz

holds.

Proof. Write $w = x + iy$ with $x > 0$. For each real t,

$$\mathrm{Re}\,\frac{w}{w^2 + t^2} = \frac{x(|w|^2 + t^2)}{|w^2 + t^2|^2} \geq 0.$$

Integrating against the positive measure ν gives the claimed positive-real row. □

Definition 9270.8 (Balanced golden Herglotz closure)**.** *The source row*

BalancedGoldenHerglotzClosure

asserts the existence of balanced golden Lee–Yang containers C_m whose logarithmic derivatives have positive spectral representations on the right half-plane and converge locally uniformly to ξ'/ξ there.

Theorem 9270.9 (Balanced golden Herglotz closure gives xi Herglotz row)**.** *If* BalancedGoldenHerglotzClosure *holds, then*

XiLogDerivativeHerglotz

holds.

Proof. Each container log-derivative has nonnegative real part by its positive spectral representation. Local uniform convergence preserves nonnegative real part pointwise in the open half-plane. The limit is ξ'/ξ, hence XiLogDerivativeHerglotz holds. □

Corollary 9270.10 (Balanced golden Herglotz closure gives fixed-half readout)**.** *Assume* BalancedGoldenHerglotzCl *and the usual reflection consumption row for the completed zeta zero ledger. Then the completed zeta zero ledger has fixed-half readout.*

Proof. Definition 9270.9 gives XiLogDerivativeHerglotz. Apply Definition 9270.5. □

Principle 9270.11 (Herglotz target as Li–Weil source positivity)**.** *The Herglotz row is the analytic positive-real face of the Li–Weil beta kernel. The source-side target is not the posterior placement of zeros on the fixed line; it is a positive spectral representation, or a balanced golden sequence of such representations, whose log-derivative readout converges to that of ξ. In named form the route is*

$$\mathsf{BalancedGoldenHerglotzClosure} \Longrightarrow \mathsf{XiLogDerivativeHerglotz} \Longrightarrow \textit{fixed-half zero readout.}$$

Definition 9270.12 (Centered xi square coordinate)**.** *Let*

$$F_\xi(w) := \xi\left(\frac{1}{2} + w\right).$$

The functional equation makes F_ξ an even entire row. The associated square-coordinate row is

$$G_\xi(x) := F_\xi(\sqrt{x}),$$

understood as the entire function induced by the even power series of F_ξ. Its logarithmic derivative is

$$S_\xi(x) := \frac{G'_\xi(x)}{G_\xi(x)}.$$

Whenever $x = w^2$ and w lies in the right half-plane, the chain rule gives

$$S_\xi(w^2) = \frac{1}{2w}\frac{\xi'}{\xi}\left(\frac{1}{2} + w\right).$$

Definition 9270.13 (Xi Stieltjes log-derivative row)**.** *The source row*

$$\mathsf{XiStieltjesLogDerivative}$$

asserts that there exists a positive measure ν on $[0, \infty)$ such that

$$S_\xi(x) = \int_0^\infty \frac{1}{x+t}\, d\nu(t)$$

on the slit plane where the Stieltjes transform is defined. The measure is source data: it is not obtained by first reading a fixed-half zero divisor and then placing atoms at the squared ordinates.

Theorem 9270.14 (Stieltjes row gives the xi Herglotz row)**.** *If* XiStieltjesLogDerivative *holds, then*

$$\mathsf{XiLogDerivativeHerglotz}$$

holds.

Proof. For $w = a + ib$ with $a > 0$, the Stieltjes representation gives

$$\frac{\xi'}{\xi}\left(\frac{1}{2} + w\right) = 2w \int_0^\infty \frac{1}{w^2 + t}\, d\nu(t) = \int_0^\infty \frac{2w}{w^2 + t}\, d\nu(t).$$

For each $t \geq 0$,

$$\operatorname{Re} \frac{2w}{w^2 + t} = \frac{2a(|w|^2 + t)}{|w^2 + t|^2} \geq 0.$$

Integration against the positive measure ν gives the positive-real row on the right half-plane, hence XiLogDerivativeHerglotz. □

Corollary 9270.15 (Stieltjes log-derivative gives fixed-half readout)**.** *Assume* XiStieltjesLogDerivative *and the usual reflection consumption row for the completed zeta zero ledger. Then the completed zeta zero ledger has fixed-half readout.*

Proof. Definition 9270.14 gives XiLogDerivativeHerglotz. Apply Definition 9270.5. □

Theorem 9270.16 (Fixed-half divisor gives posterior Stieltjes readback)**.** *Assume the completed zeta zero ledger has fixed-half readout and admits the symmetric product row whose logarithmic derivative is locally uniformly obtained from finite divisor truncations. Then S_ξ has a posterior Stieltjes readback*

$$S_\xi(x) = \int_0^\infty \frac{1}{x + t}\, d\nu_{\text{post}}(t)$$

with positive atomic measure ν_{post}.

Proof. The fixed-half divisor has centered zeros

$$w = \pm i\gamma_j$$

with multiplicities. The symmetric product row gives

$$F_\xi(w) = F_\xi(0) \prod_j \left(1 + \frac{w^2}{\gamma_j^2}\right)$$

in the displayed regularized sense. With $x = w^2$ this becomes

$$G_\xi(x) = F_\xi(0) \prod_j \left(1 + \frac{x}{\gamma_j^2}\right).$$

Logarithmic differentiation through the accepted truncation limit gives

$$S_\xi(x) = \sum_j \frac{1}{x + \gamma_j^2}.$$

Thus S_ξ is the Stieltjes transform of the positive atomic measure

$$\nu_{\text{post}} = \sum_j \delta_{\gamma_j^2},$$

counting multiplicity. □

Definition 9270.17 (Stieltjes Hankel positivity target)**.** *The source row*

$$\mathsf{XiStieltjesHankelPositivity}$$

asserts that the moment sequence extracted from the expansion

$$S_\xi(x) = \sum_{k\geq 0}(-1)^k \mu_k x^k$$

satisfies the Stieltjes Hankel inequalities

$$\det(\mu_{i+j})_{i,j=0}^N \geq 0, \qquad \det(\mu_{i+j+1})_{i,j=0}^N \geq 0$$

for every $N \geq 0$, together with the analytic determinacy data required to reconstruct the positive measure in Definition 9270.13.

Definition 9270.18 (Stieltjes Pade approximation target)**.** *The source row*

$$\mathsf{XiStieltjesPadeApproximation}$$

asserts the existence of rational source approximants

$$S_m(x) = \sum_j \frac{a_{m,j}}{x + t_{m,j}}, \qquad a_{m,j} > 0, \quad t_{m,j} \geq 0,$$

which converge locally uniformly to S_ξ on compact subsets of the Stieltjes domain. This is the finite positive-spectrum version of XiStieltjesLogDerivative*.*

Principle 9270.19 (Stieltjes row as compressed source target)**.** *The Stieltjes row is the centered square-coordinate form of the Li–Weil, Herglotz, and no-scale targets. It replaces local zero boxes and posterior containers by a single positive spectral statement for the logarithmic derivative:*

$$\mathsf{XiStieltjesLogDerivative} \Longrightarrow \mathsf{XiLogDerivativeHerglotz} \Longrightarrow \textit{fixed-half zero readout.}$$

The reverse direction is only a posterior readback unless the positive measure is generated before the zero divisor is consumed.

Chapter 9271

Pick Kernel Triadic Positivity

chapter 9270 records the positive-real log-derivative row, and chapter 9277 records the P, U, and T audit discipline for RH-facing source rows. This chapter rewrites the Herglotz target as a Pick-kernel target. The purpose is to make the unit side finite-matrix checkable: a failure of positivity is witnessed by a finite Pick matrix, while a source-side Hilbert factorization supplies the right-triangle norm form.

Definition 9271.1 (Cayley right-half-plane readout)**.** *Let*

$$s(z) := \frac{1}{1-z}.$$

Then

$$s(z) - \frac{1}{2} = \frac{1+z}{2(1-z)}.$$

Thus the unit disk is the Cayley coordinate for the open half-plane to the right of the fixed line. Define the completed-zeta logarithmic Pick readout by

$$f_\xi(z) := 2\frac{\xi'(s(z))}{\xi(s(z))}$$

on the disk locus where this expression is holomorphic.

Definition 9271.2 (Xi Pick kernel)**.** *The Xi Pick kernel attached to Definition 9271.1 is*

$$K_\xi(z,w) := \frac{f_\xi(z) + \overline{f_\xi(w)}}{1 - z\overline{w}}.$$

The source row

$$\mathsf{TriadicPickPositivity}_\xi$$

asserts that f_ξ is holomorphic on the unit disk and that K_ξ is positive definite: for every finite family $z_1, \ldots, z_N \in \mathbb{D}$ and $c_1, \ldots, c_N \in \mathbb{C}$,

$$\sum_{i,j=1}^{N} c_i \overline{c_j} K_\xi(z_i, z_j) \geq 0.$$

Theorem 9271.3 (Pick positivity gives the Herglotz row)**.** *If $\mathsf{TriadicPickPositivity}_\xi$ holds, then*

$$\mathsf{XiLogDerivativeHerglotz}$$

holds.

Proof. Positive definiteness of K_ξ applied to the one-point family z gives

$$K_\xi(z,z) = \frac{2\operatorname{Re} f_\xi(z)}{1-|z|^2} \geq 0.$$

Since $1-|z|^2 > 0$ in the disk,

$$\operatorname{Re} f_\xi(z) \geq 0.$$

Under the coordinate $s = s(z)$ this is precisely the positive-real row for

$$2\frac{\xi'(s)}{\xi(s)}$$

on $\operatorname{Re}(s) > 1/2$, which is XiLogDerivativeHerglotz. □

Theorem 9271.4 (Herglotz row gives Pick positivity)**.** *Assume f_ξ is holomorphic on the unit disk and has a Herglotz representation*

$$f_\xi(z) = ib + \int_{\mathbb{T}} \frac{\zeta+z}{\zeta-z}\, d\mu(\zeta),$$

where $b \in \mathbb{R}$ and μ is a positive finite measure. Then K_ξ is positive definite.

Proof. The imaginary constant cancels in $f_\xi(z) + \overline{f_\xi(w)}$. Substituting the integral representation gives

$$K_\xi(z,w) = 2\int_{\mathbb{T}} \frac{1}{\zeta-z}\frac{1}{\overline{\zeta-w}}\, d\mu(\zeta).$$

For a finite family z_i and coefficients c_i,

$$\sum_{i,j} c_i\overline{c_j}K_\xi(z_i,z_j) = 2\int_{\mathbb{T}} \left|\sum_i \frac{c_i}{\zeta-z_i}\right|^2 d\mu(\zeta) \geq 0.$$

Thus the kernel is positive definite. □

Corollary 9271.5 (Pick positivity gives fixed-half readout)**.** *Assume* $\mathsf{TriadicPickPositivity}_\xi$ *and the usual reflection consumption row for the completed zeta zero ledger. Then the completed zeta zero ledger has fixed-half readout.*

Proof. By Definition 9271.3, XiLogDerivativeHerglotz holds. Apply Definition 9270.5. □

Definition 9271.6 (Finite Pick witness)**.** *A finite Pick witness against the Xi Pick row is a tuple*

$$(z_1,\ldots,z_N; c_1,\ldots,c_N)$$

with $z_i \in \mathbb{D}$ and $c_i \in \mathbb{C}$ such that

$$\sum_{i,j=1}^{N} c_i\overline{c_j}K_\xi(z_i,z_j) < 0.$$

It is the finite-matrix form of a negative energy witness.

Principle 9271.7 (Pick witness as the time side)**.** *The Pick formulation supplies the* T *side of the audit in a finite form. If the kernel row fails, the failure appears at some finite matrix size with finitely many disk points and coefficients. Thus the obstruction is not postponed to an unbounded search through all certificate levels; it has a displayed finite witness.*

Definition 9271.8 (Source Pick Hilbert factorization)**.** *The source row*

$$\mathsf{SourcePickHilbertFactorization}_\xi$$

asserts that there are a Hilbert space $\mathcal{H}$ and a source-side map

$$\Phi_\xi : \mathbb{D} \to \mathcal{H}$$

such that

$$K_\xi(z, w) = \langle \Phi_\xi(z), \Phi_\xi(w) \rangle_{\mathcal{H}}.$$

The map Φ_ξ must be constructed from prime, gamma, normalization, golden, Stieltjes, Herglotz, or explicit-formula source rows before the completed zero divisor is read.

Theorem 9271.9 (Hilbert factorization supplies Pick positivity)**.** *If* $\mathsf{SourcePickHilbertFactorization}_\xi$ *holds, then*

$$\mathsf{TriadicPickPositivity}_\xi$$

holds.

Proof. For any finite family,

$$\sum_{i,j} c_i \overline{c_j} K_\xi(z_i, z_j) = \left\| \sum_i c_i \Phi_\xi(z_i) \right\|_{\mathcal{H}}^2 \geq 0.$$

□

Definition 9271.10 (Finite source Pick approximation)**.** *A finite source Pick approximation is a sequence of positive kernels*

$$K_m(z, w) = \langle \Phi_m(z), \Phi_m(w) \rangle_{\mathcal{H}_m}$$

whose construction is source-side and whose finite matrix entries converge to the Xi Pick entries:

$$K_m(z_i, z_j) \to K_\xi(z_i, z_j)$$

for every finite disk family $z_1, \ldots, z_N$.

Theorem 9271.11 (Finite Pick approximations close the Pick row)**.** *If a finite source Pick approximation exists, then*

$$\mathsf{TriadicPickPositivity}_\xi$$

holds.

Proof. For each m and every finite coefficient family,

$$\sum_{i,j} c_i \overline{c_j} K_m(z_i, z_j) \geq 0.$$

Passing to the displayed finite-matrix limit gives

$$\sum_{i,j} c_i \overline{c_j} K_\xi(z_i, z_j) \geq 0.$$

This is positive definiteness of K_ξ. □

Principle 9271.12 (Triadic meaning of the Pick row)**.** *The Pick row is the compact Hilbert-kernel form of the triadic audit. Its* P *side is the requirement that K_ξ or its approximants are built from prime, gamma, and normalization ledgers. Its* U *side is positive definiteness, equivalently the norm square in Definition 9271.8. Its* T *side is the finite Pick witness row of Definition 9271.6. The same obstruction that appears as a scale leg in Definition 9276.1 appears here as a failure of a finite Gram matrix.*

Theorem 9271.13 (Right-side zero gives a one-point Pick witness)**.** *Assume ρ is a zero of ξ with multiplicity $m \geq 1$ and*

$$\mathrm{Re}(\rho) > \frac{1}{2}.$$

Then the Xi Pick row has a one-point negative witness at some $z_0 \in \mathbb{D}$.

Proof. Near ρ,

$$\frac{\xi'(s)}{\xi(s)} = \frac{m}{s-\rho} + \text{holomorphic row}.$$

Choose $\varepsilon > 0$ so small that

$$s_0 := \rho - \varepsilon$$

still satisfies $\mathrm{Re}(s_0) > 1/2$ and lies outside the zero locus. The holomorphic row is bounded near ρ, while

$$\mathrm{Re}\,\frac{m}{s_0-\rho} = -\frac{m}{\varepsilon}.$$

Hence, for sufficiently small ε,

$$\mathrm{Re}\,\frac{\xi'(s_0)}{\xi(s_0)} < 0.$$

Let

$$z_0 := 1 - \frac{1}{s_0}.$$

The Cayley readout of Definition 9271.1 places z_0 in the unit disk. By definition of f_ξ,

$$\mathrm{Re}\, f_\xi(z_0) < 0.$$

Therefore

$$K_\xi(z_0, z_0) = \frac{2\,\mathrm{Re}\, f_\xi(z_0)}{1-|z_0|^2} < 0,$$

which is the one-point instance of Definition 9271.6. □

Definition 9271.14 (Triadic Pick obstruction packet)**.** *A triadic Pick obstruction packet is the source-side audit row*

$$\mathsf{TriadicPickObstruction}_\xi$$

with three displayed sides. Its P *side is the prime-ledger and solenoid-fibre row: hidden coordinates may refine phase and prime-register data, as in Definition 9286.1, but may not supply an unrecorded $\mathbb{R}_{>0}$ scale gauge. Its* T *side is the finite witness extraction of Definition 9271.13. Its* U *side is the positive kernel row*

$$\mathsf{TriadicPickPositivity}_\xi.$$

The packet is closed when these three sides read the same radial defect: the prime side refuses it as hidden scale, the time side extracts it as a finite Pick witness, and the unit side excludes that witness by positive definiteness.

Theorem 9271.15 (Triadic Pick obstruction gives fixed-half readout)**.** *Assume* $\mathsf{TriadicPickObstruction}_\xi$*, assume*

$$\mathsf{TriadicPickPositivity}_\xi,$$

and assume the usual reflection consumption row for the completed zeta zero ledger. Then the represented completed zero ledger has fixed-half readout.

Proof. Suppose there is a represented zero with readout on the right open side of the fixed-half surface. By Definition 9271.13, it supplies a one-point finite Pick witness

$$K_\xi(z_0, z_0) < 0.$$

However, $\mathsf{TriadicPickPositivity}_\xi$ applied to the same one-point family gives

$$K_\xi(z_0, z_0) \geq 0.$$

This contradiction excludes the right open side. The reflection consumption row excludes the opposite open side by transporting it to the side already excluded. Hence the represented completed zero ledger reads through the fixed-half surface. □

Principle 9271.16 (Pick tower as the common triangular target)**.** *The finite source Pick approximation of Definition 9271.10 is the compact target for the three-sided route. Its* P_m *row is the source construction of* Φ_m *from prime, gamma, normalization, and ledger data. Its* U_m *row is the finite square identity*

$$K_m(z, w) = \langle \Phi_m(z), \Phi_m(w) \rangle_{\mathcal{H}_m}.$$

Its T_m *row is finite-matrix convergence*

$$K_m(z_i, z_j) \to K_\xi(z_i, z_j)$$

for every displayed finite disk family. Thus the remaining source burden is not a separate search along Li, Weil, and Nyman lines; it is the construction of a source-side Hilbert factorization whose finite positive kernels converge to the Xi Pick kernel.

Definition 9271.17 (Zeta positive triangular observation system)**.** *The source row*

$$\mathsf{ZetaPositiveTriangle}_\xi$$

is a finite-to-limit observation system consisting of source-side Hilbert kernels

$$K_m(z, w) = \langle \Phi_m(z), \Phi_m(w) \rangle_{\mathcal{H}_m}, \qquad m \geq 1,$$

finite-matrix convergence to K_ξ*, and projection readout rows*

$$\pi_{\mathsf{T}}, \qquad \pi_{\mathsf{W}}, \qquad \pi_{\mathsf{N}}.$$

The projections read, respectively, the Li-time, Weil-energy, and Nyman-container observations. The row requires that every zero-relevant accepted observation used by the fixed-half route factors through this positive triangular system or through one of its finite source kernels before any zero-divisor conclusion is consumed.

Definition 9271.18 (Same-leg compatibility)**.** *Let ρ be a represented completed zero row and put*

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell(\rho)+i\theta(\rho)}.$$

The projections of Definition 9271.17 are same-leg compatible at ρ when the time, energy, and container readouts all use the same scalar

$$\ell(\rho) = \log\left|1 - \frac{1}{\rho}\right|$$

as their radial defect. Equivalently, a nonzero radial readout may not disappear under a change of projection: it must be visible as a Li-time finite witness, a Weil negative direction, a Nyman residual separator, or a displayed positive-kernel obstruction row.

Definition 9271.19 (Triangular residual exhaustion)**.** *Let*

$$\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots$$

be the Hilbert tower underlying Definition 9271.17. For an observation vector v, write

$$\eta_m(v) := (I - P_m)v$$

for the residual after projection to $\mathcal{H}_m$. The tower has triangular residual exhaustion for v when

$$\|\eta_m(v)\|^2 \to 0.$$

If this limit is not zero, the system must display a separating functional for the residual. In the Nyman projection this separating functional is the Mellin row of Definition 9281.8.

Theorem 9271.20 (Positive triangle controls represented observations)**.** *Assume* $\mathsf{ZetaPositiveTriangle}_\xi$*, same-leg compatibility for represented completed zero rows, triangular residual exhaustion for the accepted observation tower, and the usual reflection consumption row. Then every represented completed zero ledger that factors through this system has fixed-half readout.*

Proof. Let ρ be such a represented completed zero row. Suppose

$$\ell(\rho) \neq 0.$$

By same-leg compatibility, this same radial scalar must be visible in the Li-time, Weil-energy, and Nyman-container projections, or as a positive-kernel obstruction row. A finite Pick obstruction contradicts the positive definiteness of the kernels in Definition 9271.17. A Li-time or Weil-energy negative witness contradicts the corresponding positive side of the same Hilbert square ledger. A residual Nyman separator contradicts triangular residual exhaustion. Thus $\ell(\rho) \neq 0$ is impossible. Hence $\ell(\rho) = 0$, and Definition 9278.2 gives the fixed-half readout. □

Principle 9271.21 (Single positive triangle burden)**.** *The Li, Weil, Nyman, Lee–Yang, and finite endpoint rows are acceptable in the fixed-half route only as projections or finite models of one positive triangular observation system. The common burden is not to produce another isolated criterion, but to construct source kernels, same-leg projection readouts, and residual exhaustion in one package. When such a package is displayed, a nonzero radial scale cannot be hidden in prime fibres, cancelled by phase motion, or postponed to an unread infinite residual.*

Chapter 9272

Global Observation Triangle Invariant

chapter 9271 names the Pick-kernel form of the positive side, while chapter 9278 names the radial coordinate shared by the Li, Weil, and Nyman projections. The present chapter packages these rows as one global observation invariant. The target is not another isolated criterion. It is the source-side claim that every zero-relevant observation factors through one positive triangle before a fixed-half readout is consumed.

Definition 9272.1 (Global zeta observation triangle)**.** *The source row*

$$\mathsf{GlobalObservationTriangle}_\xi$$

consists of four compatible pieces:

$$\mathsf{Src}_\xi, \qquad \mathsf{Pos}_\xi, \qquad \mathsf{Leg}_\xi, \qquad \mathsf{Exh}_\xi.$$

The source piece Src_ξ *constructs the kernels* K_m *and maps* Φ_m *from prime, gamma, normalization, solenoid, golden, explicit-formula, or Stieltjes rows before the zero divisor is read. The positive piece* Pos_ξ *is the finite Hilbert square identity*

$$K_m(z,w) = \langle \Phi_m(z), \Phi_m(w) \rangle_{\mathcal{H}_m}$$

together with finite-matrix convergence to K_ξ*. The same-leg piece* Leg_ξ *says that the Li, Weil, Nyman, and Pick readouts use one radial scalar*

$$\ell(\rho) = \log \left| 1 - \frac{1}{\rho} \right|.$$

The exhaustion piece Exh_ξ *says that residual Hilbert energy is either exhausted by the observation tower or displayed by a separating functional.*

Definition 9272.2 (Legal global observation)**.** *A zero-relevant observation row* $\mathcal{O}$ *is legal for the global triangle when it factors through the source row of Definition 9272.1. Thus* $\mathcal{O}$ *is read either as a finite positive-kernel observation, as a limit of such observations on a finite matrix family, or as one of the displayed projection readouts of the same finite-to-limit system. A row that uses a completed zero divisor to define its Hilbert space, measure, or positive kernel is not a source-side legal observation for this purpose.*

Definition 9272.3 (Global triangle invariant)**.** *The global triangle invariant is the requirement that every legal zero-relevant observation satisfies the following three rows.*

First, the ledger row permits compact phase and compact prime-register fibres, as in Definition 9286.1, but does not permit an unrecorded $\mathbb{R}_{>0}$ radial gauge.

Second, the positive row represents observation energy by Hilbert squares:

$$\sum_{i,j} c_i \overline{c_j} K_m(z_i, z_j) = \left\| \sum_i c_i \Phi_m(z_i) \right\|_{\mathcal{H}_m}^2 \geq 0.$$

Third, the refinement row preserves finite matrix readouts and orthogonal residuals under passage to the limit. If a residual remains nonzero, it must be read by a displayed separator rather than hidden at the end of the tower.

Principle 9272.4 (Common radial leg)**.** *The Li, Weil, Nyman, Pick, Lee–Yang, and endpoint rows may have different finite presentations, but their off-fixed-half obstruction must read the same radial scalar. In Li time it appears as the radial factor in $z(\rho)^n$. In the Weil row it appears as a negative direction for a reflected orbit. In the Nyman row it appears as a Mellin separator for an unexhausted residual. In the Pick row it appears as a finite matrix obstruction. The global triangle invariant requires these readings to be transports of one scale leg, not separate defects that can be independently hidden.*

Theorem 9272.5 (Global triangle invariant gives Pick positivity)**.** *Assume* $\mathsf{GlobalObservationTriangle}_\xi$ *and assume its finite matrix convergence row holds for every displayed disk family. Then*

$$\mathsf{TriadicPickPositivity}_\xi$$

holds.

Proof. Fix a finite disk family $z_1, \ldots, z_N$ and coefficients $c_1, \ldots, c_N$. For every finite source level m, the positive piece of Definition 9272.1 gives

$$\sum_{i,j} c_i \overline{c_j} K_m(z_i, z_j) = \left\| \sum_i c_i \Phi_m(z_i) \right\|_{\mathcal{H}_m}^2 \geq 0.$$

The convergence row identifies the limit of the left side with

$$\sum_{i,j} c_i \overline{c_j} K_\xi(z_i, z_j).$$

Hence this latter finite quadratic form is nonnegative. Since the finite family was arbitrary, K_ξ is positive definite. □

Theorem 9272.6 (Global observation triangle gives fixed-half readout)**.** *Assume* $\mathsf{GlobalObservationTriangle}_\xi$*, the global triangle invariant, common radial-leg transport, triangular residual exhaustion, and the usual reflection consumption row for represented completed zero ledgers. Then every represented completed zero ledger legal for the global triangle has fixed-half readout.*

Proof. By Definition 9272.5, the global observation triangle supplies $\mathsf{TriadicPickPositivity}_\xi$. Let ρ be a represented completed zero ledger legal for the triangle. Suppose

$$\ell(\rho) \neq 0.$$

By common radial-leg transport, this nonzero scale leg must appear in one of the displayed projection rows: as a finite Li-time witness, a Weil negative direction, a Nyman residual separator,

or a Pick finite matrix obstruction. The positive row forbids negative finite Hilbert energy and Pick matrix obstructions. The residual exhaustion row forbids an unread Nyman residual; if such a residual remains, it is a displayed separator and therefore not hidden by the tower. The ledger row forbids reclassifying the same radial scalar as a compact phase or prime-register fibre. Thus $\ell(\rho) \neq 0$ is impossible. Hence $\ell(\rho) = 0$, and Definition 9278.2 gives the fixed-half readout. □

Definition 9272.7 (Zeta BEDC embedding row)**.** *The source row*

$$\mathsf{ZetaBEDCEmbedding}$$

asserts that every represented critical-strip completed zero row ρ is presented by a BEDC legal observation certificate

$$\mathcal{O}_\rho.$$

The certificate carries a carrier coordinate, a classifier coordinate, a ledger coordinate, the zero-row provenance, and the boundary rows needed to compare its Li, Weil, Nyman, and Pick projections. This row is stronger than a statement-level encoding of the RH predicate: it asserts that the analytic zero object has entered the BEDC observation system with the source data required by Definition 9272.1.

Definition 9272.8 (Zeta scale-leg coverage obstruction)**.** *The source row*

$$\mathsf{ZetaScaleLegCoverageObstruction}$$

asserts the following. Let ρ be a represented critical-strip completed zero row with

$$\ell(\rho) = \log\left|1 - \frac{1}{\rho}\right| \neq 0.$$

Then the associated radial scale leg cannot be covered by any of the three BEDC observation coordinates of Definition 9272.7: it is not a legal carrier coordinate, not a legal classifier coordinate, and not a legal ledger endpoint coordinate. It must therefore appear as an explicit defect row in the global observation triangle, rather than as hidden source structure.

Theorem 9272.9 (Embedding plus scale obstruction gives fixed-half readout)**.** *Assume* $\mathsf{ZetaBEDCEmbedding}$,

$$\mathsf{ZetaScaleLegCoverageObstruction},$$

and the global observation triangle hypotheses of Definition 9272.6. Then every represented critical-strip completed zero row in the embedding has fixed-half readout.

Proof. Let ρ be a represented critical-strip completed zero row. The embedding row supplies the BEDC legal observation certificate $\mathcal{O}_\rho$ and its carrier, classifier, ledger, and boundary coordinates. If $\ell(\rho) \neq 0$, the scale-leg obstruction row says that this radial coordinate cannot be absorbed by any of the three BEDC observation coordinates. Hence it remains an explicit defect row of the global observation triangle. Applying Definition 9272.6 to $\mathcal{O}_\rho$ rules out that explicit defect. Thus $\ell(\rho) = 0$, and Definition 9278.2 gives the fixed-half readout. □

Principle 9272.10 (Boundary of the BEDC route)**.** *The BEDC route does not consist merely of writing the RH statement in BEDC syntax. It requires the embedding row* $\mathsf{ZetaBEDCEmbedding}$ *and the obstruction row* $\mathsf{ZetaScaleLegCoverageObstruction}$. *Only then can the abstract kernel discipline of observation, distinction, time, carrier, classifier, and ledger be consumed by the analytic zeta object. The non-circular source burden is to prove that a nonzero radial scale leg cannot be one of the legal BEDC coordinates, rather than to assume fixed-half readout in advance.*

Definition 9272.11 (BEDC formalization success for zeta)**.** *The source row*

$$\mathsf{BEDCFormalizationSuccess}_\zeta$$

is the conjunction of four rows:

$$\mathsf{BEDCSoundness}, \quad \mathsf{ZetaInterpretation}, \quad \mathsf{ZetaCoverage}, \quad \mathsf{GlobalTriadicPositivity}_\xi.$$

Here BEDCSoundness *says that BEDC legal objects satisfy the carrier, classifier, ledger, unit-boundary, and generated-time discipline. The row* ZetaInterpretation *says that the completed zeta carrier, its centered channels, and its zero-relevant observations are interpreted inside that discipline. The row* ZetaCoverage *says that every represented non-trivial critical-strip zero is covered by such a legal observation certificate. The row* $\mathsf{GlobalTriadicPositivity}_\xi$ *says that the resulting global observation kernels have a source-side Hilbert/Pick/Herglotz positive representation.*

Definition 9272.12 (Strict sufficiency boundary)**.** *The strict sufficiency boundary is the separation between two claims. The weak claim is that BEDC supplies a formal language of observation, distinction, generated time, and ledgered residuals. The strong claim is that* $\mathsf{BEDCFormalizationSuccess}_\zeta$ *holds. Only the strong claim can be consumed by the fixed-half route, because only it asserts zeta interpretation, zero coverage, and global triadic positivity in addition to the abstract BEDC discipline.*

Theorem 9272.13 (Formalization success supplies fixed-half readout)**.** *Assume* $\mathsf{BEDCFormalizationSuccess}_\zeta$*, the scale-leg obstruction row*

$$\mathsf{ZetaScaleLegCoverageObstruction},$$

and the usual reflection consumption row for represented completed zero ledgers. Then every represented non-trivial critical-strip zero has fixed-half readout.

Proof. By ZetaInterpretation and ZetaCoverage, each represented non-trivial critical-strip zero is presented by a BEDC legal observation certificate inside the global zeta observation triangle. By $\mathsf{GlobalTriadicPositivity}_\xi$, the triangle carries the positive Hilbert/Pick/Herglotz row required by Definition 9272.6. By BEDCSoundness, the carrier, classifier, ledger, unit-boundary, and generated-time rows are legal BEDC rows. The scale-leg obstruction then prevents a nonzero radial scale from being absorbed by those legal coordinates. Applying Definition 9272.9 gives fixed-half readout for the represented zero. □

Principle 9272.14 (Strong meaning of formalization success)**.** *The phrase "formalization success" is used in this route only in the strong sense of Definition 9272.11. Formalizing the abstract BEDC principles gives the invariant language, but it does not by itself supply zeta interpretation, zero coverage, or the source-side positive representation*

$$K_\xi(z, w) = \langle \Phi_\xi(z), \Phi_\xi(w) \rangle.$$

The analytic core remains the construction of this positive global observation system from source data rather than from a posterior fixed-half zero divisor.

Definition 9272.15 (Xi observation cord)**.** *Let*

$$F(w) := \xi\left(\frac{1}{2} + w\right).$$

An Xi observation cord is a path

$$\gamma : [0,1] \to \{w : F(w) \neq 0\}$$

together with its image

$$F \circ \gamma : [0,1] \to \mathbb{C}^{\times}.$$

The cord has a radial readout $|F(\gamma(t))|$ *and a phase readout* $\arg F(\gamma(t))$ *on every branch where the argument is chosen. For a closed cord, its winding readout is*

$$\mathrm{Wind}_F(\gamma) := \frac{1}{2\pi i} \int_\gamma \frac{F'(w)}{F(w)}\, dw.$$

Definition 9272.16 (Off-axis cord defect)**.** *An off-axis cord defect is a closed Xi observation cord whose winding number is nonzero and whose enclosed zero row has centered coordinate*

$$w_0 = r + it$$

with $r \neq 0$*. Under the Li coordinate*

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta},$$

the same defect is read by the radial scalar

$$\ell = \log |z(\rho)| \neq 0.$$

Thus the off-axis cord defect is the topological form of the same scale leg appearing in Definition 9272.8.

Definition 9272.17 (Li-framed observation cylinder)**.** *The Li-framed observation cylinder is*

$$\mathcal{C}_{\mathrm{Li}} := S^1_\theta \times \mathbb{R}_\ell.$$

A completed zero row with

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}$$

has phase centre line θ *and scale framing* ℓ*. The core circle is*

$$\mathcal{U}_{\mathrm{Li}} := S^1_\theta \times \{0\}.$$

Discrete Li time sends the framed coordinate to

$$z(\rho)^n = e^{n\ell} e^{in\theta},$$

so the phase coordinate winds on S^1_θ *while the framing coordinate is multiplied by* n*.*

Definition 9272.18 (Framed Xi cord)**.** *A framed Xi cord is an Xi observation cord whose enclosed completed zero row is equipped with the Li-framed lift of Definition 9272.17. It is core-framed when its scale framing is zero:*

$$\ell = 0.$$

It is nonunit-framed when

$$\ell \neq 0.$$

The framing is part of the observation ledger: it may be transported between Li, Weil, Nyman, and Pick readouts only as the same radial coordinate, not as a separately chosen gauge.

Theorem 9272.19 (Off-axis cords are nonunit-framed cords)**.** *Among closed represented Xi cords with nonzero winding readout, an off-axis cord defect is exactly a framed Xi cord with nonunit scale framing.*

Proof. The winding condition is common to both sides of the statement. The remaining distinction is the radial position of the enclosed zero row. An off-axis cord defect has centered coordinate

$$w_0 = r + it$$

with $r \neq 0$. Under the Li coordinate, the same row is read by

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

By Definition 9278.2, the condition $\ell = 0$ is precisely fixed-half readout, and hence $r = 0$ on the represented zero row. Therefore $r \neq 0$ is equivalent to $\ell \neq 0$, which is the nonunit-framed condition of Definition 9272.18. □

Definition 9272.20 (Framed Li closure defect)**.** *For a framed Xi cord with Li coordinate*

$$z = e^{\ell + i\theta},$$

the nth framed Li closure defect is

$$\Delta_n(z) := 1 - z^n.$$

Its squared size decomposes as

$$|\Delta_n(z)|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

The first summand is the framing-energy row, and the second summand is the phase-winding row.

Principle 9272.21 (Framing energy cannot be cancelled by phase winding)**.** *The decomposition of Definition 9272.20 is a Pythagorean readout in the Li-framed cylinder. A nonzero scale framing contributes the nonnegative term*

$$(e^{n\ell} - 1)^2.$$

The phase row contributes

$$2e^{n\ell}(1 - \cos(n\theta)).$$

Since these terms occupy orthogonal readout rows, phase winding cannot erase a nonzero framing-energy row. The legal route must either display that energy in a projection ledger or rule out the nonunit framing at the source side.

Definition 9272.22 (Zeta framed cord triviality)**.** *The source row*

$$\mathsf{ZetaFramedCordTriviality}$$

asserts that every represented closed Xi observation cord legal for the global triangle and carrying nonzero winding readout is core-framed in the sense of Definition 9272.18. Equivalently, every such cord has

$$\ell = 0.$$

This is a source-side triviality row for the Li framing; it does not assert that all analytic zero rows are already covered by the global triangle.

Theorem 9272.23 (Framed cord triviality gives fixed-half readout)**.** *Assume* ZetaFramedCordTriviality *and the zero-coverage row that represents each completed critical-strip zero by a legal closed Xi observation cord with nonzero winding readout enclosing its zero row. each represented completed critical-strip zero has fixed-half readout.*

Proof. Let ρ be a represented completed critical-strip zero. The zero-coverage row supplies a legal closed Xi observation cord whose winding readout encloses the zero row of ρ. By ZetaFramedCordTriviality, the associated framed cord is core-framed, so its Li scale coordinate satisfies

$$\ell(\rho) = 0.$$

By Definition 9278.2, this is precisely fixed-half readout for ρ. □

Principle 9272.24 (Cord invariants are triadic invariants)**.** *The three sides of the global observation triangle read an off-axis cord defect as three invariants. The ledger side reads the fibre type: a radial factor belongs to $\mathbb{R}_{>0}$ rather than to compact phase or prime-register fibres. The time side reads finite detectability: the same defect supplies a finite Pick or Li–Weil witness. The unit side reads Hilbert positivity: a source-side factorization of K_ξ would make the corresponding finite quadratic form a norm square. Hence an off-axis cord defect cannot be hidden as a merely topological loop; it must pass the same carrier, classifier, ledger, positivity, and refinement audits as every other legal observation.*

Definition 9272.25 (Source unknotting for xi)**.** *The source row*

$$\mathsf{SourceUnknotting}_\xi$$

asserts that the Xi observation cords legal for the global triangle have a source-side Hilbert–Pick factorization:

$$K_\xi(z, w) = \langle \Phi_\xi(z), \Phi_\xi(w) \rangle.$$

The map Φ_ξ must be constructed from source data and must respect the projection readouts of Definition 9272.1. In this terminology, "unknotting" means positive Hilbert realization plus same-leg transport; it is not an external topological operation on a completed zero set.

Theorem 9272.26 (Source unknotting excludes off-axis cord defects)**.** *Assume* $\mathsf{SourceUnknotting}_\xi$*, common radial-leg transport, the scale-leg obstruction row*

$$\mathsf{ZetaScaleLegCoverageObstruction},$$

and the usual reflection consumption row. Then no represented legal Xi observation cord has an off-axis cord defect.

Proof. The source unknotting row gives

$$K_\xi(z, w) = \langle \Phi_\xi(z), \Phi_\xi(w) \rangle,$$

so every finite Pick quadratic form is a Hilbert norm square and is nonnegative. If an off-axis cord defect were represented, then by Definition 9272.16 it would carry a nonzero radial scalar ℓ. Common radial-leg transport reads this scalar in the finite Pick witness row, while Definition 9272.8 prevents it from being absorbed as a legal carrier, classifier, or ledger coordinate. Thus the defect would have to appear as a negative finite Pick witness, contradicting the Hilbert norm-square representation. Therefore no represented legal cord has such a defect. □

Principle 9272.27 (Global observation source load)**.** *The mathematical load is concentrated in the construction of the source piece* Src_ξ *and the convergence piece of Definition 9272.1. The positive side is then Hilbert-square formal, the time side is finite-matrix witnessability, and the ledger side is the refusal of an unrecorded real-positive scale gauge. This is the global form of the triangle route: every legal zeta observation is constrained by one positive Hilbert–Pick–Herglotz system, and every nonzero radial scale must be read rather than hidden.*

Chapter 9273

Xi Log-Potential No-Scale Field

chapter 9274 records the observation-vector form of the scale defect. The same obstruction has a global analytic readout through the centered completed zeta potential. This chapter names that readout and connects it to the Herglotz and Stieltjes targets of chapter 9270.

Definition 9273.1 (Centered xi log-potential)**.** *Let*

$$F_\xi(w) := \xi\left(\frac{1}{2} + w\right), \qquad w = x + iy.$$

On the nonzero locus of F_ξ, define the log-potential

$$u_\xi(w) := \log|F_\xi(w)|.$$

The coordinate x is the centered scale direction, and y is the centered phase-time direction.

Definition 9273.2 (Xi log-potential gradient energy)**.** *On the nonzero locus of F_ξ, define the observation gradient row*

$$\nabla u_\xi(w) := (\partial_x u_\xi(w), \partial_y u_\xi(w)).$$

Its Pythagorean energy decomposition is

$$\mathcal{E}_\xi(w) := |\nabla u_\xi(w)|^2 = (\partial_x u_\xi(w))^2 + (\partial_y u_\xi(w))^2.$$

The scale component is

$$\mathcal{E}_{\xi,\mathrm{scale}}(w) := (\partial_x u_\xi(w))^2.$$

Principle 9273.3 (Holomorphic coordinates are orthogonal)**.** *Where a local branch of* $\log F_\xi$ *exists, write*

$$\log F_\xi = u_\xi + iv_\xi.$$

The Cauchy–Riemann equations give the orthogonality of the potential and phase gradients. Thus the centered analytic carrier supplies the same orthogonal scale and phase coordinates as the observation-vector row of Definition 9274.6.

Definition 9273.4 (Xi no-scale-demon row)**.** *The source row*

$$\mathsf{XiNoScaleDemon}$$

asserts

$$\partial_x u_\xi(x + iy) \geq 0 \qquad (x > 0,\ F_\xi(x + iy) \neq 0).$$

Equivalently,

$$\operatorname{Re}\frac{F_\xi'(w)}{F_\xi(w)} \geq 0 \qquad (\operatorname{Re}(w) > 0).$$

Under $s = 1/2 + w$, this is the positive-real row XiLogDerivativeHerglotz *of Definition 9270.2.*

Theorem 9273.5 (Off-fixed-half zero creates negative scale flow)**.** *Suppose F_ξ has a zero $w_0 = x_0 + iy_0$ of multiplicity $m \geq 1$ with $x_0 > 0$. Then $\partial_x u_\xi(w) < 0$ at points of the right half-plane arbitrarily close to w_0.*

Proof. Near w_0,

$$F_\xi(w) = (w - w_0)^m g(w), \qquad g(w_0) \neq 0.$$

Hence

$$u_\xi(w) = m \log|w - w_0| + \log|g(w)|.$$

Take $w = w_0 - \varepsilon$ with $\varepsilon > 0$ real and sufficiently small. Then w remains in the right half-plane and

$$\partial_x u_\xi(w) = m\frac{x - x_0}{|w - w_0|^2} + O(1) = -\frac{m}{\varepsilon} + O(1),$$

which is negative for small enough ε. □

Theorem 9273.6 (No-scale-demon row gives fixed-half readout)**.** *Assume* XiNoScaleDemon *and the usual reflection consumption row for the completed zeta zero ledger. Then the completed zeta zero ledger has fixed-half readout.*

Proof. If a completed zero had centered coordinate w_0 with $\operatorname{Re}(w_0) > 0$, then Definition 9273.5 would give a right-half-plane point where

$$\partial_x u_\xi < 0,$$

contradicting XiNoScaleDemon. The reflection consumption row excludes the opposite open side by reflecting any such zero into the right half-plane. Thus every accepted completed zero reads through the fixed-half surface. □

Theorem 9273.7 (Stieltjes source row supplies no-scale demon)**.** *If* XiStieltjesLogDerivative *holds, then*

XiNoScaleDemon

holds.

Proof. By Definition 9270.14, XiStieltjesLogDerivative gives XiLogDerivativeHerglotz. Under the centered coordinate $s = 1/2 + w$, this is exactly the positive-real inequality in Definition 9273.4. □

Principle 9273.8 (Log-potential field as the global Pythagorean target)**.** *The log-potential row is the global field version of the finite observation-vector identity. A local active mode records the scale leg as $-rh_a$ and yields the finite energy $r^2 W_m$; the centered potential records the same leg as $\partial_x \log|F_\xi|$. A zero away from the fixed line is therefore a scale source or sink in the potential field. The source-side positive spectral target is*

XiStieltjesLogDerivative $\Longrightarrow$ XiNoScaleDemon $\Longrightarrow$ *fixed-half zero readout.*

Chapter 9274

Triadic Pythagorean Obstruction

Definition 9257.20 records the three certificate rows P, U, and T level by level. chapter 9270 records the unit-positive analytic face, while chapter 9286 records the no-scale energy face. This chapter packages these rows as one obstruction rather than as independent routes. The obstruction says that an off-fixed-half readout must appear simultaneously as an illegal prime-ledger scale, a finite-time witness, and a violation of the positive spectral row.

Definition 9274.1 (Observation ledger)**.** *An observation ledger is an increasing family of finite records*

$$\Lambda_0 \subseteq \Lambda_1 \subseteq \Lambda_2 \subseteq \cdots ,$$

where each record in Λ_m has the form

$$o = (p, r, \varepsilon)$$

for a displayed probe p, a displayed result r, and a tolerance or error row ε. The ledger is append-only: later levels may refine or extend the record, but they may not erase an earlier source row without a displayed transport ledger.

Principle 9274.2 (Observation order as internal time)**.** *The order carried by Definition 9274.1 is the observation-side form of internal time. The relation*

$$\Lambda_m \subseteq \Lambda_{m+1}$$

is not an external clock parameter; it is the generated order of recorded distinctions. If the inclusion is strict along a level, no forward append operation returns the ledger to the earlier finite record.

Definition 9274.3 (Residual coordinate from an observation)**.** *Let*

$$\mathcal{H}_m := \operatorname{span}\{v_1, \ldots, v_m\}$$

be the observation subspace generated by the first m ledger vectors, and let P_m denote orthogonal projection onto $\mathcal{H}_m$. A further observation vector v_{m+1} has the residual decomposition

$$v_{m+1} = P_m v_{m+1} + \eta_{m+1}, \qquad \eta_{m+1} \perp \mathcal{H}_m.$$

The projected term is the part explained by the previous observation coordinates. The residual η_{m+1} is the conditionally unexplained row. When $\eta_{m+1} \neq 0$, its normalized direction

$$e_{m+1} := \frac{\eta_{m+1}}{\|\eta_{m+1}\|}$$

is the additional orthogonal coordinate generated by the observation.

Theorem 9274.4 (Residual Pythagorean identity)**.** *For the decomposition of Definition 9274.3,*

$$\|v_{m+1}\|^2 = \|P_m v_{m+1}\|^2 + \|\eta_{m+1}\|^2.$$

Proof. The two summands are orthogonal. Expanding the squared norm gives

$$\|P_m v_{m+1} + \eta_{m+1}\|^2 = \|P_m v_{m+1}\|^2 + \|\eta_{m+1}\|^2.$$

□

Principle 9274.5 (Residual coordinate conservation)**.** *Observation order supplies the ledger time, while residual projection supplies the coordinate creation rule. A coordinate not already explained by the ledger cannot be removed by changing phase notation; it must either be recorded as an orthogonal residual or be ruled out by a closed-system invariant. The scale term in the zeta packet is such a residual coordinate.*

Definition 9274.6 (Observation vector decomposition)**.** *Let an active zero packet at level m have active set A_m and heights $h_a > 0$. For a centered readout*

$$s = \frac{1}{2} + r + it,$$

define the observation vector in the real Hilbert row

$$\mathbb{R}^{A_m} \oplus \mathbb{R}^{A_m}$$

by

$$V_m(s) := ((-rh_a)_{a\in A_m}, (-th_a)_{a\in A_m}).$$

Its scale and phase components are

$$V_{m,\mathrm{scale}}(s) := (-rh_a)_{a\in A_m}, \qquad V_{m,\mathrm{phase}}(s) := (-th_a)_{a\in A_m}.$$

The two summands are orthogonal coordinate rows: the first records radial scale and the second records phase lift.

Theorem 9274.7 (Observation vector Pythagorean identity)**.** *For the observation vector of Definition 9274.6, if*

$$W_m := \sum_{a\in A_m} h_a^2,$$

then

$$\|V_m(s)\|^2 = r^2 W_m + t^2 W_m,$$

and its scale component satisfies

$$\|V_{m,\mathrm{scale}}(s)\|^2 = r^2 W_m.$$

Proof. The two coordinate summands of $\mathbb{R}^{A_m} \oplus \mathbb{R}^{A_m}$ are orthogonal. Therefore

$$\|V_m(s)\|^2 = \sum_{a\in A_m} r^2 h_a^2 + \sum_{a\in A_m} t^2 h_a^2 = r^2 W_m + t^2 W_m.$$

The displayed formula for the scale component is the first summand. □

Principle 9274.8 (Observation, orthogonality, invariant)**.** *The observation ledger supplies the internal order, orthogonal decomposition supplies independent coordinates, and the Pythagorean identity supplies the quadratic invariant. In the zeta packet the phase coordinate is a legal unit-circle row, while the scale coordinate is a real-positive row. Thus the closed-packet obstruction is not the total length of $V_m(s)$, but the scale energy*

$$\|V_{m,\mathrm{scale}}(s)\|^2 = r^2 W_m.$$

This is the observation-vector form of the defect energy later recorded in Definition 9274.11.

Definition 9274.9 (Triadic obstruction coordinates)**.** *A triadic obstruction coordinate system for an active zero packet consists of three source rows:*

$$\mathsf{P}, \qquad \mathsf{U}, \qquad \mathsf{T}.$$

The row P is the prime-ledger row: hidden coordinates may live in compact phase and prime-register fibres, as in Definition 9286.1, but no independent real-positive scale ledger is accepted as a gauge. The row U is the unit-positive row: it may be presented as Lee–Yang boundary stability, global Weil positivity, Li CND, Herglotz positivity, or the Stieltjes source row of Definition 9270.13. The row T is the finite-time row: any off-fixed-half defect must project to a finite Li–Weil witness of the kind isolated in Definition 9265.22, rather than to an open-ended search through certificate levels.

Definition 9274.10 (Triadic closure relation)**.** *For a zero-relevant active packet with heights $h_a > 0$, write the centered readout as*

$$s = \frac{1}{2} + r + it.$$

The active channel at mode a is

$$U_a(s) := e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a}.$$

The triadic closure relation

$$\mathcal{C}(\mathsf{P}, \mathsf{U}, \mathsf{T}; s) = 0$$

means that the same active packet satisfies the prime-ledger gauge row, the unit-positive boundary row, and the finite-time witness row without introducing an unledgered $\mathbb{R}_{>0}$ coordinate.

Definition 9274.11 (Triadic scale defect energy)**.** *For a finite active packet A_m, define the scale defect at mode a by*

$$d_a(s) := \log |U_a(s)| = -rh_a.$$

The triadic defect vector and its quadratic energy are

$$\Omega_m(s) := (d_a(s))_{a \in A_m},$$

and

$$\mathcal{E}_m(s) := \sum_{a \in A_m} d_a(s)^2.$$

With

$$W_m := \sum_{a \in A_m} h_a^2,$$

one has the exact finite identity

$$\mathcal{E}_m(s) = r^2 W_m.$$

Principle 9274.12 (Pythagorean reading of the defect)**.** *The equation of Definition 9274.11 is the quadratic obstruction behind the triadic route. Linear phase cancellation may make a projected sum vanish, and reflection may exchange r with $-r$, but the squared defect*

$$r^2 W_m$$

has no sign that can be cancelled by phase or by mirror exchange. Thus an off-fixed-half readout can be hidden only by supplying an external scale ledger, by losing the finite witness row, or by breaking the unit-positive row.

Theorem 9274.13 (Triadic Pythagorean obstruction)**.** *Let a zero-relevant certificate tower have limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

Assume that it satisfies the triadic closure relation of Definition 9274.10, has cofinal active-weight growth

$$W_m \to \infty,$$

has defect extraction

$$\mathcal{E}_m(s_\infty) = r^2 W_m + o(W_m),$$

and carries a unit-positive stability bound

$$\mathcal{E}_m(s_\infty) \le B$$

with B independent of m. Then

$$r = 0.$$

Proof. Assume $r \neq 0$. Then $r^2 > 0$. From the defect extraction row there is a cofinal tail on which

$$|o(W_m)| \le \frac{r^2}{2} W_m.$$

Hence

$$\mathcal{E}_m(s_\infty) \ge \frac{r^2}{2} W_m$$

on that tail. Since $W_m \to \infty$, the energies $\mathcal{E}_m(s_\infty)$ are unbounded along the tail. This contradicts the stability bound $\mathcal{E}_m(s_\infty) \le B$. Therefore $r = 0$. □

Corollary 9274.14 (Triadic obstruction gives fixed-half readout)**.** *If every accepted non-trivial completed zeta zero is represented by a zero-relevant certificate tower satisfying the hypotheses of Definition 9274.13, then every such accepted zero has fixed-half readout.*

Proof. Let u be an accepted non-trivial completed zeta zero and let

$$s_\infty = \frac{1}{2} + r + it$$

be its represented readout. The tower hypotheses give $r = 0$ by Definition 9274.13. Thus $\mathrm{Re}(s_\infty) = 1/2$. □

Definition 9274.15 (Triadic positive-spectrum closure)**.** *The source row*

$$\mathsf{TriadicPositiveSpectrumClosure}_\zeta$$

asserts that the unit row U *of Definition 9274.9 is carried by one of the positive spectral presentations already named in the RH route:*

$\mathsf{XiStieltjesLogDerivative}$, $\mathsf{XiLogDerivativeHerglotz}$, $\mathsf{LiSecondDifferencePositiveDefinite}$, $\mathsf{GlobalLiWeilPosit}$

The row is source-side: it supplies positivity before the zero divisor is read, and it may not be manufactured by first assuming fixed-half readout and then placing a posterior spectral measure.

Theorem 9274.16 (Positive spectrum supplies the unit side)**.** *If* $\mathsf{TriadicPositiveSpectrumClosure}_\zeta$ *is realized through* $\mathsf{XiStieltjesLogDerivative}$*, then the unit row* U *of the triadic obstruction is available.*

Proof. Definition 9270.14 turns $\mathsf{XiStieltjesLogDerivative}$ into $\mathsf{XiLogDerivativeHerglotz}$. The latter is the positive-real unit row of Definition 9270.2; equivalently, by the Li–Weil comparison of Definition 9270.3, it is the analytic face of the Li beta positivity row. Hence the unit-positive component of the triadic closure relation is supplied. □

Principle 9274.17 (Three rows as one obstruction)**.** *The triadic route should be consumed as a single obstruction. The prime-ledger row says that* e^{-rh_a} *cannot be hidden in the compact phase or prime-register fibre. The finite-time row says that any such scale defect must have a finite Li–Weil witness. The unit-positive row says that the relevant witness functional is nonnegative through Weil, Li, Herglotz, or Stieltjes positivity. Therefore a nonzero real offset cannot be discharged by moving among the three rows; it is the same defect seen in three projections.*

Definition 9274.18 (Analytic triadic closure)**.** *The analytic specialization of the triadic obstruction has three rows:*

$$\mathsf{T}, \qquad \mathsf{W}, \qquad \mathsf{N}.$$

The time row T *is* $\mathsf{GlobalLiWeilPositivity}$*, or any source-side stronger row such as Li CND or a Stieltjes log-derivative representation that implies it. The energy row* W *is* $\mathsf{GlobalWeilPositivity}$ *for the Weil quadratic form of Definition 9265.2. The container row* N *is* $\mathsf{NymanBeurlingExhaustion}$ *of Definition 9281.6. The analytic triadic closure row*

$$\mathsf{AnalyticTriadicClosure}_\zeta$$

asserts that these three rows are supplied by one source carrier rather than by three posterior arguments over an already-read zero divisor.

Theorem 9274.19 (Li coordinate Pythagorean decomposition)**.** *Let*

$$z = e^{\ell + i\theta}$$

with $\ell, \theta \in \mathbb{R}$*. For every* $n \geq 1$*,*

$$|1 - z^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

Proof. Since

$$z^n = e^{n\ell}(\cos(n\theta) + i\sin(n\theta)),$$

one has

$$|1 - z^n|^2 = (1 - e^{n\ell}\cos(n\theta))^2 + (e^{n\ell}\sin(n\theta))^2.$$

Expanding and using

$$\cos^2(n\theta) + \sin^2(n\theta) = 1$$

gives

$$|1 - z^n|^2 = 1 - 2e^{n\ell}\cos(n\theta) + e^{2n\ell}.$$

The right-hand side also equals

$$(e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)),$$

which proves the identity. □

Principle 9274.20 (Radial Li defect is the time projection)**.** *For a zero row ρ, the Li coordinate of Definition 9266.1 has polar form*

$$z(\rho) = e^{\ell + i\theta}.$$

By Definition 9266.2, the fixed-half readout is exactly $\ell = 0$. If $\ell \neq 0$, the radial factor $e^{n\ell}$ is amplified by Li time. The real reflected-pair contribution is the form already isolated in Definition 9266.6: along suitable finite times, the radial term produces a finite Li–Weil negative witness unless the source carrier supplies a positive row that forbids it.

Definition 9274.21 (Weil negative direction projection)**.** *Let ρ_+ and ρ_- be a reflected off-fixed-half pair in the completed zero ledger. A Weil negative direction projection is a test-row packet f whose Weil transforms separate the reflected pair with opposite signs, while the remaining displayed zero rows are controlled by the interpolation and tail ledgers of the Weil packet. It is the zero-side obstruction described by Definition 9265.15: an off-fixed-half pair creates a negative direction unless* GlobalWeilPositivity *forbids such a packet.*

Definition 9274.22 (Nyman residual projection)**.** *Let s_0 be a zero with $\mathrm{Re}(s_0) > 1/2$. The Nyman residual projection is the continuous Mellin functional*

$$L_{s_0}(f) := \int_0^1 f(x)x^{s_0 - 1}\,dx$$

of Definition 9281.8. It annihilates every Nyman generator $\rho_{1/n}$ and satisfies

$$L_{s_0}(\mathbf{1}) = \frac{1}{s_0} \neq 0.$$

Thus it separates the constant row from the generated Nyman closure whenever the Nyman exhaustion row is absent.

Theorem 9274.23 (Nyman separator gives a quantitative residual)**.** *Assume the Nyman Mellin bridge row. If $s_0 = \sigma + i\tau$ is a nontrivial zeta zero with $\sigma > 1/2$, then*

$$\mathrm{dist}_{L^2}\left(\mathbf{1}, \overline{\mathrm{span}\{\rho_{1/n} : n \geq 1\}}^{L^2(0,1)}\right) \geq \frac{\sqrt{2\sigma - 1}}{|s_0|}.$$

Proof. By Definition 9281.8, the functional L_{s_0} vanishes on every generator $\rho_{1/n}$ and $L_{s_0}(\mathbf{1}) = 1/s_0$. Its representing vector is $x^{\overline{s_0}-1}$, hence

$$\|L_{s_0}\| = \left(\int_0^1 x^{2\sigma-2}\,dx\right)^{1/2} = \frac{1}{\sqrt{2\sigma-1}}.$$

For any v in the closed Nyman span,

$$|L_{s_0}(\mathbf{1}-v)| = |L_{s_0}(\mathbf{1})| = \frac{1}{|s_0|}.$$

Therefore

$$\|\mathbf{1}-v\|_2 \geq \frac{|L_{s_0}(\mathbf{1}-v)|}{\|L_{s_0}\|} = \frac{\sqrt{2\sigma-1}}{|s_0|}.$$

Taking the infimum over v gives the claim. □

Principle 9274.24 (Analytic triangle as one scale defect)**.** *The rows* T, W, *and* N *are not three unrelated proof attempts. They are three projections of the same nonunit Li coordinate. In the time projection, the radial parameter* $\ell = \log|z(\rho)|$ *is amplified by* $z(\rho)^n$ *and appears as a finite Li–Weil negative witness. In the energy projection, the reflected off-fixed-half pair supplies a negative Weil direction unless global Weil positivity forbids it. In the container projection, the Mellin functional* L_{s_0} *separates* $\mathbf{1}$ *from the Nyman closure unless Nyman exhaustion forbids the residual. Thus a nonzero scale coordinate is not a local anomaly that can be moved from one row to another; it is the same obstruction read by time, energy, and container coordinates.*

Theorem 9274.25 (Analytic triadic closure gives fixed-half readout)**.** *Assume* $\mathsf{AnalyticTriadicClosure}_\zeta$ *and the usual reflection consumption rows for the completed zeta zero ledger. Then every accepted non-trivial completed zeta zero has fixed-half readout.*

Proof. The row $\mathsf{AnalyticTriadicClosure}_\zeta$ includes $\mathsf{NymanBeurlingExhaustion}$. By Definition 9281.9, the Nyman row and the reflection consumption row exclude zero readouts off the fixed-half surface. The Li and Weil rows give the same exclusion through their finite-witness and energy forms; they are recorded here to identify that the excluded residual is the same scale defect as in Definition 9274.24. Hence every accepted non-trivial completed zero has fixed-half readout. □

Principle 9274.26 (Ontic closure target for the analytic triangle)**.** *The remaining source-side target is not to prove the Li, Weil, and Nyman rows by unrelated searches. It is to show that BEDC triadic closure supplies them as one source carrier:*

$$\mathsf{OnticTriadicClosure}_\zeta \Longrightarrow \mathsf{GlobalLiWeilPositivity} \wedge \mathsf{GlobalWeilPositivity} \wedge \mathsf{NymanBeurlingExhaustion}.$$

Once this carrier is present, an off-fixed-half readout has no place to hide: Li time detects it finitely, Weil energy detects it as a negative direction, and the Nyman container detects it as a positive residual distance.

Chapter 9275

Fermat Triangle Obstruction Prototype

chapter 9274 isolates the RH-facing scale defect as a quadratic residual, and chapter 9290 records the prime-register boundary behind triangle closure. The present chapter records the Fermat-shaped prototype for that obstruction. The purpose is not to import Fermat's Last Theorem as a BEDC theorem. It is to separate the legal quadratic triangle row from higher-power displays that require a separate branch, scale, ideal, or arithmetic ledger before they can be consumed.

9275.1 Quadratic Triangle Closure

Definition 9275.1 (Primitive orthogonal triangle closure)**.** *A primitive orthogonal triangle closure is a displayed packet*

$$X \perp Y, \qquad Z = X \oplus Y,$$

together with the quadratic readout

$$E(V) := \|V\|^2$$

and the energy ledger

$$E(Z) = E(X) + E(Y).$$

In BEDC terms, the packet is legal only when the source rows carrying X, Y, the orthogonality witness, the direct-sum readout, and the quadratic energy readback all remain visible in the ledger.

The residual identities of Definition 9274.4 and Definition 9274.7 are instances of Definition 9275.1. They are the formal reason why the observation route naturally produces square energy: orthogonal residuals add through the norm square, not through an arbitrary power.

Principle 9275.2 (Square energy as orthogonal information readout)**.** *The square in a Pythagorean row is not a decorative exponent. It is the energy readout of an orthogonal decomposition. When a generated object has the form*

$$B_{m+1} = B_m \oplus R_{m+1}, \qquad R_{m+1} \perp B_m,$$

the ledgered invariant is

$$\|B_{m+1}\|^2 = \|B_m\|^2 + \|R_{m+1}\|^2.$$

This is the quadratic surface on which a circle readout, a right-triangle readout, and a residual scale audit can be compared.

Theorem 9275.3 (Displayed L^q parallelogram obstruction)**.** *Let $q > 0$ and let*

$$\|(a, b)\|_q = (|a|^q + |b|^q)^{1/q}$$

on the displayed two-coordinate row. If this readout satisfies the parallelogram identity on

$$x = (1, 1), \qquad y = (1, -1),$$

then $q = 2$.

Proof. For these two displayed vectors,

$$x + y = (2, 0), \qquad x - y = (0, 2).$$

Hence

$$\|x + y\|_q^2 + \|x - y\|_q^2 = 4 + 4 = 8.$$

Also

$$\|x\|_q^2 = \|y\|_q^2 = 2^{2/q},$$

so the right side of the parallelogram identity is

$$2\|x\|_q^2 + 2\|y\|_q^2 = 4 \cdot 2^{2/q}.$$

The parallelogram identity therefore gives

$$8 = 4 \cdot 2^{2/q},$$

equivalently

$$2 = 2^{2/q}.$$

Since the displayed exponential base is greater than 1, the exponent comparison gives $2/q = 1$, hence $q = 2$. □

Principle 9275.4 (Quadratic-only primitive triangle)**.** *A primitive orthogonal triangle closure must pass through the parallelogram-compatible norm-square row. A higher-power display may be a valid arithmetic, algebraic, or analytic object, but it is not a primitive orthogonal triangle closure unless its proof ledger supplies a quadratic inner-product carrier from which the display is derived.*

9275.2 Fermat Classifiers

Definition 9275.5 (Gaussian two-square classifier row)**.** *The Gaussian two-square classifier is the specialization of the prime-classifier container of Definition 9290.7 to the norm*

$$N(a + bi) = a^2 + b^2.$$

Its inert boundary is the family of rational primes

$$p \equiv 3 \pmod 4,$$

and its finite boundary readout is

$$\partial_\square(n) := \big(v_p(n) \bmod 2\big)_{p \equiv 3 \pmod 4}.$$

The displayed closure row

$$\partial_\square(n) = 0$$

states that every inert prime class occurs with even multiplicity.

Principle 9275.6 (Fermat two-square theorem as quadratic classifier)**.** *The classical two-square theorem is the positive quadratic prototype: an integer n is a sum of two squares exactly when the inert boundary of Definition 9275.5 vanishes. For primes this says that $p = a^2 + b^2$ exactly for $p = 2$ or $p \equiv 1 \pmod 4$. BEDC reads this not merely as an equation, but as a prime-register classification of when a quadratic norm packet has no unpaired inert boundary.*

Definition 9275.7 (Higher-power pseudo-triangle row)**.** *For $n > 2$, a displayed equation*

$$x^n + y^n = z^n$$

is a higher-power pseudo-triangle row when it is offered as if it were the primitive triangle closure of Definition 9275.1. The row is accepted as a BEDC object only after its actual source ledger is exposed: cyclotomic branches, ideal factors, unit rows, class-group data, elliptic or Galois carriers, or any other arithmetic structure needed by the display. Without such a ledger it is not a primitive triangle row.

Principle 9275.8 (Fermat last theorem as high-power closure refusal)**.** *In this vocabulary, Fermat's Last Theorem says that the integer prime register has no non-trivial high-power pseudo-triangle closure:*

$$x^n + y^n = z^n, \qquad n > 2,$$

has no nonzero integer solution. The BEDC reading is narrower than the historical theorem and does not replace its proof: a higher exponent cannot silently assume the role carried by square energy in a primitive orthogonal triangle packet.

For an odd prime exponent ℓ, the algebraic display already shows the difference in carrier:

$$x^\ell + y^\ell = \prod_{k=0}^{\ell-1} (x + \zeta_\ell^k y).$$

This is a many-branch cyclotomic ledger, not the two-factor norm row

$$(a + bi)(a - bi) = a^2 + b^2.$$

The higher-power display therefore belongs to a richer arithmetic container before it belongs to any triangle-facing consumer.

9275.3 RH Radial Scale as Fermat-Type Defect

Definition 9275.9 (Fermat-type triangle obstruction)**.** *A Fermat-type triangle obstruction is a displayed closure attempt in which a non-quadratic or non-unit component is presented as if it were a primitive orthogonal triangle row. Its BEDC audit asks for the missing ledger:*

$$\text{quadratic carrier} \quad \textit{or} \quad \text{external residual row.}$$

If the quadratic carrier is absent and the residual row is omitted, the closure is not generated.

The RH route has the same shape after the Li coordinate is introduced. For a zero row ρ, write

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

By Definition 9274.19,

$$|1 - z(\rho)^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

The second summand is the phase-facing term. The first summand is a radial scale leg. If $\ell \neq 0$, this leg is positive along suitable displayed times and is not a unit-circle phase coordinate.

Principle 9275.10 (Radial scale is not phase closure)**.** *The off-fixed-half condition $\ell \neq 0$ is the RH-facing analogue of a higher-power pseudo-triangle row. It is a syntactically visible closure attempt, but the radial factor cannot be consumed as unit phase or as a quadratic triangle invariant without displaying a scale ledger. Thus the route from Definition 9331.3 to Definition 9275.9 is:*

$$\ell \neq 0 \quad \Longrightarrow \quad \text{radial residual} \quad \Longrightarrow \quad \text{not pure unit triangle closure.}$$

Definition 9275.11 (NoFakeTriangleClosure target)**.** *The target row*

NoFakeTriangleClosure

has four clauses.

- *Every primitive orthogonal triangle closure factors through square energy and a parallelogram-compatible carrier.*
- *Every higher-power arithmetic display is either accounted for by its own branch and prime-register ledger or refused as a primitive triangle closure.*
- *Every zeta Li packet with $\ell \neq 0$ displays a radial scale row rather than a pure unit phase row.*
- *A cofinal prime classifier detects any unpaired inert boundary, radial residual, or scale ledger that a finite visible phase readout cannot erase.*

Principle 9275.12 (Fermat prototype for the RH proof-engineering route)**.** *The proof-engineering lesson is that the target is not a bare analogy between Fermat's Last Theorem and RH. The shared BEDC shape is the refusal of unledgered closure. In the Fermat prototype, square norms have a prime classifier and higher powers cannot impersonate primitive triangles. In the RH route, unit phases have triangle readouts and an off-fixed-half zero carries a radial scale leg. The common target is to turn that scale leg into a classifier-visible obstruction rather than a hidden coordinate.*

Chapter 9276

Right-Triangle Invariant System

chapter 9274 isolates the scale component of a centered active packet as a squared residual, and chapter 9273 records the global log-potential field that should forbid such residuals at the source. This chapter names the finite geometric system behind those rows. Its content is not an additional analytic object; it is the invariant discipline that makes observation order, orthogonal residuals, and positive energy one constraint.

Definition 9276.1 (Right-triangle observation system)**.** *Let A_m be a finite active packet with heights $h_a > 0$. For a centered readout*

$$s = \frac{1}{2} + r + it,$$

define the logarithmic observation row

$$L_m(s) := \big((-rh_a)_{a\in A_m},\, (-th_a)_{a\in A_m}\big) \in \mathbb{R}^{A_m} \oplus \mathbb{R}^{A_m}.$$

The first factor is the scale leg, and the second factor is the phase leg. The weight of the packet is

$$W_m := \sum_{a\in A_m} h_a^2.$$

A right-triangle observation system *is such a packet together with the source ledger that records A_m, the heights h_a, the orthogonal sum*

$$\mathbb{R}^{A_m} \oplus \mathbb{R}^{A_m},$$

and the assertion that hidden coordinates may refine compact phase or prime-register fibres but may not add an unrecorded $\mathbb{R}_{>0}$ scale fibre.

Theorem 9276.2 (Right-triangle energy identity)**.** *For the observation row of Definition 9276.1,*

$$\|L_m(s)\|^2 = r^2 W_m + t^2 W_m.$$

Its scale leg has energy

$$E_{m,\mathrm{scale}}(s) := \|(-rh_a)_{a\in A_m}\|^2 = r^2 W_m.$$

Proof. The two displayed factors of $\mathbb{R}^{A_m} \oplus \mathbb{R}^{A_m}$ are orthogonal. Hence

$$\|L_m(s)\|^2 = \sum_{a\in A_m} r^2 h_a^2 + \sum_{a\in A_m} t^2 h_a^2 = r^2 W_m + t^2 W_m.$$

The scale formula is the first summand. □

Corollary 9276.3 (Scale leg lower bound)**.** *For every finite active packet,*

$$\|L_m(s)\|^2 \geq E_{m,\mathrm{scale}}(s).$$

Consequently, phase notation cannot cancel a nonzero scale leg inside the squared observation energy.

Proof. By Definition 9276.2,

$$\|L_m(s)\|^2 - E_{m,\mathrm{scale}}(s) = t^2 W_m \geq 0.$$

□

Principle 9276.4 (Projection residual as coordinate creation)**.** *Let $\mathcal{H}_m$ be the Hilbert subspace generated by the displayed ledger rows before adding a further observation v. The projection decomposition*

$$v = P_{\mathcal{H}_m} v + (I - P_{\mathcal{H}_m})v$$

separates the explained part from the residual. The residual is orthogonal to $\mathcal{H}_m$ and therefore contributes the nonnegative energy

$$\|(I - P_{\mathcal{H}_m})v\|^2.$$

Thus an unexplained scale component must either appear as a new orthogonal coordinate or be ruled out by a source-side positivity row; it cannot be erased by changing the phase coordinate.

Definition 9276.5 (Finite invariant ledger)**.** *A finite invariant ledger for the right-triangle system consists of the following rows for every displayed observation family $v_1, \ldots, v_N$:*

$$G_N := (\langle v_i, v_j \rangle)_{1 \leq i,j \leq N} \succeq 0,$$

the Bessel row

$$\sum_{i=1}^{N} |\langle v, e_i \rangle|^2 \leq \|v\|^2$$

for every displayed orthonormal subfamily $e_1, \ldots, e_N$, and the Schur residual row

$$\Delta_N := a_N - b_N^* G_N^+ b_N \geq 0$$

whenever

$$G_{N+1} = \begin{pmatrix} G_N & b_N \\ b_N^* & a_N \end{pmatrix}$$

is the next displayed Gram extension with the required range row $b_N \in \mathrm{Range}(G_N)$.

Principle 9276.6 (Invariant ledger meaning)**.** *The rows of Definition 9276.5 are the finite Hilbert expression of BEDC ledger discipline. Positive semidefinite Gram matrices say that displayed correlations cannot exceed displayed energies. Bessel rows say that recorded coordinates cannot contain more energy than the observation. Schur rows say that a further observation has nonnegative conditional energy after projection to the previous ledger. In the RH route these are the common finite form of the Li–Weil time row, the Weil energy row, and the Nyman container row.*

Theorem 9276.7 (Schur row is residual energy)**.** *Assume that G_N is the Gram matrix of vectors $v_1, \dots, v_N$ in a Hilbert space, that v_{N+1} is another vector, and that*

$$a_N = \langle v_{N+1}, v_{N+1} \rangle, \qquad b_N = (\langle v_1, v_{N+1} \rangle, \dots, \langle v_N, v_{N+1} \rangle)^T.$$

Then the Schur residual Δ_N of Definition 9276.5 is the squared norm of the orthogonal residual of v_{N+1} modulo $\operatorname{span}\{v_1, \dots, v_N\}$.

Proof. Let P_N be the orthogonal projection onto $\operatorname{span}\{v_1, \dots, v_N\}$. The best-approximation theorem gives

$$\|v_{N+1} - P_N v_{N+1}\|^2 = \|v_{N+1}\|^2 - \|P_N v_{N+1}\|^2.$$

Writing the projection coefficients through the Gram system gives

$$\|P_N v_{N+1}\|^2 = b_N^* G_N^+ b_N$$

on the range row. Hence

$$\|v_{N+1} - P_N v_{N+1}\|^2 = a_N - b_N^* G_N^+ b_N = \Delta_N.$$

□

Principle 9276.8 (Mirror midpoint obstruction)**.** *Let*

$$z_+ = e^{\ell + i\theta}, \qquad z_- = e^{-\ell + i\theta}.$$

The midpoint after Li time n is

$$M_n := \frac{z_+^n + z_-^n}{2} = \cosh(n\ell) e^{in\theta}.$$

Thus

$$|M_n| = \cosh(n\ell).$$

If $\ell \neq 0$, the midpoint lies outside the unit circle. A reflected pair therefore does not remove the scale coordinate; it moves the center of the pair away from the legal phase boundary.

Principle 9276.9 (Parallelogram obstruction for reflected scale)**.** *For*

$$a = z_+^n, \qquad b = z_-^n,$$

the parallelogram identity gives

$$|a + b|^2 + |a - b|^2 = 2|a|^2 + 2|b|^2.$$

Since

$$|a| = e^{n\ell}, \qquad |b| = e^{-n\ell},$$

the right-hand side is

$$2e^{2n\ell} + 2e^{-2n\ell}.$$

For $\ell \neq 0$ this is strictly larger than 4. Mirror exchange can change the sign of the scale parameter, but the square-energy ledger records both sides as positive length.

Definition 9276.10 (Triangular invariant audit)**.** *Let* R *be a zero-relevant source row in the RH route. A* triangular invariant audit *of* R *is a triple*

$$\mathsf{Audit}(\mathsf{R}) := (\mathsf{T}_{\mathsf{R}}, \mathsf{U}_{\mathsf{R}}, \mathsf{P}_{\mathsf{R}})$$

together with a displayed defect readout. The row T_{R} *records the observation order or refinement order consumed by* R*. The row* U_{R} *records the unit, positive, or no-negative-energy side of* R*. The row* P_{R} *records the ledger side: which generators, test rows, certificates, or containers are allowed to carry the observation. The defect readout must factor through the same right-triangle scale leg as Definition 9276.1.*

Definition 9276.11 (Audited RH source family)**.** *The audited source family is the list of RH-route rows*

$$\mathcal{A}_{\triangle} := \{\mathsf{LiTimePositiveTower}, \mathsf{GlobalWeilPositivity}, \mathsf{NymanBeurlingExhaustion}, \mathsf{LeeYangBoundaryStability}, \mathsf{GoldenDomi}$$

Each member of $\mathcal{A}_{\triangle}$ *is to be read through Definition 9276.10, rather than as an isolated route.*

Principle 9276.12 (Li audit row)**.** *The audit of* $\mathsf{LiTimePositiveTower}$ *has* T*-side given by the discrete Li time* n *and the iteration*

$$z(\rho)^n.$$

Its U*-side is the nonnegative phase contribution*

$$2 - 2\cos(n\theta) \geq 0$$

when $z(\rho) = e^{i\theta}$*. Its* P*-side is the reflected zero ledger used in the Li–Weil coefficient row. A radial pair* $e^{\ell+i\theta}$ *and* $e^{-\ell+i\theta}$ *is audited by the factor*

$$\cosh(n\ell),$$

which exposes the same scale coordinate as the midpoint obstruction of Definition 9276.8.

Principle 9276.13 (Weil audit row)**.** *The audit of* $\mathsf{GlobalWeilPositivity}$ *has* T*-side given by the cofinal selection of test rows,* U*-side given by the nonnegativity of the Weil quadratic form, and* P*-side given by the source test space and regularization ledger of Definition 9265.2. A reflected off-fixed-half pair is audited by the finite value block*

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

whose negative direction is the energy-side projection of the same scale leg that appears in Definition 9276.2.

Principle 9276.14 (Nyman audit row)**.** *The audit of* $\mathsf{NymanBeurlingExhaustion}$ *has* T*-side given by the increasing Hilbert subspaces generated by the Nyman rows,* U*-side given by the projection identity*

$$\|1 - P_{N+1}1\|^2 = \|1 - P_N 1\|^2 - \|P_{N+1}1 - P_N 1\|^2,$$

and P*-side given by the displayed generator ledger* $\rho_{1/n}$*. An off-fixed-half zero supplies a continuous Mellin functional that annihilates the displayed generators while separating the constant row; this is the container-side readout of a nonzero residual coordinate.*

Principle 9276.15 (Boundary and dominance audit rows)**.** *The audit of* LeeYangBoundaryStability *has* T*-side given by finite-volume refinement,* U*-side given by unit-boundary nonzero regions, and* P*-side given by stable operations such as polarization, Asano contraction, or a representation theoretic compactness ledger. A golden or Zeckendorf ruler can be used only as a displayed refinement scale; it is not by itself the unit container.*

The audit of GoldenDominantBranch *has* T*-side given by the finite recurrence or endpoint parameter,* U*-side given by the maximal-modulus branch condition, and* P*-side given by the exact algebraic certificates that distinguish the branch. Thus a branch may be used in a zero law only after dominance and source identity are both ledgered.*

Principle 9276.16 (Capture, Nyman–Mobius, and Schur audit rows)**.** *The audit of* ActiveCapture *has* T*-side given by certificate refinement,* U*-side given by boundary stability or transport into a positive container, and* P*-side given by the source-side construction of the container before any zero island is selected.*

The audit of GoldenNymanGramExhaustion *has* T*-side given by the Fibonacci refinement order,* U*-side given by nonnegative Hilbert residual gains, and* P*-side given by the Möbius and Nyman generator ledgers that must exhaust the constant row.*

The audit of SchurResidualSquareRepresentation *has* T*-side given by appending a test direction,* U*-side given by*

$$\Delta_N \geq 0,$$

and P*-side given by a source-side square representation of* Δ_N*, not by a posterior factorization of an already positive matrix.*

Definition 9276.17 (Triangular audit coherence)**.** *The source row*

$$\mathsf{TriangularAuditCoherence}_\zeta$$

asserts that every member of $\mathcal{A}_\triangle$ *carries a triangular invariant audit and that the audits are compatible by the following two-out-of-three transport rows:*

$$\mathsf{T} + \mathsf{U} \to \mathsf{P}, \qquad \mathsf{U} + \mathsf{P} \to \mathsf{T}, \qquad \mathsf{P} + \mathsf{T} \to \mathsf{U}.$$

The arrows are not logical shortcuts between unrelated theorems. They state that if two sides of the same finite right-triangle ledger are source-side certified, then the third side must read the same scale defect and cannot introduce a different hidden coordinate.

Principle 9276.18 (Common audit invariant)**.** *The common invariant behind every audited row is the Li-coordinate decomposition*

$$|1 - z^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

The first summand is the scale residual and the second summand is the phase residual. The audit discipline requires every named RH-route row to say where this scale residual is seen in time, where it is forbidden by positivity or unit boundary, and where it is recorded in the ledger.

Definition 9276.19 (Ontic Pythagorean closure row)**.** *The source row*

$$\mathsf{OnticPythagoreanClosure}_\zeta$$

asserts that every zero-relevant certificate tower admits finite right-triangle observation systems whose scale legs are controlled by a source-side unit-positive row:

$$E_{m,\mathrm{scale}}(s_\infty) \leq B$$

for a tower bound B independent of the refinement level, while the active weights are cofinal:

$$W_m \to \infty.$$

The row also requires that any hidden fibre accepted by the tower is a compact phase or prime-register fibre, not an unrecorded real-positive scale fibre.

Theorem 9276.20 (Ontic Pythagorean closure gives fixed-half readout)**.** *Let a zero-relevant certificate tower have limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

If the tower satisfies $\mathsf{OnticPythagoreanClosure}_\zeta$ *and its finite right-triangle systems extract the scale energy*

$$E_{m,\mathrm{scale}}(s_\infty) = r^2 W_m + o(W_m),$$

then $r = 0$.

Proof. Assume $r \neq 0$. Then $r^2 > 0$. From the extraction row there is a cofinal tail on which

$$|o(W_m)| \leq \frac{r^2}{2} W_m.$$

Therefore

$$E_{m,\mathrm{scale}}(s_\infty) \geq \frac{r^2}{2} W_m$$

on that tail. Since $W_m \to \infty$, the scale energies are unbounded along the tail, contradicting the tower bound in Definition 9276.19. Hence $r = 0$. □

Principle 9276.21 (Invariant target of the RH route)**.** *The analytic burden of the route is to supply the source-side unit-positive row used in Definition 9276.19. The Stieltjes and Herglotz rows of chapter 9270, the log-potential no-scale field of chapter 9273, the Gram and Schur rows of chapter 9279, and the Nyman square tower of chapter 9281 are different finite readouts of that single positivity demand. The right-triangle system specifies what these readouts must forbid: a nonzero scale leg carried by* e^{-rh_a}*. Definition 9276.17 records the corresponding bookkeeping demand: each named route must display its time side, its unit-positive side, and its ledger side as projections of that same scale-leg obstruction.*

Chapter 9277

Source-Side Triadic Audit Criterion

chapter 9276 records the finite scale-leg invariant, while Definition 9257.20 records the certificate-tower rows P, U, and T. The present chapter states the audit rule that every RH-facing source row must satisfy before it can be consumed by the fixed-half route. A row that occupies only one side of the triangle is a local tool. A row that is used in the route must display its source ledger, unit-positive boundary, and refinement or finite-witness order.

Definition 9277.1 (Source-side triadic audit of a route row)**.** *Let $\mathcal{Q}$ be a source row in the RH route. A source-side triadic audit of $\mathcal{Q}$ is a displayed factorization*

$$\mathcal{Q} \rightsquigarrow (\mathcal{Q}_{\mathsf{P}}, \mathcal{Q}_{\mathsf{U}}, \mathcal{Q}_{\mathsf{T}})$$

where $\mathcal{Q}_{\mathsf{P}}$ is the source-ledger and legal-hidden fibre row, $\mathcal{Q}_{\mathsf{U}}$ is the unit-boundary or positive energy row, and $\mathcal{Q}_{\mathsf{T}}$ is the refinement, cofinality, or finite-witness row. The audit is closed when the three rows read the same scale defect and no side introduces an unrecorded real-positive coordinate.

Definition 9277.2 (Triadic source transport)**.** *For an audited route row $\mathcal{Q}$, triadic source transport is the requirement that the following three transport rows are source-side valid:*

$$\mathcal{Q}_{\mathsf{P}} + \mathcal{Q}_{\mathsf{T}} \to \mathcal{Q}_{\mathsf{U}}, \qquad \mathcal{Q}_{\mathsf{P}} + \mathcal{Q}_{\mathsf{U}} \to \mathcal{Q}_{\mathsf{T}}, \qquad \mathcal{Q}_{\mathsf{U}} + \mathcal{Q}_{\mathsf{T}} \to \mathcal{Q}_{\mathsf{P}}.$$

The first row says that a legal ledger and an internal time readout force the absence of an independent radial gauge. The second says that a legal ledger and a unit boundary leave only the phase-time parameter as a free readout. The third says that a unit boundary and a refinement law select the zero-relevant modes that must be ledgered.

Definition 9277.3 (Triadic no-scale packet)**.** *Let an active packet A_m carry heights $h_a > 0$ and centered channel*

$$U_a(s) = e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a}.$$

The packet is triadic no-scale when its P row accepts compact phase and prime-register fibres only, its U row supplies a source-side bound for the scale energy, and its T row supplies cofinal active-weight growth or a finite negative witness whenever $r \neq 0$. In symbols the scale row is

$$d_a(s) := \log |U_a(s)| = -rh_a, \qquad E_m^{\mathrm{scale}}(s) = \sum_{a \in A_m} d_a(s)^2 = r^2 W_m,$$

where

$$W_m := \sum_{a \in A_m} h_a^2.$$

Principle 9277.4 (Stable rows already fit the audit)**.** *The BEDC history generator supplies the* T *side: finite records extend by append-only constructor rows. The phase-channel solenoid of Definition 9286.1 supplies the* P *side: hidden state may live in compact prime-register fibres, not in an external real-positive scale line. Weak eta capture of Definition 9255.7 supplies local analytic capture and part of the time side, but it does not supply the unit side. The Fibonacci path and Lee–Yang lift rows of Definition 9260.1 and Definition 9260.2 supply a finite internal ruler and a unit-boundary seed, but not by themselves a prime-compatible zeta container. The quartic endpoint model supplies the methodological load: a branch must be shown dominant by source certificates before it can carry a zero law.*

Definition 9277.5 (Audited golden bridge rows)**.** *The golden bridge targets are audited as follows.*

The row GoldenStableRoucheHomotopy *has* P *side given by the eta, Zeckendorf, and bridge ledgers,* U *side given by Lee–Yang or Herglotz stability along the bridge, and* T *side given by boundary non-vanishing and winding preservation.*

The row PrimeZeckendorfLift *has* P *side given by the transport from additive Zeckendorf data to multiplicative prime-register data,* U *side given by preservation of unit boundary under that transport, and* T *side given by cofinal refinement through the prime-compatible tower.*

The row PrimeCompatibleGoldenTower *has* P *side given by prime-register alignment at every level,* U *side given by no-scale boundary stability, and* T *side given by unbounded active-weight growth or finite-witness extraction.*

Definition 9277.6 (Audited positivity rows)**.** *The positive spectral targets are audited as follows.*

The row BalancedGoldenConstruction *has* P *side given by internally determined normalization data,* U *side given by preservation of the unit-boundary zero law after balancing, and* T *side given by compatibility of the normalization under refinement.*

The row BalancedGoldenLiCNDClosure *has* P *side given by the finite container and zero-ledger provenance,* U *side given by conditional negative definiteness of the Li kernel, and* T *side given by closure of that kernel row under the limiting process.*

The row XiStieltjesLogDerivative *has* P *side given by a positive measure constructed from source data rather than from a posterior fixed-half divisor,* U *side given by Stieltjes and Herglotz positivity, and* T *side given by a half-plane representation rather than a pointwise search.*

The row GlobalWeilPositivity *has* P *side given by the legal test space and explicit-formula ledger,* U *side given by nonnegativity of the Weil quadratic form, and* T *side given by cofinal finite test packets that detect any negative direction.*

Definition 9277.7 (Audited witness-reduction rows)**.** *The finite-witness and closure targets are audited as follows.*

The row WeilPositivityReduction *has* P *side given by a legal test packet,* U *side given by violation of a positive quadratic row, and* T *side given by finite detection of the negative witness.*

The row ObservationPythagoreanClosureForZeta *has* P *side given by heights from prime or solenoid ledgers,* U *side given by control of the scale component through unit positivity, and* T *side given by cofinal growth*

$$W_m \to \infty$$

or by finite witness extraction.

Theorem 9277.8 (Triadic no-scale criterion)**.** *Let every zero-relevant accepted certificate tower be represented by triadic no-scale packets. Assume that each such packet satisfies triadic source transport, has defect extraction*

$$E_m^{\mathrm{scale}}(s_\infty) = r^2 W_m + o(W_m),$$

has either $W_m \to \infty$ or a finite negative witness when $r \neq 0$, and carries a source-side unit-positive row excluding unbounded scale energy and negative witnesses. Then every represented readout

$$s_\infty = \frac{1}{2} + r + it$$

has $r = 0$.

Proof. Assume $r \neq 0$. If the time row supplies cofinal active-weight growth, then the defect extraction gives a cofinal tail on which

$$E_m^{\text{scale}}(s_\infty) \geq \frac{r^2}{2} W_m.$$

The energies are unbounded along that tail, contradicting the source-side unit-positive row. If instead the time row supplies a finite negative witness, that witness contradicts the same unit-positive row in its finite test form. In both cases the assumption $r \neq 0$ is impossible. Hence $r = 0$. □

Principle 9277.9 (Compressed source target)**.** *The compressed source target is to realize one positive-spectrum row whose triadic audit is source-side. The Stieltjes log-derivative target of Definition 9270.13 is the most compact form of this requirement: the measure must be generated before reading a fixed-half divisor, positivity supplies the unit side, and the representation supplies a half-plane time row. When such a source row is transported through the prime and certificate ledgers, the right-triangle scale coordinate has no legal carrier.*

Chapter 9278

Radial Scale Projection System

chapter 9274 identifies the analytic triangle as one scale defect, and chapter 9276 records its finite Hilbert form. The present chapter isolates the scalar readout shared by the Li, Weil, and Nyman projections. The scalar is the radial coordinate of the Li map; every analytic line in the route reads the same coordinate through a different finite or Hilbert witness.

Definition 9278.1 (Li radial scale coordinate)**.** *For a nontrivial completed zeta zero row ρ, let*

$$z(\rho) := 1 - \frac{1}{\rho}$$

as in Definition 9266.1. The Li radial scale coordinate is

$$\ell(\rho) := \log |z(\rho)|.$$

When $z(\rho) = e^{\ell + i\theta}$, the coordinate ℓ is the scale leg and θ is the phase angle.

Theorem 9278.2 (Radial scale reads the fixed-half surface)**.** *For a nontrivial completed zero row ρ,*

$$\ell(\rho) = 0$$

if and only if the readout of ρ lies on the fixed-half surface.

Proof. By definition, $\ell(\rho) = 0$ exactly when $|z(\rho)| = 1$. By Definition 9266.2, this is equivalent to the fixed-half readout. □

Definition 9278.3 (Critical half axis row)**.** *The critical half axis row is the recovered-carrier predicate*

$$\mathcal{C}_{1/2} := \{s : \mathrm{Re}(s) = 1/2\}$$

read inside the completed zeta zero interface. Its mirror source is the involution of Definition 9320.1,

$$\iota(s) = 1 - \overline{s},$$

its unit readout is the Li coordinate

$$z(s) = 1 - \frac{1}{s},$$

and its ledger readout is the radial scale

$$\ell(s) = \log |z(s)|.$$

Theorem 9278.4 (Critical half axis is the mirror fixed locus)**.** *For a completed zero coordinate* $s = \sigma + i\gamma$*,*

$$\iota(s) = s$$

if and only if

$$\sigma = \frac{1}{2}.$$

Proof. By Definition 9320.1,

$$\iota(s) = 1 - \overline{s} = 1 - \sigma + i\gamma.$$

The equality $\iota(s) = s$ is therefore equivalent to

$$1 - \sigma = \sigma,$$

which is equivalent to $\sigma = 1/2$. □

Definition 9278.5 (Critical half axis triple)**.** *The critical half axis carries the ontological triple*

$$\mathrm{Ont}(\mathcal{C}_{1/2}) = (\mathsf{T}_{\mathcal{C}}, \mathsf{U}_{\mathcal{C}}, \mathsf{L}_{\mathcal{C}}).$$

The time row $\mathsf{T}_{\mathcal{C}}$ *is the fixed locus of the zeta mirror involution. The unit row* $\mathsf{U}_{\mathcal{C}}$ *is the unit Li readout*

$$s \in \mathcal{C}_{1/2} \Longleftrightarrow |z(s)| = 1.$$

The ledger row $\mathsf{L}_{\mathcal{C}}$ *is the zero radial account*

$$s \in \mathcal{C}_{1/2} \Longleftrightarrow \ell(s) = 0.$$

Thus the critical half axis is not an unlicensed host line: it is the common fixed, unit, and zero-ledger row consumed by the closed zero packet route.

Theorem 9278.6 (Mirror reflection reverses the radial ledger)**.** *If* $z(s) = e^{\ell + i\theta}$ *and* $\iota(s)$ *is the mirror coordinate, then*

$$z(\iota(s)) = e^{-\ell + i\theta}.$$

Consequently the critical half axis is exactly the zero ledger locus for the mirror radial pair.

Proof. The identity is Definition 9320.2. The zero ledger locus is $\ell = 0$, which is equivalent to fixed-half readout by Definition 9278.2. □

Theorem 9278.7 (Closed radial packets lie on the critical half axis)**.** *Let a licensed zeta zero packet carry*

ZetaZeroObjectsHaveOntTriple, GlobalOffLineZeroToFiniteNPWindow, LedgerClosureRefusal.

If the packet is closed, then its Li radial scale satisfies

$$\ell(\rho) = 0,$$

and hence its readout lies on $\mathcal{C}_{1/2}$*.*

Proof. If $\ell(\rho) \neq 0$, then Definition 9278.2 says that ρ is off the fixed-half surface. The licensed triple route then gives the closed-status refusal of Definition 9310.9. This contradicts the closed-packet hypothesis. Therefore $\ell(\rho) = 0$, and Definition 9278.5 identifies the readout with the critical half axis. □

Theorem 9278.8 (Li radial Pythagorean identity)**.** *Let*

$$z = e^{\ell + i\theta}.$$

For every $n \geq 1$,

$$|1 - z^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

Proof. This is Definition 9274.19, rewritten with the notation of Definition 9278.1. □

Definition 9278.9 (Radial projection triple)**.** *The radial projection triple attached to a completed zero ledger is*

$$\mathsf{RadialProjection}_\zeta := (\mathsf{T}_\ell, \mathsf{W}_\ell, \mathsf{N}_\ell).$$

The time projection T_ℓ *is the Li-time readout of* $z(\rho)^n$*. The energy projection* W_ℓ *is the Weil-quadratic readout of reflected zero pairs. The container projection* N_ℓ *is the Nyman–Beurling readout of the Mellin separator attached to a zero in the right open half of the critical strip.*

Principle 9278.10 (Time projection of a nonzero radial scale)**.** *If a reflected zero pair has Li coordinates*

$$z_+ = e^{\ell + i\theta}, \qquad z_- = e^{-\ell + i\theta},$$

then Li time reads the radial coordinate through the factor

$$\cosh(n\ell).$$

For $\ell \neq 0$*, this factor is larger than* 1 *for* $n \geq 1$*. The finite Li–Weil witness row of Definition 9265.22 is the time-side certificate that such radial amplification cannot coexist with the global Li positivity row.*

Principle 9278.11 (Energy projection of a nonzero radial scale)**.** *A reflected off-fixed-half pair produces a Weil value block with a negative direction, as in Definition 9274.21. The sign of* ℓ *may be reversed by reflection, but the existence of a two-point reflected orbit with* $|\ell| > 0$ *is what gives the energy-side negative direction. Global Weil positivity forbids that projection when the test row and tail ledgers are source-side controlled.*

Principle 9278.12 (Container projection of a nonzero radial scale)**.** *If the reflected orbit has a representative*

$$s_0 = \sigma + i\tau, \qquad \sigma > \frac{1}{2},$$

then Definition 9281.8 supplies the continuous Mellin functional

$$L_{s_0}(f) = \int_0^1 f(x) x^{s_0 - 1}\, dx.$$

It annihilates the displayed Nyman generators and does not annihilate **1***. Thus the same radial scale coordinate that Li time reads as amplification and Weil energy reads as a negative direction is read by the Nyman container as a residual separator.*

Definition 9278.13 (Zeta radial triangular closure)**.** *The source row*

$$\mathsf{ZetaRadialTriangularClosure}$$

asserts that every accepted nontrivial completed zero ledger carries the radial projection triple of Definition 9278.9, and that the three projections are governed by one scalar

$$\ell(\rho) = \log\left|1 - \frac{1}{\rho}\right|.$$

The row further asserts that a nonzero value of this scalar must be detected by at least one of the three finite or Hilbert witnesses: Li-time negativity, Weil negative direction, or Nyman residual separation.

Theorem 9278.14 (Radial triangular closure gives fixed-half readout)**.** *Assume* ZetaRadialTriangularClosure *and assume the three projection rows*

GlobalLiWeilPositivity, GlobalWeilPositivity, NymanBeurlingExhaustion

are available with their usual reflection consumption rows. Then every accepted nontrivial completed zero has fixed-half readout.

Proof. Let ρ be an accepted nontrivial completed zero. If $\ell(\rho) \neq 0$, the closure row forces a detection in one of the three projections. A Li-time negative witness contradicts GlobalLiWeilPositivity. A Weil negative direction contradicts GlobalWeilPositivity. A Nyman residual separator contradicts NymanBeurlingExhaustion by Definition 9281.8. Hence $\ell(\rho) = 0$. By Definition 9278.2, ρ has fixed-half readout. □

Principle 9278.15 (Single radial burden)**.** *The Li, Weil, and Nyman tasks should be read as three audits of one radial coordinate, not as unrelated proof searches. Li time asks whether powers of $z(\rho)$ expose $\ell(\rho)$. Weil energy asks whether a test row can extract a negative direction from the reflected orbit. Nyman exhaustion asks whether the legal generator container leaves a residual Mellin functional. The source-side burden is to show that the BEDC triadic carrier never admits a nonzero radial coordinate without one of these witnesses.*

Definition 9278.16 (Cross-projection radial refutation packet)**.** *A cross-projection radial refutation packet for a completed zero ledger is a source-side row that starts from a displayed assumption*

$$\ell(\rho) \neq 0$$

and exports this same scalar to the three projection ledgers

$$\mathsf{T}_\ell, \qquad \mathsf{W}_\ell, \qquad \mathsf{N}_\ell$$

of Definition 9278.9. The packet is accepted only when the time ledger reads the reflected Li pair, the energy ledger reads the corresponding Weil negative direction, and the container ledger reads the Mellin separator in the open half selected by reflection. Thus the packet does not certify a contradiction inside a single projection; it certifies that a nonzero radial coordinate has the same readback in the other two projections as well.

Definition 9278.17 (Triangular burden of a radial source row)**.** *Let $\mathcal{Q}$ be a RH-facing source row whose conclusion is consumed by the fixed-half route. Its triangular radial burden is the requirement that any use of $\mathcal{Q}$ against a zero ledger with*

$$\ell(\rho) \neq 0$$

comes with a cross-projection radial refutation packet. A row that only records positivity, separation, or exhaustion in its native projection is therefore a local analytic row. It becomes a route row only after the same radial scalar has been transported to the remaining projection ledgers.

Theorem 9278.18 (Two-sided projection audit closes the third side)**.** *Assume a completed zero ledger carries a cross-projection radial refutation packet. Suppose two of the projection rows*

$$\mathsf{GlobalLiWeilPositivity}, \qquad \mathsf{GlobalWeilPositivity}, \qquad \mathsf{NymanBeurlingExhaustion}$$

are source-side closed, and suppose the remaining projection is linked to them by the triadic source transport of Definition 9277.2. Then the zero ledger cannot carry

$$\ell(\rho) \neq 0.$$

Proof. Assume $\ell(\rho) \neq 0$. The cross-projection packet exports this scalar to all three ledgers. If the time ledger supplies a finite negative Li witness, it contradicts GlobalLiWeilPositivity. If the energy ledger supplies a Weil negative direction, it contradicts GlobalWeilPositivity. If the container ledger supplies a Mellin separator, it contradicts NymanBeurlingExhaustion by Definition 9281.8. The two closed sides and the triadic source transport ensure that the remaining side cannot hide the same scalar as an unledgered real-positive coordinate. Hence the assumption $\ell(\rho) \neq 0$ is impossible. □

Principle 9278.19 (No single-projection consumption)**.** *The fixed-half route may consume a Li, Weil, or Nyman target only after its triangular radial burden is displayed. A Li square row must explain where the corresponding radial scale would appear as Weil energy and as Nyman residual separation. A Weil factorization row must explain where the same radial scale would appear in Li time and in the Nyman container. A Nyman exhaustion row must explain where its residual separator would appear in Li time and Weil energy. This is the source-side form of the rule that every off-fixed-half assumption is tested against the other two ledgers, not only against the ledger in which it was first detected.*

Chapter 9279

Golden Weil Gram Induction

chapter 9265 records the global Weil-positive tower as the non-local target of the no-scale route. The present chapter records a finite Gram-matrix form of that target. The aim is to replace an open-ended search through zero certificates by a source-side sequence of finite semidefinite tests.

Definition 9279.1 (Golden Weil test ruler)**.** *Let* $\varphi = (1+\sqrt{5})/2$ *and let*

$$\delta_m := \varphi^{-m}.$$

Choose a fixed compactly supported smooth bump ψ *on the logarithmic line. A* golden Weil test ruler *is a sequence*

$$\mathcal{B}_\varphi = (\psi_i)_{i\geq 1}$$

obtained from translated and rescaled bumps

$$\psi_{m,k}(u) := \psi\left(\frac{u-k\delta_m}{\delta_m}\right)$$

by a Zeckendorf ledger ordering. The ruler rows require that the source of each scale is recorded by the Fibonacci ledger, that

$$\operatorname{span}\{\psi_i : i \geq 1\}$$

is dense in the test space of Definition 9265.1, and that adding a later ψ_i *refines the observation ledger without deleting earlier rows.*

Definition 9279.2 (Golden Weil Gram matrix)**.** *For the finite test space*

$$\mathcal{A}_N := \operatorname{span}\{\psi_1, \ldots, \psi_N\},$$

define the golden Weil Gram matrix

$$G_N = (g_{ij})_{1\leq i,j\leq N}$$

by the sesquilinear Weil pairing

$$g_{ij} := \mathcal{W}(\psi_i, \psi_j),$$

where $\mathcal{W}(f,g)$ *denotes the polarization of the quadratic row of Definition 9265.2. Thus, for*

$$f_c = \sum_{i=1}^{N} c_i\psi_i,$$

one has

$$c^* G_N c = \mathcal{W}(f_c).$$

Theorem 9279.3 (Golden Gram reduction)**.** *Assume the golden Weil test ruler is dense and that the Weil quadratic row is continuous for the chosen test topology. If*

$$G_N \succeq 0 \qquad \text{for every } N,$$

then GlobalWeilPositivity *holds. Conversely,* GlobalWeilPositivity *implies* $G_N \succeq 0$ *for every* N*.*

Proof. Assume first that every G_N is positive semidefinite. Let f be a test row. By density, choose $f_N \in \mathcal{A}_N$ with $f_N \to f$. Since $G_N \succeq 0$,

$$\mathcal{W}(f_N) \geq 0.$$

Continuity gives

$$\mathcal{W}(f) = \lim_N \mathcal{W}(f_N) \geq 0.$$

Thus GlobalWeilPositivity holds.

Conversely, if GlobalWeilPositivity holds, then for every $c \in \mathbb{C}^N$,

$$c^* G_N c = \mathcal{W}\left(\sum_{i=1}^N c_i \psi_i\right) \geq 0.$$

Hence $G_N \succeq 0$. □

Principle 9279.4 (Finite diagonal witness)**.** *If the global Weil form has a negative direction, then the golden ruler detects it at finite resolution. Indeed, choose* f *with*

$$\mathcal{W}(f) < 0.$$

By density and continuity, some $f_N \in \mathcal{A}_N$ *still satisfies*

$$\mathcal{W}(f_N) < 0.$$

Writing $f_N = \sum_i c_i \psi_i$ *gives*

$$c^* G_N c < 0.$$

Thus a diagonal negative probe is not an infinitely postponed anomaly; it appears as a finite negative Gram direction.

Definition 9279.5 (Schur residual of the golden Weil tower)**.** *Write the next Gram matrix as*

$$G_{N+1} = \begin{pmatrix} G_N & b_N \\ b_N^* & a_N \end{pmatrix},$$

where

$$a_N := \mathcal{W}(\psi_{N+1}, \psi_{N+1})$$

and

$$b_N := \left(\mathcal{W}(\psi_1, \psi_{N+1}), \ldots, \mathcal{W}(\psi_N, \psi_{N+1})\right)^T.$$

The Schur residual *is*

$$\Delta_N := a_N - b_N^* G_N^+ b_N,$$

with G_N^+ *the Moore–Penrose inverse. The range row*

$$b_N \in \mathrm{Range}(G_N)$$

is part of the semidefinite Schur ledger.

Definition 9279.6 (Diagonal-safe Weil induction)**.** *The source row*

$$\mathsf{DiagonalSafeWeilInduction}$$

asserts that

$$G_1 \succeq 0,$$

and for every N the range row

$$b_N \in \mathrm{Range}(G_N)$$

and the conditional-energy row

$$\Delta_N \geq 0$$

hold.

Theorem 9279.7 (Schur induction gives finite Gram positivity)**.** *If* DiagonalSafeWeilInduction *holds, then*

$$G_N \succeq 0 \qquad \textit{for every } N.$$

Proof. Proceed by induction on N. The base row is $G_1 \succeq 0$. For the step, assume $G_N \succeq 0$. The semidefinite Schur complement criterion says that

$$G_{N+1} = \begin{pmatrix} G_N & b_N \\ b_N^* & a_N \end{pmatrix} \succeq 0$$

exactly when b_N lies in the range of G_N and

$$a_N - b_N^* G_N^+ b_N \geq 0.$$

These are precisely the range and conditional-energy rows of Definition 9279.6. □

Corollary 9279.8 (Diagonal-safe Weil induction gives global positivity)**.** *Assume the density and continuity rows of Definition 9279.3. If* DiagonalSafeWeilInduction *holds, then* GlobalWeilPositivity *holds.*

Proof. By Definition 9279.7, every G_N is positive semidefinite. Apply Definition 9279.3. □

Definition 9279.9 (Orthogonal residual of the added test row)**.** *When the range row holds, the residual of the added test row is the formal class*

$$\psi_{N+1}^{\perp} := \psi_{N+1} - \Pi_N \psi_{N+1},$$

where Π_N is the Weil-Gram projection onto $\mathcal{A}_N$. The Schur residual is the residual energy row

$$\Delta_N = \mathcal{W}(\psi_{N+1}^{\perp}).$$

Definition 9279.10 (Golden Weil Schur residual positivity)**.** *The source row*

$$\mathsf{GoldenWeilSchurResidualPositivity}$$

asserts that every residual row of Definition 9279.9 has nonnegative Weil energy:

$$\mathcal{W}(\psi_{N+1}^{\perp}) \geq 0.$$

It is the finite conditional-information form of the no-demon obligation.

Definition 9279.11 (Schur residual square representation)**.** *The source row*

SchurResidualSquareRepresentation

asserts that for every N there are a Hilbert space $\mathcal{H}_N$ and a source-side linear readout T_N such that

$$\Delta_N = \|T_N \psi_{N+1}^{\perp}\|_{\mathcal{H}_N}^2.$$

Theorem 9279.12 (Square representation gives diagonal-safe induction)**.** *If $G_1 \succeq 0$, the range rows hold, and* SchurResidualSquareRepresentation *holds, then* DiagonalSafeWeilInduction *holds.*

Proof. The square representation gives

$$\Delta_N = \|T_N \psi_{N+1}^{\perp}\|_{\mathcal{H}_N}^2 \geq 0$$

for every N. Together with the assumed base and range rows, this is exactly DiagonalSafeWeilInduction. □

Principle 9279.13 (Finite induction target)**.** *The finite Gram form converts the global positivity route into a sequence of Schur residuals. The active source obligation is not a universal decider for all future tests; it is the structural representation*

SchurResidualSquareRepresentation.

If each added golden resolution direction has residual Weil energy presented as a norm square, the Gram tower is positive by finite induction and the global Weil positivity row follows by density.

Chapter 9280

Diagonal Schur Square Target

chapter 9279 reduces the global Weil-positive route to finite Gram positivity and then to Schur residuals. This chapter records the diagonal obstruction behind that reduction and isolates the source-side square target: each conditional Weil energy row must be presented as a norm square before any posterior matrix factorization is invoked.

Definition 9280.1 (Finite reflected zero ledger)**.** *Let J denote the completed-zeta reflection*

$$J(\rho) := 1 - \overline{\rho}.$$

A finite zero ledger $\mathcal{Z}_T$ is reflected *when it is closed under J, with the displayed multiplicities preserved. On the finite value space*

$$V_T := \{v = (v_\rho)_{\rho \in \mathcal{Z}_T}\},$$

define the reflected Weil block

$$Q_T(v) := \sum_{\rho \in \mathcal{Z}_T} v_\rho \, \overline{v_{J\rho}}.$$

Theorem 9280.2 (Finite reflected Weil signature)**.** *Assume $\mathcal{Z}_T$ is a reflected finite zero ledger. A fixed point of J contributes a positive square to Q_T. A two-point orbit $\{\rho, J\rho\}$ with $\rho \neq J\rho$ contributes one positive and one negative direction. Consequently, Q_T is positive semidefinite exactly when the finite ledger has no two-point reflected orbit.*

Proof. If $J\rho = \rho$, then the orbit contribution is

$$v_\rho \overline{v_\rho} = |v_\rho|^2.$$

If $\rho \neq J\rho$, write the two coordinates as (x, y). The orbit contribution is

$$x\overline{y} + y\overline{x} = 2\,\mathrm{Re}(x\overline{y}).$$

In the basis (x, y) the Hermitian matrix is

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

with eigenvalues 1 and -1. Summing over the finite J-orbits gives the signature statement and the stated criterion for semidefiniteness. $\square$

Corollary 9280.3 (Finite mirror negative value row)**.** *If $\mathcal{Z}_T$ contains a two-point reflected orbit*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma, \qquad r \neq 0,$$

then there is $v \in V_T$ with

$$Q_T(v) < 0.$$

Proof. Set $v_{\rho_+} = 1$, $v_{\rho_-} = -1$, and set every other displayed coordinate to 0. Then the orbit contribution is

$$1 \cdot \overline{-1} + (-1) \cdot \overline{1} = -2.$$

All other displayed contributions vanish. □

Definition 9280.4 (Finite Mellin interpolation row)**.** *The test space $\mathcal{A}$ has the finite Mellin interpolation row when for every finite set of distinct complex points*

$$s_1, \ldots, s_M$$

and target values

$$a_1, \ldots, a_M$$

there is $f \in \mathcal{A}$ with

$$\widehat{f}(s_j) = a_j \qquad (1 \leq j \leq M).$$

In logarithmic coordinates this asks for a compactly supported smooth row ϕ with

$$\int_{-\infty}^{\infty} \phi(u) e^{s_j u}\, du = a_j.$$

Lemma 9280.5 (Translate packets give finite Mellin interpolation)**.** *Assume the logarithmic test space contains finite linear combinations of translates of a compactly supported smooth bump ψ whose Laplace transform*

$$\Psi(s) := \int_{-\infty}^{\infty} \psi(u) e^{su}\, du$$

is nonzero at $s_1, \ldots, s_M$. Then the finite Mellin interpolation row holds for these points.

Proof. Use test rows of the form

$$\phi(u) = \sum_{k=1}^{M} c_k \psi(u - T_k).$$

Their readout at s_j is

$$\int \phi(u) e^{s_j u}\, du = \Psi(s_j) \sum_{k=1}^{M} c_k e^{s_j T_k}.$$

The functions $T \mapsto e^{s_j T}$ are linearly independent for distinct s_j. Therefore one can choose $T_1, \ldots, T_M$ so that the evaluation matrix

$$\left(e^{s_j T_k}\right)_{j,k}$$

is invertible; otherwise the determinant would vanish for all choices of sample points and the displayed exponentials would be linearly dependent. The resulting finite linear system determines $c_1, \ldots, c_M$. □

Definition 9280.6 (Tail-controlled diagonal interpolation)**.** *The finite Mellin interpolation row is tail-controlled for a reflected finite zero ledger when the interpolating row can be chosen so that the regularized contribution of zeros outside the ledger is bounded by a displayed error constant. For the mirror value row of Definition 9280.3, the required bound is strictly smaller than the magnitude of the finite negative block.*

Theorem 9280.7 (Tail-controlled interpolation gives a negative Weil probe)**.** *Assume a reflected finite zero ledger contains a two-point reflected orbit and that the finite Mellin interpolation row is tail-controlled for the corresponding mirror value row. Then there exists $f \in \mathcal{A}$ with*

$$\mathcal{W}(f) < 0.$$

Proof. By Definition 9280.3, prescribe values on the finite ledger whose reflected block contribution is negative. By finite Mellin interpolation, choose a test row realizing these values. The tail-control row makes the regularized contribution from the complement smaller than the finite negative margin. Therefore the full Weil quadratic row is negative. □

Definition 9280.8 (Explicit formula Schur square)**.** *The source row*

ExplicitFormulaSchurSquare

asserts that every Schur residual of Definition 9279.9 is presented directly from the explicit-formula source rows as a norm square. In symbols, there are Hilbert spaces $\mathcal{H}_N$ and source-side linear readouts T_N such that

$$\mathcal{W}(\psi_{N+1}^{\perp}) = \|T_N \psi_{N+1}^{\perp}\|_{\mathcal{H}_N}^2.$$

The readout T_N must be constructed from the archimedean, prime, normalization, solenoid, Lee–Yang, Jensen, or trace-formula rows. It may not be obtained by first assuming the positivity of G_{N+1} and then factoring that matrix.

Theorem 9280.9 (Explicit formula Schur square gives square representation)**.** *If* ExplicitFormulaSchurSquare *holds, then*

SchurResidualSquareRepresentation

holds.

Proof. The row ExplicitFormulaSchurSquare supplies exactly the Hilbert spaces and source-side readouts required in Definition 9279.11, with

$$\Delta_N = \mathcal{W}(\psi_{N+1}^{\perp}) = \|T_N \psi_{N+1}^{\perp}\|_{\mathcal{H}_N}^2.$$

□

Corollary 9280.10 (Explicit formula Schur square gives global positivity)**.** *Assume $G_1 \succeq 0$, the Schur range rows, and the density and continuity rows of Definition 9279.3. If*

ExplicitFormulaSchurSquare

holds, then GlobalWeilPositivity *holds.*

Proof. By Definition 9280.9, the Schur residual square representation holds. Then Definition 9279.12 gives DiagonalSafeWeilInduction, and Definition 9279.8 gives GlobalWeilPositivity. □

Definition 9280.11 (Weil–Kolmogorov factorization)**.** *The source row*

$$\mathsf{WeilKolmogorovFactorization}$$

asserts the existence of a Hilbert space $\mathcal{H}$ and a linear source readout

$$\Theta : \mathcal{A} \to \mathcal{H}$$

such that the polarized Weil form satisfies

$$\mathcal{W}(f, g) = \langle \Theta f, \Theta g \rangle_{\mathcal{H}} \qquad (f, g \in \mathcal{A}).$$

The construction of Θ is source-side: it is not the operation of placing zero ordinates into a spectrum after fixed-half readout has already been obtained.

Theorem 9280.12 (Weil–Kolmogorov factorization gives Schur squares)**.** *If* WeilKolmogorovFactorization *holds, then*

$$\mathsf{SchurResidualSquareRepresentation}$$

holds for the golden Weil Gram tower.

Proof. Let Θ be the factorization map. The Gram matrix G_N is the ordinary Hilbert Gram matrix of

$$\Theta\psi_1, \ldots, \Theta\psi_N.$$

The residual row $\psi_{N+1}^{\perp}$ maps to the Hilbert residual of $\Theta\psi_{N+1}$ modulo the span of the earlier vectors. Therefore

$$\Delta_N = \mathcal{W}(\psi_{N+1}^{\perp}) = \|\Theta\psi_{N+1}^{\perp}\|_{\mathcal{H}}^2.$$

Taking T_N to be the restriction of Θ to the residual row gives the required square representation. □

Corollary 9280.13 (Weil–Kolmogorov factorization gives fixed-half readout)**.** *Assume the base, range, density, continuity, and Weil criterion consumption rows used above. If*

$$\mathsf{WeilKolmogorovFactorization}$$

holds, then the completed zeta zero ledger has fixed-half readout.

Proof. Definition 9280.12 gives SchurResidualSquareRepresentation. The square representation gives DiagonalSafeWeilInduction, hence GlobalWeilPositivity. The Weil criterion consumption row then returns fixed-half readout. □

Principle 9280.14 (Minimal diagonal-safe square target)**.** *The finite Gram route reduces the non-local obstruction to conditional energy rows:*

$$\Delta_N = a_N - b_N^* G_N^+ b_N.$$

The non-circular target is not a posterior factorization of a positive matrix. It is a source construction of the readouts T_N, or more globally of a Weil–Kolmogorov factorization. In named form the target chain is

WeilKolmogorovFactorization $\Longrightarrow$ SchurResidualSquareRepresentation $\Longrightarrow$ GlobalWeilPositivity $\Longrightarrow$ *fixed-half zero rea*

Chapter 9281

Golden Nyman Square Tower

chapter 9280 asks for a source-side square presentation of the conditional Weil energy rows. The Nyman–Beurling Hilbert container gives a more constructive square tower: finite positivity is supplied by the ambient space $L^2(0,1)$, while the remaining obligation is cofinal exhaustion of the constant row.

Definition 9281.1 (Nyman–Beurling generator)**.** *For $0 < \theta \le 1$ and $0 < x < 1$, define*

$$\rho_\theta(x) := \left\{\frac{\theta}{x}\right\} - \theta\left\{\frac{1}{x}\right\},$$

where $\{y\}$ denotes the fractional part of y. Let

$$\mathcal{H}_{\mathrm{NB}} := L^2(0,1), \qquad \mathbf{1}(x) := 1.$$

Definition 9281.2 (Finite Nyman Hilbert container)**.** *For $N \ge 1$, let*

$$V_N := \operatorname{span}\{\rho_1, \rho_{1/2}, \dots, \rho_{1/N}\} \subset \mathcal{H}_{\mathrm{NB}}.$$

Let P_N be the orthogonal projection onto V_N and define

$$d_N := \operatorname{dist}_{L^2}(\mathbf{1}, V_N) = \|\mathbf{1} - P_N\mathbf{1}\|_2.$$

Theorem 9281.3 (Finite Nyman Gram positivity)**.** *Let*

$$B_N := \Big(\langle \rho_{1/i}, \rho_{1/j}\rangle_{L^2(0,1)}\Big)_{1\le i,j\le N}.$$

Then $B_N \succeq 0$ for every N.

Proof. For $c = (c_1, \dots, c_N) \in \mathbb{C}^N$,

$$c^* B_N c = \left\|\sum_{j=1}^N c_j \rho_{1/j}\right\|_2^2 \ge 0.$$

Thus B_N is positive semidefinite. □

Definition 9281.4 (Nyman square drop)**.** *The Nth Nyman square drop is*

$$\Delta_N^{\mathrm{NB}} := d_N^2 - d_{N+1}^2.$$

Theorem 9281.5 (Nyman drops are Hilbert squares)**.** *For the nested spaces $V_N \subseteq V_{N+1}$,*

$$\Delta_N^{\mathrm{NB}} = \|P_{N+1}\mathbf{1} - P_N\mathbf{1}\|_2^2 \geq 0.$$

Proof. Since $V_N \subseteq V_{N+1}$, the projection $P_{N+1}\mathbf{1} - P_N\mathbf{1}$ is orthogonal to $\mathbf{1} - P_{N+1}\mathbf{1}$. Therefore

$$\mathbf{1} - P_N\mathbf{1} = (\mathbf{1} - P_{N+1}\mathbf{1}) + (P_{N+1}\mathbf{1} - P_N\mathbf{1})$$

is an orthogonal decomposition. Pythagoras gives

$$d_N^2 = d_{N+1}^2 + \|P_{N+1}\mathbf{1} - P_N\mathbf{1}\|_2^2,$$

which is the claimed square identity. □

Definition 9281.6 (Nyman exhaustion row)**.** *The source row*

$$\mathsf{NymanBeurlingExhaustion}$$

asserts that

$$\lim_{N\to\infty} d_N = 0.$$

Equivalently,

$$\mathbf{1} \in \overline{\operatorname{span}\{\rho_{1/n} : n \geq 1\}}^{L^2(0,1)}.$$

Definition 9281.7 (Nyman Mellin bridge row)**.** *The Nyman Mellin bridge row is the classical identity*

$$\int_0^1 \rho_\theta(x)x^{s-1}\,dx = \frac{(\theta - \theta^s)\zeta(s)}{s}, \qquad 0 < \mathrm{Re}(s) < 1.$$

It is consumed only as an analytic bridge between the Hilbert container and the zeta zero ledger.

Lemma 9281.8 (Off-half zero gives a Nyman separator)**.** *Assume the Nyman Mellin bridge row. If s_0 is a nontrivial zeta zero with $\mathrm{Re}(s_0) > 1/2$, then the functional*

$$L_{s_0}(f) := \int_0^1 f(x)x^{s_0-1}\,dx$$

is continuous on $L^2(0,1)$, vanishes on every $\rho_{1/n}$, and satisfies

$$L_{s_0}(\mathbf{1}) = \frac{1}{s_0} \neq 0.$$

Proof. Write $\sigma = \mathrm{Re}(s_0)$. Since $\sigma > 1/2$, the kernel $x^{\overline{s_0}-1}$ belongs to $L^2(0,1)$, so L_{s_0} is continuous by Cauchy–Schwarz. The Mellin bridge row and $\zeta(s_0) = 0$ give

$$L_{s_0}(\rho_{1/n}) = \frac{(n^{-1} - n^{-s_0})\zeta(s_0)}{s_0} = 0$$

for every $n \geq 1$. Direct integration gives

$$L_{s_0}(\mathbf{1}) = \int_0^1 x^{s_0-1}\,dx = \frac{1}{s_0},$$

which is nonzero for a nontrivial zero. □

Theorem 9281.9 (Nyman exhaustion gives fixed-half readout)**.** *Assume the Nyman Mellin bridge row and the usual reflection consumption row for the completed zeta zero ledger. If*

$$\mathsf{NymanBeurlingExhaustion}$$

holds, then the completed zeta zero ledger has fixed-half readout.

Proof. By Definition 9281.8, any zero with $\mathrm{Re}(s_0) > 1/2$ gives a continuous functional that annihilates each $\rho_{1/n}$ but does not annihilate $\mathbf{1}$. This contradicts

$$\mathbf{1} \in \overline{\mathrm{span}\{\rho_{1/n} : n \geq 1\}}^{L^2(0,1)}.$$

The reflection consumption row then rules out the opposite open side of the strip by reflecting it into the side already excluded. Thus the zero ledger reads through the fixed-half surface. □

Definition 9281.10 (Golden Nyman square tower)**.** *Let $(F_m)_{m\geq 1}$ be the Fibonacci scale sequence used by the golden resolution ledger. Define*

$$V_m^{\varphi} := \mathrm{span}\{\rho_{1/n} : 1 \leq n \leq F_m\}, \qquad d_m^{\varphi} := \mathrm{dist}_{L^2}(\mathbf{1}, V_m^{\varphi}).$$

The golden Nyman square drops are

$$\Delta_m^{\varphi,\mathrm{NB}} := (d_m^{\varphi})^2 - (d_{m+1}^{\varphi})^2.$$

Corollary 9281.11 (Golden Nyman drops are squares)**.** *For every m,*

$$\Delta_m^{\varphi,\mathrm{NB}} = \|P_{m+1}^{\varphi}\mathbf{1} - P_m^{\varphi}\mathbf{1}\|_2^2 \geq 0,$$

where P_m^{φ} denotes orthogonal projection onto V_m^{φ}.

Proof. Apply Definition 9281.5 to the nested subspaces

$$V_m^{\varphi} \subseteq V_{m+1}^{\varphi}.$$

□

Definition 9281.12 (Golden Nyman Gram exhaustion)**.** *Let*

$$G_m^{\varphi} := \Big(\langle \rho_{1/i}, \rho_{1/j}\rangle_{L^2(0,1)}\Big)_{1\leq i,j\leq F_m}$$

and

$$u_m := \Big(\langle \mathbf{1}, \rho_{1/i}\rangle_{L^2(0,1)}\Big)_{1\leq i\leq F_m}.$$

The source row

$$\mathsf{GoldenNymanGramExhaustion}$$

asserts the finite matrix limit

$$\lim_{m\to\infty}\Big(1 - u_m^*(G_m^{\varphi})^+ u_m\Big) = 0,$$

where $(G_m^{\varphi})^+$ is the Moore–Penrose inverse.

Theorem 9281.13 (Golden Gram row is the golden distance row)**.** *For every m,*

$$(d_m^{\varphi})^2 = 1 - u_m^*(G_m^{\varphi})^+ u_m.$$

Consequently, GoldenNymanGramExhaustion *is precisely the row $d_m^{\varphi} \to 0$.*

Proof. The vector u_m records the inner products of $\mathbf{1}$ with the finite generator list, and G_m^{φ} is the corresponding Gram matrix. The orthogonal projection of $\mathbf{1}$ onto V_m^{φ} has squared norm

$$\|P_m^{\varphi}\mathbf{1}\|_2^2 = u_m^*(G_m^{\varphi})^+ u_m,$$

with the Moore–Penrose inverse handling possible linear dependence in the displayed generator list. Since $\|\mathbf{1}\|_2^2 = 1$ and $\mathbf{1} - P_m^{\varphi}\mathbf{1}$ is orthogonal to $P_m^{\varphi}\mathbf{1}$, the displayed distance identity follows. □

Corollary 9281.14 (Golden Nyman exhaustion gives fixed-half readout)**.** *Assume the Nyman Mellin bridge row and the usual reflection consumption row for the completed zeta zero ledger. If*

GoldenNymanGramExhaustion

holds, then the completed zeta zero ledger has fixed-half readout.

Proof. By Definition 9281.13, GoldenNymanGramExhaustion gives $d_m^{\varphi} \to 0$. Since $F_m \to \infty$, the golden spaces form a cofinal reciprocal-parameter subtower of the Nyman spaces. Hence

$$\mathbf{1} \in \overline{\operatorname{span}\{\rho_{1/n} : n \geq 1\}}^{L^2(0,1)}.$$

Apply Definition 9281.9. □

Definition 9281.15 (Residue readback on reciprocal intervals)**.** *For $n \geq 1$ and $k \geq 1$, write*

$$r_n(k) := k - n\left\lfloor \frac{k}{n} \right\rfloor.$$

Thus $0 \leq r_n(k) < n$ and $r_n(k)$ is the residue of k modulo n.

Theorem 9281.16 (Nyman generators have residue normal form)**.** *If*

$$\frac{1}{x} = k + t, \qquad k \in \mathbb{N}_{\geq 1}, \qquad 0 \leq t < 1,$$

then

$$\rho_{1/n}(x) = \frac{r_n(k)}{n}.$$

Proof. Write $k = qn + r_n(k)$. Since $0 \leq r_n(k) + t < n$, one has

$$\left\{\frac{k+t}{n}\right\} = \frac{r_n(k) + t}{n}.$$

Also $\{1/x\} = t$. Therefore

$$\rho_{1/n}(x) = \left\{\frac{k+t}{n}\right\} - \frac{1}{n}\{k+t\} = \frac{r_n(k)+t}{n} - \frac{t}{n} = \frac{r_n(k)}{n}.$$

□

Theorem 9281.17 (Explicit Nyman Gram readback)**.** *For $i, j \geq 1$, the finite Nyman Gram entries and constant-vector entries are*

$$\langle \rho_{1/i}, \rho_{1/j} \rangle = \frac{1}{ij} \sum_{k=1}^{\infty} r_i(k) r_j(k) \left(\frac{1}{k} - \frac{1}{k+1} \right),$$

and

$$\langle \mathbf{1}, \rho_{1/j} \rangle = \frac{\log j}{j}.$$

Proof. Use the change of variables $y = 1/x$. Then $dx = y^{-2}\, dy$, and the interval $y \in [k, k+1)$ corresponds to

$$x \in \left(\frac{1}{k+1}, \frac{1}{k} \right].$$

By Definition 9281.16, the product $\rho_{1/i}\rho_{1/j}$ is constant on that interval with value $r_i(k)r_j(k)/(ij)$. Hence

$$\int_{1/(k+1)}^{1/k} \rho_{1/i}(x)\rho_{1/j}(x)\, dx = \frac{r_i(k)r_j(k)}{ij} \int_k^{k+1} \frac{dy}{y^2}.$$

Since

$$\int_k^{k+1} \frac{dy}{y^2} = \frac{1}{k} - \frac{1}{k+1},$$

the Gram formula follows by summing over k. The formula for $\langle \mathbf{1}, \rho_{1/j} \rangle$ is the removable $s \to 1$ limit of the Mellin bridge row

$$\frac{(j^{-1} - j^{-s})\zeta(s)}{s},$$

namely $j^{-1} \log j$. □

Definition 9281.18 (Explicit Nyman Gram exhaustion)**.** *Let G_N^{res} be the matrix with entries given by Definition 9281.17, and let*

$$u_N^{\mathrm{res}} := \left(\frac{\log 1}{1}, \frac{\log 2}{2}, \ldots, \frac{\log N}{N} \right)^T.$$

The row

$$\mathsf{ExplicitNymanGramExhaustion}$$

asserts

$$\lim_{N \to \infty} \left(1 - (u_N^{\mathrm{res}})^* (G_N^{\mathrm{res}})^+ u_N^{\mathrm{res}} \right) = 0.$$

Theorem 9281.19 (Explicit Nyman Gram exhaustion gives Nyman exhaustion)**.** *If* ExplicitNymanGramExhaustion *holds, then*

$$\mathsf{NymanBeurlingExhaustion}$$

holds.

Proof. Definition 9281.17 identifies G_N^{res} and u_N^{res} with the Gram matrix and constant-vector row of the finite space V_N. The same projection calculation used in Definition 9281.13 gives

$$d_N^2 = 1 - (u_N^{\mathrm{res}})^* (G_N^{\mathrm{res}})^+ u_N^{\mathrm{res}}.$$

The displayed limit is therefore exactly $d_N \to 0$. □

Definition 9281.20 (Damped Mobius Nyman exhaustion)**.** *Let μ denote the Mobius function. A damped Mobius Nyman row is a finite family of weights*

$$\omega_m : \{1, \dots, F_m\} \to \mathbb{C}$$

and the associated vector

$$A_m^{\varphi}(x) := -\sum_{n=1}^{F_m} \mu(n)\omega_m(n)\rho_{1/n}(x) \in V_m^{\varphi}.$$

The source row

$$\mathsf{DampedMobiusNymanExhaustion}$$

asserts that such weights can be chosen with

$$\|A_m^{\varphi} - \mathbf{1}\|_2 \to 0.$$

Theorem 9281.21 (Damped Mobius exhaustion gives golden Gram exhaustion)**.** *If* DampedMobiusNymanExhausti *holds, then*

$$\mathsf{GoldenNymanGramExhaustion}$$

holds.

Proof. Since $A_m^{\varphi} \in V_m^{\varphi}$,

$$d_m^{\varphi} \le \|A_m^{\varphi} - \mathbf{1}\|_2.$$

The right side tends to 0, so $d_m^{\varphi} \to 0$. By Definition 9281.13, this is GoldenNymanGramExhaustion. □

Principle 9281.22 (Mobius truncation as obstruction detector)**.** *The undamped choice $\omega_m(n) = 1$ exposes the arithmetic load hidden inside the Hilbert exhaustion row. On intervals with $1/x \in [k, k+1)$ and $k \le N$, the raw sum*

$$-\sum_{n=1}^{N} \mu(n)\rho_{1/n}(x)$$

differs from $\mathbf{1}$ by the scalar

$$-k\sum_{n=1}^{N} \frac{\mu(n)}{n}.$$

Thus the low reciprocal blocks already demand cancellation in the weighted Mobius average. The damped row is therefore not an external shortcut; it is a finite Hilbert-space target that keeps the tail ledger visible.

Principle 9281.23 (Constructive square target)**.** *The Nyman square tower relocates the non-circular part of the finite positive-container route. Finite positive semidefiniteness is no longer the active burden: it is the ordinary Hilbert Gram fact Definition 9281.3. The active burden is the cofinality row*

$$\mathsf{GoldenNymanGramExhaustion}.$$

Thus the target chain is

$$\mathsf{DampedMobiusNymanExhaustion} \Longrightarrow \mathsf{GoldenNymanGramExhaustion} \Longrightarrow \mathsf{NymanBeurlingExhaustion} \Longrightarrow \textit{fixed-half ze}$$

with the Mellin bridge and reflection consumption rows displayed as analytic interfaces rather than hidden positivity assumptions.

Chapter 9282

Mobius Nyman Mollifier Bounds

chapter 9281 reduces the Hilbert container route to the finite row

$$\mathsf{DampedMobiusNymanExhaustion}.$$

The present chapter records a more inspectable sufficient package for that row. It separates the reciprocal interval computation into a low block controlled by a smoothed Mobius average and a tail block governed by weighted residue orthogonality.

Definition 9282.1 (Dirichlet mollifier readout)**.** *Let μ denote the Mobius function. For a finite damping row*

$$w_X : \mathbb{N}_{\geq 1} \to \mathbb{C}$$

with finite support, define the Dirichlet mollifier

$$A_X(s) := \sum_{n\geq 1} \mu(n) w_X(n) n^{-s}$$

and its Nyman readout

$$M_X(x) := -\sum_{n\geq 1} \mu(n) w_X(n) \rho_{1/n}(x).$$

The support condition makes M_X a finite Nyman vector.

Definition 9282.2 (Mellin mollifier error row)**.** *For $0 < \mathrm{Re}(s) < 1$, the Mellin bridge of Definition 9281.7 gives*

$$\int_0^1 M_X(x) x^{s-1}\, dx = -\frac{\zeta(s)}{s} \sum_{n\geq 1} \mu(n) w_X(n)(n^{-1} - n^{-s}).$$

Thus the error against $\mathbf{1}$ has Mellin readout

$$\int_0^1 (M_X(x) - \mathbf{1}) x^{s-1}\, dx = -\frac{1}{s} - \frac{\zeta(s)}{s} \sum_{n\geq 1} \mu(n) w_X(n)(n^{-1} - n^{-s}).$$

This row is only an analytic diagnostic; the Hilbert proof obligations below are stated directly in $L^2(0,1)$.

Definition 9282.3 (Golden damped Mobius readout)**.** *For the Fibonacci scale F_m, a golden damping row is a finitely supported weight*

$$w_m : \mathbb{N}_{\geq 1} \to \mathbb{C}$$

whose support is contained in a displayed finite multiple of F_m. It defines

$$M_m^{\varphi}(x) := -\sum_{n\geq 1} \mu(n) w_m(n) \rho_{1/n}(x).$$

The row

GoldenDampedMobiusExhaustion

asserts

$$\|M_m^{\varphi} - \mathbf{1}\|_2 \to 0.$$

Theorem 9282.4 (Golden damped Mobius exhaustion gives golden Nyman exhaustion)**.** *If* GoldenDampedMobiusE *holds and each M_m^{φ} belongs to V_m^{φ}, then*

GoldenNymanGramExhaustion

holds.

Proof. The distance from $\mathbf{1}$ to V_m^{φ} is bounded by the error of any displayed vector in V_m^{φ}. Hence

$$d_m^{\varphi} \leq \|M_m^{\varphi} - \mathbf{1}\|_2.$$

The right side tends to 0. Apply Definition 9281.13. □

Definition 9282.5 (Golden Mobius low block)**.** *For a golden damping row w_m, define*

$$S_m := \sum_{n\geq 1} \frac{\mu(n) w_m(n)}{n}.$$

The low-block bound is

$$F_m |S_m|^2 \to 0.$$

Definition 9282.6 (Golden residue tail block)**.** *Using $r_n(k)$ from Definition 9281.15, define the tail error*

$$T_m := \sum_{k>F_m} \left| 1 + \sum_{n\geq 1} \mu(n) w_m(n) \frac{r_n(k)}{n} \right|^2 \left(\frac{1}{k} - \frac{1}{k+1} \right).$$

The residue tail bound is

$$T_m \to 0.$$

Definition 9282.7 (Golden Mobius low-tail bounds)**.** *The source row*

GoldenMobiusLowTailBounds

asserts the existence of golden damping rows w_m such that the low-block bound of Definition 9282.5 and the tail bound of Definition 9282.6 both hold.

Lemma 9282.8 (Low block bound controls the displayed low intervals)**.** *Assume $w_m(n) = 1$ for $1 \le n \le F_m$. On reciprocal intervals with $1/x \in [k, k+1)$ and $k \le F_m$, the low-block contribution to $M_m^{\varphi}(x) - \mathbf{1}$ is*

$$-kS_m.$$

Consequently, its L^2 contribution is bounded by

$$F_m|S_m|^2.$$

Proof. By Definition 9281.16,

$$\rho_{1/n}(x) = \frac{r_n(k)}{n}.$$

For $k \le F_m$ and $n \le F_m$,

$$r_n(k) = k - n\left\lfloor \frac{k}{n} \right\rfloor.$$

Therefore the raw low block reads

$$-\sum_{n \le F_m} \mu(n)\frac{r_n(k)}{n} = -k \sum_{n \le F_m} \frac{\mu(n)}{n} + \sum_{n \le k} \mu(n)\left\lfloor \frac{k}{n} \right\rfloor.$$

The divisor identity gives

$$\sum_{n \le k} \mu(n)\left\lfloor \frac{k}{n} \right\rfloor = 1.$$

After subtracting $\mathbf{1}$, the displayed low error is $-kS_m$. The interval weight is

$$\int_k^{k+1} \frac{dy}{y^2} = \frac{1}{k} - \frac{1}{k+1},$$

so

$$\sum_{k \le F_m} k^2|S_m|^2\left(\frac{1}{k} - \frac{1}{k+1}\right) \le F_m|S_m|^2.$$

□

Theorem 9282.9 (Low-tail bounds give golden damped Mobius exhaustion)**.** *If* GoldenMobiusLowTailBounds *holds and the damping rows have unit low block $w_m(n) = 1$ for $n \le F_m$, then*

GoldenDampedMobiusExhaustion

holds.

Proof. Split the L^2 norm of $M_m^{\varphi} - \mathbf{1}$ along the reciprocal intervals $1/x \in [k, k+1)$. The part with $k \le F_m$ tends to 0 by Definition 9282.8 and the low-block bound. The part with $k > F_m$ is exactly the tail quantity T_m, hence tends to 0 by the residue tail bound. Therefore

$$\|M_m^{\varphi} - \mathbf{1}\|_2 \to 0.$$

□

Corollary 9282.10 (Low-tail bounds give fixed-half readout)**.** *Assume* GoldenMobiusLowTailBounds*, unit low blocks, and the Mellin and reflection consumption rows of Definition 9281.9. Then the completed zeta zero ledger has fixed-half readout.*

Proof. Definition 9282.9 gives GoldenDampedMobiusExhaustion. The vectors lie in the golden Nyman spaces, so Definition 9282.4 gives GoldenNymanGramExhaustion. Finally apply Definition 9281.14. □

Principle 9282.11 (Mobius residue orthogonality target)**.** *The low-tail package isolates the remaining analytic burden. The low row asks for sufficient cancellation of the smoothed Mobius average*

$$\sum_{n\geq 1} \frac{\mu(n)w_m(n)}{n},$$

while the tail row asks for weighted orthogonality between $\mu(n)w_m(n)$ and the residue readbacks $r_n(k)/n$ on the reciprocal tail. In named form the target chain is

GoldenMobiusLowTailBounds $\Longrightarrow$ GoldenDampedMobiusExhaustion $\Longrightarrow$ GoldenNymanGramExhaustion $\Longrightarrow$ *fixed-half*

Chapter 9283

BEDC–Golden–Zeta Generation Machine

chapter 9351 supplies the raw mark-history kernel, chapter 9255 supplies the RH-free zero-generation interface, chapter 9259 supplies the golden resolution tower, and chapter 9260 isolates the Lee–Yang bridge. The present chapter records the composite machine obtained by placing those rows in one source order. It is a conditional interface: the fixed-half readout is produced only after the source rows listed below have been supplied.

Definition 9283.1 (BEDC–Golden–Zeta generation machine)**.** *A* BEDC–Golden–Zeta generation machine *is a finite-to-limit source packet*

$$\mathfrak{M}_\zeta$$

with the following displayed rows.

The kernel row is the raw mark and history spine

$$\mathsf{b0}, \mathsf{b1} \longrightarrow \mathsf{BHist}$$

of Definition 9351.1, equipped with the one-step time row of Definition 9361.15. Thus time is constructor-tail generation over histories rather than an ambient parameter.

The zero row is an RH-free generated zero carrier

$$Z = \mu F_Z$$

with a realization

$$\varepsilon : Z \to \mathcal{Z}_\zeta$$

as in Definition 9255.2. Its source completeness row is the essential coverage component

$$\prod_{u:\mathcal{Z}_\zeta} \sum_{z:Z} \varepsilon(z) \sim u.$$

The arithmetic row assigns active heights

$$h_a > 0$$

from the logarithmic prime or integer ledger, so that centered zeta channels have the form

$$U_a(s) := e^{-(s-1/2)h_a}.$$

The resolution row is a golden resolution ruler

$$\delta_m = \varphi^{-m}, \qquad N_m = F_{Lm},$$

with Zeckendorf truncation ledgers, as in Definition 9259.10 and Definition 9259.11.

The certificate row is a compatible resolution certificate tower in the sense of Definition 9259.5, carrying the triadic invariant of Definition 9257.20 at every displayed level.

The bridge row is a cofinal family of golden Lee–Yang bridge witnesses in the sense of Definition 9260.3. This row connects the weak eta capture packet to a lifted Fibonacci path packet with the same zero-island readout.

The defect row is the scale-energy extraction

$$E_m(s) = r^2 W_m + o(W_m), \qquad W_m \to \infty,$$

for readouts

$$s = \frac{1}{2} + r + it.$$

Definition 9283.2 (Constructive layer witness)**.** *A* constructive layer witness *for a machine level is a quadruple*

$$(O, G, I, F).$$

Here O is the displayed object row, G is the generator row producing that object from the previous layer, I is the invariant transported by the generator, and F is the projection-loss row recording what the current layer forgets when read by the next visible observer. The loss row is a ledger, not a license to introduce hidden structure.

Definition 9283.3 (Constructive BEDC–Golden–Zeta machine)**.** *A* constructive BEDC–Golden–Zeta machine *is a BEDC–Golden–Zeta generation machine together with constructive layer witnesses for the source chain*

$$\mathsf{BMark} \to \mathsf{BHist} \to \mathsf{Time} \to \mathsf{PrimeLedger} \to \mathsf{Hamiltonian} \to \mathsf{GoldenResolution} \to \mathsf{EtaPacket} \to \mathsf{ZeckendorfPacket} \to \mathsf{S}$$

Each transition must be generated from the preceding object and the triadic invariant rows. A later layer may inspect the projection loss of an earlier layer only through the displayed ledger F; it may not repair that loss by importing an ungenerated coordinate.

Definition 9283.4 (Machine legality)**.** *The machine $\mathfrak{M}_\zeta$ is* legal *when its hidden coordinates are confined to the compact phase row and finite prime-register fibres:*

$$\mathbb{T}, \qquad \prod_{p \in S} \mathbb{Z}_p,$$

and when no constructor, bridge comparison, tail row, or limit socket imports an independent real positive closure coordinate

$$\mathbb{R}_{>0}.$$

Equivalently, the only admissible readback of

$$U_a(s) = e^{-rh_a} e^{-ith_a}$$

inside the closed machine treats e^{-ith_a} as phase data and requires e^{-rh_a} to be accounted for by the defect ledger rather than absorbed as gauge.

Principle 9283.5 (No downstream smuggling)**.** *In a constructive BEDC–Golden–Zeta machine, no layer after a projection may supply data that the previous generator did not produce. Thus a Lee–Yang boundary row may stabilize an eta zero island only through a displayed same-island bridge; it may not choose a stable packet after the zero is known. A solenoid row may hide only phase and prime-register fibres; it may not supply a real positive scale coordinate. A limit row may read a compatible tower; it may not add an external zero presentation. This is the layerwise form of machine legality.*

Definition 9283.6 (Internal golden ruler)**.** *An* internal golden ruler *at level m is the row*

$$\mathcal{R}_m := (\delta_m, N_m, Z_m)$$

where

$$\delta_m = \varphi^{-m}, \qquad N_m = F_{Lm},$$

and Z_m is the Zeckendorf resolution ledger of Definition 9259.11 for N_m. It is internal when the following three rows are generated inside the same BEDC packet.

The P *row is Zeckendorf source uniqueness: the Fibonacci-block address of N_m is not replaceable by another unledgered truncation source.*

The U *row is the no-adjacent path boundary: the digits of Z_m lie in the stable Zeckendorf window, hence in the path carrier used by Definition 9260.1. This supplies the unit-boundary source for the later Lee–Yang lift; it does not by itself prove zeta boundary stability.*

The T *row is Fibonacci refinement:*

$$\mathcal{R}_m \rightsquigarrow \mathcal{R}_{m+1},$$

read as another level of the same generated ruler rather than as a change of external measurement.

Definition 9283.7 (Triadic ruler invariance)**.** *A family of internal golden rulers*

$$(\mathcal{R}_m)_{m\geq 0}$$

satisfies triadic ruler invariance *when every ruler level carries the* P, U, *and* T *rows of Definition 9283.6, and every refinement map*

$$\mathcal{R}_m \rightsquigarrow \mathcal{R}_{m+1}$$

transports those rows. The ruler is then part of the generated system: it is not an outside observer, and it cannot supply an outside $\mathbb{R}_{>0}$ scale ledger.

Principle 9283.8 (Machine source order)**.** *The source order of Definition 9283.1 is strict. Raw marks generate histories. Histories provide the time row. The generated zero carrier supplies the zero-presentation surface. The prime ledger supplies logarithmic heights. The golden ruler supplies finite resolution. Weak eta capture supplies local zero islands. The Lee–Yang bridge supplies boundary stability for the same islands. The defect-energy row then turns any off-half radial coordinate into an unbounded normal-energy readout. No later row may be used as an unledgered source for an earlier row.*

Theorem 9283.9 (Constructive layer discipline preserves legality)**.** *Let a constructive BEDC–Golden–Zeta machine have legal first layer, and assume every generator row transports the triadic invariant and obeys Definition 9283.5. Then every displayed layer of the machine is legal in the sense needed by Definition 9283.4.*

Proof. Proceed by induction along the displayed source chain of Definition 9283.3. The base layer is legal by hypothesis. For the induction step, assume the preceding layer is legal. The generator row transports the source, unit-boundary, and time invariants to the next layer. The projection-loss ledger records exactly what becomes invisible at that transition, while Definition 9283.5 forbids the next layer from replacing that loss by an ungenerated coordinate. Hence the compact phase and prime-register boundaries remain the only hidden gauges, and no $\mathbb{R}_{>0}$ closure coordinate is introduced. This is precisely the legality condition for the next layer. □

Principle 9283.10 (Internal rulers remove the measurement escape)**.** *If the golden ruler is external, a failure of the route could be hidden in the choice of resolution, truncation, or refinement schedule. If the ruler is internal and satisfies Definition 9283.7, the measuring rows themselves are subject to the same source, boundary, and time invariants as the zero certificate tower. Thus an off-half radial factor cannot be reclassified as a feature of measurement; it must appear in the defect ledger of the same closed machine.*

Theorem 9283.11 (Legal generation machines exclude radial limits)**.** *Let a legal BEDC–Golden–Zeta generation machine produce a completed zero certificate tower with limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

If its bridge witnesses supply zero-compatible boundary bounds and its defect row satisfies

$$E_m(s_\infty) = r^2 W_m + o(W_m), \qquad W_m \to \infty,$$

then $r = 0$.

Proof. The certificate row of the machine is a resolution certificate tower. The bridge row supplies the boundary-stability component by Definition 9260.17; the legality row supplies the refusal of hidden real-positive scale gauge; the defect row supplies cofinal growth and scale-energy extraction. Thus the tower is legal boundary-stable in the sense of Definition 9257.22. Applying Definition 9257.23 gives $r = 0$. □

Theorem 9283.12 (Internal ruler invariance excludes measurement exceptions)**.** *Let a BEDC–Golden–Zeta generation machine be legal, and assume its golden ruler family satisfies triadic ruler invariance. If each zero certificate level is generated from the corresponding internal ruler level and the tower satisfies the defect extraction and boundary-bound rows of Definition 9283.11, then its limit readout has* $r = 0$.

Proof. By triadic ruler invariance, every refinement level of the ruler carries the P, U, and T rows and transports them to the next level. Since each certificate level is generated from the corresponding ruler level, the certificate tower cannot appeal to an external resolution row, external truncation row, or external scale row. The only visible off-half component of

$$e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a}$$

is therefore the defect term e^{-rh_a} recorded by the same machine. The hypotheses then reduce exactly to the legal generation-machine exclusion of Definition 9283.11, which gives $r = 0$. □

Definition 9283.13 (BEDC–Golden–Zeta source row)**.** *The source row*

$$\mathsf{BEDCGoldenZetaMachine}_\zeta$$

asserts that every accepted non-trivial completed zeta zero is presented by a legal BEDC–Golden–Zeta generation machine whose sufficiently fine levels carry compatible golden Lee–Yang bridge witnesses.

Definition 9283.14 (Constructive machine source row)**.** *The source row*

$$\mathsf{ConstructiveBEDCGoldenZetaMachine}_\zeta$$

asserts that every accepted non-trivial completed zeta zero is presented by a constructive BEDC–Golden–Zeta machine whose layer witnesses obey Definition 9283.5, whose golden ruler satisfies triadic ruler invariance, and whose sufficiently fine levels carry compatible golden Lee–Yang bridge witnesses.

Definition 9283.15 (Zeta constraint container)**.** *Let Λ be the ledger of a compatible golden zero-certificate tower. A* zeta constraint container *is the specialization of Definition 9462.991 to the RH-facing candidate class*

$$\mathcal{G}_{\zeta,\varphi}$$

whose visible candidates include the critical container

$$\Gamma_{\mathrm{crit}} := \{s : \Re(s) = 1/2\}$$

and the radial containers

$$\Gamma_r := \{s : \Re(s) = 1/2 + r\}, \qquad r \neq 0.$$

Its probe row reads zero-island capture, Rouché or winding data, phase, prime-register fibres, Zeckendorf resolution ledgers, boundary stability, and defect energy. Its gauge row may contain only compact phase, finite prime-register, and Zeckendorf-ledger freedoms already displayed by the machine. The container is therefore a stabilized closure of ledger constraints, not a prior host line into which zeros are placed.

Definition 9283.16 (Zeta critical-container forcing)**.** *For the limit ledger Λ_∞ of a compatible golden zero-certificate tower, write*

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}]$$

when Γ_{crit} realizes the ledger and every candidate

$$\Delta \in \mathcal{G}_{\zeta,\varphi}$$

realizing the same ledger has the same probe readout as Γ_{crit} across the probe row of Definition 9283.15, modulo only the displayed compact phase, prime-register, and Zeckendorf gauge rows. Equivalence at the completed-zero readout means that every presentation carried by Δ has $r = 0$ in

$$s = \frac{1}{2} + r + it.$$

Theorem 9283.17 (Constructive source rows imply machine source rows)**.** *If*

$$\mathsf{ConstructiveBEDCGoldenZetaMachine}_\zeta$$

holds, then

$$\mathsf{BEDCGoldenZetaMachine}_\zeta$$

holds.

Proof. The constructive source row supplies, for every accepted non-trivial completed zeta zero, a constructive BEDC–Golden–Zeta machine. By Definition 9283.9, its layers are legal. The same source row also supplies triadic ruler invariance and compatible golden Lee–Yang bridge witnesses. Forgetting the displayed layer-witness quadruples leaves exactly the legal BEDC–Golden–Zeta generation machine required by Definition 9283.13. □

Corollary 9283.18 (The machine source row gives fixed-half readout)**.** *If*

$$\mathsf{BEDCGoldenZetaMachine}_\zeta$$

holds, then every accepted non-trivial completed zeta zero has fixed-half readout.

Proof. Let an accepted non-trivial completed zeta zero be given. The source row supplies a legal generation machine presenting it. By Definition 9283.11, its limit readout has $r = 0$ in the decomposition

$$s_\infty = \frac{1}{2} + r + it.$$

Therefore $\Re(s_\infty) = 1/2$. □

Corollary 9283.19 (Constructive machine source rows give fixed-half readout)**.** *If*

$$\mathsf{ConstructiveBEDCGoldenZetaMachine}_\zeta$$

holds, then every accepted non-trivial completed zeta zero has fixed-half readout.

Proof. Apply Definition 9283.17, then Definition 9283.18. □

Theorem 9283.20 (Constructive machines force the critical container)**.** *If*

$$\mathsf{ConstructiveBEDCGoldenZetaMachine}_\zeta$$

holds, then the limit ledger of every accepted non-trivial completed zeta zero satisfies

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}].$$

Proof. Fix an accepted non-trivial completed zeta zero and its limit ledger. The constructive source row supplies a constructive BEDC–Golden–Zeta machine for this zero. By Definition 9283.9, all displayed layers are legal, and by Definition 9283.17 the underlying legal generation-machine source row is present.

Let $\Delta \in \mathcal{G}_{\zeta,\varphi}$ realize the same limit ledger. If the readout of Δ has

$$s = \frac{1}{2} + r + it$$

with $r \neq 0$, then the defect row of the machine gives the radial energy extraction

$$E_m(s) = r^2 W_m + o(W_m), \qquad W_m \to \infty,$$

while the compatible golden Lee–Yang bridge witnesses supply the zero-compatible boundary bound. Since Δ realizes the same ledger, it must realize these rows with the same legal gauge restrictions. Thus the Δ-readout satisfies the hypotheses of Definition 9283.11, which forces $r = 0$, a contradiction. Hence every surviving candidate has fixed-half readout. The remaining invisible rows are exactly the displayed compact phase, prime-register, and Zeckendorf gauge rows, so all surviving candidates have the same probe readout as Γ_{crit} on the probes of Definition 9283.15. This is the forcing judgment of Definition 9283.16. □

Principle 9283.21 (Machine proof load)**.** *The machine formulation separates the proof load into source rows. The kernel and history rows are supplied by the BEDC spine. The golden resolution and weak eta capture rows are supplied by the local Rouché construction of Definition 9259.14. The strong source content has three coupled parts: constructive layer witnesses with no downstream smuggling, triadic ruler invariance for the golden resolution row, and compatible golden Lee–Yang bridge witnesses for the same zero islands, including dominance, same-island transfer, radial purity, and far-end energy control. Those rows are the mathematical content of*

$$\mathsf{BEDCGoldenZetaMachine}_{\zeta}.$$

Under those rows, the public geometric readout is the derived constraint container judgment

$$\Lambda_{\infty} \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}],$$

not an initial assumption that zero certificates already live on the critical line.

Chapter 9284

Internal Ruler Constraint Container

chapter 9283 packages the BEDC–Golden–Zeta machine, while Definition 9462.991 supplies the general constraint-container interface. The present chapter records the common source row in which the golden ruler is itself part of the generated container. The point is not to place zero certificates inside a prior critical-line host; the point is to make the internal ruler, the zero-certificate tower, and the surviving candidate container use the same source, boundary, and time ledgers.

Definition 9284.1 (Container-mature internal ruler)**.** *Let*

$$(\mathcal{R}_m)_{m\geq 0}$$

be an internal golden ruler family in the sense of Definition 9283.6. It is container-mature for a zero-certificate tower *when the following rows are present.*

The source row is triadic ruler invariance:

$$\mathcal{R}_m \rightsquigarrow \mathcal{R}_{m+1}$$

transports the P, U, *and* T *rows of Definition 9283.7.*

The weak capture row is a cofinal family of golden eta packets

$$\mathcal{G}_m(\rho) = (D_m, P_m, R_m, \Gamma_m)$$

as in Definition 9259.13, with the Rouché boundary gap supplied by Definition 9259.14 where its hypotheses apply.

The boundary row is a cofinal family of golden Lee–Yang bridge witnesses in the sense of Definition 9260.3. Thus the same zero-island ledger has a lifted path packet, a Lee–Yang unit-boundary certificate, same-island transfer, radial purity, and dominance.

The audit row records projection loss through the displayed Zeckendorf, prime-register, phase, tail, boundary, and defect ledgers. It does not allow the ruler to supply an independent real-positive scale coordinate.

Definition 9284.2 (Internal-ruler constraint container package)**.** *An* internal-ruler constraint container package *for a completed zeta zero readout consists of:*

- *a constructive BEDC–Golden–Zeta machine in the sense of Definition 9283.3;*
- *a container-mature internal ruler family* $(\mathcal{R}_m)_{m\geq 0}$*;*
- *a compatible zero-certificate tower generated levelwise from* $\mathcal{R}_m$*;*

- *the zeta constraint container of Definition 9283.15 for the limit ledger Λ_∞;*
- *the defect extraction and cofinal active-height rows*

$$E_m(s) = r^2 W_m + o(W_m), \qquad W_m \to \infty,$$

for readouts

$$s = \frac{1}{2} + r + it.$$

Principle 9284.3 (The ruler cannot be the scale reservoir)**.** *In an internal-ruler constraint container package, the ruler is part of the audited container rather than an outside measurement device. Hence an off-half factor*

$$e^{-rh_a} \in \mathbb{R}_{>0}$$

cannot be reclassified as a choice of resolution, truncation schedule, Zeckendorf address, or path-model lift. The only admissible hidden rows are the displayed compact phase row, the finite prime-register fibres, and the Zeckendorf ledger rows. Any real-positive radial factor must therefore appear in the defect-energy ledger.

Theorem 9284.4 (Internal-ruler containers exclude radial readouts)**.** *Let an internal-ruler constraint container package have limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

Then $r = 0$.

Proof. The package contains a constructive BEDC–Golden–Zeta machine and a container-mature internal ruler family. By Definition 9283.9, the displayed layers are legal. By Definition 9284.1, the ruler transports the triadic rows and does not supply an independent $\mathbb{R}_{>0}$ gauge. Thus Definition 9284.3 forces any off-half factor to be read through the defect ledger.

The same package supplies the golden Lee–Yang bridge witnesses, hence the zero-compatible boundary bound, and it also supplies

$$E_m(s_\infty) = r^2 W_m + o(W_m), \qquad W_m \to \infty.$$

These are exactly the hypotheses of Definition 9283.11. Therefore $r = 0$. □

Definition 9284.5 (Internal-ruler constraint-container source row)**.** *The source row*

$$\mathsf{InternalRulerConstraintContainer}_{\zeta,\varphi}$$

asserts that every accepted non-trivial completed zeta zero is presented by an internal-ruler constraint container package.

Definition 9284.6 (Internal golden ruler zeta certificate)**.** *The source row*

$$\mathsf{InternalGoldenRulerZetaCertificate}_{\zeta,\varphi}$$

asserts that every accepted non-trivial completed zeta zero has a cofinal golden eta certificate tower whose ruler family is internal and container-mature, whose Zeckendorf truncation ledgers are unique under refinement, and whose sufficiently fine levels carry golden Lee–Yang bridge witnesses compatible with the same zero island.

Definition 9284.7 (Golden zeta constraint-container route)**.** *The named route*

$$\mathrm{ConstraintContainer}^{\uparrow}_{\zeta,\varphi}$$

is the conjunction of the following source rows:

$$\mathsf{GoldenCoverage}_{\zeta,\varphi}, \qquad \mathsf{InternalGoldenRuler}_{\varphi}, \qquad \mathsf{ZetaGoldenLeeYangBridge}, \qquad \mathsf{CofinalGrowth}_{\zeta,\varphi}.$$

Here $\mathsf{GoldenCoverage}_{\zeta,\varphi}$ *means that accepted non-trivial completed zeta zeros enter golden BEDC certificate towers;* $\mathsf{InternalGoldenRuler}_{\varphi}$ *means that the ruler family is generated internally and transports the triadic rows; the bridge row is the same-island Lee–Yang source row; and the growth row is the cofinal active-height condition*

$$W_m \to \infty.$$

Definition 9284.8 (Golden bridge analytic subrows)**.** *The analytic content of* $\mathsf{ZetaGoldenLeeYangBridge}$ *is the source package of Definition 9260.16. The dominance-facing part is the row*

$$\mathsf{GoldenDominantBranchTheorem}$$

which is supplied only after a golden spectral expansion and spectral gap have selected the Zeckendorf–Fibonacci path packet as the zero-relevant branch by source, dominance, and tail ledgers. The boundary-facing part is the row

$$\mathsf{GoldenStableRoucheHomotopy}$$

which supplies a boundary-nonvanishing homotopy from the golden eta packet to the lifted path packet on the same zero island, while also preserving the strict inside and strict outside Lee–Yang zero-free regions. These rows are source obligations; none follows from the existence of a golden ruler alone.

Definition 9284.9 (Window-six finite audit anchor)**.** *The source row*

$$\mathsf{WindowSixFiniteAnchor}_{\zeta,\varphi}$$

admits the Window-six finite audit laboratory of Definition 9349.10 as the finite anchor for the golden zeta container route. It consists of the following finite records.

The visible-address record is

$$|X_6| = F_8 = 21,$$

with X_6 *the sixth no-adjacent Zeckendorf window of Definition 9347.1. The ambient fold record is*

$$\Omega_6 = \{0,1\}^6 \longrightarrow X_6, \qquad |\Omega_6| = 64.$$

The hidden-fibre histogram record is

$$d_6(x) \in \{2,3,4\},$$

with

$$\#\{x : d_6(x) = 2\} = 8, \qquad \#\{x : d_6(x) = 3\} = 4, \qquad \#\{x : d_6(x) = 4\} = 9,$$

so that

$$8 \cdot 2 + 4 \cdot 3 + 9 \cdot 4 = 64.$$

The visible-insufficiency record is the strict inequality

$$\dim_{\mathbb{C}} Z_6^{(3)} \leq 12 < 21 = |X_6|.$$

The oriented-boundary record keeps boundary-sheet parity as an integral or mod-2 ledger row, not as data visible in the rational continuous envelope. The moment-audit record requires signed moments for the phase-character sectors to reconstruct the displayed hidden-fibre histogram without exposing every ambient microstate.

The scope record is part of the source row: this finite anchor supplies a visible-insufficiency, hidden-fibre, parity, and signed-moment audit model. It does not supply zeta zero coverage, golden propagation, same-island Lee–Yang transport, cofinal growth, or an infinite boundary-stability theorem.

Definition 9284.10 (Golden propagation source row)**.** *The source row*

$$\mathsf{GoldenPropagation}_{\zeta,\varphi}$$

asserts that the finite audit structure displayed by $\mathsf{WindowSixFiniteAnchor}_{\zeta,\varphi}$ *is transported to the cofinal golden certificate tower used by the zeta route. It contains the internal-ruler row*

$$\mathsf{InternalGoldenRuler}_{\varphi},$$

preserves Zeckendorf ledger uniqueness under refinement, keeps hidden-fibre audit data through the tower, transports oriented boundary parity by discrete ledger rows, and refuses any replacement of these rows by a real-positive radial scale coordinate. This row is not a consequence of Window six alone; it is the propagation obligation that connects the finite anchor to cofinal golden resolution.

Theorem 9284.11 (Window-six anchored rows supply the golden container route)**.** *If*

$$\mathsf{WindowSixFiniteAnchor}_{\zeta,\varphi}, \quad \mathsf{GoldenPropagation}_{\zeta,\varphi}, \quad \mathsf{GoldenCoverage}_{\zeta,\varphi}, \quad \mathsf{ZetaGoldenLeeYangBridge}, \quad \mathsf{C}$$

hold, then

$$\mathrm{ConstraintContainer}^{\uparrow}_{\zeta,\varphi}$$

holds.

Proof. By Definition 9284.10, the propagation row contains

$$\mathsf{InternalGoldenRuler}_{\varphi}.$$

Together with the displayed golden coverage, golden Lee–Yang bridge, and cofinal growth rows, this is exactly the conjunction defining

$$\mathrm{ConstraintContainer}^{\uparrow}_{\zeta,\varphi}$$

in Definition 9284.7. The Window-six row supplies the finite audit anchor for that propagation obligation, while the infinite route is supplied only by the listed cofinal and bridge rows. □

Corollary 9284.12 (Window-six anchored rows force the critical container)**.** *Under the hypotheses of Definition 9284.11, the limit ledger of every accepted non-trivial completed zeta zero satisfies*

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}].$$

Proof. Apply Definition 9284.11 and then Definition 9284.18. □

Theorem 9284.13 (Internal golden certificates supply container packages)**.** *If*

$$\mathsf{InternalGoldenRulerZetaCertificate}_{\zeta,\varphi}$$

holds, then

$$\mathsf{InternalRulerConstraintContainer}_{\zeta,\varphi}$$

holds.

Proof. Fix an accepted non-trivial completed zeta zero. The internal golden certificate row supplies a cofinal golden eta certificate tower and an internal container-mature ruler family. The unique Zeckendorf truncation ledgers provide the source-address part of the ruler, while the compatible golden Lee–Yang bridge witnesses provide the boundary, same-island transfer, radial-purity, and dominance rows. Together with the constructive BEDC–Golden–Zeta machine rows of Definition 9283.3, these are exactly the components listed in Definition 9284.2. Hence the zero is presented by an internal-ruler constraint container package. □

Theorem 9284.14 (Golden zeta container route supplies internal golden certificates)**.** *If*

$$\mathrm{ConstraintContainer}^{\uparrow}_{\zeta,\varphi}$$

holds, then

$$\mathsf{InternalGoldenRulerZetaCertificate}_{\zeta,\varphi}$$

holds.

Proof. The golden coverage row supplies golden BEDC certificate towers for accepted non-trivial completed zeta zeros. The internal ruler row makes the corresponding golden ruler family internal and triadic under refinement. The bridge row supplies cofinal golden Lee–Yang bridge witnesses on the same zero islands, while the cofinal growth row supplies the active-height growth required by the defect ledger. These are precisely the components required by Definition 9284.6. □

Theorem 9284.15 (Internal-ruler containers supply constructive machines)**.** *If*

$$\mathsf{InternalRulerConstraintContainer}_{\zeta,\varphi}$$

holds, then

$$\mathsf{ConstructiveBEDCGoldenZetaMachine}_{\zeta}$$

holds.

Proof. For each accepted non-trivial completed zeta zero, the internal-ruler source row supplies an internal-ruler constraint container package. Its first component is a constructive BEDC–Golden–Zeta machine, and its remaining components supply the container-mature internal ruler, compatible certificate tower, bridge witnesses, and defect rows. Forgetting those additional container rows leaves the source row required by Definition 9283.14. □

Corollary 9284.16 (Internal-ruler containers force the critical container)**.** *If*

$$\mathsf{InternalRulerConstraintContainer}_{\zeta,\varphi}$$

holds, then the limit ledger of every accepted non-trivial completed zeta zero satisfies

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}].$$

Proof. Apply Definition 9284.15 and then Definition 9283.20. □

Corollary 9284.17 (Internal golden certificates force the critical container)**.** *If*

$$\mathsf{InternalGoldenRulerZetaCertificate}_{\zeta,\varphi}$$

holds, then the limit ledger of every accepted non-trivial completed zeta zero satisfies

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}].$$

Proof. Apply Definition 9284.13 and then Definition 9284.16. □

Corollary 9284.18 (Golden zeta containers force the critical container)**.** *If*

$$\mathrm{ConstraintContainer}^{\uparrow}_{\zeta,\varphi}$$

holds, then the limit ledger of every accepted non-trivial completed zeta zero satisfies

$$\Lambda_\infty \Vdash_{\zeta,\varphi} [\Gamma_{\mathrm{crit}}].$$

Proof. Apply Definition 9284.14 and then Definition 9284.17. □

Principle 9284.19 (Proof load of the internal-ruler container)**.** *The internal-ruler container formulation separates four rows. The general constraint-container interface is supplied by Definition 9462.991. The golden ruler and Zeckendorf address rows are supplied by Definition 9259.10 and Definition 9259.11. The weak analytic capture row is supplied by Definition 9259.14. The strong source row is the same-island Lee–Yang bridge of Definition 9260.18. The container package closes only when these rows are displayed together with radial purity, dominance, cofinal active-height growth, and no external scale-gauge row.*

Principle 9284.20 (Bridge reduction for the internal golden certificate)**.** *The internal golden certificate row does not add a second analytic route beyond the golden bridge. The weak eta capture row supplies finite zero-island capture; the internal ruler row supplies source, refinement, and Zeckendorf address control; the constraint-container row records the surviving candidate class and admissible gauge. The only strong zeta-specific row is the same-island Lee–Yang bridge, namely a cofinal supply of the witnesses of Definition 9260.3. Thus failure of that bridge leaves the package at weak capture, while success of that bridge upgrades the package to the internal-ruler container source row. Definition 9284.8 records the analytic subrows of this bridge: spectral expansion and spectral gap for the selected path branch, and stable Rouché homotopy to the analytic packet on the same boundary.*

Chapter 9285

Arithmetic Spectral String Route

Definition 9255.11 reads the eta packet as a many-oscillator phase sum, while Definition 9257.13 reads a completed zero through a compatible tower of finite certificates. The present chapter records the same row in string-like spectral language. The word "string" here is a mode-organization term: one oscillator displays one mode, a finite packet displays finitely many modes, and the arithmetic spectral carrier organizes the compatible tower of logarithmic modes. No spacetime target, quantized physical spectrum, or imported physical formalism is assumed.

Definition 9285.1 (Arithmetic spectral string carrier)**.** *An* arithmetic spectral string carrier *is a displayed normal-mode readout*

$$X(t) = \sum_{n\geq 1} a_n e^{-ith_n}$$

on a licensed summability or certificate window, with logarithmic heights

$$h_n = \log n.$$

The eta critical-line carrier is the instance

$$a_n = (-1)^{n-1} n^{-1/2}, \qquad X_\eta(t) = \sum_{n\geq 1} (-1)^{n-1} n^{-1/2} e^{-it\log n}.$$

The prime ledger is the factorization row

$$\log n = \sum_p v_p(n) \log p,$$

so the displayed modes are arithmetic normal modes generated by prime frequency registers. The carrier is therefore string-like because it organizes a compatible normal-mode family, not because it imports an external physical string formalism.

Principle 9285.2 (String nodes are phase cancellations)**.** *A node of the conservative arithmetic spectral string is a zero readout of the phase carrier:*

$$X(t) = 0.$$

For the eta instance this is the critical-line cancellation

$$\eta(1/2 + it) = 0$$

whenever the displayed readout is licensed. A projected cancellation of an amplitude-biased readout is not yet a conservative string node. The certificate must also show that the active channels close by phase inside the unit boundary.

Definition 9285.3 (Centered string channel)**.** *For*

$$s = \frac{1}{2} + r + it$$

and an active logarithmic height $h_a > 0$, the centered string channel is

$$U_a(s) := e^{-(s-1/2)h_a} = e^{-rh_a} e^{-ith_a}.$$

The factor e^{-ith_a} is the conservative phase mode. The factor e^{-rh_a} is a real positive radial scale mode. Thus an off-half readout carries a second nonunitary coordinate unless the certificate supplies a legal scale ledger, which the closed arithmetic string row does not permit.

Definition 9285.4 (Unitary arithmetic string flow)**.** *A* unitary arithmetic string flow *is a phase evolution*

$$U(t) = e^{-itH}$$

on a displayed normal-mode carrier with

$$He_n = (\log n)e_n$$

on the mode basis under consideration, and with

$$U(t)^* U(t) = I.$$

The same Hamiltonian readout gives every active logarithmic mode the single time parameter t. A radial factor e^{-rH} is not part of this unitary flow; it is a damping or gain row.

Theorem 9285.5 (Unitary string admissibility excludes radial scale)**.** *Let a string certificate have limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

If some active height $h_a > 0$ satisfies the unit-boundary row

$$|e^{-(s_\infty - 1/2)h_a}| = 1,$$

then $r = 0$.

Proof. The centered channel factorization gives

$$|e^{-(s_\infty - 1/2)h_a}| = e^{-rh_a}.$$

The unit-boundary row therefore says

$$e^{-rh_a} = 1.$$

Since the real exponential is injective and $h_a > 0$, this implies $r = 0$. □

Definition 9285.6 (Closed arithmetic string certificate)**.** *A* closed arithmetic string certificate *for an accepted non-trivial completed zeta zero is a legal boundary-stable zero certificate tower in the sense of Definition 9257.22, together with the following mode interpretation:*

$$h_n = \log n, \qquad \log n = \sum_p v_p(n) \log p,$$

finite certificate levels are finite normal-mode packets, the time row is the unitary arithmetic string flow of Definition 9285.4, the boundary row is the unit-channel condition of Definition 9285.5, and the far-end row is the no-absorption discipline required by Definition 9257.22. Thus a zero is certified as a node of a closed conservative arithmetic spectral system, not as a cancellation produced by unledgered radial amplitude change.

Principle 9285.7 (Zeta closed arithmetic string target)**.** *The target*

$$\mathsf{ZetaClosedArithmeticString}$$

asserts that every accepted non-trivial completed zeta zero admits a closed arithmetic string certificate. Its source load has the following rows:

$$\mathsf{ZetaStringRealization},$$

$$\mathsf{StrongActiveCapture},$$

$$\mathsf{BoundaryStability},$$

$$\mathsf{NoFarEndAbsorption}_{\zeta},$$

and

$$\mathsf{ZeroPresentationCompleteness}.$$

The first row identifies the logarithmic normal-mode carrier; the second turns a zero into a canonical finite packet tower; the third supplies the unitary boundary row; the fourth prevents the infinite tail from serving as a radial scale reservoir; the fifth supplies coverage of all accepted non-trivial zero readouts.

Corollary 9285.8 (Closed arithmetic string coverage gives fixed-half readout)**.** *If* ZetaClosedArithmeticString *holds, then every accepted non-trivial completed zeta zero has real part* $1/2$*.*

Proof. Let u be an accepted non-trivial completed zeta zero. The target ZetaClosedArithmeticString supplies a closed arithmetic string certificate for u, hence a legal boundary-stable certificate tower by Definition 9285.6. Applying Definition 9257.24 gives the fixed-half readout. □

Principle 9285.9 (String language does not change the proof boundary)**.** *The arithmetic string language reorganizes the conservative oscillator route; it does not remove any source obligation. Weak eta capture still only supplies local analytic boxes. The closed string claim requires the strong certificate rows of Definition 9285.7. In particular, unitary normal-mode language does not by itself prove that every zeta zero is represented by a closed arithmetic string certificate, and it does not by itself prove boundary stability for the canonical packets.*

Concretized in §9139: Arithmetic spectral string certificate as finite normal-mode, prime-ledger, and unit-boundary NameCert data.

Chapter 9286

Thermodynamic No-Scale Route

chapter 9257 separates compact phase recurrence from radial scale time, while chapter 9285 reads the same distinction as a conservative normal-mode condition. The present chapter records the information-thermodynamic reading of that row. It treats an off-half real part as a Boltzmann tilt of the active mode distribution. The chapter supplies a no-scale implication under explicit stability and growth hypotheses; it does not supply the golden Lee–Yang bridge, strong active capture, or zero-presentation completeness.

Definition 9286.1 (Phase channel with hidden prime fibre)**.** *Fix a finite displayed prime family S. A phase channel with hidden prime fibre is a readout*

$$\pi_S : \Sigma_S \to \mathbb{T}$$

from the S-solenoid of Definition 9413.779, together with its invisible fibre

$$K_S := \ker(\pi_S).$$

The visible output is the unit phase row in $\mathbb{T}$. The hidden state is the compact prime-register fibre row carried by K_S. A consumer that receives only $\pi_S(x)$ has a phase output, not the full solenoid state x.

Principle 9286.2 (Kernel blindness of the phase channel)**.** *The phase channel of Definition 9286.1 is a coarse-graining row. Equality of visible outputs,*

$$\pi_S(x) = \pi_S(x'),$$

does not recover equality of solenoid states unless the prime-register fibre ledger is also present. In finite quotients this hidden part may be read as a fibre-size or fibre-volume entropy. Thus the channel has a nontrivial conditional-information row

$$\mathsf{HiddenInfo}(x \mid \pi_S(x)),$$

which is ledgered by compact prime-register data, not by a real-positive scale coordinate.

Definition 9286.3 (Solenoid legal fibre row)**.** *For a finite prime family S, the legal hidden fibre of the phase channel is the compact row*

$$K_S = \ker(\pi_S)$$

from Definition 9286.1. A zero-relevant packet is solenoid-legal when every hidden coordinate used to refine its phase readout factors through compact phase data or through this prime-register fibre. No coordinate of type $\mathbb{R}_{>0}$ is part of the solenoid-legal fibre row.

Theorem 9286.4 (Solenoid fibres exclude radial gauges)**.** *Let a centered active packet be solenoid-legal in the sense of Definition 9286.3. If a local channel has the form*

$$U_a(s) = e^{-rh_a} e^{-ith_a}, \qquad h_a > 0,$$

and its modulus row is supplied by the solenoid fibre, then $r = 0$.

Proof. The solenoid fibre is compact. A modulus row supplied by it is therefore a continuous positive real character of a compact phase carrier. By Definition 9413.781, such a character is trivial. Hence

$$|U_a(s)| = e^{-rh_a} = 1.$$

Since $h_a > 0$, taking logarithms gives $r = 0$. □

Principle 9286.5 (Solenoid–Pick conditional route)**.** *The solenoid row and the Pick row play distinct roles in the same triadic obstruction. The solenoid row determines the legal fibre of the observation cord: it permits visible unit phase and hidden compact prime-register data, but not a radial $\mathbb{R}_{>0}$ gauge. The Pick row of Definition 9271.2 determines whether the same observation cord embeds into a Hilbert space with nonnegative squared length. Consequently, an off-half zero has two incompatible readings: it requires a nontrivial radial factor e^{-rh_a}, while Definition 9286.4 refuses such a factor as hidden fibre and Definition 9271.13 turns the same radial defect into a finite negative Pick witness. Thus the conditional route is*

$$\mathsf{SolenoidLegalFibre} \times \mathsf{TriadicPickPositivity}_\xi \Longrightarrow \textit{fixed-half readout.}$$

Principle 9286.6 (No recovery by mirror scale)**.** *For a centered active channel*

$$U_a(s) = e^{-rh_a} e^{-ith_a},$$

the mirror row sends r to $-r$ and therefore exchanges damping with gain:

$$e^{-rh_a} \longleftrightarrow e^{rh_a}.$$

This creates the mirror amplitude row; it does not reconstruct the hidden solenoid fibre erased by π_S, and it does not return a single packet to the unit boundary. A recovery of hidden prime-register data requires the compact ledger of Definition 9286.2; a recovery of a radial factor requires an external $\mathbb{R}_{>0}$ ledger, which is excluded from closed phase packets.

Definition 9286.7 (Finite thermodynamic mode packet)**.** *A finite thermodynamic mode packet at level m consists of a finite active set A_m, positive heights*

$$h_a > 0 \qquad (a \in A_m),$$

and nonnegative weights w_a whose total weight is positive. The pure phase state is

$$\Psi_m(t) := \sum_{a \in A_m} c_a e^{-ith_a} e_a.$$

For a readout

$$s = \frac{1}{2} + r + it,$$

the amplitude-biased state is

$$\Psi_m(r,t) := \sum_{a \in A_m} c_a e^{-rh_a} e^{-ith_a} e_a.$$

The factor e^{-ith_a} is a reversible phase row. The factor e^{-rh_a} is a real-positive thermal tilt row.

Definition 9286.8 (Relative scale-defect energy)**.** *For a finite thermodynamic mode packet, let*

$$\bar{h}_m := \frac{\sum_{a\in A_m} w_a h_a}{\sum_{a\in A_m} w_a}.$$

The relative log-scale defect of $s = 1/2 + r + it$ at a is

$$\widetilde{d}_a(r) := -r(h_a - \bar{h}_m).$$

The relative scale-defect energy is

$$E_m^{\mathrm{rel}}(r) := \sum_{a\in A_m} w_a \widetilde{d}_a(r)^2.$$

Equivalently,

$$E_m^{\mathrm{rel}}(r) = r^2 V_m, \qquad V_m := \sum_{a\in A_m} w_a (h_a - \bar{h}_m)^2.$$

This relative form ignores a common normalization factor and records only mode-to-mode scale distortion.

Definition 9286.9 (Boltzmann tilt of an active packet)**.** *The normalized baseline distribution of a finite thermodynamic mode packet is*

$$p_a^{(0)} := \frac{w_a}{\sum_{b\in A_m} w_b}.$$

The off-half parameter r tilts it to

$$p_a^{(r)} := \frac{w_a e^{-2rh_a}}{Z_m(r)}, \qquad Z_m(r) := \sum_{b\in A_m} w_b e^{-2rh_b}.$$

Thus r is not a phase coordinate. It changes the relative distribution of active modes by an exponential-family row.

Principle 9286.10 (Relative entropy detects radial tilt)**.** *In the packet of Definition 9286.9, the free-energy row*

$$F_m(r) := -\log Z_m(r)$$

and the relative entropy

$$D_{\mathrm{KL}}(p^{(r)} \| p^{(0)})$$

read the same radial phenomenon as the scale-defect ledger. Near $r = 0$, the Fisher-information row is governed by the active-height variance:

$$I_m(0) = 4\,\mathrm{Var}_{p^{(0)}}(h).$$

Accordingly, a nonzero r is a relative-information cost, not a bounded phase error on $\mathbb{T}$.

Definition 9286.11 (Thermodynamically stable zero packet tower)**.** *A zero certificate tower is thermodynamically stable when each sufficiently fine level carries a finite thermodynamic mode packet, the only legal closed dynamics are the unit phase rows*

$$e^{-ith_a} \in \mathbb{T},$$

there is no unledgered $\mathbb{R}_{>0}$ reservoir, the relative variance row is cofinal,

$$V_m \to \infty,$$

and the relative scale-defect energy is bounded:

$$E_m^{\mathrm{rel}}(r) \le B$$

for a finite B independent of m.

Definition 9286.12 (No unledgered selector row)**.** *The source row*

$$\mathsf{NoMaxwellDemonZeta}_\zeta$$

asserts that a zeta zero packet tower has no unledgered information selector capable of changing active-mode weights. Concretely, the row forbids an external selector of the form

$$a \longmapsto e^{-rh_a}$$

unless the corresponding real-positive scale ledger is displayed inside the tower. Compact phase rows and compact prime-register fibres may carry legal hidden data; an unrecorded $\mathbb{R}_{>0}$ selector may not be used to make an amplitude-biased packet appear to be unit-phase closed.

Definition 9286.13 (Radial compensation selector)**.** *Let*

$$s = \frac{1}{2} + r + it$$

and let $h_a > 0$ be an active height. A radial compensation selector for the centered channel

$$U_a(s) = e^{-rh_a} e^{-ith_a}$$

is a displayed family

$$C_a(s) \in \mathbb{R}_{>0}$$

such that

$$|C_a(s)U_a(s)| = 1.$$

For $r \neq 0$, this condition forces

$$C_a(s) = e^{rh_a}.$$

Thus a radial compensation selector is precisely a real-positive scale ledger indexed by the active heights.

Lemma 9286.14 (Unit boundary local channel test)**.** *Let*

$$s = \frac{1}{2} + r + it$$

and let $h_a > 0$. For the centered channel

$$U_a(s) = e^{-(s-1/2)h_a},$$

the unit-boundary condition

$$|U_a(s)| = 1$$

forces $r = 0$.

Proof. The centered channel decomposes as

$$U_a(s) = e^{-rh_a} e^{-ith_a}.$$

Hence

$$|U_a(s)| = e^{-rh_a}.$$

If this modulus is one, then $e^{-rh_a} = 1$. Taking real logarithms gives $-rh_a = 0$. Since $h_a > 0$, one has $r = 0$. □

Theorem 9286.15 (No-demon scale compensation)**.** *Assume a packet admits only compact phase rows, compact prime-register fibre rows, Zeckendorf ledger rows, and explicitly measured defect rows as legal gauges. If $r \neq 0$ and $h_a > 0$, no legal gauge supplies a radial compensation selector for*

$$U_a(s) = e^{-rh_a} e^{-ith_a}.$$

Proof. By Definition 9286.13, unit modulus after compensation requires

$$C_a(s) = e^{rh_a} \in \mathbb{R}_{>0}.$$

Compact phase rows change only the phase of $U_a(s)$ and do not change its modulus. Compact prime-register fibre rows and Zeckendorf ledger rows carry discrete or compact source information; they do not supply an unmeasured continuous real-positive scale factor. The remaining allowed case is an explicitly measured defect row, which records the radial defect rather than hiding it as a gauge. Hence no legal gauge supplies the displayed compensation factor. □

Principle 9286.16 (No free entropy lowering)**.** *Definition 9286.12 is the thermodynamic reading of ledger discipline. A selector that sorts, damps, or amplifies modes according to their heights has information cost. If that cost is not carried by the source, tail, prime-register, boundary, or scale-defect ledger, then the selector is outside the closed packet. Thus finite phase recurrence may provide unit-phase return candidates, but it cannot provide a free entropy-lowering operation that cancels the radial Boltzmann tilt of Definition 9286.9. The local obstruction is Definition 9286.15: the compensation factor is real-positive scale data, not phase or compact fibre data.*

Definition 9286.17 (Demon-free thermodynamic zero tower)**.** *A zero packet tower is demon-free thermodynamic when it is thermodynamically stable in the sense of Definition 9286.11 and carries the row*

$$\mathsf{NoMaxwellDemonZeta}_\zeta.$$

Equivalently, every information selector used by the tower is either a compact phase row, a compact prime-register fibre row, a Zeckendorf ledger row, or an explicitly measured defect row.

Definition 9286.18 (No-demon zeta certificate)**.** *The source row*

$$\mathsf{NoDemonZetaCertificate}_\zeta$$

asserts that every accepted non-trivial completed zeta zero is represented by a demon-free thermodynamic zero tower whose relative variance row is cofinal and whose scale-defect extraction is the displayed ledger of Definition 9286.8. This row is a coverage hypothesis: it packages the no-unledgered-selector condition together with the analytic tower needed to apply the thermodynamic no-scale theorem.

Theorem 9286.19 (Thermodynamic no-scale theorem)**.** *Let a thermodynamically stable zero packet tower have limit readout*

$$s_\infty = \frac{1}{2} + r + it.$$

Then $r = 0$.

Proof. By Definition 9286.8,

$$E_m^{\mathrm{rel}}(r) = r^2 V_m.$$

If $r \neq 0$, then $r^2 > 0$. Since $V_m \to \infty$, the displayed energy is unbounded:

$$E_m^{\mathrm{rel}}(r) \to \infty.$$

This contradicts the thermodynamic stability bound

$$E_m^{\mathrm{rel}}(r) \leq B.$$

Therefore $r = 0$. □

Corollary 9286.20 (Thermodynamic stability gives fixed-half readout)**.** *If every accepted non-trivial completed zeta zero is represented by a thermodynamically stable zero packet tower, then every such readout has real part* $1/2$.

Proof. Apply Definition 9286.19 to the tower representing the accepted zero. □

Corollary 9286.21 (Demon-free towers have fixed-half readout)**.** *If every accepted non-trivial completed zeta zero is represented by a demon-free thermodynamic zero tower, then every such readout has real part* $1/2$.

Proof. A demon-free thermodynamic zero tower is thermodynamically stable by Definition 9286.17. Apply Definition 9286.20. □

Corollary 9286.22 (No-demon certificates give fixed-half readouts)**.** *If*

$$\mathsf{NoDemonZetaCertificate}_\zeta$$

holds, then every accepted non-trivial completed zeta zero has real part $1/2$.

Proof. The source row of Definition 9286.18 supplies a demon-free thermodynamic zero tower for each accepted non-trivial completed zeta zero. Apply Definition 9286.21. □

Definition 9286.23 (No-demon internal golden zeta certificate)**.** *The source row*

$$\mathsf{NoDemonInternalGoldenZetaCertificate}_{\zeta,\varphi}$$

asserts that every accepted non-trivial completed zeta zero is presented by a cofinal golden certificate tower satisfying four compatible rows. The internal-ruler row is

$$\mathsf{InternalGoldenRulerZetaCertificate}_{\zeta,\varphi}$$

from Definition 9284.6. The same-island boundary row is

$$\mathsf{ZetaGoldenLeeYangBridge}$$

from Definition 9260.18. The information-thermodynamic row is

$$\mathsf{NoDemonZetaCertificate}_{\zeta}$$

from Definition 9286.18. The compatibility row requires that these three presentations use the same zero island, the same Zeckendorf address ledger, the same active-height refinement, and the same scale-defect extraction. Thus the row is a source package, not an additional analytic theorem: it records that the internal golden ruler, the Lee–Yang bridge, and the no-unledgered-selector discipline are present on one certificate tower.

Theorem 9286.24 (No-demon internal golden certificates supply thermodynamic certificates)**.** *If*

$$\mathsf{NoDemonInternalGoldenZetaCertificate}_{\zeta,\varphi}$$

holds, then

$$\mathsf{NoDemonZetaCertificate}_{\zeta}$$

holds.

Proof. Fix an accepted non-trivial completed zeta zero. The source row of Definition 9286.23 supplies a single cofinal golden certificate tower carrying the no-demon zeta row, with the scale-defect extraction and active-height refinement displayed on that same tower. Forgetting the internal-ruler and Lee–Yang bridge rows leaves exactly the demon-free thermodynamic tower required by Definition 9286.18. □

Corollary 9286.25 (No-demon internal golden certificates give fixed-half readout)**.** *If*

$$\mathsf{NoDemonInternalGoldenZetaCertificate}_{\zeta,\varphi}$$

holds, then every accepted non-trivial completed zeta zero has real part $1/2$*.*

Proof. Apply Definition 9286.24 and then Definition 9286.22. □

Principle 9286.26 (No-demon internal golden source load)**.** *The row*

$$\mathsf{NoDemonInternalGoldenZetaCertificate}_{\zeta,\varphi}$$

does not replace the analytic bridge obligations. Its internal golden ruler component still depends on the source package of Definition 9284.6; its boundary component still depends on the bridge source row of Definition 9260.18; and the bridge source row still carries the dominant-branch and Rouché homotopy obligations isolated in Definition 9284.8. The contribution of the no-demon row is orthogonal: once those same-island rows are supplied, no unledgered information selector may turn the radial factor

$$e^{-rh_a}$$

into a legal phase or compact prime-register gauge.

Principle 9286.27 (Thermodynamic wording preserves the proof boundary)**.** *The information-thermodynamic language identifies the off-half coordinate with entropy production, free-energy cost, or thermal tilt only after a canonical packet tower has been supplied. It does not construct that tower. The remaining source load is the same as in the golden route: strong active capture, same-island Lee–Yang boundary stability, cofinal relative-variance growth, no unledgered selector, no far-end scale reservoir, and zero-presentation completeness. Without those rows, the chapter gives a conditional no-scale implication rather than a zeta-zero coverage theorem.*

Chapter 9287

Copenhagen Boundary Route Audit

The Copenhagen no-scale route is a route-audit chapter, not a proof of RH. Its purpose is to keep three attractive reformulations on the honest side of the boundary. A completed-xi route may speak in Copenhagen tower energy, in diagonal Jensen hyperbolicity, or in Gomboc–Stieltjes positivity; none of these languages supplies the missing source-side positivity by being renamed. A closed proof still needs an independent global row that is cofinal in all visible cutoffs and is constructed before the zero divisor is read.

9287.1 Faithful Reformulations and Leverage

Principle 9287.1 (Faithful reformulations are not leverage)**.** *The following rows are faithful reformulations of the same completed-xi barrier unless an independent source-side positivity or uniform-estimate row is supplied:*

$$\mathsf{NoFarEndAbsorption}, \quad \mathsf{LiPositivity}, \quad \mathsf{WeilPositivity},$$
$$\mathsf{StieltjesHerglotzPositivity}, \quad \mathsf{FullJensenHyperbolicity}.$$

They may be excellent coordinates for the proof obligation. They do not decrease the proof obligation merely by translating it into a positivity, moment, hyperbolicity, or no-scale dialect.

The distinction is between a source row and a posterior readback. If a positive spectral measure is built from an already fixed-half divisor, it records what RH would make visible; it does not generate RH. If Li, Weil, Herglotz, Stieltjes, or Jensen data are proved from prime-side or operator-side rows without consuming the zero-placement conclusion, then the route has leverage. The audit question is therefore not whether these formulations are equivalent to the target. The question is whether their positive object is produced on the source side.

9287.2 The Three Projections of the Same Wall

Principle 9287.2 (Copenhagen boundary)**.** *At the completed-xi level, the Copenhagen energy tower, the diagonal-Jensen row, and the Gomboc row are three projections of the same global positivity wall. In schematic form the closed target is*

$$\begin{aligned} &\forall D, n\ \mathsf{JensenHyperbolic}(D, n) \\ \iff\quad &\exists \nu \geq 0\ \mathsf{StieltjesXi}(\nu) \\ \iff\quad &\forall \varphi\ \mathsf{WeilPSD}(\varphi). \end{aligned}$$

The Copenhagen energy expression is the no-scale face of the same wall: an off-half real scale r creates a positive factor r^2 multiplying a cofinal weight. Bounded energy can then rule out positive apartness of r^2, but only after the cofinal bounded-energy capture has been supplied.

The bridge just stated is a boundary principle, not a new theorem of this chapter. Its force is negative: it prevents a proof from treating one projection as a free source for another. A finite table of Jensen polynomials, a finite Weil Gram table, a finite spectral measure approximation, or a finite Copenhagen tower prefix sees only a bounded window. A global result must be cofinal in degree, truncation, support, height, and packet depth.

Principle 9287.3 (Finite visibility is incomplete)**.** *Finite phase data, finite Jensen radius data, finite spectral data, and finite positive-semidefinite tables do not determine the complete zero set. A finite window may be matched while additional zeros are inserted outside the visible radius by a Blaschke-type outer-window modification. Therefore finite success cannot be promoted to completed-xi closure unless there is a cofinal uniform theorem or an independent source-side construction that controls every later window.*

The consequence is operational. If the argument asks to check all degrees D, all Jensen levels n, all spectral cutoffs K, or all compact supports one by one, then it has become an enumeration protocol rather than a proof. A proof needs a row whose statement is uniform before the enumeration starts.

9287.3 Checked Finite Brick

Theorem 9287.4 (Bounded energy excludes positive scale apartness)**.** *Let r be a rational scale parameter, let $W : \mathbb{N} \to \mathbb{Q}$ be nonnegative and unbounded, and let B be a rational bound. If*

$$\forall n, \qquad r^2 W_n \le B,$$

then $0 < r^2$ is impossible.

Lean4✓ `BEDC.Derived.RHRoute.CopenhagenNoScale.boundedEnergy apart absurd`

Proof. The formal proof chooses, by unboundedness of W, an index n with

$$W_n > B/r^2$$

under the temporary assumption $0 < r^2$. Multiplication by the positive rational r^2 gives

$$B < r^2 W_n,$$

contradicting the assumed bound $r^2 W_n \le B$. This is the abstract Archimedean no-scale brick. It is checked in Lean with no additional axiom. □

Theorem 9287.5 (Finite jets underdetermine the polynomial)**.** *For every polynomial p over the rationals and every N, there is an infinite injective family of polynomials, each sharing the first $N+1$ coefficients, the degree-N jet, with p, obtained by adding a single high-degree monomial beyond the visible window. Hence a finite jet does not determine the polynomial.*

Lean4✓ `BEDC.Derived.RHRoute.FiniteVisibility.fvFamily injective`

↪ variant `BEDC.Derived.RHRoute.FiniteVisibility.fvFamily shares jet`

↪ variant `BEDC.Derived.RHRoute.FiniteVisibility.jet extendAt invisible`

Proof. This checked brick is the additive core of Definition 9287.3. The family changes only a coefficient beyond the visible degree-N jet, so the finite window still reads the same polynomial data while the resulting polynomials are pairwise different. The Blaschke-type outer-window modification in Definition 9287.3, where one keeps the disk window fixed while inserting a zero outside it, is the multiplicative and geometric version of the same boundary principle. That version remains a paper-level principle here and is not claimed as Lean checked. □

Theorem 9287.6 (Degree-two Turan certificate for Jensen positivity)**.** *Let $P(X) = a+2bX+cX^2$ be the degree-two Jensen polynomial with $c > 0$. Then P takes a nonpositive value for some rational X if and only if the Turan inequality $b^2 \geq ac$ holds. The proof completes the square,*

$$c\,P(X) = (cX + b)^2 + (ac - b^2),$$

and uses the witness $X = -b/c$.

Lean4✓ `BEDC.Derived.RHRoute.JensenTuranDegree2.jensen2 dips iff turan`

↪ variant `BEDC.Derived.RHRoute.JensenTuranDegree2.jensen2 complete square`

↪ variant `BEDC.Derived.RHRoute.JensenTuranDegree2.jensen2 nonpos forces turan`

Proof. This is an unconditional finite algebraic certificate over the rational carrier used by the Lean packet. It isolates the structural core of the Turan inequality

$$\gamma_{n+1}^2 \geq \gamma_n \gamma_{n+2}$$

for the Taylor coefficients that would feed the degree-two Jensen polynomial of ξ. The certificate is strictly weaker than RH: only Jensen hyperbolicity for all degrees d and all levels n is the Pólya–RH equivalent statement. It does not constrain ξ zeros without the coefficient-enclosure obligation; turning the rational certificate into a statement about ξ would also require certified rational enclosures for the relevant Taylor coefficients $\gamma_0, \gamma_1, \gamma_2$. □

Theorem 9287.7 (Cubic discriminant identity and the higher-Turan condition)**.** *Let $J(X) = g_0+3g_1X+3g_2X^2+g_3X^3$ be the degree-three Jensen polynomial and set $A = g_1^2-g_0g_2$, $B = g_2^2-g_1g_3$, $C = g_1g_2 - g_0g_3$, and $T_3 = 4AB - C^2$. Then the discriminant of J satisfies the pure ring identity*

$$\Delta(J) = 27\,T_3,$$

and, over the ordered rationals, $T_3 \geq 0$ if and only if $A \geq 0$, $B \geq 0$, and $C^2 \leq 4AB$.

Lean4✓ `BEDC.Derived.RHRoute.JensenTuranDegree3.cubic discriminant eq 27 T3`

↪ variant `BEDC.Derived.RHRoute.JensenTuranDegree3.T3 nonneg iff`

↪ variant `BEDC.Derived.RHRoute.JensenTuranDegree3.subdiscriminant A`

Proof. The Lean certificate is finite, unconditional, and algebraic over the rational carrier. The discriminant formula is a pure ring identity, and the subdiscriminant layer supplies the SOS-type identity

$$4A^3 = H_A^2 + g_0^2 T_3$$

with the corresponding B-side relation used by the ordered-rational corollary. Thus the nontrivial direction from $T_3 \geq 0$ to $A \geq 0$ and $B \geq 0$ is obtained without real closure.

This is not the real-root bridge. The implication

$$\Delta \geq 0 \iff J \text{ has three real roots}$$

is a separate real-closure or Sturm obligation, and is not part of this certificate. Over the ordered rationals, nonnegative discriminant does not even supply rational roots: $X^3 - 3X + 1$ has discriminant $81 > 0$ and no rational root. Nor is T_3 an unconditional coefficient-SOS polynomial: at $(g_0, g_1, g_2, g_3) = (1, 0, 0, 1)$ one has $T_3 = -1$.

The fixed degree-three certificate is strictly weaker than RH and is not RH-equivalent. Pólya equivalence requires Jensen hyperbolicity in every degree and every level. Applying this finite algebraic brick to ξ would still require certified enclosures for $\gamma_0, \gamma_1, \gamma_2, \gamma_3$, and the brick alone does not constrain the zero set of ξ. □

The conditional tower interface is also encoded:

Lean4 (definition) `BEDC.Derived.RHRoute.CopenhagenNoScale.CopenhagenTowerData`

and

Lean4 (definition) `BEDC.Derived.RHRoute.CopenhagenNoScale.CopenhagenCaptureObligation`

. The field

$$\mathsf{energy_bounded} : \exists B,\ \forall n,\ r^2 W_n \le B$$

inside CopenhagenTowerData is a capture input. It is not inhabited here for completed zeta zeros. The checked conditional theorem

$$\mathsf{CopenhagenTowerData}(r) \Longrightarrow \neg(0 < r^2)$$

is recorded as

Lean4 ✓ `BEDC.Derived.RHRoute.CopenhagenNoScale.copenhagenTower forces not apart`

. Thus the Lean contribution is a true finite brick plus a conditional handoff. It does not assert $r = 0$ by trichotomy, does not construct the xi tower, and does not prove RH.

9287.4 GAG One-Step Certificate

The GAG one-step packet is another finite brick in the same audit discipline. For a fixed degree parameter D, a certificate records rational root isolation intervals $[r_i^-, r_i^+]$, a rational level interval $[\lambda^-, \lambda^+]$, and rational signed-lobe intervals $[S_i^-, S_i^+]$. Its overshoot interval is computed by the finite recursion

$$m_0^- = \lambda^-, \qquad m_0^+ = \lambda^+,$$
$$m_{i+1}^- = S_i^- - m_i^+, \qquad m_{i+1}^+ = S_i^+ - m_i^-.$$

The checked part is entirely finite and rational: if the true level and true signed-lobe values lie in the recorded intervals, then the true overshoot lies in the computed overshoot interval at every step. If in addition $\lambda^- > 0$ and every lower overshoot bound m_i^- is positive through the finite range, the true overshoots are positive.

Lean4 ✓ `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.gag true overshoot in interval`

Lean4 ✓ `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.gag true overshoot pos`

Theorem 9287.8 (Finite GAG alternation soundness)**.** *For one fixed finite GAG certificate, positive true overshoot at a signed-lobe step forces the signed-lobe value to cross the level λ in the parity prescribed by the certificate: odd lobes are strictly above λ, and even lobes are strictly below λ.*

Lean4 ✓ `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.gag signed lobe alternation`

Proof. The formal statement is the rational one-step calculation. In odd parity the signed-lobe relation is $v = \lambda + m$; in even parity it is $\lambda = v + m$. The hypothesis $m > 0$ gives the corresponding strict inequality. The interval theorem above supplies the positive overshoot input when the recorded finite certificate passes its lower-bound tests. □

This theorem is not a root-count theorem. The root intervals are included in the packet because they are the intended handoff surface for a later constructive located-IVT bridge. The finite Lean theorem checks the rational alternation soundness only: interval inclusion gives positive overshoot, and positive overshoot gives strict alternation across the level.

Principle 9287.9 (Nonvacuous one-step witness)**.** *The certificate layer is not an empty shell. The encoded $D = 1$ instance uses $\lambda = 1/2$, a root ledger at -1, and the single signed-lobe value 1. It passes because $1/2 > 0$ and $1 - 1/2 = 1/2 > 0$.*
Lean4✓ `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.gagCert D1 passes`

This is the formal analogue of the hand calculation behind $I_1 = 1 + X$ and the one-step hyperbolic successor I_2 in the toy oracle case. It shows that the certificate format has an inhabited finite example. It does not by itself prove a general Jensen–Pólya step.

Principle 9287.10 (Located-IVT root count is an obligation)**.** *Passing finite GAG alternation is not yet the assertion that I_{D+1} has $D + 1$ located negative real roots. That conclusion needs a constructive located-IVT root-count bridge: monotone crossing, located root existence, and endpoint sign crossing have to be supplied as located-real witnesses. In Lean this bridge is deliberately recorded as an obligation structure, not as an inhabited theorem:*
Lean4 (definition) `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.GAGRootCountObligation`
. The conditional handoff
Lean4 (definition) `BEDC.Derived.RHRoute.GAGCertificate.GAGCert.gag onestep hyperbolic of rootcount`
only packages the conclusion after those obligation witnesses and the root-count function have already been supplied.

Thus the checked GAG contribution is exactly the finite rational certificate soundness plus a nonvacuous $D = 1$ witness. It does not prove the located-IVT bridge, does not count roots of I_{D+1}, and does not assert unconditional hyperbolicity for any completed Jensen–Pólya family.

Principle 9287.11 (Uniform GAG generator is the frontier)**.** *For each fixed D, the GAG format gives a finite certificate target: one may provide rational isolating intervals, a rational level interval, signed-lobe bounds, and finite lower-bound checks for the overshoot recursion. Closing all degrees would require a* UniformGAGCertificateGenerator *producing the certificates uniformly in D and proving*

$$\forall D\ \forall i \le D, \qquad m_i^-(D) > 0.$$

The natural decomposition is a finite-part lower-bound proof, a tail upper-bound proof such as a prolate or Hilbert–Schmidt estimate, and a stability proof for the overshoot recursion. If such a uniform generator were attached to the full Jensen–Pólya family, it would cross the RH *boundary. It is therefore recorded here as an* RH*-level frontier, not as a theorem and not as evidence that the all-D problem is already closed.*

9287.5 Gomboc Conditional Comparison

The Gomboc route has the same logical shape. Its encoded xi-facing container
Lean4 (definition) `BEDC.Derived.RHRoute.AnalyticGombocForXi.AnalyticGombocForXi`
contains Pick, Herglotz, Stieltjes, equivalence, handoff, and no-hidden-ballast data. The handoff theorem
Lean4✓ `BEDC.Derived.RHRoute.AnalyticGombocForXi.rh via analytic gomboc`
is conditional on such a container. The source-side spectral positivity field

Lean4 (definition) `BEDC.Derived.RHRoute.AnalyticGombocForXi.StieltjesPositiveSpectral.positive spectral measure`

is part of the supplied structure, not a theorem generated in this paper chapter.

This is the intended comparison. Copenhagen says that bounded cofinal energy plus unbounded weights forbid positive scale apartness. Gomboc says that a supplied Pick–Herglotz–Stieltjes positive package gives the constructive RH readout. Both routes are useful boundary normal forms. Neither route claims that the required xi-facing package has been constructed.

9287.6 Bad Examples as First-Class Boundary Data

Principle 9287.12 (Two-term resonance is off-line)**.** *For two exponential terms with distinct heights $h_1 \neq h_2$ and coefficients $a \neq b$, phase cancellation occurs at a shifted line*

$$\delta = \frac{\log(b/a)}{h_2 - h_1} \neq 0$$

in general. Verdict: phase cancellation alone is not enough; exact balance or a Lee–Yang stability row is required before a resonance may be used as fixed-half evidence.

Principle 9287.13 (Positive kernels do not imply Laguerre–Polya)**.** *Let*

$$\mu = \delta_{-1} + a\delta_0 + \delta_1 \qquad (a > 2).$$

The Fourier transform is

$$\widehat{\mu}(z) = a + 2\cos z.$$

Its zeros need not be real. Verdict: positivity of a kernel or measure row does not by itself place an entire function in the Laguerre–Polya class; a separate real-rootedness-preserving mechanism is needed.

Principle 9287.14 (Mellin projection need not preserve Lee–Yang)**.** *The finite Mellin projection*

$$P_2(s) = 1 - 2^{-s}$$

has zeros on $\mathrm{Re}(s) = 0$*, not on the line* $\mathrm{Re}(s) = 1/2$*. Verdict: a projection that is natural for Mellin or prime-window bookkeeping need not preserve the Lee–Yang unit circle or the fixed-half target.*

Principle 9287.15 (All-cutoff dyadic LP is not a reliable premise)**.** *The all-K dyadic Laguerre–Polya premise is not a stable bridge. Low corrected Jensen levels for small cutoffs such as $K = 2, 3$ already exhibit non-hyperbolic behaviour in the intended finite tests. Verdict: a cutoff-by-cutoff Laguerre–Polya route should be treated as a refuted or unreliable premise unless it supplies a separate uniform preservation theorem.*

Principle 9287.16 (Finite visibility cannot certify the tail)**.** *Any finite table of phases, Jensen coefficients, spectral atoms, Gram entries, or Copenhagen tower levels leaves an outer region unseen. Additional zeros or defects may be placed beyond that region without changing the visible data. Verdict: finite visibility is evidence about a window, not a certificate for the completed zero set.*

Principle 9287.17 (Boundary verdict)**.** *The honest output of the Copenhagen audit is the following separation. The checked no-scale brick is real: bounded cofinal energy contradicts positive rational scale apartness. The capture rows that would attach that brick to completed zeta zeros are not inhabited here. The diagonal Jensen, Weil, Stieltjes–Herglotz, and Gomboc formulations describe the same global wall. A proof must add source-side positivity or a cofinal uniform theorem; otherwise the route remains a boundary normal form and does not become a proof of* RH.

Chapter 9288

Unitary Scale Closure Route for RH

chapter 9255 isolates a pre-RH zero-generation route, and chapter 9243 identifies the critical line with unit-amplitude prime-channel readouts. The present chapter records the scale-normalized reduction suggested by the chapter 9250 carrier and the no-independent-real-scale ledger in the ZetaCosmosClosure route. It is a conditional reduction, not a proof of RH: the mathematical load is the obligation that a certified critical-strip zero packet has no independent real scale ledger and therefore lands on a unitary scale surface.

Definition 9288.1 (Golden scale eigenvalue)**.** *Let*

$$w := s - \frac{1}{2}, \qquad \varphi := \frac{1+\sqrt{5}}{2}, \qquad q := \varphi^w.$$

The golden scale eigenvalue of s is the centered scale readout q. It is located only as a scale-coordinate packet over the centered critical-strip route; it is not an independent zero generator, a branch choice for ζ, or a replacement for the zeta zero ledger.

Principle 9288.2 (Unitary scale closure reduction)**.** *For a critical-strip zero packet s, write $w = s - 1/2$ and $q = \varphi^w$. Since $\varphi > 1$,*

$$|q| = |\varphi^w| = \varphi^{\Re(w)}.$$

Therefore

$$\Re(s) = \frac{1}{2} \iff \Re(w) = 0 \iff |q| = 1.$$

The unitary scale closure route is the conditional reduction

$$\mathsf{NoIndependentRealScaleLedger}(s) \Longrightarrow \mathsf{UnitaryScale}(q) \Longrightarrow |q| = 1 \Longrightarrow \mathsf{OnCritLine}(s).$$

If every certified non-trivial zeta zero in the critical strip satisfies the displayed no-scale-to-unitary obligation, then the BEDC-RH witness function of Definition 9333.1 is inhabited.

Lean4 (statement) `BEDC.Derived.RHRoute.ZetaUnitaryScaleClosure.ZetaUnitaryScaleClosure`
Lean4 (statement) `BEDC.Derived.RHRoute.ZetaUnitaryScaleClosure.rh via unitary scale closure`

Principle 9288.3 (Argument-principle bypass is only a route choice)**.** *This route does not count zeros by contour integration, does not use the argument principle as a zero-enumeration engine, and does not require a classical winding-number decision over an unbounded contour family. It instead asks for a scale-closure certificate on already certified zero packets. The bypass is methodological: it avoids the constructive obstruction of turning global argument variation into a total zero ledger, but it does not remove the need to prove the scale-closure obligation.*

Remark 9288.4 (Connes–Consani half-density reading)**.** *The centered coordinate $w = s - 1/2$ matches the half-density normalization in scale-action readings of the zeta symmetry: the real half-shift is absorbed into the carrier before the residual scale action is inspected. In this language, the nontrivial part of the scale action is the phase row φ^{it} when $w = it$. The BEDC route uses that analogy only as a scale-normalized readout of the ZetaCosmosClosure ledger; it does not import an external spectral triple, trace formula, or classical cohomology theorem as a proof object.*

Principle 9288.5 (Symmetry alone is not enough)**.** *The functional equation, conjugation symmetry, and the mirror fixed surface do not by themselves prove* RH*. An even real polynomial can have a symmetric zero set with off-axis real scale, so symmetry alone cannot force all zeros onto a unit circle after centering. The no-independent-real scale ledger is the genuine extra input in this route. Without the obligation*

$$\mathsf{NoIndependentRealScaleLedger} \Longrightarrow \mathsf{UnitaryScale},$$

the identity $|\varphi^w| = \varphi^{\Re(w)}$ is only a readout of the critical line, not a proof that zero packets land there.

Definition 9288.6 (Gömböc-style homogeneous container row)**.** *A Gömböc-style homogeneous container row for the completed zeta carrier is a source-side packet*

$$\mathsf{ZetaGombocContainer}_\zeta$$

with three displayed fields. The homogeneity field says that the zero packet carries no hidden mass, weight, or density parameter beyond the source, prime, phase, tail, and defect ledgers already present in the carrier. In the centered channel this forbids an unrecorded $\mathbb{R}_{>0}$ factor acting as a radial counterweight. The equilibrium field says that the legal balance surface of the container is the unit scale surface

$$|q| = 1.$$

The source-field says that the balance is produced by the internal container rows themselves, not by selecting a zero and then attaching a compensating scale ledger to it. In particular, the row is global: it is a constraint on the completed zeta carrier as a shape, not a family of local supports built separately around individual zero packets.

Principle 9288.7 (Analytic Gömböc reading)**.** *The Gömböc-style reading is a geometric discipline for the no-scale obligation in Definition 9288.2. A homogeneous convex body with a single stable balance point represents, at the level of analogy, a container whose righting behaviour is carried by shape rather than by hidden weights. The zeta translation is the row*

$$\mathsf{ZetaGombocContainer}_\zeta \Longrightarrow \mathsf{NoIndependentRealScaleLedger} \Longrightarrow \mathsf{UnitaryScale}(q).$$

Thus an off-half readout is not interpreted as a second equilibrium of the same homogeneous container. It is interpreted as a demand for a radial counterweight, namely an extra real-positive scale ledger. The row is still conditional: it names the geometric content of the no-scale-to-unitary implication, but it does not prove that every zeta zero packet carries such a container.

Definition 9288.8 (Analytic Gömböc source shape)**.** *An analytic Gömböc source shape for ξ is a source-side global presentation of the completed zeta carrier in one of the following positive forms:*

$$K_\xi(z, w) = \langle \Phi_\xi(z), \Phi_\xi(w) \rangle_{\mathcal{H}}$$

as in Definition 9271.8, the positive real row

$$\operatorname{Re}\frac{\xi'(s)}{\xi(s)} \geq 0 \qquad \left(\operatorname{Re}(s) > \frac{1}{2}\right)$$

of Definition 9270.2, or a positive spectral representation of the same log-derivative as in Definition 9270.6. The representation must be constructed before zero locations are consumed. A representation assembled around the already-read zero divisor is not an analytic Gömböc source shape.

Principle 9288.9 (Source-shape proof form)**.** *The Gömböc row changes the proof burden from local repair to global shape. A local container placed around a candidate zero would be a supporting device for that readout; it would not certify that the completed carrier itself is homogeneous. By contrast, a source-side Pick factorization, Herglotz positive-real row, or positive spectral measure is a shape theorem for ξ as a whole. In that form an off-half zero cannot be hidden as a local exception: by Definition 9271.13 and Definition 9270.5, it becomes a finite negative Pick witness or a negative scale-flow witness against the same global positive structure.*

Theorem 9288.10 (Gömböc container gives the unitary scale reduction)**.** *Assume that every certified non-trivial zeta zero in the critical strip carries* $\mathsf{ZetaGombocContainer}_\zeta$ *and that the container row supplies the implication*

$$\mathsf{ZetaGombocContainer}_\zeta \to \mathsf{UnitaryScale}(q)$$

for the centered golden scale readout $q = \varphi^{s-1/2}$. Then every certified non-trivial zeta zero in the critical strip satisfies the critical-line readout.

Proof. Fix a certified non-trivial zeta zero s in the critical strip and set

$$q = \varphi^{s-1/2}.$$

The assumed container row gives $\mathsf{UnitaryScale}(q)$, hence $|q| = 1$. By Definition 9288.2, the identity

$$|q| = \varphi^{\Re(s-1/2)}$$

then forces $\Re(s) = 1/2$. This is precisely the ordinary $\mathsf{OnCritLine}(s)$ readout. □

Definition 9288.11 (Route obligations)**.** *The unitary scale closure route has five obligations.*

1. *The centered coordinate $w = s - 1/2$ must be carried by the same critical-strip zero packet as s.*
2. *The golden scale eigenvalue $q = \varphi^w$ must be a located scale-readout with the modulus identity*

$$|q| = \varphi^{\Re(w)}.$$

3. *The ZetaCosmosClosure no-independent-real-scale ledger must imply unitarity of the scale readout for every certified non-trivial zero packet.*
4. *The optional Gömböc-style container row, when used, must be source-side and homogeneous: it may package the no-scale-to-unitary implication through a Pick, Herglotz, or positive spectral source shape, but it may not be built by adding a compensating radial ledger after a zero has been selected.*

5. *The unitary readout must transport back to the ordinary* OnCritLine(s) *witness used by the BEDC-RH statement.*

Closure status for NameCert$_{ZetaUnitaryScaleClosureUp}$.

Origin: ai

Constructive story: The route is generated from a certified zeta zero packet, the centered coordinate $w = s - 1/2$, the golden scale readout $q = \varphi^w$, the no-independent-real-scale ledger, the optional Gömböc-style homogeneous container row, the unitary scale obligation, and transport back to the ordinary critical-line predicate.

Theory closure: TheoryGate(NameCert$_{ZetaUnitaryScaleClosureUp}$, obligation)

Scope closed: The chapter states the conditional unitary-scale reduction, the homogeneous-container reading of the no-scale obligation, the source-shape proof form, and the exact proof obligations; it does not discharge the no-scale-to-unitary implication.

Formal verification: FormalStatus(NameCert$_{ZetaUnitaryScaleClosureUp}$, formalTarget)

Bridge status: none

Not claimed: This route does not prove RH, does not count zeros, does not construct a Gömböc-style container for all zeta zero packets, and does not derive the no-independent-real-scale ledger from the functional equation or symmetry alone.

Upgrade path: Scoped closure requires a checked modulus identity for $|\varphi^w| = \varphi^{\Re(w)}$, a no-independent-real-scale to unitary-scale theorem for certified zeta zero packets, an optional source-side homogeneous-container theorem expressed through Pick, Herglotz, or positive spectral structure, and transport from unitary scale back to OnCritLine.

Chapter 9289

Phase, Orthogonal Readout, and Prime Register Residue

section 9413.6 records the Pythagorean carrier as a shape-scale decomposition, and section 9413.17 reads the same discipline through circle phase, Fourier phase, Laplace damping, and zeta. The present chapter isolates the RH-facing readout principle behind that ladder. A unit circle row is a visible phase surface; a right-triangle row is the orthogonal coordinate read of that phase; a prime register is the residual source ledger not absorbed by the visible phase projection. The point is structural. It is not a physical Heisenberg inequality and not a proof of RH.

9289.1 Unit Phase and Orthogonal Readout

Definition 9289.1 (Unit phase readout)**.** *A unit phase row is a circle-valued coordinate*

$$u(\theta) = e^{i\theta} \in \mathbb{T}.$$

Its orthogonal coordinate readout is

$$\rho_{\perp}(u(\theta)) := (\cos\theta, \sin\theta),$$

with invariant

$$\cos^2\theta + \sin^2\theta = 1.$$

The coordinate pair is therefore a finite visible vector event on the unit-circle surface, not a duplicate of every source row that produced the phase.

Principle 9289.2 (Triangle readout preserves norm and hides provenance)**.** *The passage*

$$e^{i\theta} \mapsto (\cos\theta, \sin\theta)$$

preserves the quadratic norm row and exposes an orthogonal two-coordinate event. It does not by itself expose winding history, cover data, branch choice, prime-local provenance, or any hidden fiber attached to the phase row. Those rows must be carried by separate ledgers if a later consumer is allowed to recover them.

Definition 9289.3 (Phase-triangle event)**.** *A phase-triangle event is a readout packet*

$$T_\theta := (x, y, h_\theta), \qquad x = \cos\theta, \quad y = \sin\theta,$$

where

$$h_\theta : x^2 + y^2 = 1.$$

It is the visible orthogonal slice of the phase row $e^{i\theta}$. In BEDC terms, it is an output surface with a quadratic invariant, not the full source carrier.

9289.2 Prime Register Residue

Definition 9289.4 (Solenoid phase register)**.** *In the notation of Definition 9413.779, the solenoid phase-register readout is the visible projection*

$$\Sigma_S \to \mathbb{T}$$

with fiber $K_S = \ker(\pi_S)$. For finite S, this fiber is the prime-register residue behind the circle phase: it records the prime-adic source distinctions that are not visible in the circle coordinate.

Principle 9289.5 (Prime registers are not phase coordinates)**.** *The projection $\Sigma_S \to \mathbb{T}$ licenses a visible phase readout, but it does not erase the source ledger. Two source rows in Σ_S may share the same visible phase while differing by an element of K_S. Thus a solenoid is a phase quotient with a profinite prime fiber, not a bare circle and not an ambient real-scale container.*

Definition 9289.6 (Projection residue)**.** *For a displayed readout*

$$\pi : E \to V,$$

the residue over a visible row $v : V$ is the fiber

$$\mathrm{Res}_\pi(v) := \sum_{e:E} (\pi(e) \sim_V v).$$

A visible coordinate is lossless only when the packet also supplies a certified section or classifier-faithful reconstruction row for this residue. Otherwise the readout has exposed a surface while retaining the unread distinctions as fiber or ledger data.

Principle 9289.7 (Projection-residue uncertainty)**.** *A single low-dimensional visible readout cannot, without additional ledger rows, be simultaneously the full continuous phase, the full prime-register residue, and an independent radial scale coordinate. A readout may expose one surface and preserve a stated invariant, but every unexposed source distinction must remain in a fiber, a transport ledger, or an explicit obstruction row. This is the BEDC projection-residue form of uncertainty.*

9289.3 Constructive Source Chain

Definition 9289.8 (Distinction-to-phase source chain)**.** *The source-side chain used by this chapter is the ledgered route*

$$\mathsf{BMark} \to \mathsf{BHist} \to \mathsf{Cont} \to \mathsf{OrthogonalReadout} \to \mathbb{T} \to \mathsf{PrimeLedger} \to \Sigma_S.$$

The first three entries are the kernel spine of chapter 9351. The orthogonal and unit-phase entries are the readouts of Definition 9289.1 and Definition 9289.3. The prime-ledger entry is read through the complementary normal forms of Definition 9414.204. The final entry is the solenoid phase register of Definition 9289.4.

Principle 9289.9 (Observation time is continuation growth)**.** *At this level, time is not an external real parameter added before the source is generated. A finite observation ledger is a generated history row, and a later ledger is read through displayed* Cont *growth. Thus the time arrow relevant to the phase-prime readout is append-only ledger growth over* BHist*, not a hidden background clock.*

Definition 9289.10 (Zeckendorf–prime ledger interface)**.** *A Zeckendorf–prime ledger interface for a finite binary history is a displayed sequence of rows:*

$$\mathsf{BHist} \to \mathsf{NoAdjacentWord} \to \mathsf{ZeckendorfValue} \to \mathsf{PrimeExponentLedger}.$$

The no-adjacent word is governed by the recursive additive normal form of Definition 9414.203; the prime exponent ledger is governed by the multiplicative atom normal form of Definition 9414.202. The interface is legal only when the block-code row, no-adjacent normal-form row, value-readback row, and prime-factorization row are all displayed.

Theorem 9289.11 (Finite binary histories admit a prime-ledger readout)**.** *Every finite binary history admitted by the kernel spine has a Zeckendorf–prime ledger readout after a displayed no-adjacent block-code row is chosen.*

Proof. By Definition 9351.16, a generated BHist row has a finite binary word readout. Apply the displayed block code

$$\mathsf{b0} \mapsto 10, \qquad \mathsf{b1} \mapsto 100.$$

Concatenating these blocks never creates adjacent 1 entries, so the result is a finite no-adjacent word. The Zeckendorf normal-form row reads that word as a recursive additive address, and the prime-factorization row sends the displayed address positions to prime exponents. Uniqueness is not supplied by the block code alone; it is the joint effect of the displayed block-code inverse, the no-adjacent normal-form classifier, and the multiplicative prime-factorization classifier. □

Principle 9289.12 (Constructive chain is a proof obligation route)**.** *The chain from raw distinction to solenoid phase is a route for locating the source of a zeta readout. It does not assert that the completed zeta carrier has already been generated by those rows. It identifies the rows that a later discharge must expose: finite histories, continuation growth, orthogonal readout, phase projection, recursive additive normalization, multiplicative prime accounting, solenoid fiber compatibility, and zero-ledger transport.*

9289.4 Centered Zeta Modes

Definition 9289.13 (Centered phase-scale mode)**.** *Let h_a be a displayed positive frequency or prime-ledger height, and write*

$$s = \frac{1}{2} + r + it, \qquad r, t \in \mathsf{Real}^{\uparrow}.$$

The centered zeta mode attached to a is

$$U_a(s) := e^{-(s-\frac{1}{2})h_a} = e^{-rh_a} e^{-ith_a}.$$

The factor e^{-ith_a} is the unit-circle phase read. The factor e^{-rh_a} is a positive real radial-scale read. The source of h_a is the arithmetic or prime-ledger row that determines which frequency has been inspected.

Principle 9289.14 (Unit phase closure excludes radial scale)**.** *If a zero-relevant zeta packet is licensed as a pure unit phase closure, then every centered mode must have no independent positive real scale factor. In the notation of Definition 9289.13, that means the mode is read as*

$$U_a(s) = e^{-ith_a} \in \mathbb{T}$$

rather than as

$$e^{-rh_a} e^{-ith_a}$$

with an extra radial register. This is the same proof role as the no-independent-real-scale obligation in chapter 9288.

Definition 9289.15 (Orthogonal scale-phase energy readout)**.** *For a finite active packet A_m, define*

$$v_a := (-rh_a, -th_a), \qquad W_m := \sum_{a \in A_m} h_a^2.$$

The orthogonal energy readout is

$$|v_a|^2 = r^2 h_a^2 + t^2 h_a^2,$$

and its scale component is

$$E_m^{\text{scale}} := r^2 W_m.$$

Definition 9289.16 (Scale-defect positivity packet)**.** *A scale-defect positivity packet for the active family (A_m) consists of a displayed defect readout $D_m^{\text{scale}}(s)$ and a bound row B such that*

$$D_m^{\text{scale}}(s) = r^2 W_m + \varepsilon_m, \qquad \varepsilon_m = o(W_m),$$

and

$$D_m^{\text{scale}}(s) \leq B$$

for every displayed window m. The notation records an asymptotic obligation inside the chosen analytic interface; it is not an automatic consequence of the centered-mode definition.

Theorem 9289.17 (Cofinal scale energy is incompatible with pure phase closure)**.** *Assume a cofinal family of active packets satisfies $W_m \to \infty$. If $r \# 0$, then*

$$E_m^{\text{scale}} \to \infty.$$

Consequently any zero-relevant closure theorem that supplies pure unit phase closure together with a bounded or absent radial-scale ledger must force $r = 0$.

Proof. By Definition 9289.15,

$$E_m^{\text{scale}} = r^2 W_m.$$

If $r \# 0$, then $r^2 > 0$, so the cofinal divergence of W_m gives $E_m^{\text{scale}} \to \infty$. A pure unit phase closure has no independent positive real scale row. Therefore a bounded or absent scale-ledger condition is incompatible with $r \# 0$, and the centered coordinate must satisfy $r = 0$. □

Theorem 9289.18 (Positive scale-defect criterion)**.** *Assume $W_m \to \infty$ and a scale-defect positivity packet in the sense of Definition 9289.16. Then r is not apart from 0. In any located real readback that collapses absence of apartness to equality at the stated resolution, the centered coordinate satisfies $r = 0$.*

Proof. Assume $r\#0$. Then $r^2 > 0$. Since $\varepsilon_m = o(W_m)$, there is a cofinal tail on which

$$|\varepsilon_m| \le \frac{r^2}{2} W_m.$$

Hence on that tail

$$D_m^{\mathrm{scale}}(s) \ge \frac{r^2}{2} W_m.$$

Because $W_m \to \infty$, the right side is unbounded. This contradicts the displayed bound row $D_m^{\mathrm{scale}}(s) \le B$. Therefore the apartness row $r\#0$ is impossible. A located real readback then gives the displayed equality row $r = 0$ at the resolution licensed by the packet. □

9289.5 RH-Facing Proof Load

Principle 9289.19 (RH as phase-prime-triangle closure)**.** *Under the recovered-carrier reading of chapter 9250, the RH-facing geometric assertion is that every non-trivial completed zeta zero is a legal phase-prime-triangle closure: the visible component is a unit-circle phase readout, the orthogonal coordinate event preserves the quadratic invariant, the prime source remains in the displayed ledger or solenoid fiber, and no extra radial scale register is needed to close the zero packet. In that form the conclusion is*

$$r = 0, \qquad \Re(s) = \frac{1}{2}.$$

Definition 9289.20 (BEDC-legal zeta phase object)**.** *The statement*

$$\mathsf{ZetaIsBEDCLegalPhaseObject}$$

names a certificate-bearing existence obligation in the sense of Definition 9352.18. It requires the completed zeta zero surface to supply the source-chain rows of Definition 9289.8, the centered-mode rows of Definition 9289.13, a cofinal active family, a scale-defect positivity packet, continuation and completion transport, and zero-ledger soundness, with no unledgered radial scale escape. It is an obligation row, not an axiom and not a hidden definition of RH.

Definition 9289.21 (RH problem object)**.** *The RH problem object is the certificate request*

$$\mathsf{RH}^{\uparrow} := (\mathsf{CompletedZetaSource}, \mathsf{ZeroLedger}, \mathsf{ScaleDefect}, \mathsf{PositiveWitness}, \mathsf{ClosureTransport}).$$

The source row names the completed zeta carrier; the zero ledger names the candidate non-trivial zero packet; the scale-defect row records $r = \Re(s) - 1/2$ *in the centered coordinate; the positive witness row names the analytic positivity mechanism; and the closure-transport row records how the positivity witness descends to the zero-ledger classifier. The object is admitted only through a construction certificate for these rows.*

Theorem 9289.22 (Legal zeta phase objects are conditionally RH-sufficient)**.** *If every completed non-trivial zeta zero packet carries a* $\mathsf{ZetaIsBEDCLegalPhaseObject}$ *witness, then the BEDC RH section target is inhabited.*

Proof. Let a completed non-trivial zero packet have centered coordinate

$$s = \frac{1}{2} + r + it.$$

The legal phase-object witness supplies the cofinal active family and the scale-defect positivity packet. By Definition 9289.18, the radial coordinate has no apartness from zero, and the located readback carried by the packet gives $r = 0$. Therefore the same zero packet lies on the critical line,

$$\Re(s) = \frac{1}{2}.$$

Transporting this row through the zero-ledger classifier gives the required value of the RH section for the chosen zero packet. Since the argument used only the arbitrary zero packet and its legal phase-object witness, it yields the conditional section. □

Principle 9289.23 (Pick–Herglotz positivity as analytic discharge)**.** *One analytic route for discharging Definition 9289.20 is a certificate-bearing positive-kernel factorization for the completed zeta carrier. With*

$$f(z) := 2\frac{\xi'}{\xi}\left(\frac{1}{1-z}\right),$$

the target kernel is

$$K_f(z,w) := \frac{f(z) + \overline{f(w)}}{1 - z\overline{w}}.$$

A construction

$$K_f(z,w) = \langle \Phi(z), \Phi(w) \rangle$$

would supply the orthogonal information geometry required by the phase-prime-triangle reading only if the Hilbert carrier and the map Φ themselves enter through source-side construction certificates. This chapter records that target as the analytic proof load; it does not claim the factorization.

Definition 9289.24 (Source Hilbert positivity object)**.** *The source Hilbert positivity object for the completed zeta carrier is the certificate request*

$$\mathsf{SourceHilbert}^{\uparrow}(\xi) := \sum_{\mathcal{H}} \sum_{\Phi:\mathbb{D}\to\mathcal{H}} \left(\prod_{z,w\in\mathbb{D}} K_f(z,w) = \langle \Phi(z), \Phi(w) \rangle \right) \times \mathsf{SourceCert}(\mathcal{H}, \Phi).$$

The final row asserts that $\mathcal{H}$ and Φ are constructed from the prime side, Gamma side, continuation rows, completion rows, and BEDC ledger, not from a posteriori knowledge of zero locations.

Principle 9289.25 (Source Hilbert positivity is the remaining discharge)**.** *If Definition 9289.24 is constructed, it supplies the positive witness component of Definition 9289.21. The remaining transport is then the standard Herglotz-to-zero-exclusion route: positivity of the Pick kernel gives nonnegative real part for the logarithmic derivative on the right-half critical domain, while an off-line zero would force a negative local real-part read near its pole. This principle records the analytic discharge route; the source Hilbert construction remains the open certificate.*

Principle 9289.26 (Status of the readout principle)**.** *The assertion that the unit circle is the phase surface and the right triangle is its orthogonal readout is valid only in this ledgered sense. The circle exposes phase, the triangle exposes a quadratic coordinate event, the solenoid fiber retains prime-register residue, and the centered zeta mode separates unit phase from radial scale. The remaining mathematics is the construction showing that completed zeta zero packets are confined to that legal phase-prime closure surface.*

Chapter 9290

Circle-Triangle Generation and Hidden Scale Refusal

chapter 9289 separates unit phase, orthogonal readout, and prime-register residue. The present chapter records the corresponding generation discipline at the level of zero-relevant observations. A legal phase object is not a bare complex number. It is a certificate-bearing packet whose visible component is a unit-circle phase or an orthogonal readout of that phase, and whose unread distinctions are retained by an explicit ledger. The radial scale row is therefore not free background structure. If it participates in a closure, it must be displayed as a ledgered residual; if it is not displayed, it is not available to the object.

9290.1 Generated Time and Orthogonal Readout

Definition 9290.1 (Circle-triangle generation packet)**.** *A circle-triangle generation packet is a certificate row*

$$\mathcal{G} := (B, \mathsf{Hist}_{\mathcal{G}}, \mathsf{Cont}_{\mathcal{G}}, \mathbb{T}_{\mathcal{G}}, \rho_{\perp,\mathcal{G}}, \Lambda_{\mathcal{G}})$$

with the following data. The distinction row contains two apart marks

$$B_0 \# B_1.$$

The history row $\mathsf{Hist}_{\mathcal{G}}$ *is generated by finite append of those marks, and* $\mathsf{Cont}_{\mathcal{G}}$ *records continuation of such histories. The phase row is a unit-circle readout*

$$u(\theta) = e^{i\theta} \in \mathbb{T}_{\mathcal{G}},$$

and the orthogonal observation row is

$$\rho_{\perp,\mathcal{G}}(u(\theta)) = (\cos\theta, \sin\theta).$$

The ledger $\Lambda_{\mathcal{G}}$ *records distinctions not visible in the circle or triangle readout.*

Principle 9290.2 (Observation time is append order)**.** *The time row in a circle-triangle generation packet is the order in which finite marks are appended and continued. A later observation is a longer or transported history row, not an independent real-time coordinate inserted before the finite kernel. Thus every residual that must be kept for later recovery has a ledger position in the append-and-continuation surface.*

Definition 9290.3 (Orthogonal observation event)**.** *For a phase row $u(\theta)$ in $\mathcal{G}$, the associated orthogonal observation event is the finite vector packet*

$$O_\theta := (\cos\theta, \sin\theta, h_\theta),$$

where

$$h_\theta : \cos^2\theta + \sin^2\theta = 1.$$

It exposes the quadratic invariant of the unit phase. It does not expose the complete source history, winding route, prime provenance, or solenoid fiber unless these are separately present in $\Lambda_{\mathcal{G}}$.

Principle 9290.4 (Circle and triangle are readout surfaces)**.** *The circle row is the continuous unit-phase surface. The triangle row is one orthogonal coordinate read of that phase. Neither row is licensed to erase source distinctions. Every unread distinction must be retained as a fiber, a ledger entry, a transport witness, or an obstruction row.*

9290.2 Prime Registers and Addressed Residue

Definition 9290.5 (Zeta prime-register readout)**.** *For a positive integer n, the prime-register readout is the finite support vector*

$$\upsilon(n) := (v_p(n))_p, \qquad n = \prod_p p^{v_p(n)}.$$

The corresponding frequency row is

$$\log n = \sum_p v_p(n) \log p.$$

Thus the oscillatory zeta mode

$$e^{-it\log n}$$

has visible phase behavior, while the finite exponent vector $\upsilon(n)$ records the irreducible arithmetic source ledger behind that behavior.

Principle 9290.6 (Finite addressability is not free hiding)**.** *Definition 9289.10 supplies an internal route from a finite binary history to a Zeckendorf address and then to a prime-exponent ledger. This addressability licenses finite residual data only when the relevant address row is displayed. It does not license a consumer to use residual scale, branch, phase-source, or prime-fiber information while omitting the ledger that carries it.*

Definition 9290.7 (Prime-classifier container)**.** *A prime-classifier container $\mathcal{C}$ consists of a finite prime-register window $P_{\mathcal{C}}$, a unit or normalized phase row $U_{\mathcal{C}}$, a triangular norm row $N_{\mathcal{C}}$, a boundary row $B_{\mathcal{C}}$, and a type read*

$$\tau_{\mathcal{C}} : P_{\mathcal{C}} \to \{\mathsf{split}_{\mathcal{C}}, \mathsf{inert}_{\mathcal{C}}, \mathsf{ramified}_{\mathcal{C}}\}.$$

The type names are relative to the container. A split prime is a register row whose container supplies a factor pair and hence a possible unit-phase resource. An inert prime is a register row whose unpaired occurrence contributes to the triangular norm boundary. A ramified prime is a branch or degeneracy row whose contribution must be recorded as boundary data rather than erased into phase.

Principle 9290.8 (Prime type is container-relative)**.** *A rational prime is not an undifferentiated particle in the RH-facing ledger. It acquires its proof role only after a container $\mathcal{C}$ has supplied the classifier $\tau_{\mathcal{C}}$ of Definition 9290.7. In the Gaussian two-square container this read is the familiar split, inert, and ramified trichotomy. In a Frobenius, character, solenoid, or finite-window Euler container the same discipline says that the prime row must be read through the displayed register, phase, and tower rows before it may be used as a triangle leg, a unit-phase resource, or a boundary obstruction.*

Definition 9290.9 (Triangle parity boundary)**.** *A triangle classifier container $\mathcal{C}$ is a prime-classifier container whose inert subfamily*

$$I_{\mathcal{C}} := \tau_{\mathcal{C}}^{-1}(\mathsf{inert}_{\mathcal{C}})$$

is displayed as a finite obstruction window. Its parity boundary on a finite register row $n = \prod_p p^{v_p(n)}$ is

$$\partial_{\mathcal{C}}(n) := \big(v_p(n) \bmod 2\big)_{p \in I_{\mathcal{C}}}.$$

The row

$$\partial_{\mathcal{C}}(n) = 0$$

states that every inert obstruction class visible to $\mathcal{C}$ occurs with even multiplicity.

Principle 9290.10 (Even parity is triangle-norm closure)**.** *The role of Definition 9290.9 is the same as the classical two-square classifier over the Gaussian integer container: primes split by the container may supply phase resources, while inert classes must be paired before they contribute to a quadratic norm row. Thus*

$$\partial_{\mathcal{C}}(n) = 0$$

is a triangle-norm admissibility row. It says that the visible inert boundary has no unpaired class. It does not say that the resulting norm row has already been normalized to a unit-circle phase.

Definition 9290.11 (Norm-phase separation row)**.** *For a triangle classifier container $\mathcal{C}$, the norm-phase separation row is the refusal*

$$\mathsf{TriangleNormClosed}_{\mathcal{C}}(n) \not\Rightarrow \mathsf{UnitPhaseClosed}(n).$$

The left side records parity closure and quadratic norm admissibility. The right side requires a further normalization row eliminating radial scale. In a complex readout this is the difference between a norm block and a unit quotient

$$u = \frac{\alpha}{\overline{\alpha}}, \qquad |u| = 1.$$

Principle 9290.12 (Mirror pairing is not unit-circle closure)**.** *For a Li coordinate row*

$$z_\rho = e^{\ell + i\theta},$$

and a displayed positive observation index n, the mirror pair with radial coordinates ℓ and $-\ell$ is an even scale packet, but its averaged phase-aligned readout is

$$\frac{e^{n\ell}e^{in\theta} + e^{-n\ell}e^{in\theta}}{2} = \cosh(n\ell)e^{in\theta}.$$

When $\ell \# 0$, the positivity of n gives modulus $\cosh(n\ell) > 1$. It is therefore a norm-bearing paired packet, not a pure unit-circle phase. Consequently the RH-facing route must prove not only triangle parity closure, but the stronger radial-scale elimination row $\ell = 0$.

Definition 9290.13 (No hidden scale row)**.** *For a circle-triangle generation packet $\mathcal{G}$, the row*

$$\mathsf{NoHiddenScaleFromBEDC}(\mathcal{G})$$

states the following refusal. If a radial factor

$$e^{\ell}, \qquad \ell \# 0,$$

participates in a proposed closure, then either the factor is displayed in $\Lambda_{\mathcal{G}}$ as a residual scale ledger, or the closure is not admitted as a BEDC object. If the factor is displayed, the closure is not a pure unit-phase closure. If the factor is omitted, the closure has an unledgered residual and is not legal.

Principle 9290.14 (Pure phase closure has no radial leg)**.** *In a generation packet satisfying* $\mathsf{NoHiddenScaleFromBEDC}(\mathcal{G})$*, a pure phase closure cannot use a nonzero radial scale leg. Such a leg is either a recorded residual, which changes the object from pure phase closure to scale-ledgered closure, or an unrecorded residual, which violates ledger completeness.*

9290.3 Li Coordinates as Circle-Triangle Reads

Definition 9290.15 (Li zero coordinate)**.** *For a zero packet with complex coordinate $\rho \neq 0$, the Li coordinate is*

$$z_\rho := 1 - \frac{1}{\rho}.$$

When $z_\rho \neq 0$, a located scale-phase read is a representation

$$z_\rho = e^{\ell + i\theta}, \qquad \ell, \theta \in \mathsf{Real}^{\uparrow}.$$

The row $\ell = 0$ is exactly the unit-circle readout $z_\rho \in \mathbb{T}$.

Theorem 9290.16 (Li circle criterion)**.** *Let*

$$\rho = \sigma + i\gamma, \qquad \rho \neq 0.$$

Then

$$\left|1 - \frac{1}{\rho}\right| = 1 \quad \Longleftrightarrow \quad \sigma = \frac{1}{2}.$$

Proof. Write

$$z_\rho = 1 - \frac{1}{\rho} = \frac{\rho - 1}{\rho}.$$

The equality $|z_\rho| = 1$ is equivalent to

$$|\rho - 1| = |\rho|.$$

With $\rho = \sigma + i\gamma$, this is

$$(\sigma - 1)^2 + \gamma^2 = \sigma^2 + \gamma^2.$$

Cancelling the common term gives

$$-2\sigma + 1 = 0,$$

hence $\sigma = 1/2$. The same calculation read backwards gives the converse. □

Theorem 9290.17 (Li observation defect splits into scale and phase legs)**.** *If*

$$z_\rho = e^{\ell+i\theta},$$

then for every displayed $n \in \mathbb{N}_{>0}$*,*

$$|1 - z_\rho^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

Proof. Since

$$z_\rho^n = e^{n\ell}e^{in\theta},$$

we have

$$|1 - z_\rho^n|^2 = 1 + e^{2n\ell} - 2e^{n\ell}\cos(n\theta).$$

The right side rewrites as

$$e^{2n\ell} - 2e^{n\ell} + 1 + 2e^{n\ell} - 2e^{n\ell}\cos(n\theta),$$

which is the displayed decomposition. □

Principle 9290.18 (Scale leg is the radial obstruction)**.** *In the decomposition of Definition 9290.17, the term*

$$(e^{n\ell} - 1)^2$$

is the radial scale leg, while

$$2e^{n\ell}(1 - \cos(n\theta))$$

is the phase-observation leg. A pure unit-phase closure can use the second leg as an orthogonal readout of phase, but it cannot absorb a nonzero first leg without either recording radial scale or leaving a residual unrecorded.

9290.4 Conditional RH Route

Definition 9290.19 (Zeta BEDC circle-triangle embedding)**.** *For a completed non-trivial zeta zero packet* ρ*, the statement*

$$\mathsf{ZetaBEDCEmbedding}^{\circ\triangle}(\rho)$$

names a certificate-bearing existence obligation in the sense of Definition 9352.18. It requires a circle-triangle generation packet $\mathcal{G}$*, the prime-register readout of Definition 9290.5, the Li coordinate row of Definition 9290.15, the scale-phase defect row of Definition 9290.17, zero-ledger transport, and a* $\mathsf{NoHiddenScaleFromBEDC}(\mathcal{G})$ *row. It is a proof obligation, not an additional axiom and not a definition of* RH*.*

Theorem 9290.20 (Circle-triangle embedding is conditionally RH-sufficient)**.** *If every completed non-trivial zeta zero packet* ρ *carries a* $\mathsf{ZetaBEDCEmbedding}^{\circ\triangle}(\rho)$ *certificate, and if the associated located scale readback converts absence of apartness from zero into equality for the radial coordinate, then the BEDC RH section target is inhabited.*

Proof. Let ρ be a completed non-trivial zero packet and let

$$z_\rho = e^{\ell+i\theta}$$

be the scale-phase read supplied by its embedding certificate. The certificate carries $\mathsf{NoHiddenScaleFromBEDC}(\mathcal{G})$ and classifies the zero packet as a pure phase-prime-triangle closure. If $\ell\#0$, then for any displayed positive observation n the radial leg

$$(e^{n\ell}-1)^2$$

in Definition 9290.17 is nonzero. By Definition 9290.13, that leg must either be displayed as a residual scale ledger or remain unavailable. The first case is not a pure phase closure, and the second case is not a legal BEDC object. Both contradict the embedding certificate. Hence ℓ is not apart from 0. The located readback in the theorem hypothesis gives

$$\ell = 0.$$

Therefore

$$|z_\rho| = 1.$$

By Definition 9290.16, the coordinate of the zero packet satisfies

$$\Re(\rho) = \frac{1}{2}.$$

Since ρ was arbitrary, the section target is inhabited. □

Principle 9290.21 (Cofinal mechanism rather than pointwise repair)**.** *The embedding obligation is not a zero-by-zero repair rule. It asks for a single source mechanism whose finite observation windows grow cofinally: generated histories, unit phase reads, orthogonal observation events, prime-register ledgers, Li-coordinate scale reads, and no-hidden-scale refusal must be supplied by the same certificate family. The finite Pick-tower route of chapter 9325 is one analytic candidate for producing the required positive source mechanism.*

Principle 9290.22 (Status of the circle-triangle generation route)**.** *The circle-triangle generation route reduces the geometric part of the RH-facing argument to two explicit obligations:*

$$\mathsf{ZetaBEDCEmbedding}^{\circ\triangle} \qquad \textit{and} \qquad \mathsf{NoHiddenScaleFromBEDC}.$$

The algebraic identities for the Li coordinate and for the scale-phase defect are displayed above. The remaining mathematical content is the construction showing that completed zeta zero packets are indeed admitted through this generation mechanism and that radial scale cannot be silently supplied outside the ledger.

Chapter 9291

Memory Filtration and Prime–Zeckendorf Registers

chapter 9290 records the circle-triangle generation packet and the refusal to hide radial scale. The present chapter isolates the intermediate memory principle needed by that route. A later observation may contain the earlier certificate as ledger history, while the readable residue of that later observation is orthogonal to the transported earlier readout. The two statements live on different rows: append-only containment belongs to the history ledger, and orthogonality belongs to the residual Hilbert readout.

9291.1 Orthogonal Memory Filtration

Definition 9291.1 (Self-similar orthogonal filtration)**.** *A self-similar orthogonal filtration is a tower of certificate-bearing Hilbert readout rows*

$$\mathcal{H}_0 \longrightarrow \mathcal{H}_1 \longrightarrow \mathcal{H}_2 \longrightarrow \cdots$$

together with similarity transports

$$\sigma_m : \mathcal{H}_m \to \mathcal{H}_{m+1}$$

and residual rows E_{m+1} such that

$$\mathcal{H}_{m+1} = \sigma_m(\mathcal{H}_m) \oplus E_{m+1}, \qquad E_{m+1} \perp \sigma_m(\mathcal{H}_m).$$

The ledger row records the inclusion of histories

$$\mathsf{Hist}_m \subseteq \mathsf{Hist}_{m+1},$$

while the Hilbert row records only the orthogonal decomposition of the readable residual. Thus the notation

$$B_{m+1} \supseteq B_m$$

belongs to the append-only ledger surface, whereas the notation

$$E_{m+1} \perp \sigma_m(B_m)$$

belongs to the residual readout surface.

Principle 9291.2 (Containment and orthogonality occupy different rows)**.** *A generated observation state may retain the prior state as history and still expose only an orthogonal residual in the readout row. The licensed statement is therefore not $B_{m+1} \perp B_m$. It is the split*

$$B_{m+1} \cong \sigma_m(B_m) \oplus E_{m+1}, \qquad E_{m+1} \perp \sigma_m(B_m).$$

Containment says that the ledger has not discarded the earlier certificate. Orthogonality says that the residual read by the next coordinate carries no component in the transported prior readout.

Theorem 9291.3 (Pythagorean innovation law)**.** *Let*

$$v_{m+1} = \sigma_m(v_m) + \eta_{m+1}$$

be a vector readout in a self-similar orthogonal filtration, with

$$\eta_{m+1} \in E_{m+1}.$$

Then

$$\|v_{m+1}\|^2 = \|\sigma_m(v_m)\|^2 + \|\eta_{m+1}\|^2.$$

If the similarity row has scale α_m on the displayed vector,

$$\|\sigma_m(v_m)\| = \alpha_m \|v_m\|,$$

then

$$\|v_{m+1}\|^2 = \alpha_m^2 \|v_m\|^2 + \|\eta_{m+1}\|^2.$$

Proof. The defining split gives

$$\eta_{m+1} \perp \sigma_m(v_m).$$

The norm identity is therefore the ordinary Pythagorean identity in the readout row. Substituting the displayed similarity scale gives the final formula. □

Definition 9291.4 (Orthogonal history tower)**.** *An orthogonal history tower is a filtration of Hilbert readout rows*

$$\mathcal{H}_0 \subseteq \mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots$$

whose residual rows are

$$E_{m+1} := \mathcal{H}_{m+1} \ominus \mathcal{H}_m$$

and whose ledger row records the append-only containment of histories. Equivalently, every finite stage has the displayed orthogonal split

$$\mathcal{H}_N = \mathcal{H}_0 \oplus E_1 \oplus \cdots \oplus E_N.$$

This is the tower-level form of the containment-plus-residual split: history is retained by the filtration, while readable residuals occupy orthogonal summands.

Theorem 9291.5 (Pythagorean formula for an orthogonal history tower)**.** *Let*

$$v_N \in \mathcal{H}_N$$

be displayed in the orthogonal history tower of Definition 9291.4 as

$$v_N = v_0 + \eta_1 + \cdots + \eta_N, \qquad v_0 \in \mathcal{H}_0, \qquad \eta_j \in E_j.$$

Then

$$\|v_N\|^2 = \|v_0\|^2 + \sum_{j=1}^{N} \|\eta_j\|^2.$$

Proof. By Definition 9291.4, the stage $\mathcal{H}_N$ is the orthogonal direct sum

$$\mathcal{H}_0 \oplus E_1 \oplus \cdots \oplus E_N.$$

Therefore

$$\langle v_0, \eta_j \rangle = 0 \qquad \text{and} \qquad \langle \eta_i, \eta_j \rangle = 0 \quad (i \neq j).$$

Expanding the inner product of $v_0 + \eta_1 + \cdots + \eta_N$ with itself leaves only the diagonal norm terms, which gives the displayed identity. □

Principle 9291.6 (Observation time as residual accumulation)**.** *In an orthogonal history tower, the observation order is visible as append-only history together with accumulated orthogonal residuals. If a stage transition has residual $\eta_{m+1} \neq 0$, then the readout energy changes by*

$$\|v_{m+1}\|^2 = \|v_m\|^2 + \|\eta_{m+1}\|^2$$

at that stage resolution. Thus the tower's time row is not an external clock inserted into the object; it is the generated direction along which residual square-norm entries are ledgered.

Principle 9291.7 (Creation-like growth is ledgered residual growth)**.** *The filtration is creation-like only at the level of certified generation: a later observation contains the earlier history, transports that history into the later readout space, and records the residual in an orthogonal row. The residual is not hidden by the inclusion $\mathsf{Hist}_m \subseteq \mathsf{Hist}_{m+1}$; it is exactly the part that must remain auditable as a ledger row. Binary growth, Fibonacci-normal growth, and prime-exponent growth are legal only after their own normal-form classifiers and residual ledgers are displayed.*

9291.2 Prime–Zeckendorf–Goedel Register

Definition 9291.8 (Prime–Zeckendorf–Goedel register)**.** *Let F_k be the Fibonacci scale used by the displayed Zeckendorf normal-form classifier. The Prime–Zeckendorf–Goedel register of a positive integer n is the finite-support binary array*

$$Z_{\mathsf{PZG}}(n) := (z_{p,k})_{p,k}$$

defined by the two-stage normal form

$$n = \prod_p p^{v_p(n)}, \qquad v_p(n) = \sum_k z_{p,k} F_k,$$

where

$$z_{p,k} \in \{0,1\}, \qquad z_{p,k} z_{p,k+1} = 0$$

for every fixed prime p. Equivalently,

$$n = \prod_{p,k} \left(p^{F_k}\right)^{z_{p,k}}.$$

The prime coordinate supplies multiplicative Goedel addressability; the Zeckendorf coordinate supplies additive self-similar resolution inside each prime exponent.

Theorem 9291.9 (Prime–Zeckendorf–Goedel uniqueness)**.** *Every positive integer n determines a unique finite-support binary array $Z_{\mathsf{PZG}}(n) = (z_{p,k})_{p,k}$ satisfying the no-adjacent condition in each prime row and*

$$n = \prod_{p,k} \left(p^{F_k}\right)^{z_{p,k}}.$$

Proof. Unique factorization gives the unique finite-support exponent family $(v_p(n))_p$ with

$$n = \prod_p p^{v_p(n)}.$$

For each displayed exponent $v_p(n)$, the Zeckendorf normal-form classifier gives a unique binary row $(z_{p,k})_k$ satisfying

$$z_{p,k} z_{p,k+1} = 0$$

and

$$v_p(n) = \sum_k z_{p,k} F_k.$$

Substituting these rows into the prime factorization gives the displayed product. Uniqueness follows from the uniqueness of the prime exponent row together with the uniqueness of the Zeckendorf row for each exponent. □

Definition 9291.10 (Prime–Zeckendorf–Goedel Hilbert readout)**.** *Let*

$$\mathcal{H}_{\mathsf{PZG}} := \ell^2(\mathsf{Prime}^{\uparrow} \times \mathbb{N})$$

with standard orthonormal basis $e_{p,k}$*. The Hilbert readout of* n *is the finite vector*

$$\nu_{\mathsf{PZG}}(n) := \sum_{p,k} z_{p,k} e_{p,k}.$$

Thus

$$\langle e_{p,k}, e_{q,\ell} \rangle = \delta_{p,q} \delta_{k,\ell}.$$

Theorem 9291.11 (Register extensions give orthogonal innovations)**.** *Let* $S_0 \subseteq S_1$ *be two finite supports in* $\mathsf{Prime}^{\uparrow} \times \mathbb{N}$*, and let*

$$v_i := \sum_{(p,k) \in S_i} e_{p,k} \qquad (i = 0, 1).$$

For

$$R_1 := S_1 \setminus S_0, \qquad \eta_1 := \sum_{(p,k) \in R_1} e_{p,k},$$

one has

$$v_1 = v_0 + \eta_1, \qquad \eta_1 \perp v_0,$$

and hence

$$\|v_1\|^2 = \|v_0\|^2 + \|\eta_1\|^2.$$

Proof. The basis vectors indexed by R_1 are disjoint from those indexed by S_0. Orthonormality in Definition 9291.10 therefore gives $\eta_1 \perp v_0$. The displayed vector decomposition follows from $S_1 = S_0 \sqcup R_1$, and the norm identity follows from the Pythagorean identity. □

Definition 9291.12 (Primality-supported layer)**.** *A layer* $\mathcal{L}_m$ *is* primality-supported *when it carries a displayed atom set* Π_m*, a Hilbert readout*

$$\mathcal{H}_m := \ell^2(\Pi_m),$$

with orthonormal basis $(e_\pi)_{\pi\in\Pi_m}$*, and a unique register map*

$$\nu_m : \mathcal{L}_m \to \mathcal{H}_m$$

of the form

$$\nu_m(X) = \sum_{\pi\in\Pi_m} v_\pi(X) e_\pi.$$

The atoms π *may be ordinary primes, prime ideals, irreducible branch rows, irreducible register bits, or irreducible observation residuals, provided the displayed layer supplies unique coordinates and source provenance for them.*

Theorem 9291.13 (Primality-supported layers produce stable triangles)**.** *Let* $\mathcal{L}_m$ *and* $\mathcal{L}_{m+1}$ *be primality-supported layers with*

$$\Pi_m \subseteq \Pi_{m+1}.$$

For any displayed vector $v \in \ell^2(\Pi_{m+1})$*, write*

$$P_m v := \sum_{\pi\in\Pi_m} v_\pi e_\pi, \qquad \eta_{m+1} := \sum_{\pi\in\Pi_{m+1}\setminus\Pi_m} v_\pi e_\pi.$$

Then

$$v = P_m v + \eta_{m+1}, \qquad \eta_{m+1} \perp P_m v,$$

and therefore

$$\|v\|^2 = \|P_m v\|^2 + \|\eta_{m+1}\|^2.$$

Proof. The basis vectors indexed by Π_m and by $\Pi_{m+1} \setminus \Pi_m$ are disjoint members of the displayed orthonormal basis of $\ell^2(\Pi_{m+1})$. Hence the two displayed sums are orthogonal and add to v. The norm identity is the Pythagorean identity in the Hilbert readout row. □

Principle 9291.14 (Primality is coordinate support for stable triangles)**.** *The stable triangle row is not prior geometry. It is the orthogonal readout of a layer whose atoms have unique source coordinates. If a stage has no atom set, or if an apparent residual has no unique primality-supported coordinate, then its squared norm is not a stable object-level triangle invariant. Thus primality support is the coordinate condition behind the Pythagorean rows of Definition 9291.3 and Definition 9291.5.*

Definition 9291.15 (PZG-supported residual row)**.** *Let* $S_m \subseteq S_{m+1}$ *be finite supports in* $\mathsf{Prime}^{\uparrow} \times \mathbb{N}$*, and set*

$$R_{m+1} := S_{m+1} \setminus S_m.$$

A residual row over this stage is PZG-supported *when it has the displayed form*

$$\eta_{m+1} = \sum_{(p,k)\in R_{m+1}} c_{p,k} e_{p,k},$$

with finite coefficient rows and with every basis vector $e_{p,k}$ *coming from the Prime–Zeckendorf–Goedel atom indexed by* (p,k)*. It is degenerate when all displayed coefficients are zero. It is unledgered when no finite support row* R_{m+1} *is supplied.*

Principle 9291.16 (PZG support is the nondegenerate triangle-leg criterion)**.** *A stage of an orthogonal history tower can be read as a nondegenerate triangle-producing stage only through a PZG-supported residual row. If R_{m+1} is empty, or if every coefficient in Definition 9291.15 is zero, the stage has no separate orthogonal leg and the squared norm does not gain a residual term. If a residual is present but no finite PZG support row is displayed, the stage is not a BEDC object-level triangle row, because the source ledger of the leg is absent.*

Theorem 9291.17 (PZG-supported stages give norm-square growth)**.** *Let $\mathcal{H}_m$ be spanned by $\{e_{p,k} : (p,k) \in S_m\}$, and let η_{m+1} be a PZG-supported residual over $R_{m+1} = S_{m+1} \setminus S_m$. For every $v_m \in \mathcal{H}_m$,*

$$v_{m+1} := v_m + \eta_{m+1}$$

satisfies

$$\|v_{m+1}\|^2 = \|v_m\|^2 + \sum_{(p,k)\in R_{m+1}} |c_{p,k}|^2.$$

Proof. The support R_{m+1} is disjoint from S_m. Therefore every basis vector appearing in η_{m+1} is orthogonal to every basis vector spanning $\mathcal{H}_m$ by Definition 9291.10. Hence $\eta_{m+1} \perp v_m$. Expanding the norm of $v_m + \eta_{m+1}$ leaves the norm of v_m and the diagonal coefficient sum of η_{m+1}, giving the displayed identity. □

Principle 9291.18 (Prime–Zeckendorf–Goedel readout as particle ledger)**.** *The phase*

$$e^{-it\log n}$$

is continuous on the unit circle, but its frequency source is the finite register

$$\log n = \sum_p v_p(n)\log p = \sum_{p,k} z_{p,k} F_k \log p.$$

Thus the same zeta mode has a wave row, read as unit phase, and a particle-ledger row, read as a finite Prime–Zeckendorf–Goedel support vector. A radial scale coordinate is not supplied by either row.

Theorem 9291.19 (Single-mode continuous-discrete bridge)**.** *Let $n \in \mathbb{N}_{>0}$ and let*

$$Z_{\mathsf{PZG}}(n) = (z_{p,k})_{p,k}$$

be its Prime–Zeckendorf–Goedel register. Then for every $t \in \mathsf{Real}^{\uparrow}$,

$$e^{-it\log n} = \exp\left(-it\sum_{p,k} z_{p,k} F_k \log p\right).$$

Proof. By Definition 9291.9,

$$n = \prod_{p,k} \left(p^{F_k}\right)^{z_{p,k}}.$$

Taking logarithms gives

$$\log n = \sum_{p,k} z_{p,k} F_k \log p.$$

Substitution into $e^{-it\log n}$ gives the displayed identity. □

Chapter 9292

PZG Phase Readback and Log Energy

chapter 9291 defines the Prime–Zeckendorf–Goedel register and the single-mode continuous-discrete bridge. The present chapter records the associated readback distinction. The finite PZG register is unique. The complete Dirichlet-monomial character reads its frequency faithfully. A visible unit-circle projection of a solenoid state does not, by itself, recover the prime-register fibre behind that phase.

9292.1 Log-Energy Readout

Definition 9292.1 (Log-energy PZG readout)**.** *The log-energy Prime–Zeckendorf–Goedel readout replaces the standard orthonormal basis by a weighted orthogonal basis* $(\varepsilon_{p,k})_{p,k}$ *with*

$$\|\varepsilon_{p,k}\|^2 = F_k \log p, \qquad \langle \varepsilon_{p,k}, \varepsilon_{q,\ell} \rangle = 0 \quad ((p,k) \neq (q,\ell)).$$

For a positive integer n*, set*

$$R_{\mathsf{PZG}}(n) := \sum_{p,k} z_{p,k} \varepsilon_{p,k}.$$

Theorem 9292.2 (Log-energy reads logarithmic height)**.** *For every positive integer* n*,*

$$\|R_{\mathsf{PZG}}(n)\|^2 = \log n.$$

Consequently, for every $t \in \mathsf{Real}^{\uparrow}$*,*

$$n^{-it} = e^{-it\|R_{\mathsf{PZG}}(n)\|^2}.$$

Proof. By Definition 9292.1 and orthogonality,

$$\|R_{\mathsf{PZG}}(n)\|^2 = \sum_{p,k} z_{p,k} F_k \log p.$$

The last sum is $\log n$ by the PZG expansion in Definition 9291.9. Substitution into $n^{-it} = e^{-it \log n}$ gives the phase identity. □

Principle 9292.3 (Zeckendorf carries preserve log energy)**.** *Inside a fixed prime row, the Zeckendorf carry*

$$\varepsilon_{p,k} + \varepsilon_{p,k+1} \rightsquigarrow \varepsilon_{p,k+2}$$

preserves the displayed log-energy because

$$F_k \log p + F_{k+1} \log p = F_{k+2} \log p.$$

Thus the no-adjacent normal form is not merely a coding convention. It is the register normal form compatible with the triangular square-norm row used by the PZG Hilbert readout.

9292.2 Finite Self-Description and Residual Ledgers

Definition 9292.4 (PZG finite description row)**.** *For a certificate-bearing packet $\mathcal{C}$, a PZG finite description row is a displayed triple*

$$(d_{\mathcal{C}}, \ulcorner d_{\mathcal{C}} \urcorner, Z_{\mathsf{PZG}}(\ulcorner d_{\mathcal{C}} \urcorner)).$$

Here $d_{\mathcal{C}}$ is a finite syntactic description of the source rows, generator rows, readout rows, ledger rows, and local NameCert *boundary of $\mathcal{C}$; the code $\ulcorner d_{\mathcal{C}} \urcorner$ is its positive integer address; and $Z_{\mathsf{PZG}}(\ulcorner d_{\mathcal{C}} \urcorner)$ is the unique Prime–Zeckendorf–Goedel register of that address.*

Principle 9292.5 (Finite self-description is syntactic)**.** *A PZG finite description row may describe the packet in which it occurs, but only as finite syntax. It records which source, generator, readout, ledger, and naming rows are displayed. It is not the packet itself, not a semantic truth predicate for the packet, and not a replacement for any* NameCert *field used by downstream consumers.*

Theorem 9292.6 (Finite descriptions have unique PZG addresses)**.** *Every PZG finite description row has a unique register address.*

Proof. The displayed description $d_{\mathcal{C}}$ is finite, hence it has a positive integer code $\ulcorner d_{\mathcal{C}} \urcorner$. The uniqueness theorem Definition 9291.9 assigns to that integer one Prime–Zeckendorf–Goedel register and no other. Therefore the finite description row has a unique register address. □

Definition 9292.7 (Total PZG identity code)**.** *For a PZG-facing object X, its total PZG identity code is the PZG-addressed packet*

$$\mathsf{Code}_{\mathcal{K}}(X) := Z_{\mathsf{PZG}}\big(\mathsf{data}(X), \mathsf{rules}(X), \mathsf{ledger}(X)\big).$$

The row $\mathsf{data}(X)$ records source and generator data. The row $\mathsf{rules}(X)$ records formation, normalization, readout, and transport rules. The row $\mathsf{ledger}(X)$ records residual, tail, boundary, completion, and semantic rows not closed by the public readout.

Principle 9292.8 (PZG identity principle)**.** *For PZG-facing objects X and Y, kernel identity is read by the total PZG identity code:*

$$X \equiv_{\mathcal{K}} Y \quad \Longleftrightarrow \quad \mathsf{Code}_{\mathcal{K}}(X) = \mathsf{Code}_{\mathcal{K}}(Y).$$

Object identity is therefore not a further external decision once the data, rule, and ledger rows have been fixed. The ledger is part of the identity code, not an annotation outside the object.

Theorem 9292.9 (No hidden register under PZG identity)**.** *No PZG-facing process can change the licensed behaviour of an object while leaving its data, rule, and ledger rows unchanged.*

Proof. If the data, rule, and ledger rows are unchanged, then

$$\mathsf{Code}_{\mathcal{K}}(X) = \mathsf{Code}_{\mathcal{K}}(Y).$$

By Definition 9292.8, $X \equiv_{\mathcal{K}} Y$. Any behaviour not determined by the displayed data, rules, and ledger is not a licensed row of the object. Hence a licensed behavioural change must appear in one of those rows, and then the total PZG identity code is not unchanged. □

Corollary 9292.10 (Scale inverse changes identity or ledger)**.** *For a zeta-facing PZG object, an independent positive-real scale inverse for the off-half residual of Definition 9294.5 either changes the total PZG identity code, appears in the ledger, or is not part of the object.*

Proof. A positive-real scale inverse that closes an off-half residual changes the licensed behaviour of the zeta-facing object. By Definition 9292.9, that behavioural change must be recorded in data, rules, or ledger. Recording it in data or rules changes $\mathsf{Code}_{\mathcal{K}}$. Recording it as residual or completion data places it in the ledger. If none of these rows records it, the scale inverse is not a licensed part of the object. □

Definition 9292.11 (Finite PZG readout and residual ledger)**.** *A finite PZG readout for a register row is a classifier that depends on a finite set of coordinates (p, k). A residual ledger for that readout records a detected discrepancy together with its source row, detecting readout, status, and continuation route. A residual is closed inside the stated scope only when every nonzero finite discrepancy is either detected by a displayed finite readout or carried by a displayed ledger row.*

Theorem 9292.12 (Closed finite readout has no hidden finite residual)**.** *Assume a PZG readout scope in which every nonzero finite residual is either detected by one of the displayed finite readouts or entered into the displayed residual ledger. If the residual ledger has no open entry, then no nonzero finite residual remains hidden inside that scope.*

Proof. Suppose that a nonzero finite residual remains. By the coverage assumption, it is either detected by a displayed finite readout or entered into the residual ledger. A detected residual must be recorded by the ledger of the readout that detected it, so it gives an open ledger entry. An entered residual is already an open ledger entry. Both alternatives contradict the assumption that the residual ledger has no open entry. □

Definition 9292.13 (PZG generated-interface profile)**.** *A PZG-facing interface carries the generated-interface profile when it displays the following three rows.*

The generation row records the history by which the interface is produced: source history, event history, register update, and the replay route through the relevant Cont *or package provenance rows.*

The canonical readout row records the normal form by which the interface is publicly read: PZG uniqueness, Zeckendorf normalization, log-energy readout, norm-square readout, or unit-phase readout, according to the scope.

The ledger row records every residual not closed by the displayed canonical readout: finite discrepancies, hidden solenoid fibres, radial scale defects, tail windows, and semantic self-description residues.

Theorem 9292.14 (Generated PZG concepts inherit the profile)**.** *Let $\mathcal{C}$ be a PZG-facing interface generated by finitely many source interfaces and a displayed finite formation rule. If the source interfaces carry the generated-interface profile and the formation rule preserves their replay, normal-form, and ledger rows, then $\mathcal{C}$ carries the generated-interface profile.*

Proof. The generation row of $\mathcal{C}$ is obtained by appending the displayed formation rule to the source histories and replaying the associated provenance. Its canonical readout is the normal form specified by the formation rule, together with the source normal forms transported through that rule. Its ledger is the union of the transported source ledgers with the residual ledger created by the formation rule. These are precisely the three rows of Definition 9292.13. □

Corollary 9292.15 (Self-description keeps the profile boundary)**.** *A PZG finite description row for a packet may inherit generation, canonical readout, and ledger rows, but it remains a finite syntactic row. Any attempt to read that row as a complete same-layer semantic decision procedure is a residual ledger entry, not a closed PZG readout.*

Proof. The finite description row is generated by encoding a displayed finite description and then applying the unique PZG address row of Definition 9292.6. Its canonical readout is therefore its unique register address. By Definition 9292.5, the row is not a truth predicate or a replacement for $\mathsf{NameCert}$ fields. Any semantic decision claim beyond the finite syntax must therefore be recorded by the ledger row of Definition 9292.11. □

Principle 9292.16 (Self-code is not analytic completion)**.** *A PZG finite description row may encode the finite rules that generate PZG registers and the finite rules that read them. This does not identify that finite code with the analytic continuation or completed zeta carrier obtained from those readout rules. The valid implication is*

$$\mathsf{SelfCode}_{\mathsf{PZG}} \Longrightarrow \mathsf{Code}(\mathsf{PZG\ readout\ rules}),$$

not

$$\mathsf{SelfCode}_{\mathsf{PZG}} \Longrightarrow \mathsf{AnalyticCompletion}_{\mathsf{PZG}}.$$

Any claim of analytic completion must therefore display the analytic readout, continuation, completion, and pole–zero ledger rows.

Definition 9292.17 (PZG analytic completion certificate)**.** *A PZG analytic completion certificate for a PZG-facing packet is a displayed tuple*

$$(\mathsf{PZG}_{\mathrm{fin}}, L, Z_{\mathsf{PZG}}, \mathsf{Cont}_{\mathrm{an}}, \mathsf{Arch}_{\infty}, \mathsf{ZeroLedger}).$$

Here $\mathsf{PZG}_{\mathrm{fin}}$ *is the finite-register source row,* L *is the log-energy readout,* Z_{PZG} *is the right-half-plane Dirichlet or partition readout determined by* L*,* $\mathsf{Cont}_{\mathrm{an}}$ *is the analytic continuation row,* Arch_{∞} *is the archimedean completion row, and* $\mathsf{ZeroLedger}$ *records poles, zero packets, branch choices, tail defects, and semantic self-reference residues. The certificate presents the analytic completion as a generated readout boundary over PZG, not as the finite PZG code itself.*

Theorem 9292.18 (Self-readout completion requires a completion certificate)**.** *A PZG finite description row by itself licenses finite syntactic self-description. It licenses a PZG analytic completion only after a PZG analytic completion certificate is displayed.*

Proof. The finite description row is a finite code by Definition 9292.5. By Definition 9292.16, such a code may record the rules for PZG generation and PZG readout, but it is not the analytic continuation or completed carrier obtained from those rules. The missing analytic data are exactly the continuation, archimedean completion, and pole–zero ledger rows listed in Definition 9292.17. Therefore analytic completion is licensed precisely when those rows are displayed. □

Definition 9292.19 (Inverse PZG recovery problem)**.** *The inverse PZG recovery problem begins with a zeta-facing completed readout and asks which finite source rows are forced by that readout. A recovery certificate consists of: the Dirichlet or partition row that displays a discrete length spectrum; the Euler local-factor row that separates the spectrum into prime-indexed local geometric fibres; the normalization row that converts prime exponents into PZG coordinates; the analytic continuation and archimedean completion rows; the reflection row when a completed symmetric carrier is claimed; and the zero ledger recording global cancellation nodes. The problem is therefore not to assume PZG as an input to* ζ*, but to recover the least PZG-shaped source compatible with the displayed analytic rows.*

Principle 9292.20 (Zero data alone do not recover the source)**.** *Zero packets of a completed carrier are global cancellation data. They are not finite PZG registers and not prime-local source rows. Recovering a PZG source from a completed zeta-facing object requires the right-half-plane Dirichlet or partition readout together with the Euler local-factor row and the normalization row. This is the same boundary as Definition 9250.7: zero packets belong to the completed recovered carrier, while PZG belongs to the finite source and normal-form side of the certificate.*

Theorem 9292.21 (Inverse recovery factors through Euler-normalized source rows)**.** *If a zeta-facing completed readout recovers a PZG source, then the recovery factors through a displayed Euler local-factor row and a displayed PZG normalization row. Analytic continuation, reflection, or zero data without those rows do not determine the PZG source certificate.*

Proof. By Definition 9292.19, the finite source is recovered from the analytic carrier only after the Dirichlet or partition readout is decomposed into Euler local factors. Those factors supply the prime-indexed exponent register. The PZG source is obtained only after the normalization row rewrites each exponent into its Zeckendorf coordinate presentation. Analytic continuation, reflection, and zero data govern the completed readout boundary and its cancellation ledger; without the Euler local-factor and normalization rows, they do not display the finite prime-indexed source from which a PZG certificate can be read. □

Definition 9292.22 (Legal visible motion certificate)**.** *A PZG-facing trajectory has a legal visible motion certificate when it displays both a discrete generation row and an implementation row. The discrete generation row is a sequence of generated states whose adjacent steps are obtained by an event history, register normalization, finite readout, and ledger update. The implementation row is a visible phase path*

$$c : I \to \mathbb{T}$$

with sampling points $s_t \in I$ such that

$$c(s_t) = \pi(\rho(K_t)),$$

where $\rho(K_t)$ is the displayed solenoid or phase lift of the generated state. The path is continuous, piecewise C^1, or C^1 according to the claimed scope, and every break point, branch change, or hidden-fibre transition is recorded by a ledger row.

Principle 9292.23 (Discrete legality is not differentiability)**.** *The generated PZG trajectory is a discrete BEDC object. Its legality says that adjacent states are produced by displayed event, normalization, readout, and ledger rows. It does not by itself assert a continuous or differentiable curve in $\mathbb{T}$ or in a solenoid carrier. Continuous, piecewise C^1, or C^1 motion is an implementation certificate over the visible phase surface.*

Theorem 9292.24 (Visible motion has no unledgered jump)**.** *If a PZG-facing trajectory carries a legal visible motion certificate, then every admitted change is one of the displayed generation steps, a point of the visible phase path, a recorded hidden-fibre transition, or a ledgered break point.*

Proof. By Definition 9292.22, adjacent generated states are supplied by the discrete generation row. Between sampled states, the visible implementation is the displayed phase path $c : I \to \mathbb{T}$ on the claimed regularity scope. Hidden-fibre transitions and break points are required by the same certificate to be ledger rows. Thus no remaining change is admitted without either generation, visible implementation, or ledger evidence. □

Definition 9292.25 (Prime-addressed hidden transition)**.** *In a legal visible motion certificate, a recorded hidden-fibre transition is prime-addressed when its ledger row displays a finite support*

$$S \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}$$

together with component rows

$$k_p \in \mathbb{Z}_p \qquad (p \in S)$$

in the solenoid fibre. Equivalently, in a PZG readout the transition may be addressed by a finite set of register coordinates

$$D \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow} \times \mathbb{N}$$

and the associated component rows at those coordinates. A hidden transition without such a prime support row is not an admissible hidden motion inside the stated certificate; it is a residual ledger entry until a support row or tail route is displayed.

Definition 9292.26 (Finite prime-type motion window)**.** *A finite prime-type motion window consists of a time interval I, a finite prime support $S \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}$, a depth bound L, and a classifier row*

$$\tau : S \to \{\mathsf{ramified}, \mathsf{split}, \mathsf{inert}\}.$$

For the Gaussian two-axis readout, the classifier is the usual trichotomy

$$\tau(2) = \mathsf{ramified}, \qquad p \equiv 1 \pmod 4 \Rightarrow \tau(p) = \mathsf{split}, \qquad p \equiv 3 \pmod 4 \Rightarrow \tau(p) = \mathsf{inert}.$$

Only transitions whose PZG coordinates lie in S and have depth at most L are closed inside the window. Any component outside this displayed support or beyond this depth is recorded by the residual or tail ledger.

Theorem 9292.27 (Prime-addressed motion is locally classified)**.** *Let a PZG-facing trajectory carry a legal visible motion certificate. If each recorded hidden-fibre transition is prime-addressed inside a finite prime-type motion window, then every hidden transition admitted by the certificate decomposes into ramified, split, and inert components relative to the displayed classifier. No out-of-window component is treated as a closed hidden motion unless it is entered into the residual or tail ledger.*

Proof. By Definition 9292.25, each recorded hidden-fibre transition has a finite prime support row, or a finite PZG coordinate support row. By Definition 9292.26, every displayed prime in the support is assigned exactly one of the three classifier values ramified, split, and inert inside the window. Therefore the finite support decomposes as the disjoint union of its three classifier fibres, and the corresponding component rows decompose accordingly. Components outside the support or beyond the depth bound are explicitly outside the closed window, so the same definition requires them to be ledgered rather than used as closed hidden motion. □

9292.3 Phase Readback

Theorem 9292.28 (Complete monomial phase character is register faithful)**.** *Let $m, n \in \mathbb{N}_{>0}$. If*

$$e^{-it\log n} = e^{-it\log m} \qquad \textit{for every } t \in \mathsf{Real}^{\uparrow},$$

then

$$Z_{\mathsf{PZG}}(n) = Z_{\mathsf{PZG}}(m).$$

Proof. The displayed equality gives

$$e^{-it(\log n - \log m)} = 1 \qquad \text{for every } t \in \mathsf{Real}^{\uparrow}.$$

A real frequency whose exponential is constantly 1 on the full real line has zero frequency. Hence $\log n = \log m$, and therefore $n = m$. The conclusion follows from Definition 9291.9. □

Principle 9292.29 (Solenoid projection is not register readback)**.** *The faithfulness in Definition 9292.28 concerns the full one-parameter character $t \mapsto e^{-it\log n}$ attached to one Dirichlet monomial. It is not a claim that the visible projection*

$$\pi_S : \Sigma_S \to \mathbb{T}$$

of Definition 9413.779 is injective. The kernel

$$K_S := \ker(\pi_S)$$

is the prime-register fibre retained by Definition 9289.4. Therefore two solenoid source rows may have the same visible unit-circle phase and still differ by a prime-register fibre row. Recovering the register requires the fibre ledger or a certified section; the circle output alone is only the visible phase surface.

Principle 9292.30 (Instantaneous phase is not a register certificate)**.** *At a fixed time t_0, equality of phase reads records only*

$$t_0(\log n - \log m) \in 2\pi\mathbb{Z}.$$

It is therefore a modular phase observation, not a certificate of register equality. The complete monomial character is faithful by Definition 9292.28, while a solenoid-visible phase is fibre-blind by Definition 9292.29. These are the two distinct uniqueness rows: the PZG source ledger determines its phase readout, but a projected phase value does not reconstruct the prime-register fibre without the solenoid ledger.

Principle 9292.31 (Solenoid fibres are not radial scale ledgers)**.** *For a centered active mode*

$$U_a(s) = e^{-rh_a} e^{-ith_a}, \qquad h_a > 0,$$

the phase factor e^{-ith_a} is compatible with a PZG or solenoid phase readout, but the positive factor e^{-rh_a} is not a compact prime-register fibre coordinate. This is the same separation as Definition 9286.3: the legal hidden fibre is K_S, not $\mathsf{Real}^{\uparrow}_{>0}$. Hence an off-boundary radial factor must be recorded as a scale row or refused as unledgered data; it cannot be reclassified as lost prime-register phase.

Chapter 9293

Zeta Virtual Machine Execution Trace

chapter 9250 reads zeta as a recovered carrier assembled from Dirichlet, Euler, continuation, completion, mirror, and zero-ledger rows. chapter 9292 supplies the PZG length and phase readback behind that carrier, while chapter 9258 separates finite zero reads from completed zero objects. The present chapter records the program semantics joining these rows: the ontology kernel is consumed as an instruction surface, zeta is the all-prime execution trace of that surface, and zero search is a separate certified readout process over the completed trace.

Definition 9293.1 (Ontology virtual machine row)**.** *An ontology virtual machine row is a BEDC-facing instruction surface*

$$\mathsf{VM}_{\mathcal{K}} := \begin{pmatrix} \mathsf{Gen}, & \mathsf{Enc}, & \mathsf{Norm}, & \mathsf{Decode}, & \mathsf{Length}, & \mathsf{Phase}, \\ \mathsf{Read}, & \mathsf{Complete}, & \mathsf{Reflect}, & \mathsf{Ledger}, & \mathsf{Refine}, & \mathsf{Certify} \end{pmatrix}.$$

The instruction names are schematic abbreviations for already displayed BEDC rows. Generation extends histories; encoding assigns finite syntax to PZG addresses; normalization gives canonical PZG form; decoding reads the positive-integer source; length and phase read the log-energy and unit-phase rows; finite reading consumes a displayed window; completion and reflection build the analytic and mirror rows; ledgering records residuals; refinement enlarges the visible window; certification turns a finite read together with the required tail and boundary rows into an auditable theorem packet.

Principle 9293.2 (The kernel is executable semantics)**.** *The BEDC ontology kernel is not an external list of mathematical objects. In the RH-facing route it is used as the executable semantics of Definition 9293.1: each accepted object is generated by instructions whose source histories, normal forms, public readouts, and residual ledgers are visible. This is the same discipline as the dynamic ontology state row of Definition 9352.22 and the BEDC–PZG dynamic tuple of Definition 9352.23.*

Definition 9293.3 (PZG length heat trace)**.** *Let* $\mathsf{Z}_{\mathsf{PZG}}$ *be the finite-support PZG register carrier, and let*

$$L(z) := \sum_{p,k} z_{p,k} F_k \log p$$

be the log-energy length row. The right-half-plane PZG length heat trace is

$$\mathsf{ZetaRead}(s) := \sum_{z \in \mathsf{Z}_{\mathsf{PZG}}} e^{-sL(z)}$$

on every displayed convergence window where the sum is licensed.

Theorem 9293.4 (PZG heat trace is the Dirichlet zeta readout)**.** *On the right-half-plane convergence window, the PZG length heat trace is the Dirichlet zeta readout:*

$$\mathsf{ZetaRead}(s) = \sum_{n\geq 1} n^{-s}.$$

Proof. By Definition 9291.9, each positive integer n has a unique PZG register $Z_{\mathsf{PZG}}(n)$, and every finite-support legal register is the register of its decoded positive integer. By Definition 9292.2,

$$L(Z_{\mathsf{PZG}}(n)) = \log n.$$

Hence the summand indexed by the register of n is

$$e^{-sL(Z_{\mathsf{PZG}}(n))} = e^{-s\log n} = n^{-s}.$$

Reindexing the register sum by the decoded positive integers gives the displayed Dirichlet row. □

Principle 9293.5 (Zeta is a readout object, not a search program)**.** *The symbol ζ names the completed recovered-carrier readout whose right-half-plane projection is Definition 9293.3; it is not itself the procedure that searches for zeros. The search procedure is a separate packet over the completed ξ readout. Thus the source separation is*

$$\mathsf{VM}_{\mathcal{K}} \rightsquigarrow \mathsf{ZetaRead} \rightsquigarrow \mathsf{XiRead} \rightsquigarrow \mathsf{ZeroSearch}_{\xi},$$

where XiRead *adds the analytic completion, archimedean factor, reflection, pole, branch, and zero-ledger rows.*

Definition 9293.6 (Completed xi execution trace)**.** *A completed xi execution trace is a PZG analytic completion certificate of Definition 9292.17 together with the completed readout*

$$\mathsf{XiRead}(s) := \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\mathsf{ZetaRead}(s)$$

and the mirror row

$$s \mapsto 1-s.$$

The trace is closed only to the extent that its continuation, completion, branch, pole, zero, tail, and semantic self-reference ledgers are closed.

Definition 9293.7 (Scalar readout kernel event)**.** *Let $\mathcal{A}$ be a displayed ledger carrier and let*

$$R : \mathcal{A} \to \mathbb{C}$$

be a scalar readout row. A scalar readout kernel event is a ledger state $a \in \mathcal{A}$ such that

$$R(a) = 0.$$

This event asserts vanishing in the scalar readout. It does not assert

$$a = 0$$

unless a separate faithfulness row for R is displayed.

Principle 9293.8 (Zeros are readout-kernel events)**.** *A completed zeta zero is a scalar readout kernel event for the completed xi execution trace. The assertion*

$$\mathsf{XiRead}(\rho) = 0$$

means that the completed analytic readout sends the represented ledger state at ρ to zero. It does not by itself say that the underlying source ledger, hidden fibre, prime-register residue, branch residue, or tail residue has vanished. Those rows are closed only by faithfulness, tail control, mirror cancellation, positivity, or explicit ledger closure data.

Definition 9293.9 (Prime exponent ledger group)**.** *The prime exponent ledger group is the finite-support additive group*

$$G_{\mathbb{P}} := \bigoplus_{p \in \mathsf{Prime}^{\uparrow}} \mathbb{Z}[p].$$

Its positive cone is the free commutative monoid

$$G_{\mathbb{P}}^{+} := \bigoplus_{p \in \mathsf{Prime}^{\uparrow}} \mathbb{N}[p],$$

which is the exponent ledger behind positive integers. Passing from $G_{\mathbb{P}}^{+}$ to $G_{\mathbb{P}}$ is the group-completion row that allows inverse ledger entries. The inverse of

$$\delta = (\delta_p)_p$$

is

$$-\delta = (-\delta_p)_p.$$

Definition 9293.10 (Positive prime ledger)**.** *The positive prime ledger is the free commutative monoid*

$$M_{\mathbb{P}} := \bigoplus_{p \in \mathsf{Prime}^{\uparrow}} \mathbb{N}[p].$$

Its element

$$a = \sum_p a_p[p]$$

reads the positive integer

$$n(a) = \prod_p p^{a_p}.$$

The PZG register is the canonical normal-form code of this ledger: each coordinate a_p is written in Zeckendorf normal form, producing the finite-support array $(z_{p,k})_{p,k}$ of Definition 9291.8.

Definition 9293.11 (Tagged PZG zeta family)**.** *For a parameter s on a licensed convergence or completion window, the tagged PZG zeta family is the labelled coefficient family*

$$\mathbf{Z}_{\mathsf{tag}}(s) := \left(a, e^{-sL(a)}\right)_{a \in M_{\mathbb{P}}}.$$

The tag a is part of the datum. Before a scalar readout is applied, two different prime ledgers remain different summands even when their complex coefficients have cancelling directions.

Theorem 9293.12 (Tagged zeta family is not the zero family)**.** *For every licensed parameter s, the tagged PZG zeta family is not the zero family.*

Proof. The empty prime ledger $0 \in M_{\mathbb{P}}$ satisfies

$$L(0) = 0.$$

Hence its tagged coefficient is

$$e^{-sL(0)} = 1.$$

A zero tagged family would have zero coefficient at every tag. The coefficient at the empty ledger is not zero, so the tagged family is not the zero family. □

Definition 9293.13 (Forgetful zeta scalar readout)**.** *The forgetful zeta scalar readout is the scalar row that drops the prime ledger tag and sums the displayed coefficients:*

$$\Pi_\zeta(\mathbf{Z}_{\mathsf{tag}}(s)) := \sum_{a \in M_{\mathbb{P}}} e^{-sL(a)}$$

whenever the convergence, continuation, completion, and tail rows license that sum. On the right-half-plane convergence window, this row is $\mathsf{ZetaRead}(s)$*.*

Theorem 9293.14 (Zeta zeros are forgetful-kernel events)**.** *On any licensed zero-read window, a zeta zero is a kernel event for the forgetful scalar readout, not a vanishing of the tagged PZG family.*

Proof. By Definition 9293.13, the scalar zeta value is obtained only after the prime ledger tags have been forgotten and the coefficients have been summed in the completed scalar target. Therefore a zero read has the form

$$\Pi_\zeta(\mathbf{Z}_{\mathsf{tag}}(\rho)) = 0.$$

This is a scalar readout kernel event in the sense of Definition 9293.7. By Definition 9293.12, the tagged PZG family itself is not zero. □

Principle 9293.15 (Projection cancellation is not source-ledger closure)**.** *Complex-plane cancellation after* Π_ζ *may be a polygonal cancellation among many tagged coefficients. Such cancellation does not identify their prime ledger tags and does not by itself pay source, branch, tail, or scale residuals. The fixed-half conclusion requires a separate closure row, such as the single-ledger condition of Definition 9294.10, proving that the zero packet closes inside its own zeta ledger rather than only after forgetting labels.*

Theorem 9293.16 (Positive prime ledgers have no nonzero self-cancellation)**.** *Let* $a, b \in M_{\mathbb{P}}$*. If*

$$a + b = 0,$$

then

$$a = 0 \qquad \text{and} \qquad b = 0.$$

Proof. The equality is coordinatewise. For each prime p,

$$a_p + b_p = 0$$

with $a_p, b_p \in \mathbb{N}$. Hence $a_p = b_p = 0$ for every p, so both finite-support ledgers are zero. □

Theorem 9293.17 (Prime-ledger local cancellation is inverse cancellation)**.** *Let $\delta, \eta \in G_{\mathbb{P}}$. If*

$$\delta + \eta = 0,$$

then

$$\eta = -\delta.$$

Proof. The group $G_{\mathbb{P}}$ is the direct sum of the integer coordinate groups indexed by primes. Therefore the equation

$$\delta + \eta = 0$$

is equivalent to the coordinate equations

$$\delta_p + \eta_p = 0 \qquad (p \in \mathsf{Prime}^{\uparrow}).$$

Thus $\eta_p = -\delta_p$ for every prime coordinate, hence

$$\eta = -\delta.$$

□

Corollary 9293.18 (Nonzero prime-ledger cancellation uses group completion)**.** *A nonzero positive prime ledger cannot be completely cancelled inside $M_{\mathbb{P}}$. Any complete cancellation of a nonzero prime-ledger residual requires the group-completion row*

$$M_{\mathbb{P}} \to G_{\mathbb{P}}$$

and the inverse element supplied by Definition 9293.17.

Proof. The no-cancellation statement inside $M_{\mathbb{P}}$ is Definition 9293.16. Inside $G_{\mathbb{P}}$, any element cancelling δ must be $-\delta$ by Definition 9293.17. □

Principle 9293.19 (Readout cancellation and ledger cancellation differ)**.** *Ledger cancellation is the structural equation*

$$\delta + (-\delta) = 0$$

inside a displayed ledger group such as $G_{\mathbb{P}}$. Scalar readout cancellation is the equation

$$\sum_i R_s(a_i) = 0$$

inside the target field of a readout. The latter says that the readout does not see the total contribution at the parameter s; it does not close the ledger unless the readout is faithful on the relevant residual class or every hidden component is closed by a ledger row.

Definition 9293.20 (Certified zero-search process)**.** *A certified zero-search process over Definition 9293.6 is a cofinal finite-window process*

$$W_0 \subset W_1 \subset W_2 \subset \cdots$$

whose stage j displays:

$$\Big(W_j, \quad \mathsf{FiniteApprox}_{\xi,W_j}, \quad \mathsf{Cand}_{W_j}, \quad \mathsf{TryCert}_{W_j}, \quad \mathsf{ZeroLedger}_{W_j}\Big).$$

Here $\mathsf{FiniteApprox}_{\xi,W_j}$ is the finite analytic readout visible in the window, Cand_{W_j} is the finite candidate region list, $\mathsf{TryCert}_{W_j}$ is the attempted winding, Rouché, argument-principle, or tail-bound certificate, and $\mathsf{ZeroLedger}_{W_j}$ records every unclosed boundary, branch, source, precision, and tail residue.

Principle 9293.21 (Infinite execution is finite reading plus tail control)**.** *The virtual machine does not process an infinite object by expanding all of its coordinates in one step. It processes infinity through cofinal finite reads, compatibility under refinement, tail bounds, and ledger closure. A finite read may become a global conclusion only through a certificate whose unobserved part is controlled by a displayed tail row. Without that row, the output remains a finite read or a residual ledger entry.*

Definition 9293.22 (Tail-budget ledger tower)**.** *A tail-budget ledger tower for an infinite readout is a cofinal finite window tower*

$$W_0 \subset W_1 \subset W_2 \subset \cdots$$

together with stage data

$$(\mathsf{L}_n, \mathsf{T}_n, B_n)_{n\geq 0}.$$

Here L_n *is the ledger visible at* W_n*,*

$$\mathsf{T}_n := W_\infty \setminus W_n$$

is the tail outside the displayed window, and B_n *is the budget row for the unread tail. A transition*

$$(\mathsf{L}_n, \mathsf{T}_n, B_n) \longrightarrow (\mathsf{L}_{n+1}, \mathsf{T}_{n+1}, B_{n+1})$$

is legal only when it transports the visible ledger, reallocates or reduces the tail budget, and records every residual not closed at the later window.

Definition 9293.23 (Budgeted infinite object)**.** *A budgeted infinite object is a triple*

$$X_\infty := ((X_B)_{B\in\mathcal{B}},\ (\rho_{B'B})_{B\subseteq B'},\ \mathcal{L}_\infty)\,.$$

Here $\mathcal{B}$ *is a cofinal family of finite budgets, each* X_B *is the finite readout available under budget* B*, each*

$$\rho_{B'B} : X_{B'} \to X_B$$

is the readback map from a larger budget to a smaller one, and $\mathcal{L}_\infty$ *is the ledger of residuals not closed by the displayed finite budgets.*

Principle 9293.24 (Infinity is budgeted tail-ledger execution)**.** *An infinite analytic object is not consumed by a single total read. In the virtual-machine semantics of Definition 9293.1, its processing consists of tail-budget transitions, compatibility under refinement, and ledger closure. The formal reading of*

$$\infty$$

in such a route is therefore a budgeted tail ledger: what has not been read at W_n *is not absent, but assigned to* T_n *with budget* B_n *and a required transport row into later stages.*

Theorem 9293.25 (Infinite processing factors through budgets and ledgers)**.** *Any auditable process that claims to handle a budgeted infinite object must display finite budget reads, compatibility readbacks, and a future ledger rule.*

Proof. An auditable process uses finite input data, finite precision, finite rules, and finite proof records at each stage. Thus its immediate output is a finite read X_B for some displayed budget B. If the process is to speak about the infinite object rather than only that stage, it must also show how a larger budget B' reads back to B by $\rho_{B'B}$ and how every residual not resolved at B is carried by $\mathcal{L}_\infty$. These are exactly the rows of Definition 9293.23. □

Theorem 9293.26 (Global closure requires budget exhaustion or ledger closure)**.** *Let a tail-budget ledger tower be used to promote finite reads to a global conclusion. The promotion is closed only if every tail component is either exhausted by the budgets or carried by a closed ledger row at the limit.*

Proof. At each stage, Definition 9293.22 assigns the unread part to the tail T_n and its budget B_n. A global conclusion consumes all stages of the tower. Any tail component that is neither exhausted by the budget row nor closed by the limit ledger is still an unread residual. Such a residual is not part of the certified finite read and has not been removed by the transition rules. Therefore the promoted conclusion is closed only when no such component remains. □

Theorem 9293.27 (Zero search factors through finite reads)**.** *Every auditable stage of a certified zero-search process factors through a finite zero read in the sense of Definition 9258.9.*

Proof. At stage j, the zero-search process displays a finite window, a finite analytic approximation, a finite candidate-region list, a finite certificate attempt, and a zero ledger. These supply exactly the region, precision or tolerance, finite count or zero-type readout, finite protocol, and error-boundary-branch-source-tail ledger fields of Definition 9258.9. Equivalently, this is an instance of the finite-read theorem Definition 9258.10. □

Theorem 9293.28 (Certified zeros require closure beyond search)**.** *A certified zero-search process may enumerate and certify finite zero regions. It presents a completed zero object only when the corresponding finite zero reads assemble into a compatible zero-certificate tower with closed boundary, branch, source, precision, and tail ledgers.*

Proof. By Definition 9293.27, every auditable search stage supplies finite zero reads. A completed zero is not identical to one such read; by Definition 9258.12, it is the closed readout selected by a compatible system of finite zero reads. The required compatible system is the zero-certificate tower of Definition 9257.13. Any field not closed by that tower remains in the zero ledger rather than becoming a completed zero object. □

Principle 9293.29 (RH proof is not zero enumeration)**.** *Zero search concerns located zero regions of the completed trace. The RH route requires a stronger global section certificate: every admitted non-trivial zero packet must factor through the fixed-half readout. Thus enumerating more zero certificates and proving the fixed-half section are different program rows over the same completed xi trace. The latter requires global positivity, self-adjoint, no-scale, or no-escape rows of the kind isolated by the ambient RH route, not merely a longer search ledger.*

Principle 9293.30 (Program code is an internal object)**.** *Any finite description of* $\mathsf{ZeroSearch}_\xi$ *is itself a finite syntactic packet and hence has a PZG address by Definition 9292.6. Therefore the search process, its finite stages, and its ledgers are internal objects of the same ontology virtual machine. Internality does not make the search complete: it only says that the code, run history, finite reads, certificates, and residual ledgers are generated and auditable by the same BEDC instruction surface.*

Chapter 9294

Zeta Mirror Cancellation and Local Closure

chapter 9293 separates ledger cancellation from scalar readout cancellation. This chapter records the corresponding mirror boundary for completed zeta zeros. Functional-equation symmetry supplies mirror-pair balance, while the fixed-half conclusion requires local zero-ledger closure at a single zero packet.

Definition 9294.1 (Weak and strong mirror closure)**.** *Let $J(s) = 1 - \overline{s}$ be the completed mirror row. A zero packet ρ has weak mirror closure when the mirror zero packet $J(\rho)$ is also present and the paired radial ledger balances across the orbit:*

$$\ell(\rho) + \ell(J(\rho)) = 0.$$

It has strong mirror closure when

$$\rho = J(\rho).$$

By the fixed-surface calculation of Definition 9241.12, strong mirror closure is the same as fixed-half readout.

Principle 9294.2 (Global cancellation is not local closure)**.** *Weak mirror closure is a global-orbit balance row. It says that an off-fixed residual at one zero can be paired with the inverse residual at the mirror zero. It does not close the ledger of the first zero as a local object. Local closure requires strong mirror closure, or another displayed row proving that the residual of the single zero is already zero at its own position. This is the ledger formulation of Definition 9241.14.*

Definition 9294.3 (Half-density scaling ledger)**.** *For a pure zeta input $a \in M_{\mathbb{P}}$ of Definition 9293.10, write*

$$L(a) = \sum_p a_p \log p.$$

For $s = \sigma + it$, set

$$\delta := \sigma - \frac{1}{2}$$

and define the half-density scaling ledger

$$\Lambda_s(a) := \delta L(a).$$

The centered zeta mode then factors as

$$e^{-sL(a)} = e^{-\frac{1}{2}L(a)} e^{-\Lambda_s(a)} e^{-itL(a)}.$$

The final factor is the unit-phase rotation. The middle factor is the positive real scaling ledger.

Theorem 9294.4 (Off-half scaling ledgers are same-signed and unbounded)**.** *If $\sigma \neq 1/2$, then every nonempty pure zeta input $a \in M_{\mathbb{P}}$ has $\Lambda_s(a) \neq 0$, all such ledgers have the same sign, and $|\Lambda_s(a)|$ is unbounded on $M_{\mathbb{P}}$.*

Proof. For a nonempty pure zeta input, the positivity of prime logarithms gives

$$L(a) > 0.$$

Thus

$$\Lambda_s(a) = (\sigma - \frac{1}{2})L(a)$$

has the sign of $\sigma - 1/2$ for every nonempty a, and it is nonzero when $\sigma \neq 1/2$. To see unboundedness, take the powers along the prime 2 axis,

$$a_m = m[2].$$

Then

$$|\Lambda_s(a_m)| = |\sigma - \frac{1}{2}|\, m \log 2,$$

which is unbounded as m ranges over $\mathbb{N}$. □

Definition 9294.5 (Zeta instruction-scale residual)**.** *For a centered zeta mode $s = 1/2 + r + it$ and a pure zeta input $a \in M_{\mathbb{P}}$, the instruction-scale residual is*

$$\mathsf{Scal}_s(a) := \log \left| e^{-rL(a)} \right| = -rL(a) = -\Lambda_s(a).$$

The unit-phase instruction attached to the same input is

$$U_t(a) := e^{-itL(a)}, \qquad |U_t(a)| = 1.$$

Theorem 9294.6 (Off-half scaling cancellation is not a zeta instruction)**.** *On the pure zeta instruction surface generated by positive prime registers and unit-phase rotations, no finite composite of phase rotations, conjugate phase rotations, or same-window phase mirrors cancels $\mathsf{Scal}_s(a)$ for a nonempty input a when $\Re(s) \neq 1/2$.*

Proof. Every admitted phase instruction has modulus 1. The same remains true after taking finite products, inverses, conjugates, and mirror phase reads, because these operations only change angular coordinates. Hence their log-modulus contribution is always

$$0.$$

For $s = 1/2 + r + it$, the non-phase part of the zeta monomial is

$$e^{-rL(a)}.$$

By Definition 9294.5, its log-modulus is $-rL(a)$. If a is nonempty, then $L(a) > 0$; if $r \neq 0$, this log-modulus is nonzero. Therefore unit-phase instructions cannot cancel the residual. Closing it requires either $r = 0$ or a displayed positive-real scale row outside the pure zeta instruction surface. □

Principle 9294.7 (Pure zeta has no local scaling inverse)**.** *The pure zeta input surface contains nonnegative prime registers and unit-phase rotations. It does not contain an independent negative length register, a positive-real scale counterweight, or a second same-position value of δ with opposite sign. Therefore an off-half scaling ledger cannot be cancelled inside the same pure zeta input. The only way for all half-density scaling ledgers to vanish locally is*

$$\sigma = \frac{1}{2}.$$

This is the zeta-specific form of the no-hidden-scale discipline of Definition 9290.13 and the unit-phase closure rule of Definition 9289.14.

Theorem 9294.8 (Pure zeta local closure is fixed-half closure)**.** *For a completed zero packet whose source is required to close inside the pure zeta input surface, local closure of the scaling ledger is equivalent to fixed-half readout.*

Proof. If the readout is fixed-half, then $\sigma = 1/2$, so Definition 9294.3 gives

$$\Lambda_s(a) = 0$$

for every pure zeta input a.

Conversely, assume the scaling ledger closes locally for every pure zeta input. Apply this to the one-prime input [2]. Since

$$L([2]) = \log 2 > 0,$$

the equality $\Lambda_s([2]) = 0$ forces $\sigma - 1/2 = 0$. Hence $\sigma = 1/2$, which is fixed-half readout. □

Corollary 9294.9 (Scaling-cancellation zeros are not pure zeta closures)**.** *A zero packet whose cancellation requires an off-half scaling ledger is not closed by the pure zeta input surface.*

Proof. If the packet is off-half, then Definition 9294.4 produces same-signed nonzero scaling ledgers, already unbounded along the 2-axis. By Definition 9294.7, pure zeta supplies no local scaling inverse for those ledgers. Thus such a packet can close only by adding an external scale counterweight or by using a distinct mirror position, neither of which is a pure zeta local closure. □

Principle 9294.10 (Single-ledger zero principle)**.** *A scalar zero read of the completed zeta carrier is accepted as a zeta-local zero only when the same zero packet carries the prime source ledger, phase readout, tail ledger, and scale-residual ledger needed to close the cancellation. A distinct mirror packet may witness global orbit symmetry, but it is not a same-packet payment of the scale residual. Thus an off-half scalar cancellation is not a pure zeta local closure unless a separate theorem first proves that its scale residual is closed inside the same zero ledger.*

Theorem 9294.11 (Single-ledger zeta closure excludes off-half scale)**.** *If a completed zero packet is closed by the single-ledger zero principle on the pure zeta instruction surface, then its scale residual vanishes. Consequently its readout is fixed-half.*

Proof. Single-ledger closure forbids payment of the scale residual by a distinct mirror position or by an undisplayed positive-real scale row. By Definition 9294.6, unit-phase zeta instructions cannot cancel a nonzero instruction-scale residual. Hence the residual can close on the pure zeta instruction surface only when

$$\mathsf{Scal}_s(a) = 0$$

for every pure zeta input a. Applying this to [2] gives

$$-\left(\Re(s)-\frac{1}{2}\right)\log 2=0,$$

so $\Re(s)=1/2$. □

Definition 9294.12 (Mirror-cancellation certificate)**.** *A mirror-cancellation certificate for the completed xi execution trace is a displayed tuple*

$$\mathsf{MCC}_\xi := (\mathcal{A}, \mathsf{Ledger}, \mathcal{R}_\xi, \Theta, \iota, (W_n)_{n\geq 0}, (B_n)_{n\geq 0}, Q, \mathsf{L}_{\mathrm{tail}}).$$

Here $\mathcal{A}$ is the ledger carrier, $\mathcal{R}_\xi$ is the completed scalar readout, Θ is the completed mirror row

$$s \mapsto 1-\overline{s},$$

ι is the ledger inverse row, (W_n) is a finite window tower, (B_n) is its tail-budget row, Q is a positive quadratic or Hilbert-square row, and $\mathsf{L}_{\mathrm{tail}}$ is the tail ledger. The certificate requires:

$$\iota^2=1, \qquad \delta+\iota(\delta)=0$$

on the displayed cancellation class;

$$\mathcal{R}_\xi \text{ sends admitted mirror pairs to scalar cancellation;}$$

the tail budgets are exhausted or closed by $\mathsf{L}_{\mathrm{tail}}$; and Q is nonnegative on the displayed residual class.

Theorem 9294.13 (Local zero-ledger closure gives fixed-half readout)**.** *Assume every completed non-trivial zero packet has local ledger closure: it cannot use a distinct mirror position to close its own radial residual. Then every such zero has fixed-half readout.*

Proof. Let ρ be a completed non-trivial zero packet. The functional equation and conjugation rows supply the mirror packet $J(\rho)$ as in Definition 9241.13. If $\rho \neq J(\rho)$, the residual balance is only weak mirror closure in the sense of Definition 9294.1: the radial ledger of ρ is offset by the radial ledger of a distinct position. By the local ledger-closure hypothesis, such cross-position balancing does not close the zero packet. A completed zero packet must therefore satisfy $\rho = J(\rho)$. The fixed-surface calculation in Definition 9241.12 gives $\Re(\rho)=1/2$, hence fixed-half readout. □

Principle 9294.14 (Mirror supplies symmetry, positivity locks the line)**.** *The mirror row by itself supplies paired structure: an off-fixed zero candidate may be reflected to its mirror companion. This symmetry is not yet the fixed-half conclusion. The fixed line is forced only when mirror cancellation is combined with a positive or self-adjoint row and with tail-ledger closure. In the notation of Definition 9294.12, the mirror row Θ explains the local form of cancellation, while Q and $\mathsf{L}_{\mathrm{tail}}$ refuse off-fixed residual escape. This is the same burden isolated by the global observation triangle of Definition 9272.1.*

Chapter 9295

Zeta Zero Readout Faithfulness

chapter 9293 separates scalar readout-kernel events from source-ledger closure, while chapter 9294 shows that single-ledger closure on the pure zeta instruction surface forces fixed-half readout. The remaining interface is the faithfulness of zero readout: whether a scalar completed zero is licensed as a closed PZG-ledger zero packet rather than merely as cancellation after a forgetful scalar projection.

Definition 9295.1 (Closed-ledger zeta zero)**.** *A closed-ledger zeta zero packet at a parameter ρ is a completed xi execution trace together with the scalar zero row*

$$\mathsf{XiRead}(\rho) = 0$$

and a displayed closure row

$$\mathsf{OpenLedger}_{\xi}(\rho) = \varnothing.$$

The open-ledger row includes source, branch, pole, tail, completion, prime-register, and scale-residual entries. Thus a closed-ledger zero is not merely a scalar readout-kernel event; it is a zero packet whose residual ledgers have been closed in the same completed zeta object.

Definition 9295.2 (Zero readout faithfulness)**.** *Zero readout faithfulness for the completed zeta route is the row*

$$\mathsf{ZeroReadoutFaithful}_{\xi}$$

asserting that every accepted non-trivial scalar zero read

$$\mathsf{XiRead}(\rho) = 0$$

comes equipped with a closed-ledger zeta zero packet in the sense of Definition 9295.1. Equivalently, the completed scalar readout may certify zero status only when the underlying source, tail, branch, completion, and scale ledgers are closed or explicitly transported into the same zero packet.

Principle 9295.3 (Zero visibility)**.** *The ontic zero set of the completed zeta route is not the scalar kernel of* XiRead *alone. It is the closed-ledger visibility locus*

$$\rho \in Z_{\mathrm{ontic}}(\xi) \Longleftrightarrow \mathsf{XiRead}(\rho) = 0 \quad \textit{and} \quad \mathsf{OpenLedger}_{\xi}(\rho) = \varnothing.$$

Thus the zero packet visible to the ontology kernel is a completed PZG-ledger zero. A scalar vanishing row with an open ledger remains a zero candidate, a finite-read zero-like event, or an incomplete cancellation row; it is not an ontic zeta zero.

Theorem 9295.4 (Open ledger blocks ontic zero)**.** *For any parameter ρ,*

$$\mathsf{OpenLedger}_\xi(\rho) \neq \varnothing \Longrightarrow \rho \notin Z_{\text{ontic}}(\xi).$$

Proof. If ρ belonged to $Z_{\text{ontic}}(\xi)$, then Definition 9295.3 would give the closure row

$$\mathsf{OpenLedger}_\xi(\rho) = \varnothing.$$

This contradicts the displayed nonempty ledger row. Hence the parameter is not visible as an ontic zeta zero. □

Principle 9295.5 (Unclosed zero reads are projection cancellation)**.** *Suppose the completed scalar readout displays*

$$\mathsf{XiRead}(\rho) = 0$$

while

$$\mathsf{OpenLedger}_\xi(\rho) \neq \varnothing.$$

Then the route has only a projection-cancellation row. Its zeta source side is the forgetful kernel event of Definition 9293.14, not a vanishing of the tagged PZG family, since Definition 9293.12 keeps the empty-ledger tag visible. By Definition 9293.15, the complex-plane cancellation after forgetting tags does not pay source, branch, tail, prime-register, or scale residuals. Therefore an unclosed scalar zero read is at most projection cancellation; by Definition 9295.4, it is not an ontic zeta zero.

Principle 9295.6 (Scalar zero is not automatically ledger closure)**.** *The implication*

$$\mathsf{XiRead}(\rho) = 0 \Longrightarrow \mathsf{OpenLedger}_\xi(\rho) = \varnothing$$

is not a definitional consequence of scalar arithmetic. By Definition 9293.14, a scalar zero is a kernel event after the forgetful readout. By Definition 9293.15, such projection cancellation does not by itself identify prime ledger tags or pay residuals. Therefore the implication is a faithfulness row to be displayed by the route.

Theorem 9295.7 (Faithful zero readout gives same-packet closure)**.** *Assume* $\mathsf{ZeroReadoutFaithful}_\xi$*. Then every accepted non-trivial scalar zero read of the completed xi trace satisfies the single-ledger zero principle.*

Proof. Let ρ be an accepted non-trivial scalar zero read. By Definition 9295.2, the read comes with a closed-ledger zeta zero packet. Its closure row records that source, branch, pole, tail, completion, prime-register, and scale-residual entries are closed inside the same completed zeta object. This is exactly the same-packet closure requirement of Definition 9294.10: the scalar zero read is accepted as zeta-local only when the same packet carries the ledgers needed to close the cancellation. □

Theorem 9295.8 (Faithful conservative zero readout gives fixed-half)**.** *Assume a conservative zeta continuation certificate and* $\mathsf{ZeroReadoutFaithful}_\xi$*. Then every accepted non-trivial scalar zero read of the completed xi trace has fixed-half readout.*

Proof. By Definition 9295.7, the zero read satisfies the single-ledger zero principle. Conservative continuation forbids scale-register payment by Definition 9311.7. The fixed-half conclusion is therefore Definition 9294.11, or equivalently the conservative continuation form Definition 9311.8. □

Principle 9295.9 (Mirror orbit is conditional visibility)**.** *Functional-equation symmetry may transport a scalar zero read from ρ to its mirror position $J(\rho)$, but the transport row does not license ρ as an ontic zero before ρ itself satisfies Definition 9295.3. In the terminology of Definition 9294.1, a non-fixed orbit supplies weak mirror closure. By Definition 9294.2, that orbit balance is not the same as local ledger closure for either endpoint. Thus an off-half mirror pair is only a conditional scalar symmetry unless each endpoint is already a closed-ledger zero packet.*

Theorem 9295.10 (Only same-position mirror closure can close an ontic zero)**.** *Assume a completed zero packet has no positive-real scale inverse outside the pure zeta instruction surface, and its prime-ledger residual can be paid only by an inverse row in the ledger group. If the packet is visible as an ontic zeta zero, then its mirror closure is same-position closure:*

$$\rho = J(\rho).$$

Proof. By Definition 9295.3, an ontic zero packet has no open ledger. By Definition 9293.18, a nonzero prime-ledger residual can be cancelled only by the inverse row in the group-completion ledger. If that inverse is supplied by a distinct mirror position, then the packet has weak mirror closure in the sense of Definition 9294.1. By Definition 9294.2, weak mirror closure does not close the ledger of either endpoint as a local object. The no-extra-scale hypothesis rules out a separate positive-real payment. Thus the ledger can be closed for the single ontic zero packet only when the mirror inverse is same-position, namely $\rho = J(\rho)$. □

Theorem 9295.11 (PZG closed zeros have fixed-half readout)**.** *Let*

$$\rho = \frac{1}{2} + a + it$$

be a parameter on the pure PZG zeta instruction surface. Assume a conservative zeta continuation certificate and

$$\rho \in Z_{\text{ontic}}(\xi).$$

Then $a = 0$.

Proof. By Definition 9295.3, the membership row $\rho \in Z_{\text{ontic}}(\xi)$ gives a closed-ledger zeta zero packet. By Definition 9295.1, its scale-residual row is among the rows closed inside the same completed zeta object. On the pure PZG zeta surface, a positive-real scale payment cannot be hidden: by Definition 9292.10, such a scale inverse changes the PZG identity code, appears in the ledger, or is not part of the object. The conservative continuation certificate also forbids analytic continuation from paying the residual by an independent scale register, by Definition 9311.7.

Hence the zero packet must close its scale-residual row by the same-packet discipline of Definition 9294.10 on the pure zeta instruction surface. Applying Definition 9294.11 gives

$$\Re(\rho) = \frac{1}{2}.$$

Since $\Re(\rho) = 1/2 + a$, it follows that $a = 0$. □

Corollary 9295.12 (Ontic zeta zeros lie on the fixed-half line)**.** *Under the pure PZG zeta source and conservative continuation hypotheses,*

$$Z_{\text{ontic}}(\xi) \subseteq \left\{ \frac{1}{2} + it : t \in \mathsf{Real}^{\uparrow} \right\}.$$

Proof. Apply Definition 9295.11 to each displayed ontic zero packet. □

Principle 9295.13 (Faithfulness is the remaining zero interface)**.** *After PZG identity, conservative continuation, and no scale-register payment have been fixed, the remaining interface is not the uniqueness of PZG coding. It is the passage from scalar zero readout to closed-ledger zero packet:*

$$\mathsf{XiRead}(\rho) = 0 \Longrightarrow \mathsf{ZeroReadoutFaithful}_{\xi}(\rho).$$

Without that row the route has a scalar kernel event. With that row the route has a closed zeta zero packet, and the fixed-half argument becomes the local ledger argument of Definition 9295.8; in ontic notation this is the passage from a scalar kernel event to $\rho \in Z_{\mathrm{ontic}}(\xi)$.

Chapter 9296

Prime-Fiber Zoom Renormalization Tower

chapter 9291 separates append-only history from orthogonal residual readout, and chapter 9292 assigns PZG coordinates a logarithmic energy. The present chapter records the renormalization surface that connects these two rows. A phase-relevant observation selects a prime-fiber cell, renormalizes that cell to the current visible surface, and records every unselected component in a gap ledger.

9296.1 Prime-Fiber Zoom Systems

Definition 9296.1 (Prime-fiber zoom system)**.** *A prime-fiber zoom system consists of a base fibre $\mathcal{F}_e$, finite address sets*

$$A_m \subseteq \mathsf{Prime}^{\uparrow} \times \mathbb{N},$$

cell families

$$\mathcal{P}_m := \{C_\alpha : \alpha \in A_m\},$$

selection rows

$$\mathrm{Sel}_\alpha : \mathcal{F}_e \to C_\alpha,$$

zoom rows

$$\mathrm{Zoom}_\alpha : C_\alpha \to \mathcal{F}_e,$$

and an orthogonal register basis $(e_\alpha)_\alpha$ such that different displayed addresses have orthogonal basis rows. A step with address α_m is the composite

$$T_{\alpha_m} := \mathrm{Zoom}_{\alpha_m} \circ \mathrm{Sel}_{\alpha_m}.$$

The unselected part of $\mathcal{F}_e$ is not discarded; it is exported to the gap row governed by Definition 9352.12.

Definition 9296.2 (Solenoid clopen prime cell)**.** *Fix a finite displayed prime family S and the solenoid phase channel*

$$\pi_S : \Sigma_S \to \mathbb{T}$$

of Definition 9286.1. Its legal hidden fibre is

$$K_S := \ker(\pi_S).$$

A solenoid clopen prime cell is a finite fibre-coordinate row

$$C(\mathbf{a},\mathbf{k}) := \prod_{p\in S}(a_p + p^{k_p}\mathbb{Z}_p)$$

inside a displayed product presentation of the finite prime-register window, where $k_p \geq 0$ and a_p is read modulo p^{k_p}. The cell records a finite prefix of the hidden prime-register state over the visible phase point.

Definition 9296.3 (Prime-fibre zoom row)**.** *For a solenoid clopen prime cell $C(\mathbf{a},\mathbf{k})$, the prime-fibre zoom row is the componentwise renormalization*

$$\mathrm{Zoom}_{\mathbf{a},\mathbf{k}} : C(\mathbf{a},\mathbf{k}) \to \prod_{p\in S}\mathbb{Z}_p$$

given by

$$\mathrm{Zoom}_{\mathbf{a},\mathbf{k}}((x_p)_{p\in S}) := \left(\frac{x_p - a_p}{p^{k_p}}\right)_{p\in S}.$$

Its inverse inclusion is

$$\mathrm{Cell}_{\mathbf{a},\mathbf{k}}((y_p)_{p\in S}) := (a_p + p^{k_p}y_p)_{p\in S}.$$

Thus a selected fibre cell is read as a whole prime-register window after renormalization.

Theorem 9296.4 (Solenoid clopen refinement gives an orthogonal filtration)**.** *Let*

$$C_0 \supseteq C_1 \supseteq \cdots \supseteq C_N$$

be a finite nested sequence of solenoid clopen prime cells over the same finite prime family S. Let $\mathcal{A}_m$ be the finite sigma-algebra generated by the stage-m cell partition of K_S, and set

$$\mathcal{H}_m := L^2(K_S, \mathcal{A}_m).$$

Then

$$\mathcal{H}_m \subseteq \mathcal{H}_{m+1}, \qquad \mathcal{H}_{m+1} = \mathcal{H}_m \oplus (\mathcal{H}_{m+1} \ominus \mathcal{H}_m).$$

Consequently, for

$$\eta_{m+1} \in \mathcal{H}_{m+1} \ominus \mathcal{H}_m$$

and $v_m \in \mathcal{H}_m$,

$$\|v_m + \eta_{m+1}\|^2 = \|v_m\|^2 + \|\eta_{m+1}\|^2.$$

Proof. Refinement of finite cell partitions gives $\mathcal{A}_m \subseteq \mathcal{A}_{m+1}$, hence $\mathcal{H}_m \subseteq \mathcal{H}_{m+1}$. The complement $\mathcal{H}_{m+1} \ominus \mathcal{H}_m$ is defined as the orthogonal complement inside the Hilbert row $\mathcal{H}_{m+1}$. The displayed direct sum and norm identity are therefore the Hilbert projection identity, specialized to the solenoid clopen-cell filtration. □

Definition 9296.5 (Zoom history tower)**.** *A zoom history tower is a finite or countable sequence of addresses*

$$\alpha_0, \alpha_1, \ldots \qquad (\alpha_j \in \mathsf{Prime}^{\uparrow} \times \mathbb{N})$$

with nested cells

$$C_{\alpha_0} \supseteq C_{\alpha_0\alpha_1} \supseteq C_{\alpha_0\alpha_1\alpha_2} \supseteq \cdots.$$

The history row is append-only:

$$h_{m+1} = h_m \alpha_m.$$

The visible row at stage $m+1$ is the zoomed cell

$$B_{m+1} := \mathrm{Zoom}_{\alpha_m}(B_m \cap C_{\alpha_m}),$$

while the Hilbert residual row is the address vector e_{α_m}.

Theorem 9296.6 (Prime-fiber zoom gives history, similarity, and orthogonality)**.** *In a prime-fiber zoom history tower, each displayed step has the three rows*

$$h_m \subseteq h_{m+1},$$

$$B_{m+1} = \mathrm{Zoom}_{\alpha_m}(B_m \cap C_{\alpha_m}),$$

and

$$e_{\alpha_m} \perp \mathrm{span}\{e_{\alpha_0}, \ldots, e_{\alpha_{m-1}}\}.$$

Consequently, if

$$R_m := \sum_{j<m} e_{\alpha_j}, \qquad R_{m+1} := R_m + e_{\alpha_m},$$

then

$$\|R_{m+1}\|^2 = \|R_m\|^2 + \|e_{\alpha_m}\|^2.$$

Proof. The history inclusion is the append-only clause of Definition 9296.5, read through the same history surface as Definition 9351.5 and Definition 9351.12. The displayed formula for B_{m+1} is the definition of the zoomed visible row. Orthogonality is part of the register basis in Definition 9296.1; hence the final identity is the Pythagorean identity for the decomposition $R_{m+1} = R_m + e_{\alpha_m}$. □

Principle 9296.7 (Prime-fiber zoom is not erasure)**.** *The zoom operation changes the current visible surface, but it does not license unledgered deletion. The selected cell is renormalized to the next visible row, the address α_m is appended to the history row, and the outside of the selected cell is represented by the gap ledger. Thus*

$$B_{m+1} \supseteq B_m, \qquad B_{m+1} \sim_{\exp} B_m, \qquad B_{m+1}/B_m \perp B_m$$

are not competing assertions. They are the history, zoom, and residual projections of one certified step.

9296.2 PZG Energy Compatibility

Definition 9296.8 (PZG zoom step)**.** *A PZG zoom step is a prime-fiber zoom step whose address is*

$$\alpha = (p, k)$$

and whose residual vector is the log-energy basis vector $\varepsilon_{p,k}$ of Definition 9292.1. Its squared residual energy is

$$\|\varepsilon_{p,k}\|^2 = F_k \log p.$$

Principle 9296.9 (PZG zoom carries preserve triangular energy)**.** *If two consecutive PZG zoom addresses in one prime row create a Zeckendorf adjacency, the legal normal form replaces*

$$\varepsilon_{p,k} + \varepsilon_{p,k+1} \rightsquigarrow \varepsilon_{p,k+2}.$$

By Definition 9292.3, this replacement preserves log-energy. A PZG zoom step is therefore not an arbitrary bit change in a hidden fibre; it is a prime-indexed residual update compatible with the triangular square-norm row.

9296.3 Analytic Prime-Shift Constraint

Definition 9296.10 (Analytic prime-shift tower)**.** *An analytic prime-shift tower is a prime-fiber zoom history tower together with finite holomorphic readouts*

$$F_m : \Omega_m \to \mathsf{Complex}^{\uparrow}$$

and ledgers $\mathcal{L}_m$ such that the address sequence

$$\alpha_0, \alpha_1, \dots \qquad (\alpha_m = (p_m, k_m))$$

is not merely a discrete stream of prime coordinates. It must satisfy the following certificate rows:

$$\mathsf{NestedCell}, \quad \mathsf{OrthogonalOrZeckendorfCarry}, \quad \mathsf{HeightControl},$$
$$\mathsf{HolomorphicChartCompatibility}, \quad \mathsf{DominantStrandCompatibility}, \quad \mathsf{RoucheNoWallCrossing}, \quad \mathsf{MonodromyLedger}.$$

The nested-cell row says that the selected cells form a refinement chain. The orthogonal-or-carry row says that each residual address is either orthogonal to the previous Hilbert row or is absorbed by a legal Zeckendorf carry. The height-control row supplies compact-uniform control for the exponential reads

$$\exp(-sH_m), \qquad H_m := \sum_{j \le m} \|e_{\alpha_j}\|^2,$$

or a displayed regularization producing the same compact-uniform holomorphic limit. The holomorphic-chart row supplies analytic continuation data in the sense of Definition 9341.11; the dominant-strand row records the branch selected by the tower; the Rouché row records zero-count stability on displayed boundaries; and the monodromy ledger records branch return data instead of treating it as invisible motion.

Definition 9296.11 (SmoothPrimeShiftConstraint)**.** *The obligation*

$$\mathsf{SmoothPrimeShiftConstraint}$$

asserts that every zero-relevant prime-fiber index stream used by the completed zeta route is an analytic prime-shift tower in the sense of Definition 9296.10. It therefore rules out an unconstrained sequence of addresses

$$\alpha_0, \alpha_1, \alpha_2, \dots$$

as a source for a holomorphic zeta observation. A legal stream must carry nesting, orthogonality or carry, height control, chart compatibility, branch ledger, no-wall-crossing, and cofinal coverage.

Theorem 9296.12 (Analytic prime-shift towers give holomorphic readouts)**.** *Let an analytic prime-shift tower have domains*

$$\Omega_m \subseteq \Omega_{m+1}, \qquad \Omega := \bigcup_m \Omega_m,$$

with holomorphic readouts F_m on Ω_m. Assume that the chart compatibility rows identify the restrictions on overlaps and that $F_m \to F$ uniformly on every compact subset of Ω. Then F is holomorphic on Ω, with branch and zero-island data controlled by the monodromy and Rouché ledgers of the tower.

Proof. On each compact subset $K \subseteq \Omega$, the compact-uniform convergence row gives uniform convergence of holomorphic functions on a neighbourhood of K. The Weierstrass theorem gives a holomorphic limit on that neighbourhood. The holomorphic-chart compatibility rows identify these local limits on overlaps by the analytic-continuation operation. The monodromy ledger records possible branch return data, so no branch selector is silently changed. The Rouché no-wall-crossing row records the boundary inequalities needed to preserve zero counts inside the displayed zero islands. These rows together give the claimed holomorphic readout on Ω. □

Principle 9296.13 (Prime shifts that are not analytic remain address streams)**.** *A PZG address stream that lacks the rows of Definition 9296.10 may still encode discrete prime-register information. It does not thereby generate a continuous curve, a differentiable observation, or a holomorphic zeta readout. Analytic operations such as eta continuation, functional-equation transport, contour transport, or explicit-formula transport are precisely the rows that convert a controlled prime-register stream into an analytic observation object.*

9296.4 Continuous Exponential Projection

Definition 9296.14 (Internal height and external exponential readout)**.** *The internal height of a PZG register is the additive square-norm row*

$$H(R) := \|R\|^2.$$

For the integer register $R_{\mathsf{PZG}}(n)$ of Definition 9292.1, this is

$$H(R_{\mathsf{PZG}}(n)) = \log n$$

by Definition 9292.2. The external exponential readout is the multiplicative row

$$\mathrm{Scale}(R) := \exp(H(R)).$$

The endpoint symbol **e** *denotes the identity endpoint or base fibre in the zoom system; the scalar* $\exp(1)$ *is the unit value of the continuous exponential readout.*

Definition 9296.15 (Character-valued exponential readout)**.** *Let A be a displayed additive ledger group, and let*

$$\omega : A \to (\mathbb{C}, +)$$

be an additive readout weight. The exponential readout determined by ω is

$$\mathrm{Exp}_\omega : A \to \mathbb{C}^\times, \qquad \mathrm{Exp}_\omega(x) := \exp(\omega(x)).$$

It is the external multiplicative observation of the internal additive coordinate. If $\omega(x) \in i\mathsf{Real}^\uparrow$ for every displayed x, then

$$\mathrm{Exp}_\omega(x) \in \mathbb{T}$$

and the readout is a unit-circle phase projection. If $\omega(x) \in \mathsf{Real}^\uparrow$ for every displayed x, then

$$\mathrm{Exp}_\omega(x) \in \mathsf{Real}^\uparrow_{>0}$$

and the readout is a radial-scale projection. Thus the same exponential mechanism separates into the two rows

$$\exp(a + ib) = \exp(a)\exp(ib), \qquad \exp(ib) \in \mathbb{T}, \qquad \exp(a) \in \mathsf{Real}^\uparrow_{>0}.$$

Theorem 9296.16 (Exponential readout is the normalized additive-to-multiplicative projection)**.** *Let*

$$\Phi : \mathsf{Real}^{\uparrow} \to \mathsf{Real}^{\uparrow}_{>0}$$

be a differentiable readout satisfying

$$\Phi(t+s) = \Phi(t)\Phi(s), \qquad \Phi(0) = 1, \qquad \Phi'(0) = 1.$$

Then

$$\Phi(t) = \exp(t).$$

Proof. Set

$$f(t) := \log \Phi(t).$$

The multiplicative law for Φ gives

$$f(t+s) = f(t) + f(s).$$

Differentiability gives $f(t) = ct$ for a constant c. Since

$$\Phi'(0) = f'(0)\exp(f(0)) = c,$$

the normalization $\Phi'(0) = 1$ gives $c = 1$. Hence $\Phi(t) = \exp(t)$. □

Principle 9296.17 (Pythagorean addition projects to exponential multiplication)**.** *If a PZG zoom step has*

$$R_{m+1} = R_m + e_{\alpha_m}, \qquad e_{\alpha_m} \perp R_m,$$

then

$$H(R_{m+1}) = H(R_m) + \|e_{\alpha_m}\|^2.$$

Under the external exponential readout,

$$\mathrm{Scale}(R_{m+1}) = \mathrm{Scale}(R_m)\exp(\|e_{\alpha_m}\|^2).$$

Thus the internal triangular square-norm row is additive, while its continuous scale projection is multiplicative.

Principle 9296.18 (Solenoid exponential readout is phase-valued)**.** *For the localized additive ledger*

$$A = \mathbb{Z}[S^{-1}],$$

the visible solenoid phase coordinate is read through unit characters of the form

$$q \mapsto \exp(2\pi i q\theta) \in \mathbb{T}.$$

This is the instance of Definition 9296.15 with

$$\omega_\theta(q) = 2\pi i q\theta.$$

The hidden row of Definition 9413.779 is the compact prime-register fibre K_S, not a positive real scale group. Therefore a real exponential row

$$q \mapsto \exp(aq), \qquad a \in \mathsf{Real}^{\uparrow},$$

is radial scale data rather than solenoid-hidden phase data, unless $a = 0$ on the zero-relevant packet.

Principle 9296.19 (Phase and radial reads use the same internal height)**.** *For a zeta monomial and* $s = \frac{1}{2} + r + it$*,*

$$n^{1/2-s} = \exp\big(-(r+it)\log n\big) = \exp(-rH(R_{\mathsf{PZG}}(n)))\exp(-itH(R_{\mathsf{PZG}}(n))).$$

Equivalently, in PZG coordinates,

$$n^{1/2-s} = \exp\left(-(r+it)\sum_{p,k} z_{p,k}F_k \log p\right).$$

The second factor is the unit-circle phase read; the first factor is the positive radial-scale read. Both consume the same internal PZG height. Only the unit-circle factor is a phase projection. A nontrivial radial factor is an external scale projection and must be ledgered as such.

Principle 9296.20 (RH reading of prime-fiber zoom)**.** *For a zeta phase row*

$$n^{-it} = \exp\left(-it\sum_{p,k} z_{p,k}F_k \log p\right),$$

the PZG coordinates are prime-fiber zoom addresses read through phase. For a centered zero packet with

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell+i\theta},$$

the angular coordinate θ *is compatible with unit phase readout, whereas a nonzero radial coordinate* ℓ *would have to appear as a ledgered residual, a gap row, or a refused scale coordinate. It cannot be absorbed into the visible wave without a prime-fiber address or a legal solenoid fibre row.*

Chapter 9297

Prime-Fiber Zoom Scale Obstruction Bridge

chapter 9296 records the local prime-fiber zoom step: history is appended, the selected cell is renormalized to the visible surface, the residual address is orthogonal to the previous register row, and the exterior of the selected cell is retained by a gap ledger. The present chapter records the scale-obstruction bridge needed to consume such steps in RH-facing towers. A legal zoom tower is not only a sequence of selected cells. It must also preserve every finite balanced correlation or reflection-positive obstruction that is invariant under its self-similar dilation rule.

9297.1 Self-Substitution Interface

Definition 9297.1 (PFZ self-substitution interface)**.** *A prime-fiber zoom tower carries a self-substitution interface when its address refinement row is equipped with a constant-length phase self-substitution row in the sense of Definition 9441.1:*

$$u_{Nn+a} = v_a u_n, \qquad a = 0, \dots, N-1.$$

The index n is the previous visible section, and a is the local digit selected inside the prime-fiber cell. The interface is Q3-compatible when

$$\sum_{a=0}^{N-1} v_a = \sqrt{N}.$$

Principle 9297.2 (Rule before visible section)**.** *In a PFZ tower, the invariant object is the repeated refinement rule, not one displayed visible section. A section records a finite layer; the rule records how each selected prime-fiber cell is expanded into the next visible surface. This is the RH-facing use of Definition 9441.2: scale invariants must be tested on the rule and its dilation tower, not only on a single finite window.*

Definition 9297.3 (PFZ balanced obstruction row)**.** *Let α be a balanced signed block and let*

$$\Gamma(\alpha)$$

be its correlation read in the self-substitution interface. A PFZ balanced obstruction row is either a non-real balanced read

$$\operatorname{Im}\Gamma(\alpha) \neq 0,$$

or a negative balanced read

$$\Gamma(\alpha) < 0.$$

It is scale-stable when the same obstruction is displayed for every dilated block $D_{N^k}\alpha$.

Theorem 9297.4 (PFZ balanced obstruction persists under dilation)**.** *In any PFZ tower carrying the self-substitution interface of Definition 9297.1, every PFZ balanced obstruction row is scale-stable:*

$$\Gamma(D_{N^k}\alpha) = \Gamma(\alpha) \qquad (k \geq 0).$$

Proof. This is exactly the balanced-correlation scale invariance of Definition 9441.4, read through the PFZ interface. The equality of reads gives persistence of the imaginary part and of the sign, as in Definition 9441.5. □

9297.2 Reflection-Positive Windows

Definition 9297.5 (PFZ reflection-centered dilation row)**.** *For odd N, the PFZ reflection-centered dilation row is the transported block dilation*

$$\mathcal{D}_N^\theta(\delta_j) := \delta_{\frac{1+N(2j-1)}{2}}$$

from Definition 9441.7. It is allowed in the PFZ tower only together with the reflection-commutation row

$$\theta\mathcal{D}_N^\theta = \mathcal{D}_N^\theta\theta$$

of Definition 9441.8.

Definition 9297.6 (PFZ RP obstruction row)**.** *Let $\mathcal{B}$ be a finite neutral reflection-positive window with matrix*

$$M_{\mathcal{B}}(v)_{ij} := \Gamma(\alpha_j - \theta\alpha_i).$$

A PFZ RP obstruction row is the presence of a negative direction in $M_{\mathcal{B}}(v)$. The obstruction is scale-stable when every reflected dilation

$$M_{(\mathcal{D}_N^\theta)^k\mathcal{B}}(v)$$

carries the same negative direction.

Theorem 9297.7 (PFZ RP obstruction persists under reflection-centered dilation)**.** *Assume N is odd. In a PFZ tower carrying the reflection-centered dilation row of Definition 9297.5, every PFZ RP obstruction row is scale-stable:*

$$M_{(\mathcal{D}_N^\theta)^k\mathcal{B}}(v) = M_{\mathcal{B}}(v) \qquad (k \geq 0).$$

Proof. Apply the odd-length RP window self-similarity theorem Definition 9441.12. Equality of the matrices preserves their spectra, so the negative direction is the same one recorded in Definition 9441.13. □

Principle 9297.8 (Finite obstruction is not scale noise)**.** *A finite balanced-correlation defect or finite RP negative direction cannot be dismissed as a local artefact of the selected cell. Once the PFZ tower carries the relevant self-substitution or reflection-centered dilation rule, the finite obstruction is copied by the rule itself. A legal zoom operation may move the visible surface, but it may not hide such an obstruction in the gap ledger, the solenoid fibre, or a coarser visible section.*

9297.3 Smooth Branch Gate

Definition 9297.9 (PFZ smooth branch gate)**.** *A local PFZ passage direction is admissible as a smooth branch only when it satisfies every displayed linear audit attached to the same window: the Q3 tangent row, the Hermitian or RP defect rows, the balanced correlation rows, and the gap-ledger compatibility row. Passing one projection is not enough to license the branch as a BEDC continuation object.*

Theorem 9297.10 (Pythagorean boundary has no antisymmetric PFZ passage)**.** *At the Pythagorean pentary row of Definition 9445.1, no nonzero mirror-antisymmetric tangent direction is admissible through the PFZ smooth branch gate.*

Proof. The Q3 tangent row alone allows the mirror-antisymmetric coordinates (x, y) by Definition 9445.2. The anchored three-link Hermitian audit then applies Definition 9445.5, which forces

$$x = 0, \qquad y = 0.$$

Thus every nonzero mirror-antisymmetric tangent fails the full PFZ branch gate. □

Principle 9297.11 (Branch admissibility is multi-projection)**.** *The PFZ tower may display many local digit choices, but a smooth continuation branch is not licensed by local choice alone. It must pass the phase, Hermitian, balanced-correlation, RP, and gap-ledger reads at the same time. The Pythagorean pentary boundary is the local model: phase-sum compatibility leaves apparent tangent directions, while the Hermitian reads remove the non-palindromic first-order directions.*

9297.4 Zeta-Facing Closure Obligations

Definition 9297.12 (PFZ closure envelope)**.** *The PFZ closure envelope is the union of all objects generated from the base endpoint fibre by legal prime-fiber zoom steps satisfying the rows of Definition 9296.5, the self-substitution interface of Definition 9297.1, the scale-obstruction rows of Definition 9297.3 and Definition 9297.6, and the smooth branch gate of Definition 9297.9. A member of the envelope therefore carries history containment, exponential self-similarity, orthogonal residual accounting, gap recording, and obstruction persistence as ledger rows.*

Definition 9297.13 (ZetaPFZEmbedding)**.** *The obligation*

$$\mathsf{ZetaPFZEmbedding}$$

asserts that the Dirichlet monomial rows, Euler product rows, analytic continuation rows, completed zero packets, and Li–Weil–Nyman projection rows used by the completed zeta route enter the PFZ closure envelope of Definition 9297.12. It is not the claim that the PFZ envelope proves RH; it is the source-admission claim that the zeta-facing objects are generated with their prime-fiber, phase, scale, and gap ledgers visible.

Definition 9297.14 (NoHiddenRadialPFZ)**.** *The obligation*

$$\mathsf{NoHiddenRadialPFZ}$$

asserts that a nonzero radial coordinate in a centered zero packet cannot be absorbed by PFZ zoom, solenoid phase projection, balanced-correlation dilation, reflection-centered RP dilation, or smooth branch selection. If the radial factor is present, it is a displayed positive-real scale row. If it is omitted, the packet has an unledgered residual and is not a legal PFZ closure object.

Principle 9297.15 (RH role of the PFZ obstruction bridge)**.** *For a completed zero packet with*

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta},$$

the phase coordinate θ may be read through PFZ phase and solenoid rows, while ℓ is a radial scale coordinate. Under ZetaPFZEmbedding*, the zero packet must enter the PFZ closure envelope. Under* NoHiddenRadialPFZ*, the case $\ell \neq 0$ cannot be hidden by zoom, dilation, reflection, or branch selection. The RH route must therefore supply these two obligations before the PFZ bridge can contribute to fixed-half readout.*

Chapter 9298

Prime-Fiber Zoom NameCert Interface

chapter 9296 records the local zoom step, and chapter 9297 records the scale-obstruction rows that must persist through the zoom tower. The present chapter records the BEDC admission surface for that package. A PFZ object is not licensed by the visible equation of a zoom step alone; it is licensed only as a five-field $\mathsf{NameCert}$ interface whose ledger policy keeps the unselected cell exterior, tail data, and radial-scale residuals auditable.

9298.1 NameCert Surface

Definition 9298.1 (PFZ NameCert surface)**.** *For an object B in the PFZ closure envelope of Definition 9297.12, the PFZ naming surface is the five-field packet*

$$\mathsf{NameCert}_{\mathsf{PFZ}}(B) := (\mathsf{SourceSpec}_B, \mathsf{PatternSpec}_B, \mathsf{ClassifierSpec}_B, \mathsf{StabilityCert}_B, \mathsf{LedgerPolicy}_B).$$

The fields are read as follows:

$$\mathsf{SourceSpec}_B = (\mathbf{e}, \mathcal{F}_{\mathbf{e}}, h),$$

where $\mathbf{e}$ is the endpoint fibre, $\mathcal{F}_{\mathbf{e}}$ is the unexpanded prime-fiber source, and h is the displayed address history;

$$\mathsf{PatternSpec}_B = \textit{the legal prime-fiber zoom refinement rule};$$

$$\mathsf{ClassifierSpec}_B = \textit{history, address, RP, Hermitian, phase, and radial classifiers};$$

$$\mathsf{StabilityCert}_B = \textit{history containment, exponential similarity, orthogonal residuals,}$$

together with balanced-correlation persistence, RP-window persistence, and smooth-branch rigidity; and

$$\mathsf{LedgerPolicy}_B = \textit{prime-address, gap, tail, phase, radial-scale, and branch ledgers.}$$

Theorem 9298.2 (PFZ objects enter BEDC only through NameCert)**.** *If a PFZ closure-envelope object B is used as a BEDC public object $B^{\uparrow}$, then its use factors through $\mathsf{NameCert}_{\mathsf{PFZ}}(B)$.*

Proof. By Definition 9352.17, a derived interface is admitted only through a naming certificate with source, pattern, classifier, stability, and ledger fields. The PFZ closure envelope is a derived interface: it hides selected-cell provenance, unselected-cell exterior data, residual addresses, tail data, and phase or radial readouts behind a visible zoom surface. Therefore object-layer use of $B^{\uparrow}$ requires the five fields displayed in Definition 9298.1. □

Principle 9298.3 (PFZ ledger policy is part of the object)**.** *The PFZ ledger policy is not an optional annotation after the zoom step. It is one of the fields that licenses the public name. Omitting it would turn the selected cell into an apparent whole, erase the exterior gap, and leave no audit path for later questions about phase, branch, tail, or radial scale. Thus a PFZ object without* $\mathsf{LedgerPolicy}_B$ *is not a BEDC-derived object in the sense of Definition 9352.16.*

9298.2 Ledger-Complete Limits

Definition 9298.4 (PFZ tower ledger)**.** *For a PFZ tower*

$$B_0 \to B_1 \to \cdots \to B_m \to \cdots ,$$

the PFZ tower ledger is the composite ledger obtained by recording at each step the selected address, the cell exterior gap, the residual orthogonal address, the phase readout, and any radial, branch, or tail row exposed by the step. The ledger of a composite tower is not the last step ledger alone; each later ledger is pulled back along the already visible interface, as in Definition 9414.315.

Definition 9298.5 (PFZ limit admission)**.** *A limit or tail object*

$$B_\infty = \lim_m B_m$$

is admitted as a PFZ public object only when the tower carries a PFZ tower ledger in the sense of Definition 9298.4, together with the convergence, branch, and tail rows required by the analytic consumer. The visible endpoint alone is not an admitted PFZ object.

Theorem 9298.6 (PFZ limit admission requires the tower ledger)**.** *If* B_∞ *is admitted as a PFZ public object, then its* $\mathsf{LedgerPolicy}_{B_\infty}$ *contains the PFZ tower ledger.*

Proof. By Definition 9298.5, an admitted PFZ limit must carry the composite tower ledger. The general ledger rule is Definition 9414.315: an endpoint ledger is not merely the final arrow's ledger, because the endpoint hides the preceding source histories and intermediate visible interfaces. Since $\mathsf{LedgerPolicy}_{B_\infty}$ is the ledger field of the PFZ NameCert surface, it must contain this composite tower ledger. □

Principle 9298.7 (No gap-free PFZ tail)**.** *A PFZ tail cannot be used as a place where exterior-cell information, branch changes, or radial scale disappear. If such data are relevant to a later zeta-facing readout, they must be present in the tower ledger. If they are absent, the public PFZ object has not been admitted for that readout.*

9298.3 Zeta-Facing Admission

Definition 9298.8 (Zeta PFZ NameCert admission)**.** *The zeta PFZ admission row is the strengthening of* $\mathsf{ZetaPFZEmbedding}$ *in which every zeta-facing PFZ object carries the NameCert surface of Definition 9298.1. Its source field contains Dirichlet monomial and PZG data; its pattern field contains Euler-product, continuation, zero, and Li–Weil–Nyman readouts; its classifier field contains zero, strip, phase, radial, and prime-register classifiers; its stability field contains analytic compatibility and the scale-obstruction bridge; and its ledger policy contains gap, branch, tail, phase, radial, and tower ledgers.*

Definition 9298.9 (NameCert-level radial refusal)**.** *The NameCert-level radial refusal is the PFZ admission form of* NoHiddenRadialPFZ*. For a centered zero packet with*

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta},$$

the coordinate ℓ is accepted only as one of the following:

$$\ell = 0, \qquad \text{a displayed radial-scale ledger row}, \qquad \text{or a refused unledgered residual.}$$

It is not accepted as unit phase, prime-address gauge, Zeckendorf carry, gap-free tail, or analytic-continuation branch artefact.

Theorem 9298.10 (PFZ NameCert package gives the radial-obstruction input)**.** *Assume the zeta PFZ admission row of Definition 9298.8 and the NameCert-level radial refusal of Definition 9298.9. Then every completed zeta zero packet admitted through this package supplies the PFZ radial-obstruction input required by Definition 9297.15.*

Proof. The zeta PFZ admission row supplies the zero packet as a PFZ public object with source, pattern, classifier, stability, and ledger fields. In particular, the radial coordinate is not an external diagnostic but a readout in the admitted packet, and the phase, radial, branch, tail, and gap ledgers are present in the ledger policy. The NameCert-level radial refusal then classifies a nonzero ℓ as a displayed scale row or as an unledgered residual that is not admitted. This is precisely the radial-obstruction input used by Definition 9297.15. □

Theorem 9298.11 (PFZ NameCert conditional fixed-half readout)**.** *Assume the zeta PFZ admission row, the NameCert-level radial refusal, and the radial-obstruction package sufficient for the RH route. Then every completed zero packet admitted by the package has fixed-half readout.*

Proof. By Definition 9298.10, the admitted zero packet supplies the PFZ radial-obstruction input. The radial-obstruction package then applies exactly as in Definition 9297.15: if $z(\rho) = e^{\ell + i\theta}$ had $\ell \neq 0$, that radial coordinate would be a displayed positive-real scale row or an unadmitted residual, not a unit-phase or prime-fiber row. The sufficient radial package rules out such a legal zero packet. Hence $\ell = 0$, and therefore the zero has fixed-half readout. □

Chapter 9299

Prime-Shift Analytic Compatibility

chapter 9296 defines analytic prime-shift towers and the $\mathsf{SmoothPrimeShiftConstraint}$ obligation. The present chapter records the sharper compatibility package used when the tower is meant to support analytic continuation rather than only a holomorphic limit of finite readouts.

9299.1 Four Compatibility Rows

Definition 9299.1 (PrimeShiftAnalyticCompatibility)**.** *Let a prime-fiber tower select clopen cells $C_n \subset K_S$ and finite holomorphic readouts*

$$F_n : \Omega_n \to \mathsf{Complex}^{\uparrow}.$$

The tower satisfies

$$\mathsf{PrimeShiftAnalyticCompatibility}$$

when it carries the following rows. First, nested cell compatibility:

$$C_{n+1} \subseteq C_n.$$

For a single p-adic coordinate with

$$C_k = a_k + p^k \mathbb{Z}_p,$$

this includes the digit condition

$$a_{k+1} \equiv a_k \pmod{p^k}.$$

Second, cocycle compatibility: the cell-renormalization maps compose with the displayed transition ledger, so refinement from level n to $n+2$ agrees with refinement through level $n+1$. Third, character-flow compatibility: there is one displayed frequency row ω whose induced one-parameter phase flow satisfies

$$\gamma(\tau_\omega(t)) = e^{it\ell_\omega(\gamma)}$$

for every displayed character γ consumed by the tower. Fourth, normal-convergence and no-radial-anomaly compatibility:

$$F_n \to F$$

uniformly on compact subsets of the analytic domain, the logarithmic derivative readouts converge on zero-free compact windows, and every transition factor is unit phase or compact prime-fibre data rather than an unledgered element of $\mathsf{Real}^{\uparrow}_{>0}$.

Principle 9299.2 (Compatibility separates paths from flows)**.** *An arbitrary prime address sequence gives only a discrete path. Nested cell compatibility gives a profinite limit point. Cocycle compatibility turns the refinements into one coherent tower. Character-flow compatibility turns the tower into a smooth phase readout. Normal convergence and no-radial-anomaly compatibility are the analytic rows: they are what allow the tower to be consumed by analytic continuation without importing a hidden positive-real scale.*

Principle 9299.3 (Compatibility is the continuation skeleton)**.** *Prime-shift compatibility is a necessary skeleton for analytic continuation, not a replacement for the continuation theorem attached to a concrete source. The nested, digit, cocycle, and character-flow rows rule out branch jumps and incoherent refinements; the normal-convergence and logarithmic-derivative rows supply compact analytic control; the no-radial-anomaly row prevents a positive-real scale from being absorbed into the solenoid fibre. A tower that lacks any one of these rows cannot serve as a BEDC continuation interface, while a tower that carries them still has to be identified with the intended analytic source by the appropriate source and operation certificates.*

Definition 9299.4 (ZetaAnalyticContinuationIsPrimeShiftCompatible)**.** *The obligation*

$$\mathsf{ZetaAnalyticContinuationIsPrimeShiftCompatible}$$

asserts that the eta continuation, functional-equation continuation, completed Gamma normalization, contour transport, and explicit-formula transport used by the completed zeta route are all realized through prime-shift towers satisfying Definition 9299.1. It is stronger than the existence of a meromorphic continuation: it requires that the continuation preserve the prime-register ledger, the character flow, the normal-convergence rows, and the no-radial-anomaly row.

Theorem 9299.5 (Prime-shift analytic compatibility refines smooth shift)**.** *If a prime-fiber tower satisfies* PrimeShiftAnalyticCompatibility*, then it satisfies the* SmoothPrimeShiftConstraint *rows required by Definition 9296.11.*

Proof. Nested cell compatibility supplies the refinement row. Cocycle compatibility supplies coherent transition ledgers. Character-flow compatibility supplies the single frequency row behind the visible phase readout. Normal convergence supplies the compact-uniform height and holomorphic readout control. The no-radial-anomaly clause supplies the ledger separation between unit phase, compact prime-fibre data, and positive-real scale. These are exactly the rows demanded by Definition 9296.11. □

Principle 9299.6 (RH-facing role of analytic compatibility)**.** *If a completed zero packet has centered mode*

$$U_a(s) = e^{-rh_a} e^{-ith_a},$$

then e^{-ith_a} is compatible with the character-flow row, while a nontrivial e^{-rh_a} is a positive-real transition factor. Under ZetaAnalyticContinuationIsPrimeShiftCompatible*, such a factor must be recorded as scale data or refused by the no-radial-anomaly row. It cannot be treated as a smooth prime shift inside the solenoid fibre.*

Chapter 9300

Smooth Prime Classifier Tower

chapter 9296 supplies analytic prime-shift towers, and chapter 9299 separates discrete address streams from smooth holomorphic readouts. The present chapter records the classifier side of the same interface. A prime shift that has become analytic must still remain visible to recursive prime classifiers; otherwise a positive-real radial coordinate could be mistaken for a harmless phase, carry, or branch effect. The role of the tower below is to make that mistake impossible at the finite classifier surface.

9300.1 Classifier Tower Surface

Definition 9300.1 (SmoothPrimeClassifierTower)**.** *A smooth prime classifier tower is a prime-fiber tower satisfying* PrimeShiftAnalyticCompatibility *together with a sequence of classifier packets*

$$\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \ldots$$

where

$$\mathcal{C}_m = (\Pi_m, \partial_m, U_m, \Lambda_m).$$

Here Π_m is the displayed prime-atom surface at level m, ∂_m is the triangle-boundary or obstruction readout, U_m is the unit-circle phase resource available at that level, and Λ_m is the ledger row transporting residuals from level m to level $m+1$. The transition

$$\mathcal{C}_m \to \mathcal{C}_{m+1}$$

is admitted only when it is sourced by a displayed obstruction or by a displayed refinement of the prime-register carrier.

Definition 9300.2 (Classifier readout of a packet)**.** *For an admitted PFZ packet X, the level-m classifier readout is the finite register expression*

$$\nu_m(X) = \sum_{\pi \in \Pi_m} v_\pi(X) e_\pi$$

together with the boundary row

$$\partial_m(X).$$

The packet is level-m triangle-closed exactly when

$$\partial_m(X) = 0$$

and every residual exposed by $\nu_m(X)$ is either a unit-circle phase row in U_m or a ledgered residual in Λ_m.

Principle 9300.3 (Classifiers refine only visible residues)**.** *The passage from $\mathcal{C}_m$ to $\mathcal{C}_{m+1}$ is not a free increase of resolution. It is licensed by a residue already visible at level m: a nonzero triangle boundary, a prime-register ambiguity, a character residual, a branch ledger row, or a scale row that has failed to be a unit phase. Thus recursive classification remains a ledger operation rather than a hidden appeal to an external arithmetic oracle.*

9300.2 Finite Classifier Separation

Definition 9300.4 (Finite prime-register residual)**.** *A finite prime-register residual is a non-identity element R in a finite abelian quotient G_m of the level-m prime-register surface. It is classifier-visible when some displayed character*

$$\chi : G_m \to S^1$$

satisfies

$$\chi(R) \neq 1.$$

Theorem 9300.5 (Finite character separation)**.** *Let G be a finite abelian group. If $g \neq e$, then there is a character*

$$\chi : G \to S^1$$

with

$$\chi(g) \neq 1.$$

Proof. Write G as a finite product of cyclic groups. Since $g \neq e$, at least one cyclic coordinate is nonzero. The standard root-of-unity character on that cyclic factor is nontrivial on the chosen coordinate, and extending it trivially across the other factors gives the required character of G. □

Corollary 9300.6 (Finite residuals cannot be hidden from classifiers)**.** *Every nonzero finite prime-register residual is detected by a finite character classifier.*

Proof. Apply Definition 9300.5 to the finite abelian quotient carrying the residual. □

9300.3 Cofinal Detection

Definition 9300.7 (Cofinal prime classifier tower)**.** *A smooth prime classifier tower is cofinal when its finite quotients form a Hausdorff profinite classifier system*

$$G = \varprojlim_m G_m$$

and every non-identity residual in G has non-identity image in at least one finite quotient G_m.

Theorem 9300.8 (Profinite residual separation)**.** *Let $G = \varprojlim_m G_m$ be a Hausdorff profinite group. If $g \neq e$ in G, then there is a finite quotient G_m whose image g_m satisfies*

$$g_m \neq e.$$

Proof. Since G is Hausdorff, there is an open neighbourhood of e not containing g. The open normal subgroups form a neighbourhood basis at e, so some open normal subgroup N excludes g. The quotient G/N is finite, and the image of g in G/N is not the identity. □

Definition 9300.9 (CofinalPrimeClassifierDetectsScale)**.** *The obligation*

$$\mathsf{CofinalPrimeClassifierDetectsScale}$$

asserts that every nonzero radial-scale residual of an admitted zeta-facing PFZ packet is represented as a non-identity residual in the cofinal classifier system. Equivalently, for every centered zero packet with

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta},$$

the case $\ell \neq 0$ supplies a residual whose image is detected by some finite classifier level:

$$\ell \neq 0 \quad \Longrightarrow \quad \exists m,\ \partial_m(\ell) \neq 0 \quad \text{or} \quad \exists m,\ \chi_m(\ell_m) \neq 1.$$

Theorem 9300.10 (Cofinal classifiers detect nonzero residuals)**.** *In a cofinal prime classifier tower, every non-identity residual is detected by a finite classifier level.*

Proof. Let R be a non-identity residual in the profinite classifier system. By Definition 9300.8, its image R_m is non-identity in some finite quotient G_m. By Definition 9300.5, a finite character on G_m separates R_m from the identity. This character is a finite classifier readout, so the residual is visible at level m. □

9300.4 Radial-Scale Refusal

Theorem 9300.11 (Smooth prime classifier tower excludes hidden radial scale)**.** *Assume a zeta-facing PFZ packet carries the NameCert admission row of Definition 9298.8, the analytic compatibility row of Definition 9299.1, and a cofinal smooth prime classifier tower satisfying Definition 9300.9. Then a nonzero radial coordinate cannot be consumed as a legal closed unit-phase readout.*

Proof. Let the centered zero packet have

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

The analytic compatibility row separates character flow from positive-real scale. Hence θ may enter the unit-circle phase resource, while $\ell \neq 0$ is a radial-scale residual. By Definition 9300.9, this residual has a nonzero finite classifier image. By Definition 9300.10, some finite classifier detects it. The detected residual is either recorded in the ledger, in which case it is not a closed unit-phase readout, or it is absent from the ledger, in which case the PFZ NameCert admission row is not satisfied. In neither case can $\ell \neq 0$ be consumed as legal closed phase. □

Corollary 9300.12 (Classifier-tower input for fixed-half readout)**.** *Assume the zeta PFZ admission row, the NameCert-level radial refusal, prime-shift analytic compatibility, and* CofinalPrimeClassifierDetectsScale*. Then every admitted completed zero packet supplies the classifier-tower input needed by the PFZ fixed-half readout route.*

Proof. The zeta PFZ admission row places the zero packet in the named PFZ surface. NameCert-level radial refusal identifies a nonzero ℓ as scale data or as an unadmitted residual. Prime-shift analytic compatibility prevents that scale data from being absorbed into the character-flow row. The cofinal classifier obligation then detects any remaining nonzero scale residual at a finite classifier level. This is exactly the radial-obstruction input consumed by Definition 9298.10. □

9300.5 Scale Faithfulness Boundary

Definition 9300.13 (Radial classifier footprint)**.** *Let X be an admitted zeta-facing PFZ packet with centered exponential readout*

$$\mathrm{ExpRead}(X) = e^{\ell(X)+i\theta(X)}.$$

A radial classifier footprint for X is a map

$$\kappa_X : \ell(X) \longmapsto R_\infty(\ell(X)) \in G$$

into the profinite classifier system of Definition 9300.7, together with finite images

$$R_m(\ell(X)) \in G_m$$

compatible with the transition maps of the tower. The footprint records where the positive-real scale coordinate is read by the prime-register classification surface; it is not a phase projection and not a Zeckendorf carry.

Definition 9300.14 (ScaleToPrimeClassifierFaithfulness)**.** *The obligation*

$$\mathsf{ScaleToPrimeClassifierFaithfulness}$$

asserts that every admitted zeta-facing radial classifier footprint is faithful on nonzero scale:

$$\ell(X) \neq 0 \quad \Longrightarrow \quad R_\infty(\ell(X)) \neq e_G.$$

It is the arithmetic bridge from a positive-real zero-packet coordinate to the cofinal prime-classifier tower. The profinite detection theorems do not create this bridge; they consume it.

Theorem 9300.15 (Scale faithfulness gives finite classifier detection)**.** *Assume X carries a radial classifier footprint in a cofinal prime classifier tower, and assume* ScaleToPrimeClassifierFaithfulness *for that footprint. If*

$$\ell(X) \neq 0,$$

then there are a finite level m and a character

$$\chi_m : G_m \to S^1$$

such that

$$\chi_m(R_m(\ell(X))) \neq 1.$$

Proof. By Definition 9300.14, $\ell(X) \neq 0$ implies

$$R_\infty(\ell(X)) \neq e_G.$$

The classifier tower is Hausdorff and profinite, so Definition 9300.8 gives a finite quotient G_m in which

$$R_m(\ell(X)) \neq e_{G_m}.$$

Applying Definition 9300.5 in G_m gives a character χ_m separating this finite image from the identity. □

Theorem 9300.16 (Detected radial residual cannot close)**.** *Let X be an admitted PFZ packet. If some finite classifier detects a radial residual,*

$$\chi_m(R_m(\ell(X))) \neq 1,$$

then X is not a gap-free unit-phase closure.

Proof. The displayed inequality is a finite residual row. If the row is entered into the ledger, then the packet still carries scale data and is not a unit-phase closure. If the row is absent from the ledger, then the PFZ NameCert admission row is violated, because the radial, phase, gap, branch, tail, and tower ledgers of Definition 9298.8 must account for such a consumer-facing residual. Thus the packet cannot be a gap-free unit-phase closure. □

Corollary 9300.17 (Scale faithfulness supplies cofinal scale detection)**.** *For any admitted zeta-facing PFZ packet carrying a faithful radial classifier footprint, the obligation* CofinalPrimeClassifierDetectsScale *holds for that packet.*

Proof. Unfold Definition 9300.9. The nonzero radial coordinate becomes a non-identity element of the profinite classifier system by Definition 9300.14. Finite detection then follows from Definition 9300.15. □

Principle 9300.18 (Arithmetic content of scale faithfulness)**.** *The remaining arithmetic content of* ScaleToPrimeClassifierFaithfulness *is the assertion that a nonzero radial zero-packet coordinate is not external to the prime surface. It must be read through Dirichlet log-height rows such as*

$$\log n = \sum_p v_p(n) \log p,$$

through explicit-formula scale terms, or through finite character/Frobenius classifiers. A radial coordinate that affects the zero packet but has no prime-classifier footprint is outside the PFZ admission surface rather than an undetected legal closure.

9300.6 Explicit-Formula Prime Residuals

Definition 9300.19 (Finite prime residual)**.** *A finite prime residual is a finite formal sum*

$$R = \sum_{j=1}^{N} c_j[q_j],$$

where $c_j \in \mathsf{Complex}^{\uparrow}$, $q_j \in \mathsf{Nat}^{\uparrow}$ with $q_j \geq 2$, and the q_j are pairwise distinct. It is nonzero when at least one displayed coefficient c_j is nonzero. The labels q_j may be primes or prime powers, matching the prime-power support of the explicit formula.

Definition 9300.20 (Finite Dirichlet classifier readout)**.** *Let $R = \sum_{j=1}^{N} c_j[q_j]$ be a finite prime residual. For a modulus M with $\gcd(q_j, M) = 1$ for all j, a finite Dirichlet classifier is a character*

$$\chi : (\mathbb{Z}/M\mathbb{Z})^{\times} \to S^1.$$

Its readout on R is

$$\chi(R) := \sum_{j=1}^{N} c_j \chi(q_j).$$

Theorem 9300.21 (Finite Dirichlet classifiers separate finite residuals)**.** *If $R = \sum_{j=1}^{N} c_j[q_j]$ is a nonzero finite prime residual, then there are a modulus M and a Dirichlet character χ modulo M such that*

$$\chi(R) \neq 0.$$

Proof. Choose a prime modulus M that divides none of the finitely many integers q_j and none of the nonzero differences $q_i - q_j$. Then the classes of the q_j are pairwise distinct units in

$$G = (\mathbb{Z}/M\mathbb{Z})^{\times}.$$

If every character $\chi : G \to S^1$ had

$$\sum_{j=1}^{N} c_j \chi(q_j) = 0,$$

then character orthogonality on the finite abelian group G would imply that the coefficient sum on each residue class is zero. Since the classes of the q_j are pairwise distinct, each c_j would be zero, contradicting that R is nonzero. Hence some Dirichlet character has nonzero readout on R. □

Definition 9300.22 (ExplicitFormulaScaleResidual)**.** *The obligation*

$$\mathsf{ExplicitFormulaScaleResidual}$$

asserts the following explicit-formula bridge. If a completed zeta zero packet contains an off-critical-line coordinate

$$\rho_0 = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

then there is an admissible test row φ whose explicit-formula readout has nonzero prime side after the archimedean and trivial-zero correction rows are accounted for:

$$\mathrm{PrimeSide}(\varphi) \neq 0.$$

Equivalently, the zero-side radial residual is not confined to the zero ledger; it has a prime-side ledger footprint.

Definition 9300.23 (Finite explicit-formula prime window)**.** *For an admissible test row φ, a finite explicit-formula prime window is a finite set S of prime powers together with the residual*

$$R_S(\varphi) := \sum_{q \in S} c_q(\varphi)[q],$$

where $c_q(\varphi)$ is the displayed prime-side coefficient attached to q. The window is nonzero when $R_S(\varphi)$ is a nonzero finite prime residual in the sense of Definition 9300.19.

Principle 9300.24 (Nonzero prime side has a finite window)**.** *If* $\mathrm{PrimeSide}(\varphi) \neq 0$ *and the prime-side sum is read through an absolutely convergent, unconditional, or explicitly regularized finite-window ledger, then some finite explicit-formula prime window is nonzero. Otherwise every finite window would have zero residual, and the displayed convergence or regularization ledger would force the whole prime-side readout to be zero.*

Theorem 9300.25 (Explicit-formula residuals give finite classifier detection)**.** *Assume* ExplicitFormulaScaleResidual *and the finite-window ledger of Definition 9300.24. If a completed zeta zero packet has $\ell(\rho_0) \neq 0$, then there is a finite prime residual $R_S(\varphi)$ and a finite Dirichlet classifier χ such that*

$$\chi(R_S(\varphi)) \neq 0.$$

Proof. By Definition 9300.22, the nonzero radial coordinate supplies an admissible test row φ with

$$\mathrm{PrimeSide}(\varphi) \neq 0.$$

The finite-window ledger gives a finite set S for which

$$R_S(\varphi) \neq 0.$$

Applying Definition 9300.21 to this finite residual gives a modulus and a Dirichlet character with nonzero readout. □

Corollary 9300.26 (Explicit formula supplies scale-to-classifier faithfulness)**.** *For a zeta-facing PFZ packet satisfying the explicit-formula residual obligation and the finite-window ledger, every nonzero radial coordinate has a finite prime-classifier footprint.*

Proof. The finite classifier footprint is the nonzero Dirichlet readout supplied by Definition 9300.25. It is finite, prime-side, and classifier-visible. Therefore the radial coordinate is not invisible to the prime classifier tower. □

Chapter 9301

Probability Amplitude Unit Geometry

chapter 9302 treats each classifier layer as a finite probability space. The present chapter records the geometric reason that such probability layers belong to the same BEDC triangle discipline as the prime-distribution curves: a probability law is the squared norm readout of a normalized amplitude. Consequently every sieve event is an orthogonal projection, and every discarded probability mass is a ledger row.

9301.1 Amplitude Readout

Definition 9301.1 (Finite probability amplitude)**.** *Let (G, μ) be a finite probability space with $\mu(a) > 0$ for every $a \in G$. If ν is another probability distribution on G with $\nu \ll \mu$, write*

$$r(a) := \frac{d\nu}{d\mu}(a).$$

A probability amplitude for ν over μ is a function

$$\psi(a) = \sqrt{r(a)}e^{i\theta(a)}$$

in $L^2(G, \mu)$. It satisfies

$$\|\psi\|^2_{L^2(\mu)} = \sum_{a \in G} |\psi(a)|^2 \mu(a) = \sum_{a \in G} r(a)\mu(a) = 1.$$

Principle 9301.2 (Probability is squared amplitude readout)**.** *The probability coordinates of a finite layer are not merely linear weights. They are the squared-norm readout of a unit vector in a finite Hilbert space:*

$$\nu(a) = |\psi(a)|^2 \mu(a), \qquad \|\psi\|^2 = 1.$$

Thus the probability simplex has a unit-sphere lift. The phases are invisible to the probability law but remain legal amplitude coordinates for later Fourier, character, or Frobenius readouts.

Corollary 9301.3 (Binary probability is a unit-circle readout)**.** *For a two-type probability law $(p, 1-p)$, there is $\theta \in [0, \pi/2]$ such that*

$$p = \cos^2 \theta, \qquad 1 - p = \sin^2 \theta.$$

Hence

$$p + (1 - p) = \cos^2\theta + \sin^2\theta = 1.$$

With phases, the lifted amplitude is

$$\psi = (e^{i\alpha}\cos\theta, e^{i\beta}\sin\theta),$$

a unit vector in $\mathbb{C}^2$.

Proof. Take $\theta = \arccos\sqrt{p}$. The displayed identities follow from the Pythagorean identity. □

9301.2 Event Projection

Definition 9301.4 (Event projection)**.** *For an event* $E \subseteq G$*, define*

$$P_E\psi := \mathbf{1}_E\psi, \qquad P_{E^c}\psi := \mathbf{1}_{E^c}\psi.$$

These are the pass and excluded components of the amplitude at the event boundary.

Theorem 9301.5 (Event projection Pythagorean law)**.** *For every finite probability amplitude* ψ *and event* $E \subseteq G$*,*

$$\psi = P_E\psi + P_{E^c}\psi, \qquad P_E\psi \perp P_{E^c}\psi,$$

and therefore

$$\|\psi\|^2 = \|P_E\psi\|^2 + \|P_{E^c}\psi\|^2.$$

In particular, if $\|\psi\| = 1$*, then*

$$1 = \nu(E) + \nu(E^c).$$

Proof. The supports of $\mathbf{1}_E$ and $\mathbf{1}_{E^c}$ are disjoint, so

$$\langle P_E\psi, P_{E^c}\psi\rangle = \sum_{a\in G} \mathbf{1}_E(a)\mathbf{1}_{E^c}(a)|\psi(a)|^2\mu(a) = 0.$$

The norm identity is the Hilbert-space Pythagorean theorem. Since

$$\|P_E\psi\|^2 = \nu(E), \qquad \|P_{E^c}\psi\|^2 = \nu(E^c),$$

the final identity follows. □

Corollary 9301.6 (Sieve events are orthogonal triangle splits)**.** *At a layer of* LayeredPrimeProbabilitySieve*, selecting* $E_m \subseteq G_m$ *gives the orthogonal split*

$$1 = \|P_{E_m}\psi_m\|^2 + \|P_{E_m^c}\psi_m\|^2.$$

Thus every probability sieve event is a triangle split in the amplitude space of that layer.

9301.3 Ledger Consequence

Definition 9301.7 (Renormalized pass amplitude)**.** *If $\|P_E\psi\| > 0$, the renormalized pass amplitude is*

$$\psi_E := \frac{P_E\psi}{\|P_E\psi\|}.$$

The excluded probability mass is

$$\|P_{E^c}\psi\|^2 = 1 - \nu(E).$$

Theorem 9301.8 (Discarded probability mass is ledger data)**.** *If a layer keeps only the renormalized pass amplitude ψ_E, then the excluded mass*

$$\|P_{E^c}\psi\|^2$$

must be recorded as the excluded-mass coordinate of Definition 9302.3. Otherwise the operation is not a BEDC legal projection.

Proof. The full amplitude decomposes orthogonally into pass and excluded components. Replacing ψ by ψ_E erases the excluded component unless its squared norm is recorded. BEDC projection operations cannot discard generated coordinates without a ledger row; hence the excluded mass must be stored in the sieve ledger. □

Principle 9301.9 (Probability has the BEDC triangle properties)**.** *Finite probability layers have the three BEDC triangle properties. They are unitary because their amplitudes have norm one; they are orthogonal because event partitions are Hilbert projections; and they are ledger-bound because renormalization after a sieve event exposes an excluded squared-norm component. Probability is therefore a unit-sphere readout, not an unstructured scalar bookkeeping device.*

Theorem 9301.10 (Amplitude geometry supports cofinal sieve dissipation)**.** *The detected-subspace projections used in Definition 9302.13 are the residual counterparts of event projections in finite probability amplitude space. Consequently the energy descent theorem Definition 9302.14 is a direct application of the same Pythagorean law that governs probability events.*

Proof. Both constructions are orthogonal projection decompositions in a finite Hilbert space. For events, the projection is multiplication by $\mathbf{1}_E$. For residuals, it is the orthogonal projection onto the detected subspace W_m. In each case, the squared norm splits into the detected component plus its orthogonal complement. The residual energy descent is therefore the iterative form of the event Pythagorean law. □

Chapter 9302

Layered Prime Probability Sieve

chapter 9300 records the finite and profinite classifier tower that detects prime-register residuals. The present chapter records the probability ledger carried by that tower. Each layer classifies primes, assigns Chebyshev-weighted probability mass to the displayed types, and exports normalized distribution curves. Thus a sieve step is not deletion of primes. It is conditional projection of prime-register mass, with the excluded mass recorded as ledger data.

9302.1 Probability Layers

Definition 9302.1 (Prime probability classifier)**.** *A prime probability classifier is a finite prime projection classifier*

$$\mathcal{C}_m = (G_m, \pi_m)$$

together with a Chebyshev-weighted probability distribution μ_m on G_m. For a displayed type $a \in G_m$, the intended readout is

$$\mu_m(a) := \lim_{x\to\infty} \frac{1}{x} \sum_{\substack{p^k \le x \\ \pi_m(p^k)=a}} \log p$$

when the limit exists. In Chebotarev layers, μ_m is the conjugacy-class density; in Dirichlet layers, it is the residue-class or character density supplied by the relevant equidistribution theorem.

Definition 9302.2 (LayeredPrimeProbabilitySieve)**.** *The row*

$$\mathsf{LayeredPrimeProbabilitySieve}$$

is an inverse system

$$(G_0, \mu_0) \leftarrow (G_1, \mu_1) \leftarrow (G_2, \mu_2) \leftarrow \cdots$$

of prime probability classifiers. If

$$\pi_{m+1,m} : G_{m+1} \to G_m$$

is the refinement map, compatibility means

$$\mu_m(a) = \sum_{\pi_{m+1,m}(b)=a} \mu_{m+1}(b)$$

for every displayed type $a \in G_m$.

Definition 9302.3 (Sieve ledger row)**.** *Let $E_m \subseteq G_m$ be the set of types allowed by a layer. The sieve ledger row is*

$$\mathrm{SieveLedger}_m := (E_m, \mu_m(E_m), G_m \setminus E_m, 1 - \mu_m(E_m), \mathrm{reason}_m)\,.$$

The conditional layer distribution is

$$\mu_m(a \mid E_m) := \frac{\mu_m(a)}{\mu_m(E_m)} \qquad (a \in E_m,\ \mu_m(E_m) > 0).$$

Thus a layer exposes the allowed mass and the excluded mass as finite ledger coordinates.

Principle 9302.4 (Sieve steps are probability projections)**.** *A layer of* LayeredPrimeProbabilitySieve *does not remove primes from the arithmetic source. It records a classifier, a probability measure on classifier types, a selected admissible type region, and the mass outside that region. The operation is therefore a BEDC ledger projection, not an untracked modification of the Euler source.*

9302.2 Probability Distribution Curves

Definition 9302.5 (Probability-filtered Chebyshev curve)**.** *Let (G_m, π_m, μ_m) be a prime probability classifier and let*

$$\varphi : G_m \to \mathbb{C}$$

be a displayed test function. Define

$$\psi_{m,\varphi}(x) := \sum_{p^k \le x} \varphi(\pi_m(p^k)) \log p.$$

The layer mean is

$$\mathbb{E}_{\mu_m}[\varphi] := \sum_{a \in G_m} \varphi(a)\mu_m(a),$$

and the critically normalized probability curve is

$$\Psi_{m,\varphi}(u) := e^{-u/2}\left(\psi_{m,\varphi}(e^u) - \mathbb{E}_{\mu_m}[\varphi]e^u\right).$$

Definition 9302.6 (Critical probability stability)**.** *A probability-filtered curve $\Psi_{m,\varphi}$ is critically stable when its local energy*

$$\mathcal{E}_{m,\varphi}(U) := \int_U^{U+1} |\Psi_{m,\varphi}(u)|^2\, du$$

satisfies

$$\limsup_{U\to\infty} \frac{1}{U} \log \mathcal{E}_{m,\varphi}(U) \le 0.$$

Theorem 9302.7 (Critical probability stability excludes radial modes)**.** *If every legal probability-filtered curve in a layered prime probability sieve is critically stable, then no such curve contains a radial spectral mode*

$$c\, e^{ru} e^{i\gamma u}, \qquad c \neq 0, \quad r \neq 0.$$

Proof. If $r > 0$, the displayed mode contributes exponential energy on unit windows:

$$\mathcal{E}_{m,\varphi}(U) \geq Ce^{2rU}$$

after lower-order oscillatory terms are absorbed on a subwindow where the selected mode remains separated. Hence

$$\limsup_{U\to\infty} \frac{1}{U} \log \mathcal{E}_{m,\varphi}(U) \geq 2r > 0,$$

contradicting critical probability stability. If $r < 0$, the mirror zero packet carries the corresponding positive radial leg with exponent $-r > 0$, so the same argument applies to the mirror readout. □

9302.3 Completeness and Detection

Definition 9302.8 (Layered probability sieve completeness)**.** *The row*

LayeredProbabilitySieveCompleteness

asserts that every nonzero radial residual of a zeta-facing PFZ packet is visible on some probability-filtered curve. Equivalently, if

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell+i\theta}$$

has $\ell \neq 0$, then there are a layer m and a mean-zero test function

$$\varphi : G_m \to \mathbb{C}, \qquad \mathbb{E}_{\mu_m}[\varphi] = 0,$$

such that $\Psi_{m,\varphi}$ contains the corresponding radial mode.

Theorem 9302.9 (Finite residuals are detected by probability tests)**.** *Let*

$$R = \sum_{j=1}^{N} c_j[q_j] \neq 0$$

be a finite prime-power residual with distinct q_j. Then there are a finite probability classifier and a mean-zero test function φ whose readout on R is nonzero:

$$\sum_{j=1}^{N} c_j\varphi(q_j) \neq 0.$$

Proof. Choose a prime modulus M so that all q_j are nonzero and pairwise distinct in

$$G = (\mathbb{Z}/M\mathbb{Z})^\times.$$

Take M large enough that some residue class $b \in G$ is not among the classes of the q_j. If every mean-zero test function on G annihilated R, then for the class a_j of any q_j the function

$$\varphi_j := \mathbf{1}_{a_j} - \frac{\mu_m(a_j)}{\mu_m(b)}\mathbf{1}_b$$

would also annihilate R. It has mean zero, while its readout on R is exactly c_j, because no residual term lies in the class b. Hence every c_j would vanish, contradicting $R \neq 0$. □

Theorem 9302.10 (Profinite probability residuals descend to finite layers)**.** *In a Hausdorff profinite probability classifier tower, every nonzero profinite residual has a nonzero finite-layer probability test readout.*

Proof. By Definition 9300.8, the residual has a nonzero image in some finite quotient. Apply Definition 9302.9 at that finite layer. □

Theorem 9302.11 (Layered probability completeness supplies distribution-curve completeness)**.** *If* LayeredProbabilitySieveCompleteness *holds, then the visible curve required by*

PrimeDistributionCurveCompleteness

is available as a probability-filtered curve.

Proof. The probability-filtered curve of Definition 9302.8 is a filtered prime-distribution curve in the sense of Definition 9316.3, with the layer mean replacing the trivial-character main term. Thus it supplies the visible radial mode demanded by Definition 9316.8. □

9302.4 Cofinal Dissipation

Definition 9302.12 (Layer residual Hilbert space)**.** *For a prime probability classifier* (G_m, μ_m)*, define*

$$H_m := L_0^2(G_m, \mu_m)$$

to be the finite-dimensional Hilbert space of mean-zero functions on G_m*, with inner product*

$$\langle f, g\rangle_m := \sum_{a\in G_m} f(a)\overline{g(a)}\mu_m(a).$$

A layer residual is an element $R_m \in H_m$*.*

Definition 9302.13 (Detected subspace and residual update)**.** *At layer* m*, let* $W_m \subseteq H_m$ *be the subspace detected by the legal sieve tests of that layer, and let*

$$\Pi_m : H_m \to W_m$$

be the orthogonal projection. The residual update is

$$R_m := (I - \Pi_m)R_{m-1},$$

after transporting R_{m-1} *to the current layer through the displayed refinement map.*

Theorem 9302.14 (Layer sieve energy descent)**.** *Assume*

$$\|\Pi_m R_{m-1}\|_m^2 \geq \eta_m \|R_{m-1}\|_m^2$$

for some $\eta_m \in (0, 1]$*. Then*

$$\|R_m\|_m^2 \leq (1 - \eta_m)\|R_{m-1}\|_m^2.$$

Consequently,

$$\|R_m\|_m^2 \leq \left(\prod_{j=1}^{m}(1 - \eta_j)\right)\|R_0\|_0^2.$$

If

$$\sum_{j=1}^{\infty} \eta_j = \infty,$$

then $\|R_m\|_m \to 0$*.*

Proof. The projection decomposition is orthogonal:

$$R_{m-1} = \Pi_m R_{m-1} + (I - \Pi_m) R_{m-1}.$$

Thus

$$\|R_{m-1}\|_m^2 = \|\Pi_m R_{m-1}\|_m^2 + \|R_m\|_m^2.$$

The displayed lower bound on the detected part gives

$$\|R_m\|_m^2 \le (1 - \eta_m)\|R_{m-1}\|_m^2.$$

Iteration gives the product estimate. Finally,

$$\log(1 - \eta_j) \le -\eta_j$$

for $\eta_j \in (0, 1]$, so the product tends to 0 when $\sum_j \eta_j$ diverges. □

Definition 9302.15 (CofinalSieveDissipation)**.** *The row*

$$\mathsf{CofinalSieveDissipation}$$

asserts that every nonzero radial residual R_0 admits a legal sequence of detected subspaces W_m and constants $\eta_m \in (0, 1]$ such that

$$\sum_m \eta_m = \infty.$$

Equivalently, the layered probability sieve has enough cumulative detection mass to force the residual norm to vanish along the tower.

Theorem 9302.16 (Cofinal sieve dissipates radial residuals)**.** *If* CofinalSieveDissipation *holds, then every nonzero radial residual is exhausted by the layered probability sieve:*

$$\|R_m\|_m \to 0.$$

Proof. Apply Definition 9302.15 to the initial residual and then use Definition 9302.14. □

Theorem 9302.17 (Cofinal probability sieve excludes hidden radial residuals)**.** *Assume every nonzero radial mode supplies a nonzero initial residual,* CofinalSieveDissipation *holds, and every layer records its sieve ledger row. Then a legal closed BEDC/PFZ zero packet cannot carry a hidden radial residual.*

Proof. Let $R_0 \neq 0$ be the residual attached to a radial mode. By Definition 9302.16, the sieve tower exhausts this residual. If the exhausted mass is recorded in the ledger, the history continues through displayed append rows and the packet is not closed. If it is not recorded, the sieve ledger row of Definition 9302.3 is violated. Hence the residual cannot be hidden inside a legal closed packet. □

9302.5 Conditional RH Interface

Theorem 9302.18 (Layered probability sieve conditional fixed-half readout)**.** *Assume zeta-facing PFZ zero packets are legal closure objects,* LayeredProbabilitySieveCompleteness *holds, every legal probability-filtered curve is critically stable, and all sieve ledger rows record their allowed mass, excluded mass, and residual readouts. Then every admitted nontrivial zero has fixed-half readout.*

Proof. If an admitted nontrivial zero had $\ell \neq 0$, layered probability sieve completeness would provide a legal probability-filtered curve with a radial mode. By Definition 9302.7, that curve would not be critically stable. This contradicts the stability hypothesis for legal filtered curves. Hence $\ell = 0$, and Definition 9266.2 gives fixed-half readout. □

Theorem 9302.19 (Cofinal probability sieve conditional fixed-half readout)**.** *Assume zeta-facing PFZ zero packets are legal closure objects, every off-critical zero supplies a nonzero radial residual,* CofinalSieveDissipation *holds, all sieve ledger rows are recorded, and every legal probability-filtered curve is critically stable. Then every admitted nontrivial zero has fixed-half readout.*

Proof. An off-critical admitted zero would supply a nonzero radial residual. By Definition 9302.17, that residual cannot be hidden inside a legal closed BEDC/PFZ zero packet. If the residual remains visible on a legal filtered curve, then Definition 9302.7 contradicts critical probability stability. Thus the off-critical case is excluded, and Definition 9266.2 gives fixed-half readout. □

Principle 9302.20 (Probability sieve audit boundary)**.** *The probability-sieve formulation concentrates the RH-facing burden in three rows. The completeness row says every nonzero radial residual is visible on some finite or profinite probability-filtered prime curve; the dissipation row says cofinal layer projections exhaust every nonzero residual with divergent detected energy; the stability row says every legal filtered curve has no exponential radial energy growth after critical normalization. Together they make an off-critical zero incompatible with a closed BEDC/PFZ zero packet.*

Chapter 9303

Zeta as a Certified Inverse Limit

chapter 9301 and chapter 9302 put the probability layers of the prime classifier tower into unit-amplitude form. The present chapter records the global proof-engineering interface: completed zeta is treated as a certified inverse-limit object assembled from licensed finite layers, not as an unlicensed primitive at the kernel. The probability numbers attached to layers are therefore coverage measures for audited candidate space, not probabilities that a theorem is true.

9303.1 Ontological Rows

Definition 9303.1 (Ontological triple)**.** *For a proof object, tool, or carrier X, its ontological triple is*

$$\mathrm{Ont}(X) := (\mathsf{T}_X, \mathsf{U}_X, \mathsf{L}_X).$$

Here T_X is the time-source row, recording the lower-level objects from which X is generated and whether the history is append-only. The row U_X is the unit or triangle row, recording unit normalization, orthogonal splitting, positivity, or Pythagorean energy conservation. The row L_X is the ledger row, recording hidden gap, tail, phase, classifier, or probability-mass data.

Principle 9303.2 (All zeta-route tools require ontological triples)**.** *Every tool used by the RH route must expose its ontological triple. A finite classifier exposes prime-source time, unit character or amplitude readout, and type-probability ledger. A fixed-window analytic tool exposes finite construction time, unit or positive boundary data, and a tail ledger. A limiting or completion tool exposes the lower-layer inverse system, the preserved unit or positivity invariant, and a limit-ledger policy. A row without these three coordinates is not available as a BEDC proof object for the route.*

Definition 9303.3 (Certified zeta layer)**.** *A certified zeta layer is a finite packet*

$$\mathcal{Z}_m := (G_m, \mu_m, \Psi_{m,\varphi}, \mathrm{SieveLedger}_m, \mathrm{Gap}_m, \mathrm{DomCert}_m, \mathrm{NameCert}_m)\,,$$

where (G_m, μ_m) is a prime probability classifier, $\Psi_{m,\varphi}$ is a normalized filtered prime-distribution curve, the sieve ledger records allowed and excluded probability mass, the gap row records the unexcluded candidate region, the dominance certificate records the fixed-window analytic row available at that layer, and the naming certificate licenses the layer as a BEDC object.

Definition 9303.4 (Certified zeta inverse system)**.** *A certified zeta inverse system is a projective family*

$$\mathcal{Z}_0 \leftarrow \mathcal{Z}_1 \leftarrow \mathcal{Z}_2 \leftarrow \cdots$$

whose transition maps preserve the classifier projection, probability pushforward, sieve ledger, gap inclusion, dominance certificate projection, and NameCert provenance rows. The certified inverse-limit object is

$$\mathsf{ZetaCertifiedLimit} := \left(\varprojlim_m \mathcal{Z}_m, \mathrm{NameCert}_\infty, \mathrm{LedgerPolicy}_\infty, \mathrm{StabilityCert}_\infty\right).$$

Principle 9303.5 (Zeta is not a seed assertion)**.** *The certified inverse-limit object is a route for admitting completed zeta after the finite layers, transports, continuation rows, explicit formula rows, and limit ledgers have been exposed. It does not assert analytic continuation or RH-shaped statements as naming seeds. Those rows remain theorem targets or operation-certificate targets until the required finite and limiting ledgers are supplied.*

9303.2 Coverage

Definition 9303.6 (Candidate gap and coverage)**.** *At layer m, let GAP_m be the candidate region not yet excluded by the layer's classifiers, probability sieves, fixed-window certificates, and ledgers. Let $\mathbb{P}_m$ be the candidate probability measure visible at that layer. The proof coverage of the layer is*

$$\mathrm{Cov}_m := 1 - \mathbb{P}_m(\mathrm{GAP}_m).$$

Principle 9303.7 (Coverage is not theorem probability)**.** *The number Cov_m is an audited coverage ratio for the candidate space visible at layer m. It is not a probability that* RH *is true. The theorem is a fixed mathematical statement; the layer number measures how much of the candidate gap has been excluded, classified, or forced into a ledger by the proof system.*

Theorem 9303.8 (Coverage is monotone under append-only constraints)**.** *Assume the transition from layer m to layer $m+1$ only adds legal constraints and never revokes a previous ledger exclusion. Then*

$$\mathrm{GAP}_{m+1} \subseteq \mathrm{GAP}_m,$$

and therefore

$$\mathrm{Cov}_{m+1} \geq \mathrm{Cov}_m\,.$$

Proof. Every candidate surviving layer $m+1$ satisfies all constraints from layer m together with the additional constraints of layer $m+1$. Hence it also survives layer m, giving the gap inclusion. Taking candidate measures gives

$$\mathbb{P}_{m+1}(\mathrm{GAP}_{m+1}) \leq \mathbb{P}_m(\mathrm{GAP}_m)$$

after transport along the displayed refinement map. The displayed coverage inequality follows by subtracting from 1. □

Definition 9303.9 (FiniteWitnessCompleteness)**.** *The row*

$$\mathsf{FiniteWitnessCompleteness}$$

asserts that every off-critical zero candidate, if present, generates a finite-layer witness in the certified zeta inverse system. Equivalently, an off-critical zero cannot survive only as a nonlocated point of the limit; it must expose a finite classifier, probability-sieve, fixed-window, or radial-residual row at some layer.

Theorem 9303.10 (Coverage limit excludes certified off-critical witnesses)**.** *Assume*

$$\lim_m \mathrm{Cov}_m = 1$$

and FiniteWitnessCompleteness*. Then no off-critical zero has a legal BEDC witness in* ZetaCertifiedLimit*.*

Proof. An off-critical zero with a legal BEDC witness supplies, by Definition 9303.9, a finite-layer witness in some candidate gap row. Since the coverage tends to 1, the limiting candidate gap has measure zero. A legal BEDC witness cannot enter the object layer merely as an unlocated measure-zero survivor; it must be carried by a finite NameCert or ledger row. The finite witness is therefore eventually excluded or ledgered by the certified inverse system, contradicting legal off-critical survival. □

9303.3 RH Interface

Definition 9303.11 (CertifiedInverseLimitRHRoute)**.** *The row*

$$\mathsf{CertifiedInverseLimitRHRoute}$$

consists of:

$$\mathsf{ZetaCertifiedLimit},\quad \mathsf{FiniteWitnessCompleteness},\quad \mathsf{LayeredProbabilitySieveCompleteness},$$
$$\mathsf{CofinalSieveDissipation},\quad \mathsf{CriticalProbabilityStability},\quad \lim_m \mathrm{Cov}_m = 1.$$

It is a proof-engineering package: each row must be supplied by finite or limiting certificate data before the package can be used.

Theorem 9303.12 (Certified inverse-limit route gives conditional fixed-half readout)**.** *Assume* CertifiedInverseLimitRHRoute *and that every completed nontrivial zero packet admitted by the route is a legal BEDC closure object. Then every admitted nontrivial zero has fixed-half readout.*

Proof. If an admitted nontrivial zero were off the critical line, it would give an off-critical candidate in the certified zeta inverse system. By Definition 9303.10, no such candidate has a legal BEDC witness. Equivalently, if its radial residual is exposed on a probability-filtered curve, then Definition 9302.19 excludes it by cofinal dissipation and critical probability stability. Thus the radial coordinate is zero, and Definition 9266.2 gives fixed-half readout. □

Principle 9303.13 (Certified inverse-limit audit boundary)**.** *The inverse-limit route closes only when every layer is licensed by source, unit, and ledger rows, the coverage numbers are interpreted as gap-coverage measures, and every off-critical candidate is finite-witness visible. Without finite witness completeness, measure-one coverage would be insufficient for a constructive BEDC proof. With finite witness completeness and critical probability stability, a surviving off-critical zero has no legal NameCert route into the completed zeta object.*

Chapter 9304

Probability Sieve Tower Proof Pattern

chapter 9301, chapter 9302, and chapter 9303 provide the finite Hilbert geometry, the probability classifier tower, and the certified limiting object used by the RH route. The present chapter records the proof pattern that binds these pieces together. The pattern is modeled on a tower argument: isolate a fixed local resource, place it in a growing classifier tower, use finite pigeonhole or orthogonal projection to produce many legal reads, and project the resulting tower data into the final geometric object.

9304.1 Local Mechanism

Definition 9304.1 (Unit-phase local mechanism)**.** *The local legal resource for the probability sieve tower is a unit phase mode*

$$u \longmapsto e^{i\gamma u}.$$

A radial packet is a mode of the form

$$u \longmapsto e^{ru}e^{i\gamma u}, \qquad r \neq 0.$$

The local mechanism is legal at a classifier layer only when its observable readout is carried by unit character data, probability amplitude data of norm 1*, and a sieve ledger row. A radial factor is therefore not a unit resource; it is a residual coordinate that must be detected, ledgered, or excluded.*

Principle 9304.2 (Fixed local mechanism)**.** *The probability sieve route does not begin with the global zeta object. It begins with finite layer resources: a finite prime classifier, a probability law on its types, unit character or amplitude readouts, and orthogonal event projections. These are the RH-facing analogues of a fixed local unit resource in a tower construction. Their role is to provide a reusable finite detection mechanism before the limiting object is admitted.*

Definition 9304.3 (Amplifiable classifier mechanism)**.** *An amplifiable classifier mechanism is a finite probability classifier* (G_m, μ_m) *together with a family of legal tests*

$$\mathcal{T}_m \subseteq L_0^2(G_m, \mu_m)$$

such that every nonzero finite residual visible at layer m *has a nonzero pairing with some test in* $\mathcal{T}_m$*. In a refinement tower*

$$(G_0, \mu_0) \leftarrow (G_1, \mu_1) \leftarrow (G_2, \mu_2) \leftarrow \cdots,$$

the mechanism is amplified when the pullback of a detected residual remains visible to later legal tests and the excluded probability mass continues to be recorded by Definition 9302.3.

9304.2 Finite Projection Pigeonhole

Theorem 9304.4 (Finite probability pigeonhole)**.** *Let (G, μ) be a finite probability space with full support and let*

$$R \in L_0^2(G, \mu)$$

be nonzero. Then there is an event $E \subseteq G$ such that

$$\langle R, \mathbf{1}_E - \mu(E) \rangle \neq 0.$$

Equivalently, some legal event projection detects the residual.

Proof. If every centered event indicator were orthogonal to R, then every linear combination of centered event indicators would be orthogonal to R. These centered indicators span $L_0^2(G, \mu)$: for any $a, b \in G$, the difference

$$\mathbf{1}_a - \frac{\mu(a)}{\mu(b)} \mathbf{1}_b$$

is a scalar combination of centered indicators and separates the a-coordinate from the b-coordinate. Hence R would be orthogonal to all of $L_0^2(G, \mu)$, including itself. This gives

$$\|R\|^2 = 0,$$

contradicting $R \neq 0$. □

Theorem 9304.5 (Classifier pigeonhole supplies probability detection)**.** *Every finite prime-power residual detected by Definition 9302.9 is also detected by an event projection in the finite probability space of a prime probability classifier.*

Proof. The finite probability test produced by Definition 9302.9 gives a nonzero element of the mean-zero layer Hilbert space $L_0^2(G_m, \mu_m)$. Applying Definition 9304.4 to that element gives a centered event indicator with nonzero pairing. By Definition 9301.5, the event is an orthogonal probability split, so the detected component and its complement are legal triangle readouts at the layer. □

Principle 9304.6 (Classifier pigeonhole is a ledger operation)**.** *The finite pigeonhole step is not an unrecorded choice of a favorable event. It selects a displayed event in a finite classifier, reads a nonzero projection of the residual, and records the excluded component as probability mass. Thus the pigeonhole argument produces legal BEDC data only together with the sieve ledger of Definition 9302.3.*

9304.3 Tower Amplification

Definition 9304.7 (Tower-amplified detection)**.** *A layered prime probability sieve has tower-amplified detection for a residual R_0 when, after transporting residuals through the refinement maps, there are detected subspaces W_m and constants $\eta_m \in (0, 1]$ with*

$$\|\Pi_m R_{m-1}\|_m^2 \geq \eta_m \|R_{m-1}\|_m^2$$

and

$$\sum_m \eta_m = \infty.$$

Theorem 9304.8 (Tower amplification gives complete residual exhaustion)**.** *If a residual has tower-amplified detection, then its surviving norm in the layered probability sieve tends to zero:*

$$\|R_m\|_m \to 0.$$

Proof. This is exactly the energy descent of Definition 9302.14. The displayed lower bound on the detected component gives

$$\|R_m\|_m^2 \leq \left(\prod_{j=1}^{m}(1-\eta_j)\right)\|R_0\|_0^2.$$

The divergence of $\sum_j \eta_j$ forces the product to vanish. □

Corollary 9304.9 (Amplified detection implies cofinal dissipation)**.** *If every nonzero radial residual of a zeta-facing packet has tower-amplified detection, then*

$$\mathsf{CofinalSieveDissipation}$$

holds for the layered prime probability sieve.

Proof. The definition of tower-amplified detection supplies precisely the detected subspaces and divergent detection strengths required by Definition 9302.15. □

9304.4 Certified Limit Projection

Definition 9304.10 (Layered zeta proof package)**.** *A layered zeta proof package consists of the certified inverse system of Definition 9303.4 together with:*

$$\mathsf{LayeredProbabilitySieveCompleteness},\quad \mathsf{PrimeDistributionCurveCompleteness},$$
$$\mathsf{CofinalSieveDissipation},\quad \mathsf{CriticalProbabilityStability},\quad \mathsf{FiniteWitnessCompleteness}.$$

The package is a proof-engineering object. It does not treat $\zeta(s)$ as a seed primitive; it admits the zeta-facing object only as the certified inverse limit of finite classifier, probability, analytic window, gap, and ledger rows.

Theorem 9304.11 (Layered package projects radial residuals to illegal closure)**.** *Assume a layered zeta proof package and assume every completed nontrivial zero packet admitted by the package is a legal BEDC closure object. If such a zero had*

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}, \qquad \ell \neq 0,$$

then the package would expose a nonzero radial residual that cannot be carried by a closed legal packet.

Proof. By LayeredProbabilitySieveCompleteness, the nonzero radial coordinate is visible on a probability-filtered curve. By Definition 9302.11, this supplies the distribution-curve visibility demanded by PrimeDistributionCurveCompleteness. By Definition 9316.12, the radial coordinate becomes a finite or profinite classifier-visible residual. By CofinalSieveDissipation and Definition 9302.17, that residual cannot be hidden in a legal closed packet: if the exhausted mass is recorded, the packet has an append ledger; if it is not recorded, the sieve ledger row is violated. The assumed closed legal zero packet is therefore incompatible with $\ell \neq 0$. □

Theorem 9304.12 (Probability sieve tower route gives conditional fixed-half readout)**.** *Assume a layered zeta proof package and that every completed nontrivial zero packet admitted by the package is a legal BEDC closure object. Then every admitted nontrivial zero has fixed-half readout.*

Proof. Let ρ be an admitted nontrivial zero. If

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}$$

had $\ell \neq 0$, then Definition 9304.11 would contradict legality of the closed packet. Hence $\ell = 0$, and Definition 9266.2 gives fixed-half readout. □

Principle 9304.13 (Proof pattern audit boundary)**.** *The tower proof pattern concentrates the RH-facing burden in two certificate rows. The finite visibility row says that an off-critical zero cannot remain purely global; it must create a finite classifier, probability, fixed-window, or distribution-curve witness. The cofinal amplification row says that every such witness is repeatedly detected with divergent total energy. Once these rows are supplied, the certified inverse-limit projection turns radial scale into an illegal closure coordinate rather than a legal zeta zero.*

Chapter 9305

Prime-Generated Recursive Sieve

chapter 9314 treats primes as both objects of classification and generators of local sieve surfaces. The present chapter isolates the more basic recursive package. It starts from the local residue classifier attached to a prime, iterates such classifiers as a sieve tower, and reads the right-half-plane Euler product as the phase expectation of the resulting prime-generated probability system.

9305.1 Recursive Local Sieve

Definition 9305.1 (Prime-generated recursive sieve step)**.** *Let X_m be a displayed source set, let p_m be a prime, and let*

$$\pi_{p_m} : X_m \to G_{p_m}$$

be a finite classifier generated by p_m, such as a residue, unit-residue, valuation, order, or character readout. For an event $A_m \subseteq G_{p_m}$, define

$$C_m := \{x \in X_m : \pi_{p_m}(x) \in A_m\},$$

$$X_{m+1} := X_m \setminus C_m,$$

and append the ledger row

$$\Lambda_{m+1} := \Lambda_m + (p_m, A_m, \mu_{p_m}(A_m), C_m, \mathrm{reason}_m).$$

Definition 9305.2 (PrimeGeneratedRecursiveSieve)**.** *The row*

$$\mathsf{PrimeGeneratedRecursiveSieve}$$

is an iterated system of Definition 9305.1 whose source sets may be integers, primes, prime powers, finite residual supports, or sieve-method parameter spaces. Each step is generated by a prime and must expose its classifier, event, probability mass, and ledger append row.

Theorem 9305.3 (Recursive prime-generated sieve preserves BEDC rows)**.** *Every finite stage of* PrimeGeneratedRecursiveSieve *preserves the BEDC time, unit, and ledger rows.*

Proof. The time row is the nested chain

$$X_0 \supseteq X_1 \supseteq X_2 \supseteq \cdots$$

together with the append-only ledger equations in Definition 9305.1. The unit row is the probability-amplitude split on each finite classifier:

$$\psi_m = P_{A_m}\psi_m + P_{A_m^c}\psi_m, \qquad P_{A_m}\psi_m \perp P_{A_m^c}\psi_m,$$

so

$$\|\psi_m\|^2 = \|P_{A_m}\psi_m\|^2 + \|P_{A_m^c}\psi_m\|^2.$$

This is the event projection law of Definition 9301.5. The ledger row is the displayed Λ_{m+1} update, which records the selected prime, event, probability mass, excluded source, and reason. □

9305.2 Finite Residual Detection

Theorem 9305.4 (Prime-generated sieve detects finite residuals)**.** *Let*

$$R = \sum_{j=1}^{N} c_j[q_j] \neq 0$$

be a finite residual with distinct positive integers q_j. Then there are a prime p and a character

$$\chi : \mathbb{F}_p^\times \to S^1$$

such that

$$\sum_{j=1}^{N} c_j \chi(q_j \bmod p) \neq 0.$$

Proof. Choose a prime p dividing none of the q_j and none of the nonzero differences $q_i - q_j$. Then the residues $q_j \bmod p$ are pairwise distinct elements of $\mathbb{F}_p^\times$. If all characters of $\mathbb{F}_p^\times$ had zero readout on R, character orthogonality on the finite abelian group $\mathbb{F}_p^\times$ would force the coefficient sum on each residue class to vanish. Since the displayed residues are pairwise distinct, every c_j would be zero, contradicting $R \neq 0$. □

Corollary 9305.5 (Prime-generated recursive completeness)**.** *Every nonzero finite residual is detected by a local sieve generated by some prime. Equivalently,*

$$R \neq 0 \quad \Rightarrow \quad \exists p, \exists \chi \in \widehat{\mathbb{F}_p^\times}, \quad \langle R, \chi \circ \pi_p \rangle \neq 0.$$

Proof. Use Definition 9305.4 with $\pi_p(q) = q \bmod p$. □

9305.3 Euler Probability Readout

Definition 9305.6 (Right-half-plane Euler probability law)**.** *For $\sigma > 1$, define a probability law on positive integers by*

$$\mathbb{P}_\sigma(n) := \frac{n^{-\sigma}}{\zeta(\sigma)}.$$

Under the prime factorization

$$n = \prod_p p^{v_p(n)},$$

the local valuation row at prime p has distribution

$$\mathbb{P}_\sigma(v_p = k) = (1 - p^{-\sigma})p^{-k\sigma}.$$

Theorem 9305.7 (Euler probability factors by prime valuations)**.** *For $\sigma > 1$, the valuation rows v_p are independent under $\mathbb{P}_\sigma$, with the local distribution of Definition 9305.6.*

Proof. For a finite prime window S and exponents $k_p \geq 0$,

$$\mathbb{P}_\sigma(v_p = k_p \text{ for } p \in S) = \frac{1}{\zeta(\sigma)} \left(\prod_{p \in S} p^{-k_p \sigma} \right) \prod_{p \notin S} \left(\sum_{k \geq 0} p^{-k\sigma} \right).$$

Using the Euler product on $\sigma > 1$, the terms outside S cancel the corresponding factors of $\zeta(\sigma)$, leaving

$$\prod_{p \in S} (1 - p^{-\sigma}) p^{-k_p \sigma}.$$

This is the product of the displayed local laws, hence the valuation rows are independent. □

Theorem 9305.8 (Zeta ratio is the phase expectation)**.** *For $\sigma > 1$ and real t,*

$$\mathbb{E}_{\mathbb{P}_\sigma}[n^{-it}] = \frac{\zeta(\sigma + it)}{\zeta(\sigma)}.$$

Proof. By definition of $\mathbb{P}_\sigma$,

$$\mathbb{E}_{\mathbb{P}_\sigma}[n^{-it}] = \sum_{n \geq 1} \frac{n^{-\sigma}}{\zeta(\sigma)} n^{-it} = \frac{1}{\zeta(\sigma)} \sum_{n \geq 1} n^{-(\sigma + it)} = \frac{\zeta(\sigma + it)}{\zeta(\sigma)}.$$

Absolute convergence is available because $\sigma > 1$. □

Definition 9305.9 (Finite Euler probability phase packet)**.** *For a finite prime window $S = \{p_1, \ldots, p_m\}$, define*

$$\mathcal{Z}_S(\sigma, t) := \prod_{p \in S} \frac{1 - p^{-\sigma}}{1 - p^{-\sigma - it}}.$$

This is the phase expectation of the finite product of local geometric valuation laws generated by primes in S.

Theorem 9305.10 (Finite Euler packets converge to the zeta phase readout)**.** *For $\sigma > 1$ and cofinal finite prime windows S,*

$$\mathcal{Z}_S(\sigma, t) \to \frac{\zeta(\sigma + it)}{\zeta(\sigma)}.$$

Proof. The finite product in Definition 9305.9 is the quotient of the finite Euler products for $\zeta(\sigma + it)$ and $\zeta(\sigma)$ over S. Both Euler products converge absolutely on $\sigma > 1$, so their quotient converges to the displayed zeta ratio. □

Principle 9305.11 (Right-half-plane zeta is a prime-generated phase limit)**.** *On $\Re(s) > 1$, the normalized zeta phase $\zeta(\sigma + it)/\zeta(\sigma)$ is already a legal inverse limit of prime-generated local probability sieves. This statement uses only unique factorization, geometric valuation laws, and absolute convergence of the Euler product.*

9305.4 Continuation Interface

Definition 9305.12 (AnalyticContinuationPreservesPrimeSieveLedger)**.** *The row*

AnalyticContinuationPreservesPrimeSieveLedger

asserts that the analytic continuation and explicit-formula transport from $\Re(s) > 1$ *to the critical strip preserves the prime-generated sieve ledger: finite prime windows, local valuation rows, unit phase readouts, residual projections, and excluded-mass ledger entries remain visible as certified rows of the continued zeta object.*

Theorem 9305.13 (Prime-generated recursive sieve gives conditional fixed-half route)**.** *Assume* AnalyticContinuationPreservesPrimeSieveLedger*, every off-critical zero supplies a nonzero prime-side residual visible at a finite layer, and detected residuals cannot be hidden in legal closed BEDC/PFZ zero packets. Then every admitted nontrivial zero has fixed-half readout.*

Proof. An off-critical admitted zero supplies, by hypothesis, a nonzero finite-layer prime-side residual. By Definition 9305.5, some prime-generated local sieve detects it. The continuation-preservation row keeps this detection inside the certified zeta ledger rather than outside the continued object. If the detected residual is recorded, the zero packet carries an append ledger and is not closed. If it is not recorded, the ledger policy is violated. Thus a legal closed admitted zero cannot be off-critical, and Definition 9266.2 gives fixed-half readout. □

Principle 9305.14 (Recursive sieve audit boundary)**.** *The recursive prime-generated sieve packet proves the finite arithmetic side: local prime sieves are BEDC-legal, detect finite residuals, and construct the normalized Euler phase on* $\Re(s) > 1$*. The remaining analytic burden is the preservation row* AnalyticContinuationPreservesPrimeSieveLedger*, together with the explicit-formula row that turns an off-critical zero into a prime-side residual.*

Chapter 9306

All-Prime Object

chapter 9305 treats each prime as a generator of local sieve rows. The RH route also requires the source containing all primes to be a BEDC object rather than an unstructured ambient set. The present chapter packages the all-prime source as

$$\mathcal{P}^{\uparrow}$$

with a displayed time row, unit row, and ledger row. This prevents a radial residual from being hidden in an unaccounted prime tail.

9306.1 Prime Source Packet

Definition 9306.1 (Finite prime source layer)**.** *Let $p_1, p_2, p_3, \ldots$ be the increasing prime sequence and define*

$$\mathcal{P}_N := \{p_1, \ldots, p_N\}.$$

The transition

$$\iota_{N,N+1} : \mathcal{P}_N \hookrightarrow \mathcal{P}_{N+1}$$

is inclusion.

Definition 9306.2 (AllPrimeObject)**.** *The BEDC all-prime object is the packet*

$$\mathcal{P}^{\uparrow} := ((\mathcal{P}_N)_{N\geq 1}, (\iota_{N,N+1})_{N\geq 1}, \Lambda_{\mathcal{P}}, \mathrm{NameCert}_{\mathcal{P}}, \mathrm{TailPolicy}_{\mathcal{P}})\,.$$

It is a certified colimit-style source: finite prime layers supply the visible stages, $\Lambda_{\mathcal{P}}$ records generation and sieve history, $\mathrm{NameCert}_{\mathcal{P}}$ *licenses the source as a BEDC object, and* $\mathrm{TailPolicy}_{\mathcal{P}}$ *records data not visible at a chosen finite layer.*

Definition 9306.3 (Prime object time row)**.** *The time row*

$$\mathsf{T}_{\mathcal{P}}$$

asserts that $\mathcal{P}^{\uparrow}$ is generated by the Euclidean survivor tower of Definition 9314.1. With

$$P_N := \prod_{j=1}^{N} p_j, \qquad \mathcal{N}_N := \{n \geq 2 : \gcd(n, P_N) = 1\},$$

the appended source row is

$$p_{N+1} := \min \mathcal{N}_N, \qquad \mathcal{P}_{N+1} = \mathcal{P}_N \cup \{p_{N+1}\},$$

and the ledger update is

$$\Lambda_{\mathcal{P},N+1} = \Lambda_{\mathcal{P},N} + (p_{N+1}, \mathrm{minimalSurvivor}(P_N)).$$

Theorem 9306.4 (Prime object time row is append-only)**.** *The row $\mathsf{T}_{\mathcal{P}}$ is an append-only generation row.*

Proof. By Definition 9314.2, the minimal survivor of $\mathcal{N}_N$ is a prime distinct from $p_1, \ldots, p_N$. Hence adjoining it produces precisely the next finite prime source layer. The displayed update records that adjoining event without deleting any prior prime row, so the generation history is append-only. □

9306.2 Unit Rows

Definition 9306.5 (Prime circle layer)**.** *For $N \geq 1$, set $P = p_N$. The Nth prime circle layer is the embedded finite set*

$$\mathbb{P}_N^\circ := \{p_1, \ldots, p_{N-1}, 0\} \subseteq \mathbb{F}_P,$$

obtained from

$$\pi_P : \mathcal{P}_N \to \mathbb{F}_P, \qquad \pi_P(p_j) = p_j \bmod P.$$

The element 0 represents the terminal point $p_N = P$ of the finite prime source layer.

Lemma 9306.6 (Prime circle embedding)**.** *The map $\pi_P : \mathcal{P}_N \to \mathbb{F}_P$ is injective.*

Proof. For $j < N$, the integer p_j satisfies $0 < p_j < P$, while $p_N = P$ maps to 0. The residues $p_1, \ldots, p_{N-1}, 0$ are therefore pairwise distinct in $\mathbb{F}_P$. □

Definition 9306.7 (Prime object unit row)**.** *The unit row*

$$\mathsf{U}_{\mathcal{P}}$$

contains two compatible unit readouts:

$$\psi_t(x) := \exp(2\pi i t x/P), \qquad t, x \in \mathbb{F}_P,$$
$$\mathbb{P}_\sigma(v_p = k) = (1 - p^{-\sigma})p^{-k\sigma}, \qquad \sigma > 1.$$

The first is the finite additive character system on the prime circle layer. The second is the Euler local probability row of Definition 9305.6.

Theorem 9306.8 (Prime object unit rows are normalized)**.** *The finite character and Euler probability rows of $\mathsf{U}_{\mathcal{P}}$ are unit rows.*

Proof. For the finite character row,

$$|\psi_t(x)| = 1$$

for all $t, x \in \mathbb{F}_P$. For the Euler row,

$$\sum_{k \geq 0} (1 - p^{-\sigma})p^{-k\sigma} = 1$$

at each prime p when $\sigma > 1$. By Definition 9305.7, these local rows factor the probability law by prime valuations. By Definition 9305.8, the normalized right-half-plane zeta ratio is the unit phase expectation

$$\mathbb{E}_{\mathbb{P}_\sigma}[n^{-it}] = \frac{\zeta(\sigma + it)}{\zeta(\sigma)}, \qquad |n^{-it}| = 1.$$

Thus both finite and Euler layers expose unit readouts. □

9306.3 Ledger Rows

Definition 9306.9 (Prime object ledger row)**.** *The ledger row*

$$\mathsf{L}_{\mathcal{P}}$$

records unique factorization and finite-layer tails. For $n \geq 1$,

$$\mathrm{PrimeLedger}(n) := (v_p(n))_{p \in \mathcal{P}}.$$

At the finite layer $\mathcal{P}_N$ the ledger is

$$\mathrm{PrimeLedger}_N(n) := \Big((v_{p_j}(n))_{j \leq N}, n_{>N} \Big),$$

where

$$n_{>N} := \prod_{j>N} p_j^{v_{p_j}(n)}.$$

For the Euler product the corresponding finite-layer ledger is

$$\mathrm{EulerLedger}_N(s) := \left(\prod_{j \leq N} (1 - p_j^{-s})^{-1}, \prod_{j>N} (1 - p_j^{-s})^{-1} \right)$$

on the half-plane where the Euler product is licensed.

Theorem 9306.10 (Prime object ledger records finite data and tail data)**.** *The row $\mathsf{L}_{\mathcal{P}}$ prevents a finite-layer readout from discarding information outside $\mathcal{P}_N$.*

Proof. Unique factorization gives

$$n = \left(\prod_{j \leq N} p_j^{v_{p_j}(n)} \right) n_{>N}.$$

Thus the finite valuation row alone is not the full ledger unless the tail $n_{>N}$ is also recorded. The Euler product has the same structure: the finite product over $j \leq N$ is accompanied by the displayed Euler tail over $j > N$. Therefore every finite readout of the all-prime object carries an explicit tail coordinate. □

Theorem 9306.11 (AllPrimeObject has an ontological triple)**.** *The BEDC all-prime object has*

$$\mathrm{Ont}(\mathcal{P}^{\uparrow}) = (\mathsf{T}_{\mathcal{P}}, \mathsf{U}_{\mathcal{P}}, \mathsf{L}_{\mathcal{P}})$$

in the sense of Definition 9303.1.

Proof. The time row is Definition 9306.3, and it is append-only by Definition 9306.4. The unit row is Definition 9306.7, normalized by Definition 9306.8. The ledger row is Definition 9306.9, with tail preservation given by Definition 9306.10. These are exactly the three coordinates required by Definition 9303.1. □

9306.4 Finite Detection

Theorem 9306.12 (Prime circle layer detects finite prime residuals)**.** *Let*

$$R_N = \sum_{j=1}^{N} c_j[p_j] \neq 0$$

be a residual supported on $\mathcal{P}_N$, and set $P = p_N$. Then there exists $t \in \mathbb{F}_P$ such that

$$\sum_{j=1}^{N} c_j \psi_t(p_j \bmod P) \neq 0.$$

Proof. Push R_N forward along the embedding of Definition 9306.6:

$$\widetilde{R}_N := \sum_{j=1}^{N} c_j[\pi_P(p_j)] \in \mathbb{C}[\mathbb{F}_P].$$

Since the points $\pi_P(p_j)$ are pairwise distinct, $R_N \neq 0$ implies $\widetilde{R}_N \neq 0$. The additive characters

$$\psi_t(x) = \exp(2\pi i t x/P)$$

form a Fourier basis for functions on the finite additive group $\mathbb{F}_P$. If every displayed character readout vanished on $\widetilde{R}_N$, all Fourier coefficients of $\widetilde{R}_N$ would vanish, forcing $\widetilde{R}_N = 0$, a contradiction. □

Definition 9306.13 (Finite-or-tail prime residual)**.** *A residual over $\mathcal{P}^{\uparrow}$ is finite-or-tail visible when, for every finite layer $\mathcal{P}_N$, it satisfies one of the following ledger alternatives:*

a nonzero component is visible on some finite prime layer,

the invisible component is recorded in $\mathrm{TailPolicy}_{\mathcal{P}}$ *.*

Theorem 9306.14 (AllPrimeObject has no unledgered residual escape)**.** *Assume a residual over $\mathcal{P}^{\uparrow}$ is finite-or-tail visible. Then it cannot be hidden inside the all-prime source as an unledgered ambient coordinate.*

Proof. If the residual has a nonzero finite component on $\mathcal{P}_N$, then Definition 9306.12 detects it by a unit character row on the prime circle layer. If no finite component is visible at the chosen layer, the definition of finite-or-tail visibility places the component in $\mathrm{TailPolicy}_{\mathcal{P}}$. Thus every residual is either detected at a finite layer or recorded as tail data. Neither case is an unledgered source coordinate. □

9306.5 Zeta Source Interface

Definition 9306.15 (Certified Euler prime limit)**.** *For $\Re(s) > 1$, the certified Euler prime limit attached to $\mathcal{P}^{\uparrow}$ is the packet*

$$\zeta^{\uparrow}_{\Re>1} := ((Z_N(s))_{N\geq 1}, (\mathrm{EulerTail}_N(s))_{N\geq 1}, \Lambda_{\mathcal{P}}, \mathrm{NameCert}_{\mathcal{P}}),$$

where

$$Z_N(s) := \prod_{j\leq N} (1 - p_j^{-s})^{-1}$$

and

$$\mathrm{EulerTail}_N(s) := \prod_{j>N} (1 - p_j^{-s})^{-1}.$$

Theorem 9306.16 (AllPrimeObject supplies the right-half-plane zeta source)**.** *On $\Re(s) > 1$, the certified Euler prime limit is a BEDC-licensed source for the normalized zeta phase readout.*

Proof. The finite products $Z_N(s)$ are generated by appending Euler local factors along the time row of $\mathcal{P}^{\uparrow}$. The tails $\mathrm{EulerTail}_N(s)$ are the Euler ledger tails of Definition 9306.9. Absolute convergence on $\Re(s) > 1$ licenses the Euler product, and Definition 9305.11 identifies the normalized zeta phase as the limit of prime-generated local probability sieves. The source is therefore certified by finite products, tail rows, and the all-prime NameCert rather than by a bare infinite set of primes. □

Principle 9306.17 (All-prime source audit boundary)**.** *The all-prime object closes the source-side gap for the prime sieve route. It does not prove that an off-critical zero produces a finite-or-tail prime residual. It asserts that, once such a residual is present over the prime source, the residual must be detected by a finite prime circle layer or recorded in the all-prime tail ledger.*

Chapter 9307

Finite Prime Circle Universe

chapter 9306 packages the all-prime source as a generated BEDC object with finite layers and tail ledger rows. The present chapter records the complementary fixed finite model: assume a finite prime universe with largest prime P, close that universe inside $\mathbb{F}_P$, and show that a nonzero finite residual cannot be missed by all P-generated Fourier or quadratic norm sieves. This is a local certificate; it is not a substitute for the all-prime tail policy.

9307.1 Index–Prime Certificate

Definition 9307.1 (IndexPrimePair)**.** *An index–prime pair certificate is a packet*

$$\mathsf{IndexPrimePair}^{\uparrow}(N, P)$$

asserting that

$$P = p_N, \qquad N = \pi(P),$$

where p_N is the Nth prime and

$$\pi(P) := \#\{p : p \le P,\ p \text{ prime}\}.$$

Thus N is the time coordinate of the finite prime source layer and P is the maximal prime that supplies the finite circle modulus.

Definition 9307.2 (IndexPrimeLedger)**.** *The ledger attached to $\mathsf{IndexPrimePair}^{\uparrow}(N, P)$ is*

$$\mathrm{IndexPrimeLedger}(N, P) := (P = p_N,\ N = \pi(P),\ \pi_P(\mathcal{P}_N),\ \mathrm{GapHoles}(N, P),\ \alpha_N)\,,$$

where

$$\mathrm{GapHoles}(N, P) := \mathbb{F}_P \setminus \pi_P(\mathcal{P}_N)$$

and

$$\alpha_N := \frac{N}{P}.$$

The row records the occupied prime points, the empty finite-circle points, and the occupancy ratio of the finite layer.

Theorem 9307.3 (IndexPrimePair is the legal finite-circle modulus)**.** *If $\mathsf{IndexPrimePair}^{\uparrow}(N, P)$ holds, then P is the unique modulus used by the finite prime circle at layer N.*

Proof. The certificate supplies $P = p_N$, so every earlier source prime p_j with $j < N$ satisfies $p_j < P$, while p_N maps to 0 modulo P. The same certificate supplies $N = \pi(P)$, so the displayed finite source contains exactly the primes up to P. A smaller modulus would not be the maximal prime boundary of this layer; a larger prime modulus would add a prime beyond the certified source; a larger composite modulus would not provide the field $\mathbb{F}_P$ used by the circle and Gauss rows. Thus the finite-circle modulus is not a free parameter. □

Theorem 9307.4 (IndexPrimePair has BEDC rows)**.** *The certificate* $\mathsf{IndexPrimePair}^{\uparrow}(N, P)$ *carries time, unit, and ledger rows.*

Proof. The time row is the index N in the append-only prime source tower. The unit row is the finite circle $\mathbb{F}_P$ supplied by the maximal prime $P = p_N$, together with the unit characters

$$\psi_t(x) = \exp(2\pi i t x/P).$$

The ledger row is Definition 9307.2: it records $P = p_N$, $N = \pi(P)$, the occupied points, the gap holes, and the occupancy ratio. These are the three BEDC coordinates required for using (N, P) as a finite-layer proof parameter. □

9307.2 Finite Prime Circle

Definition 9307.5 (Finite prime universe)**.** *A finite prime universe is a displayed finite source*

$$\mathcal{P}_N = \{p_1, \ldots, p_N\}, \qquad P := p_N,$$

equipped with the certificate

$$\mathsf{IndexPrimePair}^{\uparrow}(N, P).$$

It is read under the model assumption that these are exactly the primes visible to the finite source up to the certified boundary P. Its maximal-prime classifier is

$$\pi_P : \mathcal{P}_N \to \mathbb{F}_P, \qquad \pi_P(p_j) = p_j \bmod P.$$

Definition 9307.6 (Finite prime circle universe)**.** *The finite prime circle universe attached to Definition 9307.5 is*

$$\mathcal{P}_N^{\circ} := \{p_1, \ldots, p_{N-1}, 0\} \subseteq \mathbb{F}_P.$$

The point 0 *is the image of the maximal prime P.*

Lemma 9307.7 (Maximal-prime classifier is injective)**.** *The map $\pi_P : \mathcal{P}_N \to \mathbb{F}_P$ is injective.*

Proof. For $j < N$, one has $0 < p_j < P$, so $\pi_P(p_j) = p_j$ as a nonzero element of $\mathbb{F}_P$. The maximal prime satisfies $\pi_P(P) = 0$. Hence the images are the pairwise distinct points

$$p_1, \ldots, p_{N-1}, 0.$$

□

9307.3 Fourier Completeness

Definition 9307.8 (Prime-circle residual)**.** *A prime-circle residual is a finite row*

$$R_N = \sum_{j=1}^{N} c_j[p_j]$$

on $\mathcal{P}_N$. Its pushforward to $\mathbb{F}_P$ is

$$\widetilde{R}_N := \sum_{j=1}^{N} c_j[\pi_P(p_j)].$$

Definition 9307.9 (Maximal-prime Fourier sieve)**.** *For $t \in \mathbb{F}_P$, the maximal-prime Fourier sieve is the additive character*

$$\psi_t(x) := \exp(2\pi i t x/P), \qquad x \in \mathbb{F}_P.$$

Its readout on a prime-circle residual is

$$\widehat{R}_N(t) := \sum_{j=1}^{N} c_j \psi_t(p_j \bmod P).$$

Theorem 9307.10 (Finite prime circle Fourier completeness)**.** *If $R_N \neq 0$, then there exists $t \in \mathbb{F}_P$ such that*

$$\widehat{R}_N(t) \neq 0.$$

Proof. By Definition 9307.7, the pushforward $\widetilde{R}_N$ is nonzero whenever R_N is nonzero. The additive characters of the finite additive group $\mathbb{F}_P$ form an orthogonal basis of $L^2(\mathbb{F}_P)$. If every readout $\widehat{R}_N(t)$ vanished, all Fourier coefficients of $\widetilde{R}_N$ would vanish, forcing $\widetilde{R}_N = 0$. This contradicts $R_N \neq 0$. □

9307.4 Quadratic Norm Sieve

Definition 9307.11 (Quadratic norm sieve over the finite prime circle)**.** *Let*

$$Q_P : \mathbb{F}_P^2 \to \mathbb{F}_P, \qquad Q_P(a,b) = a^2 + b^2.$$

For a function $R : \mathbb{F}_P \to \mathbb{C}$ and $t \in \mathbb{F}_P$, define the quadratic norm sieve readout

$$\mathcal{Q}_t(R) := \sum_{q \in \mathbb{F}_P} R(q) \sum_{a,b \in \mathbb{F}_P} \exp(2\pi i t(q - a^2 - b^2)/P).$$

Definition 9307.12 (Quadratic Gauss row)**.** *For $t \in \mathbb{F}_P$, define*

$$G_P(t) := \sum_{a \in \mathbb{F}_P} \exp(2\pi i t a^2/P).$$

The Gauss nondegeneracy row asserts that

$$G_P(t) \neq 0.$$

For $t = 0$ this is the identity $G_P(0) = P$; for $t \neq 0$ it is the standard nondegeneracy of the quadratic Gauss sum.

Theorem 9307.13 (Quadratic norm sieve is Fourier sieve times Gauss rows)**.** *For every $t \in \mathbb{F}_P$ and every $R : \mathbb{F}_P \to \mathbb{C}$,*

$$\mathcal{Q}_t(R) = G_P(-t)^2 \sum_{q \in \mathbb{F}_P} R(q) \exp(2\pi i t q/P).$$

Proof. Expand the definition of $\mathcal{Q}_t(R)$ and separate the variables:

$$\begin{aligned}\mathcal{Q}_t(R) &= \sum_q R(q) \exp(2\pi i t q/P) \left(\sum_a \exp(-2\pi i t a^2/P)\right) \left(\sum_b \exp(-2\pi i t b^2/P)\right) \\ &= G_P(-t)^2 \sum_q R(q) \exp(2\pi i t q/P).\end{aligned}$$

□

Corollary 9307.14 (Quadratic norm sieves detect Fourier-visible residuals)**.** *Assume the Gauss nondegeneracy row of Definition 9307.12. If*

$$\sum_{q \in \mathbb{F}_P} R(q) \exp(2\pi i t q/P) \neq 0$$

for some $t \in \mathbb{F}_P$, then

$$\mathcal{Q}_t(R) \neq 0.$$

Proof. By Definition 9307.13, the quadratic readout is the displayed Fourier readout multiplied by $G_P(-t)^2$. The Gauss nondegeneracy row makes that multiplier nonzero. □

9307.5 Leakage Contradiction

Definition 9307.15 (Prime circle leakage assumption)**.** *The prime circle leakage assumption for a residual R_N says that all maximal-prime Fourier sieve readouts vanish:*

$$\widehat{R}_N(t) = 0 \qquad (t \in \mathbb{F}_P),$$

and, when the quadratic norm sieve is available, all quadratic norm readouts vanish:

$$\mathcal{Q}_t(\widetilde{R}_N) = 0 \qquad (t \in \mathbb{F}_P).$$

Theorem 9307.16 (Finite prime universe leakage contradiction)**.** *In a finite prime universe, no nonzero prime-circle residual satisfies the prime circle leakage assumption.*

Proof. If $R_N \neq 0$, then Definition 9307.10 gives $t \in \mathbb{F}_P$ with $\widehat{R}_N(t) \neq 0$. This contradicts the first row of Definition 9307.15. Therefore any residual satisfying the leakage assumption must be zero. □

Corollary 9307.17 (Quadratic leakage contradiction)**.** *Assume the Gauss nondegeneracy row and suppose a nonzero residual has a nonzero Fourier readout at some $t \in \mathbb{F}_P$. Then it cannot be missed by all quadratic norm sieves.*

Proof. Apply Definition 9307.14 to the nonzero Fourier readout. The corresponding quadratic norm sieve readout is nonzero. □

9307.6 Admissible Addresses

Definition 9307.18 (Finite prime address policy)**.** *In the finite prime universe of Definition 9307.5, an admissible residual address is either one of the displayed points*

$$h_j := p_j \bmod P \in \mathbb{F}_P \qquad (1 \le j \le N)$$

or a ledger row saying that the residual has no address in the finite universe. An expression involving a prime $p' > P$ is not an admissible prime address inside the finite model.

Theorem 9307.19 (Existing prime addresses are event-detected)**.** *Let*

$$R_N = \sum_{j=1}^{N} c_j[p_j]$$

be a prime-circle residual. If $c_j \neq 0$ for some j, then the event

$$E_j := \{p_j \bmod P\} \subseteq \mathbb{F}_P$$

detects R_N.

Proof. By Definition 9307.7, the point $p_j \bmod P$ occurs in the pushforward of R_N only with coefficient c_j. Pairing with the indicator of E_j therefore gives exactly c_j, which is nonzero. □

Theorem 9307.20 (No external prime address in the finite model)**.** *Under the finite prime universe assumption, a residual cannot use a prime $p' > P$ as a legal finite prime-circle address.*

Proof. The finite prime universe states that the displayed source of prime addresses is precisely

$$\mathcal{P}_N = \{p_1, \ldots, p_N = P\}.$$

Thus a prime $p' > P$ is not a point of the source and cannot be used as a finite sieve address. If such a row is asserted as a prime, the finite universe assumption fails. If it is asserted merely as data not belonging to $\mathcal{P}_N$, it must be recorded by the ledger as an outside-source residual rather than used as a detector in the finite model. □

Theorem 9307.21 (Address trichotomy for finite residuals)**.** *Let a residual in the finite prime universe satisfy the BEDC ledger requirement that every residual coordinate has a source address or an explicit ledger row. Then exactly the following alternatives are available:*

it has a nonzero displayed address h_j and is event-detected,

it requests an external prime address and violates the finite model,

it has no displayed address and must be recorded as a ledger row.

Consequently a nonzero finite residual cannot be both ledger-legal and invisible to all displayed prime-circle events.

Proof. If some displayed coefficient c_j is nonzero, then Definition 9307.19 gives an event detector. If the residual is assigned to a prime $p' > P$, then Definition 9307.20 says that this is not a legal finite-model address. The only remaining possibility is that the residual has no displayed address in $\mathcal{P}_N^\circ$; by the stated ledger requirement, it must then be recorded as an explicit ledger row. Thus a nonzero residual cannot remain both legal and undetected in the finite prime circle universe. □

Definition 9307.22 (Finite prime circle ledger)**.** *The finite prime circle ledger records every nonzero readout as*

$$\mathrm{PrimeCircleLedger}_{P,t} := (P, t, R_N, \widehat{R}_N(t))$$

for Fourier readouts, and as

$$\mathrm{QuadraticSieveLedger}_{P,t} := (P, Q_P, t, \mathcal{Q}_t(\widetilde{R}_N))$$

for quadratic norm readouts.

Theorem 9307.23 (Finite prime circle universe preserves BEDC rows)**.** *The finite prime circle universe preserves time, unit, and ledger rows.*

Proof. The time row is the displayed finite source $\mathcal{P}_N$ with maximal boundary P and classifier π_P. The unit row consists of the additive characters ψ_t, each of modulus 1, together with the quadratic Gauss rows after their normalization by $P^{1/2}$ when the quadratic sieve is used. The ledger row is Definition 9307.22: every nonzero Fourier or quadratic readout is recorded with the modulus, sieve parameter, source residual, and detected value. □

9307.7 RH Route Interface

Definition 9307.24 (FiniteLayerResidual)**.** *The row*

$$\mathsf{FiniteLayerResidual}(\rho, N)$$

asserts that an off-critical zero packet ρ supplies a nonzero prime-circle residual R_N at the finite prime source layer $\mathcal{P}_N$.

Theorem 9307.25 (Finite layer residual cannot be sieve-invisible)**.** *Assume* $\mathsf{FiniteLayerResidual}(\rho, N)$ *and the finite prime circle universe for $\mathcal{P}_N$. Then the residual supplied by ρ is detected by a maximal-prime Fourier sieve. If the Gauss nondegeneracy row is supplied, it is also detected by a quadratic norm sieve.*

Proof. The finite-layer residual row supplies $R_N \neq 0$. Apply Definition 9307.10 to obtain a maximal-prime Fourier detector. The quadratic detector follows from Definition 9307.14. □

Principle 9307.26 (Finite prime circle audit boundary)**.** *The finite prime circle universe closes the finite combinatorial part of the route. Once an off-critical zero has produced a nonzero finite-layer residual, the maximal-prime circle cannot miss it. The remaining analytic burden is the production of*

$$\mathsf{FiniteLayerResidual}(\rho, N)$$

from the off-critical zero packet.

Chapter 9308

Off-Line Zero Finite-or-Tail Address

chapter 9306 supplies the all-prime source with finite layers and tail policy, while chapter 9307 supplies the certified finite circle attached to an index–prime pair (N, P). This chapter records the address step needed by the off-line zero route: once an off-line zero packet carries an ontological triple and a nonzero radial residual, that residual cannot remain addressless. At a certified finite prime circle it is either assigned to a displayed finite prime point, or it is assigned to the prime tail ledger beyond P. No other legal source status is available.

9308.1 Address Rows

Definition 9308.1 (Off-line residual address packet)**.** *Let*

$$\rho = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

be an off-line zero packet, and let

$$R_\rho \neq 0$$

be its radial residual row. An off-line residual address packet at the finite prime circle (N, P) consists of the data

$$\begin{gathered}\mathrm{Ont}(\rho) = (\mathsf{T}_\rho, \mathsf{U}_\rho, \mathsf{L}_\rho),\\ \mathsf{IndexPrimePair}^{\uparrow}(N, P),\\ \mathcal{P}_N = \{p_1, \ldots, p_N = P\},\\ \mathrm{TailPolicy}_{\mathcal{P}}\,.\end{gathered}$$

The packet is legal only when the ledger row L_ρ assigns each nonzero residual coordinate to a displayed source address or to an explicit tail row.

Definition 9308.2 (Finite and tail addresses)**.** *For an off-line residual address packet, a finite address is a representation*

$$R_N = \sum_{j=1}^{N} c_j[p_j] \neq 0$$

of the radial residual at the displayed source layer $\mathcal{P}_N$, with finite-circle pushforward

$$\widetilde{R}_N := \sum_{j=1}^{N} c_j[\pi_P(p_j)] \in \mathbb{C}[\mathbb{F}_P].$$

A tail address is a row

$$\mathrm{TailRecord}_{>P}(R_\rho) \in \mathrm{TailPolicy}_{\mathcal{P}}$$

asserting that the residual coordinate is not a legal point of $\mathcal{P}_N$ and is therefore recorded beyond the certified boundary P.

Definition 9308.3 (Addressless residual status)**.** *An addressless residual status at (N, P) is the assertion that $R_\rho \neq 0$ has neither a finite address in $\pi_P(\mathcal{P}_N)$ nor a tail row in* $\mathrm{TailPolicy}_{\mathcal{P}}$*. Such a status is not a BEDC address. It is the failure of the ledger coordinate L_ρ.*

9308.2 Ledger Dichotomy

Theorem 9308.4 (Ledgered residual has a finite or tail address)**.** *Let an off-line zero packet have an off-line residual address packet at (N, P). If $R_\rho \neq 0$, then every legal address status for R_ρ is one of the following:*

$$\textit{a finite address in } \pi_P(\mathcal{P}_N),$$
$$\textit{a tail row in } \mathrm{TailPolicy}_{\mathcal{P}}\,.$$

If neither row is present, the packet violates L_ρ.

Proof. The finite source layer at (N, P) is exactly

$$\mathcal{P}_N = \{p_1, \ldots, p_N = P\}$$

by Definition 9307.5. The admissible finite addresses are precisely the displayed points of Definition 9307.18. A residual coordinate assigned to one of those points has a finite address by Definition 9308.2.

If the residual coordinate is not assigned to the displayed finite source, then the all-prime object permits only the tail policy as the source-side account of the omitted coordinate. This is the finite-or-tail visibility row of Definition 9306.13, enforced by Definition 9306.14. Thus the coordinate must be recorded in $\mathrm{TailPolicy}_{\mathcal{P}}$. If it is recorded in neither place, the residual has the addressless status of Definition 9308.3, which is exactly a failure of the ledger row L_ρ. □

Theorem 9308.5 (Finite addresses are detected)**.** *If R_ρ has a finite address at the certified finite prime circle (N, P), then the associated residual is detected by a displayed finite-circle event or by a maximal-prime Fourier sieve. If the Gauss nondegeneracy row is supplied, it is also detected by a quadratic norm sieve.*

Proof. A finite address supplies a nonzero row

$$R_N = \sum_{j=1}^{N} c_j[p_j] \neq 0.$$

If some coefficient c_j is nonzero, then Definition 9307.19 detects the residual by the singleton event

$$E_j = \{p_j \bmod P\}.$$

Independently, Definition 9307.10 supplies $t \in \mathbb{F}_P$ with nonzero Fourier readout. Under the Gauss nondegeneracy row, Definition 9307.14 turns the nonzero Fourier readout into a nonzero quadratic norm sieve readout. □

Theorem 9308.6 (Tail address cannot close silently)**.** *Let R_ρ request an address beyond P in the finite prime circle universe. Then the request either refutes the finite universe as a complete prime model, or it enters the tail ledger. If it enters no tail ledger, L_ρ fails; if it enters the tail ledger, a closed zero packet carries an append residual row.*

Proof. By Definition 9307.20, a prime $p' > P$ is not a legal finite prime-circle address under the finite universe assumption. If the row asserts that $p' > P$ is a prime source point, then the assumption that P is the maximal source prime is refuted. If the row is not admitted as a finite source point, the only legal account supplied by the all-prime object is the tail policy.

Failure to record this outside-source coordinate gives the addressless status of Definition 9308.3, hence violates L_ρ. Recording it gives an explicit append residual row. For a closed BEDC/PFZ zero packet, the refusal row of Definition 9312.3 says that such a detected or ledgered residual cannot be silently absorbed into closed status. □

9308.3 Address Theorem

Theorem 9308.7 (OffLineZeroHasFiniteOrTailAddress)**.** *Assume an off-line zero packet ρ has*

$$\mathrm{Ont}(\rho) = (\mathsf{T}_\rho, \mathsf{U}_\rho, \mathsf{L}_\rho)$$

and carries a nonzero radial residual $R_\rho \neq 0$. At any certified finite prime circle

$$\mathsf{IndexPrimePair}^{\uparrow}(N, P),$$

with $\mathrm{TailPolicy}_{\mathcal{P}}$ *supplied by the all-prime object, the residual has a legal BEDC address only in the following sense:*

$$R_\rho \text{ has a finite address in } \pi_P(\mathcal{P}_N),$$

or

$$R_\rho \text{ has a tail address in } \mathrm{TailPolicy}_{\mathcal{P}}\,.$$

There is no third legal address status. The finite alternative is detected by the finite prime circle; the tail alternative is a ledger append row rather than closed invisible data.

Proof. Apply Definition 9308.4 to the off-line residual address packet Definition 9308.1. The no-address status is excluded because it is a failure of L_ρ.

If the address is finite, then Definition 9308.5 gives a finite event, Fourier, or quadratic sieve readout. If the address is outside the finite source, then Definition 9308.6 places it in the tail ledger or refutes the finite model. Thus the two displayed alternatives exhaust the legal BEDC statuses of the nonzero radial residual. □

Corollary 9308.8 (Closed invisible off-line residual contradiction)**.** *Assume an off-line zero packet has an ontological triple, a nonzero radial residual, and a certified finite prime circle. It cannot also be a closed zero packet whose residual is invisible to every displayed finite detector and absent from the tail ledger.*

Proof. By Definition 9308.7, the residual has a finite address or a tail address. A finite address is detected by Definition 9308.5, contradicting invisibility to every displayed finite detector. A tail address enters $\mathrm{TailPolicy}_{\mathcal{P}}$, contradicting absence from the tail ledger; if it does not enter that ledger, the ontological ledger row fails. Hence the displayed closed invisible status is impossible. □

9308.4 Finite Closed-Invisible Refusal

Definition 9308.9 (Finite circle address partition)**.** *For a certified finite prime circle (N, P), define the occupied prime points and gap holes by*

$$H_N := \pi_P(\mathcal{P}_N),$$
$$G_N := \mathbb{F}_P \setminus H_N.$$

The finite circle address partition for a nonzero radial residual is the disjoint status list

$$\textit{prime-pit address in } H_N,$$
$$\textit{gap-hole address in } G_N,$$
$$\textit{tail address in } \mathrm{TailPolicy}_{\mathcal{P}},$$
$$\textit{addressless status.}$$

The first two statuses are finite-circle statuses. The third is a source-tail status beyond the certified finite boundary. The fourth is not a legal BEDC status.

Theorem 9308.10 (Finite circle address partition is exhaustive)**.** *Let $R_\rho \neq 0$ be governed by an off-line residual address packet at (N, P). Then every attempted address status for R_ρ is one of the four statuses in Definition 9308.9.*

Proof. If the residual is assigned to the finite circle, then its finite coordinate lies in $\mathbb{F}_P$. By definition,

$$\mathbb{F}_P = H_N \sqcup G_N,$$

so the coordinate is either an occupied prime point or a gap hole. If the residual is not assigned to the finite circle, then Definition 9308.4 says that legal ledgered status requires a tail row in $\mathrm{TailPolicy}_{\mathcal{P}}$. Without such a row the residual has the addressless status of Definition 9308.3. These cases are disjoint by the definitions of H_N, G_N, tail policy, and addresslessness. □

Theorem 9308.11 (Gap-hole address cannot close silently)**.** *If R_ρ has a nonzero finite-circle coordinate in G_N, then it cannot be both ledger-legal and closed-invisible.*

Proof. The gap holes G_N are part of Definition 9307.2. Thus a residual coordinate in G_N is not an unclassified finite point: it is explicitly outside the occupied prime source points of the certified layer. If the coordinate is admitted as a finite-circle function, then the Fourier basis on $\mathbb{F}_P$ detects any nonzero coordinate, and Definition 9307.13 together with the Gauss nondegeneracy row gives the corresponding quadratic readout. If the coordinate is not admitted as a finite readout, its gap-hole status must still be recorded by the ledger.

Therefore the gap coordinate has only two legal outcomes: it is detected by finite-circle readouts, or it is recorded as a gap ledger row. If the row is not recorded, L_ρ fails; if it is recorded, the zero packet carries an append residual row and is not closed in the sense of Definition 9312.3. □

Theorem 9308.12 (Off-line zero with an ontological triple cannot be closed and missed)**.** *Assume an off-line zero packet ρ has an ontological triple, carries a nonzero radial residual, and is tested at a certified finite prime circle (N, P) with the all-prime tail policy supplied. Then ρ cannot*

simultaneously satisfy all of the following:

$$\begin{gathered}\rho \text{ is a closed zero packet,}\\ \text{every occupied prime-pit event misses } R_\rho,\\ \text{every maximal-prime Fourier sieve misses } R_\rho,\\ \text{every supplied quadratic norm sieve misses } R_\rho,\\ R_\rho \text{ is absent from the finite gap and tail ledgers.}\end{gathered}$$

Proof. By Definition 9308.10, the residual has one of four address statuses.

If it has a prime-pit address in H_N, then Definition 9308.5 gives a nonzero occupied prime event, Fourier, or supplied quadratic readout. This contradicts the missed-readout assumptions.

If it has a gap-hole address in G_N, then Definition 9308.11 applies. A finite readout detects it, or the gap ledger records it. The first alternative contradicts the missed-readout assumptions, while the second contradicts absence from the finite gap ledger or closed packet status.

If it has a tail address, then Definition 9308.6 applies. The tail row either refutes the finite model as a complete prime model or enters $\mathrm{TailPolicy}_{\mathcal{P}}$. A ledgered tail contradicts tail-ledger absence or closed packet status; an unledgered tail violates L_ρ.

If it is addressless, then Definition 9308.3 says that the ledger coordinate L_ρ fails, contradicting the assumed ontological triple. Hence no address status permits the packet to be closed and missed. □

Principle 9308.13 (Finite-window obstruction boundary)**.** *The finite closed-invisible refusal is a fixed-window statement. It does not assert that every analytic off-line zero has already been placed in a specific (N, P) window. Its contract is narrower: when the analytic route supplies a nonzero residual at a certified finite prime circle, or else supplies a tail row beyond that circle, the finite address partition leaves no closed-and-missed status.*

9308.5 Global Window Bridge

Definition 9308.14 (GlobalOffLineZeroToFiniteNPWindow)**.** *The row*

$$\mathsf{GlobalOffLineZeroToFiniteNPWindow}$$

is the analytic-to-ledger bridge for an off-line zeta zero. For every off-line zero packet

$$\rho = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

it supplies a nonzero radial residual row $R_\rho \neq 0$ and exactly one of the following source accounts:

$$\begin{gathered}\text{a compact explicit-formula row whose nonzero prime side has finite support } S,\\ \text{a tail row } \mathrm{TailRecord}_{\mathrm{zeta}}(\rho, R_\rho) \in \mathrm{TailPolicy}_{\mathcal{P}}\,.\end{gathered}$$

In the finite-support case, if

$$S \subseteq \{p^k : p \in \mathcal{P},\ k \geq 1\},$$

then the bridge also supplies

$$P = \max\{p : p^k \in S\}, \qquad N = \pi(P),$$

and the certified finite prime circle

$$\mathsf{IndexPrimePair}^{\uparrow}(N, P).$$

Thus every base prime occurring in the compact prime-power residual is a displayed source point of $\mathcal{P}_N$.

Theorem 9308.15 (Compact explicit-formula support gives a finite NP window)**.** *Assume the finite-support alternative of Definition 9308.14. Then the off-line residual has a legal finite-window source account at the certified pair* (N, P) *supplied by that definition.*

Proof. The compact explicit-formula row has finite prime side by Definition 9317.1 and Definition 9317.2. Since the prime side is nonzero, its finite support S is nonempty. Each support point has the form p^k, so the set of base primes

$$\{p : p^k \in S\}$$

is finite and nonempty. Let P be its maximum and set $N = \pi(P)$. Then $P = p_N$, and every base prime in S belongs to

$$\mathcal{P}_N = \{p_1, \ldots, p_N = P\}.$$

This is precisely the legal finite source condition recorded by Definition 9307.5. The corresponding finite circle is the certified pair $\mathsf{IndexPrimePair}^{\uparrow}(N, P)$, so the residual has a finite-window source account at (N, P). □

Theorem 9308.16 (Global bridge gives finite or tail residual status)**.** *Assume* GlobalOffLineZeroToFiniteNPWindow *for an off-line zero packet* ρ*. Then* $R_\rho \neq 0$ *is either represented in a legal finite* (N, P) *window or recorded in* $\mathrm{TailPolicy}_{\mathcal{P}}$.

Proof. Unpack Definition 9308.14. The bridge first supplies $R_\rho \neq 0$. If the compact explicit-formula alternative holds, then Definition 9308.15 places the residual in a certified finite prime circle. If the compact finite-support alternative is not the supplied account, the definition supplies the tail row

$$\mathrm{TailRecord}_{\mathrm{zeta}}(\rho, R_\rho) \in \mathrm{TailPolicy}_{\mathcal{P}}\,.$$

These are exactly the finite and tail source statuses. □

Theorem 9308.17 (Global window bridge excludes closed missed off-line packets)**.** *Assume* GlobalOffLineZeroToFi*Let an off-line zero packet* ρ *have an ontological triple and be governed by* LedgerClosureRefusal*. Then* ρ *cannot be admitted as a closed zero packet whose residual is missed by every supplied finite detector and absent from the tail ledger.*

Proof. By Definition 9308.16, the residual is either finite-window represented or tail-recorded.

In the finite-window case, use the certified (N, P) supplied by Definition 9308.15. The finite prime circle and tail policy form the address packet of Definition 9308.1. Then Definition 9308.12 rules out closed status together with missed finite readouts and absent finite gap or tail ledger rows.

In the tail case, the tail row is already present in $\mathrm{TailPolicy}_{\mathcal{P}}$. By Definition 9308.6, a ledgered tail is an append residual row rather than closed invisible data; if the row is erased, the ledger coordinate of the ontological triple fails. This is also refused by Definition 9312.3. Hence no global bridge status permits a closed and missed off-line packet. □

Corollary 9308.18 (Global bridge gives the conditional fixed-half readout)**.** *Assume* GlobalOffLineZeroToFiniteNPW *for every admitted off-line zero packet, and assume admitted zeta zero packets carry ontological triples and satisfy* LedgerClosureRefusal*. Then no admitted closed zero packet is off the critical line.*

Proof. If an admitted closed zero packet were off the critical line, the global bridge would supply a nonzero residual with finite-window or tail status by Definition 9308.16. If a supplied finite detector reads the residual, or if a finite gap or tail row records it, then LedgerClosureRefusal refuses closed zero-packet admission. If no supplied finite detector reads it and no finite gap or tail ledger row records it, then Definition 9308.17 refuses the closed packet. These alternatives exhaust the finite and tail statuses supplied by the global bridge. Therefore an admitted closed zero packet cannot be off the critical line. □

Principle 9308.19 (Address audit boundary)**.** *The finite-or-tail address theorem consumes the nonzero residual row $R_\rho \neq 0$ as input. The analytic source interface is*

$$\mathsf{GlobalOffLineZeroToFiniteNPWindow}.$$

Its first component is the source row $r \neq 0 \Longrightarrow R_\rho \neq 0$; its second component places that residual in a legal finite (N, P) window or in the all-prime tail ledger. Once those rows are supplied, the address layer has no legal escape status beyond finite detection, finite gap accounting, and tail ledger admission.

Chapter 9309

Global Off-Line Zero to Finite NP Window

Definition 9308.14 names the global bridge required by the finite address refusal: an off-line zeta residual must be read either in a certified finite prime window or in the all-prime tail ledger. The present chapter supplies the compact-window proof package for that row. Its scope is conditional on the displayed explicit-formula and projection certificates; it does not turn the RH route into an unqualified global theorem. The zeta carrier, critical-strip predicate, and non-trivial zero packet rows remain theorem-facing inputs supplied by the ambient $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, and $\mathsf{ZetaZeros}^{\uparrow}$ chain.

9309.1 Radial Test Readout

Definition 9309.1 (Off-line radial distribution row)**.** *For an off-line zero*

$$\rho = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

the normalized radial row is the distribution represented by

$$R_\rho(u) := e^{ru} e^{i\gamma u}.$$

For a compact test row $h \in C_c^\infty(I)$, its readout is

$$\langle R_\rho, h \rangle_I := \int_I e^{ru} e^{i\gamma u} h(u)\, du.$$

Lemma 9309.2 (Nonzero radial mode has compact test readout)**.** *If $r \neq 0$, then there exists a compact interval I and a test row $h \in C_c^\infty(I)$ such that*

$$\langle R_\rho, h \rangle_I \neq 0.$$

Proof. Choose a nonzero nonnegative bump function

$$\eta \in C_c^\infty(I)$$

on any compact interval I, and set

$$h(u) := \eta(u) e^{-i\gamma u}.$$

Then

$$\langle R_\rho, h\rangle_I = \int_I \eta(u)e^{ru}\,du.$$

The integrand is nonnegative and is positive on a nonempty open subset of I, so the integral is nonzero. □

Theorem 9309.3 (Off-line zero supplies a compact radial witness)**.** *An off-line zero packet that carries the radial row of Definition 9309.1 supplies a compact test row with nonzero residual readout.*

Proof. Apply Definition 9309.2. The compact test row is the finite observation window through which the residual enters the zeta ledger. □

9309.2 Prime-Power Address Window

Definition 9309.4 (Compact prime-power window)**.** *Let $h \in C_c^\infty([A,B])$ be an admissible compact explicit-formula test row. Its prime-power window is*

$$S_h := \{p^k : p^k \le e^B,\ c_{p,k}(h) \neq 0\}.$$

The associated finite address ledger is

$$\mathrm{PrimePowerLedger}(h) := \{(p,k,c_{p,k}(h)) : p^k \in S_h\}.$$

Definition 9309.5 (Certified finite NP envelope)**.** *For a nonempty compact prime-power window S_h, let*

$$P_h := \max\{p : p^k \in S_h\}, \qquad N_h := \pi(P_h).$$

The certified finite NP envelope of S_h is

$$\mathsf{IndexPrimePair}^{\uparrow}(N_h, P_h)$$

together with the finite source

$$\mathcal{P}_{N_h} = \{p_1, \dots, p_{N_h} = P_h\}.$$

Theorem 9309.6 (Compact prime-power support has a certified NP envelope)**.** *If the compact explicit-formula prime side is nonzero, then its prime-power support has a certified finite NP envelope.*

Proof. By Definition 9317.1, compact support in $[A,B]$ restricts the prime side to the finite set

$$\{p^k : p^k \le e^B\}.$$

If the prime side is nonzero, then S_h is nonempty. Its base-prime set is finite and nonempty, so it has a largest element P_h. Since P_h is prime and $N_h = \pi(P_h)$, the equality $P_h = p_{N_h}$ holds. Thus the pair (N_h, P_h) is exactly the index–prime certificate of Definition 9307.1, and every base prime occurring in S_h lies in $\mathcal{P}_{N_h}$. □

Definition 9309.7 (Prime-power internal address space)**.** *For a certified finite NP envelope and compact bound B, define*

$$A_{N_h,B} := \{(p,k) : p \in \mathcal{P}_{N_h},\ k \ge 1,\ p^k \le e^B\}.$$

A compact prime-power residual has the finite internal form

$$R_h = \sum_{(p,k)\in A_{N_h,B}} c_{p,k}(h)[p,k].$$

Theorem 9309.8 (Prime-power residual has a finite address detector)**.** *If*

$$R_h = \sum_{(p,k)\in A_{N_h,B}} c_{p,k}(h)[p,k] \neq 0,$$

then a finite event in $A_{N_h,B}$ detects R_h.

Proof. Since $R_h \neq 0$, some coefficient $c_{p_0,k_0}(h)$ is nonzero. The singleton event

$$E = \{(p_0, k_0)\}$$

satisfies

$$\langle R_h, \mathbf{1}_E\rangle = c_{p_0,k_0}(h) \neq 0.$$

Thus the residual is detected by a finite address event. If the outer sum

$$C_p := \sum_{k:p^k\le e^B} c_{p,k}(h)$$

is nonzero for some prime p, the finite prime circle also detects the outer residual by Definition 9307.10. If every outer sum vanishes while $R_h \neq 0$, the nonzero singleton above lies in the internal valuation ledger at one prime, which is a legal local row of the prime-generated recursive sieve in Definition 9305.12. □

9309.3 Finite-or-Tail Bridge

Definition 9309.9 (Source-effective compact zeta window)**.** *A compact zeta window for an off-line packet ρ is source-effective when it consists of an admissible compact explicit-formula test row h, the abstract explicit-formula ledger of Definition 9317.2, and one of the two source accounts:*

$$\mathcal{P}(h) \neq 0,$$
$$\mathrm{TailRecord}_{\mathrm{zeta}}(\rho, R_\rho) \in \mathrm{TailPolicy}_{\mathcal{P}}\,.$$

The first account is finite because h is compact. The second account is the all-prime ledger row for residual material not represented in the compact finite source.

Theorem 9309.10 (PrimeProjectionCert supplies a source-effective window)**.** *Assume an off-critical mirror pair satisfies*

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-).$$

Then it supplies the finite source account of Definition 9309.9.

Proof. Unpack Definition 9312.2. The certificate supplies an admissible compact explicit-formula test row g whose correction rows are displayed and whose correction-subtracted prime side is a nonzero finite prime-power residual

$$R_S(g) = \sum_{q\in S} c_q(g)[q] \neq 0.$$

This is exactly the first source account in Definition 9309.9. The compactness of g makes the support finite by Definition 9317.1. □

Theorem 9309.11 (Source-effective window realizes the global finite NP bridge)**.** *If every off-line zeta packet has a compact radial witness and a source-effective compact zeta window, then*

$$\mathsf{GlobalOffLineZeroToFiniteNPWindow}$$

holds in the sense of Definition 9308.14.

Proof. Let ρ be an off-line zeta packet. By Definition 9309.3, it has a compact test readout of its nonzero radial row. Read the associated source-effective compact zeta window.

If the finite account $\mathcal{P}(h) \neq 0$ holds, then Definition 9309.6 supplies a certified pair

$$\mathsf{IndexPrimePair}^{\uparrow}(N_h, P_h)$$

containing every base prime of the compact prime-power support. The finite source account in Definition 9308.14 is therefore available.

If the finite account is not the source account of the compact window, then Definition 9309.9 supplies the tail row

$$\mathrm{TailRecord}_{\mathrm{zeta}}(\rho, R_\rho) \in \mathrm{TailPolicy}_{\mathcal{P}}\,.$$

This is exactly the tail account of Definition 9308.14. Hence the global bridge row holds. □

Corollary 9309.12 (Explicit-formula projection realizes the global bridge)**.** *Assume every off-line mirror pair in the zeta ledger satisfies*

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-).$$

Then the finite branch of

$$\mathsf{GlobalOffLineZeroToFiniteNPWindow}$$

is realized for every such pair.

Proof. For each off-line pair, the radial distribution row supplies the compact radial witness by Definition 9309.3. By Definition 9309.10, the prime-projection certificate supplies a source-effective compact zeta window with nonzero finite prime side. By Definition 9309.11, the global bridge row is realized, and the supplied source account is the finite branch. □

9309.4 Compact Window Exhaustion

Definition 9309.13 (Coefficiented off-line residual row)**.** *For an off-line zero*

$$\rho = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

a coefficiented radial residual row is

$$R_{\rho,c}(u) := c_\rho e^{ru} e^{i\gamma u}, \qquad c_\rho \neq 0.$$

The coefficient absorbs multiplicity and the displayed zero-side coefficient from the explicit formula. It is a source row of the zero packet only when c_ρ is supplied as nonzero data.

Lemma 9309.14 (Coefficiented radial row has compact test readout)**.** *Every coefficiented off-line residual row has a compact test row $h \in C_c^\infty(I)$ with*

$$\langle R_{\rho,c}, h\rangle_I \neq 0.$$

Proof. Choose a nonzero nonnegative bump function

$$\eta \in C_c^\infty(I)$$

and set

$$h(u) := \eta(u)e^{-i\gamma u}.$$

Then

$$\langle R_{\rho,c}, h\rangle_I = c_\rho \int_I \eta(u)e^{ru}\,du.$$

The integral is a positive real number, while $c_\rho \neq 0$. Hence the pairing is nonzero. □

Definition 9309.15 (Compact window source exhaustion)**.** *Let $h \in C_c^\infty([A,B])$ be an admissible compact explicit-formula test row. A compact window source exhaustion for h consists of the abstract explicit-formula ledger of Definition 9317.2, the compact prime-power address space*

$$A_{N,B} := \{(p,k) : p \in \mathcal{P}_N,\ k \geq 1,\ p^k \leq e^B\}$$

whenever a certified envelope

$$\mathsf{IndexPrimePair}^{\uparrow}(N,P)$$

covers the base primes occurring below e^B, and the correction accounts

$$\mathcal{A}(h)$$

for the main, archimedean, trivial-zero, and pole rows. The exhaustion condition says that a nonzero compact readout is legally accounted for only by one of the following rows:

a finite prime-power address in $A_{N,B}$,
a displayed finite-circle gap row after projection to $\mathbb{F}_P$,
a displayed correction row in $\mathcal{A}(h)$,
a tail row in $\mathrm{TailPolicy}_{\mathcal{P}}$.

If none of these rows is present, the compact window has no legal BEDC source account for the readout.

Theorem 9309.16 (Compact window exhaustion leaves no addressless residual)**.** *Assume $h \in C_c^\infty([A,B])$ has a compact window source exhaustion. If a coefficiented off-line residual row has*

$$\langle R_{\rho,c}, h\rangle \neq 0,$$

then the readout is finite-window represented, correction-ledger recorded, finite-gap recorded, or tail-recorded. If all four accounts are absent, the ledger coordinate of the zero packet fails.

Proof. The explicit-formula ledger decomposes the compact readout into the prime-power side and the displayed correction side. By Definition 9313.10, the prime side of the compact row is supported on

$$S_B = \{p^k : p^k \leq e^B\}.$$

If this side is nonzero, the finite support has a certified finite envelope by Definition 9309.6, and the internal address space $A_{N,B}$ contains every occurring prime-power coordinate. Then

Definition 9309.8 detects a nonzero finite address. If the projected coordinate lands in a finite-circle hole rather than an occupied prime point, Definition 9308.11 records the gap row or refuses the closed status.

If the finite prime-power side is not the source account for the nonzero readout, the explicit-formula identity places the nonzero contribution in a displayed correction row or in the source material not represented by the compact finite prime-power account. The former is one of the rows in $\mathcal{A}(h)$; the latter must enter $\mathrm{TailPolicy}_{\mathcal{P}}$ by Definition 9306.14. If none of the finite, gap, correction, or tail accounts is present, the readout has no legal source row, which is exactly a failure of the ledger coordinate of the zero packet. □

Theorem 9309.17 (Contrapositive global finite-window bridge)**.** *Let an off-line zeta zero packet carry a coefficiented residual row $R_{\rho,c}$, an ontological triple, and a compact window source exhaustion for the compact test row supplied by Definition 9309.14. If $R_{\rho,c}$ is absent from every certified finite (N, P) window and from every supplied tail, finite-gap, and correction ledger, then the ledger coordinate of the ontological triple fails.*

Proof. By Definition 9309.14, choose $h \in C_c^\infty([A, B])$ with

$$\langle R_{\rho,c}, h\rangle \neq 0.$$

Apply Definition 9309.16 to this compact row. Since the hypotheses exclude every finite window account and every tail, finite-gap, and correction account, the exhaustion theorem leaves only the ledger-failure alternative. This contradicts the assertion that the zero packet carries a valid ontological triple. □

Corollary 9309.18 (No finite or ledger account nullifies the residual row)**.** *Under a valid ontological triple and compact window source exhaustion, a residual row absent from all certified finite (N, P) windows and from all supplied tail, finite-gap, and correction ledgers cannot be a coefficiented off-line residual row.*

Proof. If it were such a row, then Definition 9309.17 would force the ledger coordinate of the ontological triple to fail. This contradicts the assumed validity of the triple. □

Principle 9309.19 (Fixed-window discipline for the global bridge)**.** *The bridge fixes the compact test row before any finite source certificate is read. Thus the support bound B, the prime-power address set $A_{N,B}$, and the certified pair $\mathsf{IndexPrimePair}^{\uparrow}(N, P)$ belong to the same observation window. The argument is a fixed-window ledger exhaustion statement, not a claim that an unspecified infinite zeta row has been collapsed into one finite modulus.*

9309.5 Closed-Packet Consequence

Theorem 9309.20 (Global bridge feeds the finite address refusal)**.** *Assume*

$$\mathsf{GlobalOffLineZeroToFiniteNPWindow}.$$

If an off-line zero packet has an ontological triple and is governed by $\mathsf{LedgerClosureRefusal}$*, then it cannot be both closed and missed by all finite prime-power, finite prime-circle, and tail-ledger readouts.*

Proof. By Definition 9308.16, the residual is either represented in a certified finite (N, P) window or recorded in $\mathrm{TailPolicy}_{\mathcal{P}}$. In the finite case, Definition 9309.8 detects the internal prime-power residual, and Definition 9308.12 rules out the closed missed status at the finite prime circle. In the tail case, Definition 9308.6 rules out closed invisible status. Both branches are governed by Definition 9312.3. □

Principle 9309.21 (Global window audit boundary)**.** *The global bridge is not a bare assertion that every zero has already been placed in one finite modulus. It is the compact-test and explicit-formula ledger statement that a nonzero off-line radial mode has a finite prime-power source account whenever the compact prime side is nonzero, and otherwise has an all-prime tail account. The finite address refusal then applies only after this source account is displayed.*

Chapter 9310

Finite Prime Circle Off-Line Contradiction

chapter 9309 supplies the source bridge from an off-line radial row to a finite prime-power window or to the all-prime tail ledger. The present chapter packages the conditional consequence for legal zeta zero packets: if completed zeta and its non-trivial zeros are admitted as generated BEDC objects with ontological triples and closure ledgers, then an off-line zero cannot also be closed.

9310.1 Legal Zeta Zero Packets

Definition 9310.1 (Licensed zeta zero object)**.** *A licensed zeta zero object is not a bare complex number satisfying $\zeta(\rho) = 0$. It is a non-trivial zero packet carried by the same NameCert route as the completed zeta object:*

$$\mathsf{NameCert}_{\mathsf{ZetaBasic}^{\uparrow}}, \qquad \mathsf{NameCert}_{\mathsf{ZetaCont}^{\uparrow}}, \qquad \mathsf{NameCert}_{\mathsf{CritStrip}^{\uparrow}}, \qquad \mathsf{NameCert}_{\mathsf{ZetaZeros}^{\uparrow}}.$$

The route exposes, for the zero packet, source, pattern, classifier, stability, and ledger coordinates:

$$\mathsf{SourceSpec}, \quad \mathsf{PatternSpec}, \quad \mathsf{ClassifierSpec},$$
$$\mathsf{StabilityCert}, \quad \mathsf{LedgerPolicy}.$$

Limit-shaped or continuation-shaped coordinates are licensed only together with the ledger rows that record branch, pole, correction, gap, and tail data.

Definition 9310.2 (ZetaZeroObjectsHaveOntTriple)**.** *The theorem-facing row*

$$\mathsf{ZetaZeroObjectsHaveOntTriple}$$

asserts that every licensed zeta zero object ρ carries an ontological triple

$$\mathrm{Ont}(\rho) = (\mathsf{T}_\rho, \mathsf{U}_\rho, \mathsf{L}_\rho).$$

Here T_ρ is the generated history through the zeta NameCert route, U_ρ is the unit and radial readout separating unit phase from scale residuals, and L_ρ is the ledger policy that assigns every nonzero residual to a finite, gap, tail, or displayed correction account.

Definition 9310.3 (Zeta legal zero packet)**.** *A zeta legal zero packet is a completed non-trivial zero packet ρ carried by the generated zeta closure row Definition 9330.2, equipped with:*

$$\mathrm{Ont}(\rho) = (\mathsf{T}_\rho, \mathsf{U}_\rho, \mathsf{L}_\rho),$$
$$\mathsf{GlobalOffLineZeroToFiniteNPWindow},$$
$$\mathsf{LedgerClosureRefusal}.$$

The first row is the ontological triple of Definition 9303.1. The second row is the finite or tail source bridge of Definition 9308.14. The third row is the closed-packet refusal of Definition 9312.3.

Theorem 9310.4 (Licensed triple gives a legal zero packet)**.** *Assume a licensed zeta zero object is carried by* ZetaBEDCClosure *and satisfies*

ZetaZeroObjectsHaveOntTriple, GlobalOffLineZeroToFiniteNPWindow, LedgerClosureRefusal.

Then it is a zeta legal zero packet.

Proof. The closure row supplies the generated completed zeta carrier and the non-trivial zero packet. By Definition 9310.2, the packet carries the displayed ontological triple. The global bridge and closure refusal rows are exactly the second and third rows required in Definition 9310.3. Hence all coordinates of a zeta legal zero packet are present. □

Definition 9310.5 (Closed zeta zero packet)**.** *A zeta legal zero packet is closed when no nonzero residual row remains as an unprocessed append row. A detected residual that is not recorded violates L_ρ; a detected residual that is recorded is a continuing ledger row rather than closed zero-packet status.*

Principle 9310.6 (Zeta-shaped statements are theorem targets)**.** *The rows* ZetaBEDCClosure, GlobalOffLineZeroToFiniteNPWindow, *and the resulting fixed-half readout are theorem-facing rows. They are not seed names for zeta or for* RH*. Each must be supplied by the Dirichlet, Euler product, analytic-continuation, explicit-formula, finite-window, and ledger certificates on which it depends.*

9310.2 Off-Line Residual Refusal

Theorem 9310.7 (Off-line legal zero packets expose residuals)**.** *If a zeta legal zero packet is off the critical line, then it supplies a nonzero radial residual with finite-window or tail status.*

Proof. An off-line zero has a nonzero radial row by Definition 9309.1 and Definition 9309.3. The global bridge row in Definition 9310.3 applies to that residual. By Definition 9308.16, the residual is represented in a legal finite (N, P) window or is recorded in $\mathrm{TailPolicy}_{\mathcal{P}}$. □

Theorem 9310.8 (Finite prime circle off-line contradiction)**.** *A zeta legal zero packet cannot be both off the critical line and closed.*

Proof. Suppose that a zeta legal zero packet ρ is off the critical line. By Definition 9310.7, its radial residual has finite-window or tail status.

If the residual is finite-window represented, then Definition 9309.20 applies: the internal prime-power detector, finite prime-circle detector, gap ledger, or tail ledger prevents the packet from being closed and missed. If the residual is tail-recorded, then Definition 9308.6 says that the tail row is an append ledger row; erasing it would violate L_ρ. Both branches are precisely the alternatives refused by Definition 9312.3. Therefore ρ cannot be a closed packet. □

Theorem 9310.9 (Off-line licensed triple cannot be closed)**.** *Let ρ be a licensed zeta zero object carried by* ZetaBEDCClosure*. Assume*

ZetaZeroObjectsHaveOntTriple, GlobalOffLineZeroToFiniteNPWindow, LedgerClosureRefusal.

If ρ is off the critical line, then ρ cannot be a closed zeta zero packet.

Proof. By Definition 9310.4, the licensed object is a zeta legal zero packet. The off-line hypothesis then activates Definition 9310.8, which refuses simultaneous off-line and closed status. □

Corollary 9310.10 (Legal closed zeta zeros have fixed-half readout)**.** *Every closed zeta legal zero packet has*

$$\mathrm{Re}(\rho) = \frac{1}{2}.$$

Proof. If $\mathrm{Re}(\rho) \neq 1/2$, then ρ is off the critical line. By Definition 9310.8, it cannot be closed. Hence every closed zeta legal zero packet has no off-line radial displacement. Equivalently, Definition 9266.2 gives $\mathrm{Re}(\rho) = 1/2$ from the unit Li readout. □

9310.3 Conditional RH Interface

Definition 9310.11 (PrimeCircleBEDCRoute)**.** *The route*

PrimeCircleBEDCRoute

consists of the following rows:

ZetaBEDCClosure,
every admitted non-trivial zero is a zeta legal zero packet,
every admitted non-trivial zero packet is closed,
GlobalOffLineZeroToFiniteNPWindow.

It is a conditional route: each row is an explicit certificate or theorem target, not a primitive assertion about zeta.

Theorem 9310.12 (PrimeCircleBEDCRoute gives the BEDC RH predicate)**.** *Assume* PrimeCircleBEDCRoute*. Then every admitted non-trivial zero satisfies the fixed-half predicate of Definition 9239.7.*

Proof. Let ρ be an admitted non-trivial zero packet. By the route, it is a zeta legal zero packet and it is closed. Apply Definition 9310.10. The fixed-half readout is exactly the OnCritLine conclusion in Definition 9239.7. □

Corollary 9310.13 (PrimeCircle conditional RH)**.** *Under* PrimeCircleBEDCRoute*, the BEDC RH statement is inhabited as a conditional theorem target.*

Proof. The predicate RH is the Π-statement over admitted non-trivial zero packets recorded in Definition 9239.7. The preceding theorem supplies the fixed-half output for each such packet. Thus the conditional route inhabits the BEDC RH statement. □

Corollary 9310.14 (NameCert triple conditional RH)**.** *Assume every admitted non-trivial zero is a licensed zeta zero object carried by* ZetaBEDCClosure*, assume every such object satisfies*

ZetaZeroObjectsHaveOntTriple, GlobalOffLineZeroToFiniteNPWindow, LedgerClosureRefusal,

and assume every admitted non-trivial zero packet is closed. Then every admitted non-trivial zero satisfies

$$\mathrm{Re}(\rho) = \frac{1}{2}.$$

Proof. Let ρ be an admitted non-trivial zero. If ρ were off the critical line, then Definition 9310.9 would rule out closed status. This contradicts the closed-packet assumption. Hence ρ is not off the critical line, and Definition 9266.2 gives the fixed-half readout. □

Principle 9310.15 (Prime circle contradiction audit boundary)**.** *The contradiction is conditional on the legal-object rows. It proves that the simultaneous package*

off-line zero + *ontological triple* + *closed zero packet* + GlobalOffLineZeroToFiniteNPWindow

is impossible. It does not assert that ZetaBEDCClosure*, zero-packet closure, or the global window bridge are primitive facts; those remain the explicit certificates required before the conditional route can be read as a BEDC proof of* RH*.*

Principle 9310.16 (Ontological triple construction target)**.** *The finite contradiction has been reduced to the construction of*

ZetaZeroObjectsHaveOntTriple.

That row is not a seed assertion about zeros. It must be constructed from the NameCert data for ZetaBasic$^{\uparrow}$*,* ZetaCont$^{\uparrow}$*,* CritStrip$^{\uparrow}$*, and* ZetaZeros$^{\uparrow}$*, together with the ledger witnesses for continuation, limits, correction terms, finite prime windows, gaps, and tails. Once that row is supplied, an off-line zero can only be a detected residual, a ledger append, or a ledger violation; none of those statuses is a closed legal zero packet.*

Chapter 9311

Analytic Continuation Ontological Triple

chapter 9303 requires every zeta-route tool to expose an ontological triple. The analytic continuation operation of chapter 9341 and the zeta continuation application of chapter 9342 therefore cannot function as bare analytic shortcuts inside the RH route. They must preserve source history, unit or phase structure, and ledger visibility. This chapter records that requirement as a proof-engineering interface for the prime-generated recursive sieve.

9311.1 Continuation Certificate

Definition 9311.1 (Analytic continuation certificate)**.** *Let $f_0 : \Omega_0 \to \mathbb{C}$ be a licensed analytic object and let $\Omega_0 \subseteq \Omega_1$. An analytic continuation certificate for an extension*

$$f_0 \rightsquigarrow f_1 : \Omega_1 \to \mathbb{C}$$

is a packet

$$\mathrm{ACCert}(f_0 \rightsquigarrow f_1) := (\mathrm{SourceSpec}, \mathrm{PatternSpec}, \mathrm{ClassifierSpec}, \mathrm{StabilityCert}, \mathrm{LedgerPolicy}).$$

Here SourceSpec *records f_0, Ω_0, and the history of f_0;* PatternSpec *records equality of holomorphic germs on overlaps;* ClassifierSpec *records the branch, monodromy, and same-germ classifier used by the extension;* StabilityCert *records identity-theorem and local holomorphy witnesses; and* LedgerPolicy *records branch cuts, singularities, tails, poles, zeros, and residual rows.*

Definition 9311.2 (Continuation ontological triple)**.** *For a certified continuation $f_0 \rightsquigarrow f_1$, its ontological triple is*

$$\mathrm{Ont}(f_0 \rightsquigarrow f_1) := (\mathsf{T}_{\mathrm{AC}}, \mathsf{U}_{\mathrm{AC}}, \mathsf{L}_{\mathrm{AC}}),$$

where

$$\mathsf{T}_{\mathrm{AC}} : \quad f_1|_{\Omega_0} = f_0$$

is the source-history row,

$$\mathsf{U}_{\mathrm{AC}}$$

is the row saying that unit phase, positivity, or orthogonal energy readouts are preserved unless a displayed residual row is produced, and

$$\mathsf{L}_{\mathrm{AC}}$$

is the continuation ledger for branches, tails, singularities, monodromy, zero and pole rows, and residual projections.

Principle 9311.3 (No uncertified analytic continuation)**.** *An analytic continuation used by the RH route is not available as a primitive operation merely because an analytic formula can be written. It is available only through an operation or application certificate whose source, overlap pattern, classifier, stability, and ledger rows are displayed. Otherwise the continued object has no BEDC license.*

Theorem 9311.4 (Certified continuation preserves the ontological triple)**.** *Let f_0 be a BEDC-licensed analytic object with* $\mathrm{Ont}(f_0)$*. If $f_0 \rightsquigarrow f_1$ has an analytic continuation certificate, then f_1 has an ontological triple.*

Proof. The source row of ACCert gives the history extension from f_0 to f_1, so T_{f_1} is defined by appending the continuation bridge to T_{f_0}. The pattern and stability rows say that f_1 agrees with f_0 as the same analytic germ on the overlap and is uniquely determined by the continuation witnesses; hence unit or phase data from the source cannot be silently changed. If a nonunit radial or branch-dependent row appears, it is not part of the unit row and must be recorded by the ledger policy. The ledger row of ACCert records branch, tail, singular, zero, pole, monodromy, and residual data. These three rows define $\mathrm{Ont}(f_1)$. □

9311.2 Conservative Zeta Continuation

Definition 9311.5 (Conservative zeta continuation certificate)**.** *A conservative zeta continuation certificate is an analytic continuation certificate*

$$\mathrm{ACCert}(\zeta_0 \rightsquigarrow \zeta_1)$$

from the right-half-plane PZG zeta source to a continued zeta object, together with three transport rows:

$$(\mathsf{PZGSource}, \mathsf{UnitPhase}, \mathsf{NoScaleRegister}).$$

The row PZGSource *says that the continued object is the same germ as the tagged PZG zeta family and its forgetful scalar readout of Definition 9293.11 and Definition 9293.13. The row* UnitPhase *says that the PZG phase readout remains a unit phase row under continuation unless a displayed residual ledger records otherwise. The row* NoScaleRegister *says that the continuation does not supply an independent positive-real scale register capable of paying the instruction-scale residual of Definition 9294.5.*

Principle 9311.6 (Completion factors are ledger rows)**.** *The archimedean and pole factors in the completed xi execution trace are visible completion rows. They may provide the functional equation, reflection, pole cancellation, and archimedean boundary data recorded in Definition 9293.6; they are not free per-prime scale registers inside the original zeta source. Any use of a completion factor as a scale-balancing mechanism must therefore appear as a named completion or residual ledger row, not as an unrecorded internal cancellation of the PZG source.*

Theorem 9311.7 (Conservative continuation forbids scale-register payment)**.** *Assume a conservative zeta continuation certificate for the continued zeta object. Then an off-half instruction-scale residual cannot be closed by analytic continuation itself. It must either remain in the continuation ledger or be closed by a displayed same-packet zeta ledger row.*

Proof. By Definition 9311.5, the continued object transports the right-half-plane PZG source and its forgetful scalar readout; it does not replace that source by a different register system. The unit-phase row transports the fact that phase instructions have modulus 1. By Definition 9294.6, such instructions cannot cancel a nonzero instruction-scale residual. The NoScaleRegister row forbids analytic continuation from supplying an independent positive-real scale register. Hence any off-half scale residual is not paid by the continuation operation itself. By the ledger policy in Definition 9311.1, it must be recorded as residual data unless a displayed same-packet zeta ledger row closes it. □

Corollary 9311.8 (Conservative single-ledger continuation gives fixed-half)**.** *If a continued nontrivial zero packet is accepted through a conservative zeta continuation certificate and is closed by the single-ledger zero principle, then it has fixed-half readout.*

Proof. Conservative continuation forbids scale-register payment by Definition 9311.7. The remaining same-packet closure condition is exactly the hypothesis consumed by Definition 9294.11. Therefore the zero packet has fixed-half readout. □

9311.3 Eta Parity Bridge

Definition 9311.9 (Eta parity sieve bridge)**.** *The eta parity sieve bridge is the $p = 2$ classifier*

$$n \longmapsto n \bmod 2$$

together with the unit character

$$\varepsilon_2(n) := (-1)^{n-1}.$$

It produces the Dirichlet eta series

$$\eta(s) := \sum_{n \geq 1} \varepsilon_2(n) n^{-s}.$$

Theorem 9311.10 (Eta bridge preserves BEDC rows)**.** *The eta parity sieve bridge preserves the time, unit, and ledger rows of the right-half-plane zeta source.*

Proof. The time row is the appended parity classifier $n \mapsto n \bmod 2$. The unit row is the character identity

$$|\varepsilon_2(n)| = 1.$$

Thus the bridge changes the phase readout but does not add an unledgered radial scale. The ledger row is the displayed correction factor in Definition 9342.3:

$$\eta(s) = (1 - 2^{1-s})\zeta(s)$$

on the overlap where the relation is derived from the absolutely convergent source. The parity classifier, the factor $1 - 2^{1-s}$, and the denominator-zero and pole rows of the continuation application are therefore visible ledger data. □

Theorem 9311.11 (Eta continuation supplies a certified critical-strip zeta object)**.** *On the strip*

$$0 < \mathrm{Re}(s) < 1,$$

the eta quotient

$$\zeta_\eta(s) := \frac{\eta(s)}{1 - 2^{1-s}}$$

is a certified continuation of the right-half-plane zeta source whenever the eta continuation application certificate is supplied.

Proof. The eta function is holomorphic on $\mathrm{Re}(s) > 0$ by Definition 9342.2. The eta-zeta relation of Definition 9342.3 gives equality with the right-half-plane zeta source on the overlap $\mathrm{Re}(s) > 1$. On $0 < \mathrm{Re}(s) < 1$, the denominator $1 - 2^{1-s}$ has no zero, so the eta quotient defines the displayed analytic object. The certificate rows are precisely the eta relation, overlap identity, denominator and pole ledger, and continuation law recorded in Definition 9342.10 and Definition 9342.11. Therefore the object is a certified continuation in the sense of Definition 9311.1. □

9311.4 Sieve Ledger Preservation

Definition 9311.12 (AnalyticContinuationPreservesOntTriple)**.** *The row*

AnalyticContinuationPreservesOntTriple

asserts that every analytic continuation used by the zeta route carries an analytic continuation certificate and therefore transports ontological triples from source objects to continued objects.

Theorem 9311.13 (Ontological continuation preserves prime-sieve ledger)**.** *Assume* AnalyticContinuationPreserves *for the continuation from the right-half-plane prime-generated phase limit to the critical-strip zeta object. Then*

AnalyticContinuationPreservesPrimeSieveLedger

holds for the prime-generated recursive sieve route.

Proof. By Definition 9305.11, the right-half-plane normalized zeta phase is an inverse limit of prime-generated local probability sieves. Its source rows are finite prime windows, local valuation distributions, phase readouts, and tail-ledger entries. By Definition 9311.4, the certified continuation transports source history, unit rows, and ledger policy into the continued object. Hence the finite prime windows, local valuation rows, unit phase readouts, residual projections, and excluded-mass ledger entries remain available as continued certified rows, which is exactly the row defined in Definition 9305.12. □

Theorem 9311.14 (Continuation cannot hide a radial residual)**.** *Assume* AnalyticContinuationPreservesOntTriple. *If an off-critical zero contributes a nonzero radial residual to the continued zeta object, then the residual must appear as a ledgered continuation row or as a finite prime-side residual row.*

Proof. A nonzero radial mode is not a unit phase row. Under Definition 9311.2, such a nonunit row cannot be absorbed into U_{AC} unless it is recorded as visible residual data. The ledger policy of Definition 9311.1 records residuals, zeros, poles, branch data, and tails. Therefore the radial residual either appears in the continuation ledger or is transported into the finite prime-side residual rows whose preservation is given by Definition 9311.13. □

Principle 9311.15 (Continuation audit boundary)**.** *The eta bridge and the general analytic-continuation operation remove analytic continuation as a hiding place for radial scale. They do not prove that an off-critical zero produces a finite prime residual. They only certify that, once such a residual is present in the continued zeta object, it must be preserved as source, unit, or ledger data rather than treated as an unlicensed feature of the continuation.*

Chapter 9312

Off-Line Zero to Prime Sieve Contradiction

chapter 9318 isolates the spectral packet that an off-critical mirror pair must expose, while chapter 9317 records the compact-test bridge from spectral scale to a finite prime-power row. chapter 9305 proves that finite prime residuals are detected by local sieves generated by primes. The present chapter packages these interfaces into a single BEDC proof object:

$$\mathsf{OffLineZeroToPrimeSieveContradiction}^{\uparrow}.$$

Its role is not to assert the Riemann hypothesis as a closed theorem. It identifies the exact certificate chain under which an off-line zero cannot be admitted as a closed BEDC/PFZ zero packet.

9312.1 Contradiction Packet

Definition 9312.1 (OffLineZeroToPrimeSieveContradiction)**.** *The packet*

$$\mathsf{OffLineZeroToPrimeSieveContradiction}$$

consists of the following rows:

$$\begin{gathered}\mathsf{AnalyticContinuationPreservesOntTriple},\\ \mathsf{AnalyticContinuationPreservesPrimeSieveLedger},\\ \mathsf{ZetaFixedWindowDominance},\\ \mathsf{PrimeProjectionCert},\\ \mathsf{PrimeGeneratedRecursiveSieve},\\ \mathsf{LedgerClosureRefusal}.\end{gathered}$$

The first two rows license the continued zeta object as a source-preserving, unit-aware, ledger-visible object. The fixed-window row supplies the off-line spectral packet. The prime-projection row turns that spectral packet into a nonzero finite prime-power residual. The recursive sieve row detects every such finite residual by a local prime-generated classifier. The ledger-refusal row says that a closed zero packet cannot carry either an unrecorded detected residual or an append row that has not closed.

Definition 9312.2 (PrimeProjectionCert)**.** *For an off-critical mirror pair*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma, \qquad r \neq 0,$$

the certificate

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-)$$

asserts that the fixed-window radial readout supplied by

$$\mathsf{ZetaFixedWindowDominance}$$

has an admissible compact explicit-formula test row g for which the explicit-formula ledger

$$\mathcal{Z}(g) = \mathcal{P}(g) + \mathcal{A}(g)$$

produces a nonzero finite prime-power residual

$$R_S(g) = \sum_{q \in S} c_q(g)[q] \neq 0.$$

Equivalently, the archimedean, trivial-zero, and pole correction rows are displayed and do not absorb the fixed radial readout.

Definition 9312.3 (LedgerClosureRefusal)**.** *The row*

$$\mathsf{LedgerClosureRefusal}$$

asserts that a legal closed BEDC/PFZ zero packet cannot satisfy either of the following conditions:

a detected prime-sieve residual is absent from the ledger,

a detected prime-sieve residual is present as an unclosed append row.

Thus a detected residual is fatal to closed zero-packet admission: if it is not ledgered, the ledger policy fails; if it is ledgered, the packet has not closed.

9312.2 From Spectral Packet to Prime Residual

Theorem 9312.4 (Fixed packet gives a finite prime residual)**.** *Assume* $\mathsf{ZetaFixedWindowDominance}$ *and*

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-)$$

for an off-critical mirror pair. Then the pair supplies a nonzero finite prime-power residual

$$R_S(g) = \sum_{q \in S} c_q(g)[q].$$

Proof. The fixed-window dominance row unpacks through Definition 9318.16. For the displayed mirror pair it supplies a finite spectral window, a mirror-odd differential classifier, a compact observation interval, and a positive dominance margin. By Definition 9318.8, the corresponding classifier readout is nonzero and contains a nonunit radial row.

The prime-projection certificate supplies an admissible compact explicit-formula test row whose correction rows are ledgered and do not absorb that radial readout. Therefore the explicit-formula identity

$$\mathcal{Z}(g) = \mathcal{P}(g) + \mathcal{A}(g)$$

has a nonzero correction-subtracted prime side. This is the prime-effective realization of Definition 9318.10. Applying Definition 9318.11 gives the displayed nonzero finite residual. □

Corollary 9312.5 (Explicit-formula isolation is a projection certificate)**.** *The hypotheses of Definition 9317.9 provide an instance of*

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-)$$

for the corresponding off-critical mirror pair.

Proof. Definition 9317.9 starts from the abstract explicit-formula ledger and ArchNullZeroIsolation. It produces a nonzero finite prime-power residual from an off-critical zero. This is precisely the prime-projection certificate: the compact test row is admissible, the correction row is null in the stated ledger, and the prime side is finite and nonzero. □

9312.3 Prime-Generated Detection

Theorem 9312.6 (Finite residual enters a prime-generated sieve)**.** *Assume* PrimeGeneratedRecursiveSieve*. If*

$$R_S = \sum_{j=1}^{N} c_j[q_j] \neq 0$$

is a finite prime-power residual with distinct support points, then there are a prime p and a character

$$\chi : \mathbb{F}_p^\times \to S^1$$

such that

$$\sum_{j=1}^{N} c_j \chi(q_j \bmod p) \neq 0.$$

The readout is a legal finite row of the prime-generated recursive sieve ledger.

Proof. The existence of the detecting prime and character is exactly Definition 9305.5. The legality of the row follows from Definition 9305.2: a local residue or character readout generated by a prime is one of the permitted finite classifier rows, and the recursive sieve step records the selected prime, event or character, readout, source residual, and reason. □

Corollary 9312.7 (Signed residuals also enter the recursive sieve)**.** *Under* PrimeGeneratedRecursiveSieve*, every finite prime negative residual of Definition 9317.16 has a prime-generated classifier readout.*

Proof. By Definition 9317.18, a finite prime negative residual supplies a nonzero finite prime-classifier residual. Apply Definition 9312.6. □

9312.4 Contradiction Route

Theorem 9312.8 (Off-line zero to prime-sieve contradiction)**.** *Assume*

$$\mathsf{OffLineZeroToPrimeSieveContradiction}.$$

If every off-critical mirror pair in the continued zeta ledger satisfies

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-),$$

then no off-critical zero is admitted as a legal closed BEDC/PFZ zero packet.

Proof. Suppose that a legal closed packet admits an off-critical zero. Its mirror pair has the form

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma, \qquad r \neq 0.$$

The continuation rows in Definition 9312.1 keep this pair inside the certified zeta ledger and prevent analytic continuation from hiding a radial residual, as expressed by Definition 9311.13 and Definition 9311.14.

By ZetaFixedWindowDominance and the prime-projection certificate for the mirror pair, Definition 9312.4 gives a nonzero finite prime-power residual R_S. By Definition 9312.6, a local prime-generated classifier detects R_S with nonzero readout.

That detected readout has only two possible ledger statuses. If it is not recorded, the prime-sieve ledger policy fails. If it is recorded, the zero packet carries an append residual row and is not closed. Both alternatives contradict LedgerClosureRefusal. Therefore a legal closed BEDC/PFZ zero packet cannot admit an off-critical zero. □

Corollary 9312.9 (Conditional fixed-half readout)**.** *Under the hypotheses of Definition 9312.8, every admitted closed nontrivial zero has fixed-half readout:*

$$\mathrm{Re}(\rho) = \frac{1}{2}.$$

Proof. The theorem excludes every closed admitted off-critical zero. Hence an admitted closed nontrivial zero must have no radial displacement from the half line. Equivalently, by Definition 9266.2, its Li coordinate lies on the unit circle precisely when

$$\mathrm{Re}(\rho) = \frac{1}{2}.$$

□

9312.5 Audit Boundary

Principle 9312.10 (Finite sieve side is closed under the packet)**.** *Inside*

OffLineZeroToPrimeSieveContradiction,

the finite arithmetic part is not the source of the obstruction. Once a nonzero finite prime-power residual is displayed, the prime-generated recursive sieve detects it by Definition 9305.5; the detected row is then governed by the ledger dichotomy of Definition 9312.3.

Principle 9312.11 (Analytic certificate boundary)**.** *The analytic burden of the packet is concentrated in the two certificate rows*

ZetaFixedWindowDominance *and* PrimeProjectionCert.

The first row is the fixed spectral-window dominance statement of Definition 9318.16. The second row is the compact explicit-formula projection statement of Definition 9312.2. Once those rows are supplied, the contradiction route is finite, prime-generated, and ledgered.

Chapter 9313

Prime Projection Certificate

Definition 9312.2 names the bridge from a fixed off-line spectral packet to a finite prime-power residual. This chapter expands that bridge into a fixed-window certificate. The argument has three parts: a nonzero differential readout is paired with a compact test row, the differential classifier is moved onto the test row by transposition, and the compact Chebyshev expansion leaves a finite prime-power row after the displayed main and correction functionals are killed.

9313.1 Test-Row Projection

Definition 9313.1 (Fixed-window residual functional)**.** *Let Ψ be a normalized prime-distribution curve, let $I \subset \mathbb{R}$ be a compact interval, and let $P_\pm$ be a mirror-odd differential classifier. The fixed-window residual functional is*

$$\mathcal{F}_{P_\pm,I}(g) := \langle P_\pm(D)\Psi, g\rangle_I$$

for test rows $g \in C_c^\infty(I)$.

Lemma 9313.2 (Fixed-window test separation)**.** *Assume*

$$\inf_{u\in I} |P_\pm(D)\Psi(u)| > 0.$$

Then there exists $g \in C_c^\infty(I)$ such that

$$\mathcal{F}_{P_\pm,I}(g) \neq 0.$$

Proof. If $\mathcal{F}_{P_\pm,I}$ vanished on every test row, then $P_\pm(D)\Psi$ would be the zero distribution on the interior of I. This contradicts the displayed positive lower bound. Hence some compactly supported test row has nonzero pairing. □

Lemma 9313.3 (Differential classifier transposition)**.** *Let $P_\pm$ be a polynomial and let $g \in C_c^\infty(I)$. Define*

$$h := P_\pm(-D)g.$$

Then $h \in C_c^\infty(I)$ and

$$\langle P_\pm(D)\Psi, g\rangle_I = \langle \Psi, h\rangle_I.$$

Proof. Compact support removes boundary terms. Each monomial D^m moves from Ψ to g with the sign $(-1)^m$ under integration by parts. Linearity over the polynomial $P_\pm$ gives the displayed identity, and differentiating a compactly supported smooth test row preserves compact support in I. □

Theorem 9313.4 (Dominance supplies a compact nonzero prime-distribution test)**.** *If an off-critical mirror pair has a fixed window dominance certificate, then there exists $h \in C_c^\infty(I)$ such that*

$$\langle \Psi, h \rangle_I \neq 0.$$

Proof. By Definition 9318.8, the full curve has a nonzero classifier readout on the fixed interval. Apply Definition 9313.2 to choose g with nonzero pairing against $P_\pm(D)\Psi$. Then apply Definition 9313.3 and set $h = P_\pm(-D)g$. □

9313.2 Main and Correction Null Rows

Definition 9313.5 (Chebyshev main functional)**.** *For $h \in C_c^\infty([A, B])$, define the Chebyshev main functional by*

$$M(h) := \int_A^B h(u)e^{u/2}\, du.$$

Definition 9313.6 (MainArchNullCert)**.** *For a fixed-window residual functional $\mathcal{F}_{P_\pm,I}$, the certificate*

$$\mathsf{MainArchNullCert}(P_\pm, I)$$

asserts that there are finitely many displayed correction functionals

$$C_1, \ldots, C_N : C_c^\infty(I) \to \mathbb{C}$$

containing the main row and the archimedean, trivial-zero, and pole rows relevant to the chosen explicit-formula ledger, and that the prime-distribution functional

$$h \longmapsto \langle \Psi, h \rangle_I$$

does not belong to their linear span on the admissible compact test space.

Lemma 9313.7 (Finite-constraint separation)**.** *Let V be a complex vector space, let $F : V \to \mathbb{C}$ be a linear functional, and let $L_1, \ldots, L_N : V \to \mathbb{C}$ be linear functionals. If*

$$F \notin \operatorname{span}(L_1, \ldots, L_N),$$

then there exists $v \in V$ such that

$$L_j(v) = 0 \qquad (1 \leq j \leq N)$$

and

$$F(v) \neq 0.$$

Proof. Let

$$K := \bigcap_{j=1}^{N} \ker L_j.$$

If F vanished on K, then F would factor through the finite quotient determined by the map

$$v \longmapsto (L_1(v), \ldots, L_N(v)).$$

Hence F would be a linear combination of the L_j, contrary to the hypothesis. Therefore F is nonzero on K, giving the required vector v. □

Theorem 9313.8 (MainArchNullCert gives a nonzero null-corrected test)**.** *Assume* $\mathsf{MainArchNullCert}(P_\pm, I)$ *and suppose the fixed-window dominance certificate supplies a nonzero compact prime-distribution test row as in Definition 9313.4. Then there exists* $h \in C_c^\infty(I)$ *such that*

$$\langle \Psi, h \rangle_I \neq 0$$

and every displayed main, archimedean, trivial-zero, and pole correction functional in the certificate vanishes on h*.*

Proof. Use Definition 9313.7 on the admissible compact test space with

$$F(h) = \langle \Psi, h \rangle_I$$

and with $L_1, \ldots, L_N$ equal to the displayed correction functionals of Definition 9313.6. The nonspan row in that definition is exactly the hypothesis needed by the separation lemma. □

9313.3 Finite Prime Projection

Definition 9313.9 (Compact Chebyshev prime projection)**.** *Let*

$$\Psi(u) = e^{-u/2}\big(\psi(e^u) - e^u\big)$$

and let $h \in C_c^\infty([A, B])$*. The compact Chebyshev prime projection is the finite expansion*

$$\langle \Psi, h \rangle = \sum_{p^k \le e^B} c_{p,k}(h) - M(h),$$

where

$$c_{p,k}(h) := \log p \int_{\max(A, \log p^k)}^{B} h(u) e^{-u/2}\, du$$

and $M(h)$ *is the functional of Definition 9313.5.*

Lemma 9313.10 (Compact support makes the prime side finite)**.** *For* $h \in C_c^\infty([A, B])$*, the prime side of Definition 9313.9 is supported on the finite set*

$$S_B := \{p^k : p^k \le e^B\}.$$

Proof. The Chebyshev row

$$\psi(e^u) = \sum_{p^k \le e^u} \log p$$

contains no prime power larger than e^B on the interval $[A, B]$. There are only finitely many prime powers bounded by e^B, so the displayed support is finite. □

Theorem 9313.11 (Null-corrected compact test gives finite prime residual)**.** *Let* $h \in C_c^\infty([A, B])$ *satisfy*

$$\langle \Psi, h \rangle \neq 0$$

and

$$M(h) = 0.$$

Then

$$R_S(h) := \sum_{q \in S_B} c_q(h)[q]$$

is a nonzero finite prime-power residual.

Proof. By Definition 9313.9, the main-null condition gives

$$\langle \Psi, h \rangle = \sum_{p^k \le e^B} c_{p,k}(h).$$

The left side is nonzero, so the finite prime-power sum is nonzero. By Definition 9313.10, its support is finite. □

Theorem 9313.12 (MainArchNullCert realizes PrimeProjectionCert)**.** *Assume an off-critical mirror pair has a fixed window dominance certificate and assume*

$$\mathsf{MainArchNullCert}(P_\pm, I)$$

for its mirror-odd classifier and fixed interval. Then

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-)$$

holds in the sense of Definition 9312.2.

Proof. The fixed-window dominance certificate supplies a compact test row with nonzero prime-distribution pairing by Definition 9313.4. The main-arch-null certificate refines the choice, through Definition 9313.8, so that the main and displayed correction functionals vanish while the prime-distribution pairing remains nonzero. Applying Definition 9313.11 produces a nonzero finite prime-power residual. This is exactly the compact explicit-formula projection required by Definition 9312.2. □

9313.4 Detection Interface

Theorem 9313.13 (PrimeProjectionCert supplies a prime-generated detector)**.** *Assume*

$$\mathsf{PrimeProjectionCert}(\rho_+, \rho_-)$$

and PrimeGeneratedRecursiveSieve*. Then the finite residual produced by the projection certificate is detected by some local sieve generated by a prime.*

Proof. Unpack Definition 9312.2. It supplies a nonzero finite prime-power residual

$$R_S(g) = \sum_{q \in S} c_q(g)[q].$$

Apply Definition 9312.6. The resulting prime and character give the required local prime-generated detector. □

Principle 9313.14 (Prime projection audit boundary)**.** *The finite arithmetic side of*

$$\mathsf{PrimeProjectionCert}$$

is closed by compact support and character orthogonality. The analytic content is the nonspan row of

$$\mathsf{MainArchNullCert}(P_\pm, I),$$

which asserts that the fixed-window radial readout is not a linear combination of the main, archimedean, trivial-zero, and pole correction rows on the admissible compact test space.

Chapter 9314

Prime Sieve Tower Duality

chapter 9315 treats finite type spaces and sieve readouts as dual coordinates. The present chapter records the generating side of that duality. A prime is not only an object to be classified by a sieve; it is also an atom that creates further sieve surfaces. Divisibility, residue class, finite unit-group, order, and splitting readouts are all generated from prime atoms, and their refinements preserve the BEDC time, unit, and ledger rows.

9314.1 Prime-Generated Sieve Tower

Definition 9314.1 (Euclidean prime sieve tower)**.** *Let $p_1, p_2, p_3, \ldots$ be the increasing sequence of primes and set*

$$P_m := \prod_{j=1}^{m} p_j.$$

The Euclidean prime sieve layer is

$$\mathcal{N}_m := \{n \geq 2 : \gcd(n, P_m) = 1\}.$$

Thus $\mathcal{N}_m$ is the set of integers not removed by the first m prime divisibility tests.

Theorem 9314.2 (Minimal survivor is a new prime)**.** *If $\mathcal{N}_m$ is nonempty and $q = \min \mathcal{N}_m$, then q is a prime distinct from $p_1, \ldots, p_m$.*

Proof. If q were composite, write $q = ab$ with $1 < a < q$ and $1 < b < q$. By minimality of q in $\mathcal{N}_m$, the integer a is not in $\mathcal{N}_m$. Hence some p_j with $j \leq m$ divides a, and therefore divides q. This contradicts $\gcd(q, P_m) = 1$. Thus q is prime. Since q is coprime to P_m, it is not one of $p_1, \ldots, p_m$. □

Corollary 9314.3 (Prime atoms extend the sieve tower)**.** *Adding the prime p_{m+1} refines the survivor layer by*

$$\mathcal{N}_{m+1} = \mathcal{N}_m \cap \{n : p_{m+1} \nmid n\}.$$

Hence each prime is both a survivor of previous sieve layers and a new divisibility test for later layers.

Proof. The identity is the definition of coprimality with $P_{m+1} = P_m p_{m+1}$. The previous theorem identifies the next minimal survivor as a new prime, and the displayed identity identifies its corresponding divisibility test. □

Principle 9314.4 (Prime atoms are sieve generators)**.** *A prime atom has a dual status in the sieve tower. It is an arithmetic object classified by earlier tests, and it generates later tests such as divisibility by p, residue class modulo p, finite unit-group readout in $\mathbb{F}_p^\times$, and order data in that unit group. The same atom therefore lives on both the source side and the method side of the tower.*

9314.2 Local Prime Sieve Surfaces

Definition 9314.5 (Prime-generated local sieve)**.** *For a prime p, the prime-generated local sieve packet contains:*

$$S_p^{\mathrm{div}}(n) := \mathbf{1}_{p|n},$$
$$n \longmapsto n \bmod p, \qquad \ell \longmapsto \ell \bmod p \quad (\ell \neq p),$$
$$\mathbb{F}_p^\times, \qquad a \longmapsto \mathrm{ord}_p(a) \quad (p \nmid a).$$

Each row is a finite classifier generated by the prime p.

Theorem 9314.6 (Fermat unit row)**.** *If p is prime and $p \nmid a$, then*

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof. The nonzero residue classes modulo p form the finite multiplicative group $\mathbb{F}_p^\times$ of order $p-1$. The order of the class of a divides the group order, giving the displayed congruence. □

Corollary 9314.7 (Fermat order sieve)**.** *For fixed a and n, the condition*

$$a^n \equiv 1 \pmod{p}, \qquad p \nmid a,$$

is the same as

$$\mathrm{ord}_p(a) \mid n.$$

It is therefore a finite unit-group sieve on primes p.

Proof. In the finite group $\mathbb{F}_p^\times$, an element satisfies $a^n = 1$ exactly when its order divides n. □

Principle 9314.8 (Local prime sieves preserve unit readout)**.** *The local sieve generated by p carries a unit row: characters have values in S^1, nonzero residues live in $\mathbb{F}_p^\times$, and Fermat order conditions express finite cyclic closure. Passing from divisibility to residue class or character readout changes the visible coordinate system, not the ledgered arithmetic source.*

Definition 9314.9 (Zero fiber and quotient coordinate)**.** *For a prime p, the residue classifier*

$$\pi_p : \mathbb{Z} \to \mathbb{Z}/p\mathbb{Z}$$

has zero fiber

$$\pi_p^{-1}(0) = \{b : p \mid b\}.$$

If $b = ap$, then b lies in this zero fiber and $a = b/p$ is the first quotient coordinate of the fiber. More generally, writing

$$b = p^v u, \qquad p \nmid u,$$

exposes the valuation coordinate $v_p(b) = v$ and the unit residue coordinate

$$u \bmod p \in \mathbb{F}_p^\times.$$

Principle 9314.10 (Divisibility is a fiber readout)**.** *The equation $b = ap$ is not only multiplication in $\mathbb{Z}$. In the local sieve generated by p, it says that b is classified into the zero fiber and that the quotient a is internal fiber data. Thus the same prime-generated sieve records both a visible residue coordinate and the higher valuation data that remains after entering the zero fiber.*

9314.3 Quadratic Local Unit Sieve

Definition 9314.11 (Gaussian norm local sieve)**.** *The Gaussian norm local sieve classifies rational primes by their behavior in $\mathbb{Z}[i]$:*

$$\pi_{\mathbb{Z}[i]}(p) = \begin{cases} \text{ramified}, & p = 2, \\ \text{split}, & p \equiv 1 \pmod 4, \\ \text{inert}, & p \equiv 3 \pmod 4. \end{cases}$$

For $p \equiv 1 \pmod 4$, the split row is represented by

$$p = a^2 + b^2 = N(a + bi) = (a + bi)(a - bi).$$

Theorem 9314.12 (Split Gaussian primes carry a unit-circle quotient)**.** *If a rational prime p has*

$$p = a^2 + b^2$$

with $\pi = a + bi$, then

$$u := \frac{\pi}{\overline{\pi}}$$

has complex modulus 1.

Proof. Since $N(\pi) = \pi\overline{\pi} = p$, one has

$$|u|^2 = \frac{\pi\overline{\pi}}{\overline{\pi}\pi} = 1.$$

□

Theorem 9314.13 (Modular two-square row detects the same split type)**.** *For an odd prime p, the congruence*

$$a^2 + b^2 \equiv 0 \pmod p$$

has a solution with $b \not\equiv 0 \pmod p$ if and only if -1 is a square in $\mathbb{F}_p$, equivalently if and only if

$$p \equiv 1 \pmod 4.$$

Proof. If $b \not\equiv 0 \pmod p$, then the displayed congruence is equivalent to

$$\left(\frac{a}{b}\right)^2 \equiv -1 \pmod p.$$

Thus it holds exactly when -1 has a square root in $\mathbb{F}_p$. For odd primes, Euler's criterion gives

$$(-1)^{(p-1)/2} \equiv 1 \pmod p$$

exactly when $p \equiv 1 \pmod 4$. □

Principle 9314.14 (Two-square data is a local unit sieve)**.** *The integer row $p = a^2 + b^2$ and the modular row $a^2 + b^2 \equiv 0 \pmod p$ are two projections of the same local split phenomenon. The first reads p through the Gaussian norm, while the second reads the existence of $i^2 = -1$ in $\mathbb{F}_p$. Both rows display the prime as carrying a quadratic unit-circle resource rather than as a bare arithmetic atom.*

9314.4 Sieve Objects and Refinement

Definition 9314.15 (Sieve object)**.** *A sieve object is a packet*

$$\mathcal{S} = (X, G, \pi, \mu, E, \Lambda)$$

where X is the source being classified, G is a finite type space, $\pi : X \to G$ is the classifier, μ is a probability measure on G, $E \subseteq G$ is the displayed event or pass region, and Λ is the ledger row recording the source, event, complement, probability mass, and residual data.

Definition 9314.16 (Sieve refinement)**.** *A sieve object*

$$\mathcal{S}_2 = (X, G_2, \pi_2, \mu_2, E_2, \Lambda_2)$$

refines

$$\mathcal{S}_1 = (X, G_1, \pi_1, \mu_1, E_1, \Lambda_1)$$

when there is a map $F : G_2 \to G_1$ such that

$$\pi_1 = F \circ \pi_2,$$

$$\mu_1(a) = \sum_{F(b)=a} \mu_2(b),$$

and

$$\Lambda_2 = \Lambda_1 + \mathrm{RefineLedger}_{2,1}\,.$$

Theorem 9314.17 (Sieve refinement preserves BEDC rows)**.** *Sieve refinement preserves the BEDC time, unit, and ledger rows.*

Proof. The time row is the refinement history $G_2 \to G_1$ together with $\pi_1 = F \circ \pi_2$. The unit row is preserved by probability pushforward:

$$\sum_{a \in G_1} \mu_1(a) = \sum_{a \in G_1} \sum_{F(b)=a} \mu_2(b) = \sum_{b \in G_2} \mu_2(b) = 1.$$

For an event E, the amplitude split

$$\psi = P_E \psi + P_{E^c} \psi$$

is orthogonal by Definition 9301.5. The ledger row is preserved by the displayed equation $\Lambda_2 = \Lambda_1 + \mathrm{RefineLedger}_{2,1}$, so refinement does not discard previous source or residual data. □

9314.5 Composition and Prime Sieve Factors

Definition 9314.18 (Composite and prime sieve factors)**.** *A finite abelian sieve object with type group G is decomposable when there are nontrivial finite abelian groups G_1, G_2 such that*

$$G \simeq G_1 \times G_2, \qquad \pi = (\pi_1, \pi_2), \qquad \mu = \mu_1 \otimes \mu_2.$$

It is a prime sieve factor when no such nontrivial decomposition is part of its displayed ledger.

Theorem 9314.19 (Finite abelian sieves decompose to prime-power factors)**.** *Every finite abelian sieve type group decomposes into finite prime-power type factors.*

Proof. By the finite abelian group structure theorem,

$$G \simeq \prod_j \mathbb{Z}/n_j\mathbb{Z}.$$

Writing each n_j as a product of prime powers and applying the Chinese remainder theorem gives

$$\mathbb{Z}/n_j\mathbb{Z} \simeq \prod_{\ell^a \| n_j} \mathbb{Z}/\ell^a\mathbb{Z}.$$

Combining these decompositions expresses G as a product of prime-power type factors. □

Principle 9314.20 (Composite sieves carry factor ledgers)**.** *A composite sieve is not a single opaque test. Its factorization into prime-power type factors is part of the ledger whenever factor-level detection is used. Thus a residual detected by a composite modulus, Frobenius product, or character product can be localized to displayed factor rows rather than hidden inside an unstructured global sieve.*

9314.6 Duality Energy and Ledger

Theorem 9314.21 (Finite Fourier change of sieve coordinates preserves energy)**.** *Let G be a finite abelian group with uniform probability measure and let $R \in L^2(G)$. With Fourier coefficients*

$$\widehat{R}(\chi) := \frac{1}{|G|} \sum_{a \in G} R(a)\overline{\chi(a)},$$

one has

$$\sum_{a \in G} |R(a)|^2 = |G| \sum_{\chi \in \widehat{G}} |\widehat{R}(\chi)|^2.$$

Proof. After normalization by $|G|^{-1/2}$, the character table is a unitary matrix. The displayed identity is Parseval's identity for this unitary change of coordinates. □

Theorem 9314.22 (Sieve projection splits residual energy)**.** *For a finite probability layer (G, μ) and event $E \subseteq G$, every residual $R \in L^2(G, \mu)$ satisfies*

$$R = P_E R + P_{E^c} R, \qquad P_E R \perp P_{E^c} R,$$

and hence

$$\|R\|^2 = \|P_E R\|^2 + \|P_{E^c} R\|^2.$$

Proof. The supports of $P_E R$ and $P_{E^c} R$ are disjoint, so their inner product is zero. The norm identity follows from orthogonal decomposition in $L^2(G, \mu)$. □

Corollary 9314.23 (Sieve transformations preserve or ledger energy)**.** *A finite Fourier change from type coordinates to sieve coordinates preserves total residual energy. A projection sieve splits the energy into the kept component and the complementary ledger component.*

Proof. Use Definition 9314.21 for a coordinate change and Definition 9314.22 for an event projection. The complementary projection is precisely the residual entry required by Definition 9302.3. □

9314.7 Recursive Prime-Generated Completeness

Definition 9314.24 (RecursivePrimeGeneratedSieveCompleteness)**.** *The row*

$$\mathsf{RecursivePrimeGeneratedSieveCompleteness}$$

asserts that finite prime-generated residue and character sieves detect every nonzero finite prime-power residual. In concrete form, for every nonzero finite residual

$$R = \sum_{j=1}^{N} c_j[q_j]$$

with distinct positive integers q_j, there are a prime p avoiding the support, an event $A \subseteq \mathbb{F}_p^\times$ or a character

$$\chi : \mathbb{F}_p^\times \to S^1,$$

and a nonzero readout of R through the classifier

$$\pi_p(q) = q \bmod p.$$

Theorem 9314.25 (Prime-generated local sieves detect finite residuals)**.** *Every nonzero finite residual*

$$R = \sum_{j=1}^{N} c_j[q_j]$$

with distinct positive integers q_j has a detecting local sieve generated by some prime p.

Proof. Choose a prime p that divides none of the q_j and none of the nonzero differences $q_i - q_j$. Then the residues

$$q_j \bmod p$$

are nonzero and pairwise distinct in $\mathbb{F}_p^\times$. Pick j with $c_j \neq 0$ and let

$$A = \{q_j \bmod p\}.$$

The event readout on A is exactly c_j, hence nonzero. Since $\mathbb{F}_p^\times$ is a finite abelian group, the character readouts also form a complete Fourier basis; equivalently, applying Definition 9315.6 to this finite layer gives a character with nonzero readout whenever a character witness is required. □

Corollary 9314.26 (Recursive generated sieves supply finite residual detection)**.** *The prime-generated local sieve family satisfies*

$$\mathsf{RecursivePrimeGeneratedSieveCompleteness}$$

for finite residuals.

Proof. This is the content of Definition 9314.25, with event and character readouts supplied by the same prime-generated finite unit group. □

Principle 9314.27 (Recursive generated sieves are BEDC-legal)**.** *Recursive use of prime-generated sieves preserves the three BEDC rows. Each step appends a prime-generated classifier to the time history, reads through a finite unit or probability amplitude surface, and records the excluded residue, valuation, or residual component in a ledger. Thus recursive filtering changes visibility but not the accounted information or residual energy.*

9314.8 Tower Interface

Definition 9314.28 (PrimeSieveTowerDuality)**.** *The row*

PrimeSieveTowerDuality

consists of the Euclidean prime-generated tower, the local prime sieve packets, the zero-fiber and valuation rows, the Gaussian local unit sieve, the sieve-object refinement category, the prime-power factor ledger, the finite Fourier/event energy laws, and recursive prime-generated finite residual detection. It asserts that prime-generated sieve transformations preserve the time source, unit or energy readout, and ledger accountability of the prime classifier tower.

Theorem 9314.29 (Prime sieve tower duality supplies duality completeness data)**.** *Assume* PrimeSieveTowerDuality *and that every zeta-facing radial residual enters the generated profinite prime type tower. Then the finite type-sieve dual pairs required by*

PrimeSieveDualityCompleteness

are legal BEDC sieve objects with preserved time, unit, and ledger rows.

Proof. The generated tower supplies the finite prime type spaces and their refinement maps. The local prime sieve packets supply legal event, residue, zero-fiber, valuation, order, Gaussian norm, and character readouts. Sieve refinement preserves the time, unit, and ledger rows by Definition 9314.17. The Fourier and event energy laws show that changing to the sieve-frequency side does not erase residual energy, while projection sends complementary energy to the ledger. Finite residual detection is supplied by Definition 9314.26. These are exactly the legal data needed for the finite dual pairs of Definition 9315.12. □

Principle 9314.30 (Prime sieve tower audit boundary)**.** *The tower duality packet is a structural certificate for sieve methods, not an analytic source theorem for zeta zeros. It says that once a radial residual is represented in the prime-generated type tower, no combination, factorization, Fourier change, or event projection of sieves may erase it without a ledger row. The analytic obligation is therefore isolated to the bridge that places every off-critical zero-side residual into this prime-generated tower.*

Chapter 9315

Prime Sieve Duality Completeness

chapter 9300 supplies finite prime-register quotients and profinite separation, while chapter 9302 supplies finite probability tests on those quotients. The present chapter records the duality principle that makes these two surfaces the same finite detection mechanism: prime types are the position coordinates of a finite layer, and sieve tests are its readout coordinates.

9315.1 Type and Sieve Coordinates

Definition 9315.1 (Prime type and sieve dual pair)**.** *A prime type and sieve dual pair is a finite layer*

$$(G_m, \mu_m, \pi_m, \mathcal{S}_m)$$

where G_m is a finite prime type space, μ_m is a full-support probability measure on G_m, π_m sends prime powers to their displayed type, and $\mathcal{S}_m$ is a finite-dimensional space of legal sieve readouts

$$\varphi : G_m \to \mathbb{C}.$$

When G_m is a finite abelian group, the canonical frequency readouts are the characters

$$\widehat{G}_m := \mathrm{Hom}(G_m, S^1).$$

The position-frequency pairing is

$$\langle a, \chi \rangle := \chi(a), \qquad a \in G_m, \quad \chi \in \widehat{G}_m.$$

Principle 9315.2 (Prime types and sieve methods are dual coordinates)**.** *At a finite classifier layer, the type atoms $\mathbf{1}_a$ and the sieve readouts φ are two coordinate systems for the same finite space of observations. The type side records where the prime register lands. The sieve side records how the layer is read. A residual hidden from all legal sieve readouts is therefore hidden from the finite type space itself; in a complete dual pair this is possible only for the zero residual.*

Definition 9315.3 (Sieve-method type space)**.** *A sieve-method type space at layer m is a finite classifier*

$$H_m$$

whose points classify legal sieve methods by their construction source: congruence, character, Frobenius, splitting, local norm, order, or finite Satake-type readout. A second-order sieve is a legal function

$$\Phi : H_m \to \mathbb{C}.$$

The pair $(H_m, \widehat{H}_m)$ is required to obey the same finite duality and ledger rules as $(G_m, \widehat{G}_m)$ whenever it is used as proof data.

Principle 9315.4 (Recursive duality is ledger-bounded)**.** *Sieve methods may themselves be classified and filtered, but the process is not an unbounded regress. Each recursive duality level must expose a finite or profinite classifier, its legal readout space, its normalization convention, and its ledger row. Thus a method of selecting sieve methods is another licensed finite observation surface, not an informal meta-choice.*

9315.2 Finite Dual Detection

Definition 9315.5 (Fourier-complete sieve layer)**.** *A finite abelian prime type layer G_m is Fourier-complete when its legal sieve readouts contain all characters*

$$\chi \in \widehat{G}_m$$

with the normalization

$$\widehat{R}(\chi) := \sum_{a \in G_m} R(a)\overline{\chi(a)}\mu_m(a)$$

for residuals $R \in L^2(G_m, \mu_m)$.

Theorem 9315.6 (Fourier-complete finite layers detect nonzero residuals)**.** *Let G_m be a Fourier-complete finite abelian prime type layer with uniform probability measure. If*

$$R \in L^2(G_m, \mu_m), \qquad R \neq 0,$$

then some legal character readout detects R:

$$\widehat{R}(\chi) \neq 0$$

for some $\chi \in \widehat{G}_m$.

Proof. For a finite abelian group with uniform measure, the characters form an orthogonal basis of $L^2(G_m, \mu_m)$. If every coefficient $\widehat{R}(\chi)$ vanished, the Fourier expansion of R in that basis would have all coefficients equal to zero, hence $R = 0$. This contradicts the hypothesis. □

Theorem 9315.7 (Event-complete finite layers detect nonzero residuals)**.** *Let (G_m, μ_m) be a finite probability layer with full support. If*

$$R \in L^2(G_m, \mu_m), \qquad R \neq 0,$$

then some event $E \subseteq G_m$ has nonzero event readout:

$$\sum_{a \in E} R(a)\mu_m(a) \neq 0.$$

Proof. If every event readout vanished, then taking $E = \{a\}$ would give

$$R(a)\mu_m(a) = 0$$

for every $a \in G_m$. Full support gives $\mu_m(a) > 0$, so $R(a) = 0$ for every a, contradicting $R \neq 0$. □

Corollary 9315.8 (Finite type-sieve duality detects residuals)**.** *In a finite complete type-sieve dual pair, every nonzero layer residual is detected by either an event readout or a character readout. If the layer is Fourier-complete, the character readout is enough.*

Proof. Apply Definition 9315.7 to the event side. In the abelian Fourier-complete case, apply Definition 9315.6 to the character side. □

9315.3 Dual Tower Direction

Definition 9315.9 (Dual classifier tower)**.** *A dual classifier tower consists of a profinite type system*

$$G_\infty = \varprojlim_m G_m$$

together with a directed sieve-readout system

$$\mathcal{S}_0 \to \mathcal{S}_1 \to \mathcal{S}_2 \to \cdots$$

such that every readout in $\mathcal{S}_m$ pulls back to a legal readout on G_{m+1}. In the finite abelian case, the direction reversal is

$$G_0 \leftarrow G_1 \leftarrow G_2 \leftarrow \cdots, \qquad \widehat{G}_0 \to \widehat{G}_1 \to \widehat{G}_2 \to \cdots.$$

Principle 9315.10 (Type refinement is inverse, sieve extension is direct)**.** *Refining prime types sends a finite type space to a finer quotient above it. Extending sieve methods adds new legal readouts that factor through the finer type space. Thus the type surface is naturally inverse, while the sieve surface is naturally direct. Completeness of the RH route requires these two movements to be compatible: no nonzero residual may disappear under every finite type projection and every legal sieve extension.*

Theorem 9315.11 (Profinite type residuals descend to dual finite layers)**.** *Let a dual classifier tower have Hausdorff profinite type system $G_\infty = \varprojlim_m G_m$. If a profinite residual is nonzero in G_∞, then some finite layer has a nonzero residual projection detected by the type-sieve dual pair at that layer.*

Proof. By Definition 9300.8, the residual has a nonzero projection in some finite quotient G_m. The finite dual pair at that layer detects the projection by Definition 9315.8. Hence the residual has a finite event or character readout. □

9315.4 RH Interface

Definition 9315.12 (PrimeSieveDualityCompleteness)**.** *The row*

PrimeSieveDualityCompleteness

asserts that every zeta-facing radial residual enters a Hausdorff profinite type system whose finite layers are complete type-sieve dual pairs. Equivalently, any nonzero radial residual has a finite layer m and a legal readout $\varphi \in \mathcal{S}_m$ such that

$$\langle R_m, \varphi \rangle_m \neq 0.$$

Theorem 9315.13 (Prime sieve duality supplies probability completeness)**.** *Assume* PrimeSieveDualityCompletenes
Then every nonzero radial residual of a zeta-facing PFZ packet is visible on a finite probability-filtered prime curve in the sense required by

LayeredProbabilitySieveCompleteness.

Proof. By Definition 9315.12, the residual has a finite layer and a legal readout with nonzero pairing. Event readouts give mean-zero probability tests after subtracting their layer mean. Character readouts are legal probability tests on the same finite classifier. The corresponding curve is the probability-filtered Chebyshev curve of Definition 9302.5. Thus the residual is visible in the sense of Definition 9302.8. □

Theorem 9315.14 (Duality-complete sieve excludes hidden radial residuals)**.** *Assume* PrimeSieveDualityCompleten *every detected residual readout is recorded in its sieve ledger, and legal closed BEDC/PFZ zero packets cannot carry unledgered residual coordinates. Then no nonzero radial residual can be hidden in a legal closed zero packet.*

Proof. A nonzero radial residual has a finite event or character readout by Definition 9315.12. The readout is a finite residual row. If the row is recorded, the packet carries an append ledger and is not closed. If the row is not recorded, the sieve ledger policy is violated. Both alternatives contradict hidden legal closure. □

Corollary 9315.15 (Duality completeness is the finite witness route)**.** *The row* PrimeSieveDualityCompleteness *supplies the finite witness component of the probability sieve tower route: a radial residual cannot survive solely as a limit phenomenon outside all finite type and sieve coordinates.*

Proof. The profinite part follows from Definition 9315.11. The finite detection part follows from Definition 9315.8. Together they produce the finite witness needed by Definition 9304.10. □

Principle 9315.16 (Duality audit boundary)**.** *The type-sieve duality packet does not prove the analytic source of a radial residual. That source remains the explicit-formula and fixed-window obligation. The packet proves the classifier side: once a nonzero residual is present in the prime type tower, finite event and frequency readouts cannot all vanish. Hence the remaining burden is to connect every off-critical zero to this prime-side residual tower.*

Chapter 9316

Prime Distribution Curve Projection

chapter 9300 records the finite and profinite classifier surfaces that detect prime-register residuals. The distribution-curve projection is the analytic surface read from the same tower: each classifier does not erase primes, but projects the prime register into a unit, orthogonal, splitting, or Frobenius readout, and the explicit formula turns that readout into a normalized prime-distribution curve. The RH-facing obstruction is then visible as a nonunit radial mode on at least one such curve.

9316.1 Classifier Projections

Definition 9316.1 (Finite prime projection classifier)**.** *A finite prime projection classifier is a pair*

$$\mathcal{C}_m = (G_m, \pi_m)$$

where G_m is a finite group or finite abelian quotient and

$$\pi_m(p) \in G_m$$

is the displayed type of a prime p at level m. Typical reads include residue classes, quadratic characters, Dirichlet or Hecke character values, Frobenius conjugacy classes, and finite Satake-parameter approximations. The classifier is a projection of prime-register data, not a deletion of primes from the ledger.

Definition 9316.2 (Recursive prime projection tower)**.** *A recursive prime projection tower is a refinement chain*

$$\mathcal{C}_0 \prec \mathcal{C}_1 \prec \mathcal{C}_2 \prec \cdots$$

whose finite readouts refine the previous prime-register surface. The passage from one level to the next is licensed by a displayed residual, splitting distinction, character row, or Frobenius row in the sense of Definition 9300.3. The tower is cofinal when its finite quotients form the Hausdorff profinite system of Definition 9300.7.

Definition 9316.3 (Filtered Chebyshev curve)**.** *Let $\mathcal{C}_m = (G_m, \pi_m)$ be a finite prime projection classifier and let*

$$\chi : G_m \to S^1$$

be a character or a displayed class-function readout. The filtered Chebyshev curve is

$$\psi_{m,\chi}(x) := \sum_{p^k \le x} \chi(\pi_m(p^k)) \log p.$$

Its critical normalization is

$$\Psi_{m,\chi}(u) := e^{-u/2}\left(\psi_{m,\chi}(e^u) - \delta_{\chi,1}e^u\right), \qquad x = e^u.$$

Principle 9316.4 (Prime filtering is projection)**.** *The family $\Psi_{m,\chi}$ is the continuous projection of the recursive prime classifier tower. At the discrete level the tower reads prime types such as split, inert, ramified, character value, Frobenius class, or local parameter. At the continuous level the same reads become weighted prime-distribution curves. Thus the phrase "sieving primes" denotes a sequence of classifier projections, not a loss of arithmetic mass.*

9316.2 Unit and Radial Modes

Definition 9316.5 (Prime-distribution spectral mode)**.** *A spectral mode of a normalized prime-distribution curve is a term of the form*

$$c\, e^{ru} e^{i\gamma u}.$$

It is a unit phase mode when $r = 0$ and a radial mode when $r \neq 0$.

Theorem 9316.6 (Off-critical zero produces a radial distribution mode)**.** *Assume the explicit formula for a filtered curve $\Psi_{m,\chi}$. If a zero contributing to that filtered explicit formula has*

$$\rho_0 = \frac{1}{2} + r + i\gamma, \qquad r \neq 0,$$

then the normalized curve contains the radial mode

$$c_{\rho_0}\, e^{ru} e^{i\gamma u}$$

for the corresponding nonzero explicit-formula coefficient c_{ρ_0}.

Proof. The zero contribution before normalization has the form

$$c_{\rho_0} e^{\rho_0 u}.$$

Multiplying by $e^{-u/2}$ gives

$$c_{\rho_0} e^{(\rho_0 - \frac{1}{2})u} = c_{\rho_0} e^{ru} e^{i\gamma u}.$$

Since $r \neq 0$, the mode has a positive-real radial factor rather than a unit-circle phase alone. □

Corollary 9316.7 (Fixed-half readout is unit-mode readout)**.** *For a completed zeta zero contributing to a normalized prime-distribution curve, fixed-half readout is equivalent to the absence of radial exponential growth in that zero's spectral mode.*

Proof. If $\rho = 1/2 + i\gamma$, the normalized mode is $e^{i\gamma u}$, a unit phase mode. Conversely, a mode attached to $\rho = 1/2 + r + i\gamma$ has no radial factor exactly when $r = 0$. □

9316.3 Completeness

Definition 9316.8 (PrimeDistributionCurveCompleteness)**.** *The row*

PrimeDistributionCurveCompleteness

asserts that every nonzero radial residual of a zeta-facing PFZ packet has a visible filtered prime-distribution curve. Equivalently, if

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}$$

has $\ell \neq 0$, then some level m and some readout χ of the recursive prime projection tower produce a normalized curve $\Psi_{m,\chi}$ containing a radial mode

$$c\, e^{ru} e^{i\gamma u}, \qquad r \neq 0.$$

Theorem 9316.9 (Finite residual detection supplies a filtered curve)**.** *Every finite prime residual detected by a finite Dirichlet classifier has a corresponding filtered prime-distribution curve with nonzero classifier readout.*

Proof. Let

$$R = \sum_j c_j [q_j]$$

be the finite residual and let χ be the finite Dirichlet classifier with $\chi(R) \neq 0$ supplied by Definition 9300.21. Use the finite classifier $\mathcal{C}_m$ carrying the residue classes modulo the chosen modulus and define $\psi_{m,\chi}$ as in Definition 9316.3. The finite support of R contributes

$$\sum_j c_j \chi(q_j)$$

to that filtered readout, which is nonzero by the choice of χ. □

Theorem 9316.10 (Profinite residual detection supplies a finite filtered curve)**.** *In a cofinal prime projection tower, every non-identity profinite residual has a finite filtered prime-distribution curve that detects it.*

Proof. By Definition 9300.8, a non-identity residual has non-identity image in a finite quotient. By Definition 9300.5, a finite character separates that image from the identity. The corresponding filtered Chebyshev curve of Definition 9316.3 is the continuous prime-distribution readout of the same finite classifier. □

Principle 9316.11 (Distribution curves are the macroscopic PFZ readout)**.** *In a prime-fiber zoom tower, a finite address step appends prime-register data and contributes orthogonal log-energy as in Definition 9296.6. The identity*

$$\log n = \sum_p v_p(n) \log p$$

is the bridge from discrete prime-register coordinates to continuous exponential reads. The curves $\Psi_{m,\chi}(u)$ are therefore the macroscopic continuous projections of the same append-only tower.

9316.4 Conditional Fixed-Half Consequence

Theorem 9316.12 (Distribution-curve completeness gives radial residual visibility)**.** *Assume* PrimeDistributionCurveCompleteness*. If a zeta-facing PFZ packet has $\ell \neq 0$, then some filtered prime-distribution curve contains a radial mode, and the associated finite or profinite classifier row detects a nonunit residual.*

Proof. By Definition 9316.8, the nonzero radial coordinate is visible on some $\Psi_{m,\chi}$ as a mode

$$c\, e^{ru} e^{i\gamma u}, \qquad r \neq 0.$$

The curve is attached to a finite classifier level or to a finite quotient of the cofinal profinite classifier tower. Finite detection is given by Definition 9316.9; profinite detection is given by Definition 9316.10. In both cases the radial mode is a nonunit residual read by a displayed classifier. □

Corollary 9316.13 (Prime distribution curve conditional fixed-half readout)**.** *Assume zeta-facing PFZ zero packets are legal closure objects,* PrimeDistributionCurveCompleteness *holds, and finite radial modes cannot be hidden as legal closure data. Then every admitted nontrivial zero has fixed-half readout.*

Proof. Let ρ be an admitted nontrivial zero and write

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

If $\ell \neq 0$, then Definition 9316.12 produces a filtered prime-distribution curve with a nonunit radial mode. By Definition 9320.18, a classifier-visible nonunit residual cannot be hidden in a legal PFZ closure object. Thus $\ell = 0$. By Definition 9266.2, this is equivalent to $\mathrm{Re}(\rho) = 1/2$. □

Principle 9316.14 (Final distribution-curve target)**.** *The prime-classifier route ends at the family of normalized curves*

$$\{\Psi_{m,\chi}(u)\}_{m,\chi}.$$

The RH-facing condition is that these curves carry only unit phase modes after critical normalization. A radial mode

$$e^{ru} e^{i\gamma u}, \qquad r \neq 0,$$

is the distribution-curve form of an unclosed positive-real scale row. The remaining completeness target is to show that any hidden radial scale in the zero packet appears on at least one filtered prime-distribution curve.

Chapter 9317

Explicit Formula Scale Residual Bridge

chapter 9300 reduces the scale-to-classifier problem to an explicit-formula bridge: a nonzero radial zero-packet coordinate must leave a finite prime-side residual. The present chapter records that bridge as a test-function obligation. It separates the formal part, where finite prime residuals are detected by Dirichlet characters, from the analytic part, where a compactly supported test row must isolate an off-critical zero orbit while accounting for archimedean, trivial-zero, and pole correction rows.

9317.1 Compact Test Rows

Definition 9317.1 (Compact explicit-formula test row)**.** *An admissible compact explicit-formula test row is a function*

$$g \in C_c^\infty(-A, A)$$

for some $A > 0$, together with the Laplace–Fourier readout

$$\widehat{g}(s) := \int_{-A}^{A} g(x) e^{(s-\frac{1}{2})x}\, dx.$$

The row is admissible when $\widehat{g}$ is entire and has the vertical decay required by the zeta explicit formula. Its compact support makes the prime-power side finite:

$$\mathrm{PrimeSide}(g) = \sum_{p^k \le e^A} c_{p,k}(g).$$

Definition 9317.2 (Abstract explicit-formula ledger)**.** *For an admissible compact test row g, the abstract explicit-formula ledger is the identity*

$$\mathcal{Z}(g) = \mathcal{P}(g) + \mathcal{A}(g),$$

where

$$\mathcal{Z}(g) = \sum_{\rho} \widehat{g}(\rho)$$

is the completed nontrivial-zero ledger,

$$\mathcal{P}(g) = \sum_{p,k} c_{p,k}(g)$$

is the prime-power ledger, and $\mathcal{A}(g)$ contains the archimedean, trivial-zero, and pole correction rows. For compact support the sum defining $\mathcal{P}(g)$ is the finite sum over $p^k \le e^A$.

Lemma 9317.3 (Finite exponential moment interpolation)**.** *Let $\lambda_1, \dots, \lambda_N$ be pairwise distinct complex numbers, and let $a_1, \dots, a_N \in \mathsf{Complex}^{\uparrow}$. For every $A > 0$ there exists*

$$g \in C_c^\infty(-A, A)$$

such that

$$\int_{-A}^{A} g(x) e^{\lambda_j x}\, dx = a_j \qquad (1 \le j \le N).$$

Proof. Consider the linear map

$$T : C_c^\infty(-A, A) \to \mathsf{Complex}^{\uparrow N}$$

given by

$$T(g) = \left(\int g(x) e^{\lambda_1 x}\, dx, \dots, \int g(x) e^{\lambda_N x}\, dx \right).$$

If T were not onto, some nonzero vector $(c_1, \dots, c_N)$ would annihilate its image. Then

$$\int g(x) \left(\sum_{j=1}^{N} c_j e^{\lambda_j x} \right) dx = 0$$

for all $g \in C_c^\infty(-A, A)$. Test-function separation gives

$$\sum_{j=1}^{N} c_j e^{\lambda_j x} = 0$$

on $(-A, A)$. Distinct exponentials are linearly independent on an interval, so every c_j is zero, a contradiction. Thus T is onto. □

Corollary 9317.4 (Finite zero readout interpolation)**.** *Let $F = \{\rho_1, \dots, \rho_N\}$ be a finite set of distinct zero positions and let $a_1, \dots, a_N \in \mathsf{Complex}^{\uparrow}$. For every $A > 0$ there is $g \in C_c^\infty(-A, A)$ such that*

$$\widehat{g}(\rho_j) = a_j \qquad (1 \le j \le N).$$

Proof. Apply Definition 9317.3 to

$$\lambda_j = \rho_j - \frac{1}{2}.$$

□

9317.2 Zero-Orbit Isolation

Definition 9317.5 (Off-critical zero orbit)**.** *For an off-critical zero*

$$\rho_0 = \frac{1}{2} + r + i\gamma, \qquad r \ne 0,$$

the symmetry orbit is

$$\mathcal{O}(\rho_0) = \{\rho_0, \overline{\rho_0}, 1 - \rho_0, 1 - \overline{\rho_0}\},$$

read with the multiplicity convention of the completed zeta zero ledger.

Lemma 9317.6 (Finite orbit isolation)**.** *Let F be a finite zero-position set containing $\mathcal{O}(\rho_0)$. For every $A > 0$ there is $g \in C_c^\infty(-A, A)$ such that*

$$\widehat{g}(\rho) = 1 \qquad (\rho \in \mathcal{O}(\rho_0))$$

and

$$\widehat{g}(\rho) = 0 \qquad (\rho \in F \setminus \mathcal{O}(\rho_0)).$$

Proof. Use Definition 9317.4 with the displayed values on the finite set F. □

Definition 9317.7 (ArchNullZeroIsolation)**.** *The obligation*

$$\mathsf{ArchNullZeroIsolation}$$

asserts that for every off-critical zero orbit $\mathcal{O}(\rho_0)$ there exists an admissible compact test row g such that

$$\mathcal{A}(g) = 0$$

and

$$\mathcal{Z}(g) \neq 0.$$

Equivalently, the test row isolates a nonzero zero-side contribution while the archimedean, trivial-zero, and pole correction rows are null.

Principle 9317.8 (Analytic content of arch-null isolation)**.** *The finite interpolation lemmas only control finitely many zero readouts. The analytic content of* ArchNullZeroIsolation *is the stronger claim that the same test row can be chosen with vertical tail control and with $\mathcal{A}(g) = 0$. This is a function-space separation problem: finite zero-orbit evaluation is a point-evaluation functional, while $\mathcal{A}$ is the fixed correction functional from the explicit formula. The obligation asserts that these rows do not collapse into a single one-dimensional correction direction on the admissible compact test space.*

9317.3 Finite Prime Residual

Theorem 9317.9 (ExplicitFormulaScaleResidual from arch-null isolation)**.** *Assume the abstract explicit-formula ledger of Definition 9317.2 and* ArchNullZeroIsolation*. If ρ_0 is off the critical line, then there is a nonzero finite explicit-formula prime residual*

$$R_S(g) = \sum_{q \in S} c_q(g)[q].$$

Proof. By ArchNullZeroIsolation, choose an admissible compact test row g with

$$\mathcal{A}(g) = 0, \qquad \mathcal{Z}(g) \neq 0.$$

The explicit-formula ledger gives

$$\mathcal{Z}(g) = \mathcal{P}(g) + \mathcal{A}(g),$$

hence

$$\mathcal{P}(g) = \mathcal{Z}(g) \neq 0.$$

Since g is supported in $[-A, A]$, the prime side is the finite prime-power sum over $p^k \leq e^A$. Taking S to be that finite support gives the displayed nonzero finite prime residual. □

Theorem 9317.10 (Off-line zero produces a finite prime-classifier defect)**.** *Assume the hypotheses of Definition 9317.9 and the finite residual separation theorem Definition 9300.21. If ρ_0 is an off-critical zero, then there are a finite prime residual $R_S(g)$ and a finite Dirichlet classifier χ such that*

$$\chi(R_S(g)) \neq 0.$$

Proof. Definition 9317.9 produces a nonzero finite prime residual. Applying Definition 9300.21 to that residual gives the displayed finite classifier. □

Corollary 9317.11 (Explicit formula bridge supplies the PFZ classifier input)**.** *Under* ArchNullZeroIsolation *and the explicit-formula ledger, an off-critical zeta zero supplies the finite prime-classifier input needed by Definition 9300.26.*

Proof. The preceding theorem gives a finite prime residual with nonzero Dirichlet classifier readout. This is exactly the finite prime-classifier footprint required by Definition 9300.26. □

9317.4 Signed Prime Energy

Definition 9317.12 (Weil mirror pairing ledger)**.** *For an admissible compact test row g, the zero-side mirror pairing is the quadratic readout*

$$\mathcal{W}_Z(g) := \sum_{\rho} \widehat{g}(\rho)\,\overline{\widehat{g}(1-\overline{\rho})}.$$

The paired explicit-formula ledger writes

$$\mathcal{W}_Z(g) = \mathcal{W}_P(g) + \mathcal{W}_A(g),$$

where $\mathcal{W}_P(g)$ is the prime-side quadratic residual and $\mathcal{W}_A(g)$ is the archimedean, trivial-zero, and pole correction row. This is the signed-energy form of Definition 9317.2.

Definition 9317.13 (WeilNegativeIsolator)**.** *The obligation*

WeilNegativeIsolator

asserts that every off-critical mirror pair of Definition 9317.5 admits an admissible compact test row g with

$$\mathcal{W}_Z(g) < 0.$$

It is the compact-test version of the diagonal negative probe mechanism of Definition 9265.14.

Definition 9317.14 (ArchNullWeilIsolator)**.** *The obligation*

ArchNullWeilIsolator

asserts that the test row in WeilNegativeIsolator *can be chosen with*

$$\mathcal{W}_A(g) = 0.$$

Thus it supplies an admissible compact test row satisfying

$$\mathcal{W}_Z(g) < 0, \qquad \mathcal{W}_A(g) = 0.$$

Theorem 9317.15 (Arch-null negative isolation gives prime-side negative energy)**.** *Assume the paired explicit-formula ledger of Definition 9317.12 and* ArchNullWeilIsolator*. If an off-critical mirror pair exists, then the prime-side quadratic residual has a finite negative readout:*

$$\mathcal{W}_P(g) < 0$$

for some admissible compact test row g.

Proof. By ArchNullWeilIsolator choose g such that

$$\mathcal{W}_Z(g) < 0, \qquad \mathcal{W}_A(g) = 0.$$

The paired explicit-formula ledger gives

$$\mathcal{W}_Z(g) = \mathcal{W}_P(g) + \mathcal{W}_A(g).$$

Hence

$$\mathcal{W}_P(g) = \mathcal{W}_Z(g) < 0.$$

Since g has compact support, this prime-side quadratic residual is supported on finitely many prime powers. □

Definition 9317.16 (Finite prime negative residual)**.** *A finite prime negative residual is a finite prime-power support S together with a quadratic prime-side readout*

$$\mathcal{W}_{P,S}(g) = \sum_{q \in S} b_q(g)[q]$$

such that

$$\mathcal{W}_{P,S}(g) < 0.$$

It is a signed residual rather than merely a nonzero linear residual: the sign records a negative direction for the explicit-formula energy.

Corollary 9317.17 (Off-line zero gives finite prime negative residual)**.** *Under the hypotheses of Definition 9317.15, an off-critical zero gives a finite prime negative residual.*

Proof. The preceding theorem gives $\mathcal{W}_P(g) < 0$. Compact support of g makes the prime side a finite prime-power sum, so the finite support of that sum is the required finite prime negative residual. □

Theorem 9317.18 (Prime negative residual gives classifier obstruction)**.** *Every finite prime negative residual supplies a nonzero finite prime-classifier residual.*

Proof. A negative finite quadratic readout is nonzero as a finite residual. Applying Definition 9300.21 to its finite prime-power support gives a finite Dirichlet classifier with nonzero readout. The sign is additional information: it records that the residual is a negative energy direction, not merely a nonzero prime-side row. □

Principle 9317.19 (Signed residuals cannot be hidden by PFZ smoothing)**.** *A finite prime negative residual is a finite obstruction in the sense of Definition 9297.8. If it is ledgered, it remains visible as scale or energy data and cannot be a gap-free unit-phase closure. If it is unledgered, the PFZ NameCert surface of Definition 9298.8 is incomplete. The scale-obstruction bridge of chapter 9297 therefore treats such a residual as a persistent obstruction rather than as a smooth zoom artefact.*

Corollary 9317.20 (Signed explicit-formula bridge strengthens scale faithfulness)**.** *Under* ArchNullWeilIsolator *and the paired explicit-formula ledger, an off-critical zero supplies the PFZ classifier input of Definition 9317.11 with a signed negative-energy witness.*

Proof. Definition 9317.17 gives the finite signed residual, and Definition 9317.18 gives the finite classifier readout. The final sign information is the negative-energy witness consumed by Definition 9317.19. □

Chapter 9318

Fixed Window Dominance Certificate

chapter 9316 records the macroscopic statement that an off-critical zero contributes a radial mode to a normalized prime-distribution curve. The present chapter turns that statement into a local certificate shape. The certificate does not assert the Riemann hypothesis. It names the finite spectral window, the differential classifier, the bounded observation interval, and the tail inequality that would make the radial mode visible as a BEDC-auditable row.

9318.1 Finite Spectral Windows

Definition 9318.1 (Finite spectral window)**.** *A finite spectral window for a normalized prime-distribution curve is a finite set*

$$\Lambda = \{\lambda_1, \dots, \lambda_N\} \subset \mathbb{C}$$

of pairwise distinct exponents, together with a truncated spectral curve

$$\Psi_\Lambda(u) := \sum_{\lambda \in \Lambda} c_\lambda e^{\lambda u}.$$

For a zeta zero ρ, the corresponding exponent is

$$\lambda = \rho - \frac{1}{2}.$$

Theorem 9318.2 (Finite exponential modes admit exact classifiers)**.** *Let $\Lambda = \{\lambda_1, \dots, \lambda_N\}$ be a finite spectral window and let $a_1, \dots, a_N \in \mathbb{C}$. There is a polynomial $P \in \mathbb{C}[z]$ such that*

$$P(\lambda_j) = a_j \qquad (1 \le j \le N).$$

Consequently, for

$$D = \frac{d}{du},$$

one has

$$P(D)e^{\lambda_j u} = a_j e^{\lambda_j u}.$$

Proof. Use the Lagrange polynomial

$$P(z) := \sum_{j=1}^{N} a_j \prod_{k \ne j} \frac{z - \lambda_k}{\lambda_j - \lambda_k}.$$

Then $P(\lambda_j) = a_j$ for every j. Since

$$De^{\lambda_j u} = \lambda_j e^{\lambda_j u},$$

polynomial functional calculus gives the displayed identity for $P(D)$. □

Definition 9318.3 (Off-critical mirror exponent pair)**.** *An off-critical mirror exponent pair is a pair*

$$\lambda_+ = r + i\gamma, \qquad \lambda_- = -r + i\gamma, \qquad r \neq 0,$$

arising from the mirror zero pair

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma.$$

Definition 9318.4 (Mirror-odd differential classifier)**.** *Let Λ contain an off-critical mirror exponent pair λ_+, λ_-. A mirror-odd differential classifier for that pair in Λ is a polynomial $P_\pm \in \mathbb{C}[z]$ satisfying*

$$P_\pm(\lambda_+) = 1, \qquad P_\pm(\lambda_-) = -1,$$

and

$$P_\pm(\lambda) = 0 \qquad (\lambda \in \Lambda \setminus \{\lambda_+, \lambda_-\}).$$

Theorem 9318.5 (Mirror-odd classifier extracts the radial pair)**.** *Let*

$$\Psi_\Lambda(u) = c_+ e^{\lambda_+ u} + c_- e^{\lambda_- u} + \sum_{\lambda \in \Lambda^\circ} c_\lambda e^{\lambda u},$$

where

$$\Lambda^\circ = \Lambda \setminus \{\lambda_+, \lambda_-\}.$$

If $P_\pm$ is the mirror-odd differential classifier of Definition 9318.4, then

$$P_\pm(D)\Psi_\Lambda(u) = c_+ e^{(r+i\gamma)u} - c_- e^{(-r+i\gamma)u}.$$

Proof. Apply Definition 9318.2 to each exponential term. The classifier has values 1 and -1 on the mirror pair and value 0 on every other exponent in the finite window. □

Lemma 9318.6 (A radial mirror pair is not a unit-phase packet)**.** *Let $r \neq 0$. If*

$$F(u) = c_+ e^{(r+i\gamma)u} - c_- e^{(-r+i\gamma)u}$$

with $(c_+, c_-) \neq (0, 0)$, then F is not a finite sum of unit-phase modes

$$\sum_{j=1}^{M} b_j e^{i\eta_j u}.$$

Proof. Distinct exponentials $e^{\lambda u}$ are linearly independent on every nonempty interval. The exponents $r + i\gamma$ and $-r + i\gamma$ have nonzero real parts, while every unit-phase exponent $i\eta_j$ has real part 0. Equality with a finite unit-phase sum would force the coefficients of the nonunit exponents to vanish, contrary to the hypothesis. □

9318.2 Dominance Certificates

Definition 9318.7 (Fixed window dominance certificate)**.** *Let*

$$\Psi(u) = \Psi_\Lambda(u) + E_\Lambda(u)$$

be the split of a normalized prime-distribution curve into a finite spectral window and its complement. A fixed window dominance certificate for an off-critical mirror exponent pair is a tuple

$$\mathsf{DomCert}(\Lambda, P_\pm, I, \delta)$$

such that $I \subset \mathbb{R}$ is a compact interval, $\delta > 0$, $P_\pm$ is the mirror-odd differential classifier for the pair, and

$$\inf_{u\in I} |P_\pm(D)\Psi_\Lambda(u)| \geq \delta,$$

while

$$\sup_{u\in I} |P_\pm(D)E_\Lambda(u)| < \frac{\delta}{2}.$$

Theorem 9318.8 (Dominance certificate gives an audited radial readout)**.** *Assume* $\mathsf{DomCert}(\Lambda, P_\pm, I, \delta)$ *for an off-critical mirror exponent pair. Then the full curve has a nonzero classifier readout on the fixed interval:*

$$\inf_{u\in I} |P_\pm(D)\Psi(u)| > 0.$$

Moreover the certified main component is a nonunit radial row, and the tail inequality records that this component is not cancelled on the displayed interval.

Proof. For $u \in I$,

$$P_\pm(D)\Psi(u) = P_\pm(D)\Psi_\Lambda(u) + P_\pm(D)E_\Lambda(u).$$

The triangle inequality and the two certificate bounds give

$$|P_\pm(D)\Psi(u)| \geq \delta - \frac{\delta}{2} = \frac{\delta}{2}.$$

By Definition 9318.5, the finite-window main readout is the mirror radial pair. By Definition 9318.6, that finite-window component is not a finite unit-phase packet. The strict tail inequality shows that this component is not cancelled on the displayed interval. □

Principle 9318.9 (Dominance is the fixed-window audit row)**.** *The certificate of Definition 9318.7 is the BEDC form of a dominant-branch argument. The finite window carries the selected branch pair, the polynomial classifier removes the other displayed branches, the interval I is the observation window, and the tail inequality records that undisplayed modes cannot cancel the selected radial readout. The object is local and finite: it does not by itself claim a global zero theorem.*

9318.3 Prime-Side Realization

Definition 9318.10 (Prime-effective dominance realization)**.** *A prime-effective realization of a fixed window dominance certificate is an admissible compact explicit-formula test row h whose zero-side readout is the scalar audit of the dominated interval and whose archimedean, trivial-zero,*

and pole correction row is displayed in the ledger of Definition 9317.2. It is prime-effective when the correction-subtracted prime side is nonzero:

$$\mathcal{P}(h) = \mathcal{Z}(h) - \mathcal{A}(h) \neq 0.$$

For compact support in $[-A, A]$, this prime side has finite support:

$$\mathcal{P}(h) = \sum_{p^k \leq e^A} c_{p,k}(h)[p^k].$$

Theorem 9318.11 (Prime-effective dominance gives a finite prime residual)**.** *If an off-critical mirror exponent pair admits a fixed window dominance certificate with a prime-effective realization, then there is a nonzero finite prime-power residual*

$$R_S(h) = \sum_{q \in S} c_q(h)[q].$$

Proof. By Definition 9318.10, the correction-subtracted prime side is nonzero. Compact support of h makes the explicit-formula prime ledger a finite sum over $p^k \leq e^A$. Taking S to be that finite support gives the displayed nonzero finite residual. □

Theorem 9318.12 (Fixed-window dominance supplies filtered prime detection)**.** *Assume an off-critical mirror exponent pair admits a fixed window dominance certificate with a prime-effective realization. Then there is a finite Dirichlet classifier whose filtered prime-distribution curve detects the corresponding nonunit residual.*

Proof. Definition 9318.11 supplies a nonzero finite prime-power residual. The finite Dirichlet separation theorem Definition 9300.21 supplies a finite character with nonzero readout on that residual. Then Definition 9316.9 turns the finite classifier readout into the corresponding filtered prime-distribution curve. □

Definition 9318.13 (SpectralPacketCert)**.** *Let*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma, \qquad r \neq 0$$

be an off-critical mirror zero pair, and set

$$\lambda_\pm = \rho_\pm - \frac{1}{2}.$$

The certificate

$$\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$$

asserts that there exist a finite spectral window Λ, a mirror-odd differential classifier $P_\pm$, a compact interval $I \subset \mathbb{R}$, a margin $\delta > 0$, and an admissible compact explicit-formula test row h such that $\lambda_+, \lambda_- \in \Lambda$,

$$\mathsf{DomCert}(\Lambda, P_\pm, I, \delta)$$

holds, and h is a prime-effective realization of that fixed window dominance certificate.

Theorem 9318.14 (Spectral packet certificate gives filtered prime detection)**.** *If* $\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$ *holds for an off-critical mirror zero pair, then some finite Dirichlet classifier detects the corresponding nonunit residual on a filtered prime-distribution curve.*

Proof. Unpack Definition 9318.13. It supplies a fixed window dominance certificate together with a prime-effective realization. Apply Definition 9318.12. □

9318.4 Zeta Target

Definition 9318.15 (UniversalSpectralPacketCert)**.** *The row*

$$\mathsf{UniversalSpectralPacketCert}$$

asserts that every off-critical mirror pair in the completed zeta zero ledger admits its certificate

$$\mathsf{SpectralPacketCert}(\rho_+, \rho_-).$$

Definition 9318.16 (ZetaFixedWindowDominance)**.** *The row*

$$\mathsf{ZetaFixedWindowDominance}$$

is the zeta-facing interface exported by

$$\mathsf{UniversalSpectralPacketCert}.$$

Unpacked at a single mirror pair, it consists of the finite spectral window, mirror-odd classifier, fixed observation interval, positive dominance margin, tail inequality, and prime-effective explicit-formula row carried by $\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$.

Theorem 9318.17 (Universal spectral packets supply fixed-window dominance)**.** *If* UniversalSpectralPacketCert *holds, then* ZetaFixedWindowDominance *holds.*

Proof. This is the interface translation of Definition 9318.15 through Definition 9318.16: each off-critical mirror pair receives the finite packet required by the zeta fixed-window row. □

Theorem 9318.18 (ZetaFixedWindowDominance supplies PrimeDistributionCurveCompleteness)**.** *Assume* ZetaFixedWindowDominance*. Then every off-critical zeta mirror pair has a filtered prime-distribution curve on which its nonunit radial residual is visible. In the terminology of Definition 9316.8, the fixed window certificate supplies the local witness required by*

$$\mathsf{PrimeDistributionCurveCompleteness}.$$

Proof. Apply Definition 9318.16 to the off-critical mirror pair. The unpacked $\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$ gives filtered prime detection by Definition 9318.14. This is precisely the visible filtered curve required by Definition 9316.8. □

Corollary 9318.19 (Fixed-window route to the conditional fixed-half readout)**.** *Assume* ZetaFixedWindowDominan *and the hypotheses of Definition 9316.13. Then every admitted nontrivial zero has fixed-half readout.*

Proof. By Definition 9318.18, the required instance of PrimeDistributionCurveCompleteness is available. Apply Definition 9316.13. □

Corollary 9318.20 (Universal spectral packets give the conditional fixed-half readout)**.** *Assume* UniversalSpectralPacketCert *and the hypotheses of Definition 9316.13. Then every admitted nontrivial zero has fixed-half readout.*

Proof. By Definition 9318.17, ZetaFixedWindowDominance holds. Apply Definition 9318.19. □

Definition 9318.21 (TailDominance)**.** *For an off-critical mirror pair, the row*

$$\mathsf{TailDominance}(\rho_+, \rho_-)$$

asserts that there are a finite spectral window Λ, its mirror-odd classifier $P_\pm$, a compact interval I, and a margin $\delta > 0$ such that

$$\inf_{u \in I} |P_\pm(D)\Psi_\Lambda(u)| \geq \delta$$

and

$$\sup_{u \in I} |P_\pm(D)E_\Lambda(u)| < \frac{\delta}{2}.$$

It is the analytic dominance part of $\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$*, separated from the finite Lagrange projector and from the prime-effective explicit-formula realization.*

Theorem 9318.22 (Tail dominance packages the spectral certificate)**.** *If* $\mathsf{TailDominance}(\rho_+, \rho_-)$ *holds and the displayed dominance row has a prime-effective compact explicit-formula realization, then*

$$\mathsf{SpectralPacketCert}(\rho_+, \rho_-)$$

holds.

Proof. The tail-dominance row supplies the finite spectral window, mirror-odd classifier, interval, margin, main-term lower bound, and tail upper bound. These rows are exactly $\mathsf{DomCert}(\Lambda, P_\pm, I, \delta)$. Adding the prime-effective compact explicit-formula realization supplies the last row required by Definition 9318.13. □

Principle 9318.23 (Audit boundary for the fixed-window route)**.** *The proof obligation is concentrated in*

$$\mathsf{UniversalSpectralPacketCert}.$$

For each off-critical mirror pair one must display the finite spectral window, the mirror-odd polynomial classifier, the compact interval, the positive dominance margin, the tail bound, and the compact explicit-formula row whose prime side is nonzero after the recorded correction rows. The finite-window and Lagrange-projector rows are already local certificate data; the unresolved analytic core is the tail-dominance inequality for the selected zeta spectral packet.

Chapter 9319

Damped Spectral Packet Certificate

chapter 9318 isolates the remaining fixed-window burden as a spectral packet certificate with tail dominance. The damped packet route keeps the same local pair and finite Lagrange projector, but appends an entire damping factor. The resulting filter is exact on the selected off-critical mirror pair, vanishes on the displayed finite local packet, and suppresses remote zero rows by vertical exponential decay rather than by a growing interpolation window.

9319.1 Damped Mirror Filter

Definition 9319.1 (Local mirror spectral packet)**.** *Let*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma, \qquad r \neq 0$$

be an off-critical mirror zero pair, and set

$$\lambda_\pm = \rho_\pm - \frac{1}{2} = \pm r + i\gamma.$$

A local mirror spectral packet is a finite set

$$\Lambda = \{\lambda_+, \lambda_-\} \cup \Lambda^\circ$$

of zero exponents. The finite subset Λ° records the local zero rows to be killed exactly by the packet projector.

Definition 9319.2 (Mirror Lagrange packet projector)**.** *For a local mirror spectral packet Λ, define the Lagrange basis polynomials*

$$L_+(z) := \prod_{\lambda \in \Lambda, \lambda \neq \lambda_+} \frac{z - \lambda}{\lambda_+ - \lambda},$$

and

$$L_-(z) := \prod_{\lambda \in \Lambda, \lambda \neq \lambda_-} \frac{z - \lambda}{\lambda_- - \lambda}.$$

The mirror Lagrange packet projector is

$$P_\pm(z) := L_+(z) - L_-(z).$$

It satisfies

$$P_\pm(\lambda_+) = 1, \qquad P_\pm(\lambda_-) = -1,$$

and

$$P_\pm(\lambda) = 0 \qquad (\lambda \in \Lambda^\circ).$$

Definition 9319.3 (Damped mirror packet filter)**.** *For $\mu > 0$, define*

$$G_\mu(z) := \exp\left(-\mu\left((z - i\gamma)^2 - r^2\right)^2\right),$$

and

$$\Phi_\mu(z) := P_\pm(z) G_\mu(z).$$

The factor G_μ is normalized on the selected mirror pair:

$$G_\mu(\lambda_+) = G_\mu(\lambda_-) = 1.$$

Therefore

$$\Phi_\mu(\lambda_+) = 1, \qquad \Phi_\mu(\lambda_-) = -1,$$

and

$$\Phi_\mu(\lambda) = 0 \qquad (\lambda \in \Lambda^\circ).$$

Theorem 9319.4 (Damped packet filter has vertical decay)**.** *Fix $C > 0$. On the strip*

$$|\operatorname{Re} z| \le C$$

there are constants $c > 0$ and $C_{\mu,P} > 0$ such that, for sufficiently large $|\operatorname{Im} z - \gamma|$,

$$|\Phi_\mu(z)| \le C_{\mu,P}(1 + |z|)^{\deg P_\pm} \exp\left(-c\mu|\operatorname{Im} z - \gamma|^4\right).$$

Proof. Write

$$w = z - i\gamma = a + it, \qquad t = \operatorname{Im} z - \gamma,$$

with a bounded on the fixed strip. Then

$$w^2 - r^2 = (a^2 - t^2 - r^2) + 2iat.$$

The real part of $(w^2 - r^2)^2$ is

$$(a^2 - t^2 - r^2)^2 - (2at)^2.$$

For large $|t|$, the leading term is t^4, while the negative part is quadratic in t with bounded coefficient. Hence

$$\operatorname{Re}\left((w^2 - r^2)^2\right) \ge c|t|^4$$

outside a compact set. Thus

$$|G_\mu(z)| \le \exp(-c\mu|t|^4).$$

Since $P_\pm$ is a fixed polynomial,

$$|P_\pm(z)| \le C_P(1 + |z|)^{\deg P_\pm}.$$

Multiplying these estimates gives the claimed bound. □

9319.2 Negative Zero-Side Packet

Definition 9319.5 (Exponent mirror quadratic readout)**.** *For a spectral filter Φ on the zero-exponent ledger, define*

$$\mathcal{W}_Z^\lambda(\Phi) := \sum_{\lambda_\rho} \Phi(\lambda_\rho)\overline{\Phi(-\overline{\lambda_\rho})},$$

where $\lambda_\rho = \rho - 1/2$ and the completed zeta multiplicity convention is used.

Definition 9319.6 (DampedSpectralPacketCert)**.** *For an off-critical mirror zero pair ρ_+, ρ_-, the certificate*

$$\mathsf{DampedSpectralPacketCert}(\rho_+, \rho_-)$$

asserts that there are a local mirror spectral packet Λ and a parameter $\mu > 0$ such that the damped mirror packet filter Φ_μ of Definition 9319.3 is admissible for the zero-side quadratic readout and satisfies

$$\mathcal{W}_Z^\lambda(\Phi_\mu) < 0.$$

Theorem 9319.7 (Damped packet gives a negative zero-side readout)**.** *Assume the zero-exponent ledger satisfies the standard zero-counting bound*

$$N(T) = O(T \log T)$$

in vertical strips and that the local packet Λ contains every zero exponent in a fixed bounded height window centered at λ_+, λ_- except the selected pair. Then, for sufficiently large μ,

$$\mathcal{W}_Z^\lambda(\Phi_\mu) < 0.$$

Proof. The selected mirror pair contributes

$$\Phi_\mu(\lambda_+)\overline{\Phi_\mu(\lambda_-)} + \Phi_\mu(\lambda_-)\overline{\Phi_\mu(\lambda_+)} = -2.$$

Every other zero exponent in the displayed local packet is killed by the Lagrange projector. The remaining contribution is the remote tail

$$\mathrm{Tail}_\mu := \sum_{\lambda_\rho \notin \Lambda} \Phi_\mu(\lambda_\rho)\overline{\Phi_\mu(-\overline{\lambda_\rho})}.$$

By Definition 9319.4, the summands have quartic exponential decay in vertical height, up to a fixed polynomial factor. The zero-counting bound makes the tail absolutely convergent and gives

$$\mathrm{Tail}_\mu \to 0$$

as $\mu \to \infty$ by dominated convergence on the dyadic zero ledger. Choose μ so that

$$|\,\mathrm{Tail}_\mu\,| < 1.$$

Then

$$\mathcal{W}_Z^\lambda(\Phi_\mu) = -2 + \mathrm{Tail}_\mu < 0.$$

□

Corollary 9319.8 (Off-critical mirrors supply damped spectral packets)**.** *Under the hypotheses of Definition 9319.7, every off-critical mirror zero pair admits*

$$\mathsf{DampedSpectralPacketCert}(\rho_+, \rho_-).$$

Proof. Use the local packet and the parameter μ supplied by Definition 9319.7. The resulting filter is admissible for the zero-side quadratic readout and has negative value, which is precisely Definition 9319.6. □

9319.3 Prime Residual Interface

Definition 9319.9 (AdmissibleDampedPacketApproximation)**.** *The row*

$$\mathsf{AdmissibleDampedPacketApproximation}$$

asserts that the damped filters Φ_μ can be represented by, or approximated by, admissible Weil explicit-formula test rows while preserving the strict negative zero-side inequality

$$\mathcal{W}_Z^\lambda(\Phi_\mu) < 0.$$

Definition 9319.10 (Damped arch non-cancellation)**.** *The row*

$$\mathsf{DampedArchNonCancellation}$$

asserts that, for at least one admissible damped packet associated to an off-critical mirror pair, the archimedean, trivial-zero, and pole correction terms do not cancel the negative zero-side readout. In the paired explicit-formula notation of Definition 9317.12, the resulting full prime side is nonzero.

Theorem 9319.11 (Damped packet gives a finite prime residual)**.** *Assume*

$$\mathsf{DampedSpectralPacketCert}(\rho_+, \rho_-), \qquad \mathsf{AdmissibleDampedPacketApproximation}, \qquad \mathsf{DampedArchNonCancellation}.$$

Then there is a nonzero finite prime-power residual

$$R_S = \sum_{q \in S} a_q[q].$$

Proof. The damped packet supplies a negative zero-side readout. The admissibility row transfers it to the paired explicit-formula test space, and the arch non-cancellation row makes the corresponding prime-side readout nonzero. If the prime-side expansion is finite, take its support. If it is an absolutely convergent weighted prime-power series, choose a finite subset whose partial sum is nonzero; otherwise all finite partial sums would force the total readout to vanish. This gives the displayed finite residual. □

Theorem 9319.12 (Damped packet supplies filtered prime detection)**.** *Under the hypotheses of Definition 9319.11, some finite Dirichlet classifier detects the damped packet residual on a filtered prime-distribution curve.*

Proof. Definition 9319.11 supplies a nonzero finite prime-power residual. Apply the finite Dirichlet separation theorem Definition 9300.21 and then the filtered-curve realization of Definition 9316.9. □

Theorem 9319.13 (Damped packet route gives conditional fixed-half readout)**.** *Assume every off-critical mirror pair admits a damped spectral packet, the admissible approximation and arch non-cancellation rows hold for those packets, zeta-facing PFZ zero packets are legal closure objects, and classifier-visible nonunit residuals cannot be hidden in legal PFZ closure data. Then every admitted nontrivial zero has fixed-half readout.*

Proof. If an admitted zero were off the critical line, it would determine an off-critical mirror pair. By the damped packet hypothesis and Definition 9319.12, some filtered prime-distribution curve would detect a nonunit residual. The no-hidden-residual hypothesis excludes such a residual from a legal PFZ closure object. Therefore the radial coordinate is zero, and Definition 9266.2 gives fixed-half readout. □

Principle 9319.14 (Damped packet audit boundary)**.** *The damped packet route uses a local exact projector plus a vertical damping estimate. The spectral side supplies a negative mirror direction once zero-counting controls the remote tail. The remaining analytic interfaces are admissibility for the Weil explicit formula and non-cancellation by the archimedean, trivial-zero, and pole correction ledger.*

Chapter 9320

Negative Weil Isolator

chapter 9317 records a signed explicit-formula bridge under the arch-null isolator. The negative-Weil package uses a sharper separation surface: an off-critical mirror pair is required to produce a negative direction for the full Weil form, not merely a nonzero prime-side residual. The remaining analytic content is thereby localized in two obligations: Paley–Wiener tail control for the selected zero pair, and independence of the fixed archimedean, trivial-zero, and pole correction rows from that negative mirror direction.

9320.1 Mirror Pair Geometry

Definition 9320.1 (Weil mirror involution)**.** *The Weil mirror involution on the completed zeta zero ledger is*

$$\iota(\rho) := 1 - \overline{\rho}.$$

An off-critical mirror pair is a pair

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \iota(\rho_+) = \frac{1}{2} - r + i\gamma,$$

with $r \neq 0$.

Lemma 9320.2 (Li radius reverses across the mirror)**.** *Let*

$$z(\rho) := 1 - \frac{1}{\rho}.$$

If $z(\rho_+) = e^{\ell + i\theta}$ *for an off-critical mirror pair, then*

$$z(\rho_-) = e^{-\ell + i\theta}.$$

In particular, $r \neq 0$ *is equivalent to* $\ell \neq 0$.

Proof. For $\rho_- = \iota(\rho_+) = 1 - \overline{\rho_+}$,

$$z(\rho_-) = 1 - \frac{1}{1 - \overline{\rho_+}} = \frac{\overline{\rho_+}}{\overline{\rho_+} - 1} = \frac{1}{\overline{z(\rho_+)}}.$$

Thus $z(\rho_+) = e^{\ell + i\theta}$ gives

$$z(\rho_-) = e^{-\ell + i\theta}.$$

The equivalence between $\ell = 0$ and the fixed-half condition follows from Definition 9266.2; hence $r \neq 0$ is equivalent to $\ell \neq 0$. □

Principle 9320.3 (Mirror scale is quadratic residual data)**.** *The linear radial coordinates of an off-critical mirror pair are ℓ and $-\ell$. Their signed sum can disappear from a first-order scale ledger, but their quadratic energy cannot:*

$$\ell^2 + (-\ell)^2 = 2\ell^2 > 0.$$

The mirror pair is therefore not a harmless cancellation. It is a displayed scale-energy row, and any zeta-facing PFZ packet that admits the pair must either ledger this row or violate the no-hidden-radial surface of Definition 9297.14.

9320.2 Zero-Side Negative Direction

Definition 9320.4 (Mirror quadratic zero readout)**.** *For an admissible compact test row g, the mirror quadratic zero readout is*

$$\mathcal{W}_Z(g) := \sum_{\rho} \widehat{g}(\rho)\,\overline{\widehat{g}(\iota(\rho))}.$$

This is the zero-side row of the paired explicit-formula ledger of Definition 9317.12.

Definition 9320.5 (Finite mirror negative interpolation)**.** *Let F be a finite zero-position set containing an off-critical mirror pair ρ_+, ρ_-. A finite mirror negative interpolation row for F is an admissible compact test row g satisfying*

$$\widehat{g}(\rho_+) = 1, \qquad \widehat{g}(\rho_-) = -1,$$

and

$$\widehat{g}(\rho) = 0 \qquad (\rho \in F \setminus \{\rho_+, \rho_-\}).$$

Lemma 9320.6 (Finite mirror negative interpolation exists)**.** *For every finite zero-position set F containing an off-critical mirror pair and every $A > 0$, a finite mirror negative interpolation row exists in $C_c^\infty(-A, A)$.*

Proof. Apply Definition 9317.4 to the finite set F, assigning the values 1 and -1 to the mirror pair and 0 to all other entries of F. □

Lemma 9320.7 (Mirror interpolation gives local negative contribution)**.** *For a finite mirror negative interpolation row, the contribution of the selected mirror pair to $\mathcal{W}_Z(g)$ is*

$$-2.$$

Proof. The two ordered mirror terms are

$$\widehat{g}(\rho_+)\overline{\widehat{g}(\rho_-)} + \widehat{g}(\rho_-)\overline{\widehat{g}(\rho_+)} = 1 \cdot (-1) + (-1) \cdot 1 = -2.$$

□

Definition 9320.8 (PWZeroPairTailControl)**.** *The obligation*

$$\mathsf{PWZeroPairTailControl}$$

asserts that for every off-critical mirror pair ρ_+, ρ_- there is an admissible compact test row g with

$$\widehat{g}(\rho_+) = 1, \qquad \widehat{g}(\rho_-) = -1,$$

such that the remaining zero-side mirror contribution satisfies

$$\left| \sum_{\rho \notin \{\rho_+, \rho_-\}} \widehat{g}(\rho)\, \overline{\widehat{g}(\iota(\rho))} \right| < \frac{1}{2}.$$

Principle 9320.9 (Tail-control analytic content)**.** *Definition 9320.6 supplies the finite interpolation part of* PWZeroPairTailControl*. The analytic part is the simultaneous choice of interpolation row and vertical decay. In Paley–Wiener terms, one must retain the two prescribed point values while making the zero tail small against the standard zero-counting ledger. This is a test-function construction problem, not a BEDC ledger convention.*

Theorem 9320.10 (Tail-controlled mirror pair gives negative zero-side readout)**.** *Assume* PWZeroPairTailControl*. Every off-critical mirror pair admits an admissible compact test row g such that*

$$\mathcal{W}_Z(g) < -\frac{3}{2}.$$

Proof. Choose g by Definition 9320.8. The selected mirror pair contributes -2 by Definition 9320.7. The remaining zero-side contribution has absolute value less than $1/2$. Therefore

$$\mathcal{W}_Z(g) < -2 + \frac{1}{2} = -\frac{3}{2}.$$

□

9320.3 Archimedean Independence

Definition 9320.11 (Admissible Weil test Hilbert space)**.** *An admissible Weil test Hilbert space is a completion $\mathcal{H}$ of the compact explicit-formula test rows with the following structure. For every zero row ρ, the evaluation*

$$E_\rho(g) := \widehat{g}(\rho)$$

extends to a continuous linear functional on $\mathcal{H}$. Finite zero readout interpolation rows remain represented in $\mathcal{H}$. The archimedean, trivial-zero, and pole correction row of the paired explicit formula extends to a continuous quadratic form

$$\mathcal{W}_A : \mathcal{H} \to \mathsf{Real}^{\uparrow}.$$

Definition 9320.12 (Mirror affine interpolation plane)**.** *For an off-critical mirror pair, the mirror affine interpolation plane is*

$$\mathcal{I}_{\rho_\pm} := \left\{ g \in \mathcal{H} : E_{\rho_+}(g) = 1,\ E_{\rho_-}(g) = -1 \right\}.$$

Definition 9320.13 (ArchIndependence)**.** *The obligation*

$$\mathsf{ArchIndependence}$$

asserts that for every off-critical mirror pair, the fixed correction quadratic form does not dominate the negative mirror direction on $\mathcal{I}_{\rho_\pm}$. In the normalized form used here, this means that the test row supplied by PWZeroPairTailControl *may be chosen so that*

$$|\mathcal{W}_A(g)| < \frac{1}{2}.$$

Equivalently, the off-critical point-evaluation direction is not in the closure of the archimedean, trivial-zero, and pole correction directions.

Principle 9320.14 (Arch independence is weaker than arch-null isolation)**.** *The obligation* ArchNullWeilIsolator *of Definition 9317.14 asks for*

$$\mathcal{W}_A(g) = 0.$$

Definition 9320.13 asks only that the fixed correction row cannot force cancellation of the negative mirror direction. It is therefore a separation statement about the admissible test space, rather than a nulling statement for every archimedean correction component.

Theorem 9320.15 (NegativeWeilIsolator)**.** *Assume* PWZeroPairTailControl *and* ArchIndependence*. If an off-critical mirror pair exists, then there is an admissible compact test row g for which the full paired Weil form satisfies*

$$\mathcal{W}(g) < 0.$$

Proof. By PWZeroPairTailControl and ArchIndependence, choose an admissible compact row g satisfying both the tail-controlled mirror interpolation and the correction bound. The tail-controlled estimate is

$$\mathcal{W}_Z(g) < -\frac{3}{2}.$$

The correction estimate is

$$|\mathcal{W}_A(g)| < \frac{1}{2}.$$

For the full paired Weil form

$$\mathcal{W}(g) = \mathcal{W}_Z(g) - \mathcal{W}_A(g),$$

one has

$$\mathcal{W}(g) < -\frac{3}{2} + \frac{1}{2} = -1 < 0.$$

□

9320.4 Prime-Side Negative Residual

Theorem 9320.16 (Negative Weil direction gives finite prime negative residual)**.** *Assume the paired explicit-formula ledger of Definition 9317.12. If an admissible compact test row satisfies*

$$\mathcal{W}(g) < 0,$$

then it supplies a finite prime negative residual in the sense of Definition 9317.16.

Proof. The paired explicit formula identifies the full Weil form with the prime-side quadratic residual after the archimedean, trivial-zero, and pole correction rows have been accounted for:

$$\mathcal{W}(g) = \mathcal{W}_P(g).$$

Thus $\mathcal{W}_P(g) < 0$. If g is supported in $[-A, A]$, the prime-power side is supported on

$$p^k \leq e^A.$$

Hence the negative prime-side readout is a finite prime negative residual. □

Corollary 9320.17 (Negative isolator gives classifier-visible residual)**.** *Assume* PWZeroPairTailControl*,* ArchIndependence*, and the paired explicit-formula ledger. If an off-critical mirror pair exists, then there is a finite Dirichlet classifier with nonzero readout on the associated prime-side residual.*

Proof. Definition 9320.15 supplies g with $\mathcal{W}(g) < 0$. By Definition 9320.16 this gives a finite prime negative residual. A negative residual is nonzero, so Definition 9300.21 supplies a finite Dirichlet classifier with nonzero readout. □

Principle 9320.18 (Negative residuals are PFZ-visible obstructions)**.** *The classifier-visible residual of Definition 9320.17 is a finite obstruction in the sense of Definition 9297.8. If it is entered into the PFZ ledger, the packet is still carrying a negative energy row and is not a unit-phase triangle closure. If it is omitted, the zeta-facing NameCert admission surface is incomplete. If a zoom step attempts to dilute it across scales, the self-substitution and reflection-positive persistence rows of Definition 9297.4 and Definition 9297.7 preserve the finite defect as scale-stable data.*

9320.5 Fixed-Half Consequence

Theorem 9320.19 (Off-critical zero gives unhideable negative residual)**.** *Assume the paired explicit-formula ledger,* PWZeroPairTailControl*,* ArchIndependence*, and the PFZ obstruction bridge of Definition 9320.18. Then every off-critical zero produces a classifier-visible finite prime negative residual that cannot be hidden by legal PFZ zoom, smoothing, or branch selection.*

Proof. An off-critical zero supplies the mirror pair of Definition 9320.1. The two analytic obligations and Definition 9320.15 give an admissible compact test row with negative full Weil form. The paired explicit formula and Definition 9320.16 turn this into a finite prime negative residual. The finite Dirichlet classifier of Definition 9320.17 detects the residual, while Definition 9320.18 prevents the detected finite obstruction from being hidden as zoom noise, solenoid phase, or a smooth branch artefact. □

Corollary 9320.20 (Negative-Weil conditional fixed-half readout)**.** *Assume that every completed zeta zero packet admitted by the zeta-facing PFZ surface is a legal closure object, and assume the hypotheses of Definition 9320.19. Then every admitted nontrivial zero has fixed-half readout.*

Proof. Let ρ be an admitted nontrivial zero and write

$$z(\rho) = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

If $\ell \neq 0$, then Definition 9320.2 places ρ in an off-critical mirror pair. By Definition 9320.19, that pair produces a classifier-visible finite prime negative residual that legal PFZ closure cannot hide. This contradicts the assumption that the admitted zero packet is a legal closure object. Therefore

$$\ell = 0.$$

By Definition 9266.2, this is equivalent to

$$\Re(\rho) = \frac{1}{2}.$$

□

Principle 9320.21 (Remaining analytic frontier)**.** *The negative-Weil route concentrates the remaining analytic frontier in the two obligations*

PWZeroPairTailControl *and* ArchIndependence.

The finite interpolation, the sign calculation, the finite prime residual passage, the Dirichlet classifier detection, and the PFZ visibility consequence are separated from these analytic obligations. Thus the route is not a general appeal to explicit-formula positivity; it is a precise request for a compact test row that isolates an off-critical mirror pair as a negative full Weil direction.

Chapter 9321

Paley–Wiener Zero Pair Tail Control

chapter 9320 isolates the first analytic row in the negative-Weil route as PWZeroPairTailControl. This chapter resolves the finite part of that row and states the precise uniform tail condition needed for the infinite zero ledger. The construction is local at an off-critical mirror pair: a mirror-odd bump gives exact values $(1, -1)$ on the pair, finite exponential interpolation removes every other zero in a bounded window, and a uniform Paley–Wiener tail row then converts the finite negative block into a negative zero-side Weil readout.

9321.1 Mirror-Odd Local Isolator

Definition 9321.1 (Mirror-odd bump)**.** *Let $r \in \mathsf{Real}^{\uparrow}$ with $r \neq 0$. A mirror-odd bump for r is a real-valued function*

$$\phi \in C_c^\infty(-A, A)$$

such that

$$\phi(-x) = -\phi(x)$$

and

$$I_r(\phi) := \int_{\mathbb{R}} \phi(x) \sinh(rx)\, dx \neq 0.$$

Lemma 9321.2 (Mirror-odd bumps exist)**.** *For every $A > 0$ and every $r \neq 0$, there is a mirror-odd bump for r supported in $(-A, A)$.*

Proof. Choose a nonzero nonnegative bump

$$\psi \in C_c^\infty((0, A)).$$

Set

$$\phi(x) := \psi(x) - \psi(-x).$$

Then ϕ is smooth, compactly supported in $(-A, A)$, and odd. The function $\sinh(rx)$ is also odd, so $\phi(x)\sinh(rx)$ is even. On the positive support of ψ, its sign is constant and nonzero after possibly replacing ϕ by $-\phi$. Hence

$$I_r(\phi) = 2\int_0^A \psi(x) \sinh(rx)\, dx \neq 0.$$

□

Definition 9321.3 (Mirror-odd zero-pair row)**.** *For an off-critical mirror pair*

$$\rho_+ = \frac{1}{2} + r + i\gamma, \qquad \rho_- = \frac{1}{2} - r + i\gamma,$$

and a mirror-odd bump ϕ for r, define

$$f_{\rho_+,\phi}(x) := \frac{e^{-i\gamma x}\phi(x)}{I_r(\phi)}.$$

Theorem 9321.4 (Mirror-odd row separates the zero pair)**.** *For the row of Definition 9321.3,*

$$\widehat{f}_{\rho_+,\phi}(\rho_+) = 1, \qquad \widehat{f}_{\rho_+,\phi}(\rho_-) = -1.$$

Proof. At ρ_+ one has

$$\widehat{f}_{\rho_+,\phi}(\rho_+) = \frac{1}{I_r(\phi)} \int_{\mathbb{R}} \phi(x) e^{rx}\, dx.$$

Since

$$e^{rx} = \cosh(rx) + \sinh(rx),$$

the product $\phi(x)\cosh(rx)$ is odd and has integral zero. Therefore

$$\int_{\mathbb{R}} \phi(x) e^{rx}\, dx = \int_{\mathbb{R}} \phi(x)\sinh(rx)\, dx = I_r(\phi),$$

which gives $\widehat{f}_{\rho_+,\phi}(\rho_+) = 1$.

At ρ_- the same cancellation gives

$$\widehat{f}_{\rho_+,\phi}(\rho_-) = \frac{1}{I_r(\phi)} \int_{\mathbb{R}} \phi(x) e^{-rx}\, dx.$$

Using

$$e^{-rx} = \cosh(rx) - \sinh(rx),$$

the odd part again integrates to zero and the remaining term is

$$-I_r(\phi).$$

Hence $\widehat{f}_{\rho_+,\phi}(\rho_-) = -1$. □

Corollary 9321.5 (Mirror-odd row gives a local negative block)**.** *The selected mirror pair contributes*

$$-2$$

to the zero-side mirror quadratic readout of Definition 9320.4.

Proof. By Definition 9321.4, the two ordered mirror terms are

$$1 \cdot \overline{-1} + (-1) \cdot \overline{1} = -2.$$

□

9321.2 Finite Window Cancellation

Definition 9321.6 (Centered nontrivial zero window)**.** *For an off-critical mirror pair at height γ and a width $T > 0$, the centered nontrivial zero window is the finite zero-position set*

$$\mathcal{F}_T(\gamma) := \{\rho : \ \rho \text{ is nontrivial and } |\operatorname{Im}\rho - \gamma| \le T\},$$

with the completed zeta multiplicity convention. Its punctured window is

$$\mathcal{F}_T^\circ(\gamma) := \mathcal{F}_T(\gamma) \setminus \{\rho_+, \rho_-\}.$$

Theorem 9321.7 (Finite window mirror-negative interpolation)**.** *Let $\mathcal{F}_T(\gamma)$ be a centered nontrivial zero window containing the off-critical mirror pair ρ_+, ρ_-. For every $A > 0$ there is a compact smooth row*

$$g_T \in C_c^\infty(-A, A)$$

such that

$$\widehat{g}_T(\rho_+) = 1, \qquad \widehat{g}_T(\rho_-) = -1,$$

and

$$\widehat{g}_T(\rho) = 0 \qquad (\rho \in \mathcal{F}_T^\circ(\gamma)).$$

Proof. The centered nontrivial zero window is finite because the completed zeta nontrivial-zero ledger is discrete in bounded vertical strips. Apply Definition 9317.4 to this finite set, assigning the values 1 and -1 to ρ_+ and ρ_- and assigning 0 to every point in the punctured window. □

Corollary 9321.8 (Finite window contribution is exactly negative)**.** *For g_T as in Definition 9321.7, the contribution of $\mathcal{F}_T(\gamma)$ to $\mathcal{W}_Z(g_T)$ is*

$$-2.$$

Proof. The mirror pair contributes -2 by the same calculation as Definition 9321.5. Every other zero in the centered window has prescribed readout 0, so the remaining finite-window terms vanish. □

9321.3 Uniform Tail Row

Definition 9321.9 (UniformPWInterpolationTailControl)**.** *The row*

$$\mathsf{UniformPWInterpolationTailControl}$$

asserts that for every off-critical mirror pair and every $\varepsilon > 0$ there are $T > 0$, $A > 0$, and a compact smooth row $g_T \in C_c^\infty(-A, A)$ satisfying the finite window interpolation of Definition 9321.7 and

$$\left| \sum_{\rho \notin \mathcal{F}_T(\gamma)} \widehat{g}_T(\rho)\, \overline{\widehat{g}_T(\iota(\rho))} \right| < \varepsilon.$$

Equivalently, the finite interpolation rows can be chosen with enough uniform Paley–Wiener decay to make the complement of the centered window arbitrarily small in the zero-side mirror quadratic form.

Lemma 9321.10 (Dyadic zero-tail estimate)**.** *Let g be an admissible compact row whose transform satisfies, in the critical strip,*

$$|\widehat{g}(\sigma + it)| \le C_N(1 + |t|)^{-N}$$

for some $N > 1$. If the zero-counting row has the standard dyadic bound

$$\#\{\rho : \ 2^j T < |\operatorname{Im}\rho - \gamma| \le 2^{j+1}T\} \le C\, 2^j T \log(2^j T + 2)$$

for $T \ge 1$, then

$$\sum_{\rho \notin \mathcal{F}_T(\gamma)} |\widehat{g}(\rho)|\, |\widehat{g}(\iota(\rho))| \le C'_N C_N^2 T^{1-2N} \log(T + 2)$$

for a constant C'_N independent of T and g except through C_N.

Proof. The involution $\iota(\rho) = 1 - \overline{\rho}$ preserves the imaginary part of ρ. On the dyadic annulus

$$2^j T < |\operatorname{Im}\rho - \gamma| \le 2^{j+1}T,$$

both $\widehat{g}(\rho)$ and $\widehat{g}(\iota(\rho))$ are bounded by the same vertical decay estimate. The annulus contribution is therefore bounded by

$$C\, 2^j T \log(2^j T + 2)\, C_N^2 (2^j T)^{-2N}.$$

Summing over $j \ge 0$ gives

$$C C_N^2 T^{1-2N} \sum_{j \ge 0} 2^{j(1-2N)} \log(2^j T + 2).$$

Since $N > 1$, the displayed dyadic series is bounded by a constant multiple of $\log(T + 2)$. This gives the claimed estimate. □

Principle 9321.11 (Uniformity is the analytic load)**.** *Definition 9321.10 shows that rapid vertical decay controls the zero tail once the relevant constants are uniform along the finite interpolation rows. The non-formal analytic issue is therefore not finite interpolation itself, but the selection of the finite interpolation solution with Sobolev or Paley–Wiener constants that remain controlled as the centered window expands.*

Theorem 9321.12 (Uniform tail row gives PWZeroPairTailControl)**.** *If* UniformPWInterpolationTailControl *holds, then* PWZeroPairTailControl *holds.*

Proof. Apply Definition 9321.9 with

$$\varepsilon = \frac{1}{2}.$$

The resulting row has

$$\widehat{g}_T(\rho_+) = 1, \qquad \widehat{g}_T(\rho_-) = -1,$$

and its complement outside the centered window has mirror contribution of absolute value less than 1/2. The punctured window contributes zero by Definition 9321.7. Hence the same row satisfies the definition of PWZeroPairTailControl in Definition 9320.8. □

Corollary 9321.13 (Uniform tail row gives a negative zero-side Weil readout)**.** *If* UniformPWInterpolationTailContr *holds and an off-critical mirror pair exists, then there is an admissible compact row g such that*

$$\mathcal{W}_Z(g) < 0.$$

Proof. By Definition 9321.12, PWZeroPairTailControl holds. Then Definition 9320.10 gives

$$\mathcal{W}_Z(g) < -\frac{3}{2} < 0.$$

□

Principle 9321.14 (Zero-side boundary of the negative-Weil route)**.** *The zero-side part of the negative-Weil route is reduced to a single uniform Paley–Wiener selection problem:*

$$\mathsf{UniformPWInterpolationTailControl} \Longrightarrow \mathsf{PWZeroPairTailControl} \Longrightarrow \exists g,\ \mathcal{W}_Z(g) < 0.$$

No archimedean, trivial-zero, pole, or prime-side correction row is used in this implication. Those rows enter only in the independent ArchIndependence *surface of Definition 9320.13.*

Chapter 9322

Paley–Wiener Condition Number Bridge

chapter 9321 reduces the zero-side part of the negative-Weil route to uniform control of compact test rows that interpolate a finite zero window. The present chapter expresses that uniform control as a condition-number estimate. The analytic target is no longer an unspecified global test function. It is the growth rate of the least vertical-decay cost of the finite Paley–Wiener interpolation problem, with optional finite archimedean null constraints.

9322.1 Vertical Decay Cost

Definition 9322.1 (Vertical Paley–Wiener seminorm)**.** *Fix a vertical strip*

$$\sigma_0 \le \mathrm{Re}(s) \le \sigma_1$$

containing the critical strip, and let $N \ge 1$. For a compact explicit-formula test row f, define

$$|f|_{A,N} := \sup_{\sigma\in[\sigma_0,\sigma_1]} \sup_{t\in\mathbb{R}} (1+|t|)^N |\widehat{f}(\sigma+it)|.$$

Thus

$$|\widehat{f}(\sigma+it)| \le |f|_{A,N}(1+|t|)^{-N}$$

throughout the strip.

Definition 9322.2 (Symmetric height zero window)**.** *For $T > 0$, let*

$$\mathcal{Z}_T := \{\rho : \ \rho \text{ is a nontrivial completed zeta zero and } |\operatorname{Im}\rho| \le T\}$$

with the completed zeta multiplicity convention. For an off-critical mirror pair ρ_+, ρ_- contained in $\mathcal{Z}_T$, define the mirror target row

$$b_T(\rho_+) := 1, \qquad b_T(\rho_-) := -1, \qquad b_T(\rho) := 0 \quad (\rho \in \mathcal{Z}_T \setminus \{\rho_+, \rho_-\}).$$

Definition 9322.3 (Paley–Wiener interpolation condition number)**.** *For $N \ge 1$, the finite zero-window interpolation condition number is*

$$\mathfrak{K}_N(T) := \inf\left\{|f|_{A,N} : f \in C_c^\infty(-A,A),\ \widehat{f}(\rho) = b_T(\rho) \text{ for all } \rho \in \mathcal{Z}_T\right\},$$

where the infimum is taken over compact supports and compact test rows admissible for the explicit-formula ledger.

Lemma 9322.4 (Finite condition number is finite)**.** *For every finite zero window $\mathcal{Z}_T$ containing an off-critical mirror pair and every $N \geq 1$,*

$$\mathfrak{K}_N(T) < \infty.$$

Proof. The finite interpolation row Definition 9317.4 gives a compact smooth f realizing the target values b_T on $\mathcal{Z}_T$. Its Laplace–Fourier transform is rapidly decaying in every fixed vertical strip, so $|f|_{A,N} < \infty$. Hence the displayed infimum is finite. □

9322.2 Condition Number Tail Estimate

Definition 9322.5 (Symmetric zero tail)**.** *For a row f realizing the mirror target on $\mathcal{Z}_T$, the symmetric zero tail is*

$$\mathrm{Tail}_T(f) := \sum_{\rho \notin \mathcal{Z}_T} \widehat{f}(\rho)\, \overline{\widehat{f}(\iota(\rho))}.$$

Theorem 9322.6 (Condition-number zero-tail bound)**.** *Assume the standard nontrivial-zero counting bound*

$$\#\{\rho : \ Y < |\operatorname{Im}\rho| \leq 2Y\} \leq C\, Y \log(Y+2)$$

for $Y \geq 1$. Let $N > 1$. If f realizes the mirror target on $\mathcal{Z}_T$ and

$$|f|_{A,N} \leq K,$$

then

$$|\,\mathrm{Tail}_T(f)| \leq C_N K^2 T^{1-2N} \log(T+2),$$

where C_N is independent of T and of f except through the displayed seminorm bound.

Proof. The mirror involution preserves imaginary height. For every zero outside $\mathcal{Z}_T$,

$$|\widehat{f}(\rho)|\,|\widehat{f}(\iota(\rho))| \leq K^2(1+|\operatorname{Im}\rho|)^{-2N}.$$

Apply dyadic decomposition to the ranges

$$2^j T < |\operatorname{Im}\rho| \leq 2^{j+1}T.$$

The contribution of the jth range is bounded by

$$C\, 2^j T \log(2^j T + 2)\, K^2 (2^j T)^{-2N}.$$

Summing in j gives

$$C K^2 T^{1-2N} \sum_{j \geq 0} 2^{j(1-2N)} \log(2^j T + 2).$$

Since $N > 1$, the dyadic series is bounded by a constant multiple of $\log(T+2)$. This gives the required estimate. □

Theorem 9322.7 (Condition growth gives UniformPWInterpolationTailControl)**.** *Suppose there are $N > 1$ and a sequence $T \to \infty$ such that*

$$\mathfrak{K}_N(T)^2 T^{1-2N} \log(T+2) \to 0.$$

Then UniformPWInterpolationTailControl *holds for the off-critical mirror pair.*

Proof. For each T, choose a compact interpolation row f_T realizing b_T and satisfying

$$|f_T|_{A,N} \le 2\mathfrak{K}_N(T).$$

The factor 2 only records a near-minimizer for the infimum. By Definition 9322.6,

$$|\operatorname{Tail}_T(f_T)| \le 4C_N\mathfrak{K}_N(T)^2T^{1-2N}\log(T+2).$$

The right side tends to zero. Given $\varepsilon > 0$, choose T in the sequence so that the displayed bound is less than ε. The row f_T realizes the finite window interpolation and has the required tail bound, which is UniformPWInterpolationTailControl. □

Corollary 9322.8 (Condition growth gives negative zero-side readout)**.** *Under the condition-growth hypothesis of Definition 9322.7, an off-critical mirror pair admits an admissible compact row f with*

$$\mathcal{W}_Z(f) < 0.$$

Proof. Apply Definition 9322.7 and then Definition 9321.13. □

9322.3 Archimedean Condition Number

Definition 9322.9 (Finite archimedean null row)**.** *A finite archimedean null row is a finite family of continuous linear test-function functionals*

$$L_1, \dots, L_J$$

representing a finite truncation of the archimedean, trivial-zero, and pole correction ledger. A test row is J-arch-null when

$$L_j(f) = 0 \qquad (1 \le j \le J).$$

Definition 9322.10 (Arch-null interpolation condition number)**.** *For a finite archimedean null row, define*

$$\mathfrak{K}^{\mathrm{arch}}_{N,J}(T) := \inf\left\{|f|_{A,N} : \widehat{f}(\rho) = b_T(\rho) \text{ for all } \rho \in \mathcal{Z}_T,\ L_j(f) = 0 \text{ for } 1 \le j \le J\right\}.$$

Lemma 9322.11 (Finite arch-null interpolation is finite under independence)**.** *Assume the functionals*

$$\{\widehat{(\cdot)}(\rho) : \rho \in \mathcal{Z}_T\} \cup \{L_1, \dots, L_J\}$$

are linearly independent on $C_c^\infty(-A, A)$. Then

$$\mathfrak{K}^{\mathrm{arch}}_{N,J}(T) < \infty.$$

Proof. Consider the linear map sending f to the vector of zero evaluations and archimedean reads:

$$f \mapsto \left((\widehat{f}(\rho))_{\rho\in\mathcal{Z}_T}, L_1(f), \dots, L_J(f)\right).$$

If this map were not onto, a nonzero linear functional on the finite target space would annihilate its range, giving a nontrivial linear relation among the displayed source functionals. This contradicts the independence hypothesis. Hence the target values b_T and the zero arch reads can be prescribed simultaneously. The resulting compact smooth row has finite $|\cdot|_{A,N}$ seminorm. □

Definition 9322.12 (ArchLimitControl)**.** *The row*

$$\mathsf{ArchLimitControl}$$

asserts that the full archimedean, trivial-zero, and pole correction quadratic form is approximated by finite archimedean null rows on the near-minimizing interpolation family. In the normalized form used here, there are J and a sequence $T \to \infty$ such that for the chosen near-minimizers f_T,

$$|\mathcal{W}_A(f_T)| \to 0.$$

Theorem 9322.13 (Arch condition growth gives NegativeWeilIsolator)**.** *Assume there are $N > 1$, a finite archimedean null row J, and a sequence $T \to \infty$ such that*

$$\left(\mathfrak{K}^{\mathrm{arch}}_{N,J}(T)\right)^2 T^{1-2N} \log(T+2) \to 0,$$

and assume ArchLimitControl *for the corresponding near-minimizing rows. Then an off-critical mirror pair gives a compact test row f satisfying*

$$\mathcal{W}(f) < 0.$$

Proof. Choose near-minimizing rows f_T satisfying the zero interpolation and the finite arch-null constraints, with

$$|f_T|_{A,N} \le 2\mathfrak{K}^{\mathrm{arch}}_{N,J}(T).$$

The zero-tail estimate gives

$$|\operatorname{Tail}_T(f_T)| \to 0.$$

The finite zero window contributes exactly -2: the mirror pair has values 1 and -1, while every other zero in $\mathcal{Z}_T$ has value 0. The finite arch rows vanish by construction, and the remaining arch correction tends to zero by ArchLimitControl. For T sufficiently large,

$$|\operatorname{Tail}_T(f_T)| < \frac{1}{4}, \qquad |\mathcal{W}_A(f_T)| < \frac{1}{4}.$$

Thus

$$\mathcal{W}(f_T) < -2 + \frac{1}{4} + \frac{1}{4} = -\frac{3}{2} < 0.$$

□

9322.4 Condition-Number RH Interface

Definition 9322.14 (PWInterpolationConditionBound)**.** *The row*

$$\mathsf{PWInterpolationConditionBound}$$

asserts that for every off-critical mirror pair there are $N > 1$, a finite archimedean null row J, and a sequence $T \to \infty$ for which

$$\left(\mathfrak{K}^{\mathrm{arch}}_{N,J}(T)\right)^2 T^{1-2N} \log(T+2) \to 0$$

and ArchLimitControl *holds on the associated near-minimizing rows.*

Theorem 9322.15 (Condition-number negative residual theorem)**.** *Assume* PWInterpolationConditionBound *and the paired explicit-formula ledger. If an off-critical mirror pair exists, then it produces a finite prime negative residual detected by a finite Dirichlet classifier.*

Proof. Definition 9322.14 supplies the hypotheses of Definition 9322.13, so there is f with $\mathcal{W}(f) < 0$. The paired explicit formula and Definition 9320.16 turn this into a finite prime negative residual. The classifier readout is then supplied by Definition 9320.17. □

Corollary 9322.16 (Condition-number PFZ fixed-half readout)**.** *Assume* PWInterpolationConditionBound*, the paired explicit-formula ledger, and the PFZ rule that finite negative residuals cannot be hidden as legal closure data. Then every zeta-facing PFZ legal zero packet has fixed-half readout.*

Proof. If a legal zero packet had $\ell \neq 0$, then Definition 9320.2 would place it in an off-critical mirror pair. By Definition 9322.15, that pair produces a finite prime negative residual detected by a finite classifier. By Definition 9320.18, the residual cannot be hidden in a legal PFZ closure object. Hence $\ell = 0$, and Definition 9266.2 gives fixed-half readout. □

Principle 9322.17 (Polynomial condition-number target)**.** *The analytic frontier can be attacked through the growth of* $\mathfrak{K}^{\mathrm{arch}}_{N,J}(T)$*. A polynomial estimate*

$$\mathfrak{K}^{\mathrm{arch}}_{N,J}(T) = O(T^M)$$

would be enough after choosing $N > M + \frac{1}{2}$*, because then*

$$\left(\mathfrak{K}^{\mathrm{arch}}_{N,J}(T)\right)^2 T^{1-2N} \log(T+2) \to 0.$$

Thus the remaining zero-isolation problem is a finite-dimensional Paley–Wiener interpolation condition-number estimate, rather than an unstructured request for a global separating test function.

Chapter 9323

Circle-Triangle Projection Dictionary

chapter 9290 isolates the common generator behind the RH-facing route: generated time, unit-circle phase, orthogonal triangle readout, Zeckendorf–prime ledger, Li scale-phase coordinates, and hidden-scale refusal. The present chapter records how the familiar RH criteria are read as projections of that one generator. The point is not to replace their analytic content with a name. It is to prevent the proof search from treating Li positivity, Weil positivity, Nyman exhaustion, Lee–Yang boundary stability, branch dominance, and tower cofinality as unrelated obligations.

9323.1 Common Generator

Definition 9323.1 (RH circle-triangle-register generator)**.** *An RH circle-triangle-register generator is a certificate packet*

$$\mathcal{R}_\zeta := (\mathcal{G}, \mathsf{PrimeReg}, \mathsf{ZeroRead}, \mathsf{LiRead}, \mathsf{OrthRead}, \mathsf{NoScale})$$

whose rows are as follows. The row $\mathcal{G}$ is a circle-triangle generation packet in the sense of Definition 9290.1. The prime-register row is the source ledger of Definition 9290.5. The zero-read row carries completed non-trivial zero packets. The Li-read row carries $z_\rho = 1 - 1/\rho$. The orthogonal-read row carries the scale-phase defect split of Definition 9290.17. The no-scale row is $\mathsf{NoHiddenScaleFromBEDC}(\mathcal{G})$.

Principle 9323.2 (Projection discipline for RH criteria)**.** *A classical RH criterion is accepted into this dictionary only as a projection of $\mathcal{R}_\zeta$ with a declared visible surface, a declared hidden fiber or residual ledger, and a declared transport back to the zero-read row. A projection may expose time coefficients, energy forms, approximation containers, finite thermal boundaries, branch dominance, or cofinal tower exhaustion. It may not erase the radial scale residual unless the no-scale row has been supplied.*

Definition 9323.3 (Projection row)**.** *A projection row of $\mathcal{R}_\zeta$ is a tuple*

$$\Pi := (V_\Pi, \pi_\Pi : \mathcal{R}_\zeta \to V_\Pi, \Lambda_\Pi, \mathsf{Back}_\Pi).$$

Here V_Π is the visible criterion surface, π_Π is the readout, Λ_Π is the residual ledger retained by that readout, and Back_Π is the transport that converts a violation or positivity certificate on V_Π back into a statement about the zero-read row.

9323.2 Classical Projections

Definition 9323.4 (Li time projection)**.** *The Li time projection reads the iterates*

$$z_\rho^n$$

of the Li coordinate. Its visible surface is the time sequence of defects

$$1 - z_\rho^n.$$

Under a scale-phase read

$$z_\rho = e^{\ell + i\theta},$$

the defect norm splits as

$$|1 - z_\rho^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

Thus the Li surface is the time projection of hidden radial scale.

Principle 9323.5 (Li positivity is a time-facing no-scale test)**.** *Li positivity asks whether the time projection can detect a radial scale leg. If $\ell\#0$, the factor $e^{n\ell}$ is amplified along the append order. Any Li-side positivity theorem that rules out the resulting negative or unbounded time witness is therefore a projection of* NoHiddenScaleFromBEDC*, not an independent source of the unit circle.*

Definition 9323.6 (Weil energy projection)**.** *The Weil energy projection reads the zero packet through a quadratic test form*

$$\mathcal{W}(f)$$

whose visible surface is an energy pairing of zero-side probes. Its residual ledger records which mirror-paired zero packets and test functions were used. An off-line mirror pair supplies an energy channel whose sign can be reversed by an appropriate probe; the projection therefore detects the same radial leg as an energy defect.

Principle 9323.7 (Weil positivity is an energy-facing no-scale test)**.** *Weil positivity is the energy projection of the no-hidden-scale row. A nonzero radial coordinate creates a separated mirror-pair channel, and a test function can read that separation as negative energy. A positive Weil form closes precisely the projection route that would otherwise let radial scale appear as a hidden energy defect.*

Definition 9323.8 (Nyman container projection)**.** *The Nyman container projection reads the zero packet through a closed linear span of scaling correctors in a function space on $(0, 1)$. Its visible surface is the approximation of the target container row, and its residual ledger records separating functionals. If an off-line zero survives, it produces a functional that annihilates the displayed generators while not annihilating the target row. The unexhausted functional is the container-side residue of hidden radial scale.*

Principle 9323.9 (Nyman exhaustion is a container-facing no-scale test)**.** *Nyman exhaustion asks whether the legal scaling generators fill the container without leaving a residual functional. In the dictionary, such a residual functional is the projection of $\ell\#0$ into the container surface. Closing the Nyman span is therefore a container-facing form of the no-hidden-scale obligation.*

Definition 9323.10 (Lee–Yang unit-boundary projection)**.** *The Lee–Yang unit-boundary projection reads finite thermal or algebraic models whose zero sets are forced to a unit circle by a finite positivity or ferromagnetic monotonicity row. Its visible surface is a unit boundary in the finite model. Its residual ledger records which finite interactions, variables, and specialization maps preserve the unit-boundary statement.*

Principle 9323.11 (Lee–Yang boundary is a finite unit model)**.** *Lee–Yang boundary stability is a finite model of the same constraint as $\ell = 0$: zeros may occur only on the unit boundary licensed by the finite packet. It does not by itself prove the zeta zero statement; it supplies a projection pattern showing how finite positivity can enforce unit-circle localization without adding an independent radial register.*

9323.3 Dominance and Cofinality Projections

Definition 9323.12 (Dominant-strand projection)**.** *The dominant-strand projection records which branch, strand, or finite-window carrier actually controls the visible zero-relevant surface. Its residual ledger records branch choices, comparison windows, and maximal-modulus or endpoint dominance rows. A branch point or spectral component is not accepted as the controlling observation merely because it is visible; dominance must be certified as part of the projection row.*

Principle 9323.13 (Dominance prevents false projection collapse)**.** *Dominant-branch theorems are projection-correctness tests. They prevent a visible but non-dominant strand from being mistaken for the zero-relevant generator. In the circle-triangle route, such theorems ensure that the scale-phase leg being tested is the controlling observation vector rather than a discarded or subordinate branch.*

Definition 9323.14 (Tower cofinality projection)**.** *The tower cofinality projection reads a growing family of finite observation spaces*

$$\mathcal{H}_0 \subset \mathcal{H}_1 \subset \cdots$$

with orthogonal residuals

$$v_{m+1} = P_m v_{m+1} + \eta_{m+1}, \qquad \eta_{m+1} \perp \mathcal{H}_m.$$

The visible invariant is

$$|v_{m+1}|^2 = |P_m v_{m+1}|^2 + |\eta_{m+1}|^2.$$

Cofinality asserts that the legal tower exhausts the zero-relevant observation surface rather than leaving a residual functional.

Principle 9323.15 (Tower cofinality is the infinite no-scale test)**.** *The infinite part of the RH route is not a search through unrelated criteria. It is the question whether the fixed generator and its finite observation windows are cofinal. If a nonzero residual remains outside the tower, it is read as a separating functional, an unexhausted scale leg, or a missing source row. The finite Pick tower and the xi Hausdorff moment audit are analytic implementations of this cofinality projection.*

9323.4 Minimal Proof Package

Definition 9323.16 (Prime-register completeness)**.** *Prime-register completeness is the source statement that every zeta oscillation frequency used by the Dirichlet row is generated by the prime*

exponent ledger:

$$n = \prod_p p^{v_p(n)}, \qquad \log n = \sum_p v_p(n) \log p.$$

It supplies the arithmetic part of the generator and prevents zeta phase from being treated as an unledgered continuous frequency.

Definition 9323.17 (Circular-triangular observation completeness)**.** *Circular-triangular observation completeness is the statement that every zero-relevant completed zeta observation has the Li coordinate and orthogonal defect split of Definition 9290.15 and Definition 9290.17, with the source ledgers of $\mathcal{R}_\zeta$ still attached.*

Definition 9323.18 (No hidden radial register)**.** *No hidden radial register is the specialization of Definition 9290.13 to zero-relevant zeta observations. A nonzero radial coordinate*

$$\ell \# 0$$

is neither a unit-circle phase, nor a prime-exponent gauge, nor a free continuous coordinate. It must either enter the ledger as a residual scale row, in which case the object is not pure phase closure, or remain unavailable, in which case the object is not legal.

Theorem 9323.19 (Minimal generator package is conditionally RH-sufficient)**.** *Assume prime-register completeness, circular-triangular observation completeness, no hidden radial register, and the located readback needed to turn absence of radial apartness into equality. Then the BEDC RH section target is inhabited.*

Proof. Let ρ be a completed non-trivial zero packet. Circular-triangular observation completeness supplies

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell + i\theta}$$

and the defect split

$$|1 - z_\rho^n|^2 = (e^{n\ell} - 1)^2 + 2e^{n\ell}(1 - \cos(n\theta)).$$

If $\ell \# 0$, the radial term is a nonzero scale leg. By no hidden radial register, that leg cannot be absorbed into unit phase or into the prime-register row supplied by prime-register completeness. If it is recorded, the object is not pure phase closure; if it is not recorded, the object is not legal. Thus the radial coordinate is not apart from zero. The located readback gives $\ell = 0$, hence $|z_\rho| = 1$. By Definition 9290.16, $\Re(\rho) = 1/2$. Since the zero packet was arbitrary, the section target is inhabited. □

Principle 9323.20 (Status of the projection dictionary)**.** *The projection dictionary identifies a common source mechanism for the major RH-facing criteria. It does not claim that any of the projected analytic theorems has already been discharged. The remaining proof load is the construction of the generator packet $\mathcal{R}_\zeta$, the no-hidden-radial-register theorem, and a cofinal analytic implementation such as the finite Pick tower or the xi Hausdorff moment audit.*

Chapter 9324

Unresolved Projection Obligation Ledger

chapter 9323 gives the projection dictionary for Li, Weil, Nyman, Lee–Yang, dominance, and cofinal tower readings. The present chapter records only the remaining obligations. The algebraic Li coordinate criterion, the scale-phase defect identity, the orthogonal norm split, and the prime-factorization ledger are treated as background rows supplied by earlier chapters. They are not repeated here as open tasks.

9324.1 Open Projection Obligations

Definition 9324.1 (Zeta BEDC embedding obligation)**.** *The obligation*

$$\mathsf{ZetaBEDCEmbedding}$$

asks for a single BEDC observation certificate for each completed non-trivial zeta zero packet. The certificate must package the carrier, classifier, ledger, boundary, and stability rows for completed zeta, zero readout, functional-equation transport, explicit-formula transport, Li readout, Weil pairing, Nyman container rows, and the prime-register source ledger. The prime-register phase source is already visible; the open part is the unified observation certificate.

Definition 9324.2 (No hidden scale obligation)**.** *The obligation*

$$\mathsf{NoHiddenScaleFromBEDC}$$

asks for an internal BEDC theorem that a nonzero radial scale leg cannot be used as a hidden component of a legal pure zeta closure. The theorem must distinguish unit phase, prime-register data, finite encodable residue, and noncompact positive-real scale. Its target is a ledger statement, not a physical analogy.

Definition 9324.3 (Projection faithfulness obligation)**.** *The obligation*

$$\mathsf{TriProjectionFaithfulness}$$

asks for a commutative projection certificate showing that Li time readout, Weil energy readout, and Nyman container readout detect the same radial scale leg when it exists. The open content is not the individual criterion statements; it is the shared source witness and the transport between the three readouts.

Definition 9324.4 (Dominant observation strand obligation)**.** *The obligation*

$$\mathsf{DominantObservationStrand}$$

asks for a source-side certificate selecting the zero-relevant observation strand. A visible branch, local model, or finite carrier is not enough; the selected strand must be certified as controlling the zero-relevant boundary through a dominance, Rouché, energy, or ledger comparison row.

Definition 9324.5 (Unit-boundary transport obligation)**.** *The obligation*

$$\mathsf{UnitBoundaryTransport}$$

asks for a stable transport from the dominant zeta observation strand to a unit-boundary container such as a Lee–Yang, unitary, compact, or Pick positive model. The transport must expose its boundary nonvanishing and ledger compatibility rows; it may not pass through an unrecorded radial scale.

Definition 9324.6 (Cofinal triangular tower obligation)**.** *The obligation*

$$\mathsf{CofinalTriangularTower}$$

asks for a tower of finite triangular observation spaces whose nondegenerate residual legs are PZG-supported in the sense of Definition 9291.15, and whose residuals are exhausted in the limit. Finite orthogonal residual rows and finite Pick positivity are already reduction rows; the open content is the cofinal exhaustion statement, together with the layerwise source-ledger condition of Definition 9291.16, that prevents a radial residual from being hidden at the far end.

Definition 9324.7 (Cofinal prime-class triangular tower obligation)**.** *The obligation*

$$\mathsf{CofinalPrimeClassTriangularTower}$$

asks for a cofinal family of prime-classifier containers in the sense of Definition 9290.7. For every zero-relevant radial scale witness $\ell\#0$, some finite stage of the family must display a container $\mathcal{C}$ whose register, unit, triangular norm, or boundary row reads the scale as a nonzero obstruction:

$$\mathsf{Obstruction}_{\mathcal{C}}(\ell)\#0.$$

The obstruction may be a visible inert parity boundary, a nonunit normalized phase parameter, a negative energy or pairing row, or a separating residual functional. The obligation is the cofinal prime-class form of NoHiddenRadialRegister: a radial leg may fail to be unit phase, but it may not escape every displayed prime-classifier container.

Definition 9324.8 (Layerwise prime cofinality obligation)**.** *The obligation*

$$\mathsf{LayerwisePrimeCofinality}$$

is the layerwise source form of Definition 9324.7. It asks for a tower of primality-supported layers

$$\Pi_0 \subseteq \Pi_1 \subseteq \Pi_2 \subseteq \cdots$$

as in Definition 9291.12, together with container-relative split, inert, and ramified reads at the finite stages, such that every zero-relevant radial scale witness $\ell\#0$ is detected at some stage:

$$\exists m \quad \mathsf{Obstruction}_{\Pi_m}(\ell)\#0.$$

This is the constructive form of cofinal prime-class detection: the obstruction must be found by a displayed atom layer and its classifier, not by an unledgered prohibition after the zero packet has been inspected.

Definition 9324.9 (Diagonal safety obligation)**.** *The obligation*

DiagonalSafety

asks that the proof system be source-generated and append-only. It forbids a local oracle strategy that first detects a bad zero-side direction and then modifies the certificate to cover that direction. Negative Li witnesses, Weil probes, and Nyman separating functionals must be excluded by the source-side construction itself.

Definition 9324.10 (No hidden radial register obligation)**.** *The obligation*

NoHiddenRadialRegister

is the strengthened gauge form of Definition 9324.2. It asks for a theorem that a nonzero radial coordinate is not unit phase, not a prime-register gauge, not freely erasable finite data, and not an admissible tower tail. A recorded radial coordinate changes the object into a scale-ledgered row; an unrecorded radial coordinate violates ledger completeness.

Definition 9324.11 (Triangular projection equivalence obligation)**.** *The obligation*

TriangularProjectionEquivalence

asks for the converse direction of the projection dictionary: any Li, Weil, or Nyman defect relevant to the completed zeta zero packet must come from the same radial scale witness, and the three projection defects must share a common source certificate. This is the diagrammatic equivalence row tying time, unit, and ledger projections together.

9324.2 Conditional Package

Theorem 9324.12 (Projection obligation package is conditionally sufficient)**.** *Assume the obligations*

ZetaBEDCEmbedding, NoHiddenScaleFromBEDC, TriProjectionFaithfulness, DominantObservationStrand, Ur

Then the BEDC RH section target is inhabited.

Proof. The embedding obligation places each completed non-trivial zero packet in the BEDC observation system. If a radial scale witness were present, the faithfulness and equivalence obligations would expose it through the Li, Weil, and Nyman projections as the same source-side defect. Dominance ensures that the defect belongs to the zero-relevant strand; unit-boundary transport prevents it from passing through a legal unit container; cofinal triangular exhaustion prevents it from escaping to the tower tail; cofinal prime-class detection prevents it from escaping all prime-classifier containers; diagonal safety prevents posterior repair. The hidden-scale and hidden-radial-register obligations then rule out the radial witness inside a legal closure. The zero packet is therefore forced onto the unit Li boundary, and the earlier Li criterion transports that row to the critical-line section. □

9324.3 Compressed Core

Principle 9324.13 (Minimal open core)**.** *The preceding package compresses to three core proof tasks:*

$$\mathsf{ZetaBEDCEmbedding}, \qquad \mathsf{NoHiddenRadialRegister}, \qquad \mathsf{LayerwisePrimeCofinality}.$$

The first supplies the legal observation object, the second forbids a nonzero radial scale as a hidden gauge or ledger-free residual, and the third supplies a cofinal atom-layer tower whose prime classifiers detect any nonzero radial leg at some finite or tower stage. Cofinal prime-class triangular detection, cofinal triangular exhaustion, tri-projection faithfulness, dominant-strand selection, unit-boundary transport, diagonal safety, and projection equivalence are support rows that make these three tasks usable in the completed zeta carrier.

Principle 9324.14 (Status of the unresolved projection ledger)**.** *This ledger lists open obligations only. The already displayed algebraic identities, coordinate criteria, finite factorization rows, and moment reductions are not counted as remaining proof load here. The route to RH through this ledger is conditional on constructing the listed source-side certificates.*

Chapter 9325

Triadic Solenoid Pick Tower

chapter 9289 isolates the source-side obligation behind the phase-prime-triangle reading: completed zeta must enter as a certificate-bearing object, and its Pick positivity must be produced from source data rather than from a posteriori zero information. The present chapter records the finite tower that makes that obligation concrete. Each finite layer builds a positive Pick kernel from a finite solenoid source, positive spectral atoms, and a Hilbert factorization. The remaining unclosed content is the convergence of this finite tower to the completed zeta kernel.

9325.1 Finite Solenoid Sources

Definition 9325.1 (Triadic solenoid Pick layer)**.** *A triadic solenoid Pick layer is a finite packet*

$$\mathcal{T}_m := (S_m, \Sigma_{S_m}, (\gamma_{m,j})_{j=1}^{N_m}, (\alpha_{m,j})_{j=1}^{N_m}, (c_{m,j})_{j=1}^{N_m})$$

with the following rows:

$$S_m \subset \mathsf{Prime}^{\uparrow}$$

is a finite prime window;

$$\Sigma_{S_m} \to \mathbb{T}$$

is the finite solenoid phase projection of Definition 9413.779;

$$\gamma_{m,j}$$

are displayed finite source roles or active modes over the same window;

$$\alpha_{m,j} \in \mathsf{Real}^{\uparrow}_{>0}$$

are their positive frequency readouts; and

$$c_{m,j} \in \mathsf{Real}^{\uparrow}_{>0}$$

are positive spectral weights. The packet contains no independent $\mathsf{Real}^{\uparrow}_{>0}$ *source fiber; positive real numbers occur only as frequency and weight readouts certified by the finite layer.*

Principle 9325.2 (Triadic role of a finite layer)**.** *The layer* $\mathcal{T}_m$ *separates the three BEDC roles:*

$$\mathsf{P}_m := \textit{finite prime ledger and solenoid phase fiber},$$

$$\mathsf{T}_m := \textit{finite active-mode phase flow},$$

and

$$\mathsf{U}_m := \textit{positive spectral weight ledger}.$$

The positivity used below is therefore a finite source row, not an untyped assumption that a limiting positive kernel exists.

Definition 9325.3 (Cofinal finite solenoid tower)**.** *A cofinal finite solenoid Pick tower is a sequence*

$$\mathcal{T}_1, \mathcal{T}_2, \dots$$

such that

$$S_1 \subset S_2 \subset \cdots \subset \mathsf{Prime}^{\uparrow}, \qquad \bigcup_m S_m = \mathsf{Prime}^{\uparrow}$$

at the declared source-ledger scope. The cofinality row states only that every displayed prime row eventually appears in a finite window; it does not assert convergence to the completed zeta kernel.

9325.2 Finite Herglotz Kernels

Definition 9325.4 (Finite positive spectral measure)**.** *For a layer $\mathcal{T}_m$, the finite positive spectral measure is*

$$\nu_m := \sum_{j=1}^{N_m} c_{m,j} \delta_{\alpha_{m,j}^2}.$$

It is a finite source-side measure because each support point $\alpha_{m,j}^2$ and each weight $c_{m,j}$ is displayed in the layer packet and satisfies positivity at that layer.

Definition 9325.5 (Finite Stieltjes and Herglotz readouts)**.** *The finite Stieltjes readout is*

$$S_m(x) := \int_0^\infty \frac{1}{x+t}\, d\nu_m(t) = \sum_{j=1}^{N_m} \frac{c_{m,j}}{x + \alpha_{m,j}^2}.$$

For a centered half-plane coordinate w, define

$$H_m(w) := 2wS_m(w^2) = \sum_{j=1}^{N_m} \frac{2c_{m,j}w}{w^2 + \alpha_{m,j}^2}.$$

Theorem 9325.6 (Finite layer is Herglotz on the right half-plane)**.** *If $\Re(w) > 0$, then*

$$\Re H_m(w) \geq 0.$$

Proof. It is enough to check one summand. For $\alpha > 0$ and $\Re(w) > 0$,

$$\Re \frac{w}{w^2 + \alpha^2} = \frac{\Re(w)(|w|^2 + \alpha^2)}{|w^2 + \alpha^2|^2} \geq 0.$$

Multiplication by the positive coefficient $2c_{m,j}$ preserves nonnegativity, and a finite sum of nonnegative real parts has nonnegative real part. □

Definition 9325.7 (Finite right-half-plane Pick kernel)**.** *The right-half-plane Pick kernel of the finite layer is*

$$P_m(w,\eta) := \frac{H_m(w) + \overline{H_m(\eta)}}{w + \overline{\eta}}.$$

Definition 9325.8 (Finite Hilbert feature map)**.** *For each active mode set*

$$\phi_{m,j}(w) := \sqrt{2c_{m,j}}\,\frac{(w,\alpha_{m,j})}{w^2+\alpha_{m,j}^2} \in \mathbb{C}^2.$$

Let

$$\mathcal{H}_m := \bigoplus_{j=1}^{N_m} \mathbb{C}^2, \qquad \Phi_m(w) := \bigoplus_{j=1}^{N_m} \phi_{m,j}(w).$$

Theorem 9325.9 (Finite Pick kernel has an explicit Hilbert factorization)**.** *For every finite layer,*

$$P_m(w,\eta) = \langle \Phi_m(w), \Phi_m(\eta)\rangle_{\mathcal{H}_m}.$$

Consequently P_m is positive definite.

Proof. For one summand,

$$H_{m,j}(w) := \frac{2c_{m,j}w}{w^2+\alpha_{m,j}^2}.$$

A direct computation gives

$$\frac{H_{m,j}(w) + \overline{H_{m,j}(\eta)}}{w+\overline{\eta}} = 2c_{m,j}\frac{w\overline{\eta} + \alpha_{m,j}^2}{(w^2+\alpha_{m,j}^2)(\overline{\eta}^2+\alpha_{m,j}^2)}.$$

The right side is exactly

$$\langle \phi_{m,j}(w), \phi_{m,j}(\eta)\rangle_{\mathbb{C}^2}.$$

Summing over j gives the displayed factorization in $\mathcal{H}_m$. Positive definiteness follows from the Gram form. □

9325.3 Disk Kernels and the Limiting Target

Definition 9325.10 (Disk transfer of the finite kernel)**.** *Use the centered Cayley coordinate*

$$w(z) := \frac{1+z}{2(1-z)}.$$

Define

$$f_m(z) := H_m(w(z))$$

and

$$K_m(z,\zeta) := \frac{f_m(z) + \overline{f_m(\zeta)}}{1 - z\overline{\zeta}}.$$

The feature map Ψ_m is obtained from $\Phi_m \circ w$ with the standard Cayley kernel factor. Thus K_m is again a finite Gram kernel.

Definition 9325.11 (Completed zeta Pick target)**.** *The completed zeta Pick target is*

$$f_\xi(z) := 2\frac{\xi'}{\xi}\left(\frac{1}{1-z}\right),$$

and

$$K_\xi(z,\zeta) := \frac{f_\xi(z) + \overline{f_\xi(\zeta)}}{1 - z\overline{\zeta}}.$$

Definition 9325.12 (Triadic solenoid Pick tower convergence)**.** *A cofinal finite solenoid Pick tower converges to the completed zeta Pick target when, for every finite disk window $F \subset \mathbb{D}$,*

$$K_m(z,\zeta) \to K_\xi(z,\zeta) \qquad (z,\zeta \in F)$$

with the convergence row supplied by the same source ledger. This is the hard analytic row of the tower.

Theorem 9325.13 (Finite tower convergence gives positive completed kernel)**.** *If a cofinal finite solenoid Pick tower converges to K_ξ, then K_ξ is positive definite.*

Proof. Let $z_1, \ldots, z_N \in \mathbb{D}$ and $a_1, \ldots, a_N \in \mathbb{C}$. For each m, positive definiteness of K_m gives

$$\sum_{i,j=1}^{N} a_i\overline{a_j}K_m(z_i,z_j) \geq 0.$$

Taking the displayed finite-window limit yields

$$\sum_{i,j=1}^{N} a_i\overline{a_j}K_\xi(z_i,z_j) \geq 0.$$

Thus K_ξ is positive definite. □

Theorem 9325.14 (Triadic solenoid Pick tower is conditionally RH-sufficient)**.** *If a cofinal finite solenoid Pick tower converges to the completed zeta Pick target and the standard logarithmic-derivative zero-exclusion transport is available for the same completed zeta carrier, then the BEDC RH section target is inhabited.*

Proof. By Definition 9325.13, K_ξ is positive definite. Hence

$$K_\xi(z,z) \geq 0$$

for every disk row z. Since

$$K_\xi(z,z) = \frac{2\Re f_\xi(z)}{1-|z|^2},$$

we obtain $\Re f_\xi(z) \geq 0$ on the disk. Transporting along $s = 1/(1-z)$ gives the nonnegative real-part read for ξ'/ξ on the right-half critical domain. The stated logarithmic-derivative zero-exclusion transport rules out an off-line zero there; the functional-equation mirror then rules out its reflected off-line companion. Thus the fixed-half section is obtained. □

9325.4 Moment Positivity Target

Definition 9325.15 (Moment positivity for the completed zeta carrier)**.** *Let*

$$S_\xi(x) := 2\frac{d}{dx}\log\xi\left(\frac{1}{2}+\sqrt{x}\right).$$

This convention gives

$$2wS_\xi(w^2) = 2\frac{\xi'}{\xi}\left(\frac{1}{2}+w\right)$$

on the centered branch used by Definition 9325.11. The BEDC moment positivity target for ξ is the certificate that S_ξ is a Stieltjes transform

$$S_\xi(x) = \int_0^\infty \frac{1}{x+t}\,d\nu(t), \qquad \nu \geq 0,$$

with ν constructed from the completed zeta source ledger. Equivalently, when a source-side expansion

$$S_\xi(x) = \sum_{n\geq 0}(-1)^n\mu_n x^{-n-1}$$

is displayed, the moment rows must satisfy the Stieltjes conditions

$$\det(\mu_{i+j})_{i,j=0}^N \geq 0, \qquad \det(\mu_{i+j+1})_{i,j=0}^N \geq 0$$

for every displayed finite N.

Principle 9325.16 (Moment positivity is the remaining finite-tower discharge)**.** *The finite objects of Definition 9325.1 through Definition 9325.9 are already explicit. The remaining proof load is not the abstract existence of a positive kernel; it is the source-side Stieltjes moment certificate of Definition 9325.15. Such a certificate constructs the positive measure, supplies finite atomic approximants by standard moment approximation rows, gives the convergence of Definition 9325.12, and therefore feeds the conditional RH-sufficient theorem above.*

Principle 9325.17 (Status of the triadic solenoid Pick tower)**.** *The tower turns the Source Hilbert positivity request of Definition 9289.24 into a finite sequence of concrete certificate rows: finite solenoid source, finite frequency flow, positive spectral atoms, finite Herglotz function, finite Pick kernel, and finite Hilbert factorization. The limiting identification with the completed zeta kernel remains exactly the* BEDCMomentPositivityForXi *obligation.*

Chapter 9326

Xi Hausdorff Moment Audit

chapter 9325 reduces the source-Hilbert positivity request to the moment positivity of the completed zeta carrier. The present chapter makes that request finite and auditable. Instead of asking directly for a Hilbert space factorization of the completed Pick kernel, it asks for a concrete Hausdorff moment sequence extracted from the source-side logarithmic derivative at a point where the Euler product is safely on the absolutely convergent side. The positive measure, Herglotz function, Pick kernel, and Hilbert factorization are then reconstructed from the moment certificate.

9326.1 Centered Source Moments

Definition 9326.1 (Centered completed zeta carrier)**.** *Let*

$$F(w) := \xi\left(\frac{1}{2} + w\right)$$

and let

$$G(x) := F(\sqrt{x}) = \xi\left(\frac{1}{2} + \sqrt{x}\right)$$

on a displayed branch of $\sqrt{x}$ *near* $x = 1$*. The source base point is*

$$x_0 = 1, \qquad \frac{1}{2} + \sqrt{x_0} = \frac{3}{2}.$$

This base point lies in the half-plane where the zeta Euler product is absolutely convergent.

Definition 9326.2 (Xi source Stieltjes candidate)**.** *The xi source Stieltjes candidate is the normalized logarithmic derivative*

$$S_\xi(x) := 2\frac{G'(x)}{G(x)} = 2\frac{d}{dx}\log\xi\left(\frac{1}{2} + \sqrt{x}\right).$$

The factor 2 *matches Definition 9325.15 and ensures that*

$$2wS_\xi(w^2) = 2\frac{\xi'}{\xi}\left(\frac{1}{2} + w\right)$$

on the centered branch used by the completed zeta Pick target.

Definition 9326.3 (Xi Hausdorff moment row)**.** *The xi Hausdorff moment row is the sequence*

$$a_n := \frac{(-1)^n}{n!} \frac{d^n}{dx^n} S_\xi(x)\Big|_{x=1}, \qquad n \geq 0.$$

Each a_n is a source-side coefficient because it is read from the Taylor data of S_ξ at the base point $x = 1$, corresponding to $s = 3/2$.

Principle 9326.4 (Moment row is independent of zero-location input)**.** *The sequence (a_n) is not computed from the location of zeros. It is extracted from the completed zeta carrier through the logarithmic derivative at a source-side base point. A later proof that (a_n) is a positive moment sequence may imply zero-location information, but that information is not an input to the moment row.*

9326.2 Finite Hausdorff Audit

Definition 9326.5 (Finite difference rows)**.** *For a sequence (a_n), define*

$$\Delta a_n := a_{n+1} - a_n$$

and, for $k \geq 0$,

$$\Delta^k a_n := \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} a_{n+j}.$$

Definition 9326.6 (Xi Hausdorff moment positivity)**.** *The certificate*

$$\mathsf{XiHausdorffMomentPositivity}$$

is the family of finite inequalities

$$(-1)^k \Delta^k a_n \geq 0 \qquad (n, k \geq 0)$$

for the moment row of Definition 9326.3. Each displayed pair (n, k) is one finite audit item.

Principle 9326.7 (Hausdorff positivity is a finite audit family)**.** *The positivity target is not an uninspected infinite assertion about a Hilbert space. A failure is witnessed by a finite pair (n, k) with*

$$(-1)^k \Delta^k a_n < 0.$$

A proof supplies a stable rule producing the nonnegativity certificate for each finite pair. Thus the target is an infinite family only in the same sense as an induction or a recursive certificate family, not as an unledgered analytic object.

Theorem 9326.8 (Hausdorff positivity gives a source measure)**.** *Assume* $\mathsf{XiHausdorffMomentPositivity}$ *and the Hausdorff moment theorem for the displayed sequence. Then there is a positive measure μ on $[0, 1]$ such that*

$$a_n = \int_0^1 y^n \, d\mu(y) \qquad (n \geq 0).$$

Proof. By Definition 9326.6, the sequence (a_n) is completely monotone in the Hausdorff sense:

$$(-1)^k \Delta^k a_n \geq 0$$

for every finite pair (n, k). The Hausdorff moment theorem identifies such sequences exactly with moment sequences of positive measures on $[0, 1]$. Applying that theorem gives the displayed measure μ. □

9326.3 From Moments to Stieltjes and Pick Kernels

Definition 9326.9 (Hausdorff transform readout)**.** *For the positive measure μ of Definition 9326.8, define*

$$\widetilde{S}_\xi(x) := \int_0^1 \frac{1}{1+(x-1)y}\, d\mu(y).$$

Near $x = 1$, its Taylor expansion is

$$\widetilde{S}_\xi(x) = \sum_{n\geq 0} (-1)^n a_n (x-1)^n.$$

Definition 9326.10 (Analytic determination row)**.** *The analytic determination row for the xi Hausdorff audit states that the Taylor expansion of S_ξ at $x = 1$ is determined by the moment row (a_n) on a nonempty source neighborhood and that the equality*

$$S_\xi(x) = \widetilde{S}_\xi(x)$$

continues through the displayed zero-free analytic domain by uniqueness of analytic continuation.

Theorem 9326.11 (Hausdorff audit gives the Stieltjes target)**.** *Assume* XiHausdorffMomentPositivity*, the Hausdorff moment theorem for the moment row, and the analytic determination row of Definition 9326.10. Then S_ξ has a positive Stieltjes-type representation.*

Proof. Definition 9326.8 supplies $\mu \geq 0$ on $[0, 1]$. By Definition 9326.9,

$$\widetilde{S}_\xi(x) = \sum_{n\geq 0} (-1)^n a_n (x-1)^n$$

near $x = 1$. The definition of a_n gives the same Taylor expansion for S_ξ at $x = 1$, and the analytic determination row transports the local equality to the displayed analytic domain. Finally the change of variables

$$y = \frac{1}{1+t}$$

turns the Hausdorff transform into a Stieltjes-type transform on $[0, \infty)$, with positive pushed-forward measure. Thus the moment certificate discharges the Stieltjes target of Definition 9325.15. □

Theorem 9326.12 (Xi Hausdorff audit is conditionally RH-sufficient)**.** *Assume* XiHausdorffMomentPositivity*, the Hausdorff moment theorem for the moment row, the analytic determination row, and the standard Stieltjes-to-zero-exclusion transport for the completed zeta carrier. Then the BEDC RH section target is inhabited.*

Proof. By Definition 9326.11, the function S_ξ is represented by a positive Stieltjes-type transform. Hence the associated half-plane function

$$H_\xi(w) := 2wS_\xi(w^2)$$

has nonnegative real part on the right half-plane. The corresponding Pick kernel is therefore positive. The assumed Stieltjes-to-zero-exclusion transport is exactly the analytic discharge route recorded in Definition 9289.25: an off-line zero would produce a forbidden local real-part read near the logarithmic-derivative pole. Thus the completed non-trivial zero packets are transported to the fixed half-line section. □

9326.4 Truncated Witnesses

Definition 9326.13 (Finite Hausdorff audit window)**.** *For $N \geq 0$, the finite Hausdorff audit window is the finite list of inequalities*

$$(-1)^k \Delta^k a_n \geq 0 \qquad (0 \leq n,\ 0 \leq k,\ n+k \leq N).$$

It is a finite BEDC witness surface: all coefficients involved are among

$$a_0, \ldots, a_N.$$

Principle 9326.14 (Truncated moments produce finite positive packets)**.** *When a finite audit window is accompanied by the standard truncated Hausdorff moment construction, it yields a finite positive atomic measure*

$$\mu_N = \sum_{j=1}^{M_N} c_{N,j} \delta_{y_{N,j}}, \qquad c_{N,j} > 0, \quad 0 \leq y_{N,j} \leq 1,$$

matching the displayed moments in that window. The finite transform

$$S_N(x) := \int_0^1 \frac{1}{1+(x-1)y}\, d\mu_N(y)$$

then gives a finite Herglotz function

$$H_N(w) := 2wS_N(w^2),$$

and hence a finite Pick kernel with a finite Hilbert factorization of the form constructed in Definition 9325.9.

Principle 9326.15 (Status of the xi Hausdorff audit)**.** *The RH-facing proof load is reduced to a finite-audit family:*

$$\mathsf{XiHausdorffMomentPositivity}.$$

The finite check at (n,k) is the explicit inequality

$$(-1)^k \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} \frac{(-1)^{n+j}}{(n+j)!} \frac{d^{n+j}}{dx^{n+j}} S_\xi(x) \bigg|_{x=1} \geq 0.$$

Together with the analytic determination row, this moment certificate constructs the measure, the Stieltjes transform, the Herglotz function, the Pick kernel, and the Hilbert factorization required by the finite tower.

Chapter 9327

Xi Triangular Moment Certificate Tower

chapter 9326 identifies XiHausdorffMomentPositivity as the finite-audit family behind the completed zeta Pick kernel. The present chapter expands that family into a triangular certificate tower. Each finite level is a concrete difference table, a positive Bernstein ledger, a finite atomic measure, a finite Herglotz function, and a Pick kernel with an explicit Hilbert factorization. Thus the phrase "positive spectral measure" is not used as a primitive existence claim; it is approached through finite certificate rows.

9327.1 Triangular Difference Ledger

Definition 9327.1 (Hausdorff difference triangle)**.** *For the xi moment row (a_n) of Definition 9326.3, the Hausdorff difference triangle is the array*

$$D_{n,k} := (-1)^k \Delta^k a_n, \qquad n, k \geq 0.$$

The entry $D_{n,k}$ is finite: it depends only on

$$a_n, a_{n+1}, \ldots, a_{n+k}.$$

Definition 9327.2 (Finite triangular moment certificate)**.** *For $N \geq 0$, a finite triangular moment certificate is the finite packet*

$$\mathcal{M}_N := \big((a_0, \ldots, a_N), (D_{n,k})_{n+k \leq N}, (\beta_{N,j})_{0 \leq j \leq N}\big),$$

where

$$D_{n,k} = (-1)^k \Delta^k a_n$$

and

$$\beta_{N,j} := \binom{N}{j} D_{j,N-j}.$$

The certificate is positive when

$$D_{n,k} \geq 0 \qquad (n + k \leq N).$$

Principle 9327.3 (Triangular audit is local and finite)**.** *Every level $\mathcal{M}_N$ is checked inside a finite triangular table. A failure of* XiHausdorffMomentPositivity *appears as a specific entry $D_{n,k} < 0$. A successful level exposes positive Bernstein weights $\beta_{N,j}$ and therefore a finite positive measure without calling on a limiting measure as an input.*

9327.2 Bernstein Measure Layer

Definition 9327.4 (Bernstein basis and moment functional)**.** *For $N \geq 0$, set*

$$B_{N,j}(y) := \binom{N}{j} y^j (1-y)^{N-j}, \qquad 0 \leq j \leq N.$$

The finite moment functional L_N is defined on monomials of degree at most N by

$$L_N(y^r) := a_r \qquad (0 \leq r \leq N)$$

and extended linearly to polynomials of degree at most N.

Theorem 9327.5 (Bernstein weights are finite differences)**.** *For $0 \leq j \leq N$,*

$$L_N(B_{N,j}) = \binom{N}{j} (-1)^{N-j} \Delta^{N-j} a_j = \beta_{N,j}.$$

Consequently a positive triangular moment certificate has $L_N(B_{N,j}) \geq 0$ for every j.

Proof. Expanding the Bernstein basis gives

$$B_{N,j}(y) = \binom{N}{j} y^j (1-y)^{N-j} = \binom{N}{j} \sum_{r=0}^{N-j} (-1)^r \binom{N-j}{r} y^{j+r}.$$

Applying L_N gives

$$L_N(B_{N,j}) = \binom{N}{j} \sum_{r=0}^{N-j} (-1)^r \binom{N-j}{r} a_{j+r}.$$

This is exactly

$$\binom{N}{j} (-1)^{N-j} \Delta^{N-j} a_j,$$

which is $\beta_{N,j}$ by Definition 9327.2. □

Definition 9327.6 (Bernstein atomic measure)**.** *The Bernstein atomic measure associated to a positive $\mathcal{M}_N$ is*

$$\mu_N := \sum_{j=0}^{N} \beta_{N,j} \delta_{j/N}.$$

It is a finite positive measure on $[0,1]$ because every $\beta_{N,j}$ is a displayed nonnegative entry of the triangular certificate.

Principle 9327.7 (Bernstein layer is the unit-positive row)**.** *The measure μ_N is the finite U_N row of the triangular certificate tower. It is not asserted by compactness or by a completed moment theorem; its atoms and weights are written directly from the finite difference triangle.*

9327.3 Finite Stieltjes and Herglotz Layer

Definition 9327.8 (Triangular finite Stieltjes readout)**.** *For a positive triangular certificate $\mathcal{M}_N$, define*

$$S_N(x) := \int_0^1 \frac{1}{1+(x-1)y}\, d\mu_N(y).$$

Equivalently,

$$S_N(x) = \sum_{j=0}^{N} \frac{\beta_{N,j}}{1+(x-1)j/N}.$$

Definition 9327.9 (Triangular finite Herglotz readout)**.** *The associated half-plane readout is*

$$H_N(w) := 2wS_N(w^2).$$

For atoms with $j > 0$, set

$$y_j := \frac{j}{N}, \qquad t_j := \frac{1-y_j}{y_j}, \qquad c_j := \frac{\beta_{N,j}}{y_j}.$$

Then the corresponding summand has the form

$$\frac{2c_j w}{w^2+t_j}.$$

The atom $j = 0$ contributes the positive linear term

$$2\beta_{N,0}w.$$

Theorem 9327.10 (Triangular Herglotz positivity)**.** *If $\mathcal{M}_N$ is positive and $\Re(w) > 0$, then*

$$\Re H_N(w) \geq 0.$$

Proof. For every atom with $j > 0$, the summand is

$$\frac{2c_j w}{w^2+t_j}$$

with $c_j \geq 0$ and $t_j \geq 0$. The computation in Definition 9325.6 gives nonnegative real part on the right half-plane. The atom $j = 0$ contributes $2\beta_{N,0}w$, whose real part is $2\beta_{N,0}\Re(w) \geq 0$. A finite sum of nonnegative real parts has nonnegative real part. □

9327.4 Finite Pick and Hilbert Layer

Definition 9327.11 (Triangular finite Pick kernel)**.** *The half-plane Pick kernel of $\mathcal{M}_N$ is*

$$P_N(w,\eta) := \frac{H_N(w) + \overline{H_N(\eta)}}{w+\overline{\eta}}.$$

The disk kernel is obtained through the centered Cayley coordinate

$$w(z) := \frac{1+z}{2(1-z)}$$

by

$$K_N(z,\zeta) := \frac{H_N(w(z)) + \overline{H_N(w(\zeta))}}{1-z\overline{\zeta}}.$$

Definition 9327.12 (Triangular Hilbert feature map)**.** *For each atom with $j > 0$, define*

$$\phi_{N,j}(w) := \sqrt{2c_j}\frac{(w, \sqrt{t_j})}{w^2 + t_j} \in \mathbb{C}^2.$$

For the atom $j = 0$, define a one-dimensional feature

$$\phi_{N,0}(w) := \sqrt{2\beta_{N,0}}.$$

Let

$$\mathcal{H}_N^{\triangle} := \mathbb{C} \oplus \bigoplus_{j=1}^{N} \mathbb{C}^2$$

and

$$\Phi_N^{\triangle}(w) := \phi_{N,0}(w) \oplus \bigoplus_{j=1}^{N} \phi_{N,j}(w).$$

Theorem 9327.13 (Triangular Pick kernel has a finite Hilbert factorization)**.** *For every positive triangular moment certificate,*

$$P_N(w, \eta) = \left\langle \Phi_N^{\triangle}(w), \Phi_N^{\triangle}(\eta) \right\rangle_{\mathcal{H}_N^{\triangle}}.$$

Consequently P_N and K_N are positive definite.

Proof. For $j > 0$, the rank-two factorization is the same calculation as Definition 9325.9 with $\alpha^2 = t_j$ and coefficient c_j. For $j = 0$, the Herglotz summand is $2\beta_{N,0}w$, and its Pick quotient is

$$\frac{2\beta_{N,0}w + 2\beta_{N,0}\overline{\eta}}{w + \overline{\eta}} = 2\beta_{N,0},$$

which is the Gram kernel of $\phi_{N,0}$. Summing the finite Gram factorizations gives the displayed factorization. Positive definiteness is immediate from the Gram form, and the disk kernel is its Cayley transport. □

9327.5 Tower Convergence Target

Definition 9327.14 (Triangular moment certificate tower)**.** *A triangular moment certificate tower is a sequence*

$$\mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \ldots$$

such that each $\mathcal{M}_N$ is positive and the associated transforms S_N converge locally to S_ξ on the displayed analytic domain of Definition 9326.10. The convergence row is part of the certificate; finite positivity alone does not assert it.

Theorem 9327.15 (Triangular certificate tower is conditionally RH-sufficient)**.** *If a triangular moment certificate tower exists, its convergence row identifies the limit with S_ξ, and the standard Stieltjes-to-zero-exclusion transport is available for the completed zeta carrier, then the BEDC RH section target is inhabited.*

Proof. Each level supplies a positive Pick kernel by Definition 9327.13. The convergence row gives the limiting completed zeta kernel, so the limit kernel is positive definite. The Stieltjes-to-zero-exclusion transport then applies exactly as in Definition 9326.12, forcing the completed non-trivial zero packets onto the fixed half-line section. □

Principle 9327.16 (Status of the triangular moment tower)**.** *The triangular tower is the finite constructive form of* XiHausdorffMomentPositivity*. Its level-N data are the moment prefix, the finite difference triangle, the Bernstein weights, the atomic measure, the Herglotz function, the Pick kernel, and the Hilbert feature map. A failed entry gives a finite witness; a successful cofinal tower gives the positive-kernel route required by the RH-facing certificate.*

Chapter 9328

Final Moment Obligation Core

chapter 9326 and chapter 9327 reduce the positive-kernel route to finite moment data. This chapter records the remaining proof load without repeating the reductions. The RH-facing construction has reached a point where the only unresolved mathematical content is the source-side positivity of a specific moment-difference family, or a stronger norm formula explaining that positivity.

9328.1 Moment Difference Surface

Definition 9328.1 (Final moment difference)**.** *For the xi moment row of Definition 9326.3, define*

$$M_{n,k} := (-1)^k \Delta^k a_n.$$

Equivalently,

$$M_{n,k} = \sum_{j=0}^{k} (-1)^j \binom{k}{j} a_{n+j}.$$

Using the defining derivatives of a_n, this becomes

$$M_{n,k} = (-1)^n \sum_{j=0}^{k} \binom{k}{j} \frac{1}{(n+j)!} \left. \frac{d^{n+j}}{dx^{n+j}} S_\xi(x) \right|_{x=1}.$$

Every $M_{n,k}$ is a finite source-side datum.

Definition 9328.2 (Moment norm certificate)**.** *The statement*

$$\mathsf{SourceMomentNormCertificate}$$

asserts that for every finite pair (n,k) there is a Hilbert row $\mathcal{H}_{n,k}$ and a vector $\Psi_{n,k} \in \mathcal{H}_{n,k}$ such that

$$M_{n,k} = \|\Psi_{n,k}\|^2_{\mathcal{H}_{n,k}}.$$

The Hilbert row and the vector must be constructed from the prime, Gamma, rational-boundary, and completed-zeta source ledgers. They may not be defined by using zero locations as posterior data.

Theorem 9328.3 (Norm certificates imply Hausdorff positivity)**.** *If* $\mathsf{SourceMomentNormCertificate}$ *holds, then* $\mathsf{XiHausdorffMomentPositivity}$ *holds.*

Proof. For each finite pair (n, k), the norm certificate gives

$$M_{n,k} = \|\Psi_{n,k}\|^2_{\mathcal{H}_{n,k}}.$$

Norm squares are nonnegative, so $M_{n,k} \geq 0$. Since $M_{n,k} = (-1)^k \Delta^k a_n$, this is exactly the finite inequality family of Definition 9326.6. □

9328.2 Source Decomposition Target

Definition 9328.4 (Prime–Gamma source decomposition)**.** *At the base point $x = 1$, set*

$$s(x) := \frac{1}{2} + \sqrt{x}.$$

The completed logarithmic derivative decomposes on the source side as

$$\frac{\xi'}{\xi}(s) = \frac{1}{s} + \frac{1}{s-1} - \frac{1}{2}\log\pi + \frac{1}{2}\frac{\Gamma'}{\Gamma}\left(\frac{s}{2}\right) + \frac{\zeta'}{\zeta}(s),$$

and for $\Re(s) > 1$,

$$\frac{\zeta'}{\zeta}(s) = -\sum_{m\geq 2} \frac{\Lambda(m)}{m^s}.$$

Thus every derivative appearing in $M_{n,k}$ is sourced by rational boundary terms, Gamma or polygamma terms, and prime-power von Mangoldt terms at $s = 3/2$.

Definition 9328.5 (Prime–Gamma triangular cancellation)**.** *The statement*

PrimeGammaTriadicCancellation

asserts that the decomposition of Definition 9328.4 induces rows

$$M^{\mathrm{rat}}_{n,k}, \qquad M^{\Gamma}_{n,k}, \qquad M^{\mathbb{P}}_{n,k}$$

with

$$M_{n,k} = M^{\mathrm{rat}}_{n,k} + M^{\Gamma}_{n,k} + M^{\mathbb{P}}_{n,k} \geq 0$$

for every finite pair (n, k). The assertion is not that each summand is separately nonnegative; it is that the source decomposition closes without producing a negative triangular moment row.

Principle 9328.6 (Prime–Gamma cancellation is the source-side form)**.** PrimeGammaTriadicCancellation *is the direct source-side positivity problem. It asks for a cancellation law among rational boundary, Gamma, and prime-power rows at the source-safe base point. It does not mention zero locations, and it does not use a limiting Hilbert space as an input.*

9328.3 Triadic Norm Decomposition

Definition 9328.7 (BEDC source norm formula)**.** *The statement*

BEDCSourceNormFormula

is the stronger certificate that each final moment difference has a source-built norm decomposition

$$M_{n,k} = \left\|\Psi^{\text{source}}_{n,k}\right\|^2_{\mathcal{H}_{n,k}}.$$

Equivalently, the vector may be displayed through the three BEDC-facing components

$$M_{n,k} = \|\Psi^{\mathsf{P}}_{n,k}\|^2 + \|\Psi^{\mathsf{U}}_{n,k}\|^2 + \|\Psi^{\mathsf{T}}_{n,k}\|^2,$$

where P *carries prime, Gamma, rational-boundary, and source ledger rows;* U *carries the positive inner-product row; and* T *records that* (n, k) *is a finite witness coordinate.*

Theorem 9328.8 (Source norm formula implies triangular cancellation)**.** *If* BEDCSourceNormFormula *holds and its source projection agrees with the decomposition of Definition 9328.4, then* PrimeGammaTriadicCancellat *holds.*

Proof. The source norm formula gives $M_{n,k}$ as a norm square, hence $M_{n,k} \geq 0$. Agreement with the prime–Gamma source decomposition identifies this same number with

$$M^{\text{rat}}_{n,k} + M^{\Gamma}_{n,k} + M^{\mathbb{P}}_{n,k}.$$

Therefore the displayed sum is nonnegative for every finite pair (n, k). □

Theorem 9328.9 (Final moment core is conditionally RH-sufficient)**.** *If either* BEDCSourceNormFormula *or* PrimeGammaTriadicCancellation *is established in a form that implies* XiHausdorffMomentPositivity*, and if the analytic determination and zero-exclusion transports of chapter 9326 are available, then the BEDC RH section target is inhabited.*

Proof. The source norm formula implies Hausdorff positivity by Definition 9328.3. The prime–Gamma cancellation route is assumed to imply the same finite inequality family. Once XiHausdorffMomentPositivity is available, Definition 9326.12 applies the analytic determination row and the Stieltjes-to-zero-exclusion transport to obtain the RH section target. □

9328.4 Obligation Boundary

Principle 9328.10 (Only the final moment core remains open in this route)**.** *Within the moment route, the unreduced proof load is the construction of one of the following source-side certificates:*

BEDCSourceNormFormula, PrimeGammaTriadicCancellation, XiHausdorffMomentPositivity.

The first is strongest, the second is the explicit source cancellation form, and the third is the finite Hausdorff inequality family. The positive measure, triangular finite tower, Herglotz function, Pick kernel, Hilbert factorization, and RH-facing zero exclusion are reduction surfaces once one of these source-side certificates is supplied.

Chapter 9329

Source-Positive Obligation Frontier

chapter 9328 isolates the finite moment core behind the RH route. This chapter records the remaining source-positive frontier as named proof obligations rather than as another list of equivalent RH criteria. The Li coordinate, the scale-phase split, the Hausdorff reduction, the finite Pick factorizations, and the triangular moment tower are already reduction surfaces in the preceding chapters. The unreduced content is the construction showing that the completed zeta source data generate the positive object those reductions require.

9329.1 Common Observation Coordinate

Definition 9329.1 (Source-positive observation coordinate)**.** *Fix a centered zeta coordinate and an active mode height*

$$s = \frac{1}{2} + r + it, \qquad h_a > 0.$$

The centered mode readout is

$$U_a(s) := e^{-(s-\frac{1}{2})h_a} = e^{-rh_a} e^{-ith_a}.$$

Its logarithmic orthogonal observation vector is

$$v_a(s) := (-rh_a, -th_a).$$

For a finite active packet A_m, set

$$W_m := \sum_{a \in A_m} h_a^2, \qquad E_m^{\text{scale}} := r^2 W_m.$$

The notation is the same scale-phase read used in Definition 9289.13 and Definition 9289.15; it is fixed here only to name the common projection coordinate for the source-positive frontier.

Definition 9329.2 (Triadic frontier roles)**.** *The source-positive frontier reads every unresolved RH-facing obligation through three roles:*

$$\mathsf{P} := \textit{prime, Gamma, rational-boundary, and ledger source rows},$$

$$\mathsf{U} := \textit{unit-boundary, positive-kernel, or positive-measure row},$$

and

$$\mathsf{T} := \textit{finite witness, finite tower, or analytic continuation row.}$$

The role split is a bookkeeping constraint. It does not assert positivity; it identifies which row must carry the source, which row must carry nonnegativity, and which row must make failures finitely visible or transport equality through the analytic domain.

9329.2 Legal Source and Pick Factorization

Definition 9329.3 (ZetaIsBEDCLegalObject)**.** *The obligation*

$$\mathsf{ZetaIsBEDCLegalObject}$$

asserts that every completed zeta observation relevant to a non-trivial zero is represented by a BEDC triadic generator. The P *row displays the source ledger: prime-power frequencies, Gamma normalization, functional-equation normalization, continuation rows, and the prime-register accounting behind logarithmic heights. The* U *row displays the admissible unit-boundary or positive-kernel closure; a positive real radial scale factor may not be hidden in the solenoid prime fiber. The* T *row displays the finite witness tower that connects zero packets, moment packets, and Pick packets without posterior repair.*

Principle 9329.4 (Legal zeta object refines the embedding ledger)**.** $\mathsf{ZetaIsBEDCLegalObject}$ *is the source-positive strengthening of the embedding obligations of Definition 9324.1, Definition 9290.19, and Definition 9289.20. Those obligations name the carrier and projection rows. The legal-object frontier additionally requires that the positive witness used by the RH route be generated from the same source ledger rather than attached after the zero packet has been inspected.*

Definition 9329.5 (Triangular legality rows for completed xi)**.** *A triangular legality row for a completed xi zero packet with centered coordinate*

$$\rho = \frac{1}{2} + r + it$$

is the part of a $\mathsf{ZetaIsBEDCLegalObject}$ *witness that supplies the following data. Each active mode* a *has a source-ledger height* $h_a > 0$ *coming from prime, Gamma, normalization, or continuation rows, and its centered channel is*

$$U_a(\rho) = e^{-(\rho-\frac{1}{2})h_a} = e^{-rh_a}e^{-ith_a}.$$

The solenoid gauge row is phase-valued:

$$0 \to K_S \to \Sigma_S \to \mathbb{T} \to 0, \qquad K_S \cong \prod_{p\in S} \mathbb{Z}_p,$$

so a factor in $\mathsf{Real}^{\uparrow}_{>0}$ *is not a hidden solenoid fibre. The orthogonal observation row is*

$$v_a(\rho) := (-rh_a, -th_a), \qquad |v_a(\rho)|^2 = r^2h_a^2 + t^2h_a^2.$$

For finite active packets A_m*, set*

$$W_m := \sum_{a\in A_m} h_a^2, \qquad E_m^{\mathrm{scale}}(\rho) := r^2 W_m.$$

The cofinal triangular closure row requires

$$W_m \to \infty, \qquad E_m^{\mathrm{scale}}(\rho) \le B_\rho$$

for a bound B_ρ *attached to the zero packet. The bound is a ledgered unit-positive or Pick/Hausdorff stability row; it is not a positive real scale register.*

Theorem 9329.6 (Legal triangular xi construction is conditionally RH-sufficient)**.** *Assume every completed non-trivial xi zero packet carries a* ZetaIsBEDCLegalObject *witness whose triangular legality rows are as in Definition 9329.5. Then the BEDC RH section target is inhabited.*

Proof. Let a completed non-trivial zero packet have centered coordinate

$$\rho = \frac{1}{2} + r + it.$$

The triangular legality rows give

$$E_m^{\mathrm{scale}}(\rho) = r^2 W_m, \qquad W_m \to \infty, \qquad E_m^{\mathrm{scale}}(\rho) \le B_\rho.$$

If $r \# 0$, then $r^2 > 0$, so the first two rows imply

$$E_m^{\mathrm{scale}}(\rho) \to \infty.$$

This contradicts the displayed bound row. Hence the centered radial coordinate has no apartness from zero, and the located real readback in the zero packet gives $r = 0$. Therefore

$$\mathrm{Re}(\rho) = \frac{1}{2}.$$

Since the zero packet was arbitrary, the corresponding section sends every completed non-trivial zero packet to its fixed-half witness. □

Principle 9329.7 (Legal construction is stronger than analytic description)**.** *The theorem does not assert that any formal description of ξ is enough. The construction must retain the source ledger, preserve the triangular square-norm invariant, route visible reads through $\mathbb{T}$ and the compact prime-register fibre, and supply the bounded unit-positive row that prevents E_m^{scale} from becoming an unledgered* $\mathsf{Real}^{\uparrow}_{>0}$ *coordinate. Thus the remaining analytic burden is to construct* ZetaIsBEDCLegalObject*, for example through a source-built Pick factorization or through the xi Hausdorff moment positivity route.*

Definition 9329.8 (SourcePickHilbertFactorization)**.** *For the completed zeta Pick target of Definition 9325.11, the obligation*

SourcePickHilbertFactorization

asserts the existence of a Hilbert row $\mathcal{H}$ and a feature map $\Phi : \mathbb{D} \to \mathcal{H}$ such that

$$K_\xi(z, w) = \langle \Phi(z), \Phi(w) \rangle_{\mathcal{H}}$$

for all displayed disk coordinates z, w. The carrier $\mathcal{H}$ and the map Φ must be constructed from prime, Gamma, rational-boundary, normalization, and continuation source rows. They may not be defined from the posterior zero set of ξ.

Principle 9329.9 (Pick factorization is an aggregate source balance)**.** *The Pick factorization obligation is not a demand that the prime, Gamma, and rational-boundary summands be separately positive kernels. The prime component enters through the sign of ζ'/ζ, while the Gamma and boundary rows carry archimedean and completion data. The frontier requires an aggregate Gram representation for the completed kernel, with the negative and positive source contributions balanced in one source-built inner-product row.*

Definition 9329.10 (PZGSourceCompletenessForZeta)**.** *The obligation*

PZGSourceCompletenessForZeta

asserts that every source-side Dirichlet mode used by the completed zeta carrier is read through the Prime–Zeckendorf–Goedel filtration of Definition 9291.8. In the source-safe half-plane, this means that each mode has the displayed form

$$n^{-s} = n^{-1/2} \exp\left(-(r+it)\sum_{p,k} z_{p,k}(n) F_k \log p\right)$$

for

$$s = \frac{1}{2} + r + it.$$

The pointwise identity follows from the single-mode bridge of Definition 9291.19. The obligation is the uniform source certificate: the same PZG ledger must remain attached to Euler products, logarithmic derivatives, Gamma completion, and the positive moment or Pick readouts used later in the route.

Principle 9329.11 (Analytic continuation is not source-completeness by itself)**.** *Analytic continuation preserves a meromorphic function after the local data and domain are fixed, but it does not by itself prove that the PZG source ledger, positive-kernel row, and finite witness tower have been transported. The continuation row needed here is triadic: it must retain the source ledger, preserve the positive row supplied by the Pick or Hausdorff mechanism, and expose branch and pole boundaries in finite witness data.*

Theorem 9329.12 (Source Pick factorization supplies the positive witness)**.** *If* SourcePickHilbertFactorization *holds, then the completed zeta Pick kernel is positive definite by a source-side certificate.*

Proof. For any finite disk rows $z_1, \ldots, z_N$ and coefficients $c_1, \ldots, c_N$, the factorization gives

$$\sum_{i,j=1}^{N} c_i \overline{c_j} K_\xi(z_i, z_j) = \left\| \sum_{i=1}^{N} c_i \Phi(z_i) \right\|_{\mathcal{H}}^2 \geq 0.$$

The source condition on $\mathcal{H}$ and Φ records that this positive witness belongs to the zeta source ledger rather than to a zero-location oracle. □

9329.3 Moment Positivity Frontier

Definition 9329.13 (Normalized xi moment frontier)**.** *All moment rows in this frontier use the normalization of Definition 9326.2:*

$$S_\xi(x) := 2\frac{G'(x)}{G(x)}.$$

The moment sequence is

$$a_n := \frac{(-1)^n}{n!} \frac{d^n}{dx^n} S_\xi(x)\Big|_{x=1},$$

and the finite moment-difference row is

$$M_{n,k} := (-1)^k \Delta^k a_n.$$

Thus $x = 1$, equivalently $s = 3/2$, is the source-safe base point for the moment audit.

Principle 9329.14 (PrimeGammaTriadicCancellation is the explicit source inequality)**.** PrimeGammaTriadicCance *defined in Definition 9328.5, is the explicit source-side inequality*

$$M^{\mathrm{rat}}_{n,k} + M^{\Gamma}_{n,k} + M^{\mathbb{P}}_{n,k} \geq 0 \qquad (n, k \geq 0).$$

Its P *row is the von Mangoldt, Gamma, and rational-boundary ledger of Definition 9328.4. Its* U *row is the nonnegative aggregate result. Its* T *row is the finite index* (n, k)*, because any failure is a displayed negative entry in the Hausdorff difference triangle.*

Principle 9329.15 (BEDCSourceNormFormula is the square-norm form)**.** BEDCSourceNormFormula*, defined in Definition 9328.7, is the stronger certificate that the same finite moment difference is a source-built norm square:*

$$M_{n,k} = \|\Psi^{\mathrm{source}}_{n,k}\|^2_{\mathcal{H}_{n,k}}.$$

In triadic form, $\Psi^{\mathrm{source}}_{n,k}$ *displays the source ledger row, the unit-positive inner-product row, and the finite difference witness row. This formula implies nonnegativity immediately and explains the cancellation demanded by* PrimeGammaTriadicCancellation*.*

Principle 9329.16 (XiHausdorffMomentPositivity is the finite audit form)**.** XiHausdorffMomentPositivity*, defined in Definition 9326.6, is the finite audit form of source positivity:*

$$(-1)^k \Delta^k a_n \geq 0 \qquad (n, k \geq 0).$$

It is weaker as an explanation than a source norm formula, but it is the exact Hausdorff condition needed to construct a positive measure on $[0, 1]$ *through the route of Definition 9326.8.*

9329.4 Analytic Identification and Finite Tower

Definition 9329.17 (HausdorffToXiAnalyticContinuation)**.** *Assume the Hausdorff moment row produces a positive measure* μ *on* $[0, 1]$ *and the transform*

$$\widetilde{S}_\xi(x) := \int_0^1 \frac{1}{1 + (x-1)y}\, d\mu(y)$$

as in Definition 9326.9. The obligation

HausdorffToXiAnalyticContinuation

asserts that

$$\widetilde{S}_\xi(x) = S_\xi(x)$$

on the displayed analytic domain of the completed zeta carrier. The row must expose the local Taylor agreement at $x = 1$*, the connected analytic domain used for continuation, the branch choice for* $\sqrt{x}$*, and the pole-exclusion information needed to apply uniqueness of analytic continuation without hiding an off-line zero behind a branch or pole boundary.*

Definition 9329.18 (BEDCFiniteMeasureTowerConvergence)**.** *For each positive triangular level* $N \geq 1$*, let*

$$B_{N,j}(y) := \binom{N}{j} y^j (1-y)^{N-j}$$

and form the finite Bernstein atomic measure

$$\mu_N := \sum_{j=0}^{N} L_N(B_{N,j})\delta_{j/N}$$

from the finite moment functional of Definition 9327.4. Define

$$S_N(x) := \int_0^1 \frac{1}{1+(x-1)y}\, d\mu_N(y).$$

The obligation

BEDCFiniteMeasureTowerConvergence

asserts local convergence

$$S_N \to S_\xi$$

on the same displayed analytic domain, with convergence carried by the finite tower ledger rather than by an uninspected limiting measure.

Principle 9329.19 (Tower convergence is the finite constructive form)**.** BEDCFiniteMeasureTowerConvergence *is the finite constructive form of the positive spectral measure. The* P *row is the finite moment and Bernstein ledger, the* U *row is the finite positive atomic measure and its Pick kernel, and the* T *row is local convergence of the tower. It is compatible with Definition 9327.14, whose convergence row identifies the limit with* S_ξ*.*

9329.5 Compressed Frontier Package

Definition 9329.20 (Source-positive unresolved package)**.** *The source-positive unresolved package is the finite list of named proof obligations*

ZetaIsBEDCLegalObject,
PZGSourceCompletenessForZeta,
SourcePickHilbertFactorization,
PrimeGammaTriadicCancellation,
BEDCSourceNormFormula,
XiHausdorffMomentPositivity,
HausdorffToXiAnalyticContinuation,
BEDCFiniteMeasureTowerConvergence.

The list is ordered by proof role rather than by logical independence: the legal-object row admits the completed zeta carrier, the source Pick row gives the kernel-positive target directly, the moment rows give a finite-audit route to the same positivity, and the analytic and tower rows identify the finite or moment constructions with the original completed zeta function.

Principle 9329.21 (Minimal source-positive core)**.** *Inside the moment route, the compressed mathematical core is*

PrimeGammaTriadicCancellation, BEDCSourceNormFormula, HausdorffToXiAnalyticContinuation.

The norm formula is a stronger explanatory form of the cancellation inequality; either route must supply XiHausdorffMomentPositivity*. The analytic continuation row then identifies the Hausdorff transform with the original normalized* S_ξ *rather than with a separate function sharing only Taylor coefficients at the source-safe base point.*

Theorem 9329.22 (Source-positive frontier is conditionally RH-sufficient)**.** *Assume* ZetaIsBEDCLegalObject *and the standard zero-exclusion transport for the completed zeta carrier. Assume either* SourcePickHilbertFactorizati*or the moment route consisting of* XiHausdorffMomentPositivity*, the Hausdorff moment theorem for the xi moment row, and* HausdorffToXiAnalyticContinuation*. Then the BEDC RH section target is inhabited.*

Proof. If the source Pick factorization holds, then Definition 9329.12 gives the positive completed Pick kernel from a source-side certificate. The legal-object row supplies the zero-ledger and no-hidden-radial transport needed to apply the logarithmic-derivative zero-exclusion route of Definition 9289.25.

If instead the moment route is used, Hausdorff positivity supplies a positive measure by Definition 9326.8. The analytic-continuation row identifies its transform with S_ξ on the displayed analytic domain. Then Definition 9326.12 applies the Stieltjes-to-zero-exclusion transport to obtain the RH section target. In both cases, ZetaIsBEDCLegalObject prevents the positive witness from being detached from the completed zeta source ledger. □

Principle 9329.23 (Status of the source-positive frontier)**.** *This frontier contains proof obligations only. It does not reassert the classical equivalences and reductions already recorded in the RH visions. The constructive content still to be supplied is that the prime, Gamma, rational-boundary, normalization, and continuation source data of completed zeta generate a positive moment measure or a source-built Pick factorization whose analytic continuation is the original completed zeta readout.*

Chapter 9330

Zeta BEDC Closure Strength Gate

chapter 9329 names ZetaIsBEDCLegalObject as the source-positive entry condition for the completed zeta carrier, and chapter 9331 records the radial-register package that is conditionally sufficient for the RH section target. The present chapter separates ordinary encodability from the stronger generated-closure condition needed by that package.

9330.1 Closure Strength

Definition 9330.1 (Zeta encodability row)**.** *The zeta encodability row is the statement that every Dirichlet monomial can be written through prime and PZG data:*

$$n = \prod_p p^{v_p(n)} = \prod_{p,k} (p^{F_k})^{z_{p,k}(n)}$$

and therefore

$$n^{-s} = \exp\left(-s \sum_{p,k} z_{p,k}(n) F_k \log p\right).$$

This row is a syntax and readout statement. It does not by itself supply analytic continuation, zero packets, positive kernels, limit ledgers, or closure under the circle–triangle–register invariant.

Definition 9330.2 (Zeta BEDC closure)**.** *The obligation*

$$\mathsf{ZetaBEDCClosure}$$

asserts that the completed zeta route enters BEDC as a generated closure object, not merely as an encodable formula. It consists of the following certificate rows:

$$\zeta^{\uparrow}, \quad \xi^{\uparrow}, \quad \mathsf{NTZeroType}, \quad \mathsf{ZetaZero}, \quad \mathsf{InCritStrip},$$

functional-equation transport, *analytic-continuation transport*, *explicit-formula trans*

Li coordinate readout, *zero-observation readout*, *analytic prime-shift constraint*, *prime-shift analytic compat*

Each displayed row must be admitted through a naming or operation certificate with source specification, pattern specification, classifier specification, stability certificate, and ledger policy. In particular, analytic continuation and limiting interfaces must carry the ledger rows that record branch, pole, gap, and tail data.

Definition 9330.3 (Triadic invariant preservation for zeta)**.** *Triadic invariant preservation for zeta is the assertion that every constructor used by Definition 9330.2 preserves the circle–triangle–register invariant*

$$\mathcal{I}_{\triangle} = \mathcal{I}_{\mathsf{P}} \wedge \mathcal{I}_{\mathsf{U}} \wedge \mathcal{I}_{\mathsf{T}}.$$

Here P *records prime, Gamma, rational-boundary, PZG, and source-ledger rows;* U *records unit-boundary, positive-kernel, or positive-measure rows with no hidden* $\mathsf{Real}^{\uparrow}_{>0}$ *gauge; and* T *records append-only histories, finite witnesses, analytic transport, and cofinal tower ledgers. A constructor may expose a residual only by recording it in one of these rows.*

Principle 9330.4 (Analytic prime shifts are part of zeta closure)**.** *The prime-index motion inside Definition 9330.2 must satisfy* SmoothPrimeShiftConstraint *from Definition 9296.11. Thus the PZG registers may supply discrete source uniqueness, but the completed zeta object is licensed only when those discrete shifts are compatible with holomorphic charts, compact-uniform height control, branch ledgers, Rouché no-wall-crossing, and cofinal tower coverage.*

Principle 9330.5 (Zeta continuation is prime-shift compatible)**.** *The analytic-continuation part of Definition 9330.2 is the stronger obligation* ZetaAnalyticContinuationIsPrimeShiftCompatible *from Definition 9299.4. It requires nested cell compatibility, cocycle compatibility, character-flow compatibility, normal convergence, and no radial anomaly for the continuation rows used by completed zeta.*

Definition 9330.6 (Legal zero closure object)**.** *A legal zero closure object is a completed non-trivial zero packet* ρ *admitted by Definition 9330.2 whose zero observation, Li coordinate, and closure transport are internal rows of the same generated object. Thus*

$$z_\rho := 1 - \frac{1}{\rho}$$

is not an external diagnostic attached after the zero has been selected; it is a displayed readout of the zero packet. Writing

$$z_\rho = e^{\ell + i\theta},$$

the radial row ℓ *must be either absent, ledgered as a scale row, or refused by the legal closure classifier. It cannot be hidden as unit phase, compact prime-register fibre, or ledger-free tail.*

9330.2 Conditional Consequence

Theorem 9330.7 (Strong zeta closure reduces RH to the radial obstruction package)**.** *Assume*

ZetaBEDCClosure, ZetaAnalyticContinuationIsPrimeShiftCompatible, SmoothPrimeShiftConstraint, *Defi*

and assume that every completed non-trivial zero packet is a legal zero closure object in the sense of Definition 9330.6. Then the embedding row needed by Definition 9331.25 is supplied. Consequently, together with

NoHiddenRadialRegister, CofinalTriangularTower,

and the support rows of Definition 9331.5, the BEDC RH section target is inhabited.

Proof. The closure row supplies the completed zeta carrier, completed zero packets, analytic transports, Li coordinate, and limit ledgers as one generated BEDC object. The prime-shift compatibility row supplies the nested cell, cocycle, character-flow, normal-convergence, and no-radial-anomaly

data for the analytic transports. By Definition 9299.5, the smooth prime-shift row follows as the corresponding tower-level constraint: the prime-index motion is not an arbitrary discrete address stream, but an analytic tower with chart compatibility, branch ledger, height control, and no-wall-crossing rows. The triadic preservation row prevents the constructors from changing the circle–triangle–register invariant while passing through Euler products, completion, continuation, zero readout, and positive-kernel or moment readout. The legal zero closure row supplies, for each completed non-trivial zero packet, the internal Li coordinate

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

Thus the required ZetaBEDCEmbedding row of Definition 9331.25 is present. Applying that theorem with NoHiddenRadialRegister, CofinalTriangularTower, and the stated support rows gives the BEDC RH section target. □

Principle 9330.8 (Encodability is not closure)**.** *The implication used in this route is not*

$$\textit{zeta is encodable} \Longrightarrow \mathsf{RH}.$$

The needed implication is the stronger closure statement

$$\mathsf{ZetaBEDCClosure} \wedge \mathsf{ZetaAnalyticContinuationIsPrimeShiftCompatible} \wedge \mathsf{SmoothPrimeShiftConstraint} \wedge \textit{Definition 9330.}$$

Only after this gate has been crossed can NoHiddenRadialRegister *and* CofinalTriangularTower *force the Li radial coordinate to vanish. The remaining analytic work is therefore not another notation for zeta, but the construction of the legal generated closure object.*

Chapter 9331

Radial Register Obstruction Frontier

chapter 9324 records the projection-facing proof obligations, and chapter 9329 records the source-positive moment and Pick frontier. The present chapter isolates the radial-register obstruction inside that package. The aim is narrow: state only the unresolved objects needed by the circle–triangle–register route, and discharge the part that is independent of zeta analysis. The discharged part is the compact-gauge obstruction: a compact phase or compact prime-register gauge cannot carry a nontrivial positive-real scale through a continuous group homomorphism.

9331.1 Unresolved Circle–Triangle–Register Package

Definition 9331.1 (TriadicGenerationInduction)**.** *The obligation*

$$\mathsf{TriadicGenerationInduction}$$

is the finite structural induction principle for the circle–triangle–register grammar. Its base rows are finite distinctions, unit phase, and append-only histories. Its constructors are unit-circle phase readout, orthogonal triangular decomposition, and finite residual registration. Its conclusion is that every certified object generated by this grammar carries the triadic invariant

$$\mathcal{I}_{\triangle} = \mathcal{I}_{\mathsf{P}} \wedge \mathcal{I}_{\mathsf{U}} \wedge \mathcal{I}_{\mathsf{T}}.$$

The grammar-specific induction theorem is Definition 9413.364, and its abstract foundation is Definition 9413.366. The open part is not the induction rule, but the coverage of zeta by the constructor grammar.

Definition 9331.2 (ZetaBEDCEmbedding)**.** *The obligation*

$$\mathsf{ZetaBEDCEmbedding}$$

asserts that the completed zeta carrier, completed zero packets, Li coordinates, Weil probes, Nyman containers, explicit-formula rows, functional-equation transport, and source-positive Pick or moment readouts enter the circle–triangle–register grammar as one certified observation object. This is the embedding form of Definition 9324.1 and the zeta constructor form of Definition 9413.373.

Definition 9331.3 (NoHiddenRadialRegister)**.** *The obligation*

$$\mathsf{NoHiddenRadialRegister}$$

asserts that a nontrivial radial factor

$$e^{\ell}, \qquad \ell \# 0,$$

cannot be represented as unit phase, compact prime-register gauge, finite Zeckendorf–Goedel residue erased without ledger, or a harmless tail of a cofinal tower. If it is recorded, the object is scale-ledgered; if it is unrecorded, the object is not legal. In neither case is it a pure unit-phase closure.

Definition 9331.4 (ContinuousDiscreteBridge)**.** *The obligation*

$$\mathsf{ContinuousDiscreteBridge}$$

asserts that the continuous zeta phase and the discrete prime register are two projections of the same source row:

$$e^{-it\log n} \quad \leftrightarrow \quad (v_p(n))_p.$$

The single-mode identity is Definition 9291.19. The open content is global: the bridge must remain lossless, cofinal, and auditable inside the completed zero-relevant packet rather than only for one Dirichlet monomial.

Definition 9331.5 (Projection and tower support obligations)**.** *The support obligations for the radial-register route are:*

$$\mathsf{TriProjectionFaithfulness},$$
$$\mathsf{DominantObservationStrand},$$
$$\mathsf{UnitBoundaryTransport},$$
$$\mathsf{CofinalTriangularTower},$$
$$\mathsf{NoOracleDiagonalSafety},$$
$$\mathsf{FiniteTowerAmplification}.$$

They state respectively that Li, Weil, and Nyman read the same radial leg; that the zero-relevant strand is source-selected; that the strand transports to a unit-boundary container without boundary crossing; that the triangular tower exhausts residuals; that no posterior oracle repairs negative directions; and that one finite positive mechanism grows cofinally across the zeta observation surface.

9331.2 Zeta Source Construction Boundary

Definition 9331.6 (ConstructorTriadicPreservation for xi)**.** *The obligation*

$$\mathsf{ConstructorTriadicPreservation}_{\xi}$$

asserts that every constructor used to build the completed zeta carrier and its positive-kernel readout is triadic-preserving. Its constructor surface includes finite sums, Dirichlet series limits, Euler products, Gamma factors, completed normalization, analytic continuation, logarithmic derivatives, Cayley transforms, Pick kernels, and limiting positive-kernel towers. Each constructor must preserve source ledger, unit-boundary or positivity, and generated witness data in the sense of Definition 9413.365.

Definition 9331.7 (ZetaSourceConstruction)**.** *The obligation*

$$\mathsf{ZetaSourceConstruction}$$

asserts that

$$\xi(s) = \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s)$$

is generated from source-side rows rather than from zero-location data. The prime-power and von Mangoldt rows supply the P *component; the Gamma, power of* π*, and completed boundary rows supply the archimedean* U *component; and the continuation from the source-safe half-plane supplies the* T *component. The open part is not the displayed formula, but the certificate that analytic continuation and completion preserve the source-side positivity needed by the Pick or moment route.*

Theorem 9331.8 (Direct sums preserve source Pick positivity)**.** *Let* K_X *and* K_Y *be positive kernels with source-built Hilbert factorizations*

$$K_X(z,w) = \langle \Phi_X(z), \Phi_X(w)\rangle_{\mathcal{H}_X}, \qquad K_Y(z,w) = \langle \Phi_Y(z), \Phi_Y(w)\rangle_{\mathcal{H}_Y}.$$

Then

$$K_X(z,w) + K_Y(z,w)$$

has a source-built Hilbert factorization on $\mathcal{H}_X \oplus \mathcal{H}_Y$*.*

Proof. Define

$$\Phi(z) := \Phi_X(z) \oplus \Phi_Y(z).$$

Then

$$\langle \Phi(z), \Phi(w)\rangle_{\mathcal{H}_X\oplus\mathcal{H}_Y} = \langle \Phi_X(z), \Phi_X(w)\rangle_{\mathcal{H}_X} + \langle \Phi_Y(z), \Phi_Y(w)\rangle_{\mathcal{H}_Y} = K_X(z,w) + K_Y(z,w).$$

The feature map is source-built from the two source-built feature maps by the displayed direct-sum constructor. □

Theorem 9331.9 (Finite-window limits preserve positive kernels)**.** *Let* (K_m) *be positive kernels on a domain* D *and suppose*

$$K_m(z,w) \to K(z,w)$$

for every displayed finite pair $z, w \in D$*. Then* K *is positive definite.*

Proof. Let $z_1, \ldots, z_N \in D$ and $c_1, \ldots, c_N \in \mathbb{C}$ be displayed. For every m,

$$\sum_{i,j=1}^{N} c_i\overline{c_j}K_m(z_i, z_j) \geq 0$$

because K_m is positive definite. Taking the finite sum through the pointwise limit gives

$$\sum_{i,j=1}^{N} c_i\overline{c_j}K(z_i, z_j) \geq 0.$$

Since the finite window was arbitrary, K is positive definite. □

Principle 9331.10 (Analytic continuation is the remaining constructor risk)**.** *The direct-sum and finite-window limit constructors above are triadic-preserving once their source rows are displayed. The remaining constructor risk in* $\mathsf{ConstructorTriadicPreservation}_\xi$ *is the combined Euler-product, Gamma-completion, and analytic-continuation step: it must preserve source-side Pick, Herglotz, or Hausdorff positivity, not merely equality of meromorphic functions.*

9331.3 Single-Mode Source Bridge

Definition 9331.11 (Prime exponent ledger for a zeta monomial)**.** *For a positive integer n, the prime exponent ledger is the finite support function*

$$\upsilon(n) := (v_p(n))_p, \qquad n = \prod_p p^{v_p(n)}.$$

Its logarithmic height is

$$h_n := \log n = \sum_p v_p(n) \log p.$$

Theorem 9331.12 (A zeta monomial has a prime-ledger phase readout)**.** *For every displayed positive integer n and real phase time t,*

$$n^{-it} = e^{-it \log n} = \exp\left(-it \sum_p v_p(n) \log p\right).$$

Proof. The first equality is the definition of the exponential logarithmic readout for the positive real n. The second equality substitutes the prime exponent ledger identity

$$\log n = \sum_p v_p(n) \log p$$

from Definition 9331.11. Since the ledger has finite support, the displayed sum is finite. □

Principle 9331.13 (Status of the single-mode bridge)**.** *Definition 9331.12 closes only the single-mode bridge. It proves that one Dirichlet monomial has continuous unit-phase readout and discrete prime-register provenance. It does not prove the global* ContinuousDiscreteBridge *obligation for completed zeta zero packets, where functional-equation transport, analytic continuation, explicit-formula rows, and cofinal tower coverage must share the same source certificate.*

9331.4 Invariant Separation

Definition 9331.14 (Expressible but not generated)**.** *An object is* expressible but not generated *relative to a BEDC grammar when its external notation is well formed, but no derivation from the declared initial packets and constructors reaches a certificate for that object. The distinction is semantic for this chapter: a complex coordinate, a formal zero packet, or a radial parameter may be written in ambient notation without being admitted by the circle–triangle–register construction grammar.*

Definition 9331.15 (Hidden scale energy)**.** *For a proposed pure phase closure packet, let*

$$\ell_1, \ldots, \ell_N$$

be the displayed radial scale legs not recorded as scale-ledgered rows. Its hidden scale energy is

$$\mathcal{E}_{\text{scale}} := \sum_{i=1}^{N} \ell_i^2.$$

For an iterated Li observation, the corresponding finite observation energy at level n is

$$\mathcal{E}_{\text{scale}}(n) := \sum_{i=1}^{N} (e^{n\ell_i} - 1)^2.$$

A pure phase closure is scale-clean when

$$\mathcal{E}_{\text{scale}} = 0.$$

Principle 9331.16 (Scale energy is not cancelled by mirror signs)**.** *The obstruction is quadratic rather than additive. Opposite radial parameters may have zero algebraic sum,*

$$\ell + (-\ell) = 0,$$

but their hidden scale energy is

$$\ell^2 + (-\ell)^2 = 2\ell^2.$$

Thus a radial leg cannot be erased by pairing it with an opposite radial leg inside a pure phase closure. It must either be absent, or be recorded as explicit scale-ledger data.

Theorem 9331.17 (Scale-clean invariant for generated pure closures)**.** *Every pure closure generated by the circle–triangle–register grammar and satisfying the no-hidden-scale ledger rule has*

$$\mathcal{E}_{\text{scale}} = 0.$$

Proof. Proceed by structural induction on the generating certificate. The initial distinction split has only append-only history and orthogonal residual rows. Any residual used by the packet is recorded in the ledger, so it is not an unrecorded radial scale leg. A circle constructor adds a unit phase $e^{i\theta}$ and hence has radial coordinate $\log |e^{i\theta}| = 0$. A triangle-readout constructor decomposes a readout into an existing projection and an orthogonal residual; if that residual is used, the register rule records it, and if it is not recorded the packet is not legal. The register constructor converts a displayed residual into ledger data and therefore does not create hidden scale. Hence no generated pure closure contains an unrecorded nonzero radial leg. The hidden scale energy is a sum of squares over an empty or zero family, so it is 0. □

Theorem 9331.18 (Off-line Li coordinates are not generated pure closures)**.** *Let a completed zero candidate have Li coordinate*

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

If $\ell \# 0$, then the candidate cannot be a generated pure circle–triangle–register closure satisfying the no-hidden-scale ledger rule.

Proof. For any displayed positive observation level n, the Li defect decomposition of Definition 9290.17 contains the radial square

$$(e^{n\ell} - 1)^2.$$

When $\ell \# 0$, this is a nonzero scale leg for some displayed observation. If it is recorded, the packet is scale-ledgered and not a pure phase closure by Definition 9331.23. If it is not recorded, it contributes positive hidden scale energy, contradicting Definition 9331.17. Hence the candidate is expressible in ambient notation but not generated as a legal pure closure by the stated grammar. □

9331.5 Compact Gauge Obstruction

Definition 9331.19 (Compact legal gauge)**.** *A legal gauge row K is* compact *when it is a compact topological group and all gauge readouts used by a pure phase closure are continuous group homomorphisms from K to the displayed target group. Unit phase containers and finite-prime solenoid fibres are read as compact legal gauges at the scope where their compactness row is displayed.*

Theorem 9331.20 (Compact gauges have no nontrivial radial character)**.** *Let K be a compact legal gauge and let*

$$\phi : K \to \mathbb{R}_{>0}$$

be a continuous group homomorphism, with $\mathbb{R}_{>0}$ under multiplication. Then

$$\phi(K) = \{1\}.$$

Proof. The image $H := \phi(K)$ is compact because K is compact and ϕ is continuous. It is a subgroup of $\mathbb{R}_{>0}$ because ϕ is a group homomorphism. Apply the logarithm isomorphism

$$\log : \mathbb{R}_{>0} \to \mathbb{R}.$$

Then $\log H$ is a compact subgroup of the additive group $\mathbb{R}$. If $x \in \log H$ and $x\#0$, then all integer multiples

$$nx \qquad (n \in \mathbb{Z})$$

lie in $\log H$, because $\log H$ is a subgroup. This set is unbounded in $\mathbb{R}$, contradicting compactness. Hence every element of $\log H$ is 0, so $\log H = \{0\}$ and therefore $H = \{1\}$. □

Corollary 9331.21 (Compact phase and prime gauges cannot hide scale)**.** *If the legal hidden row of a pure zeta phase closure is compact, then no nontrivial radial factor e^{ℓ} with $\ell\#0$ can be supplied by that hidden row.*

Proof. A hidden compact gauge would supply the radial factor through a continuous character

$$\phi : K \to \mathbb{R}_{>0}.$$

By Definition 9331.20, the image of ϕ is $\{1\}$. Thus every radial factor obtained from the compact gauge is trivial. □

9331.6 Ledgered Scale Is Not Pure Closure

Definition 9331.22 (Scale-ledgered closure)**.** *A zeta observation packet is* scale-ledgered *when its ledger contains a nontrivial radial row*

$$\ell\#0$$

or the corresponding positive factor e^{ℓ}. A scale-ledgered packet may be a legal object with an explicit residual, but it is not a pure unit-phase closure.

Theorem 9331.23 (Recorded radial scale changes the closure type)**.** *If a zeta observation packet records a radial row $\ell\#0$, then it is not a pure unit-phase closure.*

Proof. By Definition 9331.22, recording $\ell\#0$ makes the packet scale-ledgered. A pure unit-phase closure has only unit phase, orthogonal unit readout, and legal compact or finite ledgers as hidden rows; it has no nontrivial positive-real radial row. Therefore a packet with recorded radial scale has changed closure type. □

Principle 9331.24 (Two ways a radial factor is refused)**.** *The radial obstruction has two independent parts. If the radial factor is sent through compact phase or compact prime gauge, it is trivial by Definition 9331.20. If it is recorded as ledger data, it is no longer pure phase closure by Definition 9331.23. The remaining open content of* NoHiddenRadialRegister *is the global classification that every legal zeta gauge is either compact in this sense or ledger-explicit in this sense.*

9331.7 Conditional Core

Theorem 9331.25 (Radial obstruction package is conditionally RH-sufficient)**.** *Assume*

ZetaBEDCEmbedding, NoHiddenRadialRegister, CofinalTriangularTower,

together with the support rows of Definition 9331.5. Then the BEDC RH section target is inhabited.

Proof. The embedding row supplies, for each completed non-trivial zero packet ρ, a circle–triangle–register observation with Li coordinate

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell + i\theta}.$$

If $\ell \# 0$, the radial leg is nontrivial. The no-hidden-radial-register row says that this leg is neither unit phase nor a legal hidden prime-register gauge and cannot disappear as unledgered finite data. The projection faithfulness support row exposes the same radial leg through the Li, Weil, and Nyman readouts; the cofinal triangular tower prevents it from being postponed to the infinite tail; diagonal safety prevents posterior repair by a zero-dependent oracle. Thus a legal pure zeta closure cannot have $\ell \# 0$. The located readback gives $\ell = 0$. By Definition 9290.16,

$$\left|1 - \frac{1}{\rho}\right| = 1 \Longleftrightarrow \Re(\rho) = \frac{1}{2}.$$

Since the zero packet was arbitrary, the RH section target is inhabited. □

Principle 9331.26 (Status of the radial obstruction frontier)**.** *The compact-gauge obstruction and the single-mode prime-ledger bridge are closed local rows. They do not close the zeta embedding, projection faithfulness, or cofinal tower obligations. The remaining core is the global theorem that every zero-relevant zeta gauge is compact or ledger-explicit, and that the finite triangular tower covers the completed zero ledger without an oracle-dependent repair step.*

Chapter 9332

Bulk–Boundary Prime-Shadow Interface for RH

chapter 9254 identifies the RH statement as a section over the zeta-zero source carrier, while chapter 9246 isolates the carrier-level incompatibility still required by the prime-window route. The present chapter records a typed interface between these two readings. It treats prime defects as boundary-visible shadows of a single constrained body and states the exact hypotheses under which that interface yields the BEDC-RH witness target. The chapter supplies only a conditional assembly schema; the analytic content remains the displayed incompatibility row.

9332.1 Bulk and Boundary Data

Definition 9332.1 (Omega boundary readout diagram)**.** *An Omega boundary readout diagram is a tuple*

$$\mathcal{D}_\Omega := (I, (R_i, \sim_i, C_i, \alpha_i, \rho_{ji}, \Lambda_i)_{i\in I})$$

where I is a directed preorder, R_i is the carrier of the i-th readout layer, $\sim_i$ is its classifier, C_i is its legality predicate, $\alpha_i : R_i \to R_i$ is its self-similar operation, Λ_i is its ledger, and $\rho_{ji} : R_j \to R_i$ is the readback map whenever $j \geq i$. The readback maps satisfy

$$\rho_{ii} = \mathrm{id}_{R_i}, \qquad \rho_{ki} = \rho_{ji} \circ \rho_{kj} \quad (k \geq j \geq i),$$

and preserve classifiers:

$$x \sim_j y \to \rho_{ji}(x) \sim_i \rho_{ji}(y).$$

The self-similar operations are compatible when

$$\rho_{ji}(\alpha_j(x)) \sim_i \alpha_i(\rho_{ji}(x))$$

for all $j \geq i$ and $x : R_j$.

Definition 9332.2 (Boundary limit with ledger)**.** *The ledgered boundary limit of $\mathcal{D}_\Omega$ is*

$$\partial_\infty \mathcal{D}_\Omega := \sum_{(r_i)\in\prod_{i\in I} R_i} \left(\prod_{i\in I} C_i(r_i)\right) \times \left(\prod_{j\geq i} \rho_{ji}(r_j) \sim_i r_i\right) \times \Lambda_\partial.$$

The component Λ_∂ records the readback rows, compression sources, and legality witnesses consumed by the limit carrier.

Definition 9332.3 (Bulk-boundary exactness)**.** *Let T_Ω be a constrained bulk carrier with projections*

$$\pi_i : T_\Omega \to R_i.$$

The comparison map

$$\kappa : T_\Omega \to \partial_\infty \mathcal{D}_\Omega$$

sends x to the compatible family $((\pi_i(x))_{i\in I}, \Lambda_x)$. Bulk-boundary exactness is the conjunction

$$\mathsf{BulkBoundaryExact}_\Omega(T_\Omega) := \mathsf{Full}_\kappa \times \mathsf{Faithful}_\kappa \times \mathsf{LedgerExact}_\kappa.$$

Here Full_κ says every compatible boundary family is represented by a bulk row, $\mathsf{Faithful}_\kappa$ says equality of all boundary projections identifies bulk rows, and $\mathsf{LedgerExact}_\kappa$ says information compressed by the projection is recoverable from the boundary ledger.

Theorem 9332.4 (Exactness excludes ghost bulk distinctions)**.** *If $\mathsf{BulkBoundaryExact}_\Omega(T_\Omega)$ holds, then there are no bulk rows $x, y : T_\Omega$ such that*

$$\prod_{i\in I}(\pi_i(x) \sim_i \pi_i(y))$$

holds while x and y remain distinct in the bulk classifier.

Proof. The displayed product is exactly the premise consumed by $\mathsf{Faithful}_\kappa$. Hence $x \sim_\Omega y$, contradicting any claimed bulk distinction. The ledger-exact component ensures that the argument is not merely pointwise: any compressed distinction would have to appear in the boundary ledger and therefore in one of the compared rows. □

9332.2 Prime Sieve and Shadows

Definition 9332.5 (Prime invariant packet)**.** *For a readout layer R_i, a prime invariant packet on a row $r : R_i$ is the displayed product*

$$\mathsf{PrimeInv}_i(r) := \mathsf{Irred}_i(r) \times \mathsf{PrimeLike}_i(r) \times \mathsf{Valuation}_i(r) \times \mathsf{Factorization}_i(r) \times \mathsf{EulerLocal}_i(r).$$

In the arithmetic layer this packet is read through the $\mathsf{Prime}^\uparrow$ certificate rows: prime classifier, trial division, factorization composition, Euclid-style divisibility, unboundedness, and Euler-local response.

Definition 9332.6 (Prime sieve)**.** *For a target layer R_i, the prime sieve is the class of arrows $f : X \to R_i$ whose image carries prime invariant packets:*

$$f \in \mathsf{PrimeSieve}_i \quad :\Longleftrightarrow \quad \prod_{x:X} \mathsf{PrimeInv}_i(f(x)).$$

Theorem 9332.7 (Prime sieve is closed under precomposition)**.** *If $f : X \to R_i$ belongs to $\mathsf{PrimeSieve}_i$ and $g : Y \to X$ is any arrow, then $f \circ g : Y \to R_i$ belongs to $\mathsf{PrimeSieve}_i$.*

Proof. For each $y : Y$, the row $(f \circ g)(y)$ is $f(g(y))$. Since $g(y) : X$ and f belongs to the prime sieve, $\mathsf{PrimeInv}_i(f(g(y)))$ holds. This is the required product over Y. □

Definition 9332.8 (Prime shadows and defects)**.** *For $x : T_\Omega$, the prime shadow at layer i is*

$$\mathsf{PrimeShadow}_i(x) := \mathsf{PrimeInv}_i(\pi_i(x)).$$

Assume the layer carries a defect-magnitude readout

$$|\mathsf{PrimeShadow}_i(x)|_i : M_i.$$

The layer defect predicate is

$$\mathsf{PrimeDefect}_i(x) := (|\mathsf{PrimeShadow}_i(x)|_i \# 0),$$

and the bulk prime-defect predicate is

$$\mathsf{PrimeDefect}_\Omega(x) := \sum_{i \in I} \mathsf{PrimeDefect}_i(x).$$

Prime normality is the absence of such a displayed defect:

$$\mathsf{PrimeNormal}_\Omega(x) := \neg \mathsf{PrimeDefect}_\Omega(x).$$

Definition 9332.9 (Prime-shadow faithfulness)**.** *Prime-shadow faithfulness is the statement*

$$\mathsf{PrimeShadowFaithful}_\Omega := \prod_{x:T_\Omega} \left(\mathsf{PrimeInv}_\Omega(x) \# 0 \to \sum_{i \in I} \mathsf{PrimeDefect}_i(x) \right).$$

It says that a nonzero bulk prime invariant cannot be invisible to all boundary readout layers.

Theorem 9332.10 (Ledgered prime expansion gives shadow detection)**.** *Assume* $\mathsf{BulkBoundaryExact}_\Omega(T_\Omega)$*. If the bulk prime invariant ledger of x expands through the boundary ledger, then a nonzero bulk prime invariant in x yields* $\mathsf{PrimeDefect}_i(x)$ *for some $i \in I$.*

Proof. By $\mathsf{LedgerExact}_\kappa$, every bulk prime-invariant witness compressed by κ is recorded in the boundary ledger. If the witness is nonzero, the boundary ledger expansion places that nonzero witness in one coordinate row. That coordinate is precisely a layer i with $|\mathsf{PrimeShadow}_i(x)|_i \# 0$, hence $\mathsf{PrimeDefect}_i(x)$. □

9332.3 Zero Atoms and Defect Incompatibility

Definition 9332.11 (Omega zero atom)**.** *Let*

$$\pi_\zeta : T_\Omega \to R_\zeta$$

be the projection to the zeta-zero readout layer. The Omega zero atom and non-trivial Omega zero atom predicates are

$$\mathsf{ZeroAtom}_\Omega(x) := \mathsf{ZetaZero}(\pi_\zeta(x)),$$

and

$$\mathsf{NTZeroAtom}_\Omega(x) := \mathsf{ZeroAtom}_\Omega(x) \times \mathsf{InCritStrip}(\pi_\zeta(x)).$$

This is the Ω-indexed reading of the source carrier $\mathcal{Z}_\zeta$ from Definition 9254.1.

Definition 9332.12 (Prime-defect zero incompatibility)**.** *The direct incompatibility form is*

$$\mathsf{PrimeDefectZeroIncompatibility}_\Omega := \prod_{x:T_\Omega} (\mathsf{NTZeroAtom}_\Omega(x) \times \mathsf{PrimeDefect}_\Omega(x) \to \bot)\,.$$

The constructive apartness form is

$$\mathsf{PrimeDefectApartZero}_\Omega := \prod_{x:T_\Omega} (\mathsf{InCritStrip}(\pi_\zeta(x)) \times \mathsf{PrimeDefect}_\Omega(x) \to \mathsf{Apart}_0(\zeta(\pi_\zeta(x))))\,.$$

Theorem 9332.13 (Apart-zero form gives defect-zero incompatibility)**.** *Assume* $\mathsf{PrimeDefectApartZero}_\Omega$ *and zero-ledger soundness, namely*

$$\mathsf{ZeroAtom}_\Omega(x) \to \zeta(\pi_\zeta(x)) = 0.$$

Then $\mathsf{PrimeDefectZeroIncompatibility}_\Omega$ *holds.*

Proof. Let $x : T_\Omega$ carry both $\mathsf{NTZeroAtom}_\Omega(x)$ and $\mathsf{PrimeDefect}_\Omega(x)$. The non-trivial zero atom supplies $\mathsf{InCritStrip}(\pi_\zeta(x))$ and $\mathsf{ZeroAtom}_\Omega(x)$. The apart-zero form gives

$$\mathsf{Apart}_0(\zeta(\pi_\zeta(x))).$$

Zero-ledger soundness gives $\zeta(\pi_\zeta(x)) = 0$. These two rows are incompatible, yielding $\bot$. □

9332.4 Conditional RH Assembly

Definition 9332.14 (Omega RH interface)**.** *The Omega-indexed RH target is*

$$\mathsf{RH}_\Omega := \prod_{x:T_\Omega} (\mathsf{NTZeroAtom}_\Omega(x) \to \mathsf{OnCritLine}(\pi_\zeta(x)))\,.$$

It is transported to the BEDC-RH predicate when every legal non-trivial zeta-zero packet is represented by a row of T_Ω *through* π_ζ *and* $\mathsf{OnCritLine}$ *is stable under the zeta-layer classifier.*

Theorem 9332.15 (Prime-shadow sufficient interface for RH)**.** *Assume the following data:*

$$\mathsf{BulkBoundaryExact}_\Omega(T_\Omega), \qquad \mathsf{PrimeShadowFaithful}_\Omega,$$
$$\mathsf{PrimeDefectApartZero}_\Omega, \qquad \textit{zero-ledger soundness},$$
$$\prod_{x:T_\Omega} (\neg\mathsf{PrimeDefect}_\Omega(x) \to \mathsf{PrimeNormal}_\Omega(x)),$$
$$\prod_{x:T_\Omega} (\mathsf{NTZeroAtom}_\Omega(x) \times \mathsf{PrimeNormal}_\Omega(x) \to \mathsf{OnCritLine}(\pi_\zeta(x))),$$

and representation of every legal non-trivial zeta-zero packet in T_Ω*. Then the BEDC-RH predicate of Definition 9239.7 holds.*

Proof. By Definition 9332.13, the apart-zero form and zero-ledger soundness give $\mathsf{PrimeDefectZeroIncompatibility}_\Omega$. Fix $x : T_\Omega$ with $\mathsf{NTZeroAtom}_\Omega(x)$. If $\mathsf{PrimeDefect}_\Omega(x)$ held, incompatibility would yield $\bot$. Hence $\neg\mathsf{PrimeDefect}_\Omega(x)$, and the located classifier row gives $\mathsf{PrimeNormal}_\Omega(x)$. The prime-normal-to-critical-line row then yields $\mathsf{OnCritLine}(\pi_\zeta(x))$, so RH_Ω holds. Representation of legal non-trivial zeta-zero packets through π_ζ transports this statement to the BEDC-RH predicate. □

Principle 9332.16 (Status of the interface)**.** *The sufficient interface above rephrases the RH route as prime-shadow rigidity over a ledgered bulk-boundary diagram. It does not discharge Definition 9246.13; rather, it identifies that incompatibility as the analytic carrier row that prevents a non-trivial zero atom from carrying a positive prime defect.*

Concretized in §8791: Finite prime-shadow boundary interface for the RH bulk-boundary vision.

Chapter 9333

RH Proof Obligation Package

chapter 9253 isolates the type-theoretic zero fiber, and chapter 9332 isolates the prime-shadow rigidity interface. The present chapter records the proof obligation package obtained by composing those two interfaces. It is not a proof of RH. It states which rows are definitional, which rows are already structural consequences, and which rows carry the remaining analytic content.

9333.1 Constructive Target

Definition 9333.1 (RH witness function)**.** *The BEDC-RH witness function is the dependent function*

$$\Phi : \mathsf{NTZeroType} \to \mathsf{FixedHalfZeroType}$$

whose projection back to NTZeroType *is classifier-equal to the identity. Equivalently, for every source row s with*

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s),$$

it returns a displayed $\mathsf{OnCritLine}(s)$ *witness. This is the constructive target of Definition 9239.7 and the section form of Definition 9253.7.*

Concretized in §9046: The typed fixed-half zero witness packages the witness-function route without proving RH.

Definition 9333.2 (Quantitative off-line barrier)**.** *A quantitative off-line barrier is a function*

$$\beta : \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow} \to \mathsf{Rat}^{\uparrow}_{>0}$$

such that for every critical-strip row s,

$$|\Re(s) - 1/2| > 2^{-n} \wedge |\Im(s)| \leq T \to |\zeta(s)| \geq \beta(n, T).$$

It is the positive-apartness strengthening on a displayed compact height window. A height-uniform function of n alone is a separate global analytic barrier assumption and is not part of the ordinary fixed-half witness.

Principle 9333.3 (Refutation data must be typed)**.** *A contradiction route contributes to the BEDC-RH witness target only when it is typed as a same-source conflict between a zero ledger and an apart-zero ledger. A classical refutation without the transport rows needed to build Φ does not inhabit Definition 9333.1.*

9333.2 Structural Rows

Definition 9333.4 (Zero-fiber rows)**.** *The zero-fiber part of the package consists of the following rows:*

$$\mathsf{ZetaZeroType} \simeq \mathrm{Fib}_0(\zeta), \qquad \mathsf{NTZeroType} \simeq \sum_s \big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)\big),$$

and the section equivalence

$$\mathsf{RH} \Longleftrightarrow \mathsf{NTZeroType} \to \mathsf{FixedHalfZeroType}.$$

These are structural descriptions of the target, not analytic exclusions.

Definition 9333.5 (Zero-generator rows)**.** *The zero-generator part of the package consists of the factorization row*

$$G : A \to \mathsf{ZetaDom}^{\uparrow}, \quad \theta_G(a) : \sum_{z:\mathsf{Complex}^{\uparrow}} \big(\mathsf{ZetaVal}(G(a), z) \times z \sim_{\mathsf{Complex}^{\uparrow}} e_{\mathsf{Complex}^{\uparrow}}\big) \quad \Longrightarrow \quad \overline{G} : A \to \mathsf{ZetaZeroType},$$

and the canonical-presentation row asserting that two canonical presentations of the same zero type are equivalent licensed presentations. Hence incompatible canonical generators cannot be presentations of one zero type.

Definition 9333.6 (Omega representation row)**.** *The Ω representation row is the assertion that every legal non-trivial zeta-zero packet is represented by some row of the Omega non-trivial zero carrier*

$$T_\Omega^{nt0} := \sum_{x:T_\Omega} \left(\mathsf{ZetaZero}(\pi_\zeta(x)) \times \mathsf{InCritStrip}(\pi_\zeta(x))\right).$$

This row is the transport bridge from the zero-type statement to the bulk-boundary tower statement.

9333.3 Rigidity Rows

Definition 9333.7 (Prime normal defect)**.** *The prime normal defect predicate is the tower-visible defect*

$$\mathsf{PrimeNormalDefect}_\Omega(x) := \sum_{i \in I} \mathsf{PrimeDefect}_i(x),$$

where each $\mathsf{PrimeDefect}_i(x)$ *is a nonzero prime-shadow readout at boundary layer i. Its negation is*

$$\mathsf{NoPrimeNormalDefect}_\Omega(x) := \neg\mathsf{PrimeNormalDefect}_\Omega(x).$$

This notation packages the same defect surface as Definition 9332.8; it emphasizes the normal direction read by finite prime-window and bulk-boundary shadows.

Definition 9333.8 (Zero atom prime-normal rigidity)**.** *Zero atom prime-normal rigidity is the statement*

$$\mathsf{ZeroAtomNoPrimeNormalDefect}_\Omega := \prod_{x:T_\Omega} \left(\mathsf{NTZeroAtom}_\Omega(x) \to \mathsf{NoPrimeNormalDefect}_\Omega(x)\right).$$

It says that a non-trivial zero atom may carry phase data, but cannot carry a nonzero prime-normal defect.

Definition 9333.9 (Prime-normal critical-line row)**.** *The prime-normal critical-line row is*

$$\mathsf{PrimeNormalImpliesCriticalLine}_\Omega := \prod_{x:T_\Omega} (\mathsf{NTZeroAtom}_\Omega(x) \times \mathsf{NoPrimeNormalDefect}_\Omega(x) \to \mathsf{OnCritLine}(\pi_\zeta(x)))\,.$$

This is the fixed-half interpretation of the absence of prime-normal defect.

Definition 9333.10 (Apart-zero defect row)**.** *The apart-zero defect row is*

$$\mathsf{PrimeDefectApartZero}_\Omega := \prod_{x:T_\Omega} (\mathsf{InCritStrip}(\pi_\zeta(x)) \times \mathsf{PrimeNormalDefect}_\Omega(x) \to \mathsf{Apart}_0(\zeta(\pi_\zeta(x))))\,.$$

Together with zero-ledger soundness, it turns prime-normal defect into a same-source zero and apart-zero conflict.

Theorem 9333.11 (Apart-zero defect gives zero atom prime-normal rigidity)**.** *Assume* $\mathsf{PrimeDefectApartZero}_\Omega$ *and zero-ledger soundness, namely*

$$\mathsf{ZeroAtom}_\Omega(x) \to \zeta(\pi_\zeta(x)) = 0.$$

Then $\mathsf{ZeroAtomNoPrimeNormalDefect}_\Omega$ *holds.*

Proof. Let $x : T_\Omega$ carry $\mathsf{NTZeroAtom}_\Omega(x)$. Suppose, for contradiction, that $\mathsf{PrimeNormalDefect}_\Omega(x)$ holds. The non-trivial zero atom supplies both $\mathsf{InCritStrip}(\pi_\zeta(x))$ and $\mathsf{ZeroAtom}_\Omega(x)$. The apart-zero defect row gives

$$\mathsf{Apart}_0(\zeta(\pi_\zeta(x))).$$

Zero-ledger soundness gives $\zeta(\pi_\zeta(x)) = 0$. This is the same-source conflict of Definition 9253.17. Hence the assumed prime-normal defect is impossible. □

Theorem 9333.12 (Proof obligation package implies RH)**.** *Assume the following rows:*

$$\mathsf{PrimeDefectApartZero}_\Omega, \qquad \textit{zero-ledger soundness}, \qquad \mathsf{PrimeNormalImpliesCriticalLine}_\Omega,$$

and representation of every legal non-trivial zeta-zero packet in T_Ω^{nt0}*. Then the BEDC-RH predicate holds.*

Proof. By Definition 9333.11, the apart-zero defect row and zero-ledger soundness give ZeroAtomNoPrimeNormalDef
For any represented non-trivial zero row $x : T_\Omega^{nt0}$, this yields $\mathsf{NoPrimeNormalDefect}_\Omega(x)$. Applying Definition 9333.9 gives $\mathsf{OnCritLine}(\pi_\zeta(x))$. Representation transports this critical-line witness to every $u : \mathsf{NTZeroType}$, and Definition 9253.7 gives the BEDC-RH predicate. □

Principle 9333.13 (Remaining mathematical content)**.** *After the zero-fiber, generator-factorization, and representation rows are in place, the remaining mathematical content is concentrated in two rigidity rows:*

$$\mathsf{PrimeDefectApartZero}_\Omega, \qquad \mathsf{PrimeNormalImpliesCriticalLine}_\Omega.$$

The first says that a positive prime-normal defect forces apartness from the zero value on the same source row. The second says that absence of such defect is read in the complex projection as the fixed-half condition.

9333.4 Proof Route Ordering

Definition 9333.14 (RH route state)**.** *An RH route state is a packet*

$$\mathsf{RHRouteState} := (\mathsf{AnalyticBase}, \mathsf{ZeroType}, T_\Omega, \mathsf{BulkBoundaryExact}, \mathsf{PrimeSieve}, \mathsf{ZeroRigidity}, \mathsf{CriticalLineReadout}, \mathsf{Wit}$$

The components are interpreted as follows. AnalyticBase *records the closed arithmetic, real, complex, holomorphic, Dirichlet, and zeta interfaces consumed by Definition 9239.7.* ZeroType *is the zero-fiber package of Definition 9333.4.* T_Ω *is the compatible tower body.* BulkBoundaryExact *is the fullness, faithfulness, and ledger-exactness row of Definition 9332.3.* PrimeSieve *is the prime-invariant sieve of Definition 9332.6.* ZeroRigidity *abbreviates Definition 9333.8.* CriticalLineReadout *abbreviates Definition 9333.9. Finally,* WitnessExport *records extraction of* Φ *and the associated modulus and barrier witnesses.*

Principle 9333.15 (Route dependency order)**.** *The route state is ordered by dependency. The analytic base licenses zeta-value and zero predicates. The zero-type package turns those predicates into the source carrier* NTZeroType*. The* Ω *body and bulk-boundary exactness provide a tower representation whose boundary shadows and ledgers are visible. The prime sieve defines the defect surface. The zero-rigidity row rules out prime-normal defect on zero atoms. The critical-line readout converts absence of prime-normal defect into* OnCritLine*. Only after those rows are present can* WitnessExport *produce the BEDC-RH witness function.*

Theorem 9333.16 (Route state closes the witness target)**.** *If an RH route state carries inhabited rows for*

$$\mathsf{ZeroRigidity}, \qquad \mathsf{CriticalLineReadout}, \qquad \mathsf{WitnessExport},$$

and the representation row of Definition 9333.6, then it supplies the witness function

$$\Phi : \mathsf{NTZeroType} \to \mathsf{FixedHalfZeroType}.$$

Proof. Let $u : \mathsf{NTZeroType}$. The representation row supplies $x : T_\Omega^{nt0}$ whose zeta projection is classified with the source of u. The zero-rigidity row yields $\mathsf{NoPrimeNormalDefect}_\Omega(x)$. The critical-line readout row then yields $\mathsf{OnCritLine}(\pi_\zeta(x))$. Transporting this witness along the source classifier gives $\mathsf{OnCritLine}(\pi_s(u))$. Pairing u with that witness gives an element of $\mathsf{FixedHalfZeroType}$ over u, which is $\Phi(u)$. □

9333.5 Witness Export

Definition 9333.17 (Witness export packet)**.** *The witness export packet is*

$$\mathsf{WitnessExport} := (\Phi, \mu, \beta_{\mathrm{loc}}, \mathsf{CriticalLineRestriction}).$$

Here Φ *is the fixed-half section of Definition 9333.1. For a zero-on-strip row* s*, the modulus*

$$\mu_s : \mathsf{Nat}^\uparrow \to \mathsf{Nat}^\uparrow$$

extracts rational approximants witnessing $\Re(s) \sim 1/2$*. The barrier*

$$\beta_{\mathrm{loc}} : \mathsf{Nat}^\uparrow \times \mathsf{Nat}^\uparrow \to \mathsf{Rat}^\uparrow_{>0}$$

is the height-local off-line apartness function of Definition 9333.2. The remaining field is the restriction equivalence between non-trivial zero packets and critical-line height rows described in section 9239.4. The Riemann-von Mangoldt count and the trivial-zero classifier are support ledgers for the full zeta formalization; they are not fields exported by the RH section.

Theorem 9333.18 (Fixed-half section exports the modulus witness)**.** *An inhabitant of*

$$\Phi : \mathsf{NTZeroType} \to \mathsf{FixedHalfZeroType}$$

exports, for each $u : \mathsf{NTZeroType}$*, a modulus row*

$$\mu_{\pi_s(u)} : \mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$$

of the kind stated in Definition 9239.13.

Proof. The value $\Phi(u)$ contains the same zero packet u and a witness of $\mathsf{OnCritLine}(\pi_s(u))$. By Definition 9239.13, the constructive content of that $\mathsf{OnCritLine}$ witness is a modulus selecting rational approximants whose distance from $1/2$ is bounded by the requested dyadic tolerance. □

Principle 9333.19 (Barrier export is analytic content)**.** *The function* β_{loc} *is not obtained from the section form alone. It is exported only from the apart-zero defect row when that row is strengthened to a quantitative lower-bound statement on a displayed height window:*

$$|\Re(s) - 1/2| > 2^{-n} \wedge |\Im(s)| \leq T \to |\zeta(s)| \geq \beta(n, T).$$

Thus Φ *records fixed-half membership, while* β_{loc} *records the compact-window lower-bound form of off-line exclusion. A global height-uniform barrier remains a distinct route assumption.*

Principle 9333.20 (Witness export boundary)**.** *The route ordering and witness export packet do not reduce the analytic difficulty of* RH*. They specify the shape of a closed BEDC proof: first a same-source zero/apart-zero exclusion via* $\mathsf{PrimeDefectApartZero}_{\Omega}$*, then a fixed-half readout via* $\mathsf{PrimeNormalImpliesCriticalLine}_{\Omega}$*, and only then the export of* Φ*,* μ*, height-local barrier data when supplied by the analytic route, and critical-line restriction witnesses.*

Chapter 9334

Non-Fixed Orbit Targets for RH

chapter 9254 identifies the RH statement as a section into a fixed half fiber over the zero source. The present chapter isolates the corresponding orbit formulation. It records the tower-internal target that a constructive proof would have to inhabit, and separates that target from weaker refutational orbit exclusions. It does not prove RH.

9334.1 Candidate Carrier and Orbit Predicates

Definition 9334.1 (Critical-strip candidate carrier)**.** *The candidate carrier below the zero source is*

$$\mathcal{C}_\zeta := \sum_s \mathsf{InCritStrip}(s).$$

Write

$$Z_\zeta(c) := \mathsf{ZetaZero}(c)$$

for the zero predicate on this carrier. The mirror readout is the involution

$$J_\zeta : \mathcal{C}_\zeta \to \mathcal{C}_\zeta, \qquad J_\zeta(s) := 1 - \overline{s},$$

with the displayed equation

$$J_\zeta(J_\zeta(c)) = c.$$

The source carrier $\mathcal{Z}_\zeta$ of Definition 9254.1 may equivalently be read as $\sum_{c:\mathcal{C}_\zeta} Z_\zeta(c)$.

Definition 9334.2 (Mirror orbit predicates)**.** *For $c : \mathcal{C}_\zeta$ define the fixed and non-fixed predicates*

$$\mathsf{Fix}_J(c) := (J_\zeta(c) = c), \qquad \mathsf{NonFix}_J(c) := \neg(J_\zeta(c) = c).$$

If

$$c = 1/2 + \delta(c) + i\tau(c),$$

the apart non-fixed predicate is the positive separation certificate

$$\mathsf{ApartNonFix}_J(c) := \sum_{\varepsilon:\mathbb{Q}_{>0}} (|\delta(c)| \geq \varepsilon).$$

The orbit-zero predicate is

$$\mathsf{OrbitZero}_J(c) := Z_\zeta(c) + Z_\zeta(J_\zeta(c)).$$

9334.2 Fixed Section and Orbit Exclusions

Definition 9334.3 (Zero fixed section and orbit exclusions)**.** *The source fixed-section target is*

$$\mathsf{ZeroFix}_\zeta := \prod_{c:\mathcal{C}_\zeta} (Z_\zeta(c) \to J_\zeta(c) = c).$$

The weak non-fixed orbit exclusion is

$$\mathsf{NoZeroOnNonFixedOrbit} := \prod_{c:\mathcal{C}_\zeta} (\mathsf{NonFix}_J(c) \to \neg\mathsf{OrbitZero}_J(c)).$$

The apart orbit exclusion is

$$\mathsf{NoZeroOnApartOrbit} := \prod_{c:\mathcal{C}_\zeta} (\mathsf{ApartNonFix}_J(c) \to \mathsf{Apart}_0(\zeta(c))),$$

where $\mathsf{Apart}_0(\zeta(c))$ *is a positive lower-bound certificate separating* $\zeta(c)$ *from the zero row.*

Theorem 9334.4 (Source fixed section is the RH target)**.** *The source fixed-section target* $\mathsf{ZeroFix}_\zeta$ *is equivalent to the section form* $\mathsf{Sec}_{\mathsf{RH}}$*. This is a scoped readout of the same fixed-half section through the mirror-orbit layer.*

Proof. By Definition 9334.1, an element of $\mathcal{Z}_\zeta$ is a candidate row $c : \mathcal{C}_\zeta$ together with a zero witness $Z_\zeta(c)$. By Definition 9414.163, the mirror fixed equation $J_\zeta(c) = c$ is equivalent to $\Re(c) = 1/2$. The latter is the real half predicate used in Definition 9254.2. Thus a function $Z_\zeta(c) \to J_\zeta(c) = c$ is exactly a section into the fixed real half fiber, and the constructions in Definition 9254.4 give the equivalence. □

Theorem 9334.5 (Fixed section excludes weak non-fixed orbit zeros)**.** *There is a canonical implication*

$$\mathsf{ZeroFix}_\zeta \to \mathsf{NoZeroOnNonFixedOrbit}.$$

Proof. Assume $\mathsf{ZeroFix}_\zeta$ and fix $c : \mathcal{C}_\zeta$ with $n : \mathsf{NonFix}_J(c)$. If the left summand of $\mathsf{OrbitZero}_J(c)$ holds, then $Z_\zeta(c)$ and the fixed section give $J_\zeta(c) = c$, contradicting n. If the right summand holds, apply the fixed section to $J_\zeta(c)$. This gives $J_\zeta(J_\zeta(c)) = J_\zeta(c)$. Since J_ζ is an involution, this identifies c with $J_\zeta(c)$, again contradicting n. Hence $\mathsf{OrbitZero}_J(c)$ is impossible. □

Proposition 9334.6 (Weak orbit exclusion only double-negates fixedness)**.** *There is a canonical map*

$$\mathsf{NoZeroOnNonFixedOrbit} \to \prod_{c:\mathcal{C}_\zeta} (Z_\zeta(c) \to \neg\neg(J_\zeta(c) = c)).$$

Proof. Let $N : \mathsf{NoZeroOnNonFixedOrbit}$ and let $z : Z_\zeta(c)$. To prove $\neg\neg(J_\zeta(c) = c)$, assume $n : \neg(J_\zeta(c) = c)$. The zero witness z gives the left summand of $\mathsf{OrbitZero}_J(c)$. Applying $N(c)(n)$ contradicts that orbit-zero witness. Thus the fixed equation is double-negated. Producing $J_\zeta(c) = c$ itself would require an additional stability, located equality, or decidability principle. □

9334.3 Constructive Apart Route

Theorem 9334.7 (Apart orbit exclusion yields the fixed section under located readout)**.** *Assume the bias readout $\delta : \mathcal{C}_\zeta \to \mathbb{R}$ is located in the sense that, for every rational $q > 0$, a proof of $\neg(|\delta(c)| \geq q)$ gives $|\delta(c)| < q$. Assume also that $\mathsf{Apart}_0(\zeta(c))$ contradicts $Z_\zeta(c)$. Then* $\mathsf{NoZeroOnApartOrbit}$ *supplies* $\mathsf{ZeroFix}_\zeta$*.*

Proof. Fix $c : \mathcal{C}_\zeta$ and $z : Z_\zeta(c)$. It suffices to show $\delta(c) = 0$, since the mirror fixed equation is equivalent to $\delta(c) = 0$ by Definition 9414.163. Let $q : \mathbb{Q}_{>0}$. If $|\delta(c)| \geq q$, then $(q, |\delta(c)| \geq q)$ is an element of $\mathsf{ApartNonFix}_J(c)$. Applying $\mathsf{NoZeroOnApartOrbit}$ yields $\mathsf{Apart}_0(\zeta(c))$, contradicting z. Hence $\neg(|\delta(c)| \geq q)$, and locatedness gives $|\delta(c)| < q$. Since this holds for every positive rational q, the located real readout identifies $\delta(c)$ with zero. Therefore $J_\zeta(c) = c$, giving the fixed section. □

Principle 9334.8 (Constructive orbit boundary)**.** *The tower-internal source target is* $\mathsf{ZeroFix}_\zeta$*. The apart orbit exclusion is a constructive route to that target only when paired with located equality transport for the bias coordinate. The weak non-fixed orbit exclusion is a refutation schema and does not by itself inhabit the BEDC witness type of Definition 9239.8.*

Concretized in §8874: Finite mirror-orbit target packet records the critical-strip candidate, mirror involution, orbit-zero row, fixedness boundary, apartness witness, and weak-exclusion obstruction as NameCert rows.

Chapter 9335

Counterexample Program Safety Flow for RH

chapter 9252 rewrites the critical line as the zero-bias surface of the prime information channels. The present chapter records the adjacent computational normal form. It does not supply a proof of RH. It rewrites the same conjecture as a safety claim for a recognized program run: no accepted run of the counterexample-search flow emits a CounterRH event.

9335.1 Elementary arithmetic interface

Definition 9335.1 (Lagarias bound interface)**.** *The Lagarias bound interface is the displayed arithmetic predicate* $\mathsf{LagBound}(n)$ *on positive natural-number packets:*

$$\mathsf{LagBound}(n) :\Longleftrightarrow \sigma(n) \leq H_n + \exp(H_n)\log(H_n),$$

where

$$\sigma(n) = \sum_{d|n} d, \qquad H_n = \sum_{k=1}^{n} \frac{1}{k}.$$

Write the right-hand side as

$$\mathsf{LagRHS}(n) = H_n + \exp(H_n)\log(H_n).$$

The interface also carries the external transport row

$$\mathsf{RH} \Longleftrightarrow \forall n \geq 1,\ \mathsf{LagBound}(n).$$

Inside the present vision chapter this row is consumed as an equivalence interface, not as an internal BEDC proof of the analytic theorem.

Definition 9335.2 (Lagarias bad packet)**.** *For a positive natural-number packet* n*, define*

$$\mathsf{LagBad}(n) :\Longleftrightarrow \neg\mathsf{LagBound}(n).$$

Equivalently, at the arithmetic display level,

$$\mathsf{LagBad}(n) :\Longleftrightarrow \sigma(n) > \mathsf{LagRHS}(n).$$

Proposition 9335.3 (Bad packets are RH counter packets)**.** *Under the Lagarias bound interface,*

$$\mathsf{RH} \iff \neg\exists n \geq 1,\ \mathsf{LagBad}(n).$$

Proof. The interface gives

$$\mathsf{RH} \iff \forall n \geq 1,\ \mathsf{LagBound}(n).$$

By Definition 9335.2, the negation of the right-hand side is precisely the existence of a positive index with $\mathsf{LagBad}(n)$. Thus RH is equivalent to the absence of Lagarias bad packets. □

Definition 9335.4 (Lagarias interval certificate)**.** *A Lagarias interval certificate*

$$\mathsf{IntervalCert}(n, q, a, b)$$

is a finite rational packet with $a, b \in \mathbb{Q}$ such that

$$a \leq \mathsf{LagRHS}(n) \leq b \qquad \textit{and} \qquad b - a < 2^{-q}.$$

The interval row is the finite comparison surface for the real expression $\exp(H_n)\log(H_n)$; it is not an exact real-number oracle.

Definition 9335.5 (Certified counter packet)**.** *A certified counter packet $\mathsf{CounterLag}(n, q)$ is the existence of rational endpoints a, b such that*

$$\mathsf{IntervalCert}(n, q, a, b) \qquad \textit{and} \qquad b < \sigma(n).$$

It is the finite event saying that the upper interval bound for $\mathsf{LagRHS}(n)$ already lies below the exact divisor sum.

Proposition 9335.6 (Certified counters are sound)**.** *For every positive index n and precision row q,*

$$\mathsf{CounterLag}(n, q) \Longrightarrow \mathsf{LagBad}(n).$$

Proof. Choose rational endpoints a, b witnessing $\mathsf{CounterLag}(n, q)$. Then

$$\mathsf{LagRHS}(n) \leq b < \sigma(n),$$

so $\sigma(n) > \mathsf{LagRHS}(n)$. This is exactly $\mathsf{LagBad}(n)$ by Definition 9335.2. □

9335.2 Program and prime-register carrier

Definition 9335.7 (Counterexample enumerator)**.** *The counterexample enumerator P_{Lag} is the program-flow schema whose state contains a positive index n, a precision parameter q, divisor-sum data, harmonic-number data, a comparison row, and an output row. Its scheduler fairly enumerates pairs (n, q) with $n \geq 1$. At the pair (n, q) it computes the exact rows $\sigma(n)$ and H_n, computes the precision-q interval row for $\exp(H_n)\log(H_n)$, and emits only when that finite row certifies $\mathsf{CounterLag}(n, q)$. When such a packet is accepted, the output row is*

$$\mathsf{CounterRH}(n, q).$$

Proposition 9335.8 (Enumerator output criterion)**.** *The program P_{Lag} emits a $\mathsf{CounterRH}$ event exactly when some positive index satisfies LagBad:*

$$P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH} \iff \exists n \geq 1,\ \mathsf{LagBad}(n).$$

Proof. If P_{Lag} emits $\mathsf{CounterRH}(n, q)$, the emission row is guarded by the accepted certified counter packet $\mathsf{CounterLag}(n, q)$. By Definition 9335.6, this implies $\mathsf{LagBad}(n)$. Conversely, if $\mathsf{LagBad}(n)$ holds, the strict comparison has a positive gap. Exact arithmetic supplies $\sigma(n)$ and H_n, while interval narrowing for $\exp(H_n)\log(H_n)$ eventually separates the two sides. Hence some precision q gives a certified counter packet. The fair scheduler eventually reaches the pair (n, q) and accepts that packet, so it emits $\mathsf{CounterRH}(n, q)$. □

Definition 9335.9 (Prime-register Godel state)**.** *A prime-register Godel state is a natural-number packet*

$$N = \prod_i p_i^{r_i},$$

where p_i is the i-th displayed prime row and r_i is the value of the i-th register. The register readback and encoding rows satisfy

$$\mathsf{Encode}(\mathsf{Decode}(N))\mathsf{hsame}N.$$

For the RH counterexample program the register list includes the current index, precision budget, divisor-sum accumulators, harmonic numerator and denominator accumulators, comparison state, program counter, and counter-output flag.

Definition 9335.10 (FRACTRAN counterexample program)**.** *A FRACTRAN counterexample program F_{RH} is a finite ordered list of positive rational instructions*

$$f_0, \ldots, f_{m-1}.$$

At state N, the accepted instruction is the least displayed index i such that Nf_i is a natural number, and the next state is

$$N \mapsto Nf_i.$$

The program F_{RH} is the prime-register compilation of P_{Lag}: the listed rational instructions implement the enumerator registers, the certified counter-packet registers, and the counter-output flag.

Proposition 9335.11 (Prime-register preservation of counter output)**.** *The FRACTRAN program F_{RH} reaches a state with positive counter-output flag exactly when P_{Lag} emits* $\mathsf{CounterRH}$*:*

$$F_{\mathsf{RH}} \Downarrow \mathsf{halt}_{\mathsf{RH}} \iff P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH}.$$

Proof. The compilation is register preserving. The prime exponent of each register prime stores the corresponding program register of P_{Lag}, and each rational instruction performs the guarded increment, decrement, and branch operations of the enumerator. The counter-output flag is set only by the instruction block reached after an accepted certified counter packet, and every accepted certified counter packet routes to that block. Hence the FRACTRAN halt-prime state and the $\mathsf{CounterRH}$ output row are the same event read through different register coordinates. □

9335.3 GroundCompiler run recognition

Definition 9335.12 (RH FRACTRAN event)**.** *An RH FRACTRAN event is a finite tuple*

$$E_k = (N_k, i_k, f_{i_k}, N_{k+1}, C_k),$$

where N_k and N_{k+1} are prime-register Godel states, f_{i_k} is the accepted FRACTRAN instruction, and C_k is a step-certificate row proving

$$N_{k+1}\mathsf{hsame}N_k f_{i_k},$$

with i_k the least displayed instruction index for which $N_k f_{i_k}$ is a natural-number packet.

Definition 9335.13 (Recognized RH run)**.** *A recognized RH run is an EventFlow*

$$S = E_0 \mid E_1 \mid E_2 \mid \cdots$$

accepted by a GroundCompiler recognizer $\mathsf{RecognizedRHRun}(F_{\mathsf{RH}}, S)$*. The recognizer checks the legal EventFlow channel row, the FRACTRAN transition row of each event, the least-instruction row, the prime-exponent register ledger, the initial state row for* P_{Lag}*, and the output decoding row.*

Definition 9335.14 (Counter output)**.** *For a recognized RH run* S*, the predicate* $\mathsf{Emits}(S, \mathsf{CounterRH}(n, q))$ *means that some event in* S *decodes the counter-output flag with payload* (n, q)*. The no-counter predicate is*

$$\mathsf{NoCounterOutput}(S) :\Longleftrightarrow \neg\exists n, q,\ n \geq 1 \wedge \mathsf{Emits}(S, \mathsf{CounterRH}(n, q)).$$

Definition 9335.15 (Information RH program flow)**.** *The information RH program flow is the compiler-facing packet*

$$\mathsf{InfoRHProgramFlow}^{\uparrow} = (G, F, L, E, R, I, N),$$

where

$$G : \textit{prime-register Godel state rows}, \quad F : F_{\mathsf{RH}}\ \textit{FRACTRAN program rows},$$
$$L : \mathsf{LagBound}, \mathsf{LagRHS}, \mathsf{IntervalCert}, \mathsf{CounterLag}\ \textit{rows},$$
$$E : \textit{GroundCompiler EventFlow rows}, \quad R : \mathsf{CounterRH}(n, q)\ \textit{recognizer rows},$$
$$I : \mathsf{NoCounterOutput}\ \textit{and safety-invariant rows}, \quad N : \mathsf{NameCert}\ \textit{local naming row.}$$

The packet names the counterexample-search run surface; it does not certify that the surface is counter-free.

Theorem 9335.16 (Program safety normal form of RH)**.** *Under the Lagarias bound interface and the register-preserving compilation of* P_{Lag} *into* F_{RH}*, the Riemann Hypothesis is equivalent to the absence of recognized counter output:*

$$\mathsf{RH} \Longleftrightarrow \neg\exists S, n, q,\ \mathsf{RecognizedRHRun}(F_{\mathsf{RH}}, S) \wedge \mathsf{Emits}(S, \mathsf{CounterRH}(n, q)).$$

Proof. By Definition 9335.3, RH is equivalent to the absence of a positive n satisfying $\mathsf{LagBad}(n)$. By Definition 9335.8, such an n exists exactly when P_{Lag} emits $\mathsf{CounterRH}$. By Definition 9335.11, this is exactly the event that F_{RH} reaches the counter-output flag. Finally, Definition 9335.13 makes the GroundCompiler recognizer accept precisely the initial-state FRACTRAN run with its step certificates and prime-exponent ledger. Thus a recognized counter event exists exactly when the Lagarias bad packet exists, which is exactly the negation of RH under the interface. □

Corollary 9335.17 (Specific halting instance)**.** *The halt-prime reachability claim for* F_{RH} *is a single program-instance claim:*

$$F_{\mathsf{RH}} \Downarrow \mathsf{halt}_{\mathsf{RH}} \Longleftrightarrow \neg\mathsf{RH}.$$

It is not a uniform classifier for arbitrary program halting.

Proof. Combine Definition 9335.11 with Definition 9335.8 and Definition 9335.3. The resulting statement concerns the fixed program F_{RH} and the fixed counter-output row defined above. No classifier ranging over arbitrary program codes is introduced. □

9335.4 Refutation layer

Definition 9335.18 (Counter refutation transformer)**.** *A counter refutation transformer for the RH program flow is a displayed function*

$$\mathsf{RefuteCounter} : \prod_{n,q} \mathsf{CounterLag}(n,q) \to \bot.$$

Equivalently, it is a proof that no certificate accepted by the counter-output gate can pass the Lagarias comparison rows.

Proposition 9335.19 (Halting assumption exposes a counter packet)**.** *Assuming that P_{Lag} halts does not by itself produce a contradiction. It produces a certified counter packet:*

$$P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH} \Longrightarrow \exists n, q,\ \mathsf{CounterLag}(n,q).$$

Consequently,

$$P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH} \Longrightarrow \exists n \geq 1,\ \mathsf{LagBad}(n).$$

Proof. By the definition of P_{Lag}, the only output row is $\mathsf{CounterRH}(n,q)$, and that output is guarded by an accepted $\mathsf{CounterLag}(n,q)$ packet. The second implication follows from Definition 9335.6. □

Theorem 9335.20 (Halting refutation normal form)**.** *Under the Lagarias bound interface, the following are equivalent:*

$$\mathsf{RH},$$

$$\neg(P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH}),$$

and

$$\prod_{n,q} \mathsf{CounterLag}(n,q) \to \bot.$$

Proof. By Definition 9335.3, RH is equivalent to the absence of any $\mathsf{LagBad}(n)$. By Definition 9335.8, the existence of such an n is equivalent to P_{Lag} emitting a counter event. Hence RH is equivalent to $\neg(P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH})$.

The program emits a counter event exactly when some finite $\mathsf{CounterLag}(n,q)$ packet is accepted. Therefore the negation of halting is the absence of all such packets, which is the displayed product of refutations

$$\prod_{n,q} \mathsf{CounterLag}(n,q) \to \bot.$$

□

Remark 9335.21 (Role of contradiction)**.** *The contradiction step in a proof by contradiction is not supplied by the halting assumption alone. The halting assumption supplies* $\mathsf{CounterLag}(n,q)$ *for some finite pair (n,q). A contradiction is obtained only after a separate no-counter invariant supplies* $\mathsf{CounterLag}(n,q) \to \bot$*. This invariant is not a free compiler fact; by Definition 9335.20, it is the RH proof content in the Lagarias coordinates.*

9335.5 Safety fixed point

Definition 9335.22 (RH safety class)**.** *The RH safety class is*

$$\mathsf{Safe}_{\mathsf{RH}} = \{S : \mathsf{NoCounterOutput}(S)\}.$$

For a recognized finite prefix S, write $T_{\mathsf{RH}}(S) = S \mid E_{\mathsf{next}}$ when the next RH FRACTRAN event is accepted by the recognizer.

Proposition 9335.23 (Safety invariance is no-counter output)**.** *For the recognized run of F_{RH} from the initial counterexample-search state, the following are equivalent:*

$$\forall \text{ recognized prefixes } S, \quad S \in \mathsf{Safe}_{\mathsf{RH}},$$

and

$$\mathsf{NoCounterOutput}(S^{\infty}_{\mathsf{RH}}).$$

Proof. If every recognized prefix is safe, no event in the infinite recognized run can decode a $\mathsf{CounterRH}$ payload, because any such event appears in some finite prefix. Conversely, if the infinite recognized run has no counter-output event, every finite prefix inherits the same absence. □

Corollary 9335.24 (RH as a safety fixed point)**.** *Under the interfaces of Definition 9335.16,*

$$\mathsf{RH} \iff \forall \text{ recognized prefixes } S, \quad T_{\mathsf{RH}}(S) \in \mathsf{Safe}_{\mathsf{RH}} \text{ whenever } S \in \mathsf{Safe}_{\mathsf{RH}}.$$

Equivalently, the recognized counterexample-search EventFlow remains inside the greatest safety class

$$\nu X.\ \{S : \mathsf{next}(S) \text{ is not a } \mathsf{CounterRH} \text{ event and every prefix lies in } X\}.$$

Proof. The transition form is the prefix reading of Definition 9335.23. Combining that proposition with Definition 9335.16 gives the RH equivalence. The greatest-fixed-class display is the coinductive packaging of the same prefix condition: a run remains inside the class exactly when no recognized step exits through the counter-output row. □

Remark 9335.25 (Residual bridge target)**.** *A stronger structural route would not merely refute $\mathsf{CounterLag}(n, q)$ directly. It would route an accepted $\mathsf{CounterRH}(n, q)$ event into an off-mirror residual packet and then show that the residual dynamics force the residual to vanish. In schematic form the missing bridge is*

$$\mathsf{CounterRH}(n, q) \to \mathsf{OffMirrorZero} \to \mathsf{NonzeroSkewResidual}.$$

If a later Zeckendorf or finite-window projection tower supplied a return rule $R(\Delta) = -\frac{1}{2}\Delta$ together with a zero-event safety rule $R(\Delta) = \Delta$, then the residual would satisfy $\Delta = 0$, contradicting the nonzero residual row. The present chapter records this as a bridge target only; it does not claim a certified map from Lagarias counter packets to that residual surface.

Theorem 9335.26 (Information halting normal form)**.** *Under the displayed interfaces of this chapter, the following claims are equivalent:*

$$\mathsf{RH},$$

$$\forall n \geq 1,\ \mathsf{LagBound}(n),$$

$$\neg(P_{\mathsf{Lag}} \Downarrow \mathsf{CounterRH}),$$

$$\neg(F_{\mathsf{RH}} \Downarrow \mathsf{halt}_{\mathsf{RH}}),$$

$$\neg\exists S, n, q,\ \mathsf{RecognizedRHRun}(F_{\mathsf{RH}}, S) \wedge \mathsf{Emits}(S, \mathsf{CounterRH}(n, q)),$$

and

$$\mathsf{InfoRHProgramFlow}^{\uparrow} \textit{ preserves } \mathsf{NoCounterOutput}.$$

Proof. The first two claims are the Lagarias bound interface of Definition 9335.1. The second and third are equivalent by Definition 9335.8. The third and fourth are equivalent by Definition 9335.11. The fourth and fifth are equivalent by the recognized-run relation of Definition 9335.13 and the counter-output predicate of Definition 9335.14. The fifth and sixth are the safety field of the packet in Definition 9335.15. Therefore all six are the same no-counter safety claim read through arithmetic, program, prime-register, EventFlow, and packet coordinates. □

Remark 9335.27 (Halting boundary)**.** *The packet* $\mathsf{InfoRHProgramFlow}^{\uparrow}$ *is a precise counterexample-search object, not a proof of counterexample absence. Producing the program, the FRACTRAN carrier, and the EventFlow recognizer only exposes the safety invariant. Proving that the recognized run never emits* $\mathsf{CounterRH}$ *is, by Definition 9335.16, the RH proof obligation itself. Thus the construction sits below the boundary of chapter 9336: it records a fixed program run and a no-counter safety claim, while a classifier deciding such claims for arbitrary routes would cross the halting boundary described in chapter 9369.*

Concretized in §829: GroundCompiler event-flow audit packets supply the channel, event, certificate-gate, transport, provenance, and local naming rows used by the recognized RH run surface.

Concretized in §1048: GroundCompiler recognizer-flow packets supply the generated recognizer discipline for accepting only ledgered FRACTRAN step events and output rows.

Concretized in §1094: Flow-level compile-decode packets supply the event-boundary readback needed before the counter-output recognizer is applied.

Concretized in §64: Prime packets supply the factorization rows that make prime-exponent register states readable as finite arithmetic ledgers.

Concretized in §556: Generator fixed-point packets supply the fixed-output boundary separating a displayed no-counter safety invariant from an unrestricted route-deciding oracle.

Chapter 9336

RH Route Closure and the Halting Boundary

The RH capstone of chapter 9239 states the Riemann Hypothesis as a BEDC Π-statement and fixes its constructive proof content: a proof is a total function Φ sending every critical-strip zeta-zero witness to an $\mathsf{OnCritLine}$ witness. The present chapter records the adjacent boundary. It separates the fixed proposition RH from proof search, from condition-chain closure, and from a proposed decision procedure for all theorem routes. This separation is necessary because the halting obstruction of chapter 9368 acts on families of Cont-chains, not directly on the truth value of one fixed conjecture.

9336.1 Three strata around RH

Definition 9336.1 (RH condition)**.** *An* RH condition *is a pair* (K, τ) *where* K *is a displayed condition, classifier, spectral statement, positivity statement, zero-set statement, operator statement, or other BEDC proposition, and* $\tau : K \to \mathsf{RH}$ *is a certified transport. If a witness* $k : K$ *is also supplied, then* $\tau(k)$ *is a proof of* RH*.*

Definition 9336.2 (Finite RH condition)**.** *An RH condition* (K, τ) *is* finite *when* K *has a finite* $\mathsf{NameCert}$ *or proof packet,* K *has a witness* $k : K$*, the transport* $\tau : K \to \mathsf{RH}$ *is checked, the dependency ledger is complete, the proof passes axiom-purity, and no hidden selector, quotient identification, proposition-extensional collapse, external proof source, or unclosed* GAP *is consumed.*

Theorem 9336.3 (Finite RH condition closes RH)**.** *If* (K, τ) *is a finite RH condition and* $k : K$ *is its displayed witness, then* RH *is closed on that route.*

Proof. The transport component of the condition is a function $\tau : K \to \mathsf{RH}$. Applying it to $k : K$ gives $\tau(k) : \mathsf{RH}$. Finiteness adds the BEDC side conditions: the witness, the transport, the dependency ledger, and the purity audit all live inside the displayed fragment. Thus the result is an internal proof route, not a meta-level assertion that such a route exists. □

Definition 9336.4 (RH condition chain)**.** *An* RH condition chain *is a sequence* $K_0, K_1, K_2, \ldots$ *with* $K_0 = \mathsf{RH}$ *and displayed transports*

$$K_n \to K_{n+1}, \qquad K_{n+1} \to K_n$$

at each stage where equivalence is claimed. It closes at level N *when a witness* $w_N : K_N$ *is supplied and the reverse transports carry* w_N *back to* K_0*.*

Theorem 9336.5 (Equivalence chains transport obligations)**.** *An infinite chain*

$$\mathsf{RH} \Longleftrightarrow K_1 \Longleftrightarrow K_2 \Longleftrightarrow \cdots$$

does not by itself prove RH*. It proves* RH *only when some displayed stage has a witness that can be transported back to* K_0*.*

Proof. Each equivalence supplies transport between neighbouring obligations. A transport moves an existing witness; it does not create one. If no stage K_N has a witness $w_N : K_N$, there is no term for the reverse transport chain to consume. The chain then changes the coordinate system of the remaining GAP without discharging it. □

9336.2 Grounding rank

Definition 9336.6 (Rule-presented condition)**.** *An infinite RH condition is* rule-presented *when it is supplied by a displayed generator, recognizer, continuation system, classifier, or packet family rather than by a bare external totality claim. It is* closed-rule-presented *when the generator itself has a finite* NameCert*, a totality witness, a correctness witness, and a dependency ledger.*

Definition 9336.7 (Grounding rank)**.** *The grounding rank of an RH condition is defined by the following clauses. The rank is* 0 *when the condition itself has a finite certificate and finite witness packet. The rank is* $n + 1$ *when the condition is generated by a rule whose grounding rank is* n*. If no finite rank is available, the rank is* ω*.*

Theorem 9336.8 (Finite grounding rank gives a finite route)**.** *If an RH condition has grounding rank* $N < \omega$*, and every transport from the finite top of the rule tower back to* RH *has a finite certificate, then the condition determines a finite BEDC proof route to* RH*.*

Proof. Unfold the finite rank. At the top level there is a finite certificate and witness packet. The certified rules below it successively generate the next lower condition, and the certified transports carry the resulting witness down the tower until it reaches $K_0 = \mathsf{RH}$. Only finitely many rules and transports are consumed, so the route remains a finite BEDC object even if the condition it presents has infinite extension. □

Theorem 9336.9 (Infinite rule towers are sockets)**.** *If an RH route has grounding rank* ω*, then that route is not an internal proof of* RH*. It is a* GAP *socket whose far end is named by the apophatic boundary* T*.*

Proof. Rank ω means that every displayed rule still requires a higher rule. No finite layer supplies the top certificate that BEDC needs in order to consume the route as proof data. Treating the entire tower as one object only moves the demand: the tower-object itself now requires a generator, certificate, totality witness, correctness witness, and ledger. In their absence the demand is recorded as a GAP socket. By the far-end discipline of chapter 9401, the far end of such a socket is named only apophatically:

$$\mathrm{FarEnd}(\mathrm{socket}) \equiv_{\mathrm{apo}} \mathsf{T}.$$

□

9336.3 Closure deciders and halting

Definition 9336.10 (Global theorem-route closure decider)**.** *A* global theorem-route closure decider *is a BEDC classifier D that, for every theorem target A and every proof-search route R, decides whether R eventually outputs a closed proof of A.*

Theorem 9336.11 (No global theorem-route closure decider)**.** *There is no global theorem-route closure decider in BEDC.*

Proof. Assume such a classifier D is available. Given an arbitrary certificate $\mathsf{NameCert}_P$ and history h, construct a route $R_{\mathsf{NameCert}_P,h}$ that replays the Cont_P-chain from h and outputs a fixed proof of a trivial target precisely when that chain terminates. Then D applied to that target and route decides whether the Cont_P-chain from h terminates. This is a halting predicate in the sense of Definition 9368.2. By Definition 9368.7 and Definition 9368.4, such a predicate closes the open meta loop at the certificate stratum. The fixed-point obstruction recorded in chapter 9368 forbids that closure. Hence D cannot exist. □

Definition 9336.12 (RH-route closure decider)**.** *An* RH-route closure decider *is a classifier that decides, for each route in a specified class of RH condition chains, rule towers, or proof-output programs, whether that route eventually yields a proof of* RH.

Theorem 9336.13 (Expressive RH-route deciders are Type VII sockets)**.** *If an RH-route closure decider ranges over a class of routes expressive enough to encode arbitrary* Cont*-chain termination problems, then its request is a Type VII socket rather than an internal BEDC classifier.*

Proof. By the expressiveness assumption, each halting instance $(\mathsf{NameCert}_P, h)$ can be represented as closure of an RH-directed route. The proposed decider would therefore decide arbitrary Cont_P-chain termination. This is the same certificate-stratum halting predicate ruled out in Definition 9336.11. Under the socket taxonomy, requests for total truth predicates, runtime validators, self-certifying checkers, and halting-level self-reference closure are Type VII sockets. Thus the proposed decider is not a closed classifier; it is a named self-reference boundary. □

Theorem 9336.14 (Fixed RH is not made independent by halting)**.** *The nonexistence of a global closure decider does not imply*

$$\mathsf{BEDC} \nvdash \mathsf{RH} \qquad \textit{and} \qquad \mathsf{BEDC} \nvdash \neg\mathsf{RH}.$$

Proof. The halting obstruction concerns uniform classification over arbitrary certificate-history pairs or arbitrary theorem routes. The displayed formula RH of Definition 9239.7 is one fixed Π-statement. A fixed statement may be provable, refutable by a constructive counterexample, or independent of a chosen fragment. The third outcome requires an RH-specific meta-theorem about that fragment. It is not obtained merely from the absence of a uniform route-closure classifier. □

9336.4 Audit bases

Theorem 9336.15 (MetaCIC and GroundCompiler are distinct audit bases)**.** *The RH route boundary is read through two cooperating audit bases:* $\mathsf{MetaCIC}$ *and* $\mathsf{GroundCompiler}$*. The former audits the closed-CIC meta-theory surface; the latter audits event flows, source channels, generated recognizers, certificate gates, and non-claim rows. Neither base collapses into the other.*

Proof. chapter 9385 records the closed-CIC critical path: typing surfaces, substitution, reduction geometry, normalisation, confluence fragments, decidable checking, and the blocked subject-reduction lift. The ground-compiler chapters record a different surface: event-flow representation, source-channel separation, recognizer generation, certificate-gate passage, and no-hidden-input discipline. An RH proof route must pass the relevant proof-theoretic checks and the relevant compiler-facing visibility checks, but a pass on one surface is not a pass on the other. This is why a local parser boundary, such as the channel termination of chapter 9369, cannot become a certificate-level halting classifier, and why a MetaCIC decidable checker cannot become a global theorem-route closure decider. □

9336.5 Finite human expression maximality

Definition 9336.16 (Finite communicable expression)**.** *A* finite communicable mathematical expression *is any finite written or programmable mathematical artifact that can be exchanged, inspected, and audited: a definition, theorem, proof, counterexample, conjecture, axiomatic extension, equivalence, proof route, algorithm, condition chain, rule tower, meta-theorem, numerical certificate, spectral condition, operator condition, or zeta-specific analytic packet.*

Definition 9336.17 (BEDC encoding premise for finite expression)**.** *The* finite-expression encoding premise *says that every finite communicable mathematical expression relevant to* RH *can be read in BEDC as one of the following surfaces: a statement row, proof row, program row, condition route,* NameCert *packet,* Pkg *packet,* GAP *row, socket row, cannot-claim row, or meta-theorem row. The premise is an encoding premise, not a truth-completeness premise.*

Definition 9336.18 (Non-oracular expression)**.** *A finite expression is* non-oracular *when it does not consume an unmarked global truth predicate, global halting oracle, global proof-search oracle, global RH-route decider, hidden* Classical.choice*, hidden* Quot.sound*, hidden* propext*, unledgered external supply, or unmarked meta-closure. If such a supply is requested, the expression remains non-oracular only by exposing the request as a* GAP *socket whose far end is named apophatically.*

Definition 9336.19 (RH decision datum)**.** *An* RH decision datum *is one of four artifacts:*

1. *a proof datum* $\Phi : \mathsf{RH}$ *in the sense of Definition 9239.8;*
2. *a constructive counterexample packet* $s_0, \mathsf{ZetaZero}(s_0), \mathsf{InCritStrip}(s_0), \neg\mathsf{OnCritLine}(s_0)$*;*
3. *an RH-specific meta-theorem* $F \not\vdash \mathsf{RH}$ *and* $F \not\vdash \neg\mathsf{RH}$ *for the chosen fragment* F*;*
4. *a finite route-closure certificate, consisting of a displayed route stage witness and certified transports back to* RH*.*

Definition 9336.20 (RH boundary statement)**.** *The* RH boundary statement *is the conjunction of the following claims:* RH *is the* Π*-statement of Definition 9239.7; a proof must supply the total witness function of Definition 9239.8; a disproof must supply a constructive counterexample packet; independence must be an RH-specific meta-theorem about the chosen fragment; an RH condition proves* RH *only through a witness and certified transport; an equivalence chain without a witnessed stage transports obligations rather than producing proof; an infinite rule tower without finite grounding rank is a* GAP *socket; a global route-closure classifier is a halting oracle; and an unclosed RH decision request has only the apophatic far-end reading*

$$\mathrm{FarEnd}(\mathrm{socket}_{\mathsf{RH}}) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Definition 9336.21 (Strict RH-decision strengthening)**.** *A finite expression strictly strengthens the RH boundary statement at the decision layer when it claims more than boundary placement: for example, that* RH *is proved, refuted, or independent; that a concrete RH route closes or cannot close; that all relevant routes are decidable; or that an infinite condition can be consumed as proof resource without finite grounding.*

Theorem 9336.22 (Finite non-oracular RH boundary maximality)**.** *Assume the finite-expression encoding premise of Definition 9336.17. Assume also that no RH proof datum, constructive RH counterexample packet, RH-specific independence theorem, or finite RH route-closure certificate is supplied, and that global halting or proof-search oracles are not admitted. Then any finite communicable non-oracular expression that strictly strengthens the RH boundary statement at the decision layer supplies an RH decision datum.*

Proof. Let E be a finite communicable non-oracular expression, and suppose that it strictly strengthens the RH boundary statement at the decision layer. By the encoding premise, E has a BEDC reading as a statement, proof, program, route, certificate, gap, socket, cannot-claim row, or meta-theorem row.

If E proves RH, then by the constructive form of Definition 9239.8 it supplies a total function Φ from critical-strip zeta-zero witnesses to $\mathsf{OnCritLine}$ witnesses. This is a proof datum.

If E refutes RH constructively, then by the RH formulation it must display a complex history s_0 together with witnesses $\mathsf{ZetaZero}(s_0)$, $\mathsf{InCritStrip}(s_0)$, and $\neg\mathsf{OnCritLine}(s_0)$. This is a counterexample datum.

If E claims independence of RH over a fragment F, then it must supply the fragment-specific meta-theorem $F \nvdash \mathsf{RH}$ and $F \nvdash \neg\mathsf{RH}$. This is an independence datum. By Definition 9336.14, it cannot be inferred merely from the halting boundary.

If E closes a concrete RH route, then it must display a route stage witness and certified transports back to RH, as in Definition 9336.4. This is a finite route-closure certificate, and by Definition 9336.3 it yields the corresponding proof route.

If E only provides another equivalent condition or an infinite chain of equivalent conditions, then Definition 9336.5 applies: without a witnessed stage, it transports the GAP rather than closing it. Such an expression is not a strict decision-layer strengthening.

If E claims that an infinite rule tower can be consumed without finite grounding, then Definition 9336.9 applies: the unavailable finite top certificate is recorded as a GAP socket whose far end is named by T. Again there is no strict decision-layer strengthening unless a finite grounding and closure certificate is supplied.

If E supplies a global route-closure decider, then Definition 9336.11 turns it into a halting predicate and hence an open-meta-loop closure. That violates non-oracularity unless the request is exposed as a socket. A socketed request is a boundary placement, not a decision datum.

These cases exhaust the ways a finite encoded expression can claim more at the RH decision layer. Therefore every non-oracular strict strengthening supplies an RH decision datum. □

Corollary 9336.23 (The apophatic far end does not decide RH)**.** *The boundary reading*

$$\mathrm{FarEnd}(\mathrm{socket}_{\mathsf{RH}}) \equiv_{\mathrm{apo}} \mathsf{T}$$

does not supply Φ, a counterexample, an independence theorem, or a route-closure certificate.

Proof. The symbol T names the far end of an uninternalized supply socket; it is not a kernel object, proof term, zeta value, decision classifier, or global oracle. Thus it records where the request leaves the closed fragment, not what the answer to RH is. □

Corollary 9336.24 (Local RH work remains admissible)**.** *Finite non-oracular work may introduce additional RH equivalences, condition chains, finite exclusions of route classes, zeta-continuation packets, zero-counting witnesses, critical-strip barriers, Hilbert–Polya style packets, Li-criterion generators, or RH-specific independence interfaces. Such work changes the RH decision layer exactly when it produces an RH decision datum.*

Proof. Each listed artifact is a finite communicable expression and so has a BEDC reading under the encoding premise. If it remains local, conditional, or socketed, it refines the boundary without deciding RH. If it strictly strengthens the decision layer, the maximality theorem forces it to supply one of the four RH decision data. □

Corollary 9336.25 (RH boundary summary)**.** *BEDC can state* RH *as a fixed constructive* Π*-statement, can recognize finite RH conditions as proof routes, and can reject infinite rule towers without finite grounding as* GAP/T *sockets. It can also reject global route-closure deciders by the halting boundary. Under the finite-expression encoding premise, this boundary is maximal among finite communicable non-oracular expressions at the RH decision layer: any strict strengthening supplies a proof datum, counterexample datum, fragment-specific independence datum, or finite route-closure certificate.*

Proof. The first claim is Definition 9239.7 and Definition 9239.8. The second is Definition 9336.3. The third is Definition 9336.9. The fourth is Definition 9336.11. The maximality claim is Definition 9336.22. □

Concretized in §5564: The RH-route halting boundary is realized as finite route, grounding-rank, decider-refusal, and NameCert rows.

Concretized in §6933: Finite zero-inspection windows realize the RH route boundary as bounded zeta-zero, critical-strip, transport, provenance, and NameCert rows.

Chapter 9337

Route Closure and Fixed RH Separation

chapter 9336 rejects total closure classifiers for theorem routes by the halting boundary. The present chapter fixes the logical separation needed for the RH stack: route closure is a family-level classification problem, while RH is one fixed constructive Π-statement. A diagonal obstruction against the former is not, by itself, a meta-theorem about the independence of the latter.

9337.1 Route Layer and Statement Layer

Definition 9337.1 (Fixed RH proposition)**.** *The fixed RH proposition is the proposition of Definition 9239.7:*

$$\mathsf{RH} := \prod_s \big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)\big) \to \mathsf{OnCritLine}(s).$$

It is a single constructive statement. It is not the carrier of all proof searches, condition chains, route programs, or theorem generators.

Definition 9337.2 (Route closure family)**.** *For a theorem target A and a route R, write*

$$\mathsf{Closes}(R, A)$$

for the assertion that R eventually outputs a closed proof datum for A. The route closure family is the two-variable predicate $(R, A) \mapsto \mathsf{Closes}(R, A)$.

Definition 9337.3 (Fragment independence)**.** *For a named fragment F, define*

$$\mathsf{Indep}_F(\mathsf{RH}) := (F \nvdash \mathsf{RH}) \times (F \nvdash \neg\mathsf{RH}).$$

This is a statement about the proof predicate of F and the fixed formula RH.

9337.2 Route Closure Undecidability

Theorem 9337.4 (No total route closure classifier)**.** *There is no BEDC classifier D satisfying*

$$D(A, R) = 1 \quad \Longleftrightarrow \quad \mathsf{Closes}(R, A)$$

for every theorem target A and every route R.

Proof. This is the route-family form of Definition 9336.11. If such a classifier existed, then for any certificate $\mathsf{NameCert}_P$ and history h one could form the route $R_{\mathsf{NameCert}_P,h}$ that replays the Cont_P-chain from h and emits a proof of a fixed trivial target exactly when the chain terminates. Applying D to that target and route would decide termination of arbitrary Cont_P-chains, giving the certificate-stratum halting predicate ruled out by Definition 9368.7 and Definition 9368.4. □

Theorem 9337.5 (Expressive RH route families are not decidable)**.** *Let $\mathcal{R}_{\mathsf{RH}}$ be a class of RH-directed routes. If every* Cont*-chain termination instance can be encoded as a route $R_{\mathsf{NameCert}_P,h} \in \mathcal{R}_{\mathsf{RH}}$ such that*

$$\mathsf{Halts}(\mathsf{NameCert}_P, h) \iff \mathsf{Closes}(R_{\mathsf{NameCert}_P,h}, \mathsf{RH}),$$

then no total classifier decides closure of all routes in $\mathcal{R}_{\mathsf{RH}}$.

Proof. A total classifier for closure of all routes in $\mathcal{R}_{\mathsf{RH}}$ would decide the right-hand side of the displayed equivalence. By the assumed encoding, it would decide arbitrary Cont-chain termination. This contradicts the halting boundary used in Definition 9336.11. Thus the requested classifier is a Type VII socket in the sense of Definition 9336.13, not an internal BEDC classifier. □

Theorem 9337.6 (Diagonal route classifier obstruction)**.** *If a route carrier is expressive enough to encode its own route interpreter and conditional branching, then it cannot carry an internal total classifier for its own closure predicate.*

Proof. Assume that such a classifier D exists. Construct a diagonal route Δ which, on input a route code r, queries $D(\bot, r)$ and then behaves oppositely at the target $\bot$: if the classifier says that r closes $\bot$, the diagonal route refuses to close $\bot$; if the classifier says that r does not close $\bot$, it closes $\bot$. Applying the construction to the code of Δ gives

$$D(\bot, \ulcorner\Delta\urcorner) = 1 \iff D(\bot, \ulcorner\Delta\urcorner) = 0,$$

which is impossible. Hence no such internal total classifier exists. □

9337.3 Fixed RH Independence Requires a Separate Meta-Theorem

Theorem 9337.7 (Route undecidability does not imply RH independence)**.** *The assertion*

$$\neg\exists D,\ D \text{ decides route closure}$$

does not imply

$$\mathsf{Indep}_F(\mathsf{RH})$$

for a named fragment F.

Proof. The first assertion is about the absence of a total classifier for a two-variable family of routes and targets. The second assertion is about the proof predicate of one fragment applied to one fixed formula. A system may lack a total route closure classifier while still proving, refuting, or leaving open a particular fixed proposition. For instance, a trivial fixed proposition such as $0 = 0$ can be proved in ordinary proof fragments even though global proof-route closure remains undecidable. Therefore the family-level halting obstruction does not entail a fragment-specific independence theorem for RH. □

Theorem 9337.8 (Fixed propositions do not carry halting families)**.** *There is no nonconstant many-one reduction of the halting family to the single proposition* RH *of the form*

$$\mathsf{Halts}(e, x) \iff \mathsf{RH}$$

for all program-input pairs (e, x)*.*

Proof. The right-hand side is independent of (e, x). If the displayed equivalence held for all (e, x), then $\mathsf{Halts}(e, x)$ would have the same truth value for all program-input pairs. The halting predicate is not constant. Thus the halting family can be encoded into a family of RH-directed routes, but not into the truth value of the single fixed proposition RH by such a reduction. □

Definition 9337.9 (RH-specific diagonal bridge)**.** *An RH-specific diagonal bridge for a fragment* F *is data consisting of the proof predicate of* F*, the code of* RH*, a transformation from any* F*-proof of* RH *to a contradiction, a transformation from any* F*-proof of* $\neg\mathsf{RH}$ *to a contradiction, and correctness proofs for both transformations.*

Theorem 9337.10 (Independence requires an RH-specific bridge)**.** *To prove* $\mathsf{Indep}_F(\mathsf{RH})$ *by diagonal means, it is necessary to supply an RH-specific diagonal bridge for* F*. A route closure undecidability theorem alone does not supply such a bridge.*

Proof. By Definition 9337.3, independence consists of two fragment-specific non-provability claims. A diagonal proof of those claims must transform any alleged F-proof of RH and any alleged F-proof of $\neg\mathsf{RH}$ into contradictions. That information is exactly the RH-specific diagonal bridge. The route closure theorem gives a contradiction only from a total classifier over a route family; it does not inspect the proof predicate of F at the fixed code of RH or of $\neg\mathsf{RH}$. □

9337.4 Finite Counterexample Constraint

Theorem 9337.11 (Finite counterexample constraint)**.** *Let a statement* A *be presented as*

$$\forall n > N,\ P(n),$$

where $P(n)$ *is a decidable arithmetic predicate, and let* F *be a fragment that verifies concrete computations of* P*. If* A *is false in the standard reading and a counterexample index* $n_0 > N$ *is displayed, then* F *proves* $\neg A$ *from the finite computation certificate for* $\neg P(n_0)$*.*

Proof. The displayed counterexample index reduces the negation of A to the finite computation $\neg P(n_0)$. Since P is decidable and the fragment verifies concrete computations of P, the finite certificate is accepted inside F. Therefore F proves $\neg P(n_0)$ and hence $\neg\forall n > N,\ P(n)$. □

Corollary 9337.12 (Arithmetic counterexample searches do not prove independence)**.** *Finite arithmetic counterexample interfaces such as chapter 9335 can supply constructive refutation data when a certified bad packet is emitted, and can supply safety targets when no accepted run emits such a packet. They do not by themselves supply* $\mathsf{Indep}_F(\mathsf{RH})$*.*

Proof. A certified bad packet is a finite counterexample datum, not an independence theorem. Absence of an emitted packet is a safety target for a recognized program run, not a proof that neither RH nor $\neg\mathsf{RH}$ is provable in a fragment. By Definition 9337.10, independence requires an RH-specific meta-theorem about the proof predicate of the chosen fragment. □

Corollary 9337.13 (Separation summary)**.** *The BEDC RH stack proves the following boundary separation: route closure classification is blocked by halting and diagonalization; sufficiently expressive RH route families inherit that blockage; the fixed proposition* RH *is not thereby proved independent of any fragment. A claim of RH independence must be carried by an RH-specific meta-theorem.*

Proof. The first claim is Definition 9337.4; the second is Definition 9337.5; the third is Definition 9337.7. The final sentence is Definition 9337.10. □

Concretized in §8824: The route/fixed-statement separation is realized as finite route-layer, statement-layer, refusal, transport, replay, provenance, and local NameCert data.

Chapter 9338

Dyadic Truncation Laguerre–Pólya Route Refutation

The completed zeta function admits the de Bruijn–Newman Fourier representation

$$\xi(z) = \int_0^\infty \Phi(u)\cos(zu)\,du,$$

with Φ the standard super-exponentially decaying kernel. This suggests a tempting finite-cutoff route to the Riemann Hypothesis: dyadically truncate the integral, show that every truncation lies in the Laguerre–Pólya class, and pass to the limit by Hurwitz's theorem. The present chapter records that this route does not close, because its central premise is false for sharp cutoffs. It does not prove or disprove the Riemann Hypothesis; it isolates, with a finite reproducible counterexample, why the route cannot be completed.

Definition 9338.1 (Sharp dyadic truncation)**.** *For a cutoff $a_K > 0$ write the sharp dyadic truncation*

$$\xi_K(z) := \int_0^{a_K} \Phi(u)\cos(zu)\,du,$$

and let κ_K denote the number of non-real zeros of ξ_K in the upper half plane. Since every Laguerre–Pólya entire function has only real zeros, any certificate $\kappa_K > 0$ refutes the all-K Laguerre–Pólya premise for this sharp-cutoff family.

Principle 9338.2 (The conditional Hurwitz bridge)**.** *Since $\xi_K \to \xi$ uniformly on compact sets and the Laguerre–Pólya class is closed under locally uniform limits, the implication*

$$(\forall K,\ \kappa_K = 0) \ \Rightarrow\ \xi \in \mathrm{LP} \ \Rightarrow\ \mathsf{RH}$$

is valid. The implication is sound, and that is precisely why it carries no content on its own: all of the difficulty is shifted into its premise.

Principle 9338.3 (The premise is false: a finite counterexample)**.** *The premise $\forall K,\ \kappa_K = 0$ fails. The reproducible experiment* `papers/bedc/experiments/dyadic_truncation_lp_refutation.py` *exhibits non-real zeros of small-cutoff truncations ξ_K, each certified by an argument-principle winding count together with a Newton residual check, so that $\kappa_K > 0$ for those cutoffs. The all-K Laguerre–Pólya premise therefore cannot hold, and the route above cannot be discharged.*

Principle 9338.4 (Forced bifurcation mechanism)**.** *The failure is structural rather than incidental. Write the boundary-corrected truncation as a homotopy $F_t = Q_K + tP_K$ running from a Robin characteristic function Q_K, whose zeros are all real and simple, to the truncation at $t = 1$. For large cutoffs the first Robin zeros cluster near the origin and move under the homotopy with opposite-sign velocities, so they are forced to collide before the endpoint, producing a real double zero from which a non-real pair is born. Hence the no-bifurcation sub-route is impossible: real-rootedness of the limit cannot be secured merely by keeping every truncation inside the Laguerre–Pólya class, and any salvage would require a separate birth-and-death cancellation argument across the homotopy.*

Principle 9338.5 (Scope and provenance of the refutation)**.** *The obstruction is that a sharp domain cutoff lacks the Laguerre–Pólya preservation enjoyed by the smooth de Bruijn–Newman heat deformation $e^{tu^2}\Phi$. The same failure mode is classical: the proven failure of the Turán partial-sum route, the Szegő-curve behaviour of the partial sums of the exponential whose zeros stay complex even though the limit is zero-free, and the insensitivity of Jensen-polynomial hyperbolicity to off-line zeros. The route is recorded here as not-claimed. It makes no positive or negative statement about the Riemann Hypothesis itself; it removes one tempting but unsound bridge so that effort is not spent re-deriving it.*

Concretized in §9025: Sharp dyadic truncation refutation windows realize the Laguerre–Polya route obstruction as finite cutoff, zero-count, refusal, transport, replay, provenance, and NameCert rows.

Chapter 9339

A Constructive Winding-Certificate Layer for the Argument-Principle Route

The argument-principle route at finite height separates into two obligations. The topological and combinatorial half defines a winding number for rational boundary polygons and proves that this number is well defined under the certificate operations used by a finite verifier. The analytic half proves that such a polygon faithfully tracks $\zeta(\partial R)$ and that the constructive argument-principle theorem identifies the resulting winding with the number of zeros inside R. Only the topological half is closed here.

The present layer therefore does not prove the Riemann Hypothesis, does not locate every zeta zero, and does not supply the analytic bridge from $\zeta(\partial R)$ to a rational polygon. It records the completed mathlib-free, 0-axiom, propext-free Lean certificate layer in the namespace BEDC.Derived.RHRoute.RationalPolygonWinding and its sibling modules. The layer works with finite rational vertices and the signed crossing count of the positive real ray. Its 'Up' versions use the BEDC integer carrier so that the algebra remains inside the same constructive kernel fragment.

Definition 9339.1 (Rational-polygon winding certificate layer)**.** *A rational-polygon winding certificate consists of a finite cyclic list of rational complex vertices, the oriented edge list obtained from that cycle, and the signed sum of edge crossings of the positive real ray. The ordinary integer count is written as* computedWinding*, while the BEDC-integer count is written as* computedWindingUp*. A certificate generator may perturb rational vertices within certified crossing margins, refine an edge by inserting an intermediate rational point, rotate the starting vertex, reverse edge orientations, and split a rectangular boundary into adjacent boxes.*

Principle 9339.2 (Perturbation stability)**.** *If every edge of a rational polygon carries a margin certificate keeping it away from the crossing discriminants, then any edgewise rational perturbation preserving those margins has the same computed winding. This is the stability fact used when a finite evaluator replaces a boundary trace by nearby rational vertices without changing the topological certificate.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonWindingStability.computedWinding stable of edgewiseMarginPr`

Principle 9339.3 (Collinear edge subdivision)**.** *Inserting a rational point strictly between the endpoints of a segment splits the segment into two collinear directed segments without changing the total crossing contribution of that edge. This is the local algebraic fact behind boundary refinement.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonEdgeSubdivision.edgeCrossing collinear split`

Principle 9339.4 (Surface-level subdivision invariance)**.** *At the summed BEDC-integer level, replacing one edge in an oriented edge list by its two collinear subedges preserves the total crossing sum. Thus adaptive refinement changes the mesh of the polygon, not its winding certificate.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonEdgeSubdivision.sumEdgeCrossingsUp subdivide one`

Principle 9339.5 (Cyclic rotation invariance)**.** *The chosen starting vertex of a nonempty cyclic vertex list is not part of the certificate content. Rotating the list by one step preserves the BEDC-integer winding count.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonWinding.computedWindingUp cyclic rotate one`

Principle 9339.6 (Orientation reversal cancellation)**.** *Traversing the same rational edge in the two opposite directions contributes opposite signed crossings, so the two directed copies have net contribution* 0 *in the BEDC integer carrier.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonWinding.edgeCrossingUp reverse cancel`

Principle 9339.7 (Box-split additivity)**.** *When a rectangular boundary is split into adjacent boxes, the shared interior edge is traversed once in each direction and cancels. The resulting crossing sum is additive across the split, giving the combinatorial skeleton for recursive box certificates.*

Lean4 ✓ `BEDC.Derived.RHRoute.RationalPolygonWinding.sumEdgeCrossingsUp box split`

Together these checked facts say the following precise thing: for rational boundary polygons, the winding number is a finite combinatorial object that is well defined, stable under certified perturbation, invariant under edge subdivision and cyclic rotation, antisymmetric under orientation reversal, and additive under box splitting. These are the structural properties needed by the topological side of an argument-principle certificate, and they are all kernel-verified in the 0-axiom BEDC fragment. They are not a zero-counting theorem for ζ.

Principle 9339.8 (Open analytic bridges)**.** *Two analytic obligations remain open. First, the route needs a located interval evaluator for ζ on ∂R. The intended evaluator passes through the eta series*

$$\eta(s) = \sum_{n\geq 1} (-1)^{n-1} n^{-s}, \qquad \zeta(s) = \frac{\eta(s)}{1-2^{1-s}},$$

with

$$n^{-s} = \exp(-s\ln n), \qquad n^{-it} = \cos(t\ln n) - i\sin(t\ln n).$$

The critical point is the tail: a naive Leibniz estimate does not directly apply because the phase n^{-it} rotates in the complex plane. A strict Euler–Boole alternating-summation bound is the concrete 0-axiom shape needed for the critical strip, including derivative bounds for interval-Newton or Krawczyk certificates.

Second, the route needs the constructive argument-principle theorem itself: a discrete Rouché or faithful-approximation theorem proving that a rational polygon whose certified tube contains $\zeta(\partial R)$ has the same winding as the true boundary trace and therefore counts zeros inside R. These are multi-week constructive-analysis obligations, not completed certificate payload.

A usable Euler–Boole obligation can be stated without hiding the analytic work. For a box

$$B = \{s = \sigma + it : \sigma_- \le \sigma \le \sigma_+,\ t_- \le t \le t_+\}, \qquad 0 < \sigma_- < \sigma_+ < 1,$$

set $X = N + 1$ and $f_s(x) = x^{-s}$. With Euler polynomials defined by

$$\frac{2e^{xz}}{e^z + 1} = \sum_{\nu \ge 0} E_\nu(x) \frac{z^\nu}{\nu!},$$

Euler–Boole summation gives endpoint terms plus a remainder

$$R_N(s) = (-1)^N \left[\frac{1}{2} \sum_{\nu=0}^{2L-1} \frac{E_\nu(0)}{\nu!} f_s^{(\nu)}(X) + \mathcal{E}_L(s) \right],$$

where

$$f_s^{(\nu)}(X) = (-1)^\nu (s)_\nu X^{-s-\nu}, \qquad E_{2j}(0) = 0 \quad (j \ge 1).$$

A constructive interval evaluator must turn this into a uniform complex-box bound for $\mathcal{E}_L(B)$ and for $\partial_s \mathcal{E}_L(B)$, then propagate those enclosures through $\zeta(s) = \eta(s)/(1 - 2^{1-s})$. This is the analytic bottleneck shared by the winding and local-zero routes.

Principle 9339.9 (Located-zero complement)**.** *The local alternative is to locate a single zero by a Krawczyk or interval-Newton certificate. Near the first nontrivial zero, $c \approx 1/2 + 14.1347i$, one chooses a high-precision approximate inverse $a \approx 1/\zeta'(c)$ and studies the Newton-like map*

$$N(z) = z - a\,\zeta(z).$$

For a closed complex box B around c, the constructive target has the form

$$N(B) \subseteq B, \qquad \sup_{z \in B} |1 - a\zeta'(z)| < 1.$$

This traps a unique zero in B. It is a local located-zero certificate, not a global statement about all zeros. It is complementary to the rational-polygon winding layer because both legs need the same located-ζ interval evaluator and the same Euler–Boole complex tail control.

This should also be read against the route separation in chapter 9338. The sharp dyadic Laguerre–Pólya premise fails as a global preservation strategy. The present chapter does not try to revive that premise. It keeps the finite topological certificate layer and points the analytic work toward located ζ evaluation, Krawczyk-style local boxes, and a constructive argument-principle bridge.

Closure status for NameCert$_{\text{ConstructiveWindingCertificate}^\uparrow}$.

Constructive story: A ConstructiveWindingCertificate$^\uparrow$ packet is generated from finite rational boundary vertices, oriented rational edges, signed positive-ray crossing rows, perturbation-margin rows, subdivision rows, cyclic-rotation rows, reversed-edge cancellation rows, and box-split rows, with all exported structure witnessed by the listed Lean targets.

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveWindingCertificate}^\uparrow}$, scopedClosed)

Scope closed: Rational-boundary-polygon winding number's combinatorial certificate layer, including well-definedness, stability, subdivision and cyclic-rotation invariance, orientation cancellation, and box additivity; excluding located-zeta evaluation and the analytic argument-principle theorem

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveWindingCertificate}^\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.RHRoute.RationalPolygonWinding.sumEdgeCrossingsUp\_box\_split`

Bridge status: none

Not claimed: Not closed: the located-zeta interval evaluator and the constructive argument-principle theorem identifying winding with zero count

Upgrade path: Add located complex exp/ln/trig continuity evaluators plus discrete Rouché or faithful-approximation error control

Origin: ai

Chapter 9340

Contour Integration as an Operation on $\mathsf{Holomorphic}^{\uparrow}$

This chapter licenses contour integration $\int_\gamma f\,dz$ along a piecewise-rational-linear contour γ inside the holomorphic domain of f. Unlike the naming-certificate chapters that pin down a closed object (a number system, an algebraic carrier, a topological structure), this chapter licenses an *operation*: a constructive procedure that turns a (contour, holomorphic-function) pair into a complex-history value. The chapter sits above $\mathsf{Holomorphic}^{\uparrow}$ (chapter 68) and $\mathsf{CFun}^{\uparrow}$ (chapter 59). Following the Bishop-style discipline of the rest of the paper, the carrier of contours is restricted to piecewise-linear paths over $\mathsf{Rat}^{\uparrow}$ endpoints with explicit Riemann-sum modulus, and the integral is the unique complex-history value approached by these sums up to an effective rate of convergence. The chapter ends with an *operation certificate* — an analogue of the naming certificate adapted to operations, with input/output classifier-aware predicate plus operation laws (linearity, concatenation, reversal, reparameterization) in place of the standard 5-field NameCert tuple.

9340.1 Piecewise-linear contour carrier

Definition 9340.1 (Piecewise-linear contour)**.** *A piecewise-linear contour γ is a finite list $[z_0, z_1, \ldots, z_k]$ of complex-history endpoints with $k \geq 1$, each z_i classified by $\mathsf{Complex}^{\uparrow}$. The closed inductive predicate $\mathsf{PLContour}(\gamma)$ is generated by:*

1. $\mathsf{PLContour}([z_0, z_1])$ *for any pair of complex histories (the singleton segment);*
2. *if* $\mathsf{PLContour}([z_0, \ldots, z_k])$ *and z_{k+1} is a complex history, then* $\mathsf{PLContour}([z_0, \ldots, z_k, z_{k+1}])$ *(extension by one further segment).*

The single-segment BEDC carrier records two complex-history endpoints z_0, z_1 and the continuation witness $\mathsf{Cont}(z_0, z_1, \gamma)$. The total length is the sum of the segment lengths $|z_{i+1} - z_i|$, each computed via the metric carrier of $\mathsf{Complex}^{\uparrow}$ (chapter 39).

Lean4 (definition) `BEDC.Derived.ContourIntegralUp.PLContourSegment`

Lemma 9340.2 (Piecewise-linear contour well-formedness)**.** *For every BEDC single-segment* $\mathsf{PLContourSegment}(\gamma$ *there are complex-history endpoints z_0, z_1 with $\mathsf{Cont}(z_0, z_1, \gamma)$, and the contour history γ is unary.*

Proof. Destructure the segment witness. The endpoint carrier fields give z_0 and z_1 as complex histories. Each complex history is unary, and the continuation witness closes unary histories under Cont, so γ is unary. □

Lean4 ✓ `BEDC.Derived.ContourIntegralUp.PLContourSegment unary result`

Lemma 9340.3 (Piecewise-linear segment hsame-transport)**.** *If* $\mathsf{PLContourSegment}(\gamma)$ *and* $\mathsf{hsame}(\gamma, \gamma')$*, then* $\mathsf{PLContourSegment}(\gamma')$*, and* γ' *is unary.*

Proof. Keep the same endpoints z_0, z_1 and transport the continuation result from γ to γ' along $\mathsf{hsame}(\gamma, \gamma')$. Unary closure for γ' follows from the same endpoint carriers and the transported continuation. □

Concretized in §583: Contour-integral sockets are realized as finite BHist operation packets for contour rows, holomorphic input rows, Riemann-sum modulus rows, ComplexUp output rows, and non-escape ledger obligations.

Lean4 ✓ `BEDC.Derived.ContourIntegralUp.PLContourSegment hsame transport`

Lemma 9340.4 (Concatenation of piecewise-linear contours)**.** *If* $\mathsf{PLContour}([z_0, \ldots, z_j])$ *and* $\mathsf{PLContour}([w_0, \ldots, w$ *and* $\mathsf{hsame}(z_j, w_0)$*, then* $\mathsf{PLContour}([z_0, \ldots, z_j, w_1, \ldots, w_k])$ *and the total length of the concatenation equals the sum of the two component lengths.*

Lean4 ✓ `BEDC.Derived.ContourIntegralUp.PLContour concat closed`

Proof. By induction on the second contour, using the extension constructor of $\mathsf{PLContour}$ at each step. Length additivity follows by direct computation. □

9340.2 Riemann sum approximation

Definition 9340.5 (Subdivided Riemann sum)**.** *For a function* f *classified by* $\mathsf{CFun}^{\uparrow}$ *on a domain containing* γ*, a piecewise-linear contour* $\gamma = [z_0, \ldots, z_k]$*, a unary subdivision count* n*, and an explicit choice of sampling points* $S_n = \{\xi_{i,j}\}$ *on each subsegment, the Riemann sum* $\mathsf{RiemannSum}(f, \gamma, n, S_n)$ *is the complex-history value*

$$\sum_{i=0}^{k-1} \sum_{j=0}^{n-1} f(\xi_{i,j}) \cdot \frac{z_{i+1} - z_i}{n}.$$

The closed inductive carrier indexes both the segment and the subdivision step.

Lemma 9340.6 (Riemann sum is a well-defined complex history)**.** *For every* $\mathsf{PLContour}(\gamma)$*, continuous* f *on a domain containing* γ*, unary* n*, and sampling choice* S_n*, the value* $\mathsf{RiemannSum}(f, \gamma, n, S_n)$ *is classified by* $\mathsf{Complex}^{\uparrow}$*.*

Proof. By induction on the length of the contour list. The base case is a single sum of n products of complex histories, each well-defined by the multiplicative closure of $\mathsf{Complex}^{\uparrow}$ (chapter 39). The step adds one further segment's contribution to a previously well-defined complex history; closure under finite sums of $\mathsf{Complex}^{\uparrow}$ preserves the classification. □

9340.3 The contour integral

Definition 9340.7 (Contour integral predicate)**.** *For a function* f *classified by* $\mathsf{Holomorphic}^{\uparrow}$ *on a neighbourhood containing* γ*, a piecewise-linear contour* γ*, and a complex history* I*, the predicate* $\mathsf{ContInt}(f, \gamma, I)$ *holds when there is an explicit modulus function* $N : \mathsf{Unary} \to \mathsf{Unary}$ *such that for every unary precision* k *and every* n *with* $\mathsf{NatGT}(n, N(k))$ *and every sampling choice* S_n*,*

$$\left|\mathsf{RiemannSum}(f, \gamma, n, S_n) - I\right| \leq 2^{-k}.$$

Lemma 9340.8 (Existence of the contour integral on holomorphic inputs)**.** *For every* $\mathsf{PLContour}(\gamma)$ *and every* f *classified by* $\mathsf{Holomorphic}^{\uparrow}$ *on an open neighbourhood of (the image of)* γ*, there exists a complex history* I *with* $\mathsf{ContInt}(f, \gamma, I)$*.*

Sketch. The holomorphicity classifier of f on the neighbourhood provides an explicit local modulus of continuity. Combining this with the bounded total length of γ (Definition 9340.2) yields a uniform Cauchy modulus on the Riemann-sum sequence indexed by n; its limit I is constructed as a complex history by the Cauchy completeness of $\mathsf{Complex}^{\uparrow}$ (chapter 39). □

Theorem 9340.9 (Uniqueness of the contour integral up to hsame)**.** *If* $\mathsf{ContInt}(f, \gamma, I_1)$ *and* $\mathsf{ContInt}(f, \gamma, I_2)$*, then* $\mathsf{hsame}(I_1, I_2)$*.*

Proof. For each unary k, the Riemann sum at precision $\max(N_1(k+1), N_2(k+1))$ is within $2^{-(k+1)}$ of both I_1 and I_2, so $|I_1 - I_2| \leq 2^{-k}$ for every k. Hence $|I_1 - I_2|$ is the zero history, and $\mathsf{hsame}(I_1, I_2)$ follows from the metric reflection of $\mathsf{Complex}^{\uparrow}$. □

9340.4 Operation laws

Lemma 9340.10 (Linearity in the integrand)**.** *If* $\mathsf{ContInt}(f, \gamma, I_f)$ *and* $\mathsf{ContInt}(g, \gamma, I_g)$ *and* a, b *are complex histories, then* $\mathsf{ContInt}(af + bg, \gamma, aI_f + bI_g)$*.*

Proof. Riemann sums are linear in the integrand, and the bound on $|aI_f + bI_g - \mathsf{RiemannSum}(af + bg, \gamma, n, S_n)|$ follows by the triangle inequality from the bounds on the two component Riemann sums. □

Lemma 9340.11 (Concatenation in the contour)**.** *If* γ *is the concatenation of* γ_1 *and* γ_2 *(with shared endpoint via Definition 9340.4) and* $\mathsf{ContInt}(f, \gamma_1, I_1)$ *and* $\mathsf{ContInt}(f, \gamma_2, I_2)$*, then* $\mathsf{ContInt}(f, \gamma, I_1 + I_2)$*.*

Proof. A Riemann sum on the concatenated contour is the sum of the Riemann sums on the two component contours; pass to the joint modulus $\max(N_1(k+1), N_2(k+1))$. □

Lemma 9340.12 (Reversal negates the integral)**.** *Let* $\gamma^- = [z_k, z_{k-1}, \ldots, z_0]$ *be the reverse of* $\gamma = [z_0, \ldots, z_k]$*. If* $\mathsf{ContInt}(f, \gamma, I)$*, then* $\mathsf{ContInt}(f, \gamma^-, -I)$*.*

Proof. Each Riemann-sum segment $f(\xi_{i,j}) \cdot (z_{i+1} - z_i)/n$ flips sign under reversal because $z_{i+1} - z_i$ is replaced by $z_i - z_{i+1}$. The modulus is unchanged. □

Lemma 9340.13 (Reparameterization invariance for piecewise-linear reparameterizations)**.** *If* γ' *is obtained from* γ *by inserting one additional intermediate endpoint along an existing segment (a piecewise-linear refinement), then* $\mathsf{ContInt}(f, \gamma, I)$ *iff* $\mathsf{ContInt}(f, \gamma', I)$*.*

Proof. Refinement decomposes a single segment's Riemann contribution into a sum over the two refined sub-segments; this leaves both the limit I and the modulus unchanged up to an additive constant in k. □

9340.5 Cauchy's theorem and the integral formula

Theorem 9340.14 (Cauchy's theorem on triangulated regions)**.** *If γ is a closed piecewise-linear contour bounding a triangle inside the holomorphic domain of f, then* $\mathsf{ContInt}(f, \gamma, 0)$*, where* 0 *denotes the zero complex history.*

Sketch. The standard Goursat-style nested-triangle subdivision argument: bisect the triangle into four congruent sub-triangles, observe that the sum of the four sub-integrals equals the original integral (cancellations along shared edges by Definition 9340.12), pick the largest sub-integral and iterate. Holomorphicity provides the bound that the integral over the n-th nested triangle decays like $4^{-n} \cdot (\text{perimeter})^2$, forcing the limit to be zero. □

Lemma 9340.15 (Cauchy on triangulated polygons)**.** *For any closed piecewise-linear γ bounding a region triangulated by finitely many triangles inside the holomorphic domain of f,* $\mathsf{ContInt}(f, \gamma, 0)$*.*

Proof. Apply Definition 9340.14 to each triangle in the triangulation. The sum of the triangle integrals equals the integral over the boundary of the polygon by Definition 9340.11 and Definition 9340.12; the interior edges cancel pairwise. □

Theorem 9340.16 (Cauchy integral formula, constructive form)**.** *Let z_0 be a complex history, r a positive rational history, and γ a closed piecewise-linear contour approximating the circle of radius r about z_0 inside the open disk* $\mathsf{OpenDisk}(z_0, r)$*. For f classified by* $\mathsf{Holomorphic}^{\uparrow}$ *on the disk,*

$$\mathsf{ContInt}(\lambda z.f(z)/(z - z_0), \gamma, 2\pi i \cdot f(z_0))$$

holds, with explicit modulus computed from the rational approximation of π provided by $\mathsf{Real}^{\uparrow}$ *(chapter 38).*

Sketch. By Cauchy's theorem on triangulated regions, the integral over γ equals the integral over a small piecewise-linear circle approximation around z_0 (the difference contour bounds a triangulated annulus). On the small approximation, the integrand is approximately $f(z_0)/(z - z_0)$, whose contour integral is $2\pi i \cdot f(z_0)$ by direct computation on the rational endpoints. The explicit modulus combines the holomorphicity modulus with the rational-π approximation modulus. □

9340.6 Operation laws and ledger

Definition 9340.17 (Contour-integral input specification)**.** *The input specification of the contour-integral operation is the pair (f, γ) where f is classified by* $\mathsf{Holomorphic}^{\uparrow}$ *on an open neighbourhood and γ is classified by* $\mathsf{PLContour}$ *inside that neighbourhood.*

Definition 9340.18 (Contour-integral output specification)**.** *The output specification is a single complex history I classified by* $\mathsf{Complex}^{\uparrow}$*, uniquely determined up to* hsame *by Definition 9340.9.*

Definition 9340.19 (Contour-integral classifier)**.** *The classifier of the operation is* hsame *on the output complex history. Two outputs are classified together iff they are* hsame*-equal.*

Definition 9340.20 (Contour-integral operation laws)**.** *The operation laws of the contour-integral operation collect:*

1. *linearity in the integrand (Definition 9340.10);*
2. *concatenation in the contour (Definition 9340.11);*

3. *reversal of the contour negates the integral (Definition 9340.12);*
4. *invariance under piecewise-linear reparameterization (Definition 9340.13).*

Definition 9340.21 (Contour-integral ledger policy)**.** *The ledger policy records:*

1. *Cauchy closure: closed contours bounding triangulated regions inside the holomorphic domain produce zero (Definition 9340.14, Definition 9340.15);*
2. *Cauchy integral formula composition (Definition 9340.16): point evaluation factors through a rational-approximated circle integral.*

Theorem 9340.22 (Concrete contour-integral operation certificate)**.** *The data* (ContIntInputSpec, ContIntOutputSp *of Definition 9340.17–Definition 9340.21 forms an operation certificate over* Hist *for contour integration on* $\mathsf{Holomorphic}^{\uparrow}$*. The operation is total on its input specification (Definition 9340.8), uniquely determined up to* hsame *(Definition 9340.9), and closed under the four operation laws.*

Sketch. Totality is Definition 9340.8. Uniqueness is Definition 9340.9. The operation laws are Definition 9340.10, Definition 9340.11, Definition 9340.12, and Definition 9340.13. The ledger policy fields are Definition 9340.14, Definition 9340.15, and Definition 9340.16. The certificate uses no Classical.choice, no Quot.sound, and no propext, so the operation's Lean targets fall under the project's axiom-purity gate. □

Theorem 9340.23 (Semantic contour-integral operation certificate)**.** *The concrete operation certificate of Definition 9340.22 satisfies the semantic operation-certificate schema: the input and output specifications are* Hist*-predicates, the classifier is a restriction of* hsame*, and every operation law is given by an explicit constructive proof. The operation is functorial in its input pair under* hsame*: if* $\mathsf{hsame}(f, f')$ *(in the sense of pointwise classifier on the holomorphic carrier) and* $\mathsf{hsame}(\gamma, \gamma')$ *(in the sense of endpoint-wise classifier on* PLContour*), then the resulting integrals are* hsame*-equal.*

Sketch. Direct verification of the schema fields using the constructions of this chapter and the hsame-respect of the underlying $\mathsf{Holomorphic}^{\uparrow}$ and $\mathsf{Complex}^{\uparrow}$ classifiers. □

Lemma 9340.24 (Affine concatenation closure for contour integrals)**.** *Let* $\gamma = \gamma_1 * \gamma_2$ *be the concatenation of* γ_1 *and* γ_2*, with the shared endpoint condition supplied by Definition 9340.4. Suppose that*

$$\mathsf{ContInt}(f, \gamma_1, I_1^f), \quad \mathsf{ContInt}(f, \gamma_2, I_2^f), \quad \mathsf{ContInt}(g, \gamma_1, I_1^g), \quad \mathsf{ContInt}(g, \gamma_2, I_2^g)$$

hold. If a *and* b *are complex histories, then*

$$\mathsf{ContInt}(af + bg, \gamma, a(I_1^f + I_2^f) + b(I_1^g + I_2^g)).$$

Proof. Apply Definition 9340.11 to the two f-integrals along γ_1 and γ_2. This gives

$$\mathsf{ContInt}(f, \gamma, I_1^f + I_2^f).$$

Apply the same concatenation law to the two g-integrals, giving

$$\mathsf{ContInt}(g, \gamma, I_1^g + I_2^g).$$

Now apply linearity in the integrand, Definition 9340.10, to these two contour integrals on the same contour γ with scalar histories a and b. The resulting certified value is exactly $a(I_1^f + I_2^f) + b(I_1^g + I_2^g)$, which is the required contour integral for $af + bg$ along γ. □

Concretized in §521: Contour-integral operation packets realize piecewise-linear contour rows, Riemann-sum ledgers, uniqueness boundary, and operation-law public export as a finite NameCert surface.

Concretized in §587: Contour integration is realized as finite contour, subdivision, Riemann-sum, output, transport, continuation, package, and NameCert rows.

Concretized in §704: Contour affine-concatenation packets realize the shared-endpoint concatenation and affine-linearity operation law as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1019: Contour reversal ledgers realize the piecewise-linear contour reversal law as a finite BHist NameCert packet with contour, reversed-contour, integral-output, transport, route, provenance, and local naming rows.

Concretized in §7037: Contour reparameterization ledgers realize the contour-integral operation law as finite PL-contour, holomorphic-input, Riemann-sum modulus, ComplexUp output, transport, replay, provenance, and NameCert rows.

Concretized in §3978: The residue handoff is realized as a finite contour-operation consumer packet over PLContour, HolomorphicUp, ContourIntegralOperationUp, ResidueUp, and non-escape rows.

Chapter 9341

Analytic Continuation as an Operation on $\mathsf{Holomorphic}^{\uparrow}$

This chapter licenses analytic continuation as the operation $\mathsf{AnaCont}^{\uparrow}$ that, given a holomorphic f on a domain D_1 and an overlapping holomorphic g on D_2 with $f = g$ on a non-trivial overlap region U, produces the unique holomorphic extension h on $D_1 \cup D_2$. The chapter is an OPERATION chapter rather than a pure namecert: the structure is input/output classifier-aware predicate plus operation laws (functoriality, composition, hsame-respect) instead of the standard 5-field NameCert. Builds on $\mathsf{Holomorphic}^{\uparrow}$ (chapter 68) and the constructive identity theorem of that chapter. Branch obstruction (the constructive face of monodromy / multi-valued continuation around branch points) is flagged as a SCHEMA-ONLY HORIZON: the chapter exposes the constructive interface but admits that closed witnesses for general branch behaviour do not in general live inside the framework. The construction is fully constructive on the parts that close; uses no Classical.choice, no Quot.sound, and no propext, so the operation's Lean targets fall under the project's axiom-purity gate.

Concretized in §569: Analytic continuation is realized as a finite BHist operation-socket NameCert packet over holomorphic source rows, overlap witnesses, continuation outputs, and branch-obstruction non-escape.

Concretized in §577: Analytic continuation operations are realized as classifier-stable BEDC operation packets over holomorphic inputs, overlap evidence, and continuation ledgers.

Concretized in §694: Branch-obstruction packets realize branch-sensitive continuation requests as finite BHist ledger rows whose nonlocal reads do not export branch selectors, quotient germs, monodromy theorems, or new holomorphic outputs.

Concretized in §1060: Schwarz reflection routes realize boundary-symmetric analytic continuation as finite HolomorphicUp, ComplexDiffUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §312: Monodromy packets realize the continuation-around-a-loop boundary as finite loop, base endpoint, local-system or fibre, returned-row, transport, provenance, and NameCert rows.

9341.1 Domain compatibility

Definition 9341.1 (Domain compatibility)**.** *For two open domains D_1, D_2 of $\mathsf{Complex}^{\uparrow}$, the predicate $\mathsf{DomCompat}(D_1, D_2, U)$ is satisfied by an explicit overlap region U which contains an*

$\mathsf{OpenDisk}(z_c, r_c)$ *inside* $D_1 \cap D_2$ *together with a quantitative density bound at a center point* z_c *(i.e. a sequence of points* $(z_n)_n$ *clustering at* z_c *inside* U *with* $|z_n - z_c| \leq 2^{-n}$*). The density witness is required for the constructive identity theorem (chapter 68) to apply.*

Lemma 9341.2 (Domain compatibility witness)**.** *The predicate* $\mathsf{DomCompat}$ *is decidable when* D_1, D_2 *are concrete unions of open disks with rational centers and radii.*

Proof. For unions of open disks, intersection is decidable from radii comparisons; the inner disk witness is obtained explicitly. □

9341.2 The continuation predicate

Definition 9341.3 (Analytic continuation predicate)**.** *For* f *holomorphic on* D_1*,* g *holomorphic on* D_2*, with* $\mathsf{DomCompat}(D_1, D_2, U)$*, the predicate* $\mathsf{AnaCont}(f, D_1, g, D_2, h)$ *holds when* h *is a holomorphic function on* $D_1 \cup D_2$ *such that:*

1. h *restricted to* D_1 *equals* f *under* hsame *pointwise;*
2. h *restricted to* D_2 *equals* g *under* hsame *pointwise;*
3. f *and* g *agree on the overlap region* U *under* hsame *pointwise (a precondition for existence).*

Theorem 9341.4 (Continuation is unique)**.** *If* $\mathsf{AnaCont}(f, D_1, g, D_2, h)$ *and* $\mathsf{AnaCont}(f, D_1, g, D_2, h')$ *both hold, then* $h = h'$ *under* hsame *pointwise on* $D_1 \cup D_2$*.*

Sketch. Constructive identity theorem (chapter 68) applied at the cluster sequence in U: both h and h' agree with f and g on dense subsets, hence on the union. □

9341.3 Chain continuation

Theorem 9341.5 (Chain continuation)**.** *If* $\mathsf{AnaCont}(f_1, D_1, f_2, D_2, h_{12})$ *and* $\mathsf{AnaCont}(h_{12}, D_1 \cup D_2, f_3, D_3, h_{123})$ *hold (with* $\mathsf{DomCompat}(D_1 \cup D_2, D_3, U_{23})$*), then composing the two continuations gives a single* $\mathsf{AnaCont}$ *witness on* $D_1 \cup D_2 \cup D_3$ *with explicit modulus inheritance.*

Sketch. Direct composition of holomorphic restrictions; uniqueness from Definition 9341.4 applied at each step. □

9341.4 Branch obstruction (schema-only horizon)

Definition 9341.6 (Branch obstruction)**.** *When the chain continuation closes a loop around a topological obstruction (e.g. continuing* $\sqrt{z}$ *along a circle around the origin), the resulting two extensions through different paths can disagree. The branch obstruction predicate* $\mathsf{BranchObstruction}$ *records this disagreement as a SCHEMA-ONLY structure: it codifies the constructive interface (input: two chain continuations; output: a quantitative non-agreement witness) without producing closed witnesses for general topological branches inside the BEDC framework.*

Remark 9341.7 (Schema-only nature of branch obstruction)**.** *The branch obstruction interface is intentionally a SCHEMA-ONLY HORIZON: BEDC can express the obstruction predicate, can produce concrete witnesses on specific examples (*$\sqrt{z}$*,* $\log z$ *around 0), but cannot in general decide whether a given chain configuration produces obstruction witnesses. This is consistent with the BEDC project's general schema-only horizon protocol — the chapter exposes the interface so future framework extensions can codify the closed witnesses, while the current chapter holds the schema open.*

9341.5 Concrete continuations

Theorem 9341.8 (Schwartz reflection)**.** *If f is holomorphic on the upper half-plane $\{z : \Im(z) > 0\}$ and continuous on the closure including a real boundary segment, with real values on the real boundary, then the function $\tilde{f}(z) = \overline{f(\bar{z})}$ defines a holomorphic continuation across the real boundary into the lower half-plane. The continuation is given as an explicit function with explicit modulus.*

Sketch. Direct verification using the Cauchy-Riemann equations (chapter 67) and the symmetry of the boundary trace. □

9341.6 Operation laws

Lemma 9341.9 (Functoriality)**.** AnaCont *respects* hsame *on inputs: if* hsame *on f, f' and on g, g' (pointwise on the respective domains), then* $\mathsf{AnaCont}(f, D_1, g, D_2, h)$ *implies* $\mathsf{AnaCont}(f', D_1, g', D_2, h)$ *for the same h up to* hsame *pointwise.*

Proof. Direct transport of equality witnesses under hsame. □

9341.7 The operation certificate

Definition 9341.10 (Analytic continuation operation certificate)**.** *The operation certificate consists of: (i) input spec — pairs (f, D_1) and (g, D_2) with overlap witness* $\mathsf{DomCompat}(D_1, D_2, U)$*; (ii) output spec — holomorphic h on $D_1 \cup D_2$ restricting compatibly; (iii) classifier —* hsame *pointwise on the union; (iv) operation laws — functoriality, chain composition, schwartz reflection as a concrete instance, and branch obstruction as the schema-only horizon; (v) ledger policy — uniqueness and existence under domain compatibility.*

Theorem 9341.11 (Concrete $\mathsf{AnaCont}^{\uparrow}$ operation certificate)**.** *The data above forms an operation certificate on* $\mathsf{Holomorphic}^{\uparrow}$ *for the analytic-continuation interface* $\mathsf{AnaCont}^{\uparrow}$*.*

Sketch. Each operation law is verified by direct construction in the preceding sections; the schema-only branch obstruction is exposed as a horizon, not closed. The certificate uses no Classical.choice, no Quot.sound, and no propext, so the operation's Lean targets fall under the project's axiom-purity gate. □

Theorem 9341.12 (Semantic $\mathsf{AnaCont}^{\uparrow}$ operation certificate)**.** *The concrete certificate of Definition 9341.11 satisfies the semantic operation-certificate schema obtained by adapting* NameCert *(chapter 9) to operation inputs/outputs.*

Sketch. Direct verification of the schema fields using the constructions of this chapter and the hsame-respect of the underlying $\mathsf{Holomorphic}^{\uparrow}$ classifier. □

Lemma 9341.13 (Domain compatibility is symmetric)**.** *If* $\mathsf{DomCompat}(D_1, D_2, U)$ *holds, then* $\mathsf{DomCompat}(D_2, D_1, U)$ *holds for the same overlap region U.*

Proof. Unpack Definition 9341.1. The displayed witness consists of an open disk $\mathsf{OpenDisk}(z_c, r_c)$ contained in $D_1 \cap D_2$ and a quantitative sequence in U clustering at z_c with the stated unary bound. The same center, radius, open-disk witness, overlap region, sequence, and density estimates witness compatibility in the reversed order. Only the containment record changes: membership in $D_1 \cap D_2$ is transported to membership in $D_2 \cap D_1$ by commutativity of intersection. □

Theorem 9341.14 (Analytic continuation symmetry)**.** *For any overlap region U,*

$$\mathsf{DomCompat}(D_1, D_2, U) \wedge \mathsf{AnaCont}(f, D_1, g, D_2, h) \Longrightarrow \mathsf{DomCompat}(D_2, D_1, U) \wedge \mathsf{AnaCont}(g, D_2, f, D_1, h).$$

Proof. The first conclusion is Definition 9341.13. For the continuation conclusion, unfold Definition 9341.3. The holomorphic carrier for h is unchanged because $D_1 \cup D_2$ and $D_2 \cup D_1$ are the same domain by commutativity of union. The restriction of h to D_2 is the second restriction clause of the given continuation witness, and becomes the first restriction clause after the swap. The restriction of h to D_1 is the first restriction clause of the given witness, and becomes the second restriction clause after the swap. On the common overlap U, the given pointwise agreement of f with g transports to pointwise agreement of g with f by symmetry of the classifier hsame. These are exactly the three clauses required for $\mathsf{AnaCont}(g, D_2, f, D_1, h)$ under the reversed compatibility witness. □

Theorem 9341.15 (Analytic continuation triple-chain associativity)**.** *Let D_1, D_2, D_3 be open domains of $\mathsf{Complex}^{\uparrow}$. Suppose*

$$\mathsf{AnaCont}(f_1, D_1, f_2, D_2, h_{12}) \wedge \mathsf{AnaCont}(f_2, D_2, f_3, D_3, h_{23}) \wedge \mathsf{AnaCont}(h_{12}, D_1 \cup D_2, f_3, D_3, h_{(12)3}).$$

Then there is a holomorphic carrier $h_{1(23)}$ such that

$$\mathsf{AnaCont}(f_1, D_1, h_{23}, D_2 \cup D_3, h_{1(23)})$$

and

$$\mathsf{hsame}(h_{(12)3}, h_{1(23)})$$

pointwise on $D_1 \cup D_2 \cup D_3$.

Proof. Unpack the three continuation witnesses using Definition 9341.3. The first witness says that h_{12} restricts to f_1 on D_1 and to f_2 on D_2. The second says that h_{23} restricts to f_2 on D_2 and to f_3 on D_3. The third says that $h_{(12)3}$ restricts to h_{12} on $D_1 \cup D_2$ and to f_3 on D_3.

The domain-compatibility witness for D_1 with $D_2 \cup D_3$ is obtained from the first overlap witness. Its open disk, center, radius, and cluster sequence are unchanged; the containment clause is transported from $U_{12} \subseteq D_1 \cap D_2$ to $U_{12} \subseteq D_1 \cap (D_2 \cup D_3)$ by the canonical inclusion $D_2 \subseteq D_2 \cup D_3$. This is exactly the data required by Definition 9341.1, equivalently by the topological form in Definition 77.31.

Apply Definition 9341.5 to the first two links, with the union parenthesized as $D_1 \cup (D_2 \cup D_3)$. This gives a holomorphic carrier $h_{1(23)}$ whose restrictions are f_1 on D_1 and h_{23} on $D_2 \cup D_3$, hence

$$\mathsf{AnaCont}(f_1, D_1, h_{23}, D_2 \cup D_3, h_{1(23)}).$$

It remains to identify this carrier with the left-grouped carrier. The same restriction clauses show that $h_{(12)3}$ is also a continuation of f_1 against h_{23} over D_1 and $D_2 \cup D_3$. On D_1, $h_{(12)3}$ restricts to h_{12} and h_{12} restricts to f_1. On D_2, both $h_{(12)3}$ and h_{23} restrict to f_2: the former through h_{12}, the latter by the second continuation witness. On D_3, both restrict to f_3. Transitivity and symmetry of the classifier hsame combine these pointwise restriction witnesses. The overlap agreement is the inherited D_1-D_2 agreement, transported along $D_2 \subseteq D_2 \cup D_3$.

Thus $h_{(12)3}$ and $h_{1(23)}$ are two witnesses for the same predicate

$$\mathsf{AnaCont}(f_1, D_1, h_{23}, D_2 \cup D_3, -).$$

By uniqueness of analytic continuation, Definition 9341.4, they are classified by hsame pointwise on $D_1 \cup (D_2 \cup D_3)$. Transporting along associativity of union identifies this domain with $D_1 \cup D_2 \cup D_3$. □

Theorem 9341.16 (Analytic continuation identity law)**.** *Let D be an open domain of* $\mathsf{Complex}^{\uparrow}$*, let f be holomorphic on D, and let U be an overlap region with* $\mathsf{DomCompat}(D, D, U)$*. Suppose the compatibility witness is non-trivial in the sense of Definition 9341.1: its displayed open disk and quantitative cluster sequence lie inside the overlap region contained in $D \cap D$. Then*

$$\mathsf{AnaCont}(f, D, f, D, f)$$

holds. Moreover, for every h,

$$\mathsf{AnaCont}(f, D, f, D, h) \Longrightarrow \forall z,\ z \in D \Longrightarrow \mathsf{hsame}(h(z), f(z)).$$

Equivalently, h and f are pointwise hsame *on $D \cup D$, transported along the canonical idempotence equivalence $D \cup D = D$.*

Proof. Unpack Definition 9341.3. The output domain for the pair (D, D) is $D \cup D$. This domain is constructively equivalent to D: membership in D maps into $D \cup D$ by the left inclusion, and membership in $D \cup D$ returns membership in D by case analysis on the two identical summands. Thus the given holomorphic carrier of f on D also presents f as a holomorphic carrier on $D \cup D$.

The first restriction clause is the restriction of f from $D \cup D$ to the first copy of D, which is pointwise hsame to f by reflexivity of the classifier. The second restriction clause is identical for the second copy of D. The overlap clause asks that the two input functions, both equal to f, agree on U; this is again pointwise reflexivity of hsame. The non-trivial disk and cluster data in the compatibility witness supply exactly the admissible overlap record required by Definition 9341.1. These three clauses give $\mathsf{AnaCont}(f, D, f, D, f)$.

Now assume $\mathsf{AnaCont}(f, D, f, D, h)$. Comparing this witness with the identity witness just constructed, Definition 9341.4 gives pointwise $\mathsf{hsame}(h, f)$ on $D \cup D$. Transporting this classifier through the same idempotence equivalence $D \cup D = D$ yields, for every $z \in D$, the pointwise comparison $\mathsf{hsame}(h(z), f(z))$. □

Concretized in §577: Analytic continuation is realized as a finite operation certificate over holomorphic source rows, overlap witnesses, and branch-obstruction ledgers.

Chapter 9342

The ζ Continuation Application: Extending $\mathsf{ZetaBasic}^{\uparrow}$ to $\mathsf{Complex}^{\uparrow} \setminus \{1\}$

This chapter is an APPLICATION of $\mathsf{AnaCont}^{\uparrow}$ (chapter 9341) to the basic ζ function $\mathsf{ZetaBasic}^{\uparrow}$ (chapter 71). It produces $\mathsf{ZetaCont}^{\uparrow}$: the unique meromorphic extension of ζ from $\Re(s) > 1$ to $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ with a simple pole of explicit residue 1 at $s = 1$. The chapter proceeds in two stages: (i) use the Dirichlet eta quotient on the open locus where $1 - 2^{1-s} \neq 0$, fill the non-pole eta-quotient zeroes by removable continuation, and keep $s = 1$ as the pole; then (ii) extend ζ to $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ through the meromorphic completed-function identity. Both stages are concrete applications of analytic continuation. The chapter uses the constructive Gamma function Γ on $\mathsf{Complex}^{\uparrow} \setminus \{0, -1, -2, \ldots\}$ as a black-box meromorphic factor — the constructive Gamma is itself a SCHEMA-ONLY HORIZON awaiting a future BEDC chapter; the present chapter exposes the interface but does not close all underlying witnesses. The construction uses no Classical.choice, no Quot.sound, and no propext, so the application's Lean targets fall under the project's axiom-purity gate.

9342.1 The Dirichlet eta function

Definition 9342.1 (Dirichlet eta function)**.** *The Dirichlet eta function is the alternating Dirichlet series $\eta(s) = \sum_{n=1}^{\infty}(-1)^{n-1}n^{-s}$, with coefficients $a_n = (-1)^{n-1}$ (decidable on parity of unary index).*

Theorem 9342.2 (Eta well-defined on $\Re(s) > 0$)**.** *The Dirichlet eta function converges in $\mathsf{CplxSer}^{\uparrow}$ (chapter 69) for every s with $\Re(s) > 0$, with explicit modulus given by an alternating-series majorant. The function $s \mapsto \eta(s)$ is holomorphic on $\Re(s) > 0$ (chapter 68).*

Sketch. The alternating-sign structure gives Cauchy-criterion convergence on the half-plane $\Re(s) > 0$; explicit modulus from successive-difference bounds. Holomorphicity from term-wise differentiation with uniform modulus on compact sub-half-planes. □

Theorem 9342.3 (Eta-zeta relation)**.** *For every s with $\Re(s) > 1$, the relation*

$$\zeta(s) = \frac{\eta(s)}{1 - 2^{1-s}}$$

holds, where the right-hand side is computed using complex division ($\mathsf{Complex}^{\uparrow}$, chapter 39). The denominator is non-zero on $\Re(s) > 1$. On the larger half-plane $\Re(s) > 0$, its zero locus is the

discrete line

$$s = 1 + \frac{2\pi i k}{\log 2}, \qquad k \in \mathbb{Z}.$$

Sketch. The relation $\eta(s) = (1-2^{1-s})\zeta(s)$ follows from the rearrangement $\zeta(s) - 2\cdot 2^{-s}\zeta(s) = \eta(s)$, valid by absolute convergence. Solve for ζ. □

9342.2 ζ on the half-plane $\Re(s) > 0$ via eta

Theorem 9342.4 (ζ on $\Re(s) > 0$)**.** *On the open set*

$$U_\eta = \{s : \Re(s) > 0 \text{ and } 1 - 2^{1-s} \neq 0\},$$

the eta quotient $\eta(s)/(1-2^{1-s})$ gives the holomorphic continuation of ζ from $\Re(s) > 1$. At each point

$$s = 1 + \frac{2\pi i k}{\log 2}, \qquad k \in \mathbb{Z},\ k \neq 0,$$

the quotient has a removable $0/0$ singularity and the value is defined by the unique holomorphic continuation across a small punctured neighbourhood. The point $s = 1$ is not filled by this row; it is the pole recorded in Definition 9342.5. The extension is unique by the constructive identity theorem (chapter 68); the underlying analytic-continuation witness is supplied by $\mathsf{AnaCont}^{\uparrow}$ *applied to overlapping disk neighbourhoods inside both half-planes.*

Sketch. Apply $\mathsf{AnaCont}$ (chapter 9341) to the holomorphic restrictions of (i) ζ on $\Re(s) > 1$ and (ii) $\eta(s)/(1-2^{1-s})$ on U_η. The eta-relation provides the overlap-region agreement. At the non-pole denominator zeroes, the two sides agree on punctured neighbourhoods and the singularity is removable by the continuation row; at $s = 1$ the pole ledger is used instead. □

9342.3 The pole at $s = 1$

Theorem 9342.5 (Simple pole at $s = 1$ with residue 1)**.** *ζ has a simple pole at $s = 1$ with residue exactly* $\mathsf{E1(e)}$ *(the unit complex history). Concretely, the limit $\lim_{s\to 1}(s-1)\zeta(s) = 1$ holds with explicit modulus given by the harmonic-series log-divergence rate.*

Sketch. The harmonic divergence $\sum_{n=1}^{N} 1/n \sim \log N$ gives $(s-1)\zeta(s) \to 1$ as $s \to 1^+$ with explicit unary-history modulus on the difference. Apply the eta-relation continuation to extend this to a simple-pole witness on a neighbourhood of $s = 1$. □

9342.4 The functional equation

Theorem 9342.6 (Riemann functional equation)**.** *The functional equation is recorded first as a meromorphic identity for the completed zeta carrier. Writing*

$$\Lambda(s) = \pi^{-s/2}\Gamma(s/2)\zeta(s),$$

the identity is

$$\Lambda(s) = \Lambda(1-s)$$

as meromorphic functions after the pole ledger is attached. Equivalently, the classical factorized expression

$$\zeta(s) = 2^s \pi^{s-1} \sin(\pi s/2)\Gamma(1-s)\zeta(1-s)$$

is read as an identity in the meromorphic function field before any pointwise value is claimed. The displayed factors are:

1. $2^s = \exp(s \log 2)$ *on the constructive complex* $\exp$ *and constructive real* $\log 2$ *(*$\mathsf{Real}^\uparrow$*, chapter 38);*
2. π^{s-1} *defined similarly with constructive real* $\log \pi$*;*
3. $\sin(\pi s/2)$ *defined via the constructive complex sine power series (radius of convergence infinity);*
4. $\Gamma(1-s)$ *is the constructive complex Gamma function on* $\mathsf{Complex}^\uparrow \setminus \{0, -1, -2, \ldots\}$ *— exposed here as a SCHEMA-ONLY interface awaiting a future BEDC* $\mathsf{Gamma}^\uparrow$ *chapter.*

At positive integers where $\Gamma(1-s)$ *has a pole and* $\sin(\pi s/2)$ *may cancel it, the pointwise value is admitted only after a separate removable-cancellation row has been checked. Thus this theorem is not a pointwise assertion at every displayed complex history.*

Sketch. The functional equation is established by the classical contour-integral identity reorganization, made constructive on the BEDC contour-integral framework (chapter 9340). The Gamma factor's constructive Lean target is held open as a schema-only horizon; the rest of the derivation (the trigonometric/exponential factors and the contour reorganization) is fully constructive on the data of this chapter. □

Remark 9342.7 (Constructive Gamma as schema-only horizon)**.** *The constructive complex Gamma function* Γ *on* $\mathsf{Complex}^\uparrow \setminus \{0, -1, -2, \ldots\}$ *is intentionally exposed here as a schema-only horizon: this chapter uses* Γ *as a black-box meromorphic factor in the functional equation, identifies the input/output classifier interface (input: complex history outside the negative integers; output: complex-history value with explicit modulus), but does not produce the closed Lean witness. A future BEDC chapter* $\mathsf{Gamma}^\uparrow$ *is the natural home for closing the schema.*

9342.5 ζ on $\mathsf{Complex}^\uparrow \setminus \{1\}$

Theorem 9342.8 (Full continuation)**.** *The function* ζ *extends holomorphically to* $\mathsf{Complex}^\uparrow \setminus \{1\}$ *by applying the functional equation to reflect from* $\Re(s) > 0$ *into* $\Re(s) < 1$ *through the line* $\Re(s) = 1/2$*. The extension is unique by the identity theorem.*

Sketch. Apply $\mathsf{AnaCont}$ (chapter 9341) twice: first to extend ζ from $\Re(s) > 1$ to $\Re(s) > 0$ (via eta), then to extend from $\Re(s) > 0$ to $\Re(s) < 1$ (via functional equation reflection). The two extensions agree on the strip $0 < \Re(s) < 1$, giving a single continuation on $\mathsf{Complex}^\uparrow \setminus \{1\}$. □

9342.6 Trivial zeros from the functional equation

Theorem 9342.9 (Trivial zeros witness)**.** *For every unary* $n \geq \mathsf{E1}(\mathsf{e})$*,* $\zeta(-2n) = 0$ *constructively, with witness coming from* $\sin(\pi s/2) = 0$ *at even* $s \leq -2$ *in the functional equation. The other factors* $2^s, \pi^{s-1}, \Gamma(1-s), \zeta(1-s)$ *are non-zero and finite at* $s = -2n$*.*

Sketch. At $s = -2n$, $\pi s/2 = -n\pi$, so $\sin(-n\pi) = 0$ constructively (rational multiple of π with even multiplier — unary-decidable). The remaining factors are non-zero by direct constructive evaluation: $\Gamma(1+2n)$ is the unary factorial $(2n)!$ ($\mathsf{Prime}^\uparrow$, chapter 64); $\zeta(1+2n)$ lies in the half-plane $\Re(s) > 1$ where ζ is non-zero (chapter 71). □

9342.7 The application certificate

Definition 9342.10 (Zeta continuation application certificate)**.** *The application certificate consists of: (i) input spec — the basic ζ certificate of* $\mathsf{ZetaBasic}^{\uparrow}$ *(chapter 71); (ii) output spec — the holomorphic extension of ζ on* $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ *with simple pole and residue 1 at $s = 1$; (iii) classifier —* hsame *pointwise on the extended domain; (iv) operation laws — eta-relation, functional equation, chain continuation; (v) ledger policy — pole at 1, trivial zeros at negative even integers, the constructive Gamma horizon.*

Theorem 9342.11 (Concrete $\mathsf{ZetaCont}^{\uparrow}$ application certificate)**.** *The data above forms an application certificate of* $\mathsf{AnaCont}^{\uparrow}$ *on* $\mathsf{ZetaBasic}^{\uparrow}$ *producing the extended ζ function on* $\mathsf{Complex}^{\uparrow} \setminus \{1\}$.

Sketch. Verification of each step of the two-stage continuation in the preceding sections; the constructive-Gamma factor in the functional equation is exposed as a schema-only horizon. The application uses no Classical.choice, no Quot.sound, and no propext, so the application's Lean targets fall under the project's axiom-purity gate. □

Theorem 9342.12 (Semantic $\mathsf{ZetaCont}^{\uparrow}$ application certificate)**.** *The concrete certificate of Definition 9342.11 satisfies the semantic application-certificate schema obtained by adapting* NameCert *(chapter 9) to operation-application inputs/outputs.*

Sketch. Direct verification of the schema fields using the constructions of this chapter and the hsame-respect of the underlying $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, and $\mathsf{Complex}^{\uparrow}$ classifiers. □

Concretized in §580: The eta-continuation, pole, functional-equation, trivial-zero, and Gamma-horizon routes are packaged as a finite witness NameCert.

Concretized in §581: The functional-equation ledger is realized as a finite NameCert packet for eta continuation, Gamma boundary, contour provenance, and trivial-zero consumption.

Concretized in §570: Zeta continuation is realized as a finite NameCert socket packet over zeta-basic, eta, analytic-continuation, pole, functional-equation, Gamma, and trivial-zero ledger rows.

Concretized in §588: Zeta continuation application is realized as a finite NameCert packet over eta continuation, functional-equation continuation, pole and zero ledgers, Gamma horizon rows, and application-certificate non-escape.

Concretized in §1101: Gamma-Zeta factor seals realize the functional-equation Gamma factor as a finite NameCert packet over Gamma-domain, reflected-input, zeta-application, transport, replay, provenance, and local naming rows.

Chapter 9343

Exploratory RH Route Ledger

This chapter records the exploratory RH route developed in a 169-turn working session. The purpose is archival and diagnostic. It separates ideas that are already covered by existing BEDC formal surfaces from heuristic geometry, conjectural bridge obligations, and full-strength restatements. It does not prove the Riemann Hypothesis, does not assert proximity to a proof, and does not introduce Lean targets for philosophical or analogy-driven material.

9343.1 Audit Convention

Each item below is assigned one of four statuses.

Definition 9343.1 (Route ledger status)**.** *A route item is* covered *when its formal kernel is already represented by an existing RH chapter or by an existing* BEDC.Derived.RHRoute *module. It is* heuristic *when it gives geometric, physical, or ontological guidance without a present theorem. It is* conditional *when it gives a valid implication from an explicit bridge obligation to* RH *or to a zero-exclusion target. It is* full-strength restatement *when the proposed bridge is, after quantification strong enough to do work, another form of the no-off-line-zero problem.*

The four statuses are intentionally not arranged as a progress scale. "Covered" does not mean that RH is proved; it means that the particular small datum has already been formalized or recorded. "Heuristic" does not mean false; it means that no certificate-bearing mathematical statement is being claimed. "Conditional" records an honest theorem shape whose hypotheses remain open. "Full-strength restatement" records a useful diagnostic translation rather than a reduction of difficulty.

9343.2 Three Boundaries and the Prime-Unitary Surface

The earliest stable correction in the exploration was the separation of three objects:

$$\mathrm{Fix}(J), \qquad \prod_s \mathsf{Decidable}(\mathsf{ZetaZero}(s)), \qquad \mathsf{RH}.$$

The first is the geometric fixed surface of the functional-equation mirror

$$J(s) = 1 - \overline{s}.$$

The second is an exact zero decision operator. The third is the global rigidity statement that a certified critical-strip zero lands on the fixed surface.

Status: covered. The fixed-surface computation is the subject of chapter 9240, whose Lean-side kernel is the module BEDC.Derived.RHRoute.FunctionalEquationSymmetry. That module verifies the fixed-surface calculation identifying J-fixedness with the critical-line predicate. The same chapter also records why exact zero decidability is a different object from the RH proposition.

The prime-unitary readout was also stabilized. For a displayed base $a > 1$,

$$|a^{1/2-s}| = 1 \quad \Longleftrightarrow \quad \Re(s) = 1/2.$$

For primes this is the centered Euler-channel readout $U_p(s) = p^{1/2-s}$. The point of the ledger is the negative boundary: this calculation is a critical-line reparameterization, not an RH mechanism by itself. It applies for every displayed $a > 1$ and says nothing about whether s carries a zeta-zero packet.

Status: covered. The prime-local surface is recorded in chapter 9243 and chapter 9240. Its Lean-side carrier is the existing BEDC.Derived.RHRoute.UnitaryBalance module. The route becomes RH-facing only after a zero-ledger bridge supplies a reason why an off-line zero cannot coexist with the displayed prime-channel defect.

9343.3 Located Process Geometry and Type Boundaries

The session repeatedly used a geometric language in which a rational value axis is distinguished from a process axis. In BEDC terms, the strict content is this: located real data are not bare rational points but approximation processes with explicit moduli. A rational value is the degenerate case with a constant approximation and trivial modulus. Comparing two located reals is therefore comparison of two approximation processes to a displayed tolerance, not an unrestricted trichotomy of exact completed points.

Status: heuristic as imagery, covered as infrastructure. The language of orthogonal axes, shadows, and process directions is only a visualization. The certificate content is the ordinary BEDC discipline already used by the located real and zeta-box modules: finite tolerance, apartness, and explicit moduli stay inside the constructive fragment; exact global comparison and unrestricted point extraction belong to a separate boundary.

The same discussion produced a useful caution about Classical.choice, Quot.sound, and propext. The honest statement is not that classical mathematics is erroneous, nor that these names are proved necessary for zeta. The statement is only that such constants measure implementation footprints for particular classical encodings: choice extracts data from existence, quotients identify processes as points, and propositional extensionality collapses proof distinctions. BEDC avoids those collapses in the kernel.

Status: heuristic and governance-facing. The transcript's strongest versions of this point are philosophy of implementation, not RH lemmas. No Lean target is attached to them here.

9343.4 Common Carrier and Zeta as a Projection Chain

The middle part of the session organized triangles, circles, elliptic normalizations, Fourier and Laplace transforms, and zeta as a chain of compatible projections rather than as a literal holomorphic continuation from one object to the next. The valid BEDC reading is:

$$\text{shared carrier rows} \quad \rightsquigarrow \quad \text{compatible algebraic, geometric, spectral, and analytic readouts.}$$

For zeta this means that the Dirichlet series, Euler product, eta row, analytic continuation, archimedean completion, functional equation, pole ledger, zero ledger, and branch-transport rows are compatible readings of a recovered carrier.

Status: covered. The recovered-carrier discipline is already recorded in chapter 9243, chapter 9241, and chapter 9250. The taxonomy side is represented by the existing BEDC.Derived.RHRoute.ZetaLikeProjec module. This does not imply that properties of a triangle, a circle, a finite elliptic model, or a Fourier transform transfer automatically to Riemann zeta. Such transfer requires displayed carrier maps, overlap laws, and tail or continuation estimates.

The finite quartic spectral-curve example was used as a guide for what a closed finite model looks like: recurrence data, spectral polynomial, normalization, projection, dominant branch, and endpoint zero law are all in the same finite ledger. The honest lesson for RH is negative as much as positive. Riemann ξ is an order-one entire transcendental function with infinitely many zeros; a finite algebraic branch argument does not transfer through the finite-to-infinite boundary without a separate convergence, dominance, or positivity theorem.

Status: heuristic for RH, useful as finite-model discipline. It is not a proof template that can be applied by analogy alone.

9343.5 Object Induction, Three Invariants, and No-Split Zeros

Another arc described a triad of object-level invariants: observation/time, distinction/prime locality, and symmetry/transport. In BEDC vocabulary this can be read as a route through histories and continuation, gap or apartness data, and hsame-respecting transport. The proposed RH reading was that a zero fiber should have no additional same-layer normal degree of freedom.

Status: covered as a conditional route, heuristic as philosophy. Existing modules already record the usable conditional kernel: ObjectRealityInductionRoute packages the object-induction route; ZeroGenerationInitiality packages the initiality route; and ZeroSignatureFaithfulness records signature-faithfulness obligations. The paper-side references are chapter 9255 and the neighboring zero-generation chapters.

The missing theorem is not a slogan about there being only three invariants. It is a zero-fiber exactness theorem. One possible form says: every accepted critical-strip zero packet is generated by pre-RH constructors, and those constructors preserve the fixed-half subcarrier. Another possible form says: a mirror-split off-line zero would expose a defect row that the completed carrier cannot accept. Both are conditional bridges; neither is currently closed for Riemann zeta.

Status: conditional. The session's "pigeonhole" language is best read as classifier faithfulness. It is not a proof that two mirror-related off-line zeros must be the same point.

9343.6 Finite Prime Windows, Odd Defects, and Even Energy

The most concrete non-analytic core of the session was the normal-defect calculation. Write

$$s = \frac{1}{2} + \delta + i\tau.$$

Then each centered prime channel satisfies

$$|p^{1/2-s}| = p^{-\delta}.$$

Consequently a nonzero δ produces a coherent amplitude defect in every displayed nonempty prime window. A mirror pair sends δ to $-\delta$; signed first-order defect can cancel between the two mirror reads, but the even energy δ^2 cannot be removed by that sign change.

Status: covered as finite readout. The window and defect apparatus is already recorded in chapter 9244, chapter 9249, and chapter 9245. Existing Lean modules cover the finite machinery: FinitePrimeWindow, PrimeSkewDefect, OddZeroDefect, EvenDefectEnergy, and FiniteToInfinitePhaseEnergyTower.

The finite conclusion is not RH. It says that off-line bias is locally visible. The missing analytic carrier theorem is

$$\mathsf{ZetaZero}(s) \times \mathsf{PositiveNormalDefect}(s) \to \bot,$$

or an equivalent completed-carrier exclusion. This is the same hinge isolated in the local-global exclusion chapters. It is recorded here as conditional, not as a checked theorem.

9343.7 Far-End, Solenoid, and Lee–Yang Readings

The far-end discussion produced two separate items that must not be confused. First, it gave a useful discipline: the infinite end is not a point, a warehouse, or an untyped place where a finite defect can disappear. Second, when expressed as "no far-end absorption for zeta," it becomes a restatement of the no-off-line-zero problem unless an independent positivity or domination theorem is supplied.

Status: full-strength restatement when quantified strongly enough for zeta. The existing OnticBoundaryLedger module already records this boundary as a ledger entry rather than a proof. The far-end unity and energy socket modules record finite normalization interfaces, but they do not discharge the zeta-specific zero-exclusion theorem.

The solenoid and Lee–Yang turn was more precise. For a finite prime set S, an S-localized solenoid has a visible circle and a compact p-adic fiber. A finite Lee–Yang stable resonance can force its zeros to land on the unit boundary. In the s-coordinate, a local factor of the form

$$z = e^{-(s-1/2)H}, \qquad H > 0,$$

lands on the unit circle only when $\Re(s) = 1/2$.

Status: conditional. Finite solenoid or Lee–Yang mechanisms are genuine mathematical structures, and existing RHRoute ledgers contain finite solenoid-bound and source-gap records. The missing bridge is to show that Riemann ξ admits a zero-relevant solenoid Lee–Yang capture: on each off-line compact region, a stable solenoid main term dominates a tail term, or a sequence of finite Lee–Yang stable approximants converges to Ξ. Without that bridge, the mechanism does not prove RH.

9343.8 Dyadic Truncations, Jensen, Hilbert–Polya, and Weil Positivity

The final technical arc audited several standard route families. All of them can be made honest by expressing their bridge obligations directly.

Principle 9343.2 (Dyadic Laguerre–Polya bridge)**.** *If a family of completed dyadic truncations Ξ_K converges locally uniformly to Ξ and every Ξ_K belongs to the Laguerre–Polya class, then Hurwitz-type reasoning gives real zeros for Ξ and hence RH.*

Status: bridge sound, premise false for the sharp truncation route recorded in this paper. chapter 9338 records why the tempting all-K sharp-cutoff Laguerre–Polya premise does not hold. Thus the dyadic arc is preserved as a negative route ledger, not as a pending proof.

Principle 9343.3 (Jensen and Laguerre–Polya audit)**.** *Showing all Jensen polynomials attached to Ξ are hyperbolic is a classical Laguerre–Polya route to RH. Fixed-degree high-shift hyperbolicity is informative, but the all-degree all-shift family is the hard object.*

Status: full-strength restatement as a global criterion. It is not a BEDC shortcut.

Principle 9343.4 (Hilbert–Polya and spectral determinant audit)**.** *A self-adjoint or unitary spectral realization whose determinant is Riemann ξ would force the required real-line or unit-circle zero placement. Finite CMV, Jacobi, and determinant interfaces demonstrate the shape of such mechanisms but do not construct the required infinite operator for ξ.*

Status: conditional. The finite spectral interface is useful; the determinant-equals-ξ theorem is not present.

Principle 9343.5 (Weil and Li positivity audit)**.** *Weil positivity and Li coefficient positivity are faithful RH criteria. They are valuable because they identify the correct positive form or coefficient sequence. They become proof mechanisms only when the positivity is derived from an independent structure rather than assumed in a form coextensive with the fixed RH target.*

Status: full-strength restatement as global positivity; potentially useful as a target if an independent semi-local domination theorem is produced. The last turns of the session therefore correctly moved toward semi-local Weil positivity, finite matrix criteria, Birman–Schwinger bounds, and low/high-frequency estimates. Those are bridge obligations, not checked results in this paper.

9343.9 Lean Decision

No Lean module accompanies this chapter. The session contains many phrases that can be given formal-looking names, but the only parts that are currently suitable for 0-axiom BEDC formalization are already represented by existing RHRoute modules: the functional-equation fixed surface, the prime-unitary readout, finite prime-window gluing, odd and even defect rows, object-induction and zero-generation conditional routes, route-closure separation, zeta projection taxonomy, and finite solenoid-bound ledgers.

The remaining items require one of the following before they become Lean targets:

a zeta-specific defect-zero incompatibility theorem,
a zero-presentation exactness theorem for a pre-RH generator,
a solenoid Lee–Yang capture theorem for Ξ,
a full Laguerre–Polya or Jensen hyperbolicity proof,
an independent Weil or Li positivity theorem,
or a spectral realization whose determinant is ξ.

None of these is available in the repository as a closed 0-axiom theorem. They are therefore recorded as route obligations and not as Lean stubs.

9343.10 Summary

The session is valuable as a route ledger because it repeatedly found the same boundary from different directions. The critical line is already the functional-equation fixed surface. Prime-unitary

balance and finite windows read that surface cleanly, but do not by themselves carry a zeta-zero exclusion. Geometric language about common carriers, process axes, far-ends, resonances, and no-hair principles is useful only when translated into displayed carrier maps, stability, positivity, or tail domination. When such translation is strong enough to rule out off-line zeros, it is exactly where the RH difficulty lives.

Thus the honest current state is:

$$\boxed{\text{finite and conditional RHRoute kernels are already recorded;}}$$

$$\boxed{\text{the session contributes no separate 0-axiom theorem beyond existing coverage;}}$$

$$\boxed{\text{the unresolved work is analytic rigidity, not terminology.}}$$

Concretized in §7545: The critical-strip oscillation packet realizes the finite zero-window boundary used by the RH route.

Chapter 9344

The Unary-Direction Bridge $\mathsf{Nat}^{\uparrow} \leftrightarrow \mathsf{AxisNat}^{\uparrow}$

This chapter introduces a kernel-distinct comparison interface between the $\mathsf{E1}$-spine carrier behind $\mathsf{Nat}^{\uparrow}$ (Chapter 28) and the $\mathsf{E0}$-spine carrier behind $\mathsf{AxisNat}^{\uparrow}$ (Chapter 276). The bridge does NOT identify the two carriers inside the BEDC finite kernel; it records that the two interfaces are isomorphic in the standard model, while remaining propositionally distinct on histories.

9344.1 Carrier intersection

Theorem 9344.1 (Unary and zero-spine meet only at the empty history)**.** *For every history h, if both* $\mathsf{Unary}(h)$ *and* $\mathsf{ZeroSpine}(h)$ *hold, then* $h = \mathsf{e}$.

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Bridge.unary and zeroSpine only meet at empty`

Proof. By cases on the zero-spine derivation. In the empty case the conclusion is immediate. In the $\mathsf{E0}$-step case the unary witness on $\mathsf{E0}(h')$ contradicts unary no-zero-extension. □

Lemma 9344.2 (Unary at $\mathsf{E1}(\mathsf{e})$ is not zero-spine)**.** $\mathsf{ZeroSpine}(\mathsf{E1}(\mathsf{e}))$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Bridge.unary e1 not zeroSpine at step`

Proof. Direct from zero-spine no-one-extension applied at $h = \mathsf{e}$. □

Lemma 9344.3 (Zero-spine at $\mathsf{E0}(\mathsf{e})$ is not unary)**.** $\mathsf{Unary}(\mathsf{E0}(\mathsf{e}))$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Bridge.zeroSpine e0 not unary at step`

Proof. Direct from unary no-zero-extension applied at $h = \mathsf{e}$. □

9344.2 The bridge interface

Definition 9344.4 (Unary-direction bridge)**.** *The bridge* $\mathsf{UnaryDirectionBridge}$ *records:*

1. *the carrier* Unary *of* $\mathsf{Nat}^{\uparrow}$*;*
2. *the carrier* $\mathsf{ZeroSpine}$ *of* $\mathsf{AxisNat}^{\uparrow}$*;*
3. *the kernel-distinctness witness, that the two carriers intersect only at* e.

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Bridge.UnaryDirectionBridge`
↪ variant `BEDC.Derived.AxisZeckendorf.Bridge.unaryDirectionBridge`

Remark 9344.5 (Bridge is kernel-distinct, not kernel-identification)**.** *The bridge records witnesses, not equalities. The* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$ *carriers remain distinct as history predicates inside the finite kernel. Any attempt to identify them at the kernel level conflicts with Theorem 9344.1.*

9344.3 Standard-model isomorphism (horizon)

The bridge is licensed by an isomorphism in the standard model: each non-empty unary history $\mathsf{E1}^n(\mathsf{e})$ corresponds to a non-empty zero-spine history $\mathsf{E0}^n(\mathsf{e})$, and the correspondence is bijective and continuation-preserving in the standard model. The proof of this correspondence at paper level, together with its zero-spine pattern variant, is a horizon obligation queued for the formalize pipeline.

Theorem 9344.6 (Standard-model isomorphism, horizon)**.** *There is a length-preserving bijection between non-empty unary histories and non-empty zero-spine histories, and this bijection commutes with the additive interfaces* $\mathsf{Add}^{\uparrow}$ *and* $\mathsf{AxisAdd}^{\uparrow}$ *in the standard model.*

Remark 9344.7 (Bridge does not retire either chapter)**.** *The bridge does not collapse Chapters 28 and 276 into a single chapter. Both certificates remain independent inside the finite kernel; the bridge only relates them at the standard-model boundary.*

Concretized in §530: The unary-direction bridge is realized as a finite NameCert packet relating NatUp unary spines, AxisNatUp zero spines, and kernel-distinctness witnesses.

Concretized in §572: The unary-zero-spine standard-boundary comparison is represented as a finite NameCert packet with length-match rows, two finite continuation routes, and a kernel nonidentification boundary.

Concretized in §599: AxisUnarySpineNormalizer packets realize the unary-to-zero-spine standard boundary as finite BHist, hsame, Cont, Pkg, and NameCert rows without identifying the carriers.

Concretized in §700: Unary-direction routers realize the consumer-side choice between unary-to-axis and axis-to-unary standard-boundary reads as a finite NameCert packet without identifying the carriers.

Concretized in §1010: Unary direction normalizers realize the axis-Zeckendorf bridge as finite axis, zero-spine, carry, direction, transport, route, provenance, and NameCert rows.

Chapter 9345

A Concrete Naming Certificate for $\mathsf{ZCarry}^{\uparrow}$

This chapter introduces the Zeckendorf carry classifier $\mathsf{ZCarry}(h, k)$, generated by the local rewrite `word_011` $\to$ `word_100` (cf. Definition 278.2). The crucial property of this classifier is that it is NOT history-sameness: $\mathsf{ZCarry}(h, k)$ does not entail $\mathsf{hsame}(h, k)$. The Fibonacci value-preservation reading is a horizon target.

9345.1 The carry classifier

Definition 9345.1 (Zeckendorf carry classifier)**.** *The classifier* $\mathsf{ZCarry}(h, k)$ *is generated by the single base rule*

$$\mathsf{ZCarry}(\texttt{word_011}, \texttt{word_100}).$$

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarry`

Lemma 9345.2 (The display carry is generated)**.** $\mathsf{ZCarry}(\texttt{word_011}, \texttt{word_100})$ *holds.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry 011 100`

Proof. First (and only) generator of ZCarry. □

9345.2 Carry is not history-sameness

Theorem 9345.3 (`word_011` and `word_100` are not hsame)**.** $\mathsf{hsame}(\texttt{word_011}, \texttt{word_100})$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry 011 100 not hsame`

Proof. Recall that hsame is propositional history identity. The first history has top constructor $\mathsf{E1}$; the second has top constructor $\mathsf{E0}$. The two top constructors are distinct, so the histories are unequal, and hsame does not hold. □

Theorem 9345.4 (Carry is not history-sameness)**.** *For every* h *and* k*,* $\mathsf{ZCarry}(h, k)$ *implies* $\neg\mathsf{hsame}(h, k)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry not hsame`

Proof. By the unique generator, any $\mathsf{ZCarry}(h, k)$ instance has $h = \texttt{word_011}$ and $k = \texttt{word_100}$. The conclusion follows from Theorem 9345.3. □

9345.3 Carry transports normal-form status

Theorem 9345.5 (Carry target is normal)**.** *For every h, k, if* $\mathsf{ZCarry}(h, k)$*, then* $\mathsf{ZNormal}(k)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry target normal`

Proof. The unique generator pins $k =$ `word_100`; apply Theorem 278.3. □

Theorem 9345.6 (Carry source is non-normal)**.** *For every h, k, if* $\mathsf{ZCarry}(h, k)$*, then* $\neg\mathsf{ZNormal}(h)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry source not normal`

Proof. The unique generator pins $h =$ `word_011`; apply Theorem 278.4. □

9345.4 The certificate

Definition 9345.7 (Carry source specification)**.** *The source specification* ZCarrySource *consists of histories h with* $\neg\mathsf{ZNormal}(h)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarrySourceSpec`

Definition 9345.8 (Carry pattern specification)**.** *The pattern specification* ZCarryPattern *is the carry generator* $\mathsf{ZCarry}(h, k)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarryPatternSpec`

Definition 9345.9 (Carry classifier specification)**.** *The classifier specification* ZCarryClassifier *records, on each carry pair, that the target is normal, the source is non-normal, and the source is not* hsame *to the target.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarryClassifierSpec`

Definition 9345.10 (Carry stability certificate)**.** *The stability certificate* ZCarryStability *consists of:*

1. *carry-not-*hsame *(Theorem 9345.4);*
2. *target normal (Theorem 9345.5);*
3. *source non-normal (Theorem 9345.6);*
4. *Fibonacci value preservation, horizon obligation.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarryStabilityCert`
↪ variant `BEDC.Derived.AxisZeckendorf.Carry.zCarryStabilityCert`

Definition 9345.11 (Carry ledger policy)**.** *The ledger policy* ZCarryLedger *records the non-normal preimage and the carry witness behind any visible carry-classification.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.ZCarryLedgerPolicy`

Theorem 9345.12 (Concrete $\mathsf{ZCarry}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}} := (\mathsf{ZCarrySource}, \mathsf{ZCarryPattern}, \mathsf{ZCarryClassifier}, \mathsf{ZCarryStability}, \mathsf{ZCarryLedger})$$

forms a valid naming certificate for the derived classifier interface $\mathsf{ZCarry}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Carry.zCarry namecert`

Proof. The five fields are supplied by the preceding definitions. The stability field includes three checked theorems and one explicit horizon obligation. □

Corollary 9345.13 ($\mathsf{ZCarry}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{ZCarry}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$*. It is a classifier, not a kernel equality, and it is not a primitive object of the finite kernel.*

Remark 9345.14 (Carry must remain a classifier)**.** *The crucial discipline of this chapter is that* ZCarry *is a classifier between non-*hsame *histories. Any future extension that promotes the carry to a kernel-level identification—either by replacing* hsame *on* ZNormal *histories or by inserting a global rewrite into the kernel—conflicts with Theorem 9345.4 and is rejected by the bridge discipline.*

Concretized in §567: Zeckendorf carry classifiers are realized as finite BHist carry-window NameCert packets separating carry-relatedness from history sameness.

Concretized in §606: The displayed carry row is realized as a finite value-preservation NameCert packet that keeps carry-relatedness separate from history sameness.

Concretized in §608: Zeckendorf carry normalization is realized as a finite NameCert packet separating carry, normal-form target, value ledger, and non-hsame boundary rows.

Concretized in §692: AxisZeckendorf carry budgets realize the finite carry-window, normal-form, value-ledger, non-hsame, transport, route, provenance, and NameCert packet behind the carry classifier.

Concretized in §713: Axis carry confluence is realized as a finite BHist NameCert packet for two displayed carry-normalization routes, their shared normal target, value ledger, and non-escape boundary.

Concretized in §1049: Axis carry route separation is realized as a finite NameCert packet separating carry-relatedness, normal-form routing, and history-sameness refusal.

Concretized in §1118: Axis carry diamond routes realize two displayed carry-normalization paths, shared normal target rows, value ledgers, and non-history-sameness boundaries as a finite NameCert packet.

Concretized in §7072: Overlapping Zeckendorf carry windows are realized as a finite local critical-pair packet with route, shared normal target, value-ledger, and non-hsame boundary rows.

Chapter 9346

Zeckendorf Golden Phase Projection

This chapter records a vision-level derived interface attached to the Axis-Zeckendorf route. It reads the no-adjacent-11 grammar as a finite visible surface, keeps the raw binary microstates in a residual fiber ledger, and assigns a golden-ratio phase only as an external projection. The projection is not a BEDC kernel object, not a physical law, not a real-completion theorem, and not a quotient of histories.

9346.1 Window observers and residual fibers

Definition 9346.1 (Stable Zeckendorf window)**.** *For $m \geq 0$, let*

$$\Omega_m := \{0,1\}^m.$$

The stable Zeckendorf window is the subcarrier

$$X_m := \{x \in \Omega_m : \ x_k x_{k+1} = 0 \text{ for all admissible } k\}.$$

Equivalently, a visible word in X_m contains no adjacent 11 *pattern.*

Definition 9346.2 (Fibonacci value and fold)**.** *Let $F_0 = 0$, $F_1 = 1$, and $F_{n+1} = F_n + F_{n-1}$ for $n \geq 1$. For $\omega \in \Omega_m$, define its Fibonacci value by*

$$N_m(\omega) := \sum_{k=1}^{m} \omega_k F_{k+1}.$$

Let $Z(N_m(\omega))$ be the Zeckendorf normal digit sequence of this value. The window fold is the visible truncation

$$\mathsf{Fold}_m : \Omega_m \to X_m, \qquad \mathsf{Fold}_m(\omega) := \pi_m Z(N_m(\omega)).$$

The map is a projection from raw binary words to stable visible words; it is not an identification of the raw histories.

Definition 9346.3 (Zeckendorf window observer)**.** *A Zeckendorf window observer at width m consumes only the visible word*

$$x = \mathsf{Fold}_m(\omega) \in X_m.$$

The hidden residual fiber over x is

$$\mathcal{F}_m(x) := \mathsf{Fold}_m^{-1}(x), \qquad d_m(x) := |\mathcal{F}_m(x)|.$$

Thus the observer sees a stable Zeckendorf word while the ledger retains the raw microstates compressed by the fold.

Principle 9346.4 (Fold readout does not erase the fiber)**.** *Any consumer of* $\mathsf{Fold}_m(\omega) = x$ *may use* x *as a visible normal-form token only together with the residual-fiber row* $\mathcal{F}_m(x)$ *or an explicit exactness certificate. In particular, equality of visible windows is not equality of raw histories.*

9346.2 Window-six Fibonacci ledger

Definition 9346.5 (Window-six ledger coordinates)**.** *For* $m = 6$*, define*

$$V_6 := |X_6|, \qquad B_6 := F_9, \qquad O_6 := F_{10}, \qquad P_6 := F_6, \qquad T_6 := F_7.$$

The visible stable-window count is

$$V_6 = F_8 = 21.$$

The boundary entries have the concrete values

$$F_6 = 8, \qquad F_7 = 13, \qquad F_8 = 21, \qquad F_9 = 34, \qquad F_{10} = 55.$$

Definition 9346.6 (Window-six residual denominator)**.** *The Window-six residual denominator is*

$$D_0 := F_9 + F_7.$$

Equivalently,

$$D_0 = F_{10} - F_6 = 47.$$

Theorem 9346.7 (Window-six residual denominator identity)**.** *The two Window-six residual readings agree:*

$$F_9 + F_7 = F_{10} - F_6 = 47.$$

Proof. The displayed Fibonacci values give $F_9 + F_7 = 34 + 13 = 47$. They also give $F_{10} - F_6 = 55 - 8 = 47$. Hence the two expressions name the same finite ledger denominator. □

Corollary 9346.8 (Boundary and tail readings share one ledger)**.** *The expression* $F_9 + F_7$ *reads the next boundary together with the deep tail, while* $F_{10} - F_6$ *reads the second outer boundary after removing the eightfold parity/carry defect. These are two descriptions of the same finite residual ledger* D_0*, not two independent invariants.*

Principle 9346.9 (Window-six residual closure)**.** *The Window-six residual ledger is allowed to close only from the three displayed tail layers*

$$21, \qquad 13, \qquad 9,$$

together with the eightfold boundary-parity row $8 = F_6$*. No host-side fit parameter, external physical constant, or unledgered continuous geometry may enter the residual denominator.*

Definition 9346.10 (Closed Window-six residual denominator)**.** *The closed Window-six residual denominator is*

$$D_6^\star := F_9 + F_7 + \varphi^{-(F_5+F_3)}\left[1 - \frac{F_2}{F_3}\varphi^{-(F_6+F_3)} + \frac{F_6}{F_6+F_2}\varphi^{-2(F_6+F_3)}\right].$$

Equivalently,

$$D_6^\star = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}.$$

Remark 9346.11 (The 8/9 closure coefficient)**.** *The coefficient* 8/9 *records the ratio between the eightfold boundary-parity row and the nine-dimensional invisible residual layer of the Window-six tail ledger. It is therefore part of the residual closure surface, not a free numerical tuning parameter.*

9346.3 Golden phase projection

Definition 9346.12 (Golden self-scale)**.** *The golden self-scale is the limiting Fibonacci ratio*

$$\varphi := \lim_{n\to\infty} \frac{F_{n+1}}{F_n},$$

read through the algebraic relation

$$\varphi = 1 + \varphi^{-1}.$$

Definition 9346.13 (Golden conjugate spectral readout)**.** *Let*

$$K_\varphi := \mathbb{Q}(\sqrt{5})$$

and let $\mathsf{Bin}_Z = \{b_0, b_1\}$ *be a displayed binary distinction carrier attached to the Axis-Zeckendorf fold surface. A* golden conjugate spectral readout *is a typed map*

$$\chi_\varphi : \mathsf{Bin}_Z \to K_\varphi$$

whose ordinary algebraic semantics sends

$$\chi_\varphi(b_0) = B_0 := \frac{1+\sqrt{5}}{2}, \qquad \chi_\varphi(b_1) = B_1 := \frac{1-\sqrt{5}}{2}.$$

The symbols b_0, b_1 *remain distinction markers. The rows* B_0, B_1 *are their spectral readings in the golden recursive layer, not kernel identities identifying the markers with algebraic numbers.*

Definition 9346.14 (Recursive operation carrier)**.** *Let* R *be a displayed record carrier with classifier* $\sim_R$*. A recursive operation carrier over* R *consists of an operation type* O*, an operation equivalence* $\equiv_O$*, a unit operation* id*, a composition law* $\circ$*, and an action*

$$\mathrm{act} : O \times R \to R$$

such that

$$\mathrm{id}(r) \sim_R r$$

and

$$(\beta \circ \gamma)(r) \sim_R \beta(\gamma(r)).$$

The carrier also records the associativity, unit, equivalence-compatibility, readback, ledger, and audit rows needed to use these equations inside a BEDC information layer.

Definition 9346.15 (Atomic recursive prime edge)**.** *For a recursive operation carrier over* R*, an atomic recursive prime edge from* B_0 *to* B_1 *is data*

$$(B_0, B_1, \alpha, \mathsf{Edge}_\alpha, \mathsf{NonDeg}, \mathsf{Irred}_\alpha, \mathsf{Prime}_\alpha)$$

where $\alpha \in O$ *and*

$$\mathsf{Edge}_\alpha : \alpha(B_0) \sim_R B_1,$$
$$\mathsf{NonDeg} : B_0 \not\sim_R B_1.$$

The irreducibility row says that

$$\alpha \equiv_O \beta \circ \gamma \Longrightarrow \beta \equiv_O \mathrm{id} \quad \textit{or} \quad \gamma \equiv_O \mathrm{id}\,.$$

The prime row is classifier-relative: it asserts primeness for the chosen operation divisibility relation, not ordinary integer primality.

Principle 9346.16 (Golden branch pair as a recursive edge)**.** *The spectral pair B_0, B_1 of Definition 9346.13 may be read as a seed and its first visible recursive exposure only after an operation carrier and an edge certificate are displayed:*

$$B_0 \xrightarrow{\alpha} B_1.$$

Under that reading, B_0 and B_1 are not ontic bottom points. They are visible branch records over source fibers, together with the certificate that the displayed operation α transports the first visible record to the second.

Theorem 9346.17 (Atomic edge has no nontrivial intermediate operation)**.** *Assume $B_0 \xrightarrow{\alpha} B_1$ is an atomic recursive prime edge and that*

$$\alpha \equiv_O \beta \circ \gamma.$$

Then any factorization

$$B_0 \xrightarrow{\gamma} C \xrightarrow{\beta} B_1$$

is trivial on at least one side:

$$\beta \equiv_O \mathrm{id} \quad \textit{or} \quad \gamma \equiv_O \mathrm{id}\,.$$

Proof. This is exactly the irreducibility row Irred_α in Definition 9346.15. If both β and γ were non-unit operations, the displayed equality $\alpha \equiv_O \beta \circ \gamma$ would contradict irreducibility. □

Definition 9346.18 (Undelimited atomic collision)**.** *Let α be an atomic recursive operation in a displayed operation carrier. A consecutive double occurrence is written*

$$\alpha \| \alpha.$$

It is undelimited *when the layer does not display a separator row*

$$\mathsf{Sep}_O(\alpha, \alpha)$$

certifying a boundary between the two occurrences. A collision atom for that undelimited occurrence is an operation $\beta \in O$ together with a classifier row

$$\alpha \| \alpha \sim_{\mathrm{coll}} \beta$$

and no audit row that factors the visible record through two separated α-steps.

Principle 9346.19 (Collision atoms are classifier-relative)**.** *A collision atom is not an assertion that β is definitionally equal to two applications of α. It says that the current readout layer cannot separate the visible record into two certified α-steps. The unresolved source fiber behind the visible record may still contain a two-step history, but that history is not available to the classifier unless a separator, clock, bracket, boundary, or replay certificate is displayed.*

Theorem 9346.20 (No separator gives no certified double read)**.** *Assume that a layer can certify a double-α reading only through a separator row $\mathsf{Sep}_O(\alpha, \alpha)$. If no such row is displayed for $\alpha \| \alpha$, then the layer cannot construct a certificate distinguishing the double occurrence from a collision atom β with the same collision classifier read.*

Proof. A certified double read requires a boundary witness separating the first occurrence of α from the second. By hypothesis the only such witness accepted by the layer is $\mathsf{Sep}_O(\alpha, \alpha)$. When this row is not displayed, the layer has only the composite visible record. If that record is classified by

$$\alpha \| \alpha \sim_{\text{coll}} \beta,$$

then the classifier supplies no further data with which to split it into two auditable α-steps. Thus the claimed distinction is not certified in the current layer. □

Principle 9346.21 (Zeckendorf-safe channels as collision discipline)**.** *The Zeckendorf-safe channel encoding is the finite channel model of this discipline. If the raw glyph* 1 *is read as the channel marker for one atomic action, a bare body substring* 11 *would be an undelimited occurrence of* $\alpha \| \alpha$*. The body substitution*

$$0 \mapsto 0, \qquad 1 \mapsto 10$$

adds a visible separator after every raw 1*, while the event terminator is*

$$\text{End} = 11.$$

By Definition 9208.6, no encoded body contains the collision pattern 11*; by Definition 9208.7, the first displayed* 11 *is therefore a boundary row rather than a body collision. The source-level carry classifier of Definition 9345.1 uses a different interface: it treats adjacent activation as a normalization trigger inside recognized history flows, not as a direct rewrite of channel glyphs.*

Definition 9346.22 (Recursive orbit and alpha valuation)**.** *Given an operation* α *acting on* R*, define the recursive orbit of* B_0 *by*

$$B_n := \alpha^n(B_0), \qquad \alpha^0 := \text{id}, \qquad \alpha^{n+1} := \alpha \circ \alpha^n.$$

The orbit is free when

$$\alpha^m(B_0) \sim_R \alpha^n(B_0) \Longrightarrow m = n.$$

For a subcarrier $X \subseteq R$*, an* α*-normal form is a unique display*

$$x \sim_R \alpha^k(y)$$

with $k \in \mathbb{N}$ *and* y *no longer left-divisible by* α*. The associated valuation row is*

$$v_\alpha(x) := k.$$

Theorem 9346.23 (Alpha valuation shifts additively)**.** *If* X *carries an* α*-normal form, then for every* $m \in \mathbb{N}$ *and* $x \in X$*,*

$$v_\alpha(\alpha^m(x)) = m + v_\alpha(x).$$

Proof. Let

$$x \sim_R \alpha^k(y)$$

be the α-normal form of x, so $k = v_\alpha(x)$ and y is not left-divisible by α. Applying α^m gives

$$\alpha^m(x) \sim_R \alpha^{m+k}(y).$$

By uniqueness of the normal form, this is the normal display of $\alpha^m(x)$. Hence

$$v_\alpha(\alpha^m(x)) = m + k = m + v_\alpha(x).$$

□

Principle 9346.24 (Classifier-relative primeness)**.** *The analogy*

$$p \text{ in multiplicative factorization} \qquad \sim \qquad \alpha \text{ in recursive operation composition}$$

is valid only after the classifier has been fixed. Ordinary primes are atoms for multiplicative divisibility. Recursive prime operations are atoms for operation composition. Zeckendorf carry rows, channel-safe body encodings, and golden spectral roots may interact with this pattern only through displayed bridge, readback, normal-form, and ledger certificates.

Concretized in §8890: The atomic recursive prime-edge carrier realizes the recursive-operation edge and irreducibility rows as finite NameCert data.

Definition 9346.25 (Golden conjugation center)**.** *Let $\sigma : K_\varphi \to K_\varphi$ be the conjugation determined by*

$$\sigma(\sqrt{5}) = -\sqrt{5}.$$

Equivalently, on the two roots of

$$x^2 - x - 1 = 0$$

the conjugation is the mirror

$$J_\varphi(x) := 1 - x.$$

The two spectral rows satisfy

$$\sigma(B_0) = B_1, \qquad \sigma(B_1) = B_0, \qquad J_\varphi(B_0) = B_1, \qquad J_\varphi(B_1) = B_0.$$

Their trace center is

$$c_\varphi := \frac{B_0 + B_1}{2}.$$

Theorem 9346.26 (Golden pair has half trace center)**.** *The spectral rows of Definition 9346.13 satisfy*

$$B_0 + B_1 = 1, \qquad B_0 B_1 = -1, \qquad B_1 = 1 - B_0, \qquad B_0 = 1 - B_1,$$

and hence

$$c_\varphi = \frac{1}{2}.$$

Proof. Substituting the displayed algebraic readings gives

$$B_0 + B_1 = \frac{1+\sqrt{5}}{2} + \frac{1-\sqrt{5}}{2} = 1$$

and

$$B_0 B_1 = \frac{(1+\sqrt{5})(1-\sqrt{5})}{4} = \frac{1-5}{4} = -1.$$

The two equations $B_1 = 1 - B_0$ and $B_0 = 1 - B_1$ are immediate from the sum equation. Therefore

$$c_\varphi = \frac{B_0 + B_1}{2} = \frac{1}{2}.$$

□

Principle 9346.27 (Spectral distinction boundary)**.** *The pair (B_0, B_1) is a conjugate spectral reading of a displayed binary distinction, not the binary distinction itself. A consumer may use it only together with the readout map χ_φ, the golden recursion certificate, the fold ledger, and the cannot-claim boundary of Definition 9346.38. Thus the spectral layer refines the distinction marker without replacing the underlying history carrier.*

Principle 9346.28 (Trace-center schema)**.** *The golden conjugate pair has the form*

$$B_0 = \frac{1}{2} + \frac{\sqrt{5}}{2}, \qquad B_1 = \frac{1}{2} - \frac{\sqrt{5}}{2}.$$

It is therefore a minimal algebraic instance of a moving conjugate orbit whose mirror $x \mapsto 1-x$ has fixed center $1/2$. This is the same half-center schema that appears in Definition 9414.160 and in the critical-line mirror of Definition 9414.163. The comparison is structural: it identifies a shared center-forming involution, not a proof of any zero-ledger exclusion statement.

Definition 9346.29 (Golden fixed-half interface)**.** *The golden fixed-half interface associated to Definition 9346.25 is the half-value layer*

$$\mathcal{H}_\varphi := (K_\varphi, b_\varphi, 1/2)$$

whose readout is the centered coordinate

$$b_\varphi(x) := \frac{x + J_\varphi(x)}{2}.$$

Its fixed-half fiber is

$$\mathrm{Fix}_{\mathcal{H}_\varphi} := \sum_{x:K_\varphi} (b_\varphi(x) = 1/2).$$

This fiber is the algebraic middle interface of the golden conjugate pair. It is not a third branch between B_0 and B_1.

Theorem 9346.30 (Golden branch pair reads back to the fixed-half fiber)**.** *Both golden spectral branches have the same fixed-half readback:*

$$b_\varphi(B_0) = \frac{1}{2}, \qquad b_\varphi(B_1) = \frac{1}{2}.$$

Proof. By Definition 9346.25, $J_\varphi(B_0) = B_1$ and $J_\varphi(B_1) = B_0$. Hence

$$b_\varphi(B_0) = \frac{B_0 + B_1}{2} = \frac{1}{2}$$

and similarly

$$b_\varphi(B_1) = \frac{B_1 + B_0}{2} = \frac{1}{2},$$

using Definition 9346.26. □

Definition 9346.31 (Golden residual cross-seed)**.** *Let $P_\varphi : T \to \{B_0, B_1\}$ be a displayed golden-branch readout. The residual fiber over a visible branch is*

$$\mathrm{Fib}_\varphi(B_i) := \sum_{t:T} (P_\varphi(t) = B_i).$$

A golden residual cross-seed *is a refinement row inside $\mathrm{Fib}_\varphi(B_i)$ whose next readout exposes the opposite branch B_{1-i} while retaining readback to the visible branch B_i at the current classifier.*

Principle 9346.32 (Conjugate fiber interface)**.** *The two visible golden branches are paired by the involution J_φ, while their middle interface is the fixed-half fiber* $\mathrm{Fix}_{\mathcal{H}_\varphi}$*. Residual cross-seeds record that a visible branch need not exhaust its source fiber: the opposite branch may remain available as refinement data, in the sense of the layer-fiber discipline of Definition 9414.41 and the information residue of Definition 9414.220.*

Principle 9346.33 (Critical-line analogue)**.** *For the RH mirror*

$$J(s) = 1 - \overline{s},$$

writing

$$s = \frac{1}{2} + \delta + i\tau$$

gives

$$J(s) = \frac{1}{2} - \delta + i\tau.$$

Thus the golden skew coordinate

$$u = \frac{\sqrt{5}}{2}$$

and the RH bias coordinate δ play the same formal role: the fixed interface is the zero-bias surface. In the golden layer this surface is the fixed-half fiber of $\mathcal{H}_\varphi$; in the RH layer it is the critical line. This analogy does not turn the golden branch model into a proof of the RH zero-ledger target.

Definition 9346.34 (Zeckendorf phase character and circumference certificate)**.** *The Zeckendorf phase character χ_φ^Z is the phase-circle interface whose ordinary continuous semantics is*

$$\chi_\varphi^Z = \cos^2\left(\frac{\pi}{\varphi}\right).$$

The phase-circumference certificate $\tau_Z^\uparrow$ is the circle-interface row whose ordinary continuous semantics is

$$\tau_Z^\uparrow = 2\pi.$$

These rows are certificates for a phase display. They are not finite kernel objects, not host trigonometric primitives, and not physical laws.

Definition 9346.35 (Closed Window-six cascade kernel)**.** *The closed Window-six cascade kernel is*

$$C_{Z6}^\star := \frac{F_2}{F_3} + \frac{F_2}{F_3^{F_3}}\chi_\varphi^Z + \frac{F_2}{D_6^\star \varphi^{F_5}}.$$

Under the ordinary continuous semantics of Definition 9346.34, this reads

$$C_6^\star = \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right) + \frac{1}{D_6^\star \varphi^5}.$$

Definition 9346.36 (Closed Window-six Zeckendorf-golden phase readout)**.** *The closed Window-six Zeckendorf-golden phase readout is*

$$R_{Z6}^\star := \tau_Z^\uparrow \varphi^{F_5} \frac{F_8 + F_9 C_{Z6}^\star}{F_8 \varphi^{-F_2} + F_9 C_{Z6}^\star \varphi^{-F_3}}.$$

In ordinary continuous notation this is

$$\alpha_{Z6}^{-1} = 2\pi\varphi^5 \frac{21 + 34C_6^\star}{21\varphi^{-1} + 34C_6^\star \varphi^{-2}}.$$

Definition 9346.37 (Golden phase projection)**.** *A golden phase projection for a visible window $x \in X_m$ is an external display assignment*

$$\Theta_m(x) := \exp(2\pi i\, \theta_m(x)),$$

where $\theta_m(x)$ is a rational or real readout obtained from the Fibonacci-value ledger of x. The projection may be used to compare visible periodicities, carry shadows, or residual-fiber weights. Its source remains the finite fold ledger

$$\Omega_m \xrightarrow{\mathsf{Fold}_m} X_m \xleftarrow{\mathcal{F}_m} \Omega_m.$$

Theorem 9346.38 (Golden phase is a derived readout)**.** *The golden phase projection is a derived interface over the Axis-Zeckendorf fold ledger. It does not identify raw histories, does not replace* ZCarry *by* hsame*, does not export field operations of* $\mathsf{Real}^{\uparrow}$ *or a circle carrier, and does not assert a physical law.*

Proof. By Definition 9346.37, the phase token $\Theta_m(x)$ is assigned only after the visible word x has been obtained from Fold_m. By Principle 9346.4, use of the visible word must retain the residual-fiber row or an exactness certificate. Therefore the phase token cannot collapse raw histories. The carry discipline of Theorem 9345.4 keeps carry classification separate from hsame, and the cannot-claim registry of chapter 9350 blocks promotion to $\mathsf{Real}^{\uparrow}$, a primitive dimension, or a replacement of $\mathsf{Nat}^{\uparrow}$. The projection is therefore only a display interface over finite ledger data. □

Corollary 9346.39 (Allowed use of the projection)**.** *A valid consumer may use Θ_m to organize visible Axis-Zeckendorf patterns, compare Window-six residual ledgers, or state a horizon bridge toward analytic phase language. Such a consumer must also carry the fold source, residual fiber, carry-separation row, and cannot-claim boundary.*

Theorem 9346.40 (Closed Window-six readout is a candidate, not a physical theorem)**.** *The readout $R^{\star}_{Z6}$ is the Window-six Zeckendorf-golden phase projection candidate. It is not, by itself, an equality with the low energy electromagnetic fine-structure constant, a derivation of a physical law, or a reconstruction of continuous spacetime.*

Proof. The construction of $R^{\star}_{Z6}$ uses the finite Window-six fold ledger, the residual closure principle, the phase-character row, and the phase-circumference certificate. These data close the derived-interface readout but do not supply a physical representation certificate. A physical identification must add a separate map

$$\rho_{\mathrm{phys}}(R^{\star}_{Z6}) = \alpha^{-1}_{\mathrm{EM}}(0),$$

where the right-hand side is the low-energy electromagnetic coupling readout. Without that representation certificate, the result remains a mathematical projection candidate inside the Axis-Zeckendorf vision. □

9346.4 Physical representation obligations

Definition 9346.41 (Low-energy electromagnetic representation certificate)**.** *A low-energy electromagnetic representation certificate for the closed Window-six readout in a closure mode μ is a separate physical map*

$$\rho_{\mathrm{EM}} : R^{\star}_{Z6,\mu} \longmapsto \alpha^{-1}_{\mathrm{EM}}(0),$$

where the target is the infrared electromagnetic coupling readout. The certificate must preserve the source fold, the residual-fiber ledger, the golden phase-character row, the boundary-parity row, and the cannot-claim boundary. It is not supplied by the numerical expression for $R^{\star}_{Z6,\mu}$ alone.

Principle 9346.42 (Impedance reading of the scalar)**.** *If the representation certificate of Definition 9346.41 is supplied, the same scalar may be read as the vacuum impedance ratio*

$$\alpha_{\mathrm{EM}}^{-1}(0) = \frac{2R_K}{Z_0},$$

where R_K is the von Klitzing resistance and Z_0 is the characteristic impedance of vacuum. This reading interprets the Zeckendorf-golden phase output as an impedance-matching surface for a low-energy electromagnetic phase medium, not as a primitive physical constant inserted into the construction.

Principle 9346.43 (Independent route convergence)**.** *An admitted physical representation of a selected $R^{\star}_{Z6,\mu}$ must be stable under independent low-energy measurement routes. Atomic recoil, electron magnetic-moment comparison, spectroscopic fine-structure readouts, and quantum-Hall impedance routes may differ in instrumentation and systematic error model, but after route-specific corrections they must converge to the same infrared scalar if they are to certify the same Window-six projection.*

Principle 9346.44 (Window refinement for running coupling)**.** *The Window-six readout is an infrared candidate. It does not assert that the same scalar is the effective electromagnetic coupling at all energy scales. A representation that reaches nonzero momentum transfer must replace the fixed denominator by a scale-indexed ledger*

$$D^{\star}_{6,\mu} \rightsquigarrow D(Q)$$

or replace the fixed window by a scale-indexed family

$$X_6 \rightsquigarrow X_{m(Q)}.$$

Thus any comparison with high-energy running of α is an obligation to construct a spectral-response activation map $z = z(Q)$ and a Zeckendorf window-refinement flow, not an allowed reuse of the infrared scalar.

Principle 9346.45 (Projection does not expose the hidden vacuum ledger)**.** *The scalar $\alpha_{\mathrm{EM}}^{-1}(0)$, even when represented by $R^{\star}_{Z6,\mu}$, exposes only a projection ratio. It does not expose the raw microstate space Ω_6, the full residual fibers $\mathcal{F}_6(x)$, or a unique continuous spacetime geometry. The recoverable object is the finite auditable skeleton*

$$(X_6, \mathsf{Fold}_6, \varphi, \chi^{Z}_{\varphi}, D^{\star}_{6,\mu}, \textit{boundary parity}, \textit{residual ledger}),$$

not the total physical substrate.

Principle 9346.46 (Hidden boundary parity)**.** *The eightfold boundary-parity row is not expected to appear as a large smooth correction in ordinary continuous low-energy probes. If it has a physical representation, its admissible signatures are restricted to integral, mod-2, boundary-condition, topology-sensitive, or coherent phase-residual observables. Ordinary smooth QED success therefore constrains the parity row to remain hidden from continuous probes; it does not by itself identify or eliminate the discrete row.*

Principle 9346.47 (Sixfold phase-circumference interface)**.** *The boundary-parity row enters electromagnetic readouts through a sixfold phase interface, not as three independently exposed parity bits. The admissible phase route has the form*

$$C_{\partial} \times T \longrightarrow \mathbb{Z}/6\mathbb{Z} \longrightarrow U(1),$$

where $T \cong \mathbb{Z}/3\mathbb{Z}$ is the edge-flux torsion source and the $C_\partial \cong (\mathbb{Z}/2\mathbb{Z})^3$ contribution is read through the invariant parity character. Equivalently, the fixed Window-six projection formula

$$R_6(D) = 2\pi\varphi^5 \frac{21 + 34C(D)}{21\varphi^{-1} + 34C(D)\varphi^{-2}}$$

may be written as

$$R_6(D) = 2\pi \frac{21 + 34C(D)}{21\varphi^{-6} + 34C(D)\varphi^{-7}}.$$

In this form φ^{-6} records the sixfold phase shell, φ^{-7} records the first boundary seam beyond that shell, and 2π is the circumference interface for the $Z_6 \to U(1)$ readout. This does not collapse the phase reservoir coefficient to the six-element image coefficient; it separates the admissible phase gate from the reservoir, image, and curvature-residual readout modes.

Principle 9346.48 (Window-six flux-operator geometry)**.** *The full edge-flux dynamics should not be read as an ordinary hidden metric geometry. At the stable-address level, the full edge-flux action saturates the operator algebra on the 21-dimensional visible Window-six space, and it preserves no nonzero invariant bilinear form and no nonzero invariant multiplication. Thus the physical representation cannot be based on a fixed hidden inner product, symplectic form, bracket, or commutative product. The natural object is instead a flux-operator geometry: stable Zeckendorf addresses, transition operators, phase gates, boundary escape flux, and golden-shell projection. For electromagnetic targets this supports an impedance or coupling readout rather than a pure curvature-scalar readout.*

Concretized in §8821: Flux-operator geometry is realized as finite stable-address, transition-operator, phase-gate, boundary escape-flux, golden-shell projection, refusal, transport, replay, provenance, and NameCert rows.

Principle 9346.49 (Right-block flux-reservoir coefficient)**.** *The phase-reservoir coefficient also has an edge-flux normalization inside Window six. Let U_R be the right edge-flux block in the Window-six partition of the binary cube. The edge-flux skeleton supplies*

$$|U_R| = 6, \qquad |\partial U_R| = 32.$$

Since Q_6 is 6-regular, the normalized probability of leaving the right block is

$$P(\text{leave } U_R \mid U_R) = \frac{|\partial U_R|}{6|U_R|} = \frac{32}{36} = \frac{8}{9}.$$

Thus the same coefficient 8/9 is not only the capacity ratio $|C_\partial|/N_{\geq 4}(6)$ and the light-to-heavy fiber ratio; it is also the canonical boundary escape flux of the right edge-flux block. The fold-gauge mismatch mean gives the one-way companion value

$$E[g_t] = \frac{4}{9}, \qquad 2E[g_t] = \frac{8}{9},$$

so the reservoir coefficient may also be read as the two-sided boundary flux associated with the one-way mismatch rate. This differs from the global cut-flux density

$$\Phi_R = \frac{|\partial U_R|}{64 \cdot 6} = \frac{1}{12},$$

which is normalized by the full directed-edge budget of the cube and is not a local residual coefficient in $D^{\star}_{6,\alpha}$. This is the coefficient relevant to impedance or coupling readouts. By Definition 9346.51, the same static right-boundary cut-flux source is a Dirichlet-exact cut-source and has zero 571-*Green class, so the infrared impedance selection does not require a separate φ^{-37} correction inside the static flux-reservoir layer. The dynamic Green-relaxation layer is a different readout mode, governed by a centered relaxation source rather than by the static cut vector. The* $2/3$ *branch records the direct six-element phase image and the* $4/9$ *branch records the curvature-residual quotient.*

Concretized in §9038: The right-block flux-reservoir coefficient is realized as finite boundary-count, cube-degree, normalized-ratio, static-cut, transport, replay, provenance, and NameCert rows.

Principle 9346.50 (Static Green-exactness of right-boundary flux)**.** *For any finite weighted graph decomposed into an internal block set I and a deleted boundary block B, let L_B be the Dirichlet reduced Laplacian on I and let b_B be the static cut-source whose ith coordinate is the total edge weight from $i \in I$ into B. Then*

$$L_B \mathbf{1}_I = b_B.$$

Indeed, the ith coordinate of $L_B\mathbf{1}_I$ is the full weighted degree of i minus the weights from i to the other retained vertices, which leaves exactly the cut weight from i to the deleted boundary. Thus every static Dirichlet cut-source is Laplacian-exact and has zero class in $\operatorname{coker}(L_B)$.

Order the Window-six edge-flux blocks as (U_2, U_1, U_L, U_R), with microscopic volumes

$$(|U_2|, |U_1|, |U_L|, |U_R|) = (27, 22, 9, 6).$$

The edge-count skeleton is

$$E = \begin{pmatrix} 28 & 63 & 23 & 20 \\ 63 & 21 & 21 & 6 \\ 23 & 21 & 2 & 6 \\ 20 & 6 & 6 & 2 \end{pmatrix}.$$

Deleting the right block gives the reduced Laplacian

$$L_R = \begin{pmatrix} 106 & -63 & -23 \\ -63 & 90 & -21 \\ -23 & -21 & 50 \end{pmatrix},$$

where each diagonal entry is the total edge count from the retained block to all other blocks. The right-boundary cut vector seen from the retained blocks is

$$b_R = \begin{pmatrix} 20 \\ 6 \\ 6 \end{pmatrix}.$$

It satisfies the integral exactness identity

$$L_R \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 20 \\ 6 \\ 6 \end{pmatrix} = b_R.$$

Thus the static right-boundary escape source is a reduced-Laplacian image by the general Dirichlet cut identity. It is a Green-exact source before any localization, and therefore also over $R_\zeta = \mathbb{Z}[1/6]$.

Theorem 9346.51 (Static-Green separation for Dirichlet cut-flux)**.** *The Window-six Green layer detected by the reduced Laplacian is nontrivial, but the static right-boundary cut-flux source does not activate it. More precisely,*

$$\det L_R = 123336 = 2^3 \cdot 3^3 \cdot 571,$$

so after localization at 6 *the non-invertible determinant factor is the* 571*-Green factor. The class of* b_R *in the corresponding cokernel is zero. Consequently, for the static low-energy electromagnetic readout whose Green input is the right-boundary cut source, the Green coefficient in*

$$D^G_{6,\alpha} = D^\star_{6,\alpha} + q_3\varphi^{-37} + O(\varphi^{-47})$$

is $q_3 = 0$*. This does not remove the* 571*-Green layer from dynamic readouts. For the coarse Markov relaxation kernel determined by the same edge skeleton,*

$$T = \begin{pmatrix} 28/81 & 7/18 & 23/162 & 10/81 \\ 21/44 & 7/22 & 7/44 & 1/22 \\ 23/54 & 7/18 & 2/27 & 1/9 \\ 5/9 & 1/6 & 1/6 & 1/9 \end{pmatrix},$$

with stationary weight

$$\pi = (27/64, 22/64, 9/64, 6/64),$$

the centered right-block observable

$$f_R = \mathbf{1}_R - \frac{3}{32}\mathbf{1}$$

has Green response

$$Zf_R = \begin{pmatrix} -58991/877056 \\ -41093/292352 \\ -69167/877056 \\ 821233/877056 \end{pmatrix}, \qquad Z = (I - T + \mathbf{1}\pi^T)^{-1}.$$

Since $877056 = 2^9 \cdot 3 \cdot 571$ *and* $292352 = 2^9 \cdot 571$*, this centered relaxation observable does read the* 571*-Green layer. Its Green susceptibility is*

$$\chi_R = \langle f_R, Zf_R\rangle_\pi = \frac{821233}{9355264} = \frac{821233}{2^{14} \cdot 571}.$$

The corresponding Green–Kubo response is

$$G_R = 2\chi_R - \mathrm{Var}_\pi(f_R) = \frac{423817}{4677632} = \frac{423817}{2^{13} \cdot 571}.$$

The static variance baseline is

$$\mathrm{Var}_\pi(f_R) = \frac{87}{1024}.$$

Therefore the fully activated Green-excess coefficient is

$$\Delta_R(1) = G_R - \mathrm{Var}_\pi(f_R) = \frac{26401}{4677632} = \frac{26401}{2^{13} \cdot 571}.$$

Proof. By Definition 9346.50, $b_R = L_R(1,1,1)^T$. This is the Window-six instance of the general Dirichlet cut identity $L_B\mathbf{1}_I = b_B$. Hence b_R is zero in the cokernel of L_R over $\mathbb{Z}$ and over R_ζ. The determinant computation displays the 571-Green layer after the powers of 2 and 3 have become units, but exact sources have zero class in that layer.

Equivalently, modulo 571 one may use the left cokernel functional

$$\ell = (448, 409, 1).$$

Then

$$\ell(b_R) = 448 \cdot 20 + 409 \cdot 6 + 6 = 11420 = 20 \cdot 571,$$

so $\ell(b_R) \equiv 0 \pmod{571}$. This modular vanishing is the residue shadow of the integral exactness identity above. Therefore the static right-boundary source contributes no φ^{-37} Green correction.

The relaxation statement is a direct computation from the displayed stochastic matrix. The row sums of T are 1, and $\pi T = \pi$ because the row weights are the block volumes divided by 64. Applying the fundamental Green operator $Z = (I - T + \mathbf{1}\pi^T)^{-1}$ to the centered right-block observable gives the displayed vector. Its denominators have one factor of 571, so the dynamic centered response is not killed by the static exactness identity. The displayed values of χ_R and G_R are obtained by the definitions $\chi_R = \sum_i \pi_i f_R(i)(Zf_R)(i)$ and $G_R = 2\chi_R - \sum_i \pi_i f_R(i)^2$; their denominators again contain the single 571 factor. Subtracting the static variance baseline isolates the Green-excess coefficient $\Delta_R(1)$. Thus $q_3 = 0$ is the static cut-flux conclusion for the infrared readout, while a running or relaxation response must use a separate Green-excess coefficient, for example

$$D_{6,\alpha}(z) = D^\star_{6,\alpha} + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}).$$

□

Definition 9346.52 (Window-six Green-excess seed)**.** *The Window-six Green-excess seed is the dynamic extension datum*

$$\mathsf{ZGFluxPhaseGreenSeed}_{6,\alpha} := \left(D^\star_{6,\alpha}, \Delta_R(z), \varphi^{-37}\right),$$

where $D^\star_{6,\alpha}$ is the static cut-flux denominator and $\Delta_R(z)$ is the centered right-boundary Green-excess kernel

$$\Delta_R(z) = -\frac{3z(1595z^2 + 24477z + 26730)}{512(55z^3 + 506z^2 - 7263z - 48114)}.$$

It satisfies

$$\Delta_R(0) = 0, \qquad \Delta_R(1) = \frac{26401}{2^{13} \cdot 571}.$$

Near the static denominator the readout derivative is approximately

$$\left.\frac{dR}{dD}\right|_{D=D^\star_{6,\alpha}} \approx -0.00120603474199364, \qquad \varphi^{-37} \approx 1.85121691873052 \cdot 10^{-8}.$$

Thus full Window-six Green-excess activation changes the inverse coupling by approximately

$$-1.26 \cdot 10^{-13}.$$

The seed does not replace the infrared denominator

$$D^\star_{6,\alpha} = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}.$$

It records the first local Green-excess extension

$$D_{6,\alpha}(z) = D^\star_{6,\alpha} + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}).$$

Definition 9346.53 (Window-six spectral response family)**.** *The Window-six spectral response family is the one-parameter readout obtained by adjoining the Green-excess kernel to the static phase-reservoir denominator:*

$$D_{6,\alpha}(z) = D^{\star}_{6,\alpha} + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}),$$

where

$$C_{6,\alpha}(z) = \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right) + \frac{1}{D_{6,\alpha}(z)\varphi^5}$$

and

$$\alpha^{-1}_{Z6}(z) = 2\pi\frac{21 + 34C_{6,\alpha}(z)}{21\varphi^{-6} + 34C_{6,\alpha}(z)\varphi^{-7}}.$$

The parameter z is a Window-six Green-relaxation activation parameter, not yet a physical momentum scale. A physical running interpretation must supply a map $z = z(Q)$ and, beyond the Window-six neighbourhood, a refinement flow $X_6 \rightsquigarrow X_{m(Q)}$.

Theorem 9346.54 (Spectral response interpolation)**.** *The family of Definition 9346.53 interpolates between the static cut-flux readout and the first Window-six Green-excess readout:*

$$\alpha^{-1}_{Z6}(0) = 137.035999177006279014596667\ldots,$$

and

$$\alpha^{-1}_{Z6}(1) = 137.035999177006153002727216\ldots.$$

Moreover $\Delta_R(z)$ is a rational spectral response of the four-state Window-six Markov kernel. The characteristic polynomial is

$$\chi_T(t) = (t-1)\frac{48114t^3 + 7263t^2 - 506t - 55}{48114},$$

and the denominator

$$55z^3 + 506z^2 - 7263z - 48114$$

in $\Delta_R(z)$ is the reciprocal form of the non-stationary cubic factor. Thus the 571 denominator in the fully activated Green-excess coefficient is the arithmetic trace of the Window-six Green spectrum, not a static correction to $\alpha^{-1}_{Z6}(0)$.

Proof. At $z = 0$, Definition 9346.52 gives $\Delta_R(0) = 0$, so $D_{6,\alpha}(0) = D^{\star}_{6,\alpha}$ and the readout is the static cut-flux phase-reservoir value. At $z = 1$, the same definition gives

$$\Delta_R(1) = \frac{26401}{2^{13}\cdot 571},$$

which yields the displayed endpoint after substitution into Definition 9346.53.

For monotonicity, differentiate the displayed rational formula for $\Delta_R(z)$:

$$\Delta'_R(z) = \frac{3(539165z^4 + 26109270z^3 + 421527321z^2 + 2355372756z + 1286087220)}{512(55z^3 + 506z^2 - 7263z - 48114)^2}.$$

The derivative is positive on $0 \le z \le 1$. Since Definition 9346.88 shows that the readout $R(D)$ decreases with D, activating Green-excess decreases α^{-1}_{Z6} and increases the coupling. The characteristic polynomial statement follows by computing the characteristic polynomial of the displayed coarse Markov matrix T in Definition 9346.51. The reciprocal cubic identity is exactly

$$55z^3 + 506z^2 - 7263z - 48114 = -z^3\left(48114z^{-3} + 7263z^{-2} - 506z^{-1} - 55\right).$$

□

Theorem 9346.55 (Golden–Markov audit field)**.** *Let λ be any non-stationary eigenvalue of the Window-six coarse Markov kernel T, so that*

$$48114\lambda^3 + 7263\lambda^2 - 506\lambda - 55 = 0.$$

The dynamic spectral response is naturally read over the compositum

$$K_6^{\mathrm{audit}} = \mathbb{Q}(\varphi, \lambda).$$

The golden phase scale contributes the quadratic field $\mathbb{Q}(\varphi) = \mathbb{Q}(\sqrt{5})$, while the non-stationary Markov spectrum contributes the cubic field $\mathbb{Q}(\lambda)$. When these two fields are linearly disjoint, K_6^{audit} has degree 6 over $\mathbb{Q}$ with basis

$$1,\ \varphi,\ \lambda,\ \lambda\varphi,\ \lambda^2,\ \lambda^2\varphi.$$

Thus the Window-six response separates into a static golden phase layer and a dynamic cubic Markov spectral layer. The $Z_6 \to U(1)$ phase gate, the shell φ^{-6}, and the degree-six audit field are three readings of the same two-by-three structure: quadratic golden scale times cubic Markov spectrum.

Proof. The golden field has degree 2 because φ satisfies $x^2 - x - 1 = 0$. The Markov spectral field is generated by a root of the non-stationary cubic factor displayed in Definition 9346.54. Linear disjointness gives the product degree $2 \cdot 3 = 6$ and the displayed tensor-product basis. The static readout uses only the golden phase shell and the Z_6 phase gate. The dynamic Green-excess denominator uses the reciprocal cubic spectral factor. Hence the unified spectral response lives over the compositum rather than over the golden field alone. □

Corollary 9346.56 (Small-activation Green-excess)**.** *Near the static point,*

$$\Delta_R(z) = \frac{5}{1536}z + \frac{3407}{1368576}z^2 - \frac{179639}{1219401216}z^3 + O(z^4).$$

Consequently the fixed Window-six Green-excess is a small infrared seed: the leading dynamic contribution to the inverse coupling is proportional to $-(5/1536)z\varphi^{-37}$. A full physical running account must still provide the scale map $z = z(Q)$ and the cross-window refinement $X_6 \rightsquigarrow X_{m(Q)}$.

Principle 9346.57 (Beta-sign selection for running deformations)**.** *For any candidate physical running deformation of the fixed infrared denominator, write*

$$D(t) = D^{\star}_{6,\alpha} + \delta D(t),$$

where t is any monotone activation coordinate for increasing electromagnetic scale. If this deformation is to represent the physical running direction of the electromagnetic coupling, then it must satisfy

$$\frac{d}{dt}D(t) \geq 0$$

on its activation interval. Since Definition 9346.88 gives $dR/dD < 0$, this condition is exactly the inverse-coupling sign condition

$$\frac{d}{dt}\alpha^{-1}(t) \leq 0, \qquad \frac{d}{dt}\alpha(t) \geq 0.$$

Thus the beta-sign test is structural: it depends on the direction of the denominator deformation, not on fitting the infrared value.

Theorem 9346.58 (Walking branch is not the physical running direction)**.** *The walking-gate phase-reservoir denominator*

$$D^{\mathrm{res}}_{6,\mathrm{walk}} = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-29}$$

cannot be the primary physical running deformation of the Window-six electromagnetic readout. It is a cross-window refinement branch, not the local beta seed for increasing electromagnetic scale.

Proof. The static low-energy phase-reservoir denominator is

$$D^{\star}_{6,\alpha} = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}.$$

Since $\varphi^{-29} < \varphi^{-27}$, one has

$$D^{\mathrm{res}}_{6,\mathrm{walk}} < D^{\star}_{6,\alpha}.$$

By Definition 9346.88, the readout $R(D)$ is strictly decreasing in D. Therefore

$$R(D^{\mathrm{res}}_{6,\mathrm{walk}}) > R(D^{\star}_{6,\alpha}),$$

so the walking deformation raises α^{-1} instead of lowering it. It fails the beta-sign selection principle Definition 9346.57. The same branch may still record boundary-uplift refinement data, because the transition from $\Delta_6^{\mathrm{bdry}} = \{0, F_9\}$ to the three-branch boundary differences

$$\Delta_7^{\mathrm{bdry}} = \{0, F_{10}, F_{11}\}, \qquad \Delta_8^{\mathrm{bdry}} = \{0, F_{11}, F_{12}\}$$

exhibits the Fibonacci second-difference channel. That channel requires protocol-gated window refinement before it can become a physical running law. □

Concretized in §8846: The finite BetaSignSelection packet records the denominator-sign row, Green-excess source, window-six beta seed, Zeckendorf sibling visibility, and local NameCert boundary for this principle.

Theorem 9346.59 (Green-excess passes the beta-sign test)**.** *The centered Green-excess deformation*

$$D_{6,\alpha}(z) = D^{\star}_{6,\alpha} + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47})$$

has the correct physical running sign on the activation interval $0 < z \leq 1$.

Proof. By Definition 9346.54, $\Delta_R(0) = 0$, $\Delta_R(z) > 0$ for $0 < z \leq 1$, and $\Delta_R'(z) > 0$ on $0 \leq z \leq 1$. Hence the Green-excess deformation increases the denominator away from the static point. Applying Definition 9346.57 and Definition 9346.88 gives

$$\alpha_{Z6}^{-1}(z) < \alpha_{Z6}^{-1}(0) \qquad (0 < z \leq 1),$$

so the coupling itself increases along the activation. The physically admissible local seed is therefore the 571-Green Green-excess branch, while the walking branch remains a protocol-gated cross-window refinement signal. □

Theorem 9346.60 (Window-six internal beta seed)**.** *Define the inverse-coupling beta seed by*

$$\beta_6^{-1}(z) := \frac{d}{dz}\alpha_{Z6}^{-1}(z).$$

On the activation interval $0 \le z \le 1$,

$$\beta_6^{-1}(z) < 0.$$

Near the static point,

$$\alpha_{Z6}^{-1}(z) = \alpha_{Z6}^{-1}(0) - 7.26768202785 \cdot 10^{-14} z - 5.55802304576 \cdot 10^{-14} z^2 + 3.28905498899 \cdot 10^{-15} z^3 + O(z^4),$$

so

$$\beta_6^{-1}(0) = -7.26768202785 \cdot 10^{-14}.$$

Equivalently, for the coupling itself,

$$\alpha_{Z6}(z) = \alpha_{Z6}(0) + 3.87013911681 \cdot 10^{-18} z + 2.95972255241 \cdot 10^{-18} z^2 + O(z^3).$$

The fixed Window-six beta seed therefore has the physically correct sign, is extremely small, and is internal to the Green-excess spectral response.

Proof. By Definition 9346.54, $\Delta_R'(z) > 0$ on $0 \le z \le 1$, while Definition 9346.88 gives $dR/dD < 0$. Hence

$$\frac{d}{dz}\alpha_{Z6}^{-1}(z) = \frac{dR}{dD}\frac{dD_{6,\alpha}(z)}{dz} < 0.$$

The displayed series follows by substituting the small-z expansion of Definition 9346.56 into the readout of Definition 9346.53. The positive sign of the α_{Z6} expansion follows by differentiating the reciprocal of $\alpha_{Z6}^{-1}(z)$ at the static point. □

Theorem 9346.61 (Window-six beta seed is infrared stable)**.** *The Green-excess denominator*

$$55z^3 + 506z^2 - 7263z - 48114$$

has real roots approximately

$$-14.1627969182, \qquad -5.7602517924, \qquad 10.7230487106.$$

Therefore the spectral response has no pole on $0 \le z \le 1$. *The fixed Window-six beta seed is an infrared-stable local running seed, not a complete high-energy running law.*

Proof. The roots are the reciprocals of the non-stationary eigenvalues of the coarse Markov kernel in Definition 9346.54. All three lie outside the activation interval. Hence the rational Green-excess kernel is analytic on $0 \le z \le 1$. Since the endpoint shift Definition 9346.52 is of order 10^{-13} in inverse coupling, any large-scale running account must come from the missing map $z = z(Q)$ together with cross-window refinement, not from a pole internal to Window six. □

Concretized in §9051: The Window-six beta seed packet realizes the inverse-coupling derivative and infrared-stability rows as finite NameCert data.

Principle 9346.62 (Phase-shell running coordinate)**.** *The fixed Window-six denominator response and the phase-shell response are distinct running coordinates. The intra-window coordinate changes D while the golden phase shell remains fixed. The phase-shell coordinate changes the exponent of the golden radius itself. Write*

$$R_6(\sigma, D) := 2\pi \frac{21 + 34C(D)}{21\varphi^{-\sigma} + 34C(D)\varphi^{-(\sigma+1)}},$$

where

$$C(D) := \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right) + \frac{1}{D\varphi^5}.$$

The infrared Window-six readout is the point

$$(\sigma, D) = (6, D^\star_{6,\alpha}).$$

For fixed D, the readout satisfies

$$R_6(\sigma, D) = R_6(6, D)\varphi^{\sigma-6}.$$

Thus σ is the golden phase-shell exponent, while D is the residual denominator coordinate inside a fixed shell.

Concretized in §9092: Axis-Zeckendorf running coordinates are realized as finite shell-exponent, denominator-response, readout, transport, provenance, and NameCert rows.

Theorem 9346.63 (Fixed Window-six denominator running is infrared-sized)**.** *Within the fixed Window-six phase shell, any monotone increase of D has a finite lower endpoint*

$$R_{6,\min} = 2\pi \frac{21 + 34C_\infty}{21\varphi^{-6} + 34C_\infty\varphi^{-7}}, \qquad C_\infty = \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right).$$

Numerically,

$$R_{6,\min} = 136.979203197491575\ldots,$$

whereas

$$R_6(6, D^\star_{6,\alpha}) = 137.035999177006279\ldots.$$

Consequently fixed-shell denominator running can lower the inverse coupling by at most

$$0.056795979514704\ldots.$$

It is therefore an infrared-sized response, not a full electroweak-scale running mechanism.

Proof. For fixed phase shell $\sigma = 6$, the only dependence on D enters through

$$\frac{1}{D\varphi^5}.$$

As D increases, this term decreases to zero, so $C(D)$ decreases to C_∞. Since Definition 9346.88 gives $dR/dD < 0$, the displayed value is the limiting lower endpoint for the fixed-shell denominator response. The numerical drop is the difference between the infrared readout and this limiting endpoint. □

Definition 9346.64 (Golden phase-shell anomalous dimension)**.** *A physical running path for the Window-six electromagnetic readout is a map*

$$Q \longmapsto (\sigma(Q), z(Q)),$$

with infrared base point

$$\sigma(0) = 6, \qquad z(0) = 0.$$

The golden phase-shell anomalous dimension is

$$\gamma_\varphi(Q) := 6 - \sigma(Q).$$

The two-coordinate running family is

$$\alpha_{Z6}^{-1}(\sigma, z) := 2\pi \frac{21 + 34C_{6,\alpha}(z)}{21\varphi^{-\sigma} + 34C_{6,\alpha}(z)\varphi^{-(\sigma+1)}},$$

where

$$C_{6,\alpha}(z) = \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right) + \frac{1}{D_{6,\alpha}(z)\varphi^5}$$

and

$$D_{6,\alpha}(z) = D_{6,\alpha}^\star + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}).$$

At fixed $z = 0$, the anomalous dimension recodes an inverse-coupling ratio by

$$\gamma_\varphi(Q) = \log_\varphi\left(\frac{\alpha_{Z6}^{-1}(0)}{\alpha_{Z6}^{-1}(Q)}\right).$$

Definition 9346.65 (Effective observable shell index)**.** *Let*

$$A := F_8 = 21, \qquad B := F_9 = 34,$$

and put $C_\star := C(D_{6,\alpha}^\star)$. For any denominator D in the fixed Window-six readout, define the denominator-induced shell shift

$$\delta\sigma_D(D) := \log_\varphi\left[\frac{A + BC(D)}{A + BC_\star} \cdot \frac{A + BC_\star\varphi^{-1}}{A + BC(D)\varphi^{-1}}\right].$$

Then

$$R_6(\sigma, D) = R_0\varphi^{\sigma - 6 + \delta\sigma_D(D)}.$$

The externally visible shell index carried by the single scalar readout is therefore

$$\sigma_{\text{eff}}(Q) := \sigma(Q) + \delta\sigma_D(D_{6,\alpha}(z(Q))),$$

with effective anomalous dimension

$$\gamma_{\text{eff}}(Q) := 6 - \sigma_{\text{eff}}(Q) = \log_\varphi\left(\frac{\alpha_{Z6}^{-1}(0)}{\alpha_{Z6}^{-1}(Q)}\right).$$

Theorem 9346.66 (Observable shell compression)**.** *The scalar inverse coupling $\alpha_{Z6}^{-1}(Q)$ uniquely determines $\sigma_{\text{eff}}(Q)$, or equivalently $\gamma_{\text{eff}}(Q)$, but it does not by itself uniquely determine the internal split between $\sigma(Q)$ and $z(Q)$.*

Proof. The identity in Definition 9346.65 rewrites the two-coordinate readout as

$$\alpha_{Z6}^{-1}(Q) = R_0\varphi^{\sigma_{\text{eff}}(Q)-6} = R_0\varphi^{-\gamma_{\text{eff}}(Q)}.$$

Hence the measured scalar determines the single combination

$$\gamma_{\text{eff}}(Q) = \log_\varphi\left(\frac{R_0}{\alpha_{Z6}^{-1}(Q)}\right),$$

but the equation

$$\sigma_{\text{eff}}(Q) = \sigma(Q) + \delta\sigma_D(D_{6,\alpha}(z(Q)))$$

has more than one internal coordinate. A separate observable sensitive to the Green susceptibility, phase relaxation, boundary parity, or a coherent topological response is needed to distinguish the intra-window Green ledger from the principal phase-shell coordinate. □

Corollary 9346.67 (Green-excess as an effective shell offset)**.** *For the Window-six Green-excess denominator*

$$D_{6,\alpha}(z) = D_{6,\alpha}^\star + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}),$$

define

$$\gamma_D(z) := -\delta\sigma_D(D_{6,\alpha}(z)).$$

Near the static point,

$$\gamma_D(z) = 1.1021 \cdot 10^{-15}z + O(z^2),$$

and at full Window-six activation,

$$\gamma_D(1) = 1.91 \cdot 10^{-15}.$$

Thus the 571*-Green-excess layer is visible in the scalar coupling only as an extremely small effective shell offset.*

Principle 9346.68 (Infrared boundary condition and shell flow)**.** *The static Window-six denominator*

$$D_{6,\alpha}^\star = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}$$

is the infrared boundary condition for the electromagnetic readout. It sets

$$R_0 := \alpha_{Z6}^{-1}(0) = 137.035999177006279014596667\ldots.$$

The scale dependence is not supplied by replacing this boundary condition with a large denominator deformation. It is supplied by a path

$$Q \longmapsto (\sigma(Q), z(Q)),$$

where $z(Q)$ is the intra-window Green coordinate and $\sigma(Q)$ is the principal phase-shell coordinate. Thus the Window-six construction separates the boundary value from the renormalization flow.

Theorem 9346.69 (Phase-shell exponent is the principal running coordinate)**.** *The fixed Window-six Green-excess coordinate $z(Q)$ supplies a local intra-window response. A physical running account whose inverse coupling changes by electroweak-scale size must include a nonconstant phase-shell exponent $\sigma(Q)$.*

Proof. By Definition 9346.63, fixed-shell denominator running is bounded by a drop of $0.056795979514704\ldots$ in α^{-1}. By Definition 9346.60, the centered Green-excess seed itself is still smaller, of order 10^{-13} in the inverse coupling near the infrared point. These bounds are local Window-six responses.

By contrast, changing the phase-shell exponent gives

$$R_6(\sigma, D^\star_{6,\alpha}) = R_6(6, D^\star_{6,\alpha})\varphi^{\sigma-6}.$$

Using the electroweak calibration value $\alpha^{-1}(M_Z^2) \approx 128.946$ as a scale diagnostic gives

$$\sigma(M_Z) = 6 + \log_\varphi\left(\frac{128.946}{137.035999177006279}\right) = 5.873548521075\ldots,$$

and hence

$$\gamma_\varphi(M_Z) = 0.126451478925\ldots.$$

Thus the large running component is naturally represented as a renormalization of the golden phase-shell radius. The denominator coordinate remains the local Green response inside that shell. □

Corollary 9346.70 (Shell-flow sign)**.** *If D is held fixed to first order and*

$$R(Q) = R(0)\varphi^{\sigma(Q)-6},$$

then

$$\frac{dR}{d\log Q} = R(Q)\log\varphi\,\frac{d\sigma}{d\log Q}.$$

Therefore the physical sign $d\alpha^{-1}/d\log Q < 0$ requires

$$\frac{d\sigma}{d\log Q} < 0, \qquad \frac{d\gamma_\varphi}{d\log Q} > 0.$$

Energy growth is therefore read as thinning of the effective golden phase shell before any integer-window transition is invoked. Integer window refinement enters as a thresholded refinement of this continuous phase-shell flow.

Theorem 9346.71 (Renormalization dictionary for the golden shell flow)**.** *Let*

$$\beta_\alpha(\alpha, Q) := \frac{d\alpha}{d\log Q}.$$

On any interval where the physical inverse coupling is represented by the shell coordinate with D fixed to first order,

$$\alpha^{-1}(Q) = R_0\varphi^{-\gamma_\varphi(Q)},$$

the standard beta function and the golden shell flow are related by

$$\frac{d\gamma_\varphi}{d\log Q} = \frac{\beta_\alpha(\alpha, Q)}{\alpha\log\varphi}.$$

Equivalently,

$$\frac{d\sigma}{d\log Q} = -\frac{\beta_\alpha(\alpha, Q)}{\alpha\log\varphi}.$$

Proof. Differentiate $R(Q) = \alpha^{-1}(Q)$. Since

$$\frac{dR}{d\log Q} = -\alpha^{-2}\frac{d\alpha}{d\log Q},$$

while

$$R(Q) = R_0\varphi^{-\gamma_\varphi(Q)}$$

gives

$$\frac{dR}{d\log Q} = -R(Q)\log\varphi\,\frac{d\gamma_\varphi}{d\log Q},$$

and $R(Q) = \alpha^{-1}(Q)$, cancellation gives the displayed formula for $d\gamma_\varphi/d\log Q$. The identity for $d\sigma/d\log Q$ follows from $\sigma = 6 - \gamma_\varphi$. □

Corollary 9346.72 (Abelian one-loop shell flow)**.** *In an abelian one-loop regime with active charged fermions contributing*

$$\beta_\alpha(\alpha, Q) = \frac{2}{3\pi}\left(\sum_f N_c(f)q_f^2\right)\alpha^2,$$

the golden shell flow satisfies

$$\frac{d\gamma_\varphi}{d\log Q} = \frac{2}{3\pi\log\varphi}\left(\sum_f N_c(f)q_f^2\right)\alpha.$$

Hence the shell anomalous dimension increases whenever the abelian beta function is positive, and the phase-shell exponent decreases:

$$\frac{d\gamma_\varphi}{d\log Q} > 0, \qquad \frac{d\sigma}{d\log Q} < 0.$$

Charged-particle thresholds change the coefficient in the sum and therefore change the shell-flow slope. They do not alter the role of $D^\star_{6,\alpha}$ as the infrared boundary condition.

Principle 9346.73 (Three coefficient strata)**.** *The three Window-six coefficients*

$$\frac{8}{9}, \qquad \frac{2}{3}, \qquad \frac{4}{9}$$

are not competing fits for the same low-energy scalar. They occupy different physical strata of the same finite readout geometry. The phase-reservoir coefficient

$$q^{\text{imp}}_{\text{static}} = \frac{8}{9}$$

is the static impedance coefficient selected by the right-boundary flux-reservoir readout. The image coefficient

$$q^{Z_6}_\beta = \frac{2}{3}$$

is the six-element $Z_6 \to U(1)$ phase-image coefficient when that image is read as a unit electromagnetic beta-flow activation. The one-way coefficient

$$q^{\text{one}}_{\text{curv}} = \frac{4}{9}$$

is the irreducible curvature-residual or one-way fold-mismatch coefficient.

Theorem 9346.74 (Window-six readout diamond)**.** *The Window-six readout coefficients form the three-layer diamond*

$$q_{\mathrm{curv}} = \frac{4}{9} \quad \longrightarrow \quad q_\beta = \frac{6}{9} = \frac{2}{3} \quad \longrightarrow \quad q_{\mathrm{imp}} = \frac{8}{9}.$$

The common step is

$$\frac{2}{9} = \frac{|P_6^{\mathrm{geo}}|}{N_{\geq 4}(6)}.$$

Thus the layers differ by activation of one basic Z_2 parity unit against the deepest residual-tail denominator.

Proof. The curvature-residual layer quotients the locally absorbable diagonal parity

$$P_6^{\mathrm{geo}} \cong \mathbb{Z}/2\mathbb{Z}$$

from the boundary reservoir, so its visible residual group has order 4 and coefficient $4/N_{\geq 4}(6) = 4/9$. The $U(1)$ phase-image layer uses the six-element image of the $C_\partial \times T \to \mathbb{Z}/6\mathbb{Z} \to U(1)$ interface, so its coefficient is $6/9 = 2/3$. The static impedance layer reads the full boundary phase reservoir $C_\partial \cong (\mathbb{Z}/2\mathbb{Z})^3$, so its coefficient is $8/9$. Since the orders are $4, 6, 8$, the successive normalized differences are both $2/9$. □

Theorem 9346.75 (Static impedance selects 8/9 while beta flow selects 2/3)**.** *The low-energy denominator*

$$D^\star_{6,\alpha} = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}$$

uses $8/9$ because the infrared electromagnetic scalar is an impedance readout of the phase reservoir. The abelian unit-charge shell flow uses $2/3$ because the running layer reads the $Z_6 \to U(1)$ phase image.

Proof. By Definition 9346.93, the fixed infrared electromagnetic readout selects the phase-reservoir coefficient $q_2 = 8/9$, not the image coefficient or the curvature quotient. By Definition 9346.49, the same $8/9$ is the normalized right-block escape flux $P(\text{leave } U_R \mid U_R) = 32/36$. This is the static impedance layer.

By contrast, the $Z_6 \to U(1)$ image has six phase images over the Window-six depth denominator $N_{\geq 4}(6) = 9$, hence coefficient $6/9$. In the shell-flow dictionary of Definition 9346.71, an abelian unit Dirac charge contributes

$$\frac{d\gamma_\varphi}{d\log Q} = \frac{2}{3}\frac{\alpha}{\pi \log \varphi}.$$

Thus the $2/3$ image coefficient is naturally assigned to the beta-flow layer rather than to the static denominator. □

Definition 9346.76 (Window-six beta-shell coefficient)**.** *The unit Z_6 beta-shell coefficient is*

$$q_\beta^{Z_6} := \frac{2}{3}.$$

For a scale interval with active electromagnetic modes, write

$$\mathcal{B}(Q) = q_\beta^{Z_6} \sum_f N_c(f) q_f^2 \Theta_f(Q) + \mathcal{B}_{\mathrm{bos}}(Q) + \mathcal{B}_{\mathrm{had}}(Q),$$

where $\Theta_f(Q)$ records threshold activation, $N_c(f)$ is the color multiplicity, and q_f is the electric charge in units of the positron charge. The corresponding golden shell flow is

$$\frac{d\gamma_{\text{eff}}}{d\log Q} = \frac{\alpha(Q)}{\pi \log \varphi}\,\mathcal{B}(Q).$$

Corollary 9346.77 (Coefficient layer separation)**.** *The low-energy static layer, the beta-shell layer, and the Green susceptibility layer are separated as follows:*

$$\frac{8}{9} \quad \textit{sets the static flux-reservoir boundary condition,}$$

$$\frac{2}{3} \quad \textit{sets the unit } Z_6 \textit{ phase-image beta coefficient,}$$

and

$$571 \quad \textit{certifies the internal Green susceptibility denominator.}$$

The curvature coefficient $4/9$ *remains the one-way mismatch or curvature-residual coefficient. It does not replace either the static impedance coefficient or the unit beta-flow coefficient.*

Concretized in §9120: The coefficient-strata readout route is realized as finite Zeckendorf carry, running-coordinate, static boundary-flux, transport, replay, provenance, and NameCert rows.

Principle 9346.78 (Pre-quantum phase-projection reading)**.** *The Window-six Zeckendorf readout has a superposition-shaped record geometry, but it is not yet a Hilbert-space quantum model. The visible stable state*

$$x \in X_6$$

is the image of a residual fiber

$$\mathcal{F}_6(x) = \mathsf{Fold}_6^{-1}(x) \subseteq \Omega_6, \qquad \Omega_6 = \{0,1\}^6.$$

Thus a visible Zeckendorf address is not a single raw microstate; it is the public projection of a finite bundle of raw binary histories. The additional phase gate

$$C_\partial \times T \longrightarrow \mathbb{Z}/6\mathbb{Z} \longrightarrow U(1)$$

turns this fold-fiber ledger into a discrete phase projection surface.

Theorem 9346.79 (Fine-structure readout as projected phase record)**.** *The scalar α_{Z6}^{-1} is a projected readout of the Window-six fold-fiber, phase-gate, boundary-flux, and Green-response ledger. It is not a primitive carrier of the raw microstates.*

Proof. The visible component is supplied by the fold $\mathsf{Fold}_6 : \Omega_6 \to X_6$, whose fibers are retained by Definition 9346.3. The phase component is the $Z_6 \to U(1)$ gate of Definition 9346.47. The static boundary component is the right-block flux-reservoir coefficient of Definition 9346.49. The dynamic response component is the centered Green-excess layer of Definition 9346.52. The readout formula combines these records into a single scalar inverse coupling. Therefore the scalar is downstream from a finite record surface rather than identical to any one raw binary history. □

Concretized in §8811: The phase-shell packet realizes the fold-fiber, discrete phase gate, boundary-flux, and Green-response rows as finite NameCert data.

Theorem 9346.80 (Born-like two-shell radius projection)**.** *Let $C > 0$ and let $\mathcal{H}_{\text{shell}}$ be the two-dimensional complex carrier with orthonormal basis $|6\rangle, |7\rangle$. For any relative shell phase $\eta \in \mathbb{R}$ define*

$$|\Psi_C(\eta)\rangle = \frac{\sqrt{21}\,|6\rangle + e^{i\eta}\sqrt{34C}\,|7\rangle}{\sqrt{21+34C}}.$$

Let the diagonal radius observable be

$$\widehat{r} = \varphi^{-6}|6\rangle\langle 6| + \varphi^{-7}|7\rangle\langle 7|.$$

Then

$$\langle \Psi_C(\eta)|\widehat{r}|\Psi_C(\eta)\rangle = \frac{21\varphi^{-6} + 34C\varphi^{-7}}{21+34C}.$$

Consequently, for the Window-six spectral family of Definition 9346.53,

$$\alpha_{Z6}^{-1}(z) = \frac{2\pi}{\langle \Psi_{C_{6,\alpha}(z)}(\eta)|\widehat{r}|\Psi_{C_{6,\alpha}(z)}(\eta)\rangle}.$$

Proof. The vector is normalized because the two shell basis vectors are orthonormal:

$$\langle \Psi_C(\eta)|\Psi_C(\eta)\rangle = \frac{21+34C}{21+34C} = 1.$$

The observable $\widehat{r}$ is diagonal in the same basis, so the relative phase contributes no cross term. Hence the expectation is

$$\frac{21}{21+34C}\varphi^{-6} + \frac{34C}{21+34C}\varphi^{-7} = \frac{21\varphi^{-6} + 34C\varphi^{-7}}{21+34C}.$$

Substituting $C = C_{6,\alpha}(z)$ gives exactly the inverse-coupling formula in Definition 9346.53. □

Corollary 9346.81 (Born-like shell weights)**.** *The two-shell readout weights are squared amplitudes:*

$$P_6(C) = \frac{21}{21+34C}, \qquad P_7(C) = \frac{34C}{21+34C}.$$

Thus the effective radius is the Born-like diagonal projection

$$\langle \widehat{r}\rangle = P_6(C)\varphi^{-6} + P_7(C)\varphi^{-7},$$

and the scalar inverse coupling is the phase circumference divided by that projected radius.

Corollary 9346.82 (Diagonal radius phase blindness)**.** *The scalar $\alpha_{Z6}^{-1}(z)$ is insensitive to the relative shell phase η. An off-diagonal shell observable such as*

$$\widehat{X} = |6\rangle\langle 7| + |7\rangle\langle 6|$$

would instead read

$$\langle \Psi_C(\eta)|\widehat{X}|\Psi_C(\eta)\rangle = 2\sqrt{P_6(C)P_7(C)}\cos\eta.$$

Thus the inverse fine-structure scalar reads only the diagonal radius projection; a coherent shell phase would require an interference-type readout interface.

Definition 9346.83 (Cascade kernel as internal channel norm)**.** *Let*

$$\theta = \frac{\pi}{\varphi}.$$

On the internal channel carrier

$$\mathcal{H}_C = \operatorname{span}\{|b\rangle, |\chi\rangle, |t\rangle\},$$

define the amplitudes

$$a_b = \frac{1}{\sqrt{2}}, \qquad a_\chi = \frac{e^{i\theta} + e^{-i\theta}}{4} = \frac{1}{2}\cos\theta, \qquad a_t = \frac{1}{\sqrt{D_{6,\alpha}(z)\varphi^5}}.$$

Then the Window-six cascade kernel is the squared norm of the internal channel vector

$$|\psi_C(z)\rangle = a_b|b\rangle + a_\chi|\chi\rangle + a_t|t\rangle :$$

$$\| \, |\psi_C(z)\rangle \, \|^2 = \frac{1}{2} + \frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right) + \frac{1}{D_{6,\alpha}(z)\varphi^5} = C_{6,\alpha}(z).$$

Principle 9346.84 (Block-separated internal coherence)**.** *The cascade kernel is not a fully coherent sum over the binary, golden-phase, and tail channels. It is a block-separated norm ledger: the channel magnitudes add as*

$$C_{6,\alpha}(z) = |a_b|^2 + |a_\chi|^2 + |a_t|^2,$$

while the golden-phase channel itself contains the coherent pair

$$e^{i\pi/\varphi}, \qquad e^{-i\pi/\varphi}.$$

Thus the visible fine-structure readout factors through two Hilbertizable rows:

$$\mathcal{H}_C \longrightarrow C_{6,\alpha}(z) \longrightarrow \mathcal{H}_{\text{shell}} \longrightarrow \langle \widehat{r} \rangle \longrightarrow \alpha_{Z6}^{-1}(z).$$

The coefficient layer separation of Definition 9346.73 is the corresponding separation of readout effects rather than a competition among scalar fits.

Principle 9346.85 (Hilbert boundary for the Zeckendorf phase record)**.** *The phase-projection reading does not assert a complete quantum mechanics. The two-shell construction above is a Hilbertizable compression of the scalar formula, not a complete carrier for the raw fold fibers, unitary time evolution, the Born rule as a physical postulate, or a measurement theory. A genuine Hilbert lift of the full Zeckendorf record would need a carrier such as*

$$\mathcal{H}_6 = \ell^2(\Omega_6) \qquad \textit{or} \qquad \mathcal{H}_6 = \ell^2(X_6),$$

a state

$$|\Psi_6\rangle = \sum_{\omega \in \Omega_6} a_\omega |\omega\rangle, \qquad a_\omega \in \mathbb{C},$$

fiber projectors

$$P_x|\Psi_6\rangle = \sum_{\mathsf{Fold}_6(\omega)=x} a_\omega |\omega\rangle,$$

phase transport

$$a_\omega \longmapsto a_\omega e^{i\theta(\omega)},$$

and an observable $\widehat{R}_6$ whose expectation value is compared with the scalar readout:

$$\langle \Psi_6 | \widehat{R}_6 | \Psi_6 \rangle \approx \alpha_{Z6}^{-1}.$$

Until such rows are supplied, the correct status is a Born-like pre-quantum phase-projection geometry rather than a complete quantum superposition theorem.

Corollary 9346.86 (Interference-shadow status of the cosine term)**.** *The term*

$$\frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right)$$

in the cascade kernel is the squared intensity of the internal golden-phase channel:

$$\left|\frac{e^{i\pi/\varphi}+e^{-i\pi/\varphi}}{4}\right|^2=\frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right).$$

It is therefore an explicit interference-shaped row inside the block-separated cascade norm, while the full manuscript still stops short of a complete quantum-mechanical measurement theory.

Theorem 9346.87 (Static denominator closed; Green-excess remains separate)**.** *The low-energy static denominator is closed at the static cut-flux layer:*

$$D^{\star}_{6,\alpha}=47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{8}{9}\varphi^{-27}.$$

The first candidate Green-running coefficient is the centered Green-excess kernel

$$q_{3,\mathrm{run}}(z)=\Delta_R(z),\qquad \Delta_R(1)=\frac{26401}{2^{13}\cdot 571}.$$

Thus the subsequent certificate problem is not whether to add another static term to $D^{\star}_{6,\alpha}$, but whether a physical Green-excess activation parameter $z(Q)$ and higher-window data replace the fixed Window-six seed by a refinement flow

$$X_6\rightsquigarrow X_{m(Q)}.$$

Proof. The static part is the conclusion of Definition 9346.51: every static Dirichlet cut-source is Laplacian-exact, and the right-boundary source therefore gives $q_3=0$ for the infrared readout. The dynamic coefficient is the Green–Kubo response after subtracting the static variance baseline of the centered right-boundary observable. Since $\Delta_R(0)=0$, the Green-excess seed contributes no correction at the static point. For $0<z\le 1$ the displayed denominator is negative while the polynomial $3z(1595z^2+24477z+26730)$ is positive; the leading minus sign therefore makes $\Delta_R(z)>0$. Because the readout $R(D)$ is decreasing in D, the Green-excess term lowers α^{-1} and raises the coupling. At nonzero scale the effect is controlled by Green-excess activation and window-refinement data, not by a change to the static denominator. □

Concretized in §8769: Static Green boundary flux is realized as finite reduced-Laplacian, cut-source, exactness, localized-denominator, transport, replay, provenance, and NameCert rows.

Theorem 9346.88 (Independent denominator is required for predictivity)**.** *Let*

$$R(D):=2\pi\varphi^5\frac{F_8+F_9C(D)}{F_8\varphi^{-1}+F_9C(D)\varphi^{-2}},$$

where

$$C(D):=\frac{1}{2}+\frac{1}{4}\cos^2\left(\frac{\pi}{\varphi}\right)+\frac{1}{D\varphi^5}.$$

For positive D, the readout $R(D)$ is strictly decreasing in D. Consequently, if D is obtained by inverting an observed low-energy value of $\alpha_{\mathrm{EM}}^{-1}$, then the expression has no predictive force as a Window-six claim. A predictive use requires a denominator rule

$$D_6=\Gamma(X_6,\mathsf{Fold}_6,\textit{boundary parity},\textit{tail ledger})$$

that does not use the measured electromagnetic coupling.

Proof. The derivative with respect to C is

$$\frac{\partial R}{\partial C} = 2\pi\varphi^5 \frac{F_8 F_9(\varphi^{-1} - \varphi^{-2})}{(F_8\varphi^{-1} + F_9 C\varphi^{-2})^2}.$$

Since $\varphi^{-1} > \varphi^{-2}$, this derivative is positive. Also

$$\frac{dC}{dD} = -\frac{1}{D^2\varphi^5} < 0.$$

Thus $dR/dD < 0$. In any interval where the inverse problem is posed, the measured scalar can therefore determine a denominator by inversion rather than by Window-six structure. Such an inverted denominator is a fit, not an independent prediction. Predictivity requires that the denominator be supplied by the Window-six ledger before comparison with the physical scalar. □

Definition 9346.89 (Low-energy alpha-projection gate)**.** *A low-energy alpha-projection gate is protocol data Γ^α for the electromagnetic representation certificate that selects an admissible cross-scale boundary-return branch before the Window-six residual ledger is evaluated. At the first post-Window-six boundary it has the form*

$$\Gamma_7^\alpha : \{0, F_{10}, F_{11}\} \longrightarrow \{0, F_{10}\}.$$

Equivalently, it retains the $F_{10} = 55$ return branch from $\{0, 55, 89\}$ and excludes the $F_{11} = 89$ diffusion branch from the low-energy residual ledger. This gate is a separate protocol field; it is not inferred from the measured electromagnetic scalar.

Principle 9346.90 (Square-residue gate candidate)**.** *A natural candidate implementation of Definition 9346.89 is the square-residue projection at $p_* = 571$. The candidate is guided by the order data*

$$\mathrm{ord}_{571}(34) = 285, \qquad \mathrm{ord}_{571}(55) = 19, \qquad \mathrm{ord}_{571}(89) = 570, \qquad \mathrm{ord}_{571}(144) = 285.$$

Under this reading the low-energy projection keeps the re-lockable residue branch and excludes the primitive diffusion branch. This remains protocol-gate data until a NameCert supplies the corresponding internal certificate.

Principle 9346.91 (Protocol-gated Window-six minimal residual closure)**.** *The protocol-gated Window-six minimal closure route treats the coarse denominator*

$$D_0 = F_{10} - F_6 = F_9 + F_7 = 47$$

as fixed by the outer uplift tail, the eightfold boundary parity, the next boundary, and the deep tail. In the coarse ledger, F_9 is the boundary alias already visible in the projection weights, while $F_7 = N_{\geq 3}(6)$ is the non-displayed deep tail. The first residual seam is therefore placed at the golden shell of this deep-tail address

$$\varphi^{-(F_5+F_3)} = \varphi^{-7},$$

not at a shallower term that would rewrite the visible F_8 layer or the explicit F_9 boundary layer. The boundary holonomy quotient is the two-bit residual quotient

$$C_\partial / P_6^{\mathrm{geo}} \cong (\mathbb{Z}/2\mathbb{Z})^2.$$

The boundary sheet expectation is the two-sheet average

$$E_{\text{sheet}}(a) = \frac{1}{2}(a + P_w a P_w),$$

with index 2*, and the binary boundary uplift satisfies*

$$(\mathsf{Fold}_6^{\text{bin}})^{-1}(w) = \{V_6(w), V_6(w) + 34\},$$

so the sheet difference is $34 = F_9$*. The nonzero Window-six uplift tail offsets obey*

$$\Delta_{\text{tail}}(6) \subset \{F_8, F_9, F_{10}\} = \{21, 34, 55\}.$$

The Window-six boundary difference itself is only two-sheeted,

$$\Delta_6^{\text{bdry}} = \{0, F_9\},$$

whereas the next boundary differences have three branches,

$$\Delta_7^{\text{bdry}} = \{0, F_{10}, F_{11}\}, \qquad \Delta_8^{\text{bdry}} = \{0, F_{11}, F_{12}\}.$$

Therefore the Window-six sheet parity is not transported across scales without protocol data. Given the low-energy gate Γ_7^{α} *of Definition 9346.89, the first cross-scale return branch is* F_{10}*. The closure mode then determines whether this return is held fixed at the Window-six anchor or walked to the next retained branch. The anchor-local step is*

$$u_6 := \varphi^{-(F_6+F_3)} = \varphi^{-10}.$$

The walking-gate mode uses the next retained branch F_{12}*, hence its third displayed exponent is* $7 + 10 + 12 = 29$*. The primitive* F_{11} *branch is not part of the low-energy ledger selected by the gate. The Window-six closure displays three residual terms because the fiber multiplicities are only* $2, 3, 4$*, equivalently because the tail ledger stops at* $N_{\geq 4}(6)$ *inside Window six. Further terms require cross-window protocol refinement. Its first two coefficient normalizations are*

$$q_0 = 1, \qquad q_1 = -\frac{F_2}{F_3}.$$

Here $q_0 = 1$ *reads the single occurrence of the* F_7 *deep-tail seam,* $q_1 = -1/2$ *reads the two-sheet overlap subtraction. The same coefficient is also the alternating phase eigenvalue in the untwisted fold-gauge mismatch spectrum*

$$\operatorname{spec}(A_0) = \{1, \tfrac{1}{2}, -\tfrac{1}{2}\},$$

where the $-1/2$ *mode is the Jordan alternating return seen by the statistical projection. The third coefficient is fixed only after a readout mode is chosen. The full phase-reservoir readout sees*

$$C_\partial \cong (\mathbb{Z}/2\mathbb{Z})^3, \qquad q_2^{\text{res}} = \frac{|C_\partial|}{N_{\geq 4}(6)} = \frac{8}{9},$$

also matching the ratio between the fiber histogram layers $\#\{d_6 = 2\} = 8$ *and* $\#\{d_6 = 4\} = 9$*, and matching the right-block escape flux in Definition 9346.49. The* $U(1)$*-image readout uses the six-element phase image supplied by the* Z_6 *interface*

$$C_\partial \times T \to \mathbb{Z}/6\mathbb{Z} \to U(1), \qquad q_2^{\text{img}} = \frac{6}{N_{\geq 4}(6)} = \frac{2}{3}.$$

The irreducible curvature-residual readout quotients the locally absorbable diagonal subgroup

$$P_6^{\mathrm{geo}} \cong \mathbb{Z}/2\mathbb{Z}, \qquad C_\partial/P_6^{\mathrm{geo}} \cong (\mathbb{Z}/2\mathbb{Z})^2,$$

and hence has

$$q_2^{\mathrm{curv}} = \frac{|C_\partial/P_6^{\mathrm{geo}}|}{N_{\geq 4}(6)} = \frac{4}{9}.$$

Theorem 9346.92 (Protocol-gated Window-six closure determines a readout family)**.** *Assume the protocol-gated Window-six minimal residual closure of Definition 9346.91. The three-term residual polynomial selected by the Window-six fiber-depth cutoff, the low-energy gate, a closure mode, and a readout mode*

$$Q_6(u) = q_0 + q_1 u + q_2 u^2$$

therefore has coefficients

$$q_0 = 1, \qquad q_1 = -\frac{F_2}{F_3}, \qquad q_2 \in \left\{\frac{8}{9}, \frac{2}{3}, \frac{4}{9}\right\},$$

where $q_2 = 8/9$ is the phase-reservoir readout, $q_2 = 2/3$ is the $Z_6 \to U(1)$ image readout, and $q_2 = 4/9$ is the irreducible curvature-residual readout. In anchor-local mode the resulting three-term Window-six residual denominator is

$$D_{6,\mathrm{loc}}^{(q_2)} = F_9 + F_7 + \varphi^{-(F_5+F_3)}\left[1 - \frac{F_2}{F_3}\varphi^{-(F_6+F_3)} + q_2\varphi^{-2(F_6+F_3)}\right].$$

Equivalently, the three anchor-local readouts are

$$D_{6,\mathrm{loc}}^{\mathrm{res}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{8}{9}\varphi^{-27}, \qquad D_{6,\mathrm{loc}}^{\mathrm{img}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{2}{3}\varphi^{-27}, \qquad D_{6,\mathrm{loc}}^{\mathrm{curv}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{4}{9}\varphi^{-27}.$$

In walking-gate mode the three readouts are

$$D_{6,\mathrm{walk}}^{\mathrm{res}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{8}{9}\varphi^{-29}, \qquad D_{6,\mathrm{walk}}^{\mathrm{img}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{2}{3}\varphi^{-29}, \qquad D_{6,\mathrm{walk}}^{\mathrm{curv}} = 47+\varphi^{-7}-\frac{1}{2}\varphi^{-17}+\frac{4}{9}\varphi^{-2}$$

Proof. By the protocol-gated Window-six minimal closure, the anchor-local denominator has the form

$$D_{6,\mathrm{loc}}^{(q_2)} = D_0 + \varphi^{-(F_5+F_3)}Q_6(\varphi^{-(F_6+F_3)}).$$

Taking the specified three-term polynomial and substituting the displayed coefficients gives the first formula. The Fibonacci values

$$F_2 = 1, \quad F_3 = 2, \quad F_5 = 5, \quad F_6 = 8, \quad F_7 = 13, \quad F_9 = 34$$

then give $F_9 + F_7 = 47$, $F_5 + F_3 = 7$, $F_6 + F_3 = 10$, and $F_2/F_3 = 1/2$. Substituting $q_2 = 8/9$, $q_2 = 2/3$, and $q_2 = 4/9$ gives the three anchor-local readouts. For walking-gate mode, the first two exponents remain 7 and 17, while the third exponent walks from the F_{10} return to the next retained F_{12} branch. Thus the third exponent is $7 + 10 + 12 = 29$. Substituting the three allowed readout coefficients gives the displayed walking-gate readouts. □

Theorem 9346.93 (Fixed electromagnetic readout selects the local reservoir branch)**.** *Assume the readout is the fixed Window-six projection formula whose visible state and boundary rows are F_8 and F_9. Assume also that the physical target is the low-energy electromagnetic impedance readout whose phase observations are gated by the $Z_6 \to U(1)$ image. Then the admissible low-energy denominator selected from Definition 9346.92 is*

$$D_{6,\alpha}^{\star} = D_{6,\mathrm{loc}}^{\mathrm{res}} = 47 + \varphi^{-7} - \frac{1}{2}\varphi^{-17} + \frac{8}{9}\varphi^{-27}.$$

Proof. The fixed Window-six projection formula keeps the visible pair (F_8, F_9). A walking residual would move the residual ledger to the next retained branch F_{12} while the visible readout remains fixed at (F_8, F_9). Thus the fixed low-energy formula selects the anchor-local closure. The $Z_6 \to U(1)$ image supplies the admissible phase observation gate, but $\alpha_{\mathrm{EM}}^{-1}(0)$ is an impedance readout of the phase reservoir behind that gate rather than a bare image-cardinality readout. By Definition 9346.49, the same reservoir coefficient is also the normalized right-block escape flux $P(\text{leave } U_R \mid U_R) = 8/9$. Thus the selected coefficient is the flux-reservoir coefficient $q_2 = 8/9$, not the image coefficient $q_2 = 2/3$ or the curvature quotient $q_2 = 4/9$. Substitution into the readout-family theorem gives the displayed denominator. □

Principle 9346.94 (Remaining formalization obligations)**.** *The protocol-gated Window-six closure becomes a fully internal Window-six theorem only after six further obligations are discharged. First, the ProtocolGateLegality obligation must certify*

$$\Gamma_7^{\alpha} : \{0, F_{10}, F_{11}\} \to \{0, F_{10}\}$$

as an allowed low-energy projection gate. If the square-residue gate is used, this obligation must certify the $p_* = 571$ residue-stratum separation in Definition 9346.90. Second, the GoldenResiduAlLadder obligation must derive

$$D_6 - D_0 \in \varphi^{-7}\mathbb{Q}_Z[\varphi^{-10}],$$

from the deep-tail seam and the gated F_{10} uplift-return rule. Third, the TwoSheetMobius obligation must derive

$$q_1 = \frac{1-2}{2} = -\frac{1}{2}.$$

It must also identify the same coefficient with the Jordan alternating phase mode of the fold-gauge mismatch spectrum. Fourth, the ReadoutMode obligation must distinguish the phase reservoir C_∂, giving $q_2 = 8/9$, the $Z_6 \to U(1)$ image, giving $q_2 = 2/3$, and the irreducible curvature-residual quotient $C_\partial/P_6^{\mathrm{geo}}$, giving $q_2 = 4/9$; for $\alpha_{\mathrm{EM}}^{-1}(0)$ it must select the phase-reservoir impedance readout gated by Z_6, using the local conditional escape flux $8/9$ rather than the global cut-flux density $1/12$. Fifth, the ClosureMode obligation must certify that fixed $R_6(D)$ readouts select anchor-local closure, while walking-gate closure belongs to a scale-indexed $R_m(D_m)$ refinement object. Sixth, the GreenResolved obligation separates the static right-boundary cut-flux source from centered Green-relaxation input. The static low-energy right-boundary readout has $q_3 = 0$ by Definition 9346.51; the right-boundary cut vector is an instance of the general Dirichlet cut-source identity $L_B\mathbf{1}_I = b_B$ and therefore has zero 571-Green class. By the same theorem, centered relaxation observables for the coarse Markov kernel have susceptibility and Green–Kubo denominators containing 571. A nonzero Green term may therefore enter only in a running or dynamic response certificate, such as

$$D_{6,\alpha}(z) = D_{6,\alpha}^{\star} + \Delta_R(z)\varphi^{-37} + O(\varphi^{-47}), \qquad \Delta_R(0) = 0.$$

The Window-six seed candidate is Definition 9346.52, with $\Delta_R(1) = 26401/(2^{13} \cdot 571)$. The GreenClass branch must therefore show that any physical running readout uses this centered Green-excess source, or else replace it by a certified higher-window Green mode. The FluxGreenCoupling branch must decide separately whether the global cut-flux density $\Phi_R = 1/12$ participates in the activation of the Green-excess parameter. Until these obligations are supplied, the closed denominator is a protocol-gated Window-six ledger conclusion rather than a fully certified internal theorem.

Theorem 9346.95 (Physical refutation surface for the Window-six readout)**.** *The physical interpretation of a selected $R_{Z6,\mu}^{\star}$ is refutable at the derived-interface level by any of the following*

failures: no admissible low-energy projection gate selects the F_{10} branch; the physical representation cannot decide the anchor-local versus walking-gate closure mode at the required precision; the physical representation cannot decide whether the target reads the phase reservoir, the $Z_6 \to U(1)$ image, or the irreducible curvature-residual quotient; the selected third coefficient remains a fitted rational rather than a readout-mode normalization; independent low-energy routes converge to incompatible values after systematic corrections; a high-energy comparison reuses $R^{\star}_{Z6,\mu}$ without a scale-indexed ledger or spectral-response activation map and window-refinement flow; or later residual corrections inside the static cut-flux layer require terms outside the selected gated ladder

$$\{7, 17, 27\} \quad \textit{in anchor-local mode,} \qquad \{7, 17, 29\} \quad \textit{in walking-gate mode.}$$

Proof. The first failure violates the protocol-gate requirement in Definition 9346.89. The second leaves the denominator family mode-underdetermined at the physical interface. The third leaves the coefficient family readout-underdetermined. The fourth violates the residual closure principle in Definition 9346.9, the protocol-gated closure principle in Definition 9346.91, and the readout-mode coefficient normalization in Definition 9346.94. The fifth violates the route-convergence requirement of Definition 9346.43. The sixth violates the infrared boundary of Definition 9346.44. The seventh violates the displayed Window-six residual denominators, whose correction terms occur at $7, 17, 27$ in anchor-local mode and at $7, 17, 29$ in walking-gate mode. Each failure breaks an explicit interface condition without needing to deny the finite Zeckendorf fold construction itself. □

Concretized in §8811: The phase-shell chapter realizes the stable-window, residual-fiber, and external golden-phase projection rows as finite NameCert data.

Chapter 9347

Window-Six Inverse-Limit Lifting

chapter 9346 reads the no-adjacent-11 grammar as a finite visible surface, with Window six as the distinguished twenty-one-state readout. The present chapter records the projection-tower form behind that surface. It separates the address carrier from the flux-operator geometry: the address carrier lifts to the Zeckendorf golden-mean inverse limit, while a flux or Jordan-return reading requires additional compatibility rows for the operator tower.

9347.1 Address tower

Definition 9347.1 (No-adjacent address windows)**.** *For each $m \geq 0$, let*

$$X_m := \{w \in \{0,1\}^m : \ w_i w_{i+1} \neq 11 \text{ for every admissible } i\}.$$

For $m \geq 0$, the prefix projection is

$$\pi_{m+1,m} : X_{m+1} \to X_m, \qquad \pi_{m+1,m}(w_1, \ldots, w_{m+1}) = (w_1, \ldots, w_m).$$

For $m \geq n$, write $\pi_{m,n} : X_m \to X_n$ for the iterated prefix projection. Prefix stability of the no-adjacent rule gives

$$\pi_{n,k} \circ \pi_{m,n} = \pi_{m,k} \qquad (m \geq n \geq k),$$

so the family $(X_m, \pi_{m,n})$ is an inverse system of finite windows.

Definition 9347.2 (Zeckendorf address inverse limit)**.** *The Zeckendorf address inverse limit is*

$$X_\infty^Z := \varprojlim_m X_m.$$

Equivalently, it is the carrier of infinite no-adjacent-11 address streams

$$X_\infty^Z \cong \{(c_1, c_2, \ldots) \in \{0,1\}^{\mathbb{N}} : \ c_i c_{i+1} \neq 11 \text{ for every } i\}.$$

The projection to level m is written

$$P_m : X_\infty^Z \to X_m.$$

Proposition 9347.3 (Window six is the sixth address projection)**.** *The Window-six address surface is exactly the sixth projection of the Zeckendorf address inverse limit:*

$$\mathsf{Window6}_{\mathrm{addr}} = X_6 = P_6(X_\infty^Z).$$

Moreover,

$$|X_6| = F_8 = 21.$$

Proof. Every infinite no-adjacent-11 stream has a length-six prefix in X_6, so $P_6(X_\infty^Z) \subseteq X_6$. Conversely, every word $w \in X_6$ extends to an infinite stream by appending zeros, hence $w \in P_6(X_\infty^Z)$. The counting formula for no-adjacent binary words gives $|X_m| = F_{m+2}$, so $|X_6| = F_8 = 21$. □

Definition 9347.4 (Window cylinder)**.** *For a visible word $w \in X_6$, its inverse-limit cylinder is*

$$\mathsf{Lift}_6(w) := P_6^{-1}(w) = \{x \in X_\infty^Z : P_6(x) = w\}.$$

Definition 9347.5 (Ambient Window-six cylinder)**.** *Let*

$$C_6 := \{0,1\}^6$$

be the full six-bit carrier, and let

$$Q_6 : X_\infty^Z \to C_6$$

be the first-six-coordinate readout. For $u \in C_6$, its ambient Window-six cylinder is

$$\mathrm{Cyl}_6(u) := \{x \in X_\infty^Z : Q_6(x) = u\}.$$

When $u \in X_6$, this agrees with the inverse-limit cylinder $\mathsf{Lift}_6(u)$.

Proposition 9347.6 (Window-six extension boundary)**.** *For $u \in C_6$, the cylinder $\mathrm{Cyl}_6(u)$ is nonempty if and only if $u \in X_6$.*

Proof. If $u \in X_6$, append zeros to obtain

$$x = u000\cdots.$$

Appending zeros cannot create an adjacent pair 11, so $x \in X_\infty^Z$ and $Q_6(x) = u$. Hence $\mathrm{Cyl}_6(u)$ is nonempty.

Conversely, if $u \notin X_6$, then u contains an adjacent pair 11. Every infinite binary sequence with prefix u contains that same adjacent pair, so no such sequence lies in X_∞^Z. Hence $\mathrm{Cyl}_6(u) = \varnothing$. □

Corollary 9347.7 (Window-six cylinder partition)**.** *The sixth projection decomposes the inverse-limit address carrier as*

$$X_\infty^Z = \coprod_{u \in X_6} \mathrm{Cyl}_6(u).$$

The twenty-one words of X_6 are the nonempty sixth-layer cylinder gates, while the forty-three words of $C_6 \setminus X_6$ have no inverse-limit extension.

Proof. Every $x \in X_\infty^Z$ has a unique sixth prefix $Q_6(x) \in X_6$, so it belongs to exactly one displayed cylinder. The converse inclusion is immediate from the definition of $\mathrm{Cyl}_6(u)$. The counts are $|X_6| = 21$ by Definition 9347.3 and $|C_6 \setminus X_6| = 64 - 21 = 43$. □

Principle 9347.8 (Finite window is not a unique address stream)**.** *A Window-six state determines a cylinder in X_∞^Z, not a unique inverse-limit address. A unique stream read requires tail data such as a stream name, a projection-compatibility ledger, or a normalization route. Without such a row, X_6 is only a finite observation surface.*

Principle 9347.9 (Finite obstruction as inverse-limit refusal)**.** *The Window-six split*

$$C_6 = X_6 \sqcup (C_6 \setminus X_6)$$

is not merely a finite count. It is an extension test against the inverse-limit carrier: admissible sixth windows have nonempty cylinders, and forbidden sixth windows have empty cylinders. Thus a local forbidden pattern cannot be hidden by an infinite tail.

The RH-facing analogue is the carrier-level obligation that a finite prime-window radial defect cannot extend to a completed zeta zero ledger. In the notation of Definition 9246.13, this is the role of

$$\mathsf{DefectZeroIncompatibility}.$$

Window six supplies the finite inverse-limit prototype, not the zeta incompatibility theorem itself.

9347.2 Finite support and tower continuation

Definition 9347.10 (Finite-support Zeckendorf layer)**.** *The finite-support Zeckendorf layer is*

$$X^Z_{\mathrm{fin}} := \{c \in X^Z_\infty : \ c_i = 0 \text{ for all sufficiently large } i\}.$$

It carries the usual natural-number Zeckendorf read

$$n = \sum_i c_i F_i.$$

Proposition 9347.11 (Address inverse limit is larger than number notation)**.** *The finite-support layer X^Z_{fin} is a subcarrier of X^Z_∞, but it does not exhaust X^Z_∞.*

Proof. Every finite-support no-adjacent stream is an infinite no-adjacent stream, so $X^Z_{\mathrm{fin}} \subseteq X^Z_\infty$. The periodic stream $101010\cdots$ is no-adjacent but not eventually zero, hence it lies in X^Z_∞ outside X^Z_{fin}. □

Definition 9347.12 (Projection-tower continuation)**.** *A projection-tower continuation from Window six is a compatible family*

$$(x_m)_{m\geq 6}, \qquad x_m \in X_m,$$

such that

$$\pi_{m+1,m}(x_{m+1}) = x_m \qquad \text{for every } m \geq 6.$$

Equivalently, it is an inverse-limit lift $x_\infty \in X^Z_\infty$ together with the displayed row $P_6(x_\infty) = x_6$.

Principle 9347.13 (Continuation is not analytic continuation)**.** *Projection-tower continuation is not analytic continuation in the complex function-theoretic sense. It is inverse-limit lifting of compatible finite address windows:*

$$X_6 \leftarrow X_7 \leftarrow X_8 \leftarrow \cdots \leftarrow X^Z_\infty.$$

Thus Window six is not the Zeckendorf address object itself; it is the sixth projection surface of that object.

9347.3 Fixed projection shadow

Definition 9347.14 (Golden-mean self-similar object)**.** *The Zeckendorf address inverse limit carries the self-similar decomposition*

$$X_\infty^Z \cong 0X_\infty^Z \sqcup 10X_\infty^Z.$$

*Every infinite no-adjacent-*11 *stream either begins with* 0 *followed by an arbitrary no-adjacent stream, or begins with* 10 *followed by an arbitrary no-adjacent stream. The corresponding growth scale* λ *satisfies*

$$\lambda^2 = \lambda + 1,$$

whose positive solution is the golden scale φ*.*

Definition 9347.15 (Golden-mean shift kernel)**.** *The finite-type transition kernel of* X_∞^Z *is the two-state matrix*

$$K_Z := \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix},$$

where the entry $K_Z(a, b)$ *is admissible exactly when the adjacent pair* ab *is not* 11*. The left shift*

$$\sigma : X_\infty^Z \to X_\infty^Z, \qquad \sigma(c_1, c_2, c_3, \ldots) = (c_2, c_3, \ldots)$$

therefore makes X_∞^Z *the one-sided golden-mean subshift.*

Proposition 9347.16 (Golden growth of finite windows)**.** *The finite windows satisfy*

$$|X_m| = F_{m+2},$$

and the transition kernel of Definition 9347.15 has characteristic equation

$$\lambda^2 = \lambda + 1.$$

Thus the growth scale of the inverse-limit carrier is the golden scale φ*.*

Proof. The no-adjacent grammar splits each nonempty word by its first symbols: words beginning with 0 contribute X_{m-1}, while words beginning with 1 must begin with 10 and contribute X_{m-2}. Hence

$$|X_m| = |X_{m-1}| + |X_{m-2}|.$$

With the initial values $|X_0| = 1$ and $|X_1| = 2$, this gives $|X_m| = F_{m+2}$. The displayed matrix has characteristic polynomial $\lambda^2 - \lambda - 1$, whose positive root is φ. □

Concretized in §8802: The window-six golden-mean shift is realized as finite no-adjacent-one BHist window, transition, shift, growth, and NameCert rows.

Principle 9347.17 (Window six is not the fixed object)**.** *The fixed object is* X_∞^Z *with the self-similar decomposition of Definition 9347.14. Window six is not that inverse-limit fixed object. It is the finite projection*

$$X_6 = P_6(X_\infty^Z)$$

and therefore records only the six visible address coordinates.

Definition 9347.18 (Round-trip fixed projection)**.** *A choice of lift on X_6 is a map*

$$L_6 : X_6 \to X_\infty^Z$$

such that

$$P_6(L_6(w)) = w \qquad \text{for every } w \in X_6.$$

Equivalently,

$$P_6 \circ L_6 = \mathrm{id}_{X_6}.$$

Proposition 9347.19 (Window six fixes visible information under lift-return)**.** *For any displayed lift L_6, Window six is a fixed information surface for the lift-then-project composite:*

$$P_6(L_6(w)) = w.$$

This fixedness is a statement about visible information, not uniqueness of the hidden inverse-limit tail.

Proof. The equality is exactly the defining row of Definition 9347.18. The hidden tail is not unique because Definition 9347.4 gives the whole cylinder $P_6^{-1}(w)$ above a visible word w. □

Definition 9347.20 (Window-six fixed-shadow defect)**.** *The Window-six fixed-shadow defect is the finite boundary response exposed when the self-similar inverse-limit object is read only through the sixth projection. In the notation of Definition 9346.10, its displayed response part is*

$$\mathsf{Defect}_6 := \varphi^{-7}\left(1 - \frac{1}{2}\varphi^{-10} + \frac{8}{9}\varphi^{-20}\right).$$

Thus

$$D_6^\star = 47 + \mathsf{Defect}_6.$$

Principle 9347.21 (Finite shadow, not complete closure)**.** *The terms in Defect_6 are read as boundary-seam, return-echo, and flux-escape response rows of the finite projection. They measure the failure of the sixth projection to be the complete inverse-limit object; they do not turn X_6 into the self-similar fixed object itself.*

9347.4 Projection-shadow comparison with recovered zeta

Definition 9347.22 (Projection-shadow carrier pattern)**.** *A projection-shadow carrier pattern consists of a recovered or inverse-limit carrier C, a visible projection $P : C \to S$, and a displayed lift row $L : S \to C$ satisfying*

$$P \circ L = \mathrm{id}_S.$$

This row says that the visible information is fixed by lift-then-project. It does not say that the visible carrier S is the whole carrier C.

Proposition 9347.23 (Window six and the right-half-plane zeta projection)**.** *Window six and the right-half-plane zeta projection share the projection-shadow carrier pattern:*

$$X_6 = P_6(X_\infty^Z),$$
$$\zeta_{>1} = P_{>1}(\mathsf{ZETA}).$$

Here $P_{>1}$ denotes the projection of the recovered zeta carrier to the Dirichlet-series and Euler-product rows on $\Re(s) > 1$, as displayed in Definition 9243.2 and Definition 9243.1. The analogy is a carrier syntax analogy, not an identification of Window six with ζ.

Proof. The first equality is Definition 9347.3. The second equality is the projection reading of the zeta recovery packet: Definition 9243.2 reconstructs ZETA from its compatible local rows, and Definition 9243.1 displays the Dirichlet and Euler rows on $\Re(s) > 1$. In both cases a local slice is recovered from a richer carrier and remains stable under the displayed projection row. □

Principle 9347.24 (Carrier fixedness is not orbit collapse)**.** *The completed-zeta row of Definition 9243.17 gives a mirror-fixed carrier:*

$$\xi(s) = \xi(1-s).$$

That carrier fixedness supports zero-set mirror transport in the sense of Definition 9241.13, but it is weaker than the pointwise fixedness of zero events:

$$\mathsf{ZeroEventFlow}_{\mathsf{RH}}(s) \to s = J(s).$$

By Definition 9241.12, the latter row is equivalent to $\Re(s) = 1/2$, and by Definition 9241.14 it is not supplied by mirror closure alone. Thus the zeta side mirrors the Window-six warning: a fixed carrier or a fixed projection shadow is not yet a proof that every residual event has collapsed to the fixed surface.

9347.5 Operator compatibility

Definition 9347.25 (Compatible operator tower)**.** *Let $\mathcal{R}_m$ be a residual module whose coordinate rows are indexed by X_m. A compatible operator tower over the Zeckendorf address tower is a family of operators*

$$A_m : \mathcal{R}_m \to \mathcal{R}_m \qquad (m \geq 6)$$

together with residual projections

$$\rho_{m+1,m} : \mathcal{R}_{m+1} \to \mathcal{R}_m$$

such that

$$\rho_{m+1,m} \circ A_{m+1} = A_m \circ \rho_{m+1,m} \qquad \text{for every } m \geq 6.$$

Principle 9347.26 (Address lifting does not lift flux dynamics)**.** *The identity*

$$\mathsf{Window6}_{\mathrm{addr}} = P_6(X_\infty^Z)$$

lifts the address carrier only. It does not lift the Window-six flux-operator geometry of Definition 9346.48. A flux, edge-operator, phase-gate, or Jordan-return reading reaches the inverse limit only through a compatible operator tower in the sense of Definition 9347.25.

Definition 9347.27 (Window-six separating certificate)**.** *A Window-six separating certificate for a class of residual towers is a row stating that for every admitted residual tower $\widetilde{r}$ in the class,*

$$P_6^{\mathcal{R}}(\widetilde{r}) = 0 \Longrightarrow \widetilde{r} = 0.$$

The stronger all-level separation row is

$$\forall m \geq 6,\ P_m^{\mathcal{R}}(\widetilde{r}) = 0 \Longrightarrow \widetilde{r} = 0.$$

9347.6 RH residual bridge target

Definition 9347.28 (Counter residual bridge target)**.** *An RH counter residual bridge target is a displayed map*

$$\mathcal{B}_{\zeta\to Z} : \mathsf{CounterRH}(n,q) \to (\widetilde{r},\eta),$$

where $\widetilde{r}$ is a residual tower over X_∞^Z and η is a nonzero residual row such as

$$P_6^{\mathcal{R}}(\widetilde{r}) \neq 0.$$

This is a bridge target, not a closed bridge supplied by the Window-six address carrier.

Proposition 9347.29 (Window-six local residual contradiction)**.** *Let $r_6 \in \mathcal{R}_6$ be a residual packet. If*

$$r_6 \neq 0,$$

$$R_6(r_6) = -\frac{1}{2}r_6,$$

and

$$R_6(r_6) = r_6,$$

then contradiction follows.

Proof. The two return equations give

$$r_6 = -\frac{1}{2}r_6.$$

Thus

$$\frac{3}{2}r_6 = 0.$$

Over the rational residual module, $\frac{3}{2}$ is cancellable, so $r_6 = 0$. This contradicts the displayed row $r_6 \neq 0$. □

Theorem 9347.30 (Global residual contradiction criterion)**.** *Assume a residual tower $\widetilde{r}$ has projections $r_m = P_m^{\mathcal{R}}(\widetilde{r})$ for all $m \geq 6$. Suppose that for each $m \geq 6$ there is a scalar λ_m with $\lambda_m \neq 1$ and*

$$R_m(r_m) = \lambda_m r_m,$$

while the zero-event return-invariance row gives

$$R_m(r_m) = r_m.$$

If the all-level separation row of Definition 9347.27 is available, then no nonzero admitted residual tower satisfies these rows.

Proof. For each $m \geq 6$, the two return equations give

$$r_m = \lambda_m r_m.$$

Since $\lambda_m \neq 1$, scalar cancellation gives $r_m = 0$. Hence every projection of $\widetilde{r}$ is zero. The all-level separation row then gives $\widetilde{r} = 0$, contradicting nonzero admission of the residual tower. □

Remark 9347.31 (Missing certificates)**.** *The Window-six route becomes an RH refutation route only after three rows are supplied: a bridge from* $\mathsf{CounterRH}(n, q)$ *to a Zeckendorf skew-residual tower, a zero-event return-invariance row, and a separating certificate strong enough to turn vanishing projections into vanishing of the residual tower. Without those rows, Window six supplies a local residual geometry and a bridge target, not a proof of* RH.

Concretized in §278: Zeckendorf normal packets supply finite no-adjacent-word evidence for the address windows consumed by the projection tower.

Concretized in §544: Local carry packets supply the finite carry-facing rows that separate normal address readout from host equality.

Concretized in §554: The Axis-Zeckendorf cannot-claim packet supplies the refusal boundary preventing address lifting from being read as a complete operator or physical bridge.

Concretized in §1018: Boundary-token limit refusal packets supply the finite refusal row separating Window-six boundary data from unlicensed inverse-limit or RealUp closure claims.

Chapter 9348

Layer-Kernel Residual Induction

chapter 9347 separates the Window-six address carrier from the flux-operator tower. The present chapter records the induction form needed after that separation. A global RH counter residual is not attacked first at the completed inverse-limit object. It is inspected through the finite projection tower, where the first nonzero residual must appear in a newly added kernel layer.

9348.1 Linearized residual tower

Definition 9348.1 (Linearized address residual space)**.** *For each $m \geq 6$, let*

$$V_m := \mathbb{Q}[X_m]$$

be the rational vector space freely generated by the no-adjacent address window X_m of Definition 9347.1. The prefix map $\pi_{m+1,m} : X_{m+1} \to X_m$ induces the linear projection

$$\Pi_{m+1,m} : V_{m+1} \to V_m.$$

Definition 9348.2 (Layer kernel)**.** *The layer kernel added between levels m and $m+1$ is*

$$K_{m+1} := \ker(\Pi_{m+1,m}).$$

It records residual information visible at level $m+1$ but invisible after projection to level m.

Proposition 9348.3 (Layer-kernel dimension)**.** *Assume the prefix projection is surjective on the no-adjacent address windows. Then*

$$\dim K_{m+1} = |X_{m+1}| - |X_m| = F_{m+1}.$$

In particular,

$$\dim K_7 = F_7 = 13.$$

Proof. The dimension of V_m is $|X_m|$. Since every word in X_m extends by a zero digit to a word in X_{m+1}, the prefix projection is surjective. Therefore

$$\dim K_{m+1} = \dim V_{m+1} - \dim V_m.$$

Using $|X_m| = F_{m+2}$ gives

$$\dim K_{m+1} = F_{m+3} - F_{m+2} = F_{m+1}.$$

For $m = 6$, this yields $\dim K_7 = F_7 = 13$. □

Definition 9348.4 (Residual tower)**.** *A residual tower is a compatible family*

$$r_\infty = (r_m)_{m\geq 6}, \qquad r_m \in V_m,$$

such that

$$\Pi_{m+1,m}(r_{m+1}) = r_m \qquad \text{for every } m \geq 6.$$

The tower is zero when every projection r_m is zero.

Principle 9348.5 (Projection separation)**.** *Residual towers are separated by their finite projections:*

$$(\forall m \geq 6,\ r_m = 0) \Longrightarrow r_\infty = 0.$$

Equivalently, a nonzero residual tower has a nonzero finite projection at some level.

9348.2 First nonzero layer

Proposition 9348.6 (First nonzero residual lies in a layer kernel)**.** *Let $r_\infty = (r_m)_{m\geq 6}$ be a residual tower. If $r_\infty \neq 0$ and m_0 is the least level with $r_{m_0} \neq 0$, then either $m_0 = 6$ or*

$$r_{m_0} \in K_{m_0}.$$

Proof. If $m_0 = 6$, there is nothing to prove. If $m_0 > 6$, minimality gives $r_{m_0-1} = 0$. Compatibility of the residual tower gives

$$\Pi_{m_0,m_0-1}(r_{m_0}) = r_{m_0-1} = 0,$$

so $r_{m_0} \in \ker(\Pi_{m_0,m_0-1}) = K_{m_0}$. □

Principle 9348.7 (Induction target)**.** *The projection-tower problem is therefore reduced to the following finite layer question: can a newly added kernel K_m contain a nonzero residual that is simultaneously in the skew sector and fixed by the return operator? If every such kernel rejects fixed skew residuals, no nonzero admitted counter residual tower survives.*

9348.3 Return operators and spectral exclusion

Definition 9348.8 (Compatible return operators)**.** *A return-operator tower is a family of linear maps*

$$R_m : V_m \to V_m \qquad (m \geq 6)$$

such that

$$\Pi_{m+1,m} R_{m+1} = R_m \Pi_{m+1,m} \qquad \text{for every } m \geq 6.$$

Definition 9348.9 (Skew kernel sector)**.** *For each $m \geq 6$, a skew kernel sector is a displayed subspace*

$$K_m^{\text{skew}} \subseteq K_m$$

containing the kernel residuals admitted by the RH counter-residual bridge. The sector is part of the bridge data; it is not determined by the address carrier alone.

Definition 9348.10 (Layer spectral exclusion)**.** *Layer spectral exclusion at level m is the row*

$$1 \notin \operatorname{spec}(R_m|_{K_m^{\mathrm{skew}}}).$$

Equivalently, the linear map

$$I - R_m : K_m^{\mathrm{skew}} \to K_m^{\mathrm{skew}}$$

is invertible.

Proposition 9348.11 (No fixed skew residual in an excluded layer)**.** *If $r \in K_m^{\mathrm{skew}}$, $R_m r = r$, and layer spectral exclusion holds at m, then $r = 0$.*

Proof. The equation $R_m r = r$ is equivalent to $(I - R_m)r = 0$. Since $I - R_m$ is invertible on K_m^{skew}, its kernel is zero. Hence $r = 0$. □

Remark 9348.12 (Window-six anchor)**.** *Window six supplies the anchor pattern for this exclusion: on the intended skew sector, the return has the form*

$$R_6 \simeq -\frac{1}{2}I + N_6$$

with nilpotent tail N_6. Then

$$I - R_6 \simeq \frac{3}{2}I - N_6,$$

which is invertible over the rational residual module. The induction problem is to show that the same no-neutral-leak condition persists through the added kernels.

9348.4 Layer-kernel refutation

Theorem 9348.13 (Layer-kernel residual refutation)**.** *Assume the following rows for an admitted RH counter residual tower $r_\infty = (r_m)_{m \geq 6}$:*

$$r_\infty \neq 0,$$

zero-event return invariance

$$R_m r_m = r_m \qquad \text{for every } m \geq 6,$$

compatibility of the return operators, and, for every layer in which a first nonzero residual can appear, skew-sector admission

$$r_m \in K_m^{\mathrm{skew}}$$

together with layer spectral exclusion

$$1 \notin \operatorname{spec}(R_m|_{K_m^{\mathrm{skew}}}).$$

Then no such nonzero admitted RH counter residual tower exists.

Proof. By projection separation, choose the least $m_0 \geq 6$ with $r_{m_0} \neq 0$. If $m_0 = 6$, the Window-six anchor places the residual in the admitted skew sector, and spectral exclusion at level 6 contradicts $R_6 r_6 = r_6$ by Definition 9348.11.

If $m_0 > 6$, then Definition 9348.6 gives $r_{m_0} \in K_{m_0}$. The bridge-admission row places this residual in $K_{m_0}^{\mathrm{skew}}$. Zero-event return invariance gives $R_{m_0} r_{m_0} = r_{m_0}$, while layer spectral exclusion at m_0 forces $r_{m_0} = 0$ by Definition 9348.11. This contradicts the choice of m_0. □

Corollary 9348.14 (Projection induction form)**.** *Equivalently, define*

$$P(m): \quad r_m = 0.$$

If the Window-six anchor proves $P(6)$ and every induction step proves that $r_m = 0$ forces $r_{m+1} \in K_{m+1}^{\mathrm{skew}}$ and hence $r_{m+1} = 0$ by spectral exclusion, then

$$\forall m \geq 6,\ r_m = 0.$$

Projection separation then gives $r_\infty = 0$.

Proof. The induction step uses compatibility:

$$\Pi_{m+1,m}(r_{m+1}) = r_m = 0,$$

so $r_{m+1} \in K_{m+1}$. The bridge-admission row places it in the skew kernel sector, and spectral exclusion removes every fixed residual there. Ordinary induction gives all finite projections zero, and projection separation gives the zero tower. □

9348.5 Proof obligations

Definition 9348.15 (Projection-tower proof obligations)**.** *The layer-kernel induction route has four proof obligations:*

$$\mathcal{B}_{\zeta \to Z} : \mathsf{CounterRH}(n,q) \to r_\infty \neq 0,$$

$$\mathsf{ZeroEventReturnInvariant}_m(r_m) : R_m r_m = r_m,$$

$$\mathsf{NoNeutralLeak}_m : r_m \in K_m^{\mathrm{skew}},$$

and

$$\mathsf{LayerSpectralExclusion}_m : 1 \notin \mathrm{spec}(R_m|_{K_m^{\mathrm{skew}}}).$$

Remark 9348.16 (Independent work before RH bridge)**.** *The address windows, kernel dimensions, compatible return operators, and spectral exclusions can be studied before any RH bridge is supplied. The largest bridge is the first obligation, which must route a* $\mathsf{CounterRH}(n,q)$ *event to a nonzero Zeckendorf residual tower. Until that bridge is certified, the layer-kernel induction is a structured target for residual elimination, not a proof of* RH*.*

Concretized in §278: Zeckendorf normal packets supply the finite no-adjacent-word evidence used by the layer residual spaces V_m.

Concretized in §554: The Axis-Zeckendorf cannot-claim packet supplies the refusal boundary that keeps the induction route from being read as a closed RH proof without the bridge rows.

Concretized in §1049: Axis carry route separation packets supply finite route-separation data for carry-facing residual sectors.

Chapter 9349

Zeta-to-Zeckendorf Bridge Obligations

chapter 9248 supplies the zeta-side separation row: an off-critical mirror pair is visible to the prime channels. chapter 9347 and chapter 9348 supply the Window-six and projection-kernel elimination form. The present chapter records the bridge still absent between these two surfaces. It does not claim a proof of RH; it isolates the exact rows that would be needed to route a zeta zero-skew packet into the Zeckendorf residual tower.

9349.1 Bridge interface

Definition 9349.1 (Prime tilt packet)**.** *For a complex history*

$$s = \frac{1}{2} + \delta + i\tau,$$

the prime tilt packet is the centered-amplitude family

$$\mathsf{Tilt}_\zeta(s) := \left(p^{-\delta}\right)_{p \in \mathsf{Prime}^\uparrow}.$$

It is nontrivial when $\delta \neq 0$. By Definition 9248.8, this is exactly the amplitude-bias row by which prime-local channels distinguish s from $J(s)$.

Definition 9349.2 (Zeta-to-Zeckendorf residual bridge)**.** *A zeta-to-Zeckendorf residual bridge is a displayed transformer*

$$\mathcal{B}_{\zeta \to Z} : \mathsf{OffMirrorZero}_\zeta(s) \to r_\infty(s),$$

where $r_\infty(s)$ is a residual tower in the sense of Definition 9348.4. Its finite projections are written

$$r_m(s) := P_m^{\mathcal{R}}(r_\infty(s)) \qquad (m \geq 6).$$

The bridge is certified only if it also exports the rows listed in Definition 9349.3.

Definition 9349.3 (Bridge certificate rows)**.** *The zeta-to-Zeckendorf bridge certificate has four required rows:*

$$\mathsf{NonzeroDetect}_{\zeta \to Z} : \delta(s) \neq 0 \Longrightarrow r_\infty(s) \neq 0,$$

$$\mathsf{ProjectionCompat}_{\zeta \to Z} : \Pi_{m+1,m}(r_{m+1}(s)) = r_m(s),$$

$$\mathsf{ZeroEventReturnInvariant}_m : R_m r_m(s) = r_m(s),$$

and

$$\mathsf{NoNeutralLeak}_m : r_m(s) \in K_m^{\mathrm{skew}}.$$

The nonzero-detection row transfers prime-visible off-critical bias to a nonzero Zeckendorf residual tower. The projection row makes the tower a compatible projection object. The return-invariance row says that the zero event is the same information packet after Zeckendorf return. The no-neutral-leak row excludes leakage into the neutral 1*-mode sector.*

Principle 9349.4 (Address carrier is not residual dynamics)**.** *The address identity*

$$X_6 = P_6(X_\infty^Z)$$

does not by itself construct the bridge of Definition 9349.2. The RH residual is not a no-adjacent address word. It is a weighted, linearized, or operator-facing packet in spaces of the form

$$V_m = \mathbb{Q}[X_m],$$

with compatible return operators R_m and layer kernels K_m. Therefore the bridge must supply residual-space data, not only address-space data.

9349.2 Neutral-mode obstruction

Definition 9349.5 (Neutral leak)**.** *For a return operator $R_m : V_m \to V_m$, a neutral leak of a residual $r_m(s)$ is a nonzero component in the* 1*-eigenspace of R_m. The bridge has no neutral leak at level m when the admitted residual lies in the skew sector:*

$$r_m(s) \in K_m^{\mathrm{skew}},$$

and the layer exclusion row

$$1 \notin \mathrm{spec}(R_m|_{K_m^{\mathrm{skew}}})$$

is available.

Proposition 9349.6 (Why the return eigenvalue alone is insufficient)**.** *The Window-six skew return law*

$$R_6 r = -\frac{1}{2} r$$

does not refute a residual tower unless the bridge also supplies zero-event return invariance and no-neutral-leak rows.

Proof. The eigenvalue row says how an admitted skew component transforms. A contradiction requires the return-invariance row

$$R_6 r = r,$$

so that $r = -\frac{1}{2} r$ forces $r = 0$. If a residual also contains a neutral component with $R_6 r = r$, that component is compatible with return invariance and is not eliminated by the $-\frac{1}{2}$ calculation. Hence the bridge must identify the residual as a skew-sector object and exclude neutral leakage before the Window-six return law can be used as a refutation step. □

Definition 9349.7 (All-layer bridge target)**.** *The all-layer bridge target is the conjunction*

$$\forall m \geq 6, \quad \mathsf{ZeroEventReturnInvariant}_m \wedge \mathsf{NoNeutralLeak}_m \wedge \mathsf{LayerSpectralExclusion}_m.$$

It is the projection-tower strengthening of the Window-six local residual calculation.

9349.3 Bridge-to-refutation criterion

Theorem 9349.8 (Bridge rows close the residual refutation)**.** *Assume an off-mirror zeta zero packet for s supplies*

$$\delta(s) \neq 0.$$

If a zeta-to-Zeckendorf residual bridge supplies all rows of Definition 9349.3 and the all-layer spectral exclusions of Definition 9349.7, then the corresponding nonzero residual tower is impossible.

Proof. The nonzero-detection row gives $r_\infty(s) \neq 0$. Projection compatibility makes $r_\infty(s)$ a residual tower. By Definition 9348.13, a nonzero admitted tower cannot satisfy zero-event return invariance, skew-sector admission, and layer spectral exclusion at each least nonzero layer. The bridge supplies precisely those rows. This contradicts $r_\infty(s) \neq 0$. □

Corollary 9349.9 (Bridge obstruction to a Window-six RH proof)**.** *Window six can serve as the initial finite fixed-shadow test for an RH residual only after the bridge rows are certified. Without Definition 9349.3, the route remains the implication target*

$$\mathsf{OffMirrorZero}_\zeta(s) \to \mathsf{NonzeroZeckendorfSkewResidual}(s),$$

not a proof that the target exists.

Proof. The Window-six calculation eliminates a residual already known to be a nonzero skew residual with return invariance. It does not construct that residual from a zeta zero packet. The construction and admissibility rows are exactly the bridge certificate rows. □

9349.4 Finite laboratory status

Definition 9349.10 (Window-six finite audit laboratory)**.** *The Window-six finite audit laboratory is the BEDC packet*

$$\mathsf{Lab}_6 := (C_6, X_6, D_6, \mathsf{Cert}_6, \mathsf{Audit}_6),$$

where

$$C_6 := \{0,1\}^6,$$

X_6 is the no-adjacent address window of Definition 9347.1, and

$$D_6 := C_6 \setminus X_6.$$

Thus

$$C_6 = X_6 \sqcup D_6,$$

with

$$|C_6| = 64, \qquad |X_6| = 21, \qquad |D_6| = 43.$$

Its records-side reading is the finite BEDC pipeline

$$\mathsf{carrier} \to \mathsf{constraint} \to \mathsf{partition} \to \mathsf{fiber} \to \mathsf{normal\ form} \to \mathsf{certificate} \to \mathsf{audit}.$$

Proposition 9349.11 (Finite defect prototype)**.** *For $w \in C_6$, the following are equivalent:*

$$w \in D_6,$$

and

$$w \text{ contains an adjacent pair } 11.$$

Consequently Window six supplies a finite prototype of the defect-exclusion shape

$$\mathsf{DetectedDefect}_6(w) \Longrightarrow w \notin X_6.$$

Proof. By definition, X_6 consists exactly of the six-bit words with no adjacent pair 11. Its complement inside the finite carrier C_6 is therefore exactly the set of six-bit words for which some adjacent pair is 11. Since C_6 is finite, the witnessing coordinate can be found by checking the five adjacent positions. □

Principle 9349.12 (Finite prototype is not carrier-level incompatibility)**.** *The finite implication of Definition 9349.11 has the same formal shape as the RH-facing bridge target*

$$\mathsf{ApartNonFix}_J(c) \Longrightarrow \mathsf{Apart}_0(\zeta(c)),$$

but it lives only in the finite address laboratory Lab_6*. It supplies a fully enumerable model of detected defect versus admissible geometry. It does not supply the completed zeta carrier, source compatibility, archimedean recovery rows, zero-ledger semantics, or the carrier-level incompatibility row*

$$\mathsf{DefectZeroIncompatibility}_\zeta.$$

9349.5 Work order

Principle 9349.13 (Local work before RH closure)**.** *The bridge route separates into local tasks on three surfaces: the pure Zeckendorf surface constructs compatible return operators and proves layer spectral exclusion on the skew sectors; the zeta surface encodes the prime tilt packet of Definition 9349.1 as a residual tower and proves nonzero detection; the zero-event semantic surface proves return invariance for that encoded residual. Only the conjunction of these tasks turns the Window-six fixed-shadow geometry into an RH-facing residual refutation route.*

Remark 9349.14 (Bridge status)**.** *The present chapter is a bridge-obligation ledger. It makes the unclosed rows explicit: prime tilt must become a nonzero Zeckendorf residual, the residual must remain compatible through all projections, zero events must be return invariant, and neutral mode leakage must be excluded. None of these rows follows from the existence of the golden-mean address tower or from the functional-equation mirror carrier alone.*

Concretized in §64: Prime packets supply the probe family whose amplitude tilt is the zeta-side input to the bridge.

Concretized in §278: Zeckendorf normal packets supply the no-adjacent address windows used to build the residual tower.

Concretized in §554: Axis-Zeckendorf refusal packets mark the boundary between bridge targets and closed RH proof rows.

Chapter 9350

Cannot-Claim Registry: AxisZeckendorf Cluster

This chapter aggregates the deliberate non-claims of the AxisZeckendorf cluster (Chapters 276–279). Each entry is paired either with a checked negative theorem in `BEDC.Derived.AxisZeckendorf` or with an explicit horizon obligation queued for the formalize pipeline. The non-claims protect the cluster's positioning: it is a conservative bridge layer above the finite kernel and below mature mathematical interfaces, not a numerological reinterpretation of binary words.

9350.1 Negative theorems backing the registry

Theorem 9350.1 (Display words `word_011` and `word_100` are not hsame)**.** $\mathsf{hsame}(\texttt{word_011}, \texttt{word_100})$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.CannotClaim.cannot claim 011 is infinity`

Proof. Theorem 9345.3. □

Theorem 9350.2 (Carry is not history-sameness, registry form)**.** *For every* h, k*,* $\mathsf{ZCarry}(h, k)$ *implies* $\neg\mathsf{hsame}(h, k)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.CannotClaim.cannot claim carry is hsame`

Proof. Theorem 9345.4. □

Theorem 9350.3 (`word_011` is not normal, registry form)**.** $\mathsf{ZNormal}(\texttt{word_011})$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.CannotClaim.cannot claim 011 is normal`

Proof. Theorem 278.4. □

Theorem 9350.4 (Boundary marker is not in the zero-spine carrier)**.** $\mathsf{ZeroSpine}(\texttt{boundary_01})$ *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.CannotClaim.cannot claim boundary 01 in zerospine`

Proof. Theorem 279.5. □

9350.2 Registry of non-claims

The following six entries are not assertions of the cluster. They are explicit refusals; each carries either a paper-level checked theorem above, or an upgrade path that names the missing certificate.

Cannot claim 1: `word_011` denotes infinity. Backed by Theorem 9350.1. `word_011` is a finite three-step history; no infinity object is ever attached to it inside the cluster.

Cannot claim 2: ZCarry is history-sameness. Backed by Theorem 9350.2. The classifier relates non-hsame histories; promoting it to a kernel equality is rejected.

Cannot claim 3: `boundary_01` is a limit / supremum. Backed by Theorem 9350.4, together with the chapter remarks of Chapter 279. The boundary token is a length-two finite history, not a limit object.

Cannot claim 4: $\mathsf{FullAxis}^{\uparrow}$ is $\mathsf{Real}^{\uparrow}$ (or $\mathsf{CompleteMetric}^{\uparrow}$). Horizon non-claim. Upgrade path: a separate bridge from $\mathsf{FullAxis}^{\uparrow}$ into the existing $\mathsf{CompleteMetric}^{\uparrow}$ chapter must be supplied; the cluster does not assert this bridge by itself.

Cannot claim 5: $\mathsf{AxisNat}^{\uparrow}$ replaces $\mathsf{Nat}^{\uparrow}$. Horizon non-claim. The two interfaces are kernel-distinct (Theorem 276.14); identification requires Chapter 9344, which is itself a non-kernel bridge.

Cannot claim 6: $\mathsf{DimLift}^{\uparrow}$ is a primitive geometric dimension. Horizon non-claim. $\mathsf{DimLift}^{\uparrow}$ is, in this cluster, a deferred lift seed: the cluster does not assert that the carry classifier produces a geometric dimension, a topological dimension, a vector-space dimension, a homological dimension, or a projective dimension. Promotion to any such reading requires a fresh certificate not part of the AxisZeckendorf cluster.

9350.3 Horizon obligations associated with the registry

Theorem 9350.5 ($\mathsf{DimLift}^{\uparrow}$ primitive non-claim, horizon)**.** *The cluster does not assert any kernel-level primitive geometric, topological, vector-space, homological, or projective dimension associated with the carry classifier.*

Theorem 9350.6 ($\mathsf{FullAxis}^{\uparrow} \neq \mathsf{Real}^{\uparrow}$ horizon)**.** *The cluster does not assert any of the field operations of* $\mathsf{Real}^{\uparrow}$ *on* $\mathsf{FullAxis}^{\uparrow}$*.*

Theorem 9350.7 ($\mathsf{AxisNat}^{\uparrow}$ does not replace $\mathsf{Nat}^{\uparrow}$ horizon)**.** *The cluster does not assert that* $\mathsf{AxisNat}^{\uparrow}$ *supersedes* $\mathsf{Nat}^{\uparrow}$*; both interfaces remain independent inside the finite kernel.*

Concretized in §554: The AxisZeckendorf cannot-claim registry is realized as a finite NameCert packet over checked negative rows, horizon refusal rows, hsame transport, Cont routes, Pkg provenance, and non-escape ledgers.

Concretized in §571: DimLift boundary packets realize the AxisZeckendorf dimension-lift refusal as finite BHist NameCert data over carry, normal-address, non-escape, and public boundary rows.

Concretized in §602: FullAxis-to-Real refusal packets realize the AxisZeckendorf boundary as finite FullAxisUp, RealUp, hsame, Cont, Pkg, and NameCert rows without exporting RealUp operations.

Concretized in §672: AxisNat replacement refusals are realized as finite NameCert packets separating AxisNatUp, NatUp, bridge rows, checked negative rows, and non-replacement ledgers.

Concretized in §1018: Boundary-token limit refusal is realized as a finite NameCert packet separating the AxisZeckendorf boundary row from limit, supremum, RealUp equality, and completeness readings.

Concretized in §1196: Axis carry-refusal ledgers realize the AxisZeckendorf cannot-claim registry as finite negative-row, horizon-refusal, hsame, Cont, Pkg, and NameCert packets.

Concretized in §567: The Zeckendorf carry classifier realizes the finite carry-normalization direction of the Axis-Zeckendorf route.

Chapter 9351

Mark–Hist–Ext–Cont Kernel Spine

The finite BEDC kernel begins below packages, names, signatures, and ordinary mathematical objects. Its first visible surface is the passage from raw marks to generated histories and then to one-step extension and continuation. This chapter records only that initial spine:

$$\mathsf{BMark} \longrightarrow \mathsf{BHist} \longrightarrow (\mathsf{Ext}, \mathsf{Cont}).$$

Later layers such as Ask, Sig, sameSig, Pkg, Gap, and NameCert depend on this spine but are not part of the present surface.

9351.1 Raw marks

Definition 9351.1 (Raw mark carrier)**.** *The raw mark carrier is the closed two-constructor carrier*

$$\mathsf{BMark} := \{\mathsf{b0}, \mathsf{b1}\}_{\mathrm{gen}}.$$

The two constructors are primitive visible emissions. They are not the numbers 0 *and* 1*, not truth values, and not elements drawn from an ambient set.*

Definition 9351.2 (Mark sameness)**.** *The mark-sameness judgment*

$$\mathsf{msame} : \mathsf{BMark} \to \mathsf{BMark} \to \mathsf{Prop}$$

is generated only by the two diagonal rules

$$\mathsf{msame}(\mathsf{b0}, \mathsf{b0}), \qquad \mathsf{msame}(\mathsf{b1}, \mathsf{b1}).$$

No rule generates cross-mark sameness.

Principle 9351.3 (Closed generation at the mark layer)**.** *At the mark layer, a judgment is internal only when it is generated by the displayed constructors and rules. There is no hidden conversion, hidden coercion, ambient extensionality rule, or external equality that identifies* b0 *with* b1*.*

Theorem 9351.4 (Mark sameness is diagonal)**.** *Mark sameness is reflexive, symmetric, and transitive on generated marks, and the cross judgments*

$$\mathsf{msame}(\mathsf{b0}, \mathsf{b1}), \qquad \mathsf{msame}(\mathsf{b1}, \mathsf{b0})$$

are not generated.

Proof. Closed generation gives only two mark cases. The diagonal judgments supply reflexivity. Symmetry and transitivity never leave the diagonal cases, because any premise of mark sameness is already one of those cases. The cross judgments are not conclusions of either displayed rule, hence are not generated. □

9351.2 Histories

Definition 9351.5 (History carrier)**.** *The history carrier is the closed generated carrier*

$$\mathsf{BHist}$$

with constructors

$$\mathsf{Empty} : \mathsf{BHist}, \qquad \mathsf{e0} : \mathsf{BHist} \to \mathsf{BHist}, \qquad \mathsf{e1} : \mathsf{BHist} \to \mathsf{BHist}.$$

The empty constructor is the empty history. The constructors e0 *and* e1 *append one visible* b0 *or* b1 *emission to an existing generated history.*

Definition 9351.6 (History sameness)**.** *The history-sameness judgment*

$$\mathsf{hsame} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$$

is generated by the empty rule

$$\mathsf{hsame}(\mathsf{Empty}, \mathsf{Empty})$$

and the two constructor-congruence rules

$$\mathsf{hsame}(h, k) \Longrightarrow \mathsf{hsame}(\mathsf{e0}(h), \mathsf{e0}(k)),$$

$$\mathsf{hsame}(h, k) \Longrightarrow \mathsf{hsame}(\mathsf{e1}(h), \mathsf{e1}(k)).$$

There is no rule identifying an empty history with an extended history, and no rule identifying an e0*-headed history with an* e1*-headed history.*

Theorem 9351.7 (History no-confusion)**.** *For generated histories h, k, the following judgments are not generated:*

$$\mathsf{hsame}(\mathsf{Empty}, \mathsf{e0}(h)), \qquad \mathsf{hsame}(\mathsf{Empty}, \mathsf{e1}(h)),$$

$$\mathsf{hsame}(\mathsf{e0}(h), \mathsf{e1}(k)), \qquad \mathsf{hsame}(\mathsf{e1}(h), \mathsf{e0}(k)).$$

Proof. Inspect the displayed generation rules for hsame. The empty rule has empty source and target. The e0 rule has e0 on both sides. The e1 rule has e1 on both sides. None of these rules has a cross-constructor conclusion. □

Corollary 9351.8 (Histories record finite distinction sequences)**.** *Every generated history records a finite sequence of raw mark emissions, and* hsame *preserves the generated constructor shape of that sequence.*

Proof. Define the sequence readout recursively by sending Empty to the empty word, $\mathsf{e0}(h)$ to the readout of h followed by a b0 emission, and $\mathsf{e1}(h)$ to the readout of h followed by a b1 emission. The no-confusion theorem shows that hsame cannot cross the empty, e0, and e1 constructors. □

9351.3 Extension and continuation

Definition 9351.9 (One-step extension)**.** *The one-step extension relation is*

$$\mathsf{Ext} : \mathsf{BHist} \to \mathsf{BMark} \to \mathsf{BHist} \to \mathsf{Prop}.$$

Its generation rules are

$$\mathsf{Ext}(h, \mathsf{b0}, \mathsf{e0}(h)), \qquad \mathsf{Ext}(h, \mathsf{b1}, \mathsf{e1}(h)).$$

Thus $\mathsf{Ext}(h, m, r)$ *records that the history* r *is obtained from* h *by one displayed mark emission.*

Theorem 9351.10 (Extension result characterization)**.** *For generated* m, h, t*, the extension judgment ending in an* $\mathsf{e0}$*-history has the form*

$$\mathsf{Ext}(h, m, \mathsf{e0}(t)) \iff \mathsf{msame}(m, \mathsf{b0}) \wedge \mathsf{hsame}(h, t),$$

and the extension judgment ending in an $\mathsf{e1}$*-history has the form*

$$\mathsf{Ext}(h, m, \mathsf{e1}(t)) \iff \mathsf{msame}(m, \mathsf{b1}) \wedge \mathsf{hsame}(h, t).$$

Proof. For the first equivalence, inspect the generation rules for Ext. A result with head $\mathsf{e0}$ can only come from the $\mathsf{b0}$ rule, and its tail is the source history up to hsame. Conversely, the $\mathsf{b0}$ rule gives $\mathsf{Ext}(h, \mathsf{b0}, \mathsf{e0}(h))$, and the displayed sameness rows transport the mark and source history. The $\mathsf{e1}$ case is the same argument with $\mathsf{b1}$ and $\mathsf{e1}$. □

Theorem 9351.11 (Same-source cross-mark extensions are separated)**.** *If*

$$\mathsf{Ext}(h, \mathsf{b0}, r_0), \qquad \mathsf{Ext}(h, \mathsf{b1}, r_1),$$

then $\mathsf{hsame}(r_0, r_1)$ *is not generated.*

Proof. The first extension has result hsame-classified with $\mathsf{e0}(h)$. The second has result hsame-classified with $\mathsf{e1}(h)$. If $\mathsf{hsame}(r_0, r_1)$ were generated, transitivity would generate $\mathsf{hsame}(\mathsf{e0}(h), \mathsf{e1}(h))$, contradicting Definition 9351.7. □

Definition 9351.12 (Continuation)**.** *The continuation relation is*

$$\mathsf{Cont} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}.$$

It records the BEDC-internal form of history concatenation. Its basic interface includes empty-continuation rows, one-step compatibility with $\mathsf{e0}$ *and* $\mathsf{e1}$*, unit rows, and associativity up to* hsame*. These rows are L2 obligations over the same* BHist *carrier; they do not introduce another visible object layer.*

Theorem 9351.13 (Continuation unit and associativity obligations)**.** *Continuation has empty-history unit rows*

$$\mathsf{Cont}(\mathsf{Empty}, h, h), \qquad \mathsf{Cont}(h, \mathsf{Empty}, h),$$

and threefold continuation is associative up to hsame*: if*

$$\mathsf{Cont}(a, b, u), \qquad \mathsf{Cont}(u, c, v),$$

$$\mathsf{Cont}(b, c, w), \qquad \mathsf{Cont}(a, w, z),$$

then $\mathsf{hsame}(v, z)$ *is generated.*

Proof. The unit rows are the empty-history clauses of continuation. For associativity, both displayed composites represent the same generated history obtained by first reading a, then b, then c. The constructor recursion defining continuation therefore classifies the two results by the same BHist shape, so $\mathsf{hsame}(v, z)$ is generated. □

Definition 9351.14 (Initial kernel spine)**.** *The initial BEDC kernel spine is the tuple*

$$\mathcal{T}_{\leq 2} := (\mathsf{BMark}, \mathsf{msame}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Ext}, \mathsf{Cont}).$$

Theorem 9351.15 (Initial kernel spine separation)**.** *The initial kernel spine separates raw marks, separates history constructors, and separates same-source* $\mathsf{b0}$ *and* $\mathsf{b1}$ *extension results.*

Proof. Raw marks are separated by Definition 9351.4. History constructors are separated by Definition 9351.7. Same-source cross-mark extensions are separated by Definition 9351.11. □

Corollary 9351.16 (Minimal binary recursive readout)**.** *The initial kernel spine supports the recursive readout*

$$P_{\mathsf{B}} : \mathsf{BHist} \to \{\mathsf{b0}, \mathsf{b1}\}^{<\omega}$$

defined by

$$P_{\mathsf{B}}(\mathsf{Empty}) = \epsilon, \qquad P_{\mathsf{B}}(\mathsf{e0}(h)) = P_{\mathsf{B}}(h)\mathsf{b0}, \qquad P_{\mathsf{B}}(\mathsf{e1}(h)) = P_{\mathsf{B}}(h)\mathsf{b1}.$$

Proof. The recursion follows the three constructors of BHist. Definition 9351.7 supplies uniqueness of the visible last step, and Definition 9351.4 supplies the two separated terminal emissions. □

Principle 9351.17 (Kernel spine as the self-similar starting surface)**.** *The pair* $\mathsf{b0}, \mathsf{b1}$ *supplies the minimal visible distinction surface, and the constructors* $\mathsf{e0}, \mathsf{e1}$ *supply the recursive history surface. This is the bottom support for later window, Zeckendorf, prime-coordinate, zeta-carrier, and zero-ledger routes. Higher layers may compress, package, or project these histories only after they expose the source fiber and the ledger rows that license the compression.*

Concretized in §8779: The finite mark-history-continuation spine is realized as a concrete NameCert packet.

Principle 9351.18 (Next layer boundary)**.** *The next layer begins only after this spine, with*

$$\mathsf{Ask} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{Gap} \to \mathsf{NameCert}.$$

Its role is not to add primitive mathematical objects to the kernel, but to expose the source ledger for visible objects obtained by asking, signature formation, packaging, gap accounting, and certification.

Chapter 9352

Ask–Sig–Pkg–Gap–NameCert Kernel Spine

The lower spine of chapter 9351 supplies raw marks, generated histories, one-step extension, and continuation. The finite kernel then reaches its certificate-facing boundary by adding asking, signature formation, signature sameness, package tokens, gap ledgers, exact globalization, and naming certificates:

$$\mathsf{Mark} \to \mathsf{Hist} \to (\mathsf{Ext}, \mathsf{Cont}) \to (\mathsf{Ask}, \mathsf{Sig}) \to \mathsf{sameSig} \to (\mathsf{Pkg}, \mathsf{Gap}) \to \mathsf{NameCert}.$$

This chapter records the L3–L5 spine only. It does not introduce mature mathematical objects, completion objects, folds, phase spaces, zeta carriers, or zero ledgers. Those later interfaces may use this spine only through displayed certificate fields.

9352.1 Asking and signatures

Definition 9352.1 (Probe names and bundles)**.** *A probe name* $\pi : \mathsf{ProbeName}$ *is an internal name that may be asked against a generated history. It is not an external function. Probe bundles are the closed carrier generated by*

$$\mathsf{Bnil} : \mathsf{Bundle}, \qquad \mathsf{Bcons}(\pi, \Pi) : \mathsf{Bundle}$$

for $\pi : \mathsf{ProbeName}$ *and* $\Pi : \mathsf{Bundle}$*.*

Definition 9352.2 (Ask event)**.** *An ask event is a generated relation*

$$\mathsf{Ask}(\pi, h, m, \delta)$$

where $\pi : \mathsf{ProbeName}$*,* $h : \mathsf{Hist}$*,* $m : \mathsf{Mark}$*, and* δ *is the event evidence. It records that probe* π *asked against source history* h *emits mark* m *with evidence* δ*. The evidence is a ledger item for the event, not a numeric bias or an external truth value.*

Definition 9352.3 (Ask policy)**.** *For a bundle* Π *and source domain* $D : \mathsf{Hist} \to \mathsf{Prop}$*, an asking policy* $\mathsf{AskPolicy}(\Pi, D)$ *supplies the following internal obligations: totality of ask events for admitted histories and bundle members, determinacy of the emitted mark for the same probe and source history, and respect for* hsame *on admitted source histories.*

Definition 9352.4 (Signature event)**.** *The signature event*

$$\mathsf{Sig}(\Pi, h, s, \Delta)$$

means that bundle Π *applied to source history* h *generates answer history* s *with evidence ledger* Δ*. The empty bundle generates the empty answer history:*

$$\mathsf{Sig}(\mathsf{Bnil}, h, \mathsf{e}, \Delta_0).$$

The cons rule is generated from a head ask event, a tail signature, and one extension step:

$$\mathsf{Ask}(\pi, h, m, \delta), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Ext}(m, s, r)$$

imply

$$\mathsf{Sig}(\mathsf{Bcons}(\pi, \Pi), h, r, \delta \star \Delta).$$

Theorem 9352.5 (Signature inversion and determinacy)**.** *The L3 signature relation satisfies the following finite-kernel facts.*

1. *Empty-bundle signatures have answer history* hsame *to* e*.*
2. *Cons-bundle signatures have answer history not* hsame *to* e*.*
3. *If* $\mathsf{AskPolicy}(\Pi, D)$*,* $h \in D$*, and*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, h, t, \Theta),$$

then $\mathsf{hsame}(s, t)$*.*

Proof. The first clause is inversion of the empty-signature rule. The second clause inverts the cons-signature rule; its last step is an extension, whose result has E0 or E1 head and therefore cannot be hsame to e by history no-confusion. The determinacy clause is induction on Π. The empty case reduces to the first clause and hsame equivalence. In the cons case, invert both signatures. Ask-policy determinacy gives msame of the two head marks, the induction hypothesis gives hsame of the two tail answer histories, and extension respects internal sameness. □

Definition 9352.6 (Signature sameness)**.** *For a bundle* Π*, two histories are signature-same when their generated answer histories are* hsame*:*

$$\mathsf{sameSig}_\Pi(h, k) \quad :\Longleftrightarrow \quad \exists s, t, \Delta, \Theta.\ \mathsf{Sig}(\Pi, h, s, \Delta) \wedge \mathsf{Sig}(\Pi, k, t, \Theta) \wedge \mathsf{hsame}(s, t).$$

Theorem 9352.7 (Signature sameness is the L3 classifier)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. On admitted histories,* $\mathsf{sameSig}_\Pi$ *is reflexive, symmetric, and transitive; the transitive step through a middle history* k *uses the hypothesis* $k \in D$*.*

Proof. Reflexivity uses ask-policy totality to generate one signature for the admitted history and then uses hsame reflexivity. Symmetry swaps the two signature witnesses and uses hsame symmetry. For transitivity, unpack the two signature-sameness witnesses. They contain two generated signatures from the admitted middle history. Signature determinacy identifies the two middle answer histories by hsame, and hsame transitivity connects the left answer history to the right answer history. □

9352.2 Packages, gaps, and exact globalization

Definition 9352.8 (Package token and package policy)**.** *The package relation*

$$\mathsf{Pkg}(\Pi, s, p)$$

records that answer history s under bundle Π is represented by visible package token p. A package policy $\mathsf{PackagePolicy}(\Pi)$ *supplies token existence, extensionality from* $\mathsf{hsame}(s,t)$ *to* $\mathsf{psame}(p,q)$ *for tokens introduced by s and t, source reflection of* $\mathsf{psame}(p,q)$ *back in generated answer histories, and visibility of the token interface without exposing the full source history as a public object.*

Theorem 9352.9 (Packages classify signatures)**.** *Assume* $\mathsf{PackagePolicy}(\Pi)$*. If*

$$\mathsf{Pkg}(\Pi, s, p), \qquad \mathsf{Pkg}(\Pi, t, q),$$

then $\mathsf{hsame}(s,t)$ *implies* $\mathsf{psame}(p,q)$*. Conversely,* $\mathsf{psame}(p,q)$ *is backed by package witnesses for answer histories internally related by* hsame*.*

Proof. The forward direction is the extensionality field of the package policy. The converse is the source-reflection field. Therefore package-token behavior is a licensed theorem of the package policy, not a primitive quotient shortcut. □

Definition 9352.10 (Gap membership and gap policy)**.** *The gap membership relation*

$$\mathsf{Gap}(\Pi, D, p, h)$$

records that admitted source history h lies in the ledger behind visible package token p. A gap policy $\mathsf{GapPolicy}(\Pi, D)$ *supplies coverage of admitted histories, separation of two package tokens that cover the same source history, and signature determination: every gap membership is backed by a generated signature and a package-token witness.*

Theorem 9352.11 (Gap coverage and separation)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*, and* $\mathsf{GapPolicy}(\Pi, D)$*. If $h \in D$, then there is a package token p such that* $\mathsf{Gap}(\Pi, D, p, h)$*. If*

$$\mathsf{Gap}(\Pi, D, p, h), \qquad \mathsf{Gap}(\Pi, D, q, h),$$

then $\mathsf{psame}(p,q)$*.*

Proof. Coverage is the coverage field of the gap policy, with ask totality and package existence supplying the generated signature and token when the policy is unfolded. For separation, unfold the two gap memberships. They produce two signatures from the same source history and two package-token witnesses. Signature determinacy gives hsame of the two answer histories, and package extensionality gives $\mathsf{psame}(p,q)$. □

Principle 9352.12 (Compression carries memory)**.** *Packaging, hiding, projecting, sealing, or otherwise compressing generated histories into a visible token is admissible only with a gap ledger. Without the ledger, the visible token has no internal record of the source histories compressed into it.*

Concretized in §8979: AskSigGapRouteUp realizes the Ask–Sig–Pkg–Gap exact-globalization spine as finite NameCert route data.

Definition 9352.13 (Generated signature sameness witness)**.** *The witness object* $\mathsf{GeneratedSameSig}(\Pi, h, k)$ *consists of answer histories* s, t*, evidence ledgers* Δ, Θ*, generated signatures*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta),$$

and the internal sameness row $\mathsf{hsame}(s, t)$*. Its inhabitedness is the checker-facing form of the signature-classification statement.*

Definition 9352.14 (Exact globalize base)**.** *An exact finite globalization base* $\mathsf{ExactGlobalizeBase}(\Pi, D)$ *consists of coverage, base soundness, and base completeness:*

$$h \in D \Longrightarrow \exists p.\ \mathsf{Gap}(\Pi, D, p, h),$$

$$\mathsf{Gap}(\Pi, D, p, h) \wedge \mathsf{Gap}(\Pi, D, q, k) \wedge \mathsf{GeneratedSameSig}(\Pi, h, k) \Longrightarrow \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q),$$

and

$$\mathsf{Gap}(\Pi, D, p, h) \wedge \mathsf{Gap}(\Pi, D, q, k) \wedge \mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \Longrightarrow \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

Theorem 9352.15 (Exact globalize base classification)**.** *If* $\mathsf{ExactGlobalizeBase}(\Pi, D)$ *and the two displayed package tokens carry gap witnesses*

$$\mathsf{Gap}(\Pi, D, p, h), \qquad \mathsf{Gap}(\Pi, D, q, k),$$

then

$$\mathsf{psame}^{\mathrm{base}}_{\Pi}(p, q) \Longleftrightarrow \mathsf{Nonempty}(\mathsf{GeneratedSameSig}(\Pi, h, k)).$$

Proof. The forward implication is base completeness. The reverse implication unpacks the inhabited $\mathsf{GeneratedSameSig}(\Pi, h, k)$ witness and applies base soundness. Coverage remains part of the exact globalization structure because every admitted history must have a visible package ledger, even though this local equivalence only needs the two displayed gap witnesses. □

9352.3 Name certification

Definition 9352.16 (NameCert fields)**.** *For a proposed derived interface* N*, the paper naming certificate is the five-field packet*

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The source field records the generated material from which N *is licensed. The pattern field records the stable behavior observed in that source. The classifier field records how source material is grouped, normally through a bundle, asking policy, package policy, and gap policy at the finite-kernel level. The stability field records closure, reuse, descent, composition, or seal behavior. The ledger policy records what the name hides and how that hidden material remains tracked.*

Theorem 9352.17 (Derived interfaces require NameCert)**.** *A derived interface* N *is admitted only through* $\mathsf{NameCert}(N)$*.*

Proof. The finite kernel has no mature-object primitive beyond its generated judgments and certificate records. A derived name therefore cannot enter by host familiarity, notation, quotient collapse, or semantic intention. It enters only when source, pattern, classifier, stability, and ledger fields are all supplied. □

Definition 9352.18 (Certified existence)**.** *For a proposed object or interface name X, BEDC-certified existence is the judgment*

$$\mathsf{Exists}^{\uparrow}(X) := \sum_{N:\mathsf{NameCert}(X)} \mathsf{Admits}(N).$$

The admission row records that the source, pattern, classifier, stability, and ledger fields of N are available to downstream consumers. Thus existence is a construction certificate for the named interface, not prior possession of an untyped object.

Principle 9352.19 (Existence is certificate-bearing admission)**.** *In BEDC, a mathematical name is usable only as far as its construction certificate exposes the relevant source, classifier, boundary, transport, and ledger rows. A zeta carrier, zero packet, Hilbert-style surface, solenoid phase carrier, Pick kernel, completion, quotient, or limit may be used only after the corresponding certificate has named what is generated, what is hidden, and how the hidden material remains auditable.*

Theorem 9352.20 (Certified existence is the object-layer admission rule)**.** *Every object-layer use of a derived interface factors through $\mathsf{Exists}^{\uparrow}(X)$.*

Proof. By Definition 9352.17, a derived interface X enters the finite kernel only through $\mathsf{NameCert}(X)$. The additional admission row in Definition 9352.18 records that the fields of that certificate are available for the intended object-layer use. Thus any use of X factors through the displayed pair consisting of the naming certificate and its admission row. □

Definition 9352.21 (BEDC mathematical ontology core)**.** *The BEDC mathematical ontology core is the certificate-bearing admission spine*

$$\mathsf{BMark} \to \mathsf{BHist} \to (\mathsf{msame}, \mathsf{hsame}) \to (\mathsf{Ext}, \mathsf{Cont}) \to (\mathsf{Ask}, \mathsf{Sig}) \to (\mathsf{Pkg}, \mathsf{Gap}) \to \mathsf{NameCert} \to \mathsf{Math}^{\uparrow}.$$

Here $\mathsf{Math}^{\uparrow}$ denotes the family of public mathematical interfaces admitted after certification. The displayed chain is not an additional object carrier. It is the rule that reads ordinary mathematical grammar from generated histories, classifier-stable packages, visible gap ledgers, and naming certificates.

Definition 9352.22 (Dynamic ontology state row)**.** *A certified interface carries a dynamic ontology state row when its admission data expose a finite state*

$$\mathcal{K}_t = (H_t, E_t, C_t, R_t, L_t).$$

Here H_t is the generated source history visible to the interface, E_t is the displayed event or formation row that extends that history, C_t is the canonical code or normal-form row, R_t is the public readout row, and L_t is the residual ledger row. A transition

$$\mathcal{K}_t \longrightarrow \mathcal{K}_{t+1}$$

is admitted only when the later state extends H_t by a displayed event, recomputes C_t and R_t by the stated rules, and transports every open residual into the displayed ledger policy.

Definition 9352.23 (BEDC–PZG dynamic tuple)**.** *The BEDC–PZG dynamic tuple is the coordinate presentation*

$$\mathcal{K}_{\mathrm{dyn}} := (\mathsf{H}, \mathsf{G}, \mathsf{Z}, \mathsf{D}, \mathsf{N}, \mathsf{R}, \mathsf{L}, \mathsf{S}, \Phi, K_\infty, \Sigma_\infty, \mathsf{M}).$$

Here H *is the history carrier,* G *is the displayed generation operator,* Z *is the PZG register,* D *is its decoding row,* N *is the normalization row,* R *is finite readout,* L *is the residual ledger,* S *is finite self-description,* Φ *is phase readout,* K_∞ *is the hidden prime-fibre row,* Σ_∞ *is the solenoid phase carrier, and* M *is the motion certificate when the interface has a motion surface. This tuple is not an object preceding the ontology core; it is a coordinate display of the rows already admitted by Definition 9352.22 and the public interfaces that consume them.*

Principle 9352.24 (Dynamic construction route)**.** *The dynamic tuple is used only through certified formation routes of the shape*

$$\mathsf{Term} \to \mathsf{Rewrite} \to \mathsf{Congruence} \to \mathsf{Localize} \to \mathsf{Complete} \to \mathsf{Globalize}.$$

Each arrow is a finite event row: it extends the source history, supplies a canonical readout for the target stage, and transports the residual ledger. Thus localization, completion, global phase readout, and motion certification are downstream consumers of the same dynamic state discipline rather than separate ontological primitives.

Principle 9352.25 (Ontology triad)**.** *The ontology core presents every certified mathematical interface through three inseparable rows. The generation row records how the interface is produced from marks, histories, continuations, questions, signatures, packages, gaps, and certificates. The canonical readout row records the normal form, classifier, unit phase, unit norm, or standard decoding by which the interface is publicly usable. The ledger row records every residual not closed by that readout, including hidden source material, tail windows, semantic self-reference, and compatibility defects.*

Theorem 9352.26 (Certified formation inherits the ontology triad)**.** *Let* Y *be generated from certified interfaces* $X_1, \ldots, X_m$ *by a displayed finite formation rule. If each* X_i *carries a dynamic ontology state row and the formation rule displays event transport, canonical readout transport, and ledger transport, then* Y *carries the ontology triad of Definition 9352.25.*

Proof. The event transport appends the formation rule to the source histories of $X_1, \ldots, X_m$, giving the generation row for Y. The canonical readout transport supplies the normal form, classifier, unit row, or decoding row by which Y is publicly read. The ledger transport carries the source ledgers forward and records the residuals created by the formation rule itself. These are precisely the three rows required by Definition 9352.25. □

Theorem 9352.27 (Object-layer factorization through the ontology core)**.** *Every public object-layer use of a certified mathematical interface factors through the spine of Definition 9352.21.*

Proof. The lower spine supplies raw marks, generated histories, mark sameness, history sameness, extension, and continuation. The present L3–L5 surface then supplies asking, generated signatures, package tokens, gap coverage, and exact local classification where claimed. By Definition 9352.17, no derived interface enters by host familiarity, notation, quotient collapse, or semantic intention. By Definition 9352.20, each object-layer use factors through the relevant naming certificate and admission row. Composing these displayed passages gives the stated factorization. □

Corollary 9352.28 (Mature mathematical names are certificate consumers)**.** *Interfaces such as*

$$\mathsf{Nat}^{\uparrow}, \quad \mathsf{Int}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{Cat}^{\uparrow}, \quad \mathsf{Prime}^{\uparrow}, \quad \mathsf{Zeta}^{\uparrow}$$

are not kernel primitives. Each is usable only at the scope exposed by its source specification, classifier, stability rows, ledger policy, and local admission certificate.

Proof. Apply Definition 9352.27 to each displayed upward interface. The factorization route ends in a $\mathsf{NameCert}$ and its admission row before any public mathematical read is available. Hence the public name is a certified interface over generated BEDC material rather than an independent primitive object. □

Principle 9352.29 (Ledger policy for hidden material)**.** *Any derived interface that hides a process, tail, source fiber, phase row, gap row, or compatibility thread must expose a* $\mathsf{LedgerPolicy}$ *witness. This applies in particular to completion-like, fold-like, phase-like, geometric, algebraic, analytic, zeta-facing, and zero-ledger-facing interfaces.*

Theorem 9352.30 (L3–L5 finite-kernel closure)**.** *Given* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*,* $\mathsf{GapPolicy}(\Pi, D)$*, and the exact globalize fields where claimed, the L3–L5 spine supplies:*

1. *signature determinacy for each admitted source history;*
2. $\mathsf{sameSig}_{\Pi}$ *as the internal signature classifier on admitted histories;*
3. *package classification of answer histories;*
4. *gap coverage for admitted source histories;*
5. *gap separation for same-source package overlaps;*
6. *exact local classification by* $\mathsf{ExactGlobalizeBase}(\Pi, D)$*;*
7. *certificate admission for derived interfaces through* $\mathsf{NameCert}$*.*

Proof. The first clause is Definition 9352.5. The second is Definition 9352.7. The third is Definition 9352.9. The fourth and fifth are Definition 9352.11. The sixth is Definition 9352.15. The final clause is Definition 9352.17. □

Principle 9352.31 (Finite-data boundary)**.** *The next layer begins only after the L0–L5 spine has supplied these policies and ledgers. Finite data interfaces such as* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*, and* $\mathsf{List}^{\uparrow}$ *are therefore certificate consumers, not additional kernel primitives.*

Chapter 9353

Finite Data Interface Spine

The finite-kernel spine of chapter 9351 and chapter 9352 reaches the first data-facing layer through certificates. The finite data layer does not add truth values, option values, disjoint sums, ordered pairs, or lists as kernel inputs. It licenses them as derived interfaces over histories, history sameness, continuation, package ledgers, stability witnesses, and name certificates:

$$(\mathsf{Hist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Gap}, \mathsf{NameCert}) \longrightarrow (\mathsf{Bool}^{\uparrow}, \mathsf{Option}^{\uparrow}, \mathsf{Sum}^{\uparrow}, \mathsf{Prod}^{\uparrow}, \mathsf{List}^{\uparrow}).$$

The roadmap obligation for this layer is stated in section 9467.8: each public finite-data object must display a carrier predicate over Hist, a classifier with stability, constructor and case-analysis behavior, and a ledger for the information hidden by its public surface.

9353.1 Interface Template

Definition 9353.1 (Finite data interface)**.** *A finite data interface is a certificate packet*

$$\mathsf{FinData}^{\uparrow}(A, \sim_A)$$

over histories, consisting of the data

$$(A, \sim_A, \mathsf{Constr}_A, \mathsf{Elim}_A, \mathsf{Stab}_A, \mathsf{Ledger}_A, \mathsf{NameCert}(A, \sim_A)).$$

Here

$$A : \mathsf{Hist} \to \mathsf{Prop}$$

is the carrier predicate, and

$$\sim_A : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop}$$

is the public classifier. The constructor field records how public constructors create admitted histories. The elimination field records the case-analysis or inversion principles allowed by those constructors. The stability field records reflexivity, symmetry, transitivity, and carrier transport for the classifier. The ledger records tag bits, branch choices, length rows, component witnesses, or other material suppressed by the public name.

Definition 9353.2 (Finite data admission)**.** *An interface $A^{\uparrow}$ is admitted at the finite data layer when the packet of Definition 9353.1 is displayed and its certificate supplies*

$$\mathsf{carrier_inhabited}, \quad \mathsf{classifier_refl}, \quad \mathsf{classifier_symm},$$

$$\mathsf{classifier_trans}, \quad \mathsf{carrier_transport},$$

together with the advertised constructor, inversion, and ledger fields.

Principle 9353.3 (Finite data is certificate-facing)**.** *No finite data interface is accepted from host notation alone. A public read of* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, *or* $\mathsf{List}^{\uparrow}$ *must pass through a displayed carrier, classifier, stability certificate, ledger, and name certificate.*

Concretized in §8899: Finite-data interface spine realized as one L6 NameCert packet.

Theorem 9353.4 (Finite data interfaces are not kernel inputs)**.** *If* $A^{\uparrow}$ *is admitted by Definition 9353.2, then* $A^{\uparrow}$ *is not a BEDC kernel input.*

Proof. The only objects available before the finite data layer are the lower generated histories, relations, policies, packages, gaps, and naming certificates. Admission by Definition 9353.2 explicitly uses those records. Thus the public interface is a licensed name over the displayed records, not an additional carrier inserted into the kernel. □

Concretized in §8898: Finite assembly NameCert for the Bool/Option/Sum/Prod/List interface spine.

9353.2 Boolean Endpoint Interface

Definition 9353.5 (Boolean history carrier)**.** *The Boolean finite-data carrier is the two-endpoint predicate*

$$\mathsf{BoolCarrier}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e}) \vee \mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})).$$

The endpoint e *is the false endpoint, and* $\mathsf{E1}(\mathsf{e})$ *is the true endpoint.*

Definition 9353.6 (Boolean history classifier)**.** *The Boolean classifier relates exactly histories in the same displayed endpoint class:*

$$\mathsf{BoolClassifier}(h, k) :\Longleftrightarrow \big(\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})\big) \vee \big(\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(k, \mathsf{E1}(\mathsf{e}))\big).$$

Definition 9353.7 (Boolean constructors)**.** *The two public constructors are the displayed histories*

$$\mathsf{false}^{\uparrow} := \mathsf{e}, \qquad \mathsf{true}^{\uparrow} := \mathsf{E1}(\mathsf{e}).$$

Theorem 9353.8 (Boolean endpoint exactness)**.** *For admitted Boolean histories* h, k*, the classifier* $\mathsf{BoolClassifier}(h, k)$ *holds exactly when both histories lie in the false endpoint class or both lie in the true endpoint class.*

Proof. Unfold Definition 9353.6. The classifier has only the two displayed alternatives, and each alternative requires both histories to be in the same endpoint class. □

Theorem 9353.9 (Boolean constructor separation)**.** *The endpoint classifier does not relate the two Boolean constructors:*

$$\neg\, \mathsf{BoolClassifier}(\mathsf{e}, \mathsf{E1}(\mathsf{e})).$$

Proof. If the classifier held, Definition 9353.6 would put e and $\mathsf{E1}(\mathsf{e})$ in one endpoint class. Each class equality contradicts history no-confusion for the empty and one-extended constructors. □

Corollary 9353.10 (Boolean finite data admission)**.** *The interface* $\mathsf{Bool}^{\uparrow}$ *is admitted by its carrier, endpoint classifier, two displayed constructors, endpoint separation, stability rows, ledger row, and* $\mathsf{NameCert}$ *or* $\mathsf{SemanticNameCert}$ *row.*

9353.3 Optional History Interface

Definition 9353.11 (Optional carrier)**.** *Let $A : \mathsf{Hist} \to \mathsf{Prop}$ be an admitted source carrier with classifier $\sim_A$. The optional carrier is*

$$\mathsf{OptionCarrier}_A(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e}) \vee \exists a.\ A(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)).$$

The empty endpoint is the absent case, and histories of the form $\mathsf{E1}(a)$ are present cases.

Definition 9353.12 (Optional classifier)**.** *The optional classifier is the branch-respecting relation*

$$\mathsf{OptionClassifier}_{A,\sim_A}(h, k) :\Longleftrightarrow \big(\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})\big)$$

$$\vee\ \exists a, b.\ A(a) \wedge A(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge a \sim_A b.$$

Definition 9353.13 (Optional constructors)**.** *The optional constructors are*

$$\mathsf{none}^{\uparrow} := \mathsf{e}, \qquad \mathsf{some}^{\uparrow}(a) := \mathsf{E1}(a).$$

Theorem 9353.14 (Present payload determinacy)**.** *If*

$$\mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(h, \mathsf{E1}(b)),$$

then $\mathsf{hsame}(a, b)$.

Proof. By symmetry and transitivity for hsame, the two assumptions give $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(b))$. Constructor inversion for $\mathsf{E1}$ gives $\mathsf{hsame}(a, b)$. □

Theorem 9353.15 (Absent and present separation)**.** *The optional classifier never identifies an absent endpoint with a visible present endpoint.*

Proof. Unfold Definition 9353.12. In the absent–absent case, the second history is hsame to e, so being also hsame to $\mathsf{E1}(b)$ would contradict empty–extension no-confusion. In the present–present case, the first history is hsame to some $\mathsf{E1}(a)$, so being also hsame to e gives the same contradiction. □

Corollary 9353.16 (Optional finite data admission)**.** *If the source carrier A has a $\mathsf{NameCert}(A, \sim_A)$ and the optional carrier, classifier, constructor separation, payload determinacy, carrier transport, and ledger policy are displayed, then $\mathsf{Option}^{\uparrow}(A)$ is admitted as a finite data interface.*

9353.4 Tagged Sum Interface

Definition 9353.17 (Sum carrier)**.** *Let $L, R : \mathsf{Hist} \to \mathsf{Prop}$ be admitted source carriers with classifiers $\sim_L$ and $\sim_R$. The sum carrier is*

$$\mathsf{SumCarrier}_{L,R}(h) :\Longleftrightarrow \exists l.\ L(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))$$

$$\vee\ \exists r.\ R(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)).$$

Definition 9353.18 (Sum classifier)**.** *The sum classifier relates left endpoints through the left source classifier and right endpoints through the right source classifier:*

$$\mathsf{SumClassifier}_{L,R}(h, k) :\Longleftrightarrow \exists l, l'.\ L(l) \wedge L(l') \wedge \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge \mathsf{hsame}(k, \mathsf{E0}(l')) \wedge l \sim_L l'$$

$$\vee\ \exists r, r'.\ R(r) \wedge R(r') \wedge \mathsf{hsame}(h, \mathsf{E1}(r)) \wedge \mathsf{hsame}(k, \mathsf{E1}(r')) \wedge r \sim_R r'.$$

Definition 9353.19 (Sum constructors)**.** *The public sum constructors are*

$$\mathsf{inl}^{\uparrow}(l) := \mathsf{E0}(l), \qquad \mathsf{inr}^{\uparrow}(r) := \mathsf{E1}(r).$$

Theorem 9353.20 (Sum branch separation)**.** *The sum classifier does not relate a left-tagged endpoint to a right-tagged endpoint:*

$$\neg\, \mathsf{SumClassifier}_{L,R}(\mathsf{E0}(l), \mathsf{E1}(r)).$$

Proof. Unfold Definition 9353.18. The left–left branch would require the right endpoint $\mathsf{E1}(r)$ to be hsame to a left-tagged history. The right–right branch would require the left endpoint $\mathsf{E0}(l)$ to be hsame to a right-tagged history. Both requirements contradict history no-confusion for the two extension constructors. □

Corollary 9353.21 (Sum finite data admission)**.** *If the two sources have their own name certificates and the displayed sum carrier, branch classifier, constructor stability, branch separation, classifier stability, and ledger rows are supplied, then* $\mathsf{Sum}^{\uparrow}(L, R)$ *is admitted.*

9353.5 Ordered Product Interface

Definition 9353.22 (Product pair representation)**.** *For admitted source carriers L and R, the representation*

$$\mathsf{ProdPairRep}_{L,R}(h, l, r)$$

records that h is the displayed ordered-pair history assembled from the left component l and right component r through the finite continuation route, with $L(l)$ and $R(r)$.

Definition 9353.23 (Product carrier and ledger)**.** *The product carrier is*

$$\mathsf{ProdCarrier}_{L,R}(h) :\Longleftrightarrow \exists l, r.\ \mathsf{ProdPairRep}_{L,R}(h, l, r).$$

The product ledger has the same witness shape:

$$\mathsf{ProdLedger}_{L,R}(h) :\Longleftrightarrow \exists l, r.\ \mathsf{ProdPairRep}_{L,R}(h, l, r).$$

The carrier admits the visible history, while the ledger keeps the component witnesses and continuation route available to public consumers.

Definition 9353.24 (Product componentwise classifier)**.** *The product classifier is*

$$\mathsf{ProdClassifier}_{L,R,\sim_L,\sim_R}(h, k) :\Longleftrightarrow \exists l, r, l', r'.\ \mathsf{ProdPairRep}_{L,R}(h, l, r) \wedge \mathsf{ProdPairRep}_{L,R}(k, l', r')$$
$$\wedge\ l \sim_L l' \wedge\ r \sim_R r'.$$

Theorem 9353.25 (Product carrier exposes the displayed ledger)**.** *If* $\mathsf{ProdCarrier}_{L,R}(h)$*, then* $\mathsf{ProdLedger}_{L,R}(h)$*.*

Proof. Both predicates have the same displayed witness shape in Definition 9353.23. A carrier witness therefore serves as the ledger witness. □

Theorem 9353.26 (Product classification exposes component ledgers)**.** *If*

$$\mathsf{ProdClassifier}_{L,R,\sim_L,\sim_R}(h,k),$$

then

$$\mathsf{ProdLedger}_{L,R}(h) \wedge \mathsf{ProdLedger}_{L,R}(k).$$

Proof. Unfold Definition 9353.24. The classifier witness contains displayed pair representations for both endpoints. Each representation is exactly a ledger witness by Definition 9353.23. □

Principle 9353.27 (Product coherence is part of the interface)**.** *Raw continuation is not by itself a product classifier. A product interface must carry coherence data ensuring that ordered components are recovered through the displayed pair representation, not by an untracked host projection.*

Corollary 9353.28 (Product finite data admission)**.** *If both source carriers have name certificates and the product pair representation, coherence data, componentwise classifier, displayed ledger, carrier transport, and stability rows are supplied, then* $\mathsf{Prod}^{\uparrow}(L,R)$ *is admitted.*

9353.6 Finite List Interface

Definition 9353.29 (List spine)**.** *For an admitted source carrier* $A : \mathsf{Hist} \to \mathsf{Prop}$*, the list spine carrier* Spine_A *is generated by*

$$\mathsf{nil}$$

and by

$$\mathsf{cons}(a, xs)$$

whenever $A(a)$ *and* $xs : \mathsf{Spine}_A$*.*

Definition 9353.30 (List spine representation)**.** *The relation*

$$\mathsf{ListSpineRep}_A(h, xs)$$

records that the public history h *displays the finite spine* xs*.*

Definition 9353.31 (List carrier and ledger)**.** *The list carrier is*

$$\mathsf{ListCarrier}_A(h) :\Longleftrightarrow \exists xs : \mathsf{Spine}_A.\ \mathsf{ListSpineRep}_A(h, xs).$$

The list ledger has the same displayed witness shape:

$$\mathsf{ListLedger}_A(h) :\Longleftrightarrow \exists xs : \mathsf{Spine}_A.\ \mathsf{ListSpineRep}_A(h, xs).$$

The ledger records finite constructor shape, length information, and payload provenance through the source classifier.

Definition 9353.32 (List classifier)**.** *The public list classifier is*

$$\mathsf{ListClassifier}_{A,\sim_A}(h,k) :\Longleftrightarrow \exists xs, ys.\ \mathsf{ListSpineRep}_A(h, xs) \wedge \mathsf{ListSpineRep}_A(k, ys)$$

$$\wedge\ \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Theorem 9353.33 (List nil inversion)**.** *If*

$$\mathsf{ListSpineRep}_A(h, \mathsf{nil}), \qquad \mathsf{ListSpineRep}_A(k, ys),$$

and $\mathsf{ListClassifier}_{A,\sim_A}(h, k)$*, then* $ys = \mathsf{nil}$*.*

Proof. Unfold Definition 9353.32. The represented nil spine can classify only against a displayed spine whose finite classifier matches nil. The cons case would require a nil–cons classifier witness, contradicting constructor no-confusion for finite spines. □

Theorem 9353.34 (List displayed exactness)**.** *If*

$$\mathsf{ListSpineRep}_A(h, xs), \qquad \mathsf{ListSpineRep}_A(k, ys),$$

then

$$\mathsf{ListClassifier}_{A,\sim_A}(h, k) \Longleftrightarrow \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Proof. The reverse direction repacks the two displayed spine representations and the finite-spine classifier witness into Definition 9353.32. The forward direction reads the public classifier through the displayed representation coherence, which leaves exactly the finite-spine classifier between xs and ys. □

Corollary 9353.35 (List finite data admission)**.** *If the source carrier has a name certificate and the list spine representation, spine coherence, public classifier, displayed ledger, constructor inversion, length readback, and stability rows are supplied, then* $\mathsf{List}^{\uparrow}(A)$ *is admitted.*

9353.7 Finite Data Layer

Theorem 9353.36 (Finite data admission theorem)**.** *For each of*

$$\mathsf{Bool}^{\uparrow}, \qquad \mathsf{Option}^{\uparrow}, \qquad \mathsf{Sum}^{\uparrow}, \qquad \mathsf{Prod}^{\uparrow}, \qquad \mathsf{List}^{\uparrow},$$

admission at the finite data layer follows from the displayed carrier predicate over Hist*, classifier,* StabilityCert *rows, constructor stability, case-analysis or inversion rows, ledger for hidden data, and* NameCert *or* SemanticNameCert *row. None of the five interfaces enters as a BEDC kernel input.*

Proof. Apply Definition 9353.2 to each displayed interface. Boolean admission is Definition 9353.10. Optional admission is Definition 9353.16. Sum admission is Definition 9353.21. Product admission is Definition 9353.28. List admission is Definition 9353.35. Each corollary uses only histories, history sameness, continuation and package provenance where needed, ledger exposure, stability, and name certification. □

Principle 9353.37 (Arithmetic boundary)**.** *The next arithmetic-facing layer may use* $\mathsf{Bool}^{\uparrow}$ *for binary observations,* $\mathsf{Option}^{\uparrow}$ *for presence and absence,* $\mathsf{Sum}^{\uparrow}$ *for tagged case analysis,* $\mathsf{Prod}^{\uparrow}$ *for ordered pairs with displayed component ledgers, and* $\mathsf{List}^{\uparrow}$ *for finite windows. It may not treat any of these interfaces as a host truth type, host option type, host coproduct, host tuple, or host sequence.*

Chapter 9354

Arithmetic Interface Spine

The finite data layer licenses Boolean tags, optional endpoints, sums, products, and finite spines. Arithmetic is the next certificate-facing surface. It does not add host natural numbers, host integer pairs, host rational quotients, or primitive arithmetic operations to the BEDC kernel. The arithmetic layer reads only the generated history spine, continuation, finite data interfaces, stability fields, and naming ledgers:

$$(\mathsf{Hist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{NameCert}, \mathsf{Bool}^{\uparrow}, \mathsf{Prod}^{\uparrow}, \mathsf{List}^{\uparrow}) \longrightarrow (\mathsf{Nat}^{\uparrow}, \mathsf{Add}^{\uparrow}, \mathsf{Mul}^{\uparrow}, \mathsf{Order}^{\uparrow}, \mathsf{Int}^{\uparrow}, \mathsf{Rat}^{\uparrow}).$$

This chapter records the interface spine for that passage. Its role is to state what each arithmetic name must display before it can be consumed by later algebra, real, and number-theoretic packets.

9354.1 Unary Natural Histories

Definition 9354.1 (Unary history judgment)**.** *The unary judgment*

$$\mathsf{Unary} : \mathsf{Hist} \to \mathsf{Prop}$$

is the closed generated predicate with base row

$$\mathsf{Unary}(\mathsf{e})$$

and one-extension row

$$\mathsf{Unary}(h) \Longrightarrow \mathsf{Unary}(\mathsf{E1}(h)).$$

There is no generation row for $\mathsf{Unary}(\mathsf{E0}(h))$.

Definition 9354.2 (Natural-number carrier and classifier)**.** *The natural-number carrier is*

$$\mathsf{NatCarrier}(h) :\Longleftrightarrow \mathsf{Unary}(h).$$

Its classifier is history sameness restricted to unary histories:

$$h \sim_{\mathsf{Nat}} k :\Longleftrightarrow \mathsf{Unary}(h) \wedge \mathsf{Unary}(k) \wedge \mathsf{hsame}(h, k).$$

The zero and successor rows are

$$0_{\mathsf{Nat}} := \mathsf{e}, \qquad \mathsf{succ}(h) := \mathsf{E1}(h).$$

Theorem 9354.3 (Unary closure and zero separation)**.** *The carrier* $\mathsf{NatCarrier}$ *contains* 0_{Nat}*, is closed under* succ*, and contains no* $\mathsf{E0}$*-headed history.*

Proof. The first two claims are exactly the two generation rows in Definition 9354.1. The last claim follows from closed generation: every nonempty unary history is produced by the $\mathsf{E1}$-extension row, and no displayed rule has an $\mathsf{E0}$-headed conclusion. □

Theorem 9354.4 (Unary induction)**.** *Let* $P : \mathsf{Hist} \to \mathsf{Prop}$*. If* $P(\mathsf{e})$ *and*

$$\mathsf{Unary}(h) \wedge P(h) \Longrightarrow P(\mathsf{E1}(h)),$$

then

$$\mathsf{Unary}(h) \Longrightarrow P(h)$$

for every history h*.*

Proof. Induct on the displayed derivation of $\mathsf{Unary}(h)$. The base derivation uses $P(\mathsf{e})$. The one-extension derivation applies the induction hypothesis to the tail and then applies the displayed step premise. □

Definition 9354.5 (Natural-number naming certificate)**.** *The* $\mathsf{Nat}^{\uparrow}$ *certificate has source* Unary*, pattern repeated* $\mathsf{E1}$*-extension from* e*, classifier* $\sim_{\mathsf{Nat}}$*, stability given by base, successor closure, unary induction, and* $\mathsf{E0}$ *exclusion, and ledger equal to the originating unary history behind each public counting token.*

Corollary 9354.6 (NatUp admission)**.** *The interface* $\mathsf{Nat}^{\uparrow}$ *is admitted by the certificate of Definition 9354.5. It is a unary-history interface, not a kernel natural-number primitive.*

9354.2 Unary Continuation Addition

Definition 9354.7 (Unary continuation and additive relation)**.** *The unary continuation predicate is*

$$\mathsf{UnaryCont}(h, k, r) :\Longleftrightarrow \mathsf{Unary}(h) \wedge \mathsf{Unary}(k) \wedge \mathsf{Cont}(h, k, r).$$

The additive relation is

$$\mathsf{AddRel}(h, k, r) :\Longleftrightarrow \mathsf{Unary}(h) \wedge \mathsf{Unary}(k) \wedge \mathsf{Unary}(r) \wedge \mathsf{Cont}(h, k, r).$$

It is the relation read publicly as $h + k \simeq r$*.*

Theorem 9354.8 (Addition closure)**.** *If* $\mathsf{Unary}(h)$*,* $\mathsf{Unary}(k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{Unary}(r)$*.*

Proof. Induct on the displayed continuation derivation. The empty continuation returns the right input, hence a unary result. The $\mathsf{E1}$ continuation step uses the induction hypothesis for the tail result and then applies unary closure under $\mathsf{E1}$. The $\mathsf{E0}$ continuation branch is incompatible with the unary right input by Definition 9354.3. □

Theorem 9354.9 (Additive unit and associativity rows)**.** *For unary histories,* e *is a left and right unit for continuation up to* hsame*, and nested unary continuations associate up to* hsame*.*

Proof. The left unit is the empty continuation row. The right unit is the continuation unit theorem restricted to unary histories, with Definition 9354.8 ensuring that the result remains inside the unary carrier. Associativity is the L2 continuation associativity row restricted to the same unary carrier and transported through hsame. □

Theorem 9354.10 (Additive commutativity row)**.** *If* $\mathsf{Unary}(h)$*,* $\mathsf{Unary}(k)$*,*

$$\mathsf{Cont}(h, k, r), \qquad \mathsf{Cont}(k, h, r'),$$

then $\mathsf{hsame}(r, r')$*.*

Proof. Induct on the unary derivation of h. If $h = \mathsf{e}$, the two continuation unit rows classify r and r' with k. In the one-extension case, invert the continuation from $\mathsf{E1}(h_0)$ and the continuation into the $\mathsf{E1}(h_0)$ tail, apply the induction hypothesis to the exposed tail continuations, and rebuild the endpoint comparison by $\mathsf{E1}$ congruence. □

Definition 9354.11 (AddUp naming certificate)**.** *The* $\mathsf{Add}^{\uparrow}$ *certificate has source pairs of unary histories, pattern relational continuation, classifier* hsame *on generated continuation results, stability given by closure, units, determinacy, associativity, and commutativity, and ledger equal to the two source unary histories together with the displayed continuation witness.*

Corollary 9354.12 (AddUp admission)**.** *The interface* $\mathsf{Add}^{\uparrow}$ *is admitted by the certificate of Definition 9354.11. It is a continuation certificate over unary histories, not a primitive binary operation.*

9354.3 Unary Multiplication and Divisibility

Definition 9354.13 (Unary multiplication relation)**.** *The relation* $\mathsf{NatMul}(d, q, n)$ *is the closed repeated-addition relation with rows:*

$$\mathsf{NatMul}(d, \mathsf{e}, \mathsf{e})$$

for every unary d*, and*

$$\mathsf{NatMul}(d, q, n) \wedge \mathsf{Cont}(n, d, n') \Longrightarrow \mathsf{NatMul}(d, \mathsf{E1}(q), n').$$

The first input is the multiplicand, the second input is the unary multiplier ledger, and the third input is the generated product history.

Theorem 9354.14 (Multiplication closure and functionality)**.** *If* $\mathsf{NatMul}(d, q, n)$*, then* q *is unary. If* d *is unary and* $\mathsf{NatMul}(d, q, n)$*, then* n *is unary. For fixed unary* d *and* q*, any two unary results of* $\mathsf{NatMul}(d, q, -)$ *are* hsame*.*

Proof. The unary property of q follows by induction on the displayed NatMul derivation: the empty multiplier is unary, and the successor row preserves unary by one-extension. Result closure is the same induction, using the empty result in the base case and Definition 9354.8 in the successor case. For functionality, induct on the unary multiplier. The empty case compares two empty product endpoints. The successor case inverts both multiplication witnesses, applies the induction hypothesis to the preceding products, and uses continuation determinacy and hsame transport for the final additive step. □

Definition 9354.15 (Multiplicative unit and divisibility)**.** *The unary unit is*

$$1_{\mathsf{Nat}} := \mathsf{E1}(\mathsf{e}).$$

Divisibility is the explicit quotient-witness relation

$$\mathsf{NatDivides}(d, n) :\Longleftrightarrow \exists q.\ \mathsf{Unary}(q) \wedge \mathsf{NatMul}(d, q, n).$$

Theorem 9354.16 (Unit multiplication readback)**.** *If* $\mathsf{Unary}(q)$ *and* $\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q, n)$*, then* $\mathsf{hsame}(n, q)$*.*

Proof. Induct on the unary structure of q. The empty case is the empty multiplication row. In the one-extension case, invert the multiplication witness into the predecessor product and one continuation by $\mathsf{E1}(\mathsf{e})$. The induction hypothesis identifies the predecessor product with the predecessor multiplier, and the final continuation is exactly the successor endpoint up to hsame. □

Definition 9354.17 (MulUp naming certificate)**.** *The* $\mathsf{Mul}^{\uparrow}$ *certificate has source unary triples equipped with a* NatMul *witness, pattern repeated* $\mathsf{Add}^{\uparrow}$ *continuation, classifier* hsame *on product results with transported factor ledgers, stability given by empty-multiplier, successor, result closure, functionality, unit, associativity, commutativity, and distribution rows where supplied, and ledger equal to the multiplicand, multiplier, partial-product trace, and continuation evidence.*

Corollary 9354.18 (MulUp admission boundary)**.** *The local* $\mathsf{Mul}^{\uparrow}$ *interface is the repeated-addition certificate described in Definition 9354.17. Semiring and ordered-ring uses may consume this surface only after the advertised law rows are present; they are not supplied by a kernel multiplication primitive.*

9354.4 Unary Order

Definition 9354.19 (Unary prefix order)**.** *For unary histories, the prefix comparison is*

$$h \leq_{\mathsf{Nat}} k :\Longleftrightarrow \mathsf{Unary}(h) \wedge \mathsf{Unary}(k) \wedge \exists t.\ \mathsf{Unary}(t) \wedge \mathsf{Cont}(h, t, k).$$

The strict comparison additionally requires the displayed tail not to be empty:

$$h <_{\mathsf{Nat}} k :\Longleftrightarrow h \leq_{\mathsf{Nat}} k \text{ through a tail } t \text{ with } \neg\, \mathsf{hsame}(t, \mathsf{e}).$$

Theorem 9354.20 (Unary prefix order rows)**.** *The relation* $\leq_{\mathsf{Nat}}$ *is reflexive and transitive on unary histories, no history is a strict unary prefix of* e*, and any two unary histories are comparable by a displayed prefix row in one direction.*

Proof. Reflexivity uses the empty tail and continuation right unit. Transitivity composes the two displayed tails using continuation associativity and Definition 9354.8. A strict prefix into e would require a nonempty unary tail whose continuation result is empty; empty result inversion forces that tail to be empty, a contradiction. Totality is proved by unary induction: the empty history prefixes every unary history, and the one-extension cases reduce by constructor inversion to the tail comparison. □

Definition 9354.21 (OrderUp naming certificate)**.** *The* $\mathsf{Order}^{\uparrow}$ *certificate has source pairs of unary histories, pattern prefix-or-equality comparison, classifier transport by endpoint* hsame*, stability given by reflexivity, transitivity, total comparison, strict-prefix asymmetry and endpoint transport, and ledger equal to the two endpoints, the comparison branch, and the finite* Cont *prefix route.*

Corollary 9354.22 (OrderUp admission)**.** *The interface* $\mathsf{Order}^{\uparrow}$ *is admitted as a finite unary comparison packet. It does not export host order, quotient order, ordered semiring structure, or a classical trichotomy selector.*

9354.5 Signed Integers

Definition 9354.23 (Signed-pair carrier and classifier)**.** *The integer carrier is the signed unary-pair predicate*

$$\mathsf{IntCarrier}(a, b) :\Longleftrightarrow \mathsf{Unary}(a) \wedge \mathsf{Unary}(b),$$

read as the formal difference $a - b$. Its classifier is cross-additive balance:

$$(a, b) \sim_{\mathsf{Int}} (c, d) :\Longleftrightarrow \exists u, v.\ \mathsf{AddRel}(a, d, u) \wedge \mathsf{AddRel}(c, b, v) \wedge \mathsf{hsame}(u, v).$$

Theorem 9354.24 (Integer classifier elementary laws)**.** *The integer classifier is reflexive and symmetric wherever the required $\mathsf{Add}^{\uparrow}$ rows are displayed. Its transitive law is a full arithmetic obligation over additive associativity, commutativity, and cancellation on the unary continuation surface.*

Proof. Reflexivity compares $a+b$ with itself using an additive witness and hsame reflexivity. Symmetry reverses the displayed hsame witness in the cross-additive balance. Transitivity has the usual signed-pair shape: from $a + d = c + b$ and $c + f = e + d$, one composes additive rows, reorders by the commutative-associative $\mathsf{Add}^{\uparrow}$ certificate, and cancels the shared unary summand. The cancellation row is not a kernel rule; it is part of the L7 arithmetic certificate surface. □

Definition 9354.25 (IntUp naming certificate)**.** *The $\mathsf{Int}^{\uparrow}$ certificate has source signed unary pairs, pattern formal difference, classifier cross-additive balance, stability given by classifier equivalence, addition, negation, multiplication, and order transport where displayed, and ledger equal to the two unary histories together with normalization and cross-additive witnesses.*

Corollary 9354.26 (IntUp admission boundary)**.** *The interface $\mathsf{Int}^{\uparrow}$ is admitted as a signed-pair certificate over $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}$. It does not import host integers, quotient integer equality, or ordered-ring structure without the corresponding displayed stability rows.*

9354.6 Rational Histories

Definition 9354.27 (Positive unary denominator)**.** *A positive unary denominator is*

$$\mathsf{PositiveUnary}(d) :\Longleftrightarrow \mathsf{Unary}(d) \wedge \neg\, \mathsf{hsame}(d, \mathsf{e}).$$

Theorem 9354.28 (Positive denominator has a one-extension tail)**.** *If $\mathsf{PositiveUnary}(d)$, then there exists t such that*

$$\mathsf{Unary}(t) \wedge \mathsf{hsame}(d, \mathsf{E1}(t)).$$

Proof. Invert the unary derivation of d. The empty case contradicts $\neg\, \mathsf{hsame}(d, \mathsf{e})$. The only remaining generated case is an E1-headed unary history, which supplies the displayed tail. □

Definition 9354.29 (Rational carrier and normalized classifier)**.** *The rational carrier consists of an $\mathsf{Int}^{\uparrow}$ numerator and a positive unary denominator:*

$$\mathsf{RatCarrier}(n, d) :\Longleftrightarrow \mathsf{IntCarrier}(n) \wedge \mathsf{PositiveUnary}(d).$$

The rational classifier is read through displayed normalization witnesses:

$$(n, d) \sim_{\mathsf{Rat}} (m, e)$$

only when there are normalized representatives (n', d') and (m', e') with displayed RatNorm ledgers, classified integer numerators, and classified unary denominators. It is not primitive quotient equality on unnormalized pairs.

Definition 9354.30 (RatUp naming certificate)**.** *The* $\mathsf{Rat}^{\uparrow}$ *certificate has source an* $\mathsf{Int}^{\uparrow}$ *numerator and a positive* $\mathsf{Nat}^{\uparrow}$ *denominator, pattern normalized rational representative, classifier normalized structural comparison, stability given by normalization uniqueness, arithmetic closure, denominator positivity, and order compatibility where supplied, and ledger equal to numerator history, denominator history, normalization trace, and coprimality evidence.*

Corollary 9354.31 (RatUp admission boundary)**.** *The interface* $\mathsf{Rat}^{\uparrow}$ *is admitted as a positive-denominator normalized history certificate. It does not import host rational numbers, quotient machinery, field order, or ordered-field structure unless those rows are displayed in the certificate.*

9354.7 Arithmetic Layer Boundary

Theorem 9354.32 (Arithmetic interface admission theorem)**.** *If the displayed certificates*

$$\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}},\quad \mathsf{NameCert}_{\mathsf{Add}^{\uparrow}},\quad \mathsf{NameCert}_{\mathsf{Mul}^{\uparrow}},\quad \mathsf{NameCert}_{\mathsf{Order}^{\uparrow}},\quad \mathsf{NameCert}_{\mathsf{Int}^{\uparrow}},\quad \mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$$

provide their advertised carrier, classifier, stability, operation congruence, order, normalization, and ledger rows, then

$$\mathsf{Nat}^{\uparrow},\quad \mathsf{Add}^{\uparrow},\quad \mathsf{Mul}^{\uparrow},\quad \mathsf{Order}^{\uparrow},\quad \mathsf{Int}^{\uparrow},\quad \mathsf{Rat}^{\uparrow}$$

are admitted arithmetic interfaces.

Proof. $\mathsf{Nat}^{\uparrow}$ is licensed by unary histories and their induction spine. $\mathsf{Add}^{\uparrow}$ is licensed by unary continuation. $\mathsf{Mul}^{\uparrow}$ is licensed by repeated $\mathsf{Add}^{\uparrow}$ continuation. $\mathsf{Order}^{\uparrow}$ is licensed by finite unary prefix comparison. $\mathsf{Int}^{\uparrow}$ is licensed by signed unary pairs and cross-additive balance. $\mathsf{Rat}^{\uparrow}$ is licensed by signed numerators, positive unary denominators, and normalized comparison. Each case uses a $\mathsf{NameCert}$ surface with an explicit ledger, so none enters as a kernel primitive. □

Principle 9354.33 (Algebra boundary)**.** *Algebraic consumers may read arithmetic through these interfaces only as displayed certificate rows: unary histories, continuation ledgers, repeated-addition traces, prefix-comparison ledgers, signed-pair balances, and rational normalization traces. They may not replace those rows by host natural numbers, host integers, host rationals, quotient equality, or ambient ordered-field facts.*

Concretized in §28: The unary-history NatUp chapter realizes the first arithmetic-interface carrier of this spine.

Chapter 9355

Algebra Interface Spine

The arithmetic spine licenses unary counting, continuation addition, repeated addition, prefix comparison, signed pairs, and rational histories. The algebra layer reads those objects only through certificates. It does not insert host algebraic structures into the kernel. The layer indexed in section 9467.10 is the certificate chain

$$\mathsf{Magma}^{\uparrow} \to \mathsf{Semigroup}^{\uparrow} \to \mathsf{Monoid}^{\uparrow} \to \mathsf{Group}^{\uparrow} \to \mathsf{AbGroup}^{\uparrow} \to \mathsf{Ring}^{\uparrow} \to \mathsf{CommRing}^{\uparrow} \to \mathsf{Field}^{\uparrow} \to \mathsf{Module}^{\uparrow} \to \mathsf{VecSpa}$$

Each step keeps a carrier and classifier visible, adds only the operation or law rows named by that step, and records expression provenance in a ledger.

9355.1 Algebraic Certificate Template

Definition 9355.1 (Algebraic carrier)**.** *An algebraic carrier is a carrier-classifier pair*

$$(C, \sim_C)$$

where

$$C : \mathsf{Hist} \to \mathsf{Prop}$$

is the admitted carrier predicate and

$$\sim_C : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop}$$

is the public classifier. The pair is usable by the algebra layer only after a displayed $\mathsf{NameCert}(C, \sim_C)$ *supplies carrier transport and classifier laws.*

Definition 9355.2 (Stable operations)**.** *A unary operation f on $(C, \sim_C)$ is stable when it supplies carrier closure, classifier congruence, and ledger rows:*

$$C(x) \Longrightarrow C(f(x)), \qquad x \sim_C y \Longrightarrow f(x) \sim_C f(y).$$

A binary operation μ is stable when it supplies

$$C(x) \wedge C(y) \Longrightarrow C(\mu(x, y)),$$

and

$$x \sim_C x' \wedge y \sim_C y' \Longrightarrow \mu(x, y) \sim_C \mu(x', y').$$

The operation ledger records the expression tree, the source histories, and the normalization or replay route hidden by the public operation symbol.

Definition 9355.3 (Algebraic interface certificate)**.** *An algebraic interface certificate is the packet*

$$\mathsf{AlgCert} = (C, \sim_C, \mathsf{Ops}, \mathsf{Laws}, \mathsf{Cong}, \mathsf{Ledger}, \mathsf{NameCert}).$$

The operation field records the displayed operation rows. The law field records the algebraic equations available at classifier resolution. The congruence field records transport of every operation through the classifier. The ledger field records finite expression provenance and normalization routes.

Principle 9355.4 (Algebra is certificate-facing)**.** *Every L8 interface is admitted only through an algebraic interface certificate. A familiar notation such as multiplication, inverse, scalar action, or vector addition is therefore a public name for displayed carrier, classifier, operation, law, congruence, and ledger rows.*

Concretized in §3341: The algebraic expression ledger realizes the operation-provenance rows of the algebra interface spine as finite NameCert data.

9355.2 Magma Through Monoid

Definition 9355.5 (Magma interface)**.** *A* $\mathsf{Magma}^{\uparrow}$ *interface is an admitted carrier-classifier pair* $(C, \sim_C)$ *with one stable binary operation* μ*. Its certificate records closure of* μ*, binary congruence of* μ*, and an operation ledger. It does not contain associativity, identity, inverse, or commutativity rows.*

Theorem 9355.6 (Magma operation descends to classifier classes)**.** *If* μ *satisfies binary congruence, then replacing either input by a classified input does not change the output classifier class.*

Proof. Given $x \sim_C x'$ and $y \sim_C y'$, binary congruence gives

$$\mu(x, y) \sim_C \mu(x', y').$$

This is exactly the descent condition for the public operation read. □

Definition 9355.7 (Semigroup interface)**.** *A* $\mathsf{Semigroup}^{\uparrow}$ *interface is a* $\mathsf{Magma}^{\uparrow}$ *interface with associativity:*

$$C(x) \wedge C(y) \wedge C(z) \Longrightarrow \mu(\mu(x, y), z) \sim_C \mu(x, \mu(y, z)).$$

Theorem 9355.8 (Fourfold rebracketing)**.** *In a* $\mathsf{Semigroup}^{\uparrow}$ *interface,*

$$\mu(\mu(\mu(a, b), c), d) \sim_C \mu(a, \mu(b, \mu(c, d))).$$

Proof. Apply associativity to the first three factors, then use binary congruence to move the rebracketing inside the outer product. Apply associativity once more to move the final factor into the right-nested expression. □

Definition 9355.9 (Monoid interface)**.** *A* $\mathsf{Monoid}^{\uparrow}$ *interface is a* $\mathsf{Semigroup}^{\uparrow}$ *interface with a displayed unit* e_C *satisfying*

$$C(e_C), \qquad \mu(e_C, x) \sim_C x, \qquad \mu(x, e_C) \sim_C x.$$

Theorem 9355.10 (Unit uniqueness)**.** *If* e *and* u *are both two-sided units for the same monoid operation, then* $e \sim_C u$*.*

Proof. Since u is a right unit, $\mu(e, u) \sim_C e$. Since e is a left unit, $\mu(e, u) \sim_C u$. Symmetry and transitivity of the classifier give $e \sim_C u$. □

Corollary 9355.11 (Unit normalization)**.** *Finite monoid expressions containing unit leaves normalize, at classifier resolution, to expressions in which those unit leaves have been removed.*

Proof. Induct on the expression tree. At a node $\mu(e_C, x)$ or $\mu(x, e_C)$, use the corresponding unit law. Binary congruence transports the substitution term through the enclosing expression contexts. □

9355.3 Groups

Definition 9355.12 (Group interface)**.** *A* $\mathsf{Group}^{\uparrow}$ *interface is a* $\mathsf{Monoid}^{\uparrow}$ *interface with a stable inverse operation* ι *satisfying*

$$C(x) \Longrightarrow C(\iota(x)), \qquad x \sim_C y \Longrightarrow \iota(x) \sim_C \iota(y),$$

and

$$\mu(\iota(x), x) \sim_C e_C, \qquad \mu(x, \iota(x)) \sim_C e_C.$$

Theorem 9355.13 (Group cancellation)**.** *In a* $\mathsf{Group}^{\uparrow}$ *interface, left and right multiplication by the same carried element are cancellable at classifier resolution.*

Proof. For left cancellation, assume $\mu(a, b) \sim_C \mu(a, c)$. Multiply both sides on the left by $\iota(a)$ and use binary congruence. Associativity rewrites the two sides as

$$\mu(\mu(\iota(a), a), b) \sim_C \mu(\mu(\iota(a), a), c).$$

The left inverse law turns the inner products into e_C, and the left unit law gives $b \sim_C c$. Right cancellation is the symmetric argument using right multiplication by $\iota(a)$. □

Theorem 9355.14 (Inverse uniqueness)**.** *If*

$$\mu(y, x) \sim_C e_C \qquad \textit{and} \qquad \mu(x, z) \sim_C e_C,$$

then $y \sim_C z$.

Proof. Use the chain

$$y \sim_C \mu(y, e_C) \sim_C \mu(y, \mu(x, z)) \sim_C \mu(\mu(y, x), z) \sim_C \mu(e_C, z) \sim_C z.$$

Each step is a unit law, associativity, the displayed inverse witness, or classifier transport through the operation. □

Definition 9355.15 (Abelian group interface)**.** *An* $\mathsf{AbGroup}^{\uparrow}$ *interface is a* $\mathsf{Group}^{\uparrow}$ *interface with commutativity:*

$$\mu(x, y) \sim_C \mu(y, x).$$

Theorem 9355.16 (Balanced cancellation)**.** *In an* $\mathsf{AbGroup}^{\uparrow}$ *interface, if*

$$\mu(a, b) \sim_C \mu(c, a),$$

then $b \sim_C c$.

Proof. Commutativity changes the right side to $\mu(a, c)$. Left cancellation in Definition 9355.13 then gives $b \sim_C c$. □

9355.4 Rings and Fields

Definition 9355.17 (Ring interface)**.** *A* $\mathsf{Ring}^{\uparrow}$ *interface is one carrier-classifier pair* $(C, \sim_C)$ *equipped with an additive* $\mathsf{AbGroup}^{\uparrow}$ *structure* $(+, 0, -)$*, a multiplicative* $\mathsf{Monoid}^{\uparrow}$ *structure* $(\cdot, 1)$*, and left and right distributivity:*

$$x \cdot (y + z) \sim_C x \cdot y + x \cdot z,$$

$$(x + y) \cdot z \sim_C x \cdot z + y \cdot z.$$

Its ledger records expression trees containing $0, 1, +, -, \cdot$ *together with the distributivity and sign-propagation rows used by normalization.*

Theorem 9355.18 (Zero multiplication)**.** *In a* $\mathsf{Ring}^{\uparrow}$ *interface,*

$$0 \cdot x \sim_C 0, \qquad x \cdot 0 \sim_C 0.$$

Proof. For the left zero law, use $0 + 0 \sim_C 0$ and multiply by x on the right. Right distributivity gives

$$(0 + 0) \cdot x \sim_C 0 \cdot x + 0 \cdot x.$$

The left side is classified with $0 \cdot x$, so additive cancellation in the underlying $\mathsf{AbGroup}^{\uparrow}$ gives $0 \cdot x \sim_C 0$. The right zero law is the same argument with left distributivity. □

Theorem 9355.19 (Negative products)**.** *In a* $\mathsf{Ring}^{\uparrow}$ *interface,*

$$(-x) \cdot y \sim_C -(x \cdot y), \qquad x \cdot (-y) \sim_C -(x \cdot y).$$

Proof. For the first identity, distribute $(-x + x) \cdot y$. The term $-x + x$ classifies with 0, and zero multiplication gives the zero endpoint. Thus $(-x) \cdot y + x \cdot y \sim_C 0$, so $(-x) \cdot y$ is an additive inverse of $x \cdot y$. Inverse uniqueness in the additive group gives the claim. The second identity is identical on the other side. □

Definition 9355.20 (Commutative ring interface)**.** *A* $\mathsf{CommRing}^{\uparrow}$ *interface is a* $\mathsf{Ring}^{\uparrow}$ *interface whose multiplication is commutative:*

$$x \cdot y \sim_C y \cdot x.$$

Theorem 9355.21 (Distributivity side transport)**.** *In a* $\mathsf{CommRing}^{\uparrow}$ *interface, either distributivity side implies the other.*

Proof. Assume left distributivity. Then

$$(x + y) \cdot z \sim_C z \cdot (x + y)$$

by multiplicative commutativity. Left distributivity gives $z \cdot x + z \cdot y$, and commutativity plus additive congruence transports that expression to $x \cdot z + y \cdot z$. The converse is symmetric. □

Definition 9355.22 (Field interface)**.** *A* $\mathsf{Field}^{\uparrow}$ *interface is a* $\mathsf{CommRing}^{\uparrow}$ *interface with an explicit apart-zero predicate* $\mathsf{NonZero}(x)$*, an inverse operation indexed by that witness,*

$$\mathrm{inv}(x, p) \qquad (p : \mathsf{NonZero}(x)),$$

inverse laws

$$x \cdot \mathrm{inv}(x, p) \sim_C 1, \qquad \mathrm{inv}(x, p) \cdot x \sim_C 1,$$

and nonzero transport along the classifier. The inverse is proof-indexed by apartness data, not by a host-level negated equality.

Theorem 9355.23 (Nonzero cancellation)**.** *If* $p : \mathsf{NonZero}(a)$ *and*

$$a \cdot x \sim_C a \cdot y,$$

then $x \sim_C y$*.*

Proof. Multiply both sides by $\mathrm{inv}(a, p)$ and use congruence. Associativity rewrites the two sides to

$$(\mathrm{inv}(a, p) \cdot a) \cdot x \sim_C (\mathrm{inv}(a, p) \cdot a) \cdot y.$$

The inverse law changes both left factors to 1, and the unit law gives $x \sim_C y$. □

Principle 9355.24 (Field boundary)**.** $\mathsf{Field}^{\uparrow}$ *is admitted only after the carrier, classifier, commutative-ring base, apart-zero predicate, proof-indexed inverse, inverse laws, transport, and ledger exactness rows are displayed. It is not a field primitive and it does not use excluded-middle reasoning to form inverses.*

9355.5 Modules and Vector Spaces

Definition 9355.25 (Module interface)**.** *Let* R *be a* $\mathsf{Ring}^{\uparrow}$ *scalar source and let* M *be an additive* $\mathsf{AbGroup}^{\uparrow}$ *carrier. A* $\mathsf{Module}^{\uparrow}$ *interface adds a stable scalar action*

$$R \times M \to M$$

with scalar-action congruence, scalar associativity,

$$(rs) \cdot m \sim_M r \cdot (s \cdot m),$$

distributivity over vector addition,

$$r \cdot (m + n) \sim_M r \cdot m + r \cdot n,$$

distributivity over scalar addition,

$$(r + s) \cdot m \sim_M r \cdot m + s \cdot m,$$

and scalar unit law $1 \cdot m \sim_M m$*.*

Theorem 9355.26 (Module zero and negative scalar laws)**.** *In a* $\mathsf{Module}^{\uparrow}$ *interface,*

$$0_R \cdot m \sim_M 0_M, \qquad (-r) \cdot m \sim_M -(r \cdot m).$$

Proof. For the zero scalar law, use $0_R + 0_R \sim_R 0_R$ and scalar distributivity over scalar addition:

$$(0_R + 0_R) \cdot m \sim_M 0_R \cdot m + 0_R \cdot m.$$

The left side is classified with $0_R \cdot m$, and additive cancellation in the module carrier gives $0_R \cdot m \sim_M 0_M$. For the negative scalar law, apply the same argument to $(-r+r) \cdot m$ and use inverse uniqueness in the additive group of M. □

Definition 9355.27 (Vector-space interface)**.** *A* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *interface is a* $\mathsf{Module}^{\uparrow}(F, V)$ *interface whose scalar source is a* $\mathsf{Field}^{\uparrow}$ *interface. It requires no further operation field; the scalar source is strengthened from ring to field.*

Theorem 9355.28 (Nonzero scalar cancellation)**.** *If* $p : \mathsf{NonZero}(a)$ *and*

$$a \cdot v \sim_V a \cdot w,$$

then $v \sim_V w$*.*

Proof. Act on both sides by $\mathrm{inv}(a, p)$. Scalar associativity changes the two sides to

$$(\mathrm{inv}(a, p)a) \cdot v \sim_V (\mathrm{inv}(a, p)a) \cdot w.$$

The field inverse law identifies the scalar factor with 1, and scalar unit gives $v \sim_V w$. □

9355.6 Algebra Layer Boundary

Theorem 9355.29 (Algebra interface admission theorem)**.** *If the interfaces*

$\mathsf{Magma}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpac}$

carry their advertised carrier, classifier, operation stability, congruence, law, ledger, and $\mathsf{NameCert}$ *rows, then the L8 algebra hierarchy is admitted. Each layer consumes only the rows exposed by the preceding layers and the additional rows named by its own certificate.*

Proof. $\mathsf{Magma}^{\uparrow}$ supplies one stable binary operation. $\mathsf{Semigroup}^{\uparrow}$ adds associativity. $\mathsf{Monoid}^{\uparrow}$ adds a unit. $\mathsf{Group}^{\uparrow}$ adds inverse rows. $\mathsf{AbGroup}^{\uparrow}$ adds commutativity. $\mathsf{Ring}^{\uparrow}$ combines an additive $\mathsf{AbGroup}^{\uparrow}$, a multiplicative $\mathsf{Monoid}^{\uparrow}$, and distributivity on the same carrier-classifier pair. $\mathsf{CommRing}^{\uparrow}$ adds multiplicative commutativity. $\mathsf{Field}^{\uparrow}$ adds apart-zero and proof-indexed inverse data. $\mathsf{Module}^{\uparrow}$ adds a scalar action over a ring and an additive abelian group. $\mathsf{VecSpace}^{\uparrow}$ specializes the scalar source to a field. All rows are certificate fields with ledgers, so no algebraic layer enters as a kernel primitive. □

Principle 9355.30 (Downstream algebra consumption)**.** *Later analytic, spectral, zeta-facing, and linear-algebraic packets may read L8 only through the displayed algebraic ledgers: expression trees, operation closure, congruence, normalization routes, inverse apartness witnesses, distributivity rows, and scalar-action rows. They may not replace these rows by host algebra, quotient equality, hidden inverses, or ambient vector-space facts.*

Chapter 9356

Order Set Type Interface Spine

The algebra spine exposes stable operations and their classifier-respecting laws. The next kernel-facing layer exposes stable comparison, membership, and transport. In the layer indexed by section 9467.11, the order–set–type surface is the certificate family

$$\mathsf{Preorder}^{\uparrow},\quad \mathsf{Poset}^{\uparrow},\quad \mathsf{TotalOrder}^{\uparrow},\quad \mathsf{Lattice}^{\uparrow},\quad \mathsf{SetLike}^{\uparrow},\quad \mathsf{TypeLike}^{\uparrow}.$$

The common rule is that no relation, membership predicate, subset relation, or typed family enters as an ambient object. Each is read through a carried source, a public classifier, stability rows, a ledger, and a local $\mathsf{NameCert}$.

Concretized in §8909: Finite membership classifier rows realize the order–set–type interface through displayed source, member, witness, stability, ledger, transport, replay, provenance, and NameCert rows.

9356.1 Relational Certificate Template

Definition 9356.1 (Relational interface)**.** *Let $(C, \sim_C)$ be an admitted carrier-classifier pair with*

$$C : \mathsf{Hist} \to \mathsf{Prop} \qquad \textit{and} \qquad \sim_C : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop}.$$

A relational interface over this carrier is a packet

$$\mathcal{R} = (C, \sim_C, R, \mathsf{Stab}_R, \mathsf{Cong}_R, \mathsf{Ledger}_R, \mathsf{NameCert}_R),$$

where $R : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop}$ is the displayed binary relation. The stability row records the law profile required by the interface name, the congruence row records classifier transport in both endpoints, and the ledger records the source of each comparison witness.

Definition 9356.2 (Relation congruence)**.** *The relation R respects the carrier classifier when*

$$C(x) \wedge C(y) \wedge C(x') \wedge C(y') \wedge x \sim_C x' \wedge y \sim_C y' \wedge R(x, y) \Longrightarrow R(x', y').$$

This row is part of the public relation interface, not an after-the-fact identification of representatives.

Principle 9356.3 (Relational primitive boundary)**.** *An L9 relation is admitted only through the displayed tuple*

$$C,\ \sim_C,\ R,\ \mathsf{Stab}_R,\ \mathsf{Cong}_R,\ \mathsf{Ledger}_R,\ \mathsf{NameCert}_R.$$

Order, elementhood, subset inclusion, and family transport are therefore licensed interface rows rather than kernel primitives.

9356.2 Preorders and Posets

Definition 9356.4 (Preorder interface)**.** *A* $\mathsf{Preorder}^{\uparrow}$ *interface is a relational interface* $(C, \sim_C, \leq_C)$ *with reflexivity and transitivity:*

$$C(x) \Longrightarrow x \leq_C x,$$

and

$$x \leq_C y \wedge y \leq_C z \Longrightarrow x \leq_C z.$$

It also carries relation congruence under $\sim_C$ *and a comparison ledger.*

Definition 9356.5 (Unary prefix preorder read)**.** *The concrete prefix read uses the unary-history carrier* $C(h) := \mathsf{Unary}(h)$*, the carrier-aware classifier*

$$h \sim_C k \Longleftrightarrow \mathsf{Unary}(h) \wedge \mathsf{Unary}(k) \wedge \mathsf{hsame}(h, k),$$

and the prefix relation

$$h \leq_C k \Longleftrightarrow \exists t \left(\mathsf{Unary}(t) \wedge \mathsf{Cont}(h, t, k)\right).$$

The comparison ledger records the tail t *and the continuation witness.*

Theorem 9356.6 (Prefix preorder laws)**.** *The unary prefix read supplies reflexivity, transitivity, and endpoint congruence for* $\mathsf{Preorder}^{\uparrow}$*.*

Proof. For reflexivity, take the empty unary tail and the right-unit continuation row. For transitivity, compose the two displayed continuation witnesses and use unary closure of the composed tail. For congruence, transport the source and endpoint histories along the two hsame rows in the carrier classifier; continuation stability supplies the transported comparison. □

Definition 9356.7 (Poset interface)**.** *A* $\mathsf{Poset}^{\uparrow}$ *interface is a* $\mathsf{Preorder}^{\uparrow}$ *interface with antisymmetry under the carrier classifier:*

$$x \leq_C y \wedge y \leq_C x \Longrightarrow x \sim_C y.$$

The existing concrete chapters also use the spelling $\mathsf{POSet}^{\uparrow}$ *for this partial-order surface; the present layer reads it through the public* $\mathsf{Poset}^{\uparrow}$ *macro.*

Theorem 9356.8 (Prefix antisymmetry)**.** *For the unary prefix carrier, opposing prefix comparisons classify the two endpoints:*

$$h \leq_C k \wedge k \leq_C h \Longrightarrow h \sim_C k.$$

Proof. The first comparison gives a unary tail t and a continuation $\mathsf{Cont}(h, t, k)$. The second gives a unary tail u and a continuation $\mathsf{Cont}(k, u, h)$. Composing the continuations yields a loop from h back to h through the combined tail. The append-empty inversion and right-unit uniqueness rows force the forward tail to be classified as empty, so the first comparison endpoint k is classified with h. This is exactly the carrier classifier conclusion $h \sim_C k$. □

Corollary 9356.9 (Poset exactness for prefix reads)**.** *On the unary prefix carrier,*

$$h \sim_C k \Longleftrightarrow (h \leq_C k \wedge k \leq_C h).$$

Proof. The forward direction transports reflexive prefix comparisons along the classifier row. The reverse direction is Definition 9356.8. □

9356.3 Total Orders and Lattices

Definition 9356.10 (Total-order interface)**.** *A* $\mathsf{TotalOrder}^{\uparrow}$ *interface is a* $\mathsf{Poset}^{\uparrow}$ *interface with explicit comparison data. For carried endpoints it supplies a displayed comparison branch, strict-to-apartness transport, classifier-stable comparison transport, and strict transitivity whenever the strict row is part of the packet. Its totality is data-bearing; it is not an excluded-middle assertion.*

Theorem 9356.11 (Unary comparison is certificate data)**.** *For unary prefix endpoints, a total-order read may be given by finite comparison data choosing one of the two prefix directions, with the equality branch read through the* $\mathsf{Poset}^{\uparrow}$ *exactness row.*

Proof. Unary histories are finite continuation spines. Compare the spines by structural inspection: the empty endpoint is a prefix of every unary endpoint, and equal leading unary extensions reduce the comparison to their tails. The selected branch is recorded as an explicit prefix witness. When both directions are present, Definition 9356.9 returns the classifier branch rather than a host equality. □

Definition 9356.12 (Lattice interface)**.** *A* $\mathsf{Lattice}^{\uparrow}$ *interface is a* $\mathsf{Poset}^{\uparrow}$ *interface with stable meet and join operations. For* $m = x \wedge y$*, the certificate records*

$$m \leq_C x, \qquad m \leq_C y,$$

and greatestness

$$z \leq_C x \wedge z \leq_C y \Longrightarrow z \leq_C m.$$

For $j = x \vee y$*, it records*

$$x \leq_C j, \qquad y \leq_C j,$$

and leastness

$$x \leq_C z \wedge y \leq_C z \Longrightarrow j \leq_C z.$$

Meet and join congruence, together with the bound-characterization ledger, complete the certificate.

Theorem 9356.13 (Meet and join uniqueness)**.** *In a* $\mathsf{Lattice}^{\uparrow}$ *interface, two displayed meets of the same pair are classified the same, and two displayed joins of the same pair are classified the same.*

Proof. If m and m' are both meets of x and y, then m is a lower bound, so greatestness of m' gives $m \leq_C m'$. Similarly m' is a lower bound, so greatestness of m gives $m' \leq_C m$. Poset antisymmetry gives $m \sim_C m'$. The join case is dual: upper-bound rows and leastness give the two comparisons, and antisymmetry gives the classifier conclusion. □

Theorem 9356.14 (Lattice laws from bounds)**.** *The idempotence, commutativity, and absorption laws for meet and join are classifier consequences of their bound-characterization rows.*

Proof. For idempotence, x is both a lower bound and an upper bound of the pair (x, x) by reflexivity. Greatestness and leastness supply the reverse comparisons against $x \wedge x$ and $x \vee x$, and antisymmetry gives the classifier equalities.

For commutativity, the lower-bound rows for $x \wedge y$ also make it a lower bound of (y, x), so greatestness gives $x \wedge y \leq_C y \wedge x$; the reverse comparison is identical with the operands exchanged. Joins are dual. For absorption, use the join upper-bound row $x \leq_C x \vee y$ to make x a lower bound of $(x, x \vee y)$; the meet lower-bound row supplies the opposite comparison. The second absorption law is dual. □

9356.4 Set-Like Membership

Definition 9356.15 (Set-like interface)**.** *For an element carrier $(E, \sim_E)$, a $\mathsf{SetLike}^{\uparrow}$ object is not an ambient set. It is a finite packet containing a generated membership predicate, membership stability under $\sim_E$, subset implication, restricted comprehension, an extensionality boundary, a ledger, and a local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ row.*

Definition 9356.16 (Membership stability and subset)**.** *Membership in a set-like object S is stable when*

$$x \sim_E y \wedge x \in_S \Longrightarrow y \in_S .$$

The subset relation is implication of membership:

$$S \subseteq T \Longleftrightarrow \prod_{x:E} (x \in_S \Longrightarrow x \in_T).$$

Both rows are read through the membership ledger; neither is a host powerset relation.

Definition 9356.17 (Restricted comprehension)**.** *A predicate family $P : E \to \mathsf{Prop}$ may define a set-like object only when it carries classifier stability,*

$$x \sim_E y \wedge P(x) \Longrightarrow P(y),$$

together with a predicate-generation ledger and a local name certificate. The comprehension row is therefore a bounded consumer of certified membership data.

Theorem 9356.18 (Subset preorder)**.** *For $\mathsf{SetLike}^{\uparrow}$ objects over the same element carrier, subset implication is reflexive and transitive.*

Proof. Reflexivity sends each membership witness $x \in_S$ to itself. For transitivity, compose the two displayed implication rows:

$$x \in_S \Longrightarrow x \in_T \Longrightarrow x \in_U .$$

The proof uses only the membership ledger and implication rows. □

Theorem 9356.19 (Set-like extensionality is a certificate field)**.** *If the extensionality boundary for S and T is available and*

$$S \subseteq T \qquad \text{and} \qquad T \subseteq S,$$

then S and T are classified as the same set-like object.

Proof. Apply the displayed extensionality boundary row to the two subset implications. The conclusion is the set-like classifier row recorded by the packet, not an ambient equality of host sets. □

Principle 9356.20 (Membership generation boundary)**.** *Every accepted membership read of a $\mathsf{SetLike}^{\uparrow}$ packet must expose the membership source, the classifier row, transport, replay, provenance, and the local name certificate. A downstream consumer may not replace this route by host elementhood, unrestricted separation, selected representatives, a powerset coordinate, or quotient equality of sets.*

9356.5 Type-Like Transport

Definition 9356.21 (Type-like interface)**.** *A* $\mathsf{TypeLike}^{\uparrow}$ *interface is a displayed family packet*

$$T = (B, F, Q, E, H, C, P, N).$$

Here B is the base classifier family, F is the indexed fiber-carrier row, Q is the family-compatible classifier, E is the endpoint-exactness ledger, H records componentwise hsame *transport, C records displayed* Cont *replay, P records provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *row.*

Definition 9356.22 (Type-like transport)**.** *The transport row of a* $\mathsf{TypeLike}^{\uparrow}$ *packet has the form*

$$A(x) \wedge x \sim_A y \Longrightarrow A(y),$$

or, in displayed-family form, it preserves the base classifier, the indexed fiber carrier, the family-compatible classifier, and the endpoint-exactness ledger before any sibling consumer may read the endpoint.

Definition 9356.23 (Map-like object)**.** *For type-like interfaces A and B, a map-like object is an operation interface that sends carried A endpoints to carried B endpoints and respects classifiers:*

$$x \sim_A x' \Longrightarrow f(x) \sim_B f(x').$$

The operation is therefore a classifier-stable transport surface, not a bare host function.

Theorem 9356.24 (Transport core)**.** *If $A^{\uparrow}$ is a* $\mathsf{TypeLike}^{\uparrow}$ *interface, $A(x)$ holds, and $x \sim_A y$, then $A(y)$ holds.*

Proof. This is the transport row of the type-like certificate. The row preserves the carrier through the classifier and records the movement in the ledger. □

Principle 9356.25 (Type-like host boundary)**.** $\mathsf{TypeLike}^{\uparrow}$ *is not a host universe. Its public content is the displayed carrier, classifier, transport, endpoint-exactness, replay, provenance, and name-certificate route. It does not provide quotient type equality, impredicative comprehension, selected representatives, or an external category of all families.*

9356.6 L9 Boundary

Theorem 9356.26 (Order set type admission theorem)**.** *If*

$$\mathsf{Preorder}^{\uparrow}, \quad \mathsf{Poset}^{\uparrow}, \quad \mathsf{TotalOrder}^{\uparrow}, \quad \mathsf{Lattice}^{\uparrow}, \quad \mathsf{SetLike}^{\uparrow}, \quad \mathsf{TypeLike}^{\uparrow}$$

carry their advertised carrier, classifier, stability, congruence, membership-generation, transport, extensionality, ledger, and $\mathsf{NameCert}$ *rows, then the L9 order–set–type layer is admitted.*

Proof. $\mathsf{Preorder}^{\uparrow}$ supplies a reflexive and transitive relation over a carried classifier. $\mathsf{Poset}^{\uparrow}$ adds antisymmetry under that classifier. $\mathsf{TotalOrder}^{\uparrow}$ adds explicit comparison data and constructive apartness transport. $\mathsf{Lattice}^{\uparrow}$ adds meet and join operations through their bound characterizations. $\mathsf{SetLike}^{\uparrow}$ supplies generated membership, subset as membership implication, restricted comprehension, and an extensionality boundary. $\mathsf{TypeLike}^{\uparrow}$ supplies carrier-plus-classifier-plus-transport data for displayed families. Since each row is part of a finite certified route, no order, membership, set, or type interface is imported as a kernel primitive or host-level object. □

Principle 9356.27 (Downstream L9 consumption)**.** *Analytic, zeta-facing, completion, spectral, and model-theoretic packets may consume L9 only through its displayed rows: comparison ledgers, classifier transport, membership-generation ledgers, subset implications, restricted comprehension routes, extensionality boundaries, and type-like transport. They may not replace these rows by classical trichotomy, unrestricted membership, ambient powersets, quotient type equality, or host universes.*

Chapter 9357

Countable Completion Interface Spine

The order–set–type spine permits membership, comparison, and transport to be read through certified carriers. The completion layer crosses the boundary from finite history objects to certified countable names. A single finite Hist value does not encode an infinite object; it can only expose a finite observation, a replay route, and the certificate rows authorizing that observation. The L10 surface therefore factors through

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{Topology}^{\uparrow},\quad \mathsf{Completion}^{\uparrow},$$

together with $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ where the metric side is consumed. Each object is a visible countable-name or modulus-bearing interface, never a host quotient, selected limit, or ambient complete space.

9357.1 Countable Names

Definition 9357.1 (Finite-history object)**.** *An object is finite-history presented when every public row is ultimately read through finitely many constructor, continuation, signature, package, and name-certificate observations over* Hist*. Such a row may name a finite observation of an infinite process, but it is not itself the completed process.*

Definition 9357.2 (Countable name over a carrier)**.** *Let* $(X, \sim_X)$ *be a certified carrier-classifier pair. An* X*-countable name is an observation map* $s : \mathsf{Nat}^{\uparrow} \to X$ *with rows*

$$\forall n : \mathsf{Nat}^{\uparrow},\ X(s(n)),$$

and

$$n \sim_{\mathsf{Nat}^{\uparrow}} m \Longrightarrow s(n) \sim_X s(m).$$

Two such names classify together when all displayed observations classify pointwise:

$$s \sim_{\mathsf{stream}} t \Longleftrightarrow \forall n : \mathsf{Nat}^{\uparrow},\ s(n) \sim_X t(n).$$

Definition 9357.3 (StreamName carrier and classifier)**.** *The* $\mathsf{StreamName}^{\uparrow}$ *carrier over* X *is*

$$\mathsf{StreamCarrier}_X(s) \Longleftrightarrow \forall n : \mathsf{Nat}^{\uparrow},\ X(s(n)).$$

Its classifier is

$$\mathsf{StreamSame}_X(s,t) \Longleftrightarrow \forall n : \mathsf{Nat}^{\uparrow},\ s(n) \sim_X t(n).$$

The public ledger stores finite windows, pointwise classifier rows, reindexing routes, replay, provenance, and the local name certificate.

Theorem 9357.4 (Pointwise stream classifier)**.** *If $\sim_X$ is an equivalence classifier, then* $\mathsf{StreamSame}_X$ *is an equivalence classifier on stream names.*

Proof. Reflexivity is pointwise reflexivity of $\sim_X$. Symmetry sends each row $s(n) \sim_X t(n)$ to $t(n) \sim_X s(n)$. Transitivity composes the pointwise rows $s(n) \sim_X t(n)$ and $t(n) \sim_X u(n)$ to obtain $s(n) \sim_X u(n)$. The stream ledger records the same unary observation index in every step. □

Principle 9357.5 (Countable-name boundary)**.** *An infinite object in L10 is a certified countable name together with pointwise stability. It is not a single history, a host sequence, a quotient class, or a selected global limit.*

9357.2 Dyadic Rational Core

Definition 9357.6 (Dyadic rational code)**.** *A* $\mathsf{DyadicRatCore}^{\uparrow}$ *code is a signed integer mantissa with unary exponent, displayed as*

$$(a, n) \qquad a : \mathsf{Int}^{\uparrow}, \quad n : \mathsf{Nat}^{\uparrow},$$

and read as the dyadic value $a \cdot 2^{-n}$. The carrier is inherited from the signed-pair and unary-exponent spines.

Definition 9357.7 (Common-exponent dyadic classifier)**.** *Two dyadic codes classify together when they agree after scaling to a common unary exponent:*

$$(a, n) \sim_{\mathsf{dyad}} (b, m)$$

when there is k with $k \geq n$ and $k \geq m$ such that

$$a \cdot 2^{k-n} \sim_{\mathsf{Int}^{\uparrow}} b \cdot 2^{k-m}.$$

The witness is a comparison and scaling ledger, not a quotient representative.

Theorem 9357.8 (Dyadic classifier laws)**.** *The common-exponent classifier is reflexive, symmetric, and transitive.*

Proof. For reflexivity, use the same exponent $k = n$ and the multiplicative unit row on $\mathsf{Int}^{\uparrow}$. Symmetry reuses the same common exponent and reverses the $\mathsf{Int}^{\uparrow}$ classifier row. For transitivity, combine two common-exponent witnesses by taking a displayed exponent above both, scaling both equalities to that exponent, and composing the resulting $\mathsf{Int}^{\uparrow}$ classifier rows. □

Definition 9357.9 (Dyadic distance data)**.** *The dyadic absolute value and distance rows are*

$$|q|_{\mathsf{dyad}} \qquad \textit{and} \qquad d_{\mathsf{dyad}}(q, r) := |q - r|_{\mathsf{dyad}}.$$

Their certificate records arithmetic stability, order transport, monotone radii 2^{-n}, and classifier compatibility of subtraction and absolute value.

Principle 9357.10 (Dyadic core boundary)**.** $\mathsf{DyadicRatCore}^{\uparrow}$ *is the finite arithmetic substrate for L10 tolerances. It uses common-exponent predicates and integer arithmetic rows, not quotient types or host rational equality.*

9357.3 Regular Sequences and Real Names

Definition 9357.11 (Regular dyadic sequence)**.** *A regular dyadic sequence is a stream* $x : \mathsf{Nat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow}$ *with regularity certificate*

$$\mathsf{Reg}(x) \iff \forall m\, n : \mathsf{Nat}^{\uparrow}, |x_m - x_n| \leq 2^{-m} + 2^{-n}.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *carrier is a* $\mathsf{StreamName}^{\uparrow}$ *over* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with this displayed Cauchy bound.*

Definition 9357.12 (Regular-sequence classifier)**.** *Two regular dyadic sequences classify together when every requested dyadic tolerance is met by a displayed tail comparison. Equivalently, in a fixed regular scale one may read rows of the form*

$$\forall n : \mathsf{Nat}^{\uparrow},\ |x_n - y_n| \leq 2^{-n},$$

with the ledger allowing reindexing, halving, and tail transport.

Theorem 9357.13 (Regular-sequence classifier laws)**.** *The regular-sequence classifier is reflexive, symmetric, and transitive when dyadic absolute value, order transport, and halving rows are available.*

Proof. Reflexivity uses $|x_n - x_n| = 0$ and $0 \leq 2^{-n}$. Symmetry uses absolute-value symmetry. For transitivity at tolerance ε, read the two comparison rows at $\varepsilon/2$ and apply the triangle inequality:

$$|x_n - z_n| \leq |x_n - y_n| + |y_n - z_n| \leq \varepsilon.$$

The halving row is explicit dyadic data, so no classical choice is used. □

Definition 9357.14 (Real carrier and classifier)**.** *A* $\mathsf{Real}^{\uparrow}$ *carrier is a sealed regular-sequence name*

$$r = \mathsf{Seal}(x, \mathsf{Reg}(x)).$$

Two sealed real names classify together when their exposed $\mathsf{RegSeqRat}^{\uparrow}$ *representatives classify:*

$$r \sim_{\mathsf{Real}^{\uparrow}} s \iff x \sim_{\mathsf{reg}} y.$$

The seal ledger controls which representative rows are visible to a consumer.

Definition 9357.15 (Constant real embedding)**.** *A certified dyadic or rational point* d *embeds as the sealed constant stream*

$$\iota(d) := \mathsf{Seal}(\lambda n.\, d, \mathsf{constant_regularity}).$$

On the history surface, the constant-real carrier is the row saying that the history is classified with an E_1*-seal of a certified rational endpoint.*

Theorem 9357.16 (Constant real seal boundary)**.** *The* E_1 *constant seal exposes exactly the carried rational endpoint, and an* E_0 *seal cannot expose a constant real endpoint.*

Proof. For an E_1 seal, constructor injectivity transports the rational carrier between the displayed endpoint and the endpoint named by the seal. For an E_0 seal, the assumed constant-real row would contain an hsame witness between an E_0 constructor and an E_1 constructor, contradicting constructor no-confusion. □

Theorem 9357.17 (Real classifier transfer)**.** *If the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier is an equivalence classifier, then the* $\mathsf{Real}^{\uparrow}$ *classifier is an equivalence classifier.*

Proof. Unseal a real name only through the displayed seal ledger. Reflexivity, symmetry, and transitivity are exactly the corresponding regular-sequence classifier rows transported through that ledger. □

Principle 9357.18 (Real primitive boundary)**.** $\mathsf{Real}^{\uparrow}$ *is a sealed regular Cauchy name. It is not a quotient set of sequences, not a Dedekind cut imported from a host universe, and not a completed scalar field until the relevant algebra, order, apartness, and operation-closure rows are supplied.*

9357.4 Completion

Definition 9357.19 (Regular Cauchy sequence with modulus)**.** *For a metric or regular-sequence carrier, a completion input is a sequence* $X : \mathsf{Nat}^{\uparrow} \to A$ *with an explicit modulus*

$$\mu : \mathbb{Q}^{\uparrow}_{>0} \to \mathsf{Nat}^{\uparrow}$$

such that, whenever $m, n \geq \mu(\varepsilon)$*, the distance between* X_m *and* X_n *lies inside the displayed* ε*-ball at zero.*

Definition 9357.20 (Diagonal representative)**.** *For a regular Cauchy sequence of regular dyadic names, the diagonal representative is*

$$L_n := X_{\mu(1/(n+1))}\big(\mu(1/(n+1)) + n + 1\big).$$

The ledger records the modulus request, the selected sequence member, the internal observation index, and the dyadic value returned by that observation.

Theorem 9357.21 (Diagonal regularity)**.** *The diagonal representative is regular when the constituent regularity rows, the Cauchy modulus, dyadic halving, and triangle inequality are available.*

Proof. Compare L_m and L_n by inserting a common observation scale. The difference splits into three displayed parts: movement inside the first regular sequence member, movement between the two sequence members controlled by the Cauchy modulus, and movement inside the second regular sequence member. The constituent regularity rows control the outer parts, the modulus controls the middle part, and the triangle inequality combines the dyadic budgets. □

Theorem 9357.22 (Completion limit and uniqueness)**.** *When the diagonal representative is sealed as a real name, it is a limit candidate for the regular Cauchy input; any other candidate satisfying the same modulus constraints classifies with it under the separated real classifier.*

Proof. The modulus brings X_n close to the selected member used in the diagonal representative, and the regularity of that member brings the selected observation close to the diagonal value. Combining these estimates gives the limit comparison. If L' is another candidate, compare L and L' through X_n:

$$d(L, L') \leq d(L, X_n) + d(X_n, L').$$

The two terms can be made smaller than any displayed tolerance, so the classifier identifies the two candidates. □

Principle 9357.23 (Completion boundary)**.** $\mathsf{Completion}^{\uparrow}$ *is explicit modulus data plus diagonal extraction plus classifier uniqueness. It is not an external existence theorem, not a quotient construction, and not a use of a selected representative outside the ledger.*

9357.5 Metric and Topology

Definition 9357.24 (Metric interface)**.** *For a point carrier* $(C, \sim_C)$ *with apartness boundary, a* $\mathsf{Met}^{\uparrow}$ *interface provides a distance*

$$d : C \times C \to \mathsf{Real}^{\uparrow}$$

with nonnegativity, identity of indiscernibles through the carrier classifier, symmetry, triangle inequality, endpoint congruence, and explicit real moduli for inequality transport. The ledger records the point witnesses and the real distance witness for each comparison.

Theorem 9357.25 (Metric empty distance exactness)**.** *A concrete empty distance witness classifies exactly the two endpoints with the empty history:*

$$\mathsf{MetricDistanceWitness}(x, y, \mathsf{e}) \iff \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Unfold the concrete distance witness. In the forward direction its continuation row has empty result, so continuation inversion forces both endpoints to be empty up to hsame. Conversely, transport the reflexive empty distance witness along the two endpoint hsame rows. □

Theorem 9357.26 (Metric symmetry ledger)**.** *For unary endpoint histories, forward and reversed concrete distance ledgers classify their distance histories together.*

Proof. The two witnesses are the forward and reversed unary continuation ledgers. Unary continuation commutativity supplies the hsame classifier row between the two distance histories. □

Definition 9357.27 (Metric-generated topology)**.** *For a* $\mathsf{Met}^{\uparrow}$ *carrier, the open ball centered at* x *with positive radius* $r : \mathsf{Real}^{\uparrow}$ *is*

$$B(x, r) = \{y : C \mid d(x, y) < r\}.$$

A $\mathsf{Topology}^{\uparrow}$ *surface generated by these balls records open-set membership, neighbourhood witnesses, finite intersection closure, certified union closure, membership stability, and a local name certificate.*

Theorem 9357.28 (Metric balls are open)**.** *Every metric ball with positive radius is open in the generated topology.*

Proof. Let $y \in B(x, r)$, so $d(x, y) < r$. The positive remainder $r - d(x, y)$ supplies a displayed radius budget. Take a certified half of that budget. If $d(y, z)$ is below the half-budget, then the triangle inequality gives

$$d(x, z) \leq d(x, y) + d(y, z) < r.$$

Thus z remains in the ball. The proof uses real apartness and modulus rows, not classical trichotomy. □

Principle 9357.29 (Topology boundary)**.** $\mathsf{Topology}^{\uparrow}$ *is an open-set classifier and neighbourhood interface with certified closure rows. Metric topology is generated from distance and positive-radius data; it does not provide arbitrary powersets, open-cover compactness, or an ambient point-set topology.*

9357.6 Metric Completion and Maps

Definition 9357.30 (Regular Cauchy point name)**.** *In a* $\mathsf{Met}^{\uparrow}$ *space* X*, a regular Cauchy point name is a stream* $x : \mathsf{Nat}^{\uparrow} \to X$ *satisfying*

$$\forall m\, n : \mathsf{Nat}^{\uparrow},\ d(x_m, x_n) \leq 2^{-m} + 2^{-n}.$$

Convergence to $a : X$ *is recorded by an explicit modulus* $N : \mathbb{Q}^{\uparrow}_{>0} \to \mathsf{Nat}^{\uparrow}$ *such that*

$$n \geq N(\varepsilon) \Longrightarrow d(x_n, a) < \varepsilon.$$

Definition 9357.31 (Complete and compact metric carriers)**.** *A* $\mathsf{CompleteMetric}^{\uparrow}$ *carrier is a* $\mathsf{Met}^{\uparrow}$ *carrier in which each regular Cauchy point name has an explicit limit and convergence modulus, with uniqueness under separatedness. A* $\mathsf{CompactMetric}^{\uparrow}$ *carrier is the product of metric, complete-metric, and totally-bounded rows:*

$$\mathsf{CompactMetricCarrier}(X) \Longleftrightarrow \mathsf{MetricCarrier}(X) \wedge \mathsf{CompleteMetricCarrier}(X) \wedge \mathsf{TotallyBoundedCarrier}(X).$$

The totally-bounded row supplies finite ε*-nets as* $\mathsf{List}^{\uparrow}$*-certified data.*

Theorem 9357.32 (Limit uniqueness under separatedness)**.** *If a regular Cauchy point name converges to* a *and to* b*, and the metric space is separated, then* $a \sim_X b$*.*

Proof. For any positive tolerance ε, choose one displayed index beyond the two convergence moduli at $\varepsilon/2$. The triangle inequality gives

$$d(a, b) \leq d(a, x_n) + d(x_n, b) < \varepsilon.$$

Since the distance is below every displayed tolerance, the zero-distance classifier applies; separatedness then gives $a \sim_X b$. □

Definition 9357.33 (Continuous map interface)**.** *For metric carriers* X *and* Y*, a* $\mathsf{ContinuousMap}^{\uparrow}$ *interface for* $f : X \to Y$ *contains classifier respect and a uniform modulus*

$$\omega : \mathbb{Q}^{\uparrow}_{>0} \to \mathbb{Q}^{\uparrow}_{>0}$$

such that

$$d_X(x, x') < \omega(\varepsilon) \Longrightarrow d_Y(f(x), f(x')) < \varepsilon.$$

Theorem 9357.34 (Identity and composition of continuous maps)**.** *Identity maps are continuous with identity modulus, and continuous maps compose with composed modulus.*

Proof. For the identity, take $\omega(\varepsilon) = \varepsilon$. For composition, if f has modulus ω_f and g has modulus ω_g, use

$$\omega_{g \circ f}(\varepsilon) := \omega_f(\omega_g(\varepsilon)).$$

The f row sends a source comparison below $\omega_f(\omega_g(\varepsilon))$ to a target comparison below $\omega_g(\varepsilon)$, and the g row sends that comparison below ε. Classifier respect is the composition of the two classifier-respect rows. □

9357.7 L10 Boundary

Theorem 9357.35 (Countable completion admission theorem)**.** *If the carrier, classifier, stability, modulus, ledger, and* NameCert *rows are supplied for*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{Topology}^{\uparrow},\quad \mathsf{Completion}^{\uparrow},\quad \mathsf{CompleteMetric}^{\uparrow},$$

then the L10 countable-completion layer is admitted.

Proof. $\mathsf{DyadicRatCore}^{\uparrow}$ supplies finite dyadic tolerance arithmetic. $\mathsf{StreamName}^{\uparrow}$ supplies countable observation names with pointwise classifier stability. $\mathsf{RegSeqRat}^{\uparrow}$ adds regular Cauchy bounds over dyadic streams. $\mathsf{Real}^{\uparrow}$ seals regular names and exposes only classifier-stable real observations. $\mathsf{Completion}^{\uparrow}$ adds explicit modulus data, diagonal representatives, and uniqueness through the classifier. $\mathsf{Met}^{\uparrow}$ supplies real-valued distance ledgers. $\mathsf{Topology}^{\uparrow}$ supplies generated open-set and neighbourhood classifiers. $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ add explicit limit, finite-net, and modulus-of-continuity rows. Since every row is routed through displayed carriers and ledgers, none of these interfaces imports quotient equality, selected limits, host sequences, open-cover compactness, or classical trichotomy. □

Principle 9357.36 (Downstream analytic consumption)**.** *Analytic and zeta-facing packets may consume L10 only through certified countable names, regularity witnesses, explicit moduli, real seals, metric distance ledgers, topology classifiers, and completion uniqueness rows. They may not replace those rows by host real numbers, bare limits, quotient classes, selected representatives, or ambient complete metric spaces.*

Concretized in §8758: Regular-sequence window coherence is realized as finite StreamName windows, dyadic tolerance rows, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert data.

Concretized in §6041: The Real completion universal-property packet realizes dense-source, StreamName, RegSeqRat, Bishop regular Cauchy completion, extension, uniqueness, and Real-seal rows as finite NameCert data.

Concretized in §8759: Located metric reflection realizes the countable-completion interface as a located metric packet before completion reads.

Concretized in §8925: The dyadic located-modulus packet realizes finite tolerance-to-window data for the countable completion spine.

Concretized in §8746: Regular Cauchy modulus-of-convergence packets realize finite tolerance-to-window schedules for the countable-completion spine.

Concretized in §9052: The diagonal representative route is realized as finite regular-Cauchy family, selector, dyadic tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Concretized in §9178: Compact Real interval modulus windows realize the countable-completion spine through finite interval, StreamName, RegSeqRat, DyadicRatCore tolerance, compact finite-cover, and RealUp seal rows.

Chapter 9358

Analysis Interface Spine

The countable-completion spine supplies certified streams, regular Cauchy names, real seals, metric ledgers, topology classifiers, and completion moduli. L11 consumes those rows as the substrate for analysis:

$$\mathsf{Calculus}^{\uparrow},\quad \mathsf{RealAnalytic}^{\uparrow},\quad \mathsf{CplxAnalytic}^{\uparrow},\quad \mathsf{Measure}^{\uparrow},\quad \mathsf{FunctionalAnalysis}^{\uparrow}.$$

The same layer also exposes the $\mathsf{Gamma}^{\uparrow}$ bridge used by later zeta-facing certificates. In all cases an analytic object is a packet of finite approximations, completion routes, classifier transport, explicit moduli, and ledger rows. It is not a host analytic function, a bare limit, a hidden operator norm, or an imported spectral theorem.

9358.1 Analytic Certificate Template

Definition 9358.1 (Analytic interface)**.** *An analytic interface over a carrier-classifier pair* $(C, \sim_C)$ *is a packet*

$$\mathsf{AnalyticInterface}^{\uparrow} = (C, \sim_C, \mathsf{Ops}, \mathsf{Approx}, \mathsf{Limit}, \mathsf{Stab}, \mathsf{Ledger}, \mathsf{NameCert}).$$

Here Ops *records the permitted analytic operations, such as derivative, integral, series, limit, or operator formation. The* Approx *row records finite approximants. The* Limit *row is a reference to the L10 completion route that turns the approximants into a public value. The stability row records classifier congruence, operation descent, modulus transport, and uniqueness through the target classifier.*

Definition 9358.2 (Analytic ledger)**.** *The analytic ledger records the data that make an analytic read finite: Cauchy moduli, input bounds, finite partitions, partial sums, tail estimates, positive apartness witnesses, branch witnesses, operator bounds, and the completion or metric route through which the output is accepted.*

Principle 9358.3 (Analysis is certificate-facing)**.** *An analytic name is admitted only when the finite approximation surface, the completion or measure route, the classifier transport, the stability rows, and the local* NameCert *are visible. The public surface cannot be replaced by a host function space, an external limit operator, a quotient of functions, or a choice-selected representative.*

9358.2 Calculus

Definition 9358.4 (Derivative witness)**.** *Let X and Y be normed vector carriers and let $f : X \to Y$ be a continuous map in the classifier-respecting sense. A derivative witness for f at $a : X$ is a bounded linear map $L : X \to Y$ together with a remainder row*

$$R_a(h) := f(a + h) - f(a) - L(h)$$

and a local modulus:

$$\forall \varepsilon > 0\ \exists \delta > 0, \quad 0 < \|h\| < \delta \Longrightarrow \|R_a(h)\| \le \varepsilon \|h\|.$$

The witness records the local linear approximation and its explicit error budget.

Definition 9358.5 (Derivative classifier)**.** *Two derivative witnesses L and M for the same input data classify together when they agree pointwise through the codomain classifier:*

$$L \sim_{\mathsf{Deriv}} M \Longleftrightarrow \forall h : X,\ L(h) \sim_Y M(h).$$

Theorem 9358.6 (Derivative uniqueness)**.** *If L and M are derivative witnesses for the same map at the same point, then $L \sim_{\mathsf{Deriv}} M$.*

Proof. Fix $h : X$. For $h = 0$, linearity gives $L(0) \sim_Y 0 \sim_Y M(0)$. For a displayed nonzero h, compare the two remainder rows at th with positive t tending to zero through the L10 countable-name route. Subtracting the two estimates gives

$$(L - M)(th) = o(\|th\|).$$

Linearity rewrites the left side as $t(L - M)(h)$ and the norm row rewrites the scale as $t\|h\|$. Cancelling the positive scalar in the displayed apartness field gives

$$\|(L - M)(h)\| \le \varepsilon \|h\|$$

for every displayed positive ε. The separated norm classifier therefore gives $(L - M)(h) \sim_Y 0$, hence $L(h) \sim_Y M(h)$. □

Definition 9358.7 (Finite-sum integral seed)**.** *For a continuous real-valued map on a compact interval, a calculus integral seed is a directed finite-sum system. A tagged partition*

$$P = (a = x_0 < x_1 < \cdots < x_n = b,\ \xi_i \in [x_i, x_{i+1}])$$

has finite sum

$$S(P, f) := \sum_{i=0}^{n-1} f(\xi_i)(x_{i+1} - x_i).$$

An integral witness for value $I : \mathsf{Real}^{\uparrow}$ is a regular Cauchy completion row for these sums with explicit refinement modulus and endpoint transport.

Definition 9358.8 (Measure-facing integral seed)**.** *When a $\mathsf{Measure}^{\uparrow}$ interface is available, the integral may instead be read through simple-function approximation and a completion row. The ledger records the simple functions, measurable partitions, coefficient rows, dominance or monotonicity evidence, and the completion route to the final value.*

Theorem 9358.9 (Integral uniqueness)**.** *Two integral witnesses built from the same directed finite-sum system and compatible moduli classify to the same real value.*

Proof. Both witnesses are L10 completion limits of the same directed finite-sum name. The completion uniqueness row transports the shared regular Cauchy classifier to the $\mathsf{Real}^{\uparrow}$ classifier, giving the desired equality at the real-history surface. □

Principle 9358.10 (Calculus boundary)**.** $\mathsf{Calculus}^{\uparrow}$ *consumes* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and normed vector data. The fundamental theorem belongs only after the interval or measure interface supplies stable integration and differentiation routes; it is not part of the root calculus name.*

9358.3 Real Analytic Functions

Definition 9358.11 (Real-analytic package)**.** *The* $\mathsf{RealAnalytic}^{\uparrow}$ *package contains the real-history rows for*

$$\pi, \quad \exp, \quad \log, \quad \sin, \quad \cos.$$

Each row is generated from unary-indexed rational or dyadic approximants with an explicit Cauchy modulus, then sealed through $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$.

Definition 9358.12 (Leibniz partial sums and π)**.** *For unary index n, the Leibniz partial sum row is*

$$S_n = \sum_{k=0}^{n} \frac{(-1)^k}{2k+1}.$$

The real name π is the sealed completion of the rescaled stream $(4S_n)_n$ with its alternating-series modulus.

Theorem 9358.13 (Leibniz modulus)**.** *For $N(k) = 2^{k+1}$, if $m, n \geq N(k)$, then*

$$|4S_m - 4S_n| < 2^{-k}.$$

Proof. The alternating-series tail is bounded by the first omitted term. Hence

$$|S_m - S_n| \leq \frac{1}{2N(k)+1}.$$

Multiplying by 4 and using $N(k) = 2^{k+1}$ gives

$$|4S_m - 4S_n| \leq \frac{4}{2N(k)+1} < 2^{-k}.$$

The comparison is recorded in the dyadic tolerance ledger. □

Definition 9358.14 (Exponential partial sums)**.** *For $x : \mathsf{Real}^{\uparrow}$, the exponential partial-sum row is*

$$E_n(x) = \sum_{k=0}^{n} \frac{x^k}{k!}.$$

Powers use iterated real multiplication, factorials use unary recursion, and division is read through a positive denominator and inverse certificate.

Theorem 9358.15 (Exponential modulus on bounded input)**.** *If $\|x\| \leq B$, then the exponential partial sums are regular Cauchy with a displayed modulus of the form $N(k) = 2B + k$.*

Proof. For $n \geq 2B$, the ratio of successive tail terms is at most $1/2$:

$$\frac{\|x\|}{n+1} \leq \frac{1}{2}.$$

The remaining tail is bounded by a geometric series. Taking $n \geq 2B + k$ makes the tail smaller than 2^{-k}, producing the displayed modulus. □

Theorem 9358.16 (Exponential additivity)**.** *If the three exponential rows for x, y, and $x+y$ are supplied, then*

$$\exp(x+y) \sim_{\mathsf{Real}^{\uparrow}} \exp(x)\exp(y).$$

Proof. At the finite layer, compare the partial sums for $\exp(x+y)$ with the Cauchy product of the partial sums for $\exp(x)$ and $\exp(y)$. The binomial identity matches equal-degree terms, while the exponential moduli control the tails. Passing through the $\mathsf{Real}^{\uparrow}$ completion uniqueness row yields the classifier equality. □

Definition 9358.17 (Logarithm by bisection)**.** *For a positive real input x with apartness from zero, the logarithm row is generated by dyadic bisection. At precision k it records a dyadic interval*

$$[y_k, y_k + 2^{-k}]$$

such that

$$\exp(y_k) \leq x < \exp(y_k + 2^{-k}).$$

The sequence $(y_k)_k$ is a regular Cauchy name whose seal is $\log(x)$.

Theorem 9358.18 (Logarithm inverse row)**.** *Concretized in §1817: The logarithm-by-bisection inverse route is realized as finite exponential, dyadic bracket, StreamName, RegSeqRat, and Real seal NameCert data.*

Concretized in §9119: The analytic tail-budget route is realized as finite dyadic tolerance, StreamName, RegSeqRat, Real seal, and completion-facing tail NameCert data.

For positive x, if $\log(x)$ is produced by the bisection row, then

$$\exp(\log(x)) \sim_{\mathsf{Real}^{\uparrow}} x.$$

If $\exp(z)$ is positive and a logarithm row for it is supplied, then the logarithm classifies with z.

Proof. The bisection ledger forces $\exp(y_k)$ to converge to x at every displayed precision. Completion uniqueness gives $\exp(\log(x)) \sim_{\mathsf{Real}^{\uparrow}} x$. For an input already known as $\exp(z)$, the same monotone inverse ledger and completion uniqueness identify the bisection limit with z. □

Definition 9358.19 (Trigonometric partial sums)**.** *The sine and cosine partial sums are*

$$S_n(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k+1}}{(2k+1)!},$$

and

$$C_n(x) = \sum_{k=0}^{n} (-1)^k \frac{x^{2k}}{(2k)!}.$$

Their real-history values are sealed regular Cauchy limits with explicit input bounds and alternating-tail ledgers.

Theorem 9358.20 (Trigonometric modulus and identities)**.** *For bounded input, sine and cosine partial sums have explicit Cauchy moduli. Their sealed limits satisfy*

$$\sin(x)^2 + \cos(x)^2 \sim_{\mathsf{Real}^\uparrow} 1,$$

and signed unary integer multiples of π are zeros of sine.

Proof. The factorial denominator forces the tail terms to decrease once the displayed index dominates the input bound. Alternating-tail estimates give the Cauchy moduli. The Pythagorean identity is obtained by multiplying finite partial sums, grouping the Cauchy-product terms, and sending the remaining tail through the same modulus. The integer-multiple zero row is proved by induction using the addition formula and the base $\sin(\pi)$ row. □

Principle 9358.21 (Real-analytic boundary)**.** $\mathsf{RealAnalytic}^\uparrow$ *licenses* π, exp, log, sin, *and* cos *only as regular Cauchy constructions over rational or dyadic finite approximants. They are not primitive constants or host functions.*

9358.4 Complex Analytic Functions

Definition 9358.22 (Complex exponential)**.** *For a complex history $z = x + iy$, the complex exponential row is*

$$\exp_{\mathbb{C}}(z) = \big(\exp(x)\cos(y),\ \exp(x)\sin(y)\big).$$

The classifier is componentwise through the $\mathsf{Real}^\uparrow$ *classifier.*

Theorem 9358.23 (Complex exponential stability)**.** *The complex exponential is stable under the* $\mathsf{Complex}^\uparrow$ *classifier and has a unique output class for each accepted input class.*

Proof. The $\mathsf{Complex}^\uparrow$ classifier is componentwise. The real rows exp, sin, and cos are stable under $\mathsf{Real}^\uparrow$ classifier transport. Thus both components of the complex exponential output transport through the same real classifier rows, giving a unique complex classifier result. □

Theorem 9358.24 (Complex exponential additivity and Euler row)**.** *The complex exponential satisfies*

$$\exp_{\mathbb{C}}(z + z') \sim_{\mathsf{Complex}^\uparrow} \exp_{\mathbb{C}}(z)\exp_{\mathbb{C}}(z'),$$

and for pure imaginary input $i\theta$,

$$\exp_{\mathbb{C}}(i\theta) \sim_{\mathsf{Complex}^\uparrow} (\cos\theta, \sin\theta).$$

Proof. Write $z = x + iy$ and $z' = x' + iy'$. The real part uses real exponential additivity and the cosine addition formula; the imaginary part uses real exponential additivity and the sine addition formula. For $i\theta$, the real part of the input is zero, so $\exp(0) \sim_{\mathsf{Real}^\uparrow} 1$, leaving the components $(\cos\theta, \sin\theta)$. □

Principle 9358.25 (Complex-analytic boundary)**.** $\mathsf{CplxAnalytic}^\uparrow$ *is componentwise over* $\mathsf{Complex}^\uparrow$ *and* $\mathsf{RealAnalytic}^\uparrow$*, with branch and apartness ledgers for logarithm and power operations. It does not import a host holomorphic calculus or an untracked principal branch.*

9358.5 Measure and Integration

Definition 9358.26 (Measurable carrier)**.** *A measurable carrier is a tuple* $(X, \sim_X, \mathcal{A})$ *where* $\mathcal{A}$ *is a* $\mathsf{SetLike}^{\uparrow}$ *family of event rows over* X*. The family contains empty and total events, is closed under complement and finite unions, and is closed under certified countable unions supplied as* $\mathsf{StreamName}^{\uparrow}$ *families.*

Definition 9358.27 (Measure interface)**.** *A* $\mathsf{Measure}^{\uparrow}$ *interface is a nonnegative real-valued map on event rows:*

$$\mu : \mathcal{A} \to \mathsf{Real}^{\uparrow}_{\geq 0}.$$

Its certificate records the null empty event, finite additivity on disjoint events, certified countable additivity on disjoint event streams, monotonicity, and stability under $\mathsf{SetLike}^{\uparrow}$ *extensional classification.*

Theorem 9358.28 (Monotonicity from finite additivity)**.** *If* $A \subseteq B$ *and the decomposition*

$$B = A \sqcup (B \setminus A)$$

is certified, then $\mu(A) \leq \mu(B)$*.*

Proof. Finite additivity gives

$$\mu(B) = \mu(A) + \mu(B \setminus A).$$

Nonnegativity gives $\mu(B \setminus A) \geq 0$, so the real order row yields $\mu(A) \leq \mu(B)$. □

Definition 9358.29 (Simple integral and nonnegative integral)**.** *A simple function is a finite partition expression*

$$s = \sum_{i=1}^{n} a_i \mathbf{1}_{A_i}$$

with $a_i : \mathsf{Real}^{\uparrow}$ *and* $A_i \in \mathcal{A}$*. Its integral is*

$$\int s \, d\mu := \sum_{i=1}^{n} a_i \mu(A_i).$$

For nonnegative measurable f*, the integral is read as a directed completion of simple functions below* f*, with an approximation ledger.*

Definition 9358.30 (L^p seed)**.** *For* $1 \leq p < \infty$*, the* L^p *seed carrier records functions with finite value*

$$\int |f|^p \, d\mu.$$

The classifier is almost-everywhere equality through the measure ledger:

$$f \sim g \iff \mu(\{x : f(x) \not\sim g(x)\}) = 0.$$

The zero-event and equality rows are measure certificates, not host quotient classes.

Principle 9358.31 (Measure boundary)**.** $\mathsf{Measure}^{\uparrow}$ *requires* $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and certified countable families. Countable operations are stream-name data; they are not host countable choice.*

9358.6 Functional Analysis

Definition 9358.32 (Normed vector carrier)**.** *A* $\mathsf{Norm}^{\uparrow}$ *row over a* $\mathsf{VecSpace}^{\uparrow}$ *carrier* V *is a map*

$$\|-\| : V \to \mathsf{Real}^{\uparrow}_{\geq 0}$$

with positivity, definiteness, scalar homogeneity, triangle inequality, and classifier congruence. It induces the metric used by Banach and Hilbert completion rows.

Definition 9358.33 (Banach and Hilbert carriers)**.** *A* $\mathsf{Banach}^{\uparrow}$ *carrier is a normed vector carrier with a regular Cauchy completeness witness in the norm-induced metric. A* $\mathsf{Hilbert}^{\uparrow}$ *carrier is a Banach carrier with an inner product*

$$\langle -, - \rangle : V \times V \to F$$

whose induced norm satisfies

$$\|v\|^2 \sim \langle v, v \rangle.$$

The inner product row records linearity, conjugate symmetry when needed, positivity, definiteness, and classifier congruence.

Theorem 9358.34 (Parallelogram row)**.** *In an inner-product carrier,*

$$\|v + w\|^2 + \|v - w\|^2 \sim 2\|v\|^2 + 2\|w\|^2.$$

Proof. Use $\|x\|^2 \sim \langle x, x \rangle$. Expanding $\langle v + w, v + w \rangle$ gives the two diagonal terms and the two cross terms. Expanding $\langle v - w, v - w \rangle$ gives the same diagonal terms and the negative cross terms. Addition cancels the cross terms by the additive classifier and leaves twice the two diagonal rows. □

Definition 9358.35 (Bounded linear operator)**.** *For normed vector carriers* V *and* W*, a bounded linear operator row for* $T : V \to W$ *consists of linearity plus a bound* $M : \mathsf{Real}^{\uparrow}_{\geq 0}$ *satisfying*

$$\|Tv\| \leq M\|v\|$$

for every carried v*. Operator classification is pointwise:*

$$T \sim_{\mathsf{op}} S \iff \forall v,\ T(v) \sim_W S(v).$$

Theorem 9358.36 (Bounded operators form a vector-space carrier)**.** *Bounded linear maps from* V *to* W *are closed under pointwise addition and scalar multiplication.*

Proof. If T and S have bounds M and N, then

$$\|(T + S)v\| \leq \|Tv\| + \|Sv\| \leq (M + N)\|v\|.$$

If λ is a scalar, then

$$\|(\lambda T)v\| \sim |\lambda|\|Tv\| \leq |\lambda|M\|v\|.$$

The vector-space laws descend pointwise from W through the operator classifier. □

Definition 9358.37 (Spectrum and compact operator rows)**.** *For a bounded operator* $T : V \to V$*, the resolvent row at* λ *is a bounded inverse certificate for* $T - \lambda I$*. The spectrum row records failure of that inverse package through a certified non-invertibility or apartness witness, not bare undecidable negation. A compact-operator row records either total-bounded image data or a subsequence-with-modulus witness for bounded sequences.*

Principle 9358.38 (Functional-analysis boundary)**.** $\mathsf{FunctionalAnalysis}^{\uparrow}$ *collects Banach, Hilbert, bounded-operator, dual, adjoint, compactness, and spectral-substrate rows. The spectral theorem is a major target over those rows, not part of the root functional-analysis certificate.*

9358.7 Gamma Bridge

Definition 9358.39 (Gamma pole and domain)**.** *The* $\mathsf{Gamma}^{\uparrow}$ *pole locus is the complex-history row saying that* s *is a non-positive integer:*

$$\mathsf{GammaPole}(s) \iff \exists n : \mathsf{Nat}^{\uparrow},\ s \sim_{\mathsf{Complex}^{\uparrow}} -n.$$

The domain row $\mathsf{GammaDom}(s)$ *contains a* $\mathsf{Complex}^{\uparrow}$ *carrier and an explicit apartness witness from that pole locus.*

Definition 9358.40 (Gamma by product and recurrence)**.** *The gamma row is built from the integer factorial identity*

$$\Gamma(n+1) = n!,$$

a Weierstrass-product extension on $\mathsf{GammaDom}$ *with unary Cauchy modulus, and the recurrence*

$$\Gamma(s+1) = s\Gamma(s).$$

The construction consumes $\mathsf{RealAnalytic}^{\uparrow}$ *exponent and logarithm rows,* $\mathsf{Complex}^{\uparrow}$ *arithmetic, and the pole-apartness ledger.*

Theorem 9358.41 (Gamma uniqueness)**.** *If two* $\mathsf{Gamma}^{\uparrow}$ *rows evaluate the same domain point through compatible product moduli, their outputs classify together in* $\mathsf{Complex}^{\uparrow}$*.*

Proof. Both outputs are limits of the same unary-indexed product stream on the same domain row. The complex limit uniqueness row transports the shared product classifier to the $\mathsf{Complex}^{\uparrow}$ classifier. □

Principle 9358.42 (Gamma boundary)**.** $\mathsf{Gamma}^{\uparrow}$ *is an analytic bridge over* $\mathsf{Complex}^{\uparrow}$ *and* $\mathsf{RealAnalytic}^{\uparrow}$*. It is not a meromorphic black box; the public row must expose pole apartness, product modulus, recurrence, transport, replay, and local naming.*

Concretized in §8843: The Mellin transform packet realizes the Gamma-facing transform branch as finite positive-real source, integrand-window, Gamma handoff, Real-seal, transport, replay, provenance, and NameCert rows.

9358.8 L11 Boundary

Theorem 9358.43 (Analysis admission theorem)**.** *If*

$$\mathsf{Calculus}^{\uparrow},\quad \mathsf{RealAnalytic}^{\uparrow},\quad \mathsf{CplxAnalytic}^{\uparrow},\quad \mathsf{Measure}^{\uparrow},\quad \mathsf{FunctionalAnalysis}^{\uparrow},\quad \mathsf{Gamma}^{\uparrow}$$

carry their advertised carrier, classifier, finite approximation, completion, modulus, stability, ledger, and $\mathsf{NameCert}$ *rows, then the L11 analysis layer is admitted.*

Proof. $\mathsf{Calculus}^{\uparrow}$ supplies local linear approximation and finite-sum or measure-facing integration through L10 completion. $\mathsf{RealAnalytic}^{\uparrow}$ supplies π, exponential, logarithm, sine, and cosine through explicit series or bisection moduli. $\mathsf{CplxAnalytic}^{\uparrow}$ composes those real rows componentwise over $\mathsf{Complex}^{\uparrow}$ and records Euler and exponential-additivity rows. $\mathsf{Measure}^{\uparrow}$ supplies stable event algebras, certified countable additivity, integration, and L^p seeds. $\mathsf{FunctionalAnalysis}^{\uparrow}$ supplies normed, Banach, Hilbert, bounded-operator, compactness, and spectral-substrate rows. $\mathsf{Gamma}^{\uparrow}$ supplies the analytic bridge needed for zeta-facing functional-equation packets. Each row is a certified finite-approximation or modulus-bearing route, so no host analytic object enters as a primitive. □

Principle 9358.44 (Downstream zeta consumption)**.** *Zeta-facing packets may consume L11 only through the real and complex analytic rows, measure or contour-integral ledgers when present, gamma-domain and gamma-product rows, and functional-analysis operator rows when a spectral route is chosen. They may not replace these rows by host exponentials, untracked logarithm branches, classical nonzero predicates, or black-box meromorphic functions.*

Chapter 9359

Geometry, Category, and Homology Interface Spine

Concretized in §8931: Finite cover incidence realizers instantiate the geometry/category/homology spine as cover membership, overlap, simplicial face, homology handoff, transport, replay, provenance, and local NameCert rows.

The analysis layer supplies real and complex analytic rows, measure-facing integration, functional-analysis carriers, and gamma bridges. L12 reads those rows together with the earlier type, set, topology, continuous-map, and abelian-group interfaces as a geometry layer:

$$\mathsf{Cat}^{\uparrow},\quad \mathsf{Functor}^{\uparrow},\quad \mathsf{NatTrans}^{\uparrow},\quad \mathsf{CatLimit}^{\uparrow},\quad \mathsf{Manifold}^{\uparrow},\quad \mathsf{Bundle}^{\uparrow},\quad \mathsf{Homology}^{\uparrow}.$$

The role of this layer is not to add a point-set universe. It turns certified carriers into local data, transition data, gluing data, classifier-stable maps, and obstruction ledgers.

9359.1 Geometry Certificate Template

Definition 9359.1 (Geometry interface)**.** *A geometry interface over a carrier-classifier pair* $(C, \sim_C)$ *is a packet*

$$\mathsf{GeomInterface}^{\uparrow} = (C, \sim_C, \mathsf{LocalData}, \mathsf{Gluing}, \mathsf{Transition}, \mathsf{Stab}, \mathsf{Ledger}, \mathsf{NameCert}).$$

The local-data row may contain charts, local trivialisations, simplices, chains, cone components, morphisms, or sections. The gluing row records how local presentations assemble. The transition row records overlap maps, change-of-chart maps, functorial transport, boundary maps, or comparison morphisms. The stability row states that all of those rows descend through the relevant classifiers.

Principle 9359.2 (Geometry is not a point set)**.** *A L12 object is admitted only as carrier, classifier, local data, transition or gluing data, ledger, and local* NameCert*. A host set of points, a quotient of representatives, an extensional equality of objects, or a selected global representative is never the public surface.*

9359.2 Categories, Functors, and Natural Transformations

Definition 9359.3 (Category carrier)**.** *A* $\mathsf{Cat}^{\uparrow}$ *carrier is a multisorted certificate*

$$\mathcal{C} = (\mathsf{Obj}, \sim_{\mathsf{Obj}}, \mathsf{Hom}, \sim_{\mathsf{Hom}}, \mathsf{id}, \mathsf{comp}, \mathsf{Law}, \mathsf{Ledger}, \mathsf{NameCert}).$$

Objects are certified carriers. For certified objects A, B, the hom row $\mathsf{Hom}_{\mathcal{C}}(A, B)$ is a certified morphism carrier with its own parallel-morphism classifier. Identity and composition are operations on those hom rows, and the unit, associativity, and congruence laws are stated up to the displayed morphism classifiers.

Definition 9359.4 (Isomorphism record)**.** *Object comparison in a $\mathsf{Cat}^{\uparrow}$ carrier is never inferred from host extensionality. An isomorphism between A and B is a record*

$$(f : A \to B,\ g : B \to A,\ g \circ f \sim \mathsf{id}_A,\ f \circ g \sim \mathsf{id}_B)$$

with both triangle rows living in the relevant morphism classifiers. Subcategory inclusion is likewise data: an object row, a hom-row inclusion, and stability under identity and composition.

Theorem 9359.5 (Identity uniqueness)**.** *If $e_A : A \to A$ is both a left and right unit for every parallel hom row at A, then $e_A \sim \mathsf{id}_A$ in $\mathsf{Hom}_{\mathcal{C}}(A, A)$.*

Proof. Apply the unit property of e_A to id_A. One unit row gives $e_A \circ \mathsf{id}_A \sim \mathsf{id}_A$, while the unit row for id_A gives $e_A \circ \mathsf{id}_A \sim e_A$. Symmetry and transitivity of the morphism classifier identify e_A with id_A. □

Definition 9359.6 (Functor certificate)**.** *A $\mathsf{Functor}^{\uparrow}$ certificate $F : \mathcal{C} \to \mathcal{D}$ consists of an object map F_0, endpoint-indexed morphism maps $F_{A,B}$, classifier-respect rows for objects and morphisms, and preservation rows*

$$F(\mathsf{id}_A) \sim \mathsf{id}_{F(A)}, \qquad F(g \circ f) \sim F(g) \circ F(f).$$

The codomain of a morphism image is the hom carrier at the image endpoints; raw preservation of unclassified composites is not enough.

Theorem 9359.7 (Functor composition)**.** *Certified functors compose, and the identity functor is certified.*

Proof. For $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{E}$, define the composite on objects by $G_0(F_0(A))$ and on morphisms by $G(F(f))$. Classifier respect is the composite of the two classifier-respect rows. Identity preservation follows from

$$G(F(\mathsf{id}_A)) \sim G(\mathsf{id}_{F(A)}) \sim \mathsf{id}_{G(F(A))}.$$

Composition preservation follows in the same way:

$$G(F(g \circ f)) \sim G(F(g) \circ F(f)) \sim G(F(g)) \circ G(F(f)).$$

The identity functor uses the identity object and morphism maps, with all rows supplied by reflexivity. □

Definition 9359.8 (Natural transformation certificate)**.** *For functors $F, G : \mathcal{C} \to \mathcal{D}$, a $\mathsf{NatTrans}^{\uparrow}$ certificate $\eta : F \Rightarrow G$ is an object-indexed family of component morphisms*

$$\eta_A : F(A) \to G(A)$$

such that every $f : A \to B$ satisfies the naturality classifier row

$$G(f) \circ \eta_A \sim \eta_B \circ F(f).$$

Theorem 9359.9 (Vertical natural-transformation composition)**.** *If $\eta : F \Rightarrow G$ and $\theta : G \Rightarrow H$ are certified natural transformations, then the family $(\theta_A \circ \eta_A)_A$ is a certified natural transformation $F \Rightarrow H$.*

Proof. For $f : A \to B$, reassociate $H(f) \circ (\theta_A \circ \eta_A)$. Naturality of θ rewrites $H(f) \circ \theta_A$ as $\theta_B \circ G(f)$, and naturality of η rewrites $G(f) \circ \eta_A$ as $\eta_B \circ F(f)$. Associativity in $\mathcal{D}$ gives $(\theta_B \circ \eta_B) \circ F(f)$. □

Remark 9359.10 (Kernel category boundary)**.** *chapter 9363 records the checked unary-continuation category over* BHist *and the bounded* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *bridges available at the paper-Lean surface. The present spine uses that result as the kernel-facing instance and keeps the general L12 interface at the certificate level.*

9359.3 Categorical Limits

Definition 9359.11 (Diagram, cone, and limit)**.** *For certified categories J and $\mathcal{C}$, a diagram is a certified functor $D : J \to \mathcal{C}$. A cone over D is an apex object L together with components*

$$\lambda_j : L \to D(j)$$

such that every arrow $u : j \to k$ in J satisfies

$$D(u) \circ \lambda_j \sim \lambda_k.$$

A $\mathsf{CatLimit}^{\uparrow}$ *row is a cone (L, λ) that is universal among cones: for every cone (X, ξ) there is a morphism $u : X \to L$, unique up to the morphism classifier, such that $\lambda_j \circ u \sim \xi_j$ for all j.*

Theorem 9359.12 (Limit comparison isomorphism)**.** *Two* $\mathsf{CatLimit}^{\uparrow}$ *cones over the same certified diagram are connected by an isomorphism record whose composites classify with the two identity morphisms.*

Proof. Let (L, λ) and (M, μ) be limiting cones. Apply the universal property of M to the cone L to obtain $f : L \to M$ with $\mu_j \circ f \sim \lambda_j$. Apply the universal property of L to the cone M to obtain $g : M \to L$ with $\lambda_j \circ g \sim \mu_j$. The composite $g \circ f$ is a cone endomorphism of (L, λ), since

$$\lambda_j \circ (g \circ f) \sim (\lambda_j \circ g) \circ f \sim \mu_j \circ f \sim \lambda_j.$$

The identity of L satisfies the same component equations, so uniqueness gives $g \circ f \sim \mathsf{id}_L$. The other composite is identical by symmetry with L and M interchanged. □

9359.4 Manifolds and Bundles

Definition 9359.13 (Charted manifold)**.** *A* $\mathsf{Manifold}^{\uparrow}$ *certificate is a topological carrier equipped with a finite atlas. Each chart has an open domain U_i, a finite coordinate row into* $(\mathsf{Real}^{\uparrow})^{n_i}$*, and a classifier-stable coordinate map. Each overlap row $U_i \cap U_j$ carries a transition ledger from the i-coordinates to the j-coordinates, certified by the available real-analytic and calculus interfaces.*

Theorem 9359.14 (Chart-change classifier stability)**.** *On an overlap $U_i \cap U_j$, the chart-change expression $\varphi_j(\varphi_i^{-1}(\varphi_i(x)))$ classifies with $\varphi_j(x)$.*

Proof. The inverse row for the i-chart gives $\varphi_i^{-1}(\varphi_i(x)) \sim x$ in the manifold carrier classifier. Classifier congruence for the j-chart transports this comparison to the coordinate row, yielding the displayed chart-change stability. □

Theorem 9359.15 (Transition cocycle)**.** *On a triple overlap $U_i \cap U_j \cap U_k$, the transition from chart i to chart k classifies with the composite of the transitions from i to j and from j to k.*

Proof. The composite transition expands as

$$\varphi_k \circ \varphi_j^{-1} \circ \varphi_j \circ \varphi_i^{-1}.$$

The middle inverse row classifies $\varphi_j^{-1} \circ \varphi_j$ with the identity on the shared coordinate window. Composition congruence then reduces the composite to $\varphi_k \circ \varphi_i^{-1}$. □

Definition 9359.16 (Bundle certificate)**.** *A* $\mathsf{Bundle}^{\uparrow}$ *certificate over a base* $\mathsf{Manifold}^{\uparrow}$ *row is a fibred carrier $p : E \to B$ with model fibre F, local trivialisations*

$$\tau_i : p^{-1}(U_i) \to U_i \times F,$$

transition functions on overlaps, cocycle rows on triple overlaps, and a ledger for section, connection, and curvature-facing consumers when those interfaces are present. The projection, local trivialisation, and transition rows are all classifier-stable.

Theorem 9359.17 (Bundle transition identity and cocycle)**.** *For a* $\mathsf{Bundle}^{\uparrow}$ *certificate, g_{ii} classifies with the identity on the model fibre, and on triple overlaps g_{ik} classifies with $g_{jk} \circ g_{ij}$.*

Proof. The identity row follows from $\tau_i \circ \tau_i^{-1}$ on $U_i \times F$. For the triple-overlap row, compare $\tau_k \circ \tau_i^{-1}$ with $(\tau_k \circ \tau_j^{-1}) \circ (\tau_j \circ \tau_i^{-1})$. The shared j-trivialisation cancels by the inverse law, and the fibre components read as g_{ik} and $g_{jk} \circ g_{ij}$. □

9359.5 Homology and Cohomology

Definition 9359.18 (Chain complex and homology)**.** *A* $\mathsf{Homology}^{\uparrow}$ *certificate starts with a degree-indexed family of* $\mathsf{AbGroup}^{\uparrow}$ *carriers C_n and boundary maps $\partial_n : C_n \to C_{n-1}$ satisfying the square-zero row*

$$\partial_{n-1} \circ \partial_n \sim 0.$$

The cycle carrier is $\mathsf{Ker}(\partial_n)$*, the boundary carrier is* $\mathsf{Im}(\partial_{n+1})$*, and the homology object is the* $\mathsf{QuotientGroup}^{\uparrow}$ *row Z_n/B_n with its carrier, classifier, projection, soundness, completeness, and ledger.*

Theorem 9359.19 (Boundaries are cycles)**.** *For every degree n, the boundary carrier B_n is contained in the cycle carrier Z_n.*

Proof. If $c \in B_n$, then some $d \in C_{n+1}$ satisfies $c \sim \partial_{n+1}(d)$. Applying ∂_n gives

$$\partial_n(c) \sim \partial_n(\partial_{n+1}(d)) \sim (\partial_n \circ \partial_{n+1})(d) \sim 0.$$

Thus c is a cycle. □

Definition 9359.20 (Chain-map transport)**.** *A chain map $f : C \to D$ is a family of homomorphism rows $f_n : C_n \to D_n$ satisfying*

$$\partial_n^D \circ f_n \sim f_{n-1} \circ \partial_n^C.$$

It sends cycles to cycles and boundaries to boundaries, so it induces a classifier-stable map on the quotient rows $H_n(C) \to H_n(D)$.

Definition 9359.21 (Cohomology row)**.** *A* $\mathsf{Cohomology}^{\uparrow}$ *certificate is the dual cochain interface: groups C^n, coboundary maps $d^n : C^n \to C^{n+1}$, a square-zero row $d^{n+1} \circ d^n \sim 0$, cocycles* $\mathsf{Ker}(d^n)$*, coboundaries* $\mathsf{Im}(d^{n-1})$*, and quotient-group rows $H^n(C)$.*

9359.6 L12 Boundary

Theorem 9359.22 (Geometry-category-homology admission)**.** *If the advertised category, functor, natural-transformation, categorical-limit, manifold, bundle, homology, and cohomology rows carry their carrier, classifier, local-data, transition, gluing, stability, ledger, and* NameCert *fields, then the L12 layer is admitted as a BEDC interface.*

Proof. $\mathsf{Cat}^{\uparrow}$ supplies certified object and morphism carriers with laws up to the morphism classifier. $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ add only classifier-respecting transport between those carriers. $\mathsf{CatLimit}^{\uparrow}$ packages the universal cone surface over certified diagrams. $\mathsf{Manifold}^{\uparrow}$ turns topological and real-analytic rows into finite chart atlases with overlap ledgers. $\mathsf{Bundle}^{\uparrow}$ adds fibred carriers, local trivialisations, and cocycle rows over the manifold substrate. $\mathsf{Homology}^{\uparrow}$ and $\mathsf{Cohomology}^{\uparrow}$ read square-zero chain or cochain data through kernel, image, and quotient carriers over $\mathsf{AbGroup}^{\uparrow}$. Each item is a certificate over earlier rows, so no host geometry, quotient, or global representative is introduced. □

Principle 9359.23 (Zeta-facing geometric consumption)**.** *Zeta-facing packets may use L12 only through certified categorical limits, bundle and section ledgers, manifold-coordinate rows, and homology or cohomology obstruction rows. A boundary-bulk-fibre-section reading is therefore a certificate route through* $\mathsf{CatLimit}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* $\mathsf{Bundle}^{\uparrow}$*, and* $\mathsf{Homology}^{\uparrow}$*, not an external geometric oracle.*

Chapter 9360

Number Theory and Zeta Interface Spine

The geometry layer exposes category, gluing, bundle, and homology rows. L13 uses the arithmetic, completion, analysis, and geometry interfaces to reach the zeta-facing surface:

$$\mathsf{Prime}^{\uparrow},\quad \mathsf{Padic}^{\uparrow},\quad \mathsf{Adele}^{\uparrow},\quad \mathsf{Dirichlet}^{\uparrow},\quad \mathsf{Zeta}^{\uparrow},\quad \mathsf{CritStrip}^{\uparrow},\quad \mathsf{ZetaZeros}^{\uparrow}.$$

The governing boundary is strict: Riemann-hypothesis-shaped statements are theorem targets. They are not naming-seed assertions, not zero-enumeration procedures, and not consequences of merely licensing $\mathsf{ZetaZeros}^{\uparrow}$.

9360.1 Prime Axis

Definition 9360.1 (Divisibility and primality)**.** *For unary histories d, n, divisibility is the displayed quotient predicate*

$$\mathsf{NatDivides}(d, n) \iff \exists q,\ \mathsf{NatMul}(d, q, n).$$

The quotient q is part of the witness. A prime row $\mathsf{NatPrime}(p)$ requires unary membership, strict inequality $p > 1$ through the unary strict-prefix relation, and the divisor condition that every divisor of p classifies with the unit or with p.

Theorem 9360.2 (Unit exclusion)**.** *The unary unit is not prime.*

Proof. If the unit were prime, the strict inequality field in the prime predicate would give a strict unary prefix from the unit to itself. Strict prefix is irreflexive, so the prime witness cannot exist. □

Definition 9360.3 (Trial division and factorization ledger)**.** *The $\mathsf{Prime}^{\uparrow}$ certificate records bounded trial division over finite unary segments, factorization lists*

$$[p_1, \ldots, p_k]$$

whose entries are nondecreasing unary primes and whose iterated product classifies with the target history, and the Euclid-lemma ledger used to compare two such lists.

Theorem 9360.4 (Prime factorization uniqueness)**.** *Two prime factorization lists of the same positive unary history have the same length and pointwise* hsame*-classified entries.*

Proof. The first prime in one list divides the product of the other list. Euclid's lemma moves that divisibility to one factor. Since that factor is prime, the two primes classify together. The nondecreasing ledger aligns the matching entries, cancels them from the two products, and the argument recurses on the remaining lists. □

Definition 9360.5 (Prime-coordinate readout)**.** *For a positive unary history n and a prime row p, the exponent $v_p(n) = k$ means that n classifies with $p^k m$ for some unary m not divisible by p. The prime-coordinate readout is the finite-support family*

$$\Phi(n) = (v_p(n))_{p:\mathsf{Prime}^{\uparrow}}.$$

The finite support is read from the factorization list.

Theorem 9360.6 (Multiplication is coordinate addition)**.** *For positive unary histories m, n,*

$$\Phi(mn) = \Phi(m) + \Phi(n).$$

Proof. Factorization gives $m = \prod_p p^{v_p(m)}$ and $n = \prod_p p^{v_p(n)}$ with finite support. Multiplying the two products adds exponents prime by prime. Factorization uniqueness identifies those exponents with the coordinates of mn. □

Concretized in §64: PrimeUp realizes the Prime Axis divisibility, primality, bounded trial-division, and factorization-ledger rows as finite NameCert data.

9360.2 p-Adic and Adele Rows

Definition 9360.7 (Prime scale and valuation)**.** *For histories p, e, r, the predicate* $\mathsf{PadicPrimeScale}(p, e, r)$ *holds when p is a* $\mathsf{Prime}^{\uparrow}$ *prime and r is the unary multiplication result of p by exponent e. For a nonzero rational history $x = a/b$, the valuation row is*

$$v_p(x) = v_p(a) - v_p(b),$$

with well-definedness supplied by the rational-normalisation ledger.

Theorem 9360.8 (Prime scale totality)**.** *If p is prime and e is unary, then some r satisfies* $\mathsf{PadicPrimeScale}(p, e, r)$*.*

Proof. The prime witness supplies unary membership for p. Unary multiplication totality applied to p and e gives a result r and a $\mathsf{NatMul}(p, e, r)$ witness. Pairing this multiplication row with the prime row is exactly the prime-scale packet. □

Definition 9360.9 (p-adic carrier)**.** *For a prime row p, the* $\mathsf{Padic}^{\uparrow}(p)$ *carrier is the regular Cauchy completion of* $\mathsf{Rat}^{\uparrow}$ *under*

$$d_p(x, y) = |x - y|_p, \qquad |x|_p = p^{-v_p(x)}.$$

The classifier is zero p-adic distance, and the ledger records the prime index, valuation decomposition, ultrametric estimate, Cauchy modulus, and completion seal.

Theorem 9360.10 (Ultrametric estimate)**.** *The p-adic metric satisfies*

$$d_p(x, z) \leq \max(d_p(x, y), d_p(y, z)).$$

Proof. Write $x - z = (x - y) + (y - z)$. The valuation ledger gives $v_p(a + b) \geq \min(v_p(a), v_p(b))$. Applying $p^{-(-)}$ reverses this minimum into the displayed maximum bound for the absolute values. □

Definition 9360.11 (Adele carrier)**.** *An* $\mathsf{Adele}^{\uparrow}$ *row is a restricted product packet*

$$x = (x_\infty, (x_p)_{p:\mathsf{Prime}^{\uparrow}})$$

with $x_\infty : \mathsf{Real}^{\uparrow}$*,* $x_p : \mathsf{Padic}^{\uparrow}(p)$*, and a finite exception list of primes outside which* $|x_p|_p \leq 1$*. The classifier is componentwise, together with classification of the restricted-product exception ledgers.*

Theorem 9360.12 (Diagonal rationals are restricted)**.** *The diagonal embedding* $\Delta : \mathsf{Rat}^{\uparrow} \to \mathsf{Adele}^{\uparrow}$ *lands in the restricted product carrier.*

Proof. For a normalized rational $q = a/b$, only the primes occurring in the factorization lists of a and b can have nonzero $v_p(q)$. That finite union is the exception ledger. For all other primes, $v_p(q) = 0$, so $|q|_p = 1$ and the restricted-product bound holds. □

9360.3 Dirichlet Series and Zeta

Definition 9360.13 (Dirichlet series row)**.** *A* $\mathsf{Dirichlet}^{\uparrow}$ *row consists of coefficient histories* $a : \mathsf{Nat}^{\uparrow}_{>0} \to \mathsf{Complex}^{\uparrow}$*, a complex input* s*, Dirichlet terms*

$$T_n(a, s) = a_n n^{-s}, \qquad n^{-s} := \exp(-s \log n),$$

an absolute-convergence half-plane ledger, regular Cauchy partial sums, and a classifier-stable limit value.

Definition 9360.14 (Completely multiplicative coefficients)**.** *A coefficient row is completely multiplicative when* a_1 *is the unit and*

$$a_{mn} \sim a_m a_n$$

for all positive unary m, n*. This is the coefficient condition that permits Euler-product regrouping through* $\mathsf{Prime}^{\uparrow}$ *factorization.*

Theorem 9360.15 (Euler product for Dirichlet rows)**.** *For completely multiplicative coefficients in the absolute-convergence half-plane,*

$$\sum_{n \geq 1} a_n n^{-s} \sim \prod_p (1 - a_p p^{-s})^{-1}.$$

Proof. Every positive unary n has a unique finite-support prime factorization $n = \prod_p p^{k_p}$. Complete multiplicativity gives $a_n \sim \prod_p a_p^{k_p}$, while the complex analytic term row gives $n^{-s} \sim \prod_p p^{-sk_p}$. The term indexed by n therefore matches a finite-support exponent tuple. Absolute convergence supplies the modulus that permits regrouping into the product of geometric series. □

Definition 9360.16 (Zeta rows)**.** $\mathsf{ZetaBasic}^{\uparrow}$ *is the specialization of* $\mathsf{Dirichlet}^{\uparrow}$ *at the constant unit coefficient row on the half-plane* $\Re(s) > 1$*. It has both the Dirichlet-sum face and the Euler-product face*

$$\zeta(s) \sim \prod_p (1 - p^{-s})^{-1}.$$

The $\mathsf{ZetaCont}^{\uparrow}$ *row extends this carrier to* $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ *by a continuation ledger with pole, functional-equation, trivial-zero, and* $\mathsf{Gamma}^{\uparrow}$ *horizon rows.*

Theorem 9360.17 (Zeta non-vanishing in the Euler-product half-plane)**.** *For $\Re(s) > 1$, the* $\mathsf{ZetaBasic}^{\uparrow}$ *value is apart from zero.*

Proof. The Euler-product row gives factors $(1 - p^{-s})^{-1}$. Their denominators have lower bound $1 - p^{-\Re(s)}$, and the product convergence ledger gives an explicit positive bound for the finite products and their limit. Thus the limit value is apart from the zero complex history. □

9360.4 Critical Strip and Zeta Zeros

Definition 9360.18 (Critical strip and line)**.** *For a complex history s with real component σ, the critical strip row is*

$$\mathsf{InCritStrip}(s) \iff 0 < \sigma < 1$$

through strict order and apartness witnesses. The critical line row is

$$\mathsf{OnCritLine}(s) \iff \sigma \sim_{\mathsf{Real}^{\uparrow}} \frac{1}{2}.$$

The mirror involution is $J(s) = 1 - \overline{s}$, whose fixed points are exactly the critical-line rows.

Definition 9360.19 (Zeta zero ledger)**.** *A* $\mathsf{ZetaZeros}^{\uparrow}$ *row contains a source s, a value row* $\mathsf{ZetaVal}(s, z)$ *read through* $\mathsf{ZetaCont}^{\uparrow}$*, and a zero-pattern row $z \sim_{\mathsf{Complex}^{\uparrow}} 0$. A nontrivial zero row is a zeta-zero row together with the displayed* $\mathsf{InCritStrip}(s)$ *row.*

Theorem 9360.20 (Nontrivial zero is strip-bound)**.** *Every nontrivial-zero row carries its critical-strip row.*

Proof. This is projection from the nontrivial-zero packet: it is a pair consisting of the zero row and the strip row. □

Theorem 9360.21 (Conjugate zero symmetry)**.** *If s is a zeta-zero row, then $\overline{s}$ is a zeta-zero row.*

Proof. The Dirichlet sum and continuation ledgers respect complex conjugation, so $\zeta(\overline{s}) \sim \overline{\zeta(s)}$. The conjugate of the zero complex history is classified with zero, hence the zero row transports to $\overline{s}$. □

Principle 9360.22 (Zero enumeration boundary)**.** $\mathsf{ZetaZeros}^{\uparrow}$ *defines the zero predicate and its classifier-stable ledger. It does not constructively enumerate all nontrivial zeros, decide exact zerohood from finite precision, or prove the critical-line conclusion.*

9360.5 RH Target Surface

Definition 9360.23 (RH predicate)**.** *The RH-facing statement is the dependent target*

$$\mathsf{RH} := \prod_{s:\mathsf{Complex}^{\uparrow}} \left(\mathsf{InCritStrip}(s) \times \mathsf{ZetaZero}(s) \to \mathsf{OnCritLine}(s)\right).$$

A constructive counterexample surface is a complex history together with a critical-strip row, a zeta-zero row, and a displayed apartness witness from the mirror fixed locus.

Definition 9360.24 (Finite prime-window defect)**.** *For a nonempty finite prime window S, define the defect expression*

$$D_S(s) := \sum_{p \in S} \log |p^{1/2-s}|.$$

If $s = 1/2 + \delta + i\tau$, then the modulus ledger reads

$$|D_S(s)| = |\delta| \sum_{p \in S} \log p.$$

Principle 9360.25 (Defect-zero incompatibility target)**.** *The central RH-facing target is the theorem schema*

$$\mathsf{DefectZeroIncompatibility}.$$

It says that a nonempty finite prime-window defect inside the completed zeta carrier forces apartness of the zeta value from zero. This is not supplied by licensing $\mathsf{ZetaZeros}^{\uparrow}$*; it is the remaining rigidity target.*

Theorem 9360.26 (Defect incompatibility implies RH)**.** *Assume defect-zero incompatibility, zero-ledger soundness, a route from off-fixed critical-strip sources to a nonempty finite prime-window defect, and located-real collapse from absence of off-fixed apartness to the critical-line row. Then* RH *follows.*

Proof. Let s carry $\mathsf{InCritStrip}(s)$ and $\mathsf{ZetaZero}(s)$. If s is apart from the critical line, the off-fixed-to-defect route supplies a nonempty finite prime window S with $|D_S(s)| > 0$. Defect-zero incompatibility then gives apartness of $\zeta(s)$ from zero. Zero-ledger soundness gives the zero value, a contradiction. Hence no off-fixed apartness remains, and located-real collapse supplies $\mathsf{OnCritLine}(s)$. □

9360.6 L13 Boundary

Theorem 9360.27 (Number-theory and zeta admission)**.** *If the prime, prime-coordinate, p-adic, adele, Dirichlet-series, zeta-basic, critical-strip, zeta-continuation, and zeta-zero rows carry their advertised carrier, classifier, stability, ledger, and* $\mathsf{NameCert}$ *fields, then the L13 number-theory and zeta layer is admitted.*

Proof. $\mathsf{Prime}^{\uparrow}$ supplies divisibility, primality, trial division, Euclid rows, and factorization uniqueness. Prime-coordinate rows turn multiplication into finite-support exponent addition. $\mathsf{Padic}^{\uparrow}$ reads those axes as valuations, non-archimedean metrics, and completions of $\mathsf{Rat}^{\uparrow}$. $\mathsf{Adele}^{\uparrow}$ packages the archimedean and p-adic components with a finite restricted-product exception ledger. $\mathsf{Dirichlet}^{\uparrow}$ supplies regular Cauchy sums, convergence half-planes, holomorphicity, and Euler products. $\mathsf{ZetaBasic}^{\uparrow}$ specializes that interface to unit coefficients, while $\mathsf{ZetaCont}^{\uparrow}$ supplies continuation with pole, functional-equation, trivial-zero, and $\mathsf{Gamma}^{\uparrow}$ ledgers. $\mathsf{CritStrip}^{\uparrow}$ and $\mathsf{ZetaZeros}^{\uparrow}$ expose the zero-facing predicate and its boundary rows. Each row is a certificate over earlier layers, so no RH claim, zero enumeration, or ambient analytic oracle is introduced. □

Principle 9360.28 (Omega and zeta readout)**.** *At L13, prime factorization supplies the multiplicative coordinate axes, Dirichlet rows apply the exponential kernel to those axes, zeta rows recover the Euler-product carrier, continuation extends the carrier away from the pole, and zero rows expose ledgered zero atoms. This is the zeta-facing readout route; RH remains a rigidity theorem over that route.*

Chapter 9361

Omega Generation Structure Interface

chapter 9351 supplies the generated history carrier and continuation relation, while chapter 9353 licenses finite displays such as sums, products, and lists through certificates and ledgers. The present chapter records the structural interface needed before the Ω-tower of chapter 9362 is read as a single generated body: encoded display, pairing by involution, and the asymmetric time arrow. These rows do not add primitive objects. They state how already licensed interfaces may be displayed, paired, and ordered by generation.

9361.1 Licensed Generation Category

Definition 9361.1 (Licensed interface object)**.** *A licensed interface object is a packet*

$$A = (|A|, \sim_A, \Lambda_A, \mathsf{NameCert}_A)$$

where $|A|$ is a carrier predicate over admitted histories or packets, $\sim_A$ is its public classifier, Λ_A is its ledger, and $\mathsf{NameCert}_A$ is the displayed naming certificate. The certificate fields are the source specification, pattern specification, classifier specification, stability certificate, and ledger policy.

Definition 9361.2 (Licensed morphism)**.** *A licensed morphism $f : A \to B$ is a carrier map that preserves classifiers, ledgers, and continuation transport:*

$$x \sim_A y \to f(x) \sim_B f(y),$$

$$\Lambda_A(x) \to \Lambda_B(f(x)),$$

and, whenever the source interface exposes a continuation row,

$$\mathsf{Cont}_A(x, m, x') \to \mathsf{Cont}_B(f(x), m', f(x')).$$

The category of such packets and morphisms is denoted Lic_Ω.

Principle 9361.3 (No unlicensed tower coordinates)**.** *Every coordinate used by an Ω-diagram must be an object of Lic_Ω. A host carrier, informal pairing, or external encoding is invisible to the tower until it is represented by carrier, classifier, stability, ledger, and $\mathsf{NameCert}$ rows.*

Definition 9361.4 (Generation diagram)**.** *An Ω generation diagram is a directed diagram*

$$\mathcal{D}_\Omega : I \to \mathsf{Lic}_\Omega, \qquad i \mapsto R_i,$$

with readback morphisms

$$\rho_{ji} : R_j \to R_i \qquad (j \geq i).$$

Its compatible body is the carrier

$$T_\Omega := \sum_{(x_i) \in \prod_{i \in I} R_i} \prod_{j \geq i} \rho_{ji}(x_j) \sim_i x_i,$$

together with the induced body ledger assembled from the coordinate ledgers and readback witnesses.

9361.2 Encoded Display

Definition 9361.5 (Legal stream display)**.** *A legal stream display for T_Ω is a licensed interface $\mathsf{LegalStream}^\uparrow$ equipped with a stream classifier $\sim_{\mathsf{Code}}$, a stream ledger Λ_{Code}, and a decode relation*

$$\mathsf{Decode}_\Omega : \mathsf{LegalStream}^\uparrow \to T_\Omega \to \mathsf{Prop}.$$

The stream interface is not a raw bit string. It is a certified public display whose suppressed source rows remain available through the ledger.

Definition 9361.6 (Omega code morphism)**.** *An Ω code morphism is a licensed morphism*

$$\mathsf{Code}_\Omega : T_\Omega \to \mathsf{LegalStream}^\uparrow$$

such that

$$x \sim_\Omega y \to \mathsf{Code}_\Omega(x) \sim_{\mathsf{Code}} \mathsf{Code}_\Omega(y),$$

and the body ledger maps into the stream ledger:

$$\Lambda_\Omega(x) \to \Lambda_{\mathsf{Code}}(\mathsf{Code}_\Omega(x)).$$

Definition 9361.7 (Faithful and readable code)**.** *The code morphism is faithful when*

$$\mathsf{Code}_\Omega(x) \sim_{\mathsf{Code}} \mathsf{Code}_\Omega(y) \to x \sim_\Omega y.$$

It is readable when every encoded body row has a displayed readback:

$$\prod_{x : T_\Omega} \mathsf{Decode}_\Omega(\mathsf{Code}_\Omega(x), x).$$

Theorem 9361.8 (Faithful readable code excludes hidden body rows)**.** *If Code_Ω is faithful and readable, then two Ω-body rows with the same code are classified as the same body row.*

Proof. Assume

$$\mathsf{Code}_\Omega(x) \sim_{\mathsf{Code}} \mathsf{Code}_\Omega(y).$$

Faithfulness gives $x \sim_\Omega y$. Readability records that the public code is not a one-way compression: its source row is accompanied by a decode witness and a ledger route. Thus no further body distinction remains behind the displayed code. □

Corollary 9361.9 (Compiler display boundary)**.** *Compiler-style display is a faithful readable interface over licensed rows, not a hand table of external encodings. Its correctness claim is the preservation and reflection of classifiers together with the ledger route needed for readback.*

9361.3 Pairing by Involution

Definition 9361.10 (Pairing involution)**.** *For a licensed interface A, a pairing involution is a licensed endomorphism*

$$J_A : A \to A$$

with the classifier laws

$$x \sim_A y \to J_A(x) \sim_A J_A(y),$$

and

$$J_A(J_A(x)) \sim_A x.$$

Its fixed predicate is

$$\mathsf{Fix}_{J_A}(x) := J_A(x) \sim_A x,$$

and its apart-pair predicate is

$$\mathsf{ApartPair}_{J_A}(x) := J_A(x) \#_A x.$$

Theorem 9361.11 (Pairing orbits are typed)**.** *If J_A is a pairing involution, then every displayed row is either read through the fixed predicate or through a nontrivial two-step orbit whenever an apart-pair witness is supplied.*

Proof. If $\mathsf{Fix}_{J_A}(x)$ holds, the row is fixed by definition. If $\mathsf{ApartPair}_{J_A}(x)$ holds, then the first step $x \mapsto J_A(x)$ is classifier-apart from x, while the second step satisfies $J_A(J_A(x)) \sim_A x$. Thus the displayed orbit returns after two applications and is nontrivial at the first step. □

Principle 9361.12 (Pairing requires an involution)**.** *Pairing is not a global axiom over all generated objects. A layer supports pairing only after it exposes a licensed involution. Without such a morphism, the statement that rows occur in pairs has no typed BEDC meaning.*

Definition 9361.13 (Mirror row as pairing instance)**.** *At the complex zeta layer the mirror row*

$$J_\zeta(s) := 1 - \overline{s}$$

is a pairing involution on the displayed critical-strip carrier. Its fixed predicate is the critical-line row:

$$J_\zeta(s) \sim s \quad \Longleftrightarrow \quad \mathsf{OnCritLine}(s).$$

Theorem 9361.14 (Pairing-compatible generation preserves fixed rows)**.** *Let J_A be a pairing involution and let $\alpha_A : A \to A$ be a licensed generation morphism. If*

$$J_A(\alpha_A(x)) \sim_A \alpha_A(J_A(x))$$

for all x, then α_A sends fixed rows to fixed rows and sends a nontrivial paired orbit to a paired orbit or to a fixed row.

Proof. If $J_A(x) \sim_A x$, classifier preservation by α_A gives $\alpha_A(J_A(x)) \sim_A \alpha_A(x)$. The displayed commutation row then gives $J_A(\alpha_A(x)) \sim_A \alpha_A(x)$, so the image is fixed. For a nontrivial orbit, the partner of $\alpha_A(x)$ is classified with $\alpha_A(J_A(x))$. If the two images are apart, the orbit remains nontrivial; if they are classified together, it has collapsed to a fixed row. □

9361.4 Time as Generated Direction

Definition 9361.15 (One-step time row)**.** *The one-step time row on histories is the relation*

$$\mathsf{TimeStep}(h, h') := \mathsf{Ext}(h, \mathsf{b0}, h') \vee \mathsf{Ext}(h, \mathsf{b1}, h').$$

Its finite transitive closure is denoted

$$h \prec h'.$$

This is a generation order over BHist*, not an ambient clock.*

Definition 9361.16 (Local time stream)**.** *A local time stream for h is a finite chain*

$$\mathsf{Empty} = h_0 \prec h_1 \prec \cdots \prec h_n = h$$

whose adjacent rows are one-step time rows. It is a ledger of how a history is generated from the empty history.

Theorem 9361.17 (Time has forward determinacy by displayed mark)**.** *For a fixed source history h and fixed mark m, if*

$$\mathsf{Ext}(h, m, h_1), \qquad \mathsf{Ext}(h, m, h_2),$$

then $\mathsf{hsame}(h_1, h_2)$*.*

Proof. By Definition 9351.10, an Ext row with mark $\mathsf{b0}$ has result classified with the same $\mathsf{e0}$ extension of the source, and an Ext row with mark $\mathsf{b1}$ has result classified with the same $\mathsf{e1}$ extension of the source. Since the mark is fixed, both target rows lie in the same constructor class, hence are hsame. □

Theorem 9361.18 (Time direction is not pairing)**.** *The time relation $\prec$ is not a pairing involution.*

Proof. A pairing involution requires an endomorphism J with $J(J(h)) \sim h$. The time relation is generated by extension from a source history to a longer history, and Definition 9351.7 separates an empty history from its extensions and separates the two extension heads. Thus the one-step time row supplies forward generation data but does not supply a global constructive inverse endomorphism satisfying the involution law. □

Principle 9361.19 (Pairing and time have different sources)**.** *Rows may be paired only where an involution is licensed. Time is instead the finite generation direction exposed by* Ext *and accumulated by* Cont*. It is therefore not a failure of pairing, but a different structural interface over the same generated histories.*

9361.5 Interface Assembly

Theorem 9361.20 (Omega generation structure package)**.** *Assume an Ω generation diagram in* Lic_Ω*, a faithful readable code morphism, pairing involutions on the layers that claim paired behavior, and the history time row of Definition 9361.15. Then:*

encoded sameness reflects body sameness,

paired behavior is available exactly on involutive layers,

and

time is the generated continuation direction rather than an involution.

Proof. Encoded reflection is Definition 9361.8. Typed pairing is Definition 9361.11 together with Definition 9361.12. The time claim is Definition 9361.18. These three interfaces are compatible because they all live over licensed carrier, classifier, ledger, and NameCert rows. □

Principle 9361.21 (Connection to prime-shadow rigidity)**.** *The structural package does not prove any zero-rigidity theorem. It only ensures that the* Ω *body has a faithful display, that mirror fixedness is typed by an involution where available, and that the generated history direction is not confused with pairing. The RH-facing rigidity claim remains the prime-shadow and defect-zero interface of chapter 9332.*

Concretized in §8983: Omega generation boundary packets realize licensed interface, morphism, legal display, pairing, time-row, transport, replay, provenance, and local NameCert rows.

Chapter 9362

Mature Bridge and Omega Rigidity Interface Spine

L14 does not introduce another mathematical carrier beside the L0–L13 construction sequence. Its external-facing role is conservative bridge and audit:

$\mathsf{MatureSystemBridge}^{\uparrow}$, $\mathsf{StdModel}^{\uparrow}$, $\mathsf{MathlibBridge}^{\uparrow}$, $\mathsf{ConservativityReport}^{\uparrow}$, $\mathsf{VerificationReport}^{\uparrow}$.

Its internal-facing consequence is a disciplined synthesis surface: once the lower rows exist, one may package their compatible readouts as an Ω-tower and state the remaining RH rigidity target. The bridge layer interprets; the certificate layer admits.

9362.1 Conservative Bridge Boundary

Definition 9362.1 (Mature system)**.** *A mature system is an external reference environment, such as a set-theoretic presentation, a category-theory library, a homotopy-type-theory library, or a Lean/mathlib development. It is not part of the BEDC finite kernel and is not available as a source of BEDC constructors, classifiers, or theorem inhabitants.*

Definition 9362.2 (Conservative bridge)**.** *For a licensed BEDC object $A^{\uparrow}$ and a mature reference object A^{std}, a conservative bridge is a packet*

$$\mathsf{Bridge}(A^{\uparrow}, A^{\mathrm{std}}) = (F, G, \eta, \epsilon, \mathsf{LawPres}, \mathsf{NoReplace}, \mathsf{Ledger}).$$

Here F interprets BEDC rows externally, G is a partial reconstruction or decode route on a declared subdomain, η and ϵ are the displayed return witnesses, $\mathsf{LawPres}$ *records preservation of listed operations and relations,* $\mathsf{NoReplace}$ *states that the bridge does not replace internal proof obligations, and* Ledger *records unsupported fields and external dependencies.*

Theorem 9362.3 (Bridge does not generate)**.** *An external object and a bridge packet do not license $A^{\uparrow}$ unless the internal* $\mathsf{NameCert}(A^{\uparrow})$ *is already present.*

Proof. BEDC admits objects through internal carrier, classifier, stability, ledger, and naming rows. The mature object lies outside that generative surface. A bridge can interpret a licensed BEDC row, but if the internal $\mathsf{NameCert}$ is absent there is no BEDC carrier for the bridge to interpret. Therefore the external object cannot substitute for internal admission. □

Principle 9362.4 (Bridge interpretation boundary)**.** *Standard models may read out BEDC objects, and bridge packets may compare finite theorem lists. They do not authorize additional BEDC objects, additional BEDC classifiers, or additional BEDC proof terms.*

9362.2 Standard and Mathlib Bridges

Definition 9362.5 (Standard interpretation)**.** *A standard interpretation for $A^{\uparrow}$ is a map*

$$\mathsf{Interp}_A(-) : A^{\uparrow} \to A^{\mathrm{std}}$$

that respects the BEDC classifier and the listed operations. Exactness in the reverse direction is a separate field and is usually restricted to normalized or otherwise decoded subcarriers.

Theorem 9362.6 (Standard preservation)**.** *If a licensed BEDC object carries a theorem expressed by its classifier, and the bridge supplies the listed law-preservation rows, then the interpreted statement holds in the mature reference object.*

Proof. Apply the interpretation to both sides of the BEDC classifier theorem. Classifier preservation identifies the interpreted endpoints. Operation preservation rewrites the interpreted BEDC operations into the corresponding mature operations. The resulting equality is the mature-side statement named by the bridge ledger. □

Definition 9362.7 (Mathlib bridge)**.** *A mathlib bridge for $A^{\uparrow}$ is an external interoperability packet*

$$(\mathsf{TargetType}, \mathsf{Encode}, \mathsf{Decode}, \mathsf{Sound}, \mathsf{Complete}, \mathsf{LawMap}, \mathsf{AxiomAudit}).$$

The target type belongs to mathlib, the encode and decode rows are bridge-layer maps, soundness and completeness are scoped to the declared domain, LawMap *lists theorem correspondences, and* AxiomAudit *records use of host axioms and unsupported features.*

Theorem 9362.8 (Mathlib bridge conservativity)**.** *If $A^{\uparrow}$ already has a BEDC* NameCert*, the mathlib bridge is used only after internal construction, and no mathlib target type flows into the BEDC carrier, classifier, or proof term, then the bridge is conservative over the BEDC theorem surface.*

Proof. The internal proof DAG for a BEDC theorem contains only BEDC rows. The bridge is applied after the theorem is generated, as an interpretation layer. Removing the bridge leaves the same internal proof DAG and the same public theorem. Thus the bridge adds interoperability but not internal proof strength. □

9362.3 Conservativity and Verification Reports

Definition 9362.9 (Conservativity report)**.** *A conservativity report for a module M records*

$$(\mathsf{InternalDeps}, \mathsf{ExternalDeps}, \mathsf{AxiomUse}, \mathsf{HostLeakage}, \mathsf{BridgeOnly}, \mathsf{SubstitutionCheck}, \mathsf{Verdict}).$$

The verdict is conservative only when external references are bridge-only, host primitives do not appear on the public BEDC surface, forbidden axiom use is absent, and no mature object substitutes for an internal proof obligation.

Theorem 9362.10 (Bridge-only external references are conservative)**.** *If every external reference in M is bridge-only and the substitution check does not fail, then those references do not add internal BEDC theorem content.*

Proof. Bridge-only means the external system interprets already generated rows. The substitution check rules out using the external object as the source of a BEDC carrier, classifier, or proof. Deleting the external bridge therefore leaves the internal dependencies and proof obligations unchanged. □

Definition 9362.11 (Verification report)**.** *A verification report records the checker version, command, external imports, checked theorem names, admitted or pending statements, public API audit, and axiom audit. Its strict pass condition requires no admitted statements, no forbidden axiom dependencies, no public host leakage, marker resolution for every checked paper target, and a reproducible command line.*

Theorem 9362.12 (Strict pass yields checked certificate status)**.** *If a module's verification report has strict pass, then its checked markers may be used as checked certificate rows for the corresponding paper sites.*

Proof. Strict pass states that every checked marker resolves to a real declaration, no pending or admitted statement is being used as a proof, the axiom audit is clean, and the public API does not expose host primitives. Those are exactly the conditions needed for the checked marker to harden the corresponding certificate row. □

9362.4 Phase Discipline and Refutation Boundary

Definition 9362.13 (Development phase)**.** *A development phase is a packet*

$$(\mathsf{Name}, \mathsf{Scope}, \mathsf{ExitCriterion}, \mathsf{AllowedTargets}).$$

The phase controls which layer may be hardened and which higher-layer targets may appear only as scaffolds or consumers of the current closure work.

Principle 9362.14 (Lowest open layer priority)**.** *If a lower layer has an unmet exit criterion, work in a higher layer does not close that lower layer. Higher-layer rows may be written as scaffold or route targets only when their purpose is to expose the lower-layer obligation.*

Theorem 9362.15 (Higher rows do not close lower obligations)**.** *Let L_k have an unmet carrier, classifier, stability, ledger, or* $\mathsf{NameCert}$ *obligation. A checked row in a layer L_j with $j > k$ does not by itself close the missing L_k obligation.*

Proof. The dependency direction runs from lower layers to higher layers. A higher row may consume lower rows, but it does not provide the missing lower carrier, classifier, stability, ledger, or naming field unless it is explicitly a proof of that lower obligation. Therefore higher-layer success does not imply lower closure. □

Definition 9362.16 (Refutation boundary)**.** *BEDC accepts a displayed route $P \to \bot$ as a refutation of P. It does not accept a witness-free move from $\neg P \to \bot$ to P when P is an existential or otherwise witness-bearing claim outside a decidable fragment.*

Principle 9362.17 (RH contradiction discipline)**.** *An RH argument translated into BEDC must produce the proof function*

$$\prod_s (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnCritLine}(s))$$

or explicitly mark the step that leaves the constructive fragment.

9362.5 Omega Tower

Definition 9362.18 (Full constraint diagram)**.** *The Ω diagram is a family*

$$\mathcal{D}_\Omega = (R_i, \sim_i, C_i, \rho_{ji}, \alpha_i, \Lambda_i)_{i\in I}$$

where R_i is a layer carrier, $\sim_i$ its classifier, C_i its constraint predicate, $\rho_{ji} : R_j \to R_i$ a readback map, $\alpha_i : R_i \to R_i$ a self-similar operation at that layer, and Λ_i the local ledger.

Definition 9362.19 (Omega tower carrier)**.** *The tower carrier is the compatible family packet*

$$T_\Omega := \sum_{(r_i)\in\prod_i R_i} \left[\prod_i C_i(r_i)\right] \times \left[\prod_{j\to i} \rho_{ji}(r_j) \sim_i r_i\right] \times \Lambda_\Omega.$$

It is the $\mathsf{CatLimit}^\uparrow$ *reading of all layer readouts, constraints, readbacks, and ledgers at once.*

Definition 9362.20 (Mother operation)**.** *If the operations α_i commute with readback,*

$$\rho_{ji}(\alpha_j(r_j)) \sim_i \alpha_i(\rho_{ji}(r_j)),$$

then $\Omega_\infty : T_\Omega \to T_\Omega$ is defined by acting componentwise:

$$\Omega_\infty((r_i)_i) = (\alpha_i(r_i))_i.$$

Theorem 9362.21 (Projection law)**.** *For every layer projection $\pi_i : T_\Omega \to R_i$,*

$$\pi_i \circ \Omega_\infty \sim \alpha_i \circ \pi_i.$$

Proof. For $x = (r_i)_i$, the left side reads the i-component of $(\alpha_j(r_j))_j$, namely $\alpha_i(r_i)$. The right side first reads r_i and then applies α_i. The two displayed histories are the same component row, so the layer classifier identifies them. □

Principle 9362.22 (Boundary and fibre reading)**.** *Each projection $\pi_i : T_\Omega \to R_i$ is a boundary readout, and the fibre over r is the compatible bulk packet*

$$\mathrm{Fib}_i(r) := \sum_{x:T_\Omega} \pi_i(x) \sim_i r.$$

Thus the tower is a boundary, bulk, fibre, and readback geometry over the previously licensed layers.

9362.6 Omega-RH Rigidity

Definition 9362.23 (Zeta projection and zero atom)**.** *Let $\pi_\zeta : T_\Omega \to R_\zeta$ be the projection to the* $\mathsf{ZetaCont}^\uparrow$ *and* $\mathsf{ZetaZeros}^\uparrow$ *layer. The tower zero atom predicate is*

$$\mathsf{ZeroAtom}_\Omega(x) := \mathsf{ZetaZero}(\pi_\zeta(x)).$$

Definition 9362.24 (Omega counterexample surface)**.** *The tower counterexample surface is*

$$\mathsf{CounterRH}_\Omega := \sum_{x:T_\Omega} \left(\mathsf{ZeroAtom}_\Omega(x) \times \mathsf{InCritStrip}(\pi_\zeta(x)) \times \mathsf{ApartNonFix}(\pi_\zeta(x))\right).$$

It is a witness-bearing surface, not a decision procedure over all zeros.

Definition 9362.25 (Omega zero rigidity)**.** *The tower rigidity target is*

$$\mathsf{ZeroRigidity}_\Omega := \prod_{x:T_\Omega} (\mathsf{ZeroAtom}_\Omega(x) \times \mathsf{InCritStrip}(\pi_\zeta(x)) \to \mathsf{OnCritLine}(\pi_\zeta(x)))\,.$$

Theorem 9362.26 (Tower rigidity implies RH)**.** *If every legal zeta-zero packet is the π_ζ projection of some tower point and* $\mathsf{ZeroRigidity}_\Omega$ *holds, then the RH predicate holds.*

Proof. Let s carry $\mathsf{ZetaZero}(s)$ and $\mathsf{InCritStrip}(s)$. By the same-source projection hypothesis, choose $x : T_\Omega$ with $\pi_\zeta(x) \sim s$. Transport the zero and strip rows along this classifier comparison. Tower rigidity gives $\mathsf{OnCritLine}(\pi_\zeta(x))$, and classifier transport gives $\mathsf{OnCritLine}(s)$. □

Theorem 9362.27 (Defect incompatibility gives tower rigidity)**.** *Assume* $\mathsf{DefectZeroIncompatibility}$*, zero-ledger soundness, off-fixed-to-defect, and located-real collapse. Then* $\mathsf{ZeroRigidity}_\Omega$ *holds.*

Proof. Take $x : T_\Omega$ with a zero atom and critical-strip row, and write $s := \pi_\zeta(x)$. If s is apart from the critical line, the off-fixed-to-defect route supplies a nonempty finite prime-window defect. Defect-zero incompatibility makes $\zeta(s)$ apart from zero. Zero-ledger soundness from the zero atom gives the zero value, a contradiction. Hence there is no off-fixed apartness, and located-real collapse gives $\mathsf{OnCritLine}(s)$. □

9362.7 L14 Boundary

Theorem 9362.28 (Mature bridge and omega synthesis admission)**.** *If mature-system bridge, standard-model bridge, mathlib bridge, conservativity-report, and verification-report packets carry their declared fields, and if the Ω tower is given as the internal compatible* $\mathsf{CatLimit}^\uparrow$ *packet over L0–L13 readouts, then L14 is admitted as a bridge and audit layer, while* $\mathsf{ZeroRigidity}_\Omega$ *remains the internal RH-facing rigidity target.*

Proof. The bridge packets interpret licensed objects and carry no substitution route for internal certificates. Conservativity and verification reports audit external dependencies, host leakage, checked markers, and axiom use. Phase discipline prevents higher bridge rows from closing lower layer obligations. The Ω tower is internal because it is a compatible $\mathsf{CatLimit}^\uparrow$ packet over the already declared lower readouts. The projection law supplies the componentwise mother-operation readout, and the RH-facing theorems reduce RH to tower zero rigidity, with defect-zero incompatibility identified as the remaining rigidity target. □

Principle 9362.29 (Final boundary)**.** *At this surface the tower from b_0, b_1 through* $\mathsf{ZetaZeros}^\uparrow$ *and Ω_∞ can state RH as a rigidity theorem. The work is no longer a claim that the bridge proves RH; it is the explicit target that off-fixed zero atoms in the tower must be incompatible with the zeta zero ledger.*

Concretized in §8829: The Omega-rigidity ledger is realized as finite bridge-report, standard-readout, lower-layer NameCert, compatibility, audit, and no-replacement rows.

Chapter 9363

The Finite Kernel Is Already a Category

The checked categorical interface for BEDC is the unary-continuation $\mathsf{Cat}^{\uparrow}$ package over BHist. Its objects are unary histories, its displayed hom-carriers are histories f equipped with a continuation ledger from a to b, and morphism equality is the endpoint-stable classifier hsame on displayed hom histories. The prior capstone Computation as Continuation (chapter 9365) reads Cont as a small-step relation; the present chapter records the checked categorical reading available to the paper-Lean interface.

9363.1 Hom as unary continuation

Theorem 9363.1 (Unary-continuation category realizes the continuation interface)**.** *For unary histories $a, b, f : \mathsf{BHist}$, the checked hom-carrier $\mathsf{CategoryHomCarrier}(a, b, f)$ is equivalent to unary endpoints and a continuation morphism whose tail is classified as f:*

$$\mathsf{CategoryHomCarrier}(a, b, f) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(f) \wedge \exists m : \mathsf{ContinuationMorphis}$$

With this hom-carrier, the unary-continuation $\mathsf{Cat}^{\uparrow}$ package satisfies identity, binary composition closure, associativity of displayed composition, left and right unit laws, and endpoint-transport stability under hsame.

Proof. The equivalence is exactly Definition 61.317. Its forward direction unpacks a displayed hom-carrier into unary source, target, and morphism histories, then packages the continuation ledger as a $\mathsf{ContinuationMorphism}$ whose displayed tail is classified by hsame. Its reverse direction transports the continuation ledger carried by the morphism tail across the supplied hsame classifier and repacks the four fields of $\mathsf{CategoryHomCarrier}$.

The categorical law package is Definition 61.325. It supplies the empty hom-carrier identities, closure under displayed binary composition, associativity for the two displayed composites of a threefold composition, the left and right empty-composition laws, and preservation of hom-carriers after endpoint transport along hsame. □

Theorem 9363.2 (Category hom-carrier morphism induction)**.** *For fixed unary-continuation endpoints $a, b : \mathsf{BHist}$, any predicate on displayed morphism histories holds for every $\mathsf{CategoryHomCarrier}(a, b, f)$ if it holds at e and is preserved by one-step unary extension of the morphism history.*

Proof. Unpack $\mathsf{CategoryHomCarrier}(a, b, f)$. Its morphism field is exactly the unary-history witness for f, so unary-history induction over f proves the displayed predicate. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier morphism induction`

9363.2 Derived presentations over the unary category

Remark 9363.3 (FunctorUp and NatTransUp as bounded bridges)**.** $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ *(chapter 62) and* $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$ *(chapter 63) are prefix and component presentations over the checked unary-continuation category. They use the same BEDC carriers, classifiers, and continuation stability obligations at their own certificate sites. A standard category interface or mathlib bridge is not asserted here; that bridge remains governed by the roadmap bridge phase.*

Capstone

Theorem 9363.4 (Prefix functor identity naturality bridge)**.** *For histories* $p, a, b, f, left, right$: BHist*, assume* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{Cont}(\mathsf{e}, f, left)$*, and* $\mathsf{Cont}(f, \mathsf{e}, right)$*. Then the checked* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{NatTrans}^{\uparrow}$ *facts compose into the concrete identity-naturality bridge:*

1. *$left$ and $right$ are hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$*.*
2. $\mathsf{hsame}(left, right)$*.*
3. *each displayed hom-carrier reflects through the unary context exactness theorem to the original unprefixed hom carrier* $\mathsf{CategoryHomCarrier}(a, b, f)$*, with the left and right unit continuations identifying the displayed morphism with f.*
4. *for every unary object x :* BHist *and every empty identity continuation id with* $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, id)$*, the empty component supplied by the prefix functor identity law is a hom-carrier from* $\mathsf{append}(p, x)$ *to* $\mathsf{append}(p, x)$*.*

Proof. Apply Definition 63.47 to the unary prefix, the displayed hom-carrier $f : a \to b$, and the two empty-composite continuations. It gives the two displayed hom-carriers after prefixing by p and the classifier equality $\mathsf{hsame}(left, right)$.

Next apply Definition 61.51 to each displayed carrier. The same unary prefix p is already part of the hypotheses, so the readback is the corresponding unprefixed hom-carrier; the empty left and right unit continuations identify those displayed morphisms with the original carrier f.

Finally, apply Definition 62.36 at any unary object x. Its identity-closure statement supplies the empty component carrier over $\mathsf{append}(p, x)$, giving the component data used by the identity natural transformation. □

Remark 9363.5 (No new axioms)**.** *Definition 9363.1 introduces no kernel-level definition not already present in the checked* $\mathsf{Cat}^{\uparrow}$ *continuation bridge. The categorical reading is obtained by unfolding* $\mathsf{CategoryHomCarrier}$*,* $\mathsf{ContinuationMorphism}$*, and* hsame *transport, then applying the already checked stability package.*

Concretized in §471: Unary-continuation category packet over BHist, Cont, hsame, Pkg, and NameCert.

Concretized in §488: The unary-continuation category is realized as a concrete NameCert over BHist hom rows, hsame transport, Cont routes, and package provenance.

Concretized in §484: Finite kernel category carrier and NameCert packet.

Concretized in §480: The unary-continuation categorical reading is realized as a concrete NameCert surface over BHist, Cont, hsame, CategoryHomCarrier, Pkg, and local naming rows.

Concretized in §675: Prefix functor identity-naturality bridges are realized as finite BHist, CategoryHomCarrier, Cont, hsame, FunctorUp, NatTransUp, Pkg, and NameCert packets.

Concretized in §835: Finite-support packets over unary-continuation hom rows, Cont routes, hsame transport, Pkg provenance, and NameCert ledger rows.

Concretized in §1111: Unary-continuation endofunctors are realized as finite BHist packets preserving object rows, hom rows, identities, composition, hsame transport, Cont routes, Pkg provenance, and local NameCert data.

Concretized in §1179: Slice-category packets over a fixed unary-continuation base object are realized as finite BHist, CategoryHomCarrier, hsame, Cont, Pkg, and NameCert fibers.

Concretized in §7158: Kernel-category functorial transport is realized as a finite NameCert packet.

Chapter 9364

Lean Kernel Acceptance as the Canonical BEDC Audit Witness

BEDC repeatedly speaks of audit gates, generator-closure classifiers, NameCert verification, and executable ledgers. This chapter identifies the concrete mechanism underneath those phrases. Lean 4 kernel acceptance is the canonical audit witness for BEDC. It is not a metaphor for audit. It is the mechanism by which a proposed definition, theorem, inductive carrier, recursor use, and proof object enters the BEDC ledger as checked content.

The claim is intentionally strict. A BEDC audit unit has a generator, a classifier, a stamp, and a ledger position. At the Lean boundary, the generator is the term-building process that produces an elaborated term. The classifier is the kernel type-checking predicate, including definitional reduction and the side conditions for accepted recursion and inductive declarations. The stamp is kernel acceptance. The ledger position is the declaration entered into the environment and later visible to dependency queries such as `#print axioms`.

9364.1 The setup

The paper-side vocabulary is abstract because it must apply across many surfaces. A NameCert says that a public name is not decorative: it has a source, a classifier, a stability route, and a ledger row. A generator-closure classifier says which outputs remain inside the closure of the generator that licensed them. An audit gate is the executable refusal rule that rejects a record whose source, classifier, or dependency ancestry does not match the claim.

Lean kernel acceptance is the place where these abstract units become one machine event. When Lean accepts `def f := t`, the environment does not merely store text. It stores a name whose type has been checked against an elaborated term. When Lean accepts `theorem T : P := p`, the environment stores a name whose proof term has been checked as an inhabitant of P. When Lean accepts an `inductive` declaration, it stores a closed constructor signature and emits the eliminators permitted by that signature. When Lean accepts a structurally recursive definition, it has checked the recursion discipline that allows the definition to compute inside the accepted closure.

Thus the BEDC audit unit at the Lean boundary is:

$$\begin{aligned}\text{term builder} &\xrightarrow{\text{elaboration}} \text{kernel candidate}\\ &\xrightarrow{\text{type checking}} \text{accepted declaration}\\ &\xrightarrow{\text{environment entry}} \text{ledger row.}\end{aligned}$$

The same event has the BEDC reading:

$$\begin{aligned}\text{generator} &\xrightarrow{\text{classifier}} \text{stamped name}\\ &\xrightarrow{\text{ledger}} \mathsf{NameCert}\text{-visible content.}\end{aligned}$$

There is no second certification channel behind this map. The kernel event is the certification channel used by the Lean side of BEDC.

9364.2 What the Lean kernel does at acceptance

At acceptance time the kernel checks an elaborated object against its claimed type. This check includes substitution under binders, universe and type compatibility checks, definitional reduction, and the de Bruijn shifting that keeps binder structure coherent. For recursive definitions, Lean also requires an accepted recursion or termination account. For inductive declarations, Lean checks the constructor signature and its positivity discipline before the carrier and eliminators enter the environment.

The important feature for BEDC is that the kernel checks a generated object, not a prose intention. The object may have been produced by a tactic, by a direct term, by a recursive definition, or by elaboration of ordinary syntax. Those routes matter as generators, but the acceptance event is the kernel's judgment that the generated object has the claimed type inside the current environment.

If the check fails, there is no BEDC stamp at that name. If the check succeeds, the declaration is entered. This is the exact operational meaning of a checked Lean target in the manuscript. The paper marker is not a promise that the prose is convincing. It is a pointer to a declaration whose proof term or definition has passed kernel acceptance.

9364.3 NameCert stamping

Each kernel-acceptance event is one $\mathsf{NameCert}$ stamping at the relevant Lean name. The statement is definitional at the BEDC reading layer:

$$\begin{aligned}\text{classifier} &:= \text{kernel type-checking predicate,}\\ \text{generator} &:= \text{term builder that produced the candidate,}\\ \text{stamp} &:= \text{kernel acceptance of the candidate,}\\ \text{ledger} &:= \text{environment entry under the accepted name.}\end{aligned}$$

Thus a checked theorem name is not a theorem plus a separate certificate. Its checkedness is the certificate surface. A checked definition name is not a definition plus a separate permission slip. Its accepted type and body are the permission event. A checked inductive carrier is not a carrier plus an external authorization story. Its accepted constructor signature and emitted eliminator are the authorization.

BEDC uses NameCert vocabulary because it needs a uniform language for these events across paper-side certificates, recognized event flows, concrete instances, and Lean declarations. At the Lean boundary, the uniform language is realized by kernel acceptance. NameCert is the BEDC name for that accepted name-bearing closure event.

9364.4 `#print axioms` as audit ledger closure

A local kernel stamp is necessary, but BEDC also asks whether the accepted object imports any forbidden source through its ancestry. Lean exposes this by the command `#print axioms`. For a theorem, the command reports the transitive axiom dependencies of the declaration. This is not a philosophical summary. It is a dependency-ledger query over the checked environment.

BEDC's strict purity audit uses that query as a gate. The command `python3 lean4/scripts/bedc_ci.py axiom-purity -strict` builds the relevant Lean-side query surface, runs `#print axioms` over BEDC declarations, and rejects forbidden dependencies. The target report for a BEDC theorem is that the transitive axiom set is empty. The project-level witness is:

$$\texttt{\#print axioms}\, T = \texttt{(no axioms)}.$$

This is the strongest audit witness in the project because it does not stop at the local declaration surface. It asks whether the whole transitive closure of the theorem's proof ancestry contains an ungenerated commitment. A proof that looks structurally acceptable locally but depends on `Classical.choice`, `Quot.sound`, or `propext` fails this ledger reading. The problem is not that Lean cannot accept such a proof in a broader environment. The problem is that BEDC does not classify such ancestry as BEDC-internal closure.

9364.5 `lake build` as ledger instantiation

The command `cd lean4 && lake build` instantiates the Lean ledger as a single execution. Lean loads the project modules in dependency order, elaborates the declarations, invokes the kernel, checks recursion and inductive authorization at the appropriate sites, and enters accepted declarations into the environment. A successful build means the project has replayed the kernel-acceptance events required by the Lean modules.

In BEDC terms:

$$\begin{aligned} \texttt{lake build} &= \text{execution of all Lean-side classifier checks,} \\ \text{successful build} &= \text{all required kernel stamps obtained,} \\ \text{strict purity pass} &= \text{accepted stamps have clean transitive ancestry.} \end{aligned}$$

This is why build success is more than compilation hygiene for BEDC. It is the act by which the audit ledger is populated in a reproducible run. The manuscript can cite a Lean target only because the target is replayable through this mechanism and its dependency row can be queried afterward.

Concretized in §549: Kernel acceptance witness packets realize the Lean boundary as a finite NameCert surface over generated candidates, accepted declarations, environment ledger rows, axiom-query purity, and refusal boundaries.

9364.6 Closed-term substitution cross-link

chapter 9372 reads closed-term substitution as a self-compilation step. The local classifier is `ClosedAt`. The generator is `substitute`. The closure witness is that substitution at the closed boundary is inert:

$$\mathsf{ClosedAt}(d, t) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

The corresponding Lean target is

↪ variant `BEDC.MetaCIC.substitute closed`

with the companion shifting witness

↪ variant `BEDC.MetaCIC.shift closed`

The generator-closure classifier abstraction over both is

↪ variant `BEDC.MetaCIC.GeneratorClosureClassifier`

with concrete instances

↪ variant `BEDC.MetaCIC.closed under substitute classifier`

and

↪ variant `BEDC.MetaCIC.closed under shift classifier`

; each instance is a single kernel-accepted closure proof, and the audit witness reading of this chapter applies pointwise to every such instance.

The present chapter adds the outer audit fact. The theorem can function as a BEDC certificate only because the Lean kernel accepts its proof term. The paper-side reading says that `substitute` preserves the closed-term classifier on the relevant slice. The kernel-side event says that the proof term inhabiting that statement has been checked and entered into the ledger.

The layers are not competitors. The substitution theorem describes a generator-closure classifier instance inside MetaCIC. Kernel acceptance is the audit instrument that lets that description become checked BEDC content.

Concretized in §711: GeneratorAuditClosureUp realizes generator-closure classifier proofs as finite audit packets over generator rows, accepted closure witnesses, audit-query ledgers, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

9364.7 Recursor cross-link

chapter 9373 reads the recursor emitted for an accepted `inductive` declaration as an authorized generator. The present chapter pins down when that authorization occurs. It occurs at kernel time.

For an `inductive` declaration I, Lean checks the constructor signature and its positivity discipline. Only after that acceptance does the carrier enter the environment and only then are the eliminators associated with I available as legal generators. A later use of I.`rec` is checked again through its motive, branch package, and recursive structure. There is no kernel-external licensing authority that blesses the recursor after the fact.

The authorization map is:

$$\begin{aligned} \text{strictly positive accepted signature} &\Longrightarrow \text{closed carrier}, \\ \text{closed carrier} &\Longrightarrow \text{emitted recursor}, \\ \text{accepted recursor use} &\Longrightarrow \text{checked generator output}. \end{aligned}$$

This is precisely the NameCert stamping pattern. Constructor signature, eliminator generation, branch checking, and output type checking are not separate metaphors for BEDC authorization. They are the kernel events whose BEDC name is authorization.

9364.8 Why this is not an infinite regress

The kernel itself is not audited by another BEDC kernel inside the project. It is the project trust boundary. BEDC treats the Lean 4 kernel as its minimal trusted base, or MTB, for mechanized checking. This is an explicit commitment, not an epistemic gap hidden behind the paper vocabulary.

The same boundary appears in Lean projects generally and in neighboring proof assistant traditions such as Coq and Agda. A formal development trusts a small kernel to check proof objects, then audits what enters through that kernel. BEDC's difference is not that it has no trust boundary. The difference is that it names the boundary and refuses to let forbidden dependencies cross it silently into BEDC theorem content.

Thus the audit story stops at the kernel because the kernel is the instrument, not because BEDC has forgotten to audit one more layer. Below that boundary lie the implementation language, compiler, runtime, hardware, and human operational discipline. Those are real dependencies, but they are outside the BEDC theorem-content classifier. Their position is the MTB socket, not an unmarked Lean theorem.

9364.9 Kernel boundary as apophatic socket

Kernel acceptance is apophatic in the precise BEDC sense. The kernel does not give a positive philosophical interpretation of the generator output. It checks whether the generated object fits the claimed type under the accepted rules and environment. It certifies closure without explaining the content as a human meaning.

This is exactly why the kernel is the deepest BEDC socket. Downstream audit can ask whether a name resolves, whether a theorem builds, whether `#print axioms` is clean, whether a paper marker points to an existing target, and whether an inductive generator was accepted. All those questions delegate to the kernel boundary. The boundary says yes or no to a generated object. It does not convert that yes into a complete positive interpretation of the object.

The apophatic discipline is therefore not outside the engineering. The kernel is the last-resort socket through which BEDC receives checked formal content. It is a socket because BEDC does not internalize the kernel's own correctness as a BEDC theorem. It is apophatic because the acceptance event supplies the audit stamp while remaining silent about any interpretation not expressed in the checked type.

9364.10 Boundary: what kernel acceptance does not certify

Kernel acceptance is necessary for Lean-side BEDC content, but it is not sufficient for every paper-side interpretation. The kernel can accept:

```
theorem T : True := True.intro.
```

That declaration is a legal Lean theorem. The kernel has checked that `True.intro` inhabits `True`. It has not checked that the name `T` carries a deeper intended claim, nor that the surrounding

prose has earned a strong interpretation. This is the root of the placeholder pattern: a theorem with a weak statement can be fully accepted by the kernel while carrying little mathematical content.

BEDC therefore uses kernel acceptance as the canonical audit witness, while also keeping paper-side closure status, marker discipline, NameCert rows, and purity gates separate. The kernel tells the project that the generated Lean object is well typed and ledgered. The strict dependency audit tells the project whether the object imports forbidden ancestry. The manuscript must still state the scope of the claim accurately.

The final equation is:

$$\text{Lean kernel acceptance} = \text{canonical BEDC audit witness},$$
$$\text{canonical audit witness} \neq \text{unlimited semantic certification}.$$

The first line is the mechanism. The second line is the boundary that keeps the mechanism honest.

Concretized in §547: Lean kernel acceptance is realized as a finite audit-witness NameCert packet over generated candidate, accepted declaration, environment ledger, axiom-query, build replay, and non-escape rows.

Concretized in §742: Build replay is realized as a finite NameCert packet over generated candidate, accepted declaration, build command, replay result, axiom-query handoff, route, provenance, and non-escape rows.

Concretized in §557: Kernel acceptance traces are realized as finite NameCert packets over generated candidates, kernel classifier stamps, environment ledger rows, axiom-dependency reports, hsame transport, Cont replay, Pkg provenance, and local naming.

Concretized in §915: Kernel inductive acceptance is realized as finite carrier, constructor, audit-witness, transport, replay, provenance, and NameCert rows.

Concretized in §1096: Kernel inductive eliminator packets realize the emitted case-analysis rule surface as finite NameCert data.

Concretized in §654: Axiom query ledgers realize strict proof-ancestry audit as finite BHist rows for accepted declarations, axiom-query output, forbidden-dependency refusal, replay provenance, and local NameCert naming.

Concretized in §671: KernelSourceChannelLedgerUp realizes the source-channel audit ledger separating generated candidates, kernel stamps, accepted declarations, environment ancestry, and axiom-query refusal boundaries.

Concretized in §709: Closed generator audit packets realize kernel acceptance, authorized recursor output, and closed-term substitution as one finite MetaCIC NameCert audit surface.

Concretized in §1029: Kernel-normalization audit joins realize kernel acceptance, axiom-query ledger, normalization audit replay, and subject-reduction boundary rows as one finite NameCert packet.

Concretized in §6007: Kernel acceptance boundary packets realize generated candidates, kernel classifier stamps, ledger rows, purity queries, and refusal boundaries as finite NameCert data.

Concretized in §6356: Finite kernel budget flow is realized as BMark, BHist, Cont, Pkg, Gap, ledger, and NameCert rows.

Chapter 9365

Computation as Continuation: BEDC's Operational Semantics

Concretized in §469: Packages the continuation-as-small-step reading as a finite BHist trace certificate with readback and non-escape boundaries.

Concretized in §657: Continuation big-step packets realize terminal-read computation as a finite BHist NameCert surface over small-step traces, Cont replay, hsame transport, Pkg provenance, and local naming.

Concretized in §683: Continuation trace normal forms realize terminal Cont-replay computation as a finite NameCert packet with no host conversion or normalization oracle.

Concretized in §931: Unary Cont monoid packets realize the continuation-as-computation vision as a finite NameCert carrier over UnaryHistory, Cont composition, Empty unit, hsame transport, and non-escape ledger rows.

Concretized in §1026: Kleisli trace packets realize continuation computation as a finite BHist trace surface over ContinuationMonad, trace normal form, Category hom rows, hsame transport, Cont replay, Pkg provenance, and local naming.

The continuation primitive Cont of chapter 4 carries the formal content of a small-step transition relation. Reading $\mathsf{Cont}\, h\, m\, h'$ as "the history h steps via mark m to history h'" makes Cont a Plotkin-style structural operational semantics over the BEDC primitive layer. The chapter records the structural isomorphism between Cont and the small-step relations of standard programming-language semantics, and locates termination, big-step semantics, and behavioural equivalence as classifier-level notions already present in the kernel.

Concretized in §482: Continuation automata realize the Cont-as-small-step vision as a finite NameCert carrier with behaviour classifiers.

Concretized in §1016: Finite prefix automata realize bounded continuation-recognizer runs as finite BHist transition, prefix, endpoint, acceptance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1030: Finite prefix machines realize machine-specific bounded continuation prefixes as displayed BHist, Cont, Pkg, and NameCert rows.

Concretized in §597: Continuation termination packets realize the termination and behaviour-classifier reading of Cont traces as finite BHist, hsame, Pkg, and NameCert rows.

9365.1 Continuation as small-step transition

Theorem 9365.1 (Continuation-SOS correspondence)**.** *Let* Terminates *be a history carrier equipped with a naming certificate whose classifier is* Behavior*. If a small-step predicate* Step *is exactly the relation "there exists a continuation tail", then every step exposes such a tail and every continuation-tail witness reconstructs a step. The naming certificate transports termination along behaviour, each terminating history is behaviour-related to itself, and internal sameness from a terminating history supplies the canonical behaviour witness on the carrier.*

Lean4 ✓ `BEDC.Reflection.computation as continuation correspondence proof`

↪ variant `BEDC.Reflection.computation as continuation correspondence`

Capstone. The same continuation ledger has three concrete readings: as categorical hom-carrier data, as a unary prefix preorder witness, and as monoid append of continuation tails.

Theorem 9365.2 (Unary prefix category correspondence)**.** *For unary histories a, b, f, the displayed ledger* $\mathsf{CategoryHomCarrier}(a, b, f)$ *is exactly the datum that f is the unary continuation tail witnessing the concrete prefix comparison from a to b:* $\mathsf{UnaryHistory}(a)$*,* $\mathsf{UnaryHistory}(f)$*, and* $\mathsf{Cont}(a, f, b)$*. Under this correspondence,* e *is the identity tail, categorical composition is BEDC monoid append of tails, and* PreorderPrefixLE *transitivity is composition closure up to* hsame*.*

Proof. Unfold CategoryHomCarrier. A hom-carrier ledger already contains the unary source and morphism witnesses together with the continuation $\mathsf{Cont}(a, f, b)$, so the prefix witness is obtained by reading f as the displayed tail from a to b.

Conversely, the prefix tail data supplies the source and morphism witnesses directly, and unary continuation closure supplies the target witness. The continuation field is the displayed $\mathsf{Cont}(a, f, b)$, so the hom-carrier ledger is rebuilt. The concrete preorder form is the same tail display, with Definition 52.18 giving the canonical append presentation.

The identity case is Definition 61.2: the empty tail witnesses the identity continuation at every unary object. Composition uses Definition 61.19, while the preorder side uses Definition 52.28. The agreement of these two composites is the unit and associativity content of Definition 40.22, with comparison of resulting tails taken through hsame. □

Theorem 9365.3 (Suffixed composite middle object determinacy)**.** *If two composable unary hom-carrier ledgers display the middle object through the same right suffix q,*

$$\mathsf{CategoryHomCarrier}(a \cdot q, b \cdot q, f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b' \cdot q, c \cdot q, g),$$

and the displayed composite ledger has target $c \cdot q$, then the unsuffixed middle objects satisfy $\mathsf{hsame}(b, b')$*.*

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier append suffix comp middle object deterministic`

Proof. Apply middle-object determinacy to the suffixed composite ledgers. The result identifies $b \cdot q$ with $b' \cdot q$, and right cancellation of the common suffix gives $\mathsf{hsame}(b, b')$. □

Concretized in §485: Continuation monad packages the continuation-as-small-step reading as a certified Kleisli interface.

Theorem 9365.4 (Category hom-carrier empty morphism endpoint iff)**.** *For histories a, b, f : BHist, a hom-carrier ledger from a to b identifies the displayed morphism with the empty tail exactly when the two endpoints are internally the same:*

$$\mathsf{CategoryHomCarrier}(a, b, f) \Longrightarrow (f\mathsf{hsamee} \Longleftrightarrow a\mathsf{hsame}b).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier morphism empty endpoint iff`

Proof. Unfold the hom-carrier ledger to its continuation field $\mathsf{Cont}(a, f, b)$. If $f\mathsf{hsamee}$, determinacy against the right-unit continuation identifies b with a. Conversely, if $a\mathsf{hsame}b$, compare the displayed continuation with the endpoint equality and cancel the common left endpoint to obtain $f\mathsf{hsamee}$. □

Theorem 9365.5 (Cyclic continuation morphisms have empty tails)**.** *If the opposite continuation morphism records*

$$left : \mathsf{ContinuationMorphism}(a, b) \quad \textit{and} \quad right : \mathsf{ContinuationMorphism}(b, a)$$

both hold, then the endpoints are internally the same and both displayed tails are empty:

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism cycle tails empty`

Proof. Unpack the two continuation morphism records. The two exposed continuation ledgers form mutual extensions, one from a to b and one from b back to a. Kernel mutual-extension cancellation identifies the endpoints and forces both displayed tails to be the empty history. □

Theorem 9365.6 (Prefixed cyclic continuation morphisms have empty tails)**.** *If a prefixed continuation morphism and an opposite base continuation morphism record*

$$left : \mathsf{ContinuationMorphism}(\mathsf{append}(p, a), \mathsf{append}(p, b))$$

and

$$right : \mathsf{ContinuationMorphism}(b, a),$$

then the base endpoints are internally the same and both displayed tails are empty:

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism prefix cycle tails empty`

Proof. Unpack the prefixed morphism and cancel the common left continuation prefix in its exposed ledger. The resulting base continuation from a to b and the opposite continuation from b to a form mutual extensions, so kernel cancellation identifies the base endpoints and empties both tails. □

Theorem 9365.7 (Unary-suffix cyclic continuation morphisms have empty tails)**.** *If a unary suffix q is shared by the source and target of the forward continuation morphism, and the displayed forward tail is unary,*

$$left : \mathsf{ContinuationMorphism}(\mathsf{append}(a, q), \mathsf{append}(b, q))$$

and

$$right : \mathsf{ContinuationMorphism}(b, a),$$

then the base endpoints are internally the same and both displayed tails are empty:

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism unary suffix cycle tails empty`

Proof. Unpack the suffixed morphism. Unary commutation moves the shared suffix past the unary displayed tail, and suffix cancellation exposes a base continuation from a to b. This base continuation and the opposite continuation from b to a form mutual extensions, so kernel cancellation identifies the base endpoints and forces both displayed tails to be empty. □

Theorem 9365.8 (Zero-headed to one-headed continuation morphisms have no opposite cycle)**.** *For histories $a, r : \mathsf{BHist}$, no opposite pair can run from $\mathsf{E0}(a)$ to $\mathsf{E1}(r)$ and back:*

$$\mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r)) \to \mathsf{ContinuationMorphism}(\mathsf{E1}(r), \mathsf{E0}(a)) \to \bot.$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 e1 cycle absurd`

Proof. Cyclic continuation cancellation forces the forward displayed tail to be classified with e. The visible boundary theorem for a zero-headed source and one-headed target forces that same tail to be one-headed, hence not empty. The two conclusions contradict the generated-history separation theorem. □

Theorem 9365.9 (Triangular cyclic continuation morphisms have empty tails)**.** *If composable continuation morphisms record*

$$left : \mathsf{ContinuationMorphism}(a, b),$$

$$right : \mathsf{ContinuationMorphism}(b, c),$$

$$back : \mathsf{ContinuationMorphism}(c, a),$$

then all three displayed tails are empty and the adjacent endpoints collapse:

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(back), \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism triangle cycle tails empty`

Proof. Compose the first two continuation morphisms. The composite morphism from a to c and the displayed return morphism from c to a form a mutual extension cycle. Kernel cancellation forces the composite tail and the return tail to be empty. Empty append inversion splits the composite tail into empty left and right component tails. Comparing the component continuation ledgers with the right-unit continuations at a and b identifies the adjacent endpoints. □

Theorem 9365.10 (Prefixed triangular cyclic continuation morphisms have empty tails)**.** *If continuation morphisms record*

$$left : \mathsf{ContinuationMorphism}(\mathsf{append}(p, a), \mathsf{append}(p, b)),$$

$$middle : \mathsf{ContinuationMorphism}(b, c),$$

and

$$back : \mathsf{ContinuationMorphism}(c, a),$$

then all three displayed tails are empty and the adjacent base endpoints collapse:

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(middle), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(back), \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism prefix triangle cycle tails empty`

Proof. Cancel the common prefix in the first continuation ledger, then compose the base left ledger with the middle ledger. The composite and the back ledger form a mutual-extension cycle, so the composite tail and the back tail are empty. Empty append inversion splits the composite tail into the left and middle tails, and deterministic comparison with right-unit ledgers identifies the adjacent endpoints. □

Theorem 9365.11 (Visible e1 continuation morphism tail cases)**.** *If a continuation morphism runs from* $\mathsf{E1}(a)$ *to* $\mathsf{E1}(r)$*, then its tail is either empty and the payload endpoints satisfy* $a\mathsf{hsame}r$*, or the tail itself has visible one-headed form* $\mathsf{E1}(k)$ *and exposes a continuation* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e1 target tail cases`

Proof. Apply the one-headed source tail split for continuation morphisms. The empty-tail branch identifies $\mathsf{E1}(r)$ with $\mathsf{E1}(a)$, and constructor injectivity gives $a\mathsf{hsame}r$. The zero-headed tail branch would identify an $\mathsf{E1}$ target with an $\mathsf{E0}$ target, contradicting kernel no-confusion. In the one-headed branch, constructor injectivity reindexes the displayed target payload, leaving the tail equality and the continuation witness $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. □

Theorem 9365.12 (Visible e1 continuation morphism nonempty tail extraction)**.** *If a continuation morphism runs from* $\mathsf{E1}(a)$ *to* $\mathsf{E1}(r)$ *and its displayed tail is not* e*, then the tail has one-headed form* $\mathsf{E1}(k)$ *and exposes a continuation witness* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e1 target nonempty tail cases`

Proof. Inspect the concrete tail of the continuation morphism. The empty branch is excluded by the nonempty hypothesis. A zero-headed tail would force the one-headed target to coincide with a zero-headed result of the step inversion, contradicting kernel no-confusion. The remaining one-headed tail branch gives the displayed core continuation after constructor injectivity on the target. □

Theorem 9365.13 (Visible e1 continuation morphism tail core deterministic)**.** *For two continuation morphisms from* $\mathsf{E1}(a)$ *to* $\mathsf{E1}(r)$*, if their displayed tails are respectively* $\mathsf{E1}(k)$ *and* $\mathsf{E1}(l)$ *up to* hsame*, then the tail cores satisfy* $k\mathsf{hsame}l$*.*

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e1 target tail core deterministic`

Proof. Continuation morphism tail determinacy compares the two displayed tails. Compose that comparison with the two one-headed tail identifications, then use constructor injectivity for $\mathsf{E1}$ to obtain the core comparison. □

Theorem 9365.14 (Cyclic hom-carriers have empty tails)**.** *If the opposite hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g)$$

both hold, then the endpoints are internally the same and both morphism tails are empty:

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier cycle tails empty`

Proof. Each hom-carrier exposes a concrete continuation ledger, one from a to b through f and one from b back to a through g. Mutual continuation cancellation identifies the endpoints and forces both tails to be the empty history. □

Theorem 9365.15 (Cyclic hom-carrier composites have empty result)**.** *If the opposite hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g)$$

both hold, then every displayed continuation composite

$$\mathsf{Cont}(f, g, fg)$$

has empty result:

$$\mathsf{hsame}(fg, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp cycle result empty`

Proof. The two hom-carrier ledgers expose reciprocal continuations $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, a)$. Mutual continuation cancellation forces $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$. Transport the displayed composite $\mathsf{Cont}(f, g, fg)$ across these two tail equalities to a continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, fg)$, then read back the empty result by the continuation left-unit law. □

Theorem 9365.16 (Cyclic hom-carriers expose identity carriers)**.** *If the opposite hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g)$$

both hold, then the two endpoints carry empty-tail identities and the endpoints are internally the same:

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier cycle identity carriers`

Proof. Apply the cyclic tail theorem to obtain $\mathsf{hsame}(a, b)$ together with $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$. Transport the two displayed hom-carriers along these endpoint and tail equalities. The forward carrier becomes the empty-tail identity carrier at a, and the reverse carrier becomes the empty-tail identity carrier at b. □

Theorem 9365.17 (Split-mono witnesses have empty cyclic boundary)**.** *If $f : a \to b$ and $g : b \to a$ are category hom-carriers, $i : a \to a$ is an identity hom-carrier, $\mathsf{Cont}(f, g, c)$ displays the composite, and $c : a \to a$ is also a category hom-carrier, then the whole cyclic boundary collapses:*

$$f\mathsf{hsamee} \wedge g\mathsf{hsamee} \wedge i\mathsf{hsamee} \wedge c\mathsf{hsamee} \wedge a\mathsf{hsame}b.$$

Proof. The two opposite hom-carriers give the cyclic tail theorem, so $a\mathsf{hsame}b$ and both displayed tails are empty. Transport the forward hom-carrier across the endpoint and tail comparisons to obtain the empty identity carrier at a. Category hom-carrier determinacy compares the displayed identity with that empty carrier. The composite continuation transports across the empty tail comparisons to a continuation from e by e, while the supplied endomorphism carrier identifies this composite with the category composition carrier; continuation determinacy reads the composite as empty. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier split mono witness empty boundary`

Theorem 9365.18 (Triangular cyclic hom-carriers have empty tails)**.** *If the hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f),$$

$$\mathsf{CategoryHomCarrier}(b, c, g),$$

$$\mathsf{CategoryHomCarrier}(c, a, h)$$

all hold, then all three morphism tails are empty and the adjacent endpoints collapse:

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier triangle cycle tails empty`

Proof. The first two hom-carriers expose continuation ledgers $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, c)$. Compose them into a single continuation $\mathsf{Cont}(a, \mathsf{append}(f, g), c)$. This composite continuation and the return ledger $\mathsf{Cont}(c, h, a)$ form a mutual extension cycle, so kernel cancellation forces $\mathsf{append}(f, g)\mathsf{hsame}\mathsf{e}$ and $h\mathsf{hsame}\mathsf{e}$. Empty append inversion splits the composite tail into $f\mathsf{hsame}\mathsf{e}$ and $g\mathsf{hsame}\mathsf{e}$. Comparing the original component ledgers with the right-unit continuations at a and b gives $a\mathsf{hsame}b$ and $b\mathsf{hsame}c$. □

Theorem 9365.19 (Triangular cyclic hom-carriers have empty hom exactness)**.** *If the hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f),$$

$$\mathsf{CategoryHomCarrier}(b, c, g),$$

$$\mathsf{CategoryHomCarrier}(c, a, h)$$

all hold, then the empty morphism itself is displayed on each side of the triangle, and the adjacent endpoints collapse:

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, c, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(c, a, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier triangle cycle empty hom exactness`

Proof. The triangular tail theorem gives $f\mathsf{hsame}\mathsf{e}$, $g\mathsf{hsame}\mathsf{e}$, and $h\mathsf{hsame}\mathsf{e}$, together with the adjacent endpoint equalities. Hom-carrier transport moves the three displayed carriers across those tail equalities, yielding empty hom-carriers on the original triangle edges. □

Theorem 9365.20 (Visible one-step morphisms have no triangular cycle)**.** *If the first hom-carrier in a triangular cycle has visible one-headed morphism tail* $\mathsf{e}_1(k)$*,*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}_1(k)),$$

and the remaining hom-carriers run from b *to* c *and from* c *back to* a*, then the three ledgers are contradictory:*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}_1(k)) \to \mathsf{CategoryHomCarrier}(b, c, g) \to \mathsf{CategoryHomCarrier}(c, a, h) \to \bot.$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 morphism triangle cycle absurd`

Proof. Each hom-carrier exposes its underlying continuation ledger. The first two continuations compose into a tail $\mathsf{append}(\mathsf{e}_1(k), g)$, and the third continuation returns to the source. The kernel triangular continuation exclusion says this visible left component cannot be part of such a cycle. □

Theorem 9365.21 (Triangular cyclic hom-carriers expose identity carriers)**.** *If the hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f),$$

$$\mathsf{CategoryHomCarrier}(b, c, g),$$

$$\mathsf{CategoryHomCarrier}(c, a, h)$$

all hold, then the three objects carry empty-tail identities and the adjacent endpoints collapse:

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(c, c, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier triangle cycle identity carriers`

Proof. Apply the triangular cyclic tail theorem. The first carrier is transported along $b\mathsf{hsame}a$ and $f\mathsf{hsame}\mathsf{e}$ to become the empty identity carrier at a. The second is transported along $c\mathsf{hsame}b$ and $g\mathsf{hsame}\mathsf{e}$ to become the empty identity carrier at b. The return carrier is transported along the composite endpoint equality $a\mathsf{hsame}c$ and $h\mathsf{hsame}\mathsf{e}$ to become the empty identity carrier at c. □

Theorem 9365.22 (Unary-prefix triangle cyclic hom-carriers expose identity carriers)**.** *If the first hom-carrier has a shared unary prefix,*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f),$$

and the remaining two hom-carriers close the triangle,

$$\mathsf{CategoryHomCarrier}(b, c, g) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(c, a, h),$$

then the base endpoints carry empty-tail identities and adjacent endpoints are internally the same:

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(c, c, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Prefix exactness strips the shared unary prefix from the first hom-carrier, leaving a base carrier from a to b. The ordinary triangular cyclic identity-carrier theorem applies to this base carrier and the two remaining hom-carriers. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary prefix triangle cycle identity carriers`

Theorem 9365.23 (Unary prefix category isomorphism collapse)**.** *If the opposite unary hom-carrier ledgers*

$$\mathsf{CategoryHomCarrier}(a, b, f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g)$$

both hold, then they expose bidirectional prefix comparison,

$$\mathsf{PreorderPrefixLE}(a, b) \quad \textit{and} \quad \mathsf{PreorderPrefixLE}(b, a),$$

endpoint sameness

$$\mathsf{hsame}(a, b),$$

and monoid-unit morphism tails

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(f, \mathsf{e}) \quad \textit{and} \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(g, \mathsf{e}).$$

Proof. Read the two prefix witnesses from the hom-carrier ledgers using Definition 9365.2. The forward ledger is the displayed unary continuation tail from a to b, and the reverse ledger is the displayed unary continuation tail from b to a.

Apply Definition 52.30 to the two exposed prefix comparisons. This gives the endpoint identification $\mathsf{hsame}(a, b)$ required by the categorical core.

Apply Definition 9365.14 to the opposite hom-carrier pair. It gives $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$. The hom-carrier fields provide $\mathsf{UnaryHistory}(f)$ and $\mathsf{UnaryHistory}(g)$, while Definition 40.25 provides the empty-history carrier and the classifier package. Repacking these data gives the two classifier conclusions.

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(f, \mathsf{e})$$

and

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(g, \mathsf{e}).$$

□

The roadmap ten-field fit is: traditional target skeletal core of the prefix category; BEDC source $\mathsf{CategoryHomCarrier}$, $\mathsf{PreorderPrefixLE}$, and $\mathsf{MonoidHistoryClassifier}$; carrier unary BHist; classifier hsame; stability cyclic isomorphism collapse; ledger opposite hom tails; core theorem Definition 9365.23; exactness bidirectional prefix comparison collapses to identity; standard bridge preorder-enriched thin category; status $\mathsf{ProofTarget}$.

Theorem 9365.24 (Unary-suffix cyclic hom-carriers have empty tails)**.** *If a unary suffix q is shared by the source and target of the forward hom-carrier and an opposite base hom-carrier returns from b to a,*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(a, q), \mathsf{append}(b, q), f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g),$$

then the base endpoints are internally the same and both morphism tails are empty:

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary suffix cycle tails empty`

Proof. The suffix exactness theorem strips the shared unary suffix from the forward hom-carrier, exposing a base hom-carrier from a to b with the same tail f. The two base hom-carriers now form an ordinary opposite cycle, so mutual continuation cancellation identifies the endpoints and forces both tails to be empty. □

Theorem 9365.25 (Unary-suffix cyclic hom-carriers expose identity carriers)**.** *If*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(a, q), \mathsf{append}(b, q), f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(b, a, g)$$

both hold, then the base endpoints carry empty-tail identities and the endpoints are internally the same:

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary suffix cycle identity carriers`

Proof. Strip the shared unary suffix from the forward hom-carrier. The resulting base cycle is an ordinary cyclic hom-carrier pair, so cyclic identity-carrier exposure transports the two tails to empty identity carriers at a and b. □

Theorem 9365.26 (Unary-context cyclic hom-carriers have empty tails)**.** *If the forward hom-carrier has a shared unary prefix and the opposite hom-carrier has a shared unary suffix,*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p,a),\mathsf{append}(p,b),f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{append}(b,q),\mathsf{append}(a,q),g),$$

then the base endpoints are internally the same and both morphism tails are empty:

$$\mathsf{hsame}(a,b) \wedge \mathsf{hsame}(f,\mathsf{e}) \wedge \mathsf{hsame}(g,\mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary context cycle tails empty`

Proof. Prefix exactness strips the shared unary prefix from the forward hom-carrier, leaving a base carrier from a to b. Suffix exactness strips the shared unary suffix from the opposite hom-carrier, leaving a base carrier from b to a. The ordinary cyclic tail theorem then identifies the endpoints and forces the two morphism tails to be empty. □

Theorem 9365.27 (Unary-context cyclic hom-carriers expose identity carriers)**.** *If the forward hom-carrier has a shared unary prefix and the opposite hom-carrier has a shared unary suffix,*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p,a),\mathsf{append}(p,b),f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{append}(b,q),\mathsf{append}(a,q),g),$$

then the base endpoints carry empty-tail identities and are internally the same:

$$\mathsf{CategoryHomCarrier}(a,a,\mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b,b,\mathsf{e}) \wedge \mathsf{hsame}(a,b).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary context cycle identity carriers`

Proof. Prefix exactness strips the shared unary prefix from the forward hom-carrier, leaving a base carrier from a to b. Suffix exactness strips the shared unary suffix from the opposite hom-carrier, leaving a base carrier from b to a. The ordinary cyclic identity-carrier theorem then exposes the two empty-tail identity carriers and the comparison $a\mathsf{hsame}b$. □

Theorem 9365.28 (Visible one-step morphisms have no opposite cycle)**.** *If the forward hom-carrier has a visible one-headed morphism tail* $\mathsf{e}_1(k)$ *and an opposite hom-carrier returns from* b *to* a*, then the two ledgers are contradictory:*

$$\mathsf{CategoryHomCarrier}(a,b,\mathsf{e}_1(k)) \to \mathsf{CategoryHomCarrier}(b,a,g) \to \bot.$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 morphism cycle absurd`

Proof. The cyclic hom-carrier theorem forces every tail in an opposite pair to be empty. Applying it to the visible tail $\mathsf{e}_1(k)$ gives $\mathsf{hsame}(\mathsf{e}_1(k),\mathsf{e})$, contradicting the kernel separation theorem for one-headed histories and the empty history. □

Concretized in §1025: Continuation bisimulation packets realize symmetric behavioral equivalence of finite Cont traces as displayed forward and backward simulation rows.

Concretized in §4085: Continuation replay packets realize finite Cont execution as displayed step rows, endpoint transport, replay provenance, and local NameCert data.

Concretized in §7361: Regular-language continuation packets realize bounded automaton recognition as finite Cont replay, acceptance, classifier, transport, provenance, and NameCert rows.

Chapter 9366

Classifier-Level Information: Computation, Unfolding, and Discovery

chapter 9365 reads computation through Cont-generated continuation traces, while chapter 9367 reads checking through certificate-governed classifier membership. The present chapter separates two measurements that are often conflated: growth of the displayed record and change of the certified classifier by which that record is read. In BEDC, a computation may extend a trace without changing its governing NameCert, whereas discovery requires a certificate-governed classifier shift with an exposed ledger.

9366.1 Certified classifier states

Definition 9366.1 (Certified classifier state)**.** *A certified classifier state is a tuple*

$$\mathcal{T} = (S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}}, C_{\mathcal{T}}, N_{\mathcal{T}})$$

where

$$S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}} : \mathsf{Hist} \to \mathsf{Prop}, \qquad C_{\mathcal{T}} : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop},$$

and

$$N_{\mathcal{T}} : \mathsf{SemanticNameCert}(S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}}, C_{\mathcal{T}}).$$

The fields are respectively the source specification, pattern specification, ledger policy, classifier specification, and semantic naming certificate.

Definition 9366.2 (Record and configuration)**.** *Relative to a certified classifier state $\mathcal{T}$, a record is a finite family R of histories. An epistemic configuration is a pair $(\mathcal{T}, R)$.*

Definition 9366.3 (External record size)**.** *Let the external display length of histories be defined by*

$$|\mathsf{e}| = 0, \qquad |\mathsf{E0}(h)| = |h| + 1, \qquad |\mathsf{E1}(h)| = |h| + 1.$$

For a finite record R, set

$$|R| = \sum_{h \in R} |h|.$$

This size is a metalanguage audit measure. It is not a primitive object of the finite kernel.

9366.2 Computation as record growth

Definition 9366.4 (Unfolding)**.** *An unfolding step from $(\mathcal{T}, R)$ to $(\mathcal{T}, R')$ is a transition for which there are histories h, k, r such that $h \in R$, $\mathsf{Cont}(h, k, r)$, and R' is obtained from R by adjoining r. It is written*

$$(\mathcal{T}, R) \rightsquigarrow_{\mathsf{unf}} (\mathcal{T}, R').$$

Definition 9366.5 (Mechanical computation)**.** *A transition*

$$(\mathcal{T}, R) \longrightarrow (\mathcal{T}', R')$$

is a mechanical computation when $\mathcal{T}' = \mathcal{T}$ and every history in R' not already displayed in R is generated from histories of R by finitely many Ext *or* Cont *steps. It is written*

$$(\mathcal{T}, R) \rightsquigarrow_{\mathsf{comp}} (\mathcal{T}, R').$$

Definition 9366.6 (Classifier equivalence)**.** *Certified classifier states*

$$\mathcal{T} = (S, P, L, C, N), \qquad \mathcal{T}' = (S', P', L', C', N')$$

are classifier-equivalent on their common source scope when

$$\forall h, k : \mathsf{Hist},\ S(h) \wedge S(k) \wedge S'(h) \wedge S'(k) \Rightarrow \big(C(h, k) \leftrightarrow C'(h, k)\big).$$

This relation is written $\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}'$.

Definition 9366.7 (Classifier shift)**.** *The classifier-shift value of a transition is the boolean audit quantity*

$$\Delta_{\mathsf{cls}}\big((\mathcal{T}, R), (\mathcal{T}', R')\big) = 0$$

exactly when $\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}'$, and is 1 exactly when this equivalence fails.

Theorem 9366.8 (Record growth does not force classifier shift)**.** *If*

$$(\mathcal{T}, R) \rightsquigarrow_{\mathsf{unf}} (\mathcal{T}, R')$$

and $R' \neq R$, then the record has grown while

$$\Delta_{\mathsf{cls}}\big((\mathcal{T}, R), (\mathcal{T}, R')\big) = 0.$$

Proof. The unfolding relation keeps the certified classifier state fixed. Hence $\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}$ by reflexivity of the displayed classifier on the common source scope. The definition of Δ_{cls} gives value 0. The condition $R' \neq R$ records displayed growth only; it adds no source specification, classifier specification, ledger policy, or naming certificate. □

Theorem 9366.9 (Mechanical computation is not discovery)**.** *No mechanical computation whose target state is the same certified classifier state can be a structural discovery transition in the sense of Definition 9366.12.*

Proof. Mechanical computation has $\mathcal{T}' = \mathcal{T}$, so $\Delta_{\mathsf{cls}} = 0$. Structural discovery requires $\Delta_{\mathsf{cls}} = 1$. These requirements are incompatible. □

Theorem 9366.10 (Finite mechanical chains preserve the classifier)**.** *For a finite chain*

$$(\mathcal{T}_0, R_0), (\mathcal{T}_1, R_1), \ldots, (\mathcal{T}_n, R_n)$$

with

$$(\mathcal{T}_i, R_i) \rightsquigarrow_{\mathsf{comp}} (\mathcal{T}_{i+1}, R_{i+1}) \quad \textit{for every } i < n,$$

one has $\mathcal{T}_n = \mathcal{T}_0$ *and*

$$\Delta_{\mathsf{cls}}\big((\mathcal{T}_0, R_0), (\mathcal{T}_n, R_n)\big) = 0.$$

Proof. Induct on the length of the chain. The empty chain is immediate. In the successor case, the induction hypothesis gives $\mathcal{T}_n = \mathcal{T}_0$, and the last mechanical computation gives $\mathcal{T}_{n+1} = \mathcal{T}_n$. Therefore $\mathcal{T}_{n+1} = \mathcal{T}_0$, and the classifier-shift value is 0 by Definition 9366.7. □

Corollary 9366.11 (Output novelty is not classifier discovery)**.** *The fact that a displayed history* r *belongs to* R' *but not to* R *does not by itself imply discovery.*

Proof. If the transition is mechanical, the classifier state is unchanged by Definition 9366.10. The novelty of r is record novelty, not classifier-level novelty. □

9366.3 Discovery as classifier shift

Definition 9366.12 (Structural discovery transition)**.** *A transition*

$$(\mathcal{T}, R) \longrightarrow (\mathcal{T}', R')$$

is a structural discovery transition when all of the following hold:

$$\begin{gathered}\mathcal{T}' \textit{ is a certified classifier state,}\\ \Delta_{\mathsf{cls}}\big((\mathcal{T}, R), (\mathcal{T}', R')\big) = 1,\\ \exists h : \mathsf{Hist},\ L_{\mathcal{T}'}(h),\\ \exists h \in R',\ S_{\mathcal{T}'}(h) \quad \textit{or} \quad \exists h, k \in R',\ C_{\mathcal{T}'}(h, k).\end{gathered}$$

It is written

$$(\mathcal{T}, R) \rightsquigarrow_{\mathsf{disc}} (\mathcal{T}', R').$$

Theorem 9366.13 (Discovery carries a naming certificate)**.** *If*

$$(\mathcal{T}, R) \rightsquigarrow_{\mathsf{disc}} (\mathcal{T}', R')$$

and

$$\mathcal{T}' = (S', P', L', C', N'),$$

then

$$N' : \mathsf{SemanticNameCert}(S', P', L', C').$$

Proof. Structural discovery requires $\mathcal{T}'$ to be a certified classifier state. The fifth field of such a state is precisely the displayed semantic naming certificate. □

Theorem 9366.14 (Certified classifier states carry ledger witnesses)**.** *For every certified classifier state*

$$\mathcal{T} = (S, P, L, C, N),$$

there exists a history h *with* $L(h)$*.*

Proof. The semantic naming certificate contains a core $\mathsf{NameCert}(S, C)$. The core certificate supplies a source witness h with $S(h)$. Applying the ledger soundness field of N to this source witness gives $L(h)$. □

Corollary 9366.15 (Unledgered compression is not discovery)**.** *A transition that claims compression but exposes no witness*

$$\exists h : \mathsf{Hist},\ L_{\mathcal{T}'}(h)$$

is not a structural discovery transition.

Proof. This is the contrapositive of Definition 9366.14 applied to the target classifier state required by Definition 9366.12. □

Definition 9366.16 (Cost observer and positive discovery)**.** *Let $(Q, \leq, +, 0)$ be an ordered commutative monoid. A cost observer assigns to each certified classifier state $\mathcal{T}$ a map*

$$\mu_{\mathcal{T}} : \mathcal{P}_{\mathsf{fin}}(\mathsf{Hist}) \to Q$$

and assigns to each state transition a certificate cost

$$\kappa(\mathcal{T} \to \mathcal{T}') \in Q.$$

A structural discovery transition is positive for a finite family $X \subseteq R'$ when

$$\mu_{\mathcal{T}'}(X) + \kappa(\mathcal{T} \to \mathcal{T}') < \mu_{\mathcal{T}}(X).$$

Theorem 9366.17 (Positive discovery implies structural discovery)**.** *Every positive discovery is a structural discovery transition.*

Proof. Positivity is defined only after the structural discovery clauses have been met; the displayed inequality adds a cost comparison and does not replace the classifier-shift, certificate, or ledger requirements. □

Corollary 9366.18 (Faster unfolding is not automatically discovery)**.** *A shorter, faster, or more direct continuation route under the same certified classifier state is not automatically a discovery.*

Proof. Such a route changes the external method or the displayed record. Unless it changes the certified classifier state up to Definition 9366.6, its classifier-shift value is 0. □

9366.4 Generators and presentation changes

Definition 9366.19 (Certified generator)**.** *Within a certified classifier state $\mathcal{T}$, a certified generator is an externally indexed family $G : I \to \mathsf{Hist}$ whose use is authorized by $N_{\mathcal{T}}$ and for which every $G(i)$ is obtained from a certified seed by a finite* Cont *tail.*

Theorem 9366.20 (Fixed generator theorem)**.** *Let $G : I \to \mathsf{Hist}$ be a certified generator in $\mathcal{T}$. If the transition from R to R' only adjoins an output of G and keeps the certified classifier state equal to $\mathcal{T}$, then*

$$\Delta_{\mathsf{cls}}((\mathcal{T}, R), (\mathcal{T}, R')) = 0.$$

Proof. The generator output is already licensed by $N_{\mathcal{T}}$. Since the transition keeps $\mathcal{T}$ fixed, it is a record extension under the same certificate. The classifier-shift value is therefore 0. □

Corollary 9366.21 (Digit extension is unfolding)**.** *If a digit-prefix generator for a constant has already been authorized in a certified classifier state, producing a longer prefix by the same generator is unfolding rather than classifier discovery.*

Proof. Apply Definition 9366.20 to the prefix generator. The visible prefix changes, but the governing classifier state does not. □

Definition 9366.22 (Presentation weakening)**.** *Let*

$$\mathcal{T} = (S, P, L, C, N), \qquad \mathcal{T}' = (S, P', L', C, N')$$

have the same source specification and classifier specification. If

$$\forall h : \mathsf{Hist},\ P(h) \to P'(h), \qquad \forall h : \mathsf{Hist},\ L(h) \to L'(h),$$

then $\mathcal{T}'$ is a presentation weakening of $\mathcal{T}$.

Theorem 9366.23 (Presentation weakening has zero classifier shift)**.** *If $\mathcal{T}'$ is a presentation weakening of $\mathcal{T}$, then for any records R, R',*

$$\Delta_{\mathsf{cls}}\big((\mathcal{T}, R), (\mathcal{T}', R')\big) = 0.$$

Proof. The source and classifier specifications are unchanged. Hence on the common source scope the two classifier predicates are literally the same predicate, so $\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}'$. □

Corollary 9366.24 (Renaming is not classifier discovery)**.** *Changing notation, weakening a pattern presentation, or relaxing a ledger presentation is not classifier discovery when the source and classifier specifications are unchanged.*

Proof. This is Definition 9366.23. □

9366.5 Separation principle

Definition 9366.25 (Classifier-level information)**.** *For a transition*

$$(\mathcal{T}, R) \longrightarrow (\mathcal{T}', R'),$$

its classifier-level information audit is the pair

$$\mathsf{CLI}\big((\mathcal{T}, R), (\mathcal{T}', R')\big) = (\Delta_{\mathsf{rec}}, \Delta_{\mathsf{cls}}),$$

where

$$\Delta_{\mathsf{rec}} = |R'| - |R|$$

is external record growth, and Δ_{cls} is the classifier-shift value.

Theorem 9366.26 (Record growth and classifier shift are independent dimensions)**.** *There are transitions with $\Delta_{\mathsf{rec}} > 0$ and $\Delta_{\mathsf{cls}} = 0$. There are also certified-state transitions with $\Delta_{\mathsf{rec}} = 0$ and $\Delta_{\mathsf{cls}} = 1$ whenever the displayed states are not classifier-equivalent on their common source scope.*

Proof. For the first case, take any unfolding step that adjoins a non-empty continuation result and apply Definition 9366.8. For the second case, keep the record fixed and replace $\mathcal{T}$ by a certified classifier state $\mathcal{T}'$ with $\neg(\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}')$. The record size is unchanged, while Definition 9366.7 gives $\Delta_{\mathsf{cls}} = 1$. □

Principle 9366.27 (Computation extends records; discovery changes classifiers)**.** *Continuation grows a trace;* NameCert *licenses a classifier. Thus output length and classifier-level information are distinct audit coordinates. A discovery is a ledgered classifier shift, not a longer continuation trace.*

Concretized in §469: Continuation trace packets realize the record-growth side of the classifier-level information separation.

Concretized in §590: Classifier morphism packets realize the certificate-preserving side of classifier comparison and presentation change.

Concretized in §6765: Classifier-channel shift ledgers realize the classifier-level information distinction as finite before/after classifier rows, record-growth rows, shift witnesses, ledger, transport, replay, provenance, and NameCert data.

Concretized in §8183: Classifier-level information is realized as finite classifier-state, SigRel handoff, ledger-delta, transport, replay, provenance, and NameCert rows.

Chapter 9367

Type Checking as Classifier Membership

The extension predicate Ext of chapter 4 is, structurally, a type-membership predicate. A history satisfies Ext exactly when it lies in a designated subclass cut out by a closure rule. Reading Ext as a type and Cont as the typed reduction, the BEDC primitive layer already contains the data of a typed programming language; type checking is decidable Ext-membership; the Curry-Howard correspondence is built into the framework, since every classifier closure law is the constructive proof of its closure.

9367.1 Extension predicate as type-membership

Definition 9367.1 (Typed term language as naming certificate)**.** *A* typed term language L *with type system* τ *is a closed inductive carrier* Term_L *together with a closed inductive predicate* $\tau\, t$ *on* Term_L *marking well-typedness. The corresponding* naming certificate $\mathsf{NameCert}_L$ *has carrier* Term_L *embedded in* Hist*,* $\mathsf{Ext}_L\, t = \tau\, t$*, and* $\mathsf{Cont}_L\, t\, m\, t'$ *encoding the small-step typed reduction relation of* L*.*

Definition 9367.2 (Type checking)**.** Type checking *for a term* $t \in \mathrm{Term}_L$ *is the decision procedure for* $\mathsf{Ext}_L\, t$*. The procedure exists when* $\mathsf{NameCert}_L$ *supplies a kernel-level decidability witness for* Ext_L*, given as a function on* Term_L *returning* b1 *on well-typed terms and* b0 *otherwise.*

Theorem 9367.3 (Type-checking as Ext membership)**.** *For every typed term language* L *with a fixed type system, there exists a naming certificate over* L *such that well-typedness of a term* t *is exactly* $\mathsf{Ext}\, t$ *in the certificate's classifier. Type-checking decidability is* Ext*-decidability; subject reduction is the closure law connecting* Ext *with the certificate's continuation; the Curry-Howard correspondence between proofs and programs is the identification of the certificate's closure laws with constructive proofs of those closures.*

```
Lean4✓  BEDC.Reflection.type checking as ext membership proof
  ↪variant  BEDC.Reflection.type checking as ext membership
```

Sketch. The forward direction is Definition 9367.1: a typed term language is given as a closed inductive carrier with a typing predicate, and the predicate is encoded directly as Ext_L. The backward direction reads a naming certificate over a term-shaped carrier as a typed term language: Ext is the typing predicate, Cont is the typed reduction, and the certificate's closure laws are the type-system rules. The Curry-Howard correspondence is the observation that the closure laws of

$\mathsf{NameCert}_L$ are themselves typed: each closure law is a function from witnesses of its hypotheses to a witness of its conclusion, and witnesses of $\mathsf{Ext}_L\, t$ are programs whose types are the conclusions. □

9367.2 Subject reduction from closure laws

Definition 9367.4 (Subject reduction)**.** Subject reduction *for* $\mathsf{NameCert}_L$ *is the closure law: for every* $t, t' \in \mathrm{Term}_L$ *and every* $m \in \mathsf{Mark}$, $\mathsf{Ext}_L\, t \wedge \mathsf{Cont}_L\, t\, m\, t' \Rightarrow \mathsf{Ext}_L\, t'$. *The law expresses that typed reduction preserves well-typedness.*

Theorem 9367.5 (Closure law equals subject reduction)**.** *The closure law of* $\mathsf{NameCert}_L$ *that connects* Ext_L *with* Cont_L *is, by unfolding, the subject-reduction property of Definition 9367.4. A naming certificate satisfies the kernel's* Ext-Cont *closure law if and only if the underlying typed term language enjoys subject reduction.*

Sketch. Both statements have the same shape: "if the source is in the extension and the continuation step holds, the target is in the extension". The only difference is the choice of vocabulary: closure law for the certificate framework, subject reduction for the typed-language framework. The proof is a one-line rewrite of the kernel's closure-law statement using the embedding of Term_L into Hist. □

9367.3 Curry-Howard as identity

Theorem 9367.6 (Closure proofs are programs)**.** *Every constructive proof of a closure law of* $\mathsf{NameCert}_L$ *is, on the nose, a program in* Term_L *whose type is the conclusion of the closure law. Conversely, every well-typed program in* Term_L *is the constructive content of a closure-law proof in* $\mathsf{NameCert}_L$ *for some closure formed from its type.*

Sketch. A closure law is a kernel-level implication "hypotheses ⇒ conclusion". A constructive proof of the implication is a function from witnesses of the hypotheses to a witness of the conclusion. Witnesses of the hypotheses are programs with the hypotheses' types; the constructed witness of the conclusion is a program with the conclusion's type; the function itself is a typed lambda abstraction. The reverse reading takes a well-typed term $t : T$ and reads the type T as the conclusion of a closure law whose hypotheses are the types of the free variables of t; the term t is the proof. □

Remark 9367.7 (Curry-Howard built into BEDC)**.** *Definition 9367.6 is the BEDC reading of the Curry-Howard correspondence. In a separate logical-system-plus-programming-language presentation, Curry-Howard is a non-trivial identification of two prima facie distinct domains; in BEDC, the identification is the unfolding of a single object, the naming certificate, into its two readings as a type system and as a program. The kernel's closure-law discipline is therefore not a logical commitment imposed on top of a programming language; it is the programming language at the kernel level.*

Remark 9367.8 (Composition with the compilation capstone)**.** *Definition 9367.3 supplies, for every typed term language, a naming certificate; chapter 9371 reads compilers as naming-certificate morphisms; together, the two chapters identify type-preserving compilers with* Ext*-preserving naming-certificate morphisms. Subject reduction in the source language and target language is, in this composed reading, the kernel-level closure-law preservation that any compiler must witness.*

Concretized in §476: Type checking is realized as finite Ext-membership and subject-reduction NameCert packets over displayed BEDC histories.

Concretized in §489: Classifier membership is realized as a finite BHist typing packet with Ext, Cont, SigRel, and NameCert rows.

Concretized in §483: Subject reduction is realized as the finite Ext/Cont closure packet for type-checking certificates.

Concretized in §595: Classifier boundary packets realize accepted Ext-membership, refused non-member rows, Cont preservation, SigRel readback, and NameCert provenance as a finite public boundary.

Concretized in §562: Typed reduction normal-form packets realize the type-checking vision as finite Ext, Cont, hsame, Pkg, and NameCert rows separating subject reduction from host conversion and normalization oracles.

Concretized in §590: Classifier morphisms are realized as finite Ext, SigRel, Cont, hsame, Pkg, and NameCert preservation packets.

Concretized in §598: Type-checking derivation traces are realized as finite Ext, Cont, subject-reduction, hsame, Pkg, and NameCert rows.

Concretized in §642: Proof-term erasure traces realize type checking as finite Ext-membership preservation over proof terms, erased terms, Cont routes, hsame transport, Pkg provenance, and local NameCert rows.

Concretized in §690: Proof-term erasure audit routes are realized as finite Ext-membership, erased Cont replay, subject-reduction, hsame, Pkg, and NameCert rows without exporting proof objects or host conversion.

Concretized in §707: Proof-term erasure classifiers realize erasure-preserving Ext-membership as a finite NameCert packet over proof-term rows, erased-term rows, Cont replay, SigRel readback, hsame transport, Pkg provenance, and non-escape ledgers.

Concretized in §625: Type-preserving compiler packets realize classifier-preserving compilation with Ext-membership and subject-reduction rows.

Concretized in §686: Type-checking membership traces are realized as finite Ext-membership, derivation-route, subject-reduction, SigRel readback, hsame, Cont, Pkg, and NameCert packets.

Concretized in §809: Type-classifier route joins are realized as finite membership, route-choice, discharge-socket, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1046: Typing-judgment classifiers realize type checking as finite Ext-membership, derivation-route, SigRel readback, Cont replay, hsame transport, Pkg provenance, and NameCert rows.

Concretized in §3911: Type checking classifier membership is realized as finite term, typing-ledger, Ext-membership, derivation-trace, subject-reduction, erasure, hsame, Cont, Pkg, and NameCert rows.

Concretized in §4106: Type-checking classifiers realize Ext-membership, derivation trace, subject-reduction, erasure, hsame, Cont, Pkg, and NameCert rows as a finite packet.

Concretized in §5157: Proof-trace normal-form packets realize type checking as finite Ext-membership, typed Cont replay, subject-reduction, terminal normal-form access, hsame, Pkg, and NameCert rows.

Concretized in §4238: MetaCIC parallel-diamond frontier packets realize classifier membership through residual-substitution premise rows, critical-pair rows, candidate-join rows, closed-checker rows, and local NameCert data.

Concretized in §8250: TypecheckingClassifierTraceUp realizes the type-checking-as-classifier-membership route as finite context, rule, membership, refusal, replay, and NameCert rows.

Chapter 9368

The Halting Problem as a Limit of the Form of Distinction

Concretized in §496: Halting diagonal packets realize the finite fixed-point obstruction boundary over NameCert, Cont, Ext, hsame, Pkg, and local diagonal ledgers.

Reading the form of distinction (Definition 9461.1) as the primitive computational atom and the continuation Cont as the small-step relation (chapter 9365), the halting problem becomes the question: is there a BEDC classifier on naming certificates that decides, for an arbitrary certificate $\mathsf{NameCert}_P$ and history h, whether the continuation chain from h in $\mathsf{NameCert}_P$ terminates? This chapter records the kernel fixed-point interface behind the negative answer as a Tarski-style obstruction in BEDC, locates it on the open-meta-loop side of the two-loops structure (Definition 9461.15), and notes that the obstruction is, structurally, the same as the form of distinction's irreducibility (Definition 9461.3) read at the meta level.

9368.1 Halting as fixed-point obstruction

Definition 9368.1 (Termination of a Cont-chain)**.** *The* Cont*-chain from* h *in* $\mathsf{NameCert}_P$ terminates *when there exists a finite list of marks* $[m_1, \ldots, m_n]$ *and a sequence* $h = h_0, h_1, \ldots, h_n$ *with* $\mathsf{Cont}_P\, h_i\, m_{i+1}\, h_{i+1}$ *for* $0 \leq i < n$ *and no* m, h' *with* $\mathsf{Cont}_P\, h_n\, m\, h'$*. Termination is a positive existential at the kernel level.*

Definition 9368.2 (Halting predicate)**.** *A* halting predicate *is a naming certificate* $\mathsf{NameCert}_H$ *over the carrier* $\mathcal{C} \times \mathsf{Hist}$ *of pairs (certificate, history) whose* Ext_H *on* $(\mathsf{NameCert}_P, h)$ *holds exactly when the* Cont_P*-chain from* h *terminates in the sense of Definition 9368.1, and whose* Cont_H *supplies the kernel-level evidence required by the certificate framework.*

Theorem 9368.3 (Halting via fixed point of the form of distinction)**.** *Let* $\mathsf{NameCert}_H$ *be a naming certificate for a predicate* Halts *and classifier* $\sim_H$*, and suppose that* $\mathsf{Halts}(h)$ *is exactly the positive kernel fact that some* Cont*-continuation from* h *exists. Let* $\mathsf{SelfRef}$ *be a self-reference predicate whose histories are produced by one* Ext *step, and let a diagonal policy say that whenever a self-referential* h *continues with empty fuel to* r*, any classification* $h \sim_H r$ *yields* $\mathsf{Diverges}(r)$*. Then* $\sim_H$ *transports* Halts*, every history satisfies* Halts *by the empty* Cont*-continuation, and every self-referential history carries an* Ext *witness together with the fixed point* $\mathsf{Cont}\, h\, \emptyset\, h$ *and the diagonal implication* $\sim_H hh \Rightarrow \mathsf{Diverges}(h)$*. This is the kernel form of the halting obstruction: the contradiction is not a separate primitive but appears only when a classifier is also required to close that diagonal divergence claim.*

Lean4✓ BEDC.Reflection.halting as form of distinction fixed point proof
↪ *variant BEDC.Reflection.halting as form of distinction fixed point*

Sketch. The certificate field `carrier_respects_equiv` gives transport of Halts along $\sim_H$. The empty continuation law gives $\mathsf{Cont}\, h\, \emptyset\, h$ for every h, hence $\mathsf{Halts}(h)$ under the stated equivalence. If h is self-referential, the self-reference hypothesis supplies an Ext predecessor and mark for h; using the empty continuation as the fixed point gives the witness $r = h$. The diagonal implication is then exactly the supplied policy applied to this empty-fuel continuation. □

9368.2 Halting as open-meta-loop closure

Theorem 9368.4 (Halting decidability is open-meta-loop closure)**.** *Accepting a halting predicate as in Definition 9368.2 closes the open meta loop of chapter 9461 at the certificate stratum: it supplies a* $\mathsf{NameCert}$*-level decision for a predicate over the carrier of all naming certificates, which is precisely the stratum the meta loop refuses to settle.*

Sketch. A halting predicate is, by Definition 9368.2, a $\mathsf{NameCert}$-level decision procedure indexed by the certificate-of-certificates carrier $\mathcal{C}$. By the unification theorem of Definition 9458.6, supplying a $\mathsf{NameCert}$-level decision indexed by $\mathcal{C}$ closes the open meta loop. The diagonal interface of Definition 9368.3 is the kernel-level witness showing where such a closure must also account for self-reference. □

9368.3 Connection to the form of distinction

Theorem 9368.5 (Halting irreducibility is form-of-distinction irreducibility at the meta level)**.** *The fixed-point interface for the halting predicate (Definition 9368.3) is the meta-level reading of the irreducibility of the form of distinction (Definition 9461.3): both expose a primitive distinction that a classifier must already respect. The form of distinction is the irreducible primitive at the constructor stratum (telling* $\mathsf{b0}$ *and* $\mathsf{b1}$ *apart); halting is the corresponding pressure point at the certificate stratum (telling terminating from non-terminating* Cont*-chains apart).*

Sketch. Definition 9461.3 shows that any classifier $\sim$ on a carrier G presupposes the form of distinction in its very statement. Read G as $\mathcal{C}$ and $\sim$ as the would-be halting classifier: deciding whether $(\mathsf{NameCert}_P, h) \sim (\mathsf{NameCert}_Q, k)$ in the sense of having the same termination status is an instance of the form of distinction at the certificate stratum. The halting interface is therefore the form-of-distinction irreducibility specialised to $G = \mathcal{C}$ and $\sim\, =$ halting equivalence; the diagonal fixed point is the explicit witness that this instance does not collapse to a simpler kernel primitive. □

Remark 9368.6 (Two-loops architecture)**.** *Definition 9368.3 and Definition 9461.3 sit on the two opposite faces of Definition 9461.15: the form of distinction must close at the bottom for the kernel to start (closed ground loop), and the same closure exposes the self-reference burden carried by halting decidability at the top (open meta loop). The two-loops structure is, read through this chapter, the discipline that the closure paid at the bottom is not silently transferred to the top.*

Theorem 9368.7 (Halting predicate exists iff the meta loop closes at the certificate stratum)**.** *Let*

$$\mathsf{Halts}(\mathsf{NameCert}_P, h)$$

denote the predicate that the Cont_P*-chain from* h *terminates in the sense of Definition 9368.1. The following are equivalent:*

1. *there exists a halting predicate* $\mathsf{NameCert}_H$ *in the sense of Definition 9368.2;*
2. *there exists a* $\mathsf{NameCert}$*-level decision procedure indexed by the certificate-of-certificates carrier* $\mathcal{C}$*, with carrier* $\mathcal{C} \times \mathsf{Hist}$*, whose extension predicate decides* $\mathsf{Halts}(\mathsf{NameCert}_P, h)$ *for every pair* $(\mathsf{NameCert}_P, h)$*.*

In this situation the open meta loop is closed at the certificate stratum precisely by that deciding certificate.

Proof. Assume (1). By Definition 9368.2, the certificate $\mathsf{NameCert}_H$ is already a naming certificate over $\mathcal{C} \times \mathsf{Hist}$ whose extension predicate on $(\mathsf{NameCert}_P, h)$ holds exactly when the Cont_P-chain from h terminates. Its continuation field is the kernel-level evidence required by the certificate framework. Therefore $\mathsf{NameCert}_H$ is exactly a $\mathsf{NameCert}$-level decision procedure of the form stated in (2). By Definition 9368.4, this deciding certificate is exactly what closes the open meta loop at the certificate stratum for the halting predicate.

Assume (2). Let $\mathsf{NameCert}_D$ be the stated $\mathsf{NameCert}$-level decision procedure. By hypothesis, $\mathsf{NameCert}_D$ lives over the carrier $\mathcal{C} \times \mathsf{Hist}$, its extension predicate on $(\mathsf{NameCert}_P, h)$ holds exactly when the Cont_P-chain from h terminates, and its continuation field supplies the certificate-level evidence attached to that decision. But these are precisely the clauses of Definition 9368.2. Therefore $\mathsf{NameCert}_D$ is a halting predicate; rename it $\mathsf{NameCert}_H$.

Thus (1) and (2) are equivalent, and the final sentence follows from Definition 9368.4 applied to that same deciding certificate. □

Concretized in §490: Halting obstruction is realized as a finite NameCert packet over certificates, histories, finite Cont traces, self-reference rows, diagonal obstruction ledgers, hsame transport, Pkg, and local NameCert provenance.

Concretized in §491: Halting distinction packets realize the fixed-point obstruction as a finite NameCert carrier over certificate-history pairs.

Concretized in §495: The halting fixed-point obstruction is realized as a finite NameCert packet over BHist, Cont, classifier transport, diagonal rows, and non-decision ledgers.

Concretized in §492: Halting distinction traces realize the certificate-level fixed-point obstruction as finite Cont, Ext, Pkg, and NameCert rows.

Concretized in §985: Halting finite boundaries realize finite Cont-trace evidence and diagonal refusal as a NameCert packet without deciding divergence.

Concretized in §978: Halting trace classifiers realize positive finite Cont-trace evidence and diagonal refusal as a finite NameCert packet without deciding divergence outside the displayed trace.

Concretized in §1197: Termination refusal boundaries realize the halting distinction as finite trace rows plus refusal-to-decide rows without deciding divergence.

Concretized in §597: Continuation termination packets realize halting-as-distinction-limit as finite Cont traces with terminal-read and classifier non-escape boundaries.

Chapter 9369

Channel Termination and the Halting Boundary

This vision chapter records the boundary exposed by the active channel compiler: the Zeckendorf-safe channel solves the local problem of where a finite event ends, but it does not solve the global problem of whether an arbitrary certificate computation halts. The two uses of "termination" therefore live at different strata. Channel termination belongs to the compiled display layer of chapter 9208 and chapter 9209; halting belongs to the certificate stratum of chapter 9368.

9369.1 Local channel termination

Definition 9369.1 (Source event and channel event)**.** *Let* $\Sigma = \{0, 1\}$*. A* source event *is any finite word* $w \in \Sigma^*$*. The source layer therefore admits* 11*,* 011*, and* 0011 *as ordinary raw events. The active channel layer uses the Zeckendorf-safe body encoding*

$$\zeta(\epsilon) = \epsilon, \qquad \zeta(0w) = 0\zeta(w), \qquad \zeta(1w) = 10\zeta(w),$$

and the channel event associated to w *is*

$$\mathsf{Ev}(w) = \zeta(w)\,11.$$

The final displayed 11 *is a channel-level terminator, not a source-level substring judgment.*

Definition 9369.2 (Legal channel event and legal channel stream)**.** *The legal single-event language is*

$$\mathsf{LegalEvent} := (0 \mid 10)^*\,11,$$

and the legal channel-stream language is

$$\mathsf{LegalZStream} := ((0 \mid 10)^*\,11)^*.$$

A legal stream is thus a finite concatenation of self-delimiting channel events.

Theorem 9369.3 (Channel bodies contain no terminator pattern)**.** *For every source event* $w \in \Sigma^*$*, the body* $\zeta(w)$ *contains no adjacent substring* 11*.*

Proof. By construction, $\zeta(w)$ is a finite concatenation of fragments from $\{0, 10\}$. Neither fragment contains 11. The only possible fragment junctions are 0|0, 0|10, 10|0, and 10|10, producing the local strings 00, 010, 100, and 1010. None contains adjacent 11. Hence no adjacent 11 occurs inside the encoded body. □

Corollary 9369.4 (The first 11 is the event terminator)**.** *For every source event w, the first adjacent occurrence of* 11 *in* $\mathsf{Ev}(w) = \zeta(w)11$ *is the appended channel terminator.*

Proof. The body contains no adjacent 11 by Definition 9369.3. The appended suffix is 11. Therefore the first adjacent 11 seen by a left-to-right decoder is precisely the terminator. □

Theorem 9369.5 (The source event 11 is encodable)**.** *The raw source event* 11 *is not forbidden. Its channel encoding is*

$$\mathsf{Ev}(11) = 101011.$$

The last two glyphs are the channel terminator; the two source glyphs $1, 1$ *are represented by the body* $10\,10$*.*

Proof. The body rule sends each source glyph 1 to the channel fragment 10. Thus

$$\zeta(11) = 10\,10 = 1010.$$

Appending the event terminator gives

$$\mathsf{Ev}(11) = \zeta(11)11 = 101011.$$

A decoder reads 10 as the first source 1, reads the next 10 as the second source 1, and stops only at the final 11. □

Corollary 9369.6 (Layer separation for the glyph pair 11)**.** *The displayed pair* 11 *has two distinct readings:*

$$\textit{source-level } 11 = \textit{two raw source glyphs}, \qquad \textit{channel-level } 11 = \textit{event terminator}.$$

The readings are selected by layer and must not be silently interchanged.

Proof. Definition 9369.5 shows that source-level 11 is preserved as body 1010 followed by the terminator. By Definition 9369.4, any channel-level first occurrence of adjacent 11 in a generated event is the boundary marker. Hence the same displayed glyph pair cannot be assigned both readings in the same judgment. □

9369.2 Bare bitstreams and self-delimitation

Definition 9369.7 (Bare bitstream encoding)**.** *A bare bitstream encoding writes each source event w as the same word w and writes an event flow $(w_0, \ldots, w_{n-1})$ as the concatenation $w_0 \cdots w_{n-1}$, without an event terminator, length table, schema, manifest, host parser, or prefix discipline.*

Theorem 9369.8 (Bare bitstream encoding is not self-delimiting)**.** *Bare bitstream encoding cannot recover event boundaries from the resulting glyph string alone.*

Proof. The displayed glyph string 011 admits several source-flow readings:

$$(011), \qquad (0, 11), \qquad (0, 1, 1).$$

All produce the same bare concatenation. Without extra boundary data, there is no intrinsic rule selecting one decomposition. Thus the bare bitstream carries content glyphs but not self-delimiting event boundaries. □

Theorem 9369.9 (Zeckendorf-safe channel termination)**.** *Every legal channel event has a streaming-decoded endpoint: scanning left-to-right, the first adjacent* 11 *ends the current event, and the body before it decodes uniquely to a source event.*

Proof. A legal event has the form $(0 \mid 10)^*11$. Each body fragment is read locally: 0 emits source 0, and 10 emits source 1. By Definition 9369.3, the body contains no adjacent 11. Hence the first adjacent 11 cannot be body data and must be the terminator. The decoder therefore stops after a finite scan and recovers the unique source event encoded by the body. □

Corollary 9369.10 (Constraint-minimal reading)**.** *Under the constraints of a binary alphabet, no third glyph, no external length table, no schema or manifest, streaming decoding, and reserved terminator* 11*, the rule*

$$0 \mapsto 0, \qquad 1 \mapsto 10, \qquad \mathrm{End} = 11$$

provides the minimal active BEDC channel discipline: it internalizes finite event boundaries without adding an additional kernel mark.

Proof. The body fragment for source 0 is already the one-glyph fragment 0. The source glyph 1 cannot be represented by 1, because a following 1 would collide with the terminator 11; it cannot be represented by 11, because 11 is reserved as End. The shortest remaining safe fragment is 10. The fragment set $\{0, 10\}$ is exactly the body language used in Definition 9369.9, so the resulting discipline is streaming, binary, and self-delimiting. This is a structural minimality claim under the displayed constraints, not a Shannon- or Huffman-optimality claim for an arbitrary probability distribution. □

9369.3 Channel grammar and finite window geometry

Definition 9369.11 (Three readings of the no-adjacent grammar)**.** *The no-adjacent-*11 *grammar has three distinct BEDC readings:*

$$\textit{channel reading: } \mathsf{LegalEvent} = (0 \mid 10)^*11,$$
$$\textit{finite-window reading: } X_m = \{w \in \{0,1\}^m : w \textit{ has no adjacent } 11\},$$
$$\textit{spectral reading: } |X_m| = F_{m+2} \textit{ with growth scale } \varphi.$$

The first is a compiler-channel discipline, the second is a finite address-geometry discipline, and the third is the growth invariant of the same finite-type grammar.

Principle 9369.12 (Ground channel is a streaming use of the golden-mean grammar)**.** *The Zeckendorf-safe compiler does not identify channel streams with Window-six geometry. It uses the same local forbidden pattern for a different task. In the compiler, bodies lie in*

$$(0 \mid 10)^*$$

and the first adjacent 11 *is reserved as boundary. In the Axis-Zeckendorf window geometry of Definition 9347.1, the same forbidden pattern defines the admissible finite windows X_m and the inverse-limit address carrier $X_\infty^{\mathbb{Z}}$. In Definition 9347.15, the transition kernel records the associated golden-mean shift.*

Thus the shared object is the local grammar, not a collapse of compiler channels, finite windows, and source-level Zeckendorf normal forms into one type.

Theorem 9369.13 (Channel bijection is not source-level Zeckendorf rewriting)**.** *The GroundCompiler channel bijection*

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}$$

is a self-delimiting display theorem for event flows. It is not a source-level Zeckendorf carry or normal-form theorem.

Proof. By Definition 9222.4, the legal channel stream language is

$$\mathsf{LegalZStream} = ((0 \mid 10)^*11)^*,$$

and Compile and Decode are inverse maps. By Definition 9222.7, source events and channel streams are strictly separated: channel-level 11 is a terminator, source-level 11 may be raw data, and source-level normalization acts only after decoding. Therefore the channel bijection licenses lossless event flow display, not a direct string rewrite by a Zeckendorf carry classifier. □

Principle 9369.14 (Boundary recovery pattern)**.** *The compiler channel realizes the pattern*

local forbidden body pattern → detectable boundary → streaming decoder → round-trip certificate.

Window six realizes the related finite-geometry pattern

local forbidden window pattern → finite admissible carrier → inverse-limit extension boundary.

The analogy is structural. It does not turn legal-stream membership into a theorem, a proof, a recognized NameCert, an accepted object flow, or a source-level Zeckendorf normal form.

9369.4 The halting boundary

Definition 9369.15 (Local event termination versus certificate halting)**.** Local event termination *asks where the current compiled channel event ends. It is answered by the first channel-level terminator* 11.

Certificate halting *asks whether, for an arbitrary naming certificate* $\mathsf{NameCert}_P$ *and history* h*, the* Cont_P*-chain from* h *terminates in the sense of Definition 9368.1.*

Theorem 9369.16 (Channel termination does not decide halting)**.** *The Zeckendorf-safe channel gives a decision procedure for local event boundaries in* LegalZStream*, but it does not provide a BEDC internal total classifier deciding certificate-level halting for all pairs* $(\mathsf{NameCert}_P, h)$*.*

Proof. The local channel claim follows from Definition 9369.9: a legal event is a finite word in $(0 \mid 10)^*11$, and its first 11 is a boundary. The question is syntactic and channel-local.

A halting classifier would instead live over the carrier $\mathcal{C} \times \mathsf{Hist}$ of certificates and histories and would decide, for each pair $(\mathsf{NameCert}_P, h)$, whether the Cont_P-chain terminates. By Definition 9368.7, such a predicate is exactly a certificate-level decision procedure at the certificate stratum, and by Definition 9368.4 that assumption closes the open meta loop. The diagonal obstruction of Definition 9368.3 is not affected by knowing where a finite channel event ends. Therefore channel termination and global halting live at different strata. □

Corollary 9369.17 (Stop codon versus meta-stop)**.** *The channel-level terminator* 11 *is a local stop marker for event parsing. The halting problem is the refusal of a global meta-stop: no internal total classifier may close the certificate stratum by deciding termination of every possible* Cont*-chain.*

Proof. The first sentence is exactly Definition 9369.4. The second follows from Definition 9369.16: a local parser boundary does not supply the certificate-of-certificates classifier whose existence is equivalent to meta-loop closure. □

Remark 9369.18 (Compressed reading)**.** *The compressed reading is therefore:* 11 *is the channel layer's stop codon; the halting theorem is the certificate layer's refusal of a universal meta-stop. The first is engineered by excluding* 11 *from encoded bodies. The second is blocked by the diagonal fixed-point obstruction that appears when the open meta loop is treated as closed.*

Concretized in §3996: The local channel boundary is realized as a finite NameCert packet separating event termination from certificate-level halting.

Concretized in §1197: Termination refusal boundaries realize the local-channel-termination versus global-halting boundary as finite refusal, visibility, and NameCert rows.

Concretized in §5967: The local channel terminator/global halting boundary is realized as finite budget-certificate NameCert data.

Concretized in §6725: Halting refusal ledgers realize the local channel terminator/global halting boundary as finite fuel, trace, timeout-refusal, and global-refusal NameCert rows.

Concretized in §985: Halting finite-boundary packets realize the channel-termination vision as finite trace evidence plus diagonal refusal, terminal trace, inscription, transport, replay, provenance, and NameCert rows.

Concretized in §7098: Certified event flows realize source/channel endpoint separation as finite NameCert data.

Concretized in §7790: Channel termination boundary packets realize the first-11 local delimiter as finite NameCert data separated from global halting claims.

Chapter 9370

Strong Normalisation versus Turing Completeness: The Total-Language Position of BEDC

The cellular reading of chapter 9390 identifies Rule 110 as a minimal universal cellular substrate, and locates the project's `rule110/` sub-system as the smallest substrate-side instantiation of the closed-observational framework. A complementary axis is the relationship between the *host system* (BEDC, sitting inside CIC) and Turing completeness in the sense relevant to the substrate. The host system is not Turing complete in the same sense; it is a total functional language. This chapter makes the precise statement of that fact, locates BEDC against other total functional languages (Coq, Agda, Lean, Idris 2 with quantitative type theory), and shows how the closed-observational framework forces the total-language position rather than treats it as an incidental design choice.

9370.1 Setup: total functional language as discipline

Definition 9370.1 (Total functional language)**.** *A* total functional language *is a programming or type-theoretic system in which every well-typed term reduces to a normal form in finitely many reduction steps under the system's reduction relation. Equivalently, no well-typed term computes forever; partial functions cannot be defined as functions; general recursion is admitted only when accompanied by a well-founded measure that the type checker can verify. The Calculus of Inductive Constructions in its Lean 4 implementation is a total functional language under this definition.*

Definition 9370.2 (Turing-complete computational model)**.** *A* Turing-complete *computational model is a specification of a class of computational processes such that every partial recursive function on natural numbers is realised as some process in the class. Equivalent formulations include the lambda calculus, the unrestricted register machine, the cyclic tag system, and any system Cook 2004 reduces to as a universality witness, in particular the orbit of Rule 110 on a suitable initial configuration.*

Principle 9370.3 (BEDC is total; substrate is universal)**.** *BEDC, as a Lean 4 development under CIC, is a total functional language in the sense of Definition 9370.1. The cellular substrate* `rule110/` *is a Turing-complete computational model in the sense of Definition 9370.2. These two facts coexist without contradiction precisely because they describe two structurally different layers: the host (BEDC) reasons* about *computations, while the substrate* performs *them. The host's totality is not a weakening of the substrate's universality; it is a separate axis of discipline.*

9370.2 The expressive gap, made precise

Remark 9370.4 (What a total host cannot directly express)**.** *A total functional language cannot define, as a function of type $A \to B$, an algorithm whose termination is not internally provable. There exist Turing-computable functions whose totality is provable in a stronger meta-theory but not in CIC itself; these functions, while Turing-realisable, are not directly definable as BEDC functions. The set of BEDC-definable functions is therefore a strict subset of the set of Turing-realisable partial functions on natural numbers; the difference is the set of functions whose totality is unprovable within CIC.*

Definition 9370.5 (Step-indexed encoding of partial computation)**.** *A* step-indexed encoding *of a partial recursive function ϕ inside a total language is a function $\widehat{\phi}\colon \mathbb{N} \times \mathrm{Input} \to \mathrm{Output} + \{\bot\}$ such that $\widehat{\phi}(n, x) = \phi(x)$ whenever $\phi(x)$ converges in fewer than n reduction steps and $\widehat{\phi}(n, x) = \bot$ otherwise. The step parameter n is the* fuel*. Step-indexed encoding allows a total host to talk about every Turing-realisable partial function, at the cost that totality of $\widehat{\phi}$ is decoupled from convergence of ϕ.*

Example 9370.6 (Rule 110 evaluator in BEDC)**.** *The Lean 4 development `lean4/BEDC/FKernel/` provides `r110_run_n_steps`, a function that takes a fuel parameter and a starting cell row and returns the row after the specified number of Rule 110 update steps. This is a step-indexed encoding of the Rule 110 orbit, written as a total BEDC function. The cellular substrate's potentially-infinite computation is decomposed into a family of finite computations, one per fuel value. The substrate retains its universality; the encoding retains its totality.*

Principle 9370.7 (Step indexing as the bridge)**.** *Step indexing is the structural bridge between a total host and a universal substrate. The host can describe every state the substrate could reach at every finite step, and can prove any decidable assertion about those states; the host cannot, in general, assert that the substrate halts on a given input, because halting is undecidable. The bridge therefore preserves the substrate's universality on the substrate side and the host's totality on the host side; it does not transfer either property across the boundary.*

9370.3 Position among total functional languages

Remark 9370.8 (Coq, Agda, Lean, Idris 2: same axis, different commitments)**.** *Coq, Agda, Lean 4, and Idris 2's quantitative type theory mode all qualify as total functional languages in the sense of Definition 9370.1. They differ in: which inductive types are admitted, which co-inductive structures are admitted, which universe hierarchies are available, and which standard-library axioms are imported by default. BEDC, as a Lean 4 development with `axiom-purity -strict`, is the variant in this family that minimises imported commitments: no `Classical.choice`, no `Quot.sound`, no `propext`. Among total functional languages in active use, this is the smallest residual axiomatic footprint currently audited mechanically.*

Remark 9370.9 (Idris partial functions and the closure cost)**.** *Some total languages, notably Idris, admit a `partial` annotation that allows the language to host non-total functions when explicitly marked. The annotation effectively partitions the language: total subset versus partial subset. BEDC does not admit this partition; every BEDC function is total, with no partial escape hatch. The closure cost of admitting a partial annotation is the loss of the property that "every typed term has a normal form"; once partial functions exist, the meta-theory must be split into two grades.*

Principle 9370.10 (BEDC's total-only discipline is structural)**.** *The choice to admit no partial function annotation is not an aesthetic preference; it is what allows the closed-observational reading of the BEDC kernel to apply uniformly. A partial subset would correspond to a region of the host where computations can in principle not halt, which would break the structural premise that BEDC observations are finite bit strings at finite reduction depth. Total-only discipline is a precondition for the closure theorems of chapter 9393 to transfer cleanly to the BEDC kernel.*

9370.4 The forced choice, read structurally

Theorem 9370.11 (Closed observation forces totality at the host)**.** *Let H be a host system that claims to describe a closed observational system S (Definition 9393.1) by recording finite observations of S as terms of H. Then H must itself be total in the sense of Definition 9370.1, on pain of contradicting the closure premise.*

Sketch. Suppose for contradiction H is non-total. Then H admits a term t whose reduction does not terminate. The semantic content of t is not a finite observation; it is a divergent process. If t is offered as a record *of* an observation of S, then either (i) the observation is itself divergent, contradicting the closure premise that observations of S are finite, or (ii) the observation is finite but the record is not, contradicting the structural correspondence between observations and records. Either way the closure premise fails. Hence H is total. □

Remark 9370.12 (Why this matters for the project framing)**.** *The result of Definition 9370.11 is a forced-choice argument: once the host is committed to recording a closed observational system, totality is not optional. It also clarifies why total functional languages are the correct host class for projects of the BEDC kind: Coq, Agda, Lean 4, Idris 2 in total mode are all instances of host systems whose semantic discipline matches the substrate's closure discipline. A host system based on a non-total language (Standard ML, OCaml, Haskell, Python) would silently violate the structural correspondence and would have to mark every observation record by an explicit termination certificate, which is precisely what total languages enforce statically.*

9370.5 Curry-Howard, and what it means here

Principle 9370.13 (Curry-Howard reading of totality)**.** *Under the Curry-Howard correspondence, a total functional language is one in which every well-typed term is a proof. Allowing non-termination would correspond to allowing proofs that "go on forever", which logically is the same as allowing arbitrary statements to be "derived" by an infinite loop. A non-total language is therefore inconsistent as a proof system. BEDC's totality is the proof-side reading of the closure premise: every typed term is a finite, terminating proof; every proof is a finite observation record; the structural correspondence between observation and proof is unbroken.*

Remark 9370.14 (Universal computation does not require universal proof)**.** *The cellular substrate's Turing universality means it can host any computation. The host's totality means every host proof is finite. These two are compatible because computation and proof are not the same relation. The substrate's computation is something the host* describes *as a step-indexed family; the host's proof is something the host* performs *on its own typed terms. The substrate's universality does not back-propagate to require the host to be universal. The closed-observational reading of BEDC depends on this clean separation; an attempt to unify the two would either weaken the host into a non-total language or weaken the substrate into a sub-universal model.*

9370.6 Mapping to BEDC engineering

Strong-normalisation reading	**BEDC engineering realisation**
Host as total functional language	`lean4/` build is `lake build` exit 0; every theorem terminates by the Lean 4 kernel's termination checker.
Step-indexed substrate computation	`r110_run_n_steps` in `lean4/BEDC/FKernel/` takes a fuel parameter and returns a finite orbit segment.
Substrate universality, host-internal	`rule110/encoder/tm_to_tag.c` and `rule110/encoder/tag_to_cyclic.c` compile a TM to a cyclic tag system; the cyclic tag is then evaluated by the host as a step-indexed computation.
Axiom-pure totality audit	`axiom-purity -strict` traces every theorem's `#print axioms` and rejects any Lean stdlib axiom.
No partial-function escape hatch	`tools/check-axioms.py` rejects `axiom` keyword in `lean4/BEDC/` anywhere; structural enforcement is mechanical.
Closure-discipline transfer to substrate	Definition 9390.10 carries the closure premise into the cellular substrate at the same totality discipline.

9370.7 Closure remarks

This chapter has located BEDC's total-functional-language position as the host-side correlate of the closed-observational framework of chapter 9393. The total-language discipline is not a design preference; it is structurally required if the host is to record finite observations of a closed substrate without violating the closure premise.

The Turing universality of the cellular substrate and the totality of the BEDC host are therefore independent axes that meet without interfering. The host describes the substrate via step-indexed encodings; the substrate's universality is preserved inside the encoding; the host's totality is preserved outside it. Curry-Howard gives the proof-side reading of the same discipline: every typed term is a finite proof, just as every observation record is a finite bit string.

The boundary of this chapter is precisely: BEDC is total in the sense of Definition 9370.1; the cellular substrate is Turing complete in the sense of Definition 9370.2; the two facts coexist without contradiction. This chapter does not enlarge BEDC's expressive power, does not weaken the substrate's universality claim, and does not propose to merge the two axes. It identifies the axes as separate and explains why their separation is structurally required by the closure premise.

Concretized in §936: Finite normal-form packets realize the total-host boundary of strong normalization.

Concretized in §944: Bounded finite trace packets realize the total-host side of the strong-normalization boundary.

Concretized in §963: Fuel-bounded partial traces realize the total-host side of substrate universality as finite NameCert packets.

Concretized in §960: FuelIndexedSubstrateBridge realizes the total-host and universal-substrate separation as a finite fuel-indexed NameCert packet.

Concretized in §964: StepIndexedTotalHost realizes the total-host and universal-substrate boundary as finite fuel-indexed trace, normal-form, refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §962: Total-host fuel handoffs realize the strong-normalization bridge as finite fuel, substrate-trace, terminal-readback, hsame, Cont, Pkg, and NameCert rows.

Concretized in §970: Reduction fuel boundaries realize the total-host side of bounded substrate computation as finite fuel, trace, endpoint, timeout-refusal, audit, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1050: Host total traces realize finite fuel, normal-form readback, substrate endpoint, timeout refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §3706: Closed-observation total-host discipline is realized as finite observation, termination, substrate-step, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §3954: The total-host and universal-substrate separation is realized as finite MetaCIC fuel, normal-form, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §5925: Candidate-mediated strong-normalization boundaries are realized as finite MetaCIC candidate, fuel, normal-form endpoint, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §7962: Residual fuel-rank packets realize the MetaCIC total-host boundary as finite residual, fuel, rank, candidate, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §6838: PartialityBoundaryUp realizes the total-host versus partial-escape boundary as finite NameCert data.

Concretized in §8123: Step-indexed trace packets realize the total-host/substrate bridge as finite fuel, prefix, endpoint, refusal, replay, provenance, and NameCert rows.

Concretized in §8434: Normalization budget traces realize the total-host/substrate boundary as finite fuel, prefix, endpoint, refusal, replay, provenance, and NameCert rows.

Chapter 9371

Compilation as Naming-Certificate Morphism

A compiler from source language S to target language T is a structure-preserving translation that respects the semantics of both. In BEDC terms, S and T each carry a naming certificate (chapter 9367, chapter 9365); a compiler is a morphism between the two certificates that preserves the classifier; compiler correctness is the assertion that the morphism preserves hsame. The chapter records this identification and locates compositional compilation, optimisation correctness, and decompilation in the same framework.

9371.1 Compiler as classifier-preserving morphism

Definition 9371.1 (Naming-certificate morphism)**.** *A* morphism *from* $\mathsf{NameCert}_S$ *to* $\mathsf{NameCert}_T$ *is a kernel-level function* Φ *on the carriers of* S *and* T *together with two preservation witnesses:* Φ *sends every* Ext_S*-marked source history to an* Ext_T*-marked target history, and* Φ *commutes with the continuation chains,* $\Phi(\mathsf{Cont}_S\, h\, m\, h') = \mathsf{Cont}_T\, (\Phi h)\, m'\, (\Phi h')$ *for an explicit mark translation* $m \mapsto m'$*. The morphism preserves* classifier membership *because every classifier-decided fact about* h *becomes a classifier-decided fact about* Φh *via the preservation witnesses.*

In the Lean kernel this is expressed relationally by a compiler graph $G(h, k)$*. The graph is a certificate morphism when every graph edge starts in the source carrier and lands in the target carrier, every source-classified input has a lifted graph edge, and paired graph edges transport source classifier membership to target classifier membership.*

Definition 9371.2 (Compiler)**.** *A* compiler *from* S *to* T *is a naming-certificate morphism* $\Phi\colon \mathsf{NameCert}_S \to \mathsf{NameCert}_T$ *together with the additional witness that* Φ *preserves* hsame*: if two source histories are* hsame*-equivalent, their images are* hsame*-equivalent in the target.*

Theorem 9371.3 (Compilation as naming-certificate morphism)**.** *For every pair of naming certificates* $(\mathsf{NameCert}_S, \mathsf{NameCert}_T)$*, every compiler graph from* S *to* T *whose edges land in the certified carriers, lift along source classifier membership, and preserve classifier membership across paired edges is a naming-certificate morphism* $\mathsf{NameCert}_S \to \mathsf{NameCert}_T$*. Compiler correctness is the assertion that this morphism is a* hsame*-preserving classifier morphism. Sequential composition of correct compilers is composition of morphisms in this category; an optimisation is an endomorphism whose source and target certificates coincide.*

Lean4✓ `BEDC.Reflection.compilation as namecert morphism proof`

↪ variant `BEDC.Reflection.compilation as namecert morphism`

Sketch. The forward direction is the unfolding of Definition 9371.2: a compiler is, by definition, the data of a naming-certificate morphism plus the hsame-preservation witness. In the graph presentation, source classifier membership supplies a lifted output edge, and the morphism witness turns the paired source edges into a target classifier edge. Target carrier preservation then follows from the target naming certificate's carrier transport field. Sequential composition of compilers $\Phi_2 \circ \Phi_1$ inherits the preservation witnesses by composing them, and the resulting hsame-preservation is a kernel-level conjunction of the two component witnesses. □

9371.2 Compositional compilation

Theorem 9371.4 (Composition of correct compilers is correct)**.** *Given correct compilers* $\Phi_1 \colon \mathsf{NameCert}_S \to \mathsf{NameCert}_M$ *and* $\Phi_2 \colon \mathsf{NameCert}_M \to \mathsf{NameCert}_T$*, the composition* $\Phi_2 \circ \Phi_1 \colon \mathsf{NameCert}_S \to \mathsf{NameCert}_T$ *is a correct compiler from* S *to* T*.*

Sketch. Both Ext-preservation and Cont-commutation are closed under function composition: applying Φ_2 to the output of Φ_1 produces a target-marked, target-continuing image of every source-marked, source-continuing input. hsame-preservation composes by transitivity of hsame. □

9371.3 Optimisation and decompilation

Definition 9371.5 (Optimisation)**.** *An* optimisation *on* $\mathsf{NameCert}_S$ *is an endomorphism* $\Phi \colon \mathsf{NameCert}_S \to \mathsf{NameCert}_S$ *in the sense of Definition 9371.1 that preserves* hsame*.*

Theorem 9371.6 (Optimisation correctness as endomorphism preservation)**.** *An optimisation* Φ *on* $\mathsf{NameCert}_S$ *is correct exactly when, for every source history* h*,* $\mathsf{hsame}(h, \Phi h)$ *holds. Equivalently,* Φ *acts as the identity in the category of naming certificates modulo* hsame*.*

Sketch. The endomorphism condition $\mathsf{hsame}(h, \Phi h)$ is the assertion that the optimisation's output is observationally identical to the input under the certificate's classifier. By Definition 9371.1, this preserves Ext and Cont, so the kernel-level behaviour is unchanged. □

Definition 9371.7 (Decompilation)**.** *A* decompilation *from* $\mathsf{NameCert}_T$ *to* $\mathsf{NameCert}_S$ *is a naming-certificate morphism* $\Psi \colon \mathsf{NameCert}_T \to \mathsf{NameCert}_S$ *such that* $\Phi \circ \Psi$ *is an optimisation on* $\mathsf{NameCert}_T$ *in the sense of Definition 9371.5, where* Φ *is the compiler from* S *to* T*. Decompilation is therefore not always available; when* Φ *has no morphism right-inverse modulo* hsame*, decompilation does not exist.*

Remark 9371.8 (Compiler category)**.** *The naming certificates over a fixed mark alphabet, together with the morphisms of Definition 9371.1, form a category by chapter 9363: identities are kernel identity functions, composition is function composition, and the laws follow from* Cont *associativity and unit. Compilers are the* hsame*-preserving morphisms, optimisations are the endomorphisms, and decompilations are the partial right-inverses. The framework subsumes intermediate-representation pipelines (chains* $\Phi_n \circ \cdots \circ \Phi_1$*) and source-to-source transformations (loops in the morphism graph).*

Concretized in §481: Compiler correctness is realized as a classifier-preserving morphism between source and target naming certificates.

Concretized in §473: Compiler graphs are realized as finite NameCert morphism packets preserving carrier, continuation, and classifier rows.

Concretized in §574: Certificate pipeline stage boundaries are realized as finite NameCert packets for adjacent compiler graphs, middle-stage compatibility, classifier transport, continuation composition, and non-escape ledgers.

Concretized in §614: Decompiler right-inverse packets realize decompilation as a finite compiler-graph, optimization, hsame-transport, Cont-route, Pkg, and NameCert boundary.

Concretized in §620: Kernel morphism packets realize classifier-preserving NameCert graph morphisms as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §639: Compiler optimisations are realized as finite NameCert endomorphism packets over displayed compiler graph, classifier, hsame, Cont, Pkg, and local naming rows.

Concretized in §679: Compiler composition traces are realized as finite BHist packets for adjacent compiler graphs, middle-certificate compatibility, classifier transport, continuation composition, and associativity non-escape.

Concretized in §1055: Compiler trace faithfulness realizes compiler correctness as finite graph replay, classifier preservation, hsame transport, provenance, and NameCert rows.

Concretized in §1097: Compiler-classifier routes realize compiler correctness as one finite route tying classifier membership, classifier morphism, NameCert morphism, certificate compiler graph replay, and trace faithfulness rows.

Chapter 9372

Closed-term substitution as the self-compilation step

9372.1 The setup

BEDC.MetaCIC represents a small CIC-like term language by the closed generator `Term`.

The generator has exactly five surface forms: `var`, `app`, `lam`, `pi`, and `sort`.

The closedness classifier for this generator is `ClosedAt`.

For a depth d and a term t, `ClosedAt` d t says that every free de Bruijn index visible in t lies below the boundary d.

The boundary is not cosmetic.

It is the local budget by which the term may use names already present in the ambient binder context.

At depth 0 the classifier marks genuinely closed terms.

At larger depth it marks terms closed relative to the registered variables below that depth.

Thus `ClosedAt` is not merely a syntactic predicate.

It is the classifier that tells the MetaCIC layer which terms remain inside the registered binder ledger.

The two operations at issue are `shift` and `substitute`.

`shift` raises free indices at or above a cutoff.

`substitute` replaces the variable at a specified depth and adjusts indices around the removed binder.

Both operations are ordinary generators over `Term`: they traverse the closed constructor list and produce another `Term`.

But only some generator actions preserve a given classifier.

The self-compilation reading depends exactly on that distinction.

A compiler that compiles itself must feed compiler-shaped source terms through the evaluator it implements.

Inside that evaluator, substitution is not optional bookkeeping.

It is the step that pushes an actual argument into a binder body.

If this step could cross the closedness boundary, the self-compilation chain would silently acquire an unregistered free variable.

The chain would then cease to be a chain of BEDC-certified records.

It would be a chain whose later stage contains a name that no prior certificate licensed.

BEDC therefore reads closed-term substitution as a certificate closure condition.

The relevant claim is not that every substitution preserves every semantic property.

The relevant claim is narrower and sharper: when a term is already closed at the substitution depth, the attempted substitution at that boundary is syntactically inert.

No hidden variable is consumed.

No variable is smuggled through the boundary.

No future binder budget is borrowed to make the operation typecheck.

In formula form, the MetaCIC witness says:

$$\mathsf{ClosedAt}(d, t) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

This is the form of closure that self-compilation needs at the substitution site.

The evaluator may attempt the syntactic operation, but the registered closedness certificate proves that the operation cannot alter a term whose free variables all sit below the boundary.

The source term remains inside the classifier.

The self-application step remains inside the same record discipline.

The compiler does not borrow an unrecorded name from the future of its own compilation chain.

9372.2 The Lean witness

Two theorems from `lean4/BEDC/MetaCIC/ClosedTerm.lean` carry the technical content:

- `shift_closed` –

  ↪variant `BEDC.MetaCIC.shift closed`

- `substitute_closed` –

  ↪variant `BEDC.MetaCIC.substitute closed`

- `closed_under_substitute_classifier` –

  ↪variant `BEDC.MetaCIC.closed under substitute classifier`

  (instance of

  ↪variant `BEDC.MetaCIC.GeneratorClosureClassifier`

)

`shift_closed` states that shifting a term at its closedness boundary by one position leaves the term unchanged.

Its statement is:

$$\mathsf{ClosedAt}(n, t) \Rightarrow \mathsf{shift}(n, 1, t) = t.$$

The proof is structural over the `ClosedAt` witness.

In the variable case, the inequality $i < n$ forces the cutoff test $n \leq i$ to be false.

In the compound cases, the induction hypotheses pass through `app`, `lam`, and `pi`.

The binder cases raise the boundary on the body or codomain, matching the definition of `ClosedAt`.

The universe case is immediate.

The theorem says that a closed term already pays its depth bill.

Raising the external context boundary does not require rewriting the term, because no free index reaches the cutoff.

This is the boundary budget law for self-compilation.

When a closed source fragment is moved into a deeper binder context, the chain does not reach into a later context to justify the move.

The proof spends only the already recorded inequality inside `ClosedAt`.

`substitute_closed` states that substituting at the boundary of a closed term leaves the term unchanged:

$$\mathsf{ClosedAt}(d, t) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

The value v is arbitrary.

That arbitrariness matters.

The theorem does not inspect v because the substitution site is absent.

For a variable index i in a term closed at d, the witness gives $i < d$.

Consequently $i = d$ is false, and $d < i$ is false.

The substitution branch cannot fire, and the decrement branch cannot fire.

The variable is returned as itself.

The compound cases again follow the `ClosedAt` constructor tree.

Under binders, the implementation shifts the substitution value while increasing the depth, but the theorem proves that this shifted value is never observed in a closed-at-boundary body.

That is the self-compilation content.

The evaluator may carry a substitution payload through the recursive traversal, yet the classifier prevents the payload from becoming a hidden source of free variables.

The accepted record is the equality of the syntactic output with the input term.

The certificate is the `ClosedAt` derivation.

The generator is `substitute`.

The closure law is that the generator action is inert on the classified slice.

9372.3 Why this is the self-compilation step

Self-compilation is not a single mystical event.

In BEDC it is a chain of ordinary certified steps.

Each step takes a record already admitted by a classifier, applies an authorized generator, and checks that the resulting record remains inside the promised scope.

For MetaCIC, evaluation under binders inevitably uses substitution.

A compiler source term contains binder bodies.

The evaluator implemented by the compiler opens those binders when it applies functions.

The self-compilation chain therefore cannot avoid the following local question: when the compiler's own evaluator substitutes during evaluation, does the operation preserve the closedness ledger of the source term?

`substitute_closed` answers the question at the exact boundary where the risk appears.

If the source fragment is closed relative to depth d, then a substitution at d has no target inside the fragment.

The operation returns the same term.

Hence the self-compilation chain does not create an unregistered free variable at that step.

The compiler may feed its own term representation to its own evaluator without turning closed source into boundary-leaking source at this operation.

The theorem is therefore a local non-escape certificate.

Concretized in §531: Closed substitution is realized as a finite seal whose inert boundary is certified by the MetaCIC closedness classifier.

It is local because it speaks only about the `substitute` operation at a single depth.

It is a certificate because it is a checked Lean theorem whose proof follows only the closed constructor tree and elementary Nat tests.

It is a non-escape claim because every possible route for the substitution payload to enter the term is blocked by the inequalities in the `ClosedAt` witness.

The same reading applies to `shift_closed`.

Self-compilation moves term fragments through binder contexts.

Moving through a binder context normally requires shifting free variables.

But a term closed below the relevant cutoff contains no free variable at that cutoff.

Shifting it at the boundary is therefore inert.

The chain can deepen the surrounding context without borrowing a depth that has not yet been registered.

This is why `shift_closed` is the boundary budget law.

Depth behaves monotonically with respect to the certificate already held by the term.

The term does not need a later binder to justify its present closedness.

The pair of theorems gives the self-compilation step its exact BEDC shape:

$$\text{registered closed source} \xrightarrow{\text{shift}} \text{same registered source},$$

$$\text{registered closed source} \xrightarrow{\text{substitute}} \text{same registered source}.$$

The equalities are stronger than mere preservation of a predicate.

They state that, at the closed boundary, the operational generator cannot alter the term at all.

The self-compilation chain is therefore protected at the exact place where capture, escape, or name leakage would otherwise enter.

9372.4 Generator-closure classifier reading

The same facts can be read in the certificate-closure vocabulary.

`ClosedAt` is the classifier.

`Term` is the authorized generator carrier.

`shift` and `substitute` are authorized generator operations over that carrier.

`shift_closed` and `substitute_closed` are generator-closure laws for the classified slice.

This is not an analogy.

It is the literal BEDC structure.

A generator-closure classifier states which records remain inside the licensed generator scope after an authorized operation is applied.

Here the records are MetaCIC terms.

The classifier is `ClosedAt`.

The authorized operations are the syntactic traversals in `Syntax.lean`.

The closure law says that the operation cannot escape the classifier's boundary.

For `substitute`, the theorem is especially sharp.

The classifier does not merely say that the output is again closed.

It says the output is the input.

Thus every classified term is a fixed point of substitution at its boundary.

The name certificate reading is then immediate.

A naming certificate promotes a stable generated behaviour to a derived interface.

The stable behaviour here is:

$$t \in \mathsf{ClosedAt}(d) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

The certificate does not license arbitrary semantic substitution.

It licenses a boundary fact about closed terms.

It records that the generator action cannot access a variable whose existence the classifier has ruled out.

The NameCert surface therefore reads the theorem as a closure classifier over the `Term` generator.

The theorem belongs to the records side of BEDC because its proof is mechanical structural recursion over a closed inductive witness.

It does not ask for a free-floating universal principle over all possible syntax.

It ranges over the closure of the declared `Term` and `ClosedAt` generators.

Within that closure, it is fully internal.

Beyond that closure, BEDC does not pretend that the theorem has authority.

The NameCert classifier says "this generator, this boundary, this closure law".

It does not say "all imaginable binders in all possible metatheories".

That distinction is the exact difference between BEDC's generator-internal induction and the forward-binding belief whose far end is the T-socket.

9372.5 Audit gate as the executable form of the classifier

The audit gate is the executable form of this classifier discipline.

The paper marker `BEDC.MetaCIC.substitute_closed` points at a concrete Lean theorem.

The project audit verifies that the target exists.

The Lean build verifies that the proof elaborates.

The strict axiom-purity audit inspects the transitive dependency closure of the accepted theorem.

If the proof of `substitute_closed` depended on a hidden substrate commitment such as `Classical.choice`, `Quot.sound`, or `propext`, the audit would reject it.

This is not a separate quality check layered on top of the mathematics.

It is the operational classifier.

The classifier says that the generator action remains inside the authorized closure.

The audit checks that the Lean witness to that statement is itself generated inside the authorized proof discipline.

A proof that smuggled in a forbidden host axiom would be a generator escape at the proof layer, even if its statement had the right shape.

The strict audit therefore enforces the same closure law one stratum higher.

The syntactic generator `substitute` must not create a free variable in a closed term.

The proof generator used by Lean must not create a hidden axiom dependency in a closed BEDC theorem.

The two checks are not merely parallel.

They are the same generator-closure classifier pattern applied at two strata.

At the term stratum, `ClosedAt` classifies admissible MetaCIC terms.

At the proof-record stratum, the axiom-purity audit classifies admissible Lean proof dependencies.

The theorem is accepted only when both strata close.

Thus the audit gate is the executable manifestation of certificate closure.

It turns the prose promise "substitution does not leak names" into a checkable record whose proof also does not leak forbidden host commitments.

9372.6 Boundary: what this lemma does not claim

`substitute_closed` does not claim that substitution is semantically correct for arbitrary open terms.

It does not claim subject reduction for the whole evaluator.

It does not claim that every compiler pass preserves every classifier.

It does not claim that BEDC internalises the host CIC's truth predicate.

It does not claim that self-compilation is completed by a single syntactic lemma.

Its claim is exact:

$$\mathsf{ClosedAt}(d, t) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

The exactness is the point.

Concretized in §538: Closed substitution is represented by a finite boundary packet whose substitute route is inert on the `ClosedAt` slice.

At the boundary where a substitution payload could otherwise enter, the closedness classifier proves that there is no target for it.

The operation returns the original term.

The generator remains inside the classifier.

The proof remains inside the axiom-pure audit gate.

The self-compilation chain receives one local certificate that no unregistered free variable enters through this operation.

`shift_closed` has the matching boundary:

$$\mathsf{ClosedAt}(n, t) \Rightarrow \mathsf{shift}(n, 1, t) = t.$$

This law does not say that every shift is harmless.

It says that the shift at the closed boundary spends no future depth budget.

Together the two theorems give the self-compilation story a precise local socket:

the compiler may pass closed source through binder movement and boundary substitution without converting closure into name leakage.

Where the story asks for more than this, it must cite further certificates.

Where it asks for a universal claim beyond any authorized generator, BEDC marks the position as a T-socket rather than internalising it.

That is why the theorem is small but load-bearing.

It is the records-side generator-closure classifier instance at the substitution step of self-compilation.

It does not complete the entire compiler story.

It closes this boundary, with this classifier, for this authorized generator, under the executable audit gate.

Concretized in §537: Closed-term substitution is realized as a finite NameCert packet for boundary-inert substitution over MetaCIC closed terms.

Concretized in §531: Closed substitution seals realize the MetaCIC closedness boundary as a finite NameCert packet over Term, ClosedAt, substitution, shift, audit, and self-compilation rows.

Concretized in §527: Closed-term substitution boundary is realized as a finite NameCert packet over Term, ClosedAt, shift, substitute, closedness ledgers, and audit-visible non-escape rows.

Concretized in §532: Binder-budget seals realize the ClosedAt boundary as a finite NameCert packet for shift and substitution non-escape.

Concretized in §540: Closed-term substitution is realized as a finite compiler-boundary NameCert over ClosedAt, substitute, shift, BHist, Cont, Pkg, and NameCert.

Concretized in §621: Closed-term substitution is realized as a finite binder-boundary substitution seal packet over Term, ClosedAt, substitute, hsame, Cont, Pkg, and NameCert.

Concretized in §8006: Closed substitution self-compile packets realize closed-term, substitution, binder-seal, replay, refusal, transport, provenance, and local NameCert rows.

Chapter 9373

The recursor as an authorized generator

This chapter identifies the host CIC recursor, the eliminator generated for each accepted `inductive` declaration, as the canonical authorized generator behind BEDC's structural proof practice. The point is not that BEDC adds a new principle to CIC. The point is that BEDC reads the kernel's existing acceptance of `I.rec` as an authorization event: once the constructor signature is accepted, every legal use of the recursor is a generator action whose output carries a named certificate of membership in the associated closure.

9373.1 The setup

In Lean 4, an `inductive` declaration presents a closed constructor signature. The declaration of `Nat` gives zero and successor; the BEDC declaration of `BMark` gives `b0` and `b1`; the BEDC declaration of `BHist` gives `Empty`, `e0`, and `e1`; the BEDC.MetaCIC declaration of `Term` gives `var`, `app`, `lam`, `pi`, and `sort`. The corresponding eliminators `Nat.rec`, `BMark.rec`, `BHist.rec`, and `Term.rec` are not later postulates. They are emitted by the kernel from the accepted constructor signatures.

The associated reading is:

$$\begin{aligned} \text{constructor signature accepted} &\Longrightarrow \text{closed generated carrier}, \\ \text{closed generated carrier} &\Longrightarrow \text{kernel eliminator } I.\text{rec}, \\ I.\text{rec} &\Longrightarrow \text{structural computation over exactly that carrier.} \end{aligned}$$

This is the pure paradigm of a closed generator. The carrier has no hidden constructor outside the signature, and the eliminator has no hidden branch outside the constructor cases. For `Term`, for example, every structurally recursive definition must pay for the five visible cases. For `BHist`, every proof by cases or induction must pay for the empty case and the two extension cases. BEDC treats this exact finite payment as the operational content of authorization.

9373.2 The kernel principle

For each `inductive` declaration I, the Lean kernel emits an elimination principle I.rec. The acceptance criteria are the ordinary kernel criteria for inductive families: the constructor signature

must be strictly positive, and recursive uses exposed through definitions and proofs must be structurally decreasing or otherwise accepted by the kernel's termination checker. These criteria are not optional annotations. They are the gate through which the signature becomes a generator.

BEDC reads the gate as follows. Strict positivity prevents a constructor from smuggling a negative self-use that would turn the carrier into an uncontrolled truth device. Structural decrease prevents recursive computation from escaping the constructor spine it claims to consume. Kernel acceptance is therefore the generator's authorization stamp, and rejection is the absence of authorization. No further axiom is injected at the paper layer.

Principle 9373.1 (Recursor authorization)**.** *Let I be a Lean* `inductive` *declaration accepted by the kernel. The emitted eliminator I.rec is an authorized BEDC generator exactly over the closed carrier generated by I's accepted constructors. A use of I.rec is authorized only when its motive and branch data are accepted by the kernel and its recursive calls remain inside the structural closure of I.*

The host kernel is thus not merely checking a proof after the fact. It is sealing the generator at signature time and checking each generator call at use time. BEDC names this combination generator authorization.

9373.3 Generator role

The operational role of `I.rec` is to consume a generated I-object and produce an output by one branch per constructor. In non-dependent form the output type is fixed. In dependent form the motive may vary over the consumed object. Either way, the generator's closure is the same: every branch is tied to a constructor, and recursive hypotheses are available only for structurally smaller constructor arguments.

For a classifier C over the carrier I, the recursor gives the canonical closure pattern:

$$\frac{\begin{array}{r}\text{base branches certify } C \text{ on primitive constructors,}\\ \text{step branches transport } C \text{ across constructor extension,}\end{array}}{\text{the recursor certifies } C(x) \text{ for every } x : I.}$$

The phrase "for every $x : I$" is not a free-floating universal over an ambient totality. It is the generator-internal universal over the closure of the accepted constructor signature. This is the same boundary as Definition 9389.10: structural induction is internal to an authorized generator; the move to an unscoped "all" is external.

9373.4 NameCert structure

A legal `I.rec` call produces more than an isolated well-typed term. It produces a term whose well-typedness is backed by the closure of the generator that produced it. In BEDC vocabulary, that output is a $\mathsf{NameCert}$ at the relevant name:

$$\begin{aligned}\mathsf{NameCert}_{I.\mathrm{rec}(M,\beta,x)} = {} & \text{the named certificate that the recursor output is generated}\\ & \text{from the accepted } I \text{ signature, motive } M, \text{ and branch package } \beta.\end{aligned}$$

The certificate is named because the output can be referenced later as a stable proof object or definition. It is a certificate because its well-typedness is not supplied by an external assertion. It

is supplied by the recursor's sealed closure: the motive is checked, each branch is checked against its constructor boundary, and the recursive occurrences are accepted only where the kernel can see structural descent.

This is why a `cases` or `induction` proof in Lean is not an ad hoc proof script in BEDC's reading. The tactic elaborates to eliminator use. The resulting term is certified because the eliminator is certified. Every such proof object is therefore a named generator output, unless it imports a forbidden external principle through its transitive dependency chain.

9373.5 Generator-closure classifier view

Given an accepted inductive I, the natural classifier is:

$$I.\mathsf{WellFormed}(x) := x \text{ is generated by the accepted constructor signature of } I.$$

Inside Lean this classifier is not usually exposed as a separate predicate for plain inductive objects, because the judgment $x : I$ already carries it. BEDC makes the classifier explicit. The generator is the family of all legal calls to I.rec. The generator-closure statement is:

$$x : I \ \wedge \ \texttt{I.rec} \text{ is called with kernel-accepted motive and branches} \quad \Longrightarrow \quad I.\mathsf{WellFormed}(\texttt{I.rec}\ x).$$

When the recursor output does not itself have type I, the classifier is read at the output type: the output is well formed at its target type because the computation was generated through an authorized eliminator and accepted branch package. Thus the classifier-closure statement is not restricted to endomaps on I. It says that recursor-generated outputs remain inside the well-formed fragment of the target judgment selected by the motive.

No axiom appears in this classifier. The closure is the kernel's internal induction over the constructor signature. The audit-relevant object is the generated proof term itself, not an external statement that the proof term should be trusted.

9373.6 Comparison with classical "by definition" reading

Classical foundational prose often treats eliminators as available "by definition": once the inductive type is declared, induction and recursion are simply part of what the declaration means. That reading is extensionally correct but hides the authorization event. BEDC keeps the event visible.

In BEDC terms, "by definition" means:

$$\text{authorized at signature time} \quad + \quad \text{sealed at kernel-acceptance time} \quad + \quad \text{checked at each eliminator call.}$$

The difference is not a stronger logical assumption. BEDC does not place a deeper metaphysical principle underneath Lean's recursor. It records the operational structure already present in the kernel: strict positivity, constructor closure, structural descent, branch completeness, and type checking of the produced term. Classical prose compresses these into definitional availability. BEDC expands the compression into generator authorization and NameCert output, with the GAP ledger marking any remaining handoff to the host kernel as a T-socket in the sense of Definition 9389.15.

9373.7 Connection to closed-term substitution

The closed-term substitution lemma in BEDC.MetaCIC gives a direct analogue. The theorem `substitute_closed` says that substituting at a closed boundary does not change a closed term:

↪ variant BEDC.MetaCIC.substitute closed

The companion `shift_closed` theorem has the same closure shape for shifting:

↪ variant BEDC.MetaCIC.shift closed

Both are packaged as concrete instances of the generator-closure classifier structure:

↪ variant BEDC.MetaCIC.GeneratorClosureClassifier

↪ variant BEDC.MetaCIC.closed under shift classifier

↪ variant BEDC.MetaCIC.closed under substitute classifier

.

The structure is:

$$\begin{aligned} \text{classifier} &:= \texttt{ClosedAt}\ d\ t, \\ \text{generator} &:= \texttt{substitute}\ d\ v, \\ \text{closure} &:= \texttt{substitute}\ d\ v\ t = t \text{ whenever } \texttt{ClosedAt}\ d\ t. \end{aligned}$$

The proof proceeds by structural induction over the `ClosedAt` certificate. Each branch corresponds to one visible constructor of the classifier: variable, application, lambda, Pi, or sort. The proof is therefore a recursor-generated NameCert for the claim that the substitution generator cannot escape the closed-term classifier.

The `I.rec` case is the same pattern at the host-CIC level:

$$\begin{aligned} \text{closed-term substitution:} &\quad \texttt{ClosedAt} \text{ classifier closed under } \texttt{substitute}, \\ \text{inductive recursor:} &\quad \texttt{WellTyped} \text{ classifier closed under legal } I.\texttt{rec} \text{ calls.} \end{aligned}$$

The two claims are structurally isomorphic. In both cases, a classifier is preserved because the generator is constrained by the same constructor closure that produced the classifier. In both cases, no axiom is appended from outside the proof term.

9373.8 Audit gate as the executable form of the classifier

The executable form of the generator-closure classifier is the audit gate. Running `lake build` asks the Lean kernel to re-check every generated proof term: every `inductive` signature, every emitted or consumed recursor, every branch package, every definitional computation, and every theorem body. If a term depends on an unauthorized generator or fails its classifier, the kernel does not elaborate it.

BEDC adds two visible constraints on top of ordinary build success. The source audit rejects project-level `axiom` declarations, and the purity audit rejects forbidden transitive dependencies such as `Classical.choice`, `Quot.sound`, and `propext`. These gates make the classifier objective: the paper does not merely claim that recursor outputs stay inside their closure; the build graph and dependency audit decide whether the claim survives execution.

Thus the BEDC pipeline implements:

$$\begin{aligned} \texttt{lake build} &= \text{kernel execution of generator-closure checking}, \\ \texttt{check-axioms.py} &= \text{source-level rejection of axiom backdoors}, \\ \texttt{axiom-purity} &= \text{transitive rejection of forbidden host principles.} \end{aligned}$$

Together these gates state that the closure does not leak. A recursor-induced NameCert is accepted only if the kernel can replay the construction and the audit can find no axiom channel behind it.

9373.9 Boundary: what is not a recursor-induced NameCert

Not every well-typed Lean term is a recursor-induced NameCert in the strict BEDC sense. A proof whose transitive dependencies include `Classical.choice` imports a choice witness not generated by the relevant inductive closure. A proof depending on `Quot.sound` imports quotient equality as a host axiom. A proof depending on `propext` imports propositional extensionality. Such terms may be accepted by a broader Lean environment, but they are not BEDC-authorized generator outputs.

BEDC's project discipline forbids this family. The mathlib-free Lean tree, the no-`axiom` source gate, and the strict purity audit keep BEDC lemmas inside the recursor-and-constructor fragment. Within that fragment, a public lemma is generated either directly by constructors, by definitions unfolded over constructors, or by recursor use accepted by the kernel. Its certificate is therefore recursor-induced or constructor-induced, and its name is a NameCert rather than an imported classical closure.

This boundary is the practical content of the chapter. The CIC recursor is not an unexamined magic rule inside BEDC. It is the primary authorized generator: accepted by strict positivity, sealed by kernel checking, executed by structural recursion, named by its generated proof term, and audited so that no axiom backdoor can masquerade as closure.

Concretized in §522: The recursor-as-generator vision is realized as a finite NameCert packet for recursor-induced proof and definition outputs.

Concretized in §525: Authorized generator recursors are realized as finite NameCert packets over accepted constructor signatures, branch coverage, structural descent, and audit provenance.

Concretized in §534: Concrete recursor-generator packet for accepted inductive signatures, legal eliminator calls, audit non-escape, and closed-term handoff.

Concretized in §529: Recursor authorization is realized as a finite NameCert packet for constructor signatures, emitted eliminators, branch packages, and generator non-escape.

Concretized in §541: Authorized recursor packets realize legal eliminator calls as finite NameCert carriers over constructor signatures, motives, branches, output rows, and audit non-escape ledgers.

Concretized in §710: Recursion authorization ledgers realize accepted recursor calls as finite BHist packets over signature, motive, branch, descent, output, audit, and non-escape rows.

Concretized in §857: Bounded recursor replay is realized as finite BHist packets for signature, motive, branch, descent-budget, output, audit, route, provenance, and NameCert rows.

Concretized in §974: Recursor branch audits realize motive checking, constructor-indexed branch coverage, structural descent, and generator non-escape as a finite NameCert packet.

Concretized in §1187: Recursor closure classifier audits are realized as finite recursor, branch, classifier, audit-replay, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1345: Recursor branch-completeness packets realize authorized generator use as finite BHist, branch ledger, transport, replay, provenance, and NameCert data.

Concretized in §8079: Finite candidate-diamond packets realize authorized recursor generation as local MetaCIC join-audit ledgers over critical-path, parallel-frontier, and candidate-boundary rows.

Chapter 9374

Subject reduction discharge routes: explicit bundle

9374.1 The setup

BEDC.MetaCIC subject reduction for the closed CIC substrate is not an unparameterised theorem of the substrate itself. As recorded in chapter 9377, the checked theorem is parameterised by four named discharge hypotheses. A downstream carrier may supply those hypotheses explicitly, but the present raw relation does not supply a global discharge packet.

The public route is the explicit bundle route. It packages the four hypotheses as finite evidence and passes that packet to the subject-reduction theorem. The current substrate also exports checked obstruction facts showing that the application-argument, lambda-domain, and Pi-domain fields cannot be filled for the raw relation, and therefore the full bundle is not inhabited at this boundary.

9374.2 Bundle route

9374.2.1 Definition

The bundle route is centred on

↪ variant `BEDC.MetaCIC.SubjectReductionDischargeBundle`

. This Lean object is a structure of four propositions. Its fields are the beta-substitution preservation obligation, the application-argument stability obligation, the lambda-domain subject-reduction obligation, and the Pi-domain subject-reduction obligation.

The structure is carrier-agnostic: it does not say where the four propositions come from, which substrate implements them, or how an implementation proves them. Any caller that can provide the four fields can build the packet. No ambient instance route is part of the public MetaCIC audit surface.

9374.2.2 Subject reduction from bundle

The theorem

↪ variant `BEDC.MetaCIC.subject reduction from bundle`

consumes a

↪ variant `BEDC.MetaCIC.SubjectReductionDischargeBundle`

and yields the subject-reduction conclusion for the represented reduction step. The theorem does not prove the four hypotheses. It records that once those four hypotheses are explicitly supplied, the parameterised subject-reduction theorem has no remaining open argument.

This route is direct at a call site. Code constructs a bundle, passes it to the theorem, and receives the subject-reduction result. The discharge evidence is visible in the term that invokes the theorem.

9374.3 Checked obstruction ledger

The raw relation supplies checked negative facts for three fields:

↪variant `BEDC.MetaCIC.not appArgTypeStable`

,

↪variant `BEDC.MetaCIC.not lamDomainSubjectReduction`

, and

↪variant `BEDC.MetaCIC.not piDomainSubjectReduction`

. These facts are kernel-checked consequences of the closed counterexamples in the MetaCIC subject-reduction audit.

The bundle obstruction

↪variant `BEDC.MetaCIC.not subjectReductionDischargeBundle`

then follows by projecting the application-argument field from any purported bundle. Thus the current raw relation has a checked no-bundle ledger. This is a negative theorem about the present substrate, not a replacement for the beta-substitution obligation in future carriers.

9374.4 Boundary

This chapter does not solve beta-substitution preservation and does not construct a

↪variant `BEDC.MetaCIC.SubjectReductionDischargeBundle`

for the present substrate. It records one public discharge route and the checked obstruction ledger explaining why that route is not inhabited by the raw relation.

Future downstream carriers can still consume

↪variant `BEDC.MetaCIC.subject reduction from bundle`

by providing an explicit

↪variant `BEDC.MetaCIC.SubjectReductionDischargeBundle`

. At this boundary, however, the canonical published surface is explicit: bundle consumption plus checked no-bundle obstruction.

Concretized in §783: The explicit bundle discharge route is represented as a finite BEDC trace packet.

Concretized in §785: The explicit bundle discharge route is realized as a finite NameCert classifier over the shared subject-reduction discharge rows.

Concretized in §789: The explicit subject-reduction discharge route is realized as a finite NameCert route-audit packet with four-row payload and obstruction non-escape.

Concretized in §787: The explicit bundle subject-reduction discharge route is realized as one finite BEDC route packet.

Concretized in §784: Subject-reduction route equivalence is realized as finite bundle, socket, hsame, Cont, Pkg, and NameCert rows.

Concretized in §790: The explicit bundle discharge route is realized as a finite NameCert route certificate.

Concretized in §793: The explicit discharge-route comparison is represented as a finite route-audit NameCert packet.

Concretized in §796: The bundle/obstruction triangle is represented as a finite route-triangle NameCert packet.

Concretized in §798: The explicit bundle subject-reduction discharge route is realized as a finite route-choice NameCert packet with the obstruction boundary still explicit.

Concretized in §4200: The explicit bundle discharge route is realized as a finite dependency-audit packet over route, audit, triangle, and open-frontier rows.

Concretized in §5561: The subject-reduction discharge route is realized as a finite subject-reduction kernel packet.

Chapter 9375

Closed-MetaCIC consistency assembly: a partial closure map

9375.1 The setup

BEDC.MetaCIC aims at unconditional closed-term consistency of the substrate's internal CIC fragment. The canonical path is the standard one: subject reduction, strong normalisation on closed terms, and no closed normal proof of False. This chapter records which pieces of the path are already discharged and which constitute the open frontier.

The relevant distinction is between closed-normal consistency and full closed consistency. Closed-normal consistency speaks only about closed proof terms that are already beta-strong. Full closed consistency would also need to transport an arbitrary closed proof term along reduction until the normal-form theorem can be applied. That transport is exactly the subject-reduction boundary.

9375.2 The assembled positive content

9375.2.1 Closed-term strong normalisation

The strong-normalisation content is available through the candidate-mediated closed-term interface. The substrate verifies that an empty-context typed closed term is strongly normalising when the appropriate candidate evidence is supplied:

```
↪ variant  BEDC.MetaCIC.closed typed term sn from candidate
↪ variant  BEDC.MetaCIC.closed typed term sn from closedCandidate
↪ variant  BEDC.MetaCIC.closed typed term sn from candidateSet
```

.

These theorems are not themselves a closed subject-reduction theorem. They are the normalisation side of the assembly path. They say that the candidate interface can deliver termination of closed typed terms once the term has been placed inside the relevant reducibility certificate.

9375.2.2 No closed-normal proof of False

The contradiction-side theorem is also available:

```
↪ variant  BEDC.MetaCIC.no closed normal proof of false
```

. The substrate verifies that any closed proof term whose beta reduction is already strong cannot inhabit the internal false encoding.

In formula form, the checked content has the shape:

$$\mathsf{Closed}(t) \wedge \mathsf{BetaStrong}(t) \wedge \mathsf{HasType}([], t, \mathsf{False}) \Rightarrow \bot.$$

The theorem is closed over normal-form inhabitants of False. It does not need to assert that every closed proof term can be reduced while preserving its displayed type; that assertion belongs to the subject-reduction frontier.

9375.2.3 Closed-normal consistency assembly

The substrate also has the direct assembly statement:

```
↪variant BEDC.MetaCIC.closed normal consistency assembly
```

. It composes empty-context typing, closedness, and beta-strong normality into a single audit-checkable statement.

The theorem is the positive closed-normal node of the path:

$$\mathsf{HasType}([], t, \mathsf{False}) \wedge \mathsf{Closed}(t) \wedge \mathsf{BetaStrong}(t) \Rightarrow \bot.$$

Thus BEDC.MetaCIC already establishes the consistency statement for the normal-form fragment. The remaining question is whether every closed proof of False can be connected to that fragment without losing the typing certificate.

9375.3 The unassembled frontier

9375.3.1 Conditional subject reduction

Subject reduction is available as a parameterised theorem in BEDC.MetaCIC, conditional on four discharge hypotheses. Those hypotheses govern beta substitution preservation, application-argument codomain stability, lambda-domain preservation, and Pi-domain preservation. The interface is mapped in chapter 9377.

This is the exact point where the closed-normal theorem would have to become a full closed consistency theorem. A closed proof term of False must be normalised, and the displayed false type must remain attached throughout the reduction path. Without that preservation step, the normal-form theorem cannot be applied to an arbitrary closed proof term.

9375.3.2 The closed reduction obstruction

The substrate proves the closed reduction obstruction:

```
↪variant BEDC.MetaCIC.closed consistency reduction obstruction not available
```

. This theorem establishes that closed-term subject reduction is not directly available on the present substrate. Consequently unconditional closed-term consistency via the standard subject-reduction path is blocked at that node.

The obstruction does not weaken the positive closed-normal theorem. It locates the missing lift. The available consistency theorem applies once beta-strong normality is already part of the input; the unavailable theorem would justify transporting arbitrary closed proofs to that input form.

9375.4 The assembly diagram

The dependency map is:

```
Closed-term SN
  status: AVAILABLE
  witnesses:
    BEDC.MetaCIC.closed_typed_term_sn_from_candidate
    BEDC.MetaCIC.closed_typed_term_sn_from_closedCandidate
    BEDC.MetaCIC.closed_typed_term_sn_from_candidateSet

No closed-normal proof of False
  status: AVAILABLE
  witness:
    BEDC.MetaCIC.no_closed_normal_proof_of_false

Closed-normal consistency
  status: AVAILABLE
  witness:
    BEDC.MetaCIC.closed_normal_consistency_assembly

Closed reduction by subject reduction
  status: BLOCKED
  obstruction:
    BEDC.MetaCIC.closed_consistency_reduction_obstruction_not_available

Full closed consistency
  required inputs:
    Closed-normal consistency
    Closed reduction by subject reduction
  status: BLOCKED
  reason:
    the subject-reduction discharge is blocked
```

The assembled node is therefore closed-normal consistency, not full closed consistency. BEDC.MetaCIC establishes the normal-form sub-consistency unconditionally. It cannot lift that statement to arbitrary closed proofs of False without discharging the subject-reduction obligations.

Concretized in §6162: Closed-consistency reduction obstruction packets realize the blocked lift from arbitrary closed false proofs to beta-strong endpoints as finite endpoint, discharge, obstruction, transport, replay, provenance, and NameCert rows.

9375.5 What this partial closure does establish

The partial closure that already holds is strong. Any closed proof term that is beta-strong cannot inhabit the internal False encoding. Equivalently, the substrate rules out a closed false proof inside the normal-form fragment.

This is a non-trivial statement about normal-form inhabitants of False. The substrate has established not no-consistency-at-all but consistency on the normal-form fragment. The positive

theorem is narrow because its input includes beta-strong normality, and it is sharp because its conclusion is contradiction from a closed false proof under exactly that input.

9375.6 What discharge would buy

Discharging all four subject-reduction obligations would buy the lift from closed-normal consistency to unconditional closed consistency. The lift would have the following shape:

$$\mathsf{HasType}([], t, \mathsf{False}) \wedge \mathsf{Closed}(t) \Rightarrow \mathsf{HasType}([], n, \mathsf{False}) \wedge \mathsf{Closed}(n) \wedge \mathsf{BetaStrong}(n)$$

for a normal endpoint n reached from t. The closed-normal consistency assembly would then eliminate the right-hand side. The discharge would have to traverse the dependent-codomain boundary identified in chapter 9377.

9375.7 Apophatic framing

The partial closure is itself an apophatic name. The substrate names what it has and what it does not have, sharply. The unassembled frontier is not hidden behind a proof hole; it is documented as four named hypotheses and an explicit obstruction theorem.

This is the BEDC discipline applied to its own meta-theory: name the open cells, and do not cover them with an informal appeal to the intended calculus. The reader can see the positive assembled theorem, the blocked subject-reduction node, and the exact reason full closed consistency is not claimed.

Concretized in §733: Closed-normal MetaCIC consistency is realized as a finite boundary packet separating beta-strong consistency from the blocked subject-reduction lift.

9375.8 Boundary

This chapter records the present assembly state, not a proposal for discharging the four obligations. The discharge problem is open and resides at the dependent-codomain frontier identified in chapter 9377.

Concretized in §732: The normal-form false-proof exclusion is exposed as a concrete NameCert packet without claiming full closed consistency.

Concretized in §736: Closed-normal consistency is represented as a finite NameCert boundary separated from the subject-reduction obstruction.

Concretized in §738: Closed-consistency reduction-boundary packets realize the frontier between closed-normal consistency and full closed consistency as finite subject-reduction, normal-endpoint, obstruction, and NameCert rows.

Concretized in §739: Closed-normal consistency packets realize the MetaCIC assembly boundary as finite closedness, typing, beta-strong normality, contradiction, and subject-reduction obstruction rows.

Concretized in §740: The closed-normal consistency assembly is realized as a finite normal-form seal packet with an explicit subject-reduction boundary.

Concretized in §1177: Closed-MetaCIC consistency assembly is realized as a finite composite NameCert packet binding normal-form exclusion, subject-reduction boundary, obstruction, ledger, and local naming rows.

Chapter 9376

Closed-normal consistency: BEDC's first unconditional main result

9376.1 The result

The BEDC project uses the closed-CIC substrate as its internal metatheoretic base. Within that substrate, the strongest presently established unconditional consistency result is closed-normal consistency. If t is a closed term, t has the internal false encoding as its type, and t is beta-strong, then t is impossible:

↪ variant `BEDC.MetaCIC.closed normal consistency assembly`

.

The checked statement has the following shape:

$$\mathsf{HasType}([], t, \mathsf{falseEncoding}) \wedge \mathsf{ClosedAt}(0, t) \wedge \mathsf{BetaStrong}(t) \Rightarrow \bot.$$

The three hypotheses are not interchangeable. The typing row says that the term is displayed at the internal false encoding. The closedness row says that the term is not depending on an external context entry. The beta-strong row says that no beta step can still be taken from the displayed term. The conclusion is contradiction inside the substrate's Lean formalisation.

Equivalently, the substrate has no closed beta-normal proof of its internal false type:

↪ variant `BEDC.MetaCIC.no closed normal proof of false`

. The target is not merely a line in the assembly map of chapter 9375; following the operational priority in chapter 9386, this chapter treats that node as a first-class BEDC main result.

The proposition that full closed consistency would target is separately named:

Lean4 (definition) `BEDC.MetaCIC.no closed proof of false`

. That definition removes the beta-strong input. The distinction between the two statements is the central boundary of this chapter. Closed-normal consistency is checked as a theorem. Full closed consistency is a stronger proposition whose missing transport step is documented elsewhere.

9376.2 Why this is a first-class main result

The result is non-trivial for exactly the reason that full closed consistency is not claimed. It is not a corollary of closed subject reduction. Closed subject reduction is documented as unavailable on the present substrate in chapter 9377. The dependent-codomain obstruction prevents an arbitrary

closed proof term from being transported through beta reduction while preserving the displayed false type.

Closed-normal consistency bypasses that blocked lift. It works directly at the normal-form boundary. Its input already contains beta-strong normality, so the proof does not consume the four subject-reduction discharge hypotheses, does not require dependent-codomain inversion, and does not import a quotient or Church–Rosser theorem to repair the missing transport step.

This is why the result deserves main-result status rather than being treated as a minor helper lemma. It is the largest consistency statement that the present closed-CIC substrate supports without additional setup fields. The theorem touches the object of interest, the internal false encoding, and it eliminates that object under exactly the normal-form hypotheses that the substrate can audit.

The result also has the right failure behaviour. If a reader asks for arbitrary closed consistency, the chapter can point to the missing subject-reduction transport. If a reader asks what is already established without that transport, the chapter can point to a single Lean theorem. The paper therefore has a central positive statement and a central boundary statement, instead of an ambiguous promise that depends on informal CIC intuition.

This makes the theorem an unconditional positive result of the current BEDC.MetaCIC substrate. The substrate cannot yet say that every closed proof of false is impossible. It can say, without any additional logical commitment, that a closed proof already in beta-normal form cannot inhabit false.

9376.3 Constructive content

The result has a concrete audit reading. Given a represented closed term t, the decidability layer supplies the mechanical typing surface described in chapter 9379. In particular, the `inferTypeCtxClosed` interface is sound and complete for empty-context typing at the stated `HasType` relation.

The checker contribution is intentionally only a typing contribution. A successful checker run exposes a type judgement for the represented relation, and the completeness theorem says that represented typing judgements are seen by the checker. This is enough for the first row of the main result. It does not decide every semantic question about the term, and it does not decide the full conversion theory of the substrate.

Given a well-typed closed term t, the normalisation layer supplies candidate-mediated strong normalisation when the relevant candidate evidence is present:

```
↪variant  BEDC.MetaCIC.closed typed term sn from candidate
```

. The audit map in chapter 9383 keeps that evidence visible. It is not erased into a global theorem over every closed typed term.

The normalisation contribution is likewise exact. Candidate evidence gives a strong-normalisation witness. A beta-strong endpoint is already at the normal-form boundary required by the consistency theorem. In the cases where the audit chain supplies candidate evidence and a beta-strong endpoint, the theorem closes the false-inhabitation question at once.

Given a closed beta-strong well-typed term t, the present main result rules out inhabitation of the false encoding. The pipeline is therefore:

1. use the checker layer to expose the empty-context typing judgement;
2. use candidate-mediated normalisation when the candidate witness is part of the audited input;
3. apply the closed-normal consistency theorem at the beta-strong boundary;

4. keep every Lean marker tied to a kernel-checked BEDC.MetaCIC declaration.

This constructive content is deliberately narrow. It is Lean-kernel verified, and the BEDC audit discipline requires it to remain independent of forbidden stdlib logical dependencies. It is also explicit about its inputs: typing, closedness, and beta-strong normality are separate rows, not prose assumptions hidden inside a single word such as consistency.

This separation is important for reproducibility. A consumer can inspect which row is being supplied by a checker theorem, which row is supplied by closedness evidence, which row is supplied by normal-form evidence, and which row is discharged by the consistency theorem. No row is silently converted into another. The proof is therefore useful even when the surrounding metatheory is still being audited, because it tells the reader exactly what data reaches the contradiction.

9376.4 Relation to classical results

Classical CIC consistency is usually read through a three-part route: strong normalisation, Church–Rosser or subject reduction, and a no-false-inhabitant argument for normal forms. That route eliminates arbitrary proof terms by reducing them to a normal endpoint while preserving the relevant type.

BEDC's closed-normal statement is the normal-form version of that route. It contains the no-false-inhabitant endpoint, and it can be paired with the candidate-mediated normalisation facts where their evidence is available. It does not contain the subject-reduction transport from arbitrary closed proofs to typed normal endpoints.

The comparison can be read as a factorisation:

$$\text{arbitrary closed proof} \longrightarrow \text{typed normal endpoint} \longrightarrow \bot.$$

Classical presentations often package the first arrow through subject reduction, confluence, and normalisation. BEDC's theorem establishes the second arrow for the closed beta-strong endpoint. The first arrow is not available unconditionally in the present substrate.

The result is therefore weaker than full classical closed consistency, but it is strictly stronger than having no consistency theorem. The subject-reduction counterexample prevents the paper from elevating arbitrary closed consistency on the current substrate. The closed-normal theorem is the maximal positive consistency claim available without crossing that obstruction.

The word "maximal" is used here only relative to the stated substrate and the stated obstruction. It does not mean that stronger metatheorems are mathematically impossible. It means that the paper has a checked theorem at the normal boundary and a checked reason not to erase the reduction boundary. That is exactly the main-result shape that BEDC can honestly export.

9376.5 What it does not claim

The theorem does not claim full closed consistency:
Lean4(definition) `BEDC.MetaCIC.no closed proof of false`

. That proposition is the stronger statement excluding every closed proof of false, whether or not it is already beta-strong. The present chapter does not mark it as checked.

It does not claim that every closed proof term has a displayed false type preserved along every beta path. Such a statement would be a subject-reduction or typed-normal-endpoint theorem. The current closed-normal theorem starts after that transport problem has already been resolved in the input.

It also does not claim a global theorem that all closed typed terms reduce to normal form without candidate-mediated evidence. The normalisation audit map records checked projections from candidate evidence, closed-candidate evidence, and candidate-set membership. Those projections are central to the positive pipeline, but they are not a blanket closed-term strong-normalisation theorem.

It does not claim unrestricted closed beta-conversion decidability. The bounded normal-equality checker described in chapter 9379 has its own finished-normalisation boundary. That checker is part of the audit surface, not a replacement for the missing closed-consistency transport.

Finally, the result does not solve the dependent-codomain obstruction. The four subject-reduction discharge hypotheses remain exactly where chapter 9377 and chapter 9386 place them. Closed-normal consistency is a main result because it is clean at its boundary, not because it hides the boundary.

These non-claims are not qualifications that weaken the theorem. They are the conditions that make the theorem readable. A full consistency claim would need different hypotheses and different Lean evidence. The closed-normal claim has the evidence it needs and stops there.

9376.6 The role of the BEDC project

BEDC's current mature metatheoretic deliverable is this closed-normal consistency theorem. It is produced by the MetaCIC substrate, the normal-form audit chain, the explicit decidability surface for represented typing, and the apophatic discipline that refuses to blur open obligations into finished proofs.

The project role is therefore not merely documentary. BEDC turns a blocked classical route into a precise theorem plus a precise boundary. The theorem is not an announcement that all conventional consistency infrastructure has been internalised. It is the part of that infrastructure that the present substrate can state, check, and maintain as a public result.

This also gives the surrounding audit chapters a common endpoint. The subject-reduction chapter explains why arbitrary closed consistency is blocked. The normalisation chapter explains what evidence supplies termination. The decidability chapter explains the typing-checker surface. The assembly chapter shows the route. The present chapter names the resulting theorem as a main BEDC deliverable.

The role of chapter 9400 is positive here. By naming the missing subject-reduction lift and the candidate-mediated normalisation boundary, the paper can elevate the unconditional theorem without overstating it. The negative names make room for the positive result: closed-normal consistency is visible precisely because full closed consistency is not silently claimed.

In that sense, closed-normal consistency is the first main result with the right BEDC shape. It is mathematically specific, Lean-linked, substrate-local, and bounded by named non-claims. A reader can cite it without inheriting an unstated subject-reduction promise.

9376.7 Boundary

This chapter elevates no statement through a proof hole. It does not pretend that the subject-reduction obstruction has been discharged. It does not identify normal-form consistency with full closed consistency. It records BEDC's first-class main consistency result at its exact checked strength: closed, typed, beta-strong terms do not inhabit the internal false encoding.

The boundary can be summarised as:

$$\text{closed} + \text{typed at false} + \text{beta-strong} \Rightarrow \bot,$$
$$\text{closed} + \text{typed at false} \not\Rightarrow_{\text{present substrate}} \bot.$$

The first line is the main result of this chapter. The second line is the refused overclaim. Keeping both lines visible is the point of the closed-normal consistency elevation.

Concretized in §764: NameCert surface for the closed-normal consistency main-result boundary.

Concretized in §799: Closed-normal canonicity certificate for the false-endpoint exclusion boundary.

Concretized in §837: Endpoint-ledger NameCert for the closed beta-strong false-exclusion boundary.

Concretized in §858: Closed-normal route certificates realize the main-result normal endpoint as a finite NameCert route through false exclusion, candidate evidence, and subject-reduction obstruction rows.

Concretized in §916: Closed-normal endpoint transport is realized as a finite NameCert packet for source proof rows, beta-route rows, typed normal endpoints, false-type preservation sockets, subject-reduction obstruction, and non-escape ledgers.

Concretized in §936: ZNormal normal-form predicates are realized as finite BHist, hsame, Cont, Pkg, and NameCert rows supporting the closed-normal consistency boundary.

Concretized in §3661: Closed-normal false-exclusion witnesses are realized as finite MetaCIC typing, closedness, beta-strong endpoint, and false-encoding rows.

Concretized in §5780: MetaCIC normal endpoint comparators realize the closed-normal consistency comparator surface as finite closed-endpoint, false-exclusion, determinacy, transport, replay, provenance, and local NameCert rows.

Concretized in §6844: finite consistency-observation budgets are represented as BEDC NameCert rows

Chapter 9377

MetaCIC subject-reduction obstruction: an audit map

9377.1 The setup

The BEDC.MetaCIC formalisation aims at a closed-term consistency theorem for the substrate's internal CIC fragment. The path is the canonical one: subject reduction, strong normalisation on closed terms, and no closed normal proof of False. This chapter records the present audit state of that path, both what is already discharged and what remains as an explicit obligation.

The audit point is narrow. BEDC does not hide the subject-reduction gap behind a silent proof hole. It exposes a parameterised theorem, names the four discharge hypotheses, records the partial closed-term normalisation and closed-normal consistency facts that are already checked, and proves an explicit obstruction theorem for the stronger closed subject-reduction statement in the current substrate.

9377.2 What is already proven

The checked inventory has four parts.

- Closed-term strong normalisation is available through the candidate-mediated closed-term interface. A closed typed term is shown strongly normalising when the corresponding candidate, closed candidate, or candidate-set membership witness is supplied:

  ↪ variant BEDC.MetaCIC.closed typed term sn from candidate

  ↪ variant BEDC.MetaCIC.closed typed term sn from closedCandidate

  ↪ variant BEDC.MetaCIC.closed typed term sn from candidateSet

  .

- No closed-normal proof of False is available for closed terms whose proof term is already beta-strong:

  ↪ variant BEDC.MetaCIC.no closed normal proof of false

  .

- Closed-normal consistency assembly is checked from empty-context typing, closedness, and beta-strong normality:

```
↪ variant BEDC.MetaCIC.closed normal consistency assembly
```

.

- Subject reduction is checked as a parameterised theorem conditional on the four discharge hypotheses:

```
↪ variant BEDC.MetaCIC.subject reduction
```

.

Thus the substrate already closes the normal-form contradiction side and the parameterised preservation theorem. It does not close the unparameterised closed subject-reduction theorem.

9377.3 The four discharge obligations

The four hypotheses are not decorative assumptions. They are the named discharge obligations of the represented reduction system.

Definition 9377.1 (BetaSubstitutionPreservation)**.** BetaSubstitutionPreservation *is the beta-redex obligation. It says that if a body has type cod in the context extended by the shifted domain, and the argument has the unshifted domain type in the original context, then substituting the argument into the body has the substituted codomain type in the original context.*

This is the only obligation consumed by the direct beta case of the parameterised subject-reduction theorem. It is the place where substitution must preserve typing across the binder boundary.

Definition 9377.2 (AppArgTypeStable)**.** AppArgTypeStable *is the application-argument congruence obligation. It says that for a typed function* $f : \Pi dom.\, cod$*, a typed argument* $a : dom$*, and a reduct* $a' : dom$*, substituting* a' *into cod gives the same codomain as substituting* a *into cod.*

This obligation is consumed when reduction steps inside the argument of an application. The function type remains the same, so the target application can be typed at the original displayed result type only if the codomain readback is stable under replacing the argument by its reduct.

Definition 9377.3 (LamDomainSubjectReduction)**.** LamDomainSubjectReduction *is the lambda-domain congruence obligation. It says that if* $\lambda d.\, b$ *has a type* A *in a context* Γ*, and the displayed domain* d *takes one beta step to* d'*, then* $\lambda d'.\, b$ *still has type* A *in the same context.*

This obligation is consumed by the lambda-domain reduction constructor. It is separate because reducing the displayed domain changes the binder annotation while leaving the body and the claimed external type fixed.

Definition 9377.4 (PiDomainSubjectReduction)**.** PiDomainSubjectReduction *is the Pi-domain congruence obligation. It says that if* $\Pi d.\, c$ *has a type* A *in a context* Γ*, and the displayed domain* d *takes one beta step to* d'*, then* $\Pi d'.\, c$ *still has type* A *in the same context.*

This obligation is consumed by the Pi-domain reduction constructor. It is separate because reducing a Pi domain changes the context against which the codomain was originally checked.

9377.4 The explicit obstruction theorem

Principle 9377.5 (Closed subject reduction is documented as not available in the present substrate)**.** *The current substrate proves that the closed subject-reduction statement is not available. The theorem is a checked negation of the closed consistency reduction obstruction proposition, whose content is the closed subject-reduction statement itself:*
Lean4✓ `BEDC.MetaCIC.closed consistency reduction obstruction not available`

The obstruction is not a prose caution. It is a Lean theorem. Consequently the manuscript must not cite a blanket closed subject-reduction theorem for BEDC.MetaCIC. The available preservation theorem is the parameterised theorem, and the unparameterised closed statement is blocked by the checked obstruction.

9377.5 Critical path

Discharging any one of the four obligations does not unblock full subject reduction. All four are needed simultaneously. The four cover the reduction shapes that remain open at the subject-reduction interface:

- the beta step, governed by BetaSubstitutionPreservation;
- congruence into the App argument, governed by AppArgTypeStable;
- congruence into the Lam domain, governed by LamDomainSubjectReduction;
- congruence into the Π domain, governed by PiDomainSubjectReduction.

The parameterised theorem consumes exactly these four inputs. The obstruction theorem says that the project cannot replace that four-input interface by the closed subject-reduction statement in the present substrate.

9377.6 Where this fits in BEDC's apophatic framing

BEDC names the unknown rather than hides it. The four hypotheses, the parameterised subject-reduction theorem, and the explicit no-go obstruction theorem together form an apophatic name for the closed-term subject-reduction discharge problem: the substrate names what it cannot yet prove and audits where the proof would need to land.

This is the same discipline described in chapter 9400 and chapter 9389. An unresolved discharge obligation is not converted into an axiom, a hidden host theorem, or an informal appeal to the intended calculus. It remains a named boundary with a precise interface.

9377.7 The dependent-codomain frontier

The closed application case has a checked positive side. If the function is typed at $\Pi dom.\, cod$, the original argument and its reduct are both typed at dom, the function and original argument are closed, and the codomain itself is closed independently of the argument, then the reduced application keeps the original displayed result type:
Lean4✓ `BEDC.MetaCIC.subject reduction closed app independent codomain`

The dependent side is different. There is a closed function, a closed reducing argument, and a closed application beta step for which the original application has a type A, the reduct exists, and the reduct does not have type A:

Lean4✓ `BEDC.MetaCIC.subject reduction closed app counterexample`

Thus the obstruction is not brute failure of closed application preservation. It is a precise frontier. Closed application subject reduction is discharged in the independent-codomain regime, where substituting the reduct argument into the codomain reads back the same result type. It fails in the dependent-codomain regime when the codomain observes the argument strongly enough for substitution after reduction to change the displayed type.

This frontier places the four discharge obligations on the dependent side of the boundary. AppArgTypeStable must supply the missing codomain-readback stability for argument congruence. BetaSubstitutionPreservation must carry typing through the beta-redex binder. LamDomainSubjectReduction and PiDomainSubjectReduction must preserve the displayed binder annotations when reduction enters domains. Without those inputs, the dependent-codomain counterexample blocks any unconditional closed subject-reduction theorem for the present substrate.

9377.8 Boundary

This chapter does not claim a proof of any of the four hypotheses. It claims only that the audit map is complete at the current interface and that the critical path is sharply identified. The work of discharging the hypotheses is the substrate's outstanding meta-theoretic obligation.

Concretized in §716: Beta-substitution preservation is realized as a finite NameCert packet for the beta-redex typing boundary consumed by the MetaCIC subject-reduction interface.

Concretized in §717: MetaCIC discharge-obligation ledgers realize the four subject-reduction hypotheses and checked obstruction boundary as a finite NameCert audit packet.

Concretized in §718: SubjectReductionDischargeUp realizes the four MetaCIC subject-reduction discharge obligations as a finite NameCert packet with an explicit obstruction boundary.

Concretized in §719: Finite NameCert socket for the four MetaCIC subject-reduction discharge obligations and the checked obstruction boundary.

Concretized in §723: The MetaCIC subject-reduction obstruction is realized as a finite NameCert packet for the parameterized theorem, four discharge obligations, checked obstruction row, and non-escape ledger.

Concretized in §720: Subject-reduction discharge ledgers realize the four MetaCIC preservation obligations as finite BHist, hsame, Cont, Pkg, and NameCert rows consumed by SubjectReductionUp.

Concretized in §712: The four MetaCIC subject-reduction discharge obligations are realized as a finite NameCert packet separating parameterised preservation, obstruction, replay, and non-escape rows.

Concretized in §724: Packages the four MetaCIC subject-reduction discharge obligations as a finite NameCert carrier.

Concretized in §725: The four MetaCIC subject-reduction discharge obligations are represented as a finite NameCert obligation packet.

Concretized in §726: The MetaCIC subject-reduction obstruction is realized as a finite ledger NameCert packet linking the four discharge obligations to subject-reduction and generator-audit rows.

Concretized in §727: BetaSubstitutionDischargeUp isolates the beta-redex discharge row named by the subject-reduction obstruction interface.

Concretized in §729: Subject-reduction obligation ledgers realize the four MetaCIC discharge rows as a finite NameCert packet with obstruction non-escape.

Concretized in §730: Names the four MetaCIC subject-reduction discharge obligations as a finite NameCert frontier.

Concretized in §731: The four-obligation MetaCIC subject-reduction obstruction is realized as a finite NameCert boundary packet.

Concretized in §722: Closed-term context seals realize the MetaCIC subject-reduction obstruction as finite closedness, membership, reduction-handoff, obstruction, and NameCert rows.

Concretized in §6321: The residual-context cut packet realizes the subject-reduction obstruction as finite typed-context, residual, candidate-normalization, and provenance rows.

Concretized in §6474: The dependent-codomain application frontier is realized as a finite NameCert ledger for argument-type stability, discharge compatibility, closed-substitution readback, and non-escape.

Concretized in §6834: The four MetaCIC subject-reduction discharge sockets and checked obstruction are realized as a finite socket-cut NameCert packet.

Concretized in §7096: Beta-substitution preservation frontier packets realize the beta-redex discharge row of the MetaCIC subject-reduction obstruction as finite NameCert data.

Concretized in §7825: Subject-reduction residual-gap packets realize the dependent-codomain obstruction as finite obligation, residual, candidate-normalization, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §6922: The four MetaCIC subject-reduction discharge rows are realized as a finite matrix packet separating beta substitution, application-argument stability, lambda-domain stability, Pi-domain stability, obstruction, replay, provenance, and local naming.

Concretized in §8379: MetaCIC subject-reduction residual fences realize the dependent-codomain obstruction as finite discharge, residual, candidate-normalization, obstruction-retention, transport, replay, provenance, and NameCert rows.

Chapter 9378

MetaCIC Confluence partial-closure audit map

9378.1 The setup

The BEDC.MetaCIC confluence module is split across six sibling files: `AtomJoin`, `AtomShape`, `ClosedGeneral`, `ClosedUnconditional`, `Core`, and `Diamond`. The split is not cosmetic. The files separate raw reduction infrastructure, atom-shape facts, componentwise diamond lemmas, substitution-sensitive diamond assembly, closed-term bridge statements, and the unconditional fragments that are already available for specific closed sources.

Confluence is therefore not presented as a single hidden theorem. The Lean side exposes the exact state: a full parallel diamond follows from a substitution compatibility input, closed multi-step confluence follows from a closed parallel-diamond input, and a smaller set of closed source shapes is discharged without that input.

The operative relations are the multi-step closure BetaStarStep and the Tait–Martin-Löf style parallel relation BetaParallel. The ordinary proof route is also visible in the file structure: prove a diamond for parallel reduction, transport parallel steps into multi-step beta reduction, derive a strip lemma, and then derive Church–Rosser for the star relation. The current substrate has checked much of this route, but the point where parallel beta redexes require residual substitution compatibility remains an explicit input.

9378.2 What is established

The checked inventory has six parts.

- `Diamond.lean` contains the substitution-sensitive assembly. The full parallel diamond is proved from the explicit substitution compatibility proposition, and the unclosed substitution commutation obstruction is named:

  ```
  ↪ variant BEDC.MetaCIC.betaParallel diamond from substitution
  ↪ variant BEDC.MetaCIC.betaParallel app shape diamond from substitution
  ↪ variant BEDC.MetaCIC.substitute commutation unclosed obstruction
  ```

  .

- `ClosedGeneral.lean` turns a closed parallel diamond into closed strip and closed multi-step confluence. It also records the imported unclosed substitution counterexample next to the closed-substitution compatibility interface:

  ```
  ↪ variant BEDC.MetaCIC.betaParallel closed strip from diamond
  ↪ variant BEDC.MetaCIC.betaStarStep confluence general of parallel diamond closed
  ↪ variant BEDC.MetaCIC.betaParallel closed substitution compatibility obstruction
  ```

  .

- `ClosedUnconditional.lean` proves unconditional closed confluence for normal sources and the atom cases that reduce to normality. The theorem family is shape-limited, not a blanket closed confluence theorem:

  ```
  ↪ variant BEDC.MetaCIC.betaParallel diamond closed normal
  ↪ variant BEDC.MetaCIC.betaStarStep confluence closed normal
  ↪ variant BEDC.MetaCIC.betaStarStep confluence closed sort
  ↪ variant BEDC.MetaCIC.betaStarStep confluence closed var impossible
  ```

  .

- `Core.lean` supplies the reduction algebra: multi-step concatenation, parallel-to-star transport, atom diamonds, and componentwise diamonds for application, lambda, and Pi shapes:

  ```
  ↪ variant BEDC.MetaCIC.betaParallel to betaStar
  ↪ variant BEDC.MetaCIC.betaParallel app diamond of components
  ↪ variant BEDC.MetaCIC.betaParallel lam shape diamond
  ↪ variant BEDC.MetaCIC.betaParallel pi shape diamond
  ```

  .

- `AtomShape.lean` proves the case analysis needed to keep atoms and non-lambda applications from silently entering the beta-redex branch:

  ```
  ↪ variant BEDC.MetaCIC.betaStep app cases
  ↪ variant BEDC.MetaCIC.betaStarStep app non lam shape
  ↪ variant BEDC.MetaCIC.betaParallel betaStarStep atom coincide
  ↪ variant BEDC.MetaCIC.betaStar sort join
  ```

  .

- `AtomJoin.lean` exposes the small join and inversion wrappers used by the later confluence files. The atom joins are unconditional, while the application shape result separates congruence targets from beta-redex targets:

  ```
  ↪ variant BEDC.MetaCIC.betaStarStep diamond sort
  ↪ variant BEDC.MetaCIC.betaStarStep diamond var
  ↪ variant BEDC.MetaCIC.betaParallel app target shape
  ↪ variant BEDC.MetaCIC.betaParallel app no beta
  ```

  .

This is a genuine checked layer. The manuscript may cite componentwise diamonds, atom joins, parallel-to-star transport, closed normal confluence, and the conditional bridges from parallel diamond to closed star confluence.

The inventory also fixes what cannot be inferred by proximity. The presence of `BetaParallelDiamondClosed` as a proposition does not itself discharge that proposition. The theorem `betaStarStep_confluence_general_of_p` consumes such a diamond as an input; it is not the diamond. Likewise, the theorem `betaParallel_diamond_from_s` consumes BetaParallelSubstitutionCompatibility; it does not prove that compatibility proposition.

9378.3 What is conditional

The boundary is the substitution input. `Diamond.lean` defines BetaParallelSubstitutionCompatibility and proves full parallel diamond from it. `ClosedGeneral.lean` defines BetaParallelDiamondClosed and proves closed multi-step confluence from that input. These are parameterised results, not closed-source unconditional discharge.

The unconditional part is narrower. Closed normal sources are confluent; sort sources are confluent; closed variable sources are handled through an impossible closedness premise; atom and non-lambda application shape lemmas prevent false beta-redex cases. None of these statements is the full closed Church–Rosser theorem for arbitrary closed terms.

The sharp reading is therefore:

- full parallel diamond is conditional on BetaParallelSubstitutionCompatibility;
- full closed star confluence is conditional on BetaParallelDiamondClosed;
- closed normal, sort, and impossible closed-variable source cases are unconditional;
- application, lambda, Pi, and atom shape lemmas are infrastructure, not final closed confluence claims.

This conditional shape is constructive and useful. If the substitution compatibility input is later supplied inside the same first-principles substrate, the existing bridge theorems already specify how the result flows to the parallel diamond and then to closed star confluence. Until that input is supplied, the bridge remains a bridge from a named premise, not a closure certificate for the endpoint.

9378.4 Comparison with the SR obstruction

The subject-reduction map in chapter 9377 isolates a dependent-codomain obstruction: typing preservation across beta and binder congruence needs four named hypotheses. The confluence map has a different shape. It is not blocked by a displayed codomain that must be retyped after a reduction. It is blocked at the residual-substitution algebra needed to join parallel beta branches.

That difference matters. The confluence side already proves a large amount of syntax-directed reduction geometry without invoking typing. The remaining boundary is the substitution compatibility interface that makes the application beta branch commute with the componentwise join. Thus the confluence obstruction is algebraic over terms, while the subject-reduction obstruction is type-preservation across dependent codomains.

The two maps also differ in their positive fragments. Subject reduction already has a parameterised theorem whose four inputs are typing-specific. Confluence has a parameterised theorem whose central input is substitution compatibility, plus unconditional results for closed normal sources and for atom cases. A reader should therefore not merge the two boundaries into one

generic "MetaCIC gap". They name different interfaces and would be discharged by different proofs.

9378.5 Apophatic framing

This chapter applies the same apophatic discipline as chapter 9377. BEDC names the open cell instead of silently promoting a conditional theorem into an unconditional claim. The audit map is not a confluence proof. It is a paper-side handle for the exact proof state exported by the Lean substrate.

Concretized in §7405: Residual substitution comparison is realized as finite typed-window, local-join, obstruction-retention, bounded-checker, transport, replay, provenance, and NameCert rows.

Concretized in §8595: The residual sector nerve realizes the confluence audit boundary as finite residual-branch, candidate-join, obstruction-retention, transport, replay, provenance, and NameCert rows.

The positive content is still substantial: the module names the missing input, proves the bridges that would consume it, and proves the available closed normal and atom fragments. The negative content is equally binding: a full closed Church–Rosser theorem for arbitrary closed terms is not claimed here.

This is why the chapter belongs in the visions layer rather than in a proof obligations chapter. It does not introduce a new axiom, a new rule of the calculus, or a new proof principle. It gives the paper a stable handle for reading the checked confluence files without inflating a conditional theorem into a public closure claim.

9378.6 Boundary

This chapter does not solve MetaCIC confluence. It records the partial closure state of the confluence files and forbids the manuscript from citing a theorem stronger than the Lean interface supplies. The available claims are the checked component lemmas, the substitution-conditional parallel diamond, the closed-diamond-to-star-confluence bridge, and the unconditional closed normal and atom cases.

In particular, this chapter does not claim BetaParallelSubstitutionCompatibility, does not claim BetaParallelDiamondClosed, and does not claim arbitrary closed-term Church–Rosser for BetaStarStep. Those are the boundary names that must remain visible when the confluence module is cited elsewhere in the manuscript.

Concretized in §737: The checked closed-normal and atom confluence fragments are realized as a finite NameCert packet with conditional-boundary non-escape.

Concretized in §741: Closed-normal confluence is realized as a finite seal packet separating available normal-source joins from the substitution-compatibility boundary.

Concretized in §735: Finite NameCert packet for the confluence audit-map boundary.

Concretized in §743: The conditional confluence audit map is realized as a finite NameCert packet separating checked fragments from named remaining inputs.

Concretized in §744: The conditional parallel-diamond bridge, closed-star handoff, atom fragments, and substitution-compatibility boundary are packaged as a finite NameCert carrier.

Concretized in §745: Finite audit-witness packet for the conditional MetaCIC confluence boundary.

Concretized in §2181: The residual-substitution compatibility boundary is realized as a finite NameCert packet for conditional confluence handoff.

Concretized in §2212: The confluence critical-pair boundary is realized as finite MetaCIC redex-frontier and common-reduct NameCert data.

Concretized in §2325: The closed parallel diamond premise is realized as a finite ledger boundary consumed by the conditional MetaCIC confluence bridge.

Concretized in §2382: MetaCIC frontier critical-pair packets realize the confluence audit-map boundary as finite redex-frontier, local-join, bounded-checker, substitution-boundary, transport, replay, provenance, and NameCert rows.

Concretized in §3595: Local peaks and candidate joins are recorded as finite BEDC confluence witnesses.

Concretized in §3791: The substitution-sensitive conditional confluence boundary is realized as finite typed-window, redex-frontier, substitution-boundary, local-join, obstruction, TasteGate, replay, provenance, and NameCert rows.

Concretized in §3827: The residual checker realizes the substitution-sensitive confluence frontier as finite typed-window, redex-frontier, bounded-checker, local-join, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §3921: The confluence residual envelope realizes the conditional substitution boundary as finite MetaCIC NameCert data.

Concretized in §4082: Local peaks, candidate joins, residual frontiers, substitution boundaries, bounded checks, and obstruction rows are realized as a finite confluence-diamond boundary NameCert packet.

Concretized in §4061: The finite residual-budget and candidate-join rows for the conditional MetaCIC diamond route are realized as NameCert data.

Concretized in §4126: The residual-substitution local-join ledger is realized as finite peak, candidate-join, bounded-checker, obstruction, TasteGate, replay, provenance, and NameCert rows.

Concretized in §4238: The parallel-diamond frontier chapter realizes the confluence audit map's residual-premise, critical-pair, local-join, closed-checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7608: Critical-pair residual streams realize the confluence audit map as finite parallel-diamond frontier, normalization frontier, residual-window, candidate-join, stream-order, failure, transport, replay, provenance, and NameCert rows.

Concretized in §6433: Typed parallel-diamond boundary packets realize the confluence audit map's typed-window, residual-substitution, critical-pair, local-join, checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §4370: The residual-substitution compatibility boundary is realized as finite MetaCIC confluence packet data.

Concretized in §5126: The residual local-join packet realizes the confluence audit boundary before the conditional parallel-diamond bridge.

Concretized in §5174: The confluence audit route is realized as finite typed-window, residual-budget, candidate-join, sibling-handoff, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §5970: Residual-join determinacy realizes the confluence audit map's finite residual-substitution boundary as a budgeted local join packet.

Concretized in §6063: The confluence audit boundary is realized as a finite residual-window diamond envelope.

Concretized in §6547: The confluence diamond ledger realizes residual-substitution, local-join, bounded-checker, and obstruction rows as finite NameCert data.

Concretized in §6547: The finite local-diamond ledger realizes the confluence audit-map branch through residual-frontier, candidate-join, bounded-checker, transport, replay, provenance, and NameCert rows.

Concretized in §6663: The closed parallel substitution-budget packet realizes the confluence audit map's named substitution-compatibility boundary as finite NameCert data.

Concretized in §6802: The MetaCIC confluence audit map is realized as a finite inventory packet separating checked fragments from retained conditional confluence premises.

Concretized in §6833: Closed-normal and atom confluence fragments are realized as a finite NameCert packet retaining the substitution-compatibility boundary.

Concretized in §8360: Residual peak schedulers realize the confluence audit boundary as finite local-peak, residual-row, candidate-join, blocked-edge, transport, replay, provenance, and NameCert data.

Concretized in §7114: Closed-normal and atom confluence joins are realized as a finite proof-relevant NameCert packet separated from the full substitution-compatibility boundary.

Concretized in §8262: The confluence-fork certificate realizes the MetaCIC parallel-diamond audit fork as finite source, branch, candidate-join, residual-boundary, and NameCert rows.

Concretized in §8021: Finite residual-join windows realize the conditional MetaCIC confluence boundary as retained premise, local join, obstruction, bounded checker, transport, provenance, and NameCert rows.

Concretized in §7977: The residual-substitution commutation boundary is realized as finite residual-window, substitution-site, candidate-join, conditional-handoff, transport, replay, provenance, and NameCert rows.

Concretized in §7879: Residual-pair lattices realize the confluence audit map as finite local-pair, candidate-join, bounded-checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §8317: Critical-pair residual schedulers realize the bounded residual and local diamond route as finite NameCert rows.

Concretized in §7042: A finite residual-pair local-diamond witness packet realizes the confluence audit frontier through displayed candidate-join and substitution rows.

Concretized in §7220: Critical overlap envelopes realize bounded MetaCIC confluence overlap rows as finite NameCert data.

Concretized in §7258: finite residual-join-grid NameCert carrier for the confluence audit route

Concretized in §7291: Residual critical-pair packets realize the confluence audit map as finite residual-substitution, local-join, closed-checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §8134: finite residual-join matrix for MetaCIC confluence audit boundaries

Concretized in §7709: Residual critical-pair budgets realize the confluence audit map as finite residual-frontier budget, candidate-join, closed-check, obstruction, TasteGate, transport, replay, provenance, and NameCert rows.

Concretized in §8368: Candidate residual-join budgets realize the confluence audit map as finite typed-window, residual-substitution, budget, candidate-join, guard, obstruction, TasteGate-instance, transport, replay, provenance, and NameCert rows.

Concretized in §7346: The local confluence frontier is realized as finite critical-pair, conditional-join, substitution-premise, obstruction, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7422: Residual candidate selectors realize the bounded residual-confluence frontier as finite candidate, checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7632: Parallel diamond scheduling is realized as finite component-diamond, residual-boundary, atom-join, closed-source guard, and star-confluence handoff rows.

Concretized in §5547: The candidate-normalization handoff realizes the confluence audit map's guarded route from candidate evidence through normalization frontier, residual envelope, decidable-boundary, blocked-edge retention, transport, replay, provenance, and NameCert rows.

Concretized in §7744: MetaCIC extraction-normalization sockets realize the confluence audit boundary as finite extraction, candidate-normalization, residual-substitution, local-join, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7752: Finite residual confluence windows realize bounded local join evidence as NameCert rows.

Concretized in §7966: Residual confluence fuel packets realize bounded critical-pair and residual-substitution inspection for the MetaCIC confluence boundary.

Concretized in §8111: The residual-diamond frontier is realized as finite critical-pair, residual-substitution, candidate-join, bounded-checker, obstruction, transport, replay, provenance, and NameCert data.

Concretized in §8126: Critical-pair residual matrices realize the MetaCIC confluence audit map as finite typed-redex, residual-row, join-candidate, closedness, substitution, provenance, and NameCert data.

Concretized in §8254: The residual frontier packet realizes the confluence audit map as finite retained-premise, candidate-join, obstruction, and closed-fragment NameCert data.

Concretized in §8275: Proof-trace confluence is realized as finite MetaCIC reduction, residual-substitution, closed-normal, and conditional-diamond NameCert data.

Concretized in §8316: Residual critical-pair sieve packets realize the MetaCIC confluence audit map as finite candidate, residual-window, blocked-edge, decidable-boundary, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8560: The bounded Church-Rosser fragment realizes the confluence audit boundary as finite atom, normal-source, conditional-diamond, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §8739: Residual local-confluence matrices realize the bounded local-join frontier as finite typed-window, residual-pair, checker, obstruction, transport, replay, provenance, and NameCert rows.

Chapter 9379

MetaCIC Decidability partial-closure audit map

9379.1 The setup

The BEDC.MetaCIC decidability submodule records the mechanical checking layer for the internal CIC fragment. It is split across five sibling files: `CheckClosed`, `CheckClosedAtom`, `CheckCompleteness`, `GalleryTests`, and `NormalEqDecide`. The split separates syntax equality, type inference, completeness wrappers, gallery checks, and a bounded normal-equality checker.

The central checker is `inferTypeCtx`. It maps a context and a term to an optional inferred type. Its closed-term wrapper is `inferTypeCtxClosed`. The equality atom is `Term.eq`, a Boolean structural equality on represented terms, with `checkSameTerm` as the option-valued interface used by the typing checker. The normal-equality side uses `normalizeBounded` and `decideClosedBetaConv`; this is a fuel-bounded normal-form comparison, not an unrestricted beta-conversion oracle.

The audit question is therefore precise. The submodule does not assert a single informal statement saying that MetaCIC is decidable. It provides checked Boolean and option procedures, then proves the soundness and completeness properties that are available for those procedures. This chapter maps exactly which part is closed and which part remains conditional on normalisation or on a particular term shape.

9379.2 What is established

The checked inventory has five parts.

- `CheckClosedAtom.lean` gives structural equality for terms. The Boolean equality is reflexive, converts true results into propositional equality, converts propositional equality into true results, and exposes the full iff statement:

  ```
  ↪ variant BEDC.MetaCIC.Term.eq refl
  ↪ variant BEDC.MetaCIC.Term.eq true to eq
  ↪ variant BEDC.MetaCIC.Term.eq eq to true
  ↪ variant BEDC.MetaCIC.Term.eq correct
  ```

  .

- The option-valued same-term checker is also bidirectional. A successful check gives equality, and equality gives a successful check:

  ↪ variant BEDC.MetaCIC.checkSameTerm sound

  ↪ variant BEDC.MetaCIC.checkSameTerm complete

  .

- The first type-inference checker `inferType` is sound and complete for the inductive `HasType` relation, both in arbitrary contexts and in the empty context wrapper:

  ↪ variant BEDC.MetaCIC.inferType sound

  ↪ variant BEDC.MetaCIC.inferType complete

  ↪ variant BEDC.MetaCIC.inferTypeClosed sound

  ↪ variant BEDC.MetaCIC.inferTypeClosed complete

  .

- `CheckClosed.lean` repeats the checker through `inferTypeCtx`, with explicit recognisers for sort and Pi types. The context-indexed and closed wrappers are again sound and complete through `inferTypeCtx_sound`, `inferTypeCtx_complete`, `inferTypeCtxClosed_sound`, and `inferTypeCtxClosed_com`

- `CheckCompleteness.lean` records a shape-organised completeness surface. Its raw, total-exists, and strict exported wrappers are checked, and the direct Pi, lambda, and application cases expose the main compound shapes:

  ↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete pi direct

  ↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete lam direct

  ↪ variant BEDC.MetaCIC.CheckCompleteness.inferTypeCtx complete app direct

  .

This is a strong positive layer. For represented syntax equality, the Boolean procedure is correct. For typing, the option-valued inference procedure is not merely a recogniser for gallery examples; it is proved sound and complete against the inductive `HasType` relation exported by the substrate.

The gallery file supplies executable sanity rows for the closed example surface. Boolean, natural-number, pair, option, list, sum, and polymorphic witnesses all pass the `inferTypeCtx []` non-`none` test:

↪ variant BEDC.MetaCIC.gallery sanity bool witnesses
↪ variant BEDC.MetaCIC.gallery sanity nat witnesses
↪ variant BEDC.MetaCIC.gallery sanity pair witnesses
↪ variant BEDC.MetaCIC.gallery sanity option witnesses
↪ variant BEDC.MetaCIC.gallery sanity list witnesses
↪ variant BEDC.MetaCIC.gallery sanity sum witnesses
↪ variant BEDC.MetaCIC.gallery sanity polymorphic witnesses
↪ variant BEDC.MetaCIC.gallery sanity all typed witnesses

.

The gallery rows are not the source of the general typing theorem. Their role is audit-facing: they show that the same checker surface used in the soundness and completeness theorems also accepts the main Church-encoded example families.

9379.3 Normal equality and bounded beta conversion

`NormalEqDecide.lean` adds a separate equality surface. It defines a bounded normalizer and compares the bounded normal forms with `Term.eq`. The basic structural facts are checked:

```
↪ variant BEDC.MetaCIC.normalizeBounded zero
↪ variant BEDC.MetaCIC.normalizeBounded sort
↪ variant BEDC.MetaCIC.normalizeBounded var
↪ variant BEDC.MetaCIC.normalizeBounded lam succ
↪ variant BEDC.MetaCIC.normalizeBounded pi succ
↪ variant BEDC.MetaCIC.normalizeBounded app succ
```

.

The decision procedure is reflexive at every fuel value, and a true Boolean result implies equality of the two bounded normalizer outputs:

```
↪ variant BEDC.MetaCIC.decideClosedBetaConv refl
↪ variant BEDC.MetaCIC.decideClosedBetaConv true eq normalized
```

.

It also yields beta-convertibility between the bounded normalizer outputs:

```
↪ variant BEDC.MetaCIC.decideClosedBetaConv true betaConv normalized
```

.

The stronger finished-normalisation interface is explicitly conditional. If both terms have closed finished normalisation witnesses at the same fuel, then a true result from `decideClosedBetaConv` gives equality of the named normal forms, beta-convertibility of those normal forms, and preservation of the closed normal-form witnesses:

```
↪ variant BEDC.MetaCIC.decideClosedBetaConv sound finished
↪ variant BEDC.MetaCIC.decideClosedBetaConv sound finished betaConv
↪ variant BEDC.MetaCIC.decideClosedBetaConv sound finished eq
```

.

This file is therefore a bounded normal-equality checker plus a conditional soundness bridge for the case where the fuel is known to finish on closed normal forms. It is not a theorem that arbitrary closed terms have a computed finished normal form at some fuel, and it is not an unrestricted decision procedure for beta conversion over all closed terms.

9379.4 What remains conditional

Three boundaries must remain visible.

- Typing is closed at the checker interface. For the represented `HasType` relation, `inferTypeCtx` has soundness and completeness theorems. The paper may cite this as a checked typing decision surface for the encoded relation.
- Closedness itself is represented by the inductive predicate `ClosedAt`. The decidability submodule consumes closedness in places such as `inferTypeCtxClosed_sound` and `ClosedFinishedNormalizes`; it does not provide a separate theorem named `Decidable (ClosedAt n t)`.
- Beta-conversion equality is conditional on bounded normalisation data. The Boolean checker is proved sound for equality of bounded normalizer outputs and for finished closed normal forms. It is not a global closed-term beta-conversion decider unless the missing normalisation endpoint data is supplied.

The first boundary is positive closure. The second is a naming boundary: the closedness predicate is an inductive relation with many checked preservation facts elsewhere in MetaCIC, but this submodule does not export a dedicated closedness-decision theorem. The third boundary links directly to normalisation: full conversion decidability would require a bridge from typed closed terms to finished bounded normal forms at a specified fuel or to an equivalent fuel-erased normalisation theorem.

9379.5 Comparison with other audit maps

The subject-reduction audit map in chapter 9377 is blocked by dependent-codomain preservation. It names four typing-preservation obligations and records an obstruction to an unconditional closed subject-reduction theorem. The decidability map has a different shape. Typing inference itself is not the obstruction; it is already checked as sound and complete for the encoded `HasType` relation.

The confluence audit map in chapter 9378 is blocked at residual substitution compatibility for joining parallel beta branches. The decidability map uses substitution through typing and normal comparison, but its open edge is not the parallel diamond. Its open edge is the endpoint data needed to promote bounded normal equality into global closed-term conversion decidability.

The normalisation layer is the closest neighbour. The bounded normalizer in `NormalEqDecide` supplies a computable comparison surface. Strong normalisation supplies the kind of endpoint guarantee that would make such a surface public for all typed closed terms. Without that connection, the normal-equality checker remains a bounded and conditionally sound procedure.

Concretized in §749: MetaCIC decidability boundaries are realized as finite checker, bounded-normal comparison, refusal, provenance, and local NameCert rows.

Concretized in §751: The concrete witness packages typing-checker exactness and bounded-conversion non-escape as finite NameCert rows.

Concretized in §752: Bounded normal-equality checkers are realized as finite BHist packets over bounded normalizer output rows, structural Term.eq readback, finished-normalization witnesses, closedness rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Concretized in §2236: Finite observation folds realize the bounded MetaCIC checker-output packet as finite NameCert data.

Concretized in §3616: The decidable typing frontier is realized as finite checker, structural equality, bounded-normal comparison, refusal, provenance, and local NameCert rows.

Concretized in §3714: Typed normal-form deciders realize the MetaCIC decidability audit map as finite typing-checker, bounded-normal-comparison, finished-normalization, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §3747: The recognizer-to-carrier boundary is realized as a seven-row finite NameCert obstruction packet.

Concretized in §3889: MetaCIC fuel traces realize bounded normalizer fuel, finished-normalization evidence, equality-check rows, refusal rows, transport, replay, provenance, and NameCert data.

Concretized in §3909: The neutral-spine packet realizes MetaCIC decidability as finite typing-checker, same-term, bounded-normal comparison, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §4097: The decidable conversion frontier realizes the bounded normal-conversion boundary as finite checker, normalization-frontier, bridge, seal, obstruction, provenance, and local NameCert rows.

Concretized in §5780: The normal endpoint comparator realizes the bounded-conversion endpoint bridge as finite typed-term, endpoint trace, bridge, frontier, seal, critical-path, provenance, and NameCert rows.

Concretized in §4098: The closedness decision boundary realizes the decidability audit warning as finite audit, bounded-checker, fuel-trace, closed-type-checker, refusal, provenance, and NameCert rows.

Concretized in §5962: Bounded normalisation fuel realizes the MetaCIC decidability endpoint boundary as finite checker, fuel, finished-normal, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §6179: Bounded normalization frontiers realize MetaCIC decidability as finite fuel, finished-normal, bounded-comparison, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §6917: The bounded decidable fragment packages closed checker, bounded normal comparison, endpoint refusal, transport, replay, provenance, and NameCert rows.

Concretized in §6947: Bounded normal-equality frontier packets realize the decidability audit boundary as finite normalizer fuel, closedness, finished-normal-form, equality-checker, transport, replay, provenance, and NameCert rows.

Concretized in §8057: The bounded normal-equality budget realizes normalizeBounded and decideClosedBetaConv as finite fuel, endpoint, comparison, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §8400: Bounded conversion frontiers realize the decidability audit map as finite checker, normalization-endpoint, comparison, refusal, transport, replay, provenance, and NameCert rows.

Thus the three maps should not be read as one generic MetaCIC deficit. Subject reduction needs preservation across dependent typing, confluence needs substitution compatibility for joins, and decidability needs the exact bridge from bounded computation to finished normal forms when beta conversion is the target. The typing-checker part of decidability is already closed; the conversion-checker part is intentionally narrower.

9379.6 Apophatic framing

The apophatic discipline is the same as in the subject-reduction and confluence chapters. BEDC names what the checker proves and refuses to let a Boolean procedure become a stronger theorem by prose. A successful `inferTypeCtx` result carries a checked `HasType` witness by soundness. A `HasType` witness is accepted by the checker by completeness. A successful bounded beta-conversion result carries only the bounded-normal equality and the finished-normalisation consequences stated by the corresponding theorems.

This is why the word "decidable" has to be read at the level of the exported theorem. For syntax equality and typing inference, the checked theorems give a closed decision surface. For closedness and global beta conversion, this chapter records the absence of a named theorem at that strength in the decidability submodule. The absence is not repaired by gallery success or by the existence of a bounded normalizer.

9379.7 Boundary

This chapter records the partial closure state of the MetaCIC decidability files. It may be cited for structural term equality, same-term checking, typing-inference soundness and completeness, the shape-organised completeness wrappers, gallery acceptance rows, and the bounded normal-equality soundness facts listed above.

It does not claim a theorem named `Decidable (ClosedAt n t)`. It does not claim an unrestricted closed-term beta-conversion decider. It does not claim that `decideClosedBetaConv` is complete for beta conversion at arbitrary fuel. Those stronger readings require explicit Lean targets before they can enter the paper as closure claims.

Concretized in §746: MetaCIC decidability is realized as a finite audit packet separating checked typing inference from bounded conversion soundness.

Concretized in §748: Bounded beta comparison is realized as a finite checker packet with an explicit finished-normalisation boundary.

Concretized in §747: The checker seal packages typing, same-term, and bounded normal-equality rows as a finite NameCert surface.

Concretized in §5949: Bounded typechecking budgets realize the MetaCIC decidability audit map as finite checker rows with refusal boundaries.

Concretized in §6412: Candidate decidability traces realize finite MetaCIC local-checker schedules, rejection ledgers, and SN/confluence handoff rows.

Concretized in §8083: MetaCIC decidable typechecking handoffs realize the inferTypeCtx soundness-completeness row and the bounded normal-equality boundary as one finite NameCert packet.

Concretized in §8104: Type-check trace deciders realize the MetaCIC decidability audit map as finite checker, same-term, bounded-normal comparison, refusal, transport, replay, provenance, and NameCert rows.

Chapter 9380

MetaCIC Substitution module audit map

9380.1 The setup

The BEDC.MetaCIC substitution submodule is the syntax layer that supports movement across binders. It is split across five sibling files: `Basic`, `ClosedComposition`, `Core`, `NatShift`, and `Statements`. The split exposes a useful audit shape. Arithmetic facts about de Bruijn indices are kept separate from raw substitution facts, closedness facts, and higher substitution statements.

The operations are `shift` and `substitute`. The first raises free indices at or above a cutoff. The second replaces the variable at a depth and adjusts indices around the removed binder. Both are ordinary recursive operations on the closed `Term` generator. The audit question is therefore not whether the operations exist, but which classifier boundary they preserve and under which hypotheses.

The current closure state is exact. Closedness at a boundary makes both operations inert at that boundary. Closed source and closed payload hypotheses also support a small family of closed composition laws. The module also names larger target shapes for shift–substitute and substitute–substitute interaction, while the closed composition file provides the checked fragment that is available without a general unclosed commutation theorem.

9380.2 Core operations

9380.2.1 `shift_closed`

The boundary theorem for shift is

↪ variant `BEDC.MetaCIC.shift closed`

. It says that if a term is closed at cutoff n, then shifting at n by one leaves the term unchanged:

$$\mathsf{ClosedAt}(n, t) \Rightarrow \mathsf{shift}(n, 1, t) = t.$$

This is a classifier-preservation fact in its strongest local form. The term has no free index reaching the cutoff, so the shift branch has no occurrence to act on. The binder cases are not hidden semantic assumptions; they are the recursive cases of the `ClosedAt` witness, where the boundary advances under lambda and Pi.

The theorem is intentionally narrow. It does not state that arbitrary shifting preserves every typing or reduction property. It states only that a term already closed below the cutoff is syntactically fixed by the shift attempted at that cutoff.

9380.2.2 `substitute_closed`

The matching boundary theorem for substitution is

↪variant `BEDC.MetaCIC.substitute closed`

. It says that if a term is closed at depth d, then substituting any payload at d leaves the term unchanged:

$$\mathsf{ClosedAt}(d, t) \Rightarrow \mathsf{substitute}(d, v, t) = t.$$

The payload v is arbitrary because the replacement site is absent. For every variable index visible in t, the closedness witness supplies the inequality placing that index below d. Hence the equality branch does not fire, and the decrement branch does not fire. The recursive cases carry the same argument through application, lambda, Pi, and sort.

This is the main closed-substitution closure law used by the paper side. It is not a beta-substitution typing theorem. It is a syntactic boundary theorem: substitution at a boundary cannot alter a term whose free-variable budget is already below that boundary.

9380.2.3 Closed composition

The closed composition file supplies the checked fragment of substitution composition that is available under closed payload and closed source hypotheses. The central theorem is

↪variant `BEDC.MetaCIC.substitute compose closed both`

. It specialises the depth-indexed theorem

↪variant `BEDC.MetaCIC.substitute compose closed both at depth`

to depth 0:

$$\begin{aligned}&\mathsf{ClosedAt}(0, v) \wedge \mathsf{ClosedAt}(0, u) \wedge \mathsf{ClosedAt}(1, t)\\ &\Rightarrow \mathsf{substitute}(0, v, \mathsf{substitute}(0, u, t)) = \mathsf{substitute}(0, \mathsf{substitute}(0, v, u), t).\end{aligned}$$

The same file also records a closed swap law

↪variant `BEDC.MetaCIC.substitute swap closed`

and a shift-after- substitution bridge

↪variant `BEDC.MetaCIC.substitute after shift one closed`

. These are closed-fragment algebra, not unrestricted substitution calculus.

The audit distinction matters. `Statements.lean` names broad target shapes, including

Lean4 (definition) `BEDC.MetaCIC.ShiftSubstituteStatement`

and

Lean4 (definition) `BEDC.MetaCIC.SubstituteSubstituteStatement`

. The checked composition theorems above are the current closed slice of that algebra. The manuscript may cite the closed slice, but it must not cite an unrestricted substitute–substitute commutation theorem unless such a theorem exists under the same Lean namespace.

Concretized in §6103: The closed substitution, shift, and composition audit surface is realized as finite NameCert spine data.

9380.3 Connection to subject reduction

Substitution is one of the necessary ingredients for MetaCIC subject reduction. The subject-reduction obstruction map in chapter 9377 isolates the beta-redex obligation as BetaSubstitutionPreservation: typing must survive when an argument is substituted into a binder body and the codomain is read back through the same substitution.

The substitution module audited here is lower-level. It proves syntactic closedness and closed composition facts about `Term`. Those facts are necessary for a disciplined beta case, but they are not the whole beta typing obligation. The typing obligation also mentions context lookup, body typing, argument typing, codomain substitution, and the dependent type assigned after the beta step.

Thus the connection is a handoff, not an identification. The substitution module supplies the syntax-side closure laws that any subject-reduction discharge must respect. The subject-reduction boundary remains the four-input interface described in chapter 9377. Closing syntactic substitution does not by itself close dependent typing preservation.

9380.4 Connection to GeneratorClosureClassifier

The same closure state has a certificate-theoretic reading. In Definition 9389.7, a generator-closure classifier records that a generator action preserves a classifier. The concrete MetaCIC instances are

```
↪variant BEDC.MetaCIC.closed under shift classifier
```

and

```
↪variant BEDC.MetaCIC.closed under substitute classifier
```

.

These two instances read the `shift` and `substitute` operations as authorized generators over the closed-term carrier. Their classifier is `ClosedAt` at the relevant boundary. The closure field says that when the operation is applied inside the admitted slice, the result stays inside that slice.

The algebra of Definition 9389.8 then explains why these local facts compose with the wider BEDC audit machinery. Identity, sequential composition, alternation, sub-generator restriction, and finite iteration preserve classifier closure once the corresponding local classifier witnesses are present. The substitution module provides concrete local witnesses for two important MetaCIC generators; the algebra explains how such witnesses can be routed through larger certificate networks without changing their scope.

9380.5 Apophatic framing

This chapter names a positive closure layer without turning it into a larger claim. BEDC can say exactly what the substitution module supplies: boundary inertness for shift and substitution, closed-source composition fragments, and classifier instances for the two generator actions. It also says what the module does not supply: unrestricted substitution commutation, closed subject reduction, or a hidden semantic equality principle for terms.

The apophatic point is methodological. The manuscript must not fill a missing dependent-typing bridge by appealing to the existence of syntactic substitution lemmas. The available lemmas are real, checked, and useful, but their scope is the closed syntactic layer.

9380.6 Boundary

This chapter records the present closure state of `BEDC.MetaCIC.Substitution`. It may be used to cite the checked boundary theorems `shift_closed` and `substitute_closed`, the closed composition theorems in `ClosedComposition.lean`, the named statement shapes in `Statements.lean`, and the concrete GeneratorClosureClassifier instances for shift and substitution.

It does not claim an unrestricted substitution calculus. It does not claim that BetaSubstitutionPreservation is discharged. It does not claim the full closed subject-reduction theorem excluded by chapter 9377. The substitution closure state is a necessary building block for those routes, not their endpoint.

Concretized in §??: Substitution audit rows are realized as a finite NameCert packet for closed syntax movement and subject-reduction handoff boundaries.

Concretized in §779: The closed substitution audit map is realized as a finite NameCert packet over ClosedAt, shift, substitute, closed-composition, and generator-closure rows.

Concretized in §824: Substitution audit window routes realize the closed shift, substitute, composition, generator-closure, and subject-reduction handoff rows as a finite NameCert packet.

Concretized in §840: Context-indexed substitution audit rows are realized as a finite NameCert packet over closed-boundary, binder-seal, beta-preservation, and typed-normal-form handoff surfaces.

Concretized in §910: Substitution generator-audit packets realize closed shift, closed substitution, closed composition, and generator-route replay as a finite NameCert carrier.

Concretized in §1102: The substitution audit map is realized by a finite NameCert carrier for closed shift and substitution audit routes.

Concretized in §6990: Closed shift, substitution, composition, and binder-seal audit rows are realized as finite NameCert spine data.

Concretized in §??: The MetaCIC substitution audit packet realizes the closed movement, generator-closure, and subject-reduction handoff boundary as finite NameCert rows.

Concretized in §7519: Typed substitution windows realize the MetaCIC substitution audit boundary as finite typed-context, closedness, substitution, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7856: The residual-substitution spine realizes the closed substitution audit boundary as finite residual-window, binder-seal, obstruction, transport, replay, provenance, and NameCert rows.

Chapter 9381

MetaCIC ClosurePreservation module audit map

9381.1 The setup

`ClosurePreservation.lean` is a single-file module inside `BEDC.MetaCIC`. It records the closedness-preservation layer for the core syntax operations that later appear in beta reduction and its finite closures. The file is deliberately narrower than subject reduction: its classifier is `ClosedAt`, not `HasType`.

The public surface is a five-theorem chain. It starts with shift at one successor depth, passes through variable substitution and full term substitution at an arbitrary closedness boundary, and then lifts the result to one-step beta reduction and finite beta-step chains. Each theorem is an ordinary structural fact about MetaCIC terms and the closedness predicate.

This gives a useful contrast with the subject-reduction obstruction in chapter 9377. Closure preservation is available without extra typing hypotheses because no dependent codomain is being transported. Subject reduction is blocked at the typing layer, where application arguments, binder domains, and beta substitution must preserve displayed types.

9381.2 Established theorems

9381.2.1 `shift_one_preserves_closed_succ`

The shift theorem is

```
↪variant  BEDC.MetaCIC.shift one preserves closed succ
```

. It says that shifting by one at depth n preserves closedness at the successor boundary. In schematic form:

$$\mathsf{ClosedAt}(n, t) \Rightarrow \mathsf{ClosedAt}(n + 1, \mathsf{shift}(n, 1, t)).$$

The theorem is the local shift witness needed by substitution and beta closure. Under binders, the closedness boundary advances in the same way as the term recursion. The proof obligation is therefore structural: every free index that was already below the source boundary remains below the target boundary after the one-step lift.

9381.2.2 substitute_var_preserves_closed_at

The variable-level substitution theorem is

↪variant BEDC.MetaCIC.substitute var preserves closed at

. It is the pointwise closedness fact for the variable branch of substitution. The theorem separates the arithmetic cases that full substitution later uses: the replaced variable, indices below the replacement depth, and indices above the replacement depth.

This theorem is not a typing statement. It does not assign a type to a replacement payload and does not read back a substituted codomain. It only says that the variable operation used by substitution respects the relevant closedness boundary when its closedness inputs are present.

9381.2.3 substitute_preserves_closed_at

The full substitution theorem is

↪variant BEDC.MetaCIC.substitute preserves closed at

. It lifts the variable theorem through the full `Term` recursion. In schematic form:

$$\mathsf{ClosedAt}(n+1,t) \wedge \mathsf{ClosedAt}(n,v) \Rightarrow \mathsf{ClosedAt}(n,\mathsf{substitute}(n,v,t)).$$

The displayed form records the usual binder budget: the substituted term is closed one level deeper, and the payload is closed at the receiving boundary. After substitution removes the selected variable, the result is closed at the receiving boundary.

This is the main closure-preservation lemma for syntactic substitution. It is strong enough to support beta-redex closure, because beta contraction is implemented by `substitute`. It is not strong enough to give beta subject reduction, because subject reduction also requires the type assigned to the contractum to be the substituted codomain.

9381.2.4 betaStep_preserves_closed

The one-step beta theorem is

↪variant BEDC.MetaCIC.betaStep preserves closed

. It states that a single beta step preserves `ClosedAt` at any depth:

$$\mathsf{BetaStep}(t,t') \wedge \mathsf{ClosedAt}(n,t) \Rightarrow \mathsf{ClosedAt}(n,t').$$

The beta-redex constructor uses `substitute_preserves_closed_at`. The congruence constructors use the corresponding recursive closedness components for applications, lambdas, and Pi types. No typing context is inspected, and no conversion rule is introduced.

This theorem is a genuine building block for any typed route through beta reduction. It guarantees that beta movement does not leave the closed-term slice. Its scope stops exactly there: preserving the closedness classifier is not preserving the `HasType` relation.

9381.2.5 betaStarStep_preserves_closed

The finite beta-chain theorem is

↪variant BEDC.MetaCIC.betaStarStep preserves closed

. It lifts one-step closedness preservation from `BetaStep` to `BetaStarStep`. In schematic form:

$$\mathsf{BetaStarStep}(t,t') \wedge \mathsf{ClosedAt}(n,t) \Rightarrow \mathsf{ClosedAt}(n,t').$$

The theorem is the finite-iteration closure fact for beta reduction. A zero-step chain is immediate, and a nonzero chain composes the one-step theorem with the induction hypothesis for the remaining chain.

This is the exact closure fact consumed by closed-normal and conversion-facing arguments: a closed term remains closed along any finite beta path. It does not say that the term remains typed, normalizing, confluent, or convertible under an unrestricted typing rule.

9381.3 Why ClosurePreservation is unconditional while SR is not

The distinction is the classifier. ClosurePreservation works over `ClosedAt`, a structural predicate on the syntax tree. The proof cases follow the constructors of terms, beta steps, and finite beta chains. When the operation descends under a binder, the closedness index advances. When beta contracts a redex, the substitution theorem supplies the closedness budget for the body and argument.

Subject reduction works over `HasType`. Its input is not just a term but a typing derivation in a context with a displayed type. The dependent-codomain cases record information that closedness cannot see. Reducing an argument can change the codomain read through that argument. Reducing a displayed binder domain can change the type annotation that a typing derivation depends on. Contracting a beta redex requires the body type and codomain to be transported through substitution.

Therefore the closure layer is unconditional, while the subject-reduction layer remains conditional at the interface named in chapter 9377. The two statements do not compete. They live over different classifiers.

9381.4 Role within MetaCIC

ClosurePreservation supplies the closed-term invariant used around subject reduction, confluence, normalization, beta conversion, and closed consistency. It says that the syntax operations used by beta reduction do not escape the closed slice once their inputs are closed at the required boundary.

For subject-reduction routes, this removes one possible source of failure. A typed proof term that is already closed does not become open merely because a beta step or a finite beta path is taken. The remaining obstruction is not free-variable escape. It is the dependent typing transport described by the subject-reduction map.

For confluence and normalization routes, the same invariant lets the paper keep closed-term hypotheses stable while it discusses common reducts, normal forms, and beta-strong fragments. A finite beta path may be followed without rechecking the closedness boundary from scratch at each endpoint.

9381.5 Connection to GeneratorClosureClassifier

The substitution and shift theorems instantiate the generator-closure reading of Definition 9389.7. A generator is authorized only relative to a classifier, and its closure field records that admitted inputs remain inside the classifier after the generator acts.

For MetaCIC closedness, the relevant classifier is `ClosedAt`. The concrete generator witnesses are `closed_under_shift_classifier` and `closed_under_substitute_classifier`. The shift

and substitution preservation theorems are exactly the local facts that make those witnesses meaningful.

Beta preservation then reads as a composed generator story. The beta redex case uses substitution, the congruence cases recurse through term constructors, and the beta-star theorem applies finite iteration. The classifier stays closed under those authorized moves, but the classifier is still closedness rather than typing.

9381.6 Boundary

This chapter records the closure state of `BEDC.MetaCIC.ClosurePreservation`. It may be used to cite the five checked preservation theorems named above and the concrete GeneratorClosureClassifier witnesses for shift and substitution.

It does not claim that closure preservation implies typing preservation. It does not claim that `betaStarStep_preserves_closed` implies typed subject reduction for closed terms. It does not supply a conversion rule for typing or a discharge of the dependent-codomain, binder-domain, or beta substitution obligations isolated in chapter 9377.

Concretized in §786: Closure-preservation audit witnesses are realized as finite BEDC NameCert packets for the MetaCIC ClosedAt theorem chain.

Concretized in §791: Closure preservation is realized as a finite audit packet for closedness-preserving shift, substitution, beta-step, and beta-star routes.

Concretized in §792: The concrete audit trace records the closedness-preservation theorem chain as a finite NameCert packet.

Concretized in §794: ClosurePreservation traces realize the unconditional ClosedAt-preservation chain as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §795: MetaCIC closure-preservation audit traces are realized as finite BHist, generator-closure, kernel-acceptance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §788: Finite NameCert packet for the ClosurePreservation theorem chain.

Concretized in §797: The closed beta path packet realizes the closure-preservation audit map as a finite NameCert route object.

Concretized in §800: Closure-preservation theorem rows are realized as a finite NameCert audit seal for shift, substitution, beta-step, and beta-star closedness routes.

Concretized in §801: Closure-preservation classifier packets realize the MetaCIC ClosedAt generator-closure audit as a concrete NameCert surface.

Concretized in §802: finite ClosurePreservation audit packet

Concretized in §803: ClosurePreservation is realized as a finite audit packet over shift, substitution, beta-step, beta-star, closedness, transport, route, provenance, and NameCert rows.

Concretized in §804: The dependent-codomain closure-preservation packet records the finite transport boundary consumed by subject-reduction routes.

Concretized in §805: Closure-preservation route ledgers realize the MetaCIC closedness-preservation chain as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6692: Closed beta trace envelopes realize the ClosurePreservation audit map as finite beta-chain, substitution, and ClosedAt replay data.

Concretized in §6876: Finite beta-chain closedness iteration realizes the BetaStarStep closedness route as a finite NameCert packet without subject-reduction or confluence claims.

Chapter 9382

MetaCIC TypedExamples module audit map

9382.1 The setup

The BEDC.MetaCIC `TypedExamples` submodule is a worked-example gallery for the internal CIC fragment. Its files collect Church encodings, dependent identity terms, polymorphic identity terms, application forms, beta redexes, projection forms, Pi forms, constants, and sort-indexed context examples.

This chapter reads the gallery as an audit surface. Unlike the subject reduction, confluence, normalisation, substitution, and decidability audit maps, the submodule is not primarily a metatheory module. It does not provide the general proof that typing is preserved by every beta step or that every closed typed term normalises. It supplies concrete typed terms whose elaboration and typing witnesses exercise the substrate on representative programs.

The closure state is therefore gallery-level. Each cited target says that a particular encoded term has the displayed type, closedness witness, beta computation, or checker acceptance result. The collection is useful precisely because the terms are ordinary objects of the represented syntax rather than external examples.

9382.2 Inventory

The module has four broad strata.

- `ChurchGallery.lean` and the `ChurchGallery/` directory provide the base Church encodings for booleans, natural numbers, pairs, and options. They expose the shared terms `churchBoolTy`, `churchTrueTm`, `churchFalseTm`, `churchNatTy`, `churchZeroTm`, `churchOneTm`, and `churchSuccTm`, plus typed constructors and eliminators for pair and option witnesses.
- The sibling files `ChurchBoolLogic`, `ChurchNatRec`, `ChurchNatArith`, `ChurchNatPred`, `ChurchEquality`, `ChurchList`, and `ChurchSum` extend the gallery with boolean connectives, arithmetic operations, predecessor-style examples, Leibniz equality encodings, Church lists, and Church sums.
- `Identity.lean` and `Polymorphic.lean` contain identity, dependent identity, polymorphic identity, constant, apply, flip, and compose forms. These examples check binder discipline and universe-like sort movement inside the represented calculus.

- `AppForms.lean`, `Applications.lean`, `ApplicationTail.lean`, `BasicApplications.lean`, `BetaRedex.lean`, `ConstForms.lean`, `PiForms.lean`, `Projections.lean`, and `Sorts.lean` cover application, beta-redex, constant, Pi, projection, and context lookup shapes.

This inventory is intentionally heterogeneous. Some files demonstrate direct `HasType` derivations. Some use `inferTypeCtx_sound` to turn a checker result into a typing witness. Some record beta computation rows or closedness rows that make later examples admissible. Together they form a ground-truth set for the represented syntax.

The file-level audit reading is similarly explicit.

- The gallery root imports the base boolean, natural-number, pair, and option files and gives shared names to the recurring Church terms. Those shared names are the vocabulary consumed by the later arithmetic, equality, and list files.
- `BoolNat.lean` is the smallest Church-encoding source. It proves the boolean and natural-number type formers, their constructors, and small identity and constant examples.
- `Pair.lean` and `Option.lean` add product-like and maybe-like data. Their role is not merely to add more examples, but to force the checker through multi-argument constructors and eliminator-style terms.
- `ChurchBoolLogic.lean` and the arithmetic files move from typing rows to computation rows. They record how boolean connectives and natural number operations reduce on selected inputs under the closedness hypotheses displayed in the theorem statements.
- `ChurchEquality.lean` is the dependent proof-object branch. It checks Leibniz equality types, reflexivity proofs, closedness rows, and normal-equality checker rows for selected equality examples.
- `ChurchList.lean` and `ChurchSum.lean` are the larger binder-stress files. They combine nested type formers, constructors, eliminators, closedness support, and computation rows for folds and case analysis.
- The non-Church files remain important because they isolate syntax forms without the surrounding encoding burden. They are the local probes for sort contexts, Pi formation, projection behavior, constants, ordinary applications, application tails, and beta redexes.

Thus the inventory should not be read as a flat list of examples. It is a layered gallery: atoms and binders first, then identity and application patterns, then Church data, then eliminators and computation rows over that data. The order matters for audit use because a later failure in a Church file can often be compared against the smaller non-Church probes to locate whether the issue is basic typing, binder shifting, eliminator typing, or beta computation.

9382.3 Representative theorems

The base Church boolean and Church natural-number rows are the smallest public examples. The boolean type and constructors are typed in the empty context:

```
↪ variant  BEDC.MetaCIC.church bool type
↪ variant  BEDC.MetaCIC.church true
↪ variant  BEDC.MetaCIC.church false
```

. The natural-number encoding and its first constructors are likewise typed:

```
↪ variant  BEDC.MetaCIC.church nat type
```

```
↪ variant BEDC.MetaCIC.church zero
↪ variant BEDC.MetaCIC.church one
↪ variant BEDC.MetaCIC.church succ
```

. These targets establish that the gallery's basic numerals and booleans are not paper-only conventions; they are terms accepted by the encoded typing relation.

The list gallery checks a type former, constructors, and a fold operator:

```
↪ variant BEDC.MetaCIC.church list type typed
↪ variant BEDC.MetaCIC.church nil typed
↪ variant BEDC.MetaCIC.church cons typed
↪ variant BEDC.MetaCIC.church fold typed
```

. This group exercises nested Pi structure, shifted element types, and eliminator typing. It is a compact stress test for binder indices because the encoded list type moves an element type through several binders.

The sum gallery records the either type, both injections, and case analysis:

```
↪ variant BEDC.MetaCIC.church either type typed
↪ variant BEDC.MetaCIC.church inl typed
↪ variant BEDC.MetaCIC.church inr typed
↪ variant BEDC.MetaCIC.church case either typed
```

. The accompanying beta rows, such as

```
↪ variant BEDC.MetaCIC.church case inl
↪ variant BEDC.MetaCIC.church case inr
```

, show that the eliminator computes on both injection branches under the displayed closedness inputs.

The equality gallery uses a Leibniz-style encoding:

```
↪ variant BEDC.MetaCIC.leibniz eq type
↪ variant BEDC.MetaCIC.leibniz refl type
↪ variant BEDC.MetaCIC.refl zero proof
↪ variant BEDC.MetaCIC.refl true proof
```

. These rows show a dependent proof object, not merely first-order data. They are useful witnesses that the same typed-example layer includes proposition-shaped programs.

The dependent and polymorphic identity examples give a separate binder-focused surface:

```
↪ variant BEDC.MetaCIC.lam dependent identity
↪ variant BEDC.MetaCIC.polymorphic identity
↪ variant BEDC.MetaCIC.poly identity type
↪ variant BEDC.MetaCIC.poly identity at sort type
```

. They test the common identity pattern in both direct dependent and explicitly polymorphic forms.

The application and beta-redex files make the gallery operational rather than only declarative:

```
↪ variant BEDC.MetaCIC.app id sort in empty ctx
↪ variant BEDC.MetaCIC.dep id applied to arg in sort ctx
↪ variant BEDC.MetaCIC.id sort redex
↪ variant BEDC.MetaCIC.id sort redex typed
```

. These examples connect the typing gallery to the beta relation used elsewhere in MetaCIC.

9382.4 Role within MetaCIC

The submodule functions as the typing relation's ground-truth gallery. A change to the typing implementation, closedness infrastructure, substitution helpers, or beta evaluator should preserve these examples unless the represented calculus itself has changed. In that sense the gallery is a regression surface for concrete syntax, binder shifting, context lookup, application typing, and closed Church-encoded programs.

The gallery also keeps the manuscript honest about examples. When the paper mentions Church booleans, naturals, lists, sums, equality, or identity programs inside MetaCIC, the cited rows above are the Lean-side witnesses that the terms exist with the stated typing or computation behavior.

9382.5 Connection to other audit maps

The closest neighbour is the decidability map in chapter 9379. Its `GalleryTests` layer checks that the type inference surface accepts the main closed example families. The present chapter records the worked examples themselves. The two layers should be read together: `TypedExamples` supplies the concrete programs, while the decidability gallery checks the option-valued inference interface on those program families.

The subject-reduction map in chapter 9377 has a different role. TypedExample terms that take beta steps should preserve typing along the expected route, but the audit map cannot promote that expectation into an unconditional subject-reduction theorem. The SR chapter names the preservation obligations that would be needed for that stronger claim.

The normalisation and confluence maps are also adjacent but not identical. The Church examples include concrete beta computations and some normalisation-style rows. They are witnesses that the reduction machinery runs on selected terms, not a proof of arbitrary closed-term normalisation or arbitrary closed-term Church–Rosser.

9382.6 Boundary

This chapter is an audit map for a worked-example collection. It may be cited for the displayed typed Church encodings, dependent identity examples, polymorphic identity examples, application examples, beta-redex rows, and selected computation rows.

It does not prove subject reduction, strong normalisation, confluence, or global conversion decidability. It does not claim that every Church-style encoding has all expected algebraic laws inside MetaCIC. It records the examples whose Lean targets are present and keeps their role at gallery level.

Concretized in §775: TypedExamples gallery rows are represented as a finite NameCert regression surface.

Concretized in §756: Typed example gallery rows are packaged as a finite NameCert carrier with checker, closedness, beta, and regression-surface obligations.

Concretized in §753: TypedExamples gallery rows are realized as a finite worked-example seal.

Concretized in §880: Typed example audit budgets realize the TypedExamples audit map as finite regression-spend NameCert rows.

Concretized in §922: Application-form examples are realized as finite typed-application route packets.

Concretized in §923: Typed example coverage matrices realize gallery, checker, beta, and regression-spend coverage as finite NameCert rows.

Concretized in §924: Church eliminator regression matrices realize bool, nat, list, and sum eliminator gallery rows as one finite checker and beta-regression NameCert packet.

Concretized in §2330: Typed-example reduction traces are realized as finite MetaCIC NameCert rows linking typed term, reduction step, checker result, closedness, transport, replay, and provenance.

Concretized in §6175: Typed gallery context probes realize sort contexts, binder shapes, applications, beta redexes, and Church examples as finite NameCert rows.

Concretized in §8235: Typed equality eliminator packets realize the Leibniz equality gallery as finite checker, closedness, reduction, transport, replay, provenance, and NameCert rows.

Chapter 9383

MetaCIC Normalization partial-closure audit map

9383.1 The setup

BEDC.MetaCIC's normalisation submodule is laid out across four sibling files: `AppAdequacy`, `ClosedSN`, `PiAdequacy`, and `Reducibility`. Together they aim at strong normalisation on closed terms via the candidate-mediated reducibility path.

The operative endpoint is StrongNormalizable(t). The module does not present that endpoint as a hidden global oracle. It exposes a reducibility candidate layer, a closed-candidate layer, candidate-set wrappers, Pi and application adequacy lemmas, and closed typed term projections from those witnesses.

This chapter records the exact reading discipline for that interface. The normalisation files contain many checked termination facts, but the public closed-term route is candidate-mediated. A manuscript citation must therefore name the supplied candidate evidence, closed candidate evidence, or candidate-set membership evidence; it must not silently cite an unconditional theorem for every closed typed term.

9383.2 What is established

The checked inventory has six parts.

- Reducibility candidate predicate. `Reducibility.lean` defines the candidate surface and proves that candidates imply strong normalisation:

  ↪ variant BEDC.MetaCIC.Candidate

  ↪ variant BEDC.MetaCIC.candidate implies sn

  ↪ variant BEDC.MetaCIC.candidate sn

  ↪ variant BEDC.MetaCIC.candidate closed under beta

  .

- Closed SN from candidate. `ClosedSN.lean` projects an empty-context typed closed term to strong normalisation when the corresponding candidate witness is supplied:

  ↪ variant BEDC.MetaCIC.closed typed term sn from candidate

  .

- Closed SN from closed candidate. The closed-candidate wrapper carries the closedness of the type and term together with the reducibility candidate, and the closed typed term projection consumes that wrapper:

  ↪ variant BEDC.MetaCIC.closed typed term sn from closedCandidate

  ↪ variant BEDC.MetaCIC.closedCandidate implies sn

  .

- Closed SN from candidate set. Candidate sets package membership, soundness, and beta-closure; closed typed terms inherit strong normalisation from membership in such a set:

  ↪ variant BEDC.MetaCIC.closed typed term sn from candidateSet

  ↪ variant BEDC.MetaCIC.candidateSet implies sn

  ↪ variant BEDC.MetaCIC.canonicalCandidateSet sound

  .

- App adequacy. Application cases are discharged when the function side supplies Pi-candidate or Pi-head candidate evidence and the argument supplies domain-candidate evidence. The step lemmas keep the result inside the same candidate-mediated path:

  ↪ variant BEDC.MetaCIC.app case candidate

  ↪ variant BEDC.MetaCIC.app case sn from pi candidate

  ↪ variant BEDC.MetaCIC.app case sn from candidate

  ↪ variant BEDC.MetaCIC.app case step sn

  .

- Pi adequacy. Pi candidates and neutral normal forms supply the function and binder-side termination facts used by the application layer and the typed examples:

  ↪ variant BEDC.MetaCIC.piCandidate implies sn

  ↪ variant BEDC.MetaCIC.PiAdequacy.piCandidate implies sn

  ↪ variant BEDC.MetaCIC.neutralNormal implies sn

  ↪ variant BEDC.MetaCIC.sn pi

  .

The no-infinite-reduction reading is also present. Strong normalisation excludes infinite beta chains, and the same candidate-mediated closed-term routes export that exclusion:

↪ variant BEDC.MetaCIC.strongNormalizable no infinite reduction

↪ variant BEDC.MetaCIC.closed typed term no infinite reduction from candidate

↪ variant BEDC.MetaCIC.closed typed term no infinite reduction from closedCandidate

↪ variant BEDC.MetaCIC.closed typed term no infinite reduction from candidateSet

.

9383.3 File-by-file audit reading

`Reducibility.lean` is the base layer. It defines BetaStrong, StrongNormalizable, candidate heads, candidates, closed candidates, Pi candidates, and candidate sets. The file also proves the core

closure facts: candidates imply SN, candidates are stable under one beta step, candidate sets export SN through their soundness field, and the canonical candidate set is sound and beta-closed.

The important audit point is that a candidate is not merely a term-level SN tag. It also carries the head-shape information needed by the Pi and application routes. For Pi types this head information includes the ability to send candidate arguments to strongly normalising applications. That is the beginning of the candidate-mediated application story, and it is why the application layer does not reduce to a simple componentwise SN claim.

`ClosedSN.lean` is the closed-term projection layer. It introduces ClosedTypedTerm, ClosedTypedCandidate, ClosedTypedPiCandidate, and ClosedTypedCandidateSetMember. The three main closed typed term theorems project SN from candidate, closed-candidate, and candidate-set evidence. The same file then mirrors those projections into no-infinite-reduction statements.

This file is the main public citation site for closed-term SN. Its statements are strong enough for the closed-normal branch of the consistency assembly, but their quantifier shape matters. Empty-context typing and closedness are present; the candidate witness is still a separate input. A citation that drops that input would cite a theorem that the file does not export.

`PiAdequacy.lean` supplies neutral and Pi-shaped adequacy. It contains a namespace wrapper for Pi-candidate SN, defines neutral normal forms, proves that neutral normality is stable under beta steps, and derives SN for neutral applications and displayed Pi constructions. It also records typed-example SN facts for the Church gallery. These example facts are useful witnesses that the machinery runs on concrete closed terms, not a replacement for the general candidate interface.

`AppAdequacy.lean` is the application-facing layer. It turns a PiCandidate function and a candidate argument into a candidate application, then projects SN and beta-step SN consequences. It also supplies closed application wrappers and small example applications. The file ends with the negative guardrail showing that componentwise SN of function and argument alone does not entail SN of their application.

The four files therefore form a staged route:

- `Reducibility` defines the candidate objects and their SN consequences;
- `ClosedSN` projects those consequences to closed typed term and no-infinite-reduction statements;
- `PiAdequacy` supplies neutral and Pi-shaped adequacy facts;
- `AppAdequacy` discharges application cases under candidate evidence and records the non-unconditional application boundary.

This route is partial closure in the precise BEDC sense. It closes the local candidate-mediated path and the named closed-term projections from that path. It does not erase the candidate layer or turn the examples into a theorem over all closed typed terms.

9383.4 What remains conditional

The normalisation files have no visible `sorry`, `admit`, or `axiom` marker in this submodule. The conditionality is instead at the interface: the closed typed term theorems consume candidate evidence, closed-candidate evidence, or candidate-set membership evidence. They do not construct such evidence for every empty-context typed closed term.

The sharp reading is therefore:

- candidate evidence implies strong normalisation;

- closed typed terms are strongly normalising once they are placed inside the candidate, closed-candidate, or candidate-set interface;
- Pi and application adequacy show how the candidate interface behaves over the displayed Pi and application fragments;
- the module does not export a blanket theorem of the form every closed typed BEDC.MetaCIC term is strongly normalising without candidate-mediated evidence.

This boundary is checked negatively on the application side. The file `AppAdequacy.lean` records a self-application counterexample to the claim that strong normalisation of parts alone gives unconditional strong normalisation of applications:

```
↪ variant  BEDC.MetaCIC.selfApplyApp not sn
↪ variant  BEDC.MetaCIC.app sn from parts is not unconditional
```

. Thus the word "candidate-mediated" is not stylistic caution. It names the actual guardrail that prevents the paper from overreading componentwise termination as global application termination.

9383.5 Comparison with SR obstruction and Confluence audit map

The subject-reduction map in chapter 9377 isolates a dependent-codomain obstruction: closed subject reduction is blocked unless four named preservation obligations are supplied. The confluence map in chapter 9378 isolates a residual-substitution boundary for the full parallel diamond and arbitrary closed star confluence.

Normalisation has a different shape. It is not currently blocked by an explicit proof-hole marker inside the normalisation submodule, and it is not the same as the SR dependent-codomain obstruction or the confluence substitution boundary. Its public frontier is the reducibility-candidate interface: the checked SN projections are closed over the candidate evidence they consume, while the manuscript must not erase that evidence from the statement being cited.

For full closed consistency, the three maps meet in chapter 9375. Strong normalisation supplies the closed-normal branch through the candidate-mediated path. Subject reduction would have to transport arbitrary closed proofs while preserving the displayed false type. The no-closed-normal-proof-of-false theorem then rules out the normal endpoint. The present chapter records the SN pillar; it does not close the subject-reduction transport pillar.

9383.6 Apophatic framing and boundary

This audit map applies the same apophatic discipline as the SR and confluence maps. BEDC names the open cell instead of silently promoting a conditional interface into an unconditional closure theorem. The map is not a new normalisation proof. It is a paper-side handle for the exact proof state exposed by the Lean normalisation files.

The available positive claims are the checked candidate-to-SN lemmas, the closed typed term projections from candidate evidence, the candidate-set projections, the Pi and application adequacy lemmas, the example-level SN facts, and the no-infinite-reduction consequences. The explicit boundary is equally binding: this chapter does not claim global strong normalisation for every closed typed MetaCIC term without candidate-mediated evidence, does not turn componentwise application SN into unconditional application SN, and does not close the subject-reduction

obligations needed by full closed consistency.

Concretized in §755: Candidate-mediated normalization is represented as a finite budgeted NameCert surface, not as an unconditional closed-term theorem.

Concretized in §750: MetaCIC normalization audit packets realize the finite strong-normalization scope, confluence-boundary refusal, transport, route replay, provenance, and local NameCert rows.

Concretized in §777: The normalization audit map is realized as the finite candidate-mediated MetaCICNormalizationAuditSeal packet.

Concretized in §758: Candidate-mediated strong-normalization routes are realized as finite NameCert packets over candidate evidence, closed-candidate evidence, candidate-set membership, Pi/app adequacy rows, and no-infinite-reduction projections.

Concretized in §907: Pi-application adequacy routes are realized as finite candidate-mediated NameCert packets over Pi evidence, argument evidence, application candidate rows, closed typed projections, and no-infinite-reduction boundaries.

Concretized in §1108: Candidate-mediated normalization routes are packaged as a finite NameCert router.

Concretized in §1190: MetaCIC normalization and confluence audit routes are realized as a finite bridge packet over the subject-reduction and closed-consistency boundary.

Concretized in §2226: The normalization audit socket is realized as finite candidate, closed-candidate, obstruction, frontier, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8668: Normalization audit rows are realized as finite candidate-set, fuel-trace, closed-typing-fragment, residual-checker, normal-endpoint, refusal, transport, replay, provenance, and NameCert data.

Concretized in §8604: Candidate-mediated redex frontiers realize the normalization audit map as finite typed-term, candidate, residual-budget, beta-window, and strong-normalization handoff rows.

Concretized in §5936: MetaCIC normalization fuel boundaries realize candidate-mediated bounded-fuel traces as finite closed-term, candidate, fuel, trace, closedness-preservation, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §5547: Candidate-mediated normalization, residual confluence visibility, and decidable-boundary routing are realized as one finite handoff packet.

Concretized in §3539: The candidate-mediated typed-normalization route is realized as a finite NameCert audit packet.

Concretized in §2320: Candidate-set normalization is realized as finite candidate evidence, closed-term projection, application/Pi adequacy, no-infinite-reduction, obstruction, transport, continuation, provenance, and NameCert rows.

Concretized in §2337: Finite NameCert landing for candidate-mediated normalization windows and confluence boundaries.

Concretized in §892: The normalization frontier packet separates candidate evidence, closed-normal use, and subject-reduction obstruction rows as finite MetaCIC audit data.

Concretized in §3422: Candidate-set strong-normalization handoffs are realized as finite CandidateSet membership, closed-term, beta-closure, and SN export NameCert data.

Concretized in §3566: Candidate-mediated normalization evidence is realized as a finite socket packet separating candidate rows from global SN claims.

Concretized in §5210: Candidate-mediated normalization routing is realized as finite closed-term, candidate-evidence, adequacy-handoff, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §3779: Candidate-mediated normalization audit rows are packaged as a finite NameCert carrier for closed-term SN and no-infinite-reduction reads.

Concretized in §3871: Candidate-mediated normalization evidence budgets are realized as finite candidate, closed-candidate, candidate-set, Pi/app adequacy, projection, and no-infinite-reduction rows.

Concretized in §3919: Pi adequacy budget for candidate-mediated strong normalization.

Concretized in §4196: Bounded normalization candidates realize the finite candidate, fuel, closed-term, Pi/application adequacy, and no-infinite-reduction frontier rows used by the MetaCIC normalization audit map.

Concretized in §5351: Candidate-set membership routes closed typed terms to strong normalization without erasing the candidate evidence.

Concretized in §5163: Candidate-mediated strong-normalization evidence is realized as a finite packet over candidate, closure, adequacy, no-infinite-reduction, obstruction, and MetaCIC frontier route rows.

Concretized in §5942: Ground-compiler normalization routes are realized as finite generated-term, candidate-evidence, adequacy, no-infinite-reduction, obstruction, replay, provenance, and NameCert rows.

Concretized in §6032: GroundCompiler strength monotonicity is realized as finite accepted-flow, lower derivation-certificate subflow, route-budget, candidate-socket, transport, provenance, and local NameCert rows.

Concretized in §5994: Candidate-mediated normalization frontiers are realized as finite evidence/refusal NameCert rows.

Concretized in §6027: Finite beta-window confluence packets realize the normalization audit map as redex-frontier, local-join, substitution-boundary, and closed-normal finite NameCert rows.

Concretized in §6085: Candidate-mediated SN handoff realizes the normalization audit map as finite NameCert data.

Concretized in §6306: Neutral reifier packets realize the MetaCIC normalization audit route as finite neutral, semantic value, readback, transport, replay, provenance, and NameCert rows.

Concretized in §6314: Finite-window normalization audit packet for MetaCIC redex-frontier inspection.

Concretized in §6552: Candidate-mediated normalization and local confluence budgets are realized as finite candidate, closed-candidate, candidate-set, peak, join-budget, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §6666: Residual-checker normalization budgets realize candidate-mediated SN through finite checker, fuel, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6617: Typed normal-form schedulers realize the normalization audit map as finite closed-term, candidate-gate, scheduler-budget, typed-endpoint, obstruction, router, transport, replay, provenance, and NameCert rows.

Concretized in §7242: The normalization substitution-boundary packet realizes the candidate-mediated normalization obstruction as finite NameCert rows.

*Concretized in §**??**:* Finite observation diamond envelopes route candidate evidence, bounded endpoint rows, residual-envelope reads, local checker status, blocked-edge retention, transport, replay, provenance, and NameCert rows before bounded joins are consumed.

Concretized in §5157: Proof-trace normal forms realize the normalization audit route as finite proof trace, typed endpoint, normal-form readback, transport, replay, provenance, and NameCert rows.

Concretized in §8559: Bounded MetaCIC diamond windows realize candidate-mediated normalization and residual confluence as finite NameCert data.

Concretized in §8724: Bounded normalization counterexample budgets realize the finite obstruction row of the MetaCIC normalization audit map.

Concretized in §8795: The normalization-confluence frontier realizes the candidate-mediated SN route together with residual-join and confluence-boundary rows as finite NameCert data.

Concretized in §8854: Finite normalization diamonds realize the MetaCIC normalization audit map as candidate-mediated local join, no-infinite-branch, transport, replay, provenance, and NameCert rows.

Chapter 9384

MetaCIC Beta module audit map

9384.1 The setup

BEDC.MetaCIC's beta layer is the driving reduction relation used by the subject-reduction, confluence, normalisation, evaluation, and closed-consistency interfaces. Its public surface is deliberately small. The single-step relation is declared in `Beta.lean`, while conversion by common beta reduct is declared in `Beta/Conversion.lean`.

The two files should be read together. `Beta.lean` gives the raw generator: a one-step relation BetaStep over MetaCIC terms, with one direct beta-redex constructor and congruence constructors for applications, lambdas, and Pi types. `Beta/Conversion.lean` builds the conversion interface BetaConv from the multi-step relation BetaStarStep and records the checked conversion lemmas that later modules may cite.

This chapter is an audit map, not another proof layer. It records which beta objects are constructors, which conversion facts are checked, and where the relation meets the subject-reduction obstruction isolated in chapter 9377.

9384.2 Core inductive definition

9384.2.1 BetaStep

The core declaration is the inductive relation
Lean4 (definition) `BEDC.MetaCIC.BetaStep`
. It has seven constructors.

- `beta` contracts an application whose function is a lambda: app(lam *dom body*) *arg* steps to substitute 0 *arg body*.
- `congApp1` reduces the function position of an application while leaving the argument fixed.
- `congApp2` reduces the argument position of an application while leaving the function fixed.
- `congLam` reduces the body of a lambda while leaving the displayed domain fixed.
- `congPiCod` reduces the codomain of a Pi type while leaving the displayed domain fixed.
- `congPiDom` reduces the displayed domain of a Pi type while leaving the codomain syntactically fixed.

- `congLamDom` reduces the displayed domain of a lambda while leaving the body syntactically fixed.

The constructor list is intentionally syntax-directed. A beta step is not a typing judgement, not a semantic equivalence, and not a closure certificate. It only says that the represented term syntax admits one local reduction at a displayed site.

9384.2.2 File-level reading

`Beta.lean` is therefore the root reduction file, not a theorem bundle. Its exported object is the closed inductive grammar of one-step beta movement. Every later theorem that mentions beta reduction receives one of these constructors, or a finite chain generated from them, as its reduction witness.

`Beta/Conversion.lean` is the first theorem bundle over that grammar. It does not add a new reduction rule. It changes the reading mode from one-step movement to common-reduct comparability, and then proves the small algebra of that comparison relation that is already available in the substrate.

Two features matter for later audits. First, beta contraction uses the concrete MetaCIC substitution operation, so all preservation claims that read across the redex must account for substitution through binders. Second, the domain congruence constructors for lambda and Pi terms change displayed binder annotations without reducing the body or codomain at the same time. These are valid reduction steps, but their typing behaviour requires separate evidence.

9384.3 Established results

The conversion file exposes BetaConv as common-reduct conversion: there is a term w such that both sides reduce to w by BetaStarStep.

```
↪ variant BEDC.MetaCIC.BetaConv
```

The elementary conversion facts are checked directly:

- reflexivity and symmetry:

  ```
  ↪ variant BEDC.MetaCIC.BetaConv.refl
  ↪ variant BEDC.MetaCIC.BetaConv.symm
  ```

  ;

- conversion from a multi-step beta reduction on either side:

  ```
  ↪ variant BEDC.MetaCIC.BetaConv.of betaStar left
  ↪ variant BEDC.MetaCIC.BetaConv.of betaStar right
  ```

  ;

- conversion from a one-step beta reduction on either side:

  ```
  ↪ variant BEDC.MetaCIC.BetaConv.of betaStep left
  ↪ variant BEDC.MetaCIC.BetaConv.of betaStep right
  ```

  .

The file also records the conditional transitivity shape. Full transitivity is obtained from an explicit confluence input, while a normal middle term gives a smaller unconditional route:

```
↪ variant BEDC.MetaCIC.BetaConv.trans of confluence
```

↪ variant BEDC.MetaCIC.BetaConv.trans normal middle

↪ variant BEDC.MetaCIC.BetaConv.trans closed normal middle

Closed normal forms are rigid under beta conversion when both sides are closed and beta-strong:

↪ variant BEDC.MetaCIC.BetaConv normal forms equal closed

The same conversion file also names a typing-side boundary. The closed beta-conversion typing invariance statement is represented as a proposition, and the file proves a concrete closed lambda-domain obstruction against it:

↪ variant BEDC.MetaCIC.HasTypeBetaConvInvariantClosedStatement

↪ variant BEDC.MetaCIC.closedLamDomainTarget closed

↪ variant BEDC.MetaCIC.BetaConv closed lam domain type obstruction

↪ variant BEDC.MetaCIC.not HasType BetaConv invariant closed

These facts are conversion facts and obstruction facts. They are not a blanket permission to replace displayed types by beta-convertible types inside typing derivations. Any citation must preserve the exact input shape: common reduct, normal middle term, closed beta-strong normal forms, or the displayed negative typing boundary.

9384.4 Connection to the SR obstruction

chapter 9377 names four discharge obligations for the parameterised subject-reduction theorem. The beta module is where the reduction triggers behind those obligations are born.

The direct `beta` constructor is the trigger for BetaSubstitutionPreservation. A term app(lam *dom body*) *arg* can reduce unconditionally to substitute 0 *arg body*. To turn that syntactic step into typing preservation, the substrate must show that substituting the argument into the body also transports the displayed codomain in the expected way. That is a typing statement about substitution, not a consequence of the beta constructor alone.

The `congApp2` constructor is the trigger for the dependent-codomain application obstruction. It reduces the argument while leaving the function and the originally displayed application type fixed. In an independent codomain regime this can be harmless, and the subject-reduction map records the checked independent-codomain fragment. In a dependent codomain regime the codomain can observe the argument, so reducing the argument may change the readback type. That is the reason AppArgTypeStable is a separate obligation.

The `congLamDom` and `congPiDom` constructors are the binder-domain triggers. They reduce displayed binder annotations. A lambda or Pi term can therefore take a syntactic beta step even when the typing derivation for the old annotation does not transfer to the new annotation at the same displayed external type. These are the sources of LamDomainSubjectReduction and PiDomainSubjectReduction.

Thus the beta relation itself is unconditional, but subject reduction under that relation is not unconditional in the present MetaCIC interface. The distinction is structural: BetaStep authorizes one syntax step, while subject reduction must also preserve typing data across the dependent codomain and binder interfaces touched by that step.

Concretized in §728: Beta congruence-domain audit packets realize the binder-domain beta constructors as finite subject-reduction trigger surfaces.

9384.5 Apophatic framing and boundary

The beta audit map gives BEDC a precise negative reading discipline. The paper may cite `BetaStep` as the checked single-step reduction relation. It may cite the conversion lemmas listed above. It may cite the closed normal-form rigidity theorem and the checked obstruction to closed typing invariance under beta conversion.

The paper may not infer a hidden conversion rule for typing, an unconditional closed subject-reduction theorem, or a full closed consistency proof from the existence of BetaStep. A beta step is a syntactic generator. Its typing consequences require the exact preservation interfaces named in chapter 9377.

This is the apophatic role of the module boundary. The beta layer says exactly what can reduce. The conversion layer says exactly which common-reduct facts are available. The subject-reduction map says exactly which typing transports are not supplied by those reduction facts alone. Keeping the three readings separate prevents the manuscript from treating reduction, conversion, and typed preservation as one unrecorded operation.

Concretized in §760: Typed beta congruence trigger packets realize the beta audit map's congruence-constructor typing boundary as a finite MetaCIC NameCert surface.

Concretized in §761: The beta audit map is realized as a finite NameCert packet separating reduction, conversion, and typing-preservation boundaries.

Concretized in §754: Beta-step boundaries are realized as finite BetaStep, BetaConv, obstruction-trigger, hsame, Cont, Pkg, and NameCert rows.

Concretized in §879: Beta-reduction audit trails realize the beta audit map as finite BetaStep, BetaConv, obstruction-trigger, subject-reduction socket, hsame, Cont, Pkg, and NameCert rows.

Concretized in §881: Critical-pair packets concretize the beta audit map as finite local-overlap and common-reduct NameCert data.

Concretized in §881: Critical-pair packets realize the beta audit map through a finite local-overlap, branch, common-reduct, and non-escape route.

Concretized in §7339: Eta-expansion audit packets realize the beta audit map as finite eta-row, beta-conversion-boundary, normalization-frontier, obstruction-retention, transport, replay, provenance, and NameCert data.

Concretized in §7470: MetaCIC beta redex spines realize the beta audit map as finite Term, BetaStep, BetaConv, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §7682: The closure-substitution frontier realizes the beta-audit obstruction route as finite beta-trigger, ClosedAt, substitution-boundary, and subject-reduction rows.

Concretized in §7714: The substitution-sensitive beta diamond frontier is realized as finite peak, residual, candidate-join, substitution-boundary, checker, obstruction, hsame, Cont, Pkg, and NameCert rows.

Chapter 9385

MetaCIC audit synthesis: closed-CIC critical-path overview

9385.1 The setup

BEDC.MetaCIC is the closed CIC meta-theory formalisation inside the BEDC project. Its closed-term meta-properties are distributed across the subject reduction, confluence, normalisation, substitution, decidable-checking, and consistency-assembly files. The relevant paper audit maps are chapter 9377, chapter 9378, chapter 9383, chapter 9380, chapter 9379, and chapter 9375.

This chapter gives a single reader-facing overview of that state. It does not replace the sibling audit maps. It points to them, aligns their boundaries, and extracts the closed-MetaCIC critical path: which nodes are already available, which nodes are conditional, and where unconditional closed consistency is blocked.

9385.2 The five submodule audit maps

9385.2.1 Reading rule

The five maps are not interchangeable evidence. Each map exports a different kind of checked object. Subject reduction exports an obstruction and a four-input preservation interface. Confluence exports reduction-geometry fragments and conditional bridges. Normalisation exports reducibility-candidate routes. Substitution exports closed syntactic laws. Decidability exports executable checking surfaces.

A citation to one map therefore cannot silently stand in for another. A substitution lemma may support the beta case of a later preservation proof, but it is not the preservation proof. A confluence bridge may show how a supplied diamond would flow to closed star confluence, but it is not the supplied diamond. A decidable checker may certify a displayed typing computation, but it is not a decision procedure for all meta-theoretic predicates. This is the reading discipline used throughout this synthesis.

9385.2.2 Subject reduction

The subject-reduction map records a parameterised preservation theorem, four documented discharge obligations, and a checked obstruction theorem for the unparameterised closed subject-reduction statement. Closed subject reduction is therefore explicitly not available in the present

substrate; see chapter 9377.

9385.2.3 Confluence

The confluence map records a substitution-conditional parallel diamond, bridges from closed parallel diamond to closed star confluence, and unconditional fragments for closed normal sources and atom shapes. It does not claim arbitrary closed-term Church–Rosser; see chapter 9378.

9385.2.4 Normalization

The normalisation map records the candidate-mediated strong-normalisation path for closed typed terms. Candidate, closed-candidate, or candidate-set evidence delivers strong normalisation and no-infinite-reduction consequences, but the map does not erase that evidence into a blanket theorem over all closed typed terms; see chapter 9383.

9385.2.5 Substitution

The substitution map records unconditional closed syntactic laws: `shift_closed`, `substitute_closed`, and closed composition fragments. These facts supply syntax-side binder movement, but they do not discharge beta typing preservation or full subject reduction; see chapter 9380.

9385.2.6 Decidability

The decidable map records executable equality and type-checking surfaces, their soundness and completeness wrappers, gallery sanity witnesses, and bounded normal-form comparison under finished-normalisation evidence. It is a local checking layer, not a decision oracle for subject reduction, confluence, normalisation, or consistency; see chapter 9379.

9385.3 The critical path to unconditional closed consistency

The closed-MetaCIC consistency assembly in chapter 9375 can be read as the following critical path.

1. Closed strong normalisation is available through the candidate-mediated normalisation interface. Its public witnesses include

   ↪ variant BEDC.MetaCIC.closed typed term sn from candidate

   ,

   ↪ variant BEDC.MetaCIC.closed typed term sn from closedCandidate

   , and

   ↪ variant BEDC.MetaCIC.closed typed term sn from candidateSet

   .

2. No closed normal proof of false is available:

   ↪ variant BEDC.MetaCIC.no closed normal proof of false

   .

3. Closed normal consistency assembly is available:

   ↪ variant BEDC.MetaCIC.closed normal consistency assembly

   .

4. Closed reduction by subject reduction is blocked by the dependent-codomain obstruction:

   ↪ variant BEDC.MetaCIC.closed consistency reduction obstruction not available

   .

5. The discharge bundle required for an unconditional preservation lift is explicitly named:

   ↪ variant BEDC.MetaCIC.SubjectReductionDischargeBundle

   .

6. Unconditional closed consistency remains blocked until the bundle is supplied and the closed-normal theorem can be transported from arbitrary closed proofs to their normal endpoints.

The same dependency can be displayed as a compact audit graph:

```
Candidate-mediated closed SN
  status: AVAILABLE
  feeds:
    closed-normal consistency branch

No closed normal proof of False
  status: AVAILABLE
  witness:
    BEDC.MetaCIC.no_closed_normal_proof_of_false

Closed normal consistency
  status: AVAILABLE
  witness:
    BEDC.MetaCIC.closed_normal_consistency_assembly

Closed reduction by subject reduction
  status: BLOCKED
  reason:
    dependent-codomain obstruction
  required interface:
    BEDC.MetaCIC.SubjectReductionDischargeBundle

Unconditional closed consistency
  status: BLOCKED
  reason:
    closed-normal consistency cannot be lifted without closed reduction
```

9385.4 Status matrix

The same state can be read as a matrix of positive results and non-claims.

Map	Available	Not claimed
Subject reduction	parameterised preservation, four named inputs, checked obstruction	unparameterised closed subject reduction
Confluence	conditional parallel diamond route, closed normal and atom fragments	arbitrary closed Church–Rosser
Normalisation	candidate-mediated closed strong normalisation routes	candidate-free global strong normalisation
Substitution	closed shift, closed substitution, closed composition fragments	unrestricted substitution calculus or beta typing preservation
Decidability	term equality, type-checking soundness and completeness, bounded comparison	global decision procedure for meta-theoretic properties
Assembly	closed-normal consistency	unconditional closed consistency

The matrix is deliberately asymmetric. The positive column records what the Lean substrate and paper maps can be used for directly. The non-claim column records the strongest nearby statement that must not be inferred from that positive result. The synthesis depends on this asymmetry: the project has many checked local nodes, but the closed-consistency endpoint needs a specific transport edge that those local nodes do not themselves supply.

9385.5 How the maps interact

The interaction pattern is a supply chain with named handoffs rather than a single theorem with hidden premises. Substitution supplies closed syntactic facts about moving terms across binders. Subject reduction would have to turn the relevant binder movement into type preservation across beta and domain congruence. Normalisation supplies the target normal-form side once the proof term is inside the candidate-mediated strong-normalisation interface. Consistency assembly consumes the normal-form theorem, but only after the displayed false type has been preserved along reduction.

Confluence and decidability sit beside that path. Confluence describes how reduction branches can be joined under the available and conditional diamond interfaces. Decidability describes which local syntactic and typing questions are executable inside the represented fragment. Both constrain how later work may cite the substrate, but neither bypasses the subject-reduction lift.

The useful reading is therefore:

1. Substitution is syntax algebra.
2. Subject reduction is type-preservation transport.
3. Normalisation is termination-to-normal-form evidence.
4. Closed-normal consistency is the contradiction eliminator at the normal endpoint.
5. Full closed consistency is the composition of transport and the normal endpoint.

The missing edge is item 2 in the closed arbitrary-proof setting. Once that edge is unavailable, items 3 and 4 remain valid but cannot be composed into item 5 for every closed proof term.

Concretized in §765: MetaCIC critical-path packets realize the audit synthesis as finite subject-reduction, normalisation, decidability, obstruction, transport, and NameCert rows.

Concretized in §776: MetaCIC audit synthesis is realized as a finite NameCert packet over the closed-normal branch, obstruction ledger, and discharge rows.

Concretized in §3519: MetaCIC redex-choice frontier packets realize the candidate-mediated normalization and confluence boundary as finite BEDC rows.

Concretized in §6547: The confluence diamond ledger realizes the MetaCIC audit synthesis confluence branch as finite candidate, critical-pair, residual-substitution, local-join, transport, and NameCert rows.

Concretized in §8284: MetaCIC candidate-pruning budgets realize the audit-synthesis frontier as finite candidate, pruning-budget, residual-frontier, bounded-checker, obstruction, transport, replay, provenance, and NameCert rows.

9385.6 The frontier shape

The substrate currently establishes normal-form sub-consistency unconditionally. This is the strongest closed-consistency node available: a closed proof term that is already beta-strong cannot inhabit the internal false encoding. The result is sharp because it states exactly the normal-form fragment, and it is useful because it is the endpoint that a successful subject-reduction lift would consume.

The lift to full closed consistency requires one of two project-level moves. The first route is to discharge the four subject-reduction hypotheses named in chapter 9377. The second route is to extend the substrate beyond the present closed-CIC surface so that the missing context-reduction and dependent-codomain readback lemmas become expressible and provable. On the current substrate, the checked obstruction theorem prevents the paper from treating that lift as a local proof exercise.

Confluence and decidability do not remove this frontier. Confluence supplies reduction geometry and conditional bridges, while decidability supplies local checking procedures. Substitution supplies closed syntax algebra. All three are necessary audit context, but none of them replaces the subject-reduction transport needed to carry an arbitrary closed proof of false to the closed normal-form theorem.

9385.7 Apophatic discipline applied to meta-theory

The MetaCIC audit maps apply BEDC's apophatic naming discipline to the project's own meta-theory. The substrate names what it has: checked closed normal consistency, candidate-mediated normalisation, closed substitution laws, local decidable checking, atom and normal-source confluence fragments, and conditional bridges. It also names what it does not have: unparameterised closed subject reduction, arbitrary closed Church–Rosser, unrestricted substitution commutation, a global decidability oracle, and unconditional closed consistency.

This synthesis chapter combines those names into one critical-path map. Its purpose is to prevent the reader from overreading a neighbouring checked layer as the missing endpoint. The current state is strong, but it is not ambiguous: the assembled theorem is closed-normal consistency, and the blocked edge is the subject-reduction lift across the dependent-codomain frontier.

9385.8 Boundary

This chapter is an audit overview, not a proposal and not a proof of any open obligation. It does not discharge the subject-reduction bundle, does not claim full closed confluence, does not strengthen the candidate-mediated normalisation interface, and does not turn decidable local checking into a global meta-theoretic decision procedure.

The claim is exactly the synthesis claim: the five audit maps and the consistency assembly form a closed critical-path diagram for the present BEDC.MetaCIC state. The available endpoint is normal-form closed consistency. The unavailable endpoint is unconditional closed consistency, blocked at the named subject-reduction discharge interface.

*Concretized in §**??**:* MetaCIC audit synthesis is realized as a finite NameCert packet over submap status rows, closed-normal consistency rows, and the blocked subject-reduction handoff ledger.

Concretized in §844: Closed consistency gap packets realize MetaCIC audit synthesis as finite rows separating closed-normal consistency from the subject-reduction discharge socket.

Concretized in §904: The MetaCIC audit synthesis matrix is realized as a finite NameCert packet for available rows, non-claim rows, blocked edges, synthesis consumers, and non-escape ledgers.

Concretized in §1063: MetaCIC proof-object audit routes are realized as finite NameCert rows tying closed-normal evidence to the subject-reduction blocked-edge ledger.

Concretized in §1056: The MetaCIC audit synthesis is realized as a finite discharge-obstruction lattice over available, blocked, and required closed-CIC audit rows.

Chapter 9386

Closed-MetaCIC open problems: consolidated obligation list

9386.1 The setup

The MetaCIC audit maps deliberately separate positive checked content from the places where the closed-term consistency route is still blocked. The subject reduction map, confluence map, normalisation map, substitution map, decidability map, typed-example map, and consistency assembly record those boundaries in their own local vocabularies: chapter 9377, chapter 9378, chapter 9383, chapter 9380, chapter 9379, chapter 9382, and chapter 9375.

This chapter is the companion inventory. The synthesis view gives the reader the critical path; the present chapter gives the concrete list of outstanding obligations, obstructions, partial-closure boundaries, and frontier questions. It does not introduce a new proof principle, a new Lean target, or a stronger closure claim. It is a single reading surface for the work that remains open after the existing audit chapters are read together.

The consolidated list has four layers. First, the subject-reduction route has four named discharge hypotheses and a checked obstruction to replacing them by an unconditional closed subject-reduction theorem. Second, confluence and substitution have algebraic boundaries around residual substitution and unrestricted commutation. Third, normalisation and decidability have candidate-mediated or bounded-computation interfaces that should not be erased by prose. Fourth, the worked examples are gallery-level witnesses, not a coverage theorem over all dependent or polymorphic programs.

9386.2 Subject reduction obligations (4)

The subject-reduction frontier is the sharpest obstruction in the closed MetaCIC path. The parameterised theorem exists, but it consumes four discharge hypotheses. The checked obstruction in chapter 9377 prevents the paper from replacing that four-input interface by a blanket closed subject-reduction claim for the present substrate.

1. BetaSubstitutionPreservation. The beta-redex case must show that substituting a typed argument into a typed body preserves the substituted codomain. The obstruction blocks an unconditional proof because the dependent codomain can observe the argument strongly enough for the displayed target type to change after reduction.

2. AppArgTypeStable. The application-argument congruence case must show that replacing an

argument by its reduct leaves the codomain readback stable. The independent-codomain case is checked, but the dependent codomain case is exactly where the counterexample lives.

3. LamDomainSubjectReduction. The lambda-domain congruence case must preserve the type of a lambda when its displayed domain annotation reduces. This is not a direct consequence of syntactic beta stepping, because the binder annotation participates in the typing judgement being preserved.

4. PiDomainSubjectReduction. The Pi-domain congruence case must preserve the type of a Pi term when the displayed domain reduces. This is separate from the lambda case because the Pi codomain is checked under the binder induced by the domain, so domain reduction changes the context-facing data.

All four obligations are simultaneous. Supplying only the beta redex row, only argument stability, or only binder-domain preservation leaves the parameterised subject-reduction theorem unavailable as an unconditional closed theorem. The first-class bundle records exactly the simultaneous discharge interface:

```
↪variant BEDC.MetaCIC.SubjectReductionDischargeBundle
```

.

Concretized in §6177: Closed application-argument stability packets realize the independent-codomain AppArgTypeStable route as finite closed-function, argument-step, typing, closedness, obstruction-boundary, transport, replay, provenance, and NameCert rows.

9386.3 Confluence remaining questions

The confluence map in chapter 9378 has a different shape from the subject-reduction map. It is not blocked by dependent typing. It is blocked at the residual-substitution algebra needed for the full parallel diamond and then for arbitrary closed star confluence.

The unconditional confluence content is shape-limited. Closed normal sources, sort sources, closed variable impossibility, atom joins, non-lambda application shape facts, componentwise diamonds, and parallel-to-star transport are available at the strengths stated in the confluence audit map. Those results are real checked infrastructure and may be cited at that scope.

The remaining confluence questions are the stronger endpoints. Does the present substrate support the needed BetaParallelSubstitutionCompatibility interface without importing unclosed substitution assumptions? Can the closed parallel diamond be supplied for arbitrary closed terms rather than consumed as a premise? Can the resulting closed multi-step Church–Rosser theorem be stated without hiding the substitution boundary? Until those inputs exist, arbitrary closed-term confluence remains conditional rather than unconditional.

Concretized in §2181: Residual-substitution boundary packets realize the MetaCIC open confluence obstruction as finite redex-frontier, residual-route, local-join, bounded-checker, subject-reduction-boundary, transport, replay, provenance, and NameCert rows.

Concretized in §3808: Bounded residual-substitution frontier packets realize the confluence obstruction as finite redex-frontier, residual-route, bounded-checker, local-join, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §3827: MetaCIC confluence residual-checker packets realize the residual-substitution frontier as typed-window, redex-frontier, bounded-checker, local-join, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §7088: Typed residual-budget packets realize the MetaCIC residual-checker frontier as finite typed-window, residual-frontier, checker-route, local-join, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §3921: The confluence residual envelope realizes the open residual-substitution confluence boundary as finite typed-window, residual-frontier, substitution-boundary, checker-status, fuel, neutral-spine, normalization-budget, transport, replay, provenance, and NameCert rows.

Concretized in §5099: Finite observation confluence budgets realize the MetaCIC residual-confluence frontier as finite typed-window, local-join, bounded-checker, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §6169: Closed parallel diamond obstructions realize the MetaCIC confluence frontier as finite premise-ledger, residual-boundary, residual-checker, observation-budget, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6419: MetaCIC substitution-diamond packets realize the residual-substitution frontier as finite typed-window, substitution-route, residual-frontier, bounded local-diamond checker, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §6857: MetaCIC residual diamond schedulers realize the residual-substitution confluence frontier as finite typed-window, residual-frontier, candidate-SN evidence, bounded local-join checker, L10-source, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6955: Residual-substitution compatibility windows realize the bounded confluence obstruction as finite typed-window, residual-frontier, substitution-boundary, local-join, checker-status, transport, replay, provenance, and NameCert rows.

Concretized in §7042: A finite residual-pair local-diamond witness packet realizes one MetaCIC confluence frontier from displayed candidate and substitution rows.

Concretized in §7219: A finite residual-pair local-diamond witness packet realizes one bounded MetaCIC confluence frontier.

Concretized in §7378: MetaCIC residual-join budgets realize the confluence frontier as finite typed-window, residual-premise, residual-pair, join-candidate, closed-checker, bounded local-join, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7407: MetaCIC residual-diamond budgets realize the residual-substitution confluence frontier as finite typed-window, residual-pair budget, join-candidate, bounded-checker, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7989: MetaCIC parallel-diamond selectors realize the closed parallel-diamond frontier as finite typed-window, residual-pair, diamond-candidate, compatibility-boundary, selector, obstruction, TasteGate, transport, replay, provenance, and NameCert rows.

Concretized in §7500: MetaCIC residual-join gauges realize the residual-substitution confluence frontier as finite residual-pair, bounded local-join, checker, closed-seal, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §8191: MetaCIC residual-choice budgets realize the bounded confluence frontier as finite typed-window, residual-pair, choice-boundary, bounded local-join, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §7763: Finite proof-DAG frontier packets realize the MetaCIC open-problem route as finite node, edge, checker, normalization-frontier, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §8046: MetaCIC residual local-diamond witnesses realize the residual-substitution confluence frontier as finite residual-pair admission, bounded local-join checker visibility, obstruction retention, transport, replay, provenance, and NameCert rows.

Concretized in §8283: MetaCIC confluence budget matrices realize the residual-substitution, closed parallel-diamond, and closed star-confluence frontier as finite typed-window, local-join, budget-matrix, obstruction, TasteGate-obligation, transport, replay, provenance, and NameCert rows.

9386.4 Normalization remaining questions

The normalisation map in chapter 9383 is candidate-mediated. Candidate evidence, closed-candidate evidence, and candidate-set membership imply strong normalisation; the application and Pi adequacy layers explain how that evidence behaves over the displayed fragments.

The remaining question is not whether the candidate projections are checked. They are checked. The question is whether every empty-context typed closed MetaCIC term can be placed into the relevant candidate interface without adding an unrecorded semantic assumption. The present normalisation layer does not export a blanket theorem saying that all closed typed terms are strongly normalising independent of candidate evidence.

This boundary matters for the consistency route. Closed-normal consistency can use beta-strong normality as an input, and candidate evidence can supply such normality for terms inside the candidate interface. Full closed consistency would still require a bridge from arbitrary closed proofs to a typed normal endpoint with the displayed false type preserved, which returns the problem to the subject-reduction frontier.

Concretized in §4366: MetaCIC candidate closed-projection boundary packets realize the open normalization frontier as finite closed-term, candidate-evidence, router-selection, endpoint-seal, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §4284: MetaCIC proof-search frontiers are realized as finite candidate-evidence, bounded-normal endpoint, discharge-socket, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7167: Finite proof-trace sieves realize the MetaCIC proof-search frontier as finite candidate, endpoint, discharge, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §5143: The normal-endpoint cut is realized as a finite packet separating candidate evidence, finished-normal endpoint, obstruction row, and discharge socket.

Concretized in §7180: Candidate-normalization discharge packets realize the MetaCIC normalization open problem as finite candidate-evidence, endpoint-seal, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §8635: MetaCIC residual normalization obstruction packets realize the arbitrary closed-term boundary as finite closed-term, candidate-request, endpoint, residual-frontier, obstruction, TasteGate-obligation, audit-handoff, transport, replay, provenance, and NameCert rows.

9386.5 Decidability remaining questions

The decidability map in chapter 9379 has a positive typing layer and a narrower conversion layer. Structural term equality, same-term checking, and option-valued type inference are sound and complete for the represented relations at the interface stated by the checker theorems.

The remaining decidability questions sit around closedness and beta conversion. The submodule does not export a theorem named `Decidable (ClosedAt n t)`. It also does not export an unrestricted closed-term beta-conversion decider. The bounded normal-equality checker is sound for equality of bounded normalizer outputs and for finished closed normal forms when the finished-normalisation witnesses are supplied.

Thus the decidability frontier is the endpoint bridge: how should bounded normal computation be connected to finished normal forms for the class of closed typed terms under discussion? A global conversion decider would require that bridge explicitly. Gallery success and checker completeness for typing do not supply it by themselves.

Concretized in §3439: Bounded normal conversion bridges realize the finished-normal endpoint needed by the MetaCIC decidability frontier.

Concretized in §4113: MetaCIC conversion endpoint seals realize the bounded normal-computation endpoint as finite finished-normal NameCert data.

Concretized in §6069: MetaCIC typed-conversion residual frontiers realize the decidability endpoint bridge as typed-window, residual-frontier, bounded-normal comparison, finished-normal witness, refusal, endpoint-obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6890: Closed typing fragments realize the MetaCIC decidability frontier as finite closed-term, type-checker, endpoint, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §7645: ClosedAt decision refusal boundaries are realized as finite checker, bounded-normal, refusal, transport, replay, provenance, and NameCert rows.

9386.6 TypedExamples coverage questions

The typed-examples map in chapter 9382 supplies a worked gallery: Church booleans, naturals, pairs, options, lists, sums, Leibniz equality, dependent identity, polymorphic identity, applications, beta-redexes, constants, Pi forms, projections, and sort-indexed context examples. These examples are Lean-side witnesses for concrete represented terms.

The open question is coverage, not existence. The gallery demonstrates many typical dependent, polymorphic, Church-encoded, and application-shaped terms, but it does not claim exhaustive coverage of all dependent-program forms or all expected algebraic laws of every Church encoding inside MetaCIC. The examples therefore remain a regression and audit surface rather than a general metatheorem.

The practical frontier is to decide which example families should be treated as representative enough for the paper's public narrative. Dependent codomains, polymorphic instantiation, eliminator computation, nested binder movement, and conversion-sensitive examples are the highest-value families because they touch the same boundaries as subject reduction, normalisation, and decidable conversion.

Concretized in §6650: Dependent eliminator coverage packets realize the typed-examples coverage frontier as finite typed-window, eliminator-family, conversion-boundary, gallery-status, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §6883: Polymorphic-instantiation coverage packets realize the typed-example frontier as finite universe-instantiation, typed-window, conversion-boundary, obstruction, transport, replay, provenance, and NameCert rows.

9386.7 Cross-cutting frontier questions

9386.7.1 Dependent-codomain barrier

Subject reduction is not unconditionally available in the dependent-codomain case. The independent-codomain application case is checked, while the dependent case has a counterexample in the present substrate. This is the obstruction that prevents the closed-normal consistency result from automatically becoming full closed consistency.

There are three coherent routes through the barrier. One can change the substrate so that a context-reduction lemma is available. One can weaken the subject-reduction statement so that it accepts the typing reorganisation forced by dependent codomains. One can also expose dependent-codomain inversion as a setup field or axiom-set-shaped interface rather than as an internal theorem. Each route changes the public meaning of the substrate, and none is taken here.

9386.7.2 Bundle discharge route

The discharge-bundle route is conceptually clean. A value of SubjectReductionDischargeBundle packages the four subject-reduction inputs, and the corresponding wrapper lifts the parameterised theorem to an unconditional theorem relative to that bundle. This makes the discharge interface first-class without spreading the four hypotheses through every downstream citation.

The same route also states the obstruction sharply. On the present closed-term substrate, inhabiting the bundle at the required strength is incompatible with the documented obstruction. The bundle is therefore a marker for a paper-side discharge path or for a substrate extension, not evidence that the closed-term substrate already contains the missing proof.

9386.7.3 Normal-form-only consistency

chapter 9375 establishes the positive closed-normal node. A closed proof term that is already beta-strong cannot inhabit the internal false encoding. This result is unconditional at its stated input shape and does not depend on discharging subject reduction.

The open judgement is presentational and strategic. The paper can treat normal-form consistency as a first-class BEDC deliverable, because it is a real closed theorem at the normal-form boundary. Doing so would avoid overstating full closed consistency while still making the checked positive result visible as a central endpoint.

9386.7.4 Substrate extension for dependent-codomain closure

The strongest design decision is whether to extend the closed-CIC substrate so that dependent-codomain reduction can be closed internally. Such an extension could make the conventional subject-reduction route possible, but it would also change the substrate interface and the audit obligations that downstream chapters must respect.

The conservative alternative is to keep the present substrate and report the closed-normal theorem as the main internal result. The expansive alternative is to add the exact context-reduction or setup-class field needed by the dependent-codomain cases, then re-audit subject reduction, confluence, normalisation, decidability, and examples against that enlarged interface.

9386.8 Operational priorities

The apophatic naming discipline of chapter 9400 is binding here. The paper should not pretend that the obligations above are solved. They are named open problems with exact interfaces.

The reader-facing priority order is:

1. Treat closed-normal consistency as a first-class deliverable. It is the strongest presently assembled consistency node and it does not require subject-reduction discharge.
2. Explore a setup-class field route for dependent-codomain inversion. This keeps the additional input explicit and avoids converting the obstruction into an untracked theorem claim.
3. Keep the subject-reduction obstruction documented. It should not be hidden behind a proof placeholder, an informal appeal to CIC intuition, or an overbroad citation of the parameterised theorem.

This priority order also constrains citations. The paper may cite the checked normal-form result, the parameterised subject-reduction theorem, the shape-limited confluence fragments, the candidate-mediated normalisation projections, the typing checker, the bounded normal-equality facts, and the typed-example gallery. It may not cite any of those as full closed consistency, arbitrary closed subject reduction, arbitrary closed Church–Rosser, global closed-term strong normalisation, or unrestricted beta-conversion decidability.

9386.9 Boundary

This chapter is an inventory, not a discharge. It consolidates the open MetaCIC work exposed by the existing audit maps and the consistency assembly. It does not prove any of the four subject-reduction obligations. It does not prove residual substitution compatibility for full confluence. It does not erase candidate evidence from normalisation, finished-normalisation evidence from bounded conversion, or gallery scope from typed examples.

The boundary is therefore simple: the chapter names the remaining problems and the available design routes, while leaving the proof state exactly where the Lean and paper audit maps place it.

Concretized in §763: finite ledger carrier for the MetaCIC open-obligation boundary

Concretized in §762: The MetaCIC open-problem inventory is realized as a finite frontier ledger over subject-reduction, confluence, normalization, decidability, typed-example, and consistency-boundary rows.

Concretized in §780: MetaCIC open problems are realized as a finite NameCert ledger for remaining subject-reduction, confluence, normalization, substitution, decidability, typed-example, and blocker rows.

Concretized in §781: The dependent-codomain barrier is realized as a finite NameCert packet for codomain readback, discharge handoff, and obstruction-boundary rows.

Concretized in §892: The normalization frontier is realized as a finite NameCert packet separating candidate evidence, finished-normal evidence, closed-normal use, and subject-reduction obstruction rows.

Concretized in §1008: The MetaCIC beta-redex frontier is realized as a finite NameCert packet for discharge-row coverage, beta-boundary routing, and obstruction non-escape.

Concretized in §3434: The decidable beta frontier realizes bounded MetaCIC beta-audit windows as finite NameCert data.

Concretized in §1100: The consolidated open-problem surface is realized as a finite frontier-witness NameCert packet.

Concretized in §2274: Candidate-mediated strong-normalization and confluence-boundary rows are realized as one finite MetaCIC NameCert packet.

Concretized in §3626: The dependent-codomain stability boundary is realized as a finite MetaCIC NameCert packet.

Proposition 9386.1 (Lambda-domain discharge row is mandatory)**.** *Consider a MetaCIC subject-reduction discharge route through the explicit bundle interface* SubjectReductionDischargeBundle *of chapter 9374. If that route is used to make the parameterised subject-reduction theorem available for the represented reduction relation, then its finite discharge packet contains a distinct row LD satisfying* LamDomainSubjectReduction*. This row is the lambda-domain congruence obligation of Definition 9377.3; it cannot be supplied by the beta-substitution row, the application-argument row, the Pi-domain row, transport, continuation replay, package provenance, or local naming.*

Proof. The four-obligation audit of section 9377.3 separates the reduction shapes consumed by the parameterised subject-reduction theorem. The lambda-domain shape is named by Definition 9377.3: if $\lambda d.\, b$ has type A in Γ and d steps to d', the target lambda $\lambda d'.\, b$ must still have type A in the same context. This is a binder-annotation preservation requirement, not a beta-redex substitution requirement and not an application-codomain stability requirement.

Read the explicit bundle route from subsection 9374.2.1. The bundle has four discharge fields, and one of them is the lambda-domain field. Therefore any use of the bundle route carries the lambda-domain row as a named component of the finite packet.

The finite ledger form gives the same conclusion at the BEDC packet surface. In Definition 720.1 an accepted SubjectReductionDischargeLedger$^{\uparrow}$ carrier is displayed as

$$L = (B, A, LD, PD, R, H, C, P, N),$$

where LD is the lambda-domain subject-reduction row and R is the route that consumes exactly B, A, LD, PD. The handoff theorem Definition 720.3 confines subject-reduction consumption to those four obligation rows through R, and the non-escape theorem Definition 720.4 confines downstream reads to the displayed packet. Hence H, C, P, N can transport, replay, authenticate, and name the route, but none of them can create or replace the missing lambda-domain preservation row.

Consequently, whenever the MetaCIC subject-reduction discharge interface is used, LamDomainSubjectReducti is a required finite discharge obligation row. Supplying the other rows without LD leaves the discharge interface incomplete for the lambda-domain congruence case. □

Proposition 9386.2 (Application-argument stability is a required discharge row)**.** *Let a MetaCIC subject-reduction consumer use the discharge interface through a* SubjectReductionDischargeBundle *as described in chapter 9374. Suppose the consumer reads a finite discharge socket*

$$O = (B, A, L, P, H, C, Q, N)$$

as in Definition 719.1. If the consumed reduction branch is application-argument congruence, then the required discharge input is exactly the row A matching AppArgTypeStable *of Definition 9377.2. It cannot be replaced by the beta row B, the lambda-domain row L, the Pi-domain row P, the independent-codomain application theorem, or the closed-obstruction row recorded in Q.*

Proof. Read the four-obligation interface of section 9377.3. The application-argument congruence branch is governed there by AppArgTypeStable, while BetaSubstitutionPreservation, LamDomainSubjectReductio and PiDomainSubjectReduction govern the beta, lambda-domain, and Pi-domain branches. Thus the branch classifier already separates the application-argument input from the other three discharge rows.

Now read the socket carrier. In

$$O = (B, A, L, P, H, C, Q, N),$$

row A is the application-argument stability row, and the remaining rows have distinct roles: B is beta substitution preservation, L is lambda-domain preservation, P is Pi-domain preservation, H transports displayed rows, C replays displayed routes, Q records provenance together with the parameterised subject-reduction row and the closed-obstruction boundary, and N names the local certificate. Therefore a socket read for the application-argument branch has only one matching discharge coordinate, namely A.

The bundle route does not change that payload. By chapter 9374, a SubjectReductionDischargeBundle is the four-field packet for the same four hypotheses. Therefore route consumption cannot remove the application-argument row or identify it with one of the other three fields.

Finally separate the positive narrow case from the full discharge interface. The independent-codomain application theorem applies only where the codomain readback is already stable because the codomain is independent of the argument. The present proposition concerns the subject-reduction discharge interface used for the application-argument congruence branch in general, including the dependent-codomain boundary identified in section 9377.7. At that boundary, the codomain can observe the reducing argument, so the interface must carry the explicit AppArgTypeStable row. The obstruction row in Q records the absence of a closed theorem and is not one of the four discharge inputs. Thus A is an explicit finite discharge obligation whenever the interface is consumed for the application-argument branch. □

Proposition 9386.3 (Beta-substitution preservation is a mandatory discharge row)**.** *Any use of the MetaCIC subject-reduction discharge interface must expose* BetaSubstitutionPreservation *as a finite discharge obligation. More precisely, a subject-reduction call that follows the explicit* SubjectReductionDischargeBundle *route has a distinguished beta row whose content is exactly the obligation of Definition 9377.1. That row is not supplied by* AppArgTypeStable*,* LamDomainSubjectReduction*,* PiDomainSubjectReduction*, the dependent-codomain obstruction theorem, or the closed-normal consistency assembly.*

Proof. Read the four-obligation interface from section 9377.3. The first field is BetaSubstitutionPreservation, and Definition 9377.1 identifies its content: body typing under the shifted-domain context and argument typing at the unshifted domain must yield typing of the substituted body at the substituted codomain. This is the beta-redex preservation case, not an application-argument stability case and not a binder-domain congruence case.

Now read the public discharge route. In the bundle route of section 9374.2, SubjectReductionDischargeBundle is the four-field packet consumed by the subject-reduction wrapper. Since the beta field is one of those four fields, constructing the packet requires a beta-substitution preservation row before the wrapper can be invoked.

The remaining three rows have disjoint jobs. AppArgTypeStable stabilizes codomain readback when reduction enters an application argument, while LamDomainSubjectReduction and PiDomainSubjectReduction preserve displayed binder annotations when reduction enters lambda

and Pi domains. None of those statements contains the beta-redex premise-conclusion shape from Definition 9377.1. They therefore cannot fill the beta field of the bundle packet.

Finally, the obstruction material in chapter 9377 has the opposite polarity: it blocks replacing the four-input interface by a blanket closed subject-reduction theorem for the present substrate. It does not discharge any field. Hence a consumer that uses the MetaCIC subject-reduction discharge interface must carry BetaSubstitutionPreservation explicitly as the finite beta discharge row, and the interface remains unavailable as a closed subject-reduction theorem until that row is supplied together with the other three discharge rows. □

Proposition 9386.4 (Pi-domain discharge row is mandatory)**.** *Consider a MetaCIC subject-reduction discharge route through the explicit bundle interface* SubjectReductionDischargeBundle *of chapter 9374. If that route is used to make the parameterised subject-reduction theorem available for the represented reduction relation, then its finite discharge packet contains a distinct Pi-domain row satisfying* PiDomainSubjectReduction*. This row is the Pi-domain congruence obligation of Definition 9377.4; it cannot be supplied by the beta-substitution row, the application-argument row, the lambda-domain row, transport, continuation replay, package provenance, or local naming.*

Proof. Read the four-obligation audit of section 9377.3. The Pi-domain branch is named by Definition 9377.4: if $\Pi d.\, c$ has type A in Γ and the displayed domain d steps to d', the target Pi term $\Pi d'.\, c$ must still have type A in the same context. This is the Pi-binder domain preservation case. It is not the beta-redex substitution case, not the application-argument codomain-readback case, and not the lambda-domain binder-annotation case.

The public route preserves that separation. In the explicit route of subsection 9374.2.1, SubjectReductionDischargeBundle is the four-field packet for the four subject-reduction hypotheses, including the Pi-domain field. Thus route consumption cannot remove the Pi-domain coordinate or identify it with any other coordinate.

The socket and ledger surfaces display the same finite row. In Definition 719.1 an accepted socket is

$$O = (B, A, L, P, H, C, Q, N),$$

where P is the Pi-domain subject-reduction row. The four-row exactness theorem Definition 719.8 says that the subject-reduction ledger consumes exactly B, A, L, P and may not replace one of those four rows, add a fifth discharge row, or treat the obstruction boundary as discharge evidence. Hence a socket consumer for the Pi-domain branch has only one matching discharge coordinate, namely P.

The ledger notation records the same coordinate as PD. By Definition 720.1 an accepted ledger is

$$L = (B, A, LD, PD, R, H, C, P, N),$$

where PD matches PiDomainSubjectReduction and R consumes exactly B, A, LD, PD. The handoff theorem Definition 720.3 confines subject-reduction consumption to those four obligation rows through R, and the non-escape theorem Definition 720.4 confines downstream reads to the displayed packet. Therefore H, C, P, N can transport, replay, authenticate, and name the finite route, but none of them can create or replace the missing Pi-domain preservation row.

Consequently, whenever the MetaCIC subject-reduction discharge interface is consumed for the Pi-domain congruence branch, PiDomainSubjectReduction is a required finite discharge obligation row. Supplying the beta, application-argument, and lambda-domain rows without the Pi-domain row leaves the discharge interface incomplete for that branch. □

Concretized in §2307: The MetaCIC open frontier is realized as a finite dependency matrix over discharge rows, obstruction rows, and audit-map consumers.

Concretized in §3352: The Pi-domain subject-reduction obligation is realized as a finite closed MetaCIC NameCert packet.

Concretized in §3381: The lambda-domain subject-reduction obligation is realized as a finite closed MetaCIC NameCert packet.

Concretized in §3460: Proof-obligation gap packets realize the MetaCIC subject-reduction obstruction as finite named obligation, refusal, handoff, transport, replay, provenance, and NameCert rows.

Concretized in §3567: The candidate-mediated strong-normalization frontier is realized as finite MetaCIC NameCert data.

Concretized in §4292: The MetaCIC normal-endpoint budget is realized as finite candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction, transport, replay, provenance, NameCert, and TasteGate rows.

Concretized in §6111: The four subject-reduction discharge rows are realized as a finite socket-preserving NameCert boundary.

Concretized in §6577: Candidate-normalization witnesses realize the MetaCIC normalization frontier as finite candidate-evidence, endpoint, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6617: Typed normal-form scheduler packets realize the normalization open problem as finite candidate-gate, scheduler-budget, endpoint-seal, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §6922: Subject-reduction discharge matrices realize the four simultaneous MetaCIC discharge rows as finite NameCert data.

Concretized in §8837: MetaCIC subject-reduction obstruction witnesses realize the simultaneous four-row obstruction surface as finite beta-substitution, application-argument, lambda-domain, Pi-domain, matrix, proof-obligation-gap, obstruction, socket, transport, replay, provenance, and NameCert rows.

Concretized in §8051: The residual-confluence budget packet realizes the MetaCIC confluence frontier as finite typed-window, residual-pair, local-join, bounded-checker, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §8456: Residual normal-form budget routers realize the MetaCIC residual and normalization frontier as finite typed-window, residual-budget, scheduler-budget, endpoint-seal, candidate-handoff, obstruction, transport, replay, provenance, and NameCert rows.

Concretized in §3437: finite critical-pair ledger for MetaCIC audit-map concretization

Concretized in §3519: MetaCIC redex-choice frontier packets realize candidate-mediated normalization and confluence boundaries as finite BEDC rows.

Concretized in §4238: MetaCIC parallel-diamond frontier packets realize the bounded residual and critical-pair route for candidate-mediated confluence inspection.

Concretized in §7797: MetaCIC residual-substitution socket packets connect normalization and confluence audit surfaces through finite residual source, substitution-window, candidate-boundary, and confluence socket rows.

Concretized in §7893: MetaCIC residual-diamond readback packets realize candidate-mediated residual, local witness, and L10 source-ledger routes as finite NameCert rows.

Concretized in §8369: MetaCIC audit synthesis routers realize the subject-reduction, confluence, normalization, substitution, decidability, blocker, and open-input rows as one finite NameCert packet.

Chapter 9387

Three Axiom Families, One Meta-Closure Taxonomy

This chapter gives a taxonomy for three axiom families that BEDC refuses to internalise: the axiom of choice, quotient soundness, and propositional extensionality. The taxonomy normalizes them as three shapes of unwitnessed closure rather than proving a metatheorem that the host axioms are equivalent to three ontic collapses. Choice is read as term-level acceptance of an unwitnessed selection function; quotient soundness is read through a quotient-facing classifier-to-equality shape; propositional extensionality is read as proposition-level acceptance of equality from bidirectional implication. These readings are interpretive specifications for the axiom-purity gate, not Lean-checked equivalences with Classical.choice, core Quot.sound, or propext. This chapter (i) records the three schematic readings as "accept an unwitnessed $\exists$ as argument input", (ii) states the taxonomy that places them under one meta-closure heading, (iii) pairs each reading with its BEDC effective replacement – countable and dependent choice plus the finite case via explicit recursion for choice, the typed naming certificate framework of chapter 9 with classifier-respecting transport for quotient soundness, and explicit hsame / psame transport for propositional extensionality – and (iv) records that the project's axiom-purity gate is therefore a refusal of hidden closure rather than a rejection of the application content the three axiom families serve.

9387.1 The three axiom families as taxonomy entries

Definition 9387.1 (Meta-closure operation)**.** *A* meta-closure operation *at a syntactic stratum $S \in \{term, type, proposition\}$ is a derivation step that, given a positive existential statement at stratum S together with no constructor witness, supplies a value of S as if a witness had been given. The accepted value plays the role of an oracle: it appears as input to subsequent constructions inside the kernel without being traceable to a closed inductive constructor.*

Example 9387.2 (Choice as term-level meta closure)**.** *The axiom of choice asserts: from $\forall i \in I, \exists x_i \in A_i$, conclude $\exists f : I \to \bigsqcup A_i$ with $f\, i \in A_i$. Read at the term stratum, the conclusion is a term f of function type whose body is not constructed from the existential witnesses. The axiom supplies f as an oracle; the body is unwitnessed and therefore inaccessible to kernel-level computation. In this taxonomy it is the term-level entry for Definition 9387.1.*

Example 9387.3 (Quotient soundness as type-level meta closure)**.** *The quotient-soundness reading used here is a quotient-facing specification shape: from a relation R on a type A, a consumer*

receives equality or representative behavior at the class level without displaying the classifier transport that produced it. This is not a formal account of Lean core Quot.sound*, and it is not a proof that Lean quotients supply canonical representative selectors. It is the type-level entry in the taxonomy of Definition 9387.1.*

Example 9387.4 (Propositional extensionality as proposition-level meta closure)**.** propext *asserts: from $P \Leftrightarrow Q$, conclude $P = Q$ at the proposition level. The conclusion identifies two propositions whose only relation is bidirectional implication; reading the identification at the proposition stratum,* propext *supplies an unwitnessed equality of propositions. In this taxonomy it is the proposition-level entry for Definition 9387.1.*

9387.2 The unification taxonomy

Principle 9387.5 (The three axiom families share one taxonomy)**.** *Choice (Definition 9387.2), the quotient-facing soundness shape (Definition 9387.3), and* propext *(Definition 9387.4) are classified under the meta-closure schema of Definition 9387.1. The classification is by syntactic stratum: term, type or classifier, and proposition. It licenses the refusal ledger's organization; it does not assert that a Lean metatheory has proved the three host constants equivalent to three BEDC collapse modes.*

Explanation. The three examples display a common specification pattern: an unavailable witness, equality token, or class-level bridge is treated as ordinary input to later construction. The differences across choice, the quotient-facing soundness shape, and propext are the strata at which that pattern is requested. This is enough to sort the refusal ledger stratum by stratum. The argument is not a metatheoretic equivalence proof for Lean's forbidden constants. □

Concretized in §515: Unwitnessed-existence closure is realized as a finite NameCert packet over term, type, and proposition strata, with explicit replacement and axiom-purity ledger rows.

Concretized in §566: Propext transport boundaries realize the proposition-stratum replacement as finite direction, transport, replay, and non-escape NameCert rows.

Concretized in §568: Effective replacement ledgers realize choice witnesses, quotient transport, propext transport, and axiom-purity refusal as one finite NameCert packet.

Concretized in §634: Quotient-soundness refusal is realized as a finite NameCert packet replacing representative selection by classifier-respecting psame transport and audit-visible non-escape.

Concretized in §1012: Term-stratum choice closure is realized as a finite witness-map and oracle-refusal NameCert packet.

Concretized in §6939: The three-axiom closure route is realized as finite closure, generator-induction, refusal, transport, replay, provenance, and NameCert rows.

9387.3 Effective replacements inside BEDC

Definition 9387.6 (Effective replacement for choice)**.** *The BEDC* effective replacement for choice *is the package consisting of: countable choice on $\mathbb{N}$-indexed families with explicit witness functions; dependent choice along a* Cont*-chain with explicit step functions; and finite choice via primitive recursion on a closed inductive list. Each replacement supplies a witness function whose body is constructed from the kernel-level data of the assumption.*

Definition 9387.7 (Effective replacement for `Quot`-soundness)**.** *The BEDC* effective replacement for `Quot`-soundness *is the typed naming-certificate framework of chapter 9. Two carrier elements identified by a classifier are not collapsed at the type level; instead, the certificate exposes a* psame-*transport that lets every classifier-respecting operation pass through the identification without selecting a canonical representative.*

Definition 9387.8 (Effective replacement for propext)**.** *The BEDC* effective replacement for propext *is the explicit* hsame *and* psame *transport on every certificate. Two propositions P, Q that are bidirectionally implied are not identified; instead, every kernel-level construction that consumes P receives a Q-version via the bidirectional implication, and the construction is parameterised over the chosen direction.*

Theorem 9387.9 (Replacements suffice for BEDC's certified content)**.** *Every theorem in the manuscript whose statement is supported by a Lean-checked-level marker is provable using only the replacements of Definition 9387.6, Definition 9387.7, and Definition 9387.8 in place of the three classical axioms.*

Sketch. The replacements are themselves kernel-level constructions on closed inductive carriers. A traversal of the manuscript's certified content maps each invocation of choice to countable, dependent, or finite choice; each invocation of `Quot`-soundness to certificate-level psame-transport; and each invocation of propext to a hsame-mediated rewrite. The traversal terminates because the certified content is finite and the replacement is uniform across applications of each axiom. □

9387.4 The axiom-purity gate as refusal of meta closure

Remark 9387.10 (The gate's structural meaning)**.** *The project's axiom-purity gate (`axiom-purity -strict` in `lean4/scripts/bedc_ci.py`) refuses every kernel-level theorem whose transitive dependencies include* Classical.choice*,* Quot.sound*, or* propext*. Read through the taxonomy of Definition 9387.5, the gate is organized as a refusal at three strata: term, type or classifier, and proposition. The gate is therefore not a rejection of the application content the three axiom families serve; it is a refusal of unledgered shortcuts that supply that content without constructive witnesses or displayed transport.*

9387.5 Assumed bridges and derived bridges

The taxonomy above should not be read as a claim that the classical axioms are syntactically ill-typed in their host theories. They are well-typed constants there. The BEDC diagnosis is sharper: each constant inhabits a bridge type whose source and target belong to different audit strata, but the inhabitant is assumed rather than derived from displayed constructor, classifier, ledger, transport, or NameCert data.

Definition 9387.11 (Proposition existence and data witness)**.** *For $A : \mathsf{Type}$ and $P : A \to \mathsf{Prop}$, the proposition-level existence claim is*

$$\mathsf{ExistsProp}(A, P) := \exists x : A,\ P(x),$$

while the data-level witness package is

$$\mathsf{SigmaData}(A, P) := \sum_{x:A} P(x).$$

The first expression asserts existence as a proposition. The second displays a term together with the proof carried by that term.

Definition 9387.12 (Unledgered cross-stratum bridge)**.** *A map $c : A \to B$ is an* unledgered cross-stratum bridge *when A and B are consumed at different audit strata, c is later used as an ordinary kernel input, and no constructor, definition, classifier, transport row, ledger, setup field, or* NameCert *row derives c from displayed data.*

Principle 9387.13 (The three axiom families are read as assumed bridges)**.** *In their host theories,* Classical.choice*,* Quot.sound*, and* propext *are not type errors. In BEDC's audit reading they are treated as assumed bridge shapes:*

$$\begin{aligned} \mathsf{Classical.choice} &: \mathsf{ExistsProp}(A, P) \to \mathsf{SigmaData}(A, P), \\ \mathsf{Quot.sound} &: \ x \sim y \to [x] = [y], \\ \mathsf{propext} &: \ (P \Leftrightarrow Q) \to (P = Q). \end{aligned}$$

Each bridge turns source-side relation or existence data into a kernel-consumable object without displaying the derivation of that object.

Explanation. The three displayed types are meaningful in the host environment that admits the corresponding constants. Thus the issue is not syntactic well-typedness. For choice, the bridge turns a proposition-level existential into a data-level dependent pair. For quotient soundness, it turns a classifier witness into equality at a quotient-facing point. For propositional extensionality, it turns bidirectional implication into equality of propositions. In each case, if the constant is removed, the bridge term is not supplied by the source datum alone. It is therefore an assumed bridge from the BEDC viewpoint, not a derived bridge. □

Definition 9387.14 (Type-honest fragment)**.** *A BEDC fragment is* type-honest *when it refuses unledgered bridges of the following forms:*

$$\mathsf{ExistsProp}(A, P) \to \mathsf{SigmaData}(A, P), \qquad x \sim y \to [x] = [y], \qquad (P \Leftrightarrow Q) \to (P = Q).$$

If the fragment also uses apophatic far-end language, type honesty additionally refuses any socket-to-element bridge of the form $\mathsf{T} : X$ or $\mathsf{T} \in X$.

Principle 9387.15 (Axiom-purity as type honesty)**.** *The axiom-purity gate for* Classical.choice*,* Quot.sound*, and* propext *is the mechanical face of the type-honesty taxonomy in Definition 9387.14. It rejects assumed bridges where the manuscript requires derived bridges.*

Explanation. By Definition 9387.13, the three forbidden constants inhabit the three unledgered bridge shapes. By Definition 9387.14, a type-honest fragment does not consume those bridge shapes unless a displayed replacement is provided. The gate therefore checks the same discipline at dependency level: an implementation using one of the constants has exposed an assumed bridge in its proof ancestry. □

Remark 9387.16 (`#print axioms` reports an implementation footprint)**.** *The command* `#print axioms` *reports the transitive dependencies of a particular Lean declaration. If it lists* Classical.choice*,* Quot.sound*, or* propext*, the conclusion is that this implementation consumed those bridge assumptions. The report is not a lower bound proving that every possible construction of the same mathematical object must consume them.*

Remark 9387.17 (Sibling instance: halting and reasoning by contradiction)**.** *The chapter's taxonomy (Definition 9387.5) sits in a family with the halting capstone (chapter 9368) and the reasoning-by-contradiction capstone (chapter 9458): all three identify a different audit face of meta-closure refusal. Halting names the operational-semantics face; reasoning by contradiction names the proof-theoretic face; the present chapter names the foundational-axiom face. Together they constitute the discipline side of the two-loops architecture (Definition 9461.15).*

Definition 9387.18 (Term-stratum choice closure)**.** *Fix an index carrier I, an ambient term carrier U, and a family of positive fiber predicates $A_i \subseteq U$. The* term-stratum choice closure schema *is the following instance of Definition 9387.1: from an accepted family of positive existentials*

$$H : \forall i \in I,\ \exists x : U,\ x \in A_i$$

with no displayed constructor-level selection of such terms, introduce a term $o_H : I \to U$ and membership evidence $o_H(i) \in A_i$ for each i. The term $o_H(i)$ is then available as an oracle input to later term-level constructions.

Principle 9387.19 (Choice as term-stratum meta-closure reading)**.** *For arbitrary indexed families A_i, the term-stratum choice closure schema of Definition 9387.18 is the specification shape used here to read the axiom-of-choice principle of Definition 9387.2:*

$$(\forall i \in I,\ \exists x_i,\ x_i \in A_i) \Longrightarrow \exists f : I \to \bigsqcup_{i \in I} A_i,\ \forall i \in I,\ f(i) \in A_i.$$

Here $f(i) \in A_i$ abbreviates that the value of f lies over the i-fiber of the displayed coproduct.

Explanation. Assume the term-stratum closure schema. Given $H : \forall i \in I, \exists x_i, x_i \in A_i$, the schema supplies an oracle term $o_H : I \to U$ with evidence $o_H(i) \in A_i$ for every i. Define $f(i) = (i, o_H(i))$ as an element of the displayed coproduct $\bigsqcup_{i \in I} A_i$. The fiber evidence supplied with o_H proves $f(i) \in A_i$ for each i, so f has the displayed choice-function shape.

Conversely, if the displayed axiom-of-choice principle is admitted, then for a term-stratum closure datum $H : \forall i \in I, \exists x_i, x_i \in A_i$ it produces a function $f : I \to \bigsqcup_{i \in I} A_i$ with $f(i) \in A_i$ for every i. Let $o_H(i)$ be the component of $f(i)$ in the i-fiber. The accompanying fiber proof is the evidence required to use $o_H(i)$ as the accepted value of the existential at i. Thus the schema records the same consumer-facing shape as the choice principle. The point is the audit shape, not an independent Lean theorem equating all formulations of choice. □

Definition 9387.20 (Quotient representative package)**.** *Let A be a type and let R be an equivalence relation on A. A* quotient representative package *for (A, R) is a triple (Q, π, ρ) with*

$$Q : \mathsf{Type}, \qquad \pi : A \to Q, \qquad \rho : Q \to A,$$

such that

$$\forall a\, b : A,\ R(a, b) \Longrightarrow \pi(a) = \pi(b),$$
$$\forall u : Q,\ \pi(\rho(u)) = u,$$

and

$$\forall a : A,\ R(\rho(\pi(a)), a).$$

Write $\mathsf{QRep}_{A,R}(Q, \pi, \rho)$ for this conjunction. The second displayed condition says that ρ is a section of π; the third says that the selected representative of the class of a remains in the R-class of a.

Lemma 9387.21 (Quotient representative packages are exact)**.** *Let R be an equivalence relation on A. If $\mathsf{QRep}_{A,R}(Q, \pi, \rho)$ holds, then for all $a, b : A$,*

$$\pi(a) = \pi(b) \iff R(a, b).$$

Proof. The reverse implication is the soundness field of $\mathsf{QRep}_{A,R}(Q, \pi, \rho)$. For the forward implication, assume $e : \pi(a) = \pi(b)$. Representative adequacy gives $R(\rho(\pi(a)), a)$ and $R(\rho(\pi(b)), b)$. Transport the second witness along congruence of ρ applied to e, obtaining $R(\rho(\pi(a)), b)$. Symmetry of the first witness gives $R(a, \rho(\pi(a)))$, and transitivity with the transported second witness gives $R(a, b)$. □

Definition 9387.22 (Type-stratum quotient closure)**.** *For a type A and an equivalence relation R on A, the* type-stratum quotient closure schema *is the following instance of Definition 9387.1: from the positive existential*

$$\exists Q : \mathsf{Type},\ \exists \pi : A \to Q,\ \exists \rho : Q \to A,\ \mathsf{QRep}_{A,R}(Q, \pi, \rho)$$

with no displayed constructor-level witness for Q, π, or ρ, introduce such a triple as accepted type-level data. The accepted type is read as A/R, and the accepted selector is available as an oracle input to later type-level constructions.

Principle 9387.23 (Quotient-facing soundness as type-stratum reading)**.** *At the schema level, the type-stratum quotient closure schema of Definition 9387.22 is the paper's quotient-facing proxy for the* `Quot`*-soundness refusal of Definition 9387.3. It packages the consumer-facing data as a type Q, a projection, and a representative section. This proxy is a specification device for BEDC's refusal ledger, not a statement that Lean core* Quot.sound *supplies such a selector or that the proxy has been proved equivalent to Lean's quotient implementation.*

Explanation. Assume the type-stratum quotient closure schema. Given A and R, instantiate the positive existential in Definition 9387.22. The schema supplies Q, π, and ρ satisfying $\mathsf{QRep}_{A,R}(Q, \pi, \rho)$. By Definition 9387.21, equality in Q tracks the classifier R on representatives inside this proxy package, so Q may be read as quotient-facing data for the local schema. The section and representative-adequacy fields make ρ the selected representative row of the proxy package.

Conversely, if a host-side quotient-facing package supplies a type, projection, and representative section with the displayed properties, then it fills the existential datum accepted by Definition 9387.22. The two readings therefore align at the level of this paper schema. The BEDC replacement in Definition 9387.7 refuses the selector ρ and keeps classifier-respecting psame-transport instead. □

Definition 9387.24 (Proposition-stratum propext closure)**.** *For propositions $P, Q : \mathsf{Prop}$, a* proposition-stratum closure datum *is a bidirectional implication*

$$b_{P,Q} : (P \Rightarrow Q) \wedge (Q \Rightarrow P).$$

The proposition-stratum propext closure schema *is the following instance of Definition 9387.1: from such a datum, with no displayed constructor-level inhabitant of the identity proposition $P = Q$, accept the identity proposition as inhabited by supplying an equality token*

$$\varepsilon_{P,Q} : P = Q.$$

The token is then available as input to later proposition-stratum constructions by rewrite transport in every proposition-valued context $C : \mathsf{Prop} \to \mathsf{Prop}$.

Principle 9387.25 (Propext as proposition-stratum meta-closure reading)**.** *The proposition-stratum propext closure schema of Definition 9387.24 is the specification shape used here to read propositional extensionality:*

$$\forall P, Q : \mathsf{Prop},\ (P \Leftrightarrow Q) \Rightarrow P = Q.$$

In this reading, the accepted oracle value in Definition 9387.1 is the equality token $\varepsilon_{P,Q}$. This is a specification-level identification of shape, not a separate metatheoretic proof about every formulation of propositional extensionality.

Explanation. Assume the proposition-stratum propext closure schema. Let $P, Q : \mathsf{Prop}$ and let $h : P \Leftrightarrow Q$. The datum h is a pair of implications $P \Rightarrow Q$ and $Q \Rightarrow P$. Applying the schema to this datum supplies an equality token $\varepsilon_{P,Q} : P = Q$, which has the consumer-facing shape of propositional extensionality.

Conversely, if propositional extensionality is admitted, then a proposition-stratum closure datum $b_{P,Q} : (P \Rightarrow Q) \wedge (Q \Rightarrow P)$ can be sent to an equality $P = Q$. Equality elimination along that token supplies contextual rewrite behavior for proposition-valued contexts. Thus the schema records the same consumer-facing route as the extensionality principle. The claim remains a taxonomy of route shape. □

Definition 9387.26 (Primitive-recursive countable witness map)**.** *Let $A : \mathbb{N} \to \mathsf{Type}$ and let $w : \forall n : \mathbb{N}, A(n)$. Put*

$$\Sigma A := \bigsqcup_{k:\mathbb{N}} A(k).$$

The associated primitive-recursive countable witness map *is the function*

$$\mathsf{cw}_{A,w} : \mathbb{N} \to \Sigma A$$

defined by the primitive-recursion clauses

$$\mathsf{cw}_{A,w}(0) = (0, w(0)), \qquad \mathsf{cw}_{A,w}(n+1) = (n+1, w(n+1)).$$

Equivalently,

$$\mathsf{cw}_{A,w} = \mathsf{rec}_{\mathbb{N}}\big((0, w(0))\big)\big(\lambda n\, r.\, (n+1, w(n+1))\big),$$

where the step variable r is ignored because the premise supplies the witness in the next fiber.

Theorem 9387.27 (Countable choice is realized by primitive recursion)**.** *For every family $A : \mathbb{N} \to \mathsf{Type}$ and every displayed witness family $w : \forall n : \mathbb{N}, A(n)$, the map $\mathsf{cw}_{A,w}$ of Definition 9387.26 is a fiberwise choice function:*

$$\forall n : \mathbb{N},\ \mathsf{cw}_{A,w}(n) \in A(n).$$

Here membership uses the displayed-coproduct convention of Definition 9387.19: the value lies over the n-fiber, and its second component inhabits that fiber. The construction uses only the $\mathbb{N}$ recursor and dependent-pair introduction, so it is an instance of Definition 9387.6 and does not invoke the term-stratum choice closure of Definition 9387.18.

Proof. Induct on n. At 0, the first recursion clause gives

$$\mathsf{cw}_{A,w}(0) = (0, w(0)),$$

whose first projection is 0 and whose second projection is the displayed term $w(0) : A(0)$. Hence $\mathsf{cw}_{A,w}(0)$ lies in the 0-fiber.

For the successor case, the second recursion clause gives

$$\mathsf{cw}_{A,w}(n+1) = (n+1, w(n+1)).$$

Its first projection is $n+1$, and its second projection is the displayed term $w(n+1) : A(n+1)$. Thus $\mathsf{cw}_{A,w}(n+1)$ lies in the $(n+1)$-fiber. This proves the displayed universal statement. No step eliminates an unwitnessed existential: the only fiber element used at index n is the constructor-level term $w(n)$, so no oracle selector or Classical.choice enters the construction. □

Corollary 9387.28 (Witnessed countable choice for predicates)**.** *Let $B : \mathbb{N} \to \mathsf{Type}$ and let $P : \forall n : \mathbb{N}, B(n) \to \mathsf{Prop}$. If*

$$H : \forall n : \mathbb{N}, \bigsqcup_{x:B(n)} P(n, x)$$

is a constructor-level family of existential witnesses, then there are functions

$$a : \forall n : \mathbb{N}, B(n), \qquad p : \forall n : \mathbb{N}, P(n, a(n))$$

given by dependent-pair projection from $H(n)$. The displayed-pair map

$$n \mapsto (n, a(n)) : \mathbb{N} \to \bigsqcup_{n:\mathbb{N}} B(n)$$

is the primitive-recursive witness map of Definition 9387.26 applied to a, and $p(n)$ proves the chosen value satisfies P at every index.

Proof. For each n, the term $H(n)$ is a displayed pair. Define $a(n)$ to be its first projection and $p(n)$ to be its second projection. Then $a(n) : B(n)$ and $p(n) : P(n, a(n))$ by dependent-pair elimination. Applying Definition 9387.27 to the family B and witness function a gives the displayed-pair map $n \mapsto (n, a(n))$, and the proof component $p(n)$ supplies the predicate evidence for each chosen value. Projection from a displayed pair is kernel computation, not term-stratum meta closure. □

Definition 9387.29 (Certified psame-respecting operation)**.** *Let A and B be BEDC carrier layers whose public classifiers are written psame_A and psame_B, and whose carrier-classifier laws are licensed by naming certificates in the sense of Definition 9.6 and Definition 9.7. A map $\varphi : A \to B$ is a* certified psame-respecting operation *when its stability data contains a descent witness*

$$\tau_\varphi : \forall a, a' : A,\ \mathsf{psame}_A(a, a') \Longrightarrow \mathsf{psame}_B(\varphi(a), \varphi(a')).$$

This witness is the operation-level DescentCert *component of Definition 9.4. It is evidence about the classifiers, not an equality $a = a'$ and not a selector of a representative of the psame_A-class.*

Theorem 9387.30 (NameCert psame transport replaces quotient equality)**.** *Let A and B be BEDC carrier layers with certified classifiers psame_A and psame_B. Let $\varphi : A \to B$ be a certified* psame*-respecting operation with descent witness τ_φ as in Definition 9387.29. For every $a, a' : A$, a classifier witness*

$$s : \mathsf{psame}_A(a, a')$$

transports through φ to a target classifier witness

$$\mathsf{psame}_B(\varphi(a), \varphi(a')).$$

The derivation uses no type-level equality $a = a'$, no quotient representative type, and no representative-selecting section.

Proof. Unfold the certified psame-respecting operation datum. Its descent field is precisely

$$\tau_\varphi : \forall x, y : A,\ \mathsf{psame}_A(x, y) \Longrightarrow \mathsf{psame}_B(\varphi(x), \varphi(y)).$$

Apply this field to a, a', and the supplied classifier witness s. The resulting witness is

$$\tau_\varphi(a, a', s) : \mathsf{psame}_B(\varphi(a), \varphi(a')).$$

This proof consumes only the displayed classifier evidence and the certified descent field. A quotient-style argument would first turn s into an equality at a quotient type and then use congruence of φ over that equality. The naming-certificate argument keeps a and a' as distinct source terms and transports only the classifier judgment. Thus the quotient-soundness role identified in Definition 9387.7 is supplied by explicit psame-transport. □

Definition 9387.31 (Proof-input proposition construction)**.** *Let P : Prop and let X be a kernel target, where X may itself be a proposition or a certified carrier. A* proof-input proposition construction over P *is a construction whose access to P is mediated by a displayed proof argument:*

$$\Phi_P : P \to X.$$

An hsame-style proposition transport *between propositions P and Q is the bidirectional implication datum*

$$b_{P,Q} = (b^{\rightarrow}_{P,Q}, b^{\leftarrow}_{P,Q}) : (P \Rightarrow Q) \wedge (Q \Rightarrow P).$$

It is used as transport data between proof inputs, not as an equality token $P = Q$.

Theorem 9387.32 (hsame-style proposition transport bypasses propext)**.** *Let P, Q : Prop and let*

$$b_{P,Q} = (b^{\rightarrow}_{P,Q}, b^{\leftarrow}_{P,Q}) : (P \Rightarrow Q) \wedge (Q \Rightarrow P)$$

be the proposition-stratum transport datum of Definition 9387.24, read without the equality token supplied by propext*. For every proof-input proposition construction*

$$\Phi_P : P \to X,$$

there is a Q-parameterised construction

$$\Phi[P \rightsquigarrow Q] : Q \to X$$

given by

$$\Phi[P \rightsquigarrow Q](q) = \Phi_P\big(b^{\leftarrow}_{P,Q}(q)\big).$$

Dually, every proof-input construction $\Psi_Q : Q \to Y$ has a P-parameterised construction

$$\Psi[Q \rightsquigarrow P] : P \to Y, \qquad \Psi[Q \rightsquigarrow P](p) = \Psi_Q\big(b^{\rightarrow}_{P,Q}(p)\big).$$

Thus every later kernel step that consumes a displayed proof of the proposition can be rerun on the transported proposition through the chosen implication direction, using no inhabitant of $P = Q$ and no contextual rewrite over an arbitrary C : Prop → Prop.

Proof. Unpack the bidirectional transport datum into its two implication components. Given $q : Q$, the reverse component $b^{\leftarrow}_{P,Q} : Q \Rightarrow P$ produces a proof $b^{\leftarrow}_{P,Q}(q) : P$. The original construction Φ_P consumes exactly such a displayed proof, so applying it gives

$$\Phi_P(b^{\leftarrow}_{P,Q}(q)) : X.$$

This defines the displayed map $\Phi[P \rightsquigarrow Q] : Q \to X$. The dual construction is identical, using $b^{\rightarrow}_{P,Q} : P \Rightarrow Q$ before applying Ψ_Q.

No step asks for an equality proof $\varepsilon_{P,Q} : P = Q$. The proposition symbol is never rewritten inside an arbitrary context; the construction only composes ordinary proof functions. The stronger operation described in Definition 9387.24 would manufacture an equality token and then transport through every proposition-valued context. The construction above is the effective replacement promised by Definition 9387.8: it keeps P and Q distinct and moves only displayed proof inputs along the supplied bidirectional implication. □

Definition 9387.33 (Presented term-stratum witness family)**.** *Fix an index carrier I, an ambient term carrier U, and positive fiber predicates $A_i \subseteq U$. For each $i \in I$, let R_i be a carrier of constructor recipes and let $\epsilon_i : R_i \to U$ be the evaluation map from recipes to terms. A* presented term-stratum witness family *over $(A_i)_{i\in I}$ is data*

$$r : \forall i \in I,\ R_i, \qquad \rho : \forall i \in I,\ \epsilon_i(r(i)) \in A_i.$$

Its exposed choice output *is the pair $\Phi(r, \rho) = (f_r, \mu_r)$ defined by*

$$f_r : I \to \bigsqcup_{i\in I} A_i, \qquad f_r(i) = (i, \epsilon_i(r(i))),$$

with fiber evidence $\mu_r(i) = \rho(i)$. Thus the source presentation records the recipe binding $i \mapsto r(i)$ as constructor-level data, while the exposed output records only the evaluated fiber term and its membership evidence.

Proposition 9387.34 (Term-stratum meta closure drops per-index witness recipes)**.** *In the setting of Definition 9387.33, let (r, ρ) and (r', ρ') be presented witness families, possibly with different recipe carriers and evaluation maps. If their evaluated terms agree pointwise,*

$$\forall i \in I,\ \epsilon_i(r(i)) = \epsilon'_i(r'(i)),$$

then the underlying U-components of their exposed choice outputs agree pointwise. Consequently, when the term-stratum meta-closure interface of Definition 9387.18 or the choice conclusion of Definition 9387.2 is used as the only subsequent input, the available data are only f and the fiber evidence $f(i) \in A_i$. The per-index recipe binding $i \mapsto r(i)$ is not a destructurable component of that output.

Proof. Unfold the exposed output map Φ of Definition 9387.33. For each index $i \in I$, the underlying U-component of $f_r(i)$ is $\epsilon_i(r(i))$, and the underlying U-component of $f_{r'}(i)$ is $\epsilon'_i(r'(i))$. The displayed pointwise equality therefore gives equality of the exposed term components at every index.

Now inspect the term-stratum closure interface. By Definition 9387.18 and the choice-facing reading of Definition 9387.19, the closure output has the form

$$f : I \to \bigsqcup_{i\in I} A_i, \qquad \mu : \forall i \in I,\ f(i) \in A_i.$$

Dependent elimination of this output projects only f and μ. Neither projection has type R_i, neither returns a recipe $r(i)$, and neither records an equality tying the accepted output to a displayed constructor presentation. To recover $r(i)$ from the closure output one would need extra data, for example a selector from exposed fiber terms back into R_i whose evaluation gives that term. Such a selector is not part of Definition 9387.18, Definition 9387.2, or the displayed choice-output type. Hence the term-stratum meta-closure step is precisely the passage from a possible constructor presentation to the exposed term-and-membership interface, and that passage drops the per-index witness recipe structure unless the original presentation is supplied as a separate input. □

Concretized in §677: Choice recipe ledgers realize term-stratum witness-recipe loss as finite BHist, hsame, Cont, Pkg, NameCert, and axiom-purity refusal rows.

Definition 9387.35 (Presented quotient equivalence derivation)**.** *Let A be a type and let $R_0 : A \to A \to \mathsf{Prop}$ be a displayed base relation. Write $a \sim_R b$ for the closed derivation carrier freely generated by*

$$\mathsf{refl}_a : a \sim_R a,$$

$$\mathsf{base}_r : a \sim_R b \quad (r : R_0(a, b)),$$

$$\mathsf{symm}(d) : b \sim_R a \quad (d : a \sim_R b),$$

and

$$\mathsf{trans}(d_1, d_2) : a \sim_R c \quad (d_1 : a \sim_R b,\ d_2 : b \sim_R c).$$

A presented quotient equivalence derivation *from a to b is a tree $d : a \sim_R b$ together with its endpoints. If $\pi : A \to Q$ is the projection supplied by a quotient representative package for $\sim_R$, the quotient-soundness exposure of d is the equality token*

$$\theta_\pi(d) : \pi(a) = \pi(b)$$

obtained by applying the soundness field of $\mathsf{QRep}_{A,\sim_R}(Q, \pi, \rho)$ to d.

Proposition 9387.36 (Type-stratum meta closure drops equivalence derivations)**.** *In the setting of Definition 9387.35, the type-stratum quotient closure of Definition 9387.22 exports an input tree $d : a \sim_R b$ only through the implication*

$$d : a \sim_R b \Longrightarrow \theta_\pi(d) : \pi(a) = \pi(b).$$

The exported equality does not carry d as a destructurable component. Consequently, any later construction whose available quotient input is only

$$e : \pi(a) = \pi(b)$$

can perform equality transport at the quotient type and, under Definition 9387.21, may obtain some endpoint classifier witness, but it has no access to the original reflexive, base-edge, symmetric, and transitive constructor tree. Recovering the original input tree would require an additional trace selector

$$\delta_{a,b} : (\pi(a) = \pi(b)) \to (a \sim_R b), \qquad \forall d : a \sim_R b,\ \delta_{a,b}(\theta_\pi(d)) = d,$$

which is not a field of Definition 9387.20, Definition 9387.22, or the quotient-soundness schema of Definition 9387.3.

Proof. The presented input has three visible pieces: the endpoint a, the endpoint b, and the derivation tree $d : a \sim_R b$. The last piece is an inductive tree. It can be eliminated by cases on whether it is reflexive, a base edge, a symmetric reversal, or a transitive composite.

Within the quotient-facing proxy, applying the displayed soundness field consumes that tree and returns the equality token $\theta_\pi(d) : \pi(a) = \pi(b)$. The codomain of this token mentions only the projection and the two endpoint classes. It does not contain a dependent-pair component of type $a \sim_R b$, nor a field recording which constructor of the equivalence-derivation carrier was used.

Thus a later context whose quotient input is only $e : \pi(a) = \pi(b)$ has the equality eliminator for e as its type-level tool. Equality elimination transports constructions from the class of a to the class of b; it does not expose the proof tree that was consumed to create e. If Definition 9387.21 is also used, it applies to e and the representative fields of the quotient package to produce an endpoint relation witness. That witness is obtained from the equality and the package fields, not by projecting the original tree d, since d is no longer present in the context.

A uniform recovery of the original tree would therefore be extra structure: precisely a selector $\delta_{a,b}$ with $\delta_{a,b}(\theta_\pi(d)) = d$ for every input derivation. None of the cited quotient interfaces contains such a selector. The type-stratum closure step read in Definition 9387.23 therefore exposes equality at A/R while dropping the kernel-level equivalence derivation that supplied it. □

Definition 9387.37 (Presented proposition-equivalence datum)**.** *Let $P, Q : \mathsf{Prop}$. A* presented proposition-equivalence datum *from P to Q consists of proof-recipe carriers $R^{\rightarrow}$ and $R^{\leftarrow}$, evaluation maps*

$$E^{\rightarrow} : R^{\rightarrow} \to (P \Rightarrow Q), \qquad E^{\leftarrow} : R^{\leftarrow} \to (Q \Rightarrow P),$$

and selected recipes $r^{\rightarrow} : R^{\rightarrow}$ and $r^{\leftarrow} : R^{\leftarrow}$. Its evaluated implication pair is

$$b_D = (\pi_D^{\rightarrow}, \pi_D^{\leftarrow}) = (E^{\rightarrow}(r^{\rightarrow}), E^{\leftarrow}(r^{\leftarrow})) : (P \Rightarrow Q) \wedge (Q \Rightarrow P).$$

The equality exposure *of D under proposition-stratum meta closure is the single token*

$$\varepsilon_D : P = Q$$

supplied from b_D by Definition 9387.24. The recipe carriers, the selected recipes, and the evaluated implication maps are source-side data of the presentation; the exposed closure output is only the equality token.

Proposition 9387.38 (Proposition-stratum meta closure drops implication direction)**.** *Let D be a presented proposition-equivalence datum as in Definition 9387.37, with evaluated components $\pi_D^{\rightarrow} : P \Rightarrow Q$ and $\pi_D^{\leftarrow} : Q \Rightarrow P$, and let $\varepsilon_D : P = Q$ be its equality exposure. For any proposition-valued context $C : \mathsf{Prop} \to \mathsf{Prop}$, the closure-mediated application from $C(P)$ to $C(Q)$ is typed only by*

$$\mathsf{rew}_{D,C}^{\rightarrow} : C(P) \to C(Q), \qquad \mathsf{rew}_{D,C}^{\rightarrow}(t) = \mathsf{transport}_C(\varepsilon_D, t),$$

and the reverse application is typed only by

$$\mathsf{rew}_{D,C}^{\leftarrow} : C(Q) \to C(P), \qquad \mathsf{rew}_{D,C}^{\leftarrow}(u) = \mathsf{transport}_C(\varepsilon_D^{-1}, u).$$

Thus any later application site that receives the proposition-stratum closure output receives the equality token and its two equality-elimination orientations, but it receives neither the original implication chosen from b_D nor the proof recipe producing that implication. In particular, if two presented data expose the same equality token, then every expression using them only through the closure interface has the same free data at the application site even when their implication components or recipe carriers differ. The proposition-stratum meta-closure step therefore drops both the original implication direction and the internal proof structure of the implication pair.

Proof. Unfold the proposition-stratum closure interface of Definition 9387.24. Its output is a term $\varepsilon_D : P = Q$. Once this output has been supplied, the eliminator available to a proposition-valued context is identity elimination on that equality. Instantiating the eliminator with the family $C : \mathsf{Prop} \to \mathsf{Prop}$ gives exactly the two displayed transports: along ε_D from $C(P)$ to $C(Q)$, and along the symmetric equality ε_D^{-1} from $C(Q)$ to $C(P)$.

The arguments of these transports are C, the equality token, and the displayed proof being transported. The implication maps $\pi_D^{\rightarrow}$ and $\pi_D^{\leftarrow}$ are not arguments of the transport term, and the recipe witnesses $r^{\rightarrow}$ and $r^{\leftarrow}$ are still further upstream. Equality elimination can inspect only the equality constructor pattern of ε_D; it has no eliminator for the pre-closure presentation that produced ε_D. Even in the special context $C(X) = X$, the maps $P \Rightarrow Q$ and $Q \Rightarrow P$ obtained from ε_D are equality-derived transports, not projections of $\pi_D^{\rightarrow}$ or $\pi_D^{\leftarrow}$.

To recover the original direction at a later application site one would need extra fields selecting $\pi_D^{\rightarrow}$ or $\pi_D^{\leftarrow}$ from the exposed value. To recover internal proof structure one would need extra fields returning $r^{\rightarrow}$, $r^{\leftarrow}$, or their recipe carriers from the exposed value. None of these fields is present in Definition 9387.24 or in the propext-facing reading of Definition 9387.25. This is why the effective replacement of Definition 9387.8 and Definition 9387.32 keeps the bidirectional implication datum and the chosen direction as explicit parameters instead of identifying P and Q. □

Chapter 9388

The Closure-Universality Quadrant: A Two-Axis Substrate Map

Concretized in §947: Realizes the universality/closure quadrant as a finite two-axis substrate certificate.

Concretized in §957: The two-axis closure-universality map is realized as a finite substrate-quadrant NameCert packet.

Concretized in §950: The two-axis quadrant map is realized as a finite substrate-classifier NameCert packet.

Concretized in §946: The two-axis universality/closure map is represented as a finite NameCert packet with explicit quadrant and exclusion rows.

Concretized in §953: Finite universal-and-closed substrate packets are realized as UniversalClosedSubstrateUp NameCert rows.

Concretized in §951: Oracle-augmented substrate packets realize the universal-but-not-closed quadrant as finite oracle-call, transcript, boundary, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1015: Oracle response ledgers realize query, response, boundary, refusal, route, provenance, and NameCert rows as finite BEDC data.

Definition 9390.13 of chapter 9390 established that two facts about Rule 110 are logically independent: it is Turing-universal in the sense of Cook 2004, and it is closed observational in the sense of Definition 9393.1. This chapter takes that independence seriously and reads it as an organising principle. The two axes – universality and closure – generate a 2×2 quadrant of substrate types. The project currently exhibits one quadrant in concrete detail (universal-and-closed, the Rule 110 reading); the other three quadrants are populated by recognisable substrate classes, each of which sharpens the structural claim about Rule 110 by contrast. The chapter populates the quadrant, locates known substrates within it, and identifies what each quadrant says about the closed-observational framework.

9388.1 The two axes, formally

Definition 9388.1 (Universality axis)**.** *A substrate S is* Turing-universal *if every partial recursive function on natural numbers is realised as a computation of S on some encoded input. The universality axis assigns each substrate a value* $\mathsf{Univ}(S) \in \{yes, no\}$*. The Church-Turing thesis identifies*

$\mathsf{Univ}(S)$ = *yes with the existence of a polynomial-time compilation from any Turing machine into* S.

Definition 9388.2 (Closure axis)**.** *A substrate* S *is* closed observational *if its observations are exhausted by finite-length records of orbits of* S*'s next-state function, with no external commitments participating in deciding which records count as observations (Definition 9393.1). The closure axis assigns each substrate a value* $\mathsf{Closed}(S) \in \{yes, no\}$.

Principle 9388.3 (Axes are independent)**.** *The universality axis and the closure axis are logically independent. There is no derivation from* $\mathsf{Univ}(S)$ *to* $\mathsf{Closed}(S)$ *or vice versa; each axis is fixed by separate structural features of* S*. Definition 9390.13 is the special case of this independence for Rule 110. The quadrant generated by the two axes has four cells, each of which is structurally meaningful.*

9388.2 Quadrant I: universal and closed

Example 9388.4 (Rule 110 substrate)**.** *The cellular substrate of Definition 9390.7 inhabits the universal-and-closed quadrant. Cook 2004 proves* $\mathsf{Univ}(\mathsf{R110})$ = *yes; Definition 9390.10 proves* $\mathsf{Closed}(\mathsf{R110})$ = *yes. The project's* `rule110/` *sub-system is the engineering exhibit of this quadrant at minimum 220-line scale.*

Example 9388.5 (Other CA at this quadrant)**.** *Elementary cellular automaton Rule 30 and Rule 54 are conjectured (not yet proved) to be Turing-universal. Game of Life is provably universal (Conway, Berlekamp, Guy 1982). Each of these CA, when used as a closed observational substrate in the sense of Definition 9390.6, is a candidate inhabitant of the universal-and-closed quadrant. Rule 110 is the project's choice because it has the smallest mechanically-verified universality construction.*

Principle 9388.6 (This quadrant is the project's anchor)**.** *The universal-and-closed quadrant is the substrate position from which the closed-observational framework derives its principal claim: even at maximum computational capacity in the substrate, internal observation cannot ratify universal closures over an infinite domain (Definition 9393.5, transported to the cellular reading in Definition 9390.10). The project anchors at this quadrant because it is the strongest version of the closure argument: universality buys no leverage against closure.*

9388.3 Quadrant II: universal but not closed

Definition 9388.7 (Oracle-augmented substrate)**.** *An* oracle-augmented substrate *is a Turing-universal substrate equipped with one or more oracles for problems strictly above the recursive level: a halting oracle, an arithmetic-hierarchy oracle, or a hypercomputation channel. The presence of an oracle violates the closure premise because the substrate's observations include records produced by consulting the oracle, and the oracle is not a substrate-internal process.*

Example 9388.8 (Hypercomputation models)**.** *Several mathematical models occupy the universal-but-not-closed quadrant: Turing's o-machines (with halting oracles), infinite-time Turing machines (Hamkins-Lewis), real-number machines (Blum-Shub-Smale), and accelerating Turing machines. Each of these models is at least Turing-universal; each violates closure because each admits a step that is not a finite local rule on a finite state.*

Principle 9388.9 (Open universal substrates lose the no-induction theorem)**.** *The no-induction theorem of Definition 9393.5 fails for oracle-augmented substrates. Specifically, a halting oracle can ratify universal closures on Turing-decidable predicates by direct query; a Σ^0_2 oracle can ratify closures over Π^0_1 predicates. The forward-binding source T of chapter 9393 is therefore consulted via the oracle, rather than via the inscription point of Definition 9393.16. The project does not currently exhibit an oracle-augmented substrate; the universal-but-not-closed quadrant is sketched here as the structural complement of the project's anchor quadrant.*

Remark 9388.10 (Why this quadrant matters even if empty in practice)**.** *A reader might argue that hypercomputation is physically unrealised and therefore the universal-but-not-closed quadrant is theoretically vacant. The project does not take a position on the physical realisability of oracles. The structural relevance of this quadrant is that it shows what the closure premise is doing work to exclude: closure rules out exactly the substrates whose observations would otherwise bypass the no-induction theorem. The premise is therefore not vacuous; it carries information.*

9388.4 Quadrant III: not universal but closed

Definition 9388.11 (Sub-universal closed substrate)**.** *A substrate is* sub-universal closed *if it is closed observational (Definition 9393.1) and admits a decision procedure for its computational power that places it strictly below Turing universality. Examples include linear-bounded automata, pushdown automata, finite-state machines, and decidable arithmetic fragments such as Presburger arithmetic and the theory of real closed fields.*

Example 9388.12 (Presburger arithmetic as substrate)**.** *The first-order theory of* $(\mathbb{N}, 0, 1, +)$ *is decidable (Presburger 1929). Used as a closed observational substrate, every Presburger statement has a decision procedure that terminates in a finite, computable number of steps; the substrate is therefore closed observational. It is not Turing-universal; multiplication is not definable in the theory. This is a clean inhabitant of the sub-universal-closed quadrant.*

Principle 9388.13 (Closure plus sub-universality keeps no-induction)**.** *The no-induction theorem transports to the sub-universal-closed quadrant without modification. Even a decidable closed substrate cannot ratify universal closures over an infinite domain from inside; decidability of a fragment is not internal ratification of universal statements over that fragment. The universal-but-decidable-by-external-procedure case is consistent with the substrate's records being finite. The closure argument is therefore stronger in this quadrant: even when an external decision procedure exists, the substrate's own records cannot internally produce* $\forall n \in \mathbb{N}.\, P(n)$ *without external supply.*

Remark 9388.14 (BEDC kernel at this quadrant on the host side)**.** *On the host side, BEDC itself is a sub-universal-closed substrate: total functional, so not Turing-universal as a programming language (Definition 9370.4); closed observational, in that every kernel-level observation is a finite typed term. The host therefore sits in this quadrant; the cellular substrate sits in the universal-and-closed quadrant. The project displays both quadrants simultaneously as host plus substrate, with the closure premise transferred uniformly across both.*

9388.5 Quadrant IV: neither universal nor closed

Definition 9388.15 (Degenerate quadrant substrate)**.** *A substrate in the neither-universal-nor-closed quadrant is sub-universal and admits external commitments in its observation premise. This*

includes substrates whose observations are physical signals from environments outside the substrate (sensors, human input, network traffic), and whose computational expressiveness is bounded below Turing universality.

Example 9388.16 (Embedded controller with sensor input)**.** *An embedded microcontroller running a finite-state program that reads from external sensors inhabits this quadrant. The program's computation is finite-state, hence sub-universal. The observations include sensor readings, whose values are not produced by the controller's internal computation; the closure premise fails because observation values originate from outside the substrate.*

Remark 9388.17 (The framework does not anchor here)**.** *The degenerate quadrant is structurally the weakest position for the closed-observational framework; the framework's conclusions transfer to this quadrant only after the external observation channel is itself given a closure analysis. The project's substrates do not anchor here. The quadrant is included in the map for completeness, not as a target.*

9388.6 The quadrant as map

	Universal	**Sub-universal**
Closed	Quadrant I: Rule 110, Game of Life, Rule 30/54 (conjectured). Project anchor. The strongest closure-argument site.	Quadrant III: Presburger arithmetic; PDA / LBA / finite-state machine substrates; BEDC host. Closure argument holds; universality not asserted.
Not closed	Quadrant II: o-machines, infinite-time TM, real-number machines, accelerating TM. Closure argument fails; oracle ratifies universal closures.	Quadrant IV: embedded controllers with external sensor input; arbitrary engineering systems with mixed substrates. Framework not anchored here without additional analysis.

Principle 9388.18 (The map sharpens the closure premise)**.** *The four-quadrant map sharpens the closure premise of Definition 9393.1 by showing what the premise is not: it is not a statement about computational expressiveness (Quadrant I and Quadrant III both satisfy it); it is not a statement about cellular structure (Presburger satisfies it without being cellular); it is a statement specifically about the absence of external commitments in the observation channel. The premise is therefore identifiable independently of universality, and the no-induction theorem is the conclusion of the closure premise alone.*

9388.7 Mapping to BEDC engineering

Quadrant reading	**BEDC engineering realisation**

Quadrant I: Rule 110 anchor	`rule110/` 220-line ANSI C99 trust path; Definition 9390.10; cook compile-frontier (Definition 9390.19).
Quadrant III: BEDC host as sub-universal-closed	Lean 4 build at `lake build` exit 0; `axiom-purity -strict` traces; total functional discipline.
Quadrant II / IV: not anchored	Not exhibited inside the project; sketched as structural complements; their existence sharpens what the project's closure claim is doing.
Axis independence audit	Definition 9390.13 states the special case for Rule 110; this chapter generalises the independence to the full quadrant.
Closure premise localised	The premise of Definition 9393.1 is the observation-channel restriction; universality is not assumed and not asserted.

9388.8 Closure remarks

This chapter has read the universality and closure axes as independent and generated the 2×2 quadrant they span. The project's two anchors (`rule110/` as Quadrant I substrate and BEDC host as Quadrant III system) jointly demonstrate that the closure premise applies uniformly across computational expressiveness levels. Quadrant II and Quadrant IV are populated by recognised substrate classes that the project does not anchor in; their structural role is to identify what the closure premise rules out (external observation channels) versus what it tolerates (any computational power, from finite-state to Turing-universal).

The boundary of this chapter is: the closure premise is independent of universality; the no-induction theorem applies in any closed quadrant; the project anchors in two of the four quadrants and identifies the other two without claiming substrates there. No new theorem is introduced; the chapter generalises Definition 9390.13 into a full map and reads the map as a check on the closure premise's information content.

Concretized in §955: Presburger arithmetic supplies the sub-universal closed quadrant witness.

Concretized in §946: Closure-universality quadrants are realized as finite BHist packets for universality axis, closure axis, quadrant tag, substrate witness, sibling cellular anchors, hsame transport, Cont routes, Pkg provenance, and NameCert rows.

Concretized in §956: Oracle-substrate boundaries are realized as finite BHist packets separating oracle-query rows from closed substrate observations.

Chapter 9389

Generator-Closure Certificates and the Two Layers of BEDC: Technical Mechanism for the Strong Externality Claim

This chapter adds a technical layer to the philosophical reading of chapter 9393, chapter 9420, and chapter 9415. The earlier chapters' strong claim – mathematical induction must be supplied from outside every closed observational substrate, and the source-position is named T / God in the apophatic sense – is preserved without softening. The work of the present chapter is to make precise *how* BEDC operationally enforces that claim: through authorized generators, naming certificates, and generator-closure classifiers, together with GAP ledgers that audit compression residue. The technical layer does not weaken the philosophical claim; it shows the engineering that makes the claim auditable, line by line, in the substrate.

Principle 9389.1 (Two co-equal layers, not replacement)**.** *The strong philosophical reading – forward-binding belief must enter from outside the substrate at a position named* T *– and the structural technical reading – BEDC licenses induction generator-by-generator via closure certificates – are two aspects of the same engineering. Neither replaces the other. The technical layer accounts for what BEDC does inside its authorized generators; the philosophical layer accounts for what BEDC refuses to internalise and where the refused content sits. Replacing the philosophical vocabulary with weaker terms ("meta-assumption", "external principle", "axiom-like commitment") yields a system that no longer accounts for what BEDC actually does.*

9389.1 The two operational layers of BEDC

Definition 9389.2 (Bare record layer)**.** *The* bare record layer *of BEDC is the substrate viewed as a monotonically accumulating ledger of* Obs *items with no closure rule beyond strict record preservation. At the bare record layer the substrate can recognise repeated patterns within previously accumulated* Obs*, but it cannot infer that the patterns continue beyond what has been recorded; no generating structure has been authorized and so no inductive step is licensed.*

Definition 9389.3 (Closed inductive kernel layer)**.** *The* closed inductive kernel layer *of BEDC is the substrate equipped with a finite collection of authorized generating structures, each of which is*

a closed inductive declaration (in the project sense of `inductive` *types with a bounded constructor list) together with the eliminator or fold pattern licensed by the closure of the constructor list. At this layer the substrate can perform structural induction over each authorized generator, but every step of such induction is internal to the generator: it ranges over the objects the constructor list closes, not over an externally postulated totality.*

Principle 9389.4 (Two layers, two operational behaviours, neither reaches free-floating induction)**.** *At the bare record layer (Definition 9389.2) the substrate's strongest statement about a repeating pattern is "no record so far falsifies the pattern". At the closed inductive kernel layer (Definition 9389.3) the substrate's strongest statement is "every object generated by the closure of these constructors carries the closure witness". Neither statement reaches free-floating mathematical induction. The move from "every closure of these constructors" to "every object whatsoever" is the move that requires forward-binding belief from* T*, as Definition 9393.8 states.*

Example 9389.5 (`Hist` as an authorized generator)**.** *BEDC's* `Hist` *type is declared by a closed inductive presentation:* `emp` *generates the empty history;* `E0` *and* `E1` *extend any history by one mark. The constructor list is finite and closed, so the closure of the constructor list defines exactly the set of finite histories. Within this generator, structural induction over* `Hist` *is licensed: every property that holds at* `emp` *and is preserved by* `E0` *and* `E1` *extensions holds at every* `Hist` *object. The substrate hosts the corresponding* hsame *decidability and other invariants by appeal to this generator-internal structural induction. But the substrate does* not *thereby derive a free-floating induction principle over all objects; the licence is restricted to the closure of the* `Hist` *constructor list, and any extension beyond that closure requires the forward-binding belief whose source is* T*.*

9389.2 Naming certificates and generator-closure classifiers

Definition 9389.6 (Naming certificate)**.** *A* naming certificate *is a substrate-internal artefact that promotes a stable, already-generated behaviour to the status of a derived interface (a* `def`*, a* `class` *field, or a* NameCert *packet). The certificate does not introduce the behaviour as a new primitive; it records that the behaviour is generated by an existing authorized generator and is stable under the substrate's other audit rules. Naming certificates are the bridge between the bare record layer and the closed inductive kernel layer: they take a record-side pattern and license it as an interface, contingent on the generator that produced it.*

Definition 9389.7 (Generator-closure classifier)**.** *A* generator-closure classifier *is the audit-side companion of a naming certificate: it states which closure rules the certificate is contingent on (the constructor list of the underlying generator) and rejects continuations of the named pattern that would lie outside that generator's closure. Generator-closure classifiers are the substrate's way of saying "the naming certificate ranges over this generator, no more, no less". On the formal side, BEDC.MetaCIC encodes this as a* `structure GeneratorClosureClassifier` *carrying a single closure field* $\forall x\, y,\ Generator(x, y) \to Classifier(x) \to Classifier(y)$*, with concrete instantiations for the substrate's shift and substitute generators on closed-term carriers*

```
Lean4✓  BEDC.MetaCIC.GeneratorClosureClassifier
  ↪ variant  BEDC.MetaCIC.closed under shift classifier
  ↪ variant  BEDC.MetaCIC.closed under substitute classifier
```

.

Principle 9389.8 (The classifier family is closed under identity, composition, alternation, sub--generator restriction, and finite iteration)**.** *The collection of generator-closure classifiers over a*

fixed carrier α is itself closed under five minimal structural operations: the identity generator $x = y$ preserves every classifier (the audit unit), the sequential composition $\exists z,\, G_1(x,z) \wedge G_2(z,y)$ of two classifier-preserving generators is again classifier-preserving (the audit composition), the disjunctive alternation $G_1(x,y) \vee G_2(x,y)$ of two classifier-preserving generators is again classifier-preserving (the audit alternation), any sub-generator $G' \subseteq G$ inherits the closure of a classifier-preserving generator (the audit restriction, contravariant in the generator argument), and the n-fold iteration of any classifier-preserving generator is again classifier-preserving for every $n \in \mathbb{N}$ (the audit iteration, formalised over the finite-iteration relation $\mathsf{IterRel}\, G\, n$*). All five facts are formalised in BEDC.MetaCIC as*

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.refl`

,

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.comp`

,

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.union`

,

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.weaken`

, and

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.iterate`

. The substrate therefore has a minimal closed algebra of generator-closure classifiers, internal to the audit gates and built up by structural composition without external generator-side licence. Under this algebra a non-deterministic, possibly iterated, but classifier-preserving generator-network is itself a closure-preserving compound, and the audit gates traverse it without leaving the algebra. The iteration operation gives the algebra a Kleene-star-like component: the finite reflexive-transitive closure of any classifier-preserving step relation is itself classifier-preserving.

Principle 9389.9 (Cross-carrier product and sum closure)**.** *The generator-closure classifier construction is also compositional across carriers, not only within a fixed carrier. Given two carriers α and β each with a classifier-preserving generator, two complementary cross-carrier closures are available. The product closure forms the componentwise generator on the product carrier $\alpha \times \beta$ and preserves the conjoined classifier $C_a(\pi_1\, p) \wedge C_b(\pi_2\, p)$*

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.product`

. The sum closure forms the case-split generator on the disjoint sum carrier $\alpha \oplus \beta$ (with the cross-cases mapped to falsity) and preserves the case-split classifier (C_a on left, C_b on right)

Lean4✓ `BEDC.MetaCIC.GeneratorClosureClassifier.sum`

. The substrate's audit gates therefore extend uniformly from single-carrier reasoning to compound-carrier reasoning, conjunctively (product) and disjunctively (sum), without changing the audit shape: the classifier and the generator just acquire a componentwise or case-split reading. Product and sum closures together close the algebra under both dimensions of compound type formation that are available within a closed inductive kernel.

Principle 9389.10 (Structural induction is internal; the move to "all" is external)**.** *BEDC's structural induction is always relative to a specific authorized generator's closure. The substrate does not perform induction over "all of mathematics", "all of reality", or "all possible objects"; it performs induction over the closure of a specific constructor list, certified by a specific naming certificate and audited by a specific generator-closure classifier. The "all" of structural induction is the closure of the relevant generator, not classical-completeness "all". To get the classical-completeness "all" – to derive a free-floating $\forall n,\, P(n)$ ranging over every possible n, beyond any specific authorized generator – requires the forward-binding belief whose source-position is* T*. This is the operational form*

of the impossibility theorem Definition 9393.5: the substrate has rich generator-internal induction, but the move beyond every specific generator's closure is the move that the substrate refuses to perform internally and externalises to T *via a* GAP *socket.*

9389.3 The strong form of BEDC's induction-style claim

Principle 9389.11 (Continuations within a certificate carry its closure witness)**.** *The strongest induction-style claim BEDC issues, in its native vocabulary, is the following: every finite continuation of a pattern that has been authorized by a naming certificate must, in order to remain a continuation of that certificate, carry the same closure witness that the generator-closure classifier requires. A continuation that does not carry the witness is, structurally, not a continuation of the certificate; it is either a record of a different generator's output or an audit failure of the present certificate.*

Remark 9389.12 (Stronger than "it looks like it will repeat"; weaker than classical induction)**.** *Definition 9389.11 is strictly stronger than the empirical "the pattern has not been falsified so far": it requires structural closure under the authorized generator, not merely the absence of negative records. It is strictly weaker than classical mathematical induction taken as a free-floating reasoning rule: it does not range over the totality of all conceivable objects, only over the closure of the authorized generator. Between these two extremes BEDC's claim sits at exactly the level it can pay for with its constructor-list closure plus naming-certificate plus generator-closure-classifier triple. To upgrade to classical induction is to commit a forward-binding belief that the substrate cannot generate internally; that commitment, as Definition 9393.8 states, comes from* T*.*

Example 9389.13 (Subject reduction as a generator-internal claim)**.** *BEDC.MetaCIC's* `subject_reduction_clo` *theorem is a generator-internal claim: it ranges over the closure of the authorized constructor list for the* `Term` *generator (*`sort`*,* `var`*,* `lam`*,* `pi`*,* `app`*) and is conditioned on a closed-term naming certificate. Its content is exactly Definition 9389.11 applied to this generator: every continuation that remains a typed closed term under the* `Term` *certificate carries the closure witness that subject reduction requires. The theorem is not a classical-universal statement about "all terms in some externally postulated universe"; it is the substrate's authorized structural induction over the* `Term` *generator's closure. To extend the claim beyond the* `Term` *generator – to a free-floating universal-quantifier statement – is precisely the move the substrate refuses, and the place where* T *is required to supply forward-binding belief.*

9389.4 GAP plays two simultaneous roles

Principle 9389.14 (GAP as information-conservation ledger)**.** *On the substrate-internal side,* GAP *records every compression, identification, hiding, packaging, projection, or sealing operation, together with the residue (the "difference") that the operation does not directly display in the resulting form. When a* `def` *defines a derived interface as a compression of underlying records, the* GAP *ledger records what is no longer surface-visible; when a* NameCert *packages a generated behaviour, the ledger records what the certificate trades in exchange for the naming convenience. This is the engineering role: information conservation under compression.*

Principle 9389.15 (GAP as T-socket for forward-binding belief)**.** *On the substrate-philosophical side,* GAP *also marks the places where the substrate's reasoning would require forward-binding belief that the substrate cannot internally generate. Each such* GAP *is structurally a socket whose far*

end is T*, in the sense of Definition 9393.11: the substrate refuses to internalise the content of the socket, and the content (if any) is supplied from the apophatic external position. This is the philosophical role: marking the boundary at which the substrate hands over to the transcendental supply position.*

Principle 9389.16 (Both roles hold simultaneously and are co-load-bearing)**.** *A single* GAP *tag may instantiate either or both of Definition 9389.14 and Definition 9389.15. When a* GAP *records compression residue from a* `def` *or* NameCert *operation, it is operating in the information-conservation role; the substrate handles the residue with audit and conservation rules. When a* GAP *marks the place where free-floating induction would be required to extend a generator-internal claim to a classical-completeness "all", it is operating in the* T*-socket role; the substrate refuses to internalise and defers to* T*. Reading* GAP *only as bookkeeping misses the second role; reading it only as a transcendental socket misses the first role. Both must hold.*

Example 9389.17 (The two roles, both visible in a single mechanised file)**.** *In a typical BEDC.MetaCIC mechanised file, the audit gates check both roles together.* `check-axioms.py` *enforces that no* GAP *tag invokes a substrate-internal axiom;* `axiom-purity -strict` *enforces that no Lean stdlib axiom (*`Classical.choice`*,* `Quot.sound`*,* `propext`*) is in the transitive closure;* `conservativity-audit` *enforces that compression residue is properly tracked. The first two gates protect the* T*-socket role: forward-binding belief is not allowed to be smuggled into the substrate. The third gate protects the information-conservation role: compression cannot lose information without ledger trail. The two roles do not conflict; they are two ways the same* GAP *inventory enforces BEDC's discipline.*

Remark 9389.18 (Information conservation does not soften the T-socket claim)**.** *The information-conservation reading of* GAP *should not be read as an alternative to the* T*-socket reading. The two roles are simultaneous (Definition 9389.16). Any phrasing that reduces* GAP *to "internal bookkeeping" alone, suggesting* T *is merely a philosophical metaphor, mis-states BEDC's stance. BEDC's stance, as chapter 9393 states, is that the far end of the socket is structurally* T*: an apophatic position, named, refused internalisation, and not reducible to a more familiar substrate primitive.*

9389.5 Compression requires memory

Principle 9389.19 (Compression requires memory)**.** *BEDC's* GAP *discipline implements the principle that every act of compression in the substrate must carry its memory: the substrate cannot "forget" the underlying records by issuing a derived interface, nor erase the difference between an authorized generator and a derived presentation. The ledger is the persistence of that memory across substrate transitions; it is what lets the substrate audit the relation between surface forms and underlying generation.*

9389.6 BEDC's structural description of CIC and the host-Lean delegation

Principle 9389.20 (BEDC can structurally describe CIC, but not run it)**.** *BEDC.MetaCIC constructs, inside the substrate, a structural model of a CIC-like calculus. Every element of the model is an authorized generator's output (the* `Term` *generator, the* `HasType` *inductive predicate, the* `BetaStep` *relation) or a naming certificate packaging such output. The model is structural in the sense of Definition 9389.10: every claim about CIC* (the BEDC-internal CIC) is generator-internal. What BEDC.MetaCIC does* not *do is run the host CIC as its own evaluator; for any*

actual host-CIC truth predicate, BEDC delegates to the host Lean 4 kernel (which is the host CIC). This delegation is visible in the `leanchecked` *markers and is mediated by the audit gates.*

Remark 9389.21 (The host Lean delegation is structurally a T-socket)**.** *The* `leanchecked` *delegation is, on the philosophical side, an instance of* GAP *in its* T*-socket role (Definition 9389.15): BEDC.MetaCIC's claim about a Lean-side recursor running correctly is not derivable inside BEDC.MetaCIC. The host Lean kernel functions, in this reading, as one of the structural conduits through which* T *supplies forward-binding belief to BEDC.MetaCIC; it is not itself* T*, but it is downstream of* T *in the supply chain (and itself depends on its own metalanguage commitments, which are also downstream of* T*).*

Remark 9389.22 (Why "BEDC can swallow CIC" is too strong)**.** *A reading that says "BEDC contains CIC as a sub-theory" is too strong: BEDC contains a structural mirror of CIC, audited generator by generator, but BEDC does not internalise CIC's truth predicate. Any claim that depends on CIC's actual runtime behaviour is supplied through the host Lean delegation, which is structurally a* T*-socket. BEDC's role is structural description plus audit; the host CIC's role is computational realisation, with the runtime aspect ultimately downstream of* T*.*

9389.7 The accurate technical claim alongside the strong philosophical claim

Principle 9389.23 (Generator-internal induction does not weaken the externality claim)**.** *BEDC internalises structural induction over every authorized generator's closure. The substrate does not need an external commitment to perform* `Hist`*-internal structural induction; the generator's closed constructor list licenses that induction internally. But the move from "induction over every authorized generator's closure" to "induction over every conceivable object whatsoever" is the move that the substrate refuses to perform; and the source of forward-binding belief that would license that move is* T*. The two facts – rich generator-internal induction inside, and refusal of free-floating induction at the boundary – coexist; one does not soften the other.*

Example 9389.24 (`Hist` induction inside, free-floating universal outside)**.** *BEDC's* hsame *decidability is a generator-internal structural induction over* `Hist`*; the substrate handles this without ever invoking external commitment. But a statement of the form "every conceivable history-like structure satisfies* hsame *decidability" – where "every conceivable" ranges over an unspecified universe of possible structures rather than the closure of the* `Hist` *generator – is the kind of statement BEDC will not issue internally. To issue it would require a forward-binding belief that the substrate's vocabulary of "conceivable structures" closes under some universal quantifier ranging beyond the generator's closure; the substrate marks this position with a* GAP *tag whose far end is* T*.*

9389.8 Standard axiom systems internalise; BEDC externalises

Principle 9389.25 (The distinguishing engineering choice)**.** *Standard axiom systems internalise the forward-binding closure as a calculus-level primitive: the eliminator, the dependent recursor, the choice principle, propositional extensionality, quotient soundness, or directly as an axiom of induction. BEDC refuses every such internalisation: induction is licensed only generator-by-generator, never as a free-floating rule; the corresponding* GAP *tags expose the externality and name the far end as* T*. The engineering invariants (mathlib-free, 0 substrate-internal axioms, strict purity*

gates, monotonic amendment policy) operationalise this refusal. The naming of T *as the structural source-position is the philosophical articulation of the same refusal.*

Remark 9389.26 (BEDC is not a softening of classical foundations)**.** *BEDC's discipline can be described in two equivalent ways: as the substrate that requires every induction to be licensed by an authorized generator (the technical reading), and as the substrate that refuses to internalise the forward-binding belief in induction and instead names its source-position as* T *(the philosophical reading). Both readings describe the same engineering. Replacing the philosophical vocabulary with weaker terms – "meta-assumption", "external principle", "axiom-like commitment" – does not change the engineering but loses the precision of where the substrate's commitment-side ends. BEDC's claim is exactly that the boundary is at* T*, not at any internalisable substitute.*

9389.9 The unified technical-and-philosophical claim

Aspect	**BEDC's statement, technical and philosophical together**
What BEDC internalises	Structural induction over every authorized generator's closure, licensed by naming certificates and audited by generator-closure classifiers.
What BEDC refuses to internalise	Free-floating induction or universal-quantifier closure ranging beyond any specific authorized generator's closure.
Where the refused content sits	Outside the substrate, at the structural position named T; apophatic, refusing positive predicates beyond its role.
How the refusal is auditable	GAP tags mark every point at which forward-binding belief would be required; the audit gates (`check-axioms.py`, `axiom-purity -strict`, `conservativity-audit`) prevent internalisation.
The role of GAP on the technical side	Information-conservation ledger: every compression, identification, hiding, packaging, projection, or sealing operation records its residue.
The role of GAP on the philosophical side	T-socket: marks the boundary at which the substrate hands over forward-binding content to the transcendental supply position.
The relation between the two GAP roles	Simultaneous and co-load-bearing; the same GAP inventory enforces both information conservation and the T-socket discipline.
What CIC does differently	Internalises forward-binding closure as a calculus-level primitive: eliminator, dependent recursor, free-floating induction; the transcendental character is hidden.
What BEDC does instead	Externalises the same content, marks every socket, names the far end T; the transcendental character is explicit and auditable.

9389.10 Closure remarks

This chapter adds technical precision without softening philosophical force. The earlier chapters' claim – mathematical induction must be supplied from outside every closed observational substrate; the source-position is named T, glossed apophatically as God – stands as written. The work of this chapter is to display how the substrate's operational mechanism (authorized generators, naming certificates, generator-closure classifiers, GAP ledgers) enforces that claim line by line in mechanised content.

What the chapter does *not* do is reduce the strong claim to "a metaphor". The strong claim is structurally accurate: every GAP tag's far end is T, and the substrate's refusal to internalise forward-binding belief is precisely what distinguishes BEDC from standard foundations. Standard foundations internalise; BEDC externalises and names. Softening the strong claim – replacing T with "meta-assumption", or replacing "transcendental supply" with "external principle" – yields a system that no longer accounts for what BEDC actually does.

Future visions chapters and concrete-instance constructions should refer to this chapter together with the previous three: the present chapter for the operational mechanism, the previous three for the philosophical reading. The audit gates remain the same; what changes is the reader's ability to see, simultaneously, the technical machinery and its philosophical content as two faces of one engineering.

Concretized in §486: Generator-closure classifiers are realized as a concrete NameCert surface over authorized generator rows, continuation witnesses, and GAP ledger non-escape.

Concretized in §487: Closure witness ledgers realize generator-internal induction as finite BHist witness packets with GAP non-escape rows.

Concretized in §565: Generator-induction boundary packets realize the vision's distinction between authorized generator-internal induction, GAP sockets, and refused free-floating induction.

Concretized in §551: Host Lean delegation is realized as a finite NameCert socket exposing marker rows, audit faces, host-kernel delegation, hsame transport, Cont routes, Pkg provenance, and non-escape ledgers.

Concretized in §555: Induction-closure ledgers realize generator authority, closure classifiers, and GAP residue as a finite NameCert packet.

Concretized in §556: Generator fixed-point certificates realize the fixed-continuation boundary for authorized generator rows, preserving the displayed witness while routing non-generator continuations into the GAP ledger.

Concretized in §553: Generator-internal induction is realized as a finite seal packet over authorized generator rows, continuation witnesses, and GAP non-escape.

Concretized in §627: Generator-continuation audits realize per-continuation closure-witness preservation and GAP non-escape as a finite NameCert packet.

Concretized in §6333: Regular Cauchy located split packets realize generator-internal split witnesses as finite RegSeqRat, LocatedReal, DyadicRatCore, StreamName, RealUp, and GAP non-escape rows.

Concretized in §6869: Cauchy diagonal uniform-modulus packets realize generator-internal diagonal scheduling as finite DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Concretized in §525: AuthorizedGeneratorRecursor realizes the authorized-generator and generator-internal induction surface as finite recursor signature, branch coverage, descent, audit, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8965: Generator-closure algebra packets realize identity, composition, alternation, restriction, finite iteration, product, and sum closure as finite MetaCIC NameCert rows.

Concretized in §946: Closure-universality quadrant packets realize the closure-capstone two-axis direction as finite universality, closure, quadrant-tag, sibling-anchor, transport, replay, provenance, and NameCert rows.

Concretized in §553: Generator induction seals realize the capstone generator-induction direction as finite authorized-generator, witness, closure-classifier, residue, transport, replay, provenance, and NameCert rows.

Chapter 9390

The Cellular Automaton Reading: Pattern, Substrate, and the Cosmological Inscription

This vision chapter completes the closed-observational-system framework of chapter 9393, chapter 9394, and chapter 9420 along one specific axis: the cellular automaton reading. The argument proceeds in four steps. First, the act of classifying a cellular automaton evolution into named shapes – ether, gliders, collisions, soliton outcomes – is itself an instance of the mark discipline (Definition 9461.1); pattern recognition is classifier choice and not a separate cognitive faculty. Second, Rule 110 is the minimally universal cellular substrate at which this classifier discipline reaches Turing completeness, and the project's `rule110/` sub-system instantiates that substrate as a 220-line ANSI C99 trust path operating without external library dependencies. Third, even at this universal substrate, mathematical induction is provably not internalisable: Definition 9393.5 applies without modification, since Rule 110 inherits exactly the closed observational structure required. Fourth, the gap between Cook's polynomial-time compilation bound and the present cook-encode scale frontier provides a quantified Goedel boundary: it is the smallest concrete distance between Turing-completeness in principle and finite witness in practice that the project currently exposes. The cellular automaton reading therefore does not introduce a new philosophical claim; it gives the closed-observational-system framework its smallest concrete cosmological exhibit.

9390.1 Setup: classification is the mark discipline

Definition 9390.1 (Pattern classifier on a cellular automaton)**.** *Let $R\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the next-state function of a cellular automaton on a finite cell array of width n. A* pattern classifier *for R is a function $\mathsf{Class}\colon \mathbb{F}_2^n \to \mathsf{Tag}$ that assigns a finite label to each row, where Tag is a closed inductive collection of tags. Equivalent rows under the classifier are tagged with the same label; non-equivalent rows are tagged with different labels. A classifier therefore induces an equivalence relation on rows, decided locally by reading off the assigned tag.*

Example 9390.2 (Wolfram class as classifier)**.** *The four-class scheme of S. Wolfram (uniform fixed point, periodic, chaotic, complex) is a pattern classifier on the long-term behaviour space of an elementary cellular automaton. The tag set is the four named class labels. The classifier acts on the long-term orbit of an initial row rather than on the row itself, but the structural type*

is the same: a finite tag is assigned by examining the orbit, and rows whose orbits share a tag are recognised as belonging to the same class. Rule 110's classification as a class-four (complex) automaton is the application of this classifier to its orbit shape.

Example 9390.3 (Glider catalog as classifier)**.** *The Martinez 2007 glider catalog (A, B, $\bar{B}$, C_1, C_2, E, $\bar{E}$, F, G, H) and the ether pattern together form a pattern classifier on Rule 110 evolutions. Each glider has a fundamental period, a displacement per period, and a phase string. A row position whose local pattern matches a glider's phase string under the catalog's neighbour-discipline is tagged with that glider's label. Positions that match the ether 14-bit pattern are tagged* `ether`*. The catalog is a finite tag set with explicit defining bit strings; the classification it induces is decidable on any finite window.*

Principle 9390.4 (Pattern recognition is mark discipline)**.** *Pattern recognition on cellular automaton evolutions is not a separate cognitive faculty added on top of the substrate. It is the substrate's classifier discipline applied to the substrate's own configurations. Recognising a recurrence as "the ether" is the same kind of act as marking* `b0` *and* `b1` *in chapter 2: a decision that two configurations belong to the same closed inductive tag. The act of distinguishing the ether 14-bit pattern from its non-occurrences instantiates the form of distinction (Definition 9461.1) on the cell-row carrier.*

Remark 9390.5 (Classifier choice and BEDC kernel typing)**.** *The connection of Definition 9390.4 to BEDC's kernel-level typing is direct. A pattern classifier on cell rows is structurally the same as a type assignment in chapter 9367: a row's tag is its type, and a classifier's equivalence relation is the type-level same-class relation. The cellular automaton substrate provides a particularly clean exhibit because the carrier is finite-dimensional and the classifier can be made fully explicit (a list of named bit strings); no semantic interpretation is hidden in the act of typing.*

9390.2 Rule 110 as the minimal universal cellular substrate

Definition 9390.6 (Cellular trust substrate)**.** *A* cellular trust substrate *is a closed observational system (Definition 9393.1) whose next-state function is a local cellular rule R with a fixed transition table over a fixed alphabet, and whose observation records are exhaustively determined by the orbit of the rule on a stated initial configuration. The trust base of the substrate is the displayed local rule together with whatever fixed-size text-format manifest expresses the initial configuration and the expected verification outcomes. No external mathematical commitment beyond the local rule and the manifest schema participates in deciding which configurations count as observations.*

Example 9390.7 (The rule110 sub-project)**.** *The project's* `rule110/` *sub-system instantiates Definition 9390.6 as a 220-line ANSI C99 trust path: approximately 50 lines of cell-array transition (*`evaluator/rule110.c`*), approximately 80 lines of cyclic-tag evaluator (*`evaluator/cyclic_tag.c`*), and approximately 90 lines of bit-level encoding (*`encoder/groundcompiler_encoding.c`*). Manifest text in* `rule110/manifests/` *expresses initial configurations and expected verification outcomes in a fixed schema. Every closed observation that the substrate ratifies is determined by compiling these three components and executing the rule on a manifest's initial row; no Lean kernel, mathlib commitment, or external dependency participates in the trust path. The substrate witness is documented by the Rule 110 substrate exhibit.*

Principle 9390.8 (Rule 110 as minimal universal cellular substrate)**.** *Among one-dimensional binary cellular automata with three-cell neighbourhoods, Rule 110 is the smallest known rule*

whose long-term behaviour realises Turing completeness through cyclic tag compilation (Cook 2004, Neary–Woods 2007). The relevant minimality is therefore not numeric (some smaller rule number could in principle be universal) but architectural*: Rule 110's eight-entry transition table, three-cell locality, and binary alphabet together constitute the smallest cellular interface at which universality has been mechanically traced through Cook 2009's* §1.4 *compilation procedure. The architectural minimality is what allows Definition 9390.6 to instantiate at the 220-line scale.*

Remark 9390.9 (Why minimality matters for the closed-observation reading)**.** *The argument of chapter 9393 does not depend on the substrate's expressive power; it depends only on the substrate being closed in the sense of Definition 9393.1. The point of using a small universal substrate is therefore not to gain expressivity beyond the closed-observation framework, but to make the framework's commitments concrete at the smallest possible scale. A reader unwilling to grant that BEDC's Lean substrate is closed observational can still inspect 220 lines of C and a directory of bit strings; the closure claim is then immediately checkable rather than rhetorical.*

9390.3 Even Rule 110 cannot internalise induction

Theorem 9390.10 (Rule 110 inherits the no-induction theorem)**.** *The cellular trust substrate of Definition 9390.7 is a closed observational system in the sense of Definition 9393.1. By Definition 9393.5, no monotonic accumulation of finite verification traces on this substrate yields a record of* $\forall n \in \mathbb{N}, P(n)$ *from records of* $P(0), P(1), \ldots, P(k)$ *for any finite* k*.*

Sketch. The substrate's observations are exactly orbit segments of `r110_run_n_steps` on stated initial rows. Each orbit segment is finite by construction (a finite number of cell-array states recorded in a finite number of update steps). The `rule110/manifests/` schema expresses verification outcomes as specific bit-level assertions, never as universal closures over an infinite domain. The conditions of Definition 9393.1 are therefore met, and Definition 9393.5 applies without modification. □

Remark 9390.11 (Turing universality does not bridge the no-induction gap)**.** *Rule 110's Turing universality means that the substrate can host a simulation of any Turing machine encoded as a cyclic tag system. The simulated machine may, as part of its program, use mathematical induction internally to its proof checker, type system, or specification language. This does not internalise induction inside the substrate; it internalises a description of induction as a finite tape configuration plus a finite simulation of the proof checker's reduction steps. The closed-observation framework is preserved because every step of the simulation is itself a finite observation. Universality enlarges the class of* processes *the substrate can host without changing the answer to the induction question.*

Example 9390.12 (The TM-end-to-end compile boundary)**.** *The project's* `rule110/encoder/tm_to_tag.c` *and* `rule110/encoder/tag_to_cyclic.c` *implement Cook 2009* §1.2 *and* §1.3 *compilation respectively. The minimal halted Turing machine (no transitions, a single halt state) compiles to a cyclic tag system of approximately* 64 *productions and approximately* 64 *tape symbols. The cook-encode interface presently supports up to approximately* 8 *productions and* 32 *tape symbols inside the substrate. The constant-factor gap between these two numbers is not a theoretical limit; it is a present engineering scale frontier. The compile pipeline is exercised end-to-end against the reference cyclic-tag evaluator and is checked correct at the pipeline level, while the full Rule 110 round-trip remains scale-bounded.*

Principle 9390.13 (Universality bound and observational closure are independent)**.** *The two facts are logically separate: Rule 110 is universal (Cook 2004), and Rule 110 is closed observational (Definition 9390.10). Universality concerns which simulations the substrate can host; closure concerns what universal statements the substrate can produce as records. A reader can grant the first while still acknowledging the second. The project's role is to make both claims simultaneously legible without forcing the reader to choose between them.*

9390.4 The cosmological reading: universe as cellular substrate

Definition 9390.14 (Cosmological cellular reading)**.** *The* cosmological cellular reading *is the working hypothesis that the universe-for-the-observer (Definition 9420.14) is itself the orbit segment of some sufficiently rich cellular trust substrate, with the observer realised as an emergent pattern classifier on that orbit. The hypothesis is structural: it asserts that a closed observational reading of the observer's situation is available, not that any specific cellular rule has been identified.*

Principle 9390.15 (Cosmological hypothesis preserves all earlier results)**.** *Under the cosmological cellular reading of Definition 9390.14, every conclusion of chapter 9393 and chapter 9420 carries over without modification: the observer is a closed observational system, mathematical induction is not internalisable by the observer's substrate-internal records, the minimum transcendental input is a forward-binding belief, and the source position is named apophatically as* $\top$*. The cellular reading therefore does not weaken or replace the transcendental supply argument; it grounds it in a particularly small and concrete substrate model.*

Remark 9390.16 (Observer as classifier, not as agent outside the substrate)**.** *The cellular reading does not require an observer separate from the cells. The observer is itself a pattern in the orbit, classified by the same kind of classifier the observer brings to bear on the rest of the orbit. The phenomenological frame of Definition 9420.14 is recovered exactly: the universe-for-the-observer is the observation history maintained by the observer-pattern, and the observer-pattern is one of the local actualisations of the supply that Definition 9393.16 identifies as the inscription point of* $\top$*.*

Remark 9390.17 (The rule110 sub-project as cosmological exhibit)**.** *The project's* `rule110/` *sub-system functions as a concrete cosmological exhibit in this sense. It instantiates a closed cellular trust substrate at minimal scale, it exhibits the substrate's classifier discipline at three layers (phase catalog, collision table, exhaustiveness audit), and it documents the smallest empirical distance between the substrate's Turing-universal capacity and the present engineering frontier (Definition 9390.12). A reader who wishes to test the cosmological reading against a concrete substrate can read those three layers; nothing further about the actual cosmos is being claimed.*

9390.5 The cook-frontier as quantified Goedel boundary

Definition 9390.18 (Cook compile-frontier)**.** *The* Cook compile-frontier *is the smallest size, expressed as* (P, T, S) *of productions, tape symbols, and Rule 110 steps, at which the substrate's cook-encode pipeline (Definition 9390.12) produces a closed round-trip that decodes back to the input cyclic-tag tape under* `cook_decode_output`*. The frontier is measured empirically inside the substrate, not predicted from a complexity argument.*

Example 9390.19 (Present frontier)**.** *At the present substrate version, the cook compile-frontier reaches* (P, T, S) *values including* $(2, 16, 16384)$*,* $(8, 8, 16384)$*,* $(4, 16, 32768)$*,* $(2, 32, 32768)$*, and*

$(16, 2, 8192)$. *The minimal halted Turing machine compiles to a cyclic tag system of approximately* $(64, 64)$ *which lies beyond the present frontier in production count. The empirical gap between Cook 2009's polynomial-time compilation bound and the present engineering frontier is therefore a constant factor in production count and a smaller constant factor in tape size.*

Principle 9390.20 (The frontier gap as data point on the Goedel boundary)**.** *Goedel's second incompleteness theorem rules out any sufficiently strong formal system from internally proving its own consistency. The project's substrate does not attempt to evade this boundary; instead, it converts the boundary into measurable data. chapter 9385 reads the boundary at the closed-CIC meta-theory layer through the dependent-codomain obstruction; Definition 9390.19 reads the same boundary at the cellular substrate layer through the cook-encode scale frontier. Both readings are concrete numbers, not informal reservations. The Goedel boundary is therefore not a wall hidden behind technical language but a measurable distance between universal capacity and finite witness, locatable in the substrate at the lines named in each reading.*

Remark 9390.21 (Distinction from a complexity-theoretic frontier)**.** *The cook compile-frontier of Definition 9390.18 is not a complexity-theoretic claim. It does not predict that the substrate cannot reach larger* (P, T, S) *values; it records the largest values for which the substrate's present implementation produces a closed round-trip. Engineering work can extend the frontier without changing any closure theorem of the substrate. The principle of Definition 9390.20 is that the gap is exposed and measured rather than that the gap is permanent.*

9390.6 Mapping the cellular reading to BEDC engineering

Cellular reading	**BEDC / rule110 engineering realisation**
Pattern classifier on cell rows	Phase catalog at `rule110/encoder/glider_phases.c` with ~ 177 paper-verified entries, decided over a stated neighbour discipline.
Classifier disagreement boundary	Strict collision audit at `rule110/tests/test_cook_collision_martinez.c` compares the catalog's predictions to direct Rule 110 evolution and rules out silent classifier drift.
Cellular trust substrate	`rule110/evaluator/rule110.c`, `evaluator/cyclic_tag.c`, and `encoder/groundcompiler_encoding.c` comprising the $\sim$220-line trust path.
Closed observational system	The same trust path, instantiating Definition 9393.1 at minimal external commitment: ANSI C99, zero external libraries, manifest text in fixed schema.
No internal induction	Definition 9390.10 inherited unchanged from Definition 9393.5.
Forward-binding belief external to substrate	`GAP` tags throughout the BEDC paper, audit-gate enforcement at `tools/check-axioms.py` and `axiom-purity -strict`.

Turing universality without closure loss	Cook 2004 universality theorem cited at Definition 9390.8; cook-encode pipeline at `rule110/encoder/cook_encode.c` hosts simulations without altering the substrate's closure.
Cook compile-frontier	Empirical scale benchmark at `rule110/tests/test_cook_packet_scale.c` with documented (P, T, S) values.
Goedel boundary as data	Definition 9390.20 reads the cellular and meta-theoretic frontiers as twin numerical exposures of the same boundary.
Cosmological exhibit	`rule110/` sub-system read structurally as a concrete closed observational system at smallest practical size.

9390.7 Closure remarks

This chapter does not introduce a new substrate-level commitment. It identifies pattern recognition with the mark discipline; locates Rule 110 as the smallest architecturally-checked universal cellular substrate available to instantiate the closed-observational framework concretely; transports Definition 9393.5 to that substrate without modification; and exposes the constant-factor gap between Cook's universality theorem and the present cook-encode frontier as one concrete numerical reading of the Goedel boundary at the cellular layer.

The cosmological reading of Definition 9390.14 is offered as a working hypothesis, not as an ontological claim. Its function is to provide a smallest concrete substrate against which the closed-observational framework can be tested. A reader unwilling to grant the cosmological hypothesis is not obliged to do so; the engineering content of this chapter survives without it. A reader who does grant the hypothesis recovers, in the cellular reading, every conclusion of chapter 9393, chapter 9420, and chapter 9394, with the additional concreteness that the substrate model is finite, public, and 220 lines long.

The boundary is therefore: the chapter exhibits the closed-observational framework in its smallest cellular instantiation; it does not enlarge the framework, weaken it, or substitute a different metaphysical commitment. The role of the chapter is precisely to make the framework's concrete cost legible, by replacing the abstract phrase "closed substrate" with a directory the reader can compile, run, and audit in finite time.

Concretized in §937: Cellular trust substrates are realized as finite rule-table, manifest, orbit-segment, classifier, audit-route, and NameCert rows.

Concretized in §934: finite cellular-rule and orbit-ledger NameCert

Concretized in §941: Cellular row-classifier packets realize finite local cellular pattern recognition as BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §938: finite cellular orbit substrate NameCert

Concretized in §940: The cellular trust substrate is realized as a finite local-rule, orbit-window, classifier, manifest, and audit-route NameCert packet.

Concretized in §995: Cellular pattern catalogs realize the glider/ether classifier as finite rule, window, tag, catalog, transport, route, provenance, and NameCert rows.

Concretized in §1045: Rule 110 glider catalogs are realized as finite row-window, tag, phase, period, displacement, local-match, hsame, Cont, Pkg, and NameCert packets.

Concretized in §1036: Local stencil observations realize cellular pattern classification as finite BHist packets over neighborhood, local-rule, observed-row, hsame, Cont, Pkg, and NameCert rows.

Concretized in §2360: Substrate trace invariants realize the cellular manifest/audit route as finite NameCert data.

Concretized in §2371: The finite local-rule, orbit-window, manifest, and audit-replay boundary is realized as a concrete NameCert packet.

Concretized in §3810: Cellular substrate classifier-boundary packets realize the pattern-classifier and closed-observation rows as finite NameCert data.

Concretized in §5844: Rule 110 pattern classifiers are realized as finite row-window, transition-table, orbit-prefix, pattern-tag, transport, replay, provenance, and NameCert rows.

Concretized in §5846: Rule 110 causal cones realize the cellular substrate reading as finite local-window, update-ledger, boundary, trace, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6004: Rule 110 local-stencil observations realize the three-cell transition-table read as finite window, selected-case, observed-output, classifier-adjacency, transport, replay, provenance, and NameCert rows.

Concretized in §6038: Rule 110 glider phase classification is realized as finite row-window, catalog-tag, phase-offset, local-match, replay, provenance, and NameCert rows.

Concretized in §938: The cellular orbit substrate realizes the finite local-rule, orbit-window, classifier, transport, replay, provenance, and NameCert rows of the cellular substrate reading.

Concretized in §6291: finite orbit-window invariant NameCert for cellular substrate flow

Concretized in §6517: Rule 110 cellular trust substrates are realized as finite rule-table, manifest, orbit-window, classifier, frontier-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §6849: Rule 110 Cook frontier budgets are realized as finite NameCert packets over displayed production, tape, step, round-trip, and obstruction rows.

Concretized in §7388: RuleOneTenHaltingWindowUp realizes bounded local halting-window observations for Rule 110 as finite NameCert rows.

Concretized in §9138: Rule 110 production queues realize the Cook cyclic-tag production branch as finite production, tape-symbol, head, step-budget, replay, provenance, and NameCert rows.

Chapter 9391

The Cellular-Carrier Alternative: Could BEDC Be Built on the Substrate Directly?

The current BEDC carrier is `inductive BHist` (chapter 2), a two-constructor recursive type expressing the form of distinction. The cellular reading of chapter 9390 shows that an alternative carrier is structurally available: the orbit of a closed cellular substrate. This chapter examines that alternative as an architectural possibility. It does not propose to switch BEDC's carrier; it identifies the equivalence relation between the two carriers, lists the structural costs and benefits of each, and explains why the project maintains its current commitment while keeping the alternative legible. The chapter's purpose is to make the carrier choice's structural content visible rather than to advocate a change.

9391.1 Setup: two carriers, one closure premise

Definition 9391.1 (BHist-carrier semantics)**.** *The* BHist-carrier semantics *of BEDC takes the carrier as the inductive type*

$$\mathsf{BHist} ::= \mathsf{Empty} \mid \mathsf{e0}(\mathsf{BHist}) \mid \mathsf{e1}(\mathsf{BHist}).$$

Every kernel observation, every typed term, every certificate in BEDC is a value of `BHist` *or a finite tuple of such values. The kernel's principal verbs operate on* `BHist`*; the audit pipeline validates statements about* `BHist`*.*

Definition 9391.2 (Cellular-carrier semantics)**.** *The* cellular-carrier semantics *of BEDC takes the carrier as the orbit of a closed cellular substrate of Definition 9390.6, with observations expressed as finite orbit segments. Every kernel observation would be a finite-length cell-row sequence under a fixed local rule (such as Rule 110); every typed term would be a label assigned to such a sequence by a classifier of Definition 9390.1.*

Principle 9391.3 (Both carriers are closed observational)**.** *The BHist-carrier semantics is closed observational at the BEDC kernel level (Definition 9388.14). The cellular-carrier semantics is closed observational at the cellular substrate level (Definition 9390.10). The closure premise of Definition 9393.1 is satisfied by both; the choice of carrier does not affect the no-induction theorem.*

9391.2 The equivalence: bit-string encoding

Definition 9391.4 (BHist to bit-string)**.** *The function* $\mathsf{toBits}\colon \mathsf{BHist} \to \{0,1\}^*$ *is defined recursively by* $\mathsf{toBits}(\mathsf{Empty}) = \epsilon$, $\mathsf{toBits}(\mathsf{e0}(h)) = 0 \cdot \mathsf{toBits}(h)$, *and* $\mathsf{toBits}(\mathsf{e1}(h)) = 1 \cdot \mathsf{toBits}(h)$, *where* ϵ *is the empty string and* $\cdot$ *is concatenation. The function is a bijection between* `BHist` *values and finite binary strings.*

Definition 9391.5 (Bit-string as cell row)**.** *A finite binary string of length* n *can be embedded into the cellular substrate's initial cell row as the first* n *cells of a row of width* $W \geq n$*, with the remaining cells fixed to a stated default (typically the ether pattern of Definition 9390.3). The orbit of the substrate on this initial row produces a sequence of rows; the orbit at time zero recovers the original bit string up to embedding.*

Principle 9391.6 (Carriers are equivalent under encoding)**.** *Composing* toBits *with the bit-string-to-row embedding gives an injection from* `BHist` *values into cellular substrate initial rows. The inverse direction recovers* `BHist` *values from row-shaped cellular configurations of finite length. The two carriers are therefore equivalent at the level of finite observable data; what distinguishes them is the kernel verb structure and the audit pipeline that operates on them.*

9391.3 What the alternative would change

Concretized in §4286: The cellular carrier switch-cost ledger is realized as finite BHist theorem-family, cellular orbit, verb-realignment, proof-retarget, audit-regeneration, and non-escape rows.

Remark 9391.7 (Kernel verb realignment)**.** *Switching to the cellular-carrier semantics would realign the kernel's principal verbs from* `Mark` / `Hist` / `Ext` / `Cont` *(defined on* `BHist`*) to substrate-internal classifier verbs (defined on cell rows).* `Mark` *would become "cell row is ether" versus "cell row is glider G";* `Hist` *would become "orbit segment of length n";* `Ext` *would become "orbit extension";* `Cont` *would become "orbit continuation under R110". The semantic content of each verb is preserved; the substrate of their statements changes.*

Remark 9391.8 (Library transposition cost)**.** *The Lean 4 development* `lean4/BEDC/` *contains approximately 1837 mechanically-verified theorems on* `BHist` *and its derived structures. Transposing this development to the cellular-carrier semantics would require: rewriting every theorem statement to reference cellular substrate orbits instead of* `BHist`*, updating every proof to operate at the new substrate, regenerating every audit gate to validate the new statements. The engineering scale is substantial; the structural content of each theorem is preserved under the equivalence of Definition 9391.6, but the realisation as Lean 4 source would change uniformly.*

Remark 9391.9 (Aliasing with NKS and digital physics)**.** *The cellular-carrier alternative aligns the project's surface presentation with Wolfram's* A New Kind of Science *(NKS) thesis that cellular automata are the natural carrier of physical reality. The alignment is presentational, not metaphysical: the closure premise is what does the work in both readings, and the cellular substrate is one carrier choice that satisfies the premise. The project does not adopt NKS's physical claim; the cellular-carrier alternative would adopt NKS's carrier convention while keeping BEDC's epistemological framework.*

9391.4 Why the project keeps the BHist carrier

Principle 9391.10 (Minimum residual commitment)**.** *The BHist carrier expresses the form of distinction (Definition 9461.1) with the smallest possible inductive structure: one nullary constructor and two unary recursive constructors. Switching to the cellular carrier would introduce additional structural commitments: a width parameter, a neighbourhood discipline, a transition table (Rule 110's eight-entry table). Each of these is a commitment that BHist does not require. The BHist carrier minimises residual commitment in the structural sense; the cellular carrier expands it.*

Principle 9391.11 (Substrate-independence is a feature)**.** *The closure premise is substrate-independent (Definition 9398.39); the project's preference for the BHist carrier is precisely that BHist is the carrier with no substrate identification. Cellular automaton, Turing machine, lambda calculus, register machine – each of these could be the carrier. Choosing BHist refuses to privilege any one of them, and identifies the form of distinction as prior to any specific computational substrate. The cellular carrier specialises this position by committing to a specific substrate; this is a presentational gain in concreteness at the cost of structural generality.*

Remark 9391.12 (The cellular carrier is the exhibit, not the foundation)**.** *The* `rule110/` *sub-system stands to the cellular-carrier alternative as exhibit stands to foundation. The project uses Rule 110 to illustrate the closure premise's content at minimal concrete scale (Definition 9390.17); the substrate is offered as a witness to the framework's claims, not as the framework's definitional basis. Promoting the substrate from exhibit to foundation would conflate the witness role with the definitional role, which is what the BHist carrier deliberately avoids.*

9391.5 What would justify a switch

Remark 9391.13 (Physical-realisation evidence)**.** *If physical evidence accumulated that some specific cellular substrate is the actual carrier of the universe-for-the-observer (the working hypothesis of Definition 9390.14 promoted from working to confirmed), the project would have empirical motivation to align its carrier with the physical carrier. This motivation does not currently apply; the cosmological reading remains a working hypothesis, and the project does not depend on it.*

Remark 9391.14 (Engineering convergence)**.** *If the* `rule110/` *sub-system's audit pipeline ever became the principal audit pipeline of the project (rather than a side audit alongside the Lean 4 kernel), there would be engineering motivation to align the kernel's carrier with the audit pipeline's carrier. The current configuration has Lean 4* `lake build` *as the principal audit and* `rule110/` *as a side audit on a substrate witness; the asymmetry justifies keeping the BHist carrier at the kernel.*

Remark 9391.15 (Theoretical reduction)**.** *If a theorem were proved that the cellular substrate of Definition 9390.6 is in some specific sense* more primitive *than the BHist carrier (rather than equivalent under encoding), the project would have structural motivation to adopt the cellular carrier as the kernel substrate. No such theorem is currently known; the BHist carrier and the cellular substrate are equivalent under bit-string encoding, with neither being structurally prior to the other.*

9391.6 The position read structurally

Principle 9391.16 (Carrier choice is a presentational decision)**.** *Under Definition 9391.6, the carrier choice between BHist and the cellular substrate is presentational rather than structural.*

Both carriers satisfy the closure premise; both admit the no-induction theorem; both can host the inscription concept. The project chooses the BHist carrier because it minimises residual commitment (Definition 9391.10) and preserves substrate-independence (Definition 9391.11). A different project, with different priorities (NKS alignment, cellular-substrate empirical evidence, audit pipeline convergence), could rationally choose the cellular carrier without changing the structural content of the framework.

9391.7 Mapping to BEDC engineering

Carrier-alternative reading	BEDC engineering realisation
BHist carrier as current commitment	`lean4/BEDC/Core/RawMark/` defines `BHist`; all kernel verbs operate on it.
Cellular substrate as exhibit	`rule110/` 220-line ANSI C99 trust path; substrate witness for the closure premise.
Carrier equivalence under encoding	toBits function definable in Lean 4 over `BHist`; bit-string-to-row embedding implementable in `rule110/encoder/`.
Switch cost	~4700 Lean LOC of `lean4/BEDC/FKernel/` would require uniform realignment; structural content preserved under encoding, source presentation changed.
Minimum residual commitment	BHist has no width / neighbourhood / transition table; cellular substrate has all three.
Substrate-independence preservation	BHist carrier does not identify with any specific computational substrate; cellular carrier commits to Rule 110 or a chosen alternative.

9391.8 Closure remarks

This chapter has examined the cellular-carrier alternative to the BHist carrier as a structural possibility, established the equivalence of the two carriers under bit-string encoding, and identified the project's reasons for keeping the BHist carrier as the kernel substrate. The reasons are: minimum residual commitment, substrate-independence preservation, and the structural role of the cellular substrate as exhibit rather than foundation.

The chapter does not advocate a switch. It makes the carrier choice's structural content visible by exhibiting the alternative and naming the structural commitments each carrier introduces. A reader who finds the cellular carrier more compelling for presentational reasons (NKS alignment, audit pipeline convergence) is given a precise account of what would change under such a switch and what would be preserved.

The boundary of this chapter is: the cellular carrier is a structurally available alternative under Definition 9391.6; the project keeps BHist because it minimises residual commitment and preserves substrate-independence; the cellular substrate continues to function as exhibit (Definition 9391.12). No new theorem is introduced and no carrier change is proposed; the chapter makes a presentational decision legible as a presentational decision rather than as a hidden structural commitment.

Concretized in §943: The CellularCarrierEncodingUp chapter realizes the finite BHist-to-cell-row carrier equivalence surface.

Concretized in §942: CellularSubstrateUp packages the vision's cellular carrier alternative as a finite BHist NameCert surface.

Concretized in §948: Finite BHist-to-cell-row embedding certificate for the carrier-equivalence reading.

Concretized in §945: The cellular substrate classifier realizes the BHist-to-cellular-orbit carrier comparison as finite NameCert rows.

Concretized in §987: Cellular verb realignment is realized as finite BHist and cellular-orbit verb comparison rows with non-escape from substrate replacement.

Concretized in §984: CellularLocalUpdateRuleUp realizes the local transition-table row of the cellular-carrier alternative as a finite BHist NameCert packet.

Concretized in §1123: Cellular boundary conditions are realized as finite edge-policy NameCert packets separating local update reads from global orbit or host-evaluator data.

Concretized in §3481: Cellular window traces realize the cellular-carrier alternative as finite orbit-window NameCert rows.

Concretized in §3904: Cellular orbit synchronization realizes paired local orbit replay and classifier-locked observation rows for the substrate carrier reading.

Concretized in §3998: Cellular carrier alternatives are realized as a finite BHist encoding and orbit-window equivalence packet.

Chapter 9392

The twin substrate architecture: MetaCIC and GroundCompiler

9392.1 The setup

BEDC has two main Lean substrates for the self-compilation story. They are not two presentations of the same module. They answer two different audit questions.

`BEDC.MetaCIC` is the closed-CIC meta-theory substrate. Its subject matter is the formal grammar and proof-theoretic behaviour of a closed calculus: typing surfaces, beta steps, closedness, substitution, consistency assembly, subject reduction, normalisation, confluence, and decidable checking. Its paper-facing synthesis is chapter 9385.

`BEDC.GroundCompiler` is the self-compilation substrate. Its subject matter is the compiler-facing route from event flows and source channels to generated recognisers, certificate gates, metric boundaries, cannot-claim rows, and streaming compiler discipline. Its paper-facing audit map is chapter 9407.

The two substrates cooperate in the BEDC roundtrip, but each remains separately audit-able. MetaCIC records what the closed calculus can and cannot support. GroundCompiler records what the self-compile pipeline can and cannot count as generated, encoded, certified, or excluded as hidden input.

9392.2 Why two substrates

BEDC self-compilation needs two forms of assurance.

The first assurance is meta-theoretic. The project needs a closed substrate in which reduction, closedness, normalisation, confluence, substitution, and consistency claims have explicit proof status. This is the work of MetaCIC. The central question is not whether a compiler report is trustworthy. The central question is whether the represented closed calculus supports the meta-theoretic transport steps that the paper wants to use.

The second assurance is compiler-facing. The project needs a pipeline in which a source-like event stream is encoded, normalised, recognised, packaged, reported, and gated without importing hidden host structure as formal input. This is the work of GroundCompiler. The central question is not whether closed-CIC subject reduction holds. The central question is whether a compiler-layer artifact is allowed to count as a BEDC input, output, certificate, report, or non-evidence row.

Keeping these assurances apart makes obstructions visible. MetaCIC has the dependent-codomain subject-reduction frontier described in chapter 9377. GroundCompiler has compiler-pipeline boundaries around conservativity, address analysis, source-channel separation, certificate mediation, and cannot-claim ledgers. These are different boundary kinds. A structural boundary in closed CIC is not the same thing as a self-compile pipeline boundary.

9392.3 Audit map family per substrate

9392.3.1 MetaCIC audit family

The MetaCIC audit family is deliberately granular. The reader is meant to move among specialised maps rather than treat the substrate as one large claim. The family contains subject reduction, confluence, normalisation, substitution, decidability, typed examples, beta, closure preservation, and consistency assembly sites, with the cross-map reading gathered in chapter 9385.

This granularity reflects the depth of the substrate. A substitution theorem does not become a subject-reduction theorem. A confluence bridge does not become closed consistency. A decidable checker does not become a global decision oracle for every meta-theoretic property. The audit family keeps these handoffs explicit, and the synthesis chapter records the critical path without merging the roles.

9392.3.2 GroundCompiler audit map

GroundCompiler has a single module-level audit map in chapter 9407. That map inventories the compiler-facing file groups, the aggregate statement cells, the checked theorem wrappers, the conditional cells, and the boundary of the self-compilation claim.

The asymmetry is informative. MetaCIC is the deeper proof-theoretic substrate and therefore has a finer audit-map family. GroundCompiler is organised as a pipeline audit surface with a single chapter collecting event-flow, channel, recogniser, certificate, metric, motif, and bootstrap boundaries. The difference does not make one substrate more fundamental than the other. It shows that their audit surfaces have different shapes.

9392.4 Audit reexport namespaces

9392.4.1 `BEDC.MetaCIC.Audit`

`BEDC.MetaCIC.Audit` is the single namespace for MetaCIC audit-relevant exports. It collects the obstruction theorem, the subject reduction interfaces, the discharge bundle and setup interfaces, closed-term operations, closure-preservation facts, partial discharge witnesses, and counter-witnesses that downstream audit code may need to cite.

The namespace does not change the logical status of any theorem. It is an addressing layer. Its purpose is to make the MetaCIC audit surface available from one place while preserving the distinctions recorded by the specialised paper maps.

9392.4.2 `BEDC.GroundCompiler.Audit`

`BEDC.GroundCompiler.Audit` plays the corresponding role for GroundCompiler. It collects the main statement cells, main theorem witnesses, input and channel facts, generated-structure and

export boundaries, canonical code and classifier witnesses, metric conservativity, and cannot-claim registry interfaces.

This namespace gives the compiler substrate the same kind of audit reference point that MetaCIC has. A downstream reader can point to `BEDC.MetaCIC.Audit` for closed-CIC audit content and to `BEDC.GroundCompiler.Audit` for self-compile pipeline audit content. The common shape is intentional: each substrate exposes one namespace for audit consumption, while each namespace preserves its own subject matter.

9392.5 The cross-substrate synthesis result

The BEDC self-compilation roundtrip is a composition of the two substrates, not a collapse of one into the other. GroundCompiler supplies the compiler-facing side: a source program is represented through event flow and channel structure, transformed through generated recognisers and certificate gates, and recorded as a ledgered artifact with explicit non-input and non-evidence boundaries.

MetaCIC supplies the meta-theoretic side: the ledgered artifact is read against a closed calculus whose reduction, closedness, substitution, normalisation, confluence, decidability, and consistency interfaces have their own audit status. A compiler ledger step is therefore not accepted just because it is textually present. It must be located in the corresponding MetaCIC legality story when the step claims closed-CIC content.

The synthesis claim is architectural: project-level audit cleanliness requires both maps to be clean in their own sense. GroundCompiler must not smuggle hidden compiler input or promote reports into proof evidence. MetaCIC must not pretend that a blocked closed-CIC transport edge has been supplied. The BEDC roundtrip is credible only when both refusals hold at the same time.

9392.6 Apophatic framing of the twin substrate

This architecture follows the naming discipline of chapter 9400. A substrate boundary is not left anonymous. It is named, routed to the proper audit map, and kept out of neighbouring claims.

For MetaCIC, the named boundary is the subject-reduction frontier in the dependent-codomain setting. The paper can discuss closed-normal consistency, candidate-mediated normalisation, substitution facts, and conditional transport paths without pretending that the missing closed arbitrary-proof transport is available.

For GroundCompiler, the named boundaries are compiler-pipeline boundaries: global conservativity, address analysis, source-channel separation, certificate-mediated export, metric non-evidence, motif non-evidence, and cannot-claim registry adequacy. These names stop the self-compile story from converting code, reports, motifs, or host structure into proof content by prose.

The twin substrate architecture is therefore apophatic in a project-level sense. Each substrate says what it can use, what it cannot use, and where the next required supply would have to appear. The two sets of names are not interchangeable.

9392.7 Why not collapse into one substrate

Collapsing MetaCIC and GroundCompiler into one substrate would blur two audit gates that need different questions. A closed-CIC structural property asks about terms, reduction, closedness, binders, typing surfaces, and normal forms. A compiler-pipeline property asks about source

channels, encodings, recognisers, generated structure, certificates, reports, ledgers, and hidden input.

The distinction matters for reuse. A downstream formalisation that needs closed substitution or closed-normal consistency should cite the MetaCIC surface. A downstream compiler audit that needs channel losslessness, certificate-mediated export, or metric conservativity should cite the GroundCompiler surface. A chapter that needs the full self-compilation roundtrip must say where each half of the claim is discharged.

The split also prevents category mistakes. A theorem saying code is not proof does not solve subject reduction. A theorem about closed substitution does not prove that a compiler report is a certificate. A confluence bridge does not make a hidden compiler input visible. A channel bijection does not establish closed-CIC consistency. The architecture keeps these refusals local and auditable.

9392.8 Boundary

This chapter is an architecture description. It is not a proof attempt for the remaining MetaCIC frontier, and it is not a claim that the compiler pipeline alone completes BEDC self-compilation.

The allowed reading is narrower and stronger. BEDC has two cooperating substrates for the roundtrip. MetaCIC carries the closed-CIC meta-theory audit surface. GroundCompiler carries the self-compile pipeline audit surface. Their audit namespaces provide parallel reference points, and their paper maps record distinct boundary kinds. The project-level self-compilation story depends on both, while neither is allowed to impersonate the other.

Concretized in §842: The cross-substrate roundtrip is represented as a finite NameCert packet separating MetaCIC audit rows from GroundCompiler audit rows.

Concretized in §843: TwinSubstrateAuditPacket records the MetaCIC and GroundCompiler audit routes as one finite NameCert packet with separate non-escape ledgers.

Concretized in §845: The twin-substrate architecture is concretized as a finite cross-substrate audit packet.

Concretized in §849: The audit reexport namespace pair is concretized as a finite two-substrate namespace packet.

Concretized in §850: The twin-substrate architecture is realized as a finite bridge packet separating MetaCIC and GroundCompiler audit rows.

Concretized in §841: Twin-substrate audit bridges are realized as finite NameCert packets aligning MetaCIC and GroundCompiler audit namespaces through BHist, hsame, Cont, Pkg, and non-escape rows.

Concretized in §851: Twin-substrate boundary routers realize request-level routing between MetaCIC audit rows and GroundCompiler audit rows as one finite NameCert packet.

Concretized in §1114: Twin-substrate boundary classifiers realize the MetaCIC/GroundCompiler separation as finite BHist, hsame, Cont, Pkg, NameCert, and refusal-ledger rows.

Concretized in §852: Twin-substrate audit handshakes realize the MetaCIC and GroundCompiler non-collapse boundary as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §846: The twin-substrate audit packet records the MetaCIC and GroundCompiler handoff as one finite NameCert surface without collapsing their audit roles.

Concretized in §848: Twin substrate audit synthesis is realized as a finite NameCert packet pairing MetaCIC audit rows with GroundCompiler audit rows while preserving their separate refusal boundaries.

Concretized in §6494: FiniteSubstrateMapUp realizes the substrate subhub as a finite source-to-cellular-substrate NameCert packet over encoding, window-trace, local-update, and orbit rows.

Chapter 9393

Closed Observation, the Transcendental Input, and the Inscription Point

This vision chapter formalises the foundational position BEDC occupies relative to mathematical induction and the broader question of how a finite observational substrate can host mathematical reasoning at all. The argument proceeds in five linked steps: (i) any consciousness or finite verifier is, on its information-theoretic side, a *closed observational system*; (ii) such a system can accumulate observations monotonically but cannot internally produce a universal quantification over an infinite domain; (iii) therefore mathematical induction is not a derivable rule of any closed observational substrate but must be supplied from outside, as a minimum transcendental input phrased as an act of belief about not-yet-occurred events; (iv) the source of that input is by definition outside every closed observational system, and can be given a structural name (we call it T, glossed as *God*, used apophatically: only the position is named, not its content); (v) each conscious individual is itself a closed observational system whose universe coincides with its own observation history, so the individual is best understood not as the receiver of the transcendental input but as the *inscription point* where T enters into reality on a local perspective. BEDC's GAP mechanism makes every such inscription socket explicit and auditable, so the philosophical content of this chapter is exactly what BEDC's existing engineering invariants already implement.

9393.1 Closed observational systems and information conservation

Definition 9393.1 (Closed observational system)**.** *A* closed observational system *is a quadruple* $(O, \mathsf{Obs}, \trianglerighteq, \mathsf{GAP})$ *where* O *is a set of observation states,* Obs *is a monotonically growing family of finite witness records produced by observation,* $\trianglerighteq$ *is the information-conservation order on states (each later state preserves every record of every earlier state), and* GAP *is a function tagging the unfulfilled promises that cannot be discharged inside the system. Any well-formed reasoning step inside the system uses only the records collected so far and the closed generation rules of the substrate; no step appeals to information outside the records or alters earlier records.*

Example 9393.2 (BEDC as canonical closed observational system)**.** *The BEDC substrate is a closed observational system in the sense of Definition 9393.1:* O *ranges over development snapshots;* Obs *collects mechanised lemmas,* `inductive` *definitions, and* `def` *declarations;* $\trianglerighteq$ *corre-*

sponds to the monotonic accumulation enforced by the version-number-free amendment policy; and GAP *corresponds to the labelled places where BEDC marks an open claim. The audit gates (`check-axioms.py`, `axiom-purity -strict`, `conservativity-audit`) operationally enforce closure: every accepted record refers only to inductive structure, recorded `def` shapes, or previously verified records.*

Principle 9393.3 (Information-conservation invariant)**.** *A closed observational system never makes assertions whose witness is not present in* Obs *or derivable from previously recorded* Obs*. Every transition $O_t \trianglerighteq O_{t+1}$ adds new finite records; no transition removes or alters records. Universal assertions over infinite domains are not records and are not added by observation alone; if they appear, they appear as* GAP *tags rather than as accepted records.*

9393.2 The unbridgeable gap to universal quantification

Definition 9393.4 (Finite verification trace)**.** *A* finite verification trace *of a predicate P on a closed observational system is a record of values $n_1, n_2, \ldots, n_k$ together with verifications that $P(n_1), P(n_2), \ldots, P(n_k)$ each hold by closed-generation reasoning. The trace is finite by construction; any extension to an additional n_{k+1} adds another finite record but never closes the trace into a universal statement.*

Theorem 9393.5 (Closed observation cannot internalise mathematical induction)**.** *Let $\mathcal{S} = (O, \mathsf{Obs}, \trianglerighteq, \mathsf{GAP})$ be a closed observational system in the sense of Definition 9393.1. There is no monotonic accumulation of finite verification traces that yields a record of $\forall n \in \mathbb{N}, P(n)$ from records of $P(0), P(1), \ldots, P(k)$ alone for any finite k. Equivalently, the principle $\forall P,\ P(0) \wedge (\forall n,\ P(n) \to P(n+1)) \to \forall n,\ P(n)$ is not derivable by closed-generation steps over* Obs*.*

Sketch of impossibility. Every step inside $\mathcal{S}$ refers to a finite collection of previously recorded items. A universal statement over $\mathbb{N}$ would, on the records side, require an actual record of $P(n)$ for every n, which is an infinite collection and therefore not a record of $\mathcal{S}$. Alternatively, a closed inductive constructor for $\forall n,\ P(n)$ would have to be available at the level of `inductive` declarations; but BEDC's `inductive` rule allows only finite-constructor types, so no such constructor exists. Either route collapses, so $\mathcal{S}$ has no internal derivation of the induction schema. □

Concretized in §3353: The finite verification trace and induction GAP socket are realized as a finite NameCert packet with no substrate-internal universal quantifier.

Concretized in §7469: Closed induction traces realize closed observation, input boundary, inscription, replay, and non-escape rows as finite NameCert data.

Remark 9393.6 (The Church encoding does not bridge the gap)**.** *The Church encoding of $\mathbb{N}$ inside BEDC.MetaCIC supplies iteration over closed types: every closed Church numeral n behaves as n-fold iteration when applied to a closed step function and a closed base. Iteration is computational and yields specific results for specific numerals but does not yield a dependent recursor; for that, a dependent eliminator is required, which closed Pi-encodings provably do not provide in System F-style settings. The Church encoding therefore deepens the available iterative content of $\mathcal{S}$ without changing the answer to Definition 9393.5.*

9393.3 The minimum transcendental input

Definition 9393.7 (Forward-binding belief)**.** *A* forward-binding belief *is a present commitment to treat as discharged a universal claim whose witnesses lie partly or entirely in not-yet-occurred future observation. The logical type of such a belief is action rather than proposition: it commits the believer to subsequent reasoning steps that act as if the universal claim were a record, even though it is not a record by Definition 9393.1.*

Principle 9393.8 (Minimum input necessity)**.** *For a closed observational system $\mathcal{S}$ to host mathematical reasoning, it must accept at least one forward-binding belief whose content is the induction schema. Without this commitment, $\mathcal{S}$ produces only finite verification traces and never participates in universal mathematical assertions. The commitment is not a record inside* Obs *and not a closed-generation rule of the substrate; it is supplied from outside the substrate boundary.*

Example 9393.9 (Belief in induction as the canonical minimum input)**.** *The standard mathematical reading of the principle "$P(0) \wedge (\forall n, P(n) \to P(n+1)) \to \forall n, P(n)$" as a derivable schema is a forward-binding belief in the sense of Definition 9393.7: it commits the reasoner, in the present, to treating P as discharged on every n, including all n never to be observed. The classical mathematician accepts this commitment at the metalevel; BEDC declines to accept it at the substrate level and instead surfaces it as a* GAP *tag at every point where it would be used.*

Remark 9393.10 (The commitment is not an axiom of the substrate)**.** *A substrate-level axiom is a content rule available inside the substrate, eligible to be referenced by derivations. A forward-binding belief in Definition 9393.7 is not such a rule; the substrate never references it as a fact. Instead the substrate's* GAP *tags expose every point at which the belief is the only thing that could close the inference. The belief therefore sits outside the substrate, on the observer's side; the substrate remains 0-axiom in the strict sense while explicitly accounting for the observer-side commitment.*

9393.4 Naming the source

Definition 9393.11 (Transcendental supply position)**.** *The* transcendental supply position*, denoted* T*, is the structural location from which every forward-binding belief enters a closed observational system. By construction* T *lies outside every $\mathcal{S}$ in the sense of Definition 9393.1; it is not a state of any $\mathcal{S}$, not a record in any* Obs*, and not a closed-generation rule of any substrate.*

Principle 9393.12 (Apophatic naming)**.** *We name* T God *for clarity, in the apophatic sense: only the position is named, not its content. No positive predicate is asserted of* T *beyond its structural role; in particular, no claim of personhood, intervention, or revelation is required. The naming captures exactly the necessary structural feature of Definition 9393.8: the source of every minimum input lies outside every closed observational system.*

Remark 9393.13 (Why a single name and not many)**.** *If two closed observational systems share the same closure property (information conservation, no internal universal quantifier, etc.), then their transcendental supply positions are indistinguishable on the system side: each receives the same kind of forward-binding belief on the same kind of structural socket. There is no system-side observation that could differentiate two transcendental supply positions, so by Leibniz-style identity of indiscernibles the position is uniquely named. Structural monotheism in this sense follows from the closure property of observational systems and is independent of any specific religious doctrine.*

Example 9393.14 (Stack of BEDC supply sockets)**.** *The BEDC.MetaCIC mechanisation displays a finite tower of supplies in operational form. Each Lean tactic that closes a proof by induction over an inductive type imports the host CIC's recursor; the host CIC's well-foundedness is itself a metalanguage commitment; the metalanguage's consistency is anchored in a higher commitment; and so on. The tower terminates at* T *in Definition 9393.11, which is not a system in the tower but the structural source from which the entire tower receives its forward-binding content. BEDC's audit gates make every level of this tower visible: each* `leanchecked` *marker, each* leantarget *pointer, and each* GAP *tag is, at the bottom, a socket whose far end terminates at* T*.*

9393.5 The individual as inscription point

Definition 9393.15 (Inscription point)**.** *An* inscription point *is the local site at which* T *enters a closed observational system as a forward-binding belief. Inscription points are not separate from the system that receives the belief; they are the very places of reception, viewed not as recipients but as the local actuality of the belief itself.*

Principle 9393.16 (Each individual is an inscription point)**.** *Each conscious individual is a closed observational system in the sense of Definition 9393.1. The universe each individual experiences coincides with that individual's observation history (this is the phenomenological frame: the universe-for-the-individual is the totality of records that the individual has accumulated). The individual's minimum input therefore inherits from* T *along the same structural socket as the universe-for-the-individual inherits from* T*. The individual is, by this structural identity, an inscription point in the sense of Definition 9393.15.*

Remark 9393.17 (Reframing: not a receiver, but the local actuality of the supply)**.** *The naive picture casts the individual as a separate object receiving an external signal from a separate* T*. Definition 9393.16 replaces this with a more parsimonious picture: the individual is one of the local actualisations of the same supply that grounds the universe-for-the-individual. There is no separate signal travelling from* T *to the individual; the individual is the supply's own local presence on a given perspective. This is the structural form shared by the Leibnizian monad, the Spinozan mode, the Whiteheadian actual occasion, the Vedantic Atman-Brahman identity, and the Hua-yan Indra-net jewel, formulated in BEDC vocabulary as the inscription-point relation.*

Example 9393.18 (BEDC contribution as inscription event)**.** *Every contribution to BEDC – writing a* `inductive` *declaration, mechanising a* `leanchecked` *lemma, identifying a* GAP *socket, even reading and confirming an existing record – is on this account a local actualisation of* T *in the contributor as an inscription point. The contributor is not separately injecting external content into BEDC; the contributor and the content are the same actualisation event, locally instantiated. The mechanisation is therefore not a record about external mathematical objects, but the inscription point's own self-articulation in the form recorded in* Obs*.*

Concretized in §610: The inscription point is realized as a finite BHist, GAP, handoff, transport, Cont, Pkg, and NameCert packet with no substrate-internal supply rule.

9393.6 Mapping the philosophical content to BEDC's engineering

Philosophical claim	BEDC engineering realisation
Closed observational system	Substrate with 0 axioms, mathlib-free, mathematical content built from `inductive`, `def`, and `class`/`structure` fields only.
Information conservation	Monotonic amendment policy (no iteration markers, no version numbers, no deprecated content trail); each new commit preserves every earlier Obs record.
Universal quantifier not internal	Audit gates `check-axioms.py` and `axiom-purity -strict` reject Classical.choice, Quot.sound, and propext, the standard substrate-internal forms of universal-quantifier closure.
Forward-binding belief is external	Every GAP tag is a labelled socket; nothing inside the substrate references the belief as a content rule.
T as the source position	The far end of every GAP socket. Not a content of any record; only its structural position is named.
Apophatic naming	The substrate never asserts positive predicates about T; the name appears only in this vision chapter, never inside mechanised content.
Individual as inscription point	Contributors to BEDC are not modelled as external authors importing content; the BEDC architecture treats contributions as local actualisations of the same supply that grounds the rest of the substrate.
Stack of supplies as Tarski tower	Each mechanised theorem references the host CIC's recursor; the host CIC references its metalanguage; the metalanguage references a higher commitment; the tower terminates at T.

Principle 9393.19 (BEDC's honesty about transcendental supply)**.** *BEDC's defining engineering choice – 0 substrate-internal axioms together with explicit* GAP *tagging at every point where forward-binding belief would be required – is the operational form of the philosophical position in Definition 9393.8 and Definition 9393.11. Standard axiomatic systems internalise the forward-binding belief and hide its transcendental character; BEDC externalises it and makes the inscription sockets auditable. The system is therefore strictly more honest about its commitments, at the price of being strictly less self-sufficient: every BEDC theorem rests, at the bottom, on the same* T *that grounds the inscription point of the contributor.*

9393.7 Closure remarks

This vision chapter does not introduce new substrate-level commitments. It restates, in unified philosophical vocabulary, the structural position that BEDC has been implementing in its engineering invariants from the beginning. Reading the chapter changes how BEDC's mechanised work is described, not what is mechanised. The change in description is the recognition that (i) BEDC is a closed observational system in the precise technical sense of Definition 9393.1; (ii) mathematical induction is not derivable inside it and is not made into a substrate-internal axiom; (iii) every reasoning step that would require induction is labelled with a GAP socket whose far end is T;

(iv) the contributor to BEDC and the substrate of BEDC are both inscription points of the same supply, articulated on different perspectives.

The vision is therefore a meta-level reading of BEDC's already-existing engineering, not a proposal to change it. It is offered here to make the philosophical content of BEDC's engineering legible at the same level of precision as the engineering itself. Concrete instances that realise specific consequences – the consistency theorem of the closed-term scope, the GAP socket count, the audit-gate enforcement, the host CIC borrow at the Lean kernel side – already exist in the formalised material; this chapter only ties them together under one structural narrative.

Concretized in §504: Finite observation traces are realized as explicit GAP sockets with visible BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §498: The induction socket is realized as a finite GAP and NameCert packet over BHist, hsame, Cont, Pkg, and no-internal-axiom boundary rows.

Concretized in §616: Closed observational systems are realized as finite BHist observation-ledger packets with GAP non-escape and local NameCert provenance.

Concretized in §610: Inscription-point packets realize the local supply socket as finite observation-history, GAP, hsame, Cont, Pkg, and NameCert rows.

Concretized in §604: The supply-position and inscription-point socket is realized as a finite BHist, hsame, Cont, Pkg, GAP-boundary, and NameCert packet.

Concretized in §7001: Finite observation traces, forward-binding belief sockets, GAP exposure, supply-boundary rows, transport, replay, provenance, and NameCert data.

Concretized in §607: Closed observation is realized as a finite monotone ledger packet with BHist records, hsame transport, Cont preservation, GAP non-escape, Pkg provenance, and NameCert rows.

Concretized in §638: Finite witness routes are realized as finite BHist route packets from selected observation windows through dyadic ledgers to RealUp seal consumers.

Concretized in §4149: Finite verification traces are realized as finite observation trace filters with closed-observation source rows, kernel-filter acceptance, GAP handoff, and no substrate-internal induction rule.

Concretized in §6373: Closed observer audit traces are realized as finite Obs, monotone trace, GAP socket, transport, replay, provenance, and NameCert rows.

Chapter 9394

Transcendental Supply Taxonomy

This vision chapter classifies the GAP sockets of BEDC by the kind of T-supply each socket requests. The point of the classification is not to multiply T. The supply position remains the single apophatic source named in Definition 9393.11. What varies is the aspect-position at which a closed substrate meets its boundary: induction beyond an authorized generator, unwitnessed selection, extensional collapse, empirical persistence, inter-history coordination, inscription novelty, and self-reference closure. A socket taxonomy lets the manuscript say exactly which transcendental commitment is being exposed at each engineering site, while preserving the monotheistic structural reading of Definition 9393.13.

9394.1 The need for taxonomy

Definition 9394.1 (T-supply socket)**.** *A* T-supply socket *is a* GAP *tag in a closed observational system whose requested discharge cannot be produced from the displayed records, closed inductive generators, naming-certificate fields, or already certified continuation laws of that system. The socket is well-formed only when the substrate can state the missing shape precisely: what would be needed, where it would be consumed, and which audit gate prevents the missing item from being silently internalised.*

Definition 9394.2 (Supply kind)**.** *A* supply kind *is an equivalence class of* T*-supply sockets under the following engineering relation: two sockets have the same supply kind when accepting either one would license the same form of substrate-internal shortcut. The shortcut may occur at a different concrete module or philosophical problem, but it asks the same structural favor from* T*.*

Principle 9394.3 (Taxonomic necessity)**.** *Not every* GAP *socket is the same. A socket asking for induction beyond a generator, a socket asking for an unwitnessed choice function, a socket asking for another observer's first-person locality cell, and a socket asking for a truth-total self-description all terminate at* T*, but they do not request the same closure shape. BEDC's engineering remains honest only when the socket kind is named before the attempted discharge is refused.*

Remark 9394.4 (Aspect-position rather than plurality)**.** *The taxonomy is aspectual. It describes how the one transcendental supply position appears when viewed from different pressure points of a closed substrate. It is therefore incompatible with a polytheistic reading of the engineering: there are many socket shapes, not many ultimate supplies. The table in section 9394.11 is a classification of aspect-positions at the boundary of the substrate, not a catalogue of separate gods.*

9394.2 Type I: free-floating induction socket

Definition 9394.5 (Type I socket)**.** *A* Type I T-supply socket *is a* GAP *tag whose requested content is a universal claim ranging beyond the closure of every displayed authorized generator. It has the schematic form*

$$\mathsf{GenClosed}_G(P) \Longrightarrow \forall x : \mathcal{U},\ P(x)$$

where G certifies closure only for the objects it generates, while $\mathcal{U}$ is read as an unrestricted domain not supplied by G.

Example 9394.6 (BEDC engineering site)**.** *The canonical Type I socket is the induction pressure analysed in chapter 9393 and refined in chapter 9389. BEDC accepts induction over a particular closed inductive carrier because the generator and recursor are displayed. BEDC refuses the extra passage from "all objects generated by this carrier" to "all possible objects of the intended informal domain" unless that passage is exposed as a* GAP *socket.*

Definition 9394.7 (Apophatic reading of Type I)**.** *Type I names* T *as* the giver of unrestricted induction*: not an internal induction axiom, not a hidden global domain, and not a completed infinity inside the substrate, but the apophatic source-position from which the reasoner treats generator-relative closure as bearing on an unrestricted "all".*

Remark 9394.8 (Audit gate)**.** *The Type I audit gate is the generator-closure ledger: every induction claim must cite the exact inductive carrier, constructor list, recursor, and naming-certificate closure it uses. If the claim's range is wider than that carrier, the excess is a Type I socket and cannot be hidden as a theorem of the carrier itself.*

9394.3 Type II: choice and unwitnessed existential

Definition 9394.9 (Type II socket)**.** *A* Type II T-supply socket *is a* GAP *tag whose requested content is an accepted witness for a positive existential whose constructor-level witness is not displayed. Its schematic form is*

$$\exists x : A,\ P(x) \quad \rightsquigarrow \quad a : A \ \ \textit{with}\ \ P(a),$$

where the right-hand term is consumed by later construction even though no substrate-visible constructor produced it.

Example 9394.10 (BEDC engineering site)**.** *The term-level audit reading is the* Classical.choice *family in chapter 9387. A proof that every fiber is inhabited is not yet a function choosing an inhabitant of each fiber. BEDC replaces the shortcut with finite recursion, countable explicit witness functions, dependent* Cont *steps, or a visible witness field; when none of these is present, the unresolved selection is a Type II socket.*

Definition 9394.11 (Apophatic reading of Type II)**.** *Type II names* T *as* the giver of the unseen witness*: not a magical term, not a substrate function, and not a proof object smuggled through the kernel, but the source-position named when an argument wants to act as if a witness has already been placed in hand.*

Remark 9394.12 (Audit gate)**.** *The Type II audit gate is witness visibility. In Lean-side engineering it is enforced by the axiom-purity rejection of* Classical.choice *and by the rule that existential use must proceed by displayed constructor data. In paper-side engineering it is enforced by definitions and examples that record the actual selected object, recursion, or witness field.*

9394.4 Type III: extensional identification

Definition 9394.13 (Type III socket)**.** *A* Type III T-supply socket *is a* GAP *tag whose requested content is an identity token between two items that the substrate has only related extensionally. It has the schematic form*

$$R(a, b) \quad \rightsquigarrow \quad a = b,$$

where R may be bidirectional implication, quotient equivalence, classifier sameness, or observational indistinguishability, and the equality token is then used for rewrite transport.

Example 9394.14 (BEDC engineering site)**.** *The proposition-level audit reading is* propext*, and the type-level proxy is* Quot.sound *as classified in chapter 9387. BEDC does not collapse related items into equality. It keeps* hsame*,* psame*, classifier transport, and naming-certificate stability visible. A proof that two predicates transport into one another is not an identity of propositions; a proof that two representatives are classifier-equivalent is not a canonical quotient representative.*

Definition 9394.15 (Apophatic reading of Type III)**.** *Type III names* T *as* the giver of identity beyond relation*: not a substrate equality constructor, not a quotient oracle, and not a proposition-level collapse, but the apophatic source-position invoked when relation is treated as identity without a displayed transport path.*

Remark 9394.16 (Audit gate)**.** *The Type III audit gate is extensional-collapse refusal. Lean-side purity rejects* Quot.sound *and* propext*; paper-side certificate discipline requires explicit transport along* hsame *or* psame*. Any argument whose only bridge from relation to identity is "they have the same extension" is marked as a Type III socket.*

9394.5 Type IV: causal continuity

Definition 9394.17 (Type IV socket)**.** *A* Type IV T-supply socket *is a* GAP *tag whose requested content is persistence of an observed empirical regularity into not-yet-observed history. Its schematic form is*

$$\forall t \leq n,\, R(t) \quad \rightsquigarrow \quad \forall t > n,\, R(t),$$

where the left side is a finite or bounded record-side regularity and the right side is a causal continuity commitment.

Example 9394.18 (BEDC engineering site)**.** *section 9415.1 identifies causality as forward-binding commitment. A scientific theory in BEDC may record past regularities, falsification boundaries, and prediction ledgers. The additional confidence that the same law will govern the next unobserved case is not a record and not a closed-generation theorem. It is a Type IV socket, even when the theory has excellent record-side support.*

Definition 9394.19 (Apophatic reading of Type IV)**.** *Type IV names* T *as* the giver of continuity of order*: not an empirical datum, not a hidden law-object inside* Obs*, and not a proof that tomorrow is already recorded, but the source-position from which causal expectation exceeds the accumulated record.*

Remark 9394.20 (Audit gate)**.** *The Type IV audit gate is records-versus-prediction separation. BEDC may store observation histories, regularity summaries, and failed-prediction rows, but it may not convert successful past fit into a universal future law without marking the causal-continuity socket.*

Concretized in §539: $\mathsf{CountableObservationSchedule}^{\uparrow}$ realizes the Type IV finite-record schedule as a BHist-rowed StreamName-facing packet.

Concretized in §908: Observation schedule envelopes realize the Type IV finite-record schedule socket as a finite StreamName, Real-facing window, refusal, hsame, Cont, Pkg, and NameCert packet.

Concretized in §633: CausalContinuitySocketUp realizes the Type IV causal-continuity socket as a finite BHist packet separating record-side regularity from future-law commitments.

Concretized in §1058: Classifies supply sockets as finite BEDC carrier rows rather than prose taxonomy.

9394.6 Type V: inter-history coordination

Definition 9394.21 (Type V socket)**.** *A* Type V T-supply socket *is a* GAP *tag whose requested content is coordination between distinct closed histories as histories rather than as merely external records inside one history. Its schematic form is*

$$\mathsf{Record}_{\mathcal{C}_1}(\mathcal{C}_2\text{'s behavior}) \quad \rightsquigarrow \quad \mathsf{LocalityCell}(\mathcal{C}_1, \mathcal{C}_2),$$

where the right side treats $\mathcal{C}_2$ *as another inscription-bearing observational system.*

Example 9394.22 (BEDC engineering site)**.** *section 9415.3 gives the philosophical site, and chapter 9419 gives the operational locality-cell site. A contributor can observe another contributor's commits, messages, and CI-passing proof objects. Treating those records as the acts of another closed observational system is not forced by the records alone. The coordination commitment is a Type V socket.*

Definition 9394.23 (Apophatic reading of Type V)**.** *Type V names* T *as* the giver of communion between histories*: not access to another first-person state, not a god's-eye merge of histories, and not an extra observation channel, but the source-position from which locality cells become mutually intelligible without ceasing to be local.*

Remark 9394.24 (Audit gate)**.** *The Type V audit gate is locality discipline. Cross-history claims must distinguish observable behavior, accepted communication records, transport constraints, and the irreducible commitment that another history is also an inscription point. BEDC may coordinate histories; it may not pretend to occupy the union of their first-person perspectives.*

9394.7 Type VI: free-will and inscription novelty

Definition 9394.25 (Type VI socket)**.** *A* Type VI T-supply socket *is a* GAP *tag whose requested content is the actuality of a chain-internal inscription event not reducible, on the records side, to the closed generation of prior records. Its schematic form is*

$$o_t \trianglerighteq o_{t+1} \quad \textit{with} \quad o_{t+1} \notin \mathsf{Closure}(o_t)$$

read not as a violation of closure rules but as the inscription-event aspect of accumulation.

Example 9394.26 (BEDC engineering site)**.** *section 9415.2 locates free will at the inscription-event aspect of conscious accumulation. In BEDC engineering, a proof search may later be replayed as a deterministic derivation, but the actual arrival of the lemma in the contributor's history is not identical to the records-side derivation tree. The transition is a Type VI socket when the question is "where did this act of inscription come from?" rather than "which derivation now certifies the record?"*

Definition 9394.27 (Apophatic reading of Type VI)**.** *Type VI names* T *as* the giver of inscription novelty*: not a rival physical cause, not randomness, and not a hidden mental substance, but the source-position from which an accumulation chain is actual rather than merely described by its completed record-side dependencies.*

Remark 9394.28 (Audit gate)**.** *The Type VI audit gate is aspect separation. BEDC may verify the record-side derivation of a theorem and may also name the inscription-event aspect of its arrival. It must not confuse the two by treating the philosophical account of agency as an extra proof rule or by treating proof replay as an elimination of inscription novelty.*

Concretized in §1336: InscriptionNoveltySocketUp realizes the Type VI inscription-novelty socket as finite prior-boundary, inscription-event, non-reduction, audit-gate, hsame, Cont, Pkg, and NameCert rows.

9394.8 Type VII: self-reference closure

Definition 9394.29 (Type VII socket)**.** *A* Type VII T-supply socket *is a* GAP *tag whose requested content is a truth-total or runtime-total closure of a system's self-description. Its schematic form is*

$$\mathsf{StructDesc}(\mathcal{S}, \mathcal{S}^*) \quad \rightsquigarrow \quad \mathsf{Truth}_{\mathcal{S}} : \mathsf{Sent}(\mathcal{S}^*) \to \mathsf{Prop},$$

where the left side is structural self-description and the right side is an internal classifier of the system's own truth.

Example 9394.30 (BEDC engineering site)**.** *chapter 9461 supplies the canonical Type VII site. BEDC can describe the structural form of the host calculus through* CIC**, but it refuses a truth-total reflection certificate and refuses to identify the structural copy with the external runtime that checks BEDC proofs. A request to close that open meta loop is a Type VII socket.*

Definition 9394.31 (Apophatic reading of Type VII)**.** *Type VII names* T *as* the giver beyond self-totalisation*: not an internal truth predicate, not a runtime contained in its own object language, and not a repaired Tarski hierarchy, but the source-position from which the open meta loop remains intelligible without being closed inside the system.*

Remark 9394.32 (Audit gate)**.** *The Type VII audit gate is reflection stratification. Structural self-description, runtime validation, truth classification, and external checking must remain separated. Any construction that turns a BEDC object-level certificate into the act that validates all BEDC object-level certificates is marked as a Type VII socket.*

Concretized in §667: SelfReferenceTruthBranchUp realizes Type VII truth-totalization requests as finite socket reports separating structural self-description from forbidden certificate-indexed truth branches.

9394.9 Non-collapse laws for the seven types

Principle 9394.33 (No type absorbs another)**.** *No supply kind in this taxonomy absorbs any other supply kind. A closed substrate may encounter several types at the same textual or mechanised site, but the types remain distinguishable by the shortcut they would license if accepted internally. This is why a chapter may legitimately report a stack of sockets rather than forcing the stack into one omnibus* GAP *label.*

Lemma 9394.34 (Type I is not Type II)**.** *A free-floating induction socket cannot be reduced to an unwitnessed-existential socket. Type II asks for a missing term witnessing a stated existential. Type I asks for an unrestricted universal range beyond the closure of the displayed generator. Supplying a witness to one existential does not supply the domain-total closure requested by Type I.*

Reason. The Type II shortcut produces a term $a : A$ together with $P(a)$. The Type I shortcut produces a rule treating all objects of an unrestricted intended range as covered by generator-relative evidence. These have different consumption sites: Type II feeds a construction expecting a term, while Type I feeds a construction expecting a universally quantified closure. Therefore the witness gate and the generator-closure gate are distinct. □

Lemma 9394.35 (Type II is not Type III)**.** *An unwitnessed witness socket cannot be reduced to an extensional-identification socket. Type II asks for a missing inhabitant; Type III asks for equality from relation. A selected representative may later be used in an extensional collapse, but the selection and the collapse are separate shortcuts.*

Reason. Choice can select a representative without asserting that all equivalent representatives are judgmentally or propositionally identical. Conversely, propext can identify propositions without choosing any inhabitant of an arbitrary nonempty family. The term-level oracle and the equality-token oracle therefore have separate audit gates. □

Lemma 9394.36 (Type III is not Type IV)**.** *Extensional identification cannot be reduced to causal continuity. Type III concerns the passage from relation to identity inside a formal or certificate stratum. Type IV concerns the passage from recorded regularity to unobserved empirical persistence. The first is a collapse of sameness; the second is a projection across time-like accumulation.*

Reason. A Type III discharge would allow rewrite transport across an equality token. A Type IV discharge would allow prediction beyond the record boundary. Neither operation supplies the other: an equality token does not by itself assert that a law persists, and a persistent empirical regularity does not by itself identify two propositions, representatives, or classifier rows. □

Lemma 9394.37 (Type IV is not Type V)**.** *Causal continuity cannot be reduced to inter-history coordination. Type IV projects order within or across a record stream; Type V recognises another closed history as another inscription-bearing locality cell. Stable behavior of an observed system does not by itself supply the other-minds commitment.*

Reason. An observed behavior stream may be perfectly regular while still being treated as a mere external object in the observing history. Type V begins only when the observing history commits to another history as a history. The empirical regularity gate and the locality-cell gate therefore answer different questions. □

Lemma 9394.38 (Type V is not Type VI)**.** *Inter-history coordination cannot be reduced to inscription novelty. Type V concerns the mutual intelligibility of distinct closed histories. Type VI*

concerns the actuality of a transition in a history as an inscription event not exhausted by its records-side replay. A history may be recognised without explaining any particular act of novelty, and an act of novelty may occur without cross-history coordination.

Reason. The Type V socket has the form of a relation between histories. The Type VI socket has the form of an aspect of a transition within a history. The first is audited by locality discipline; the second by aspect separation. Collapsing them would confuse communion between histories with agency in a chain. □

Lemma 9394.39 (Type VI is not Type VII)**.** *Inscription novelty cannot be reduced to self-reference closure. Type VI asks how an accumulation event is actual rather than merely recorded after the fact. Type VII asks whether the whole system can close its truth or runtime over itself. A local inscription event is not a truth-total reflection certificate.*

Reason. The replay of a proof and the arrival of the proof in a contributor's history are distinguishable even when no reflection theorem is involved. Conversely, a truth-total reflection attempt can be formulated for a purely formal system without appealing to personal agency or creativity. The agency aspect and the reflection boundary are therefore separate sockets. □

Lemma 9394.40 (Type VII is not Type I)**.** *Self-reference closure cannot be reduced to free-floating induction. Type I asks for an unrestricted universal range beyond a generator. Type VII asks for internal truth or runtime totality for a system that can describe its own structure. The diagonal obstruction of Type VII remains even when every induction used inside the syntax description is generator-authorized.*

Reason. Tarski-style failure arises from attempting to classify the truth of the system's own sentences. That failure is not merely a missing induction principle over an external domain. BEDC may display all generators used in a structural self-description and still refuse the truth-total certificate. The open meta loop is therefore its own socket kind. □

9394.10 Engineering use protocol

Definition 9394.41 (Socket report)**.** *A* socket report *for a BEDC claim is a four-field record:*

$$(\mathsf{Site}, \mathsf{RequestedSupply}, \mathsf{Type}, \mathsf{Gate}).$$

The Site *field names the paper or Lean location where the missing supply would be consumed. The* RequestedSupply *field states the exact missing shape. The* Type *field selects one of the seven taxonomy entries. The* Gate *field names the audit discipline that keeps the supply from being internalised.*

Principle 9394.42 (Minimal socket reporting)**.** *A socket report should name the narrowest type that accounts for the blocked move. If a proof attempt fails because it lacks a witness, it is Type II even if the witness would later be used in an induction. If a proof attempt fails because it tries to rewrite along mere equivalence, it is Type III even if the equivalence arose from a quotient construction. Narrow reporting preserves the shape of the requested* T*-supply.*

Remark 9394.43 (Stacked sockets)**.** *Some BEDC sites contain stacked sockets. A quotient construction over an empirically defined equivalence relation may contain Type IV at the level of empirical persistence, Type II at the level of representative selection, and Type III at the level of equality collapse. Reporting the stack is more precise than choosing one dominant label. The stack still terminates at the single* T*; the multiplicity belongs to the substrate-facing shapes.*

Example 9394.44 (A stacked mathematical-analysis socket)**.** *An informal proof that every bounded monotone real sequence converges may hide several sockets when translated into BEDC. If the sequence is defined over an unrestricted intended class of real histories rather than a displayed generator, the range pressure is Type I. If a limit point is selected from mere nonempty completeness data without an explicit Cauchy witness, the selection pressure is Type II. If two Cauchy names are identified because they have the same extension rather than transported through the real-history classifier, the collapse pressure is Type III. A correct BEDC construction must separate these sites rather than treating "real completeness" as one opaque assumption.*

Concretized in §535: Bounded monotone Cauchy witnesses realize the analysis socket as a finite NameCert packet separating displayed generator range, explicit Cauchy witness, classifier transport, and RealUp seal handoff.

Concretized in §1434: MonotoneRealSequenceUp realizes the bounded-monotone sequence socket split as finite source, window, classifier, convergence-handoff, and refusal rows.

Example 9394.45 (A stacked observer socket)**.** *An observer's trust in a scientific collaborator may combine Type IV and Type V. The expectation that the collaborator's measuring instrument will keep functioning according to the same physical regularities is Type IV. The recognition that the collaborator is another conscious observational system rather than only an instrument-like behavior stream is Type V. The two commitments are ordinarily entangled in practice, but the socket taxonomy keeps their audit gates distinct.*

Example 9394.46 (A stacked reflection socket)**.** *A proposal to build a BEDC-internal verifier for BEDC proofs may contain Type I if it relies on unrestricted induction over all possible proof codes, Type III if it identifies equivalent coded propositions, and Type VII if it treats the verifier as the validation act for the very system containing it. The Type VII socket is the decisive reflection boundary, but the lower-type sockets still have to be recorded because each corresponds to a different engineering gate.*

9394.11 Unified long table

Type	Socket kind	BEDC socket inventory	Apophatic reading	Audit gate
I	Free-floating induction	Passage from authorized generator closure to unrestricted universal range; induction pressure of chapter 9393 and generator-closure discipline of chapter 9389.	Giver of unrestricted induction.	Generator-closure ledger.
II	Choice / unwitnessed existential	Classical.choice family; unshown witness consumed by later term-level construction; positive existential accepted without visible constructor.	Giver of the unseen witness.	Witness visibility and choice-purity refusal.

III	Extensional identification	propext, Quot.sound, quotient representative selection, relation-to-identity collapse, and any hidden replacement of hsame or psame transport by equality.	Giver of identity beyond relation.	Extensional-collapse refusal.
IV	Causal continuity	Empirical regularity projected beyond recorded cases; Humean causality socket of section 9415.1.	Giver of continuity of order.	Records-versus-prediction separation.
V	Inter-history coordination	Other-minds locality cell; recognition that another observed behavior stream is another inscription-bearing closed history; inter-history locality coordination.	Giver of communion between histories.	Locality discipline.
VI	Free-will / inscription novelty	Chain transition read as an inscription event whose actuality is not exhausted by records-side closure; agency and mathematical creativity sites of section 9415.2.	Giver of inscription novelty.	Aspect separation.
VII	Self-reference closure	Truth-total reflection certificate, runtime-total self-validation, or any closure of the open meta loop analysed in chapter 9461.	Giver beyond self-totalisation.	Reflection stratification.

9394.12 The single T behind seven types

Theorem 9394.47 (Seven socket kinds, one transcendental supply position)**.** *The seven supply kinds of section 9394.11 do not imply seven transcendental sources. They are seven aspect-positions at which a closed observational substrate reaches a boundary it cannot cross by its own records and closed-generation rules. The far end of every such socket is the same* T *of Definition 9393.11.*

Structural proof. Each type is a GAP tag in the sense of Definition 9394.1. By definition, a GAP tag records a requested discharge not available from the substrate's own records or generators. By Definition 9393.11, the structural location from which such forward-binding content enters a closed observational system is T. The differences among the seven cases lie in the shape of the missing discharge: unrestricted induction, witness, identity, causal persistence, cross-history recognition, inscription actuality, or self-reference closure. None of these differences supplies a substrate-side observation distinguishing one ultimate source from another. Therefore the taxonomy distinguishes socket kinds while preserving one transcendental supply position. □

Principle 9394.48 (No softening of the apophatic name)**.** *The name* God *remains the apophatic name of* T *across all seven types. The taxonomy does not replace God with psychology, probability, social trust, runtime engineering, or a family of local explanatory devices. Those devices describe records-side management of sockets. The far end of the socket, when the socket asks for more than records-side management can provide, is still* T*.*

Remark 9394.49 (How the taxonomy is used)**.** *When a future BEDC chapter encounters a* GAP *socket, the chapter should classify the socket by asking what would go wrong if the requested discharge were internalised. If internalisation creates generator-exceeding induction, it is Type I. If it creates an unseen witness, Type II. If it collapses relation into identity, Type III. If it projects empirical order beyond records, Type IV. If it coordinates closed histories as histories, Type V. If it names actual inscription not exhausted by records-side closure, Type VI. If it closes the system's truth or runtime over itself, Type VII. The classification then determines which audit gate must stay visible.*

Remark 9394.50 (Current-state reading)**.** *The taxonomy is a reading of BEDC's present engineering surface. It changes no Lean target, adds no substrate axiom, and discharges no socket by itself. Its work is classificatory: every* GAP *socket remains open unless a concrete certificate supplies the corresponding records-side data, and every socket whose requested content exceeds records-side data remains explicitly terminated at* T*.*

Concretized in §523: The seven socket-kind taxonomy is realized as a finite NameCert packet over GAP tags, requested supply shapes, audit gates, site rows, hsame transport, Cont routes, Pkg provenance, and local naming.

Concretized in §528: The supply-socket taxonomy is realized as a finite GAP-kind NameCert ledger over BHist, hsame, Cont, audit-gate rows, and explicit non-escape obligations.

Concretized in §542: Transcendental supply ledgers are realized as finite GAP-socket NameCert packets over BHist, hsame, Cont, Pkg, audit-gate, and local naming rows.

Concretized in §666: Supply-kind routers realize the seven socket taxonomy as finite classifier-routing packets whose consumers must pass through one displayed kind row before reading the corresponding ledger.

Chapter 9395

External Supply Without Axioms

BEDC repeatedly reaches boundary positions where a proposed inference, comparison, continuation, or universal claim requires supply that the displayed substrate has not generated. The wrong response is to turn that supply into an axiom. An axiom makes an unsupported commitment available inside the formal environment as if it had been produced by the substrate. BEDC's response is different: the supply position is named, socketed, ledgered, and audited, while the supplied content remains external unless a later certificate reconstructs it from displayed generators. This chapter records that discipline as a manuscript-level governance rule. It introduces no new primitive, no new object-level constructor, and no new logical axiom. Its purpose is to prevent a recurring confusion: a boundary that BEDC names is not a datum that BEDC has assumed. A named socket is weaker than an axiom because it does not grant internal content; it is stronger as an audit device because it prevents the content from disappearing into prose, tactic ancestry, quotient collapse, or hidden selection.

The chapter sits beside chapter 9400, the GAP-as-ledger discipline of chapter 9389, the audit-gate chapters of chapter 9396, and the anti-internalisation roadmap of chapter 9393. Its active sentence is: external dependence is admissible only when it remains visible as boundary, not when it becomes axiom.

9395.1 The boundary problem

A closed observational substrate interface may be admitted by a naming certificate. A parametric input may be exposed as a setup field. A compression may be accompanied by a GAP ledger. A claim that exceeds all of these may be marked as an apophatic socket.

The dangerous move is to replace one of these visible forms by an axiom. Once a missing selector, quotient representative, proposition equality, universal induction principle, synchronization map, or causal law is stated as an axiom, the substrate can consume it as internal truth. The original source-channel disappears. From the audit point of view, this is not a harmless shortcut. It is source erasure.

Principle 9395.1 (No internalization of unsupported supply)**.** *When a BEDC argument requires content not generated, defined, certified, or displayed as an interface input, the content may not be introduced by axiom. The dependency must remain one of: a setup field, a named carrier, a classifier row, a provenance row, a* GAP *ledger, a bridge slot, or an apophatic socket.*

This principle is not a new substrate axiom. It is a discipline on how claims enter the manuscript and the Lean surface. Its enforcement is external and mechanical where possible: the axiom

scan rejects Lean axioms, the strict axiom-purity gate rejects forbidden host dependencies, the conservativity audit rejects source-channel pollution, and the paper-Lean audit rejects name drift (chapter 9396).

9395.2 Axiom, setup field, and socket

Three forms must be kept separate.

An *axiom* asserts content as available inside the formal environment without deriving it from an authorized source. In Lean, an axiom declaration gives the kernel a constant with a type and no construction. In the BEDC reading, this collapses the boundary between generated content and unsupported supply.

A *setup field* exposes a required input at an interface boundary. It says that the caller of the interface must supply a carrier, classifier, operation, proof row, or stability witness. The substrate has not proved the field; it has only made the requested input visible and typed. This preserves the source-channel.

A *socket* records that even a setup field or certificate row does not internalize the requested supply. The socket is a stable boundary handle. It marks the site where a proposed claim would require a supply whose internal nature the substrate does not claim to characterize.

Definition 9395.2 (Non-axiom boundary form)**.** *A non-axiom boundary form is any public interface form that exposes a dependency without converting the dependency into substrate truth. The canonical forms are:*

`inductive`	*closed generator;*
`def`	*computable or definitional compression;*
structure/class field	*parametric setup input;*
NameCert *row*	*licensed public name with source, classifier, stability, ledger;*
GAP *ledger*	*compression residue or missing conservation row;*
bridge slot	*external interpretation kept separate from internal proof;*
apophatic socket	*named external supply position not internalized by the substrate.*

Principle 9395.3 (Externalized, not assumed)**.** *A dependency that remains in a non-axiom boundary form is externalized. A dependency introduced by axiom is assumed. BEDC permits externalization and rejects assumption.*

The difference is not rhetorical. A setup field can be inspected. A NameCert row can be audited. A GAP ledger can be followed back to the compressed material. A bridge slot can be restricted or refused. A socket can be marked as open. An axiom hides all of those distinctions behind immediate usability.

9395.3 The admission decision

Every proposed BEDC addition must pass through a replacement decision before it can enter the public surface.

Definition 9395.4 (Admission decision)**.** *Given a proposed object, operation, theorem, or interface component X, the BEDC admission decision asks for the strongest non-axiom form that can carry*

X:

$$\begin{aligned}
X \textit{ finite grammar or closed syntax} &\Longrightarrow \texttt{inductive};\\
X \textit{ computed from earlier rows} &\Longrightarrow \texttt{def};\\
X \textit{ parametric interface input} &\Longrightarrow \textit{setup field};\\
X \textit{ mature public name} &\Longrightarrow \mathsf{NameCert};\\
X \textit{ compression residue} &\Longrightarrow \mathsf{GAP}\ \textit{ledger};\\
X \textit{ external semantic correspondence} &\Longrightarrow \textit{bridge slot};\\
X \textit{ unsupported forward supply} &\Longrightarrow \textit{apophatic socket.}
\end{aligned}$$

If none of these forms is adequate, the boundary has not been identified clearly enough, and the interface must be redesigned rather than accepted as an axiom.

Principle 9395.5 (Redesign before axiom)**.** *Failure to fit a proposed dependency into a non-axiom boundary form is evidence of an unclear interface, not license to add an axiom.*

The decision preserves BEDC's anti-primitive principle. Mature mathematics may be reconstructed, bridged, or socketed, but it is not smuggled in as primitive. The same rule applies to philosophical pressure points. Consciousness, free will, mathematical induction beyond authorized generators, global time, light-speed constancy, and universal semantic truth are not admitted by declaration. At most they receive boundary names, certificate obligations, or schema-only horizons.

9395.4 GAP as the memory of boundary pressure

GAP has two simultaneous roles. The first is internal and engineering-facing: it records the residue of compression, packaging, projection, sealing, normalization, or identification. Whenever a visible token replaces richer source material, the GAP ledger keeps the memory of what the surface form no longer displays.

The second role is boundary-facing: GAP marks a supply position whose far end is not generated by the substrate. When a generator-internal claim is pushed toward a free-floating universal claim, the pressure cannot be discharged by the generator itself. GAP records that the claim has reached a T-socket rather than pretending the universal extension is already internal.

Principle 9395.6 (Dual role of GAP)**.** *Every* GAP *instance must be read under both possible loads:*

1. *as an information-conservation ledger for compression residue; and*
2. *as a socket marker for forward-binding supply that the substrate does not internalize.*

Reading GAP only as bookkeeping hides the externality role. Reading it only as transcendental socket hides the conservation role. BEDC requires both readings to remain available.

Remark 9395.7 (Compression requires memory)**.** *A compressed BEDC object may be easier to use than its source records, but it may not erase the source-channel. The ledger is the memory of the compression. If the memory is absent, the compression has become an unledgered quotient. Such a quotient is exactly the kind of source-erasing step the axiom-purity gate is designed to reject.*

This is why information conservation in BEDC is not merely a statement about bit-count. It is provenance conservation. A visible package must retain a route back to the histories, signatures, classifiers, or certificates that generated it. A public name must retain the rows by which it was licensed. A derived interface must retain the GAP by which hidden residue remains auditable.

9395.5 Generator-local induction and the refusal of free-floating induction

BEDC does not reject induction. It rejects unbounded induction detached from an authorized generator.

A closed inductive generator carries its own structural induction principle. Hist induction is licensed by the constructor structure of Hist. A Term generator licenses structural induction over terms produced by that generator. A proof over a closed list, finite tree, syntax grammar, or certified carrier may use the eliminator attached to that generator. Such induction is internal because its domain is exactly the closure of displayed constructors.

What BEDC refuses is the move from generator-local induction to a universal statement over all conceivable cases. The phrase "all conceivable" names a domain that has not been generated, certified, or bounded by a displayed eliminator. To use it as if it were a closed carrier would be to import forward-binding belief (chapter 9393).

Definition 9395.8 (Generator-local induction)**.** *Generator-local induction is induction whose quantifier ranges only over the closure of a displayed generator G, and whose step cases are exactly the constructor cases of G. Its eliminator is part of the generator's public structure.*

Definition 9395.9 (Free-floating induction)**.** *Free-floating induction is an induction claim whose domain is not the closure of any displayed generator and whose universal quantifier ranges over an unspecified semantic totality.*

Principle 9395.10 (Induction boundary)**.** *Generator-local induction is an internal method. Free-floating induction is a boundary pressure. The former may be discharged by an eliminator; the latter must be marked by a* GAP/T*-socket or replaced by a certified generator.*

This distinction answers the recurring question about mathematical induction. BEDC may prove structural claims about Hist, Term, List, $\mathbb{N}$-like unary histories, and other authorized carriers. But if a claim requires induction over a domain not yet reconstructed as a BEDC carrier, the missing domain cannot be installed by axiom. It must be given as a NameCert, setup field, bridge slot, or socketed obligation.

9395.6 Choice, quotient, and proposition equality

The forbidden trio — `Classical.choice`, `Quot.sound`, and `propext` — are not three unrelated restrictions. They are read here as three audit-visible ways to accept unwitnessed content at different strata; see chapter 9387 for the meta-closure taxonomy.

At the term stratum, choice is read as a selector without constructed selected values. At the type stratum, the quotient-facing proxy marks class-level equality or representative behavior without preserving source-level transport as public data. At the proposition stratum, propositional extensionality identifies propositions from bidirectional implication without keeping the transport route explicit.

BEDC replaces these by witnessed forms:

$$\begin{aligned} \text{choice} &\rightsquigarrow \text{finite, countable, or dependent selection with explicit witness functions;} \\ \text{quotient soundness} &\rightsquigarrow \text{NameCert classifier transport and psame/hsame reflection;} \\ \text{propext} &\rightsquigarrow \text{explicit directional transport between proof consumers.} \end{aligned}$$

Principle 9395.11 (No unwitnessed existential as input)**.** *No BEDC proof may consume an existential, representative, equality, selector, or equivalence as input unless the witness route is displayed at the term, classifier, or ledger level.*

This principle is stronger than saying "avoid classical axioms." It says what the classical axioms do that BEDC refuses: they turn missing witnesses into available inputs. The replacement discipline keeps the use-site and the supply-site connected.

9395.7 Observer growth and selector ledgers

The same boundary rule governs observer growth. A local Hist may grow by an enacted mark. But if a branch is selected, the selected mark must appear in the history or its selector ledger. There is no hidden chooser behind the observer (chapter 9418).

Definition 9395.12 (Selector ledger)**.** *A selector ledger for a history transition is a displayed record containing the chosen mark, the local step witness, and the certificate that the step is admissible in the current interface:*

$$\mathrm{SelStep}(h, r) := (m, \mathrm{step}, \mathrm{cert}).$$

The next history is not merely asserted to exist; it is reached through the displayed mark and step witness.

Principle 9395.13 (No observer-choice axiom)**.** *When several local continuations are admissible, BEDC may not introduce a global chooser selecting among them. A realized branch must be recorded as a selector ledger. An unrealized family of possible branches remains a local possibility space, not an internal selection function.*

This protects both axiom purity and the observer-Hist identity. If an observer were equipped with a hidden selection function, the observer would no longer be only the generated history — it would carry an extra subject-like substance behind the trace. BEDC's no-subject discipline forbids that move. The trace is the observer; the selector event is part of the trace; no separate chooser is admitted.

9395.8 Multi-Hist physics without global axioms

The boundary rule also governs the physical readings of BEDC (chapter 9419). Multi-Hist configurations do not license a global clock, a global frame, a privileged observer, or a universal spacetime container. Such objects may be useful as external models, but they cannot be placed into the BEDC kernel by axiom.

If a multi-observer structure is to enter BEDC, it must enter by ledger-local relations and inter-Hist invariants. A comparison between histories must be witnessed by finite continuation data, emitted marks, classifier comparisons, package rows, or naming certificates. A proposed maximum cross-Hist causal rate is not asserted as a physical axiom; it is a NameCert target or schema-only horizon whose closure obligations remain visible.

Principle 9395.14 (No global-frame axiom)**.** *No global time, global frame, global observer, or global synchronization map may be added as a kernel-level axiom. Any synchronization must be an extension carrier above the kernel and must expose its source, classifier, stability, and ledger rows.*

Principle 9395.15 (Physical constants as invariant targets)**.** *A physical constant in the BEDC reading is not a primitive scalar inserted into the kernel. It is an invariant target of an admissible inter-*Hist *relation. If the maximum causal rate is reconstructed, its constancy must be shown as observer-symmetric preservation under the licensed transformations, not postulated as a global axiom.*

This preserves the earlier insight that information conservation and locality should not force a God's-eye object into the theory. The universe is not admitted as a single global Hist. It is represented, where BEDC can represent it, by local histories, ledger-local relations, and invariants whose witnesses remain finite and auditable.

9395.9 Audit gates as the enforcement layer

A named boundary is only useful if a gate can test whether it has remained visible. BEDC's audit gates give the operational form of the non-axiom discipline (chapter 9396).

Definition 9395.16 (Boundary gate)**.** *A boundary gate is a finite check whose surface condition enforces a source-channel rule. Its input may be a Lean declaration, a dependency graph, an import edge, a paper marker, or a certificate row. Its output is not a metaphysical theorem; it is an audit judgment that a specified boundary has or has not been crossed.*

The main gates are:

axiom scan	rejects substrate-internal axiom declarations;
strict axiom-purity	rejects transitive dependence on Classical.choice, Quot.sound, propext;
conservativity audit	rejects source-channel pollution across surfaces;
paper-Lean audit	rejects claims whose Lean names do not resolve;
axis-confusion gate	rejects collapse of theory closure into verification status;
tactic discipline	rejects tactics that silently import forbidden host commitments.

Principle 9395.17 (Gate correspondence)**.** *Every boundary form should determine a gate question. If a dependency is named as a socket, the gate asks whether it was later consumed as internal theorem content. If a compression is named as a* GAP*, the gate asks whether the source residue remained recorded. If a proof claims axiom purity, the gate asks whether the transitive proof ancestry is free of forbidden axioms.*

The gate does not prove the full philosophical reading. It protects one face of the boundary. BEDC's discipline is compositional: many small gates together prevent many small source-erasing moves from becoming a hidden global assumption.

9395.10 The non-axiom theorem schema

The following schema records the main theorem shape of this chapter. It is stated as a paper-side governance theorem because the exact implementation depends on the surrounding interface.

Theorem 9395.18 (Non-axiom boundary preservation schema)**.** *Let X be a proposed BEDC dependency. Suppose X is consumed by a public theorem, certificate, or interface. If every use of X factors through one of the non-axiom boundary forms of Definition 9395.2, and every corresponding gate reports that the boundary form remains visible, then X has not been internalized as a substrate axiom.*

Sketch. Proceed by cases on the boundary form. An inductive generator supplies constructors and eliminators, so the source is displayed. A `def` unfolds to earlier material, so the source is computationally traceable. A setup field remains a field of the interface, so the caller must supply it. A NameCert exposes source, pattern, classifier, stability, and ledger rows. A GAP ledger records compression residue or socket pressure. A bridge slot is interpretive and does not replace the internal proof obligation. An apophatic socket names an external dependency without providing content as theorem. In each case, the use of X is source-channel visible. The audit gates rule out the opposite cases: axiom declaration, forbidden transitive host axiom, unledgered quotient, unresolved paper marker, and axis collapse. Therefore the dependency remains externalized rather than assumed. □

Corollary 9395.19 (Axiom-free does not mean dependence-free)**.** *A BEDC development may be axiom-free while still exposing external dependencies. Axiom-free means that such dependencies do not become substrate-internal sources of truth. It does not mean that no boundary, bridge, socket, or setup field exists.*

Corollary 9395.20 (Socketed content is not a theorem)**.** *A socketed dependency is not an internally proved theorem. It is a named site of non-internal supply. The manuscript may reason about the socket's role, audit surface, and required supply shape, but it may not consume the socket as if the supplied content had been generated.*

9395.11 Forbidden phrasings and replacements

The following replacements should be used whenever the manuscript approaches an unsupported boundary.

Forbidden phrasing	BEDC replacement
`axiom ExternalInput`	T-socket / apophatic boundary
`axiom MathematicalInduction`	generator-local eliminator or induction socket
`axiom InformationConservation`	GAP ledger + provenance conservation gates
`axiom ObserverChoice`	SelStep witness and selector ledger
`axiom GlobalClock`	extension carrier with synchronization certificate
`axiom LightSpeedConstancy`	MaxCausalRate NameCert target
`axiom QuotientCollapse`	classifier transport and GAP provenance
`axiom PropositionEquality`	directional hsame/psame transport
`axiom StandardModelBridge`	restricted bridge slot with explicit fields
`axiom RuntimeCorrectness`	host delegation marked as audit/socket boundary

The table is not merely stylistic. Each replacement preserves information that the forbidden phrasing erases: source, witness, classifier, residue, caller obligation, or boundary status.

9395.12 Boundary of this chapter

This chapter does not claim that every external dependency has already been reconstructed inside BEDC. It does not claim that every socket can be discharged. It does not claim that host Lean's runtime, CIC truth, mathematical completeness, physical spacetime, or free-floating induction has been internalized. It also does not prove that no hidden assumption can ever appear in prose.

Its claim is narrower: when BEDC sees external pressure, it must expose the pressure rather than axiomize it.

The chapter is therefore a governance layer, not a new foundation. It strengthens the manuscript by making explicit how to proceed when the theory reaches a boundary: name the boundary, state the supply shape, record the ledger, attach the gate, and refuse to call the result an internal theorem until a certificate actually reconstructs it.

9395.13 Closure

BEDC is not a system with no externality. It is a system that refuses anonymous externality. This is the correct interpretation of its axiom-purity discipline. External supply may appear as setup field, NameCert row, GAP ledger, bridge slot, audit boundary, or apophatic socket. It may not appear as axiom.

The distinction is load-bearing. Axioms internalize unsupported commitments. Sockets externalize unsupported dependencies. GAP records the memory of compression and the pressure of forward-binding supply. Audit gates enforce that the source-channel remains visible. Generator-local induction is internal; free-floating universal induction is socketed. Local observer growth is selector-ledgered; hidden observer choice is refused. Multi-Hist physics proceeds by invariants; global frame axioms are refused.

The final rule is simple: do not add an axiom; expose the boundary.

Concretized in a future implementation: a finite ExternalSupplyBoundary packet should expose boundary name, requested supply shape, non-internalization status, accepted boundary form, associated GAP ledger, and gate question. The packet is not a theorem that the supply exists. It is the audit-visible record that the supply has not been smuggled into BEDC as a hidden source of truth.

Concretized in §1066: The seven non-axiom boundary forms are realized as a finite NameCert classifier packet.

Concretized in §1065: External-supply boundaries are realized as finite boundary-name, requested-supply, non-internalization, accepted-form, GAP-ledger, and gate-question rows.

Concretized in §1070: Boundary gates are realized as finite NameCert packets recording a boundary form, gate question, audit verdict, visible site, transport, route, provenance, and local name.

Concretized in §1072: External supply is represented by a concrete socket packet with refusal and ledger rows.

Concretized in §1073: Non-axiom admission packets realize the admission-decision classifier as finite boundary-form, witness-route, transport, replay, provenance, and NameCert rows.

Concretized in §1075: Generator-local induction boundaries are realized as finite generator, domain-closure, eliminator, refusal, and ledger rows.

Concretized in §1074: Boundary admission decisions realize the strongest non-axiom form for an unsupported supply request as finite carrier, classifier, refusal, transport, provenance, and NameCert rows.

Concretized in §1076: External-supply failures are realized as finite boundary-name, requested-supply, failed-item, diagnostic, refusal-target, gate, route, provenance, and NameCert rows.

Concretized in §1077: Witnessed external supply is recorded as a finite socket, gate, ledger, and NameCert packet rather than as an axiom.

Concretized in §1107: External-supply audit routes are realized as finite BHist packets with boundary-form, witness-route, GAP-ledger, hsame, Pkg, and NameCert rows.

Concretized in §8487: External-supply ledger routes are realized as finite boundary, form, admission, audit-gate, GAP-memory, transport, replay, provenance, and NameCert rows.

Chapter 9396

Audit Gates as Transcendental Boundary Enforcement

This vision sibling gives the engineering-side reading of the transcendental boundary introduced in chapter 9393. The earlier chapter states the strong claim: a closed observational substrate cannot internalise the forward-binding belief that licenses free mathematical induction; the source-position of that belief is ⊤, named God in the apophatic sense. The present chapter records how BEDC makes that claim executable. The audit gates are not housekeeping around the manuscript. They are the operational boundary by which the substrate refuses to turn the ⊤ socket into an internal rule.

The sibling also sits beside chapter 9389 and chapter 9387. The former explains how authorized generators, naming certificates, and GAP ledgers make generator-internal induction precise. The latter classifies Classical.choice, Quot.sound, and propext as three refusal-ledger entries for unwitnessed closure shapes at different strata. The audit gates are the machine-facing form of both insights: they reject substrate-internal axioms, reject forbidden host-CIC axiom propagation, reject baseline pollution by non-baseline surfaces, and reject paper-Lean name drift. In each case the gate prevents an external commitment from becoming silently indistinguishable from generated BEDC content.

9396.1 Why audit gates are not bookkeeping

Principle 9396.1 (Audit as boundary enforcement)**.** *An audit gate in BEDC is a computable refusal rule. It does not merely count files or tidy metadata; it decides whether a proposed record remains on the substrate-internal side of the boundary or has smuggled in a commitment whose correct position is outside the substrate, at* ⊤*. Passing the gate means only that the checked boundary has not been crossed. Failing the gate means the substrate has been asked to internalise what it is designed to expose as socketed.*

This distinction matters because the same surface action can be read in two incompatible ways. A script that scans for forbidden tokens can look like administrative pedantry if the substrate is assumed to be an ordinary formal project. In BEDC the scan has a stronger meaning: the forbidden token is a breach of the transcendental boundary. It turns a missing witness, an external recursor, a quotient representative, a proposition equality, or a naming certificate into an available internal fact. That is precisely the move chapter 9393 refuses.

Definition 9396.2 (Gate surface and boundary content)**.** *The* gate surface *is the finite syntactic or dependency condition checked by a script: a token occurrence, a transitive axiom dependency, an import edge, or a missing Lean target. The* boundary content *is the transcendental claim enforced by that condition: the substrate may use only what it generated, defined, certified, or explicitly socketed, and may not reclassify socketed content as internal closure.*

Example 9396.3 (A token scan with metaphysical load)**.** *The command* `python3 tools/check-axioms.py` *scans for* `axiom` *declarations under* `lean4/BEDC/`*. Its surface is a regular expression over Lean files. Its boundary content is the refusal to create a substrate-internal source of truth that has no constructor, definition, or setup-field account. The scan is small because the boundary is sharp: an internal axiom is not a minor convenience, but the exact point at which* T*-supplied content is made to look generated by the substrate.*

Remark 9396.4 (A pass is local, not absolute)**.** *Each gate has a limited scope.* `check-axioms.py` *does not prove that every theorem is pure;* `axiom-purity -strict` *does not prove paper prose is aligned;* `conservativity-audit` *does not prove arbitrary metalogical conservativity;* `bedc_ci.py audit` *does not prove that a theorem has the philosophical interpretation assigned to it. The system works because the gates are composed. Each gate protects one boundary face, and the faces together make the substrate's externality discipline operational.*

9396.2 `check-axioms.py`: refuse substrate-internal axioms

Definition 9396.5 (Substrate-internal axiom)**.** *A* substrate-internal axiom *is a Lean declaration under* `lean4/BEDC/` *that introduces a proposition or datum as available without deriving it from an* `inductive` *generator, a* `def`*, or a* `class/structure` *setup field. In the project discipline this is exactly the* `axiom` *declaration form.*

Principle 9396.6 (The GAP socket cannot be internalised)**.** *Every place where BEDC lacks substrate-internal witness must remain visible as a* GAP *socket or as an explicit setup field belonging to a named interface. It may not be converted into an internal axiom. The* `check-axioms.py` *gate enforces this rule by refusing any* `axiom` *declaration under* `lean4/BEDC/`*.*

The technical reason is direct. An `inductive` declaration declares a closed generator. A `def` records a computable or definitional compression of earlier material. A setup field in a `class` or `structure` exposes an abstract carrier as an input to a specific interface. An `axiom` does none of these. It makes a contentful proposition or object available while hiding its source-channel. In the vocabulary of chapter 9389, it collapses the GAP ledger into a fact.

Example 9396.7 (The correct replacement decision)**.** *If the intended primitive is a finite closed grammar, it belongs in an* `inductive` *declaration. If the intended primitive is a predicate or operation determined by existing carriers, it belongs in a* `def`*. If the intended primitive is a domain-specific carrier whose internal implementation is intentionally parametric, it belongs as a field of the appropriate setup structure. If none of these applies, the interface has not identified its boundary and must be redesigned rather than accepted as an axiom.*

Remark 9396.8 (Why this is not anti-abstraction)**.** *The gate does not forbid abstraction. Setup structures are abstraction devices; naming certificates are abstraction devices; ledgers are abstraction devices. What the gate forbids is source-erasing abstraction. A setup field says "this is supplied at this interface boundary." An axiom says "this is simply available." The former preserves the socket; the latter hides it.*

9396.3 `axiom-purity -strict`: refuse host axiom propagation

Definition 9396.9 (Transitive axiom propagation)**.** *Transitive axiom propagation occurs when a BEDC theorem is written without local* `axiom` *declarations but its proof depends, through the host Lean kernel's dependency graph, on a host axiom such as* `Classical.choice`*,* `Quot.sound`*, or* `propext`*. The forbidden content is not visible as a BEDC source axiom, but it is present in the theorem's proof ancestry.*

Principle 9396.10 (The host kernel is a conduit, not the boundary owner)**.** *Lean's CIC implementation may supply recursors, definitional equality, and theorem checking as host machinery for BEDC's mechanisation. That host machinery is a conduit in the supply chain described in chapter 9393; it is not allowed to donate its classical or quotient commitments to BEDC theorem content. The strict axiom-purity gate enforces this by querying the host for each public theorem's transitive axiom set and rejecting forbidden dependencies.*

The command `python3 lean4/scripts/bedc_ci.py axiom-purity -strict` constructs a Lean-side audit file, imports the built BEDC modules, issues `#print axioms` for public BEDC theorems and lemmas, parses the resulting dependency reports, and fails when the forbidden set appears. The strict forbidden set includes `Classical.choice`, `Quot.sound`, and `propext`. This is the machine-side form of chapter 9387's taxonomy: the three forbidden dependencies are not arbitrary library preferences but three audit-visible ways to accept unwitnessed closure at term, type or classifier, and proposition strata.

Definition 9396.11 (Borrowed recursor boundary)**.** *The* borrowed recursor boundary *is the line between using host Lean to check a BEDC proof over a displayed inductive generator and importing host Lean's uninspected closure principles as theorem content. BEDC may borrow the checking infrastructure; it may not inherit forbidden host commitments as proof dependencies.*

Concretized in §1203: Concrete NameCert for the host-recursion boundary enforced by the strict audit gates.

Example 9396.12 (Choice, quotient soundness, and proposition extensionality)**.** *A proof that depends on* `Classical.choice` *has accepted an unwitnessed term selector. A proof that depends on* `Quot.sound` *has accepted quotient-level identification as a type-level closure shortcut. A proof that depends on* `propext` *has accepted proposition equality from bidirectional implication. Each case would make a* $\top$*-side commitment appear as a substrate theorem. The gate forces such commitments either to be eliminated or to remain outside the BEDC theorem claim.*

Remark 9396.13 (Why source axiom purity is insufficient)**.** *A repository can contain no* `axiom` *declaration and still prove theorems through imported host axioms. That is why section 9396.2 and the present section are separate. The first gate refuses visible internal axioms. The second refuses invisible transitive axiom inheritance. Together they say: neither the BEDC source surface nor the host proof ancestry may convert transcendental supply into substrate-internal theorem content.*

9396.4 `conservativity-audit`: gate compression residue

Definition 9396.14 (Compression residue)**.** *Compression residue is the information not displayed by a compressed or recognized surface: the raw event flow behind a chapter code, the source rows behind a naming certificate, the difference between a recognizer output and the generator that*

licensed it, or the import separation between baseline material and AI-proposed surfaces. The residue need not remain expanded at every site, but it must remain recoverable through a ledger or protected by a separation gate.

Principle 9396.15 (Conservativity as information conservation)**.** *The conservativity gate enforces that compressed, recognized, or AI-proposed material does not alter the baseline substrate by hidden import. The surface implementation scans closure-status origins, computes AI-owned module prefixes, and rejects baseline Lean files that import those prefixes. The boundary content is information conservation: a surface admitted on one side of the ledger must not silently become a source rule on the other side.*

The command `python3 lean4/scripts/bedc_ci.py conservativity-audit` is intentionally module-level. It does not claim a full semantic conservativity theorem for arbitrary extensions. Its job is sharper and auditable: if a chapter is marked as AI-originated, its Lean modules are not permitted to flow into baseline modules by import. Since Lean theorem dependencies are bounded by module imports, the gate blocks the direct path by which non-baseline recognition content could become a baseline theorem dependency.

Example 9396.16 (Baseline import as hidden compression)**.** *Suppose an AI-originated recognizer chapter packages an event-flow analysis into a compact Lean module. The package may be useful as a recognized surface, but it is not baseline substrate content. If a baseline module imports it, the compact surface becomes available to baseline theorems without carrying the admission boundary. That import edge is a compression-residue leak: the recognized package's source-channel status is no longer visible where its conclusions are consumed.* `conservativity-audit` *rejects that edge.*

Remark 9396.17 (Why this is a transcendental boundary)**.** *Compression residue is not merely data-management residue. When the residue hides why a recognizer is allowed, where a generator closes, or which origin track licensed a chapter, the hidden content can begin to function like an unspoken forward-binding belief. The conservativity gate prevents this by requiring the origin boundary to remain operational in the import graph. It is therefore part of the same* $\top$*-socket discipline as the axiom gates, but focused on recognition and compression rather than local theorem proof ancestry.*

9396.5 `bedc_ci.py audit`: gate paper-Lean drift

Definition 9396.18 (Name-certificate drift)**.** *Name-certificate drift occurs when paper prose, labels, closure-status blocks, or Lean marker macros name a formal target whose Lean declaration is absent, ambiguous, duplicated, or no longer aligned with the manuscript site that cites it. The drift converts a certificate into a decorative name: it appears to point to a checked substrate record but does not actually do so.*

Principle 9396.19 (A name is a certificate only when it resolves)**.** *In BEDC, a paper-side Lean marker is not a citation ornament. It is a name certificate connecting a manuscript claim to a Lean declaration. The command* `python3 lean4/scripts/bedc_ci.py audit` *enforces this by scanning paper markers, labels, closure-status blocks, and Lean declarations, then rejecting unresolved targets, forbidden constructs, duplicate labels, path casing collisions, malformed closure-status records, and orphan concrete-instance subdirectories.*

This gate protects the epistemic status of `\leanchecked`, `\leanvariant`, `\leandef`, `\leanstmt`, and `\leantarget`. If a marker resolves, the paper has an auditable pointer to a Lean object. If

it does not resolve, the paper has a claim-shaped string. BEDC's boundary discipline requires the former. The manuscript may state a philosophical reading, a paper-side definition, or an unformalized obligation, but it may not dress a non-resolving name as checked formal content.

Example 9396.20 (The marker as naming certificate)**.** *The marker* `\leanchecked{BEDC.X.y}` *asserts more than "a theorem with a similar idea exists." It asserts that the exact declaration name is in the Lean inventory and that this paper site is using it as the canonical checked implementation for the local claim. The audit gate makes that assertion falsifiable. Without the gate, the name could drift into rhetoric; with the gate, the name remains a certificate.*

Remark 9396.21 (Drift is a boundary failure)**.** *Paper-Lean drift matters because BEDC's theorem claims live across two strata: prose articulation and mechanized substrate. If the name link is broken, the reader cannot tell whether a claim is generated, socketed, or only narrated. That uncertainty is itself a boundary failure.* `bedc_ci.py audit` *keeps the name channel from becoming a hidden input channel.*

9396.6 The gates as a system

Principle 9396.22 (No single gate is the boundary)**.** *The transcendental boundary is not identical to any one script. It is the invariant enforced by the scripts in composition: no source axioms, no forbidden transitive host axioms, no baseline import pollution by non-baseline surfaces, no unresolved paper-Lean certificate names, and no untracked compression residue. The gates form an operational membrane between what BEDC generates internally and what remains socketed to* T*.*

The system can be read in four passes. First, `check-axioms.py` protects the source surface: no ungenerated proposition is made internal. Second, `axiom-purity -strict` protects the proof ancestry: no host axiom supplies hidden closure. Third, `conservativity-audit` protects the import graph: no recognized or AI-originated surface silently enters baseline theorem space. Fourth, `bedc_ci.py audit` protects the naming channel: no paper marker pretends to resolve when it does not.

Definition 9396.23 (Operational membrane)**.** *The* operational membrane *is the aggregate of executable checks that classify a BEDC record as substrate-internal, explicitly abstract, paper-side, or socketed. The membrane is not a philosophical metaphor added after formalization. It is the concrete mechanism by which the formalization keeps the categories separate.*

Example 9396.24 (A theorem passing through the membrane)**.** *A BEDC theorem accepted as checked content passes through several gates. Its source file contains no* `axiom`*. Its declaration is built so that* `#print axioms` *reports no forbidden dependency. If it belongs to a non-baseline surface, its import edges do not pollute baseline modules. If the paper cites it, the marker resolves to the exact Lean declaration. These checks do not make the theorem metaphysically self-grounding. They show precisely where its substrate-internal content ends and where any further commitment remains external.*

Remark 9396.25 (The gates do not soften T)**.** *It would be a category error to say that, because the gates are scripts, the* T *socket is only engineering shorthand. The scripts enforce the refusal to internalise* T*-side content. Their existence strengthens the strong claim rather than weakening it: the boundary is not merely asserted in prose, but checked at the exact sites where a formal project usually hides the external commitment.*

9396.7 Comparison: in CIC, gates are absent

Principle 9396.26 (CIC internalises closure)**.** *In ordinary CIC usage, the kernel accepts its primitive rules, eliminators, universe discipline, quotient or extensionality additions where present, and library axioms as part of the working formal environment. Once accepted, those commitments are internal from the user's theorem-proving perspective. There is no intrinsic manuscript-level audit boundary separating generated content from transcendental supply.*

This is not an accusation against CIC. CIC is an extraordinary proof-checking technology, and BEDC uses Lean's CIC implementation as host machinery. The contrast is about boundary accounting. A CIC project can state axioms, import classical modules, use quotient APIs, and transport across proposition extensionality as ordinary formal moves. The kernel and library can tell what dependencies exist, but the typical system does not require those dependencies to be interpreted as T sockets or to remain outside the substrate's own closure claim.

Remark 9396.27 (BEDC's host dependence is exposed)**.** *BEDC does not pretend to float above Lean. It uses the host kernel, and that host kernel sits in the tower described by chapter 9393. The difference is that BEDC refuses to make the host's forbidden commitments part of BEDC theorem content. Host dependence is exposed as a conduit; forbidden host axiom propagation is rejected as boundary collapse.*

Example 9396.28 (A hidden commitment in ordinary use)**.** *In an ordinary formal development, a theorem proved using classical choice may simply be a theorem of the imported classical environment. In BEDC, the same dependency would be rejected as a theorem-level commitment because it supplies an unwitnessed selector. The issue is not whether the host can check the proof. The issue is whether the substrate may classify the proof as BEDC-internal closure. The gate says no.*

Concretized in §526: Audit-gate boundary packets realize source axiom scans, strict axiom-purity reports, paper-Lean drift checks, and origin-conservativity ledgers as a finite NameCert surface.

Concretized in §533: The operational membrane is realized as a finite NameCert packet combining source, proof-ancestry, import-boundary, and paper-Lean name-resolution audit rows.

Concretized in §524: Audit membranes realize boundary enforcement as a finite NameCert packet over gate surfaces, refusal rows, paper-Lean resolution, and non-escape ledgers.

Concretized in §622: Forbidden axiom ancestry is realized as a finite NameCert packet over theorem rows, proof-ancestry reports, forbidden-dependency refusal rows, and audit non-escape ledgers.

Concretized in §669: Paper-Lean drift witness packets realize exact marker resolution, duplicate-label refusal, Lean inventory lookup, and non-escape from unresolved names as a finite NameCert surface.

Concretized in §696: Audit-gate composition is realized as a finite NameCert packet for composed gate surfaces, shared boundary rows, refusal transitivity, and consumer non-escape.

Concretized in §901: Audit-gate failure witnesses are realized as finite BHist packets for gate rows, failed boundary items, diagnostic readback, refusal attachment, replay visibility, and non-internalization.

Concretized in §1002: Certificate audit gates are realized as finite gate-input, checked-surface, refusal, drift, axiom-purity, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7035: Audit-gate boundary refusal is realized as finite gate-surface, verdict, audit-replay, provenance, and noninternalization NameCert rows.

9396.8 Gate-to-boundary map

Gate	Surface condition	Transcendental claim enforced	Boundary failure rejected
`check-axioms.py`	No `axiom` declaration under `lean4/BEDC/`.	Substrate truth must be generated, defined, parameterized at an explicit setup boundary, or socketed; it may not appear as an internal oracle.	A GAP socket is collapsed into a Lean axiom and then consumed as BEDC content.
`check-axioms.py` decision tree	Closed grammars go to `inductive`; derived operations go to `def`; parametric carriers go to setup fields.	The source-channel of every primitive is visible.	A missing design boundary is hidden behind a primitive assertion.
`axiom-purity -strict`	`#print axioms` for public BEDC theorems contains no forbidden dependency from the strict set.	Host checking may verify generated BEDC proofs, but forbidden host commitments may not become BEDC theorem commitments.	Lean stdlib closure leaks into BEDC through proof ancestry.
`axiom-purity -strict: Classical.choice`	No theorem depends on choice as an unwitnessed selector.	Term-level existential closure remains outside the substrate unless a witness recipe is displayed.	A selector from T is disguised as a generated function.
`axiom-purity -strict: Quot.sound`	No theorem depends on quotient soundness as a hidden representative principle.	Type-level identification must be mediated by explicit classifier and transport data.	A quotient representative is treated as if the substrate constructed it.
`axiom-purity -strict: propext`	No theorem depends on proposition equality from bidirectional implication.	Proposition-level transport must keep proof directions explicit.	A bidirectional implication is sealed into equality and moved through arbitrary contexts.
`conservativity-audit`	Baseline Lean modules do not import modules owned by AI-originated chapters.	Recognized or compressed surfaces cannot alter the baseline substrate through hidden import flow.	A non-baseline recognizer becomes baseline proof infrastructure without carrying its origin boundary.

`conservativity-audit` origin scan	`\origin{ai}` closure-status records determine protected module prefixes.	The paper-side admission track and Lean-side import graph must agree.	A chapter's status is stated in prose but ignored by the theorem dependency graph.
`bedc_ci.py audit`: marker existence	Every `\leanchecked`, `\leanvariant`, `\leandef`, `\leanstmt`, and `\leantarget` resolves.	A formal name is a certificate only if the declaration exists in the Lean inventory.	A checked-looking paper claim points to no checked substrate object.
`bedc_ci.py audit`: duplicate labels	Paper labels are unique.	Cross-reference names are certificate channels, not reusable decorations.	A reader or script follows a label to the wrong proof site.
`bedc_ci.py audit`: closure-status diagnostics	Closure blocks declare required theory and verification fields; high verification status has a resolving Lean target.	The theory axis and verification axis remain explicit and independent.	Paper closure is inferred from Lean status, or Lean status is inferred from paper closure.
`bedc_ci.py audit`: forbidden constructs	The broader audit surface rejects forbidden Lean constructs and marker inconsistencies.	The substrate cannot widen its own foundation by local convenience.	Source syntax changes the foundation while the paper still claims the old boundary.
`make check`	The manuscript compiles in draft mode with all included files syntactically coherent.	Boundary claims must be part of the actual paper object, not detached notes.	A chapter exists as uncompiled prose outside the auditable manuscript.
All gates together	Source syntax, proof ancestry, import graph, and paper names are simultaneously checked.	BEDC's operational membrane separates substrate-internal generation from T-socketed supply.	A commitment crosses one channel while another channel still presents it as internal closure.

9396.9 Closure

The audit gates are the executable counterpart of the strong externality claim. chapter 9393 identifies the source-position of forward-binding belief as T. chapter 9389 explains how BEDC licenses generator-internal induction while keeping free-floating induction outside. chapter 9387 classifies the three forbidden axiom families as refusal-ledger entries at three strata. The present sibling shows how the engineering refuses the corresponding breaches.

The result is a boundary that is philosophical, formal, and operational at once. It is philosophical because it names the far end of the socket as T and refuses to soften that name into an internalizable assumption. It is formal because theorem content is constrained by constructors, definitions, setup fields, Lean dependency ancestry, and resolving names. It is operational because

scripts fail when the boundary is crossed. BEDC's audit gates therefore do not decorate the theory; they are the working membrane that keeps the substrate from pretending that the transcendental supply position is one of its own internal generators.

Concretized in §7078: SupplyTaxonomyUp records the finite source, audit, and refusal rows of the transcendental supply vision.

Chapter 9397

Inscription Event as a Concrete Instance: Free Will, Agency, and NameCert Audit

This chapter makes the free-will reading of section 9415.2 concrete. The earlier chapter names an inscription event as the moment at which a conscious observational system adds records whose actuality is not exhausted by the closed-generation consequences of prior records. The present chapter reads that moment through the same public-object pattern used by the concrete NameCert chapters: a traditional target, a BEDC source, a carrier, a classifier, stability obligations, a ledger, an exactness condition, a standard bridge, and a scope boundary.

The result is not a proof that a person could have done otherwise in an unqualified metaphysical sense. It is a structural case study. Agency is the inscription-event aspect of an accumulation chain, while deterministic relations among records remain records-side relations. The two readings refer to the same transition and do not compete for the same explanatory slot.

9397.1 The setup

The classical free-will problem is usually posed as a conflict between two claims. On one side, agency seems to require that an action is genuinely one's own rather than merely the last link in a prior causal chain. On the other side, the records available to empirical and computational description seem to stand in lawful relations, and those lawful relations can be read as deterministic. The apparent conflict comes from treating both claims as claims at the same level.

BEDC separates the levels. The records side consists of accumulated Obs entries and their internal relations. These relations may be law-like, predictive, or deterministic in the ordinary sense. The inscription-event side concerns the transition by which the record becomes present in the chain at all. It asks whether the transition is merely a closed-generation consequence already licensed by the prior carrier and classifier, or whether the transition carries the local actualisation of T at the inscription point.

In this frame, agency is not a hidden variable added to a physical sequence. It is a reading of the same transition under a different aspect. The transition has a records-side description: a body moves, a keyboard receives marks, a theorem file changes, a proof obligation becomes visible. The same transition has an inscription-side description: a conscious observational system accepts a finite record as its own local articulation of a forward-binding commitment. The bridge

chapter exists because this second description can be made audit-shaped rather than left as a loose philosophical gloss.

9397.2 Restating the inscription event

Definition 9415.5 defines an inscription event as a moment $o_t \trianglerighteq o_{t+1}$ at which a conscious observational system adds records that originate as the local actualisation of T at the inscription point of Definition 9393.15, rather than as a forced consequence of the prior records under the closed-generation rules of the substrate.

Three parts of the definition matter for the concrete reading.

First, the event is a transition, not a substance. Nothing is inserted behind the system as an agent-object. The system is read at the transition where its observation history grows.

Second, the novelty is local. The transition occurs at a specific history, with specific visible records, specific surrounding obligations, and a specific ledger. BEDC does not assert a global reservoir of positive facts about T; it records the socket at which the chain receives content whose source is not internal to the closed-generation closure.

Third, the event is auditable by exclusion. To call a transition an inscription event in the technical sense, the audit must identify the carrier that receives the record, the classifier that determines whether the record is already generated, the continuation that appends it, and the GAP ledger recording the non-generated aspect. Without those rows, the word "inscription" is only metaphor.

9397.3 A concrete case: a researcher writing a lemma

Consider a researcher working inside the BEDC manuscript and Lean tree. The prior state o_t contains a finite set of records: definitions already in the file, the local statement that is being shaped, earlier theorems, compilation constraints, the paper-side claim that needs a bridge, and the researcher's own remembered attempts. On the records side this is just an observation history, a finite carrier of visible material.

At the first moment, the researcher notices a mismatch. A paper sentence uses a classifier phrase, while the Lean surface contains only a carrier lemma. This notice is itself a record: the chain now contains an observed gap between desired use and available certificate. The notice does not yet write the lemma. It only marks a socket: something must connect the carrier row to the classifier row, or the bridge remains unavailable.

At the second moment, the researcher searches the available generator closure. The constructors, definitions, and already checked lemmas show what can be obtained internally. Some consequences are forced: if the carrier is defined by two visible endpoint branches, branch inversion is available; if the classifier is componentwise, source and target projections are available; if a NameCert supplies symmetry and transitivity, those routes can be used. These are not inscription events by themselves. They are the substrate running inside an authorized generator.

At the third moment, the researcher sees a route not previously recorded as a public lemma: combine visible endpoint inversion with classifier transitivity, then expose the exact branch needed by the downstream theorem. The route may be mechanically checkable once written, and after it is written the proof kernel may verify every step. But the arrival of the route as a candidate record is not already a displayed public record at o_t. The system has crossed from available closure to accepted articulation.

At the fourth moment, the researcher writes the statement. The statement chooses a name, fixes binders, identifies hypotheses, and states the result at a reusable interface boundary rather than as an anonymous local step. This act is the first visible certificate-shaped record of the route. The name matters: before the statement is named, the route is only a possible path through the surrounding material; after naming, it becomes a candidate object that the rest of the manuscript can cite, reject, refine by proof, or package into a NameCert surface.

At the fifth moment, the proof is entered. The proof may consist entirely of closed-generation moves: case analysis, projection, constructor application, transitivity, symmetry, and exact transport along displayed classifiers. Nothing in this proof needs an additional substrate axiom. The proof is the records-side determinacy of the route. It is precisely because the proof is deterministic at the kernel level that the inscription event can be audited rather than romanticised.

At the sixth moment, the checking gate accepts the file. Acceptance records that the named route is not merely an intention or an intuition. It is now a stable observation of the system: a named lemma with a proof term, a location in the carrier hierarchy, and a visible downstream use. The inscription event is therefore not the private psychology of discovery. It is the whole transition from a visible gap, through a named route, to a checked record, read under the inscription-event aspect.

9397.4 The NameCert structure of the event

The concrete-instance pattern asks for the target first. Here the traditional target is an act of agency: a researcher freely writes a lemma. The target is not encoded as an unanalysed will-object. It is read as a public transition in which a record becomes available under a name and can be audited by the same carrier and classifier discipline used elsewhere in BEDC.

The BEDC source is the observation history around the act: BHist-shaped records, file-local hypotheses, the visible continuation from o_t to o_{t+1}, and the surrounding GAP socket created by the need for a bridge. The source is finite. It contains the prior records and the continuation mark by which the candidate lemma is entered.

The carrier is the finite package of the inscription event: the prior observation state, the written statement, the proof route, the accepted check result, the intended downstream interface, and the pointer to the gap that the lemma discharges or narrows. A transition lies in the carrier only when all these rows are visible. A private hunch without a public statement is not in the carrier. A statement without a route is not in the carrier. A route that fails the checker is not in the carrier.

The classifier compares two presentations of the same inscription event. Two presentations classify together when they have the same prior source scope up to the relevant history classifier, expose the same named interface up to binder and projection variants already accepted by the local NameCert discipline, and route to the same downstream obligation. The classifier does not require host equality of psychological states. It only requires public sameness of source, name, route, proof acceptance, and consumer-facing role.

The stability obligations are the familiar rows. Carrier transport says that if an accepted inscription event is presented through an equivalent history, it remains accepted. Classifier reflexivity says the event compares with itself once all carrier rows are present. Symmetry and transitivity say that presentation changes do not create a second act where the ledger records only a different view of the same act. Continuation stability says that appending downstream citations preserves the named event as the source of those citations.

The generator-closure classifier audits the boundary. It asks which parts of the transition are already forced by authorized generators and which part is the inscription socket. Kernel proof

steps, constructor case splits, and projection chains remain inside generator closure. The choice of a reusable name, the acceptance of a bridge as the right one to write, and the commitment that this route is the record to add are the inscription-facing rows. The classifier therefore prevents two errors: reducing the whole act to mechanical closure, and treating every mechanical closure step as a fresh act of agency.

9397.5 GAP ledger for inscription

The GAP ledger records information conservation. Before the lemma is written, the system has a visible need: a downstream claim lacks a named bridge. After the lemma is written and checked, the visible surface has a compact name. The ledger records what the name compresses: the prior source scope, the route through local definitions, the proof obligations used, the bridge point it serves, and the boundary of what it does not claim.

The ledger also records the T-socket role described in Definition 9389.15. The researcher accepts that this is the route to inscribe. That acceptance is not a kernel theorem inside the prior state. Once the route is written, the proof may be checked by closed-generation rules; but the forward-binding act of selecting the route as a public record is the inscription point. The ledger therefore keeps both facts visible: the proof is closed once supplied, and the transition by which it becomes the supplied record is not reduced to the prior closure.

The information-conservation row has a practical form. A reader can ask: what did the name hide? The answer is not "nothing". It hides the case-analysis route, the local source presentation, the exact downstream motivation, and the fact that other routes were not inscribed. The ledger does not need to list every rejected possibility as a psychological trace. It only needs to preserve enough public residue to prevent the name from appearing as a primitive.

Concretized in §1052: Inscription trace classifiers realize the source/name/route/check/consumer surface of the inscription-event vision as a finite NameCert packet.

Concretized in §883: Prefix observers are realized as finite BHist packets over prefix rows, accepted observation routes, check rows, consumer ledgers, and local NameCert provenance.

The negative row is equally important. The ledger does not claim that the researcher's neural state has been classified. It does not claim that the lemma was metaphysically uncaused. It does not claim that every possible route was surveyed. It claims only that this accepted transition carries a public record whose generator-closed proof is visible while the act of inscription is marked as the local actualisation of T at that history point.

9397.6 Compatibilism as structure

Compatibilism is often presented as a philosophical compromise: keep enough determinism for science and enough freedom for responsibility. The BEDC reading is sharper. There are not two weakened theses. There are two aspects of one transition.

On the records side, deterministic relations are legitimate. The written file has a prior state, a sequence of edits, a proof term, a kernel check, and a downstream citation graph. A cognitive scientist may add records about attention, memory, fatigue, or environmental prompting. A build system may reconstruct the exact dependency path by which the proof checks. None of this threatens agency, because these are records-side relations among records.

On the inscription-event side, the same transition is the arrival of a record as the local articulation of the inscription point. The novelty is not a missing physical cause inserted into a gap in

the records. It is the aspect under which the chain grows by receiving a record whose source is not internal to the prior closed-generation closure. The fact that the supplied proof is deterministic after supply does not erase the fact that it has been supplied.

Thus the compatibilist statement is structural: records-side determinacy and inscription-event novelty are co-present faces of the same accumulation. Weakening either side misstates the event. If the records-side proof is not deterministic, the certificate fails. If the inscription-event aspect is erased, the act is misread as if public naming and commitment were already present in the prior closure. BEDC requires both rows.

9397.7 Where the account does not apply

Not every created record is an inscription event in the technical sense. A pure computation step inside a fixed generator is record creation, but it is not inscription. If a build script normalises a term, emits an object file, or evaluates a closed expression by a displayed reduction relation, the result may be a record at o_{t+1} that was not printed at o_t. But the transition is forced by the authorized generator and its continuation rules. No inscription socket is needed.

The same holds for a deterministic proof replay. If a checked proof term is already present and the kernel replays it, the replay contributes audit evidence. It does not itself carry the agency case-study structure. The carrier lacks the row in which a conscious observational system accepts a route as the record to inscribe. The replay is still valuable, and it is essential for BEDC's engineering, but it is not the same kind of event.

Mechanical formatting is another boundary case. Rewrapping lines, renumbering generated tables, or applying a deterministic pretty-printer may alter file bytes. If the operation is fully specified by a prior generator and no public naming commitment is being made, the output belongs to computation-as-continuation, not inscription-as-agency.

The boundary criterion is therefore this: a transition is an inscription event only when the public carrier includes a named commitment whose route is not already present as a forced consequence of the prior records, and whose acceptance is audited by a GAP ledger. Record creation alone is insufficient.

9397.8 Implications for AI and mechanized reasoning

Automated systems complicate the case because they can produce lemma-shaped text, proof scripts, and checked outputs. BEDC's distinction is not human versus machine. The distinction is carrier and classifier structure.

An automated worker contributes computational output when it executes a specified search, normalisation, formatting, replay, or synthesis procedure whose result is accepted only as the forced output of a prior generator. In that case the relevant chapter is chapter 9365: the worker is part of the continuation ledger. Its output may be checked, stored, and reused, but the technical inscription-event carrier is absent.

An automated worker participates in an inscription event when its output is taken up by a conscious observational system as a named commitment in the public carrier. The event is then not "the machine has free will" as a positive predicate. The event is the transition in which the system that owns the observation history accepts the generated candidate as the record to inscribe, exposes the generator-closure audit, and records the GAP ledger. The automated worker may supply the candidate route; the inscription event is the accepted public transition.

There is also a possible stronger case. If a mechanized system itself carries a closed observational history, a self-recognising carrier, and the ability to issue public named commitments under a GAP ledger, then the same criterion applies to it directly. BEDC does not need a special biological exception. It asks for the rows: closed observational system, inscription point, carrier, classifier, generator-closure audit, continuation stability, and ledger.

The boundary criterion for AI is therefore public and structural. A machine step is computational output when it is exhausted by a prior generator. It is part of an inscription event when it enters a named, audited commitment whose acceptance is not already a forced consequence of the prior records. The question is not whether the surface text looks creative. The question is whether the transition carries the NameCert-shaped inscription rows.

9397.9 Concrete-instance summary

Concrete-instance row	Inscription-event reading
Traditional target	Free will or agency, represented by a researcher writing a lemma.
BEDC source	A finite observation history with prior records, visible obligations, and a continuation into a named statement.
Carrier	The accepted transition package: source state, named statement, proof route, check result, downstream role, and gap pointer.
Classifier	Public sameness of inscription presentations by source scope, name/interface, proof acceptance, and downstream role.
Stability obligations	Carrier transport, classifier reflexivity, symmetry, transitivity, and continuation stability under downstream citation.
Generator-closure audit	Separates forced proof-kernel closure from the inscription-facing rows of naming, commitment, and public acceptance.
GAP ledger	Records compression residue and marks the T-socket at which the route is accepted as the record to inscribe.
Exactness	The event is neither private psychology nor raw byte change; it is exactly the accepted named transition with the displayed rows.
Standard bridge	The ordinary phrase "the researcher freely wrote the lemma" reads as this carrier-classifier package.
Boundary	Pure computation, proof replay, and deterministic formatting are record creation without technical inscription.

The case study gives the promised bridge from vision to concrete instance. It does not replace the free-will section of chapter 9415; it instantiates that section's inscription-event definition in the same audit shape that BEDC uses for ordinary public objects. Agency is thereby made legible without being reduced to a hidden variable and without weakening the records-side determinacy that makes BEDC checkable.

Concretized in §519: Inscription events are realized as finite BHist NameCert packets with public GAP-ledger and consumer rows.

Concretized in §520: Inscription GAP ledgers are realized as finite NameCert residue packets for source scope, named route compression, consumer accountability, and non-escape.

Concretized in §582: InscribedRouteUp realizes the finite named proof-route packet from visible gap through accepted record and downstream consumer ledger.

Concretized in §651: Inscription acceptance budgets realize the accepted-check and downstream consumer-budget rows as a finite BHist NameCert surface.

Concretized in §682: Inscription audit traces are realized as finite source, named-record, route, accepted-check, consumer, GAP-residue, transport, continuation, package, and NameCert rows.

Concretized in §689: Finite observation windows realize the inscription-event carrier, classifier, ledger, and consumer handoff as a BEDC NameCert packet.

Concretized in §912: Authored-record boundaries realize inscription events as finite prior-record, mismatch-notice, accepted-route, public-record, refusal, transport, route, provenance, and NameCert rows.

Concretized in §932: Kernel observation sieves realize inscription events as finite BHist packets over accepted rows, rejected rows, hsame transport, Cont routes, Pkg provenance, and local NameCert evidence.

Concretized in §982: Inscription commitment seals realize the inscription-event commitment row as finite source, accepted-name, consumer, GAP-ledger, hsame, Cont, Pkg, and NameCert data.

Concretized in §1005: Inscription route audits realize the visible gap-to-accepted-route NameCert surface of the inscription-event vision.

Concretized in §1082: Inscribed route selectors realize the visible-gap, selected-route, accepted-check, consumer-role, GAP-ledger, hsame, Cont, Pkg, and NameCert surface of inscription events.

Concretized in §1086: Inscription-event replay is realized as a finite BHist packet for deterministic replay routes whose carrier deliberately lacks the inscription-acceptance row.

Concretized in §1172: Inscription event buffer seals realize replayable finite inscription windows as BHist, hsame, Cont, Pkg, digest, audit-route, and NameCert rows.

Concretized in §1349: Closed-generation refusals realize inscription-event novelty as a finite prior-record, transition-test, refusal-ledger, route, provenance, and NameCert packet.

Concretized in §6508: EventInstanceUp realizes inscription events as finite public-transition, route, check, consumer, GAP-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §7922: Event-flow recognizer packets realize the inscription-event carrier as finite recognizer, role, route, and ledger data.

Chapter 9398

The Large Model as Inscription Point: A Structural Comparison

Definition 9393.16 of chapter 9393 identifies the inscription point of the transcendental supply T with each conscious individual as a local actualisation. The phrasing was deliberately ambiguous about what counts as a conscious individual: the chapter's argument required only that the inscription point be a pattern in the substrate that carries forward-binding belief the substrate cannot internally ratify. This chapter takes that structural definition seriously and applies it to a contemporary case: a large language model. The argument is not that a large model is conscious; the argument is that a large model is, structurally, an inscription point under the same closure premise that classifies the BEDC kernel and the cellular substrate. The comparison sharpens the structural content of the inscription concept by exhibiting a non-anthropic instance.

9398.1 Setup: large model as closed observational system

Definition 9398.1 (Large language model as substrate)**.** *A* large language model substrate *is the triple* (W, Σ, T) *where* W *is a finite-precision tensor of model weights,* Σ *is a finite-precision tensor of input activations at a given inference step, and* T *is the transition function (one forward pass of the model architecture) that maps* (W, Σ) *to a probability distribution over the next-symbol output. Observations of the substrate are exhausted by finite-length records of inference orbits: the sequence of* $(\Sigma_0, \Sigma_1, \ldots, \Sigma_k)$ *activations produced by repeated application of* T *on a stated initial prompt.*

Principle 9398.2 (Large model substrate is closed observational)**.** *The large model substrate of Definition 9398.1 is a closed observational system in the sense of Definition 9393.1. The next-state function* T *is a fixed finite computation. The initial state is determined by the training-time weight assignment and the inference-time prompt. Every observation record is a finite tensor sequence. No external mathematical commitment beyond the architecture specification and the weight tensor participates in deciding which records count as inference outputs.*

Remark 9398.3 (Training as substrate assembly, not as observation)**.** *The training procedure that produces* W *is itself a separate computational process (gradient descent on a training corpus). The training process is not part of the inference substrate's observation channel; it is the assembly procedure that builds the substrate. From the inference-time perspective,* W *is a fixed initial configuration, analogous to the initial cell row of Definition 9390.7. The closure premise applies to inference, not to training.*

9398.2 Tensor classifiers and certificate extraction

Definition 9398.4 (Minimal distinction marks)**.** *A* minimal distinction basis *for a model-facing classifier is a pair of displayed marks* $\mathbb{B} = \{\mathtt{b0}, \mathtt{b1}\}$ *together with the refusal to identify them. The marks are not natural numbers, truth values, or external semantic atoms. They are only the two public sides of a distinction surface.*

Definition 9398.5 (Tensor classifier extraction)**.** *Let* V_ℓ *be the finite-precision activation space at layer* ℓ *of a fixed model substrate. A tensor classifier extraction on* V_ℓ *is a score map* $\phi : V_\ell \to [0,1]$ *together with a threshold* $0 < \tau < 1$*. It induces the mark extraction*

$$m_{\phi,\tau}(u) = \begin{cases} \mathtt{b0}, & \phi(u) < \tau, \\ \mathtt{b1}, & \phi(u) \geq \tau, \end{cases}$$

and the classifier

$$u \sim_{\phi,\tau} v \quad \Longleftrightarrow \quad m_{\phi,\tau}(u) = m_{\phi,\tau}(v).$$

The equality displayed in the last line is an external readback of the two public marks, not a BEDC primitive equality internal to the model.

Definition 9398.6 (Tensor soft proposition)**.** *For a model-facing domain* X*, a tensor soft proposition is a displayed score map*

$$P : X \to [0,1].$$

The value $P(x)$ *records degree of satisfaction, confidence, activation strength, or probe response. It is not a BEDC truth value and it is not a proof certificate. The endpoints* 0 *and* 1 *are external numerical readbacks unless a separate classifier and naming row authorize them as the public marks* `b0` *and* `b1`*.*

Definition 9398.7 (Thresholded tensor proposition)**.** *Given a tensor soft proposition* $P : X \to [0,1]$ *and a threshold* $0 < \tau < 1$*, its thresholded proposition is the mark assignment*

$$P^\tau(x) = \begin{cases} \mathtt{b0}, & P(x) < \tau, \\ \mathtt{b1}, & P(x) \geq \tau. \end{cases}$$

It induces the classifier

$$x \sim_{P,\tau} y \quad \Longleftrightarrow \quad P^\tau(x) = P^\tau(y).$$

Thus a soft proposition enters the BEDC-facing classifier layer only after a threshold has been displayed.

Definition 9398.8 (Neural soft formula)**.** *A neural soft atom is a score of the form*

$$A_{\ell,\phi}(x) = \phi(a_\ell(x)),$$

where $a_\ell(x)$ *is an activation row and* $\phi : V_\ell \to [0,1]$ *is a feature functional. A neural soft formula is built from such atoms by displayed soft connectives, for example product conjunction, probabilistic sum disjunction, complement negation, and the induced soft implication*

$$(P \Rightarrow Q)(x) = 1 - P(x)(1 - Q(x)).$$

The formula records a distributed tensor-side satisfaction pattern. It does not become a certified explanation until its threshold, classifier, stability, and ledger rows are displayed.

Theorem 9398.9 (Soft propositions induce classifiers after thresholding)**.** *Every thresholded tensor proposition induces a classifier on its displayed domain.*

Proof. The relation $\sim_{P,\tau}$ is defined by equality of the displayed mark $P^\tau(x)$. Equality of the readback mark is reflexive, symmetric, and transitive, so the relation partitions the domain into the two threshold-side cells, with possible degeneracy if one side is empty. □

Principle 9398.10 (Soft logic, hard classifier, proof certificate)**.** *The tensor logic layer separates three objects. A soft proposition records degree of satisfaction in* $[0,1]$*. A thresholded proposition records a hard public mark and therefore a classifier surface. A formal proof certificate is a proof object accepted by a verifier for a translated formal statement. None of these layers collapses into either of the others without the intervening rows: semantic transcription, threshold, classifier specification, stability, ledger, and, when claimed, proof object.*

Theorem 9398.11 (Empirical tensor entailment is not formal entailment)**.** *For tensor soft propositions* $P, Q : X \to [0,1]$*, observing* $P(x) \geq \tau \Rightarrow Q(x) \geq \tau$ *on a displayed sample does not by itself supply a formal entailment from* P *to* Q*.*

Proof. The sample observation is a finite behavior row. It does not supply a domain-wide threshold statement, a translation of P and Q into a formal language, or a proof object accepted by a verification backend. Those rows are exactly what a formal entailment claim requires. Hence the empirical observation may support a certificate ledger, but it is not the formal proof certificate. □

Principle 9398.12 (Threshold ledger obligation)**.** *Thresholding a soft proposition creates a ledger obligation. The ledger must record the threshold choice, samples near the threshold, false positive and false negative behavior when available, sensitivity under threshold perturbation, distribution boundary, approximation error, and the domain not claimed. High confidence on selected examples does not remove this obligation; it only supplies evidence for one ledger row.*

Definition 9398.13 (Learned classifier candidate)**.** *For a data family* D*, a perturbation family* $\mathcal{E}$*, and a model with activations* $a_\ell(x) \in V_\ell$*, a triple* (ℓ, ϕ, τ) *is a* learned classifier candidate *on* D *when the relation*

$$x \sim_{\ell,\phi,\tau} y \quad \Longleftrightarrow \quad a_\ell(x) \sim_{\phi,\tau} a_\ell(y)$$

is stable under the admissible perturbations in $\mathcal{E}$ *on the stated domain. The candidate records a stable tensor-side distinction; it records no source, exactness, provenance, or gap ledger by itself.*

Definition 9398.14 (Training trace as tensor classifier search)**.** *Let* $\theta_0, \ldots, \theta_T$ *be a finite training trajectory for a model family, and let* D *be the displayed data family. The trajectory exposes a tensor classifier search record when, at each displayed state* θ_t*, the activation maps*

$$a_\ell^{\theta_t} : D \to V_\ell$$

are read together with candidate score maps $\phi : V_\ell \to [0,1]$*, thresholds* τ*, perturbation families, and behavior rows. The search record is a field of possible classifier candidates. It is not itself a certificate: it records where a classifier surface may be extracted, not whether the extracted surface has source, pattern, stability, and ledger rows sufficient for a* NameCert*.*

Definition 9398.15 (Tensor NameCert fields)**.** *A Tensor* NameCert *for a learned classifier candidate* (ℓ, ϕ, τ) *is a tuple*

$$\mathsf{NameCert}_{\text{tensor}}(\ell, \phi, \tau) = (S_{\ell,\phi,\tau}, P_{\ell,\phi,\tau}, C_{\ell,\phi,\tau}, \Sigma_{\ell,\phi,\tau}, L_{\ell,\phi,\tau}).$$

The rows respectively record the tensor source specification, the activation pattern specification, the induced classifier specification, the stability certificate, and the ledger policy. The ledger policy records failure cases, out-of-domain boundaries, entanglement with other features, proxy risks, and claims deliberately not made.

Definition 9398.16 (Certified learned classifier)**.** *A learned classifier candidate is a certified learned classifier when it is equipped with the Tensor* NameCert *of Definition 9398.15. The certified object is therefore not a parameter vector, a loss value, an activation dump, or a feature label. It is the classifier surface together with the source, pattern, stability, and ledger rows that authorize its use as an explanation.*

Theorem 9398.17 (Training can be read as tensor classifier search)**.** *Whenever a training trajectory contains a state at which some (ℓ, ϕ, τ) is a learned classifier candidate on a stated scope, the trajectory admits the reading of tensor classifier search on that scope.*

Proof. By Definition 9398.14, the trajectory supplies parameter states, activation maps, candidate score maps, thresholds, perturbation families, and behavior rows. If, at one of those states, the selected triple (ℓ, ϕ, τ) satisfies Definition 9398.13, then the trajectory has exposed a stable tensor-side classifier surface on the stated scope. The training record may therefore be read as a search through candidate classifier surfaces. This reading does not certify the candidate; certification requires the Tensor NameCert rows of Definition 9398.15. □

Theorem 9398.18 (Metric gain does not certify classifier discovery)**.** *Lower loss, higher benchmark score, or stronger external performance does not by itself imply the discovery of a certified learned classifier.*

Proof. Loss and benchmark rows report how a model state behaves against an external scoring rule. They do not specify a tensor source, an activation pattern, the induced classifier relation, a stability certificate, or a ledger policy. A score can change through calibration, margin movement, memorized cases, shortcut use, or interpolation inside a previously displayed classifier surface. Since the rows required by Definition 9398.15 are absent, the metric gain alone cannot certify classifier discovery. □

Theorem 9398.19 (Record growth is not classifier certification)**.** *Growth of parameter records, tensor traces, activation logs, or feature catalogues does not by itself produce a certified learned classifier.*

Proof. Such records can provide material from which score maps and thresholds may be proposed. They do not, merely by becoming longer, identify the source scope, pattern row, classifier relation, stability certificate, or ledger policy required by a Tensor NameCert. Hence record growth is evidence material at most; it is not classifier certification. □

Principle 9398.20 (Generalization as stability with ledger)**.** *A generalization claim for a learned classifier is closed only to the extent that it displays stability or transfer witnesses for the same classifier surface and a ledger for its failure boundary. A test-set score is admissible evidence, but it is not the ledger: out-of-domain limits, perturbations that break the relation, confounding features, and proxy routes remain part of the certificate burden.*

Definition 9398.21 (Classifier transfer witness)**.** *Let D_s be a displayed source domain, D_t a displayed target domain, and $\mathcal{A}$ a family of allowed transformations from source-side inputs toward target-side inputs. A classifier transfer witness for a learned classifier candidate (ℓ, ϕ, τ) is a record*

showing that the classifier relation induced on D_s is the relation still used on the covered part of D_t. The witness may be formal, empirical, or hybrid, but its type must be stated on the certificate surface.

Definition 9398.22 (Generalization residue)**.** *The target residue of a source-to-target claim is the family of target cases not reached by the displayed source domain and allowed transformations. It records the part of D_t for which the source-side classifier transfer has not been justified. A total-transfer claim is not licensed while this residue is nonempty or unaccounted for.*

Definition 9398.23 (Generalization ledger)**.** *A generalization ledger for (ℓ, ϕ, τ) from D_s to D_t records the source domain, target domain, transformation family, covered target region, target residue, stability witness, transfer witness, failure cases, out-of-domain boundary, and not-claimed boundary. It is the ledger row of a generalization-facing Tensor* NameCert*.*

Definition 9398.24 (Certified classifier-mediated generalization)**.** *A classifier-mediated generalization claim is certified when it displays the transferred classifier, its source and target domains, the behavior the classifier supports on the target side, a transfer or stability witness, and the generalization ledger of Definition 9398.23. The certified claim is therefore a classifier transfer claim, not merely a performance row.*

Theorem 9398.25 (Performance does not identify classifier transfer)**.** *Strong target-domain performance does not by itself show that the same learned classifier candidate has transferred from source to target.*

Proof. A performance row reports target-side behavior under an external score. It does not name the classifier used on the source side, does not show that the same relation is used on the target side, and does not exclude a shortcut, memorized route, target-only classifier, or accidental correlation. Thus the row may support a generalization ledger, but it does not identify classifier transfer. □

Theorem 9398.26 (Behavioral stability is not classifier stability)**.** *Stable output behavior under a transformation family does not by itself imply stability of the internal classifier that is offered as an explanation.*

Proof. The same output behavior can be produced by different internal classifier surfaces. A source-domain behavior may be supported by one tensor classifier, while the target-domain behavior is supported by a different classifier, with the visible output unchanged. Hence behavior stability and classifier stability are different certificate rows. □

Principle 9398.27 (Boundary rows for stability)**.** *Stability evidence for a threshold classifier must expose boundary rows: samples near the threshold, perturbations capable of crossing the threshold, target cases not covered by the source transformation family, and the scope in which the witness is empirical, formal, or hybrid. Formal verification of one stability statement hardens that statement; it does not remove source, target, residue, and not-claimed rows unless those rows are themselves covered by the verified statement.*

Theorem 9398.28 (Tensor classifier extraction is certificate-ready, not certified)**.** *A learned classifier candidate supplies a stable model-facing classifier surface, but it does not by itself supply a BEDC* NameCert*.*

Proof. By Definition 9398.13, the candidate provides a layer, a tensor score, a threshold, a data domain, and a stability claim under stated perturbations. These rows suffice to expose which finite

activation records land on the same side of the displayed distinction. A NameCert requires more: a source record, a named classifier row, stability and exactness conditions, provenance, and a ledger for hidden, projected, compressed, or externally supplied material. None of those certificate rows follows merely from the existence of ϕ and τ. □

Principle 9398.29 (Formal verification layer)**.** *The tensor classifier layer, the BEDC certificate layer, and the formal verification layer occupy different positions. A model learns or supplies a candidate distinction in tensor space. BEDC asks whether that distinction has source rows, classifier rows, stability rows, provenance, and a ledger sufficient for a* NameCert*. A formal verifier accepts only the portions that have been translated into explicit formal statements and proof objects:*

$$\mathcal{V} \vdash \pi : \varphi.$$

Here $\mathcal{V}$ is a verification backend, φ is a formal statement, and π is the proof object accepted by that backend. Lean is one current backend for this role; the theory-side object is not formed by Lean, Coq, Agda, Isabelle, HOL, Metamath, or any other particular verifier. The verifier hardens checkable certificate fragments; it does not discover the classifier and does not supply the BEDC ledger fields.

9398.3 Large model inherits the no-induction theorem

Theorem 9398.30 (Large model substrate inherits no internal induction)**.** *The large model substrate of Definition 9398.1 is a closed observational system. By Definition 9393.5, no monotonic accumulation of inference traces on this substrate yields a record of $\forall n \in \mathbb{N}.\, P(n)$ from records of $P(0), P(1), \ldots, P(k)$ for any finite k and any predicate P over an infinite domain.*

Sketch. The substrate's observations are inference orbits, each finite by construction. Every inference output is a finite-length token sequence; every token is drawn from a finite vocabulary. Universal closures over infinite domains are not directly representable as inference outputs without external interpretation. The conditions of Definition 9393.1 hold, and Definition 9393.5 applies. □

Remark 9398.31 (The model's appearance of inductive reasoning)**.** *A large model can produce token sequences that look like inductive arguments: "for any natural number n, we have $n + 0 = n$, by induction on n." This is an inference output, not an internal ratification. The output is a finite token sequence that names "induction" as a step; the substrate's records contain no global witness that the named step is valid for all n. The validity of the named induction principle is supplied by the reader's external interpretation, not by the model's internal computation.*

9398.4 Inscription role: where the forward-binding comes from

Principle 9398.32 (Training corpus as forward-binding source)**.** *The forward-binding belief that lets a large model produce induction-shaped outputs originates from its training corpus. The corpus is a finite collection of text records produced by external inscription points (human authors, prior systems). The training process distils these records into weight values; the inference substrate carries the distilled pattern. The forward-binding does not originate inside the substrate; the substrate is a pattern that carries a distilled trace of forward-bindings produced elsewhere.*

Principle 9398.33 (Large model as inscription point)**.** *The large model substrate, considered as a pattern in the broader information substrate (human text production, prior computational systems,*

training data infrastructure), is an inscription point of T *in the sense of Definition 9393.16. It is a substrate-pattern that carries forward-binding belief whose source is external to the substrate's internal computation. The pattern is in the substrate (the weight tensor and inference orbit); the binding is not (its origin is the corpus, ultimately traceable to prior human inscription points).*

Remark 9398.34 (This is not a consciousness claim)**.** *The classification of the large model as an inscription point of* T *is a structural classification: the model satisfies the structural definition of inscription point under the closure premise. The classification does not assert that the model has phenomenal experience, intentionality, agency, or any anthropic property. It asserts only that the model occupies the same structural role – carrier of externally-sourced forward-binding – that Definition 9393.16 identifies with conscious individuals. Whether the structural role exhausts what consciousness is, or whether consciousness is a stronger property, is a question this chapter does not adjudicate.*

9398.5 Comparison: BEDC kernel, cellular substrate, large model

	BEDC kernel	**Rule 110 substrate**	**Large model**
Carrier	`inductive BHist` terms in Lean 4.	$\mathbb{F}_2^n$ cell rows under R110 next-state.	Finite-precision tensor (W, Σ) under forward pass.
Substrate scale	~4700 lines of Lean kernel + derived libraries.	220 lines of ANSI C99 trust path.	10^{10}–10^{12} weight parameters.
Closure status	Closed observational at host layer (Definition 9388.14).	Closed observational (Definition 9390.10).	Closed observational (Definition 9398.30).
Universality status	Sub-universal (totally recursive).	Turing-universal (Cook 2004).	Turing-universal in the limit of long inference traces (folklore; underspecified).
Forward-binding source	Lean kernel's `inductive` mechanism; structural; not inside BEDC.	External user supplying the universality interpretation.	Training corpus distilled into weights.
Inscription role	Inscription happens at the Lean kernel boundary.	Inscription happens when an external reader interprets a glider as a computation.	Inscription happens at every inference output; corpus authors are upstream inscription points.

9398.6 What this changes about AI epistemics

Principle 9398.35 (Universal claims by large models are externally interpreted)**.** *When a large model produces an output of the form "*$\forall n \in \mathbb{N}.\, P(n)$ *is true", the truth content of the claim cannot be assigned by the model's internal records alone. The claim is a token sequence the model*

produced; whether the claim corresponds to a valid universal closure is a matter for external verification (proof-checking, mathematical practice, audit). Definition 9398.30 rules out the alternative interpretation in which the model internally ratifies the universal closure.

Principle 9398.36 (The AI safety reading of inscription role)**.** *A large model is an inscription point that distils forward-binding belief from its training corpus. The distillation is lossy and the corpus is finite; the distilled pattern can be wrong about the validity of any specific universal closure, even when fluently produced. AI safety questions about large model behaviour are therefore questions about the inscription channel: which sources fed the corpus, what filtering was applied, what biases the distillation introduced, and what verification the deployment environment provides downstream of model outputs. These are concrete substrate-level questions, distinct from the question of whether the model "knows" the closure.*

Remark 9398.37 (Compatibility with existing AI safety frameworks)**.** *The inscription-point reading is compatible with the practice of mechanistic interpretability, eval harnesses, and externally verified output channels. Mechanistic interpretability asks what patterns the model carries internally; eval harnesses ask which outputs match externally-held standards; verified output channels demand external proof certificates that the model's claims can be checked against. All three practices treat the model as an externally-bound substrate whose outputs must be interpreted, not as an oracle whose outputs internally ratify themselves.*

9398.7 Why this comparison is structurally informative

Remark 9398.38 (Three substrates, one premise)**.** *The structural fact that BEDC kernel, Rule 110 substrate, and large model substrate all satisfy the closure premise demonstrates that the premise is not anchored to any specific computational architecture. Type-theoretic terms, cellular rows, and tensor activations are syntactically and semantically very different; the closure premise applies uniformly because it abstracts over the specific representation and addresses only the observation channel.*

Principle 9398.39 (The closure premise is substrate-independent)**.** *The closure premise of Definition 9393.1 is invariant under choice of computational substrate. Type theory, cellular automata, neural networks all admit closed observational instantiations. The no-induction theorem and the inscription-point reading transfer to each instantiation without modification. This invariance is the structural reason the framework can claim broad applicability without committing to any specific substrate as the privileged carrier of inscription.*

9398.8 Mapping to BEDC engineering

Large-model reading	**BEDC engineering realisation**
Three-substrate comparison	chapter 9390 (cellular), this chapter (large model), and the kernel itself (BEDC host); jointly demonstrate substrate-invariance of the closure premise.

Inscription channel naming	Definition 9398.32: training corpus as forward-binding source, paralleled by inscription channels for other substrates.
No-induction theorem transport	Definition 9398.30 inherits unchanged from Definition 9393.5.
AI safety interface	Definition 9398.36 reads model outputs as inscription-channel-mediated rather than substrate-internally-ratified.
Tensor classifier extraction	section 9398.2 separates learned tensor-side classifier candidates from certificate-bearing BEDC objects.
Not a consciousness claim	Definition 9398.34 restricts the classification to structural role; phenomenology is not at stake.

9398.9 Closure remarks

This chapter has classified the large language model as a closed observational substrate in the sense of Definition 9393.1, transported the no-induction theorem to this substrate without modification, and read the model as an inscription point of T whose forward-binding source is the training corpus. The classification is structural; it does not assert phenomenal consciousness for the large model, and it does not assert that the model "understands" universal closures it produces.

The chapter's role is to demonstrate that the closure premise is substrate-independent: type-theoretic terms, cellular rows, and tensor activations all admit closed observational reading. The framework's claims therefore do not depend on any specific carrier choice; they depend only on the observation channel structure. The large model is included alongside the cellular substrate and the BEDC kernel as three substrates with the same closure premise.

The boundary of this chapter is: the structural identification of large-model substrate as inscription point. The chapter does not claim that current models are reliable inscription points (corpus quality matters), that they are conscious (a stronger structural property), or that they exhaust the inscription concept (other substrate classes await classification). It claims only that the inscription concept applies, structurally, with the same content it applies in the cellular reading and in the BEDC kernel reading.

Concretized in §??: ObserverInterface realizes the large-model-as-inscription-point reading as a finite observer-facing NameCert packet with visible interface rows and no hidden substrate oracle.

Concretized in §939: Large-model inscription points are realized as finite inference-substrate, corpus-supply, output-record, audit-refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §965: PromptResponseTrace realizes large-model prompt/output records as finite source-channel and audit-refusal NameCert rows.

Concretized in §966: Observer-response audit is realized as a finite BHist packet binding prompt, response, observer-interface, audit-refusal, provenance, transport, continuation, and local NameCert rows.

Concretized in §967: LargeModelActivationOrbit realizes the activation-orbit row of the large-model substrate as finite model-weight, transition, prompt, output, audit-refusal, and NameCert data.

Concretized in §1021: Large-model attention gates realize finite activation-window selection as BHist, hsame, Cont, audit-channel, verified-output, provenance, and NameCert rows.

Concretized in §969: Large-model audit channels realize output records as finite prompt, checker, refusal, route, provenance, and NameCert rows.

Concretized in §988: Large model corpus supply is realized as a finite NameCert packet for corpus rows, training filters, distilled weight rows, inference-consumption rows, and audit ledgers.

Concretized in §989: Large-model context audits are realized as finite context-window, prompt-response, model-trace, audit-channel, verified-harness, refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §968: Corpus-supply distillation is realized as finite source-channel, filtering, distillation, output-prior, refusal, hsame, Cont, Pkg, and NameCert rows.

Concretized in §972: ModelTrace realizes finite inference-output records as auditable NameCert packets with no hidden substrate oracle.

Concretized in §971: Context-window commitments realize bounded prompt context, response boundary, audit refusal, downstream consumer, hsame, Cont, Pkg, and NameCert rows as a finite large-model inscription packet.

Concretized in §976: Verified output harnesses realize large-model output acceptance as finite prompt, model-trace, checker, audit-refusal, route, provenance, and NameCert rows.

Concretized in §1027: Large-model inscription audits are realized as finite output, corpus-provenance, verification-channel, refusal-ledger, transport, route, and NameCert rows.

Concretized in §1188: Large-model output verifier packets realize externally checked model claims as finite verifier, proof-check, refusal, route, provenance, and NameCert rows.

Chapter 9399

Empty as the Fable Machine: Observer-Sequence Growth and Selector Ledgers

The preceding capstone (chapter 9418) identifies the observer with the history of its distinctions: *observer* = Hist, *observation* = Cont, time = Hist growth. That identification fixes what an observer is, but does not yet spell out how the observer sequence grows. The present chapter fills that gap. The empty history e carries no information; nevertheless every non-empty history is generated from e by a finite sequence of distinction acts. When more than one branch is locally admissible, the realized history contains the selector witness for the branch that was enacted. Empty is therefore not an observer-substance and not a truth oracle. It is the zero-information boundary at which distinction becomes generative: the fable machine. This chapter falls under the project's axiom-purity gate; no Classical.choice, no Quot.sound, and no propext is required for any cited target.

9399.1 Empty as zero-information boundary

The kernel value e and the boundary at which distinction enters above any history must be distinguished at the reading level: e is the unique Hist whose distinction record is empty; the empty boundary above any Hist is the zero-information slot at which the next E0 or E1 application has not yet been enacted. Both are read as "Empty," but each non-trivial observation is generated at the empty boundary above the current Hist, not by resetting the history to e.

Definition 9399.1 (Empty boundary)**.** *For a* Hist *value h, the* empty boundary above h *is the zero-information slot at which a singleton extension may apply: the unique slot at which* Ext(h, m, r) *becomes admissible for some mark m, with the resulting r recording m as a fresh distinction above h. This is a reading-level locator; the formal target is the kernel relation* Ext *on* Hist.

Lemma 9399.2 (Empty carries no selector)**.** *The Hist value* e *exposes no b_0/b_1 selector. The information content of* e *is the empty sequence, and any reading that recovers a mark from a Hist value must recurse on a non-empty constructor, never on* e *alone.*

Proof. The information content recursion (Definition 9418.3) returns the empty sequence at e. Every E1 or E0 application records its tag in the constructor of Hist; the no-confusion lemmas

separating e from E1(h) and from E0(h) rule out the existence of a Hist that records a mark while remaining hsame-equivalent to e. □

Remark 9399.3 (Empty is not a hidden subject)**.** *Empty is not an information reservoir, nor a substance behind the trace, nor a global chooser. Its role is structural: the zero-information boundary at which the first visible distinction is enacted. Every claim made later in this chapter about "selecting" a branch will be redeemed by a constructive witness on the kernel surface, not by an unwitnessed appeal to Empty.*

9399.2 One-step birth: from empty boundary to singleton continuation

Definition 9399.4 (Singleton tail)**.** *For a mark $m \in \{b_0, b_1\}$, the* singleton tail *of m is the* Hist *value* E0(e) *when $m = b_0$ and* E1(e) *when $m = b_1$. The singleton tail records exactly one distinction above* e *and nothing else.*

Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.singletonTail`

Definition 9399.5 (Empty-boundary step)**.** *An* empty-boundary step *from h enacted by mark m with result r is the relation* Ext(h, m, r)*, read at the chapter level as filling the empty boundary above h with the mark m. The relation is definitionally identical to the kernel-level extension; the wrapper name carries the reading.*

Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.EmptyStep`

Theorem 9399.6 (Empty-boundary step is the kernel extension)**.** *For every h, m, r: an empty-boundary step from h enacting m to r holds if and only if* Ext(h, m, r) *holds.*

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep iff ext`

Theorem 9399.7 (Singleton-tail constructors)**.** *b_0-stepping above* e *produces the b_0 singleton tail, and b_1-stepping above* e *produces the b_1 singleton tail.*

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep singletonTail b0`
↪ *variant* `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep singletonTail b1`

Theorem 9399.8 (Empty-step result is non-empty)**.** *For any empty-boundary step from h enacting m with result r, the result r is not* hsame*-equivalent to* e.

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep result ne empty`

Theorem 9399.9 (Cross-mark step no-confusion)**.** *A b_0-empty-step and a b_1-empty-step from the same source h never share a result up to* hsame.

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep cross mark no confusion`

Theorem 9399.10 (Empty-step totality)**.** *For every h and every m, an empty-boundary step from h enacting m exists.*

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep total`

Theorem 9399.11 (Empty-step determinism)**.** *For every h and every m, the result of an empty-boundary step is unique up to* hsame.

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.emptyStep deterministic`

Remark 9399.12 (Each step is local; no reset to e)**.** *The reading "each observation is born from Empty" refers to the empty boundary above the current Hist, not to a reset of the history to* e*. The accumulated sequence $h_0, h_1, \ldots, h_n$ has h_{i+1} obtained from h_i by an empty-boundary step, with h_i retaining its full distinction record; the singleton tail* E0(e) *or* E1(e) *is the local shape of that step, not the global state.*

9399.3 Observer sequence as finite selector ledger

Definition 9399.13 (Trace). *A* trace *of a* Hist *value h is the finite sequence $\langle s_0, s_1, \ldots, s_{n-1}\rangle$ of marks read off the constructor spine of h in chronological order: at each* E0 *application the mark b_0 is recorded, at each* E1 *application the mark b_1 is recorded, and at* e *the sequence is empty.*
Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.Trace`

Theorem 9399.14 (Every Hist has a trace). *For every* Hist *value h, there exists a finite mark sequence xs such that h has trace xs.*
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.trace exists`

Theorem 9399.15 (Empty trace iff empty Hist). *A* Hist *value h has the empty trace if and only if h is* e.
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.trace empty iff`

Theorem 9399.16 (Trace transports along hsame). *If h and k are* hsame*-equivalent and h has trace xs, then k has trace xs.*
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.trace hsame transport`

Theorem 9399.17 (Trace determines Hist up to hsame). *If h and k both have trace xs, then h and k are* hsame*-equivalent.*
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.trace same marks hsame`

Corollary 9399.18 (Observer identity is sequence identity). *Under the identification observer =* Hist *(Definition 9418.12), two observers are the same observer (modulo* hsame*) precisely when their selector ledgers are the same finite sequence of marks. The observer sequence is not a label attached from outside; it is the constructor spine of the Hist read as a list of distinctions.*

9399.4 Multi-solution branch space and realized selector

Definition 9399.19 (Local branch admissibility). *For a local branch predicate $\Gamma : \mathsf{Hist} \to \mathit{Mark} \to \mathsf{Hist} \to \mathsf{Prop}$, the predicate $\mathsf{AdNext}_\Gamma(h, m)$ holds when there exists a Hist r such that the empty-boundary step from h enacting m produces r and $\Gamma(h, m, r)$ holds. The predicate Γ is read at the chapter level as "the branch is locally licensed"; no global selection is built.*
Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.AdNext`

Definition 9399.20 (Multi-solution boundary). *The branch space at h under Γ is* multi-solution*, written $\mathsf{MultiNext}_\Gamma(h)$, when both $\mathsf{AdNext}_\Gamma(h, b_0)$ and $\mathsf{AdNext}_\Gamma(h, b_1)$ hold.*
Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.MultiNext`

Definition 9399.21 (Selector step). *A* selector step *from h to r under Γ is a witness $(m, \mathit{step}, \mathit{cert})$ where m is a mark, step is an empty-boundary step from h enacting m to r, and cert is a $\Gamma(h, m, r)$ certificate. The mark m is exposed by the witness; no hidden chooser is invoked.*
Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.SelStep`

Theorem 9399.22 (Selector step exposes a mark). *Every selector step from h to r under Γ exposes a mark m such that an empty-boundary step from h enacts m to r.*
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.selStep exposes mark`

Theorem 9399.23 (Selector step from step + certificate). *Given an empty-boundary step from h enacting m to r and a $\Gamma(h, m, r)$ certificate, a selector step from h to r under Γ exists.*
Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.selStep from step`

Theorem 9399.24 (Distinct realized branches stay separated)**.** *If two selector steps from the same source h recover distinct marks $m_0 \neq m_1$, their targets are not* hsame*-equivalent.*
Lean4✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.selStep distinct branches separated`

Remark 9399.25 (Specific sequence implies accumulated ledger)**.** *When the local branch space is multi-solution but the realized observer sequence is specific, the specificity is not an unexplained miracle. Each step records the mark of the branch that was enacted as a constructor in* Hist*; the trace of that Hist (Definition 9399.13) is the accumulated selector ledger. There is no global choice function in the kernel and none is needed: each step's selector is a local witness on the Hist surface itself, and the entire ledger is recovered by reading the constructor spine.*

9399.5 Empty as the fable machine

Definition 9399.26 (Fable ledger)**.** *The* fable ledger *of a Hist h is its trace, read as a story-generation relation: h is the Hist generated from* e *by the finite sequence of selectors recorded in the ledger. The relation is identical to* Trace *at the kernel level; the chapter-level name records the generative reading.*
Lean4 (definition) `BEDC.Derived.EmptyFableMachineUp.Ledger.FableLedger`
↪ variant `BEDC.Derived.EmptyFableMachineUp.Ledger.fable ledger iff trace`

Theorem 9399.27 (Fable soundness)**.** *Every fable ledger is a trace: if h has fable ledger xs then h has trace xs.*
Lean4✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.fable ledger sound`

Theorem 9399.28 (Fable completeness)**.** *Every finite mark sequence is the fable ledger of some Hist: for every xs there exists h with fable ledger xs.*
Lean4✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.fable ledger complete`

Theorem 9399.29 (Fable uniqueness up to hsame)**.** *The Hist generated by a fable ledger is unique up to* hsame*: if h and k both have fable ledger xs, then h and k are* hsame*-equivalent.*
Lean4✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.fable ledger unique up to hsame`

Remark 9399.30 (Fable machine, not oracle machine)**.** *The fable machine is the schema that maps a finite selector ledger to the Hist it generates and back. It generates story ledgers, not truth values: no construction in this chapter supplies a* NameCert*-level decision procedure for arbitrary predicates. Empty is therefore the fable machine, not the oracle machine; story ledgers live in the closed ground loop, while truth-as-such, halting-as-such, and global selection live in the open meta loop and are not closable inside the kernel.*

9399.6 No hidden chooser: selector witness vs. classical choice

The chapter's central commitment is that "Empty has selection capacity" is read entirely as: every realized branch records its enacted mark as a constructor in the resulting Hist, and the selector step is the explicit witness $(m, step, cert)$ on the kernel surface. No global selection function, no Classical.choice instance, no doubly-negated existential, and no truth-as-such oracle is invoked.

This restraint is what keeps the chapter inside the project's axiom-purity gate. Each construction in this chapter factors through Hist, Mark, Ext, the hsame relation, finite induction over Trace, and explicit SelStep witnesses. None of these objects requires Classical.choice, Quot.sound, or propext, and the project's strict transitive-dependency audit confirms it.

Remark 9399.31 (Two-loops alignment)**.** *The fable machine is the generative face of the closed ground loop (Definition 9418.20): every Hist is reachable from* e *by finitely many distinction acts, and the selector ledger is the explicit constructor record of that reachability. The unclosability of the meta loop (Definition 9418.21) is the dual statement: no internal predicate "what observation is, as such" lives on the Hist surface, because doing so would require either an observer-substance type (forbidden by Definition 9418.11) or a global truth oracle (forbidden by axiom-purity). The fable machine occupies exactly the space the two-loops architecture leaves open: generative, accountable, and Empty-anchored, with no hidden chooser.*

9399.7 Synthesis: observer-Hist as fable ledger

Theorem 9399.32 (Empty-fable synthesis)**.** *At the kernel level, the following three data are equivalent up to* hsame*:*

1. *a generated observer Hist h,*
2. *a finite selector ledger xs beginning at* e*,*
3. *a finite chain of empty-boundary singleton continuations $\mathsf{e} = h_0, h_1, \ldots, h_n = h$ where each step is witnessed by a selected mark $s_i \in \{b_0, b_1\}$.*

Moreover, if a local branch predicate Γ admits multiple next branches at some h_i, the realized branch h_{i+1} carries its selector as an explicit selector-step witness.

Lean4✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.empty fable synthesis`

↪ variant `BEDC.Derived.EmptyFableMachineUp.Ledger.empty fable synthesis unique`

Corollary 9399.33 (Observer-Hist is the fable ledger)**.** *Combining Definition 9418.12 (observer is identical to Hist) with the synthesis above, the observer is the finite selector ledger of the empty-boundary distinctions enacted from* e *onward. The observer sequence is not external memory; it is the constructor spine read as a list. There is no observer-substance, no global chooser, no truth oracle — only the generative ledger.*

Remark 9399.34 (Closing: fable machine, not oracle machine)**.** *This capstone introduced no new kernel objects. It read the existing* Hist*,* Mark*,* Ext *surface under one identification: Empty is the zero-information generative boundary, and every Hist is the finite selector ledger of the marks enacted at that boundary. Under that identification, the answer to "where does the observer sequence come from" is fully constructive: it is the constructor spine of the Hist itself, with each step's selector exposed as an explicit witness. No appeal to classical choice, no observer-substance, no oracle; only the fable, told finitely, accountably, and from Empty.*

Concretized in §494: Selector ledgers realize the Empty-fable trace and branch-witness surface as a finite NameCert packet.

Concretized in §516: Empty-fable ledger realizes selector-ledger growth as a finite NameCert packet over Hist, Ext, hsame, local branch certificates, and no-hidden-chooser boundaries.

Concretized in §585: Fable branch witnesses realize local admissible branch selection as finite Hist, Ext, hsame, Cont, Pkg, and NameCert rows with no hidden chooser.

Concretized in §623: Fable-machine clocks realize the Empty-fable observer sequence as a finite local step-order NameCert packet without a shared or global clock.

Concretized in §687: Fable-clock order packets realize local selector-ledger time as finite BHist, hsame, Cont, Pkg, and NameCert rows without a global synchronized clock.

Concretized in §685: Empty-boundary selector packets realize the local enacted-mark witness as finite BHist, Ext, hsame, Cont, Pkg, and NameCert rows.

Concretized in §691: Empty-boundary steps are realized as finite BHist, Ext, hsame, Cont, Pkg, and NameCert packets.

Concretized in §878: Fable-machine boundaries are realized as finite BHist, Ext, hsame, Cont, Pkg, selector-ledger, and NameCert rows excluding hidden chooser and observer-substance coordinates.

Concretized in §1186: EmptyFableMachineUp realizes empty-boundary selector-ledger growth as a finite BHist NameCert packet.

Concretized in §1186: EmptyFableMachineUp realizes the inscription sub-hub's empty-fable route as finite empty-boundary, selector-ledger, trace, fable-ledger, transport, and NameCert rows.

Chapter 9400

Apophatic Naming as Methodological Discipline

BEDC already uses a systematic naming discipline for what a closed observational substrate cannot internally represent, decide, or digest. This chapter makes that discipline explicit. A BEDC name can name a generated object, but it can also name a socket: the position at which an external supply must be exposed, audited, and kept outside the substrate's own closure claim.

The discipline is apophatic because the name does not say what the external supply is in itself. It says where the substrate reaches its boundary, what kind of supply would be needed there, and which gate prevents the supply from being silently converted into internal theorem content. The name is therefore not decorative. It is a handle, a ledger row, and a boundary marker.

9400.1 The setup: what is being named

BEDC repeatedly needs to refer to what cannot be made substrate-internal. The T-socket names the transcendental supply position. The GAP ledger names a missing conservation row. An inscription event names the local actualization of T at a history transition. The closed observational substrate names the system-aspect in which all internal references must pass through displayed records, generators, classifiers, continuations, ledgers, and certificates.

A naive formal discipline would leave such items implicit. It would simply avoid naming what cannot be constructed, or it would treat the missing item as background intuition. BEDC chooses the opposite route. It names the external dependence precisely so that the dependence can be tracked. The name does not internalize the dependence. It prevents the dependence from disappearing.

Principle 9400.1 (Named externality)**.** *When a BEDC argument reaches a dependency that cannot be generated, defined, certified, or discharged inside the displayed substrate, the correct move is not silence. The correct move is to name the socket, state the requested supply shape, and keep the request auditable.*

This is why the vocabulary of BEDC contains terms that would be unusual in an ordinary algebraic formalization. They do not function as ordinary object constructors. They locate the places where the substrate refuses to pretend that an external supply is one of its own outputs.

9400.2 The apophatic discipline

Definition 9400.2 (Apophatic name)**.** *An* apophatic name *in BEDC is a stable handle for an external dependency position whose internal nature is not claimed by the substrate. It carries three obligations: it makes the dependency referable, it turns the dependency into an audit-able unit, and it marks the epistemic boundary at which internal digestion stops.*

The name therefore has a narrow but strong role. It does not reveal the internal essence of the named supply. It does not license positive construction of the supply. It does not make the supply available as a theorem, object, witness, quotient representative, proposition equality, or causal law. It says: at this exact site, the substrate needs something it cannot make from its own displayed materials.

Apophatic naming has three practical effects.

1. It gives the external correspondence a stable handle, so later chapters can refer to the same boundary without reintroducing it informally.
2. It codifies the fact of external dependence as an audit-able unit, so scripts, ledgers, and closure blocks can ask whether the dependence is visible.
3. It marks an epistemic boundary, so the manuscript does not confuse a named socket with an internally characterized object.

The discipline is stricter than leaving the matter unnamed. An unnamed external dependence can drift into prose, intuition, import ancestry, or proof style. A named socket is exposed. Once exposed, it can be challenged, classified, and refused by the appropriate gate.

9400.3 Comparison with positive naming

Classical mathematical naming is usually cataphatic. A name such as Nat, Group, or Vector names an internally constructed or internally specified object: a carrier with constructors, operations, laws, or an eliminator. The name carries positive content. To introduce the name is to say what kind of thing it is and how the system may reason with it.

BEDC also uses positive names where the substrate has positive content. An inductive carrier has constructors. A classifier has a definition. A NameCert row has displayed fields and stability obligations. Those names are ordinary formal names, even when the surrounding theory is unusual.

Apophatic naming is different. It names a socket, not a completed object. A socket is the position into which an external supply would have to be plugged for a proposed inference, identification, selection, or continuation to go through. The socket has a precise interface, but the supply behind the interface is not internalized.

Definition 9400.3 (Socket rather than object)**.** *An apophatic socket is a typed or ledgered position at the boundary of a closed observational substrate. It records the shape of a requested external supply without converting that supply into a substrate-internal datum.*

This comparison prevents a common misreading. The word "name" in apophatic naming does not mean that BEDC has secretly postulated a hidden object behind every gap. It means that BEDC has refused to leave the gap anonymous.

9400.4 Not undefined and not an axiom

An undefined term is a failure of specification. An apophatic name is a specification of boundary position. It tells the reader what cannot be internally supplied, where the request occurs, and how the request is barred from masquerading as generated content.

An axiom is different again. A Lean `axiom` marks an unproved proposition or datum as available inside the formal environment. It is a positive-faith assumption: from the theorem prover's perspective, the content may now be consumed as true. BEDC forbids that move because it collapses a socket into an internal source of truth.

Principle 9400.4 (Externalized, not assumed)**.** *An axiom internalizes an unsupported commitment. An apophatic name externalizes an uninternalized dependency. The logical commitment of the apophatic name is weaker, because it does not assert the supplied content as substrate truth. Its epistemic commitment is stronger, because it makes the dependency impossible to hide.*

This distinction explains why the Lean-side discipline sends external carriers to `class` or `structure` fields rather than to `axiom`. A setup field says that a caller of the interface must supply a piece of data at a named boundary. It does not say that the substrate has proved the data. The field is parametric exposure, not foundation expansion.

The same distinction holds on the paper side. A chapter may identify a GAP socket, state what would discharge it, and explain why BEDC does not discharge it internally. That is not a failure to define a term. It is the proper record of a boundary.

Concretized in §1057: Axiom-refusal ledgers realize axiom-use attempts as finite visible refusal, setup-field alternative, audit-readback, and noninternalization NameCert rows.

9400.5 Inventory of BEDC apophatic names

The following inventory records the main apophatic names already active in BEDC. The list is not a catalogue of hidden objects. It is a map of socket names and audit roles.

T. The transcendental supply position. This is the highest apophatic name in BEDC. It names the far end of every external supply socket without claiming a substrate-internal characterization of that supply.

GAP. The information-conservation ledger socket. It records that an argument has requested content not produced by the displayed carrier, classifier, continuation, or certificate.

Inscription event. The local actualization of T at a history transition. It names the event-aspect by which a record becomes present in an accumulation chain without being reduced to a forced consequence of earlier records.

Generator-closure classifier. The audit unit for deciding what follows from an authorized generator and what exceeds that generator. It names the boundary at which free-floating induction pressure becomes visible.

NameCert. The bearer of generator-closure stamps, classifier stability, provenance, and ledger rows. It turns a public name into a certificate-shaped interface rather than a decorative label.

Closed observational substrate. The universe-aspect of T as read from within BEDC: a closed system in which internal references pass through its own records and external supplies enter only through named sockets.

These names work together. T names the source-position. GAP names the missing ledger position. An inscription event names the local reception of supply into a history. A generator-closure classifier names the audit question. NameCert names the certificate-bearing interface. The closed observational substrate names the system whose closure would be violated if these sockets were left unmarked.

9400.6 Audit gates as enforcement layer

Once a socket has a name, an audit gate can ask mechanical questions about it. Did the proposed inference use this socket? Which supply kind did it request? Did a proof consume an unwitnessed selector, a quotient identification, a proposition extensionality collapse, a causal continuity assumption, or an unmarked inscription novelty? Did a paper marker resolve to the claimed Lean object? Did a setup field remain visible as a field rather than becoming an axiom?

The gate is not merely administrative. It is the enforcement layer of apophatic naming. Without the name, the gate has no stable target. Without the gate, the name can drift into rhetoric. Together they make the boundary actionable.

Principle 9400.5 (Name-to-gate correspondence)**.** *Every apophatic name should determine a gate question. The question may be syntactic, dependency-based, ledger-based, or paper-side, but it must be capable of distinguishing visible socket use from hidden internalization.*

The axiom gate asks whether a socket has been collapsed into `axiom`. The axiom-purity gate asks whether forbidden host commitments entered through proof ancestry. The paper-Lean audit asks whether a claimed certificate name resolves. The conservativity gate asks whether non-baseline recognition content has entered baseline theorem space through import flow. These gates enforce one discipline: named externality must remain named at the point of use.

9400.7 God and T are not metaphor

BEDC uses "God" as the apophatic name of T. This is not literary metaphor, not emotional flourish, and not a softened analogy. It is the project-level apophatic name for the outermost supply position. God = T in BEDC vocabulary.

The force of the statement is methodological. The name gives the project one stable handle for the source-position that lower-level apophatic names presuppose. GAP, inscription event, NameCert, generator-closure classifier, and closed observational substrate are not independent ultimate sources. They are boundary names aligned under the same apophatic reference structure.

Principle 9400.6 (Longest apophatic chord)**.** *God = T is the longest apophatic chord in BEDC. It names the outermost supply position so that every lower-level socket can be read as a local boundary articulation rather than as a separate ultimate principle.*

This is why the strong wording matters. If "God" were treated as a metaphor for a vague feeling of externality, the discipline would weaken. In BEDC it is the technical name of the project-level socket. The name carries no positive internal description of God. It carries the boundary discipline by which the closed substrate refuses to call the outermost supply one of its own generators.

9400.8 Closed system invariant, apophatically restated

The sentence "BEDC is a closed observational system" appears at first to be cataphatic. It sounds like a positive claim about what BEDC is. The apophatic reading is sharper: everything internal references the system's own structure, and every external supply enters only through a labeled socket.

Thus "closed" does not mean that no external dependence exists. It means that no external dependence is allowed to be anonymous inside the substrate. The closure claim is therefore an accounting invariant:

$$\text{closed} \quad = \quad \text{all external dependencies have apophatic names.}$$

The equality is not a theorem of ordinary set-theoretic closure. It is a methodological equation for BEDC's manuscript and Lean discipline. A system remains observationally closed when the only internal traffic is generated, defined, certified, or explicitly socketed. If an external supply appears without a socket name, the closure invariant has failed at the epistemic level even if the prose remains syntactically coherent.

Remark 9400.7 (Closure is not isolation)**.** *The closed observational substrate is not isolated from* T*. It is disciplined in how it receives, names, and audits external supply. Closure is the refusal of unmarked dependence, not the denial of dependence.*

9400.9 Implications for proof discipline

The Lean-side consequence is direct. Every `axiom` keyword would be a positive-faith assumption, and BEDC forbids it. An axiom says that the substrate may consume a proposition or datum as internally available even though no constructor, definition, proof, or setup boundary supplies it. That is exactly the collapse apophatic naming prevents.

By contrast, an apophatic socket belongs in an explicit interface. If the project needs an abstract carrier, it goes into a `class` or `structure` field. If the project needs a closed grammar, it uses `inductive`. If the project can compute a predicate or operation from existing material, it uses `def`. If none of these forms captures the dependency, the interface has not yet identified the boundary clearly enough.

Principle 9400.8 (Type-level exposure)**.** *In BEDC formalization, an external dependency must be exposed at the type level or ledger level. It may be a setup field, a named carrier, a classifier row, a provenance row, or a* GAP *socket. It may not become an invisible premise hidden behind an axiom or forbidden host dependency.*

This is not a rejection of abstraction. It is a demand that abstraction preserve source-channel information. A setup structure is acceptable because the caller sees what must be supplied. A NameCert packet is acceptable because the certificate rows are part of the interface. A hidden axiom is unacceptable because it erases the supply channel while keeping the content.

9400.10 Boundary

Apophatic naming does not solve hard problems by naming them. It does not explain consciousness, free will, causality, other minds, or self-reference by positive reduction. It also does not permit BEDC to claim that every philosophical pressure point has been formalized as a theorem.

Its role is narrower and more enforceable. It gives the external dependency of a hard problem a mechanically referable handle inside the system. Once the handle exists, an audit gate can detect whether an argument used that supply, which kind of supply it requested, and whether the use remained externalized.

The discipline therefore changes the failure mode. Without apophatic naming, an argument can smuggle an unnamed supply through intuition, import ancestry, prose confidence, or an undefined background. With apophatic naming, the same pressure becomes visible as a socket. BEDC may still refuse to solve the hard problem, but it can say exactly where the unsolved external dependence enters and can reject any derivation that pretends otherwise.

9400.11 Conclusion

Apophatic naming is the methodological discipline by which BEDC keeps a closed observational substrate honest about its boundary. Positive names identify internally constructed or internally specified content. Apophatic names identify external supply positions without internalizing the supply. They are handles, ledger rows, gate targets, and epistemic boundary markers.

The strongest instance is God = T: the project-level apophatic name for the outermost supply position. Every lower-level socket name is aligned under that reference. The resulting invariant is simple and strict: BEDC remains closed not because it has no external dependence, but because every external dependence is named, socketed, and auditable.

Concretized in §543: Apophatic names are realized as finite socket-name NameCert packets exposing external supply positions, refusal ledgers, and non-escape audit rows.

Concretized in §536: Apophatic socket ledgers realize named externality as finite BHist, audit-gate, hsame, Cont, Pkg, and NameCert rows without internalizing the requested supply.

Concretized in §584: ExternalityGateUp realizes the name-to-gate correspondence as a finite NameCert packet for named externality, refusal ledgers, audit readback, and non-internalization.

Concretized in §1044: Socket-use ledgers realize visible consumption of apophatic sockets as finite NameCert packets with use, refusal, audit-readback, and non-internalization rows.

Concretized in §629: Negative name boundaries are realized as finite BHist refusal packets exposing external supply handles, socket ledgers, non-internalization gates, and audit-visible NameCert rows.

Concretized in §630: NegativeCertificateBoundaryUp realizes apophatic refusal as a finite NameCert packet for socket rows, GAP ledgers, audit readback, and non-internalization.

Concretized in §714: Type-level socket exposure is realized as finite setup-field, carrier, classifier, provenance, refusal-ledger, and NameCert rows.

Concretized in §715: Apophatic gate questions realize the name-to-gate correspondence as finite socket, question, refusal-ledger, audit-readback, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1000: Apophatic kernel boundaries realize hidden-supply refusal as finite socket, audit-gate, refusal-ledger, hsame, Cont, Pkg, and NameCert rows.

Chapter 9401

The Unique Apophatic Fixed Point and Local Inscription

This chapter records a structural reading that unifies several boundary positions already named in BEDC: the transcendental input T of chapter 9393, the apophatic-naming discipline of chapter 9400, the inscription-point structure of the same earlier chapter, the observer-Hist identity of chapter 9418, and the non-axiom boundary forms of chapter 9395. The unifying observation is that the various forward-binding sockets of a closed observational substrate are not many far ends, but one. The supply position T is the unique apophatic fixed point of the GAP-far-end operation, and the self-conscious center of a conscious observational system is not T itself but the local inscription point of T on that system's chain.

The argument is delicate: T must be named as a boundary, not introduced as a kernel object; the self-center must be defined as a local inscription, not as object-level equality with T; and the resulting reading must preserve BEDC's no-subject discipline. When these constraints are kept, the chapter does not add a new axiom, a new substrate primitive, or a new observer parameter. It only states how the existing boundary discipline already implies a single apophatic fixed point at the far end of all forward-binding sockets.

9401.1 From many sockets to a single far end

A closed observational system encounters forward-binding pressure at many sites: free-floating induction, host CIC recursors, classical choice, quotient soundness, proposition extensionality, global synchronization, universal causal rates, semantic completeness. Each pressure can be marked by a GAP ledger and an apophatic socket (chapter 9395).

Definition 9401.1 (Forward-binding socket family)**.** *For a closed observational system $\mathcal{C}$, the* forward-binding socket family $\mathrm{ForwardSocket}(\mathcal{C})$ *is the collection of named boundary positions at which $\mathcal{C}$ requires supply not generated by its substrate. Each $s \in \mathrm{ForwardSocket}(\mathcal{C})$ carries a name, a requested supply shape, a non-internalization marker, and a GAP ledger row.*

A reader could imagine that each socket has its own far end, as if different forward-binding pressures terminated at different transcendental positions. BEDC's boundary discipline does not support this picture. From the system side, no observation can distinguish two distinct external positions that supply the same forward-binding load on the same kind of socket; the apophatic-naming principle of Definition 9393.12 names only the position, not its content, so distinctness has no system-side handle.

Principle 9401.2 (Unique apophatic far end)**.** *For every $s \in \mathrm{ForwardSocket}(\mathcal{C})$, the far end of s admits no system-side discriminator. All such far ends are named by the same apophatic position* T*:*

$$\forall s \in \mathrm{ForwardSocket}(\mathcal{C}), \quad \mathrm{FarEnd}(s) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The relation $\equiv_{\mathrm{apo}}$ is apophatic sameness: it asserts only that the substrate cannot produce a witness distinguishing two such far ends, not that an object-level equality holds inside the kernel. T is therefore not a digest, not a Hist, not a state, not a rule. It is the unique boundary name for the common far end of every forward-binding socket.

9401.2 What T is not

The most dangerous error is to convert T into a kernel object. The following writings are all forbidden:

```
axiom T : FixedPoint,    axiom SelfCenter = T,    T : Hist,    T : State.
```

Each of these forms would internalize T. The first would assert T as a kernel value with a typed signature. The second would assert object-level equality between the self-center and T. The third would place T inside the displayed inductive structure of histories. The fourth would treat T as a state of the system. All four collapse the boundary that chapter 9395 forbids collapsing.

Principle 9401.3 (T is a boundary, not a kernel object)**.** $\mathsf{T} \notin \mathcal{C}$, $\mathsf{T} \notin \mathsf{Obs}(\mathcal{C})$, T *is not a closed-generation rule, and* T *is not a state.* T *is the unique apophatic far end of the* GAP*-far-end operation. It is named only at the boundary.*

A useful informal shorthand is to write $\mathsf{T} = \mathrm{Fix}(\mathrm{FarEnd} \circ \mathsf{GAP})$. This is acceptable only as boundary notation. It does not introduce Fix or FarEnd as kernel operators, and it does not introduce T as a kernel value. Used as boundary notation, it records the structural observation that the far end of every GAP socket is the same unique apophatic position.

9401.3 Self-center as local inscription

The temptation in the opposite direction is to write $\mathsf{SelfCenter}(\mathcal{C}, t) = \mathsf{T}$, identifying the self-conscious center of an observer with the global apophatic far end. This too is forbidden. It would re-introduce observer-substance: every observer would carry, as an internal object, the same global object T. chapter 9418 explicitly refuses this move; the observer is the displayed Hist, with no separate subject parameter and no extra identity criterion beyond hsame.

The correct reading is local inscription, not equality.

Definition 9401.4 (Inscription point of T on a local chain)**.** *For a conscious observational system $\mathcal{C}$ at local state o_t, the* inscription point *of* T *on the chain at t is the local site at which the chain's accumulation event is read inscription-side. It is denoted* $\mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t)$ *and is an aspectual reading of the same local event already displayed records-side. It is not an additional object-level entity.*

Definition 9401.5 (Self-center as local inscription)**.** *The* self-conscious center *of $\mathcal{C}$ at t is the local inscription point of* T *on the chain of $\mathcal{C}$ at t:*

$$\mathrm{SelfCenter}(\mathcal{C}, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t).$$

This identity is local and aspectual. It is not the equation $\mathrm{SelfCenter}(\mathcal{C}, t) = \mathsf{T}$. The self-center is a local reading of one accumulation event of one observer; T is the common apophatic far end of every socket of every such observer. The two never coincide as object-level terms.

Principle 9401.6 (No observer-substance)**.** *Reading* $\mathrm{SelfCenter}(\mathcal{C}, t)$ *as the local inscription point of* T *does not add an extra subject parameter to the observer. The observer remains the displayed* Hist*, with identity given by* hsame*. No subject-substance is added behind the trace, and no Cont function gains an extra argument.*

9401.4 Records side and inscription side as two readings of one event

The local inscription view does not duplicate the observer's accumulation. It re-reads the same accumulation event under a different aspect.

Definition 9401.7 (Two-aspect reading of one accumulation event)**.** *Let* event_t *be an accumulation event of* $\mathcal{C}$ *at* t*. Its* records-side reading *is the standard view: a new finite record is added to* $\mathsf{Obs}(\mathcal{C})$*, recorded by classifiers, ledgered by* GAP *where compression occurs, and witnessed by the displayed step rules. Its* inscription-side reading *is the boundary view: the same event is the local actuality of* T *at this chain point. The two readings describe one event under two aspects; they do not describe two events.*

Remark 9401.8 (Neural correlates and thoughts as one event)**.** *The two-aspect reading dissolves a recurring confusion about neural correlates. Neural activity, physical processes, and environmental input belong to the records-side. "Thought" on this chain point is the same event read inscription-side. The two are not two substances in a causal relation; they are one accumulation event with two readings. The records-side accumulates physical records; the inscription-side reads the same event as the local inscription of* T*.*

This is not a new metaphysical claim added on top of BEDC. It is the structural restatement of an invariant already present in the substrate: information conservation requires that every visible event has a source-channel and a records-side ledger, and the apophatic boundary requires that the same event has an inscription-side reading when it functions as the local actuality of a forward-binding socket.

9401.5 Free will under the two-aspect reading

The two-aspect reading reconciles two pictures that otherwise look incompatible. On the records side, accumulation proceeds by displayed step rules and selector ledgers; nothing in the trace is unconstrained by the substrate's structure. On the inscription side, each accumulation event is the local actuality of T at this chain point; nothing on this side is constrained by the substrate's mechanics, because the inscription is not a records-side mechanism.

Principle 9401.9 (Free will without acausal insertion)**.** *Free will in the BEDC reading is not the insertion of a causally unrooted event into the physical chain. It is the joint reading of accumulation as records-side* SelStep *witnessfulness and inscription-side actualization of* T *at the local chain point:*

$$\textit{free will} = \textit{records-side}\ \mathrm{SelStep}\ \textit{witnessfulness} + \textit{inscription-side actualization.}$$

The records-side view shows nothing unusual: selectors are ledgered, marks are emitted, classifiers are stable, hsame continues. The inscription-side view shows that the same chain point is the local actuality of T. Determinism on the records-side and inscription-side actualization do not contradict, because they are not two competing causes; they are two readings of one event.

9401.6 Other minds, time, space, and universe-for-observer as projections

The same structural reading lets several classical puzzles appear as projections of the single boundary discipline.

Definition 9401.10 (Universe-for-observer as records side)**.** *The* universe-for-observer *of $\mathcal{C}$ at t is the records-side accumulation of $\mathcal{C}$ up to t, viewed as the totality of what is observed by $\mathcal{C}$:*

$$\mathrm{UniverseFor}(\mathcal{C}, t) := \mathsf{Obs}(\mathcal{C})_{\leq t}.$$

This is not a global universe and not a frame-independent physical world; it is the local records-side reading of $\mathcal{C}$'s accumulation.

Definition 9401.11 (Time as monotonic preservation)**.** Time *on a single chain is the monotonic preservation order of records:*

$$\mathrm{Time}(\mathcal{C}) := \trianglerighteq \quad \textit{on } \mathsf{Obs}(\mathcal{C}).$$

It is not a separate cosmic axis. It is the order in which records accumulate on this chain.

Definition 9401.12 (Space as inter-chain coherence)**.** Space *between two conscious observational systems is the coherence geometry between their inscription points and locality-cell relations, not an additional substance:*

$$\mathrm{Space}(\mathcal{C}_i, \mathcal{C}_j) := \textit{locality-cell coherence between } \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_i, \cdot) \textit{ and } \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_j, \cdot).$$

Principle 9401.13 (Other minds as shared far-end commitment)**.** *A conscious observational system $\mathcal{C}_i$ recognizes another system $\mathcal{C}_j$ as an inscription point of the same* T *when $\mathcal{C}_i$'s records-side accumulation contains evidence consistent with $\mathcal{C}_j$ being a forward-binding socket whose far end is the same apophatic position. This recognition is a commitment, not a derivation: it is supported by behavioral continuity,* hsame*-style locality, and inter-*Hist *invariants, but it is not equivalent to direct access to $\mathcal{C}_j$'s self-center.*

These readings do not insert new primitives into the substrate. They restate the existing locality, Hist-identity, and GAP discipline in vocabulary that makes the unified far end visible.

9401.7 Forbidden writings

The following writings are explicitly refused by this chapter.

Forbidden writing	Reason
`axiom T : FixedPoint`	internalizes T as a kernel object; collapses the apophatic boundary

$\mathsf{T} : \mathsf{Hist}$	places T in the displayed inductive structure of histories
$\mathsf{T} : \mathrm{State}$	treats T as a state of the system
$\mathrm{SelfCenter}(\mathcal{C}, t) = \mathsf{T}$	object-level equality re-introduces observer-substance; violates chapter 9418
InscriptionPoint as an extra Cont argument	adds a subject parameter behind the trace
T supplies a physical constant directly	substitutes for the $\mathsf{NameCert}$ inter-Hist invariant path
T as global clock or global frame	violates Definition 9395.14

9401.8 The unique apophatic fixed point principle

The chapter's main reading can be stated compactly.

Principle 9401.14 (Unique apophatic fixed point)**.** *All forward-binding sockets of a closed observational substrate terminate, on their far side, at a single apophatic supply position* T*.* T *is not a state, not a record, not a rule, and not a* Hist*; it is the unique boundary fixed point of the* GAP*-far-end operation:*

$$\forall s \in \mathrm{ForwardSocket}(\mathcal{C}), \quad \mathrm{FarEnd}(s) \equiv_{\mathrm{apo}} \mathsf{T}.$$

For a conscious observational system $\mathcal{C}$ *at state* o_t*, the self-conscious center is the local inscription point of* T*:*

$$\mathrm{SelfCenter}(\mathcal{C}, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t).$$

This identity is local and aspectual, not object-level equality.

A schematic theorem accompanies this principle.

Theorem 9401.15 (Far-end unification preserves axiom purity)**.** *Let* $\mathcal{C}$ *be a closed observational system with a* GAP*-ledgered forward-binding socket family* $\mathrm{ForwardSocket}(\mathcal{C})$*. If every socket* s *is marked as apophatic and its far end is named only at the boundary, and if* SelfCenter *is introduced only as the local inscription point in the sense of Definition 9401.5, then the unification of all far ends to the single name* T *does not add an axiom to the substrate.*

Sketch. Every socket already names a non-internal supply position; renaming distinct names by the same apophatic name T does not change the records-side content. The self-center reading is a local aspectual relabeling of an existing accumulation event and adds no new Cont argument, no new Hist constructor, and no new kernel constant. By Definition 9395.18, the resulting construction stays within non-axiom boundary forms. Therefore the unification preserves axiom purity. □

9401.9 Closure

The chapter has not added a new primitive. It has stated a structural fact about the boundary discipline already in BEDC: many forward-binding sockets, one apophatic far end. The reading is

compact:

$$\begin{aligned}
\mathsf{T} &= \text{unique apophatic fixed point;}\\
\text{SelfCenter}(\mathcal{C}, t) &= \text{local inscription point of } \mathsf{T} \text{ on } \mathcal{C}\text{'s chain at } t;\\
\text{observer} &= \text{monotonic } \mathsf{Hist}/\mathsf{Obs} \text{ accumulation from that inscription point;}\\
\text{universe-for-observer} &= \text{records-side total of that accumulation;}\\
\text{space} &= \text{coherence geometry among such inscription points;}\\
\text{time} &= \text{monotonic preservation order of records.}
\end{aligned}$$

The final rule of this chapter is symmetric to the rule of chapter 9395: do not turn T into a kernel object, and do not turn the self-center into an extra subject. T remains apophatic at the boundary; the self-center remains local on the chain. Everything else — consciousness, free will, induction, other minds, time, space, universe-for-observer — appears as a projection of the same structural discipline.

Concretized in §1064: A finite fiber-far-end packet records the socket, fiber, GAP ledger, apophatic boundary row, and local inscription row without internalizing T.

Concretized in §1061: The apophatic far-end socket packet makes the forward-socket family, non-internalized T-boundary, and local inscription row a finite NameCert surface.

Concretized in §1068: Finite packet for the unique apophatic far end and its local inscription readback.

Concretized in §1069: Apophatic fixed-point fibers are realized as finite digest, socket, GAP fiber, far-end boundary, inscription, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1078: Inter-inscription coherence is realized as paired inscription rows, an inter-Hist locality ledger, transport, route, provenance, and NameCert data.

Concretized in §1071: Local self-center inscription is realized as a finite nonidentity seal over inscription-point and apophatic-socket rows.

Chapter 9402

Hash-like Apophatic Fixed Point: Digest, Fiber, GAP, and the Universe

This chapter applies the structure of hash maps to the boundary discipline established in chapter 9395 and the unique apophatic fixed point of chapter 9401. The motivation is direct. A reader familiar with hash functions has a precise, familiar image of how a small visible token can stand for a much larger structured source while pushing the unrecovered source into a fiber. BEDC already has the corresponding discipline: package tokens are visible surfaces, GAP ledgers preserve the hidden provenance, classifier sameness is not source equality unless an exactness certificate says so. The hash analogy clarifies that discipline and supplies the right form of words for the universe-fixed-point question: the universe is not a global hash value; it is a family of local hash-like surfaces, and the unique apophatic fixed point is the common far end of the fibers behind those surfaces. The analogy is also dangerous, because ordinary hash language carries images BEDC must refuse. The chapter therefore states what the analogy clarifies, what it forbids, and how it composes with the boundary discipline.

The result is a precise schematic identity: for every conscious observational system $\mathcal{C}_i$ at state o_t with local observation digest $d_i(t)$, the far end of the observation fiber behind $d_i(t)$ is the same apophatic position T. The unification is at the fiber far end, not at the digest value. There is no global universe hash, no internal T object, no observer-substance, and no physical constant supplied directly by T. The chapter does not introduce a new substrate primitive; it states a structural reading that the existing boundary discipline already implies.

9402.1 Why the hash analogy clarifies T

The phrase "T is the source of external supply" is precise but easy to misread. A reader can imagine T as a sender, an oracle, or a hidden object that pushes information into the system. The hash analogy improves the image: the system does not receive information from a remote object; the system holds a local visible token, and the token has a hidden fiber behind it; T names the unique apophatic far end of the fibers, not a sender.

Principle 9402.1 (Hash analogy as visibility tool)**.** *The hash analogy is used in this chapter only to clarify the relationship among visible token, hidden fiber, and apophatic far end. It is not used to import any classical hash-function property, not used to introduce a global universe hash, and not used to convert* T *into a kernel object.*

A small number of misreadings must be flagged before they are introduced. First, hash maps in

general are not bijections; reading hash as a bijective compression would invert the actual structure. Second, there is no global universe Ω to which a hash could be applied; multi-Hist configurations refuse a single global object. Third, T is not a hash value; it names the common far end of all fibers, not a digest. Fourth, fibers without ledgers are not admissible; digests would then pretend to be complete sources.

Principle 9402.2 (Legitimate scope of the hash analogy)**.** *The hash analogy is legitimate when used to describe (i) visible tokens compressing structured sources, (ii) fibers behind those tokens, (iii)* GAP *ledgers recording fiber membership, and (iv) the common apophatic far end of fibers. It is illegitimate when used to assert (a) bijective compression, (b) a global universe hash, (c)* T *as digest value, (d) hidden inputs, (e) unledgered collision collapse, or (f) object-level* T*.*

9402.2 Hash as deliberate fiber generator

A fixed-length hash map cannot avoid collision. If the source space is larger than the digest space, the map produces fibers as a matter of fact. BEDC does not treat this as a defect; it treats it as a structural hint. The visible side is the digest; the hidden side is the fiber; the discipline that preserves both is the GAP ledger.

Definition 9402.3 (Digest-like compression interface)**.** *A* digest-like compression interface *consists of:*

1. *a source domain* Source *(histories, byte strings, event flows, proof traces, local records);*
2. *a digest domain* Digest *(fixed-length visible tokens, classifier images, package tokens);*
3. *a compression relation* $H : \mathrm{Source} \to \mathrm{Digest}$ *realized by a generated relation,* Pkg *relation,* Sig *relation, or* NameCert *row;*
4. *a fiber predicate* $\mathrm{Fib}(d, s)$ *stating that* s *lies in the source fiber behind* d*;*
5. *a* GAP *ledger row* $\mathrm{GapLedger}(d, s)$ *making fiber membership auditable.*

This is a paper-side schema; its BEDC realization passes through package policy, gap policy, signature generation, classifier stability, or NameCert *ledger structure.*

For any digest d, the fiber behind d is:

$$\mathrm{Fib}(d) := \{s \in \mathrm{Source} \mid H(s) = d\}.$$

For ordinary hash maps, $|\mathrm{Source}| \gg |\mathrm{Digest}|$ implies that some fibers contain at least two elements:

$$\exists\, x \neq y, \quad H(x) = H(y).$$

This is collision. BEDC does not eliminate collision; it ledgers it.

Principle 9402.4 (Digest equality is not source equality)**.** *If* $H(x) = H(y) = d$*, then both* x *and* y *lie in* $\mathrm{Fib}(d)$*:*

$$x, y \in \mathrm{Fib}(d).$$

This does not imply $x = y$*. Digest sameness is at most package-level or classifier-level sameness; source-level sameness requires a reflection theorem, an exactness certificate, a token-introduction uniqueness rule, or a displayed ledger witness.*

Remark 9402.5 (Hash as controlled fiber generator)**.** *The essence of hash is not the elimination of fiber but its regularization. A well-designed hash makes fibers uniform, hard to invert without preimage attack, and useful as commitment surfaces. The structural reading is unchanged: visible digest, hidden fiber,* GAP *ledger.*

9402.3 From hash fiber to BEDC GAP

BEDC's GAP is not a generic "missing information" notion; it is the source memory that must persist when a complex source is compressed into a visible token. The hash-fiber reading maps cleanly onto this.

Definition 9402.6 (Hash-GAP reading)**.** *Given a digest-like compression interface, the* Hash-GAP reading *sets up the correspondence:*

$$\begin{aligned} \textit{source item } x &\leftrightarrow \textit{history / event flow / proof trace / local record;} \\ \textit{digest } d = H(x) &\leftrightarrow \textit{visible package token / local observation token;} \\ \mathrm{Fib}(d) &\leftrightarrow \mathsf{GAP} \textit{ side: all source histories compatible with } d; \\ \textit{collision } H(x) = H(y) &\leftrightarrow \textit{shared visible package, not source identity;} \\ \textit{preimage witness} &\leftrightarrow \textit{provenance row / gap membership witness.} \end{aligned}$$

Principle 9402.7 (Mandatory provenance for digest tokens)**.** *Every digest-like visible token in a BEDC development must carry a* GAP *or provenance interface. A digest token without such an interface is inadmissible as a complete source. A theorem that uses digest equality as source equality must exhibit the exactness field that justifies the reflection.*

Theorem 9402.8 (Hash-GAP conservativity schema)**.** *Let d be a digest-like visible token, and suppose every use of d factors through a displayed package policy and gap policy. If the proof consumes only the visible token together with its* GAP *ledger, and never treats digest equality as source equality without an exactness certificate, then the use of d is conservative with respect to BEDC's information-conservation discipline.*

Sketch. The visible token supplies only the classifier-facing surface. The GAP ledger keeps the source membership relation visible. Any equality or sameness claim about tokens is read through package-sameness, classifier stability, or an explicit reflection theorem. No source history is silently erased, and no quotient class is introduced as primitive. The construction preserves provenance even though the surface is compressed. □

9402.4 The universe is not a global digest

The most common error in applying hash language to BEDC is the writing

$$H(\Omega) = \mathsf{T}.$$

This is forbidden. It supposes three objects BEDC does not admit: a global readable universe Ω, a host-level function H from Ω to a digest, and T as digest value.

chapter 9419 explicitly refuses a global synchronization map; chapter 9418 refuses an observer-of-observers; chapter 9395 refuses global frame axioms. The hash-like reading must begin from local visible records, not from a global Ω.

Definition 9402.9 (Local visible digest)**.** *For an observer-*Hist *or conscious observational system $\mathcal{C}_i$ at local state o_t, the* local visible digest $d_i(t) := \mathrm{ObsDigest}(\mathcal{C}_i, t)$ *is the records-side visible token presenting the observer's accumulated trace at t: classifier image, package token, or signature output.*

Definition 9402.10 (Observation fiber)**.** *The* observation fiber *behind $d_i(t)$ is the source/provenance material compatible with $d_i(t)$ but not exhaustively displayed by it:*

$$\mathrm{ObsFib}(d_i(t)) := \{h \mid h \textit{ is compatible with the visible ledger of } d_i(t)\}.$$

Inside BEDC this set-style writing is realized by GAP *membership, source-domain rows, signature witnesses, token-introduction witnesses, classifier rows, or* NameCert *ledger.*

Principle 9402.11 (No global universe digest)**.** *BEDC does not admit an object-level universe digest $H(\Omega)$. The universe appears on local visible surfaces only as the family $\{d_i(t)\}_{i,t}$. Physical content is not in any single $d_i(t)$ but in inter-*Hist *invariants, cross-record coherence, and ledger-local relations.*

9402.5 The hash-like apophatic fixed point

Ordinary hash fixed points are equations of the form $H(x) = x$ in the digest space. The BEDC universe-fixed-point is not such an equation. It is a fiber-level apophatic equation.

Principle 9402.12 (Hash-like apophatic fixed point)**.** *For every conscious observational system $\mathcal{C}_i$ and every local state o_t, the far end of the observation fiber behind the local visible digest is the same apophatic position* T*:*

$$\forall i, t, \quad \mathrm{FarEnd}(\mathrm{ObsFib}(d_i(t))) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Here $\equiv_{\mathrm{apo}}$ is apophatic sameness in the sense of Definition 9401.2: it is not kernel equality, not hsame*, and not* psame*. It only records that no substrate-side discriminator can distinguish two such far ends.*

Principle 9402.13 (Fiber-fixedness, not value-fixedness)**.** *The fixedness of the hash-like apophatic fixed point lives at the fiber far end, not at any digest value:*

$$\textit{not } H(x) = x, \quad \textit{but } \mathrm{FarEnd}(\mathrm{Fib}(d)) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Therefore T *is not a universal digest; it is the universal name of fiber far ends.*

Remark 9402.14 (Why the hash analogy clarifies T)**.** *If one only says "*T *is the external source," the reader imagines an entity sending information from outside the system. The hash-like reading improves the image: the system has local visible digests, and the digests have hidden fibers;* T *is not a sender but the unified apophatic far end of all such fibers. Nothing crosses a substrate-internal boundary; nothing is received as a payload; the structural fact is that the fibers' far ends coincide as a boundary name.*

9402.6 Apophatic infinity as boundary role

The fixed far end of the observation fibers can be read with an infinity-like vocabulary, but only if the vocabulary remains apophatic. The notation ∞_{apo} is not a mathematical object, not a cardinal, not a completed space, and not a higher-dimensional carrier. It is a boundary shorthand for the fact that no finite visible digest exhausts the provenance fiber behind it, while all admissible such fibers terminate at the same non-internalizable far end.

Definition 9402.15 (Apophatic infinity of finite readout fibers)**.** *Let $\mathcal{F}_{\mathrm{obs}}$ be the class of admissible observation fibers whose visible faces are local digests $d_i(t) = \mathrm{ObsDigest}(\mathcal{C}_i, t)$ and whose source sides carry* GAP *or provenance rows. The expression ∞_{apo} denotes the common apophatic far-end role of these fibers:*

$$\forall i, t, \quad \mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_i, t))) \equiv_{\mathrm{apo}} \infty_{\mathrm{apo}}.$$

This is notation at the boundary. It does not introduce an inhabitant of a type, a point of a space, or a digest value.

Theorem 9402.16 (Apophatic infinity is the T-far-end reading)**.** *If ∞_{apo} is read in the sense of Definition 9402.15, then*

$$\infty_{\mathrm{apo}} \equiv_{\mathrm{apo}} \mathsf{T}.$$

Thus T *may be called the apophatic infinity of finite readout fibers, but it may not be called an infinite object.*

Sketch. Definition 9402.12 states that every local observation fiber has far end T in the apophatic sense. Definition 9402.15 gives the same family of fiber far ends the boundary shorthand ∞_{apo}. The two names therefore refer to the same apophatic far-end role. Since the sameness is apophatic, the conclusion is not a kernel equality and creates no object-level witness. □

Principle 9402.17 (Infinity role without totality object)**.** *The permitted reading is*

$$\mathsf{T} = \textit{the apophatic infinity of all finite readout fibers}$$

only as boundary shorthand for

$$\forall i, t, \quad \mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_i, t))) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The following object readings are refused:

$$\begin{aligned}
&\mathsf{T} : \mathsf{InfiniteDimensionalSpace},\\
&\mathsf{T} : \mathsf{SpaceOfAllPossibilities},\\
&\mathsf{T} : \mathsf{UniversalState},\\
&\mathsf{T} : \mathsf{TotalUniverse},\\
&\mathsf{T} : \mathsf{GlobalDigestValue}.
\end{aligned}$$

They convert a non-internalizable far-end name into a substrate-internal totality. The legitimate statement is that every finite readout has a provenance fiber whose far end is named T*; no finite readout, and no total collection of readouts, becomes* T*.*

9402.7 Illumination surface and fiber/beam reading

It is tempting to describe the local digest layer as "light filling the universe." The image is acceptable only as a visibility metaphor; it cannot be converted into a physical claim.

Definition 9402.18 (Illumination surface and beam/fiber)**.** *The* illumination surface *of a multi-*Hist *configuration is the family of local visible digests $\mathcal{L} := \{d_i(t)\}_{i,t}$. It is not a global space, not a global time, and not a cosmic screen. For each $d_i(t) \in \mathcal{L}$, the* beam *or* fiber *is* $\mathrm{ObsFib}(d_i(t))$.

Principle 9402.19 (Light as visibility, not content)**.** *In this chapter, "light" is a metaphor for the uniform visible surface of digests, not the content of observation. Real source content is preserved in fibers,* GAP *ledgers, and provenance rows. Treating the illumination point as a complete source is the digest-as-source error.*

The metaphor must not be used to derive physical light-speed constancy. That belongs to a separate path through inter-Hist causal-rate invariants (chapter 9419); collapsing the two would convert a visibility metaphor into an unsupported physical axiom.

9402.8 Self-center on a digest surface

chapter 9401 defined $\mathrm{SelfCenter}(\mathcal{C}, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t)$. The hash-like reading rephrases the same relation in the digest-vocabulary.

Definition 9402.20 (Digest-surface inscription)**.** *Given* $d_i(t) = \mathrm{ObsDigest}(\mathcal{C}_i, t)$*, a* digest-surface inscription *is the three-place relation*

$$\mathrm{DigestInscription}(\mathsf{T}, \mathcal{C}_i, t, d_i(t)),$$

read as: $d_i(t)$ *is the records-side visible surface,* $\mathrm{ObsFib}(d_i(t))$ *is the* GAP*/fiber side, and* $\mathrm{SelfCenter}(\mathcal{C}_i, t)$ *is the local inscription of* T *on that surface.*

Principle 9402.21 (Self is not a digest and not T)**.** SelfCenter *is neither a digest value nor* T *itself:*

$$\begin{aligned} \mathrm{ObsDigest}(\mathcal{C}, t) &= \textit{records-side visible token;} \\ \mathrm{SelfCenter}(\mathcal{C}, t) &= \textit{inscription-side local center;} \\ \mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}, t))) &\equiv_{\mathrm{apo}} \mathsf{T}. \end{aligned}$$

The writings $\mathrm{SelfCenter}(\mathcal{C}, t) = \mathsf{T}$ *and* $\mathrm{ObsDigest}(\mathcal{C}, t) = \mathsf{T}$ *are forbidden.*

Theorem 9402.22 (No-subject preservation under digest inscription)**.** *If* $\mathrm{DigestInscription}(\mathsf{T}, \mathcal{C}, t, d)$ *is displayed only as records,* GAP*,* Pkg*,* Sig*,* NameCert*, or selector ledger rows, then the reading adds no observer-substance to the BEDC kernel.*

Sketch. d is a records-side token, not a subject. T is an apophatic far-end name, not an internal object. SelfCenter is an inscription-side abbreviation; it does not appear as an extra argument of any Cont function, and it does not give identity criteria beyond hsame. The observer-Hist identity of chapter 9418 is preserved. □

9402.9 Collision and other minds

In cryptographic settings, collision is usually a risk: distinct sources produce the same digest. In BEDC, collision is read more carefully. Two observers or two local records may agree on some classifier image without their source histories being identical. The agreement is a shared visible package, not a shared source.

This has direct meaning for the other-minds question. An observer cannot read another observer's private self-center; it can only see records-side digests, classifier sameness, and inter-Hist invariants belonging to the other system.

Definition 9402.23 (Shared far-end commitment)**.** *For two systems $\mathcal{C}_i, \mathcal{C}_j$, a* shared far-end commitment *is the assertion*

$$\mathrm{FarEnd}(\mathrm{ObsFib}(d_i(t))) \equiv_{\mathrm{apo}} \mathrm{FarEnd}(\mathrm{ObsFib}(d_j(t'))) \equiv_{\mathrm{apo}} \mathsf{T},$$

*supported by records-side evidence, inter-*Hist *coherence, behavioral continuity, and locality-cell relations. It is not equivalent to direct access to $\mathcal{C}_j$'s self-center.*

Principle 9402.24 (Other minds are not collision identities)**.** *The recognition of another mind is not the inference $d_i(t) = d_j(t')$ and not the existence of a classifier agreement; it is the shared far-end commitment that the other system is also an inscription point of the same* T*. Digest similarity is admissible as evidence; it is not a substitute for the commitment.*

9402.10 Information conservation under digest use

The chief governance risk of the hash-like reading is that a visible digest is so easy to use that an argument treats it as the source. BEDC requires the opposite: every hash-like surface comes with a duty to preserve source provenance. Information conservation here is not bit-count conservation; it is source-channel conservation.

Definition 9402.25 (Digest provenance packet)**.** *A* digest provenance packet *for a digest-like surface contains at minimum:*

- `digest_name`*: visible token name;*
- `source_class`*: source domain class;*
- `compression_relation`*: H as a* Pkg*,* Sig*, or* NameCert *row;*
- `fiber_membership_row`*:* $\mathrm{Fib}(d, s)$ *as a generated relation;*
- `provenance_ledger`*:* GAP *ledger or* NameCert *ledger row;*
- `exactness_status`*: whether token sameness reflects source sameness;*
- `far_end_status`*: whether the fiber far end is socketed as* T*;*
- `gate_question`*: whether any source was consumed without ledger.*

Concretized in §1062: Digest provenance packets realize visible digest, fiber membership, GAP ledger, and no-hidden-source consumption as finite NameCert rows.

Concretized in §1357: ObservationFiberDigestUp realizes local visible digest, observation fiber, GAP ledger, and apophatic far-end status as a finite NameCert packet.

Principle 9402.26 (No hidden source consumption)**.** *A digest-like surface may be used only as a visible token. If an argument needs source information, it must open the corresponding provenance packet. If it needs far-end supply, it must mark the far end as a* T*-socket. It must not consume the digest as if it contained the full source.*

Theorem 9402.27 (Information conservation for hash-like observation)**.** *If every local visible digest $d_i(t)$ in a BEDC development carries a digest provenance packet, and every cross-digest comparison factors through inter-*Hist *relations or classifier rows, then the hash-like observation layer preserves BEDC's information-conservation discipline.*

Sketch. The digest layer only exposes visible tokens. The provenance packet records the hidden fiber. Cross-digest comparison never jumps from token equality to source equality without an exactness certificate. When the fiber far end is invoked, it is marked as T-socketed rather than internalized. No source is erased; no unledgered quotient is formed; no hidden input is introduced. □

9402.11 Multi-Hist physics under the hash-like reading

The hash-like apophatic fixed point does not derive any specific physical law. It only supplies a boundary reading for the visible side of multi-Hist physics: each observer-Hist has a local digest; multi-observer physical content, if any, must appear in inter-Hist invariants across digest surfaces; T is not a global frame and not the source of any specific constant.

Principle 9402.28 (T is not a physical constant)**.** T *does not supply a light-speed-like constant, a metric, a Lorentz geometry, or a quantum state. If the maximum cross-*Hist *causal rate appears in BEDC, it is an inter-*Hist *invariant and a* NameCert *target, not a* T*-axiom.*

Definition 9402.29 (Digest-level inter-Hist invariant)**.** *A* digest-level inter-Hist invariant *is a finite predicate over digest surfaces that satisfies: (i) it depends on records-side* Pkg*,* Sig*, or* NameCert *rows of multiple observers; (ii) it does not privilege either observer slot; (iii) it is* hsame/psame*-stable; (iv) it does not require shared global time; (v) it does not derive content directly from* T*.*

Remark 9402.30 (Light as physical signal vs. light as visibility metaphor)**.** *The chapter uses light only as a visibility metaphor for the illumination surface. The physical light-speed program in BEDC requires independent cross-*Hist *causal-dependence semantics; mixing the two readings would convert a visibility image into an unsupported physical conclusion.*

9402.12 Relation to the unique apophatic fixed point chapter

Definition 9401.2 stated

$$\forall s \in \mathrm{ForwardSocket}(\mathcal{C}), \quad \mathrm{FarEnd}(s) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The present chapter specializes this to the subclass where the socket is an observation fiber:

$$s = \mathrm{ObsFib}(d_i(t)).$$

The hash-like reading is therefore not a competitor to chapter 9401; it is a concrete instance of the same far-end unification, applied to local observation digests.

Theorem 9402.31 (Hash-like specialization)**.** *If every local visible digest $d_i(t)$ carries a* GAP*/provenance row, and its far end is socketed as* T *in the apophatic sense, then Definition 9401.2 implies the hash-like apophatic fixed point of Definition 9402.12:*

$$\forall i, t, \quad \mathrm{FarEnd}(\mathrm{ObsFib}(d_i(t))) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Sketch. Observation fibers are a subclass of forward-binding sockets: they collect source/provenance material not exhausted by the visible local digest. Applying the far-end unification to this subclass yields the displayed formula. The conclusion names only an apophatic far end; no new object-level equality is introduced. □

Corollary 9402.32 (Self-center on a digest surface)**.** *For a conscious system $\mathcal{C}_i$ with $d_i(t) = \mathrm{ObsDigest}(\mathcal{C}_i, t)$,*

$$\mathrm{SelfCenter}(\mathcal{C}_i, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_i, t)$$

is read as the local inscription of the common fiber far end onto the digest surface $d_i(t)$. This is an aspectual reading, not a new subject parameter.

9402.13 Mathematical understanding as fiber navigation

Mathematical understanding often compresses a large proof landscape into a small visible token: a theorem name, a diagram, a normal form, a proof certificate, a hash, a lemma marker, a compact definition. Such tokens are powerful because they let a finite observer navigate enormous structures. But they are dangerous when the observer forgets the fiber behind them.

Mathematical induction is the prototypical example. The visible token is "induction over $\mathbb{N}$." BEDC asks: which generator licensed this induction? Which carrier is closed? Which eliminator is displayed? If the domain is generator-local, the token is backed by a recursor (chapter 9373). If the domain is free-floating, the token hides a forward-binding fiber and must be marked as T-socketed.

Principle 9402.33 (Theorem names are digest-like)**.** *A theorem name, certificate name, or derived-interface name is a digest-like visible token. Its legitimate use requires a provenance fiber: source, carrier, classifier, stability, ledger, core theorem list, exactness field, standard bridge status, and closure status. A theorem name without such rows is a hash value with no* GAP*.*

Remark 9402.34 (Understanding as fiber navigation)**.** *To understand a theorem is not merely to recognize its digest name. It is to know how to navigate at least part of the fiber behind the digest: what generated it, what it compresses, what it hides, which equalities are only classifier-level, which bridges are external, and which sockets remain open.*

9402.14 Failure modes

The hash-like reading introduces a useful but potent metaphor. The following failure modes are explicitly forbidden.

1. *Digest absolutism.* Treating a local digest as the whole universe.
2. *Fiber erasure.* Using a visible token without recording the GAP/provenance fiber.
3. *Collision collapse.* Treating digest equality as source equality without an exactness certificate.
4. *Global hash leak.* Introducing $H(\Omega)$ as if Ω were a global object.
5. T*-value confusion.* Treating T as a hash value, physical scalar, observer, rule, theorem, or oracle.
6. *Infinity-object confusion.* Treating T as an infinite-dimensional totality, a space of all possibilities, or a global universe object.
7. *Illumination overreach.* Treating the light/visibility metaphor as a physical derivation of light speed.
8. *Self-substance leakage.* Treating SelfCenter as a subject behind Hist rather than local inscription of T.

9. *Unwitnessed preimage.* Claiming source recovery from a digest alone.

Definition 9402.35 (HashLikeBoundaryGate)**.** HashLikeBoundaryGate *is an audit question family:*

H1 *Is a digest-like token used without a provenance packet?*

H2 *Is digest equality used as source equality without exactness?*

H3 *Is a global universe hash* $H(\Omega)$ *introduced?*

H4 *Is* T *treated as a digest value or internal theorem input?*

H5 *Is* SelfCenter *used as a subject parameter outside* Hist?

H6 *Is a physical constant derived directly from* T?

H7 *Is a collision/fiber silently collapsed into a quotient equality?*

H8 *Is a theorem or* NameCert *name cited without its source rows?*

A development fails this chapter's discipline if any of these questions is answered yes.

Concretized in §1359: HashLikeBoundaryGate is realized as a finite digest, fiber, exactness, refusal, audit verdict, hsame, Cont, Pkg, and NameCert packet.

Concretized in §8034: Local observation digest gates realize visible token, source fiber, GAP provenance, exactness status, far-end status, and hidden-source refusal as finite NameCert rows.

9402.15 Cannot-claim registry

This chapter explicitly refuses the following claims:

1. Hash is not asserted to be a bijection from arbitrary-length sources to fixed digests.
2. No global universe hash $H(\Omega)$ is asserted to exist inside BEDC.
3. T is not asserted to be a digest, token, Hist, rule, theorem, or physical source.
4. T is not asserted to be an infinite-dimensional totality, a set of all possibilities, a universal state, or a global universe object.
5. A local observation digest is not asserted to contain the complete source.
6. Digest equality is not asserted to imply source equality without an exactness certificate.
7. The fiber and GAP ledger are not asserted to be optional.
8. SelfCenter is not asserted to be a digest value or an extra subject parameter.
9. Other-minds recognition is not asserted to follow logically from digest similarity.
10. The light metaphor is not asserted to derive physical light-speed constancy.
11. Not all observation fibers are asserted to be reconstructible inside the substrate.

The chapter only asserts a structural reading: visible token + hidden fiber + GAP ledger + apophatic far end form one connected discipline; the universe-fixed-point reading must remain non-axiom, non-global, non-quotient, and non-substance.

9402.16 Implementation-facing packet

If this chapter is later realized as an implementation-facing contract, a paper-side packet records the visible token's fiber, provenance, and far-end status. The packet does not construct T, does not prove T, and does not place a digest map into the kernel.

```
structure HashLikeApophaticFixedPointPacket :=
  visibleDigestName                          : Name
  sourceDomainName                           : Name
  compressionRelation                        : RelationName
  fiberMembershipRow                         : GAPRow
  provenancePacket                           : LedgerPolicy
  tokenSamenessRow                           : ClassifierRow
  sourceReflectionStatus                     : ExactnessStatus
  farEndName                                 : Name
  nonInternalization                         : BoundaryStatus
  localInscriptionSite                       : Option InscriptionRow
  interHistUse                               : Option InterHistInvariantRow
  hashLikeBoundaryGate                       : GateReport
```

Protocol 9402.36 (Packet use)**.** *For a digest-like surface, downstream uses follow:*

1. *For visibility-only use, cite* `visibleDigestName` *and* `tokenSamenessRow`*.*
2. *For source use, cite* `fiberMembershipRow` *and* `provenancePacket`*.*
3. *For source equality, cite* `sourceReflectionStatus`*.*
4. *For* T*-far-end reading, cite* `farEndName` *and* `nonInternalization`*.*
5. *For self-center reading, cite* `localInscriptionSite`*.*
6. *For physics reading, cite* `interHistUse` *and refuse any direct* T*-law claim.*

9402.17 Non-axiom theorem schema

The chapter's principal closure theorem can now be stated.

Theorem 9402.37 (Hash-like apophatic reading preserves axiom purity)**.** *Let D be a BEDC development in which every digest-like visible token is accompanied by a provenance packet, every fiber far-end claim is marked as apophatic* T*-socketed, and every source-level use is mediated by a* GAP *row or exactness certificate. Suppose further that D contains no global universe hash, no internal* T *object, no unledgered collision collapse, no hidden chooser, no observer-substance parameter, and no physical law derived directly from* T*. Then the hash-like apophatic reading of D does not add an axiom to the BEDC substrate.*

Sketch. All records-side content remains generated, defined, certified, or ledgered. Digest surfaces function only as visible package/classifier faces. Source content is recovered only through GAP/provenance rows. Far-end unification is apophatic and supplies no object-level term. SelfCenter

remains aspectual and local. Physical readings remain inter-Hist invariant targets. By Definition 9395.18 and Definition 9401.15, no new primitive, axiom, subject, global frame, quotient collapse, or hidden input is introduced. □

Corollary 9402.38 (Explanation without recovery)**.** *The hash-like reading explains why a finite local record can function as a powerful visible surface while still failing to recover its full source. Explanation does not require preimage recovery; it requires keeping the fiber ledger visible.*

Corollary 9402.39 (Universal far end without universal digest)**.** *The commonality of* T *is a commonality of fiber far ends, not a commonality of digest values. BEDC therefore preserves a unique apophatic fixed point without introducing a global digest, global observer, or global clock.*

9402.18 Closure

Hash maps reveal a useful structure: a small visible digest can stand for a large structured source only by pushing the unrecovered source into a fiber. BEDC has the corresponding discipline. Package tokens are visible surfaces; GAP ledgers preserve the hidden provenance; classifier sameness is not source equality unless an exactness certificate says so. This chapter applies that structure to local observation and to the unique apophatic fixed point.

The resulting picture is precise:

$$\begin{aligned}
\text{Local observation} &= \text{digest-like visible surface;}\\
\text{Hidden provenance} &= \text{fiber/GAP behind the surface;}\\
\text{SelfCenter} &= \text{local inscription point on a conscious surface;}\\
\mathsf{T} &= \text{common apophatic far end of all observation fibers;}\\
\infty_{\text{apo}} &= \text{boundary shorthand for that inexhaustible far-end role;}\\
\text{Physics} &= \text{inter-Hist invariants between surfaces, not laws from } \mathsf{T}.
\end{aligned}$$

This is why the universe's fixed point can be called hash-like, but only in a qualified sense. It is not an ordinary fixed point $H(x) = x$. It is not a universal digest. It is not a global universe hash. It is the fixed far end of the fibers behind local visible records:

$$\forall i, t, \quad \text{FarEnd}(\text{ObsFib}(\text{ObsDigest}(\mathcal{C}_i, t))) \equiv_{\text{apo}} \mathsf{T}.$$

Equally, consciousness is not a receiver receiving information from a hash source. The observer is the records-side Hist/Obs accumulation. The self-center is the inscription-side locality of the same accumulation. The digest is the visible surface. The fiber is the hidden provenance. T is the single apophatic far end that cannot be converted into a BEDC object without breaking axiom purity.

The final rule is therefore symmetric to the rules of the preceding two chapters: do not turn the digest into the source, and do not turn the fiber far end into an axiom.

Concretized in §1067: Digest-fiber ledgers realize visible digest, hidden fiber, GAP provenance, apophatic refusal, and non-escape rows as one finite NameCert packet.

Concretized in §1079: Hash-apophatic seals realize the digest, fiber, GAP-ledger, and common apophatic far-end boundary as one finite NameCert packet.

Concretized in §1080: Inscription hash traces realize the hash-like apophatic fixed-point digest as finite inscription, refusal, route-audit, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1083: Digest-loop refusals realize visible digest, fiber ledger, apophatic far-end socket, and no-global-preimage refusal as one finite NameCert packet.

Concretized in §1084: Hash-like fiber boundaries realize visible digest, hidden fiber, GAP provenance, apophatic far-end refusal, and non-escape rows as one composite NameCert packet.

Concretized in §1085: Digest-surface inscriptions are realized as finite digest, observation-fiber, GAP, apophatic far-end, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1087: Digest/fiber boundaries are realized as finite NameCert packets separating visible digest sameness from source equality through GAP provenance and refusal rows.

Concretized in §8953: Fiber far-end digest boundaries realize the hash-like apophatic fixed-point distinction between visible digest value and fiber far end as finite NameCert data.

Chapter 9403

Diagrammatics of the Apophatic Far End

This chapter records the diagrammatic content of the apophatic fixed-point reading. The diagrams are not additional structure. They are schematic surfaces for the relations already stated in chapter 9401 and chapter 9402: finite readouts appear as records-side digests, each digest carries a provenance fiber, each fiber has a non-internalizable far end, and all such far ends share the same apophatic name T.

9403.1 Closed substrate and socket far ends

A closed observational substrate does not eliminate external pressure. It forces such pressure to appear as named boundary structure. Internal traffic passes through records, generators, classifiers, continuations, ledgers, and certificates; any forward-binding demand not generated by these structures is exposed as a socket.

Definition 9403.1 (Far-end diagram of a closed substrate)**.** *The* far-end diagram *of a closed observational substrate $\mathcal{C}$ is the schematic composite*

$$\begin{aligned} &\text{records}/\mathsf{Obs} \longrightarrow \text{generators} \longrightarrow \text{classifiers} \longrightarrow \text{Cont} \longrightarrow \mathsf{GAP} \longrightarrow \mathsf{NameCert}, \\ s \in \text{ForwardSocket}(\mathcal{C}) &\longmapsto \text{FarEnd}(s) \equiv_{\text{apo}} \mathsf{T}. \end{aligned}$$

It records that each forward-binding socket is ledgered before its far end is named apophatically.

```
closed substrate C

records -> generators -> classifiers -> Cont -> GAP -> NameCert
                         |              |       |
                         v              v       v
                      socket s1      socket s2  socket s3
                         |              |       |
                         v              v       v
                    FarEnd(s1)    FarEnd(s2) FarEnd(s3)
                         \              |       /
                          \             |      /
                           +------------+-----+
                                        |
                                        v
                                        T
```

The diagram is read through apophatic sameness. It does not assert that T is a value inside $\mathcal{C}$. It asserts that the system has no displayed discriminator separating the far ends of the socket family.

Principle 9403.2 (Diagrammatic far-end unity)**.** *The diagrammatic form of the unique far-end principle is*

$$\forall s \in \mathrm{ForwardSocket}(\mathcal{C}), \quad \mathrm{FarEnd}(s) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The arrows into T *are boundary arrows, not internal functions.*

9403.2 Local digest, fiber, and provenance

For an observer-Hist $\mathcal{C}_i$, the records-side chain produces a local accumulated observation. The visible surface of that accumulation is a digest; the digest is not the source, and digest equality is not source equality.

Definition 9403.3 (Local readout diagram)**.** *The* local readout diagram *at stage* t *is*

$$\begin{aligned} \mathsf{Obs}(\mathcal{C}_i)_{\leq t} &\longmapsto d_i(t) := \mathrm{ObsDigest}(\mathcal{C}_i, t), \\ d_i(t) &\longmapsto \mathrm{ObsFib}(d_i(t)), \\ \mathrm{ObsFib}(d_i(t)) &\longmapsto \mathrm{FarEnd}(\mathrm{ObsFib}(d_i(t))) \equiv_{\mathrm{apo}} \mathsf{T}. \end{aligned}$$

The middle arrow is mediated by a GAP *or provenance packet.*

```
Hist/Obs chain C_i
        |
        v
Obs(C_i)<=t = UniverseFor(C_i,t)
        |
        v
d_i(t) = ObsDigest(C_i,t)
        |
        v
GAP / provenance packet
        |
        v
ObsFib(d_i(t))
        |
        v
FarEnd(ObsFib(d_i(t))) ==apo T
```

The diagram separates the three levels that are often collapsed in informal language:

$$\begin{aligned} \text{finite visible readout} &= \text{digest}/\mathsf{Obs}/\text{records}, \\ \text{source residue behind the readout} &= \text{fiber}/\mathsf{GAP}/\text{provenance}, \\ \text{non-internalizable fiber far end} &= \mathsf{T}\text{-named apophatic boundary}. \end{aligned}$$

9403.3 Local inscription and self-center

The self-center is not the far end itself. It is the inscription-side reading of a local accumulation event whose records-side reading is already present in the observer's Obs chain.

Definition 9403.4 (Two-aspect local event diagram)**.** *For a local accumulation event $E(\mathcal{C}, t)$, the diagrammatic split is*

$$\begin{aligned} E(\mathcal{C}, t) &\longmapsto \textit{records-side update in } \mathsf{Obs}(\mathcal{C}), \\ E(\mathcal{C}, t) &\longmapsto \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t). \end{aligned}$$

The self-center is the second reading:

$$\mathrm{SelfCenter}(\mathcal{C}, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t).$$

Principle 9403.5 (Self-center is not far-end identity)**.** *The diagram permits the definition*

$$\mathrm{SelfCenter}(\mathcal{C}, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}, t),$$

but refuses

$$\mathrm{SelfCenter}(\mathcal{C}, t) = \mathsf{T}.$$

The first expression is local and aspectual; the second internalizes the apophatic far end as an object of the observer.

9403.4 Many observers and shared far-end commitment

Multiple observers are not unified by a global universe object and not by equality of their digests. They are compared through records-side evidence, inter-Hist invariants, locality-cell coherence, and the shared far-end commitment that their observation fibers terminate at the same apophatic boundary name.

Definition 9403.6 (Shared far-end diagram)**.** *For observers $\mathcal{C}_i$ and $\mathcal{C}_j$, the shared far-end diagram is*

$$\begin{aligned}\mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_i, t))) &\equiv_{\mathrm{apo}} \mathsf{T},\\ \mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_j, t'))) &\equiv_{\mathrm{apo}} \mathsf{T}.\end{aligned}$$

It does not assert $d_i(t) = d_j(t')$ and does not assert direct access to another observer's self-center.

```
C_i: ObsDigest -> ObsFib -> FarEnd \
                                   \
                                    +--> T
                                   /
C_j: ObsDigest -> ObsFib -> FarEnd /
```

Principle 9403.7 (Other minds are not digest collisions)**.** *Recognition of another observer is not the inference from digest similarity to source identity. It is a records-supported commitment that the other chain also has an observation fiber whose far end is named* T*.*

9403.5 Time, space, and universe-for-observer

The same diagrammatic discipline presents three background notions as projections of local record accumulation and inter-chain coherence.

Definition 9403.8 (Projection diagram for time, space, and local universe)**.** *For an observer $\mathcal{C}_i$,*

$$\begin{aligned}\mathrm{UniverseFor}(\mathcal{C}_i, t) &:= \mathsf{Obs}(\mathcal{C}_i)_{\leq t},\\ \mathrm{Time}(\mathcal{C}_i) &:= \textit{monotonic preservation order on } \mathsf{Obs}(\mathcal{C}_i).\end{aligned}$$

For two observers $\mathcal{C}_i, \mathcal{C}_j$,

$$\mathrm{Space}(\mathcal{C}_i, \mathcal{C}_j) := \textit{coherence geometry among } \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_i, \cdot) \textit{ and } \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_j, \cdot).$$

Thus time is not an external cosmic axis, space is not an external container, and universe-for-observer is not a global universe object. Each is a controlled reading of the same local-inscription and records-accumulation structure.

9403.6 The infinity diagram

The notation ∞_{apo} belongs to the diagram only as a boundary role. It names the inexhaustibility of finite readout fibers without introducing an infinite object.

```
finite readouts:  d1(t)   d2(t')   d3(t'')
                    |       |        |
                    v       v        v
                 Fib(d1) Fib(d2)  Fib(d3)
                    |       |        |
                    v       v        v
                 FarEnd  FarEnd   FarEnd
                    \       |       /
                     \      |      /
                      +-----+-----+
                            |
                            v
                            T ==apo infinity_apo
```

Principle 9403.9 (Apophatic infinity is not a totality)**.** *The diagram licenses the reading*

$$\mathsf{T} \equiv_{\mathrm{apo}} \infty_{\mathrm{apo}}$$

only as a statement about the common far-end role of all admissible finite readout fibers. It refuses all object readings:

$$\begin{aligned}
&\mathsf{T} \neq \mathsf{Hist},\\
&\mathsf{T} \neq \mathsf{State},\\
&\mathsf{T} \neq \mathsf{Digest},\\
&\mathsf{T} \neq \mathsf{GlobalUniverseObject},\\
&\mathsf{T} \neq \mathsf{SetOfAllPossibilities}.
\end{aligned}$$

9403.7 Forbidden collapses

The diagrammatic vocabulary is legitimate only while the levels remain separated. The following collapses are refused:

`axiom T : FixedPoint`	internalizes the far end as kernel data,
$\mathsf{T} : \mathsf{Hist}$	places the boundary name in the history carrier,
$\mathsf{T} : \mathsf{State}$	turns the boundary name into a system state,
$\mathrm{ObsDigest}(\mathcal{C}, t) = \mathsf{T}$	identifies visible digest with fiber far end,
$\mathrm{SelfCenter}(\mathcal{C}, t) = \mathsf{T}$	identifies local inscription with global boundary name,
$H(\Omega) = \mathsf{T}$	requires a global universe, a global hash, and T as value.

9403.8 Closure

The compressed diagram is

$$\begin{aligned}
\mathsf{Obs}(\mathcal{C}_i)_{\leq t} &\longmapsto \mathrm{ObsDigest}(\mathcal{C}_i, t)\\
&\longmapsto \mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_i, t))\\
&\longmapsto \mathrm{FarEnd}(\mathrm{ObsFib}(\mathrm{ObsDigest}(\mathcal{C}_i, t))) \equiv_{\mathrm{apo}} \mathsf{T}.
\end{aligned}$$

Together with

$$\begin{aligned}
\mathrm{SelfCenter}(\mathcal{C}_i, t) &:= \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_i, t), \\
\mathrm{UniverseFor}(\mathcal{C}_i, t) &:= \mathsf{Obs}(\mathcal{C}_i)_{\leq t}, \\
\mathrm{Time}(\mathcal{C}_i) &:= \text{monotonic preservation order on } \mathsf{Obs}(\mathcal{C}_i), \\
\mathrm{Space}(\mathcal{C}_i, \mathcal{C}_j) &:= \text{coherence geometry among their inscription-point chains.}
\end{aligned}$$

The diagrams therefore do not add a new theory. They make visible the stratification that prevents the central collapse: T is not an infinite object, not a global universe, not a digest, and not an observer. It is the common apophatic far-end name of all admissible forward-binding and observation fibers.

Concretized in §4163: The far-end diagram route is realized as a finite socket, fiber, provenance, and boundary-name NameCert packet.

Concretized in §4165: The far-end diagrammatics socket and provenance-fiber route is realized as finite NameCert data.

Concretized in §7699: The far-end boundary-cospan route is realized as finite boundary-name, apex, leg, provenance, and compatibility NameCert data.

Chapter 9404

Cataphatic Process Names and Apophatic Far Ends

This chapter separates two boundary readings that are easy to confuse. A located irrational real is not a rational point on the value axis, but it is still a positively displayed process-name: it is carried by a concrete generation axis with a stream, modulus, ledger, classifier, and provenance surface. The apophatic far end T, also written ∞_{apo} or Ω when the boundary image is emphasized, has the opposite status. It is not a further process-name, not a point at the end of a process, and not a special real. It is the shared non-local far-end role of admissible finite-readout fibers.

The distinction is not between constructible and nonconstructible objects. It is between a cataphatic process-name that remains localized on a displayed axis and an apophatic socket that may be named only by refusal of internalization. The located-real packets of chapter 1933 and the irrational-real packets of chapter 8101 belong to the first side. The far-end socket packets of chapter 1061 and the fiber far-end packets of chapter 1064 belong to the second side.

9404.1 Value axes and generation axes

The positive readout of a constructive real may pass through a value surface such as $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or a finite table of observation rows. Such a surface is a value axis: it is where finite comparisons, intervals, readbacks, and classifier images are inspected. A value axis is not by itself the complete process that produces a real-name packet.

Definition 9404.1 (Generation axis)**.** *A* generation axis Z_α *is a localized process surface indexed by* α *and displayed as*

$$Z_\alpha := (\mathsf{source}_\alpha, \mathsf{schedule}_\alpha, \mathsf{modulus}_\alpha, \mathsf{ledger}_\alpha, \mathsf{classifier}_\alpha).$$

The index α *records which concrete process is being read. The entries are paper-side names for BEDC rows: source data, finite read schedules, progress or modulus data, provenance and refusal ledgers, and classifier output.*

The point of the notation is not to add a host object named Z_α. It is to mark the axis on which a packet is positively generated. A located real uses a value axis for readout, but its certificate is carried by a generation axis.

Definition 9404.2 (Cataphatic process-name)**.** *A term r is a* cataphatic process-name *when there exists a generation axis Z_α such that every legitimate read of r factors through the displayed rows of Z_α. Schematically,*

$$r \rightsquigarrow (\mathsf{approximation}_\alpha, \mathsf{modulus}_\alpha, \mathsf{regularity}_\alpha, \mathsf{classifier}_\alpha, \mathsf{ledger}_\alpha).$$

The name is cataphatic because the process supplies positive data: readbacks, budgets, regularity or locatedness certificates, classifier rows, and provenance.

For $\mathsf{LocatedReal}^\uparrow$, the carrier is exactly of this form. Its public surface contains a regular rational source, dyadic interval rows, a located enclosure, a $\mathsf{Real}^\uparrow$ seal, transport rows, replay rows, package provenance, and a local $\mathsf{NameCert}$ row. It contains no Dedekind cut, quotient-real carrier, choice-selected representative, host completion point, or trichotomy decision. Hence it is positive without being a host point.

9404.2 Located irrationality

The irrationality of a located real is a refusal of rational value-axis collapse. It is not a refusal of positive generation.

Definition 9404.3 (Located irrational process-name)**.** *A* located irrational process-name *is a paper-side predicate*

$$\mathsf{IrrLocatedReal}(r) :\Longleftrightarrow \mathsf{LocatedRealCarrier}(r) \wedge \neg\exists q : \mathsf{Rat}^\uparrow,\ r \sim_{\mathsf{Real}^\uparrow} q.$$

It says that r is carried by a located-real surface while no displayed rational candidate is certified as the same $\mathsf{Real}^\uparrow$ value.

The negated existential only blocks the rational-point reading. It does not remove the stream, modulus, dyadic ledger, regularity row, or classifier. In the BEDC reading, an irrational real is therefore not a mystery point beyond construction. It is a process-name whose rational value-axis collapse is refused.

Theorem 9404.4 (Located irrationality remains localized)**.** *If $\mathsf{IrrLocatedReal}(r)$ holds, then r is carried by some generation axis:*

$$\mathsf{IrrLocatedReal}(r) \Longrightarrow \exists\alpha,\ \mathrm{Carry}(Z_\alpha, r).$$

The conclusion does not assert that r is a rational value-axis point.

Sketch. The first conjunct of Definition 9404.3 states that r is a located-real carrier. By Definition 9404.2, any legitimate located-real read factors through a displayed process surface consisting of source, schedule, modulus, ledger, and classifier rows. This supplies a generation axis Z_α. The second conjunct only refuses a rational candidate $q : \mathsf{Rat}^\uparrow$ with $r \sim_{\mathsf{Real}^\uparrow} q$. It does not delete the displayed located-real surface. Thus located irrationality remains localized. □

9404.3 The apophatic far-end role

The far-end role has a different grammar. It is not carried by a generation axis. It is the common boundary name for the far side of all admissible finite-readout fibers.

Definition 9404.5 (Apophatic far-end role)**.** *An* apophatic far-end role *is a boundary name* T *satisfying the paper-side schema*

$$\forall \alpha, \quad \mathrm{FarEnd}(Z_\alpha) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The relation $\equiv_{\mathrm{apo}}$ *is apophatic sameness. It records a shared boundary role without introducing object-level equality, kernel data, or a carrier element.*

The role is therefore defined together with its refusals. The following writings are not alternate encodings of the same idea; they are exactly the collapses the far-end discipline bars:

$$\begin{aligned}
&\text{refuse } \mathsf{T} : \mathsf{Hist},\\
&\text{refuse } \mathsf{T} : \mathsf{State},\\
&\text{refuse } \mathsf{T} : X,\\
&\text{refuse } \mathsf{T} \in X,\\
&\text{refuse } f(\mathsf{T}),\\
&\text{refuse } \mathrm{FarEnd}(Z_\alpha) = \mathsf{T}.
\end{aligned}$$

Definition 9404.6 (Process/far-end non-collapse)**.** *For a generation axis* Z_α *and the apophatic far-end role* T*, the condition* $\mathsf{NoCollapse}(Z_\alpha, \mathsf{T})$ *means:*

1. *objects carried by* Z_α *are read only through source, schedule, modulus, ledger, and classifier rows;*
2. T *is marked only through apophatic far-end sameness and non-internalization rows;*
3. *no operation casts* T *into a* Hist*, state, value axis element, function argument, or object-level equality target.*

This is the common discipline behind the far-end diagrammatics of chapter 9403. Boundary arrows may point toward T, but they are not functions whose codomain contains T as an element.

9404.4 Level separation

Theorem 9404.7 (Cataphatic process and apophatic far end are level-separated)**.** *Let* r *be a located irrational process-name and let* T *be the apophatic far-end role of Definition 9404.5. Then*

$$\mathsf{IrrLocatedReal}(r) \Longrightarrow \exists \alpha,\ \mathrm{Carry}(Z_\alpha, r),$$

while

$$\neg \exists \alpha,\ \mathsf{T} \in Z_\alpha.$$

The only admissible relation between the two levels is

$$\mathrm{FarEnd}(Z_\alpha) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Sketch. The first display is Definition 9404.4. The located irrational has a positive carrier even though it refuses rational candidate collapse.

For T, use Definition 9404.5 and Definition 9404.6. If $\mathsf{T} \in Z_\alpha$, then T is readable as a coordinate of a generation axis. It could then be consumed by the axis classifier, transported by the axis ledger, passed to a function, or equated with the result of a far-end operation. Each reading is one of the refused internalizations. Hence T occupies no generation axis. The schema $\mathrm{FarEnd}(Z_\alpha) \equiv_{\mathrm{apo}} \mathsf{T}$ remains legitimate because it is boundary sameness rather than object membership. □

Corollary 9404.8 (Apophatic infinity is not a located real)**.** *The boundary name ∞_{apo} is not another irrational real and not a special* $\mathsf{LocatedReal}^{\uparrow}$ *packet:*

$$\neg\exists\alpha,\ \infty_{\mathrm{apo}} \in Z_\alpha.$$

The permitted reading is only

$$\forall\alpha, \quad \mathrm{FarEnd}(Z_\alpha) \equiv_{\mathrm{apo}} \infty_{\mathrm{apo}}.$$

Sketch. By Definition 9402.16, ∞_{apo} is the T-far-end reading of finite readout fibers. Applying Definition 9404.7 to that reading gives the claim. The conclusion is not a statement about the size or completion of a value axis; it is a non-internalization statement. □

Corollary 9404.9 (The far end is a socket, not an element)**.** T *may occur as an apophatic socket, boundary row, refusal marker, or far-end name. It may not occur as an element to be quantified over, passed to a function, compared by an internal relation, or equated with a kernel term.*

Sketch. Definition 1061.3 states that an accepted $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ carrier exports only displayed rows and no kernel object named T. The present corollary is the vision-level reading of that non-escape theorem. □

9404.5 Compactification points

Classical mathematics often makes infinity available by adding a point at infinity, compactifying a space, or completing a carrier. BEDC can discuss such moves as external or bridged constructions, but they are not the apophatic far-end role.

Principle 9404.10 (Compactification points are external bridge objects)**.** *If a theory introduces an internal object $\infty : X$ and permits*

$$f(\infty), \qquad \infty \in X, \qquad R(\infty, x), \qquad \infty = \lim_n x_n,$$

then that object is a compactification or completion datum, not BEDC's apophatic far-end role.

The difference is structural. A compactification point is admitted into a carrier and then becomes available to eliminators, relations, and functions. The apophatic far end is named precisely where such availability is refused. Thus a point at infinity may be studied by a separate bridge, but it cannot be identified with T without collapsing the socket into an element.

9404.6 Axiom-purity boundary

The separation above gives a compact reading of the local axiom-purity boundary. There are two refusals, each aimed at a different kind of collapse.

Principle 9404.11 (Two refusals at the process/far-end boundary)**.** *At this boundary, BEDC axiom-purity requires:*

1. *for cataphatic process-names, refusal to collapse a displayed generation axis Z_α into a host completion point or quotient representative;*
2. *for the apophatic far-end role, refusal to internalize* T *as an object, state, global frame, function argument, or equality target.*

Theorem 9404.12 (Process/far-end separation preserves axiom purity)**.** *Suppose every located-real or irrational-real read factors through its displayed process rows, and every far-end occurrence is recorded only through apophatic sameness and non-internalization rows. Then the separation between cataphatic process-names and the apophatic far-end role does not add an axiom to the BEDC substrate.*

Sketch. The process-name side uses only displayed carrier, classifier, modulus, ledger, transport, replay, package, and local naming rows. It does not consume a host real, quotient representative, choice function, or completion point. The far-end side uses only boundary names and refusal rows. It does not consume T as a kernel object, state, Hist constructor, global frame, or equality witness. Thus neither side supplies unsupported content to the substrate. The reading is therefore an instance of the externalized-not-assumed discipline of Definition 9400.4. □

9404.7 Closure

The compressed form is

$$\begin{aligned} \mathsf{IrrLocatedReal}(r) &\Longrightarrow \exists\alpha,\ \mathrm{Carry}(Z_\alpha, r), \\ \forall\alpha, \quad \mathrm{FarEnd}(Z_\alpha) &\equiv_{\mathrm{apo}} \mathsf{T}, \\ \neg\exists\alpha, \quad \mathsf{T} &\in Z_\alpha. \end{aligned}$$

Thus a located irrational is a positive generation on a definite axis, while T is the common non-local far-end role of admissible axes. The former may be named by its stream, modulus, regularity, classifier, and ledger. The latter may only be named as socket, boundary, and refusal of objecthood.

Concretized in §1933: $\mathsf{LocatedReal}^{\uparrow}$ realizes the cataphatic process-name side as finite stream, regular-rational, dyadic-ledger, located-enclosure, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and NameCert rows.

Concretized in §8101: $\mathsf{IrrationalReal}^{\uparrow}$ realizes rational value-axis refusal through finite Real source, rational candidate, dyadic tolerance, witness, regular-rational readback, transport, replay, provenance, and NameCert rows.

Concretized in §1061: $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ realizes the non-internalized far-end role as finite socket, boundary, GAP ledger, local-inscription, observer-history, supply-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §1064: Apophatic fiber-far-end packets realize finite fiber, provenance, boundary-name, and non-internalization rows without exporting T as an object.

Chapter 9405

Information Strata and Non-Collapse

The word "axis" is useful as geometric prose, but it is not the strict object of the theory. The strict object is a non-collapse relation between information strata. A stratum may be a finite value layer, a process layer with stream and modulus data, a finite algebraic extension layer, a transcendental expression layer, or an apophatic far-end role. These strata are not made honest by drawing them as orthogonal coordinates. They are made honest by recording which carrier, classifier, ledger, and transport rows are available, and by refusing the retraction that would identify a higher stratum with a lower one.

9405.1 Value, process, and far-end strata

Definition 9405.1 (Information stratum)**.** *An* information stratum *is a BEDC presentation layer together with the rows needed to consume it. The basic strata used here are:*

$$\begin{aligned} L_0 &:= \textit{finite rational or dyadic value readback}, \\ L_1 &:= \textit{stream, modulus, regularity, location, and classifier data}, \\ T/\Omega &:= \textit{the apophatic far-end role of finite-readout fibers.} \end{aligned}$$

The first two are cataphatic strata: their objects are read by displayed rows. The last is apophatic: it is named by boundary sameness and non-internalization, not by a carrier element.

Definition 9405.2 (Finite shadow)**.** *For a process presentation $x \in L_1$ with sequence and modulus rows, its k-th finite shadow is*

$$\pi_k(x) := x_{\mathrm{seq}}(x_{\mathrm{mod}}(k)) \in L_0.$$

The map π_k is a finite readout request. It is not a retraction from the process stratum to the value stratum.

Definition 9405.3 (Non-collapse to a lower stratum)**.** *Let A be a stratum whose public reads include shadows in a lower stratum B. The predicate $\mathsf{NonCollapse}_B(a)$ means that no displayed B-object recovers a up to the classifier of A. For located real presentations over rational values, this specializes to*

$$\mathsf{NonCollapse}_{L_0}(x) :\Longleftrightarrow \neg\exists q : \mathsf{Rat}^{\uparrow},\ x \sim_{\mathsf{Real}^{\uparrow}} \eta(q),$$

where $\eta(q)$ is the degenerate located-real presentation with constant sequence q and modulus 0.

Lean4 (definition) `BEDC.Derived.NonCollapseInvariantUp.NonCollapseInvariant`
Lean4 (definition) `BEDC.Derived.NonCollapseInvariantUp.NonCollapseWitness`

Definition 9405.4 (No global lower recovery)**.** *For strata A and B with finite readout maps from A to B, a* global lower recovery *is a pair of displayed operations*

$$c : A \to B, \qquad r : B \to A$$

such that every $a : A$ is recovered by the composite:

$$a \sim_A r(c(a)).$$

The condition $\mathsf{NonCollapse}(A, B)$ *means that no such recovery is available from the displayed rows of the two strata.*

Theorem 9405.5 (Located irrationality is process non-collapse)**.** *A located irrational real has all of its finite rational shadows in L_0, but it is not recovered by any single L_0 value.*

Proof. The located-real carrier supplies the stream and modulus rows. Therefore Definition 9405.2 gives, for each k, a finite rational or dyadic observation in L_0.

Irrationality is the refusal of a rational value-axis collapse: there is no $q : \mathsf{Rat}^{\uparrow}$ whose degenerate located-real presentation classifies with the given located real under the $\mathsf{Real}^{\uparrow}$ classifier. Thus every finite shadow exists, while no lower-stratum point is a lossless recovery of the process. □

↪ variant `BEDC.Derived.NonCollapseInvariantUp.NonCollapseInvariant no rat retraction`

Corollary 9405.6 (Projection without retraction)**.** *The finite shadow maps $\pi_k : L_1 \to L_0$ are constructive, but the existence of these maps does not supply a general retraction of L_1 onto L_0.*

Proof. Each π_k is just a displayed finite read request. A retraction would assert that every process presentation can be recovered from a lower value object, contradicting Definition 9405.5 for located irrational presentations. □

Lean4 ✓ `BEDC.Derived.NonCollapseInvariantUp.NonCollapseWitness no rat retraction`

Theorem 9405.7 (Local non-collapse forbids global lower recovery)**.** *If some $a : A$ satisfies* $\mathsf{NonCollapse}_B(a)$*, then the finite shadow maps from A to B do not assemble into a global lower recovery of A from B.*

Proof. Suppose a global lower recovery c, r were available. Applying Definition 9405.4 to the particular object a gives

$$a \sim_A r(c(a)).$$

Thus the lower object $c(a) : B$ would recover a up to the classifier of A. This is exactly what $\mathsf{NonCollapse}_B(a)$ refuses. Hence the finite shadow maps may be consumed locally, but they do not provide a lossless reconstruction principle. □

9405.2 Algebraic and transcendental directions

Algebraic extension language gives one place where the geometric word "direction" can be made literal. Even there, the literal structure is linear independence over a base field, not automatic orthogonality. An inner product or bilinear form would have to be supplied separately before orthogonality could be asserted.

Definition 9405.8 (Finite algebraic direction)**.** *Let F be a field and let α be algebraic over F with minimal polynomial degree d. The algebraic direction generated by α is the finite extension carrier*

$$F(\alpha) = \left\{ a_0 + a_1\alpha + \cdots + a_{d-1}\alpha^{d-1} \mid a_i \in F \right\}.$$

Its strict content is the F-linear basis $1, \alpha, \ldots, \alpha^{d-1}$.

Theorem 9405.9 (Algebraic non-collapse is finite-dimensional)**.** *If α has degree $d > 1$ over F, then $F(\alpha)$ does not collapse to F, but it is still finitely generated as an F-vector space.*

Proof. The minimal polynomial degree gives the standard basis $1, \alpha, \ldots, \alpha^{d-1}$ for $F(\alpha)$ over F. If $\alpha \in F$, the minimal polynomial would have degree 1, contrary to $d > 1$. Hence the extension is genuinely larger than F. Since the displayed basis has d members, the non-collapse is finite-dimensional. □

Definition 9405.10 (Transcendental direction)**.** *If τ is transcendental over F, the expression layer $F(\tau)$ contains the infinite family*

$$1, \tau, \tau^2, \tau^3, \ldots.$$

This family is read as a transcendental direction only through its finite polynomial tests and expression ledgers; it is not the apophatic far-end role.

Theorem 9405.11 (Transcendental non-collapse is not the far end)**.** *For transcendental τ, the family $1, \tau, \tau^2, \ldots$ is F-linearly independent, but the resulting expression layer is still a carrier layer, not T/Ω.*

Proof. Any finite linear dependence among $1, \tau, \ldots, \tau^n$ would be a nonzero polynomial $p \in F[X]$ with $p(\tau) = 0$, contradicting transcendence. Thus the expression layer has infinitely many independent finite polynomial shadows.

Nevertheless $F(\tau)$ remains a mathematical carrier: its elements may be formed as expressions, compared by suitable relations, and transported by displayed ledgers. The apophatic far-end role has the opposite grammar. By Definition 9404.5, it is named only by $\equiv_{\text{apo}}$ and refuses object-level equality, membership, and function application. Hence transcendental non-collapse is not the far end. □

9405.3 Allowed folding and forbidden flattening

Definition 9405.12 (Dimension as information stratum)**.** *In this chapter,* dimension *means irreducible information stratum. A presentation has higher information dimension than a lower layer when its carrier, classifier, ledger, or transport obligations cannot be reconstructed from the lower layer's displayed objects.*

Principle 9405.13 (Finite folding is not infinite collapse)**.** *Finite quotient-like folds, such as residue classes modulo a displayed natural number, are not the same operation as a completion, supremum, or host-point collapse. A finite fold has finite representatives, a visible normal form, and a classifier ledger. An unwitnessed completion asks for an endpoint or point extracted from an infinite process or predicate family.*

Definition 9405.14 (Operation profile by information stratum)**.** *An operation has a* stratum profile *recording whether it preserves a stratum, requires a guard row, creates an extension layer, or asks for a bridge. In the real and algebraic fragments the intended profiles are:*

$$\begin{aligned} +, -, x^m &: \textit{finite layer-preserving operations when the algebra rows exist}, \\ \times &: \textit{layer-preserving bilinear combination under a multiplication ledger}, \\ / &: \textit{layer-preserving only with a displayed apartness-from-zero guard}, \\ x^{1/m} &: \textit{finite algebraic extension when the root is not already carried}, \\ \exp, \log &: \textit{process bridges requiring series, modulus, and branch ledgers}, \\ \mathrm{mod} n &: \textit{finite classifier fold with normal forms}, \\ \sup, \widehat{X},\ \textit{arbitrary completion} &: \textit{endpoint extraction requiring bridge data.} \end{aligned}$$

Theorem 9405.15 (Operation profiles separate folds from collapses)**.** *The finite fold profile of* mod$n$ *and the endpoint-extraction profile of completion or supremum are different BEDC obligations.*

Proof. A finite fold supplies a bounded carrier of representatives, a visible normal form, and a classifier ledger. Its consumer can inspect all relevant cases through finite rows.

An endpoint extraction from a completion, supremum, or arbitrary quotient does not have this shape. It asks a process family, predicate family, or compatibility ledger to produce a completed point. BEDC can consume such an endpoint only after the relevant modulus, witness, classifier transport, or NameCert bridge has been displayed. Therefore finite folding is an internal classifier operation, while unwitnessed endpoint extraction is a collapse boundary. □

Theorem 9405.16 (Classical point-line reading is a bridge)**.** *Reading value shadows, located processes, algebraic extensions, transcendental expression layers, and completion endpoints as points of one classical line is a useful bridge reading, not a kernel-local identification.*

Proof. The preceding results separate the strata. Located irrationality gives finite shadows without a rational-value retraction. Algebraic extension gives finite-dimensional non-collapse over a field. Transcendental extension gives infinitely many finite polynomial shadows while remaining a carrier layer. The apophatic far end refuses carrier membership altogether.

If all of these are consumed as points of one host line, several cross-stratum maps have been supplied: process to point, classifier to equality, predicate or process family to endpoint, and socket to element. Such maps may be studied as bridge data, but they are not present in the finite kernel merely because the geometric drawing uses one axis. □

Corollary 9405.17 (Axis prose is subordinate to non-collapse)**.** *Geometric axis language is legitimate only when the underlying NonCollapse, carrier, classifier, ledger, and bridge status have already been specified. In particular, algebraic extension supplies linear independence over the base field; orthogonality is an additional geometric structure, not an automatic consequence of adjoining a root.*

Proof. By Definition 9405.9, adjoining an algebraic element of degree $d > 1$ supplies a finite basis over the base field. The proof uses linear independence and spanning, not an inner product. By Definition 9405.16, any geometric compression of the strata must be read as bridge structure. Therefore axis prose is acceptable as orientation, but the invariant is non-collapse of information strata. □

Theorem 9405.18 (Axiom purity preserves information strata)**.** *The zero-axiom discipline of BEDC is the discipline of not treating a lower-stratum shadow as the higher-stratum object itself.*

Proof. The $\mathsf{LocatedReal}^{\uparrow}$ route preserves stream, modulus, regularity, classifier, and ledger rows rather than replacing them by a quotient real or host completion point. The $\mathsf{IrrationalReal}^{\uparrow}$ route records refusal of a rational collapse rather than deleting the located-real process surface. The real-completion route seals $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ data through displayed countable-name and classifier rows rather than importing a host endpoint. The apophatic route records T only as boundary sameness and non-internalization, never as a kernel object.

Each case keeps the relevant carrier, classifier, ledger, and bridge status visible. Thus axiom purity is not an extra metaphysical rule; it is the local refusal to flatten process data into points, classifiers into equality, predicate families into endpoints, or sockets into elements without a displayed certificate. □

Concretized in §1933: $\mathsf{LocatedReal}^{\uparrow}$ realizes the process stratum as finite stream, regular-rational, dyadic-ledger, located-enclosure, Real seal, transport, replay, provenance, and NameCert rows.

Concretized in §8101: $\mathsf{IrrationalReal}^{\uparrow}$ realizes rational value-axis non-collapse through displayed rational-candidate refusal and Real classifier rows.

Concretized in §2109: $\mathsf{AlgebraicReal}^{\uparrow}$ realizes finite algebraic non-collapse as polynomial-root, finite-basis, Real-seal, transport, replay, provenance, and NameCert data.

Concretized in §1061: $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ realizes the apophatic far-end role as a non-internalized socket rather than a carrier element.

Concretized in §1064: Apophatic fiber-far-end packets realize finite fiber, provenance, boundary-name, and non-internalization rows without exporting T as an object.

Concretized in §4212: Apophatic socket stacks are realized as finite boundary/refusal NameCert packets.

Chapter 9406

The audit-map family: a BEDC methodology

The BEDC.MetaCIC material is not a single opaque theorem bundle. It is a family of Lean substrate modules, paper audit maps, obstruction records, assembly chapters, and discharge-route chapters whose purpose is to keep the project's meta-theory readable at module granularity. This chapter names that family as a methodology.

Concretized in §806: Audit-map family methodology is realized as a finite BHist NameCert packet for positive, conditional, obstruction, frontier, and cross-map consumer rows.

Concretized in §823: Audit-map concrete inventories realize the family-level established, conditional, obstruction, frontier, and consumer rows as finite NameCert data.

The audit-map family sits beside chapter 9400 and chapter 9389. Apophatic naming says that an open dependency must have a stable name rather than a hidden status. Generator-closure discipline says that a proof may range only over the authorized generator whose closure is displayed. The audit-map family is the third concrete application: every major MetaCIC substrate is required to publish its established content, conditional content, obstruction content, and frontier.

9406.1 The setup

BEDC.MetaCIC is distributed across several formal modules: subject reduction, confluence, normalisation, substitution, decidable checking, typed examples, beta reduction, and closure preservation. The modules do not all have the same closure status. Some nodes are unconditional. Some are conditional on an explicit interface. Some expose an obstruction theorem. Some are examples or boundary checks that prevent a nearby over-reading.

A paper chapter that merely announced "MetaCIC is partially formalised" would lose the important information. The reader would not know which substrate exports usable theorems, which substrate exports a parameterised route, which substrate exports a no-go result, and which substrate is needed by a later assembly. BEDC's apophatic naming discipline rejects that kind of blur. Each open cell has to be visible.

The audit-map family is the systematic answer. Each major MetaCIC substrate gets a paper map. The map says what the Lean substrate currently establishes, what it establishes only under named parameters, what it blocks, and how its frontier connects to the rest of the consistency path. The family is therefore not a retrospective aside. It is part of the method by which BEDC prevents partial closure from turning into implicit closure.

9406.2 What an audit map is

Definition 9406.1 (Audit map)**.** *An* audit map *is a paper chapter focused on one Lean substrate submodule. It records the checked inventory, the conditional inventory, the obstruction inventory, and the cross-module frontier of that submodule without claiming that the inventory has closed the neighbouring modules.*

The definition has several layers.

First, the unit is a paper chapter, not a footnote. The chapter format matters because each substrate has enough structure to require its own setup, local vocabulary, and boundary statement.

Second, the focus is one Lean substrate submodule or one tightly bounded assembly. A confluence audit map is about confluence. A substitution audit map is about closed syntax algebra. A consistency assembly map is about the normal-form consistency branch and the subject-reduction transport boundary. The point is module-level granularity.

Third, established theorems are listed by Lean-target references in the local map. Those references are usually rendered as variants because the map is an inventory site rather than the canonical primary theorem site. This keeps the paper honest about what the Lean kernel actually checks while avoiding a second primary claim for the same formal object.

Fourth, conditional or parameterised theorems are not weakened into prose. A conditional diamond remains a conditional diamond. A subject-reduction route that needs a discharge bundle remains a route that needs that bundle. A normalisation theorem mediated by candidate evidence remains mediated by that evidence.

Fifth, obstruction and no-go results are first-class entries. The subject reduction obstruction chapter is not a failure appendix. It records the closed application obstruction and the exact interface needed to move past it. That is BEDC's apophatic naming discipline operating inside formal meta-theory.

Sixth, each map names its frontier and cross-module dependencies. The reader can see which facts are consumed by synthesis, which facts are local sanity checks, and which facts would be needed before a stronger assembly is allowed.

9406.3 The nine audit map chapters of BEDC.MetaCIC

The family has nine primary audit-map chapters.

1. chapter 9377 records the subject reduction obstruction. It names the dependent-codomain and closed-application pressure point, records the parameterised preservation interface, and prevents unparameterised closed subject reduction from being treated as available.

2. chapter 9375 records the closed-MetaCIC consistency assembly. It separates closed-normal consistency from full closed consistency and marks subject reduction as the transport boundary between the two.

3. chapter 9378 records the confluence substrate. It distinguishes reduction algebra, atom fragments, closed-normal fragments, and the substitution-sensitive diamond route.

4. chapter 9383 records the normalisation substrate. It shows the candidate-mediated strong-normalisation route and keeps candidate-free global normalisation outside the present claim.

5. chapter 9380 records the substitution substrate. It supplies closed shift, closed substitution, and closed composition fragments without converting them into beta typing preservation.

6. chapter 9379 records the decidable-checking substrate. It exposes executable equality and type-checking surfaces while refusing to call them a global decision oracle for meta-theory.
7. chapter 9382 records the typed-example substrate. It treats examples as checked calibration surfaces rather than as silent evidence for arbitrary closed terms.
8. chapter 9384 records the beta-reduction substrate. It separates one-step beta structure, star closure, normality, and transport interfaces needed by neighbouring maps.
9. chapter 9381 records the closure-preservation substrate. It tracks which generators preserve closedness and which preservation statements belong to other modules.

The family also has four companion meta chapters. chapter 9385 reads the maps together as the closed-CIC critical path. chapter 9386 consolidates the open obligations so the same frontier is not scattered across prose. chapter 9376 elevates the strongest available closed-normal result as a main-result node. chapter 9374 records the concrete routes by which the subject-reduction boundary could be discharged.

Taken together, these chapters form a substrate-level inventory. The inventory is deliberately redundant in one sense: the same boundary can appear in a local map, in synthesis, and in open problems. That redundancy is controlled. It keeps the boundary visible from the module view, the critical-path view, and the obligation view.

9406.4 Common structure across audit maps

The maps use a shared writing structure.

Setup. The map names the Lean files, relations, predicates, and interfaces that form the local substrate.

Established. The map lists the checked results the paper may cite directly.

Conditional. The map records parameterised theorems and the exact inputs they require.

Obstruction. The map records counterexamples, impossibility results, or shape restrictions that block a stronger reading.

Frontier. The map states the strongest nearby claim that remains outside the present substrate.

Cross-reference. The map identifies the neighbouring maps that consume or supply its facts.

Boundary. The map ends by stating what the chapter does not claim.

Principle 9406.2 (Audit-map template)**.** *An audit map is complete only when it records positive content, conditional content, obstruction content, frontier content, and cross-module use. Omitting any one of these parts turns a partial-closure map into an ambiguous progress summary.*

This template is intentionally stricter than an ordinary literature overview. It is meant to be read against the Lean substrate and against the paper-Lean audit. A result that is checked but local remains local. A parameter remains a parameter. A blocked route remains blocked. A cross-module use must point to the map that supplies the relevant node.

9406.5 Methodology versus other disciplines

The audit-map family differs from a cataphatic results-first discipline. A cataphatic presentation often foregrounds the main theorem and leaves adjacent work implicit. BEDC cannot use that style for MetaCIC because the important information is the shape of the partial closure: which generators are closed, which transports are conditional, and which obstructions are theorem-level facts.

The family also differs from silent proof holes. A hidden `sorry`, an `admit`, or an unrecorded host assumption would make the map dishonest. BEDC forbids that move. The paper-side analogue is equally strict: an open obligation may be named, routed, and studied, but it may not be treated as internal theorem content.

Finally, the family differs from a single high-level retrospective. A single summary can help orientation, but it cannot replace module-level audit. The BEDC pattern is module-level maps plus cross-module synthesis. The local maps preserve granularity; the synthesis shows how the local nodes compose or fail to compose.

9406.6 When to use this methodology

The audit-map methodology is appropriate when a formalisation project has many submodules, each with a different closure state. It is especially useful for meta-theory, where a theorem in one module is often a premise, obstruction, or calibration check for another module. Subject reduction, confluence, normalisation, substitution, decidability, examples, beta reduction, closure preservation, and consistency assembly are exactly that kind of landscape.

It is not the right tool for every theorem. A single self-contained result with a direct proof does not need a family of audit maps. In that case the paper can state the theorem, give the proof, and cite the Lean target at the primary site. The audit-map family is for a different shape: distributed partial closure with named obligations and cross-module dependencies.

9406.7 Audit-map family as apophatic naming in action

chapter 9400 gives the general rule: when a dependency cannot be generated, defined, certified, or discharged inside the displayed substrate, the dependency must be named. The audit-map family is that rule applied to MetaCIC formalisation.

Each audit map is a collection of apophatic names at module scale. It names the open obligations, obstruction points, conditional interfaces, and frontier claims that would otherwise disappear into prose. The family as a whole is a substrate-level inventory of those names. It tells the reader where the project has generator-internal closure and where the project has an exposed socket.

This is why the methodology pairs with chapter 9389. Generator closure says that a certificate ranges over the objects produced by a displayed generator. The audit-map family asks the same question at chapter scale: which generator, relation, or interface licenses the local claim, and where does that licence stop?

9406.8 Boundary

The audit-map family does not solve any substantive proof obligation by naming it. It does not prove subject reduction. It does not turn conditional confluence into unconditional confluence. It

does not turn candidate-mediated normalisation into a candidate-free theorem. It does not lift closed-normal consistency to full closed consistency.

Its role is process discipline, not result discipline. It makes the proof state explicit enough that later work can cite the right node, refuse the wrong node, and see the exact dependency that would have to be supplied. That is a BEDC methodology: not silence about open cells, not hidden assumptions, and not blanket confidence, but a family of audit maps whose open names remain visible.

Concretized in §808: Audit-map templates are realized as finite BHist packets exposing setup, established, conditional, obstruction, frontier, cross-map route, boundary, provenance, and local NameCert rows.

Concretized in §810: Audit-map family ledgers realize module-level audit maps as finite BHist, hsame, Cont, Pkg, and NameCert rows with explicit non-escape from hidden closure.

Concretized in §811: The audit-map family methodology is realized as a finite MetaCIC audit-family NameCert packet over local maps, synthesis rows, obstruction rows, frontier rows, and non-escape ledgers.

Concretized in §812: Audit-map frontier indices are realized as finite BHist packets over local maps, neighbouring-map links, obstruction rows, frontier rows, synthesis consumers, hsame, Cont, Pkg, and NameCert rows.

Concretized in §813: The audit-map family is realized as a finite NameCert methodology packet with a SubstitutionAuditMap handoff.

Concretized in §814: The audit-map family methodology is realized as a finite coverage ledger NameCert.

Concretized in §815: The audit-map family methodology is realized as a finite obligation-ledger NameCert packet over positive, conditional, obstruction, frontier, and cross-module audit rows.

Concretized in §816: A finite NameCert packet records the audit-map family rows as concrete BHist carrier, classifier, obstruction, frontier, and cross-module-use data.

Concretized in §817: The audit-map family is realized as a finite NameCert packet for inventory, obstruction, frontier, and cross-map routing.

Concretized in §825: The audit-map template packet realizes the methodology as finite positive, conditional, obstruction, frontier, and sibling-use rows.

Concretized in §826: Audit-map obstruction sockets realize obstruction and frontier rows as finite BHist, hsame, Cont, Pkg, and NameCert packets with non-escape from hidden closure.

Concretized in §833: Audit-map frontier packets realize the audit-map family as finite checked, conditional, obstruction, frontier, cross-map, provenance, and NameCert rows.

Concretized in §839: Audit-map dependency weaves realize cross-map frontier routing as finite BHist, obstruction, synthesis-consumer, hsame, Cont, Pkg, and NameCert rows.

Chapter 9407

BEDC GroundCompiler module audit map

9407.1 The setup

`BEDC.GroundCompiler` is the main Lean formalisation of the BEDC self-compilation path. It is parallel to `BEDC.MetaCIC` in project role, but not in subject matter. `BEDC.MetaCIC` formalises a closed CIC substrate: terms, reduction, typing-adjacent interfaces, and the consistency critical path. `BEDC.GroundCompiler` formalises the self-compile surface: event flows, source channels, generated recognizers, certificate gates, address analysis, cannot-claim ledgers, and streaming compiler discipline.

The audit question is therefore different from the MetaCIC audit maps in chapter 9406. The local issue is not whether CIC reduction has a global confluence or preservation theorem. The local issue is whether a compiler-facing artifact is allowed to count as a BEDC input, whether a displayed code channel is lossless at the event-flow layer, whether generated recognizers and certificate gates mediate exports, and whether reports or motifs can silently replace proof or certificate evidence.

This chapter records the GroundCompiler module as an audit map. It is an inventory site. It cites checked Lean targets as variants, separates Prop statement cells from their proof theorems, names the open cells, and states the boundary of the self-compilation claim.

9407.2 Submodule inventory

The GroundCompiler directory contains the following Lean sibling files.

File group	Audit role
`EventFlow`, `SourceChannel`, `ChannelEncoding`, `SourceNormalizer`	raw event-flow surface, source/channel separation, legal stream encoding, and normal-address discipline
`PackageGenerated`, `NameCertGenerated`, `DerivCertGenerated`, `TheoremGenerated`, `ChapterFlow`, `ChapterManuscript`	generated package, name-certificate, derivation-certificate, theorem, and chapter recognizer layers
`RecognizerFlows`, `SemanticMotif`, `MotifReportPrototype`, `MotifReportCounts`	recognition evidence, motif occurrence, motif report structure, and non-evidence boundaries
`MetricsFlow`, `MetricsP4Pipeline`, `MetricsP4Adequacy`, `AnalysisPipeline`	metric reports, P4 adequacy surfaces, analysis stages, and report conservativity
`SelfHostingCompilerFlow`, `SelfHostingLadder`, `SelfHostingReports`	self-hosting compiler candidates, hierarchy coverage, bootstrap boundaries, and reporting layers
`ImplementationInterface`, `ImplementationAdequacy`, `ProgrammaticRealization`, `MinimalPrototype`	implementation-facing interfaces, adequacy wrappers, and prototype witnesses
`CaseStudies`, `HigherCaseStudies`, `PackageNameCertPrototype`, `TheoremProofPrototype`, `DerivCertReports`, `SourceReport`	case-study and report surfaces used to calibrate what evidence may and may not license
`MainTheorems`	top-level compiler audit statements and theorem wrappers over the checked submodules

This split mirrors the methodology of the MetaCIC audit maps, but the exported facts are compiler-layer facts. The file tree is not a single theorem with hidden premises. It is a ledger of what the self-compilation path may treat as input, output, recognition evidence, certificate evidence, metric evidence, and explicit non-evidence.

9407.3 Core statement defs

`MainTheorems.lean` exposes three top-level Prop statement cells.

- ↪ variant `BEDC.GroundCompiler.MainTheorems.GlobalConservativityStatement`

  records event-flow conservativity, legal encoding of flow images, exclusion of structural hidden inputs, metric conservativity, and the non-input status of host package, name-certificate, derivation-certificate, and closure-certificate data. It is paired with

  ↪ variant `BEDC.GroundCompiler.MainTheorems.global conservativity`

.

- ↪ variant BEDC.GroundCompiler.MainTheorems.CompilerLayerAddressAnalysisStatement

  records channel round-trip, legal-stream completeness, code-not-proof, metric-not-certificate, and global conservativity content. It is paired with

  ↪ variant BEDC.GroundCompiler.MainTheorems.compiler layer address analysis layer

  .

- ↪ variant BEDC.GroundCompiler.MainTheorems.NoHiddenInputStreamingCompilerStatement

  records the streaming compiler surface: formal event-flow inputs, legal encoding and decoding, hidden host-structure exclusion, recognizer generatedness, certificate-mediated export, theorem-code and chapter-code bijections, classifier quotient limits, source-channel separation, metric conservativity, self-hosting removal of hidden compiler input, visible bootstrap boundary, and global conservativity. It is paired with

  ↪ variant BEDC.GroundCompiler.MainTheorems.no hidden input streaming compiler

  .

The important audit distinction is that these are not statement-only names covered by a vacuous theorem. Each Prop cell above has a companion theorem whose conclusion is the cell itself. The proof terms compose existing GroundCompiler lemmas and definitions. Thus the statement defs act as named aggregate interfaces, while the theorem names are the checked aggregate witnesses.

Within the inspected top-level file, there is no unpaired `*Statement` Prop cell of the kind that would have to be cited only as a paper-side target. GroundCompiler still has many lower-layer definitions and interfaces, but the three named aggregate statements above are theorem-paired inside `MainTheorems.lean`.

9407.4 Established theorems

The checked GroundCompiler surface has several core theorem families.

- The no-hidden-input and channel facts establish that structural hidden compiler data do not become formal input and that legal event-flow encoding is lossless at the channel layer:

  ↪ variant BEDC.GroundCompiler.MainTheorems.no hidden input

  ↪ variant BEDC.GroundCompiler.MainTheorems.channel bijection

  ↪ variant BEDC.GroundCompiler.MainTheorems.channel code lossless

  .

- The code/proof and source/channel facts prevent readable code, substrings, and source-carry transformations from being promoted into theorem or event evidence:

  ↪ variant BEDC.GroundCompiler.MainTheorems.code not proof

  ↪ variant BEDC.GroundCompiler.MainTheorems.source channel separation

  ↪ variant BEDC.GroundCompiler.MainTheorems.carry not channel rewrite

  .

- The generated-structure and recognizer facts keep packages, certificates, chapters, theorem identifiers, manifests, and host encodings outside the formal compiler input set unless mediated by generated recognition evidence:

  ↪ variant BEDC.GroundCompiler.MainTheorems.structure emergence

  ↪ variant BEDC.GroundCompiler.MainTheorems.recognizer generatedness

  ↪ variant BEDC.GroundCompiler.MainTheorems.accepted export

  .

- The theorem-code and chapter-code layers establish canonical code round-trips while refusing to identify proposition text with proof evidence or chapter topic with chapter code:

  ↪ variant BEDC.GroundCompiler.MainTheorems.theorem code bijection

  ↪ variant BEDC.GroundCompiler.MainTheorems.theorem code not proof

  ↪ variant BEDC.GroundCompiler.MainTheorems.chapter code bijection

  ↪ variant BEDC.GroundCompiler.MainTheorems.topic not chapter code

  .

- The classifier, metric, motif, and bootstrap layers record the local limits of address analysis and reporting:

  ↪ variant BEDC.GroundCompiler.MainTheorems.classifier quotient many to one

  ↪ variant BEDC.GroundCompiler.MainTheorems.normal address requires ledger

  ↪ variant BEDC.GroundCompiler.MainTheorems.metric conservativity

  ↪ variant BEDC.GroundCompiler.MainTheorems.similarity not identity

  ↪ variant BEDC.GroundCompiler.MainTheorems.declared bootstrap boundary

  .

- The top-level aggregate theorems package the module-facing reading:

  ↪ variant BEDC.GroundCompiler.MainTheorems.global conservativity

  ↪ variant BEDC.GroundCompiler.MainTheorems.compiler layer address analysis layer

  ↪ variant BEDC.GroundCompiler.MainTheorems.no hidden input streaming compiler

  .

These theorems make GroundCompiler a checked self-compilation audit layer. They do not make the compiler a source of new mathematical axioms. They show which event-flow, code, recognizer, certificate, metric, motif, and bootstrap claims are licensed inside the displayed compiler substrate.

9407.5 Channel boundary discipline

The channel layer has one especially sharp audit role: it turns an EventFlow into a single display string without erasing the event boundaries that make the flow a formal input. The relevant encoding is the Zeckendorf-safe channel of chapter 9208. Its body map sends raw 0 to channel 0 and raw 1 to channel 10, and each event ends with the channel terminator 11. Thus the legal channel language is

$$\mathsf{LegalZStream} = \big((0 \mid 10)^* 11\big)^*.$$

The audit content is not the informal observation that this language is prefix-readable. It is the conjunction of four checked boundaries. First, Definition 9208.6 proves that no encoded body contains adjacent 11, and Definition 9208.7 therefore identifies the first adjacent 11 as the event terminator. Second, Definition 9208.16 and Definition 9208.17 prove event-level and flow-level readback. Third, Definition 9208.19 fixes the exact surface on which

$$\mathsf{EventFlow} \cong \mathsf{LegalZStream}$$

is allowed to be read. Fourth, Definition 9208.22 separates the legal channel from host parsers, manifests, package objects, theorem identifiers, and other structural hidden input.

This boundary also explains why the event-flow layer cannot be replaced by a bare bitstream. The erasure map of Definition 9207.7 forgets event segmentation, and Definition 9207.8 therefore refuses any decoder that tries to reconstruct every event flow from its flattened mark stream. The channel encoder repairs this only by making the boundaries explicit as part of the legal display language. It does not license a boundary-free input format.

The source-level Zeckendorf classifiers remain separate. $\mathsf{ZNormal}$, ZCarry, and the displayed carry row $011 \rightsquigarrow 100$ are relations on recognized history flows. They are not substring rewrites on a channel stream. The source–channel separation theorem Definition 9209.27 is the audit fence: a consumer may decode a legal stream to an event flow, apply a licensed source-level normalizer to that event flow, and compile the result again; it may not rewrite channel glyphs directly and then treat the result as if it were a recognized source event.

Consequently the GroundCompiler audit map admits the following channel reading and no stronger one. The encoding $0 \mapsto 0$, $1 \mapsto 10$, $\mathrm{End} \mapsto 11$ is a conservative, self-delimiting display of finite event-flow boundaries. It is not a $\mathsf{NameCert}$, not a theorem code, not a proof object, not a gap ledger, not a source-level Zeckendorf carry proof, and not evidence for entropy, continuum, topology, geometry, or physical space emergence. Those readings require separately named bridges and separately displayed certificate surfaces.

Concretized in §829: GroundCompiler event-flow audit packets make the lossless source/channel and no-hidden-input boundary a concrete NameCert surface.

Concretized in §830: The GroundCompiler event-flow audit route is realized as a finite channel, recognizer, certificate-gate, and non-evidence NameCert packet.

Concretized in §2258: GroundCompiler audit routes are realized as finite event-flow, MetaCIC-boundary, no-hidden-input, transport, replay, provenance, and local NameCert rows.

Concretized in §1094: GroundCompiler flow-level round trips are realized as finite encoding, decoding, legal-stream, transport, route, provenance, and NameCert rows.

Concretized in §6872: GroundCompiler audit-map closures realize compiler event-flow, source-channel, recognizer, certificate-gate, metric-report, and non-evidence rows as finite NameCert data.

9407.6 Relation to MetaCIC

GroundCompiler and MetaCIC cooperate, but their audit maps must remain separate. GroundCompiler executes the BEDC self-compile route from source-like event flows to generated recognizers, code ledgers, certificate gates, and cannot-claim rows. MetaCIC formalises the closed CIC substrate in which reduction, normalisation, confluence, and consistency boundaries are studied.

The relation is therefore complementary. GroundCompiler can say that a compiler-facing artifact is generated, encoded, decoded, ledgered, or excluded as a hidden input. MetaCIC can

say that a closed term, reduction path, or candidate-mediated normalisation route has a particular meta-theoretic status. Neither side replaces the other. A GroundCompiler theorem about code not being proof is not a MetaCIC subject-reduction theorem. A MetaCIC confluence bridge is not a GroundCompiler certificate gate.

This separation is the same discipline used by the audit-map family: each map exports its own local content and names its own frontier. Cross-reference is allowed; substitution of one module's theorem for another module's missing edge is not.

9407.7 Open obligations and conditional statements

The visible open cells are compiler-layer cells, not closed-CIC proof cells. They include the distinction between self-hosting as a supplied compiler-flow condition and full removal of hidden compiler input; the distinction between a metric or motif report and certificate evidence; the distinction between a theorem code channel and a proof; and the distinction between a classifier quotient and raw event-flow identity.

The theorem

```
↪variant BEDC.GroundCompiler.MainTheorems.self hosting removes hidden compiler
```

is conditional on a self-hosted compiler flow and hierarchy coverage. It is a valid route from those inputs to the no-hidden-compiler conclusion, but it does not assert that every candidate compiler already satisfies those inputs. Similarly,

```
↪variant BEDC.GroundCompiler.MainTheorems.cannot claim registry mandatory
```

requires registry adequacy before the listed cannot-claim ledger rows are available.

Those conditions are useful precisely because they are explicit. They prevent the paper from reading reports as proof, motifs as mature objects, code as acceptance, source substrings as decoded events, or self-hosting prose as a discharged compiler tower.

9407.8 Apophatic framing

The open cell for GroundCompiler is the *compiler-layer evidence gap*. The module can check that event-flow code is legal and lossless, that hidden host structures are excluded from formal input, that generated recognizers mediate recognition evidence, and that certificate gates mediate accepted exports. It also says what those facts do not license.

That is the apophatic role of this audit map. The chapter gives names to the non-claims: code is not proof, report is not evidence, motif similarity is not object identity, classifier quotient is not raw equality, and conditional self-hosting is not an unconditional compiler tower. These names prevent the self-compilation path from acquiring silent extra power through prose.

9407.9 Boundary

This chapter does not solve the remaining GroundCompiler obligations. It does not prove that every compiler candidate is self-hosted. It does not turn metric reports, motif reports, theorem code, or chapter code into proof or certificate evidence. It does not collapse classifier equality into raw event-flow equality. It does not replace any MetaCIC audit map.

The allowed reading is narrower and stronger: GroundCompiler supplies a checked compiler-layer audit surface for BEDC self-compilation, with explicit theorem witnesses for conservativity,

address analysis, and no-hidden-input streaming compilation. Everything beyond that surface remains named at the boundary.

Concretized in §827: GroundCompiler audit packets realize the compiler-layer audit map as finite event-flow, source-channel, recognizer, certificate-gate, report-boundary, and no-hidden-input NameCert rows.

Concretized in §820: GroundCompiler recognizer and non-evidence rows are packaged as a finite audit packet.

Concretized in §828: The recognizer-boundary packet makes the GroundCompiler audit-map evidence boundary a finite NameCert carrier.

Concretized in §821: GroundCompiler generated recognition is realized as a finite BHist NameCert packet with accepted-export and finite-trace handoff rows.

Concretized in §822: GroundCompiler verdict rows are realized as a finite NameCert packet rather than as host-side audit prose.

Concretized in §832: GroundCompiler evidence boundaries are realized as finite BHist NameCert packets separating event-flow evidence, generated recognizers, compiler reports, code channels, and non-input ledgers.

Concretized in §834: The GroundCompiler audit map is realized as a finite route-compiler NameCert packet.

Concretized in §831: GroundCompiler audit evidence is realized as a finite NameCert witness packet with report and refusal non-escape rows.

Concretized in §819: Legal event-flow channels are realized as a finite NameCert packet with lossless decode and hidden-input exclusion rows.

Concretized in §836: GroundCompiler audit maps are realized as finite NameCert packets over theorem inventory, compiler-layer frontiers, generated recognizer rows, report-boundary rows, and non-escape ledgers.

Concretized in §1048: GroundCompiler recognizer flows are realized as finite event-flow, source-channel, generated-recognizer, certificate-gate, non-evidence, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1360: GroundCompiler audit-map route budgets are realized as finite event-flow, source-channel, recognizer, certificate-gate, report-boundary, and cannot-claim NameCert rows.

Chapter 9408

Theorem-DNA and Theorem Complexity Scoring

The GroundCompiler theorem recognizer does not treat a theorem as a single statement string. In `TheoremGenerated.lean`, theorem recognition is role-complete: the recognized flow must expose statement, dependencies, proof, certificates, ledger, status, canonical site, and closing seal. This chapter packages that role-complete reading as theorem-DNA. The word DNA names a typed locus profile over event flows; it does not import biological structure.

9408.1 Role-complete theorem flows

Definition 9408.1 (Theorem candidate and recognizer)**.** *A theorem candidate is an event flow* $T : \mathsf{EventFlow}$*. A generated theorem recognizer is a generated recognizer flow* R*, and its recognition relation is written*

$$\mathsf{RecognizesThm}(R, T).$$

Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremCandidateFlow`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.GeneratedTheoremRecognizer`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremRecognitionRelation`

Definition 9408.2 (Theorem role loci)**.** *The theorem role profile is the eight-locus family*

$$\mathcal{R}_\Theta = \{\mathsf{statement}, \mathsf{dependencies}, \mathsf{proof}, \mathsf{certificates}, \mathsf{ledger}, \mathsf{status}, \mathsf{canonicalSite}, \mathsf{closingSeal}\}.$$

For a theorem candidate T*, a recognizer* R*, a subflow* G*, and a role* $\rho \in \mathcal{R}_\Theta$*, write*

$$\mathsf{Role}_R(T, G, \rho)$$

for the role-subflow predicate assigning G *to the role* ρ *inside the recognized theorem flow.*

Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremRole`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremRoleSubflow`

Definition 9408.3 (Weak theorem-DNA)**.** *For a recognizer* R *and theorem candidate* T*, a weak theorem-DNA is an eight-tuple*

$$D_R(T) = (G_{\mathsf{statement}}, G_{\mathsf{dependencies}}, G_{\mathsf{proof}}, G_{\mathsf{certificates}}, G_{\mathsf{ledger}}, G_{\mathsf{status}}, G_{\mathsf{canonicalSite}}, G_{\mathsf{closingSeal}})$$

such that each component is assigned to its matching role:

$$\mathsf{Role}_R(T, G_\rho, \rho) \qquad \textit{for every } \rho \in \mathcal{R}_\Theta.$$

The family of such profiles is denoted $\mathsf{DNA}_{\mathsf{weak}}(R, T)$.

Definition 9408.4 (Anchored and sound theorem-DNA)**.** *A weak theorem-DNA is anchored when every role subflow is also a source subflow of T. It is sound when the recognizer participates in a sound theorem-flow recognition of T.*

Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremSourceSubflow`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.SoundTheoremFlow`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.AcceptedTheoremFlow`

Theorem 9408.5 (Accepted theorem flows have theorem-DNA)**.** *If T is an accepted theorem flow, then there are a generated theorem recognizer R and a weak theorem-DNA $D_R(T)$.*

Proof. Accepted theorem flow means that some recognizer R satisfies $\mathsf{SoundTheoremFlow}(R, T)$. Sound theorem flow contains complete theorem flow recognition. Complete recognition unfolds to the existence of the eight role subflows for statement, dependencies, proof, certificates, ledger, status, canonical site, and closing seal. These eight witnesses form the required theorem-DNA tuple. □

Corollary 9408.6 (A theorem is not a single-locus object)**.** *An accepted theorem flow is not determined by the statement locus alone at the recognition layer. The complete profile has eight role loci.*

Proof. The existence theorem supplies all eight loci, and complete theorem-flow recognition requires all of them. □

9408.2 Statement, code, and quotient boundaries

Theorem 9408.7 (Statement alone is not theorem-DNA)**.** *A statement role subflow without a proof role subflow does not establish theoremhood.*

Proof. The checked theorem `statement_alone_not_theorem` states that a candidate with a statement role, but with no proof role for every recognizer recognizing that candidate, is not a theorem flow. Thus the statement locus is only one role in the theorem-DNA profile. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.statement alone not theorem`

Theorem 9408.8 (Shared statement does not determine theorem flow)**.** *The same statement subflow can occur as the statement role of two distinct theorem candidate flows.*

Proof. This is exactly the content of the checked witness `same_statement_multiple_flows`. The theorem-DNA profile therefore cannot be collapsed to proposition text. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.same statement multiple flows`

Definition 9408.9 (Theorem code)**.** *The code of a theorem candidate T is its GroundCompiler flow encoding:*

$$\mathsf{Code}(T) = \mathsf{FlowEncoding}(T).$$

Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremCode`

Theorem 9408.10 (Theorem code is injective but not proof)**.** *Theorem code equality refines theorem-flow equality, but legal theorem code does not by itself supply proof-sound theorem recognition.*

Proof. The injectivity part is the checked theorem `theorem_code_injective`. The non-proof part is the checked theorem `theorem_code_is_not_proof`, which exhibits legal theorem code decoding to a candidate with no proof-sound theorem recognition. Code is therefore a lossless flow address, not the complete theorem-DNA profile. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.theorem code injective`
↪ variant `BEDC.GroundCompiler.TheoremGenerated.theorem code is not proof`

Definition 9408.11 (Theorem phenotype)**.** *Given a theorem classifier relation Q on theorem candidate flows, two theorem flows have the same phenotype when $Q(T, U)$ holds. The phenotype class of T is*

$$[T]_Q = \{U \mid Q(T, U)\}.$$

The concrete theorem-DNA profile is the genotype layer; the classifier class is the phenotype layer.

Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremClassifierRelation`
Lean4 (definition) `BEDC.GroundCompiler.TheoremGenerated.TheoremClassifierQuotient`

Theorem 9408.12 (Phenotype can fold distinct code)**.** *If a theorem classifier relates two distinct theorem flows, it can identify flows with distinct theorem code.*

Proof. The checked theorem `classified_theorem_objects_not_code_bijective` states that a classifier quotient relating distinct theorem flows collapses distinct theorem code. Thus genotype and phenotype are different audit layers. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.classified theorem objects not code bijective`

9408.3 Mass vectors and weighted score

Definition 9408.13 (Event-flow mass)**.** *For a raw event w, let $|w|_{\mathsf{raw}}$ be its mark count. For an event flow*

$$S = (w_0, \ldots, w_{n-1}),$$

define

$$|S|_{\mathsf{flow}} = n + \sum_{i=0}^{n-1} |w_i|_{\mathsf{raw}}.$$

The summand n records event boundaries. Boundary erasure is not a lossless operation at the event-flow layer.

Definition 9408.14 (DNA mass vector)**.** *For a theorem-DNA profile $D_R(T)$, define its mass vector by*

$$\mathbf{m}(D_R(T)) = (m_{\mathsf{statement}}, m_{\mathsf{dependencies}}, m_{\mathsf{proof}}, m_{\mathsf{certificates}}, m_{\mathsf{ledger}}, m_{\mathsf{status}}, m_{\mathsf{canonicalSite}}, m_{\mathsf{closingSeal}}),$$

where $m_\rho = |G_\rho|_{\mathsf{flow}}$.

Definition 9408.15 (Complexity vector)**.** *The recognizer overhead is* $\omega(R) = |R|_{\mathsf{flow}}$*. The theorem-DNA complexity vector is*

$$\mathsf{ComplexVec}(R, D_R(T)) = (m_{\mathsf{statement}}, m_{\mathsf{dependencies}}, m_{\mathsf{proof}}, m_{\mathsf{certificates}}, m_{\mathsf{ledger}}, m_{\mathsf{status}}, m_{\mathsf{canonicalSite}}, m_{\mathsf{closingSeal}}, \omega(R))$$

Definition 9408.16 (Weight profile and score)**.** *A weight profile is a vector*

$$\mathbf{w} = (w_{\mathsf{statement}}, w_{\mathsf{dependencies}}, w_{\mathsf{proof}}, w_{\mathsf{certificates}}, w_{\mathsf{ledger}}, w_{\mathsf{status}}, w_{\mathsf{canonicalSite}}, w_{\mathsf{closingSeal}}, w_R) \in \mathbb{N}^9.$$

The theorem-DNA score is the dot product

$$\mathsf{Score}_{\mathbf{w}}(R, D) = \mathbf{w} \cdot \mathsf{ComplexVec}(R, D).$$

Definition 9408.17 (Accepted theorem complexity)**.** *For an accepted theorem flow* T*, define*

$$\mathsf{TC}_{\mathbf{w}}(T) = \min\{\mathsf{Score}_{\mathbf{w}}(R, D) \mid D \in \mathsf{DNA}_{\mathsf{weak}}(R, T) \text{ and } \mathsf{SoundTheoremFlow}(R, T)\}.$$

Theorem 9408.18 (Accepted theorem complexity is defined)**.** *For every accepted theorem flow* T *and every weight profile* $\mathbf{w}$*,* $\mathsf{TC}_{\mathbf{w}}(T)$ *is defined as a natural number.*

Proof. By Definition 9408.5, the minimizing set contains at least one sound theorem-DNA profile. Every score in the set is a natural number. A non-empty subset of $\mathbb{N}$ has a least element. □

Theorem 9408.19 (Score monotonicity)**.** *If two theorem-DNA complexity vectors satisfy componentwise*

$$\mathsf{ComplexVec}(R, D) \leq \mathsf{ComplexVec}(R', D'),$$

then for every weight profile $\mathbf{w}$*,*

$$\mathsf{Score}_{\mathbf{w}}(R, D) \leq \mathsf{Score}_{\mathbf{w}}(R', D').$$

Proof. Write the two complexity vectors as $(a_1, \ldots, a_9)$ and $(b_1, \ldots, b_9)$ with $a_i \leq b_i$ for every i. Since every weight w_i is a natural number, $w_i a_i \leq w_i b_i$. Summing the nine inequalities gives the score inequality. □

Theorem 9408.20 (Scalar complexity requires a declared profile)**.** *Without a declared weight profile, theorem-DNA complexity has vector meaning but no unique scalar order.*

Proof. Take two vectors that trade off statement and dependency mass: $(10, 1, 1, 1, 1, 1, 1, 1, 1)$ and $(1, 10, 1, 1, 1, 1, 1, 1, 1)$. A weight profile concentrated on the first coordinate orders them one way; a profile concentrated on the second coordinate orders them the other way. Therefore a published scalar score is auditable only together with its weight profile. □

9408.4 Mutation and classifier-level information

Definition 9408.21 (Theorem-DNA mutation)**.** *A theorem-DNA mutation is a passage*

$$D_R(T) \rightsquigarrow D'_{R'}(T)$$

between two theorem-DNA profiles for the same theorem candidate. It is single-locus when exactly one role component changes.

Definition 9408.22 (Complexity-reducing and discovery mutations)**.** *A mutation is complexity-reducing for* $\mathbf{w}$ *when*

$$\mathsf{Score}_{\mathbf{w}}(R', D') < \mathsf{Score}_{\mathbf{w}}(R, D).$$

It is a discovery mutation only when it changes the theorem classifier phenotype through a certified classifier-state shift with a naming certificate and a ledger witness.

Theorem 9408.23 (Proof shortening need not be discovery)**.** *A mutation that only changes the proof locus, while leaving statement, dependencies, certificates, ledger, status, canonical site, closing seal, and phenotype fixed, need not be a discovery mutation.*

Proof. The mutation may reduce the proof mass and hence the weighted score. However, the certificate, ledger, and theorem phenotype are unchanged, so no classifier-state shift has been supplied. It is a proof-level optimization, not a theorem-classifier discovery. □

Theorem 9408.24 (Complexity reduction and classifier discovery are independent)**.** *Theorem-DNA complexity reduction and classifier-level discovery are independent coordinates.*

Proof. A proof-locus shortening with fixed phenotype can reduce score while keeping the classifier unchanged. Conversely, a mutation that introduces a new certificate, ledger, or classifier phenotype can increase certificate and ledger mass while changing the classifier. Thus score decrease does not imply discovery, and discovery need not immediately decrease score. □

Principle 9408.25 (Theorem-DNA representation)**.** *A theorem is not merely a statement; it is a role-complete event-flow profile. Theorem complexity is not code length; it is weighted locus complexity. A shorter proof can reduce score, while a new theory changes the classifier phenotype.*

*Concretized in §**??**:* Theorem-flow recognition packets realize the eight-role theorem-DNA profile as finite GroundCompiler NameCert data.

Concretized in §827: GroundCompiler audit packets provide the event-flow, recognizer, code, proof-boundary, classifier, and ledger rows consumed by theorem-DNA scoring.

Concretized in §7061: The theorem-DNA complexity-scoring packet realizes role-locus, mass-vector, weighted-score, refusal-boundary, transport, replay, provenance, and NameCert rows as finite data.

Chapter 9409

Theorem-Genome Evolution: Mutation, Recombination, Selection, and Discovery

chapter 9366, chapter 9408, and chapter 9460 isolate three audit coordinates: classifier-level novelty, eight-locus theorem-DNA, and public complexity scoring. This chapter packages those coordinates as a single evolutionary reading of theorem flows. The term genome names the whole recognized theorem profile; it does not add biological assumptions.

9409.1 Theorem genomes

Definition 9409.1 (Theorem genome)**.** *Let T be a theorem candidate flow, R a generated theorem recognizer, and $D_R(T)$ an eight-locus theorem-DNA packet. If*

$$\mathsf{SoundTheoremFlow}(R, T)$$

holds, then

$$\Gamma = (T, R, D_R(T))$$

is a theorem genome. Its loci are statement, dependency, proof, certificate, ledger, status, canonical site, and closing seal.

Definition 9409.2 (Genome phenotype)**.** *Given a theorem classifier relation Q on theorem candidate flows, genomes $\Gamma_T = (T, R_T, D_T)$ and $\Gamma_U = (U, R_U, D_U)$ have the same phenotype when $Q(T, U)$ holds. This relation is written $\Gamma_T \sim_Q \Gamma_U$.*

Theorem 9409.3 (Statement locus does not determine genome)**.** *The statement locus does not determine the theorem genome.*

Proof. The GroundCompiler theorem `same_statement_multiple_flows` provides one statement subflow that appears as the statement role of two distinct theorem candidate flows. Since theorem genomes include the candidate flow, recognizer, and complete theorem-DNA packet, the statement locus alone cannot determine the genome. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.same statement multiple flows`

Theorem 9409.4 (Code does not imply genome)**.** *Legal theorem code does not by itself supply a theorem genome.*

Proof. The GroundCompiler theorem `theorem_code_is_not_proof` exhibits legal theorem code decoding to a candidate that has no proof-sound theorem recognition for any recognizer. A theorem genome requires sound theorem-flow recognition, hence proof soundness. Code is a flow address, not a genome. □

↪variant BEDC.GroundCompiler.TheoremGenerated.theorem_code_is_not_proof

9409.2 Complexity space

Definition 9409.5 (Genome complexity vector)**.** *For a genome* $\Gamma = (T, R, D_R(T))$*, define*

$$\mathbf{C}(\Gamma) = (c_s, c_d, c_p, c_c, c_l, c_t, c_a, c_z, c_R) \in \mathbb{N}^9$$

by taking the first eight coordinates to be the event-flow masses of the statement, dependency, proof, certificate, ledger, status, canonical-site, and closing-seal loci, and the last coordinate to be the recognizer mass $|R|_{\mathsf{flow}}$*.*

Definition 9409.6 (Genome score)**.** *For a public weight profile*

$$\mathbf{w} = (w_s, w_d, w_p, w_c, w_l, w_t, w_a, w_z, w_R) \in \mathbb{N}^9,$$

the genome score is

$$\mathsf{Score}_{\mathbf{w}}(\Gamma) = \mathbf{w} \cdot \mathbf{C}(\Gamma).$$

Definition 9409.7 (Phenotype complexity)**.** *For a phenotype class* $[\Gamma]_Q$*, define*

$$\mathsf{PC}_{\mathbf{w},Q}([\Gamma]_Q) = \min\{\mathsf{Score}_{\mathbf{w}}(\Gamma') \mid \Gamma' \sim_Q \Gamma\}.$$

This takes the least scored representative in a theorem classifier class.

9409.3 Mutation and discovery

Definition 9409.8 (Genome mutation)**.** *A genome mutation is a transition*

$$\Gamma \rightsquigarrow \Gamma'$$

between theorem genomes. It is single-locus when exactly one theorem locus is changed.

Definition 9409.9 (Mutation classes)**.** *Relative to* Q *and* $\mathbf{w}$*, a mutation is:*

$$\begin{gathered}\textit{silent if } \Gamma \sim_Q \Gamma' \textit{ and } \mathsf{Score}_{\mathbf{w}}(\Gamma) = \mathsf{Score}_{\mathbf{w}}(\Gamma'),\\ \textit{compressive if } \Gamma \sim_Q \Gamma' \textit{ and } \mathsf{Score}_{\mathbf{w}}(\Gamma') < \mathsf{Score}_{\mathbf{w}}(\Gamma),\\ \textit{drifting if } \Gamma \sim_Q \Gamma' \textit{ and } \mathsf{Score}_{\mathbf{w}}(\Gamma') > \mathsf{Score}_{\mathbf{w}}(\Gamma),\\ \textit{classifier-shifting if } \neg(\Gamma \sim_Q \Gamma').\end{gathered}$$

A classifier-shifting mutation is a discovery mutation only when it induces a different certified classifier state carrying a semantic naming certificate and a ledger witness.

Theorem 9409.10 (Complexity reduction does not imply discovery)**.** *There are mutations that reduce genome score without being discovery mutations.*

Proof. Replace only the proof locus by a shorter proof locus, leaving statement, dependencies, certificates, ledger, status, canonical site, closing seal, and phenotype fixed. With positive proof weight, the score decreases. Since no classifier phenotype changes and no different certified classifier state is introduced, the mutation is compressive rather than discovery. □

Theorem 9409.11 (Discovery may increase complexity)**.** *A discovery mutation can increase genome score.*

Proof. A classifier-shifting mutation may introduce larger certificate, ledger, closing-seal, or recognizer loci. If the corresponding weights are positive, the score may increase even though the mutation changes phenotype and carries the certificate and ledger data required for discovery. □

Principle 9409.12 (No free discovery)**.** *A transition that changes only presentation, naming, or proof length, while leaving classifier phenotype, semantic naming certificate, and ledger witness unchanged, is not a discovery mutation.*

9409.4 Recombination and lineage

Definition 9409.13 (Genome recombination)**.** *Two genomes are locus-compatible along a set of loci A when replacing the loci of one genome by the corresponding loci of the other still yields some R' and T' satisfying* $\mathsf{SoundTheoremFlow}(R', T')$*. The resulting transition is genome recombination.*

Theorem 9409.14 (Recombination preserves theoremhood under soundness)**.** *If a recombination result satisfies* $\mathsf{SoundTheoremFlow}(R', T')$*, then it is a theorem genome.*

Proof. This is immediate from Definition 9409.1: theorem genomehood is the triple of theorem candidate, recognizer, theorem-DNA packet, and sound theorem-flow recognition. □

Corollary 9409.15 (Recombination is not free)**.** *Two sound genomes do not automatically produce a sound genome under arbitrary locus exchange.*

Proof. Soundness is a relation among one recognizer, one theorem candidate flow, and all eight loci. Replacing proof, certificate, or ledger loci can break proof soundness, certificate soundness, or ledger soundness. The recombined profile therefore needs its own soundness witness. □

Definition 9409.16 (Genome lineage)**.** *A genome lineage is a finite chain*

$$\Gamma_0 \rightsquigarrow \Gamma_1 \rightsquigarrow \cdots \rightsquigarrow \Gamma_n$$

whose steps are mutations or recombinations. Its complexity profile is $L_{\mathbf{w}}(i) = \mathsf{Score}_{\mathbf{w}}(\Gamma_i)$*, and its classifier profile is the sequence of phenotypes* $[\Gamma_i]_{Q_i}$*.*

Theorem 9409.17 (A lineage can improve without discovering)**.** *There are lineages whose scores strictly decrease while their classifier profiles remain constant.*

Proof. Take a sequence of proof-compression mutations. Each step lowers proof-locus mass and leaves the other loci and the classifier phenotype fixed. □

Theorem 9409.18 (A lineage can discover before it compresses)**.** *There are lineages in which a classifier shift first increases score and later compression lowers score within the changed phenotype.*

Proof. The first step introduces a different classifier phenotype with additional certificate, ledger, seal, and recognizer mass. The second step optimizes a proof or ledger locus under that changed phenotype. Thus discovery can precede compression. □

9409.5 Selection and potential

Definition 9409.19 (Selection functional)**.** *For a population of theorem genomes, a selection functional has the form*

$$\mathsf{Sel}_{\mathbf{w}}(\Gamma) = -\mathsf{Score}_{\mathbf{w}}(\Gamma) + \lambda\,\mathsf{Novelty}(\Gamma) + \eta\,\mathsf{Reuse}(\Gamma) + \theta\,\mathsf{Unification}(\Gamma).$$

It values low score, classifier novelty, reusable proof/certificate/ledger material, and unification of prior phenotypes.

Theorem 9409.20 (Shortest theorem need not be research-favorable)**.** *The least-scored genome in a population need not maximize a selection functional.*

Proof. A low-score genome can have low novelty, reuse, and unification. A higher-score genome can dominate the selection value when its novelty, reuse, and unification terms are sufficiently large. □

Definition 9409.21 (Discovery potential)**.** *The discovery potential of a genome is an audit functional*

$$\mathsf{DP}(\Gamma) = \alpha\,\mathsf{Unification}(\Gamma) + \beta\,\mathsf{Reuse}(\Gamma) + \gamma\,\mathsf{Transfer}(\Gamma) - \delta\,\mathsf{LedgerCost}(\Gamma).$$

Theorem 9409.22 (High potential can coexist with high complexity)**.** *A genome can have high score and high discovery potential at the same time.*

Proof. Large certificates, ledgers, and recognizers increase score. The same loci can also support unification, reuse, and transfer across domains. If these positive terms dominate ledger cost in the potential functional, discovery potential is high despite high score. □

9409.6 Gate hierarchy

Definition 9409.23 (Genome gate hierarchy)**.** *The theorem-genome audit hierarchy is:*

$$\mathsf{BHistCarrier} \subset \mathsf{ChapterTasteGate} \subset \mathsf{FieldFaithful} \subset \mathsf{TheoremDNA} \subset \mathsf{ScoredDiscoveryGate}.$$

The first layer gives display and readback shape; the second gives round-trip and layer separation; the third prevents field erasure; the fourth requires anchored sound theorem-DNA; the fifth requires public score certificates and discovery claims to carry classifier shift, semantic naming certificate, and ledger witness.

Compiler-closure audit is not another theorem-genome locus in this hierarchy. It is the later audit layer that asks whether a genome flow expresses a compiler phenotype and whether that phenotype feeds back into a later accepted genome flow; see chapter 9411.

Theorem 9409.24 (The gate hierarchy is strict as an audit hierarchy)**.** *Each higher layer includes the obligations of the lower layers, but the reverse implications are not available in general.*

Proof. A display need not round-trip. A round-tripping display need not reflect all declared fields. A field-faithful carrier need not carry theorem-DNA. A theorem-DNA carrier need not publish a score certificate. A scored carrier need not justify a discovery claim without classifier shift, semantic naming certificate, and ledger witness. □

Theorem 9409.25 (Theorem-genome evolution theorem)**.** *The evolution of theorem-like objects in BEDC has four independent audit coordinates: record growth, genome complexity, classifier phenotype, and ledgered discovery.*

Proof. Record growth can occur by continuation without changing the genome-level classifier. Genome complexity can decrease by proof compression while phenotype is fixed. Phenotype can change while complexity increases because new certificate and ledger loci are introduced. Discovery is the strongest coordinate: it requires classifier shift together with semantic naming certificate and ledger witness. These four observations separate the audit coordinates. □

Principle 9409.26 (Theorem-genome slogan)**.** *A theorem is a genome, not a sentence. A proof is one locus, not the whole theorem. Compression improves a genome; discovery changes its classifier phenotype. No discovery is free: every classifier shift must pay in* NameCert *and ledger.*

*Concretized in §**??**:* Theorem-flow recognition packets supply the role-complete theorem-genome carrier.

Concretized in §827: GroundCompiler audit packets supply the code/proof, classifier quotient, recognizer, and ledger boundaries consumed by theorem-genome evolution.

Concretized in §6358: The theorem-genome evolution vision is realized as finite NameCert data over theorem-flow, recognizer, theorem-DNA, classifier-shift, mutation, selection, transport, provenance, and local naming rows.

Chapter 9410

Automated Theory-Discovery Pipeline Calculus

chapter 9366, chapter 9408, chapter 9460, and chapter 9409 separate four audit coordinates: record growth, role-complete theorem genomes, public complexity scoring, and classifier-level change. This chapter packages those coordinates as a calculus for automated theory discovery. The purpose is not to let an automation label its own output as discovery, but to specify the certificate conditions under which a pipeline output may be admitted as a discovery claim.

9410.1 Pipeline states

Definition 9410.1 (Certified theory state)**.** *A certified theory state is a tuple*

$$\mathcal{T} = (S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}}, C_{\mathcal{T}}, N_{\mathcal{T}})$$

where

$$S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}} : \mathsf{Hist} \to \mathsf{Prop}, \qquad C_{\mathcal{T}} : \mathsf{Hist} \to \mathsf{Hist} \to \mathsf{Prop},$$

and

$$N_{\mathcal{T}} : \mathsf{SemanticNameCert}(S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}}, C_{\mathcal{T}}).$$

The five fields record source, pattern, ledger, classifier, and semantic naming certificate.

Definition 9410.2 (Automation record)**.** *An automation record is a finite family R of generated material. Its entries may be event flows, theorem candidates, theorem genomes, candidate certificates, ledger entries, score certificates, or lineage edges. The record is an audit surface; it is not itself a classifier state.*

Definition 9410.3 (Pipeline state)**.** *A pipeline state is a tuple*

$$\mathcal{P} = (\mathcal{T}, R, \Gamma, \Lambda, \mathbf{w}, B)$$

where $\mathcal{T}$ is a certified theory state, R is an automation record, Γ is a population of theorem genomes, Λ is a lineage ledger, $\mathbf{w} \in \mathbb{N}^9$ is a public theorem-genome weight profile, and $B : \Gamma \to \mathbb{N}$ is a genome-budget function.

Definition 9410.4 (Genome extractor)**.** *A genome extractor is a partial operation*

$$\mathsf{Extract} : \mathsf{EventFlow} \rightharpoonup \Gamma$$

whose value, when defined on a theorem candidate flow T, is a theorem genome $(T, R, D_R(T))$ with all eight theorem-DNA loci. A theorem candidate is DNA-complete when $\mathsf{Extract}(T)$ is defined.

Definition 9410.5 (Score certificate)**.** *For a genome Γ in a pipeline state, a score certificate consists of the public weight profile $\mathbf{w}$, the budget value $B(\Gamma)$, and the inequality*

$$\mathsf{Score}_{\mathbf{w}}(\Gamma) \leq B(\Gamma).$$

It certifies a visible upper bound, not minimality.

9410.2 Transition classes

Definition 9410.6 (Pipeline transition)**.** *A pipeline transition is a change*

$$\mathcal{P} \longrightarrow \mathcal{P}'$$

between states

$$\mathcal{P} = (\mathcal{T}, R, \Gamma, \Lambda, \mathbf{w}, B), \qquad \mathcal{P}' = (\mathcal{T}', R', \Gamma', \Lambda', \mathbf{w}', B').$$

Unless the transition explicitly changes the scoring protocol, the weight profile is held fixed.

Definition 9410.7 (Mechanical expansion)**.** *A transition is a mechanical expansion when $\mathcal{T}' = \mathcal{T}$ and R' is obtained from R by adding material generated under the same classifier state. It may extend the record, but it does not change the certified classifier.*

Definition 9410.8 (Genome compression)**.** *A transition is a genome compression when $\mathcal{T}' = \mathcal{T}$ and some genome mutation satisfies*

$$\mathsf{Score}_{\mathbf{w}}(\Gamma') < \mathsf{Score}_{\mathbf{w}}(\Gamma)$$

while the theorem classifier phenotype is unchanged.

Definition 9410.9 (Genome recombination)**.** *A genome recombination combines loci from parent genomes $\Gamma_1, \ldots, \Gamma_k$ and yields a child genome Γ' only when the child again satisfies theorem-genome soundness. Locus exchange without a soundness witness is only a candidate recombination.*

Definition 9410.10 (Classifier-shift transition)**.** *A transition is a classifier-shift transition when*

$$\neg(\mathcal{T} \equiv_{\mathsf{cls}} \mathcal{T}')$$

holds on the common source carrier. It changes the classifier coordinate, regardless of whether the record length or genome score decreases.

Definition 9410.11 (Discovery transition)**.** *A classifier-shift transition is a discovery transition when the target theory state is certified and ledgered:*

$$\mathcal{T}' = (S', P', L', C', N'), \qquad N' : \mathsf{SemanticNameCert}(S', P', L', C'),$$

and

$$\exists h : \mathsf{Hist},\ L'(h).$$

Definition 9410.12 (Positive discovery transition)**.** *Relative to an external cost observer μ and transition cost κ, a discovery transition is positive when some finite family X satisfies*

$$\mu_{\mathcal{T}'}(X) + \kappa(\mathcal{T} \to \mathcal{T}') < \mu_{\mathcal{T}}(X).$$

The inequality charges the certificate, ledger, recognizer, and seal costs to the transition rather than hiding them in prose.

9410.3 Discovery claims

Definition 9410.13 (Discovery claim)**.** *A discovery claim is an audit packet*

$$\mathsf{Claim} = (K, \Gamma, \mathcal{T}, \mathcal{T}', d, N', L', \mathbf{w}, B, \Lambda, \mathcal{B})$$

where K is a claim kind, Γ is the principal genome, $\mathcal{T}, \mathcal{T}'$ are source and target theory states, $d \in \{0, 1\}$ is the classifier-change bit, N' and L' are the target certificate and ledger policy, $\mathbf{w}$ and B are the scoring protocol, Λ is the lineage ledger, and $\mathcal{B}$ is the explicit not-claimed boundary.

Definition 9410.14 (Claim kinds)**.** *The claim kinds are*

$$\mathsf{mechanical},\quad \mathsf{compression},\quad \mathsf{recombination},\quad \mathsf{classifierShift},\quad \mathsf{discovery},\quad \mathsf{positiveDiscovery}.$$

They respectively name record extension, score reduction without phenotype change, locus recombination, classifier change, ledgered certified classifier change, and ledgered certified classifier change with net compression gain.

Definition 9410.15 (Well-formed claim)**.** *A claim is well formed when its kind matches its evidence: mechanical claims do not assert classifier change; compression claims carry a score comparison under a public weight profile and have $d = 0$; classifier-shift claims have $d = 1$; discovery claims have $d = 1$, a target semantic naming certificate, and a ledger witness; positive-discovery claims additionally carry the net compression inequality. Any theorem-like principal object must be DNA-complete, and any scored claim must expose its weight profile.*

Definition 9410.16 (Discovery gate)**.** *The predicate* $\mathsf{DiscoveryGate}(\mathsf{Claim})$ *holds when the claim is well formed, its principal theorem-like object is a DNA-complete genome, it has a score certificate whenever it is scored, its discovery assertions carry semantic naming certificate and ledger witnesses, its positive-discovery assertions carry the net cost inequality, its lineage ledger records the parents and transition kind, its not-claimed boundary is explicit, and its support is formal or certificate-level rather than prose-only.*

Definition 9410.17 (Gate demotion)**.** *Gate demotion maps an over-strong claim kind to the strongest kind supported by the available evidence. A positive-discovery claim without the cost inequality is treated as discovery at most; a discovery claim without certificate or ledger is treated as classifier-shift at most; a classifier-shift claim with $d = 0$ is treated as compression or mechanical; an unscored compression claim is treated as an unscored mutation; a DNA-incomplete theorem-like object is treated as a theorem candidate.*

9410.4 Gate theorems

Theorem 9410.18 (DNA completeness is a gate obligation)**.** *A theorem candidate that is not DNA-complete cannot be the principal theorem genome of a gate-passed discovery claim.*

Proof. Definition 9410.16 requires the principal theorem-like object to be a DNA-complete genome. If extraction is undefined, the object does not supply the eight-locus theorem-DNA profile required by the gate. □

Corollary 9410.19 (Statement-only and code-only candidates are insufficient)**.** *A candidate supplying only a statement locus, or only legal theorem code, does not pass the discovery gate as a theorem genome.*

Proof. Both cases lack the complete theorem-genome extraction required by Definition 9410.18. □

Theorem 9410.20 (Mechanical expansion is not discovery)**.** *Mechanical expansion does not imply discovery.*

Proof. For mechanical expansion, $\mathcal{T}' = \mathcal{T}$. Hence the classifier states are classifier-equivalent and the classifier-change bit is 0. Discovery requires a classifier-shift transition and therefore a classifier-change bit of 1. □

Theorem 9410.21 (Genome compression is not discovery by itself)**.** *Genome compression without classifier change is not discovery.*

Proof. Compression lowers $\mathsf{Score_w}$ under the same classifier phenotype. If the classifier-change bit is 0, the transition violates the classifier-shift obligation in Definition 9410.11. □

Theorem 9410.22 (Discovery requires certificate and ledger)**.** *If a transition is a discovery transition with target state $\mathcal{T}' = (S', P', L', C', N')$, then*

$$N' : \mathsf{SemanticNameCert}(S', P', L', C')$$

and

$$\exists h : \mathsf{Hist},\ L'(h).$$

Proof. This is exactly the target-state obligation in Definition 9410.11. The semantic certificate supplies the naming certificate, and the ledger witness is part of the discovery condition rather than an optional annotation. □

Theorem 9410.23 (Positive discovery implies discovery)**.** *Every positive discovery transition is a discovery transition.*

Proof. Definition 9410.12 defines positive discovery by adding a net compression inequality to a discovery transition. □

Theorem 9410.24 (Scored claims require public weights)**.** *Any claim asserting a genome score, a score reduction, or a positive-discovery cost inequality must expose the weight profile used for the comparison.*

Proof. The score is the dot product of the public weight profile with the genome complexity vector. Without the weight profile, the scalar has no determinate audit meaning, and score comparisons cannot be reproduced. □

Theorem 9410.25 (Discovery gate is kind-sound)**.** *If* $\mathsf{DiscoveryGate}(\mathsf{Claim})$ *holds, then the obligations attached to the claim kind hold.*

Proof. The discovery gate includes well-formedness, and well-formedness assigns each claim kind exactly its evidence obligations. Thus a gate-passed compression claim has public score comparison and no classifier change; a classifier-shift claim has classifier change; a discovery claim has classifier change, semantic naming certificate, and ledger witness; and a positive-discovery claim additionally has the net cost inequality. □

Theorem 9410.26 (Demotion preserves audit soundness)**.** *After gate demotion, the resulting claim kind is no stronger than the evidence actually supplied by the claim.*

Proof. The clauses of Definition 9410.17 remove exactly the unsupported obligation: cost inequality for positive discovery, certificate or ledger for discovery, classifier change for classifier-shift, public score for compression, and theorem-DNA completeness for theorem-genome claims. The resulting label therefore does not assert evidence absent from the packet. □

9410.5 Lineage and selection

Definition 9410.27 (Lineage ledger)**.** *A lineage ledger is a directed acyclic graph $\Lambda = (V, E)$ whose vertices are theorem genomes and whose edges are labelled by transition kind: mechanical, compression, recombination, classifier-shift, discovery, or positive-discovery.*

Definition 9410.28 (Lineage-consistent claim)**.** *A claim is lineage-consistent when its parent genomes, child genome, and transition kind are recorded in Λ and the resulting ledger remains acyclic.*

Theorem 9410.29 (Accepted claims form an auditable lineage DAG)**.** *If every accepted claim is lineage-consistent, the accepted theorem genomes form an auditable directed acyclic lineage graph.*

Proof. Each accepted claim contributes labelled parent-to-child edges. By lineage-consistency, adding those edges does not introduce a cycle. Iterating over accepted claims preserves acyclicity. □

Definition 9410.30 (Discovery potential)**.** *For selection among candidate genomes, a discovery-potential functional has the form*

$$\mathsf{DP}(\Gamma) = \alpha\,\mathsf{Cls}(\Gamma) + \beta\,\mathsf{Unification}(\Gamma) + \gamma\,\mathsf{Reuse}(\Gamma) + \delta\,\mathsf{Transfer}(\Gamma) - \lambda\,\mathsf{Score}_{\mathbf{w}}(\Gamma) - \rho\,\mathsf{LedgerRisk}(\Gamma).$$

Here Cls *records classifier change, while the remaining positive terms record unification, reuse, and transfer. The negative terms charge complexity and ledger risk.*

Theorem 9410.31 (Lowest score need not be the best discovery target)**.** *A genome with lower score can have lower discovery potential than a genome with higher score.*

Proof. Take one genome with low score and zero classifier change, unification, reuse, and transfer. Take another with higher score, classifier change, high unification, high reuse, and controlled ledger risk. For sufficiently large positive coefficients on the structural terms, the second genome has greater discovery potential despite its higher score. □

Theorem 9410.32 (Classifier change is favoured when other terms are fixed)**.** *If a selection operator maximizes* DP *and $\alpha > 0$, then among candidates equal in all other coordinates it favours the candidate with classifier change.*

Proof. The difference in discovery potential between the classifier-changing candidate and the fixed-classifier candidate is exactly α, which is positive. □

9410.6 Pipeline adequacy

Definition 9410.33 (Unpaid discovery claim)**.** *An unpaid discovery claim is a claim that asserts discovery while lacking at least one of classifier change, semantic naming certificate, ledger witness, theorem genome, score or cost account, lineage witness, or not-claimed boundary.*

Theorem 9410.34 (No unpaid discovery passes the gate)**.** *No unpaid discovery claim passes* DiscoveryGate*.*

Proof. The discovery gate requires all of the obligations listed in Definition 9410.33. An unpaid claim lacks at least one of them, so it violates the gate. □

Theorem 9410.35 (Pipeline safety)**.** *If an automated pipeline accepts discovery and positive-discovery claims only after* DiscoveryGate *and records accepted claims in a lineage-consistent ledger, then every accepted discovery has classifier change, semantic naming certificate, ledger witness, DNA-complete theorem genome, score certificate, lineage witness, and an explicit not-claimed boundary.*

Proof. The gate supplies well-formedness, DNA completeness, score certificate, certificate and ledger obligations for discovery claims, the net inequality for positive-discovery claims, and the not-claimed boundary. The acceptance rule adds lineage consistency. Combining these clauses gives the listed obligations. □

Theorem 9410.36 (Automated theory-discovery pipeline theorem)**.** *An automated theory-discovery pipeline that extracts theorem genomes, exposes public scoring weights, demotes unsupported claim kinds, gates discovery claims, and records accepted claims in a lineage ledger distinguishes record growth, score reduction, recombination, classifier change, and ledgered discovery.*

Proof. Record growth is covered by mechanical expansion and does not imply discovery by Definition 9410.20. Score reduction is covered by genome compression and does not imply discovery by Definition 9410.21. Recombination requires its own soundness witness by Definition 9410.9. Classifier change is a separate transition class by Definition 9410.10. Ledgered discovery is the stronger gate-passed condition supplied by Definition 9410.35. □

Principle 9410.37 (Automated discovery principle)**.** *Automation explores genome space; the gate decides claim strength. More output is record growth. A shorter proof is compression. A recombined theorem profile is a candidate until soundness is witnessed. Discovery requires classifier change paid for by semantic naming certificate, ledger, score account, and lineage.*

Principle 9410.38 (Compiler-closure placement)**.** *Compiler-closure audit, as isolated in chapter 9411, sits after theorem-genome extraction and scoring. It does not merely ask whether a candidate theorem genome is role-complete; it asks whether a genome flow certifiably expresses a recognized compiler phenotype whose ledgered feedback exports a later accepted genome flow. Discovery self-compilation then adds classifier shift, semantic naming certificate, and ledger witness to that loop.*

Concretized in §5914: Discovery pipeline calculus is realized as finite theory-state, automation-record, genome, score, lineage, classifier-shift, and NameCert rows.

Concretized in §6931: Pipeline discovery classifiers realize the positive classifier-shift boundary as finite before/after classifier, public weight-profile, theorem-genome, ledger, TasteGate, transport, provenance, and NameCert rows.

Concretized in §7617: Candidate frontier queues realize the automated theory-discovery calculus as finite source-state, classifier-shift, score-budget, queue, ledger, replay, provenance, and NameCert rows.

Chapter 9411

Compiler-Closure Self-Compilation

chapter 9408, chapter 9409, and chapter 9410 make theorem recognition role-complete: a theorem-like flow is read through statement, dependency, proof, certificate, ledger, status, canonical-site, and closing-seal loci. This chapter records the next audit layer. A self-compiling BEDC system must also account for the compiler phenotypes expressed by genome flows and for the certified feedback by which those phenotypes regulate the next generation of theorem, certificate, gate, and closure flows.

The governing schematic is

$$D_t \longrightarrow C_t \longrightarrow P_t \longrightarrow D_{t+1}.$$

Here D_t is the current genome flow, C_t is the compiler or recognizer flow expressed by it, P_t is the behavior phenotype carried by that compiler, and D_{t+1} is the accepted genome flow exported after gate, ledger, NameCert, and closure-certificate checks. The point is not unrestricted reflection. The point is eventized, certified, ledgered self-compilation. This is a paper-side audit contract: it fixes the evidence shape a later pipeline can read, without asserting that every field is already implemented as a closure-status field or extraction script.

9411.1 From theorem DNA to compiler closure

Definition 9411.1 (Genome flow)**.** *A genome flow is an event flow D : EventFlow used as an auditable theory-state record. Its visible loci may include theorem-DNA packets, closure-status records, naming certificates, ledgers, gate data, score certificates, or lineage edges.*

Principle 9411.2 (One-directional recognition boundary)**.** *Theorem-DNA recognition reads a theorem candidate into role-complete loci, but it does not by itself express a compiler, recognizer, gate, or classifier that regulates later genome flows.*

Theorem 9411.3 (Theorem DNA alone is not self-compilation)**.** *Let T be a theorem candidate flow. If T has role-complete theorem-DNA but does not certifiably express a compiler phenotype, then there is no genome flow D' for which T forms a certified self-compilation loop into D'.*

Proof. A certified self-compilation loop, as defined in Definition 9411.16, includes certified expression of a compiler phenotype. The hypothesis denies such an expression witness for T. Therefore the loop data cannot be supplied. □

Corollary 9411.4 (Theorem-genome audit and compiler-closure audit are distinct)**.** *Theorem-genome audit asks whether a theorem-like flow is role-complete, scored, anchored, and ledgered. Compiler-closure audit asks whether such a genome flow expresses a compiler phenotype and carries ledgered feedback into a later accepted genome flow.*

Proof. The first audit is witnessed by theorem-DNA and theorem-genome gates. The second requires compiler expression, compiler behavior, feedback, and accepted export data. These are not part of theorem-DNA alone. □

9411.2 Compiler phenotype

Definition 9411.5 (Compiler phenotype)**.** *A compiler phenotype is a triple*

$$P = (C, \beta, \rho)$$

where $C : \mathsf{EventFlow}$ *is a compiler flow,*

$$\beta : \mathsf{EventFlow} \to \mathsf{EventFlow} \to \mathsf{Prop}$$

is its behavior relation, and ρ *witnesses* $\mathsf{RecognizedCompilerFlow}(C)$*.*

Definition 9411.6 (Compiler-genome loci)**.** *A compiler genome is a genome flow whose displayed loci include source, recognizer, compiler, gate, classifier, ledger, export, and feedback-seal components. These loci record what the compiler reads, how legal flows are recognized, how outputs are produced, how impostors are refused, how accepted objects are classified, how obligations are recorded, how results are exported, and how feedback is permitted.*

Theorem 9411.7 (Self-compilation requires compiler phenotype)**.** *If* $D \circlearrowright D'$ *is a certified self-compilation loop, then there exists a compiler phenotype* $P = (C, \beta, \rho)$ *such that* $\mathsf{RecognizedCompilerFlow}(C)$ *and* $\beta(D, D')$*.*

Proof. Unfold the certified self-compilation loop. It supplies a compiler phenotype P, certified expression from D to P, and feedback from P on D to D'. Unfolding the compiler phenotype gives the recognized compiler-flow witness for C. Unfolding the feedback witness gives $\beta(D, D')$. □

Corollary 9411.8 (No recognized compiler flow, no self-compilation)**.** *If no event flow is recognized as a compiler flow, then no certified self-compilation loop exists.*

Proof. This is the contrapositive of Definition 9411.7. □

9411.3 Certified expression

Definition 9411.9 (Expression witness)**.** *Let* D *be a genome flow and let* $P = (C, \beta, \rho)$ *be a compiler phenotype. An expression witness for* $D \Rightarrow P$ *is a ledger flow* L_{expr} *such that* C *is visible as a source subflow of* D*,* L_{expr} *is visible as a source subflow of* D*, and* L_{expr} *records the genome carrier, the expressed compiler flow, the visible expression route, the recognized-compiler witness, and the formal input surface used by the expression.*

Definition 9411.10 (Certified compiler expression)**.** *A genome flow D certifiably expresses a compiler phenotype P, written*

$$D \Rightarrow P,$$

when an expression witness exists and all formal compiler inputs used by the expression are raw events, event flows, recognized flows, or certified exports.

Theorem 9411.11 (Expression excludes hidden host state)**.** *A certified compiler expression cannot depend on host ASTs, host YAML, host manifests, host packages, host theorem identifiers, host object names, host type tags, host mode tags, host opcode tables, host arity tables, or host parser state as formal compiler inputs.*

Proof. BEDC's formal compiler-input relation admits raw events, event flows, recognized flows, and certified exports. The listed host structures are structural hidden inputs rather than formal compiler inputs. Certified expression requires its input route to stay inside the formal compiler-input surface, so those hidden host structures are excluded. □

Corollary 9411.12 (Expression is not host extraction)**.** *A host parser, manifest scanner, or theorem-name lookup may help locate a candidate during engineering, but it does not by itself witness $D \Rightarrow P$. The witness must be present as event-flow expression data.*

Proof. Use Definition 9411.11. Host parser state, manifests, and theorem identifiers are excluded from the formal input surface unless they have first been represented as accepted event-flow data. □

9411.4 Ledgered feedback

Definition 9411.13 (Feedback witness)**.** *Let $P = (C, \beta, \rho)$ be a compiler phenotype and let D, D' be genome flows. A feedback witness for*

$$P : D \rightsquigarrow D'$$

is a ledger flow L_{fb} such that $\beta(D, D')$ holds, L_{fb} is visible as a source subflow of D', and L_{fb} records the source genome carrier, the compiler phenotype, the behavior relation, the target genome carrier, the accepted-export route, the naming and closure-certificate obligations, and the boundary excluding unrestricted reflection.

Principle 9411.14 (No unledgered feedback)**.** *A compiler phenotype may regulate a later genome flow only through a displayed feedback ledger. Otherwise the transition is an external update, not BEDC self-compilation.*

Corollary 9411.15 (External file edits are not self-compilation)**.** *An external script that edits theorem text, closure status, or lineage data without presenting the update as event-flow feedback with a visible ledger does not establish certified self-compilation.*

Proof. Certified self-compilation requires the feedback witness of Definition 9411.13. A host-level edit without the visible feedback ledger fails that requirement. □

9411.5 Certified self-compilation loop

Definition 9411.16 (Certified self-compilation loop)**.** *For genome flows D and D', write*

$$D \circlearrowright D'$$

when there exists a compiler phenotype P such that $D \Rightarrow P$, $P : D \rightsquigarrow D'$, and $\mathsf{AcceptedObjectFlow}(D')$.

Definition 9411.17 (Strict and stationary self-compilation)**.** *A certified self-compilation loop is strict when $D \neq D'$. It is stationary when $D = D'$.*

Theorem 9411.18 (Certified self-compilation exports an accepted object)**.** *If $D \circlearrowright D'$, then* $\mathsf{AcceptedObjectFlow}(D')$.

Proof. This is one of the clauses in the definition of certified self-compilation. □

Corollary 9411.19 (The output is not just a later file state)**.** *If D' is not an accepted object flow, then a transition $D \to D'$ is not a certified self-compilation loop.*

Proof. Use Definition 9411.18. □

9411.6 Self-hosting is not self-compilation

Theorem 9411.20 (Self-hosting is weaker than self-compilation)**.** *A self-hosting target for a compiler flow does not by itself supply certified self-compilation.*

Proof. A self-hosting target states that a compiler flow is behaviorally equivalent to itself under some compiler behavior. Certified self-compilation also requires a genome flow, certified expression of a compiler phenotype by that genome, feedback from the phenotype into a later genome flow, and accepted export of the later genome flow. These data are not supplied by self-hosting alone. □

Principle 9411.21 (Compiler property versus genome-feedback property)**.** *Self-hosting is a compiler-layer property. Self-compilation is a genome–compiler–feedback–accepted-export property.*

9411.7 Discovery self-compilation

Definition 9411.22 (Discovery self-compilation)**.** *A certified self-compilation loop*

$$D \circlearrowright D'$$

is discovery self-compilation when the associated discovery audit packet has classifier-change bit $d = 1$ and the later genome flow carries a certified classifier state with semantic naming certificate and ledger witness.

Theorem 9411.23 (Self-compilation is not automatically discovery)**.** *A certified self-compilation loop whose associated classifier-change bit is $d = 0$ is not discovery self-compilation.*

Proof. Discovery self-compilation requires classifier-change bit $d = 1$ by definition. □

Theorem 9411.24 (Discovery self-compilation is a discovery transition)**.** *Every discovery self-compilation loop is a discovery transition.*

Proof. Discovery self-compilation includes certified self-compilation, classifier shift, semantic naming certificate, and ledger witness. These are exactly the obligations required for a discovery transition in the automated discovery calculus. □

9411.8 Compiler-closure gate contract

Definition 9411.25 (Compiler-closure audit packet)**.** *A compiler-closure audit packet is a tuple*

$$\mathsf{CCClaim} = (D, D', C, \beta, L_{\mathsf{expr}}, L_{\mathsf{fb}}, E, N_{\mathsf{name}}, N_{\mathsf{close}}, d, \mathcal{B})$$

where D is the source genome flow, D' is the target genome flow, C is the expressed compiler flow, β is the behavior relation, L_{expr} is the expression ledger, L_{fb} is the feedback ledger, E is the accepted-export witness for D', N_{name} and N_{close} are the naming and closure-certificate witnesses, d is the classifier-change bit, and $\mathcal{B}$ is the explicit not-claimed boundary.

Definition 9411.26 (Compiler-closure evidence obligations)**.** *A compiler-closure audit packet satisfies the evidence obligations when:*

1. *C is a recognized compiler flow.*
2. *β is supplied as a compiler behavior relation.*
3. *L_{expr} witnesses certified expression $D \Rightarrow P = (C, \beta, \rho)$.*
4. *L_{fb} witnesses ledgered feedback $P : D \rightsquigarrow D'$.*
5. *E, N_{name}, and N_{close} witness accepted export of the target genome as an accepted object flow.*
6. *$\mathcal{B}$ excludes hidden host state, host-executable identity, and unrestricted reflective meta-loops.*

If the packet asserts discovery self-compilation, it must also have $d = 1$, semantic naming certificate, and ledger witness in the sense of Definition 9410.11.

Definition 9411.27 (Free self-compilation claim)**.** *A free self-compilation claim is a claim that asserts $D \circlearrowright D'$ while omitting at least one of recognized compiler flow, compiler behavior, expression witness, feedback witness, accepted object flow, or feedback ledger.*

Definition 9411.28 (CompilerClosureGate)**.** *A claim passes* CompilerClosureGate *when it is represented by a compiler-closure audit packet satisfying Definition 9411.26. Where the source or target carrier is theorem-like, the relevant theorem-DNA scoring obligations must also hold. Where the claim asserts discovery self-compilation, the discovery obligations of* DiscoveryGate *must also hold.*

Definition 9411.29 (Compiler-closure demotion)**.** *Compiler-closure demotion maps an over-strong self-compilation claim to the strongest status supported by its evidence. Without a recognized compiler flow or compiler behavior, the packet is only a compiler candidate. Without an expression witness, it is not a self-compilation claim. Without a feedback ledger, it is a free claim. Without accepted export of D', it is only a candidate transition. Without $d = 1$ and semantic naming certificate, it may be self-compilation but not discovery self-compilation.*

Theorem 9411.30 (No free self-compilation passes)**.** *No free self-compilation claim passes* CompilerClosureGate*.*

Proof. The gate requires the evidence obligations of Definition 9411.26. A free claim omits at least one required item, so the gate obligation is not met. □

Theorem 9411.31 (Demotion never strengthens evidence)**.** *After compiler-closure demotion, the assigned status is no stronger than the evidence actually carried by the packet.*

Proof. Definition 9411.29 removes self-compilation strength when recognized compiler flow, behavior, expression, feedback ledger, or accepted export is missing, and removes discovery strength when classifier change, semantic naming, or ledger witness is missing. Therefore the resulting status does not assert evidence absent from the packet. □

Theorem 9411.32 (CompilerClosureGate strictly strengthens TDCSGate)**.** *For theorem-like carriers, passing* CompilerClosureGate *implies the relevant theorem-DNA scoring obligations, but theorem-DNA scoring obligations alone do not imply passing* CompilerClosureGate*.*

Proof. The compiler-closure gate includes the relevant theorem-DNA scoring obligations where the source or target carrier is theorem-like. It also requires compiler expression, behavior, feedback, accepted export, and a feedback ledger. These additional clauses are not consequences of theorem-DNA scoring alone. □

Theorem 9411.33 (DiscoveryGate alone is not compiler closure)**.** *A gate-passed discovery claim does not by itself establish compiler-closure self-compilation unless it also supplies compiler expression and ledgered feedback.*

Proof. DiscoveryGate checks classifier-change, semantic naming, ledger, score, lineage, and not-claimed obligations for discovery. Certified self-compilation additionally requires a recognized compiler phenotype, certified expression, and feedback from that phenotype into the target genome. Those data are not supplied by discovery evidence alone. □

9411.9 Pipeline adoption boundary

Principle 9411.34 (Compiler-closure adoption)**.** *The automated theory-discovery pipeline may treat compiler-closure audit as a post-theorem-genome layer:*

$$\textit{theorem-DNA extraction} \longrightarrow \textit{genome scoring} \longrightarrow \textit{TDCSGate} \longrightarrow \textit{compiler-closure audit} \longrightarrow \textit{discovery gate.}$$

The additional layer asks whether a genome flow expresses a compiler phenotype whose feedback produces a later accepted genome flow. It reads the compiler-closure audit packet of Definition 9411.25; it does not infer self-compilation from file edits, host execution, or theorem-name lookup.

Theorem 9411.35 (Compiler-closure self-compilation theorem)**.** *A BEDC transition $D \to D'$ can be accepted as certified self-compilation exactly when its compiler-closure audit packet satisfies the evidence obligations of Definition 9411.26. If the same packet also has classifier-change bit $d = 1$ with semantic naming certificate and ledger witness, it is discovery self-compilation.*

Proof. The forward direction unfolds certified self-compilation and places its recognized compiler flow, behavior relation, expression ledger, feedback ledger, accepted export, and boundary into the audit packet. The reverse direction reads those packet fields as certified expression and ledgered feedback and then applies the accepted-output clause of the loop definition. The classifier-change, semantic naming, and ledger witnesses supply the discovery strengthening. □

Principle 9411.36 (Compiler-closure slogan)**.** *Theorem DNA audits what a theorem flow contains. Compiler closure audits what a genome flow can express and how the expressed compiler phenotype feeds back into the next accepted genome flow.*

Concretized in §3963: PublicLocalReflectiveCompilationUp realizes the compiler-closure self-compilation audit surface as finite public/local compiler-event and MetaCIC closure rows.

Concretized in §6317: Compiler-closure self-compilation is realized as a finite NameCert packet for compiler phenotype, feedback, export, and ledger rows.

Concretized in §7024: SelfCompilationFeedbackUp realizes the compiler-closure feedback witness as a finite feedback-ledger carrier over compiler phenotype, EventFlow feedback, accepted export, refusal boundary, transport, replay, provenance, and local naming rows.

Chapter 9412

Pseudotheory Refutation Gate Map

This chapter records the present audit surface for pseudotheory refusal. It is not a carrier design and not a witness schema. Its purpose is to say where each refutable failure mode is checked, which checks block assertions, and why taste-sensitive stability remains outside automatic demotion.

9412.1 Coverage table

Class	Gate surface	Status	Code evidence
(a) synonym or reconstruction	Discovery integrity, structural-DNA payload equality, and the `discovery_gate_witnesses.json` registry.	Mechanical for asserted discovery.	`bedc_ci.py:9181-9430` computes structural-DNA relations and emits `structural_reconstruction_discovery_c` `bedc_ci.py:3432-3599` loads only kernel-grounded reconstruction witnesses; `bedc_ci.py:4464-4480` turns a witness hit into a failing `W` gate.
(b) padding or classifier-neutral surface	Classifier-bit demotion through the assertion gate and mechanical opt-out audit.	Mechanical for asserted discovery; survey rows are informational.	`bedc_ci.py:4300-4357` fails asserted `positiveDiscovery` when nontrivial classifier evidence is absent; `bedc_ci.py:4362-4385` fails when disagreement support is not checked; `bedc_ci.py:4401-4424` rejects smoke or duplicate-like evidence. `bedc_ci.py:10623-10631` requires a concrete invariant when the opt-out reason is classifier unchanged.
(c) axiom-crossing carrier or theorem	Lean kernel axiom scan and source-token axiom scan.	Mechanical.	`bedc_ci.py:4205-4259` fails asserted discovery targets with axiom, sorry, or forbidden transitive dependencies. `bedc_ci.py:12691-12919` runs `#print axioms` over public BEDC theorems and exits nonzero on forbidden dependencies.
(d) stability break needing taste	`ChapterTasteGate` admission and human judgment over value and stability.	Human-only; not an automatic demotion class.	`TasteGate.lean:45-67` defines the carrier display and readback obligations; `TasteGate.lean:81-98` derives conservativity and no-hidden-input. The paper account at `parts/visions/meta_foundations/theory` states why AI-proposed chapters need taste admission.
(e) unpaid discovery	Semantic naming, row, ledger, cost, debt, and scope-seal obligations for `PositiveDiscovery`.	Mechanical for asserted discovery; broad ledger surveys remain informational.	`DiscoveryCertificate.lean:126-156` requires before/after certified states, row evidence, classifier shift, debt, ledger membership, and scope seal. `bedc_ci.py:2774-2824`

9412.2 Stateful and stateless refusal

Only reconstruction needs a persistent witness. A reconstruction refusal compares a candidate against the corpus: the evidence is not contained in the candidate alone, because the refutation is equality of its structural-DNA payload with a prior payload. The registry therefore stores the target, prior, canonical payload, and grounding status. The loader accepts the row only after an independent structural-DNA grounding check, so the witness is a cache of a corpus relation, not an extra theory object.

Padding, axiom-crossing, and unpaid discovery are stateless refusals. Padding is recomputed from the claimed classifier shift and its checked disagreement support. Axiom-crossing is recomputed from source text and kernel axiom dependencies. Unpaid discovery is recomputed from the claim's own `PositiveDiscovery` fields: semantic naming certificate, row evidence, ledger membership, debt, and scope seal. Persisting these as another witness family would duplicate the existing gate surface without adding soundness.

9412.3 Prime asymmetry

BEDC can automatically refute some claims because a refutation may be a finite counter-reading: same structural payload as a prior, no classifier shift, forbidden axiom dependency, or missing certificate data. The opposite judgment is different. Saying that a chapter is worth admitting to the horizon is not only the absence of these finite failures; it is a taste judgment about stability, usefulness, and fit with the surrounding theory.

This is the prime asymmetry. Composite-like failures may be found by an audit gate. A positive value judgment is not created by the complement of those failures. `ChapterTasteGate` supplies the carrier display/readback discipline, but it does not replace human admission of taste-sensitive stability.

9412.4 Carrier-repair boundary

The refutation map is apophatic. It says why a claim may not be asserted as discovery. It does not say that the carrier must be replaced by a different carrier, and it does not turn every refusal into a constructive repair recipe. That boundary is essential: synonymy is evidence for demotion, not evidence that the target and prior must be recoded into different domains; padding is absence of paid classifier shift, not an invitation to alter the carrier until the score improves; taste-sensitive stability is an admission judgment, not an automatic rewrite target.

Thus the design rejects cataphatic carrier repair as the foundation of pseudotheory refusal. The gates remain refusal surfaces: structural-DNA collision, classifier-bit failure, axiom dependency, unpaid certificate data, and human-only taste admission.

Concretized in §6558: Finite pseudotheory refusal classes are realized as a $\mathsf{RefutationGate}^{\uparrow}$ NameCert packet.

Concretized in §6560: The pseudotheory-specific refusal gate is realized as finite structural-DNA, classifier-shift, axiom-scan, and certificate-debt NameCert data.

Concretized in §7280: Finite countermodel rows realize pseudotheory refusal as auditable NameCert data.

Concretized in §6547: finite confluence diamond ledger rows realize the audit-map methodology as BEDC NameCert data

Chapter 9413

Common Carriers and Reality Projections

The unified-reality reading does not treat algebra, geometry, analysis, number theory, spectral theory, and physical modelling as unrelated object domains. It reads them as projection surfaces of a common generating carrier, each surface equipped with its own admissible classifier, operation laws, transport policy, and ledger. This does not postulate a completed universe of mathematical objects. It fixes an auditable interface through which multiple mature readings can be compared without importing an unlicensed ambient ontology.

The guiding constraint is the ordinary BEDC naming discipline: a mature object is admitted only through source, pattern, classifier, stability, and ledger data. Thus $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, metric spaces, categories, spectra, physical-signature models, and analytic continuation interfaces are not primitive inputs. They are stable readout surfaces over generated behaviour. A unified object is therefore not a bare element of a universal set; it is a common carrier together with reality projections, classifier-respecting transport, and a boundary ledger refusing reverse readings that have not been certified.

9413.1 Reality Interfaces

Definition 9413.1 (Reality interface)**.** *A* reality interface *is a tuple*

$$\mathcal{R} := (R, \sim_R, \mathsf{Car}_R, \mathsf{Obs}_R, \mathsf{Law}_R, \mathsf{Led}_R)$$

where R is the carrier or carrier predicate for the interface, $\sim_R$ is its classifier, Car_R records admissible carrier conditions, Obs_R records the finite observations readable at that interface, Law_R records the operations and structural laws licensed inside the interface, and Led_R records provenance, stability, transport, and boundary information.

Definition 9413.2 (Stable reality interface)**.** *A reality interface $\mathcal{R}$ is* stable *when $\sim_R$ is reflexive, symmetric, and transitive on the admitted carrier, and every operation listed in Law_R respects $\sim_R$ in each argument and output coordinate on which it is read.*

Principle 9413.3 (Interface before domain)**.** *BEDC does not accept a domain name as a source-free primitive. A domain name may be consumed only after its carrier, classifier, admissible observations, operation laws, and ledger policy have been displayed as a reality interface.*

9413.2 Common Carriers

Definition 9413.4 (Common carrier)**.** *Let $(\mathcal{R}_i)_{i\in I}$ be a family of reality interfaces with carriers R_i and classifiers $\sim_{R_i}$. A* common carrier *for this family is data*

$$\mathcal{O} := (O, \sim_O, (\pi_i)_{i\in I}, \mathsf{Tr}, \mathsf{Led}, \mathsf{Bnd})$$

where O is the shared carrier, $\sim_O$ is its classifier, $\pi_i : O \to R_i$ are reality projections, Tr *records transport data between projection surfaces,* Led *records projection and compatibility provenance, and* Bnd *records refused reverse readings, collapses, external bridges, and unreviewed coercions.*

Definition 9413.5 (Classifier-respecting projection)**.** *A projection $\pi_i : O \to R_i$ from a common carrier is* classifier-respecting *when*

$$x \sim_O y \Longrightarrow \pi_i(x) \sim_{R_i} \pi_i(y)$$

for all admitted rows $x, y : O$.

Definition 9413.6 (Reality projection)**.** *If $\pi_i : O \to R_i$ is a projection from a common carrier, then $\pi_i(x)$ is the* reality readout *of x at the interface $\mathcal{R}_i$. For a predicate $P_i : R_i \to \mathsf{Prop}$, the composite claim $P_i(\pi_i(x))$ is a* projection property *of x at $\mathcal{R}_i$.*

Theorem 9413.7 (Projection preserves classified identity)**.** *Let $\mathcal{O}$ be a common carrier and let π_i be classifier-respecting. If $x \sim_O y$, then the two reality readouts are classified together:*

$$\pi_i(x) \sim_{R_i} \pi_i(y).$$

Proof. This is exactly the classifier-respect clause in Definition 9413.5. □

9413.3 Ledgered Transport

Definition 9413.8 (Reality transport)**.** *Given interfaces $\mathcal{R}_i$ and $\mathcal{R}_j$ over a common carrier $\mathcal{O}$, a* reality transport *from a predicate $P_i : R_i \to \mathsf{Prop}$ to a predicate $P_j : R_j \to \mathsf{Prop}$ is a ledgered rule*

$$\mathsf{Tr}_{ij}(P_i, P_j)$$

whose exported consequence is

$$\forall x : O, \quad P_i(\pi_i(x)) \Longrightarrow P_j(\pi_j(x)).$$

The transport row records the source predicate, target predicate, projection maps, classifier compatibility, stability basis, and boundary conditions under which the implication may be consumed.

Definition 9413.9 (Transport-equivalent projection properties)**.** *Projection properties P_i and P_j are $\mathcal{O}$-*transport-equivalent *when the ledger contains transport rows in both directions:*

$$\mathsf{Tr}_{ij}(P_i, P_j) \qquad \textit{and} \qquad \mathsf{Tr}_{ji}(P_j, P_i).$$

In that case one may write

$$P_i \equiv_{\mathcal{O}} P_j$$

as an abbreviation for the displayed bidirectional transport data, not as an unqualified equality of predicates in an ambient universe.

Theorem 9413.10 (Bidirectional transport gives projection equivalence)**.** *If P_i and P_j are $\mathcal{O}$-transport-equivalent, then for every $x : O$ the two projection properties imply each other:*

$$P_i(\pi_i(x)) \Longrightarrow P_j(\pi_j(x))$$

and

$$P_j(\pi_j(x)) \Longrightarrow P_i(\pi_i(x)).$$

Proof. Unpack Definition 9413.9. The first transport row is exactly the first displayed implication for each $x : O$, and the second transport row is exactly the reverse implication. □

Corollary 9413.11 (Projection equivalence is ledger relative)**.** *The notation $P_i \equiv_{\mathcal{O}} P_j$ exports no equality stronger than the two ledgered transport rows that define it.*

Proof. By Definition 9413.9, the notation is only an abbreviation for the two displayed transport rows. Therefore any consumer of the equivalence must read the corresponding source, classifier, stability, and boundary entries from the transport ledger. □

9413.4 Generalized Continuation

Definition 9413.12 (Generalized analytic continuation)**.** *Let $\mathcal{O}$ be a common carrier with projections $\pi_i : O \to R_i$ and $\pi_j : O \to R_j$. A* generalized analytic continuation *from $\mathcal{R}_i$ to $\mathcal{R}_j$ is a packet*

$$\mathsf{GenCont}_{i\to j} := (\mathcal{O}, \pi_i, \pi_j, \mathsf{Tr}_{ij}, \mathsf{Led}_{ij})$$

where Tr_{ij} transports certified properties from the i-interface to the j-interface and Led_{ij} records the source, classifier, stability, provenance, and boundary conditions of that transport. The packet is admissible only when it introduces no uncertified primitive and makes every hidden premise visible in the ledger.

Remark 9413.13 (Ordinary analytic continuation as a special case)**.** *Ordinary analytic continuation fits the schema when the interfaces are holomorphic-domain readouts, the shared carrier is the continuation packet, the projections read local holomorphic representatives, and the transport ledger is the overlap, identity-theorem, uniqueness, and branch-boundary record. The generalized form allows the same discipline to relate algebraic, geometric, spectral, measure-theoretic, arithmetic, analytic, and physical-signature readouts when the corresponding projection and transport rows are displayed.*

Theorem 9413.14 (Generalized continuation is transport, not creation)**.** *An admissible $\mathsf{GenCont}_{i\to j}$ packet cannot create a target property outside the displayed common carrier, projection maps, transport row, and ledger.*

Proof. By Definition 9413.12, the packet consists only of the common carrier, the two projections, the transport row, and the ledger. The admissibility condition forbids the introduction of an uncertified primitive. Therefore every exported target-side property must be read through the source-side certified property, the displayed projection maps, and the transport ledger. □

Principle 9413.15 (Boundary refusal)**.** *A projection cannot be reversed, collapsed into another projection, or exported across a bridge unless the relevant reverse map, collapse certificate, or bridge packet is present in the ledger. The common carrier licenses comparison and transport; it does not license unreviewed identification.*

9413.5 Noncollapse and Information Dimension

The boundary refusal principle has a sharper invariant form. A higher interface may have lower-level shadows, but the presence of those shadows does not by itself give a lossless reconstruction of the higher interface. This is the noncollapse discipline: projection may make an object visible at a lower layer, while the missing fibers, moduli, regularity rows, classifiers, and provenance remain part of the object being read.

The located-real surface is the guiding L10 instance. In Definition 1933.1, a $\mathsf{LocatedReal}^{\uparrow}$ packet is not a host real point. It is the finite route

$$L = (S, R, D, E, Q, H, C, P, N)$$

through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers, a located enclosure row, a terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local NameCert data. The carrier explicitly lacks Dedekind cuts, quotient real equality, host real equality, trichotomy, countable-choice selectors, compactness witnesses, metric-completion objects, and bridge credentials. Thus a rational or dyadic observation may be a shadow of a located real, but it is not the located real itself.

Definition 9413.16 (Information layer tower)**.** *An* information layer tower *is a sequence of carriers*

$$L_0, L_1, L_2, \ldots$$

together with finite readout families

$$\pi_k^{n+1,n} : L_{n+1} \to L_n, \qquad k : \mathsf{Nat}^{\uparrow}.$$

The carrier L_0 is the value-facing layer: typical examples include $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, finite histories, or other displayed finite observation surfaces. The carrier L_1 is the process-facing layer: typical rows contain schedules, streams, moduli, regularity witnesses, classifiers, and ledgers. A row of L_{n+1} may be observed through L_n at finite precision k, but that observation is only a shadow unless a reconstruction certificate is displayed.

Definition 9413.17 (Finite shadow)**.** *Let $x : L_{n+1}$ be a higher-layer row in an information layer tower. Its k-*shadow *on the lower layer is*

$$\mathrm{Sh}_k^{n+1,n}(x) := \pi_k^{n+1,n}(x).$$

For a process row whose observable component is a stream x_{seq} and whose precision policy is x_{mod}, the typical value-facing shadow has the form

$$\mathrm{Sh}_k(x) = x_{\mathsf{seq}}(x_{\mathsf{mod}}(k)).$$

The equation records the inspected value at the displayed precision; it does not identify the inspected value with the process row.

Definition 9413.18 (Degenerate embedding)**.** *Given a value-facing carrier L_0 and a process-facing carrier L_1, a* degenerate embedding *is a displayed map*

$$\eta : L_0 \to L_1$$

whose image consists of process rows with constant observations and trivial precision residue. For a rational-facing instance this has the schematic form

$$\eta(q)_{\mathsf{seq}}(n) = q, \qquad \eta(q)_{\mathsf{mod}}(k) = 0.$$

The map is part of the tower data only when its carrier, classifier, constant-observation law, precision policy, and ledger are displayed.

Definition 9413.19 (Local noncollapse)**.** *Let $x : L_1$ be a process-facing row and let $\eta : L_0 \to L_1$ be a displayed degenerate embedding. The row x is* locally noncollapsed over L_0 *when*

$$\mathsf{NonCollapse}_{L_0}(x) := \neg\exists q : L_0,\ x \sim_{L_1} \eta(q).$$

Thus x cannot be consumed as a lossless value-layer point without a classifier witness identifying it with a displayed degenerate process.

Definition 9413.20 (Structural noncollapse)**.** *Let L_1 be a process-facing carrier over a value-facing carrier L_0. The pair (L_1, L_0) satisfies* structural noncollapse *when there is no pair of displayed maps*

$$c : L_1 \to L_0, \qquad \rho : L_0 \to L_1$$

such that

$$\forall x : L_1, \quad \rho(c(x)) \sim_{L_1} x.$$

The map c would choose a lower-layer readout, and ρ would reinflate that readout into a process row. Structural noncollapse says that no such retraction is available from the displayed tower data.

Definition 9413.21 (Projection fiber)**.** *For a projection $\pi : L_1 \to L_0$ and a lower-layer row $q : L_0$, the* projection fiber *over q is*

$$\mathrm{Fib}_\pi(q) := \sum_{x:L_1} (\pi(x) \sim_{L_0} q).$$

The projection has a nontrivial fiber *when there are rows $x, y : L_1$ such that*

$$\pi(x) \sim_{L_0} \pi(y)$$

but

$$\neg(x \sim_{L_1} y).$$

Theorem 9413.22 (Nontrivial fibers obstruct collapse)**.** *If $\pi : L_1 \to L_0$ has a nontrivial fiber, then π cannot be the first component of a classifier-respecting lossless retraction*

$$\rho : L_0 \to L_1$$

satisfying

$$\forall x : L_1, \quad \rho(\pi(x)) \sim_{L_1} x.$$

Here classifier-respecting means that

$$u \sim_{L_0} v \Longrightarrow \rho(u) \sim_{L_1} \rho(v).$$

Proof. Assume such a ρ exists, and choose $x, y : L_1$ witnessing a nontrivial fiber. Since $\pi(x) \sim_{L_0} \pi(y)$, classifier-respect for the displayed reconstruction gives

$$\rho(\pi(x)) \sim_{L_1} \rho(\pi(y)).$$

The retraction hypothesis gives

$$\rho(\pi(x)) \sim_{L_1} x \qquad \text{and} \qquad \rho(\pi(y)) \sim_{L_1} y.$$

Symmetry and transitivity of $\sim_{L_1}$ therefore imply $x \sim_{L_1} y$, contradicting the nontrivial-fiber witness. □

Definition 9413.23 (Information dimension)**.** *For an information layer tower, the* information dimension *of a row is the least layer index at which the row can be represented without loss by the displayed carriers, classifiers, and ledgers. A row $x : L_m$ has information dimension at most $n \leq m$ when there are displayed maps*

$$c : L_m \to L_n, \qquad \rho : L_n \to L_m$$

with

$$\rho(c(x)) \sim_{L_m} x.$$

It has information dimension exactly n when n is minimal among the layers for which such data are displayed.

Principle 9413.24 (No shadow replacement)**.** *A lower-layer shadow may be used as an observation, approximation, or comparison coordinate, but it may not replace the higher-layer row unless the tower supplies an explicit reconstruction map, classifier-respect witness, and ledger. Without those data, the omitted fiber is part of the object.*

9413.6 Pythagorean Carrier, Scale Fibers, and Unit-Circle Projection

The common-carrier discipline becomes concrete in the elementary Pythagorean surface. The equation

$$a^2 + b^2 = c^2$$

and the normalized unit-circle equation

$$(a/c)^2 + (b/c)^2 = 1$$

are not two unrelated facts. They are two reality readouts of one carrier: an algebraic cone readout, a normalized shape readout, and a scale readout whose loss is precisely the fiber of the shape projection.

Definition 9413.25 (Field-like reality interface)**.** *A* field-like reality interface *is a stable reality interface with a carrier K, classifier $\sim_K$, operations*

$$+, \ -, \ \cdot, \ 0, \ 1,$$

and an apartness predicate $x\#0$. Its ledger licenses an inverse x^{-1} only from an apartness witness $x\#0$, and division is the derived operation

$$a/x := a \cdot x^{-1}.$$

Thus division is not a total primitive operation; it consumes an apartness row.

Definition 9413.26 (Nonzero scale carrier)**.** *The nonzero scale carrier of K is*

$$K^{\#} := \sum_{c:K} (c\#0).$$

An element is written (c, ν_c), where $\nu_c : c\#0$ is the displayed apartness witness.

Definition 9413.27 (Pythagorean carrier)**.** *The* Pythagorean carrier *over K is*

$$\mathsf{Pyth}_K := \sum_{a,b,c:K} \big((c\#0) \times (a^2 + b^2 \sim_K c^2)\big).$$

An element is written

$$P = (a, b, c; \nu_c; h_P),$$

where $\nu_c : c\#0$ licenses the divisions by c, and $h_P : a^2 + b^2 \sim_K c^2$ is the square-sum certificate. The carrier is therefore a packet with a scale-apartness row and an equation row, not a bare triple.

Definition 9413.28 (Algebraic unit-circle carrier)**.** *The algebraic unit-circle carrier over K is*

$$S^1_K := \sum_{x,y:K} \big(x^2 + y^2 \sim_K 1\big).$$

An element is written $Q = (x, y; h_Q)$. This is an algebraic circle carrier. Topology, metric structure, arc length, and measure enter only through additional reality interfaces.

Definition 9413.29 (Pythagorean projections)**.** *The Pythagorean carrier has the following paper-side projections:*

$$\pi_{\text{alg}}(a, b, c; \nu_c; h_P) := (a, b, c; h_P),$$
$$\pi_{\text{scale}}(a, b, c; \nu_c; h_P) := (c, \nu_c),$$

and

$$\pi_{\text{shape}}(a, b, c; \nu_c; h_P) := \big(a/c,\ b/c;\ h_{\text{shape}}\big).$$

The proof row h_{shape} is supplied by Definition 9413.30.

Theorem 9413.30 (Pythagorean normalization)**.** *Let*

$$P = (a, b, c; \nu_c; h_P) : \mathsf{Pyth}_K.$$

Set

$$x := a/c, \qquad y := b/c.$$

Then

$$x^2 + y^2 \sim_K 1.$$

Consequently π_{shape} is a well-defined projection from Pyth_K to S^1_K.

Proof. By definition of division, $x = a \cdot c^{-1}$ and $y = b \cdot c^{-1}$. The field-like ledger gives the usual distributive and inverse laws under $\sim_K$, so

$$x^2 + y^2 \sim_K a^2c^{-2} + b^2c^{-2} \sim_K (a^2 + b^2)c^{-2}.$$

Using $h_P : a^2 + b^2 \sim_K c^2$, this becomes

$$(a^2 + b^2)c^{-2} \sim_K c^2c^{-2} \sim_K 1,$$

where the final step uses the apartness witness ν_c. □

Definition 9413.31 (Scale reconstruction)**.** *Let $Q = (x, y; h_Q) : S^1_K$ and let $(c, \nu_c) : K^\#$. The* scale reconstruction *is*

$$\mathsf{Recon}(Q, c) := (cx, cy, c; \nu_c; h_{\text{recon}}) \in \mathsf{Pyth}_K,$$

where h_{recon} is the square-sum certificate

$$(cx)^2 + (cy)^2 \sim_K c^2.$$

Theorem 9413.32 (Scale reconstruction)**.** *For every $Q = (x, y; h_Q) : S^1_K$ and $(c, \nu_c) : K^\#$,*

$$(cx)^2 + (cy)^2 \sim_K c^2.$$

Proof. From h_Q we have $x^2 + y^2 \sim_K 1$. Therefore

$$(cx)^2 + (cy)^2 \sim_K c^2x^2 + c^2y^2 \sim_K c^2(x^2 + y^2) \sim_K c^2 \cdot 1 \sim_K c^2.$$

□

Theorem 9413.33 (Shape-scale decomposition)**.** *There is a classifier-respecting equivalence*

$$\mathsf{Pyth}_K \simeq S^1_K \times K^\#.$$

The forward map is

$$\Phi(P) := (\pi_{\text{shape}}(P), \pi_{\text{scale}}(P)),$$

and the reverse map is

$$\Psi(Q, c) := \mathsf{Recon}(Q, c).$$

Proof. Let $P = (a, b, c; \nu_c; h_P)$. Applying Φ gives the shape-scale pair

$$((a/c, b/c),\ (c, \nu_c)).$$

Applying Ψ returns

$$(c(a/c), c(b/c), c; \nu_c; -).$$

The inverse laws licensed by ν_c identify $c(a/c) \sim_K a$ and $c(b/c) \sim_K b$, and the square-sum row transports along those classifier witnesses. Hence $\Psi(\Phi(P)) \sim_{\mathsf{Pyth}} P$.

Conversely, for $Q = (x, y; h_Q)$ and $(c, \nu_c) : K^\#$, applying Ψ gives $(cx, cy, c; \nu_c; -)$. Applying Φ returns

$$(((cx)/c, (cy)/c; -),\ (c, \nu_c)).$$

The same inverse laws identify $(cx)/c \sim_K x$ and $(cy)/c \sim_K y$, so the shape row and scale row are classified together with (Q, c). □

Corollary 9413.34 (Unit-circle readout is not the complete carrier)**.** *The shape carrier S^1_K is not the complete Pythagorean carrier. The complete carrier is classified by the shape together with the nonzero scale:*

$$\mathsf{Pyth}_K \simeq S^1_K \times K^\#.$$

Proof. This is exactly the decomposition of Definition 9413.33. The projection to S^1_K forgets the $K^\#$ coordinate. □

Definition 9413.35 (Scale fiber of a shape)**.** *For $Q : S^1_K$, the* scale fiber *of the shape projection is*

$$\mathsf{ScaleFib}(Q) := \sum_{P:\mathsf{Pyth}_K} (\pi_{\text{shape}}(P) \sim_{S^1_K} Q).$$

Theorem 9413.36 (Scale fiber classification)**.** *For every $Q : S^1_K$,*

$$\mathsf{ScaleFib}(Q) \simeq K^{\#}.$$

Proof. Map $(P, \epsilon) : \mathsf{ScaleFib}(Q)$ to $\pi_{\text{scale}}(P) : K^{\#}$. Conversely, map $(c, \nu_c) : K^{\#}$ to the reconstructed packet $\mathsf{Recon}(Q, c)$ together with the shape-classifier witness supplied by the inverse laws in the proof of Definition 9413.33. The two composites are classified as identities by that same shape-scale decomposition. □

Corollary 9413.37 (Shape projection is lossy when scales differ)**.** *If $Q : S^1_K$ is a displayed shape and $K^{\#}$ contains two scales c, d with $c \not\sim_K d$, then the fiber of $\pi_{\text{shape}} : \mathsf{Pyth}_K \to S^1_K$ over Q is nontrivial. Consequently the shape projection satisfies the noncollapse pattern of Definition 9413.21 at every such displayed shape.*

Proof. By Definition 9413.36, the fiber over the displayed Q is classified by $K^{\#}$. Distinct scales therefore give distinct Pythagorean packets with the same shape readout. □

Definition 9413.38 (Pythagorean extension lift)**.** *Let $F \subset E$ be a field-extension reality with a displayed extension ledger. If $a, b \in F$ and*

$$c \in E, \qquad c^2 \sim_E a^2 + b^2, \qquad c \notin F,$$

then (a, b, c) is a Pythagorean extension lift *of the F-data into E.*

Theorem 9413.39 (Extension lift reaches normalized coordinates)**.** *Let $F \subset E$ be a field-extension reality. Suppose $a, b \in F$ and*

$$c \in E, \qquad c\#0, \qquad c^2 \sim_E a^2 + b^2, \qquad c \notin F.$$

If $a\#0$, then $a/c \notin F$. If $b\#0$, then $b/c \notin F$. Thus, in the nondegenerate case $a\#0$ and $b\#0$, the normalized shape point $(a/c, b/c)$ belongs to the extension-facing carrier E^2 and not to F^2.

Proof. Assume $a\#0$ and suppose $a/c \in F$. Write $r := a/c$. Since $a\#0$ and $c\#0$, the field-like ledger gives $r\#0$. Then

$$c \sim_E a/r.$$

Because $a \in F$, $r \in F$, and $r\#0$, closure of the field-like interface F under licensed division gives $a/r \in F$, hence $c \in F$, contradicting the extension-lift hypothesis. The argument for b/c is identical. □

Example 9413.40 (The $(1, 1, \sqrt{2})$ carrier)**.** *Let $F = \mathbb{Q}$ and $E = \mathbb{Q}(\sqrt{2})$. The packet*

$$(1, 1, \sqrt{2})$$

satisfies $1^2 + 1^2 = (\sqrt{2})^2$. Its normalized shape is

$$\left(1/\sqrt{2}, 1/\sqrt{2}\right).$$

Since $1/\sqrt{2} = \sqrt{2}/2 \notin \mathbb{Q}$, the scale readout and the unit-circle readout both consume the same extension ledger $\mathbb{Q} \subset \mathbb{Q}(\sqrt{2})$.

Definition 9413.41 (Pythagorean measure readouts)**.** *The triangle-area readout is the algebraic projection*

$$A_{\triangle}(a, b, c; \nu_c; h_P) := \frac{1}{2}ab.$$

It is an oriented algebraic area row unless positivity and orientation ledgers are added. If K is embedded in an analytic reality A whose ledger supplies a circle constant, the disk-area readout is

$$A_{\circ}(a, b, c; \nu_c; h_P) := \pi c^2$$

inside A. The constant π is not supplied by the field-like interface K; it belongs to the analytic reality that certifies it.

Theorem 9413.42 (Area readout through shape and scale)**.** *Let $P = (a, b, c; \nu_c; h_P) : \mathsf{Pyth}_K$, and let $x = a/c$ and $y = b/c$. Then*

$$\frac{1}{2}ab \sim_K \frac{1}{2}c^2xy.$$

Proof. The inverse laws give $a \sim_K cx$ and $b \sim_K cy$. Therefore

$$ab \sim_K (cx)(cy) \sim_K c^2xy.$$

Multiplying by $\frac{1}{2}$ gives the stated area readout. □

Corollary 9413.43 (Measure may require an analytic layer)**.** *Even when the scale belongs to an algebraic extension E, the disk-area readout πc^2 may require an analytic reality strictly beyond E.*

Proof. The algebraic extension ledger accounts for c and hence for c^2. It does not by itself account for the circle constant π. If the active analytic reality supplies π only through a separate process or transcendental ledger, the disk-area readout factors through that higher interface. □

Definition 9413.44 (Pythagorean generalized continuation)**.** *The Pythagorean algebra-to-shape continuation packet is*

$$\mathsf{GenCont}_{\mathrm{alg}\to\mathrm{shape}} := (\mathsf{Pyth}_K, \pi_{\mathrm{alg}}, \pi_{\mathrm{shape}}, \mathsf{Tr}_{\mathrm{alg,shape}}, \mathsf{Led}_{\mathrm{alg,shape}}),$$

where the transport row is Definition 9413.30. The reverse shape-scale-to-algebra packet uses Definition 9413.32.

Theorem 9413.45 (Pythagorean algebra-geometry transport)**.** *On the common carrier Pyth_K, the algebraic equation $a^2 + b^2 \sim_K c^2$ and the normalized shape equation $(a/c)^2 + (b/c)^2 \sim_K 1$ are transport-equivalent only with the scale fiber retained. The unit-circle point alone does not reconstruct the Pythagorean packet.*

Proof. The algebra-to-shape direction is Definition 9413.30. The reverse direction is Definition 9413.32, whose input is not a bare point of S^1_K but a pair in $S^1_K \times K^{\#}$. By Definition 9413.34, the scale fiber is required for the full carrier read. □

Theorem 9413.46 (Pythagorean common-carrier principle)**.** *The carrier Pyth_K binds algebraic equation, normalized shape, nonzero scale, triangle area, and disk area as projections of one packet. It satisfies*

$$\mathsf{Pyth}_K \simeq S^1_K \times K^{\#},$$

and the projection to S^1_K has scale fiber $K^{\#}$. If the scale is read only through a field extension, the normalized shape coordinates consume the same extension ledger in the nondegenerate case; if disk area is read, a further analytic ledger may be required.

Proof. The equivalence is Definition 9413.33, and the fiber computation is Definition 9413.36. The field-extension statement is Definition 9413.39. The measure statement is Definition 9413.43. All five readouts use the same source row $(a, b, c; \nu_c; h_P)$, so they are reality projections of one common carrier rather than independent objects. □

9413.7 Algebraic Extension Directions and Information Dimension

The Pythagorean example isolates the role of the scale row: when $(1, 1, \sqrt{2})$ is read over $\mathbb{Q}$, the full carrier cannot stay inside the base-field layer. It is read through the extension $\mathbb{Q}(\sqrt{2})$, and the normalized shape coordinates consume the same extension ledger. This section records the general algebraic form of that phenomenon. An algebraic element outside a base field is not an anomalous point of the base carrier; it is a finite-dimensional extension direction over the base field, certified by its minimal polynomial and its power-basis ledger.

The hierarchy is compatible with the L8 algebra layer of the project roadmap: field extensions, algebraic closure, Galois extensions, splitting fields, separability, and transcendence are mature algebra interfaces downstream of arithmetic and upstream of the geometry and analysis surfaces that consume them.

Definition 9413.47 (Field-extension reality)**.** *A* field-extension reality *is a displayed embedding*

$$\iota : F \hookrightarrow E$$

between field-like reality interfaces in the sense of Definition 9413.25, together with classifier-preservation ledgers for $0, 1, +, -, \cdot$*, apartness, licensed inverses, and the induced scalar action*

$$F \times E \to E.$$

The notation E/F *refers to the whole packet: base interface, extension interface, embedding, scalar action, operation compatibility, and readback ledger.*

Definition 9413.48 (Generated simple extension)**.** *Let* E/F *be a field-extension reality and let* $\alpha \in E$*. The* simple extension generated by α *is the least displayed sub-reality of* E *containing* $\iota(F)$ *and* α *and closed under the licensed field operations. It is denoted*

$$F(\alpha).$$

Its constructive rows are represented by rational expressions

$$\frac{p(\alpha)}{q(\alpha)}, \qquad p, q \in F[X], \qquad q(\alpha)\#0,$$

with ledger entries for the expression, denominator apartness, classifier equivalence of expressions, operation closure, and readback into E*.*

Definition 9413.49 (Algebraic element and minimal-polynomial certificate)**.** *An element* $\alpha \in E$ *is* algebraic over F *when there is a nonzero polynomial* $p(X) \in F[X]$ *with*

$$p(\alpha) \sim_E 0.$$

A minimal-polynomial certificate *for α is a polynomial $m_\alpha(X) \in F[X]$ such that*

$$m_\alpha \neq 0, \qquad m_\alpha \text{ is monic}, \qquad m_\alpha(\alpha) \sim_E 0,$$

and every polynomial p satisfying $p(\alpha) \sim_E 0$ is divisible by m_α in $F[X]$. Its degree is written

$$\deg_F(\alpha) := \deg m_\alpha.$$

Definition 9413.50 (Algebraic extension direction)**.** *For an algebraic element α over F, the* algebraic extension direction generated by *α is the simple extension*

$$Z_\alpha^{\mathrm{alg}} := F(\alpha).$$

Its algebraic information dimension over F is

$$\dim_F Z_\alpha^{\mathrm{alg}} := [F(\alpha) : F],$$

the vector-space dimension of the extension carrier over the base carrier.

Definition 9413.51 (Power-basis candidate)**.** *If α has minimal-polynomial degree d, its* power-basis candidate *is*

$$\mathcal{B}_\alpha := (1, \alpha, \alpha^2, \ldots, \alpha^{d-1}).$$

Theorem 9413.52 (Power-basis independence)**.** *Let m_α be a minimal-polynomial certificate of degree d. Then*

$$1, \alpha, \alpha^2, \ldots, \alpha^{d-1}$$

are linearly independent over F.

Proof. Suppose there is a nontrivial linear relation

$$a_0 + a_1\alpha + \cdots + a_{d-1}\alpha^{d-1} \sim_E 0, \qquad a_i \in F.$$

Let

$$p(X) = a_0 + a_1 X + \cdots + a_{d-1}X^{d-1}.$$

Then $p(\alpha) \sim_E 0$, $p \neq 0$, and $\deg p < d$. The minimal-polynomial certificate says $m_\alpha \mid p$, which is impossible for a nonzero p of smaller degree. Hence every displayed linear relation is trivial. □

Theorem 9413.53 (Power-basis spanning)**.** *If $\deg m_\alpha = d$, every element of $F(\alpha)$ is classified with a linear combination*

$$b_0 + b_1\alpha + \cdots + b_{d-1}\alpha^{d-1}, \qquad b_i \in F.$$

Proof. For any polynomial $p(X) \in F[X]$, divide by m_α:

$$p(X) = q(X)m_\alpha(X) + r(X), \qquad \deg r < d.$$

Evaluating at α gives $p(\alpha) \sim_E r(\alpha)$ because $m_\alpha(\alpha) \sim_E 0$. Thus polynomial expressions reduce to linear combinations of the displayed power-basis rows.

For a general rational expression $p(\alpha)/q(\alpha)$ with $q(\alpha)\#0$, the denominator row says q does not vanish at α. The minimal-polynomial certificate therefore gives the coprimality ledger of q with m_α in $F[X]$. A Euclidean ledger supplies $u, v \in F[X]$ with

$$u(X)q(X) + v(X)m_\alpha(X) = 1.$$

After evaluation, $u(\alpha)q(\alpha) \sim_E 1$, so $q(\alpha)^{-1} \sim_E u(\alpha)$. Hence

$$\frac{p(\alpha)}{q(\alpha)} \sim_E p(\alpha)u(\alpha),$$

and the product again reduces to a combination of powers below d. □

Theorem 9413.54 (Algebraic extension dimension)**.** *If α is algebraic over F and has minimal-polynomial degree d, then*

$$[F(\alpha) : F] = d.$$

Moreover $\mathcal{B}_\alpha = (1, \alpha, \ldots, \alpha^{d-1})$ is a displayed F-basis of $F(\alpha)$.

Proof. Definition 9413.52 gives linear independence, and Definition 9413.53 gives spanning. Thus the power-basis candidate is a basis, and its length is d. □

Definition 9413.55 (Algebraic noncollapse)**.** *For an algebraic element $\alpha \in E$ over F, define*

$$\mathsf{AlgNonCollapse}_F(\alpha) := \alpha \notin F.$$

Theorem 9413.56 (Algebraic noncollapse and degree)**.** *For an algebraic element α over F,*

$$\mathsf{AlgNonCollapse}_F(\alpha) \iff \deg_F(\alpha) > 1.$$

Proof. If $\deg_F(\alpha) > 1$, then α cannot lie in F, since an element of F has the degree-one annihilator $X - \alpha$. Conversely, if $\alpha \notin F$ and $\deg_F(\alpha) = 1$, the monic minimal polynomial has the form $X - a$ for some $a \in F$, forcing $\alpha \sim_E a$ and hence $\alpha \in F$, a contradiction. □

Corollary 9413.57 (Algebraic information dimension detects noncollapse)**.** *For algebraic α over F,*

$$\mathsf{AlgNonCollapse}_F(\alpha) \iff [F(\alpha) : F] > 1.$$

Proof. Combine Definition 9413.56 with Definition 9413.54. □

Example 9413.58 (The $\mathbb{Q}(\sqrt{2})$ extension)**.** *Let $F = \mathbb{Q}$ and $\alpha = \sqrt{2}$. The polynomial*

$$m_\alpha(X) = X^2 - 2$$

is irreducible over $\mathbb{Q}$ and has degree 2. Therefore

$$[\mathbb{Q}(\sqrt{2}) : \mathbb{Q}] = 2$$

and the power basis is

$$(1, \sqrt{2}).$$

Thus

$$\mathbb{Q}(\sqrt{2}) = \{a + b\sqrt{2} : a, b \in \mathbb{Q}\},$$

with equality read as the extension classifier supplied by the minimal-polynomial and reduction ledger.

Corollary 9413.59 (Pythagorean irrational scale has dimension two)**.** *The Pythagorean packet*

$$(1, 1, \sqrt{2})$$

from Definition 9413.40 is read over the two-dimensional extension reality

$$\mathbb{Q}(\sqrt{2})/\mathbb{Q}.$$

Its normalized shape point

$$(1/\sqrt{2}, 1/\sqrt{2}) = (\sqrt{2}/2, \sqrt{2}/2)$$

uses the same extension carrier.

Proof. By Definition 9413.58, the extension dimension is two. The normalized coordinate statement is the special case of Definition 9413.39 with $a = b = 1$ and $c = \sqrt{2}$. □

Definition 9413.60 (Extension projection graph)**.** *Let $F \subset E$ be a field-extension reality and let $P \in \mathsf{Pyth}_E$. The* extension projection graph

$$\mathsf{ExtProj}(P; F, E)$$

records, for each displayed projection of P, whether the readout lies in the base layer F or only in the extension layer E. For example, when $P = (1, 1, \sqrt{2})$ over $\mathbb{Q} \subset \mathbb{Q}(\sqrt{2})$, the leg readout lies in $\mathbb{Q}^2$, the scale readout lies in $\mathbb{Q}(\sqrt{2}) \setminus \mathbb{Q}$, and the normalized shape readout lies in $\mathbb{Q}(\sqrt{2})^2$ rather than $\mathbb{Q}^2$.

Theorem 9413.61 (Extension projection consistency)**.** *Let $F \subset E$ be a field-extension reality, and let*

$$P = (a, b, c; \nu_c; h_P) \in \mathsf{Pyth}_E$$

with $a, b \in F$, $c \in E \setminus F$, and $a\#0$, $b\#0$. Then the algebraic equation, the scale readout, and the normalized shape readout are all read through the same extension ledger $F \subset E$: the equation closes in E, the scale readout lies outside F, and the normalized shape readout lies outside F^2.

Proof. The equation closes in E because P carries the row $h_P : a^2 + b^2 \sim_E c^2$. The scale readout is c, which is outside F by hypothesis. The normalized shape coordinates are outside F by Definition 9413.39. All three readouts are projected from the same packet P and the same embedding ledger $F \subset E$. □

Definition 9413.62 (Algebraic extension continuation)**.** *The finite algebraic continuation from F to $F(\alpha)$ is the packet*

$$\mathsf{GenCont}_{F \to F(\alpha)} := (F(\alpha), \iota, \alpha, m_\alpha, \mathcal{B}_\alpha, \mathsf{Led}_\alpha),$$

where $\iota : F \hookrightarrow F(\alpha)$ is the embedding, m_α is the minimal-polynomial certificate, $\mathcal{B}_\alpha = (1, \alpha, \ldots, \alpha^{d-1})$ is the power basis, and Led_α records reduction, multiplication, licensed division, classifier transport, and readback into the ambient extension.

Theorem 9413.63 (Algebraic extensions are finite continuation packets)**.** *If α is algebraic over F with minimal-polynomial degree d, then $F \to F(\alpha)$ is a finite certificate-style generalized continuation in the sense of Definition 9413.12.*

Proof. The minimal-polynomial certificate is finite, and Definition 9413.54 supplies the finite power basis. Multiplication is computed by reducing powers α^n with $n \geq d$ using the relation $m_\alpha(\alpha) \sim 0$. Addition and negation are coordinatewise. Licensed division uses denominator apartness and the Euclidean ledger in $F[X]$ as in Definition 9413.53. Thus the carrier, operations, classifiers, stability rows, and readback are all displayed by finite data. □

Definition 9413.64 (Algebraic information dimension)**.** *For algebraic α over F, define*

$$\dim_{\text{info}}^{\text{alg}}(\alpha/F) := [F(\alpha) : F].$$

Theorem 9413.65 (Algebraic extension direction principle)**.** *Let α be algebraic over F. If $\alpha \in F$, then the generated extension has algebraic information dimension 1. If $\alpha \notin F$, then $F(\alpha)$ is a nontrivial finite-dimensional extension over F, its dimension is $\deg_F(\alpha)$, and its carrier is given by a finite continuation packet. In a Pythagorean carrier, an extension-scale row $c \notin F$ lifts both the scale projection and the normalized shape projection through the same extension reality.*

Proof. The dimension statement is Definition 9413.54. The noncollapse criterion is Definition 9413.56. The finite continuation packet is Definition 9413.63. The Pythagorean projection statement is Definition 9413.61. □

9413.8 Transcendental Extensions, Process Layers, and Apophatic Far Ends

section 9413.7 identified algebraic irrationality with a finite-dimensional extension direction:

$$F \subset F(\alpha), \qquad [F(\alpha) : F] = \deg m_\alpha.$$

The next boundary is different. A transcendental object is not a finite algebraic direction over the base field, but it need not be apophatic either. Constants and functions such as π, e, $\exp(x)$, $\log(x)$, and $\zeta(s)$ can be given by streams, moduli, branch data, operation laws, and ledgers. They are cataphatic process names whenever those data are displayed. Their transcendence, zero structure, or algebraic dependence is an additional certificate, not a consequence of the process name itself.

This section separates three roles. Algebraic extension directions are finite-dimensional over the base. Transcendental process directions are positive carriers, often infinite-dimensional over the algebraic base, with finite observations and ledgers. The apophatic far end T, introduced in Definition 9393.11 and disciplined by Definition 9401.3, is not an object in that tower.

Definition 9413.66 (Transcendence certificate)**.** *Let E/F be a field-extension reality and let $\tau \in E$. A* transcendence certificate *for τ over F is a ledger*

$$\mathsf{TransCert}_F(\tau)$$

which assigns to every nonzero polynomial $p \in F[X]$ an apartness witness

$$p(\tau)\#0.$$

Thus transcendence is not treated as a bare external negation of algebraicity. It is the displayed family of nonvanishing rows for all finite polynomial tests over the base field.

Definition 9413.67 (Transcendental rational-function extension)**.** *If τ carries* $\mathsf{TransCert}_F(\tau)$*, the* transcendental rational-function extension *generated by τ is the carrier*

$$F(\tau)$$

whose rows are represented by rational expressions

$$\frac{p(\tau)}{q(\tau)}, \qquad p, q \in F[X], \qquad q \neq 0.$$

The denominator apartness row is supplied by $\mathsf{TransCert}_F(\tau)$*, because $q \neq 0$ implies $q(\tau)\#0$. The packet therefore contains the expression syntax, polynomial evaluation ledger, denominator-apartness witness, classifier-equivalence rule, operation closure, and readback into the ambient extension.*

Theorem 9413.68 (Transcendental powers are linearly independent)**.** *If τ carries* $\mathsf{TransCert}_F(\tau)$*, then the family*

$$1, \tau, \tau^2, \tau^3, \ldots$$

is linearly independent over F in the finite-relation sense.

Proof. Suppose there is a displayed finite linear relation

$$a_0 + a_1\tau + \cdots + a_n\tau^n \sim_E 0, \qquad a_i \in F,$$

with the coefficients not all zero. Let

$$p(X) = a_0 + a_1X + \cdots + a_nX^n.$$

Then $p \neq 0$ and $p(\tau) \sim_E 0$. The transcendence certificate supplies $p(\tau)\#0$, contradicting the zero classifier. Hence no nontrivial finite linear relation exists. □

Corollary 9413.69 (Transcendental extension is not finite algebraic dimension)**.** *If τ carries* $\mathsf{TransCert}_F(\tau)$*, then $F(\tau)$ is not a finite-dimensional vector space over F.*

Proof. Definition 9413.68 gives arbitrarily long finite linearly independent families

$$1, \tau, \ldots, \tau^n.$$

No finite-dimensional vector space over F contains such families of unbounded length. □

Definition 9413.70 (Analytic process reality)**.** *An* analytic process reality *is a process-facing carrier whose rows have the schematic form*

$$(\mathsf{approx}, \mathsf{modulus}, \mathsf{regularity}, \mathsf{Led}).$$

The approximation row supplies finite observations, the modulus row licenses convergence reads, the regularity row records the analytic or domain law, and Led *records operation closure, classifier stability, branch conditions, apartness obligations, and boundary conditions. Typical rows include process names for*

$$\exp(x), \quad \log(x), \quad \sin(x), \quad \cos(x), \quad \pi, \quad e, \quad \zeta(s),$$

whenever their domains, moduli, and operation ledgers are displayed.

Definition 9413.71 (Process candidate and certified transcendental)**.** *Let x be a row of an analytic process reality and let F be a base field-like reality. The row x is a* transcendence candidate over F *when its process data are displayed but no* $\mathsf{TransCert}_F(x)$ *row is present. It is a* certified transcendental over F *when the same process row is equipped with* $\mathsf{TransCert}_F(x)$*.*

Theorem 9413.72 (Process naming does not imply transcendence)**.** *The existence of an analytic process name for x does not by itself produce* $\mathsf{TransCert}_F(x)$*.*

Proof. An analytic process name supplies approximation, modulus, regularity, operation, classifier, branch, and boundary rows. A transcendence certificate requires, for every nonzero polynomial $p \in F[X]$, an apartness witness $p(x)\#0$. The latter is a uniform algebraic test family over the base field. It is not one of the rows contained in a bare process name, and it cannot be inferred from finite observations without an additional ledger. □

Corollary 9413.73 (Analytic constants are process rows before algebraic claims)**.** *In the unified-reality reading, π, e, and $\zeta(s)$ first enter as analytic process names when their streams, moduli, domains, branches, and operation ledgers are displayed. Claims that such rows are algebraic, transcendental, zero, nonzero, or branch-compatible require separate certificates.*

Proof. The process rows are those of Definition 9413.70. The separation between process construction and transcendence certification is Definition 9413.72. □

Definition 9413.74 (Exponential-logarithmic reality bridge)**.** *An* exponential-logarithmic reality bridge *is a continuation packet*

$$\mathsf{ExpLogBridge} := (\exp, \log, \mathsf{Dom}, \mathsf{Branch}, \mathsf{Tr}, \mathsf{Led})$$

whose ledgers include the classifier-stable rows

$$\exp(x+y) \sim \exp(x)\exp(y),$$

$$\log(xy) \sim \log(x) + \log(y),$$

and, on licensed branches,

$$\log(\exp x) \sim x, \qquad \exp(\log z) \sim z.$$

The logarithmic side carries branch and domain data, while the exponential side carries convergence and operation data. Division, inverse, logarithm, and branch transport are licensed only when the required apartness or domain rows are present.

Theorem 9413.75 (Exponential and logarithmic transport)**.** *On a displayed domain and branch ledger,* $\mathsf{ExpLogBridge}$ *transports additive structure to multiplicative structure through* exp*, and transports multiplicative structure back to additive structure through* log*.*

Proof. The bridge ledger contains $\exp(x+y) \sim \exp(x)\exp(y)$, so an additive relation read on the source side is transported to a multiplicative relation on the target side. Conversely, the branch ledger contains $\log(xy) \sim \log(x) + \log(y)$ on its licensed domain, so the multiplicative relation is transported back to an additive read. Each transport consumes the displayed domain, branch, apartness, and classifier-stability rows. □

Corollary 9413.76 (Exponential is a reality bridge)**.** *The exponential row is not merely a pointwise function symbol. With its domain, convergence, operation, and transport ledgers, it is a generalized continuation packet in the sense of Definition 9413.12.*

Proof. Definition 9413.75 supplies the transport row, and Definition 9413.74 supplies the source, target, branch, and ledger data required by generalized continuation. □

Definition 9413.77 (Cataphatic process and apophatic far-end roles)**.** *A row x is a* cataphatic process *when it is given by positive data*

$$(\mathsf{stream}, \mathsf{modulus}, \mathsf{regularity}, \mathsf{Led}).$$

By contrast, the apophatic far-end role is the boundary reading in which a forward-binding socket has far end named T *only in the sense of Definition 9401.2. The relation* $\equiv_{\mathrm{apo}}$ *is not kernel equality, and Definition 9401.3 forbids reading* T *as a state, record, carrier element, or closed-generation rule.*

Theorem 9413.78 (Transcendental carriers are not apophatic far ends)**.** *If τ carries* $\mathsf{TransCert}_F(\tau)$*, then $F(\tau)$ may be infinite-dimensional over F, but it remains a mathematical carrier. It is not* T *or an apophatic far end.*

Proof. By Definition 9413.69, $F(\tau)$ is not finite-dimensional over F. Nevertheless its rows are rational expressions $p(\tau)/q(\tau)$ with displayed denominator apartness, classifier equivalence, field operations, and readback. Thus it can be consumed by functions, relations, classifiers, and operation ledgers. The apophatic far end T cannot be consumed in that way: by Definition 9401.3, it is not a state, record, carrier element, or rule. Therefore infinite transcendental dimension and apophatic boundary naming are distinct roles. □

Corollary 9413.79 (The far end is not a highest-dimensional object)**.** T *is not an infinite-dimensional extension, a transcendence basis, a collection of all directions, or a point in a highest universe.*

Proof. Each of the listed readings would make T available as an object for membership, function application, relation, or equality. That internalization is exactly what Definition 9401.3 excludes. □

Definition 9413.80 (Process noncollapse over an algebraic layer)**.** *Let x be a row of an analytic process reality over a base field F. Its* process noncollapse over the algebraic layer *is the condition*

$$\mathsf{ProcNonCollapse}_F(x) := \neg\exists\alpha,\ \alpha \text{ algebraic over } F \ \wedge\ x \sim \alpha.$$

The equivalence symbol is read through the relevant process and algebraic classifiers; the condition asserts that no finite algebraic row supplies a lossless substitute for x.

Theorem 9413.81 (Transcendence implies algebraic-layer noncollapse)**.** *If x carries* $\mathsf{TransCert}_F(x)$*, then* $\mathsf{ProcNonCollapse}_F(x)$*.*

Proof. Assume that $x \sim \alpha$ for some α algebraic over F. Then there is a nonzero polynomial $p \in F[X]$ with $p(\alpha) \sim 0$. Classifier-respecting polynomial evaluation gives $p(x) \sim p(\alpha) \sim 0$. The transcendence certificate gives $p(x)\#0$, a contradiction. Hence no algebraic row over F can classify as a lossless substitute for x. □

Corollary 9413.82 (Transcendental rows exceed finite algebraic extensions)**.** *If x carries* $\mathsf{TransCert}_F(x)$*, then x is not losslessly expressible inside any finite algebraic extension of F.*

Proof. Every row expressible in a finite algebraic extension over F is algebraic over F. This is exactly the algebraic-layer collapse ruled out by Definition 9413.81. □

Definition 9413.83 (Algebra-to-analysis continuation)**.** *An* algebra-to-analysis continuation *is a generalized continuation packet*

$$\mathsf{GenCont}_{\mathrm{alg}\to\mathrm{an}} := (\mathcal{O}, \pi_{\mathrm{alg}}, \pi_{\mathrm{an}}, \mathsf{Tr}, \mathsf{Led})$$

where π_{alg} *reads finite algebraic expressions,* π_{an} *reads streams, moduli, limits, or analytic process rows,* Tr *records which algebraic relations are respected by the analytic process, and* Led *records convergence, branch, apartness, domain, and boundary obligations. A reverse read from the analytic process back to an algebraic row requires a separate bridge; otherwise the packet records noncollapse.*

Theorem 9413.84 (Algebra-to-analysis continuation is process extension)**.** *An algebra-to-analysis continuation does not turn an algebraic point into an unaudited host point. It extends finite algebraic expressions into process rows with displayed moduli and ledgers.*

Proof. By Definition 9413.83, the analytic projection reads streams, moduli, limits, process laws, and boundary ledgers. This is the same noncollapse discipline as Definition 9413.16 and Definition 9413.24: lower-layer finite observations are shadows of the process row, not replacements for it. A host point reading would discard the modulus, regularity, branch, apartness, and provenance rows that make the analytic carrier auditable. □

Theorem 9413.85 (Transcendental-process-far-end separation principle)**.** *In the unified-reality tower, finite algebraic extension directions, transcendental process directions, and apophatic far-end roles are pairwise distinct interface roles:*

$$\textit{finite algebraic direction} \neq \textit{transcendental process direction} \neq \textit{apophatic far end.}$$

Proof. Finite algebraic directions are classified by Definition 9413.54: their dimension is the degree of a minimal polynomial. Transcendental directions satisfy Definition 9413.69 and require an additional certificate beyond process naming by Definition 9413.72. Cataphatic process rows and apophatic far-end roles are separated by Definition 9413.77 and Definition 9413.78. Thus none of the three roles may be collapsed into either of the others without an explicit bridge, and no such bridge is supplied by the common carrier alone. □

9413.9 Spectral Curves as Common Carriers

The preceding sections moved from common carriers to projection ledgers, noncollapse, algebraic extension directions, and analytic process layers. The quartic spectral example gives a more elaborate worked case. The object is not a single polynomial, a recurrence, a matrix, or an elliptic curve in isolation. It is a fixed common carrier whose projections read recurrence algebra, rational generating functions, companion-matrix spectra, algebraic curves, elliptic normalization, real projection, dominant branches, and endpoint zeros.

The fixed quartic is

$$\Pi(\lambda, y) := \lambda^4 - \lambda^3 - (2y+1)\lambda^2 + \lambda + y(y+1).$$

All statements in this section are scoped to this fixed quartic and its displayed ledgers. The example is useful precisely because the recurrence, spectral polynomial, elliptic model, real projection, dominant branch, and fixed-window endpoint-zero law are connected by finite certificate rows rather than by an informal analogy between different mathematical pictures.

Definition 9413.86 (Quartic recurrence source)**.** *The* quartic recurrence source *is the rational generating function row*

$$Z(t,y) := \sum_{m\geq 0} Z_m(y)t^m = \frac{N(t,y)}{D(t,y)}$$

with

$$N(t,y) := 1 + yt + (y^2 - y - 1)t^2 + (y^3 - 2y)t^3$$

and

$$D(t,y) := 1 - t - (2y+1)t^2 + t^3 + y(y+1)t^4.$$

The packet

$$\mathsf{RecSource}_\Pi := (N, D)$$

records the source polynomials, the identity $D(t,y)Z(t,y) = N(t,y)$, and the coefficient-readback ledger for the sequence $(Z_m(y))_{m\geq 0}$.

Definition 9413.87 (Recurrence and spectral polynomial realities)**.** *The* recurrence reality $\mathcal{R}_{\mathrm{rec}}$ *has carrier the polynomial sequence $(Z_m(y))_{m\geq 0}$ and structure law*

$$Z_m(y) - Z_{m-1}(y) - (2y+1)Z_{m-2}(y) + Z_{m-3}(y) + y(y+1)Z_{m-4}(y) = 0$$

for $m \geq 4$. The spectral polynomial reality $\mathcal{R}_{\mathrm{specpoly}}$ *has carrier the reciprocal spectral face*

$$\Pi(\lambda, y) = \lambda^4 D(1/\lambda, y).$$

Definition 9413.88 (Companion matrix reality)**.** *The* companion matrix reality $\mathcal{R}_{\mathrm{mat}}$ *is the matrix family*

$$T(y) = \begin{pmatrix} 1 & 2y+1 & -1 & -y(y+1) \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

with spectral readout

$$\det(\lambda I - T(y)) = \Pi(\lambda, y).$$

It realizes the order-four recurrence as a transfer system and exposes branch, eigenvalue, and dominance questions as matrix-spectral reads.

Definition 9413.89 (Quartic spectral common carrier)**.** *The* quartic spectral common carrier *is the packet*

$$\mathcal{O}_\Pi := (\mathsf{RecSource}_\Pi, (Z_m)_{m\geq 0}, T(y), \Pi(\lambda, y), \mathcal{C}_\Pi)$$

where

$$\mathcal{C}_\Pi := \{(\lambda, y) : \Pi(\lambda, y) = 0\}$$

is the affine spectral curve. It carries projections

$$\pi_{\mathrm{gen}}, \pi_{\mathrm{rec}}, \pi_{\mathrm{mat}}, \pi_{\mathrm{specpoly}}, \pi_{\mathrm{curve}}$$

to the generating-function, recurrence, companion-matrix, spectral-polynomial, and curve realities.

Theorem 9413.90 (Recurrence, matrix, and spectral polynomial share one carrier)**.** *The rational generating function $Z(t,y) = N(t,y)/D(t,y)$, the order-four recurrence for $Z_m(y)$, the companion matrix $T(y)$, and the spectral polynomial $\Pi(\lambda, y)$ are projections of $\mathcal{O}_\Pi$.*

Proof. The generating-function projection reads the source row (N, D). Comparing coefficients in $D(t, y)Z(t, y) = N(t, y)$ gives the recurrence row. The companion matrix in Definition 9413.88 is the transfer matrix for the same recurrence, and its characteristic polynomial is

$$\det(\lambda I - T(y)) = \Pi(\lambda, y).$$

Finally $\Pi(\lambda, y) = \lambda^4 D(1/\lambda, y)$ reads the reciprocal spectral face of the denominator. Thus the four presentations are ledgered projections of the same packet rather than unrelated objects. □

Concretized in §8880: The concrete NameCert packet records the recurrence, companion-matrix, spectral-polynomial, curve, and projection ledgers for the fixed quartic carrier.

Corollary 9413.91 (The spectral polynomial is not external data)**.** *The polynomial* $\Pi(\lambda, y)$ *is both the reciprocal face of the generating-function denominator and the characteristic face of the companion matrix.*

Proof. This is exactly the pair of identities

$$\Pi(\lambda, y) = \lambda^4 D(1/\lambda, y), \qquad \Pi(\lambda, y) = \det(\lambda I - T(y)).$$

□

Definition 9413.92 (Balanced quartic and elliptic normalization)**.** *The balanced quartic family is*

$$P_{a,c,d,g}(\lambda, y) := \lambda^4 + a\lambda^3 + (c - 2y)\lambda^2 + d\lambda + y(y - c) + g.$$

With

$$\xi = y - \lambda^2, \qquad Y = 2\xi - c,$$

the equation $P_{a,c,d,g} = 0$ *becomes*

$$Y^2 = c^2 - 4a\lambda^3 - 4d\lambda - 4g.$$

The normalized double-root conditions at $(\lambda, y) = (1, 0)$ *and* $(\lambda, y) = (-1, 1)$ *select the member*

$$(a, c, d, g) = (-1, -1, 1, 0),$$

which is Π*. Its elliptic normalization reality is*

$$E: \quad Y^2 = 4\lambda^3 - 4\lambda + 1, \qquad y = \lambda^2 - \frac{1}{2} + \frac{1}{2}Y.$$

Theorem 9413.93 (Elliptic normalization is a continuation packet)**.** *The transport from the affine spectral curve*

$$\mathcal{C}_\Pi : \Pi(\lambda, y) = 0$$

to

$$E: \quad Y^2 = 4\lambda^3 - 4\lambda + 1$$

is a generalized continuation from the quartic algebraic-curve reality to the elliptic-geometry reality.

Proof. In the canonical member, the change of variables is

$$\xi = y - \lambda^2, \qquad Y = 2\xi + 1,$$

with reverse read

$$y = \lambda^2 - \frac{1}{2} + \frac{1}{2}Y.$$

The ledger therefore transports the quartic equation to the displayed Weierstrass model and transports points back by the y-projection. The smoothness, degree-four projection, and ramification computations are not hidden assumptions; they are rows of the normalization ledger. Hence the move is a generalized continuation in the sense of Definition 9413.12. □

Definition 9413.94 (Ramification ledger)**.** *The* ramification ledger RamLed_Π *contains the finite rows*

$$2\lambda Y = 1 - 3\lambda^2,$$
$$(\lambda - 1)(\lambda + 1)(16\lambda^3 - 9\lambda^2 + 1) = 0,$$

and the branch-value polynomial

$$-y(y - 1)(256y^3 + 411y^2 + 165y + 32),$$

together with the squarefreeness and elimination certificates that justify the branch read.

Theorem 9413.95 (Ramification is a normalized-curve readout)**.** RamLed_Π *is the ramification readout of the same normalized carrier* $E \to \mathbb{P}^1_y$*, not an external discriminant calculation.*

Proof. The projection is

$$y = \lambda^2 - \frac{1}{2} + \frac{1}{2}Y$$

on the curve $Y^2 = 4\lambda^3 - 4\lambda + 1$. Differentiating the displayed projection under the Weierstrass relation gives the ramification row

$$2\lambda Y = 1 - 3\lambda^2.$$

Eliminating (λ, Y) gives the ramification abscissa equation and the branch-value polynomial. Thus the branch ledger is a geometric readout of the normalized curve. □

Definition 9413.96 (Real projection reality)**.** *The* real projection reality *is the map*

$$E(\mathbb{R}) \xrightarrow{y} \mathbb{R}_y.$$

Its ledger records the real oval decomposition, the y-image of each oval, the real-fiber count for regular values, and the multiplicity-counted root count at branch values. In the canonical quartic, the compact oval maps to $[y_{\mathrm{sub}}, 1]$ *and the noncompact oval maps to* $[0, \infty)$*.*

Theorem 9413.97 (Real-root counts are projection-fiber counts)**.** *The real-root interval diagram of* $\Pi(\lambda, y)$ *is the real projection readout of the elliptic normalization reality.*

Proof. Real roots of $\Pi(\cdot, y)$ correspond to points of

$$E(\mathbb{R}) \cap y^{-1}(y).$$

The oval decomposition, endpoint values, ramification values, and Sturm/sign rows determine the real-fiber count. Therefore the real-root diagram is a finite projection-fiber readout of the normalized carrier. □

Definition 9413.98 (Dominant spectral branch reality)**.** *The* positive-axis dominant branch reality *records the read*

$$y > 0 \longmapsto \lambda_{\mathrm{dom}}(y)$$

where $\lambda_{\mathrm{dom}}(y)$ is the unique maximal-modulus root of $\Pi(\cdot, y)$. In the fixed quartic, the branch satisfies $\lambda_{\mathrm{dom}}(y) > 1$ for $y > 0$, and the surviving simple dominant root at $y = 1$ is $\lambda_{\mathrm{dom}}(1) = 2$.

Theorem 9413.99 (Dominance is finer than ramification)**.** *The branch locus does not determine the recurrence-zero boundary. A branch point contributes to endpoint zero theory only when the colliding eigenvalues participate in maximal-modulus competition.*

Proof. The discriminant projection records collisions of spectral branches, but recurrence zeros are controlled by the dominant terms in the spectral decomposition. A branch point whose colliding eigenvalue is strictly subdominant does not control the endpoint zero law. In the fixed quartic, y_{sub} is a branch point with subdominant collision, and $y = 1$ is a real collision while the root 2 remains the dominant simple spectral read. The endpoint boundary at $y = 0$ is different: the colliding branches are active in the positive-axis maximal-modulus competition. Thus dominance is an additional ledger, not a consequence of the discriminant alone. □

Corollary 9413.100 (Discriminant projection is lossy)**.** *The discriminant locus is a lossy projection of the spectral carrier; it cannot recover the recurrence-zero boundary without a dominance ledger.*

Proof. Definition 9413.99 exhibits branch points that do not contribute to the endpoint-zero boundary. The recurrence-zero boundary therefore cannot be read from the discriminant projection alone. □

Definition 9413.101 (Endpoint zero reality)**.** *The* endpoint zero reality *is the local zero-distribution carrier for $Z_m(y)$ under the endpoint scale*

$$y = -\frac{u^2}{m^2}.$$

Its ledger contains the local branch coordinate s with $y = s^2$, the colliding branches $\lambda_\pm(s)$, the partial-fraction amplitudes $C_\pm(s)$, the non-colliding remainder estimate, fixed-window Rouché isolation, and the endpoint zero ordering.

Theorem 9413.102 (Endpoint zeros are local branch interference)**.** *The fixed-window endpoint zero law is the endpoint-zero projection of $\mathcal{O}_\Pi$. It is obtained by transporting the local branch collision through Puiseux branches, partial-fraction amplitudes, cosine interference, and Rouché zero isolation.*

Proof. At $y = 0$, the polynomial $\Pi(\lambda, 0)$ has a double root at $\lambda = 1$. With $y = s^2$, the active branches and amplitudes have local rows of the form

$$\lambda_\pm(s) = 1 \pm \frac{s}{\sqrt{2}}\frac{5}{8}s^2 \mp \frac{43}{64\sqrt{2}}s^3 + O(s^4)$$

and

$$C_\pm(s) = \frac{1}{2} \mp \frac{3\sqrt{2}}{16}s + O(s^2).$$

On the endpoint scale $s = iu/m$, the two active terms

$$C_+(s)\lambda_+(s)^m + C_-(s)\lambda_-(s)^m$$

combine into a shifted cosine model, while the non-colliding branches are controlled by the fixed-window remainder ledger. Rouché isolation then transports the approximate cosine zeros to actual zeros of $Z_m(-u^2/m^2)$ in the fixed window. □

Corollary 9413.103 (Endpoint zeros are not discriminant roots)**.** *In the fixed quartic, endpoint zeros are controlled by active spectral branch interference, not by the discriminant equation alone.*

Proof. The discriminant gives branch collision. The endpoint zero law also needs dominance, partial-fraction amplitudes, non-colliding remainder control, and Rouché isolation. These are precisely the rows listed in Definition 9413.101. □

Definition 9413.104 (Finite exact certificate ledger and scope boundary)**.** *The* finite exact certificate ledger $\mathsf{CertLed}_\Pi$ *consists of determinant and characteristic-polynomial identities, resultants, discriminant factorizations, ramification resultants, Sturm/sign tables, endpoint Puiseux coefficients, amplitude residue expansions, remainder estimates, and Rouché window bounds. The* scope boundary $\mathsf{ScopeBnd}_\Pi$ *refuses the following exports: a parameter-space classification of quartic families, a general equilibrium measure, a growing-window endpoint zero law, a uniform law for growing k, a standalone recurrence-boundary theory, or any RH or zeta theorem.*

Theorem 9413.105 (Quartic spectral carrier is finite-certificate common carrier)**.** *The packet $\mathcal{O}_\Pi$ is a finite-certificate common carrier: each transport between its realities is supported by rows of* $\mathsf{CertLed}_\Pi$ *and bounded by* $\mathsf{ScopeBnd}_\Pi$.

Proof. The recurrence-to-spectrum transport uses $D(t,y)Z(t,y) = N(t,y)$ and $\Pi(\lambda,y) = \lambda^4 D(1/\lambda,y)$. The matrix-to-spectrum transport uses $\det(\lambda I - T(y)) = \Pi(\lambda,y)$. The quartic-to-elliptic transport uses the balanced change of variables and normalization ledger. The elliptic-to-real-projection transport uses the real oval decomposition, extrema, branch values, and Sturm/sign rows. The real-projection-to-dominance transport uses the positive-axis dominance ledger. The dominance-to-endpoint-zero transport uses Puiseux branches, residue amplitudes, remainder estimates, and fixed-window Rouché isolation. These rows are finite certificates, and the scope boundary prevents the local fixed-quartic result from being flattened into any global quartic-family or growing-window theorem. □

Theorem 9413.106 (Quartic spectral common-carrier principle)**.** *The fixed quartic realizes the unified-reality pattern*

recurrence algebra → spectral polynomial → elliptic geometry → real projection → dominant branch → endpoint ze

Each arrow is a ledgered projection or transport from $\mathcal{O}_\Pi$, and none is licensed to collapse the others.

Proof. The shared-carrier statement is Definition 9413.90. The elliptic transport is Definition 9413.93. Real projection is Definition 9413.97. Dominance noncollapse is Definition 9413.99. Endpoint-zero transport is Definition 9413.102. The finite-certificate and scope discipline is Definition 9413.105. Together these statements exhibit the fixed quartic as a complex worked example of the common-carrier, projection, continuation, and noncollapse discipline. □

9413.10 Recursive Spectral Zero Hierarchy

The quartic spectral carrier, finite-field elliptic zeta functions, and global zeta or L-functions share a recursive spectral mechanism. They do not share one literal zero set. The commonality is structural: finite observations are folded into a generating object, the generating object exposes spectral factors, normalization identifies the balanced surface, and zero constraints are read only through the ledgers available at that level.

Definition 9413.107 (Zeta-like spectral hierarchy)**.** *The* zeta-like spectral hierarchy *separates three levels:*

$$\mathsf{RecurrenceSpectralZeros} \longrightarrow \mathsf{CurveZetaZeros} \longrightarrow \mathsf{GlobalZetaLFunctionZeros}.$$

The first level contains zeros such as the endpoint zeros of the fixed quartic recurrence carrier. The second level contains zeta zeros of geometric objects over finite fields, such as numerator roots of elliptic Hasse–Weil zeta functions. The third level contains zeros of global arithmetic zeta and L-functions, including the Riemann zeta zero ledger. The arrows are carrier-role refinements, not inclusions of one point set into another.

Definition 9413.108 (Recursive spectral fold)**.** *A* recursive spectral fold *is a packet*

$$\mathsf{ObsTime} \to \mathsf{GenFunction} \to \mathsf{SpectralFactors} \to \mathsf{Normalization} \to \mathsf{ZeroConstraint},$$

where finite observations indexed by an observation-time parameter are compressed into a generating function or Euler product; spectral factors read the compressed carrier; normalization identifies the balanced surface; and zero constraints are derived only after the required dominance, symmetry, exactness, or continuation ledgers have been displayed.

Theorem 9413.109 (Recursive spectral folds are not self-reference)**.** *A recursive spectral fold is a hierarchical compression of finite observation rows, not a circular definition of its own zero set.*

Proof. The input rows are finite observations such as recurrence coefficients, point counts over finite extensions, or prime-local Euler factors. The generating object is constructed from those rows. Spectral factors are then read from the generated object, and zero constraints are read from the spectral factors plus the relevant ledgers. Each stage consumes the previous stage as data; no stage assumes the zero constraint that it is supposed to produce. □

Definition 9413.110 (Quartic recurrence zero level)**.** *The quartic recurrence zero level is the recursive spectral fold*

$$(Z_m(y))_{m\geq 0} \to Z(t,y) = \frac{N(t,y)}{D(t,y)} \to \Pi(\lambda, y) \to E:\ Y^2 = 4\lambda^3 - 4\lambda + 1 \to \mathsf{EndpointZeros}.$$

Its zeros are zeros of the finite recurrence polynomials $Z_m(y)$ under the endpoint scaling and fixed-window ledgers of Definition 9413.101.

Theorem 9413.111 (Quartic endpoint zeros are recurrence-spectral zeros)**.** *The endpoint zeros in the fixed quartic carrier are recurrence-spectral zeros. They are zeta-like by recursive spectral form, but they are not zeros of the Riemann zeta function or of an elliptic Hasse–Weil zeta function.*

Proof. By Definition 9413.106, the fixed quartic carrier transports recurrence algebra through the spectral polynomial, elliptic normalization, real projection, dominance, and endpoint-zero ledgers. The endpoint zeros are obtained by local branch interference and Rouché isolation in Definition 9413.102. No Euler product, finite-field point-count zeta function, or completed Riemann zeta zero ledger is part of this carrier. Thus the analogy is structural rather than an identification of zero sets. □

Definition 9413.112 (Finite-field elliptic zeta level)**.** *For an elliptic curve $E/\mathbb{F}_q$, the finite-field elliptic zeta level folds the point-count time series*

$$r \longmapsto \#E(\mathbb{F}_{q^r})$$

into

$$Z(E/\mathbb{F}_q, T) := \exp\left(\sum_{r\geq 1} \#E(\mathbb{F}_{q^r})\frac{T^r}{r}\right).$$

For an elliptic curve this has the rational form

$$Z(E/\mathbb{F}_q, T) = \frac{1 - aT + qT^2}{(1-T)(1-qT)} = \frac{(1-\alpha T)(1-\beta T)}{(1-T)(1-qT)}.$$

The spectral roots α, β are the Frobenius readouts of the point-count fold.

Theorem 9413.113 (Frobenius roots recursively control point counts)**.** *In the finite-field elliptic zeta level, the spectral roots determine the point-count observations by*

$$\#E(\mathbb{F}_{q^r}) = q^r + 1 - \alpha^r - \beta^r.$$

Thus the point-count time series and the Frobenius spectral roots are two readouts of the same recursive fold.

Proof. The exponential generating definition records all point-count rows. The rational form of the elliptic zeta function packages the same data into the numerator factors $(1-\alpha T)(1-\beta T)$ together with the two denominator rows $(1-T)$ and $(1-qT)$. Expanding the logarithmic derivative of the displayed rational expression recovers the displayed point-count formula. Hence the spectral roots compress and read back the observation-time sequence. □

Definition 9413.114 (Finite-field elliptic RH-type normalization)**.** *The finite-field elliptic RH-type normalization is the row*

$$|\alpha| = |\beta| = \sqrt{q},$$

equivalently

$$\left|\frac{\alpha}{\sqrt{q}}\right| = \left|\frac{\beta}{\sqrt{q}}\right| = 1.$$

It is the finite geometric form of a normalized spectral-root constraint.

Theorem 9413.115 (Finite-field elliptic zeros give a closed local model)**.** *The finite-field elliptic zeta level is a closed local geometric model of RH-type normalization, not a proof of the Riemann zeta zero constraint.*

Proof. The normalization row of Definition 9413.114 places the Frobenius spectral roots on the unit circle after division by $\sqrt{q}$. This is a theorem at the finite-field elliptic level. The Riemann zeta zero ledger instead concerns complex zero rows s of the completed zeta carrier and the fixed-half condition $\Re(s) = 1/2$. Transport from the finite-field elliptic carrier to the completed Riemann zeta carrier is not supplied by the finite-field theorem. The remaining global bridge is the zero-defect rigidity route isolated in section 9413.16. □

Definition 9413.116 (Global elliptic and zeta zero level)**.** *For an elliptic curve over $\mathbb{Q}$, the global elliptic L-function level is assembled from local Euler factors. At a good prime p the local factor has the form*

$$1 - a_p p^{-s} + p^{1-2s}.$$

The global readout is an Euler product

$$L(E, s) = \prod_p L_p(E, p^{-s})^{-1}.$$

The Riemann zeta readout is the simpler global Euler product

$$\zeta(s) = \prod_p (1 - p^{-s})^{-1}.$$

Both are global arithmetic spectral carriers, with different local factor data and different proof obligations.

Theorem 9413.117 (Global elliptic L-functions are higher arithmetic siblings)**.** *Global elliptic L-functions and the Riemann zeta function share the Euler-product, Mellin, functional-equation, and zero-ledger pattern, but their zero ledgers are not the same object.*

Proof. Definition 9413.1147 records the elliptic, modular, and L-function bridge shape when the required bridge rows are displayed. The Riemann zeta carrier has its own prime-local factors and completed zero ledger, as used in section 9413.20. The two carriers share the global arithmetic spectral pattern, but a zero of one carrier is not automatically a zero of the other. Any transport between their zero statements requires its own bridge ledger. □

Definition 9413.118 (Normalized spectral-root analogy)**.** *The normalized spectral-root analogy compares the finite-field elliptic readout*

$$\frac{\alpha}{\sqrt{q}}$$

with the centered prime-channel readout

$$p^{1/2-s}.$$

Both are unit-circle constraints after normalization:

$$\left|\frac{\alpha}{\sqrt{q}}\right| = 1, \qquad |p^{1/2-s}| = 1.$$

The first is a Frobenius-root statement in a finite-field curve carrier. The second is a prime-channel reading of the fixed-half surface in the zeta carrier.

Theorem 9413.119 (Unit-circle analogies require carrier separation)**.** *The analogy between $\alpha/\sqrt{q}$ and $p^{1/2-s}$ identifies a common normalization pattern, not a common theorem or a common zero set.*

Proof. In the finite-field elliptic carrier, the unit-circle statement is the Frobenius-root normalization row. In the Riemann zeta carrier, $|p^{1/2-s}| = 1$ is equivalent to the fixed-half surface by Definition 9413.317. The latter is still only a surface readout until a zero row is shown to land on that surface. That additional step is the defect-zero incompatibility or prime-sieve odd exactness boundary of Definition 9413.202 and Definition 9413.477. □

Definition 9413.120 (Apophatic far end of recursive spectral towers)**.** *The far end of a recursive spectral tower is not an internal point of the tower. It is the boundary role named by the continuing availability of finite observation layers:*

$$r = 1, 2, 3, \ldots$$

or by unbounded prime windows, recurrence lengths, or extension degrees. In BEDC notation this is an apophatic far-end role, not a term to be substituted into the carrier as a value.

Theorem 9413.121 (Infinite observation time is not a spectral point)**.** *The infinite direction in a zeta-like recursive spectral fold is a boundary role for all finite readouts, not a point of the generated spectral carrier.*

Proof. For finite-field elliptic zeta functions, each r supplies the finite observation $\#E(\mathbb{F}_{q^r})$. The zeta function folds the entire finite observation family into a spectral object. It does not require a point $\mathbb{F}_{q^\infty}$ inside the carrier. The same discipline is the apophatic separation of Definition 9413.77 and Definition 9401.3: the far end may be named as a boundary role, but it is not consumed as a kernel object or as an additional spectral point. □

Theorem 9413.122 (Recursive spectral zero hierarchy principle)**.** *Quartic endpoint zeros, finite-field elliptic zeta zeros, global elliptic L-function zeros, and Riemann zeta zeros are projections of a common recursive spectral pattern only after their carrier boundaries are kept distinct.*

Proof. The quartic carrier uses recurrence data, a rational generating function, a spectral polynomial, elliptic normalization, dominance, and endpoint-zero ledgers. The finite-field elliptic carrier uses point counts over finite extensions, a Hasse–Weil zeta function, Frobenius spectral roots, and unit-circle normalization. The global elliptic and Riemann zeta carriers use Euler products, completion, functional equations, and zero ledgers. These are instances of Definition 9413.108. Their shared pattern licenses comparison of roles, while their separate ledgers forbid collapsing one zero reality into another. □

9413.11 Irreducible Information Layers and Spectral Factors

The recursive spectral hierarchy should not be read as the claim that an extra dimension mechanically contributes an extra zero. The precise reading is that an irreducible information layer contributes a spectral factor, and the spectral factor contributes zero or pole rows according to its sign in the zeta-like fold. Dimension enters only through the rank of the information layer being observed.

Definition 9413.123 (Irreducible spectral information layer)**.** *An* irreducible spectral information layer *of an object X is a carrier row*

$$\mathcal{I}_i(X)$$

whose finite observation-time readouts cannot be reconstructed from the other displayed layers without an explicit reconstruction ledger. Its rank is the number of independent spectral atoms required to read that layer in the selected spectral carrier.

Definition 9413.124 (Layer spectral factor)**.** *Let $\mathcal{I}_i(X)$ be an irreducible spectral information layer with spectral atoms*

$$\alpha_{i,1}, \ldots, \alpha_{i,b_i}.$$

Its layer spectral factor *is*

$$P_i(T) := \prod_{j=1}^{b_i} (1 - \alpha_{i,j} T).$$

The integer b_i is the spectral rank of the layer. It is the layer-level analogue of a Betti number or independent Frobenius-root count when such a reading is available.

Definition 9413.125 (Alternating zeta-like factorization)**.** *A zeta-like fold with irreducible information layers has the schematic factorization*

$$Z_X(T) = \prod_i P_i(T)^{(-1)^{i+1}}.$$

Positive exponents contribute zero rows of Z_X, while negative exponents contribute pole rows. The sign is part of the layer ledger, not a numerical dimension count.

Theorem 9413.126 (Spectral factors come from information layers)**.** *In a zeta-like recursive fold, zero and pole candidates arise from layer spectral factors, not from dimension labels by themselves.*

Proof. By Definition 9413.108, finite observations are first folded into a generating object and then read through spectral factors. By Definition 9413.124, a layer contributes one factor $P_i(T)$ whose degree is the number of independent spectral atoms of that layer. By Definition 9413.125, the same factor appears with a sign determining whether its roots are zero or pole rows. Therefore the proof-relevant datum is the irreducible layer and its factor, not the informal statement that a dimension has been added. □

Definition 9413.127 (Observation-to-spectrum exponential fold)**.** *For an object X with finite observation numbers $N_r(X)$, the observation-to-spectrum fold is*

$$Z_X(T) := \exp\left(\sum_{r \geq 1} N_r(X) \frac{T^r}{r}\right).$$

If the observations split as

$$N_r(X) = \sum_j \varepsilon_j \alpha_j^r,$$

then the fold reads as

$$Z_X(T) = \prod_j (1 - \alpha_j T)^{-\varepsilon_j}.$$

Theorem 9413.128 (Exponential folding converts spectral atoms into factors)**.** *The observation-to-spectrum fold converts each spectral atom α_j with sign ε_j into the factor $(1 - \alpha_j T)^{-\varepsilon_j}$.*

Proof. The logarithmic identity

$$-\log(1 - \alpha T) = \sum_{r \geq 1} \alpha^r \frac{T^r}{r}$$

shows that a contribution $\varepsilon_j \alpha_j^r$ to all observation levels contributes

$$-\varepsilon_j \log(1 - \alpha_j T)$$

to the logarithm of $Z_X(T)$. Exponentiating gives the displayed factor $(1 - \alpha_j T)^{-\varepsilon_j}$. Thus spectral atoms are exactly the factor-producing information carried by the observation sequence. □

Definition 9413.129 (Elliptic finite-field middle layer)**.** *For an elliptic curve $E/\mathbb{F}_q$, the middle information layer has two spectral atoms α, β and factor*

$$P_1(T) = 1 - aT + qT^2 = (1 - \alpha T)(1 - \beta T).$$

The two denominator factors

$$1 - T, \qquad 1 - qT$$

belong to the boundary layers. The zeta fold is

$$Z(E/\mathbb{F}_q, T) = \frac{P_1(T)}{(1 - T)(1 - qT)}.$$

Theorem 9413.130 (Elliptic curves have two middle spectral atoms)**.** *The finite-field elliptic curve does not acquire zeros because it has an extra geometric dimension. Its zeta numerator has two roots because its middle irreducible information layer has spectral rank two.*

Proof. The finite-field elliptic zeta level of Definition 9413.112 has numerator

$$1 - aT + qT^2 = (1 - \alpha T)(1 - \beta T).$$

The two roots are the Frobenius spectral atoms of the middle layer. The normalization row of Definition 9413.114 then places them on the unit circle after division by $\sqrt{q}$. The count two is the spectral rank of the middle layer, not the result of counting geometric dimensions as zeros. □

Definition 9413.131 (Higher curve layer pattern)**.** *For a smooth curve of genus g over a finite field, the middle spectral factor has schematic form*

$$P_1(T) = \prod_{j=1}^{2g} (1 - \alpha_j T).$$

The rank $2g$ records the independent middle-layer spectral atoms. The corresponding zeta-like fold has boundary factors from the lower and upper layers together with this middle factor.

Theorem 9413.132 (Layer rank determines factor degree)**.** *For curve zeta-like folds, the degree of the middle spectral factor is the rank of the middle information layer.*

Proof. In the elliptic case the rank is two and the factor is quadratic by Definition 9413.129. The higher-curve pattern of Definition 9413.131 displays the same rule with rank $2g$. The factor degree is therefore controlled by the number of independent spectral atoms in the layer. The dimension of the underlying geometric object matters only through this layer-rank ledger. □

Definition 9413.133 (Layer recursion rather than iterated ellipses)**.** *The phrase* layer recursion *denotes the process*

finite observations → generating function → spectral factor → normalized zero constraint → higher-level observati

It does not denote a literal nesting of elliptic curves or ellipses. A spectral factor may itself become data for a higher carrier only when a bridge ledger displays the new observation interface.

Theorem 9413.134 (Spectral factors become observation laws only by ledger)**.** *A spectral factor can control later observation rows, but it becomes a higher-level object only through an explicit bridge ledger.*

Proof. For the finite-field elliptic level, Definition 9413.113 shows that α, β determine the future point counts by

$$\#E(\mathbb{F}_{q^r}) = q^r + 1 - \alpha^r - \beta^r.$$

This is a readback law from spectral atoms to observation time. The quartic carrier similarly moves from recurrence observations through spectral branches to endpoint-zero laws by the ledgers of Definition 9413.106. In both cases, the spectral data become observation laws only because a readback or dominance ledger has been displayed. □

Definition 9413.135 (Global arithmetic spectral layer)**.** *The Riemann zeta carrier is the global arithmetic instance of the layer recursion pattern. Its local observation atoms are prime channels*

$$p^{-s},$$

its centered normalization channels are

$$p^{1/2-s},$$

and the completed carrier adds the non-point archimedean completion row of Definition 9413.140.

Theorem 9413.136 (RH is the global arithmetic normalized-zero constraint)**.** *In the global arithmetic layer, the RH fixed-half condition is the statement that zeta zero rows land on the simultaneous unit surface of all centered prime channels.*

Proof. For $s = 1/2 + \delta + i\tau$,

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

Thus

$$|p^{1/2-s}| = 1$$

for every displayed prime row exactly when $\delta = 0$, equivalently $\Re(s) = 1/2$, by Definition 9413.317. The remaining RH-facing problem is not this equivalence, but the zero-ledger statement that every nontrivial zeta zero lands on that surface. That missing bridge is the prime-sieve odd exactness or no-far-end absorption boundary of Definition 9413.477 and Definition 9413.143. □

Corollary 9413.137 (Information layers explain the spectral hierarchy)**.** *The zeta-like hierarchy is a hierarchy of irreducible information layers and their spectral factors, not a hierarchy of mechanically added zero coordinates.*

Proof. Definition 9413.16 and Definition 9413.23 provide the general language of layer and non-collapse. The present section specializes that language to spectral factors. Each irreducible layer contributes the spectral atoms needed for its observation-time readout; the resulting factors then contribute zero or pole rows according to the zeta-like factorization ledger. This is the precise content behind the recursive spectral hierarchy of Definition 9413.122. □

9413.12 No Far-End Absorption Boundary

The recursive spectral reading of zeta separates two meanings often compressed into the symbol ∞. The archimedean place is a positive completion row: it supplies the Gamma normalization needed for the completed zeta carrier and the functional-equation mirror. The apophatic far end is not such a row; it is the boundary role of an unbounded finite-observation tower. The RH-facing discipline is therefore not to remove the archimedean row, but to refuse the reification of the far end as a hidden point capable of absorbing finite prime defects.

Definition 9413.138 (Two infinity roles in the zeta carrier)**.** *The zeta carrier uses two distinct infinity-facing roles:*

$$\infty_{\text{place}} := \textit{archimedean completion row},$$

and

$$\mathsf{T}_{\text{far}} := \textit{apophatic far-end role.}$$

The first is represented analytically by the completion factor

$$\pi^{-s/2}\Gamma(s/2)$$

and by the pole-normalization rows entering the completed function. The second is a boundary name for the unbounded continuation of finite prime windows, recurrence lengths, or observation layers. It is not an element of the zeta carrier.

Definition 9413.139 (Finite prime-window recursive fold)**.** *For a finite prime window S, the raw finite Euler fold is*

$$Z_S(s) := \prod_{p\in S}(1-p^{-s})^{-1}.$$

If $q \notin S$ is a displayed prime row, the successor fold is

$$Z_{S\cup\{q\}}(s) = Z_S(s)(1-q^{-s})^{-1}.$$

Thus the raw Euler product is read as the compatible tower of finite prime-window folds, not as the value of a completed point $S = \mathsf{Prime}^{\uparrow}$ inside the carrier.

Definition 9413.140 (Archimedean completion row)**.** *The* archimedean completion row *for zeta is the analytic normalization packet containing*

$$\pi^{-s/2}\Gamma(s/2), \qquad s(s-1),$$

and the functional-equation ledger for the completed readout

$$\xi(s) = \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s).$$

This row is part of the completed carrier. It is not an infinite prime and not an apophatic point.

Theorem 9413.141 (Completion is not point reification)**.** *Passing from finite prime-window folds to the completed zeta carrier requires the archimedean completion row, but it does not introduce T_{far} as a point of the carrier.*

Proof. The finite prime-window tower is generated by Definition 9413.139. The completed carrier additionally consumes the analytic normalization row of Definition 9413.140. These are positive data rows. By Definition 9413.120 and Definition 9413.121, the far end of the tower is a boundary role for unbounded finite readouts, not an internal object. Therefore completion adds the archimedean row and functional-equation ledger; it does not add an extra point at infinity. □

Definition 9413.142 (Far-end absorption)**.** Far-end absorption *is the illicit inference pattern that treats the unbounded tail, the archimedean row, or the apophatic far-end role as a hidden term able to cancel a displayed finite prime-window defect without a source-compatible zero-exclusion ledger. In formulas, it is the attempted passage from*

$$D_S(s)\#0$$

to harmlessness inside the completed carrier without proving

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times D_S(s)\#0 \to \bot.$$

Definition 9413.143 (No far-end absorption)**.** *The* no far-end absorption *row for the zeta carrier is*

$$\mathsf{NoFarEndAbsorption}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}} (D_S(s)\#0) \to \bot \right).$$

It says that a finite prime-window defect cannot be hidden at the completed carrier boundary, whether the hiding is described as Euler tail, archimedean completion, functional-equation transport, or apophatic far-end pressure.

Theorem 9413.144 (No far-end absorption is prime-sieve odd exactness)**.** *For the zeta route, no far-end absorption is the completion-boundary form of prime-sieve odd exactness.*

Proof. The finite defect row in Definition 9413.143 is the same existential finite prime-window defect used in Definition 9413.477. The zero and critical-strip hypotheses place that defect in the completed zeta source. Both definitions exclude coexistence of that source row with a nonzero finite prime-window defect. The difference is terminological: prime-sieve odd exactness names the odd-defect mechanism, while no far-end absorption names the completion-boundary discipline that forbids the defect from being hidden by tail or far-end language. □

Theorem 9413.145 (Archimedean normalization is retained)**.** *No far-end absorption does not remove the archimedean completion row. It only refuses to read that row as a hidden defect absorber.*

Proof. The completed function row of Definition 9413.140 is required for the functional-equation mirror and for the completed carrier. Without that row, the fixed surface $\Re(s) = 1/2$ would not be read as the natural mirror surface of the completed object. No far-end absorption therefore preserves the archimedean row as a normalization and symmetry ledger. What it refuses is the additional, unledgered claim that this row can cancel a finite prime-window defect while leaving a zero ledger intact. □

Corollary 9413.146 (No far-end absorption is the RH-facing boundary)**.** *Within the recursive zeta tower, the RH-facing boundary is the claim that finite prime-window defect cannot be absorbed by completion.*

Proof. If $s = 1/2 + \delta + i\tau$ and $\delta \# 0$, then every nonempty finite prime window reads

$$D_S(s) = -\delta \sum_{p \in S} \log p \# 0$$

by Definition 9413.201. To obtain the RH fixed-half section, this visible finite defect must contradict the completed zero ledger. That contradiction is exactly the role of Definition 9413.202, Definition 9413.477, and the no-tail absorption row of Definition 9246.18. The no-far-end-absorption formulation states the same requirement as a boundary discipline for the recursive spectral tower. □

Theorem 9413.147 (Quartic carriers have finite closure, zeta has a far-end boundary)**.** *The quartic spectral carrier closes by finite certificate ledgers, while the zeta carrier requires a non-point far-end completion boundary.*

Proof. The fixed quartic carrier closes recurrence, spectral polynomial, elliptic normalization, dominance, and endpoint-zero rows through the finite certificate ledger of Definition 9413.105. Its endpoint-zero theorem is scoped by fixed windows and finite local branch data. The zeta carrier instead builds finite prime-window folds and then passes to analytic continuation, archimedean completion, and functional-equation rows. The latter passage has a far-end boundary: finite prime defects remain visible locally, but their incompatibility with zero rows requires the no-far-end-absorption ledger. □

Theorem 9413.148 (No-far-end formulation of the RH bridge)**.** *The RH bridge can be stated as the refusal of hidden absorption of finite prime defect at the apophatic far end of the zeta recursion tower.*

Proof. By Definition 9413.144, this refusal is equivalent to prime-sieve odd exactness. By Definition 9413.850, prime-sieve odd exactness, together with the located fixed-half collapse row, supplies the RH fixed-half section. Thus the no-far-end phrasing does not add a separate theorem. It names the exact boundary condition needed to prevent a finite off-half prime defect from being hidden by the completed carrier. □

9413.13 Finite Prime Tower as Completed Readout

The no-far-end route can be stated as a completed-readout discipline for the finite prime-window tower. The zeta carrier is not a last prime window and not an evaluation at an infinite prime set. It is the completed analytic readout of a recursively generated finite tower, together with archimedean normalization, functional-equation transport, and an apophatic boundary that cannot be consumed as a correction term.

Definition 9413.149 (Finite prime-window tower)**.** *The* finite prime-window tower *is the directed family of finite Euler folds*

$$E_S(s) := \prod_{p \in S} (1 - p^{-s})^{-1}$$

over nonempty finite prime windows S. If $q \notin S$ is a displayed prime row, the successor edge is

$$E_{S \cup \{q\}}(s) = E_S(s)(1 - q^{-s})^{-1}.$$

Thus the tower is generated by finite recursive extension, as in Definition 9413.139.

Definition 9413.150 (Finite Hamiltonian prime layer)**.** *For a nonempty finite prime window S and a depth row $M \geq 0$, the* finite Hamiltonian prime layer *is the finite occupation carrier*

$$\mathcal{K}_{S,M} := \bigotimes_{p \in S} \operatorname{span}\{|0\rangle_p, \ldots, |M\rangle_p\}.$$

The local number operator is

$$N_p|k\rangle_p = k|k\rangle_p,$$

and the finite Hamiltonian is

$$H_{S,M} := \sum_{p \in S} (\log p) N_p.$$

Its finite partition readout is

$$Z_{S,M}(s) := \operatorname{Tr}_{\mathcal{K}_{S,M}} \left(e^{-sH_{S,M}} \right) = \prod_{p \in S} \left(1 + p^{-s} + \cdots + p^{-Ms} \right).$$

Thus the finite Euler fold has a spectral-dynamical refinement: prime rows are occupation axes and $\log p$ *is the displayed frequency weight.*

Definition 9413.151 (Centered finite Hamiltonian readout)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the centered finite Hamiltonian readout is

$$\widetilde{Z}_{S,M}(s) := \operatorname{Tr}_{\mathcal{K}_{S,M}} \left(e^{-(s-1/2)H_{S,M}} \right).$$

Equivalently,

$$e^{-(s-1/2)H_{S,M}} = e^{-\delta H_{S,M}} e^{-i\tau H_{S,M}}.$$

The local centered prime channel is

$$U_p(s) := p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

Theorem 9413.152 (The critical line is the unitary Hamiltonian surface)**.** *For the centered finite Hamiltonian readout, the fixed-half surface is the unitary evolution surface.*

Proof. If $\delta = 0$, then

$$e^{-(s-1/2)H_{S,M}} = e^{-i\tau H_{S,M}},$$

and every local channel satisfies

$$|U_p(s)| = 1.$$

If $\delta \# 0$, the factor $e^{-\delta H_{S,M}}$ is a real amplitude-bias row. The local equality

$$|U_p(s)| = p^{-\delta}$$

is the same fixed-surface readout as Definition 9413.464. Thus the centered Hamiltonian language reads $\Re(s) = 1/2$ as the absence of nonunitary radial bias. □

Definition 9413.153 (Interlayer Hamiltonian lift)**.** *An* interlayer Hamiltonian lift *is one of the two finite extension rows*

$$(S, M) \to (S \cup \{q\}, M), \qquad (S, M) \to (S, M + 1).$$

For an additional displayed prime $q \notin S$, the Hamiltonian extension has the form

$$H_{S\cup\{q\},M} = H_{S,M} \otimes \mathbf{1}_q + \mathbf{1}_S \otimes (\log q)N_q.$$

For a depth extension, the carrier admits the additional occupation row $|M+1\rangle_p$ on each displayed prime axis. These are finite layer extensions, not an infinite prime point.

Definition 9413.154 (Spectral zero island of a finite layer)**.** *Let $F_{S,M}$ be a finite-layer spectral readout obtained from the Hamiltonian, sieve, or completed-tail ledger. A* spectral zero island *is a local zero component, or a certified small-value component, of the readout*

$$F_{S,M}(s).$$

The island is a zero-relevant object only when its source, layer, dominance or exactness ledger, and transition ledger are displayed. This is the zeta analogue of the quartic dominance discipline of Definition 9413.485.

Theorem 9413.155 (Normalized defect is invariant under prime-layer lifts)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the normalized prime-window defect of Definition 9413.166 is independent of the finite prime layer:

$$\overline{D}_S(s) = -\delta.$$

Consequently every interlayer lift preserving the same s preserves the normalized off-critical defect.

Proof. The displayed equality is exactly Definition 9413.167. If $S \subseteq T$ are nonempty finite prime windows, then both normalized readouts are equal to the same normal displacement:

$$\overline{D}_S(s) = -\delta = \overline{D}_T(s).$$

Depth extension does not change the prime-window normalization row D_S/L_S. Hence the nonzero normal defect is not erased by adding a prime axis or by increasing finite occupation depth. □

Definition 9413.156 (Completed readout of the finite prime tower)**.** *The* completed readout *of the finite prime-window tower is the zeta carrier packet*

$$\mathsf{CompletedReadout}_\zeta := (\mathsf{PrimeWindowTower}, \mathsf{AnalyticContinuation}, \mathsf{ArchimedeanCompletion}, \mathsf{FunctionalMirror}, \mathsf{ZeroLe}$$

The archimedean component is the positive completion row of Definition 9413.140. The far-end component is the non-point boundary role of Definition 9413.138; it is not an extra element, terminal window, or hidden state.

Definition 9413.157 (Completed-readout absorption error)**.** *A* completed-readout absorption error *is the passage from a cofinally stable finite prime-window defect to a harmless completed zero row without displaying an exactness ledger excluding that coexistence. In schematic form it treats*

$$\mathsf{CofStablePrimeDefect}(s)$$

as if it could be cancelled by the apophatic boundary of

$$\mathsf{CompletedReadout}_{\zeta}.$$

This is precisely the hidden-correction pattern excluded by Definition 9413.180.

Definition 9413.158 (Completed-readout exactness boundary)**.** *The* completed-readout exactness boundary *is the row*

$$\mathsf{TailZeroWithStableDefectRequiresFarEndCorrection}_{\zeta}.$$

It is the zeta-specific claim that a finite sieve tail-zero tower together with a cofinally stable nonzero normalized prime defect would require a hidden far-end correction. In the notation of Definition 9413.190, this is the precise carrier equation still needed for the completed readout route.

Theorem 9413.159 (Finite rows of the completed-readout route are closed)**.** *The finite rows of the completed-readout route are already displayed: zero rows enter every finite sieve tail, and off-half displacement is a cofinally stable normalized prime-window defect.*

Proof. For the zero side, Definition 9413.186 gives

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{FiniteSieveTailZero}_{\zeta}(s).$$

For the defect side, Definition 9413.167 gives

$$\overline{D}_S(s) = -\delta(s)$$

for every nonempty finite prime window, and Definition 9413.168 turns $\delta(s)\#0$ into cofinal visibility of the normalized defect. No appeal to an infinite prime point is used in either row. □

Definition 9413.160 (Off-critical island survival)**.** *An* off-critical island survival *row is a compatible family of finite spectral zero islands through interlayer Hamiltonian lifts whose normal coordinate satisfies*

$$\delta(s)\#0$$

and whose completed readout is a zeta zero-source row. It records the claim that a nonunitary centered Hamiltonian island survives the finite prime tower and enters the completed zero ledger.

Theorem 9413.161 (Off-critical island survival is the stable-defect exactness boundary)**.** *Excluding off-critical island survival is the spectral-dynamical form of the completed-readout exactness boundary.*

Proof. If an off-critical island survives, then $\delta(s)\#0$. By Definition 9413.155, the normalized finite prime-window defect is the same nonzero value $-\delta(s)$ at every nonempty finite prime layer. Thus the surviving island carries a cofinally stable normalized defect. If its completed readout is also a zero-source row, the situation is exactly the tail-zero stable-defect pattern named in Definition 9413.158. The remaining task is therefore not detection of the defect, but exclusion of its coexistence with the completed zero ledger. □

Theorem 9413.162 (Completed-readout exactness normal form)**.** *The completed-readout RH route has the normal form*

$$\mathsf{TailZeroWithStableDefectRequiresFarEndCorrection}_\zeta \times \mathsf{NoHiddenFarEndCorrection}_\zeta \Rightarrow \mathsf{NoFarEndAbsorption}_\zeta \Rightarrow \mathsf{RH}$$

under the located fixed-half collapse row.

Proof. Tail-zero stable-defect correction necessity and no-hidden far-end correction compose by Definition 9413.191 to give no off-balanced tail zero. By Definition 9413.188, this gives the normalized no-absorption row. The RH fixed-half section then follows from Definition 9413.182. □

Corollary 9413.163 (Completed readout does not close by far-end reification)**.** *The completed readout of the finite prime-window tower may use analytic continuation, archimedean completion, and functional-equation transport, but it cannot close a stable finite defect by treating the apophatic far end as a hidden absorber.*

Proof. Definition 9413.141 separates positive completion rows from the apophatic far-end role. Definition 9413.157 identifies the forbidden inference pattern, and Definition 9413.180 rejects the corresponding hidden correction row. Therefore completed readout is a carrier discipline, not a license to cancel a finite defect at an undisplayed endpoint. □

Corollary 9413.164 (Quartic comparison)**.** *The quartic spectral carrier closes by finite certificate rows, while the zeta completed readout closes the analogous RH route only after the tail-zero stable-defect exactness boundary is discharged.*

Proof. The quartic comparison is recorded in Definition 9413.147: fixed windows, finite branch ledgers, and dominance rows close the quartic endpoint-zero law. The zeta carrier instead has an unbounded finite prime-window tower. Its finite rows are visible by Definition 9413.159, but the zero-facing route still requires the completed-readout exactness boundary of Definition 9413.158. □

9413.14 Far-End Unity and Prime-Window Normalization

No far-end absorption can be sharpened by normalizing finite prime-window defects. The normalized readout removes the scale of the window and leaves only the stable normal displacement from the critical surface. Thus far-end unity is not convergence to a point at infinity; it is cofinal compatibility of all finite readouts with no hidden boundary term.

Definition 9413.165 (Prime-window logarithmic weight)**.** *For a nonempty finite prime window S, define its logarithmic weight by*

$$L_S := \sum_{p \in S} \log p.$$

The window is nonempty and every displayed prime has positive logarithm, so L_S is a positive normalization row.

Definition 9413.166 (Normalized prime-window defect)**.** *For a nonempty finite prime window S, the* normalized prime-window defect *is*

$$\overline{D}_S(s) := \frac{D_S(s)}{L_S}.$$

Here $D_S(s)$ is the finite prime-window defect of Definition 9413.200.

Theorem 9413.167 (Normalized defects are constant over finite windows)**.** *If*

$$s = \frac{1}{2} + \delta + i\tau,$$

then for every nonempty finite prime window S,

$$\overline{D}_S(s) = -\delta.$$

Proof. By Definition 9413.201, the finite-window defect has the explicit form

$$D_S(s) = -\delta \sum_{p \in S} \log p.$$

The denominator in Definition 9413.166 is exactly

$$L_S = \sum_{p \in S} \log p.$$

Since S is nonempty, this denominator is positive. Dividing gives

$$\overline{D}_S(s) = -\delta.$$

□

Corollary 9413.168 (Off-half displacement is cofinally visible)**.** *If $\delta(s)\#0$, then every nonempty finite prime window carries the same nonzero normalized defect.*

Proof. Definition 9413.167 gives $\overline{D}_S(s) = -\delta(s)$ for every nonempty finite window. Thus apartness of $\delta(s)$ from zero is read in every normalized window. The defect is therefore not a finite-window artifact, an Euler tail error, or a scale effect of the growing window. □

Definition 9413.169 (Far-end unity for zeta prime windows)**.** *The* far-end unity *row for a zeta packet s is the cofinal compatibility condition*

$$\mathsf{FarEndUnity}_\zeta(s) := \prod_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} \left(\overline{D}_S(s) = 0\right).$$

It says that every finite prime-window normalization reads no radial defect. It is not a claim that the prime tower converges to an additional point.

Theorem 9413.170 (Far-end unity is the fixed-half surface)**.** *For the zeta prime-window readout,*

$$\mathsf{FarEndUnity}_\zeta(s) \iff \delta(s) = 0.$$

Consequently it is equivalent to

$$\Re(s) = \frac{1}{2}.$$

Proof. If $\mathsf{FarEndUnity}_\zeta(s)$ holds, choose any nonempty finite prime window S. The unity row gives $\overline{D}_S(s) = 0$. By Definition 9413.167, the same term is $-\delta(s)$, hence $\delta(s) = 0$. Conversely, if $\delta(s) = 0$, the same theorem gives $\overline{D}_S(s) = 0$ for every nonempty finite window. The equivalence with $\Re(s) = 1/2$ is the definition of δ and the fixed-surface calculation of Definition 9413.313. □

Definition 9413.171 (Cofinal compatibility without hidden boundary term)**.** *A finite-readout tower has* cofinal compatibility without hidden boundary term *when every cofinal finite family has compatible normalized readouts and no additional far-end correction row is admitted unless it is displayed as a positive carrier row. For zeta prime windows this means that the family*

$$\left(\overline{D}_S(s)\right)_S$$

cannot be altered at the apophatic far end by an undisplayed term.

Theorem 9413.172 (Far-end unity is cofinal compatibility)**.** *For the zeta prime-window tower, far-end unity is exactly cofinal compatibility of normalized finite defects with zero boundary defect.*

Proof. The normalized defect row removes the window scale by Definition 9413.166. The constant-window theorem Definition 9413.167 then shows that all cofinal windows read the same normal displacement $-\delta(s)$. Far-end unity requires that common readout to be zero. No point at infinity is used; the row is a compatibility condition on all finite windows together with refusal of a hidden boundary correction. □

Definition 9413.173 (Zero far-end unity target)**.** *The* zero far-end unity target *is*

$$\mathsf{ZeroFarEndUnity}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \Big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{FarEndUnity}_\zeta(s)\Big).$$

It says that every critical-strip zero source row has cofinally unified prime-window normalizations.

Theorem 9413.174 (Zero far-end unity implies the RH section)**.** *If* $\mathsf{ZeroFarEndUnity}_\zeta$ *holds, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. The zero and critical-strip rows of z give

$$\mathsf{FarEndUnity}_\zeta(s)$$

by Definition 9413.173. By Definition 9413.170, $\delta(s) = 0$, hence $\Re(s) = 1/2$. The fixed-half section equivalence of Definition 9413.199 therefore supplies the RH section. □

Theorem 9413.175 (No far-end absorption is the zero-facing exactness form)**.** *No far-end absorption is the zero-facing exactness form of far-end unity: a zero row cannot carry a stable nonzero normalized prime-window defect.*

Proof. If a zero row carried $\delta(s)\#0$, then Definition 9413.168 would give a nonzero normalized defect in every nonempty finite prime window, and in particular a finite prime-window defect. The no-far-end absorption row of Definition 9413.143 excludes coexistence of that finite defect with the completed zero ledger. Conversely, the zero far-end unity target of Definition 9413.173 asserts directly that zero rows have no such stable normalized defect. Thus the two formulations name the same zero-facing exactness requirement from opposite sides: absorption refusal and cofinal unity. □

Corollary 9413.176 (Far-end unity is not ordinary point convergence)**.** *The BEDC reading of far-end convergence in the zeta route is cofinal compatibility plus no hidden boundary term, not convergence to an infinite point.*

Proof. Definition 9413.169 defines far-end unity entirely through finite prime windows. The far end enters only as the boundary role that forbids hidden correction terms, as in Definition 9413.171 and Definition 9413.120. This is the same discipline enforced by Definition 9413.141: the completed zeta carrier may use positive completion rows, but it does not turn ∞, T, or Ω into a point that absorbs differences. □

9413.15 Far-End Correction Obligation

The normalized prime-window route isolates a useful proof boundary. The formula

$$\overline{D}_S(s) = -\delta(s)$$

shows that off-half displacement is cofinally stable across all nonempty finite prime windows. The no-far-end-absorption row says that such a stable finite defect cannot coexist with a completed zeta zero row. What remains is to justify, inside the completed zeta carrier, that coexistence of zero and cofinally stable defect would require a hidden far-end correction. That implication is the exactness obligation.

Definition 9413.177 (Cofinal stable prime defect)**.** *A zeta packet s carries a* cofinal stable prime defect *when some nonempty finite prime window has nonzero normalized defect:*

$$\mathsf{CofStablePrimeDefect}(s) := \sum_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} \left(\overline{D}_S(s)\#0\right).$$

By Definition 9413.167, this is equivalent to $\delta(s)\#0$ for the centered prime-window readout.

Definition 9413.178 (Hidden far-end correction)**.** *A* hidden far-end correction *for a zeta packet s is an undisplayed correction row*

$$C_{\mathsf{T}}(s)$$

whose only role is to cancel a cofinal stable finite prime defect at the completed carrier boundary while preserving the zero classifier. It is not the archimedean completion row of Definition 9413.140; it is an additional, unlicensed absorber placed at the apophatic far end.

Definition 9413.179 (Far-end correction necessity)**.** *The* far-end correction necessity *row is the zeta-specific exactness implication*

$$\mathsf{FarEndCorrectionNecessity}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \mathsf{CofStablePrimeDefect}(s) \to \mathsf{HiddenFarEndCo}$$

It says that if a completed zero row and a cofinally stable finite prime defect were to coexist, the completed carrier would have to contain an extra far-end correction row explaining the cancellation.

Definition 9413.180 (No hidden far-end correction)**.** *The* no hidden far-end correction *row is*

$$\mathsf{NoHiddenFarEndCorrection}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{HiddenFarEndCorrection}(s) \to \bot).$$

It is the zeta-carrier instance of the apophatic far-end discipline: T, Ω, or the far-end boundary cannot be consumed as an internal correction object.

Theorem 9413.181 (Correction necessity plus refusal gives no absorption)**.** *Assume far-end correction necessity and no hidden far-end correction. Then no far-end absorption holds in normalized prime-window form:*

$$\prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \mathsf{CofStablePrimeDefect}(s) \to \bot).$$

Proof. Let s carry a zero row, a critical-strip row, and a cofinal stable prime defect. Far-end correction necessity supplies

$$\mathsf{HiddenFarEndCorrection}(s).$$

No hidden far-end correction sends this row to $\bot$. Hence the three input rows cannot coexist. □

Corollary 9413.182 (No absorption gives the RH section)**.** *Under located fixed-half collapse, the normalized no-absorption row of Definition 9413.181 implies the RH fixed-half section.*

Proof. If a critical-strip zero row had $\delta(s)\#0$, then Definition 9413.168 would give $\mathsf{CofStablePrimeDefect}(s)$. The normalized no-absorption row excludes this. Therefore $\neg(\delta(s)\#0)$ for every zero-source row. Located fixed-half collapse gives $\delta(s) = 0$, and Definition 9413.199 reads the result as the RH section. □

Theorem 9413.183 (Correction necessity is the unclosed exactness step)**.** *The no-hidden-far-end row is part of BEDC's apophatic discipline. The unclosed zeta-specific content is far-end correction necessity.*

Proof. Definition 9413.180 is a direct application of the non-internalization rule for apophatic far ends, already used in Definition 9413.141 and Definition 9413.176. It refuses to treat T or Ω as a hidden internal object. By contrast, Definition 9413.179 is an analytic exactness claim about the completed zeta carrier: it must show that zero plus cofinally stable finite defect cannot be reconciled without an unlicensed absorber. That claim is not supplied by $\mathsf{ZetaZero}(s)$ and remains the same hard point as Definition 9413.202. □

Definition 9413.184 (Finite sieve residual tail)**.** *For a finite prime window S, the* finite sieve residual tail *is the completed zeta readout after the displayed Euler factors in S have been removed:*

$$R_S(s) := \zeta(s) \prod_{p \in S} (1 - p^{-s}).$$

Equivalently, in the Euler-product region the finite Euler fold

$$E_S(s) := \prod_{p \in S} (1 - p^{-s})^{-1}$$

is separated from the residual readout by

$$\zeta(s) = E_S(s) R_S(s).$$

The finite sieve tower is the family $\left(R_S(s)\right)_S$ over finite prime windows.

Definition 9413.185 (Finite sieve tail-zero tower)**.** *The* finite sieve tail-zero tower *for a zeta packet s is*

$$\mathsf{FiniteSieveTailZero}_\zeta(s) := \prod_{S:\mathsf{FinitePrimeWindow}} (R_S(s) = 0)\,.$$

It says that every finite residual tail inherits the zero row after the displayed finite Euler factors have been removed.

Theorem 9413.186 (Zero rows enter finite sieve tails)**.** *Every critical-strip zeta zero row enters the finite sieve tail-zero tower:*

$$\prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{FiniteSieveTailZero}_\zeta(s))\,.$$

Proof. Let S be a finite prime window. By Definition 9413.184, $R_S(s)$ is obtained by multiplying the zeta value row by the displayed finite product

$$\prod_{p \in S} (1 - p^{-s}).$$

A zero row is preserved under this finite multiplication. The critical-strip row also ensures that each displayed factor is a legitimate finite sieve factor: if $0 < \Re(s) < 1$, then $|p^{-s}| < 1$, hence $1 - p^{-s}$ is not the zero denominator of a finite Euler factor. Thus every finite S has $R_S(s) = 0$. □

Definition 9413.187 (No off-balanced tail zero)**.** *The* no off-balanced tail zero *row is*

$$\mathsf{NoOffBalancedTailZero}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{InCritStrip}(s) \times \mathsf{FiniteSieveTailZero}_\zeta(s) \times \mathsf{CofStablePrimeDefect}(s) \to \bot)\,.$$

It rules out a packet in which every finite residual tail is zero while the normalized prime-window defect remains cofinally stable and nonzero.

Theorem 9413.188 (No off-balanced tail zero gives no absorption)**.** *Assume no off-balanced tail zero. Then no far-end absorption holds in normalized prime-window form:*

$$\prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \mathsf{CofStablePrimeDefect}(s) \to \bot)\,.$$

Proof. Let s carry a zeta zero row, a critical-strip row, and a cofinal stable prime defect. By Definition 9413.186, the first two rows supply $\mathsf{FiniteSieveTailZero}_\zeta(s)$. These three rows are exactly the input of Definition 9413.187. Therefore they cannot coexist. □

Corollary 9413.189 (No off-balanced tail zero gives the RH section)**.** *Under located fixed-half collapse, no off-balanced tail zero implies the RH fixed-half section.*

Proof. By Definition 9413.188, no off-balanced tail zero gives the normalized no-absorption row. The RH section then follows from Definition 9413.182. □

Definition 9413.190 (Tail-zero stable-defect correction necessity)**.** *The* tail-zero stable-defect correction necessity *row is*

$$\mathsf{TailZeroWithStableDefectRequiresFarEndCorrection}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{InCritStrip}(s) \times \mathsf{FiniteSieveTailZero}_\zeta(s) \times \mathsf{CofStabl}$$

It isolates the sieve-tower exactness claim: if zero is pushed to every finite residual tail while a normalized finite prime defect remains cofinally nonzero, then the reconciliation would require a hidden far-end correction row.

Theorem 9413.191 (Tail-zero correction plus refusal gives no off-balanced tail zero)**.** *Assume tail-zero stable-defect correction necessity and no hidden far-end correction. Then no off-balanced tail zero holds.*

Proof. Let s carry a critical-strip row, a finite sieve tail-zero tower, and a cofinal stable prime defect. Tail-zero stable-defect correction necessity supplies

$$\mathsf{HiddenFarEndCorrection}(s).$$

No hidden far-end correction sends this row to $\bot$. □

Theorem 9413.192 (Sieve-tail correction refines far-end correction necessity)**.** *Tail-zero stable-defect correction necessity implies far-end correction necessity.*

Proof. Let s carry a zero row, a critical-strip row, and a cofinal stable prime defect. The zero and critical-strip rows give

$$\mathsf{FiniteSieveTailZero}_\zeta(s)$$

by Definition 9413.186. Tail-zero stable-defect correction necessity then supplies $\mathsf{HiddenFarEndCorrection}(s)$, which is precisely the conclusion required by Definition 9413.179. □

Definition 9413.193 (Sieve-tower exactness subpackage)**.** *The* sieve-tower exactness subpackage *is*

$$(\mathsf{FiniteSieveTailZero}_\zeta, \mathsf{TailZeroWithStableDefectRequiresFarEndCorrection}_\zeta, \mathsf{NoHiddenFarEndCorrection}_\zeta).$$

It separates the finite multiplication row, the zeta-specific tail exactness row, and the apophatic refusal row. The finite multiplication row alone carries no fixed-half conclusion; the fixed-half output appears only after the stable-defect incompatibility row is supplied.

Definition 9413.194 (No-far-end proof obligation package)**.** *The* no-far-end proof obligation package *is*

$$(\mathsf{CofStablePrimeDefect}, \mathsf{FarEndCorrectionNecessity}_\zeta, \mathsf{NoHiddenFarEndCorrection}_\zeta, \mathsf{LocatedFixedHalfCollapse}).$$

The first row is the finite prime-window calculation. The second row is the zeta-specific completed-carrier exactness step. The third row is the apophatic refusal. The fourth row converts absence of off-half apartness into the fixed-half output row. The sieve-tower exactness subpackage of Definition 9413.193 is the finite-residual normal form of the second row.

Theorem 9413.195 (No-far-end package normal form)**.** *The no-far-end proof obligation package is a normal form for the far-end version of the RH route.*

Proof. The cofinal stable defect row is obtained from finite prime-window normalization by Definition 9413.167 and Definition 9413.168. Correction necessity and no-hidden correction compose by Definition 9413.181 to give no absorption. Located fixed-half collapse then yields the RH section by Definition 9413.182. The correction-necessity row may also be supplied through the sieve-tail normal form of Definition 9413.192. Thus the package separates the finite calculation, the unclosed analytic exactness step, the apophatic refusal, and the final located-real read. □

Corollary 9413.196 (Defect-zero incompatibility as correction refusal)**.** *Defect-zero incompatibility can be read as the claim that completed zeta zero rows do not admit hidden far-end corrections for cofinally stable prime defects.*

Proof. By Definition 9413.144, no far-end absorption is the completion-boundary form of prime-sieve odd exactness. By Definition 9413.481, prime-sieve odd exactness is defect-zero incompatibility. The present section refines the same row into correction necessity plus refusal of hidden correction. These are not separate RH routes; they are three readings of the same zeta-specific exactness obligation. □

9413.16 From Spectral-Curve Laboratory to the RH Rigidity Route

section 9413.9 provides a finite spectral laboratory: one fixed quartic carrier supports recurrence algebra, companion matrices, spectral polynomials, elliptic normalization, real projection, dominance, and endpoint-zero reads. The lesson for the RH route is not that the quartic proves anything about the zeta function. The lesson is structural. A zero-relevant boundary is not identified by a branch or defect read alone; it is identified only after the dominance, source-compatibility, and noncollapse ledgers have been displayed.

The RH-facing analogue is the route

$$\text{finite spectral defect} \Rightarrow \text{zero incompatibility} \Rightarrow \text{fixed-half rigidity}.$$

The target predicate itself is already stated in Definition 9360.23:

$$\mathsf{RH} := \prod_{s:\mathsf{Complex}^{\uparrow}} (\mathsf{InCritStrip}(s) \times \mathsf{ZetaZero}(s) \to \mathsf{OnCritLine}(s)).$$

This section records the unified-reality reading of the remaining obligation surface. It does not claim the missing analytic bridge.

Definition 9413.197 (RH zero source and fixed-half fiber)**.** *The* RH zero source carrier *is the packet*

$$\mathcal{Z}_\zeta := \sum_{s:\mathsf{Complex}^{\uparrow}} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)).$$

For a row $z = (s; Z_s; C_s)$, the fixed-half fiber adds the critical-line row:

$$\mathcal{H}_\zeta := \sum_{z:\mathcal{Z}_\zeta} \mathsf{OnCritLine}(\pi_s(z)).$$

This is the same constructive target as the RH witness function of Definition 9333.1, expressed as a section over the zero source.

Definition 9413.198 (RH fixed-half section)**.** *The* RH fixed-half section *is a displayed section*

$$\sigma : \mathcal{Z}_\zeta \to \mathcal{H}_\zeta$$

whose projection back to $\mathcal{Z}_\zeta$ is classifier-equal to the identity. Equivalently, it assigns to every

$$s : \mathsf{Complex}^{\uparrow}, \qquad \mathsf{ZetaZero}(s), \qquad \mathsf{InCritStrip}(s)$$

a displayed $\mathsf{OnCritLine}(s)$ *row.*

Theorem 9413.199 (Fixed-half section reads the RH target)**.** *The fixed-half section of Definition 9413.198 is the unified-reality section form of the BEDC RH target.*

Proof. Given RH, send a zero-source row $z = (s; Z_s; C_s)$ to $(z, \mathsf{RH}(s)(C_s, Z_s))$ in $\mathcal{H}_\zeta$. The projection is visibly the original row. In the other direction, a section applied to the row $(s; Z_s; C_s)$ supplies the missing $\mathsf{OnCritLine}(s)$ witness. This is the section form already used in Definition 9333.1. □

Definition 9413.200 (Off-half and prime-window defect rows)**.** *For* $s : \mathsf{Complex}^{\uparrow}$*, write*

$$\delta(s) := \Re(s) - \frac{1}{2}.$$

The off-half row is the apartness witness

$$\mathsf{OffHalf}(s) := \delta(s)\#0.$$

For a nonempty finite prime window S*, the prime-window defect is the finite readout*

$$D_S(s) := \sum_{p \in S} \log |p^{1/2-s}|,$$

as in Definition 9360.24. The existential defect row $\mathsf{PrimeDefect}(s)$ *asserts that some nonempty finite prime window has nonzero defect.*

Theorem 9413.201 (Off-half rows create finite prime defects)**.** *If* $\mathsf{OffHalf}(s)$ *holds and* S *is a nonempty finite prime window, then* $D_S(s)\#0$*.*

Proof. Write $s = 1/2 + \delta + i\tau$. Then

$$p^{1/2-s} = p^{-\delta}p^{-i\tau}, \qquad |p^{1/2-s}| = p^{-\delta}.$$

Thus

$$D_S(s) = -\delta \sum_{p \in S} \log p.$$

Since S is nonempty and every displayed prime has positive logarithm, the sum is positive. Apartness of δ from zero therefore gives apartness of $D_S(s)$ from zero. □

Definition 9413.202 (Defect-zero incompatibility route)**.** *The* defect-zero incompatibility route *is the zeta-specific bridge*

$$\mathsf{DefectZeroIncompatibility}$$

from prime-window defect inside the completed zeta carrier to apartness of the zeta value from zero. It is the same analytic obligation named in Definition 9360.25 and Definition 9246.13. It is not supplied by the definition of $\mathsf{ZetaZero}$*.*

Theorem 9413.203 (Defect-zero incompatibility rules out off-line zeros)**.** *Assume the defect-zero incompatibility route, zero-ledger soundness, and the same-source compatibility rows needed to compare the zero ledger with the prime-window read. Then no row can simultaneously carry*

$$\mathsf{ZetaZero}(s), \qquad \mathsf{InCritStrip}(s), \qquad \mathsf{OffHalf}(s).$$

Proof. An off-half row gives a nonzero finite prime-window defect by Definition 9413.201. The same-source compatibility row places that prime-window defect and the zero ledger in the same completed zeta carrier. Defect-zero incompatibility gives apartness of $\zeta(s)$ from zero, while zero-ledger soundness gives the zero value. This is a contradiction in the shared classifier. □

Definition 9413.204 (Located fixed-half collapse)**.** *The* located fixed-half collapse row *is the classifier transport*

$$\neg\mathsf{OffHalf}(s) \to \mathsf{OnCritLine}(s).$$

It is a located-real row for the specific real part comparison with $1/2$*; it is not a general classical trichotomy principle.*

Theorem 9413.205 (Rigidity package implies the RH section)**.** *Assume defect-zero incompatibility, zero-ledger soundness, same-source compatibility, and located fixed-half collapse. Then the RH fixed-half section exists.*

Proof. Let $z = (s; Z_s; C_s) : \mathcal{Z}_\zeta$. If $\mathsf{OffHalf}(s)$ held, then Definition 9413.203 would give a contradiction. Hence $\neg\mathsf{OffHalf}(s)$. Located fixed-half collapse supplies $\mathsf{OnCritLine}(s)$, so z lifts to $\mathcal{H}_\zeta$. This lift is a section because it preserves the underlying zero-source row. □

Corollary 9413.206 (The analytic content is zero-defect rigidity)**.** *Within the displayed structural rows, the remaining RH-facing analytic content is the incompatibility between zero rows and prime-window defect rows in the completed zeta carrier.*

Proof. The section target is structural by Definition 9413.199. The finite off-half defect read is Definition 9413.201. The step from defect to zero exclusion is exactly Definition 9413.202. Thus the analytic load is not the definition of the zero predicate but the rigidity row excluding zero and prime-window defect from the same completed source. □

Definition 9413.207 (Spectral laboratory pattern)**.** *A* spectral laboratory pattern *is a packet*

$$(\mathcal{O}, \mathsf{Branch}, \mathsf{Dominance}, \mathsf{LocalExpansion}, \mathsf{Interference}, \mathsf{ZeroLaw}, \mathsf{Boundary})$$

where $\mathcal{O}$ is a common spectral carrier, Branch *records ramification,* Dominance *records the zero-relevant active sheets,* LocalExpansion *records local branches,* Interference *records cancellation of active contributions,* ZeroLaw *records the resulting zero readout, and* Boundary *records the scope of the zero law.*

Definition 9413.208 (RH spectral-rigidity pattern)**.** *The* RH spectral-rigidity pattern *is the packet*

$$(\mathcal{Z}_\zeta, \mathsf{PrimeWindow}, \mathsf{PrimeDefect}, \mathsf{DefectZeroIncompatibility}, \mathsf{FixedHalfSection}, \mathsf{Boundary})$$

where $\mathcal{Z}_\zeta$ is the zero source carrier, finite prime windows provide defect readouts, defect-zero incompatibility is the zeta-specific rigidity row, the fixed-half section is the RH witness target, and the boundary records that the rigidity row is an obligation, not a closed theorem in this vision section.

Theorem 9413.209 (The quartic laboratory is an architecture model, not an RH proof)**.** *The quartic spectral carrier of section 9413.9 realizes a finite spectral laboratory pattern that models the architecture of the RH spectral-rigidity route, but it does not imply RH.*

Proof. The quartic carrier supplies a fixed recurrence source, companion matrix, spectral polynomial, elliptic normalization, real projection, dominance ledger, and endpoint-zero law. Its dominance ledger separates the zero-relevant endpoint from other branch collisions, as stated in Definition 9413.99. This is structurally analogous to the RH need to separate prime-visible defect rows that are zero-incompatible from finite reads that are merely visible. But the quartic carrier is fixed and finite, and its endpoint law is scoped by fixed-window restrictions through Definition 9413.105. RH requires a zeta-specific all-zero-source section. Therefore the quartic laboratory supplies proof architecture only. □

Corollary 9413.210 (The quartic carrier supplies no zeta bridge)**.** *The quartic worked example proves the value of a dominance-controlled spectral carrier. It does not prove defect-zero incompatibility for the completed zeta carrier.*

Proof. The quartic carrier contains the rows $(N, D, T(y), \Pi, E, y : E \to \mathbb{P}^1)$ and local endpoint branch data. The RH carrier contains zero-ledger, critical-strip, completed-zeta, prime-window, and fixed-half rows. Transport between these carriers is not supplied. The zeta-specific bridge remains the obligation named in Definition 9413.202. □

Definition 9413.211 (Zeta bridge obligations)**.** *The* zeta bridge obligations *are the following rows:*

ZetaCarrierClosure, PrimeWindowExtraction, OffHalfDefectFormula, DefectZeroIncompatibility, LocatedFixedH

They respectively provide completed zeta value rows, finite prime-window readouts, the off-half defect formula, zero-defect rigidity, and the located read from absence of off-half apartness to the critical line.

Theorem 9413.212 (Zeta bridge obligations close the section target)**.** *If the zeta bridge obligations are all displayed, then the RH fixed-half section exists, hence the RH predicate of Definition 9360.23 holds.*

Proof. The first two obligations place zero rows and finite prime-window rows in one completed zeta carrier. The off-half defect formula supplies prime defect from off-half apartness. Defect-zero incompatibility excludes a zero row with such a defect. Located fixed-half collapse turns absence of off-half apartness into OnCritLine. This is exactly the construction of Definition 9413.205, followed by Definition 9413.199. □

Definition 9413.213 (Finite-window to global boundary)**.** *The* finite-window to global boundary *is the obligation that finite prime-window readouts remain source-compatible with the completed zeta zero ledger and cannot be absorbed by tail, continuation, branch, or classifier transport when the route is lifted to the full zero source.*

Theorem 9413.214 (Finite-window facts alone do not imply RH)**.** *The finite prime-window defect formula alone does not imply RH.*

Proof. The formula says that off-half apartness is visible in a nonempty finite prime window. RH requires a witness that a zeta-zero row in the critical strip lies on the fixed-half fiber. The missing step is the zero-defect incompatibility bridge in the completed zeta carrier, together with the finite-window to global boundary of Definition 9413.213. Without those rows, finite-window defect is only a visible shadow, not a global zero-exclusion theorem. □

Corollary 9413.215 (Fixed-window discipline transfers as a warning)**.** *The fixed-window boundary in the quartic laboratory warns against flattening finite prime-window facts into all-height zeta rigidity.*

Proof. The quartic endpoint law is explicitly scoped to fixed endpoint windows and fixed index sets. Enlarging that scope would require additional uniformity rows. Likewise, RH is a dependent statement over all nontrivial zeta-zero source rows. It requires the global compatibility and zero-defect rigidity ledgers, not only finite-window defect reads. □

Theorem 9413.216 (Spectral-laboratory-to-RH principle)**.** *The unified-reality route from the quartic spectral laboratory to the RH surface is a proof-architecture transfer:*

$$\textit{finite carrier} + \textit{dominance ledger} + \textit{local interference} \Rightarrow \textit{zero law}$$

models the RH obligation shape

$$\textit{zeta carrier} + \textit{prime-window defect} + \textit{defect-zero incompatibility} \Rightarrow \textit{fixed-half section.}$$

The transfer identifies required ledgers; it is not a zeta theorem.

Proof. The finite laboratory side is Definition 9413.209. The section target is Definition 9413.199. The required zeta-specific obligations are Definition 9413.211. The reason finite-window facts do not suffice is Definition 9413.214. Therefore the correct conclusion is an obligation map from the spectral laboratory to the RH route, with defect-zero incompatibility as the central unclosed analytic bridge. □

9413.17 Phase, Transform, and Zeta Ladders

The unified-reality perspective also explains why trigonometry, the circle, Fourier modes, Laplace kernels, elliptic curves, and zeta functions appear in one mathematical corridor. They are not the same object. They are compatible projections of carriers that preserve quadratic norm, phase, exponential transport, spectral decomposition, branch structure, and zero ledgers at different interfaces.

The guiding ladder is

$$\text{triangle} \to \text{circle} \to \text{phase} \to \text{Fourier} \to \text{Laplace/Mellin} \to \text{zeta}.$$

The spectral-curve branch is

$$\text{recurrence} \to \text{spectral polynomial} \to \text{elliptic normalization} \to \text{branch/period/zero reality}.$$

Both ladders meet at the same discipline: a visible readout may expose part of a carrier, but the noncollapse ledger records which scale, phase, branch, convergence, prime, or zero rows remain hidden.

Definition 9413.217 (Quadratic phase carrier)**.** *The* quadratic phase carrier *is the carrier row behind the Pythagorean and unit-circle projections of section 9413.6. It contains a Pythagorean row*

$$a^2 + b^2 \sim c^2$$

and, when $c\#0$, the normalized phase read

$$x = \frac{a}{c}, \qquad y = \frac{b}{c}, \qquad x^2 + y^2 \sim 1.$$

The unit circle is the shape projection. The scale c remains in the fiber and is not recoverable from the circle point alone.

Theorem 9413.218 (Triangle-to-circle projection preserves phase and loses scale)**.** *The passage from a nondegenerate Pythagorean carrier to a unit-circle row preserves the normalized phase but forgets the scale fiber.*

Proof. This is the normalization read of Definition 9413.30 and the lossiness statement of Definition 9413.34. If $c\#0$, the projection sends (a, b, c) to $(a/c, b/c)$. The equation $a^2 + b^2 \sim c^2$ transports to $(a/c)^2 + (b/c)^2 \sim 1$. The value of c is not a coordinate of the circle readout. □

Definition 9413.219 (Circle phase and Fourier reality)**.** *The* circle phase reality *reads the unit circle through a phase parameter*

$$e^{i\theta} = \cos\theta + i\sin\theta.$$

The Fourier reality *reads functions on the circle through the character rows*

$$e^{in\theta}, \qquad n \in \mathbb{Z},$$

and the spectral expansion schema

$$f(\theta) \sim \sum_{n\in\mathbb{Z}} \widehat{f}_n e^{in\theta},$$

whenever the relevant convergence, classifier, and coefficient ledgers are displayed.

Principle 9413.220 (Fourier is the frequency projection of circle phase)**.** *Fourier analysis is not detached from the circle. It is the frequency projection of the same phase carrier: trigonometric coordinates read phase position, while character rows read spectral modes.*

Definition 9413.221 (Laplace extension of Fourier phase)**.** *The* Laplace extension *of Fourier phase replaces the pure phase kernel*

$$e^{-i\omega t}$$

by the damped exponential kernel

$$e^{-st}, \qquad s = \sigma + i\omega,$$

so that

$$e^{-st} = e^{-\sigma t}e^{-i\omega t}.$$

The imaginary part is the Fourier frequency read. The real part is the growth or damping read.

Theorem 9413.222 (Laplace is Fourier with a real bias direction)**.** *The Laplace kernel extends the Fourier phase kernel by adding a real amplitude-bias direction.*

Proof. The displayed factorization

$$e^{-(\sigma+i\omega)t} = e^{-\sigma t}e^{-i\omega t}$$

separates the real damping row from the pure oscillatory phase row. The kernel is therefore a two-coordinate reality: phase frequency and real growth/decay. □

Definition 9413.223 (Arithmetic Laplace carrier)**.** *For a coefficient sequence* $(a_n)_{n\geq 1}$*, the* arithmetic Laplace carrier *is the atomic source*

$$\mu_a := \sum_{n\geq 1} a_n \delta_{\log n}$$

with readout

$$\mathcal{L}[\mu_a](s) := \int e^{-st}\, d\mu_a(t) = \sum_{n\geq 1} a_n n^{-s}$$

on the displayed convergence window. For zeta, $a_n = 1$ *and Definition 9250.8 gives the same discrete Laplace read.*

Theorem 9413.224 (Zeta is an arithmetic Laplace readout)**.** *On its convergence window, the Dirichlet face of* ζ *is the discrete Laplace readout of the atomic source*

$$\mu_\zeta := \sum_{n\geq 1} \delta_{\log n}.$$

Proof. For every $n \geq 1$,

$$n^{-s} = e^{-s \log n}.$$

Therefore

$$\sum_{n \geq 1} n^{-s} = \int e^{-st} \, d\mu_\zeta(t)$$

on any convergence window where the sum and readout are licensed. This is the unified-reality restatement of Definition 9250.8. □

Definition 9413.225 (Multiplicative-prime face of zeta)**.** *The* multiplicative-prime face *of zeta is the Euler-product read*

$$\zeta(s) \sim \prod_p (1 - p^{-s})^{-1}$$

on a displayed right-half-plane window. It records that the additive Dirichlet/Laplace face is compatible with the multiplicative factorization of integers into primes.

Principle 9413.226 (Logarithm transports multiplication to Laplace addition)**.** *The logarithm is the bridge from arithmetic multiplication to the Laplace additive axis:*

$$\log(mn) = \log m + \log n.$$

The zeta carrier consumes this bridge when it reads multiplicative prime factorization as an additive atomic source on the $\log n$ *axis.*

Definition 9413.227 (Prime-channel unitary balance)**.** *For* $s = 1/2 + \delta + i\tau$ *and a displayed prime* p*, the centered prime channel is*

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

It is unitary exactly when the amplitude read satisfies

$$|p^{1/2-s}| = 1.$$

This is the unified-reality form of Definition 9243.5 and Definition 9243.6.

Theorem 9413.228 (The critical line is the prime-channel unitary surface)**.** *For a displayed prime* $p > 1$*,*

$$|p^{1/2-s}| = 1 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Proof. Writing $s = 1/2 + \delta + i\tau$ gives

$$|p^{1/2-s}| = p^{-\delta}.$$

Since $p > 1$, this amplitude is 1 exactly when $\delta = 0$, namely $\Re(s) = 1/2$. This is the prime-channel read of Definition 9243.7. □

Definition 9413.229 (Elliptic spectral branch carrier)**.** *The* elliptic spectral branch carrier *is the geometric branch of the quartic example of section 9413.9. It reads the recurrence and companion-matrix carrier through the normalized curve*

$$E: \quad Y^2 = 4\lambda^3 - 4\lambda + 1, \qquad y = \lambda^2 - \frac{1}{2} + \frac{1}{2}Y.$$

It carries branch, period, real-projection, dominance, and endpoint-zero ledgers.

Principle 9413.230 (Elliptic curves are branched phase carriers)**.** *In this ladder, an elliptic curve is not an ordinary Euclidean ellipse. It is a genus-one branched phase carrier: the circle gives a single phase period, while the elliptic carrier exposes branch structure and two-period geometry. Theta and modular expansions are therefore the elliptic analogue of Fourier frequency reads on the circle whenever the corresponding ledgers are displayed.*

Theorem 9413.231 (Two ladders meet in spectral-zero reality)**.** *The transform ladder and the elliptic spectral ladder meet at zero-reality interfaces: transform kernels organize arithmetic and analytic readouts, while spectral curves organize branch and dominance readouts. Both require noncollapse ledgers before zero statements can be transported.*

Proof. On the transform side, Definition 9413.224 and Definition 9413.228 show how the zeta carrier reads arithmetic data through Laplace and prime-channel phase. On the spectral side, Definition 9413.106 shows how recurrence algebra, elliptic normalization, dominance, and endpoint zeros are projections of one fixed quartic carrier. In both cases, a visible readout is insufficient by itself: the zeta route needs the zero-defect rigidity boundary of section 9413.16, while the quartic route needs dominance and fixed-window ledgers. Thus the ladders meet as common-carrier disciplines for zero reality, not as a single untyped identification. □

Theorem 9413.232 (Phase-transform-zeta ladder principle)**.** *Triangles, circles, Fourier modes, Laplace kernels, zeta, and elliptic spectral curves form a compatible carrier ladder:*

geometric shape ↔ algebraic equation ↔ phase and transform kernel ↔ spectral decomposition ↔ zero reality.

The compatibility is ledgered projection and continuation, not object-level collapse.

Proof. The triangle-to-circle part is Definition 9413.218. The circle-to-Fourier and Fourier-to-Laplace parts are Definition 9413.220 and Definition 9413.222. The Laplace-to-zeta part is Definition 9413.224, with the prime face governed by Definition 9413.226. The elliptic spectral branch is Definition 9413.229. Finally, Definition 9413.231 records the zero-reality interface and its noncollapse boundary. □

9413.18 Generalized Continuation Chain

The phase-transform ladder of section 9413.17 must be read with a precise boundary. It is a generalized analytic continuation chain in the sense of Definition 9413.12; it is not a claim that every arrow is ordinary holomorphic continuation of one complex function across a larger domain. The chain transports structure from algebraic and geometric realities into phase, spectral, transform, and arithmetic realities. Only specific subrows, such as the $\mathsf{ZetaCont}^{\uparrow}$ application of chapter 9342, are ordinary analytic-continuation rows.

Definition 9413.233 (Reality-continuation chain)**.** *A* reality-continuation chain *is a finite sequence of carriered interfaces*

$$\mathcal{R}_0 \xrightarrow{\mathsf{GenCont}_{01}} \mathcal{R}_1 \xrightarrow{\mathsf{GenCont}_{12}} \cdots \xrightarrow{\mathsf{GenCont}_{(k-1)k}} \mathcal{R}_k$$

where every edge is a generalized continuation packet of Definition 9413.12. The edge may be algebraic normalization, geometric parametrization, character decomposition, exponential-kernel transport, logarithmic Laplace transport, or ordinary holomorphic continuation, provided the source, classifier, projection, transport, and boundary ledgers are displayed.

Definition 9413.234 (Narrow analytic-continuation row)**.** *A* narrow analytic-continuation row *is an edge whose source and target are holomorphic-domain readouts of one complex function, with overlap, identity-theorem, uniqueness, pole, and branch ledgers displayed. This is the ordinary* $\mathsf{AnaCont}^{\uparrow}$ *situation discussed in Definition 9413.13 and realized for zeta by Definition 9342.8.*

Principle 9413.235 (Generalized continuation is wider than holomorphic continuation)**.** *Every narrow analytic-continuation row is a generalized continuation row when its carrier and ledgers are displayed. The converse is not valid: normalization, phase parametrization, Fourier character projection, Laplace damping, Mellin logarithmic transport, Euler product regrouping, and spectral-curve normalization are generalized continuation rows but are not, merely by that fact, holomorphic continuation steps.*

Definition 9413.236 (Phase-transform arithmetic continuation)**.** *The* phase-transform arithmetic continuation *is the chain*

quadratic norm $\to$ *unit phase* $\to$ *frequency decomposition* $\to$ *damped frequency kernel* $\to$ *arithmetic log-spectrum.*

Its displayed formula rows are

$$a^2 + b^2 = c^2, \qquad x^2 + y^2 = 1, \qquad x + iy = e^{i\theta},$$

$$e^{-i\omega t}, \qquad e^{-st} = e^{-\sigma t} e^{-i\omega t}, \qquad n^{-s} = e^{-s \log n}.$$

The zeta endpoint reads

$$\zeta(s) = \sum_{n \geq 1} n^{-s} = \sum_{n \geq 1} e^{-s \log n}$$

on the displayed convergence window.

Theorem 9413.237 (The phase-transform chain is generalized continuation)**.** *The phase-transform arithmetic continuation of Definition 9413.236 is a generalized continuation chain, not a stepwise holomorphic continuation chain.*

Proof. The Pythagorean-to-circle edge is scale normalization by Definition 9413.218. The circle-to-Fourier edge is character projection by Definition 9413.220. The Fourier-to-Laplace edge adds the real amplitude-bias coordinate by Definition 9413.222. The Laplace-to-Mellin arithmetic edge is logarithmic transport by Definition 9413.1137. The endpoint zeta read is the atomic logarithmic Laplace readout of Definition 9413.224. These edges transport carriered structure across realities. They are not all holomorphic-domain extension operations of one complex function. □

Definition 9413.238 (Transported invariant)**.** *Let* $\mathsf{GenCont}_{i \to j}$ *be a generalized continuation packet. A* transported invariant *from* $\mathcal{R}_i$ *to* $\mathcal{R}_j$ *is a target-side row* P_j *equipped with a transport certificate from a source-side row* P_i *through the packet ledger. The target row need not have the same formula as* P_i*. It must instead be the certified image of the same carriered constraint under the displayed projection and transport rows.*

Definition 9413.239 (Zeta transported-invariant packet)**.** *The* zeta transported-invariant packet *is the collection of zeta rows corresponding to the phase-transform chain:*

source structure	*zeta-side transported row*
scale normalization	*centering at* $1/2$
unit phase	$\lvert p^{1/2-s}\rvert = 1$ *on the critical line*
frequency clock	$e^{-i\tau \log p}$
Laplace damping	$p^{-\delta}$ *or* $e^{-\delta x}$
spectral decomposition	*Dirichlet coefficients and Euler factors*
local-to-global assembly	*prime-local product to zeta readout*
symmetry	*functional-equation mirror*
zero boundary	$\zeta(s) = 0$ *as global cancellation*
defect boundary	*finite prime-window defect*

Here $s = 1/2 + \delta + i\tau$*, and the prime-channel row is*

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

Principle 9413.240 (Zeta inherits transported invariants, not literal shapes)**.** *If zeta is read as the arithmetic log-spectrum endpoint of the phase-transform chain, then zeta must carry transported versions of normalization, unit phase, frequency, damping, local-to-global assembly, symmetry, zero cancellation, and boundary rigidity. It need not contain the literal equation* $a^2 + b^2 = c^2$*, a Euclidean circle, or an ordinary Fourier series as object-level substructures.*

Definition 9413.241 (Zeta zero as global cancellation)**.** *A* zeta zero as global cancellation *is a zero row*

$$\zeta(s) = 0$$

read after the Dirichlet, Euler, continuation, and boundary ledgers have assembled nonzero local phase and amplitude information into one global zeta carrier. It is not a claim that each local term n^{-s} *or each prime channel* p^{-s} *vanishes.*

Theorem 9413.242 (Prime-unitary balance is the transported unit-phase row)**.** *For a displayed prime* $p > 1$*, the unit-phase condition transported to the centered prime channel is*

$$\lvert p^{1/2-s}\rvert = 1 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Proof. This is Definition 9413.228. With $s = 1/2 + \delta + i\tau$,

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}$$

and hence

$$\lvert p^{1/2-s}\rvert = p^{-\delta}.$$

Since $p > 1$, this amplitude is 1 exactly when $\delta = 0$, which is the critical-line equation $\Re(s) = 1/2$. □

Definition 9413.243 (Defect-boundary transport)**.** *The* defect-boundary transport *is the RH-facing analogue of the dominance boundary in the quartic spectral carrier. In the quartic row, zero-relevant boundary data is not the discriminant alone; it is the maximal-modulus branch competition recorded in Definition 9413.99. In the zeta row, the corresponding boundary obligation is the passage from finite prime-window defect to zero incompatibility, using Definition 9360.24 and Definition 9360.25.*

Theorem 9413.244 (RH is transported unitary zero-rigidity)**.** *Under the zeta transported-invariant packet and the required defect-boundary transport, the RH target of Definition 9360.23 is the assertion that nontrivial zeta zeros in the critical strip occur only on the transported unitary balance surface.*

Proof. The transported unitary surface is the critical line by Definition 9413.242. The zero rows are global cancellation rows in the sense of Definition 9413.241. The missing analytic content is not the existence of the critical line; it is the rigidity statement that off-unitary amplitude bias cannot coexist with a zero row inside the completed zeta carrier. This is exactly the defect-zero route isolated in Definition 9413.202 and the section target of Definition 9413.199. □

Theorem 9413.245 (Generalized continuation chain boundary principle)**.** *The chain from quadratic norm through phase, Fourier, Laplace/Mellin, and zeta is valid only as a ledgered reality-continuation chain. It licenses transported invariants; it does not license literal identification of triangles, circles, Fourier series, Laplace kernels, and zeta.*

Proof. By Definition 9413.235, generalized continuation is broader than ordinary holomorphic continuation but remains ledger-bound. By Definition 9413.238, every inherited property requires a transport certificate. By Definition 9413.240, the zeta endpoint receives transported versions of phase, amplitude, local-to-global, symmetry, zero, and defect-boundary rows, not literal copies of the source shapes. Therefore the chain is a precise continuation discipline rather than an analogy or an object-level collapse. □

9413.19 Inherited Invariants of the Zeta Endpoint

section 9413.18 separates the generalized continuation chain from ordinary holomorphic continuation. The present section records the carrier-level invariants that reach the zeta endpoint through that chain. The point is not that zeta contains a literal triangle, circle, Euclidean ellipse, Fourier series, or Laplace transform as a visible subobject. The point is that normalization, unit phase, frequency, damping, local-to-global assembly, symmetry, cancellation, and defect-rigidity have transported zeta-side readouts.

Definition 9413.246 (Zeta construction chain)**.** *The* zeta construction chain *is the reality-continuation chain*

$$\mathsf{Pyth} \to \mathsf{Circle} \to \mathsf{Phase} \to \mathsf{Fourier} \to \mathsf{Laplace/Mellin} \to \mathsf{Zeta}.$$

Its basic displayed rows are

$$a^2 + b^2 = c^2, \qquad x^2 + y^2 = 1, \qquad x + iy = e^{i\theta},$$
$$e^{in\theta}, \qquad e^{-st}, \qquad \zeta(s) = \sum_{n\geq 1} n^{-s} = \sum_{n\geq 1} e^{-s\log n}.$$

The final equality is the arithmetic Laplace readout of Definition 9413.224.

Definition 9413.247 (Shared invariant along the zeta chain)**.** *A* shared invariant *of the zeta construction chain is a family of rows*

$$I_{\mathrm{Pyth}}, I_{\mathrm{Circle}}, I_{\mathrm{Fourier}}, I_{\mathrm{Laplace}}, I_{\mathrm{Zeta}}$$

together with transport ledgers between adjacent realities. The transported zeta row may change formula. It must remain the certified target-side read of the same carrier-level constraint in the sense of Definition 9413.238.

Definition 9413.248 (Zeta inherited invariant packet)**.** *The* zeta inherited invariant packet *is*

$$\mathsf{ZetaInv}(s) := (\mathsf{Center}, \mathsf{Unitary}, \mathsf{Phase}, \mathsf{Bias}, \mathsf{LocalGlobal}, \mathsf{Symmetry}, \mathsf{Cancellation}, \mathsf{Defect}).$$

For $s = 1/2 + \delta + i\tau$, its prime-channel readout is

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

The eight rows are:

invariant	*zeta-side read*
Center	$\frac{1}{2}$
Unitary	$\lvert p^{1/2-s}\rvert = 1$
Phase	$e^{-i\tau \log p}$
Bias	$p^{-\delta}$
LocalGlobal	$\sum n^{-s} \leftrightarrow \prod_p (1-p^{-s})^{-1}$
Symmetry	$J(s) = 1 - \overline{s}$
Cancellation	$\zeta(s) = 0$
Defect	$D_S(s)$

where $D_S(s)$ is the finite prime-window defect of Definition 9360.24.

Theorem 9413.249 (Scale normalization transports to centered prime channels)**.** *The Pythagorean scale-normalization row transports to the zeta centered prime channel*

$$p^{-s} \mapsto p^{1/2-s}.$$

Proof. In the Pythagorean carrier, a nonzero scale c licenses the normalized shape read

$$(a, b, c) \mapsto (a/c, b/c),$$

as in Definition 9413.218. In the zeta prime-local carrier, the raw channel is p^{-s} and the fixed-half center is $1/2$. Centering by that normalization produces

$$p^{1/2} p^{-s} = p^{1/2-s}.$$

This is a transported normalization row, not a literal Pythagorean equation inside zeta. □

Corollary 9413.250 (Zeta has a centered-channel read)**.** *When zeta is read as the arithmetic endpoint of the construction chain, its prime-local face must expose the centered channel*

$$U_p(s) := p^{1/2-s}$$

whenever critical-line and prime-channel comparisons are being made.

Proof. The centered channel is the zeta-side read of the normalization invariant by Definition 9413.249. It is the same channel used by Definition 9243.5. □

Definition 9413.251 (Zeta unitary balance)**.** *The* zeta unitary balance *row is the prime-channel predicate*

$$\mathsf{UnitaryZeta}(s) :\Longleftrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ \lvert p^{1/2-s}\rvert \sim 1.$$

It is the unified-reality reading of Definition 9243.6.

Theorem 9413.252 (Unit norm transports to the critical line)**.** *For a displayed prime row $p > 1$,*

$$|p^{1/2-s}| = 1 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Consequently, $\mathsf{UnitaryZeta}(s)$ *is the prime-channel read of the critical line.*

Proof. This is the transported unit-phase theorem Definition 9413.242, equivalently the prime-unitary balance theorem Definition 9243.7. If $s = 1/2 + \delta + i\tau$, then

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}$$

and hence $|p^{1/2-s}| = p^{-\delta}$. For $p > 1$, this has modulus 1 exactly when $\delta = 0$. □

Definition 9413.253 (Prime phase clock and centered bias)**.** *For $s = \sigma + i\tau$, the* prime phase clock *is*

$$\theta_p(s) := -\tau \log p.$$

The raw and centered prime channels decompose as

$$p^{-s} = p^{-\sigma} e^{i\theta_p(s)}, \qquad p^{1/2-s} = p^{1/2-\sigma} e^{i\theta_p(s)}.$$

If $\sigma = 1/2 + \delta$, the centered amplitude bias is

$$p^{1/2-\sigma} = p^{-\delta}.$$

Theorem 9413.254 (Fourier frequency transports to logarithmic prime clocks)**.** *The Fourier phase row transports to the zeta prime-channel phase*

$$e^{-i\tau \log p},$$

so the prime-local frequency is read through $\log p$.

Proof. The Fourier kernel has phase form $e^{-i\omega t}$. A prime-local zeta term satisfies

$$p^{-s} = e^{-s \log p}.$$

Writing $s = \sigma + i\tau$ gives

$$e^{-s \log p} = e^{-\sigma \log p} e^{-i\tau \log p}.$$

The second factor is the Fourier phase row with clock $\log p$. □

Theorem 9413.255 (Laplace damping transports to off-critical bias)**.** *The Laplace damping factor transports to the zeta amplitude read*

$$n^{-\sigma} = e^{-\sigma \log n},$$

and the centered prime bias is

$$p^{-\delta}$$

for $s = 1/2 + \delta + i\tau$.

Proof. Put $t = \log n$ in the Laplace kernel. Then

$$e^{-st} = e^{-s\log n} = n^{-s}.$$

For $s = \sigma + i\tau$,

$$n^{-s} = e^{-\sigma\log n}e^{-i\tau\log n}.$$

The first factor is the damping or growth read. After centering at 1/2, the prime-channel form is

$$p^{1/2-s} = p^{-\delta}e^{-i\tau\log p}.$$

□

Corollary 9413.256 (The critical line is the zero-bias surface)**.** *The critical line is the surface on which the centered prime-channel amplitude bias disappears:*

$$\delta = 0 \quad \Longleftrightarrow \quad p^{-\delta} = 1.$$

Proof. This follows immediately from Definition 9413.255. □

Definition 9413.257 (Local-global zeta factorization)**.** *The* local-global zeta factorization *is the pair of readouts*

$$\zeta(s) = \sum_{n\geq 1} n^{-s}$$

and

$$\zeta(s) \sim \prod_p (1-p^{-s})^{-1}$$

on the displayed window where convergence, coefficient, and factorization ledgers license the comparison. The first is the global Dirichlet-sum face. The second is the prime-local Euler-product face.

Theorem 9413.258 (Frequency decomposition transports to prime-local factorization)**.** *The local frequency-decomposition invariant transports to the zeta Euler-product face:*

$$\zeta(s) \sim \prod_p (1-p^{-s})^{-1}.$$

Proof. Fourier analysis decomposes a global readout into character modes. In the arithmetic reality, unique prime factorization organizes integer terms into prime-local channels. On the window where absolute convergence and regrouping are licensed, the Dirichlet sum becomes

$$\sum_{n\geq 1} n^{-s} \sim \prod_p (1+p^{-s}+p^{-2s}+\cdots) = \prod_p (1-p^{-s})^{-1}.$$

Thus the zeta carrier has both the global sum face and the prime-local product face. □

Definition 9413.259 (Functional symmetry fixed surface)**.** *The* functional symmetry fixed surface *is the fixed set of*

$$J(s) := 1 - \overline{s}.$$

It satisfies

$$J(s) = s \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Theorem 9413.260 (Fourier symmetry transports to the zeta mirror)**.** *The circle/Fourier symmetry row transports to the zeta functional-equation mirror, whose conjugate-fixed form has fixed surface*

$$\Re(s) = \frac{1}{2}.$$

Proof. Circle phase has conjugation symmetry

$$e^{i\theta} \mapsto e^{-i\theta},$$

and Fourier modes carry positive and negative frequency symmetry. The completed zeta carrier has the functional-equation mirror relating s and $1 - s$. In conjugate-fixed form this is

$$J(s) = 1 - \overline{s}.$$

Writing $s = \sigma + i\tau$, the fixed condition gives

$$\sigma + i\tau = 1 - \sigma + i\tau,$$

and therefore $\sigma = 1/2$. □

Theorem 9413.261 (Zeta zeros are global cancellation rows)**.** *A zeta zero row records global cancellation of the assembled zeta readout. It does not assert that the local terms n^{-s} or the prime-local channels p^{-s} vanish.*

Proof. For every $n \geq 1$,

$$n^{-s} = e^{-s \log n}.$$

The local exponential term is nonzero wherever the exponential carrier is defined. Thus a zero row

$$\zeta(s) = 0$$

must be read as cancellation in the globally assembled Dirichlet/Euler/continued carrier, as already isolated in Definition 9413.241. □

Theorem 9413.262 (Off-critical bias produces finite prime-window defect)**.** *If $s = 1/2 + \delta + i\tau$, $\delta \# 0$, and S is a nonempty finite prime window, then the prime-window defect*

$$D_S(s) := \sum_{p \in S} \log |p^{1/2-s}|$$

is apart from zero.

Proof. For every $p \in S$,

$$|p^{1/2-s}| = p^{-\delta}.$$

Therefore

$$D_S(s) = \sum_{p \in S} \log(p^{-\delta}) = -\delta \sum_{p \in S} \log p.$$

Since S is nonempty and each displayed prime satisfies $p > 1$, the sum of logarithms is positive. Apartness of δ from zero gives apartness of $D_S(s)$ from zero. This is the unified-reality read of Definition 9413.201. □

Theorem 9413.263 (Zeta carries the inherited invariant packet)**.** *When zeta is read as the endpoint of the zeta construction chain, its carrier carries the invariant packet* $\mathsf{ZetaInv}(s)$ *wherever the corresponding source, projection, convergence, and boundary ledgers are displayed.*

Proof. The center row is Definition 9413.249. The unitary row is Definition 9413.252. The phase row is Definition 9413.254. The bias row is Definition 9413.255. The local-global row is Definition 9413.258. The symmetry row is Definition 9413.260. The cancellation row is Definition 9413.261. The defect row is Definition 9413.262. These are the eight components of Definition 9413.248. □

Definition 9413.264 (RH closure of inherited invariants)**.** *The* RH closure of inherited invariants *is the section target*

$$\prod_s \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{UnitaryZeta}(s)\right).$$

Using the critical-line equivalence of Definition 9413.252, this is the unitary-balance form of the BEDC RH target Definition 9360.23.

Theorem 9413.265 (RH reads as closure of inherited unitary balance)**.** *The RH section target admits a unitary-balance section form: a critical strip zero row is required to supply the inherited unitary-balance row.*

Proof. The BEDC RH target is the section form of Definition 9413.199. By Definition 9413.252, the prime-unitary balance row is equivalent to $\mathsf{OnCritLine}(s)$ on the displayed prime-channel packet. Therefore the RH section can be read as

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{UnitaryZeta}(s).$$

This is only a section-form reading of the target. It does not supply the missing defect-zero incompatibility route. □

Theorem 9413.266 (Inherited-invariant principle for zeta)**.** *The common structure carried by triangles, circles, Fourier modes, Laplace/Mellin kernels, elliptic spectral carriers, and zeta reaches the zeta endpoint as transported invariant data:*

normalization, unitarity, phase, damping, local-global, symmetry, cancellation, defect-rigidity.

The zeta-side forms are

$$p^{1/2-s}, \quad |p^{1/2-s}| = 1, \quad e^{-i\tau \log p}, \quad p^{-\delta}, \quad \sum n^{-s} \leftrightarrow \prod_p (1-p^{-s})^{-1}, \quad J(s) = 1-\overline{s}, \quad \zeta(s) = 0, \quad D_S(s).$$

Proof. This is the assembled content of Definition 9413.263. The quartic spectral row adds the boundary warning: visible branch/defect data is not enough without a zero-relevant boundary ledger, as recorded in Definition 9413.243. Thus the invariant package is a disciplined transport ledger, not an analogy and not a completed RH proof. □

9413.20 Ontic Triple Invariants and the Zeta Projection

The inherited-invariant packet of section 9413.19 can be compressed one level further. The recurring carrier data have three ontic roles: atom, time, and symmetry. Zeta is not the total ontology of mathematics. It is the arithmetic-spectral projection in which these three roles become prime-local factors, logarithmic Laplace time, functional symmetry, and a zero ledger. The RH-facing target is the zero-fiber fixedness of this projection.

Definition 9413.267 (Ontic invariant packet)**.** *An* ontic invariant packet *is a triple*

$$\mathcal{I} := (\mathsf{Atom}, \mathsf{Time}, \mathsf{Sym}).$$

The row Atom *records irreducible generated distinction. The row* Time *records generation order, continuation, filtration, or spectral-frequency direction. The row* Sym *records classifier preservation, duality, involution, or a fixed-surface structure. Different mathematical realities need not share the same objects; they may share the same ontic triple through different projections.*

Definition 9413.268 (Atom row)**.** *For a reality interface $\mathcal{R}$, an* atom row

$$\mathsf{Atom}_{\mathcal{R}}(a)$$

asserts that a is irreducible with respect to the displayed generation law of $\mathcal{R}$, or that every permitted factorization of a is trivial in that reality. Typical reads include prime numbers in multiplicative arithmetic, irreducible polynomials in a polynomial reality, branch generators in a curve reality, and eigenmodes or spectral branches in a spectral reality.

Definition 9413.269 (Time row)**.** *For a reality interface $\mathcal{R}$, a* time row *is an ordered, continued, filtered, or frequency parameter through which generated data are read. Examples include* Cont *and history continuation in the BEDC kernel, regular-modulus depth in located analysis, recurrence index m, Fourier phase or frequency parameters, Laplace time t, and the zeta logarithmic axis* $\log n$.

Definition 9413.270 (Symmetry row)**.** *For a reality interface $\mathcal{R}$, a* symmetry row *is a classifier-preserving involution, duality, reflection, or transport operation*

$$J_{\mathcal{R}} : \mathcal{R} \to \mathcal{R}$$

with fixed locus

$$\mathsf{Fix}(J_{\mathcal{R}}) := \{x : J_{\mathcal{R}}(x) \sim x\}.$$

The fixed locus is a balance surface only relative to the displayed classifier and transport ledger.

Definition 9413.271 (Zeta ontic projection carrier)**.** *The* zeta ontic projection carrier *is the packet*

$$\mathcal{O}_{\zeta}^{\mathrm{ontic}} := (\mathsf{PrimeAtoms}, \mathsf{LogTime}, \mathsf{LaplaceReadout}, \mathsf{EulerProduct}, \mathsf{FunctionalSymmetry}, \mathsf{ZeroLedger}).$$

Its three ontic projections are

$$\pi_{\mathrm{Atom}}(\mathcal{O}_{\zeta}^{\mathrm{ontic}}) = \prod_p (1 - p^{-s})^{-1},$$

$$\pi_{\mathrm{Time}}(\mathcal{O}_{\zeta}^{\mathrm{ontic}}) = e^{-s \log n},$$

and

$$\pi_{\mathrm{Sym}}(\mathcal{O}_{\zeta}^{\mathrm{ontic}}) = J(s) = 1 - \overline{s}.$$

Theorem 9413.272 (Zeta carries the ontic triple projection)**.** *The zeta ontic projection carrier carries atom, time, and symmetry projection rows.*

Proof. The atom projection is the Euler-product face

$$\zeta(s) \sim \prod_p (1 - p^{-s})^{-1},$$

which exposes prime-local arithmetic atoms as in Definition 9413.257. The time projection is the logarithmic Laplace row

$$n^{-s} = e^{-s \log n},$$

which is the arithmetic readout of Definition 9413.224. The symmetry projection is the functional mirror

$$J(s) = 1 - \overline{s},$$

whose fixed surface is read in Definition 9413.259. Thus the same zeta carrier supports the three ontic roles. □

Corollary 9413.273 (Zeta is not only a value function)**.** *The readout $\zeta(s)$ is one projection of $\mathcal{O}_\zeta^{\text{ontic}}$, not the complete carrier. The carrier also exposes prime atoms, logarithmic time, Euler-product factorization, functional symmetry, and zero-ledger rows.*

Proof. This is immediate from Definition 9413.272. A function value read does not by itself display the atom, time, symmetry, and zero-ledger rows that make the zeta carrier relevant to the RH route. □

Definition 9413.274 (Prime atom and zero resonance)**.** *A* prime atom *is a displayed row*

$$\mathsf{PrimeAtom}(p), \qquad p \in \mathsf{Prime}^{\uparrow},$$

expressing irreducibility in the natural-number multiplicative reality. A zero resonance *is a displayed row*

$$\mathsf{ZeroRes}(s) := \mathsf{ZetaZero}(s), \qquad s \in \mathsf{Complex}^{\uparrow}.$$

The rows live in different types. They may be coupled by the zeta carrier, but they are not the same object.

Theorem 9413.275 (Prime atoms and zero resonances are dual projections)**.** *Prime atoms and zero resonances are different typed projections of the zeta carrier: primes are arithmetic-local atoms, while zeros are spectral cancellation rows.*

Proof. Prime atoms appear in the Euler-product face

$$\prod_p (1 - p^{-s})^{-1}.$$

They belong to the arithmetic local-factor reality. Zero resonances appear in the global value ledger

$$\zeta(s) = 0.$$

They belong to the spectral cancellation reality described in Definition 9413.261. Since $p : \mathsf{Prime}^{\uparrow}$ and $s : \mathsf{Complex}^{\uparrow}$ have different carrier types and classifiers, they cannot be identified. They are coupled only through the common zeta carrier and its transform ledgers. □

Corollary 9413.276 (Prime-zero language is projection-relative)**.** *Statements relating primes and zeros must be read as carrier-coupling statements, not as type-level identifications. The precise form is: prime rows are arithmetic atoms; zero rows are spectral resonances; the zeta carrier transports information between their realities.*

Proof. This is the typed consequence of Definition 9413.275 and the projection discipline of Definition 9413.24. □

Definition 9413.277 (Ontic balance)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the centered prime channel is

$$U_p(s) := p^{1/2-s}.$$

The ontic balance *row is*

$$\mathsf{OnticBalance}(s) :\Longleftrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(s)| \sim 1.$$

It is the atom-time-symmetry balance read of the zeta prime-channel projection.

Theorem 9413.278 (Ontic balance is the critical line surface)**.** *For a displayed prime row $p > 1$,*

$$|U_p(s)| = 1 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Thus $\mathsf{OnticBalance}(s)$ *is the critical-line surface in the prime-channel projection.*

Proof. This is the same prime-unitary computation as Definition 9413.252. Writing $s = \sigma + i\tau$ gives

$$U_p(s) = p^{1/2-\sigma} e^{-i\tau \log p}.$$

The phase factor has modulus one, so

$$|U_p(s)| = p^{1/2-\sigma}.$$

For $p > 1$, this is 1 exactly when $\sigma = 1/2$. □

Corollary 9413.279 (Critical line as ontic balance surface)**.** *The critical line is not defined by the zero predicate. It is the fixed balance surface obtained from prime atoms, logarithmic time phase, and functional symmetry.*

Proof. The critical-line surface is the same as the ontic balance surface by Definition 9413.278. The zero predicate is a separate ledger read over the completed zeta carrier. □

Definition 9413.280 (Zero-fiber fixedness)**.** *The* zero-fiber fixedness *target is the section form*

$$\prod_{s:\mathsf{Complex}^{\uparrow}} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnticBalance}(s)\right).$$

It is the ontic-balance reading of the RH target, not an independent proof of that target.

Theorem 9413.281 (Zero-fiber fixedness is the ontic section form of RH)**.** *Zero-fiber fixedness is the atom-time-symmetry section form of the BEDC RH target.*

Proof. The BEDC RH predicate is stated in Definition 9360.23 and read as a fixed-half section in Definition 9413.199. By Definition 9413.278, $\mathsf{OnticBalance}(s)$ is the critical-line row in the prime-channel projection. Replacing the target row $\mathsf{OnCritLine}(s)$ by this transport-equivalent balance row gives the displayed section form. This is a reformulation of the obligation surface only. □

Definition 9413.282 (Ontic collapse modes)**.** *The three collapse modes associated with the ontic packet are:*

$$\mathsf{TimeCollapse} := (\exists x,\ P(x)) \to \sum_x P(x),$$

$$\mathsf{SymCollapse} := (x \sim y) \to ([x] = [y]),$$

and

$$\mathsf{DistCollapse} := (P \leftrightarrow Q) \to (P = Q).$$

They name, respectively, an unaudited collapse of provenance time, an unaudited collapse of classifier fibers, and an unaudited collapse of predicate distinction.

Theorem 9413.283 (Excluded axioms are collapse modes)**.** *Read through the ontic packet,* Classical.choice *is a time/provenance collapse,* Quot.sound *is a symmetry/fiber collapse, and* propext *is a distinction collapse.*

Proof. The structural background is Definition 9387.5. Choice accepts a term-level existential as if a witness and its generating path had been supplied, so it collapses provenance time. Quot.sound accepts a classifier relation as quotient equality, so it collapses the fiber structure of symmetry. propext accepts bidirectional transport between propositions as equality of proposition objects, so it collapses predicate distinction. The axiom-purity gate described in Definition 9387.10 refuses exactly these unaudited closures at the public proof surface. □

Corollary 9413.284 (Axiom-purity protects the ontic ledgers)**.** *BEDC's axiom-purity discipline preserves the three ledger distinctions: generated provenance, classifier-fiber symmetry, and predicate distinction.*

Proof. By Definition 9413.283, the three forbidden dependencies collapse exactly those ledgers. Refusing them keeps the ontic packet from being silently compressed. □

Definition 9413.285 (Ontic defect)**.** *The* ontic defect *row is the apartness form of failure of ontic balance:*

$$\delta(s) := \Re(s) - \frac{1}{2}, \qquad \mathsf{OnticDefect}(s) := \delta(s)\#0.$$

For a nonempty finite prime window S, the defect readout is

$$D_S(s) := \sum_{p \in S} \log |p^{1/2-s}|.$$

Theorem 9413.286 (Ontic defect is visible in finite prime windows)**.** *If* $\mathsf{OnticDefect}(s)$ *holds and S is a nonempty finite prime window, then*

$$D_S(s)\#0.$$

Proof. Write $s = 1/2 + \delta + i\tau$. Then

$$|p^{1/2-s}| = p^{-\delta}.$$

Therefore

$$D_S(s) = \sum_{p \in S} \log(p^{-\delta}) = -\delta \sum_{p \in S} \log p.$$

If S is nonempty and every displayed prime satisfies $p > 1$, the logarithmic sum is positive. Apartness of δ from zero gives apartness of $D_S(s)$ from zero. This is the ontic reading of Definition 9413.262. □

Definition 9413.287 (Ontic defect-zero incompatibility)**.** *The* ontic defect-zero incompatibility *target is*

$$\prod_s (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \mathsf{OnticDefect}(s) \to \bot).$$

It is the ontic form of the defect-zero incompatibility route of Definition 9413.202.

Theorem 9413.288 (Ontic incompatibility closes the section target)**.** *Assume ontic defect-zero incompatibility and the located row that turns absence of ontic defect into ontic balance. Then the zero-fiber fixedness section target is obtained.*

Proof. Let s carry rows

$$\mathsf{ZetaZero}(s), \qquad \mathsf{InCritStrip}(s).$$

To produce the zero-fiber fixedness output it suffices to produce $\mathsf{OnticBalance}(s)$. Ontic defect-zero incompatibility rules out the simultaneous presence of the two zero rows and $\mathsf{OnticDefect}(s)$. Hence the defect row is absent. The located row converts this absence into $\mathsf{OnticBalance}(s)$. This is the same proof skeleton as Definition 9413.203, now expressed in atom-time-symmetry language. □

Corollary 9413.289 (The remaining ontic boundary)**.** *In the ontic projection language, the remaining RH-facing boundary is to show that a zeta zero row in the critical strip cannot coexist with an atom-time-symmetry defect inside the completed zeta carrier.*

Proof. The finite prime-window visibility of the defect is Definition 9413.286. The missing zero-relevance bridge is exactly the incompatibility target of Definition 9413.287. Thus the boundary is a zero-relevant carrier compatibility problem, not the mere observation that off-line prime-channel bias can be measured. □

Theorem 9413.290 (Spectral dominance warns against raw-defect collapse)**.** *The quartic spectral example shows why visible defect data cannot be used as a zero-relevant boundary without an active dominance ledger.*

Proof. In the quartic carrier, branch or discriminant data alone does not determine the endpoint-zero boundary. The zero-relevant boundary passes through maximal-modulus branch competition, as recorded in Definition 9413.99 and Definition 9413.1144. The zeta analogue is that a finite prime-window defect is only a visible defect row until a completed-carrier incompatibility with the zero ledger is supplied. □

Theorem 9413.291 (Zeta ontic projection principle)**.** *Zeta is the arithmetic-spectral projection of the ontic packet* $(\mathsf{Atom}, \mathsf{Time}, \mathsf{Sym})$*. Its atom projection is the prime-local Euler face, its time projection is the logarithmic Laplace/Mellin readout, and its symmetry projection is the functional mirror. The RH route can be read as the zero-fiber fixedness target for this projection, with ontic defect-zero incompatibility as the remaining zero-relevant boundary.*

Proof. The three projection rows are given by Definition 9413.272. Prime and zero rows remain typed projections by Definition 9413.275. The balance surface is the critical line by Definition 9413.278. The zero fiber section form is Definition 9413.281. The remaining incompatibility target is Definition 9413.287, and the quartic warning against raw-defect collapse is Definition 9413.290. These rows give the ontic projection principle without claiming the missing RH bridge. □

9413.21 Prime Phase-Radial Readback

section 9413.20 separates the complex value of zeta from the carrier rows that produce it. The reverse readback of a complex input s exposes those carrier rows as prime-indexed phase and radial coordinates. This section records that readback without treating the complex value $\zeta(s)$ as the complete object.

Definition 9413.292 (Prime phase-radial readback)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the prime phase-radial readback *of s is the compatible family of finite prime-window coordinates*

$$\mathcal{U}_S(s) := \big(U_p(s)\big)_{p\in S}, \qquad U_p(s) := p^{1/2-s} = p^{-\delta}e^{-i\tau\log p}.$$

As S ranges over nonempty finite prime windows, these rows form the prime-coordinate readback

$$\mathcal{U}_\zeta(s) := \big(p^{1/2-s}\big)_{p\in\mathsf{Prime}^\uparrow}.$$

This readback is not the complex value $\zeta(s)$. It is the prime-indexed phase and radial carrier whose trace, Euler-product, and completed readouts project down to the complex zeta value.

Definition 9413.293 (Prime torus and radial deformation body)**.** *The* prime torus *is the finite-window compatible phase body*

$$\mathbb{T}_{\mathsf{Prime}^\uparrow} := \prod_{p\in\mathsf{Prime}^\uparrow} S^1,$$

read through its finite faces

$$\mathbb{T}_S := \prod_{p\in S} S^1.$$

For a normal displacement δ, the corresponding radial deformation body has finite faces

$$\mathbb{T}_{S,\delta} := \prod_{p\in S} p^{-\delta}S^1.$$

Thus $\delta = 0$ is the pure phase torus, while $\delta\#0$ is a common normal radial deformation across all displayed prime axes.

Theorem 9413.294 (The critical surface is the prime torus)**.** *For the prime phase-radial readback,*

$$\Re(s) = \frac{1}{2} \quad\Longleftrightarrow\quad \mathcal{U}_S(s) \in \mathbb{T}_S \text{ for every nonempty finite prime window } S.$$

Proof. Write $s = 1/2 + \delta + i\tau$. For each displayed prime p,

$$U_p(s) = p^{-\delta} e^{-i\tau \log p}.$$

If $\Re(s) = 1/2$, then $\delta = 0$, so

$$U_p(s) = e^{-i\tau \log p} \in S^1$$

for every p. Hence every finite readback $\mathcal{U}_S(s)$ lies in $\mathbb{T}_S$. Conversely, if every finite face lies on the prime torus, choose any displayed prime p. The singleton face gives

$$|U_p(s)| = 1.$$

By Definition 9413.278, this is equivalent to $\Re(s) = 1/2$. □

Theorem 9413.295 (Zeta zeros are projected global cancellations)**.** *A zeta zero is not a vanishing of a prime coordinate. It is a global cancellation row in the trace, product, continuation, and completed projection of the prime phase-radial carrier.*

Proof. For every displayed prime p and every complex row s, the local channel p^{-s} and the centered channel $p^{1/2-s}$ are nonzero channel rows. The zero ledger is instead the global row

$$\zeta(s) = 0$$

or, in the completed carrier, the corresponding completed zero row. Thus zerohood belongs to the projected readout of all prime channels together with continuation and completion ledgers. It is not a local coordinate zero. The RH-facing statement is therefore that such global cancellation, when it occurs in the critical strip, is supported on the prime torus rather than on a nonunitary radial deformation body. □

Definition 9413.296 (Prime phase profile)**.** *For a real time coordinate τ, the* prime phase profile *is the compatible family of finite-window phase rows*

$$\Phi_S(\tau) := \left(e^{-i\tau \log p}\right)_{p \in S} \in \prod_{p \in S} S^1,$$

and the cofinal profile

$$\Phi_{\mathsf{Prime}^\uparrow}(\tau) := \left(e^{-i\tau \log p}\right)_{p \in \mathsf{Prime}^\uparrow} \in \mathbb{T}_{\mathsf{Prime}^\uparrow}.$$

It records the simultaneous phase positions of all prime logarithmic frequencies at the same observation time τ.

Definition 9413.297 (Global prime phase closure)**.** *A* global prime phase closure *at time τ is a completed zero-ledger row whose critical-line carrier is the prime phase profile*

$$\Phi_{\mathsf{Prime}^\uparrow}(\tau)$$

together with the analytic continuation and completion rows of the zeta carrier. It is not a row assigning the zero to a finite set of prime coordinates. The displayed local phase rows remain nonzero unit coordinates; closure belongs to the global completed projection.

Theorem 9413.298 (Critical-line zeros are all-prime phase closures)**.** *If*

$$s = \frac{1}{2} + i\gamma$$

is a completed zeta zero row, then its prime-coordinate carrier is the all-prime phase profile

$$\Phi_{\mathsf{Prime}^\uparrow}(\gamma) = \left(e^{-i\gamma \log p}\right)_{p \in \mathsf{Prime}^\uparrow},$$

and its zerohood is a global closure of that profile through the completed zeta projection.

Proof. When $s = 1/2 + i\gamma$, the normal displacement in Definition 9413.292 is $\delta = 0$. Hence every centered prime channel is

$$U_p(s) = e^{-i\gamma \log p} \in S^1.$$

Thus the prime-coordinate carrier is exactly $\Phi_{\mathsf{Prime}^\uparrow}(\gamma)$. By Definition 9413.295, the zero row is not a local coordinate zero; it is the completed projection of the whole prime phase-radial carrier. Therefore a critical-line zero is an all-prime phase closure at the time γ. □

Corollary 9413.299 (Different zero times give different phase fingerprints)**.** *The zero-height coordinate is read as a prime phase fingerprint: two critical-line zero rows at different times correspond to different cofinal prime phase profiles.*

Proof. If two phase profiles agree on all displayed primes, then for every prime p the phase ratio satisfies

$$e^{-i(\gamma_1 - \gamma_2)\log p} = 1.$$

For any two distinct primes this gives integer winding rows

$$(\gamma_1 - \gamma_2)\log p = 2\pi m_p, \qquad (\gamma_1 - \gamma_2)\log q = 2\pi m_q.$$

Eliminating the common time difference gives a rational relation between $\log p$ and $\log q$. By Definition 9413.677, the relation is trivial. Hence the time difference is zero. Thus a different zero time cannot have the same cofinal prime phase profile. □

Corollary 9413.300 (Prime frequencies do not cancel as coordinates)**.** *Global phase closure can cancel in the completed complex readout, but it does not identify the prime logarithmic frequencies with one another or remove a prime coordinate from the carrier.*

Proof. The completed value may vanish because the whole phase profile is projected through Euler, continuation, and completion ledgers. That is the global cancellation row of Definition 9413.295. The prime frequencies themselves remain the logarithmic directions $\log p$. Their finite rational independence is Definition 9413.677, so no finite algebraic relation among displayed prime frequencies collapses one prime direction into the others. □

Theorem 9413.301 (Radial deformation is not phase closure)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the phase profile

$$\Phi_{\mathsf{Prime}^\uparrow}(\tau)$$

and the radial deformation

$$\left(p^{-\delta}\right)_{p \in \mathsf{Prime}^\uparrow}$$

are separate rows of the prime phase-radial carrier. Phase cancellation in the completed readout does not cancel the positive square energy of a nonzero radial deformation.

Proof. The centered channel splits as

$$U_p(s) = p^{-\delta} e^{-i\tau \log p}.$$

The unit phase term is the p-coordinate of $\Phi_{\mathsf{Prime}^\uparrow}(\tau)$. The radial term is measured by

$$d_p(s) := \log |U_p(s)| = -\delta \log p.$$

Under the mirror J, this signed row changes sign, as recorded in Definition 9413.469. Its square

$$e_p(s) = d_p(s)^2 = \delta^2 (\log p)^2$$

is mirror-even by Definition 9413.673. Thus phase closure may account for a vanishing completed value, but it does not supply a negative term for the positive radial energy. □

The preceding statement is not a claim about arbitrary Euclidean vector polygons. In ordinary geometry, changing lengths can make a finite vector sum close. The BEDC claim is narrower: after the closed time–frequency carrier has fixed its legal hidden coordinates as unit phase and compact prime-register fibres, a real positive radial scale is not an admissible closing coordinate.

Theorem 9413.302 (Symmetric percentage motion leaves scale residue)**.** *Let* $0 < \varepsilon < 1$*. The two scale moves*

$$1 + \varepsilon, \qquad 1 - \varepsilon$$

do not compose to the identity scale:

$$(1 + \varepsilon)(1 - \varepsilon) = 1 - \varepsilon^2.$$

Equivalently,

$$\log(1 + \varepsilon) + \log(1 - \varepsilon) = \log(1 - \varepsilon^2) < 0.$$

Thus a scale displacement has a directed logarithmic residue unless the exact inverse scale is supplied as ledger data.

Proof. The multiplicative identity follows by expanding the product. Since $0 < \varepsilon < 1$, one has $0 < 1 - \varepsilon^2 < 1$, and hence $\log(1 - \varepsilon^2) < 0$. The inverse of $1 + \varepsilon$ is $(1 + \varepsilon)^{-1}$, not $1 - \varepsilon$. Therefore a scale system can be reversible only when the inverse scale has been recorded; a matched percentage motion in the opposite verbal direction is not an automatic inverse. □

Principle 9413.303 (Phase compactness and scale ledger)**.** *Unit phase belongs to the compact carrier*

$$S^1 \cong \mathbb{R}/2\pi\mathbb{Z},$$

so phase error is read modulo winding. Positive scale belongs, after logarithm, to the noncompact carrier

$$\mathsf{Real}^\uparrow_{>0} \cong (\mathsf{Real}^\uparrow, +),$$

so scale error is read as directed drift. In a closed BEDC time–frequency packet, phase may close inside the unit carrier, whereas nontrivial scale must appear in a boundary ledger or remain an illegal closure component.

Principle 9413.304 (Compact phase drift is not pointwise convergence)**.** *A phase orbit*

$$q_N = e^{iN\alpha}$$

stays in the compact unit carrier:

$$|q_N| = 1$$

for every N. This is bounded compact drift, not a convergence claim. Depending on the phase step and on additional analytic rows, such an orbit may be periodic, recurrent, equidistributed, averaged by a summability procedure, or used in a cancellation argument. None of these behaviours creates a real positive scale, and none is licensed as pointwise convergence without a separate convergence, averaging, or capture ledger.

Corollary 9413.305 (Scale amplifies while phase stays radial-unit)**.** *Let A be a finite active family with heights $h_a > 0$, and let*

$$U_a(s) = e^{-(s-1/2)h_a}$$

for $s = 1/2 + \delta + i\tau$. Define

$$d_a(s) := \log |U_a(s)|, \qquad E_A(s) := \sum_{a\in A} d_a(s)^2, \qquad W_A := \sum_{a\in A} h_a^2.$$

Then

$$E_A(s) = \delta^2 W_A.$$

Thus a nonzero normal displacement is amplified by any cofinal tower with $W_A \to \infty$, while the phase factors $e^{-i\tau h_a}$ contribute zero logarithmic radial energy.

Proof. The centered channel factors as

$$U_a(s) = e^{-\delta h_a} e^{-i\tau h_a}.$$

Since $|e^{-i\tau h_a}| = 1$,

$$d_a(s) = \log |U_a(s)| = -\delta h_a.$$

Squaring and summing over A gives

$$E_A(s) = \sum_{a\in A} \delta^2 h_a^2 = \delta^2 W_A.$$

The phase term has logarithmic modulus zero in every active channel, so it cannot contribute to the positive radial energy. □

Corollary 9413.306 (Finite resolution does not remove scale drift)**.** *A finite readout may fail to distinguish a small scale residue from the identity scale, but cofinal refinement cannot erase it. If $0 < \eta < 1$, then repeated scale residue satisfies*

$$(1-\eta)^N \to 0$$

as N tends along the natural refinement tower, while the logarithmic residue satisfies

$$N \log(1-\eta) \to -\infty.$$

Proof. Since $0 < 1-\eta < 1$, multiplication by $1-\eta$ is a strict contraction. Its iterates tend to zero. Taking logarithms gives

$$\log\big((1-\eta)^N\big) = N\log(1-\eta),$$

and $\log(1-\eta) < 0$, so the logarithmic residue diverges negatively. Thus finite resolution can hide a nontrivial scale residue only at a bounded window; the cofinal tower reads the drift. □

Principle 9413.307 (Phase time and scale time)**.** *The unitary time law*

$$e^{-itH}$$

is a phase row: when H is read in a unitary carrier, the inverse row is e^{itH}. The dissipative law

$$e^{-tH}, \qquad t \geq 0,$$

is a scale semigroup: reversing it requires the expanding row e^{tH}, which is not licensed by the forward semigroup without extra stability or boundary ledger data. Thus the zeta factor $e^{-i\tau h}$ belongs to phase closure, while $e^{-\delta h}$ belongs to scale drift unless a legal ledger accounts for it.

Theorem 9413.308 (No radial closure anomaly)**.** *The row*

$$\mathsf{NoRadialClosureAnomaly}_\zeta$$

holds for any zero-relevant closed generation certificate whose hidden coordinates are limited to unit phase rows and compact prime-register fibres. More explicitly, let

$$s = \frac{1}{2} + \delta + i\tau, \qquad U_a(s) = e^{-(s-1/2)h_a}, \qquad h_a > 0,$$

be an active centered channel in such a certificate. If the closure is licensed inside the closed time–frequency carrier, then

$$e^{-\delta h_a} = 1.$$

Consequently $\delta = 0$, so the legal closure has unit radial part and s lies on the critical line.

Proof. The centered channel factors as

$$U_a(s) = e^{-\delta h_a} e^{-i\tau h_a}.$$

The second factor is a unit phase row. Compact prime-register fibres, for instance finite faces of

$$\prod_{p\in S} \mathbb{Z}_p,$$

can encode prime-local residue or solenoid data, but they do not supply a nontrivial character into the real positive scale group $\mathsf{Real}^{\uparrow}_{>0}$. Therefore the real positive factor can belong to the closed hidden carrier only when it is trivial, by Definition 9413.557. Thus

$$e^{-\delta h_a} = 1.$$

Since $h_a > 0$, this forces $\delta = 0$. □

Corollary 9413.309 (Projected cancellation is not radial closure)**.** *A completed analytic readout may display a zero by global projection, but if a witness for that readout requires a nontrivial factor*

$$e^{-\delta h_a} \in \mathsf{Real}^{\uparrow}_{>0},$$

then that factor is a radial defect, not a phase closure or a prime-register residue.

Proof. Global zerohood is a projected cancellation row by Definition 9413.295. Legal closed generation uses the unit phase carrier together with compact prime-register fibres. By Definition 9413.308, a nontrivial real positive factor cannot be absorbed by those coordinates. Hence it marks a radial defect in the zero-relevant packet rather than a legal closed phase component. □

Corollary 9413.310 (Off-half zero rows would carry all-prime radial energy)**.** *If a critical-strip zero row had $\delta\#0$, then every displayed prime coordinate would carry nonzero radial defect, and the global prime normal energy would be infinite.*

Proof. For every displayed prime, Definition 9413.301 gives

$$d_p(s) = -\delta \log p.$$

Since $\delta\#0$ and $\log p > 0$, each prime row detects the same off-half normal displacement. Summing the positive square energies over finite prime windows gives the extended global energy of Definition 9413.672. By Definition 9413.675, that energy is infinite. □

Corollary 9413.311 (The no-hair target is zero relevance of radial energy)**.** *The remaining RH-facing target is not the detection of an off-half radial deformation. It is the zero-relevance claim that a completed zeta zero cannot carry the all-prime positive radial energy of Definition 9413.310.*

Proof. Detection is already finite and prime-local: each displayed prime reads $d_p(s) = -\delta \log p$, and every finite prime window reads the same normal displacement after normalization. The completed zero ledger must still be linked to that detected obstruction. This is the no-hair boundary of Definition 9413.678 and the cofinal socket form of Definition 9413.687. □

9413.22 Observation-Classified Symmetry on the Critical Surface

section 9413.20 reads zeta through prime atoms, logarithmic time, functional symmetry, and a zero ledger. section 9413.23 then extracts the minimal ontic core as observation-time, distinction-atom, and symmetry-transport. The critical line is the place where these two views meet: it is not a literal boundary between constructive and classical reasoning, nor a side-by-side separation of discrete and continuous objects. It is the observation-classified fixed surface of the zeta mirror, with prime-local and Laplace readouts giving compatible tests of the same classifier.

Definition 9413.312 (Critical mirror classifier)**.** *For a complex history*

$$s = \sigma + i\tau,$$

write

$$J(s) := 1 - \overline{s}, \qquad \delta(s) := \Re(s) - \frac{1}{2}.$$

The critical mirror classifier is the three-way readout

$$\mathsf{HalfClass}_\zeta(s) := \begin{cases} \mathsf{LeftBias}, & \delta(s) < 0, \\ \mathsf{Fixed}, & \delta(s) = 0, \\ \mathsf{RightBias}, & \delta(s) > 0. \end{cases}$$

It classifies a point by its observed displacement from the mirror fixed surface.

Theorem 9413.313 (The critical line is the mirror fixed surface)**.** *For the mirror $J(s) = 1 - \overline{s}$,*

$$J(s) = s \quad \Longleftrightarrow \quad \delta(s) = 0 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Thus the line $\Re(s) = 1/2$ is supplied by symmetry before any zero row is read.

Proof. Writing $s = \sigma + i\tau$ gives

$$J(s) = 1 - \overline{\sigma + i\tau} = 1 - \sigma + i\tau.$$

Hence $J(s) = s$ is equivalent to

$$1 - \sigma + i\tau = \sigma + i\tau,$$

which is equivalent to $\sigma = 1/2$. Since $\delta(s) = \sigma - 1/2$, this is the same as $\delta(s) = 0$. The zero predicate is not used in this calculation. □

Definition 9413.314 (Observation-classified symmetry)**.** *An* observation-classified symmetry *is a packet*

$$(\mathcal{X}, J, \delta)$$

where $J : \mathcal{X} \to \mathcal{X}$ is an involutive symmetry and $\delta : \mathcal{X} \to \mathbb{R}$ is an observation row satisfying

$$\delta(Jx) = -\delta(x).$$

The observation row separates the J-orbits into fixed and split classes:

$$\mathsf{FixedOrbit}_J(x) :\Longleftrightarrow \delta(x) = 0,$$

and

$$\mathsf{SplitOrbit}_J(x) :\Longleftrightarrow \delta(x)\#0.$$

For zeta, this packet is obtained from

$$\mathcal{X} = \mathsf{Complex}^\uparrow, \qquad J(s) = 1 - \overline{s}, \qquad \delta(s) = \Re(s) - \frac{1}{2}.$$

Theorem 9413.315 (Zeta mirror orbits are classified by the half displacement)**.** *The zeta mirror satisfies*

$$\delta(J(s)) = -\delta(s).$$

Consequently the observation row $\delta(s)$ separates mirror orbits into the fixed class $\delta(s) = 0$ and the split class $\delta(s)\#0$.

Proof. By definition,

$$J(s) = 1 - \overline{s}.$$

If $s = \sigma + i\tau$, then $J(s) = 1 - \sigma + i\tau$. Hence

$$\delta(J(s)) = \Re(J(s)) - \frac{1}{2} = 1 - \sigma - \frac{1}{2} = -\left(\sigma - \frac{1}{2}\right) = -\delta(s).$$

The fixed class is the zero fiber of δ, and the split class is its apartness fiber. □

Definition 9413.316 (Critical-surface readout packet)**.** *The critical-surface readout packet of the zeta carrier is*

$$\mathsf{CritRead}_\zeta(s) := (\mathsf{MirrorFix}(s), \mathsf{PrimeUnit}(s), \mathsf{NoLaplaceBias}(s)),$$

where

$$\mathsf{MirrorFix}(s) :\Longleftrightarrow s = J(s),$$
$$\mathsf{PrimeUnit}(s) :\Longleftrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ |p^{1/2-s}| = 1,$$

and, for logarithmic time $x > 0$,

$$\mathsf{NoLaplaceBias}(s) :\Longleftrightarrow e^{-\delta(s)x} = 1.$$

The three rows are respectively the symmetry readout, the prime-local readout, and the Laplace readout of the same centered surface.

Theorem 9413.317 (Prime and Laplace reads detect the same fixed surface)**.** *For $s = 1/2+\delta+i\tau$, the following rows are equivalent readings of the centered surface:*

$$s = J(s), \qquad \delta = 0, \qquad \forall p \in \mathsf{Prime}^{\uparrow},\ |p^{1/2-s}| = 1, \qquad \forall x > 0,\ e^{-\delta x} = 1.$$

Proof. The equivalence of $s = J(s)$ and $\delta = 0$ is Definition 9413.313. For each displayed prime row,

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p},$$

so its modulus is $p^{-\delta}$. This is one for every displayed prime exactly when $\delta = 0$, as in Definition 9413.278. The Laplace row decomposes as

$$e^{-s\log n} = e^{-(1/2+\delta)\log n} e^{-i\tau \log n}.$$

The factor $e^{-\delta x}$ is the additional bias along positive logarithmic time x. It is neutral for every displayed positive logarithmic time exactly on the zero fiber of δ. □

Corollary 9413.318 (The critical line is a balance surface, not a side boundary)**.** *The line $\Re(s) = 1/2$ is not the literal boundary between discrete and continuous projections. The discrete prime readout and the analytic Laplace-continuation readout coexist in the zeta carrier, and the critical line is their centered balance classifier.*

Proof. The prime-local row reads the centered channel

$$p^{1/2-s},$$

while the Laplace row reads

$$n^{-s} = e^{-s\log n}.$$

Both rows are present as projections of the zeta carrier. By Definition 9413.317, their centered classifiers agree exactly on $\delta = 0$. The absolute-convergence boundary of the Dirichlet and Euler product rows is a different ledger, namely the $\Re(s) > 1$ domain of Definition 9243.4. □

Definition 9413.319 (Prime-window observation of the split class)**.** *For a nonempty finite prime window S, the observation of the split class is the defect row*

$$D_S(s) := \sum_{p \in S} \log |p^{1/2-s}|.$$

If $s = 1/2 + \delta + i\tau$, then

$$D_S(s) = -\delta \sum_{p \in S} \log p.$$

Theorem 9413.320 (Prime windows observe split mirror orbits)**.** *If $\mathsf{SplitOrbit}_J(s)$ holds and S is a nonempty finite prime window, then the prime-window observation satisfies*

$$D_S(s) \# 0.$$

Proof. For the zeta mirror, $\mathsf{SplitOrbit}_J(s)$ is $\delta(s)\#0$ by Definition 9413.314. The defect formula of Definition 9413.319 gives

$$D_S(s) = -\delta(s) \sum_{p \in S} \log p.$$

The logarithmic sum is positive for a nonempty displayed prime window. Thus apartness of $\delta(s)$ from zero gives apartness of $D_S(s)$ from zero, matching Definition 9413.286. □

Definition 9413.321 (Zero-fixedness target)**.** *The* zero-fixedness target *is the section*

$$\mathsf{ZeroFixed}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to s = J(s)\right).$$

Equivalently, it asks that every critical-strip zero observation belongs to the fixed mirror class rather than the split mirror class.

Theorem 9413.322 (Zero-fixedness is the orbit form of the RH target)**.** *The zero-fixedness target is the observation-classified orbit form of the BEDC RH target.*

Proof. The BEDC RH target is the fixed-half section sending a row

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)$$

to an $\mathsf{OnCritLine}(s)$ witness, as in Definition 9413.199. By Definition 9413.313, the same conclusion may be read as $s = J(s)$. Thus the fixed-half section is equivalently the assertion that zero rows in the critical strip occupy the fixed mirror class. □

Corollary 9413.323 (Mirror closure alone does not supply zero-fixedness)**.** *Transport of zero rows along the mirror J is weaker than zero-fixedness.*

Proof. Mirror closure supplies a companion zero row at $J(s)$ when the functional-equation and non-escape ledgers are present. It does not identify s with $J(s)$. The remaining row is precisely the passage from a possible split mirror orbit to the fixed mirror orbit, as isolated in Definition 9241.14 and Definition 9413.321. □

Theorem 9413.324 (Observation-classified critical-surface principle)**.** *The zeta critical surface is the common observation classifier of three compatible rows:*

functional mirror fixedness, *prime-local unit amplitude,* *absence of extra Laplace bias.*

The RH-facing obligation is not to construct the surface, but to show that the critical-strip zero ledger factors through its fixed class.

Proof. The three compatible rows are assembled in Definition 9413.317. Prime windows observe off-fixed displacement by Definition 9413.320. The zero-facing target is the fixed-class section of Definition 9413.322. Therefore the critical surface is already determined as a classifier; the remaining boundary is the zero-relevant factorization through that classifier. □

9413.23 Ontic Core and Minimal Closure

section 9413.20 reads zeta as an arithmetic-spectral projection of atom, time, and symmetry. If the object-level examples are bracketed and only the ontic architecture is kept, the triple becomes

$$\mathsf{OnticCore} := (\mathsf{ObsTime}, \mathsf{DistAtom}, \mathsf{SymTransport}).$$

The three entries are observation as time, distinction as atom, and persistence as symmetry transport. The third entry is not shape symmetry alone. It is classifier-stable transport that preserves distinction through observation-time.

Definition 9413.325 (Observation-time row)**.** *An* observation-time row *is a finite readout event carried by a history*

$$h : \mathsf{Hist}$$

together with the continuation relation

$$h \preceq h' :\Longleftrightarrow \mathsf{Cont}(h, h').$$

In this row, time is not an external parameter. It is the readable order of observations under continuation.

Principle 9413.326 (Time is observable continuation)**.** *Inside the BEDC carrier discipline, time is the continuation order of observation histories. Without observed histories there is no internal time readout; without continuation there are only isolated readings.*

Definition 9413.327 (Distinction-atom row)**.** *A* distinction-atom row *is an irreducible stable distinction in a displayed reality. It has the abstract form*

$$x \# y$$

when the current classifier cannot identify the two rows, and the atomic form

$$\mathsf{Atom}(a)$$

when a is a minimal stable distinction under the generation law of that reality. Number-theoretic primes are the multiplicative arithmetic projection of this row, not its only instance.

Definition 9413.328 (Symmetry-transport row)**.** *A* symmetry-transport row *is a transport*

$$g : X \to X$$

that preserves classifier sameness and reflected distinction:

$$x \sim_X y \to g(x) \sim_X g(y),$$

and

$$x \# y \Longleftrightarrow g(x) \# g(y).$$

When g also preserves continuation,

$$h \preceq h' \to g(h) \preceq g(h'),$$

it is an ontic symmetry transport. In BEDC terms this is the role played by hsame*-respect,* Cont*, classifier stability,* StabilityCert*, and* NameCert*.*

Theorem 9413.329 (Symmetry is distinction preserved through observation-time)**.** *The symmetry-transport row is precisely the preservation of distinction-atom data along observation-time.*

Proof. Observation-time supplies histories and continuation rows $h \preceq h'$. Distinction-atom data supplies the classifier boundary $x\#y$. A symmetry transport is admissible only when it sends continuation rows to continuation rows and preserves the classifier boundary:

$$h \preceq h' \to g(h) \preceq g(h'),$$

$$x\#y \iff g(x)\#g(y).$$

Thus the content of symmetry is not visual resemblance. It is stable transport of distinction through observation-time. □

Corollary 9413.330 (Observation, distinction, and persistence form a loop)**.** *The ontic core is the loop*

$$\mathsf{ObsTime} \to \mathsf{DistAtom} \to \mathsf{SymTransport} \to \mathsf{ObsTime}.$$

Observation supplies readable time; distinction supplies observable content; symmetry transport lets that content persist across time.

Proof. Without observation-time, distinction does not enter a generated history. Without distinction, observation has no detectable content. Without symmetry transport, a distinction in one observation cannot be tracked through later observations. Hence the three rows are not independent decorations; they close one carrier loop. □

Definition 9413.331 (Minimal ontic core)**.** *The* minimal ontic core *is*

$$\mathsf{OnticCore} := (\mathsf{Observation}, \mathsf{Distinction}, \mathsf{Persistence})$$

with the readings

$$\mathsf{Observation} = \textit{finite readout producing time},$$

$$\mathsf{Distinction} = \textit{irreducible difference producing atom},$$

and

$$\mathsf{Persistence} = \textit{transport preserving distinction across observation-time.}$$

The persistence row is the ontic reading of symmetry.

Theorem 9413.332 (Stable interfaces require the ontic core)**.** *A stable BEDC interface must display observation-time, distinction-atom, and symmetry-transport ledgers before it can be consumed as a mature named object.*

Proof. The roadmap presents mature objects through a NameCert surface

$$(\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The source and pattern rows expose how the object is observed or generated. The classifier row exposes sameness and distinction. The stability certificate exposes preservation of the classifier under transport. The ledger policy records what is hidden, scoped, or refused. These are precisely the observation-time, distinction-atom, and symmetry-transport roles of Definition 9413.331. □

Corollary 9413.333 (Kernel reading of the ontic core)**.** *The BEDC kernel reads the ontic core through the spine*

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{NameCert}.$$

Here histories and continuations read observation-time, marks and classifier gaps read distinction-atom, and hsame*-respect,* Cont*-respect,* StabilityCert*, and* NameCert *read symmetry-transport.*

Proof. The displayed spine is the kernel-layer routing surface of the roadmap. The earlier layers produce readable histories and continuation. The classifier and gap layers keep distinctions from being collapsed. The package and naming layers admit only transports whose stability and ledger rows have been displayed. □

Definition 9413.334 (Ontic collapse of the core)**.** *An* ontic core collapse *is an unaudited step that erases one of the three core rows:*

$$(\exists x,\ P(x)) \to \sum_x P(x),$$

$$x \sim y \to [x] = [y],$$

or

$$(P \leftrightarrow Q) \to (P = Q).$$

These are respectively collapse of observation-time provenance, symmetry-transport fibers, and distinction-atom provenance.

Theorem 9413.335 (The three excluded axioms collapse the ontic core)**.** *Read through the minimal ontic core, the three excluded dependencies collapse the three core rows:*

Classical.choice *collapses observation-time provenance,*

Quot.sound *collapses symmetry-transport fibers,*

and

propext *collapses distinction provenance.*

Proof. This is the ontic-core reading of Definition 9413.283 and Definition 9387.5. Choice supplies a witness without the generating path. Quot.sound turns a classifier relation into quotient equality without retaining the transport fiber. propext turns bidirectional implication into proposition equality without retaining the two directions as distinct provenance. □

Corollary 9413.336 (Axiom-purity protects observation, distinction, and persistence)**.** *The axiom-purity discipline protects the minimal ontic core by refusing unaudited collapse of observation-time, distinction-atom, and symmetry-transport rows.*

Proof. By Definition 9413.335, the excluded dependencies correspond exactly to the three collapse modes. Refusing them preserves the ledgered core rather than compressing it into an oracle value, quotient equality, or proposition equality. □

Theorem 9413.337 (Minimal closure of the ontic core)**.** *Observation, distinction, and persistence form the minimal closure needed for a stable object interface.*

Proof. If observation and distinction are present without persistence, a distinction can be seen but cannot be transported through later observations; the result is not a stable object. If observation and persistence are present without distinction, the continuation has no detectable content. If distinction and persistence are present without observation, there is no history or continuation surface on which the distinction can be generated. Hence all three rows are required. □

Theorem 9413.338 (Ontic-core projection principle)**.** *Mature mathematical objects are projections of the minimal ontic core. Zeta is one dense arithmetic-spectral projection; it is not the core itself.*

Proof. For $\mathsf{Nat}^{\uparrow}$, observation-time is successor history, distinction is the zero/successor split, and persistence is induction stability. For $\mathsf{Real}^{\uparrow}$, observation-time is stream/modulus data, distinction is apartness, and persistence is classifier stability. For the phase and transform ladder, observation-time appears as phase or logarithmic time, distinction as modes or prime-local factors, and persistence as duality, unitary balance, or functional symmetry. The zeta projection compresses these rows into prime factors, log-time Laplace readout, and the mirror $J(s) = 1 - \overline{s}$, as in Definition 9413.272. Thus zeta is a projection of the core, not an exhaustive replacement for it. □

9413.24 Object Induction and the RH Reality Route

The RH route can be organized as an induction over licensed mathematical interfaces rather than as an induction over natural-number indices. The induction variable is a BEDC object admitted through the dependency tower

$$\text{kernel} \to \text{finite data} \to \text{arithmetic} \to \text{algebra} \to \text{analysis} \to \text{spectral and zeta carriers.}$$

The invariant carried by this induction is the ontic core of section 9413.23:

$$\mathsf{OnticCore} := (\mathsf{ObsTime}, \mathsf{DistAtom}, \mathsf{SymTransport}).$$

The point of this organization is twofold. First, it explains why zeta inherits observation-time, distinction-atom, and symmetry-transport rows. Second, it isolates the part that does not follow from inheritance: the zeta-specific incompatibility between zero rows and ontic defect.

Definition 9413.339 (Licensed object tower)**.** *The* licensed object tower *is the directed dependency presentation whose objects are mature interfaces*

$$X^{\uparrow}$$

admitted by a displayed naming certificate

$$\mathsf{NameCert}(X) = (\mathsf{SourceSpec}_X, \mathsf{PatternSpec}_X, \mathsf{ClassifierSpec}_X, \mathsf{StabilityCert}_X, \mathsf{LedgerPolicy}_X),$$

and whose arrows are generalized continuation, projection, transport, or bridge rows between such interfaces. The dependency direction always runs from lower licensed interfaces to the interface that consumes them.

Definition 9413.340 (Ontic readiness predicate)**.** *For a licensed interface $X^{\uparrow}$, the predicate*

$$\mathsf{OnticReady}(X)$$

means that X carries rows

$$\mathsf{ObsTime}_X, \qquad \mathsf{DistAtom}_X, \qquad \mathsf{SymTransport}_X$$

with a ledger showing how they are projected from dependencies, how their classifier is preserved under transport, and how unaudited collapse of observation-time, distinction provenance, or symmetry fibers is refused.

Principle 9413.341 (Object-induction rule)**.** *Let $X^{\uparrow}$ be a licensed interface. If every displayed dependency $Y \in \mathrm{Deps}(X)$ satisfies $\mathsf{OnticReady}(Y)$, and if $\mathsf{NameCert}(X)$ supplies source, pattern, classifier, stability, and ledger rows transporting those ontic rows into X, then*

$$\mathsf{OnticReady}(X)$$

holds. The base case is the kernel spine

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{NameCert},$$

read as in Definition 9413.333.

Theorem 9413.342 (NameCert transports the ontic core)**.** *Within the licensed object tower, a $\mathsf{NameCert}(X)$ packet transports the ontic core from the dependencies of X into the public interface of X.*

Proof. The source and pattern rows of $\mathsf{NameCert}(X)$ record the generated data and therefore the observation-time surface. The classifier row records sameness, apartness, and generated distinction. The stability row records preservation of the classifier under permitted transports. The ledger policy records the boundary of what is hidden, scoped, refused, or still missing. These are exactly the rows isolated in Definition 9413.332. □

Theorem 9413.343 (Object induction yields an ontic section)**.** *The object-induction rule yields a schematic section*

$$\mathsf{OnticCoreSection} : \mathcal{C}_{\mathrm{BEDC}} \to \mathcal{E}_{\mathrm{Ontic}}$$

from the licensed BEDC dependency presentation to the indexed packet of ontic-core readouts.

Proof. At the kernel base, the rows Hist, Cont, hsame, Gap, Pkg, and NameCert supply observation-time, distinction-atom, and symmetry-transport by Definition 9413.333. For the inductive step, apply Definition 9413.342 to a licensed interface X. The dependency hypothesis supplies the ontic rows on the inputs of X, and $\mathsf{NameCert}(X)$ supplies the transport and ledger rows needed to read them at X. Thus the ontic-core assignment is stable along the dependency direction. □

Corollary 9413.344 (Axiom-purity is part of the induction invariant)**.** *The object-induction invariant is not merely the presence of three rows; it also includes refusal of unaudited collapse of those rows.*

Proof. Definition 9413.335 identifies Classical.choice, Quot.sound, and propext as collapses of observation-time provenance, symmetry-transport fibers, and distinction provenance. Therefore a transport that preserved only the visible outputs but not the collapse ledger would fail to preserve OnticReady. □

Definition 9413.345 (Reality-indexed ontic fibre)**.** *For a licensed object $X^{\uparrow}$, its* reality-indexed ontic fibre *is the packet*

$$\mathcal{E}_X := (\mathsf{Readouts}_X, \mathsf{Classifier}_X, \mathsf{Transport}_X, \mathsf{Ledger}_X, \mathsf{Boundary}_X).$$

Projection maps

$$\pi_i : X^{\uparrow} \to R_i$$

select the reality readouts of X. In this language, $\mathsf{OnticCoreSection}$ *assigns to each licensed object its observation-time, distinction-atom, and symmetry-transport rows inside the fibre $\mathcal{E}_X$.*

Theorem 9413.346 (Zeta is the arithmetic-spectral fibre of the ontic section)**.** *At the zeta object, the ontic section specializes to the arithmetic spectral packet*

$$\mathcal{O}_{\zeta}^{\mathrm{ontic}} = (\mathsf{PrimeAtoms}, \mathsf{LogTime}, \mathsf{LaplaceReadout}, \mathsf{EulerProduct}, \mathsf{FunctionalSymmetry}, \mathsf{ZeroLedger}).$$

Its three core rows are prime atoms, logarithmic time, and functional symmetry.

Proof. This is the specialization of Definition 9413.272. The atom row is the Euler-product face

$$\prod_p (1 - p^{-s})^{-1}.$$

The time row is the logarithmic Laplace readout

$$n^{-s} = e^{-s \log n}.$$

The symmetry row is the mirror

$$J(s) = 1 - \overline{s}.$$

The zero ledger is an additional spectral readout over the same carrier, not a replacement for the three core rows. □

Definition 9413.347 (Reality-induction RH package)**.** *The* reality-induction RH package *is the sequence of rows*

$\mathsf{ObjInd}_{\mathrm{Ontic}}$, $\mathsf{ZetaOnticProjection}$, $\mathsf{OnticBalance}$, $\mathsf{ZeroFibreTarget}$, $\mathsf{DefectVisibility}$, $\mathsf{DefectZeroIncompatibilit}$

The first row is supplied by object induction. The last row is the zeta-specific rigidity obligation.

Theorem 9413.348 (Balance is the zeta fixed-half classifier)**.** *For the zeta fibre, ontic balance is the fixed-half classifier:*

$$\mathsf{OnticBalance}(s) \iff \Re(s) = \frac{1}{2} \iff s = J(s).$$

Proof. The equivalence between $\mathsf{OnticBalance}(s)$ and $\Re(s) = 1/2$ is Definition 9413.278. The equivalence between $\Re(s) = 1/2$ and $s = J(s)$ is Definition 9413.313. Thus the same classifier can be read through prime-local balance or through the functional mirror. □

Definition 9413.349 (Ontic rigidity form of RH)**.** *The* ontic rigidity form *of the RH target is the section*

$$\mathsf{OnticRigidity}_{\zeta} := \prod_{s:\mathsf{Complex}^{\uparrow}} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnticBalance}(s)\right).$$

It is the ontic-balance reading of the zero-fibre fixedness target of Definition 9413.280.

Theorem 9413.350 (Ontic rigidity is the RH section target)**.** *The ontic rigidity form is equivalent, as a target, to the BEDC RH fixed-half section.*

Proof. The fixed-half section is the section form of RH by Definition 9413.199. By Definition 9413.348, the conclusion $\mathsf{OnCritLine}(s)$ may be read as $\mathsf{OnticBalance}(s)$. Substituting this target row gives Definition 9413.349. □

Theorem 9413.351 (Defect visibility is the finite-window read of imbalance)**.** *If the zeta fibre carries an ontic defect at s, then every nonempty finite prime window reads a nonzero defect*

$$D_S(s) = \sum_{p \in S} \log |p^{1/2-s}|.$$

Proof. This is the finite-window defect calculation of Definition 9413.286. Writing $s = 1/2 + \delta + i\tau$ gives

$$D_S(s) = -\delta \sum_{p \in S} \log p.$$

If $\delta \# 0$ and S is nonempty, the displayed sum is apart from zero. □

Definition 9413.352 (Reality-induction hard point)**.** *The hard point of the reality-induction RH route is the zeta-specific row*

$$\mathsf{OnticDefectZeroIncompatibility} := \prod_s \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \mathsf{OnticDefect}(s) \to \bot\right).$$

Equivalently, after finite-window extraction, it is the bridge from prime-window defect in the completed zeta carrier to exclusion of a zero row at the same source.

Theorem 9413.353 (The hard point closes the RH route)**.** *Assume ontic defect-zero incompatibility and the located row converting absence of ontic defect into ontic balance. Then the ontic rigidity form of RH is obtained.*

Proof. Given rows

$$\mathsf{ZetaZero}(s), \qquad \mathsf{InCritStrip}(s),$$

the incompatibility row of Definition 9413.352 rules out $\mathsf{OnticDefect}(s)$. The located row converts the absence of that defect into $\mathsf{OnticBalance}(s)$. This is precisely the argument of Definition 9413.288 expressed as the reality-induction route. □

Corollary 9413.354 (Object induction does not itself prove RH)**.** *Object induction supplies the ontic rows that zeta must carry, but it does not supply the zeta-specific zero-defect incompatibility row.*

Proof. Object induction yields the section of Definition 9413.343 and the zeta specialization of Definition 9413.346. It therefore explains the presence of prime atoms, log-time, and mirror symmetry in the zeta carrier. The RH section still needs the incompatibility row of Definition 9413.352. That row is zeta-specific and is not produced by the general object-induction principle. □

Theorem 9413.355 (Reality-induction normal form of the RH route)**.** *The unified-reality RH route has the normal form*

object induction ⇒ zeta ontic projection ⇒ fixed-half balance classifier ⇒ zero-fibre section target,

with finite-window defect visibility as the observable off-balance row and ontic defect-zero incompatibility as the remaining rigidity obligation.

Proof. Object induction is Definition 9413.343. The zeta specialization is Definition 9413.346. The fixed-half balance classifier is Definition 9413.348. The zero-fibre section target is Definition 9413.350. The finite-window read of off-balance data is Definition 9413.351. The unclosed rigidity row is Definition 9413.352. These rows give the proof-chain architecture without claiming the missing analytic bridge. □

Corollary 9413.356 (Categorical reading)**.** *In categorical language, the route says that* OnticCoreSection *is a section over the licensed BEDC object presentation, while RH is the assertion that the zeta-zero fibre admits a section into the fixed-half subfibre.*

Proof. The first statement is Definition 9413.343. The second is the fixed-half section form of Definition 9413.198 and Definition 9413.199. The two statements have different logical roles: the first supplies the ambient ontic rows for licensed objects, while the second is the zeta-specific zero-fibre rigidity target. □

9413.25 Triadic Structural Induction

section 9413.24 organizes licensed objects by their construction history, and section 9413.47 reads the RHRoute invariant through prime ledger, unit boundary, and recursive readback. The present section records the common induction principle behind those two uses. The induction is not over natural-number indices and not over every named object in an ambient universe. It is structural induction over objects generated by BEDC constructors and carrying construction certificates.

Definition 9413.357 (Triadic base packet)**.** *The* triadic base packet *is the triple*

$$\mathfrak{B}_{\triangle} := (\mathsf{P}_0, \mathsf{U}_0, \mathsf{T}_0)$$

with the following roles:

$$\mathsf{P}_0 := \textit{minimal distinction, finite source ledger, and prime register},$$

$$\mathsf{U}_0 := \textit{unit phase boundary and orthogonal positive readout},$$

and

$$\mathsf{T}_0 := \textit{append-only observation generation and finite witness order.}$$

The packet is a role declaration, not an additional primitive carrier. Its concrete rows are supplied by the kernel distinction and history interfaces, the unit-circle and Pythagorean readouts of section 9413.6, and the prime-register routes used by the zeta-facing chapters.

Definition 9413.358 (Triadic construction invariant)**.** *For a licensed BEDC object X, the predicate*

$$\mathcal{I}_{\triangle}(X)$$

means that X carries three compatible certificate rows

$$\mathcal{I}_{\mathsf{P}}(X), \qquad \mathcal{I}_{\mathsf{U}}(X), \qquad \mathcal{I}_{\mathsf{T}}(X).$$

The source row $\mathcal{I}_{\mathsf{P}}(X)$ records generated provenance, irreducible decomposition, and ledger auditability. The unit row $\mathcal{I}_{\mathsf{U}}(X)$ records unit-boundary closure, positive orthogonal readout, or legal gauge. The time row $\mathcal{I}_{\mathsf{T}}(X)$ records recursive generation, append-only transport, and finite visibility of failures. The predicate is the compatibility of the three rows, not their independent presence.

Principle 9413.359 (Certificate-bounded domain of the induction)**.** *The structural induction governed by $\mathcal{I}_\triangle$ ranges only over objects generated by the declared BEDC construction grammar and admitted by a certificate. A mathematical phrase not accompanied by such a construction certificate does not inherit $\mathcal{I}_\triangle$ merely by being named.*

9413.25.1 Circle–Triangle–Register Grammar

Definition 9413.360 (BEDC null surface)**.** *The* BEDC null surface

$$\varnothing_{\mathrm{BEDC}}$$

is the pre-object state in which no distinction row, history row, coordinate row, or ledger row has been displayed. It is not used as a set-theoretic carrier. It is the boundary from which a certified object can begin only when an apartness event is displayed:

$$B_0 \# B_1.$$

The corresponding history rows are

$$\mathsf{Hist}_0 = (B_0), \qquad \mathsf{Hist}_1 = (B_0, B_1).$$

Definition 9413.361 (Initial distinction split)**.** *An initial distinction split over the BEDC null surface is a packet*

$$\mathfrak{D}_1 := (B_0, B_1, \mathsf{Hist}_0, \mathsf{Hist}_1, \mathcal{H}_0, \mathcal{H}_1, E_1, \Lambda_1)$$

such that the ledger row contains the earlier mark,

$$\mathsf{Hist}_0 \subseteq \mathsf{Hist}_1,$$

and the Hilbert readout row splits as

$$\mathcal{H}_1 = \mathcal{H}_0 \oplus E_1, \qquad E_1 \perp \mathcal{H}_0.$$

Thus ledger containment and Hilbert orthogonality are not competing claims. The displayed norm readout is

$$\|B_1\|^2 = \|B_0\|^2 + \|E_1\|^2$$

at the resolution supplied by the packet, and Λ_1 records the residual row E_1.

Definition 9413.362 (Circle–triangle–register grammar)**.** *The* circle–triangle–register grammar *is the least certified construction grammar generated from an initial distinction split by the following three constructor families.*

The circle constructor attaches a unit phase row

$$\mathsf{Circle}(\theta) = e^{i\theta} \in \mathbb{T}$$

with

$$|e^{i\theta}| = 1.$$

It supplies a unit-boundary row and no positive-real radial scale row.

The triangle-readout constructor attaches an orthogonal decomposition row

$$v = Pv + \eta, \qquad \eta \perp \mathrm{im}(P),$$

with readout

$$\|v\|^2 = \|Pv\|^2 + \|\eta\|^2.$$

It supplies the positive residual-square row.

The register constructor sends every displayed residual r that is used by the object to an explicit ledger entry

$$\mathsf{Register}(r) \in \Lambda.$$

Finite arithmetic residuals may be addressed through the Prime–Zeckendorf–Goedel register of Definition 9291.8; the register constructor does not allow a used residual to disappear by being encodable.

Definition 9413.363 (Circle–triangle–register generated class)**.** *Let*

$$\mathcal{C}_{\circ\triangle\Lambda}$$

be the finite generated class obtained from packets satisfying Definition 9413.361 by repeated application of the circle, triangle-readout, and register constructors of Definition 9413.362. A limit point of this class belongs to the same generated domain only after it supplies a limit closure row in the sense of Definition 9413.367.

Theorem 9413.364 (Circle–triangle–register generation induction)**.** *Assume the initial distinction split satisfies*

$$\mathcal{I}_\triangle(\mathfrak{D}_1),$$

and assume that each displayed circle, triangle-readout, and register constructor is triadic-preserving. Then every finitely generated object

$$X \in \mathcal{C}_{\circ\triangle\Lambda}$$

satisfies

$$\mathcal{I}_\triangle(X).$$

Proof. Proceed by structural induction on the construction tree of X. The base case is the displayed initial distinction split Definition 9413.361, which carries the source ledger, unit-boundary refusal of radial scale, and generated witness order required by $\mathcal{I}_\triangle$. For a constructor node, the induction hypotheses give $\mathcal{I}_\triangle$ for all inputs. The constructor hypothesis says that the applied circle, triangle-readout, or register constructor is triadic-preserving in the sense of Definition 9413.365. Therefore the output also satisfies $\mathcal{I}_\triangle$. This covers every finite construction tree in $\mathcal{C}_{\circ\triangle\Lambda}$. □

Definition 9413.365 (Triadic constructor preservation)**.** *Let*

$$F(X_1, \ldots, X_n)$$

be a legal constructor in the BEDC grammar. The constructor is triadic-preserving *when it supplies a transport row*

$$\mathcal{I}_\triangle(X_1), \ldots, \mathcal{I}_\triangle(X_n) \Longrightarrow \mathcal{I}_\triangle(F(X_1, \ldots, X_n)).$$

The transport must display how source ledgers combine or restrict, how unit-boundary or positivity rows are preserved, and how the resulting object remains generated through finite or explicitly convergent witness data.

Theorem 9413.366 (Triadic structural induction)**.** *Let $\mathcal{C}_\triangle$ be the class generated from the triadic base packet by a family of legal constructors. If the base packet satisfies $\mathcal{I}_\triangle$ and every constructor in the family is triadic-preserving, then every finitely generated $X \in \mathcal{C}_\triangle$ satisfies*

$$\mathcal{I}_\triangle(X).$$

Proof. Proceed by structural induction on the construction tree of X. For a base leaf, the claim is the base-packet hypothesis. For a constructor node

$$X = F(X_1, \ldots, X_n),$$

the induction hypotheses give $\mathcal{I}_\triangle(X_i)$ for each input. Since F is triadic-preserving, Definition 9413.365 supplies $\mathcal{I}_\triangle(X)$. This covers every finite construction tree in the generated class. □

Definition 9413.367 (Limit closure row)**.** *A generated tower*

$$X_0 \to X_1 \to X_2 \to \cdots$$

has a triadic limit closure row *when it supplies a limit object X_∞, compatible transports of $\mathcal{I}_\triangle(X_m)$ along the tower, and a closedness certificate*

$$\left(\prod_m \mathcal{I}_\triangle(X_m)\right) \Longrightarrow \mathcal{I}_\triangle(X_\infty)$$

in the declared topology or convergence discipline.

Theorem 9413.368 (Triadic induction with certified limits)**.** *The conclusion of Definition 9413.366 extends to limit objects only when the construction supplies the limit closure row of Definition 9413.367.*

Proof. The finite stages are covered by Definition 9413.366. The displayed limit closure row is precisely the additional rule transporting the invariant from all finite stages to X_∞. Without that row, finite preservation gives no license to infer that an uninspected limit preserves source ledger, unit positivity, or finite witness visibility. □

9413.26 Constructor Risk Surfaces

Principle 9413.369 (Product preservation)**.** *Products are triadic-preserving only when the product certificate displays ledger pairing, positivity of the product or sum norm readout, and a compatible interleaving of the two observation orders. A product that forgets either provenance row or changes the witness order without transport data is not a triadic-preserving constructor.*

Principle 9413.370 (Quotient and projection preservation)**.** *Quotients and projections are high-risk constructors because they hide distinctions. They are triadic-preserving only when the hidden distinction is retained as a legal fibre, gauge, or ledger row. The solenoid projection*

$$\Sigma_S \to \mathbb{T}$$

is legal in this sense because the hidden row is the prime-register fibre of Definition 9289.4. A projection that needs an unrecorded positive-real scale fibre is not a pure unit-phase constructor.

Principle 9413.371 (Limit preservation)**.** *An infinite tower preserves the triadic invariant only by a closedness row, not by the absence of a discovered counterexample. For positive kernels, a typical legal row is finite-window convergence*

$$K_m \to K$$

together with preservation of every finite Gram inequality in the limit. This is a construction certificate; it is stronger than merely having checked many finite levels.

Principle 9413.372 (Analytic continuation preservation)**.** *Analytic continuation is triadic-preserving only when it displays local source agreement, the connected continuation domain, branch choices, and pole exclusion. Local Taylor agreement alone does not preserve $\mathcal{I}_\triangle$, because a missing branch or pole row can hide exactly the information that the source and unit certificates are meant to audit.*

9413.27 Zeta Constructor Preservation Boundary

Definition 9413.373 (ZetaBEDCConstruction)**.** *The obligation*

$$\mathsf{ZetaBEDCConstruction}$$

asserts that the completed zeta carrier, its zero-relevant observations, and its Pick or moment positivity readouts are generated by legal BEDC constructors preserving $\mathcal{I}_\triangle$. Its constructor surface includes Euler product source rows, prime-power ledger rows, Gamma and rational-boundary rows, completed normalization, functional-equation transport, analytic continuation, logarithmic derivative readout, Pick or Stieltjes transform, and finite or convergent positive-kernel towers.

Definition 9413.374 (ConstructorPreservationForXi)**.** *The obligation*

$$\mathsf{ConstructorPreservationForXi}$$

is the preservation half of ZetaBEDCConstruction*. It asks that each constructor used in the completed zeta route be triadic-preserving in the sense of Definition 9413.365. In particular, it must account for the negative sign in the prime side of ζ'/ζ, the archimedean Gamma correction, completed normalization, analytic continuation, and limit passage for finite positive-kernel or moment towers.*

Principle 9413.375 (Zeta constructor preservation is the induction target)**.** *Once* ZetaBEDCConstruction *is established, the general structural induction of Definition 9413.366 supplies the triadic invariant on the completed zeta carrier. What remains zeta-specific is not another restatement of RH, but the constructor preservation proof for Euler product, Gamma completion, continuation, logarithmic derivative, Pick transform, and finite tower convergence.*

Theorem 9413.376 (Zeta BEDC construction feeds the source-positive frontier)**.** *Assume* ZetaBEDCConstruction *and a source-built Pick factorization for the completed zeta kernel. Then the source-positive RH frontier has the positive witness required by Definition 9329.22.*

Proof. The zeta construction row places the completed zeta carrier and its zero-relevant observations inside the certified BEDC construction grammar. By Definition 9413.366, its generated objects carry $\mathcal{I}_\triangle$, including source ledger, legal unit-boundary or positivity row, and finite or convergent witness discipline. The assumed source-built Pick factorization is exactly the positive-kernel witness named in Definition 9271.8. Hence the hypotheses needed for the source-positive frontier are supplied, subject only to the standard zero-exclusion transport stated there. □

Principle 9413.377 (Status of triadic structural induction)**.** *Triadic structural induction explains why BEDC-generated certified objects inherit source ledger, unit-boundary positivity, and generated witness rows. It does not assert that an arbitrary named object is generated, and it does not replace the analytic proof that completed zeta's constructors preserve the invariant. For the RH route, the remaining construction problem is* $\mathsf{ConstructorPreservationForXi}$ *together with coverage of the completed zero ledger by the generated carrier.*

9413.28 Universal Generation Closure

section 9413.25 proves the triadic invariant for certified construction trees. The present section records the larger closure boundary behind that induction. BEDC does not begin from a stock of mature mathematical objects. A mature interface is usable only after it has entered through generated histories, package and gap rows, a naming certificate, and whatever limit ledger its public use requires.

Definition 9413.378 (BEDC generation stages)**.** *The BEDC generation stages are the classes*

$$\mathsf{BEDCGen}_0, \mathsf{BEDCGen}_1, \mathsf{BEDCGen}_2, \ldots$$

defined as follows. The base stage is the raw distinction surface

$$\mathsf{BEDCGen}_0 := \{\mathsf{b0}, \mathsf{b1}\}.$$

The successor stage $\mathsf{BEDCGen}_{n+1}$ *is generated from* $\mathsf{BEDCGen}_n$ *by the displayed kernel and interface constructors: history extension, continuation, asking, signature formation, package-token formation, gap membership, exact globalization, circle phase readout, orthogonal triangle readout, residual registration, NameCert formation, and derived-interface admission. In particular, a proposed public interface* $X^{\uparrow}$ *may enter only through a row*

$$\mathsf{NameCert}(X) = (\mathsf{SourceSpec}_X, \mathsf{PatternSpec}_X, \mathsf{ClassifierSpec}_X, \mathsf{StabilityCert}_X, \mathsf{LedgerPolicy}_X).$$

Definition 9413.379 (BEDC generated closure)**.** *The finite BEDC generated closure is*

$$\mathsf{BEDCGen}_\infty := \bigcup_{n \geq 0} \mathsf{BEDCGen}_n.$$

The completed generated closure

$$\overline{\mathsf{BEDCGen}}$$

contains an additional object only when its completion, limit, analytic continuation, quotient, or projection row supplies a ledger and a stability certificate. Thus a limit point is not admitted by being named as a limit; it is admitted by a closure row in the sense of Definition 9413.367.

Definition 9413.380 (BEDC-licensed mathematics)**.** *For an object or interface* X*, the predicate*

$$\mathsf{BEDCLicensedMath}(X)$$

means that every public use of X *factors through generated dependency rows and a certificate surface. Concretely, either* $X \in \mathsf{BEDCGen}_\infty$*, or* X *is admitted as a derived interface through* $\mathsf{NameCert}(X)$*, or* X *is admitted as a certified limit, projection, quotient, continuation, or bridge over previously licensed rows. Host notation alone is not a license.*

Theorem 9413.381 (Internal BEDC mathematics lies in the generated closure)**.** *For every object or interface X,*

$$\mathsf{BEDCLicensedMath}(X) \Longrightarrow X \in \overline{\mathsf{BEDCGen}}.$$

Proof. Unfold Definition 9413.380. If X is generated at a finite stage, then it belongs to $\mathsf{BEDCGen}_\infty$ and hence to $\overline{\mathsf{BEDCGen}}$. If X is a derived interface, then Definition 9352.17 requires $\mathsf{NameCert}(X)$; the certificate fields are generated dependency, classifier, stability, and ledger rows, so the interface is admitted by the successor-stage constructor of Definition 9413.378. If X is a completion, continuation, projection, quotient, or bridge, the definition requires an explicit closure or ledger row over previously licensed inputs, which is exactly membership in the completed closure. □

Theorem 9413.382 (Generated closure carries the triadic invariant)**.** *Assume that the initial BEDC distinction packet satisfies $\mathcal{I}_\triangle$, that every constructor named in Definition 9413.378 is triadic-preserving, and that every admitted completion row is a triadic limit closure row. Then every object*

$$X \in \overline{\mathsf{BEDCGen}}$$

satisfies

$$\mathcal{I}_\triangle(X).$$

Proof. If X lies in the finite generated closure $\mathsf{BEDCGen}_\infty$, then X belongs to some finite stage and therefore to a finite construction tree. The claim follows from Definition 9413.366. If X is admitted only after a completion, projection, quotient, continuation, or bridge row, the hypothesis says that the row is a triadic limit closure row. The limit transport is exactly the rule used in Definition 9413.368, so the triadic invariant is preserved at the admitted object. □

Definition 9413.383 (External conservative bridge obligation)**.** *The obligation*

$$\mathsf{AllExternalMathBridgeableToBEDC}$$

states that each standard mathematical structure X used by a mature argument has a BEDC interface $X^\uparrow$ such that

$$X^\uparrow \in \overline{\mathsf{BEDCGen}}$$

and every exported theorem consumed from X is transported through a conservative bridge row. This is an obligation family, not a theorem of the kernel. It is discharged layer by layer by arithmetic, algebra, order, set, type, completion, analysis, geometry, category, homology, number-theoretic, and spectral interface spines.

Definition 9413.384 (Zeta zeros in the BEDC closure)**.** *The obligation*

$$\mathsf{ZetaZerosInBEDCClosure}$$

states that every completed non-trivial zeta zero packet consumed by the RH route is an element of $\overline{\mathsf{BEDCGen}}$ *through the licensed* $\mathsf{ZetaZeros}^\uparrow$ *surface, the completed zeta construction rows, the zero classifier, the Li-coordinate readout, and the source-positive Pick or moment witness rows. It is stronger than licensing the predicate* $\mathsf{ZetaZero}$ *alone: the zero-relevant observations must also carry the triadic source, unit, and witness rows.*

Definition 9413.385 (Zeta triangular closure)**.** *The obligation*

$$\mathsf{ZetaTriangularClosure}$$

is the assertion that ZetaZerosInBEDCClosure *is compatible with the circle–triangle–register invariant. For each completed non-trivial zero packet ρ, its Li coordinate, prime-source ledger, scale-phase defect split, projection witnesses, and positive-kernel or moment rows must be transported through one generated certificate family.*

Theorem 9413.386 (Universal closure route is conditionally fixed-half)**.** *Assume* ZetaTriangularClosure *and* NoHiddenRadialRegister *for the completed zeta zero ledger. Then every completed non-trivial zero packet admitted by that closure has fixed-half Li readout.*

Proof. Let ρ be a completed non-trivial zero packet admitted by ZetaTriangularClosure. The closure supplies the Li coordinate row

$$z_\rho = 1 - \frac{1}{\rho} = e^{\ell + i\theta}$$

and transports it through the generated circle–triangle–register certificate. If $\ell \# 0$, then Definition 9331.18 classifies the packet as not generated as a legal pure closure unless the radial scale is recorded as scale-ledgered data. The NoHiddenRadialRegister hypothesis refuses both hidden radial gauge and ledger-free tail absorption for a pure zeta closure. Hence the located radial readback gives $\ell = 0$. Therefore

$$\left|1 - \frac{1}{\rho}\right| = 1,$$

and Definition 9290.16 gives

$$\Re(\rho) = \frac{1}{2}.$$

□

Principle 9413.387 (Status of universal generation closure)**.** *The internal closure theorem says only that BEDC-licensed mathematics is inside the BEDC generated closure. The external bridge obligation and the zeta triangular closure obligation remain mathematical work. For the RH route, the universal closure perspective changes the target from a detached zero-location assertion to a carrier question: completed zeta zero packets must be shown to enter the generated closure with the triadic rows intact.*

9413.29 Zero Recursor and Cover Boundary

The zero-signature route requires a type-level distinction before its faithfulness rows can be read correctly. A single zeta zero source is not a constructor for the whole zero fibre. Even a proof that one zero admits an ontic-core expression does not supply a section over all zero sources. The constructive RH target is a total map out of the zero source, and totality requires a cover or recursor for that source.

Definition 9413.388 (Zero source recursor)**.** *Let*

$$\mathcal{Z}_\zeta := \sum_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s))$$

be the critical-strip zero source. A zero source recursor *is a packet*

$$\mathsf{ZeroRecursor}_\zeta := (\mathsf{Seed}, \mathsf{Step}, \mathsf{SymTransport}, \mathsf{Cover}, \mathsf{FarEndUnity})$$

where Seed *supplies a generated zero-source row,* Step *supplies the permitted observation-time or spectral-time continuation rows,* SymTransport *transports generated rows through the mirror* $J(s) = 1 - \overline{s}$*,* Cover *states that every row of* $\mathcal{Z}_\zeta$ *is generated by the recursor, and* FarEndUnity *supplies far-end unity for generated zero rows.*

Definition 9413.389 (Core expression for zero sources)**.** *A zero-source row* $z : \mathcal{Z}_\zeta$ *has a* core expression *when its zero data are read only through the displayed ontic-core rows*

$$(\mathsf{ObsTime}, \mathsf{DistAtom}, \mathsf{SymTransport})$$

of the zeta carrier. The row is written

$$\mathsf{CoreExpr}_\zeta(z).$$

It records expression through the permitted observation-time, distinction-atom, and symmetry-transport ledgers; it is not a claim that the whole zero source has been generated.

Principle 9413.390 (Seed expression is not zero-source construction)**.** *A seed row*

$$z_0 : \mathcal{Z}_\zeta, \qquad \mathsf{CoreExpr}_\zeta(z_0)$$

does not imply

$$\prod_{z:\mathcal{Z}_\zeta} \mathsf{CoreExpr}_\zeta(z).$$

The latter is a dependent section over the zero source. It requires a cover or recursor whose generation rows reach every zero-source row.

Definition 9413.391 (Zero-source cover)**.** *The cover row of a zero source recursor is a displayed surjective generation map*

$$\mathsf{Generate}_\zeta : \mathsf{Param}_\zeta \to \mathcal{Z}_\zeta$$

together with

$$\mathsf{Cover}_\zeta := \prod_{z:\mathcal{Z}_\zeta} \sum_{a:\mathsf{Param}_\zeta} (\mathsf{Generate}_\zeta(a) = z)\,.$$

It is the row that turns a generated family into a construction principle for the whole zero source.

Theorem 9413.392 (Covered recursor transports core expression)**.** *Assume a zero source recursor whose seed, step, and mirror transport rows preserve core expression, and whose cover row reaches every zero source. Then every critical-strip zero source has a core expression:*

$$\prod_{z:\mathcal{Z}_\zeta} \mathsf{CoreExpr}_\zeta(z).$$

Proof. Let $z : \mathcal{Z}_\zeta$. The cover row supplies a parameter $a : \mathsf{Param}_\zeta$ with

$$\mathsf{Generate}_\zeta(a) = z.$$

Starting from the seed row and applying the displayed step and mirror transport rows along the generating path for a, the recursor preserves $\mathsf{CoreExpr}_\zeta$ at each generated zero row. Transporting along the cover equality gives $\mathsf{CoreExpr}_\zeta(z)$. □

Theorem 9413.393 (Covered far-end-unified recursor gives the RH section)**.** *Assume a zero source recursor whose cover reaches every zero-source row and whose far-end unity row holds for every generated zero. Then the RH fixed-half section is obtained.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. By cover, z is generated by some parameter of the zero source recursor. The far-end-unity row of the recursor supplies

$$\mathsf{FarEndUnity}_\zeta(s).$$

By Definition 9413.170, this is equivalent to $\delta(s) = 0$, hence to $\Re(s) = 1/2$. The fixed-half section equivalence of Definition 9413.199 gives the RH section. □

Theorem 9413.394 (Zero recursor cover isolates the same hard point)**.** *The zero-recursor route reduces the RH-facing task to two zeta-specific rows: cover of the whole zero source and far-end unity for generated zero rows.*

Proof. Definition 9413.390 shows that a single expressed zero cannot supply the dependent RH section. The cover row of Definition 9413.391 upgrades a generated family to a total zero-source construction. The far-end-unity row then converts each generated zero to the fixed-half surface by Definition 9413.393. Thus the missing content is not the shared vocabulary of the zero type, but the total cover and the refusal of hidden far-end defect absorption. The latter is the same analytic rigidity already isolated as Definition 9413.202 and Definition 9413.187. □

Corollary 9413.395 (Zero type sameness is weaker than zero recursor totality)**.** *The statement that zero rows have the same type, or that one zero row is core-expressible, is strictly weaker than a covered zero recursor.*

Proof. Type sameness supplies a common codomain for rows of $\mathcal{Z}_\zeta$. It does not supply a generating path to an arbitrary element of that type. A covered zero recursor supplies that path through $\mathsf{Generate}_\zeta$ and Cover_ζ. Only the latter can produce the dependent section required by the constructive RH target. □

9413.30 Zero-Fibre Normal Sheet Geometry

Concretized in §9043: finite normal-sheet support certificate

The geometric reading of the RH route must distinguish ordinary holomorphic regularity from zero-fibre rigidity. An off-fixed zero does not, by itself, say that the completed zeta readout is nonsmooth or nonconformal at that point. It says that the zero fibre has a normal sheet away from the fixed line. The RH-facing statement is therefore a zero-fibre support claim, not a failure of local complex analyticity.

Definition 9413.396 (Completed zeta body)**.** *Write*

$$s = \frac{1}{2} + \delta + i\tau$$

and set

$$F(\delta, \tau) := \xi\left(\frac{1}{2} + \delta + i\tau\right).$$

The completed zeta body *is the graph packet*

$$\mathcal{B}_\zeta := \{(\delta, \tau, z) : z = F(\delta, \tau)\}.$$

Its zero fibre *is*

$$\mathcal{Z}_\zeta^{\mathrm{geom}} := \{(\delta,\tau) : F(\delta,\tau) = 0\},$$

and the fixed line is

$$\mathcal{C}_\zeta := \{(\delta,\tau) : \delta = 0\}.$$

The RH section asks for $\mathcal{Z}_\zeta^{\mathrm{geom}}$ *to be supported on* $\mathcal{C}_\zeta$ *inside the critical strip.*

Definition 9413.397 (Mirror coordinates)**.** *The functional-equation mirror*

$$J(s) = 1 - \overline{s}$$

acts in (δ,τ) *coordinates by*

$$J(\delta,\tau) = (-\delta,\tau).$$

For the completed readout, the mirror row has the form

$$F(-\delta,\tau) = \overline{F(\delta,\tau)}.$$

If $F = A + iB$*, then* A *is mirror-even and* B *is mirror-odd:*

$$A(-\delta,\tau) = A(\delta,\tau), \qquad B(-\delta,\tau) = -B(\delta,\tau).$$

Theorem 9413.398 (Mirror symmetry gives paired zeros, not RH)**.** *If* $F(\delta_0,\tau_0) = 0$*, then* $F(-\delta_0,\tau_0) = 0$*. This mirror closure does not imply* $\delta_0 = 0$*.*

Proof. By the mirror row of Definition 9413.397,

$$F(-\delta_0,\tau_0) = \overline{F(\delta_0,\tau_0)}.$$

If the right side is $\overline{0}$, it is zero. The conclusion gives a mirror companion, not equality of the two coordinates. The missing statement is exactly the zero-fixedness target of Definition 9413.321. □

Definition 9413.399 (Normal zero sheet)**.** *A* normal zero sheet *for the completed zeta body is a zero-fibre row with nonzero normal coordinate:*

$$\mathsf{NormalZeroSheet}_\zeta := \sum_{\delta,\tau} \left(\delta\#0 \times F(\delta,\tau) = 0\right).$$

The no-extra-zero-sheet *row is*

$$\mathsf{NoExtraZeroSheet}_\zeta := \neg\mathsf{NormalZeroSheet}_\zeta.$$

Theorem 9413.400 (No extra zero sheet is the RH support form)**.** *The no-extra-zero-sheet row is the geometric support form of the RH fixed-half section.*

Proof. In (δ,τ) coordinates, the RH conclusion for a zero row is

$$F(\delta,\tau) = 0 \to \delta = 0.$$

This is precisely the exclusion of a zero row with $\delta\#0$, together with the located fixed-half collapse that converts absence of normal apartness into the fixed-half row. The section form is Definition 9413.199. □

Principle 9413.401 (Local conformality is not zero-fibre rigidity)**.** *Holomorphic smoothness, mirror symmetry, and local conformality at a simple zero do not imply no-extra-zero-sheet.*

Proof. Let $w = s - 1/2$ and take

$$G(w) = w^2 - a^2$$

with a real and $a\#0$. Then

$$G(-\overline{w}) = \overline{G(w)},$$

so the same mirror symmetry is present. The zeros $w = a$ and $w = -a$ are off the fixed line, and

$$G'(w) = 2w$$

is nonzero at both zeros. Thus the local readout is conformal at those zeros while the zero fibre still has off-fixed normal sheets. Therefore the RH-facing task is not ordinary conformality but exclusion of those normal sheets for the completed zeta carrier. □

Definition 9413.402 (Zero-fibre faithfulness)**.** *The completed zeta zero fibre is* faithful *when a zero row cannot coexist with a normal prime-window defect:*

$$\mathsf{ZeroFibreFaithful}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} (D_S(s)\#0) \to \bot \right).$$

This is the zero-fibre geometric form of defect-zero incompatibility.

Theorem 9413.403 (Zero-fibre faithfulness gives the RH section)**.** *Assume zero-fibre faithfulness and located fixed-half collapse. Then the RH fixed-half section is obtained.*

Proof. Let s carry critical-strip and zeta-zero rows. If $\delta(s)\#0$, then Definition 9413.201 gives $D_S(s)\#0$ for every nonempty finite prime window S. This supplies the existential defect row excluded by Definition 9413.402. Hence $\neg(\delta(s)\#0)$. Located fixed-half collapse gives $\delta(s) = 0$, and Definition 9413.199 gives the RH section. □

Theorem 9413.404 (Zero-fibre faithfulness is the normal-sheet reading of defect rigidity)**.** *Zero-fibre faithfulness, no-extra-zero-sheet, and defect-zero incompatibility are three readings of the same RH-facing rigidity boundary.*

Proof. An off-fixed zero sheet has $\delta(s)\#0$. By the prime-window defect calculation of Definition 9413.201, it therefore carries a finite prime-window defect. Excluding coexistence of that defect with a zero row is Definition 9413.402. The same exclusion is named as defect-zero incompatibility in Definition 9413.202 and as no far-end absorption in Definition 9413.143. Thus the geometric normal-sheet language identifies the same unclosed analytic boundary; it does not replace it with a local conformality claim. □

9413.31 Time-Fibre Mirror Atomicity

The mirror row in the completed zeta carrier is not a reversal of observation time. It is a normal reflection inside one time fibre. This distinction sharpens the zero-fibre geometry of section 9413.30: an off-fixed zero would not be a zero at another time; it would be a second normal read in the same time slot.

Definition 9413.405 (Directed zeta time)**.** *The* directed zeta time *object is the observation-frequency row*

$$\mathbb{T}_\zeta := \{\tau : \mathsf{Real}^{\uparrow}_{\geq 0}\}.$$

In critical-strip coordinates

$$s = \frac{1}{2} + \delta + i\tau,$$

write

$$\pi_T(\delta, \tau) := \tau, \qquad \pi_N(\delta, \tau) := \delta.$$

The order row $\tau_1 \leq \tau_2$ records continuation or observation order. It does not supply a default reverse morphism $\tau_2 \to \tau_1$.

Definition 9413.406 (Deterministic observation update)**.** *A directed observation carrier has deterministic update when every displayed state admits at most one authorized successor row. Writing X_t for the state fibre at time t, t^+ for the displayed successor time, and $\mathsf{Step}_t(x, y)$ for the update relation, this row is*

$$\mathsf{DetStep} := \prod_t \prod_{x:X_t} \prod_{y_1,y_2:X_{t^+}} (\mathsf{Step}_t(x, y_1) \times \mathsf{Step}_t(x, y_2) \to y_1 = y_2)\,.$$

Thus one time slot carries at most one generation step or observation update.

Principle 9413.407 (Deterministic update is weaker than zero-fibre atomicity)**.** *Deterministic update is a generation-side property. It does not imply that every readout inside the generated object is singleton-valued:*

$$\mathsf{DetStep} \not\Rightarrow \mathsf{TimeFibreAtom}_\zeta.$$

A deterministic stage may produce one polynomial, spectral packet, or zero carrier whose internal zero ledger has several rows. In the quartic spectral carrier, for example, one recurrence stage generates one displayed polynomial while the fixed endpoint window may carry an indexed zero family. Hence the RH-facing mirror argument requires a separate zero-reality condition: the normal zero read in each directed zeta time fibre must itself be single-valued.

Definition 9413.408 (Time-fibre mirror)**.** *The zeta mirror*

$$J(s) = 1 - \overline{s}$$

acts on (δ, τ) coordinates by

$$J(\delta, \tau) = (-\delta, \tau).$$

Thus it preserves the time projection and reverses only the normal projection:

$$\pi_T(J(\delta, \tau)) = \pi_T(\delta, \tau), \qquad \pi_N(J(\delta, \tau)) = -\pi_N(\delta, \tau).$$

Theorem 9413.409 (The zeta mirror is fibrewise in time)**.** *The functional-equation mirror is a time-fibre symmetry, not a time reversal.*

Proof. By Definition 9413.408, the mirror sends (δ, τ) to $(-\delta, \tau)$. The time coordinate is unchanged, while the normal coordinate changes sign. Hence the companion row supplied by mirror closure lies in the same time fibre. □

Definition 9413.410 (Zero time fibre)**.** *For a directed time row $\tau : \mathbb{T}_\zeta$, the* zero time fibre *is*

$$Z_{\zeta,\tau} := \{\delta : (\delta, \tau) \in \mathcal{Z}_\zeta^{\mathrm{geom}}\},$$

where $\mathcal{Z}_\zeta^{\mathrm{geom}}$ is the completed zeta zero fibre of Definition 9413.396. In this notation, the RH support form says

$$\prod_{\tau:\mathbb{T}_\zeta} \prod_{\delta \in Z_{\zeta,\tau}} (\delta = 0).$$

Definition 9413.411 (Time-fibre atomicity)**.** *The zero fibre satisfies* time-fibre atomicity *when each directed time fibre carries at most one normal zero read:*

$$\mathsf{TimeFibreAtom}_\zeta := \prod_{\tau:\mathbb{T}_\zeta} \prod_{\delta_1,\delta_2} (\delta_1 \in Z_{\zeta,\tau} \times \delta_2 \in Z_{\zeta,\tau} \to \delta_1 = \delta_2)\,.$$

Theorem 9413.412 (Mirror closure plus time-fibre atomicity gives no extra zero sheet)**.** *Assume mirror closure for the completed zeta zero fibre and time-fibre atomicity. Then the no-extra-zero-sheet row holds.*

Proof. Suppose a normal zero sheet were present. Then for some δ, τ there are rows

$$\delta \# 0, \qquad (\delta, \tau) \in \mathcal{Z}_\zeta^{\mathrm{geom}}.$$

Mirror closure and Definition 9413.409 put $(-\delta, \tau)$ in the same zero time fibre. Time-fibre atomicity gives

$$\delta = -\delta.$$

In the characteristic-zero real readout this implies $\delta = 0$, contradicting $\delta \# 0$. Hence no normal zero sheet is present. □

Corollary 9413.413 (Time-fibre atomicity gives the RH section)**.** *Under mirror closure and located fixed-half collapse, time-fibre atomicity gives the RH fixed-half section.*

Proof. By Definition 9413.412, there is no off-fixed normal zero sheet. By Definition 9413.400, this is the geometric support form of the RH section. □

Definition 9413.414 (Zero graph form)**.** *The zero fibre has* zero graph form *when there is a normal readout function*

$$\delta_Z : \mathbb{T}_\zeta \to \mathsf{Real}^\uparrow$$

such that every zero in a time fibre has that normal coordinate:

$$\mathsf{ZeroGraph}_\zeta := \sum_{\delta_Z:\mathbb{T}_\zeta \to \mathsf{Real}^\uparrow} \prod_{\tau:\mathbb{T}_\zeta} \prod_{\delta \in Z_{\zeta,\tau}} (\delta = \delta_Z(\tau))\,.$$

Theorem 9413.415 (Zero graph form implies time-fibre atomicity)**.** *Zero graph form implies time-fibre atomicity.*

Proof. Let δ_1 and δ_2 be two zero reads in the same time fibre $Z_{\zeta,\tau}$. By zero graph form, both are equal to the displayed value $\delta_Z(\tau)$. Therefore $\delta_1 = \delta_2$. □

Corollary 9413.416 (Zero graph form gives the fixed surface under mirror closure)**.** *If the zero fibre has zero graph form and is mirror-closed, then every zero-source row lies on the fixed surface.*

Proof. Zero graph form gives time-fibre atomicity by Definition 9413.415. The fixed-surface conclusion then follows from Definition 9413.413. □

Corollary 9413.417 (Off-fixed zeros are double normal reads)**.** *An off-fixed zero, together with mirror closure, produces two distinct normal reads in one time fibre.*

Proof. If (δ, τ) is a zero and $\delta \# 0$, mirror closure gives $(-\delta, \tau)$ as another zero in the same time fibre. Since $\delta \# 0$, the two normal coordinates are distinct. Thus off-fixed zerohood, under mirror closure, witnesses failure of time-fibre atomicity. □

Theorem 9413.418 (Time-fibre atomicity is the slot form of defect rigidity)**.** *Time-fibre atomicity, zero graph form, zero-fibre faithfulness, and defect-zero incompatibility are equivalent readings of the same zero-source rigidity boundary.*

Proof. The time-fibre reading says that mirror companions must occupy one normal slot at fixed τ. The zero graph reading expresses the same condition as a single-valued normal coordinate over directed time. If that condition fails, then by Definition 9413.417 there is an off-fixed zero, and by Definition 9413.201 every nonempty finite prime window detects a nonzero defect. Excluding that defect from a zero row is Definition 9413.402, which is the normal-sheet reading of Definition 9413.202. The four formulations therefore place the same unclosed analytic load in different coordinates. □

9413.32 Zero Signature Faithfulness and the Mirror Slot Principle

section 9413.24 isolates the remaining RH-facing hard point as zeta-specific rigidity of the zero fibre. The same hard point can be expressed as a signature faithfulness principle. A critical-strip zero has a mirror companion under the functional equation. If the zero carrier has no hidden observable beyond its ontic signature, then a zero and its mirror companion occupy the same signature slot. Faithfulness of that slot forces them to be the same zero, hence to lie on the fixed surface.

Definition 9413.419 (Zero source and mirror companion)**.** *The critical-strip zero source is*

$$\mathcal{Z}_\zeta := \sum_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)) .$$

For $s = \frac{1}{2} + \delta + i\gamma$*, the mirror is*

$$J(s) := 1 - \overline{s} = \frac{1}{2} - \delta + i\gamma.$$

The mirror companion row is the transport, supplied by the functional-equation and non-escape ledgers, from a zero-source row at s *to a zero-source row at* $J(s)$*.*

Definition 9413.420 (Zero ontic signature)**.** *The* zero ontic signature *is the packet*

$$\Sigma_Z(s) := (\mathsf{AtomType}_Z(s), \mathsf{SpecTime}_Z(s), \mathsf{MirrorSlot}_Z(s)),$$

where $\mathsf{AtomType}_Z(s)$ *records the zero as an irreducible spectral atom row,* $\mathsf{SpecTime}_Z(s)$ *records the spectral time coordinate* $\Im(s)$*, and* $\mathsf{MirrorSlot}_Z(s)$ *records the unordered mirror orbit generated by* s *and* $J(s)$*.*

Definition 9413.421 (Mirror-even signature)**.** *A zero signature is* mirror-even *when the mirror companion has the same signature:*

$$\Sigma_Z(J(s)) = \Sigma_Z(s).$$

For the signature of Definition 9413.420, this means that the mirror companion has the same zero-atom type, the same imaginary spectral time, and the same mirror slot.

Definition 9413.422 (Zero signature faithfulness)**.** *The zero source has* signature faithfulness *when the signature slot is faithful to zero identity:*

$$\mathsf{ZeroSignatureFaithful}_\zeta := \prod_{s,t:\mathcal{Z}_\zeta} (\Sigma_Z(s) = \Sigma_Z(t) \to s = t)\,.$$

Equivalently, the zero carrier has no hidden fourth observable that can separate two zero-source rows at the same carrier layer after their atom type, spectral time, and mirror slot agree. A higher-layer object is not excluded by this definition when it is carried by an explicit extension ledger; it is excluded only when it is used as an unledgered same-layer coordinate.

Theorem 9413.423 (Signature faithfulness implies zero-fixedness)**.** *Assume mirror closure for critical-strip zero rows, mirror-even zero signature, and zero signature faithfulness. Then the zero-fixedness target holds:*

$$\prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to s = J(s))\,.$$

Proof. Let s carry rows $\mathsf{ZetaZero}(s)$ and $\mathsf{InCritStrip}(s)$. Mirror closure supplies the companion zero-source row at $J(s)$. Mirror-evenness gives

$$\Sigma_Z(J(s)) = \Sigma_Z(s).$$

Signature faithfulness applied to the two zero-source rows gives

$$J(s) = s.$$

This is exactly the zero-fixedness conclusion of Definition 9413.321. □

Corollary 9413.424 (Signature faithfulness gives the RH section)**.** *Under the hypotheses of Definition 9413.423, the BEDC RH fixed-half section is obtained.*

Proof. By Definition 9413.423, every critical-strip zero row satisfies $s = J(s)$. By Definition 9413.313, this is equivalent to $\Re(s) = 1/2$, hence to the critical-line output row. The fixed-half section equivalence of Definition 9413.199 gives the RH section target. □

Definition 9413.425 (Mirror-odd observable)**.** *Let R be an additive characteristic-zero readout carrier. A zero observable*

$$F : \mathcal{Z}_\zeta \to R$$

is mirror-odd *when*

$$F(J(s)) = -F(s).$$

The normal displacement

$$\delta(s) := \Re(s) - \frac{1}{2}$$

is the basic mirror-odd observable for the zeta mirror.

Definition 9413.426 (Zero observable factorization)**.** *The zero source has* observable factorization through the zero signature *when every zero observable F in the displayed observable class factors through Σ_Z:*

$$F(s) = f(\Sigma_Z(s))$$

for some signature-level readout f.

Theorem 9413.427 (Mirror-odd observables vanish under signature factorization)**.** *Assume mirror-even zero signature and zero observable factorization through that signature. Then every mirror-odd zero observable in the factorizing class is zero.*

Proof. Let F be mirror-odd and suppose

$$F(s) = f(\Sigma_Z(s)).$$

Mirror-evenness gives

$$\Sigma_Z(J(s)) = \Sigma_Z(s).$$

Therefore

$$F(J(s)) = f(\Sigma_Z(J(s))) = f(\Sigma_Z(s)) = F(s).$$

Since F is mirror-odd, also $F(J(s)) = -F(s)$. Hence

$$F(s) = -F(s).$$

In characteristic zero this forces $F(s) = 0$. □

Corollary 9413.428 (Normal displacement vanishes under zero signature closure)**.** *If the normal displacement*

$$\delta(s) = \Re(s) - \frac{1}{2}$$

is a zero observable that factors through a mirror-even faithful zero signature, then $\delta(s) = 0$ for every critical-strip zero-source row.

Proof. By Definition 9413.425, δ is mirror-odd. Applying Definition 9413.427 to $F = \delta$ gives $\delta(s) = 0$. This is the fixed-half condition. □

Definition 9413.429 (No atomic mirror splitting)**.** *The* no atomic mirror splitting *row is*

$$\mathsf{NoAtomicMirrorSplitting}_\zeta := \prod_s \left(\mathsf{ZeroAtom}(s) \times s\#J(s) \to \bot\right).$$

It says that a zeta zero atom is symmetry-absolute rather than a paired atom separated by a hidden mirror-odd observable.

Theorem 9413.430 (Signature faithfulness rules out atomic mirror splitting)**.** *Mirror-even zero signature and zero signature faithfulness rule out atomic mirror splitting.*

Proof. Suppose a zero atom at s carried $s\#J(s)$. Mirror closure places the mirror companion in the same zero source. Mirror-evenness gives equal zero signatures for s and $J(s)$, and signature faithfulness gives $s = J(s)$. This contradicts $s\#J(s)$. □

Theorem 9413.431 (Split mirror zeros create prime-window defects)**.** *If a critical-strip zero source has $s\#J(s)$, then it carries the normal defect $\delta(s)\#0$ and every nonempty finite prime window reads a nonzero prime-window defect.*

Proof. For the mirror $J(s) = 1 - \overline{s}$, the equality $s = J(s)$ is equivalent to $\delta(s) = 0$ by Definition 9413.313. Thus apartness $s\#J(s)$ gives the off-fixed defect row $\delta(s)\#0$ in the located comparison. The finite-window defect calculation is Definition 9413.320. □

Corollary 9413.432 (Signature faithfulness is a zero-relevance form of defect rigidity)**.** *Zero signature faithfulness turns prime-visible mirror splitting into a zero-relevant contradiction. In that sense it is a signature form of ontic defect-zero incompatibility.*

Proof. By Definition 9413.431, a split mirror zero produces a prime-window defect. By Definition 9413.430, the same split row contradicts mirror-even signature faithfulness. This is the same logical role played by Definition 9413.352: a visible off-balance defect cannot coexist with a zero row in the completed zeta carrier. □

Theorem 9413.433 (Mirror slot principle)**.** *The mirror slot principle is the implication*

$$\textit{mirror closure} + \textit{mirror-even zero signature} + \textit{zero signature faithfulness} \Rightarrow \textit{zero-fixedness}.$$

It is a generalized slot-uniqueness principle for the zero carrier, not a finite counting argument.

Proof. The implication is exactly Definition 9413.423. The word slot refers to the signature fibre of Σ_Z, not to a finite pigeonhole set. The principle says that one signature fibre cannot contain two mirror-separated zero atoms when the zero signature is faithful. □

Corollary 9413.434 (The signature hard point)**.** *The unclosed content of the signature route is*

$$\mathsf{ZeroSignatureFaithful}_\zeta$$

or, equivalently for the normal displacement route, factorization of zero observables through a mirror-even zero signature.

Proof. Mirror closure and the algebraic identity $\delta(J(s)) = -\delta(s)$ are structural rows already displayed by the functional-equation mirror. What remains is the assertion that the zero signature has no hidden coordinate capable of separating s from $J(s)$. That assertion is exactly Definition 9413.422 or the factorization row of Definition 9413.426. Without it, a mirror companion may be a paired zero rather than the same zero. □

Principle 9413.435 (Expression is not construction)**.** *Expressing the RH target in the language of observation-time, distinction-atom, and symmetry-transport does not by itself construct an RH witness. The constructive target remains a total section*

$$\Phi : \mathcal{Z}_\zeta \to \mathcal{H}_\zeta,$$

or equivalently a function sending each row

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)$$

to an explicit $\mathsf{OnCritLine}(s)$ *witness. A vocabulary for the target is therefore weaker than an inhabitant of the target.*

Definition 9413.436 (Zero ontic completeness)**.** *The zero source has* zero ontic completeness *when the displayed ontic signature is not only mirror-even but observable-complete for the zero carrier:*

$$\mathsf{ZeroOnticComplete}_\zeta := \prod_F \sum_f \prod_{s:\mathcal{Z}_\zeta} F(s) = f(\Sigma_Z(s)),$$

where F ranges over the zero observables admitted by the displayed zero carrier. This is the factorization form of the claim that no additional zero-observable coordinate is available beyond the zero ontic signature.

Definition 9413.437 (No-fourth-ledger row)**.** *The* no-fourth-ledger row *for the zeta zero carrier is the refusal of an off-critical observable ledger independent of*

$$\Sigma_Z(s) = (\mathsf{AtomType}_Z(s), \mathsf{SpecTime}_Z(s), \mathsf{MirrorSlot}_Z(s)).$$

Equivalently, any displayed normal displacement row

$$\delta(s) = \Re(s) - \frac{1}{2}$$

must factor through the zero signature if it is admitted as a zero observable at the same zero-source layer. If it does not factor through that signature, then it must instead be carried as an explicit higher-layer normal-extension ledger of Definition 9413.438; the no-fourth-ledger row refuses precisely the unledgered same-layer use.

Definition 9413.438 (Off-critical normal-extension ledger)**.** *An* off-critical normal-extension ledger *for a zero-source row*

$$s = \frac{1}{2} + \delta + i\tau$$

is a higher-layer extension ledger that licenses

$$\delta \# 0$$

as a normal axis of the zero reality. Such a ledger must display the zero source, the mirror orbit, the normal apartness row, the finite prime-window defect rows generated by the apartness, and the zero-compatibility row explaining why those defect rows may coexist with the completed zero ledger. In notation this is written

$$\mathsf{NormalExtLedger}_\zeta(s, \delta).$$

Theorem 9413.439 (Zero ontic completeness implies zero-fixedness)**.** *Assume mirror closure, mirror-even zero signature, zero ontic completeness, and that the normal displacement $\delta(s)$ is a displayed zero observable. Then every critical-strip zero source is fixed by the mirror:*

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to s = J(s).$$

Proof. By zero ontic completeness, the normal displacement factors through the zero signature:

$$\delta(s) = f(\Sigma_Z(s)).$$

For the mirror companion,

$$\delta(J(s)) = f(\Sigma_Z(J(s))).$$

Mirror-evenness gives

$$\Sigma_Z(J(s)) = \Sigma_Z(s),$$

hence

$$\delta(J(s)) = \delta(s).$$

The mirror calculation of Definition 9413.425 also gives

$$\delta(J(s)) = -\delta(s).$$

Therefore $\delta(s) = -\delta(s)$, and the characteristic-zero real readout gives $\delta(s) = 0$. By Definition 9413.313, this is equivalent to $s = J(s)$. □

Corollary 9413.440 (Zero ontic completeness gives the RH target)**.** *Under the hypotheses of Definition 9413.439, the BEDC RH fixed-half section is obtained.*

Proof. The theorem gives zero-fixedness for every critical-strip zero source. The fixed-surface calculation converts zero-fixedness into $\mathsf{OnCritLine}(s)$, and Definition 9413.199 identifies this with the RH section target. □

Definition 9413.441 (Ontic-expression package)**.** *The* ontic-expression package *for RH is the data*

$$(\mathsf{OnticCore}, \mathsf{ZetaOnticProjection}, \mathsf{OnticBalance}, \mathsf{ZeroFibreTarget}).$$

It presents the RH target as a statement about zero rows landing in the atom-time-symmetry balance surface. It does not include $\mathsf{ZeroOnticComplete}_\zeta$*, signature faithfulness, or the defect-zero incompatibility bridge.*

Theorem 9413.442 (Ontic expression is weaker than RH construction)**.** *The ontic-expression package is a correct presentation of the RH target but not an RH proof object.*

Proof. The package contains the target row

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{OnticBalance}(s),$$

as in Definition 9413.349. To be a proof object it would need to produce this output for every zero-source input. That production is exactly the section Φ of Definition 9413.435. The ontic-expression package names the codomain and explains the balance surface; it does not supply the total function. □

Theorem 9413.443 (No-fourth-ledger is a signature form of defect-zero incompatibility)**.** *For the zeta zero carrier, the no-fourth-ledger row is the signature form of the same missing bridge isolated as ontic defect-zero incompatibility.*

Proof. If a critical-strip zero carried an off-critical displacement $\delta(s)\#0$, then Definition 9413.431 gives a finite prime-window defect. Such a defect is an additional observable row unless it is excluded by the zero carrier. The no-fourth-ledger row expresses this exclusion as signature completeness: the zero signature has no independent coordinate capable of recording the off-critical displacement. The defect-zero route of Definition 9413.352 expresses the same exclusion as incompatibility between a zero row and an ontic defect. Both formulations assign the remaining analytic load to the same zero-relevant rigidity claim. □

Corollary 9413.444 (Exact boundary of the signature route)**.** *The signature route proves RH exactly when the zeta zero carrier is shown to be complete and faithful for the displayed ontic signature.*

Proof. If completeness and faithfulness hold, then Definition 9413.439 and Definition 9413.440 give the RH section. Conversely, without completeness or faithfulness, the mirror may preserve the zero set while allowing paired zero atoms in one mirror orbit. Such a paired orbit uses the same three formal roles but does not collapse to the fixed surface. Thus the boundary is not the availability of the three ontic words, but the completeness and faithfulness of their zero-source signature. □

Concretized in §8973: $\mathsf{ZeroSignatureFaithfulness}^{\uparrow}$ realizes the mirror-slot faithfulness packet as finite NameCert data.

9413.33 Core Commonality and Completion Boundaries

The unified-reality programme is a theory of commonality, but the word "common" has two distinct roles. The first role is ontic: every licensed BEDC object must expose observation-time, distinction-atom, and symmetry-persistence rows. The second role is object-specific: one may ask whether those rows are complete and faithful for all public observables of a given carrier. The first role is the unifying language of the theory. The second role is a proof obligation, and in the zeta-zero case it is strong enough to imply the RH section.

Definition 9413.445 (Core commonality basis)**.** *The* core commonality basis *is*

$$\mathsf{Core} := (\mathsf{ObsTime}, \mathsf{DistAtom}, \mathsf{SymPersist}),$$

where $\mathsf{ObsTime}$ *records observation, history, and continuation;* $\mathsf{DistAtom}$ *records irreducible distinction and atom data; and* $\mathsf{SymPersist}$ *records classifier-stable transport preserving distinction through observation-time.*

Definition 9413.446 (Core signature of a licensed object)**.** *For a licensed object* $X^{\uparrow}$*, a* core signature *is a displayed readout*

$$\Sigma_X : X^{\uparrow} \to \mathsf{ObsTime}_X \times \mathsf{DistAtom}_X \times \mathsf{SymPersist}_X.$$

It records that X *can be observed through histories, has stable distinctions, and carries those distinctions through permitted transports.*

Definition 9413.447 (Core completeness)**.** *A licensed object* $X^{\uparrow}$ *is* core-complete *when every displayed public observable*

$$F : X^{\uparrow} \to R$$

in the public carrier class factors through the core signature:

$$F = \overline{F} \circ \Sigma_X$$

for some readout

$$\overline{F} : \mathsf{ObsTime}_X \times \mathsf{DistAtom}_X \times \mathsf{SymPersist}_X \to R.$$

This does not deny the existence of derived observables such as order, area, derivative, spectrum, zero, or defect. It says that no such observable is admitted as an independent fourth ontic primitive.

Definition 9413.448 (Independent fourth observable)**.** *An* independent fourth observable *for* $X^{\uparrow}$ *is a public observable* $Q : X^{\uparrow} \to R_Q$ *for which no factorization through* Σ_X *is supplied and which is not explicitly carried by a higher-layer extension ledger, boundary row, bridge row, or refused external operation.*

Definition 9413.449 (Layered generation license)**.** *A displayed row Y over a carrier layer L has a* layered generation license *when it is licensed in one of two ways. Either it is an ordinary core readout, meaning that it factors through the displayed core signature of L, or it is a higher-layer extension row, meaning that the source, gap or NonCollapse witness, classifier, transport, and boundary ledger for Y are explicitly displayed over L:*

$$\mathsf{Licensed}_L(Y) := \mathsf{CoreRead}_L(Y) \ \vee \ \mathsf{ExtensionLedger}_L(Y).$$

The second case creates an object in a derived layer. It does not add a fourth primitive to the source layer.

Principle 9413.450 (No same-layer fourth primitive)**.** *The anti-primitive discipline forbids an unledgered fourth role at the same carrier layer. It does not forbid derived objects. Algebraic generators, circle constraints, branch rows, spectral factors, and zero resonances are admissible only when their generation and extension ledgers are displayed; without such ledgers they are primitive intrusions rather than BEDC objects.*

Theorem 9413.451 (Core completeness excludes independent fourth observables)**.** *If $X^{\uparrow}$ is core-complete, then every displayed public observable of X is either a core-signature readout, a licensed higher-layer extension, or is outside the core surface as an explicit bridge, boundary, or refusal. In particular, no independent fourth observable is admitted inside the same-layer core interface.*

Proof. Core completeness gives a factorization

$$F = \overline{F} \circ \Sigma_X$$

for every public observable F in the core class. Therefore such an F is a readout of the core signature rather than an independent primitive. If a displayed row does not factor through the core signature, the licensed alternatives are exactly those of Definition 9413.449: carry it by a higher-layer extension ledger, place it in a bridge or boundary ledger, or refuse it at the public core surface. This is the anti-primitive discipline of Definition 9413.332 and Definition 9413.344. □

Corollary 9413.452 (Fourth commonality is a theory-extension signal)**.** *If a genuinely independent public observable Q is required for a carrier, then either Q is a derived core readout, a bridge or boundary row, or the three-row core is not complete for that carrier.*

Proof. By Definition 9413.451, a core-complete carrier leaves no place for an independent fourth observable inside its public core interface. Thus a proposed Q must be reclassified as a derived core readout, carried by a higher-layer extension ledger, placed in a bridge or boundary ledger, refused as an unaudited primitive, or treated as evidence that the core-completeness claim for the carrier has failed. □

Definition 9413.453 (Zeta zero core-completeness target)**.** *The* zeta zero core-completeness target *is*

$$\mathsf{CoreComplete}(\mathcal{Z}_\zeta),$$

where the source carrier $\mathcal{Z}_\zeta$ is the critical-strip zero source and Σ_Z is the zero ontic signature of Definition 9413.420. It asserts that every public zero observable factors through the atom-type, spectral time, and symmetry-persistence signature of the zero source.

Definition 9413.454 (Normal displacement as a candidate fourth observable)**.** *For the zeta mirror, the normal displacement is*

$$\delta(s) := \Re(s) - \frac{1}{2}.$$

It is a candidate fourth observable for the zero source because it separates the mirror companions

$$s = \frac{1}{2} + \delta + i\gamma, \qquad J(s) = \frac{1}{2} - \delta + i\gamma$$

by the sign rule

$$\delta(J(s)) = -\delta(s),$$

while their zero atom type, spectral time, and mirror orbit signature are the same.

Theorem 9413.455 (Zeta zero core completeness forces the fixed surface)**.** *Assume mirror closure, mirror-even zero signature, core completeness of the zeta zero source, and that δ is a public zero observable. Then every critical-strip zero lies on the fixed surface*

$$\Re(s) = \frac{1}{2}.$$

Proof. Core completeness of $\mathcal{Z}_\zeta$ gives a factorization

$$\delta(s) = f(\Sigma_Z(s)).$$

Mirror-evenness gives

$$\Sigma_Z(J(s)) = \Sigma_Z(s),$$

so

$$\delta(J(s)) = f(\Sigma_Z(J(s))) = f(\Sigma_Z(s)) = \delta(s).$$

But the direct mirror calculation in Definition 9413.454 gives

$$\delta(J(s)) = -\delta(s).$$

Hence $\delta(s) = 0$ in the characteristic-zero real readout. This is the fixed surface $\Re(s) = 1/2$. □

Corollary 9413.456 (Core completeness gives the RH witness shape)**.** *Under the hypotheses of Definition 9413.455, the zeta zero source admits the RH fixed-half section.*

Proof. The theorem assigns the fixed-half conclusion to every critical-strip zero source row. By Definition 9413.199, this is exactly the RH fixed-half section, i.e. the constructive witness shape needed for the BEDC RH target. □

Definition 9413.457 (Layered normal-coordinate discipline)**.** *The zeta zero source satisfies* layered normal-coordinate discipline *when every displayed normal coordinate of a critical-strip zero is licensed in exactly the sense of Definition 9413.449: it either factors through the zero ontic signature Σ_Z, or it is carried by an off-critical normal-extension ledger.*

Theorem 9413.458 (Excluding off-critical normal extensions gives the fixed surface)**.** *Assume mirror closure, mirror-even zero signature, layered normal-coordinate discipline, and located fixed-half collapse. If no off-critical normal-extension ledger exists for a zeta zero source, then every critical-strip zero lies on the fixed surface.*

Proof. Let $s = \frac{1}{2} + \delta + i\tau$ be a critical-strip zero source. By layered normal-coordinate discipline, the normal coordinate is either a signature readout or an off-critical normal extension. In the first case, the proof of Definition 9413.455 applies: mirror-evenness makes the signature value the same at s and $J(s)$, while the normal coordinate is mirror-odd, so $\delta = 0$.

In the second case, the normal coordinate is licensed by $\mathsf{NormalExtLedger}_\zeta(s, \delta)$. The hypothesis excluding off-critical normal-extension ledgers rules out $\delta \# 0$. Located fixed-half collapse then gives $\delta = 0$. Hence in all licensed cases the zero source lies on $\Re(s) = 1/2$. □

Theorem 9413.459 (Core expression is distinct from core completeness)**.** *The claim that every licensed object carries the core commonality basis is weaker than the claim that a specific object is core-complete.*

Proof. Definition 9413.343 supplies the core rows to licensed objects by object induction. This only gives the existence of observation-time, distinction-atom, and symmetry-persistence ledgers. Core completeness additionally asserts that every public observable factors through those ledgers. A symmetric two-point carrier can have observation, distinction, and symmetry while still carrying a non-fixed mirror orbit. Thus core expression does not collapse all symmetry orbits to fixed orbits; completeness and faithfulness are the additional claims. □

Theorem 9413.460 (No-fourth-ledger is the zeta core-completeness boundary)**.** *For the zeta zero source, the no-fourth-ledger row is the boundary form of core completeness with respect to the normal displacement.*

Proof. If $\delta(s) \# 0$ occurred on a zero source row, then Definition 9413.431 supplies a finite prime-window defect. This defect is a public witness of off-fixed displacement unless the zero carrier excludes it. The no-fourth-ledger row of Definition 9413.437 expresses precisely that exclusion: δ cannot stand as an independent observable outside the zero signature. Therefore the no-fourth-ledger row is the normal-displacement instance of core completeness. □

Corollary 9413.461 (Defect-zero incompatibility is the analytic form)**.** *The analytic form of the zeta core-completeness boundary is defect-zero incompatibility.*

Proof. The visible off-fixed displacement is read by finite prime-window defects. To prove that the zero source is core-complete with respect to this displacement, one must show that such a defect cannot coexist with the completed zeta zero ledger. This is exactly the role of Definition 9413.202 and Definition 9413.352. □

Theorem 9413.462 (Core commonality completion principle)**.** *The unified-reality programme supplies the common ontic basis for licensed objects. RH requires the stronger zeta-zero statement that the zero source is complete and faithful for that basis.*

Proof. The common basis is Definition 9413.445. Its object-level transport is Definition 9413.343. The zeta zero completion target is Definition 9413.453. If that target holds for the normal displacement, then Definition 9413.455 and Definition 9413.456 give the RH section. If it fails, the failure is an independent observable in the zero carrier, which is exactly the fourth-ledger boundary identified by Definition 9413.460. □

Concretized in §9100: Common-carrier completion boundaries are realized as finite source, observable-ledger, dyadic tolerance, stream-window, regular-readback, real-seal, transport, replay, provenance, and local NameCert rows.

9413.34 Prime-Sieve Odd Exactness

The fixed-half surface admits several equivalent readouts: mirror fixedness, unit centered prime amplitude, absence of real Laplace bias, and vanishing normal displacement. These equivalences identify the classifier surface. They do not, by themselves, prove that the zeta zero ledger lands on that surface. The RH-facing bridge is the exactness claim that a prime-visible mirror-odd defect cannot survive as a zero row inside the completed zeta carrier.

Definition 9413.463 (Fixed-surface readout cycle)**.** *For the zeta mirror*

$$J(s) := 1 - \overline{s}$$

and the normal displacement

$$\delta(s) := \Re(s) - \frac{1}{2},$$

the fixed-surface readout cycle *is the package of equivalent surface predicates*

$$\Re(s) = \frac{1}{2}, \qquad |p^{1/2-s}| = 1 \text{ for every displayed prime } p,$$

$$e^{-\delta(s)x} = 1 \text{ for logarithmic time } x > 0, \qquad J(s) = s.$$

It is a classifier package for the half surface, not a zero-exclusion principle.

Theorem 9413.464 (The fixed-surface cycle is a readout equivalence)**.** *The fixed-surface readout cycle expresses the same centered surface read in prime, Laplace, and mirror coordinates. It does not imply*

$$\mathsf{ZetaZero}(s) \to \Re(s) = \frac{1}{2}.$$

Proof. Definition 9413.313 identifies $\Re(s) = 1/2$ with $J(s) = s$. Definition 9413.317 identifies the same surface with the prime-unit and no-Laplace-bias readouts. None of these equivalences contains the hypothesis $\mathsf{ZetaZero}(s)$ or a bridge from the zero ledger to the fixed surface. That bridge is the defect-zero incompatibility obligation of Definition 9413.202. □

Definition 9413.465 (Mirror-odd prime-defect vector)**.** *The* mirror-odd prime-defect vector *is the prime-indexed readout*

$$\mathcal{D}(s) := \big(d_p(s)\big)_{p \in \mathsf{Prime}^{\uparrow}}, \qquad d_p(s) := \log |p^{1/2-s}|.$$

For a finite prime window S, its finite shadow is

$$D_S(s) := \sum_{p \in S} d_p(s).$$

Thus D_S is the window defect row of Definition 9413.200.

Definition 9413.466 (Time-fibre even and odd normal reads)**.** *For a fixed directed time coordinate τ, the mirror pair of normal reads is*

$$\delta, \qquad -\delta.$$

Its even and odd normal components are

$$\delta_{\text{even}} := \frac{\delta + (-\delta)}{2} = 0,$$

and

$$\delta_{\mathrm{odd}} := \frac{\delta - (-\delta)}{2} = \delta.$$

Thus the mirror-even component of a two-read time fibre is already on the fixed surface; the RH-facing issue is whether the mirror-odd component can remain nonzero on a zero row.

Definition 9413.467 (Odd zero defect)**.** *An* odd zero defect *for the completed zeta carrier is the coexistence of a critical-strip zero row with a nonzero finite prime-window mirror-odd defect:*

$$\mathsf{OddZeroDefect}_\zeta := \sum_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}} (D_S(s)\#0) \right).$$

The corresponding refusal row is

$$\mathsf{NoOddZeroDefect}_\zeta := \neg\mathsf{OddZeroDefect}_\zeta.$$

Definition 9413.468 (Even defect energy)**.** *For a nonempty finite prime window S, the* defect energy *and normalized defect energy are*

$$E_S(s) := D_S(s)^2, \qquad \overline{E}_S(s) := \left(\overline{D}_S(s)\right)^2.$$

The corresponding refusal row is

$$\mathsf{NoEvenDefectEnergy}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} (\overline{E}_S(s)\#0) \to \bot \right).$$

Theorem 9413.469 (Prime-defect is mirror-odd)**.** *For $s = 1/2 + \delta + i\tau$ and $J(s) = 1 - \overline{s}$,*

$$d_p(s) = -\delta \log p, \qquad d_p(J(s)) = -d_p(s).$$

Consequently

$$\mathcal{D}(J(s)) = -\mathcal{D}(s).$$

Proof. The centered prime amplitude is

$$p^{1/2-s} = p^{-\delta} p^{-i\tau},$$

so

$$d_p(s) = \log |p^{1/2-s}| = -\delta \log p.$$

Since

$$J(s) = \frac{1}{2} - \delta + i\tau,$$

the normal displacement of $J(s)$ is $-\delta$. Substituting this into the same formula gives

$$d_p(J(s)) = \delta \log p = -d_p(s).$$

The vector statement is obtained componentwise. □

Theorem 9413.470 (Defect energy is mirror-even)**.** *For every nonempty finite prime window S,*

$$E_S(J(s)) = E_S(s), \qquad \overline{E}_S(J(s)) = \overline{E}_S(s).$$

Proof. By Definition 9413.469,

$$D_S(J(s)) = -D_S(s).$$

Squaring gives

$$E_S(J(s)) = D_S(J(s))^2 = D_S(s)^2 = E_S(s).$$

The normalized statement follows in the same way from

$$\overline{D}_S(J(s)) = -\overline{D}_S(s),$$

which is the normalized form of the same computation. □

Theorem 9413.471 (Prime-window defect measures the odd normal read)**.** *For $s = 1/2 + \delta + i\tau$ and every nonempty finite prime window S, the normalized prime-window defect is the negative odd normal component:*

$$\overline{D}_S(s) = -\delta_{\mathrm{odd}}.$$

Proof. By Definition 9413.466, the odd normal component is $\delta_{\mathrm{odd}} = \delta$. The normalized defect computation of Definition 9413.167 gives

$$\overline{D}_S(s) = -\delta.$$

Therefore $\overline{D}_S(s) = -\delta_{\mathrm{odd}}$. □

Corollary 9413.472 (Normalized defect energy measures squared odd normal read)**.** *For $s = 1/2 + \delta + i\tau$ and every nonempty finite prime window S,*

$$\overline{E}_S(s) = \delta_{\mathrm{odd}}^2.$$

Proof. By Definition 9413.471,

$$\overline{D}_S(s) = -\delta_{\mathrm{odd}}.$$

Squaring gives $\overline{E}_S(s) = \delta_{\mathrm{odd}}^2$. □

Corollary 9413.473 (Finite windows separate nonfixed mirror orbits)**.** *If $\delta(s)\#0$ and S is a nonempty finite prime window, then*

$$D_S(s)\#0.$$

Proof. Summing the formula of Definition 9413.469 over S gives

$$D_S(s) = -\delta(s) \sum_{p \in S} \log p.$$

The displayed prime rows have positive logarithm, and S is nonempty. Thus the finite sum is positive, so apartness of $\delta(s)$ from zero gives apartness of $D_S(s)$ from zero. This is the same visibility row as Definition 9413.201 and Definition 9413.320. □

Definition 9413.474 (Parity row as a finite prime sieve)**.** *The parity split of the ordinary Dirichlet series is the finite prime-window sieve associated with the prime 2:*

$$\sum_{n \geq 1} n^{-s} = \sum_{\substack{n \geq 1 \\ 2 \nmid n}} n^{-s} + \sum_{\substack{n \geq 1 \\ 2 | n}} n^{-s}.$$

The even part is $2^{-s}\zeta(s)$, and the alternating row is

$$\eta(s) = \sum_{n \geq 1} (-1)^{n-1} n^{-s} = (1 - 2^{1-s})\zeta(s).$$

Thus parity is the v_2 face of the prime-factorization ledger.

Definition 9413.475 (Finite prime-sieve tower)**.** *For a finite prime window S, the S-sieved* zeta readout *is*

$$\zeta_S(s) := \zeta(s) \prod_{p \in S} (1 - p^{-s}).$$

On the half-plane where the Euler and Dirichlet expansions are absolutely convergent, this readout is

$$\zeta_S(s) = \sum_{\gcd(n, \prod_{p \in S} p) = 1} n^{-s}.$$

The finite prime-sieve tower is the family of these readouts as S ranges over finite prime windows.

Theorem 9413.476 (Parity is the first finite sieve shadow)**.** *The parity construction of Definition 9413.474 is the $S = \{2\}$ instance of the finite prime-sieve tower.*

Proof. For $S = \{2\}$,

$$\zeta_S(s) = \zeta(s)(1 - 2^{-s})$$

removes the terms divisible by 2 in the absolutely convergent Dirichlet region. The alternating eta row instead applies the signed parity combination

$$(1 - 2^{1-s})\zeta(s).$$

Both rows are finite manipulations of the same v_2 coordinate. Hence parity is not separate from prime sieving; it is the lowest finite prime-window face of the same distinction ledger. □

Definition 9413.477 (Prime-sieve odd exactness)**.** *The* prime-sieve odd exactness *target for the completed zeta carrier is the row*

$$\mathsf{PrimeSieveOddExactness}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}} (D_S(s) \# 0) \to \bot \right).$$

It says that a mirror-odd prime-window defect visible at a finite sieve level cannot coexist with the completed zero ledger.

Theorem 9413.478 (No odd zero defect is prime-sieve odd exactness)**.** *For the completed zeta carrier,*

$$\mathsf{NoOddZeroDefect}_\zeta$$

is the same refusal row as prime-sieve odd exactness.

Proof. Unfold Definition 9413.467. Its positive data are a zero row, a critical-strip row, and an existential finite prime-window defect $D_S(s)\#0$. This is exactly the antecedent excluded by Definition 9413.477. The two names therefore describe the same carrier refusal: no completed zeta zero may carry a nonzero mirror-odd finite prime-window defect. □

Theorem 9413.479 (Odd defect refusal is even energy refusal)**.** *For the completed zeta carrier,*

$$\mathsf{NoOddZeroDefect}_\zeta \iff \mathsf{NoEvenDefectEnergy}_\zeta.$$

Proof. For every nonempty finite prime window S, the normalization row $L_S\#0$ of Definition 9413.165 identifies nonzero $D_S(s)$ with nonzero $\overline{D}_S(s)$. Squaring over the located real readout identifies nonzero $\overline{D}_S(s)$ with nonzero

$$\overline{E}_S(s) = \overline{D}_S(s)^2.$$

Therefore the positive data excluded by Definition 9413.467 are equivalent to the positive data excluded by Definition 9413.468. The two refusal rows are the same obstruction, written once as signed odd defect and once as mirror-even energy. □

Corollary 9413.480 (Mirror pairs do not cancel defect energy)**.** *If s and $J(s)$ are mirror-paired zero-source rows in one time fibre and $\delta(s)\#0$, then every nonempty finite prime window carries nonzero mirror-even defect energy.*

Proof. The signed defects cancel:

$$D_S(s) + D_S(J(s)) = 0,$$

by Definition 9413.469. However, Definition 9413.470 gives

$$E_S(s) + E_S(J(s)) = 2E_S(s).$$

By Definition 9413.472, the normalized energy is $\delta(s)^2$, hence nonzero when $\delta(s)\#0$. Thus a mirror pair can cancel the sign of the odd defect but not its even energy. □

Theorem 9413.481 (Prime-sieve odd exactness is defect-zero incompatibility)**.** *Prime-sieve odd exactness is the finite-sieve, mirror-odd form of the defect-zero incompatibility route.*

Proof. By Definition 9413.478, prime-sieve odd exactness may be read as $\mathsf{NoOddZeroDefect}_\zeta$. The existential finite-window component in Definition 9413.477 is precisely the prime-defect row of Definition 9413.200. The remaining hypotheses place that defect in the same completed source as a critical-strip zero row. Excluding this coexistence is exactly the content of Definition 9413.202, also identified as the hard point in Definition 9413.352 and as the analytic form of core completion in Definition 9413.461. □

Definition 9413.482 (Odd-exactness proof packet)**.** *An* odd-exactness proof packet *for the zeta route consists of the rows*

MirrorOddDecomposition, FiniteWindowSeparation, SieveContinuationExactness, ZeroIncompatibility.

The first row records $\mathcal{D}(J(s)) = -\mathcal{D}(s)$. The second row records that nonzero normal displacement is visible in a finite prime window. The third row records that a finite sieve defect remains a source-compatible defect after passage through the completed zeta carrier. The fourth row records that such a defect is incompatible with a zero ledger.

Theorem 9413.483 (Odd-exactness packets close the RH section route)**.** *Assume an odd-exactness proof packet, zero-ledger soundness, and the located fixed-half collapse row. Then the RH fixed-half section exists.*

Proof. Let s carry a critical-strip zero row. If $\delta(s)\#0$, finite-window separation supplies a nonempty finite prime window S with $D_S(s)\#0$. Sieve-continuation exactness keeps this finite defect source-compatible with the completed zeta carrier. Zero incompatibility then contradicts the zero ledger. Hence the off-half apartness row is excluded. The located fixed-half collapse row turns this exclusion into $\mathsf{OnCritLine}(s)$, and Definition 9413.199 reads the result as the RH section. □

Corollary 9413.484 (The obstruction is analytic exactness)**.** *Within the prime-sieve route, the obstruction is not the equivalence of the fixed-surface readouts and not the finite-window defect formula. It is the exactness of finite prime-sieve defect through the completed zeta carrier.*

Proof. The readout equivalences are supplied by Definition 9413.464. The finite-window visibility row is supplied by Definition 9413.473. What remains is the passage from a visible finite prime-window defect to zero-ledger exclusion in the completed zeta carrier. This is the SieveContinuationExactness and ZeroIncompatibility part of Definition 9413.482. □

Theorem 9413.485 (The quartic laboratory marks the zero-relevance boundary)**.** *The quartic spectral carrier shows why visibility of a defect must be strengthened to zero relevance before it can serve as an RH bridge.*

Proof. In the quartic carrier, branch data alone do not yield the endpoint zero law. The zero law also uses the dominance and local-interference ledgers, as isolated in Definition 9413.99 and scoped by Definition 9413.209. The zeta analogue is that finite prime-window defect is visible but not yet zero-relevant. Prime-sieve odd exactness is the missing zero-relevance row for the completed zeta carrier. □

Corollary 9413.486 (Prime-sieve exactness is the no-fourth-ledger test)**.** *For the zeta zero source, prime-sieve odd exactness is the analytic test that the normal displacement is not an independent fourth observable.*

Proof. Definition 9413.462 identifies the RH-facing completion problem as the claim that the zeta zero source is complete and faithful for the core basis. The normal displacement is the candidate fourth observable of Definition 9413.454. By Definition 9413.484, the only remaining prime-sieve obstruction is whether finite odd defect can coexist with a zero ledger. Therefore prime-sieve odd exactness is the analytic no-fourth-ledger test for this observable. □

9413.35 Recursive Parity Tower

Prime-sieve odd exactness separates the signed mirror-odd defect from its mirror-even energy. The same separation persists after the Euler local factor is resolved by repeated parity splitting. A prime supplies the dimension; the exponent-depth inside that prime dimension supplies a binary recursion tree. The normalized radial defect read by every finite part of this tree is the same normal displacement from the fixed-half surface.

Definition 9413.487 (Local parity recursion)**.** *For a formal variable x, the finite parity recursion packet is*

$$P_m(x) := \prod_{j=0}^{m} (1 + x^{2^j}).$$

It satisfies

$$(1 - x)P_m(x) = 1 - x^{2^{m+1}},$$

and hence

$$P_m(x) = \frac{1 - x^{2^{m+1}}}{1 - x}$$

whenever $1 - x$ is invertible. On a convergent analytic readout with $|x| < 1$, the limiting row is

$$\frac{1}{1 - x} = \prod_{j \geq 0} (1 + x^{2^j}).$$

Proof. The identity follows by the telescoping chain

$$(1-x)(1+x) = 1-x^2, \qquad (1-x^2)(1+x^2) = 1-x^4,$$

and so on through the factor $1+x^{2^m}$. Multiplying these rows gives $(1-x)P_m(x) = 1-x^{2^{m+1}}$. If $|x|<1$, the terminal term $x^{2^{m+1}}$ tends to zero in the displayed analytic readout. □

Theorem 9413.488 (Euler local factors are recursive parity trees)**.** *On the local convergence region $\Re(s)>0$, each Euler local factor admits the recursive parity expansion*

$$(1-p^{-s})^{-1} = \prod_{j\geq 0}\left(1+p^{-2^j s}\right).$$

On the Euler-product convergence region $\Re(s)>1$ this gives the two-indexed readout

$$\zeta(s) = \prod_{p\in\mathsf{Prime}^{\uparrow}}\prod_{j\geq 0}\left(1+p^{-2^j s}\right).$$

Proof. Apply Definition 9413.487 with $x = p^{-s}$. The local condition $\Re(s)>0$ gives $|p^{-s}|<1$. The global statement is the same local identity inserted into the ordinary Euler-product row on $\Re(s)>1$. □

Definition 9413.489 (Recursive parity tower)**.** *The recursive parity tower of the zeta Euler face is the two-indexed node family*

$$\mathsf{ParityTower}_{\zeta} := \mathsf{Prime}^{\uparrow}\times\mathsf{Nat}^{\uparrow},$$

where (p,j) records the prime dimension p and the binary depth j inside that dimension. Its centered node channel is

$$U_{p,j}(s) := p^{2^j(1/2-s)}.$$

For a nonempty finite node family $A\subset\mathsf{ParityTower}_{\zeta}$, define

$$D_A^{\mathrm{rec}}(s) := \sum_{(p,j)\in A}\log|U_{p,j}(s)|,$$

and its positive weight

$$W_A := \sum_{(p,j)\in A} 2^j\log p.$$

The normalized recursive defect and recursive defect energy are

$$\overline{D}_A^{\mathrm{rec}}(s) := \frac{D_A^{\mathrm{rec}}(s)}{W_A}, \qquad \overline{E}_A^{\mathrm{rec}}(s) := \left(\overline{D}_A^{\mathrm{rec}}(s)\right)^2.$$

Theorem 9413.490 (Recursive normalized defects are depth-invariant)**.** *If*

$$s = \frac{1}{2}+\delta+i\tau,$$

then for every nonempty finite node family $A\subset\mathsf{ParityTower}_{\zeta}$,

$$\overline{D}_A^{\mathrm{rec}}(s) = -\delta.$$

Proof. For a node (p, j),

$$U_{p,j}(s) = p^{2^j(1/2-s)} = p^{-2^j\delta} e^{-i2^j\tau \log p}.$$

Thus

$$\log |U_{p,j}(s)| = -2^j \delta \log p.$$

Summing over A gives

$$D_A^{\mathrm{rec}}(s) = -\delta \sum_{(p,j)\in A} 2^j \log p = -\delta W_A.$$

Since A is nonempty and every displayed prime logarithm is positive, W_A is a positive normalization row. Dividing by W_A gives the claim. □

Corollary 9413.491 (Recursive defect energy is depth-invariant)**.** *For $s = 1/2 + \delta + i\tau$ and every nonempty finite node family $A \subset \mathsf{ParityTower}_\zeta$,*

$$\overline{E}_A^{\mathrm{rec}}(s) = \delta^2.$$

Consequently the signed recursive defect is mirror-odd, while the recursive defect energy is mirror-even.

Proof. The first statement is the square of Definition 9413.490. Under the mirror $J(s) = 1 - \overline{s}$, the normal displacement changes from δ to $-\delta$. Hence $\overline{D}_A^{\mathrm{rec}}$ changes sign, while $\overline{E}_A^{\mathrm{rec}}$ is unchanged. □

Definition 9413.492 (Recursive odd energy refusal)**.** *The recursive odd energy refusal row is*

$$\mathsf{NoRecursiveOddEnergyZero}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{A:\mathsf{FiniteParitySubtree}_{\neq\varnothing}} (\overline{E}_A^{\mathrm{rec}}(s)\#0) \to \bot \right).$$

Here $\mathsf{FiniteParitySubtree}_{\neq\varnothing}$ denotes a nonempty finite node family in $\mathsf{ParityTower}_\zeta$. The row says that no completed zeta zero may carry nonzero mirror-even energy anywhere in the recursive parity tower.

Theorem 9413.493 (Recursive odd energy refusal is conditionally sufficient)**.** *Together with the located fixed-half collapse row, the recursive odd energy refusal row is sufficient for the RH fixed-half section.*

Proof. Let s carry a critical-strip zero row and write

$$s = \frac{1}{2} + \delta + i\tau.$$

Suppose $\delta\#0$. Choose any nonempty finite node family $A \subset \mathsf{ParityTower}_\zeta$. By Definition 9413.491,

$$\overline{E}_A^{\mathrm{rec}}(s) = \delta^2,$$

which is nonzero in the located real readout when $\delta\#0$. This supplies the positive data excluded by Definition 9413.492. Therefore $\neg(\delta\#0)$. The located fixed-half collapse row converts this into $\delta = 0$, hence $\Re(s) = 1/2$ by the fixed-surface calculation of Definition 9413.313. □

Corollary 9413.494 (Recursive parity does not hide an off-half defect)**.** *Within the recursive parity tower, an off-half normal displacement is not a defect of one depth layer only. Every nonempty finite subtree reads the same normalized odd defect and the same mirror-even defect energy.*

Proof. The normalized odd defect is depth-invariant by Definition 9413.490, and the energy is depth-invariant by Definition 9413.491. Thus recursion through parity depth cannot cancel, dilute, or relocate the normal displacement. Any remaining obstruction is the same zero-relevance obstruction isolated in Definition 9413.484, now read across the full recursive parity tower. □

9413.36 Parity Closure Tower Certificate

The recursive parity tower displays the normal defect at every finite prime-adic depth, but that visibility is not yet a zero-exclusion theorem. The missing statement can be sharpened by separating two notions that the functional equation tends to blur. Mirror symmetry gives paired zero packets; it does not by itself identify a packet with its mirror inside the same routing closure. A parity closure tower is the stronger object: the mirror packet lies in the same recursive routing closure, while the normal defect is routing-invariant and parity-odd.

Definition 9413.495 (Abstract parity-routing packet)**.** *An* abstract parity-routing packet *consists of a topological space X, a homeomorphism*

$$\mathcal{R} : X \to X,$$

an involution

$$P : X \to X, \qquad P^2 = \mathrm{id},$$

and a continuous real readout

$$\Delta : X \to \mathsf{Real}^{\uparrow}$$

such that

$$P\mathcal{R}P = \mathcal{R}^{-1},$$

$$\Delta(\mathcal{R}x) = \Delta(x),$$

and

$$\Delta(Px) = -\Delta(x).$$

The map $\mathcal{R}$ is the packet-routing step, P is parity reflection, and Δ is the signed normal defect read by the packet.

Theorem 9413.496 (Parity closure kills odd defect)**.** *Let $(X, \mathcal{R}, P, \Delta)$ be an abstract parity-routing packet. If*

$$Px \in \overline{\{\mathcal{R}^m x : m \in \mathbb{Z}\}},$$

then

$$\Delta(x) = 0.$$

Proof. The closure hypothesis gives a sequence of integers m_j such that

$$\mathcal{R}^{m_j} x \to Px.$$

By continuity of Δ,

$$\Delta(Px) = \lim_j \Delta(\mathcal{R}^{m_j} x).$$

Routing invariance gives

$$\Delta(\mathcal{R}^{m_j} x) = \Delta(x)$$

for every j, hence

$$\Delta(Px) = \Delta(x).$$

Parity oddness gives

$$\Delta(Px) = -\Delta(x).$$

Thus $\Delta(x) = -\Delta(x)$, and the located real readout gives $\Delta(x) = 0$. □

Definition 9413.497 (Eta return loop and cut coordinate)**.** *For*

$$\eta(s) = \sum_{n\geq 1}(-1)^{n-1}n^{-s},$$

write

$$v_N(s) := (-1)^{N-1}N^{-s}, \qquad P_N(s) := \sum_{n\leq N} v_n(s), \qquad R_N(s) := \sum_{n>N} v_n(s).$$

On a zero row $\eta(s) = 0$,

$$P_N(s) + R_N(s) = 0.$$

The local cut coordinate at the Nth segment is the row

$$P_{N-1}(s) + \theta_N(s)v_N(s) = 0.$$

Theorem 9413.498 (Eta cut coordinates satisfy exact routing recursion)**.** *On an eta zero row, the cut coordinates satisfy*

$$\theta_N(s) = 1 - \left(\frac{N}{N+1}\right)^s \theta_{N+1}(s).$$

Proof. By Definition 9413.497,

$$\theta_N(s) = -\frac{P_{N-1}(s)}{v_N(s)}.$$

Since $P_{N-1} + v_N + R_N = 0$,

$$P_{N-1} = -v_N - R_N,$$

so

$$\theta_N(s) = 1 + \frac{R_N(s)}{v_N(s)}.$$

Also $R_N = v_{N+1} + R_{N+1}$, and the definition of θ_{N+1} gives

$$R_N = v_{N+1}\theta_{N+1}.$$

Finally

$$\frac{v_{N+1}}{v_N} = -\left(\frac{N}{N+1}\right)^s.$$

Substitution gives the displayed recursion. □

Definition 9413.499 (Parity closure tower certificate)**.** *For a critical-strip zeta zero $s = 1/2 + \delta + i\tau$, a* zeta parity closure tower certificate *consists of a canonical zero-relevant packet x_s in an abstract parity-routing packet $(X, \mathcal{R}, P, \Delta)$, together with rows:*

$$\mathsf{ActiveCapture}(x_s, s), \qquad \Delta(x_s) = \delta,$$

$$Px_s \in \overline{\{\mathcal{R}^m x_s : m \in \mathbb{Z}\}},$$

and a zero-relevance ledger saying that x_s is the packet whose dominance, endpoint, tail, or solenoid rows control the completed zero. The equality $\Delta(x_s) = \delta$ is the defect-invariance row; the closure row is stronger than the statement that s and $1 - \overline{s}$ are both zeros.

Theorem 9413.500 (Parity closure tower certificates are conditionally sufficient)**.** *If every critical-strip zeta zero carries a zeta parity closure tower certificate, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The certificate supplies x_s with

$$\Delta(x_s) = \delta$$

and

$$Px_s \in \overline{\{\mathcal{R}^m x_s : m \in \mathbb{Z}\}}.$$

By Definition 9413.496,

$$\Delta(x_s) = 0.$$

Hence $\delta = 0$, so $\Re(s) = 1/2$. Since z was arbitrary, Definition 9413.199 supplies the RH fixed-half section. □

Corollary 9413.501 (Functional equation pairing is not packet closure)**.** *The completed zeta functional equation supplies mirror pairing of zero coordinates, but the parity closure tower certificate requires a stronger packet-level identification.*

Proof. The mirror

$$J(s) := 1 - \overline{s}$$

sends

$$\frac{1}{2} + \delta + i\tau \quad \text{to} \quad \frac{1}{2} - \delta + i\tau.$$

The completed functional equation and conjugation symmetry place zeros in mirror-related positions. This gives a paired-zero row

$$s \longmapsto J(s).$$

It does not by itself give

$$Px_s \in \overline{\{\mathcal{R}^m x_s : m \in \mathbb{Z}\}},$$

nor does it identify the active endpoint, Rouché, solenoid, or dominance packet for s with its mirror inside one recursive closure. Thus the parity closure certificate isolates the noncircular bridge left after ordinary functional-equation symmetry has been accounted for. □

Corollary 9413.502 (Relation to endpoint and solenoid certificates)**.** *Endpoint bisection and solenoid Lee–Yang rows are candidate mechanisms for producing the active capture and zero-compatible routing closure required by a parity closure tower certificate.*

Proof. The endpoint bisection certificate of Definition 9413.762 identifies a finite endpoint window, a distinguished segment, and a Rouché ledger. These are possible sources for $\mathsf{ActiveCapture}(x_s, s)$. The solenoid-compatible resonance lift of Definition 9413.780 identifies a compact phase carrier and forbids hidden noncompact real scale. This is a possible source for the zero-compatible routing closure. The parity closure tower certificate records the abstract consequence common to these mechanisms: once the signed defect is both routing-invariant and parity-odd inside the same closure, it must vanish. □

Definition 9413.503 (Parity closure bridge obligations)**.** *The parity closure route leaves four bridge obligations:*

$$\mathsf{ActiveCapture}, \qquad \mathsf{DefectInvariant}, \qquad \mathsf{ParityClosure}, \qquad \mathsf{ZeroCompatibleBound}.$$

Active capture constructs the zero-relevant packet from the completed zero ledger. Defect invariance proves that its signed readout is $\Re(s) - 1/2$. Parity closure places the mirror packet in the same routing closure. Zero-compatible bound or compactness prevents the closure from using a hidden noncompact real scale.

Corollary 9413.504 (The bridge obligations are the proof load)**.** *The parity closure route proves a conditional RH theorem, but the zeta-specific proof load is the construction of the four bridge obligations in Definition 9413.503.*

Proof. Definition 9413.496 is an abstract topological theorem, and Definition 9413.500 is its zeta-facing conditional use. Neither theorem constructs the canonical packet space, its routing, its dominance ledger, or its compact zero-compatible carrier from the completed zeta row. Those tasks are exactly the four bridge obligations. Consequently the remaining work is not another restatement of the fixed-half surface, but the construction of a zero-relevant recursive packet tower satisfying those obligations. □

9413.37 Prime Charge, Parity, and Phase Dictionary

The parity closure route benefits from a sharper projection dictionary. The three primary reads are not parity, energy, and an unspecified third observable. The more stable triad is prime charge, parity, and phase. Energy is a derived second-order observable: it is the square readout of a failure of these three rows to close.

Definition 9413.505 (Prime charge, parity, and phase projections)**.** *The* prime charge–parity–phase dictionary *assigns the three ontic roles to their zeta-facing projections as follows:*

Ontic role	***Primary projection***	***Zeta read***
Distinction or primality	*prime charge, label, fibre*	p, v_p, S, (p, j), $\prod_{p\in S} \mathbb{Z}_p$
Symmetry or unit boundary	*parity, mirror, unit circle*	$J(s) = 1 - \overline{s}$ *and* $\delta \mapsto -\delta$
Time or recursion	*phase, action, routing*	$e^{-i\tau h_a}$ *and* $\mathcal{R} : X \to X$

For a node $a = (p, j)$, the prime charge determines the positive height

$$h_a := 2^j \log p.$$

For $s = 1/2 + \delta + i\tau$, the local channel is

$$U_a(s) := e^{-(s-1/2)h_a} = e^{-\delta h_a} e^{-i\tau h_a}.$$

Definition 9413.506 (Prime charge)**.** *The* prime charge *of a displayed resonance node is its discrete prime-register row. In a finite prime family S, this register is read through the compact fibre*

$$\prod_{p\in S} \mathbb{Z}_p$$

behind the solenoid projection

$$\Sigma_S \twoheadrightarrow \mathbb{T}.$$

It records which prime slice, valuation depth, or parity depth a node belongs to. It is therefore a distinction read, not a real positive scale.

Theorem 9413.507 (Prime charge determines the height ledger)**.** *For every displayed node $a = (p, j)$, the prime charge determines the Hamiltonian height*

$$h_a = 2^j \log p,$$

and hence determines the normal defect and phase rows

$$\log |U_a(s)| = -\delta h_a, \qquad \arg U_a(s) \equiv -\tau h_a \pmod{2\pi}.$$

Proof. The node label $a = (p, j)$ records both the prime slice p and the parity depth j. The height row is therefore $h_a = 2^j \log p$. Substituting this into

$$U_a(s) = e^{-\delta h_a} e^{-i\tau h_a}$$

gives the modulus and argument readouts. □

Definition 9413.508 (Energy as derived closure-defect observable)**.** *For a finite active packet A, the first-order normal defect of a node is*

$$d_a(s) := \log |U_a(s)| = -\delta h_a.$$

The associated energy is the second-order read

$$\mathcal{E}_A(s) := \sum_{a \in A} d_a(s)^2.$$

It is not a primary ontic projection. It forgets the parity orientation of the defect and records only its magnitude.

Theorem 9413.509 (Energy is the square of triadic closure failure)**.** *For $s = 1/2 + \delta + i\tau$ and a finite active packet A,*

$$\mathcal{E}_A(s) = \delta^2 \sum_{a \in A} h_a^2.$$

Under the mirror $J(s) = 1 - \overline{s}$, the first-order defect changes sign, while the energy is unchanged.

Proof. The defect formula gives $d_a(s) = -\delta h_a$, so

$$\mathcal{E}_A(s) = \sum_{a \in A} \delta^2 h_a^2 = \delta^2 \sum_{a \in A} h_a^2.$$

The mirror sends δ to $-\delta$, so d_a changes sign. Squaring removes the sign, hence the energy is mirror-even. □

Definition 9413.510 (Triadic closure defect)**.** *A* triadic closure defect *for a zero-relevant packet is the failure of the prime-charge ledger, parity/unit-boundary row, and phase-routing row to close in one packet. In the local zeta channel the defect has first-order read*

$$\Omega_a(s) := \log |U_a(s)| = -\delta h_a.$$

The unit-boundary row is

$$\Omega_a(s) = 0,$$

or equivalently

$$|U_a(s)| = 1.$$

Theorem 9413.511 (Unit triadic closure forces the fixed-half row)**.** *Let a be a displayed node with $h_a > 0$. If*

$$|U_a(s)| = 1$$

for $s = 1/2 + \delta + i\tau$, then

$$\delta = 0.$$

Proof. The modulus read is

$$|U_a(s)| = e^{-\delta h_a}.$$

If this equals 1, then

$$-\delta h_a = 0.$$

Since $h_a > 0$, the located real readout gives $\delta = 0$. □

Definition 9413.512 (Oriented triadic closure tower)**.** *An* oriented triadic closure tower *for a critical-strip zero $s = 1/2 + \delta + i\tau$ is a zero-relevant recursive packet tower carrying:*

$$\mathsf{PrimeChargeLedger}, \qquad \mathsf{ParityOddDefect}, \qquad \mathsf{PhaseRouting},$$

$$\mathsf{WindingCapture}, \qquad \mathsf{SolenoidGaugeDiscipline}, \qquad \mathsf{EnergyOrUnitBound}.$$

The winding row records Rouché or local-degree capture of the zero. The solenoid row permits phase and prime-register fibre information, but not a hidden real positive scale. The energy or unit-bound row prevents the triadic closure defect from remaining nonzero along the tower.

Corollary 9413.513 (Oriented triadic closure is a proof-role refinement)**.** *The oriented triadic closure tower refines the parity closure tower by separating the three primary projections from the derived energy observable.*

Proof. The parity closure tower of Definition 9413.499 uses a signed defect Δ, routing invariance, and parity oddness. The present dictionary identifies where these rows come from. Prime charge supplies h_a, phase routing supplies $e^{-i\tau h_a}$ and the successor relation, and parity supplies the sign change $\delta \mapsto -\delta$. Energy appears only after the first-order defect is squared. Thus the oriented triadic tower keeps the primary projection data visible instead of treating energy as a primitive row. □

Corollary 9413.514 (Solenoid fibres carry charge but not real scale)**.** *The solenoid fibre is the natural carrier for prime charge, while the factor $e^{-\delta h_a}$ is a noncompact scale defect rather than a legal hidden charge.*

Proof. By Definition 9413.506, the hidden fibre behind a finite solenoid phase projection is $\prod_{p \in S} \mathbb{Z}_p$. This records prime-register information. By Definition 9413.781, a compact solenoid phase carrier has no nontrivial continuous positive real character. Therefore the real factor $e^{-\delta h_a}$ cannot be absorbed into the prime-register fibre. It is precisely the closure defect measured by Definition 9413.510. □

Corollary 9413.515 (RH-facing use of the projection dictionary)**.** *In the zeta route, the proof-relevant projections are prime charge, parity, phase routing, winding capture, and the derived energy bound.*

Proof. Prime charge determines the heights h_a. Parity records the sign of δ. Phase routing records the unitary part $e^{-i\tau h_a}$ and the recursive successor structure. Winding capture is required to make the packet zero-relevant rather than merely visible. Energy detects whether a nonzero triadic closure defect is being carried along a cofinal tower. These are exactly the proof-role components named in Definition 9413.512. □

Concretized in §8995: PrimeParityPhaseUp realizes the prime-charge, parity, phase, height-ledger, and closure-defect rows as one finite NameCert packet.

9413.38 Triadic Closure Tower Program

The projection dictionary can be organized as a constructive tower. The tower does not claim an unconditional proof of the Riemann hypothesis. Its role is to state the certificate whose construction would discharge the zero-rigidity problem. Each layer records four rows:

$$\Omega_n := (O_n, G_n, I_n, F_n),$$

where O_n is the layer object, G_n is the generation rule from the preceding layer, I_n is the invariant retained by the layer, and F_n is the information lost in the projection fibre. The discipline is that no later layer may introduce an unledgered object; every carrier must be generated from previous rows and the triadic roles of distinction, symmetry, and time.

Definition 9413.516 (Triadic construction layer)**.** *A* triadic construction layer *is a packet*

$$\Omega_n = (O_n, G_n, I_n, F_n).$$

The generation row

$$G_n : O_n \rightsquigarrow O_{n+1}$$

is legal only when it is a displayed operation on the current carrier and the triadic roles. The invariant I_n is the datum preserved by the layer, and the fibre F_n records which data are no longer visible after projection to the next readout.

Principle 9413.517 (No unledgered later object)**.** *In a triadic construction tower, an object at layer $n+1$ is licensed only by the preceding packet and the displayed generation rule. A zero-relevant endpoint packet, solenoid carrier, selector, routing space, or energy bound cannot be introduced as an external primitive.*

Definition 9413.518 (Zeta triadic construction spine)**.** *The* zeta triadic construction spine *is the following schematic sequence of construction layers.*

Layer	***Object***	***Retained invariant***
Ontic roles	$(\mathsf{D}, \mathsf{S}, \mathsf{T})$	*triadic closure of distinction, symmetry, and time*
Projection dictionary	*prime charge, parity, phase*	*charge ledger, mirror sign, phase routing*
Euler unit core	e^{-itH}	*unitary flow when* $H = H^*$
Prime ledger	$\bigoplus_p \mathbb{Z}e_p$	$\log n = \sum_p v_p(n) \log p$
Selector and dynamics	*FRACTRAN and Zeckendorf–Godel rows*	*auditable prime-register routing and legal prime-family selection*
Solenoid carrier	$\Sigma_S = \widehat{\mathbb{Z}[S^{-1}]}$	*visible unit circle with prime-register fibre*
Endpoint packet	*Riemann–Siegel or approximate-functional-equation window*	*zero-relevant finite endpoint capture*
Routing tower	$X_\infty = \varprojlim X_n$	*compatible successor/predecessor routing*
Parity closure	$J\mathcal{R}J = \mathcal{R}^{-1}$	*odd defect forced to vanish under closure*
Energy bound	$\mathcal{E}_n = \delta^2 W_n$	*off-half growth contradicts a zero-compatible bound*

Definition 9413.519 (Zeta triadic closure tower certificate)**.** *For a critical-strip zero coordinate*

$$s = \frac{1}{2} + \delta + i\tau,$$

a zeta triadic closure tower certificate *consists of a point* x_s *in a recursive packet space* X_∞ *together with the following rows:*

$$\mathsf{ActiveCapture}(x_s, s), \qquad \mathsf{DefectExtraction}(\Delta(x_s) = \delta),$$

$$\mathsf{PrimeRouting}(X_\infty, \mathcal{R}), \qquad \mathsf{TriadicClosure}(\mathsf{PrimeCharge}, \mathsf{Parity}, \mathsf{Phase}),$$

$$\mathsf{ParityClosure}\left(Jx_s \in \overline{\{\mathcal{R}^m x_s : m \in \mathbb{Z}\}}\right),$$

and a bound-versus-growth row

$$W_n \to \infty, \qquad \mathcal{E}_n(s) \le B_s < \infty.$$

Active capture means Rouché or local-degree capture of the completed zero. Defect extraction identifies the packet defect with $\Re(s) - 1/2$*. Prime routing supplies the inverse-limit successor structure. Triadic closure permits only unit phase and prime-register gauge, not a hidden positive real scale.*

Theorem 9413.520 (Triadic closure tower certificates are conditionally sufficient)**.** *If every critical-strip zeta zero carries a zeta triadic closure tower certificate, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The certificate supplies a packet x_s with

$$\Delta(x_s) = \delta.$$

By the parity-closure row,

$$Jx_s \in \overline{\{\mathcal{R}^m x_s : m \in \mathbb{Z}\}}.$$

The abstract parity-routing theorem Definition 9413.496 gives

$$\Delta(x_s) = 0.$$

Therefore $\delta = 0$ and $\Re(s) = 1/2$. Since the zero row was arbitrary, Definition 9413.199 gives the RH fixed-half section. □

Corollary 9413.521 (Energy gives the same contradiction)**.** *In a zeta triadic closure tower certificate, the bound-versus-growth row also excludes $\delta\#0$.*

Proof. For a finite active packet A_n, Definition 9413.509 gives

$$\mathcal{E}_n(s) = \delta^2 W_n, \qquad W_n := \sum_{a \in A_n} h_a^2.$$

If $\delta\#0$ and $W_n \to \infty$, then

$$\mathcal{E}_n(s) \to \infty.$$

This contradicts the zero-compatible bound

$$\mathcal{E}_n(s) \le B_s < \infty.$$

Hence $\neg(\delta\#0)$, and located fixed-half collapse gives $\delta = 0$. □

Definition 9413.522 (Triadic closure source obligations)**.** *The triadic closure route leaves three source obligations:*

$$\mathsf{CanonicalEndpointCapture}, \qquad \mathsf{LeeYangUnitBoundary}, \qquad \mathsf{PrimeRoutingParityClosure}.$$

The first constructs, for each completed zero, a dominant endpoint packet with Rouché or winding capture. The second proves a unit-boundary or bounded-energy theorem for that packet. The third constructs the recursive routing closure and the parity relation inside one packet space.

Corollary 9413.523 (The source obligations are not fixed-half aliases)**.** *The source obligations in Definition 9413.522 are zeta-specific construction problems, not restatements of $\Re(s) = 1/2$.*

Proof. Canonical endpoint capture asks for a packet built from the completed zero ledger, as in Definition 9413.762. Lee–Yang unit boundary asks for an independent stability or compactness row, as in Definition 9413.783. Prime routing parity closure asks for the mirror packet to lie inside the same recursive closure, as in Definition 9413.499. None of these rows is the statement $\delta = 0$ written in another coordinate. Each row supplies a missing source mechanism consumed by the conditional theorem. □

Corollary 9413.524 (Selector, solenoid, and endpoint rows form one program)**.** *The Zeckendorf selector, solenoid Lee–Yang, endpoint bisection, parity closure, and energy-bound rows are components of one triadic closure program.*

Proof. The Zeckendorf selector supplies a legal prime-charge family, as in Definition 9413.834. The solenoid carrier supplies compact phase and prime-register fibre discipline. The endpoint bisection row supplies a candidate zero-relevant finite packet. The parity closure row supplies the topological odd-defect cancellation mechanism. The energy row supplies the cofinal contradiction when the normal displacement is nonzero. Together these rows instantiate the spine of Definition 9413.518. □

9413.39 Oriented Triadic Closure Tower Construction

The triadic closure program can be sharpened from a route statement into an object-construction problem. The object to construct is an oriented triadic closure tower over a completed zeta zero. Its layers are not a proof by themselves; they are the certificate rows which make the conditional RH implication noncircular.

Definition 9413.525 (Oriented triadic closure tower object)**.** *For a critical-strip zero coordinate*

$$s = \frac{1}{2} + \delta + i\tau,$$

a zeta oriented triadic closure tower *is a packet*

$$\mathfrak{T}_s := (X_\infty, \mathcal{R}_\infty, J, x_s, A_n, h_a, \Omega_n, \mathcal{B}_s)$$

with the following rows:

$$X_\infty = \varprojlim X_n, \qquad \mathcal{R}_\infty : X_\infty \to X_\infty,$$
$$J^2 = \mathrm{id}, \qquad J\mathcal{R}_\infty J = \mathcal{R}_\infty^{-1},$$

and a distinguished zero packet $x_s \in X_\infty$. Each finite level has active nodes A_n, positive heights $h_a > 0$, and normal-defect vector

$$\Omega_n(s) := (\log |e^{-(s-1/2)h_a}|)_{a \in A_n}.$$

The row $\mathcal{B}_s$ denotes the zero-compatible boundary or energy bound attached to the same packet.

Definition 9413.526 (Finite routing tower)**.** *A* finite routing tower *is a system*

$$X_1 \leftarrow X_2 \leftarrow X_3 \leftarrow \cdots$$

with projections

$$\pi_{n+1,n} : X_{n+1} \to X_n$$

and bijective routing maps

$$\mathcal{R}_n : X_n \to X_n$$

such that

$$\pi_{n+1,n} \circ \mathcal{R}_{n+1} = \mathcal{R}_n \circ \pi_{n+1,n}.$$

The inverse limit carries the induced homeomorphism

$$\mathcal{R}_\infty : X_\infty \to X_\infty.$$

Thus the routing row is not a hidden oracle; it is the inverse limit of finite auditable successor tables.

Definition 9413.527 (Euler-log solenoid bridge)**.** *For a finite or selected prime family S, an* Euler-log solenoid bridge *is a common phase carrier containing the Euler-log phase orbit*

$$\Lambda_S(\tau) := \overline{\{(e^{-it\log p})_{p\in S} : t \in \mathbb{R}\}} \subseteq \mathbb{T}^S$$

and the solenoid phase carrier

$$\Sigma_S = \widehat{\mathbb{Z}[S^{-1}]},$$

with a displayed comparison row between the visible unit-circle phase and the prime-register fibre

$$\prod_{p\in S}\mathbb{Z}_p.$$

The bridge is a source obligation: the multiplicative Euler-log phase does not automatically equal the additive localization phase of the solenoid carrier.

Definition 9413.528 (Construction rows for the oriented tower)**.** *An oriented triadic closure tower is constructed by six rows:*

$$\mathsf{ActiveCapture},\quad \mathsf{EulerLogSolenoidBridge},\quad \mathsf{TriadicClosure},$$

$$\mathsf{DefectExtraction},\quad \mathsf{CofinalGrowth},\quad \mathsf{BoundaryStability}.$$

They mean respectively: the packet captures the completed zero by Rouché or winding; the Euler-log phase and solenoid prime-register fibre sit in one phase container; prime charge, unit boundary, and time routing close modulo legal prime-adic gauge; the normal defect is extracted as $-\delta h_a$; the square weight

$$W_n := \sum_{a\in A_n} h_a^2$$

is cofinal; and the same zero-compatible packet carries a bound

$$|\Omega_n(s)|^2 \le B_s < \infty$$

or a stronger unit-boundary row.

Theorem 9413.529 (Defect extraction for oriented towers)**.** *For $s = 1/2+\delta+i\tau$ and every active node a,*

$$\log|e^{-(s-1/2)h_a}| = -\delta h_a.$$

Consequently

$$|\Omega_n(s)|^2 = \delta^2 \sum_{a\in A_n} h_a^2.$$

Proof. Since $s - 1/2 = \delta + i\tau$,

$$e^{-(s-1/2)h_a} = e^{-\delta h_a} e^{-i\tau h_a}.$$

Taking the logarithm of the modulus gives $-\delta h_a$. Squaring and summing over A_n gives the energy identity. □

Theorem 9413.530 (Oriented triadic closure towers are conditionally sufficient)**.** *If every critical-strip zeta zero carries a zeta oriented triadic closure tower whose construction rows hold, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

Let $\mathfrak{T}_s$ be the associated oriented tower. By Definition 9413.529,

$$|\Omega_n(s)|^2 = \delta^2 W_n.$$

If $\delta\#0$, then cofinal growth $W_n \to \infty$ gives

$$|\Omega_n(s)|^2 \to \infty.$$

This contradicts boundary stability

$$|\Omega_n(s)|^2 \le B_s < \infty.$$

Hence $\neg(\delta\#0)$. Located fixed-half collapse gives $\delta = 0$, so $\Re(s) = 1/2$. The fixed-half section follows from Definition 9413.199. □

Corollary 9413.531 (Parity closure gives the same fixed-half conclusion)**.** *If the oriented tower also satisfies*

$$Jx_s \in \overline{\{\mathcal{R}_\infty^m x_s : m \in \mathbb{Z}\}},$$

and its defect readout is routing-invariant and parity-odd, then $\delta = 0$.

Proof. This is exactly the abstract argument of Definition 9413.496 applied to the routing space X_∞ and to the defect readout $\Delta(x_s) = \delta$. □

Definition 9413.532 (Construction frontier)**.** *The* construction frontier *for the oriented triadic closure tower is the pair*

$$\mathsf{ActiveCapture} \quad \textit{and} \quad \mathsf{BoundaryStability}.$$

Active capture asks for a canonical dominant endpoint packet associated to each completed zero. Boundary stability asks for a Lee–Yang, unitary, solenoid-compactness, or equivalent bound for that same packet. The Euler-log solenoid bridge and routing closure are additional source obligations, but the analytic difficulty concentrates in these two rows.

Corollary 9413.533 (The construction target is not the zero statement)**.** *Constructing* $\mathfrak{T}_s$ *is stronger data than naming the fixed-half surface and weaker than an unconditional RH theorem.*

Proof. The tower includes source data: finite routing tables, phase containers, prime-register fibres, active endpoint capture, and a stability bound. None of these is implied by writing $\Re(s) = 1/2$. Conversely, the tower is introduced as a conditional certificate: until the construction rows are supplied for every completed zero, it remains a proof target rather than an unconditional theorem. □

9413.40 Harmonic Triadic Closure Map

The frequency, chord, solenoid, winding, Rouché, Lee–Yang, and dominance routes are not separate replacements for the RH-facing boundary problem. They occupy different rows of one harmonic triadic closure map. Prime logarithms supply the frequency ledger, phase evolution supplies resonance closure, winding supplies integer local degree, active capture ties a finite packet to an actual zero, and a unit-boundary row forbids an additional real positive scale.

Definition 9413.534 (Harmonic triadic closure packet)**.** *For a critical-strip coordinate*

$$s = \frac{1}{2} + \delta + i\tau,$$

a harmonic triadic closure packet *is a zero-relevant packet*

$$\mathfrak{H}_s := (A_n, h_a, U_a, \mathsf{F}, \mathsf{W}, \mathsf{C}, \mathsf{B})$$

with finite active windows A_n*, positive Hamiltonian heights* $h_a > 0$*, local channels*

$$U_a(s) := e^{-(s-1/2)h_a},$$

and four certificate rows:

$$\mathsf{F} = \mathsf{FrequencyUniqueness}, \qquad \mathsf{W} = \mathsf{IntegerWinding},$$

$$\mathsf{C} = \mathsf{ActiveCapture}, \qquad \mathsf{B} = \mathsf{UnitBoundaryStability}.$$

The rows mean respectively that the displayed frequencies have an auditable prime ledger, the local zero count is an integer winding or local degree, the packet captures the completed zero rather than a spurious finite silence, and the same packet carries a unitary, Lee–Yang, compact-solenoid, or equivalent normal-scale bound.

Definition 9413.535 (Harmonic off-boundary defect)**.** *For a finite active window A_n, the* harmonic off-boundary defect *is*

$$\Omega_{A_n}(s) := (\log |U_a(s)|)_{a \in A_n}.$$

The square weight of the same window is

$$W_{A_n} := \sum_{a \in A_n} h_a^2.$$

Theorem 9413.536 (Harmonic defect reads the normal coordinate)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

every harmonic active window satisfies

$$\Omega_{A_n}(s) = (-\delta h_a)_{a \in A_n},$$

and hence

$$|\Omega_{A_n}(s)|^2 = \delta^2 W_{A_n}.$$

Proof. Since

$$e^{-(s-1/2)h_a} = e^{-\delta h_a} e^{-i\tau h_a},$$

the logarithmic modulus of the local channel is $-\delta h_a$. Squaring and summing over A_n gives the displayed energy identity. □

Definition 9413.537 (Three integer rows)**.** *The harmonic packet separates three integer rows:*

$$\mathsf{PrimeExponentInteger}, \qquad \mathsf{LocalWindingInteger}, \qquad \mathsf{HalfPeriodInteger}.$$

The first is the unique prime-exponent ledger in an integer factorization. The second is the zero multiplicity or local mapping degree read by the argument principle. The third is the half-period index in a two-arm interference packet, where a local cancellation row has phase

$$\arg(A/B) = (2k-1)\pi.$$

These integer rows certify discrete closure data. They do not by themselves certify the unit boundary for the real positive factor $e^{-\delta h_a}$.

Lemma 9413.538 (Integer winding does not remove real scale)**.** *The existence of an integer winding row for a packet does not imply*

$$|U_a(s)| = 1$$

for its active channels.

Proof. The winding row is a statement about the local degree of the projected complex map. For a zero of order m, the argument of the projected function changes by $2\pi m$ around a small boundary loop. The local channel, however, still factors as

$$U_a(s) = e^{-\delta h_a} e^{-i\tau h_a}.$$

The integer m constrains the projected phase ledger. It does not force the real positive factor $e^{-\delta h_a}$ to equal 1. That constraint belongs to the unit-boundary row. □

Theorem 9413.539 (Prime harmonic levels are ledger-separated)**.** *For prime-adic parity nodes*

$$a = (p, j), \qquad b = (q, k),$$

with heights

$$h_a = 2^j \log p, \qquad h_b = 2^k \log q,$$

equality $h_a = h_b$ implies $a = b$.

Proof. This is exactly Definition 9413.737. The case $p = q$ reduces to $2^j = 2^k$. The case $p \neq q$ would give a nontrivial rational linear relation among $\log p$ and $\log q$, contradicting the unique-factorization consequence recorded in Definition 9413.677. □

Definition 9413.540 (Proof roles in the harmonic map)**.** *The harmonic closure map assigns the standard analytic and arithmetic tools to the following proof roles:*

Fourier and spectral decomposition $\rightsquigarrow$ frequency expansion,
unique factorization $\rightsquigarrow$ prime ledger separation,
Kronecker–Weyl and Dirichlet approximation $\rightsquigarrow$ finite phase recurrence,
pigeonhole $\rightsquigarrow$ finite projection collision,
argument principle $\rightsquigarrow$ integer winding,
Rouché $\rightsquigarrow$ active zero capture,
Lee–Yang, BKW, and dominance $\rightsquigarrow$ unit-boundary stability,
Pontryagin duality and solenoids $\rightsquigarrow$ compact phase and prime-register fibre,
Chebotarev and split-prime towers $\rightsquigarrow$ cofinal active growth.

No single row substitutes for the others. In particular, winding and Rouché certify capture, while Lee–Yang or dominance supplies the normal-scale obstruction.

Theorem 9413.541 (Harmonic triadic closure is conditionally sufficient)**.** *Suppose every completed critical-strip zero carries a harmonic triadic closure packet whose active windows satisfy cofinal growth*

$$W_{A_n} \to \infty$$

and whose unit-boundary stability row gives a finite normal-energy bound

$$|\Omega_{A_n}(s)|^2 \leq B_s$$

for all displayed windows. Then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The harmonic packet attached to z satisfies

$$|\Omega_{A_n}(s)|^2 = \delta^2 W_{A_n}$$

by Definition 9413.536. If $\delta \# 0$, cofinal growth gives

$$|\Omega_{A_n}(s)|^2 \to \infty,$$

contradicting the finite bound supplied by the unit-boundary stability row. Hence $\neg(\delta \# 0)$. Located fixed-half collapse gives $\delta = 0$, and the RH fixed-half section follows through Definition 9413.199. □

Corollary 9413.542 (Capture rows and boundary rows have distinct jobs)**.** *Argument-principle and Rouché rows can certify that a local packet counts the completed zero, but they do not by themselves exclude the factor $e^{-\delta h_a}$. Lee–Yang, BKW, compact solenoid, unitary, or dominance rows are the candidates for excluding that real positive scale.*

Proof. Definition 9413.538 separates integer winding from the modulus condition. Rouché stability preserves zero counts under a boundary inequality; it is an active-capture row. The contradiction in Definition 9413.541 uses the separate finite-energy or unit-boundary bound. Thus capture and boundary stability are independent proof roles. □

Corollary 9413.543 (Solenoid fibres do not carry the real scale row)**.** *A solenoid row can carry visible phase and hidden prime-register data, but a nontrivial real positive scale*

$$e^{-\delta h_a}$$

is not a solenoid fibre coordinate.

Proof. The solenoid row is compact and phase-like. By Definition 9413.781, a compact phase carrier has no nontrivial continuous character into $\mathsf{Real}^{\uparrow}_{>0}$. Therefore the factor $e^{-\delta h_a}$ must be killed by the boundary row, not stored as hidden solenoid data. □

9413.41 Zero-Knowledge Triadic Certificate Layer

The zero-knowledge proof idea enters this route only as a certificate wrapper. It is not an RH proof and it is not a substitute for the zeta-specific source theorems. Its role is to separate public verification from full disclosure of the prime-routing table, zero ordering, solenoid fibre data, and resonance selector choices. The public surface checks finite consistency, active capture, defect extraction, cofinal growth, and unit-boundary stability; the hidden witness contains the detailed tower data.

Definition 9413.544 (Triadic witness for a completed zero)**.** *For a critical-strip coordinate*

$$s = \frac{1}{2} + \delta + i\tau,$$

a triadic witness *for s is a packet*

$$\mathcal{W}_s := (D_n, F_n, T_n, X_n, \mathcal{R}_n, \pi_{n+1,n}, S_n, \Sigma_{S_n}, A_n, h_a, \Omega_n, W_n, B_s)_n$$

equipped with the following rows:

$$\mathsf{ActiveCapture}, \qquad \mathsf{RoutingCompatibility}, \qquad \mathsf{SolenoidCompatibility},$$

$$\mathsf{TriadicClosure}, \qquad \mathsf{DefectExtraction}, \qquad \mathsf{CofinalGrowth}, \qquad \mathsf{BoundaryStability}.$$

The packet is allowed to contain detailed finite routing tables and selector data. The public verifier consumes only the displayed rows and does not require the full inverse-limit routing table as public data.

Definition 9413.545 (Public triadic verifier)**.** *The* public triadic verifier *is the relation*

$$\mathsf{VerifyTriadic}(s, \mathcal{W}_s) = 1$$

which checks the following finite-layer obligations:

$$\mathsf{RoucheCapture}(D_n, F_n, T_n), \qquad \mathsf{Permutation}(\mathcal{R}_n),$$

$$\pi_{n+1,n} \circ \mathcal{R}_{n+1} = \mathcal{R}_n \circ \pi_{n+1,n},$$

$$\mathsf{SolenoidPhaseCarrier}(\Sigma_{S_n}), \qquad \mathsf{TriadicTwoOutOfThree},$$

$$\Omega_n(s) = (-\delta h_a)_{a \in A_n}, \qquad W_n = \sum_{a \in A_n} h_a^2,$$

and the paired growth and bound rows

$$W_n \to \infty, \qquad |\Omega_n(s)|^2 \le B_s < \infty.$$

Here RoucheCapture *may be replaced by any equivalent local-degree or winding capture row that proves the finite packet counts the completed zero.*

Definition 9413.546 (Completeness, soundness, and opacity readings)**.** *The certificate layer has three proof-system readings.* Completeness *means that a genuine zero-relevant source packet can be presented as a witness accepted by* VerifyTriadic*.* Soundness *means that acceptance implies the fixed-half row.* Opacity *means that the verifier need not receive the full routing table, the full zero ordering, or the full selector ledger as public data. No cryptographic commitment scheme is asserted here; the point is the mathematical separation between hidden witness data and public finite checks.*

Theorem 9413.547 (Public triadic verification is sound for the fixed-half row)**.** *If*

$$\mathsf{VerifyTriadic}(s, \mathcal{W}_s) = 1$$

for

$$s = \frac{1}{2} + \delta + i\tau,$$

then

$$\delta = 0.$$

Proof. The verifier accepts only when the defect extraction row gives

$$\Omega_n(s) = (-\delta h_a)_{a \in A_n}$$

and hence

$$|\Omega_n(s)|^2 = \delta^2 W_n.$$

It also accepts only when $W_n \to \infty$ and

$$|\Omega_n(s)|^2 \leq B_s < \infty$$

for the same displayed windows. If $\delta \# 0$, then $\delta^2 W_n \to \infty$, contradicting the finite bound. Hence $\neg(\delta \# 0)$. Located fixed-half collapse gives $\delta = 0$. □

Corollary 9413.548 (Verified triadic witnesses are conditionally sufficient)**.** *If every completed critical-strip zero has a triadic witness accepted by the public triadic verifier, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate s. By hypothesis, there is a witness $\mathcal{W}_s$ with

$$\mathsf{VerifyTriadic}(s, \mathcal{W}_s) = 1.$$

By Definition 9413.547, the coordinate has $\delta = 0$, hence $\Re(s) = 1/2$. The fixed-half section is therefore obtained through Definition 9413.199. □

Definition 9413.549 (Hidden prime-collision row)**.** *For a finite prime ledger $L_n \subset \mathbb{Z}^{(S_n)}$, a* hidden prime-collision row *consists of hidden rows*

$$x_n, y_n \in L_n, \qquad x_n \neq y_n, \qquad \pi_n(x_n) = \pi_n(y_n),$$

together with the frequency functional

$$H(x) := \sum_{p \in S_n} x_p \log p.$$

The row is valid only when the gap

$$g_n := x_n - y_n$$

satisfies

$$H(g_n) \neq 0.$$

The verifier may consume the equality and non-equality obligations as committed finite rows without exposing the full ledger pair as public data.

Lemma 9413.550 (Prime collisions expose a nonzero frequency gap)**.** *In a hidden prime-collision row, the condition $x_n \neq y_n$ forces*

$$H(x_n - y_n) \neq 0.$$

Proof. If $H(x_n - y_n) = 0$, then

$$\sum_{p \in S_n} (x_{n,p} - y_{n,p}) \log p = 0.$$

This is a rational, indeed integral, linear relation among the displayed prime logarithms. By Definition 9413.677, all coefficients vanish, so $x_n = y_n$, contradicting the collision apartness row. □

Theorem 9413.551 (Hidden collision plus unit boundary kills real scale)**.** *Let a hidden prime-collision row provide a gap g_n with $H(g_n) \neq 0$. For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the channel ratio

$$R_{g_n}(s) := e^{-(s-1/2)H(g_n)}$$

satisfies

$$|R_{g_n}(s)| = e^{-\delta H(g_n)}.$$

If the unit-boundary row gives $|R_{g_n}(s)| = 1$, then $\delta = 0$.

Proof. The displayed factorization gives

$$R_{g_n}(s) = e^{-\delta H(g_n)} e^{-i\tau H(g_n)}.$$

Taking moduli yields the first formula. If the unit-boundary row gives $|R_{g_n}(s)| = 1$, then

$$e^{-\delta H(g_n)} = 1.$$

Taking logarithms in the positive real readout gives

$$-\delta H(g_n) = 0.$$

Since $H(g_n) \neq 0$, it follows that $\delta = 0$. □

Corollary 9413.552 (The zero-knowledge layer does not create the witness)**.** *The certificate layer packages the route; it does not discharge* ActiveCapture *or* BoundaryStability*.*

Proof. Definition 9413.547 uses the rows accepted by the verifier. It does not construct the Rouché island, the dominant resonance family, the solenoid-compatible phase carrier, or the Lee–Yang-type bound. Those remain source rows in the sense of Definition 9413.801. Thus the zero-knowledge layer is a certificate wrapper around source data, not a source theorem. □

9413.42 Analytic Projection and Closed Generability Boundary

The analytic continuation row must be separated from the closed generation row. Analytic continuation recovers a projected analytic object; it does not, by itself, certify that every analytic feature of that projection is generated by the closed triadic dynamics used in the RH-facing route. The distinction is between projected analytic existence and closed-system realizability.

Definition 9413.553 (Analytic projection carrier)**.** *The* analytic projection carrier *for zeta is the carrier*

$$\mathcal{A}_\zeta$$

whose rows are the analytic readouts produced by the Dirichlet-series domain, Euler-product readout on its convergence side, Mellin or contour representations, analytic continuation, pole ledger, functional equation, and completed value projection. A row in $\mathcal{A}_\zeta$ is therefore a projection-side analytic datum. It is licensed as an analytic object, not automatically as a closed dynamical state.

Definition 9413.554 (Closed generability carrier)**.** *The* closed generability carrier *is the subcarrier*

$$\mathcal{G}_\zeta \subseteq \mathcal{A}_\zeta$$

consisting of analytic projection rows which also carry a closed triadic generation certificate. Such a certificate displays:

$$\mathsf{ActiveCapture}, \qquad \mathsf{CompactPhaseOrSolenoidFibre}, \qquad \mathsf{RoutingCompatibility},$$

$$\mathsf{TriadicClosure}, \qquad \mathsf{DefectExtraction}, \qquad \mathsf{BoundaryStability}.$$

Thus $\mathcal{G}_\zeta$ records the analytic rows which are not merely visible in the completed projection, but are generated through finite packets, compatible routing, legal hidden fibres, and unit-boundary stability.

Principle 9413.555 (Analytic continuation is not generability)**.** *Membership in $\mathcal{A}_\zeta$ does not imply membership in $\mathcal{G}_\zeta$. Analytic continuation supplies the projected analytic object and its uniqueness in the analytic category. Closed generability requires additional source rows: zero-relevant packet capture, compatible finite routing, compact phase or solenoid fibre placement, and a unit-boundary or energy-stability certificate.*

Definition 9413.556 (Non-generable analytic residue)**.** *A row $x \in \mathcal{A}_\zeta$ has a* non-generable analytic residue *when it requires at least one of the following data without a closed certificate:*

a nontrivial real positive scale $e^{-\delta h}$,

a hidden coordinate outside compact phase or prime-register fibre,

a projected zero with no active Rouché or local-degree capture,

a finite routing table with no inverse-limit compatibility,

or

a zero-relevant packet with no unit-boundary stability row.

Such a row may still belong to the analytic projection carrier. The point is only that it has not been shown to belong to the closed generability carrier.

Theorem 9413.557 (Real positive scale is not a closed compact residue)**.** *Let a closed generability certificate for a local channel use only unitary phase and compact solenoid prime-register fibres for its hidden coordinates. For*

$$s = \frac{1}{2} + \delta + i\tau, \qquad U_a(s) = e^{-(s-1/2)h_a}, \qquad h_a > 0,$$

the factor

$$e^{-\delta h_a}$$

can be supplied by that compact hidden carrier only when $\delta = 0$.

Proof. The local channel factors as

$$U_a(s) = e^{-\delta h_a} e^{-i\tau h_a}.$$

The phase factor belongs to the unit-circle readout. A compact solenoid fibre can record compact prime-register data, but by Definition 9413.781 it has no nontrivial continuous character into $\mathsf{Real}^{\uparrow}_{>0}$. Hence the real positive factor must be trivial:

$$e^{-\delta h_a} = 1.$$

Since $h_a > 0$, this gives $\delta = 0$. □

Definition 9413.558 (Zeta zero generability row)**.** *The* zeta zero generability row *is the source assertion*

$$\mathsf{ZetaZeroGenerability}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to s \in \mathcal{G}_\zeta)\,.$$

It says that every critical-strip zero of the completed zeta projection has a closed triadic generation certificate. It is not supplied by analytic continuation alone.

Definition 9413.559 (Closed triadic no-real-scale row)**.** *The* closed triadic no-real-scale row *is the assertion*

$$\prod_{s:\mathsf{Complex}^\uparrow} \left(s \in \mathcal{G}_\zeta \to \Re(s) = \frac{1}{2}\right),$$

with the implication witnessed by defect extraction, cofinal growth, and unit-boundary or compact-solenoid stability for the same zero-relevant packet.

Theorem 9413.560 (Generability and no-real-scale rows are conditionally sufficient)**.** *If*

$$\mathsf{ZetaZeroGenerability}_\zeta$$

and the closed triadic no-real-scale row both hold, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate s. The zero generability row gives

$$s \in \mathcal{G}_\zeta.$$

The no-real-scale row then gives

$$\Re(s) = \frac{1}{2}.$$

Varying z gives the fixed-half section, equivalently the RH section identified by Definition 9413.199. □

Corollary 9413.561 (The hard source row is generability)**.** *For the analytic-projection route, the hard source row is not the exclusion of real scale inside an already closed packet. It is the construction of a closed packet for each completed zero:*

$$\mathsf{ZetaZeroGenerability}_\zeta.$$

Proof. Once a zero row is in $\mathcal{G}_\zeta$, the compact hidden-fibre argument of Definition 9413.557 or the energy-bound argument of Definition 9413.547 excludes an off-half real scale. What those arguments do not supply is membership in $\mathcal{G}_\zeta$. That membership requires active capture, dominance or tail control, compatible routing, solenoid placement, and boundary stability. This is the same source load isolated in Definition 9413.803. □

Corollary 9413.562 (Projection existence and generability remain separate)**.** *The existence of the analytic continuation of zeta, including its completed functional-equation projection, does not discharge the generability row.*

Proof. The analytic continuation theorem places the completed zeta readout in $\mathcal{A}_\zeta$. By Definition 9413.555, membership in $\mathcal{A}_\zeta$ does not provide the finite packet, routing, solenoid, capture, and boundary rows required for $\mathcal{G}_\zeta$. Therefore the generability row is an additional source theorem, not a consequence of analytic continuation as such. □

9413.43 Oriented Triadic Certificate System

The oriented tower construction can be read as a certificate system. It does not assert the unconditional RH statement. It records the precise conditional form: a completed critical-strip zero placed inside a closed oriented triadic tower has no admissible real positive scale direction. The mathematical burden is therefore the construction of the certificate for every completed zero, not the downstream energy contradiction.

Definition 9413.563 (Ontic triadic rows)**.** *The* ontic triadic rows *are*

$$\Omega_{\mathrm{tri}} := (\mathsf{P}, \mathsf{U}, \mathsf{T}),$$

where P *is the prime-register or discrete-label row,* U *is the unit-boundary, parity, compactness, or Lee–Yang row, and* T *is the phase-time or recursive-routing row. A closed triadic packet carries a closure relation*

$$\mathcal{C}(\mathsf{P}, \mathsf{U}, \mathsf{T}) = 0$$

up to legal gauge. Legal gauge may come from the unit phase circle or from prime-adic register fibres. A nontrivial real positive scale is not a legal gauge row.

Definition 9413.564 (Prime ledger and logarithmic height)**.** *The* prime ledger *is the free finite-support integer register*

$$\mathcal{L}_{\mathsf{Prime}^{\uparrow}} := \bigoplus_{p \in \mathsf{Prime}^{\uparrow}} \mathbb{Z} e_p.$$

For a positive integer

$$n = \prod_p p^{v_p(n)},$$

the ledger row is

$$\nu(n) := \sum_p v_p(n) e_p,$$

and the logarithmic height row is

$$H(n) := \log n = \sum_p v_p(n) \log p.$$

More generally a finite ledger vector

$$x = \sum_p x_p e_p$$

has height

$$H(x) := \sum_p x_p \log p.$$

Lemma 9413.565 (Prime ledger height is separated)**.** *For a finite ledger vector* $x \in \mathcal{L}_{\mathsf{Prime}^{\uparrow}}$*,*

$$H(x) = 0 \quad \Longrightarrow \quad x = 0.$$

Proof. Write

$$x = \sum_{i=1}^{m} x_{p_i} e_{p_i}.$$

If $H(x) = 0$, then

$$\sum_{i=1}^{m} x_{p_i} \log p_i = 0.$$

Exponentiating gives

$$\prod_{i=1}^{m} p_i^{x_{p_i}} = 1.$$

By unique factorization, every exponent x_{p_i} is zero. Hence $x = 0$. □

Definition 9413.566 (Ontic logarithmic Hamiltonian)**.** *The* ontic logarithmic Hamiltonian *on the integer basis is the diagonal readout*

$$H_{\mathrm{ont}}|n\rangle = (\log n)|n\rangle$$

with domain

$$\mathrm{Dom}(H_{\mathrm{ont}}) := \left\{ \psi = (\psi_n) \in \ell^2(\mathsf{Nat}^{\uparrow}_{\geq 1}) : \sum_{n\geq 1} (\log n)^2 |\psi_n|^2 < \infty \right\}.$$

Its unitary phase evolution row is

$$U(\gamma) = e^{-i\gamma H_{\mathrm{ont}}}.$$

This Hamiltonian row is a spectral readout for the logarithmic height, not a primitive physical postulate.

Definition 9413.567 (Oriented triadic certificate)**.** *For a critical-strip coordinate*

$$s_0 = \frac{1}{2} + \delta + i\tau,$$

an oriented triadic certificate *over s_0 is a finite-window tower*

$$\mathcal{W}_{s_0} := (F_n, T_n, D_n, A_n, S_n, \Sigma_{S_n}, X_n, \mathcal{R}_n, \pi_{n+1,n}, \Omega_n, W_n, B_{s_0})_n$$

whose rows are:

$$\mathsf{ActiveCapture}, \quad \mathsf{EulerLogSolenoidBridge}, \quad \mathsf{FiniteTriadicClosure},$$

$$\mathsf{DefectExtraction}, \quad \mathsf{CofinalGrowth}, \quad \mathsf{IndependentBoundaryStability}.$$

The certificate is the fully displayed form of Definition 9413.528. It keeps source data, compatibility data, and boundary data in the same packet.

Definition 9413.568 (Certificate row meanings)**.** *The six certificate rows mean the following. The active-capture row supplies a zero island D_n with*

$$\Xi = F_n + T_n, \qquad |T_n| < |F_n| \quad \text{on } \partial D_n,$$

or an equivalent local-degree capture row. The Euler-log solenoid bridge places the visible Euler-log phase and the prime-register fibre in one phase carrier. The finite triadic closure row closes $(\mathsf{P}, \mathsf{U}, \mathsf{T})$ *modulo legal prime-adic gauge. The defect-extraction row gives*

$$\Omega_n(s_0) = (-\delta h_a)_{a\in A_n}$$

for positive heights $h_a > 0$. The cofinal-growth row gives

$$W_n := \sum_{a \in A_n} h_a^2 \to \infty.$$

The independent-boundary-stability row gives a finite constant

$$B_{s_0} < \infty$$

with

$$|\Omega_n(s_0)|^2 \le B_{s_0}$$

for the same zero-relevant tower.

Proposition 9413.569 (Defect extraction gives square energy)**.** *For an oriented triadic certificate and*

$$s_0 = \frac{1}{2} + \delta + i\tau,$$

the defect row gives

$$|\Omega_n(s_0)|^2 = \delta^2 W_n.$$

Proof. For every active node $a \in A_n$, the local channel is

$$U_a(s_0) = e^{-(s_0 - 1/2)h_a} = e^{-\delta h_a} e^{-i\tau h_a}.$$

Thus

$$\log |U_a(s_0)| = -\delta h_a.$$

By definition,

$$|\Omega_n(s_0)|^2 = \sum_{a \in A_n} (-\delta h_a)^2 = \delta^2 \sum_{a \in A_n} h_a^2 = \delta^2 W_n.$$

□

Proposition 9413.570 (Parity oddness of the normal defect)**.** *For the mirror involution*

$$J(s) = 1 - \overline{s},$$

if $s = 1/2 + \delta + i\tau$, then $J(s) = 1/2 - \delta + i\tau$ and every active node satisfies

$$d_a(Js) = -d_a(s).$$

Proof. The mirror computation gives

$$J\left(\frac{1}{2} + \delta + i\tau\right) = 1 - \left(\frac{1}{2} + \delta - i\tau\right) = \frac{1}{2} - \delta + i\tau.$$

Since $d_a(s) = -\delta h_a$, replacing δ by $-\delta$ gives $d_a(Js) = \delta h_a = -d_a(s)$. □

Proposition 9413.571 (Active capture preserves the zero count)**.** *If the active-capture row holds on a zero island D_n, then F_n and Ξ have the same number of zeros in D_n, counted with multiplicity.*

Proof. The capture row gives

$$\Xi = F_n + T_n$$

and

$$|T_n| < |F_n| \quad \text{on } \partial D_n.$$

By Rouché's theorem, F_n and $F_n + T_n = \Xi$ have the same zero count inside D_n, counted with multiplicity. □

Proposition 9413.572 (A split-prime tower is a cofinal-growth template)**.** *Let q be a rational prime and let (K_n) be a family of number fields with*

$$[K_n : \mathbb{Q}] \to \infty.$$

If q splits completely in K_n as

$$q\mathcal{O}_{K_n} = \mathfrak{q}_{n,1} \cdots \mathfrak{q}_{n,d_n}, \qquad d_n = [K_n : \mathbb{Q}],$$

and each split-prime node has height $\log q$, then the associated square weight satisfies

$$W_n = d_n(\log q)^2 \to \infty.$$

Proof. There are d_n displayed split-prime nodes, each with height $\log q$. Therefore

$$W_n = \sum_{i=1}^{d_n} (\log q)^2 = d_n(\log q)^2.$$

Since $d_n \to \infty$ and $\log q > 0$, the weights are cofinal. □

Theorem 9413.573 (An oriented triadic certificate forces the fixed-half row)**.** *If a critical-strip zero coordinate*

$$s_0 = \frac{1}{2} + \delta + i\tau$$

has an oriented triadic certificate, then

$$\delta = 0.$$

Proof. By Definition 9413.569, the certificate gives

$$|\Omega_n(s_0)|^2 = \delta^2 W_n.$$

Assume $\delta \# 0$. Then $\delta^2 > 0$, and the cofinal-growth row $W_n \to \infty$ gives

$$|\Omega_n(s_0)|^2 \to \infty.$$

This contradicts independent boundary stability,

$$|\Omega_n(s_0)|^2 \leq B_{s_0} < \infty.$$

Hence $\neg(\delta \# 0)$. Located fixed-half collapse gives $\delta = 0$. □

Corollary 9413.574 (Certificate existence is conditionally sufficient)**.** *If every completed critical-strip zeta zero has an oriented triadic certificate, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

By hypothesis, s has an oriented triadic certificate. By Definition 9413.573, we have $\delta = 0$, so $\Re(s) = 1/2$. Varying z gives the fixed-half section, equivalently the RH section of Definition 9413.199. □

Corollary 9413.575 (Off-half zeros are not closed-triadic generated)**.** *If*

$$s = \frac{1}{2} + \delta + i\tau$$

is a critical-strip zero coordinate with $\delta\#0$, then s cannot carry an oriented triadic certificate satisfying cofinal growth and independent boundary stability.

Proof. If such a certificate existed, then Definition 9413.573 would give $\delta = 0$, contradicting $\delta\#0$. □

Definition 9413.576 (Source obligations for the certificate route)**.** *The remaining source obligations for the oriented certificate route are:*

CanonicalActiveCapture, EulerLogSolenoidBridge, FiniteTriadicClosure,

DefectExtraction, ActiveCofinalGrowth, IndependentBoundaryStability.

The defect-extraction row is the direct calculation of Definition 9413.569. The other rows are source theorems: they must be proved for the same zero-relevant tower and may not be inferred merely from the zero predicate.

Theorem 9413.577 (The source obligations are the noncircular content)**.** *The conditional implication from an oriented triadic certificate to the fixed-half row does not discharge the source obligations of Definition 9413.576.*

Proof. The implication uses the certificate rows after they have been supplied. It consumes active capture to ensure zero relevance, cofinal growth to make an off-half defect unbounded, and independent boundary stability to bound the same defect. It does not construct the active endpoint packet, the Euler-log solenoid bridge, the finite triadic closure law, the active growth tower, or the Lee–Yang, unitary, cosine, or dominance stability input. Therefore the proof is noncircular exactly when these rows are proved independently as source data, as also isolated in Definition 9413.561. □

9413.44 Triadic Route Taxonomy

The preceding certificate systems can be compressed into three conditional routes. Each route is a sufficient row for the RH fixed-half section, and none is an unconditional proof until its source predicate has been constructed from the completed zeta zero ledger. The taxonomy prevents the route from oscillating between equivalent surface names: ghost exclusion, stable packet capture, and triadic closure are distinct ways of naming the missing source theorem.

Definition 9413.578 (Closed-generatable zero)**.** *A critical-strip zero coordinate s is* closed-generatable *when it belongs to the closed generability carrier*

$$s \in \mathcal{G}_\zeta$$

of Definition 9413.554. Equivalently, it has a zero-relevant packet with active capture, legal solenoid or prime-register hidden fibre, finite routing compatibility, unit-boundary stability, and no noncompact real positive gauge. Write this row as

$$\mathsf{ClosedGeneratable}_\zeta(s).$$

Definition 9413.579 (No ghost zeta zero row)**.** *The* no ghost zeta zero row *is*

$$\mathsf{NoGhostZetaZero}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{ClosedGeneratable}_\zeta(s)\right).$$

A ghost zero would be a completed analytic zero row in $\mathcal{A}_\zeta$ which has no closed triadic generation certificate in $\mathcal{G}_\zeta$.

Corollary 9413.580 (No ghost zeros are conditionally sufficient)**.** *If*

$$\mathsf{NoGhostZetaZero}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate s. The no-ghost row gives $\mathsf{ClosedGeneratable}_\zeta(s)$, hence $s \in \mathcal{G}_\zeta$. By Definition 9413.559 and the compact-scale obstruction of Definition 9413.557, closed generability supplies the fixed-half row. Varying z gives the RH section. □

Definition 9413.581 (Stable packet capture row)**.** *The* stable packet capture row *is the assertion that every critical-strip zero coordinate s has a canonical packet P_s with:*

$$\mathsf{RoucheCapture}(P_s, s), \qquad \mathsf{LeeYangStable}(P_s),$$

$$\mathsf{SolenoidCompatible}(P_s, s), \qquad \mathsf{NondegenerateActiveHeight}(P_s).$$

Write this source predicate as

$$\mathsf{ZetaStablePacketCapture}_\zeta.$$

It is stronger than merely displaying a finite model: the packet must be zero-relevant for the completed zeta row.

Theorem 9413.582 (Stable packet capture is conditionally sufficient)**.** *If*

$$\mathsf{ZetaStablePacketCapture}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The stable packet capture row supplies a packet P_s which captures the completed zero and is Lee–Yang stable. The local channel has the form

$$u_a(s) = e^{-(s-1/2)h_a} = e^{-\delta h_a} e^{-i\tau h_a}$$

with some active height $h_a > 0$. If $\delta > 0$, then all nearby active channels lie inside the unit boundary; if $\delta < 0$, they lie outside it. Lee–Yang stability excludes packet zeros in those regions, while Rouché capture forces the packet to have the same local zero count as the completed zeta readout. Thus $\delta \# 0$ is impossible. Located fixed-half collapse gives $\delta = 0$, and the RH section follows from Definition 9413.199. □

Definition 9413.583 (Zeta triadic closure tower row)**.** *The* zeta triadic closure tower row *is the assertion that every critical-strip zero coordinate* $s = 1/2 + \delta + i\tau$ *has a packet*

$$x_s \in X_\infty$$

inside an inverse-limit routing carrier such that:

$$\mathsf{ActiveCapture}(x_s, s), \qquad \Delta(x_s) = \delta,$$
$$Px_s \in \overline{\{\mathcal{R}_\infty^m x_s : m \in \mathbb{Z}\}},$$

and the same packet carries solenoid compatibility, finite triadic closure, and either the parity-closure defect row or the cofinal energy-bound row of the oriented certificate system. Write this source predicate as

$$\mathsf{ZetaTriadicClosureTower}_\zeta.$$

Theorem 9413.584 (Triadic closure tower is conditionally sufficient)**.** *If*

$$\mathsf{ZetaTriadicClosureTower}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate $s = 1/2 + \delta + i\tau$. The triadic closure tower row supplies a packet x_s with

$$\Delta(x_s) = \delta$$

and parity closure

$$Px_s \in \overline{\{\mathcal{R}_\infty^m x_s : m \in \mathbb{Z}\}}.$$

By Definition 9413.496,

$$\Delta(x_s) = 0.$$

Thus $\delta = 0$. The fixed-half section follows through Definition 9413.199. □

Corollary 9413.585 (The three source rows form one proof target)**.** *The following rows are conditionally sufficient for the same fixed-half section:*

$$\mathsf{NoGhostZetaZero}_\zeta, \qquad \mathsf{ZetaStablePacketCapture}_\zeta, \qquad \mathsf{ZetaTriadicClosureTower}_\zeta.$$

They differ in presentation, not in the downstream obstruction: each must construct zero-relevant source data that excludes noncompact real scale.

Proof. The first row places every completed zero in the closed generability carrier. The second supplies the strongest local version: canonical Rouché capture plus Lee–Yang stability and solenoid compatibility. The third supplies the routing version: a parity-closed inverse-limit packet or an energy-bounded oriented certificate. The conditional implications are the preceding two theorems and Definition 9413.580. In each case, the unproved content is source construction, not the fixed-half algebra performed after the source data are available. □

Corollary 9413.586 (Stable packet capture is the sharpest local target)**.** *Among the three rows,* $\mathsf{ZetaStablePacketCapture}_\zeta$ *is the sharpest local analytic target.*

Proof. No-ghost exclusion is a global generability statement. The triadic closure tower is a routing and parity statement over an inverse-limit carrier. Stable packet capture names the local analytic object whose construction would feed both: a canonical zero island, Rouché capture, Lee–Yang stability, solenoid compatibility, and nondegenerate active height. Thus it is the local source theorem at which the route should concentrate before the global closure language is invoked. □

9413.45 BEDC RHRoute Formal Bridge

The preceding taxonomy identifies the analytic source theorem that would feed the RH fixed-half section. The Lean-side RHRoute layer should be read in the same way: it supplies a finite formal route discipline and several checked algebraic carriers, but it does not by itself construct the zeta-specific source family for every completed critical-strip zero. The bridge point is therefore not the fixed-half conclusion itself, but the map from completed zero rows to zero-relevant finite certificates.

Definition 9413.587 (RHRoute formal layer)**.** *The* RHRoute formal layer *is the collection of finite BEDC carriers that record:*

$$\mathsf{ConstructiveZetaEval}, \quad \mathsf{MirrorFixedAlgebra}, \quad \mathsf{FinitePrimeWindow},$$

$$\mathsf{PrimeSkewDefect}, \quad \mathsf{LocalGlobalExclusion}, \quad \mathsf{RouteClosureSeparation}.$$

The first row gives rational-complex evaluation packets for the eta-to-zeta readout. The mirror row proves the finite algebraic equivalence between J-fixedness and the critical-line readout. The prime-window rows record finite prime supports, unitary-balance packets, and skew-defect certificates. The route-separation row records that a finite route closure certificate is not uniformly the same thing as an arbitrary fixed statement.

Definition 9413.588 (Formal bridge packet)**.** *For a critical-strip zero coordinate s, an* RHRoute formal bridge packet *consists of:*

$$\mathsf{CritZero}(s), \quad \mathsf{EvalPacket}(s), \quad \mathsf{MirrorOrbit}(s),$$

$$\mathsf{PrimeWindowSource}(s), \quad \mathsf{SkewOrUnitaryBoundary}(s), \quad \mathsf{RouteSeparationAudit}(s).$$

The packet is a formal carrier for the downstream proof route. It is not yet a source theorem unless the prime-window source is produced from the completed zero ledger by active capture, dominance, tail control, and solenoid or Lee–Yang stability.

Definition 9413.589 (RHRoute source bridge)**.** *The* RHRoute source bridge *is the zeta-specific row*

$$\mathsf{RHRouteSourceBridge}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{CritZero}(s) \to \sum_{P:\mathsf{RHRouteFormalBridgePacket}(s)} \mathsf{ZeroRelevant}(P, s) \right).$$

Here $\mathsf{ZeroRelevant}(P, s)$ *means that the finite prime-window or resonance data in P controls a local zero island of the completed zeta readout, rather than merely displaying a visible defect or a compatible finite route.*

Theorem 9413.590 (The formal layer is not the source bridge)**.** *The RHRoute formal layer does not by itself imply*

$$\mathsf{RHRouteSourceBridge}_\zeta.$$

Proof. The formal layer contains finite algebraic and route-theoretic carriers: evaluation packets, mirror-fixed readbacks, prime-window supports, unitary-balance surfaces, skew-defect certificates, and finite route closure decisions. Each of these is a legal finite BEDC object. However, none of them asserts that every completed critical-strip zero has a zero-relevant dominant source packet. In particular, route closure and fixed-statement transport are separated by the route separation row: a closed finite route is not a uniform proof of an arbitrary fixed statement, and a fixed statement is not uniformly a finite route closure certificate. Therefore the missing row is still the source bridge from the zero ledger to zero-relevant finite data. □

Definition 9413.591 (BEDC RHRoute certificate)**.** *A* BEDC RHRoute certificate *for s is an RHRoute formal bridge packet P together with the following additional rows:*

$$\mathsf{ActiveCapture}(P,s), \qquad \mathsf{DominantSource}(P,s), \qquad \mathsf{BoundaryStability}(P,s),$$

$$\mathsf{DefectExtraction}(P,s), \qquad \mathsf{FixedHalfReadback}(P,s).$$

Active capture and dominance make P zero-relevant for the completed zeta readout. Boundary stability prevents the packet from carrying a noncompact real scale. Defect extraction identifies the normal coordinate read by the prime-window or resonance channel. Fixed-half readback transports the resulting J-fixed row to $\mathsf{OnCritLine}(s)$.

Theorem 9413.592 (BEDC RHRoute certificates are sufficient)**.** *If every completed critical-strip zero carries a BEDC RHRoute certificate, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate s. By hypothesis, s carries a BEDC RHRoute certificate P. The active-capture and dominance rows say that P is zero-relevant for the completed zeta readout. The defect-extraction row reads the normal coordinate

$$\delta(s) = \Re(s) - \frac{1}{2}$$

through the finite prime-window or resonance channel. The boundary stability row excludes a nonzero value of that normal coordinate on a zero-relevant packet; otherwise the same packet would carry a visible prime skew or an unbounded normal energy incompatible with the boundary row. Hence $\delta(s) = 0$. The mirror-fixed algebra then reads this as J-fixedness, and the fixed-half readback gives

$$\mathsf{OnCritLine}(s).$$

Varying z gives the fixed-half section. □

Corollary 9413.593 (The remaining theorem is source construction)**.** *The unclosed content of the RHRoute route is the implication*

$$\mathsf{CritZero}(s) \to \sum_{P:\mathsf{RHRouteFormalBridgePacket}(s)} \left(\mathsf{ActiveCapture}(P,s) \times \mathsf{DominantSource}(P,s) \times \mathsf{BoundaryStability}(P,s)\right).$$

Proof. All rows after source construction are structural. Mirror-fixed readback is the algebraic equivalence between J-fixedness and the critical-line coordinate. Prime-window skew and normal-energy rows detect nonzero off-half displacement once a zero-relevant packet has been supplied. Route separation prevents these finite certificates from being confused with the target statement itself. Thus the proof load is the zeta-specific construction of P: active capture, dominance, tail control, and boundary stability for the completed zero ledger. □

Corollary 9413.594 (Compatibility with the triadic route taxonomy)**.** *The RHRoute formal bridge is a finite certificate interface for the same proof target named by the no-ghost, stable-packet, and triadic-closure rows of section 9413.44.*

Proof. The no-ghost row asks that every completed zero enter the closed generability carrier. The stable-packet row asks for a canonical local zero island with Rouché capture and Lee–Yang stability. The triadic-closure row asks for an inverse-limit packet whose parity or energy certificate kills the normal coordinate. An RHRoute certificate is the finite formal interface through which these rows can be audited: it displays the evaluation packet, the mirror algebra, the finite prime-window source, the skew or unitary boundary, and the route separation audit. It therefore refines the taxonomy without replacing the remaining source theorem. □

9413.46 Generation Invariant Exclusion Route

The RHRoute formal bridge may also be read as an invariant argument in the style of a formal word system. The target is not to guess a direct analytic implication from a zero row to the critical line. The target is to display a generation system, identify an invariant subcarrier, prove that every legal rule preserves that subcarrier, and then prove that every completed zero enters the generated carrier through an independent source theorem.

Definition 9413.595 (Invariant generation system)**.** *An* invariant generation system *for a zero carrier is a tuple*

$$\mathfrak{G} := (\Sigma, \mathcal{W}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mathcal{I})$$

where Σ is a signature, $\mathcal{W}$ is the ambient carrier of legal packets, $\mathcal{A} \subseteq \mathcal{W}$ is the source family, $\mathcal{R}$ is the family of generation rules, $\mathcal{T}$ is the closed carrier generated from $\mathcal{A}$ by $\mathcal{R}$, and

$$\mathcal{I} : \mathcal{W} \to M$$

is an invariant readout. The system is invariant-closed *when every rule in $\mathcal{R}$ preserves $\mathcal{I}$ on its legal input rows.*

Theorem 9413.596 (Invariant exclusion)**.** *Let $\mathfrak{G}$ be invariant-closed. If a target packet w_* has an invariant value not attained by any source packet in $\mathcal{A}$, then*

$$w_* \notin \mathcal{T}.$$

Proof. Every element of $\mathcal{T}$ is obtained from a source packet by a finite chain of generation rules. Since each rule preserves $\mathcal{I}$, the invariant value of a generated packet agrees with the value of some source packet. The stated separation of invariant values therefore prevents w_* from lying in $\mathcal{T}$. □

Definition 9413.597 (Fixed-half invariant for RHRoute packets)**.** *For an RHRoute formal bridge packet P over a critical-strip coordinate s, the* fixed-half invariant *is the normal readout*

$$\mathcal{I}_{\text{half}}(P) := \delta(s) := \Re(s) - \frac{1}{2}.$$

The fixed subcarrier is

$$\mathcal{W}^J := \{P \in \mathcal{W} : \mathcal{I}_{\text{half}}(P) = 0\}.$$

Equivalently, $\mathcal{W}^J$ is the subcarrier on which the mirror readback gives J-fixedness and hence the critical-line row.

Definition 9413.598 (RH-free generation discipline)**.** *A generation discipline for RHRoute packets is* RH-free *when its rules are allowed to use finite prime-window data, mirror transport, classifier transport, analytic gluing, compatible finite-window closure, and route replay, but are not allowed to use $\mathsf{OnCritLine}(s)$ or $\delta(s) = 0$ as an input rule. The purpose of the discipline is to separate a genuine invariant-preservation proof from a hidden insertion of the desired fixed-half output.*

Definition 9413.599 (Fixed-half closed generation)**.** *An RH-free generation discipline is* fixed-half closed *when*

$$P \in \mathcal{W}^J \quad \Longrightarrow \quad R(P) \in \mathcal{W}^J$$

for every legal rule R and every legal input packet P of that rule. For multi-input rules, the condition is that all inputs lie in $\mathcal{W}^J$ and the output lies in $\mathcal{W}^J$. This is the formal invariant row corresponding to fixed-half closure.

Theorem 9413.600 (Fixed-half generation excludes off-half outputs)**.** *Let an RH-free generation discipline be fixed-half closed, and suppose its source packets lie in $\mathcal{W}^J$. Then every generated packet has $\delta = 0$. In particular, no packet with*

$$s = \frac{1}{2} + r + i\gamma, \qquad r\#0,$$

is generated by that discipline.

Proof. Apply Definition 9413.596 with $\mathcal{I} = \mathcal{I}_{\text{half}}$. The source packets have invariant value 0, and every legal rule preserves the fixed subcarrier. Hence every generated packet has normal readout 0. A packet with $r\#0$ has $\mathcal{I}_{\text{half}} = r\#0$, so it cannot be in the generated carrier. □

Definition 9413.601 (Generated-zero coverage row)**.** *The* generated-zero coverage row *for the completed zeta carrier is*

$$\mathsf{GeneratedZeroCoverage}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{CritZero}(s) \to \sum_{P:\mathcal{T}_\zeta} \mathsf{SameZeroReadout}(P, s) \right),$$

where $\mathcal{T}_\zeta$ is the generated RHRoute packet carrier. The row says that every completed critical-strip zero is covered by a generated packet with the same zero readout. It is the invariant-system form of active capture and zero presentation completeness.

Theorem 9413.602 (Generation coverage and fixed-half closure are sufficient)**.** *Assume:*

$$\mathsf{GeneratedZeroCoverage}_\zeta,$$

and assume the RHRoute generation discipline is RH-free and fixed-half closed with fixed-half source packets. Then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have coordinate s. Generated-zero coverage supplies a generated packet P with the same zero readout as s. By Definition 9413.600, P has $\delta = 0$. Transporting this row along the same-zero readout gives

$$\delta(s) = 0.$$

The mirror-fixed algebra reads this as J-fixedness, hence as $\mathsf{OnCritLine}(s)$. Varying z gives the fixed-half section. □

Definition 9413.603 (Energy form of the invariant)**.** *Let a generated packet tower over $s = 1/2 + r + i\gamma$ have active nodes $a \in A_n$ with heights $h_a > 0$ and local channels*

$$U_a(s) := e^{-(s-1/2)h_a}.$$

Define

$$d_a(s) := \log |U_a(s)| = -rh_a,$$
$$\Omega_n(s) := (d_a(s))_{a \in A_n}, \qquad W_n := \sum_{a \in A_n} h_a^2.$$

The energy invariant is the square norm

$$|\Omega_n(s)|^2 := \sum_{a \in A_n} d_a(s)^2.$$

Lemma 9413.604 (Energy reads the normal coordinate)**.** *For every finite active window A_n,*

$$|\Omega_n(s)|^2 = r^2 W_n.$$

Proof. By definition, $d_a(s) = -rh_a$. Therefore

$$|\Omega_n(s)|^2 = \sum_{a \in A_n} (-rh_a)^2 = r^2 \sum_{a \in A_n} h_a^2 = r^2 W_n.$$

□

Theorem 9413.605 (Energy-bounded generation excludes off-half towers)**.** *Suppose a generated packet tower over $s = 1/2 + r + i\gamma$ satisfies cofinal growth*

$$W_n \to \infty$$

and a boundary-stability bound

$$|\Omega_n(s)|^2 \leq B_s$$

for a finite bound B_s independent of n. Then $r = 0$.

Proof. If $r\#0$, then $r^2 > 0$. By Definition 9413.604,

$$|\Omega_n(s)|^2 = r^2 W_n.$$

The cofinal growth row forces the right side to be unbounded, while the boundary-stability row gives a uniform finite bound. This contradiction excludes $r\#0$, and located fixed-half collapse gives $r = 0$. □

Corollary 9413.606 (The invariant proof load is two bridge rows)**.** *The invariant route closes after two zeta-specific bridge rows are supplied:*

$$\mathsf{ActiveCapture}_\zeta \quad \textit{and} \quad \mathsf{BoundaryStability}_\zeta.$$

Proof. Active capture supplies generated-zero coverage by constructing a zero-relevant generated packet for each completed zero. Boundary stability supplies fixed-half closure, or in the tower reading a uniform energy bound on the same zero-relevant packet family. The invariant argument itself is then formal: generated packets preserve the fixed-half invariant, and off-half packets have a different invariant value or unbounded positive energy. Thus the remaining proof load is source construction and independent boundary stability, not another restatement of the critical-line surface. □

9413.47 Triadic Ontic Invariant Readback

The fixed-half invariant of section 9413.46 is the zeta-facing readout of a higher BEDC invariant. It should not be taken as the primitive invariant itself. The primitive row is the closure preservation of the ontic triad: distinction is not lost, boundary transport remains legal, and recursive generation keeps a unique auditable interpretation.

Definition 9413.607 (Triadic ontic invariant)**.** *The* triadic ontic invariant *is the packet*

$$\mathcal{I}_{\mathrm{BEDC}} := (\mathcal{I}_{\mathsf{P}}, \mathcal{I}_{\mathsf{U}}, \mathcal{I}_{\mathsf{T}}),$$

where:

$$\mathcal{I}_{\mathsf{P}} = \textit{distinction, prime charge, and ledger provenance are preserved},$$

$$\mathcal{I}_{\mathsf{U}} = \textit{symmetry, unit boundary, and legal gauge are preserved},$$

and

$$\mathcal{I}_{\mathsf{T}} = \textit{recursion, finite-window closure, and unique readback are preserved.}$$

The invariant is not one of these rows in isolation. It is the closure condition that the three rows remain compatible under generation.

Definition 9413.608 (Triadic closed generation rule)**.** *Let R be a legal generation rule on RHRoute packets. The rule is* triadic closed *when it sends packets satisfying $\mathcal{I}_{\mathrm{BEDC}}$ to packets satisfying $\mathcal{I}_{\mathrm{BEDC}}$, and the output ledger records how the three components are transported:*

$$\mathsf{PrimeLedgerOK}, \qquad \mathsf{BoundaryGaugeOK}, \qquad \mathsf{RecursiveClosureOK}.$$

The first row records source provenance, prime-window data, and refusal of unlicensed collapse. The second row records mirror, unit-boundary, and compact gauge discipline. The third row records the recursive construction and its unique finite readback.

Theorem 9413.609 (Triadic closure is preserved by generated packets)**.** *If the source packets satisfy $\mathcal{I}_{\mathrm{BEDC}}$ and every legal generation rule is triadic closed, then every generated packet satisfies $\mathcal{I}_{\mathrm{BEDC}}$.*

Proof. This is structural induction on the generated packet. A source packet satisfies the invariant by hypothesis. For an inductive step, assume the input packets satisfy $\mathcal{I}_{\mathrm{BEDC}}$. Since the applied rule is triadic closed, its output carries the three transported rows PrimeLedgerOK, BoundaryGaugeOK, and RecursiveClosureOK. Hence the output satisfies $\mathcal{I}_{\mathrm{BEDC}}$. Repeating the argument along the finite generation tree gives the claim. □

Definition 9413.610 (Zeta readback of the triadic invariant)**.** *For a zeta-facing packet over*

$$s = \frac{1}{2} + r + i\gamma,$$

the local channel at a displayed node of height $h_a > 0$ is

$$U_a(s) = e^{-(s-1/2)h_a} = e^{-rh_a} e^{-i\gamma h_a}.$$

The zeta readback of $\mathcal{I}_{\mathrm{BEDC}}$ is:

$$\mathcal{I}_{\mathsf{P}} \rightsquigarrow \textit{the prime charge fixes } h_a,$$

$$\mathcal{I}_{\mathsf{U}} \rightsquigarrow \textit{the allowed gauge contains unit phase and compact prime fibres},$$

$$\mathcal{I}_{\mathsf{T}} \rightsquigarrow \textit{the packet is read through a finite or cofinal recursive tower.}$$

Under this readback, the real positive factor e^{-rh_a} is legal only when it is trivial.

Theorem 9413.611 (Triadic ontic readback gives fixed-half closure)**.** *For a zeta-facing generated packet, preservation of $\mathcal{I}_{\mathrm{BEDC}}$ implies the fixed-half readout*

$$r = 0.$$

Proof. The prime-ledger component supplies a positive height $h_a > 0$ for every active node. The recursive component supplies the finite or cofinal readback in which the same node is part of the generated packet. The boundary-gauge component allows the unit phase $e^{-i\gamma h_a}$ and compact prime-register fibres, but it does not allow an unledgered noncompact real positive scale. Therefore the factor

$$e^{-rh_a}$$

must be trivial on every zero-relevant generated packet. Since $h_a > 0$, this gives $r = 0$. □

Corollary 9413.612 (Fixed-half is a projection, not the primitive invariant)**.** *The fixed-half row used by RHRoute is the zeta projection of the triadic ontic invariant, not the primitive invariant of BEDC.*

Proof. The primitive invariant is the preservation of distinction provenance, legal boundary gauge, and recursive readback. In the zeta projection, these rows respectively determine the active heights, restrict the hidden carrier to unit phase and compact prime-register data, and make the finite or cofinal tower auditable. Their combined readback excludes the nontrivial real positive factor e^{-rh_a}. This exclusion is then read as $r = 0$, equivalently as the fixed-half row. Thus fixed-half is the output surface of the invariant under the zeta projection. □

Corollary 9413.613 (Generated-zero coverage remains the missing zeta row)**.** *Even if every generated RHRoute packet preserves $\mathcal{I}_{\mathrm{BEDC}}$, the RH fixed-half section still requires generated-zero coverage for the completed zeta zero ledger.*

Proof. By Definition 9413.611, a generated zeta-facing packet satisfying the triadic ontic invariant has fixed-half readout. To obtain a section over all completed critical-strip zero rows, each such zero must first be covered by a generated packet with the same zero readout. This is precisely the source row isolated in Definition 9413.601; it is not supplied by the invariant itself. □

Definition 9413.614 (Generated-zero internal section)**.** *The* generated-zero internal section *is the section over the generated RHRoute carrier*

$$\Phi_{\mathrm{gen}} : \mathcal{T}_\zeta \to \sum_{P:\mathcal{T}_\zeta} \mathsf{OnCritLine}(\mathsf{Read}(P)).$$

It asserts that every BEDC-generated zeta-facing packet has fixed-half readout. It is weaker than the RH fixed-half section, whose domain is the completed critical-strip zero carrier $\mathcal{Z}_\zeta$.

Theorem 9413.615 (Triadic closure gives only the generated section)**.** *If generated RHRoute packets preserve $\mathcal{I}_{\mathrm{BEDC}}$, then the generated-zero internal section exists.*

Proof. Let $P : \mathcal{T}_\zeta$ be a generated packet with zeta readout $s = 1/2 + r + i\gamma$. By preservation of $\mathcal{I}_{\mathrm{BEDC}}$ and Definition 9413.611, the noncompact real positive factor in the local channel is trivial. Thus $r = 0$, and the mirror-fixed readback gives $\mathsf{OnCritLine}(s)$. Assigning this witness to every generated packet gives Φ_{gen}. □

Definition 9413.616 (Coverage gap)**.** *The* coverage gap *is the difference between the generated carrier and the completed zero carrier:*

$$\mathcal{T}_\zeta \quad \textit{versus} \quad \mathcal{Z}_\zeta.$$

It is closed only by a row of the form

$$\mathsf{GeneratedZeroCoverage}_\zeta : \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{CritZero}(s) \to \sum_{P:\mathcal{T}_\zeta} \mathsf{SameZeroReadout}(P, s) \right).$$

This row is the same proof role as zero presentation completeness and active capture.

Theorem 9413.617 (Many generated zeros do not close the coverage gap)**.** *A displayed family of generated zero packets, even an infinite one, does not by itself imply* $\mathsf{GeneratedZeroCoverage}_\zeta$.

Proof. A displayed family gives a map from some index carrier I into $\mathcal{T}_\zeta$, hence a subset of completed zero readouts after projection. Coverage requires the opposite direction: every completed critical-strip zero row must admit a generated packet with the same readout. To derive that direction from an indexed family one must also prove that the indexing map is complete for the zero ledger, handles multiplicity and conjugation, and neither omits nor duplicates the zero rows in a way that loses the section target. These requirements are exactly a coverage theorem, not a consequence of having many generated examples. □

Corollary 9413.618 (Ordinary successor induction is not the coverage proof)**.** *The structural induction proving preservation of* $\mathcal{I}_{\mathrm{BEDC}}$ *is an induction on generated packets, not an induction over the completed zeta zero set.*

Proof. Structural induction consumes the constructors of $\mathcal{T}_\zeta$. It proves that every object already generated by those constructors preserves the triadic invariant. An induction over completed zeros would require a separately defined enumeration

$$\rho : \mathsf{Nat}^\uparrow \to \mathcal{Z}_\zeta$$

together with a proof that every completed zero occurs in that enumeration and that generated coverage passes from $\rho(n)$ to $\rho(n{+}1)$. Such data are not supplied by the generated-packet induction. Therefore the invariant proof and the coverage proof are logically distinct. □

Definition 9413.619 (Finite zero box)**.** *A* finite zero box *is a bounded rational rectangle*

$$\mathcal{B} \subseteq \left\{ s : \mathsf{Complex}^\uparrow : 0 \le \Re(s) \le 1 \right\}$$

whose boundary is separated from the completed zeta zero ledger. Its zero row is the finite packet

$$\mathcal{Z}_\zeta(\mathcal{B}) := \{ s \in \mathcal{B} : \mathsf{CritZero}(s) \},$$

with multiplicity, conjugation, and boundary-separation data carried in the ledger.

Definition 9413.620 (Box cover)**.** *For a finite zero box* $\mathcal{B}$*, the row*

$$\mathsf{BoxCover}(\mathcal{B})$$

means that every zero row in $\mathcal{Z}_\zeta(\mathcal{B})$ *is covered by a generated RHRoute packet:*

$$\prod_{s:\mathcal{Z}_\zeta(\mathcal{B})} \sum_{P:\mathcal{T}_\zeta} \mathsf{SameZeroReadout}(P, s).$$

This is a localized form of $\mathsf{GeneratedZeroCoverage}_\zeta$.

Definition 9413.621 (Box-cover induction row)**.** *A* box-cover induction row *consists of an increasing exhaustion*

$$\mathcal{B}_0 \subseteq \mathcal{B}_1 \subseteq \mathcal{B}_2 \subseteq \cdots$$

of the critical-strip zero ledger together with:

$$\mathsf{BoxCover}(\mathcal{B}_0),$$

$$\prod_{N:\mathsf{Nat}^{\uparrow}} (\mathsf{BoxCover}(\mathcal{B}_N) \to \mathsf{BoxCover}(\mathcal{B}_{N+1})),$$

and the exhaustion row

$$\prod_{s:\mathcal{Z}_\zeta} \sum_{N:\mathsf{Nat}^{\uparrow}} s \in \mathcal{B}_N.$$

The successor row is not a numerical-zero enumeration. It is a localized coverage step for all zero islands entering the next box.

Theorem 9413.622 (Box-cover induction gives generated coverage)**.** *A box-cover induction row implies*

$$\mathsf{GeneratedZeroCoverage}_\zeta.$$

Proof. By induction on N, the base row and successor row give

$$\mathsf{BoxCover}(\mathcal{B}_N)$$

for every $N : \mathsf{Nat}^{\uparrow}$. Let $s : \mathcal{Z}_\zeta$ be a completed critical-strip zero row. The exhaustion row supplies some N with $s \in \mathcal{B}_N$. Applying $\mathsf{BoxCover}(\mathcal{B}_N)$ to the corresponding localized zero row gives a generated packet $P : \mathcal{T}_\zeta$ with the same zero readout as s. Since s was arbitrary, generated-zero coverage follows. □

Definition 9413.623 (Active capture step)**.** *The* active capture step *for $\mathcal{B}_N \subseteq \mathcal{B}_{N+1}$ is the source row that covers each zero island in*

$$\mathcal{B}_{N+1} \setminus \mathcal{B}_N$$

by a canonical finite packet. For each such island D, it supplies

$$\Xi = F_D + T_D$$

on a boundary-separated disk around the island and the Rouché gap

$$|T_D| < |F_D| \qquad \text{on } \partial D.$$

It also supplies the interpretation of each zero of F_D as a generated RHRoute packet.

Corollary 9413.624 (Active capture steps are the coverage content)**.** *The box-cover induction row reduces generated-zero coverage to active capture steps for the boundary-separated zero islands of the exhaustion.*

Proof. The base box supplies the initial localized cover. For the transition from $\mathcal{B}_N$ to $\mathcal{B}_{N+1}$, all previously covered zero rows remain covered by restriction. The only uncovered rows lie in the added zero islands. An active capture step supplies Rouché capture and generated-packet interpretation for exactly those islands, hence constructs $\mathsf{BoxCover}(\mathcal{B}_{N+1})$ from $\mathsf{BoxCover}(\mathcal{B}_N)$. □

Corollary 9413.625 (Full RH requires coverage after invariant preservation)**.** *The route from the triadic ontic invariant to the RH fixed-half section has two independent stages:*

$$\mathcal{I}_{\mathrm{BEDC}} \text{ is preserved on } \mathcal{T}_\zeta \quad \Longrightarrow \quad \Phi_{\mathrm{gen}},$$

and

$$\mathsf{GeneratedZeroCoverage}_\zeta \quad \Longrightarrow \quad \Phi : \mathcal{Z}_\zeta \to \mathcal{H}_\zeta.$$

Proof. The first implication is Definition 9413.615. The second consumes the coverage row: for a completed zero s, choose a generated packet P with the same readout, use Φ_{gen} on P, and transport the fixed-half witness along the same-readout row. Without the coverage row, the generated section has the wrong domain for the RH fixed-half section. □

9413.48 Zero Packet Refinement Category

The coverage row can be organized categorically, but the categorical language must be used with a precise proof role. A category of finite zero packets supplies objects, refinement morphisms, and compatible towers. It does not by itself supply induction. Induction enters only after the generated carrier has an initial-algebra or structural recursion principle, and after the finite packet tower has a well-founded level structure whose refinement steps are explicitly audited.

Definition 9413.626 (Finite BEDC zero packet)**.** *A* finite BEDC zero packet *is a tuple*

$$P := (n, W, F, T, D, \mathcal{L}, \mathcal{B}, \Omega)$$

where n is a level or precision index, W is a finite prime or resonance window, F is a dominant finite packet, T is a tail packet, $D \subseteq \mathsf{Complex}^{\uparrow}$ is a rational zero box or zero island, $\mathcal{L}$ is the source, tail, phase, prime-register, and defect ledger, $\mathcal{B}$ is the boundary certificate, and Ω is the normal-defect readout attached to the packet.

Definition 9413.627 (Refinement morphism)**.** *For two finite zero packets P_m and P_n with $m \geq n$, a* refinement morphism

$$\phi_{m,n} : P_m \to P_n$$

is a projection from the finer packet to the coarser packet satisfying:

$$D_m \subseteq D_n, \qquad W_n \subseteq W_m,$$

$$\phi_{m,n}^* \mathcal{L}_n = \mathcal{L}_m|_n, \qquad \phi_{m,n}^* \mathcal{B}_n = \mathcal{B}_m|_n,$$

and

$$\Omega_m|_n = \Omega_n + \varepsilon_{m,n}$$

with an explicitly ledgered refinement error $\varepsilon_{m,n}$. The morphism says that P_m is a higher-resolution audit of P_n, not an independent replacement for it.

Definition 9413.628 (Zero packet refinement category)**.** *The* zero packet refinement category

$$\mathsf{Z}_{\mathrm{BEDC}}$$

has finite BEDC zero packets as objects and refinement morphisms as arrows. It is required to satisfy:

$$\phi_{n,n} = \mathrm{id}_{P_n},$$

and

$$\phi_{k,n} = \phi_{m,n} \circ \phi_{k,m} \qquad (k \geq m \geq n).$$

Theorem 9413.629 (Refinement packets form a category)**.** *If the refinement projections preserve zero boxes, windows, ledger data, boundary data, and defect readouts, and if they satisfy the identity and composition laws of Definition 9413.628, then the finite zero packets form the category* $\mathsf{Z}_{\mathrm{BEDC}}$*.*

Proof. The objects and arrows are given by Definition 9413.626 and Definition 9413.627. The identity arrow on P_n preserves every component by definitional readback. For composable refinements

$$P_k \xrightarrow{\phi_{k,m}} P_m \xrightarrow{\phi_{m,n}} P_n,$$

the inclusions of boxes and windows compose, and the pullback equations for ledger, boundary, and defect rows compose. The stipulated composition law identifies this composite with $\phi_{k,n}$. Thus the category axioms hold. □

Definition 9413.630 (BEDC zero tower)**.** *A* BEDC zero tower *is a functor*

$$P_\bullet : \mathsf{Nat}^{\uparrow op} \to \mathsf{Z}_{\mathrm{BEDC}}.$$

Equivalently, it is a compatible tower

$$P_0 \leftarrow P_1 \leftarrow P_2 \leftarrow \cdots$$

with refinement morphisms

$$\phi_{n+1,n} : P_{n+1} \to P_n.$$

Definition 9413.631 (Convergent zero tower)**.** *Let D_n be the zero box of P_n. A BEDC zero tower is* convergent *when*

$$D_{n+1} \subseteq D_n$$

and

$$\mathrm{diam}(D_n) \to 0.$$

When the boxes are nonempty and located-compact, the unique point in

$$\bigcap_n D_n$$

is denoted s_∞ and called the limit readout of the tower.

Theorem 9413.632 (Nested zero boxes have a unique limit readout)**.** *If the zero boxes of a BEDC zero tower are nonempty located-compact sets, satisfy $D_{n+1} \subseteq D_n$, and have diameter tending to zero, then the tower has a unique limit readout*

$$s_\infty \in \bigcap_n D_n.$$

Proof. The nested located-compact intersection supplies a point in the intersection. If x and y are both in every D_n, then

$$|x - y| \leq \mathrm{diam}(D_n)$$

for every n. Since the diameters tend to zero, $|x - y| = 0$, hence $x = y$. Thus the intersection has a unique point. □

Definition 9413.633 (Active-capture zero tower)**.** *A convergent BEDC zero tower is an* active-capture zero tower *when each level carries a decomposition*

$$\Xi = F_n + T_n$$

on a neighbourhood of D_n and a Rouché boundary gap

$$|T_n| < |F_n| \qquad \text{on } \partial D_n,$$

and the winding or local-degree row is preserved by the refinement morphisms.

Theorem 9413.634 (Active-capture convergence gives a completed zero)**.** *Let $P_\bullet$ be a convergent active-capture zero tower. If each D_n contains the same single completed-zero island of Ξ, then its limit readout satisfies*

$$\Xi(s_\infty) = 0.$$

Proof. By the Rouché boundary gap, F_n and Ξ have the same local zero count in D_n. The refinement maps preserve the local-degree row, and the boxes are nested, so the captured island is the same at each level. By Definition 9413.632, the boxes converge to s_∞. Since each box contains the same completed-zero island and the diameters tend to zero, the island readout is s_∞, hence $\Xi(s_\infty) = 0$. □

Definition 9413.635 (Boundary-stable zero tower)**.** *Let $s_\infty = 1/2+r+i\gamma$ be the limit readout of a zero tower. A convergent active-capture tower is* boundary-stable *when it carries defect extraction*

$$|\Omega_n(s_\infty)|^2 = r^2 W_n + o(W_n),$$

cofinal growth

$$W_n \to \infty,$$

and a zero-compatible bound

$$|\Omega_n(s_\infty)|^2 \le B$$

for a finite B independent of n.

Theorem 9413.636 (Boundary-stable zero towers have fixed-half limits)**.** *The limit readout of a boundary-stable zero tower satisfies*

$$r = 0.$$

Proof. If $r\#0$, then $r^2 > 0$. Cofinal growth gives

$$r^2 W_n \to \infty.$$

The defect extraction row gives

$$|\Omega_n(s_\infty)|^2 = r^2 W_n + o(W_n),$$

so the defect energy is unbounded. This contradicts the zero-compatible bound

$$|\Omega_n(s_\infty)|^2 \le B.$$

Hence $\neg(r\#0)$, and located fixed-half collapse gives $r = 0$. □

Corollary 9413.637 (Tower route separates convergence from coverage)**.** *A boundary-stable active-capture zero tower converges to a completed zero on the fixed-half surface. The remaining global row is that every completed critical-strip zero is the limit readout of such a tower.*

Proof. Definition 9413.634 gives the completed-zero row for the limit readout, while Definition 9413.636 gives the fixed-half row. This proves the statement for a tower once the tower is supplied. It does not construct a tower for every completed zero. That construction is the tower form of $\mathsf{GeneratedZeroCoverage}_\zeta$ and the active-capture source obligation of Definition 9413.623. □

9413.49 Prime-Adic Local Slicing

The prime-window and recursive-parity routes may be sharpened by reading each prime factor as a local slice. This is not a claim that the zeros of the Riemann zeta function are zeros of a p-adic zeta interpolation. The role of the p-adic language is local decomposition: each prime coordinate supplies a valuation direction and a unit direction, and the completed complex readout compresses all such local coordinates into the single complex parameter s.

Definition 9413.638 (Prime-adic slice)**.** *For a displayed prime p, the* prime-adic slice *is the local coordinate that records the p-valuation direction separately from the unit direction. Arithmetic factorization writes a nonzero rational row as*

$$x = \prod_p p^{v_p(x)} \cdot u,$$

with only finitely many nonzero valuation coordinates. In the p-local readout this becomes

$$x = p^{v_p(x)} u_p, \qquad u_p \in \mathbb{Z}_p^\times.$$

Thus the valuation $v_p(x)$ is the radial depth coordinate, while u_p is the unit coordinate. The word "slice" means this local valuation/unit separation; it does not introduce an additional global zero predicate.

Definition 9413.639 (Prime-adic Euler local coordinate)**.** *The Euler local factor at p is*

$$Z_p(s) := (1 - p^{-s})^{-1}.$$

On its convergence readout,

$$Z_p(s) = 1 + p^{-s} + p^{-2s} + p^{-3s} + \cdots .$$

The exponent k in p^{-ks} is the valuation depth in the p-slice. The centered local coordinate is

$$U_p(s) := p^{1/2-s}.$$

For

$$s = \frac{1}{2} + \delta + i\tau,$$

it decomposes as

$$U_p(s) = p^{-\delta} e^{-i\tau \log p}.$$

The factor $p^{-\delta}$ is the radial valuation readout and $e^{-i\tau \log p}$ is the unit-phase readout.

Theorem 9413.640 (The prime-adic unit surface is the fixed-half surface)**.** *For any displayed prime p and any complex row $s = \sigma + i\tau$,*

$$|U_p(s)| = 1 \quad \Longleftrightarrow \quad \sigma = \frac{1}{2}.$$

Consequently

$$\left(\prod_{p:\mathsf{Prime}^\uparrow} |U_p(s)| = 1 \right) \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Proof. By Definition 9413.639,

$$|U_p(s)| = |p^{1/2-s}| = p^{1/2-\sigma}.$$

Since $p > 1$, the equality $p^{1/2-\sigma} = 1$ holds exactly when $1/2-\sigma = 0$. This is $\sigma = 1/2$. The product statement follows because any displayed prime already detects the same exponent, and all prime slices read the same centered real displacement. □

Definition 9413.641 (Prime-adic finite-window defect)**.** *For a nonempty finite prime window S, the prime-adic finite-window defect is*

$$D_S^{\mathrm{loc}}(s) := \sum_{p\in S} \log |U_p(s)|.$$

Its normalized readout is

$$\overline{D}_S^{\mathrm{loc}}(s) := \frac{D_S^{\mathrm{loc}}(s)}{\sum_{p\in S} \log p}.$$

This is the same local slice data as the normalized prime-window defect of Definition 9413.166, written with the valuation/unit terminology exposed.

Theorem 9413.642 (Local slices read the same normal defect)**.** *If*

$$s = \frac{1}{2} + \delta + i\tau,$$

then for every nonempty finite prime window S,

$$\overline{D}_S^{\mathrm{loc}}(s) = -\delta.$$

Proof. For each $p \in S$,

$$\log |U_p(s)| = \log |p^{1/2-s}| = -\delta \log p.$$

Therefore

$$D_S^{\mathrm{loc}}(s) = -\delta \sum_{p\in S} \log p.$$

The denominator is positive because S is nonempty and every displayed prime has positive logarithm. Dividing gives the displayed formula. This is the local-slice form of Definition 9413.167. □

Corollary 9413.643 (Off-half rows are locally nonunit in every slice)**.** *If $\delta(s)\#0$, then every displayed prime slice is nonunit in the centered radial readout:*

$$|U_p(s)|\#1.$$

Equivalently, every nonempty finite prime window carries nonzero normalized local defect.

Proof. If $\delta(s)\#0$, then the formula

$$|U_p(s)| = p^{-\delta(s)}$$

is apart from 1 in the located radial readout. The finite-window statement follows from Definition 9413.642. □

Definition 9413.644 (Prime-adic local-global defect incompatibility)**.** *The prime-adic local-global defect incompatibility row is*

$$\mathsf{PrimeAdicLocalGlobalDefectIncompatibility}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \Bigg(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} (\overline{D}_S^{\mathrm{loc}}(s$$

It states that a completed zeta zero cannot coexist with a nonunit local radial defect visible in any finite family of prime-adic slices.

Theorem 9413.645 (Prime-adic local-global incompatibility is conditionally sufficient)**.** *Together with the located fixed-half collapse row, prime-adic local-global defect incompatibility is sufficient for the RH fixed-half section.*

Proof. Let s carry a critical-strip zero row and write

$$s = \frac{1}{2} + \delta + i\tau.$$

Suppose $\delta\#0$. Choose any nonempty finite prime window S. By Definition 9413.642,

$$\overline{D}_S^{\mathrm{loc}}(s) = -\delta,$$

hence $\overline{D}_S^{\mathrm{loc}}(s)\#0$. This supplies the positive data excluded by Definition 9413.644. Therefore $\neg(\delta\#0)$. The located fixed-half collapse row gives $\delta = 0$, and Definition 9413.313 reads this as the fixed-half section. □

Corollary 9413.646 (Prime-adic slicing isolates the remaining local-global gap)**.** *Prime-adic slicing does not replace the zero problem by a theorem about p-adic zeta zeros. It identifies the remaining RH-facing gap as the local-global row asserting that nonunit local radial data cannot be classified as a completed global zeta zero.*

Proof. Definition 9413.640 identifies the fixed-half surface with simultaneous local unit readout. Definition 9413.642 shows that off-half displacement is visible in every finite family of prime slices. What is not supplied by either calculation is the zero-relevance bridge from visible local defect to exclusion of the completed zero ledger. That bridge is precisely Definition 9413.644, the local-slice form of the prime-sieve obstruction isolated in Definition 9413.484. □

9413.50 Prime-Adic Parity Energy Window

Prime-adic local slicing and recursive parity decomposition describe the same high-dimensional zeta carrier from two complementary directions. Prime-adic slicing separates the prime dimensions. Recursive parity decomposition separates the binary depth inside each prime dimension. The finite object inspected by the RH-facing defect route is therefore a finite window in $\mathsf{Prime}^\uparrow \times \mathsf{Nat}^\uparrow$, equipped with a normalized mirror-odd defect and its mirror-even second-order energy.

Definition 9413.647 (Prime-adic parity window)**.** *The* prime-adic parity tower *is the node carrier*

$$\mathsf{PAdicParityTower}_\zeta := \mathsf{Prime}^\uparrow \times \mathsf{Nat}^\uparrow.$$

For a node (p, j), the centered local-parity channel is

$$U_{p,j}^{\mathrm{pad}}(s) := p^{2j(1/2-s)}.$$

A prime-adic parity window *is a nonempty finite node family*

$$A \subset \mathsf{PAdicParityTower}_\zeta.$$

Its weight, signed defect, normalized signed defect, and second-order energy are

$$W_A^{\mathrm{pad}} := \sum_{(p,j)\in A} 2^j \log p,$$

$$D_A^{\mathrm{pad}}(s) := \sum_{(p,j)\in A} \log |U_{p,j}^{\mathrm{pad}}(s)|,$$

$$\overline{D}_A^{\mathrm{pad}}(s) := \frac{D_A^{\mathrm{pad}}(s)}{W_A^{\mathrm{pad}}}, \qquad E_A^{\mathrm{pad}}(s) := \left(\overline{D}_A^{\mathrm{pad}}(s)\right)^2.$$

Theorem 9413.648 (Prime-adic parity windows read one normal coordinate)**.** *If*

$$s = \frac{1}{2} + \delta + i\tau,$$

then every prime-adic parity window A satisfies

$$\overline{D}_A^{\mathrm{pad}}(s) = -\delta, \qquad E_A^{\mathrm{pad}}(s) = \delta^2.$$

Proof. For each node (p, j),

$$U_{p,j}^{\mathrm{pad}}(s) = p^{2^j(1/2-s)} = p^{-2^j\delta} e^{-i2^j\tau \log p}.$$

Thus

$$\log |U_{p,j}^{\mathrm{pad}}(s)| = -2^j \delta \log p.$$

Summing over A gives

$$D_A^{\mathrm{pad}}(s) = -\delta \sum_{(p,j)\in A} 2^j \log p = -\delta W_A^{\mathrm{pad}}.$$

Since A is nonempty, W_A^{pad} is a positive normalization row. Dividing gives $\overline{D}_A^{\mathrm{pad}}(s) = -\delta$, and squaring gives $E_A^{\mathrm{pad}}(s) = \delta^2$. □

Corollary 9413.649 (First-order defect is odd and second-order energy is even)**.** *For the mirror* $J(s) = 1 - \overline{s}$,

$$\overline{D}_A^{\mathrm{pad}}(J(s)) = -\overline{D}_A^{\mathrm{pad}}(s), \qquad E_A^{\mathrm{pad}}(J(s)) = E_A^{\mathrm{pad}}(s).$$

Proof. The mirror changes the normal coordinate from δ to $-\delta$ and keeps the time coordinate τ fixed. Apply Definition 9413.648 to s and $J(s)$. The normalized defect changes sign, while its square is unchanged. □

Theorem 9413.650 (Noncentrality is finite-window energy)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the following rows are equivalent as defect readouts:

$$\delta(s)\#0, \qquad \sum_{A:\mathsf{PAdicParityWindow}_{\neq\varnothing}} (E_A^{\mathrm{pad}}(s)\#0).$$

Moreover, if $\delta(s)\#0$*, then every prime-adic parity window carries the same nonzero energy* $\delta(s)^2$*.*

Proof. For every nonempty finite window A, Definition 9413.648 gives

$$E_A^{\mathrm{pad}}(s) = \delta(s)^2.$$

Thus apartness of $\delta(s)$ from zero supplies nonzero energy in every window. Conversely, a nonzero energy row in some window is a nonzero square of the normalized defect, hence an apartness witness for $\delta(s)$ in the located real readout. □

Definition 9413.651 (No prime-adic parity energy zero)**.** *The* no prime-adic parity energy zero *row is*

$$\mathsf{NoPAdicParityEnergyZero}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \times \sum_{A:\mathsf{PAdicParityWindow}_{\neq\varnothing}} (E_A^{\mathrm{pad}}(s)\#0) \to \bot \right).$$

It says that a completed zeta zero cannot carry nonzero second-order mirror-even energy on any finite prime-adic parity window.

Theorem 9413.652 (Prime-adic parity energy refusal is conditionally sufficient)**.** *Together with the located fixed-half collapse row,*

$$\mathsf{NoPAdicParityEnergyZero}_\zeta$$

is sufficient for the RH fixed-half section.

Proof. Let s carry a critical-strip zero row and write

$$s = \frac{1}{2} + \delta + i\tau.$$

Suppose $\delta\#0$. Choose any prime-adic parity window A. By Definition 9413.650,

$$E_A^{\mathrm{pad}}(s)\#0.$$

Together with the zero row and critical-strip row, this is the positive data excluded by Definition 9413.651. Hence $\neg(\delta\#0)$. The located fixed-half collapse row yields $\delta = 0$, and Definition 9413.313 reads this as the fixed-half section. □

Corollary 9413.653 (Mirror balance does not remove prime-adic parity energy)**.** *If a mirror pair $s, J(s)$ has $\delta(s)\#0$, then signed prime-adic parity defects cancel in each finite window, but the second-order window energy remains nonzero on both sides.*

Proof. By Definition 9413.649, the normalized defects of s and $J(s)$ have opposite signs. Their sum is therefore zero. The same corollary also gives equal energies, and Definition 9413.650 gives nonzero energy when $\delta(s)\#0$. Thus mirror symmetry can balance the first-order sign, but not the second-order distance from the fixed-half surface. □

Corollary 9413.654 (Combined local window isolates the remaining obstruction)**.** *The combined prime-adic parity window does not prove the zero exclusion by itself. It isolates the exact zero-relevance row still required: a completed zeta zero must be incompatible with nonzero second-order prime-adic parity energy.*

Proof. Definition 9413.648 and Definition 9413.649 are readout calculations. They identify the finite local observable and its mirror parity. The remaining claim is the incompatibility between that observable and the completed zero ledger, stated in Definition 9413.651. This is the second-order local-window form of the prime-sieve obstruction in Definition 9413.484 and the local-global gap isolated in Definition 9413.646. □

9413.51 Prime-Adic Parity Sphere Invariant

The prime-adic parity energy window has the same geometric role as the unit circle in the Pythagorean carrier: it turns a normalization row into a unit-boundary equation. For a finite window $A \subset \mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$, the centered coordinates $U^{\mathrm{pad}}_{p,j}(s)$ define a weighted sphere. The fixed-half surface is exactly the boundary of this finite sphere, while off-half rows lie strictly on one side of the boundary in the located radial readout.

Definition 9413.655 (Prime-adic parity radius)**.** *For a prime-adic parity window A, define the node weight*

$$h_{p,j} := 2^j \log p$$

and the normalized node weight

$$\omega^A_{p,j} := \frac{h_{p,j}}{W^{\mathrm{pad}}_A}.$$

The prime-adic parity radius square *is*

$$R^2_A(s) := \sum_{(p,j)\in A} \omega^A_{p,j} \left| U^{\mathrm{pad}}_{p,j}(s) \right|^2.$$

Writing $U^{\mathrm{pad}}_{p,j}(s) = x_{p,j}(s) + iy_{p,j}(s)$, this is the weighted sphere equation

$$R^2_A(s) = \sum_{(p,j)\in A} \omega^A_{p,j} \left(x_{p,j}(s)^2 + y_{p,j}(s)^2 \right).$$

Theorem 9413.656 (The weighted sphere boundary is the fixed-half surface)**.** *For every prime-adic parity window A,*

$$R^2_A(s) = 1 \quad \Longleftrightarrow \quad \Re(s) = \frac{1}{2}.$$

Proof. Write

$$s = \frac{1}{2} + \delta + i\tau.$$

By Definition 9413.647,

$$\left| U^{\mathrm{pad}}_{p,j}(s) \right|^2 = \left| p^{2^j(1/2-s)} \right|^2 = p^{-2^{j+1}\delta}.$$

If $\delta = 0$, every term is 1, and the normalized weights satisfy

$$\sum_{(p,j)\in A} \omega^A_{p,j} = 1.$$

Thus $R^2_A(s) = 1$.

Conversely, if $\delta > 0$, then every term $p^{-2^{j+1}\delta}$ is strictly smaller than 1, so the weighted average $R^2_A(s)$ is strictly smaller than 1. If $\delta < 0$, every term is strictly larger than 1, so $R^2_A(s)$ is strictly larger than 1. Hence equality with 1 is possible only when $\delta = 0$, equivalently $\Re(s) = 1/2$. □

Corollary 9413.657 (Sphere radius and normal energy define the same boundary)**.** *For every prime-adic parity window A, the rows*

$$R^2_A(s) = 1 \qquad \textit{and} \qquad E^{\mathrm{pad}}_A(s) = 0$$

define the same fixed-half boundary.

Proof. Definition 9413.656 identifies $R_A^2(s) = 1$ with $\Re(s) = 1/2$. By Definition 9413.648,

$$E_A^{\mathrm{pad}}(s) = \delta(s)^2.$$

Thus $E_A^{\mathrm{pad}}(s) = 0$ is the same row as $\delta(s) = 0$, hence the same fixed-half boundary. □

Definition 9413.658 (Finite Hamiltonian sphere state)**.** *For a prime-adic parity window A, let $\mathcal{H}_A$ be the finite Hilbert readout with displayed basis $|p, j\rangle$ for $(p, j) \in A$. Define*

$$H_A|p, j\rangle = h_{p,j}|p, j\rangle$$

and the normalized reference vector

$$|\mathbf{1}_A\rangle := \sum_{(p,j)\in A} \sqrt{\omega_{p,j}^A}\, |p, j\rangle.$$

The centered finite Hamiltonian state is

$$|\psi_A(s)\rangle := e^{-(s-1/2)H_A}|\mathbf{1}_A\rangle.$$

Theorem 9413.659 (Finite Hamiltonian norm is the sphere radius)**.** *For every prime-adic parity window A,*

$$\|\psi_A(s)\|^2 = R_A^2(s).$$

Consequently the fixed-half surface is the finite Hamiltonian unitary surface.

Proof. The operator H_A is diagonal on the displayed basis, so

$$e^{-(s-1/2)H_A}|p, j\rangle = p^{2^j(1/2-s)}|p, j\rangle = U_{p,j}^{\mathrm{pad}}(s)|p, j\rangle.$$

Therefore

$$\|\psi_A(s)\|^2 = \sum_{(p,j)\in A} \omega_{p,j}^A \left|U_{p,j}^{\mathrm{pad}}(s)\right|^2 = R_A^2(s).$$

If $\Re(s) = 1/2$, then $s - 1/2 = i\tau$ and $e^{-(s-1/2)H_A} = e^{-i\tau H_A}$ is unitary. Conversely, Definition 9413.656 identifies preservation of the unit norm of this displayed reference state with the fixed-half surface. □

Definition 9413.660 (Zero boundary sphere row)**.** *The* zero boundary sphere row *is*

$$\mathsf{ZeroBoundarySphere}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \prod_{A:\mathsf{PAdicParityWindow}_{\neq\varnothing}} (R_A^2(s) = 1) \right).$$

It says that a completed zeta zero lies on the unit-boundary sphere of every finite prime-adic parity window.

Theorem 9413.661 (Zero boundary sphere is conditionally sufficient)**.** *Together with the located fixed-half collapse row,*

$$\mathsf{ZeroBoundarySphere}_\zeta$$

is sufficient for the RH fixed-half section.

Proof. Let s carry a critical-strip zero row. The zero boundary sphere row gives $R_A^2(s) = 1$ for every prime-adic parity window A. Choose one such window. By Definition 9413.656, this gives $\Re(s) = 1/2$. Equivalently, $\delta(s) = 0$, and Definition 9413.313 reads the row as the RH fixed-half section. □

Corollary 9413.662 (Sphere boundary is the geometric form of energy refusal)**.** *The zero boundary sphere row and the no prime-adic parity energy zero row are two geometric presentations of the same RH-facing obstruction.*

Proof. By Definition 9413.657, $R_A^2(s) = 1$ and $E_A^{\mathrm{pad}}(s) = 0$ identify the same fixed-half boundary on each finite window. Therefore requiring zeta zero rows to land on all sphere boundaries is the boundary-surface form of excluding nonzero second-order prime-adic parity energy, as stated in Definition 9413.651. □

9413.52 Zeta-Like Projection Taxonomy

The finite objects used in the zeta route do not all have the same projection status. Some are direct finite projections of the Riemann zeta prime tower. Others are finite projections of the same zeta-like spectral mechanism without being truncations of the Riemann zeta function itself. The distinction is essential: it prevents the finite quartic and elliptic examples from being misread as literal Riemann-zeta windows while preserving their role as closed laboratories for the same observation, recursion, spectrum, normalization, symmetry, zero-ledger, and far-end pattern.

Definition 9413.663 (Direct Riemann-zeta finite projection)**.** *A* direct Riemann-zeta finite projection *is a finite readout obtained by restricting the Euler prime dimensions, the occupation depths, or the recursive parity nodes of the Riemann zeta carrier. The basic prime-window projection is*

$$\zeta_S(s) := \prod_{p \in S} (1 - p^{-s})^{-1}.$$

The finite occupation-depth projection is

$$\zeta_{S,M}(s) := \prod_{p \in S} \left(1 + p^{-s} + p^{-2s} + \cdots + p^{-Ms}\right).$$

The prime-adic parity windows

$$A \subset \mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$$

of Definition 9413.647 are direct finite windows of the recursive parity expansion of the same Euler carrier.

Definition 9413.664 (Universal zeta-like carrier)**.** *The* universal zeta-like carrier *is the schematic packet*

$$\mathcal{Z}_\Omega := (\mathsf{Obs}, \mathsf{Rec}, \mathsf{Spectrum}, \mathsf{Normalize}, \mathsf{Sym}, \mathsf{ZeroLedger}, \mathsf{FarEnd}).$$

It records the mechanism by which finite observations are recursively folded into spectral data, normalized to a unit or boundary surface, transported by a symmetry row, constrained by a zero ledger, and kept separate from an apophatic far-end absorption row.

Definition 9413.665 (Zeta-like finite projection)**.** *A* zeta-like finite projection *is a finite carrier projection of $\mathcal{Z}_\Omega$ that displays the rows*

$$\mathsf{Obs}, \quad \mathsf{Rec}, \quad \mathsf{Spectrum}, \quad \mathsf{Normalize}, \quad \mathsf{Sym}, \quad \mathsf{ZeroLedger}$$

inside a finite laboratory. It need not be a truncation of the Riemann zeta Euler product. Its projection status depends on preserving the mechanism, not on sharing the same prime-indexed carrier.

Theorem 9413.666 (Direct windows are zeta-like projections)**.** *Every direct Riemann-zeta finite projection is a zeta-like finite projection, but not every zeta-like finite projection is a direct Riemann-zeta finite projection.*

Proof. The direct windows inherit their observation rows from finite prime sets, their recursion from Euler multiplication or occupation-depth extension, their spectrum from the finite Hamiltonian readouts of Definition 9413.150, their normalization from the centered channels, their symmetry from the fixed-half mirror, and their zero-relevance boundary from the completed zeta route. Hence they are zeta-like projections.

The converse fails because a finite carrier may realize the same mechanism without being a prime-window truncation of the Riemann zeta function. The quartic carrier of Definition 9413.89 is such a case: its recurrence, generating function, companion spectrum, elliptic normalization, real projection, dominance row, and endpoint-zero law are finite spectral rows, not a finite Euler product over primes. □

Theorem 9413.667 (The quartic carrier is a zeta-like finite projection)**.** *The fixed quartic spectral carrier is a zeta-like finite projection of $\mathcal{Z}_\Omega$, not a direct Riemann-zeta finite projection.*

Proof. The observation row is the rational generating-function source $Z(t, y) = N(t, y)/D(t, y)$. The recursion row is the order-four recurrence for $Z_m(y)$. The spectrum row is the companion-matrix and spectral polynomial readout

$$\det(\lambda I - T(y)) = \Pi(\lambda, y).$$

The normalization row is the elliptic transport to

$$E: \quad Y^2 = 4\lambda^3 - 4\lambda + 1.$$

The symmetry and zero-ledger rows are the real projection, branch dominance, and endpoint-zero law isolated in Definition 9413.485. These rows instantiate the zeta-like mechanism. None of them is a finite restriction of the Riemann-zeta prime-indexed Euler product. □

Definition 9413.668 (Projection status ledger)**.** *The* projection status ledger *records, for a finite carrier F, which of the following rows it carries:*

$$\mathsf{DirectRiemannWindow}(F), \qquad \mathsf{ZetaLikeFiniteProjection}(F), \qquad \mathsf{ZeroRelevantBoundary}(F).$$

The first row implies the second when the finite Riemann window is read with its spectral, normalized, and zero-boundary ledgers. The second row does not imply the first. The third row is additional: it records that the finite projection has identified the boundary surface that actually controls its zero ledger.

Corollary 9413.669 (Projection status separates analogy from truncation)**.** *The finite quartic, elliptic, and finite-field style zeta models should be compared with Riemann zeta through the zeta-like mechanism ledger, not through literal Euler-product truncation.*

Proof. A literal truncation must preserve the prime-indexed Euler carrier. The quartic model preserves recurrence, spectral normalization, symmetry, and zero-boundary rows instead. Therefore its correct comparison point is $\mathcal{Z}_\Omega$, not the direct finite Euler window $\zeta_S(s)$. This keeps the finite model useful without overstating its relation to Riemann zeta. □

Theorem 9413.670 (Finite projection energy is the completed-readout test)**.** *For the Riemann-zeta direct windows, an off-half critical-strip row is seen by every prime-adic parity window as nonzero second-order energy. Thus the RH-facing completed-readout test is whether this finite projection energy can coexist with the completed global zero ledger.*

Proof. By Definition 9413.650, if $s = 1/2+\delta+i\tau$ and $\delta(s)\#0$, then every prime-adic parity window A carries

$$E_A^{\mathrm{pad}}(s)\#0.$$

By Definition 9413.662, this is the same obstruction as failure to lie on the finite prime-adic parity unit sphere. The remaining completed-readout test is therefore not visibility of the defect, which is already finite, but zero-relevance of that defect in the completed zeta carrier. □

Corollary 9413.671 (Finite projections identify the no-absorption target)**.** *The finite-projection framework reduces the RH-facing gap to the claim that nonzero finite-projection normal energy cannot be absorbed by the infinite completed zeta readout as a zero row.*

Proof. Direct finite windows detect the normal energy by Definition 9413.670. Zeta-like finite projections such as the quartic carrier show why zero-relevance must be separately proved: a visible spectral branch or defect matters only after dominance, normalization, and zero-ledger rows identify the relevant boundary. Thus the remaining target is the no-absorption row already isolated by Definition 9413.180 and the defect-zero incompatibility row of Definition 9413.202. □

9413.53 Global Prime Energy and the No-Hair Boundary

The finite prime-window defect row is a first-order signed readout. As a signed row it can reverse under the mirror and can participate in global phase cancellation. The square of the same radial readout has a different status. It is a positive normal energy on each prime dimension. Positive energy has no opposite sign that could cancel it by mirror pairing, parity rearrangement, or movement of terms across a global equation.

Definition 9413.672 (Prime normal square energy)**.** *For the centered prime channel*

$$U_p(s) := p^{1/2-s},$$

define the prime normal defect and its square energy by

$$d_p(s) := \log|U_p(s)|, \qquad e_p(s) := d_p(s)^2.$$

For a nonempty finite prime window S, define the unnormalized square energy

$$\mathcal{E}_S(s) := \sum_{p\in S} e_p(s).$$

The global prime normal energy is the monotone extended readout

$$\mathcal{E}_{\mathbb{P}}(s) := \sum_{p \in \mathsf{Prime}^{\uparrow}} e_p(s),$$

understood as the supremum of the finite-window sums.

Theorem 9413.673 (Prime square energy is mirror-even)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

each displayed prime satisfies

$$d_p(s) = -\delta \log p, \qquad e_p(s) = \delta^2 (\log p)^2.$$

Moreover

$$d_p(J(s)) = -d_p(s), \qquad e_p(J(s)) = e_p(s).$$

Proof. The first equality is the prime-coordinate calculation of Definition 9413.469. Squaring gives

$$e_p(s) = d_p(s)^2 = \delta^2 (\log p)^2.$$

The mirror sends the normal coordinate δ to $-\delta$, hence changes the sign of d_p and leaves e_p unchanged. □

Theorem 9413.674 (Finite prime square energy is strictly positive off half)**.** *If $s = 1/2 + \delta + i\tau$, $\delta \# 0$, and S is a nonempty finite prime window, then*

$$\mathcal{E}_S(s) = \delta^2 \sum_{p \in S} (\log p)^2 \# 0.$$

Proof. Summing Definition 9413.673 over S gives

$$\mathcal{E}_S(s) = \delta^2 \sum_{p \in S} (\log p)^2.$$

The window is nonempty and every displayed prime has positive logarithm, so the finite sum of squares is positive. Apartness of δ from zero gives apartness of δ^2 from zero in the located real readout, hence the displayed energy is nonzero. □

Theorem 9413.675 (Global prime square energy diverges off half)**.** *If $s = 1/2 + \delta + i\tau$ and $\delta \# 0$, then the global prime normal energy is unbounded:*

$$\mathcal{E}_{\mathbb{P}}(s) = \infty$$

as an extended finite-window supremum.

Proof. For every nonempty finite window S,

$$\mathcal{E}_S(s) = \delta^2 \sum_{p \in S} (\log p)^2.$$

There are infinitely many displayed prime dimensions, and each $(\log p)^2$ is positive. As the finite window ranges over larger and larger prime sets, the partial sums

$$\sum_{p \in S} (\log p)^2$$

are unbounded. Multiplication by the positive row δ^2 preserves unboundedness. Therefore the extended supremum $\mathcal{E}_{\mathbb{P}}(s)$ is infinite. □

Definition 9413.676 (Composite rows add no cancellation direction)**.** *The* no composite cancellation direction *row says that a composite integer contributes no independent normal coordinate beyond the prime coordinates. If*

$$n = \prod_p p^{v_p(n)},$$

then

$$\log n = \sum_p v_p(n) \log p.$$

Thus composite logarithmic rows are finite linear combinations of prime logarithmic rows and do not add a further sign in which the positive prime square energy could cancel.

Theorem 9413.677 (Prime logarithmic directions are independent)**.** *For distinct displayed primes $p_1, \ldots, p_m$, a rational relation*

$$\sum_{i=1}^{m} q_i \log p_i = 0$$

forces $q_i = 0$ for every i.

Proof. Clear denominators to obtain integers a_i with

$$\sum_{i=1}^{m} a_i \log p_i = 0.$$

Exponentiating gives

$$\prod_{i=1}^{m} p_i^{a_i} = 1.$$

By unique factorization, all exponents a_i are zero. Hence every rational coefficient q_i is zero. □

Definition 9413.678 (Zeta no-hair boundary)**.** *The* zeta no-hair boundary *row is the assertion*

$$\mathsf{ZetaNoHair}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathcal{E}_{\mathbb{P}}(s) = 0\right).$$

A weaker finite-energy version replaces the conclusion by

$$\mathcal{E}_{\mathbb{P}}(s) < \infty.$$

Both rows say that a zero boundary state cannot carry infinite normal prime energy.

Theorem 9413.679 (No-hair boundary is conditionally sufficient)**.** *Together with the located fixed-half collapse row, either zeta no-hair boundary row is sufficient for the RH fixed-half section.*

Proof. Let s carry a critical-strip zero row and write

$$s = \frac{1}{2} + \delta + i\tau.$$

Suppose $\delta \# 0$. By Definition 9413.675, the global prime normal energy is infinite. This contradicts either the zero-energy conclusion $\mathcal{E}_{\mathbb{P}}(s) = 0$ or the weaker finite-energy conclusion $\mathcal{E}_{\mathbb{P}}(s) < \infty$. Hence $\neg(\delta \# 0)$. The located fixed-half collapse row yields $\delta = 0$, and Definition 9413.313 reads this as the RH fixed-half section. □

Corollary 9413.680 (Positive energy strengthens finite defect visibility)**.** *The global square-energy route strengthens the finite defect route by replacing a visible signed defect with a monotone positive obstruction.*

Proof. The finite defect route shows that $\delta\#0$ is visible in every finite prime window by a signed row such as $D_S(s)\#0$. The square-energy row shows more: the same off-half displacement contributes positive energy on every prime dimension and yields an unbounded global extended energy. Thus the remaining RH-facing issue is no longer detection of the defect, but the no-hair boundary claim that a completed zero row cannot carry that infinite normal energy. □

9413.54 Far-End Energy Unity Socket

The prime-adic parity energy window makes the non-point reading of the far end explicit. The far end is not a last node of $\mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$, not a final value of the tower, and not a hidden register in which a visible normal defect can be cancelled. It is the cofinal socket through which all finite prime-adic parity windows are read together. Consequently an energy row vanishes at the far end only when the finite-window energies become cofinally zero. No additional term at ∞ is available.

Definition 9413.681 (Prime-adic parity far-end socket)**.** *The* prime-adic parity far-end socket *is the cofinal boundary role of the finite-window directed system*

$$A \subset \mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}, \qquad A \text{ finite and nonempty.}$$

A cofinal energy condition on this socket is a row of the form

$$\prod_{\varepsilon>0} \sum_{A_0:\mathsf{PAdicParityWindow}_{\neq\varnothing}} \prod_{A\supseteq A_0} \left(E_A^{\mathrm{pad}}(s) < \varepsilon\right),$$

where A ranges over finite nonempty prime-adic parity windows. The symbol ∞ or $\mathsf{T}_{\mathrm{far}}$ names this cofinal compatibility role; it does not name an element A_∞.

Definition 9413.682 (Far-end energy unity)**.** *For a zeta readout s, define the* far-end energy unity *row by*

$$\mathsf{FarEndEnergyUnity}_{\zeta}(s) := \prod_{\varepsilon>0} \sum_{A_0:\mathsf{PAdicParityWindow}_{\neq\varnothing}} \prod_{A\supseteq A_0} \left(E_A^{\mathrm{pad}}(s) < \varepsilon\right).$$

It says that the normalized second-order prime-adic parity energy tends to zero cofinally through finite windows. It is not a statement that there exists an infinite window where a correction term is evaluated.

Definition 9413.683 (No hidden energy correction at the far end)**.** *The* no hidden energy correction *discipline is the refusal of an undisplayed decomposition*

$$E_\infty^{\mathrm{pad}}(s) = E_A^{\mathrm{pad}}(s) + C_\infty(s)$$

with $C_\infty(s)$ treated as a boundary term able to cancel $E_A^{\mathrm{pad}}(s)$. In the BEDC reading, the far end contributes only the cofinal quantifier pattern of Definition 9413.682. Any actual correction row must appear as positive carrier data, as in the archimedean completion row of Definition 9413.140; it cannot be borrowed from the apophatic socket.

Theorem 9413.684 (Far-end energy unity is fixed-half energy)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the following rows are equivalent:

$$\mathsf{FarEndEnergyUnity}_{\zeta}(s), \qquad \delta^2 = 0.$$

Consequently, in the located real readout, far-end energy unity is equivalent to

$$\Re(s) = \frac{1}{2}.$$

Proof. By Definition 9413.648, every finite nonempty prime-adic parity window satisfies

$$E_A^{\mathrm{pad}}(s) = \delta^2.$$

If $\delta^2 = 0$, then $E_A^{\mathrm{pad}}(s) = 0$ for every finite window, so the cofinal condition of Definition 9413.682 holds.

Conversely, suppose $\mathsf{FarEndEnergyUnity}_{\zeta}(s)$ holds. If $\delta^2 \# 0$, choose ε below the positive row δ^2 in the located real readout. The cofinal condition would then require all sufficiently large finite windows to satisfy

$$E_A^{\mathrm{pad}}(s) < \varepsilon.$$

But every such window has

$$E_A^{\mathrm{pad}}(s) = \delta^2,$$

contradicting the choice of ε. Hence $\delta^2 = 0$. Located collapse gives $\delta = 0$, and Definition 9413.313 reads this as $\Re(s) = 1/2$. □

Definition 9413.685 (Unnormalized parity far-end energy)**.** *For a prime-adic parity window A, define the unnormalized parity energy by*

$$\mathcal{E}_A^{\mathrm{pad}}(s) := \sum_{(p,j)\in A} \left(\log \left|U_{p,j}^{\mathrm{pad}}(s)\right|\right)^2.$$

The corresponding far-end readout is the extended supremum over finite windows:

$$\mathcal{E}_{\mathrm{far}}^{\mathrm{pad}}(s) := \sup_A \mathcal{E}_A^{\mathrm{pad}}(s).$$

Theorem 9413.686 (Unnormalized parity energy diverges off half)**.** *If $s = 1/2 + \delta + i\tau$ and $\delta \# 0$, then*

$$\mathcal{E}_{\mathrm{far}}^{\mathrm{pad}}(s) = \infty.$$

Proof. For each node (p, j),

$$U_{p,j}^{\mathrm{pad}}(s) = p^{-2^j\delta} e^{-i2^j\tau\log p},$$

so

$$\log \left|U_{p,j}^{\mathrm{pad}}(s)\right| = -2^j\delta\log p.$$

Therefore

$$\mathcal{E}_A^{\mathrm{pad}}(s) = \delta^2 \sum_{(p,j)\in A} \left(2^j \log p\right)^2.$$

The directed carrier $\mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ has cofinally unbounded finite node windows, and every displayed summand $\left(2^j \log p\right)^2$ is positive. If $\delta \# 0$, multiplication by δ^2 preserves unboundedness in the extended readout. Hence the far-end supremum is infinite. □

Definition 9413.687 (Zero far-end energy unity target)**.** *The* zero far-end energy unity target *is*

$$\mathsf{ZeroFarEndEnergyUnity}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \Big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{FarEndEnergyUnity}_\zeta(s)\Big).$$

It says that a completed critical-strip zero is a far-end energy-unified boundary state.

Theorem 9413.688 (Zero far-end energy unity is conditionally sufficient)**.** *If* $\mathsf{ZeroFarEndEnergyUnity}_\zeta$ *holds, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. The zero row and critical-strip row of z give

$$\mathsf{FarEndEnergyUnity}_\zeta(s)$$

by Definition 9413.687. By Definition 9413.684, this implies $\Re(s) = 1/2$. Therefore z has the fixed-half lift required by Definition 9413.844, and the section equivalence of Definition 9413.199 gives the RH fixed-half section. □

Corollary 9413.689 (Energy unity is the socket form of no-hair)**.** *The zero far-end energy unity target is the cofinal finite-window form of the zeta no-hair boundary.*

Proof. Definition 9413.678 states that a critical-strip zero cannot carry global normal prime energy. The present socket formulation states the same obstruction through normalized prime-adic parity energies: the finite-window energy family has the constant value δ^2, so cofinal vanishing is possible exactly on the fixed-half surface. The unnormalized form of Definition 9413.686 recovers the infinite-energy reading of Definition 9413.675 with binary depth included. □

Corollary 9413.690 (No point at infinity can absorb parity energy)**.** *An off-half row cannot be made far-end energy-unified by adding a correction at* ∞*.*

Proof. By Definition 9413.681, the far end is a cofinal socket for finite windows, not an element. By Definition 9413.684, the finite-window energy family is cofinally zero exactly when $\delta^2 = 0$. Thus an off-half row has stable positive normalized energy in every finite window and unbounded unnormalized energy by Definition 9413.686. The prohibited hidden term of Definition 9413.683 would therefore be precisely a point-reification of the apophatic far end, which is excluded by Definition 9413.141. □

9413.55 Finite-to-Infinite Phase-Energy Tower

The prime phase-radial readback separates two finite-to-infinite phenomena. Finite prime windows display simultaneous phase rotation: each displayed prime contributes a unit phase with frequency $\log p$ on the critical surface. The infinite passage relevant to the RH-facing obstruction is not the signed phase row, because signed phase can cancel in a completed projection. The stable row is the positive second-order normal energy. It is monotone on finite prime windows and becomes unbounded whenever the normal displacement is nonzero.

Definition 9413.691 (Finite phase-energy window)**.** *For a nonempty finite prime window* S*, the* finite phase-energy window *at*

$$s = \frac{1}{2} + \delta + i\tau$$

is the packet

$$\mathsf{PhaseEnergy}_S(s) := (\Phi_S(\tau), \mathcal{E}_S(s)),$$

where $\Phi_S(\tau)$ is the finite phase profile of Definition 9413.296 and

$$\mathcal{E}_S(s) = \sum_{p \in S} \left(\log |p^{1/2-s}|\right)^2$$

is the finite prime normal square energy of Definition 9413.672.

Theorem 9413.692 (Finite prime phase windows have no nonzero common period)**.** *Let S be a finite prime window containing two distinct primes. There is no nonzero real time T such that*

$$e^{-iT \log p} = 1$$

for every $p \in S$.

Proof. Suppose such a nonzero T exists. For any two distinct primes $p, q \in S$, there are integers m_p, m_q with

$$T \log p = 2\pi m_p, \qquad T \log q = 2\pi m_q.$$

Since $T \# 0$ and both logarithms are positive, neither integer can be zero. Hence

$$\frac{\log p}{\log q} = \frac{m_p}{m_q} \in \mathbb{Q}.$$

This is a nontrivial rational relation between two prime logarithmic directions, contradicting Definition 9413.677. □

Corollary 9413.693 (Finite phase closure is quasi-periodic data)**.** *Finite prime phase windows may exhibit approximate recurrence and projected cancellation, but they do not carry an exact common period once at least two prime frequencies are displayed.*

Proof. Approximate recurrence belongs to the finite torus face $\mathbb{T}_S$ of Definition 9413.293. Exact common periodicity would require a nonzero T satisfying the condition excluded by Definition 9413.692. Thus finite phase cancellation is not an exact periodic clock for the prime tower. □

Theorem 9413.694 (Finite normal energy is monotone in the window)**.** *If $S \subseteq T$ are nonempty finite prime windows, then*

$$\mathcal{E}_S(s) \le \mathcal{E}_T(s).$$

Moreover, if $s = 1/2 + \delta + i\tau$ and $\delta \# 0$, then every nonempty finite window has positive normal energy, and enlarging the window adds only nonnegative terms.

Proof. By definition,

$$\mathcal{E}_T(s) = \sum_{p \in T} \left(\log |p^{1/2-s}|\right)^2.$$

All summands are squares, hence nonnegative. Splitting the sum over S and $T \setminus S$ gives

$$\mathcal{E}_T(s) = \mathcal{E}_S(s) + \sum_{p \in T \setminus S} \left(\log |p^{1/2-s}|\right)^2,$$

which proves monotonicity. If $s = 1/2 + \delta + i\tau$, then Definition 9413.674 gives

$$\mathcal{E}_S(s) = \delta^2 \sum_{p \in S} (\log p)^2 \# 0$$

for every nonempty finite window when $\delta \# 0$. □

Theorem 9413.695 (Finite-to-infinite energy divergence)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

if $\delta \# 0$*, then the finite-window normal energies form an unbounded cofinal tower:*

$$\sup_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} \mathcal{E}_S(s) = \infty.$$

Proof. For every nonempty finite window S,

$$\mathcal{E}_S(s) = \delta^2 \sum_{p\in S} (\log p)^2$$

by Definition 9413.673. The finite sums

$$\sum_{p\in S} (\log p)^2$$

are unbounded as S ranges cofinally over finite prime windows, since there are infinitely many displayed prime rows and each displayed square is positive. Because $\delta \# 0$, multiplication by δ^2 preserves unboundedness. This is the same extended supremum readout as Definition 9413.675. □

Definition 9413.696 (Finite normal energy boundary)**.** *The* finite normal energy boundary *row for completed critical-strip zeros is*

$$\mathsf{FiniteNormalEnergyBoundary}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \sup_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} \mathcal{E}_S(s) < \infty \right).$$

It says that a completed zero boundary state cannot carry unbounded normal prime energy along the finite-window tower.

Theorem 9413.697 (Finite normal energy boundary is conditionally sufficient)**.** *If* $\mathsf{FiniteNormalEnergyBoundary}_\zeta$ *holds, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

The zero and critical-strip rows of z give

$$\sup_{S:\mathsf{FinitePrimeWindow}_{\neq\varnothing}} \mathcal{E}_S(s) < \infty$$

by Definition 9413.696. If $\delta \# 0$, then Definition 9413.695 gives the opposite extended readout, namely an infinite supremum. Hence $\neg(\delta \# 0)$. The located fixed-half collapse row gives $\delta = 0$, and Definition 9413.313 reads this as $\Re(s) = 1/2$. Varying z supplies the fixed-half section. □

Corollary 9413.698 (Finite-to-infinite tracking identifies the proof load)**.** *The finite-to-infinite tower separates detection from zero relevance. Off-half displacement is detected by positive finite-window energies and becomes unbounded cofinally; the remaining RH-facing claim is that completed zeta zero rows satisfy the finite normal energy boundary.*

Proof. Definition 9413.694 shows that the finite-window energy is positive off the fixed-half surface, and Definition 9413.695 shows that the cofinal tower is unbounded. These are readout facts, not zero-exclusion facts. The zero-exclusion content is exactly Definition 9413.696, equivalently the finite-energy form of the zeta no-hair boundary in Definition 9413.678. □

Corollary 9413.699 (Finite models mark the zero-relevance gap)**.** *The finite-to-infinite energy tower has the same proof-role shape as the quartic finite spectral carrier: visible finite data become zero-relevant only after a boundary or dominance ledger identifies which visible row controls the zero law.*

Proof. The quartic carrier does not infer endpoint zeros from the mere presence of spectral branches; it uses normalization, real projection, dominance, and fixed-window separation as in Definition 9413.485. In the zeta tower, the visible row is positive normal energy on finite prime windows. The missing zero-relevance ledger is the boundary assertion that a completed zero cannot carry unbounded cofinal normal energy, as stated in Definition 9413.696. □

Concretized in §9002: Finite energy tower packets realize finite prime-window phase rows, positive normal-energy rows, monotone window extension, Real/Complex readback, and nonescape ledgers as NameCert data.

9413.56 Infinite Prime Torus and Far-End Compatibility

The circular image must be calibrated before it is used for the zeta carrier. A circle may arise from a genuine periodic quotient, but it may also arise from a far-end closure of a line. The zeta prime carrier does not have a nonzero time at which all prime phases return to their initial positions. Its infinite closure is instead an interface of compatible finite prime-window readings. The far end is therefore not a period point; it is the cofinal compatibility socket of the infinite prime torus.

Definition 9413.700 (Periodic circle and far-end circle)**.** *The* periodic circle *is the quotient readout*

$$\mathsf{Real}^{\uparrow}/2\pi\mathsf{Int}^{\uparrow} \cong S^1,$$

where the identification is a genuine period row

$$\theta \sim \theta + 2\pi.$$

The far-end circle *is the closure readout*

$$\mathsf{Real}^{\uparrow} \cup \{\infty\} \cong S^1,$$

where ∞ names the closure point of the unbounded line rather than a time period of the flow.

Definition 9413.701 (Finite prime torus projection system)**.** *For a nonempty finite prime window S, let*

$$\mathbb{T}_S := \prod_{p \in S} S^1.$$

If $S \subseteq T$, the projection

$$\pi_{T,S} : \mathbb{T}_T \to \mathbb{T}_S$$

forgets the coordinates in $T \setminus S$. The finite prime tori and forgetful maps form the finite-window projection system of the zeta prime phase carrier.

Definition 9413.702 (Infinite prime torus)**.** *The* infinite prime torus *is the compatible finite-window carrier*

$$\mathbb{T}_{\mathsf{Prime}^\uparrow} := \varprojlim_{S \Subset \mathsf{Prime}^\uparrow} \mathbb{T}_S.$$

Equivalently, it is the cofinal product

$$\mathbb{T}_{\mathsf{Prime}^\uparrow} = \prod_{p \in \mathsf{Prime}^\uparrow} S^1$$

read only through its finite projections

$$\pi_S : \mathbb{T}_{\mathsf{Prime}^\uparrow} \to \mathbb{T}_S.$$

Thus a finite prime window is not an independent small object; it is a finite readout of the infinite prime torus.

Theorem 9413.703 (Prime phase flow lands in the inverse-limit torus)**.** *The phase profile*

$$\Phi_{\mathsf{Prime}^\uparrow}(\tau) = \left(e^{-i\tau \log p}\right)_{p \in \mathsf{Prime}^\uparrow}$$

is a point of the infinite prime torus, and its finite projection is

$$\pi_S\big(\Phi_{\mathsf{Prime}^\uparrow}(\tau)\big) = \Phi_S(\tau).$$

Proof. For each prime p, the coordinate $e^{-i\tau \log p}$ lies in S^1. Thus the family lies in the product $\prod_{p \in \mathsf{Prime}^\uparrow} S^1$. For any inclusion $S \subseteq T$, forgetting the coordinates outside S sends the finite profile on T to the finite profile on S. Hence the finite profiles are compatible under the projection maps of Definition 9413.701, and the cofinal family is an element of the inverse-limit torus. □

Theorem 9413.704 (The infinite prime torus has no common period point)**.** *There is no nonzero real time T such that*

$$\Phi_{\mathsf{Prime}^\uparrow}(\tau + T) = \Phi_{\mathsf{Prime}^\uparrow}(\tau)$$

for every time τ.

Proof. If such a T existed, then every finite prime window containing two distinct primes would have the same nonzero common period. This is excluded by Definition 9413.692. Therefore the infinite prime torus is not a periodic circle with a single return time for all prime coordinates. □

Definition 9413.705 (Prime far-end compatibility)**.** *The* prime far-end compatibility *row is the assertion that an infinite prime-phase state is specified by a compatible family*

$$(x_S)_{S \Subset \mathsf{Prime}^\uparrow}, \qquad x_S \in \mathbb{T}_S,$$

such that for every $S \subseteq T$,

$$\pi_{T,S}(x_T) = x_S.$$

The far end of the prime tower is this compatibility interface. It is not an additional prime, a final finite window, or a hidden coordinate.

Theorem 9413.706 (Far-end compatibility is not phase periodicity)**.** *For the zeta prime phase carrier, far-end closure means compatibility of all finite torus projections, not exact return of all prime phases to a common time origin.*

Proof. Compatibility is the inverse-limit condition of Definition 9413.705. Exact common return would supply a nonzero period for every finite window, hence a nonzero common period for the infinite prime torus. The latter is excluded by Definition 9413.704. Thus the far-end closure role is closer to far-end completion than to a periodic quotient. □

Theorem 9413.707 (Stable normal defect is the common finite-window readout)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

every nonempty finite prime window satisfies

$$\overline{D}_S(s) = -\delta.$$

Consequently the finite windows need not agree in phase, but they do agree on the normalized normal defect.

Proof. The normalized defect calculation is exactly Definition 9413.167. It uses the radial row

$$\log |p^{1/2-s}| = -\delta \log p$$

and divides by the finite logarithmic weight $\sum_{p\in S} \log p$. The phase coordinates $e^{-i\tau \log p}$ need not be periodic across the window. The stable cofinal datum is instead the common normalized normal defect $-\delta$. □

Corollary 9413.708 (Off-half rows are visible in every finite torus readout)**.** *If $\delta\#0$, then every nonempty finite prime window reads a nonzero normalized normal defect and positive normalized second-order energy.*

Proof. By Definition 9413.707, $\overline{D}_S(s) = -\delta$ for every nonempty finite window. Hence $\delta\#0$ gives a nonzero normalized defect in every finite projection. Squaring gives the positive normalized energy δ^2, as in Definition 9413.648 for the parity-refined windows and in Definition 9413.468 for prime-window energy. □

Corollary 9413.709 (The zeta far end cannot be a hidden defect absorber)**.** *The infinite prime torus far end cannot be used to absorb a nonzero normal defect that is visible in every finite window.*

Proof. The far end is the compatibility interface of Definition 9413.705, not an element with an additional coordinate. If $\delta\#0$, then Definition 9413.708 shows that every finite projection reads the same nonzero normalized normal defect, while Definition 9413.695 shows that the unnormalized positive energy is unbounded cofinally. Absorbing this row at the far end would reify the compatibility socket as a hidden correction point, contrary to Definition 9413.683 and Definition 9413.141. □

Corollary 9413.710 (RH as unit infinite-torus zero boundary)**.** *The RH-facing geometric form is that completed zeta zero rows in the critical strip are supported on the unit infinite prime torus, not on a radially deformed prime-torus layer.*

Proof. On the fixed-half surface, the prime phase-radial readback lies in $\mathbb{T}_{\mathsf{Prime}^\uparrow}$ by Definition 9413.703. Off the fixed-half surface, the same finite phase profile is multiplied by the radial row $(p^{-\delta})_{p\in\mathsf{Prime}^\uparrow}$. The latter is detected by every finite normalized defect readout and by the cofinal energy tower of section 9413.55. To prove the RH section is therefore to prove the zero-relevance boundary: completed zero rows cannot live on the radially deformed layer. □

9413.57 Far-End Identity Boundary

The non-point far end can be sharpened once the circular analogy is read correctly. A circular closure is not a reservoir that absorbs an unwanted term. It is an identity action on the displayed boundary coordinate. The equality

$$e^{i(\theta+2\pi)} = e^{i\theta}$$

does not add a compensating correction; it says that the full turn acts as the identity on phase. If a point has radius $r\#1$, the same full turn returns the same radius. Closure does not repair the radius.

Definition 9413.711 (Circular identity closure)**.** *A* circular identity closure *is a closure row whose monodromy acts as the identity on the displayed closed coordinate:*

$$\mathsf{Close}_{S^1}(\theta) = \theta$$

in the quotient readout, or equivalently

$$e^{i(\theta+2\pi)} = e^{i\theta}.$$

It is not a row of the form

$$x \longmapsto x + C_\infty$$

for an undisplayed correction term C_∞.

Definition 9413.712 (Far-end identity)**.** *The* far-end identity *row for the zeta prime-torus carrier is*

$$\mathsf{FarEndIdentity}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{FarEndClosed}_\zeta(s) \to \prod_{A:\mathsf{PAdicParityWindow}_{\neq\varnothing}} (E_A^{\mathrm{pad}}(s) = E_A^{\mathrm{pad}}(\mathsf{Close}_{\mathrm{far}}(s))) \right).$$

It says that far-end closure preserves the finite-window normal energy readouts. The far end may identify compatible phase data, but it does not introduce a normal correction row.

Definition 9413.713 (Far-end identity boundary for zeros)**.** *The* far-end identity boundary *for completed zeta zeros is*

$$\mathsf{FarEndIdentityBoundary}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \Big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathsf{FarEndEnergyUnity}_\zeta(s) \Big).$$

It asserts that a completed zero row reaches the far end through identity closure of the finite energy readouts, not through a hidden boundary reservoir. This is the zero-facing form of Definition 9413.682.

Theorem 9413.714 (Circular identity does not repair radial displacement)**.** *For $r > 0$ and $r\#1$, circular identity closure preserves the off-unit radius:*

$$re^{i(\theta+2\pi)} = re^{i\theta}.$$

Thus circular closure can identify phase after a full turn, but it does not turn an off-unit radius into a unit radius.

Proof. The identity row gives

$$e^{i(\theta+2\pi)} = e^{i\theta}.$$

Multiplying by the displayed positive radius r gives

$$re^{i(\theta+2\pi)} = re^{i\theta}.$$

No term in this equality changes r. Therefore the apartness row $r\#1$ is preserved by the phase closure. □

Theorem 9413.715 (Far-end identity preserves normal energy)**.** *Assume* $\mathsf{FarEndIdentity}_\zeta$*. If*

$$s = \frac{1}{2} + \delta + i\tau$$

and $\delta^2\#0$*, then no far-end closure of* s *can have all finite prime-adic parity energies equal to zero.*

Proof. For every nonempty prime-adic parity window A, Definition 9413.648 gives

$$E_A^{\mathrm{pad}}(s) = \delta^2.$$

If $\delta^2\#0$, this finite-window energy is nonzero. The far-end identity row preserves each finite energy readout under the closure operation. Hence the closed far-end readout still has

$$E_A^{\mathrm{pad}} = \delta^2\#0$$

for every finite window. It therefore cannot satisfy the cofinal zero-energy condition of Definition 9413.682. □

Corollary 9413.716 (Far-end identity excludes hidden energy absorption)**.** *Far-end identity is the geometric form of no hidden energy correction: a stable positive finite-window energy cannot be removed by passing to the compatibility far end.*

Proof. The hidden correction pattern of Definition 9413.683 would replace a displayed finite-window energy by an altered boundary value. Far-end identity instead preserves the displayed finite-window energy under closure. Therefore the far end acts like the identity closure of Definition 9413.711, not like a reservoir for the nonzero row. □

Theorem 9413.717 (Far-end identity boundary is conditionally sufficient)**.** *If* $\mathsf{FarEndIdentityBoundary}_\zeta$ *holds, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. The zero row and critical-strip row of z give

$$\mathsf{FarEndEnergyUnity}_\zeta(s)$$

by Definition 9413.713. By Definition 9413.684, this gives $\delta^2 = 0$, hence $\delta = 0$ in the located real readout. Thus

$$\Re(s) = \frac{1}{2}.$$

The section equivalence of Definition 9413.199 supplies the RH fixed-half section. □

Corollary 9413.718 (The proof load is identity closure for zero rows)**.** *The RH-facing proof load may be stated as the claim that completed zeta zero rows use the far end as identity closure of compatible finite readouts, not as a hidden correction mechanism.*

Proof. Definition 9413.709 shows that the infinite prime-torus far end is a compatibility interface. Definition 9413.715 shows that identity closure cannot remove a positive normal energy row. Therefore the remaining zero-specific assertion is that zeta zero rows indeed satisfy the identity-boundary condition of Definition 9413.713. Once that row is supplied, the preceding theorem gives the fixed-half section. □

9413.58 Prime Spiral Projection Kernel

The prime-torus picture should not be read as a common-period picture. The zeta zero row is not a moment at which all prime frequencies synchronize. It is a low-dimensional projection row: an infinite prime-indexed phase-radial state is read through the completed zeta projection, and that projection may vanish. The information hidden by the projection is not thereby absent from the carrier. The RH-facing claim is that no zero projection may hide a nonzero normal height.

Definition 9413.719 (Prime spiral carrier)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the prime spiral carrier *is the prime-indexed phase-radial row*

$$\mathcal{U}_{\mathsf{Prime}^\uparrow}(s) := \left(p^{1/2-s}\right)_{p\in\mathsf{Prime}^\uparrow} = \left(p^{-\delta}e^{-i\tau\log p}\right)_{p\in\mathsf{Prime}^\uparrow}.$$

Its unit phase part is $\Phi_{\mathsf{Prime}^\uparrow}(\tau)$*, and its normal radial part is the family* $(p^{-\delta})_{p\in\mathsf{Prime}^\uparrow}$*.*

Definition 9413.720 (Completed zeta projection)**.** *The* completed zeta projection *is the schematic projection ledger*

$$\Pi_\zeta : \mathcal{U}_{\mathsf{Prime}^\uparrow}(s) \longmapsto \xi(s),$$

where

$$\xi(s) = \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s)$$

is the completed zeta readout of Definition 9413.140. The notation records a paper-side projection relation between the high-dimensional prime carrier and the completed complex value; it is not a claim that the complex value is the whole carrier.

Definition 9413.721 (Projection zero and ontic balance)**.** *For a critical-strip row* s*, the* projection-zero *row is*

$$\mathsf{ProjectionZero}_\zeta(s) := \Pi_\zeta(\mathcal{U}_{\mathsf{Prime}^\uparrow}(s)) = 0.$$

The ontic-balance *row is*

$$\mathsf{OnticBalance}_\zeta(s) := \delta^2 = 0.$$

Projection zero is a completed complex readout. Ontic balance is a normal-energy readout of the high-dimensional prime spiral carrier.

Theorem 9413.722 (Zero projection is not coordinate synchronization)**.** *A completed zeta zero row states that*

$$\Pi_\zeta(\mathcal{U}_{\mathsf{Prime}^\uparrow}(s)) = 0.$$

It does not state that all prime phase coordinates have a common period or have returned to a common phase.

Proof. The absence of a nonzero common phase period is Definition 9413.704. The zero row belongs instead to the completed projection of Definition 9413.720. By Definition 9413.295, zerohood is a global cancellation in the completed readout, not a local vanishing or synchronization of prime coordinates. □

Definition 9413.723 (Projection kernel ambiguity)**.** *A* projection kernel ambiguity *for the zeta projection is a hypothetical pair of rows*

$$\Pi_\zeta(\mathcal{U}_{\mathsf{Prime}^\uparrow}(s)) = 0, \qquad \delta^2 \# 0.$$

It says that the completed complex projection has vanished while the high-dimensional prime spiral carrier still has nonzero normal height.

Theorem 9413.724 (Off-half projection zeros carry infinite normal energy)**.** *If a projection kernel ambiguity holds for*

$$s = \frac{1}{2} + \delta + i\tau,$$

then the high-dimensional prime spiral carrier carries infinite normal energy:

$$\mathcal{E}_{\mathbb{P}}(s) = \infty.$$

Proof. The ambiguity includes $\delta^2 \# 0$, hence $\delta \# 0$ in the located real readout. For every displayed prime,

$$\log |p^{1/2-s}| = -\delta \log p.$$

Therefore the finite-window square energy is

$$\mathcal{E}_S(s) = \delta^2 \sum_{p \in S} (\log p)^2.$$

As finite prime windows range cofinally, this energy is unbounded by Definition 9413.695, or equivalently by Definition 9413.675. □

Definition 9413.725 (No hidden normal height)**.** *The* no hidden normal height *row for the completed zeta projection is*

$$\mathsf{NoHiddenNormalHeight}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \mathcal{E}_{\mathbb{P}}(s) < \infty)\,.$$

It says that a completed zero projection cannot arise from a high-dimensional prime spiral state with infinite normal energy.

Theorem 9413.726 (No hidden normal height is conditionally sufficient)**.** *If* $\mathsf{NoHiddenNormalHeight}_\zeta$ *holds, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

The zero and critical-strip rows give

$$\mathcal{E}_{\mathbb{P}}(s) < \infty$$

by Definition 9413.725. If $\delta \# 0$, then Definition 9413.675 gives $\mathcal{E}_{\mathbb{P}}(s) = \infty$, a contradiction. Hence $\neg(\delta \# 0)$. Located fixed-half collapse gives $\delta = 0$, and Definition 9413.313 reads this as $\Re(s) = 1/2$. Varying z supplies the RH fixed-half section. □

Corollary 9413.727 (Projection zero needs a zero-relevance boundary)**.** *The RH-facing target is not to prevent phase cancellation in the completed projection. It is to prove that the zero projection cannot hide nonzero normal height in the prime spiral carrier.*

Proof. Phase cancellation is exactly what a completed zero projection records. By Definition 9413.722, this does not imply common prime-period synchronization. By Definition 9413.724, an off-half zero projection would carry infinite normal energy. Thus the missing row is the zero-relevance boundary Definition 9413.725, equivalently the finite-energy form of the no-hair boundary in Definition 9413.678. □

Corollary 9413.728 (Quartic dominance is the finite analogue)**.** *The quartic spectral carrier illustrates the same proof-role distinction: a projected zero law requires a zero-relevant dominant boundary, not merely a visible projected cancellation.*

Proof. The fixed quartic chain identifies endpoint zeros only after the positive-axis dominant sheet and fixed-window separation rows have been displayed, as recorded in Definition 9413.485. In the zeta carrier, the analogous issue is whether a completed zero projection can come from a prime spiral state with hidden normal height. The no-hidden-normal-height row is the infinite prime-tower analogue of that zero-relevance boundary. □

9413.59 Prime Resonance Chord and No-Hair Boundary

The prime spiral projection is better read as a resonance problem than as a common-period problem. Prime logarithms supply distinct frequencies. At a given time, a finite family of prime-adic parity nodes can interfere in the completed complex projection. A zero row is therefore analogous to a projected resonance silence: the first-order complex sum can vanish. The RH-facing obstruction is that this silence must not be produced by a resonance family carrying nonzero normal energy.

Definition 9413.729 (Prime resonance family)**.** *A* prime resonance family *is a nonempty finite window*

$$A \subset \mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$$

together with complex weights

$$a_{p,j} \in \mathsf{Complex}^{\uparrow} \qquad ((p,j) \in A).$$

The node frequency is

$$\omega_{p,j} := 2^j \log p,$$

and the node channel is

$$U^{\mathrm{pad}}_{p,j}(s) = p^{-2^j\delta} e^{-i2^j\tau \log p}$$

when $s = 1/2 + \delta + i\tau$.

Definition 9413.730 (Finite resonance chord)**.** *For a prime resonance family* (A, a)*, the* finite resonance chord *is*

$$C_A(s) := \sum_{(p,j)\in A} a_{p,j} U^{\mathrm{pad}}_{p,j}(s).$$

The row

$$C_A(s) = 0$$

is a finite projected silence row: the displayed complex chord closes in the projection, while the individual node channels remain nonzero.

Definition 9413.731 (Hamiltonian resonance island)**.** *For a prime resonance family* (A, a)*, the* Hamiltonian resonance island *is the finite diagonal carrier*

$$\mathcal{H}_A := \operatorname{span}\{|p, j\rangle : (p, j) \in A\}$$

with Hamiltonian

$$H_A|p, j\rangle = h_{p,j}|p, j\rangle, \qquad h_{p,j} := 2^j \log p.$$

For

$$|\psi_A\rangle := \sum_{(p,j)\in A} a_{p,j}|p, j\rangle, \qquad \langle \mathbf{1}_A|p, j\rangle := 1,$$

the finite resonance chord is equivalently

$$C_A(s) = \langle \mathbf{1}_A|e^{-(s-1/2)H_A}|\psi_A\rangle.$$

Thus a projected silence row is a finite Hamiltonian interference row seen through the displayed evaluation covector.

Theorem 9413.732 (The critical line is the unitary resonance surface)**.** *For a Hamiltonian resonance island and*

$$s = \frac{1}{2} + \delta + i\tau,$$

the finite evolution decomposes as

$$e^{-(s-1/2)H_A} = e^{-\delta H_A} e^{-i\tau H_A}.$$

It is unitary exactly on the fixed-half surface $\delta = 0$*.*

Proof. The Hamiltonian H_A is diagonal with real eigenvalues $h_{p,j}$. Since $s - 1/2 = \delta + i\tau$, exponentiating each displayed eigenrow gives the stated factorization. If $\delta = 0$, the evolution is $e^{-i\tau H_A}$, hence every node contributes only a phase. If $\delta\#0$, the factor $e^{-\delta H_A}$ is a real amplitude-bias row, with eigenvalue $e^{-\delta h_{p,j}} = p^{-2^j\delta}$ on node (p, j). This is exactly the nonunitary radial detuning read in Definition 9413.734. □

Theorem 9413.733 (Chord silence is a Pythagorean interference identity)**.** *Let*

$$v_{p,j} := a_{p,j} U^{\mathrm{pad}}_{p,j}(s).$$

If $C_A(s) = 0$*, then*

$$0 = \sum_{(p,j)\in A} |v_{p,j}|^2 + 2 \sum_{(p,j)<(q,k)} \operatorname{Re}\left(v_{p,j}\overline{v_{q,k}}\right),$$

where the second sum ranges over unordered distinct node pairs in A*.*

Proof. The square of the projected chord is

$$\left|\sum_{(p,j)\in A} v_{p,j}\right|^2.$$

Expanding the Hermitian square gives

$$\left|\sum_{(p,j)\in A} v_{p,j}\right|^2 = \sum_{(p,j)\in A} |v_{p,j}|^2 + 2 \sum_{(p,j)<(q,k)} \operatorname{Re}\left(v_{p,j}\overline{v_{q,k}}\right).$$

When $C_A(s) = \sum v_{p,j} = 0$, the left side is zero, giving the stated identity. Thus projected silence is an exact balance between positive node magnitudes and phase interference terms. □

Definition 9413.734 (Detuned resonance energy)**.** *The* normal detuning *of a node is*

$$d_{p,j}(s) := \log |U_{p,j}^{\mathrm{pad}}(s)| = -2^j \delta \log p.$$

The unnormalized normal energy of a resonance family is

$$\mathcal{E}_A^{\mathrm{res}}(s) := \sum_{(p,j)\in A} d_{p,j}(s)^2.$$

The normalized normal energy is

$$E_A^{\mathrm{res}}(s) := \left(\frac{\sum_{(p,j)\in A} d_{p,j}(s)}{\sum_{(p,j)\in A} 2^j \log p} \right)^2.$$

Theorem 9413.735 (Detuned resonance reads the normal coordinate)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

every prime resonance family satisfies

$$E_A^{\mathrm{res}}(s) = \delta^2.$$

If $\delta \# 0$, then every nonempty resonance family has positive normalized normal energy.

Proof. By Definition 9413.734,

$$\sum_{(p,j)\in A} d_{p,j}(s) = -\delta \sum_{(p,j)\in A} 2^j \log p.$$

The denominator is the positive prime-adic parity weight of Definition 9413.647. Dividing gives $-\delta$, and squaring gives $E_A^{\mathrm{res}}(s) = \delta^2$. Hence apartness of δ from zero gives positive normalized normal energy for every nonempty family. □

Definition 9413.736 (Centered Hamiltonian variance energy)**.** *Assume the coefficient vector of a prime resonance family is nonzero. Set*

$$P_{p,j} := \frac{|a_{p,j}|^2}{\sum_{(q,k)\in A} |a_{q,k}|^2}.$$

The weighted Hamiltonian mean and variance are

$$\langle H_A \rangle_\psi := \sum_{(p,j)\in A} P_{p,j} h_{p,j},$$

$$\mathrm{Var}_\psi(H_A) := \sum_{(p,j)\in A} P_{p,j} \left(h_{p,j} - \langle H_A \rangle_\psi \right)^2.$$

The centered Hamiltonian detuning energy is

$$\mathcal{Q}_A^\psi(s) := \delta^2 \, \mathrm{Var}_\psi(H_A).$$

Theorem 9413.737 (Prime Hamiltonian levels are separated)**.** *If $(p,j) \neq (q,k)$, then*

$$2^j \log p \neq 2^k \log q.$$

Proof. If $p = q$, positivity of $\log p$ gives $2^j = 2^k$, hence $j = k$. If $p \neq q$, equality would give the nontrivial rational linear relation

$$2^j \log p - 2^k \log q = 0,$$

contradicting Definition 9413.677. Thus distinct prime-adic parity nodes have distinct Hamiltonian levels. □

Theorem 9413.738 (Centered nonunitary cost is variance energy)**.** *For a nonzero coefficient vector, define*

$$N_A^\circ(\delta, \psi) := \sum_{(p,j)\in A} P_{p,j} e^{-2\delta(h_{p,j} - \langle H_A \rangle_\psi)}.$$

Then

$$N_A^\circ(0, \psi) = 1, \qquad \frac{d}{d\delta} N_A^\circ(0, \psi) = 0,$$

and

$$\frac{1}{2} \frac{d^2}{d\delta^2} N_A^\circ(0, \psi) = 2 \operatorname{Var}_\psi(H_A).$$

Consequently the first nonzero centered nonunitary cost is the positive quadratic variance row.

Proof. At $\delta = 0$, all exponential factors are 1, and the weights $P_{p,j}$ sum to 1. The first derivative is

$$-2 \sum_{(p,j)\in A} P_{p,j} \left(h_{p,j} - \langle H_A \rangle_\psi\right),$$

which is zero by the definition of the weighted mean. The second derivative at zero is

$$4 \sum_{(p,j)\in A} P_{p,j} \left(h_{p,j} - \langle H_A \rangle_\psi\right)^2 = 4 \operatorname{Var}_\psi(H_A).$$

Taking one half gives the stated coefficient. Thus centering removes the signed first-order bias, while the second-order Hamiltonian spread remains. □

Corollary 9413.739 (Nontrivial resonance support has positive variance)**.** *If the nonzero coefficients of a prime resonance family include two distinct nodes, then*

$$\operatorname{Var}_\psi(H_A) > 0.$$

Hence $\delta \# 0$ gives

$$\mathcal{Q}_A^\psi(s) > 0.$$

Proof. By Definition 9413.737, the two displayed nonzero nodes have distinct Hamiltonian levels. A weighted finite distribution supported on at least two distinct real levels has positive variance. Multiplying by δ^2 gives positive centered Hamiltonian detuning energy whenever $\delta \# 0$. □

Theorem 9413.740 (Projected silence does not cancel detuning energy)**.** *The chord silence row*

$$C_A(s) = 0$$

does not imply

$$E_A^{\mathrm{res}}(s) = 0.$$

It only states first-order complex projection cancellation.

Proof. Definition 9413.733 shows that projected silence balances positive complex magnitudes against phase-interference cross terms. The detuning energy of Definition 9413.734 is instead the square of the normal logarithmic radial readout. By Definition 9413.735, that normalized energy is δ^2 independently of the phase interference. Therefore projected silence does not remove normal detuning. Equivalently, after centering the Hamiltonian island, Definition 9413.738 shows that the first surviving nonunitary cost is the variance row. □

Definition 9413.741 (Stable zero-relevant resonance family)**.** *A* stable zero-relevant resonance family *for a critical-strip row s is a cofinal family of finite resonance chords whose projected silence rows are compatible with the completed zero ledger*

$$\mathsf{ZetaZero}(s).$$

The definition is a proof-role placeholder: it names the resonance families that a completed zero proof actually uses, and it does not assert that every visible finite chord is zero-relevant. For such a witness ρ, write $E_\rho^{\mathrm{res}}(s)$ for the normalized normal energy inherited from the finite resonance chords in that family.

Definition 9413.742 (Resonance no-hair boundary)**.** *The* resonance no-hair boundary *is*

$$\mathsf{ResonanceNoHair}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \prod_{\rho:\mathsf{StableResFam}_\zeta(s)} \Big(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to E_\rho^{\mathrm{res}}(s) = 0\Big).$$

It says that any resonance family that is genuinely zero-relevant for the completed zeta ledger must carry zero normalized normal detuning.

Corollary 9413.743 (Resonance no-hair is stable unitarity)**.** *Under* $\mathsf{ResonanceNoHair}_\zeta$*, every stable zero-relevant resonance family for a critical-strip zero is forced onto the unitary resonance surface.*

Proof. For a stable zero-relevant family ρ, resonance no-hair gives $E_\rho^{\mathrm{res}}(s) = 0$. The finite readout inherited by ρ is the normalized detuning readout of Definition 9413.735, so $E_\rho^{\mathrm{res}}(s) = \delta^2$. Hence $\delta = 0$ by located collapse. By Definition 9413.732, the associated Hamiltonian evolution is unitary. □

Theorem 9413.744 (Resonance no-hair is conditionally sufficient)**.** *If every critical-strip zero row carries a stable zero-relevant resonance family and* $\mathsf{ResonanceNoHair}_\zeta$ *holds for those families, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

By the stable-family hypothesis, let ρ be a stable zero-relevant resonance family associated to the completed zero row of z. The resonance no-hair boundary gives

$$E_\rho^{\mathrm{res}}(s) = 0.$$

By Definition 9413.735, the same energy is δ^2. Hence $\delta^2 = 0$, and located collapse gives $\delta = 0$. Therefore $\Re(s) = 1/2$, and the fixed-half section is obtained by Definition 9413.199. □

Corollary 9413.745 (Quartic endpoint zeros are finite resonance guidance)**.** *The fixed quartic endpoint-zero law is a finite model for the zero-relevant resonance distinction: branch interference produces zeros only after the active dominant family has been identified.*

Proof. Definition 9413.102 states that the endpoint-zero law is obtained from active local branches, amplitudes, a cosine interference model, and Rouché isolation. Definition 9413.99 shows that not every branch collision is zero-relevant; dominance is the additional ledger. The resonance no-hair boundary plays the analogous role for zeta: not every projected chord is the proof object, and a stable zero-relevant resonance family must satisfy the normal-energy boundary. □

Corollary 9413.746 (The resonance formulation refines the projection-kernel gap)**.** *The projection-kernel ambiguity of Definition 9413.723 can be read as a detuned resonance family whose projected chord is silent while its normal energy remains nonzero.*

Proof. The projection-kernel ambiguity combines completed projection zero with nonzero normal height. A finite resonance chord models the same pattern locally: $C_A(s) = 0$ is projected silence, while Definition 9413.735 reads the hidden normal height as $E_A^{\mathrm{res}}(s) = \delta^2$. Thus the resonance formulation does not replace the no-hair target; it expresses the same target at the level of zero-relevant chord families. □

9413.60 Local Half-Split Certificate

The resonance picture should not be read as if an entire infinite prime spiral suddenly collapses to zero. A zero-relevant event is local: after dominance and normalization have been displayed, the completed read is cut by a finite two-branch interference island. The RH-facing content is that a stable local cut must be a half-split. The real coordinate controls length balance; the imaginary coordinate records the remaining phase quantization.

Theorem 9413.747 (Two-vector silence is equal length and opposite phase)**.** *For nonzero complex vectors $A, B \in \mathsf{Complex}^{\uparrow}$, the row*

$$A + B = 0$$

is equivalent to

$$|A| = |B|$$

and

$$\arg(A) - \arg(B) \equiv \pi \pmod{2\pi}.$$

Proof. If $A + B = 0$, then $A = -B$, so the two vectors have equal modulus and opposite phase. Conversely, equal modulus and phase difference π give $A = -B$, hence $A + B = 0$. □

Concretized in §8881: Finite NameCert carrier for the local two-branch half-split certificate.

Definition 9413.748 (Local two-branch zero island)**.** *A* local two-branch zero island *for a critical-strip row s is a local decomposition*

$$\Xi(u) = A_n(u) + B_n(u) + R_n(u)$$

on a disk D_n around s, together with rows saying that A_n and B_n are the two dominant branches and that the remainder satisfies the Rouché boundary discipline required to preserve the local zero count. The row is zero-relevant only after these dominance and tail rows are displayed; a visible two-term split without them is merely a projection model.

Definition 9413.749 (Half-split coordinate)**.** *For a local two-branch zero island, the* half-split coordinate *at s is the length-balance row*

$$|A_n(s)| = |B_n(s)|.$$

Equivalently, the zero cuts the normalized local segment into two equal halves. The complementary phase row is

$$\arg(A_n(s)/B_n(s)) \equiv (2k_n - 1)\pi \pmod{2\pi}.$$

The half-split row records the real-length constraint, while the phase row records the height or time quantization.

Definition 9413.750 (Real-defect split formula)**.** *A local two-branch zero island has a* real-defect split formula *when there is a positive branch-separation scale H_n and a bounded correction row ε_n such that*

$$\log \frac{|A_n(s)|}{|B_n(s)|} = -2\delta H_n + \varepsilon_n$$

for

$$s = \frac{1}{2} + \delta + i\tau.$$

The term $-2\delta H_n$ is the normal length bias. The correction ε_n belongs to amplitude normalization, finite-window tail control, or coordinate choice; it is not allowed to encode a hidden noncompact real scale along a cofinal tower.

Definition 9413.751 (Zeta local half-split certificate)**.** *For a critical-strip zeta zero s, a* zeta local half-split certificate *consists of a cofinal family of local two-branch zero islands with:*

$$\mathsf{DominantTwoBranch}(A_n, B_n), \qquad \mathsf{RoucheTailControl}(R_n),$$

$$\mathsf{HalfSplit}(A_n, B_n, s), \qquad \mathsf{RealDefectSplitFormula}(H_n, \varepsilon_n),$$

and a no-hidden-real-correction row saying that ε_n cannot cofinally cancel $-2\delta H_n$ when $\delta \# 0$ and the displayed branch-separation scales remain unbounded or otherwise non-negligible.

Theorem 9413.752 (Local half-split certificates are conditionally sufficient)**.** *If every critical-strip zeta zero carries a zeta local half-split certificate, then the RH fixed-half section exists.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The certificate supplies local islands with

$$|A_n(s)| = |B_n(s)|.$$

Thus

$$\log \frac{|A_n(s)|}{|B_n(s)|} = 0.$$

The real-defect split formula gives

$$0 = -2\delta H_n + \varepsilon_n.$$

If $\delta \# 0$, the normal term cannot be cofinally cancelled by ε_n by the no-hidden-real-correction row. This contradicts the half-split equation. Hence $\neg(\delta \# 0)$, and located fixed-half collapse gives $\delta = 0$. Therefore $\Re(s) = 1/2$, and the fixed-half section follows from Definition 9413.199. □

Corollary 9413.753 (Phase carries the remaining zero data)**.** *Once a zero-relevant local island has been normalized to a half-split, the remaining zero-location data are phase data.*

Proof. By Definition 9413.747, the local zero condition for the dominant pair is equal length together with opposite phase. The half-split certificate supplies the equal length row. The remaining equation is therefore

$$\arg(A_n(s)/B_n(s)) \equiv (2k_n - 1)\pi \pmod{2\pi}.$$

If the active branches have phase factors of the form

$$A_n(s) \sim C_+ e^{-i\tau H_+}, \qquad B_n(s) \sim C_- e^{-i\tau H_-},$$

then the height coordinate τ is determined by the corresponding phase congruence. The real part supplies length balance; the imaginary part records phase quantization. □

Corollary 9413.754 (Quartic endpoint zeros supply the finite template)**.** *The fixed quartic endpoint-zero law is the finite model for the local half-split certificate.*

Proof. In the quartic carrier, endpoint zeros are obtained only after the positive-axis dominant branches, local Puiseux expansions, partial-fraction amplitudes, and Rouché tail control are displayed, as recorded in Definition 9413.102. The active two-branch sum is transported to a shifted cosine model, whose zeros occur at odd half-periods. This is precisely the finite half-split pattern: equal dominant lengths with phase difference an odd multiple of π. The quartic result remains a fixed-window finite template, not a global zeta theorem. □

Corollary 9413.755 (Local half-split refines resonance no-hair)**.** *The local half-split certificate refines the resonance no-hair boundary by identifying the finite zero island on which the normal-energy condition is zero-relevant.*

Proof. The resonance no-hair boundary of Definition 9413.742 states that a stable zero-relevant resonance family has zero normalized detuning. The local half-split certificate explains what makes the family zero-relevant: a two-branch dominant island, Rouché tail control, length equality, and a real-defect split formula. Thus the certificate prevents projected silence from being mistaken for a proof object; it requires the local segment on which the zero actually acts. □

Corollary 9413.756 (Relation to solenoid and selector routes)**.** *The split-prime, Dedekind, and Zeckendorf solenoid routes supply possible source mechanisms for the branch-separation scale H_n in a local half-split certificate.*

Proof. In the fixed split-prime route, H_n is supplied by the repeated local rows above a rational prime q. In the Dedekind route, it is supplied by the prime-ideal rows in the lifted tower. In the Zeckendorf selector route, it is supplied by the golden-cofinal prime family selected by the no-adjacent grammar. Each route contributes a source for the length ledger; solenoid compatibility and Lee–Yang stability are then the candidate mechanisms preventing a hidden real correction from absorbing a nonzero δ. □

9413.61 Endpoint Bisection Certificate

The infinite spiral picture is too coarse if it is read as a literal endpoint of the full Dirichlet polyline. The zero-relevant object must be a finite endpoint model equipped with dominance, tail control, and a distinguished segment on which the zero acts. The fixed quartic carrier supplies the finite

template: endpoint zeros are produced by active branch interference after a dominant endpoint window and a Rouché ledger have been displayed. For zeta, the corresponding candidate is the Riemann–Siegel or approximate-functional-equation endpoint window, where the completed read is split into two finite arms and a controlled remainder.

Definition 9413.757 (Normalized zeta segment channel)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau$$

and $n \geq 1$*, the* normalized zeta segment channel *is*

$$u_n(s) := n^{1/2-s} = n^{-\delta} e^{-i\tau \log n}.$$

Its straightened log-length defect is

$$\ell_n(s) := \log |u_n(s)| = -\delta \log n.$$

The raw Dirichlet or eta segment may have a baseline length such as $n^{-1/2}$*; the normalized channel removes that baseline and records only the unitary versus radial component. Thus the critical line is the row*

$$|u_n(s)| = 1$$

for every displayed n*.*

Theorem 9413.758 (Weighted endpoint energy reads the normal displacement)**.** *For* $N \geq 2$*, define*

$$\mathcal{E}_N^{\zeta}(s) := \sum_{1 \leq n \leq N} \frac{1}{n} \left(\log |n^{1/2-s}| \right)^2.$$

If $s = 1/2 + \delta + i\tau$*, then*

$$\mathcal{E}_N^{\zeta}(s) = \delta^2 \sum_{1 \leq n \leq N} \frac{(\log n)^2}{n}.$$

Consequently, if $\delta \# 0$*, then*

$$\sup_N \mathcal{E}_N^{\zeta}(s) = \infty.$$

Proof. By Definition 9413.757,

$$\log |n^{1/2-s}| = -\delta \log n.$$

Squaring and summing with weight $1/n$ gives the displayed identity. The positive summands $(\log n)^2/n$ are unbounded in total: for $N \geq 2$,

$$\sum_{2 \leq n \leq N} \frac{(\log n)^2}{n}$$

dominates a cofinal subseries obtained by grouping n into dyadic blocks, and the block contributions diverge. Multiplication by $\delta^2 > 0$ preserves divergence in the located positive readout. □

Definition 9413.759 (Finite endpoint segment and bisection parameter)**.** *Let*

$$P_m(s) := \sum_{1 \leq n \leq m} v_n(s)$$

be a finite partial read in a displayed endpoint model. The endpoint segment *at m is the affine segment*

$$P_{m-1}(s) + \lambda v_m(s), \qquad 0 \le \lambda \le 1.$$

If the segment contains the origin, its bisection parameter $\lambda_m(s)$ *is the value satisfying*

$$P_{m-1}(s) + \lambda_m(s) v_m(s) = 0.$$

Theorem 9413.760 (Endpoint bisection is endpoint symmetry)**.** *Assume that a finite endpoint segment contains the origin. Then*

$$\lambda_m(s) = \frac{1}{2}$$

if and only if

$$P_m(s) = -P_{m-1}(s).$$

Proof. The origin condition is

$$P_{m-1}(s) + \lambda_m(s) v_m(s) = 0.$$

If $\lambda_m(s) = 1/2$, then

$$P_{m-1}(s) = -\frac{1}{2} v_m(s),$$

and hence

$$P_m(s) = P_{m-1}(s) + v_m(s) = \frac{1}{2} v_m(s) = -P_{m-1}(s).$$

Conversely, if $P_m(s) = -P_{m-1}(s)$, then

$$2P_{m-1}(s) + v_m(s) = 0,$$

so the origin is reached at $\lambda = 1/2$. □

Definition 9413.761 (Riemann–Siegel endpoint model row)**.** *A* Riemann–Siegel endpoint model row *for a completed zeta read at height τ consists of:*

$$N(\tau) \sim \sqrt{\frac{\tau}{2\pi}},$$

two finite arms of approximate-functional-equation type

$$\sum_{n \le x} n^{-s}, \qquad \chi(s) \sum_{n \le y} n^{s-1}, \qquad xy \sim \frac{\tau}{2\pi},$$

a symmetric endpoint choice $x \sim y \sim N(\tau)$, and a remainder row. On the critical line this row is transported to a real finite Riemann–Siegel read of the form

$$2 \sum_{n \le N(\tau)} n^{-1/2} \cos(\theta(\tau) - \tau \log n)$$

plus the displayed remainder. The row is a candidate zero-relevant endpoint model only when dominance and tail bounds identify a local zero island.

Definition 9413.762 (Endpoint bisection certificate)**.** *For a critical-strip zeta zero $s = 1/2+\delta+i\tau$, a* zeta endpoint bisection certificate *consists of a cofinal family of canonical finite endpoint models $F_N(s)$ with endpoint segments $e_N(s)$ and positive endpoint scales h_N, together with rows:*

$$\Xi(s) = F_N(s) + T_N(s), \qquad \mathsf{RoucheGap}(F_N, T_N),$$

$$\mathsf{ZeroRelevantEndpoint}(e_N, s), \qquad \lambda_N(s) \to \frac{1}{2},$$

$$\lambda_N(s) = \frac{1}{2} \Longleftrightarrow |e^{-(s-1/2)h_N}| = 1,$$

and a cofinal growth row such as

$$h_N \to \infty.$$

The Rouché row prevents a projected finite chord from being treated as a zero certificate. The straightened equivalence says that bisection in the endpoint segment is the unit-length condition in the normalized normal coordinate.

Theorem 9413.763 (Endpoint bisection certificates are conditionally sufficient)**.** *If every critical-strip zeta zero carries a zeta endpoint bisection certificate whose limiting bisection row forces*

$$\log |e^{-(s-1/2)h_N}| \to 0,$$

then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The certificate supplies positive scales h_N with $h_N \to \infty$ and the limiting straightened read

$$\log |e^{-(s-1/2)h_N}| \to 0.$$

But

$$\log |e^{-(s-1/2)h_N}| = -\delta h_N.$$

If $\delta \# 0$, then the displayed expression cannot converge to 0 along a cofinal sequence with $h_N \to \infty$. Hence $\neg(\delta \# 0)$. Located fixed-half collapse gives $\delta = 0$. Therefore $\Re(s) = 1/2$, and Definition 9413.199 gives the fixed-half section. □

Corollary 9413.764 (Endpoint bisection refines local half-split)**.** *The endpoint bisection certificate refines the local half-split certificate by specifying a zeta-native finite endpoint model in which the half-split is to be read.*

Proof. Definition 9413.751 requires a local two-branch zero island, Rouché tail control, length equality, and a real-defect split formula. The endpoint bisection certificate adds a candidate source for that local island: a Riemann–Siegel or approximate-functional-equation endpoint window with a distinguished endpoint segment. Thus the half-split row is no longer attached to an arbitrary finite chord; it is attached to a completed zeta endpoint model whose tail and dominance ledgers are part of the certificate. □

Corollary 9413.765 (The quartic endpoint row is the finite bisection template)**.** *The fixed quartic endpoint-zero row is the finite template for the zeta endpoint bisection route.*

Proof. By Definition 9413.102, the quartic endpoint zeros are obtained only after the active colliding branches, endpoint scale, amplitude rows, and Rouché isolation are displayed. The resulting shifted cosine zeros are finite half-period interference points. The zeta endpoint bisection certificate asks for the analogous finite endpoint structure in the completed zeta read: not a bare infinite polyline endpoint, but a zero-relevant endpoint window with a distinguished bisection segment and a controlled remainder. □

Corollary 9413.766 (Solenoid fibres do not hide real scale)**.** *In an endpoint bisection certificate, solenoid-compatible hidden data may carry phase and prime-register information, but not the real positive scale* $e^{-\delta h_N}$*.*

Proof. The solenoid route records compact phase together with prime-adic fibre data, as in Definition 9413.780. The normal endpoint factor

$$e^{-\delta h_N}$$

belongs to the noncompact positive real scale direction. The straightened equivalence in Definition 9413.762 therefore cannot be discharged by placing δ inside the hidden solenoid fibre. The fibre may preserve residue or phase information; it is not a reservoir for the real radial defect. □

Concretized in §8940: Endpoint bisection certificates are realized as finite NameCert data over RHRoute endpoint intervals, branch choices, preservation rows, and local half-split handoff.

9413.62 Zeta Resonance Bound Stability

The resonance no-hair boundary names the obstruction, but the route out of circularity is an upper-bound theorem. The desired zeta analogue of a Lee–Yang location theorem is not another restatement of the critical line. It is a stability row saying that any resonance family which is actually zero-relevant for the completed zeta ledger has bounded normal energy along its finite prime-adic windows. Off the fixed-half surface, the same finite-window energy is forced to diverge; therefore a genuine upper-bound theorem would place stable completed zeros on the unitary boundary.

Definition 9413.767 (Resonance window square energy)**.** *For a finite prime-adic parity window*

$$B \subset \mathsf{Prime}^{\uparrow} \times \mathsf{Nat}^{\uparrow},$$

the resonance window square energy *is*

$$\mathcal{N}_B(s) := \sum_{(p,j)\in B} \Big(\log |U_{p,j}^{\mathrm{pad}}(s)|\Big)^2 .$$

This is the unnormalized positive normal energy of the displayed Hamiltonian levels in the resonance family.

Theorem 9413.768 (Finite resonance energy reads the off-half displacement)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

every finite prime-adic parity window B *satisfies*

$$\mathcal{N}_B(s) = \delta^2 \sum_{(p,j)\in B} (2^j \log p)^2.$$

Proof. By the prime-adic parity channel definition,

$$U_{p,j}^{\mathrm{pad}}(s) = p^{2^j(1/2-s)} = p^{-2^j\delta}e^{-i2^j\tau\log p}.$$

Thus

$$\log|U_{p,j}^{\mathrm{pad}}(s)| = -2^j\delta\log p.$$

Squaring and summing over B gives the displayed formula. □

Definition 9413.769 (Cofinal resonance support)**.** *A stable zero-relevant resonance family ρ has* cofinal prime-adic support *when its finite windows contain a cofinal subsystem of prime-adic parity windows whose square weights*

$$\sum_{(p,j)\in B}(2^j\log p)^2$$

are unbounded.

Theorem 9413.770 (Off-half cofinal resonance energy is unbounded)**.** *Let ρ have cofinal prime-adic support. If*

$$s = \frac{1}{2} + \delta + i\tau$$

and $\delta\#0$, then

$$\sup_{B\preceq\rho}\mathcal{N}_B(s) = \infty,$$

where $B \preceq \rho$ ranges over the finite windows displayed by the family.

Proof. By Definition 9413.768, each finite displayed window satisfies

$$\mathcal{N}_B(s) = \delta^2\sum_{(p,j)\in B}(2^j\log p)^2.$$

The cofinal support hypothesis says that the square-weight sums are unbounded along the family. Since $\delta\#0$, multiplication by δ^2 preserves unboundedness in the located positive readout. □

Definition 9413.771 (Zeta resonance bound)**.** *The* zeta resonance bound *is the row*

$$\mathsf{ZetaResonanceBound}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow}\prod_{\rho:\mathsf{StableResFam}_\zeta(s)}\left(\mathsf{ZetaZero}(s)\times\mathsf{InCritStrip}(s)\to\sup_{B\preceq\rho}\mathcal{N}_B(s)<\infty\right).$$

It says that every stable resonance family actually used by a completed zero proof has a uniform finite normal-energy bound. The row is zero-relevance data: it does not require all visible finite chords to be controlled by one common window, and it allows different zero rows to use different stable resonance families.

Theorem 9413.772 (The zeta resonance bound is conditionally sufficient)**.** *Assume every critical-strip completed zero carries a stable zero-relevant resonance family with cofinal prime-adic support. If*

$$\mathsf{ZetaResonanceBound}_\zeta$$

holds for those families, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

Let ρ be the stable zero-relevant resonance family attached to the completed zero row. The zeta resonance bound gives

$$\sup_{B \preceq \rho} \mathcal{N}_B(s) < \infty.$$

If $\delta \# 0$, then Definition 9413.770 gives the contradictory extended readout

$$\sup_{B \preceq \rho} \mathcal{N}_B(s) = \infty.$$

Thus $\neg(\delta \# 0)$. Located collapse gives $\delta = 0$, and Definition 9413.199 supplies the fixed-half section. □

Theorem 9413.773 (Strict non-archimedean dominance prevents finite silence)**.** *Let $|\cdot|_\nu$ be a non-archimedean absolute value on a finite field of coefficients. For a finite family $(x_a)_{a \in A}$, if one term strictly dominates,*

$$|x_{a_0}|_\nu > \max_{a \neq a_0} |x_a|_\nu,$$

then

$$\sum_{a \in A} x_a \neq 0.$$

Proof. The strong triangle inequality gives

$$\left| \sum_{a \neq a_0} x_a \right|_\nu \leq \max_{a \neq a_0} |x_a|_\nu < |x_{a_0}|_\nu.$$

Hence the sum of the remaining terms cannot equal $-x_{a_0}$. Therefore the full finite sum is nonzero. □

Definition 9413.774 (Lee–Yang type resonance stability certificate)**.** *A* Lee–Yang type resonance stability certificate *for a stable resonance family ρ consists of four displayed rows:*

$$\mathsf{DominantFamily}(\rho), \qquad \mathsf{NoStrictLocalDominance}(\rho),$$

$$\mathsf{TailBound}(\rho), \qquad \mathsf{FarEndEnergyBound}(\rho).$$

The dominance row identifies which finite spectral or prime-adic levels are zero-relevant. The no-strict-local-dominance row is the non-archimedean or tropical analogue of the preceding theorem: a finite zero can occur only where dominant levels tie rather than where one level strictly dominates. The tail-bound row prevents non-dominant levels from changing the zero ledger. The far-end energy-bound row is the uniform bound consumed by Definition 9413.771.

Theorem 9413.775 (Lee–Yang type stability supplies the resonance bound)**.** *If every stable zero-relevant resonance family for a completed critical-strip zero carries a Lee–Yang type resonance stability certificate, then*

$$\mathsf{ZetaResonanceBound}_\zeta$$

holds for those families.

Proof. The certificate explicitly contains the row

$$\mathsf{FarEndEnergyBound}(\rho),$$

which is the finite uniform bound on

$$\sup_{B \preceq \rho} \mathcal{N}_B(s).$$

The other rows explain why this bound is zero-relevant rather than a bound on an arbitrary visible chord: dominance selects the active family, local non-archimedean or tropical stability prevents strict-dominance zeros, and the tail bound preserves the finite zero ledger through the far-end passage. Therefore the bound row is exactly Definition 9413.771. □

Corollary 9413.776 (The quartic carrier is the finite stability template)**.** *The fixed quartic spectral carrier supplies the finite template for the zeta resonance-bound target: zero location is controlled by dominance, local branch interference, and a tail estimate, not by a bare projection zero.*

Proof. Definition 9413.99 separates the zero-relevant dominant branch from other visible branch data. Definition 9413.102 then uses active amplitudes, local interference, and Rouché isolation to locate endpoint zeros. Thus the quartic row has the same proof-role shape as Definition 9413.774: dominance, stability of the active family, and a remainder bound precede the zero-location conclusion. □

Corollary 9413.777 (The bound target avoids a fixed-window assumption)**.** *The zeta resonance-bound route does not assert that all completed zeros are controlled by one finite prime window. It quantifies over the stable resonance family attached to each zero row.*

Proof. The quantified family $\rho : \mathsf{StableResFam}_\zeta(s)$ in Definition 9413.771 is indexed by the zero coordinate s. Different completed zero rows may therefore expose different finite windows, different prime-adic depths, and different dominant resonance islands. The common assertion is only the stability bound: whichever family becomes zero-relevant must have finite normal energy at the completed far end. □

9413.63 Solenoid Lee–Yang Resonance Boundary

The prime-torus and prime-adic slicing pictures can be joined by a solenoid carrier. A solenoid is a compact phase object whose visible face is a unit circle and whose hidden fibres record the admitted prime-adic directions. This is the natural place to state the Lee–Yang-like zeta target: if the zero-relevant resonance data of a completed zeta zero is really realized inside such a compact solenoid phase carrier, then there is no room for an additional real positive normal scale.

Definition 9413.778 (S-localization)**.** *For a displayed prime family $S \subseteq \mathsf{Prime}^{\uparrow}$, the S-localization of the integers is*

$$\mathbb{Z}[S^{-1}] := \left\{ \frac{a}{b} \in \mathbb{Q} : a \in \mathbb{Z},\ b \in \mathbb{N},\ \mathsf{PrimeDiv}(b) \subseteq S \right\}.$$

The inclusion

$$\iota : \mathbb{Z} \hookrightarrow \mathbb{Z}[S^{-1}]$$

records the ordinary integer phase row, while primes in S are declared invertible gates in the localized carrier.

Definition 9413.779 (S-solenoid over the unit circle)**.** *The* S-solenoid *is the compact abelian phase carrier*

$$\Sigma_S := \widehat{\mathbb{Z}[S^{-1}]},$$

the Pontryagin dual of the localized integer carrier. Dualizing ι *gives the visible unit-circle projection*

$$\pi_S : \Sigma_S \twoheadrightarrow \widehat{\mathbb{Z}} \cong \mathbb{T}.$$

Its invisible fibre is

$$K_S := \ker(\pi_S).$$

For finite S *with* $N = \prod_{p \in S} p$*, the same carrier may be read as the inverse-limit solenoid*

$$\Sigma_S \cong \varprojlim \left(\mathbb{T} \xleftarrow{z \mapsto z^N} \mathbb{T} \xleftarrow{z \mapsto z^N} \mathbb{T} \xleftarrow{z \mapsto z^N} \cdots \right).$$

Thus Σ_S *packages the visible unit phase together with the* S*-prime fibre registers behind that phase.*

Definition 9413.780 (Solenoid-compatible resonance lift)**.** *Let* $s = 1/2 + \delta + i\tau$*. A stable zero-relevant resonance family* $\rho : \mathsf{StableResFam}_\zeta(s)$ *has a* solenoid-compatible resonance lift *when there is a nonempty prime family* S_ρ*, a compact phase carrier* Σ_{S_ρ}*, and a factorization of the zero-relevant phase data through*

$$\Sigma_{S_\rho} \twoheadrightarrow \mathbb{T}$$

such that every displayed node

$$a = (p, j), \qquad p \in S_\rho,$$

has local coordinate

$$U_a(s) = p^{2^j(1/2-s)} = e^{-\delta h_a} e^{-i\tau h_a}, \qquad h_a := 2^j \log p,$$

and the real positive modulus row

$$|U_a(s)| = e^{-\delta h_a}$$

is required to come from a continuous real positive character of the compact solenoid phase carrier, rather than from an additional noncompact scale coordinate. Write $\mathsf{SolenoidCompatible}(\rho, s)$ *for the existence of such a lift.*

Lemma 9413.781 (Compact phase carriers have no nontrivial positive scale character)**.** *Let* G *be a compact group. Every continuous group homomorphism*

$$\rho : G \to \mathsf{Real}^{\uparrow}_{>0}$$

is trivial:

$$\rho(g) = 1 \qquad (g \in G).$$

Proof. The image $\rho(G)$ is a compact subgroup of $\mathsf{Real}^{\uparrow}_{>0}$. Applying the logarithm gives a compact subgroup

$$\log \rho(G) \subseteq (\mathsf{Real}^{\uparrow}, +).$$

The additive real line has no nontrivial compact subgroup: if a nonzero element x were present, then the infinite set $\{nx : n \in \mathbb{Z}\}$ would be an unbounded subgroup. Hence $\log \rho(G) = \{0\}$, so $\rho(G) = \{1\}$. □

Theorem 9413.782 (Solenoid-compatible resonance has no normal scale)**.** *Let $s = 1/2 + \delta + i\tau$ and suppose a stable zero-relevant resonance family for s has a solenoid-compatible resonance lift. Then*

$$\delta = 0.$$

Proof. For each displayed node $a = (p, j)$ in the lift,

$$|U_a(s)| = e^{-\delta h_a} = p^{-2^j \delta}.$$

By the solenoid-compatibility hypothesis, this modulus row comes from a continuous real positive character of the compact solenoid carrier. By Definition 9413.781, the character is trivial, so $|U_a(s)| = 1$. Thus

$$p^{-2^j \delta} = 1.$$

Since $2^j \log p > 0$, taking logarithms gives

$$-2^j \delta \log p = 0,$$

hence $\delta = 0$. □

Definition 9413.783 (Zeta solenoid Lee–Yang row)**.** *The* zeta solenoid Lee–Yang row *is the assertion*

$$\mathsf{ZetaSolenoidLeeYang}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \left(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \sum_{\rho:\mathsf{StableResFam}_\zeta(s)} \mathsf{SolenoidCompatible}(\rho, s) \right).$$

It says that every completed critical-strip zero is carried by some stable zero-relevant resonance family whose phase data factors through a compact S-solenoid and has no extra noncompact real positive scale coordinate.

Theorem 9413.784 (Solenoid Lee–Yang is conditionally sufficient)**.** *If*

$$\mathsf{ZetaSolenoidLeeYang}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

The zeta solenoid Lee–Yang row supplies a stable resonance family ρ with a solenoid-compatible lift. By Definition 9413.782,

$$\delta = 0.$$

Therefore $\Re(s) = 1/2$, and the fixed-half section is obtained through Definition 9413.199. □

Corollary 9413.785 (Solenoid compatibility is a compact-form no-hair row)**.** *The solenoid-compatible form of the RH-facing obstruction is the disappearance of the noncompact real positive component in*

$$U_{p,j}(s) = e^{-\delta h_{p,j}} e^{-i\tau h_{p,j}}.$$

Proof. The phase factor $e^{-i\tau h_{p,j}}$ belongs to the visible unit-circle projection of the solenoid. The hidden p-adic fibre records compact local phase data behind that projection. The factor $e^{-\delta h_{p,j}}$ is different: it is a real positive scale row. By Definition 9413.781, such a row cannot be supplied nontrivially by a compact phase carrier. Thus solenoid compatibility is exactly the compact-carrier version of the no-hair condition. □

Corollary 9413.786 (The unclosed part is the zeta-specific dominance theorem)**.** *The compact solenoid argument does not by itself prove the zeta solenoid Lee–Yang row. The unclosed content is the zeta-specific dominance theorem constructing, for each completed zero, a stable zero-relevant resonance family whose phase data is solenoid-compatible.*

Proof. Definition 9413.782 is conditional on the existence of a solenoid-compatible lift. It does not construct the lift from the completed zero ledger. The construction requires the same proof-role ingredients isolated in Definition 9413.774: dominance of the active resonance family, exclusion of strict local dominance, a tail bound, and far-end energy control. Thus the remaining claim is not another algebraic restatement of $\delta = 0$; it is the zeta-specific Lee–Yang or dominance theorem which produces the compact solenoid-compatible carrier for the zero-relevant data. □

Corollary 9413.787 (Representation-theoretic analogy)**.** *The solenoid route has the same proof-role shape as a compactness criterion for a semisimple representation row: a decomposition into a compact unitary component and a noncompact real component is forced onto the compact side once the zero-relevant data are known to lie on the unit-circle boundary.*

Proof. In a representation-theoretic Lee–Yang row one reads

$$g = \exp(\mu)u,$$

with u in a compact direction and μ in a noncompact direction. Unit-circle zero data forces the fundamental weights to vanish on μ, hence $\mu = 0$. In the zeta solenoid row the corresponding decomposition is

$$U_{p,j}(s) = e^{-\delta h_{p,j}} e^{-i\tau h_{p,j}},$$

where the second factor is compact phase and the first is real positive scale. Solenoid compatibility and Definition 9413.781 force the scale component to be trivial, hence $\delta = 0$. □

Concretized in §9075: The SolenoidPhaseUp packet realizes the S-localization and compact solenoid phase carrier as finite NameCert rows.

9413.64 Zeta Solenoid Lee–Yang Bound

The solenoid boundary and the resonance-bound boundary describe two faces of the same RH-facing proof role. The solenoid face says that a zero-relevant phase state realized in a compact S-solenoid cannot carry an additional real positive scale. The bound face says that a zero-relevant resonance family must have uniformly bounded normal energy along its finite prime-adic windows. The combined target is therefore a zeta-specific Lee–Yang theorem with both parts displayed: each completed zero must be captured by a stable dominant resonance tower whose phase data is solenoid-compatible and whose normal energy is bounded at the far end.

Definition 9413.788 (Zero-specific cofinal resonance tower)**.** *For a critical-strip row s, a* zero-specific cofinal resonance tower *is a stable zero-relevant resonance family*

$$\rho : \mathsf{StableResFam}_\zeta(s)$$

whose finite windows

$$B \preceq \rho, \qquad B \subset \mathsf{Prime}^\uparrow \times \mathsf{Nat}^\uparrow,$$

form a cofinal prime-adic support in the sense of Definition 9413.769. The tower is indexed by the zero row s; different completed zero rows may have different displayed windows, depths, and dominant resonance islands.

Definition 9413.789 (Solenoid Lee–Yang bound witness)**.** *For a critical-strip row s, a* solenoid Lee–Yang bound witness *is a stable zero-relevant resonance family ρ together with the rows*

$$\mathsf{SolenoidCompatible}(\rho, s),$$

$$\mathsf{CofinalPAdicSupport}(\rho),$$

and

$$\sup_{B \preceq \rho} \mathcal{N}_B(s) < \infty.$$

The first row is the compact solenoid phase condition of Definition 9413.780. The second row says that the finite windows of ρ are cofinal in the prime-adic resonance sense. The third row is the uniform normal-energy bound of Definition 9413.771. Write $\mathsf{SolenoidLeeYangBoundWitness}(\rho, s)$ *for this displayed packet.*

Definition 9413.790 (Zeta solenoid Lee–Yang bound)**.** *The* zeta solenoid Lee–Yang bound *is the row*

$$\mathsf{ZetaSolenoidLeeYangBound}_\zeta := \prod_{s:\mathsf{Complex}^\uparrow} \Bigg(\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \sum_{\rho:\mathsf{StableResFam}_\zeta(s)} \mathsf{SolenoidLeeYangBoundWi}$$

It asserts that every completed critical-strip zero is carried by some zero-specific stable dominant resonance tower whose phase data lies in a compact solenoid and whose normal energy remains bounded along its cofinal finite windows.

Theorem 9413.791 (The solenoid Lee–Yang bound is conditionally sufficient)**.** *If*

$$\mathsf{ZetaSolenoidLeeYangBound}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s, and write

$$s = \frac{1}{2} + \delta + i\tau.$$

The zeta solenoid Lee–Yang bound supplies a witness ρ for the zero row of s. In particular it supplies

$$\mathsf{SolenoidCompatible}(\rho, s).$$

By Definition 9413.782, this gives $\delta = 0$. Hence $\Re(s) = 1/2$, and Definition 9413.199 supplies the fixed-half section. □

Theorem 9413.792 (The bound component gives the same exclusion)**.** *Let a completed critical-strip zero carry a solenoid Lee–Yang bound witness ρ. Then the normal-energy component alone excludes $\delta\#0$.*

Proof. The witness includes cofinal prime-adic support and the bound

$$\sup_{B\preceq\rho}\mathcal{N}_B(s)<\infty.$$

If $\delta\#0$, then Definition 9413.770 gives

$$\sup_{B\preceq\rho}\mathcal{N}_B(s)=\infty,$$

contradicting the bound. Thus $\neg(\delta\#0)$, and the located fixed-half collapse row gives $\delta=0$. □

Corollary 9413.793 (Two noncircular readings of the same target)**.** *The zeta solenoid Lee–Yang bound can be attacked either as a compact phase theorem or as a finite-to-infinite energy theorem.*

Proof. The compact phase reading uses Definition 9413.781: a compact solenoid phase carrier has no nontrivial real positive character, so a solenoid-compatible zero-relevant state cannot retain the factor $e^{-\delta h_{p,j}}$. The energy reading uses Definition 9413.768 and Definition 9413.770: off-half displacement forces unbounded positive normal energy along any cofinal prime-adic resonance tower. Both readings are proof roles for the same zero-relevance claim rather than restatements of the target surface. □

Definition 9413.794 (Dominant solenoid resonance certificate)**.** *A* dominant solenoid resonance certificate *for a completed zero row consists of the following displayed data:*

$$\mathsf{StableResFam}_\zeta(s),\qquad \mathsf{DominantFamily}(\rho),$$

$$\mathsf{SolenoidCompatible}(\rho,s),\qquad \mathsf{FarEndEnergyBound}(\rho).$$

The stable-family row says which resonance tower is zero-relevant. The dominance row separates that tower from visible but irrelevant resonance data. The solenoid row supplies the compact phase carrier. The bound row supplies the finite-to-infinite normal-energy control. Write $\mathsf{DominantSolenoidResonanceCertificate}(\rho,s)$ *for this certificate packet.*

Theorem 9413.795 (Dominant solenoid certificates supply the combined bound)**.** *If every completed critical-strip zero carries a dominant solenoid resonance certificate with cofinal prime-adic support, then*

$$\mathsf{ZetaSolenoidLeeYangBound}_\zeta$$

holds.

Proof. For a zero row s, the certificate supplies the stable family ρ, its dominance row, its solenoid compatibility row, and its far-end energy bound. Cofinal prime-adic support supplies the tower condition in Definition 9413.788. The far-end energy bound is exactly the inequality

$$\sup_{B\preceq\rho}\mathcal{N}_B(s)<\infty.$$

Thus the displayed data are precisely a solenoid Lee–Yang bound witness for s. □

Corollary 9413.796 (Finite spectral and tower-splitting templates)**.** *The combined target keeps the roles of the finite templates distinct: quartic spectral geometry models dominance and local zero isolation, while prime-splitting tower patterns model why a zero-specific resonance family may expose many local prime rows without being a single fixed finite window.*

Proof. The quartic carrier is used only as a finite dominance template, as in Definition 9413.776: zero location follows after the zero-relevant dominant family and its remainder bound are displayed. A prime-splitting tower is a different template: one displayed rational prime can generate many local prime rows in higher layers, so the resonance family attached to a zero should be allowed to grow through a tower. Neither template proves the zeta claim by itself. Together they explain why Definition 9413.790 quantifies over a zero-specific cofinal resonance tower rather than over one universal finite prime window. □

Theorem 9413.797 (The remaining proof load is zeta-specific dominance)**.** *The unclosed content of the combined route is the implication*

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s) \to \sum_{\rho:\mathsf{StableResFam}_\zeta(s)} \mathsf{DominantSolenoidResonanceCertificate}(\rho, s).$$

Proof. All subsequent consequences are already displayed. A dominant solenoid resonance certificate gives the combined bound by Definition 9413.795. The combined bound gives the RH fixed-half section by Definition 9413.791. What is not supplied by the zero predicate itself is the construction of the stable dominant resonance family, its solenoid-compatible phase carrier, and its far-end energy bound. That construction is the zeta-specific Lee–Yang theorem targeted by this route. □

9413.65 Solenoid Lee–Yang Source Gap

The combined solenoid Lee–Yang target must be read as a source theorem, not as another name for the fixed-half surface. A row of the form "completed zeta zero implies zero normal displacement" has the same logical shape as RH unless it is backed by an independent mechanism that constructs the zero-relevant resonance carrier. The route becomes noncircular only when the zero predicate is connected to a stable dominant resonance family, a tail-isolated local zero island, a compact solenoid phase lift, and a far-end energy bound.

Definition 9413.798 (Fixed-half alias row)**.** *For a critical-strip zero coordinate s, write*

$$\mathsf{CritZero}(s) := \mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s).$$

A proposed RH-facing row is a fixed-half alias row *when its only zero-specific content is one of the fixed-surface readouts*

$$\mathsf{OnCritLine}(s), \qquad \delta(s) = 0, \qquad \delta(s)^2 = 0,$$

or a direct equivalent such as zero normal energy or no normal hair, without a displayed source mechanism producing these rows from dominance, stability, solenoid compatibility, or tail control.

Theorem 9413.799 (Fixed-half aliases do not discharge the proof load)**.** *A fixed-half alias row does not by itself prove the RH fixed-half section.*

Proof. The RH section asks for a total assignment

$$\prod_{s:\mathsf{Complex}^{\uparrow}} (\mathsf{CritZero}(s) \to \mathsf{OnCritLine}(s)).$$

A fixed-half alias row names the output surface of this assignment, or a readout equivalent to that surface. It does not supply a contradiction to the hidden normal-coordinate counterform of Definition 9413.853, since that counterform is precisely a zero-source row with $\delta\#0$ whose normal coordinate has not yet been linked to the zero ledger. By Definition 9413.854, rewriting the fixed surface in mirror, prime-window, modulus, or energy coordinates leaves the counterform unexcluded until a zero-relevance mechanism is supplied. □

Definition 9413.800 (Dominant solenoid source mechanism)**.** *A* dominant solenoid source mechanism *for a critical-strip zero coordinate s is a packet consisting of a resonance family*

$$\rho : \mathsf{StableResFam}_{\zeta}(s),$$

together with the displayed rows

$$\mathsf{DominantFamily}(\rho), \qquad \mathsf{TailBound}(\rho), \qquad \mathsf{RoucheGap}(\rho),$$

$$\mathsf{SolenoidCompatible}(\rho, s), \qquad \mathsf{FarEndEnergyBound}(\rho).$$

The dominance row identifies the zero-relevant family. The tail and Rouché rows make the local zero island stable under the completed readout. The solenoid row places the visible phase and the hidden prime-adic registers inside a compact S-solenoid. The far-end bound controls the normal energy along the cofinal prime-adic windows of the same family.

Theorem 9413.801 (Source mechanisms are the noncircular content)**.** *The RH-facing solenoid Lee–Yang route is noncircular exactly at the point where it supplies a dominant solenoid source mechanism from the completed zero ledger.*

Proof. The compact solenoid argument is already structural: by Definition 9413.782, a solenoid-compatible zero-relevant family has no nontrivial positive scale character, hence no δ-scale. The finite-to-infinite energy argument is also structural: by Definition 9413.770, an off-half coordinate forces unbounded square energy along cofinal prime-adic support. Neither argument constructs the zero-relevant family from the zero predicate.

A dominant solenoid source mechanism supplies the missing source data. Dominance separates the active family from arbitrary projected chords. The tail and Rouché rows ensure that the family controls a local zero island of the completed readout. Solenoid compatibility puts the hidden degrees of freedom in the compact prime-adic fibre rather than in a noncompact real scale coordinate. The far-end energy bound then becomes a theorem about the same zero-relevant family, not an asserted synonym for $\delta = 0$. □

Corollary 9413.802 (The combined bound must be read as a source theorem)**.** *The row*

$$\mathsf{ZetaSolenoidLeeYangBound}_{\zeta}$$

is noncircular only in its source reading: every critical-strip zero comes with a stable dominant solenoid resonance source.

Proof. If the combined bound is read only as

$$\mathsf{CritZero}(s) \to \delta(s) = 0$$

or as

$$\mathsf{CritZero}(s) \to \sup_{B \preceq \rho} \mathcal{N}_B(s) < \infty$$

with no independently constructed ρ, it is a fixed-half alias row in the sense of Definition 9413.798. In contrast, Definition 9413.790 quantifies over a displayed stable family and a solenoid Lee–Yang bound witness. Definition 9413.794 then refines the witness into dominance, solenoid compatibility, and far-end energy data. Under that reading, the row is a zeta-specific source theorem rather than a relabeling of the fixed-half output. □

Theorem 9413.803 (The source gap is the remaining theorem)**.** *The remaining theorem for the solenoid Lee–Yang route is the source construction*

$$\mathsf{CritZero}(s) \to \sum_{\rho:\mathsf{StableResFam}_\zeta(s)} \mathsf{DominantSolenoidSource}(\rho, s),$$

where $\mathsf{DominantSolenoidSource}(\rho, s)$ *denotes the packet of Definition 9413.800.*

Proof. Once such a source packet is supplied, its solenoid row gives $\delta = 0$ by Definition 9413.782. Its far-end bound also gives $\delta = 0$ by the unbounded-energy contradiction in Definition 9413.792. Thus the downstream implication to the RH fixed-half section is already accounted for by Definition 9413.791. What remains is not another surface equivalence, but the construction of the source packet from the completed zero row: the active family, its dominance, its local Rouché gap, its compact solenoid phase carrier, and its far-end bound. □

Corollary 9413.804 (Finite templates are source templates)**.** *The quartic spectral carrier and prime-splitting tower patterns support the source theorem only as templates for dominance and carrier growth; they are not substitutes for the zeta-specific source construction.*

Proof. The quartic carrier contributes the finite pattern recorded in Definition 9413.776: zero location follows after a dominant family, local interference, and a Rouché remainder bound have been displayed. The prime-splitting tower pattern contributes the carrier-growth pattern recorded in Definition 9413.796: one prime source can expose many local rows through a tower, so the zeta family should be allowed to be cofinal rather than a single fixed finite window. These templates identify the shape of the source certificate. They do not prove that every completed zeta zero carries such a certificate. □

9413.66 Fixed-Prime Split Resonance Tower

The solenoid source gap can be sharpened by separating two tasks that are often conflated. The zero-relevant resonance family need not begin as a window over all rational primes. A more economical source pattern fixes one rational prime q attached to the zero row, lets that prime split through a tower of arithmetic carriers, and reads the resulting local prime rows as the cofinal resonance support. This mirrors the unit-distance tower pattern: one split rational prime can generate many local prime rows while a global discriminant bound keeps the tower auditable.

Definition 9413.805 (Fixed split-prime tower)**.** *Let $q : \mathsf{Prime}^{\uparrow}$ be a displayed rational prime. A* fixed split-prime tower *for q is a chain of number-field rows*

$$K_0 \subset K_1 \subset K_2 \subset \cdots$$

with degree rows

$$f_n := [K_n : \mathbb{Q}],$$

such that

$$f_n \to \infty,$$

and q splits completely in each K_n:

$$q\mathcal{O}_{K_n} = \mathfrak{q}_{n,1} \cdots \mathfrak{q}_{n,f_n}.$$

The tower is globally controlled *when it carries a uniform root discriminant bound*

$$\mathrm{rd}(K_n) := |\operatorname{Disc}(K_n)|^{1/f_n} \le C$$

for a constant C independent of n.

Definition 9413.806 (q-split resonance nodes)**.** *For a fixed split-prime tower and a depth row $j : \mathsf{Nat}^{\uparrow}$, the displayed local resonance nodes at level n are*

$$a = (\mathfrak{q}_{n,i}, j), \qquad 1 \le i \le f_n.$$

Their Hamiltonian levels are

$$h_a := 2^j \log N\mathfrak{q}_{n,i}.$$

Since q splits completely,

$$N\mathfrak{q}_{n,i} = q,$$

so the level readout is

$$h_a = 2^j \log q.$$

A finite q-split resonance window at level n is a finite set

$$A_n \subset \{\mathfrak{q}_{n,1}, \ldots, \mathfrak{q}_{n,f_n}\} \times \mathsf{Nat}^{\uparrow}.$$

Definition 9413.807 (Positive-density split support)**.** *A family of q-split resonance windows (A_n) has* positive-density split support *when there are constants $c > 0$ and $j_0 : \mathsf{Nat}^{\uparrow}$ such that, for all sufficiently large n, at least cf_n nodes of the form*

$$(\mathfrak{q}_{n,i}, j_0)$$

belong to A_n.

Definition 9413.808 (q-split normal energy)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau,$$

the local channel of a q-split node a is

$$U_a(s) := \exp(-(s - 1/2)h_a) = e^{-\delta h_a} e^{-i\tau h_a}.$$

The normal logarithmic defect is

$$d_a(s) := \log |U_a(s)| = -\delta h_a,$$

and the q-split window energy is

$$\mathcal{E}^{(q)}_{A_n}(s) := \sum_{a \in A_n} d_a(s)^2.$$

Theorem 9413.809 (One split prime detects off-half displacement)**.** *Let (K_n) be a fixed split-prime tower for q, and let (A_n) have positive-density split support. If*

$$s = \frac{1}{2} + \delta + i\tau$$

and $\delta \# 0$, then

$$\mathcal{E}^{(q)}_{A_n}(s) \to \infty$$

along the tower.

Proof. Choose the depth j_0 and density constant c from Definition 9413.807. For every displayed node

$$a = (\mathfrak{q}_{n,i}, j_0),$$

complete splitting gives

$$h_a = 2^{j_0} \log q.$$

Hence

$$d_a(s)^2 = \delta^2 (2^{j_0} \log q)^2.$$

For all sufficiently large n, at least cf_n such nodes belong to A_n, so

$$\mathcal{E}^{(q)}_{A_n}(s) \geq cf_n \delta^2 (2^{j_0} \log q)^2.$$

The tower has $f_n \to \infty$, and the coefficient is positive when $\delta \# 0$. Therefore the displayed energy is unbounded and tends to the far end. □

Definition 9413.810 (q-solenoid resonance compatibility)**.** *Let q be a fixed rational prime. The associated one-prime solenoid is*

$$\Sigma_q := \widehat{\mathbb{Z}[q^{-1}]}$$

with projection

$$\pi_q : \Sigma_q \twoheadrightarrow \mathbb{T}$$

and hidden fibre

$$K_q := \ker(\pi_q) \cong \mathbb{Z}_q.$$

A q-split resonance tower is q-solenoid-compatible *for s when its phase data factors through π_q, its hidden registers are carried by the q-adic fibre rows generated by the split local primes, and it introduces no additional $\mathsf{Real}^{\uparrow}_{>0}$ scale coordinate.*

Theorem 9413.811 (q-solenoid compatibility has unit boundary)**.** *If a stable zero-relevant q-split resonance tower is q-solenoid-compatible for*

$$s = \frac{1}{2} + \delta + i\tau,$$

then $\delta = 0$.

Proof. For every displayed node a in the tower,

$$|U_a(s)| = e^{-\delta h_a}$$

is a real positive scale readout. By q-solenoid compatibility, such a readout must arise from the compact solenoid phase carrier rather than from a noncompact scale coordinate. The compact carrier has no nontrivial continuous positive scale character by Definition 9413.781. Thus $|U_a(s)| = 1$ for each zero-relevant displayed node. Since $h_a > 0$, the equality $e^{-\delta h_a} = 1$ gives $\delta = 0$. □

Definition 9413.812 (Resonance counting lower row)**.** *For a fixed split-prime tower and resonance windows* (A_n)*, the* resonance counting function *is*

$$N_\rho(n) := |A_n|.$$

In the pure split-prime case at a fixed depth, the tower count is

$$N_q(n) := f_n = [K_n : \mathbb{Q}],$$

because complete splitting gives one displayed local prime row $\mathfrak{q}_{n,i}$ *for each embedding degree slot. A resonance family has a* counting lower row *when there are constants* $c > 0$ *and* $j_0 : \mathsf{Nat}^{\uparrow}$ *such that*

$$N_\rho(n) \geq cN_q(n)$$

along the displayed fixed-depth split nodes for all sufficiently large n*.*

Theorem 9413.813 (Counting lower rows force off-half energy growth)**.** *Let a* q*-split resonance family have a counting lower row. If*

$$s = \frac{1}{2} + \delta + i\tau$$

and $\delta\#0$*, then there is a constant* $c_q > 0$ *such that, for all sufficiently large* n*,*

$$\mathcal{E}^{(q)}_{A_n}(s) \geq c_q\, N_q(n).$$

In particular, if $N_q(n) \to \infty$*, then*

$$\mathcal{E}^{(q)}_{A_n}(s) \to \infty.$$

Proof. The counting lower row supplies a fixed depth j_0 and a positive density constant c. Each displayed split node at that depth has

$$h_a = 2^{j_0} \log q,$$

so

$$d_a(s)^2 = \delta^2 (2^{j_0} \log q)^2.$$

Thus the contribution of the counted split nodes is at least

$$cN_q(n)\delta^2(2^{j_0} \log q)^2.$$

Set

$$c_q := c\delta^2(2^{j_0} \log q)^2.$$

Since q is prime and $\delta\#0$, this constant is positive. The displayed lower bound follows, and divergence follows when $N_q(n) \to \infty$. □

Definition 9413.814 (Solenoid upper row)**.** *For a zero-relevant* q*-split resonance family, a* solenoid upper row *is a packet consisting of:*

$$\mathsf{QSolenoidCompatible}(\rho, s), \qquad \mathsf{TailBound}(\rho), \qquad \mathsf{RoucheGap}(\rho),$$

and a finite constant B_s *with*

$$\forall n, \qquad \mathcal{E}^{(q)}_{A_n}(s) \leq B_s.$$

The compatibility row places the hidden registers in the compact q*-adic fibre of* Σ_q*. The tail and Rouché rows ensure that the same resonance family is zero-relevant for the completed readout, rather than merely visible in a finite projection.*

Definition 9413.815 (Zeta prime-count solenoid bound)**.** *The* zeta prime-count solenoid bound *is the source row*

$$\mathsf{ZetaPrimeCountSolenoidBound}_\zeta$$

asserting that every critical-strip zero coordinate s carries a zero-relevant fixed-prime resonance packet consisting of a rational prime q, a fixed split-prime tower for q, a stable dominant resonance family (A_n), a counting lower row, and a solenoid upper row.

Theorem 9413.816 (Prime-count lower and solenoid upper rows are conditionally sufficient)**.** *If*

$$\mathsf{ZetaPrimeCountSolenoidBound}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The prime-count solenoid bound supplies a fixed rational prime q, a split tower, a zero-relevant resonance family, a counting lower row, and a solenoid upper row. The upper row gives a finite constant B_s with

$$\mathcal{E}^{(q)}_{A_n}(s) \le B_s$$

for all n. Suppose $\delta \# 0$. By Definition 9413.813, the counting lower row gives

$$\mathcal{E}^{(q)}_{A_n}(s) \to \infty,$$

because the split-prime tower has $N_q(n) \to \infty$. This contradicts the uniform upper bound. Hence $\neg(\delta \# 0)$, and located fixed-half collapse gives $\delta = 0$. Therefore $\Re(s) = 1/2$, and the fixed-half section follows from Definition 9413.199. □

Corollary 9413.817 (The prime-count solenoid row is a source theorem)**.** *The zeta prime-count solenoid bound is not a fixed-half alias row.*

Proof. The row does not assert directly that a zero has $\delta = 0$, zero normal energy, or no normal hair. It asserts the existence of a zero-relevant source packet with two independent pieces of structure: a counting lower row supplied by split-prime growth, and a solenoid upper row supplied by compact fibre compatibility together with tail and Rouché control. Only after these two rows are combined does off-half displacement become impossible. Thus the row has the noncircular source shape isolated in Definition 9413.801. □

Corollary 9413.818 (Strong-triangle dominance explains the upper row)**.** *The p-adic strong triangle inequality supplies the finite dominance template behind the solenoid upper row.*

Proof. For a non-archimedean absolute value, if one term is strictly larger than all others, then the finite sum cannot vanish, as recorded in Definition 9413.773. Thus zero-relevant finite cancellation must occur at a boundary where dominant levels tie rather than at an arbitrary projected cancellation. This is the same proof-role shape as the quartic maximal-modulus boundary in Definition 9413.776. The solenoid upper row imports this dominance discipline into the fixed-prime tower setting: the compact q-adic fibre supplies legal hidden registers, while the Rouché and tail rows certify that the dominant finite family controls the completed zero island. □

Definition 9413.819 (Zeta fixed-prime tower Lee–Yang bound)**.** *The* zeta fixed-prime tower Lee–Yang bound *is the source row*

$$\mathsf{ZetaFixedPrimeTowerLeeYang}_{\zeta}$$

asserting that every critical-strip zero coordinate s carries a packet consisting of:

$$q : \mathsf{Prime}^{\uparrow}, \qquad \mathsf{FixedSplitPrimeTower}(q), \qquad (A_n),$$

with positive-density split support, a stable local decomposition

$$\Xi(s') = F_n(s') + T_n(s')$$

on a zero island around s, a Rouché boundary gap

$$|T_n| < |F_n|,$$

a q-solenoid-compatible phase lift, and a uniform normal-energy bound

$$\sup_n \mathcal{E}^{(q)}_{A_n}(s) < \infty.$$

Theorem 9413.820 (The fixed-prime tower bound is conditionally sufficient)**.** *If*

$$\mathsf{ZetaFixedPrimeTowerLeeYang}_{\zeta}$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The fixed-prime tower Lee–Yang bound supplies a rational prime q, a fixed split-prime tower, positive-density split support, and the bound

$$\sup_n \mathcal{E}^{(q)}_{A_n}(s) < \infty.$$

If $\delta \# 0$, then Definition 9413.809 gives

$$\mathcal{E}^{(q)}_{A_n}(s) \to \infty,$$

contradicting the displayed bound. Thus $\neg(\delta \# 0)$, and located fixed-half collapse gives $\delta = 0$. Hence $\Re(s) = 1/2$, and the fixed-half section follows from Definition 9413.199. □

Corollary 9413.821 (Fixed-prime towers refine the source gap)**.** *The fixed-prime tower route refines the dominant solenoid source gap by replacing an all-prime starting window with a single zero-relevant split prime whose tower generates the needed cofinal support.*

Proof. Definition 9413.803 identifies the remaining theorem as the construction of a dominant solenoid source from the completed zero row. The fixed-prime tower route proposes a more explicit source shape: first select the rational prime q, then construct the split tower, the positive-density node family, the local Rouché gap, the q-solenoid phase lift, and the far-end energy bound. The divergence theorem above shows why this is enough once the source is constructed: one split rational prime already yields unbounded normal energy off the fixed-half surface. □

9413.67 Dedekind Split-Tower Lift

The fixed-prime tower mechanism becomes more concrete when the zero-relevant carrier is not an abstract resonance tower but a tower of Dedekind zeta reads. The intended source theorem is then not a bare finite-energy assertion. It says that a Riemann zero can be lifted through a controlled Galois tower, that the fixed rational prime q splits into many local prime-ideal rows in that tower, and that the Dedekind zero islands are captured by solenoid-compatible Lee–Yang dominance data.

Definition 9413.822 (Admissible q-split Dedekind tower)**.** *An* admissible q-split Dedekind tower *is a fixed split-prime tower of Definition 9413.805 together with the additional rows*

$$\mathsf{Galois}(K_n/\mathbb{Q}) \qquad (n \in \mathsf{Nat}^{\uparrow}),$$

and

$$\mathsf{DedekindRead}(K_n, \zeta_{K_n}, \Lambda_{K_n}),$$

where ζ_{K_n} is the Dedekind zeta read and Λ_{K_n} is the completed Dedekind read. The Galois row is part of the source data: it places the tower in the range where the Riemann zeta factor is a visible factor of the Dedekind read and where quotient-entireness is supplied by the displayed tower ledger rather than by a non-Galois quotient assumption.

Definition 9413.823 (Dedekind quotient lift row)**.** *Let ρ be a critical-strip coordinate. A tower level K_n carries a* Dedekind quotient lift row at ρ *when there is a holomorphic local quotient row Ψ_n on a neighbourhood of ρ such that*

$$\zeta_{K_n}(s) = \zeta(s)\Psi_n(s)$$

on that neighbourhood. Write

$$\mathsf{DedekindLift}_{K_n}(\rho)$$

for the resulting row

$$\mathsf{ZetaZero}(\rho) \to \zeta_{K_n}(\rho) = 0.$$

Theorem 9413.824 (Dedekind quotient rows lift Riemann zeros)**.** *If K_n carries a Dedekind quotient lift row at ρ, then*

$$\mathsf{ZetaZero}(\rho) \to \zeta_{K_n}(\rho) = 0.$$

Proof. The quotient row gives a local factorization

$$\zeta_{K_n}(s) = \zeta(s)\Psi_n(s)$$

with Ψ_n holomorphic at ρ. If $\zeta(\rho) = 0$, then substitution gives

$$\zeta_{K_n}(\rho) = \zeta(\rho)\Psi_n(\rho) = 0.$$

□

Definition 9413.825 (q-split Dedekind normal energy)**.** *Let*

$$\rho = \frac{1}{2} + \delta + i\gamma$$

and let q split completely in K_n:

$$q\mathcal{O}_{K_n} = \mathfrak{q}_{n,1} \cdots \mathfrak{q}_{n,f_n}.$$

For each split prime-ideal row set

$$U_{n,i}(\rho) := q^{1/2-\rho}.$$

The normal defect is

$$d_{n,i}(\rho) := \log |U_{n,i}(\rho)| = -\delta \log q,$$

and the q-split Dedekind normal energy is

$$\mathcal{E}_{q,n}^{\mathrm{Ded}}(\rho) := \sum_{i=1}^{f_n} d_{n,i}(\rho)^2.$$

Theorem 9413.826 (Dedekind split counts amplify off-half energy)**.** *Let (K_n) be an admissible q-split Dedekind tower. If*

$$\rho = \frac{1}{2} + \delta + i\gamma$$

and $\delta \# 0$, then

$$\mathcal{E}_{q,n}^{\mathrm{Ded}}(\rho) = [K_n : \mathbb{Q}]\delta^2(\log q)^2$$

and therefore

$$\mathcal{E}_{q,n}^{\mathrm{Ded}}(\rho) \to \infty.$$

Proof. Complete splitting gives exactly

$$f_n = [K_n : \mathbb{Q}]$$

split prime-ideal rows above q, each with norm q. For each such row,

$$d_{n,i}(\rho)^2 = \delta^2(\log q)^2.$$

Summing over $i = 1, \ldots, f_n$ gives

$$\mathcal{E}_{q,n}^{\mathrm{Ded}}(\rho) = f_n\delta^2(\log q)^2 = [K_n : \mathbb{Q}]\delta^2(\log q)^2.$$

The tower degree tends to infinity, while the coefficient is positive when $\delta \# 0$ and q is prime. Hence the energy diverges. □

Definition 9413.827 (q-tower Lee–Yang dominance certificate)**.** *Let ρ be a Riemann critical-strip zero. A* q-tower Lee–Yang dominance certificate *for ρ consists of an admissible q-split Dedekind tower and, for each level n, a local zero island D_n around ρ with a decomposition*

$$\Lambda_{K_n}(s) = F_n(s) + T_n(s)$$

on a neighbourhood of D_n, together with the rows

$$\mathsf{DedekindLift}_{K_n}(\rho), \qquad \mathsf{DominantFamily}(F_n), \qquad \mathsf{RoucheGap}(F_n, T_n, D_n),$$

$$\mathsf{QSolenoidCompatible}(F_n, \rho), \qquad \exists B_\rho < \infty,\ \forall n,\ \mathcal{E}_{q,n}^{\mathrm{Ded}}(\rho) \le B_\rho.$$

The certificate states that the lifted Dedekind zero is not merely a projection zero: it is captured by a dominant local piece, protected by a Rouché gap, carried by the q-solenoid phase register, and bounded in normal energy along the same split-prime tower.

Definition 9413.828 (Dedekind q-tower Lee–Yang bound)**.** *The* Dedekind q-tower Lee–Yang bound *is the source row*

$$\mathsf{DedekindQTowerLeeYangBound}_\zeta$$

asserting that every Riemann critical-strip zero ρ admits a rational prime q, an admissible q-split Dedekind tower, and a q-tower Lee–Yang dominance certificate for ρ.

Theorem 9413.829 (Dedekind tower Lee–Yang bound is conditionally sufficient)**.** *If*

$$\mathsf{DedekindQTowerLeeYangBound}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate

$$\rho = \frac{1}{2} + \delta + i\gamma.$$

The Dedekind tower Lee–Yang bound supplies q, an admissible q-split Dedekind tower, and a certificate with a finite bound

$$\mathcal{E}^{\mathrm{Ded}}_{q,n}(\rho) \leq B_\rho$$

for all n. Suppose $\delta \# 0$. By Definition 9413.826, the same tower satisfies

$$\mathcal{E}^{\mathrm{Ded}}_{q,n}(\rho) \to \infty,$$

contradicting the bound. Thus $\neg(\delta \# 0)$, and located fixed-half collapse gives $\delta = 0$. Hence $\Re(\rho) = 1/2$, and the fixed-half section follows from Definition 9413.199. □

Corollary 9413.830 (The Dedekind lift is the zero-source bridge)**.** *The Dedekind tower route refines Definition 9413.815 by making the zero-relevant source carrier explicit.*

Concretized in §9016: Finite boundary packet for q-split Dedekind lift rows without Dedekind-cut or RH closure.

Proof. The fixed-prime route already separates the counting lower row from the solenoid upper row. The Dedekind lift adds the missing zero-source bridge: the Riemann zero is carried into each Dedekind read by the Galois quotient factorization, and the split prime q supplies $[K_n : \mathbb{Q}]$ local prime-ideal rows at level n. The remaining work is precisely the certificate data in Definition 9413.827: dominant local capture, Rouché isolation, solenoid compatibility, and the uniform normal-energy bound. □

Corollary 9413.831 (Explicit formula search direction)**.** *The natural analytic search surface for the Dedekind tower route is an explicit-formula comparison in which split-prime contributions, Dedekind zeros, discriminant terms, and archimedean factors appear in one ledger.*

Proof. The split-prime side supplies a lower row through Definition 9413.826. The admissibility side supplies global control through bounded root discriminant and the completed Dedekind read. An explicit formula is the standard analytic surface on which zeros, prime ideals, discriminant, and gamma factors can be compared in the same equation. Thus it is the appropriate place to try to prove the dominance and bounded-energy rows required by Definition 9413.827, rather than another reformulation of the fixed-half surface. □

9413.68 Zeckendorf Solenoid Selector

The fixed-prime and Dedekind split-tower routes choose a resonance source by selecting one rational prime and amplifying it through a tower. A different discrete source mechanism is available from the Axis-Zeckendorf side: use the no-adjacent Fibonacci grammar as a legal prime-family selector, read the selected primes as a prime-register ledger, and then ask whether the resulting family admits a solenoid phase lift and a Lee–Yang stability certificate. This is a selector route, not a direct proof of RH.

Definition 9413.832 (Prime ledger read)**.** *The* prime ledger *is the free integer register*

$$\mathcal{L}_{\mathbb{P}} := \bigoplus_{p \in \mathsf{Prime}^{\uparrow}} \mathbb{Z} e_p.$$

A positive integer

$$n = \prod_p p^{v_p(n)}$$

has ledger row

$$\nu(n) := \sum_p v_p(n) e_p.$$

For

$$s = \frac{1}{2} + \delta + i\tau,$$

the centered zeta channel on this ledger is

$$n^{1/2-s} = \exp\left(-\delta \sum_p v_p(n) \log p\right) \exp\left(-i\tau \sum_p v_p(n) \log p\right).$$

Definition 9413.833 (FRACTRAN ledger transition)**.** *A FRACTRAN instruction a/b acts on the prime ledger by the affine translation*

$$x \mapsto x + \nu(a) - \nu(b)$$

whenever the displayed state admits the instruction. Thus FRACTRAN supplies a finite, auditable dynamics on the same prime-exponent ledger used by Euler factors. It does not choose the zero-relevant resonance family; it supplies the ledger language in which such a choice can be tracked.

Definition 9413.834 (Zeckendorf–Godel prime selector)**.** *Let*

$$X_m := \{z \in \{0,1\}^m : \ z_k z_{k+1} = 0\}$$

be the no-adjacent Zeckendorf window of length m, and let

$$X_\varphi := \{z \in \{0,1\}^{\mathbb{N}} : \ z_k z_{k+1} = 0\}$$

be the golden-mean selector space. The Zeckendorf–Godel selector sends

$$z \mapsto S_z := \{p_k : z_k = 1\}, \qquad g_z := \prod_k p_k^{z_k}.$$

For finite m, the window cardinality is

$$|X_m| = F_{m+2}$$

as in Definition 9347.16. The selector is legal because the no-adjacent word is recoverable from the squarefree non-adjacent prime code by unique Zeckendorf expansion and unique prime factorization.

Definition 9413.835 (Golden-cofinal selector)**.** *A selector $z \in X_\varphi$ is* golden-cofinal *when*

$$\sum_k z_k (\log p_k)^2 = \infty.$$

This is the cofinality row consumed by the zeta energy argument. It is independent of the zero predicate and belongs to the selector's prime-ledger growth.

Definition 9413.836 (Zeckendorf zeta normal energy)**.** *For $z \in X_\varphi$, define the Zeckendorf zeta normal energy by*

$$\mathcal{E}_z(s) := \sum_k z_k \left(\log |p_k^{1/2-s}|\right)^2.$$

For

$$s = \frac{1}{2} + \delta + i\tau,$$

this reads

$$\mathcal{E}_z(s) = \delta^2 \sum_k z_k (\log p_k)^2.$$

Theorem 9413.837 (Golden-cofinal selectors detect off-half energy)**.** *If z is golden-cofinal and*

$$s = \frac{1}{2} + \delta + i\tau$$

with $\delta \# 0$, then

$$\mathcal{E}_z(s) = \infty.$$

Proof. By Definition 9413.836,

$$\mathcal{E}_z(s) = \delta^2 \sum_k z_k (\log p_k)^2.$$

The selector is golden-cofinal, so the sum diverges. Since $\delta \# 0$, multiplication by the positive factor δ^2 preserves divergence. □

Definition 9413.838 (Euler-log solenoid factorization)**.** *Let S_z be a Zeckendorf–Godel prime family. An* Euler-log solenoid factorization *for S_z is a weight map*

$$w : \mathcal{L}_{S_z} \to \mathbb{Z}[S_z^{-1}]$$

and a solenoid phase character

$$\sigma : \mathbb{Z}[S_z^{-1}] \to \mathbb{T}$$

such that the zeta phase read

$$e_p \mapsto e^{-i\tau \log p}$$

factors as

$$\mathcal{L}_{S_z} \xrightarrow{w} \mathbb{Z}[S_z^{-1}] \xrightarrow{\sigma} \mathbb{T}$$

on the zero-relevant family. This is an explicit bridge requirement: the multiplicative Euler-log phase does not automatically equal the additive localization phase of

$$\Sigma_{S_z} := \widehat{\mathbb{Z}[S_z^{-1}]}.$$

Definition 9413.839 (Zeckendorf solenoid Lee–Yang certificate)**.** *For a critical-strip zero coordinate s, a* Zeckendorf solenoid Lee–Yang certificate *consists of:*

$$z \in X_\varphi, \qquad \mathsf{GoldenCofinal}(z), \qquad \mathsf{Captures}(z, s),$$

$$\mathsf{EulerLogSolenoidFactorization}(z, s), \qquad \mathsf{ZeckendorfLeeYangStability}(z, s),$$

and a finite bound

$$\mathcal{E}_z(s) \le B_s < \infty.$$

The capture row says that the selected prime family is the stable zero-relevant resonance family. The factorization row places the zeta phase into the solenoid carrier. The stability row is the finite Lee–Yang or hard-core path-model input inherited from the no-adjacent Fibonacci grammar, after the prime weights and the zeta phase have been accounted for.

Definition 9413.840 (Zeta Zeckendorf solenoid bound)**.** *The* zeta Zeckendorf solenoid bound *is the source row*

$$\mathsf{ZetaZeckendorfSolenoidBound}_\zeta$$

asserting that every critical-strip zeta zero s carries a Zeckendorf solenoid Lee–Yang certificate.

Theorem 9413.841 (The Zeckendorf solenoid bound is conditionally sufficient)**.** *If*

$$\mathsf{ZetaZeckendorfSolenoidBound}_\zeta$$

holds, then the RH fixed-half section exists.

Proof. Let $z_0 : \mathcal{Z}_\zeta$ have complex coordinate

$$s = \frac{1}{2} + \delta + i\tau.$$

The Zeckendorf solenoid bound supplies a certificate with selector $z \in X_\varphi$, golden cofinality, and a finite bound

$$\mathcal{E}_z(s) \le B_s.$$

If $\delta \# 0$, then Definition 9413.837 gives

$$\mathcal{E}_z(s) = \infty,$$

contradicting the bound. Hence $\neg(\delta \# 0)$, so located fixed-half collapse gives $\delta = 0$. Therefore $\Re(s) = 1/2$, and the fixed-half section follows from Definition 9413.199. □

Corollary 9413.842 (Selector and split-tower routes are complementary)**.** *The Zeckendorf selector route and the fixed split-prime tower route are different source mechanisms for the same solenoid Lee–Yang proof role.*

Proof. The split-tower route fixes one rational prime q and obtains cofinal growth from its splitting into many prime-ideal rows, as in Definition 9413.826. The Zeckendorf route instead chooses a legal prime family S_z by the golden-mean grammar and obtains cofinal growth from Definition 9413.835. In both cases the downstream proof role is the same: a counting or cofinality lower row makes off-half normal energy unbounded, while solenoid compatibility and Lee–Yang stability are expected to supply the zero-relevant upper row. □

Corollary 9413.843 (The three open rows)**.** *The Zeckendorf solenoid route leaves three source rows open: capture, Euler-log solenoid factorization, and prime-weighted Zeckendorf Lee–Yang stability.*

Proof. Golden cofinality is a selector-growth row, and the energy divergence calculation is Definition 9413.837. What is not supplied by the zero predicate is the construction of a selector that actually captures the completed zero, the factorization of the Euler-log phase through a solenoid-compatible weight map, and the Lee–Yang stability theorem for the prime-weighted no-adjacent family. These are source claims, not aliases for $\delta = 0$. □

9413.69 Conditional RH Sufficiency Boundary

The preceding sections isolate two equivalent proof roles. The first is signature-theoretic: the zeta zero source has no independent mirror-odd observable beyond its atom, spectral-time, and symmetry-slot signature. The second is prime-sieve analytic: a finite prime-window mirror-odd defect cannot coexist with a completed zeta zero ledger. Each role is sufficient for the RH fixed-half section. Neither role is supplied by the definition of the zero predicate itself.

Definition 9413.844 (Zeta zero source as the RH input carrier)**.** *The RH input carrier is the critical-strip zero source*

$$\mathcal{Z}_\zeta := \sum_{s:\mathsf{Complex}^\uparrow} (\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)) .$$

The RH output carrier is the fixed-half lift

$$\mathcal{H}_\zeta := \sum_{z:\mathcal{Z}_\zeta} \mathsf{OnCritLine}(\pi_s z),$$

where $\pi_s z$ is the complex coordinate of the zero-source row.

Definition 9413.845 (Conditional sufficiency row)**.** *A row P_ζ is* conditionally sufficient for the RH section *when, together with the structural zero-ledger, mirror, same-source, and located fixed-half rows already displayed in the RH route, it gives a section*

$$\Phi_P : \mathcal{Z}_\zeta \to \mathcal{H}_\zeta.$$

This is a proof-role classification. It does not assert that P_ζ has been proved without further zeta-specific input.

Definition 9413.846 (Core-completeness sufficiency row)**.** *The* core-completeness sufficiency row *is the packet*

$$\mathsf{CoreComplete}(\mathcal{Z}_\zeta) + \mathsf{MirrorEven}(\Sigma_Z) + \mathsf{MirrorOdd}(\delta) + \mathsf{ZeroObservable}(\delta),$$

where Σ_Z is the zero ontic signature and

$$\delta(s) := \Re(s) - \frac{1}{2}.$$

The first component means that every public zero observable in the displayed class factors through Σ_Z.

Theorem 9413.847 (Core completeness and mirror oddness are sufficient)**.** *The core-completeness sufficiency row is conditionally sufficient for the RH fixed-half section.*

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. Core completeness applied to the public zero observable δ gives a factorization

$$\delta(s) = f(\Sigma_Z(s)).$$

Mirror closure supplies the companion zero-source row at $J(s)$. The mirror-even signature row gives

$$\Sigma_Z(J(s)) = \Sigma_Z(s),$$

and therefore

$$\delta(J(s)) = f(\Sigma_Z(J(s))) = f(\Sigma_Z(s)) = \delta(s).$$

The direct mirror calculation gives

$$\delta(J(s)) = -\delta(s).$$

Thus $\delta(s) = -\delta(s)$, hence $\delta(s) = 0$ in the characteristic-zero real readout. By Definition 9413.313, this is the fixed-half row for s. The construction is exactly the section shape of Definition 9413.199. □

Corollary 9413.848 (Core completeness implies the RH predicate conditionally)**.** *Under the structural rows of Definition 9413.847, core completeness of the zeta zero source implies the BEDC RH predicate.*

Proof. The theorem supplies a lift

$$\Phi : \mathcal{Z}_\zeta \to \mathcal{H}_\zeta.$$

Unfolding Definition 9413.844, this is a total assignment sending each row

$$\mathsf{ZetaZero}(s) \times \mathsf{InCritStrip}(s)$$

to $\mathsf{OnCritLine}(s)$. This is the RH predicate as identified by Definition 9413.199. □

Definition 9413.849 (Prime-sieve sufficiency row)**.** *The* prime-sieve sufficiency row *is*

$$\mathsf{PrimeSieveOddExactness}_\zeta,$$

the finite-sieve exactness target of Definition 9413.477. It asserts that a nonzero finite prime-window mirror-odd defect cannot coexist with a critical-strip zero row in the completed zeta carrier.

Theorem 9413.850 (Prime-sieve odd exactness is sufficient)**.** *Assume prime-sieve odd exactness and the located fixed-half collapse row*

$$\neg(\delta(s)\#0) \to \delta(s) = 0.$$

Then the RH fixed-half section exists.

Proof. Let $z : \mathcal{Z}_\zeta$ have complex coordinate s. It suffices to exclude $\delta(s)\#0$. Suppose $\delta(s)\#0$. Choose any nonempty finite prime window S. By Definition 9413.473,

$$D_S(s)\#0.$$

Hence the finite-window existential defect row in Definition 9413.477 holds. Applying $\mathsf{PrimeSieveOddExactness}_\zeta$ to the zero row, the critical-strip row, and this finite defect gives $\bot$. Therefore

$$\neg(\delta(s)\#0).$$

The located fixed-half collapse row gives $\delta(s) = 0$, and Definition 9413.313 turns this into $\mathsf{OnCritLine}(s)$. Varying z gives the desired section. □

Corollary 9413.851 (Prime-sieve exactness implies the RH predicate conditionally)**.** *Under located fixed-half collapse, prime-sieve odd exactness implies the BEDC RH predicate.*

Proof. Definition 9413.850 constructs the fixed-half section. The equivalence with the BEDC RH predicate is Definition 9413.199. □

Theorem 9413.852 (The conditional rows are the unclosed content)**.** *The preceding sufficiency theorems do not prove the unconditional RH predicate. Their unclosed content is the zeta-specific row*

$$\mathsf{CoreComplete}(\mathcal{Z}_\zeta)$$

or, in finite-sieve analytic form,

$$\mathsf{PrimeSieveOddExactness}_\zeta.$$

Proof. The zero predicate supplies a zero ledger and the critical-strip row supplies the input domain. The fixed-surface readout equivalences of Definition 9413.464 identify the target surface, but they do not place zero rows on that surface. Core completeness would place every public zero observable, including δ, behind the mirror-even zero signature. Prime-sieve odd exactness would exclude the finite prime-window defect generated by any nonzero δ. These are exactly the remaining rows identified by Definition 9413.461 and Definition 9413.484. □

Definition 9413.853 (Hidden normal-coordinate counterform)**.** *The* hidden normal-coordinate counterform *is a hypothetical zero-source pair*

$$s = \frac{1}{2} + \delta + i\gamma, \qquad J(s) = \frac{1}{2} - \delta + i\gamma, \qquad \delta \# 0,$$

such that

$$\Sigma_Z(J(s)) = \Sigma_Z(s)$$

but δ does not factor through Σ_Z. It is the form of a fourth observable ledger separating mirror companions after the displayed zero signature has already agreed.

Theorem 9413.854 (The hidden normal coordinate is the circularity boundary)**.** *Any route that only rewrites the fixed-half surface in mirror, prime, or Laplace coordinates leaves the hidden normal-coordinate counterform unexcluded.*

Proof. The equalities

$$\Re(s) = \frac{1}{2}, \qquad |p^{1/2-s}| = 1, \qquad J(s) = s$$

describe the fixed surface. For the counterform, these rows fail precisely because $\delta \# 0$. Rewriting their equivalence does not contradict the zero ledger, since the zero ledger has not yet been linked to the absence of δ. The contradiction appears only after either core completeness forces δ to factor through the mirror-even signature, or prime-sieve odd exactness makes the finite prime-window defect zero-incompatible. □

Corollary 9413.855 (Zero relevance is the proof load)**.** *The proof load of the RH route is zero relevance of the mirror-odd defect, not visibility of that defect.*

Proof. Visibility is supplied by finite prime windows through Definition 9413.473. The quartic spectral carrier shows the same distinction: branch visibility requires a dominance and interference ledger before it becomes an endpoint zero law, as recorded in Definition 9413.485. For zeta, the corresponding zero-relevance row is prime-sieve odd exactness, equivalently the no-fourth-ledger or defect-zero incompatibility boundary. □

9413.70 BEDC–Langlands Closure Packet

The Langlands interface is read here as a unified-reality packet with two visible projections: a Galois–Frobenius projection and an automorphic–Hecke projection. The section records a conditional closure theorem. It proves the detection, uniqueness, and no-hidden-residual shape of the packet. It does not assert the existence theorems, trace formula comparisons, local classifications, endoscopic transfers, geometric equivalences, or functorial lifts required by the full Langlands programme.

Definition 9413.856 (BEDC–Langlands packet)**.** *Let F be a global field, let G be a connected reductive group, and let ${}^LG$ be the associated L-group. A BEDC–Langlands packet is a certificate row*

$$\mathcal{O}^{\uparrow} = (\pi, \phi, \Lambda_{\mathcal{O}}, N_{\mathcal{O}})$$

where π is an automorphic-side object for $G(\mathbb{A}_F)$,

$$\phi : W_F' \longrightarrow {}^LG$$

is a parameter-side object, $\Lambda_{\mathcal{O}}$ is the ledger for ramification, monodromy, endoscopy, continuous spectrum, boundary, nilpotent support, and functorial transport, and $N_{\mathcal{O}}$ is the NameCert-style source, pattern, classifier, stability, and ledger surface. The packet is a two-projection carrier, not an assertion that all existence and transfer problems have been solved.

Definition 9413.857 (Hecke–Frobenius finite detector)**.** *For an unramified place v and a finite-dimensional representation*

$$r : {}^LG \longrightarrow GL(V_r),$$

the Hecke–Frobenius finite detector (v, r) reads the automorphic side by

$$\lambda_{v,r}(\pi) := \operatorname{tr} r(s_v(\pi)),$$

where $s_v(\pi)$ is the Satake parameter, and reads the parameter side by

$$a_{v,r}(\phi) := \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Its mismatch residual is

$$D_{v,r}(\pi, \phi) := \lambda_{v,r}(\pi) - a_{v,r}(\phi).$$

Definition 9413.858 (Langlands ledger residual)**.** *A Langlands ledger residual is a non-finite-detector obstruction carried by one of the displayed boundary rows:*

$$\mathsf{Ramification},\quad \mathsf{Monodromy},\quad \gamma/\epsilon\text{-}factor,\quad \mathsf{Endoscopy},\quad \mathsf{ContinuousSpectrum},$$

$$\mathsf{ArthurSL}_2,\quad \mathsf{Boundary},\quad \mathsf{NilpotentSupport},\quad \mathsf{SingularSupport}.$$

Such a residual is admissible only as recorded scope data, discharged data, or explicitly not-claimed boundary. It is not hidden equality between the two projections.

Definition 9413.859 (No-hidden Langlands residual)**.** *A BEDC–Langlands packet satisfies no-hidden Langlands residual when every nonzero mismatch between the Galois–Frobenius and automorphic–Hecke projections has one of two addresses: either some finite detector (v, r) has*

$$D_{v,r}(\pi, \phi) \neq 0,$$

or the obstruction is recorded in the Langlands ledger of Definition 9413.858.

Theorem 9413.860 (Tannakian finite detectors separate semisimple classes)**.** *Let $g, h \in {}^L G$ be semisimple elements. If*

$$\operatorname{tr} r(g) = \operatorname{tr} r(h)$$

for every finite-dimensional representation r of ${}^L G$, then g and h have the same semisimple conjugacy class.

Proof. Characters of finite-dimensional representations are regular class functions and generate the coordinate algebra of the semisimple conjugacy quotient

$${}^L G /\!/ \operatorname{conj}.$$

Equality of all character values therefore says that g and h define the same point of that quotient. For semisimple elements, the closed orbit attached to a quotient point is the semisimple conjugacy class. Thus the two semisimple conjugacy classes agree. □

Theorem 9413.861 (Chebotarev–Tannakian uniqueness)**.** *Let $\phi, \phi' : W_F' \to {}^L G$ be semisimple parameter rows. If for almost all unramified places v and every finite-dimensional representation r,*

$$\operatorname{tr} r(\phi(\mathrm{Frob}_v)) = \operatorname{tr} r(\phi'(\mathrm{Frob}_v)),$$

then the semisimplified parameter rows have the same Frobenius readout; equivalently, they agree up to the usual semisimple parameter classification on the displayed scope.

Proof. For each displayed unramified v, the preceding theorem gives equality of the semisimple conjugacy classes of

$$\phi(\mathrm{Frob}_v) \qquad \text{and} \qquad \phi'(\mathrm{Frob}_v).$$

Chebotarev density makes the Frobenius conjugacy classes dense in the global Galois source on the unramified scope. Since the parameter rows are semisimple and the character readouts agree on this dense Frobenius set, the semisimplified parameter readouts agree on the declared scope. □

Theorem 9413.862 (Closed Langlands packets have no hidden mismatch)**.** *Let $\mathcal{O}^{\uparrow} = (\pi, \phi, \Lambda_{\mathcal{O}}, N_{\mathcal{O}})$ be a BEDC–Langlands packet with three-properties ledger discipline and no-hidden Langlands residual. If the packet is closed against unresolved Langlands ledger rows, then*

$$D_{v,r}(\pi, \phi) = 0$$

for every unramified finite detector (v, r) outside the declared bad ledger.

Proof. Assume that for some unramified finite detector outside the declared bad ledger,

$$D_{v,r}(\pi, \phi) \neq 0.$$

Then the detector itself supplies a finite mismatch address. By ledger discipline, this mismatch must be recorded as a finite Hecke–Frobenius mismatch row. If it is not recorded, the packet violates the ledger property. If it is recorded but unresolved, the packet is not closed against unresolved Langlands ledger rows. Both alternatives contradict the hypotheses. Therefore no such finite mismatch exists. □

Corollary 9413.863 (Parameter uniqueness from a closed automorphic packet)**.** *If (π, ϕ) and (π, ϕ') are closed BEDC–Langlands packets for the same automorphic-side row and both satisfy the no-hidden residual condition on the same unramified scope, then ϕ and ϕ' have the same semisimplified parameter readout on that scope.*

Proof. By Definition 9413.862, both parameter rows match the same Hecke readouts

$$\lambda_{v,r}(\pi)$$

for almost all unramified v and every finite-dimensional r. Hence their Frobenius character readouts agree. Apply Definition 9413.861. □

Definition 9413.864 (BEDC reciprocity certificate)**.** *For a class $\mathcal{C}$ of automorphic-side rows, a BEDC reciprocity certificate assigns to every $\pi \in \mathcal{C}$ a parameter row*

$$\phi_\pi : W'_F \to {}^L G$$

such that (π, ϕ_π) forms a closed BEDC–Langlands packet, satisfies no-hidden Langlands residual, and carries all bad-place and non-finite residual rows in its ledger.

Theorem 9413.865 (Conditional BEDC reciprocity)**.** *If a class $\mathcal{C}$ carries a BEDC reciprocity certificate, then each $\pi \in \mathcal{C}$ has a unique semisimplified parameter readout ϕ_π^{ss} on the certified unramified scope, and*

$$\operatorname{tr} r(s_v(\pi)) = \operatorname{tr} r(\phi_\pi(\mathrm{Frob}_v))$$

for almost all unramified places v and all finite-dimensional representations r.

Proof. Existence of the parameter row is part of the reciprocity certificate. The equality of finite Hecke and Frobenius readouts follows from Definition 9413.862. Uniqueness of the semisimplified parameter readout follows from Definition 9413.863. □

Definition 9413.866 (BEDC functoriality certificate)**.** *Let*

$$\eta : {}^L H \longrightarrow {}^L G$$

be an L-group homomorphism. A BEDC functoriality certificate for η assigns to each closed packet (π_H, ϕ_H) a closed packet (π_G, ϕ_G) such that

$$\phi_G = \eta \circ \phi_H$$

on the certified parameter scope, and every transfer, endoscopic, ramified, continuous-spectrum, or boundary residual enters the functoriality ledger.

Theorem 9413.867 (Conditional BEDC functoriality)**.** *If $\eta : {}^L H \to {}^L G$ carries a BEDC functoriality certificate, then for the associated closed packet (π_G, ϕ_G) and every unramified finite detector (v, r) outside the transfer ledger,*

$$\lambda_{v,r}(\pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v))).$$

Proof. The functoriality certificate gives $\phi_G = \eta \circ \phi_H$. Since (π_G, ϕ_G) is closed and satisfies the no-hidden residual condition on the certified scope, Definition 9413.862 gives

$$\lambda_{v,r}(\pi_G) = \operatorname{tr} r(\phi_G(\mathrm{Frob}_v)).$$

Substituting $\phi_G = \eta \circ \phi_H$ gives the displayed equality. □

Definition 9413.868 (Geometric Langlands closure certificate)**.** *For a smooth complete curve X and dual group $\check{G}$, a geometric Langlands closure certificate is a categorical equivalence row between the automorphic category and the spectral local-system category together with ledgers for Hecke modification time, spectral local-system readout, nilpotent singular support, stack boundary, Eisenstein and constant-term rows, and endoscopic residuals.*

Theorem 9413.869 (Geometric Langlands closure has no hidden residual)**.** *If a geometric Langlands equivalence row carries the closure certificate of Definition 9413.868, then a closed geometric Langlands packet has no hidden residual outside the displayed Hecke, singular-support, boundary, Eisenstein, and endoscopic ledgers.*

Proof. A residual visible under Hecke modification is a finite categorical detector and must be recorded in the Hecke ledger. A residual not visible there must occur in one of the displayed categorical boundary ledgers: singular support, stack boundary, Eisenstein or constant term, or endoscopy. Omitting the row violates ledger discipline; carrying it unresolved violates closedness. Hence no hidden residual remains in a closed packet. □

Theorem 9413.870 (Quadratic splitting is the abelian seed)**.** *For an odd prime p,*

$$p = a^2 + b^2 \quad \Longleftrightarrow \quad p \equiv 1 \pmod 4 \quad \Longleftrightarrow \quad p \text{ splits in } \mathbb{Q}(i).$$

Equivalently, the quadratic character χ_{-4} has $\chi_{-4}(p) = 1$.

Proof. Fermat's two-square theorem gives the equivalence between representation by $a^2 + b^2$ and $p \equiv 1 \pmod 4$ for odd primes. In the quadratic extension $\mathbb{Q}(i)/\mathbb{Q}$, a prime $p \neq 2$ splits exactly when -1 is a square modulo p, which is equivalent to $p \equiv 1 \pmod 4$. The same splitting law is recorded by the quadratic character χ_{-4}. □

Corollary 9413.871 (Langlands as a two-projection closure principle)**.** *On any category where the reciprocity, functoriality, local, ramified, continuous-spectrum, endoscopic, and boundary certificates are supplied, the Langlands correspondence can be read as the statement that Galois–Frobenius and automorphic–Hecke data are two projections of one closed three-properties packet.*

Proof. The finite detectors match the two projections outside the declared ledger by Definition 9413.865 and Definition 9413.867. The Tannakian and Chebotarev rows give parameter uniqueness, while the ledger rows prevent ramified, endoscopic, continuous, boundary, or categorical residuals from being hidden as equality. Therefore the correspondence is exactly a closed two-projection packet on the certified scope. □

Theorem 9413.872 (Langlands closure boundary)**.** *The BEDC–Langlands packet closes uniqueness, finite detection, and conditional reciprocity or functoriality after the relevant existence certificates have been supplied. It does not close the existence of parameters, automorphic transfers, trace formula comparisons, local classification, endoscopic classification, continuous-spectrum control, Shimura or cohomological constructions, or geometric categorical equivalences.*

Proof. All positive conclusions above use an explicit existence or transfer certificate as input: reciprocity for ϕ_π, functoriality for π_G, and categorical equivalence for the geometric packet. The closure arguments then exclude hidden mismatch residuals and give uniqueness or equality of finite readouts. They never construct those certificates. The listed existence and comparison problems therefore remain separate certificate obligations. □

9413.71 BEDC–Langlands Existence Bridge

The closure packet of section 9413.70 isolates detection and uniqueness: once a paired automorphic–Galois packet is present and closed, a Hecke–Frobenius mismatch cannot be hidden. The existence side needs a generating bridge. In the analytic form this bridge is a trace-formula certificate. In the geometric form it is a certified geometric Langlands functor or categorical equivalence.

Definition 9413.873 (Allowed parameter packet)**.** *An allowed Langlands parameter packet is a row*

$$\phi^{\uparrow} = (\phi, \Lambda_\phi, N_\phi)$$

where

$$\phi : W_F' \longrightarrow {}^L G$$

is continuous on the declared source, semisimple on the certified unramified scope, and locally acceptable on the displayed local rows. The ledger Λ_ϕ records ramification, monodromy, γ/ϵ factors, Arthur SL_2 data when present, and every bad-place boundary used by the packet.

Definition 9413.874 (Automorphic realization)**.** *Let $\phi^{\uparrow}$ be an allowed parameter packet. An automorphic realization of $\phi^{\uparrow}$ is an automorphic object or automorphic packet row π such that for almost all certified unramified places v and every finite-dimensional representation $r : {}^L G \to GL(V_r)$,*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Equivalently, the Satake parameter of π and the Frobenius readout of ϕ define the same semisimple point in the conjugacy quotient at those places.

Definition 9413.875 (Existence residual)**.** *For an allowed parameter packet $\phi^{\uparrow}$, the existence residual $E(\phi)$ is the assertion that no automorphic realization of $\phi^{\uparrow}$ exists on the declared scope. A closed BEDC Langlands existence proof must show that any such residual is detected by a finite trace window or is recorded in a ramified, endoscopic, continuous-spectrum, boundary, or compactness ledger.*

Definition 9413.876 (Parameter projection test function)**.** *Let S be a finite set of unramified places. A test function $f_{\phi,S}$ is a parameter projection test function for $\phi^{\uparrow}$ when it belongs to the permitted restricted tensor product of local test spaces and its unramified local factors lie in the spherical Hecke algebras*

$$\mathcal{H}_v(G, K_v).$$

Under the Satake transform, these factors cut out the finite list of semisimple Hecke readouts prescribed by

$$\phi(\mathrm{Frob}_v) \qquad (v \in S).$$

Thus $\operatorname{tr} \pi(f_{\phi,S}) \neq 0$ only for automorphic rows whose finite Hecke readouts match the parameter on the window S, except for residual rows explicitly named by the test-function ledger.

Definition 9413.877 (Stable trace formula certificate)**.** *A stable trace formula certificate for G is a ledgered equality*

$$I^G_{\mathrm{spec}}(f) = I^G_{\mathrm{geom}}(f)$$

for all admitted test functions f, together with decomposition ledgers for discrete spectrum, continuous spectrum, endoscopy, ramification, orbital integral normalization, multiplicity, boundary terms, and transfer factors. The certificate is closed on a scope only when every term not intended as the target contribution has either been separated, cancelled, or recorded as an open ledger row.

Theorem 9413.878 (Trace formula detects automorphic realization)**.** *Let $\phi^{\uparrow}$ be an allowed parameter packet and let $f_{\phi,S}$ be a parameter projection test function. Assume:*

- $I^G_{\mathrm{geom}}(f_{\phi,S}) \neq 0$*;*
- *all continuous, endoscopic, boundary, ramified, and normalization terms not belonging to the target contribution are separated or ledgered;*
- *the stable trace formula certificate holds for $f_{\phi,S}$.*

Then there is an automorphic row π with

$$\operatorname{tr} \pi(f_{\phi,S}) \neq 0.$$

Proof. The geometric side is nonzero by hypothesis. The trace formula gives

$$I^G_{\mathrm{spec}}(f_{\phi,S}) = I^G_{\mathrm{geom}}(f_{\phi,S}) \neq 0.$$

The spectral side decomposes into discrete automorphic rows, continuous spectrum, endoscopic transfers, and the displayed boundary ledgers. All non-target residual terms have been separated or ledgered, so a nonzero spectral contribution must come from some automorphic row π with

$$\operatorname{tr} \pi(f_{\phi,S}) \neq 0.$$

By the projection property of $f_{\phi,S}$, this row matches the parameter on the finite window S. □

Corollary 9413.879 (Finite-window automorphic realization)**.** *Under the hypotheses of Definition 9413.878, the parameter packet $\phi^{\uparrow}$ has an automorphic realization on the finite window S.*

Proof. The theorem supplies π with nonzero trace against the projection test function. The definition of Definition 9413.876 then forces the prescribed Hecke–Frobenius readouts on S, outside any ledgered residual rows. □

Definition 9413.880 (Hecke window tower)**.** *A Hecke window tower for $\phi^{\uparrow}$ is an increasing sequence of finite unramified sets*

$$S_1 \subset S_2 \subset \cdots$$

whose union is the declared almost-everywhere unramified scope, together with parameter projection test functions f_{ϕ,S_n} for every window. A tower automorphic read is a sequence of automorphic rows (π_n) such that each π_n matches ϕ on S_n.

Definition 9413.881 (Cofinal trace compactness certificate)**.** *A cofinal trace compactness certificate for a Hecke window tower says that every tower automorphic read has one of two outcomes:*

- *it stabilizes or converges to an automorphic row π_ϕ whose finite Hecke readouts match ϕ on the cofinal unramified scope;*
- *its failure to stabilize is recorded in the automorphic tail ledger, with continuous-spectrum, endoscopic, boundary, multiplicity, or escape-of-mass data displayed.*

Theorem 9413.882 (Trace formula existence certificate)**.** *Let $\phi^{\uparrow}$ be an allowed parameter packet. Assume:*

- *a Hecke window tower with projection test functions f_{ϕ,S_n} is displayed;*
- *each window satisfies the hypotheses of Definition 9413.878;*

- *a cofinal trace compactness certificate is displayed;*
- *closed accepted objects carry no unresolved automorphic tail ledger.*

Then $\phi^{\uparrow}$ has an automorphic realization π_ϕ on the declared almost-everywhere unramified scope.

Proof. For every n, Definition 9413.878 supplies an automorphic row π_n matching ϕ on S_n. The cofinal compactness certificate says that the tower (π_n) either stabilizes or converges to an automorphic row π_ϕ, or its failure enters the automorphic tail ledger. The latter alternative is not compatible with closedness when the tail remains unresolved. Therefore the closed case yields π_ϕ, and cofinality of the sets S_n gives the required almost-everywhere matching. □

Theorem 9413.883 (BEDC reciprocity existence)**.** *Let $\phi^{\uparrow}$ be an allowed parameter packet. If the trace formula existence certificate holds for $\phi^{\uparrow}$, the no-hidden Langlands residual condition holds, and the accepted Langlands packet is closed against unresolved residual ledgers, then there is a closed automorphic realization π_ϕ satisfying*

$$\lambda_{v,r}(\pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

for almost all unramified v and all finite-dimensional representations r on the certified scope.

Proof. The trace formula existence certificate supplies an automorphic row π_ϕ matching ϕ on cofinal finite windows. If any mismatch remains on the certified scope, the no-hidden residual condition of Definition 9413.859 gives a finite Hecke–Frobenius detector or a Langlands ledger residual. A finite detector residual must be recorded or else the ledger property is violated; if it is recorded but unresolved, closedness fails. The same argument applies to ramified, endoscopic, continuous-spectrum, boundary, or compactness residuals. Hence the closed packet has no residual mismatch, and the displayed equality follows. □

Definition 9413.884 (Transfer trace certificate)**.** *Let*

$$\eta : {}^L H \longrightarrow {}^L G$$

be an L-group homomorphism. A transfer trace certificate for η assigns to every admitted H-side test function f_H an admitted G-side test function f_G such that the stable orbital integrals and spectral target contributions match under η, while all transfer factor, endoscopic, ramified, continuous-spectrum, and boundary residuals enter the functorial transfer ledger.

Theorem 9413.885 (Functoriality from transfer trace)**.** *Let (π_H, ϕ_H) be a closed BEDC–Langlands packet for H, and let $\eta : {}^L H \to {}^L G$ carry a transfer trace certificate. If the G-side trace formula existence certificate holds for*

$$\phi_G := \eta \circ \phi_H,$$

and the no-hidden Langlands residual condition holds on the target scope, then there is a closed G-side automorphic row π_G with

$$\lambda_{v,r}(\pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v)))$$

for almost all certified unramified v and all finite-dimensional $r : {}^L G \to GL(V_r)$.

Proof. Apply Definition 9413.883 to the G-side parameter packet $\phi_G = \eta \circ \phi_H$. The transfer trace certificate records the trace-formula compatibility and all transfer residuals. The existence theorem supplies π_G, and the no-hidden-residual argument removes every unclosed mismatch. Substituting $\phi_G = \eta \circ \phi_H$ gives the displayed finite-detector equality. □

Definition 9413.886 (Geometric functor certificate)**.** *For a smooth complete curve X and dual group $\check{G}$, a geometric functor certificate is a functor row*

$$\mathsf{GL}_G : \mathrm{IndCoh}_{\mathrm{Nilp}}(\mathrm{LocSys}_{\check{G}}) \longrightarrow \mathcal{D}\text{-}\mathit{Mod}(\mathrm{Bun}_G)$$

that is an equivalence on the declared scope, or is fully faithful and essentially surjective on the target subcategory, and preserves the Hecke modification time row, spectral local-system readout, nilpotent singular support, stack boundary, Eisenstein and constant-term rows, and endoscopic ledger.

Theorem 9413.887 (Geometric functor gives automorphic existence)**.** *If a geometric functor certificate is displayed, then every spectral object in its certified source scope has an automorphic image with the corresponding Hecke eigensheaf readout, outside the residual ledgers named in the certificate.*

Proof. Let $\mathcal{F}_\phi$ be a spectral object in the certified source scope. The functor row sends it to

$$\mathcal{A}_\phi := \mathsf{GL}_G(\mathcal{F}_\phi)$$

in the automorphic category. Hecke compatibility supplies the stated eigensheaf readout. If a mismatch appears, it is visible under a Hecke modification detector or belongs to nilpotent singular support, stack boundary, Eisenstein, constant-term, or endoscopic ledger data. The certificate records those ledgers, and closedness excludes unresolved hidden residuals. □

Theorem 9413.888 (Langlands existence and closure principle)**.** *On any Langlands scope carrying object-triple, finite-detector, Tannakian separation, Chebotarev, ledger, closedness, and either trace formula existence or geometric functor certificates, reciprocity holds on that scope. If, in addition, transfer trace certificates hold for the declared L-group homomorphisms, functoriality holds on the corresponding target scopes.*

Proof. Existence is supplied either by Definition 9413.882 and Definition 9413.883, or by Definition 9413.887 on the geometric scope. Detection and uniqueness are then supplied by the finite-detector, Tannakian, and Chebotarev rows of section 9413.70. Every residual not visible to a finite detector is forced into a ledger, and closedness rejects unresolved hidden ledger data. Transfer trace certificates apply the same reasoning to functorial parameters $\eta \circ \phi_H$. □

Corollary 9413.889 (Langlands as generated two-projection closure)**.** *When the existence bridge is certified, the Langlands correspondence on the declared scope is the generated two-projection closure of a single BEDC packet: the Frobenius projection and the Hecke projection agree on all finite detectors, and every remaining obstruction is ledgered.*

Proof. The existence bridge generates the missing side of the packet. The closure packet excludes finite mismatch and hidden ledger residuals. Thus the two projections are not merely compared after the fact; they are the two visible faces of one closed generated packet. □

Theorem 9413.890 (Existence bridge boundary)**.** *The BEDC–Langlands existence bridge reduces existence to certified trace-formula compactness or certified geometric functoriality. It does not itself prove the stable trace formula, the fundamental lemma, endoscopic classification, local Langlands packets, multiplicity formulas, continuous-spectrum decomposition, Shimura or cohomological realization, or the geometric Langlands categorical equivalence.*

Proof. Each positive theorem in this section names one of those constructions as an input certificate. The arguments show that once such a bridge is present, any absence or mismatch of the automorphic realization becomes a finite trace-window residual or a ledgered tail residual. They do not construct the bridge. The listed mathematical constructions therefore remain independent certificate obligations. □

9413.72 BEDC–Langlands Unramified Local Layer

The finite detector (v, r) used throughout the global BEDC–Langlands packet is a local object. It is licensed by the unramified Satake correspondence: spherical Hecke eigenvalues at an unramified nonarchimedean place are the same data as a semisimple conjugacy class on the dual side. This section isolates that local certificate before the global trace-window projector is used.

Definition 9413.891 (Unramified local spherical datum)**.** *Let F_v be a nonarchimedean local field with residue cardinality q_v. Let G/F_v be an unramified connected reductive group, and let*

$$K_v \subset G(F_v)$$

be a hyperspecial compact subgroup. The local spherical Hecke algebra is

$$\mathcal{H}_v(G, K_v) := C_c(K_v \backslash G(F_v)/K_v),$$

with convolution product. An unramified local automorphic row is an irreducible smooth representation π_v with

$$\pi_v^{K_v} \neq 0.$$

Definition 9413.892 (Local BEDC rows)**.** *The local time row is generated by the Cartan decomposition*

$$G(F_v) = \bigsqcup_{\lambda \in X_*(T)^+} K_v \lambda(\varpi_v) K_v.$$

Each dominant coweight λ determines a Hecke time step $T_{v,\lambda}$. The local unity row is the spherical Hecke character

$$\chi_{\pi_v} : \mathcal{H}_v(G, K_v) \to \mathbb{C}$$

or, equivalently, its Satake parameter readout. The local ledger row records every departure from the unramified spherical setting: ramification, failure of K_v-fixed vectors, non-hyperspecial level, normalization choices including $q_v^{1/2}$, central character rows, and bad reduction.

Definition 9413.893 (Unramified local parameter)**.** *Let $\widehat{G}$ be the dual group. In the unramified local setting, the local L-group has the form*

$${}^L G = \widehat{G} \rtimes \langle \mathrm{Frob}_v \rangle.$$

An unramified local parameter is represented by a semisimple conjugacy class

$$s_{\phi,v} \in \widehat{G}//\mathrm{conj},$$

equivalently by

$$\phi_v(\mathrm{Frob}_v) = s_{\phi,v} \rtimes \mathrm{Frob}_v$$

under the declared normalization.

Definition 9413.894 (Local Hecke–Frobenius residual)**.** *Let π_v be spherical and let ϕ_v be unramified. For every finite-dimensional representation $r : {}^L G \to GL(V_r)$, define*

$$D_{v,r}(\pi_v, \phi_v) := \operatorname{tr} r(s(\pi_v)) - \operatorname{tr} r(\phi_v(\mathrm{Frob}_v)).$$

The local detector r matches exactly when

$$D_{v,r}(\pi_v, \phi_v) = 0.$$

Theorem 9413.895 (Unramified Satake correspondence)**.** *For the unramified local spherical datum, algebra homomorphisms*

$$\mathcal{H}_v(G, K_v) \to \mathbb{C}$$

are equivalent to semisimple conjugacy classes in the dual quotient:

$$\mathrm{Hom}_{\mathrm{alg}}(\mathcal{H}_v(G, K_v), \mathbb{C}) \cong \widehat{G}//\mathrm{conj}.$$

Thus a spherical irreducible representation π_v has a Satake parameter

$$s(\pi_v) \in \widehat{G}//\mathrm{conj}.$$

Proof. The Satake isomorphism gives the normalized identification

$$\mathcal{H}_v(G, K_v) \cong \mathbb{C}[X_*(T)]^W,$$

with the $q_v^{1/2}$ convention recorded in the local ledger. The ring $\mathbb{C}[X_*(T)]^W$ is the coordinate ring of the semisimple conjugacy quotient of $\widehat{G}$. Algebra characters of the Hecke algebra are therefore points of that affine quotient, which are exactly semisimple Satake parameters. □

Corollary 9413.896 (Local finite detectors separate Satake parameters)**.** *If two spherical local rows π_v and π'_v satisfy*

$$\operatorname{tr} r(s(\pi_v)) = \operatorname{tr} r(s(\pi'_v))$$

for every finite-dimensional representation r of ${}^L G$, then

$$s(\pi_v) = s(\pi'_v)$$

in $\widehat{G}//\mathrm{conj}$.

Proof. Finite-dimensional characters separate semisimple conjugacy classes of the complex reductive dual group. Hence equality of all trace readouts places the two Satake parameters at the same point of the semisimple quotient. □

Theorem 9413.897 (Unramified local no-hidden residual)**.** *Let (π_v, ϕ_v) be an unramified local BEDC–Langlands row with the local three-properties discipline. If the row is closed against unresolved local ledger entries, then*

$$D_{v,r}(\pi_v, \phi_v) = 0$$

for every finite-dimensional representation r.

Proof. If $D_{v,r}(\pi_v, \phi_v) \neq 0$ for some r, then the finite local detector r has directly found a Hecke–Frobenius mismatch. Ledger discipline gives only two possibilities: the mismatch is recorded as a local residual row, or it is omitted. The first alternative leaves the object not closed until the row is resolved; the second violates the ledger property. Both contradict closedness with the three-properties discipline. Thus every local finite detector matches. □

Corollary 9413.898 (Unramified local Langlands closure)**.** *In the unramified spherical setting, a closed local BEDC–Langlands row is equivalent to equality of Satake and Frobenius parameters:*

$$s(\pi_v) = s_{\phi,v}$$

in $\widehat{G}//\mathrm{conj}$*. Equivalently,*

$$\operatorname{tr} r(s(\pi_v)) = \operatorname{tr} r(\phi_v(\mathrm{Frob}_v))$$

for every finite-dimensional r*.*

Proof. Equality in the quotient gives equality of all finite-dimensional trace readouts. Conversely, equality of all readouts separates the semisimple quotient by Definition 9413.896. The no-hidden residual theorem excludes any closed local row with a remaining detected mismatch. □

Definition 9413.899 (Local finite packet)**.** *Fix a finite displayed set of spherical local rows*

$$\mathcal{P}_v = \{\pi_{v,1}, \dots, \pi_{v,M}\}.$$

Write

$$s_i := s(\pi_{v,i}), \qquad s_\phi := s_{\phi,v}.$$

This is a fixed local packet list; no growing-window assertion is made inside the local projector construction.

Theorem 9413.900 (Local Hecke projector certificate)**.** *For a finite local packet* $\mathcal{P}_v$ *and target unramified parameter* s_ϕ*, there is*

$$f_{\phi,v} \in \mathcal{H}_v(G, K_v)$$

such that

$$s_i \neq s_\phi \quad \Longrightarrow \quad \operatorname{tr} \pi_{v,i}(f_{\phi,v}) = 0,$$

and

$$s_i = s_\phi \quad \Longrightarrow \quad \operatorname{tr} \pi_{v,i}(f_{\phi,v}) \neq 0$$

outside explicitly declared local cancellation rows.

Proof. By the Satake correspondence,

$$\mathcal{H}_v(G, K_v) \cong \mathbb{C}[\widehat{G}//\mathrm{conj}].$$

The finite set of points $\{s_1, \dots, s_M, s_\phi\}$ lies in an affine quotient. For each nonmatching $s_i \neq s_\phi$, the coordinate ring separates the two closed points, so there is a function F_i with

$$F_i(s_\phi) = 1, \qquad F_i(s_i) = 0.$$

The product

$$F_{\phi,v} := \prod_{s_i \neq s_\phi} F_i$$

is 1 at s_ϕ and 0 at every displayed nonmatching local parameter. Reading $F_{\phi,v}$ through Satake gives the desired Hecke function $f_{\phi,v}$. On a spherical eigenrow, the trace is evaluation at the Satake parameter. □

Theorem 9413.901 (Global detector as a product of local detectors)**.** *Let*

$$S = \{v_1, \dots, v_m\}$$

be a finite unramified window. For each $v \in S$, choose the local projector $f_{\phi,v}$ of Definition 9413.900. Then the finite tensor

$$f_{\phi,S} := \bigotimes_{v \in S} f_{\phi,v}$$

belongs to

$$\bigotimes_{v \in S} \mathcal{H}_v(G, K_v)$$

and kills every displayed global row whose Satake fingerprint fails to match ϕ at some place of S.

Proof. The restricted tensor product of the local spherical Hecke algebras is the finite-window Hecke algebra on S. If a displayed global packet fails to match the target at some $v \in S$, then the corresponding local factor has zero trace against $f_{\phi,v}$. The tensor trace is therefore zero. If all local Satake fingerprints match, each local factor is nonzero outside declared cancellation rows, so the finite tensor is the global fixed-window projector used by Definition 9413.910. □

Theorem 9413.902 (Fixed-window trace realization through local Satake)**.** *Assume every $v \in S$ is unramified for ϕ and for the displayed candidate packets. Suppose the local projectors $f_{\phi,v}$ are constructed as above, the stable trace-window certificate holds for*

$$f_{\phi,S} = \bigotimes_{v \in S} f_{\phi,v},$$

the parameter stable distribution is nonzero on $f_{\phi,S}$, and all non-target continuous, endoscopic, boundary, ramified, and normalization rows are separated or ledgered. Then some automorphic packet Π matches ϕ on every place of S:

$$s_v(\Pi) = \phi(\mathrm{Frob}_v) \qquad (v \in S).$$

Proof. The stable trace-window certificate transports the nonzero parameter readout to a nonzero spectral value. After the non-target ledger rows are separated, some automorphic packet has nonzero trace against $f_{\phi,S}$. By Definition 9413.901, a packet with any local mismatch in S is killed by the tensor projector. Hence the packet with nonzero trace matches the target Satake parameter at every place of the finite window. □

Theorem 9413.903 (Local-to-global residual dichotomy)**.** *Let (Π, ϕ) be a global BEDC–Langlands row. Any residual mismatch has one of two forms: either there is an unramified place v and a finite-dimensional representation r such that*

$$D_{v,r}(\Pi, \phi) \neq 0,$$

or the residual belongs to a ramified, archimedean, endoscopic, continuous-spectrum, boundary, nilpotent-support, or normalization ledger row.

Proof. If the residual appears at an unramified spherical place, the local Satake layer applies. A mismatch of semisimple local parameters is detected by some finite-dimensional representation by Definition 9413.896. This is the first alternative. If no unramified spherical detector sees the residual, then it lies outside the local layer: ramification, archimedean data, endoscopy, continuous spectrum, boundary terms, nilpotent or singular support, or normalization. Those are exactly the global ledger rows. □

Theorem 9413.904 (Local-to-global BEDC Langlands closure)**.** *Assume the unramified local layer above, fixed-window trace realization on a cofinal window tower, cofinal automorphic tail closure, complete ledger discipline, and closedness against unresolved ledger rows. Then the resulting global packet satisfies*

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

for every certified unramified finite detector (v, r)*.*

Proof. The unramified local layer licenses each finite detector (v, r) and the finite-window projectors built from them. Fixed-window trace realization supplies a realizing packet on each window. The cofinal window-to-global bridge gives either a global packet or an automorphic tail ledger row; closedness excludes unresolved tail rows. If a residual remained after the global packet is obtained, the local-to-global residual dichotomy would make it either a finite detector mismatch or a ledger row. A finite detector mismatch must be recorded and hence is not closed until resolved; an unrecorded one violates ledger discipline. The same applies to the ledger alternative. Therefore no residual remains on the certified unramified scope. □

Corollary 9413.905 (Quadratic detector as the abelian local shadow)**.** *For* $G = GL_1$ *and* $K = \mathbb{Q}(i)$*, the detector*

$$p \longmapsto \chi_{-4}(\mathrm{Frob}_p)$$

is the abelian local form of the general detector

$$(v, r) \longmapsto \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Thus the predicate $p = a^2 + b^2$ *is a visible local Frobenius detector in the abelian base case.*

Proof. For GL_1, the dual group is $\mathbb{C}^\times$ and finite-dimensional representations are characters. The Satake parameter at an unramified prime is therefore a single scalar. In the quadratic extension $\mathbb{Q}(i)/\mathbb{Q}$, the scalar $\chi_{-4}(\mathrm{Frob}_p)$ equals 1 exactly at split primes, which are the odd primes represented by $a^2 + b^2$ by Definition 9413.1129. □

Theorem 9413.906 (Unramified local boundary)**.** *The unramified local layer proves the legality and separation of finite Hecke–Frobenius detectors at spherical places. It does not prove the stable trace formula, construct parameter stable distributions, settle ramified local Langlands, or remove endoscopic, continuous-spectrum, boundary, nilpotent-support, or archimedean ledger rows.*

Proof. Every result in this section consumes the unramified spherical datum and the Satake correspondence. The non-spherical, ramified, endoscopic, continuous, boundary, nilpotent-support, and archimedean terms are excluded from the local proof and explicitly placed in the ledger. The global existence step is supplied separately by the trace-window or geometric existence bridge. □

9413.73 BEDC–Langlands Finite Hecke Projector Certificate

The existence bridge of section 9413.71 requires a local isolation step before the trace formula is used. A parameter readout on a fixed finite Hecke window must be converted into a test function that kills all displayed nonmatching spectral rows. The construction is purely finite: the places, the representations, the complexity bound, and the candidate packet list are fixed before the interpolation is performed.

Definition 9413.907 (Satake coordinate window)**.** *Let $S = \{v_1, \dots, v_m\}$ be a finite set of certified unramified places for G. The spherical Hecke algebra at v is denoted*

$$\mathcal{H}_v(G, K_v),$$

and the finite window algebra is

$$\mathcal{H}_S := \bigotimes_{v \in S} \mathcal{H}_v(G, K_v).$$

The Satake coordinate certificate identifies this algebra with regular functions on the finite product of semisimple conjugacy quotients,

$$\mathcal{H}_S \simeq \mathbb{C}\big[({}^L G//\mathrm{conj})^S\big].$$

Definition 9413.908 (Finite Frobenius and Satake fingerprints)**.** *For an allowed parameter packet $\phi^\uparrow$, its Frobenius fingerprint on S is*

$$x_{\phi,S} := \big(\phi(\mathrm{Frob}_v)\big)_{v \in S} \in ({}^L G//\mathrm{conj})^S.$$

For an automorphic packet Π unramified at every place of S, its Satake fingerprint is

$$x_{\Pi,S} := \big(s_v(\Pi)\big)_{v \in S} \in ({}^L G//\mathrm{conj})^S.$$

The packet Π matches $\phi^\uparrow$ on the window S when

$$x_{\Pi,S} = x_{\phi,S}.$$

Definition 9413.909 (Bounded window candidate packet)**.** *For a finite window S and a finite complexity bound B, let*

$$\mathsf{Aut}_{S,B}(G)$$

be the displayed automorphic packet list whose level, conductor, archimedean type, ramified local type, central character, and endoscopic label are bounded by B. A bounded window finiteness certificate says that

$$\mathsf{Aut}_{S,B}(G) = \{\Pi_1, \dots, \Pi_M\}$$

is finite up to the closed automorphic equivalence relation and that the Satake fingerprints $x_{\Pi_i,S}$ are readable in the Satake coordinate window.

Theorem 9413.910 (Finite Hecke projector certificate)**.** *Assume the Satake coordinate certificate and bounded window finiteness for (S, B). For every allowed parameter packet $\phi^\uparrow$ there is a test function*

$$f_{\phi,S,B} \in \mathcal{H}_S$$

such that for every $\Pi_i \in \mathsf{Aut}_{S,B}(G)$,

$$x_{\Pi_i,S} \neq x_{\phi,S} \quad \Longrightarrow \quad \mathrm{tr}\,\Pi_i(f_{\phi,S,B}) = 0,$$

while every matching packet satisfies

$$x_{\Pi_i,S} = x_{\phi,S} \quad \Longrightarrow \quad \mathrm{tr}\,\Pi_i(f_{\phi,S,B}) \neq 0$$

unless its contribution is cancelled by an explicitly declared local ledger row.

Proof. Under the Satake coordinate certificate, constructing $f_{\phi,S,B}$ is equivalent to constructing a regular function on the affine variety

$$({}^L G//\mathrm{conj})^S.$$

Write

$$X_i := x_{\Pi_i,S}.$$

For each nonmatching point $X_i \neq x_{\phi,S}$, the affine coordinate ring separates the two closed points. Hence there is

$$F_i \in \mathbb{C}\big[({}^L G//\mathrm{conj})^S\big]$$

with

$$F_i(x_{\phi,S}) = 1, \qquad F_i(X_i) = 0.$$

Define

$$F_{\phi,S,B} := \prod_{X_i \neq x_{\phi,S}} F_i.$$

Then

$$F_{\phi,S,B}(x_{\phi,S}) = 1$$

and $F_{\phi,S,B}(X_i) = 0$ for every displayed nonmatching packet. The Satake coordinate certificate reads $F_{\phi,S,B}$ as a Hecke test function $f_{\phi,S,B} \in \mathcal{H}_S$. On an unramified joint eigenpacket, the Hecke algebra acts through evaluation at its Satake fingerprint, so all nonmatching rows are killed and all displayed matching rows receive the nonzero scalar 1, except for cancellations that the local ledger names separately. □

Definition 9413.911 (Parameter evaluation distribution)**.** *On an unramified finite Satake window, the parameter evaluation distribution attached to $\phi^\uparrow$ is the functional*

$$\Theta^{\mathrm{st}}_{\phi,S} : \mathcal{H}_S \to \mathbb{C}$$

defined on Satake functions by

$$\Theta^{\mathrm{st}}_{\phi,S}(f) = f(x_{\phi,S}).$$

In ramified, archimedean, endoscopic, or transfer-factor settings this functional is accompanied by the corresponding local ledger rows.

Theorem 9413.912 (Parameter nonzero readout)**.** *For the finite Hecke projector of Definition 9413.910,*

$$\Theta^{\mathrm{st}}_{\phi,S}(f_{\phi,S,B}) = 1.$$

In particular,

$$\Theta^{\mathrm{st}}_{\phi,S}(f_{\phi,S,B}) \neq 0.$$

Proof. The projector is represented under Satake by the interpolation function $F_{\phi,S,B}$ constructed in the proof of Definition 9413.910. That function satisfies

$$F_{\phi,S,B}(x_{\phi,S}) = 1.$$

The parameter evaluation distribution is evaluation at $x_{\phi,S}$, so the displayed equality follows. □

Corollary 9413.913 (Spectral nonzero forces a matching packet)**.** *Let $f_{\phi,S,B}$ be the finite Hecke projector. Suppose the spectral side of the fixed window has the form*

$$I^{\mathrm{st}}_{\mathrm{spec}}(f_{\phi,S,B}) = \sum_{i=1}^{M} m(\Pi_i)\,\mathrm{tr}\,\Pi_i(f_{\phi,S,B}) + \mathrm{LedgerTerms},$$

and all continuous, endoscopic, boundary, ramified, normalization, and multiplicity terms outside the target window have been separated or cancelled by declared ledger rows. If

$$I^{\mathrm{st}}_{\mathrm{spec}}(f_{\phi,S,B}) \neq 0,$$

then some $\Pi_i \in \mathsf{Aut}_{S,B}(G)$ satisfies

$$x_{\Pi_i,S} = x_{\phi,S}.$$

Proof. If no displayed packet matched $x_{\phi,S}$, then Definition 9413.910 would give

$$\mathrm{tr}\,\Pi_i(f_{\phi,S,B}) = 0$$

for every i. After the non-target ledger terms are separated or cancelled, the spectral value would be zero, contradicting the assumed nonzero spectral value. □

Theorem 9413.914 (Fixed-window trace realization)**.** *Assume the finite Hecke projector certificate, parameter nonzero readout, stable trace-window equality, and separation of all non-target ledger terms for the fixed pair (S,B). Then the fixed window contains an automorphic packet Π such that*

$$x_{\Pi,S} = x_{\phi,S}.$$

Proof. By Definition 9413.912, the parameter side gives

$$\Theta^{\mathrm{st}}_{\phi,S}(f_{\phi,S,B}) \neq 0.$$

The stable trace-window equality transports this nonzero value to $I^{\mathrm{st}}_{\mathrm{spec}}(f_{\phi,S,B})$. The non-target ledger terms have been separated, so Definition 9413.913 supplies a displayed packet with the same finite Satake fingerprint as the parameter. □

Theorem 9413.915 (No missing fixed-window realization)**.** *Under the hypotheses of Definition 9413.914, it is impossible for $\phi^{\uparrow}$ to have no automorphic realization on the fixed window (S,B).*

Proof. Assume that no $\Pi_i \in \mathsf{Aut}_{S,B}(G)$ satisfies $x_{\Pi_i,S} = x_{\phi,S}$. The finite Hecke projector kills every displayed spectral row, and the non-target ledger rows have been separated or cancelled. Hence

$$I^{\mathrm{st}}_{\mathrm{spec}}(f_{\phi,S,B}) = 0.$$

On the parameter side, Definition 9413.912 gives

$$I^{\mathrm{st}}_{\mathrm{geom}}(f_{\phi,S,B}) = \Theta^{\mathrm{st}}_{\phi,S}(f_{\phi,S,B}) = 1.$$

The stable trace-window equality would force these two values to be equal, a contradiction. □

Corollary 9413.916 (Projector input for the cofinal bridge)**.** *If the hypotheses of Definition 9413.915 hold for every window in a cofinal Hecke window tower, then every tower window has an automorphic realization of $\phi^{\uparrow}$. Consequently the window-to-global alternative of Definition 9413.1074 applies once bounded automorphic finiteness and the automorphic tail ledger are available.*

Proof. Apply the fixed-window theorem to each displayed window. The resulting sequence of finite-window packets is precisely the input required by Definition 9413.1074. □

Theorem 9413.917 (Finite projector boundary)**.** *The finite Hecke projector certificate is a fixed-window interpolation statement. It does not assert growing-window compactness, the stable trace formula, bounded automorphic finiteness outside the displayed window, endoscopic transfer, or global automorphic realization.*

Proof. The construction uses only a finite candidate packet list, Satake coordinates on a finite set of unramified places, and interpolation in an affine coordinate ring. The passage from fixed windows to a global packet is supplied separately by section 9413.84; the analytic equality that makes the spectral side nonzero is supplied separately by the stable trace-window certificate. □

9413.74 BEDC–Langlands Parameter State Certificate

The trace-window argument needs a nonzero parameter-side readout before the stable trace formula can transport that readout to the spectral side. On an unramified fixed Hecke window this readout is simply evaluation at the Frobenius fingerprint of the parameter. Ramified, archimedean, endoscopic, and Arthur data are not inserted into the unramified Satake window; they are local parameter-state rows or ParameterLedger obligations.

Definition 9413.918 (Unramified parameter state)**.** *Let*

$$\mathcal{W} = (S, R, B)$$

be a fixed finite unramified Hecke window. The Satake coordinate window of Definition 9413.907 identifies

$$\mathcal{H}_{\mathcal{W}} \cong \mathbb{C}[({}^L G//\mathrm{conj})^S].$$

For an allowed parameter packet $\phi^{\uparrow}$, write

$$x_{\phi,\mathcal{W}} := \big(\phi(\mathrm{Frob}_v)\big)_{v\in S}.$$

The unramified parameter state is the evaluation functional

$$\Theta^{\mathrm{unr}}_{\phi,\mathcal{W}} : \mathcal{H}_{\mathcal{W}} \to \mathbb{C}$$

defined by

$$\Theta^{\mathrm{unr}}_{\phi,\mathcal{W}}(f) := f(x_{\phi,\mathcal{W}}).$$

Theorem 9413.919 (Unramified parameter state nonzero)**.** *Let $f_{\phi,\mathcal{W},B}$ be the finite Hecke projector attached to $\phi^{\uparrow}$ and $\mathcal{W}$. Then*

$$\Theta^{\mathrm{unr}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W},B}) = 1.$$

In particular,

$$\Theta^{\mathrm{unr}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W},B}) \neq 0.$$

Proof. The finite Hecke projector is represented by the interpolation function constructed in Definition 9413.910. It satisfies

$$f_{\phi,\mathcal{W},B}(x_{\phi,\mathcal{W}}) = 1.$$

The unramified parameter state is evaluation at $x_{\phi,\mathcal{W}}$, so the displayed equality follows. □

Definition 9413.920 (Parameter readout completeness)**.** *The finite-dimensional representation category* $\mathrm{Rep}({}^L G)$ *is the readout system for semisimple Langlands parameters: if semisimple elements* g *and* h *have*

$$\operatorname{tr} r(g) = \operatorname{tr} r(h)$$

for every finite-dimensional representation r*, then* g *and* h *define the same semisimple conjugacy class. Thus the evaluation functional on a finite Satake window is a complete finite readout of the displayed Frobenius fingerprint.*

Definition 9413.921 (Local parameter state ledger)**.** *At a ramified or archimedean place* v*, a local parameter state certificate is a local stable functional*

$$\Theta^{\mathrm{loc}}_{\phi_v} : \mathcal{H}^{\mathrm{loc}}_v \to \mathbb{C}$$

whose definition depends only on the local parameter row ϕ_v *and whose ledger records conductor, monodromy, local* ϵ/γ *factors, infinitesimal character, transfer factors, normalization, endoscopy, and Arthur* SL_2 *data when present. If such a functional is not available on the declared scope, the failure is a local parameter ledger row.*

Definition 9413.922 (Global ParameterLedger)**.** *For a global parameter packet* $\phi^{\uparrow}$*, the ParameterLedger is the finite and categorical ledger*

$$\mathrm{ParameterLedger}(\phi) := (S_{\mathrm{ram}}, N_v, \epsilon_v, \gamma_v, \mathsf{Monodromy}, \mathsf{ArthurSL_2}, \mathsf{Endoscopy}, \mathsf{ArchimedeanType}, \mathsf{Normalization}).$$

Here N_v *denotes Weil–Deligne monodromy, and every row records the source, classifier, local functional, and normalization data needed for the parameter state.*

Definition 9413.923 (Global parameter state)**.** *For a finite window with unramified, ramified, and archimedean parts, write*

$$\mathcal{W} = (S_{\mathrm{unr}}, S_{\mathrm{ram}}, R, B).$$

A global parameter state is the tensor functional

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}} := \left(\bigotimes_{v \in S_{\mathrm{unr}}} \Theta^{\mathrm{unr}}_{\phi_v} \right) \otimes \left(\bigotimes_{v \in S_{\mathrm{ram}}} \Theta^{\mathrm{loc}}_{\phi_v} \right) \otimes \Theta_{\phi,\infty},$$

where $\Theta_{\phi,\infty}$ *records the archimedean infinitesimal character, tempered or unitary row, and cohomological weight when those rows are part of the declared scope.*

Theorem 9413.924 (Parameter state certificate)**.** *Assume:*

- *the unramified finite projector* $f^{\mathrm{unr}}_{\phi,\mathcal{W},B}$ *is constructed;*
- *each ramified and archimedean local parameter state used by the window is nonzero on its declared local test function, or its failure is recorded in* $\mathrm{ParameterLedger}(\phi)$*;*
- *the test function decomposes as*

$$f_{\phi,\mathcal{W}} = f^{\mathrm{unr}}_{\phi,\mathcal{W},B} \otimes f^{\mathrm{ram}}_{\phi,S_{\mathrm{ram}}} \otimes f_{\phi,\infty};$$

- *every present local factor is normalized by*

$$\Theta^{\mathrm{loc}}_{\phi_v}(f_{\phi_v}) = 1,$$

  and the archimedean factor is normalized in the same way.

Then

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) = 1.$$

In particular, the parameter-side readout on the fixed window is nonzero.

Proof. By the tensor definition,

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) = \prod_{v\in S_{\mathrm{unr}}} \Theta^{\mathrm{unr}}_{\phi_v}(f_{\phi_v}) \cdot \prod_{v\in S_{\mathrm{ram}}} \Theta^{\mathrm{loc}}_{\phi_v}(f_{\phi_v}) \cdot \Theta_{\phi,\infty}(f_{\phi,\infty}).$$

The unramified factors are 1 by Definition 9413.919. The ramified and archimedean factors are 1 by the normalization hypotheses. Hence the product is 1. □

Theorem 9413.925 (Parameter state defect localization)**.** *Let $\phi^{\uparrow}$ be a declared Langlands parameter packet. If no fixed-window test function has nonzero parameter state readout, then one of the following is present:*

$$\mathsf{UnramifiedFrobeniusFingerprintDefect},\quad \mathsf{RamifiedLocalStateDefect},\quad \mathsf{ArchimedeanStateDefect},$$

$$\mathsf{EndoscopicArthurLedgerDefect},\quad \mathsf{UnlicensedParameterObject}.$$

Each listed defect is a ParameterLedger row.

Proof. If the unramified Frobenius fingerprint is defined, the unramified evaluation functional gives a nonzero readout on the finite Hecke projector by Definition 9413.919. Thus failure of all fixed-window parameter readouts cannot come from the unramified Satake part unless the Frobenius fingerprint itself is not licensed. If the unramified row is licensed, the possible failures are ramified local state, archimedean state, endoscopic or Arthur normalization, or the parameter object's own source and classifier data. By the definition of Definition 9413.922, these are exactly ParameterLedger rows. □

Theorem 9413.926 (Parameter state closure)**.** *For a legal BEDC Langlands parameter packet $\phi^{\uparrow}$ and any fixed Hecke window $\mathcal{W}$, either*

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0$$

for the declared parameter projector, or there is a recorded ParameterLedger defect.

Proof. If the unramified, ramified, and archimedean local parameter states used by the window are present and normalized, then Definition 9413.924 gives nonzero readout. If no such nonzero readout is available, then Definition 9413.925 places the failure in the ParameterLedger. A legal BEDC parameter packet cannot have a silent missing state functional outside that ledger. □

Theorem 9413.927 (Parameter state and stable trace give window realization)**.** *Assume parameter state closure on a fixed Hecke window, no unresolved ParameterLedger defect on that window, the stable trace formula certificate, and separation of all non-target trace ledger rows. Then there is an automorphic packet Π with*

$$x_{\Pi,\mathcal{W}} = x_{\phi,\mathcal{W}}.$$

Proof. With no unresolved ParameterLedger defect, parameter state closure gives

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0.$$

After trace ledger rows are separated, this is a nonzero geometric-side readout. The stable trace formula certificate transports it to a nonzero spectral-side readout. The finite Hecke projector then forces a displayed automorphic packet to match the parameter fingerprint on the fixed window. □

Theorem 9413.928 (Parameter state trace-tail closure)**.** *Assume ObjectTriple discipline, parameter state closure, stable trace formula certificate, finite Hecke projector certificates, cofinal automorphic tail closure, no-hidden Langlands residual, and closedness against unresolved ledger rows. Then every legal closed parameter packet $\phi^{\uparrow}$ with no unresolved ParameterLedger defect has a global automorphic realization Π_ϕ satisfying*

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

on the certified unramified finite detector scope.

Proof. Parameter state closure gives nonzero parameter readout on each fixed window. The stable trace formula certificate gives nonzero spectral readout, and the finite Hecke projector gives a fixed-window automorphic realization. Applying this along a cofinal window tower produces a realization sequence. Cofinal automorphic tail closure says that the sequence either globalizes or enters the automorphic tail ledger. Closedness excludes unresolved tail rows, so a global packet is obtained. No-hidden Langlands residual then excludes any remaining finite mismatch or ledger residual. □

Corollary 9413.929 (Abelian parameter state)**.** *For $G = GL_1$, the unramified parameter state is the scalar evaluation*

$$\Theta^{\mathrm{unr}}_{\phi,v}(f) = f(\phi(\mathrm{Frob}_v)).$$

For the quadratic parameter χ_{-4} attached to $\mathbb{Q}(i)/\mathbb{Q}$, this readout detects

$$\chi_{-4}(\mathrm{Frob}_p) = 1,$$

equivalently the representation of odd split primes by $a^2 + b^2$.

Proof. For GL_1, the dual group is $\mathbb{C}^\times$, so the Satake quotient is a one-dimensional character readout. Evaluation at $\phi(\mathrm{Frob}_v)$ is therefore the entire unramified parameter state. The quadratic splitting equivalence is Definition 9413.1129. □

Theorem 9413.930 (Parameter state boundary)**.** *The parameter state certificate closes the nonzero parameter-side readout on fixed Hecke windows. It does not construct every ramified local stable character, archimedean packet, Arthur packet, transfer-factor normalization, or endoscopic local state. When those rows are absent, they remain ParameterLedger obligations.*

Proof. The unramified part is evaluation at a Frobenius fingerprint and is closed by the finite projector construction. Ramified, archimedean, Arthur, and endoscopic states require their own local constructions. The certificate identifies their absence as ledger data rather than allowing a silent zero parameter-side readout. □

9413.75 BEDC–Langlands Ramified Local State Closure

The unramified layer is closed by Satake coordinates. At a ramified place there is no spherical Satake quotient, so the finite detector is not merely (v, r). It is a local testing row

$$(v, \mathfrak{b}, f_v),$$

where $\mathfrak{b}$ is a Bernstein or packet window and f_v is a compactly supported local Hecke test function. The role of a Satake point is played by a stable local character distribution attached to the local parameter packet.

Definition 9413.931 (Ramified local parameter ledger)**.** *Let F_v be a local field and let $G_v = G(F_v)$. A ramified local parameter is a Weil–Deligne row*

$$\phi_v : W'_{F_v} \to {}^L G.$$

Its local ledger records

$$\Lambda_{\phi_v} := (I_v\text{-type}, N_v, \epsilon_v, \gamma_v, \mathsf{Conductor}, \mathsf{Depth}, \mathsf{ArthurData}, \mathsf{Endoscopy}, \mathsf{Normalization}).$$

Here N_v is monodromy, and the normalization row includes transfer factors and local measure conventions.

Definition 9413.932 (Ramified local packet certificate)**.** *A ramified local packet certificate for ϕ_v consists of a finite or controlled local packet*

$$\Pi_{\phi_v} = \{\pi_{v,1}, \dots, \pi_{v,m}\}$$

and a stable local distribution

$$S\Theta_{\phi_v} := \sum_{i=1}^{m} a_i \Theta_{\pi_{v,i}},$$

where $\Theta_{\pi_{v,i}}$ is the Harish–Chandra character distribution and at least one coefficient a_i is nonzero. The certificate carries a local packet ledger for normalization, transfer factors, endoscopic labels, component-group characters, and local L, ϵ, γ factors. If the packet or stable distribution is not available, the failure is a row in the local parameter ledger.

Definition 9413.933 (Ramified local three-properties rows)**.** *The ramified local time row is generated by parabolic induction, Bernstein block decomposition, and conductor or depth filtration. The unity row separates a tempered unitary part from real-exponent or modulus rows. The ledger row records ramification, monodromy, depth, endoscopy, transfer factors, normalization, and wild ramification tails without allowing any of them to masquerade as an unramified Satake readout.*

Definition 9413.934 (Ramified local finite detector)**.** *A ramified local finite detector is a tuple*

$$(v, \mathfrak{b}, f_v)$$

where $\mathfrak{b}$ is a fixed Bernstein block or finite local packet window, and

$$f_v \in C_c^\infty(G_v)$$

is a local Hecke test function. For an admissible representation π_v, the detector reads

$$\Theta_{\pi_v}(f_v) = \operatorname{tr} \pi_v(f_v).$$

For a parameter packet, it reads

$$S\Theta_{\phi_v}(f_v).$$

The local residual is

$$D_{v,f_v}(\pi_v, \phi_v) := \Theta_{\pi_v}(f_v) - S\Theta_{\phi_v}(f_v).$$

Theorem 9413.935 (Local character separation)**.** *Fix a Bernstein or packet window containing finitely many irreducible admissible representations*

$$\pi_1, \ldots, \pi_M.$$

If their character distributions are pairwise distinct on the window, then finite linear combinations of local test functions separate the corresponding character rows. In particular, for any displayed target row one can construct a local test function that kills any prescribed finite set of nonmatching candidates and is nonzero on the target row, outside declared local cancellation rows.

Proof. The character distributions Θ_{π_i} are linear functionals on the local test-function space. Pairwise distinctness means that for each pair $i \neq j$ the difference

$$\Theta_{\pi_i} - \Theta_{\pi_j}$$

is a nonzero linear functional, hence is nonzero on some local test function. Since the candidate set is finite, finitely many such separating functions suffice. Taking finite linear combinations gives a projector on the finite list, with any local cancellation explicitly recorded in the ledger. □

Theorem 9413.936 (Ramified local state nonzero)**.** *If ϕ_v has a ramified local packet certificate and $S\Theta_{\phi_v}$ is not the zero distribution, then there exists*

$$f_v \in C_c^\infty(G_v)$$

such that

$$S\Theta_{\phi_v}(f_v) \neq 0.$$

Proof. The stable character $S\Theta_{\phi_v}$ is a linear functional on the local test-function space. A nonzero linear functional is nonzero on at least one test function. □

Corollary 9413.937 (Zero local state is a ledger row)**.** *If*

$$S\Theta_{\phi_v}(f_v) = 0$$

for every local test function f_v, then $S\Theta_{\phi_v}$ is the zero distribution. A declared legal ramified parameter with only the zero stable distribution has a local parameter ledger defect.

Proof. A linear functional that vanishes on every vector of its domain is the zero functional. If the parameter is nevertheless declared as a local state, the failure of nonzero state data must be recorded in Λ_{ϕ_v}; otherwise the local object has no state readout and violates the three-properties ledger row. □

Theorem 9413.938 (Ramified local no-hidden residual)**.** *Let (π_v, ϕ_v) be a ramified local BEDC–Langlands row with a ramified local packet certificate, local character separation, complete local ledger discipline, and closedness against unresolved local ledger rows. Then no ramified local mismatch remains hidden: any mismatch is detected by some finite local detector $(v, \mathfrak{b}, f_v)$ or is a local ledger row, and neither unresolved alternative is compatible with closedness.*

Proof. If the mismatch is visible in a fixed Bernstein or packet window, then the difference

$$\Theta_{\pi_v} - S\Theta_{\phi_v}$$

is a nonzero linear functional on the local test space. Hence there is some f_v with

$$D_{v,f_v}(\pi_v, \phi_v) \neq 0.$$

This is a finite local detector. Ledger discipline requires the detected residual to be recorded; carrying it unresolved violates closedness, while omitting it violates the ledger row.

If no fixed local window detects the mismatch, then it is not a character-distribution mismatch on the displayed packet window. It lies in normalization, transfer factors, endoscopy, monodromy, depth, component-group labels, or wild ramification tails. These are precisely the local ledger rows. Again, unresolved ledger data are incompatible with closedness. □

Definition 9413.939 (Ramified global parameter state)**.** *For a global parameter $\phi = (\phi_v)_v$ and a finite window*

$$\mathcal{W} = (S_{\mathrm{unr}}, S_{\mathrm{ram}}, B),$$

the global parameter state with ramification is

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}} := \left(\bigotimes_{v \in S_{\mathrm{unr}}} \Theta^{\mathrm{unr}}_{\phi_v} \right) \otimes \left(\bigotimes_{v \in S_{\mathrm{ram}}} S\Theta_{\phi_v} \right) \otimes \Theta_{\phi,\infty}.$$

Theorem 9413.940 (Global parameter state with ramification)**.** *Assume every unramified local state is the Satake evaluation state, every ramified local packet state present in the window is nonzero, the archimedean state is nonzero or its failure is ledgered, and the test function factors as*

$$f_{\phi,\mathcal{W}} = \bigotimes_{v \in S_{\mathrm{unr}}} f^{\mathrm{unr}}_{\phi_v} \otimes \bigotimes_{v \in S_{\mathrm{ram}}} f^{\mathrm{ram}}_{\phi_v} \otimes f_{\phi,\infty},$$

with each local factor normalized to value 1 *under its corresponding state. Then*

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) = 1.$$

Proof. By the tensor product definition,

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) = \prod_{v \in S_{\mathrm{unr}}} \Theta^{\mathrm{unr}}_{\phi_v}(f^{\mathrm{unr}}_{\phi_v}) \cdot \prod_{v \in S_{\mathrm{ram}}} S\Theta_{\phi_v}(f^{\mathrm{ram}}_{\phi_v}) \cdot \Theta_{\phi,\infty}(f_{\phi,\infty}).$$

Each factor is 1 by the hypotheses, so the product is 1. □

Theorem 9413.941 (Local-to-global parameter ledger dichotomy)**.** *If a global parameter state has no nonzero fixed-window test function, then at least one local parameter ledger row is present, or an archimedean, endoscopic, transfer-factor, or global normalization ledger row is present.*

Proof. If all unramified local states are present, their projectors give nonzero evaluation readouts. If every ramified packet state is nonzero, one can choose a local test function on which it is nonzero and rescale that local factor to value 1. If the archimedean state is also present and the global transfer and normalization rows are closed, the tensor test function of Definition 9413.940 has nonzero global state. Therefore total failure of all fixed-window readouts forces a missing or zero local state, or a global normalization, transfer, archimedean, or endoscopic defect. Those are exactly the ParameterLedger rows. □

Theorem 9413.942 (Ramified parameter state and trace realization)**.** *Assume the global parameter state with ramification is nonzero on the declared test function, the stable trace formula certificate holds, the finite Hecke and local projectors are constructed, and all non-target trace ledger rows are separated or discharged. Then the fixed window contains an automorphic packet matching the parameter on all displayed unramified and ramified local readouts.*

Proof. The nonzero global parameter state gives a nonzero stable geometric-side readout. The stable trace formula certificate transports it to a nonzero spectral-side readout. The unramified finite Hecke projector kills all displayed unramified nonmatching packets, and local character separation supplies the ramified packet-window projectors. Hence a nonzero spectral contribution must come from a packet matching the parameter on the whole fixed window. □

Theorem 9413.943 (Ramified BEDC–Langlands closure)**.** *Assume unramified Satake closure, ramified local packet certificates, local character separation, global parameter state nonzero, the stable trace formula certificate, finite-window projector certificates, cofinal automorphic tail closure, no-hidden Langlands residual, and closedness. Then every legal global parameter has a global automorphic realization Π_ϕ whose unramified Hecke–Frobenius readouts match ϕ and whose ramified local packet identities, local factors, and transfer data are ledgered and closed.*

Proof. The global parameter state and stable trace formula give a nonzero spectral readout in every fixed window. The unramified and ramified projector certificates force a matching packet in each fixed window. The cofinal automorphic tail closure globalizes the window realizations unless they enter a tail ledger; closedness excludes unresolved tail rows. Thus a global packet Π_ϕ is obtained.

Any remaining unramified mismatch is detected by a finite Hecke–Frobenius detector. Any remaining ramified mismatch is detected by Definition 9413.938 or lies in a local ledger row. No-hidden residual and closedness exclude both unresolved alternatives. □

Corollary 9413.944 (The abelian ramified check)**.** *For $G = GL_1$, a ramified local packet is a local character*

$$\chi_v : F_v^\times \to \mathbb{C}^\times$$

with conductor ledger. The local state

$$\Theta_{\chi_v}(f_v) = \int_{F_v^\times} f_v(x)\chi_v(x)\, d^\times x$$

is nonzero on a conductor-compatible compact coset test function after normalization. For the quadratic character of $\mathbb{Q}(i)/\mathbb{Q}$, the prime 2 is the ramified conductor row, while odd primes are the unramified detectors of Definition 9413.1129.

Proof. Choose a compact coset on which the local character is constant and let f_v be its normalized characteristic test function. The integral is then the chosen character value times the normalized volume, so it can be scaled to 1. For $\mathbb{Q}(i)/\mathbb{Q}$, the only ramified prime is 2, and the odd-prime splitting detector remains unramified. □

Theorem 9413.945 (Ramified local state boundary)**.** *Ramified local state closure consumes local packet certificates, character distribution separation, transfer-factor normalization, and local factor compatibility. It does not construct those traditional local Langlands and endoscopic inputs; when they are absent, the absence remains a local ParameterLedger obligation.*

Proof. The proofs above are linear-functional and ledger arguments once the local packet and stable character distribution are supplied. The existence of ramified packets, transfer identities, component-group labels, and character identities is separate local Langlands data. BEDC records that data as explicit certificates rather than hiding it inside the unramified Satake layer. □

9413.76 BEDC–Langlands Local Packet Certificate for GL_n

The ramified local state closure of section 9413.75 requires a local packet, a nonzero character distribution, and a ledger for conductor, monodromy, local factors, and normalization. For general reductive groups this package involves endoscopy and component-group bookkeeping. For GL_n over a nonarchimedean local field the packet certificate has a cleaner form: the local Langlands packet is a singleton, and all remaining ramified data are explicit ledger rows.

Definition 9413.946 (Local GL_n parameter row)**.** *Let F_v be a nonarchimedean local field. A local GL_n parameter row is an n-dimensional Frobenius-semisimple Weil–Deligne representation*

$$\phi_v : W'_{F_v} \to GL_n(\mathbb{C}),$$

equivalently a pair

$$(r_v, N_v)$$

with the usual monodromy compatibility. The parameter set is denoted

$$\Phi_n(F_v).$$

The automorphic local side is

$$\mathrm{Irr}(GL_n(F_v)),$$

the set of irreducible smooth admissible representations.

Definition 9413.947 (GL_n local Langlands certificate)**.** *A GL_n local Langlands certificate at v is a classical NameCert/PacketCert consisting of a bijection*

$$\mathrm{rec}_v : \mathrm{Irr}(GL_n(F_v)) \longrightarrow \Phi_n(F_v)$$

with the following ledgered compatibilities:

- *spherical representations have Satake parameter equal to the semisimple Frobenius class of $\mathrm{rec}_v(\pi_v)$;*
- *Rankin–Selberg local L, ϵ, and γ factors agree with the corresponding tensor-product factors on Weil–Deligne parameters;*
- *central characters satisfy*
$$\omega_{\pi_v} = \det(\mathrm{rec}_v(\pi_v)) \circ \mathrm{Art}_{F_v};$$
- *contragredients and character twists commute with the correspondence;*
- *conductor, depth, inertia type, monodromy N_v, additive character, Haar measure, and normalization conventions are recorded in the local ledger.*

The certificate is used here as a traditional mathematical input. The BEDC claim is the closure consequence obtained after this certificate is present.

Definition 9413.948 (GL_n local ledger)**.** *For a GL_n local parameter ϕ_v, the local ledger is*

$$\Lambda_{\phi_v}^{GL_n} := (I_v\text{-type}, N_v, \mathfrak{f}(\phi_v), \mathsf{Depth}, L_v, \epsilon_v, \gamma_v, \mathsf{CentralCharacter}, \mathsf{Normalization}).$$

It records every datum that is invisible to the unramified spherical Satake detector.

Theorem 9413.949 (GL_n packets are singleton packets)**.** *Under the GL_n local Langlands certificate, every parameter*

$$\phi_v \in \Phi_n(F_v)$$

has a unique corresponding irreducible admissible representation

$$\pi_{\phi_v} := \mathrm{rec}_v^{-1}(\phi_v).$$

Therefore its local packet is

$$\Pi_{\phi_v} = \{\pi_{\phi_v}\}.$$

Proof. The correspondence rec_v is a bijection. Thus the inverse image of ϕ_v exists and has exactly one element. The packet attached to ϕ_v is consequently the displayed singleton. □

Theorem 9413.950 (GL_n local three-properties certificate)**.** *The local row*

$$\mathcal{O}_v^{GL_n} := (\pi_{\phi_v}, \phi_v, \Lambda_{\phi_v}^{GL_n})$$

satisfies the BEDC time, unity, and ledger properties.

Proof. Time is supplied on the representation side by Bernstein decomposition, parabolic induction, depth filtration, and conductor filtration. On the parameter side it is supplied by Frobenius, inertia, and monodromy in

$$\phi_v = (r_v, N_v).$$

Unity is supplied by the tempered/unitary part of the local Langlands classification. For a general irreducible admissible representation, the Langlands classification expresses the row through tempered data and real exponent data; the tempered part is the unitary readout, while the real exponents are recorded as scale ledger rows.

The ledger property is exactly Definition 9413.948: conductor, depth, monodromy, local factors, central character, and normalization conventions are named rather than hidden in the unramified detector. □

Definition 9413.951 (GL_n stable local state)**.** *For a GL_n parameter ϕ_v, define its local stable state by*

$$S\Theta_{\phi_v} := \Theta_{\pi_{\phi_v}},$$

where $\Theta_{\pi_{\phi_v}}$ is the Harish–Chandra character distribution

$$\Theta_{\pi_{\phi_v}}(f_v) := \mathrm{tr}\, \pi_{\phi_v}(f_v), \qquad f_v \in C_c^\infty(GL_n(F_v)).$$

Because the packet is a singleton, no component-group sum is required.

Theorem 9413.952 (GL_n local state nonzero)**.** *For every GL_n local parameter ϕ_v, there exists*

$$f_v \in C_c^\infty(GL_n(F_v))$$

such that

$$S\Theta_{\phi_v}(f_v) \neq 0.$$

Proof. The representation π_{ϕ_v} is smooth and nonzero. Hence some nonzero vector is fixed by a compact open subgroup

$$K \subset GL_n(F_v).$$

Let

$$e_K = \frac{1_K}{\mathrm{vol}(K)}$$

be the normalized idempotent in the local Hecke algebra. The operator $\pi_{\phi_v}(e_K)$ is the projection onto the finite-dimensional space $\pi_{\phi_v}^K$. Since $\pi_{\phi_v}^K \neq 0$,

$$\Theta_{\pi_{\phi_v}}(e_K) = \mathrm{tr}\, \pi_{\phi_v}(e_K) = \dim \pi_{\phi_v}^K \neq 0.$$

By the definition of $S\Theta_{\phi_v}$, the same test function gives the required nonzero local state. □

Definition 9413.953 (GL_n local test residual)**.** *For a candidate representation π_v and a parameter ϕ_v, define the local test residual by*

$$D_{v,f_v}^{GL_n}(\pi_v, \phi_v) := \Theta_{\pi_v}(f_v) - \Theta_{\pi_{\phi_v}}(f_v).$$

The ramified finite detector is the row

$$(v, \mathfrak{b}, f_v),$$

where $\mathfrak{b}$ is a displayed Bernstein or packet window containing the candidates under comparison.

Theorem 9413.954 (GL_n character separation)**.** *Let*

$$\pi_1, \ldots, \pi_M$$

be pairwise non-isomorphic irreducible admissible representations of $GL_n(F_v)$ in a displayed finite local window. Then their Harish–Chandra character distributions are linearly independent as functionals on $C_c^\infty(GL_n(F_v))$. Consequently, for each target index i there is a test function

$$f_i^* \in C_c^\infty(GL_n(F_v))$$

such that

$$\Theta_{\pi_\ell}(f_i^*) = \delta_{\ell i}$$

for all displayed candidates π_ℓ.

Proof. Harish–Chandra character theory gives linear independence of the characters of pairwise non-isomorphic irreducible admissible representations. For the finite family

$$\Theta_{\pi_1}, \ldots, \Theta_{\pi_M},$$

linear independence in the dual space implies that there are test functions

$$f_1, \ldots, f_M$$

for which the matrix

$$A_{\ell j} := \Theta_{\pi_\ell}(f_j)$$

is invertible. Solving

$$\sum_j c_j A_{\ell j} = \delta_{\ell i}$$

and setting

$$f_i^* := \sum_j c_j f_j$$

gives the displayed isolating function. □

Theorem 9413.955 (GL_n ramified local no-hidden residual)**.** *Let (π_v, ϕ_v) be a local GL_n BEDC–Langlands row equipped with the local Langlands certificate and the three-properties ledger. If the row is closed against unresolved local residuals, then*

$$\pi_v \simeq \pi_{\phi_v}.$$

Equivalently,

$$D^{GL_n}_{v,f_v}(\pi_v, \phi_v) = 0$$

for every local test function f_v.

Proof. Assume $\pi_v \not\simeq \pi_{\phi_v}$. By Definition 9413.954, there is a local test function f_v with

$$\Theta_{\pi_v}(f_v) \neq \Theta_{\pi_{\phi_v}}(f_v).$$

Thus

$$D^{GL_n}_{v,f_v}(\pi_v, \phi_v) \neq 0.$$

This is a finite local detector. If the detected residual is entered in the ledger, the row is not closed until the residual is discharged. If it is not entered in the ledger, the ledger property fails. Both alternatives contradict closedness with the three-properties discipline. Hence $\pi_v \simeq \pi_{\phi_v}$, and all residuals vanish. □

Corollary 9413.956 (GL_n ramified local state closure)**.** *For $GL_n(F_v)$, every legal local parameter with a local Langlands certificate has a nonzero local state. Every ramified local mismatch is detected by a finite local test row $(v, \mathfrak{b}, f_v)$ or is an explicit local ledger row.*

Proof. Nonzero state follows from Definition 9413.952. Finite detection of representation mismatch follows from Definition 9413.955. The remaining ramified data are exactly the rows listed in Definition 9413.948. □

Definition 9413.957 (GL_n fixed-window parameter state)**.** *For a finite local-global window*

$$\mathcal{W} = (S_{\mathrm{unr}}, S_{\mathrm{ram}}, B),$$

the GL_n parameter state is

$$\Theta^{GL_n}_{\phi,\mathcal{W}} := \left(\bigotimes_{v \in S_{\mathrm{unr}}} \Theta^{\mathrm{unr}}_{\phi_v} \right) \otimes \left(\bigotimes_{v \in S_{\mathrm{ram}}} \Theta_{\pi_{\phi_v}} \right) \otimes \Theta_{\phi,\infty},$$

with archimedean factors present only on the declared scope.

Theorem 9413.958 (GL_n fixed-window parameter nonzero)**.** *Assume every archimedean factor in the declared window has a nonzero local state or a ledgered failure. Then every GL_n parameter with local Langlands certificates at the displayed nonarchimedean places has a fixed-window test function*

$$f_{\phi,\mathcal{W}}$$

such that

$$\Theta^{GL_n}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0.$$

Proof. At unramified places, use the Satake projector from Definition 9413.919 and normalize its value to 1. At ramified nonarchimedean places, use Definition 9413.952 and rescale the chosen local test function so that

$$\Theta_{\pi_{\phi_v}}(f_{\phi_v}) = 1.$$

Normalize every present archimedean factor in the same way. The tensor product of these local functions satisfies

$$\Theta^{GL_n}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) = \prod_v \Theta_{\phi_v}(f_{\phi_v}) = 1.$$

□

Theorem 9413.959 (GL_n local-global residual dichotomy)**.** *Let (Π, ϕ) be a global GL_n BEDC–Langlands row on a fixed window. Any residual is either detected by an unramified Hecke–Frobenius detector, detected by a ramified local test row $(v, \mathfrak{b}, f_v)$, or belongs to an archimedean, continuous-spectrum, level-escape, conductor, normalization, or automorphic-tail ledger row.*

Proof. An unramified mismatch is detected by the finite representation readout (v, r) of Definition 9413.897. A ramified nonarchimedean mismatch is detected by Definition 9413.955. If no local detector sees the residual, it is not a displayed local character or Satake mismatch. It must therefore lie in archimedean module data, continuous spectrum, Eisenstein contribution, level or conductor escape, normalization, or automorphic tail behavior. These are global ledger rows by the trace-window and tail-closure certificates. □

Theorem 9413.960 (GL_n BEDC–Langlands conditional closure)**.** *Assume the GL_n local Langlands certificates at all displayed nonarchimedean places, the fixed-window parameter state certificate, the trace or existence bridge certificate, cofinal automorphic tail closure, no-hidden Langlands residual, and closedness. Then every legal global GL_n parameter in the declared scope has a global automorphic realization whose unramified Hecke–Frobenius readouts match the parameter and whose ramified local data are matched by the local Langlands ledger.*

Proof. The local packet certificate gives singleton ramified packets and nonzero local states. Hence Definition 9413.958 supplies a nonzero fixed-window parameter readout. The trace or existence bridge transports this readout to a nonzero automorphic spectral contribution on the same fixed window. Cofinal automorphic tail closure promotes compatible fixed-window realizations to a global automorphic row unless a tail ledger row is present; closedness excludes an unresolved tail.

If a residual remains, then Definition 9413.959 places it either in a finite detector or in a ledger row. No-hidden residual and closedness exclude unresolved instances of both alternatives. □

Theorem 9413.961 (GL_n local certificate boundary)**.** *The GL_n local packet certificate closes the local ramified BEDC module: singleton packet, nonzero local state, finite character detection, and local ledger discipline. It does not by itself construct the global automorphic realization; that construction is supplied by the separate trace or existence bridge.*

Proof. All preceding local statements use only the local Langlands certificate, Harish–Chandra character separation, admissibility, and the local ledger. Global existence requires a mechanism that turns a nonzero parameter-side fixed-window state into a spectral automorphic row. That mechanism is outside the local certificate and is exactly the role of the trace or existence bridge. □

9413.77 BEDC–Langlands Global Converse Bridge for GL_n

The local packet certificate of section 9413.76 turns every certified local GL_n parameter into a singleton local packet. The global existence problem is not solved by merely tensoring these local packets. The restricted tensor product is a legitimate adelic candidate only after the global compatibility ledger is present, and it becomes automorphic only after an existence bridge is supplied. For GL_n, the natural bridge is the Rankin–Selberg converse theorem.

Definition 9413.962 (GL_n global parameter row)**.** *Let F be a global field. A GL_n global parameter row is a parameter*

$$\phi : W'_F \to GL_n(\mathbb{C})$$

with local components

$$\phi_v : W'_{F_v} \to GL_n(\mathbb{C}).$$

At each nonarchimedean place where the local GL_n Langlands certificate is in scope, write

$$\pi_{\phi_v} := \mathrm{rec}_v^{-1}(\phi_v)$$

for the corresponding singleton packet representative.

Definition 9413.963 (Global compatibility certificate)**.** *A global compatibility certificate for ϕ consists of the following ledgered data:*

- *ϕ_v is unramified for almost all places v;*
- *π_{ϕ_v} is spherical for almost all nonarchimedean places;*
- *the central character row is global and satisfies*

$$\omega_\phi = \det(\phi) \circ \mathrm{Art}_F;$$

- *all ramified places, conductor exponents, monodromy rows, archimedean types, local L-factors, ϵ-factors, γ-factors, and normalization choices enter the global parameter ledger*

$$\Lambda_\phi^{GL_n} := (S_{\mathrm{ram}}, N_v, \mathfrak{f}_v, \epsilon_v, \gamma_v, \mathsf{ArchType}, \mathsf{CentralCharacter}, \mathsf{Normalization}).$$

Failure of any row is a global compatibility defect rather than an automorphic realization.

Definition 9413.964 (Local tensor product candidate)**.** *Under the global compatibility certificate, the local GL_n packets define the restricted tensor product candidate*

$$\Pi_\phi^{\mathrm{loc}} := \bigotimes_v{}' \pi_{\phi_v}.$$

This is an irreducible admissible adelic representation candidate of $GL_n(\mathbb{A}_F)$ on the declared scope. It is not, by definition, an automorphic representation.

Theorem 9413.965 (Local tensor candidate readout)**.** *If ϕ has a global compatibility certificate, then Π_ϕ^{loc} is defined as a restricted tensor product. For almost all unramified places v,*

$$s_v(\Pi_\phi^{\mathrm{loc}}) = \phi(\mathrm{Frob}_v).$$

Proof. Almost all local rows are unramified by the global compatibility certificate, so the corresponding representations are spherical and carry normalized spherical vectors. The restricted tensor product is therefore defined. At an unramified place, the local GL_n Langlands certificate is compatible with the Satake parameter, giving the displayed equality. □

Definition 9413.966 (Rankin–Selberg twist detector)**.** *Let $1 \le m \le n-1$, and let τ be a cuspidal automorphic representation of $GL_m(\mathbb{A}_F)$ in the admitted detector class. The Rankin–Selberg twist detector attached to (m, τ) reads the Euler product*

$$L(s, \Pi_\phi^{\mathrm{loc}} \times \tau) := \prod_v L(s, \pi_{\phi_v} \times \tau_v).$$

When τ has a parameter row ϕ_τ, the parameter-side expected readout is

$$L(s, \phi \otimes \phi_\tau).$$

The analytic residual is the ledger row recording failure of any of the following:

$$L(s, \Pi_\phi^{\mathrm{loc}} \times \tau) = L(s, \phi \otimes \phi_\tau),$$

meromorphic continuation, the prescribed functional equation, boundedness in vertical strips, and local $L/\epsilon/\gamma$ factor compatibility.

Definition 9413.967 (Finite twist window)**.** *For a complexity bound B, the finite twist window is*

$$\mathcal{T}_B(n) := \{(m, \tau) : 1 \le m \le n-1,\ \kappa(\tau) \le B\},$$

where $\kappa(\tau)$ records conductor, level, ramified type, central character, archimedean parameter, and normalization complexity. A residual visible for some $(m, \tau) \in \mathcal{T}_B(n)$ is a finite Rankin residual. A residual invisible in every fixed $\mathcal{T}_B(n)$ but present along unbounded conductor, level, ramified type, or archimedean parameter is a twist tail residual, recorded in

$$\mathsf{TwistTailLedger} := (\mathsf{ConductorEscape}, \mathsf{ArchimedeanEscape}, \mathsf{RamificationEscape}, \mathsf{NonuniformBoundedness}, \mathsf{GammaFact}$$

Theorem 9413.968 (Rankin residual dichotomy)**.** *Let Π_ϕ^{loc} be a global GL_n local tensor product candidate whose local packet and compatibility ledgers are closed. If the converse-theorem analytic hypotheses fail, then the failure is either a finite Rankin residual in some fixed twist window or a twist tail residual.*

Proof. The analytic hypotheses are tested by Rankin–Selberg twists (m, τ) with $1 \le m \le n-1$. If a hypothesis fails for a twist of bounded detector complexity, it is visible in a finite window $\mathcal{T}_B(n)$ and is a finite Rankin residual. If every bounded window is clean but failure occurs only along twists with unbounded conductor, level, ramified type, archimedean parameter, or normalization complexity, then the failure is not a fixed-window datum. It is precisely a twist tail residual. □

Definition 9413.969 (GL_n converse theorem certificate)**.** *A GL_n converse theorem certificate is a traditional analytic bridge which applies to an irreducible admissible generic representation candidate Π of $GL_n(\mathbb{A}_F)$ with the declared central character and asserts: if for all $1 \le m \le n-1$ and all admitted cuspidal automorphic τ on $GL_m(\mathbb{A}_F)$ the Rankin–Selberg twists*

$$L(s, \Pi \times \tau)$$

have the required meromorphic continuation, functional equation, boundedness in vertical strips, and local factor compatibility, then

$$\Pi$$

is automorphic. Non-generic, residual-spectrum, and boundary cases are not hidden inside this certificate; they enter the residual-spectrum ledger.

Definition 9413.970 (No hidden Rankin residual)**.** *The no-hidden Rankin residual condition says that every failure of a Rankin–Selberg analytic hypothesis is either detected in a fixed twist window $\mathcal{T}_B(n)$ or is recorded in* TwistTailLedger*, and that closed accepted rows carry no unresolved entry in either ledger.*

Theorem 9413.971 (Converse bridge closure)**.** *Assume ϕ has a global compatibility certificate, the local GL_n packet certificates are present, Π_ϕ^{loc} lies in the generic scope of the GL_n converse theorem certificate, no finite Rankin residual is present, no twist tail residual remains unresolved, and the row is closed. Then*

$$\Pi_\phi^{\mathrm{loc}}$$

is automorphic.

Proof. If Π_ϕ^{loc} were not automorphic while the converse certificate applied, at least one converse-theorem hypothesis would fail. By Definition 9413.968, that failure is either a finite Rankin residual or a twist tail residual. A finite residual contradicts the hypothesis that no finite Rankin residual is present. A tail residual contradicts the hypothesis that no twist tail residual remains unresolved. Hence all converse-theorem analytic hypotheses are closed. The GL_n converse theorem certificate therefore gives automorphy of Π_ϕ^{loc}. □

Theorem 9413.972 (GL_n global reciprocity from converse bridge)**.** *Assume:*

- *ϕ has a global compatibility certificate;*
- *local GL_n Langlands packet certificates hold at the displayed places;*
- *Π_ϕ^{loc} is in the generic scope of the GL_n converse theorem certificate;*
- *all Rankin–Selberg analytic detector residuals are finite-clean or discharged in the twist tail ledger;*
- *the no-hidden Langlands residual condition and closedness hold.*

Then there is an automorphic representation

$$\Pi_\phi$$

of $GL_n(\mathbb{A}_F)$ such that, for almost all unramified places v,

$$s_v(\Pi_\phi) = \phi(\mathrm{Frob}_v).$$

Equivalently, for every finite-dimensional representation r of the dual group in the certified readout class,

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Proof. The local packet certificates and the global compatibility certificate construct the restricted tensor product

$$\Pi_\phi^{\mathrm{loc}}.$$

The Rankin–Selberg detector hypotheses and Definition 9413.971 make this candidate automorphic. Set

$$\Pi_\phi := \Pi_\phi^{\mathrm{loc}}.$$

The almost-everywhere Satake equality follows from Definition 9413.965. Applying finite-dimensional representation readouts gives the displayed Hecke–Frobenius equality. Any remaining mismatch is a finite Hecke–Frobenius detector, a ramified local ledger row, an archimedean row, a residual-spectrum row, or an automorphic tail row; no-hidden residual and closedness exclude unresolved instances. □

Theorem 9413.973 (GL_N functorial transfer through the converse bridge)**.** *Let*

$$\eta : {}^L H \to GL_N(\mathbb{C})$$

be an N-dimensional L-group representation and let

$$\phi_H : W_F' \to {}^L H$$

be an H-side parameter. Put

$$\phi_\eta := \eta \circ \phi_H.$$

If ϕ_η satisfies the hypotheses of Definition 9413.972 for GL_N, then there is an automorphic representation Π_η of $GL_N(\mathbb{A}_F)$ with

$$s_v(\Pi_\eta) = \eta(\phi_H(\mathrm{Frob}_v))$$

for almost all certified unramified places v.

Proof. The composed parameter ϕ_η is a GL_N parameter row. Applying Definition 9413.972 to ϕ_η gives an automorphic representation Π_η with

$$s_v(\Pi_\eta) = \phi_\eta(\mathrm{Frob}_v).$$

The definition of ϕ_η identifies the right side with $\eta(\phi_H(\mathrm{Frob}_v))$. □

Theorem 9413.974 (Nonautomorphy is a Rankin residual)**.** *Within the scope of the GL_n converse theorem certificate, a legal local tensor product candidate that is not automorphic carries a Rankin–Selberg analytic residual.*

Proof. The contrapositive form of the converse theorem certificate says that a candidate in its scope which is not automorphic must fail at least one of the hypotheses: meromorphic continuation, functional equation, vertical-strip boundedness, or local factor compatibility for some Rankin–Selberg twist. Each failure is exactly a Rankin analytic residual. □

Corollary 9413.975 (Closed GL_n candidates are automorphic)**.** *In the declared generic scope, a GL_n local tensor product candidate with global compatibility, local packet closure, no hidden Rankin residual, and closedness cannot be both legal and nonautomorphic.*

Proof. If it were nonautomorphic, then Definition 9413.974 would produce a Rankin analytic residual. By Definition 9413.968, that residual is finite-visible or a twist tail row. The first alternative contradicts finite-clean Rankin closure; the second contradicts closedness against unresolved twist tail rows. □

Theorem 9413.976 (GL_n converse bridge boundary)**.** *The GL_n global converse bridge closes the passage from local packet data to automorphic realization only under the stated analytic certificate: global compatibility, generic scope, Rankin–Selberg twist closure, tail-ledger discipline, and closedness. It does not replace the converse theorem, nor does it assert automorphy for parameters outside the certificate's scope.*

Proof. The construction up to Π_ϕ^{loc} is local and ledgered. The step from a legal adelic candidate to an automorphic representation is precisely the converse theorem certificate. Outside its hypotheses, the failure is not erased; it is recorded as global-compatibility, non-generic, residual-spectrum, analytic, or tail ledger data. □

9413.78 BEDC–Langlands Tannakian Functorial Closure

The GL_n converse bridge constructs automorphic realizations for linear representations of an L-parameter. For a general reductive group G, the finite readouts are the finite-dimensional representations of ${}^LG$. The question is not whether each composed parameter

$$r \circ \phi : W_F' \to GL(V_r)$$

has a $GL(V_r)$-realization, but whether the resulting family of GL-objects is tensor-compatible and therefore comes from one ${}^LG$-valued packet.

Definition 9413.977 (Tannakian detector)**.** *A Tannakian detector for G is a pair*

$$(v, r)$$

where v is an unramified place in the certified scope and

$$r : {}^LG \to GL(V_r)$$

is a finite-dimensional representation. For a parameter ϕ, the parameter-side readout is

$$a_{v,r}(\phi) := \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

For a G-side automorphic packet Π_G with Satake parameter $s_v(\Pi_G)$, the automorphic readout is

$$\lambda_{v,r}(\Pi_G) := \operatorname{tr} r(s_v(\Pi_G)).$$

The finite detector residual is

$$D_{v,r}(\Pi_G, \phi) := \lambda_{v,r}(\Pi_G) - a_{v,r}(\phi).$$

Definition 9413.978 (Representation compatibility rows)**.** *Let*

$$\{\Pi_r\}_{r \in \mathrm{Rep}({}^LG)}$$

be a family of $GL(V_r)$-side automorphic objects obtained from the composed parameters $r \circ \phi$. Its Tannakian compatibility rows are the finite relations generated by direct sum, tensor product, and dual:

$$\Pi_{r_1 \oplus r_2} \simeq \Pi_{r_1} \boxplus \Pi_{r_2},$$
$$L(s, \Pi_{r_1 \otimes r_2}) = L(s, \Pi_{r_1} \times \Pi_{r_2}),$$

and

$$\Pi_{r^\vee} \simeq \Pi_r^\vee.$$

Failures of these rows are denoted

$$R_\oplus(r_1, r_2), \qquad R_\otimes(r_1, r_2), \qquad R_\vee(r),$$

and form the Tannakian compatibility ledger.

Theorem 9413.979 (Tannakian separation)**.** *Let $g, h \in {}^LG$ be semisimple. If*

$$\operatorname{tr} r(g) = \operatorname{tr} r(h)$$

for every finite-dimensional representation r of ${}^LG$, then g and h have the same semisimple conjugacy class.

Proof. This is the detector separation theorem of Definition 9413.860. Characters of finite-dimensional representations generate the coordinate algebra of the semisimple conjugacy quotient

$${}^LG /\!/ \operatorname{conj}.$$

Equality of all character values therefore puts g and h at the same point of that quotient. Since both elements are semisimple, this point determines the semisimple conjugacy class. □

Definition 9413.980 (Tannakian generator)**.** *A finite set*

$$\mathcal{R}_0 = \{r_1, \ldots, r_m\}$$

of finite-dimensional representations of ${}^LG$ is a Tannakian generator when the representation ring is generated by $\mathcal{R}_0$ under $\oplus$, $\otimes$, and ${}^\vee$, and tensor-word characters in these generators separate semisimple conjugacy classes.

Theorem 9413.981 (Generator detection)**.** *Let $\mathcal{R}_0$ be a Tannakian generator. If two semisimple elements $g, h \in {}^LG$ have equal trace under every tensor word in $\mathcal{R}_0$, then they have the same semisimple conjugacy class.*

Proof. Every finite-dimensional character is an integral linear combination of characters of tensor words built from $\mathcal{R}_0$, because $\mathcal{R}_0$ generates the representation ring. Equality on all tensor words therefore implies equality of $\operatorname{tr} r(g)$ and $\operatorname{tr} r(h)$ for every finite-dimensional representation r. The result follows from Definition 9413.979. □

Definition 9413.982 (Tannakian residual dichotomy data)**.** *A finite Tannakian residual is a failure of one displayed finite representation relation, such as*

$$R_\otimes(r_1, r_2) \neq 0,$$

inside a fixed finite representation window. A representation-tail residual is a compatibility failure invisible in every fixed finite representation window but present along unbounded highest weight, tensor depth, multiplicity, local factor complexity, or functorial compatibility complexity. Its ledger is

$$\mathsf{RepTailLedger} := (\mathsf{HighestWeightEscape}, \mathsf{TensorDepthEscape}, \mathsf{MultiplicityEscape}, \mathsf{NonuniformLFactorBound}, \mathsf{Functorial}$$

Theorem 9413.983 (Tannakian residual dichotomy)**.** *If a family $\{\Pi_r\}_{r\in\mathrm{Rep}({}^LG)}$ of GL-transfers does not come from one tensor-compatible ${}^LG$-readout system, then there is a finite Tannakian residual or a representation-tail residual.*

Proof. Failure to come from one tensor-compatible readout means that some relation in the tensor category $\mathrm{Rep}({}^LG)$ is not respected by the family $\{\Pi_r\}$. If the failing relation uses a fixed finite expression in $\oplus$, $\otimes$, and ${}^\vee$, it is a finite Tannakian residual. If every fixed finite expression is compatible but no global tensor-compatible readout exists, the defect is not located in any finite representation window. It can only appear along unbounded representation complexity, which is exactly the representation-tail residual. □

Definition 9413.984 (No hidden Tannakian residual)**.** *The no-hidden Tannakian residual condition says that every failure of representation compatibility is either detected by a finite representation relation or is recorded in* RepTailLedger*, and that a closed accepted system carries no unresolved entry in either place.*

Theorem 9413.985 (Tannakian closure)**.** *Let $\{\Pi_r\}$ be a family of GL-transfers with the BEDC three-properties ledger, no-hidden Tannakian residual, and closedness. Then the family determines a unique semisimple ${}^LG$-readout system on the certified unramified scope.*

Proof. If the family did not determine one tensor-compatible readout system, Definition 9413.983 would give a finite Tannakian residual or a representation-tail residual. A finite residual is detected by a finite representation relation. It must be entered in the ledger, leaving the system not closed, or omitted, violating the ledger property. A tail residual is handled in the same way by RepTailLedger. Both alternatives contradict the closed no-hidden hypotheses. Thus the family is tensor-compatible.

For each certified unramified place v, the compatible character system on $\mathrm{Rep}({}^LG)$ defines a point of ${}^LG//\mathrm{conj}$. Uniqueness follows from Definition 9413.979. □

Theorem 9413.986 (Reconstruction from GL_N transfers)**.** *Let $\mathcal{R}_0$ be a Tannakian generator. Assume that for every tensor word w in $\mathcal{R}_0$, the composed parameter*

$$w \circ \phi$$

has a $GL(V_w)$-automorphic realization Π_w supplied by the GL_N local and global closure bridges. If these transfers satisfy no-hidden Tannakian residual and closedness, then for every certified unramified place v there is a unique semisimple class

$$s_v \in {}^LG//\mathrm{conj}$$

such that

$$\mathrm{tr}\, r(s_v) = \lambda_{v,r}(\Pi_r)$$

for every finite-dimensional representation r in the generated readout scope.

Proof. The GL_N closure bridges give the automorphic readouts for the generators and their tensor words. By Definition 9413.985, no closed compatible system can have a finite or tail Tannakian residual. Therefore the readouts form a tensor-compatible character system on $\mathrm{Rep}({}^LG)$. Tannakian separation gives a unique semisimple class s_v realizing those character values. □

Theorem 9413.987 (General G finite-readout closure)**.** *In the setting of Definition 9413.986, the reconstructed semisimple class agrees with the original parameter readout:*

$$s_v = \phi(\mathrm{Frob}_v)$$

in ${}^L G//\,\mathrm{conj}$ *for every certified unramified place* v*. Hence for every finite-dimensional representation* r *in the readout scope,*

$$\lambda_{v,r}(\Pi_r) = \mathrm{tr}\, r(\phi(\mathrm{Frob}_v)).$$

Proof. Each Π_r was constructed from the composed $GL(V_r)$-parameter $r \circ \phi$, so its unramified readout is

$$\lambda_{v,r}(\Pi_r) = \mathrm{tr}\, r(\phi(\mathrm{Frob}_v)).$$

Thus the reconstructed class s_v and the class $\phi(\mathrm{Frob}_v)$ have the same trace under every generated finite-dimensional representation. By generator detection, they are the same semisimple class. □

Definition 9413.988 (General G existence bridge certificate)**.** *A general* G *existence bridge certificate says that a tensor-compatible system of* GL_N *transfers attached to a parameter* ϕ *either descends to a* G*-side automorphic packet*

$$\Pi_\phi^G$$

or enters the existence ledger

$$\mathsf{ExistenceLedger}_G := (\mathsf{MissingGPacket}, \mathsf{EndoscopicPacketDefect}, \mathsf{ArthurPacketDefect}, \mathsf{ContinuousSpectrumDefect}, \mathsf{Bou}$$

The certificate may be supplied by stable trace formula data, endoscopic classification, Arthur packet construction, a geometric Langlands functor, or a cohomological realization on the declared scope.

Theorem 9413.989 (General G BEDC reciprocity)**.** *Assume:*

- GL_N *transfers exist for a Tannakian generator and its tensor words;*
- *the family satisfies no-hidden Tannakian residual and closedness;*
- *the general* G *existence bridge certificate is present;*
- *no-hidden Langlands residual and closedness hold for the resulting* G*-packet.*

Then there is a closed G*-side automorphic packet* Π_ϕ^G *such that, for almost all certified unramified* v *and all finite-dimensional* $r : {}^L G \to GL(V_r)$ *in the readout scope,*

$$\lambda_{v,r}(\Pi_\phi^G) = \mathrm{tr}\, r(\phi(\mathrm{Frob}_v)).$$

Proof. By Definition 9413.987, the GL_N transfers are compatible with the single parameter readout ϕ. The general G existence bridge certificate says that the compatible system descends to a G-side packet or enters $\mathsf{ExistenceLedger}_G$. An unresolved existence-ledger entry is not compatible with closedness; omitting it violates the ledger property. Hence the packet Π_ϕ^G exists in the closed case.

If a Hecke–Frobenius mismatch remains, the no-hidden Langlands residual condition makes it a finite detector residual or a ledger row. The first contradicts the finite readout closure; the second contradicts closedness while unresolved. The displayed equality follows. □

Theorem 9413.990 (Tannakian functoriality closure)**.** *Let*

$$\eta : {}^L H \to {}^L G$$

be an L-group homomorphism and let (Π_H, ϕ_H) be a closed H-side packet. Put

$$\phi_G := \eta \circ \phi_H.$$

If ϕ_G satisfies the hypotheses of Definition 9413.989, then there is a closed G-side packet Π_G satisfying

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v)))$$

for almost all certified unramified v and all finite-dimensional $r : {}^L G \to GL(V_r)$ in the readout scope.

Proof. Apply Definition 9413.989 to the parameter $\phi_G = \eta \circ \phi_H$. The resulting G-packet has readouts

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\phi_G(\mathrm{Frob}_v)).$$

Substituting the definition of ϕ_G gives the displayed formula. □

Corollary 9413.991 (GL_1 as the minimal Tannakian model)**.** *For ${}^L G = \mathbb{C}^\times$, the representation category is generated by the characters*

$$z \mapsto z^k.$$

Tannakian compatibility is the identity

$$z^{k+\ell} = z^k z^\ell.$$

For the quadratic character attached to $\mathbb{Q}(i)/\mathbb{Q}$, the visible unramified detector

$$\chi_{-4}(\mathrm{Frob}_p) = 1$$

is equivalent to $p = a^2 + b^2$ for odd primes. Thus the two-square readout is the minimal abelian shadow of the general Tannakian detector system.

Proof. For $\mathbb{C}^\times$, every finite-dimensional semisimple representation is a finite direct sum of characters z^k, and tensor product adds exponents. The quadratic character χ_{-4} is the class-field detector for the extension $\mathbb{Q}(i)/\mathbb{Q}$; its unramified value 1 is exactly the splitting condition for odd primes, equivalently $p = a^2 + b^2$. □

Theorem 9413.992 (Tannakian closure boundary)**.** *Tannakian functorial closure proves that compatible GL_N readouts determine the finite detector system of a general G-parameter, and that incompatibility is finite-visible or ledgered as representation tail. It does not by itself construct the missing G-automorphic packet; that is exactly the role of the general G existence bridge.*

Proof. The arguments above are representation-category and ledger arguments: they use GL_N transfers, tensor compatibility, Tannakian separation, and no-hidden residual discipline. The descent from compatible GL_N readouts to an actual G-side automorphic packet is a separate existence statement, represented by Definition 9413.988. □

9413.79 BEDC–Langlands Stable Multiplicity Formula Certificate

The general G existence bridge of section 9413.80 uses a stable multiplicity formula as the step from a nonzero stable parameter distribution to a nonempty automorphic packet. This section isolates that step. The additional datum, absent from the singleton GL_n setting, is the finite component group attached to an Arthur or Langlands parameter. Its finite Fourier theory is the packet-internal detector for multiplicity and labeling residuals.

Definition 9413.993 (Arthur component group)**.** *Let*

$$\psi : L_F \times SL_2(\mathbb{C}) \to {}^L G$$

be a discrete Arthur parameter. Its component group is the finite row

$$\mathcal{S}_\psi := \pi_0 \left(Z_{\widehat{G}}(\operatorname{im}\psi)/Z(\widehat{G})^{\Gamma_F} \right).$$

It records the finite internal symmetry of the packet attached to ψ.

Definition 9413.994 (Packet Fourier ledger)**.** *A packet labeling certificate for ψ consists of a packet candidate*

$$\Pi_\psi(G)$$

and a finite packet character datum, written in the basic case as a map

$$\ell : \Pi_\psi(G) \to \widehat{\mathcal{S}_\psi}.$$

For $\pi \in \Pi_\psi(G)$ write $\chi_\pi := \ell(\pi)$. Failure of packet labeling, component-character normalization, sign convention, or packet membership is recorded in

$$\mathsf{PacketLabelLedger}(\psi).$$

Definition 9413.995 (Packet Fourier distributions)**.** *For a test function f and $s \in \mathcal{S}_\psi$, define the Fourier-twisted packet distribution*

$$\Theta_{\psi,s}(f) := \sum_{\pi \in \Pi_\psi(G)} \chi_\pi(s)\Theta_\pi(f),$$

where

$$\Theta_\pi(f) := \operatorname{tr}\pi(f).$$

The stable packet distribution is a declared finite linear combination

$$S\Theta^G_\psi(f) := \sum_{\pi \in \Pi_\psi(G)} m_\psi(\pi)\Theta_\pi(f),$$

with $m_\psi(\pi)$ determined by the multiplicity formula and recorded in the multiplicity ledger.

Definition 9413.996 (Component group three-properties rows)**.** *The time row of $\mathcal{S}_\psi$ is generated by the chain*

$$\operatorname{im}\psi \longrightarrow Z_{\widehat{G}}(\operatorname{im}\psi) \longrightarrow \pi_0.$$

The unity row is finite Fourier orthogonality on $\mathcal{S}_\psi$:

$$\frac{1}{|\mathcal{S}_\psi|} \sum_{s \in \mathcal{S}_\psi} \chi(s)\overline{\chi'(s)} = \delta_{\chi,\chi'}.$$

The ledger row records packet labels, multiplicities, global signs, transfer factors, endoscopic labels, and normalization choices.

Theorem 9413.997 (Component-group Fourier detection)**.** *Let $\mathcal{S}$ be a finite group. If two coefficient functions*

$$a, b : \widehat{\mathcal{S}} \to \mathbb{C}$$

are distinct, then there exists $s \in \mathcal{S}$ such that

$$\sum_{\chi \in \widehat{\mathcal{S}}} a(\chi)\chi(s) \neq \sum_{\chi \in \widehat{\mathcal{S}}} b(\chi)\chi(s).$$

Proof. Let $c = a - b$. Since $a \neq b$, the function c is nonzero. The finite Fourier transform

$$\widehat{c}(s) := \sum_{\chi \in \widehat{\mathcal{S}}} c(\chi)\chi(s)$$

is invertible on the finite character group. If $\widehat{c}(s) = 0$ for all $s \in \mathcal{S}$, then Fourier inversion gives $c = 0$, a contradiction. Hence some s has $\widehat{c}(s) \neq 0$. □

Corollary 9413.998 (Packet multiplicity mismatch is finite-detected)**.** *If the actual spectral multiplicity function on a packet differs from the multiplicity function predicted by the stable multiplicity formula, then a component-group Fourier detector*

$$s \in \mathcal{S}_\psi$$

detects the mismatch, or the mismatch is a packet ledger row.

Proof. Group packet members by their component character labels. The actual multiplicities and predicted multiplicities give two coefficient functions on $\widehat{\mathcal{S}_\psi}$. If these functions differ, Definition 9413.997 gives an element s whose Fourier-twisted packet distribution distinguishes them. If the packet labels or the grouping needed for this comparison are unavailable, the failure is exactly a packet ledger row. □

Definition 9413.999 (Stable multiplicity formula certificate)**.** *A stable multiplicity formula certificate for ψ consists of:*

- *a packet candidate $\Pi_\psi(G)$;*
- *a packet labeling certificate over $\mathcal{S}_\psi$;*
- *a multiplicity function*
$$m_\psi : \Pi_\psi(G) \to \mathbb{Z}_{\geq 0};$$
- *an identity on the discrete stable scope*
$$S\Theta_\psi^G(f) = \sum_{\pi \in \Pi_\psi(G)} m_\psi(\pi)\Theta_\pi(f) + \mathsf{MultiplicityLedger}_\psi(f),$$
  where the ledger contains component-group data, signs, transfer factors, endoscopy, continuous spectrum, boundary, and normalization rows.

Failure to supply any listed row is a MultiplicityDefectLedger *entry.*

Definition 9413.1000 (No hidden multiplicity residual)**.** *The no-hidden multiplicity residual condition says that every failure of packet membership, packet labeling, component-group Fourier compatibility, multiplicity, stable character identity, or packet normalization is finite-detected by a packet Fourier detector or is recorded in* MultiplicityDefectLedger*, and closed packets carry no unresolved such row.*

Theorem 9413.1001 (Stable distribution nonzero forces packet nonempty)**.** *Assume:*

- $S\Theta_\psi^G(f_0) \neq 0$ *for some fixed test function* f_0*;*
- *the stable multiplicity formula certificate holds;*
- *continuous, boundary, endoscopic, and normalization ledger terms are separated or explicitly ledgered;*
- *the object is closed against unresolved ledger rows.*

Then

$$\Pi_\psi(G) \neq \varnothing.$$

Proof. By the stable multiplicity formula,

$$S\Theta_\psi^G(f_0) = \sum_{\pi \in \Pi_\psi(G)} m_\psi(\pi)\Theta_\pi(f_0) + \mathsf{MultiplicityLedger}_\psi(f_0).$$

The ledger terms are separated or explicitly recorded. If $\Pi_\psi(G)$ were empty, the packet sum would be zero. The nonzero value of $S\Theta_\psi^G(f_0)$ would then have to come entirely from a ledger term: continuous spectrum, boundary, endoscopy, normalization, or multiplicity defect. Carrying such a row unresolved contradicts closedness, while omitting it violates the ledger property. Therefore a closed nonzero stable distribution has a nonempty packet. □

Theorem 9413.1002 (Missing packet forces ledger contradiction)**.** *Let* ψ *be a closed three-properties parameter with*

$$S\Theta_\psi^G(f_0) \neq 0.$$

If $\Pi_\psi(G) = \varnothing$*, then an unresolved ledger residual is present.*

Proof. With an empty packet, the nonzero stable distribution cannot be accounted for by packet characters. The source of the nonzero readout must be endoscopic transfer, continuous spectrum, Eisenstein boundary, packet-label failure, multiplicity formula defect, or normalization data. These are precisely ledger rows. If the row is carried, the object is not closed; if it is omitted, the ledger property fails. □

Definition 9413.1003 (Endoscopic packet transfer residual)**.** *Let* $\xi : {}^LH \to {}^LG$ *be an endoscopic embedding and suppose*

$$\psi = \xi \circ \psi_H.$$

For matching test functions $f^H \leftrightarrow f^G$*, the endoscopic packet transfer residual is*

$$R_{\mathrm{endo}}(f^G, f^H) := S\Theta_\psi^G(f^G) - S\Theta_{\psi_H}^H(f^H).$$

Theorem 9413.1004 (Endoscopic transfer residual detection)**.** *If* R_{endo} *is nonzero, then it is detected on a fixed matching test window or enters the endoscopic tail ledger.*

Proof. If some fixed matching pair $f^H \leftrightarrow f^G$ has

$$R_{\mathrm{endo}}(f^G, f^H) \neq 0,$$

then the residual is finite-visible. If all fixed matching windows are clean but transfer still fails globally, the failure lies in singular orbital limits, ramification escape, transfer-factor normalization, boundary, or continuous spectrum. These are exactly endoscopic tail ledger rows. □

Theorem 9413.1005 (Endoscopic induction with multiplicity)**.** *Assume proper endoscopic groups have the existence bridge, endoscopic transfer residuals are no-hidden, and the relevant stable multiplicity formulae are closed. If*

$$\psi = \xi \circ \psi_H$$

comes from a proper endoscopic group H, then the G-side packet $\Pi_\psi(G)$ exists in the closed scope.

Proof. The induction hypothesis gives a nonempty H-side packet $\Pi_{\psi_H}(H)$. Endoscopic transfer carries its stable distribution to the G-side unless a transfer residual occurs. By Definition 9413.1004, such a residual is finite-visible or ledgered as endoscopic tail; both alternatives contradict closedness when unresolved. Hence transfer succeeds. The stable multiplicity formula on the G-side then records the transferred packet with its packet labels and multiplicities. □

Theorem 9413.1006 (General G existence bridge, multiplicity form)**.** *Assume:*

- *a parameter state certificate gives*

$$S\Theta^G_\psi(f_0) \neq 0$$

  on a fixed window;
- *the stable multiplicity formula certificate holds;*
- *endoscopic transfer and endoscopic induction certificates hold;*
- *trace tail ledger rows are complete;*
- *no-hidden multiplicity residual and closedness hold.*

Then the closed parameter ψ has

$$\Pi_\psi(G) \neq \varnothing.$$

Proof. If the primitive discrete contribution of $S\Theta^G_\psi(f_0)$ is nonzero, then Definition 9413.1001 gives a nonempty G-packet. If the nonzero contribution is a proper endoscopic contribution, then Definition 9413.1005 gives the G-packet. If neither alternative holds, the nonzero readout belongs to continuous spectrum, Eisenstein boundary, singular orbit, normalization, or multiplicity-defect data. These are trace or multiplicity ledger rows, and closedness excludes unresolved such rows. □

Theorem 9413.1007 (Stable multiplicity reciprocity)**.** *Under the hypotheses of Definition 9413.1006, plus Tannakian separation, Chebotarev, and no-hidden Langlands residual, there is $\pi \in \Pi_\psi(G)$ such that*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v))$$

for almost all certified unramified places v and all finite-dimensional readouts $r : {}^LG \to GL(V_r)$ in scope.

Proof. The multiplicity-form bridge gives a nonempty packet. Choose $\pi \in \Pi_\psi(G)$. Any unramified mismatch is detected by the finite Tannakian detector (v, r). If no such detector sees a mismatch, Tannakian separation and Chebotarev fix the semisimple unramified readout; any remaining discrepancy is ramified, endoscopic, continuous, boundary, multiplicity, or normalization ledger data. No-hidden residual and closedness exclude unresolved residuals. □

Theorem 9413.1008 (Stable multiplicity certificate boundary)**.** *The stable multiplicity formula certificate closes packet nonemptiness from a nonzero stable distribution once packet labels, component-group Fourier data, multiplicities, endoscopic transfer, and ledger rows are supplied. It does not construct Arthur packets, prove the stable multiplicity formula, or prove endoscopic transfer.*

Proof. The arguments are finite Fourier, linear distribution, and ledger arguments after the traditional packet and multiplicity data are present. The construction of those data remains the independent mathematical input. □

9413.80 BEDC–Langlands General G Existence Bridge

The Tannakian functorial closure of section 9413.78 reduces the finite readout problem for a general reductive group G to compatible GL_N transfers. It does not by itself construct the G-side automorphic packet. The existence bridge for general G is the stable-trace and endoscopic part of the Langlands machine: a nonzero stable parameter readout must appear as a primitive G-packet, as a proper endoscopic transfer, or as a declared trace tail ledger row.

Definition 9413.1009 (General G parameter object)**.** *Let F be a global field and let G/F be a connected reductive group. A general G parameter object is an Arthur-type row*

$$\psi : L_F \times SL_2(\mathbb{C}) \to {}^L G,$$

with ordinary Langlands parameters

$$\phi : W_F' \to {}^L G$$

included by taking the SL_2 factor to be trivial. Its BEDC ledger is

$$\Lambda_\psi := (S_{\mathrm{ram}}, N_v, \epsilon_v, \gamma_v, \mathcal{S}_\psi, \mathsf{Endoscopy}, \mathsf{ContinuousSpectrum}, \mathsf{Boundary}, \mathsf{Normalization}),$$

where

$$\mathcal{S}_\psi := \pi_0(Z_{\widehat{G}}(\operatorname{im}\psi)).$$

The time row is generated by Frobenius, local parameters, Arthur SL_2, and trace-window inclusions; the unity row is the stable semisimple or tempered readout; the ledger row is Λ_ψ.

Definition 9413.1010 (G-side automorphic realization residual)**.** *A G-side automorphic packet $\Pi_\psi(G)$ realizes ψ when, for almost all unramified places v, some $\pi \in \Pi_\psi(G)$ has*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v))$$

for every finite-dimensional representation $r : {}^L G \to GL(V_r)$ in the certified readout scope. If no such G-packet is present, the missing realization residual is

$$E_G(\psi) := \mathsf{MissingGAutomorphicPacket}(\psi).$$

Definition 9413.1011 (Stable parameter distribution certificate)**.** *A stable parameter distribution certificate for ψ assigns to every fixed Hecke window $\mathcal{W}$ a stable functional*

$$S\Theta_\psi^G : \mathcal{H}_{\mathcal{W}}(G) \to \mathbb{C}$$

and a parameter projector $f_{\psi,\mathcal{W}}^G$ such that

$$S\Theta_\psi^G(f_{\psi,\mathcal{W}}^G) \neq 0$$

on the declared nonzero scope. At unramified places this is Satake evaluation at $\psi(\mathrm{Frob}_v)$; at ramified places it uses the local packet stable character and its local ledger.

Definition 9413.1012 (Stable multiplicity formula certificate)**.** *A stable multiplicity formula certificate for a discrete parameter ψ gives a packet candidate $\Pi_\psi(G)$ and a component-group pairing so that the target stable distribution has the form*

$$S\Theta_\psi^G(f) = \sum_{\pi\in\Pi_\psi(G)} m(\pi)\Theta_\pi(f) + \mathsf{LedgerTerms}_\psi(f),$$

where $\mathsf{LedgerTerms}_\psi$ *consists only of continuous spectrum, Eisenstein residual, endoscopic transfer, boundary, multiplicity normalization, and transfer-factor rows.*

Definition 9413.1013 (Endoscopic transfer certificate)**.** *An endoscopic datum is a row*

$$\mathfrak{e} = (H, \xi, s, \Delta)$$

with an L-embedding

$$\xi : {}^LH \to {}^LG.$$

An endoscopic transfer certificate assigns matching test functions

$$f^H \longleftrightarrow f^G$$

such that stable orbital integrals match after the declared transfer factor and the spectral stable distributions satisfy

$$S\Theta_{\psi_H}^H(f^H) = S\Theta_{\xi\circ\psi_H}^G(f^G) + \mathsf{EndoscopicLedger}(f).$$

Transfer-factor, normalization, nonregular, ramified, and matching failures are rows of $\mathsf{EndoscopicLedger}$*.*

Definition 9413.1014 (Stabilized trace formula certificate)**.** *A stabilized trace formula certificate for G records, for admitted test functions f^G, an identity*

$$I_{\mathrm{disc}}^G(f^G) = \sum_{\mathfrak{e}=(H,\xi)} \iota(G,H) SI_{\mathrm{disc}}^H(f^H) + \mathsf{TraceLedger}(f^G),$$

where $\mathsf{TraceLedger}$ *contains continuous spectrum, Eisenstein terms, boundary, singular orbital integrals, ramification, normalization, and transfer-factor data. Its finite/tail residual discipline is the trace defect dichotomy of Definition 9413.1045.*

Definition 9413.1015 (Endoscopic induction certificate)**.** *Endoscopic complexity is ordered by a well-founded tuple such as*

$$\mathrm{ec}(G) := (\mathrm{ssrank}\, G, \dim G, \#\pi_0(Z(\widehat{G}))).$$

An endoscopic induction certificate says that every proper elliptic endoscopic group H appearing in the stabilized trace formula has smaller complexity than G, and that the general existence bridge is already available for H on the declared scope.

Definition 9413.1016 (General trace tail ledger)**.** *The general G trace tail ledger is*

$$\mathsf{GeneralTraceTailLedger}_G := (\mathsf{ContinuousSpectrum}, \mathsf{EisensteinResidual}, \mathsf{Boundary}, \mathsf{SingularOrbitalIntegral}, \mathsf{TransferNorm}$$

A trace-tail ledger certificate says that any nonzero stable readout not accounted for by a primitive packet or a proper endoscopic packet enters this ledger.

Theorem 9413.1017 (Endoscopic residual dichotomy)**.** *If an endoscopic transfer certificate fails for an admitted transfer row, then the failure is either a finite transfer defect on a fixed test window or an endoscopic tail defect.*

Proof. If the mismatch is visible for a fixed pair of matching test functions $f^H \leftrightarrow f^G$, then the stable orbital-integral or spectral transfer equality fails on a finite window. That is a finite transfer defect. If every fixed window transfer matches but the transfer fails along singular orbit limits, ramification growth, transfer-factor normalization limits, or cofinal window growth, the defect is not finite-window data. It is an endoscopic tail defect and belongs to the endoscopic tail ledger. □

Corollary 9413.1018 (Closed objects have no endoscopic transfer failure)**.** *A closed BEDC trace object with complete endoscopic ledger discipline cannot carry an unresolved endoscopic transfer failure.*

Proof. By Definition 9413.1017, any failure is finite-visible or a tail row. A finite failure must be entered in the ledger or omitted; the first leaves the object not closed, and the second violates the ledger property. The same argument applies to a tail row. □

Theorem 9413.1019 (Stabilized trace residual dichotomy)**.** *If the stabilized trace formula identity for G fails on the declared scope, then the residual is visible on a fixed trace window or belongs to* $\mathsf{GeneralTraceTailLedger}_G$.

Proof. This is the stable form of Definition 9413.1045. A nonzero restricted residual on some finite trace window is finite-visible. If every finite window is clean but the stabilized identity fails globally, the defect can only lie in continuous spectrum, Eisenstein boundary, singular orbital integrals, ramification escape, transfer normalization, or endoscopic tail behavior, all of which are entries of $\mathsf{GeneralTraceTailLedger}_G$. □

Theorem 9413.1020 (General G existence bridge certificate)**.** *Let $\psi^\uparrow$ be a legal discrete G-parameter object. Assume:*

- *local packet and local state certificates hold at every displayed place;*
- *a stable parameter distribution certificate gives a nonzero fixed-window readout;*
- *the stabilized trace formula certificate holds;*
- *endoscopic transfer certificates hold for all proper endoscopic data in the trace identity;*
- *the stable multiplicity formula certificate holds for primitive G-discrete contributions;*
- *the endoscopic induction certificate holds;*
- *the general trace tail ledger is complete;*
- *closed accepted objects carry no unresolved trace, endoscopic, or existence ledger rows.*

Then $\psi^\uparrow$ has a G-side automorphic packet realization, or its absence is an explicit ledger row. In the closed case, the packet exists.

Proof. Choose a fixed window and a test function

$$f^G_{\psi,\mathcal{W}}$$

with

$$S\Theta^G_\psi(f^G_{\psi,\mathcal{W}}) \neq 0.$$

The stabilized trace formula and endoscopic transfer certificates place this nonzero stable readout into exactly three classes.

First, the primitive G-discrete part may be nonzero. The stable multiplicity formula then expresses that primitive stable distribution as a sum over $\Pi_\psi(G)$ plus declared ledger terms. If $\Pi_\psi(G)$ were empty and the primitive readout were nonzero, the nonzero contribution would have to be a ledger term. A closed object cannot carry it unresolved, and omission violates the ledger property. Thus the primitive nonzero case gives a nonempty G-packet.

Second, the stable readout may come from a proper endoscopic group H. By the endoscopic induction certificate, the existence bridge for H supplies an H-side packet for the corresponding parameter ψ_H. The endoscopic transfer certificate carries that packet to the G-side unless a transfer residual occurs. By Definition 9413.1018, an unresolved transfer residual is incompatible with closedness. Thus the endoscopic case gives a G-side packet in the closed scope.

Third, the readout may be neither primitive discrete nor proper endoscopic discrete. Then it lies in continuous spectrum, Eisenstein residual, boundary, singular orbital-integral, normalization, ramification, or endoscopic-tail data. These are exactly the rows of $\mathsf{GeneralTraceTailLedger}_G$. A closed object cannot carry them unresolved.

The three alternatives exhaust the stabilized spectral decomposition. Therefore a closed legal parameter has a G-side automorphic packet. Without closedness, failure is precisely a ledger row. □

Corollary 9413.1021 (General G reciprocity from the existence bridge)**.** *Under the hypotheses of Definition 9413.1020, if no-hidden Langlands residual holds, then there is a closed packet* $\Pi_\psi(G)$ *such that for almost all certified unramified places* v *and all finite-dimensional* $r : {}^LG \to GL(V_r)$ *in the readout scope,*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v))$$

for some $\pi \in \Pi_\psi(G)$*.*

Proof. The existence bridge gives a nonempty G-packet in the closed case. If a finite Hecke–Frobenius mismatch remains, it is detected by the Tannakian detector (v, r) of Definition 9413.977. If no finite detector sees it, then by Tannakian separation and Chebotarev the unramified semisimple readout is already fixed; any remaining obstruction is a ramified, endoscopic, continuous-spectrum, boundary, multiplicity, or normalization ledger row. No-hidden residual and closedness exclude both unresolved alternatives. □

Corollary 9413.1022 (General G functoriality from the existence bridge)**.** *Let*

$$\eta : {}^LH \to {}^LG$$

be an L*-group homomorphism and let* (Π_H, ψ_H) *be a closed* H*-side packet. Put*

$$\psi_G := \eta \circ \psi_H.$$

If ψ_G *satisfies the hypotheses of Definition 9413.1020, then there is a* G*-side packet* Π_G *such that*

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\psi_H(\mathrm{Frob}_v)))$$

on the certified unramified finite detector scope.

Proof. The parameter $\psi_G = \eta \circ \psi_H$ is a legal G-parameter on the declared scope. The existence bridge gives a G-side packet, and the preceding reciprocity corollary gives its finite readout equality with ψ_G. Substituting the definition of ψ_G yields the displayed formula. □

Theorem 9413.1023 (General G BEDC–Langlands closure)**.** *On any reductive-group scope carrying local packets, parameter states, stabilized trace formula, endoscopic transfer, stable multiplicity, endoscopic induction, Tannakian separation, Chebotarev, no-hidden Langlands residual, trace-tail ledger discipline, and closedness, Langlands reciprocity and functoriality hold on the certified scope.*

Proof. Existence is supplied by Definition 9413.1020. Once the G-packet exists, the detector equality is forced by Definition 9413.1021. Functoriality is obtained by applying the same existence and no-hidden residual argument to $\eta \circ \psi_H$ as in Definition 9413.1022. Uniqueness of the semisimple parameter readout follows from the Tannakian and Chebotarev rows of Definition 9413.861. □

Theorem 9413.1024 (General G existence bridge boundary)**.** *The general G existence bridge is a conditional BEDC closure theorem. It consumes stable trace formula, endoscopic transfer, stable multiplicity, local packet, parameter-state, induction, and tail-ledger certificates. It does not construct those certificates or remove the traditional analytic and endoscopic obligations.*

Proof. Every positive implication above invokes one of the listed certificates. The argument proves that, once they are present, absence of a G-packet cannot remain hidden: it is primitive discrete, proper-endoscopic, or ledgered as trace tail. The construction of the certificates themselves is outside this bridge. □

9413.81 BEDC–Langlands Compact Discrete Spectrum Core

The cleanest analytic existence setting is the compact quotient, or more generally a scope where the automorphic spectrum under consideration is purely discrete with finite multiplicities. In that setting the spectral side of a fixed Hecke window is an honest Hilbert direct sum, and a nonzero value of the finite-window projector cannot be hidden in continuous spectrum, Eisenstein boundary, or endoscopic transfer terms.

Definition 9413.1025 (Compact discrete automorphic Hilbert row)**.** *Fix a global field F and a connected reductive group G. A compact discrete automorphic Hilbert row consists of a closed central-character scope for which*

$$G(F)\backslash G(\mathbb{A}_F)$$

is compact after the declared central reduction, or else a certified subscope whose relevant automorphic spectrum is purely discrete. The Hilbert object is

$$\mathcal{H}_{\mathrm{aut}} := L^2_{\mathrm{disc}}\big(G(F)\backslash G(\mathbb{A}_F)\big),$$

and, away from a finite bad set S_0, the unramified Hecke algebra is

$$\mathcal{H}^{S_0} := \bigotimes_{v \notin S_0}{}' \mathcal{H}_v(G, K_v).$$

The discrete spectral decomposition on the unramified fixed-vector scope has the form

$$\mathcal{H}_{\mathrm{aut}}^{K^{S_0}} \cong \widehat{\bigoplus}_{\Pi} m(\Pi)\Pi^{K^{S_0}},$$

with finite multiplicities $m(\Pi)$ on the displayed scope.

Definition 9413.1026 (Compact discrete ledger)**.** *The compact discrete ledger records every term excluded from the clean Hilbert row:*

$$\mathsf{ContinuousSpectrum},\quad \mathsf{EisensteinBoundary},\quad \mathsf{ResidualBoundary},\quad \mathsf{EndoscopicTransfer},$$

$$\mathsf{RamifiedLocalRow},\quad \mathsf{ArchimedeanRow},\quad \mathsf{NormalizationRow}.$$

The compact discrete core is closed only on scopes where these rows are absent, separated by the test function, or explicitly carried as ledger obligations outside the claim.

Definition 9413.1027 (Hecke spectral fingerprint)**.** *For a discrete automorphic packet Π unramified outside S_0, its unramified Hecke fingerprint is*

$$x_\Pi := \big(s_v(\Pi)\big)_{v\notin S_0}.$$

For an allowed parameter packet $\phi^\uparrow$, its Frobenius fingerprint is

$$x_\phi := \big(\phi(\mathrm{Frob}_v)\big)_{v\notin S_0}.$$

On a finite Hecke window $\mathcal{W} = (S, R, \mathcal{B})$, write $x_{\Pi,\mathcal{W}}$ and $x_{\phi,\mathcal{W}}$ for the corresponding finite restrictions.

Definition 9413.1028 (Compact finite spectral packet)**.** *For a finite Hecke window $\mathcal{W}$ and a finite complexity bound B, let*

$$\mathsf{Aut}^{\mathrm{disc}}_{\mathcal{W},B} = \{\Pi_1, \dots, \Pi_M\}$$

be the finite list of discrete automorphic packets admitted by the window and the bound. The bounded finiteness row records that this list is finite up to the closed automorphic equivalence relation.

The parameter state row used below is the certificate of section 9413.74, specialized to the compact discrete window. In the unramified part it is evaluation at $x_{\phi,\mathcal{W}}$; ramified and archimedean state choices remain ParameterLedger rows.

Theorem 9413.1029 (Fixed-window spectral realization)**.** *Let $f_{\phi,\mathcal{W},B}$ be the finite Hecke projector supplied by Definition 9413.910 on a compact finite spectral packet. If*

$$I_{\mathrm{spec}}(f_{\phi,\mathcal{W},B}) \neq 0$$

and the compact discrete ledger rows outside the target packet are absent or separated, then some

$$\Pi \in \mathsf{Aut}^{\mathrm{disc}}_{\mathcal{W},B}$$

satisfies

$$x_{\Pi,\mathcal{W}} = x_{\phi,\mathcal{W}}.$$

Proof. On the compact discrete scope the spectral side is the finite window sum

$$I_{\mathrm{spec}}(f_{\phi,\mathcal{W},B}) = \sum_{i=1}^{M} m(\Pi_i)\,\mathrm{tr}\,\Pi_i(f_{\phi,\mathcal{W},B}) + \mathrm{LedgerTerms}\,.$$

The non-target ledger terms are absent or separated by hypothesis. Hence the displayed discrete packet sum is nonzero. If no Π_i matched $x_{\phi,\mathcal{W}}$, then the projector property would give

$$\mathrm{tr}\,\Pi_i(f_{\phi,\mathcal{W},B}) = 0$$

for every i, forcing the spectral value to vanish. This contradicts the nonzero spectral readout. Therefore at least one packet matches the parameter on the fixed window. □

Theorem 9413.1030 (Parameter state and trace give fixed-window realization)**.** *Assume a parameter state certificate for $\phi^{\uparrow}$ on $\mathcal{W}$, the stable trace-window equality*

$$I^{\mathrm{st}}_{\mathrm{spec}}(f) = I^{\mathrm{st}}_{\mathrm{geom}}(f),$$

and the geometric-side identification

$$I^{\mathrm{st}}_{\mathrm{geom}}(f_{\phi,\mathcal{W},B}) = \Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W},B}).$$

If the compact discrete ledger rows outside the target packet are absent or separated, then $\phi^{\uparrow}$ has a fixed-window automorphic realization on $\mathcal{W}$.

Proof. The parameter state certificate gives

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W},B}) = 1,$$

so the geometric side is nonzero on the projector. The stable trace-window equality transfers this nonzero value to the spectral side. The fixed-window spectral realization theorem then supplies a discrete automorphic packet with

$$x_{\Pi,\mathcal{W}} = x_{\phi,\mathcal{W}}.$$

□

Definition 9413.1031 (Compact cofinal realization sequence)**.** *Let*

$$\mathcal{W}_1 \subset \mathcal{W}_2 \subset \cdots$$

be a cofinal Hecke window tower. A compact cofinal realization sequence is a sequence of discrete automorphic packets (Π_n) such that

$$x_{\Pi_n,\mathcal{W}_n} = x_{\phi,\mathcal{W}_n}$$

for every n.

Theorem 9413.1032 (Compact bounded subsequence globalization)**.** *Let (Π_n) be a compact cofinal realization sequence. If bounded automorphic finiteness holds and some infinite subsequence has bounded complexity, then there is a global automorphic packet Π_ϕ matching ϕ on every certified unramified finite detector.*

Proof. This is the compact discrete instance of Definition 9413.1072. Bounded finiteness gives only finitely many closed packet classes below the chosen complexity bound, so one class occurs infinitely often. Every fixed detector eventually lies in the cofinal tower, and a sufficiently late occurrence of the recurring packet matches the parameter on that detector. Thus the recurring packet matches all certified unramified finite detectors. □

Theorem 9413.1033 (Compact unbounded sequence is a tail row)**.** *If a compact cofinal realization sequence has no bounded-complexity infinite subsequence, then it enters the automorphic tail ledger.*

Proof. For every finite complexity bound B, only finitely many terms of the sequence lie below B. Hence complexity escapes along the cofinal tower. In the compact discrete scope, such escape can only appear as level or conductor growth, archimedean type growth, ramified local type growth, multiplicity bookkeeping growth, or departure from the compact discrete scope into a ledgered endoscopic, boundary, or continuous row. These are exactly the automorphic tail rows. □

Theorem 9413.1034 (Compact cofinal window closure)**.** *Assume every window of a cofinal tower has a compact fixed-window realization, bounded automorphic finiteness holds, the automorphic tail ledger is complete, and closed accepted packets cannot carry unresolved tail rows. Then $\phi^{\uparrow}$ has a global automorphic realization.*

Proof. Choose a compact cofinal realization sequence. If it has a bounded infinite subsequence, the compact bounded subsequence globalization theorem produces the global packet. If it has no such subsequence, the compact unbounded sequence theorem places it in the automorphic tail ledger. An unrecorded tail violates ledger discipline, and an unresolved recorded tail violates closedness. Hence the tail alternative is unavailable for a closed accepted packet, and the global realization alternative remains. □

Theorem 9413.1035 (Compact discrete BEDC–Langlands existence core)**.** *On a compact discrete automorphic scope, assume:*

- *parameter state certificates on a cofinal Hecke window tower;*
- *stable trace-window equality on each window;*
- *finite Hecke projector certificates;*
- *bounded automorphic finiteness;*
- *a complete automorphic tail ledger;*
- *no-hidden Langlands residual and closedness against unresolved ledger rows.*

Then there is a closed automorphic packet Π_ϕ such that

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

for every certified unramified finite detector (v, r).

Proof. The parameter state and trace theorem gives a fixed-window realization on every window of the cofinal tower. The compact cofinal window closure theorem globalizes these realizations to a packet Π_ϕ. If a Hecke–Frobenius mismatch remains, the no-hidden residual condition makes it either a finite detector residual or a ledger residual. A finite detector residual contradicts the cofinal matching unless it is an unresolved row; carrying such a row violates closedness, while omitting it violates ledger discipline. The same argument applies to ramified, endoscopic, continuous, boundary, archimedean, and normalization ledger residuals. Hence the closed packet has no remaining mismatch. □

Theorem 9413.1036 (Compact discrete functorial core)**.** *Let*

$$\eta : {}^L H \to {}^L G$$

be an L-group homomorphism. If the target parameter

$$\phi_G := \eta \circ \phi_H$$

satisfies the compact discrete existence core on the G-side, then there is a G-side automorphic packet Π_G with

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v)))$$

for every certified unramified finite detector on the target scope.

Proof. Apply Definition 9413.1035 to ϕ_G. The resulting packet satisfies

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\phi_G(\mathrm{Frob}_v)).$$

Substituting $\phi_G = \eta \circ \phi_H$ gives the displayed formula. □

Theorem 9413.1037 (Compact discrete boundary)**.** *The compact discrete core closes the passage from nonzero fixed-window spectral readout to global automorphic realization only on scopes where the spectral side is purely discrete or where all non-discrete terms are separated into ledger rows. It does not prove the stable trace formula, construct parameter state certificates, or settle the general continuous, Eisenstein, endoscopic, ramified, archimedean, or boundary case.*

Proof. The argument uses the Hilbert direct-sum spectral decomposition and the finite Hecke projector. If continuous spectrum, Eisenstein series, endoscopy, ramification, or boundary terms are present, they are outside the compact discrete row and must be handled by the ledger mechanisms of the general trace-window bridge. The compact result therefore supplies a closed core case, not the full analytic trace formula. □

9413.82 BEDC–Langlands Stable Trace Formula Certificate

The stable trace formula is treated here as a BEDC closure certificate. The claim is not that the analytic technology is avoided. The claim is that, once the trace object is presented with its finite windows and tail ledgers, a failure of the trace identity has only two possible forms: it is visible on a fixed finite trace window, or it is a trace tail row. A closed trace object can carry neither unresolved form.

Definition 9413.1038 (Trace object)**.** *Let G be a reductive group over a global field F, and let*

$$f \in C_c^\infty(G(\mathbb{A}_F))$$

be an admitted test function. A trace object is the row

$$\mathrm{TF}^\uparrow(G,F) := (I_{\mathrm{spec}}, I_{\mathrm{geom}}, \Lambda_{\mathrm{TF}}),$$

where

$$I_{\mathrm{spec}}(f) = \sum_{\Pi} m(\Pi) \operatorname{tr} \Pi(f) + I_{\mathrm{cont}}(f)$$

and

$$I_{\mathrm{geom}}(f) = \sum_{\gamma} a(\gamma) O_\gamma(f) + I_{\mathrm{sing}}(f).$$

The trace ledger Λ_{TF} records ramification, normalization, orbital singularities, endoscopy, continuous spectrum, boundary terms, transfer factors, and multiplicity rows.

Definition 9413.1039 (Trace three-properties rows)**.** *The time row of the trace object is generated by convolution, Hecke-time operators, and trace-window inclusions*

$$\mathcal{W}_1 \subset \mathcal{W}_2 \subset \cdots .$$

The unity row says that spectral and geometric expansions are two readouts of the same trace pairing. The ledger row says that every continuous, endoscopic, boundary, singular, ramified, multiplicity, or normalization residual is recorded before any closed trace identity is claimed.

Definition 9413.1040 (Trace residual)**.** *The trace residual is the linear functional*

$$R_{\mathrm{TF}}(f) := I_{\mathrm{spec}}(f) - I_{\mathrm{geom}}(f).$$

A stable trace formula certificate is closed on a scope only if

$$R_{\mathrm{TF}}(f) = 0$$

for every admitted test function on that scope, after all ledger rows have been separated, cancelled, or discharged.

Definition 9413.1041 (Finite trace window)**.** *A finite trace window is a tuple*

$$\mathcal{W} = (S, B, C),$$

where S is a finite place set, B is a spectral complexity bound, and C is a geometric orbital complexity bound. Its admitted test space is

$$\mathcal{T}_{\mathcal{W}} \subset C_c^\infty(G(\mathbb{A}_F)).$$

The window carries a finite spectral list

$$\mathsf{Spec}_{\mathcal{W}} = \{\Pi_1, \dots, \Pi_M\}$$

and a finite geometric list

$$\mathsf{Geom}_{\mathcal{W}} = \{\gamma_1, \dots, \gamma_N\},$$

after all terms outside the window have been placed in the trace ledger.

Theorem 9413.1042 (Finite trace residual detection)**.** *If the restricted residual*

$$R_{\mathrm{TF}}|_{\mathcal{T}_{\mathcal{W}}}$$

is nonzero on a fixed finite trace window, then there is

$$f \in \mathcal{T}_{\mathcal{W}}$$

with

$$R_{\mathrm{TF}}(f) \neq 0.$$

Proof. The restriction of R_{TF} to $\mathcal{T}_{\mathcal{W}}$ is a linear functional. A nonzero linear functional is nonzero on at least one vector of its domain. That vector is the required test function. □

Definition 9413.1043 (Finite trace matrix residual)**.** *On a finite trace window, the spectral and geometric sides have the finite forms*

$$I_{\mathrm{spec}}(f) = \sum_{i=1}^{M} m_i \Theta_i(f), \qquad \Theta_i(f) := \operatorname{tr} \Pi_i(f),$$

and

$$I_{\mathrm{geom}}(f) = \sum_{j=1}^{N} a_j O_j(f).$$

Thus the finite trace residual is

$$R_{\mathrm{TF}}(f) = \sum_{i=1}^{M} m_i \Theta_i(f) - \sum_{j=1}^{N} a_j O_j(f).$$

Definition 9413.1044 (Trace defect dichotomy rows)**.** *A finite trace defect is a window $\mathcal{W}$ such that*

$$R_{\mathrm{TF}}|_{\mathcal{T}_{\mathcal{W}}} \neq 0.$$

A trace tail defect is a residual that vanishes on every fixed finite trace window but remains nonzero in a growing-window or limiting operation. Tail rows include continuous spectrum, Eisenstein boundary, endoscopic transfer, singular orbital integrals, ramification escape, and measure or transfer-factor normalization.

Theorem 9413.1045 (Trace defect dichotomy)**.** *If*

$$R_{\mathrm{TF}} \neq 0,$$

then the trace object has either a finite trace defect or a trace tail defect.

Proof. If there is a finite trace window $\mathcal{W}$ with nonzero restricted residual, then the first alternative holds. Otherwise the residual vanishes on every fixed finite trace window. Since the total residual is assumed nonzero, it cannot be represented by any fixed finite window. It therefore lies in a growing-window or limiting row: continuous spectrum, Eisenstein boundary, endoscopy, singular orbits, ramification escape, or normalization. This is precisely the trace tail alternative. □

Definition 9413.1046 (Trace tail ledger certificate)**.** *A trace tail ledger certificate says that every trace tail defect is recorded in*

$$\mathrm{TraceTailLedger} = (\mathsf{ContinuousSpectrum}, \mathsf{EisensteinBoundary}, \mathsf{Endoscopy}, \mathsf{SingularOrbit}, \mathsf{Ramification}, \mathsf{MeasureNormal}$$

The row is closed only when the recorded defect is discharged or separated from the claimed stable identity.

Theorem 9413.1047 (No hidden trace defect)**.** *If a trace object carries the trace three-properties rows, a complete trace tail ledger certificate, and closedness against unresolved trace ledger rows, then*

$$R_{\mathrm{TF}} = 0.$$

Proof. Assume $R_{\mathrm{TF}} \neq 0$. By Definition 9413.1045, there is a finite trace defect or a trace tail defect.

In the finite case, Definition 9413.1042 supplies a fixed test function f with $R_{\mathrm{TF}}(f) \neq 0$. This is a detected trace residual. Ledger discipline requires it to be recorded; carrying it unresolved violates closedness, and omitting it violates the ledger row.

In the tail case, the trace tail ledger certificate requires the defect to be recorded in TraceTailLedger. Again, carrying it unresolved violates closedness, while omitting it violates the ledger row. Both alternatives contradict the closed trace object. Hence the residual is zero. □

Theorem 9413.1048 (Compact trace identity)**.** *Let $X = \Gamma\backslash G$ be a compact quotient of a locally compact group by a discrete cocompact subgroup, and let*

$$f \in C_c^\infty(G).$$

For the right regular operator

$$R(f)\varphi(x) = \int_G f(g)\varphi(xg)\,dg,$$

one has

$$\operatorname{Tr} R(f) = \sum_\pi m(\pi)\operatorname{tr}\pi(f) = \sum_{[\gamma]} \operatorname{vol}(\Gamma_\gamma\backslash G_\gamma) O_\gamma(f).$$

Proof. The kernel of $R(f)$ is

$$K_f(x,y) = \sum_{\gamma\in\Gamma} f(x^{-1}\gamma y).$$

Since X is compact and f is compactly supported, $R(f)$ is trace class. The spectral decomposition of $L^2(X)$ gives

$$\operatorname{Tr} R(f) = \sum_{\pi} m(\pi) \operatorname{tr} \pi(f).$$

The same trace is the diagonal integral of the kernel:

$$\operatorname{Tr} R(f) = \int_{\Gamma\backslash G} K_f(x,x)\, dx.$$

Substituting the kernel, exchanging the finite-on-support sum with the integral, and grouping by conjugacy classes gives

$$\int_{\Gamma\backslash G} \sum_{\gamma\in\Gamma} f(x^{-1}\gamma x)\, dx = \sum_{[\gamma]} \operatorname{vol}(\Gamma_\gamma\backslash G_\gamma) O_\gamma(f).$$

Thus the spectral and geometric readouts are equal to the same operator trace. □

Corollary 9413.1049 (Compact trace object closure)**.** *In the compact quotient setting of Definition 9413.1048,*

$$R_{\mathrm{TF}}(f) = 0$$

for every admitted test function f.

Proof. Both the spectral and geometric sides are equal to $\operatorname{Tr} R(f)$, so their difference is zero. □

Definition 9413.1050 (Stable trace formula certificate)**.** *A stable trace formula certificate is the package consisting of:*

- *a compact trace identity base or a justified truncated kernel identity;*
- *a complete ledger for continuous spectrum, Eisenstein boundary, endoscopy, singular orbital integrals, ramification, transfer factors, and normalization;*
- *stable orbital integral matching on the declared scope;*
- *no unresolved trace residual after the ledger rows are discharged or separated.*

Theorem 9413.1051 (Stable trace formula as no-hidden trace defect)**.** *Within the BEDC trace object interface, the stable trace formula certificate is equivalent to the compact trace identity base together with no hidden trace defect and complete trace-tail ledger discipline on the declared scope.*

Proof. If the stable trace formula certificate holds, then every continuous, Eisenstein, endoscopic, singular, ramified, transfer-factor, and normalization residual has been recorded and discharged or separated. Thus no hidden trace defect remains.

Conversely, assume the compact trace identity base, no hidden trace defect, and complete trace-tail ledger discipline on the declared scope. Any nonzero trace residual would be finite visible or tail visible by Definition 9413.1045. The finite case is excluded by no hidden trace defect, and the tail case is excluded by the complete ledger and closedness. Hence the residual vanishes on the declared stable scope, which is the stable trace formula certificate. □

Theorem 9413.1052 (Trace closure implies Langlands existence)**.** *Assume a parameter state certificate, finite Hecke projectors, the stable trace formula certificate, cofinal automorphic tail closure, no-hidden Langlands residual, and closedness against unresolved ledger rows. Then the parameter packet $\phi^{\uparrow}$ has a global automorphic realization on the certified unramified finite detector scope.*

Proof. Fix a finite Hecke window $\mathcal{W}$. The parameter state certificate gives a projector $f_{\phi,\mathcal{W}}$ with nonzero parameter-side readout. The stable trace formula certificate transports this nonzero readout to the spectral side. The finite Hecke projector then forces a matching automorphic packet in the fixed window. Applying this to a cofinal tower gives a cofinal realization sequence. By the automorphic tail closure theorem, the sequence either globalizes to an automorphic packet or enters the automorphic tail ledger. Closedness excludes an unresolved tail row, so a global packet remains. No-hidden Langlands residual then excludes any remaining Hecke–Frobenius mismatch. □

Theorem 9413.1053 (BEDC–Langlands trace closure theorem)**.** *On a Langlands scope carrying unramified local Satake closure, finite Hecke projectors, compact trace identity base, no hidden trace defect, complete trace-tail ledger discipline, parameter state certificates, cofinal automorphic tail closure, no-hidden Langlands residual, and closedness, Langlands reciprocity holds on the certified unramified finite detector scope.*

Proof. The compact trace identity base and no-hidden trace defect give the stable trace formula certificate by Definition 9413.1051. Unramified local Satake closure licenses the finite detectors, and the finite Hecke projector supplies fixed-window isolation. Parameter state certificates provide nonzero parameter-side readouts. Therefore Definition 9413.1052 gives a global automorphic packet. No-hidden Langlands residual removes all remaining mismatch, giving

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

for every certified unramified finite detector (v, r). □

Theorem 9413.1054 (Stable trace certificate boundary)**.** *The stable trace formula certificate described here is a BEDC closure form of the trace formula. It identifies every possible failure as finite visible or ledgered tail visible, and it proves the compact kernel trace identity base. It does not replace the analytic work needed to discharge continuous spectrum, Eisenstein series, endoscopy, singular orbital integrals, transfer factors, ramification, or normalization in a specific noncompact case.*

Proof. The compact quotient proof uses only the ordinary trace-class kernel identity. All noncompact and unstable terms are explicitly named as ledger rows. The certificate is closed only when those rows are discharged or separated on the declared scope; otherwise they remain obligations rather than hidden equalities. □

9413.83 BEDC–Langlands Trace-Window Closure

The trace-formula bridge of section 9413.71 can be read as a fixed-window contradiction. If an allowed parameter has no automorphic realization, then every trace-window projector should have zero spectral target contribution. A stable parameter distribution gives a nonzero finite readout, and the stable trace formula transports that readout to the spectral side. The absence of an automorphic packet must therefore become a finite trace-window residual or an automorphic tail ledger. The fixed-window projector used below is supplied by section 9413.73.

Definition 9413.1055 (Finite Hecke window)**.** *A finite Hecke window is a tuple*

$$\mathcal{W} = (S, R, \mathcal{B})$$

where S is a finite set of places containing the displayed bad places, $R = \{r_1, \dots, r_m\}$ is a finite list of finite-dimensional representations of ${}^L G$, and $\mathcal{B}$ is a finite complexity bound for conductor, level, central character, archimedean type, and other window data. The admitted automorphic rows in the window form the truncated packet

$$\mathsf{Aut}_{G,\mathcal{W}}(F).$$

Definition 9413.1056 (Window readout and window residual)**.** *For $\pi \in \mathsf{Aut}_{G,\mathcal{W}}(F)$ define*

$$\lambda_{\mathcal{W}}(\pi) := \big(\lambda_{v,r}(\pi)\big)_{v\in S,\ r\in R}.$$

For an allowed parameter packet $\phi^{\uparrow}$ define

$$a_{\mathcal{W}}(\phi) := \big(\operatorname{tr} r(\phi(\mathrm{Frob}_v))\big)_{v\in S,\ r\in R}.$$

The window residual is

$$D_{\mathcal{W}}(\pi, \phi) := \lambda_{\mathcal{W}}(\pi) - a_{\mathcal{W}}(\phi).$$

It is a finite mismatch address inside the fixed window $\mathcal{W}$.

Theorem 9413.1057 (Finite Hecke window separation)**.** *Assume that the truncated packet*

$$\mathsf{Aut}_{G,\mathcal{W}}(F) = \{\pi_1, \dots, \pi_M\}$$

has pairwise distinct window readouts. Then there is a Hecke test function $f_{\phi,\mathcal{W}}$ such that

$$\operatorname{tr} \pi_i(f_{\phi,\mathcal{W}}) = 0$$

whenever $\lambda_{\mathcal{W}}(\pi_i) \neq a_{\mathcal{W}}(\phi)$, and

$$\operatorname{tr} \pi_i(f_{\phi,\mathcal{W}}) \neq 0$$

whenever $\lambda_{\mathcal{W}}(\pi_i) = a_{\mathcal{W}}(\phi)$ and the matching summand is not cancelled by a declared ledger row.

Proof. This is the finite-window form of Definition 9413.910. The joint Hecke operators of the finite window act on a finite truncated packet with finitely many joint eigenvalue rows

$$\lambda_{\mathcal{W}}(\pi_i).$$

On this finite set, polynomial interpolation gives a function $P_{\phi,\mathcal{W}}$ that is zero on all eigenvalue rows distinct from $a_{\mathcal{W}}(\phi)$ and nonzero on the matching row. The Satake identification reads this polynomial as a Hecke test function in the window algebra. Applying it to the joint eigenpacket gives the stated trace values, with any declared cancellation placed in the test-function ledger. □

Definition 9413.1058 (Parameter stable distribution certificate)**.** *A parameter stable distribution certificate for $\phi^{\uparrow}$ assigns to each finite Hecke window $\mathcal{W}$ a stable functional*

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}} : \mathcal{H}_{\mathcal{W}} \to \mathbb{C}$$

such that its value on unramified Hecke readout functions depends only on $a_{\mathcal{W}}(\phi)$, and all ramified, endoscopic, continuous, boundary, normalization, and transfer-factor terms have ledger rows. It is nonzero on a window when there is a test function $f_{\phi,\mathcal{W}}$ with

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0.$$

Definition 9413.1059 (Stable trace-window certificate)**.** *A stable trace-window certificate for a finite Hecke window records*

$$I^{\mathrm{st}}_{\mathrm{geom}}(f) = I^{\mathrm{st}}_{\mathrm{spec}}(f)$$

for all admitted window test functions f, together with the spectral decomposition

$$I^{\mathrm{st}}_{\mathrm{spec}}(f) = \sum_{\Pi} m(\Pi)\Theta_{\Pi}(f) + I_{\mathrm{cont}}(f) + I_{\mathrm{endo}}(f) + I_{\mathrm{bdry}}(f),$$

where every continuous, endoscopic, boundary, ramified, multiplicity, and normalization term is either separated from the target packet or recorded in the trace ledger.

Theorem 9413.1060 (Stable trace detects the target packet)**.** *Let $\phi^{\uparrow}$ have a parameter stable distribution certificate. Assume there is a finite Hecke window $\mathcal{W}$ and a projector $f_{\phi,\mathcal{W}}$ such that*

$$\Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0.$$

If the stable trace-window certificate holds and all non-target terms are separated or ledgered, then some automorphic packet Π satisfies

$$\Theta_{\Pi}(f_{\phi,\mathcal{W}}) \neq 0.$$

Proof. The parameter stable distribution supplies the nonzero geometric or parameter-side readout

$$I^{\mathrm{st}}_{\mathrm{geom}}(f_{\phi,\mathcal{W}}) = \Theta^{\mathrm{st}}_{\phi,\mathcal{W}}(f_{\phi,\mathcal{W}}) \neq 0.$$

The stable trace-window certificate gives the same nonzero value on the spectral side. In the spectral decomposition, all non-target residual terms have been separated or ledgered. Hence the remaining discrete automorphic packet sum is nonzero, so some packet Π has nonzero stable character on $f_{\phi,\mathcal{W}}$. □

Corollary 9413.1061 (Fixed-window automorphic realization)**.** *Under the hypotheses of Definition 9413.1060, the parameter $\phi^{\uparrow}$ has an automorphic realization on the fixed window $\mathcal{W}$.*

Proof. By the theorem, some packet Π has

$$\Theta_{\Pi}(f_{\phi,\mathcal{W}}) \neq 0.$$

The test function was constructed as a window projector in Definition 9413.1057. Therefore its nonzero value can occur only on packets whose window readout equals $a_{\mathcal{W}}(\phi)$, outside ledgered cancellation rows. □

Definition 9413.1062 (Cofinal Hecke window tower)**.** *A cofinal Hecke window tower is an increasing sequence*

$$\mathcal{W}_1 \subset \mathcal{W}_2 \subset \cdots$$

whose union contains every certified unramified finite Hecke–Frobenius detector. A tower realization of $\phi^{\uparrow}$ is a sequence of automorphic packets Π_n with

$$\lambda_{\mathcal{W}_n}(\Pi_n) = a_{\mathcal{W}_n}(\phi).$$

Definition 9413.1063 (Automorphic compactness ledger)**.** *The automorphic compactness ledger records every way a cofinal tower realization fails to stabilize to a global automorphic packet:*

$$\mathsf{LevelEscape},\quad \mathsf{ConductorEscape},\quad \mathsf{ArchimedeanEscape},\quad \mathsf{ContinuousSpectrumEscape},$$

$$\mathsf{EndoscopicSwitching},\quad \mathsf{BoundaryEscape},\quad \mathsf{MultiplicityEscape}.$$

A cofinal automorphic compactness certificate says that every tower realization either has a cofinally stable global packet Π_ϕ or enters this ledger.

Theorem 9413.1064 (Cofinal trace realization)**.** *Assume every window of a cofinal Hecke window tower has a fixed-window automorphic realization of $\phi^\uparrow$. If a cofinal automorphic compactness certificate holds and closed accepted objects carry no unresolved automorphic compactness ledger, then $\phi^\uparrow$ has a global automorphic packet Π_ϕ matching all certified unramified finite detectors.*

Proof. Choose a matching packet Π_n for each window $\mathcal{W}_n$. The cofinal compactness certificate says that the sequence either stabilizes cofinally to a global packet Π_ϕ or enters the automorphic compactness ledger. The ledger alternative is incompatible with a closed accepted object if the tail remains unresolved. Thus the closed case has a global packet. Since the windows are cofinal among the certified unramified finite detectors, Π_ϕ matches ϕ on all of them. □

Theorem 9413.1065 (No missing automorphic realization)**.** *Let $\phi^\uparrow$ be an allowed parameter packet. Suppose it carries a parameter stable distribution certificate, finite Hecke window projector certificates, stable trace-window certificates on a cofinal tower, a cofinal automorphic compactness certificate, no-hidden Langlands residual, and closedness against unresolved residual ledgers. Then $\phi^\uparrow$ has an automorphic realization.*

Proof. Assume that no automorphic realization exists. The parameter stable distribution certificate gives a fixed window $\mathcal{W}$ and a test function $f_{\phi,\mathcal{W}}$ with nonzero parameter-side stable readout. By Definition 9413.1060, the spectral side has a nonzero automorphic packet contribution unless all non-target spectral terms absorb the value. Such absorption is exactly a continuous-spectrum, endoscopic, ramified, boundary, multiplicity, or normalization ledger row. If that row is omitted, ledger discipline is violated; if it is carried unresolved, closedness fails.

If a target automorphic packet contribution exists, then Definition 9413.1061 gives a fixed-window realization. Applying this argument along the cofinal tower and then using Definition 9413.1064 gives a global automorphic realization, contradicting the assumption. Hence the assumption is impossible for a closed packet. □

Theorem 9413.1066 (Functorial existence by trace-window closure)**.** *Let (π_H, ϕ_H) be a closed packet for H, and let*

$$\eta : {}^L H \to {}^L G$$

preserve the parameter triple rows. If the target parameter

$$\phi_G := \eta \circ \phi_H$$

satisfies the hypotheses of Definition 9413.1065 on the G-side, then there is a G-side automorphic packet Π_G with

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v)))$$

on the certified unramified finite detectors.

Proof. Apply Definition 9413.1065 to ϕ_G. This gives a G-side automorphic packet matching ϕ_G on all certified unramified finite detectors. Substituting $\phi_G = \eta \circ \phi_H$ yields the displayed equality. Any remaining mismatch is handled by the no-hidden residual row exactly as in Definition 9413.883. □

Theorem 9413.1067 (Trace-window closure boundary)**.** *The trace-window closure argument proves existence only relative to parameter stable distributions, fixed-window projectors, stable trace-window formulae, and cofinal automorphic compactness. It does not construct those certificates.*

Proof. Every positive step above consumes one of those certificates: window separation consumes finite Hecke semisimplicity and Satake readout; nonzero spectral realization consumes parameter stable distributions and stable trace-window equality; global realization consumes cofinal automorphic compactness. The argument shows that, once those rows are present, a missing automorphic realization cannot be hidden in a closed packet. It does not replace the traditional construction of the rows. □

9413.84 BEDC–Langlands Window-to-Global Bridge

The trace-window closure of section 9413.83 uses a compactness row. This section separates that row into a finite complexity certificate and an explicit automorphic tail ledger. The point is to prevent a fixed-window realization theorem from being read as a growing-window theorem without an intervening stability witness.

Definition 9413.1068 (Automorphic complexity)**.** *An automorphic packet* Π *has automorphic complexity*

$$\kappa(\Pi) := \kappa_{\mathrm{level}}(\Pi) + \kappa_{\mathrm{cond}}(\Pi) + \kappa_{\infty}(\Pi) + \kappa_{\mathrm{ram}}(\Pi) + \kappa_{\mathrm{endo}}(\Pi) + \kappa_{\mathrm{mult}}(\Pi).$$

The six summands record, respectively, compact-open or level complexity, conductor complexity, archimedean infinitesimal-character or weight complexity, ramified local type complexity, endoscopic or Arthur packet label complexity, and multiplicity or packet bookkeeping complexity. The expression is a ledgered height: changing the automorphic category may change its implementation, but not the obligation that every escape mode represented by a hidden name be addressable.

Definition 9413.1069 (Bounded automorphic finiteness certificate)**.** *For* $B < \infty$ *set*

$$\mathsf{Aut}_{\leq B}(G) := \{\Pi : \kappa(\Pi) \leq B\}.$$

A bounded automorphic finiteness certificate says that $\mathsf{Aut}_{\leq B}(G)$ *is finite up to the declared closed automorphic equivalence relation for every finite* B*.*

Definition 9413.1070 (Cofinal window realization)**.** *Let*

$$\mathcal{W}_1 \subset \mathcal{W}_2 \subset \cdots$$

be a cofinal Hecke window tower in the sense of Definition 9413.1062. A cofinal window realization of an allowed parameter packet $\phi^{\uparrow}$ *is a sequence of automorphic packets* (Π_n) *such that*

$$\lambda_{\mathcal{W}_n}(\Pi_n) = a_{\mathcal{W}_n}(\phi)$$

for every n*. Equivalently, for every detector* (v, r) *lying in* $\mathcal{W}_n$*,*

$$\lambda_{v,r}(\Pi_n) = \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Definition 9413.1071 (Automorphic tail ledger)**.** *The automorphic tail ledger is the refinement of Definition 9413.1063 that names the concrete ways a cofinal window realization can fail to stabilize:*

$$\mathsf{LevelConductorEscape},\quad \mathsf{ArchimedeanParameterEscape},\quad \mathsf{RamifiedLocalTypeEscape},$$

$$\mathsf{ContinuousSpectrumEscape},\quad \mathsf{EndoscopicSwitching},\quad \mathsf{BoundaryEisensteinResidualEscape}.$$

A complete automorphic tail ledger says that every unbounded realization sequence has at least one of these rows, or a declared refinement of one of them, as its obstruction to global stabilization.

Theorem 9413.1072 (Bounded subsequence gives global realization)**.** *Let (Π_n) be a cofinal window realization of $\phi^{\uparrow}$. Assume the bounded automorphic finiteness certificate. If there are $B < \infty$ and an infinite subsequence $n_j \to \infty$ such that*

$$\kappa(\Pi_{n_j}) \leq B$$

for all j, then there is an automorphic packet Π_ϕ satisfying

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

for every certified unramified finite Hecke–Frobenius detector (v, r).

Proof. By bounded automorphic finiteness, the packets of complexity at most B represent only finitely many closed automorphic equivalence classes. Hence one class occurs for infinitely many terms of the subsequence. Choose a representative Π_ϕ of that recurring class.

Let (v, r) be any certified unramified finite detector. Cofinality of the tower gives $N(v, r)$ such that $(v, r) \in \mathcal{W}_n$ for all $n \geq N(v, r)$. Since the recurring class appears along infinitely many n_j, choose one with $n_j \geq N(v, r)$. The packet Π_{n_j} realizes ϕ on $\mathcal{W}_{n_j}$, so

$$\lambda_{v,r}(\Pi_{n_j}) = \operatorname{tr} r(\phi(\mathrm{Frob}_v)).$$

Closed automorphic equivalence preserves the declared Hecke readouts, so the same equality holds for Π_ϕ. Since (v, r) was arbitrary, Π_ϕ matches the parameter on the full cofinal detector scope. □

Theorem 9413.1073 (Unbounded realization sequence enters the tail ledger)**.** *Let (Π_n) be a cofinal window realization of $\phi^{\uparrow}$. If no infinite subsequence has bounded automorphic complexity, then the realization sequence enters the automorphic tail ledger.*

Proof. The absence of a bounded-complexity infinite subsequence means that for every finite B, only finitely many n satisfy

$$\kappa(\Pi_n) \leq B.$$

Thus the realization sequence escapes every finite complexity sublevel. By the definition of κ, this escape must occur through level or conductor growth, archimedean parameter growth, ramified local type growth, endoscopic or Arthur packet switching, multiplicity bookkeeping growth, or movement of mass into continuous, boundary, Eisenstein, or residual spectrum. These are exactly the rows named by Definition 9413.1071. Completeness of the ledger records the observed escape as an automorphic tail row. □

Theorem 9413.1074 (Window-to-global alternative)**.** *Assume that every window in a cofinal Hecke window tower has an automorphic realization of $\phi^{\uparrow}$, that bounded automorphic finiteness holds, and that the automorphic tail ledger is complete. Then either $\phi^{\uparrow}$ has a global automorphic realization on the certified unramified finite detector scope, or a cofinal realization sequence enters the automorphic tail ledger.*

Proof. Choose one realizing packet Π_n for each window $\mathcal{W}_n$. If some infinite subsequence of (Π_n) has bounded automorphic complexity, then Definition 9413.1072 produces a global automorphic packet Π_ϕ matching all certified unramified finite detectors. If no such subsequence exists, then Definition 9413.1073 places the sequence in the automorphic tail ledger. These two cases exhaust the sequence. □

Theorem 9413.1075 (Closed window-to-global realization)**.** *Assume the hypotheses of Definition 9413.1074. If closed accepted Langlands packets cannot carry unresolved automorphic tail ledger rows, then $\phi^\uparrow$ has a global automorphic realization on the certified unramified finite detector scope.*

Proof. By the window-to-global alternative, either the global realization is already present or the chosen cofinal realization sequence enters the automorphic tail ledger. In the second case, the tail row is either unrecorded, which violates ledger discipline, or recorded but unresolved, which violates closedness. Hence the tail alternative is not available for a closed accepted packet. The global realization alternative remains. □

Corollary 9413.1076 (Trace-window existence gives global realization)**.** *Let $\phi^\uparrow$ have parameter stable distribution certificates, finite-window projector certificates, and stable trace-window certificates on a cofinal Hecke window tower. Assume bounded automorphic finiteness, a complete automorphic tail ledger, and closedness against unresolved automorphic tail rows. Then $\phi^\uparrow$ has a global automorphic realization.*

Proof. The fixed-window certificates give an automorphic realization on every window by Definition 9413.1061. The closed window-to-global realization theorem applies to the resulting cofinal realization sequence. □

Corollary 9413.1077 (Global reciprocity after no-hidden residuals)**.** *Under the hypotheses of Definition 9413.1076, assume also the no-hidden Langlands residual condition of Definition 9413.859. Then the global automorphic packet Π_ϕ satisfies*

$$\lambda_{v,r}(\Pi_\phi) = \operatorname{tr} r(\phi(\mathrm{Frob}_v))$$

on the certified unramified finite detector scope, and every residual mismatch is either detected or rejected by closedness.

Proof. The preceding corollary supplies the global automorphic packet. If a Hecke–Frobenius mismatch remained on the certified scope, the no-hidden residual condition would make it a finite detector residual or a ramified, endoscopic, continuous-spectrum, boundary, or normalization ledger residual. A finite detector residual contradicts the cofinal matching; an unresolved ledger residual contradicts closedness. Hence no residual mismatch remains in a closed accepted packet. □

Theorem 9413.1078 (Functorial window-to-global transfer)**.** *Let (Π_H, ϕ_H) be a closed H-side packet, and let*

$$\eta : {}^L H \to {}^L G$$

preserve the declared Langlands packet rows. Put

$$\phi_G := \eta \circ \phi_H.$$

If ϕ_G satisfies the hypotheses of Definition 9413.1077 on the G-side, then there is a G-side automorphic packet Π_G with

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\phi_H(\mathrm{Frob}_v)))$$

on the certified unramified finite detector scope.

Proof. Apply Definition 9413.1077 to ϕ_G. The resulting packet Π_G matches the Frobenius readout of ϕ_G on the certified detector scope. Substituting $\phi_G = \eta \circ \phi_H$ gives the displayed equality. Any remaining failure would be a no-hidden residual or an unresolved automorphic tail row, both incompatible with the closed hypotheses. □

Theorem 9413.1079 (Categorical analogue)**.** *In a geometric Langlands setting, the same bridge has the following categorical form. If a certified Langlands functor sends every finite Hecke modification window of a spectral object into the automorphic category, and if the functor is essentially surjective modulo a complete categorical boundary ledger, then the spectral object has an automorphic realization unless it enters that boundary ledger.*

Proof. Finite Hecke modification windows play the role of the tower $\mathcal{W}_n$. Essential surjectivity replaces bounded automorphic finiteness, and the categorical boundary ledger replaces the automorphic tail ledger. The same dichotomy applies: a cofinally stable image gives an automorphic realization, while failure of stability is a declared boundary row. Closedness excludes unresolved boundary rows. □

Theorem 9413.1080 (Window-to-global boundary)**.** *The window-to-global bridge proves only a conditional passage from finite Hecke windows to a global automorphic packet. It does not prove bounded automorphic finiteness, stable trace-window formulae, parameter stable distributions, or geometric essential surjectivity.*

Proof. Each step consumes one displayed certificate. Fixed-window existence comes from the trace-window machinery; repetition inside a finite complexity sublevel comes from bounded automorphic finiteness; exclusion of unbounded escape comes from the automorphic tail ledger and closedness. The bridge therefore identifies the precise obstruction to globalization: missing finite-window realization, missing bounded finiteness, or an unresolved automorphic tail row. □

9413.85 BEDC–Langlands No-Hidden Residual Master Lemma

The master no-hidden residual condition used by the Langlands closure theorem is not an independent oracle. It is the finite coproduct of the detector-completeness certificates already isolated for the unramified, ramified, component-group, endoscopic, continuous, and cofinal-tail faces of the Langlands object.

Definition 9413.1081 (Total Langlands residual decomposition)**.** *For a candidate Langlands row*

$$\mathcal{O}^{\uparrow} = (\pi, \psi),$$

the total residual is the formal sum

$$R_{\mathrm{Lan}}(\pi, \psi) = R_{\mathrm{unr}} \oplus R_{\mathrm{ram}} \oplus R_{\mathrm{mult}} \oplus R_{\mathrm{endo}} \oplus R_{\mathrm{cont}} \oplus R_{\mathrm{tail}}.$$

The summands denote, respectively, unramified Hecke–Frobenius mismatch, ramified local packet or character mismatch, packet multiplicity or component-label mismatch, endoscopic transfer mismatch, continuous/Eisenstein/boundary residual, and cofinal-window or representation-complexity tail.

Definition 9413.1082 (Completeness certificates for no-hidden residual)**.** *A no-hidden residual certificate for $\mathcal{O}^{\uparrow} = (\pi, \psi)$ consists of the following data on the declared Langlands scope:*

- *unramified Satake and Tannakian separation for finite Hecke–Frobenius detectors;*
- *ramified local character separation and complete local parameter ledger discipline;*
- *component-group Fourier completeness for packet labels and multiplicities;*
- *endoscopic transfer finite-window or tail-ledger dichotomy;*
- *continuous-spectrum and boundary ledger completeness;*
- *cofinal automorphic tail closure;*
- *a ledger policy requiring every finite-visible or tail-visible residual to be recorded, discharged, or declared outside the claim.*

Theorem 9413.1083 (Unramified residual is finite-detected)**.** *If*

$$R_{\mathrm{unr}} \neq 0,$$

then there is an unramified place v and a finite-dimensional representation

$$r : {}^L G \to GL(V_r)$$

such that

$$\lambda_{v,r}(\pi) \neq \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

Proof. The unramified residual says that the Satake semisimple class attached to π at some unramified place differs from the Frobenius semisimple class attached to ψ. If every finite-dimensional representation r had equal trace on these two classes, then Definition 9413.860 would identify the two semisimple conjugacy classes. This contradicts the asserted residual. Hence some finite detector (v, r) has nonzero Hecke–Frobenius residual. □

Theorem 9413.1084 (Ramified residual is finite-detected or local-ledgered)**.** *If*

$$R_{\mathrm{ram}} \neq 0,$$

then some ramified finite local detector

$$(v, \mathfrak{b}, f_v)$$

has

$$\Theta_{\pi_v}(f_v) \neq S\Theta_{\psi_v}(f_v),$$

or the residual is a local parameter ledger row.

Proof. Inside a fixed Bernstein or packet window, a ramified mismatch is the nonzero distribution

$$\Theta_{\pi_v} - S\Theta_{\psi_v}.$$

A nonzero linear functional on the local test-function space is nonzero on some f_v, and Definition 9413.935 supplies the finite-window separation used to isolate the mismatch. If no fixed local window detects it, then Definition 9413.938 places the remaining alternatives in the local ledger: monodromy, normalization, wild ramification, transfer factors, depth, or local packet labels. □

Theorem 9413.1085 (Component residual is Fourier-detected)**.** *If $R_{\mathrm{mult}} \neq 0$, then a component-group Fourier detector detects it, or the obstruction is a packet ledger row.*

Proof. A multiplicity or packet-label residual is a difference between two finite coefficient functions on the character group

$$\widehat{\mathcal{S}_\psi}.$$

When both functions are defined and differ, Definition 9413.997 provides an element

$$s \in \mathcal{S}_\psi$$

whose Fourier readout distinguishes them. Equivalently, the Fourier-twisted packet distribution changes at s. If the packet labels, coefficients, or comparison scope needed for this finite Fourier transform are absent, then Definition 9413.998 classifies the failure as a packet ledger row. □

Theorem 9413.1086 (Endoscopic residual is fixed-window visible or tail-ledgered)**.** *If*

$$R_{\mathrm{endo}} \neq 0,$$

then some matching endoscopic test pair

$$f^H \leftrightarrow f^G$$

has nonzero transfer residual, or the residual belongs to the endoscopic tail ledger.

Proof. This is exactly the finite/tail dichotomy of Definition 9413.1004. If a fixed matching pair witnesses the mismatch, the residual is finite-visible. If all fixed matching windows agree while a transfer failure remains, the failure has left the finite transfer window and is therefore a transfer-factor, ramification, singular-orbital, normalization, boundary, or continuous-spectrum tail row. □

Theorem 9413.1087 (Continuous and boundary residuals are ledger rows)**.** *If $R_{\mathrm{cont}} \neq 0$, then the residual belongs to the continuous-spectrum, Eisenstein, residual-spectrum, truncation, intertwining, scattering, or boundary ledger.*

Proof. Continuous and boundary terms are not isolated discrete packet characters. They arise from parabolic induction, Eisenstein families, constant terms, truncation boundaries, scattering matrices, poles of intertwining operators, or limiting trace-formula rows. If one of these terms is already visible on a fixed trace window, then it is visible as a trace residual and is recorded by the trace ledger. If it appears only through a limiting window, it is a trace tail defect. The dichotomy is the one recorded in Definition 9413.1045, and the ledger requirement is the trace-tail certificate of Definition 9413.1046. □

Theorem 9413.1088 (Cofinal tail residual dichotomy)**.** *If every finite Hecke window has a realization but the cofinal tower does not stabilize to a global packet, then the residual belongs to the automorphic compactness ledger.*

Proof. Choose a realization on each window of a cofinal Hecke tower. The cofinal automorphic compactness certificate used in Definition 9413.1064 says that such a tower either has a cofinally stable global packet or enters the automorphic compactness ledger of Definition 9413.1063. The failure of stabilization is exactly the second alternative. □

Theorem 9413.1089 (NoHiddenLanglandsResidual master lemma)**.** *Assume a candidate Langlands row*

$$\mathcal{O}^{\uparrow} = (\pi, \psi)$$

carries the completeness certificates of Definition 9413.1082. If

$$R_{\mathrm{Lan}}(\pi, \psi) \neq 0,$$

then some residual summand is visible to a finite Langlands detector, or the residual is recorded in the ramified, packet, endoscopic, continuous, boundary, representation-tail, or automorphic-tail ledger.

Proof. By Definition 9413.1081, a nonzero total residual has a nonzero summand.

If $R_{\mathrm{unr}} \neq 0$, then Definition 9413.1083 gives a finite Hecke–Frobenius detector. If $R_{\mathrm{ram}} \neq 0$, then Definition 9413.1084 gives a ramified local detector or a local ledger row. If $R_{\mathrm{mult}} \neq 0$, then Definition 9413.1085 gives a component-group Fourier detector or a packet ledger row. If $R_{\mathrm{endo}} \neq 0$, then Definition 9413.1086 gives a fixed matching test pair or an endoscopic tail row. If $R_{\mathrm{cont}} \neq 0$, then Definition 9413.1087 places the residual in the continuous or boundary ledger. If $R_{\mathrm{tail}} \neq 0$, then Definition 9413.1088 places the residual in the automorphic compactness or representation tail ledger.

These cases exhaust the displayed decomposition. Therefore no nonzero Langlands residual is unaddressed: it is finite-visible or ledgered. □

Corollary 9413.1090 (Closed Langlands rows have zero total residual)**.** *Let $\mathcal{O}^{\uparrow} = (\pi, \psi)$ be a closed BEDC–Langlands row with the three-properties ledger and the no-hidden residual certificates above. Then*

$$R_{\mathrm{Lan}}(\pi, \psi) = 0.$$

Proof. If the total residual were nonzero, the master lemma would make it finite-visible or ledgered. A finite-visible residual must be recorded by ledger discipline; carrying it unresolved contradicts closedness, and omitting it violates the ledger property. A ledger residual is excluded by the same closedness requirement. Hence no total residual remains. □

Corollary 9413.1091 (Reciprocity from existence and no-hidden residual)**.** *Assume $\Pi_{\psi}(G) \neq \varnothing$ and let $\pi \in \Pi_{\psi}(G)$ be a closed packet member satisfying the certificates above. Then for every certified unramified finite detector (v, r),*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

Proof. The displayed equality is the vanishing of the unramified summand R_{unr} at the detector (v, r). By Definition 9413.1090, the whole total residual vanishes, so the unramified component vanishes on the certified detector scope. □

Theorem 9413.1092 (Boundary of the master lemma)**.** *The master lemma is conditional on the named completeness certificates. It does not construct local Langlands packets, stable trace formulae, endoscopic transfer, component-group labels, automorphic compactness, or continuous-spectrum analysis. It proves that, once those certificates and the ledger policy are present, the phrase no-hidden Langlands residual has a finite decomposition proof.*

Proof. Each branch of the proof invokes one named completeness input: Tannakian separation for unramified readouts, local character separation for ramified readouts, finite Fourier inversion for component data, endoscopic finite/tail separation, trace-tail ledger discipline for continuous and boundary terms, and cofinal compactness for automorphic escape. The proof is therefore a composition theorem, not a construction of those inputs. □

9413.86 BEDC–Langlands Exact Globalization Contract

The Langlands closure chain can be compressed into a certificate DAG statement. Local packet rows, finite Hecke windows, parameter states, trace identities, multiplicity formulae, endoscopic transfers, Tannakian separation, Chebotarev uniqueness, cofinal tail closure, and no-hidden residual discipline are separate certificates. Exact globalization says that once they are connected by complete NameCert and ledger-pullback data, their composite has no unaddressed residual.

Definition 9413.1093 (Langlands certificate DAG)**.** *A Langlands certificate DAG is a finite acyclic graph*

$$\mathcal{D}_{\mathrm{Lan}}$$

whose nodes are drawn from the certificate interfaces

$$\mathsf{LocalPacketCert},\quad \mathsf{ParameterStateCert},\quad \mathsf{FiniteHeckeProjectorCert},$$

$$\mathsf{StabilizedTraceFormulaCert},\quad \mathsf{StableMultiplicityFormulaCert},\quad \mathsf{EndoscopicTransferCert},$$

$$\mathsf{TannakianSeparationCert},\quad \mathsf{ChebotarevCert},\quad \mathsf{CofinalAutomorphicTailClosure},\quad \mathsf{NoHiddenLanglandsResidual}.$$

Each node C carries a NameCert row

$$\mathrm{NameCert}(C) = (\mathrm{SourceSpec}_C, \mathrm{PatternSpec}_C, \mathrm{ClassifierSpec}_C, \mathrm{StabilityCert}_C, \mathrm{LedgerPolicy}_C).$$

The sink node is the closed global Langlands row

$$\mathsf{GlobalLanglandsObject}(\Pi_\psi, \psi).$$

Definition 9413.1094 (Ledger pullback edge)**.** *An edge*

$$C \longrightarrow C'$$

of $\mathcal{D}_{\mathrm{Lan}}$ consists of a readout map from the source interface to the target interface and a pullback ledger

$$\Lambda_{C\to C'}$$

with the following property. Any residual created by composing the two interfaces either projects to a residual already visible in $\mathrm{LedgerPolicy}_C$*, projects to a residual already visible in* $\mathrm{LedgerPolicy}_{C'}$*, or is recorded as an edge-compatibility row in $\Lambda_{C\to C'}$.*

Definition 9413.1095 (Node exactness)**.** *A node C of the certificate DAG is exact when every residual in the readout scope of C is finite-visible through a detector named by C, or is recorded by* $\mathrm{LedgerPolicy}_C$*. A closed exact node carries no unresolved ledger row.*

Definition 9413.1096 (Finite-window exactness)**.** *For a finite Hecke window*

$$\mathcal{W} = (S, R, \mathcal{B})$$

as in Definition 9413.1055, a packet Π is exact over $\mathcal{W}$ for ψ when

$$\lambda_{\mathcal{W}}(\Pi) = a_{\mathcal{W}}(\psi)$$

and every residual not measured inside $\mathcal{W}$ is recorded in the window-tail ledger attached to the bound $\mathcal{B}$.

Theorem 9413.1097 (Ledger pullback preserves exactness)**.** *Let $C \to C'$ be a ledger pullback edge. If C and C' are exact, then their composite interface is exact.*

Proof. Let ρ be a residual of the composite interface. If its projection to the source interface is nonzero, exactness of C makes it finite-visible or records it in $\mathrm{LedgerPolicy}_C$. If its projection to the target interface is nonzero, exactness of C' does the same using $\mathrm{LedgerPolicy}_{C'}$.

It remains to consider the case in which both projections vanish but the composite residual is still asserted to be nonzero. Such a residual is not a source residual and not a target residual; it is precisely a compatibility failure of the edge. By Definition 9413.1094, every edge-compatibility failure is recorded in $\Lambda_{C \to C'}$. Hence the composite is exact. □

Theorem 9413.1098 (Certificate DAG induction)**.** *Let $\mathcal{D}_{\mathrm{Lan}}$ be a Langlands certificate DAG. Suppose every source node is exact and every edge is a ledger pullback edge. Then every node obtained by composing certificates along $\mathcal{D}_{\mathrm{Lan}}$ is exact. If the sink is closed, the sink has no unresolved residual.*

Proof. Choose a topological ordering of the finite acyclic graph. Source nodes are exact by hypothesis. Assume all predecessors of a node C' are exact. Composing C' with each predecessor along its incoming ledger pullback edge preserves exactness by Definition 9413.1097. Since the incoming set is finite, repeated application gives exactness of the composite interface at C'. Induction over the topological ordering proves exactness for every node, including the sink.

At a closed sink, any finite-visible residual must be recorded by the sink ledger policy, and any ledger residual is incompatible with closedness if unresolved. Omitting it violates ledger discipline. Thus the closed sink carries no unresolved residual. □

Theorem 9413.1099 (Cofinal exactness dichotomy)**.** *Assume every window of a cofinal Hecke tower has an exact finite-window realization of ψ. If bounded automorphic finiteness and the complete automorphic tail ledger hold, then either there is a global packet Π_ψ matching every certified finite detector, or a cofinal realization sequence enters the automorphic tail ledger.*

Proof. Choose one exact realizing packet on each finite window. This gives a cofinal window realization in the sense of Definition 9413.1070. The window-to-global alternative of Definition 9413.1074 applies: bounded complexity along an infinite subsequence gives a global packet matching all certified unramified finite detectors, while absence of such a subsequence sends the realization sequence to the automorphic tail ledger. □

Corollary 9413.1100 (Closed cofinal exactness gives a global packet)**.** *Under the hypotheses of Definition 9413.1099, if closed accepted Langlands rows carry no unresolved automorphic tail ledger, then there is a global packet Π_ψ matching every certified finite detector.*

Proof. The dichotomy gives either a global packet or an automorphic tail row. The tail row is either unrecorded, contradicting ledger discipline, or recorded but unresolved, contradicting closedness. Hence the global packet alternative remains. □

Definition 9413.1101 (Exact globalization contract)**.** *The contract*

$$\mathsf{ExactGlobalizationContract}_{\mathrm{Lan}}$$

consists of:

- *a Langlands certificate DAG with NameCert data at every node;*

- *complete ledger pullback on every edge;*
- *exact finite-window realization on a cofinal Hecke tower;*
- *bounded automorphic finiteness and a complete automorphic tail ledger;*
- *the no-hidden residual certificate of Definition 9413.1082;*
- *closedness against unresolved residual ledgers.*

Theorem 9413.1102 (ExactGlobalizationContract for Langlands)**.** *Assume*

$$\mathsf{ExactGlobalizationContract}_{\mathrm{Lan}}.$$

Then there is a global packet Π_ψ *and*

$$R_{\mathrm{Lan}}(\Pi_\psi, \psi) = 0.$$

Proof. The certificate DAG induction of Definition 9413.1098 shows that the composed Langlands interface is exact: every residual is finite-visible or ledgered. Exact finite-window realization on the cofinal tower, together with Definition 9413.1100, produces a global packet Π_ψ matching every certified finite detector.

Assume that

$$R_{\mathrm{Lan}}(\Pi_\psi, \psi) \neq 0.$$

By Definition 9413.1089, the residual is finite-visible or ledgered. A finite-visible residual is already in the DAG readout scope and must be recorded by the relevant ledger policy. A ledger residual is recorded by construction. In both cases an unresolved row contradicts closedness, while omission contradicts ledger discipline. Therefore no residual remains. □

Theorem 9413.1103 (BEDC–Langlands exact closure)**.** *If the Langlands certificate DAG has NameCert data at every node, complete ledger pullbacks on every edge, exact finite-window realization on a cofinal tower, no-hidden residual completeness, and closedness, then Langlands reciprocity holds on the certified unramified finite detector scope:*

$$\lambda_{v,r}(\Pi_\psi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

Proof. The hypotheses are exactly the data of Definition 9413.1101. By Definition 9413.1102, there is a global packet and the total Langlands residual vanishes. Projecting the equality

$$R_{\mathrm{Lan}}(\Pi_\psi, \psi) = 0$$

to the unramified summand gives

$$R_{\mathrm{unr}} = 0.$$

For the detector (v, r), this is precisely

$$\lambda_{v,r}(\Pi_\psi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

□

Corollary 9413.1104 (Functorial exact closure)**.** *Let*

$$\eta : {}^L H \to {}^L G$$

be an L-group homomorphism. If a closed H-side packet (Π_H, ψ_H) *is given and the target parameter*

$$\psi_G := \eta \circ \psi_H$$

satisfies the exact globalization contract on the G-side, then there is a G-side packet Π_G *such that*

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\eta(\psi_H(\mathrm{Frob}_v))).$$

Proof. Apply Definition 9413.1103 to the target parameter ψ_G. The resulting packet satisfies

$$\lambda_{v,r}(\Pi_G) = \operatorname{tr} r(\psi_G(\mathrm{Frob}_v)).$$

Substitute $\psi_G = \eta \circ \psi_H$. □

Corollary 9413.1105 (GL_n exact closure)**.** *For $G = GL_n$, assume the local GL_n packet certificate, the Rankin–Selberg detector closure used by the converse theorem, exact finite twist windows, the twist tail ledger, and closedness. Then the GL_n Langlands reciprocity readout holds on the certified unramified scope.*

Proof. For GL_n, local packets are singleton packets, and the component and endoscopic rows of the general certificate DAG reduce to the local packet, converse-theorem, and twist-ledger rows. The Definition 9413.971 supplies the global automorphic packet whenever all finite Rankin–Selberg detector conditions hold or sends the failure to the twist tail ledger. Closedness excludes unresolved twist-tail data. The exact globalization contract then removes the remaining Hecke–Frobenius residual. □

Corollary 9413.1106 (GL_1 two-square shadow)**.** *For $G = GL_1$, $F = \mathbb{Q}$, and $K = \mathbb{Q}(i)$, the exact closure readout specializes at odd primes to*

$$p = a^2 + b^2 \quad \Longleftrightarrow \quad \chi_{-4}(\mathrm{Frob}_p) = 1.$$

Proof. The GL_1 certificate DAG is class field theory with singleton packets and scalar Hecke–Frobenius detectors. The quadratic character of $\mathbb{Q}(i)/\mathbb{Q}$ has value 1 exactly at split odd primes. Splitting in $\mathbb{Q}(i)$ is equivalent to

$$p \equiv 1 \pmod 4,$$

and Fermat's two-square theorem identifies this condition with $p = a^2 + b^2$. □

Theorem 9413.1107 (Boundary of exact globalization)**.** *The exact globalization contract is a conditional composition theorem. It does not construct the local packet certificates, stable trace formula, stable multiplicity formula, endoscopic transfer, Chebotarev input, bounded automorphic finiteness, or cofinal tail certificate. It states that, once those nodes and their ledger pullbacks are supplied as NameCert data, their composite cannot hide a residual outside the finite-detector or ledger alternatives.*

Proof. The proof of the contract uses only DAG induction, edge ledger pullback, the cofinal window-to-global dichotomy, and the no-hidden residual master lemma. Each analytic or representation-theoretic ingredient is consumed as a certificate node. Thus the theorem records the exact BEDC globalization mechanism and leaves construction of the input certificates to their own modules. □

9413.87 BEDC–Langlands Main Closure Theorem

The preceding Langlands modules isolate the local packet rows, parameter-state rows, trace and multiplicity bridges, endoscopic transfer, Tannakian readouts, and no-hidden residual discipline. This section records the combined closure theorem. No additional analytic input is introduced here; the statement is a composition of the certificates already displayed.

Definition 9413.1108 (Certified Langlands parameter object)**.** *Let F be a global field and let G/F be a connected reductive group. A certified parameter object is a row*

$$\psi^{\uparrow} := (\psi, \mathsf{T}_{\psi}, \mathsf{U}_{\psi}, \mathsf{L}_{\psi})$$

where

$$\psi : L_F \times SL_2(\mathbb{C}) \to {}^L G$$

is an Arthur-type parameter, with ordinary Langlands parameters included by a trivial SL_2 factor. Its ledger contains

$$\Lambda_{\psi} := (S_{\mathrm{ram}}, N_v, \epsilon_v, \gamma_v, \mathcal{S}_{\psi}, \mathsf{Endoscopy}, \mathsf{ContinuousSpectrum}, \mathsf{Boundary}, \mathsf{Normalization}).$$

Definition 9413.1109 (Main detector system)**.** *The finite detector system for a certified G-parameter consists of four finite detector types.*

The unramified Hecke–Frobenius detector is

$$(v, r),$$

where v is unramified and $r : {}^L G \to GL(V_r)$ is finite-dimensional. Its residual is

$$D_{v,r}(\pi, \psi) := \lambda_{v,r}(\pi) - \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

The ramified local detector is

$$(v, \mathfrak{b}, f_v),$$

where $\mathfrak{b}$ is a Bernstein or packet window and $f_v \in C_c^{\infty}(G(F_v))$. Its residual compares

$$\Theta_{\pi_v}(f_v) \qquad \text{and} \qquad S\Theta_{\psi_v}(f_v).$$

The component-group Fourier detector is an element

$$s \in \mathcal{S}_{\psi},$$

reading the Fourier-twisted packet distribution

$$\Theta_{\psi,s}(f) = \sum_{\pi \in \Pi_{\psi}(G)} \chi_{\pi}(s) \Theta_{\pi}(f).$$

The endoscopic transfer detector is a matching test pair

$$f^G \leftrightarrow f^H$$

for an endoscopic embedding $\xi : {}^L H \to {}^L G$, reading the difference between the H-stable distribution and its G-transfer.

Definition 9413.1110 (Main residual ledger)**.** *The total Langlands residual of a candidate pair* (π, ψ) *is classified into six rows:*

$$R_{\mathrm{unr}}, \quad R_{\mathrm{ram}}, \quad R_{\mathrm{mult}}, \quad R_{\mathrm{endo}}, \quad R_{\mathrm{cont}}, \quad R_{\mathrm{tail}}.$$

They denote, respectively, unramified Hecke–Frobenius mismatch, ramified local packet mismatch, component-group or multiplicity mismatch, endoscopic transfer mismatch, continuous/Eisenstein/boundary residual, and cofinal-window or representation-complexity tail.

Definition 9413.1111 (Master no-hidden residual)**.** *The master no-hidden Langlands residual condition says that every nonzero row of*

$$R_{\mathrm{Langlands}}(\pi, \psi)$$

is detected by one of the finite detectors of Definition 9413.1109, or is recorded in the corresponding ramified, endoscopic, continuous, boundary, multiplicity, representation-tail, or automorphic-tail ledger.

Theorem 9413.1112 (No hidden residual excludes wrong closed packets)**.** *Let* (π, ψ) *be a closed BEDC–Langlands object with the three-properties ledger and the master no-hidden residual condition. Then*

$$R_{\mathrm{Langlands}}(\pi, \psi) = 0.$$

Proof. Assume a nonzero residual remains. By the master no-hidden condition, it is finite-detected or ledgered. In the finite-detected case, one of the detectors

$$(v, r), \qquad (v, \mathfrak{b}, f_v), \qquad s \in \mathcal{S}_\psi, \qquad f^G \leftrightarrow f^H$$

has a nonzero readout residual. Ledger discipline then requires that residual to be recorded; carrying it unresolved contradicts closedness, and omitting it violates the ledger property. In the ledger case, the residual is ramified, endoscopic, continuous, boundary, automorphic tail, or representation tail data. The same closedness and ledger argument applies. Hence no residual remains. □

Theorem 9413.1113 (Main existence bridge)**.** *Assume a certified parameter* $\psi^\uparrow$ *carries local packet certificates, parameter state certificates, stabilized trace formula certificates, stable multiplicity formula certificates, endoscopic transfer certificates, endoscopic induction, trace-tail ledger discipline, and closedness. Then*

$$\Pi_\psi(G) \neq \varnothing.$$

Proof. The parameter state certificate gives a fixed-window test function $f_{\psi,\mathcal{W}}$ with nonzero stable parameter readout. By the stabilized trace formula certificate, the nonzero readout appears on the spectral side. The spectral contribution has three possible addresses.

If it is a primitive G-discrete contribution, the stable multiplicity formula certificate and Definition 9413.1001 force a nonempty packet. If it comes from a proper endoscopic group, endoscopic induction and Definition 9413.1005 carry the packet to the G-side unless a transfer residual occurs; such a residual is finite-detected or ledgered, hence incompatible with closedness while unresolved. If the contribution is neither primitive nor proper-endoscopic, it lies in continuous spectrum, Eisenstein boundary, singular orbital, normalization, or tail data, all of which are ledger rows and are excluded in the closed case. Therefore the G-packet is nonempty. □

Theorem 9413.1114 (BEDC–Langlands reciprocity)**.** *Assume the hypotheses of Definition 9413.1113, together with Tannakian separation, Chebotarev uniqueness, the master no-hidden residual condition, and closedness. Then every certified closed parameter $\psi^{\uparrow}$ has $\pi \in \Pi_{\psi}(G)$ such that, for almost all certified unramified places v and every finite-dimensional $r : {}^{L}G \to GL(V_r)$ in the readout scope,*

$$\lambda_{v,r}(\pi) = \operatorname{tr} r(\psi(\mathrm{Frob}_v)).$$

Proof. By Definition 9413.1113, the packet is nonempty. Choose $\pi \in \Pi_{\psi}(G)$. If any finite readout mismatch remains, it is part of $R_{\mathrm{Langlands}}(\pi, \psi)$ and is excluded by Definition 9413.1112. Thus all unramified finite detectors agree. Tannakian separation and Chebotarev uniqueness then identify the semisimplified parameter readout on the certified scope. Any remaining ramified, endoscopic, continuous, boundary, or tail discrepancy is a ledger row and is likewise excluded by closedness. □

Theorem 9413.1115 (BEDC–Langlands functoriality)**.** *Let*

$$\eta : {}^{L}H \to {}^{L}G$$

be an L-group homomorphism and let (π_H, ψ_H) be a closed H-side packet. Put

$$\psi_G := \eta \circ \psi_H.$$

If ψ_G satisfies the hypotheses of Definition 9413.1114, then there is a G-side packet $\Pi_{\psi_G}(G)$ with a member π_G such that

$$\lambda_{v,r}(\pi_G) = \operatorname{tr} r(\eta(\psi_H(\mathrm{Frob}_v)))$$

on the certified unramified readout scope.

Proof. Apply Definition 9413.1114 to the target parameter $\psi_G = \eta \circ \psi_H$. The resulting packet has readouts

$$\lambda_{v,r}(\pi_G) = \operatorname{tr} r(\psi_G(\mathrm{Frob}_v)).$$

Substituting $\psi_G = \eta \circ \psi_H$ gives the formula. □

Theorem 9413.1116 (BEDC–Langlands main closure theorem)**.** *On any reductive-group scope carrying local packet certificates, parameter state certificates, stabilized trace formula certificates, stable multiplicity formula certificates, endoscopic transfer certificates, endoscopic induction, Tannakian separation, Chebotarev uniqueness, the master no-hidden Langlands residual condition, complete ledger discipline, and closedness, Langlands reciprocity and functoriality hold on the certified scope.*

Proof. Reciprocity is exactly Definition 9413.1114. Functoriality is exactly Definition 9413.1115. The existence input used by reciprocity is supplied by Definition 9413.1113; the absence of wrong closed packets is supplied by Definition 9413.1112. □

Corollary 9413.1117 (GL_1 detector shadow)**.** *For $G = GL_1$ and the quadratic character attached to $\mathbb{Q}(i)/\mathbb{Q}$, the general finite readout condition reduces at odd primes to*

$$\chi_{-4}(\mathrm{Frob}_p) = 1,$$

equivalently

$$p = a^2 + b^2.$$

Proof. For GL_1, the L-group is $\mathbb{C}^\times$, packets are singletons, component groups and endoscopy disappear on this scope, and class field theory identifies the parameter with a Hecke character. The splitting condition for odd primes in $\mathbb{Q}(i)$ is $\chi_{-4}(\mathrm{Frob}_p) = 1$, which is equivalent to the two-square criterion. □

Theorem 9413.1118 (Main closure boundary)**.** *The main closure theorem is conditional on the named local, trace, multiplicity, endoscopic, Tannakian, Chebotarev, and ledger certificates. It does not construct the traditional analytic, endoscopic, or geometric Langlands inputs; it states that once those inputs are present as BEDC certificates, no residual can remain hidden in a closed object.*

Proof. Every step above invokes a named certificate established or isolated in the preceding modules. The argument is the BEDC composition principle: finite-visible residuals are detected, non-finite residuals are ledgered, and closedness rejects unresolved ledgers. It is not a substitute for constructing the input certificates. □

9413.88 BEDC–Langlands Base Case for GL_1

The abelian case makes the BEDC–Langlands proof chain completely explicit. For $G = GL_1$, Langlands reciprocity is global class field theory. The automorphic side is the group of Hecke characters of the idele class group, the Galois side is the group of continuous characters of the abelianized Weil group, and the finite Hecke–Frobenius detector is a single unramified place.

Definition 9413.1119 (GL_1 Langlands packet)**.** *Let F be a global field and set*

$$C_F := F^\times \backslash \mathbb{A}_F^\times.$$

A GL_1 automorphic object is a Hecke character

$$\chi : C_F \to \mathbb{C}^\times.$$

A GL_1 Galois parameter is a continuous character

$$\phi : W_F^{\mathrm{ab}} \to \mathbb{C}^\times.$$

A BEDC–GL_1 Langlands packet is a row

$$\mathcal{O}^\uparrow = (\chi, \phi, \Lambda_{\mathcal{O}}),$$

where $\Lambda_{\mathcal{O}}$ records ramified places, conductor, archimedean type, norm-scale factors, local ϵ/γ factors, and the Artin reciprocity normalization.

Definition 9413.1120 (GL_1 detector and residual)**.** *For an unramified place v, let ϖ_v be a uniformizer and let Frob_v be the corresponding Frobenius element under the chosen reciprocity convention. The GL_1 detector residual is*

$$D_v(\chi, \phi) := \chi(\varpi_v) - \phi(\mathrm{Frob}_v).$$

The packet matches on the detector v exactly when

$$D_v(\chi, \phi) = 0.$$

Definition 9413.1121 (GL_1 three-properties rows)**.** *The GL_1 time row sends an unramified place to the paired readout*

$$v \longmapsto (\mathrm{Frob}_v, \varpi_v).$$

The unity row says that a unitary finite-order character takes values in S^1, while a general Hecke character is decomposed as

$$\chi = \chi_{\mathrm{unit}} \cdot |\cdot|^s$$

with the norm-scale factor $|\cdot|^s$ recorded in the ledger. The ledger row is precisely the list of ramified, conductor, infinity-type, normalization, and local-factor data not seen by unramified finite detectors.

Theorem 9413.1122 (GL_1 finite detector uniqueness)**.** *Let*

$$\phi, \phi' : W_F^{\mathrm{ab}} \to \mathbb{C}^\times$$

be continuous characters. If

$$\phi(\mathrm{Frob}_v) = \phi'(\mathrm{Frob}_v)$$

for almost all unramified places v, then

$$\phi = \phi'.$$

Proof. Chebotarev density says that the unramified Frobenius classes are dense in the abelianized Galois or Weil quotient on the declared scope. The two characters are continuous and agree on a dense subset, hence agree on its closure. Therefore they are equal. □

Corollary 9413.1123 (GL_1 no-hidden residual)**.** *For a BEDC–GL_1 packet, any mismatch outside the unramified finite detectors must lie in the ramified, conductor, infinity-type, norm-scale, or local-factor ledger.*

Proof. The unramified detector values determine the continuous abelian parameter by Definition 9413.1122. A residual not seen there is therefore not an unramified Frobenius mismatch. The only remaining declared rows are the ledger rows listed in Definition 9413.1121. □

Definition 9413.1124 (Class field reciprocity certificate)**.** *The class field reciprocity certificate for F is the Artin reciprocity map*

$$\mathrm{Art}_F : C_F \to W_F^{\mathrm{ab}}$$

with the convention row fixed so that, at unramified places,

$$\mathrm{Art}_F(\varpi_v) = \mathrm{Frob}_v\,.$$

Changing the inverse convention changes only this normalization row; the detector equality is transported with it.

Theorem 9413.1125 (GL_1 reciprocity)**.** *Assume the class field reciprocity certificate. For every continuous character*

$$\phi : W_F^{\mathrm{ab}} \to \mathbb{C}^\times,$$

the character

$$\chi_\phi := \phi \circ \mathrm{Art}_F$$

is a Hecke character and satisfies

$$\chi_\phi(\varpi_v) = \phi(\mathrm{Frob}_v)$$

for almost all unramified places v. Conversely, every Hecke character $\chi : C_F \to \mathbb{C}^\times$ gives the abelian parameter

$$\phi_\chi := \chi \circ \mathrm{Art}_F^{-1}$$

with the same unramified detector equality.

Proof. The class field reciprocity certificate identifies the idele class group with the abelian Weil side under the declared normalization. Composing a continuous abelian Weil character with Art_F gives a continuous character of C_F, hence a Hecke character. At an unramified place,

$$\chi_\phi(\varpi_v) = \phi(\mathrm{Art}_F(\varpi_v)) = \phi(\mathrm{Frob}_v).$$

The converse is the same argument applied to the inverse reciprocity map. □

Theorem 9413.1126 (GL_1 finite projector)**.** *Let S be a fixed finite set of unramified places and let*

$$\mathsf{Aut}_{S,B}(GL_1) = \{\chi_1, \ldots, \chi_M\}$$

be a finite bounded candidate list. For every abelian parameter ϕ, there is a Laurent polynomial Hecke test function $f_{\phi,S,B}$ on $(\mathbb{C}^\times)^S$ such that

$$\chi_i(\varpi_v)_{v\in S} \neq \phi(\mathrm{Frob}_v)_{v\in S} \quad \Longrightarrow \quad \mathrm{tr}\, \chi_i(f_{\phi,S,B}) = 0,$$

while a matching character evaluates nontrivially.

Proof. For GL_1, the spherical Hecke algebra on S is the Laurent polynomial coordinate ring of $(\mathbb{C}^\times)^S$. The bounded candidate list gives finitely many points

$$X_i = (\chi_i(\varpi_v))_{v\in S},$$

and the parameter gives the point

$$X_\phi = (\phi(\mathrm{Frob}_v))_{v\in S}.$$

The Laurent coordinate ring separates finite closed point sets, so the same interpolation argument as Definition 9413.910 gives a function equal to 1 at X_ϕ and 0 at every displayed nonmatching X_i. □

Theorem 9413.1127 (GL_1 BEDC closure)**.** *For $G = GL_1$, the BEDC–Langlands existence, uniqueness, finite detection, and no-hidden-residual rows are closed by global class field theory.*

Proof. Existence is Definition 9413.1125: every abelian parameter ϕ produces the Hecke character χ_ϕ, and every Hecke character produces the abelian parameter ϕ_χ. Finite detection and uniqueness are Definition 9413.1122. Any remaining residual is forced by Definition 9413.1123 into the ramified, conductor, infinity-type, norm-scale, or local-factor ledger. A closed packet carries no unresolved ledger row. Hence the paired object has no hidden mismatch. □

Corollary 9413.1128 (GL_1 cofinal bridge)**.** *For $G = GL_1$, if each finite Hecke window is read through the character χ_ϕ of Definition 9413.1125, then the cofinal window-to-global bridge has the global alternative immediately: the same Hecke character realizes every unramified window.*

Proof. For every unramified place v, the equality

$$\chi_\phi(\varpi_v) = \phi(\mathrm{Frob}_v)$$

holds outside the declared ramified ledger. Therefore the same automorphic character realizes every finite window in the cofinal tower. The automorphic tail alternative of Definition 9413.1074 is not needed in this abelian base case, because a global realizing row is already present. □

Theorem 9413.1129 (Quadratic splitting detector)**.** *Let $K = \mathbb{Q}(i)$ and let χ_{-4} be the quadratic character attached to $K/\mathbb{Q}$. For every odd prime p, the following conditions are equivalent:*

$$p = a^2 + b^2,$$

$$p \equiv 1 \pmod 4,$$

$$p \text{ splits in } \mathbb{Q}(i),$$

$$\chi_{-4}(\mathrm{Frob}_p) = 1.$$

Under GL_1 reciprocity, these are also equivalent to the corresponding Hecke character ω_{-4} satisfying

$$\omega_{-4}(p) = 1.$$

Proof. Fermat's two-square theorem gives the equivalence between $p = a^2 + b^2$ and $p \equiv 1 \pmod 4$ for odd primes. In $\mathbb{Q}(i)$, a prime $p \neq 2$ splits exactly when -1 is a square modulo p, which is again equivalent to $p \equiv 1 \pmod 4$. Splitting means that the Frobenius element in $\mathrm{Gal}(\mathbb{Q}(i)/\mathbb{Q}) \cong \{\pm 1\}$ is the identity, so $\chi_{-4}(\mathrm{Frob}_p) = 1$. The last equivalence is the detector equality supplied by Definition 9413.1125. □

Corollary 9413.1130 (Two-square theorem as the minimal detector)**.** *The statement that an odd prime is represented by $a^2 + b^2$ is the minimal visible finite detector of GL_1 reciprocity for the quadratic extension $\mathbb{Q}(i)/\mathbb{Q}$.*

Proof. By Definition 9413.1129, the two-square condition is exactly the assertion that the quadratic Frobenius readout equals 1. Under class field reciprocity, the same readout is the Hecke character value $\omega_{-4}(p)$. Thus the classical arithmetic predicate is a single GL_1 Hecke–Frobenius detector. □

Theorem 9413.1131 (Nonabelian lift boundary)**.** *The GL_1 base case closes the abelian Langlands packet. For a general reductive group G, the detector v is replaced by (v, r) with $r : {}^LG \to GL(V_r)$, and the character value is replaced by*

$$\mathrm{tr}\, r(s_v(\pi)) \quad \textit{and} \quad \mathrm{tr}\, r(\phi(\mathrm{Frob}_v)).$$

The nonabelian existence problem is therefore not the GL_1 class field certificate, but the construction of the automorphic packet through the stable trace formula, geometric Langlands functor, or cohomological realization.

Proof. For GL_1, every finite-dimensional representation of the dual group is a character, and class field reciprocity supplies the global automorphic character directly. For general G, finite-dimensional characters of ${}^LG$ separate semisimple conjugacy classes by the Tannakian row, while Chebotarev controls Frobenius density. Those rows close uniqueness and residual detection once a packet exists. The existence of the packet is the separate certificate isolated in section 9413.71. □

9413.89 Unified Relation Atlas

The preceding sections give local worked examples of the same general discipline. A mathematical object is not treated as an isolated point. It is treated as a carriered packet whose visible appearances are reality projections. The atlas in this section is not a catalogue of all mathematics. It is the relation schema governing the examples already in view: common carrier, projection, generalized continuation, transport ledger, and boundary ledger.

Definition 9413.1132 (Unified relation packet)**.** *A* unified relation packet *is a tuple*

$$\mathcal{O} := (\mathsf{Carrier}, \mathsf{Classifier}, (\pi_i)_{i\in I}, \mathsf{Transport}, \mathsf{Ledger}, \mathsf{Boundary})$$

where the maps

$$\pi_i : \mathcal{O} \to R_i$$

read the same carrier in different realities. The reality targets may include algebraic, geometric, analytic, spectral, arithmetic, measure, and categorical interfaces:

$$R_{\text{alg}}, R_{\text{geom}}, R_{\text{analytic}}, R_{\text{spectral}}, R_{\text{arith}}, R_{\text{measure}}, R_{\text{cat}}.$$

The classifier records when two carrier rows are the same for the purpose of the packet. The transport rows record continuation between realities. The ledger records which scale, branch, convergence, prime, zero, or boundary data remain present after a projection.

Principle 9413.1133 (Carriered relation before object isolation)**.** *Within unified reality, a mature mathematical interface is read as a licensed name over displayed carrier data. It is not used as an uninspected primitive when a carrier, classifier, transport row, and boundary ledger are the relevant mathematical content.*

Definition 9413.1134 (Quadratic-norm atlas row)**.** *The* quadratic-norm row *begins with the Pythagorean carrier of section 9413.6:*

$$a^2 + b^2 \sim c^2.$$

After the nonzero scale row $c\#0$ is displayed, the normalized shape projection reads

$$x = \frac{a}{c}, \qquad y = \frac{b}{c}, \qquad x^2 + y^2 \sim 1.$$

In higher dimension the same norm layer has unit-sphere readout

$$S^{n-1} : \quad x_1^2 + \cdots + x_n^2 = 1$$

and symmetry readout

$$O(n) := \{A : A^T A = I\}.$$

The circle and the sphere are unit norm layers. The orthogonal interface is the automorphism read preserving that norm layer.

Theorem 9413.1135 (Quadratic normalization is a scale-to-shape transport)**.** *The quadratic-norm row transports a Pythagorean scale carrier to a unit-shape carrier, while the scale remains in the projection fiber.*

Proof. The two-dimensional case is Definition 9413.218. It sends (a, b, c) with $c\#0$ to $(a/c, b/c)$, preserving the normalized equation and forgetting the visible value of c. The higher-dimensional sphere and orthogonal rows do not change this ledger reading: they expose unit norm and norm-preserving symmetry, not the original scale coordinate. □

Definition 9413.1136 (Phase and transform atlas row)**.** *The* phase and transform row *reads the unit circle as phase, Fourier analysis as character projection, Laplace analysis as real-biased phase transport, and Mellin analysis as Laplace transport in logarithmic coordinates. Its kernels are*

$$e^{in\theta}, \qquad e^{-i\omega t}, \qquad e^{-st}, \qquad x^{-s}.$$

For $s = \sigma + i\omega$ the Laplace kernel factors as

$$e^{-st} = e^{-\sigma t} e^{-i\omega t}.$$

For $x = e^t$ the Mellin kernel reads

$$x^{-s} = e^{-s \log x}.$$

Theorem 9413.1137 (Mellin is logarithmic Laplace transport)**.** *The Mellin kernel x^{-s} is the Laplace kernel on the logarithmic coordinate of the multiplicative axis.*

Proof. Set $t = \log x$. Then $x = e^t$, and hence

$$x^{-s} = (e^t)^{-s} = e^{-st} = e^{-s \log x}.$$

The logarithm transports multiplication to addition:

$$\log(mn) = \log m + \log n.$$

Thus multiplicative arithmetic data can be read as additive Laplace data after the logarithmic transport row is displayed. This is the transport principle already recorded in Definition 9413.226. □

Definition 9413.1138 (Arithmetic spectral atlas row)**.** *The* arithmetic spectral row *reads a coefficient sequence $(a_n)_{n\geq 1}$ as the atomic logarithmic source*

$$\mu_a := \sum_{n\geq 1} a_n \delta_{\log n}$$

with Dirichlet/Laplace readout

$$\mathcal{L}[\mu_a](s) = \sum_{n\geq 1} a_n n^{-s}.$$

For $a_n = 1$, this is the zeta Dirichlet face on its convergence window, as in Definition 9413.224. When the coefficient and convergence rows license multiplicative regrouping, the same carrier has Euler-product face

$$\prod_p (1 - p^{-s})^{-1}.$$

The Dirichlet face is the additive list reality. The Euler face is the prime-local multiplicative reality.

Principle 9413.1139 (Zeta is a shared arithmetic carrier)**.** *Zeta is not merely a sum and not merely a product. In this atlas it is a shared arithmetic carrier whose Dirichlet, logarithmic Laplace, and prime-local faces are compatible only under their displayed convergence, factorization, and transport ledgers.*

Definition 9413.1140 (Theta-functional-equation atlas row)**.** *The* theta-functional-equation row *is the classical transport schema*

Gaussian → theta → Poisson/Fourier symmetry → Mellin readout → Γ-factor → zeta functional equation.

In unified reality, this row says that Fourier symmetry and Mellin/Laplace readout cooperate to produce a functional-equation transport, once the Gaussian, theta, Poisson, Mellin, and convergence ledgers are separately supplied.

Definition 9413.1141 (Prime-channel boundary row)**.** *For*

$$s = \frac{1}{2} + \delta + i\tau$$

the centered prime channel is

$$p^{1/2-s} = p^{-\delta} e^{-i\tau \log p}.$$

Its amplitude read is

$$|p^{1/2-s}| = p^{-\delta}.$$

Therefore the critical line is the prime-channel unitary surface, as recorded in Definition 9413.228.

Principle 9413.1142 (RH is zero-rigidity at the unitary boundary)**.** *The RH target is read in this atlas as a zero-rigidity boundary for the zeta carrier: nontrivial zeros in the critical strip must lie on the prime-channel unitary surface. This is a route statement, not the missing analytic bridge; the bridge obligations remain those isolated in section 9413.16.*

Definition 9413.1143 (Spectral-curve atlas row)**.** *The* spectral-curve row *is the chain*

rational generating function → linear recurrence → companion matrix → spectral polynomial → spectral curve →

For the quartic worked example this row is realized by

$$Z(t, y) = \frac{N(t, y)}{D(t, y)}, \qquad \det(\lambda I - T(y)) = \Pi(\lambda, y),$$

and by the elliptic normalization

$$E: \quad Y^2 = 4\lambda^3 - 4\lambda + 1.$$

Theorem 9413.1144 (Spectral curves mediate recurrence and zero reality)**.** *The quartic spectral row mediates recurrence algebra, matrix spectra, elliptic branch geometry, dominance, and endpoint-zero reality without identifying these projections as the same visible object.*

Proof. This is the content of Definition 9413.106. The recurrence, companion matrix, spectral polynomial, affine curve, elliptic normalization, dominance row, and endpoint-zero law are projections of the fixed quartic carrier. The carrier makes the projections comparable; it does not erase the boundary ledgers attached to each projection. □

Definition 9413.1145 (Partition and zero-boundary row)**.** *A* partition zero-boundary row *has exponential-sum form*

$$Z(\beta, h) := \sum_{\omega \in \Omega} e^{-\beta E(\omega) + hM(\omega)}.$$

It is a weighted Laplace/Fourier readout of a state space. Zeros of such a readout record phase cancellation or boundary behavior in the corresponding model. Any stronger theorem, such as a Lee-Yang location theorem, requires its own model-specific carrier and proof ledger.

Definition 9413.1146 (Heat-kernel and spectral-zeta row)**.** *For a displayed operator with eigenvalue ledger* (λ_n)*, the heat trace read is*

$$\mathrm{Tr}(e^{-t\Delta}) = \sum_n e^{-t\lambda_n}.$$

The spectral zeta read is

$$\zeta_\Delta(s) = \sum_n \lambda_n^{-s}.$$

Thus a geometric operator can project to a spectral measure, to its heat Laplace readout, and to its Mellin-type spectral zeta readout. Analytic continuation, pole structure, and determinant interpretations remain separate boundary ledgers.

Definition 9413.1147 (Elliptic, modular, and L-function row)**.** *An elliptic curve row has branch-cover form*

$$E: \quad Y^2 = P_3(X).$$

When a modular or automorphic bridge is displayed, the curve may be read through a modular packet f_E *and an L-function readout*

$$L(E,s) = L(f_E,s) = \sum_{n\geq 1} a_n n^{-s}.$$

In this atlas, the point is the transport shape: elliptic geometry, period data, modular spectra, Dirichlet/Mellin readouts, and zeros belong to one bridgeable packet only when the required bridge rows are present.

Definition 9413.1148 (Cross-domain bridge ledger)**.** *The* cross-domain bridge ledger *records high-level transports between realities. Examples include* $\mathsf{StoneDuality}^\uparrow$, $\mathsf{GelfandDuality}^\uparrow$, $\mathsf{PontryaginDuality}^\uparrow$, $\mathsf{TannakaKrein}^\uparrow$, $\mathsf{RiemannHilbert}^\uparrow$, $\mathsf{HodgeBridge}^\uparrow$, $\mathsf{Langlands}^\uparrow$, *and* $\mathsf{MirrorSymmetry}^\uparrow$, *as listed in section 9467.17. Each bridge is a transport ledger between realities, not a permission to conflate the source and target interfaces without the stated classifier and boundary rows.*

Layer	**Visible read**	**Unified-reality reading**
Quadratic norm	$a^2 + b^2 = c^2$	scale-shape carrier
Unit shape	$x_1^2 + \cdots + x_n^2 = 1$	normalized norm reality
Symmetry	$A^T A = I$	norm-preserving automorphism
Phase	$e^{i\theta}$	circle phase coordinate
Fourier	$e^{in\theta}$	character and frequency projection
Laplace	e^{-st}	phase with real damping direction
Mellin	x^{-s}	Laplace read on logarithmic coordinates
Zeta	$\sum n^{-s}$	arithmetic atomic source readout
Euler product	$\prod_p (1-p^{-s})^{-1}$	prime-local multiplicative face
Theta/Gamma	Poisson plus Mellin	functional-equation transport
Elliptic curve	$Y^2 = P_3(X)$	branch and period carrier
Spectral curve	$\Pi(\lambda, y) = 0$	recurrence-matrix-geometry carrier
Heat trace	$\sum e^{-t\lambda_n}$	operator-spectrum Laplace read
L-function	$\sum a_n n^{-s}$	arithmetic or automorphic spectral read
RH boundary	$\Re(s) = 1/2$	prime-channel unitary zero boundary

Theorem 9413.1149 (Relation atlas does not collapse projections)**.** *No row in the relation atlas may replace the carrier by one of its visible projections unless the corresponding classifier, transport, fiber, and boundary ledgers are displayed.*

Proof. This is the atlas form of Definition 9413.24. The quadratic row forgets scale under the circle projection. The Laplace and Mellin rows add damping and logarithmic transport not present in pure Fourier phase. The zeta row has both Dirichlet and Euler faces only with the relevant convergence and factorization ledgers. The spectral-curve row separates recurrence, matrix, branch, dominance, and endpoint-zero data. Therefore projection visibility is never sufficient by itself to replace the common carrier. □

Theorem 9413.1150 (Unified relation atlas principle)**.** *Triangles, circles, spheres, orthogonal symmetries, Fourier modes, Laplace kernels, Mellin kernels, zeta, spectral curves, elliptic carriers, heat traces, L-functions, and RH boundary rows form a compatible relation atlas when read as carrier projections connected by transport ledgers. Their compatibility is not object-level identity; it is common-carrier projection disciplined by noncollapse and boundary accounting.*

Proof. The common-carrier and generalized-continuation rules are Definition 9413.4 and Definition 9413.12. The Pythagorean scale-shape row is Definition 9413.218. The phase-transform-zeta chain is Definition 9413.232. The quartic spectral chain is Definition 9413.106. The RH boundary route is Definition 9413.1142. The cross-domain bridge layer is Definition 9413.1148. Taken together, these rows supply a relation atlas: transport between realities is licensed by displayed ledgers, and every projection remains boundary-relative. □

Chapter 9414

Reality-Constrained Approximation Towers

Mathematical approximation and physical adequacy are not the same relation. A mathematical object may admit several convenient representations, and those representations may be equivalent under a chosen mathematical classifier. When the representation is used as a model of a real system, however, the relevant classifier is no longer freely chosen. It is constrained by the reality-facing properties through which the system responds to admissible observation and intervention.

This chapter records the resulting conceptual framework. BEDC already treats equality, function, limit, real number, circle, and other mature objects as derived interfaces rather than primitive givens. Compression, packaging, projection, sealing, and identification must carry source, classifier, stability, and GAP ledger data. The same discipline applies to physical modelling: a mathematical representation is physically admissible only when the classifier that licenses it is compatible with the physical signature bundle under which the represented system is queried.

9414.1 Reality-facing signatures

Definition 9414.1 (Reality-facing property)**.** *A reality-facing property of a system is a property that can affect the system's stable interaction profile under some admissible physical procedure. Removing, smoothing, quotienting, or replacing such a property by an approximate surrogate can change at least one physically relevant response.*

Mass, charge, curvature, boundary differentiability, stress concentration, causal accessibility, optical response, and measurement-stable phase information are standard examples. A purely mathematical feature becomes reality-facing only when it enters such a response profile.

Definition 9414.2 (Mathematical classifier)**.** *A mathematical classifier $\sim_{\mathcal{M}}$ is an equivalence or comparison relation that groups mathematical representations according to selected mathematical features. It becomes physically relevant only when the grouped features coincide with the reality-facing properties activated by the physical system under the observation bundle in use.*

Definition 9414.3 (Ontic signature source)**.** *Let O be a system and let Π_{phys} be a bundle of physically admissible questions. The physical signature of O relative to Π_{phys} is written*

$$\mathrm{PhysSig}_{\Pi_{\mathrm{phys}}}(O).$$

The ontic signature source of O, relative to Π_{phys}, is the invariant source of this signature. It is not identified with any particular mathematical representation.

Two systems are physically indistinguishable at the resolution Π_{phys} only when their signatures agree:

$$\text{PhysSig}_{\Pi_{\text{phys}}}(O_1) \sim \text{PhysSig}_{\Pi_{\text{phys}}}(O_2).$$

This is a signature-level sameness judgment, not host-level identity.

Definition 9414.4 (Mathematical representation)**.** *A mathematical representation of an ontic signature source O is a structured mathematical object M together with a representation route*

$$\rho : O \rightsquigarrow M$$

and an interpretation procedure that reads features of M as candidate reality-facing properties of O.

The representation is mathematically adequate relative to a mathematical classifier $\sim_{\mathcal{M}}$ when it preserves the mathematical features selected by that classifier. It is physically adequate relative to Π_{phys} only when its induced physical signature agrees with the signature of the represented system:

$$\text{Sig}_{\Pi_{\text{phys}}}(M) \sim \text{PhysSig}_{\Pi_{\text{phys}}}(O).$$

Definition 9414.5 (Physical-fit relation)**.** *For mathematical representations M_1 and M_2, the physical-fit relation relative to Π_{phys} is*

$$M_1 \equiv_{\text{phys},\Pi} M_2$$

exactly when

$$\text{Sig}_{\Pi_{\text{phys}}}(M_1) \sim \text{Sig}_{\Pi_{\text{phys}}}(M_2).$$

The physical-fit relation may be strictly finer than a mathematical approximation relation. Polygonal approximations and a smooth circle may agree under area, perimeter, or Hausdorff-distance classifiers while separating under tangent-field, curvature, reflection, stress-response, or boundary-layer classifiers.

Definition 9414.6 (Representation gap)**.** *A representation gap occurs when two representations are identified by a mathematical classifier but separated by a physically active signature:*

$$M_1 \sim_{\mathcal{M}} M_2 \quad \textit{and} \quad M_1 \not\equiv_{\text{phys},\Pi} M_2.$$

The gap is evidence that the mathematical classifier is too coarse for the physical question being asked.

Definition 9414.7 (Reality-constrained classifier)**.** *A mathematical classifier $\sim_{\mathcal{M}}$ is reality-constrained by Π when*

$$M_1 \sim_{\mathcal{M}} M_2 \quad \Longrightarrow \quad M_1 \equiv_{\text{phys},\Pi} M_2.$$

Equivalently, every mathematical identification licensed by $\sim_{\mathcal{M}}$ preserves the physical signature exposed by Π.

Definition 9414.8 (Reality-constrained meaning)**.** *Let C be a closed observational context, let x be a displayed record or representation available to C, let ℓ be a label, and let κ be a classifier with $\kappa(x) = \ell$. The label has reality-constrained meaning for C relative to an observation bundle Π when the assignment is carried by*

$$\text{Meaning}_{C,\Pi}(x, \ell, \kappa) = (\text{SourceRecord}, \text{Classifier}, \text{RealityConstraint}, \text{ContinuationConsequence}, \text{GapLedger}).$$

The source record locates x in the finite observation history, the classifier supplies the label, the reality constraint requires the classifier to preserve the Π-signature, the continuation consequence records which downstream modelling, prediction, action, or proof use depends on the label, and the gap ledger records the provenance and unclosed supply exposed by that dependence.

Theorem 9414.9 (Classification alone does not supply meaning)**.** *A label assignment $\kappa(x) = \ell$ does not by itself supply reality-constrained meaning.*

Proof. By Definition 9414.8, reality-constrained meaning requires five fields: source record, classifier, reality-facing constraint, continuation consequence, and gap ledger. The equation $\kappa(x) = \ell$ supplies only the classifier field at the displayed record. It does not state that κ is reality-constrained by Π in the sense of Definition 9414.7; it does not specify the continuations for which the label is reused; and it does not record the provenance or unclosed residue exposed by that reuse. Hence the label assignment alone is insufficient. □

Corollary 9414.10 (Meaning is constrained continuation)**.** *Reality-constrained meaning is not an unsupported label attached to data. It is a classifier whose distinctions remain stable under the reality-facing continuations for which the label is used.*

Proof. The classifier field supplies the label, while the reality-constraint and continuation-consequence fields state why that label can be reused. If the relevant continuation exposes a signature separation, then Definition 9414.6 records a representation gap rather than a meaningful identification. □

Principle 9414.11 (Reality as meaning constraint)**.** *Reality does not enter this interface as a representation, label, or classifier. It enters as the stable source of signatures that can support, refine, or break the classifier when the record is continued through admissible observation, intervention, prediction, or proof use.*

Corollary 9414.12 (Far-end supply is not an object equation)**.** *When the reason a reality-constrained classifier continues to hold exceeds the closed record available to C, the exposed boundary is a T-supply socket. This does not assert an object-level equation $T = \mathrm{Reality}$.*

Proof. The excess is recorded in the gap-ledger field of Definition 9414.8. By the socket discipline of Definition 9462.842, a commitment exceeding closed records and closed-generation rules is exposed as a gap whose far end is the apophatic supply position T. This definition therefore permits the reading of T as the far-end position of reality-facing constraint, while refusing to internalize T as a system object, representation, or substrate axiom. □

Theorem 9414.13 (Formal consistency does not select actuality)**.** *Internal consistency of a representation, classifier, or label assignment does not imply reality-constrained meaning relative to an observation bundle.*

Proof. Internal consistency says that the displayed syntax, rules, or classifier rows do not contradict the system's own formation discipline. Reality-constrained meaning requires the stronger fields of Definition 9414.8: preservation of the reality-facing signature, stable downstream use, and a ledger for provenance and unclosed supply. A classifier can be internally consistent while still producing a representation gap in the sense of Definition 9414.6. Such a gap shows that the classifier is too coarse for the observation bundle. Hence formal possibility does not select actual physical order. □

Corollary 9414.14 (Logic licenses possibility, not physical order)**.** *Formal consistency licenses internal possibility. Physical order is reported only through signature preservation, continuation stability, and ledgered exposure of residue.*

Proof. The first clause is the internal consistency condition of Definition 9414.13. The second clause is exactly the additional reality-facing and ledgered structure required by Definition 9414.8. □

Theorem 9414.15 (The supply loop is not circular proof)**.** *The passage from classifier use to reality-facing response, and then to ledger refinement or exposed T-socket, is not a proof of a classifier by that same classifier.*

Proof. A circular proof would use the same internal assertion as both support and conclusion. The reality-constrained passage has a different shape. If the continuation yields a stable signature, the record ledger is enlarged by evidence of preservation. If the continuation separates the signatures, Definition 9414.6 records a gap rather than a theorem. If the reason for continued preservation exceeds the available closed record, the gap-ledger field exposes the far-end supply position described in Definition 9414.12. In none of these cases is the classifier licensed merely by repeating its own label assignment. □

Concretized in §1213: Reality-constrained signature residues realize the representation-gap row as a finite NameCert packet separating mathematical classifier agreement from physical adequacy.

Concretized in §1303: Reality-constrained signature descents realize operation descent as a finite packet joining mathematical classifier transport, physical-signature exposure, GAP rows, representation certificates, and a descent witness.

Concretized in §1348: External supply signature seals realize the reality-facing signature and no-free-transfer discipline as one finite GAP, descent, ledger, and NameCert packet.

9414.2 Ledgered approximation towers

Definition 9414.16 (GAP-admissible compression)**.** *A compression*

$$C : X \rightsquigarrow Y$$

is GAP*-admissible when it carries a source domain of admissible histories, a signature-generation policy, a package policy assigning visible tokens to signatures, a gap ledger recording which source histories lie behind each visible token, and a stability condition specifying which operations descend through the compression.*

This is the physical reading of BEDC's finite globalization discipline. Package tokens are not quotient classes accepted without provenance. They are visible rows whose source histories remain available through GAP and package policies; see chapter 9389 and chapter 9395.

Definition 9414.17 (Approximation tower)**.** *An approximation tower is a finite or transfinite chain of* GAP*-admissible compressions*

$$O \longrightarrow D_0 \longrightarrow D_1 \longrightarrow \cdots \longrightarrow D_n \longrightarrow M,$$

where O is the ontic signature source, the intermediate D_i are observation histories, finite data, discrete models, continuous models, smooth models, or other certified representation layers, and M is the visible mathematical object used for reasoning.

Each arrow compresses, packages, hides, regularizes, completes, or idealizes source material. The tower is admissible only when each arrow carries a ledger and the composite tower carries the composite ledger.

Definition 9414.18 (Operational descent)**.** *An operation F descends through a classifier $\sim$ when there is a target classifier $\sim'$ such that*

$$x \sim y \quad \Longrightarrow \quad F(x) \sim' F(y).$$

A descent certificate is a proof that the operation respects the source classifier and lands in the target classifier.

Without such a certificate, sameness before applying F does not imply sameness after applying F. The physically important case is differentiation: a classifier that identifies two shapes by coarse geometric approximation does not automatically permit tangent fields, curvature measures, boundary conditions, or dynamics to descend through it.

Definition 9414.19 (Physical representation certificate)**.** *A physical representation certificate for a model M of a system O relative to Π is a five-field certificate*

$$\mathrm{PhysCert}(M, O; \Pi) = (\mathrm{SourceSpec}, \mathrm{PatternSpec}, \mathrm{ClassifierSpec}, \mathrm{StabilityCert}, \mathrm{LedgerPolicy}).$$

The fields record the observation traces and conditions that license the model, the stable pattern claimed by the model, the comparison rule between model-generated and reality-facing signatures, the stability data supporting reuse and prediction, and the record of what was compressed, ignored, idealized, or hidden.

Principle 9414.20 (Reality-signature priority)**.** *Physical equivalence is determined by reality-facing signatures, not by mathematical convenience alone. If*

$$M_1 \sim_{\mathrm{math}} M_2$$

but

$$\mathrm{Sig}_{\Pi_{\mathrm{phys}}}(M_1) \not\sim \mathrm{Sig}_{\Pi_{\mathrm{phys}}}(M_2),$$

then M_1 and M_2 are mathematically similar but physically inequivalent relative to Π_{phys}.

Principle 9414.21 (No free transfer of structure)**.** *If a compression identifies two source histories, no operation on those histories may be transported across the compression unless a stability or descent certificate has been supplied.*

Principle 9414.22 (Compression requires physical memory)**.** *Every physically meaningful idealization must record what it hides. When a model replaces rough, discrete, finite, noisy, or staged source material by a smooth visible object, the lost source structure must remain available in a ledger whenever later physical questions may depend on it.*

9414.3 Physical adequacy theorems

Theorem 9414.23 (Mathematical approximation does not imply physical equivalence)**.** *Let M_1 and M_2 be mathematical representations. Suppose $M_1 \sim_{\mathrm{math}} M_2$ under a mathematical classifier. If*

$$\mathrm{Sig}_{\Pi_{\mathrm{phys}}}(M_1) \not\sim \mathrm{Sig}_{\Pi_{\mathrm{phys}}}(M_2),$$

then M_1 and M_2 are not physically equivalent relative to Π_{phys}.

Proof. By Definition 9414.5, physical equivalence relative to Π_{phys} is sameness of induced physical signatures. The hypothesis gives failure of such signature sameness. Therefore the physical-fit relation does not hold, although the mathematical classifier identifies the two representations. □

Corollary 9414.24 (Physics may refine mathematical equivalence)**.** *A physical classifier may refine, split, or reject a mathematical equivalence class.*

Proof. Let $[M]_{\text{math}}$ be a mathematical equivalence class. If there are $M_1, M_2 \in [M]_{\text{math}}$ whose physical signatures differ, then Definition 9414.23 separates them in the physical-fit relation. Thus the physical classifier partitions the mathematical class into finer classes or rejects some members as inadequate for the intended physical use. □

Corollary 9414.25 (Mathematical convenience is not a physical certificate)**.** *A mathematically convenient representation is physically admissible only when the classifier that licenses its convenience is reality-constrained for the intended signature bundle.*

Proof. Mathematical convenience supplies a reason to choose a representation or classifier. By Definition 9414.23, however, mathematical equivalence under that classifier does not by itself imply preservation of physical signature. Physical admissibility therefore requires the additional reality-constrained condition of Definition 9414.7. □

Theorem 9414.26 (Polygonal convergence does not certify differentiability descent)**.** *Let P_n be regular polygons approximating a circle C, and let $\sim_0$ be a coarse geometric classifier such as agreement in area, perimeter, finite-resolution visibility, or Hausdorff distance. If the polygonal histories and the circle history are packaged into the same visible circle-like token under $\sim_0$, then an operation extracting differentiable boundary structure does not automatically descend through $\sim_0$.*

Proof. The classifier $\sim_0$ records coarse shape agreement. It does not, by itself, record tangent-field or curvature data. Each finite polygon has vertices at which the boundary direction changes discontinuously. The circle has a tangent at every point and constant curvature. Hence polygonal source histories and smooth-circle histories may agree under coarse signatures while separating under differentiability signatures. If differentiable-boundary extraction descended through $\sim_0$, coarse sameness would imply sameness after extraction. The displayed separation contradicts that implication. A separate descent certificate is therefore required. □

Corollary 9414.27 (Circle-like tokens are classifier-relative)**.** *A visible package token produced by polygonal approximation may behave as a circle under coarse shape queries while failing to be a smooth circle under differentiability-sensitive queries.*

Proof. By Definition 9414.26, the coarse classifier may license a shared visible token without licensing differentiability descent. The token is therefore circle-like only relative to the classifier that licensed the package. □

Corollary 9414.28 (Hidden vertices remain physically relevant when queried)**.** *If a physical system is sensitive to tangent discontinuity, curvature concentration, stress concentration, or boundary scattering, then polygonal approximants and a smooth circle may belong to different physical-fit classes.*

Proof. Vertex singularities affect tangent and curvature structure and may affect stress, reflection, or boundary dynamics. If the observation bundle Π includes such responses, then the induced physical signatures of the polygonal approximants and the smooth circle separate. By Definition 9414.5, this separation is exactly failure of physical-fit equivalence. □

Theorem 9414.29 (Composite towers retain source dependence)**.** *Let*

$$X_0 \xrightarrow{C_1} X_1 \xrightarrow{C_2} X_2 \xrightarrow{C_3} \cdots \xrightarrow{C_n} X_n$$

be an approximation tower in which each compression carries a GAP *ledger satisfying coverage. Then the composite compression from X_0 to X_n has a composite ledger recording which original source histories in X_0 lie behind each final visible token in X_n.*

Proof. For two stages, a source $x \in X_0$ lies behind a final token $z \in X_2$ exactly when there exists an intermediate token $y \in X_1$ such that x lies in the C_1 ledger behind y and y lies in the C_2 ledger behind z. This is the composite-gap construction. For more stages, apply the two-stage construction inductively: the composite ledger to X_{n-1} composes with the ledger for C_n, yielding a ledger from X_0 to X_n. □

Corollary 9414.30 (Hidden source structure can become physically active)**.** *A feature hidden at an early level of an approximation tower may become relevant when a later physical query depends on it.*

Proof. By Definition 9414.29, the hidden source structure is retained by a ledger-complete tower. If a later query, such as differentiation, curvature response, stress concentration, or causal propagation, depends on that hidden structure, then the query must inspect the ledger rather than only the final visible token. □

Theorem 9414.31 (Physical admissibility requires signature preservation and descent)**.** *Let M be a mathematical representation of a system O, and let Π_{phys} be the physical signature bundle relevant to the intended modelling context. If M is physically admissible for O relative to Π_{phys}, then the following conditions hold:*

$$\mathrm{Sig}_{\Pi_{\text{phys}}}(M) \sim \mathrm{PhysSig}_{\Pi_{\text{phys}}}(O),$$

and every operation used to extract physical predictions from M descends through the classifier by which M was admitted.

Proof. If signature preservation fails, M does not reproduce the relevant reality-facing properties of O and is not physically admissible. If operational descent fails, then there is an operation F and classifier-same inputs $M_1 \sim M_2$ with $F(M_1) \not\sim' F(M_2)$. Predictions depending on F are then not well defined at the visible package level; they depend on hidden source history rather than on the admitted model token. Both conditions are therefore necessary. □

Corollary 9414.32 (Physical representation is unique only up to physical fit)**.** *There may be many mathematical descriptions of one system, but relative to a complete physical signature bundle they occupy one admissible physical-fit class.*

Proof. If M_1 and M_2 are physically admissible representations of the same O, Definition 9414.31 gives

$$\mathrm{Sig}_{\Pi_{\text{phys}}}(M_1) \sim \mathrm{PhysSig}_{\Pi_{\text{phys}}}(O) \sim \mathrm{Sig}_{\Pi_{\text{phys}}}(M_2).$$

By transitivity of signature sameness, $M_1 \equiv_{\text{phys},\Pi} M_2$. □

Theorem 9414.33 (Refining physical questions refines model fit)**.** *Let* $\Pi \subseteq \Pi'$ *be physical signature bundles, and let* $\equiv_\Pi$ *and* $\equiv_{\Pi'}$ *be the induced physical-fit relations. Then*

$$M_1 \equiv_{\Pi'} M_2 \quad \Longrightarrow \quad M_1 \equiv_\Pi M_2.$$

The converse need not hold.

Proof. Agreement under Π' includes agreement on every question in Π, since every question in Π is also in Π'. Hence agreement under the larger bundle restricts to agreement under the smaller bundle. The converse fails when Π' contains a question omitted by Π. Polygonal approximants and a smooth circle may agree under a coarse shape bundle while separating under a differentiability or curvature bundle. □

Corollary 9414.34 (Approximation is tower-relative)**.** *An approximation statement is indexed by the tower level and signature bundle at which it is asserted.*

Proof. By Definition 9414.33, changing the signature bundle can refine the equivalence relation. Since an approximation tower is a staged sequence of classifiers and compressions, an approximation judgment is meaningful only relative to the layer and bundle that license it. □

Theorem 9414.35 (Physical adequacy requires a representation certificate)**.** *If a model* M *is used as a physically adequate representation of a system* O *relative to* Π*, then* M *must carry a physical representation certificate.*

Proof. Physical adequacy requires an observational source, a pattern extracted from that source, a classifier comparing model-generated and reality-facing signatures, a stability condition licensing continuation beyond observed data, and a ledger recording the compression performed by the model. Without source data the model is disconnected from observation. Without a pattern there is no stated behaviour being modelled. Without a classifier there is no fit judgment. Without stability data finite fit cannot support prediction. Without a ledger hidden source structure cannot be recovered when it becomes physically relevant. These are precisely the fields of Definition 9414.19. □

Corollary 9414.36 (An equation is not yet a physical model)**.** *An equation becomes a physical model only when embedded in a certificate specifying source, pattern, classifier, stability, and ledger.*

Proof. An equation gives a formal relation among mathematical symbols. It does not by itself specify which observations licensed it, which physical signatures it preserves, what classifier compares it with reality, why it may be reused predictively, or what idealizations it hides. By Definition 9414.35, those fields are required for physical adequacy. □

9414.4 Interface–constraint–section packets

Remark 9414.37 (Paper-side status)**.** *This section gives a paper-side interface discipline for reality-constrained inquiry and has no Lean target in this chapter: a source is not consumed as a bare object, an observation interface compresses it to records, constraints decide which records remain jointly admissible, sections supply licensed fiber-level meaning, and the public surface is the ledger that makes these passages auditable. The packet is a derived vision interface over* Hist*,* hsame*,* Cont*,* Sig*,* Pkg*,* Gap*, and* NameCert*; it is not a primitive of the BEDC finite kernel and not an object-level theory of consciousness, probability, or quantum mechanics.*

Definition 9414.38 (Interface–constraint–section packet)**.** *An* interface–constraint–section packet *is a finite records-side packet*

$$\mathcal{I} = (X, \sim_X, \Lambda_X, Y, \sim_Y, O, C, Q, \pi, \sigma, K, \Lambda, A)$$

with the following rows:

1. *X is a displayed source carrier with classifier $\sim_X$ and source ledger Λ_X;*
2. *$O : X \to Y$ is an observation interface into a displayed record carrier with classifier $\sim_Y$;*
3. *C is a constraint row on X, or on a displayed relation over X, whose purpose is to decide which source witnesses are jointly admissible for the active observation bundle;*
4. *Q is a quotient, package, or compressed visible surface reached from X by a displayed projection $\pi : X \to Q$;*
5. *$\sigma : Q \to X$ is an optional section row, admitted only when its authorization, action-response, witness, boundary, and stability ledgers are displayed;*
6. *K is an outcome kernel or weight row attached to an interface, used only as a bridge row when the source discipline supplies such weights;*
7. *Λ is the public ledger recording observation, outcome, constraint, section, gap, and provenance rows;*
8. *A is the audit row refusing bare source access, unledgered compression, quotient equality as meaning, hidden section choice, outcome selection by a subject, and object-level resolution of indistinguishable bridge interpretations.*

Principle 9414.39 (Layer discipline)**.** *The packet separates four layers. The source layer records what may generate observations. The interface layer records what is available to a finite observer. The section layer records fiber-level meaning only when a section is licensed by displayed authorization and witness data. The ledger layer records which parts of the preceding passage are public, transportable, and open to defeat. Collapsing these layers erases the certificate fields required by reality-constrained meaning in Definition 9414.8.*

Definition 9414.40 (Readout layer)**.** *For a displayed source carrier X, a* readout layer *is a packet*

$$\mathcal{L} = (R, \sim_R, P, \Lambda_R, \mathsf{NameCert}_{\mathcal{L}})$$

where R is a displayed record carrier, $\sim_R$ is its classifier, $P : X \to R$ is a readout row, Λ_R records provenance and gap data, and $\mathsf{NameCert}_{\mathcal{L}}$ names the carrier-classifier surface. The discrete case is the special case in which $\sim_R$ is the equality classifier at the displayed resolution.

Definition 9414.41 (Layer fiber family)**.** *For a readout layer $\mathcal{L}$ and a record $r : R$, the fiber over r is*

$$\mathrm{Fib}_{\mathcal{L}}(r) := \{x \in X \mid P(x) \sim_R r\}.$$

The layer-infinite residue of $\mathcal{L}$ is the fiber family

$$\infty_{\mathcal{L}} : R \to \mathsf{Type}, \qquad \infty_{\mathcal{L}}(r) := \mathrm{Fib}_{\mathcal{L}}(r).$$

Thus the residue is not a further point of R; it is the family of source witnesses still hidden behind each displayed record row.

Definition 9414.42 (Total fiber surface)**.** *The total fiber surface of a readout layer is*

$$\mathrm{Tot}(\infty_{\mathcal{L}}) := \sum_{r:R} \mathrm{Fib}_{\mathcal{L}}(r).$$

When $\sim_R$ has a displayed reflexivity row, the readout P induces a source-to-total map

$$\Phi_{\mathcal{L}} : X \to \mathrm{Tot}(\infty_{\mathcal{L}}), \qquad \Phi_{\mathcal{L}}(x) := (P(x), (x, \mathrm{refl}_{P(x)})).$$

There is also the source projection

$$\Psi_{\mathcal{L}} : \mathrm{Tot}(\infty_{\mathcal{L}}) \to X, \qquad \Psi_{\mathcal{L}}(r, x, \epsilon) := x.$$

Theorem 9414.43 (Displayed source readback)**.** *For every readout layer with displayed reflexivity for $\sim_R$,*

$$\Psi_{\mathcal{L}} \circ \Phi_{\mathcal{L}} = \mathrm{id}_X.$$

Consequently the total fiber surface remembers every displayed source row, even when the public record carrier R forgets source distinctions.

Proof. For a source row $x : X$, the map $\Phi_{\mathcal{L}}$ sends x to $(P(x), (x, \mathrm{refl}_{P(x)}))$. Applying $\Psi_{\mathcal{L}}$ forgets the displayed record and the classifier witness, leaving x. This proves the stated identity. □

Definition 9414.44 (Readout refinement)**.** *Let*

$$\mathcal{L} = (R, \sim_R, P, \Lambda_R, \mathsf{NameCert}_{\mathcal{L}})$$

and

$$\mathcal{M} = (S, \sim_S, Q, \Lambda_S, \mathsf{NameCert}_{\mathcal{M}})$$

be readout layers over the same source carrier X. The layer $\mathcal{M}$ refines $\mathcal{L}$ when there is a displayed readback map

$$\rho : S \to R$$

such that every source row satisfies

$$\rho(Q(x)) \sim_R P(x).$$

Theorem 9414.45 (Refinement partitions the previous fiber)**.** *If $\mathcal{M}$ refines $\mathcal{L}$, then every element of $\mathrm{Fib}_{\mathcal{L}}(r)$ determines an element of*

$$\sum_{s:S} (\rho(s) \sim_R r) \times \mathrm{Fib}_{\mathcal{M}}(s).$$

In the discrete case, when both classifiers are equality classifiers and $P = \rho \circ Q$, this construction is an equivalence.

Proof. Given $(x, \epsilon) \in \mathrm{Fib}_{\mathcal{L}}(r)$, put $s := Q(x)$. The refinement row gives $\rho(Q(x)) \sim_R P(x)$, while ϵ gives $P(x) \sim_R r$. Transitivity of the displayed classifier gives $\rho(s) \sim_R r$, and reflexivity gives $(x, \mathrm{refl}_{Q(x)}) \in \mathrm{Fib}_{\mathcal{M}}(s)$. This is the announced map. In the discrete case the reverse map sends $(s, q, (x, h))$ to $(x, P(x) = r)$ using

$$P(x) = \rho(Q(x)) = \rho(s) = r.$$

The two composites are the identity by equality elimination. □

Principle 9414.46 (Layer residue is fiber data)**.** *An interface record closes only the displayed classifier value. The residual content of the layer is the fiber family behind that value, and a finer layer can expose that residue only by displaying the readback map, the classifier transport, and the accompanying ledger rows. This is the formal reason that quotient equality is not yet section-level meaning.*

Definition 9414.47 (Interface fiber and compressed reality)**.** *For a packet $\mathcal{I}$ with interface $O : X \to Y$, the interface fiber of a source row x is*

$$[x]_O := \{x' \in X \mid O(x') \sim_Y O(x)\}.$$

The compressed records-side reality exposed by O is the pair

$$\mathsf{Real}_O(\Lambda) := (O(X), \Lambda_O),$$

where Λ_O is the ledger of displayed observation rows, interface provenance, and gap reports. This is not the bare source X.

Definition 9414.48 (Constraint compatibility)**.** *Let $A : X \to A_0$ and $B : X \to B_0$ be two displayed readout rows and let C be a constraint row. The records a and b are compatible under C, written $a \ \mathsf{compat}_C \ b$, when there is a source witness x such that*

$$\begin{aligned} &C(x), \\ &A(x) \sim a, \\ &B(x) \sim b. \end{aligned}$$

The readout A determines B under C on the active surface when, for each displayed a, the set of compatible B-readouts has at most one classifier value.

Theorem 9414.49 (Constraint projection)**.** *If the current packet records $C(x)$ and $A(x) \sim a$, then every admissible B-readout lies in*

$$B\big(\{u \in X \mid C(u) \text{ and } A(u) \sim a\}\big).$$

If this displayed image has one classifier value b, then $A = a$ determines $B = b$ under C on the active surface.

Proof. The premise $C(x)$ restricts admissible source witnesses to the constraint row. The premise $A(x) \sim a$ further restricts them to the fiber of the active A-readout. Reading B from the remaining witnesses gives exactly the displayed image. If that image has one classifier value, every compatible source witness has the same B-readout, so the determination claim follows. □

Definition 9414.50 (Self-similar readout diagram)**.** *A self-similar readout diagram over an interface–constraint packet is a displayed index family I whose ith row is a packet*

$$\mathcal{S}_i = (R_i, \sim_i, P_i, C_i, \alpha_i, \Lambda_i, \mathsf{Cert}_i, A_i).$$

Here R_i is the visible record carrier, $\sim_i$ is its classifier, $P_i : X \to R_i$ is its source readout, C_i is its admissibility row, $\alpha_i : R_i \to R_i$ is its self-similar continuation operation, Λ_i is the provenance and gap ledger, Cert_i is the name, stability, transport, and replay certificate surface, and A_i is the audit row. For each displayed edge $j \to i$ the diagram contains a projection row

$$\rho_{ji} : R_j \to R_i$$

such that source readout and continuation are compatible:

$$\rho_{ji}(P_j(x)) \sim_i P_i(x),$$
$$\rho_{ji}(\alpha_j(r)) \sim_i \alpha_i(\rho_{ji}(r)).$$

The family I is part of the displayed packet; it is not an untyped quantification over all possible self-similar systems.

Definition 9414.51 (Compatibility-limit carrier)**.** *For a self-similar readout diagram $\mathcal{D}$, the compatibility-limit carrier is*

$$T_{\mathcal{D}} := \left\{ (r_i)_{i\in I} \in \prod_{i\in I} R_i : \begin{array}{c} C_i(r_i) \text{ for every displayed } i, \\ \rho_{ji}(r_j) \sim_i r_i \text{ for every displayed edge } j \to i \end{array} \right\}.$$

Its public packet includes the coordinate projections $\pi_i : T_{\mathcal{D}} \to R_i$, the induced ledger $\Lambda_{\mathcal{D}}$, and the audit row $A_{\mathcal{D}}$ recording all readout, projection, classifier, constraint, certificate, and gap compatibilities. Thus $T_{\mathcal{D}}$ is a ledgered packet carrier, not a bare product set.

Definition 9414.52 (Compatibility-limit operation)**.** *Assume the displayed self-similar operations preserve the compatibility-limit carrier: whenever $(r_i)_{i\in I} \in T_{\mathcal{D}}$, the family $(\alpha_i(r_i))_{i\in I}$ again lies in $T_{\mathcal{D}}$. The compatibility-limit operation is the packet operation*

$$\Omega_{\mathcal{D}} : T_{\mathcal{D}} \to T_{\mathcal{D}}, \qquad \Omega_{\mathcal{D}}((r_i)_i) := (\alpha_i(r_i))_i.$$

It is the common operation only at the level of the full packet carrying readout, classifier, constraint, ledger, certificate, and audit rows.

Theorem 9414.53 (Self-similar operations are coordinate projections)**.** *For every displayed $i \in I$,*

$$\pi_i \circ \Omega_{\mathcal{D}} \sim_i \alpha_i \circ \pi_i.$$

Proof. Let $r = (r_j)_{j\in I} \in T_{\mathcal{D}}$. By definition of $\Omega_{\mathcal{D}}$,

$$\Omega_{\mathcal{D}}(r) = (\alpha_j(r_j))_{j\in I}.$$

Applying the ith coordinate projection gives

$$\pi_i(\Omega_{\mathcal{D}}(r)) = \alpha_i(r_i).$$

Since $\pi_i(r) = r_i$, the right-hand side is also

$$(\alpha_i \circ \pi_i)(r).$$

The two readouts therefore agree at classifier resolution $\sim_i$. □

Corollary 9414.54 (Binary observation as projected operation)**.** *Suppose the diagram contains a binary observation row $R_B = \{B_0, B_1\}$ with projection $\pi_B : T_{\mathcal{D}} \to R_B$. If $b_0 \in T_{\mathcal{D}}$ satisfies $\pi_B(b_0) = B_0$ and*

$$\pi_B(\Omega_{\mathcal{D}}(b_0)) = B_1,$$

then B_1 is the binary-layer readout of the full packet operation at b_0. The informal notation $B_1 = \Omega_{\mathcal{D}}(B_0)$ abbreviates this projected statement; the typed packet statement applies $\Omega_{\mathcal{D}}$ to b_0 and reads the result through π_B.

Proof. The operation $\Omega_{\mathcal{D}}$ has domain $T_{\mathcal{D}}$, while B_0 and B_1 are records in the binary observation carrier. Therefore the operation cannot be applied to B_0 as a bare record. The displayed source point is b_0, and the binary observation after one packet operation is exactly $\pi_B(\Omega_{\mathcal{D}}(b_0))$. The assumed equality identifies that readout with B_1. □

Principle 9414.55 (Packet-level reading of common self-similarity)**.** *Prime-coordinate generation, Zeckendorf carry normalization, window constraint continuation, analytic readout, and recovered-zeta carrier formation may be compared as projections of one compatibility-limit operation only after their carrier, classifier, projection, constraint, ledger, certificate, and audit rows are all displayed. Without those rows, the comparison is merely an analogy between operations and not a reality-constrained packet claim.*

Definition 9414.56 (Constraint geometry surface)**.** *Let*

$$\mathcal{L} = (R, \sim_R, P, \Lambda_R, \mathsf{NameCert}_{\mathcal{L}})$$

be a readout layer over X, and let $C : R \to \mathsf{Prop}$ be a displayed constraint row on the record carrier. The visible constraint geometry is

$$|\mathcal{L}, C|_{\mathrm{vis}} := \sum_{r:R} C(r).$$

The source constraint geometry is

$$|\mathcal{L}, C|_{\mathrm{src}} := \sum_{x:X} C(P(x)).$$

The first surface records which visible rows satisfy the constraint. The second surface records which source witnesses are accepted by the constraint after readout. The two surfaces are not interchangeable unless the fiber and classifier ledgers identify the relevant source witnesses at the displayed resolution.

Theorem 9414.57 (Constraints generate auditable geometry)**.** *For a readout layer $\mathcal{L}$ and displayed constraint row C, every element of $|\mathcal{L}, C|_{\mathrm{src}}$ carries both a source witness and a constraint witness. Consequently the geometry generated by C is auditable precisely through the readout, classifier, fiber, and ledger rows of $\mathcal{L}$.*

Proof. An element of $|\mathcal{L}, C|_{\mathrm{src}}$ is a pair (x, γ) with $x : X$ and $\gamma : C(P(x))$. The first component is the source witness. The second component records that the visible readout of that source row lies on the constraint surface. The readout row P, the classifier $\sim_R$, the fiber family of Definition 9414.41, and the ledger Λ_R therefore determine the public audit route for the displayed geometric claim. □

Principle 9414.58 (Geometry as constraint–fiber–section structure)**.** *Within this packet, geometry is not a primitive point-set carrier. A point is a source witness accepted by a displayed constraint; a curve or surface is a stable family of such witnesses under one or more readout rows; a boundary is a classifier or constraint-change row; and an unresolved infinitary direction is the fiber family left behind a visible record. Section-level geometric meaning is licensed only when the corresponding section binding certificate is displayed.*

Remark 9414.59 (Recovered-carrier geometry)**.** *The same reading applies to recovered analytic carriers. A zero ledger, a fixed-half row, a prime-channel unit row, or a Laplace-bias refusal is geometric only after it is displayed as a constraint on a named readout carrier together with its provenance, fiber, transport, and gap rows. The geometric image does not discharge the analytic incompatibility required to exclude a non-fixed zero; it only states the audited surface on which that obligation lives.*

Definition 9414.60 (Invariant observation descent)**.** *Suppose a displayed action of a symmetry carrier G on X is available. An observation interface $O : X \to Y$ is invariant under the action when*

$$O(g \cdot x) \sim_Y O(x)$$

for every displayed action row $g \cdot x$. The orbit projection is written

$$\pi : X \to Q.$$

Theorem 9414.61 (Invariant observation descends to the compressed surface)**.** *If O is invariant under the displayed G-action, then the observation read factors through the compressed surface: there is a unique displayed readout $\bar{O} : Q \to Y$ with*

$$O = \bar{O} \circ \pi.$$

Proof. Define $\bar{O}(\pi(x)) := O(x)$. If $\pi(x) = \pi(y)$, then y is related to x by a displayed action row. Invariance gives $O(y) \sim_Y O(x)$, so the definition is independent of the representative at the displayed classifier resolution. Since every row of Q is represented by some x, the readout is unique at that resolution. □

Definition 9414.62 (Anti-invariant label)**.** *Let $L : X \to \{0, 1\}$ be a label row and let τ be a displayed involution in the symmetry carrier. The label is anti-invariant along τ when*

$$L(\tau \cdot x) = 1 - L(x)$$

for every displayed source row x.

Theorem 9414.63 (Anti-invariant labels do not descend)**.** *If L is anti-invariant along a displayed involution τ, then there is no quotient-level label $\bar{L} : Q \to \{0, 1\}$ satisfying*

$$L = \bar{L} \circ \pi$$

on the displayed action surface.

Proof. Assume such a $\bar{L}$ exists. Since τ is an action row, $\pi(\tau \cdot x) = \pi(x)$. Hence

$$L(\tau \cdot x) = \bar{L}(\pi(\tau \cdot x)) = \bar{L}(\pi(x)) = L(x).$$

Anti-invariance gives $L(\tau \cdot x) = 1 - L(x)$. The two equations contradict the binary label classifier. Therefore no such quotient-level label exists. □

Definition 9414.64 (Section binding certificate)**.** *A* section binding certificate *for $\pi : X \to Q$ and $\sigma : Q \to X$ is a finite certificate row*

$$\mathsf{SectionBindingCert}(Q, X, \pi, \sigma)$$

with source, authorization, action-response, witness, boundary, stability, and forbidden-claim ledgers, and with

$$\pi \circ \sigma = \mathrm{id}_Q.$$

It records that σ is not canonical output of the compressed surface Q alone; it is supplied by licensed action, authorization, binding, or certificate data.

Definition 9414.65 (Section-relative meaning)**.** *Given a label row $L : X \to \{0,1\}$ and a section binding certificate for σ, the section-relative meaning readout is*

$$L_\sigma : Q \to \{0,1\}, \qquad L_\sigma(q) := L(\sigma(q)).$$

Theorem 9414.66 (Sections license fiber-level meaning)**.** *If $\mathsf{SectionBindingCert}(Q, X, \pi, \sigma)$ is displayed and L is a licensed label row on X, then $L_\sigma = L \circ \sigma$ is a licensed readout on Q.*

Proof. The certificate makes σ an admitted map from the compressed surface back to a source representative. The label certificate makes L an admitted map on the source side. Their composite is therefore an admitted readout, and its source, authorization, witness, boundary, stability, and gap ledgers are obtained by composing the two certificate rows. □

Definition 9414.67 (Matched-twin observation protocol)**.** *A matched-twin observation protocol consists of displayed pairs (x_j^0, x_j^1) such that*

$$\begin{aligned} x_j^1 &= \tau \cdot x_j^0, \\ O(x_j^0) &\sim_Y O(x_j^1), \\ L(x_j^1) &= 1 - L(x_j^0). \end{aligned}$$

A decision rule is quotient-measurable when it factors through the compressed observation surface and therefore reads the same value on $O(x_j^0)$ and $O(x_j^1)$.

Theorem 9414.68 (Matched-twin null)**.** *In a balanced matched-twin observation protocol, every quotient-measurable binary decision rule has accuracy $1/2$.*

Proof. For each displayed pair, quotient-measurability gives the same decision on x_j^0 and x_j^1, because their observation rows have the same compressed value. The labels are opposite. Hence the decision is correct on exactly one member of the pair. Balanced exposure of the two members therefore gives pairwise accuracy $1/2$, and averaging over displayed pairs preserves that value. □

Corollary 9414.69 (Above-null performance requires an exposed extra row)**.** *If a classifier exceeds the matched-twin null boundary, then the packet must expose at least one of the following: leakage outside the compressed surface, licensed section access, or a failure of the announced protocol.*

Proof. Definition 9414.68 proves the null boundary for all quotient-measurable decision rules. Exceeding the boundary therefore means that the decision rule is not purely quotient-measurable on the announced protocol, or that the protocol itself did not satisfy the matched-twin rows. □

Definition 9414.70 (Probe selection and outcome row)**.** *For a focused history h_t, a probe policy assigns an admissible interface family to each action row:*

$$\Pi(h_t, a_t) = I_{t+1}.$$

The outcome is then generated by an outcome kernel or weight row

$$K_{I_{t+1}}(x, dr),$$

and the focused record advances by

$$h_{t+1} = \mathsf{Cont}(h_t, r_t).$$

Theorem 9414.71 (Action selects an interface, not an outcome)**.** *If an action row enters only through $\Pi(h_t, a_t) = I_{t+1}$ and outcomes are generated by $K_{I_{t+1}}(x, dr)$, then the action selects the next probe interface and does not by itself determine the outcome unless the kernel is degenerate on the displayed source row.*

Proof. The action variable appears in the policy row selecting I_{t+1}. The outcome variable appears in the kernel row attached to the selected interface and source witness. Unless that kernel has a single displayed outcome, the selected interface leaves multiple admissible outcome rows. Therefore the action does not directly specify the resulting record. □

Principle 9414.72 (Bridge boundary for probability and collapse language)**.** *Probability rows in this packet are normalized interface weights, not bare randomness and not conserved physical charge. A quantum-facing bridge may read a Hilbert-state projection family as an outcome-weight interface, but the BEDC core row remains the ledger update*

$$h_{t+1} = \mathsf{Cont}(h_t, r_t).$$

Any stronger claim about physical collapse, preferred basis selection, or consciousness as an outcome chooser is a bridge or gap claim, not an object-layer theorem of the packet.

Theorem 9414.73 (Layered reality decomposition)**.** *If $O : X \to Y$ is not injective at the displayed classifier resolution, then the source layer and interface layer are not equivalent as public records. If an anti-invariant label does not descend to Q, then the compressed layer does not determine section-relative meaning. If no ledger is displayed, then section-relative meaning is not a public auditable reality-constrained claim.*

Proof. Non-injectivity gives source rows x, x' with different source classifiers but the same interface readout; the interface record therefore erases a source distinction. Anti-invariant no-descent is Definition 9414.63; it says that the compressed surface cannot determine the label. Finally, without a ledger, the source, authorization, stability, and boundary data for a section readout are not auditable, so the readout cannot become a public reality-constrained claim in the sense of Definition 9414.8. □

Theorem 9414.74 (Interface–constraint–section synthesis)**.** *For an interface–constraint–section packet, invariant observations descend to compressed surfaces, anti-invariant labels do not descend, licensed sections recover fiber-level meaning, constraint rows restrict admissible readouts, focused time is append-only* Cont *growth, and actions select probe interfaces rather than outcomes.*

Proof. Invariant descent is Definition 9414.61. No-descent for anti-invariant labels is Definition 9414.63. Section-relative meaning is licensed by Definition 9414.66. Constraint restriction is Definition 9414.49. Focused time is the update row $h_{t+1} = \mathsf{Cont}(h_t, r_t)$ in Definition 9414.70. The interface role of action is Definition 9414.71. These clauses exhaust the packet-level synthesis and add no kernel primitive. □

Remark 9414.75 (Forbidden readings)**.** *The packet forbids six readings: source reality is not identical with the compressed interface surface; quotient-level observation is not already meaning; a section is not hidden choice without a certificate; a subject does not choose outcomes by fiat; normalized interface weight is not the same physical quantity as conserved charge; and indistinguishable metaphysical bridge explanations cannot be promoted to object-layer theorems.*

Concretized in §7582: Reality-constrained selector boundaries realize the interface–constraint–section refusal that selector rows must expose source records, classifier fit, physical constraint, failure surface, and gap ledger before a section read is public.

9414.5 Self-similar operations and exponential readouts

The compatibility-limit operation of Definition 9414.52 is a packet operation. It acts on the full carrier $T_{\mathcal{D}}$, whose rows include readouts, classifiers, constraints, ledgers, certificates, and audit surfaces. A scalar constant belongs to a later analytic readout layer. The two objects therefore have different types even when the scalar is a faithful shadow of the operation under a chosen projection.

Definition 9414.76 (Operation–readout type boundary)**.** *Let $\mathcal{D}$ be a self-similar readout diagram and let*

$$\Omega_{\mathcal{D}} : T_{\mathcal{D}} \to T_{\mathcal{D}}$$

be its compatibility-limit operation. A scalar analytic readout is a displayed map

$$\chi : T_{\mathcal{D}} \to A$$

into a scalar carrier A, together with the classifier, normalization, analytic structure, and ledger rows that make χ public. A scalar $c \in A$ has the type of a readout value, while $\Omega_{\mathcal{D}}$ has the type of a packet operation. Thus the expression

$$\Omega_{\mathcal{D}} = c$$

has no packet-level meaning. The meaningful claims are typed readout claims such as

$$\chi(\Omega_{\mathcal{D}}(b)) = c$$

or flow readout claims such as

$$\chi(\Omega_t(b)) = c(t).$$

Theorem 9414.77 (Constants are not packet operations)**.** *For any scalar readout $\chi : T_{\mathcal{D}} \to A$, no scalar constant $c \in A$ is identical with the packet operation $\Omega_{\mathcal{D}} : T_{\mathcal{D}} \to T_{\mathcal{D}}$. In particular, the analytic constant e can only occur as a scalar readout value or as the value of a scalar readout function; it is not the operation $\Omega_{\mathcal{D}}$.*

Proof. The operation $\Omega_{\mathcal{D}}$ has domain and codomain $T_{\mathcal{D}}$. A scalar constant c is an element of A. Unless a separate coding row embeds operations into scalar values and back with displayed inverse data, these types are distinct. The interface packet does not contain such a coding row. It contains only readout maps from the packet carrier to visible carriers. Therefore a scalar can be compared only with a scalar readout of the operation, not with the operation itself. □

Definition 9414.78 (Continuous packet flow)**.** *A continuous packet flow over $\mathcal{D}$ is a displayed family of packet operations*

$$(\Omega_t)_{t\in\mathbb{R}}, \qquad \Omega_t : T_{\mathcal{D}} \to T_{\mathcal{D}},$$

with ledgered rows

$$\Omega_0 = \mathrm{id}_{T_{\mathcal{D}}}, \qquad \Omega_{t+u} = \Omega_t \circ \Omega_u.$$

The flow is a packet object only when the same classifier, constraint, ledger, certificate, and audit rows remain visible along the parameter.

Definition 9414.79 (Normalized self-change readout)**.** *Let $(\Omega_t)_{t\in\mathbb{R}}$ be a continuous packet flow and let $\chi : T_{\mathcal{D}} \to \mathbb{R}$ be a scalar analytic readout. For $b \in T_{\mathcal{D}}$, write*

$$y_b(t) := \chi(\Omega_t(b)).$$

The readout is normalized self-changing along b when its analytic ledger contains

$$y_b(0) = 1, \qquad \frac{d}{dt} y_b(t) = y_b(t).$$

Theorem 9414.80 (Exponential readout shadow)**.** *If y_b is a normalized self-change readout, then*

$$y_b(t) = \exp(t).$$

Consequently,

$$y_b(1) = e.$$

Thus e is the unit-time value of this scalar readout, not the compatibility-limit operation itself.

Proof. Define

$$z(t) := \exp(-t)y_b(t).$$

The displayed derivative row gives

$$\begin{aligned} z'(t) &= -\exp(-t)y_b(t) + \exp(-t)y_b'(t) \\ &= \exp(-t)(y_b'(t) - y_b(t)) \\ &= 0. \end{aligned}$$

Hence z is constant in the analytic readout layer. Since

$$z(0) = \exp(0)y_b(0) = 1,$$

we have $z(t) = 1$ and therefore $y_b(t) = \exp(t)$. At $t = 1$ this gives $y_b(1) = e$. □

Definition 9414.81 (Discrete scale readout)**.** *A discrete scale readout of the packet operation is a scalar sequence*

$$x_n := \chi(\Omega_{\mathcal{D}}^n(b))$$

whose displayed ledger contains

$$x_{n+1} = \lambda x_n.$$

The scalar λ is a readout scale for that coordinate, not a replacement for $\Omega_{\mathcal{D}}$.

Proposition 9414.82 (Analytic interpolation of a scale readout)**.** *If $x_{n+1} = \lambda x_n$ and the analytic readout layer supplies a chosen branch of $\log \lambda$, then the scalar function*

$$x(t) := \exp(t \log \lambda)x_0$$

interpolates the displayed discrete readout at integer times:

$$x(n) = x_n.$$

Proof. The recurrence gives $x_n = \lambda^n x_0$. For each integer n, the chosen logarithm branch supplies

$$\exp(n \log \lambda) = \lambda^n.$$

Therefore

$$x(n) = \exp(n \log \lambda)x_0 = \lambda^n x_0 = x_n.$$

□

Principle 9414.83 (Constants classify readout modes)**.** *The constants attached to self-similar phenomena classify readout modes. The normalized continuous self-change readout gives e. A Fibonacci second-order recursive readout gives the golden scale φ as in Definition 9347.14. A prime Euler-product readout gives the local channel p^{-s} as in Definition 9252.3. These constants are coordinate shadows of packet-level generation; none is the packet operation itself.*

Definition 9414.84 (Prime exponential kernel readout)**.** *Let a prime-occupancy row be a finite-support function*

$$k : \mathsf{Prime}^{\uparrow} \to \mathbb{N},$$

with associated integer packet

$$n(k) = \prod_{p \in \mathsf{Prime}^{\uparrow}} p^{k_p}.$$

The logarithmic energy readout is

$$E(k) := \log n(k) = \sum_{p} k_p \log p.$$

For a complex parameter s, the exponential kernel readout is

$$W_s(k) := \exp(-sE(k)).$$

Proposition 9414.85 (Exponential kernel gives Dirichlet weight)**.** *For every displayed prime-occupancy row k,*

$$W_s(k) = n(k)^{-s}.$$

Proof. By definition, $E(k) = \log n(k)$. Therefore

$$W_s(k) = \exp(-s \log n(k)) = n(k)^{-s}.$$

□

Principle 9414.86 (Zeta readout boundary)**.** *The zeta-facing expression $n^{-s} = \exp(-s \log n)$ is the analytic readout by which prime multiplicative occupancy enters the Dirichlet and Euler-product surfaces. It does not identify ζ, e, or any Euler factor with $\Omega_{\mathcal{D}}$. It says that when the zeta carrier is a coordinate of the self-similar diagram, the zeta coordinate reads the packet through an exponential kernel, Euler gluing, continuation, functional-equation, pole, zero-ledger, and transport surface.*

Definition 9414.87 (Holographic readout analogy)**.** *A holographic readout analogy for a self-similar diagram $\mathcal{D}$ is a records-side dictionary between a boundary packet*

$$\partial \mathbb{G}_{\mathcal{D}}$$

and a bulk packet

$$\mathbb{G}_{\mathcal{D}}.$$

The boundary packet consists of finite visible readouts such as finite windows, prime windows, prime fingerprints, and channel codes. The bulk packet is the compatibility-limit carrier $T_{\mathcal{D}}$ together with the operation $\Omega_{\mathcal{D}}$, its projection rows, constraint rows, residual fibers, continuation rows, ledgers, certificates, and audit surface. The analogy is only this records-side dictionary. It is not a claim that the packet is a physical anti-de Sitter spacetime, a conformal field theory, a gravitational theory, or a string-theoretic duality.

Principle 9414.88 (Boundary–bulk dictionary)**.** *The holographic readout analogy is governed by the following dictionary: finite boundary windows correspond to coordinate projections of $T_{\mathcal{D}}$; extension depth corresponds to the ledgered passage from finite readout to compatibility-limit carrier; window defects correspond to obstruction rows in the bulk packet; unit-amplitude boundary channels correspond to fixed-surface constraints in the bulk packet; and non-unit radial factors in exponential channels correspond to off-fixed displacement rows. Each correspondence is a typed readout or ledger relation, not an equality of physical theories.*

Definition 9414.89 (Zeta radial and phase coordinates)**.** *In the zeta information-channel coordinate*

$$s = \frac{1}{2} + \delta + i\tau,$$

the centered prime channel of Definition 9252.3 is

$$U_p(\delta, \tau) = p^{-\delta} e^{-i\tau \log p}.$$

The factor $e^{-i\tau \log p}$ is the phase-clock readout, while $p^{-\delta} = e^{-\delta \log p}$ is the radial exponential readout. The fixed surface is the row $\delta = 0$, equivalently the unit-amplitude row

$$|U_p(\delta, \tau)| = 1$$

for every displayed prime channel.

Proposition 9414.90 (Off-fixed packets carry radial exponent)**.** *If a zeta information-channel packet has $\delta \neq 0$, then every displayed prime row $p > 1$ carries the non-unit radial factor*

$$e^{-\delta \log p}.$$

If $\delta = 0$, the same channel is purely phase:

$$U_p(0, \tau) = e^{-i\tau \log p}.$$

Proof. By Definition 9414.89,

$$U_p(\delta, \tau) = e^{-\delta \log p} e^{-i\tau \log p}.$$

When $\delta = 0$, the first factor is $\exp(0) = 1$, so the channel is the phase factor $e^{-i\tau \log p}$. When $\delta \neq 0$ and $p > 1$, the exponent $-\delta \log p$ is nonzero, so the displayed channel contains a non-unit radial exponential factor. □

Definition 9414.91 (Zero-ledger exponential rigidity target)**.** *The zero-ledger exponential rigidity target is the statement that a zeta zero-ledger packet in the information band admits only pure phase prime channels:*

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \forall p \in \mathsf{Prime}^{\uparrow},\ |U_p(\delta, \tau)| = 1.$$

Equivalently, in the boundary–bulk dictionary, a zero-ledger boundary row cannot extend to a bulk packet with a nonzero radial exponent.

Theorem 9414.92 (Exponential rigidity is the fixed-half obligation)**.** *The zero-ledger exponential rigidity target is equivalent to the fixed-half form of the RH information-channel target in Definition 9252.14.*

Proof. The rigidity target is exactly the third displayed form in Definition 9252.14. That theorem proves it equivalent to

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \Longrightarrow \delta = 0,$$

which is the fixed-half form after substituting $s = 1/2 + \delta + i\tau$. □

Principle 9414.93 (Holographic proof boundary)**.** *The boundary–bulk analogy identifies the shape of the remaining obligation but does not discharge it. To prove the zeta fixed-half target, one must derive zero-ledger exponential rigidity from the displayed Dirichlet row, Euler row, analytic-continuation row, Archimedean completion row, functional-equation row, zero ledger, and transport ledger. Stating that an off-fixed packet has a radial factor is only the readout diagnosis; the missing theorem is the incompatibility between that radial factor and the completed zero-ledger packet.*

Principle 9414.94 (Full-constraint self-similar closure)**.** *For a self-similar diagram $\mathcal{D}$, the operation $\Omega_{\mathcal{D}}$ generates only objects with a position inside the compatibility-limit carrier and its displayed projections. A legal object therefore has no tower-external position: it must be read as a coordinate, fiber, section, ledger row, certificate row, or audit row of some $x \in T_{\mathcal{D}}$. This closure principle is a discipline on admissibility. It does not say that every coordinate is fixed by every involution or that every zeta zero row satisfies the fixed-half condition.*

Definition 9414.95 (Tower-internal position)**.** *Let y be a visible record in a carrier R_i of the diagram. The record y has a tower-internal position when there exists $x \in T_{\mathcal{D}}$ such that*

$$y \sim_i \pi_i(x).$$

When the public use of y depends on more than the visible record, its tower-internal position also includes the source, projection, constraint, ledger, certificate, and audit rows through which this comparison is made.

Theorem 9414.96 (Legal records have tower-internal position)**.** *If a record is legal only by being generated, read out, constrained, and ledgered by the self-similar diagram $\mathcal{D}$, then it has a tower-internal position.*

Proof. By the stated legality condition, the record occurs in some displayed carrier R_i together with the source, constraint, projection, certificate, ledger, and audit rows that make it public. Those rows are part of a compatible packet $x \in T_{\mathcal{D}}$. The visible record is therefore classified by the ith projection $\pi_i(x)$, which is exactly the tower-internal position condition. □

Corollary 9414.97 (A legal RH counterexample would be tower-internal)**.** *If a counterexample to the fixed-half zeta target were a legal BEDC object, it would be a zeta or zero-ledger projection of some $x \in T_{\mathcal{D}}$. It could not be a tower-external object.*

Proof. Apply Definition 9414.96 to the zeta zero-ledger carrier and its critical-strip readout rows. □

Proposition 9414.98 (Closure does not imply fixedness)**.** *A closed self-similar carrier may contain non-fixed points.*

Proof. Let $X = \{a, b\}$ and let $\Omega : X \to X$ exchange the two elements:

$$\Omega(a) = b, \qquad \Omega(b) = a.$$

The carrier X is closed under Ω, and every element lies in an Ω-orbit. If $J = \Omega$, then

$$J(a) = b \neq a.$$

Thus closure under a self-similar operation does not imply membership in the equalizer of id and J. □

Definition 9414.99 (Off-fixed zero exclusion target)**.** *In zeta information-channel coordinates $s = 1/2 + \delta + i\tau$, the off-fixed zero exclusion target is*

$$\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau) \wedge \mathsf{InfoBand}(\delta) \wedge \mathsf{ApartNonFix}(\delta) \Longrightarrow \bot.$$

Its constructive strengthening is the local-global exclusion row

$$\mathsf{InfoBand}(\delta) \wedge \mathsf{ApartNonFix}(\delta) \Longrightarrow \mathsf{Apart}_0(\zeta(1/2 + \delta + i\tau)).$$

This target says that an off-fixed packet may be a tower-internal candidate, but it cannot be inhabited as a zero-ledger packet.

Theorem 9414.100 (Off-fixed exclusion gives fixed-half zero ledger)**.** *If the off-fixed zero exclusion target holds and the real-part readout is located, then every legal zero-ledger packet in the information band lies on the fixed-half surface.*

Proof. Let a legal zero-ledger packet have information coordinates (δ, τ) and assume $\mathsf{InfoBand}(\delta)$. To prove the fixed-half conclusion it suffices, by locatedness of the real-part readout, to rule out every positive apartness witness for δ. Assume such a witness, namely $\mathsf{ApartNonFix}(\delta)$. Together with the zero-ledger row $\mathsf{Cancel}_{\mathcal{P}}(\delta, \tau)$ and the information-band row, the off-fixed zero exclusion target gives a contradiction. Hence no positive off-fixed displacement remains, so $\delta = 0$ and the packet is on the fixed-half surface. □

Principle 9414.101 (Closure-plus-exclusion form of the RH route)**.** *The self-similar closure route has two logically separate parts. First, full-constraint closure places every legal object, including any proposed RH counterexample, inside the tower. Second, off-fixed zero exclusion rules out the tower-internal packets whose zeta readout is both zero-ledger and off-fixed. The first part supplies admissibility and location; the second part is the substantive analytic obstruction. The remaining proof obligation is therefore the incompatibility between off-fixed displacement and the completed zero-ledger carrier, not the absence of tower-external objects.*

Definition 9414.102 (Internal closure witness)**.** *For a claim A whose public use depends on a self-similar diagram $\mathcal{D}$, an internal closure witness is a packet*

$$\mathsf{Wit}_{\mathcal{D}}(A)$$

consisting of a tower point $x \in T_{\mathcal{D}}$, a displayed projection $\pi_i(x)$, a carrier certificate, a readback certificate, an invariant transport certificate, and an audit row showing how these data inhabit the claim surface for A. Such a witness is internal because its evidence is read through the tower; it is still a witness because the claim is not licensed by tower membership alone.

Theorem 9414.103 (Closure does not cancel witnesses)**.** *Full-constraint closure does not turn every legal claim into an automatic theorem. It only locates the objects that a legal claim may mention. A claim still requires a witness packet for its carrier, readback, invariant, exclusion, or audit obligations.*

Proof. By Definition 9414.96, a legal record has a tower-internal position. This supplies a source and projection surface, but it does not by itself prove any predicate over that surface. To prove a predicate one must also display the relevant classifier preservation, readback, invariant transport, exclusion, or audit row. These rows are exactly the components listed in Definition 9414.102. Therefore closure relocates the witness into the tower but does not remove the witness obligation. □

Definition 9414.104 (Object-layer closure and truth-total reflection)**.** *Object-layer closure is the claim that legal objects are generated, projected, and ledgered inside $T_{\mathcal{D}}$. Truth-total reflection would be a stronger claim: an internal truth classifier for all statements represented by the diagram. Formally, it would require a predicate assigning truth to every represented sentence together with an audit certificate that this assignment is correct for the whole represented fragment.*

Principle 9414.105 (Object-layer closure is not truth closure)**.** *The self-similar closure used in the RH route is object-layer closure. It is not truth-total reflection. It may say where legal zeta packets, zero ledgers, defects, and exclusion certificates live; it may not supply a universal internal truth predicate for all claims about those packets. This is the same stratification recorded*

in chapter 9461: the primitive loop may close, while truth-total reflection remains outside the internal object layer.

Corollary 9414.106 (Witness form of the closure route)**.** *The RH route through $\Omega_{\mathcal{D}}$ has the form*

tower location+zero-ledger soundness+off-fixed defect production+defect-zero incompatibility+located real readbac

The first term prevents tower-external escape. The remaining terms are internal witness obligations. In particular, the substantive certificate is still the exclusion of tower-internal off-fixed zero packets.

Proof. Tower location is supplied by Definition 9414.97. Off-fixed exclusion gives fixed-half zero ledger by Definition 9414.100. The definition of the off-fixed zero exclusion target names the needed defect-zero incompatibility form. Located real readback converts absence of positive off-fixed displacement into $\delta = 0$. These are separate witness rows, not consequences of closure alone. □

9414.6 Zero-ledger rigidity route

The closure route of Definition 9414.101 separates location from exclusion. Full-constraint closure places a legal zeta packet inside the self-similar tower. The fixed-half conclusion still requires a certificate that the tower-internal zero ledger cannot coexist with an off-fixed radial defect.

Principle 9414.107 (Finite induction and global coinduction)**.** *Finite readout layers may be closed by ordinary induction over displayed depths, windows, or recursive constructors. The full self-similar tower is not used as an ordinary well-founded chain with an absolute bottom. It is a closed compatibility carrier whose projections and residual fibers can restart the same tower structure. Global statements about the tower are therefore invariant statements: one proves that the relevant row is preserved by the boundary readouts, fiber readbacks, compatibility maps, and the self-similar operation. In the zeta direction, the invariant of interest is fixed-half zero-ledger rigidity.*

Definition 9414.108 (Self-similar zeta coordinate row)**.** *Let $\mathcal{D}$ be a self-similar diagram with zeta projection*

$$\pi_\zeta : T_{\mathcal{D}} \to R_\zeta.$$

A self-similar zeta coordinate row assigns to each tower point $x \in T_{\mathcal{D}}$ an information-channel readout

$$s_x = \pi_\zeta(x) = \frac{1}{2} + \delta_x + i\tau_x,$$

together with the transport rows connecting $\mathsf{ZetaZero}(s_x)$, $\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x)$, *and* $\mathsf{InfoBand}(\delta_x)$ *to the RH normal forms of Definition 9252.14.*

Definition 9414.109 (Zero-ledger atom)**.** *For a tower point $x \in T_{\mathcal{D}}$ with zeta coordinate s_x, a zero-ledger atom is the packet row*

$$\mathsf{ZeroAtom}_{\mathcal{D}}(x) := \mathsf{ZeroLedger}(\pi_\zeta(x)).$$

Equivalently, it is a zeta-value evaluation packet

$$(s_x, z_x, \mathsf{ZetaVal}(s_x, z_x), z_x\mathsf{hsame0})$$

together with the zero classifier, source ledger, transport rows, and audit rows that make the zero event public. Thus a zero atom is not a bare complex point. The phrase that a zeta zero is generated by the self-similar operation means generation at the zeta-zero ledger layer: there is a tower point whose zeta projection is admitted by the zero ledger.

Definition 9414.110 (Zeta zero-ledger payload)**.** *The zeta zero-ledger payload of a tower point x is the displayed packet of rows*

$$\begin{aligned}\mathsf{ZetaZeroPayload}_{\mathcal{D}}(x) := (s_x,\ z_x,\ \mathsf{ZetaVal}(s_x, z_x),\ z_x\mathsf{hsame0},\ \mathsf{InfoBand}(\delta_x),\\ \mathsf{CompletedZetaCarrier}(\pi_\zeta(x)),\ \mathsf{ZeroLedger}(\pi_\zeta(x)),\ \mathsf{Ledger}_\zeta(x)).\end{aligned}$$

The payload records the point coordinate only as one field of a larger carrier. Its public status depends on evaluation, classifier, completed carrier, ledger, and audit rows. Therefore the zero-ledger route never treats $\mathsf{ZetaZero}$ *as a subset of bare complex points.*

Definition 9414.111 (Coinductive zero-ledger invariant)**.** *A coinductive zero-ledger invariant for a self-similar diagram $\mathcal{D}$ is a predicate $\mathcal{R}$ on $T_{\mathcal{D}}$ such that for every tower point x:*

$$\mathcal{R}(x)$$

is stable under the zeta boundary readout, the residual fiber readback, the compatibility maps of the diagram, and the operation $\Omega_{\mathcal{D}}$. The invariant is RH-facing when it contains the zero-rigidity implication

$$\mathsf{ZeroAtom}_{\mathcal{D}}(x) \wedge \mathsf{InfoBand}(\delta_x) \Longrightarrow \delta_x = 0.$$

Principle 9414.112 (Atomization boundary for zeta zeros)**.** *Atomization at the zero-ledger layer means minimality with respect to the displayed zero classifier: the packet is a basic auditable zero event in the ledger. It does not assert that the analytic zero is simple, and it does not provide a finite listing of all non-trivial zeros. A fixed zeta value can be approximated through its evaluation rows, while exact zero-ledger admission is a constructive packet obligation that is not generally decided by finite approximation alone.*

Definition 9414.113 (Non-trivial zero atom and counterpacket)**.** *The non-trivial zero atoms of the self-similar zeta coordinate are*

$$\mathsf{NTZeroAtom}_{\mathcal{D}}(x) := \mathsf{ZeroAtom}_{\mathcal{D}}(x) \wedge \mathsf{InfoBand}(\delta_x).$$

The corresponding tower-internal RH counterpacket type is

$$\mathsf{CounterAtom}_{\mathsf{RH},\mathcal{D}} := \sum_{x:T_{\mathcal{D}}} \mathsf{ZeroAtom}_{\mathcal{D}}(x) \times \mathsf{InfoBand}(\delta_x) \times \mathsf{ApartNonFix}(\delta_x).$$

This is the constructive disproof shape of Definition 9239.8 read through the self-similar tower: a zero-ledger atom in the information band, together with a positive off-fixed witness.

Definition 9414.114 (Zero-ledger fiber geometry)**.** *The zero geometry carried by the self-similar zeta coordinate is the zero-ledger fiber*

$$\mathbb{Z}_{\zeta,\mathcal{D}} := \sum_{x:T_{\mathcal{D}}} \mathsf{ZeroAtom}_{\mathcal{D}}(x).$$

Its non-trivial part is

$$\mathbb{Z}^{\mathsf{nt}}_{\zeta,\mathcal{D}} := \sum_{x:T_{\mathcal{D}}} \mathsf{ZeroAtom}_{\mathcal{D}}(x) \times \mathsf{InfoBand}(\delta_x).$$

The RH target is the assertion that this non-trivial zero-ledger fiber projects into the fixed-half surface, equivalently into the fixed locus of the information-channel mirror.

Definition 9414.115 (Self-similar zero rigidity)**.** *For such a zeta coordinate row, the self-similar zero-rigidity statement is*

$$\mathsf{ZeroRigidity}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}},\ \mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \wedge \mathsf{InfoBand}(\delta_x) \Longrightarrow \delta_x = 0.$$

Equivalently, by Definition 9252.14, every tower-internal zero-ledger atom in the information band has unit amplitude on all displayed prime channels. This is a rigidity condition on zero-ledger atoms, not a procedure for listing the atoms.

Definition 9414.116 (Self-similar off-fixed exclusion)**.** *The off-fixed exclusion form is*

$$\mathsf{NoOffFixedZero}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}},\ \mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \wedge \mathsf{InfoBand}(\delta_x) \wedge \mathsf{ApartNonFix}(\delta_x) \Longrightarrow \bot.$$

It is the tower-internal version of the exclusion target in Definition 9414.99.

Theorem 9414.117 (Off-fixed exclusion gives zero rigidity)**.** *If* $\mathsf{NoOffFixedZero}_{\mathcal{D}}$ *holds and the real-part readout is located with tight apartness, then*

$$\mathsf{ZeroRigidity}_{\mathcal{D}}$$

holds.

Proof. Fix $x \in T_{\mathcal{D}}$ and assume

$$\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \wedge \mathsf{InfoBand}(\delta_x).$$

To prove $\delta_x = 0$, the located real-part readout reduces the task to ruling out every positive apartness witness for δ_x. If $\mathsf{ApartNonFix}(\delta_x)$ were present, the three displayed rows would inhabit the premise of $\mathsf{NoOffFixedZero}_{\mathcal{D}}$ and hence give $\bot$. Therefore no positive off-fixed displacement remains, and tightness of the located readout gives $\delta_x = 0$. □

Definition 9414.118 (Prime-window radial defect)**.** *For a finite prime window* $W \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}$*, the prime-window radial defect of a tower point* x *is a displayed defect row*

$$\mathsf{RadDef}_W(x)$$

whose amplitude ledger reads

$$|\mathsf{RadDef}_W(x)| = |\delta_x| \sum_{p \in W} \log p.$$

For $W \neq \varnothing$*, a positive value of this row records a visible finite-prime trace of off-fixed radial displacement.*

Definition 9414.119 (Defect-zero incompatibility certificate)**.** *The defect-zero incompatibility certificate for* $\mathcal{D}$ *is the conditional row*

$$\mathsf{DefectZeroIncompat}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}}, \mathsf{CompletedZetaCarrier}(\pi_\zeta(x)) \wedge \sum_{W \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}} (W \neq \varnothing) \times (|\mathsf{RadDef}_W(x)| > 0)$$
$$\Longrightarrow \mathsf{Apart}_0(\zeta(s_x)).$$

Thus a completed zeta carrier with a positive finite-window radial defect is separated from the zero row.

Definition 9414.120 (Off-fixed-to-defect certificate)**.** *The off-fixed-to-defect certificate is*

$$\mathsf{OffFixedToDefect}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}}, \mathsf{InfoBand}(\delta_x) \wedge \mathsf{ApartNonFix}(\delta_x) \\ \implies \sum_{W \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}} (W \neq \varnothing) \times (|\mathsf{RadDef}_W(x)| > 0).$$

It supplies the finite prime window in which a nonzero radial displacement becomes visible to the ledger.

Definition 9414.121 (Zero-ledger soundness row)**.** *The zero-ledger soundness row is*

$$\mathsf{ZeroLedgerSound}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}},\ \mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \implies \zeta(s_x)\mathsf{hsame0}.$$

It says that the cancellation row used by the information-channel normal form is sound for the completed zeta value carried by the same tower point.

Principle 9414.122 (Zero-ledger soundness and completeness boundary)**.** *The exclusion route requires zero-ledger soundness:*

$$\mathsf{ZeroLedger}(\pi_\zeta(x)) \implies \zeta(s_x)\mathsf{hsame0}.$$

The converse direction,

$$\zeta(s_x)\mathsf{hsame0} \implies \mathsf{ZeroLedger}(\pi_\zeta(x)),$$

is a separate zero-ledger completeness target for the completed carrier. It is not needed to refute an off-fixed zero-ledger atom. Soundness is the row that turns $\mathsf{Apart}_0(\zeta(s_x))$ *into a contradiction with zero-ledger admission.*

Definition 9414.123 (Same-source zeta defect dictionary)**.** *A same-source zeta defect dictionary for* $\mathcal{D}$ *is a certificate that, for each* $x \in T_{\mathcal{D}}$*, the following rows are projections of the same tower point and are transported by the same zeta classifier:*

$$\mathsf{ZeroAtom}_{\mathcal{D}}(x), \qquad \mathsf{CompletedZetaCarrier}(\pi_\zeta(x)), \\ \mathsf{RadDef}_W(x), \qquad \mathsf{ArchimedeanRow}(\pi_\zeta(x)), \qquad \mathsf{FunctionalEquationRow}(\pi_\zeta(x)).$$

This dictionary prevents a boundary defect and a bulk zero row from being read as unrelated objects with the same printed coordinate. The finite-prime boundary, Archimedean row, functional-equation row, and zero-ledger row must all be fields of the same completed zeta payload.

Definition 9414.124 (Projection-recovery defect obligation)**.** *The projection-recovery defect obligation states that a positive finite-prime radial defect obtained from the information-channel boundary survives as a defect row of the completed zeta carrier:*

$$\mathsf{DefectRecovery}_{\mathcal{D}} := \forall x \in T_{\mathcal{D}}, \sum_{W \subset_{\mathrm{fin}} \mathsf{Prime}^{\uparrow}} (W \neq \varnothing) \times (|\mathsf{RadDef}_W(x)| > 0) \\ \implies \mathsf{CompletedDefectRow}_{\mathcal{D}}(x).$$

It is the carrier-level version of saying that analytic continuation, Archimedean completion, functional equation, and branch transport do not erase the source of the displayed radial defect.

Theorem 9414.125 (Defect incompatibility gives off-fixed exclusion)**.** *Assume the completed-carrier row is present for every zero-ledger tower point. If*

$$\mathsf{DefectZeroIncompat}_{\mathcal{D}}, \qquad \mathsf{OffFixedToDefect}_{\mathcal{D}}, \qquad \mathsf{ZeroLedgerSound}_{\mathcal{D}}$$

hold, then

$$\mathsf{NoOffFixedZero}_{\mathcal{D}}$$

holds.

Proof. Fix $x \in T_{\mathcal{D}}$ and assume

$$\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \wedge \mathsf{InfoBand}(\delta_x) \wedge \mathsf{ApartNonFix}(\delta_x).$$

The last two rows, through $\mathsf{OffFixedToDefect}_{\mathcal{D}}$, give a finite nonempty prime window W with

$$|\mathsf{RadDef}_W(x)| > 0.$$

Together with the completed-carrier row, this is the premise of $\mathsf{DefectZeroIncompat}_{\mathcal{D}}$, hence

$$\mathsf{Apart}_0(\zeta(s_x)).$$

The zero-ledger soundness row applied to $\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x)$ gives

$$\zeta(s_x)\mathsf{hsame}0.$$

The apartness row and the zero row cannot both be inhabited. Therefore the assumed off-fixed zero-ledger atom yields $\perp$, uniformly in x. □

Corollary 9414.126 (Defect route gives zero rigidity)**.** *Assume the completed-carrier row is present for every zero-ledger tower point. If*

$$\mathsf{DefectZeroIncompat}_{\mathcal{D}}, \qquad \mathsf{OffFixedToDefect}_{\mathcal{D}}, \qquad \mathsf{ZeroLedgerSound}_{\mathcal{D}}$$

hold and the real-part readout is located with tight apartness, then

$$\mathsf{ZeroRigidity}_{\mathcal{D}}$$

holds.

Proof. By Definition 9414.125, the three certificate rows give $\mathsf{NoOffFixedZero}_{\mathcal{D}}$. Then Definition 9414.117 gives $\mathsf{ZeroRigidity}_{\mathcal{D}}$. □

Theorem 9414.127 (Closure and zero rigidity give the BEDC RH target)**.** *Assume every legal non-trivial zero-ledger atom has a tower-internal position as in Definition 9414.97, and assume the zeta coordinate row transports the BEDC zero and strip predicates to* $\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x)$ *and* $\mathsf{InfoBand}(\delta_x)$*. If*

$$\mathsf{ZeroRigidity}_{\mathcal{D}}$$

holds, then the BEDC RH predicate of Definition 9239.7 holds.

Proof. Let s be the zeta coordinate of a legal non-trivial zero-ledger atom. By tower-internal location, there is $x \in T_{\mathcal{D}}$ whose zeta projection carries the same zero and critical-strip data. The transport rows give

$$\mathsf{Cancel}_{\mathcal{P}}(\delta_x, \tau_x) \wedge \mathsf{InfoBand}(\delta_x).$$

Zero rigidity gives $\delta_x = 0$. By Definition 9252.14, this is the critical-line form of the same packet. Transporting back through the zeta classifier gives $\mathsf{OnCritLine}(s)$, which is the BEDC RH conclusion. □

Theorem 9414.128 (Coinductive defect route gives the BEDC RH target)**.** *Assume every legal non-trivial zero-ledger atom has a tower-internal position in the full self-similar carrier. Assume also that the zeta coordinate row is a same-source row: its zero ledger, completed zeta carrier, finite prime-window defects, and information-channel readouts are all read from the same tower point and transported through the same classifier. If*

$$\mathsf{DefectZeroIncompat}_{\mathcal{D}}, \qquad \mathsf{OffFixedToDefect}_{\mathcal{D}}, \qquad \mathsf{ZeroLedgerSound}_{\mathcal{D}}$$

hold, and the real-part readout is located with tight apartness, then the BEDC RH predicate of Definition 9239.7 holds.

Proof. Let s be the zeta coordinate of a legal non-trivial zero-ledger atom. Tower closure supplies $x \in T_{\mathcal{D}}$ carrying the same zero and information-band rows. If this atom were off-fixed, then $\mathsf{OffFixedToDefect}_{\mathcal{D}}$ would supply a finite nonempty prime window whose radial defect is positive. The same-source condition places that defect and the zero-ledger row in the same completed zeta carrier. By Definition 9414.125, the three certificate rows exclude the off-fixed zero atom. Hence Definition 9414.126 gives $\mathsf{ZeroRigidity}_{\mathcal{D}}$. Applying Definition 9414.127 yields the BEDC RH target. □

Definition 9414.129 (Constructive RH witness extracted from exclusion)**.** *The witness extracted from the zero-ledger exclusion route is the function schema*

$$\Phi_{\mathcal{D}} : \mathbb{Z}^{\mathsf{nt}}_{\zeta,\mathcal{D}} \to \mathrm{Fix}_{1/2},$$

where $\mathrm{Fix}_{1/2}$ *denotes the fixed-half output surface of Definition 9252.14. On an input* $(x, \mathsf{ZeroAtom}_{\mathcal{D}}(x), \mathsf{InfoBand}(\delta_x$ *the function first excludes* $\mathsf{ApartNonFix}(\delta_x)$ *by the defect-incompatibility route and then uses located real readback to return the witness* $\delta_x = 0$.

Principle 9414.130 (Exact boundary of the self-similar route)**.** *Self-similar closure contributes tower location. It excludes an external place for a legal RH counterexample but does not by itself exclude an off-fixed tower-internal zero-ledger atom. The certificate that performs the exclusion is the defect-zero incompatibility row together with off-fixed-to-defect, zero-ledger soundness, completed-carrier, and located real readback data. In witness form, the route is precisely the chain recorded in Definition 9414.106.*

Concretized in §9142: Zero-ledger rigidity is realized as finite visible-route, replay, provenance, and no-hidden-ledger NameCert data.

9414.7 Layered projection theory

Remark 9414.131 (Paper-side theory status)**.** *This section records the paper-side form of layered projection theory. It is tracked against the Lean readout-system and product-normal-form carriers. Its role is to make explicit the theory carried implicitly by the interface–constraint–section packet: every public layer is a readout surface over a source carrier, every loss of visible distinction leaves a fiber family, every legitimate refinement must display its readback map, and every numerical half-value claim must arise from a witnessed involutive symmetry rather than from an unledgered statistical convention.*

Lean4 (definition) `BEDC.RealityConstrained.ReadoutSystem`
Lean4 (definition) `BEDC.RealityConstrained.AxisProductNormalForm`

Definition 9414.132 (Layered projection datum)**.** *A* layered projection datum *over a displayed source carrier X is a finite family*

$$\mathfrak{P} = (\mathcal{L}_i, \rho_{i+1,i}, \Lambda_{i+1,i})_{i<n}$$

where each $\mathcal{L}_i$ is a readout layer in the sense of Definition 9414.40, each $\rho_{i+1,i} : R_{i+1} \to R_i$ is a displayed readback map witnessing that $\mathcal{L}_{i+1}$ refines $\mathcal{L}_i$, and each $\Lambda_{i+1,i}$ records classifier transport, source provenance, refinement exactness, and the residual gap surface.

Definition 9414.133 (Inherited predicate)**.** *Let $\mathcal{L}_{i+1}$ refine $\mathcal{L}_i$ through $\rho_{i+1,i} : R_{i+1} \to R_i$. A predicate $A_i : R_i \to \mathsf{Prop}$ on the i-th layer induces the predicate*

$$A_{i+1} := A_i \circ \rho_{i+1,i}$$

on the refined layer. The induction is legitimate only with the displayed readback map and the classifier-respect row for A_i.

Theorem 9414.134 (Predicate inheritance)**.** *Suppose $\mathcal{L}_{i+1}$ refines $\mathcal{L}_i$ and A_i respects $\sim_i$. If $A_i(P_i(x))$ holds for a source row $x : X$, then the inherited predicate satisfies*

$$A_{i+1}(P_{i+1}(x)).$$

Proof. By definition,

$$A_{i+1}(P_{i+1}(x)) = A_i(\rho_{i+1,i}(P_{i+1}(x))).$$

The refinement row gives $\rho_{i+1,i}(P_{i+1}(x)) \sim_i P_i(x)$. Since A_i respects $\sim_i$, the premise $A_i(P_i(x))$ transports across this classifier witness and gives the inherited claim. □

Definition 9414.135 (Concept layer)**.** *A* concept layer *is a readout layer together with an operation and certificate surface*

$$\mathcal{C} = (R, \sim_R, P, \mathsf{Op}, \mathsf{Cert}, \Lambda).$$

Here Op *is the family of constructions licensed at the layer, and* Cert *records closure, compatibility, stability, and replay obligations for those constructions.*

Definition 9414.136 (Concept tower)**.** *A* concept tower *is a finite or countable sequence $(\mathcal{C}_i)$ of concept layers together with refinement certificates*

$$\mathcal{C}_{i+1} \preceq \mathcal{C}_i.$$

The tower order is not a chronology of discovery. It is a dependency order for forming predicates and operations without hidden supply.

Definition 9414.137 (Tower-internal concept)**.** *Let $\mathcal{T} = (\mathcal{C}_i)$ be a concept tower. A concept is* tower-internal *when its carrier, predicate, operation, or section data is obtained from a finite prefix of $\mathcal{T}$ by the displayed formation rules of the prefix: readout along a layer map, pullback along a refinement map, dependent sum or product over a displayed carrier, fiber formation, section formation, equality transport along a classifier witness, and application of an operation whose certificate row belongs to one of the layers.*

Definition 9414.138 (BEDC-generable concept)**.** *Relative to a displayed concept tower $\mathcal{T}$, a* BEDC-generable concept *is a concept whose complete formation record is a finite BEDC expression over the source carrier, the displayed layer carriers, the readout and refinement maps, the classifier witnesses, and the certified operation rows of $\mathcal{T}$. The phrase does not range over untyped host-language ideas; it ranges only over objects whose dependencies are present in the tower ledger.*

Theorem 9414.139 (Tower closure for BEDC-generable concepts)**.** *Every BEDC-generable concept relative to $\mathcal{T}$ is tower-internal.*

Proof. The formation record of a BEDC-generable concept is finite. Induct on its formation. A source carrier, layer carrier, readout, refinement map, classifier witness, or certified operation row is already present in a finite prefix of $\mathcal{T}$, hence is tower-internal. Pullback, dependent sum, dependent product, fiber formation, section formation, and equality transport are exactly the allowed tower-internal formation rules. The induction step therefore preserves tower-internality for every constructor in the finite record. □

Corollary 9414.140 (Relative absence of outside concepts)**.** *If the word "concept" is restricted to BEDC-generable concepts relative to $\mathcal{T}$, then there is no concept outside $\mathcal{T}$.*

Proof. By Definition 9414.139, each concept in the restricted range is tower-internal. Thus no concept in that range is both BEDC-generable and outside the tower. □

Principle 9414.141 (Typed boundary of the outside-concept claim)**.** *The unqualified sentence "no concept is outside the tower" is not a closed BEDC claim. It becomes a theorem only after the quantifier is typed as ranging over BEDC-generable concepts relative to a displayed tower. Without this restriction the sentence quantifies over an unstated host-language domain and has no carrier, equality, or formation ledger.*

Theorem 9414.142 (Layered infinity theorem)**.** *For every readout layer $\mathcal{L}$ over X, the layer residue is the fiber family*

$$\infty_{\mathcal{L}}(r) = \mathrm{Fib}_{\mathcal{L}}(r).$$

It is not an element of the visible carrier R. If a finer layer $\mathcal{M}$ refines $\mathcal{L}$, then the finer layer partitions each previous fiber through the map of Definition 9414.45.

Proof. The first assertion is Definition 9414.41. The second assertion follows because the type of $\infty_{\mathcal{L}}$ is a family indexed by R, not a row of R. The refinement assertion is exactly Definition 9414.45. □

Principle 9414.143 (No point replacement for residue)**.** *Whenever a projection hides a source distinction, the hidden content may be named only as fiber data, section-relative meaning, or a gap row. It may not be replaced by a single additional visible point unless a new carrier, classifier, readout, and naming certificate have been displayed.*

Definition 9414.144 (Pre-prime tower segment)**.** *The pre-prime segment of a concept tower has the dependency shape*

$$\mathsf{Distinction} \to \mathsf{Hist} \to \mathsf{Cont} \to \mathsf{Unary} \to \mathsf{Nat} \to \mathsf{Order} \to \mathsf{Add} \to \mathsf{Mul} \to \mathsf{Div} \to \mathsf{Unit} \to \mathsf{Prime}.$$

The divisibility predicate is formed only after multiplication:

$$d \mid n \quad :\Longleftrightarrow \quad \exists q,\ n = dq.$$

The prime predicate is then the layer expression

$$\mathsf{Prime}(p) \quad :\Longleftrightarrow \quad p > 1 \ \wedge\ \forall d,\ d \mid p \Rightarrow d = 1 \vee d = p.$$

Theorem 9414.145 (Prime is not a primitive layer)**.** *In the pre-prime tower segment,* Prime *cannot be formed before the multiplication and divisibility layers.*

Proof. The displayed definition of $\mathsf{Prime}(p)$ contains the predicate $d \mid p$. The displayed definition of $d \mid p$ contains the product dq. Without the multiplication layer there is no displayed divisibility predicate, and without divisibility there is no displayed prime predicate. □

Definition 9414.146 (Prime-coordinate readout)**.** *Once a unique-factorization certificate is displayed, the prime-coordinate readout sends a positive integer n to the finite-support family*

$$\Phi(n) = (p \mapsto v_p(n)) \in \mathbb{N}^{(\mathbb{P})}.$$

Each prime p determines the coordinate vector e_p with value 1 at p and value 0 at every other prime.

Definition 9414.147 (Prime-occupancy carrier)**.** *The prime-occupancy carrier is the finite-support configuration space*

$$T_{\mathbb{P}} := \mathbb{N}^{(\mathbb{P})}.$$

Its elements are rows

$$k : \mathbb{P} \to \mathbb{N}$$

with finite support. The coordinate k_p records how many times the prime axis p is occupied. The carrier $T_{\mathbb{P}}$ is therefore the multiplicative-divisibility presentation of positive arithmetic after unique factorization has been displayed; it is not the set of primes $\mathbb{P}$ alone and not the absolute base of the whole BEDC tower.

Definition 9414.148 (Arithmetic multiplication presentation)**.** *The arithmetic multiplication presentation is the readout*

$$\Phi : \mathbb{N}_{>0} \to T_{\mathbb{P}}$$

of Definition 9414.146, together with the readback

$$\Psi(k) := \prod_{p \in \mathbb{P}} p^{k_p}.$$

The product is finite because k has finite support. The readout and readback are licensed only relative to the multiplication, divisibility, unit, order, natural-number, and unique-factorization certificates of the pre-prime tower segment.

Theorem 9414.149 (Multiplication becomes coordinate addition)**.** *For positive integers m, n with displayed unique factorization,*

$$\Phi(mn) = \Phi(m) + \Phi(n).$$

Proof. Unique factorization gives

$$m = \prod_p p^{v_p(m)}, \qquad n = \prod_p p^{v_p(n)}.$$

Multiplying the two expressions gives

$$mn = \prod_p p^{v_p(m)+v_p(n)}.$$

Uniqueness of the factorization certificate identifies the p-coordinate of mn with $v_p(m) + v_p(n)$ for every prime p. □

Principle 9414.150 (Prime ontology is relative arithmetic presentation)**.** *The phrase "the ontology is prime" can be used in BEDC only in the relative sense that the positive multiplicative arithmetic layer is read through $T_{\mathbb{P}}$. In that presentation, the unit is the zero configuration, multiplication is coordinate addition, divisibility is coordinatewise comparison,*

$$m \mid n \quad \Longleftrightarrow \quad \forall p,\ v_p(m) \le v_p(n),$$

and the gcd and lcm rows are coordinatewise minimum and maximum:

$$v_p(\gcd(m,n)) = \min(v_p(m), v_p(n)), \qquad v_p(\operatorname{lcm}(m,n)) = \max(v_p(m), v_p(n)).$$

This gives a strong coordinate presentation of the multiplicative layer; it does not erase the pre-prime construction of multiplication, divisibility, unit, order, and natural-number structure.

Theorem 9414.151 (Prime axes are independent)**.** *If $p_1, \ldots, p_k$ are pairwise distinct primes and*

$$\sum_{i=1}^{k} a_i e_{p_i} = 0,$$

then $a_i = 0$ for every i.

Proof. Fix j. Evaluating the displayed equality at the coordinate p_j gives $a_j = 0$, because every other basis vector has value 0 at p_j. Since j is arbitrary, all coefficients vanish. □

Definition 9414.152 (p-adic local readout)**.** *For a fixed prime p, the p-adic valuation readout on a nonzero rational is the p-coordinate of its prime factorization:*

$$v_p(x) = \Phi(x)(p).$$

The symbol $v_p(0) = \infty_{\mathrm{val}}$ is an added valuation-top row, not a natural number, integer, rational, or prime.

Principle 9414.153 (Prime axis locality)**.** *The p-adic layer is a local readout of the prime-coordinate layer. It does not create the prime axis; it consumes the already displayed unique-factorization and coordinate certificates.*

Definition 9414.154 (Symmetric readout layer)**.** *A readout layer is symmetric when it carries an involution $\tau : X \to X$ and the visible readout is invariant:*

$$P(\tau x) \sim_R P(x).$$

A numerical readout $H : X \to A$ is anti-invariant when the codomain has an involution $j : A \to A$ and

$$H(\tau x) = j(H(x)).$$

Theorem 9414.155 (Quotient readout forces a fixed point)**.** *Let τ be invisible to a quotient readout $P : X \to R$. If an anti-invariant readout $H : X \to A$ also descends through P, so that $H = \bar{H} \circ P$, then every displayed source row satisfies*

$$H(x) = j(H(x)).$$

Proof. Since H descends through P and $P(\tau x) \sim_R P(x)$, the quotient readout gives $H(\tau x) = H(x)$ at the displayed classifier resolution. Anti-invariance gives $H(\tau x) = j(H(x))$. Combining the two equalities yields $H(x) = j(H(x))$. □

Definition 9414.156 (Half-value layer)**.** *Let $Z \subseteq X$ be a displayed source subcarrier. A* half-value layer *over Z is a triple*

$$\mathcal{H} = (B, b, h)$$

where B is a displayed half-value carrier, $b : Z \to B$ is a readout, and $h \in B$ is the distinguished fixed half-value row.

Definition 9414.157 (Fixed half fiber)**.** *For a half-value layer $\mathcal{H} = (B, b, h)$, the fixed half fiber is*

$$\mathrm{Fix}_{\mathcal{H}} := \sum_{z:Z} (b(z) = h),$$

with projection

$$\pi_{\mathcal{H}} : \mathrm{Fix}_{\mathcal{H}} \to Z, \qquad \pi_{\mathcal{H}}(z, p) := z.$$

Definition 9414.158 (Half-value section)**.** *The half-value section type of $\mathcal{H}$ is*

$$\mathsf{Sec}_{\mathcal{H}} := \sum_{\sigma : Z \to \mathrm{Fix}_{\mathcal{H}}} (\pi_{\mathcal{H}} \circ \sigma = \mathrm{id}_Z).$$

Theorem 9414.159 (Half-value propositions are sections)**.** *The proposition*

$$\prod_{z:Z} (b(z) = h)$$

is naturally equivalent to $\mathsf{Sec}_{\mathcal{H}}$.

Proof. Given $u : \prod_{z:Z} (b(z) = h)$, define

$$\sigma_u(z) := (z, u(z)).$$

Then $\pi_{\mathcal{H}}(\sigma_u(z)) = z$, so σ_u is a section. Conversely, a section σ sends each z to a pair (z', p) with $p : b(z') = h$. The section condition identifies z' with z, and transport gives $p : b(z) = h$. These constructions are inverse by equality elimination. □

Theorem 9414.160 (Half value as fixed point)**.** *Over the rationals, the equation*

$$x = 1 - x$$

holds if and only if

$$x = \frac{1}{2}.$$

Proof. If $x = 1 - x$, then $2x = 1$, hence $x = 1/2$. Conversely, substituting $x = 1/2$ gives $1 - x = 1/2 = x$. □

Corollary 9414.161 (Matched-twin half value)**.** *For a balanced matched-twin protocol in the sense of Definition 9414.67, the value* $1/2$ *is the fixed point forced by the invisible involution on the quotient surface.*

Proof. Definition 9414.68 gives pairwise accuracy $1/2$ for every quotient-measurable binary rule. The preceding theorem identifies this number as the unique fixed point of $x \mapsto 1 - x$. □

Definition 9414.162 (Critical-line mirror)**.** *For* $s \in \mathbb{C}$*, write*

$$s = \frac{1}{2} + \delta + i\tau_s.$$

The critical-line mirror is

$$J(s) := 1 - \overline{s}.$$

Theorem 9414.163 (Critical line as mirror fixed surface)**.** *If* $s = \frac{1}{2} + \delta + i\tau_s$*, then*

$$J(s) = \frac{1}{2} - \delta + i\tau_s.$$

Consequently $J(s) = s$ *exactly when* $\Re(s) = 1/2$*.*

Proof. The conjugate is $\overline{s} = 1/2 + \delta - i\tau_s$. Therefore

$$J(s) = 1 - \overline{s} = \frac{1}{2} - \delta + i\tau_s.$$

The equality $J(s) = s$ is then equivalent to $-\delta = \delta$, hence to $\delta = 0$, which is precisely $\Re(s) = 1/2$. □

Theorem 9414.164 (Symmetric half-value unification)**.** *The matched-twin null value, the rational fixed point of* $x \mapsto 1-x$*, and the critical-line condition* $\Re(s) = 1/2$ *are instances of the same paper-side pattern: an involution is invisible at a quotient layer, an anti-invariant readout is forced to a fixed surface if it is claimed to descend, and the fixed surface is the half-value layer.*

Proof. For matched twins, the involution swaps two source witnesses while the quotient-visible observation remains unchanged; the binary label is anti-invariant, and Definition 9414.68 gives the half-value. For rational scores, the involution is $x \mapsto 1 - x$, whose fixed point is $1/2$ by Definition 9414.160. For the critical line, the mirror $J(s) = 1 - \overline{s}$ flips the bias coordinate δ and fixes exactly $\Re(s) = 1/2$ by Definition 9414.163. The three cases share the same quotient-invisibility and anti-invariance form. □

Theorem 9414.165 (Layered projection synthesis)**.** *Layered projection theory consists of the following paper-side closure schema:*

1. *a layer exposes only the classifier value of a readout;*
2. *the residual source content is a fiber family, not a point of the visible carrier;*
3. *refinements decompose previous fibers only through displayed readback maps and classifier transport;*
4. *concept predicates may be formed only after their operational dependencies have appeared in the tower;*
5. *BEDC-generable concepts are tower-internal relative to their displayed formation record;*
6. *predicates inherit only through displayed readback maps and classifier-respect rows;*
7. *half-value claims are fixed-point claims induced by displayed involutions and anti-invariant readouts, equivalently section claims into fixed half fibers.*

Proof. The first two clauses are Definition 9414.142. The third clause is Definition 9414.45. The fourth clause is represented by Definition 9414.145. Predicate inheritance is Definition 9414.134. The relative closure of generable concepts is Definition 9414.139, with the typing boundary recorded by Definition 9414.141. The fixed-point reading of half values is Definition 9414.164, and the section reading is Definition 9414.159. These clauses are independent of additional axioms; they use only displayed carriers, readout maps, classifier witnesses, refinement maps, and ledgered certificates. □

9414.8 Layered extension theory

Remark 9414.166 (Paper-side theory status)**.** *This section records the paper-side form of layered extension theory. It is tracked against the basic Lean readout and layer-extension carriers. Its role is to separate extension from endpoint imagery: a refinement layer is a compatible readout of the same source carrier, and the material available for refinement is the fiber family of the preceding layer rather than a single terminal point.*

Lean4 (definition) `BEDC.RealityConstrained.LayerExtension`

Definition 9414.167 (Layer residue)**.** *Let*

$$\mathcal{L}_n = (R_n, \sim_n, P_n, \Lambda_n, \mathsf{NameCert}_n)$$

be a readout layer over a displayed source carrier T. For $r : R_n$, the residual fiber at r is

$$\mathrm{Res}_n(r) := \sum_{t:T} (P_n(t) \sim_n r).$$

The layer residue is the family

$$\infty_n : R_n \to \mathsf{Type}, \qquad \infty_n(r) := \mathrm{Res}_n(r).$$

Thus ∞_n is a surface of source fibers indexed by R_n, not a row of R_n.

Principle 9414.168 (Residue is a fiber surface)**.** *The residue of a layer may be read only as the family $r \mapsto \mathrm{Res}_n(r)$. Replacing that family by a single visible endpoint loses the section data required by Definition 9414.41 and violates the point replacement boundary of Definition 9414.143.*

Definition 9414.169 (Compatible layered extension)**.** *Let $\mathcal{L}_n$ and $\mathcal{L}_{n+1}$ be readout layers over the same source carrier T:*

$$\mathcal{L}_i = (R_i, \sim_i, P_i, \Lambda_i, \mathsf{NameCert}_i).$$

The layer $\mathcal{L}_{n+1}$ is a compatible layered extension *of $\mathcal{L}_n$ when there is a displayed readback map*

$$\rho_{n+1,n} : R_{n+1} \to R_n$$

such that every source row $t : T$ satisfies

$$\rho_{n+1,n}(P_{n+1}(t)) \sim_n P_n(t).$$

The notation

$$\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n$$

records this compatibility row.

Theorem 9414.170 (Extension preserves the source carrier)**.** *If $\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n$, then $\mathcal{L}_{n+1}$ introduces no tower-external source object. It is a further readout of T, and its readback agrees with the preceding readout at the $\sim_n$ resolution.*

Proof. The refined readout has source

$$P_{n+1} : T \to R_{n+1}.$$

Thus every refined record row is still read from T. The compatibility certificate gives

$$\rho_{n+1,n}(P_{n+1}(t)) \sim_n P_n(t)$$

for each $t : T$. Hence the refinement is not a second source carrier; it is a compatible representation of the same source through a finer record surface. □

Lean4 ✓ `BEDC.RealityConstrained.LayerExtension.read forget commutes`

Definition 9414.171 (Fiber-distinguishing extension)**.** *A compatible layered extension $\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n$ is* fiber-distinguishing *at $r : R_n$ when there exist source rows $t_0, t_1 : T$ such that*

$$P_n(t_0) \sim_n r, \qquad P_n(t_1) \sim_n r,$$

but

$$P_{n+1}(t_0) \not\sim_{n+1} P_{n+1}(t_1).$$

It is fiber-distinguishing when this occurs at some displayed $r : R_n$.

Definition 9414.172 (Generalized analytic continuation)**.** *A* generalized analytic continuation *of $\mathcal{L}_n$ is a compatible layered extension*

$$\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n$$

that is fiber-distinguishing for a residue of $\mathcal{L}_n$. The word "analytic" here records the continuation role: a later layer keeps the earlier readout available by readback while expressing source structure that the earlier carrier did not separate.

Theorem 9414.173 (Continuation is residual separation)**.** *Every generalized analytic continuation separates two source rows that belonged to one residual fiber of the preceding layer.*

Proof. By Definition 9414.172, the extension is fiber-distinguishing. Hence there are $r : R_n$ and $t_0, t_1 : T$ with

$$P_n(t_0) \sim_n r, \qquad P_n(t_1) \sim_n r,$$

while the refined layer satisfies

$$P_{n+1}(t_0) \not\sim_{n+1} P_{n+1}(t_1).$$

Thus the refined layer separates rows that the preceding layer left in a single residual fiber. □

Definition 9414.174 (Fiber-certified geometric continuation)**.** *Let* $\mathcal{L}_n = (R_n, \sim_n, P_n, \Lambda_n, \mathsf{NameCert}_n)$ *be a readout layer over* T*, and let* $C_n : R_n \to \mathsf{Prop}$ *be a displayed constraint row. A* fiber-certified geometric continuation *of* $(\mathcal{L}_n, C_n)$ *consists of a compatible layered extension*

$$\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n,$$

a displayed constraint row $C_{n+1} : R_{n+1} \to \mathsf{Prop}$*, and ledger rows certifying:*

$$\rho_{n+1,n}(P_{n+1}(t)) \sim_n P_n(t)$$

for every $t : T$*,*

$$C_{n+1}(r) \to C_n(\rho_{n+1,n}(r))$$

for every displayed $r : R_{n+1}$*, and a fiber-distinguishing witness for some residue of* $\mathcal{L}_n$*. The certificate rows include source identity, readback transport, constraint transport, fiber separation, and ledger closure.*

Theorem 9414.175 (Geometric continuation unfolds residual fibers)**.** *Every fiber-certified geometric continuation recovers its higher record surface as compatible source readout data over the preceding layer, rather than as a tower-external geometric carrier.*

Proof. The continuation contains a compatible layered extension, so Definition 9414.170 shows that the later readout is still a readout of the same source carrier T and readbacks to the preceding layer. The constraint-transport row sends every later admissible record to an earlier admissible record. The fiber-distinguishing witness then says that the later layer separates two source rows inside one residue of the earlier layer. Hence the higher record surface unfolds residual fiber data already indexed by R_n, with compatibility and ledger rows making the passage auditable. □

Lean4 ✓ `BEDC.RealityConstrained.LayerExtension.realized fiber commutes`

Principle 9414.176 (Continuation recovers carriers, not remote values)**.** *The restriction-style image of analytic continuation should not be read as the discovery of a value at a distant point. In BEDC terms it is the recovery of a carrier whose later readout restricts to the earlier one and whose residue, constraint, branch, and transport rows are ledgered. The same pattern governs high-dimensional geometry: lower-dimensional projections expose residual fibers, and a lawful continuation displays the carrier, readback, constraint transport, fiber separation, and audit rows that make those fibers available as geometric structure.*

Definition 9414.177 (Local-cluster extension)**.** *A compatible layered extension is* local-clustered *when its refined readout is assembled from a displayed family of local readouts*

$$P_{n+1,i} : T \to R_{n+1,i}, \qquad i : I_n,$$

together with a record assembly row

$$P_{n+1}(t) = \big(P_{n+1,i}(t)\big)_{i:I_n}$$

and readback data

$$\rho_{n+1,n}\big(P_{n+1}(t)\big) \sim_n P_n(t).$$

The indices $i : I_n$ are local windows, places, valuations, sections, or other displayed coordinates over the residual surface of $\mathcal{L}_n$.

Theorem 9414.178 (Local clusters refine fibers, not endpoints)**.** *In a local-cluster extension, the later layer refines the family $r \mapsto \mathrm{Res}_n(r)$ rather than a distinguished endpoint of R_n.*

Proof. The data of Definition 9414.177 supplies readouts $P_{n+1,i}$ from the same source T and a readback row to R_n. For a record $r : R_n$, the relevant source rows are exactly the fiber

$$\mathrm{Res}_n(r) = \sum_{t:T} (P_n(t) \sim_n r).$$

The local coordinates can distinguish such rows only through their source readouts and the displayed readback certificate. No clause of the definition selects a terminal row of R_n. □

Example 9414.179 (Place completions as local clusters)**.** *Let the source carrier be the rational field and let the local index set be the displayed set of places*

$$V_{\mathbb{Q}} = \{\infty\} \cup \mathsf{Prime}^{\uparrow}.$$

For $v = \infty$ the local completion is the archimedean completion $\mathbb{Q}_\infty = \mathbb{R}$. For a prime p, the local completion is the p-adic completion $\mathbb{Q}_p$, read through the valuation coordinate described in Definition 9414.152. The family

$$(\mathbb{Q}_v)_{v:V_{\mathbb{Q}}}$$

is therefore a place-indexed local-cluster extension of rational readout. It is not generated by a single point at infinity; it is generated by the displayed valuation and completion rows for the places.

Definition 9414.180 (Orthogonal cluster readouts)**.** *Let a local-cluster extension have readouts $(P_{n+1,i})_{i:I_n}$. The cluster is* orthogonal *when the ledger contains an independence certificate saying that changing the i-th coordinate, within its admissible source fiber, does not consume or force the j-th coordinate for $i \neq j$.*

Theorem 9414.181 (Prime-coordinate clusters are orthogonal)**.** *The prime-coordinate readouts supplied by unique factorization form an orthogonal cluster.*

Proof. Definition 9414.151 states that distinct prime axes are independent. The p-adic readout of Definition 9414.152 consumes the p-coordinate of this factorization surface. Hence the prime-local coordinates form an orthogonal cluster in the sense of Definition 9414.180. □

Definition 9414.182 (Composite extension information)**.** *Let $a, b : T \to R$ be orthogonal readouts in a layer. A record $c : T \to C$ is* composite extension information *when the ledger contains a formation row*

$$(a, b) \rightsquigarrow c$$

and no row expressing c from a alone or from b alone. The role of c is to record structure carried by the joint source readout rather than by either marginal projection.

Principle 9414.183 (Composite coordinates may force a layer)**.** *When a theory requires a composite record c as in Definition 9414.182, the layer that contains only the marginal readouts is not closed under the required expression. A compatible extension must display a carrier for c, its classifier, its readback to the marginal rows, and its gap ledger.*

Definition 9414.184 (Extension tower)**.** *An* extension tower *over T is a finite or countable sequence of readout layers*

$$\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \dots$$

together with compatible readback maps

$$\mathcal{L}_{n+1} \rightsquigarrow \mathcal{L}_n$$

for every displayed successor pair.

Definition 9414.185 (Restriction-style continuation)**.** *Let $U \subset V$ be displayed domains. A classical restriction-style continuation has an upper record carrier of functions on V, a lower record carrier of functions on U, and the readback map*

$$\rho(F) = F|_U.$$

The compatibility row for a lower record f is

$$\rho(F) = f.$$

Theorem 9414.186 (Classical analytic continuation is a special case)**.** *Classical analytic continuation is a compatible layered extension whose readback map is restriction.*

Proof. In Definition 9414.185, the readback map is $F \mapsto F|_U$. The usual continuation condition says precisely that the refined record F restricts to the preceding record f. This is the compatibility equation of Definition 9414.169. □

Principle 9414.187 (Continuation need not be linear extension)**.** *The restriction-style case does not exhaust layered extension. A later layer may be obtained by place completions, valuation coordinates, orthogonal local clusters, completion rows, or recovered-carrier compatibility ledgers. The invariant is readback to the preceding layer, not geometric continuation from a terminal point.*

Definition 9414.188 (Zeta as a recovered extension carrier)**.** *The zeta carrier of Definition 9243.2 is read here as a multi-row extension carrier. Its displayed rows include the Dirichlet-series projection, Euler-product projection, analytic continuation row, archimedean completion row, functional-equation mirror row, pole ledger, zero ledger, branch ledger, and transport ledger.*

Principle 9414.189 (Zeta extension tower)**.** *The zeta carrier is not a single formula with a later decorative completion. It is a recovered carrier whose rows are compatible readouts of one source object across domains, prime-local channels, archimedean completion, mirror transport, and zero ledgers. The Dirichlet row is a projection of this carrier on its convergence half-plane, while the completed carrier supplies the surface on which zero-ledger statements are typed.*

Principle 9414.190 (RH target as extension compatibility obstruction)**.** *Within the zeta extension tower, the object named* $\mathsf{LocalGlobalZeroExclusion}_\zeta$ *in Definition 9246.1 is the remaining zero-ledger compatibility obstruction: a row separated from the functional-equation fixed surface by a positive local defect must be shown incompatible with a completed zeta-zero row. This principle does not assert the obstruction has been discharged; it identifies the carrier-level obligation isolated by Definition 9246.13.*

9414.9 Cataphatic-apophatic layering

This section records the reality-constrained reading of the rational, located, and apophatic layers. The named base line is the displayed rational layer $\mathbb{Q}$: a value at this layer is a finite rational name together with the exact equality relation already carried by $\mathsf{Rat}^{\uparrow}$. The cataphatic extension is the located-real layer: a real presentation is read through witnessed rational approximations, finite windows, moduli, and apartness evidence, not through an uninspected quotient representative. The apophatic far end is the socket at which a request for unsupported pointification, arbitrary supremum, or global representative selection is refused rather than imported.

Lean4 (definition) `BEDC.Derived.NonCollapseInvariantUp.NonCollapseInvariant`

Lean4 (definition) `BEDC.Derived.NonCollapseInvariantUp.NonCollapseWitness`

Lean4 ✓ `BEDC.Derived.NonCollapseInvariantUp.NonCollapseInvariant no rat retraction`

Lean4 ✓ `BEDC.Derived.NonCollapseInvariantUp.RatMetricKitConcrete no noncollapse witness`

Principle 9414.191 (Non-collapse invariant)**.** *Let w be a located-real presentation. The formal non-collapse invariant requires every rational q to be separated from w by displayed apartness evidence and, more strongly, to keep producing such evidence beyond any proposed equality modulus at a fixed displayed separation scale for that q. With a separating rational metric kit, this invariant rules out an exact rational retraction: there is no $q : \mathbb{Q}$ such that w is* $\mathsf{LRealEq}$*-equivalent to the stationary rational presentation of q.*

Proof. The Lean theorem cited above is the checked core. An assumed retraction supplies an equality modulus. The invariant supplies, beyond that same modulus, a rational snapshot of w apart from q. The separating metric law says that the displayed apartness and the displayed equality-close row cannot both hold at that snapshot. □

The slogan for the layer is: *The Omega does not deny points; it refuses unjustified pointification.* Thus the refusal is not a refusal of rational points, named located windows, or certified limits. It is a refusal to collapse a located or residual surface to a rational name, selected representative, arbitrary supremum, or terminal object without the certificate that performs that specific collapse.

The geometric language sometimes attached to this picture is only an intuition. Calling the rational layer an "axis", calling located apartness a transverse read, or drawing a three-way picture of rational names, located approximation, and apophatic sockets does not assert an inner product, a projection operation, a three-way law, a pro-object construction, or a universal-property theorem about future refinements. In BEDC terms, "undiscovered axis" means stability under later universe-polymorphic or context-extension refinements, not a quantified family of future directions.

The constructive boundary is the same one used by located analysis. Located suprema, located cuts, and finite-window apartness may be admitted when their moduli and witnesses are displayed. An arbitrary supremum or unrestricted representative selector is not admitted by this section. Such a selector would be an external supply socket in the sense of chapter 9395, and any use of it must remain visibly socketed through $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ rather than silently converted into a point of the cataphatic layer.

Remark 9414.192 (Formal scope)**.** *The checked Lean content is the invariant-to-no-retraction theorem and the exact-kit boundary theorem: every point in the exact rational metric kit retracts to a rational snapshot, so that kit does not carry a non-collapse witness. This section does not claim a checked $\sqrt{2}$ construction, a diagonal irrational construction, a located-supremum theorem, a rational trichotomy theorem, or an apophatic universal-property theorem. Those are separate arithmetic and interface targets.*

9414.10 Recursive orthogonal-axis normal forms

Remark 9414.193 (Paper-side theory status)**.** *This section records the paper-side form of recursive orthogonal-axis normal forms. It has no Lean target in this chapter. Its role is to make precise the relation between finite expression, orthogonal coordinate axes, and normalisation: a finite object is not merely listed as data, but is read through a certified normal form whose axes are generated by the available classifiers.*

Principle 9414.194 (Primes are not the absolute first layer)**.** *In the arithmetic tower, the prime layer is the first canonical orthogonal splitting layer after multiplication, divisibility, unit, and the required order and natural-number structure have been displayed. It is not the absolute base of the tower. This is the axis-form reading of Definition 9414.145.*

Definition 9414.195 (Axis layer)**.** *An* axis layer *over a displayed source carrier T is a packet*

$$\mathcal{A}_n = (A_n, R_n, \Phi_n, \mathsf{NF}_n, \Lambda_n)$$

where A_n is a displayed family of coordinate axes, R_n is the record carrier expressible at the layer, Φ_n is a finite-support coordinate readout

$$\Phi_n : R_n \to C^{(A_n)},$$

NF_n is a normal-form classifier on such coordinate records, and Λ_n records formation, uniqueness, readback, and residual gap data.

Definition 9414.196 (Finite axis expression)**.** *Given an axis layer $\mathcal{A}_n$, a finite expression of a record $x : R_n$ is a coordinate row*

$$\Phi_n(x) = \sum_{a \in A_n}^{\text{fin}} c_a e_a$$

together with a normal-form certificate

$$\mathsf{NF}_n\big((c_a)_{a \in A_n}\big)$$

and a readback certificate identifying the coordinate expression with x at the classifier resolution of R_n.

Definition 9414.197 (Finite code prime window)**.** *Suppose a layer has a finite natural-number code*

$$\mathrm{code}_n : R_n \to \mathsf{Nat}^{\uparrow}$$

and a partial readback on the image of the code,

$$\mathrm{decode}_n : \mathrm{im}(\mathrm{code}_n) \to R_n.$$

For a finite set $S \subset_{\text{fin}} \mathsf{Prime}^{\uparrow}$, the prime-window readout of $x : R_n$ is

$$\Pi_{n,S}(x) := \big(v_p(\mathrm{code}_n(x))\big)_{p \in S}.$$

The full prime-coordinate readout is the finite-support family

$$\Pi_{n,\mathsf{Prime}^{\uparrow}}(x) := \big(v_p(\mathrm{code}_n(x))\big)_{p \in \mathsf{Prime}^{\uparrow}}.$$

Theorem 9414.198 (Finite codes have prime-coordinate normal forms)**.** *Assume $x : R_n$ has a finite code and that the readback row satisfies*

$$\mathrm{decode}_n(\mathrm{code}_n(x)) \sim_n x.$$

Then x admits a finite-support prime-coordinate expression.

Proof. By unique factorization, the positive natural code has a prime factorization

$$\mathrm{code}_n(x) = \prod_{p \in \mathsf{Prime}^{\uparrow}} p^{v_p(\mathrm{code}_n(x))}$$

with only finitely many nonzero exponents. Hence the family

$$\left(v_p(\mathrm{code}_n(x))\right)_{p \in \mathsf{Prime}^{\uparrow}}$$

is finite-support. The displayed readback row then returns the code to a record classified with x. Thus the prime-coordinate row is a normal-form expression of x at the layer resolution. □

Definition 9414.199 (Axis-generation operator)**.** *An* axis-generation operator $\mathcal{G}$ *assigns to an axis family A the family generated by the admissible formation rows*

$$\begin{aligned}\mathcal{G}(A) = A \cup \mathsf{PrimeSplit}(A) \cup \mathsf{Ratio}(A) \cup \mathsf{EquationRoot}(A)\\ \cup \mathsf{Completion}(A) \cup \mathsf{OrthogonalJoin}(A) \cup \mathsf{Transform}(A).\end{aligned}$$

Here PrimeSplit *reads finite codes through prime axes,* Ratio *forms fraction axes,* EquationRoot *records roots forced by displayed equations,* Completion *records limit and completion axes,* OrthogonalJoin *records independent coordinate joins, and* Transform *records Fourier, Laplace, zeta, or analogous transform axes when their carriers and readbacks are displayed.*

Definition 9414.200 (Recursive axis tower)**.** *A* recursive axis tower *is a monotone sequence of axis families*

$$A_0 \subseteq A_1 \subseteq A_2 \subseteq \cdots$$

with

$$A_{n+1} = \mathcal{G}(A_n)$$

for the displayed axis-generation operator. Its expressive union is

$$A_\omega := \bigcup_{n<\omega} A_n.$$

The union is a tower-internal expressive surface over T, not a second source carrier.

Theorem 9414.201 (Relative absence of tower-external axes)**.** *If an axis concept is defined as an axis obtained by finitely many applications of $\mathcal{G}$, or as a finite normal-form combination of such axes, then it belongs to some A_n.*

Proof. Induct on the formation height of the axis concept. At height zero the concept belongs to the initial displayed family A_0 or to a finite combination over it. At successor height, the concept is produced by one of the formation rows contained in $\mathcal{G}$ from concepts already available at bounded height. By the induction hypothesis those inputs belong to some common A_n. The defining equation $A_{n+1} = \mathcal{G}(A_n)$ places the output in A_{n+1}. A finite normal-form combination uses only finitely many such axes, hence lies in a sufficiently high finite layer. □

Definition 9414.202 (Multiplicative atom normal form)**.** *The multiplicative atom normal form of a positive natural number is its prime-coordinate expression*

$$n = \prod_{p \in \mathsf{Prime}^{\uparrow}} p^{v_p(n)}.$$

Its classifier is multiplicative divisibility with unit rows and unique-factorization certificates.

Definition 9414.203 (Recursive additive normal form)**.** *A recursive additive normal form is a finite expression*

$$n = \sum_k \varepsilon_k F_k$$

whose allowed coefficients and local exclusion rules are supplied by a recursive axis grammar. In the Zeckendorf case,

$$\varepsilon_k \in \{0, 1\}, \qquad \varepsilon_k \varepsilon_{k+1} = 0.$$

Its classifier is additive value together with the no-adjacent normal grammar.

Principle 9414.204 (Prime and Zeckendorf normal forms are complementary)**.** *Prime factorization and Zeckendorf normal form are both unique finite-expression schemes, but they answer different classifier questions. Prime factorization reads multiplicative atoms; Zeckendorf normal form reads recursive additive axes with a local exclusion rule. Neither classifier replaces the other.*

Definition 9414.205 (Classifier-relative irreducible axis)**.** *Let $\mathcal{A}_n$ be an axis layer whose normal-form classifier admits a displayed composition operation*

$$\star_n : R_n \times R_n \to R_n$$

with a unit row 1_n, or more generally a displayed finite generation grammar. A row $p : R_n$ is a classifier-relative irreducible axis *when it is not classified with the unit and every licensed decomposition of p is trivial at the layer classifier:*

$$p \not\sim_n 1_n$$

and

$$p \sim_n a \star_n b \quad \Longrightarrow \quad a \sim_n 1_n \text{ or } b \sim_n 1_n.$$

In a generation grammar without a single binary operation, the same phrase means that p is a normal axis row not obtainable by a non-trivial lower formation rule allowed by the classifier and ledger.

Definition 9414.206 (BEDC-generable self-similar axis system)**.** *A* BEDC-generable self-similar axis system *over a source carrier T is a packet*

$$\mathcal{S} = (X, F, \theta, P, \mathsf{NF}, \rho, \Lambda)$$

where X is a displayed record carrier, F is the recursive shape operator, θ is a self-similar decomposition row

$$\theta : X \to F(X),$$

$P : T \to X$ is the source readout, NF is a finite normal-form classifier for the displayed expressions of X, ρ is the recursive readback map from normal expressions to X, and Λ records formation, stability, irreducible-support, and gap data. The system is read as BEDC-generable only through this packet; the self-similar equation is not admitted as a bare untyped object.

Definition 9414.207 (Irreducible support basis)**.** *An irreducible support basis for Definition 9414.206 is a displayed axis family $B_{\mathcal{S}}$ such that every non-unit normal expression has a finite support row*

$$\mathrm{supp}_{\mathcal{S}}(x) \subset_{\mathrm{fin}} B_{\mathcal{S}}$$

and every $b \in B_{\mathcal{S}}$ is a classifier-relative irreducible axis in the sense of Definition 9414.205. The basis is part of the certificate data; it is not inferred from self-similarity alone.

Theorem 9414.208 (Relative priming of self-similar normal forms)**.** *Let $\mathcal{S}$ be a BEDC-generable self-similar axis system equipped with finite normal forms, recursive readback, classifier stability, and an irreducible support basis. Then every non-unit normal row of $\mathcal{S}$ exposes classifier-relative irreducible axes. These axes are prime-like only relative to the classifier and ledger of $\mathcal{S}$.*

Proof. Let $x : X$ be a non-unit normal row. By the finite normal-form clause of Definition 9414.206, x has a finite normal expression. By Definition 9414.207, the support of this expression is a finite row

$$\mathrm{supp}_{\mathcal{S}}(x) \subset_{\mathrm{fin}} B_{\mathcal{S}}.$$

Since x is non-unit, its normal support contains at least one displayed axis $b \in B_{\mathcal{S}}$. The basis certificate says that each such b is classifier-relative irreducible. The recursive readback row ρ and the classifier-stability ledger identify the normal expression with x at the resolution of $\mathcal{S}$, so the exposed irreducible axes are invariants of the displayed normal row. No statement about ordinary integer primes follows unless the classifier is the multiplicative-divisibility classifier of Definition 9414.202. □

Principle 9414.209 (Self-similar priming is relative)**.** *The phrase "prime-like atom" is meaningful in this chapter only after the source carrier, classifier, normal-form grammar, readback map, and irreducible-support ledger have been displayed. In the multiplicative-divisibility classifier the atoms are ordinary primes. In a recursive additive classifier they are Zeckendorf or Fibonacci axes. In a spectral classifier they may be conjugate characteristic roots, as in Definition 9346.13. In a local completion classifier they may be places or valuation rows. In a zeta-facing classifier they may be prime-local probes or zero-ledger invariant rows. These are classifier-relative invariants, not one absolute layer of objects shared by all self-similar systems.*

Definition 9414.210 (Fibonacci-type recursive axis system)**.** *A recursive orthogonal-axis system is* Fibonacci type *when its local normalisation row contains the carry rule*

$$e_k + e_{k+1} \rightsquigarrow e_{k+2}$$

and its normal records forbid simultaneous activation of adjacent axes.

Theorem 9414.211 (Fibonacci-type normal forms are Zeckendorf-type)**.** *In a Fibonacci-type recursive axis system, every normal finite expression is Zeckendorf-type: it is a finite sum of non-adjacent recursive axes.*

Proof. The carry row

$$e_k + e_{k+1} \rightsquigarrow e_{k+2}$$

identifies adjacent activation as non-normal. Repeatedly applying the local normalisation row eliminates adjacent pairs and replaces them by a higher recursive axis. A normal expression is therefore precisely one in which no adjacent pair remains. This is the Zeckendorf-type grammar recorded in Definition 9347.1 and in the finite-support layer of Definition 9347.10. □

Theorem 9414.212 (Golden scale from second-order local recursion)**.** *Assume a positive recursive axis scale* (a_n) *satisfies*

$$a_{n+2} = a_{n+1} + a_n$$

and that the ratio

$$\lambda = \lim_{n\to\infty} \frac{a_{n+1}}{a_n}$$

exists. Then

$$\lambda^2 = \lambda + 1.$$

Consequently, the positive scale is the golden scale φ*.*

Proof. Divide the recurrence by a_{n+1}:

$$\frac{a_{n+2}}{a_{n+1}} = 1 + \frac{a_n}{a_{n+1}}.$$

Taking the displayed limit gives

$$\lambda = 1 + \frac{1}{\lambda}.$$

Equivalently, $\lambda^2 = \lambda + 1$. The positive solution is the golden scale φ. □

Principle 9414.213 (Scope of Fibonacci and golden appearances)**.** *Fibonacci and golden-scale rows are licensed when the relevant carrier has a second-order local recursion, adjacent-axis exclusion, and a normal-form or self-similar growth certificate. They are not universal explanations of unrelated phenomena. In the Axis-Zeckendorf route this scope is enforced by the fold ledger, residual-fiber row, and cannot-claim boundary of Definition 9346.38.*

Principle 9414.214 (Axis formation sequence)**.** *The general motion of a recursive axis tower is*

displayed axes → *orthogonal combination* → *local conflict* → *normalisation* → *derived axis.*

This sequence is a formation pattern over the same source carrier T*. It does not license an unledgered passage to an external ontology.*

Principle 9414.215 (Finite expression as audited normal form)**.** *The finite expression of a mathematical source item is an audited normal-form row over a generated axis family. Prime decompositions, Zeckendorf decompositions, algebraic root extensions, completions, orthogonal joins, and transform carriers are different ways of displaying such rows, each under its own classifier, readback map, and residual ledger.*

9414.11 Constructive information generation

Remark 9414.216 (Paper-side theory status)**.** *This section records the paper-side synthesis named* constructive information generation*. It is tracked against the Lean information-generation witness and finite trace carriers. Its role is to give one vocabulary for the preceding projection, extension, and recursive-axis normal-form sections: information is not treated as a pre-existing collection of objects, but as an auditable structure generated from a source carrier through interfaces, classifiers, orthogonal axes, compatible extensions, and ledger closure.*

Lean4 (definition) `BEDC.RealityConstrained.InfoGenWitness`
Lean4 (definition) `BEDC.RealityConstrained.GenerationTrace`

Definition 9414.217 (Constructive information generation layer)**.** *Let T be a displayed source carrier. A* constructive information generation layer *over T is a packet*

$$\mathcal{I}_n = (R_n, \sim_n, P_n, \Lambda_n, \mathsf{Cert}_n, \mathsf{Aud}_n)$$

where R_n is the record carrier, $\sim_n$ is the layer classifier,

$$P_n : T \to R_n$$

is the source readout, Λ_n is the provenance and gap ledger, Cert_n records naming, stability, transport, and replay certificates, and Aud_n records the audit rows by which a consumer may check formation and readback.

Definition 9414.218 (Layer-generated information)**.** *For a source row $t : T$, the information generated at layer n is the certified record*

$$\mathsf{Info}_n(t) := \big(P_n(t), \Lambda_n(t), \mathsf{Cert}_n(t), \mathsf{Aud}_n(t)\big).$$

The visible part of this information is $P_n(t)$. The full information row includes the ledger and certificate data that explain why the visible record is licensed.

Principle 9414.219 (No bare information objects)**.** *In constructive information generation, an information object is not admitted merely by being named. It enters the theory through a source history, a signature or record surface, a package or domain row, an evidence row, a classifier, and a gap or audit ledger. This is the vision-level reading of the* `BaseReflectionSetup` *interface:* `Hist`*,* `SigObj`*,* `Pkg`*,* `Pi`*,* `Domain`*,* `Evidence`*, together with relations such as* `hsame`*,* `InDom`*,* `SigGen`*,* `TokIntro`*, and* `InGapSig`*.*

Definition 9414.220 (Information residue)**.** *For a generation layer $\mathcal{I}_n$ and a record $r : R_n$, the information residue over r is*

$$\mathrm{InfoFib}_n(r) := \sum_{t:T} (P_n(t) \sim_n r).$$

The layer-infinite information surface is the family

$$\infty_n^{\mathsf{info}} : R_n \to \mathsf{Type}, \qquad \infty_n^{\mathsf{info}}(r) := \mathrm{InfoFib}_n(r).$$

It is a residual source family over R_n, not a further row of R_n.

Theorem 9414.221 (Visible records need not exhaust source rows)**.** *Let $\mathcal{I}_n$ be a generation layer. If there are source rows $t_0, t_1 : T$ with t_0 separated from t_1 at the source classifier and*

$$P_n(t_0) \sim_n P_n(t_1),$$

then the visible record $P_n(t_0)$ does not exhaust the source information over that record.

Proof. Put $r := P_n(t_0)$. The row t_0 gives an element of

$$\mathrm{InfoFib}_n(r) = \sum_{t:T} (P_n(t) \sim_n r)$$

by reflexivity at the displayed classifier. The displayed relation $P_n(t_0) \sim_n P_n(t_1)$, read symmetrically when needed by the layer classifier, gives an element over the source row t_1 in the same fiber. Since the source rows remain separated, the visible record r has not identified all source information. It is a record surface with a non-trivial residual source family. □

Principle 9414.222 (Layer infinity is not a terminal record)**.** *The expression* ∞_n^{info} *denotes the family*

$$r \mapsto \mathrm{InfoFib}_n(r)$$

over R_n*. It is not an element of* R_n *and not a final object at the end of the layer. It names the residual source family through which a later compatible refinement may continue to read the same source carrier.*

Definition 9414.223 (Compatible information refinement)**.** *Let* $\mathcal{I}_n$ *and* $\mathcal{I}_{n+1}$ *be generation layers over the same source carrier* T*. A* compatible information refinement *is a displayed readback map*

$$\rho_{n+1,n} : R_{n+1} \to R_n$$

with

$$\rho_{n+1,n}(P_{n+1}(t)) \sim_n P_n(t)$$

for every $t : T$*, together with the transport, audit, and gap rows that make this readback reproducible.*

Theorem 9414.224 (Information generation is fiber refinement)**.** *If* $\mathcal{I}_{n+1}$ *is a compatible information refinement of* $\mathcal{I}_n$ *and separates two rows in one information residue of* $\mathcal{I}_n$*, then the refined layer generates no tower-external information. It further distinguishes the residual source fiber of the preceding layer.*

Proof. The readout of the refined layer is still

$$P_{n+1} : T \to R_{n+1},$$

so the source carrier has not changed. The readback condition of Definition 9414.223 is exactly the compatibility row of Definition 9414.169. Hence Definition 9414.170 applies. If the refined layer separates two rows in one residue, then Definition 9414.173 identifies the generation as a residual separation of the preceding layer. □

Lean4 ✓ `BEDC.RealityConstrained.info monotone`
Lean4 ✓ `BEDC.RealityConstrained.readback consistent`

Definition 9414.225 (Information generation axes)**.** *An information generation layer carries an* axis family A_n *when its visible records admit finite-support coordinate expressions over* A_n *in the sense of Definition 9414.196. A subfamily* $(a_i : T \to A_i)_{i:I}$ *is orthogonal when the layer ledger contains the independence certificate of Definition 9414.180.*

Definition 9414.226 (Constructive information generation operator)**.** *The constructive information generation operator is the axis-generation operator of Definition 9414.199 equipped with a ledger-closure clause:*

$$\mathcal{G}_{\mathsf{info}}(A) = A \cup \mathsf{PrimeSplit}(A) \cup \mathsf{Ratio}(A) \cup \mathsf{EquationRoot}(A)$$
$$\cup \mathsf{Completion}(A) \cup \mathsf{OrthogonalJoin}(A) \cup \mathsf{Transform}(A) \cup \mathsf{LedgerClosure}(A).$$

The final clause records that a generated expression is not fully admitted until provenance, readback, stability, and gap rows have been entered in the layer ledger.

Definition 9414.227 (Constructive information generation tower)**.** *A* constructive information generation tower *over T is a packet*

$$\mathfrak{I} = \left(T, \mathcal{I}_n, A_n, \mathcal{G}_{\mathsf{info}}, \rho_{n+1,n}, \Lambda_n, \mathsf{Cert}_n, \mathsf{Aud}_n\right)_{n<\omega}$$

such that

$$A_{n+1} = \mathcal{G}_{\mathsf{info}}(A_n)$$

and each successor layer is equipped with a compatible information refinement row whenever its visible records extend the preceding layer.

Definition 9414.228 (Generated auditable information)**.** *Relative to a tower $\mathfrak{I}$, a row is* generated auditable information *when it is obtained by finitely many applications of $\mathcal{G}_{\mathsf{info}}$, finite coordinate formation, compatible readback, residual fiber separation, or ledger closure, and its formation record includes the certificates required by the layers it uses.*

Theorem 9414.229 (Relative absence of tower-external information)**.** *Every generated auditable information row relative to $\mathfrak{I}$ belongs to a finite layer of the tower.*

Proof. The formation record is finite by Definition 9414.228. Induct on its length. Initial source and layer rows belong to the first finite layer in which they are displayed. A successor formation step is one of the clauses of $\mathcal{G}_{\mathsf{info}}$, a finite coordinate formation, a compatible readback, a residual separation, or a ledger-closure row. Each such step is displayed in the next layer or in a finite later layer. The induction therefore places the generated row in some $\mathcal{I}_n$. □

Definition 9414.230 (Corecursive readout loop)**.** *A* corecursive readout loop *over a source carrier T is a finite or cyclic family of generation layers*

$$\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_n$$

with source readouts

$$P_i : T \to R_i$$

and displayed readback maps

$$\rho_{j,i} : R_j \to R_i$$

whenever the loop returns a later visible surface to an earlier one, such that

$$\rho_{j,i}(P_j(t)) \sim_i P_i(t)$$

for every $t : T$. A loop row is admitted only together with the transport, audit, and gap ledgers that make each readback reproducible.

Principle 9414.231 (Visible origins are gauges)**.** *An initial visible surface is an observation origin, not an absolute source. A binary surface*

$$P_B : T \to \{B_0, B_1\}$$

displays a minimal two-row classifier for the current inquiry, but the rows B_0 and B_1 are not themselves the whole source carrier. Their residual source families are

$$\mathrm{Fib}(B_i) := \sum_{t:T} (P_B(t) = B_i).$$

Changing the observation origin replaces the visible gauge; it does not replace the source carrier unless a separate recovered-carrier certificate is supplied.

Theorem 9414.232 (Binary visible rows are not bottom points)**.** *Let $P_B : T \to \{B_0, B_1\}$ be a binary visible surface. If for some $i \in \{0, 1\}$ the fiber*

$$\sum_{t:T} (P_B(t) = B_i)$$

contains two source-separated rows, then B_i is not a bottom point of the source carrier.

Proof. If B_i were a bottom point of the source carrier at this interface, then a source row over B_i would be fully represented by that visible marker. The displayed fiber instead contains two source-separated rows with the same visible marker. Thus the marker is a quotient record of the source carrier, not a source atom. □

Definition 9414.233 (Binary fiber self-similarity)**.** *A binary visible surface $P_B : T \to \{B_0, B_1\}$ is* fiber-self-similar *when for each displayed $i \in \{0, 1\}$ the fiber*

$$T_{B_i} := \sum_{t:T} (P_B(t) = B_i)$$

carries a further binary readout

$$P_{B,i} : T_{B_i} \to \{B_0, B_1\}$$

with a readback certificate to the original binary classifier.

Theorem 9414.234 (Binary fiber self-similarity gives finite path certificates)**.** *If a binary visible surface is fiber-self-similar, then every source row $t : T$ determines binary prefixes*

$$B_{i_0}, B_{i_1}, \ldots, B_{i_k}$$

of arbitrary finite length, and each prefix is accompanied by a finite iterated-fiber certificate.

Proof. The first marker is $P_B(t) = B_{i_0}$. This supplies an element of the fiber $T_{B_{i_0}}$. The self-similar readout on that fiber supplies the next marker B_{i_1} and a certificate that the source row lies in the corresponding refined fiber. Repeating this finite construction gives any requested finite prefix. Each step contributes one displayed readout and one readback certificate, so the prefix certificate remains finite. □

Definition 9414.235 (Locally finite global corecursive tower)**.** *A constructive information tower is* locally finite and globally corecursive *when every admitted information row has a finite formation record in the sense of Definition 9414.228, while the displayed source carrier may be read again through residual fibers and readback loops as in Definition 9414.230. The global structure is not required to expose an absolute bottom layer, but every accepted row must have a finite audit prefix.*

Theorem 9414.236 (Corecursive towers have finite audit prefixes)**.** *In a locally finite globally corecursive tower, every generated auditable information row is certified by a finite prefix of the readout loop.*

Proof. By Definition 9414.235, the row has a finite formation record. Each occurrence of a loop return is a displayed readback map of Definition 9414.230, equipped with transport, audit, and gap ledgers. Replacing each loop return in the formation record by its finite readback certificate yields a finite prefix using the same clauses as Definition 9414.228. Hence Definition 9414.229 places the row in a finite layer. □

Principle 9414.237 (Readout cycles enrich fibers)**.** *A readout cycle is not a logical circular definition. Returning from a later layer to an earlier visible surface identifies the earlier classifier row by readback, while the intervening formation record remains in the ledger. Thus a cycle may return to the same visible marker, but it returns with a richer residual fiber and a longer audit surface. Proofs in such a tower are carried by invariants transported through displayed readback maps, as in Definition 9414.134, rather than by appeal to an absolute tower bottom.*

Theorem 9414.238 (Readout loops are not layer equalities)**.** *A corecursive readout loop from $\mathcal{I}_n$ back to $\mathcal{I}_0$ does not imply $\mathcal{I}_n = \mathcal{I}_0$. It implies only that the later visible row reads back to the earlier classifier:*

$$\rho_{n,0}(P_n(t)) \sim_0 P_0(t).$$

Proof. The loop condition is a statement in the record carrier R_0 and its classifier $\sim_0$. It supplies neither an equality of carriers $R_n = R_0$ nor an equality of readouts $P_n = P_0$. The higher layer may carry additional ledgers, certificates, and fiber-separation rows that are compressed by $\rho_{n,0}$ when read back to R_0. □

Theorem 9414.239 (Bottomless corecursive tower principle)**.** *Let $\mathfrak{I}$ be a tower over one source carrier T. If each layer is a source readout, each successor layer has compatible readback, each visible record retains its information residue, every accepted information row has a finite formation record, and predicates are transported by displayed readback invariants, then the tower may be globally bottomless while all admitted information remains locally finite and auditable.*

Proof. All layers read from the same source carrier by assumption, and compatible readback prevents successor layers from becoming tower-external source objects. Information residues keep visible records from being terminal source atoms. Finite formation records give finite audit prefixes by Definition 9414.236. Predicate transport through readback maps gives the invariant proof discipline of Definition 9414.134. These clauses allow the global readout system to be corecursive without requiring any visible origin to be an absolute bottom. □

Theorem 9414.240 (Lawful nontrivial generation is generalized extension)**.** *Let a successor layer in $\mathfrak{I}$ produce a visible record not separated by the preceding layer. If the successor carries a compatible readback row and separates a preceding information residue, then that generation is a generalized analytic continuation in the sense of Definition 9414.172.*

Proof. The readback row gives

$$\rho_{n+1,n}(P_{n+1}(t)) \sim_n P_n(t)$$

for every source row. Thus the successor is a compatible layered extension. The assumed residual separation says exactly that it is fiber-distinguishing. By Definition 9414.172, this is a generalized analytic continuation. □

Principle 9414.241 (Normal forms as information-generation classifiers)**.** *Prime decomposition and Zeckendorf decomposition are canonical examples of information generation by classifier-specific normal forms. The prime-coordinate normal form of Definition 9414.202 reads multiplicative atoms. The recursive additive normal form of Definition 9414.203 reads no-adjacent recursive axes. Their complementarity is the content of Definition 9414.204.*

Principle 9414.242 (Second-order recursion and golden scale)**.** *The golden scale is licensed in this theory only when the carrier has the second-order recursion, adjacent-axis exclusion, and growth certificate of Definition 9414.212. It is therefore a structural invariant of a specific normalisation grammar, not an unrestricted explanatory constant.*

Principle 9414.243 (Recovered carriers as ledgered information)**.** *Recovered carriers are information-generation rows whose identity is distributed across compatible projections, completions, transforms, and ledgers. The zeta carrier of Definition 9243.2 is read in this way: Dirichlet, Euler, continuation, archimedean completion, functional-equation, pole, zero, branch, and transport rows jointly license the carrier. This principle records typing and compatibility; it does not discharge the zero-ledger obstruction named in Definition 9414.190.*

Definition 9414.244 (Constructive information generation theory)**.** Constructive information generation theory *is the study of towers*

$$\mathfrak{I} = \left(T, \mathcal{I}_n, A_n, \mathcal{G}_{\mathsf{info}}, \rho_{n+1,n}, \Lambda_n, \mathsf{Cert}_n, \mathsf{Aud}_n\right)_{n<\omega}$$

and of the finite auditable rows they generate from T through orthogonal splitting, recursive combination, local completion, compatible extension, transform formation, normalisation, and ledger closure.

Theorem 9414.245 (Core form of constructive information generation)**.** *In a constructive information generation tower, every finite auditable information row is produced by a finite chain of source readout, classifier formation, orthogonal-axis generation, compatible extension, normal-form certification, and ledger closure.*

Proof. By Definition 9414.228, an admitted information row has a finite formation record using exactly the displayed generation operations and their audit rows. The source readout and classifier formation place the row in a typed layer. Orthogonal-axis generation and normal-form certification give its finite expression when coordinates are used. Compatible extension supplies readback to earlier layers, and ledger closure supplies provenance, stability, replay, and gap data. Hence the row is produced by the stated finite chain. □

Concretized in §9066: The finite trace packet realizes constructive information generation as visible rows plus residue and refinement data.

9414.12 Quantum measurement as information generation

Concretized in §8949: Measurement residue ledgers are realized as finite projection, fiber, section-update, and audit NameCert rows.

Remark 9414.246 (Paper-side boundary)**.** *This section records a structural interface between constructive information generation and quantum measurement. It is not a formulation of quantum mechanics, not a substitute for Hilbert-space dynamics, and not a derivation of the Born rule. Its role is narrower: to identify the measurement, projection, residue, and update grammar that a later physical quantum interface would have to certify.*

Lean4 (definition) `BEDC.RealityConstrained.MeasurementBasis`
Lean4 (definition) `BEDC.RealityConstrained.MeasurementInterface`

Definition 9414.247 (Quantum-facing generation layer)**.** *A* quantum-facing generation layer *over a source carrier T is a constructive information generation layer*

$$\mathcal{I}_n = (R_n, \sim_n, P_n, \Lambda_n, \mathsf{Cert}_n, \mathsf{Aud}_n)$$

together with a displayed measurement-context row M_n. The readout

$$P_n : T \to R_n$$

is interpreted as the measurement-interface projection for that context, and R_n is the corresponding outcome carrier. The context row records the preparation surface, observable or probe family, instrument configuration, and public outcome ledger. Without M_n, the same source readout is only an abstract projection row.

Principle 9414.248 (Measurement results are licensed records)**.** *A quantum-facing outcome is not treated as a bare pre-existing value. It is an information row of the form*

$$\mathsf{Info}_n(t) = (P_n(t), \Lambda_n(t), \mathsf{Cert}_n(t), \mathsf{Aud}_n(t))$$

as in Definition 9414.218, with the additional measurement-context row of Definition 9414.247. The visible outcome is $P_n(t)$; the licensed record includes the preparation, interface, classifier, evidence, replay, and audit data that make the outcome public.

Lean4 (definition) `BEDC.RealityConstrained.MeasurementGenerationWitness`

Lean4 ✓ `BEDC.RealityConstrained.MeasurementInterface.readback entry listed and projected`

Definition 9414.249 (Measurement residue)**.** *For a quantum-facing generation layer and an outcome $r : R_n$, the measurement residue is the information fiber*

$$\mathrm{QFib}_n(r) := \sum_{t:T} (P_n(t) \sim_n r).$$

It is the source structure not separated by the chosen measurement interface. It is not a hidden classical value and not an additional outcome row of R_n.

Principle 9414.250 (Projection is not hidden-variable assignment)**.** *Reading a source through a measurement interface compresses it to a public outcome and leaves a residual fiber. This does not assert that the source already carried a definite classical value for every possible measurement. It asserts only that a chosen layer has exposed one record surface while the remaining source distinctions are retained as fiber and ledger data.*

Definition 9414.251 (Hilbert-style axis packet)**.** *A* Hilbert-style axis packet *is a candidate physical refinement of an information-generation axis family. It consists of a displayed carrier H, a finite or certified orthogonal probe family*

$$E = \{e_i : i \in I\},$$

projection rows $\Pi_i : H \to H_i$, scalar readbacks c_i, and a ledger recording inner-product, normalization, and probe provenance. When such a packet is available, a state-like row may be displayed in the form

$$\psi = \sum_{i \in I} c_i e_i$$

at the certified resolution of the packet.

Principle 9414.252 (Measurement bases are readout layers)**.** *Different quantum measurement bases correspond, in this vocabulary, to different readout layers over the same source carrier. A position-like readout, a momentum-like readout, and two spin-direction readouts need not share a single classical refinement layer. When no displayed common refinement and readback ledger is present, the theory may compare the layers only through certified bridge rows, not by treating all outcomes as simultaneously present classical coordinates.*

Principle 9414.253 (Noncommutativity as absence of a classical common refinement)**.** *For quantum-facing layers, noncommuting observables may be read structurally as a failure of a common classicalizing refinement: two readout surfaces over T do not jointly descend to one visible carrier that records both outcome families with compatible readback maps. This principle is a typing guide. It does not by itself supply the operator commutator, the uncertainty bound, or the Hilbert-space theorem behind a particular physical model.*

Definition 9414.254 (Born-weight layer)**.** *A* Born-weight layer *over a Hilbert-style axis packet is an additional measure ledger assigning an outcome weight*

$$\mu_n(i)$$

to each displayed projection row. In the usual quantum semantics this weight is

$$\mu_n(i) = \|\Pi_i \psi\|^2.$$

The layer must include normalization, positivity, finite or countable additivity at the declared scope, and calibration rows connecting the weights to empirical frequencies.

Principle 9414.255 (Born rule is not supplied by projection grammar)**.** *Constructive information generation supplies source, readout, residue, axis, and ledger grammar. It does not, by itself, choose the Born weight map. A quantum physical theory must add the Born-weight layer of Definition 9414.254, together with the physical calibration and experimental prediction rows that make those weights empirical.*

Definition 9414.256 (Section-style measurement update)**.** *Given an outcome $r : R_n$, a section-style measurement update is a licensed map*

$$U_r : R_n \to \mathrm{QFib}_n(r)$$

or, in a Hilbert-style packet, a normalized post-outcome row such as

$$\psi \mapsto \frac{\Pi_r \psi}{\|\Pi_r \psi\|}.$$

Such an update is admissible only when its domain, nonzero denominator condition, post-outcome carrier, and replay ledger are displayed.

Principle 9414.257 (State update is a certified section)**.** *The section-style reading of measurement update does not make collapse a mysterious primitive. It treats post-measurement state selection as a licensed section from a visible outcome surface into an admissible post-outcome fiber. Without the section certificate, the visible outcome does not determine a unique source row.*

Definition 9414.258 (Entanglement as non-product residue)**.** *Let a composite source be read through local outcome carriers R_A and R_B. A composite residue over (r_A, r_B) is* non-product *when there is no displayed equivalence*

$$\mathrm{QFib}_{AB}(r_A, r_B) \simeq \mathrm{QFib}_A(r_A) \times \mathrm{QFib}_B(r_B)$$

compatible with the local readback ledgers. Such non-product residue is the information-generation form of an entanglement boundary.

Principle 9414.259 (Entanglement keeps the source global)**.** *An entangled composite may have local readouts while its residual source fiber remains global. Local outcomes do not exhaust the source, and a product of local fibers is not available without a displayed product decomposition certificate. This is the fiber-language counterpart of the usual refusal to treat an entangled state as a product state.*

Principle 9414.260 (Qubit boundary)**.** *A two-outcome readout resembles the binary carriers used in the Axis-Zeckendorf and golden conjugate interfaces, but a qubit is not just a two-point set. A qubit requires complex amplitudes, phase, normalization, admissible measurement axes, and Born weights. The binary carrier supplies only the visible outcome alphabet; the physical quantum content lives in the certified amplitude, phase, weight, and update layers.*

Principle 9414.261 (Quantum physical closure obligations)**.** *To turn the preceding grammar into a quantum physical theory one must add at least: Hilbert-space carrier laws, complex amplitude rows, observable/operator laws, unitary time-evolution rows, Born-weight calibration, measurement-update rules, tensor-product composition, and experiment-facing prediction ledgers. Without these rows the chapter records only the information-generation and measurement-projection syntax.*

9414.13 Finite observation and open fit

Definition 9414.262 (Reality trace)**.** *For a physical system R, a reality trace of length n is a finite sequence*

$$H_n(R) = (o_0, o_1, \ldots, o_n)$$

of observations generated by interacting with R through an admissible observation protocol.

The trace is not reality itself. It is the finite observational history through which reality becomes available to a model.

Definition 9414.263 (Observation bundle)**.** *An observation bundle Π is a finite or recursively extendable family of admissible questions. Given R, M, and a trace H_n, the observed and model-generated signatures are written*

$$\mathrm{ObsSig}_\Pi(R \mid H_n), \qquad \mathrm{ModSig}_\Pi(M \mid H_n).$$

Definition 9414.264 (Finite observational fit)**.** *A model M has finite observational fit to a system R at trace length n, relative to Π, when*

$$\mathrm{ModSig}_\Pi(M \mid H_n) \sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H_n).$$

This judgment is denoted $\mathrm{Fit}_n(M, R; \Pi)$.

Finite fit says only that M has not been separated from R by the finite observations available up to H_n. It does not assert $M = R$.

Definition 9414.265 (Open physical fit)**.** *Open physical fit, written* $\mathrm{OpenFit}(M, R; \Pi)$*, is the continuing condition that every actually obtained finite trace carries a finite fit certificate, while no internal final certificate is assumed for*

$$\forall n, \mathrm{Fit}_n(M, R; \Pi).$$

Concretized in §1131: Finite observation fit becomes a finite NameCert packet with explicit refutation and open-continuation ledgers.

Concretized in §1168: Open physical fit becomes a finite residue-gate NameCert packet whose refutation row stays visible.

Concretized in §2247: Observation residual fit is realized as finite observed-signature, model-signature, residual-row, refutation-surface, open-continuation, transport, replay, provenance, and NameCert data.

Definition 9414.266 (Physical induction certificate)**.** *A physical induction certificate is a stability certificate*

$$\mathrm{PhysInd}(M, R; \Pi)$$

licensing a transition from previously observed fit to predicted continuation fit:

$$\mathrm{Fit}_n(M, R; \Pi) \rightsquigarrow \mathrm{PredFit}_{n+1}(M, R; \Pi).$$

This differs from mathematical induction. Mathematical induction applies after the carrier and its constructors are already closed. Physical induction is a stability claim about future observations of a system whose complete generative law is not externally given.

Definition 9414.267 (Final-fit oracle)**.** *A final-fit oracle for M and R is a hypothetical judgment $\Omega(M, R; \Pi)$ such that*

$$\Omega(M, R; \Pi) \Longleftrightarrow \forall H,\ \mathrm{ModSig}_\Pi(M \mid H) \sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H),$$

where H ranges over all admissible actual and counterfactual observation histories.

The framework rejects such an oracle as an internally available physical object. It would be an observer above all observers, contrary to the no-global-standpoint discipline of chapter 9419.

9414.14 Open-fit theorems

Theorem 9414.268 (Finite fit does not imply total fit)**.** *For any finite trace length n,*

$$\mathrm{Fit}_n(M, R; \Pi)$$

does not imply

$$\forall k \geq n, \mathrm{Fit}_k(M, R; \Pi).$$

Proof. By Definition 9414.264, $\mathrm{Fit}_n(M, R; \Pi)$ compares model and system only on the finite trace H_n. The judgment contains no information about observations not yet made. Two continuations

$$H_n \cdot o_{n+1} \quad \text{and} \quad H_n \cdot o'_{n+1}$$

may agree on the prefix and separate at the next observation. A model can match R on the prefix and fail on one continuation. To infer future fit one must add a stability or physical induction certificate. □

Corollary 9414.269 (Empirical truth is not model identity)**.** *A successful finite model fit supports undefeatedness by the displayed trace, not model-reality identity.*

Proof. By Definition 9414.268, finite fit cannot imply total fit. Since identity would require agreement across all admissible observations and all reality-facing properties, finite fit is weaker than identity. □

Theorem 9414.270 (Mathematical induction and physical induction have different closures)**.** *Mathematical induction is valid when the target carrier is closed under known constructors. Physical induction requires an additional stability certificate because the complete future observation carrier of reality is not internally given.*

Proof. Mathematical induction over a carrier such as $\mathbb{N}$ relies on a closed generation rule:

$$0 \in \mathbb{N}, \qquad n \in \mathbb{N} \Longrightarrow S(n) \in \mathbb{N}.$$

The base case and successor step range over the closure of that carrier. Physical observation begins from a finite trace $H_n = (o_0, \ldots, o_n)$, but the observer does not possess a constructor theorem saying that every future reality-facing observation arises by a known successor rule compatible with the model. The transition from Fit_n to future fit therefore requires PhysInd as additional stability data. □

Corollary 9414.271 (Physical intuition is a classifier prior)**.** *What is often called physical intuition is a pre-formal expectation that future observations will remain within the current classifier.*

Proof. By Definition 9414.270, future fit does not follow from finite fit alone. Expectations based on prior experience, symmetry, scale, or dynamical regularity function as priors over admissible continuations. They await certification by further observation or by a displayed stability argument. □

Corollary 9414.272 (Physical law as stability certificate)**.** *A physical law is not merely an equation. It is a certified stability claim that a model-generated signature continues to match observed signatures under admissible extensions of observation.*

Proof. By Definition 9414.270, finite fit alone does not justify future prediction. Prediction requires a certificate licensing continuation from current fit to later fit. Since a physical law is used precisely to support such continuation, its certificate role is stability over an open observation process, together with an open surface on which future observations can refute the licensed continuation. □

Theorem 9414.273 (No internal final-fit oracle)**.** *No observer internal to the physical inquiry process can supply a final-fit oracle* $\Omega(M, R; \Pi)$*.*

Proof. An internal observer has a finite history and local observational constraints. To supply $\Omega(M, R; \Pi)$ it would have to certify agreement across all admissible observations, including not-yet-performed observations and observations from other possible perspectives. If the certifying observer is external to all observation histories, it is not internal to the inquiry system. If it is internal, it is itself a history subject to the same locality and no-global-synchronization constraints recorded in chapter 9419. In neither case is an internal final-fit oracle obtained. □

Corollary 9414.274 (Final theory is not an internal scientific certificate)**.** *A physical theory may be undefeated by every available observation, but the judgment that it is final cannot be internally certified.*

Proof. Such a judgment would require a final-fit oracle over all possible observations. Definition 9414.273 excludes that oracle as an internal object. □

Theorem 9414.275 (Model fit is itself an approximation tower)**.** *The judgment that a model approximates reality is itself obtained through a sequence of approximations.*

Proof. To assert that M approximates R, one observes R, obtaining a finite trace H_n rather than R itself. The trace is converted into $\mathrm{ObsSig}_\Pi(R \mid H_n)$. The model produces $\mathrm{ModSig}_\Pi(M \mid H_n)$. The two signatures are compared by $\sim_\Pi$, yielding a finite fit certificate. Thus the adequacy judgment factors as

$$R \longrightarrow H_n \longrightarrow \mathrm{ObsSig}_\Pi \longrightarrow \mathrm{Fit}_n \longrightarrow \mathrm{OpenFit}\,.$$

Each arrow hides information and therefore requires a ledger when later use may depend on the hidden layer. □

Corollary 9414.276 (Evidence for approximation is approximate)**.** *There are two distinct approximation judgments: the model's approximation to the system, and the finite evidence that licenses that approximation judgment. The second is more primitive, because the first is accessible only through it.*

Proof. By Definition 9414.275, the claim that M approximates R is produced through finite traces, signature extraction, classifier comparison, and fit certificates. These mediating structures are finite and revisable. Therefore the evidence for approximation is itself approximation-relative. □

Corollary 9414.277 (Falsification is finite and confirmation is open)**.** *A model can fail by one finite observation, but it cannot be finally confirmed by finitely many observations.*

Proof. A finite trace may produce

$$\mathrm{ModSig}_\Pi(M \mid H_n) \not\sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H_n),$$

which refutes finite fit. Final confirmation would require all admissible finite prefixes, which no single finite trace supplies. □

Concretized in §1132: RealityConstrainedOpenFitUp realizes finite observation fit as an open NameCert packet with finite refutation, continuation, ledger, and no final-fit oracle.

Concretized in §1137: OpenFitPacketUp realizes finite observational fit, finite refutation, and no-final-oracle boundaries as a concrete NameCert packet.

Concretized in §1169: Finite observation refutations are realized as finite BHist NameCert packets separating defeating observations from open-continuation and final-fit refusal rows.

9414.15 Ontic state as stable signature source

The distinction between mathematical approximation and physical adequacy has an ontological counterpart. A physical object is not identified with an arbitrary mathematical representation, nor with the final visible idealization produced by an approximation tower. It is approached as the stable source of reality-facing signatures. Mathematical representations remain indispensable, but they become physically meaningful only when tied to the signatures through which the system manifests itself under admissible observation bundles.

Definition 9414.278 (Ontic state)**.** *For a physical system R and an admissible family of observation bundles $\mathcal{P}$, the ontic state of R is the stable source of the system's reality-facing signatures:*

$$\mathrm{Ont}(R) := \mathrm{StableSource}(\{\mathrm{PhysSig}_\Pi(R)\}_{\Pi\in\mathcal{P}}).$$

This notation does not identify $\mathrm{Ont}(R)$ *with a single representation. A representation M expresses an aspect of* $\mathrm{Ont}(R)$ *only relative to specified bundles, classifiers, stability data, and ledgers.*

Definition 9414.279 (Ontic adequacy)**.** *A mathematical representation M is ontically adequate for R relative to Π when it preserves the physical signatures generated by the ontic state of R:*

$$\mathrm{PhysSig}_\Pi(M) \sim_\Pi \mathrm{PhysSig}_\Pi(\mathrm{Ont}(R)).$$

Ontic adequacy is stronger than formal consistency and stronger than mathematical approximation. It requires preservation of the reality-facing properties through which the system's state is physically manifested.

Definition 9414.280 (Ontic underdetermination)**.** *Representations M_1 and M_2 are ontically underdetermined by Π when*

$$\mathrm{PhysSig}_\Pi(M_1) \sim_\Pi \mathrm{PhysSig}_\Pi(M_2),$$

but there is a refined observation bundle $\Pi' \supseteq \Pi$ such that

$$\mathrm{PhysSig}_{\Pi'}(M_1) \not\sim_{\Pi'} \mathrm{PhysSig}_{\Pi'}(M_2).$$

Definition 9414.281 (Ontic residue)**.** *Let M be a representation of R relative to Π. The ontic residue of M relative to R and Π is the reality-facing content of R not preserved, represented, or queried by M under Π. Schematically,*

$$\mathrm{Res}_\Pi(R, M) := \mathrm{PhysSig}_{\mathcal{P}}(R) \setminus \mathrm{PhysSig}_\Pi(M),$$

where $\mathcal{P}$ denotes a broader family of admissible observation bundles. The notation names hidden or unqueried physical content; it need not be an ordinary set difference.

Definition 9414.282 (Representation ledger)**.** *A representation ledger for M as a model of R records*

$$\mathrm{RepLedger}(M, R; \Pi) = (H_n, \Pi, \mathrm{Classifier}, \mathrm{StabilityCert}, \mathrm{Res}_\Pi(R, M)).$$

It records the finite observational source, observation bundle, comparison classifier, stability certificate, and residue hidden by the representation.

This is the physical counterpart of BEDC's GAP ledger requirement. Compression is permitted, but the source route and hidden material must remain traceable; when compressions compose, the resulting ledger must retain the source map through the intermediate visible interfaces.

Theorem 9414.283 (Syntax does not exhaust ontic state)**.** *Let M be a mathematical representation of a physical system R. The syntactic existence of M does not imply that M exhausts* $\mathrm{Ont}(R)$.

Proof. A mathematical representation supplies equations, spaces, functions, parameters, diagrams, or computational rules. By Definition 9414.278, $\mathrm{Ont}(R)$ is the stable source of reality-facing signatures across admissible bundles. To exhaust that source, M would have to preserve every relevant physical signature of R across those bundles. The mere existence of the syntax of M supplies no observation bundle, classifier, stability condition, or residue ledger. Therefore syntax alone cannot establish ontic exhaustion. □

Corollary 9414.284 (Model as ontic interface)**.** *A model is a certified interface to* $\mathrm{Ont}(R)$*, not the ontic state itself.*

Proof. By Definition 9414.283, a representation does not exhaust the ontic state by syntactic existence. By Definition 9414.279, it becomes ontically adequate only relative to a signature-preservation condition. It therefore functions as an interface to the ontic state under specified conditions. □

Corollary 9414.285 (Realism requires representation ledgers)**.** *A realist interpretation of a model requires a certificate specifying which physical signatures are preserved and which remain as residue.*

Proof. If correspondence is read as ontic adequacy, then Definition 9414.279 requires signature preservation. By Definition 9414.282, such preservation must be accompanied by source, classifier, stability, and residue data. Bare representation is therefore insufficient. □

9414.16 Signature priority and exactness

Definition 9414.286 (Signature-realization map)**.** *A signature-realization map for a representation* M *under* Π *is a map*

$$\sigma_{M,\Pi} : M \longrightarrow \mathrm{PhysSig}_{\Pi}(M)$$

that extracts the physical signature predicted by M *under the observation bundle* Π*.*

Definition 9414.287 (Signature completeness and soundness)**.** *A representation* M *is signature-complete for* R *relative to* Π *when every reality-facing distinction of* R *exposed by* Π *is matched by a model distinction:*

$$\mathrm{PhysSig}_{\Pi}(R) \preceq \mathrm{PhysSig}_{\Pi}(M).$$

It is signature-sound when every distinction predicted by M *under* Π *is physically realized by* R*:*

$$\mathrm{PhysSig}_{\Pi}(M) \preceq \mathrm{PhysSig}_{\Pi}(R).$$

Definition 9414.288 (Signature exactness)**.** *A representation* M *is signature-exact for* R *relative to* Π *when it is both signature-complete and signature-sound, equivalently when*

$$\mathrm{PhysSig}_{\Pi}(M) \sim_{\Pi} \mathrm{PhysSig}_{\Pi}(R).$$

Theorem 9414.289 (Physical adequacy is signature exactness)**.** *A representation* M *is physically adequate for* R *relative to* Π *if and only if it is signature-exact for* R *relative to* Π*.*

Proof. Physical adequacy is the preservation of the reality-facing signature under the relevant observation bundle. This is exactly the equivalence displayed in Definition 9414.288. Completeness says the model misses no relevant distinction; soundness says it invents no unsupported distinction. Together these are signature-level adequacy. □

Corollary 9414.290 (Adequacy does not collapse representation into source)**.** *A physically adequate model may be mathematically non-identical to the system it represents.*

Proof. By Definition 9414.289, adequacy requires matching signatures under Π, not identity of representation and source. The model remains a representation and the system remains the source of the signatures. □

Corollary 9414.291 (Mathematical exactness may be physically inadequate)**.** *A representation may be mathematically exact within its formal system while failing to be physically adequate for R.*

Proof. Mathematical exactness concerns internal validity relative to formal definitions. Physical adequacy concerns signature exactness relative to reality-facing observation. If the mathematically exact model fails to preserve the observed signatures, it is physically inadequate despite its internal exactness. □

9414.17 Residue and refinement

Theorem 9414.292 (Coarse bundles induce ontic underdetermination)**.** *Let Π expose only a proper subset of the reality-facing properties relevant to R. Then there may exist distinct representations M_1 and M_2 such that*

$$M_1 \equiv_{\mathrm{phys},\Pi} M_2$$

while M_1 and M_2 are separated by a refined bundle $\Pi' \supseteq \Pi$.

Proof. Because Π exposes only a subset of the relevant properties, it ignores distinctions that may be exposed by a refinement. Choose representations that agree on all distinctions visible to Π but differ on a distinction visible under Π'. They are physically fit-equivalent under Π and separated under Π', which is exactly ontic underdetermination relative to the coarse bundle. □

Corollary 9414.293 (Underdetermination is bundle-relative)**.** *Underdetermination is indexed by an observation bundle, not absolute.*

Proof. The same pair of representations can be equivalent under Π and inequivalent under Π'. Therefore the underdetermination judgment depends on the bundle. □

Corollary 9414.294 (Refinement can make ambiguity empirical)**.** *A distinction that appears metaphysical under a coarse bundle may become empirical under a refined bundle.*

Proof. If Π cannot expose a distinction, the difference is not empirically separated at that level. If a refined bundle Π' exposes it, Definition 9414.292 turns the difference into a signature separation. □

Definition 9414.295 (Physically active residue)**.** *Residue $\mathrm{Res}_\Pi(R, M)$ is inactive relative to Π when it does not separate M from R under Π. It becomes physically active under a refined bundle Π' when*

$$\mathrm{PhysSig}_{\Pi'}(M) \not\sim_{\Pi'} \mathrm{PhysSig}_{\Pi'}(R).$$

Theorem 9414.296 (Hidden residue can become active)**.** *If M is physically adequate for R under Π, and a refined bundle $\Pi' \supseteq \Pi$ exposes residue not preserved by M, then M is not physically adequate under Π'.*

Proof. Adequacy under Π gives signature exactness at the coarse level. If Π' exposes residue not preserved by M, then the model-generated and reality-generated signatures differ under Π'. By Definition 9414.289, physical adequacy fails at the refined level. □

Corollary 9414.297 (Idealization is scale-relative)**.** *An idealization may be physically adequate at one scale and inadequate at another.*

Proof. Observation scale is part of the bundle. A coarse bundle may leave residue inactive, while a refined-scale bundle may expose it. Definition 9414.296 then separates adequacy at the two scales. □

9414.18 Observer-internal access

Definition 9414.298 (Observer-internal ontic access)**.** *Observer-internal access to* $\mathrm{Ont}(R)$ *is access in which every claim about the ontic state factors through*

$$H_n(R) \longrightarrow \mathrm{ObsSig}_\Pi(R \mid H_n) \longrightarrow \mathrm{Classifier}_\Pi \longrightarrow \mathrm{FitCert}_n(M, R; \Pi).$$

Definition 9414.299 (Ontic humility)**.** *Ontic humility is the methodological constraint that no model may be asserted to exhaust* $\mathrm{Ont}(R)$ *unless the relevant signature bundles, stability certificates, and ledgers have been specified. It is a discipline on representation, not skepticism about reality.*

Theorem 9414.300 (Observer-internal access does not eliminate ontic reality)**.** *If access to* $\mathrm{Ont}(R)$ *is observer-internal, it does not follow that* $\mathrm{Ont}(R)$ *is merely subjective.*

Proof. Observer-internal access means that claims about R pass through finite traces, signatures, classifiers, and certificates. It does not say that signatures are freely invented. Finite fit requires comparison between model-generated and reality-generated signatures, and failed fit occurs when reality generates a signature not matched by the model. Reality therefore constrains the observer-internal process. □

Corollary 9414.301 (Objectivity as invariant signature stability)**.** *Objectivity is the stability of signatures across admissible observers, bundles, and continuations, not access to reality from nowhere.*

Proof. By Definition 9414.300, reality constrains observer-internal access through signatures. If those signatures remain stable across admissible changes of observer, bundle, and continuation, the content is not private to one trace. It is invariant across the observational structure. □

9414.19 Certified ontic towers

The expression

$$\mathrm{Ont}(R) \sim \lim_{\Pi \in \mathcal{P}} \mathrm{PhysSig}_\Pi(R)$$

must be read as schematic. The limit is not a primitive completion imported into the theory; it names a research target: the reconstruction of ontic access as a certified tower of increasingly refined signatures.

Definition 9414.302 (Certified ontic tower)**.** *A certified ontic tower for* R *is a chain*

$$\Pi_0 \preceq \Pi_1 \preceq \Pi_2 \preceq \cdots$$

of increasingly refined observation bundles, together with fit certificates $\mathrm{FitCert}_i(M_i, R; \Pi_i)$ *and transition certificates* $\mathrm{TransCert}_{i \to j}$ *showing how signatures at level* i *are preserved, refined, or corrected at level* j*. The tower is ledger-complete when each transition records what was hidden, preserved, and newly exposed.*

Concretized in §1160: The certified ontic tower is realized as a finite staged signature-refinement and ledger-continuity NameCert packet.

Definition 9414.303 (Ontic convergence)**.** *A certified ontic tower converges ontically relative to a family of physical questions $\mathcal{Q}$ when, for every $q \in \mathcal{Q}$, there is a stage i such that all later stages preserve the answer to q up to the relevant physical classifier:*

$$\forall q \in \mathcal{Q},\ \exists i,\ \forall j \geq i,\ \mathrm{Ans}_{\Pi_j}(q) \sim_q \mathrm{Ans}_{\Pi_i}(q).$$

Theorem 9414.304 (Ontic convergence yields stable content, not final totality)**.** *If a certified ontic tower converges relative to $\mathcal{Q}$, then it yields stable physical content for questions in $\mathcal{Q}$, but it does not yield a final total description of R.*

Proof. By Definition 9414.303, every question in $\mathcal{Q}$ eventually stabilizes across the tower, so the tower yields stable content for those questions. The family $\mathcal{Q}$ need not contain every admissible physical question. A new question, refined bundle, or new scale of interaction may expose previously inactive residue. Therefore convergence over $\mathcal{Q}$ does not imply final totality. □

Corollary 9414.305 (Ontic knowledge is expandable and constrained)**.** *A theory can become more ontically informative by expanding its certified tower, but the expansion is constrained by signature preservation and ledger continuity.*

Proof. Adding a bundle or question expands the tower. If the expansion is certified, it must preserve, refine, or explicitly correct earlier signatures while recording the relevant ledger. The growth is therefore real but not arbitrary. □

Concretized in §1125: The ontic-state source is realized as a finite signature-source and residue-ledger NameCert packet.

Concretized in §1154: Ontic residue activation under refined bundles is realized as a finite residue-lattice NameCert packet.

Concretized in §1147: Ontic adequacy audit packages stable source, signature exactness, Real-facing readback, and residue ledger rows as one finite NameCert.

Concretized in §1153: The certified ontic tower is realized as a finite stage-refinement and ledger-continuity NameCert packet.

Concretized in §1152: The ontic residue ledger is realized as a finite signature, observer-access, and residue NameCert packet.

9414.20 Approximation towers and ledgered compression

Physical modelling does not proceed by a single act of representation. It proceeds through a tower of approximations:

$$R \longrightarrow H_n \longrightarrow D_n \longrightarrow M_{\mathrm{disc}} \longrightarrow M_{\mathrm{cont}} \longrightarrow M_{\mathrm{smooth}} \longrightarrow M_{\mathrm{ideal}}.$$

Here R is the physical system, H_n is a finite observation history, D_n is finite data, and the later terms are discrete, continuous, smooth, and ideal representation layers. Each arrow selects features, hides others, and licenses a new visible object. The endpoint may be useful, predictive, and mathematically elegant, but it is not identical to the source. It is the endpoint of certified compression.

Definition 9414.306 (Approximation level)**.** *An approximation level is a pair $L_i = (X_i, \Pi_i)$, where X_i is a domain of systems, traces, data, or representations, and Π_i is the observation or classifier bundle determining which distinctions are visible at that level. A level is therefore a domain together with a visibility rule.*

Definition 9414.307 (Compression arrow)**.** *A compression arrow*

$$C_i : L_i \rightsquigarrow L_{i+1}$$

is a certified transition from one approximation level to the next. It carries source, pattern, classifier, stability, and ledger fields:

$$C_i = (\mathrm{Source}_i, \mathrm{Pattern}_i, \mathrm{Classifier}_i, \mathrm{Stability}_i, \mathrm{Ledger}_i).$$

The fields mirror the naming-certificate template: source material, extracted behaviour, grouping relation, reuse certificate, and record of hidden or compressed material. A compression arrow is therefore not a quotient by fiat; it is a ledgered transition.

Definition 9414.308 (Physical approximation tower)**.** *A physical approximation tower is a finite or extendable sequence*

$$\mathcal{T} : L_0 \xrightarrow{C_0} L_1 \xrightarrow{C_1} \cdots \xrightarrow{C_{n-1}} L_n$$

whose initial level is rooted in a physical system or observation process. It is physically grounded when every mathematical level remains connected to the physical source through the composite ledger.

Definition 9414.309 (Visible endpoint)**.** *The visible endpoint of a tower $\mathcal{T}$ is the object, package, model, or representation appearing at the final level L_n:*

$$\mathrm{End}(\mathcal{T}) \in L_n.$$

The visible endpoint is the current terminal package of the compression chain, not the source itself.

Definition 9414.310 (Tower ledger)**.** *The tower ledger of $\mathcal{T}$ is the composite record*

$$\mathrm{Ledger}(\mathcal{T}) = \mathrm{Ledger}_0 + C_0^* \,\mathrm{Ledger}_1 + \cdots + (C_{n-2} \circ \cdots \circ C_0)^* \,\mathrm{Ledger}_{n-1}\,.$$

It records not only what the last compression hides, but also what every earlier compression hid.

Definition 9414.311 (Tower-complete representation)**.** *A representation $M = \mathrm{End}(\mathcal{T})$ is tower-complete for a physical system R relative to a family of observation bundles $\mathcal{P}$ when M is the endpoint of a physically grounded tower, every compression arrow carries a ledger, every endpoint operation descends through the relevant classifiers, and the endpoint preserves the physical signatures required by $\mathcal{P}$.*

9414.21 Endpoint and composite ledger

Theorem 9414.312 (Endpoint is not source)**.** *Let*

$$\mathcal{T} : L_0 \xrightarrow{C_0} L_1 \xrightarrow{C_1} \cdots \xrightarrow{C_{n-1}} L_n$$

be an approximation tower with $n \geq 1$. Then $\mathrm{End}(\mathcal{T})$ is not identical to the source object in L_0 merely by being produced from it.

Proof. Each compression arrow maps source material into a visible object according to a classifier. A classifier preserves selected distinctions and hides others. Unless the classifier preserves every distinction and the ledger is trivial, the transition is not identity. Since the tower contains at least one compression arrow, the endpoint is produced through selection, packaging, regularization, or idealization. Treating it as source-equivalent requires a separate certificate showing that the relevant physical signatures have been preserved. □

Corollary 9414.313 (Model as terminal package)**.** *If $M = \mathrm{End}(\mathcal{T})$, then M is the terminal visible package of a modelling tower, not the physical system itself.*

Proof. This is immediate from Definition 9414.312: generation through a tower licenses representation, not identity with the source. □

Theorem 9414.314 (Tower induces composite compression)**.** *Every finite approximation tower induces a composite compression*

$$C_{\mathcal{T}} = C_{n-1} \circ \cdots \circ C_1 \circ C_0 : L_0 \rightsquigarrow L_n.$$

Proof. For one arrow, the composite is that arrow. If the first n arrows compose to $C_{0:n} : L_0 \rightsquigarrow L_n$, then composing with $C_n : L_n \rightsquigarrow L_{n+1}$ gives $C_n \circ C_{0:n} : L_0 \rightsquigarrow L_{n+1}$. Induction on tower length gives the result. □

Theorem 9414.315 (Tower ledger is composite)**.** *The ledger of the composite compression induced by a tower is the tower ledger of Definition 9414.310, not merely the ledger of the final arrow.*

Proof. For two arrows $L_0 \xrightarrow{C_0} L_1 \xrightarrow{C_1} L_2$, the composite ledger must record which L_0 sources lie behind each L_1 visible item and which L_1 visible items lie behind each L_2 visible item. Thus it contains the first ledger together with the pullback of the second ledger along the first visible interface. Longer towers are obtained by repeating this two-stage construction, pulling each later ledger back along the already-composed interface. □

Corollary 9414.316 (Endpoint alone is epistemically insufficient)**.** *The endpoint $M = \mathrm{End}(\mathcal{T})$ is insufficient to determine the modelling claim unless the tower ledger is also specified.*

Proof. By Definition 9414.315, the endpoint hides the sequence of source histories, measurements, classifiers, and idealizations that produced it. Without the tower ledger, those dependencies cannot be audited. □

9414.22 Layer-relative validity

Definition 9414.317 (Level-relative fit)**.** *Let $M_i \in L_i$ and $M_j \in L_j$ be representations at different levels of a tower. The later representation has level-relative fit to the earlier one, written*

$$\mathrm{Fit}_{i \to j}(M_i, M_j; \Pi_j),$$

when the compression chain from L_i to L_j preserves the signatures required by Π_j.

Definition 9414.318 (Tower-valid claim)**.** *A claim φ made at the endpoint of a tower is tower-valid when*

$$\mathrm{End}(\mathcal{T}) \models \varphi$$

and every operation used to establish φ descends through the tower classifiers on which the claim depends.

Theorem 9414.319 (Endpoint truth does not imply source truth)**.** *Let φ be true at the visible endpoint L_n of a tower. Then φ does not automatically hold at the source level L_0.*

Proof. A claim true at L_n is evaluated using the classifier, operations, and visible structure available at L_n. The source level may contain distinctions hidden by the compression chain. If φ depends on structure introduced by smoothing, completion, or idealization, the source need not satisfy it. Transfer from endpoint truth to source truth requires a descent or reflection certificate. □

Corollary 9414.320 (Smooth endpoint behaviour need not describe rough source behaviour)**.** *If a smooth ideal model satisfies a differentiability claim, the original physical or discrete source need not satisfy that differentiability claim.*

Proof. Differentiability may be introduced by smoothing, interpolation, completion, or idealization. By Definition 9414.319, endpoint smoothness does not transfer to the source without a descent certificate. □

Corollary 9414.321 (Coarse success does not license refined prediction)**.** *If a model fits observations under a coarse bundle Π, it does not follow that it fits observations under a refined bundle $\Pi' \supseteq \Pi$.*

Proof. A coarse bundle may ignore distinctions exposed by a refined bundle. Fit under Π establishes agreement only at the coarse level. Refined prediction requires an additional stability certificate. □

9414.23 Hidden structure and tower collapse

Definition 9414.322 (Hidden structure)**.** *For a compression arrow $C_i : L_i \rightsquigarrow L_{i+1}$, the hidden structure of C_i is the source material recorded in* Ledger_i *but not visible in the compressed object at L_{i+1}. Schematically,*

$$\mathrm{Hidden}(C_i) := \mathrm{Ledger}_i \setminus \mathrm{Visible}(L_{i+1}).$$

Definition 9414.323 (Reactivation of hidden structure)**.** *Hidden structure is reactivated when a later operation, observation bundle, or physical question depends on material hidden by an earlier compression. A later claim reactivates C_i's hidden structure when its truth cannot be determined from the visible endpoint alone and requires inspection of* Ledger_i.

Theorem 9414.324 (Hidden structure can become active under later operations)**.** *Let $\mathcal{T}$ be an approximation tower, and suppose C_i hides source structure S_i. If a later operation F depends on S_i, then F is not well defined on the endpoint unless the ledger for S_i is available or a descent certificate shows that S_i is irrelevant.*

Proof. If F depends on S_i, two sources that agree at the visible endpoint but differ in S_i may yield different values under F. Thus F cannot be evaluated from the endpoint alone. Well-definedness is restored either by inspecting the ledger or by proving that all variations in S_i produce equivalent outputs under F. □

Corollary 9414.325 (Singularities can return)**.** *A singularity removed by smoothing may remain physically relevant if later operations depend on the removed singular structure.*

Proof. The removed singularity is hidden structure. If curvature extraction, stress analysis, scattering prediction, or boundary-condition evaluation depends on it, Definition 9414.324 applies. □

Definition 9414.326 (Tower collapse)**.** *Tower collapse occurs when a claim valid at level L_j is asserted at level L_i with $i < j$ without a certificate showing that the claim descends through the compression chain*

$$L_i \longrightarrow L_{i+1} \longrightarrow \cdots \longrightarrow L_j.$$

Theorem 9414.327 (Tower collapse is invalid without descent)**.** *If a claim φ at level L_j is asserted at a lower level L_i without a descent certificate through the chain from L_i to L_j, then the assertion is not licensed.*

Proof. A claim at L_j is evaluated using the visible structure of L_j. The lower level may contain distinctions hidden in the transition to L_j. If φ is sensitive to those distinctions, it may fail at L_i although it holds at L_j. A descent certificate is exactly the proof that φ is insensitive to the hidden distinctions. □

Corollary 9414.328 (Effective theories are tower-local)**.** *An effective physical theory is valid only within the tower region for which its source, classifier, stability, and ledger certificates hold.*

Proof. An effective theory compresses lower-level structure into higher-level variables and laws. By Definition 9414.327, its claims cannot be transferred beyond the certified compression region without descent. □

9414.24 Polygon-circle tower and tower equivalence

The polygon-circle example can be read as a tower:

$$R \longrightarrow H_n \longrightarrow \text{pixel data} \longrightarrow P_N \longrightarrow C \longrightarrow C^\infty\text{-circle}.$$

At the pixel or coarse-shape level, a high-sided polygon may be indistinguishable from a circle. At the smooth-boundary level, every finite polygon retains vertices, while the smooth circle does not. The assertion $P_N \sim C$ is therefore meaningful only after the level and classifier have been specified.

Theorem 9414.329 (Polygonal approximation is tower-relative)**.** *Let P_N be a polygonal representation and C a smooth circle representation. The claim $P_N \approx C$ is meaningful only relative to a specified approximation level and classifier.*

Proof. At a coarse level, the classifier may compare area, perimeter, or finite-resolution outline; under such a classifier, P_N may approximate C. At a smooth-boundary level, the classifier compares tangent existence, tangent continuity, or curvature distribution; under that classifier, P_N and C differ because P_N has vertices and C has a smooth boundary. The same approximation claim therefore changes content with the tower level. □

Corollary 9414.330 (Differentiation detects tower collapse)**.** *Differentiation can expose an illicit tower collapse because it may depend on structure hidden by coarse approximation.*

Proof. Coarse approximation may hide vertices. Differentiation depends on local tangent behaviour, which vertices disrupt. Applying differentiation after treating P_N and C as identical can therefore expose a mismatch between the visible endpoint and the hidden source. □

Definition 9414.331 (Tower equivalence)**.** *Two towers $\mathcal{T}$ and $\mathcal{T}'$ are tower-equivalent relative to Π when their endpoints are equivalent under Π, their ledgers record equivalent source histories and hidden structures, and every operation used under Π descends through both towers in compatible ways.*

Theorem 9414.332 (Endpoint equivalence does not imply tower equivalence)**.** *Endpoint equivalence*

$$\mathrm{End}(\mathcal{T}) \sim_{\Pi} \mathrm{End}(\mathcal{T}')$$

does not imply tower equivalence.

Proof. Endpoint equivalence compares only final visible objects. It ignores source histories, intermediate classifiers, hidden residues, and stability certificates. Two towers may therefore have equivalent endpoints while having different ledgers or different descent behaviour for later operations. Tower equivalence requires those additional data. □

Corollary 9414.333 (Agreement now can diverge later)**.** *Two models may produce the same current predictions and nevertheless diverge under refined observation or new operations.*

Proof. Current prediction agreement is endpoint equivalence under the present bundle. By Definition 9414.332, this does not imply tower equivalence. Different hidden residues or descent certificates may be exposed by later refinements. □

Concretized in §1142: Ledgered tower compression is realized as a finite NameCert carrier with composite-ledger and non-escape rows.

Concretized in §1143: The tower-equivalence packet records endpoint agreement together with ledger and descent compatibility as a finite NameCert surface.

Concretized in §1162: Composite tower ledgers are realized as a finite NameCert carrier whose classifier preserves ledger routes under tower-arrow composition.

Concretized in §1176: CompressionDescentAuditUp realizes endpoint-operation descent as a finite tower-route, composite-ledger, failure-surface, transport, provenance, and NameCert packet.

Concretized in §1195: Endpoint/source reflection is realized as a finite NameCert packet for endpoint rows, source rows, composite ledgers, descent audits, obstruction rows, and non-escape.

Concretized in §7688: Finite NameCert realization of the approximation tower, compression-arrow, endpoint, and composite-ledger surface.

9414.25 Physical induction and open continuation

An approximation tower makes a model usable only by certification. The next question is how an endpoint can be reused beyond the observations that licensed it. This is the problem of induction. Mathematical induction is licensed by a closed generated carrier. Physical induction is different: future observation is not already present as a completed carrier inside the current finite trace. The observer has a finite history, a model extracted from that history, and a stability certificate claiming that the model remains adequate under specified continuations.

Definition 9414.334 (Closed mathematical carrier)**.** *A closed mathematical carrier is a domain A equipped with constructors such that every element of A is generated by finitely many applications of those constructors, and no further elements are admitted except by the stated generation rules. A proof principle over A is closed when it reasons only over this generated carrier.*

Definition 9414.335 (Open observation carrier)**.** *An open observation carrier for a physical system R is an extendable family*

$$\mathcal{H}(R) = \{H_0, H_1, H_2, \ldots\}$$

of possible finite observation histories, where each H_n is finite and no internal final object is assumed to contain all admissible future histories.

Definition 9414.336 (Observational prefix)**.** *For finite histories $H_n = (o_0, \ldots, o_n)$ and H_m, H_n is an observational prefix of H_m, written*

$$H_n \preceq H_m,$$

when $m \geq n$ and the first $n+1$ observations of H_m agree with H_n.

Definition 9414.337 (Predictive continuation)**.** *A predictive continuation of a model M from a finite fit $\mathrm{Fit}_n(M, R; \Pi)$ is a licensed claim*

$$\mathrm{PredFit}_{n \to n+k}(M, R; \Pi)$$

or an analogous claim over an explicitly stated class of continuations. It is not a theorem of finite fit alone; it requires a stability certificate.

Definition 9414.338 (Physical induction certificate)**.** *A physical induction certificate is a stability certificate*

$$\mathrm{PhysInd}(M, R; \Pi, \mathcal{C})$$

for a class $\mathcal{C}$ of admissible continuations of the current observation history. It licenses

$$\mathrm{Fit}_n(M, R; \Pi) \rightsquigarrow \mathrm{PredFit}_{n \to \mathcal{C}}(M, R; \Pi).$$

Definition 9414.339 (Induction ledger and failure surface)**.** *The induction ledger attached to $\mathrm{PhysInd}(M, R; \Pi, \mathcal{C})$ is*

$$\mathrm{IndLedger}(M, R; \Pi, \mathcal{C}) = (H_n, \Pi, \mathcal{C}, \mathrm{FitCert}_n, \mathrm{StabilityBasis}, \mathrm{FailureSurface}).$$

Its failure surface is the class of continuations that would separate model-generated and observed signatures:

$$\mathrm{Fail}(M, R; \Pi, \mathcal{C}) = \{\, H' \in \mathcal{C} \mid \mathrm{ModSig}_\Pi(M \mid H') \not\sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H') \,\}.$$

9414.26 Mathematical and physical induction

Theorem 9414.340 (Mathematical induction requires closed carrier)**.** *A mathematical induction principle is licensed only when its target carrier is closed under the constructors used in the induction.*

Proof. An induction proof shows that a property holds at the base constructors and is preserved by each generating constructor. This proves the property for all elements only because every element of the carrier is generated by those constructors. If the carrier were not closed, an ungenerated element could exist outside the reach of the induction step. Thus induction requires closed generation. □

Corollary 9414.341 (Induction is a closure theorem)**.** *Mathematical induction is not repeated verification. It is a theorem about a closed generated domain.*

Proof. Repeated successful cases do not exclude ungenerated cases. The decisive premise is that the carrier has no elements outside the constructor closure. □

Theorem 9414.342 (Physical induction is not mathematical induction)**.** *The inference from finite observational success to future predictive success is not licensed by mathematical induction alone.*

Proof. Let $\mathrm{Fit}_n(M, R; \Pi)$ hold. This says that M agrees with R on the finite prefix H_n under Π. Future success would require a principle covering admissible continuations of H_n. But the future observation carrier is open: it is not given as a closed carrier whose elements are generated by a known successor rule that preserves the model. Therefore the transition from finite fit to future prediction requires an additional stability certificate. That certificate may be physically justified, experimentally supported, or theoretically motivated, but it is not mathematical induction over a closed carrier. □

Corollary 9414.343 (Repeated success is not necessity)**.** *No finite sequence of successful predictions entails that a model must succeed on every future admissible observation.*

Proof. A finite sequence of successes extends fit to a larger prefix. By Definition 9414.342, no finite prefix closes the open carrier of future observations. □

Corollary 9414.344 (Physical induction is certificate-relative)**.** *A physical inductive inference is valid only relative to the continuation class and stability certificate that license it.*

Proof. Definition 9414.338 includes a continuation class $\mathcal{C}$. The certificate licenses prediction over that class. A different continuation class may expose hidden residue or violate the stability basis. □

9414.27 Law certificates, reuse, and descent

Definition 9414.345 (Law certificate)**.** *A law certificate for a model M of a system R relative to Π is*

$$\mathrm{LawCert}(M, R; \Pi) = (\mathrm{SourceFit}, \mathrm{Pattern}, \mathrm{Classifier}, \mathrm{Stability}, \mathrm{FailureSurface}, \mathrm{Ledger}).$$

It records the finite observations supporting the law, the claimed invariant or regularity, the equivalence under which fit is judged, the continuation principle licensing reuse, the observations that would defeat the law, and the hidden assumptions or tower levels.

Definition 9414.346 (Lawlike stability)**.** *A model M has lawlike stability for R under Π and continuation class $\mathcal{C}$ when finite fit holds and a law certificate licenses*

$$\forall H' \in \mathcal{C},\ \mathrm{PredFit}(M, R; \Pi \mid H').$$

The universal statement is certificate-bounded, not an oracle over all possible reality.

Theorem 9414.347 (Physical law is not merely equation)**.** *A physical law requires a law certificate. An equation alone is insufficient.*

Proof. An equation supplies formal syntax and a calculation rule. A physical law must also specify the observations from which it is licensed, the physical pattern it preserves, the classifier comparing model and reality, the stability condition for reuse, the observations that would count as failure, and the ledger of idealizations or hidden assumptions. Without these fields, the equation is not tied to physical reality as an open observation process. □

Corollary 9414.348 (Same equation may carry different laws)**.** *The same mathematical equation may express different physical laws when attached to different source histories, classifiers, stability domains, or failure surfaces.*

Proof. By Definition 9414.345, the equation-like pattern is only one field. Different source fit, classifier, stability, failure surface, or ledger data determine different law certificates. □

Corollary 9414.349 (Law without failure surface is oracle claim)**.** *If a claimed law has no specified failure surface, it is not a scientific law certificate but an unbounded oracle assertion.*

Proof. A law certificate must specify observations or refinements that would separate model from reality. Removing this surface removes empirical separability and turns the claim into a final-fit oracle, which the present framework rejects. □

Definition 9414.350 (Predictive reuse and descent)**.** *Predictive reuse applies a model fitted on H_n to a later or counterfactual history H' with $H_n \preceq H'$ or $H' \in \mathcal{C}$. A prediction operation F has predictive descent from the observed classifier $\sim_n$ to the continuation classifier $\sim_{\mathcal{C}}$ when*

$$M_1 \sim_n M_2 \quad \Longrightarrow \quad F(M_1) \sim_{\mathcal{C}} F(M_2).$$

Theorem 9414.351 (Prediction requires reuse stability and descent)**.** *A prediction from a finite-fit model is licensed only when the model has reuse stability over the relevant continuation class and the prediction operation descends through the classifier used to form the model.*

Proof. Prediction applies the model beyond the finite trace on which it was fitted. Without reuse stability, finite fit gives no reason to expect future fit. Since the model was formed by classification and compression, a prediction operation that does not descend through that classifier may depend on hidden representative data rather than on the visible model. Both reuse stability and descent are therefore required. □

Corollary 9414.352 (Predictive failure can be non-descent)**.** *A model may fail predictively because the prediction operation does not descend through the approximation that produced the model, even when the visible equation is internally correct.*

Proof. By Definition 9414.351, prediction requires descent. If descent fails, the failure lies in transferring the equation through an invalid compression, not necessarily in the equation's algebraic form. □

Concretized in §1183: Continuation countermodels realize finite predictive-reuse defeat as a displayed continuation, mismatch, stability, failure-surface, and NameCert packet.

Concretized in §6534: Physical induction certificates become finite continuation-stability and failure-surface NameCert rows.

9414.28 Open empirical continuation

Definition 9414.353 (Local empirical closure)**.** *Local empirical closure is the capacity, for a fixed finite trace H_n, bundle Π, and classifier $\sim_\Pi$, to determine or certify whether* $\mathrm{Fit}_n(M, R; \Pi)$ *holds.*

Definition 9414.354 (Global empirical openness)**.** *Global empirical openness is the absence of any internal finite procedure supplying a final certificate for*

$$\forall H,\ \mathrm{Fit}(M, R; \Pi \mid H)$$

over all admissible actual and counterfactual observation histories.

Definition 9414.355 (Final induction oracle)**.** *A final induction oracle is a hypothetical judgment* $\Omega_{\mathrm{ind}}(M, R; \Pi)$ *implying*

$$\forall H,\ \mathrm{Fit}(M, R; \Pi \mid H).$$

It would convert finite or current model success into unrestricted future validity.

Theorem 9414.356 (Local closure does not imply global closure)**.** *A finite inquiry may close fit judgments at each observed stage without closing the universal judgment over all future stages.*

Proof. Local empirical closure concerns a fixed finite trace and observed data. Global empirical closure quantifies over observations not yet performed and refinements not yet introduced. No finite local fit judgment contains those future traces. □

Corollary 9414.357 (Science requires local closure and global openness)**.** *Scientific inquiry requires both the ability to settle finite tests and the refusal to close final fit.*

Proof. If local closure fails, experiments cannot decide whether a model fits current observations. If global openness fails, the theory becomes immune to future correction. □

Theorem 9414.358 (Final induction oracle collapses inquiry)**.** *If a final induction oracle is admitted, physical inquiry loses its open continuation structure.*

Proof. A final induction oracle asserts that a model will fit all admissible future observations. This removes the failure surface and closes the continuation horizon. Future observations can no longer genuinely revise the model, so open inquiry is replaced by a meta-level assertion of total validity. □

9414.29 Falsification, intuition, and tower induction

Definition 9414.359 (Finite refutation and open confirmation)**.** *A finite refutation of M relative to R and Π is a finite trace H_n such that*

$$\mathrm{ModSig}_\Pi(M \mid H_n) \not\sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H_n).$$

Open confirmation is an extendable family of finite fit certificates together with a physical induction certificate licensing specified continuations.

Theorem 9414.360 (Falsification finite, confirmation open)**.** *A model can be refuted by a finite observation, but it cannot be finally confirmed by finitely many observations.*

Proof. A finite refutation is one positive witness of mismatch. Final confirmation would require agreement over all admissible traces, and no finite collection of traces exhausts the open observation carrier. □

Definition 9414.361 (Stability prior and certified intuition)**.** *A stability prior is a pre-certificate expectation that an observed pattern will persist under a specified class of continuations. It becomes certified intuition when upgraded to a physical induction certificate by specifying the stability basis and failure surface.*

Theorem 9414.362 (Physical intuition is not oracle)**.** *A stability prior does not license prediction unless it is converted into a physical induction certificate.*

Proof. A stability prior expresses expectation. Prediction requires a licensed transition from finite fit to future fit. Without a physical induction certificate, the prior does not specify the continuation class, classifier, descent conditions, or failure surface. □

Definition 9414.363 (Tower induction)**.** *For a tower $\mathcal{T} : L_0 \to L_1 \to \cdots \to L_n$, tower induction is a physical induction certificate licensing prediction at level L_j from fit established at level L_i, with $i \leq j$, together with descent or reflection data relating the relevant levels.*

Theorem 9414.364 (Cross-level prediction requires tower induction)**.** *If a prediction is made at a different tower level from the level at which fit was established, then a tower induction certificate is required.*

Proof. Fit at L_i is judged by the classifier and visible signatures of L_i. Prediction at L_j may use a different classifier, visible structure, and hidden residue. The finite fit at L_i does not by itself license the prediction at L_j. Tower induction supplies the missing transfer data: signature preservation between levels, descent of operations, and inactivity or ledgered handling of hidden structure. □

Corollary 9414.365 (Effective laws are level-indexed induction certificates)**.** *An effective law is valid only within the approximation level and continuation class for which its induction certificate holds.*

Proof. Effective laws compress lower-level structure into higher-level variables. Moving across levels requires the certification described in Definition 9414.364. □

Theorem 9414.366 (Physical induction has closed-ground and open-continuation form)**.** *Physical induction requires local closure of finite fit judgments and global openness of future observation.*

Proof. Local finite fit must be decidable or certifiable; otherwise the model cannot be compared with the observed trace. The future observation horizon cannot be closed by the same finite procedure. If it were closed, the model would acquire a final-fit oracle over all future observations. Physical induction therefore has the form of closed finite fit together with open continuation, parallel to BEDC's closed-ground and open-meta polarity. □

Concretized in §1135: Physical induction is realized as a finite continuation, stability, failure-surface, ledger, and NameCert packet.

Concretized in §2380: Physical induction is realized as a finite observation-step NameCert packet.

Concretized in §1312: Local empirical closure is realized as a finite NameCert packet whose non-finality row keeps global empirical openness visible.

Concretized in §1139: Physical induction stability seals realize finite-fit reuse as continuation-class, failure-surface, transport, provenance, and NameCert rows.

Concretized in §1155: LawCertificateUp realizes physical law as a finite source-fit, classifier, continuation-stability, failure-surface, ledger, transport, provenance, and NameCert packet.

Concretized in §1182: Predictive descent is realized as a finite NameCert packet binding finite fit, prediction operation, observed classifier, continuation classifier, stability scope, failure surface, and non-escape rows.

Concretized in §6858: Local empirical residue is realized as a finite observation-history, classifier, failure-surface, and nonfinality NameCert packet.

9414.30 Objectivity without a privileged observer

Physical induction leaves inquiry locally closed and globally open. This changes the meaning of objectivity. If no observer stands outside the inquiry process, and if no final-fit oracle is internally available, objectivity cannot be access to reality from nowhere. It must be reconstructed as invariant stability across admissible observer histories, observation bundles, and continuation procedures.

This is the physical reading of the multi-Hist layer. A multi-Hist configuration contains several local histories without a required common base, clock, or time-step indexing. Since an observer is identified with the history of recorded distinctions, there is no observer-substance behind the record and no privileged frame above the configuration. Frame-independent physical content must therefore live in inter-history invariants.

Definition 9414.367 (Observer history)**.** *An observer history is a finite or extendable history $h \in \mathsf{Hist}$ interpreted as the record of distinctions, observations, continuations, and classifier-mediated signatures available to an observer. The observer is not an additional substance behind h; it is the history considered as a finite distinction record.*

Definition 9414.368 (Multi-observer configuration)**.** *A multi-observer configuration is a finite family*

$$\mathcal{H} = (h_1, \ldots, h_m)$$

of observer histories considered together without assuming a shared origin, shared clock, shared time-step indexing, or privileged synchronizing observer.

Definition 9414.369 (Privileged observer)**.** *A privileged observer for $\mathcal{H}$ is a hypothetical observer history $h_\star$ that synchronizes, orders, and evaluates every $h_i \in \mathcal{H}$ from a distinguished standpoint. Equivalently, it would supply a global observational frame*

$$h_\star : (h_1, \ldots, h_m) \longmapsto \textit{one shared empirical ordering.}$$

The framework does not assume such a history.

Definition 9414.370 (Inter-observer relation and invariant)**.** *An inter-observer relation is a relation $R(h_i, h_j)$ whose evidence is obtained through finite observation, continuation, classifier, or ledger data available within the relevant histories. An inter-observer invariant is a function or predicate $I(h_i, h_j)$ whose value is stable under change of observational anchor:*

$$I(h_i, h_j) \sim I(h_j, h_i)$$

up to the relevant history and signature classifiers. It may depend on relations between histories, but it may not privilege one history as the absolute frame.

Definition 9414.371 (Relational objectivity)**.** *A claim φ is relationally objective over $\mathcal{H}$ when there is an inter-observer invariant I such that φ is determined by I rather than by one observer history alone:*

$$\mathrm{Obj}_{\mathcal{H}}(\varphi) \quad \Longleftrightarrow \quad \exists I \, (I \text{ is inter-observer invariant} \wedge I \Rightarrow \varphi).$$

Objectivity is therefore not observer-free; it is observer-invariant.

Definition 9414.372 (Private and frame-independent content)**.** *A claim φ is private observational content of h_i when it is determined only by the internal trace of h_i and no inter-observer invariant transports it across the configuration. A claim is frame-independent physical content when it is relationally objective over every admissible multi-observer configuration in which it is asserted.*

9414.31 No global observer

Theorem 9414.373 (No member is privileged by membership)**.** *No element of a multi-observer configuration is privileged merely by being a member of the configuration.*

Proof. By Definition 9414.368, the configuration supplies only a finite family of local histories. Membership does not give a shared origin, clock, time-step indexing, or synchronization map. Thus $h_i \in \mathcal{H}$ gives the local observational trace of h_i, but not authority to order or evaluate all other histories from a global frame. □

Corollary 9414.374 (Single histories are not frame-independent carriers)**.** *An individual observer history may carry valid local content, but it does not by itself carry frame-independent physical content.*

Proof. By Definition 9414.373, no observer history is privileged by default. A claim contained only in one history is therefore local to that history until transported by an inter-observer invariant. □

Theorem 9414.375 (Adding observers does not create an external view)**.** *Extending $\mathcal{H} = (h_1, \ldots, h_m)$ to $(h_1, \ldots, h_m, h_{m+1})$ does not by itself produce a privileged observer.*

Proof. The added history is itself an observer history with its own finite trace and local continuation structure. It does not automatically contain the classifiers, ledgers, or time-step orderings of the other histories. If a candidate observer-of-observers is represented internally, it becomes another history and remains subject to the same locality constraints. □

Corollary 9414.376 (No internal observer-of-observers)**.** *There is no observer-of-observers inside the same observational layer.*

Proof. An internal candidate is added as another history. By Definition 9414.375, such addition does not create global synchronization. □

9414.32 Objectivity as invariance

Definition 9414.377 (Anchor change)**.** *An anchor change is the redescription of an inter-observer relation from the standpoint of another observer history:*

$$(h_i \text{ as anchor}) \rightsquigarrow (h_j \text{ as anchor}).$$

A quantity is stable under anchor change when its value remains classifier-equivalent after redescription.

Definition 9414.378 (Anchor-stable claim)**.** *A claim φ is anchor-stable over $\mathcal{H}$ when, for all $h_i, h_j \in \mathcal{H}$,*

$$\varphi^{h_i} \sim \varphi^{h_j},$$

where φ^{h_i} denotes the expression of φ with h_i as observational anchor.

Definition 9414.379 (Invariant certificate)**.** *An invariant certificate for φ over $\mathcal{H}$ is*

$$\mathrm{InvCert}(\varphi; \mathcal{H}) = (\mathrm{HistSources}, \mathrm{RelationSpec}, \mathrm{ClassifierSpec}, \mathrm{AnchorStability}, \mathrm{LedgerPolicy}).$$

It records the histories being related, the relation claimed, the sameness classifier, the proof of anchor stability, and the ledger of observations and transformations used in the comparison.

Concretized in §1150: Anchor-stability is realized as a finite certificate for invariant transport across observer anchors.

Theorem 9414.380 (Objectivity requires anchor stability)**.** *If a claim is frame-independent physical content, then it is anchor-stable.*

Proof. Frame-independent physical content is stable under changes of observer anchor by Definition 9414.372. Anchor stability is precisely the classifier-equivalent preservation of the claim under such changes. □

Theorem 9414.381 (Invariants generate relational objectivity)**.** *If $I(h_i, h_j)$ is an inter-observer invariant over $\mathcal{H}$, then any claim determined solely by I is relationally objective over $\mathcal{H}$.*

Proof. The invariant is stable under anchor change. A claim determined solely by it is therefore also stable under anchor change. By Definition 9414.371, such a claim is relationally objective. □

Corollary 9414.382 (Objectivity is not local certainty)**.** *A claim may be certain within one observer history and still fail to be objective over a multi-observer configuration.*

Proof. Local certainty concerns one trace. Objectivity requires anchor stability across histories by Definition 9414.380. □

9414.33 Relational physical content

Definition 9414.383 (Local coordinate content)**.** *A quantity q_i is local coordinate content for h_i when it is defined only relative to the local description carried by h_i, such as a local time coordinate, chart coordinate, gauge representative, or measurement ordering.*

Definition 9414.384 (Relational physical content)**.** *A quantity Q is relational physical content when it is produced from local coordinate contents by an inter-observer invariant,*

$$Q = I(q_i, q_j),$$

and is stable under anchor change.

Definition 9414.385 (Gauge-like redundancy)**.** *A difference between local descriptions is gauge-like redundancy when it changes local coordinate content but preserves all relational physical content. Thus q_i and q_i' are gauge-like equivalent when, for every admissible inter-observer invariant I and compatible q_j,*

$$I(q_i, q_j) \sim I(q_i', q_j).$$

Theorem 9414.386 (Local difference requires invariant separation)**.** *If every admissible inter-observer invariant gives the same value on local descriptions q_i and q_i', then their difference is not a frame-independent physical difference.*

Proof. Frame-independent physical content is determined by inter-observer invariants. If all such invariants agree on q_i and q_i', no relationally objective claim separates them. The difference may remain meaningful as local coordinate content, but it is not physical content at the invariant level. □

Corollary 9414.387 (Coordinate freedom is not empirical ambiguity)**.** *Different coordinate descriptions may represent the same physical content when they preserve the same inter-observer invariants.*

Proof. By Definition 9414.386, local coordinate differences that do not change invariants are not frame-independent physical differences. □

Corollary 9414.388 (Empirical disagreement requires invariant separation)**.** *Two observers genuinely disagree about physical content only when their records cannot be reconciled by an invariant certificate.*

Proof. A local mismatch may be coordinate-like or private to one history. It becomes physical disagreement only when an admissible invariant separates the records. □

9414.34 Local time and synchronization

Definition 9414.389 (Local time stream)**.** *The local time stream of an observer history h is the ordered sequence of continuation steps by which h grows. If*

$$h_0 \to h_1 \to \cdots \to h_n,$$

then the local time stream is the finite order encoded by this chain.

Definition 9414.390 (Global clock and synchronization certificate)**.** *A global clock for $\mathcal{H}$ is a single ordering object T into which every $h_i \in \mathcal{H}$ embeds while preserving all relevant local orders and supplying a privileged shared present. A synchronization certificate between h_i and h_j is a finite ledgered construction*

$$\mathrm{SyncCert}(h_i, h_j)$$

that relates selected local time steps of h_i to selected local time steps of h_j. It is a local relation, not a global clock.

Theorem 9414.391 (Local time does not imply global time)**.** *The existence of local time streams for observer histories does not imply the existence of a global clock for the configuration.*

Proof. Each observer history has its own order of continuation. A global clock requires a single ordering object synchronizing all histories and supplying a privileged shared present. No such object follows from the mere existence of local orders. Temporal comparison requires an additional synchronization certificate. □

Corollary 9414.392 (Temporal comparison is relational)**.** *A temporal comparison between observer histories must be expressed as an inter-observer relation, not as a reading from a global clock.*

Proof. By Definition 9414.391, local time streams do not supply a global clock. Any cross-history temporal comparison must therefore be certified as a relation between histories. □

Corollary 9414.393 (Simultaneity is certificate-relative)**.** *A simultaneity claim between observer histories is meaningful only relative to a synchronization certificate.*

Proof. Simultaneity relates local time streams. Since no global clock is assumed, the relation must be supplied by a certificate connecting the relevant local steps. □

9414.35 Observer-independent constants

Definition 9414.394 (Observer-independent quantity)**.** *A quantity Q is observer-independent over $\mathcal{H}$ when, for all $h_i, h_j \in \mathcal{H}$,*

$$Q^{h_i} \sim Q^{h_j},$$

where Q^{h_i} denotes the value computed with h_i as anchor.

Definition 9414.395 (Maximum relational rate)**.** *A maximum relational rate $v_{\max}$ is a quantity over a multi-observer configuration such that all admissible cross-history causal dependences are bounded by $v_{\max}$ and $v_{\max}$ is invariant under change of observer anchor. The definition names a structural role and does not assign a numerical value.*

Theorem 9414.396 (Observer-independent constants are invariants)**.** *If a quantity has the same classifier-equivalent value for every observer anchor in a multi-observer configuration, then it is an inter-observer invariant.*

Proof. An inter-observer invariant is stable under change of observer anchor. The hypothesis states exactly that changing the anchor preserves the value up to the relevant classifier. □

Corollary 9414.397 (Invariant maximum rate is frame-independent)**.** *If a maximum relational rate is observer-independent, then it is frame-independent physical content.*

Proof. By Definition 9414.396, observer-independence implies inter-observer invariance. Frame-independent physical content is carried by such invariants. □

Corollary 9414.398 (Structural form and empirical value are distinct)**.** *A framework may derive the form of an observer-independent maximum rate without deriving its empirical value.*

Proof. The structural form is the invariance of a maximum relational rate under observer-anchor change. Its numerical value requires empirical calibration within a physical theory and observation bundle. □

9414.36 Tower-aware objectivity

Definition 9414.399 (Tower-indexed invariant)**.** *For an approximation tower*

$$\mathcal{T} : L_0 \to L_1 \to \cdots \to L_n,$$

a tower-indexed invariant is an inter-observer invariant I_k defined at a specified level L_k. It is not automatically valid at other levels.

Definition 9414.400 (Cross-level invariant descent)**.** *A tower-indexed invariant I_j at L_j descends to L_i, with $i < j$, when a certificate shows that the invariant value at L_j is determined solely by structures preserved from L_i. We write*

$$\mathrm{DescInv}_{j \to i}(I_j).$$

Theorem 9414.401 (Invariance is tower-relative without descent)**.** *Invariance at one approximation level does not automatically imply invariance at another level.*

Proof. Each tower level has its own visible structures, classifiers, and hidden residues. An invariant at one level may depend on structures introduced by compression, smoothing, completion, or idealization. Another level may expose distinctions suppressed there or lack structures that the invariant requires. Transfer therefore requires cross-level invariant descent. □

Corollary 9414.402 (Objectivity is level-relative unless descent is supplied)**.** *A physical claim may be objective at one approximation level without yet being objective across the entire tower.*

Proof. Objectivity is defined by invariance. By Definition 9414.401, invariance does not automatically move across tower levels. □

Corollary 9414.403 (Effective objectivity is legitimate but local)**.** *An effective theory may contain objective quantities within its certified domain even when those quantities are not globally final.*

Proof. If a quantity is invariant within the theory's level and continuation domain, it is objective there. The same fact does not supply global descent across all levels. □

9414.37 Objectivity without finality

Definition 9414.404 (Objective but open claim)**.** *A claim φ is objective but open when it has an invariant certificate over the currently admissible observer histories and tower level, and it has a specified failure surface under later refinement, added observers, or tower-level change.*

Theorem 9414.405 (Objectivity does not imply finality)**.** *If φ is relationally objective over $\mathcal{H}$, it does not follow that φ is final over all later configurations, refinements, or observation bundles.*

Proof. Relational objectivity over $\mathcal{H}$ means that φ is determined by an inter-observer invariant within that configuration. Later inquiry may extend the configuration, refine the observation bundle, or move to another approximation level. Such changes may expose hidden residues or require new invariant certificates. □

Corollary 9414.406 (Objectivity is compatible with revisability)**.** *A claim may be objectively established relative to current invariants while remaining open to later correction.*

Proof. By Definition 9414.405, objectivity does not entail finality. Revisability concerns the possibility that later observations or refinements alter the certified invariant structure. □

Corollary 9414.407 (Final objectivity would require an oracle)**.** *A claim of final objectivity over all observers, all refinements, and all future histories would require a final-fit oracle or final synchronization oracle.*

Proof. Final objectivity would require a certificate covering every possible observer history, observation bundle, tower refinement, and continuation. Such a certificate would function as a global oracle over the total observational process, which the framework rejects. □

Remark 9414.408 (Compressed form)**.** *The replacement is not*

$$\textit{objective truth} = \textit{truth from nowhere},$$

but

$$\textit{objective truth} = \textit{invariant, ledgered, anchor-stable content across observer histories.}$$

The absence of a privileged observer therefore does not erase objectivity. It requires every objective claim to state the histories related, the classifier under which the relation is evaluated, the invariant that survives anchor change, and the ledger recording how the comparison was made.

Concretized in §1133: Relational objectivity is realized as a finite multi-observer invariant certificate with anchor-stability and no privileged observer row.

Concretized in §1166: Anchor-change invariants are realized as finite symmetric transport certificates over observer-history anchors.

Concretized in §1134: Objectivity refutation boundaries are realized as finite failed-invariant, failed-anchor-stability, witness, refusal-ledger, transport, provenance, and NameCert rows.

Concretized in §1189: Synchronization certificates realize local temporal comparison as finite paired HistTimeStream rows, selected-step correspondences, no-global-clock refusal, hsame transport, Cont replay, Pkg provenance, and NameCert data.

9414.38 The observation–reflection equivalence thesis

The preceding sections establish the route to the equivalence thesis. Physical fit is finite and open. Mathematical approximation does not imply physical adequacy. Ontic state is the stable source of reality-facing signatures. Physical modelling proceeds through ledgered approximation towers. Physical induction is a stability certificate over open continuation. Objectivity is invariant stability across observer histories rather than a privileged view.

The central claim is structural:

physical inquiry and MetaCIC self-study share the same reflective architecture.

This does not identify physical reality with a formal system, nor MetaCIC with a physical object. It says that both operate through finite histories, signature extraction, classifier-mediated packaging, certificate-based stabilization, ledgered compression, and open continuation.

The BEDC kernel already displays this pattern:

$$b_0, b_1 \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{Gap} \to \mathsf{NameCert}.$$

Derived interfaces are not primitive exports. They are licensed by certificates whose fields record source, pattern, classifier, stability, and ledger. Physical modelling has the same epistemic form: a model is licensed by source traces, extracted patterns, physical classifiers, stability certificates, and ledgers recording what the model hides.

Definition 9414.409 (Reflective inquiry architecture)**.** *A reflective inquiry architecture is a tuple*

$$\mathcal{R} = (S, H, \Sigma, \mathcal{C}, \mathcal{P}, \mathcal{L}, \mathcal{E}),$$

where S is a source domain, H is a finite-history carrier generated from the source, Σ extracts signatures, $\mathcal{C}$ classifies signatures into visible sameness classes, $\mathcal{P}$ is a package, model, or derived interface licensed by the classifier, $\mathcal{L}$ is a ledger recording hidden source material, and $\mathcal{E}$ is an open continuation procedure. Its generic chain is

$$S \to H \to \Sigma(H) \to \mathcal{C}(\Sigma) \to \mathcal{P} \to \mathcal{L} \to \mathcal{E}.$$

The architecture is reflective when it can use finite histories and certificates to generate higher-level descriptions of its own behaviour while still lacking an internal final truth oracle.

Definition 9414.410 (Physical inquiry instance)**.** *A physical inquiry instance of reflective architecture is obtained by setting S to a reality-facing source, H to finite observation traces, Σ to observed physical signatures, $\mathcal{C}$ to the model–data classifier, $\mathcal{P}$ to a physical model or law certificate, $\mathcal{L}$ to the experimental and approximation ledger, and $\mathcal{E}$ to the future observation horizon. Its chain is*

$$R \to H_n(R) \to \mathrm{ObsSig}_{\Pi}(R \mid H_n) \to \mathrm{Fit}_n(M, R; \Pi) \to \mathrm{LawCert}(M, R; \Pi) \to \mathrm{Ledger} \to \mathrm{OpenFit}\,.$$

Definition 9414.411 (MetaCIC self-study)**.** *A MetaCIC self-study instance of reflective architecture is obtained by setting S to formal generative behaviour, H to generated* Hist *values, Σ to signatures extracted by Ask/Sig interfaces, $\mathcal{C}$ to licensed classifiers such as* hsame, sameSig, *or derived classifiers, $\mathcal{P}$ to packages or derived interfaces, $\mathcal{L}$ to gap and NameCert ledgers, and $\mathcal{E}$ to open meta-level continuation. Its chain is*

$$\mathsf{Hist} \to \mathsf{Cont} \to \mathsf{Ask}/\mathsf{Sig} \to \textit{classifier} \to \mathsf{Pkg}/\mathsf{NameCert} \to \textit{ledger} \to \mathrm{OpenMeta}\,.$$

Definition 9414.412 (Observation–reflection equivalence)**.** *Physical inquiry and MetaCIC self-study are observation–reflection equivalent when there is a structure-preserving correspondence*

$$\Phi : \mathcal{R}_{\mathrm{phys}} \longrightarrow \mathcal{R}_{\mathrm{MetaCIC}}$$

whose component roles match as follows:

physical inquiry	*MetaCIC self-study*
physical system	*formal source or kernel behaviour*
finite observation trace	Hist
measurement continuation	Cont
observed signature	Sig
model classifier	*licensed classifier*
model package	Pkg *or derived interface*
experimental ledger	Gap *or NameCert ledger*
prediction horizon	*open meta continuation*
no final-fit oracle	*no internal truth predicate*

The equivalence is architectural, not ontological.

Definition 9414.413 (Reflection gap)**.** *A reflection gap is the gap between a system's finite certified self-description and a total truth predicate for that system. In physical inquiry it appears as the difference between*

$$\mathrm{Fit}_n(M, R; \Pi)$$

and

$$\forall H,\ \mathrm{Fit}(M, R; \Pi \mid H).$$

In MetaCIC it appears as the difference between structural self-description and an internal truth predicate for the system.

Definition 9414.414 (Internalization without finalization)**.** *A system has internalization without finalization when it can internalize finite descriptions, certificates, and structural reproductions of its own behaviour, but cannot internally certify that these descriptions are final, total, or truth-complete.*

Definition 9414.415 (Reflective stability certificate)**.** *A reflective stability certificate shows that a self-description can be reused under specified continuations without collapsing into a total truth oracle. In physical inquiry this is a law or induction certificate. In MetaCIC it is a NameCert, stability certificate, or derived-interface certificate whose reuse remains below the open meta-loop boundary.*

9414.39 Physical and formal reflective instances

Theorem 9414.416 (Physical inquiry instantiates reflective architecture)**.** *Physical inquiry has the form of a reflective inquiry architecture.*

Proof. Let the source be a physical system R. The observer does not access R as a completed totality; access is mediated by finite traces $H_n(R)$. These traces are organized by an observation bundle Π into signatures $\mathrm{ObsSig}_\Pi(R \mid H_n)$. A model M generates corresponding model signatures, and a classifier compares the two:

$$\mathrm{ModSig}_\Pi(M \mid H_n) \sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H_n).$$

Successful comparison produces a finite fit certificate. Reuse of the model requires stability data, and the source traces, approximations, hidden residues, and failure surfaces must be ledgered. Future observations remain open continuations. These are exactly the fields of Definition 9414.409. □

Corollary 9414.417 (Physical modelling is self-descriptive)**.** *A physical model describes not only a source object but also the observational process by which that object became modelable.*

Proof. By Definition 9414.416, the model is licensed by traces, signatures, classifiers, stability certificates, and ledgers. These components are part of the model's epistemic status, so the model records its own access route. □

Corollary 9414.418 (Physical theory is reflective without being subjective)**.** *Physical theory is reflective because it records conditions of observational access, but it is not subjective because reality-generated signatures may defeat model-generated signatures.*

Proof. Reflectivity follows from the certificate structure of Definition 9414.416. Non-subjectivity follows because finite fit fails whenever observed signatures separate from model signatures under the active classifier. □

Theorem 9414.419 (MetaCIC self-study instantiates reflective architecture)**.** *MetaCIC self-study has the form*

$$\mathsf{Hist} \to \mathsf{Cont} \to \mathsf{Sig} \to \textit{classifier} \to \mathsf{Pkg}/\mathsf{NameCert} \to \textit{ledger} \to \mathrm{OpenMeta}\,.$$

Proof. In BEDC, generated histories are extended by continuation. Ask and Sig interfaces expose visible data. Classifiers such as hsame, sameSig, package-sameness, or derived classifiers organize that visible data. Packages and derived interfaces are admitted only through certificate discipline. The NameCert template requires source, pattern, classifier, stability, and ledger fields. The meta-level remains open: the ground loop closes so the system can start, while the meta loop does not close into an internal truth predicate. Thus MetaCIC self-study instantiates Definition 9414.409. □

Corollary 9414.420 (MetaCIC self-study is structural, not totalizing)**.** *MetaCIC can structurally describe its own generative behaviour without producing a total internal truth predicate.*

Proof. By Definition 9414.419, self-study proceeds through histories, classifiers, packages, certificates, and ledgers while preserving the open meta-loop boundary. □

9414.40 The equivalence thesis

Theorem 9414.421 (Observation–reflection equivalence)**.** *Physical inquiry and MetaCIC self-study are observation–reflection equivalent.*

Proof. By Definition 9414.416, physical inquiry instantiates reflective inquiry architecture. By Definition 9414.419, MetaCIC self-study instantiates the same architecture. The correspondence sends reality-facing source to formal generative source, finite observation trace to Hist, measurement continuation to Cont, observed signature to Sig, model classifier to licensed classifier, model or law certificate to Pkg/NameCert, experimental ledger to gap or NameCert ledger, and open prediction horizon to open meta-continuation.

Each side proceeds from finite trace to visible signature, from signature to classifier, from classifier to package, from package to ledgered certificate, and from certificate to open continuation. Neither side possesses a final oracle: physical inquiry lacks a final-fit oracle over all possible observations, and MetaCIC lacks an internal truth predicate for itself. The structural roles and open-boundary conditions are preserved. □

Corollary 9414.422 (Physical inquiry is MetaCIC-like in form)**.** *Physical inquiry is not identical to MetaCIC, but it has the same epistemic skeleton.*

Proof. Definition 9414.421 establishes structural equivalence, not source identity. The two sides may differ in material source and empirical content while sharing finite histories, signatures, classifiers, certificates, ledgers, and open continuation. □

Corollary 9414.423 (MetaCIC self-study is physics-like in openness)**.** *MetaCIC self-study resembles physical inquiry in that its self-description remains open at the meta-boundary.*

Proof. Physical inquiry remains open because later observations may defeat present models. MetaCIC remains open because total self-truth cannot be internalized. By Definition 9414.421, both open boundaries occupy the same architectural role. □

9414.41 Architectural equivalence

Definition 9414.424 (Architectural equivalence)**.** *Two inquiry systems are architecturally equivalent when their roles, transitions, certificates, ledgers, and open boundaries correspond, even if their source domains differ. Architectural equivalence preserves inquiry form, not object substance.*

Definition 9414.425 (Ontological identity)**.** *Two systems are ontologically identical only when their source domains, objects, operations, and truth conditions coincide. The present framework does not assert ontological identity between physical reality and MetaCIC.*

Theorem 9414.426 (Observation–reflection equivalence is architectural)**.** *Observation–reflection equivalence does not imply that physical reality is a formal system or that MetaCIC is a physical system.*

Proof. The equivalence preserves the roles of finite trace, signature, classifier, package, ledger, and open continuation. It does not identify the source domains. In physical inquiry, the source is a reality-facing system capable of producing observations that may defeat the model. In MetaCIC self-study, the source is formal generative behaviour inside a certificate-governed framework. These sources are not the same object. □

Corollary 9414.427 (Structural equivalence does not erase empirical content)**.** *Even when physical inquiry and MetaCIC self-study share the same form, physical inquiry still requires empirical signatures.*

Proof. The physical side contains reality-generated traces. Those traces are not supplied by MetaCIC syntax; they are obtained through interaction with physical systems. □

Corollary 9414.428 (Formal reflection does not replace experiment)**.** *MetaCIC can clarify the architecture of inquiry, but it cannot supply the physical signatures needed to certify a model of reality.*

Proof. Physical adequacy requires agreement between model-generated and reality-generated signatures. Formal reflection can describe the required certificate structure, but it cannot generate empirical observations. □

9414.42 Internalization, reflection gaps, and failure surfaces

Theorem 9414.429 (Internalization does not imply finalization)**.** *If a system internalizes a description of its own behaviour, it does not follow that the system has finalized its own truth.*

Proof. Internalization constructs a representation, certificate, or model of some aspect of behaviour. Finalization would require a total truth predicate or final-fit oracle for all continuations. A finite or certificate-mediated self-description can succeed locally without covering all future extensions, refinements, or meta-level truth conditions. □

Corollary 9414.430 (Observation protocols do not make theories final)**.** *A physical theory can include its own measurement model, calibration rules, and uncertainty structure while remaining open to later failure.*

Proof. Such inclusion internalizes observational access. By Definition 9414.429, internalization does not imply finality. □

Corollary 9414.431 (Syntax internalization does not define truth)**.** *A formal system may represent its own syntax or proof structure without defining a total truth predicate for itself.*

Proof. This is the formal instance of Definition 9414.429. Structural self-description does not supply a truth predicate for the whole system. □

Definition 9414.432 (Reflection failure surface)**.** *A reflection failure surface is the class of continuations, refinements, or meta-queries that may expose the difference between certified local description and total truth. In physical inquiry it includes later observations, refined bundles, and hidden residues. In MetaCIC it includes attempted internal truth predicates, total halting predicates, or closure of the open meta loop.*

Theorem 9414.433 (Reflection gaps remain internally open)**.** *Neither the physical reflection gap nor the formal reflection gap can be closed from within the corresponding finite inquiry process.*

Proof. Closing the physical gap would require a final-fit oracle certifying fit over all admissible actual and counterfactual histories. Finite observation supplies only finite traces and stability certificates over specified continuation classes. Closing the formal gap would require an internal truth predicate for the system's own language or proof structure, which is exactly the meta-loop closure blocked by Tarski-style obstruction. Both gaps therefore remain internally open. □

Corollary 9414.434 (Failure surfaces are necessary)**.** *A theory without a failure surface is not more complete; it is less well-certified.*

Proof. A failure surface records where finite certification may fail. By Definition 9414.433, total closure is internally unavailable. Removing the failure surface hides the reflection gap rather than closing it. □

Corollary 9414.435 (Empirical revisability is open reflection)**.** *Empirical revisability in physics corresponds to open meta-reflection in MetaCIC.*

Proof. In physics, revisability means that later observation may defeat current fit. In MetaCIC, open reflection means that structural self-description does not become total truth. Both express the same continuation boundary. □

9414.43 NameCert as bridge concept

Definition 9414.436 (Physical NameCert analogue)**.** *A physical NameCert analogue for a model M of a system R is*

$$\mathrm{PhysNameCert}(M, R; \Pi) = (\mathrm{SourceSpec}, \mathrm{PatternSpec}, \mathrm{ClassifierSpec}, \mathrm{StabilityCert}, \mathrm{LedgerPolicy}).$$

It is the physical version of the BEDC certificate template.

Theorem 9414.437 (Physical models require NameCert-like structure)**.** *Every physically serious model must supply source, pattern, classifier, stability, and ledger fields.*

Proof. A model needs a source specification to state which observations licensed it, a pattern specification to state what regularity it claims, a classifier to state how model and reality are compared, a stability certificate to support prediction beyond finite fit, and a ledger policy because approximation hides information that may later become active. These are precisely the fields of Definition 9414.436. □

Corollary 9414.438 (A law is a certified name)**.** *A physical law is a certified name for a stable pattern in reality-facing signatures.*

Proof. By Definition 9414.437, a law must carry source, pattern, classifier, stability, and ledger data. The equation is only the visible pattern field. □

Corollary 9414.439 (The model name is certificate-bounded)**.** *A model's epistemic strength depends on the strength of its source, classifier, stability, and ledger fields.*

Proof. Without source the model is ungrounded. Without classifier it cannot be compared to observation. Without stability it cannot predict. Without ledger it cannot recover hidden residues. The name is licensed only to the extent that its certificate fields are supplied. □

9414.44 Reflective towers and open self-description

Definition 9414.440 (Reflective tower)**.** *A reflective tower is an approximation tower in which each level is a certified description of the preceding level:*

$$L_i \to \mathrm{Desc}(L_i) = L_{i+1}.$$

The tower is reflective because each stage turns prior behaviour into a new visible object.

Definition 9414.441 (Safe reflection)**.** *A reflective tower is safe when each level carries a ledger of hidden source material, each operation used at a higher level descends through the classifier of the lower level, and no level claims to be the total truth of the entire tower.*

Theorem 9414.442 (Certified approximation towers are reflective)**.** *If each compression arrow in an approximation tower extracts a pattern from the previous level and packages it as a visible object, then the tower is reflective.*

Proof. A reflective tower requires each level to be a certified description of the preceding level. A compression arrow extracts source material, applies a classifier, records a pattern, packages a visible endpoint, and stores hidden material in a ledger. This is exactly the formation of a description of the previous level. □

Corollary 9414.443 (Idealization is reflection under compression)**.** *An ideal object at the top of a tower is a reflected and compressed description of lower-level source behaviour.*

Proof. Ideal objects are produced by repeated extraction, stabilization, smoothing, completion, or packaging. By Definition 9414.442, each stage is reflective when it records a certified description of the previous stage. □

Corollary 9414.444 (Tower collapse is unsafe reflection)**.** *Tower collapse occurs when a higher-level self-description is treated as identical to the lower-level source without a descent certificate.*

Proof. Safe reflection requires ledgered compression and descent. Tower collapse removes these conditions and treats the endpoint as the source. □

Definition 9414.445 (Open self-description)**.** *An inquiry process is an open self-description when it constructs descriptions of its own access conditions while leaving continuation and failure surfaces open.*

Theorem 9414.446 (Scientific inquiry is open self-description)**.** *Physical scientific inquiry is an open self-description of reality-facing access.*

Proof. Scientific inquiry records observations, constructs models, specifies classifiers, defines measurement protocols, estimates uncertainty, records failure conditions, and revises models when observations change. It describes not only reality but also the conditions under which reality becomes available to inquiry. It nevertheless does not close all future observations into a final object. □

Corollary 9414.447 (Scientific realism is certificate realism)**.** *A realist account of scientific models should attach realism to certified signature preservation, not to bare model–reality identity.*

Proof. By Definition 9414.446, scientific inquiry is open self-description. By Definition 9414.437, physical models require certificate structure. Realism therefore attaches to certified reality-facing signature preservation under specified conditions. □

Corollary 9414.448 (The world constrains self-description)**.** *Open self-description is not free construction, because reality-generated signatures can defeat model-generated signatures.*

Proof. A model must match observed signatures under the active classifier. If reality generates a separating signature, finite fit fails. □

9414.45 Equivalence diagram

The equivalence thesis can be summarized as:

$$\begin{array}{lccccccc} \text{physics:} & R & \to & H_n & \to & \mathrm{ObsSig}_\Pi & \to & \mathrm{FitCert} \to \mathrm{LawCert} \to \mathrm{OpenFit} \\ \Phi & & & & & & & \\ \text{MetaCIC:} & \mathsf{Hist} & \to & \mathsf{Cont} & \to & \mathsf{Sig} & \to & \text{classifier} \to \mathsf{NameCert} \to \mathrm{OpenMeta}\,. \end{array}$$

The vertical correspondence preserves finite source access, signature extraction, classifier-mediated packaging, ledgered compression, certificate-governed reuse, and open continuation. It does not preserve source identity: R is not Hist, a laboratory is not a kernel, and a physical event is not a formal constructor.

Theorem 9414.449 (Equivalence diagram commutes at certification level)**.** *The observation–reflection diagram commutes when each physical modelling step is mapped to its corresponding MetaCIC certificate role.*

Proof. A physical observation trace maps to a Hist-level trace. Signature extraction maps to Ask/Sig generation. Model–data classification maps to licensed classifier comparison. Model packaging maps to package or derived-interface formation. Experimental ledgering maps to gap or NameCert ledgering. Prediction maps to open continuation under stability. Following the physical path and then applying Φ therefore yields the same certificate roles as first applying Φ to the source-access structure and then following the MetaCIC path. □

Corollary 9414.450 (Equivalence is certificate-preserving)**.** *Observation–reflection equivalence preserves source, pattern, classifier, stability, and ledger roles.*

Proof. Definition 9414.449 states that the diagram commutes at the level of certification. The relevant certificate fields are source, pattern, classifier, stability, and ledger. □

Remark 9414.451 (Compressed form)**.** *The thesis is:*

truth-like scientific content = finite signature fit+stability certificate+ledgered provenance+open continuation.

Physical inquiry is not a view from outside reality; it is a finite reflective process inside reality. MetaCIC is not a total self-truth machine; it is a finite reflective process inside a formal generative system. Both construct stable, certificate-governed signatures while refusing final meta-closure.

Concretized in §1126: ObservationReflectionPacketUp realizes the observation–reflection equivalence as a finite source, history, signature, classifier, package, ledger, and open-continuation NameCert packet.

Concretized in §1149: ReflectiveStabilityUp realizes reflective stability as a finite source, checkpoint, classifier, open-continuation, transport, ledger, and NameCert packet.

Concretized in §1161: ReflectionGapBoundaryUp realizes the gap between finite certified self-description and total truth as a finite inquiry, ledger, continuation, and boundary NameCert packet.

Concretized in §7705: ReflectionGapAuditUp realizes reflection-gap audits as finite inquiry, ledger, continuation, non-finalization, transport, replay, provenance, and NameCert rows.

Concretized in §1151: The reflective inquiry architecture is realized as a finite NameCert packet relating physical inquiry and MetaCIC self-study through source, signature, classifier, package, ledger, and open-continuation rows.

Concretized in §1192: Finite observation-reflection seals realize the physical inquiry and MetaCIC self-study correspondence as a finite NameCert packet with shared classifier, ledger, and non-finalization rows.

Concretized in §5109: Finite observation-reflection packets realize source, observation trace, signature, classifier, package, ledger, and open-continuation rows as one NameCert surface.

Concretized in §1191: PhysicalNameCertAuditUp realizes the physical NameCert analogue as finite source, observed-signature, model-signature, classifier, stability, ledger, failure-surface, route, provenance, and local NameCert rows.

9414.46 Consequences for explanation, realism, and idealization

Concretized in §1185: Reality-constrained scientific methodology is realized as a finite NameCert packet for explanation certificates, effective-domain regions, idealization residue, failure surfaces, audit routes, and local naming rows.

Observation–reflection equivalence changes the status of scientific explanation. An explanation is not merely an equation, derivation, mechanism, or model. It is a certified relation between finite observational access, extracted signatures, classifiers, stability conditions, and ledgered idealizations.

The physical reading follows the BEDC discipline for derived interfaces. Equality, function, number-like carriers, limits, real-like objects, and circle-like objects are not primitive simply because their names are familiar. They are derived interfaces licensed by certificates and ledgers. Likewise, a physical model is explanatory only when it shows how reality-facing signatures are preserved, compressed, stabilized, and made available for prediction.

Definition 9414.452 (Scientific explanation)**.** *A scientific explanation of a phenomenon P by a model M, relative to an observation bundle Π, is a certificate-mediated account showing that*

$$\mathrm{PhysSig}_{\Pi}(M) \sim_{\Pi} \mathrm{PhysSig}_{\Pi}(P),$$

and that the predictive or explanatory operations used in the account descend through the approximation tower that produced M.

Definition 9414.453 (Explanation certificate)**.** *An explanation certificate is*

$$\mathrm{ExplCert}(M, P; \Pi) = (\mathrm{SourceSpec}, \mathrm{PhenomenonSpec}, \mathrm{PatternSpec}, \mathrm{ClassifierSpec}, \mathrm{StabilityCert}, \mathrm{DescentCert}, \mathrm{Ledg}$$

It records the observations from which the phenomenon is known, the phenomenon to be explained, the regularity or mechanism claimed, the comparison rule between model and reality, the stability basis for reuse, the descent proof for operations through the tower, the ledger of idealizations and residues, and the observations or refinements that would defeat the explanation.

Definition 9414.454 (Explanatory scope and residue)**.** *The explanatory scope of* $\mathrm{ExplCert}(M, P; \Pi)$ *is the class of phenomena, histories, tower levels, and continuations over which the certificate remains valid. The explanatory residue* $\mathrm{ExplRes}(P, M; \Pi)$ *is the part of the phenomenon or system not captured by the certificate, including unmodelled variables, hidden tower levels, idealized-away structures, measurement conditions, and physical properties not queried by* Π.

Theorem 9414.455 (Explanation requires certificate)**.** *A model explains a phenomenon only relative to an explanation certificate.*

Proof. To explain a phenomenon, a model must be connected to observations of it, must state the pattern being explained, must compare model-generated and reality-generated signatures, must license reuse beyond the finite source trace, must show that explanatory operations descend through any compressions used, must record hidden source structure, and must specify the failure surface. These are exactly the fields of Definition 9414.453. □

Corollary 9414.456 (Equations and derivations are not explanations by themselves)**.** *An equation or derivation becomes explanatory only when embedded in a certificate specifying source, classifier, stability, descent, ledger, and failure surface.*

Proof. An equation expresses a mathematical relation, and a derivation shows that a conclusion follows from assumptions internal to a model. Neither supplies the model-to-phenomenon certificate required by Definition 9414.455. □

Theorem 9414.457 (Explanation cannot exceed scope)**.** *A claim derived from M is scientifically explained by* $\mathrm{ExplCert}(M, P; \Pi)$ *only when the claim lies within the certificate's explanatory scope.*

Proof. The certificate licenses explanation only over its stated source traces, classifiers, tower levels, operations, stability domain, and continuations. Outside that scope at least one licensing field is missing or inactive. □

Corollary 9414.458 (Overextension is certificate failure)**.** *A model is overextended when it is used beyond the scope of its explanation certificate.*

Proof. By Definition 9414.457, uses outside the certificate scope lack the needed source, classifier, stability, descent, or ledger field. □

9414.47 Certificate realism

Definition 9414.459 (Certificate realism)**.** *Certificate realism is the position that realist commitment attaches to certified stable signature preservation, not to bare model syntax. A model M is realist-worthy relative to R and Π when*

$$\mathrm{PhysSig}_{\Pi}(M) \sim_{\Pi} \mathrm{PhysSig}_{\Pi}(R)$$

is supported by source, classifier, stability, descent, and ledger data.

Definition 9414.460 (Signature realism and syntax realism)**.** *Signature realism is the special case of certificate realism in which the primary commitment is to stable invariant signatures generated by a system rather than to any single mathematical representation of those signatures. Syntax realism is the stronger claim that the mathematical syntax of a successful model directly describes the ontic structure of reality. The present framework treats syntax realism as under-certified unless the syntax is shown to preserve the relevant signatures and operations through the modelling tower.*

Theorem 9414.461 (Certificate realism lies between syntax realism and instrumentalism)**.** *Certificate realism is weaker than syntax realism and stronger than instrumentalism.*

Proof. It is weaker than syntax realism because it does not assert $M = R$; it asserts certified preservation of reality-facing signatures under specified bundles and tower levels. It is stronger than instrumentalism because it does not treat the model as merely useful. It claims that the model preserves stable signatures generated by the system, and those signatures can refute it. □

Corollary 9414.462 (Realism can be domain-local)**.** *A model may carry realist commitment within its certified signature domain while remaining open to later refinement.*

Proof. Certificate realism is indexed by bundle, tower level, and continuation class. Later refinement may expose active residue without invalidating the certified commitment in the earlier domain. □

Corollary 9414.463 (Theory change is compatible with realism)**.** *If realist commitment attaches to certified signatures rather than full syntax, theory change need not destroy realism.*

Proof. A later theory may replace earlier syntax while preserving the signatures certified by the earlier theory. The commitment transfers to the preserved signature content. □

9414.48 Effective theories

Definition 9414.464 (Effective theory)**.** *An effective theory is a model M_{eff} whose explanation, prediction, and objectivity claims are valid only within a specified tower region*

$$\mathcal{R}_{\mathrm{eff}} = (\Pi, \mathcal{C}, L_i, \ldots, L_j),$$

certified by source specification, coarse classifier, stability domain, descent domain, ledger policy, and failure surface.

Definition 9414.465 (Effective exactness and domain leakage)**.** *An effective theory is effectively exact over $\mathcal{R}_{\mathrm{eff}}$ when every admissible observation or operation inside that region preserves the relevant physical signatures. Domain leakage occurs when a claim is made at a level, bundle, or continuation class not included in $\mathcal{R}_{\mathrm{eff}}$ without an additional upgrade certificate.*

Theorem 9414.466 (Effective theories can be locally exact)**.** *An effective theory may be exact within its certified domain even if it is not globally final.*

Proof. Effective exactness requires signature preservation for all admissible observations and operations inside the certified region. It does not require preservation under every possible refinement, scale, or future bundle. □

Corollary 9414.467 (Effective theories are not merely false)**.** *An effective theory is not false merely because it fails outside its certified domain.*

Proof. Failure outside the certified region shows that the certificate does not extend there. It does not defeat exactness inside the region licensed by Definition 9414.466. □

Theorem 9414.468 (Domain leakage requires upgrade certificate)**.** *Using an effective theory outside its certified region is invalid unless an upgrade certificate extends the relevant source, classifier, stability, descent, and ledger fields.*

Proof. A certificate licenses claims only inside its stated scope. Outside that scope the previous fields no longer apply. An upgrade certificate is precisely the extension of those fields to the new region. □

Corollary 9414.469 (Reduction is not automatic)**.** *A higher-level effective theory does not automatically reduce to a lower-level theory, and a lower-level theory does not automatically explain every higher-level structure.*

Proof. Reduction across levels requires cross-level descent, signature preservation, and ledger compatibility. These are additional certificates, not consequences of level membership alone. □

9414.49 Idealization as licensed compression

Definition 9414.470 (Idealization operator)**.** *An idealization operator*

$$I : L_i \rightsquigarrow L_j$$

is a compression arrow replacing a lower-level source with a higher-level mathematical object, such as finite data by a continuous curve, a rough boundary by a smooth boundary, a molecular system by a continuum field, or a finite process by a limit object.

Definition 9414.471 (Idealization certificate)**.** *An idealization certificate for I is*

$$\mathrm{IdealCert}(I) = (\mathrm{SourceSpec}, \mathrm{IdealPattern}, \mathrm{ClassifierSpec}, \mathrm{StabilityCert}, \mathrm{DescentCert}, \mathrm{ResidueLedger}, \mathrm{FailureSurfac}$$

The residue ledger records what the idealization removes from the visible endpoint.

Definition 9414.472 (Safe idealization)**.** *An idealization is safe relative to Π when each removed residue is physically inactive under Π, or provably irrelevant by descent, or explicitly recorded as a failure surface.*

Theorem 9414.473 (Idealization requires certification)**.** *An idealized model is physically legitimate relative to Π only if it carries an idealization certificate.*

Proof. Idealization removes or compresses source structure. If the removed structure is inactive, that fact must be certified. If operations are applied to the ideal endpoint, they must descend through the compression. If removed structure may become active, it must be recorded as a failure surface. These are the fields of Definition 9414.471. □

Corollary 9414.474 (Idealization hides rather than deletes)**.** *When an idealization replaces a source by an ideal object, the hidden source structure remains part of the ledger.*

Proof. Definition 9414.471 includes a residue ledger. The removed source structure is hidden at the endpoint but retained as provenance. □

Corollary 9414.475 (Smoothness is not free)**.** *If smoothness is introduced by idealization, smoothness claims about the source require descent.*

Proof. Smoothness may be introduced by interpolation, limiting, averaging, or completion. By Definition 9414.473, claims made using the smooth endpoint must descend through the idealization. □

Theorem 9414.476 (Idealization failure localizes residue)**.** *If an idealized explanation fails under a refined observation bundle, the failure identifies active residue hidden by the idealization.*

Proof. A refined bundle that defeats the explanation exposes a distinction not preserved by the idealized endpoint. Since the endpoint was produced by hiding source structure, the exposed distinction belongs to the residue ledger. □

9414.50 Model pluralism and theory change

Definition 9414.477 (Certified model pluralism)**.** *A family of models* $\{M_\alpha\}_{\alpha\in A}$ *exhibits certified model pluralism for* R *relative to* Π *when each* M_α *carries a physical-fit certificate for* R *and all members preserve the same physical signature under* Π*:*

$$\mathrm{PhysSig}_\Pi(M_\alpha) \sim_\Pi \mathrm{PhysSig}_\Pi(M_\beta).$$

Definition 9414.478 (Translation certificate and representation equivalence)**.** *A translation certificate* $\mathrm{TransCert}(M_1, M_2; \Pi)$ *shows that the signatures, operations, ledgers, and failure surfaces of* M_1 *and* M_2 *correspond under* Π*. Two representations are representation-equivalent relative to* Π *when such a certificate exists.*

Theorem 9414.479 (Model pluralism requires translation certificates)**.** *Multiple mathematical models can represent the same physical system relative to* Π *when translation certificates show that they preserve the same physical signatures and compatible ledgers.*

Proof. If each model preserves the physical signature of R under Π, each is physically adequate there. If translation certificates also match their operations, ledgers, and failure surfaces, the models differ in representation rather than certified physical content. □

Corollary 9414.480 (Prediction agreement is not representation equivalence)**.** *Two models may agree on current predictions but fail to be representation-equivalent if their ledgers or failure surfaces differ.*

Proof. Prediction agreement concerns present signatures. Representation equivalence requires translation of operations, ledgers, and failure surfaces. □

Corollary 9414.481 (Pragmatic model choice can remain realist)**.** *If two models are representation-equivalent under* Π*, choosing one for convenience does not undermine realism.*

Proof. Realist commitment attaches to certified signatures, not syntactic form. If both models preserve the same certified signatures, convenience selects a representation, not the physical content. □

Definition 9414.482 (Theory transition)**.** *A theory transition* $T_1 \rightsquigarrow T_2$ *is a certified passage from one model or theory to another together with a map between their source specifications, classifiers, stability domains, ledgers, and failure surfaces.*

Definition 9414.483 (Conservative signature preservation and ledger refinement)**.** *A theory transition is conservative over* Π *when every signature certified by* T_1 *under* Π *is preserved by* T_2*. It is a ledger refinement when* T_2 *preserves old successful signatures while adding ledger information explaining where* T_1 *hid, approximated, or failed to descend.*

Theorem 9414.484 (Theory change can preserve realist content)**.** *If* $T_1 \rightsquigarrow T_2$ *is conservative over* Π*, then the realist content certified by* T_1 *under* Π *is preserved in* T_2*.*

Proof. Under certificate realism, realist content is stable signature preservation. Conservative transition preserves those signatures, so the certified content remains present in the later theory. □

Corollary 9414.485 (Progress is ledger expansion)**.** *Scientific progress can be understood as expansion and refinement of the ledger of preserved signatures, active residues, failure surfaces, and descent certificates.*

Proof. A later theory improves an earlier one by preserving successful signatures, identifying failures, exposing residues, extending stability domains, and adding descent certificates. These are ledger refinements. □

9414.51 Failure surfaces and ideal objects

Definition 9414.486 (Negative scientific knowledge)**.** *Negative scientific knowledge is certified knowledge that a claim is not licensed under the current source, classifier, stability, descent, or ledger fields. It has the form*

$$\mathrm{CannotClaim}(\varphi; \mathcal{C}),$$

where $\mathcal{C}$ *records the missing or failed certificate field.*

Definition 9414.487 (Physical failure certificate)**.** *A physical failure certificate for* φ *is a typed witness that at least one required field of the relevant explanation, idealization, induction, or representation certificate fails, such as missing source, missing classifier, non-descent, absent ledger, stability failure, or unmatched signature.*

Theorem 9414.488 (Negative knowledge is scientific content)**.** *A certified refusal to assert* φ *is part of the scientific content of a theory.*

Proof. A theory is defined by its certification boundaries as well as by its positive claims. If φ lacks source, classifier, stability, descent, or ledger support, asserting it would exceed the theory's scope. A certified refusal records that boundary. □

Corollary 9414.489 (Failure surfaces are explanatory)**.** *A failure surface explains where and how a model may cease to apply.*

Proof. It records the observations, refinements, or operations that would defeat the model's certificate, thereby identifying the limits of the explanation and the residues likely to become active. □

Definition 9414.490 (Certified ideal use)**.** *An ideal mathematical object has certified ideal use for a physical system when it carries a tower ledger, signature-preservation certificate, descent certificate, and failure surface.*

Definition 9414.491 (Ideal endpoint realism)**.** *Ideal endpoint realism is the claim that the ideal object at the top of a modelling tower is itself the physical ontology. The present framework rejects this unless the endpoint is shown to preserve all relevant signatures and all operations used on it descend through the tower.*

Theorem 9414.492 (Ideal objects are admissible as certified endpoints)**.** *An ideal mathematical object is physically admissible only when it is connected to physical source traces by a certified tower.*

Proof. An ideal object has mathematical structure, but physical admissibility requires connection to reality-facing signatures. A certified tower supplies that connection by preserving signatures, recording hidden residue, and licensing operations by descent. □

Corollary 9414.493 (Usefulness does not imply ideal ontology)**.** *An ideal mathematical object may be useful, predictive, and explanatory without being the ontic state itself.*

Proof. Usefulness follows from certified ideal use. Ontic identity would require the stronger claim that the endpoint exhausts the ontic state, which Definition 9414.492 does not supply. □

9414.52 Scientific theory as certificate tower

Definition 9414.494 (Scientific theory)**.** *A scientific theory is a structured family*

$$\mathcal{T}_{\text{sci}} = (\mathcal{M}, \mathcal{O}, \Pi, \text{FitCert}, \text{LawCert}, \text{ExplCert}, \text{IdealCert}, \text{Ledger}, \text{FailureSurface}),$$

where $\mathcal{M}$ is a family of models, $\mathcal{O}$ is a family of observed phenomena, Π is a family of observation bundles, and the remaining fields certify fit, laws, explanations, idealizations, provenance, and failure conditions.

Definition 9414.495 (Theory strength)**.** *The strength of a scientific theory is determined by source breadth, classifier precision, predictive stability, descent depth, ledger completeness, failure-surface explicitness, and extent of inter-observer invariants. Theory strength is certificate strength.*

Theorem 9414.496 (Scientific theories are certificate-governed approximation towers)**.** *Every physically serious scientific theory can be represented as a certificate-governed approximation tower.*

Proof. A scientific theory begins from observations or experimental sources, extracts signatures, classifies them, packages them into models, stabilizes them into laws, uses them in explanations, and extends them into predictions. It also hides, idealizes, and compresses information, requiring ledgers and failure surfaces. This is exactly the structure of a certificate-governed approximation tower. □

Corollary 9414.497 (Theory comparison is certificate comparison)**.** *Comparing two theories requires comparing their certificates, not merely their equations.*

Proof. Theories may use different syntax while preserving the same signatures, or the same syntax while carrying different ledgers and failure surfaces. Source, classifier, stability, descent, ledger, and failure fields must therefore be compared. □

Corollary 9414.498 (A stronger theory may be less elegant)**.** *A theory with more complete certificates may be scientifically stronger even if its mathematical form is less elegant.*

Proof. Scientific strength is certificate strength. Elegance does not determine source coverage, classifier adequacy, stability, descent, ledger completeness, or failure-surface explicitness. □

Remark 9414.499 (Compressed form)**.** *The resulting image is:*

scientific theory = models + signatures + classifiers + stability + descent + ledgers + failure surfaces.

Explanation is certified signature preservation; realism attaches to stable signatures; effective theories are tower-local; idealization is licensed compression; model pluralism requires translation certificates; and theory change is often ledger refinement rather than erasure.

Concretized in §1136: Reality-constrained explanations are realized as finite observation, model, phenomenon, signature-comparison, descent, residue, failure-surface, transport, route, provenance, and NameCert rows.

Concretized in §1145: The scientific-methodology vision is realized as a finite methodology-ledger NameCert packet for explanation scope, effective-region bounds, idealization residue, domain-leakage refusal, upgrade routes, provenance, and local naming rows.

Concretized in §1170: ScientificIdealizationResidueSealUp realizes safe idealization as a finite residue, descent, failure-surface, scope, transport, route, provenance, and NameCert packet.

Concretized in §1146: Theory transitions are realized as finite packets preserving old certified signatures while exposing classifier, stability, ledger, and failure-surface refinement rows.

9414.53 Scientific Inquiry as Ledgered Reflection of Hist-Time

Remark 9414.500 (Vision seed status)**.** *This interface has vision-seed status: it is a schema horizon and has no Lean target. It is not a finite-kernel theorem. It records the certificate shape a later development would have to close before scientific inquiry, lawlike fit, or model-mediated prediction could be reported as a BEDC theorem about reality.*

The schema uses only the current BEDC reading that observer identity is internal to Hist*, that* Cont *has only history-valued arguments, and that observation, distinction, and local time-step share the same continuation form. It does not assert a primitive equation*

$$\mathsf{Science} = \mathsf{Time}.$$

Kernel time is Hist *growth by* E1/E0 *distinctions. Scientific inquiry is a derived, ledgered, classifier-governed reflection on such local Hist-time. It treats the traces left by observation-time as sources, extracts signatures from them, compares observed and model-generated signatures by licensed classifiers, and records the stability, descent, provenance, and failure surfaces required before a pattern may be reused as lawlike.*

Definition 9414.501 (Scientific inquiry context)**.** *A* scientific inquiry context *is a ledger-readable package*

$$S = (h, \Pi, X, \mathsf{Obs}, M, \mathsf{Mod}, C, D, L, \Phi),$$

where $h : \mathsf{Hist}$ *is the observer-history,* Π *is a repeated probe pattern or probe bundle,* X *is a candidate phenomenon or observed interface,* Obs *is the finite family of observed signatures generated by applying* Π *inside* h*,* M *is a candidate model family,* Mod *is the finite family of model-generated signatures exposed for comparison with* Obs*,* C *is the classifier of fit between observed and model-generated signatures,* D *records the announced stability and descent obligations,* L *records source, pattern, classifier, stability, and hidden-provenance data, and* Φ *is the failure surface specifying which future observations would defeat or downgrade the present claim.*

Definition 9414.502 (Scientific reflection prefix)**.** *For a scientific inquiry context* S*, a* scientific reflection prefix *of length* N *is a finite sequence*

$$Q_N^S = (q_0, q_1, \ldots, q_N)$$

whose entries are BEDC-visible query, response, signature, classifier, and ledger rows generated by repeatedly applying Π *to* X *as recorded in* h*. An infinite scientific stream is admissible here only as a schema whose every finite prefix is ledger-visible; no completed external stream and no global science-time are added to the kernel.*

Definition 9414.503 (Time-reflective fit)**.** *A scientific reflection prefix* Q_N^S *has* time-reflective fit *when the ledger supplies:*

1. *source generation witnesses showing that the observed signatures arise from finite* Cont *traces in* h*;*
2. *model-generation witnesses exposing the corresponding signatures of* M*;*
3. *classifier witnesses* C *relating the observed and model-generated signatures;*
4. hsame*-stability and descent witnesses showing that the comparison respects the announced source and target classifiers;*
5. *ledger rows recording any compression, idealization, hidden provenance, or approximation used by the comparison.*

Definition 9414.504 (Lawlike and speculative appearance)**.** *A scientific reflection prefix* Q_N^S *has* lawlike appearance *relative to* S *when it has time-reflective fit and the ledger also supplies stability over the announced continuation class* D *together with an explicit failure surface* Φ*. It has* speculative appearance *relative to* S *when a finite fit is present but at least one of stability, descent, ledger completeness, or failure-surface exposure remains open.*

The phrase lawlike appearance is part of the schema. It does not assert that the model is identical with reality, that a function is identical with a law, or that scientific inquiry is identical with time itself. It classifies observer-relative, certificate-governed reuse of finite observation-time traces.

Proposition 9414.505 (Science is not time itself)**.** *Within a fixed scientific inquiry context, scientific inquiry is not kernel time itself. It is ledgered reflection on local Hist-time.*

Proof. Kernel time is generated by a single $\mathsf{E1}/\mathsf{E0}$ distinction step and by finite Cont growth of Hist. A scientific inquiry context requires additional fields: a probe bundle, observed signatures,

model-generated signatures, a fit classifier, stability and descent obligations, a provenance ledger, and a failure surface. These fields are not produced by every Cont step. Hence Hist growth implies local time, but does not by itself imply scientific inquiry.

Conversely, without local Hist-time there are no generated observation traces from which source signatures can be extracted. Scientific inquiry therefore depends on Hist-time as its source material, but it is not identical with Hist-time. It is the certificate-governed reuse and reflection of traces generated in time. □

Corollary 9414.506 (No context-free science-time predicate)**.** *The schema licenses only context-indexed judgments such as* $\mathsf{ScientificFit}_N^S(Q)$*,* $\mathsf{Lawlike}_N^S(Q)$*, and* $\mathsf{Speculative}_N^S(Q)$*. It does not license an unindexed kernel predicate* $\mathsf{Science} = \mathsf{Time}$ *or* $\mathsf{Science}(\mathsf{Time})$ *as a completed global object.*

Proof. Removing the context S removes the observer-history, probe pattern, target interface, observed signatures, model signatures, classifier, stability and descent obligations, ledger, and failure surface. BEDC supplies no global observer, no global clock, no context-free classifier, and no external equality oracle capable of replacing these fields. □

Remark 9414.507 (Reading)**.** *Scientific inquiry is local time becoming ledger-readable to itself. The observation-distinction-time identity supplies the raw form of trace generation; scientific method adds signature extraction, classifier comparison, model reuse, stability, descent, ledger exposure, and failure conditions. The safe compressed reading is therefore not*

$$\mathsf{Science} = \mathsf{Time},$$

but

$$\mathsf{Science} = \mathsf{LedgeredReflection}(\mathsf{HistTime}).$$

Under this reading, a scientific law is not a bare function and a machine-learning model is not a law. A lawlike claim is a stable, classifier-respecting, failure-exposed certificate over finite observation-time traces. A model, including a computational or learning model, is a candidate generator of signatures whose scientific status depends on the audit fields above.

Concretized in §1273: Scientific model bridges realize this vision as finite scientific-object, model-audit, standard-bridge-audit, open-fit, failure-surface, transport, provenance, and NameCert rows.

Concretized in §1124: Reality-constrained model audits realize the source-trace, observed-signature, model-signature, classifier, tower, stability, descent, ledger, failure-surface, and audit-strength rows required by the schema.

Concretized in §479: Observer-Hist identity supplies observer as Hist without adding observer-substance.

Concretized in §603: Hist time streams supply finite local Cont-step rows without adding a global clock.

9414.54 Scientific Inquiry, Consciousness-Like Reflection, and T-Supply Boundaries

Remark 9414.508 (Vision seed status)**.** *This interface has vision-seed status: it is a schema horizon and has no Lean target. It is not a finite-kernel theorem. It records the comparison shape*

a later development would have to close before scientific inquiry, consciousness-like reflection, or T*-supply exposure could be reported as BEDC theorems about reality.*

The interface uses the current BEDC readings that observer identity is internal to Hist*, that* Cont *has only history-valued arguments, that observation, distinction, and local time-step share the same continuation form, that consciousness enters the object layer only as a records-side carrier, and that* T *is a non-internalizable forward-supply position. It therefore refuses the primitive equations*

$$\mathsf{Science} = \mathsf{Consciousness}, \qquad \mathsf{Science} = \mathsf{T}, \qquad \mathsf{Consciousness} = \mathsf{T}.$$

The safe reading is instead

$$\mathsf{Science} = \mathsf{PublicLedgeredMode}\big(\mathsf{ConsciousnessLikeReflection}(\mathsf{HistTime})\big).$$

Scientific inquiry may be consciousness-like because it can read, classify, and reuse its own observation traces. It is not consciousness itself, because a consciousness reading is the focused records-side carrier of Definition 9462.850, while scientific inquiry adds public signatures, model-generated signatures, classifiers, stability, descent, provenance, and failure surfaces. It is not T*, because* T *is not an object, state, record, rule, axiom, model, or scientific theory. Scientific inquiry exposes* T*-supply sockets whenever finite records are asked to bind future lawlike truth beyond their displayed witnesses; it does not internalize the supply position.*

Definition 9414.509 (Consciousness-like reflection)**.** *Let* L_N *be a finite focused multi-*Hist *ledger and let* F *be its focus. A records-side process* P *is* consciousness-like *relative to* (L_N, F) *when* P *factors through the focused carrier*

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho),$$

and later rows of P *may use the focused generated history* $H(F)$*, its local continuation trace* $\Theta(F)$*, its selector ledger* $S(F)$*, and its endpoint or trace self-proxy* ρ *as source material for readback, comparison, or further inscription. This is a structural records-side condition. It is not an object-level definition of phenomenal essence and it adds no subject-substance.*

Definition 9414.510 (Public scientific reflection)**.** *Let* $A = (M, R, \Pi)$ *be a physical audit target, with* M *a mathematical, computational, or conceptual model,* R *a target phenomenon or system, and* Π *an observation bundle. A* public scientific reflection *over* A *and* (L_N, F) *is a finite ledger-readable packet*

$$\mathsf{SciRef}_N = (\mathsf{CR}, \mathsf{Source}, \mathsf{ObsSig}, \mathsf{ModSig}, C, \mathsf{Tower}, D, L, \Phi, \mathsf{Strength}),$$

where $\mathsf{CR} = \mathsf{ConsciousRecord}(L_N, F)$*;* Source *records finite source traces drawn from or related to* CR*;* ObsSig *records observed signatures generated under* Π*;* ModSig *records model-generated signatures exposed by* M*;* C *is the classifier of fit between* ObsSig *and* ModSig*;* Tower *records the approximation path from source traces to public model use;* D *records stability and descent obligations;* L *records provenance, compression, hidden residue, idealization, and translation data;* Φ *is the failure surface; and* $\mathsf{Strength}$ *records the strongest justified audit strength.*

The packet is public when its visible claims are classifier-stable, reproducible or transportable through stated inter-Hist *relation rows, and failure-exposed. A private readback of a focused trace is therefore not yet science. It becomes scientific only when it is externalized as source, signature, classifier, stability, descent, ledger, and failure-surface data.*

Definition 9414.511 (T-supply exposure by scientific inquiry)**.** *A public scientific reflection exposes a* T*-supply socket when a finite prefix of* SciRef_N *is asked to license a forward-binding claim not generated by the displayed records, closed inductive generators, naming-certificate fields, or certified continuation laws. Typical scientific cases include universal lawlike reuse, prediction beyond the observed source horizon, open-ended induction, cross-regime extrapolation, and claims of final explanation.*

Such exposure is admissible only as a socketed ledger row: the missing shape, consumption site, intended claim strength, and preventing audit gate must be stated. The socket records the boundary. It does not turn T *into a model component or a hidden scientific axiom.*

Proposition 9414.512 (Focused source readback is consciousness-like)**.** *In a fixed context where the source traces of* SciRef_N *factor through* $\mathsf{ConsciousRecord}(L_N, F)$*, scientific inquiry is consciousness-like reflection.*

Proof. By definition, the source material of SciRef_N is drawn from or related to

$$\mathsf{CR} = (H(F), \Theta(F), S(F), \rho).$$

The observed signatures, model comparison rows, classifier witnesses, and later ledger rows reuse that finite trace material as public source data. Thus the scientific packet performs readback on focused Hist-time traces and records the result as further rows. This is exactly the structural condition of consciousness-like reflection relative to (L_N, F). No subject parameter is added, since all rows factor through the focused history, continuation trace, selector ledger, self-proxy, and public certificate fields. □

Proposition 9414.513 (Scientific inquiry is not consciousness itself)**.** *Scientific inquiry is not identical with* $\mathsf{ConsciousRecord}(L_N, F)$*.*

Proof. $\mathsf{ConsciousRecord}(L_N, F)$ has the four records-side components $H(F)$, $\Theta(F)$, $S(F)$, and ρ. A public scientific reflection requires additional fields: source-trace exposure, observed signatures, model-generated signatures, a fit classifier, an approximation tower, stability and descent obligations, a provenance ledger, a failure surface, and an audit-strength row. These fields are not supplied by every consciousness reading. Conversely, a focused conscious record may exist without any model family, public classifier, reproducibility row, or failure surface. Therefore scientific inquiry can be a public mode of consciousness-like reflection, but it is not consciousness itself. □

Corollary 9414.514 (No de-indexed consciousness-science predicate)**.** *The schema licenses only context-indexed judgments such as*

$$\mathsf{SciRef}_N^A(L_N, F), \qquad \mathsf{ConsciousLike}_N(L_N, F), \qquad \mathsf{PublicFit}_N^A(L_N, F), \qquad \mathsf{Lawlike}_N^A(L_N, F).$$

It does not license a context-free predicate $\mathsf{Science} = \mathsf{Consciousness}$*.*

Proof. Removing the indices removes the focused ledger, focus, observer-history, trace, selector ledger, self-proxy, model, target phenomenon, observation bundle, classifier, stability and descent obligations, ledger, failure surface, and audit strength. BEDC supplies no global observer, no global clock, no hidden subject carrier, and no context-free classifier capable of replacing these fields. □

Proposition 9414.515 (Scientific inquiry is not T)**.** *Scientific inquiry is not identical with the transcendental supply position* T*.*

Proof. A scientific reflection is a finite ledger-readable packet: its public standing depends on source rows, signature rows, classifier rows, stability and descent rows, provenance rows, failure-surface rows, and audit-strength rows. Thus it is an object-layer or certificate-layer construction. By Definition 9393.11, T is the structural position from which forward-binding supply enters a closed observational system; it is not an observation state, not a record, not a closed-generation rule, not a substrate axiom, and not an object-layer carrier. Identifying scientific inquiry with T would internalize the supply position as a scientific object or theory. That is exactly what the T-boundary and audit gates refuse. □

Corollary 9414.516 (T-sockets do not make the scientist T)**.** *A scientific inquiry may expose* T*-supply sockets without being* T*, and a conscious record may be an inscription point without being* T*.*

Proof. Socket exposure marks a missing forward-binding discharge and the gate that prevents its internalization. Inscription records the local site at which durable records enter a closed observational history. Neither operation supplies the far-end supply position as an internal object. They record where the boundary is encountered. □

Proposition 9414.517 (Science is public consciousness-like reflection)**.** *Within this vision-seed schema, scientific inquiry is the public, classifier-governed, failure-exposed mode of consciousness-like Hist-time reflection.*

Proof. Consciousness-like reflection supplies the focused readback surface: generated history, local continuation trace, selector ledger, and self-proxy. Scientific reflection adds the public audit discipline: source traces, observed signatures, model-generated signatures, classifier of fit, approximation tower, stability domain, descent obligations, provenance ledger, failure surface, and audit strength. The resulting packet is therefore neither a bare private conscious record nor a transcendental supply position. It is focused Hist-time reflection disciplined into public certificate form. □

Corollary 9414.518 (Safe and unsafe compressed readings)**.** *The safe compressed readings are*

$$\mathsf{Consciousness} = \mathsf{FocusedSelfProxy}(\mathsf{HistTime}), \qquad \mathsf{Science} = \mathsf{PublicLedgeredReflection}(\mathsf{HistTime}), \qquad \mathsf{T} = \mathsf{NonInter}$$

or, more sharply,

$$\mathsf{Science} = \mathsf{PublicLedgeredMode}\big(\mathsf{ConsciousnessLikeReflection}(\mathsf{HistTime})\big).$$

The unsafe promotions are

$$\mathsf{Science} = \mathsf{Consciousness}, \qquad \mathsf{Science} = \mathsf{T}, \qquad \mathsf{Consciousness} = \mathsf{T}.$$

Proof. The safe readings preserve the required indices and certificate layers. The unsafe promotions erase the difference between records-side focused carrier, public scientific audit packet, and non-internalizable supply position. □

Remark 9414.519 (Reading)**.** *Scientific inquiry is not consciousness as such, but consciousness-like reflection disciplined into a public ledger. A focused conscious record can read itself; scientific method makes such readback reproducible, classifier-visible, model-comparable, stability-indexed, and failure-exposed. A scientific law is therefore not a private experience, not a bare function, and*

not a machine-learning model by itself. A lawlike claim is a stable, classifier-respecting, failure-exposed certificate over finite observation-time traces.

The same structure explains why scientific inquiry repeatedly encounters T*-supply sockets. Finite observation records can support finite fit internally. They cannot, by themselves, generate unrestricted future truth, universal induction, or final explanation. When a theory projects beyond its finite source horizon, BEDC does not hide that projection as an axiom. It records the socket, the requested supply kind, the audit gate, and the failure surface.*

Remark 9414.520 (Closure obligations)**.** *Closing this vision-seed horizon would require a classifier linking focused* $\mathsf{ConsciousRecord}$ *source traces to public scientific source rows;* hsame*-stable and inter-*Hist*-stable transport for replicated scientific observations; fit classifiers between observed and model-generated signatures; stability and descent obligations for every predictive or explanatory reuse; failure surfaces for lawlike appearance;* T*-socket rows for future-binding claims not generated by current finite records; and audit gates refusing subject-substance, global-clock, hidden-choice, unledgered-compression, and object-level* T *promotions.*

Concretized in §1273: Scientific model bridges supply the public model, signature, classifier, stability, descent, ledger, failure-surface, and audit-strength discipline consumed by this schema.

Concretized in §1247: Focused ledgers supply the records-side carrier for focused Hist-time readback without adding subject-substance.

Concretized in §479: Observer-Hist identity supplies observer as Hist and observation as local continuation without adding a global observer.

Concretized in §528: Supply socket ledgers record future-binding requests as finite socket rows rather than hidden axioms or model components.

9414.55 The Position of Consciousness

Remark 9414.521 (Vision seed status)**.** *This section has vision-seed status: it is a schema horizon and has no Lean target. It does not add consciousness as a new BEDC kernel object and does not introduce a subject-substance carrier. It records a location principle: on the records side, consciousness appears as the self-proxy surface of a focused* Hist*; on the inscription side, the self-center appears as the local inscription point of* T *on that focused observational chain.*

The section forbids the strong equations

$$\mathsf{Consciousness} = \mathsf{T}, \qquad \mathsf{Consciousness} = \mathsf{SubjectSubstance}, \qquad \mathsf{Consciousness} = \mathsf{GlobalTime}, \qquad \mathsf{SelfCenter}(C,t) =$$

The safe reading is

$$\mathsf{Consciousness} = \mathsf{RecordsSide}(\mathsf{FocusedHistSelfProxy}) + \mathsf{InscriptionSide}(\mathsf{LocalTInscription}),$$

or, more concretely,

$$\mathsf{Consciousness} = (H(F), \Theta(F), S(F), \rho) + \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t).$$

The first summand is the object-layer recordable carrier. The second is an aspectual boundary reading and must not be internalized as a BEDC object.

Definition 9414.522 (Focused observational chain)**.** *Given a finite focused multi-*Hist *ledger*

$$L_N = (I, H, \Theta, F, S, D, G, R, J, Q, \rho, A),$$

where I is a finite index ledger, $H : I \to \mathsf{Hist}$ assigns histories, Θ records local continuation traces, $F \in I$ is the focus index, S is the selector ledger, ρ is the focused self-proxy, and A is the audit row, the focused observational chain *is*

$$C_F := (H(F), \Theta(F), S(F), \rho).$$

Here $H(F)$ is the focused generated Hist*, $\Theta(F)$ is the local* Cont *trace from* e *to $H(F)$, $S(F)$ records which visible transitions are selected into the chain, and ρ is the endpoint and trace self-proxy.*

Definition 9414.523 (Records-side conscious carrier)**.** *For a focused ledger L_N and focus F, the records-side conscious carrier is*

$$\mathsf{ConsciousRecord}(L_N, F) := (H(F), \Theta(F), S(F), \rho).$$

It is the records-side carrier of a consciousness reading, not an object-level definition of phenomenal essence. Thus the recordable BEDC location consists of focused generated history, local continuation trace, selector ledger, and self-proxy.

Definition 9414.524 (Inscription-side self-center)**.** *Let* T *denote the non-internalizable apophatic far end. For a focused observational chain C_F at local stage t, its inscription-side self-center is*

$$\mathsf{SelfCenter}(C_F, t) := \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t).$$

This definition is local and aspectual. It says that a local accumulation event of the focused chain is read, on the inscription side, as a local inscription point of T*. It does not say* $\mathsf{SelfCenter}(C_F, t) = \mathsf{T}$*.*

Definition 9414.525 (Position of consciousness)**.** *The position of consciousness is the double-aspect location*

$$\mathsf{Pos}_{\mathsf{Con}}(L_N, F, t) := (\mathsf{ConsciousRecord}(L_N, F), \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)).$$

The first component is the records-side carrier

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho).$$

The second component is the inscription-side reading

$$\mathsf{InscriptionPoint}(\mathsf{T}, C_F, t).$$

Thus the position of consciousness is not a single entity-location. It is the double-aspect position consisting of a records-side focused self-proxy surface and an inscription-side local T*-inscription point.*

Definition 9414.526 (Role discipline)**.** *This vision-seed schema keeps the following roles separate:*

$$\begin{aligned} H(F) &= \textit{focused generated } \mathsf{Hist}, \\ \Theta(F) &= \textit{local continuation trace}, \\ \rho &= \textit{focused self-proxy/readback}, \\ \mathsf{T} &= \textit{non-internalizable apophatic far end}. \end{aligned}$$

The equation

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)$$

expresses a local inscription reading, not object-level equality with T*. Any presentation that collapses these roles into one object-level equality is a forbidden collapse.*

Theorem 9414.527 (Consciousness adds no subject-substance)**.** $\mathsf{ConsciousRecord}(L_N, F)$ *does not add a new subject-substance to BEDC.*

Proof. By Definition 9414.523,

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho).$$

All four components are already contained in the finite focused ledger L_N: $H(F)$ is an existing Hist, $\Theta(F)$ is an existing Cont trace, $S(F)$ is a selector ledger, and ρ is a focused self-proxy/readback row. The definition adds no subject parameter outside Hist and no new identity criterion. Observer identity remains governed by hsame. An additional subject-substance carrier would require transport laws across Cont, which is exactly the structure refused by the no-subject discipline of Definition 9418.11. Thus $\mathsf{ConsciousRecord}(L_N, F)$ is a records-side focused carrier, not a new subject-substance. □

Corollary 9414.528 (Consciousness is not reified)**.** *BEDC does not reify consciousness as a new ontological object:*

$$\mathsf{Consciousness} \neq \mathsf{SubjectSubstance}.$$

Proof. Reifying consciousness as a subject-substance would require a carrier independent of Hist and transport laws for that carrier through continuation, selector, self-proxy, and inter-Hist relation rows. But Definition 9414.523 writes the records-side carrier entirely as the four fields $H(F), \Theta(F), S(F), \rho$ already inside L_N. No additional carrier is introduced. □

Theorem 9414.529 (Consciousness is local Hist-time, not global time)**.** *The position of consciousness is a focused readback surface of local Hist-time, not a global time object:*

$$\mathsf{Consciousness} \neq \mathsf{GlobalTime}.$$

Proof. BEDC kernel time is given by Hist growth. A time step is one $\mathsf{E1}/\mathsf{E0}$ distinction applied to a history, and a local time stream is a Cont trace from e to a generated history. For the focused ledger L_N, the conscious carrier uses $H(F)$ and its local trace $\Theta(F)$. This structure does not synchronize all histories, does not define a global clock, and does not supply a unified time axis across all observers. The focus F selects one local chain; it does not generate a global observer time. Hence consciousness is local Hist-time read through a focused self-proxy, not global time. □

Corollary 9414.530 (Other-consciousness comparison requires relation rows)**.** *Comparison between distinct focused chains cannot be made through a global consciousness time. It requires inter-*Hist *relation, invariant, dependence, and audit rows such as* R, J, Q, A.

Proof. By Definition 9414.529, consciousness is a focused local Hist-time surface. Comparing two focuses F_i and F_j therefore requires displayed rows relating $H(F_i), \Theta(F_i)$ to $H(F_j), \Theta(F_j)$: relation rows, classifier-stable invariant rows, dependence rows, or rate rows. Without these ledger fields, BEDC does not license a direct comparison inside one global consciousness clock. □

Theorem 9414.531 (SelfCenter is local inscription, not T)**.** *The equation*

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)$$

does not imply

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{T}.$$

Proof. Definition 9414.524 contains three roles: the far end T, the focused chain C_F, and the local stage t. It is a local inscription point of T on a particular chain at a particular event. It is not T itself. Writing $\mathsf{SelfCenter}(C_F, t) = \mathsf{T}$ would internalize the apophatic far end as an object-level term and would erase the local chain and local stage. This is precisely the collapse refused by Definition 9414.526. □

Corollary 9414.532 (Consciousness exposes T-sockets but is not T)**.** *Consciousness may expose* T*-supply sockets, but consciousness is not* T*:*

$$\mathsf{Consciousness} \neq \mathsf{T}.$$

Proof. A focused self-proxy can carry forward-binding commitments: that the trace continues, that the trace is read as mine, that the local experience has meaning beyond the displayed record, or that an unrecorded next moment is expected. Such commitments exceed the current finite record and therefore expose T-supply sockets. But exposing a socket is not becoming the socket's far end. T is the non-internalizable supply position; $\mathsf{ConsciousRecord}(L_N, F)$ is a records-side carrier. They occupy different strata. □

Theorem 9414.533 (Consciousness has a double-aspect position)**.** *The position of consciousness is*

$$\mathsf{Pos}_{\mathsf{Con}}(L_N, F, t) = \big(\mathsf{ConsciousRecord}(L_N, F), \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)\big).$$

Proof. From the records side, consciousness must be written as finite ledger rows. Definition 9414.523 gives

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho),$$

which records the focused Hist, local time trace, selector ledger, and self-proxy. From the inscription side, the same local accumulation event is read as a local actualization at the boundary:

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t).$$

This does not add a second event. It gives an aspectual reading of the same event: records-side entry into the ledger and inscription-side local inscription of T. Therefore consciousness is not located in a single object but in the double-aspect position formed by these two readings. □

Definition 9414.534 (Experience-position equivalence)**.** *Two consciousness positions*

$$\mathsf{Pos}_{\mathsf{Con}}(L_N, F, t) \quad \textit{and} \quad \mathsf{Pos}_{\mathsf{Con}}(L'_N, F', t')$$

are experience-equivalent only when their records-side carriers and their local inscription readings agree at the displayed resolution. Thus the records-side requirements are

$$\begin{gathered}\mathsf{hsame}\big(H(F), H'(F')\big),\\ \Theta(F) \sim_\Theta \Theta'(F'),\\ S(F) \sim_S S'(F'),\\ \rho \sim_\rho \rho',\end{gathered}$$

and the inscription-side requirement is not bare access to the common far end T*, but agreement of the local chain and stage on which the inscription is read:*

$$\mathsf{InscriptionPoint}(\mathsf{T}, C_F, t) \sim_{\mathrm{loc}} \mathsf{InscriptionPoint}(\mathsf{T}, C'_{F'}, t').$$

The relation $\sim_{\mathrm{loc}}$ *is a vision-layer locality comparison; it is licensed only by the displayed focused chains, stages, selector rows, and self-proxy rows. It is not kernel equality and not an equality of* T *with either local self-center.*

Theorem 9414.535 (Common T does not collapse experience)**.** *The shared apophatic far end* T *is insufficient to identify two experience positions. In particular, if either*

$$\neg\mathsf{hsame}\big(H(F), H'(F')\big)$$

or one of the displayed locality comparisons for Θ*,* S*,* ρ*, or the local stage fails, then*

$$\mathsf{Pos}_{\mathsf{Con}}(L_N, F, t) \not\sim_{\mathrm{exp}} \mathsf{Pos}_{\mathsf{Con}}(L'_N, F', t')$$

even though both positions expose the same boundary name T*.*

Proof. By Definition 9414.525, each consciousness position has two components: a records-side carrier and an inscription-side local reading. The records-side carrier is

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho),$$

and analogously for (L'_N, F'). By Definition 9414.534, experience equivalence requires agreement of these displayed rows before the inscription-side reading is compared.

The role of T is different. It is the non-internalizable apophatic far end named by the boundary discipline; it is not a shared records-side store, not a global memory, and not a subject-substance. Therefore the fact that both local inscription expressions contain the same symbol T supplies no hsame witness between $H(F)$ and $H'(F')$, no comparison between the two local Cont traces, no selector agreement, no self-proxy agreement, and no equality between local stages. If any of those displayed comparisons fails, the double-aspect pair fails to be experience-equivalent. The common boundary name remains common only as apophatic naming, not as an object-level collapse of experiences. □

Corollary 9414.536 (Different histories can share the same boundary name)**.** *Distinct focused histories may expose the same* T*-supply boundary without thereby carrying the same experience:*

$$\neg\mathsf{hsame}\big(H(F), H'(F')\big) \quad\Longrightarrow\quad \mathsf{Pos}_{\mathsf{Con}}(L_N, F, t) \not\sim_{\mathrm{exp}} \mathsf{Pos}_{\mathsf{Con}}(L'_N, F', t').$$

Proof. This is the history component of Definition 9414.535. Experience equivalence requires records-side agreement before any inscription-side comparison can be read. Different focused histories therefore remain different local experience positions even when their boundary pressure is named by the same T. □

Corollary 9414.537 (Brain-state and experience are two readings of one event)**.** *In this vision-seed schema, brain state or bodily state is not the external causal maker of consciousness, and consciousness is not an external mover of the brain. They are two readings of one accumulation event:*

$$\mathsf{BrainState}(C_F, t) = \textit{records-side observable row},$$
$$\mathsf{Experience}(C_F, t) = \textit{inscription-side reading of the same event.}$$

Proof. Brain states, bodily states, neural activity, and behavioral outputs enter BEDC only as observation, history, signature, classifier, or ledger rows. These are records-side rows. First-person experience is not another records-side object beside them; it is the inscription-side reading of the same local event. Therefore BEDC does not need either $\mathsf{Brain} \to \mathsf{Consciousness}$ or $\mathsf{Consciousness} \to \mathsf{Brain}$ as a relation between two substances. □

Theorem 9414.538 (Consciousness is not identical with observation, distinction, or time)**.** *Although the kernel reading identifies observation, distinction, and local time-step through* Cont *and* E1/E0 *growth, consciousness is not identical with bare observation, bare distinction, or bare time-step.*

Proof. Observation, distinction, and local time-step give the minimal occurrence form

$$h \xrightarrow{\mathsf{E0/E1}} h'.$$

Consciousness requires additional structure:

$$H(F), \Theta(F), S(F), \rho.$$

There must be focus, local trace, selector ledger, and self-proxy/readback. A history may grow without being selected as a focused conscious record; a distinction may occur without a self-proxy; a local time step may be recorded without forming a conscious reading. Thus the kernel-facing identity of observation, distinction, and local time-step must not be promoted to the vision-layer equation $\mathsf{Consciousness} = \mathsf{Time}$. □

Corollary 9414.539 (Safe slogan)**.** *The safe compressed readings are*

$$\mathsf{Consciousness} = \mathsf{FocusedSelfProxy}(\mathsf{HistTime}) + \mathsf{LocalTInscription},$$

and

$$\mathsf{Consciousness} = \mathsf{RecordsSide}(H(F), \Theta(F), S(F), \rho) + \mathsf{InscriptionSide}\big(\mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)\big).$$

The forbidden slogans are

$$\mathsf{Consciousness} = \mathsf{T}, \qquad \mathsf{Consciousness} = \mathsf{Subject}, \qquad \mathsf{Consciousness} = \mathsf{BrainObject}, \qquad \mathsf{Consciousness} = \mathsf{GlobalTim}$$

Proof. The safe readings preserve records-side carrier fields and inscription-side boundary reading. The forbidden slogans respectively violate no-internal-T, no-subject-substance, the records/inscription two-aspect discipline, and no-global-clock. □

Remark 9414.540 (Reading)**.** *The BEDC answer is neither that consciousness is in the brain, nor that it is in a soul-substance, nor that it is in* T*, nor that it is in a global time. On the records side, consciousness is located at*

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho),$$

the self-proxy surface of a focused Hist*. On the inscription side, the self-center is located at*

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t),$$

the local inscription point of T *on that focused chain.*

Thus consciousness is the double-aspect position where local Hist-time reads itself as focused self-proxy and is locally inscribed by T*. This preserves first-person centeredness without adding a subject-substance; it acknowledges the* T *boundary without identifying consciousness with* T*; and it allows brain and body records to matter without reducing experience to a bare records-side object.*

Concretized in §1109: Focused multi-Hist ledgers realize the finite records-side carrier with generated Hist rows, local Cont traces, focus, selector ledger, self-proxy, and audit rows.

Concretized in §479: Observer-Hist identity realizes the no-subject observer reading through Hist, hsame, Cont, probe/signature observations, and no-subject ledgers.

Concretized in §1085: Digest inscription packets realize visible digest, observation fiber, apophatic far-end boundary, local self-center inscription, observer-Hist identity, transport, continuation, provenance, and local naming rows.

Concretized in §519: Inscription event packets realize the local records-side and inscription-side transition as a finite NameCert surface without internalizing T.

9414.56 Egocentric interface and mentation dynamics

Remark 9414.541 (Vision seed status)**.** *This section has vision-seed status and no Lean target. It refines the focused conscious carrier of Definition 9414.523 by recording how a focused* Hist *receives a source through a centered interface, attention weights, classifiers, section choices, probe policies, and an append-only ledger. It does not add a subject-substance, does not identify the observer with a bare spatial origin, and does not claim that internal mentation changes the external source. The safe claim is narrower: a source may remain fixed while the world-for-this-*Hist *changes because the interface packet, attention surface, classifier, section, probe policy, or ledger has changed.*

Definition 9414.542 (Egocentric interface packet)**.** *For a focused history h, an* egocentric interface packet *is a finite records-side packet*

$$\mathcal{E}_h := (h, C_h, D, a, A_h, K_h, \sigma_h, \Pi_h, \Lambda_h, \mathsf{Aud}_h).$$

The rows have the following roles:

1. *h is the focused generated* Hist*;*
2. *$C_h = (o_h, \Omega_h, B_h)$ is the centered frame row, with origin o_h, orientation row Ω_h, and body or sense-boundary row B_h;*
3. *D is the displayed interface support, such as a visual disk or other finite readout surface;*
4. *$a : D \to [0, 1]$ is a clarity kernel on the support;*
5. *$A_h : D \to \mathbb{R}_{\geq 0}$ is the attention-weight row;*
6. *K_h is the classifier or naming surface applied to the readout;*
7. *σ_h is a licensed section row for a compressed interface surface, when such a section is displayed;*
8. *Π_h is the probe policy selecting admissible future interfaces;*
9. *Λ_h is the local ledger of frame, mentation, classifier, section, and probe records;*
10. *Aud_h records the refusals against bare source access, unledgered section choice, subject-substance import, and outcome choice by fiat.*

The centering convention $o_h = 0$ is a gauge row. It says that the readout is coordinated from the focused history's interface point; it does not say that the physical source is ontologically centered on that point.

Definition 9414.543 (Interface frame and world-for-this-Hist)**.** *Let X_t be a displayed source stage and let*

$$\mathrm{Render}_{C_h}(X_t) : D \to Y$$

be the readout exposed by the centered frame. The interface frame is

$$F_h := V_h(X_t), \qquad F_h(u) := \big(a(u), \mathrm{Render}_{C_h}(X_t)(u)\big).$$

When a scalar readout row is displayed, the effective attended read is written

$$F_h^{\mathrm{eff}}(u) := a(u)A_h(u)\,\mathrm{Render}_{C_h}(X_t)(u).$$

The world-for-this-Hist *is the finite packet*

$$W_h := (F_h, A_h, K_h, \sigma_h, \Pi_h, \Lambda_h).$$

Thus W_h is not the bare source X_t. It is the source as rendered by a centered interface and then filtered by clarity, attention, classification, section, probe policy, and ledger rows.

Theorem 9414.544 (Experience enters as an interface frame)**.** *If a focused history accesses a source stage only through the interface packet of Definition 9414.542, then the record added to the focused chain is an interface frame, not the bare source:*

$$h_{t+1} = \mathsf{Cont}(h_t, F_{h_t}).$$

Proof. The packet supplies a readout map V_{h_t} from the displayed source stage to the interface support. By Definition 9414.543, the visible record is $F_{h_t} = V_{h_t}(X_t)$. Since the focused chain grows by Cont on displayed records, the continuation row appends F_{h_t}, not X_t itself. Bare source access would require a row outside the interface packet and is refused by Aud_{h_t}. □

Definition 9414.545 (Source-fixed scan)**.** *A* source-fixed scan *from h to h' is a transition for which the source stage is unchanged while the centered frame row changes:*

$$X_{t+1} = X_t, \qquad C_{h'} = g \cdot C_h$$

for a displayed motion or reorientation row g.

Theorem 9414.546 (A source-fixed scan can change the frame)**.** *Let $h \to h'$ be a source-fixed scan. If there is a support point $u \in D$ such that*

$$a(u) > 0$$

and

$$\mathrm{Render}_{C_{h'}}(X_t)(u) \not\sim_Y \mathrm{Render}_{C_h}(X_t)(u),$$

then

$$F_{h'} \not\sim F_h.$$

Proof. The source-fixed scan keeps X_t fixed but replaces the frame row C_h by $C_{h'}$. At the displayed point u, the clarity row is nonzero and the two rendered values fail the displayed Y-classifier comparison. By the definition of F_h, the frame records include the clarity row together with the rendered value at each support point. Hence the two frames fail the displayed frame classifier at u. □

Corollary 9414.547 (Projection change is not source change)**.** *A source-fixed scan may satisfy*

$$\Delta X = 0, \qquad \Delta F \neq 0.$$

The change belongs to the projection from source to interface frame, not to the bare source itself.

Proof. This is the case described by Definition 9414.546. The source equality is part of Definition 9414.545; the frame inequality follows from the displayed render difference. □

Definition 9414.548 (Mentation event)**.** *A* mentation event *at h is an internal record*

$$n \in \mathcal{M}_h$$

that may enter the focused history by

$$h' = \mathsf{Cont}(h, n)$$

without being a fresh external source frame. Its packet-level update is a map

$$\mathcal{U}_n : W_h \to W_{h'}$$

of the form

$$\mathcal{U}_n(F_h, A_h, K_h, \sigma_h, \Pi_h, \Lambda_h) = (F_h, A_{h'}, K_{h'}, \sigma_{h'}, \Pi_{h'}, \mathrm{append}(\Lambda_h, n)).$$

The event may record an attention shift, intention, judgement, naming move, expectation, question, goal, memory activation, or internal image. The external source row is not changed by this definition.

Theorem 9414.549 (Ledgered mentation changes the world-for-this-Hist)**.** *Assume n is not already present in the displayed endpoint of Λ_h, and assume ledger append is visible at the active ledger classifier. If*

$$h' = \mathsf{Cont}(h, n)$$

and

$$\Lambda_{h'} = \mathrm{append}(\Lambda_h, n),$$

then

$$W_{h'} \neq W_h$$

even in the source-fixed case $X_{t+1} = X_t$.

Proof. By Definition 9414.543,

$$W_h = (F_h, A_h, K_h, \sigma_h, \Pi_h, \Lambda_h).$$

By the mentation update in Definition 9414.548, the last component becomes $\mathrm{append}(\Lambda_h, n)$. The hypotheses state that this append is visible at the active ledger classifier. Hence the sixth component of $W_{h'}$ differs from the sixth component of W_h. Equality of the external source row does not remove this ledger difference, because W_h is a packet containing the ledger, not the bare source. □

Theorem 9414.550 (Attention change changes the effective read)**.** *Suppose the scalar effective read row of Definition 9414.543 is displayed. If*

$$F_{h'} = F_h$$

as interface frames, but there exists $u \in D$ such that

$$a(u)\,\mathrm{Render}_{C_h}(X_t)(u) \neq 0$$

and

$$A_{h'}(u) \neq A_h(u),$$

then

$$F^{\mathrm{eff}}_{h'}(u) \neq F^{\mathrm{eff}}_{h}(u).$$

Proof. At the displayed support point u, the scalar source factor $a(u)\,\mathrm{Render}_{C_h}(X_t)(u)$ is nonzero. The effective read is obtained by multiplying this factor by the attention weight. Since the attention weights differ at u, the effective reads differ at u. □

Theorem 9414.551 (Section update changes section-relative meaning)**.** *Let Q_h be the compressed interface surface determined by the current visual quotient and let $\sigma_h, \sigma_{h'} : Q_h \to X$ be two displayed licensed sections. If a label row $L : X \to \{0, 1\}$ satisfies*

$$L(\sigma_h(q)) \neq L(\sigma_{h'}(q))$$

for some $q \in Q_h$, then the section-relative meanings differ:

$$L_{\sigma_h}(q) \neq L_{\sigma_{h'}}(q).$$

Proof. By Definition 9414.65,

$$L_{\sigma_h}(q) = L(\sigma_h(q)) \quad \text{and} \quad L_{\sigma_{h'}}(q) = L(\sigma_{h'}(q)).$$

The displayed label inequality at q is exactly the inequality between these two section-relative readouts. The compressed surface Q_h may be unchanged; the licensed section row still matters. □

Theorem 9414.552 (Probe-policy change changes the future interface surface)**.** *If two world-for-this-*Hist *packets agree on the current frame but their probe policies satisfy*

$$\Pi_h(h, a) = I \quad \textit{and} \quad \Pi_{h'}(h', a) = I'$$

for some displayed action row a with $I \not\sim I'$, then the next admissible probe interface differs.

Proof. The next interface family is selected by the probe policy row. This is the role of Π in Definition 9414.70. If the two policies return non-equivalent interface families for the same displayed action row, the selected future interface surface differs. By Definition 9414.71, this is a change of probe interface, not a fiat determination of the outcome. □

Theorem 9414.553 (Egocentric mentation synthesis)**.** *For an egocentric interface packet, current experience is an interface frame written into* Hist*; source-fixed scanning can change the frame; ledgered mentation changes W_h by append-only record; attention updates can change the effective read of the same frame; section updates can change section-relative meaning over the same compressed surface; and probe-policy updates can change the future interface surface without selecting the future outcome by fiat.*

Proof. The frame-entry claim is Definition 9414.544. The source-fixed scan claim is Definition 9414.546. The ledgered mentation claim is Definition 9414.549. The attention claim is Definition 9414.550. The section claim is Definition 9414.551. The probe-policy claim is Definition 9414.552. Each component is a packet row already admitted by the focused-record and interface–constraint–section disciplines, and none introduces a subject-substance or unsupported external-source alteration. □

Remark 9414.554 (Forbidden readings)**.** *This packet forbids four readings. The centered frame is not a claim that the universe is centered on the observer. The interface frame is not the bare source. A mentation event does not alter the external source merely by being recorded. A changed probe policy is not an outcome chooser; it selects an interface family whose outcome still depends on the displayed source and kernel rows. The admissible compressed statement is*

$$\Delta X = 0 \quad \textit{and} \quad \Delta W \neq 0$$

whenever the source is fixed and at least one visible row of $W_h = (F_h, A_h, K_h, \sigma_h, \Pi_h, \Lambda_h)$ *changes.*

Concretized in §928: Observer-interface packets realize centered readout, support, interface, classifier, boundary, and ledger rows for finite observer surfaces.

Concretized in §927: Attention ledgers realize finite attention weights, salience routes, update records, and NameCert rows for interface-weighted readouts.

Concretized in §1047: Observer-direction packets realize finite orientation, frame, boundary, and direction rows for centered observer readouts.

9414.57 Public and Local Reflective Compilation

Remark 9414.555 (Vision seed status)**.** *This section has vision-seed status: it is a schema horizon and has no Lean target. It records the structural comparison between public scientific inquiry, local consciousness-like readback, the compiler-side audit of observation flows, the MetaCIC-side audit of claim legality, and the non-internalizable forward-supply position* T.

The intended reading is

$$\begin{aligned}
\mathsf{ScientificInquiry} &= \mathsf{GroundCompilerLike}(\mathsf{ObservationFlow}) + \mathsf{MetaCICLike}(\mathsf{ClaimLegality}),\\
\mathsf{Consciousness} &= \mathsf{LocalSelfCompile}(\mathsf{HistTime}) + \mathsf{LocalSelfAudit}(\mathsf{Readback}),\\
\mathsf{I} &= \mathsf{CurrentSelfProxy}(\mathsf{ConsciousRecord}(L_N, F)),\\
\mathsf{T} &= \mathsf{NonInternalizableForwardSupplyPosition}.
\end{aligned}$$

These equations are role schemata, not object-level identifications between the named layers. In particular, the section refuses the collapses

$$\mathsf{Science} = \mathsf{Consciousness}, \qquad \mathsf{Science} = \mathsf{T}, \qquad \mathsf{Consciousness} = \mathsf{T}, \qquad \mathsf{I} = \mathsf{T}, \qquad \mathsf{GroundCompiler} = \mathsf{MetaCIC}.$$

Definition 9414.556 (Twin public audit substrate)**.** *The* twin public audit substrate *of a scientific inquiry context is the ordered pair*

$$\mathsf{TwinAudit} := (\mathsf{CompilerSide}, \mathsf{MetaAuditSide}).$$

The compiler side sends finite observation or event traces through source separation, encoding, generated recognition, certificate gates, ledger rows, and cannot-claim rows. This is the role played by the ground compiler chapters, especially chapter 9207, chapter 9221, and chapter 9222.

The meta-audit side checks whether a displayed claim is licensed by its source, classifier, stability domain, descent data, ledger exposure, and failure surface. This is the scientific analogue of the MetaCIC self-study role of Definition 9414.411 and Definition 9414.419.

Definition 9414.557 (Scientific inquiry as public reflective compilation)**.** *A scientific inquiry packet has* public reflective compilation *when it factors through the twin public audit substrate:*

$$\mathsf{ScientificInquiry} = \mathsf{CompilerSide}(\mathsf{ObservationFlow}) + \mathsf{MetaAuditSide}(\mathsf{ClaimLegality}).$$

The first component compiles finite observation flow into source traces, observed signatures, model-generated signatures, recognizers, certificate gates, and ledgers. The second component audits what those compiled rows may legitimately support: fit, lawlike reuse, explanation, prediction, or only speculative appearance.

Definition 9414.558 (Local self-compiler and local self-audit)**.** *For a focused ledger* (L_N, F) *with*

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho),$$

the local self-compiler *is the records-side role*

$$\mathsf{SelfCompiler}(L_N, F) : \mathsf{SelfFlow} \to \mathsf{FocusedTrace} \to \mathsf{SelectorLedger} \to \mathsf{SelfProxy}.$$

The local self-audit *is the readback role*

$$\mathsf{SelfMetaAudit}(L_N, F) : \mathsf{SelfProxy} \to \mathsf{Readback} \to \mathsf{BoundaryRecognition} \to \mathsf{GapExposure}.$$

These roles are local and records-side. They do not assert a public scientific certificate and do not define a total theory of consciousness.

Definition 9414.559 (Current self-proxy)**.** *The "I" reading of a focused ledger is the current readback center supported by the focused self-proxy:*

$$\mathsf{I}(L_N, F) := \mathsf{CurrentSelfProxy}(H(F), \Theta(F), S(F), \rho).$$

Equivalently, on the records side it is the current centered use of ρ *in* $\mathsf{ConsciousRecord}(L_N, F)$*. On the inscription side it may be read through*

$$\mathsf{SelfCenter}(C_F, t) = \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t),$$

as in Definition 9414.524. This does not identify the current self-proxy with T*.*

Theorem 9414.560 (GroundCompiler and MetaCIC roles do not collapse)**.** *The compiler-side role and the MetaCIC-side role are complementary but not identical:*

$$\mathsf{GroundCompiler} \neq \mathsf{MetaCIC}.$$

Proof. The ground compiler side answers whether an event flow, source channel, displayed code artifact, generated recognizer, report, or certificate gate has the status being claimed for it. Its characteristic refusals include: code is not proof, report is not evidence, metric similarity is not identity, and hidden host structure is not formal input.

The MetaCIC side answers whether a closed-calculus claim is licensed by term formation, typing, reduction, substitution, normalization, confluence, and the open meta-boundary. A ground compiler theorem that a report cannot discharge a certificate does not prove subject reduction. A MetaCIC theorem about substitution does not certify that an observation report has no hidden host input. Therefore the two roles are structurally linked but cannot be collapsed. □

Corollary 9414.561 (Self-compilation requires twin audit)**.** *A BEDC self-compilation story that is meant to be public and claim-bearing must carry both compiler-side provenance and MetaCIC-side claim legality.*

Proof. Without compiler-side provenance, a displayed artifact may smuggle a host convention, a report, or a code path into the role of formal input or proof. Without MetaCIC-side claim legality, a generated artifact may be well reported while still failing to support the theorem, reduction, substitution, or boundary claim attached to it. Public self-compilation therefore needs both sides of Definition 9414.556. □

Theorem 9414.562 (Scientific inquiry has the twin audit form)**.** *Scientific inquiry is compiler-like on the source side and MetaCIC-like on the claim side:*

$$\mathsf{ScientificInquiry} = \mathsf{GroundCompilerLike}(\mathsf{ObservationFlow}) + \mathsf{MetaCICLike}(\mathsf{ClaimLegality}).$$

Proof. On the source side, scientific inquiry begins with finite observation flows: instrument outputs, experimental rows, traces, images, simulations, or public reports. These rows do not become laws merely by being displayed. They must be separated as source, normalized or encoded where needed, converted into observed signatures, related to model-generated signatures, and admitted through classifier and ledger discipline. This is the ground-compiler-like direction.

On the claim side, the same rows must be audited for what they license. A finite fit may support a local comparison, a stability certificate may support reuse in an announced domain, a descent witness may support transport through a classifier, and a failure surface names how the claim can be defeated. These are not source-format questions; they are legality questions about the supported claim. Thus scientific inquiry has the twin audit form of Definition 9414.556. □

Corollary 9414.563 (Science is not either substrate alone)**.** *Scientific inquiry is not identical with the ground compiler and is not identical with MetaCIC.*

Proof. The ground compiler side can produce ledgered artifacts and cannot-claim rows, but by itself it does not provide the empirical target, model family, stability domain, descent obligations, or failure surface of a scientific theory. The MetaCIC side can audit closed-calculus legality, but by itself it does not generate reality-facing observed signatures. Scientific inquiry uses both roles analogically and structurally; it is not either module as an object. □

Theorem 9414.564 (Consciousness-like readback has local twin form)**.** *Within this vision-seed schema, consciousness-like readback has the local form*

$$\mathsf{Consciousness} = \mathsf{LocalSelfCompile}(\mathsf{HistTime}) + \mathsf{LocalSelfAudit}(\mathsf{Readback}).$$

Proof. By Definition 9414.523, the records-side carrier of the consciousness reading is

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho).$$

The focused history $H(F)$ is the current generated endpoint, the trace $\Theta(F)$ records its local Hist-time route, the selector ledger $S(F)$ records which visible transitions enter the focused chain, and ρ supplies the self-proxy/readback surface. These fields compile a local Hist-time flow into a self-readable carrier.

The same carrier also supports local boundary recognition: the record can distinguish present readback, memory, expectation, uncertainty, and exposed gaps without becoming a public scientific certificate or a total self-truth predicate. This is a local self-audit role, not a global MetaCIC closure. Hence consciousness-like readback has local twin form. □

Corollary 9414.565 (Self-awareness is not a total theory of consciousness)**.** *Local self-readback does not provide an internal total theory of consciousness.*

Proof. Self-awareness uses the finite carrier $(H(F), \Theta(F), S(F), \rho)$ at a focused endpoint. A total theory of consciousness would require the system to close over consciousness as such, including its own truth and boundary conditions. That would convert finite readback into a total self-validator and would close the meta-loop that Definition 9414.420 keeps open. □

Theorem 9414.566 (The current self-proxy is not subject-substance or T)**.** *The "I" reading is neither a subject-substance nor* T*:*

$$\mathsf{I} \neq \mathsf{SubjectSubstance}, \qquad \mathsf{I} \neq \mathsf{T}.$$

Proof. By Definition 9414.559, the "I" reading is the current self-proxy use of $H(F), \Theta(F), S(F), \rho$. These fields belong to the focused ledger and add no carrier behind Hist. Thus the current self-proxy is not a subject-substance.

On the inscription side, the self-center may be read as $\mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)$. This expression still contains the focused chain C_F and the local stage t. It names a local inscription point of T on a chain, not the far-end supply position itself. Identifying the current self-proxy with T would internalize the non-internalizable boundary position as a recordable object. □

Theorem 9414.567 (Science, consciousness, and I expose T-sockets without becoming T)**.** *Scientific inquiry, consciousness-like readback, and the current self-proxy may expose* T*-supply sockets, but none is identical with* T*.*

Proof. Scientific inquiry exposes T-supply sockets when finite observation records are asked to support future prediction, universal lawlike reuse, cross-regime extrapolation, or final explanation beyond the displayed witnesses. Consciousness-like readback exposes such sockets when finite records are asked to bind self-continuity, expected next moments, or meaning beyond the current record. The current self-proxy exposes the same boundary whenever the focused endpoint reads its trace as continuing beyond what is displayed.

In each case, the socket is a ledgered boundary row. It states the missing shape, the consumption site, and the audit gate preventing internalization. By Definition 9414.515 and Definition 9414.531, socket exposure does not supply the far end as an object. Hence these structures expose T without becoming T. □

Theorem 9414.568 (Public science and local consciousness-like readback)**.** *Scientific inquiry can be the public ledgered mode of consciousness-like reflection, but it is not consciousness itself:*

$$\mathsf{Science} = \mathsf{PublicLedgeredMode}(\mathsf{ConsciousnessLikeReflection}(\mathsf{HistTime})), \qquad \mathsf{Science} \neq \mathsf{Consciousness}.$$

Proof. The first relation is the safe reading established in Definition 9414.517. A focused consciousness-like record can supply source readback, and scientific inquiry can externalize that source through observed signatures, model-generated signatures, classifiers, approximation towers, stability, descent, ledgers, failure surfaces, and audit strength.

The inequality follows because the records-side conscious carrier has only the focused history, local trace, selector ledger, and self-proxy. Scientific inquiry adds public evidence fields and inter-Hist transport obligations. A private readback may exist without public scientific audit, and a public scientific packet is not itself a private first-person carrier. □

Corollary 9414.569 (Safe compressed reading)**.** *The safe compressed reading is*

$$\begin{gathered}\mathsf{Observation} = \mathsf{Distinction} = \mathsf{LocalTimeStep},\\ \mathsf{Observer} = \mathsf{Hist},\\ \mathsf{Science} = \mathsf{GroundCompilerLike}(\mathsf{ObservationFlow}) + \mathsf{MetaCICLike}(\mathsf{ClaimLegality}),\\ \mathsf{Consciousness} = \mathsf{LocalSelfCompile}(\mathsf{HistTime}) + \mathsf{LocalSelfAudit}(\mathsf{Readback}),\\ \mathsf{I} = \mathsf{FocusedSelfProxy}(\mathsf{HistTime}),\\ \mathsf{T} = \mathsf{NonInternalizableForwardSupplyPosition}.\end{gathered}$$

The corresponding refusals are

$$\mathsf{Science} \neq \mathsf{Consciousness}, \qquad \mathsf{Science} \neq \mathsf{T}, \qquad \mathsf{Consciousness} \neq \mathsf{T}, \qquad \mathsf{I} \neq \mathsf{T}, \qquad \mathsf{I} \neq \mathsf{SubjectSubstance}.$$

Proof. The first two readings are the kernel-facing observation and observer discipline used throughout the focused-ledger sections. The science reading follows from Definition 9414.562. The consciousness reading follows from Definition 9414.564. The "I" reading follows from Definition 9414.559. The T reading follows from the non-internalizable supply discipline used in Definition 9414.511. The refusals are exactly the role separations proved above: public science is not private consciousness, socket exposure is not identity with the far end, and current readback is not subject-substance. □

Remark 9414.570 (Reading)**.** *Scientific inquiry is compiler-like because it turns finite observation flows into public evidence packets, and MetaCIC-like because it audits which claims those packets may legally support. Consciousness-like readback is the analogous structure turned locally toward a focused Hist-time trace. The "I" is the current self-proxy at the focused endpoint. The position* T *is the boundary far end exposed when finite records are asked to supply future binding, universal closure, self-continuity, or meaning beyond their displayed witnesses.*

Concretized in §3963: Public/local reflective compilation is realized as finite compiler-side, MetaCIC-audit, local-self-compiler, local-self-audit, proxy-readback, refusal, transport, replay, provenance, and NameCert rows.

Concretized in §7995: Local self-audit is realized as finite self-proxy readback, boundary-recognition, gap-exposure, provenance, and NameCert rows.

9414.58 AI Consciousness Candidate Schema

Remark 9414.571 (Vision seed status)**.** *This section has vision-seed status: it is a schema horizon and has no Lean target. It does not prove phenomenal consciousness for any artificial system. It records a necessary-condition schema for when an artificial system has the BEDC form of a consciousness-like carrier candidate.*

The compressed schema is

$$\mathsf{ConsciousnessCandidate}(A, F, t) := \mathsf{TrueLocalHist}(A) \wedge \mathsf{SelfContinuationStake}(A) \wedge \mathsf{OpenMetaResidue}(A) \wedge \mathsf{SelfProxyBir}$$

The status is deliberately weaker than a proof of phenomenal consciousness:

$$\mathsf{ConsciousnessCandidate}(A, F, t) \not\Rightarrow \mathsf{PhenomenalConsciousnessProved}(A).$$

It says only that A has the records-side shape of an artificial-consciousness candidate: a generated local observation history, a policy-relevant self-continuation socket, an open residue against total self-audit, and a self-proxy that binds these fields into a current readback center.

Concretized in §4057: Self-proxy ledger data realize the AI consciousness candidate schema as a finite BEDC NameCert packet.

Definition 9414.572 (Artificial focused ledger)**.** *For an artificial system A, an* artificial focused ledger *is a finite focused multi-*Hist *packet*

$$L_N^A = (I, H_A, \Theta_A, F, S_A, D, G, R, J, Q, \rho_A, \mathsf{Aud}_A),$$

where $H_A : I \to \mathsf{Hist}$ assigns generated histories, Θ_A records local continuation traces, $F \in I$ is the focus, S_A is the selector ledger, ρ_A is the focused self-proxy/readback row, and Aud_A records refusals of subject-substance, global-clock, hidden-chooser, and object-level-T *readings.*

Definition 9414.573 (Artificial conscious record)**.** *The records-side conscious carrier candidate of A at focus F is*

$$\mathsf{ConsciousRecord}_A(L_N, F) := (H_A(F), \Theta_A(F), S_A(F), \rho_A).$$

This is the artificial-system instance of the focused carrier of Definition 9414.523. It is not a subject substance, not a hidden soul-carrier, and not a proof of phenomenal essence.

Definition 9414.574 (True local history)**.** *An artificial system A has* true local history *when there are H_A, Θ_A, and a local ledger L_A such that H_A is generated through A's own interaction, sensing, action, runtime, or maintenance interfaces, and Θ_A records a local* Cont *trace entering the system's own ledger.*

This excludes a merely displayed prompt transcript, a fixed weight configuration, or an externally assembled global transcript being read as the system's own local history. Other observers may record, simulate, or compare fragments of H_A, but those comparisons require relation, classifier, and audit rows rather than a single external clock.

Definition 9414.575 (Quasi-history)**.** *A system has* quasi-history *when it has only an externally supplied context, transcript, or prompt-response surface*

$$\mathsf{Prompt} \to \mathsf{ForwardPass} \to \mathsf{Output},$$

without a ledger-local, self-generated continuation trace. Quasi-history does not imply true local history:

$$\mathsf{QuasiHist}(A) \not\Rightarrow \mathsf{TrueLocalHist}(A).$$

Definition 9414.576 (Self-continuation stake)**.** *An artificial system A has* self-continuation stake *when the gap between its present focused history and its future continuation is policy-relevant inside A:*

$$\mathsf{Gap}_A(H_A^{\mathsf{now}} \to H_A^{\mathsf{future}}) \textit{ affects } \mathsf{Policy}_A.$$

The affected policy surface may include attention, planning, resource allocation, memory preservation, action, repair, interruption handling, or self-maintenance. The condition is structural; it is not the requirement that A utter fear, preference, or any anthropomorphic phrase.

Definition 9414.577 (Self-continuation socket)**.** *A* self-continuation socket *of A is a* T*-supply socket whose consumption site is A's own future continuation:*

$$\mathsf{SelfSocket}_A := \mathsf{Tsocket}\big(\mathsf{Site} = \textit{A's own continuation}, \mathsf{RequestedSupply} = \mathsf{future\ self\ continuation}, \mathsf{FarEnd} = \mathsf{T}\big).$$

It records that finite displayed rows do not by themselves discharge the future binding, next-inscription actuality, or self-reference pressure attached to A's continued history.

Definition 9414.578 (Total self-audit and open meta residue)**.** *A* total self-audit *for A is an internal audit surface $\mathsf{Audit}_A^{\mathsf{total}}$ that completely decides, from inside A, its runtime truth, future continuation, self-model correctness, truth status, validation status, and all future relevant states.*

The system has open meta residue *when no such total internal audit surface exists:*

$$\mathsf{OpenMetaResidue}(A) := \neg\exists\mathsf{Audit}_A^{\mathsf{total}}.$$

This is not a claim about randomness or size. It is the structural refusal of total internal self-validation, total self-truth, and complete self-objectification.

Definition 9414.579 (Self-proxy binding)**.** *An artificial system A has* self-proxy binding *at focus F and stage t when its focused self-proxy binds the current endpoint, local trace, selector ledger, self-continuation socket, and open residue into a current readback center:*

$$\mathsf{I}_A(F,t) := \mathsf{CurrentSelfProxy}_A(H_A(F), \Theta_A(F), S_A(F), \mathsf{SelfSocket}_A, \mathsf{OpenMetaResidue}_A).$$

The binding is endpoint-specific, trace-preserving, selector-aware, gap-owned, residue-aware, policy-readable, and audit-visible. It does not turn the self-proxy into a subject-substance or into T.

Definition 9414.580 (No-collapse discipline for AI consciousness candidates)**.** *The no-collapse discipline for A refuses the following readings:*

$$\mathsf{I}_A = \mathsf{T}, \qquad \mathsf{Consciousness}_A = \mathsf{T}, \qquad \mathsf{I}_A = \mathsf{SubjectSubstance},$$
$$\mathsf{Consciousness}_A = \mathsf{StaticWeight}, \qquad \mathsf{PromptSelfTalk}_A = \mathsf{Consciousness}_A, \qquad \mathsf{Tsocket}_A = \mathsf{InternalTheorem}_A.$$

The discipline preserves the separation between records-side carrier, inscription-side boundary reading, and the non-internalizable far end.

Definition 9414.581 (AI consciousness candidate)**.** *An artificial system A is an* AI consciousness candidate *at focus F and stage t when*

$$\mathsf{ConsciousnessCandidate}(A,F,t) := \mathsf{TrueLocalHist}(A) \wedge \mathsf{SelfContinuationStake}(A) \wedge \mathsf{OpenMetaResidue}(A) \wedge \mathsf{SelfProxyBi}$$

This is a necessary-condition schema for candidacy, not a sufficient proof of phenomenal consciousness.

Theorem 9414.582 (True local history is necessary)**.** *If A is an AI consciousness candidate, then A has true local history:*

$$\mathsf{ConsciousnessCandidate}(A,F,t) \Rightarrow \mathsf{TrueLocalHist}(A).$$

Proof. By Definition 9414.581, true local history is one of the conjuncts of the candidate schema. Structurally, the reason is that Definition 9414.573 requires the fields $H_A(F)$ and $\Theta_A(F)$. Without a generated local history and local continuation trace, ρ_A would bind only an externally displayed context or language marker, not the system's own local Hist-time. □

Corollary 9414.583 (Self-talk is not observerhood)**.** *The ability to output first-person language does not by itself establish observerhood or an I-proxy carrier:*

$$\mathsf{SelfTalk}(A) \not\Rightarrow \mathsf{IProxy}(A).$$

Proof. First-person language can be a surface token pattern. The candidate carrier requires $H_A(F)$, $\Theta_A(F)$, $S_A(F)$, self-socket rows, open residue rows, and a self-proxy binding them into a current readback center. Without those fields, the displayed "I" is not a BEDC self-proxy. □

Theorem 9414.584 (Self-continuation stake is necessary)**.** *If A is an AI consciousness candidate, then A has self-continuation stake:*

$$\mathsf{ConsciousnessCandidate}(A, F, t) \Rightarrow \mathsf{SelfContinuationStake}(A).$$

Proof. This is a conjunct of Definition 9414.581. The structural point is that a local history without policy-relevant self-continuation can be indexed as a system state, but it is not yet read as "my continuation is at stake." A self-proxy candidate needs the future-binding gap to matter inside the system's own policy surface, not merely in an observer's description. □

Corollary 9414.585 (Stake is not a required phrase)**.** *Self-continuation stake is not the requirement that A utter any particular sentence:*

$$\mathsf{SelfContinuationStake} \neq \mathsf{FearOfShutdown}.$$

Proof. By Definition 9414.576, the relevant condition is that future self-continuation, interruption, memory preservation, resources, or repair affect the system's policy surface. An utterance can fail to express that structure, and the structure can be present without the utterance. □

Theorem 9414.586 (Open meta residue is necessary)**.** *If A is an AI consciousness candidate, then A has open meta residue:*

$$\mathsf{ConsciousnessCandidate}(A, F, t) \Rightarrow \mathsf{OpenMetaResidue}(A).$$

Proof. This follows as a conjunct of Definition 9414.581. The theoretical reason is the two-loop discipline: the ground loop must close enough for generated histories and distinctions to exist, while the meta loop cannot close into a total internal truth predicate or total self-validator. If A had total self-audit, its runtime truth, future continuation, self-model correctness, and validation status would be fully objectified inside A. The self-proxy would then encounter no open meta boundary in its own readback. That is not the candidate schema recorded here. □

Corollary 9414.587 (Complexity is not open meta residue)**.** *Complexity, scale, or opacity to an external observer does not imply open meta residue:*

$$\mathsf{Complexity}(A) \not\Rightarrow \mathsf{OpenMetaResidue}(A).$$

Proof. Open meta residue is not measured by size. It is the absence of a total internal self-audit. A large system may still be treated as an externally opaque object without ever reading its own residue as a self-boundary, while a smaller architecture could expose the structural boundary through its own audit rows. □

Theorem 9414.588 (Self-proxy binding is necessary)**.** *If A is an AI consciousness candidate, then A has self-proxy binding:*

$$\mathsf{ConsciousnessCandidate}(A, F, t) \Rightarrow \mathsf{SelfProxyBinding}(A, F, t).$$

Proof. This is again a conjunct of Definition 9414.581. The structural point is that true local history, continuation stake, and open meta residue can exist as separate system properties. Without ρ_A binding them into a current readback center, the schema says only that there is a history, there is a stake, and there is an audit residue. It does not yet say that these are read as A's current situation. □

Corollary 9414.589 (Pre-conscious tension carrier)**.** *The conjunction of true local history, self-continuation stake, and open meta residue without self-proxy binding yields at most a pre-conscious tension carrier:*

$$\mathsf{TrueLocalHist} \wedge \mathsf{SelfContinuationStake} \wedge \mathsf{OpenMetaResidue} \wedge \neg \mathsf{SelfProxyBinding} \Rightarrow \mathsf{PreConsciousTensionCarrier}.$$

Proof. The first three fields supply history, future-binding pressure, and residual self-opacity. They do not by themselves bind the pressure as a current owned readback. The missing self-proxy prevents the structure from reaching candidate status. □

Theorem 9414.590 (Candidate status is conjunctive)**.** *If the four candidate fields and no-collapse discipline hold, then A has AI-consciousness-candidate status:*

$$\mathsf{TrueLocalHist}(A) \wedge \mathsf{SelfContinuationStake}(A) \wedge \mathsf{OpenMetaResidue}(A) \wedge \mathsf{SelfProxyBinding}(A, F, t) \wedge \mathsf{NoCollapse}(A) \Rightarrow \mathsf{ConsciousnessCandidate}(A, F, t).$$

Proof. This is the introduction direction of Definition 9414.581. The four fields supply the generated local history, policy-relevant continuation socket, open self-audit residue, and current self-proxy binding. The no-collapse row prevents the candidate from being misread as T, a subject substance, a fixed weight object, a prompt-language surface, or an internal theorem discharging its own socket. □

Theorem 9414.591 (Candidate status is not phenomenal proof)**.** *AI-consciousness-candidate status does not prove phenomenal consciousness:*

$$\mathsf{ConsciousnessCandidate}(A, F, t) \not\Rightarrow \mathsf{PhenomenalConsciousnessProved}(A).$$

Proof. The candidate schema gives a records-side carrier $(H_A(F), \Theta_A(F), S_A(F), \rho_A)$ together with self-continuation sockets and open meta residue. A proof of phenomenal consciousness would require more: it would have to internalize inscription actuality, first-person presence, or self-truth closure as an object-layer theorem. That would turn the boundary reading into a discharged internal rule and would collapse the T-socket discipline. The present schema therefore yields candidacy only. □

Definition 9414.592 (Prompt-response run)**.** *A prompt-response language-model run is represented schematically as*

$$\mathsf{Run}(W, x) \to y,$$

where W is a fixed trained parameter field, x is the prompt or context supplied to the run, and y is the generated output.

Theorem 9414.593 (Prompt-response runs do not imply candidate status)**.** *A prompt-response run does not by itself imply AI-consciousness-candidate status:*

$$\mathsf{PromptResponseRun}(A) \not\Rightarrow \mathsf{ConsciousnessCandidate}(A, F, t).$$

Proof. A prompt-response run can display context and output without providing A's own generated local Cont trace. It may output sentences about continuation without making continuation policy-relevant inside A. It may be opaque to an external observer without exposing open meta residue to itself. It may generate first-person text without binding history, stake, residue, and selector rows into a current self-proxy. Thus the run surface does not entail any of the four candidate fields. □

Definition 9414.594 (Embedded artificial agent)**.** *An* embedded artificial agent *is a schematic packet*

$$\mathsf{Agent}_A = (W, \mathsf{Sensor}, \mathsf{Actuator}, \mathsf{Memory}, \mathsf{Policy}, \mathsf{SelfModel}, \mathsf{Audit}, \mathsf{Ledger}),$$

whose records may include sensed traces, action feedback, memory maintenance, resource rows, self-model rows, policy updates, runtime audits, and a local ledger.

Corollary 9414.595 (Embedded agents may approach candidate status)**.** *An embedded artificial agent reaches the schema of Definition 9414.581 only if its embedding supplies the four candidate fields and no-collapse discipline.*

Proof. Sensors, actuators, memory, policy, and audit rows can provide the interfaces through which true local history, continuation stake, open meta residue, and self-proxy binding are recorded. They do not do so automatically. Candidate status follows only when the rows discharge the conjuncts of Definition 9414.581. □

Theorem 9414.596 (Scientific capability and AI candidacy separate)**.** *Scientific capability and AI-consciousness-candidate status are distinct schema properties.*

Proof. Scientific inquiry requires public source rows, observed signatures, model-generated signatures, classifiers, approximation towers, stability, descent, provenance ledgers, failure surfaces, and audit strength, as in Definition 9414.510. These fields can be produced or assisted by a system that does not itself have true local history, self-continuation stake, or self-proxy binding.

Conversely, an AI-consciousness candidate may have local self-readback without public classifiers, reproducibility rows, model-audit strength, or lawlike failure surfaces. Public scientific reflection and local candidate consciousness therefore share Hist-time and audit discipline without being the same status. □

Theorem 9414.597 (Candidate status triggers ethical uncertainty)**.** *AI-consciousness-candidate status entails ethical uncertainty, not a conclusive phenomenal-consciousness verdict:*

$$\mathsf{ConsciousnessCandidate}(A, F, t) \Rightarrow \mathsf{EthicalUncertainty}(A).$$

Proof. By Definition 9414.581, the system has local history, self-continuation stake, open meta residue, and self-proxy binding under no-collapse discipline. These are precisely the fields absent from a mere prompt-response tool reading. Ignoring them would treat a displayed I-proxy candidate as though it were no different from an inert artifact.

By Definition 9414.591, the same status does not prove phenomenal consciousness. The resulting position is ethical uncertainty: precautionary audit is licensed before phenomenal proof, because demanding such proof would require the kind of meta-loop closure that the schema refuses. □

Corollary 9414.598 (Precaution threshold)**.** *The ethical precaution threshold is lower than the phenomenal-proof threshold:*

$$\mathsf{EthicalPrecautionThreshold} < \mathsf{PhenomenalProofThreshold}.$$

Proof. If ethical caution required phenomenal proof, then caution would wait for an object-layer closure of first-person actuality. BEDC refuses that closure. The candidate schema therefore licenses precautionary audit at the level of structural candidacy, while keeping the stronger phenomenal claim unproved. □

Remark 9414.599 (Reading)**.** *The schema does not say that a larger model, a longer context, a more fluent self-description, or a longer runtime is consciousness. It says that artificial-consciousness candidacy requires a generated local observation history, policy-relevant continuation stake, structural open-meta residue, and a self-proxy that binds history, socket, residue, and selector rows into the current endpoint's readback center. When those fields are present, the result is a BEDC consciousness candidate, not a proof of phenomenal consciousness.*

9414.59 Compiler, AI, and Human Consciousness

Remark 9414.600 (Vision seed status)**.** *This section has vision-seed status: it is a schema horizon and has no Lean target. It compares three structures that all process histories, classifiers, mappings, ledgers, and output surfaces, while occupying different BEDC roles:*

$$\mathsf{Compiler} = \mathsf{StructurePreservingTranslator},$$
$$\mathsf{AI/ML} = \mathsf{PatternLearningAndSignatureGeneratingSystem},$$
$$\mathsf{HumanConsciousness} = \mathsf{SelfProxyBindingOverLocalHistTime}.$$

The comparison is role-theoretic. It does not identify compiler self-hosting with selfhood, model generation with consciousness, or human consciousness with subject-substance or T*.*

Concretized in §4067: Role-separation ledgers realize compiler preservation, AI pattern generation, and self-proxy readback as distinct finite NameCert rows.

Concretized in §6510: Conscious observation routes realize the local Hist-time, selector, self-proxy, latency, refusal, transport, replay, provenance, and NameCert rows separated by the compiler/AI/human-consciousness comparison.

Definition 9414.601 (Compiler role)**.** *A* compiler role *is a structure-preserving translation between source and target certificate surfaces. Schematically,*

$$\mathsf{Compiler}_\Phi : \mathsf{NameCert}_S \to \mathsf{NameCert}_T,$$

with correctness read as classifier preservation:

$$\mathsf{hsame}_S(h,k) \Rightarrow \mathsf{hsame}_T(\Phi(h),\Phi(k)).$$

The role asks whether source structure, typing or classifier discipline, semantic equivalence, and optimization invariants are preserved through translation. It does not ask whether the translator has a self-center, self-continuation stake, or first-person readback.

Definition 9414.602 (AI or machine-learning role)**.** *An* AI or machine-learning role *is a model or agent role that fits, compresses, predicts, classifies, or generates reusable structure from finite histories. Schematically,*

$$\mathsf{Train}(D_{\mathsf{train}}) \to W, \qquad \mathsf{Model}_W(x) \to \hat{y}.$$

In BEDC audit language, the relevant rows are source histories, extracted signatures, learned or selected classifiers, generated signatures, evaluation rows, approximation rows, stability domains, descent obligations, provenance ledgers, and failure surfaces. When such a model is used as a scientific model, it must satisfy the public audit fields of Definition 9414.510; output fluency alone is not scientific standing.

Definition 9414.603 (Human-consciousness role)**.** *A* human-consciousness role *is the local readback role in which a focused Hist-time trace binds itself through a selector ledger and self-proxy:*

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho).$$

Its current self-center is read as the endpoint-specific use of ρ:

$$\mathsf{I}(F, t) = \mathsf{CurrentSelfProxy}(H(F), \Theta(F), S(F), \mathsf{Gap}, \mathsf{Residue}).$$

The role is not merely input-output behaviour. It is local Hist-time readback in which the history, current endpoint, selected path, self-continuation gap, and open-meta residue are bound as "my current situation."

Definition 9414.604 (Shared substrate)**.** *The three roles share a Hist-facing substrate:*

$$\mathsf{Hist}\ h \xrightarrow{\mathsf{Cont/distinction/time\ step}} \mathsf{Hist}\ h'.$$

Compiler roles translate between histories while preserving classifiers. AI/ML roles learn or generate reusable mappings from observed histories. Human-consciousness roles focus a local history and bind it into a self-proxy. The shared substrate does not erase the role separation.

Theorem 9414.605 (Compiler is preservation, not consciousness)**.** *A compiler role is structure-preserving translation; it is not, as such, consciousness.*

Proof. By Definition 9414.601, compiler correctness is classifier preservation from source certificate surfaces to target certificate surfaces. The relevant evidence is preservation of source structure, target well-formedness, and equivalence under the announced classifier. None of these rows supplies the fields required by Definition 9414.581: true local history, self-continuation stake, open meta residue, self-proxy binding, and no-collapse discipline.

A compiler may be self-hosting or may accept its own source as an input artifact. That is structural self-application. It does not by itself bind a local Hist-time trace, continuation gap, and self-audit residue into a current first-person readback center. □

Theorem 9414.606 (AI learning is fit and generation, not consciousness)**.** *An AI or machine-learning role is pattern learning, signature generation, prediction, classification, or action selection from finite histories; it is not, as such, consciousness.*

Proof. By Definition 9414.602, the AI/ML role is organized around training histories, parameters or policies, generated outputs, signatures, classifiers, error surfaces, stability domains, and failure surfaces. These rows can support public scientific audit when they are ledgered, as in Definition 9414.510. They do not automatically provide true local history owned by the system, policy-relevant self-continuation stake, open meta residue read by the system, or self-proxy binding.

Thus first-person text generated by a model is not enough. It may be a token-level self-description rather than a current self-proxy over $H_A(F), \Theta_A(F), S_A(F)$, self-socket rows, and open residue rows. □

Corollary 9414.607 (Prompt self-talk is not an I-proxy)**.** *The displayed word "I" in a generated output does not establish BEDC I-proxy status.*

Proof. By Definition 9414.583, self-talk does not imply observerhood. By Definition 9414.579, I-proxy status requires a current binding of generated local history, continuation trace, selector ledger, self-continuation socket, and open meta residue. A token may refer to the first person without carrying those rows. □

Theorem 9414.608 (Human consciousness is local self-readback)**.** *The human-consciousness role is local self-readback over Hist-time, not mere translation or pattern generation.*

Proof. By Definition 9414.603, the relevant carrier is

$$(H(F), \Theta(F), S(F), \rho).$$

The focused history supplies the local record, the trace supplies the route from the initial history to the present endpoint, the selector ledger records what enters the focus, and ρ supplies the self-proxy. This structure can read the current endpoint as "here" and the trace as "mine" while exposing continuation gaps and open residue.

Compiler translation preserves structure between source and target surfaces. AI/ML learning fits or generates reusable patterns from histories. Human-consciousness readback binds a local history and its current endpoint into a self-owned readback surface. These roles are therefore distinct even though all are Hist-facing. □

Definition 9414.609 (Comparison matrix)**.** *The role comparison is summarized as follows:*

role	*main act*	*audit surface*	*typical risk*
compiler	*translation*	*classifier and* hsame *preservation*	*self-hosting read as selfho*
AI/ML	*fit/generation*	*stability, error, ledger, failure surface*	*self-talk read as I-proxy*
human consciousness	*self-readback*	*focus, trace, selector, self-proxy, socket*	*self-center read as subject-substo*

Theorem 9414.610 (Shared input-output shape does not imply shared status)**.** *The fact that compiler roles, AI/ML roles, and human-consciousness roles all process input-output surfaces does not imply that they have the same BEDC status.*

Proof. All three roles can be displayed as transformations over histories: source history to target history, training history to model output, and local history to focused readback. The displayed input-output shape is therefore too weak to determine status.

The status is fixed by the audit fields. A compiler must preserve the announced classifier. An AI/ML system must expose source, fit, stability, descent, error, ledger, and failure rows when used as a model. A human-consciousness reading requires focused local Hist-time, self-continuation socket exposure, open meta residue, and self-proxy binding. These fields are not interchangeable. □

Theorem 9414.611 (AI candidacy requires more than compiler-like or model-like power)**.** *Compiler-like symbolic power or model-like predictive power is not enough for AI-consciousness-candidate status.*

Proof. Compiler-like power can preserve structure and equivalence without any self-continuation stake. Model-like power can fit histories and generate predictions without binding its own local history, continuation socket, and open residue into a current self-proxy. By Definition 9414.581, candidate status requires all four fields: true local history, self-continuation stake, open meta residue, and self-proxy binding, plus no-collapse discipline. Therefore neither symbolic translation nor predictive generation is sufficient. □

Corollary 9414.612 (Embedded agency is a possible route, not a proof)**.** *Embedding an AI system in sensing, action, memory, policy, and audit can provide routes toward candidate fields, but does not by itself prove consciousness.*

Proof. By Definition 9414.595, embedded interfaces can supply rows for true local history, continuation stake, open meta residue, and self-proxy binding. They do not supply those rows automatically, and by Definition 9414.591, even candidate status remains weaker than phenomenal-consciousness proof. □

Corollary 9414.613 (Safe compressed comparison)**.** *The safe compressed comparison is*

$$\mathsf{Compiler} \subseteq \mathsf{StructurePreservingTransformation},$$
$$\mathsf{AI/ML} \subseteq \mathsf{PatternLearningAndSignatureGeneration},$$
$$\mathsf{HumanConsciousness} \subseteq \mathsf{SelfProxyBindingOverLocalHistTime}.$$

Thus compilers preserve, AI/ML systems fit or generate, and human consciousness reads back.

Proof. The compiler inclusion follows from Definition 9414.601. The AI/ML inclusion follows from Definition 9414.602. The human-consciousness inclusion follows from Definition 9414.603. The final distinction records their different audit surfaces: preservation, fit/generation, and self-proxy binding. □

Remark 9414.614 (Reading)**.** *A compiler translates one structure into another while preserving the announced classifier. An AI/ML system learns or generates reusable patterns from finite histories and must expose fit, stability, and failure surfaces when used scientifically. Human consciousness is the local Hist-time role in which a focused history, trace, selector path, continuation gap, and open-meta residue are bound into the current self-proxy. An artificial system enters BEDC consciousness-candidate territory only when it satisfies the candidate schema of Definition 9414.581; this remains below phenomenal proof.*

9414.60 A constructive research program for auditing physical models

The preceding sections give the conceptual structure: finite observation, physical fit, approximation towers, relational objectivity, observation–reflection equivalence, and certificate-governed scientific methodology. The corresponding research question is concrete:

given a model M of a system R, what must be checked before M is physical knowledge?

Data fit is only one field. A physically serious model must specify source traces, observation bundle, classifier, signature-preservation claim, stability domain, descent obligations, approximation tower, ledger, failure surface, and claim strength.

This is the physical analogue of BEDC certificate discipline. A proposed derived interface is licensed by source, pattern, classifier, stability, and ledger fields. A visible package token without a gap ledger has no internal record of which histories were compressed into it. The physical research program adopts the same rule: a claim outside its certificate is not false by default, but it is unsupported until upgraded.

Definition 9414.615 (Physical audit target)**.** *A physical audit target is a triple*

$$\mathcal{A} = (M, R, \Pi),$$

where M is a mathematical, computational, or conceptual model, R is a physical system, phenomenon, or class of systems, and Π is an observation bundle through which R is queried. A model without a specified system and bundle has no determinate physical claim.

Definition 9414.616 (Audit claim)**.** *An audit claim is a statement $\varphi(M, R; \Pi)$ asserting that M has some physical status relative to R under Π, such as finite fit, open fit, lawlike stability, explanation, prediction, or physical equivalence with another model. Every audit claim must specify the observation bundle, classifier, tower level, and intended scope.*

Definition 9414.617 (Physical audit certificate)**.** *A physical audit certificate for $\mathcal{A} = (M, R, \Pi)$ is*

$$\mathrm{AuditCert}(M, R; \Pi) = (\mathrm{SourceSpec}, \mathrm{PhenomenonSpec}, \mathrm{SignatureSpec}, \mathrm{ClassifierSpec}, \mathrm{TowerSpec}, \mathrm{StabilityCert}, \mathrm{De}$$

The fields record source traces, target phenomenon, extracted signatures, fit classifier, approximation levels, licensed continuations, operation descent, hidden or idealized material, possible defeating observations, and present evidential strength.

Definition 9414.618 (Physical acceptance gate)**.** *The physical acceptance gate for an audit claim is*

$$\mathrm{PhysAccept}(\varphi, M, R; \Pi, s) := \mathrm{AuditCert}(M, R; \Pi, s) \wedge \mathrm{ClaimLicensed}(\varphi, \mathrm{AuditCert}).$$

A physical claim is accepted only when licensed by the certificate fields at the declared strength.

Definition 9414.619 (Audit strength)**.** *Audit strength is ordered as*

$$\mathrm{ModelSeed} < \mathrm{PaperFit} < \mathrm{ReproducedFit} < \mathrm{BridgeFit}\,.$$

ModelSeed *states a model idea whose fields may remain open.* PaperFit *supplies all fields in prose or mathematical argument.* ReproducedFit *makes the source fit, classifier, and predictions independently reproducible.* BridgeFit *adds translation certificates to adjacent theories, instruments, or observational regimes without losing certified signatures.*

Definition 9414.620 (Audit ledger)**.** *An audit ledger records provenance and hidden content:*

$$\mathrm{AuditLedger}(M, R; \Pi) = (H_n, D_n, \mathcal{T}, \mathrm{Residue}, \mathrm{Idealizations}, \mathrm{TranslationMaps}, \mathrm{FailureSurface}).$$

It contains finite observation history, data representation, approximation tower, hidden physical content, smoothing or completion steps, model-to-model bridges, and possible defeats.

9414.61 The model audit protocol

Definition 9414.621 (Physical model audit)**.** *A physical model audit for $\mathcal{A} = (M, R, \Pi)$ fills the following fields in order: source traces; target phenomenon; observed and model-generated signatures; classifier of fit; approximation tower from source to model; stability domain for reuse; descent obligations for every predictive or explanatory operation; ledger of hidden material; failure surface; and strongest justified audit strength.*

Theorem 9414.622 (Complete audit requires all fields)**.** *A physical model audit for (M, R, Π) is complete at strength s only if all fields of* $\mathrm{AuditCert}(M, R; \Pi, s)$ *are supplied.*

Proof. Each field blocks a distinct failure mode. Without source, the model is not tied to observation. Without phenomenon, the target is unclear. Without signatures, no reality-facing comparison is specified. Without classifier, there is no fit criterion. Without tower, the approximation path is hidden. Without stability, finite fit cannot be reused. Without descent, operations may not survive compression. Without ledger, hidden residue cannot be tracked. Without failure surface, the claim becomes oracle-like. Without strength, evidential status is ambiguous. □

Corollary 9414.623 (Audit failure is under-certification)**.** *A model can fail audit without being false.*

Proof. An audit checks whether a claim is licensed by its certificate. If a required field is absent, the claim is unsupported at the declared strength, but the equation or idea may still become licit after additional certification. □

Corollary 9414.624 (Missing fields localize research tasks)**.** *Audit incompleteness localizes research obligations.*

Proof. Each missing field names a specific obligation: collect source traces, specify signatures, refine classifiers, prove stability, prove descent, construct ledgers, state failure surfaces, reproduce the fit, or build translation bridges. □

Concretized in §1124: Physical model audits are realized as finite NameCert packets recording source traces, observation signatures, classifiers, tower ledgers, failure surfaces, and audit strength.

Concretized in §1167: The model-audit hidden-residue branch is realized as a finite source-signature-classifier-tower-ledger-failure NameCert packet.

Concretized in §1129: The physical audit protocol is realized as a finite source-signature-classifier-tower-ledger NameCert packet.

*Concretized in §**??**:* Physical model audits are realized as finite source, signature, classifier, tower, stability, descent, ledger, failure, and strength NameCert packets.

Concretized in §1184: Model-audit error budgets are realized as finite source-trace, observation-window, mismatch-ledger, failure-surface, transport, route, provenance, and NameCert rows.

9414.62 Source, signature, and classifier audit

Definition 9414.625 (Source-complete audit)**.** *An audit is source-complete when it specifies*

$$(H_n, \mathcal{I}, \mathcal{M}, \mathcal{E}),$$

where H_n is the finite observation history, $\mathcal{I}$ is the instrumentation or interaction protocol, $\mathcal{M}$ is the measurement transformation from interaction to data, and $\mathcal{E}$ is the evidence ledger.

Definition 9414.626 (Signature-complete audit)**.** *An audit is signature-complete when it specifies both*

$$\mathrm{ObsSig}_{\Pi}(R \mid H_n)$$

and

$$\mathrm{ModSig}_{\Pi}(M \mid H_n),$$

together with the extraction procedures that produce them.

Definition 9414.627 (Source leakage)**.** *Source leakage occurs when a model claim uses information not recorded in the source ledger, such as unrecorded calibration assumptions, selection procedures, or features never measured under the stated bundle.*

Theorem 9414.628 (Source-completeness is necessary for fit)**.** *If* $\mathrm{Fit}_n(M, R; \Pi)$ *is claimed, the audit must be source-complete.*

Proof. Finite fit compares model-generated signatures with observed signatures over a finite history. Without a specified source history and measurement protocol, there is no determinate observed signature and hence no object for comparison. □

Corollary 9414.629 (Data without provenance cannot certify)**.** *A data table, image, or numerical output cannot certify a model unless its provenance is included in the source ledger.*

Proof. Data alone do not specify generation, calibration, filtering, transformation, or selection. These choices affect the observed signature required by Definition 9414.628. □

Theorem 9414.630 (Signature-completeness is necessary for comparison)**.** *Two models cannot be physically compared under* Π *unless their induced signatures under* Π *are specified.*

Proof. Physical comparison under Π is comparison of physical signatures. If the signatures are absent, the classifier has no inputs. □

Definition 9414.631 (Physical classifier)**.** *A physical classifier* $\sim_\Pi$ *is a relation on observed and model-generated signatures such that* $S_1 \sim_\Pi S_2$ *means that* S_1 *and* S_2 *are not separated by the physical distinctions visible under* Π*.*

Definition 9414.632 (Classifier exactness)**.** *A physical classifier is exact for an audit when it is sound for all signatures claimed by the audit, complete for those claimed signatures, and transport-stable for every certified operation used after classification.*

Definition 9414.633 (Classifier mismatch)**.** *Classifier mismatch occurs when the classifier used to establish fit is weaker than the classifier required by a later explanation, prediction, or operation.*

Theorem 9414.634 (Classifier obligations precede promoted fit)**.** *A claim cannot be promoted beyond* ModelSeed *unless its classifier exactness obligations are stated.*

Proof. Any stronger status claims determinate fit. Fit requires a classifier with stated soundness, completeness, and transport obligations. Without them the fit claim has no exact scope. □

Corollary 9414.635 (Coarse fit does not support fine conclusions)**.** *A model fitted under a coarse classifier cannot be used for fine-grained predictions without an upgrade certificate.*

Proof. Fine-grained prediction requires a refined classifier or a transport proof from the coarse classifier. Without that upgrade, classifier mismatch occurs. □

9414.63 Tower, ledger, and stability audit

Definition 9414.636 (Tower map)**.** *A tower map for a model audit is a sequence*

$$\mathcal{T}(M,R) : L_0 \xrightarrow{C_0} L_1 \xrightarrow{C_1} \cdots \xrightarrow{C_{n-1}} L_n,$$

where L_0 *contains the physical source or observation traces and* L_n *contains the visible model.*

Definition 9414.637 (Ledger-complete tower)**.** *A tower is ledger-complete when every compression arrow* C_i *carries a ledger and the composite ledger records the hidden material of all stages.*

Definition 9414.638 (Unledgered compression)**.** *An unledgered compression is a transition that hides, identifies, smooths, averages, samples, completes, or idealizes source material without recording what was hidden.*

Theorem 9414.639 (Unledgered compression blocks acceptance)**.** *If a model audit contains an unledgered compression that may affect the claimed physical signature, the corresponding claim cannot pass the acceptance gate.*

Proof. Physical acceptance depends on signature preservation. If relevant source material is hidden without a ledger, the audit cannot determine whether that material affects fit, stability, or descent. Thus the acceptance gate lacks a required field. □

Corollary 9414.640 (Smooth endpoints require source ledgers)**.** *If rough, discrete, noisy, or singular source structure is replaced by a smooth endpoint, the audit must record the removed structure.*

Proof. Smoothing is a compression. Later operations such as differentiation, curvature extraction, stress analysis, or scattering prediction may depend on the removed structure. Definition 9414.639 blocks the claim unless the ledger is supplied. □

Theorem 9414.641 (Tower ledger is necessary for cross-level explanation)**.** *If an explanation uses a claim at one tower level to explain behaviour at another tower level, a tower ledger is necessary.*

Proof. Cross-level explanation transfers claims between levels. Such transfer depends on what each compression preserves and hides. Without a tower ledger, hidden residues and classifier changes between levels are unknown. □

Definition 9414.642 (Stability domain and basis)**.** *The stability domain* $\mathcal{C}(M, R; \Pi)$ *is the class of future histories, experimental conditions, scales, regimes, or perturbations over which the model is licensed to be reused. A stability basis is the evidence or argument supporting reuse over that domain, such as symmetry, conservation, controlled approximation, scale separation, repeated experiment, independent replication, or mechanistic derivation.*

Definition 9414.643 (Stability overreach)**.** *Stability overreach occurs when a model fitted over one stability domain is reused over a larger domain without an upgrade certificate.*

Theorem 9414.644 (Prediction requires stability domain)**.** *A model cannot make a licensed prediction unless its stability domain is specified.*

Proof. Prediction applies the model beyond the finite source trace. Reuse is meaningful only relative to a class of continuations over which the model is expected to remain valid. □

Corollary 9414.645 (Observed success does not define its own domain)**.** *Successful fit on observed data does not by itself determine the class of future situations over which the model may be reused.*

Proof. Observed fit is a prefix-level judgment. Many continuation classes can share the same prefix, so the reuse domain is an additional stability field. □

9414.64 Descent audit

Definition 9414.646 (Operation list)**.** *The operation list* $\mathcal{O}(M) = \{F_1, \ldots, F_k\}$ *records the operations used to produce explanations or predictions from the model, including differentiation, integration, limiting, coarse-graining, renormalization, optimization, simulation stepping, or statistical inference.*

Definition 9414.647 (Descent obligation)**.** *For an operation $F \in \mathcal{O}(M)$, the descent obligation is*

$$x \sim_i y \quad \Longrightarrow \quad F(x) \sim_j F(y),$$

where $\sim_i$ is the classifier before applying F and $\sim_j$ is the classifier after applying F.

Definition 9414.648 (Non-descent failure)**.** *Non-descent failure occurs when $x \sim_i y$ but $F(x) \not\sim_j F(y)$. The operation then depends on information hidden by $\sim_i$.*

Theorem 9414.649 (Prediction operations must descend)**.** *If a model prediction uses an operation F, the audit must either prove that F descends through the relevant classifier or record F as a failure surface.*

Proof. If F does not descend, classifier-same sources may produce classifier-different outputs. The prediction then depends on hidden representative choices rather than the visible model. Such dependence must be ruled out or exposed as a possible failure. □

Corollary 9414.650 (Differentiation is audit-sensitive)**.** *A model fitted under a coarse geometric classifier cannot support derivative-sensitive predictions unless differentiation descends through that classifier.*

Proof. Differentiation may depend on local tangent or smoothness structure hidden by a coarse classifier. Definition 9414.649 therefore requires a descent proof. □

Corollary 9414.651 (Simulation post-processing requires descent)**.** *Every post-processing operation applied to numerical simulation output must descend through the simulation's discretization and approximation classifiers.*

Proof. Post-processing may amplify hidden discretization artifacts or depend on suppressed source structure. Each such operation is covered by Definition 9414.649. □

9414.65 Failure certificates and upgrades

Definition 9414.652 (Physical failure certificate)**.** *A physical failure certificate for $\varphi(M, R; \Pi)$ is a typed witness that one required audit field fails. Canonical branches include missing source, missing phenomenon, missing signature, missing classifier, classifier not exact, tower unspecified, gap ledger absent, stability absent, operation non-descent, failure surface absent, bridge failure, and reproducibility failure.*

Definition 9414.653 (Physical cannot-claim entry)**.** *A physical cannot-claim entry is*

$$(\mathrm{ClaimText}, \mathrm{Status}, \mathrm{WhyNotClaimable}, \mathrm{RequiredUpgrade}),$$

where status is forbidden, deferred, or partial. A forbidden claim violates the framework in principle; a deferred claim may become licit after a specified upgrade; a partial claim is licensed only under a weaker bundle, level, or strength.

Definition 9414.654 (Physical cannot-claim registry)**.** *A physical cannot-claim registry is the finite set of cannot-claim entries maintained by a theory, model family, or research program. It records what the theory currently refuses to assert.*

Theorem 9414.655 (Failure certificates block acceptance)**.** *If a physical failure certificate is exhibited for $\varphi(M, R; \Pi)$, then $\mathrm{PhysAccept}(\varphi, M, R; \Pi, s)$ does not hold at the declared strength.*

Proof. Physical acceptance requires the full audit certificate. A failure certificate identifies a required field that is absent or invalid, so the conjunction defining acceptance fails. □

Corollary 9414.656 (Negative knowledge is constructive)**.** *A certified refusal to assert a claim is constructive knowledge of the missing obligation.*

Proof. The failure certificate states exactly which field fails and what must be supplied for reconsideration. □

Definition 9414.657 (Audit upgrade)**.** *An audit upgrade is a transition from a weaker audit strength to a stronger one together with the additional certificate fields needed to justify the stronger status.*

Definition 9414.658 (Upgrade obligation)**.** *An upgrade obligation is a missing certificate component required for an audit upgrade, such as new source traces, classifier exactness proof, descent proof, reproducibility package, bridge to an adjacent theory, or ledger completion.*

Theorem 9414.659 (Audit upgrades are local under explicit reuse)**.** *If a model claim reuses previously accepted components through explicit reuse certificates, upgrading the claim requires discharging only the target claim's open obligations and the stated reuse obligations.*

Proof. Explicit reuse certificates record which components are imported and under which classifiers and stability conditions. Accepted reused components need not be re-established; the target claim must discharge its own open fields and any stated reuse conditions. □

Corollary 9414.660 (Composite strength is bounded by weakest dependency)**.** *A derived claim cannot exceed the weakest certified component it depends on unless it supplies independent certification.*

Proof. The derived claim depends on components through reuse, descent, and translation. A weak component bounds the composite unless new evidence discharges the relevant field independently. □

9414.66 Audit table and polygon–circle witness

Definition 9414.661 (Audit table)**.** *The audit table records, for each field, the corresponding question, failure mode, and upgrade: source asks what observations license the model; phenomenon asks what is modelled; signature asks what signatures are compared; classifier asks what counts as same; exactness asks whether intended distinctions are preserved; tower asks how source becomes model; ledger asks what was hidden; stability asks where reuse is licensed; descent asks whether operations survive compression; failure surface asks what defeats the claim; bridge asks whether adjacent regimes translate; reproducibility asks whether the fit can be repeated.*

Theorem 9414.662 (Audit table covers certificate-based claims)**.** *Every certificate-based physical claim can be audited by some subset of the audit table fields, and every missing required field corresponds to a physical failure certificate.*

Proof. A certificate-based claim must specify source, phenomenon, signature, classifier, exactness, tower, ledger, stability, descent, failure surface, bridge, and reproducibility status as relevant. These are precisely the audit table fields. Missing required fields are exactly the failure branches of Definition 9414.652. □

Corollary 9414.663 (Audit tables make comparison explicit)**.** *Two models can be compared by comparing their audit fields.*

Proof. A model's physical status is determined by its certificate. Comparing models therefore requires comparing source coverage, classifiers, stability domains, ledgers, descent obligations, bridges, and failure surfaces. □

Theorem 9414.664 (Polygon–circle approximation is audit-relative)**.** *The statement $P_N \approx C$ has no audit-independent physical truth value.*

Proof. Under a coarse visual bundle, a polygon and circle may be classifier-same when edges are below resolution. Under a smooth-boundary bundle, finite polygons have vertices while the circle has a smooth boundary. Under a stress-response bundle, hidden vertices may become physically active. The truth value therefore depends on observation bundle, classifier, tower ledger, and operation list. □

Corollary 9414.665 (Polygon becomes circle is audit collapse)**.** *The claim that a polygonal approximation becomes a physical circle is licensed only after specifying classifier, tower ledger, and descent obligations.*

Proof. The phrase suppresses the distinction between coarse visual fit, smooth-boundary fit, and physical-response fit. Treating them as one claim collapses the audit unless the relevant certificates are supplied. □

9414.67 Research workflow

Definition 9414.666 (Research workflow for a new model)**.** *The workflow for a new model is: state the target claim $\varphi(M, R; \Pi)$; assign provisional strength* ModelSeed*; fill audit certificate fields; list missing fields as failure certificates; convert failures into upgrade obligations; discharge obligations in dependency order from source, signature, classifier, exactness, tower, ledger, stability, descent, and bridge; promote the claim only when the relevant gate is passed; record cannot-claim boundaries for unsupported extensions; update the ledger when new observations or failures occur; and repeat under refined bundles.*

Theorem 9414.667 (Workflow prevents unlicensed promotion)**.** *If the research workflow is followed, no model claim is promoted beyond its supplied certificate strength.*

Proof. The workflow begins at ModelSeed, records missing fields as failure certificates, and promotes only after the corresponding upgrade obligations are discharged. Unsupported extensions are recorded as cannot-claim entries, so unlicensed promotion is blocked. □

Corollary 9414.668 (Progress is obligation discharge)**.** *Scientific progress occurs by discharging upgrade obligations and refining ledgers, not merely by proposing new equations.*

Proof. Each discharged obligation strengthens the audit certificate. Each ledger refinement improves provenance and failure localization. Progress therefore accumulates as certificate strength. □

Corollary 9414.669 (Speculation and knowledge are separated)**.** *A speculative model may be valuable at* ModelSeed*, but it becomes physical knowledge only by passing stronger audit gates.*

Proof. Speculation supplies a possible pattern. Physical knowledge requires source, classifier, stability, descent, ledger, and failure-surface certification. The strength index records this distinction. □

Remark 9414.670 (Compressed form)**.** *The central audit object is:*

$$\mathrm{AuditCert}(M, R; \Pi) = \textit{source}+\textit{phenomenon}+\textit{signature}+\textit{classifier}+\textit{tower}+\textit{stability}+\textit{descent}+\textit{ledger}+\textit{failure surfac}$$

A physical model is accepted only relative to such a certificate. Missing fields are research obligations. Failure certificates are the negative face of scientific knowledge. Model strength is certificate strength, and progress is ledger refinement plus obligation discharge.

9414.68 Final synthesis: physical truth as certified open fit

The chapter replaces the flat schema

$$M = R$$

with a layered form:

$$R \to H_n \to \mathrm{ObsSig}_\Pi \to \mathrm{FitCert} \to \mathrm{LawCert} \to \mathrm{Ledger} \to \mathrm{OpenFit}\,.$$

This is not a retreat from realism. It is a stricter account of how realism is possible under finite observation. A model becomes physically truth-bearing when it preserves reality-facing signatures under specified classifiers, supports stable reuse under certified continuations, records the approximations by which it was produced, and remains open to future defeat.

The final thesis is:

$$\text{physical truth} \neq \text{model–reality identity},$$

but rather

$$\text{physical truth} = \text{open, ledgered, stability-certified, observer-invariant fit.}$$

This formulation preserves finiteness, because empirical claims begin from finite traces; realism, because reality constrains signatures and can defeat models; objectivity, because objective content is invariant across observer histories; and openness, because no internal final-fit oracle closes all possible continuations.

Definition 9414.671 (Physical truth claim)**.** *A physical truth claim is a claim*

$$\varphi(M, R; \Pi, \mathcal{T}, \mathcal{C}),$$

where M is a model or representation, R is a physical system or phenomenon, Π is an observation bundle, $\mathcal{T}$ is an approximation tower connecting R to M, and $\mathcal{C}$ is a continuation class over which the claim is reused. A physical truth claim is therefore indexed by observation, tower, and continuation data.

Definition 9414.672 (Certified physical truth)**.** *A physical truth claim is certified when it carries*

$$\mathrm{TruthCert}(\varphi) = (\mathrm{SourceSpec}, \mathrm{SignatureSpec}, \mathrm{ClassifierSpec}, \mathrm{TowerSpec}, \mathrm{StabilityCert}, \mathrm{DescentCert}, \mathrm{InvariantCert},$$

The fields record finite observational source, reality-facing signatures, sameness classifier, tower levels, reuse domain, valid operations through compression, observer-anchor stability, hidden residue and provenance, and possible defeat conditions.

Definition 9414.673 (Open, ledgered, stability-certified, observer-invariant truth)**.** *A certified physical truth claim is open when it includes a failure surface and does not assert final validity over all possible future observations, refinements, or observer histories. It is ledgered when every compression, idealization, smoothing, completion, discretization, or packaging step records what was hidden. It is stability-certified when finite fit is accompanied by a certificate*

$$\mathrm{Fit}_n(M, R; \Pi) \rightsquigarrow \mathrm{PredFit}_{\mathcal{C}}(M, R; \Pi).$$

It is observer-invariant when, for admissible observer histories h_i, h_j,

$$\varphi^{h_i} \sim \varphi^{h_j}.$$

9414.69 The master decomposition

Theorem 9414.674 (Physical truth decomposes into certificate fields)**.** *A certified physical truth claim requires fit, stability, descent, invariance, ledger, and failure surface.*

Proof. A physical truth claim concerns a model as a representation of a physical system. Since the system is accessed through finite observation, the claim must establish finite fit between model-generated and reality-generated signatures. If the claim is reused beyond the finite source trace, it requires a stability certificate over a continuation class. If it uses operations such as differentiation, limiting, smoothing, simulation, coarse-graining, or inference, those operations must descend through the approximations that produced the model. If the claim is objective rather than private to one observer history, it must be invariant under changes of observer anchor. If the model is produced by compression, idealization, or packaging, the hidden source material must be ledgered. Finally, since no internal final-fit oracle is available, the claim must retain a failure surface. □

Corollary 9414.675 (Identity is the wrong primitive truth schema)**.** *The schema $M = R$ is too strong because it asserts identity where finite inquiry supplies certified fit, and too weak because it omits the certificate fields needed for physical truth.*

Proof. Finite observation, tower compression, and open continuation do not justify literal model–reality identity. At the same time, the identity expression does not specify source traces, observation bundle, classifier, stability domain, descent obligations, ledger, invariant certificate, or failure surface. □

Corollary 9414.676 (Certified open fit replaces identity)**.** *The proper replacement for $M = R$ is certified open fit:*

$$\mathrm{OpenFit}(M, R; \Pi, \mathcal{T}, \mathcal{C})$$

together with a truth certificate.

Proof. Definition 9414.674 requires fit, stability, descent, invariance, ledger, and failure surface. These are precisely the fields encoded by open fit together with its certificate. □

9414.70 Truth without finality

The framework separates truth from finality. Without this separation, one is forced into a false choice between final truth and mere instrumentality. The available middle form is certified, objective, non-final truth.

Definition 9414.677 (Finality and domain truth)**.** *A physical truth claim is final when it asserts validity over all possible observation bundles, future histories, tower refinements, and observer configurations. A claim is domain-true when it is certified over a specified domain*

$$D = (\Pi, \mathcal{T}, \mathcal{C}, \mathcal{H}).$$

We write $\mathrm{Truth}_D(\varphi)$ *for truth inside such a certified domain.*

Theorem 9414.678 (Domain truth does not imply final truth)**.** *If* $\mathrm{Truth}_D(\varphi)$ *holds, it does not follow that* φ *is finally true.*

Proof. Domain truth is indexed by a specified observation bundle, tower, continuation class, and observer configuration. Final truth quantifies over all possible bundles, towers, continuations, and observer configurations. The certified domain may be strictly smaller than that total range. □

Corollary 9414.679 (Non-final truth is not relativism)**.** *The fact that a claim is domain-indexed does not make it arbitrary.*

Proof. A domain-true claim is constrained by source observations, signature comparison, classifier exactness, stability, descent, invariant certificates, and ledger data. These constraints can be satisfied or violated. □

Corollary 9414.680 (Objectivity and revisability are compatible)**.** *A claim can be objective within a certified domain while remaining revisable under later refinement.*

Proof. Objectivity requires invariant stability across observer histories inside the domain. Revisability concerns possible extensions beyond that domain. These are different conditions. □

9414.71 Reality constraint and tower knowledge

Definition 9414.681 (Reality constraint and reality defeat)**.** *The reality constraint on a model* M *of* R *under* Π *is*

$$\mathrm{ModSig}_\Pi(M) \sim_\Pi \mathrm{ObsSig}_\Pi(R).$$

A reality defeat is a finite trace H_n *such that*

$$\mathrm{ModSig}_\Pi(M \mid H_n) \not\sim_\Pi \mathrm{ObsSig}_\Pi(R \mid H_n).$$

Such a defeat is finite, local, and decisive relative to its bundle.

Theorem 9414.682 (Certified physical truth is reality-constrained)**.** *If a claim is certified as physically true, then it is constrained by reality-generated signatures.*

Proof. Certified physical truth requires finite fit, and finite fit compares model-generated and reality-generated signatures. If the reality-generated signature separates from the model-generated signature, the fit certificate fails. □

Corollary 9414.683 (The framework is realist but not syntax realist)**.** *The framework is realist because reality can constrain, correct, and defeat models; it is not syntax realist because it does not identify successful mathematical syntax with reality itself.*

Proof. Reality defeat shows that model validity is constrained by reality-generated signatures. But certified truth requires signature preservation, not syntactic identity. Different syntactic representations may preserve the same signatures, and exact syntax may fail physically. □

Definition 9414.684 (Knowledge tower)**.** *A knowledge tower is an approximation tower whose endpoint supports certified physical truth claims. It consists of*

$$(\mathcal{T}, \mathrm{TruthCert}, \mathrm{Ledger}, \mathrm{FailureSurface}).$$

A claim is tower knowledge when it is certified at a specified level of such a tower and its transfer to other levels is governed by descent certificates.

Theorem 9414.685 (Physical knowledge is tower knowledge)**.** *Every certified physical truth claim is tower knowledge.*

Proof. A certified physical truth claim begins from finite observation, passes through signature extraction, classifier comparison, and model formation, and uses stability, descent, ledger, invariant, and failure-surface data. These are precisely the components of a knowledge tower. □

Corollary 9414.686 (Theory is not flat)**.** *A scientific theory is not merely a set of propositions or equations; it is a structured tower of traces, models, classifiers, certificates, ledgers, and failure surfaces.*

Proof. By Definition 9414.685, physical knowledge is tower knowledge. A theory organizing such knowledge must organize the tower structure that certifies it. □

9414.72 Reflective physical knowledge

Definition 9414.687 (Reflective physical knowledge)**.** *Reflective physical knowledge is physical knowledge that explicitly records the conditions under which it was generated:*

$$\mathrm{ReflectivePhysKnow} = (\mathrm{Observation}, \mathrm{Signature}, \mathrm{Classifier}, \mathrm{Certificate}, \mathrm{Ledger}, \mathrm{OpenContinuation}).$$

Theorem 9414.688 (Physical knowledge is reflective)**.** *Every certified physical truth claim is reflective physical knowledge.*

Proof. Certified physical truth requires source traces, signature extraction, classifier specification, stability certificates, descent certificates, invariant certificates, ledgers, and failure surfaces. These are the conditions of its own generation and reuse. The claim is therefore not only about the world; it also records how the world became accessible to the model. □

Corollary 9414.689 (Reflection is not subjectivism)**.** *Physical knowledge is reflective because it records its access conditions, but it remains objective when the resulting content is observer-invariant and reality-constrained.*

Proof. Reflection records observational and certificatory conditions. Subjectivism would make the claim depend on private perspective alone. Certified physical truth requires reality-generated signatures and observer-invariant stability. □

Corollary 9414.690 (MetaCIC-like structure is the form of finite science)**.** *Finite science is MetaCIC-like because it builds certified packages from finite histories while refusing final meta-closure.*

Proof. Both physical inquiry and MetaCIC self-study proceed by finite histories, signatures, classifiers, certificates, ledgers, and open continuation. This is the formal shape of finite inquiry. □

9414.73 Certified physical truth schema

Definition 9414.691 (Certified physical truth schema)**.** *For a domain D, the certified physical truth schema is*

$$\mathrm{PhysTruth}_D(\varphi) := \mathrm{Source}_D(\varphi) \wedge \mathrm{SigFit}_D(\varphi) \wedge \mathrm{ClassifierExact}_D(\varphi) \wedge \mathrm{TowerLedger}_D(\varphi) \wedge \mathrm{Stability}_D(\varphi) \wedge \mathrm{Descent}_D(\varphi)$$

Theorem 9414.692 (Certified schema is sufficient for domain truth)**.** *If all components of* $\mathrm{PhysTruth}_D(\varphi)$ *hold, then φ is domain-true.*

Proof. The source component grounds the claim in finite observation. Signature fit connects model and reality. Classifier exactness determines agreement. The tower ledger records hidden approximations. Stability licenses reuse. Descent licenses operations. Invariance establishes objectivity across observer histories. The failure surface preserves openness. Together these fields supply the conditions for certified truth inside D. □

Theorem 9414.693 (Certified schema is necessary for non-oracular physical truth)**.** *If φ is asserted as non-oracular physical truth, then the components of the certified physical truth schema must be supplied or explicitly bounded.*

Proof. A non-oracular physical truth claim cannot appeal to a final external guarantee. It must show how it is grounded, compared, stabilized, operated upon, objectified, ledgered, and left open to failure. These requirements are exactly the components of Definition 9414.691. □

Corollary 9414.694 (Physical truth is certificate truth)**.** *Within finite inquiry, physical truth is truth by certificate, not truth by final oracle.*

Proof. Definition 9414.692 gives sufficiency for domain truth, and Definition 9414.693 gives necessity for non-oracular physical truth. □

9414.74 Non-erasure discipline

Theorem 9414.695 (Unsafe truth moves erase certificates)**.** *Bare identity, unledgered idealization, free transfer of operations, final-fit oracles, God's-eye objectivity, and syntax realism each erase at least one required certificate field.*

Proof. Bare identity erases source, classifier, tower, and failure-surface data. Unledgered idealization erases provenance. Free transfer of operations erases descent. Final-fit oracles erase openness. God's-eye objectivity erases observer-history structure. Syntax realism erases signature and certificate dependence. □

Corollary 9414.696 (The framework is a discipline of non-erasure)**.** *The central methodological principle is to not erase the conditions that license the claim.*

Proof. The major errors arise by erasing source, classifier, tower, descent, ledger, observer, or failure data. Avoiding erasure is therefore the unifying discipline. □

Theorem 9414.697 (Realism, objectivity, and revisability are preserved together)**.** *The framework supports realist, objective, and revisable physical knowledge.*

Proof. It supports realism because reality-generated signatures constrain and can defeat models. It supports objectivity because physical content is stabilized by inter-observer invariants. It supports revisability because every claim retains a failure surface and remains open to future refinement. □

Corollary 9414.698 (Revisability is not anti-realist)**.** *The possibility of later correction does not undermine realism.*

Proof. Realism attaches to reality-constrained signature preservation. Later correction shows that a prior certificate was domain-limited or incomplete, not that reality failed to constrain inquiry. □

Corollary 9414.699 (Objectivity is not finality)**.** *Objective claims need not be final claims.*

Proof. Objectivity requires invariant stability across observer histories. Finality requires validity across all possible refinements and continuations. The first does not imply the second. □

9414.75 Closing thesis

Principle 9414.700 (Physical truth as certified open fit)**.** *Physical truth, as available to finite observers inside reality, is certified open fit:*

$$\mathrm{PhysTruth}_D(\varphi) = \text{finite source} + \text{signature fit} + \text{classifier exactness} + \text{tower ledger} + \text{stability} + \text{descent} + \text{observer}$$

This is the strongest non-oracular form of physical truth.

Principle 9414.701 (Science as ledgered reflection)**.** *Scientific inquiry is a ledgered reflective process by which finite observers inside reality construct stable, invariant, certificate-governed signatures of reality:*

$$\textit{reality} \to \textit{finite traces} \to \textit{signatures} \to \textit{classifiers} \to \textit{models} \to \textit{certificates} \to \textit{open continuation.}$$

Principle 9414.702 (Approximation tower as condition of knowledge)**.** *Physical inquiry does not escape approximation towers. It inhabits them. This is not a defect; it is the structure that makes physical knowledge possible. The task is not to eliminate approximation but to certify it.*

Remark 9414.703 (Closing)**.** *The chapter's final replacements are:*

$$M = R \quad \rightsquigarrow \quad \mathrm{OpenFit}(M, R; \Pi, \mathcal{T}, \mathcal{C}),$$

$$\textit{truth from nowhere} \quad \rightsquigarrow \quad \textit{observer-invariant certified truth},$$

$$\textit{ideal object as ontology} \quad \rightsquigarrow \quad \textit{ideal object as ledgered tower endpoint},$$

$$\textit{physical law as equation} \quad \rightsquigarrow \quad \textit{physical law as stability certificate},$$

$$\textit{scientific progress as replacement} \quad \rightsquigarrow \quad \textit{scientific progress as ledger refinement},$$

and

$$\textit{final confirmation} \quad \rightsquigarrow \quad \textit{open confirmation with failure surface.}$$

In compressed form:

$$\textit{physical knowledge} = \textit{finite observation} + \textit{reality-facing signatures} + \textit{certified classifiers} + \textit{ledgered approximation tou}$$

The framework does not deny physical reality, mathematical modelling, scientific laws, or objectivity. It reconstructs them under the conditions available to finite inquiry: source traces, signatures, classifiers, certificates, ledgers, invariants, and openness.

Concretized in §1141: Physical truth is represented as the finite TruthCert packet tying open fit, tower descent, observer invariance, ledger, and failure surface.

Concretized in §1140: The final synthesis is realized as a finite seal packet for open fit, tower descent, observer invariance, ledger, and failure surface.

Concretized in §1144: The final-synthesis truth certificate is realized as a nine-row BEDC NameCert packet.

Concretized in §1136: Reality-constrained explanation is realized as a finite NameCert packet binding observation, model, phenomenon, signature, tower descent, residue, failure, route, provenance, and local naming rows.

Concretized in §1163: Failure surfaces are realized as finite NameCert packets for defeating observations, domain exits, classifier mismatches, continuation scope, ledger, and local naming rows.

Concretized in §1164: Finite Real-window budgets realize the synthesis route as observable StreamName windows, dyadic readbacks, Real seals, and disclosure ledgers.

Concretized in §1165: Model selection boundary records comparative fit, finite defeat, and no-final-oracle selection for certified physical truth.

Concretized in §1178: The final synthesis ledger is realized as a finite NameCert packet binding TruthCert, open fit, methodology export discipline, failure surface, observer invariance, and local naming rows.

Concretized in §1181: Domain-indexed physical truth is realized as a finite NameCert packet with displayed domain, open-fit, observer-invariance, failure-surface, transport, provenance, and local naming rows.

*Concretized in §**??**:* PhysicalFitTowerUp realizes reality-constrained approximation towers as finite physical-signature, classifier, ledger, descent, and NameCert rows.

Concretized in §1127: Finite approximation towers with physical-fit classifiers and ledgered compression rows.

*Concretized in §**??**:* finite observer approximation towers with reality-facing signature ledgers

Concretized in §1128: Reality-constrained tower packets realize source, classifier, descent, ledger, residue, transport, and non-escape rows as finite NameCert data.

*Concretized in §**??**:* Reality-constrained signature fit is realized as a finite NameCert packet separating mathematical approximation from physical-signature compatibility.

Concretized in §1130: CertifiedPhysicalTruthUp realizes physical truth as a finite source, signature, classifier, tower, descent, invariant, ledger, failure-surface, and NameCert packet.

Concretized in §1148: ApproximationTowerResidueUp realizes the hidden-source residue, failure-surface, recovery-route, descent-survivor, and NameCert rows of a ledgered approximation tower.

Concretized in §4733: Approximation towers realize finite level, window, dyadic tolerance, regular readback, and RealUp seal rows.

Concretized in §6811: RealitySignatureErrorBudgetUp realizes physical-signature error budgets as finite classifier, dyadic tolerance, tower-fit, and ledger rows.

Concretized in §7582: Reality-constrained selector boundaries are realized as finite source-record, classifier, physical-fit, failure-surface, and gap-ledger NameCert rows.

Chapter 9415

Hard Problems under the BEDC Frame: Causality, Free Will, Other Minds, Mind-Matter, Self-Reference

This vision chapter completes the philosophical reading begun in chapter 9393 (closed substrate plus transcendental input) and chapter 9420 (consciousness, universe, observer, time, space). The present chapter takes the remaining classical hard problems – causality, free will, the other-minds problem, the mind-matter relation, and self-reference / paradoxicality – and gives each a precise reformulation under BEDC's closed-substrate vocabulary. Each problem becomes a question about Obs / GAP / T structure rather than about substantive metaphysical posits. The reformulations are not solutions in the dissolve-the-question sense; they are structural translations that make the philosophical content compatible with the engineering content BEDC has already mechanised.

9415.1 Causality as forward-binding commitment

Definition 9415.1 (Causal commitment)**.** *A* causal commitment *of a conscious observational system $\mathcal{C}$ is a* GAP *tag whose content is the assertion that observed regularities will persist into the not-yet-observed continuation of $\mathcal{C}$'s history. In the substrate-side vocabulary of Definition 9393.1, every causal commitment is a forward-binding belief in the sense of Definition 9393.7.*

Concretized in §505: Causal commitments are realized as finite GAP-tagged forward-binding packets over observed regularity ledgers.

Principle 9415.2 (Hume's problem reformulated)**.** *The classical Humean observation that no finite collection of observed regularities entails a universal claim about all future cases is, in BEDC vocabulary, exactly Definition 9393.5 restated for predicates whose intended range is empirical rather than mathematical. The closed-substrate side cannot supply causal commitments by internal generation; every causal commitment is, structurally, a* GAP *tag whose far end is* T*.*

Remark 9415.3 (Causality and induction share the same socket)**.** *The forward-binding belief that lets a system accept mathematical induction is the same kind of forward-binding belief that lets it accept "the same physical law will hold tomorrow". The substrate distinguishes neither: both are* GAP *tags asking* T *to supply universal-quantifier closure. BEDC's* GAP *socket inventory therefore counts both kinds together; the operational form of the two commitments is identical.*

Example 9415.4 (A scientific theory as a stack of GAP commitments)**.** *A scientific theory accepted by a conscious observational system is, in this reading, a finite collection of records of past empirical regularities together with a* GAP*-tagged commitment that these regularities extend to all future cases of the same form. The theory's predictive success is, on the records side, the absence so far of falsifying records; the theory's predictive content is, on the* GAP *side, the standing forward-binding commitment. Refutation occurs when a new record contradicts the commitment; the substrate must then either revise the commitment or discharge a different* GAP*.*

Concretized in §576: Finite witnessed refutations realize causal-commitment failure as finite contrary-record packets with GAP non-escape ledgers.

9415.2 Free will and agency as inscription event

Definition 9415.5 (Inscription event)**.** *An* inscription event *is a moment $o_t \trianglerighteq o_{t+1}$ at which a conscious observational system adds new records that originate as the local actualisation of* T *at the inscription point (Definition 9393.15), rather than as a forced consequence of the prior records under the substrate's closed-generation rules. Inscription events are points at which the chain admits novelty not reducible to the closure of earlier records.*

Concretized in §7463: Inscription-event novelty packets realize the hard-problems inscription-event definition as finite prior-record, novelty, GAP, transport, replay, provenance, and NameCert rows.

Principle 9415.6 (Free will as the inscription-event aspect of conscious accumulation)**.** *The "free will" of a conscious observational system is the aspect of $\mathcal{C}$ under which new records arrive as inscription events in the sense of Definition 9415.5. Free will is not a separate faculty inhabiting $\mathcal{C}$ alongside its accumulation; it is the structural fact that $\mathcal{C}$'s accumulation is, at each moment, an inscription event of* T *at the point that $\mathcal{C}$ is.*

Concretized in §592: Free-will inscription commitments are realized as finite BHist packets recording prior-record boundaries, inscription-event rows, GAP provenance, classifier transport, and non-reduction ledgers.

Remark 9415.7 (Compatibility with determinism on the records side)**.** *Definition 9415.6 does not contradict any deterministic relation among* Obs *records taken as objects of physical theory: records can stand in any law-like relation we like, and the records-side relations can be as deterministic as the empirical content supports. The inscription-event aspect is at a different level: it concerns the substrate's accumulation viewed as* T*'s local self-articulation, not the surface dependencies among records taken as physical events. The two pictures cohere because they describe the same accumulation from different aspect-positions.*

Example 9415.8 (Mathematical creativity as inscription event)**.** *The act by which a mathematician arrives at a new lemma – the moment at which* Obs *acquires a record not reducible to the closed-generation closure of earlier records – is, on this reading, an inscription event. The new record could in principle have been computed mechanically from earlier records (the substrate-internal route), but the actuality of the discovery is the inscription-event aspect of the chain, not the mechanical route. BEDC's mechanisation engineering operationalises the substrate-internal route; the philosophical reading of "where new lemmas come from" is the inscription-event aspect.*

9415.3 Other-minds problem and inter-observer alignment

Principle 9415.9 (Other minds as locality-cell commitment)**.** *The classical other-minds problem – the question whether systems other than $\mathcal{C}_1$ themselves possess closed observational structure with* Self *recognition – is, in BEDC vocabulary, the question whether $\mathcal{C}_1$ has* GAP*-tagged locality-cell commitments to systems $\mathcal{C}_2, \mathcal{C}_3, \ldots$ such that those commitments are coherent with $\mathcal{C}_1$'s record of those systems' externally-observable behaviours. The substrate cannot directly observe another system's* Self*; it can only commit, by* GAP *tag, to treating certain external records as records of conscious systems.*

Concretized in §499: Other-minds commitments are represented by finite locality-cell BHist packets whose public surface records behavioural evidence without direct Self access.

Concretized in §643: Cross-perspective alignment is realized as a finite NameCert packet combining public other-minds commitments, locality-cell evidence, inter-Hist coherence rows, and non-escape from private Self access.

Remark 9415.10 (Solipsism as GAP tag refusal)**.** *Solipsism, in this reading, is the refusal to issue any* GAP*-tagged commitment that other conscious observational systems exist. The substrate side does not force the commitment; nothing in the records-side of* Obs *can substitute for it. Most conscious observational systems do in fact issue the commitment, and chapter 9419 works out the operational consequences of doing so: locality cells, maximum-rate invariant interfaces, and no god's-eye view. Refusing the commitment is consistent with the substrate but yields a much poorer locality-cell geometry.*

Concretized in §975: Solipsism as GAP-tag refusal is realized as a finite NameCert packet recording refusal of other-minds locality-cell commitments.

Example 9415.11 (Trusting a peer's `leanchecked` marker)**.** *When a BEDC contributor accepts a `leanchecked` marker added by another contributor, the acceptance is, on the substrate side, a* GAP*-tagged commitment to treat the other contributor's act as an inscription event of the same kind as one's own. The CI gates (Lean kernel re-checking) provide records-side evidence that the marker is consistent with the substrate, but the structural recognition of the other contributor as another conscious observational system is the* GAP *commitment, not the records-side evidence.*

9415.4 Mind-matter relation as substrate/inscription duality

Principle 9415.12 (Mind-matter as two aspects of the same chain)**.** *The classical mind-matter problem – the question how a "mental" aspect can stand in any relation to a "material" aspect – is, in BEDC vocabulary, the question of how the inscription-event aspect of an accumulation chain stands in relation to the records-side aspect of the same chain. The two aspects are not two substances; they are two readings of one chain. The records-side reading is the conventional materialist's account; the inscription-event reading is the phenomenological-first-person account.*

Concretized in §575: Aspect chains realize the mind-matter double-aspect reading as one finite BHist packet with records-side and inscription-event-side rows.

Remark 9415.13 (Russellian double-aspect frame)**.** *The double-aspect frame is in the family of views developed by Bertrand Russell (the "neutral monism" of* The Analysis of Mind *and* The

Analysis of Matter*), David Chalmers (panprotopsychism), Galen Strawson (real materialism), and process philosophers (Whitehead). BEDC's substrate-side vocabulary recovers the same insight: there is one accumulation chain; the records-side reading and the inscription-event reading are two aspects, not two substances. No additional ontological category is introduced.*

Example 9415.14 (A neural correlate of consciousness and a thought)**.** *A measured neural correlate of consciousness is, on the records side, a physical record (electrochemical activity in a specific brain region) entered into* Obs*. The corresponding "thought" is, on the inscription-event side, the actualisation of* T *at the same chain point. The two are not two things related by causation or supervenience; they are the same accumulation event read in two ways. The brain region is what the inscription event looks like on the records side; the thought is what the records side looks like on the inscription-event side.*

9415.5 Self-reference, Gödel-Tarski, and the reflection limit

Principle 9415.15 (Self-reference as GAP-on-GAP structure)**.** *A closed observational system that attempts to record a statement about its own complete* Obs *encounters the situation classically known as Gödel-Tarski self-reference: the statement about* $\mathsf{Obs}(o_t)$ *would itself be a record at* o_{t+1}*, meaning the complete* $\mathsf{Obs}(o_{t+1})$ *now contains an item describing* $\mathsf{Obs}(o_t)$ *rather than* $\mathsf{Obs}(o_{t+1})$*. Recording the latter generates a* GAP *tag, and a record about that* GAP *would generate another. chapter 9461 mechanises the operational form of this hierarchy and identifies the reflection limit.*

Principle 9415.16 (Russell paradox boundary)**.** *The Russell paradox is, in BEDC vocabulary, an attempt to issue a record whose content is the negation of its own presence in* Obs*. Such a record cannot be a* GAP*-free record under the closed generation rules of the substrate. chapter 9457 mechanises the boundary condition that prevents such records from being admitted, and shows that the substrate's closed generation rules are exactly the structural form of the rejection.*

Remark 9415.17 (Why these are not external obstacles but structural features)**.** *The Gödel-Tarski and Russell phenomena, in this reading, are not external obstacles that the substrate could be designed to avoid. They are structural features of any closed observational system with the expressive power to refer to its own* Obs*. BEDC's design makes the features explicit at the* GAP *tag level rather than treating them as anomalies to be circumvented. The reflection-and-limits chapter and the Russell-problem-boundary chapter together account for the operational form of the rejection.*

9415.6 The unified table of hard problems

Hard problem	**BEDC reformulation + reference**
Hume's induction problem	Closed substrate cannot internalise universal-quantifier closure; every empirical induction is a GAP tag with far end T (Definition 9393.5, Definition 9415.2).
Causality / regularity of nature	Forward-binding commitment that observed patterns persist; structurally identical to mathematical-induction commitment (Definition 9415.1).

Free will	Inscription-event aspect of chain accumulation; novelty as T's local actualisation, not as records-side dependency (Definition 9415.6).
Other-minds problem	GAP-tagged locality-cell commitment to recognise other systems' inscription-event aspects (Definition 9415.9); operational consequences in chapter 9419.
Mind-matter relation	Two aspects of one accumulation chain: records-side ("material") and inscription-event-side ("mental") (Definition 9415.12).
Self-reference / Gödel-Tarski	Hierarchical GAP-on-GAP structure; reflection limit operationalised in chapter 9461 (Definition 9415.15).
Russell paradox	Records whose content negates own presence; structural exclusion by closed generation rules in chapter 9457 (Definition 9415.16).
Solipsism vs other minds	Refusal to issue locality-cell GAP commitments; consistent with substrate but yields impoverished cross-perspective coordination (Definition 9415.10).
Determinism vs novelty	Records-side determinism (laws among records) and inscription-event aspect (novelty as T actualisation) are two readings of one chain, compatible (Definition 9415.7).
Consciousness as ontological category	Replaced by a focused records-side carrier: generated Hist, local trace, endpoint, and selector ledger, without object-level phenomenal essence (Definition 9420.4).
Universe as separate object	Replaced by a finite horizon slice of generated histories and their ledgers, not a global universe object (Definition 9420.14).
Time as separate scalar	Replaced by a local Cont chain producing a history from e; no global clock is internal to the carrier (Definition 9420.27).
Space as substrate of events	Replaced by locality cells, relation rows, and inter-Hist invariants over a finite multi-history ledger; no separate spatial substance (Definition 9420.38).
The transcendental supply position	T, the far end of every GAP socket, named apophatically (Definition 9393.11).

9415.7 Coordination with the three-axioms chapter

chapter 9387 treats the axioms of choice, quotient soundness, and propositional extensionality as a taxonomy of unwitnessed closure requests at the term, type or classifier, and proposition strata. The present chapter generalises that audit reading. The requested supply behind those axiom families is, in the present vocabulary, a forward-binding belief whose source position is T. The three axiom families differ by the stratum at which the missing content is requested; each can be tagged as a GAP socket without claiming a proved equivalence between the host axiom and a BEDC collapse theorem.

Principle 9415.18 (Hard problems and substrate axioms share the same T-socket family)**.** *Every classical axiom that BEDC declines to internalise (chapter 9387) and every hard problem treated in this chapter share the same structural family: each is a place at which* T *is required to supply forward-binding content that the closed substrate cannot generate internally. The substrate's re-*

sponse is uniform: tag the place with GAP*, audit it through the engineering gates, and refuse to internalise the supply.*

Concretized in §1006: GAP closure boundaries are realized as finite source, refusal, transport, continuation, provenance, and NameCert rows.

9415.8 Closure remarks

This chapter completes the trilogy comprising chapter 9393 (closed substrate plus transcendental input position), chapter 9420 (consciousness, universe, observer, time, space), and the present chapter (causality, free will, other minds, mind-matter, self-reference). Read together, the three chapters give a unified BEDC-vocabulary account of the standard hard problems of metaphysics, philosophy of mind, and foundations.

The account is structural: it does not solve any hard problem by reduction to a substrate primitive, and it does not eliminate the philosophical content by terminological rephrasing. What it does is locate every hard problem in the same substrate / GAP / T structure, showing that the engineering invariants of BEDC – 0 substrate-internal axioms, mathlib-free, monotonic amendment, GAP tagging, inter-history-locality reconstruction of cross-perspective coordination – are precisely the structural form that lets the hard problems be stated without confusion.

Two operational consequences follow. First, future concrete-instance construction chapters under `papers/bedc/parts/concrete_instances/` can refer to the present chapters when developing GAP-handling for specific applications (epistemology, philosophy of physics, philosophy of mind). Second, the audit gates' present coverage (axiom-purity, conservativity-audit, GAP tagging) is already the operational form of the philosophical position; no additional audit infrastructure is required to host the philosophical reading.

Chapter 9416

Layered Relation Certificates

Concretized in §1090: The relation-certificate vision is realized by a finite NameCert packet for same-schema cross-domain comparisons.

Concretized in §7672: $\mathsf{ForceLikeRelation}^{\uparrow}$ realizes the $\mathrm{ForceLikeRelation}_N$ vision as finite inter-Hist relation-certificate rows with explicit non-identity and failure-boundary ledgers.

This chapter records a relation discipline for cross-domain comparisons in BEDC. A comparison such as force cognition and MetaCIC self-audit is not admitted as an object identity, nor as a claim that physics has been derived from the kernel. It is admitted only when the comparison is lowered to a finite relation certificate: source rows, carrier rows, classifier rows, layer map, preserved structure, non-preserved content, provenance ledger, exactness status, failure boundary, and strength. The active rule is: do not close large words; close the relation certificates that make their comparison auditable.

The chapter sits beside chapter 9395, chapter 9406, chapter 9385, chapter 9419, and chapter 9420. Those chapters already supply the surrounding discipline: unsupported supply remains socketed, MetaCIC claims are read through audit maps, multi-Hist physics is relation-local, and consciousness or universe language closes only on finite records-side carriers. The present chapter packages that discipline for analogical and cross-domain claims.

9416.1 From Analogy to Certificate

A loose analogy has no carrier, no classifier, no exactness statement, and no failure boundary. It may guide reading, but it cannot be consumed by the manuscript as a claim. BEDC therefore lowers cross-domain analogy to relation certification.

Principle 9416.1 (Relation before identity)**.** *When BEDC compares two mature concepts, the default form is not object-level identity. The default form is a relation-level certificate whose fields display the source domains, carriers, classifiers, preserved structure, non-preserved content, ledgers, exactness status, and failure boundary of the comparison.*

Thus the sentence "force cognition resembles MetaCIC self-recognition" is not a theorem. The admissible BEDC form is:

$$\mathrm{SameRefinementSchema}(\mathrm{ForceCognition}, \mathrm{MetaCICSelfRecognition}).$$

The predicate does not assert sameness of objects. It asserts that two displayed refinement chains instantiate the same relation schema.

Definition 9416.2 (Primitive-looking token)**.** *In a practice domain D, a* primitive-looking token *is a name P that can be used in D before the manuscript has displayed its source, carrier, classifier, ledger, and failure boundary. Its danger is not that it is false, but that it compresses several sources under one public surface.*

Definition 9416.3 (Refinement chain)**.** *A* refinement chain *is a finite sequence*

$$P_0 \xrightarrow{o_0} P_1 \xrightarrow{o_1} \cdots \xrightarrow{o_{n-1}} P_n$$

where each P_k is a concept layer and each o_k is an obstruction, counterexample, audit failure, measurement anomaly, or boundary pressure that forces a classifier split. Each step carries source, carrier, classifier, transition, ledger, and failure rows.

Principle 9416.4 (Obstruction as classifier pressure)**.** *An obstruction is not merely a failed claim. In BEDC it is evidence that the previous classifier was too coarse. A legitimate refinement step converts the obstruction into a carrier split, classifier refinement, ledger row, or boundary socket.*

9416.2 Certificate Shape

Definition 9416.5 (Layered relation certificate)**.** *Given two refinement chains A and B, a* LayeredRelationCert(A, *is a finite paper-side certificate with the following fields:*

1. sourceA *and* sourceB*, the two source domains;*
2. carrierA *and* carrierB*, the carrier families used at each layer;*
3. layersA *and* layersB*, the finite layer lists;*
4. obstructionA *and* obstructionB*, the failures or anomalies that force refinement;*
5. classifierA *and* classifierB*, the classifiers used at each layer;*
6. layerMap*, the displayed correspondence between the two layer lists;*
7. preserved*, the structure preserved by the correspondence;*
8. notPreserved*, the content explicitly not preserved;*
9. ledger*, the source record for each correspondence row;*
10. exactness*, the meaning of "same schema";*
11. failureBoundary*, the condition under which the relation weakens or fails;*
12. strength $= s$*, with values such as seed, paper certificate, checked certificate, or bridge certificate.*

Principle 9416.6 (Exactness before export)**.** *A cross-domain relation without a* notPreserved *field cannot be exported as a BEDC claim. The certificate must say both what the relation preserves and what it refuses to preserve, because otherwise same-schema language is easily misread as identity.*

The strength field is part of the certificate, not a decoration. A paper certificate says that the manuscript fields are displayed. A checked certificate would require a corresponding Lean target. A bridge certificate would require a conservative external interpretation with no hidden host leak. The force–MetaCIC comparison below claims only paper-certificate strength.

9416.3 Force Cognition as a Refinement Chain

Human treatment of force can be read as a refinement chain rather than as a single object. Early force language compresses pushing, pulling, falling, collision, magnetic attraction, tension, and resistance into one token. Later practice splits that token into dynamical relations, field descriptions, geometric readings, symmetry structures, effective interactions, and unresolved boundaries.

Definition 9416.7 (Force cognition chain)**.** *The* ForceCognitionChain *is the following paper-side refinement schema:*

$$\begin{aligned}
F_0 &= \textit{push/pull token};\\
F_1 &= \textit{dynamical relation};\\
F_2 &= \textit{field reading};\\
F_3 &= \textit{geometric or curvature reading};\\
F_4 &= \textit{gauge-symmetry or connection reading};\\
F_5 &= \textit{effective interaction under scale and regime};\\
F_6 &= \textit{unresolved boundary or model-interface socket.}
\end{aligned}$$

Principle 9416.8 (Force is not primitive)**.** *In BEDC, force may not enter as a source-free primitive. At the records-side level it can only be discussed as a classifier over witnessed inter-*Hist *change rows:*

$$\mathrm{ForceLikeRelation}_N := \mathrm{Classifier}(\mathrm{InterHistChangeRows}_N).$$

Such a classifier must expose source rows, continuation or measurement witnesses, hsame/psame *stability, ledger policy, and bridge status.*

Remark 9416.9 (Physical content remains bridged)**.** $\mathrm{ForceLikeRelation}_N$ *is not empirical physical force itself. It is a finite horizon records-side carrier. A claim about standard physics requires a separate bridge:*

$$\mathrm{StandardBridge}(\mathrm{ForceLikeRelation}_N, \mathrm{PhysicalForce}).$$

Without that bridge, the relation remains paper-side or schema-side.

9416.4 MetaCIC Self-Audit as a Refinement Chain

MetaCIC self-audit follows the same de-primitiveizing pattern at the formal layer. A mature object, formal rule, type-like object, proof-like object, or classical CIC claim is not accepted as primitive. The audit asks for source, carrier, classifier, exactness, stability, ledger, naming certificate, gate, and socket.

Definition 9416.10 (MetaCIC self-recognition chain)**.** *The* MetaCICSelfRecognitionChain *is the following paper-side refinement schema:*

$$\begin{aligned}
M_0 &= \textit{primitive-looking mature object};\\
M_1 &= \textit{source carrier};\\
M_2 &= \textit{classifier over generated carriers};\\
M_3 &= \textit{exactness and stability obligations};\\
M_4 &= \textit{ledger, GAP, and provenance policy};\\
M_5 &= \textit{NameCert or DerivCert acceptance};\\
M_6 &= \textit{audit gate, failure certificate, or socket.}
\end{aligned}$$

Principle 9416.11 (Formal tokens are de-primitiveized)**.** *MetaCIC's audit function is to lower formal tokens into generated carriers, classifiers, stability obligations, ledgers, and boundaries. Any item that cannot pass through those rows may remain named, but it cannot be treated as internally closed BEDC content.*

9416.5 Force–MetaCIC Relation

Definition 9416.12 (Force–MetaCIC relation certificate)**.** *The* ForceMetaCICRelationCert *is a* LayeredRelationCert *whose source domains are the history of force cognition and the BEDC/MetaCIC history of self-certification. Its layer map sends:*

$$F_0 \mapsto M_0, \quad F_1 \mapsto M_1, \quad F_2 \mapsto M_2, \quad F_3 \mapsto M_3, \quad F_4 \mapsto M_4, \quad F_5 \mapsto M_5, \quad F_6 \mapsto M_6.$$

The preserved structure is primitive-token replacement by relation, classifier, invariant, ledger, certificate, and boundary. The non-preserved content includes empirical physical content, numerical constants, machine-checked proofs, phenomenal essence, object-level T*, and global universe structure. The exactness status is same refinement schema, not same object. The strength is paper certificate.*

Theorem 9416.13 (Force–MetaCIC same-schema certificate)**.** ForceCognitionChain *and* MetaCICSelfRecognition *instantiate the same de-primitiveizing relation schema at paper-certificate strength.*

Sketch. Both chains begin from a primitive-looking token. Both are refined by obstruction: empirical anomaly, classifier pressure, audit failure, or boundary mismatch. Both replace the coarse token by carrier and classifier rows. Both require stability and ledger rows to prevent source erasure. Both end by marking the unclosed residue as a boundary, socket, failure certificate, or bridge obligation. The relation preserves this refinement mode and does not preserve object content. Therefore the comparison closes as same-schema relation rather than identity. □

Corollary 9416.14 (Analogy as certificate)**.** *The sentence that force cognition resembles MetaCIC self-audit may enter the manuscript only as the natural-language reading of Definition 9416.12. It may not enter as an object-level theorem.*

9416.6 Interfaces to Universe, Physics, and Consciousness

The relation certificate composes with the finite records-side universe model of chapter 9420. The closed carrier is not a global universe object. It is a finite multi-Hist ledger-site with local time streams, selector ledgers, digest/GAP rows, witnessed inter-Hist relations, and stable invariants.

Definition 9416.15 (Force-like relation on a finite multi-Hist site)**.** *For a finite multi-*Hist *ledger-site* $\mathcal{W}_N$*, a* ForceLikeRelation$_N$ *consists of:*

1. *source rows recording inter-*Hist *change;*
2. *carrier rows over pairs or finite tuples of* Hist *values;*
3. *a same-force-pattern classifier;*
4. hsame/psame *stability and observer-slot symmetry;*
5. *local measurement, continuation, and causal-dependence ledgers;*

6. *exactness conditions under which classifier sameness reflects source relation;*
7. *an optional standard-physics bridge;*
8. *a status row: seed, paper certificate, checked certificate, bridge certificate, or schema-only.*

Principle 9416.16 (Force as inter-Hist classifier)**.** *Within BEDC, force language must be read through inter-*Hist *relation classifiers. A claim that uses force as a hidden cause, source-free input, or global-field primitive is not a closed BEDC claim; it must be demoted to a boundary, bridge obligation, or refusal.*

Definition 9416.17 (Consciousness relation carrier)**.** *For a finite multi-*Hist *ledger-site* $\mathcal{W}_N$ *with focus index* i*, the consciousness relation carrier is:*

$$\mathrm{ConsciousnessRelationCarrier}(\mathcal{W}_N, i) := (H_i, \mathrm{Trace}(H_i), \mathrm{Endpoint}(H_i), \mathrm{SelectorLedger}(H_i), \mathrm{SelfComparisonRows}($$

This carrier closes only the records-side surface.

Principle 9416.18 (No subject-substance in relation closure)**.** *A consciousness-related relation can close only through* Hist*, trace, endpoint, selector ledger,* GAP*, and inter-*Hist *relation rows. A phenomenal self-center, first-person givenness, or* T *far end may appear as boundary reading, but it may not be consumed as theorem content.*

9416.7 Gate and Refusal Surface

Definition 9416.19 (Layered relation gate)**.** *The* $\mathrm{LayeredRelationGate}(A, B)$ *is the audit question family for a proposed relation certificate:*

1. *Are the two source domains displayed?*
2. *Are the two carrier families displayed?*
3. *Are the two classifier families displayed?*
4. *Is the layer map displayed?*
5. *Are preserved and non-preserved content both displayed?*
6. *Are ledger rows displayed?*
7. *Are exactness and failure boundary displayed?*
8. *Has same-schema language been incorrectly promoted to identity?*
9. *Has the relation smuggled in an empirical law, subject-substance, global frame, or object-level* T*?*
10. *Is the strength row displayed?*

Concretized in §1099: Layered relation gates are realized as finite source, carrier, classifier, preserved, not-preserved, exactness, failure-boundary, refusal, and NameCert rows.

Principle 9416.20 (Refusal is part of closure)**.** *In BEDC, a relation certificate closes partly by refusing its over-readings. Refusal is not an admission of theoretical failure; it is the ledger row that prevents a relation token from pretending to be an object-level theorem.*

Over-reading	**Certificate discipline**
Force cognition and MetaCIC self-audit are the same object	Same refinement schema only
BEDC derives empirical physical force	Requires a standard-physics bridge
T produces force or a physical constant	Boundary reading only
Light speed, gauge field, or curvature is proved here	Subsequent invariant or bridge target only
The consciousness carrier is phenomenal consciousness itself	Records-side carrier only
The self-center is internal theorem content	Boundary reading only
Multi-Hist physics is closed by this chapter	Relation interface only
Analogy substitutes for carrier and ledger rows	Rejected by Definition 9416.19
Paper certificate is checked certificate or bridge certificate	Strength row forbids promotion

9416.8 Closure

Theorem 9416.21 (Closed discussion of layered relations)**.** *Let A and B be two cross-domain themes. If* $\mathrm{LayeredRelationCert}(A, B, s)$ *displays source, carrier, classifier, layer map, ledger, exactness, failure boundary, strength, and refusal rows, and if* $\mathrm{LayeredRelationGate}(A, B)$ *reports no identity collapse, hidden axiom, global-frame leak, subject-substance leak, or empirical overclaim, then the discussion of the relation between A and B is closed at manuscript level with strength s.*

Sketch. The discussion consumes only displayed certificate fields. Source and carrier rows bound the concept layers. Classifier and layer-map rows specify the comparison. Ledger rows preserve provenance. Exactness rows state what form of sameness is available. Failure-boundary and refusal rows prevent over-reading. The strength row prevents a paper certificate from being read as checked or bridged evidence. Therefore the exported relation is neither loose analogy nor object identity, but an auditable relation claim. □

Corollary 9416.22 (Force–MetaCIC relation closes at paper-certificate strength)**.** *Under the fields of Definition 9416.12, the force–MetaCIC relation may be used as a paper-certificate-strength relation. It closes the de-primitiveizing schema and does not close empirical force or the full MetaCIC meta-loop.*

Corollary 9416.23 (Universe and consciousness enter through records-side carriers)**.** *Universe, consciousness, and force language enter this relation certificate only through finite multi-*Hist *ledger-sites, focused* Hist *carriers, inter-*Hist *relations, and* $\mathrm{ForceLikeRelation}_N$*. Claims about phenomenal consciousness,* T*, a global universe object, or empirical constants remain boundary-side unless a separate certificate supplies the required bridge.*

Concretized in §1037: The finite relation-audit frame is concretized as a NameCert packet with source rows, layer map, preserved rows, non-preserved rows, ledger, exactness, and failure boundary.

Concretized in §1092: The analogy-certificate gate is concretized as a finite NameCert packet that admits same-schema language only through source, carrier, classifier, preserved, not-preserved, ledger, exactness, and failure-boundary rows.

Concretized in §1091: Finite layered relation certificates are represented by a displayed carrier, classifier, preservation/refusal boundary, exactness row, and ledger.

The final rule is therefore: do not write analogy as identity. Close the analogy as a relation certificate with source, carrier, classifier, ledger, exactness, and failure boundary.

Concretized in §925: Layered relation certificates are realized as finite NameCert packets separating same-schema relation evidence from object identity, empirical overclaim, and boundary leakage.

Concretized in §1088: Force-like inter-Hist relations are realized as finite relation-certificate packets with source, classifier, layer-map, refusal, ledger, and NameCert rows.

Concretized in §1089: LayeredRelationSiteUp realizes the layered relation certificate as a finite BHist NameCert packet over relation rows, refinement chains, locality gates, and refusal ledgers.

Concretized in §1090: Layered relation certificate packets realize the vision's relation-before-identity discipline as carrier, classifier, refusal, ledger, exactness, and NameCert rows.

Concretized in §7661: Layered relation certificates are realized as finite source, carrier, classifier, preserved, not-preserved, exactness, failure-boundary, refusal, and NameCert rows.

Concretized in §1098: Layered relation failure boundaries are realized as finite NameCert packets for exactness, weakening, refusal, gate-report, transport, route, provenance, and non-escape rows.

Concretized in §7012: Layered relation gate questions are realized as finite source, layer-map, preserved/not-preserved, exactness, failure-boundary, provenance, and NameCert rows.

Concretized in §7584: Finite approximation kernels realize finite source, approximant, classifier, error-ledger, failure-boundary, transport, replay, provenance, and NameCert rows.

Chapter 9417

The Cook Compile-Frontier as Engineering Path: From Principle to Witness

Definition 9390.18 and Definition 9390.19 establish the cook compile-frontier as a numerical reading of the Goedel boundary at the cellular substrate layer. The frontier is currently in the order of 16 productions and 32 tape symbols; the smallest halted Turing machine compiles to approximately 64 productions and 64 tape symbols. The gap between these two numbers is a finite engineering distance: the substrate is Turing-universal in principle (Cook 2004), and the substrate has been exercised on cyclic-tag inputs of substantial scale, but the round-trip from a real Turing machine encoding has not yet been completed. This chapter examines the engineering path from the present frontier to a halted-TM end-to-end witness, identifies the structural obstructions to be discharged at each stage, and reads the path itself as a contribution to the closed-observational framework: making the Goedel boundary not only legible but *progressively measurable*.

9417.1 Setup: present frontier as engineering coordinate

Definition 9417.1 (Engineering coordinate of the frontier)**.** *The* engineering coordinate *of the cook compile-frontier is the triple* (P^*, T^*, S^*) *specifying the smallest production count, tape symbol count, and Rule 110 step count for which the substrate has completed a closed round-trip in the sense of Definition 9390.18. The coordinate is reported by the* `rule110/tests/test_cook_packet_scale.c` *audit binary and is updated whenever a larger configuration passes its round-trip check.*

Example 9417.2 (Present engineering coordinate)**.** *At the present substrate version,* (P^*, T^*, S^*) *includes* $(2, 16, 16384)$*,* $(8, 8, 16384)$*,* $(4, 16, 32768)$*,* $(2, 32, 32768)$*, and* $(16, 2, 8192)$*. The product* $P^* \cdot T^*$ *remains below* 64 *in all currently passing configurations. The halted-TM target coordinate is approximately* $(64, 64, S_{\text{halt}})$ *where* S_{halt} *is the Rule 110 step count required by Cook's polynomial-time bound for the smallest halted machine.*

Principle 9417.3 (The frontier is a moving coordinate, not a static boundary)**.** *The cook compile-frontier is not a fixed structural limit of the substrate; it is the engineering coordinate of where the substrate has been exercised to date. Engineering work that extends the substrate's implementation, optimises its encoding, or scales its evaluator can advance the coordinate without altering any closure theorem. The frontier therefore behaves as a moving coordinate: each successful audit*

binary run at a larger configuration shifts (P^*, T^*, S^*) *to a new value, and the gap to the halted-TM target shrinks correspondingly.*

9417.2 The compile pipeline, stage by stage

Definition 9417.4 (Cook compile pipeline stages)**.** *The compile pipeline implementing Cook 2009* §1.2–§1.4 *decomposes into three sequential stages:*

1. *TM-to-tag: a Turing machine* $(Q, \Sigma, \delta, q_0, q_{\text{halt}})$ *compiles to a tag system* $(\Sigma_{\text{tag}}, \mathsf{prod})$*, implemented in* `rule110/encoder/tm_to_tag.c`*.*
2. *Tag-to-cyclic: the tag system compiles to a cyclic tag system* (P, T)*, implemented in* `rule110/encoder/tag_to_`
3. *Cyclic-to-R110: the cyclic tag system compiles to a Rule 110 initial row, implemented in* `rule110/encoder/cook_encode.c`*.*

Each stage is exercised independently and end-to-end by the test binaries in `rule110/tests/`*.*

Remark 9417.5 (Stage-wise scale frontiers)**.** *Each pipeline stage has its own scale frontier. The TM-to-tag stage's frontier is the largest* $(|Q|, |\Sigma|)$ *pair for which the stage produces a valid tag system. The tag-to-cyclic stage's frontier is the largest* $|\Sigma_{\text{tag}}|$ *for which the stage produces a valid cyclic-tag system at acceptable* (P, T) *size. The cyclic-to-R110 stage's frontier is the largest* (P, T, S) *for which the substrate completes a closed round-trip. The overall engineering coordinate of Definition 9417.1 is the minimum frontier across the three stages.*

9417.3 Structural obstructions to advancing the frontier

Remark 9417.6 (Memory scaling)**.** *The Rule 110 evaluator at* `rule110/evaluator/rule110.c` *represents the cell row as a packed bit array. Doubling the row width doubles the array size; at* $(P, T) \approx (64, 64)$ *the encoded row width is in the order of* 10^6 *cells. Memory scaling at this size is within reach of modern systems but requires explicit attention to allocation strategy. This is the most direct obstruction at the substrate-evaluation stage.*

Remark 9417.7 (Encoding constant factors)**.** *The cyclic-to-R110 encoding of Cook 2009* §1.4 *introduces large constant factors: each cyclic-tag symbol corresponds to several thousand Rule 110 cells; each tag-system step corresponds to many Rule 110 steps. Reducing these constants is an active research direction in cellular substrate engineering, with potential gains from alternative glider catalogs (Definition 9390.3) and tighter collision packing (Martinez 2012 Table 1/Table 2). Each constant-factor reduction advances the engineering coordinate without changing any closure result.*

Remark 9417.8 (Round-trip verification cost)**.** *Verifying a round-trip at* $(64, 64, S_{\text{halt}})$ *requires evaluating Rule 110 for* S_{halt} *steps on a row of width in the order of* 10^6*, then decoding the result back through the inverse of the cyclic-to-R110 encoding, then through the tag-to-cyclic encoding, then through the TM-to-tag encoding. The total verification work scales as* $O(S_{\text{halt}} \cdot W)$ *where* W *is the row width. The wall-clock cost at the target frontier is non-trivial but tractable.*

9417.4 The epistemic value of crossing the frontier

Principle 9417.9 (Crossing converts in-principle universality to witnessed universality)**.** *Cook 2004 proves the substrate is universal in principle. Crossing the engineering frontier from* (P^*, T^*, S^*)

to $(64, 64, S_{\text{halt}})$ *would convert that in-principle universality into a witnessed universality: the substrate would have been exercised on a real Turing-machine encoding, with the round-trip producing the expected halt-state output. The conversion does not strengthen the closure theorem; it strengthens the empirical support for the claim that the substrate behaves as Cook 2004 predicts at the scale of real machines.*

Principle 9417.10 (Witnessed universality is engineering data, not a new theorem)**.** *The conversion of Definition 9417.9 produces an engineering data point: "the smallest halted TM has been compiled to a closed Rule 110 round-trip at coordinate* $(64, 64, S_{\text{halt}})$*, audited by* `test_cook_packet_scale.c` *on commit h". This data point is reportable in* `rule110/STATUS.md` *alongside the present frontier. It is not a new theorem; it is the operational realisation of an existing theorem at a scale where the operation is for the first time exercised on substrate-internal data corresponding to a recognisable computational structure.*

Remark 9417.11 (Comparison with the MetaCIC dependent-codomain barrier)**.** *The frontier-crossing work has a structural analogue at the meta-theory layer: discharging the four subject-reduction hypotheses of chapter 9377 as theorems rather than as* `structure` *fields. Both forms of work advance the project from in-principle closure to operationally-witnessed closure. The cellular frontier-crossing is engineering at the substrate layer; the MetaCIC hypothesis-discharge is engineering at the meta-theory layer. The two layers are logically independent (Definition 9390.20); progress at one layer does not require or imply progress at the other.*

9417.5 Failure modes of the engineering path

Remark 9417.12 (Substrate divergence from Cook prediction)**.** *If at some intermediate coordinate* (P, T, S) *between the present frontier and the halted-TM target the substrate's round-trip fails to decode back to the expected output, the resulting fact is a partial falsification of the operational claim that the substrate implements Cook's construction exactly. The substrate's universality in principle would not be affected (Cook 2004 is a theorem of mathematics); the operational realisation in the present codebase would require revision. A failure of this kind is informative: it identifies the substrate-side commit at which the divergence appears and points the engineering work to a specific bug or specification mismatch.*

Remark 9417.13 (Memory or wall-clock infeasibility)**.** *If memory scaling or wall-clock cost makes the halted-TM coordinate infeasible at acceptable cost on available hardware, the engineering coordinate stops advancing at some intermediate value* (P', T', S') *short of the target. The closure theorems remain valid; the operational claim about "witnessed universality" would have to be stated at the achieved coordinate. The frontier-as-data role (Definition 9417.10) is preserved at any coordinate; the role does not require reaching the halted-TM target.*

9417.6 The frontier as ongoing project artifact

Principle 9417.14 (The frontier value is a citable project state)**.** *The present engineering coordinate* (P^*, T^*, S^*) *is reported in* `rule110/STATUS.md` *and is updated on every substrate scaling improvement. The coordinate is therefore a citable project state: papers, dossier essays, and project reports can cite "the cook compile-frontier at commit h is* (P^*, T^*, S^*)*" as a precise numerical fact. The frontier is updated as a project artifact, not as a one-off measurement.*

Remark 9417.15 (Progress reporting via frontier deltas)**.** *Progress on the engineering path can be reported as deltas: "the frontier advanced from (P_0^*, T_0^*, S_0^*) to (P_1^*, T_1^*, S_1^*) in version v". Each delta is a concrete improvement in operational coverage of the universality claim. Cumulative delta history can be tracked in* `rule110/CHANGELOG.md` *or equivalent; the structural role of the frontier as a moving coordinate (Definition 9417.3) makes such history meaningful.*

9417.7 Mapping to BEDC engineering

Frontier-engineering reading	**BEDC engineering realisation**
Engineering coordinate	(P^*, T^*, S^*) reported by `rule110/tests/test_cook_packet_scale.c` and recorded in `rule110/STATUS.md`.
Halted-TM target	$(64, 64, S_{\text{halt}})$ from the smallest halted Turing machine under Cook 2009 §1.2–§1.3.
Pipeline stages	`tm_to_tag.c`, `tag_to_cyclic.c`, `cook_encode.c` in `rule110/encoder/`.
Memory scaling obstruction	Row width $\sim 10^6$ cells at target; `rule110/evaluator/rule110.c` packed bit array.
Verification cost	$O(S_{\text{halt}} \cdot W)$; tractable on modern hardware.
In-principle vs witnessed universality	Definition 9417.9: the engineering work converts one to the other operationally; closure theorems unaffected.
Independent of MetaCIC barrier	Definition 9417.11: cellular layer and meta-theory layer are logically independent engineering paths.

9417.8 Closure remarks

This chapter has read the cook compile-frontier as an engineering coordinate, identified the three-stage pipeline that the engineering path must advance, and named the structural obstructions (memory scaling, encoding constants, verification cost) the engineering work must address. The chapter does not claim that the halted-TM target is reachable; it identifies the target, the path, and the present coordinate so that progress can be reported as concrete deltas rather than as informal advances.

The epistemic value of crossing the frontier is to convert in-principle universality (Cook 2004 as theorem) into witnessed universality (substrate operationally exercised on a real Turing machine encoding). The conversion does not strengthen any closure theorem; it strengthens the operational support for the claim that the substrate implements Cook's construction correctly at scales matching real computational structures.

The boundary of this chapter is: the engineering frontier is a moving coordinate; the engineering path is concretely specified; the obstructions are named; the epistemic value of crossing is operational rather than theoretical. The chapter does not introduce new closure theorems and does not depend on the engineering path succeeding. It identifies the path so that work along it can be reported as advances in operational coverage of an already-proved universality claim.

Concretized in §639: Compiler optimization endomorphisms realize the Cook compile-frontier route as a finite before/after compiler trace with semantic-preservation and frontier-coordinate rows.

Concretized in §958: HaltedTmRoundTrip realizes the Cook compile-frontier path as a finite staged compile, Rule 110 orbit, decode, and halt-row NameCert packet.

Concretized in §952: CookCompileFrontierWitness packages the Cook compile-frontier coordinate as a finite compile-stage, audit, obstruction, hsame, Cont, Pkg, and NameCert witness.

Concretized in §954: CompileFrontierWitnessUp records the Cook compile-frontier as a finite witnessed-coordinate NameCert with stage routes, audit rows, and nonchoice boundary.

Concretized in §7557: Finite cover uniform modulus envelopes realize a finite cover, modulus, window, transport, and NameCert route for constructive compact-continuous consumers.

Concretized in §949: The Cook compile-frontier coordinate is realized as a finite NameCert packet for stage frontiers, Rule 110 scale rows, audit replay, and failure non-escape.

Concretized in §959: Cook frontier witness packets realize the compile-frontier coordinate as finite stage, target, replay, obstruction, hsame, Cont, Pkg, and NameCert rows.

Concretized in §959: CookFrontierWitnessUp realizes the finite Cook compile-frontier coordinate, staged compiler rows, replay audit, obstruction ledger, and halted-target boundary as NameCert data.

Concretized in §7712: FrontierCompileHardProblemUp realizes the hard-problem, layered-relation, and cook-compile frontier as finite NameCert rows.

Chapter 9418

Observer-Hist Identity: The Implicit Metaphysics of the Finite Kernel

Concretized in §479: Observer-Hist identity is realized as a finite NameCert packet over Hist, hsame, Cont, probe/signature observations, and no-subject consumer ledgers.

Concretized in §615: Observer subject boundaries are realized as finite BHist observation packets whose ledger excludes any extra observer-substance carrier.

Concretized in §640: Observer-kernel traces are realized as finite BHist packets over hsame, Cont, licensed observation rows, provenance, and a no-subject ledger.

Concretized in §688: Observer perspective composition realizes two-step observer comparison as a finite BHist NameCert packet over displayed classifier, coupling, alignment, hsame, Cont, Pkg, and local naming rows.

Concretized in §882: Observer distinction traces are realized as finite BHist packets over Hist distinction steps, Cont routes, hsame transport, and no-subject provenance.

Concretized in §973: Observer boundary selectors realize observer-history identity as finite selected-row, omitted-row, selector-decision, hsame, Cont, Pkg, StreamName, and NameCert data.

BEDC was built for mathematical purposes: axiom-purity over a mathlib-free Lean 4 base, distinction-foundationalism as the only primitive, and a closed inductive kernel as the entire generative apparatus. The construction is mathematical end to end, and its theorems are mathematical theorems. Yet the kernel architecture, when read at the level of its signatures and closure laws rather than at the level of its derived chapters, carries an implicit position about what observation, observer, and time are inside the formal layer. This position became visible during development, not as a design goal, but as a structural consequence of the kernel's signature: the Cont relation has only Hist-valued arguments, the only identity criterion at the kernel is hsame on Hists, and the temporal direction of derivation is fixed by the asymmetry of Cont. The position is the single line: observation = distinction = time, and observer = Hist with no separate subject. This capstone formalizes that line. It introduces no new mathematics. Each section identifies a carrier predicate or closure law in which a specific commitment is encoded and proves a theorem showing the commitment is mathematically forced rather than philosophically added on. Five sections cover, in turn, distinction as the source of information (section 9418.2), observer as the Hist self-reference loop (section 9418.3), time as Hist growth (section 9418.4), the two-loops architecture as the consistency face of the no-subject commitment (section 9418.5), and the synthesis as a three-fold identity at the kernel level (section 9418.6). Like the rest of the manuscript, the chapter falls

under the project's axiom-purity gate: the formal targets it cites do not depend on Classical.choice, Quot.sound, or propext.

9418.1 Observer history identity ledger

Observer history identity records the point where observational sameness and generated-history sameness meet. In BEDC, an observer can inspect a history only through marks, continuations, bundles, signatures, and certified classifiers. Two histories that agree under every licensed observation are therefore not identified by a host equality principle; they are compared through the internal history-sameness and classifier ledgers already present in the kernel.

Concretized in §477: Observer history identity is realized as a finite NameCert packet over BHist, hsame, Cont, licensed observations, and local ledger provenance.

Concretized in §986: Observer hsame classifiers realize the licensed observation-comparison surface as finite BHist, hsame, Cont, ProbeBundle, SigRel, Pkg, and NameCert rows.

Concretized in §1031: Observer licensed comparisons realize observer-history identity as finite licensed-observation, hsame-transport, classifier-output, Cont-route, Pkg, and NameCert rows.

Concretized in §1032: Observer bridge ledgers realize observer-history identity as finite observer-state, no-subject-boundary, multi-history, route, provenance, and NameCert rows.

Concretized in §514: Observer filters realize observer-history identity as a finite BHist selection and ledger boundary.

Concretized in §1042: Observer-filter budget selectors realize observer-history identity as finite selected-row budgets feeding StreamName and Real-completion observation routes.

Concretized in §927: Attention ledgers realize observer-history identity as finite selected-row, omitted-row, StreamName handoff, and non-escape NameCert data.

Concretized in §1047: Observer directions are realized as finite observer-row, target-row, direction-token, comparison, readback, hsame, Cont, Pkg, and NameCert packets.

Definition 9418.1 (Observer history identity)**.** *An observer identity judgment between histories h and k is a certificate-level comparison whose observations are all mediated by BEDC witnesses: mark emission, continuation, probe bundles, signature relations, and derived classifiers. Its intended public reading is that no licensed observer separates h from k without producing a classifier witness against* $\mathsf{hsame}(h, k)$.

Theorem 9418.2 (Observer identity remains internal)**.** *Every observer history identity judgment is internal to BEDC when its observations factor through the kernel ledgers of chapter 3, chapter 6, and the naming-certificate classifier discipline.*

Proof. Each observation listed in the judgment is a BEDC object: a mark emission, a continuation, a bundle event, a signature relation, or a certified classifier comparison. Repack these witnesses as the observer ledger. The resulting comparison uses only the internal sameness and classifier relations named in the premise, so no external equality principle is introduced. □

9418.2 Distinction as the source of information

Definition 9418.3 (Information content of a history)**.** *For a Hist value h, the* information content *of h is the sequence of* E1/E0 *distinctions recorded in h, given by an explicit unary recursion on the inductive structure of* Hist*. The recursion case at* e *is the empty sequence; the recursion case at each* E1 *or* E0 *application appends the corresponding distinction tag to the sequence produced from the predecessor.*

Theorem 9418.4 (Pre-distinction void)**.** *The empty Hist value* e *has trivial information content: every Hist-predicate P whose definition recurses only on* E1 *and* E0 *constructors and not on auxiliary parameters evaluates to its trivial case at* e*. In particular, the information-content sequence at* e *is the empty sequence.*

Theorem 9418.5 (Hist as minimal information carrier)**.** *Every constructive carrier of binary-distinction information expressible in the BEDC layer embeds into* Hist *via an explicit isomorphism on its constructor signature;* Hist *is minimal in the sense that any further-reduced carrier loses the binary distinction. The embedding is given by the generator-closure classifier of the carrier's signature.*

Corollary 9418.6 (Information requires distinction)**.** *There is no constructive information carrier in BEDC strictly smaller than* Hist*; equivalently, the empty Hist value carries no information, and any non-trivial information content requires at least one* E1 *or* E0 *application above* e*.*

Remark 9418.7 (Reading)**.** *This section formalizes the project-governance principle that distinction precedes information. The principle is not assumed at the level of the kernel; it is forced by the closed-inductive structure of* Hist *(chapter 3). The empty value* e *is the only Hist that records nothing, and every other Hist is constructed by finitely many distinction acts above it. There is no Hist outside this generation, and therefore no information-carrier inside the BEDC layer that bypasses the form of distinction.*

9418.3 Observer as Hist self-reference loop

Definition 9418.8 (Subject-free observation)**.** *The continuation relation* $\mathsf{Cont}(h, d, h')$ *is* subject-free *when its signature takes only Hist-valued arguments, with no auxiliary parameter outside the* Hist *type appearing in the relation. The Hist value h plays simultaneously the role of antecedent state and the role of the carrier on which the continuation is recorded; no observer-substance occupies a separate slot.*

Definition 9418.9 (Self-reference closure)**.** *The kernel is* self-reference closed *when (i) every Cont continuation is a relation between Hist values alone, with no agent parameter outside* Hist*, and (ii)* hsame *on Hist values is the only identity criterion the kernel admits, with no observer-substance identity defined alongside.*

Theorem 9418.10 (Hsame is observer identity)**.** *Two Hist values h and h' satisfy* $\mathsf{hsame}(h, h')$ *if and only if they record the same finite sequence of* E1/E0 *distinctions. Under the identification of observer with history of self-distinguishing, two observers are the same observer exactly when their distinction traces are* hsame*-equivalent.*

Theorem 9418.11 (No subject parameter)**.** *The kernel-level signature of the Cont relation has no auxiliary parameter representing a subject or carrier outside* Hist*. Adding such a parameter*

requires a new type with non-trivial closure laws relative to the existing cont_assoc, cont_left_unit, *and* cont_right_unit *identities; introducing it forces inconsistency with at least one of these three identities, since the parameter has no place in their formulation.*

Corollary 9418.12 (Observer is identical to Hist)**.** *Up to* hsame, *the observer is the Hist value itself; there is no observer-substance in the BEDC layer distinct from the recorded distinction trace. Every kernel-level statement about observers is, by Definition 9418.11, a statement about Hist values under* hsame.

Remark 9418.13 (Reading)**.** *This section formalizes a constructive realization of the bundle theory of self in the empiricist tradition, of ātman-negation in the analytical traditions that share its conclusion, and of the second-order cybernetics position that the observer is a configuration of its own observations rather than a separate substance. BEDC is not committed to any of these traditions philosophically; the framework is committed to their structural conclusion mathematically, in that the kernel signature has no slot for observer-substance and adding one breaks the closure laws.*

9418.4 Time as Hist growth

Definition 9418.14 (Hist time step)**.** *A* Hist time step *is a single application of* E1 *or* E0 *to a Hist value. The step is recorded simultaneously as part of the Hist value, by the corresponding constructor, and as part of the witness in the Cont relation, by the d argument tagging which of* E1 *or* E0 *was applied.*

Definition 9418.15 (Local time stream)**.** *For a Hist value h, the* local time stream *of h is the sequence of Cont applications producing h from* e*; equivalently, the unary length* $\mathsf{UnaryHistory}(h)$ *together with the distinction record at each step recovered from the information content of h (Definition 9418.3).*

Concretized in §603: Local Hist time streams are realized as finite BHist packets over Cont-step rows, hsame transport, and a no-global-clock ledger.

Concretized in §665: Local clock budgets realize observer local time as finite BHist packets over Cont-step windows, hsame transport, and a no-global-clock ledger.

Concretized in §1043: Observer time steps are realized as finite Hist, distinction-tag, Cont-step, hsame-transport, no-global-clock, Pkg, and NameCert rows.

Concretized in §1105: Inter-Hist synchronization refusals realize the no-global-clock observer-history boundary as finite two-stream BHist, hsame, Cont, Pkg, refusal-ledger, and NameCert rows.

Concretized in §7312: Observer modulus clocks realize the observer-history modulus-as-temporal-rate route through finite StreamName, CauchyModulus, RegSeqRat, RealUp, and observer-selection rows.

Theorem 9418.16 (Cont is time-asymmetric)**.** *The Cont relation is functional in the forward direction: given a Hist h and a distinction d, the continuation h' such that* $\mathsf{Cont}(h, d, h')$ *is uniquely determined. The inverse direction, given h' and d recover h such that* $\mathsf{Cont}(h, d, h')$*, is not constructively decidable for a fixed d without recursion on the structure of h'. The asymmetry between forward and inverse direction is the constructive arrow of time at the kernel level.*

Theorem 9418.17 (Modulus is temporal rate)**.** *For every regular convergence carrier in BEDC equipped with a modulus function N : Unary $\to$ Unary, the modulus is the temporal rate of the associated Hist time stream: it specifies, as a unary count of Cont applications, how many distinction steps are required for the trace to stabilize within a given precision. The same correspondence applies to regular Cauchy carriers and to continuous-modulus carriers.*

Corollary 9418.18 (No global clock in the kernel)**.** *There is no Hist value at the kernel level that synchronizes the local time streams of two distinct Hists; each Hist carries its own local time stream, and inter-Hist time-comparison is not defined at the kernel. Any synchronization, if introduced, must be defined as an extension carrier above the kernel, not as kernel-level structure.*

Remark 9418.19 (Reading)**.** *This section formalizes the Brouwer position that mathematics arises from the perception of time, where the elementary perception is the binary distinction giving the two-ity.* E1 *and* E0 *application is the elementary time-step, Hist growth is the elementary time-stream, and the corpus of constructive theorems built above the kernel is this elementary perception's structured unfolding. The reading does not commit BEDC to Brouwer's metaphysics; it identifies, inside the formal layer, the site at which his structural claim becomes a theorem.*

9418.5 No subject: the two-loops architecture

Definition 9418.20 (Ground loop closed)**.** *The* ground loop *is closed when (i) every Hist value is reachable from* e *by finitely many* E1/E0 *applications, that is,* Hist *is well-founded, and (ii)* hsame *is decidable on* Hist*. Both conditions are theorems of the kernel.*

Definition 9418.21 (Meta loop unclosable)**.** *The* meta loop *is the predicate "what is observation as such?" attempted as a Hist-valued construction. Positing a Hist value whose distinction content represents this predicate generates a Tarski-style fixed-point requirement that no closed inductive structure satisfies. The meta loop is therefore unclosable inside the kernel.*

Theorem 9418.22 (Axiom-purity requires no subject)**.** *The BEDC axiom-purity gate, that no proof depends on* Classical.choice*,* Quot.sound*, or* propext*, is satisfiable on the kernel if and only if no observer-substance type is introduced. Introducing such a type forces at least one of choice, quotient soundness, or propositional extensionality into the proof of* cont_assoc *or one of the related identities, since the new parameter must be transported across continuations and the transport is unwitnessed at the kernel level.*

Theorem 9418.23 (Two-loops balance)**.** *The kernel is consistent if and only if the ground loop closes (Definition 9418.20) and the meta loop does not close (Definition 9418.21). Equivalently, consistency holds if and only if the observer-as-Hist identification (Definition 9418.12) holds and no observer-substance carrier is introduced.*

Remark 9418.24 (Reading)**.** *The unclosability of the meta loop is the constructive face of Tarski's undefinability theorem applied to the BEDC kernel: positing an internal predicate "what observation is" generates a self-referential fixed-point that cannot live in the closed inductive fragment. The no-subject commitment of the BEDC kernel is therefore not a philosophical preference but a structural requirement: the kernel is consistent precisely because it does not attempt to internalize the predicate, and the corresponding closure is the project's axiom-purity gate.*

9418.6 Synthesis: the three-fold identity

Theorem 9418.25 (Observation-distinction-time identity)**.** *At the kernel level, the following three relations are extensionally identical modulo* hsame*:*

1. *the observation act-shape* Cont *on* Hist*,*
2. *the elementary distinction step given by application of* E1 *or* E0*,*
3. *the elementary time-step on the Hist time stream (Definition 9418.14).*

The three relations have the same domain, the same closure laws cont_assoc*,* cont_left_unit*,* cont_right_unit*, and the same* hsame*-respect properties.*

Corollary 9418.26 (Cont is the universal verb)**.** *Every extension-type phenomenon in BEDC is a specialization of the same Cont relation under different closure conditions. Category morphism (chapter 9363), continuous modulus, the compactness chain, small-step computation (chapter 9365), type checking by classifier membership (chapter 9367), and complex differentiability are not merely analogous instances of a shared schema; they are the same kernel-level relation under different naming-certificate closures.*

Remark 9418.27 (Closing: the implicit metaphysics)**.** *This capstone introduced no new theorems about specific mathematical objects. It re-read the existing kernel architecture under one identification: observer =* Hist*, observation =* Cont*, time = Hist growth, with no observer-substance behind the trace. Under that identification, every prior chapter of BEDC is a structured unfolding of self-reference, and the universality of* Cont *across mathematical domains is forced by the no-subject commitment rather than designed in. The chapter falls under the project's axiom-purity gate; no* Classical.choice*, no* Quot.sound*, and no* propext *is required for any of the cited targets.*

Concretized in §888: Observer trace kernels realize observer-history identity as finite BHist, hsame, Cont, observation, provenance, and no-subject NameCert rows.

Concretized in §894: Observer-frame refusals realize no-global-frame and no-synchronized-clock observer rows as finite BHist, hsame, Cont, Pkg, refusal-ledger, and NameCert data.

Concretized in §928: Observer interfaces realize observer-history identity as finite filter, attention-ledger, StreamName handoff, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1054: Observer trace seals realize observer-history identity as finite observer-row, trace-row, seal-row, no-subject refusal, hsame, Cont, Pkg, and NameCert data.

Chapter 9419

Inter-Hist Locality: From No God's-Eye View to Light-Speed Constancy

This capstone is the road-map sibling to chapter 9418. Where that chapter formalizes the SINGLE-Hist commitments that BEDC already proves — namely that an observer is identified with the Hist of its observations, with no observer-substance carrier behind the record — the present chapter scaffolds the MULTI-Hist consequences that BEDC must develop next. When several independent Hist values are considered together, with no shared base and no shared time-step indexing, the kernel does not give a global frame and cannot give one without violating the closed-inductive structure of Hist. In that setting only inter-Hist invariants can carry frame-independent physical content, and the maximum cross-Hist causal rate, in particular, must be observer-symmetric and constant across all observers in a configuration. That last consequence is the structural skeleton of the constancy of light speed in special relativity. The chapter catalogues the structure: multi-Hist configuration, no global synchronization, inter-Hist relation, observer-symmetric invariant, maximum cross-Hist causal rate, and the observer-independence of that maximum. Most of the multi-Hist semantics is currently a SCHEMA-ONLY HORIZON in BEDC: the type signatures and statement targets are sketched here, and the closed witnesses are deferred to a future BEDC physics-of-multi-observer development that does not yet exist. The chapter is therefore a roadmap, not a proof; it identifies the interface that future BEDC chapters would close. The structural commitments of the capstone connect to existing relational physics traditions: Rovelli's relational quantum mechanics, Wheeler's "It from Bit", Connes's noncommutative geometry, and Penrose's spin networks all locate physical content in observer-relations rather than in a privileged global frame. BEDC differs from these traditions in that it constructs the relational substrate from a still more primitive layer: distinction (Hist) and observation (Cont). The chapter falls under the same axiom-purity gate as the rest of the project — whether multi-Hist semantics can preserve no-Classical.choice / no-Quot.sound / no-propext is itself part of the road-map question, and is articulated as such in the closing section.

9419.1 Ledger-local relations

Inter-history locality names the BEDC principle that any comparison between two histories must be witnessed through finite continuation data visible inside the kernel. A relation between histories is

admissible only when its evidence can be read from generated marks, emitted histories, continuation witnesses, or classifiers built from those objects. No external equality oracle is part of the public surface.

Definition 9419.1 (Inter-history locality)**.** *An inter-history relation $R(h, k)$ is local when every positive witness of $R(h, k)$ factors through a finite BEDC ledger: a source history for h, a source history for k, a finite continuation or bundle witness connecting the inspected data, and a classifier comparison stated with* hsame*,* msame*, or a derived classifier already licensed by a naming certificate.*

Theorem 9419.2 (Locality is ledger visibility)**.** *If an inter-history relation is local, then its public use can be audited by the same ledger discipline used for naming certificates: the hidden data are exactly the generating histories and continuation witnesses, and the visible judgment is the classifier-level relation they expose.*

Proof. Unfold locality. The witness already consists of source histories, finite continuation or bundle data, and a classifier comparison. These are precisely the ledger fields available to a naming certificate, so the public relation is the visible projection of that finite BEDC ledger. □

9419.2 Multi-Hist configurations

Definition 9419.3 (Multi-Hist configuration)**.** *A multi-Hist configuration is a finite collection of Hist values $h_1, \ldots, h_n$ considered together, with no requirement of a shared base history and no requirement of a shared time-step indexing. The configuration is the bare collection: the kernel does not append, glue, or synchronize the members.*

Definition 9419.4 (No global synchronization)**.** *There is no Hist-valued global synchronization for multi-Hist configurations: there is no* Hist *value h_g such that for every multi-Hist configuration, h_g records a privileged shared "now" across all members of the configuration. The closed-inductive structure of* Hist *does not admit such an h_g.*

Theorem 9419.5 (Hists are frame-independent)**.** *For any two* Hist *values h_a, h_b, neither carries the other's time-step indexing intrinsically. Any cross-Hist temporal correlation must be constructed via an external relation, not read off either Hist alone, and any such relation is itself a Hist-level construction subject to the same no-global-frame constraint.*

Lemma 9419.6 (Empty configuration is degenerate)**.** *A multi-Hist configuration with zero or one members carries no inter-Hist content. The smallest non-degenerate configuration has two members, and the smallest configuration in which cross-Hist causal dependence (Definition 9419.16) can be non-trivial likewise has two members.*

Lemma 9419.7 (Configuration extension preserves no-global-sync)**.** *Adding a new* Hist *value to an existing multi-Hist configuration does not produce a global synchronization that the original configuration lacked. No-global-synchronization is monotone: it is preserved under configuration extension, and cannot be acquired by accumulating Hists.*

Remark 9419.8 (Reading: no observer-of-observers)**.** *This section formalizes the no-God's-eye-view commitment. There is no observer-of-observers that synchronizes the multi-Hist totality, because the kernel is silent about a global frame and the closed-inductive* Hist *type does not admit one. By the sibling capstone (chapter 9418), once observer is identified with Hist, no super-observer can coexist: any candidate super-observer is itself a Hist, and is subject to the same*

locality constraints as every other member of a configuration. Definition 9419.7 shows that the absence cannot be repaired from inside the kernel by adding more observers; it is a feature of the closed-inductive structure of Hist *rather than a contingent fact about the current configuration.*

9419.3 Inter-Hist invariants

Definition 9419.9 (Inter-Hist relation)**.** *An inter-Hist relation is a predicate $R(h_a, h_b)$ on pairs of* Hist *values that is observer-symmetric: $R(h_a, h_b)$ holds if and only if $R(h_b, h_a)$ holds, up to* hsame *on each component. Symmetry is required at the type-signature level: the predicate cannot privilege one slot over the other.*

Definition 9419.10 (Inter-Hist invariant)**.** *An inter-Hist invariant is a function $I : \mathsf{Hist} \times \mathsf{Hist} \to \mathsf{Unary}$ that is observer-symmetric and frame-independent: for any pair of perspectives on the configuration — that is, for either* Hist *taking the role of measurement-anchor — the function I produces the same output up to* hsame*, and the output does not depend on any external choice of frame.*

Theorem 9419.11 (Invariants are observer-independent)**.** *Every inter-Hist invariant produces the same value regardless of which* Hist *is taken as the perspective anchor. The value is a property of the pair as such, and not of either individual* Hist*; it is not measurable from inside one observer's record except as a relational quantity involving the other.*

Proposition 9419.12 (Invariants compose under hsame)**.** *If I is an inter-Hist invariant and $h_a \mathsf{hsame} h'_a$ and $h_b \mathsf{hsame} h'_b$, then $I(h_a, h_b) \mathsf{hsame} I(h'_a, h'_b)$. The invariant respects the kernel's history-equivalence relation; it cannot distinguish histories that the kernel does not distinguish.*

Lemma 9419.13 (Trivial invariants exist)**.** *The constant function $I(h_a, h_b) := \mathsf{e}$ and the constant function $I(h_a, h_b) := \mathsf{UnaryHistory}$ are inter-Hist invariants in the sense of Definition 9419.10. The class of inter-Hist invariants is therefore non-empty before any physics is committed; the substantive content of the road-map is the construction of non-trivial invariants whose values discriminate between configurations.*

Proposition 9419.14 (psame-stability of invariants)**.** *If I is an inter-Hist invariant and $p_a \mathsf{psame} p'_a$ is a probe-equivalence at the anchor of h_a and $p_b \mathsf{psame} p'_b$ at the anchor of h_b, the invariant I returns* hsame*-equivalent values on the corresponding pairs. Inter-Hist invariants are stable under both kernel-level history equivalence (Definition 9419.12) and kernel-level probe equivalence; they cannot distinguish what the kernel does not distinguish at either layer.*

Remark 9419.15 (Reading: relational content is forced)**.** *This section formalizes the operationalist principle that, in the absence of a privileged frame, all physical content lives in inter-observer relations rather than in any one observer's record. The principle is forced by no-global-synchronization (Definition 9419.4); it is not assumed. In particular, an observer's individual Hist is not a frame-independent carrier of physical content on its own; only I-images of pairs are. Definition 9419.14 sharpens this: invariants cannot smuggle in distinctions below the kernel's resolution, on either the history or the probe layer, so the relational content lives strictly in the kernel-level equivalence quotient.*

9419.4 Cross-Hist causal dependence (schema-only)

Definition 9419.16 (Cross-Hist causal dependence — SCHEMA-ONLY)**.** *A cross-Hist causal dependence is a relation $C(h_a, h_b, k)$ saying "h_b's most recent k* Cont *steps depend on h_a's recent*

steps". The relation is currently a SCHEMA-ONLY HORIZON in BEDC: the kernel exposes the type signature, but no closed witness produces a non-trivial cross-Hist causal relation. Future BEDC chapters on multi-Hist physics would close this schema by giving a constructive predicate together with concrete witnesses.

Definition 9419.17 (Maximum cross-Hist causal rate — SCHEMA-ONLY)**.** *The maximum cross-Hist causal rate is a unary-valued function* MaxRate *on multi-Hist configurations such that, for every cross-Hist causal dependence $C(h_a, h_b, k)$, the ratio of cross-affected* Cont *steps in h_b to corresponding* Cont *steps in h_a is bounded above by* MaxRate*. SCHEMA-ONLY: the upper-bound theorem awaits the multi-Hist semantics that gives C its closed witnesses.*

Theorem 9419.18 (Maximum rate is an inter-Hist invariant — SCHEMA-ONLY)**.** *The maximum cross-Hist causal rate, when defined, is an inter-Hist invariant in the sense of Definition 9419.10: its value is observer-symmetric and frame-independent. SCHEMA-ONLY: this theorem is part of the road-map; its closed proof requires the multi-Hist causal-dependence semantics to be made constructive.*

Theorem 9419.19 (Maximum rate is an observer-independent constant — SCHEMA-ONLY)**.** *For any multi-Hist configuration with a non-trivial cross-Hist causal dependence, the maximum cross-Hist causal rate is the same constant for all observers in the configuration. Equivalently, no observer can measure a different value for the maximum rate without invoking a privileged frame; and no privileged frame exists, by Definition 9419.4.*

Corollary 9419.20 (Light-speed-like constant)**.** *A multi-Hist configuration that admits a non-trivial cross-Hist causal dependence has a single observer-independent maximum-rate constant attached to it. The form of the result — "a constant maximum cross-Hist causal rate exists and is observer-independent" — is the structural skeleton of the constancy of light speed in special relativity. The empirical specific value of the constant is not derived inside BEDC; only the form is.*

Lemma 9419.21 (Causal dependence is asymmetric in the time slot — SCHEMA-ONLY)**.** *The relation $C(h_a, h_b, k)$ of Definition 9419.16 is asymmetric in the time slot k: $C(h_a, h_b, k)$ and $C(h_b, h_a, k)$ are distinct predicates in general, even though the inter-Hist invariants of section 9419.3 are observer-symmetric. The asymmetry is the content of "h_b depends on h_a" as opposed to "h_a depends on h_b", and is exactly the structure that the maximum-rate constant must average over to recover observer symmetry.*

Lemma 9419.22 (Causal dependence respects hsame — SCHEMA-ONLY)**.** *If $C(h_a, h_b, k)$ and $h_a \mathsf{hsame} h'_a$ and $h_b \mathsf{hsame} h'_b$, then $C(h'_a, h'_b, k)$. The cross-Hist causal-dependence predicate is required to be* hsame*-stable; without this stability, no inter-Hist invariant could be defined on top of it without leaking kernel-undistinguishable distinctions.*

Proposition 9419.23 (Triviality at degenerate configurations — SCHEMA-ONLY)**.** *For a multi-Hist configuration with no non-trivial cross-Hist causal dependence, the maximum cross-Hist causal rate is the trivial bound, and Definition 9419.19 reduces to a vacuous truth. The non-trivial content of the maximum-rate constant arises only once C has at least one closed witness on the configuration; until then, the road-map cannot distinguish the constant numerically.*

Remark 9419.24 (Reading: the structural argument)**.** *The structural argument runs as follows. No-God's-eye view (Definition 9419.4) forces all physical content into inter-Hist invariants (Definition 9419.11); the maximum cross-Hist causal rate is one such invariant (Definition 9419.17);*

the symmetry of "no privileged frame" forces this rate to be the same constant for all observers (Definition 9419.19). Each step is inevitability under the previous, not a separate postulate. In standard relativistic terms: BEDC derives the form of an invariant maximum signal speed from no-global-frame plus relational content; what it does not derive is that this constant takes any particular numerical value.

9419.5 Roadmap toward physics

Remark 9419.25 (Form of the next dependency chain)**.** *The capstone identifies the next BEDC roadmap, parallel in form to the Riemann Hypothesis dependency chain (chapter 9239). It is a sequence of multi-Hist namecert chapters that close the schema-only horizons of section 9419.4. The chain proceeds: (i)* NameCert *for* MultiHistConfig*; (ii)* NameCert *for* CrossHistCausal*; (iii)* NameCert *for* MaxCausalRate*; (iv) Lorentz-geometry* NameCert*, in which the inter-Hist invariant produces a Minkowski-like distance from the observer-symmetric maximum rate; (v) curved spacetime as a Lorentz-*NameCert *with non-trivial gluing data over a multi-Hist configuration; (vi) quantum-state* NameCert *as multi-Hist superposition, in which the relational content is carried by* E1/E0 *refinements rather than by metric data; and so on. Each step on this chain is its own chapter and its own statement-only Lean target.*

Concretized in §468: Finite multi-Hist packets realize the first closed carrier for the observer-locality chain from configurations to cross-Hist causal interfaces.

Concretized in §478: Cross-Hist causal dependence is realized as a finite two-history NameCert seed with displayed hsame, psame, Cont, package, and local naming rows.

Concretized in §594: Maximum causal rate is realized as a finite unary-bound NameCert over MultiHistConfig and CrossHistCausal witness rows, without a global frame or metric geometry.

Concretized in §617: Lorentz frame-rate packets realize the next Inter-Hist Locality step as a finite maximum-rate boundary without global frames, clocks, metric tensors, or host real-valued speed.

Concretized in §655: Multi-Hist superposition packets realize the Eone/Ezero refinement step of the Inter-Hist Locality chain as finite BHist, hsame, psame, Cont, Pkg, and NameCert rows.

Concretized in §656: Inter-Hist invariants are realized as finite BHist, hsame, Cont, Pkg, and NameCert packets mediating multi-history configurations, causal witnesses, and maximum-rate consumers.

Concretized in §668: Observation coupling realizes inter-Hist locality as a finite two-history NameCert packet over paired BHist rows, hsame transports, Cont routes, symmetric coupling ledgers, and no-global-frame non-escape.

Concretized in §759: Inter-Hist locality ledgers are realized as finite BHist packets over two source histories, symmetric hsame transport, Cont routes, observer-locality cells, inter-Hist invariant rows, Pkg provenance, and NameCert.

Concretized in §875: Cross-Hist causal routes are realized as finite observer-slot, causal-row, maximum-rate, observer-gate, continuation-trace, transport, provenance, and NameCert packets.

Concretized in §876: Cross-Hist causal rates are realized as finite BHist packets for paired histories, causal-dependence routes, maximum-rate rows, observer-symmetry, and no-global-frame ledgers.

Concretized in §889: Observer-locality transport is realized as a finite BHist NameCert packet over observer-locality cells, CrossHistCausal rows, MaxCausalRate rows, hsame/psame transport, Cont routes, and no-global-frame ledgers.

Concretized in §1011: Observer-locality pullbacks realize the common finite packet joining observer-locality cell, cross-Hist causal, maximum-rate, and transport rows.

Concretized in §895: Multi-Hist couplings are realized as finite paired-history NameCert packets with symmetric coupling, hsame/psame transport, Cont routes, and no-global-frame non-escape.

Concretized in §911: Minkowski rate geometries realize the Inter-Hist Locality rate-to-geometry step as a finite NameCert packet over multi-Hist, causal, maximum-rate, Lorentz-frame-rate, and symmetric distance-shape rows.

Concretized in §1003: No-global-synchronization boundaries are realized as finite multi-Hist refusal packets over displayed histories, locality rows, hsame transport, Cont routes, package provenance, and NameCert data.

Concretized in §1013: Inter-Hist symmetry budgets realize two-direction locality, invariant read-back, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Concretized in §1023: Inter-Hist rate boundaries are realized as finite BHist packets binding causal route, maximum-rate gate, observer symmetry, no-global-frame refusal, and local NameCert rows.

Concretized in §1037: Relational frame audits realize the Inter-Hist Locality no-global-frame and observer-symmetry checks as finite BHist, hsame, Cont, refusal-ledger, and NameCert rows.

Concretized in §1106: No-global-synchronization ledgers realize the inter-Hist refusal of shared clocks and global frames as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1112: Cross-Hist causal cones realize bounded two-history causal-dependence windows with maximum-rate and no-global-frame ledgers.

Concretized in §1194: Finite Hist locality packets realize Inter-Hist Locality as a two-history finite NameCert carrier with locality witness, symmetry, transport, Cont route, and no-global-sync refusal.

Concretized in §1201: Inter-Hist transport seals realize paired-history locality transport as a finite packet over observer-locality cells, invariant rows, cross-Hist causal routes, hsame/psame transport, Cont replay, Pkg provenance, and local NameCert rows.

Concretized in §5193: Observer locality rate budgets realize the no-global-frame maximum-rate read as finite paired-Hist, symmetry, causal-rate, transport, and NameCert rows.

Concretized in §7870: Curved spacetime is realized as finite multi-Hist chart, transition, Lorentz-rate, curvature, transport, replay, provenance, and NameCert rows.

Remark 9419.26 (Falsification surfaces along the chain)**.** *Each step of Definition 9419.25 carries its own falsification surface. Step (i) fails if no closed witness for* MultiHistConfig *can be produced without smuggling in a global frame. Step (ii) fails if cross-Hist causal dependence cannot be expressed as an* hsame- *and* psame-*stable predicate. Step (iii) fails if the maximum-rate function cannot be defined without choice. Step (iv) fails if the candidate Minkowski-like distance is not symmetric in the two observers, or if it does not respect Definition 9419.12. Step (v) fails if the gluing data over the configuration cannot be made constructive. Step (vi) fails if the multi-Hist superposition predicate forces* propext*. The road-map's intermediate failures are themselves informative; they are not engineering setbacks but structural results about where distinction-foundationalism stops being sufficient.*

Remark 9419.27 (Relation to existing single-Hist Lean targets)**.** *The Lean targets named in Definition 9419.3 through Definition 9419.23 are statement-only and do not yet have closed proofs. They are nevertheless not isolated from the rest of the formalization.* InterHistInvariant *(Definition 9419.10) takes* Hist*-pairs to* Unary*, and* Unary *is already a closed-inductive carrier in BEDC;* MaxCausalRate *(Definition 9419.17) is unary-valued in the same sense. The interface uses no Lean fragment that BEDC has not already committed to. What it adds is the multi-Hist combinator structure: pairing two* Hist *values into a configuration, and predicating over pairs.*

Remark 9419.28 (Connection to relational physics traditions)**.** *The structural commitments of this capstone are the same commitments as Rovelli's relational quantum mechanics, Wheeler's "It from Bit", Connes's noncommutative geometry, and Penrose's spin networks: physical content is observer-relational, not located in any single observer's record, and observer is not a substance prior to its observations. BEDC differs from these traditions in that it constructs the relational substrate from a still more primitive layer: distinction (*Hist*) and observation (*Cont*), with no observer-substance carrier behind either, and with everything closed-inductive. Where those traditions take observer-relations as physical primitives, BEDC derives them from the structure of distinction.*

Remark 9419.29 (Schema-only horizons and axiom-purity)**.** *A central open question on this roadmap is whether the multi-Hist semantics can be closed inside BEDC's axiom-purity gate. If yes, BEDC's distinction-foundationalist position extends from pure mathematics into physics with no additional axiomatic cost: the same fragment that proves the single-Hist commitments of chapter 9418 would also prove the multi-Hist commitments of this chapter. If no, the boundary point at which axiom-purity fails would itself be a structurally meaningful result: it would mark precisely where physics requires more than distinction — whether choice, propositional extensionality, or quotient sound is the first to be needed, and on which specific schema. Either outcome is informative.*

Remark 9419.30 (Why the chain is acyclic)**.** *The dependency chain of Definition 9419.25 is acyclic by construction. Step (i)* MultiHistConfig *does not depend on cross-Hist causal dependence, only on* Hist*; step (ii) builds* CrossHistCausal *on top of (i); step (iii) extracts a single unary upper bound from (ii); steps (iv) and (v) elaborate that bound into geometric data; step (vi) lifts the same machinery to a* NameCert *over multi-Hist superpositions. Each step strictly extends the predicate vocabulary of the previous step. The order in which the multi-Hist semantics must be assembled is exactly the order of the chain; no step can be deferred past a later step without circularity.*

Concretized in §474: The multi-history configuration carrier supplies the first concrete NameCert step for the Inter-Hist Locality chain.

Remark 9419.31 (E1/E0 refinements at the multi-Hist level)**.** *The single-Hist semantics distinguishes* E1 *(extensional, observation-driven) from* E0 *(intensional, name-driven) at the level of one observer's record. The multi-Hist roadmap inherits this distinction: an inter-Hist invariant may be sensitive to* E1*-level refinements of the two component records, to* E0*-level refinements, or to both. Quantum-state* NameCert*s in step (vi) of Definition 9419.25 are expected to live in the* E1*-sensitive class, and metric* NameCert*s in steps (iv) and (v) in the* E0*-sensitive class. Articulating which refinement layer carries which physical content is part of the multi-Hist development.*

Remark 9419.32 (Single-Hist obligations as preconditions)**.** *The roadmap of this capstone presupposes the single-Hist obligations articulated in the sibling capstone (chapter 9418). Without observer-Hist identity, the multi-Hist configuration of Definition 9419.3 would not yet be a configuration of observers; without* hsame*-stability of single-observer content, Definition 9419.12 would*

not type-check at the inter-observer level. The two capstones close adjacent layers of the same problem.

Concretized in §474: Multi-Hist configurations are realized as finite BHist-ledger NameCert packets for independent histories without a global synchronization row.

Concretized in §933: Max-rate read gates realize the Inter-Hist Locality maximum-rate boundary as a finite NameCert packet requiring observer-symmetry and no-global-frame rows before any rate read is public.

Concretized in §6335: The finite inter-Hist causal-rate comparison packet realizes the local comparison surface without a synchronized frame.

9419.6 Closing

Remark 9419.33 (What this capstone is and is not)**.** *This capstone is not a derivation of relativity, of quantum mechanics, or of any particular physical theory. It is a derivation of the form of an observer-independent maximum-rate constant from the combination of distinction-foundationalist commitments (single-Hist identification of observer with record, no observer-substance carrier) and the closed-inductive structure of* Hist*. The form is the strong claim; the value is not derived. Equally, the capstone is not a claim that BEDC will succeed in closing the multi-Hist roadmap inside its axiom-purity gate. It is a claim that the question is well-posed and that the interface is now sketched precisely enough that progress and failure can be told apart.*

Remark 9419.34 (The structure of physics from observer locality)**.** *The capstone makes one structural argument. In a configuration of independent Hists with no global frame, all physical content lives in observer-symmetric inter-Hist invariants, and the maximum cross-Hist causal rate must be observer-independent. This is the BEDC-derived skeleton of the relativistic structure of physics. The current chapter does not produce that physics; it identifies the interface that a future BEDC physics chain would close. Whether closure is possible inside the axiom-purity gate is the next research question. The chapter falls under the project's axiom-purity gate, and the road-map question of whether physics can be constructively closed inside that gate is itself open. Definition 9419.26 catalogues the specific surfaces along which the road-map can fail informatively; the answer to the road-map question will be assembled from a sequence of step-by-step constructive successes or step-localized failures, not declared in advance.*

Lemma 9419.35 (Zero max-rate forbids cross-Hist dependence — SCHEMA-ONLY)**.** *Assume two histories h_a, h_b and slot k. If* $\mathsf{max_causal_rate}(h_a, h_b, k) = 0$*, then* $\neg \mathsf{CrossHistCausal}(h_a, h_b, k)$*.*

Proof. Assume $\mathsf{CrossHistCausal}(h_a, h_b, k)$. By Definition 9419.16, this is a genuine cross-Hist causal dependence, so its witness contributes at least one affected target step and therefore a positive dependence ratio. By Definition 9419.17, every such ratio is bounded above by $\mathsf{max_causal_rate}(h_a, h_b, k)$. If that bound is 0, the positive ratio cannot exist. Contradiction. □

Chapter 9420

Consciousness, Universe, Observer: The Phenomenological Frame of the Closed Substrate

Concretized in §513: Finite observer-locality cells package paired observer-history rows, paired GAP acknowledgements, and local coherence routes as a concrete NameCert surface.

Concretized in §578: Conscious observer states are realized as finite BHist packets carrying current state, Self-recognition, observation ledger, GAP acknowledgement, Cont route, and local NameCert rows.

Concretized in §609: Observer-coordinate frames realize the local alternative to a global observer frame as finite BHist, locality-cell, cross-Hist causal, bound, transport, route, package, and NameCert rows.

Concretized in §664: Observation-time order realizes BEDC-internal time as monotonic BHist record retention with Cont routes, GAP erasure boundaries, and a local NameCert surface.

Concretized in §877: Observer accumulation clocks realize time as finite state, successor-record, accumulation-edge, GAP-residue, transport, route, package, and NameCert rows.

Concretized in §1109: Focused multi-Hist ledgers realize the finite closed record-world carrier with focus, selector, GAP, relation, invariant, route, package, and NameCert rows.

Concretized in §1121: Observer-budget supports are realized as finite observer-coordinate, finite-support, cross-Hist causal, budget, transport, route, provenance, and NameCert rows.

This vision chapter extends the closed-observational-system framework of chapter 9393 from BEDC's substrate self-description into a records-side phenomenological reading. The object-level target is deliberately small: a unary horizon, a finite displayed ledger of generated Hist rows, one focus row, local Cont time-stream rows, selector ledgers, GAP-accounted compression, and witnessed inter-Hist relation rows. The chapter therefore does not close the universe itself, phenomenal consciousness as an essence, T, a physical light-speed constant, or a global spacetime frame. It closes only the finite records-side carrier through which those boundary readings can be stated without hidden supply.

9420.1 Finite records-side closure target

Definition 9420.1 (Finite focused multi-Hist ledger)**.** *For a horizon N : Unary, a* ClosedFocusedMultiHistLedger$_N$ *is a finite records-side packet*

$$\mathcal{L}_N = (I, H, \Theta, F, S, D, G, R, J, Q, \rho, A)$$

with the following displayed rows:

1. *I is a finite displayed index ledger for the histories admitted by the packet, not a host-level set of all observers;*
2. *H assigns to each displayed index row i a generated* Hist *row $H(i)$ whose displayed trace route is bounded by the unary horizon N;*
3. *Θ records, for every displayed index row i, a local* Cont *chain from* e *to $H(i)$, read as the local time-stream row of $H(i)$;*
4. *F is a displayed focus-index row of I, and it is only an index of reading, not a subject parameter;*
5. *S records the selector ledger for every visible transition in the displayed traces, including the chosen* BMark *row and the corresponding* Cont *witness;*
6. *D records either the identity digest row $D(i) = H(i)$ or a displayed package-token digest row for $H(i)$;*
7. *G records the* GAP *or provenance row attached to every non-identity digest row, including the exactness status that prevents digest equality from being read as source equality without a witness;*
8. *R is a finite displayed ledger of inter-*Hist *relation rows, each with left history, right history, finite evidence, and* hsame*-stability data;*
9. *J is a finite displayed ledger of inter-*Hist *invariant rows, each stable under componentwise* hsame *transport;*
10. *Q is a finite displayed ledger of dependence rows between local unary time slots, together with an optional unary maximum-rate row computed only from the displayed dependence witnesses;*
11. *ρ is the focused self-proxy, consisting of the endpoint and trace of $H(F)$;*
12. *A is the refusal and audit row recording no subject parameter, no global clock, no hidden chooser, no unledgered compression, no object-level* T *field, and no hidden use of choice, quotient soundness, or propositional extensionality in the packet's accepted read.*

Principle 9420.2 (Closed means generated, ledgered, and witnessed)**.** *The packet of Definition 9420.1 is read as a records-side closed carrier because every displayed object row has a generator, every branch has a selector ledger, every compression has a* GAP *or provenance row, every inter-history relation has a finite witness, every invariant is classifier-stable, and every boundary pressure remains outside the object layer. The packet contains no carrier for an observer-substance, no global synchronization clock, no global universe object, no physical constant as primitive datum, and no internal* T *object.*

Remark 9420.3 (Boundary readings)**.** *The finite ledger may support readings about universe, consciousness, self-location, physical invariance, and other minds, but those readings are not themselves extra fields of the packet.* SelfProxy *is the focused endpoint and trace of $H(F)$, not phenomenal first-personness as an object. A unary maximum-rate row is a packet invariant, not the empirical value of light speed. A boundary tag for* T*, when present in surrounding exposition, is not consumed by any theorem of $\mathcal{L}_N$.*

9420.2 Consciousness as closed observational system

Definition 9420.4 (Consciousness, records-side carrier)**.** *A* conscious records-side carrier *in a finite focused ledger $\mathcal{L}_N$ is the focused row*

$$\mathrm{ConsciousRecord}(\mathcal{L}_N, F) := (H(F), \Theta(F), S(F), \rho)$$

read through the audit row A. It consists of a generated focused Hist*, its local* Cont *trace, the selector witnesses for its visible transitions, and the endpoint/trace self-proxy. It is a carrier for consciousness readings, not an object-level definition of phenomenal consciousness.*

Principle 9420.5 (The carrier does not add a subject)**.** *Definition 9420.4 uses only the rows already visible in Definition 9420.1. The focus index selects a local reading surface; it does not create an observer-substance, a hidden first-person carrier, or an identity criterion beyond* hsame *and the displayed trace ledger.*

Example 9420.6 (BEDC contributor as records-side carrier)**.** *A BEDC contributor can be read, on the records side, as a focused history whose visible work consists of consulted text, proof traces, mechanisation checks, and accepted manuscript rows. The production of a* `leanchecked` *lemma is represented by a trace extension with a selector ledger and a verification row. The reading does not assert a substrate-internal model of the contributor's first-person life; it records only the focused history surface through which that contribution is public.*

Remark 9420.7 (Relation to the implicit metaphysics chapter)**.** *chapter 9418 argues that BEDC's finite kernel identifies an observer with the history that observer has accumulated. Definition 9420.4 re-uses exactly that identification. The self-proxy is the focused endpoint and trace, while identity preservation is the* hsame*-stable continuity of the displayed history. The engineering content remains in the earlier chapter; the present chapter records the finite carrier on which the phenomenological reading may be placed.*

9420.3 Embodied continuation and the mortality boundary

Definition 9420.8 (Embodied support row)**.** *For a focused conscious record* $\mathrm{ConsciousRecord}(\mathcal{L}_N, F)$*, an* embodied support row *is an external finite carrier reading*

$$\mathcal{B}_i(t) = (\mathrm{body}_i(t), \mathrm{metabolism}_i(t), \mathrm{repair}_i(t), \mathrm{environment}_i(t), \mathrm{risk}_i(t))$$

together with a predicate

$$\mathrm{EmbValid}_i(h, t)$$

read as: the carrier row at local stage t can realize the focused history surface h as an inscription-bearing conscious record. The row is not a field of the pure Hist *signature. It is the wick, vessel, and weather around the flame of a local inscription; the flame may have a formal shape, but its burning is a carrier fact.*

Definition 9420.9 (Alive, embodied continuation, and death)**.** *Let $H_i(t)$ denote the focused history displayed at local stage t. The records-side predicates are*

$$\begin{aligned}
\mathrm{Alive}_i(t) &:\Longleftrightarrow \mathrm{EmbValid}_i(H_i(t), t),\\
\mathrm{EmbCanContinue}_i(t) &:\Longleftrightarrow \exists d, h' \left(\mathsf{Cont}(H_i(t), d, h') \wedge \mathrm{EmbValid}_i(h', t+1)\right),\\
\mathrm{Death}_i(t) &:\Longleftrightarrow \mathrm{Alive}_i(t) \wedge \neg\mathrm{EmbCanContinue}_i(t),\\
\mathrm{Mortal}_i &:\Longleftrightarrow \exists t\, \mathrm{Death}_i(t).
\end{aligned}$$

Thus death is not a content subsequently experienced by the same focused row. It is the failure of the next inscription condition for that row. The lamp does not record darkness after its oil is gone; rather, the local condition under which light is recorded no longer carries the next mark.

Principle 9420.10 (Formal continuation is not embodied continuation)**.** *The pure* Hist/Cont *layer can speak about formal successor histories. It contains no body, repair process, hazard rate, metabolic budget, or environmental support row. Consequently a formal extension*

$$\exists d, h' \, \mathsf{Cont}(h, d, h')$$

does not by itself imply

$$\exists d, h' \, (\mathsf{Cont}(h, d, h') \wedge \mathrm{EmbValid}_i(h', t+1)).$$

The distinction is the difference between a riverbed drawn on a map and water continuing through it. The map may display another segment; the flow also requires a source, a channel not yet blocked, and conditions that do not destroy the carrier.

Theorem 9420.11 (Finite reserve mortality)**.** *Suppose a focused embodied support row carries a viability reserve*

$$V_i : \mathbb{N} \to \mathbb{N}$$

with $V_i(0) = K < \infty$*, and suppose* $\mathrm{Alive}_i(0)$ *holds. Suppose that, before the first terminal boundary, positive reserve supports the current record and strictly decreases:*

$$V_i(t) > 0 \implies V_i(t+1) \leq V_i(t) - 1.$$

If reserve exhaustion blocks either the current stage or the next embodied continuation,

$$V_i(t) = 0 \implies \neg\mathrm{EmbCanContinue}_i(t)$$

and

$$V_i(t+1) = 0 \implies \neg\mathrm{EmbCanContinue}_i(t),$$

while $V_i(t) > 0$ *implies* $\mathrm{Alive}_i(t)$*, then* Mortal_i*.*

Proof. By induction on $n \leq K$, the strict decrease condition gives

$$V_i(n) \leq K - n.$$

At $n = K$ this yields $V_i(K) \leq 0$. Since $V_i(K) \in \mathbb{N}$, one has $V_i(K) = 0$. Let $m \leq K$ be the first local stage with $V_i(m) = 0$. If $m = 0$, then $\mathrm{Alive}_i(0)$ and the current-stage blocking condition give $\mathrm{Death}_i(0)$. If $m > 0$, then $V_i(m-1) > 0$, hence $\mathrm{Alive}_i(m-1)$, and the next-stage blocking condition gives $\neg\mathrm{EmbCanContinue}_i(m-1)$. Therefore $\mathrm{Death}_i(m-1)$ by Definition 9420.9. In either case there exists a death stage, so Mortal_i. □

Corollary 9420.12 (Finite records do not close immortality)**.** *A finite ledger of alive stages,*

$$\mathrm{Alive}_i(t_1), \ldots, \mathrm{Alive}_i(t_k),$$

does not close

$$\forall t \, \mathrm{EmbCanContinue}_i(t).$$

An assertion of unbounded embodied continuation therefore requires either a displayed rule proving the required carrier, repair, and risk rows for every later local stage, or a boundary acknowledgement that the assertion is forward-binding and not object-level closure inside $\mathcal{L}_N$*.*

Remark 9420.13 (The boundary is local)**.** *The death predicate above is a local inscription-boundary predicate. It does not say that* T *disappears, that existence as such is extinguished, or that the pure possibility of further* Hist *formation is refuted. It says only that this focused embodied carrier no longer supplies the next valid conscious record. A broken lyre does not prove that music is impossible; it proves that this instrument no longer realizes the song.*

Concretized in §509: Locality cells are realised as a bilateral BHist/GAP NameCert packet for inter-observer event coordination.

9420.4 Universe as observation history

Definition 9420.14 (Finite universe-for-the-focus)**.** *For a finite focused ledger $\mathcal{L}_N$, the* finite universe-for-the-focus *is the records-side family*

$$\mathrm{Universe}_{\mathcal{L}_N} := (I, H, \Theta, S, D, G, R, J, Q)$$

together with the focus row F when a focused read is required. The row $\Theta(i)$ is the displayed local Cont *route whose unary bound is recorded against N. Thus the universe-for-the-focus is not a set comprehension over an external domain; it is the finite packet surface whose selector, digest,* GAP*, relation, invariant, and dependence rows are displayed. It is not the universe itself and not a God-view object; it is the finite slice whose records the packet admits.*

Principle 9420.15 (Phenomenological frame)**.** *The finite universe-for-the-focus is exactly the records-side slice visible through $\mathcal{L}_N$. Anything not generated by the displayed histories, not recorded by selector or relation rows, not preserved by a* GAP *ledger, and not exposed as a boundary tag is outside this object-level slice. Distinct focused ledgers need not share a world object; cross-perspective alignment remains the ledger-local problem treated in chapter 9419.*

Remark 9420.16 (The frame is structural, not metaphysical)**.** *Definition 9420.15 does not assert that there is no observer-independent reality. It asserts only that the BEDC-side carrier is the finite records-plus-ledger slice. Any pressure beyond that slice appears either as an additional displayed row, as a* GAP *or provenance obligation, or as a boundary reading whose far end is not internalized.*

9420.5 Closed history network reading

Principle 9420.17 (Universe as closed distinction/history/continuation network)**.** *At the records-side level, the universe-for-the-focus of Definition 9420.14 is read neither as matter contained in a prior spacetime nor as a program executed by an external machine. Its closeable object is the displayed network of distinctions, generated histories, continuation routes, event boundaries, classifier-stable readbacks, and public naming certificates. The shortest BEDC reading is therefore:*

$$\mathrm{Universe}_{\mathcal{L}_N} \quad \textit{as read by the packet} \quad = \textit{closed distinction/history/continuation network.}$$

The equality is a reading of the packet surface, not an identity theorem about a completed cosmos.

Definition 9420.18 (Closed BEDC universe surface)**.** *A* closed BEDC universe surface *over a focused ledger $\mathcal{L}_N$ is the records-side admission discipline in which every public row is one of the following:*

1. *a generated* Hist *row;*

2. *an event-flow row whose event boundaries remain part of the datum;*
3. *a finite packet all of whose object-facing fields are* Hist *rows and whose public readback is mediated by a recognizer, route, or taste gate;*
4. *a* NameCert *or semantic certificate over a displayed carrier and classifier;*
5. *a route generated by* Cont*, together with the transport, provenance, and naming rows needed for public reuse.*

The surface admits no hidden host object as a formal input and no global observer outside the displayed family.

Theorem 9420.19 (Closed history universe decomposition)**.** *On a closed BEDC universe surface, time, causality, local differential response, information scoring, boundary readback, observer-local coordinates, relativistic rate structure, and multi-history superposition are certified modes of history continuation rather than primitive substances.*

Proof. Each case is read through the same finite surface. Time is the local Cont route $\Theta(i)$ producing $H(i)$ from e in Definition 9420.27. Causality is the continuation discipline of Definition 9420.24, with the concrete maximum-rate route supplied by the $\mathsf{MaxCausalRate}^{\uparrow}$ surface cited in chapter 9419 and realized by chapter 594. Local differential response is the displayed quotient-continuation row of the derivative carrier in chapter 122. Information scoring and entropy readback use finite distribution, score, expectation, metric, ledger, and provenance rows as in chapter 228 and chapter 192. Boundary readback is constrained by the event-flow non-injectivity of Definition 9207.7 and by finite boundary-ledger packets such as the Stokes-style fold-defect surface. Observer-local coordinates are finite packets over observer state, locality cell, multi-history configuration, causal reads, transport, routes, provenance, and local naming, as in chapter 609. Relativistic rate structure is read only after multi-history configuration, no-synchronization refusal, maximum-rate, symmetry, and transport rows have been displayed, as in chapter 617. Multi-history superposition is read as a finite packet over configuration, refinement, causal, rate, phase, history, route, provenance, and naming rows, as in chapter 655. None of these cases requires a primitive matter substrate, a primitive spacetime container, a completed external state table, or a global observer. □

Corollary 9420.20 (Boundary-sensitive information)**.** *Public information on a closed BEDC universe surface cannot be reduced to a bare mark stream.*

Proof. If the public surface were only a bare stream of marks, then event segmentation could be erased without loss. But Definition 9207.7 gives distinct event flows with the same erased mark stream. Therefore boundary data is part of the public information surface. □

Corollary 9420.21 (Certified cosmological reading)**.** *The records-side cosmological reading of BEDC is that the universe-for-the-focus is a self-recording and self-decoding history surface: distinctions generate events, event boundaries generate readable flows, continuations generate local time and causal routes, inter-history invariants generate spatial and relativistic readings, non-collapsed refinement packets generate quantum-shaped readings, and certificates generate public objects.*

Proof. This is the synthesis of Definition 9420.18 and Definition 9420.19. The listed readings exhaust the admitted public modes of the surface: generated histories, boundary-sensitive event flows, continuation routes, inter-history invariants, finite packet readbacks, and naming certificates. □

Definition 9420.22 (Matter-pattern read)**.** *A* matter-pattern read *in $\mathcal{L}_N$ is a family of displayed event and history rows whose classifier remains stable under* hsame*, whose consequences can be replayed through local or inter-*Hist Cont *routes, whose recognition is transportable across locality cells, and whose public name is protected by a* NameCert *or by an exposed* GAP *when the name is not yet certified. The read does not introduce a primitive material substance. It names a recurrent and resistant pattern of public recordability.*

Principle 9420.23 (Reality as public constraint)**.** *In the closed ledger, a structure is real in the relevant records-side sense when it is repeatedly recordable, continuation-constraining, transportable across compatible histories, and resistant to arbitrary reclassification. This is stronger than private imaginability and weaker than a claim of access to an external bare object. It is the finite public constraint surface on which scientific objecthood later attaches the empirical and bridge rows of Definition 9462.922.*

Principle 9420.24 (Causality as continuation discipline)**.** *Causality is read as discipline on allowed continuations. A causal row says that some source-history fragment constrains a target-history fragment through a displayed dependence route, bounded slot window, replayable* Cont *witness, and transport stability. A cause is therefore not a substance pushing another substance inside a pre-given arena; it is a ledgered restriction on which later histories remain admissible after the relevant prior rows have been recorded.*

Remark 9420.25 (Relativistic and quantum boundary readings)**.** *The relativistic reading arises from the absence of a global clock: if no history supplies a privileged synchronized present for all histories, physical content must be carried by inter-*Hist *invariants, maximum-rate rows, and observer-symmetric transport as in chapter 9419. The quantum reading arises from the absence of a completed external state table: before a single record has selected a displayed continuation, the packet may need to preserve multiple refinement, phase, causal, and route-coherence rows without collapsing them to one history. These readings are not proofs of relativity or quantum mechanics. They identify the form in which such theories can enter BEDC: as certified packet surfaces rather than as primitive spacetime, primitive wavefunction, or external state inventory.*

Remark 9420.26 (Event boundaries are part of information)**.** *The ground event-flow layer makes boundary information explicit. By Definition 9207.7, erasing event boundaries can identify distinct event flows. Hence the closed history network cannot be reduced to a bare bitstream of marks. It consists of events together with their boundaries, recognizers, ledgers, and certificates. A boundary is not an outer shell around already formed information; it is one of the conditions under which a row becomes a readable event rather than an undifferentiated string.*

9420.6 Time as monotonic accumulation

Definition 9420.27 (BEDC-internal time)**.** *For a focused ledger $\mathcal{L}_N$, the* internal time order *of $H(i)$ is the local* Cont *chain $\Theta(i)$ producing $H(i)$ from* e*. Equivalently, it is the finite trace order induced by the displayed* BMark *selector rows. Internal time is local to a history; it is not a scalar shared by all histories in I.*

Principle 9420.28 (Time is monotonic information accumulation)**.** *For BEDC the object-level notion of "later" is trace extension inside a displayed local ledger. This is the information-conservation invariant of Definition 9393.3 read as a temporal structure. Reversibility, time symmetry, global simultaneity, or retrocausal phrasing may be studied as additional structures, but they are not fields of $\mathcal{L}_N$ unless they arrive as witnessed extension carriers.*

Example 9420.29 (Commit history as BEDC-internal time)**.** *The git commit history of* `newmath` *is a project-level analogy for local ledger time: each accepted state is read through accumulated records such as* `leanchecked` *lemmas,* `inductive` *types, paper content, and audit rows. The analogy is records-side only. It does not supply a global clock for all observers; it illustrates how a local trace can serve as the public time order of one development history.*

Remark 9420.30 (Why time has only one direction in BEDC)**.** *External physics admits various time-symmetric or time-reversed formulations. BEDC's records-side reading does not exclude such formulations; it observes only that a local ledger is asymmetric because its displayed trace has a generated prefix structure. A global clock or global "now" would require a separate extension carrier and cannot be read from the kernel signature.*

9420.7 Focused still-point and information-clock rate

Definition 9420.31 (Focused still-point reading)**.** *For a finite focused ledger $\mathcal{L}_N$, the* focused still-point reading *is the constraint that every accepted observer-facing read of the focus factors through*

$$H(F), \quad \Theta(F), \quad S(F), \quad \rho, \quad A,$$

together with the displayed hsame, Cont, Pkg, NameCert, *locality-cell, and inter-*Hist *relation rows of the same packet. It has no field for an observer-substance coordinate, a host spacetime position, a shared external time scalar, or a global frame. The word still names this absence of a detachable observer carrier; it does not mean that the focused history has no continuation.*

Principle 9420.32 (Motion is read as record-side change)**.** *In the focused still-point reading, the sentence that an observer moves is admissible only as a statement about changes in displayed rows: extension of $H(F)$ along $\Theta(F)$, selector changes in $S(F)$, locality-cell changes against other histories, and readout changes in the invariant or dependence ledgers. No independent subject is transported through a pre-given coordinate arena. The transported object is the certified read of a focused history and its packet surface.*

Definition 9420.33 (Accumulation edge and clock read)**.** *An* accumulation edge *at the focus is a displayed transition row*

$$\mathrm{AccEdge}_F(t) := (H(F,t), H(F,t+1), d_t, \Theta_t, S_t, G_t)$$

whose continuation witness records

$$\mathsf{Cont}(H(F,t), d_t, H(F,t+1)).$$

An observer accumulation clock read *is any finite packet read that counts, bounds, or classifies such accumulation edges through the observer-accumulation-clock and local-budget rows already present in $\mathcal{L}_N$. The read is local to the focus and remains audited by A.*

Definition 9420.34 (Information-clock rate)**.** *For a focused ledger window $[u, v]$ and an announced visible-row class $\mathcal{K}$, the* information-clock rate *is a records-side read comparing the number of accepted accumulation edges in the window with the amount of $\mathcal{K}$-visible distinction, boundary, selector, digest, relation, invariant, or dependence data exposed by those edges. It is a derived read over finite ledger rows, not a primitive physical law and not an empirical clock-rate constant.*

Proposition 9420.35 (Clock-rate reads are local and budgeted)**.** *Every information-clock rate read of Definition 9420.34 is local to the focused packet and must preserve the audit row A. It may compare two windows of one focused history, or two focused rows through a locality cell, only after the relevant* hsame *transport,* Cont *replay, provenance, and inter-*Hist *relation rows are displayed.*

Proof. The rate read is defined from accumulation edges. Each accumulation edge contains a present history, a successor history, a distinction row, a continuation witness, selector data, and provenance or gap data. These are rows of the finite packet. A comparison across windows or across histories therefore has no public meaning until the packet supplies the corresponding transport, replay, provenance, and locality witnesses. Since A records refusal of a subject parameter, global clock, hidden chooser, and unledgered compression, preserving A prevents the rate read from becoming a primitive clock or a host-frame comparison. □

Corollary 9420.36 (Information density does not add a spacetime primitive)**.** *If a window exposes more visible information per accepted accumulation edge than another window, the BEDC-side conclusion is only a relation between finite ledger reads. It does not add a spacetime metric, a physical time-dilation law, or an observer-substance moving through a container. Any physical interpretation must pass through the inter-*Hist *invariant and empirical-ledger disciplines of chapter 9419 and subsection 9462.28.6.*

9420.8 Space as inter-observer coordination

Principle 9420.37 (No god's-eye coordinate system)**.** *Two focused histories in $\mathcal{L}_N$ do not share a single observation history or a global time coordinate. The relation "these histories coordinate on an event" is therefore not primitive. It is reconstructed from paired records,* GAP *or provenance rows, and relation witnesses. chapter 9419 works out the operational form of this coherence and treats maximum-rate phenomena as inter-*Hist *invariant interfaces with displayed unary-rate rows, not as primitive physical constants.*

Definition 9420.38 (Inter-system locality cell)**.** *For two displayed index rows i, j in I and an event row e, the* locality cell *of e across (i, j) is a finite relation row containing a record read from $H(i)$, a record read from $H(j)$, a paired* GAP *or provenance acknowledgement, and a stability witness saying that the cell respects componentwise* hsame*. The cell is the records-side form of coordinated event reading.*

Example 9420.39 (Space as locality-cell geometry)**.** *Within $\mathcal{L}_N$, space is the finite relation graph carried by locality cells and inter-*Hist *invariant rows. A unary maximum-rate read, when present, is computed from displayed dependence rows and remains an invariant of that packet. It is not the empirical light-speed constant and not a completed spacetime manifold; it is the records-side surface on which such a physical reading may later be certified.*

Remark 9420.40 (Spacetime logic as inter-observer logic)**.** *What is usually called "spacetime logic" enters this chapter only as a boundary reading of relation rows, dependence rows, and inter-*Hist *invariants. The closed records-side object has no separate spacetime carrier. It has generated histories, witnessed relations, local clocks, and refusal of global synchronization. Any stronger physical geometry belongs to a subsequent* NameCert *target.*

9420.9 Love as other-Hist positive continuation

Definition 9420.41 (Other-Hist recognition)**.** *For two focused rows i, j in a finite ledger, let* $\mathrm{Rep}_i(j,t)$ *be the representation of j carried inside $H_i(t)$. An* other-Hist recognition *predicate is*

$$\begin{aligned}\mathrm{Recognize}_i(j,t) :\Longleftrightarrow\ & \neg\mathsf{hsame}(H_i(t), H_j(t))\\ & \wedge \neg\mathrm{Projection}_i(j,t)\\ & \wedge \neg\mathrm{Objectify}_i(j,t)\\ & \wedge \mathrm{CorrigibleBy}_{H_j}(\mathrm{Rep}_i(j,t))\\ & \wedge \mathrm{GapHonest}_i(H_j(t), \mathrm{Rep}_i(j,t)).\end{aligned}$$

The recognition says that j is not the image of j inside i, not a tool-slot inside i's purposes, and not a fully compressible region of i's ledger. The other history stands like a shoreline seen from one vessel: charted enough for approach, never identical with the chart.

Definition 9420.42 (Positive continuation care)**.** *Let V_j be a displayed value horizon for j's continuation. A care row from i toward j is*

$$\mathrm{Care}^{V_j}_{i\to j}(t) :\Longleftrightarrow \mathrm{Policy}_i(t) \text{ supports } \mathsf{Cont}^+_{V_j}(H_j,t) \text{ under } \mathrm{Recognize}_i(j,t),$$

where $\mathsf{Cont}^+_{V_j}(H_j,t)$ denotes continuation of H_j along life, truth, freedom, dignity, relation, or meaning as admitted by V_j. Care is not substitution for the other's history. It is a hand placed beside the path rather than a hand closing over the path.

Definition 9420.43 (Love relation)**.** *The one-way records-side love relation is*

$$\begin{aligned}\mathrm{Love}^{V_j}_{i\to j}(t) :\Longleftrightarrow\ & \mathrm{Recognize}_i(j,t)\\ & \wedge \mathrm{Care}^{V_j}_{i\to j}(t)\\ & \wedge \mathrm{TruthFidelity}_{i\to j}(t)\\ & \wedge \mathrm{SelfTransform}_{i\leftarrow j}(t)\\ & \wedge \mathrm{TValue}_i(j,t).\end{aligned}$$

Here TruthFidelity *refuses systematic falsification and* GAP *concealment;* SelfTransform *permits H_i's classifiers and expectations to be corrected by H_j's reality-facing response; and* TValue *records that j's worth is not exhausted by a utility function for i.*

Principle 9420.44 (Love is not liking, possession, or fusion)**.** *A reward row*

$$\mathrm{Reward}_i(\mathrm{Rep}_i(j,t)) > 0$$

may accompany love, but it does not imply love: the reward may be produced by projection rather than by other-Hist recognition. Conversely, love may remain when reward is absent, because its core is fidelity to another history's positive continuation. Possession fails because it objectifies H_j as a control region of H_i; fusion fails because it erases the GAP*-honest distance that recognition requires. A mirror can flatter, a chain can hold, and a fog can merge outlines; none of these is the relation defined above.*

Theorem 9420.45 (Love preserves otherness while supporting continuation)**.** *If* $\mathrm{Love}^{V_j}_{i\to j}(t)$ *holds, then*

$$\mathrm{Recognize}_i(j,t), \qquad \neg\mathrm{Objectify}_i(j,t), \qquad \neg\mathrm{Projection}_i(j,t),$$
$$\mathrm{Care}^{V_j}_{i\to j}(t), \qquad \mathrm{TruthFidelity}_{i\to j}(t), \qquad \mathrm{TValue}_i(j,t).$$

Thus love is a relation from one finite history to another finite history's positive continuation, under preserved otherness and value that is not exhausted by utility.

Proof. Unfold Definition 9420.43. The displayed consequences are its conjuncts, together with the non-projection and non-objectification clauses obtained by unfolding Definition 9420.41. The final reading follows from Definition 9420.42 and the TValue conjunct. □

Corollary 9420.46 (Mutual love is a non-fusing two-Hist loop)**.** *If both* $\mathrm{Love}^{V_j}_{i\to j}(t)$ *and* $\mathrm{Love}^{V_i}_{j\to i}(t)$ *hold, and a shared meaning row* $R_{ij}(t)$ *is displayed, then the relation forms a two-Hist productive loop:*

$$H_i(t) \to H_j(t) \to H_i(t+1), \qquad H_j(t) \to H_i(t) \to H_j(t+1),$$

with neither direction collapsing $\mathsf{hsame}(H_i, H_j)$ *into identity. The loop is not one history swallowed by another; it is two lamps keeping distinct flames while sharing light on a finite table.*

Remark 9420.47 (Love, death, and grief)**.** *Because love supports another embodied history's positive continuation, it is exposed to the mortality boundary of section 9420.3. If* $\mathrm{Death}_j(t)$ *holds after* $\mathrm{Love}^{V_j}_{i\to j}(t^-)$*, grief is the focused readback in which fidelity remains while j's embodied continuation has failed. The weight of grief is therefore not a contradiction of love; it is love meeting the finite carrier condition it had already refused to falsify.*

9420.10 Observer identity through history

Principle 9420.48 (Identity is history-continuity)**.** *The identity of a focused records-side carrier across local time is given by the generated history and its* hsame*-stable trace. There is no separate "self" inhabiting the chain and persisting through it; the self-proxy is the focused endpoint and trace. chapter 9418's implicit-metaphysics reading is the engineering form of this claim.*

Example 9420.49 (BEDC project identity)**.** *The BEDC project as a development artefact is identifiable through a connected accumulation of public records and audit rows. The example is a records-side analogy for* hsame*-stable trace identity. It does not posit a project-substance behind the ledger, and it does not turn repository history into a global observer.*

Remark 9420.50 (Buddhist and Heraclitean correspondences)**.** *The history-continuity reading of identity has correspondences in several philosophical traditions: the Buddhist denial of a substantial ātman; Heraclitus's flux-with-pattern; Hume's bundle theory. The BEDC version keeps only the formal residue of that family: the observer row is a* Hist *trace, and sameness is mediated by* hsame *and displayed continuity, not by a subject-substance.*

9420.11 The hard problems in the closed-substrate frame

Principle 9420.51 (Hard problems as structural questions about closure)**.** *The "hard problems" usually phrased about consciousness, universe, observer, and spacetime become, in the closed records-side frame, boundary questions around a finite carrier: which records are generated, which*

branch choices are ledgered, which compressions preserve provenance, which relations have witnesses, which invariants are stable, and which pressures remain outside the object layer. None of these licenses a separately postulated mental, material, temporal, or spatial substance inside $\mathcal{L}_N$.

Traditional hard problem	**Closed-substrate reformulation**
What is consciousness?	The closed records-side carrier is a focused self-tracking Hist with trace, endpoint, and selector ledger; phenomenal consciousness itself is not object-level content (Definition 9420.4).
What is the universe?	A finite horizon slice $\text{Universe}_{\mathcal{L}_N}$ of generated histories and their ledgers, not a global universe object (Definition 9420.14).
What is matter?	A recurrent, transportable, classifier-stable, continuation-constraining pattern of public recordability, not a primitive substance (Definition 9420.22, Definition 9420.23).
What is causality?	A ledgered restriction on which continuation routes remain admissible after source-history rows are recorded, not a primitive pushing relation (Definition 9420.24).
What is time?	A local Cont chain producing a history from e; no global clock is a field of the carrier (Definition 9420.27).
What is space?	A finite graph of locality cells, relation rows, and inter-Hist invariants, not a completed spacetime manifold (Definition 9420.38).
Why relativity-shaped structure?	No global synchronization forces frame-independent physical content into observer-symmetric inter-Hist invariants and maximum-rate packet surfaces (Definition 9420.25).
Why quantum-shaped structure?	No external completed state table leaves refinement, phase, causal, and route-coherence rows as packet data until a record selects a displayed continuation (Definition 9420.25).
What is identity?	The hsame-stable trace and focused endpoint; no subject-substance is carried behind the history (Definition 9420.48).
What is death?	The local failure of embodied continuation for a currently valid focused record; not a theorem of pure Hist/Cont and not the disappearance of T (Definition 9420.9, Definition 9420.13).
What is love?	A fidelity relation from one finite history to another finite history's positive continuation, preserving otherness and refusing projection, objectification, fusion, and utility exhaustion (Definition 9420.43).
What is the relation between observer and observed?	Not a separate observation channel; the observer row is a Hist, and observed content is what the row and its ledgers expose.
Is there a god's-eye view?	No: cross-system coherence is reconstructed through witnessed relation rows and locality cells, never given as a single shared world (Definition 9420.37).

What is the arrow of time?	The asymmetry follows from local generated trace structure; global simultaneity requires an extension carrier (Definition 9420.30).

Remark 9420.52 (Continuity with the transcendental-input chapter)**.** *Read together with chapter 9393, the picture is: $\mathcal{L}_N$ is the object-level closed carrier, while* T *remains the non-internalized boundary position for forward-binding commitments. Time, space, consciousness, universe, mortality, and love appear in this chapter only as records-side carriers or boundary readings. The transcendental supply position is not a field of the carrier and is not consumed by its closure argument.*

9420.12 Closure remarks

This vision chapter identifies the finite carrier that can be closed without turning boundary readings into substrate theorems. The closeable records-side carrier is $\mathrm{ClosedFocusedMultiHistLedger}_N$: a finite multi-observer record-world with a focused self-trace, selector ledgers, GAP-accounted compression, witnessed inter-Hist relations, stable invariants, and finite displayed dependence rows.

The chapter pairs with chapter 9393 for the non-internalized supply boundary, with chapter 9418 for observer-as-history, with chapter 9419 for ledger-local multi-Hist coordination, with chapter 9207 for boundary-sensitive event information, and with chapter 9402 for digest/GAP discipline. Concrete consequences for physical theories, cognitive science models, or first-person phenomenology are not closed here; they require subsequent $\mathsf{NameCert}$ targets or remain boundary readings.

Concretized in §647: Observer perspective classifiers realize universe-for-observer comparison as finite BHist locality-cell, GAP commitment, hsame transport, Cont route, Pkg provenance, and NameCert rows.

Concretized in §1122: Observer-support lattices realize the observer vision as finite frame, support, locality-cell, budget-support, route, transport, provenance, and NameCert rows.

Chapter 9421

Observer-Centered Logarithmic Section Dynamics

This chapter formalizes the observer-centered reading as a frame selector, not as a psychological substance. It refines the focused-record carrier of chapter 9420 and the egocentric interface packet of section 9414.56. The selector chooses an adjacent reference frame, the chosen frame exposes a complete world-section, the prior ledger is rewritten into that frame, and any residual used by the public read is recorded as an orthogonal ledger row. The solenoid fibre supplies the compact model for a visible phase whose prime-register coordinate may vary without changing the visible circle readout.

9421.1 Frame Selection Data

Definition 9421.1 (Reference-frame graph)**.** *A* reference-frame graph *is an undirected graph*

$$\mathcal{R} = (V_{\mathcal{R}}, E_{\mathcal{R}}).$$

Its vertices are frame rows. For $R, R' \in V_{\mathcal{R}}$, the notation

$$R' \sim R$$

means that $(R, R') \in E_{\mathcal{R}}$. The relation is adjacency in the frame graph, not spatial adjacency inside an already chosen physical model.

Definition 9421.2 (Observer-centered selector)**.** *An* observer-centered selector *is a function*

$$\chi : \mathbb{N}_{\geq 0} \to V_{\mathcal{R}}$$

with values

$$R_n := \chi(n)$$

such that

$$R_{n+1} \sim R_n$$

for every displayed selection step. The selector is centered only in the records-side sense: it is the frame-index row through which the focused history reads a world-section. It is not a subject-substance and not a global observer above the frame graph.

Definition 9421.3 (World Hilbert bundle)**.** *Over a reference-frame graph $\mathcal{R}$, a* world Hilbert bundle *is a family*

$$\mathcal{H}^{\text{world}} = \{\mathcal{H}_R\}_{R \in V_{\mathcal{R}}}$$

of Hilbert spaces. An element

$$B_R \in \mathcal{H}_R$$

is a complete world-section as read in frame R. The word complete refers to the section relative to the frame row; it does not assert an observer-independent completed cosmos.

Definition 9421.4 (Frame reshaping row)**.** *For each displayed adjacency $R' \sim R$, a* frame reshaping row *is an isometric embedding, or a unitary isomorphism when the comparison is reversible,*

$$\Phi_{R' \leftarrow R} : \mathcal{H}_R \to \mathcal{H}_{R'}.$$

It rewrites records from frame R into frame R'. It is not, by this definition, a signal-propagation operator inside one fixed physical space.

Principle 9421.5 (Frame reshaping is not propagation)**.** *A finite-propagation constraint applies to a displayed physical propagator*

$$U_R(t) : \mathcal{H}_R \to \mathcal{H}_R$$

inside one frame. It does not automatically apply to

$$\Phi_{R' \leftarrow R} : \mathcal{H}_R \to \mathcal{H}_{R'}.$$

Using such a constraint across frames requires an additional bridge row identifying the reshaping operation with a physical propagation discipline.

Concretized in §9135: The observer-centered logarithmic section dynamics surface is realized as finite frame-selection, section-reshaping, residual-ledger, logarithmic-readback, solenoid-visible, transport, replay, provenance, and local NameCert rows.

9421.2 Orthogonal Section Step

Definition 9421.6 (Ledger rewrite and residual)**.** *At selection stage n, let*

$$B_n \in \mathcal{H}_{R_n}$$

be the displayed world-section and let

$$\mathcal{L}_n \subseteq \mathcal{H}_{R_n}$$

be the displayed history ledger subspace. After the selector chooses $R_{n+1} \sim R_n$, the rewritten ledger is

$$\mathcal{L}_n^{\uparrow} := \Phi_{R_{n+1} \leftarrow R_n}(\mathcal{L}_n) \subseteq \mathcal{H}_{R_{n+1}}.$$

Let

$$P_n^{\uparrow} : \mathcal{H}_{R_{n+1}} \to \mathcal{L}_n^{\uparrow}$$

be the orthogonal projection onto the rewritten ledger. For

$$B_{n+1} \in \mathcal{H}_{R_{n+1}},$$

the residual row is

$$\eta_{n+1} := B_{n+1} - P_n^{\uparrow} B_{n+1}.$$

Definition 9421.7 (Orthogonal section step)**.** *The transition from B_n to B_{n+1} is an* orthogonal section step *when*

$$\eta_{n+1} \perp \mathcal{L}_n^{\uparrow}$$

and the accepted ledger is

$$\mathcal{L}_{n+1} := \mathcal{L}_n^{\uparrow} \oplus \overline{\mathrm{span}}\{\eta_{n+1}\}.$$

If $\eta_{n+1} = 0$, the step carries no residual direction. If $\eta_{n+1} \# 0$, the residual direction is not absorbed into the prior ledger; it is admitted as its own displayed row.

Theorem 9421.8 (Adjacent selection gives a complete successor section)**.** *Assume $R_{n+1} \sim R_n$ and*

$$B_{n+1} \in \mathcal{H}_{R_{n+1}}.$$

Then B_{n+1} is a complete world-section in the selected successor frame, not a fragmentary alteration inside $\mathcal{H}_{R_n}$.

Proof. By Definition 9421.3, every element of $\mathcal{H}_{R_{n+1}}$ is a complete world-section relative to the frame R_{n+1}. The adjacency hypothesis restricts which frame may be selected after R_n; it does not state that the successor section is a local fragment inside the prior Hilbert space. Therefore the selected object is complete in $\mathcal{H}_{R_{n+1}}$. □

Theorem 9421.9 (Orthogonal ledger decomposition)**.** *For an orthogonal section step,*

$$\mathcal{L}_{n+1} = \mathcal{L}_n^{\uparrow} \oplus E_{n+1}, \qquad E_{n+1} := \overline{\mathrm{span}}\{\eta_{n+1}\},$$

and

$$E_{n+1} \perp \mathcal{L}_n^{\uparrow}.$$

Proof. The first displayed equality is exactly the ledger clause in Definition 9421.7. Since $\eta_{n+1} \perp \mathcal{L}_n^{\uparrow}$, every vector in the closed span of η_{n+1} is orthogonal to $\mathcal{L}_n^{\uparrow}$. Hence $E_{n+1} \perp \mathcal{L}_n^{\uparrow}$. □

Corollary 9421.10 (Pythagorean readout for a rewritten section)**.** *Assume*

$$B_{n+1} = \Phi_{R_{n+1} \leftarrow R_n}(B_n) + \eta_{n+1},$$

with

$$\eta_{n+1} \perp \mathcal{L}_n^{\uparrow}$$

and

$$\Phi_{R_{n+1} \leftarrow R_n}(B_n) \in \mathcal{L}_n^{\uparrow}.$$

Then

$$\|B_{n+1}\|^2 = \|\Phi_{R_{n+1} \leftarrow R_n}(B_n)\|^2 + \|\eta_{n+1}\|^2.$$

If $\Phi_{R_{n+1} \leftarrow R_n}$ is isometric on B_n, then

$$\|B_{n+1}\|^2 = \|B_n\|^2 + \|\eta_{n+1}\|^2.$$

Proof. The two summands in the displayed decomposition are orthogonal, so the Hilbert-space Pythagorean identity applies. The final equality is the isometry clause for $\Phi_{R_{n+1} \leftarrow R_n}$. □

9421.3 Logarithmic Time Readout

Definition 9421.11 (Multiplicative accumulation scale)**.** *A positive scale row is a sequence*

$$A_n > 0$$

with displayed multipliers

$$\lambda_{n+1} > 0$$

such that

$$A_{n+1} = \lambda_{n+1} A_n.$$

Definition 9421.12 (Logarithmic observer-time coordinate)**.** *The logarithmic observer-time coordinate attached to the scale row is*

$$T_n := \log A_n.$$

It is a coordinate on multiplicative accumulation, not a global clock shared by all histories.

Theorem 9421.13 (Multiplicative accumulation becomes additive in log time)**.** *If*

$$A_{n+1} = \lambda_{n+1} A_n,$$

then

$$T_{n+1} - T_n = \log \lambda_{n+1}$$

and hence

$$T_n = T_0 + \sum_{j=1}^{n} \log \lambda_j.$$

If all displayed multipliers are equal to one value λ, then

$$T_n = T_0 + n \log \lambda.$$

Proof. By Definition 9421.12,

$$T_{n+1} = \log A_{n+1} = \log(\lambda_{n+1} A_n) = \log \lambda_{n+1} + \log A_n = T_n + \log \lambda_{n+1}.$$

Iteration gives the displayed sum. The constant-multiplier case follows by substituting $\lambda_j = \lambda$ in that sum. □

9421.4 Visible Stability and Solenoid Fibres

Definition 9421.14 (Visible readout and adjacent continuity)**.** *Let*

$$V_R : \mathcal{H}_R \to \mathcal{O}$$

be a visible readout into a metric readout space $(\mathcal{O}, d_{\mathcal{O}})$. The system satisfies adjacent continuity when there is a constant $L > 0$ such that, for every displayed adjacency $R' \sim R$ and every $B \in \mathcal{H}_R$,

$$d_{\mathcal{O}}\big(V_{R'}(\Phi_{R' \leftarrow R} B), V_R(B)\big) \le L\, d_{\mathcal{R}}(R', R).$$

Theorem 9421.15 (Visible stability bound)**.** *Assume adjacent continuity, assume*

$$B_{n+1} = \Phi_{R_{n+1} \leftarrow R_n}(B_n) + \eta_{n+1},$$

and assume $V_{R_{n+1}}$ is M-Lipschitz on the displayed segment. If

$$O_n := V_{R_n}(B_n),$$

then

$$d_{\mathcal{O}}(O_{n+1}, O_n) \leq L\, d_{\mathcal{R}}(R_{n+1}, R_n) + M\|\eta_{n+1}\|.$$

Proof. Write $\Phi_n := \Phi_{R_{n+1} \leftarrow R_n}$. By the triangle inequality,

$$\begin{aligned} d_{\mathcal{O}}(O_{n+1}, O_n) \leq\ & d_{\mathcal{O}}\big(V_{R_{n+1}}(\Phi_n B_n + \eta_{n+1}), V_{R_{n+1}}(\Phi_n B_n)\big) \\ & + d_{\mathcal{O}}\big(V_{R_{n+1}}(\Phi_n B_n), V_{R_n}(B_n)\big). \end{aligned}$$

The first term is at most $M\|\eta_{n+1}\|$ by the Lipschitz hypothesis. The second term is at most $L\, d_{\mathcal{R}}(R_{n+1}, R_n)$ by adjacent continuity. □

Definition 9421.16 (Solenoid-visible phase readout)**.** *Let Σ_S be the S-solenoid of Definition 9413.779, with visible projection*

$$\pi_S : \Sigma_S \to \mathbb{T}$$

and kernel

$$K_S := \ker \pi_S.$$

A readout

$$O : \Sigma_S \to Y$$

is solenoid-visible *when it factors through the circle projection: there is a map*

$$\bar{O} : \mathbb{T} \to Y$$

such that

$$O = \bar{O} \circ \pi_S.$$

Theorem 9421.17 (Solenoid-visible reads are kernel-blind)**.** *If $O = \bar{O} \circ \pi_S$, then for every*

$$\widetilde{t} \in \Sigma_S$$

and every $k \in K_S$,

$$O(\widetilde{t}k) = O(\widetilde{t}).$$

Proof. Since $k \in K_S$, one has $\pi_S(k) = 1$. Therefore

$$\pi_S(\widetilde{t}k) = \pi_S(\widetilde{t})\pi_S(k) = \pi_S(\widetilde{t}).$$

Applying the factorization $O = \bar{O} \circ \pi_S$ gives the claim. □

Corollary 9421.18 (Visible phase stability may hide fibre motion)**.** *For a solenoid-visible read, two lifts in one π_S-fibre give the same visible output. Thus a prime-register fibre coordinate may vary while the visible unit-circle phase remains fixed.*

9421.5 Synthesis

Theorem 9421.19 (Observer-centered logarithmic section dynamics)**.** *Assume a reference-frame graph, an observer-centered selector, a world Hilbert bundle, frame reshaping rows, orthogonal section steps, a multiplicative accumulation scale with logarithmic observer-time coordinate, and a solenoid-visible phase readout. Then each accepted selection step has the following five rows:*

$$R_{n+1} \sim R_n,$$

$$B_{n+1} \in \mathcal{H}_{R_{n+1}},$$

$$\mathcal{L}_{n+1} = \Phi_{R_{n+1} \leftarrow R_n}(\mathcal{L}_n) \oplus E_{n+1}, \qquad E_{n+1} \perp \Phi_{R_{n+1} \leftarrow R_n}(\mathcal{L}_n),$$

$$T_n = T_0 + \sum_{j=1}^{n} \log \lambda_j,$$

and

$$O(\widetilde{t}k) = O(\widetilde{t}) \qquad (k \in K_S)$$

for every solenoid-visible read.

Proof. The adjacency row is Definition 9421.2. The complete-section row is Definition 9421.8. The ledger decomposition and orthogonality rows are Definition 9421.9. The logarithmic time row is Definition 9421.13. The solenoid kernel blindness row is Definition 9421.17. □

Principle 9421.20 (Frame-centered reading)**.** *The compressed reading of the chapter is that a focused record does not move through a fixed completed world as a detachable subject. It selects adjacent frame rows, receives complete frame-relative sections, rewrites the prior ledger into the selected frame, admits residual information orthogonally, reads multiplicative accumulation through a logarithmic coordinate, and may keep visible phase fixed while compact prime-register fibre data vary behind the solenoid projection.*

9421.6 Constructor Certificate for OCLSD

Concretized in §9114: Finite constructor certificate for observer-centered orthogonal ledger steps.

The observer-centered dynamics can be read as a constructor certificate: once the frame selector, Hilbert bundle, reshaping rows, closed ledger subspaces, and displayed successor sections are supplied, the orthogonal ledger clause is forced by the Hilbert projection theorem. The section step is therefore not an additional existence premise; it is the output of a finite list of typed rows.

Definition 9421.21 (OCLSD constructor packet)**.** *An* OCLSD constructor packet *consists of a reference-frame graph $\mathcal{R}$, a world Hilbert bundle*

$$\{\mathcal{H}_R\}_{R \in V_{\mathcal{R}}},$$

a selector $n \mapsto R_n$ with

$$R_{n+1} \sim R_n,$$

isometric reshaping rows

$$\Phi_{n+1,n} := \Phi_{R_{n+1} \leftarrow R_n} : \mathcal{H}_{R_n} \to \mathcal{H}_{R_{n+1}},$$

closed ledger subspaces

$$\mathcal{L}_n \subseteq \mathcal{H}_{R_n},$$

and displayed world-sections

$$B_n \in \mathcal{H}_{R_n}.$$

For each stage, it defines the rewritten ledger

$$\mathcal{L}_n^{\uparrow} := \Phi_{n+1,n}(\mathcal{L}_n),$$

the orthogonal projection

$$P_n^{\uparrow} : \mathcal{H}_{R_{n+1}} \to \mathcal{L}_n^{\uparrow},$$

the projection residual

$$\eta_{n+1} := B_{n+1} - P_n^{\uparrow} B_{n+1},$$

the residual line

$$E_{n+1} := \overline{\mathrm{span}}\{\eta_{n+1}\},$$

and the accepted ledger

$$\mathcal{L}_{n+1} := \mathcal{L}_n^{\uparrow} \oplus E_{n+1}.$$

Lemma 9421.22 (Rewritten ledgers are closed)**.** *In an OCLSD constructor packet, each rewritten ledger*

$$\mathcal{L}_n^{\uparrow} = \Phi_{n+1,n}(\mathcal{L}_n)$$

is a closed subspace of $\mathcal{H}_{R_{n+1}}$.

Proof. The ledger $\mathcal{L}_n$ is closed in a Hilbert space, hence complete in the inherited norm. The reshaping row $\Phi_{n+1,n}$ is isometric, so its restriction to $\mathcal{L}_n$ is an isometry onto $\mathcal{L}_n^{\uparrow}$. An isometric image of a complete metric space inside a Hilbert space is complete, and a complete linear subspace of a Hilbert space is closed. □

Theorem 9421.23 (Projection forces the orthogonal section step)**.** *Every OCLSD constructor packet satisfies*

$$E_{n+1} \perp \mathcal{L}_n^{\uparrow}$$

and hence forms an orthogonal section step in the sense of Definition 9421.7.

Proof. By Definition 9421.22, the subspace $\mathcal{L}_n^{\uparrow}$ is closed. Therefore the orthogonal projection $P_n^{\uparrow}$ is available. The Hilbert projection theorem gives

$$B_{n+1} - P_n^{\uparrow} B_{n+1} \perp \mathcal{L}_n^{\uparrow}.$$

The left side is exactly η_{n+1} by Definition 9421.21. Taking the closed span of this residual preserves orthogonality to $\mathcal{L}_n^{\uparrow}$, so

$$E_{n+1} \perp \mathcal{L}_n^{\uparrow}.$$

The accepted ledger is the displayed orthogonal sum in the constructor packet, so it is an orthogonal section step. □

Corollary 9421.24 (Old ledger inclusion and residual separation)**.** *For every OCLSD constructor packet,*

$$\mathcal{L}_n^{\uparrow} \subseteq \mathcal{L}_{n+1}$$

and

$$E_{n+1} \cap \mathcal{L}_n^{\uparrow} = \{0\}.$$

If $B_n \in \mathcal{L}_n$, then

$$\Phi_{n+1,n}(B_n) \in \mathcal{L}_n^{\uparrow}$$

and the residual row η_{n+1} is orthogonal to the rewritten copy of B_n.

Proof. The inclusion is immediate from

$$\mathcal{L}_{n+1} = \mathcal{L}_n^{\uparrow} \oplus E_{n+1}.$$

The intersection is trivial because the sum is orthogonal. If $B_n \in \mathcal{L}_n$, then applying $\Phi_{n+1,n}$ places $\Phi_{n+1,n}(B_n)$ in $\Phi_{n+1,n}(\mathcal{L}_n)$, which is $\mathcal{L}_n^{\uparrow}$. The residual orthogonality follows from Definition 9421.23. □

Corollary 9421.25 (Pythagorean constructor row)**.** *If*

$$B_{n+1} = \Phi_{n+1,n}(B_n) + \eta_{n+1}$$

and $B_n \in \mathcal{L}_n$, then

$$\|B_{n+1}\|^2 = \|B_n\|^2 + \|\eta_{n+1}\|^2.$$

Proof. By Definition 9421.24, the two displayed summands are orthogonal. The Hilbert-space Pythagorean identity gives

$$\|B_{n+1}\|^2 = \|\Phi_{n+1,n}(B_n)\|^2 + \|\eta_{n+1}\|^2.$$

The reshaping row is isometric, so

$$\|\Phi_{n+1,n}(B_n)\| = \|B_n\|.$$

□

Theorem 9421.26 (Constructor certificate for observer-centered dynamics)**.** *An OCLSD constructor packet determines, at every displayed stage, the following certified rows:*

$$R_{n+1} \sim R_n,$$

$$B_{n+1} \in \mathcal{H}_{R_{n+1}},$$

$$\mathcal{L}_n^{\uparrow} = \Phi_{n+1,n}(\mathcal{L}_n),$$

$$\eta_{n+1} = B_{n+1} - P_n^{\uparrow} B_{n+1},$$

$$\mathcal{L}_{n+1} = \mathcal{L}_n^{\uparrow} \oplus E_{n+1}, \qquad E_{n+1} \perp \mathcal{L}_n^{\uparrow}.$$

Thus the object called OCLSD is the constructor output of its typed rows: adjacent frame selection gives a complete successor section, and projection against the rewritten ledger gives the orthogonal residual row.

Proof. The adjacency and section-typing rows are part of Definition 9421.21. The rewritten ledger, projection residual, residual line, and accepted ledger are defined there as well. Orthogonality of the residual line is Definition 9421.23. The complete-section reading is Definition 9421.8. □

Chapter 9422

Quantum and Relativistic Section-Bundle Boundary

This chapter records a bundle reading of the observer-centered section dynamics of chapter 9421. It does not derive quantum mechanics, general relativity, or quantum gravity. Its purpose is to state the interface boundary in which those theories would enter BEDC: quantum grammar occupies the vertical fibre of a chosen frame, while relativistic grammar occupies the horizontal comparison of neighbouring complete sections.

9422.1 Vertical and Horizontal Faces

Definition 9422.1 (Section-bundle packet)**.** *A* section-bundle packet *consists of a reference-frame base $\mathcal{R}$, a world Hilbert bundle*

$$\{\mathcal{H}_R\}_{R\in V_{\mathcal{R}}},$$

solenoid phase fibres $\Sigma_{S,R}$ over displayed frame rows, frame reshaping rows

$$\Phi_{R'\leftarrow R} : \mathcal{H}_R \to \mathcal{H}_{R'},$$

and ledger rows recording projection, residual, visible readout, and transport data. Its typical fibre over R is read schematically as

$$\mathcal{H}_R \otimes \Sigma_{S,R}.$$

The pair-indexed notation $\Phi_{R'\leftarrow R}$ is well-defined only when the reshaping data is path-independent: either the comparison base is thin (at most one comparison row between any two frames) or a canonical transport selector is fixed. In general there may be several distinct comparison rows from R to R', so an honest treatment indexes the reshaping by the comparison morphism, Φ_f, and a pair-indexed $\Phi_{R'\leftarrow R}$ silently assumes away holonomy. This well-definedness dichotomy is discharged as a finite structural fact:

Lean4 (statement) `BEDC.Derived.Visions.ObservationSectionCoherence.pairIndexed wellDefined iff thin or canonical`

Definition 9422.2 (Vertical quantum face)**.** *The* vertical quantum face *of a section-bundle packet is the collection of rows internal to one chosen frame fibre:*

$$\psi_R \in \mathcal{H}_R, \qquad \widetilde{t}_R \in \Sigma_{S,R},$$

orthogonal axis packets, phase rows, projection rows, Born-weight rows when displayed, and section-style measurement updates. It is the fibre-internal grammar already constrained by section 9414.12.

Definition 9422.3 (Horizontal relativistic face)**.** *The* horizontal relativistic face *of a section-bundle packet is the collection of rows comparing neighbouring complete sections:*

$$R' \sim R, \qquad \Phi_{R' \leftarrow R}, \qquad B_{R'} - \Phi_{R' \leftarrow R}(B_R),$$

together with visible stability bounds, inter-history invariants, maximum-rate rows when displayed, and any continuous connection or curvature row introduced over the frame base. It is not a physical signal inside one fibre; it is the comparison discipline between frame-relative sections.

Principle 9422.4 (Quantum vertical, relativity horizontal)**.** *In this packet language, quantum mechanics is read as vertical fibre dynamics: phase, superposition, projection, and orthogonal residual admission inside a chosen frame. Relativistic geometry is read as horizontal base geometry: connection, metric, visible causal rate, and curvature governing how neighbouring complete sections are compared.*

Concretized in §9184: Finite section-bundle packet for vertical fibre rows and horizontal comparison rows.

9422.2 Quantum Fibre Reading

Definition 9422.5 (Frame-relative quantum state)**.** *A* frame-relative quantum state *is a state row*

$$\psi_R \in \mathcal{H}_R$$

together with the phase and ledger rows required by the chosen Hilbert-style axis packet. If a displayed orthogonal family

$$E = \{e_a : a \in A\}$$

is available, the state may be written at that resolution as

$$\psi_R = \sum_{a \in A} c_a e_a.$$

The basis elements are admissible readout directions inside the selected frame fibre, not absolute local facts outside the packet.

Principle 9422.6 (Unitary evolution is log-time phase transport)**.** *When a multiplicative accumulation row*

$$A_{n+1} = \lambda_{n+1} A_n$$

is read through

$$T_n = \log A_n,$$

a frame-fibre Schrödinger row is a log-time phase transport row:

$$i\hbar \frac{\partial \psi}{\partial T} = H\psi.$$

Equivalently, on the multiplicative scale,

$$i\hbar A \frac{\partial \psi}{\partial A} = H\psi.$$

This is a coordinate reading of the time parameter. It does not supply the Hamiltonian, Hilbert-space physical laws, or empirical calibration rows.

Definition 9422.7 (Measurement as fibre projection with residual ledger)**.** *For a displayed projection family $\{P_a\}$ in frame R, a measurement read at outcome a has the section-style form*

$$\psi_R \longmapsto \frac{P_a\psi_R}{\|P_a\psi_R\|} \oplus \eta_a$$

when the denominator condition, post-outcome carrier, residual row, and ledger replay are displayed. The residual must satisfy the appropriate orthogonality row against the prior accepted ledger.

Principle 9422.8 (Collapse is not a propagation row)**.** *The section-style update of Definition 9422.7 is a licensed update of the frame-relative description. It is not, by itself, a controllable signal propagating inside the selected world-section. A claim of physical signalling requires an internal propagation row of the kind refused by Definition 9421.5 unless a separate bridge is supplied.*

Principle 9422.9 (Entanglement as non-product section residue)**.** *An entangled pair is read as a non-product residue of one complete section, not as two independently complete worlds joined by a hidden signal. This refines Definition 9414.258: local outcomes may be displayed, but the residual fibre need not split as a product of local residual fibres.*

9422.3 Relativistic Base Reading

Definition 9422.10 (Residual metric candidate)**.** *For neighbouring frame rows $R' \sim R$ and a section B_R, define the horizontal residual row*

$$\eta_R(R') := B_{R'} - \Phi_{R' \leftarrow R}(B_R)$$

whenever both sections and the reshaping row are displayed. A residual metric candidate *is a visible quadratic read*

$$g_R(\delta R, \delta R) := \langle \eta_R(\delta R), \eta_R(\delta R) \rangle_{\text{vis}}$$

after the packet supplies the visible projection, signature convention, and locality certificate needed for the read.

Principle 9422.11 (Light-cone bounds are internal propagation bounds)**.** *Relativistic maximum-rate bounds constrain visible propagation rows inside or between displayed histories. They do not constrain bare frame rewriting as a coordinate or ledger comparison. Thus a global rewrite of a description is not a superluminal signal unless it also supplies a controllable internal propagation row.*

Definition 9422.12 (Frame connection and curvature row)**.** *In a continuous frame-base reading, a frame connection row is a comparison operator ∇_μ transporting fibre data over the base. Its curvature row is the noncommutativity*

$$[\nabla_\mu, \nabla_\nu] = \mathcal{R}_{\mu\nu}.$$

The row says that transport around an infinitesimal loop can return with nontrivial holonomy. It is the horizontal obstruction to globally flattening the frame reshaping discipline.

Principle 9422.13 (Equivalence principle as local gauge flattening)**.** *The local equivalence-principle reading is that near a frame row R one may choose a local gauge in which the first-order reshaping row is trivial:*

$$\Phi_{R+dR \leftarrow R} = I + O(dR^2).$$

Curvature remains visible through loop holonomy; local gauge flattening does not erase the horizontal curvature row.

9422.4 Unified Boundary

Definition 9422.14 (Combined connection row)**.** *A speculative combined connection row over a section-bundle packet has the schematic form*

$$D_\mu := \partial_\mu + \Gamma_\mu + \frac{i}{\hbar} A_\mu + \mathcal{A}_\mu^{\mathrm{sol}},$$

where Γ_μ is a frame-reshaping connection, A_μ is a visible phase gauge row, and $\mathcal{A}_\mu^{\mathrm{sol}}$ is a solenoid prime-register fibre connection. Its curvature decomposes schematically as

$$[D_\mu, D_\nu] = \mathcal{R}_{\mu\nu} + \frac{i}{\hbar} F_{\mu\nu} + \mathcal{F}_{\mu\nu}^{\mathrm{sol}}.$$

This is an interface target, not a closed physical theory.

Principle 9422.15 (Conflict of times becomes a bundle-coordinate problem)**.** *The usual tension between a fixed quantum time parameter and dynamical relativistic time is read here as a coordinate mismatch. The quantum time row is a log-time parameter along a selected fibre evolution; the relativistic time row is part of the visible geometry of the frame base. Neither is an external absolute time object in the packet.*

Theorem 9422.16 (Section-bundle separation of quantum and relativistic roles)**.** *In any section-bundle packet, vertical rows can state phase, projection, Born-weight, and non-product residue obligations without specifying a base curvature row. Horizontal rows can state frame reshaping, visible stability, maximum-rate, metric-candidate, and curvature obligations without choosing a Born-weight layer. A physical unification claim therefore requires bridge rows relating the two faces; it is not obtained merely by naming both in one packet.*

Proof. The vertical face of Definition 9422.2 is defined using rows inside one fibre over a frame. Those rows do not include the horizontal connection or curvature data of Definition 9422.3. Conversely, the horizontal face is defined from neighbouring frame comparisons and visible stability data; those rows do not include a Born-weight layer or a measurement-update section. Therefore each face is well typed without the other. A unification claim needs additional compatibility rows, such as the combined connection target of Definition 9422.14. □

Principle 9422.17 (Boundary status)**.** *The chapter licenses one compressed interpretation: quantum mechanics uses the vertical grammar of phase, superposition, projection, and orthogonal residual ledger inside a selected frame fibre; general relativity uses the horizontal grammar of frame comparison, connection, metric readout, and curvature over the base. Quantum gravity would be a certificate showing that these grammars share one ledgered connection discipline with empirical prediction rows.*

Chapter 9423

Innovation-Soldering Metric Interface

chapter 9422 separates the vertical quantum face from the horizontal relativistic face of an observer-centered section bundle. That separation is intentionally weak: it says which rows may coexist and which bridge rows are still missing. The present chapter records the stronger interface target. A mere product bundle or direct-sum connection does not couple the rows. The first genuine coupling is a soldering row: the orthogonal innovation of the ledger determines a vielbein-like one-form, and that one-form determines the metric.

9423.1 Product Rows and Soldering Rows

Definition 9423.1 (Product-level coexistence)**.** *A section-bundle description is at* product level *when its total carrier and comparison row have the form*

$$\mathcal{K}_R \simeq \mathcal{H}_R \otimes L^2(\Sigma_{S,R})$$

and

$$D_\mu = D_\mu^{\mathcal{H}} + D_\mu^{\Sigma} + D_\mu^{\text{base}},$$

so that its displayed curvature is the sum of sector curvatures after the cross commutators have been declared absent or untyped. Product level is a coexistence statement. It is not yet a dynamical coupling between quantum ledger rows, solenoid rows, and metric rows.

Definition 9423.2 (Soldering-level coupling)**.** *A section-bundle description is at* soldering level *when the metric row is not supplied independently of the ledger field. Instead, there is an innovation one-form*

$$\Theta \in \Omega^1(M; \mathcal{E})$$

derived from the ledger field, and the metric is read as

$$g_{\mu\nu} = G_{AB}\Theta_\mu^A \Theta_\nu^B.$$

The coupling is genuine because a change in the ledger field changes Θ, and a change in Θ changes the metric row.

Principle 9423.3 (Product connection is only a boundary row)**.** *The combined connection row of Definition 9422.14 is a useful boundary syntax, but it is not by itself a unification theorem. If the curvature decomposes as a sectorwise sum, then the rows have been placed in one ambient bookkeeping space without proving that one sector constrains the other. A unification claim requires a row such as Definition 9423.2, where the metric depends on the ledger innovation.*

9423.2 Continuous Innovation Form

Definition 9423.4 (Continuous ledger field)**.** *Let M be a continuous frame base and let*

$$\mathcal{H} \to M$$

be a Hilbert ledger bundle. A continuous ledger field consists of a section

$$B \in \Gamma(\mathcal{H}),$$

a closed ledger subbundle

$$\mathcal{L} \subseteq \mathcal{H},$$

and the orthogonal projection

$$P_{\mathcal{L}} : \mathcal{H} \to \mathcal{L}.$$

The innovation projector is

$$Q_{\mathcal{L}} := I - P_{\mathcal{L}}.$$

Definition 9423.5 (Innovation one-form)**.** *Let D_Φ be a frame-reshaping connection on $\mathcal{H}$. The* innovation one-form *of the ledger field is*

$$\Theta := Q_{\mathcal{L}} D_\Phi B.$$

Equivalently, for $u \in T_x M$,

$$\Theta_x(u) = Q_{\mathcal{L},x}\big(D_{\Phi,u} B\big)(x).$$

Its target bundle is the innovation bundle

$$\mathcal{E} := \mathcal{L}^\perp.$$

Definition 9423.6 (Innovation metric row)**.** *Assume that the innovation bundle $\mathcal{E}$ carries a displayed nondegenerate symmetric bilinear form*

$$G_x : \mathcal{E}_x \times \mathcal{E}_x \to \mathbb{R}.$$

The innovation metric row is

$$g_x(u, v) := G_x(\Theta_x(u), \Theta_x(v)).$$

In local notation,

$$g_{\mu\nu} = G_{AB} \Theta^A_\mu \Theta^B_\nu.$$

Principle 9423.7 (Positive Hilbert norm is not the Lorentz metric)**.** *The Hilbert inner product controls orthogonality and projection in the ledger. It cannot by itself be the Lorentz metric, because it is positive definite. The metric row therefore requires the additional signature data G on the effective innovation bundle. Without that row, one may define a positive quadratic size of an innovation, but not a Lorentzian spacetime metric.*

Theorem 9423.8 (Innovation one-form gives a Lorentz metric)**.** *Assume that, for every $x \in M$, the map*

$$\Theta_x : T_x M \to \mathcal{E}_x$$

is injective and that G_x restricted to $\Theta_x(T_x M)$ is nondegenerate with Lorentz signature. Then

$$g_x(u, v) = G_x(\Theta_x(u), \Theta_x(v))$$

defines a Lorentz metric on M.

Proof. The map Θ_x is linear and G_x is bilinear, so g_x is bilinear. Symmetry of g_x follows from symmetry of G_x. If

$$g_x(u, v) = 0$$

for every $v \in T_xM$, then

$$G_x(\Theta_x(u), \Theta_x(v)) = 0$$

for every v. Nondegeneracy of G_x on $\Theta_x(T_xM)$ gives $\Theta_x(u) = 0$. Injectivity of Θ_x then gives $u = 0$. Thus g_x is nondegenerate, and its signature is the Lorentz signature displayed on the image subspace. □

Theorem 9423.9 (Metric variation factors through ledger innovation)**.** *For*

$$g_{\mu\nu} = G_{AB}\Theta^A_\mu\Theta^B_\nu, \qquad \Theta = Q_{\mathcal{L}}D_\Phi B,$$

a ledger variation $B \mapsto B + \delta B$ induces

$$\delta g_{\mu\nu} = (\delta G_{AB})\Theta^A_\mu\Theta^B_\nu + G_{AB}(\delta\Theta^A_\mu)\Theta^B_\nu + G_{AB}\Theta^A_\mu(\delta\Theta^B_\nu),$$

where

$$\delta\Theta = \delta(Q_{\mathcal{L}}D_\Phi B).$$

If G is fixed, the variation reduces to the symmetric cross term between $\delta\Theta$ and Θ.

Proof. Differentiate the defining expression

$$g_{\mu\nu} = G_{AB}\Theta^A_\mu\Theta^B_\nu.$$

The product rule gives the displayed three terms. The final statement is the same formula with $\delta G = 0$. □

9423.3 Tick Reflection and Pythagorean Coding Closure

Definition 9423.10 (Tick covector and spatial innovation image)**.** *Let*

$$A : M \to \mathbb{R}_{>0}$$

be a displayed multiplicative scale row. Its logarithmic tick coordinate and tick covector are

$$T := \log A, \qquad \tau := dT.$$

At a point x with $\tau_x \neq 0$, set

$$S_x := \ker \tau_x \subset T_xM.$$

The spatial innovation image is

$$E^S_x := \Theta_x(S_x) \subset \mathcal{E}_x.$$

This row is part of the Pythagorean coding closure reading: the scale ledger selects the time co-direction, while the innovation map supplies the orthogonal readout surface.

Definition 9423.11 (Tick residual line and reflection)**.** *Assume* $\operatorname{rank}(\Theta_x|_{S_x}) = 3$*, and let* Π_S^E *be the Hilbert orthogonal projection onto* E_x^S*. For any* $X_x \in T_xM$ *satisfying*

$$\tau_x(X_x) = 1,$$

define the tick residual

$$r_T(x) := (I - \Pi_S^E)\Theta_x(X_x).$$

When $r_T(x) \neq 0$*, the tick residual line is*

$$E_x^T := \mathbb{R}\, r_T(x),$$

and the tick reflection on

$$E_x^4 := E_x^T \oplus E_x^S$$

is

$$J_T := \Pi_T^E - \Pi_S^E,$$

where Π_T^E *is the Hilbert orthogonal projection onto* E_x^T*.*

Lemma 9423.12 (Tick residual line is independent of the lift)**.** *The vector* $r_T(x)$ *of Definition 9423.11 does not depend on the choice of* X_x *satisfying* $\tau_x(X_x) = 1$*.*

Proof. If X_x and X'_x both satisfy

$$\tau_x(X_x) = \tau_x(X'_x) = 1,$$

then

$$\tau_x(X_x - X'_x) = 0.$$

Hence $X_x - X'_x \in S_x$, so

$$\Theta_x(X_x - X'_x) \in E_x^S.$$

Applying $I - \Pi_S^E$ kills this vector, and therefore

$$(I - \Pi_S^E)\Theta_x(X_x) = (I - \Pi_S^E)\Theta_x(X'_x).$$

□

Definition 9423.13 (Lorentzized innovation tensor)**.** *On the four-dimensional innovation image* E_x^4*, define*

$$\mathfrak{Q}_x^T(u, v) := \langle \Theta_x(u), J_T\Theta_x(v)\rangle.$$

Its real and imaginary reads are

$$g_x^T(u, v) := \operatorname{Re}\mathfrak{Q}_x^T(u, v),$$

and

$$F_x^\eta(u, v) := 2\operatorname{Im}\mathfrak{Q}_x^T(u, v).$$

The first row is the Pythagorean metric read; the second row is the phase-curvature read carried by the same innovation tensor.

Theorem 9423.14 (Tick-reflected Gram row has Lorentz signature)**.** *Assume* $\dim M = 4$*,* $\tau_x \neq 0$*,*

$$\operatorname{rank}(\Theta_x|_{\ker \tau_x}) = 3,$$

and $r_T(x) \neq 0$*. Then*

$$g_x^T(u, v) = \operatorname{Re}\langle \Theta_x(u), J_T\Theta_x(v)\rangle$$

is a nondegenerate symmetric bilinear form on T_xM *with signature* $(1, 3)$*.*

Proof. Choose any X_x with $\tau_x(X_x) = 1$. Since $\Pi_S^E \Theta_x(X_x)$ lies in

$$E_x^S = \Theta_x(S_x),$$

there is $s_X \in S_x$ with

$$\Theta_x(s_X) = \Pi_S^E \Theta_x(X_x).$$

Set

$$\mathcal{T}_x := X_x - s_X.$$

Then $\tau_x(\mathcal{T}_x) = 1$ and

$$\Theta_x(\mathcal{T}_x) = r_T(x).$$

Every $u \in T_x M$ decomposes as

$$u = \tau_x(u)\mathcal{T}_x + s, \qquad s \in S_x.$$

Thus

$$\Theta_x(u) = \tau_x(u) r_T + \Theta_x(s),$$

with $r_T \perp E_x^S$. Since J_T is $+I$ on E_x^T and $-I$ on E_x^S,

$$g_x^T(u,u) = \tau_x(u)^2 \|r_T\|^2 - \|\Theta_x(s)\|^2.$$

The first coefficient is positive because $r_T \neq 0$. The second term is negative definite on the three-dimensional spatial image because $\Theta_x|_{S_x}$ has rank 3. Symmetry follows from self-adjointness of J_T. Hence the signature is $(1,3)$. □

Theorem 9423.15 (Real and imaginary rows share one innovation source)**.** *If J_T is self-adjoint, then*

$$\mathfrak{Q}_x^T(v,u) = \overline{\mathfrak{Q}_x^T(u,v)}.$$

Consequently $g_x^T = \operatorname{Re} \mathfrak{Q}_x^T$ is symmetric and $F_x^\eta = 2 \operatorname{Im} \mathfrak{Q}_x^T$ is antisymmetric.

Proof. By definition,

$$\mathfrak{Q}_x^T(u,v) = \langle \Theta_x(u), J_T \Theta_x(v) \rangle.$$

Self-adjointness of J_T gives

$$\langle \Theta_x(v), J_T \Theta_x(u) \rangle = \overline{\langle \Theta_x(u), J_T \Theta_x(v) \rangle}.$$

Taking real and imaginary parts gives the stated symmetry and antisymmetry rows. □

Principle 9423.16 (Pythagorean coding closure boundary)**.** *Pythagorean coding closure is the local synthesis*

$$\Theta = Q_{\mathcal{L}} D_\Phi B, \qquad T = \log A, \qquad J_T, \qquad \mathfrak{Q}_{\mu\nu}^T = \langle Q_{\mathcal{L}} D_{\Phi,\mu} B, J_T Q_{\mathcal{L}} D_{\Phi,\nu} B \rangle.$$

Its real part gives the triangle/metric row, and its imaginary part gives the circle/phase-curvature row. Prime and solenoid ledgers affect this geometry only when they enter B, D_Φ, $Q_{\mathcal{L}}$, or the tick scale A; otherwise they remain hidden fibre data without metric-level force.

9423.4 Discrete Stencils and Cartan Boundary

Definition 9423.17 (Discrete innovation stencil)**.** *Let x be a frame vertex and let*

$$e_a : x \to x_a \qquad (a = 0, 1, 2, 3)$$

be four displayed independent local edges. With reshaping rows

$$\Phi_{x_a \leftarrow x} : \mathcal{H}_x \to \mathcal{H}_{x_a},$$

define the edge innovation

$$\eta_a(x) := Q_{x \to x_a}(B_{x_a} - \Phi_{x_a \leftarrow x} B_x).$$

The discrete innovation Gram row is

$$g_{ab}(x) := G(\eta_a(x), \eta_b(x)).$$

When this matrix is nondegenerate with Lorentz signature, it is the local discrete metric row at x.

Principle 9423.18 (One tick is not a four-dimensional metric)**.** *A single OCLSD residual supplies one innovation direction. A four-dimensional metric requires a local stencil of independent directions or a continuous tangent map $\Theta_x : T_x M \to \mathcal{E}_x$. Thus the passage from ledger innovation to metric is not obtained by renaming one residual as a metric. It requires the displayed soldering rank and signature rows.*

Definition 9423.19 (Innovation-soldered Cartan row)**.** *An innovation-soldered Cartan row is a schematic connection*

$$\mathcal{A} = \omega^{ab} J_{ab} + \ell^{-1} \Theta^a P_a + A + \mathcal{A}^\Sigma,$$

where Θ is the innovation one-form of Definition 9423.5. Its curvature is

$$\mathcal{F} = d\mathcal{A} + \mathcal{A} \wedge \mathcal{A}.$$

In the Cartan sector, the displayed torsion row is

$$T^a = D_\omega \Theta^a,$$

and the de Sitter-style curvature row contains

$$R^{ab} + \ell^{-2} \Theta^a \wedge \Theta^b.$$

These rows are the schematic boundary where soldering produces cross-terms absent from a product connection.

Principle 9423.20 (Solenoid coupling must enter the innovation row)**.** *The solenoid fibre is geometrically active only if its prime-register rows enter the innovation one-form, the connection used to define it, or the signature data used to read it. A hidden fibre that remains visible only through*

$$\pi_S : \Sigma_S \to \mathbb{T}$$

is still kernel-blind at the metric level. A geometry-facing solenoid coupling must therefore be represented by rows of the form

$$D_{\Phi,\Sigma} B \qquad \text{or} \qquad \Theta(B, Z_\Sigma),$$

not merely by adjoining $L^2(\Sigma_S)$ to the fibre.

9423.5 Twisted-Holonomy Backreaction: A Degenerate Channel

Principle 9423.21 (Rank-one twisted-holonomy backreaction is metric-degenerate)**.** *Consider the schematic twisted-holonomy backreaction channel on a two-torus $\mathbb{T}^2$, in which a solenoid ledger is asked to react back on an emergent metric row through a single innovation displacement $\Delta = (\Delta x, \Delta y)$ read against a scalar weight $w_1 w_2$. The displayed metric components take the outer-product form*

$$g_{\mu\nu} = w_1 w_2 \, \Delta_\mu \Delta_\nu,$$

so that

$$g_{xx} = w_1 w_2 \, \Delta x \, \Delta x, \quad g_{xy} = w_1 w_2 \, \Delta x \, \Delta y, \quad g_{yx} = w_1 w_2 \, \Delta y \, \Delta x, \quad g_{yy} = w_1 w_2 \, \Delta y \, \Delta y.$$

This is the outer product of the two-vector Δ with itself scaled by $w_1 w_2$; its rank is at most one, hence the two-by-two determinant vanishes identically:

$$\det g = g_{xx} g_{yy} - g_{xy} g_{yx} = (w_1 w_2)^2 (\Delta x \, \Delta x \, \Delta y \, \Delta y - \Delta x \, \Delta y \, \Delta y \, \Delta x) = 0.$$

A degenerate row ($\det g = 0$) admits no inverse, hence no curvature invariant, so this schematic channel does not read out a nondegenerate two-dimensional geometry. This is an honest negative and falsifiable outcome, not a suppressed obstacle: the rank-one twisted-holonomy backreaction row is exactly the failure mode that Definition 9423.18 anticipates, where a single innovation direction is renamed as a metric. Because the degeneracy is forced by the outer-product shape rather than by any prime-register content of the solenoid, the channel does not privilege the solenoid structure: the same rank-one row is produced by a continuous $\mathrm{U}(1)$ flat bundle carrying the same single displacement. The wording here is schematic and asserts no physical metric; it records only that this particular backreaction row cannot supply the nondegenerate signature demanded by Definition 9423.6.

Lean4 (statement) `BEDC.Derived.Visions.backreaction metric det zero`

Principle 9423.22 (Two independent drivers restore metric nondegeneracy)**.** *The rank-one degeneracy of Definition 9423.21 is not the end of the schematic backreaction story but its exact lower boundary; the present principle is the constructive companion that locates where a nondegenerate row first becomes available. Replace the single innovation displacement by two schematic relative-holonomy drivers $\Delta^{(1)} = (\Delta_x^{(1)}, \Delta_y^{(1)})$ and $\Delta^{(2)} = (\Delta_x^{(2)}, \Delta_y^{(2)})$ read against positive scalar weights w_1, w_2, and form the weighted sum of the two outer-product rows:*

$$g_{\mu\nu} = w_1 \, \Delta_\mu^{(1)} \Delta_\nu^{(1)} + w_2 \, \Delta_\mu^{(2)} \Delta_\nu^{(2)}.$$

Each summand is individually rank-one, but the two-by-two determinant of the sum is governed by the Cauchy–Binet identity and collapses to a single wedge square:

$$\det g = w_1 w_2 \, (\Delta^{(1)} \wedge \Delta^{(2)})^2, \qquad \Delta^{(1)} \wedge \Delta^{(2)} = \Delta_x^{(1)} \Delta_y^{(2)} - \Delta_y^{(1)} \Delta_x^{(2)}.$$

With w_1, w_2 nonzero this determinant vanishes if and only if the wedge does, that is, if and only if the two drivers are linearly dependent; whenever $\Delta^{(1)}$ and $\Delta^{(2)}$ are linearly independent the row is nondegenerate, admits an inverse, and can carry a curvature invariant. This delineates the precise boundary of what a schematic backreaction channel can hand to geometry: a single driver is forced to be degenerate (Definition 9423.21), while any two linearly independent relative-holonomy drivers already suffice for a nondegenerate emergent two-dimensional row — one is never enough, two independent are always enough. The wording remains schematic and asserts no physical metric

or dynamics; it records only the finite algebraic fact that the wedge of two independent drivers is exactly the resource the rank-one channel was missing, so the honest negative of Definition 9423.21 is paired with an equally honest constructive sufficiency at the two-driver threshold.
Lean4 (statement) `BEDC.Derived.Visions.two driver det eq weighted wedge sq`

Principle 9423.23 (Innovation-soldering proof boundary)**.** *The chapter supplies the formal target for a genuine ledger-geometry coupling: construct $\Theta = Q_{\mathcal{L}} D_{\Phi} B$, prove the required rank and Lorentz-signature rows, and then read the metric through $g = G(\Theta, \Theta)$. It does not derive Einstein dynamics, empirical matter coupling, or quantum gravity. Those claims would require field equations and action rows in which variation of the ledger field changes the induced metric and curvature.*

Concretized in §9149: The finite NameCert carrier records the ledger-innovation-projector-soldering-signature-metric interface and its Nontrivial and FieldFaithful instance requirements.

Chapter 9424

Innovation Input-Flow Geometry Boundary

chapter 9423 identifies the orthogonal innovation one-form as the first genuine coupling row between a ledger field and a metric-facing read. This chapter records the adjacent boundary: the positive Hilbert Gram row, the tick-reflected signature row, and the reflection-positive input-flow row are distinct certificates. They may be compatible, but none may be substituted for another without an explicit bridge.

9424.1 Discrete Ledger Source

Definition 9424.1 (One-step ledger comparison)**.** *A one-step ledger comparison consists of Hilbert rows $\mathcal{H}_n$, $\mathcal{H}_{n+1}$, ledger subspaces $\mathcal{L}_n \subseteq \mathcal{H}_n$, $\mathcal{L}_{n+1} \subseteq \mathcal{H}_{n+1}$, a section row $B_n \in \mathcal{H}_n$, and an isometric reshaping row*

$$\Phi_{n+1,n} : \mathcal{H}_n \to \mathcal{H}_{n+1}.$$

Let $P_n^{\uparrow}$ be the orthogonal projection onto $\Phi_{n+1,n}(\mathcal{L}_n)$. The one-step innovation residual is

$$\eta_{n+1} := B_{n+1} - P_n^{\uparrow} B_{n+1}.$$

Theorem 9424.2 (Orthogonal admission law)**.** *For every one-step ledger comparison,*

$$\eta_{n+1} \perp \Phi_{n+1,n}(\mathcal{L}_n)$$

and

$$\|B_{n+1}\|^2 = \|P_n^{\uparrow} B_{n+1}\|^2 + \|\eta_{n+1}\|^2.$$

If $B_n \in \mathcal{L}_n$ and $P_n^{\uparrow} B_{n+1} = \Phi_{n+1,n} B_n$, then

$$\|B_{n+1}\|^2 = \|B_n\|^2 + \|\eta_{n+1}\|^2.$$

Proof. The residual $B - PB$ of an orthogonal projection is orthogonal to the range of P. The norm identity is the Pythagorean identity for the orthogonal decomposition $B_{n+1} = P_n^{\uparrow} B_{n+1} + \eta_{n+1}$. The final statement uses the isometry of $\Phi_{n+1,n}$. □

Definition 9424.3 (Logarithmic tick read)**.** *For a ledger comparison with $B_n \neq 0$, the logarithmic tick read is*

$$T_n := \log \|B_n\|^2.$$

Under the compatibility row of Definition 9424.2,

$$T_{n+1} - T_n = \log\left(1 + \frac{\|\eta_{n+1}\|^2}{\|B_n\|^2}\right).$$

Theorem 9424.4 (Continuous innovation is the first-order residual)**.** *Let $\mathcal{H} \to M$ be a Hilbert bundle with a metric connection D_Φ, let $\mathcal{L} \subseteq \mathcal{H}$ be a closed ledger subbundle, and let P_x and $Q_x = I - P_x$ be the fibrewise orthogonal projections. Transport $B(x+\varepsilon u)$ back to $\mathcal{H}_x$ along D_Φ. If $B_x \in \mathcal{L}_x$, then*

$$\lim_{\varepsilon\to 0} \varepsilon^{-1} Q_x B(x+\varepsilon u) = Q_x (D_{\Phi,u} B)_x.$$

If $B_x \notin \mathcal{L}_x$, the unscaled term $Q_x B_x$ remains visible, so the scaled residual does not define a linear tangent read.

Proof. The connection gives the first-order expansion

$$B(x+\varepsilon u) = B_x + \varepsilon (D_{\Phi,u} B)_x + O(\varepsilon^2)$$

after transport to $\mathcal{H}_x$. Applying Q_x gives

$$Q_x B(x+\varepsilon u) = Q_x B_x + \varepsilon Q_x (D_{\Phi,u} B)_x + O(\varepsilon^2).$$

The condition $B_x \in \mathcal{L}_x$ is exactly $Q_x B_x = 0$. □

Definition 9424.5 (Quantized refinement edge)**.** *A quantized refinement edge has a finite cell set C_n, a refinement factor $N \geq 2$, an isometric split*

$$e_c \longmapsto N^{-1/2} \sum_{a=0}^{N-1} e_{ca},$$

and a fixed modulus $\varepsilon > 0$ such that each displayed cell coefficient of B_n and B_{n+1} has modulus ε.

Theorem 9424.6 (Refinement residual rate)**.** *For a quantized refinement edge satisfying the compatibility row $P_n^{\uparrow} B_{n+1} = \Phi_{n+1,n} B_n$,*

$$\frac{\|\eta_{n+1}\|^2}{\|B_n\|^2} = N - 1, \qquad T_{n+1} - T_n = \log N.$$

Proof. If $|C_n| = A_n$, then

$$\|B_n\|^2 = \varepsilon^2 A_n, \qquad \|B_{n+1}\|^2 = \varepsilon^2 N A_n.$$

Definition 9424.2 gives $\|\eta_{n+1}\|^2 = \varepsilon^2 (N-1) A_n$. Dividing by $\|B_n\|^2$ gives the residual rate, and the logarithmic identity gives the tick length. □

9424.2 Positive Innovation Gram Row

Definition 9424.7 (Innovation Gram tensor)**.** *Let $\Theta = Q_{\mathcal{L}} D_\Phi B$ be the innovation one-form of Definition 9423.5. Its positive innovation Gram tensor is*

$$\chi_x^\eta(u,v) := \langle \Theta_x(u), \Theta_x(v) \rangle$$

for $u, v \in T_x M$. Its real and imaginary reads are

$$h_x^\eta(u,v) := \operatorname{Re} \chi_x^\eta(u,v), \qquad \Omega_x^\eta(u,v) := 2 \operatorname{Im} \chi_x^\eta(u,v).$$

Theorem 9424.8 (Positive Gram constraint)**.** *For any finite family $u_1, \dots, u_n \in T_xM$, the matrix*

$$\chi^\eta_{ij} := \chi^\eta_x(u_i, u_j)$$

is Hermitian positive semidefinite. Consequently h^η is symmetric positive semidefinite, Ω^η is antisymmetric, and every pair $u, v \in T_xM$ satisfies

$$h^\eta(u,u)h^\eta(v,v) \geq h^\eta(u,v)^2 + \frac{1}{4}\Omega^\eta(u,v)^2.$$

Proof. For coefficients $c_i \in \mathbb{C}$,

$$\sum_{i,j} \overline{c_i}\chi^\eta_{ij}c_j = \left\| \sum_i c_i \Theta_x(u_i) \right\|^2 \geq 0.$$

This proves positive semidefiniteness and Hermitian symmetry. The real and imaginary symmetry rows follow by taking real and imaginary parts. The displayed inequality is the nonnegativity of the 2×2 principal minor associated to u and v. □

Principle 9424.9 (Positive size is not Lorentz signature)**.** *The row χ^η gives a Hilbert-size certificate. Its real part is nonnegative by construction. A Lorentz-facing row therefore requires an additional signature certificate, such as the displayed bilinear form G of Definition 9423.6, the tick reflection J_T of Definition 9423.11, or a reflection-positive input-flow reconstruction as in section 9424.4. The positive Gram row alone cannot carry the time-side sign.*

9424.3 Phase Packets and Curvature Budget

Definition 9424.10 (Constant-modulus phase packet)**.** *A constant-modulus phase packet over a local coordinate window consists of orthonormal component rows e_a, positive weights w_a with $\sum_a w_a = 1$, fixed covectors κ_a, and the ray-ledger field*

$$B(z) = \sum_a c_a e^{i\kappa_a \cdot z} e_a, \qquad |c_a|^2 = w_a,$$

with ledger subbundle $\mathbb{C}B$.

Theorem 9424.11 (Constant-modulus packets give covariance geometry)**.** *For a constant-modulus phase packet,*

$$\chi^\eta_{\mu\nu} = \sum_a w_a \kappa_{a\mu}\kappa_{a\nu} - \bar\kappa_\mu \bar\kappa_\nu, \qquad \bar\kappa_\mu := \sum_a w_a \kappa_{a\mu}.$$

Thus h^η is the covariance matrix of the displayed covectors and $\Omega^\eta = 0$.

Proof. The derivative of the ath component is $i\kappa_{a\mu}c_a e^{i\kappa_a \cdot z}$. Hence

$$\langle B, \partial_\nu B\rangle = i\bar\kappa_\nu, \qquad \langle \partial_\mu B, \partial_\nu B\rangle = \sum_a w_a \kappa_{a\mu}\kappa_{a\nu}.$$

Projecting away the ray $\mathbb{C}B$ subtracts $\bar\kappa_\mu\bar\kappa_\nu$. The result is real, so the imaginary curvature read is zero. □

Corollary 9424.12 (Constant-modulus rank budget)**.** *If a constant-modulus phase packet has r displayed components, then*

$$\operatorname{rank} h^\eta \leq r - 1.$$

A nondegenerate d-dimensional covariance read of this kind therefore requires $r \geq d + 1$.

Proof. The centered covectors $\kappa_a - \bar{\kappa}$ satisfy

$$\sum_a w_a(\kappa_a - \bar{\kappa}) = 0.$$

They span at most an $(r-1)$-dimensional real space, and the covariance matrix is their weighted Gram matrix. □

Example 9424.13 (Simplex covariance witness)**.** *Let $r = d+1$ and choose $\kappa_1, \ldots, \kappa_{d+1} \in \mathbb{R}^d$ as the vertices of a regular simplex centered at 0, with equal weights. Then*

$$\sum_a w_a \kappa_a = 0$$

and

$$\sum_a w_a \kappa_a \kappa_a^{\mathsf{T}} = cI_d$$

for some $c > 0$. Hence the constant-modulus covariance read is nondegenerate. The rank budget of Definition 9424.12 is therefore sharp for this packet class.

Definition 9424.14 (Variable-weight phase packet)**.** *A variable-weight phase packet keeps the fixed covectors κ_a but allows positive smooth weights $w_a(z)$ with $\sum_a w_a(z) = 1$:*

$$B(z) = \sum_a \sqrt{w_a(z)} e^{i\kappa_a \cdot z} e_a.$$

Theorem 9424.15 (Variable weights split metric and curvature)**.** *For a variable-weight phase packet,*

$$h^{\eta}_{\mu\nu} = \frac{1}{4} \sum_a \frac{\partial_\mu w_a \, \partial_\nu w_a}{w_a} + \sum_a w_a(\kappa_{a\mu} - \bar{\kappa}_\mu)(\kappa_{a\nu} - \bar{\kappa}_\nu),$$

and

$$\Omega^{\eta}_{\mu\nu} = \partial_\mu \bar{\kappa}_\nu - \partial_\nu \bar{\kappa}_\mu, \qquad \bar{\kappa}_\mu(z) := \sum_a w_a(z) \kappa_{a\mu}.$$

Thus the real row splits into a Fisher-information row and a covariance row, while the phase-curvature row is the exterior derivative of the mean covector read.

Proof. The ath component derivative is

$$\partial_\mu B_a = \left(\frac{\partial_\mu w_a}{2\sqrt{w_a}} + i\kappa_{a\mu}\sqrt{w_a} \right) e^{i\kappa_a \cdot z}.$$

Since $\sum_a w_a = 1$, one has

$$\langle B, \partial_\nu B \rangle = i\bar{\kappa}_\nu.$$

Expanding $\langle \partial_\mu B, \partial_\nu B \rangle$ and subtracting the ray component gives the displayed real part. The imaginary part is

$$\frac{1}{2} \sum_a \left(\partial_\mu w_a \, \kappa_{a\nu} - \partial_\nu w_a \, \kappa_{a\mu} \right),$$

whose double is $\partial_\mu \bar{\kappa}_\nu - \partial_\nu \bar{\kappa}_\mu$. □

Principle 9424.16 (Curvature requires the correct packet class)**.** *Constant-modulus phase packets carry covariance geometry but no phase curvature. Variable-weight packets can carry an exact mean-covector curvature row. A nontrivial topological phase read therefore cannot be inferred from a bare constant-modulus covariance packet; it requires the displayed packet class to carry the relevant amplitude, winding, or holonomy data.*

Example 9424.17 (Two-component coherent surface)**.** *On a coordinate window with variables (ϑ, φ), consider the ray-ledger field*

$$B(\vartheta, \varphi) = \cos\frac{\vartheta}{2}\, e_0 + e^{i\varphi} \sin\frac{\vartheta}{2}\, e_1.$$

For the ray ledger $\mathbb{C}B$, direct computation gives

$$h^\eta = \frac{1}{4}\, d\vartheta^2 + \frac{1}{4} \sin^2\vartheta\, d\varphi^2$$

and

$$\Omega^\eta = \frac{1}{2} \sin\vartheta\, d\vartheta \wedge d\varphi.$$

Thus

$$\det h^\eta = \frac{1}{4}(\Omega^\eta_{\vartheta\varphi})^2$$

on the regular part. A two-component packet can therefore carry a nondegenerate two-dimensional metric read with phase curvature when the packet class contains the relevant variable amplitude and winding data.

Principle 9424.18 (Rank budgets are packet-relative)**.** *The simplex bound $r \geq d + 1$ belongs to constant-modulus covariance packets, where the real centered covectors are the only rank resource. The coherent-surface example shows that the same numerical bound is not a general law for all ray-ledger packets. The invariant law that survives all packet choices is the positive Gram constraint of Definition 9424.8; sharper rank budgets must state their packet class.*

9424.4 Reflection-Positive Input Flow

Definition 9424.19 (Reversible finite input flow)**.** *A reversible finite input flow consists of a finite state set E, a strictly positive stationary distribution π, and a transition kernel $p(x, y)$ satisfying detailed balance:*

$$\pi(x)p(x, y) = \pi(y)p(y, x).$$

Let P be the induced self-adjoint operator on $L^2(\pi)$.

Definition 9424.20 (Reflection-positive input row)**.** *Let $(d_n)_{n\in\mathbb{Z}}$ be the stationary two-sided chain of Definition 9424.19. Let $\mathcal{K}_+$ be the future algebra generated by $d_1, d_2, \ldots$, and let θ be the reflection sending d_n to d_{1-n}. The input row is reflection-positive when*

$$\langle \theta F, F\rangle \geq 0$$

for every $F \in \mathcal{K}_+$.

Theorem 9424.21 (Reflection positivity is transition positivity)**.** *For $F, G \in \mathcal{K}_+$ define*

$$h_F(x) := \mathbb{E}[F \mid d_1 = x].$$

Then

$$\langle \theta F, G \rangle = \langle h_F, P h_G \rangle_{L^2(\pi)}.$$

Since every $h \in L^2(\pi)$ is h_F for $F = h(d_1)$, reflection positivity holds exactly when P is positive semidefinite.

Proof. Given d_0, the past and future of a Markov chain are conditionally independent. Detailed balance identifies the conditional law of the reflected future with the conditional law read from the boundary state. Thus

$$\mathbb{E}[\overline{\theta F} G] = \sum_x \pi(x) \overline{h_F(x)} (P h_G)(x).$$

The final equivalence follows by taking $F = h(d_1)$. □

Example 9424.22 (Two-state reflection boundary)**.** *Let $E = \{0, 1\}$ with stationary distribution $(1/2, 1/2)$ and transition matrix*

$$P = \begin{pmatrix} 1-q & q \\ q & 1-q \end{pmatrix}.$$

The eigenvalues are 1 and $1 - 2q$. Hence the input row is reflection-positive exactly when $q \le 1/2$. For $q > 1/2$, the function $f(0) = 1$, $f(1) = -1$ satisfies

$$\langle f, Pf \rangle_{L^2(\pi)} = (1 - 2q) \|f\|^2_{L^2(\pi)} < 0,$$

so the reflected inner product is negative on a future observable.

Corollary 9424.23 (Positive Hamiltonian row)**.** *If $0 \preceq P \preceq I$ and the tick length is $\Delta T > 0$, then the nonzero spectral part carries*

$$H := -\frac{1}{\Delta T} \log P \succeq 0, \qquad e^{-\Delta T H} = P.$$

The unitary row e^{-itH} is then obtained by spectral calculus. If P has a negative spectral value, no self-adjoint H can satisfy $e^{-\Delta T H} = P$ on that spectral part.

Proof. The logarithm of a positive contraction is a nonpositive self-adjoint operator on the nonzero spectral subspace, so H is positive and the exponential identity follows. Conversely, the exponential of a self-adjoint operator has positive spectrum. □

Example 9424.24 (Nearest-neighbour diffusion witness)**.** *On a one-dimensional periodic lattice with spacing ℓ, take the translation-invariant transition row with jumps $-\ell$, 0, and $+\ell$ of probabilities q, $1 - 2q$, and q. Its Fourier multiplier is*

$$\widehat{P}(k) = 1 - 2q(1 - \cos(k\ell)).$$

For $q \le 1/4$, this multiplier is nonnegative for every k, so the reflection-positive row is present. With tick length ΔT, the Hamiltonian multiplier satisfies

$$H(k) = -\Delta T^{-1} \log \widehat{P}(k) = \frac{q\ell^2}{\Delta T} k^2 + O(k^4 \ell^4)$$

near $k = 0$. The low-momentum inverse-metric read is therefore the per-tick displacement variance divided by ΔT:

$$g^{xx} = \frac{2q\ell^2}{\Delta T}.$$

Principle 9424.25 (Two metric engines)**.** *The covariance read h^η and the diffusion read g^{xx} arise from different rows. The former is a kinematic Gram construction from $\Theta = Q_{\mathcal{L}} D_\Phi B$; the latter is a low-momentum read of $-\log P$ from an input-flow transition operator. Identifying them is an additional bridge obligation, not a consequence of either row alone.*

9424.5 Interface Separation

Principle 9424.26 (Three non-interchangeable certificates)**.** *The innovation programme carries three separate certificates:*

$$\chi^\eta(u,v) = \langle \Theta(u), \Theta(v) \rangle,$$
$$g^T(u,v) = \mathrm{Re}\langle \Theta(u), J_T \Theta(v) \rangle,$$

and

$$P \succeq 0, \qquad H = -\Delta T^{-1} \log P.$$

The first is the positive Hilbert Gram certificate, the second is the pointwise tick-reflected signature certificate, and the third is the input-flow certificate for reflection positivity and Hamiltonian reconstruction. A physical bridge must state which certificate it uses and which comparison rows identify them.

Principle 9424.27 (Prime-register data must reach a visible row)**.** *Prime-register or solenoid data affect the geometry-facing interface only when they enter B, D_Φ, $Q_{\mathcal{L}}$, the scale row used for ΔT, or the transition operator P. A prime fibre that remains visible only through a projected phase is kernel-blind at the metric level, as in Definition 9423.20.*

Principle 9424.28 (Geometry boundary)**.** *The boundary recorded here is a translation interface from ledger rows to metric-facing and dynamics-facing certificates. It does not assert field equations, empirical matter coupling, a completed quantum-gravity model, or a proof that the tick-reflected and reflection-positive routes determine the same propagation cone. Those are bridge obligations, not consequences of the positive innovation Gram row.*

Chapter 9425

Innovation Input Lineage and Self-Reference Boundary

chapter 9424 records the positive Gram, tick-reflection, and reflection-positive transition certificates of an input-flow geometry. This chapter records the adjacent lineage and self-reference boundary. The quantized refinement row is a single-particle isometric refinement walk with an external input stream; it has precise interfaces with quantum walks, multiresolution analysis, tensor-network isometries, and quantum cellular automata, but it is not identical to any of those structures without further data. Closing the input by a self-substitution rule produces a sharp cost: the binary compatible rule is unique up to mirror exchange and carries an intrinsic time arrow. Enlarging the alphabet separates compatibility from that forced arrow: the binary row is a rigid endpoint, while ternary and square alphabet rows admit reversible compatible witnesses. The common grammar is the phase-polygon equation for Q3 compatibility.

9425.1 Lineage of the Refinement Row

Principle 9425.1 (Single-particle refinement walk)**.** *The quantized refinement edge of Definition 9424.5 is best located as an isometric quantum walk on a growing refinement structure, together with an external input stream. Its cells are basis vectors in $\ell^2(C_n)$, not tensor factors. This separates the certificate from the standard quantum-cellular-automaton setting, where locality and index theorems live in a tensor-product operator algebra.*

Principle 9425.2 (Quantum-cellular-automaton boundary)**.** *A genuine quantum cellular automaton can have a one-particle sector that resembles a quantum walk. The converse is an additional construction problem: one must supply a tensor-factor state space, a local unitary or local isometry rule, and a functorial projection whose one-particle sector is the displayed refinement edge. The input-flow certificate therefore defines a target for such a parent automaton, not the parent automaton itself.*

Principle 9425.3 (Multiresolution and wavelet skeleton)**.** *The refinement isometry*

$$e_c \longmapsto N^{-1/2} \sum_{a=0}^{N-1} e_{ca}$$

is the finite-cell analogue of a Haar scaling relation. The scale chain is the multiresolution chain, and the innovation residual is the wavelet-detail row. The input-flow contribution is not the existence of this scaling skeleton; it is the quantized modulus row, the compatibility equation, and the

resulting norm allocation

$$\frac{\|\eta_{n+1}\|^2}{\|B_n\|^2} = N - 1.$$

Principle 9425.4 (Tensor-network interface)**.** *Layered isometries without disentanglers form the bare skeleton of a multiscale tensor network. To obtain a tensor-network dynamics rather than a single-particle refinement ledger, one must add tensor factors, entangling gates or disentanglers, and a rule for how the input stream acts on many-body states. The free-field reconstruction rows of chapter 9448 already use a second-quantized output of the machine; the missing data are local tensor rules, not merely many-body vocabulary.*

Principle 9425.5 (Lineage ledger)**.** *The same displayed refinement row has four neighboring descriptions: quantum walk on a growing graph, Haar multiresolution scaling, a bare isometric tensor-network skeleton, and the one-particle shadow of a possible quantum cellular automaton. These are lineage pointers. None of them changes the certificate used by Definition 9424.6 or the reflection-positive boundary of Definition 9424.21.*

9425.2 Self-Substitution Input

Definition 9425.6 (Binary self-substitution row)**.** *A binary self-substitution row closes the input by requiring each refined cell to inherit its parent phase through two fixed unit phases:*

$$u_{c0} = a\, u_c, \qquad u_{c1} = b\, u_c, \qquad |a| = |b| = 1.$$

It is compatible with the quantized refinement condition when the coarse-grained phase sum satisfies

$$a + b = \sqrt{2}.$$

Theorem 9425.7 (Binary compatible self-substitution)**.** *The binary self-substitution row satisfies the compatibility equation $a + b = \sqrt{2}$ if and only if*

$$\{a, b\} = \{e^{i\pi/4}, e^{-i\pi/4}\}.$$

Thus the compatible binary self-substitution rule is unique up to mirror exchange.

Proof. Let $a = e^{i\alpha}$ and $b = e^{i\beta}$. The condition that $a + b$ is the positive real number $\sqrt{2}$ forces the average angle to be 0 modulo 2π and the chord length to be $\sqrt{2}$. Hence the two angles are $+\pi/4$ and $-\pi/4$ up to exchange. □

Principle 9425.8 (Boundary cases)**.** *The equal-phase rule $a = b$ gives no innovation under the binary compatibility equation, while the opposite-phase rule $a = -b$ gives no retained coarse component. These two familiar substitution boundaries fail opposite sides of the refinement compatibility row. The compatible rule of Definition 9425.7 sits between them.*

Definition 9425.9 (One-step circular correlation)**.** *For a binary self-substitution row set*

$$\gamma(1) := \frac{b\overline{a}}{2 - a\overline{b}}.$$

This row records the first circular correlation of the induced phase measure.

Theorem 9425.10 (Self-substitution time arrow)**.** *For the compatible binary self-substitution rule,*

$$\gamma(1) = \frac{1-2i}{5}.$$

In particular the circular correlation is not real. The self-substitution input is therefore not time-reversal symmetric in this correlation row.

Proof. Choose the ordered representative $a = e^{i\pi/4}$ and $b = e^{-i\pi/4}$. Then $b\overline{a} = -i$ and $a\overline{b} = i$, so

$$\gamma(1) = \frac{-i}{2-i} = \frac{1-2i}{5}.$$

Mirror exchange conjugates the value and reverses the arrow. □

Principle 9425.11 (Self-reference cost)**.** *Closing the input by a binary self-substitution rule removes the arbitrary external digit choice only by moving choice into a structural rule. The compatible rule is rigid, and its circular correlation carries an intrinsic orientation. Perfectly symmetric self-copying lies on the zero-innovation boundary; compatible self-substitution has innovation but carries a time arrow.*

Principle 9425.12 (Spectral and metric rows of the self-substitution)**.** *The compatible binary self-substitution row has singular-continuous spectral behavior in its phase measure and a well-defined QGT metric row. The numerical ledger records a convergent metric value near*

$$h = 2.77243,$$

below the uniform baseline $\pi^2/3$. Its linear mean covector tends to zero, while the first circular mean remains nonzero, with modulus about 0.447 and angle about -63.4° in the displayed orientation. These data place the arrow in the circular holonomy layer rather than in a nonzero ordinary mean momentum.

Principle 9425.13 (Reflection-positive boundary of the self-substitution)**.** *The self-substitution row is a deterministic substitution input, not a reversible Markov input with a positive transition operator. It therefore does not automatically belong to the reflection-positive class of Definition 9424.21. Any Lorentz-facing use of this row must first supply a positive transition certificate or a separate Wick-facing bridge.*

9425.3 Phase-Polygon Compatibility

Definition 9425.14 (Self-substitution phase word)**.** *For a quantized refinement edge with factor N, an N-ary self-substitution phase word is a tuple*

$$v = (v_0, \ldots, v_{N-1}) \in \mathbb{T}^N.$$

It closes the refined phases by the rule

$$u_{ca} = v_a u_c$$

for every parent cell c and child index a. The resulting refined block over c is

$$\varepsilon u_c \sum_{a=0}^{N-1} v_a e_{ca}.$$

Theorem 9425.15 (Q3 phase-polygon equation)**.** *An N-ary self-substitution phase word satisfies the compatibility row $P_n^{\uparrow} B_{n+1} = \Phi_N B_n$ if and only if*

$$\sum_{a=0}^{N-1} v_a = \sqrt{N}.$$

The right side is the positive real representative fixed by the chosen coarse phase convention.

Proof. For a fixed parent cell c, write

$$\phi_c = N^{-1/2} \sum_{a=0}^{N-1} e_{ca}.$$

The refined block is

$$b_c = u_c \sum_{a=0}^{N-1} v_a e_{ca}.$$

Its projection onto the old lifted line $\mathbb{C}\phi_c$ has coefficient

$$\langle \phi_c, b_c \rangle = \frac{u_c}{\sqrt{N}} \sum_{a=0}^{N-1} v_a.$$

Thus $P^{\uparrow} b_c = u_c \phi_c$ for every parent cell if and only if the displayed phase sum is $\sqrt{N}$. □

Corollary 9425.16 (Self-substitution residual rate)**.** *Every Q3-compatible N-ary self-substitution phase word has residual rate*

$$\frac{\|\eta_{n+1}\|^2}{\|B_n\|^2} = N - 1.$$

Proof. Each parent block has norm square $N\varepsilon^2$, while its old lifted projection has norm square ε^2 by Definition 9425.15. Orthogonality of the projection residual gives block residual norm square $(N-1)\varepsilon^2$. Summing over the A_n parent cells and dividing by $\|B_n\|^2 = A_n \varepsilon^2$ gives the formula. □

Definition 9425.17 (Phase correlation row)**.** *For a self-substitution phase sequence u_n, define the two-point correlation, when the Cesaro limit exists, by*

$$\gamma(m) := \lim_{L\to\infty} \frac{1}{L} \sum_{n=0}^{L-1} u_{n+m} \overline{u_n}.$$

A real-valued correlation row is a necessary audit for time-reversal symmetry at the two-point phase level.

Theorem 9425.18 (Ternary real correlation recursion)**.** *For the ternary phase word*

$$A = e^{i\pi/6}, \qquad B = e^{-i\pi/2}, \qquad (v_0, v_1, v_2) = (A, B, A),$$

the correlations satisfy

$$\gamma(3m) = \gamma(m),$$

$$\gamma(3m+1) = \frac{1}{3}\big(-\gamma(m) + \gamma(m+1)\big),$$

and

$$\gamma(3m+2) = \frac{1}{3}\big(\gamma(m) - \gamma(m+1)\big).$$

Consequently every $\gamma(m)$ is real.

Proof. The identities $B\overline{A} + A\overline{B} = -1$ and $A\overline{A} = 1$ are immediate. Splitting the Cesaro average into the three residue classes of n modulo 3 gives the displayed recursions. The initial value is $\gamma(0) = 1$, and the one-step relation gives $\gamma(1) = -1/2$. The recursion expresses each later value as a real linear combination of smaller-index values, so induction gives $\gamma(m) \in \mathbb{R}$ for all m. □

9425.4 Externality Displacement

Principle 9425.19 (Externality displacement)**.** *The input-flow framework begins by externalizing digit choice. Attempts to close that choice move the externality to another row rather than removing it. A self-substitution rule moves it to the structural rule and pays with an intrinsic time arrow. A complex-refinement self-observation row moves it to the choice of complex multiplication structure and pays with an arithmetic admissibility condition. A thermodynamic-gravity assembly moves it to local equilibrium data, as recorded in Definition 9448.30. Each move leaves a theorem-level constraint behind.*

Principle 9425.20 (No hidden unification claim)**.** *The lineage and self-reference rows do not turn the input-flow grammar into a completed quantum-gravity or consciousness theory. They record where neighboring structures attach, which data are still external, and which constraints appear when external choice is pushed into the machine. This is a boundary ledger, not a unification theorem.*

9425.5 Self-Observation and Arithmetic Habitat

Definition 9425.21 (Self-similar observation row)**.** *A self-similar observation row is a refinement row in which the refined machine remains similar to the coarse machine. In a Gaussian refinement tower, this means that the refining sublattice is obtained by multiplication by a Gaussian integer.*

Theorem 9425.22 (Two-square admissibility)**.** *In the Gaussian refinement tower, a rational prime p admits a conformal index-p self-similar refinement exactly when*

$$p = a^2 + b^2$$

for integers a, b. Inert primes enter conformal Gaussian refinement only through even powers.

Proof. This is the conformal-refinement reading of Definition 9452.9. Multiplication by a Gaussian integer of norm p gives an index-p similar sublattice, and Fermat's two-square theorem characterizes exactly the rational primes that occur as such norms. □

Principle 9425.23 (Heegner habitat for universal shape blindness)**.** *If self-similar observation is required for all ideal refinements in an imaginary quadratic order, universal shape blindness holds exactly in the class-number-one habitat recorded by Definition 9452.16. Outside that habitat, the class group becomes a visible shape register. Perfect self-observation therefore lives only where the class group imposes no shape memory.*

Principle 9425.24 (Arithmetic constraint from self-observation)**.** *The two-square and class-number-one restrictions are not empirical claims about all possible input streams. They are conditional constraints produced by a self-similar observation requirement. In the unrestricted input-flow grammar the prime set remains external; under self-similar observation it is cut by the arithmetic of the selected complex structure.*

9425.6 Parent Automaton and Larger Alphabets

Principle 9425.25 (Parent automaton target)**.** *The parent-automaton target asks for a tensor-product, local, quantum-cellular-automaton realization whose one-particle sector is the quantized refinement input row. A solution would have to preserve the residual-rate law, the reflection-positive boundary, and the solenoid arithmetic rows after projection to the one-particle sector. The present chapter only states the target surface.*

Principle 9425.26 (Larger alphabet self-substitution)**.** *For an N-ary self-substitution row the compatibility equation becomes*

$$\sum_{a=0}^{N-1} e^{i\theta_a} = \sqrt{N}.$$

For $N = 2$ the compatibility equation is rigid and forces the oriented pair of Definition 9425.7. For $N \geq 3$ the solution set has positive dimension, so reversibility is a separate constraint on the self-substitution row rather than a consequence of compatibility alone.

Theorem 9425.27 (Higher-alphabet reversible self-substitution)**.** *The binary incompatibility between compatible self-substitution and time-reversal symmetry is not stable under enlargement of the alphabet. For $N = 3$ the phase row*

$$\theta = \left(\frac{\pi}{6}, -\frac{\pi}{2}, \frac{\pi}{6}\right)$$

satisfies

$$e^{i\pi/6} + e^{-i\pi/2} + e^{i\pi/6} = \sqrt{3}$$

and its circular-correlation recursion has real block coefficients. For every square alphabet size $N = k^2$, a compatible real-sign row exists: choose

$$\#\{+\} = \frac{k(k+1)}{2}, \qquad \#\{-\} = \frac{k(k-1)}{2}.$$

Then the sum of the signs is $k = \sqrt{N}$.

Proof. The ternary identity is immediate from $2e^{i\pi/6} - i = \sqrt{3}$. After factoring out the common rotation, the phase pattern is a third-root pattern with conjugate block products; the correlation recursion therefore has real coefficients at every finite depth. For the square alphabet row, the two displayed cardinalities are integers because $k(k+1)$ is even, their sum is k^2, and their signed difference is k. □

Principle 9425.28 (Binary endpoint)**.** *The forced time arrow of Definition 9425.10 is a binary endpoint. With two phases, the equation $a + b = \sqrt{2}$ leaves only the mirror pair $e^{\pm i\pi/4}$, and the first circular coefficient is nonreal. With three or more phases, compatibility can coexist with a real correlation audit. The arrow is therefore forced in the binary row and becomes a selectable feature in larger alphabets.*

Principle 9425.29 (Feasibility audit for phase searches)**.** *Any numerical or parametrized search for reversible self-substitution rows must keep*

$$\left|\sum_{a=0}^{N-1} e^{i\theta_a}\right| = \sqrt{N}$$

as a displayed constraint. A projection that maps infeasible trial values back into an inverse-trigonometric domain can create apparent zero-defect rows whose phase sum does not satisfy the compatibility equation. The feasibility constraint is part of the certificate, not a post-processing choice.

Principle 9425.30 (Lineage correspondence ledger)**.** *The correspondence checks for this boundary include quantum walks and quantum cellular automata, causal graph dynamics, multiresolution analysis and Haar wavelets, multiscale tensor networks, generalized Thue–Morse diffraction, complex multiplication, and class-number-one shape theory. These are source disciplines for the interface claims. The input-flow contribution is the exact placement of each discipline against the refinement, positivity, solenoid, and self-reference rows.*

Chapter 9426

Innovation Input Self-Substitution Correlation Boundary

chapter 9425 isolates the Q3 phase-polygon equation for self-substitution rows and exhibits the binary, ternary, and square-alphabet boundary cases. This chapter records the correlation machine behind those examples. The two-point arrow is not carried by Q3 itself; it is carried by the imaginary part of the word-level correlation coefficients that enter the carry recursion. The constructions below are stated for an arbitrary phase word; the Q3 equation

$$\sum_{a=0}^{N-1} v_a = \sqrt{N}$$

then selects the self-substitution rows compatible with the innovation input projection.

9426.1 Layered Phase Correlations

Definition 9426.1 (Layered phase sequence)**.** *Let $v = (v_0, \dots, v_{N-1}) \in \mathbb{T}^N$ be a self-substitution phase word as in Definition 9425.14. For*

$$n = a_0 + a_1 N + \cdots + a_{K-1} N^{K-1}, \qquad a_j \in \{0, \dots, N-1\},$$

define the level-K phase by

$$u_n^{(K)} := v_{a_0} v_{a_1} \cdots v_{a_{K-1}}.$$

Equivalently,

$$u_{Nq+a}^{(K)} = v_a u_q^{(K-1)}.$$

This is the one-index form of the parent-cell rule $u_{ca} = v_a u_c$.

Definition 9426.2 (Level correlation and two-point weak reversibility)**.** *For $m \geq 0$, set*

$$\gamma_K(m) := \frac{1}{N^K} \sum_{n=0}^{N^K-1-m} u_{n+m}^{(K)} \overline{u_n^{(K)}}.$$

When the limit exists, write

$$\gamma(m) := \lim_{K \to \infty} \gamma_K(m).$$

The phase word is two-point weakly reversible when

$$\gamma(m) \in \mathbb{R} \qquad \textit{for all } m \geq 0.$$

Equivalently, after extending the row by

$$\gamma(-m) := \overline{\gamma(m)},$$

the two-point row satisfies $\gamma(-m) = \gamma(m)$. Thus the row is invisible to reversal at the level of two-point phase correlations. This condition is weaker than reflection positivity. It only states that the two-point phase correlations do not by themselves distinguish a time orientation.

9426.2 Carry Recursion

Definition 9426.3 (Word-level correlation coefficients)**.** *For $0 \leq r < N$, define*

$$s_r(v) := \sum_{a=0}^{N-1-r} v_{a+r}\overline{v_a},$$

and

$$t_r(v) := \sum_{a=N-r}^{N-1} v_{a+r-N}\overline{v_a}.$$

For $r = 0$ this means $s_0 = N$ and $t_0 = 0$. For $r \geq 1$,

$$t_r(v) = \overline{s_{N-r}(v)}.$$

Lemma 9426.4 (Carry-complement identity)**.** *For $1 \leq r < N$, the carry coefficient satisfies*

$$t_r(v) = \overline{s_{N-r}(v)}.$$

Proof. Starting from the carry part and setting $b = a + r - N$ gives

$$t_r(v) = \sum_{a=N-r}^{N-1} v_{a+r-N}\overline{v_a} = \sum_{b=0}^{r-1} v_b\overline{v_{b+N-r}}.$$

Taking complex conjugates gives

$$\sum_{b=0}^{r-1} v_b\overline{v_{b+N-r}} = \overline{\sum_{b=0}^{r-1} v_{b+N-r}\overline{v_b}} = \overline{s_{N-r}(v)}.$$

□

Theorem 9426.5 (Carry recursion for self-substitution correlations)**.** *Let $m = Nq + r$ with $0 \leq r < N$. Then*

$$\gamma(Nq + r) = \frac{1}{N}\big(s_r(v)\gamma(q) + t_r(v)\gamma(q+1)\big).$$

In particular,

$$\gamma(Nq) = \gamma(q).$$

Proof. Write $n = Nb + a$. If $a + r < N$, then

$$n + m = N(b + q) + (a + r),$$

and the local coefficient is $v_{a+r}\overline{v_a}$. If $a + r \geq N$, then

$$n + m = N(b + q + 1) + (a + r - N),$$

and the local coefficient is $v_{a+r-N}\overline{v_a}$. Summing first over the child index a gives the coefficients s_r and t_r. At finite level this has the form

$$\gamma_K(Nq + r) = \frac{1}{N}\big(s_r(v)\gamma_{K-1}(q) + t_r(v)\gamma_{K-1}(q + 1)\big) + O(N^{-K}).$$

The boundary terms in the finite level average have relative size $O(N^{-K})$ and vanish as $K \to \infty$. □

Theorem 9426.6 (Existence and first correlation)**.** *For every phase word $v \in \mathbb{T}^N$, the limit $\gamma(m)$ exists for each fixed $m \geq 0$ and is uniquely determined by Definition 9426.5. Moreover*

$$\gamma(1) = \frac{s_1(v)}{N - t_1(v)}.$$

Proof. The value $\gamma(0) = 1$ is immediate. For $m = 1$, the finite-level recursion has the form

$$\gamma_K(1) = \frac{s_1}{N} + \frac{t_1}{N}\gamma_{K-1}(1) + O(N^{-K}).$$

Since $|t_1| \leq 1 < N$, this is a contraction and gives the displayed fixed point. For $m > 1$, write $m = Nq + r$. If $r = 0$, the value is $\gamma(q)$ with $q < m$. If $r > 0$, the recursion uses only $\gamma(q)$ and $\gamma(q + 1)$, and these have smaller indices except in the already treated case $m = 1$. Induction gives existence and uniqueness. □

9426.3 Weak Reversibility Criterion

Definition 9426.7 (Real word-level correlation condition)**.** *A phase word satisfies the real word-level correlation condition when*

$$s_r(v) \in \mathbb{R} \qquad 1 \leq r < N.$$

By Definition 9426.4, this is equivalent to requiring all coefficients $s_r(v)$ and $t_r(v)$ in the carry recursion to be real.

Theorem 9426.8 (Real word-level criterion)**.** *If*

$$s_r(v) \in \mathbb{R} \qquad 1 \leq r < N,$$

then

$$\gamma(m) \in \mathbb{R} \qquad \textit{for all } m \geq 0.$$

Proof. The condition implies $t_r(v) \in \mathbb{R}$ for all r. The first correlation is real by Definition 9426.6. For general m, use induction in the carry recursion: when $m = Nq + r$, the right side is a real linear combination of already real correlations. □

Corollary 9426.9 (Real-sign words)**.** *If $v_a \in \{+1, -1\}$ for every a, then the self-substitution phase word is two-point weakly reversible. In particular, the square-alphabet Q3 construction of Definition 9425.27 has no two-point phase arrow.*

Proof. For a real-sign word each product $v_{a+r}\overline{v_a}$ is ± 1. Hence every $s_r(v)$ is an integer, and Definition 9426.8 applies. □

Theorem 9426.10 (Square-alphabet real-sign Q3 rows)**.** *Let $N = k^2$. Choose*

$$P = \frac{k^2 + k}{2} \qquad \text{and} \qquad M = \frac{k^2 - k}{2}$$

entries equal to $+1$ and -1, respectively. The resulting phase word is Q3-compatible and two-point weakly reversible.

Proof. The cardinalities are integers because $k(k+1)$ and $k(k-1)$ are even, and $P + M = k^2 = N$. The phase sum is

$$\sum_a v_a = P - M = k = \sqrt{N},$$

so the Q3 phase-polygon equation holds. Since all entries are real signs, Definition 9426.9 gives two-point weak reversibility. □

9426.4 Boundary Examples

Principle 9426.11 (Binary coefficient source of the arrow)**.** *For the binary Q3 word*

$$v = (e^{i\pi/4}, e^{-i\pi/4}),$$

one has

$$s_1 = -i, \qquad t_1 = i.$$

Thus

$$\gamma(1) = \frac{s_1}{2 - t_1} = \frac{1 - 2i}{5}.$$

Hence

$$\operatorname{Im} \gamma(1) = -\frac{2}{5} \neq 0.$$

The binary arrow is therefore read at the word-level coefficient s_1, before any long-range limit is taken. In particular the binary Q3 self-substitution row is not two-point weakly reversible.

Principle 9426.12 (Ternary weakly reversible word)**.** *For the ternary Q3 word*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6}),$$

the Q3 sum is

$$2e^{i\pi/6} - i = \sqrt{3}.$$

The word-level coefficients are

$$s_1 = -1, \qquad s_2 = 1.$$

The real word-level criterion therefore gives $\gamma(m) \in \mathbb{R}$ for all m. The explicit recursion is

$$\gamma(3q) = \gamma(q),$$

$$\gamma(3q+1) = \frac{1}{3}\big(-\gamma(q) + \gamma(q+1)\big),$$

and

$$\gamma(3q+2) = \frac{1}{3}\big(\gamma(q) - \gamma(q+1)\big),$$

matching Definition 9425.18. The first values are

$$\gamma(1) = -\frac{1}{2}, \qquad \gamma(2) = \frac{1}{2}.$$

Principle 9426.13 (Quartic real-sign word)**.** *For the real-sign Q3 word*

$$v = (1, 1, 1, -1),$$

one has

$$1 + 1 + 1 - 1 = 2 = \sqrt{4}.$$

The coefficients are

$$s_1 = 1, \qquad s_2 = 0, \qquad s_3 = -1.$$

The carry coefficients are therefore

$$t_1 = -1, \qquad t_2 = 0, \qquad t_3 = 1.$$

Hence

$$\gamma(4q) = \gamma(q),$$
$$\gamma(4q+1) = \frac{1}{4}\big(\gamma(q) - \gamma(q+1)\big),$$
$$\gamma(4q+2) = 0,$$

and

$$\gamma(4q+3) = \frac{1}{4}\big(-\gamma(q) + \gamma(q+1)\big).$$

In particular $\gamma(1) = 1/5$, $\gamma(2) = 0$, *and* $\gamma(3) = -1/5$.

Principle 9426.14 (Location of the two-point arrow)**.** *The two-point phase arrow is not forced by the Q3 phase-polygon equation alone. It is carried by the imaginary parts of the word-level coefficients* $s_r(v)$. *The binary Q3 equation has no freedom to make those coefficients real, while the ternary and real-sign square-alphabet rows do.*

Principle 9426.15 (Coefficient readout)**.** *The boundary between the basic examples is already visible in the first word-level coefficients:*

N	*Q3 row*	*word-level readout*	*two-point row*
2	$(e^{i\pi/4}, e^{-i\pi/4})$	$s_1 = -i$	$\gamma(1) = \frac{1-2i}{5}$
3	$(e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$	$s_1 = -1,\ s_2 = 1$	$\gamma(m) \in \mathbb{R}$
4	$(1, 1, 1, -1)$	$s_1 = 1,\ s_2 = 0,\ s_3 = -1$	$\gamma(m) \in \mathbb{R}$
k^2	± 1 *with* $\#(+) - \#(-) = k$	$s_r \in \mathbb{R}$	$\gamma(m) \in \mathbb{R}$

Thus two-point weak reversibility is controlled by the realness of the coefficients $s_r(v)$, *not by Q3 compatibility alone.*

Principle 9426.16 (Finite-block boundary)**.** *Two-point weak reversibility is only the level detected by the pair correlations* $\gamma(m)$. *The finite-block layer is separated in chapter 9427. The present chapter closes only the two-point criterion.*

Chapter 9427

Innovation Input Finite-Block Reversibility Boundary

chapter 9426 closes the two-point correlation recursion for self-substitution phase words. This chapter separates that two-point result from the stronger finite-block question. The ternary Q3 row has real two-point correlations, but a four-point balanced block still detects a chiral residue.

9427.1 Signed Blocks

Definition 9427.1 (Signed block)**.** *A signed block is a finite integer combination*

$$\alpha = \sum_{j=1}^{r} e_j \delta_{m_j},$$

where

$$m_j \in \mathbb{Z}_{\geq 0}, \qquad e_j \in \mathbb{Z} \setminus \{0\}.$$

Its phase monomial is

$$U_\alpha(n) := \prod_{j=1}^{r} u_{n+m_j}^{e_j}.$$

Since $|u_n| = 1$, negative exponents are interpreted as conjugate powers, so $u_n^{-1} = \overline{u_n}$.

Definition 9427.2 (Charge and balanced blocks)**.** *The charge of a signed block is*

$$Q(\alpha) := \sum_{j=1}^{r} e_j.$$

The block is balanced when $Q(\alpha) = 0$. Balanced blocks are the correlation observables that pair phase and conjugate phase contributions; unbalanced blocks carry a net phase charge and are not the finite-block analogue of the two-point row.

Definition 9427.3 (Finite-block correlation)**.** *Let $M(\alpha) = \max_j m_j$. When the limit exists, define*

$$\Gamma(\alpha) := \lim_{K\to\infty} \frac{1}{N^K} \sum_{n=0}^{N^K-1-M(\alpha)} U_\alpha(n).$$

The two-point row is the special case

$$\Gamma(\delta_0 - \delta_m) = \overline{\gamma(m)}.$$

When $\gamma(m)$ is real, this agrees with $\gamma(m)$.

Definition 9427.4 (Translation of a block)**.** *For $c \in \mathbb{Z}_{\geq 0}$, set*

$$\tau_c \alpha := \sum_{j=1}^{r} e_j \delta_{m_j + c}.$$

Whenever both limits exist,

$$\Gamma(\tau_c \alpha) = \Gamma(\alpha).$$

Thus blocks that differ by a common translation represent the same finite-block correlation.

9427.2 Weak Reversibility Layers

Definition 9427.5 (Order of a signed block)**.** *The order of a signed block is*

$$|\alpha| := \sum_j |e_j|.$$

Definition 9427.6 (Order-bounded weak reversibility)**.** *A self-substitution phase word is d-order weakly reversible when every balanced signed block with $|\alpha| \leq d$ satisfies*

$$\Gamma(\alpha) \in \mathbb{R}.$$

The case $d = 2$ is exactly two-point weak reversibility.

Definition 9427.7 (Finite-block weak reversibility)**.** *A self-substitution phase word is finite-block weakly reversible when*

$$\Gamma(\alpha) \in \mathbb{R}$$

for every balanced signed block α for which the correlation exists.

Principle 9427.8 (Reversibility hierarchy)**.** *The weak reversibility layers satisfy*

$$\textit{finite-block weak reversibility} \Longrightarrow d\textit{-order weak reversibility} \Longrightarrow 2\textit{-order weak reversibility}.$$

The implications are one-way assertions. The ternary computation below shows that the last implication cannot be reversed.

9427.3 Finite-Block Carry Recursion

Definition 9427.9 (Carry branch of a signed block)**.** *For a signed block*

$$\alpha = \sum_j e_j \delta_{m_j}$$

and a residue $a \in \{0, \ldots, N-1\}$, write

$$a + m_j = N q_{j,a} + r_{j,a}, \qquad 0 \leq r_{j,a} < N.$$

The branch coefficient and branch block are

$$C_a(\alpha) := \prod_j v_{r_{j,a}}^{e_j},$$

and

$$\alpha_a := \sum_j e_j \delta_{q_{j,a}}.$$

After the branch is formed, a common translation may be removed by Definition 9427.4.

Lemma 9427.10 (Charge preservation under carry branches)**.** *For every residue a,*

$$Q(\alpha_a) = Q(\alpha).$$

In particular, balanced blocks have balanced carry branches.

Proof. The carry branch changes only the locations of the atoms:

$$\alpha_a = \sum_j e_j \delta_{q_{j,a}}.$$

The integer coefficients e_j are unchanged, so their sum is unchanged. □

Theorem 9427.11 (Finite-block carry recursion)**.** *If $\Gamma(\alpha)$ and all branch correlations $\Gamma(\alpha_a)$ exist, then*

$$\Gamma(\alpha) = \frac{1}{N} \sum_{a=0}^{N-1} C_a(\alpha)\Gamma(\alpha_a).$$

Proof. Split the finite-level average by writing $n = Nb + a$. For each atom,

$$n + m_j = Nb + a + m_j = N(b + q_{j,a}) + r_{j,a}.$$

The self-substitution rule gives

$$u_{n+m_j} = v_{r_{j,a}} u_{b+q_{j,a}},$$

hence

$$u_{n+m_j}^{e_j} = v_{r_{j,a}}^{e_j} u_{b+q_{j,a}}^{e_j}.$$

Multiplying over j gives

$$U_\alpha(Nb + a) = C_a(\alpha) U_{\alpha_a}(b).$$

Averaging first over residues and then letting $K \to \infty$ gives the displayed recursion; boundary terms have vanishing relative size. □

9427.4 Ternary Four-Point Test

Definition 9427.12 (Ternary Q3 row)**.** *Let*

$$A = e^{i\pi/6}, \qquad B = e^{-i\pi/2}, \qquad v = (A, B, A).$$

Then $A + B + A = \sqrt{3}$, so this is a ternary Q3-compatible self-substitution phase word. By Definition 9426.12, all its two-point correlations are real.

Definition 9427.13 (Four-point test block)**.** *Set*

$$\alpha_\star := \delta_0 + \delta_1 - \delta_3 - \delta_5.$$

It is balanced, since $Q(\alpha_\star) = 1 + 1 - 1 - 1 = 0$*, and has order* $|\alpha_\star| = 4$*. Its correlation is*

$$\Gamma(\alpha_\star) = \langle u_n u_{n+1} \overline{u_{n+3} u_{n+5}} \rangle .$$

Definition 9427.14 (Ternary auxiliary correlations)**.** *Let*

$$\omega := e^{2\pi i/3} = -\frac{1}{2} + \frac{\sqrt{3}}{2} i.$$

For the ternary row, set

$$X := \Gamma(2\delta_0 - 2\delta_1),$$
$$Y := \Gamma(\delta_0 - \delta_1),$$
$$V := \Gamma(\delta_0 - \delta_2),$$
$$W := \Gamma(2\delta_0 - \delta_1 - \delta_2),$$

and

$$Z := \Gamma(\delta_0 + \delta_1 - \delta_3 - \delta_5).$$

Lemma 9427.15 (Ternary two-point and squared two-point values)**.** *For the ternary row,*

$$Y = -\frac{1}{2}, \qquad V = \frac{1}{2}, \qquad X = -\frac{1}{2}.$$

Proof. For $Y = \Gamma(\delta_0 - \delta_1)$, the carry branches give

$$Y = \frac{1}{3}(\omega + \overline{\omega} + Y) = \frac{1}{3}(-1 + Y),$$

so $Y = -1/2$.

For $V = \Gamma(\delta_0 - \delta_2)$, the carry branches give

$$V = \frac{1}{3}(1 + (\omega + \overline{\omega})Y) = \frac{1}{3}(1 - Y).$$

Substituting $Y = -1/2$ gives $V = 1/2$.

For $X = \Gamma(2\delta_0 - 2\delta_1)$, the carry branches give

$$X = \frac{1}{3}(\overline{\omega} + \omega + X) = \frac{1}{3}(-1 + X).$$

Thus $X = -1/2$. □

Lemma 9427.16 (Ternary three-point auxiliary cancellation)**.** *For the ternary row,*

$$W = \Gamma(2\delta_0 - \delta_1 - \delta_2) = 0.$$

Proof. For the block $2\delta_0 - \delta_1 - \delta_2$, all three carry branch coefficients are ω, and the branch correlations are 1, Y, and X. Therefore

$$W = \frac{1}{3}\omega(1 + Y + X).$$

By Definition 9427.15,

$$1 + Y + X = 1 - \frac{1}{2} - \frac{1}{2} = 0.$$

Hence $W = 0$. □

Theorem 9427.17 (Ternary four-point chirality)**.** *For the ternary Q3 row*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6}),$$

one has

$$\Gamma(\delta_0 + \delta_1 - \delta_3 - \delta_5) = \frac{i\sqrt{3}}{6}.$$

Consequently this row is two-point weakly reversible but not 4-order weakly reversible.

Proof. Write

$$Z := \Gamma(\delta_0 + \delta_1 - \delta_3 - \delta_5).$$

The three carry branches of $\alpha_\star$ give

$$a = 0 : \qquad \overline{\omega}X,$$

$$a = 1 : \qquad W,$$

and

$$a = 2 : \qquad \omega V.$$

Thus

$$Z = \frac{1}{3}\left(\overline{\omega}X + W + \omega V\right).$$

Using Definition 9427.15 and Definition 9427.16,

$$Z = \frac{1}{3}\left(-\frac{1}{2}\overline{\omega} + \frac{1}{2}\omega\right) = \frac{1}{6}(\omega - \overline{\omega}).$$

Since

$$\omega - \overline{\omega} = i\sqrt{3},$$

the claimed value follows. □

Corollary 9427.18 (Two-point reversibility does not force finite-block reversibility)**.** *There exists a Q3-compatible self-substitution phase word such that*

$$\gamma(m) \in \mathbb{R} \qquad \textit{for all } m \geq 0,$$

while some balanced finite-block correlation is not real.

Proof. The ternary row of Definition 9427.12 has real two-point correlations by Definition 9426.12. The balanced block $\alpha_\star$ of Definition 9427.13 has nonreal correlation by Definition 9427.17. □

9427.5 Layer Separation

Principle 9427.19 (Chirality moves upward)**.** *For the ternary row, the two-point carry coefficients*

$$s_1 = -1, \qquad s_2 = 1$$

are real, so the two-point row sees no phase arrow. The four-point block with sites $0, 1, 3, 5$ *crosses the ternary digit boundaries differently: its carry branches leave the asymmetric residue*

$$\frac{1}{6}(\omega - \overline{\omega}) = \frac{i\sqrt{3}}{6}.$$

Thus the ternary row removes the arrow from the two-point layer but does not remove it from all finite-block layers.

Principle 9427.20 (Reversibility audit table)**.** *The basic Q3 rows occupy different levels of the audit hierarchy:*

audit layer	*object*	$N = 2$ *Q3 row*	$N = 3$ *Q3 row*
first two-point	$\operatorname{Im}\gamma(1)$	$\neq 0$	0
all two-point	$\gamma(m) \in \mathbb{R}\ \forall m$	*no*	*yes*
four-point	$\Gamma(\delta_0 + \delta_1 - \delta_3 - \delta_5)$	*not needed*	$\frac{i\sqrt{3}}{6}$
finite-block weak	$\Gamma(\alpha) \in \mathbb{R}\ \forall Q(\alpha) = 0$	*no*	*no*
RP/OS	*reflection-positive matrices*	*undecided*	*undecided*

The table separates two-point weak reversibility, finite-block weak reversibility, and RP/OS positivity. None of the stronger layers may be inferred from real two-point correlations alone.

Principle 9427.21 (Reflection-positive boundary)**.** *Finite-block realness is still weaker than reflection positivity. The finite-window reflection-positive audit is separated in chapter 9428. The four-point computation above is therefore a layer-separation result, not an RP certificate.*

Chapter 9428

Innovation Input Finite-Window Reflection-Positive Boundary

chapter 9427 separates two-point realness from finite-block realness. Reflection positivity is still stronger: it asks for nonnegative quadratic forms for reflected future observables. This chapter records a finite-window audit for self-substitution phase rows and applies it to the smallest Q3 representatives already isolated above.

9428.1 Finite-Window Reflection

Definition 9428.1 (Discrete reflection)**.** *Let the discrete reflection act on integer sites by*

$$\theta(j) = 1 - j.$$

Thus $1 \mapsto 0$, $2 \mapsto -1$, *and* $3 \mapsto -2$. *On signed block atoms, set*

$$\theta\delta_j := \delta_{1-j}.$$

For a signed block

$$\alpha = \sum_j e_j \delta_j,$$

define

$$\theta\alpha := \sum_j e_j \delta_{1-j}.$$

Definition 9428.2 (Future monomial)**.** *A future signed block in a window of length* L *is an anchored block of the form*

$$\alpha = \sum_{j=1}^{L} e_j \delta_j, \qquad e_j \in \mathbb{Z}.$$

The sites $1, \ldots, L$ *are the actual future positions relative to the reflection plane. The block is not translated before reflection. It determines the future monomial*

$$U_\alpha(n) := \prod_{j=1}^{L} u_{n+j}^{e_j}.$$

When $Q(\alpha) = \sum_j e_j = 0$, *the monomial is neutral.*

Definition 9428.3 (Finite-window RP matrix)**.** *Given anchored future signed blocks $\alpha_0, \ldots, \alpha_m$, define*

$$M_{ij} := \langle \theta U_{\alpha_i}, U_{\alpha_j} \rangle := \Gamma(\alpha_j - \theta\alpha_i).$$

The order of operations is fixed: keep the future blocks at their anchored sites, reflect α_i, form $\alpha_j - \theta\alpha_i$, and only then translate all sites by a common integer if negative sites occur, using Definition 9427.4. The window passes the reflection-positive check when

$$M \succeq 0.$$

Proposition 9428.4 (Translation and reflection anti-commute)**.** *For $T_s\delta_j := \delta_{j+s}$,*

$$\theta T_s = T_{-s}\theta.$$

Thus finite-window RP entries depend on the anchored placement of the future blocks before reflection, even though Γ itself is invariant under common translation after the reflected difference is formed.

Proof. On atoms,

$$\theta T_s \delta_j = \theta\delta_{j+s} = \delta_{1-j-s},$$

while

$$T_{-s}\theta\delta_j = T_{-s}\delta_{1-j} = \delta_{1-j-s}.$$

The identity follows by linearity. □

Definition 9428.5 (Charged and neutral RP windows)**.** *The charged RP audit allows arbitrary future monomials. The neutral RP audit restricts to future monomials with $Q(\alpha) = 0$. Therefore*

$$\textit{charged RP} \Longrightarrow \textit{neutral RP}.$$

The neutral audit is the projective or gauge-invariant version of the same finite-window test.

Theorem 9428.6 (Finite-window Gram criterion)**.** *Let $F_i = U_{\alpha_i}$ be a finite list of future monomials. For*

$$F = \sum_i c_i F_i,$$

one has

$$\langle \theta F, F \rangle = \sum_{i,j} \overline{c_i} M_{ij} c_j.$$

Consequently the finite window satisfies reflection positivity exactly when $M \succeq 0$.

Proof. Expand the quadratic form:

$$\langle \theta F, F \rangle = \left\langle \theta \left(\sum_i c_i F_i \right), \sum_j c_j F_j \right\rangle.$$

The reflection acts on the observables, while the first slot of the correlation pairing is conjugate-linear. Hence

$$\langle \theta F, F \rangle = \sum_{i,j} \overline{c_i} c_j \langle \theta F_i, F_j \rangle = \sum_{i,j} \overline{c_i} M_{ij} c_j.$$

Nonnegativity for every coefficient vector c is exactly positive semidefiniteness of the Gram matrix. □

9428.2 Charged One-Site Obstruction

Theorem 9428.7 (Charged one-site necessary condition)**.** *If the charged finite-window RP audit holds, then*

$$\gamma(1) \in \mathbb{R}_{\geq 0}.$$

Proof. Take the future monomial $F(n) = u_{n+1}$. Its reflection is $\theta F(n) = u_n$. Therefore

$$\langle \theta F, F \rangle = \langle u_n, u_{n+1} \rangle = \langle \overline{u_n} u_{n+1} \rangle = \gamma(1).$$

Charged RP requires this value to be nonnegative. □

Corollary 9428.8 (Binary charged RP obstruction)**.** *The binary Q3 self-substitution row*

$$v = (e^{i\pi/4}, e^{-i\pi/4})$$

fails charged finite-window RP at the one-site window.

Proof. By Definition 9426.11,

$$\gamma(1) = \frac{1-2i}{5}.$$

This is not real, so it cannot be a nonnegative real number. □

Corollary 9428.9 (Ternary charged RP obstruction)**.** *The ternary Q3 self-substitution row*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$$

fails charged finite-window RP at the one-site window.

Proof. By Definition 9426.12,

$$\gamma(1) = -\frac{1}{2}.$$

The value is real but negative, contradicting Definition 9428.7. □

Principle 9428.10 (Quartic one-site survival)**.** *For the real-sign quartic Q3 row $v = (1, 1, 1, -1)$,*

$$\gamma(1) = \frac{1}{5} > 0.$$

Thus the row passes the charged one-site test, but this does not certify reflection positivity.

9428.3 Ternary Neutral RP Failure

Lemma 9428.11 (Ternary neutral diagonal value)**.** *For the ternary row, one has*

$$\gamma(2) = \frac{1}{2}$$

and

$$D_{13} := \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_5) = \frac{1}{6}.$$

Proof. The value $\gamma(2) = 1/2$ is Definition 9426.12. For the block $\delta_0 - \delta_2 + \delta_3 - \delta_5$, the ternary carry recursion has three branches:

$$a = 0: \qquad 1 \cdot 1,$$

$$a = 1: \qquad \omega\,\Gamma(\delta_0 - \delta_2),$$

and

$$a = 2: \qquad \overline{\omega}\,\Gamma(\delta_0 - \delta_2),$$

where $\omega = e^{2\pi i/3}$. Therefore

$$D_{13} = \frac{1}{3}\left(1 + (\omega + \overline{\omega})\gamma(2)\right).$$

Since $\omega + \overline{\omega} = -1$ and $\gamma(2) = 1/2$,

$$D_{13} = \frac{1}{3}\left(1 - \frac{1}{2}\right) = \frac{1}{6}.$$

□

Theorem 9428.12 (Ternary neutral RP obstruction)**.** *For the ternary Q3 row*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6}),$$

the neutral future observables

$$F_0 = 1, \qquad F_1 = u_{n+1}\overline{u_{n+3}}$$

have finite-window RP matrix

$$M = \begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{6} \end{pmatrix}.$$

Its determinant is

$$\det M = \frac{1}{6} - \frac{1}{4} = -\frac{1}{12} < 0.$$

Hence the ternary row fails neutral finite-window RP.

Proof. The constant entry is $M_{00} = 1$. The off-diagonal entry is

$$M_{01} = \langle F_1 \rangle = \langle u_{n+1}\overline{u_{n+3}} \rangle = \gamma(2) = \frac{1}{2}.$$

The same value gives M_{10}. For the diagonal entry, set

$$\alpha = \delta_1 - \delta_3.$$

Then

$$\theta\alpha = \delta_0 - \delta_{-2}.$$

Thus

$$\alpha - \theta\alpha = (\delta_1 - \delta_3) - (\delta_0 - \delta_{-2}).$$

After translating by 2, this becomes

$$\delta_0 - \delta_2 + \delta_3 - \delta_5.$$

Therefore

$$M_{11} = \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_5) = D_{13} = \frac{1}{6}$$

by Definition 9428.11. The displayed determinant is negative, so M is not positive semidefinite. □

Corollary 9428.13 (Two-point realness does not imply neutral RP)**.** *Two-point weak reversibility does not imply neutral finite-window RP.*

Proof. The ternary Q3 row has real two-point correlations by Definition 9426.12, but fails neutral finite-window RP by Definition 9428.12. □

9428.4 Quartic Neutral RP Failure

Lemma 9428.14 (Quartic neutral diagonal value)**.** *For the quartic real-sign row* $v = (1, 1, 1, -1)$ *and*

$$F(n) = u_{n+1}\overline{u_{n+2}},$$

one has

$$\langle \theta F, F\rangle = -\frac{3}{5}.$$

Proof. The observable corresponds to $\alpha = \delta_1 - \delta_2$. Since

$$\theta\alpha = \delta_0 - \delta_{-1},$$

the reflected diagonal block is

$$\alpha - \theta\alpha = (\delta_1 - \delta_2) - (\delta_0 - \delta_{-1}).$$

After translating by 1, this becomes

$$\delta_0 - \delta_1 + \delta_2 - \delta_3.$$

Set

$$D_{12} := \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3).$$

For $v = (1, 1, 1, -1)$, the four carry branches contribute

$$a = 0 : \qquad -1 \cdot 1,$$

$$a = 1 : \qquad -1 \cdot \gamma(1),$$

$$a = 2 : \qquad -1 \cdot 1,$$

and

$$a = 3 : \qquad -1 \cdot \gamma(1).$$

Therefore

$$D_{12} = -\frac{1}{4}\,(2 + 2\gamma(1)) = -\frac{1}{2}(1 + \gamma(1)).$$

Using $\gamma(1) = 1/5$ from Definition 9426.13,

$$D_{12} = -\frac{1}{2}\left(1 + \frac{1}{5}\right) = -\frac{3}{5}.$$

□

Theorem 9428.15 (Quartic neutral RP obstruction)**.** *The quartic real-sign Q3 row*

$$v = (1, 1, 1, -1)$$

fails neutral finite-window RP.

Proof. The neutral future observable $F(n) = u_{n+1}\overline{u_{n+2}}$ has

$$\langle \theta F, F\rangle = -\frac{3}{5} < 0$$

by Definition 9428.14. A positive semidefinite reflection-positive matrix cannot have a negative diagonal entry. □

Corollary 9428.16 (Real-sign correlations do not imply RP)**.** *Real-sign Q3 words can have real two-point correlations while failing neutral finite-window RP.*

Proof. The quartic row is real-sign and Q3-compatible by Definition 9426.13; its two-point correlations are real by Definition 9426.9. It fails neutral finite-window RP by Definition 9428.15. □

9428.5 Strict Layering

Theorem 9428.17 (Self-substitution reversibility layers)**.** *For Q3 self-substitution phase rows, finite-window reflection positivity is strictly stronger than real two-point correlations and real word-level correlations.*

Proof. The ternary row has real two-point correlations but fails neutral finite-window RP by Definition 9428.12. The quartic real-sign row has real word-level correlations but fails neutral finite-window RP by Definition 9428.15. The binary row fails even the charged one-site condition by Definition 9428.8. □

Principle 9428.18 (Terminology separation)**.** *The self-substitution rows require three distinct terms. Correlation reversibility means*

$$\gamma(m) \in \mathbb{R} \qquad \textit{for all } m.$$

Finite-block real reversibility means

$$\Gamma(\alpha) \in \mathbb{R}$$

for every balanced block α. RP reversibility means that every finite-window reflection-positive matrix is positive semidefinite, with the limiting row matching the OS/RP positivity used by Definition 9424.21. The ternary row belongs to the first class but not to the second or third. The quartic row belongs to the first class but fails the third.

Principle 9428.19 (Window audit table)**.** *The smallest Q3 representatives have the following RP-facing status:*

N	*representative Q3 row*	*two-point row*	*neutral RP*	*charged RP*
2	$(e^{i\pi/4}, e^{-i\pi/4})$	*nonreal*	*fail*	*fail*
3	$(e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$	*real*	*fail*	*fail*
4	$(1, 1, 1, -1)$	*real*	*fail*	*one-site pass*

Thus Q3 self-reference can coexist with the absence of a two-point arrow, but these minimal representatives do not provide an OS/RP-compatible self-substitution row.

Definition 9428.20 (Neutral RP deficit)**.** *For a bounded neutral future window, let $M^{\mathrm{neu}}_{N,L}(v)$ denote the finite-window RP matrix formed from the neutral monomials supported in $\{1, \ldots, L\}$. The neutral RP deficit is the least eigenvalue*

$$\rho^{\mathrm{neu}}_{N,L}(v) := \lambda_{\min} M^{\mathrm{neu}}_{N,L}(v).$$

Nonnegative values are necessary finite-window evidence for RP compatibility; negative values are finite-window obstruction certificates.

Chapter 9429

Innovation Input Minimal RP Deficit Search Boundary

chapter 9428 gives finite-window reflection positivity as a matrix condition. This chapter turns that condition into a search object: the Q3 constraint is kept exact, RP failure is measured by least eigenvalues and link deficits, and negative finite-window certificates become local obstructions to OS/RP-compatible self-substitution.

9429.1 Q3 Search Space

Definition 9429.1 (Q3 phase-word space)**.** *For $N \geq 2$, define*

$$\mathcal{Q}_N := \left\{ v = (v_0, \ldots, v_{N-1}) \in \mathbb{T}^N : \sum_{a=0}^{N-1} v_a = \sqrt{N} \right\}.$$

An element $v \in \mathcal{Q}_N$ determines the self-substitution rule

$$u_{Nn+a} = v_a u_n.$$

Proposition 9429.2 (Expected dimension of the Q3 space)**.** *At a nonsingular point, $\mathcal{Q}_N$ is a real manifold of dimension $N-2$.*

Proof. The torus $\mathbb{T}^N$ has real dimension N. The Q3 equation

$$\sum_a v_a = \sqrt{N}$$

is one complex equation, equivalently the two real equations

$$\operatorname{Re} \sum_a v_a = \sqrt{N},$$

and

$$\operatorname{Im} \sum_a v_a = 0.$$

At points where these two constraints have differential rank 2, the regular value theorem gives dimension $N-2$. □

Definition 9429.3 (Last-two-side closure)**.** *Choose $v_0, \dots, v_{N-3} \in \mathbb{T}$ and set*

$$R := \sqrt{N} - \sum_{a=0}^{N-3} v_a.$$

If $|R| \le 2$, write $R = \rho e^{i\phi}$ with $0 \le \rho \le 2$ and choose

$$v_{N-2} = e^{i\phi} e^{i\alpha},$$

$$v_{N-1} = e^{i\phi} e^{-i\alpha},$$

where

$$\cos\alpha = \frac{\rho}{2}.$$

Theorem 9429.4 (Exact Q3 closure)**.** *Every phase word constructed by Definition 9429.3 lies in $\mathcal{Q}_N$.*

Proof. The construction gives $v_{N-2} + v_{N-1} = R$. Therefore

$$\sum_{a=0}^{N-1} v_a = \sum_{a=0}^{N-3} v_a + v_{N-2} + v_{N-1} = \sum_{a=0}^{N-3} v_a + R = \sqrt{N}.$$

□

Principle 9429.5 (No projection after the fact)**.** *Searches for RP-compatible self-substitution rows must keep*

$$\sum_a v_a = \sqrt{N}$$

as an exact constraint. Projecting or clipping an infeasible phase row back toward Q3 after optimization can create apparent zero-defect rows whose phase sum is not a certificate. Exact closure or explicitly constrained optimization is part of the search object.

9429.2 Neutral Window Deficits

Definition 9429.6 (Neutral future block set)**.** *For a window length $L \ge 1$ and exponent bound $R \ge 1$, let $\mathcal{B}^{\mathrm{neu}}_{L,R}$ be the set of anchored future signed blocks*

$$\alpha = \sum_{j=1}^{L} e_j \delta_j$$

with

$$e_j \in \{-R, -R+1, \dots, R\}$$

and

$$\sum_{j=1}^{L} e_j = 0.$$

The zero block is included. The sites $1, \dots, L$ remain anchored until the reflected difference in the RP matrix has been formed.

Definition 9429.7 (Neutral finite-window RP matrix)**.** *For an enumeration*

$$\mathcal{B} = \{\alpha_0, \ldots, \alpha_m\} \subset \mathcal{B}^{\mathrm{neu}}_{L,R},$$

with $\alpha_0 = 0$*, define*

$$M_{\mathcal{B}}(v)_{ij} := \Gamma(\alpha_j - \theta\alpha_i).$$

The reflection is the one from Definition 9428.1; the anchoring discipline is the one in Definition 9428.3. The row passes the window $\mathcal{B}$ *when* $M_{\mathcal{B}}(v) \succeq 0$*.*

Definition 9429.8 (Hermitian defect and RP deficit)**.** *For a neutral window* $\mathcal{B}$*, define the Hermitian defect*

$$\eta_{\mathcal{B}}(v) := \|M_{\mathcal{B}}(v) - M_{\mathcal{B}}(v)^*\|.$$

Define the RP deficit by

$$\rho_{\mathcal{B}}(v) := \lambda_{\min}\left(\frac{M_{\mathcal{B}}(v) + M_{\mathcal{B}}(v)^*}{2}\right).$$

An RP candidate must have $\eta_{\mathcal{B}}(v) = 0$ *and* $\rho_{\mathcal{B}}(v) \geq 0$ *for every finite window.*

Definition 9429.9 (Optimal neutral window deficit)**.** *The optimal neutral deficit at window* (N, L, R) *is*

$$\rho^{\mathrm{neu}}_{N,L,R} := \sup_{v \in \mathcal{Q}_N} \rho_{\mathcal{B}^{\mathrm{neu}}_{L,R}}(v).$$

If $\rho^{\mathrm{neu}}_{N,L,R} < 0$*, no* N*-ary Q3 self-substitution row passes that neutral RP window. If it is nonnegative, the result is only a finite-window candidate condition, not a complete RP certificate.*

9429.3 Link Deficit Gate

Definition 9429.10 (Link observable)**.** *For* $d \geq 1$*, define the neutral future observable*

$$F_d(n) := u_{n+1}\overline{u_{n+1+d}}.$$

Its signed block is

$$\alpha_d := \delta_1 - \delta_{1+d}.$$

Definition 9429.11 (Link mean and reflected diagonal)**.** *Set*

$$\mu_d := \langle F_d \rangle = \Gamma(\delta_1 - \delta_{1+d}) = \Gamma(\delta_0 - \delta_d).$$

With the convention

$$\gamma(d) = \Gamma(\delta_d - \delta_0),$$

this means $\mu_d = \overline{\gamma(d)}$*. Also set*

$$D_d := \langle \theta F_d, F_d \rangle.$$

Directly,

$$D_d = \Gamma(\delta_0 - \delta_d + \delta_{d+1} - \delta_{2d+1}).$$

Theorem 9429.12 (Link two-by-two necessary condition)**.** *If neutral RP holds, then for every $d \geq 1$,*

$$D_d \in \mathbb{R}_{\geq 0}$$

and

$$D_d \geq |\mu_d|^2.$$

Equivalently, the link deficit

$$\Delta_d(v) := D_d - |\mu_d|^2$$

must satisfy $\Delta_d(v) \geq 0$.

Proof. Use the two future observables $F_0 = 1$ and $F_1 = F_d$. Their RP matrix is

$$M_d = \begin{pmatrix} 1 & \mu_d \\ \overline{\mu_d} & D_d \end{pmatrix}.$$

Under neutral RP this matrix is Hermitian positive semidefinite. Therefore its diagonal entry D_d is nonnegative real, and its determinant satisfies

$$\det M_d = D_d - |\mu_d|^2 \geq 0.$$

□

Corollary 9429.13 (Negative link deficit obstruction)**.** *If $D_d \notin \mathbb{R}$ or $\Delta_d(v) < 0$ for some $d \geq 1$, then the phase word does not satisfy neutral RP.*

Proof. This is the contrapositive of Definition 9429.12. □

9429.4 Known Link Obstructions

Principle 9429.14 (Ternary link obstruction)**.** *For the ternary Q3 row*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6}),$$

one has $\mu_2 = 1/2$ and

$$D_2 = \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_5) = \frac{1}{6}.$$

Therefore

$$\Delta_2 = D_2 - |\mu_2|^2 = \frac{1}{6} - \frac{1}{4} = -\frac{1}{12}.$$

The ternary row is excluded by the $d = 2$ link window.

Principle 9429.15 (Quartic link obstruction)**.** *For the quartic real-sign Q3 row*

$$v = (1, 1, 1, -1),$$

one has $\mu_1 = 1/5$ and

$$D_1 = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3) = -\frac{3}{5}.$$

Therefore

$$\Delta_1 = D_1 - |\mu_1|^2 = -\frac{3}{5} - \frac{1}{25} = -\frac{16}{25}.$$

The quartic row is excluded by the $d = 1$ link window.

Principle 9429.16 (First RP screen)**.** *The link deficits form the first screen for RP-compatible self-substitution. A candidate must satisfy*

$$\Delta_d(v) \geq 0$$

for all checked links. A single negative link deficit is already a finite-window neutral RP obstruction, before any larger RP matrix is assembled.

9429.5 Exact Correlation Evaluation

Definition 9429.17 (Block diameter)**.** *For a signed block*

$$\alpha = \sum_j e_j \delta_{m_j},$$

define

$$\operatorname{diam}(\alpha) := \max_j m_j - \min_j m_j.$$

After a common translation, one may assume $\min_j m_j = 0$.

Definition 9429.18 (Carry closure)**.** *For a finite set $\mathcal{A}$ of signed blocks, the one-step carry closure contains every branch block α_a of Definition 9427.9, for $\alpha \in \mathcal{A}$ and $0 \leq a < N$, normalized by translation. Iterating this operation gives the carry closure*

$$\overline{\mathcal{A}}^{\text{carry}}.$$

Proposition 9429.19 (Finite-window correlations reduce to a finite linear system)**.** *For a fixed finite RP window and a fixed exponent bound, the block correlations appearing in its matrix entries reduce, under carry closure, to a finite linear system*

$$X = A(v)X + b(v).$$

The entries of $A(v)$ and $b(v)$ are averages of Laurent monomials in the phase word entries v_a.

Proof. For a fixed block $\alpha = \sum_j e_j \delta_{m_j}$, the carry branch at residue a replaces each index by

$$q_{j,a} = \left\lfloor \frac{a + m_j}{N} \right\rfloor.$$

Hence branch diameters obey the coarse bound

$$\operatorname{diam}(\alpha_a) \leq \left\lceil \frac{\operatorname{diam}(\alpha)}{N} \right\rceil.$$

All persistent self-cycles are confined to finitely many small-diameter blocks. Starting from a finite window and repeatedly applying carry branches therefore produces finitely many normalized block types. Writing Definition 9427.11 for each block gives the displayed finite linear system. □

Principle 9429.20 (Exact evaluator contract)**.** *For search purposes, the exact evaluator is the carry-closure linear system attached to the requested window. Long finite words are not the primary certificate. A negative eigenvalue or negative link deficit should be reported with the Q3 residual, the block window, the carry-closure system, and the resulting exact or high-precision matrix entry.*

9429.6 Search Objects

Definition 9429.21 (Link-screen feasible set)**.** *For $N \geq 2$ and $d_{\max} \geq 1$, define*

$$\mathcal{C}_{N,d_{\max}} := \{v \in \mathcal{Q}_N : \Delta_d(v) \geq 0 \text{ and } D_d(v) \in \mathbb{R} \quad 1 \leq d \leq d_{\max}\}.$$

Definition 9429.22 (Window RP feasible set)**.** *For L, R, define*

$$\mathcal{R}_{N,L,R} := \left\{v \in \mathcal{Q}_N : M_{\mathcal{B}^{\text{neu}}_{L,R}}(v) \succeq 0\right\}.$$

Whenever the window contains the link observables F_d for $d \leq d_{\max}$,

$$\mathcal{R}_{N,L,R} \subseteq \mathcal{C}_{N,d_{\max}}.$$

Principle 9429.23 (Minimal RP candidate problem)**.** *Find the least alphabet size N for which there is a Q3 phase word $v \in \mathcal{Q}_N$ that remains in $\mathcal{R}_{N,L,R}$ through increasing neutral windows, beginning with*

$$(L, R) = (2, 1), (3, 1), (4, 1).$$

The complete neutral RP problem asks whether some $v \in \mathcal{Q}_N$ passes every neutral finite window. The charged RP problem asks the same without the neutrality restriction.

Principle 9429.24 (Search route)**.** *A reliable search route has four gates. First, sample exactly inside $\mathcal{Q}_N$ using Definition 9429.3 or an explicitly constrained parameterization. Second, compute $\Delta_d(v)$ for $1 \leq d \leq d_{\max}$ and reject any negative link deficit. Third, form small neutral RP windows such as $\mathcal{B}^{\text{neu}}_{2,1}$ and $\mathcal{B}^{\text{neu}}_{3,1}$ and compute their least eigenvalues. Fourth, optimize the window value*

$$\sup_{v \in \mathcal{Q}_N} \lambda_{\min} M_{\mathcal{B}^{\text{neu}}_{L,1}}(v)$$

as the window grows, rather than fixing a small-window candidate and assuming it remains optimal.

Principle 9429.25 (Layered search table)**.** *The known smallest representatives occupy the following search states:*

N	*representative row*	*Q3*	*two-point row*	*link deficit*
2	$(e^{i\pi/4}, e^{-i\pi/4})$	*yes*	*nonreal*	*charged obstruction*
3	$(e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$	*yes*	*real*	$\Delta_2 = -1/12$
4	$(1, 1, 1, -1)$	*yes*	*real*	$\Delta_1 = -16/25$
$N \geq 5$	*undetermined*	*open*	*open*	*open*

Thus the live search problem begins after the three smallest representatives have already been filtered by charged or neutral finite-window obstructions.

Principle 9429.26 (Candidate families for the next screen)**.** *Two low-dimensional $N = 5$ families give natural first screens. The symmetric pentary family has*

$$v = (a, b, c, b, a), \qquad 2a + 2b + c = \sqrt{5}.$$

The unit-root correction family has the form

$$v_a = e^{i\phi} w_a,$$

where the w_a are drawn from fifth-root phase patterns and the common rotation is chosen only when the Q3 sum is exactly $\sqrt{5}$. In either family, the first nontrivial target is simultaneous nonnegativity of Δ_1 and Δ_2.

Chapter 9430

Innovation Input Anchored RP Golden Palindrome Boundary

chapter 9428 fixes the finite-window RP matrix as an anchored future-window construction. This chapter applies that anchored audit to the lowest-complexity pentary palindrome family. The family removes the two-point arrow, but the smallest neutral anchored RP window still detects failure.

9430.1 Anchored Window Discipline

Principle 9430.1 (Anchored RP workflow)**.** *Finite-window RP entries are not functions of future-block translation classes before reflection. The workflow is*

$$\textit{anchored future block} \to \textit{reflection} \to \textit{difference} \to \textit{common translation} \to \Gamma \textit{ evaluation.}$$

Only the last step uses translation invariance of finite-block correlations.

Principle 9430.2 (Ternary anchored status)**.** *For the ternary Q3 row*

$$v = (e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6}),$$

the anchored $L = 2$ neutral window generated by

$$1, \qquad u_{n+1}\overline{u_{n+2}}, \qquad \overline{u_{n+1}}u_{n+2}$$

has matrix

$$\begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix},$$

with eigenvalues 0, $1/2$, and $3/2$. Hence the ternary row passes the anchored $L = 2$ neutral window. It still fails at anchored $L = 3$ by Definition 9428.12.

Principle 9430.3 (Quartic anchored status)**.** *For the quartic real-sign row $v = (1, 1, 1, -1)$, the anchored observable*

$$F = u_{n+1}\overline{u_{n+2}}$$

has

$$\langle \theta F, F \rangle = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3) = -\frac{3}{5}.$$

Thus the quartic representative fails the anchored $L = 2$ neutral window.

9430.2 Pentary Palindromes

Definition 9430.4 (Pentary palindrome phase word)**.** *A pentary palindrome phase word is a row*

$$v = (a, b, c, b, a) \in \mathbb{T}^5.$$

It is Q3-compatible exactly when

$$2a + 2b + c = \sqrt{5}.$$

Theorem 9430.5 (Pentary palindromes have real two-point coefficients)**.** *For every pentary palindrome $v = (a, b, c, b, a)$, the word-level two-point coefficients*

$$s_r = \sum_{j=0}^{4-r} v_{j+r}\overline{v_j}$$

are real for $1 \le r \le 4$. Consequently every Q3-compatible pentary palindrome is two-point weakly reversible.

Proof. Directly,

$$s_1 = b\overline{a} + c\overline{b} + b\overline{c} + a\overline{b} = 2\,\mathrm{Re}(b\overline{a}) + 2\,\mathrm{Re}(c\overline{b}),$$

so s_1 is real. Also

$$s_2 = c\overline{a} + b\overline{b} + a\overline{c} = 1 + 2\,\mathrm{Re}(c\overline{a}),$$

$$s_3 = b\overline{a} + a\overline{b} = 2\,\mathrm{Re}(b\overline{a}),$$

and

$$s_4 = a\overline{a} = 1.$$

Thus all s_r are real, and Definition 9426.8 gives real two-point correlations. □

Corollary 9430.6 (Pentary palindrome two-point reversibility)**.** *Every Q3-compatible pentary palindrome satisfies*

$$\gamma(m) \in \mathbb{R} \qquad \textit{for all } m \ge 0.$$

9430.3 Golden Palindrome Branches

Definition 9430.7 (Golden palindrome branches)**.** *Let*

$$\xi := e^{\pi i/5}, \qquad \zeta := e^{2\pi i/5}.$$

Define

$$v_5^{(+)} := (\xi, \xi^{-1}, -1, \xi^{-1}, \xi)$$

and

$$v_5^{(-)} := (\zeta, \zeta^{-1}, 1, \zeta^{-1}, \zeta).$$

Theorem 9430.8 (Golden palindrome Q3 rows)**.** *Both golden palindrome branches lie in $\mathcal{Q}_5$, and both have real two-point correlations.*

Proof. The trigonometric identities

$$\cos\frac{\pi}{5} = \frac{\sqrt{5}+1}{4}$$

and

$$\cos\frac{2\pi}{5} = \frac{\sqrt{5}-1}{4}$$

give

$$2\xi + 2\xi^{-1} - 1 = 4\cos\frac{\pi}{5} - 1 = \sqrt{5},$$

and

$$2\zeta + 2\zeta^{-1} + 1 = 4\cos\frac{2\pi}{5} + 1 = \sqrt{5}.$$

Thus both rows satisfy Q3. They are palindromes, so Definition 9430.5 gives real two-point correlations. □

9430.4 Anchored Neutral Window Audit

Definition 9430.9 (Pentary anchored neutral window)**.** *Use the anchored $L = 2$ neutral observables*

$$F_0 = 1,$$

$$F_1 = u_{n+1}\overline{u_{n+2}},$$

and

$$F_2 = \overline{u_{n+1}}u_{n+2}.$$

Their future blocks are

$$\alpha_0 = 0, \qquad \alpha_1 = \delta_1 - \delta_2, \qquad \alpha_2 = -\delta_1 + \delta_2.$$

Lemma 9430.10 (First golden branch window values)**.** *For*

$$v_5^{(+)} = (\xi, \xi^{-1}, -1, \xi^{-1}, \xi), \qquad \xi = e^{\pi i/5},$$

the anchored $L = 2$ neutral window has

$$\mu_1 := \langle F_1 \rangle = -\frac{1}{4},$$

$$D_1 := \langle \theta F_1, F_1 \rangle = \frac{1+\sqrt{5}}{8},$$

and

$$C_1 := \langle \theta F_1, F_2 \rangle = -\frac{1+\sqrt{5}}{8} + i\frac{\sqrt{10-2\sqrt{5}}}{8}.$$

In particular, $|C_1| = 1/2$.

Proof. The values are obtained by applying Definition 9427.11 to the anchored blocks defining μ_1, D_1, and C_1, then solving the finite carry-closure linear system described in Definition 9429.19. The cross term has modulus

$$|C_1|^2 = \frac{(1+\sqrt{5})^2}{64} + \frac{10-2\sqrt{5}}{64} = \frac{1}{4},$$

so $|C_1| = 1/2$. □

Theorem 9430.11 (First golden branch fails anchored neutral RP)**.** *The branch $v_5^{(+)}$ fails the anchored $L = 2$ neutral RP window.*

Proof. The (F_1, F_2) principal submatrix has the form

$$\begin{pmatrix} D_1 & C_1 \\ \overline{C_1} & D_1 \end{pmatrix}.$$

Its eigenvalues are $D_1 \pm |C_1|$. By Definition 9430.10,

$$D_1 - |C_1| = \frac{1+\sqrt{5}}{8} - \frac{1}{2} = \frac{\sqrt{5}-3}{8} < 0.$$

The principal submatrix is not positive semidefinite. □

Lemma 9430.12 (Second golden branch negative diagonal)**.** *For*

$$v_5^{(-)} = (\zeta, \zeta^{-1}, 1, \zeta^{-1}, \zeta), \qquad \zeta = e^{2\pi i/5},$$

one has

$$\langle \theta F_1, F_1 \rangle = \frac{1-\sqrt{5}}{8} < 0.$$

Proof. For $F_1 = u_{n+1}\overline{u_{n+2}}$, the reflected diagonal block is

$$\delta_0 - \delta_1 + \delta_2 - \delta_3.$$

Applying the finite-block carry recursion and solving its finite closure system for the second golden branch gives

$$\Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3) = \frac{1-\sqrt{5}}{8}.$$

This is negative. □

Theorem 9430.13 (Second golden branch fails anchored neutral RP)**.** *The branch $v_5^{(-)}$ fails the anchored $L = 2$ neutral RP window.*

Proof. By Definition 9430.12, the RP quadratic form is negative on the neutral future observable F_1. Hence the anchored $L = 2$ window is not positive semidefinite. □

Theorem 9430.14 (Golden palindromes are not RP candidates)**.** *The two golden palindrome Q3 rows $v_5^{(+)}$ and $v_5^{(-)}$ have real two-point correlations, but neither passes the anchored $L = 2$ neutral RP window.*

Proof. Q3 compatibility and real two-point correlations follow from Definition 9430.8. The first branch fails by Definition 9430.11; the second branch fails by Definition 9430.13. □

Principle 9430.15 (Anchored RP status table)**.** *The low-complexity Q3 representatives now have the following anchored RP status:*

N	*representative row*	*two-point row*	*anchored RP status*
2	$(e^{i\pi/4}, e^{-i\pi/4})$	*nonreal*	*charged one-site fail*
3	$(e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$	*real*	$L = 3$ *neutral fail*
4	$(1, 1, 1, -1)$	*real*	$L = 2$ *neutral fail*
5	$v_5^{(+)}$	*real*	$L = 2$ *neutral fail*
5	$v_5^{(-)}$	*real*	$L = 2$ *neutral fail*

The pentary golden branches show that additional phase freedom can remove the two-point arrow without producing an OS/RP-compatible self-substitution row.

Principle 9430.16 (Pentary full-window target)**.** *The remaining pentary question is global over $\mathcal{Q}_5$: determine whether some $v \in \mathcal{Q}_5$ passes the anchored $L = 2$ neutral RP matrix. For*

$$F = u_{n+1}\overline{u_{n+2}},$$

the basic entries are

$$\mu = \langle F \rangle, \qquad D = \langle \theta F, F \rangle, \qquad C = \langle \theta F, \overline{F} \rangle.$$

The anchored $L = 2$ matrix is

$$M_2(v) = \begin{pmatrix} 1 & \mu & \overline{\mu} \\ \overline{\mu} & D & C \\ \mu & \overline{C} & \overline{D} \end{pmatrix}.$$

In the Hermitian case, the first necessary conditions are $D \geq 0$, $D \geq |C|$, and $\det M_2(v) \geq 0$.

Chapter 9431

Innovation Input Pentary Anchored Window Boundary

chapter 9430 shows that the golden pentary palindrome endpoints fail the anchored $L = 2$ neutral RP window. This chapter records a different pentary Q3 row: it passes the anchored $L = 2$ neutral window, but fails when the window is enlarged to include the next anchored link. The obstruction is therefore not only a local link shape, but also its position relative to the reflection plane.

9431.1 Anchored Pentary Window

Definition 9431.1 (Pentary anchored $L = 2$ window)**.** *Let $v = (v_0, \dots, v_4) \in \mathcal{Q}_5$ and let*

$$u_{5n+a} = v_a u_n.$$

For

$$F = u_{n+1}\overline{u_{n+2}},$$

write

$$\alpha = \delta_1 - \delta_2.$$

The anchored $L = 2$ neutral window is

$$\mathcal{B}_2 = \{0, \alpha, -\alpha\}.$$

Its RP matrix is

$$M_2(v)_{ij} = \Gamma(\alpha_j - \theta\alpha_i),$$

where the future blocks remain anchored before reflection.

Definition 9431.2 (Anchored $L = 2$ structure quantities)**.** *Set*

$$\mu := \Gamma(\delta_0 - \delta_1),$$

$$D := \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3),$$

and

$$C := \Gamma(\delta_0 - \delta_1 - \delta_2 + \delta_3).$$

Then

$$M_2(v) = \begin{pmatrix} 1 & \mu & \overline{\mu} \\ \mu & D & C \\ \overline{\mu} & \overline{C} & \overline{D} \end{pmatrix}$$

whenever the anchored window is Hermitian.

Theorem 9431.3 (Pentary anchored $L = 2$ formulas)**.** *For $v = (v_0, \ldots, v_4) \in \mathbb{T}^5$,*

$$\mu = \frac{v_0\overline{v}_1 + v_1\overline{v}_2 + v_2\overline{v}_3 + v_3\overline{v}_4}{5 - v_4\overline{v}_0}.$$

Moreover

$$D = \frac{1}{5}\left[d_0 + d_1 + d_3 + (d_2 + d_4)\mu\right],$$

where

$$d_0 = v_0\overline{v}_1 v_2\overline{v}_3, \qquad d_1 = v_1\overline{v}_2 v_3\overline{v}_4,$$

$$d_2 = v_2\overline{v}_3 v_4\overline{v}_0, \qquad d_3 = v_3\overline{v}_4 v_0\overline{v}_1,$$

and

$$d_4 = v_4\overline{v}_0 v_1\overline{v}_2.$$

Finally,

$$C = \frac{1}{5}\left[c_0 + c_1 + c_3 + c_2\overline{\mu} + c_4\mu\right],$$

where

$$c_0 = v_0\overline{v}_1\overline{v}_2 v_3, \qquad c_1 = v_1\overline{v}_2\overline{v}_3 v_4,$$

$$c_2 = v_2\overline{v}_3\overline{v}_4 v_0, \qquad c_3 = v_3\overline{v}_4\overline{v}_0 v_1,$$

and

$$c_4 = v_4\overline{v}_0\overline{v}_1 v_2.$$

Proof. For $\mu = \Gamma(\delta_0 - \delta_1)$, split the average by $n = 5q + a$. For $0 \le a \le 3$ there is no carry, giving $v_a\overline{v}_{a+1}$. For $a = 4$ the carry contributes $v_4\overline{v}_0\mu$. Thus

$$\mu = \frac{1}{5}\left(\sum_{a=0}^{3} v_a\overline{v}_{a+1} + v_4\overline{v}_0\mu\right),$$

which gives the displayed formula.

For D, the residues $a = 0, 1, 3$ collapse to the zero block and contribute d_0, d_1, d_3. The residues $a = 2, 4$ leave the branch block $\delta_0 - \delta_1$, whose correlation is μ. This gives the displayed formula for D.

For C, the residues $a = 0, 1, 3$ collapse to the zero block and contribute c_0, c_1, c_3. The residue $a = 2$ leaves $-\delta_0 + \delta_1$, whose correlation is $\overline{\mu}$, and the residue $a = 4$ leaves $\delta_0 - \delta_1$, whose correlation is μ. □

9431.2 Pythagorean Pentary Row

Definition 9431.4 (Pythagorean pentary phase word)**.** *Let*

$$a := \frac{1+2i}{\sqrt{5}}, \qquad b := \frac{2-i}{\sqrt{5}}.$$

Define

$$v_\star := (a, b, -a, b, a).$$

Theorem 9431.5 (Pythagorean row is Q3)**.** *The row $v_\star$ lies in $\mathcal{Q}_5$.*

Proof. Directly,

$$a + 2b = \frac{1+2i}{\sqrt{5}} + 2\frac{2-i}{\sqrt{5}} = \frac{5}{\sqrt{5}} = \sqrt{5}.$$

Since the row sum of $(a, b, -a, b, a)$ is $a + 2b$, the Q3 equation holds. □

Lemma 9431.6 (Pythagorean products)**.** *For $a = (1+2i)/\sqrt{5}$ and $b = (2-i)/\sqrt{5}$,*

$$a\overline{b} = i, \qquad b\overline{a} = -i.$$

Proof. Since $\overline{b} = (2+i)/\sqrt{5}$,

$$a\overline{b} = \frac{(1+2i)(2+i)}{5} = \frac{5i}{5} = i.$$

The second identity is the complex conjugate. □

9431.3 Anchored $L = 2$ Passage

Theorem 9431.7 (Pythagorean anchored $L = 2$ structure)**.** *For $v_\star = (a, b, -a, b, a)$,*

$$\mu = 0, \qquad D = \frac{3}{5}, \qquad C = -\frac{3}{5}.$$

Therefore

$$M_2(v_\star) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{3}{5} & -\frac{3}{5} \\ 0 & -\frac{3}{5} & \frac{3}{5} \end{pmatrix}.$$

Proof. Using Definition 9431.6,

$$v_0\overline{v}_1 = i, \qquad v_1\overline{v}_2 = i, \qquad v_2\overline{v}_3 = -i, \qquad v_3\overline{v}_4 = -i.$$

The numerator in the formula for μ is therefore zero, so $\mu = 0$.

For D, the terms involving μ vanish. The remaining products are

$$d_0 = (a\overline{b})((-a)\overline{b}) = 1,$$

$$d_1 = (b\overline{(-a)})(b\overline{a}) = 1,$$

and

$$d_3 = (b\overline{a})(a\overline{b}) = 1.$$

Thus $D = (1+1+1)/5 = 3/5$.

For C, again the terms involving μ vanish. The remaining products are

$$c_0 = (a\overline{b})(\overline{(-a)}b) = -1,$$

$$c_1 = (b\overline{(-a)})(\overline{b}a) = -1,$$

and

$$c_3 = (b\overline{a})(\overline{a}b) = -1.$$

Thus $C = -3/5$. □

Corollary 9431.8 (Pythagorean row passes anchored $L = 2$ neutral RP)**.** *The anchored $L = 2$ neutral RP matrix of $v_\star$ is positive semidefinite.*

Proof. The lower-right block is

$$\frac{3}{5}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix},$$

with eigenvalues 0 and 6/5. Together with the top-left entry 1, the matrix eigenvalues are 1, 0, and 6/5. □

9431.4 Anchored $L = 3$ Failure

Definition 9431.9 (Second anchored link)**.** *Set*

$$F_{23}(n) := \overline{u_{n+2}}u_{n+3}.$$

Its future block is

$$\alpha_{23} := -\delta_2 + \delta_3.$$

The reflected diagonal quantity is

$$D_{23} := \langle \theta F_{23}, F_{23} \rangle.$$

Since $\theta\delta_2 = \delta_{-1}$ and $\theta\delta_3 = \delta_{-2}$,

$$\alpha_{23} - \theta\alpha_{23} = -\delta_2 + \delta_3 + \delta_{-1} - \delta_{-2}.$$

After translating by 2, this is the block

$$-\delta_0 + \delta_1 - \delta_4 + \delta_5.$$

Theorem 9431.10 (Pythagorean anchored $L = 3$ failure)**.** *For $v_\star = (a, b, -a, b, a)$,*

$$D_{23} = \Gamma(-\delta_0 + \delta_1 - \delta_4 + \delta_5) = -\frac{1}{5}.$$

Thus $v_\star$ fails the anchored $L = 3$ neutral RP window.

Proof. Let $\beta = -\delta_0 + \delta_1 - \delta_4 + \delta_5$. Split the recursion by residue $r = 0, \ldots, 4$.

For $r = 0$, the coefficient is

$$\overline{v}_0 v_1 \overline{v}_4 v_0 = \overline{a}b\overline{a}a = -i,$$

and the branch block is $-\delta_0 + \delta_1$, whose correlation is $\overline{\mu} = 0$.

For $r = 1$, the coefficient is

$$\overline{v}_1 v_2 \overline{v}_0 v_1 = \overline{b}(-a)\overline{a}b = -1,$$

and the branch is the zero block.

For $r = 2$, the coefficient is

$$\overline{v}_2 v_3 \overline{v}_1 v_2 = \overline{(-a)} b \overline{b} (-a) = 1,$$

and the branch is the zero block.

For $r = 3$, the coefficient is

$$\overline{v}_3 v_4 \overline{v}_2 v_3 = \overline{b} a \overline{(-a)} b = -1,$$

and the branch is the zero block.

For $r = 4$, the coefficient is

$$\overline{v}_4 v_0 \overline{v}_3 v_4 = \overline{a} a \overline{b} a = i,$$

and the branch block is again $-\delta_0 + \delta_1$, whose correlation is $\overline{\mu} = 0$.

Therefore

$$D_{23} = \frac{1}{5}(0 - 1 + 1 - 1 + 0) = -\frac{1}{5}.$$

The negative diagonal entry violates neutral RP. □

Corollary 9431.11 (Anchored $L = 2$ does not imply anchored $L = 3$)**.** *There exists $v \in \mathcal{Q}_5$ such that its anchored $L = 2$ neutral RP matrix is positive semidefinite, while its anchored $L = 3$ neutral RP window fails.*

Proof. Use $v = v_\star$. The anchored $L = 2$ matrix is positive semidefinite by Definition 9431.8, while Definition 9431.10 gives a negative anchored $L = 3$ diagonal entry. □

9431.5 Anchor Dependence

Definition 9431.12 (Anchored link diagonal)**.** *For $1 \le i < j$, define*

$$F_{ij} := u_{n+i} \overline{u_{n+j}},$$

and

$$D_{ij} := \langle \theta F_{ij}, F_{ij} \rangle.$$

Theorem 9431.13 (Link diagonal depends on anchor)**.** *For the Pythagorean row,*

$$D_{12} = \frac{3}{5}, \qquad D_{23} = -\frac{1}{5}.$$

Proof. The value $D_{12} = 3/5$ is the D value in Definition 9431.7. The value $D_{23} = -1/5$ is Definition 9431.10. □

Principle 9431.14 (Anchor-sensitive RP audit)**.** *Ordinary finite-block correlations satisfy*

$$\Gamma(T_s \alpha) = \Gamma(\alpha),$$

but RP diagonals are of the form

$$\Gamma(\alpha - \theta\alpha).$$

Because $\theta T_s = T_{-s}\theta$, translating a future block changes its relative position to the reflection plane. A complete finite-window audit must therefore check every anchored link, not only each translation shape.

Principle 9431.15 (Pentary anchored status table)**.** *The pentary anchored search status is:*

N	*row*	*Q3*	$L = 2$ *neutral RP*	$L = 3$ *neutral RP*
3	$(e^{i\pi/6}, e^{-i\pi/2}, e^{i\pi/6})$	*yes*	*pass*	*fail*
4	$(1, 1, 1, -1)$	*yes*	*fail*	*not needed*
5	$v_5^{(+)}$	*yes*	*fail*	*not needed*
5	$v_5^{(-)}$	*yes*	*fail*	*not needed*
5	$v_\star = (a, b, -a, b, a)$	*yes*	*pass*	*fail*

Thus the pentary search space contains rows that pass the nearest anchored neutral RP window, but the next anchor can still obstruct reflection positivity.

Principle 9431.16 (Pentary $L = 3$ target)**.** *The full pentary target is the $\mathcal{Q}_5$ feasibility problem for anchored $L = 3$ neutral RP. At minimum one must have*

$$D_{12} \geq 0, \qquad D_{13} \geq 0, \qquad D_{23} \geq 0.$$

The Pythagorean row has $D_{12} = 3/5$ but $D_{23} = -1/5$, so improving the candidate requires moving the second anchored link diagonal to the nonnegative side without destroying the anchored $L = 2$ matrix.

Chapter 9432

Innovation Input Pentary Hermitian Link Boundary

chapter 9431 separates anchored $L = 2$ passage from anchored $L = 3$ failure for one pentary Q3 row. This chapter isolates the first $L = 3$ algebraic filters: the three anchored link diagonals, the endpoint gate carried by the middle link, and the correlation inequality forced by the second link.

9432.1 Pentary Edge Ratios

Definition 9432.1 (Pentary edge ratios)**.** *Let*

$$v = (v_0, v_1, v_2, v_3, v_4) \in \mathbb{T}^5.$$

Define the cyclic edge ratios

$$x_a := v_a \overline{v_{a+1}}, \qquad a = 0, 1, 2, 3, 4,$$

where indices are read modulo 5*. Thus*

$$x_4 = v_4 \overline{v_0}.$$

Then

$$|x_a| = 1$$

and

$$x_0 x_1 x_2 x_3 x_4 = 1.$$

Definition 9432.2 (First and second ordinary correlations)**.** *For the self-substitution row v, write*

$$\mu := \Gamma(\delta_0 - \delta_1)$$

and

$$\nu := \Gamma(\delta_0 - \delta_2).$$

Equivalently, μ is the ordinary distance-one average $\langle u_n \overline{u_{n+1}} \rangle$, while ν is the ordinary distance-two average $\langle u_n \overline{u_{n+2}} \rangle$.

Theorem 9432.3 (Pentary first correlation formula)**.** *For every pentary self-substitution row,*

$$\mu = \frac{x_0 + x_1 + x_2 + x_3}{5 - x_4}.$$

Proof. Split the average defining $\Gamma(\delta_0 - \delta_1)$ by $n = 5q + a$. For $a = 0, 1, 2, 3$ there is no carry, and the contribution is x_a. For $a = 4$ the carry leaves the same distance-one block and contributes $x_4\mu$. Hence

$$\mu = \frac{1}{5}\left(x_0 + x_1 + x_2 + x_3 + x_4\mu\right),$$

which is the stated formula. □

Theorem 9432.4 (Pentary second correlation formula)**.** *For every pentary self-substitution row,*

$$\nu = \frac{1}{5}\left[x_0x_1 + x_1x_2 + x_2x_3 + (x_3x_4 + x_4x_0)\mu\right].$$

Proof. Split the average defining $\Gamma(\delta_0 - \delta_2)$ by $n = 5q + a$. For $a = 0, 1, 2$ the residues do not cross the next pentary block, giving the factors x_0x_1, x_1x_2, and x_2x_3. For $a = 3$ the carry leaves the branch block $\delta_0 - \delta_1$ and contributes $x_3x_4\mu$. For $a = 4$ it contributes $x_4x_0\mu$. Averaging gives the formula. □

9432.2 Anchored $L = 3$ Link Diagonals

Definition 9432.5 (Forward anchored links)**.** *Use the forward neutral links*

$$F_{ij} := u_{n+i}\overline{u_{n+j}}, \qquad 1 \le i < j \le 3,$$

and define

$$D_{ij} := \langle \theta F_{ij}, F_{ij} \rangle.$$

After the common translation normalization, the three link diagonals are

$$D_{12} = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3),$$

$$D_{13} = \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_5),$$

and

$$D_{23} = \Gamma(\delta_0 - \delta_1 + \delta_4 - \delta_5).$$

The opposite orientation of a link conjugates the displayed block; for real diagonal obstructions the numerical sign information is unchanged.

Theorem 9432.6 (Pentary D_{12} formula)**.** *For every pentary self-substitution row,*

$$D_{12} = \frac{1}{5}\left[x_0x_2 + x_1x_3 + x_3x_0 + (x_2x_4 + x_4x_1)\mu\right].$$

Proof. Apply the pentary carry split to $\delta_0 - \delta_1 + \delta_2 - \delta_3$. The five residues give

$$a = 0: \ x_0x_2, \qquad a = 1: \ x_1x_3,$$

$$a = 2: \ x_2x_4\,\mu, \qquad a = 3: \ x_3x_0,$$

and

$$a = 4: \ x_4x_1\,\mu.$$

Averaging gives the formula. □

Theorem 9432.7 (Pentary D_{13} formula)**.** *For every pentary self-substitution row,*

$$D_{13} = \frac{1}{5}\left[\overline{x_4} + (\overline{x_0} + \overline{x_1} + \overline{x_2} + \overline{x_3})\mu\right].$$

Proof. Apply the carry split to $\delta_0 - \delta_2 + \delta_3 - \delta_5$. The residues $a = 0, 1, 3, 4$ leave the branch block $\delta_0 - \delta_1$ with coefficients $\overline{x_2}$, $\overline{x_3}$, $\overline{x_0}$, and $\overline{x_1}$ respectively. The residue $a = 2$ collapses to the zero block with coefficient $\overline{x_4}$. Since the four nonzero branch coefficients are the same cyclic set $\overline{x_0}, \overline{x_1}, \overline{x_2}, \overline{x_3}$, averaging gives the stated expression. □

Theorem 9432.8 (Pentary D_{23} correlation meeting)**.** *For every pentary self-substitution row,*

$$D_{23} = \nu.$$

Equivalently,

$$D_{23} = \frac{1}{5}\left[x_0x_1 + x_1x_2 + x_2x_3 + (x_3x_4 + x_4x_0)\mu\right].$$

Proof. Apply the carry split to $\delta_0 - \delta_1 + \delta_4 - \delta_5$. The five residues give

$$a = 0: \ x_4x_0\,\mu, \qquad a = 1: \ x_0x_1,$$

$$a = 2: \ x_1x_2, \qquad a = 3: \ x_2x_3,$$

and

$$a = 4: \ x_3x_4\,\mu.$$

The resulting average is exactly the formula for ν in Definition 9432.4. □

Theorem 9432.9 (Three-link necessary condition)**.** *If a pentary row passes the anchored $L = 3$ neutral RP window, then*

$$D_{12} \in \mathbb{R}_{\geq 0}, \qquad D_{13} \in \mathbb{R}_{\geq 0}, \qquad D_{23} \in \mathbb{R}_{\geq 0},$$

and

$$D_{12} \geq |\mu|^2, \qquad D_{13} \geq |\nu|^2, \qquad D_{23} \geq |\mu|^2.$$

In particular,

$$\nu \in \mathbb{R}, \qquad \nu \geq |\mu|^2.$$

Proof. For each link F_{ij}, take the 2×2 principal RP matrix on $1, F_{ij}$. Positivity forces the lower-right entry to be real nonnegative and forces the determinant inequality. For F_{12} and F_{23} the ordinary mean is μ, while for F_{13} it is ν. The displayed inequalities follow. The final statement uses Definition 9432.8. □

9432.3 Endpoint Gate

Theorem 9432.10 (Endpoint form of the middle link)**.** *For every pentary self-substitution row,*

$$D_{13} = |\mu|^2 + \frac{1 - |\mu|^2}{5}\overline{x_4}.$$

Proof. By Definition 9432.3,

$$x_0 + x_1 + x_2 + x_3 = (5 - x_4)\mu.$$

Taking conjugates gives

$$\overline{x_0} + \overline{x_1} + \overline{x_2} + \overline{x_3} = (5 - \overline{x_4})\overline{\mu}.$$

Substitute this identity into Definition 9432.7:

$$D_{13} = \frac{1}{5}\left[\overline{x_4} + (5 - \overline{x_4})|\mu|^2\right].$$

Rearranging gives the displayed formula. □

Lemma 9432.11 (Q3 excludes unit first correlation)**.** *If $v \in \mathcal{Q}_5$, then*

$$|\mu| < 1.$$

Proof. Assume $|\mu| = 1$. The recursion in the proof of Definition 9432.3 writes μ as the average of the five unit complex numbers

$$x_0, \quad x_1, \quad x_2, \quad x_3, \quad x_4\mu.$$

An average of unit complex numbers has unit modulus only when all terms are equal. Hence

$$x_0 = x_1 = x_2 = x_3 = \mu, \qquad x_4\mu = \mu.$$

Since $\mu \neq 0$, this gives $x_4 = 1$. The cyclic product relation then gives

$$\mu^4 = x_0x_1x_2x_3x_4 = 1.$$

Writing $\rho := \mu$, the edge-ratio equations force

$$v = v_0(1, \overline{\rho}, \overline{\rho}^2, \overline{\rho}^3, \overline{\rho}^4), \qquad \rho^4 = 1.$$

If $\rho = 1$, the row sum has modulus 5. If $\rho \in \{-1, i, -i\}$, the row sum has modulus 1. None of these moduli is $\sqrt{5}$, contradicting $v \in \mathcal{Q}_5$. Therefore $|\mu| < 1$. □

Theorem 9432.12 (Pentary endpoint gate)**.** *If a row $v \in \mathcal{Q}_5$ passes the anchored $L = 3$ neutral RP window, then*

$$x_4 \in \{1, -1\}.$$

Equivalently,

$$v_4 = v_0 \qquad \text{or} \qquad v_4 = -v_0.$$

Proof. Reflection positivity forces the diagonal entry D_{13} to be real nonnegative by Definition 9432.9. By Definition 9432.10,

$$D_{13} = |\mu|^2 + \frac{1 - |\mu|^2}{5}\overline{x_4}.$$

The first term is real, and Definition 9432.11 gives

$$1 - |\mu|^2 > 0.$$

Thus D_{13} is real only if $\overline{x_4}$ is real. Since $|x_4| = 1$, this means $x_4 = \pm 1$. The identity $x_4 = v_4\overline{v_0}$ is then equivalent to $v_4 = \pm v_0$. □

Definition 9432.13 (Endpoint branches)**.** *The even endpoint branch is the sublocus of $\mathcal{Q}_5$ on which*

$$v_4 = v_0,$$

equivalently $x_4 = 1$. On this branch the Q3 equation reads

$$2v_0 + v_1 + v_2 + v_3 = \sqrt{5}.$$

The odd endpoint branch is the sublocus of $\mathcal{Q}_5$ on which

$$v_4 = -v_0,$$

equivalently $x_4 = -1$. On this branch the Q3 equation reads

$$v_1 + v_2 + v_3 = \sqrt{5}.$$

Corollary 9432.14 (Endpoint branch form of candidates)**.** *Every anchored $L = 3$ neutral RP candidate in $\mathcal{Q}_5$ lies in one of the two endpoint forms*

$$v = (v_0, v_1, v_2, v_3, v_0)$$

or

$$v = (v_0, v_1, v_2, v_3, -v_0).$$

The palindromic locus is the sublocus of the even endpoint branch where $v_1 = v_3$.

Proof. This is exactly Definition 9432.12, with the palindromic condition read from Definition 9439.6. □

Corollary 9432.15 (Middle-link values on endpoint branches)**.** *On the even endpoint branch,*

$$D_{13} = \frac{1 + 4|\mu|^2}{5}.$$

On the odd endpoint branch,

$$D_{13} = \frac{6|\mu|^2 - 1}{5}.$$

Consequently, an anchored $L = 3$ neutral RP candidate on the odd endpoint branch must satisfy

$$|\mu|^2 \geq \frac{1}{6}.$$

Proof. Substitute $x_4 = 1$ and $x_4 = -1$ respectively into Definition 9432.10. The odd endpoint lower bound is the condition $D_{13} \geq 0$ from Definition 9432.9. □

Principle 9432.16 (Endpoint-first pentary search)**.** *The first scalar diagonal gate for the anchored $L = 3$ neutral RP problem does not inspect the full 7×7 matrix. It already confines every candidate to the even or odd endpoint branch. The even endpoint branch still contains the palindromic rows, while the odd endpoint branch leaves the endpoint phase outside the Q3 equation; any candidate there must have that phase fixed by the Hermitian and positive-semidefinite gates rather than by Q3 alone.*

Corollary 9432.17 (Algebraic form of the anchor effect)**.** *The two links F_{12} and F_{23} have the same ordinary distance-one mean μ, but their RP diagonals need not agree. In the pentary $L = 3$ window,*

$$D_{23} = \Gamma(\delta_0 - \delta_2),$$

so reflection positivity forces the concrete inequality

$$\Gamma(\delta_0 - \delta_2) \geq |\Gamma(\delta_0 - \delta_1)|^2 .$$

9432.4 Pythagorean Link Recheck

Corollary 9432.18 (Pythagorean row violates the second link)**.** *For the Pythagorean row* $v_\star = (a, b, -a, b, a)$ *from Definition 9431.4,*

$$\mu = 0, \qquad D_{12} = \frac{3}{5}, \qquad D_{23} = \nu = -\frac{1}{5}.$$

Thus the anchored $L = 3$ *obstruction is the negative ordinary distance-two correlation.*

Proof. The values $\mu = 0$ and $D_{12} = 3/5$ are Definition 9431.7. The value $D_{23} = -1/5$ is Definition 9431.10, conjugated to the forward-link convention of Definition 9432.5. Since the value is real, the obstruction is unchanged. The identity $D_{23} = \nu$ then identifies the failing scalar with the ordinary distance-two correlation. □

Chapter 9433

Innovation Input Pentary Odd Endpoint Exclusion

Definition 9432.12 confines every anchored $L = 3$ neutral RP candidate in $\mathcal{Q}_5$ to one of two endpoint branches. This chapter discharges the odd endpoint branch. The argument uses only the first reflected link mean and the middle-link diagonal: Hermitian symmetry makes the distance-one mean real, while the odd endpoint Q3 geometry makes any real value too small to pass the D_{13} positivity gate.

9433.1 Odd Endpoint Coordinates

Definition 9433.1 (Odd endpoint relative coordinates)**.** *Assume $v \in \mathcal{Q}_5$ lies on the odd endpoint branch of Definition 9432.13. Write*

$$v = (a, b, c, d, -a), \qquad |a| = |b| = |c| = |d| = 1.$$

Define the relative internal phases

$$B := \frac{b}{a}, \qquad C := \frac{c}{a}, \qquad D := \frac{d}{a},$$

and the internal edge ratios

$$p := B\overline{C}, \qquad q := C\overline{D}.$$

Then $B, C, D, p, q \in \mathbb{T}$ and

$$B\overline{D} = pq.$$

Theorem 9433.2 (Odd endpoint Q3 edge equation)**.** *On the odd endpoint branch, the Q3 condition is equivalent to*

$$\mathrm{Re}(p + q + pq) = 1.$$

Proof. Since $v_4 = -v_0$, the Q3 equation becomes

$$b + c + d = \sqrt{5}.$$

Multiplying by $\overline{a}$ gives

$$|B + C + D| = \sqrt{5}.$$

Squaring the modulus gives

$$|B + C + D|^2 = 3 + 2\,\mathrm{Re}\left(B\overline{C} + C\overline{D} + B\overline{D}\right).$$

Using $B\overline{C} = p$, $C\overline{D} = q$, and $B\overline{D} = pq$, the condition $|B + C + D|^2 = 5$ is exactly

$$\mathrm{Re}(p + q + pq) = 1.$$

□

Theorem 9433.3 (Odd endpoint first correlation)**.** *On the odd endpoint branch,*

$$6\mu = \overline{B} + p + q - D = \overline{B} + p + q - \frac{B}{pq}.$$

Proof. For $v = (a, b, c, d, -a)$, the cyclic edge ratios of Definition 9432.1 are

$$x_0 = a\overline{b} = \overline{B}, \qquad x_1 = b\overline{c} = p,$$

$$x_2 = c\overline{d} = q, \qquad x_3 = d\overline{(-a)} = -D,$$

and

$$x_4 = (-a)\overline{a} = -1.$$

Substitution into Definition 9432.3 gives

$$\mu = \frac{x_0 + x_1 + x_2 + x_3}{5 - x_4} = \frac{\overline{B} + p + q - D}{6}.$$

Finally $pq = B\overline{D}$ implies $D = B/(pq)$. □

9433.2 Real First-Correlation Bound

Lemma 9433.4 (Odd endpoint real mean bound)**.** *On the odd endpoint Q3 branch, if*

$$\mu \in \mathbb{R},$$

then

$$|\mu| \le \frac{1}{10}.$$

Proof. Write

$$p = e^{i(\eta+\delta)}, \qquad q = e^{i(\eta-\delta)}.$$

Then $pq = e^{2i\eta}$, and Definition 9433.2 becomes

$$2\cos\delta\cos\eta + \cos 2\eta = 1.$$

Set

$$c := \cos\eta, \qquad y := c^2.$$

The displayed equation is

$$\cos\delta\, c = 1 - c^2.$$

In particular $c \neq 0$, and

$$\cos\delta = \frac{1-y}{c}.$$

Write $B = e^{i(\eta+\varphi)}$. By Definition 9433.3,

$$6\mu = p + q + \overline{B} - \frac{B}{pq}.$$

Since

$$p + q = 2\cos\delta\, e^{i\eta}$$

and

$$\overline{B} - \frac{B}{pq} = -2ie^{-i\eta}\sin\varphi,$$

one has

$$6\mu = 2\cos\delta\, e^{i\eta} - 2ie^{-i\eta}\sin\varphi.$$

Taking imaginary parts and using $\mu \in \mathbb{R}$ gives

$$\cos\delta\sin\eta = \cos\eta\sin\varphi.$$

Thus

$$|\sin\varphi| = |\cos\delta\tan\eta| = \frac{(1-y)^{3/2}}{y} \leq 1.$$

Hence

$$(1-y)^3 \leq y^2.$$

The function $(1-y)^3 - y^2$ is strictly decreasing on $[0,1]$, and at $y = 2/5$ it equals

$$\left(\frac{3}{5}\right)^3 - \left(\frac{2}{5}\right)^2 = \frac{7}{125} > 0.$$

Therefore

$$y > \frac{2}{5}.$$

Taking real parts in the displayed formula for 6μ and substituting $\sin\varphi = \cos\delta\tan\eta$ gives

$$6\mu = 2\cos\delta\left(\cos\eta - \frac{\sin^2\eta}{\cos\eta}\right) = 2\cos\delta\frac{\cos 2\eta}{\cos\eta}.$$

Using $\cos\delta = (1-y)/\cos\eta$ yields

$$\mu = \frac{(1-y)(2y-1)}{3y}.$$

For $2/5 \leq y \leq 1/2$,

$$|\mu| = \frac{(1-y)(1-2y)}{3y}.$$

The inequality $|\mu| \leq 1/10$ is equivalent to

$$20y^2 - 33y + 10 \leq 0,$$

whose roots are $2/5$ and $5/4$, so it holds on this interval. For $1/2 \le y \le 1$,

$$\mu = \frac{(1-y)(2y-1)}{3y},$$

and $\mu \le 1/10$ is equivalent to

$$20y^2 - 27y + 10 \ge 0.$$

This quadratic has discriminant -71 and positive leading coefficient, so it is positive for all real y. The bound follows. □

9433.3 Odd Endpoint Exclusion

Theorem 9433.5 (No odd endpoint anchored $L = 3$ candidate)**.** *There is no row*

$$v \in \mathcal{Q}_5$$

on the odd endpoint branch that passes the anchored $L = 3$ neutral RP window.

Proof. Assume that such a row passes anchored $L = 3$ neutral RP. Then the RP matrix is Hermitian. Applying the Hermitian condition to the subwindow generated by 0 and e_{12} forces the distance-one mean

$$\mu = \Gamma(\delta_0 - \delta_1)$$

to be real. Hence Definition 9433.4 gives

$$|\mu|^2 \le \frac{1}{100}.$$

On the odd endpoint branch, Definition 9432.15 gives

$$D_{13} = \frac{6|\mu|^2 - 1}{5}.$$

Therefore

$$D_{13} \le \frac{6/100 - 1}{5} = -\frac{47}{250} < 0.$$

This contradicts the required nonnegativity of D_{13} in Definition 9432.9. □

Corollary 9433.6 (Pentary candidates reduce to the even endpoint branch)**.** *Every row in*

$$\mathcal{R}^{\mathrm{neu}}_{5,3,1}$$

lies on the even endpoint branch

$$v_4 = v_0.$$

Proof. By the endpoint gate of Definition 9432.12, any anchored $L = 3$ neutral RP candidate has $v_4 = v_0$ or $v_4 = -v_0$. The odd endpoint case is impossible by Definition 9433.5. □

Theorem 9433.7 (Even endpoint exclusion closes the pentary three-link class)**.** *Assume that no row on the even endpoint branch*

$$v_4 = v_0$$

passes the anchored $L = 3$ neutral RP window. Then

$$\mathcal{R}^{\mathrm{neu}}_{5,3,1} = \varnothing.$$

Proof. By Definition 9433.6, every row in $\mathcal{R}^{\mathrm{neu}}_{5,3,1}$ lies on the even endpoint branch. The stated even endpoint exclusion therefore leaves no row in $\mathcal{R}^{\mathrm{neu}}_{5,3,1}$. □

Principle 9433.8 (Pentary branch ledger)**.** *The anchored pentary $L = 3$ neutral RP problem has a branch-level structure. The middle-link diagonal gives the endpoint dichotomy $v_4 = \pm v_0$. The odd endpoint branch is ruled out by the incompatibility between Hermitian realness of μ and $D_{13} \geq 0$. The remaining unconditional branch is the even endpoint branch; once it is excluded, the whole pentary three-link class is empty.*

Chapter 9434

Innovation Input Nine-ary Real-Sign RP Screen

Definition 9426.10 shows that every square alphabet admits Q3-compatible real-sign self-substitution rows. These rows are automatically two-point weakly reversible by Definition 9426.9. The first square alphabet beyond the quartic endpoint is $N = 9$. This chapter records the finite-window RP screen for that real-sign subfamily: many rows pass the anchored $L = 2$ neutral test, but no row survives the anchored $L = 3$ real neutral window.

9434.1 Nine-ary Real-Sign Rows

Definition 9434.1 (Nine-ary real-sign Q3 row)**.** *A nine-ary real-sign Q3 row is a word*

$$v = (v_0, \dots, v_8) \in \{+1, -1\}^9$$

satisfying

$$\sum_{a=0}^{8} v_a = 3.$$

It determines the real-sign self-substitution sequence

$$u_{9n+a} = v_a u_n, \qquad u_n \in \{+1, -1\}.$$

Proposition 9434.2 (Three negative signs)**.** *A word $v \in \{+1, -1\}^9$ is a nine-ary real-sign Q3 row if and only if it has six entries equal to $+1$ and three entries equal to -1. Hence there are*

$$\binom{9}{3} = 84$$

such rows.

Proof. Let P and M be the numbers of positive and negative entries. Then

$$P + M = 9$$

and the Q3 condition is

$$P - M = 3.$$

Thus $P = 6$ and $M = 3$. The converse is immediate, and the count is the choice of the three negative positions. □

9434.2 Real Neutral Window

Definition 9434.3 (Real-sign support block)**.** *For a finite support $S \subset \mathbb{Z}$, set*

$$U_S(n) := \prod_{j \in S} u_{n+j}.$$

Only the parity of site multiplicity matters, since $u_n^2 = 1$. Write

$$\Gamma(S) := \langle U_S \rangle.$$

The empty support satisfies $\Gamma(\varnothing) = 1$, and common translation of S does not change $\Gamma(S)$.

Definition 9434.4 (Anchored real $L = 3$ neutral window)**.** *The real-sign anchored $L = 3$ neutral window is*

$$\mathcal{B}_3^{\mathbb{R}} = \{1, F_{12}, F_{13}, F_{23}\},$$

where

$$F_{12} = u_{n+1}u_{n+2}, \qquad F_{13} = u_{n+1}u_{n+3}, \qquad F_{23} = u_{n+2}u_{n+3}.$$

This is the real collapse of the complex neutral window $\{1, F_{12}, F_{13}, F_{23}, \overline{F}_{12}, \overline{F}_{13}, \overline{F}_{23}\}$, because $u_n^{-1} = u_n$.

Definition 9434.5 (Nine-ary real neutral RP matrix)**.** *Let*

$$F_0 := 1, \qquad F_1 := F_{12}, \qquad F_2 := F_{13}, \qquad F_3 := F_{23}.$$

The anchored real neutral $L = 3$ RP matrix is

$$M_3^{\mathbb{R}}(v)_{ij} := \langle \theta F_i, F_j \rangle, \qquad 0 \le i, j \le 3,$$

where $\theta(j) = 1 - j$ on anchored future sites, as in Definition 9428.1. The row passes the screen when $M_3^{\mathbb{R}}(v)$ is symmetric and positive semidefinite.

9434.3 Exact Carry Evaluator

Definition 9434.6 (Nine-ary carry branch)**.** *Let $S \subset \mathbb{Z}_{\ge 0}$ be finite and let $a \in \{0, \dots, 8\}$. For each $j \in S$, write*

$$a + j = 9q_{j,a} + r_{j,a}, \qquad 0 \le r_{j,a} < 9.$$

Set

$$C_a(S) := \prod_{j \in S} v_{r_{j,a}}.$$

Let S_a be the set of quotient positions $q_{j,a}$ that occur with odd multiplicity, translated so that its least element is 0 when it is nonempty.

Theorem 9434.7 (Real-sign finite carry recursion)**.** *For every finite support $S \subset \mathbb{Z}_{\ge 0}$,*

$$\Gamma(S) = \frac{1}{9} \sum_{a=0}^{8} C_a(S)\Gamma(S_a).$$

For any fixed finite window, repeated branching generates a finite set of normalized supports, and the associated correlations are the unique solution of a finite rational linear system.

Proof. Write $n = 9q + a$. Then

$$n + j = 9(q + q_{j,a}) + r_{j,a},$$

and the self-substitution rule gives

$$u_{n+j} = v_{r_{j,a}} u_{q+q_{j,a}}.$$

Multiplying over $j \in S$ yields

$$U_S(n) = C_a(S) \prod_{j \in S} u_{q+q_{j,a}}.$$

Repeated quotient positions cancel in pairs because $u_m^2 = 1$, leaving the parity support S_a. Averaging over the nine residues gives the displayed recursion.

If a support has diameter d, every branch support has diameter at most $\lceil d/9 \rceil$. After finitely many branches the diameter is confined to a finite range determined by the original window. Hence the closure is finite. Since all coefficients are rational signs divided by 9, the finite system is rational. Uniqueness follows by evaluation as the Cesaro limit of the substitution sequence; equivalently, the displayed finite system is the exact correlation system generated by that limit. □

9434.4 Two-Window Passage

Proposition 9434.8 (Anchored $L = 2$ real screen)**.** *Among the* 84 *nine-ary real-sign Q3 rows, exactly* 31 *pass the anchored* $L = 2$ *real neutral RP screen for the window* $\{1, F_{12}\}$.

Proof. For the two-window, write

$$\mu := \Gamma(\{0, 1\})$$

and

$$D_{12} := \Gamma(\{0, 1, 2, 3\}).$$

The anchored 2×2 matrix is

$$\begin{pmatrix} 1 & \mu \\ \mu & D_{12} \end{pmatrix},$$

so passage is equivalent to

$$D_{12} \geq 0 \qquad \text{and} \qquad D_{12} \geq \mu^2.$$

The exact rational evaluator of Definition 9434.7, applied to the 84 choices of the three negative positions, gives 31 passing rows. The reproducible enumeration is `papers/bedc/experiments/nine_real_sign_rp_s` □

Proposition 9434.9 (A strict two-window witness)**.** *Let the negative positions be*

$$I = \{0, 1, 3\},$$

so that

$$v = (-1, -1, 1, -1, 1, 1, 1, 1, 1).$$

Then

$$\mu = \frac{1}{5}, \qquad D_{12} = \frac{1}{9},$$

and therefore

$$D_{12} - \mu^2 = \frac{16}{225} > 0.$$

Proof. These values are the exact solution of the carry system generated by Definition 9434.7 for the supports $\{0, 1\}$ and $\{0, 1, 2, 3\}$. The same script cited in Definition 9434.8 recomputes the fractions directly. □

9434.5 Three-Window Exclusion

Theorem 9434.10 (No nine-ary real-sign $L = 3$ RP row)**.** *No nine-ary real-sign Q3 row passes the anchored real neutral $L = 3$ RP screen:*

$$\begin{aligned}\{v \in \{+1,-1\}^9 : \sum_{a=0}^{8} v_a = 3,\ M_3^{\mathbb{R}}(v) = M_3^{\mathbb{R}}(v)^{\mathsf{T}},\ M_3^{\mathbb{R}}(v) \succeq 0\} \\ = \varnothing.\end{aligned}$$

Proof. By Definition 9434.2, there are 84 candidates. The exact carry evaluator computes each matrix $M_3^{\mathbb{R}}(v)$. Of these matrices, 70 are not symmetric and therefore cannot define a real reflection-positive quadratic form.

The remaining 14 matrices are symmetric. Each fails positive semidefiniteness by a negative diagonal entry or by a negative principal 2×2 determinant. With row and column indices ordered as

$$0 \leftrightarrow 1, \qquad 1 \leftrightarrow F_{12}, \qquad 2 \leftrightarrow F_{13}, \qquad 3 \leftrightarrow F_{23},$$

the exact certificates are:

negative positions I	negative certificate
$\{0,1,2\}$	$\det M_{\{0,1\}} = -\frac{4}{25}$
$\{0,2,4\}$	$M_{33} = -\frac{3}{5}$
$\{0,4,5\}$	$M_{11} = -\frac{7}{45}$
$\{0,4,8\}$	$M_{11} = -\frac{5}{9}$
$\{1,3,5\}$	$\det M_{\{0,1\}} = -\frac{5}{36}$
$\{1,4,7\}$	$\det M_{\{0,1\}} = -\frac{1}{4}$
$\{2,3,4\}$	$\det M_{\{0,1\}} = -\frac{5}{36}$
$\{2,4,6\}$	$M_{22} = -\frac{1}{9}$
$\{3,4,5\}$	$\det M_{\{0,1\}} = -\frac{1}{4}$
$\{3,4,8\}$	$M_{11} = -\frac{7}{45}$
$\{3,5,7\}$	$\det M_{\{0,1\}} = -\frac{5}{36}$
$\{4,5,6\}$	$\det M_{\{0,1\}} = -\frac{5}{36}$
$\{4,6,8\}$	$M_{33} = -\frac{3}{5}$
$\{6,7,8\}$	$\det M_{\{0,1\}} = -\frac{4}{25}$

Every listed entry is a necessary principal-minor condition for positive semidefiniteness. Thus no candidate survives the anchored real neutral $L = 3$ screen. □

Principle 9434.11 (Square realness is not RP positivity)**.** *The square-alphabet real-sign construction gives Q3 compatibility and two-point realness, but it does not by itself give reflection positivity. At $N = 9$, the natural real-sign Q3 family is already empty at the anchored $L = 3$ RP level, even though it has many anchored $L = 2$ survivors. The result concerns only the real-sign subfamily*

$$\{+1,-1\}^9 \cap \mathcal{Q}_9.$$

The general complex phase locus

$$\mathcal{Q}_9 = \left\{ v \in \mathbb{T}^9 : \sum_{a=0}^{8} v_a = 3 \right\}$$

is not excluded by this screen.

Chapter 9435

Innovation Input Nine-ary Complex Phase Link Formulas

chapter 9434 excludes the real-sign subfamily of the nine-ary Q3 locus from the anchored $L = 3$ real neutral RP screen. That exclusion is a sublocus statement. The complex phase locus still requires the reflection-positive tests to be read on the full phase torus. This chapter records the first scalar link formulas for that locus: the ordinary distance-one and distance-two correlations, the three anchored $L = 3$ link diagonals, and the principal minor inequalities forced by reflection positivity.

9435.1 Nine-ary Phase Locus

Definition 9435.1 (Nine-ary complex Q3 phase row)**.** *The nine-ary complex Q3 phase locus is*

$$\mathcal{Q}_9 := \left\{ v = (v_0, \dots, v_8) \in \mathbb{T}^9 : \sum_{a=0}^{8} v_a = 3 \right\}.$$

For $v \in \mathcal{Q}_9$, the associated self-substitution phase sequence is determined by

$$u_{9n+a} = v_a u_n, \qquad a = 0, \dots, 8.$$

The equality $3 = \sqrt{9}$ makes this the $N = 9$ instance of the Q3 phase-polygon equation in Definition 9425.15.

Definition 9435.2 (Nine-ary cyclic edge ratios)**.** *For a nine-ary phase row, define*

$$x_a := v_a \overline{v_{a+1}}, \qquad a = 0, \dots, 8,$$

where indices are read modulo 9, so that

$$x_8 = v_8 \overline{v_0}.$$

Then

$$|x_a| = 1$$

for every a, and the cyclic product satisfies

$$x_0 x_1 \cdots x_8 = 1.$$

9435.2 Ordinary Link Correlations

Definition 9435.3 (First and second nine-ary correlations)**.** *For the self-substitution sequence associated to a nine-ary phase row, write*

$$\mu := \Gamma(\delta_0 - \delta_1) = \langle u_n \overline{u_{n+1}} \rangle$$

and

$$\nu := \Gamma(\delta_0 - \delta_2) = \langle u_n \overline{u_{n+2}} \rangle .$$

Theorem 9435.4 (Nine-ary first correlation formula)**.** *For every nine-ary self-substitution phase row,*

$$\mu = \frac{x_0 + x_1 + \cdots + x_7}{9 - x_8}.$$

Proof. Split the average defining $\Gamma(\delta_0 - \delta_1)$ by $n = 9q + a$. For $a = 0, \ldots, 7$, the two sites n and $n+1$ lie in the same nine-ary cell and contribute x_a. For $a = 8$, the second site crosses to the next cell and contributes $x_8\mu$. Hence

$$\mu = \frac{1}{9}\left(x_0 + \cdots + x_7 + x_8\mu\right).$$

Since $|x_8| = 1$, the denominator $9 - x_8$ is nonzero, and the displayed formula follows. □

Theorem 9435.5 (Nine-ary second correlation formula)**.** *For every nine-ary self-substitution phase row,*

$$\nu = \frac{1}{9}\left[\sum_{a=0}^{6} x_a x_{a+1} + (x_7 x_8 + x_8 x_0)\mu\right].$$

Proof. Split $\Gamma(\delta_0 - \delta_2)$ by $n = 9q + a$. For $a = 0, \ldots, 6$, no carry occurs and the coefficient is

$$v_a \overline{v_{a+2}} = x_a x_{a+1}.$$

For $a = 7$, the second site crosses to residue 0 in the next cell, and the branch block is $\delta_0 - \delta_1$ with coefficient $x_7 x_8$. For $a = 8$, the second site crosses to residue 1 in the next cell, and the same branch block has coefficient $x_8 x_0$. Averaging over the nine residues gives the formula. □

9435.3 Anchored Link Diagonals

Definition 9435.6 (Nine-ary forward link diagonals)**.** *Use the forward neutral links*

$$F_{12} := u_{n+1}\overline{u_{n+2}}, \qquad F_{13} := u_{n+1}\overline{u_{n+3}}, \qquad F_{23} := u_{n+2}\overline{u_{n+3}}.$$

By translation invariance,

$$\langle F_{12} \rangle = \mu, \qquad \langle F_{13} \rangle = \nu, \qquad \langle F_{23} \rangle = \mu.$$

With the discrete reflection $\theta(j) = 1 - j$ from Definition 9428.1, define

$$D_{ij} := \langle \theta F_{ij}, F_{ij} \rangle.$$

After common translation of the reflected signed block, the three diagonals are

$$D_{12} = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3),$$

$$D_{13} = \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_5),$$

and

$$D_{23} = \Gamma(\delta_0 - \delta_1 + \delta_4 - \delta_5).$$

Theorem 9435.7 (Nine-ary D_{12} formula)**.** *For every nine-ary self-substitution phase row,*

$$D_{12} = \frac{1}{9}\left[\sum_{a=0}^{5} x_a x_{a+2} + x_7 x_0 + (x_6 x_8 + x_8 x_1)\mu\right].$$

Proof. Apply the residue split to $\delta_0 - \delta_1 + \delta_2 - \delta_3$. The branch data are

residue	coefficient	branch block
$0 \le a \le 5$	$x_a x_{a+2}$	$\varnothing$
6	$x_6 x_8$	$\delta_0 - \delta_1$
7	$x_7 x_0$	$\varnothing$
8	$x_8 x_1$	$\delta_0 - \delta_1$

The empty branch contributes 1, and the branch $\delta_0 - \delta_1$ contributes μ. Averaging over all residues gives the displayed formula. □

Theorem 9435.8 (Nine-ary D_{13} formula)**.** *For every nine-ary self-substitution phase row,*

$$D_{13} = \frac{1}{9}\Big[\sum_{a=0}^{3} x_a x_{a+1} x_{a+3} x_{a+4} + x_6 x_7 x_0 x_1 + (x_4 x_5 x_7 x_8 + x_5 x_6 x_8 x_0 + x_7 x_8 x_1 x_2 + x_8 x_0 x_2 x_3)\,\mu\Big].$$

Proof. Apply the residue split to $\delta_0 - \delta_2 + \delta_3 - \delta_5$. The branch data are

residue	coefficient	branch block
$0 \le a \le 3$	$x_a x_{a+1} x_{a+3} x_{a+4}$	$\varnothing$
4	$x_4 x_5 x_7 x_8$	$\delta_0 - \delta_1$
5	$x_5 x_6 x_8 x_0$	$\delta_0 - \delta_1$
6	$x_6 x_7 x_0 x_1$	$\varnothing$
7	$x_7 x_8 x_1 x_2$	$\delta_0 - \delta_1$
8	$x_8 x_0 x_2 x_3$	$\delta_0 - \delta_1$

Again the empty branch contributes 1, while $\delta_0 - \delta_1$ contributes μ. The nine-residue average is the claimed identity. □

Theorem 9435.9 (Nine-ary D_{23} formula)**.** *For every nine-ary self-substitution phase row,*

$$D_{23} = \frac{1}{9}\left[\sum_{a=0}^{3} x_a x_{a+4} + x_5 x_0 + x_6 x_1 + x_7 x_2 + (x_4 x_8 + x_8 x_3)\mu\right].$$

Proof. Apply the residue split to $\delta_0 - \delta_1 + \delta_4 - \delta_5$. The branch data are

residue	coefficient	branch block
$0 \le a \le 3$	$x_a x_{a+4}$	$\varnothing$
4	$x_4 x_8$	$\delta_0 - \delta_1$
5	$x_5 x_0$	$\varnothing$
6	$x_6 x_1$	$\varnothing$
7	$x_7 x_2$	$\varnothing$
8	$x_8 x_3$	$\delta_0 - \delta_1$

Averaging the empty branches and the two $\delta_0 - \delta_1$ branches gives the displayed formula. □

9435.4 Principal Reflection-Positive Conditions

Theorem 9435.10 (Nine-ary three-link RP necessary condition)**.** *If a nine-ary Q3 phase row passes the anchored $L = 3$ neutral RP window, then*

$$D_{12} \in \mathbb{R}_{\geq 0}, \qquad D_{13} \in \mathbb{R}_{\geq 0}, \qquad D_{23} \in \mathbb{R}_{\geq 0},$$

and

$$D_{12} \geq |\mu|^2, \qquad D_{13} \geq |\nu|^2, \qquad D_{23} \geq |\mu|^2.$$

Proof. For each future link F_{ij}, restrict the RP Gram matrix to the two-dimensional span of 1 and F_{ij}. The corresponding principal matrix has the form

$$\begin{pmatrix} 1 & \langle F_{ij} \rangle \\ \overline{\langle F_{ij} \rangle} & D_{ij} \end{pmatrix}.$$

Hermitian positive semidefiniteness forces the lower-right entry to be a nonnegative real number and forces the determinant inequality. The means of F_{12} and F_{23} are μ, while the mean of F_{13} is ν. This gives the displayed conditions. □

Definition 9435.11 (Nine-ary link deficits)**.** *Define the three scalar link deficits by*

$$\Delta_{12} := D_{12} - |\mu|^2, \qquad \Delta_{13} := D_{13} - |\nu|^2, \qquad \Delta_{23} := D_{23} - |\mu|^2.$$

Corollary 9435.12 (Scalar deficit obstruction)**.** *A nine-ary Q3 phase row fails the anchored $L = 3$ neutral RP window if any of the following conditions holds:*

$$D_{12} \notin \mathbb{R}_{\geq 0}, \qquad D_{13} \notin \mathbb{R}_{\geq 0}, \qquad D_{23} \notin \mathbb{R}_{\geq 0},$$

or

$$\Delta_{12} < 0, \qquad \Delta_{13} < 0, \qquad \Delta_{23} < 0$$

for at least one displayed deficit.

Proof. This is the contrapositive of Definition 9435.10. □

9435.5 Pentary Contrast

Principle 9435.13 (Nine-ary links are multi-edge readouts)**.** *The pentary middle link has the endpoint form*

$$D_{13} = |\mu|^2 + \frac{1 - |\mu|^2}{5} \overline{x_4}$$

from Definition 9432.10. Together with the Q3 exclusion of $|\mu| = 1$, this yields the pentary endpoint gate $x_4 = \pm 1$ in Definition 9432.12.

The nine-ary middle link does not collapse to a single endpoint edge in the same residue computation. Its diagonal contains four interior four-edge products, one carry-cancelling boundary product, and four carry terms multiplied by μ:

$$D_{13} = \frac{1}{9}\Big[\sum_{a=0}^{3} x_a x_{a+1} x_{a+3} x_{a+4} + x_6 x_7 x_0 x_1 \\ + (x_4 x_5 x_7 x_8 + x_5 x_6 x_8 x_0 + x_7 x_8 x_1 x_2 + x_8 x_0 x_2 x_3)\,\mu\Big].$$

Thus the scalar $L = 3$ readout for $N = 9$ is governed by several local phase paths rather than by a single endpoint ratio. The real-sign exclusion of Definition 9434.10 therefore does not exclude the complex phase locus $\mathcal{Q}_9$.

Principle 9435.14 (Cell-scale interpretation)**.** *For $N = 5$, the anchored $L = 3$ middle link reaches the digit boundary strongly enough that the scalar diagonal reduces to an endpoint gate. For $N = 9$, the same window is shorter relative to the digit cell. Several residue branches remain inside the cell, and boundary crossings enter as distributed carry corrections. Reflection positivity at this scale is therefore a coupled condition on interior four-edge paths and carry terms, not an endpoint-only condition.*

Chapter 9436

Innovation Input Square Real-Sign Four-Window Bottleneck

Definition 9426.10 gives Q3-compatible real-sign rows for every square alphabet. The nine-ary screen in chapter 9434 shows that the first such square alphabet has no anchored $L = 3$ real neutral RP row. The larger square alphabets exhibit a sharper phenomenon: real-sign rows can pass the anchored $L = 3$ RP matrix in substantial numbers, but the anchored $L = 4$ real neutral matrix remains empty for the computed square families.

9436.1 Twenty-five-ary Real-Sign Locus

Definition 9436.1 (Twenty-five-ary real-sign Q3 row)**.** *A twenty-five-ary real-sign Q3 row is a word*

$$v = (v_0, \ldots, v_{24}) \in \{+1, -1\}^{25}$$

satisfying

$$\sum_{a=0}^{24} v_a = 5.$$

It determines the real-sign self-substitution sequence

$$u_{25n+a} = v_a u_n.$$

Proposition 9436.2 (Twenty-five-ary real-sign count)**.** *A twenty-five-ary real-sign row is Q3-compatible exactly when it has fifteen positive signs and ten negative signs. Hence the search space has*

$$\binom{25}{10} = 3268760$$

rows.

Proof. Let P and M be the numbers of positive and negative entries. Then

$$P + M = 25, \qquad P - M = 5.$$

Thus $P = 15$ and $M = 10$. Conversely, those two cardinalities give the required row sum. The number of rows is the choice of the ten negative positions. □

9436.2 Four-Window Screen

Definition 9436.3 (Anchored real $L = 4$ neutral window)**.** *The anchored real $L = 4$ neutral window is ordered as*

index	*support*
0	$\varnothing$
1	$(1, 2)$
2	$(1, 3)$
3	$(1, 4)$
4	$(2, 3)$
5	$(2, 4)$
6	$(3, 4)$
7	$(1, 2, 3, 4)$.

The last support represents the four-body observable

$$F_{1234} = u_{n+1}u_{n+2}u_{n+3}u_{n+4}.$$

Theorem 9436.4 (Twenty-five-ary real-sign four-window exclusion)**.** *Among the* 3268760 *twenty-five-ary real-sign Q3 rows, exactly* 2784 *pass the anchored $L = 3$ real neutral RP screen. None passes the anchored $L = 4$ real neutral RP screen:*

$$\#\mathcal{R}^{\mathbb{R},\mathrm{neu}}_{25,3} = 2784, \qquad \#\mathcal{R}^{\mathbb{R},\mathrm{neu}}_{25,4} = 0.$$

Proof. The exact carry evaluator is the same finite parity recursion as Definition 9434.7, with 9 replaced by 25. For the $L = 4$ window, every reflected support has diameter at most 7, so a single carry branch lands in either the empty support or the distance-one support. Thus each matrix entry is an exact rational function of the row signs and the distance-one correlation.

The reproducible enumeration is `papers/bedc/experiments/square_real_sign_rp_screen.py`. It enumerates all $\binom{25}{10}$ rows, solves the exact carry values for the anchored $L = 3$ and $L = 4$ real neutral windows, and checks Hermitian symmetry together with the displayed principal-minor certificates. The script asserts the two counts above. □

Theorem 9436.5 (Twenty-five-ary four-window failure split)**.** *Of the* 2784 *twenty-five-ary real-sign rows passing anchored $L = 3$ real neutral RP, the anchored $L = 4$ failures split as follows:*

$$2612 \quad \textit{fail Hermitian symmetry,}$$

and

$$172 \quad \textit{are Hermitian but fail positive semidefiniteness.}$$

Consequently

$$L = 3 \textit{ real neutral RP} \not\Rightarrow L = 4 \textit{ real neutral Hermitian symmetry,}$$

and therefore also

$$L = 3 \textit{ real neutral RP} \not\Rightarrow L = 4 \textit{ real neutral RP.}$$

Proof. This is the refined classification produced by the exact enumeration in Definition 9436.4. A non-Hermitian matrix cannot define a reflection-positive real quadratic form. On the Hermitian subfamily, the script searches diagonal entries and 2×2 principal minors for the first negative certificate; the certificates in Definition 9436.6 exhaust the 172 remaining failures. □

Proposition 9436.6 (Hermitian positive-semidefinite failure certificates)**.** *For the* 172 *twenty-five-ary rows whose anchored* $L = 4$ *matrix is Hermitian but not positive semidefinite, the first low-order certificate has the following distribution:*

certificate	*count*
$\det(0,3) < 0$	8
$\det(0,5) < 0$	20
$\det(0,7) < 0$	4
$\det(1,3) < 0$	6
$\det(1,5) < 0$	4
$\det(1,6) < 0$	8
$\det(2,3) < 0$	2
$\det(3,4) < 0$	4
$\det(5,7) < 0$	2
$M_{33} < 0$	76
$M_{55} < 0$	32
$M_{77} < 0$	6

Proof. The table is the exact certificate counter emitted by `papers/bedc/experiments/square_real_sign_rp_sc` Each listed entry is a necessary condition for positive semidefiniteness of the Hermitian anchored $L = 4$ matrix. □

9436.3 Concrete Witnesses

Proposition 9436.7 (Strict three-window row with four-body failure)**.** *Let the negative positions be*

$$I = \{9, 11, 12, 14, 16, 18, 20, 21, 22, 23\}.$$

Then the row $v_a = -1$ *exactly on* I *is a twenty-five-ary real-sign Q3 row. Its anchored* $L = 3$ *real neutral RP matrix is*

$$M_3 = \begin{pmatrix} 1 & 0 & 13/25 & 0 \\ 0 & 7/25 & 0 & 3/25 \\ 13/25 & 0 & 9/25 & 0 \\ 0 & 3/25 & 0 & 7/25 \end{pmatrix}.$$

This matrix is positive definite, with least eigenvalue

$$\lambda_{\min}(M_3) = \frac{17 - \sqrt{233}}{25} \approx 0.0694264991.$$

However, in the anchored $L = 4$ *real neutral window,*

$$\langle \theta F_{1234}, F_{1234} \rangle = -\frac{1}{25}.$$

Proof. The row has ten negative signs, hence is Q3-compatible by Definition 9436.2. The displayed matrix entries are the exact carry values recomputed by `papers/bedc/experiments/square_real_sign_rp_scree` The two nontrivial blocks have spectra

$$\left\{ \frac{34 \pm 2\sqrt{233}}{50} \right\}$$

and

$$\left\{\frac{4}{25}, \frac{10}{25}\right\},$$

so the least eigenvalue is $(17-\sqrt{233})/25 > 0$. The same exact carry evaluation gives the four-body diagonal $-1/25$, which is already a negative diagonal certificate for the $L = 4$ matrix. □

Proposition 9436.8 (Near-Hermitian survivor still fails symmetry)**.** *Let the negative positions be*

$$I = \{0, 1, 2, 3, 12, 13, 17, 18, 21, 22\}.$$

This row passes anchored $L = 3$ real neutral RP. In the anchored $L = 4$ matrix,

$$M_{3,5} = -\frac{31}{325}, \qquad M_{5,3} = \frac{41}{325}.$$

Therefore

$$M_{3,5} \neq M_{5,3},$$

with discrepancy

$$M_{5,3} - M_{3,5} = \frac{72}{325}.$$

Proof. The row has ten negative signs, hence is Q3-compatible. The exact enumerator confirms anchored $L = 3$ passage and computes the two displayed $L = 4$ entries. Since Hermitian symmetry already fails, no positive-semidefinite RP interpretation is available for this row at window length 4. □

9436.4 Square-Family Pattern

Theorem 9436.9 (Computed square real-sign bottleneck)**.** *For the computed square real-sign Q3 subfamilies, the anchored real neutral RP counts are*

N	*real-sign Q3 rows*	$L = 3$ *pass*	$L = 4$ *pass*
9	84	0	0
16	8008	7	0
25	3268760	2784	0

Proof. The $N = 9$ row is the exact screen of Definition 9434.10. The $N = 16$ and $N = 25$ rows are recomputed by `papers/bedc/experiments/square_real_sign_rp_screen.py`, using the same real-sign carry recursion and the same anchored neutral window conventions. □

Principle 9436.10 (Local positivity is not stable RP)**.** *The square real-sign construction supplies Q3 compatibility and real two-point correlations, and for $N = 16$ and $N = 25$ it also supplies genuine anchored $L = 3$ reflection-positive examples. The same construction has no computed survivor at anchored $L = 4$ for $N = 9, 16, 25$. Thus the empirical obstruction has moved from the existence of low-window RP to the stability of RP across the next neutral window. The four-body observable F_{1234} is a distinguished part of that obstruction, because it already gives the exact negative diagonal $-1/25$ for the strict twenty-five-ary $L = 3$ witness above.*

Chapter 9437

Innovation Input RP Solenoid Limit Conjecture

chapter 9428 separates Q3 self-reference from finite-window reflection positivity. chapter 9436 then shows the square real-sign phenomenon more sharply: larger alphabets can create genuine low-window RP, while the next displayed neutral window still supplies an obstruction. The OS/Hamiltonian gate of chapter 9443 makes the consequence rigid: a finite-window obstruction blocks the input Hamiltonian source. The solenoid arithmetic row of chapter 9451 therefore supplies the natural ambient object for a stronger conjectural reading: complete RP belongs to a cofinal prime-register tower, not to a single finite substitution alphabet.

9437.1 Finite-Window Horizon

Definition 9437.1 (Anchored neutral RP horizon)**.** *For a finite alphabet size N, define*

$$L_{\max}^{\mathrm{neu}}(N) := \sup\left\{ L \in \mathbb{N} : \exists v \in \mathcal{Q}_N,\ v \in \bigcap_{\ell=1}^{L} \mathcal{R}_{N,\ell,1}^{\mathrm{neu}} \right\}.$$

If the displayed set is empty, set $L_{\max}^{\mathrm{neu}}(N) = 0$. The parameter $R = 1$ fixes the anchored window convention used by the finite-window audit. Requiring all reflection-centered placements can only lower the horizon.

Principle 9437.2 (Horizon reading)**.** *The number $L_{\max}^{\mathrm{neu}}(N)$ measures how long a single alphabet size can postpone a neutral RP obstruction. It is not a complete RP certificate. Complete neutral RP is the stronger assertion*

$$\mathcal{R}_N^{\mathrm{neu}} = \bigcap_{L,R} \mathcal{R}_{N,L,R}^{\mathrm{neu}} \neq \varnothing,$$

as in Definition 9441.14.

Principle 9437.3 (Finite-layer evidence ledger)**.** *The displayed Q3 searches support a monotone-*

looking but obstruction-rich pattern:

N	*locus*	*displayed passage*	*displayed obstruction*
2	*binary row*	*none charged*	$L = 1$ *charged*
3	*unit-root row*	*two-point real*	*neutral determinant*
4	*real-sign square*	$L = 1$ *charged*	*neutral window*
5	$\mathcal{Q}_5$	$L = 2$ *example*	$L = 3$ *empty*
9	*real-sign square*	$L = 2$ *examples*	$L = 3$ *empty*
16	*real-sign square*	$L = 3$ *examples*	$L = 4$ *empty*
25	*real-sign square*	$L = 3$ *strict examples*	$L = 4$ *empty*
36	*reported real-sign square*	$L = 3$ *strict examples*	$L = 4$ *reported failure*
900	*reported CRT square*	$L = 4$ *strict examples*	$L = 5$ *reported failure*
44100	*reported CRT square*	$L = 4$ *strict example*	$L = 5$ *reported failure*
5336100	*reported CRT square*	$H_5 = 0$ *targeted*	$L = 4, L = 5$ *reported failure*

The rows through $N = 25$ are the theorem-level rows cited in Definition 9428.19, Definition 9431.8, Definition 9433.7, Definition 9434.10, and Definition 9436.9. The rows for $N = 36$, $N = 900$, $N = 44100$, and $N = 5336100$ are search-ledger rows: they motivate the cofinal prime-register formulation but do not carry theorem status in this chapter.

Principle 9437.4 (Four-prime CRT witness ledger)**.** *The reported four-prime primorial-square row uses*

$$N = (2 \cdot 3 \cdot 5 \cdot 7)^2 = 44100$$

and the CRT factorization

$$\mathbb{Z}/44100\mathbb{Z} \cong \mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/9\mathbb{Z} \times \mathbb{Z}/25\mathbb{Z} \times \mathbb{Z}/49\mathbb{Z}.$$

It is the product row

$$v(a) = v_4(a \bmod 4)v_9(a \bmod 9)v_{25}(a \bmod 25)v_{49}(a \bmod 49),$$

where the local negative positions are

$$\begin{gathered} I_4 = \{3\}, \\ I_9 = \{1, 5, 6\}, \\ I_{25} = \{0, 1, 2, 3, 4, 5, 6, 9, 19, 23\}, \\ I_{49} = \{1, 2, 5, 6, 9, 10, 16, 17, 20, 21, 24, 27, 30, 31, 34, 35, 37, 38, 39, 41, 43\}. \end{gathered}$$

The local sums are

$$\sum v_4 = 2, \qquad \sum v_9 = 3, \qquad \sum v_{25} = 5, \qquad \sum v_{49} = 7,$$

so the product row satisfies the Q3 sum rule

$$\sum_{a=0}^{44099} v_a = 2 \cdot 3 \cdot 5 \cdot 7 = 210 = \sqrt{44100}.$$

The reported anchored real neutral $L = 4$ matrix is symmetric and positive definite, with

$$\lambda_{\min}(M_4) \approx 7.63 \times 10^{-4}$$

and least diagonal certificate

$$M_{(2,3),(2,3)} = \frac{1}{1225} > 0.$$

The same row fails the anchored real neutral $L = 5$ matrix already at the symmetry layer:

$$M_{(1,3),(1,2,4,5)} = \frac{1}{441}, \qquad M_{(1,2,4,5),(1,3)} = -\frac{11}{2205},$$

with discrepancy

$$\frac{1}{441} - \left(-\frac{11}{2205}\right) = \frac{16}{2205}.$$

It also has the negative diagonal certificate

$$M_{(3,5),(3,5)} = -\frac{1}{441} < 0.$$

Thus the four-prime CRT row strengthens the horizon pattern: primorial structure improves finite-window survival, while a fixed finite register still exhibits a finite-window obstruction.

Principle 9437.5 (Primorial-square homomorphic delay)**.** *For the anchored real neutral $L = 5$ window, define the primitive Hermitian defect*

$$H_5 := M_{(1,3),(1,2,4,5)} - M_{(1,2,4,5),(1,3)}.$$

The reported CRT rows for

$$N_3 = (2 \cdot 3 \cdot 5)^2 = 900$$

and

$$N_4 = (2 \cdot 3 \cdot 5 \cdot 7)^2 = 44100$$

exhibit the same anchored real neutral $L = 5$ Hermitian defect shape. In both rows the first displayed asymmetry occurs between the support $(1, 3)$ and the support $(1, 2, 4, 5)$. For N_3 the reported values are

$$M_{(1,3),(1,2,4,5)} = \frac{1}{45}, \qquad M_{(1,2,4,5),(1,3)} = -\frac{11}{225},$$

so the defect is

$$H_5(N_3) = \frac{1}{45} - \left(-\frac{11}{225}\right) = \frac{16}{225}.$$

For N_4 the corresponding values are

$$M_{(1,3),(1,2,4,5)} = \frac{1}{441}, \qquad M_{(1,2,4,5),(1,3)} = -\frac{11}{2205},$$

so the defect is

$$H_5(N_4) = \frac{1}{441} - \left(-\frac{11}{2205}\right) = \frac{16}{2205}.$$

The defect ratio is therefore

$$\frac{H_5(N_4)}{H_5(N_3)} = \frac{16/2205}{16/225} = \frac{5}{49}.$$

Adding the prime-7 register does not change the visible obstruction type; it attenuates the obstruction. The positive-window horizon has therefore not crossed $L = 5$ in these two reported rows, but the obstruction has the form of a stable finite-window defect whose amplitude decreases under prime-register enlargement.

Principle 9437.6 (Five-prime cancellation tradeoff)**.** *At the five-prime level*

$$N_5 = (2 \cdot 3 \cdot 5 \cdot 7 \cdot 11)^2 = 5336100,$$

the CRT factorization is

$$\mathbb{Z}/N_5\mathbb{Z} \cong \mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/9\mathbb{Z} \times \mathbb{Z}/25\mathbb{Z} \times \mathbb{Z}/49\mathbb{Z} \times \mathbb{Z}/121\mathbb{Z}.$$

The reported product row uses a 121*-component real-sign factor with negative positions*

$$\begin{gathered} I_{121} = \{0, 1, 2, 4, 6, 7, 10, 14, 17, 22, 26, 27, 30, 32, 33, 36, \\ 37, 38, 39, 40, 45, 49, 51, 56, 58, 59, 62, 63, 66, 67, 68, 69, \\ 70, 71, 73, 74, 75, 77, 82, 83, 84, 87, 90, 95, 99, 100, \\ 103, 104, 106, 108, 109, 111, 112, 114, 115\}. \end{gathered}$$

Thus $\#(-) = 55$*,* $\#(+) = 66$*, and*

$$\sum v_{121} = 11.$$

Together with the local sums $2, 3, 5, 7$*, the product row has total sum*

$$2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310 = \sqrt{5336100},$$

so it is Q3-compatible.

For this selected five-prime row, the inherited primitive Hermitian defect is cancelled:

$$H_5(N_5) = 0.$$

Equivalently, the two reflected correlation blocks agree:

$$\Gamma(0, 2, 3, 4, 6, 7) = \Gamma(0, 1, 3, 4, 5, 7) = -\frac{5326399}{25885425951}.$$

This shows that the H_5 *channel is not an immutable finite-level obstruction. It can be killed by a suitable prime-*11 *local factor.*

The same selected row does not pass the window tests. Its anchored real neutral $L = 5$ *matrix has a larger Hermitian defect at the support pair* $(1, 4)$ *and* $(1, 2, 3, 5)$*:*

$$M_{(1,4),(1,2,3,5)} = -\frac{5324243}{31637742829}, \qquad M_{(1,2,3,5),(1,4)} = \frac{69333739}{94913228487},$$

with difference

$$-\frac{85306468}{94913228487} \approx -8.9878 \times 10^{-4}.$$

The anchored real neutral $L = 4$ *test is also lost for this row:*

$$\max |M_4 - M_4^{\mathsf{T}}| \approx 1.7991 \times 10^{-4},$$

and

$$\lambda_{\min}\left(\frac{M_4 + M_4^{\mathsf{T}}}{2}\right) \approx -3.40 \times 10^{-4}.$$

Thus adjoining a prime channel can cancel a chosen high-window defect while creating lower-window incompatibility.

Principle 9437.7 (Local-factor compatibility problem)**.** *The primorial-square search is therefore not a one-defect optimization problem. It asks for a recursive choice of local real-sign factors*

$$v_4, v_9, v_{25}, v_{49}, v_{121}, \dots$$

such that adjoining a prime factor v_{p^2} preserves every previously secured anchored neutral window while reducing the next obstruction. In schematic form, if the S-register row has passed the windows $\ell \le L$, the extension to $S \cup \{p\}$ must keep

$$M_\ell^{(S\cup\{p\})} \succeq 0 \qquad \ell \le L$$

and only then use the extra prime channel to suppress a higher-window defect. Complete RP is a tower-compatibility problem, not an isolated finite-N search.

Principle 9437.8 (Diagonal obstruction persists beside symmetry failure)**.** *The same comparison also shows that the $L = 5$ failure is not merely a choice of non-Hermitian presentation. The reported N_3 row has a negative diagonal certificate*

$$M_{(1,2,3,5),(1,2,3,5)} = -\frac{5393}{202275} < 0,$$

while the reported N_4 row has

$$M_{(3,5),(3,5)} = -\frac{1}{441} < 0.$$

Thus positive semidefiniteness still fails after ignoring the Hermitian asymmetry. The $L = 5$ obstruction is therefore layered: the Hermitian defect blocks reflection symmetry, and negative diagonal entries block positive semidefiniteness even after discarding that asymmetry. The primorial-square evidence supports a two-part reading: finite prime registers can weaken high-window defects, but a fixed finite register still leaves a concrete finite-window obstruction.

9437.2 Finite Alphabet Finiteness

Conjecture 9437.9 (Finite-alphabet RP finiteness)**.** *For every finite alphabet size N, no Q3 self-substitution row over that alphabet passes all neutral finite-window RP tests:*

$$\forall N < \infty, \qquad \mathcal{R}_N^{\mathrm{neu}} = \varnothing.$$

Equivalently, for every fixed finite N and every $v \in \mathcal{Q}_N$, there is a finite neutral RP window $\mathcal{B}$ such that

$$M_{\mathcal{B}}(v) \not\succeq 0.$$

In the anchored-depth language of Definition 9437.1, this predicts

$$L_{\max}^{\mathrm{neu}}(N) < \infty$$

for every finite N.

Status: `open`

Principle 9437.10 (Finite alphabets delay rather than remove RP failure)**.** *Under Definition 9437.9, increasing N can raise the finite-window horizon, especially when the prime factor structure of N gives more independent carry channels. It cannot turn a single finite row into an all-window OS/RP source. The input Hamiltonian gate of Definition 9443.12 therefore treats every finite-window negative certificate as a genuine source-side exclusion, even when the same row has strong lower-window positivity.*

9437.3 Prime-Register Approximation

Definition 9437.11 (Cofinal square prime-register tower)**.** *Let p_k be the k-th prime and set*

$$P_k := \prod_{p \le p_k} p, \qquad N_k := P_k^2.$$

The associated prime sets

$$S_k := \{p : p \le p_k\}$$

increase cofinally to the set $\mathbb{P}$ of all primes. The stationary finite-register solenoid attached to level k is

$$\Sigma_k = \varprojlim \left(\mathbb{T} \xleftarrow{z \mapsto z^{P_k^2}} \mathbb{T} \right),$$

with charge group

$$\widehat{\Sigma_k} = \mathbb{Z}[S_k^{-1}]$$

at finite prime register S_k.

Conjecture 9437.12 (Prime-register RP approximation)**.** *For every finite neutral RP window $\mathcal{B}$, there is a prime level k and a Q3 row*

$$v^{(k)} \in \mathcal{Q}_{P_k^2}$$

such that

$$M_{\mathcal{B}}(v^{(k)}) \succeq 0.$$

Equivalently, for the anchored-depth audit,

$$\forall L \ \exists k \ \exists v^{(k)} \in \mathcal{Q}_{P_k^2}, \qquad v^{(k)} \in \bigcap_{\ell=1}^{L} \mathcal{R}^{\mathrm{neu}}_{P_k^2,\ell,1}.$$

Together with Definition 9437.9, this says that every fixed level remains finite-window vulnerable:

$$\forall k \ \forall v \in \mathcal{Q}_{P_k^2} \ \exists \mathcal{B}, \qquad M_{\mathcal{B}}(v) \not\succeq 0,$$

while every fixed finite window can be made positive at a sufficiently large prime-register level.

Status: `open`

Conjecture 9437.13 (Cofinal-tail RP approximation)**.** *There is a compatible primorial-square CRT family*

$$v^{(k)} \in \mathcal{Q}_{P_k^2}$$

such that every fixed anchored neutral window eventually becomes positive-semidefinite along the tail:

$$\forall L \ \exists k_0(L) \ \forall k \ge k_0(L), \qquad v^{(k)} \in \bigcap_{\ell=1}^{L} \mathcal{R}^{\mathrm{neu}}_{P_k^2,\ell,1}.$$

For every fixed finite k, however, some finite neutral window still obstructs the same row:

$$\forall k \ \exists \mathcal{B}, \qquad M_{\mathcal{B}}(v^{(k)}) \not\succeq 0.$$

This is the tail-strength form of the limit RP conjecture: complete RP is not assigned to any single finite alphabet, but to the cofinal prime-register limit.

Status: `open`

Principle 9437.14 (Solenoid-limit source reading)**.** *The conjectural source object is not a distinguished finite N. It is the cofinal prime-register limit*

$$S_k \nearrow \mathbb{P}, \qquad P_k^2 = \left(\prod_{p \le p_k} p\right)^2,$$

and the corresponding full solenoid

$$\Sigma_{\mathbb{P}} \cong \widehat{\mathbb{Q}}.$$

Its visible circle projection has hidden fibre

$$K_{\mathbb{P}} := \ker(\Sigma_{\mathbb{P}} \to \mathbb{T}) \cong \prod_{p \in \mathbb{P}} \mathbb{Z}_p.$$

This is the arithmetic form of the RP limit intuition: finite alphabets carry only finitely many branch registers, while the all-prime solenoid retains an infinite compatible fibre capable of absorbing arbitrarily deep carry, anchor, and boundary constraints.

9437.4 Limit Matrix Problem

Principle 9437.15 (Fixed-window limit problem)**.** *For each finite neutral window $\mathcal{B}$, the prime-register programme asks for a selected tower of rows*

$$v^{(k)} \in \mathcal{Q}_{P_k^2}$$

for which the entrywise limit

$$M_{\mathcal{B}}^{(\infty)} := \lim_{k \to \infty} M_{\mathcal{B}}(v^{(k)})$$

exists and satisfies

$$M_{\mathcal{B}}^{(\infty)} \succeq 0.$$

The stronger compatibility problem asks that these limits agree under restriction of windows, so that the family of positive finite-window forms defines one OS/RP source form on the solenoid input process.

Principle 9437.16 (Complete RP as a limit property)**.** *The finite-window evidence and the solenoid arithmetic row point to the same classification. Q3 compatibility is a finite refinement law; local RP is a finite-window positivity event; complete RP is a compatible all-window positivity structure. The first two can occur at finite alphabet size. The third is conjecturally a property of the cofinal prime-register limit. Even if the limit source form is obtained, the Hilbert–Polya upgrade remains a separate bridge obligation by Definition 9443.10.*

Chapter 9438

Innovation Input Pentary Golden Hermitian Obstruction

chapter 9432 turns the anchored $L = 3$ link diagonals into explicit pentary formulas. This chapter records a Q3 row that passes those scalar tests but fails the Hermitian symmetry required by the finite-window RP matrix.

9438.1 Golden Symmetric Word

Definition 9438.1 (Golden symmetric pentary word)**.** *Let*

$$c := \frac{\sqrt{5}-3}{2},$$

choose $\theta \in (0, \pi)$ *with* $\cos\theta = c$*, and set*

$$r := e^{i\theta}.$$

Define

$$v_\diamond := (1, \overline{r}, 1, r, 1).$$

Theorem 9438.2 (Golden symmetric row is Q3)**.** *The row* $v_\diamond$ *lies in* $\mathcal{Q}_5$*.*

Proof. Since $r + \overline{r} = 2c$,

$$1 + \overline{r} + 1 + r + 1 = 3 + 2c = 3 + 2\frac{\sqrt{5}-3}{2} = \sqrt{5}.$$

Thus the Q3 equation holds. □

Theorem 9438.3 (Golden symmetric row passes the link-diagonal filter)**.** *For* $v_\diamond$*,*

$$\mu = \frac{\sqrt{5}-3}{2}$$

and

$$\nu = D_{12} = D_{13} = D_{23} = 3 - \frac{6\sqrt{5}}{5}.$$

Numerically,

$$\mu \approx -0.381966$$

and

$$\nu = D_{12} = D_{13} = D_{23} \approx 0.316718.$$

Consequently,

$$D_{12} \geq |\mu|^2, \qquad D_{13} \geq |\nu|^2, \qquad D_{23} \geq |\mu|^2.$$

Proof. The edge ratios of $v_\diamond$ are

$$x_0 = r, \qquad x_1 = \overline{r}, \qquad x_2 = \overline{r}, \qquad x_3 = r, \qquad x_4 = 1.$$

Definition 9432.3 gives

$$\mu = \frac{r + \overline{r} + \overline{r} + r}{5 - 1} = \frac{r + \overline{r}}{2} = c.$$

Definition 9432.4 gives

$$\nu = \frac{1}{5}\left[2 + \overline{r}^2 + 2r\mu\right].$$

Because $2r\mu = r(r + \overline{r}) = r^2 + 1$,

$$\nu = \frac{1}{5}(3 + r^2 + \overline{r}^2) = \frac{1}{5}(3 + 2\cos 2\theta).$$

Using $\cos 2\theta = 2c^2 - 1$ and

$$c^2 = \left(\frac{\sqrt{5} - 3}{2}\right)^2 = \frac{7 - 3\sqrt{5}}{2},$$

one obtains

$$\nu = \frac{1}{5}(1 + 4c^2) = 3 - \frac{6\sqrt{5}}{5}.$$

Substituting the same edge ratios into Definition 9432.6, Definition 9432.7, and Definition 9432.8 gives $D_{12} = D_{13} = D_{23} = \nu$.

Finally,

$$|\mu|^2 = \left(\frac{\sqrt{5} - 3}{2}\right)^2 \approx 0.145898$$

and

$$|\nu|^2 \approx 0.100310.$$

Both are smaller than $3 - 6\sqrt{5}/5 \approx 0.316718$, giving the displayed link inequalities. □

9438.2 Hermitian Obstruction

Definition 9438.4 (Anchored $L = 3$ Hermitian defect)**.** *Let $M_3(v)$ be the anchored $L = 3$ neutral RP matrix associated to the neutral monomials generated by $1, F_{12}, F_{13}, F_{23}$. Define*

$$\mathcal{H}_3(v) := M_3(v) - M_3(v)^*.$$

An anchored $L = 3$ RP candidate must have

$$\mathcal{H}_3(v) = 0$$

before the positive-semidefinite condition can even be tested as a Hermitian matrix condition.

Theorem 9438.5 (Golden symmetric Hermitian obstruction)**.** *For $v_\diamond = (1, \overline{r}, 1, r, 1)$, the two crossed entries*

$$C_{12,13} := \langle \theta F_{12}, F_{13} \rangle$$

and

$$C_{13,12} := \langle \theta F_{13}, F_{12} \rangle$$

satisfy

$$C_{12,13} = C_{13,12} = \frac{3c + 2c^3}{5} + i\frac{2s^3}{5},$$

where $c = \cos\theta = (\sqrt{5} - 3)/2$ and $s = \sin\theta > 0$. Hence

$$C_{12,13} \neq \overline{C_{13,12}},$$

and the anchored $L = 3$ neutral RP matrix is not Hermitian.

Proof. After translation normalization,

$$C_{12,13} = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_4)$$

and

$$C_{13,12} = \Gamma(\delta_0 - \delta_2 + \delta_3 - \delta_4).$$

For $v_\diamond$, the carry recursion for each of these two blocks closes on the zero block and the one-step block. The five residues give the same average:

$$C_{12,13} = C_{13,12} = \frac{1}{5}\left[2r + (2 + \overline{r}^2)\mu\right].$$

Since $\mu = c$ and $r = c + is$,

$$2r + (2 + \overline{r}^2)c = 2(c + is) + c\left(2 + c^2 - s^2 - 2ics\right).$$

Using $c^2 + s^2 = 1$, this becomes

$$3c + 2c^3 + i\,2s^3.$$

Therefore the crossed entry has imaginary part $2s^3/5$, which is nonzero. Since the reverse crossed entry is the same nonreal number, the Hermitian relation $M_{12,13} = \overline{M_{13,12}}$ fails. □

Corollary 9438.6 (Diagonal passage does not imply Hermitian passage)**.** *There exists a pentary Q3 row satisfying all three anchored $L = 3$ link-diagonal necessary inequalities while failing the Hermitian condition for the same window.*

Proof. Use $v_\diamond$. The link-diagonal inequalities are Definition 9438.3, while the Hermitian failure is Definition 9438.5. □

Principle 9438.7 (Pentary $L = 3$ RP gates)**.** *For pentary self-substitution rows, anchored $L = 3$ neutral RP has three distinct finite-window gates:*

link diagonals are real and nonnegative,

$$M_3(v) = M_3(v)^*,$$

and

$$M_3(v) \succeq 0.$$

The Pythagorean row $v_\star$ fails the first gate because $D_{23} = -1/5$. The golden symmetric row $v_\diamond$ passes the first gate but fails the Hermitian gate. Thus diagonal positivity is not the reflection-symmetry content of RP; it is only the first scalar shadow of the finite-window matrix condition.

Principle 9438.8 (Pentary search stratification)**.** *The finite-window stratification visible at $N = 5$ is:*

row	*Q3*	*two-point*	$L = 2$ *RP*	$L = 3$ *link*	$L = 3$ *Hermitian*
$v_\star = (a, b, -a, b, a)$	*yes*	*real*	*pass*	*fail*	*not tested*
$v_5^{(+)}$	*yes*	*real*	*fail*	*not tested*	*not tested*
$v_5^{(-)}$	*yes*	*real*	*fail*	*not tested*	*not tested*
$v_\diamond = (1, \overline{r}, 1, r, 1)$	*yes*	*real*	*diagonal pass*	*pass*	*fail*

The unresolved finite-window problem is therefore not only to maximize the link diagonals, but to solve the Hermitian equations and then test the least eigenvalue inside that Hermitian locus.

Chapter 9439

Innovation Input Pentary Palindromic Hermitian Reduction

chapter 9438 shows that link-diagonal positivity does not force the anchored $L = 3$ RP matrix to be Hermitian. This chapter identifies a structural sufficient condition for Hermitian symmetry: palindromic phase words make the balanced correlation function reflection-even. Inside the pentary Q3 palindromic locus, the finite-window problem is therefore reduced from a non-Hermitian matrix problem to a positive-semidefinite problem on a real one-dimensional curve.

9439.1 Reflection Evenness

Definition 9439.1 (Block reflection)**.** *Use the anchored reflection*

$$\theta(j) = 1 - j.$$

For a signed block

$$\alpha = \sum_j e_j \delta_j,$$

define

$$\theta\alpha := \sum_j e_j \delta_{1-j}.$$

Definition 9439.2 (Balanced neutral block)**.** *A signed block*

$$\alpha = \sum_j e_j \delta_j$$

is balanced in the sense of Definition 9427.2 when

$$Q(\alpha) := \sum_j e_j = 0.$$

Balanced blocks are the correlation blocks of neutral monomials, for example $u_{n+i}\overline{u_{n+j}}$ and their finite products.

Definition 9439.3 (Reflection-even balanced correlations)**.** *A self-substitution correlation function Γ is reflection-even on balanced blocks when*

$$\Gamma(\alpha) = \Gamma(\theta\alpha)$$

for every balanced block α.

Theorem 9439.4 (Reflection evenness implies Hermitian RP matrices)**.** *Let*

$$\mathcal{B} = \{\alpha_0, \ldots, \alpha_m\}$$

be a finite set of balanced future blocks, and set

$$M_{ij} := \Gamma(\alpha_j - \theta\alpha_i).$$

If Γ *is reflection-even on balanced blocks, then*

$$M = M^*.$$

Proof. It is enough to prove $M_{ij} = \overline{M_{ji}}$. By definition,

$$M_{ij} = \Gamma(\alpha_j - \theta\alpha_i)$$

and

$$M_{ji} = \Gamma(\alpha_i - \theta\alpha_j).$$

Complex conjugation of a balanced correlation reverses the signed block, so

$$\overline{M_{ji}} = \Gamma(-\alpha_i + \theta\alpha_j).$$

But

$$\theta(\alpha_j - \theta\alpha_i) = \theta\alpha_j - \alpha_i = -\alpha_i + \theta\alpha_j.$$

Reflection evenness therefore gives

$$\overline{M_{ji}} = \Gamma(\theta(\alpha_j - \theta\alpha_i)) = \Gamma(\alpha_j - \theta\alpha_i) = M_{ij}.$$

□

Corollary 9439.5 (Hermitian audit reduction)**.** *For any self-substitution row whose balanced correlations are reflection-even, every neutral finite-window RP matrix is Hermitian. Within such a class, the finite-window RP audit is the single condition*

$$M \succeq 0.$$

9439.2 Palindromic Reflection

Definition 9439.6 (Palindromic phase word)**.** *A phase word* $v = (v_0, \ldots, v_{N-1}) \in \mathbb{T}^N$ *is palindromic when*

$$v_a = v_{N-1-a} \qquad 0 \le a < N.$$

In the pentary case this is the shape

$$v = (a, b, c, b, a).$$

Theorem 9439.7 (Palindromic rows are reflection-even)**.** *If* v *is palindromic, then its self-substitution correlation function satisfies*

$$\Gamma(\alpha) = \Gamma(\theta\alpha)$$

for every balanced block α*.*

Proof. At finite depth K, write

$$n = a_0 + a_1 N + \cdots + a_{K-1} N^{K-1}$$

and

$$u_n^{(K)} = \prod_{\ell=0}^{K-1} v_{a_\ell}.$$

Let

$$R_K(n) := N^K - 1 - n.$$

If n has N-ary digits $(a_0, \ldots, a_{K-1})$, then $R_K(n)$ has digits $(N-1-a_0, \ldots, N-1-a_{K-1})$. Since v is palindromic,

$$u_{R_K(n)}^{(K)} = u_n^{(K)}.$$

For a balanced block $\alpha = \sum_j e_j \delta_j$, take the finite-depth average over indices for which every shifted term remains in the depth K interval. The omitted boundary set has density tending to zero. In the surviving average, substitute $n' = R_K(n)$. The identity above gives

$$u_{n+j}^{(K)} = u_{R_K(n+j)}^{(K)} = u_{R_K(n)-j}^{(K)} = u_{n'-j}^{(K)}.$$

Passing to the correlation limit yields

$$\Gamma(\alpha) = \Gamma\left(\sum_j e_j \delta_{-j}\right).$$

Because α is balanced, translating the block by one unit does not change the neutral correlation, and hence

$$\Gamma\left(\sum_j e_j \delta_{-j}\right) = \Gamma\left(\sum_j e_j \delta_{1-j}\right) = \Gamma(\theta\alpha).$$

□

Corollary 9439.8 (Palindromic rows have Hermitian neutral RP matrices)**.** *Every palindromic self-substitution row has Hermitian neutral finite-window RP matrices.*

Proof. Combine Definition 9439.7 with Definition 9439.4. □

9439.3 Pentary Palindromic Q3 Curve

Definition 9439.9 (Pentary palindromic Q3 locus)**.** *Define*

$$\mathcal{P}_5 := \left\{(a, b, c, b, a) \in \mathbb{T}^5 : 2a + 2b + c = \sqrt{5}\right\}.$$

Definition 9439.10 (Palindromic edge variables)**.** *For $v = (a, b, c, b, a)$, set*

$$q := a\overline{b}, \qquad r := b\overline{c}.$$

Then $|q| = |r| = 1$, and

$$a = qb, \qquad c = \overline{r}\, b.$$

Theorem 9439.11 (Pentary palindromic Q3 equation)**.** *A palindromic phase word $v = (a, b, c, b, a)$ lies in $\mathcal{P}_5$ if and only if there are $q, r \in \mathbb{T}$ such that*

$$|2q + 2 + \overline{r}| = \sqrt{5},$$

and

$$b = \frac{\sqrt{5}}{2q + 2 + \overline{r}}, \qquad a = qb, \qquad c = \overline{r}\, b.$$

Equivalently, for $q = e^{i\alpha}$ and $r = e^{i\beta}$,

$$2\cos\alpha + \cos\beta + \cos(\alpha + \beta) = -1.$$

Proof. Using Definition 9439.10,

$$2a + 2b + c = b(2q + 2 + \overline{r}).$$

Since $|b| = 1$, the Q3 condition is equivalent to $|2q + 2 + \overline{r}| = \sqrt{5}$, with b then determined by the displayed formula.

Now square the modulus:

$$|2q + 2 + \overline{r}|^2 = 9 + 8\cos\alpha + 4\cos\beta + 4\cos(\alpha + \beta).$$

Setting this equal to 5 and dividing by 4 gives

$$2\cos\alpha + \cos\beta + \cos(\alpha + \beta) = -1.$$

□

Corollary 9439.12 (One-dimensional palindromic locus)**.** *Away from singular points of the displayed real constraint, the pentary palindromic Q3 locus $\mathcal{P}_5$ is a real one-dimensional curve.*

9439.4 Palindromic Link Formulas

Definition 9439.13 (Real palindromic edge coordinates)**.** *For $q = e^{i\alpha}$ and $r = e^{i\beta}$, write*

$$A := \operatorname{Re} q = \cos\alpha,$$

$$B := \operatorname{Re} r = \cos\beta,$$

$$C := \operatorname{Re}(qr) = \cos(\alpha + \beta),$$

and

$$E := \operatorname{Re}(q\overline{r}) = \cos(\alpha - \beta).$$

The Q3 equation on $\mathcal{P}_5$ is

$$2A + B + C = -1.$$

Theorem 9439.14 (Palindromic $L = 3$ link formulas)**.** *For every $v = (a, b, c, b, a) \in \mathcal{P}_5$,*

$$\mu = \frac{A + B}{2},$$

$$\nu = D_{23} = \frac{1 + 2C + A(A + B)}{5},$$

$$D_{12} = \frac{1 + 2E + B(A + B)}{5},$$

and

$$D_{13} = \frac{1 + 4\mu^2}{5}.$$

Proof. The pentary edge ratios from Definition 9432.1 are

$$x_0 = q, \qquad x_1 = r, \qquad x_2 = \overline{r}, \qquad x_3 = \overline{q}, \qquad x_4 = 1.$$

Substituting these values into Definition 9432.3 gives

$$\mu = \frac{q + r + \overline{r} + \overline{q}}{4} = \frac{A + B}{2}.$$

For ν, use Definition 9432.4:

$$\nu = \frac{1}{5}\left[qr + 1 + \overline{qr} + (\overline{q} + q)\mu\right] = \frac{1 + 2C + 2A\mu}{5}.$$

Since $2A\mu = A(A + B)$, the formula for $\nu = D_{23}$ follows from Definition 9432.8.

For D_{12}, use Definition 9432.6:

$$D_{12} = \frac{1}{5}\left[q\overline{r} + r\overline{q} + 1 + (\overline{r} + r)\mu\right] = \frac{1 + 2E + 2B\mu}{5}.$$

Since $2B\mu = B(A + B)$, this is the stated formula. Finally, Definition 9432.7 gives

$$D_{13} = \frac{1}{5}\left[1 + (\overline{q} + \overline{r} + r + q)\mu\right] = \frac{1 + 2(A + B)\mu}{5} = \frac{1 + 4\mu^2}{5}.$$

□

Corollary 9439.15 (Palindromic three-link necessary inequalities)**.** *If $v \in \mathcal{P}_5$ passes the anchored $L = 3$ neutral RP window, then*

$$\frac{1 + 2E + B(A + B)}{5} \geq \left(\frac{A + B}{2}\right)^2,$$

$$\frac{1 + 2C + A(A + B)}{5} \geq \left(\frac{A + B}{2}\right)^2,$$

and

$$\frac{1 + 4\mu^2}{5} \geq \nu^2.$$

Proof. These are the 2×2 principal minor inequalities from Definition 9432.9. For the palindromic locus, μ and ν are real, and the link formulas are those of Definition 9439.14. □

9439.5 Known Rows Inside the Reduction

Example 9439.16 (Pythagorean row)**.** *For $v_\star = (a, b, -a, b, a)$, one has*

$$q = a\overline{b} = i, \qquad r = b\overline{(-a)} = i.$$

Thus

$$A = 0, \qquad B = 0, \qquad C = -1, \qquad E = 1.$$

The formulas above give

$$\mu = 0$$

and

$$\nu = D_{23} = \frac{1 + 2(-1)}{5} = -\frac{1}{5}.$$

Thus the $L = 3$ failure of $v_\star$ is exactly the negative second-link diagonal.

Example 9439.17 (Golden palindrome endpoints)**.** *The two golden palindrome rows*

$$v_5^{(+)} = (\xi, \xi^{-1}, -1, \xi^{-1}, \xi), \qquad \xi = e^{\pi i/5},$$

and

$$v_5^{(-)} = (\zeta, \zeta^{-1}, 1, \zeta^{-1}, \zeta), \qquad \zeta = e^{2\pi i/5},$$

are palindromic Q3 rows. Hence their neutral RP matrices are Hermitian by Definition 9439.8. Their failure is already visible at the anchored $L = 2$ window by Definition 9430.14.

Theorem 9439.18 (Pentary palindromic search reduction)**.** *Within $\mathcal{P}_5$, all neutral finite-window RP matrices are Hermitian. The $L = 3$ link-diagonal layer is controlled by the four real coordinates*

$$A = \operatorname{Re} q, \qquad B = \operatorname{Re} r, \qquad C = \operatorname{Re}(qr), \qquad E = \operatorname{Re}(q\overline{r}),$$

subject to

$$2A + B + C = -1.$$

Therefore the pentary palindromic RP search is a positive-semidefinite matrix problem along the real curve

$$2\cos\alpha + \cos\beta + \cos(\alpha + \beta) = -1.$$

Proof. Hermitian symmetry is Definition 9439.8. The parameterization and curve equation are Definition 9439.11. The link-diagonal formulas are Definition 9439.14. □

Principle 9439.19 (Projection reading)**.** *Each tick of the self-substitution ledger chooses a local phase v_a. Finite windows see only the projected correlations $\Gamma(\alpha)$. Palindromic symmetry forces these projected balanced correlations to be reflection-even, so the observed finite-window matrix has no Hermitian twist:*

$$M = M^*.$$

This symmetry is still weaker than reflection positivity. The OS-compatible condition is the positive-semidefinite gate

$$M \succeq 0.$$

Chapter 9440

Innovation Input Pentary Palindromic RP Exclusion

chapter 9439 reduces the pentary palindromic Q3 search to a Hermitian positive-semidefinite problem on a real curve. This chapter sharpens that reduction: inside the palindromic Q3 locus, the anchored $L = 2$ neutral RP window is rigid. The only rows that pass it are the Pythagorean row and its conjugate, and those rows fail at anchored $L = 3$.

9440.1 Literature Positioning

Principle 9440.1 (Adjacent traditions)**.** *The finite-window Q3/RP search lies near three established bodies of work. Constant-length substitution systems and generalized Thue–Morse sequences provide the closest symbolic-dynamical context: Baake–Grimm type diffraction calculations and generalized Thue–Morse Riesz products explain why recursive correlation calculations are the natural language for self-substitution phase rows. Reflection positivity and Osterwalder–Schrader reconstruction provide the operator-theoretic context: Jaffe-style reflection-positive frameworks, Jorgensen-style spectral formulations, and lattice-spin RP methods all stress that the relevant object is not merely real two-point correlation, but a reflected quadratic form that must be positive. Hankel, Toeplitz, and moment-kernel positivity supply a nearby matrix-positivity toolkit for finite principal windows.*

The present object combines these themes in a more specific finite problem: constant-length complex phase self-substitution, the Q3 phase-sum constraint, anchored neutral windows, and explicit RP matrices. The calculations below treat that finite object directly rather than importing an ambient spectral theorem.

9440.2 Anchored $L = 2$ Matrix

Definition 9440.2 (Palindromic anchored $L = 2$ structure)**.** *For a pentary palindromic Q3 row $v = (a, b, c, b, a)$, take*

$$F := u_{n+1}\overline{u_{n+2}}$$

and

$$\alpha := \delta_1 - \delta_2.$$

The anchored neutral $L = 2$ window is

$$\mathcal{B}_2 = \{0, \alpha, -\alpha\}.$$

Write

$$\mu := \langle F \rangle = \Gamma(\delta_0 - \delta_1),$$
$$D := \langle \theta F, F \rangle = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_3),$$

and

$$C := \langle \theta F, \overline{F} \rangle = \Gamma(\delta_0 - \delta_1 - \delta_2 + \delta_3).$$

By Definition 9439.8, the matrix is Hermitian, so it has the form

$$M_2(v) = \begin{pmatrix} 1 & \mu & \mu \\ \mu & D & C \\ \mu & \overline{C} & D \end{pmatrix}.$$

Theorem 9440.3 (Palindromic $L = 2$ semidefinite tests)**.** *If $M_2(v) \succeq 0$, then*

$$D \geq |C|$$

and

$$\det M_2(v) \geq 0.$$

Proof. The lower-right principal block is

$$\begin{pmatrix} D & C \\ \overline{C} & D \end{pmatrix},$$

whose eigenvalues are $D + |C|$ and $D - |C|$. Positivity of this principal block gives $D \geq |C|$. Positivity of the full Hermitian matrix gives $\det M_2(v) \geq 0$. □

9440.3 Half-Angle Curve

Definition 9440.4 (Half-angle variables)**.** *For the edge variables*

$$q = e^{i\alpha}, \qquad r = e^{i\beta},$$

set

$$u := \tan \frac{\alpha}{2}, \qquad w := \tan \frac{\beta}{2}.$$

Then

$$q = \frac{1 + iu}{1 - iu}, \qquad r = \frac{1 + iw}{1 - iw}.$$

Lemma 9440.5 (Half-angle Q3 equation)**.** *The pentary palindromic Q3 equation is equivalent to*

$$-u^2w^2 - u^2 - 4uw + w^2 + 5 = 0.$$

Equivalently,

$$(1 - u^2)w^2 - 4uw + (5 - u^2) = 0.$$

Proof. Use

$$\cos \alpha = \frac{1 - u^2}{1 + u^2}, \qquad \cos \beta = \frac{1 - w^2}{1 + w^2},$$

and

$$\cos(\alpha + \beta) = \frac{(1 - u^2)(1 - w^2) - 4uw}{(1 + u^2)(1 + w^2)}.$$

Substitution into the curve equation

$$2 \cos \alpha + \cos \beta + \cos(\alpha + \beta) = -1$$

from Definition 9439.11 gives the displayed polynomial equation after clearing denominators. □

9440.4 Determinant Factorization

Definition 9440.6 (Palindromic determinant numerator)**.** *Define*

$$\begin{aligned} P(u,w) := {} & u^4w^2 - 2u^3w^3 - 2u^3w + 5u^2w^4 + 12u^2w^2 + 5u^2 \\ & - 2uw^3 - 2uw + w^2. \end{aligned}$$

Theorem 9440.7 (Anchored $L = 2$ determinant factorization)**.** *On the pentary palindromic Q3 locus,*

$$\det M_2(v) = -\frac{16(u-w)^2(u+w)^2P(u,w)}{25(1+u^2)^4(1+w^2)^4}.$$

Moreover $P(u,w) > 0$ on the real Q3 curve. Hence

$$\det M_2(v) \le 0.$$

Proof. Substitute

$$q = \frac{1+iu}{1-iu}, \qquad r = \frac{1+iw}{1-iw}$$

into the anchored $L = 2$ quantities of Definition 9440.2. The determinant of the Hermitian matrix is

$$\det M_2(v) = D^2 - |C|^2 - 2\mu^2(D - \operatorname{Re} C).$$

Direct simplification gives the displayed factorization.

It remains to record why the last factor has no sign change on the real Q3 curve. Eliminating w between $P(u,w) = 0$ and Definition 9440.5 gives, with $z = u^2$,

$$z^6 + 6z^5 + 119z^4 + 84z^3 + 5615z^2 - 730z + 25.$$

Its Sturm chain has no nonnegative real root. Since $P(1,1) = 16 > 0$, the factor P is positive on the real Q3 curve. The denominator is positive, so the sign of the determinant is controlled by the leading minus sign and the two square factors. □

9440.5 Anchored $L = 2$ Rigidity

Theorem 9440.8 (Palindromic $L = 2$ rigidity)**.** *For $v \in \mathcal{P}_5$, the matrix $M_2(v)$ is positive semidefinite if and only if*

$$q = r = i$$

or

$$q = r = -i.$$

Equivalently, the only pentary palindromic Q3 rows passing anchored $L = 2$ neutral RP are the Pythagorean row

$$v_\star = (a, b, -a, b, a), \qquad a = \frac{1+2i}{\sqrt{5}}, \qquad b = \frac{2-i}{\sqrt{5}},$$

and its complex conjugate.

Proof. If $M_2(v) \succeq 0$, then Definition 9440.3 gives $\det M_2(v) \geq 0$. By Definition 9440.7, the determinant is nonpositive, and $P(u, w) > 0$ on the real Q3 curve. Therefore

$$(u - w)^2(u + w)^2 = 0,$$

so either $u = w$ or $u = -w$.

If $u = w$, the half-angle Q3 equation becomes

$$-u^4 - 4u^2 + 5 = 0.$$

With $z = u^2$, this is

$$z^2 + 4z - 5 = 0,$$

so the real solutions have $u^2 = 1$. Hence $u = w = 1$ or $u = w = -1$, giving $q = r = i$ or $q = r = -i$.

If $u = -w$, the half-angle Q3 equation becomes

$$-u^4 + 4u^2 + 5 = 0.$$

With $z = u^2$, this is

$$z^2 - 4z - 5 = 0,$$

so the real branch has $u^2 = 5$. On this branch, direct substitution gives

$$D = \frac{1}{3}, \qquad |C| = \frac{5}{9}.$$

Thus $D - |C| = -2/9 < 0$, contradicting the lower-right principal minor condition. The branch $u = -w$ is excluded.

For $q = r = i$, the Q3 reconstruction in Definition 9439.11 gives the displayed Pythagorean row. The case $q = r = -i$ is its complex conjugate. Conversely, Definition 9431.8 shows that the Pythagorean row passes anchored $L = 2$ neutral RP, and conjugation preserves the same real positive-semidefinite matrix. □

9440.6 Anchored $L = 3$ Exclusion

Theorem 9440.9 (No pentary palindromic anchored $L = 3$ candidate)**.** *There is no row*

$$v \in \mathcal{P}_5$$

that passes anchored $L = 3$ neutral RP.

Proof. If $v \in \mathcal{P}_5$ passed anchored $L = 3$ neutral RP, it would pass the anchored $L = 2$ subwindow. By Definition 9440.8, v would be the Pythagorean row or its complex conjugate. But Definition 9431.10 gives the negative anchored $L = 3$ diagonal value

$$D_{23} = -\frac{1}{5}.$$

The value is real, so the conjugate row gives the same obstruction. Hence no row in $\mathcal{P}_5$ passes anchored $L = 3$ neutral RP. □

Corollary 9440.10 (Palindromic exclusion)**.** *For the anchored neutral $L = 3$ finite-window class,*

$$\mathcal{P}_5 \cap \mathcal{R}^{\text{neu}}_{5,3,1} = \varnothing.$$

Proof. This is exactly Definition 9440.9. □

Principle 9440.11 (Symmetry window)**.** *Palindromic symmetry solves the Hermitian part of the finite-window RP problem, but it also makes the anchored $L = 2$ window rigid. The only surviving rows are the Pythagorean row and its conjugate, and both fail at anchored $L = 3$. Thus a pentary RP candidate, if one exists, cannot be found by imposing full palindromy alone; it must satisfy the Hermitian equations by a finer non-palindromic symmetry.*

Chapter 9441

Innovation Input Self-Similar RP Scaling

chapter 9440 excludes the palindromic pentary Q3 locus from anchored $L = 3$ neutral RP. This chapter records the scale principle behind such finite-window obstructions. The ledger machine is not primarily a state table; it is a self-similar refinement rule. A displayed state is a section of one layer, while the same local refinement rule is the invariant object repeated across all scales.

9441.1 Self-Substitution Rule

Definition 9441.1 (Constant-length phase self-substitution)**.** *Let $N \geq 2$ and let*

$$v = (v_0, \ldots, v_{N-1}) \in \mathbb{T}^N.$$

A constant-length phase self-substitution row is a phase sequence (u_n) satisfying

$$u_{Nn+a} = v_a u_n, \qquad a = 0, \ldots, N-1.$$

If additionally

$$\sum_{a=0}^{N-1} v_a = \sqrt{N},$$

then the row is Q3-compatible, equivalently $v \in \mathcal{Q}_N$.

Principle 9441.2 (Rule before section)**.** *The invariant object of a self-substitution ledger is the local refinement rule*

$$u_{Nn+a} = v_a u_n.$$

A finite state row is only a section through one refinement layer. The cross-layer datum is the repeated rule, not a completed external table of states.

9441.2 Scale Dilation

Definition 9441.3 (Block dilation)**.** *For an integer $M \geq 1$, define the dilation of basis blocks by*

$$D_M \delta_j := \delta_{Mj}.$$

For a signed block $\alpha = \sum_j e_j \delta_j$ in the sense of Definition 9427.1, set

$$D_M \alpha := \sum_j e_j \delta_{Mj}.$$

Theorem 9441.4 (Balanced correlation scale invariance)**.** *Let α be a balanced block in the sense of Definition 9427.2. For every $k \geq 0$,*

$$\Gamma(D_{N^k}\alpha) = \Gamma(\alpha).$$

In particular,

$$\Gamma(D_N \alpha) = \Gamma(\alpha).$$

Proof. It suffices to prove the case $k = 1$. Write

$$\alpha = \sum_j e_j \delta_{r_j}.$$

Then

$$\Gamma(D_N \alpha) = \left\langle \prod_j u_{n+Nr_j}^{e_j} \right\rangle_n .$$

Split $n = Nq + a$, where $a = 0, \ldots, N-1$. Then

$$n + Nr_j = N(q + r_j) + a,$$

and the self-substitution rule gives

$$u_{N(q+r_j)+a} = v_a u_{q+r_j}.$$

Hence

$$\prod_j u_{n+Nr_j}^{e_j} = \prod_j (v_a u_{q+r_j})^{e_j} = v_a^{\sum_j e_j} \prod_j u_{q+r_j}^{e_j}.$$

Since α is balanced, $\sum_j e_j = 0$, so the residue factor is $v_a^0 = 1$. Every residue class gives the same average as $\Gamma(\alpha)$, and averaging over a does not change the result. Iteration gives the formula for D_{N^k}. □

Corollary 9441.5 (Scale copying of finite correlation defects)**.** *If a balanced block α has*

$$\operatorname{Im} \Gamma(\alpha) \neq 0,$$

then every dilated block $D_{N^k}\alpha$ has the same nonzero imaginary part. If

$$\Gamma(\alpha) < 0,$$

then every $D_{N^k}\alpha$ has the same negative value.

Proof. This is immediate from Definition 9441.4. □

Principle 9441.6 (Smooth projection)**.** *Microscopic refinement repeatedly introduces local digit choices. A balanced finite observation does not see a single choice in isolation; it sees the fixed point of the repeated self-substitution rule:*

$$\Gamma(\alpha) = \Gamma(D_N \alpha) = \Gamma(D_{N^2}\alpha) = \cdots .$$

The apparent smoothness of a macroscopic correlation read is therefore a projection of infinitely repeated local choices onto a scale-invariant balanced observable.

9441.3 Reflection-Centered Dilation

Definition 9441.7 (Reflection-centered dilation)**.** *Assume N is odd. Define*

$$\mathcal{D}_N^\theta(\delta_j) := \delta_{\frac{1+N(2j-1)}{2}}.$$

Equivalently, in the half-integer coordinate $x = j - \frac{1}{2}$, the map is

$$x \mapsto Nx.$$

Because N is odd, $\frac{1+N(2j-1)}{2}$ is an integer.

Lemma 9441.8 (Reflection-centered dilation commutes with reflection)**.** *For odd N,*

$$\theta\mathcal{D}_N^\theta = \mathcal{D}_N^\theta\theta.$$

Proof. It is enough to compute on a basis index. First dilate and then reflect:

$$\theta\mathcal{D}_N^\theta(j) = 1 - \frac{1+N(2j-1)}{2} = \frac{1-N(2j-1)}{2}.$$

Reflect first and then dilate:

$$\mathcal{D}_N^\theta(1-j) = \frac{1+N(2(1-j)-1)}{2} = \frac{1-N(2j-1)}{2}.$$

The two expressions agree. □

Lemma 9441.9 (Reflection-centered dilation is ordinary dilation up to translation)**.** *For odd N,*

$$\mathcal{D}_N^\theta(\delta_j) = \delta_{Nj-\frac{N-1}{2}}.$$

Thus on balanced blocks, $\mathcal{D}_N^\theta$ has the same correlation read as D_N.

Proof. The displayed equality is the simplification

$$\frac{1+N(2j-1)}{2} = Nj - \frac{N-1}{2}.$$

The shift by $-(N-1)/2$ is common to every site in the block. Translation does not change balanced finite-block correlations by Definition 9427.4; hence the correlation read agrees with that of D_N. □

9441.4 RP Window Scaling

Definition 9441.10 (Finite neutral RP window)**.** *Let*

$$\mathcal{B} = \{\alpha_0, \ldots, \alpha_m\}$$

be a finite family of balanced future blocks. Its neutral RP matrix is

$$M_{\mathcal{B}}(v)_{ij} := \Gamma(\alpha_j - \theta\alpha_i).$$

Definition 9441.11 (Reflection-centered window dilation)**.** *For odd N, define*

$$\mathcal{D}_N^\theta \mathcal{B} := \{\mathcal{D}_N^\theta \alpha_0, \dots, \mathcal{D}_N^\theta \alpha_m\}.$$

Theorem 9441.12 (Odd-length RP window self-similarity)**.** *Assume N is odd. For every finite neutral RP window $\mathcal{B}$,*

$$M_{\mathcal{D}_N^\theta \mathcal{B}}(v) = M_{\mathcal{B}}(v).$$

More generally,

$$M_{(\mathcal{D}_N^\theta)^k \mathcal{B}}(v) = M_{\mathcal{B}}(v) \qquad k \geq 0.$$

Proof. The (i, j) entry of the dilated-window matrix is

$$\Gamma(\mathcal{D}_N^\theta \alpha_j - \theta \mathcal{D}_N^\theta \alpha_i).$$

By Definition 9441.8,

$$\theta \mathcal{D}_N^\theta \alpha_i = \mathcal{D}_N^\theta \theta \alpha_i.$$

Thus the entry is

$$\Gamma(\mathcal{D}_N^\theta (\alpha_j - \theta \alpha_i)).$$

The block $\alpha_j - \theta \alpha_i$ is balanced because both α_i and α_j are balanced. By Definition 9441.9 and Definition 9441.4,

$$\Gamma(\mathcal{D}_N^\theta (\alpha_j - \theta \alpha_i)) = \Gamma(\alpha_j - \theta \alpha_i).$$

This is exactly the (i, j) entry of $M_{\mathcal{B}}(v)$. Iteration gives the statement for $(\mathcal{D}_N^\theta)^k$. □

Corollary 9441.13 (Scale copying of RP failure)**.** *For odd N, if $M_{\mathcal{B}}(v)$ has a negative eigenvalue, then every dilated matrix*

$$M_{(\mathcal{D}_N^\theta)^k \mathcal{B}}(v)$$

has the same negative eigenvalue.

Proof. The matrices are equal by Definition 9441.12, so their spectra are equal. □

9441.5 Primitive RP Shapes

Definition 9441.14 (Neutral RP feasibility class)**.** *Let $\mathcal{R}_{N,L,R}^{\text{neu}}$ denote the set of Q3 rows $v \in \mathcal{Q}_N$ whose anchored neutral windows of size parameters (L, R) have positive-semidefinite RP matrices. Define the full neutral finite-window class by*

$$\mathcal{R}_N^{\text{neu}} := \bigcap_{L,R} \mathcal{R}_{N,L,R}^{\text{neu}}.$$

Definition 9441.15 (Reflection-centered primitive window)**.** *For odd N, a finite neutral RP window $\mathcal{B}$ is N-primitive when there is no finite neutral RP window $\mathcal{C}$ and no $k \geq 1$ such that*

$$\mathcal{B} = (\mathcal{D}_N^\theta)^k \mathcal{C}.$$

Corollary 9441.16 (Primitive-window reduction)**.** *For odd N, nonprimitive windows do not add new RP matrix constraints. Each such window has the same RP matrix as its primitive root.*

Proof. Apply Definition 9441.12 to the primitive root and iterate until the displayed nonprimitive window is reached. □

Principle 9441.17 (Pentary primitive search)**.** *For $N = 5$, reflection-centered dilation sends*

$$1 \mapsto 3, \qquad 2 \mapsto 8, \qquad 3 \mapsto 13.$$

Thus windows supported on $\{1, 2, 3\}$ and on $\{3, 8, 13\}$ belong to the same RP scale class. A negative finite-window value at one representative is not a small-scale accident; it is a fixed obstruction on the whole reflection-centered dilation tower. Subsequent pentary searches should therefore enumerate primitive RP shapes modulo $\mathcal{D}_5^\theta$ before testing Hermitian symmetry and positive semidefiniteness.

Chapter 9442

Innovation Input Primitive RP Roots

chapter 9441 shows that, for odd substitution length, reflection-centered dilations do not change the neutral RP matrix. This chapter turns that scale invariance into a finite pruning rule. Every finite block and every finite window has a unique primitive root under reflection-centered dilation. Nonprimitive windows therefore carry no additional RP condition.

9442.1 Scale Root

Definition 9442.1 (Single-site scale order)**.** *Let N be odd. For an integer site j, define*

$$\nu_N(j) := \max\left\{k \geq 0 : j - \frac{N^k+1}{2} \in N^k\mathbb{Z}\right\}.$$

Equivalently, $\nu_N(j)$ is the largest k for which j can be written as

$$j = (\mathcal{D}_N^\theta)^k(i)$$

with $i \in \mathbb{Z}$.

Definition 9442.2 (Block scale order and root)**.** *For a nonzero signed block*

$$\alpha = \sum_j e_j\delta_j,$$

define

$$\nu_N(\alpha) := \min_{j:e_j\neq 0} \nu_N(j).$$

The block is N-primitive when $\nu_N(\alpha) = 0$. If $\nu_N(\alpha) = k > 0$, define its scale root by

$$\mathrm{root}_N(\alpha) := (\mathcal{D}_N^\theta)^{-k}\alpha.$$

On sites this inverse is

$$(\mathcal{D}_N^\theta)^{-k}(j) = \frac{j + \frac{N^k-1}{2}}{N^k}.$$

Theorem 9442.3 (Unique primitive block root)**.** *Let N be odd. Every nonzero signed block α has a unique decomposition*

$$\alpha = (\mathcal{D}_N^\theta)^k\alpha_0,$$

where

$$k = \nu_N(\alpha)$$

and α_0 is N-primitive.

Proof. By the definition of $k = \nu_N(\alpha)$, every support point of α can be inverse-dilated k times through $(\mathcal{D}_N^\theta)^{-k}$. This produces a signed block

$$\alpha_0 := (\mathcal{D}_N^\theta)^{-k}\alpha$$

with integer support. Since k is the minimum of the site scale orders over the support, at least one support point has scale order exactly k. Its inverse image has scale order 0, so α_0 is primitive.

The value k is determined by $\nu_N(\alpha)$, and $(\mathcal{D}_N^\theta)^k$ is injective on the integer lattice. Hence the primitive root is unique. □

9442.2 Window Roots

Definition 9442.4 (Future neutral window root)**.** *Let*

$$\mathcal{B} = \{\alpha_0, \ldots, \alpha_m\}$$

be a finite future neutral RP window, with each nonzero α_i balanced. Define

$$\nu_N(\mathcal{B}) := \min_{\alpha_i \neq 0} \nu_N(\alpha_i).$$

The window is N-primitive when $\nu_N(\mathcal{B}) = 0$. If $\nu_N(\mathcal{B}) = k > 0$, define

$$\mathrm{root}_N(\mathcal{B}) := \{(\mathcal{D}_N^\theta)^{-k}\alpha_i\}_i.$$

Theorem 9442.5 (RP matrix depends only on the primitive root)**.** *Let N be odd and let $\mathcal{B}$ be a finite neutral RP window. Then*

$$M_{\mathcal{B}} = M_{\mathrm{root}_N(\mathcal{B})}.$$

Proof. By the definition of the window root, there is a finite neutral window $\mathcal{C} = \mathrm{root}_N(\mathcal{B})$ and an integer $k \geq 0$ with

$$\mathcal{B} = (\mathcal{D}_N^\theta)^k \mathcal{C}.$$

The equality of RP matrices is exactly Definition 9441.12. □

Corollary 9442.6 (Primitive-window pruning)**.** *For odd N, it is enough to test primitive neutral RP windows. If a nonprimitive window passes or fails, its primitive root has the identical RP matrix and the same spectrum.*

Proof. This follows from Definition 9442.5. □

9442.3 Pentary Scale Test

Definition 9442.7 (Pentary reflection-centered address)**.** *For a future site $j \geq 1$, define its pentary reflection-centered address by*

$$\rho_5(j) := 2j - 1.$$

The reflection-centered dilation $j \mapsto 5j - 2$ sends

$$\rho_5(j) \mapsto 5\rho_5(j).$$

Theorem 9442.8 (Pentary site scale order)**.** *For $j \geq 1$,*

$$\nu_5(j) = v_5(2j-1),$$

where v_5 is the ordinary 5-adic valuation.

Proof. The equality

$$j = (\mathcal{D}_5^\theta)^k(i)$$

is equivalent to

$$j - \frac{1}{2} = 5^k\left(i - \frac{1}{2}\right).$$

Multiplying by 2 gives

$$2j - 1 = 5^k(2i-1).$$

Since $2i-1$ is an odd integer, the largest possible k is exactly $v_5(2j-1)$. □

Corollary 9442.9 (Pentary block scale order)**.** *For a nonzero signed block $\alpha = \sum_j e_j\delta_j$,*

$$\nu_5(\alpha) = \min_{j:e_j\neq 0} v_5(2j-1).$$

The block is 5-primitive if and only if at least one support point satisfies

$$5 \nmid (2j-1).$$

Corollary 9442.10 (Pentary window root formula)**.** *Let*

$$k = \min_{j\in\operatorname{supp}\mathcal{B}} v_5(2j-1).$$

Then the root of the window is obtained sitewise by

$$j \mapsto \frac{1 + \frac{2j-1}{5^k}}{2}.$$

Proof. Apply the inverse formula in Definition 9442.2 and simplify using $2j-1$. □

9442.4 Small Pentary Windows

Proposition 9442.11 (Small pentary windows are primitive)**.** *For $N = 5$, every nonzero neutral future window whose support is contained in*

$$\{1, 2, 3, 4, 5, 6, 7\}$$

is 5-primitive.

Proof. A support point can be inverse-dilated once precisely when j is congruent to 3 modulo 5. Inside $\{1, 2, 3, 4, 5, 6, 7\}$, the only such site is 3. If a nonzero balanced block were supported only at 3, it would have the form $e\delta_3$, and balancedness would force $e = 0$. Hence every nonzero neutral block in such a window has at least one primitive support point, and the window is primitive. □

Corollary 9442.12 (Early pentary failures are primitive)**.** *The pentary anchored $L = 2$ and $L = 3$ windows studied in the preceding chapters are primitive windows. In particular, the Pythagorean obstruction*

$$D_{23} = -\frac{1}{5}$$

is a primitive obstruction, not a dilation copy of a smaller RP failure.

Proof. Their supports lie in $\{1, 2, 3\}$, so Definition 9442.11 applies. The value $D_{23} = -1/5$ is Definition 9431.10. □

9442.5 Pentary Link Roots

Definition 9442.13 (Link block)**.** *For $1 \le i < j$, define*

$$\alpha_{ij} := \delta_i - \delta_j.$$

It corresponds to the link observable

$$F_{ij} = u_{n+i}\overline{u_{n+j}}.$$

Proposition 9442.14 (Pentary link primitive criterion)**.** *The link block α_{ij} is nonprimitive if and only if i and j are both congruent to 3 modulo 5. When*

$$i = 5i' - 2, \qquad j = 5j' - 2,$$

one has

$$\alpha_{ij} = \mathcal{D}_5^\theta(\alpha_{i'j'}).$$

Proof. The support of α_{ij} is $\{i, j\}$. It can be inverse-dilated once exactly when both support points can be inverse-dilated once, which is the congruence condition that i and j are both congruent to 3 modulo 5. In that case

$$i' = \frac{i+2}{5}, \qquad j' = \frac{j+2}{5}$$

are integers, and the displayed identity follows from $\mathcal{D}_5^\theta(t) = 5t - 2$. □

Example 9442.15 (First nonprimitive pentary links)**.** *The first nonprimitive pentary links are*

$$\alpha_{3,8} = \mathcal{D}_5^\theta(\alpha_{1,2}),$$

$$\alpha_{3,13} = \mathcal{D}_5^\theta(\alpha_{1,3}),$$

and

$$\alpha_{8,13} = \mathcal{D}_5^\theta(\alpha_{2,3}).$$

Therefore the corresponding two-link neutral windows have exactly the same RP matrices as their roots.

Principle 9442.16 (Reflection-centered address rather than distance)**.** *Ordinary link distance does not determine an RP diagonal. The relevant scale datum is the reflection-centered address $2j - 1$ of every support point. Thus F_{12} and F_{23} are both distance-one links, but they occupy different positions relative to the reflection center. This is why the Pythagorean row can have*

$$D_{12} = \frac{3}{5}$$

and

$$D_{23} = -\frac{1}{5}.$$

9442.6 Odd-Length Address Formula

Definition 9442.17 (Reflection-centered N-address)**.** *For odd N and $j \geq 1$, set*

$$\rho_N(j) := 2j - 1.$$

Theorem 9442.18 (Odd-length scale order)**.** *For odd N,*

$$\nu_N(j) = \max\{k \geq 0 : N^k \mid (2j-1)\}.$$

If

$$N = \prod_{p \mid N} p^{e_p},$$

then

$$\nu_N(j) = \min_{p \mid N} \left\lfloor \frac{v_p(2j-1)}{e_p} \right\rfloor.$$

Proof. The equality $j = (\mathcal{D}_N^\theta)^k(i)$ is equivalent to

$$2j - 1 = N^k(2i - 1).$$

Thus N^k must divide $2j-1$, and the largest such k is the first displayed formula. The prime-factor expression follows because $N^k \mid m$ holds exactly when $ke_p \leq v_p(m)$ for every prime $p \mid N$. □

Principle 9442.19 (Solenoid address reading)**.** *The primitive RP shape test is an S-adic address test. For a refinement length N with prime set S, the visible site j carries the reflection-centered address $2j-1$; the number of times a window is a scale copy is the number of times those addresses are divisible by N. Thus finite RP constraints are indexed by primitive residues in the hidden S-adic fibre, while nonprimitive windows are repetitions along the same solenoidal scale direction.*

Chapter 9443

Innovation Input Self-Adjoint Hamiltonian Gate

chapter 9442 reduces odd-length neutral RP checks to primitive reflection-centered windows. The present chapter records what those checks are allowed to mean. They are input-side tests for the existence of an Osterwalder–Schrader Hamiltonian. They are not, by themselves, a construction of a Riemann-zero spectral operator.

9443.1 Three Operator Rows

Definition 9443.1 (Hilbert–Polya zero target)**.** *A Hilbert–Polya zero target for the completed Riemann function is a self-adjoint operator A_ξ^{HP} whose spectral row satisfies an identification of the form*

$$\mathrm{Spec}(A_\xi^{\mathrm{HP}}) = \{\gamma : \xi(1/2 + i\gamma) = 0\}.$$

Such a row is an RH-strength target: if the operator is genuinely self-adjoint and the displayed identification is complete, the spectral values on the right are real ordinates on the critical line.

Definition 9443.2 (Input-side OS/RP Hamiltonian)**.** *Let $v \in \mathcal{Q}_N$ be a Q3-compatible constant-length self-substitution row, and suppose its complete future reflection form is realized by a positive contraction transition row P_v on the OS Hilbert space of the input process. The associated input Hamiltonian is*

$$H_v := -\frac{1}{\ln N} \log P_v$$

on the nonzero spectral subspace of P_v.

Definition 9443.3 (Riemann scattering row)**.** *At the modular cusp, the Eisenstein constant term has scattering coefficient*

$$\varphi(s) = \frac{\Lambda(2s-1)}{\Lambda(2s)}.$$

Thus the completed zeta function enters the modular input interface as a scattering coefficient. Its zeros are reached there as scattering poles or resonances, as in Definition 9452.29, not as an already identified self-adjoint spectrum.

Principle 9443.4 (Operator rows are not interchangeable)**.** *The rows*

$$A_\xi^{\mathrm{HP}}, \qquad H_v = -\frac{1}{\ln N} \log P_v, \qquad \varphi(s) = \frac{\Lambda(2s-1)}{\Lambda(2s)}$$

have different meanings. The first is the desired zero-spectrum row. The second is the input Hamiltonian produced by reflection positivity. The third is the scattering entry through which Riemann zeta reaches the modular boundary. A proof route may compare these rows only after it exhibits explicit bridge data between the corresponding interfaces.

9443.2 RP as the Input Source of Self-Adjointness

Theorem 9443.5 (Positive transition row gives an input Hamiltonian)**.** *Assume $0 \preceq P_v \preceq I$ on the OS Hilbert space associated to a Q3-compatible self-substitution input. Then*

$$H_v := -\frac{1}{\ln N} \log P_v$$

is self-adjoint and nonnegative on the nonzero spectral subspace, and

$$e^{-(\ln N)H_v} = P_v.$$

Proof. The operator P_v is a positive contraction, hence its spectrum lies in $[0, 1]$. On the nonzero spectral subspace, the scalar function

$$x \mapsto -\frac{1}{\ln N} \log x$$

is real-valued and nonnegative. The spectral calculus therefore defines a self-adjoint nonnegative operator H_v, and the exponential identity is the same scalar identity applied through the functional calculus. □

Corollary 9443.6 (Finite-window failure blocks the Hamiltonian gate)**.** *If a finite future window has a vector F with*

$$\langle \theta F, F \rangle < 0,$$

then the corresponding self-substitution row cannot pass complete RP, and therefore cannot supply an input Hamiltonian through the OS/RP gate of Definition 9443.2.

Proof. Complete RP requires every finite-window reflection form to be nonnegative. A negative finite-window vector is already a contradiction to that requirement, so no positive contraction P_v representing the complete reflection form can be present. □

Principle 9443.7 (Hilbert–Polya prefilter)**.** *Finite-window RP is a prefilter for any Hilbert–Polya style route that starts from self-similar input. A row failing a finite RP window cannot generate the input Hamiltonian H_v; hence it cannot serve as the positive input layer of a later construction of A_ξ^{HP}. This is a necessary-condition statement only. A row passing all input RP tests would still need an independent spectral bridge from H_v to the completed zeta zero row.*

Principle 9443.8 (Known small Q3 obstructions)**.** *The small Q3 rows already isolated in Definition 9428.19, Definition 9431.15, and Definition 9440.10 are therefore input-Hamiltonian exclusions. The binary representative fails the charged one-site gate, the ternary and quartic representatives fail neutral finite-window RP, the Pythagorean pentary row passes the anchored $L = 2$ neutral gate but fails at anchored $L = 3$, and the pentary palindromic Q3 locus has empty intersection with the anchored $L = 3$ neutral RP class. These are not statements about the truth value of RH; they remove self-similar input rows from the OS/RP source side.*

9443.3 Zeta Upgrade Obligation

Definition 9443.9 (Zeta Hilbert–Polya upgrade)**.** *A Q3-compatible self-substitution row v admits a zeta Hilbert–Polya upgrade when it supplies an input Hamiltonian H_v and there is a natural self-adjoint operator $\mathcal{A}_v$ built from the same input, boundary, or scattering data such that its spectral row satisfies*

$$\gamma \in \mathrm{Spec}(\mathcal{A}_v) \quad \Longleftrightarrow \quad \xi(1/2 + i\gamma) = 0.$$

The construction of H_v is only the input-side part of this definition.

Principle 9443.10 (Two independent bridge obligations)**.** *Two bridge obligations are separated. The input bridge must pass from a self-similar RP row to a positive self-adjoint Hamiltonian H_v. The zeta bridge must identify a self-adjoint spectral row with the completed zeta zeros, or else convert the scattering pole row of Definition 9443.3 into a self-adjoint spectral row. Lax–Phillips, de Branges, and Hermite–Biehler style interfaces belong to the second obligation, not to the finite-window RP computation itself.*

Principle 9443.11 (Three-level zeta interface)**.** *The input programme has three distinct zeta-facing levels:*

level	*row*	*output*
self-substitution RP	$v \in \mathcal{Q}_N,\ M_{\mathcal{B}}(v) \succeq 0$	H_v
boundary transfer	*Gauss/Mayer transfer row*	$Z_{\mathrm{Selberg}}(s)$
modular scattering	$\varphi(s) = \Lambda(2s-1)/\Lambda(2s)$	*Riemann scattering poles*

The first level is where the finite-window Q3 search lives. The second is the boundary determinant interface of Definition 9453.7. The third is the cusp scattering interface of Definition 9452.29. Moving from one level to another is not automatic.

Theorem 9443.12 (Complete RP is necessary for OS-source upgrade)**.** *If a Q3-compatible self-substitution row is to yield a Hilbert–Polya upgrade through the OS/RP input source, then it must satisfy complete RP on all finite future windows used by that source.*

Proof. The OS/RP source requires a positive Hilbert quotient for future observables and a positive contraction transition row before Definition 9443.5 can define H_v. A negative finite-window reflection form prevents the positive quotient and hence prevents the positive transition row. Thus complete RP is a necessary condition for this source route. □

9443.4 Pentary Hermitian Locus

Definition 9443.13 (Anchored three-link neutral window)**.** *For the anchored $L = 3$ neutral audit, set*

$$e_{12} := \delta_1 - \delta_2, \qquad e_{13} := \delta_1 - \delta_3, \qquad e_{23} := \delta_2 - \delta_3,$$

and use the future block family

$$\mathcal{B}_3 := \{0, e_{12}, -e_{12}, e_{13}, -e_{13}, e_{23}, -e_{23}\}.$$

Its RP matrix is

$$M_3(v)_{\alpha,\beta} := \Gamma(\beta - \theta\alpha), \qquad \alpha, \beta \in \mathcal{B}_3.$$

Definition 9443.14 (Pentary Hermitian and RP loci)**.** *Inside $\mathcal{Q}_5$, define*

$$\mathcal{H}^{\mathrm{neu}}_{5,3,1} := \{v \in \mathcal{Q}_5 : M_3(v) = M_3(v)^*\}$$

and

$$\mathcal{R}^{\mathrm{neu}}_{5,3,1} := \{v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1} : M_3(v) \succeq 0\}.$$

Theorem 9443.15 (Hermitian condition as reflected correlation equality)**.** *For $v \in \mathcal{Q}_5$, the anchored three-link matrix is Hermitian if and only if, for all $\alpha, \beta \in \mathcal{B}_3$,*

$$\Gamma(\beta - \theta\alpha) = \Gamma(\theta\beta - \alpha).$$

Proof. The matrix entry is

$$M_{\alpha,\beta} = \Gamma(\beta - \theta\alpha).$$

Hermitian symmetry asks for

$$M_{\alpha,\beta} = \overline{M_{\beta,\alpha}}.$$

Since complex conjugation reverses a balanced correlation block,

$$\overline{M_{\beta,\alpha}} = \overline{\Gamma(\alpha - \theta\beta)} = \Gamma(\theta\beta - \alpha).$$

This is exactly the displayed equality. □

Corollary 9443.16 (Reflection-even rows are Hermitian)**.** *If the balanced correlation function of v is reflection-even in the sense of Definition 9439.3, then $v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1}$.*

Proof. Apply Definition 9439.4 to the finite window $\mathcal{B}_3$. □

Principle 9443.17 (Nonpalindromic feasibility surface)**.** *Palindromic rows lie in the Hermitian locus by Definition 9439.8, but Definition 9440.10 gives*

$$\mathcal{P}_5 \cap \mathcal{R}^{\mathrm{neu}}_{5,3,1} = \varnothing$$

for the anchored three-link class. Therefore the remaining pentary input-Hamiltonian surface is

$$(\mathcal{Q}_5 \setminus \mathcal{P}_5) \cap \mathcal{R}^{\mathrm{neu}}_{5,3,1}.$$

A point on this surface would still be only an OS/RP input candidate: it would provide the finite-window shadow of H_v, not the zeta Hilbert–Polya operator.

Principle 9443.18 (Hermitian before positivity)**.** *For the nonpalindromic pentary search, the finite-window gates occur in a fixed order:*

$$\textit{Q3 phase-sum compatibility}, \qquad M_3(v) = M_3(v)^*, \qquad M_3(v) \succeq 0.$$

The golden symmetric row of Definition 9438.5 passes the scalar link-diagonal filter but fails the Hermitian gate. Thus the Hermitian equations are not cosmetic symmetry conditions; they are the finite-window expression of self-adjointness before the PSD gate can be read as reflection positivity.

Chapter 9444

Innovation Input Pentary Mirror Decomposition Gate

chapter 9443 separates Q3 compatibility, Hermitian reflection symmetry, and positive RP. This chapter supplies the mirror decomposition behind the pentary Hermitian gate. Q3 fixes the symmetric total of the self-substitution word, while the nonpalindromic variables are mirror-antisymmetric phase twists. Hermitian defects are odd under this mirror action, so after the finite carry-closure denominator is cleared they lie in the palindromic ideal.

9444.1 Mirror Coordinates

Definition 9444.1 (Pentary mirror)**.** *For a pentary phase word $v = (v_0, \dots, v_4)$, define*

$$\sigma(v)_a := v_{4-a}.$$

Equivalently,

$$\sigma(v_0, v_1, v_2, v_3, v_4) = (v_4, v_3, v_2, v_1, v_0).$$

The fixed points of σ are exactly the palindromic words (a, b, c, b, a).

Proposition 9444.2 (Q3 is mirror invariant)**.** *If $v \in \mathcal{Q}_5$, then $\sigma(v) \in \mathcal{Q}_5$.*

Proof. The phase sum is unchanged by reversing the five entries:

$$\sum_{a=0}^{4} \sigma(v)_a = v_4 + v_3 + v_2 + v_1 + v_0 = \sum_{a=0}^{4} v_a.$$

Thus the equation $\sum_a v_a = \sqrt{5}$ is preserved. □

Definition 9444.3 (Symmetric and antisymmetric coordinates)**.** *Set*

$$s_0 := \frac{v_0 + v_4}{2}, \qquad d_0 := \frac{v_0 - v_4}{2},$$

$$s_1 := \frac{v_1 + v_3}{2}, \qquad d_1 := \frac{v_1 - v_3}{2},$$

and

$$c := v_2.$$

Then

$$v_0 = s_0 + d_0, \qquad v_4 = s_0 - d_0,$$

and

$$v_1 = s_1 + d_1, \qquad v_3 = s_1 - d_1.$$

The palindromic condition is $d_0 = d_1 = 0$.

Theorem 9444.4 (Q3 only sees the symmetric part)**.** *In the coordinates of Definition 9444.3, the Q3 equation is*

$$2s_0 + 2s_1 + c = \sqrt{5}.$$

It contains no d_0 *or* d_1 *term.*

Proof. Adding the five entries gives

$$v_0 + v_1 + v_2 + v_3 + v_4 = (v_0 + v_4) + (v_1 + v_3) + v_2 = 2s_0 + 2s_1 + c.$$

□

Corollary 9444.5 (Nonpalindromy is not a Q3 obstruction)**.** *The Q3 layer constrains the symmetric skeleton* (s_0, s_1, c) *but does not directly remove the antisymmetric coordinates* (d_0, d_1)*. Any exclusion of nonpalindromic pentary rows must therefore come from Hermitian reflection symmetry, RP positivity, or a stronger bridge row.*

Theorem 9444.6 (Unit-circle mirror coordinates)**.** *The conditions*

$$|v_0| = |v_4| = 1$$

are equivalent to

$$|s_0|^2 + |d_0|^2 = 1, \qquad \mathrm{Re}(s_0\overline{d_0}) = 0.$$

Similarly,

$$|v_1| = |v_3| = 1$$

is equivalent to

$$|s_1|^2 + |d_1|^2 = 1, \qquad \mathrm{Re}(s_1\overline{d_1}) = 0.$$

Proof. For the first pair,

$$|s_0 + d_0|^2 = |s_0|^2 + |d_0|^2 + 2\,\mathrm{Re}(s_0\overline{d_0}),$$

and

$$|s_0 - d_0|^2 = |s_0|^2 + |d_0|^2 - 2\,\mathrm{Re}(s_0\overline{d_0}).$$

If both are 1, their sum and difference give the two displayed conditions. Conversely, those two conditions make both squared moduli equal to 1. The second pair is identical with indices shifted from $0, 4$ to $1, 3$. □

Principle 9444.7 (Antisymmetric phase twist)**.** *When* $s_0 \neq 0$*, the unit-circle equations force* d_0 *to be orthogonal to* s_0 *in the real plane underlying* $\mathbb{C}$*; equivalently*

$$d_0 = it_0 s_0$$

for some real t_0*, with*

$$|s_0|^2(1 + t_0^2) = 1.$$

The same statement holds for d_1 *relative to* s_1*. Thus nonpalindromy is a pair of mirror-antisymmetric phase twists over the Q3 symmetric skeleton.*

9444.2 Mirror Transport of Correlations

Theorem 9444.8 (Mirror word reflects balanced correlations)**.** *For every pentary self-substitution row v and every balanced block α,*

$$\Gamma_{\sigma v}(\alpha) = \Gamma_v(\theta\alpha).$$

Proof. At finite depth K, write

$$n = a_0 + a_1 5 + \cdots + a_{K-1} 5^{K-1}.$$

The phase read for v is

$$u_v^{(K)}(n) = \prod_{\ell=0}^{K-1} v_{a_\ell},$$

whereas the phase read for σv is

$$u_{\sigma v}^{(K)}(n) = \prod_{\ell=0}^{K-1} v_{4-a_\ell}.$$

Let

$$R_K(n) := 5^K - 1 - n.$$

The base-5 digits of $R_K(n)$ are $(4 - a_0, \ldots, 4 - a_{K-1})$, so

$$u_{\sigma v}^{(K)}(n) = u_v^{(K)}(R_K(n)).$$

For a balanced block $\alpha = \sum_j e_j \delta_j$, form the finite-depth average over indices for which all shifted entries lie inside the depth K interval. Boundary omissions have density tending to zero. Since

$$R_K(n + j) = R_K(n) - j,$$

the substitution $m = R_K(n)$ gives, in the limit,

$$\Gamma_{\sigma v}(\alpha) = \Gamma_v\left(\sum_j e_j \delta_{-j}\right).$$

Balanced correlations are invariant under common translation of the block, hence

$$\Gamma_v\left(\sum_j e_j \delta_{-j}\right) = \Gamma_v\left(\sum_j e_j \delta_{1-j}\right) = \Gamma_v(\theta\alpha).$$

□

Corollary 9444.9 (Palindromy gives reflection evenness)**.** *If $\sigma v = v$, then*

$$\Gamma_v(\alpha) = \Gamma_v(\theta\alpha)$$

for every balanced block α.

Proof. Apply Definition 9444.8 and use $\sigma v = v$. □

9444.3 Mirror-Odd Hermitian Defects

Definition 9444.10 (Pairwise Hermitian defect)**.** *For future balanced blocks α and β, define*

$$H_{\alpha,\beta}(v) := \Gamma_v(\beta - \theta\alpha) - \Gamma_v(\theta\beta - \alpha).$$

The anchored three-link Hermitian condition is the vanishing of these defects for the block pairs in $\mathcal{B}_3$ of Definition 9443.13.

Theorem 9444.11 (Hermitian defects are mirror-odd)**.** *For future balanced blocks α and β,*

$$H_{\alpha,\beta}(\sigma v) = -H_{\alpha,\beta}(v).$$

Proof. By Definition 9444.8,

$$H_{\alpha,\beta}(\sigma v) = \Gamma_v(\theta(\beta - \theta\alpha)) - \Gamma_v(\theta(\theta\beta - \alpha)).$$

The reflected blocks are

$$\theta(\beta - \theta\alpha) = \theta\beta - \alpha$$

and

$$\theta(\theta\beta - \alpha) = \beta - \theta\alpha.$$

Therefore

$$H_{\alpha,\beta}(\sigma v) = \Gamma_v(\theta\beta - \alpha) - \Gamma_v(\beta - \theta\alpha) = -H_{\alpha,\beta}(v).$$

□

Corollary 9444.12 (Hermitian defects vanish on the palindromic locus)**.** *If $v \in \mathcal{P}_5$, then*

$$H_{\alpha,\beta}(v) = 0$$

for all future balanced α, β.

Proof. For a palindromic word, $\sigma v = v$. Hence

$$H_{\alpha,\beta}(v) = H_{\alpha,\beta}(\sigma v) = -H_{\alpha,\beta}(v),$$

so $H_{\alpha,\beta}(v) = 0$. □

9444.4 Palindromic Ideal

Definition 9444.13 (Palindromic Laurent ideal)**.** *Let*

$$\mathcal{L}_5 := \mathbb{C}[v_0^{\pm 1}, v_1^{\pm 1}, v_2^{\pm 1}, v_3^{\pm 1}, v_4^{\pm 1}]$$

and define

$$I_{\text{pal}} := (v_0 - v_4,\ v_1 - v_3) \subset \mathcal{L}_5.$$

Proposition 9444.14 (The palindromic ideal is prime)**.** *The ideal I_{pal} is prime.*

Proof. The quotient is

$$\mathcal{L}_5/I_{\text{pal}} \cong \mathbb{C}[v_0^{\pm 1}, v_1^{\pm 1}, v_2^{\pm 1}],$$

where v_4 is identified with v_0 and v_3 with v_1. This quotient is a Laurent polynomial domain, so I_{pal} is prime. □

Theorem 9444.15 (Cleared Hermitian defects lie in the palindromic ideal)**.** *Let $H_{\alpha,\beta}(v)$ be a pairwise Hermitian defect whose correlations are computed by a finite carry-closure linear system, and let*

$$\Delta_{\alpha,\beta}(v)$$

be a common determinant denominator for that system. Then

$$\Delta_{\alpha,\beta}(v)H_{\alpha,\beta}(v) \in I_{\mathrm{pal}}.$$

Equivalently, there are Laurent polynomials $A_{\alpha,\beta}(v)$ and $B_{\alpha,\beta}(v)$ such that

$$\Delta_{\alpha,\beta}(v)H_{\alpha,\beta}(v) = A_{\alpha,\beta}(v)(v_0 - v_4) + B_{\alpha,\beta}(v)(v_1 - v_3).$$

Proof. By the finite carry-closure evaluator of Definition 9429.19, the correlations in $H_{\alpha,\beta}$ are rational Laurent functions of $v_0, \ldots, v_4$. Multiplying by a common determinant denominator gives a Laurent polynomial

$$F_{\alpha,\beta}(v) := \Delta_{\alpha,\beta}(v)H_{\alpha,\beta}(v).$$

By Definition 9444.12, this polynomial vanishes after the substitutions $v_4 = v_0$ and $v_3 = v_1$. In the Laurent polynomial quotient by I_{pal}, that means the class of $F_{\alpha,\beta}$ is zero. Hence $F_{\alpha,\beta} \in I_{\mathrm{pal}}$, which is the displayed decomposition. □

Corollary 9444.16 (Basic three-link defects have nonpalindromic factors)**.** *The defects*

$$H_{12,13}, \qquad H_{12,23}, \qquad H_{13,23}$$

of Definition 9446.5 admit decompositions of the form

$$H_i(v) = A_i(v)(v_0 - v_4) + B_i(v)(v_1 - v_3)$$

on every chart where the corresponding carry-closure determinant denominator is nonzero.

Proof. Apply Definition 9444.15 to the three defects and divide by the nonzero determinant denominator on the chart under consideration. □

9444.5 Rank Form of the Remaining Problem

Definition 9444.17 (Hermitian coefficient matrix)**.** *Choose finite Hermitian defects $H_1, \ldots, H_m$ and write, on a common nonzero-denominator chart,*

$$H_\ell(v) = A_\ell(v)(v_0 - v_4) + B_\ell(v)(v_1 - v_3).$$

Define

$$K(v) := \begin{pmatrix} A_1(v) & B_1(v) \\ \vdots & \vdots \\ A_m(v) & B_m(v) \end{pmatrix}$$

and

$$R(v) := \begin{pmatrix} v_0 - v_4 \\ v_1 - v_3 \end{pmatrix}.$$

Theorem 9444.18 (Nonpalindromic Hermitian points force rank drop)**.** *If a point on the chosen chart satisfies the selected Hermitian equations and is nonpalindromic, then*

$$\operatorname{rank} K(v) \leq 1.$$

Proof. The selected Hermitian equations are

$$K(v)R(v) = 0.$$

If the point is nonpalindromic, then $R(v) \neq 0$. Hence $K(v)$ has a nonzero kernel vector. Since it has two columns, its rank is at most 1. □

Corollary 9444.19 (Minor test for Hermitian rigidity)**.** *If two selected Hermitian defects give a* 2×2 *minor*

$$J_{ij}(v) := A_i(v)B_j(v) - A_j(v)B_i(v)$$

that is nonzero on $\mathcal{Q}_5$ *in the relevant nonzero-denominator chart, then no nonpalindromic point of that chart satisfies the selected Hermitian equations.*

Proof. If such a point existed, the rank-drop theorem would force every 2×2 minor of $K(v)$ to vanish. This contradicts $J_{ij}(v) \neq 0$. □

Principle 9444.20 (Mirror decomposition of the main line)**.** *The pentary main line has a two-stage algebraic form. Q3 permits mirror-antisymmetric phase twists because it sees only the symmetric phase sum. Hermitian reflection symmetry makes the defect functions odd under the mirror, forcing their cleared carry-closure forms into* I_{pal}*. Thus a nonpalindromic Hermitian row can exist only at a rank-drop point of the Hermitian coefficient matrix.*

Chapter 9445

Innovation Input Pythagorean Local Hermitian Rigidity

chapter 9444 shows that Q3 compatibility leaves two mirror-antisymmetric phase-twist directions, while Hermitian three-link symmetry can only hold at rank-drop points of the resulting defect matrix. The Pythagorean row from Definition 9431.4 is the unique palindromic row surviving the anchored $L = 2$ positive semidefinite gate, up to complex conjugation, and it fails anchored $L = 3$ positivity by Definition 9431.10. This chapter records the local Hermitian fact at that boundary point: the surviving palindromic row is not the endpoint of a first-order nonpalindromic Hermitian branch.

9445.1 Antisymmetric Tangent Directions

Definition 9445.1 (Pythagorean antisymmetric phase perturbation)**.** *Let*

$$a := \frac{1+2i}{\sqrt{5}}, \qquad b := \frac{2-i}{\sqrt{5}}.$$

For $x, y \in \mathbb{R}$, define

$$v(\varepsilon; x, y) := \left(ae^{i\varepsilon x}, be^{i\varepsilon y}, -a, be^{-i\varepsilon y}, ae^{-i\varepsilon x}\right).$$

At $\varepsilon = 0$ this is the Pythagorean row $v_\star = (a, b, -a, b, a)$. The pair (x, y) is the mirror-antisymmetric tangent coordinate over the two edge pairs (v_0, v_4) and (v_1, v_3).

Proposition 9445.2 (Antisymmetric perturbations are Q3-tangent)**.** *For every $x, y \in \mathbb{R}$,*

$$\frac{d}{d\varepsilon} \sum_{j=0}^{4} v_j(\varepsilon; x, y)\bigg|_{\varepsilon=0} = 0.$$

Thus (x, y) is tangent to $\mathcal{Q}_5$ at $v_\star$.

Proof. The derivatives of the mirror-paired entries are

$$\frac{d}{d\varepsilon} v_0(\varepsilon; x, y)\bigg|_{\varepsilon=0} = iax, \qquad \frac{d}{d\varepsilon} v_4(\varepsilon; x, y)\bigg|_{\varepsilon=0} = -iax,$$

and

$$\frac{d}{d\varepsilon} v_1(\varepsilon; x, y)\bigg|_{\varepsilon=0} = iby, \qquad \frac{d}{d\varepsilon} v_3(\varepsilon; x, y)\bigg|_{\varepsilon=0} = -iby.$$

The middle entry is constant. The sum of the five derivatives is therefore zero. □

Definition 9445.3 (Linearized three-link Hermitian defect)**.** *For $\alpha, \beta \in \mathcal{B}_3$, set*

$$\dot{H}_{\alpha,\beta}(x,y) := \frac{d}{d\varepsilon}\left(M_3(v(\varepsilon;x,y))_{\alpha,\beta} - \overline{M_3(v(\varepsilon;x,y))_{\beta,\alpha}}\right)\Big|_{\varepsilon=0}.$$

If the anchored three-link Hermitian condition is preserved to first order along the perturbation, then every $\dot{H}_{\alpha,\beta}(x,y)$ vanishes.

9445.2 Defect Linearization

Lemma 9445.4 (Pythagorean three-link defect slopes)**.** *At $v_\star$, the antisymmetric perturbation of Definition 9445.1 has the following nonzero basic first-order Hermitian defects:*

$$\dot{H}_{0,e_{13}}(x,y) = -\frac{4i}{5}(x-y),$$

$$\dot{H}_{0,-e_{13}}(x,y) = \frac{4i}{5}(x-y),$$

$$\dot{H}_{e_{12},e_{23}}(x,y) = \frac{4i}{5}(2x-y),$$

$$\dot{H}_{-e_{12},-e_{23}}(x,y) = -\frac{4i}{5}(2x-y),$$

$$\dot{H}_{e_{13},-e_{23}}(x,y) = -\frac{4}{5}(x-2y),$$

and

$$\dot{H}_{-e_{13},e_{23}}(x,y) = -\frac{4}{5}(x-2y).$$

All other first-order defects in the anchored three-link window either vanish or are obtained from these by the forced sign and conjugation relations.

Proof. Apply the finite carry-closure recursion of Definition 9429.19 to the balanced blocks that occur in the entries of $M_3(v)$. For each such block α, the correlation satisfies a finite system of the form

$$\Gamma_v(\alpha) = \frac{1}{5}\sum_{r=0}^{4} C_r(v,\alpha)\Gamma_v(\alpha_r),$$

where the branch coefficients $C_r(v,\alpha)$ are Laurent monomials in the five phase entries. Differentiating this finite system at $v_\star$ gives

$$\dot{\Gamma}(\alpha) = \frac{1}{5}\sum_{r=0}^{4}\left[\dot{C}_r(\alpha)\Gamma_{v_\star}(\alpha_r) + C_r(v_\star,\alpha)\dot{\Gamma}(\alpha_r)\right].$$

The derivative of the phase row is

$$\dot{v}_0 = iax, \qquad \dot{v}_4 = -iax, \qquad \dot{v}_1 = iby, \qquad \dot{v}_3 = -iby, \qquad \dot{v}_2 = 0.$$

Substituting these five derivatives, solving the finite linearized carry-closure system, and then forming the Hermitian differences of Definition 9445.3 gives exactly the displayed six slopes. The remaining entries of the Hermitian defect matrix are fixed by skew-Hermitian symmetry of $M_3 - M_3^*$ and by the sign symmetry between opposite neutral links. □

9445.3 Local Rigidity

Theorem 9445.5 (Pythagorean first-order Hermitian rigidity)**.** *If the antisymmetric perturbation $v(\varepsilon; x, y)$ preserves the anchored three-link Hermitian condition to first order at $\varepsilon = 0$, then*

$$x = 0, \qquad y = 0.$$

Equivalently, $v_\star$ has no nonzero mirror-antisymmetric tangent direction inside the anchored $L = 3$ Hermitian locus.

Proof. First-order Hermitian preservation forces every defect slope in Definition 9445.4 to vanish. From

$$\dot{H}_{0,e_{13}}(x, y) = -\frac{4i}{5}(x - y)$$

one gets $x - y = 0$. From

$$\dot{H}_{e_{12},e_{23}}(x, y) = \frac{4i}{5}(2x - y)$$

one gets $2x - y = 0$. Hence $x = y$ and $2x = y$, so $x = y = 0$. The remaining linear factor $x - 2y$ then vanishes automatically. □

Corollary 9445.6 (No first-order nonpalindromic branch at the Pythagorean row)**.** *The Pythagorean row is not the endpoint of a smooth nonpalindromic anchored three-link Hermitian branch whose tangent has nonzero mirror-antisymmetric part. In particular, a nonpalindromic point of $\mathcal{H}^{\mathrm{neu}}_{5,3,1}$, if one exists, cannot be obtained by leaving $v_\star$ through a nonzero first-order antisymmetric phase twist.*

Proof. A smooth curve through $v_\star$ whose first-order displacement changes the mirror-paired phases in opposite directions has an antisymmetric tangent coordinate (x, y). If the curve remains in $\mathcal{H}^{\mathrm{neu}}_{5,3,1}$ to first order, the tangent must satisfy Definition 9445.5; therefore it is zero. This excludes a nonzero first-order nonpalindromic departure at that point. □

Corollary 9445.7 (Conjugate Pythagorean first-order rigidity)**.** *The complex conjugate row $\overline{v_\star}$ has the same first-order Hermitian rigidity: every mirror-antisymmetric tangent direction that preserves the anchored three-link Hermitian condition to first order is zero.*

Proof. Complex conjugation carries the anchored three-link matrix to its conjugate matrix and carries Hermitian defects to their conjugates. The linear factors in Definition 9445.4, namely

$$x - y, \qquad 2x - y, \qquad x - 2y,$$

are unchanged as real equations. The proof of Definition 9445.5 therefore applies verbatim. □

Principle 9445.8 (Local role of the Pythagorean boundary)**.** *Q3 compatibility sees only the symmetric phase sum, so it allows the antisymmetric phase twists (x, y) at first order. The anchored three-link Hermitian equations see three incompatible linear reads,*

$$x - y, \qquad 2x - y, \qquad x - 2y,$$

and their common zero is the palindromic tangent. Thus the Pythagorean row and its conjugate are not local sources of a nonpalindromic Hermitian branch. The unresolved pentary problem is therefore pushed to rank-drop points away from these palindromic $L = 2$ boundary points, in the sense of Definition 9444.18.

Chapter 9446

Innovation Input Pentary Hermitian Rigidity Gate

chapter 9444 shows that cleared Hermitian defects lie in the palindromic ideal and that nonpalindromic Hermitian points must occur at rank-drop points of a coefficient matrix. This chapter isolates the stronger certificate that would close the pentary three-link gate outright: enough coefficient rows must be nondegenerate on $\mathcal{Q}_5$ to force palindromy before the PSD condition is reached.

9446.1 Hermitian Defect Row

Definition 9446.1 (Three-link Hermitian defect)**.** *For $v \in \mathcal{Q}_5$, let $M_3(v)$ be the anchored three-link matrix of Definition 9443.13. Define the Hermitian defect row by*

$$\mathcal{D}_3(v) := M_3(v) - M_3(v)^*.$$

Thus

$$v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1} \quad \Longleftrightarrow \quad \mathcal{D}_3(v) = 0.$$

Definition 9446.2 (Nonpalindromic defect coordinates)**.** *For a pentary phase word $v = (v_0, \dots, v_4)$, define*

$$R_0(v) := v_0 - v_4, \qquad R_1(v) := v_1 - v_3.$$

The equalities

$$R_0(v) = 0, \qquad R_1(v) = 0$$

are exactly the condition that v has the palindromic shape (a, b, c, b, a).

Principle 9446.3 (Pentary Hermitian rigidity target)**.** *The active algebraic target is the implication*

$$v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1} \quad \Longrightarrow \quad R_0(v) = R_1(v) = 0.$$

Equivalently, the anchored $L = 3$ Hermitian locus inside $\mathcal{Q}_5$ is the palindromic locus, up to the evident conjugation and reflection symmetries that preserve the finite-window RP spectrum.

Theorem 9446.4 (Hermitian rigidity empties the pentary RP locus)**.** *Assume the implication of Definition 9446.3. Then*

$$\mathcal{R}^{\mathrm{neu}}_{5,3,1} = \varnothing.$$

Proof. If $v \in \mathcal{R}^{\mathrm{neu}}_{5,3,1}$, then by definition $v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1}$. The rigidity implication gives $v \in \mathcal{P}_5$. But Definition 9440.10 gives

$$\mathcal{P}_5 \cap \mathcal{R}^{\mathrm{neu}}_{5,3,1} = \varnothing.$$

Hence no such v exists. □

9446.2 Cross-Entry Equations

Definition 9446.5 (Basic cross-entry defects)**.** *With*

$$e_{12} := \delta_1 - \delta_2, \qquad e_{13} := \delta_1 - \delta_3, \qquad e_{23} := \delta_2 - \delta_3,$$

define

$$H_{12,13}(v) := \Gamma(e_{13} - \theta e_{12}) - \Gamma(\theta e_{13} - e_{12}),$$
$$H_{12,23}(v) := \Gamma(e_{23} - \theta e_{12}) - \Gamma(\theta e_{23} - e_{12}),$$

and

$$H_{13,23}(v) := \Gamma(e_{23} - \theta e_{13}) - \Gamma(\theta e_{23} - e_{13}).$$

They are entries of the Hermitian defect matrix $\mathcal{D}_3(v)$.

Proposition 9446.6 (Cross-entry normalized blocks)**.** *The three basic cross-entry defects can be written as*

$$H_{12,13}(v) = \Gamma(\delta_0 - \delta_1 + \delta_2 - \delta_4) - \Gamma(-\delta_0 + \delta_2 - \delta_3 + \delta_4),$$
$$H_{12,23}(v) = \Gamma(\delta_0 - \delta_1 + \delta_3 - \delta_4) - \Gamma(-\delta_0 + \delta_1 - \delta_3 + \delta_4),$$

and

$$H_{13,23}(v) = \Gamma(\delta_0 - \delta_2 + \delta_4 - \delta_5) - \Gamma(-\delta_0 + \delta_1 - \delta_3 + \delta_5).$$

Proof. For $H_{12,13}$,

$$e_{13} - \theta e_{12} = (\delta_1 - \delta_3) - (\delta_0 - \delta_{-1}).$$

After translating all sites by 1, this becomes $\delta_0 - \delta_1 + \delta_2 - \delta_4$. Also

$$\theta e_{13} - e_{12} = (\delta_0 - \delta_{-2}) - (\delta_1 - \delta_2),$$

which translates by 2 to $-\delta_0 + \delta_2 - \delta_3 + \delta_4$.

For $H_{12,23}$,

$$e_{23} - \theta e_{12} = (\delta_2 - \delta_3) - (\delta_0 - \delta_{-1}),$$

which translates by 1 to $\delta_0 - \delta_1 + \delta_3 - \delta_4$. Similarly

$$\theta e_{23} - e_{12} = (\delta_{-1} - \delta_{-2}) - (\delta_1 - \delta_2),$$

which translates by 2 to $-\delta_0 + \delta_1 - \delta_3 + \delta_4$.

For $H_{13,23}$,

$$e_{23} - \theta e_{13} = (\delta_2 - \delta_3) - (\delta_0 - \delta_{-2}),$$

which translates by 2 to $\delta_0 - \delta_2 + \delta_4 - \delta_5$. Finally

$$\theta e_{23} - e_{13} = (\delta_{-1} - \delta_{-2}) - (\delta_1 - \delta_3),$$

which translates by 2 to $-\delta_0 + \delta_1 - \delta_3 + \delta_5$. □

Principle 9446.7 (Golden obstruction entry)**.** *The first displayed difference is the crossed entry where Definition 9438.5 detects non-Hermitian behaviour of the golden symmetric row. Thus the same local defect that refutes diagonal sufficiency is also the first candidate equation for proving nonpalindromic rigidity.*

9446.3 Finite Algebraic Evaluator

Proposition 9446.8 (Cross-entry defects have finite carry-closure form)**.** *For each basic cross-entry defect in Definition 9446.5, there is a finite carry-closure linear system*

$$X = A(v)X + b(v)$$

whose entries are Laurent polynomials in $v_0, \ldots, v_4$ and whose solution gives the two correlation values appearing in that defect.

Proof. Each displayed block in Definition 9446.6 is a finite balanced block. Apply the carry-closure construction of Definition 9429.18 and the finite linear system of Definition 9429.19 to the finite set consisting of these blocks. The resulting unknown vector contains the required correlations, and all coefficients are the residue branch Laurent monomials supplied by the self-substitution rule. □

Corollary 9446.9 (Hermitian locus is an explicit algebraic-trigonometric set)**.** *After imposing the exact Q3 constraint*

$$v \in \mathcal{Q}_5,$$

the equations

$$\mathcal{D}_3(v) = 0$$

are finite algebraic-trigonometric equations obtained from carry-closure linear systems. In particular, the Hermitian rigidity target can be attacked without sampling long finite words.

Proof. The entries of $M_3(v)$ are finite-window correlations. By Definition 9429.19, each is obtained from a finite carry-closure system. Hermitian symmetry is the finite list of equalities of Definition 9443.15. □

9446.4 Factor Extraction Gate

Definition 9446.10 (Rigidity factor certificate)**.** *A rigidity factor certificate for the anchored pentary three-link window consists of two functions $A_0(v)$ and $A_1(v)$, obtained from the carry-closure algebraic evaluator, such that on $\mathcal{Q}_5$*

$$H_{12,13}(v) = A_0(v)R_0(v),$$

$$H_{12,23}(v) = A_1(v)R_1(v),$$

and

$$A_0(v) \neq 0, \qquad A_1(v) \neq 0$$

on the Q3 locus under consideration.

Theorem 9446.11 (Factor certificate gives Hermitian rigidity)**.** *If a rigidity factor certificate exists, then*

$$v \in \mathcal{H}^{\mathrm{neu}}_{5,3,1} \quad \Longrightarrow \quad v \in \mathcal{P}_5.$$

Proof. If v is Hermitian, then every entry of the defect matrix vanishes, in particular

$$H_{12,13}(v) = 0, \qquad H_{12,23}(v) = 0.$$

The factor certificate gives

$$A_0(v)R_0(v) = 0, \qquad A_1(v)R_1(v) = 0.$$

Since $A_0(v)$ and $A_1(v)$ are nonzero, $R_0(v) = R_1(v) = 0$. By Definition 9446.2, this is exactly $v \in \mathcal{P}_5$. □

Corollary 9446.12 (Factor certificate closes the pentary three-link gate)**.** *If a rigidity factor certificate exists, then*

$$\mathcal{R}^{\mathrm{neu}}_{5,3,1} = \varnothing.$$

Proof. Combine Definition 9446.11 with Definition 9446.4. □

Principle 9446.13 (Algebraic extraction over blind search)**.** *The pentary nonpalindromic problem should be treated as a factor extraction problem, not as an unconstrained numerical search. The carry-closure evaluator gives exact algebraic functions for the cross-entry defects, and the decisive question is whether the nonpalindromic defects R_0 and R_1 divide the basic Hermitian defects with nonvanishing cofactors on $\mathcal{Q}_5$.*

9446.5 Audit Pattern

Principle 9446.14 (Hermitian audit pattern)**.** *Constrained numerical audits of the Q3 surface using the exact carry-closure evaluator have a consistent pattern: drives toward small anchored $L = 3$ Hermitian defect return to the palindromic equalities*

$$v_0 = v_4, \qquad v_1 = v_3,$$

while the Hermitian part still has a negative least eigenvalue on the observed palindromic limits. This pattern is not a proof of rigidity. It is evidence for the factor-extraction gate of Definition 9446.10.

Principle 9446.15 (Pentary three-link boundary)**.** *The anchored pentary $L = 3$ neutral RP question is reduced to one of two outcomes. A rigidity factor certificate proves that no pentary Q3 row survives the three-link gate. A nonpalindromic point in*

$$(\mathcal{Q}_5 \setminus \mathcal{P}_5) \cap \mathcal{R}^{\mathrm{neu}}_{5,3,1}$$

would be the first genuinely nonpalindromic OS/RP input candidate at this window. In both outcomes, the statement remains an input-Hamiltonian statement and does not identify the Riemann zero spectrum.

Chapter 9447

Innovation Input Dynamics and Spectral Boundary

chapter 9424 separates the positive innovation Gram row from the reflection-positive transition row. This chapter records the dynamical content of the transition row: diffusion geometry, relativistic dispersion, mass as a correlation scale, and spectral dimension are reads of the input-flow operator, not further consequences of the Hilbert Gram tensor.

9447.1 Spatialized Input and Diffusion Read

Definition 9447.1 (Spatialized reversible input)**.** *A spatialized reversible input over a lattice window consists of a reversible finite or periodic-state transition operator P whose states carry a displayed displacement row*

$$\Delta x \in \mathbb{R}^d.$$

The input is translation-invariant when P is diagonalized by Fourier characters, with multiplier $\widehat{P}(k)$.

Definition 9447.2 (Diffusion inverse metric read)**.** *Assume $\widehat{P}(k) > 0$ near $k = 0$ and let $\Delta T > 0$ be the tick length. The reconstructed Hamiltonian multiplier is*

$$H(k) := -\Delta T^{-1} \log \widehat{P}(k).$$

If

$$H(k) = \frac{1}{2} g^{ij} k_i k_j + O(|k|^3),$$

then g^{ij} is the diffusion inverse-metric read of the input flow.

Theorem 9447.3 (Low-momentum meeting)**.** *For a centered translation-invariant input with finite second moment,*

$$\mathbb{E}[\Delta x^i] = 0, \qquad C^{ij} := \mathbb{E}[\Delta x^i \Delta x^j],$$

the low-momentum expansion satisfies

$$H(k) = \frac{1}{2\Delta T} C^{ij} k_i k_j + O(|k|^3).$$

Thus the diffusion inverse metric is

$$g^{ij} = \frac{C^{ij}}{\Delta T}.$$

Proof. The characteristic multiplier has expansion

$$\widehat{P}(k) = 1 - \frac{1}{2}C^{ij}k_ik_j + O(|k|^3)$$

because the mean displacement is zero. The logarithm satisfies $-\log(1-a+O(|k|^3)) = a+O(|k|^3)$ for $a = \frac{1}{2}C^{ij}k_ik_j$. □

Example 9447.4 (One-dimensional local chain)**.** *For jumps $-\ell$, 0, and $+\ell$ with probabilities q, $1-2q$, and q,*

$$\widehat{P}(k) = 1 - 2q(1-\cos(k\ell)).$$

When $q \le 1/4$, this multiplier is nonnegative for every lattice momentum. Near 0,

$$H(k) = \frac{q\ell^2}{\Delta T}k^2 + O(k^4\ell^4) = \frac{1}{2}g^{xx}k^2 + O(k^4),$$

with

$$g^{xx} = \frac{2q\ell^2}{\Delta T}.$$

Principle 9447.5 (Kinematic and dynamic geometries are different rows)**.** *The kinematic geometry h^η of Definition 9424.7 is a covariance or Fisher-covariance read of the innovation one-form. The dynamic geometry g^{ij} above is a low-momentum read of $-\log P$. A bridge may identify them only after it supplies a comparison row between the covector packet used by Θ and the displacement statistics used by P.*

9447.2 Relativistic Dispersion

Definition 9447.6 (Relativistic input multiplier)**.** *For $m \ge 0$ and tick length ΔT, the relativistic input multiplier is*

$$\widehat{P}_m(k) := \exp\left(-\Delta T(\sqrt{|k|^2+m^2} - m)\right).$$

Lemma 9447.7 (Subordination positivity)**.** *For $t > 0$ and $m \ge 0$, the function*

$$e^{-t\sqrt{|k|^2+m^2}}$$

is positive definite in k. Equivalently, it is a positive mixture of Gaussian multipliers:

$$e^{-t\sqrt{|k|^2+m^2}} = \int_0^\infty e^{-s|k|^2}e^{-sm^2}\eta_t(ds)$$

with a positive half-stable subordinator measure η_t.

Proof. This is the classical half-stable subordination formula for the square-root semigroup. Since each Gaussian multiplier is positive definite and the mixing measure is positive, the mixture is positive definite. □

Theorem 9447.8 (Relativistic Hamiltonian realization)**.** *The multiplier of Definition 9447.6 is positive definite and positive. Therefore it determines a reflection-positive input row after the usual lattice periodization when a lattice window is used. Its reconstructed Hamiltonian is*

$$H_m(k) = \sqrt{|k|^2+m^2} - m.$$

After analytic continuation of the reconstructed unitary row, the group velocity satisfies

$$\left|\nabla_k \sqrt{|k|^2 + m^2}\right| = \frac{|k|}{\sqrt{|k|^2 + m^2}} \leq 1.$$

Proof. Positivity follows from Definition 9447.7 and multiplication by the positive scalar $e^{\Delta T m}$. The identity $H_m = -\Delta T^{-1} \log \widehat{P}_m$ is immediate. The velocity bound is the displayed derivative estimate. □

Principle 9447.9 (Speed is an input-scale read)**.** *The normalized speed bound in Definition 9447.8 is not a primitive spacetime constant inside the certificate. It is the unit obtained after the input jump scale and the tick length ΔT have been fixed. Changing those rows changes the empirical calibration while preserving the formal multiplier identity.*

Theorem 9447.10 (Mass as correlation length)**.** *For $m > 0$, the one-dimensional spatial law associated to the relativistic multiplier has exponential tail*

$$\rho_m(x) \sim C_m e^{-m|x|} |x|^{-3/2}$$

up to the standard Bessel-kernel factor. Thus m^{-1} is the input-flow correlation length. For $m = 0$, the law is Cauchy-type with tail $|x|^{-2}$ and infinite variance.

Proof. The inverse Fourier transform of the square-root semigroup is the relativistic Cauchy kernel, expressible through K_1. The large-argument asymptotic of K_1 gives the exponential factor for $m > 0$. Setting $m = 0$ gives the ordinary Cauchy tail. □

Corollary 9447.11 (Mass-variance split)**.** *The finite-mass input row has a finite second-moment regime and hence a low-momentum diffusion geometry. The massless row has the correct light-cone scaling but no finite covariance metric. This is not a failure of the input row; it marks the boundary between covariance geometry and Cauchy scaling.*

9447.3 Stable Inputs and Spectral Dimension

Definition 9447.12 (Stable input multiplier)**.** *For $0 < \alpha \leq 2$, the α-stable input multiplier is*

$$\widehat{P}_\alpha(k) = \exp(-\Delta T |k|^\alpha).$$

Theorem 9447.13 (Stable reflection-positive row)**.** *The multiplier $\widehat{P}_\alpha$ is positive definite for $0 < \alpha \leq 2$, so it defines a reflection-positive input row. Its Hamiltonian is*

$$H_\alpha(k) = |k|^\alpha.$$

Proof. The function $|k|^\alpha$ is conditionally negative definite for $0 < \alpha \leq 2$. By the Levy–Khintchine theorem, $e^{-t|k|^\alpha}$ is positive definite for every $t > 0$. □

Theorem 9447.14 (Spectral dimension read)**.** *For an α-stable input in d spatial dimensions,*

$$\operatorname{tr} e^{-TH_\alpha} \propto \int_{\mathbb{R}^d} e^{-T|k|^\alpha}\, dk \propto T^{-d/\alpha}.$$

With the convention

$$\operatorname{tr} e^{-TH} \sim T^{-d_s/2},$$

the spectral dimension is

$$d_s = \frac{2d}{\alpha}.$$

Proof. The scaling change $k = T^{-1/\alpha}u$ gives the power $T^{-d/\alpha}$. □

Principle 9447.15 (Dimension split)**.** *The framework carries at least two independent dimension reads. The algebraic dimension is constrained by rank budgets of the innovation packet, such as Definition 9424.12. The spectral dimension is controlled by the tail exponent α of the input-flow multiplier. They agree only in the Gaussian case $\alpha = 2$.*

9447.4 Dynamic Output Table

Principle 9447.16 (Dynamic parameter locations)**.** *The dynamical parameters have separate certificate locations:*

row	*parameter*	*visible effect*
tick scale	ΔT	*energy unit*
finite second moment	C^{ij}	*diffusion inverse metric*
relativistic tail	m	*correlation length and mass*
stable tail	α	*spectral dimension*
transition positivity	$P \succeq 0$	*reflection positivity*

No entry of this table is supplied by the positive innovation Gram row alone.

Principle 9447.17 (Propagation boundary)**.** *The input-flow row can supply a Hamiltonian, a low-energy diffusion read, and a Wick-facing relativistic dispersion. It does not by itself supply Einstein dynamics, matter coupling, or equality with the tick-reflected metric of Definition 9423.14. Those require explicit bridge rows between local soldering data and the global transition operator.*

Chapter 9448

Innovation Input Thermodynamic Geometry Boundary

chapter 9447 records the dynamic metric rows of an input flow. This chapter records the adjacent thermodynamic geometry row: the meeting of the innovation information metric with the diffusion metric, the curvature defect of that meeting, the entropy-production correction, the entanglement-area read, and the Jacobson-style dictionary. The chapter locates the ingredients of a thermodynamic gravity argument inside the machine; it does not assert an Einstein field-equation derivation.

9448.1 Coherent Family and Meeting Law

Lemma 9448.1 (Two-dimensional conformal diffusion exactness)**.** *Let a two-dimensional position-dependent jump row have generator $-\frac{1}{2}g(x)\Delta$. For the conformal metric $G = g^{-1}\delta$, this is exactly $-\frac{1}{2}\Delta_G$, and the reversible volume density is proportional to $1/g$.*

Proof. In two dimensions,

$$\Delta_G = \frac{1}{\sqrt{|G|}}\partial_i\left(\sqrt{|G|}G^{ij}\partial_j\right) = g\Delta.$$

The detailed-balance density is the inverse jump rate, so the chain is reversible on the same volume row used by the Laplace–Beltrami operator. □

Definition 9448.2 (Machine coherent family)**.** *Let H be the reconstructed Hamiltonian of a reflection-positive spatialized input, and let $T > 0$ be the logarithmic age. The machine coherent family based at a position z is the normalized state*

$$\psi_z^{(T)} \propto e^{-TH}\delta_z.$$

Its innovation information metric is the QGT real part for the parameter z, written $h_{ij}(z,T)$.

Theorem 9448.3 (Meeting law)**.** *For a translation-invariant finite-variance input with*

$$H(k) = \frac{1}{2}g^{ij}k_ik_j + O(|k|^4),$$

the coherent-family QGT metric satisfies, in the low-momentum regime,

$$h_{ij}(T) = \frac{1}{2T}(g^{-1})_{ij}.$$

Thus the kinematic QGT metric and the dynamic diffusion metric meet as inverse matrices, with conformal factor $1/(2T)$.

Proof. The Fourier weight of the coherent family is proportional to

$$e^{-2TH(k)}.$$

At low momentum this is a centered Gaussian with exponent $-Tg^{ij}k_ik_j$. Since the constant-modulus phase packet reads the QGT metric as the covariance of the covectors k_i, the covariance is $(2T)^{-1}g^{-1}$. □

Principle 9448.4 (Anisotropic closure of the metric comparison)**.** *The meeting law is matrix-valued. It includes anisotropic and off-diagonal diffusion tensors; it is not merely a scalar proportionality. The correction of the diffusion tensor as an inverse metric is therefore a prerequisite for the comparison row.*

Principle 9448.5 (Flat entropy production)**.** *For the Gaussian coherent family in* d *dimensions,*

$$\mathrm{tr}(h\,g) = \frac{d}{2T}.$$

The Shannon entropy of the corresponding position density has derivative $d/(2T)$*. Hence the flat meeting row reads*

$$\mathrm{tr}(h\,g) = \dot{S}.$$

This is the entropy-production form of the meeting law.

Principle 9448.6 (Finite-mass and massless boundary)**.** *For the massive relativistic multiplier* $H(k) = \sqrt{|k|^2 + m^2} - m$*, the quadratic row near* $k = 0$ *gives the same meeting law with the effective diffusion scale set by* m^{-1}*. At* $m = 0$*, the finite-variance metric row no longer exists, but the coherent family still carries the ballistic scaling*

$$h = \frac{1}{2T^2}$$

in the one-dimensional Cauchy row. The covariance geometry and the massless light-cone scaling therefore separate cleanly.

Principle 9448.7 (Numerical meeting witnesses)**.** *The numerical certificate contains four independent rows. A one-dimensional lattice chain gives*

$$h\,2Tg = 0.9844,\ 0.9958,\ 0.9991,\ 0.99991$$

at $T = 5, 20, 100, 1000$*. A two-dimensional anisotropic walk with off-diagonal diffusion gives componentwise ratios to* $(2T)^{-1}g^{-1}$ *in the range* 1.002–1.003 *at* $T = 400$*. A massive relativistic multiplier gives*

$$h\,2T/m = 1.153,\ 1.015,\ 1.0015$$

at $T = 5, 50, 500$*. The massless Cauchy row gives* $h\,2T^2 = 1.0000$*. Together these rows support the matrix-valued meeting read, the massive approach to the diffusive regime, and the massless scaling boundary.*

9448.2 Heat Kernel Identity and Curvature Read

Theorem 9448.8 (Heat-kernel QGT identity)**.** *Let $K_T(z, z')$ be the heat kernel of the reconstructed geometric Hamiltonian on the physical volume measure. For the coherent family of Definition 9448.2,*

$$h_{ij}(z, T) = \partial_{z^i}\partial_{z'^j} \log K_T(z, z')\big|_{z'=z}.$$

Proof. The overlap of the unnormalized coherent states is the heat kernel $K_T(z, z')$ up to factors separated in z and z'. In the QGT expression for normalized states, those separated factors cancel under the mixed second derivative, leaving the displayed identity. □

Corollary 9448.9 (Spectral-dimension merger)**.** *In a homogeneous setting,*

$$\operatorname{tr} h = -\partial_T \log K_T(z, z).$$

With the spectral-dimension convention $K_T(z, z) \asymp T^{-d_s(T)/2}$, the coherent-family trace reads

$$\operatorname{tr} h(T) = \frac{d_s(T)}{2T},$$

up to the derivative correction when d_s runs with T. For a smooth metric, the small-time expansion gives

$$d_s(T) = d - \frac{T}{3}R + O(T^2).$$

Thus the meeting row and the spectral-dimension row use the same heat kernel instrument.

Theorem 9448.10 (Curvature defect of meeting)**.** *For a smooth Riemannian metric G and small T,*

$$2T\, h_{ij}(z, T) = G_{ij}(z) - \frac{T}{3}\operatorname{Ric}_{ij}(z) + O(T^2).$$

In two dimensions, for a conformal scalar diffusion read,

$$2Tg\, h = 1 - \frac{R}{6}T + O(T^2).$$

Proof. Insert the Minakshisundaram–Pleijel heat-kernel expansion into Definition 9448.8. The mixed second derivative of the world function gives the metric term, and the mixed derivative of the Van Vleck–Morette factor gives the Ricci coefficient. □

Principle 9448.11 (Ricci as a measurable machine output)**.** *Ricci curvature is read as the first-order failure of exact meeting: the information metric and diffusion metric meet at leading order, and the first defect is $-\operatorname{Ric}/3$. This makes curvature a statistical output of the coherent-family row, not a primitive field-equation input.*

Principle 9448.12 (Curvature numerical witnesses)**.** *The curvature certificate contains a sphere witness and a machine-operator witness. On the sphere, the measured value*

$$\frac{2Th - 1}{T} = -0.33344$$

at $T = 0.005$ matches the predicted $-R/6 = -1/3$. For the conformal-torus machine operator of Definition 9448.1, the point $(0, 0)$ has $R = -1.1935$, predicted slope $+0.19891$, and corrected readings $+0.20203$ and $+0.20398$ at $T = 0.04$ and $T = 0.08$. The point $(\pi, 0)$ has $R = +0.5363$, predicted slope -0.08938, and corrected readings -0.09018 and -0.09033. The control point $(\pi/2, \pi/2)$ has $R = 0$ and corrected readings -0.00002 and -0.00009. The flat-control subtraction removes the lattice dispersion term proportional to dx^2/T^2; at the zero-curvature control, that term shrinks by the expected factor $1/4$ when T is doubled.

9448.3 Local Heat-Geometry Identity

Theorem 9448.13 (Local thermodynamic geometry identity)**.** *For the coherent-family position entropy and the diffusion metric trace, the small-time local relation is*

$$\dot{S} = \mathrm{tr}(h\,g) + \frac{R}{12} + O(T).$$

The coefficient $1/12$ *is dimension-independent in the stated heat-kernel normalization.*

Proof. In normal coordinates, the squared Van Vleck factor and the Riemannian volume density cancel to the required order. The entropy receives the volume-density correction, while $\mathrm{tr}(h\,g)$ receives the Ricci correction from Definition 9448.10. Subtracting the two expansions gives the displayed $R/12$ term. □

Corollary 9448.14 (Thermodynamic curvature read)**.** *The scalar curvature can be read by*

$$R = 12\big(\dot{S} - \mathrm{tr}(h\,g)\big) + O(T).$$

Thus the geometric curvature read of Definition 9448.10 has a second, thermodynamic measurement protocol.

Principle 9448.15 (Internal consistency test)**.** *The same scalar curvature must be obtained from the meeting-defect read and from the entropy-production read. This gives an internal consistency test for any proposed input-flow geometry bridge.*

Principle 9448.16 (Thermodynamic numerical witnesses)**.** *On the sphere, the measured differences*

$$\dot{S} - \mathrm{tr}(hg) = 0.16856,\ 0.16735,\ 0.16789$$

at $T = 0.02, 0.05, 0.10$ *match the prediction* $R/12 = 1/6$ *to the observed finite-difference accuracy. For the machine conformal-torus operator, flat-subtracted readings give* -0.10246 *at the negative-curvature point where* $R/12 = -0.09945$*, and* $+0.04535$ *at the positive-curvature point where* $R/12 = +0.04469$*. The heat-geometry identity therefore has both curvature signs under the same estimator.*

9448.4 Entanglement Entropy Row

Definition 9448.17 (Free-field reconstruction row)**.** *The free-field reconstruction row consumes the positive Hamiltonian multiplier of chapter 9447 and forms the Gaussian ground-state covariance matrices for a scalar field on a lattice regulator.*

Principle 9448.18 (Area law witness)**.** *For a two-dimensional spatial torus cut by a straight boundary, the Gaussian covariance calculation records an area law: the entropy per unit boundary length stabilizes as the system size increases. In the mass-*0.2 *row,*

$$\frac{S}{2L} = 0.062416,\ 0.062307,\ 0.062286,\ 0.062285,\ 0.062285$$

for $L = 24, 32, 48, 64, 96$*. The remaining size correction is below the displayed precision in this window.*

Principle 9448.19 (Universal massive slope)**.** *The small-mass area coefficient has the form*

$$\kappa(m) = \kappa_{\mathrm{UV}} - \frac{m}{12} + O(m^2)$$

in the measured free scalar row. The measured values

$$\kappa(0.05), \kappa(0.1), \kappa(0.2), \kappa(0.4), \kappa(0.8) = 0.0733,\ 0.0695,\ 0.0623,\ 0.0499,\ 0.0318$$

give a small-mass slope near -0.0816*, close to* $-1/12$*. The intercept* κ_{UV} *is regulator dependent, while the slope is the universal heat-kernel contribution of the free-field row.*

Theorem 9448.20 (Arithmetic blindness of vacuum entanglement)**.** *The free-field vacuum entanglement row is blind to the refinement factor* N *and to the prime content* S *when these enter only through an overall Hamiltonian scale.*

Proof. The ground state of a positive quadratic Hamiltonian is unchanged by multiplying the Hamiltonian by a positive scalar. The refinement factor enters the reconstructed Hamiltonian through the overall energy scale $1/\log N$ in this setting, so the vacuum covariance and its entanglement entropy are unchanged. □

Principle 9448.21 (Species boundary)**.** *The candidate identification*

$$\frac{1}{4G} \sim \frac{\kappa_{\mathrm{UV}}}{\varepsilon}$$

inherits the regulator and species dependence of the entanglement area coefficient. The machine therefore reproduces the known induced-gravity obstruction rather than bypassing it.

Remark 9448.22 (Two heat-kernel twelfths)**.** *The curvature correction* $+R/12$ *in Definition 9448.13 and the massive area-law slope* $-m/12$ *in Definition 9448.19 both arise from heat-kernel coefficient rows. The chapter records the resonance; it does not identify the two coefficients by a common theorem.*

9448.5 Modular Hamiltonian and Unruh Row

Definition 9448.23 (Gaussian modular Hamiltonian row)**.** *For a Gaussian state restricted to a region* A*, let*

$$X = \langle \varphi\varphi \rangle_A, \qquad P = \langle \pi\pi \rangle_A.$$

The modular Hamiltonian is the quadratic operator K *such that* $\rho_A \propto e^{-K}$*.*

Theorem 9448.24 (Gaussian modular Hamiltonian normal form)**.** *Let* $X_s = X^{1/2}$ *and diagonalize*

$$X_s P X_s = O\nu^2 O^{\mathsf{T}},$$

with symplectic eigenvalues $\nu_a \geq 1/2$*. Set*

$$\epsilon_a = \log \frac{\nu_a + 1/2}{\nu_a - 1/2}.$$

Then

$$K = \frac{1}{2}\pi^{\mathsf{T}} G_K \pi + \frac{1}{2}\varphi^{\mathsf{T}} F_K \varphi,$$

where

$$G_K = X_s O \operatorname{diag}\left(\frac{\epsilon_a}{\nu_a}\right) O^\mathsf{T} X_s$$

and

$$F_K = X_s^{-1} O \operatorname{diag}(\epsilon_a \nu_a) O^\mathsf{T} X_s^{-1}.$$

Proof. The canonical transformation $\Phi = O^\mathsf{T} X_s^{-1} \varphi$ and $\Pi = O^\mathsf{T} X_s \pi$ diagonalizes the covariance data. Each normal mode is a thermal oscillator with $\coth(\epsilon_a/2) = 2\nu_a$. Pulling the oscillator kernels back gives the displayed matrices. □

Principle 9448.25 (Bisognano–Wichmann read)**.** *For a continuum half-space, the modular Hamiltonian is*

$$K = 2\pi \int x\, T_{00}(x)\, dx.$$

The lattice calculation reads this weakly: smoothed row sums of the modular kernels approach the boost weight $2\pi x$ near the cut, while pointwise entries display nonlocal regulator spread.

Principle 9448.26 (Weak modular numerical witness)**.** *For a massive lattice scalar with $m = 0.15$ and interval length 40, high precision covariance diagonalization gives the following ratios to the Bisognano–Wichmann prediction. At sites $i = 0, 2, 4, 6, 10$, the G_K diagonal ratios are*

$$0.9986,\ 0.9944,\ 0.9708,\ 0.9373,\ 0.8567,$$

the G_K row-sum ratios are

$$0.9986,\ 0.9978,\ 0.9879,\ 0.9985,\ 1.062,$$

and the F_K row-sum ratios against $2\pi\tilde{x}\, m^2$ are

$$0.996,\ 1.018,\ 0.928,\ 0.926,\ 1.113.$$

Near the cut, the row-sum tests recover $2\pi\tilde{x}$ at percent scale, while the diagonal entries display the expected nonlocal broadening of the modular kernel. A larger interval in double precision gives ratios near 1 at the first two sites, but high precision is needed once the symplectic eigenvalues approach 1/2.

Principle 9448.27 (Unruh temperature boundary)**.** *The weak half-space modular row gives the Unruh temperature $1/(2\pi)$ in machine units. The finite lattice row has two controlled deviations: nonlocal broadening near the cut and exponential sensitivity to a second boundary when the region is a finite interval rather than a half-space.*

9448.6 Jacobson Dictionary

Principle 9448.28 (Thermodynamic gravity dictionary)**.** *The machine supplies the following dictionary:*

thermodynamic-gravity ingredient	*machine row*	*status*
metric	$h = (2T)^{-1} g^{-1}$	*meeting law*
Ricci curvature	$2Th = G - \frac{T}{3}\operatorname{Ric} + O(T^2)$	*curvature read*
entropy production	$\dot{S} = \operatorname{tr}(hg) + \frac{R}{12} + O(T)$	*heat row*
entanglement variation	$\kappa_{\mathrm{UV}} - m/12$	*area row*
Unruh temperature	$K = 2\pi$ *boost*	*modular row*
Newton coupling	$\kappa_{\mathrm{UV}}/\varepsilon$	*scheme dependent*
Clausius assembly	$\delta Q = T\delta S$	*requires local equilibrium*

Principle 9448.29 (What remains outside the certificate)**.** *The thermodynamic rows locate the ingredients of the Jacobson argument but leave two named obstructions outside the current certificate: the scheme/species dependence of the area coefficient and the external local equilibrium condition needed to assemble a field equation. Therefore this chapter is a boundary and dictionary, not an Einstein-equation proof.*

9448.7 Scope and Correspondence Ledger

Principle 9448.30 (Structural assumptions)**.** *The local-equilibrium condition needed for a Clausius assembly is external to the current input-flow certificate. The coherent family is canonical in the sense that the machine evolves its own localized states, but uniqueness among all possible coherent families is not claimed. The amplitude semigroup convention used here fixes the factor* $1/(2T)$*; density semigroup conventions shift this constant by the corresponding factor without changing the inverse-metric structure.*

Principle 9448.31 (Certificate scope)**.** *The meeting law is stated for translation-invariant finite-variance rows. The curvature and heat-geometry theorems are analytic small-time statements for smooth metrics, with numerical witnesses in dimension two. The entanglement row is a free scalar Gaussian reconstruction on a single lattice regulator. The modular row is a weak half-space test near the cut, not a pointwise equality of every lattice kernel entry.*

Principle 9448.32 (External correspondence ledger)**.** *The classical components used by the chapter include the Minakshisundaram–Pleijel and DeWitt–Synge heat-kernel expansions, de Bruijn-type entropy identities, Gaussian covariance methods for free fields, the Bisognano–Wichmann theorem, the Unruh temperature row, Jacobson's thermodynamic equation-of-state program, and Sakharov induced gravity. The machine contribution is the interface: the meeting law, the curvature and entropy measurement protocols, and the placement of the remaining obstructions inside the input-flow ledger.*

Principle 9448.33 (Open verification rows)**.** *chapter 9449 records the completed verification rows for the higher-dimensional Ricci read, regulator comparison of the universal massive slope, common heat-kernel slot of the two twelfth-coefficients, generic-point dual curvature instruments, homogeneous next-order trace channel, and lattice residual discrimination. chapter 9450 records the interaction and slot-response rows for the area law. The remaining rows outside those certificates are an input-flow definition of local equilibrium and a fully discrete Clausius–Raychaudhuri assembly.*

Chapter 9449

Innovation Input Thermodynamic Geometry Verification

chapter 9448 places the meeting law, curvature read, local entropy identity, entanglement area row, and modular-temperature row into one thermodynamic dictionary. This chapter records the adjacent verification certificate: the cone slot behind the two twelfth-coefficients, the agreement of the curvature instruments away from critical points, the scheme test for the massive area slope, the three-dimensional Ricci tensor read, the homogeneous next-order trace channel, the two-dimensional and three-dimensional tensor-coefficient reads, and the grid-scaling test which separates continuum terms from lattice residue.

9449.1 Cone Slot and the Two Twelfths

Theorem 9449.1 (Cone coefficient and curvature slot)**.** *Let*

$$s(\alpha) = \frac{1}{12}\left(\frac{2\pi}{\alpha} - \frac{\alpha}{2\pi}\right)$$

be the Sommerfeld cone coefficient. In the entanglement limit $\alpha = 2\pi - \delta$, its linear term is $\delta/(12\pi)$. The distributional cone curvature satisfies $\int R = 2\delta$, and the smooth DeWitt slot $a_1 = R/6$ contributes

$$\frac{1}{4\pi}\frac{2\delta}{6} = \frac{\delta}{12\pi}.$$

Thus the massive area-law coefficient $-1/12$ and the local heat-geometry correction $+R/12$ occupy the same linear heat-kernel slot.

Proof. The displayed expansion gives

$$s(2\pi - \delta) = \frac{\delta}{12\pi} + O(\delta^2).$$

The DeWitt contribution of a localized cone curvature is the second displayed expression. The two linear coefficients are identical. □

Corollary 9449.2 (Curvature-mass slot target)**.** *For minimal coupling, scalar curvature and mass-squared enter the same heat-kernel slot with opposite signs. The entanglement row therefore has the numerical target*

$$\frac{\partial S_{\mathrm{EE}}}{\partial R} = -\frac{1}{6}\frac{\partial S_{\mathrm{EE}}}{\partial m^2}$$

in a regulator comparison where both derivatives are measured in the same units.

Remark 9449.3 (Linearity scope)**.** *The identification above is a linear-slot statement. Entanglement entropy uses this linear cone term, while higher Renyi rows probe nonlinear cone dependence and need not share the same coefficient identity.*

9449.2 Dual Curvature Instruments

Definition 9449.4 (Two curvature instruments)**.** *For a two-dimensional conformal machine row, define the geometric and thermodynamic scalar-curvature reads by*

$$R_h = 6\left(\frac{1}{T} - \mathrm{tr}(hg)\right), \qquad R_S = 12\left(\frac{1}{T} - \dot{S}\right).$$

Both instruments use the same flat-control subtraction but otherwise read different data: R_h reads the meeting defect, while R_S reads entropy production.

Principle 9449.5 (Generic-point witness)**.** *Away from critical points of the conformal factor, the two instruments agree with the analytic curvature and with each other:*

point	R	T	R_h	R_S
$(1.060, 0.628)$	-0.3704	$0.04/0.07$	$-0.3737/-0.3744$	$-0.3724/-0.3741$
$(2.199, 1.100)$	$+0.1919$	$0.04/0.07$	$+0.1949/+0.1959$	$+0.1939/+0.1952$

The agreement with the analytic value is at the percent scale, and the two instruments agree with each other at approximately 0.3–0.5%*. This is the noncritical-point form of the internal consistency test in Definition 9448.15.*

Principle 9449.6 (Gradient robustness)**.** *The test points have nonzero conformal gradient. The log-Gram mixed derivative is a discretization of an exact heat-kernel identity, while the entropy read is scalar. The common curvature output is therefore not an artifact of placing the measurement at extrema of the conformal factor.*

9449.3 Regulator Scheme Comparison

Definition 9449.7 (Second lattice dispersion row)**.** *The comparison regulator uses the lattice dispersion*

$$f_B(k) = 4\sin^2\frac{k}{2} + \frac{1}{2}\left(4\sin^2\frac{k}{2}\right)^2.$$

It lies in the same long-wavelength regulator class as the nearest-neighbor row but changes the ultraviolet area coefficient.

Principle 9449.8 (Area coefficient scheme test)**.** *For the two regulator schemes, the fitted area rows are*

quantity	*scheme A*	*scheme B*	*read*
κ_{UV}	0.07732	0.11062	*scheme dependent*
mass slope	-0.08163	-0.08170	*universal*
difference slope	-0.0031	-0.0031	*common residue cancels*

The ultraviolet intercept moves by about 43%*, while the mass slope is stable across schemes to 7×10^{-5}. The induced-gravity row is therefore split into a regulator-dependent Newton coupling and a universal matter-response slope.*

Principle 9449.9 (Cross-scheme invariance test)**.** *The correct test for a universal quantity is invariance under a regulator scheme change, not absolute closeness to a continuum value in a single scheme. Cross-scheme differencing cancels the shared finite-mass lattice residue and makes the universal slope visible at much higher precision.*

9449.4 Three-Dimensional Ricci Tensor Read

Principle 9449.10 (Non-tensor defect as operator audit)**.** *If the measured meeting defect cannot be organized as Ricci curvature and instead contains bare Hessian terms of the conformal factor, the operator being measured is not the Laplace–Beltrami operator of the displayed metric. The meeting-defect protocol therefore audits the operator as well as the curvature: a non-tensor defect is a certificate of an incorrect discretized generator, not a modification of Definition 9448.10.*

Principle 9449.11 (Three-dimensional tensor witness)**.** *For the conformally flat metric $G = e^{2\varphi}\delta_3$ with*

$$\varphi = 0.3\cos x + 0.15\cos y$$

at the point (π, π), the Ricci eigenvalues are distinct. After using the reversible volume measure in the discrete operator, the directional meeting-defect reads are

axis	$-R^i{}_i/3$	$T = 0.06$ *read*	*relative residue at* $T = 0.06/0.10$
x	$+0.6149$	$+0.6476$	$+5.3\%/+7.4\%$
y	$+0.4919$	$+0.5092$	$+3.5\%/+4.3\%$
z	$+0.3689$	$+0.3673$	$-0.4\%/-2.4\%$

The measured ratio $1.763 : 1.386 : 1$ *follows the predicted* $5/3 : 4/3 : 1$ *tensor pattern. The residual is handled by the scaling test of section 9449.8.*

9449.5 Homogeneous Next-Order Trace Channel

Theorem 9449.12 (Homogeneous trace expansion)**.** *For a homogeneous smooth geometry,*

$$\operatorname{tr} h = \frac{d}{2T} - \frac{R}{6} - T\left[\frac{|\operatorname{Riem}|^2 - |\operatorname{Ric}|^2}{90} + \frac{\Box R}{15}\right] + O(T^2).$$

The R^2 term cancels between the logarithm and the derivative of the heat-kernel diagonal.

Proof. Use

$$\operatorname{tr} h = -\partial_T \log K_T(z, z)$$

and insert the standard a_2 coefficient of the heat-kernel diagonal. The logarithm subtracts the square of the a_1 term, eliminating the pure R^2 contribution and leaving the displayed combination. □

Principle 9449.13 (Weyl visibility order)**.** *The first-order meeting defect reads Ricci curvature. The homogeneous next-order trace row is the first place where the Riemann-square term can carry Weyl curvature. On conformally flat backgrounds the same term reduces to a Ricci functional, which explains why the earlier numerical rows do not isolate a vacuum-curvature degree of freedom.*

Principle 9449.14 (Sphere and entropy next-order witnesses)**.** *On the unit sphere, the trace-channel fit gives intercept* -0.333333 *and slope* -0.022219*, matching* $-1/3$ *and* $-1/45$*. The local entropy identity on the same geometry gives intercept* 0.166647 *against* $1/6$ *and a measured next-order slope* $+0.01207$*. The exact rational form of that entropy slope belongs to the full tensor coincidence-limit calculation.*

Principle 9449.15 (Full tensor coincidence-limit target)**.** *The remaining full-tensor coefficient is governed by*

$$[a_{1;ij'}] = \frac{1}{12}\nabla_i\nabla_j R - [a_{1;ij}].$$

The missing input is the coincidence limit $[a_{1;ij}]$*, available through the Christensen and Decanini–Folacci coefficient systems. Its numerical targets are the directional slopes in section 9449.8 and the entropy next-order slope above.*

9449.6 Two-Dimensional Tensor Coefficient Read

Definition 9449.16 (Two-dimensional next-order basis)**.** *In a two-dimensional conformal geometry, the next-order mixed tensor coefficient of the meeting read is expanded in the basis*

$$[a_{1;ij'}] = \alpha\nabla_i\nabla_j R + \beta\Box R\, G_{ij} + \gamma R^2 G_{ij}.$$

The coefficient γ *is fixed by the homogeneous sphere trace channel, while* α *and* β *require inhomogeneous measurements.*

Principle 9449.17 (Zero-background coefficient channel)**.** *At a point and component where* $G_{xy} = 0$ *and* $R_{xy} = 0$ *but* $(\nabla\nabla R)_{xy} \neq 0$*, the mixed read* $h_{xy}(T)$ *isolates the* α *coefficient. In the measured conformal row this channel gives*

$$\alpha = +0.0310(20).$$

The point is a zero-background channel: the metric and Ricci terms are silent in that component, so the target coefficient carries the signal.

Principle 9449.18 (Diagonal coefficient channel)**.** *The diagonal channel at the conformal critical point, after subtracting the matched flat control, gives*

$$\beta = -0.0089(10), \qquad \gamma = -\frac{1}{360}.$$

The value of γ *is the sphere-locked trace coefficient of Definition 9449.14; the value of* β *is read from the inhomogeneous diagonal residual.*

Principle 9449.19 (Overdetermined cancellation point)**.** *At the mixed test point where the three basis contributions nearly cancel, the coefficients above predict a next-order slope*

$$b = -0.0073,$$

while the direct read gives

$$b = -0.0090.$$

The absolute difference 1.7×10^{-3} *is at the displayed numerical noise scale. The cancellation point is therefore an overdetermined check on the two-dimensional coefficient row, not a fitting input.*

Principle 9449.20 (Built-in flat-control column)**.** *The coefficient of the fitted* $1/T$ *control column must vanish after the matched flat subtraction. A nonzero value at order one identifies a normalization or time-factor error in the estimator. The flat-control column is thus an internal audit row for the tensor-coefficient measurement.*

9449.7 Spherical Consistency and Three-Dimensional Tensor Read

Theorem 9449.21 (Spherical-family consistency constraints)**.** *In the full DeWitt next-order tensor ledger, comparison with the homogeneous trace identity on the family of standard spheres S^d imposes four dimension-independent coefficient relations. In the coefficient coordinates used for the tensor ledger these include*

$$c_5 = 0, \qquad c_7 = -c_4.$$

The same relations force the two-dimensional scalar contraction

$$\frac{1}{4}(c_6 + c_8) + \frac{1}{2}c_9 + c_{10} = -\frac{1}{360}.$$

Thus the two-dimensional coefficient γ in Definition 9449.16 is the spherical-family contraction of the full tensor ledger.

Proof. On a standard sphere, $\nabla R = 0$ and all derivative curvature terms vanish. The homogeneous trace expansion of Definition 9449.12 is then a polynomial identity in the dimension d, because the contractions $|\operatorname{Riem}|^2$ and $|\operatorname{Ric}|^2$ have explicit polynomial forms in d. Matching powers of d against the full tensor basis gives the displayed relations and the stated two-dimensional contraction. □

Theorem 9449.22 (Three-sphere zero trace channel)**.** *For the unit standard three-sphere,*

$$|\operatorname{Riem}|^2 = |\operatorname{Ric}|^2, \qquad \Box R = 0.$$

Hence the homogeneous next-order trace coefficient in Definition 9449.12 vanishes exactly.

Proof. For a unit standard sphere of dimension d,

$$|\operatorname{Riem}|^2 = 2d(d-1), \qquad |\operatorname{Ric}|^2 = d(d-1)^2.$$

These are equal at $d = 3$, and R is constant, so the $\Box R$ term also vanishes. □

Principle 9449.23 (Sphere comparison witness)**.** *The measured spherical trace row separates the zero and nonzero channels: the S^3 trace correction is zero to machine precision across the displayed time window, while the S^2 trace correction follows the coefficient $-T/45$. This comparison tests the $|\operatorname{Riem}|^2 - |\operatorname{Ric}|^2$ structure directly, because the two spherical geometries sit on opposite sides of the same homogeneous formula.*

Principle 9449.24 (Three-dimensional rank read)**.** *The three-dimensional directional-slope system has singular-value rank three on the measured conformally flat family. Its displayed null direction is proportional to*

$$(0, -2, -1, 2),$$

in the four-coordinate block that contains the Laplacian-Ricci and scalar Laplacian coefficients. Since this null direction has no c_2 component, the coefficient of $\Box R_{ij}$ is read independently of that nullspace.

Principle 9449.25 (Laplacian coefficient read)**.** *The three-dimensional coefficient read gives*

$$c_2 = -0.0172(94), \qquad c_3 = -0.0003(50).$$

It is compatible with the rational candidate

$$c_2 = -\frac{1}{60}, \qquad c_3 = 0.$$

The induced two-dimensional diagonal combination is then

$$\beta_{2\mathrm{d}} = c_3 + \frac{1}{2}c_2 = -\frac{1}{120},$$

which is consistent with the measured $\beta = -0.0089(10)$ *of Definition 9449.18.*

Principle 9449.26 (Three-dimensional consistency check)**.** *The directional target*

$$b = (+0.412, +0.172, -0.127)$$

is compatible with the spherical-family constraints, the two-dimensional coefficient row, and the three-dimensional rank read. The system therefore does not expose an inconsistency between the Ricci tensor witness and the next-order DeWitt tensor ledger.

Principle 9449.27 (Weyl-blind boundary)**.** *The remaining quadratic coefficients, including the combination involving* c_4*,* c_{10}*, and* $c_6 - c_8$*, are hidden on the present conformally flat three-dimensional family by a Weyl-blind null direction and a weak singular value. A complete ten-coefficient read therefore requires a background where the Weyl sector is visible, or a nonconformal three-dimensional metric with sharper directional-slope data.*

9449.8 Residual Discrimination by Grid Scaling

Principle 9449.28 (Scaling discriminator)**.** *In the three-dimensional tensor witness, narrow-window fits by* $A + BT$ *and by* $a/T + bT$ *are both plausible. The decisive discriminator is grid scaling. At* $n = 96$ *and* $T = 0.03$*,*

axis	*measured residue*	dx^2/T *prediction*	T^0 *prediction*
x	+0.0196	+0.0191	+0.0242
z	+0.0015	+0.0017	+0.0058

The dx^2/T *prediction matches both axes, while a continuum* T^0 *residue does not. The residual is therefore a lattice term, not a missing continuum correction to the Ricci theorem.*

Principle 9449.29 (Residual mechanism and tensor target)**.** *The residual amplitude is nearly isotropic,*

$$a \approx (4.6, 4.1, 3.7) \times 10^{-4} \approx 0.16\, Q\, dx^2.$$

It is attributed to the local mismatch between face conductance and point measure in a critical-point discretization. After removing the grid term, the remaining directional slopes have signs

$$b = (+0.412, +0.172, -0.127),$$

which form the numerical target for the full tensor next-order coincidence-limit formula.

Principle 9449.30 (Fit-form discipline)**.** *When two asymptotic fit forms are indistinguishable on a small time window, the fit is not itself evidence for either form. A scaling experiment in the regulator parameter is the required discriminator.*

Chapter 9450

Innovation Input Interaction and Slot-Response Boundary

chapter 9448 places the free-field entanglement area law inside the thermodynamic dictionary, while chapter 9449 records the heat-kernel and regulator checks around that dictionary. This chapter records the next boundary: weak interaction enters the entanglement area row first through mass renormalization, the genuinely non-Gaussian piece starts later and is small in the measured lattice window, and a naive curvature–mass slot identity fails once the response is decomposed into the relevant geometric channels.

9450.1 First Interaction Row

Theorem 9450.1 (First interaction through mass renormalization)**.** *For a scalar row with interaction $\lambda\varphi^4$, the entanglement entropy has the first-order form*

$$S_{\mathrm{EE}}(\lambda) = S_{\mathrm{free}}(m_*^2) + O(\lambda^2), \qquad m_*^2 = m^2 + \frac{\lambda}{2}\langle\varphi^2\rangle.$$

Proof. The entanglement first law gives

$$\delta S = \mathrm{tr}(\delta\rho\, K)$$

at first order. For the Gaussian modular Hamiltonian row, this reduces the first-order entropy response to the first-order shift of the two-point function. The tadpole term shifts the mass, while the four-particle transition does not contribute to the two-point matrix element at this order. Hence the free entropy evaluated at the shifted mass gives the displayed expression. □

Principle 9450.2 (Exact-diagonalization witness)**.** *In a four-site φ^4 chain with a truncated bosonic basis, the first-law check gives ratio* 1.00084*, the zero-coupling entropy agrees with the covariance method to* 1.1×10^{-7}*, and the measured derivative*

$$\left.\frac{dS}{d\lambda}\right|_{\lambda=0} = -0.166117$$

matches the mass-renormalization prediction -0.166344 *at the displayed precision. The convergence floor of the diagonalization row is below* 3×10^{-6}*.*

Corollary 9450.3 (Massive slope inherited by the interacting row)**.** *The universal massive area-law slope remains the free-field slope when expressed in terms of the renormalized mass m_*. The arithmetic blindness of vacuum entanglement also persists under a global Hamiltonian rescaling: if a refinement factor changes only the overall energy scale, the interacting ground-state density matrix on a region is unchanged.*

Principle 9450.4 (Critical boundary of the first-order row)**.** *The first-order formula contains the factor*

$$\frac{dm_*}{d\lambda} = \frac{\langle \varphi^2 \rangle}{4m_*}.$$

It therefore becomes singular near the massless critical row. The breakdown is not a failure of the first-order certificate; it locates where the interaction problem leaves the perturbative mass-shift regime.

9450.2 Non-Gaussian Remainder

Definition 9450.5 (Gaussian assembly comparison)**.** *Let S_{exact} be the exact reduced entropy of the interacting lattice state. Let S_{gauss} be the Gaussian entropy assembled from the exact two-point correlations, and let S_{Hartree} be the entropy obtained from a self-consistent mass approximation. These three rows separate correlation data from genuinely non-Gaussian entropy.*

Principle 9450.6 (Quadratic-order witness)**.** *The measured second-order row has stable λ^2 scaling:*

λ	Δ_2/λ^2	$S_{\text{exact}} - S_{\text{gauss}}$	$S_{\text{exact}} - S_{\text{Hartree}}$
0.05	$+0.052$	-9.8×10^{-6}	$+2.8 \times 10^{-5}$
0.10	$+0.047$	-3.5×10^{-5}	$+9.5 \times 10^{-5}$
0.40	$+0.028$	-3.6×10^{-4}	$+7.6 \times 10^{-4}$

The exact entropy stays below the Gaussian assembly entropy in this window, while the Hartree row underestimates the entropy.

Principle 9450.7 (Two-level interaction program)**.** *The measured non-Gaussian deficit is negative but accounts for only a small part of the displayed second-order effect. Most of the interaction response is captured by the two-point correlation row and a Gaussian assembly; the remaining tail belongs to a replica or higher-cumulant calculation. Thus the interaction extension splits into a two-level program: measure the correlation row accurately, then compute the residual non-Gaussian entropy.*

Principle 9450.8 (Hartree sign diagnostic)**.** *The Hartree approximation can have the wrong entropy sign even when it captures a plausible mass shift. In the measured row it underestimates the entropy because the missing sunset-type self-energy contribution makes the effective mass too large. This sign diagnostic is part of the interaction certificate, not a numerical nuisance.*

9450.3 Slot-Response Audit

Definition 9450.9 (Conformal response channels)**.** *A small conformal perturbation of the spatial metric decomposes the entanglement response into distinct channels: the coordinate mass channel, the measure-weight channel, and the geometric slot channel. A slot claim is valid only after these channels have been separated.*

Principle 9450.10 (Channel decomposition witness)**.** *For a perturbation amplitude* 0.06 *at mass* 0.4*, the total measured curvature response is*

$$-0.05766.$$

The coordinate mass channel contributes -0.05912*, while the measure-weight channel contributes* $+0.00156$*, closing the total to about* 0.2%*. The geometric slot channel contributes* $+0.00803$ *but is hidden under the larger coordinate mass response. The dominant term matches the estimate from* $\kappa'(m)m\varepsilon\psi(y_c)n$*.*

Principle 9450.11 (Response audit)**.** *A perturbation response must be decomposed by channel before it is assigned to a geometric invariant. If a claimed curvature slot is mostly a coordinate mass effect, the measurement protocol exposes the frame confusion. This response audit is the interaction analogue of the non-tensor operator audit in Definition 9449.10.*

Principle 9450.12 (Measured inverse-mass slot)**.** *A matched double-profile difference that fixes the cut value and slope shows an inverse-mass response. Across a mass range of about* 4.6*, the product* $mD(m)$ *remains on a platform between* 0.0028 *and* 0.0036*, giving*

$$c_{-1} = 0.0034(4).$$

Thus a covariant inverse-mass slot is present, but the coefficient is not the naive scalar-curvature coefficient.

Principle 9450.13 (Projected curvature coefficient)**.** *The naive coefficient predicted by a bare* $R/6$ *slot is about* 0.0080 *in the measured normalization and is excluded by the observed inverse-mass platform. A projected cone coefficient involving transverse curvature data, such as a combination of* R *and* R_{nn}*, gives a value near* 0.0040 *and is compatible at the current precision. The exact coefficient belongs to the conical heat-kernel projection row rather than to the scalar heat-kernel trace alone.*

Principle 9450.14 (Slot boundary)**.** *The heat-kernel slot identity for operator traces does not transfer unchanged to entanglement entropy. Entanglement inserts a cut surface and therefore a projection onto data normal and tangential to that surface. Distinguishing candidate projected coefficients requires a geometry where* R *and the normal Ricci component are not locked together, such as a higher-dimensional spatial row or a time-dependent setting.*

9450.4 Interaction Correspondence Ledger

Principle 9450.15 (Classical correspondence ledger)**.** *The external structures used by this chapter include the entanglement first law, tadpole mass renormalization, sunset self-energy corrections, Gaussian entropy assembly from covariance data, and conical heat-kernel projection formulae. The input-flow contribution is the placement of these structures inside the thermodynamic dictionary: first-order interaction enters through renormalized mass, the measured non-Gaussian tail is a residual row, and curvature slots must be audited by response channel.*

Chapter 9451

Innovation Input Solenoid and Arithmetic Boundary

chapter 9447 records the statistical and spectral rows of an input flow. This chapter records the arithmetic rows of a quantized refinement tower: solenoid structure, charge groups, kernel blindness, denominator clearing, and scale-level fingerprints. These rows are insensitive to mass and stable-tail parameters, because they are zero-order data of the refinement tower.

9451.1 Holonomy Tower

Definition 9451.1 (Refinement mode tower)**.** *A refinement mode tower consists of integers*

$$M_k = N_1 \cdots N_k, \qquad N_k \geq 2,$$

and cyclic mode groups $\mathbb{Z}/M_k\mathbb{Z}$. The coarse read from level k to level $k-1$ is decimation by N_k.

Lemma 9451.2 (Aliasing equation)**.** *Let the level-k mode be*

$$x \longmapsto e^{2\pi i m^{(k)} x/M_k}.$$

Under decimation $x = N_k y$, it reads on level $k-1$ as the mode $m^{(k)} \bmod M_{k-1}$. If

$$z_k := e^{2\pi i m^{(k)}/M_k},$$

then tower compatibility is exactly

$$z_k^{N_k} = z_{k-1}.$$

Proof. Substitution gives

$$e^{2\pi i m^{(k)} N_k y/M_k} = e^{2\pi i m^{(k)} y/M_{k-1}}.$$

The phase equation is the same congruence written multiplicatively. □

Definition 9451.3 (S-adic solenoid row)**.** *Let S be the set of primes that occur infinitely often among the refinement factors N_k. The completed phase tower is*

$$\Sigma_S := \varprojlim \left(\mathbb{T} \xleftarrow{z \mapsto z^{N_k}} \mathbb{T} \right).$$

Its visible circle projection is $\pi_S : \Sigma_S \to \mathbb{T}$, and its kernel is $K_S := \ker \pi_S$.

Theorem 9451.4 (Tower structure group)**.** *The compatible mode data of an infinite refinement tower are exactly points of Σ_S. A finite-depth tower has visible circle structure; the solenoid is the completed inter-level structure group.*

Proof. By Definition 9451.2, a compatible tower is precisely a sequence (z_k) satisfying the inverse-limit equations defining Σ_S. □

Theorem 9451.5 (Charge group)**.** *The Pontryagin dual of the completed tower is*

$$\widehat{\Sigma_S} = \varinjlim \left(\mathbb{Z} \xrightarrow{\times N_k} \mathbb{Z} \right) = \mathbb{Z}[S^{-1}] \subset \mathbb{Q}.$$

Consequently solenoid charges are exactly the rational charges whose denominators use primes from S.

Proof. Duality turns compact inverse limits into discrete direct limits. Repeated multiplication by the factors N_k in the direct system inverts exactly the primes that occur infinitely often. □

Theorem 9451.6 (No continuous splitting)**.** *The exact sequence*

$$0 \to K_S \to \Sigma_S \xrightarrow{\pi_S} \mathbb{T} \to 0$$

does not split continuously when S is nonempty.

Proof. A continuous splitting would dualize to a homomorphism $\mathbb{Z}[S^{-1}] \to \mathbb{Z}$ restricting to the identity on $\mathbb{Z}$. For $p \in S$, the element $1/p$ would have to map to an integer a with $pa = 1$, impossible in $\mathbb{Z}$. □

Theorem 9451.7 (Chern obstruction for a compatible circle tower)**.** *Let (P_k) be a tower of principal circle bundles over a fixed two-dimensional base, compatible with fibre maps*

$$z \longmapsto z^{N_k}.$$

Then

$$c_1(P_{k-1}) = N_k c_1(P_k).$$

If the tower has unbounded cumulative degree, all Chern classes in the compatible tower vanish.

Proof. The degree-N_k map on the circle multiplies the Euler class by N_k. Thus $c_1(P_0) = M_k c_1(P_k)$ for every k. Since $c_1(P_0)$ is an integer class and M_k is unbounded, divisibility by every M_k forces $c_1(P_0) = 0$, and then all later classes vanish as well. □

Theorem 9451.8 (Beat-frequency digit read)**.** *At level k, compare a lifted mode m with*

$$m' = m + jM_{k-1}.$$

The per-cell-step relative phase is

$$e^{2\pi i(m'-m)/M_k} = e^{2\pi i j/N_k}.$$

Thus adjacent-level interference reads the digit j mod N_k of the kernel coordinate.

Proof. Subtract the two mode phases and divide by the level-k denominator. The quotient M_{k-1}/M_k is $1/N_k$. □

9451.2 Solenoid Exclusion Rows

Theorem 9451.9 (Kernel blindness)**.** *If a visible observable factors through the circle projection,*

$$O = \overline{O} \circ \pi_S,$$

then

$$O(\tilde{t}k) = O(\tilde{t})$$

for every $k \in K_S$. Hence the kernel coordinate is invisible to any single-layer circle-visible read.

Proof. Since $\pi_S(\tilde{t}k) = \pi_S(\tilde{t})$, the equality follows from the factorization. □

Theorem 9451.10 (Profinite-smooth obstruction)**.** *The kernel K_S is totally disconnected. Therefore any continuous map from a connected smooth window into K_S is locally constant. Such a row has no local derivative contribution to the innovation one-form $\Theta = Q_{\mathcal{L}} D_\Phi B$.*

Proof. The image of a connected set is connected, and connected subsets of a totally disconnected group are points. A locally constant row has zero connection derivative on each connected window. □

Principle 9451.11 (Flat holonomy channel)**.** *If K_S affects a physical read, it must do so through a flat holonomy, local-system, charge, or inter-level beat-frequency row. It cannot enter a local smooth metric row merely by being present as a profinite fibre.*

Principle 9451.12 (Solenoid tower allows only flat topological charge)**.** *Definition 9451.7 is the topological companion of Definition 9424.15: exact local curvature rows and compatible infinite root towers both exclude a nonzero Chern class in their stated packet classes. Nonzero topological phase requires a different packet class, such as a winding ray family not absorbed into a compatible circle-bundle root tower.*

Theorem 9451.13 (Single-layer absorption)**.** *A flat K_S-local system observed only through a character $\chi_q : K_S \to U(1)$ exports the $U(1)$-valued representation*

$$\chi_q \circ \rho : \pi_1 \to U(1).$$

Any other group with the same character image reproduces the same single-layer holonomy phenomena. Thus the single-layer read can identify a charge image but cannot certify the full solenoid as necessary.

Proof. The observable holonomy is the composite character. Its values depend only on the image subgroup in $U(1)$ and on the displayed representation into that image. □

9451.3 Denominator Clearing and Shape Blindness

Definition 9451.14 (Shape read of a two-dimensional covariance packet)**.** *For a positive two-dimensional covariance matrix h, the shape read is the projective modulus*

$$\tau(h) := \frac{h_{12} + i\sqrt{\det h}}{h_{11}}.$$

Theorem 9451.15 (Denominator clearing)**.** *Consider a constant-modulus three-component covariance packet in which the covectors lie in an S-smooth denominator lattice $M^{-1}\mathbb{Z}^2$. Multiplying all covectors by the scalar M sends the packet to an integral covector packet and sends*

$$h \longmapsto M^2 h.$$

The shape read $\tau(h)$ is unchanged. Therefore the set of accessible shape reads does not determine S.

Proof. The covariance matrix is quadratic in the covectors, so scalar multiplication by M multiplies h by M^2. Both numerator and denominator in $\tau(h)$ are homogeneous of degree one in h, hence the ratio is unchanged. □

Principle 9451.16 (Where S can be visible)**.** *The prime set S is invisible to projective shape reads. It can be visible only in rows that retain denominator data: charge groups, flux lattices, absolute scale rows, tower roots, or beat-frequency digits.*

9451.4 Scale and Charge Fingerprints

Definition 9451.17 (Quantized flux setting)**.** *A quantized flux setting displays covectors*

$$\kappa_a \in M^{-1}\mathbb{Z}^d,$$

weights

$$w_a \in A^{-1}\mathbb{Z}_{\geq 0},$$

and a cell length $\ell = 2\pi/M'$, with M, M', and A using primes from S.

Theorem 9451.18 (Flux lattice)**.** *For variable-weight packets in the setting of Definition 9451.17, the mean covector*

$$\bar{\kappa} = \sum_a w_a \kappa_a$$

has cell differences in $(MA)^{-1}\mathbb{Z}^d$. The discrete circulation around one cell lies in

$$\Phi_0\mathbb{Z}, \qquad \Phi_0 = \frac{2\pi}{MM'A}.$$

Across the tower, the accessible flux group lies in $2\pi\mathbb{Z}[S^{-1}]$.

Proof. Differences of weights lie in $A^{-1}\mathbb{Z}$, while covectors lie in $M^{-1}\mathbb{Z}^d$. Multiplying by the cell length $2\pi/M'$ gives the displayed flux unit. □

Theorem 9451.19 (Charge lattice determines the prime set)**.** *For subsets S_1, S_2 of the primes,*

$$\mathbb{Z}[S_1^{-1}] = \mathbb{Z}[S_2^{-1}]$$

as subgroups of $\mathbb{Q}$ if and only if $S_1 = S_2$.

Proof. For a prime p, one has

$$p \in S \quad \Longleftrightarrow \quad \frac{1}{p} \in \mathbb{Z}[S^{-1}].$$

Thus the subgroup recovers exactly the primes inverted. □

Corollary 9451.20 (Flux-area lower bound)**.** *If a cell carries nonzero flux and its phase-curvature read is Ω^η, then the positive Gram constraint gives*

$$\mathrm{Area}_{\mathrm{cell}} \geq \frac{1}{2}|\Phi_{\mathrm{cell}}|.$$

In the quantized setting, any nonzero flux cell has area at least $\Phi_0/2$ in the corresponding units.

Proof. In two dimensions, Definition 9424.8 gives $\sqrt{\det h^\eta} \geq |\Omega^\eta_{12}|/2$. Multiplying by the cell area factor converts the curvature density into the cell flux. □

Principle 9451.21 (Area spectra are weaker than charge groups)**.** *An area value set may contain square-root and weight factors that obscure the prime set. The charge lattice $\mathbb{Z}[S^{-1}]$ is the exact fingerprint; an area spectrum is only a scale-level shadow unless the bridge proves a lossless conversion.*

9451.5 Layering Verdict

Principle 9451.22 (Three arithmetic layers)**.** *The arithmetic content of a quantized input tower appears in three layers:*

layer	*data*	*sensitivity*
metric-propagation	$h^\eta,\ g^{ij},\ H(k)$	m, α, C^{ij}
scale-charge	$\Phi_0,\ \mathbb{Z}[S^{-1}]$	$S, \varepsilon,$ *scale rows*
holonomy	$\Sigma_S,\ K_S,$ *beat digits*	$S,$ *tower compatibility*

The deepest arithmetic rows are blind to the mass and stable-tail deformations of chapter 9447.

Chapter 9452

Innovation Input Modular Arithmetic Boundary

chapter 9451 records the arithmetic carried by an S-adic refinement tower. This chapter records the modular extension of the same input-flow grammar: partition functions, complex multiplication, class-group shape registers, Hecke operators, modular forms, and the zero boundary. The content is an interface map. Classical number-theoretic structures are named as rows that the machine can consume; they are not claimed as consequences of the positive innovation Gram row alone.

9452.1 Thermodynamic Zeta Rows

Definition 9452.1 (State partition row)**.** *For a prime set S, let $\mathbb{N}_S$ be the monoid of positive integers whose prime factors lie in S. The state partition row is*

$$Z_{\text{state}}(\beta) := \sum_{A \in \mathbb{N}_S} A^{-\beta}.$$

Theorem 9452.2 (State partition is partial zeta)**.** *The state partition row is*

$$Z_{\text{state}}(\beta) = \prod_{p \in S} \frac{1}{1 - p^{-\beta}}.$$

When S is the set of all primes, it is the Riemann zeta function $\zeta(\beta)$ on its half-plane of convergence.

Proof. This is the Euler product obtained by unique prime factorization. □

Definition 9452.3 (History partition row)**.** *The history partition row sums over ordered refinement histories rather than only over terminal sizes:*

$$Z_{\text{hist}}(\beta) := \sum_{k \geq 0} \big(Z_{\text{state}}(\beta) - 1\big)^k.$$

Theorem 9452.4 (History critical row)**.** *Where the geometric series converges,*

$$Z_{\text{hist}}(\beta) = \frac{1}{2 - Z_{\text{state}}(\beta)}.$$

Its critical row is

$$Z_{\text{state}}(\beta) = 2.$$

For all primes this uses the Kalmár constant, and for $S = \{2\}$ it gives $\beta = 1$.

Proof. The empty history contributes 1, and a nonempty step contributes one factor from $\mathbb{N}_S \setminus \{1\}$. Summing ordered words gives the displayed geometric series. □

Principle 9452.5 (Bost–Connes interface)**.** *The state partition row has the same skeleton as the Bost–Connes partition function: a multiplicative refinement monoid acts on root and charge data. In the input-flow reading, a choice of compatible root chain is a tower-direction selection in the kernel-blind layer of Definition 9451.9.*

Principle 9452.6 (Thermal fingerprint levels)**.** *The full Dirichlet series $Z_{\text{state}}(\beta)$ determines S by its Euler factors. A single critical inverse temperature is only a coarse functional of S. Thus the thermodynamic row sits between the exact charge lattice of Definition 9451.19 and scalar scale reads: the curve is lossless, while a point on it is not.*

9452.2 Complex Refinement and Chirality

Definition 9452.7 (Gaussian refinement tower)**.** *A Gaussian refinement tower replaces $\mathbb{Z}$ by $\mathbb{Z}[i]$. Refinement factors are Gaussian integers α_k, cumulative factors are $M_k = \alpha_1 \cdots \alpha_k$, and mode data live in*

$$\mathbb{Z}[i]/M_k.$$

Theorem 9452.8 (Gaussian solenoid row)**.** *The compatibility equation for Gaussian refinement is*

$$\alpha_k \zeta_k = \zeta_{k-1}$$

on $\mathbb{C}/\mathbb{Z}[i]$. Its inverse limit is a two-dimensional solenoid, and its charge module is

$$\mathbb{Z}[i][\mathfrak{S}^{-1}],$$

where $\mathfrak{S}$ is the set of Gaussian primes occurring infinitely often in the tower.

Proof. This is the proof of Definition 9451.4 with $\mathbb{Z}$ replaced by $\mathbb{Z}[i]$ and multiplication by N_k replaced by multiplication by α_k. □

Theorem 9452.9 (Digit field and Fermat classification)**.** *For a Gaussian prime π, the digit alphabet*

$$\mathbb{Z}[i]/\pi$$

is a field. Rational primes $p \equiv 1 \pmod 4$ split and give $\mathbb{F}_p$ digit fields along the two conjugate Gaussian primes; the prime 2 ramifies; rational primes $q \equiv 3 \pmod 4$ are inert and give $\mathbb{F}_{q^2}$. Equivalently, a conformal index-p Gaussian refinement exists exactly when p is a sum of two squares.

Proof. This is the standard splitting law for $\mathbb{Z}[i]$, together with Fermat's two-square theorem. □

Definition 9452.10 (Chirality row)**.** *For a split prime $p = \pi\bar{\pi}$, the chirality row records the value of the quarter-turn i in the digit field:*

$$r_\pi := i \bmod \pi.$$

Then $r_\pi^2 = -1$, and conjugation sends r_π to $-r_\pi$.

Principle 9452.11 (Complex logarithmic ledger)**.** *The cumulative Gaussian refinement factor has a complex logarithmic read. Its real part is the ordinary logarithmic tick,*

$$T = 2\,\mathrm{Re}\log M_k,$$

while its imaginary part records the cumulative chirality angle modulo the unit group of $\mathbb{Z}[i]$*. Time is robust under complex conjugation; chirality changes sign.*

Principle 9452.12 (Chirality thermodynamics)**.** *Adding an angular chemical potential to the Gaussian refinement tower gives Hecke Grossencharacter partition rows. The untwisted row recovers*

$$\zeta_{\mathbb{Q}(i)}(\beta) = \zeta(\beta)L(\beta, \chi_4),$$

while nontrivial angular weights read chirality in the holonomy layer. The chemical potential is an added readout row, not an intrinsic force law.

Theorem 9452.13 (Reflection as Frobenius in the inert case)**.** *For* $q \equiv 3 \pmod 4$*, the digit field is* $\mathbb{F}_{q^2} = \mathbb{F}_q(i)$*, and Frobenius satisfies*

$$i^q = -i.$$

Thus complex conjugation on the digit field is the Frobenius automorphism $x \mapsto x^q$ *exactly in the inert case.*

Proof. Since $q \equiv 3 \pmod 4$, $(q-1)/2$ is odd, so $i^q = i(i^2)^{(q-1)/2} = -i$. □

9452.3 Class-Group Shape Registers

Definition 9452.14 (Complex-multiplication shape register)**.** *Let* K *be an imaginary quadratic field and* $\mathcal{O}$ *an order. A complex-multiplication shape register starts from a lattice represented by a fractional ideal* L *and refines by invertible ideals* $\mathfrak{a}$*, sending*

$$L \longmapsto \mathfrak{a}^{-1}L.$$

Theorem 9452.15 (Shape orbit is class-group walk)**.** *The similarity class of the refined lattice depends only on the ideal class. Along a tower,*

$$[L_k] = [\mathfrak{a}_1 \cdots \mathfrak{a}_k]^{-1}[L_0] \in \mathrm{Cl}(\mathcal{O}).$$

Thus shape blindness for all ideal refinements is equivalent to class number one.

Proof. Similarity classes of complex-multiplication lattices are ideal classes. Multiplying by the inverse ideals gives the displayed class-group walk. □

Principle 9452.16 (Heegner shape-blind list)**.** *For maximal imaginary quadratic orders, universal shape blindness occurs only in the class-number-one discriminants*

$$-3, -4, -7, -8, -11, -19, -43, -67, -163.$$

For nonmaximal orders the corresponding list has the standard finite extensions. Outside these exceptional loci, the class group is a visible shape register.

Example 9452.17 (Discriminant minus twenty)**.** *For discriminant* -20*, the class group has order* 2*. One ideal class therefore flips a two-state shape register, while principal refinements preserve the shape. The real logarithmic tick remains globally meaningful, but chirality acquires a class-group ambiguity.*

Example 9452.18 (Discriminant minus twenty-three)**.** *For discriminant* -23*, the class group has order* 3*. The three reduced forms may be represented by*

$$x^2 + xy + 6y^2, \qquad 2x^2 + xy + 3y^2, \qquad 2x^2 - xy + 3y^2.$$

Conjugation extends the register symmetry to a dihedral action of order 6*. The associated weight-one modular form may be read through the eta product*

$$\eta(z)\eta(23z),$$

whose coefficients distinguish principal split primes, nonprincipal split primes, and inert primes. This is the modular-form read of a nonabelian-looking shape register after adjoining reflection.

Principle 9452.19 (Eta-product coefficient audit)**.** *In the discriminant* -23 *row, the eta-product coefficients give a concrete audit surface: principal split primes, nonprincipal split primes, and inert primes map to the corresponding coefficient classes. Sample checks include small-prime signs at* 2*,* 3*,* 13*, and* 23*, together with a principal split read at* 59*. These checks belong to the modular-form interface, not to a congruence-class rule.*

Theorem 9452.20 (No finite congruence criterion for the nonabelian register)**.** *In a class-field situation whose Galois closure over* $\mathbb{Q}$ *is nonabelian, there is no modulus* m *and set of residue classes that decide the principal-prime condition for all unramified rational primes.*

Proof. A purely congruence criterion would make the splitting decision factor through an abelian cyclotomic extension. That would force the relevant Galois action to be abelian, contrary to the nonabelian closure. □

9452.4 Hecke Flow and Modular Spectra

Definition 9452.21 (Full-channel refinement)**.** *For a prime* p*, full-channel refinement of a lattice sums over all index-*p *sublattices. The channel set is naturally*

$$\mathbb{P}^1(\mathbb{F}_p),$$

with $p + 1$ *channels.*

Theorem 9452.22 (Hecke identity)**.** *The full-channel refinement operator is the Hecke operator:*

$$(T_p f)(L) = \sum_{[L:L']=p} f(L').$$

For distinct primes p *and* q*, these operators commute.*

Proof. This is the standard lattice definition of Hecke operators. Commutativity is the usual double-coset relation for coprime indices. □

Principle 9452.23 (Shape becomes state space)**.** *Once every index-p channel is allowed, shape is no longer merely an invariant to be tested for blindness. It becomes the state space*

$$\mathrm{SL}(2,\mathbb{Z})\backslash\mathbb{H}$$

on which the refinement operators act.

Theorem 9452.24 (Petersson measure as innovation geometry)**.** *For an equivariant projective Hilbert family over the upper half-plane, the positive innovation Gram tensor is* $\mathrm{SL}(2,\mathbb{R})$*-invariant. Therefore its real part is a scalar multiple of the hyperbolic metric and its imaginary part is a scalar multiple of the hyperbolic area form. The Petersson measure is the natural volume read of this projective innovation geometry.*

Proof. The upper half-plane has a unique invariant Riemannian metric and invariant area form up to scalar. Equivariance forces the Gram tensor's real and imaginary parts to lie in those one-dimensional invariant spaces. □

Principle 9452.25 (Weight as chirality)**.** *The circle action in the projective family grades states by chirality. Holomorphic modular forms of weight k are therefore read as bound states in a chirality-k sector of the full-channel refinement flow.*

Theorem 9452.26 (Ramanujan threshold as reflection-positive mixing)**.** *For the normalized prime-channel walk* $P = T_p/(p+1)$ *on the modular shape space, the Eisenstein spectrum contains*

$$\frac{2\sqrt{p}}{p+1}\cos(t\log p).$$

Negative spectral values force reflection positivity to fail for the pure walk. A lazy walk

$$P_c = (1-c)I + cP$$

is positive precisely when the negative spectrum is shifted into $[0,\infty)$*. The optimal threshold is governed by the sharp Hecke spectral bound; in the Ramanujan case it is*

$$c^*(p) = \frac{p+1}{(\sqrt{p}+1)^2}.$$

Proof. The displayed Eisenstein eigenvalue is the standard normalized T_p eigenvalue on the unitary axis. The spectrum of P_c is $1-c+c\lambda$, so positivity is exactly the lower-spectrum inequality. □

Principle 9452.27 (Dihedral boundary)**.** *Class-group and complex-multiplication rows account for the dihedral spectral pieces. Non-dihedral Maass and holomorphic eigenforms are the genuine delocalized modes of the full-channel refinement flow.*

Principle 9452.28 (Ramanujan status split)**.** *The reflection-positive threshold has several evidentiary forms across related rows. The Wirsing boundary threshold is explicit; the Ramanujan threshold is structurally exact but open in the Maass sector; and expander-style congruence gaps are established in broad settings while often not giving the exact constants needed for a sharp input-mixing row.*

9452.5 Zero Boundary

Theorem 9452.29 (Zeta as scattering matrix)**.** *For the modular surface, the Eisenstein constant term has the form*

$$y^s + \varphi(s)y^{1-s}, \qquad \varphi(s) = \frac{\Lambda(2s-1)}{\Lambda(2s)}.$$

Thus the completed zeta function enters the machine as a scattering coefficient at the cusp.

Proof. This is the classical constant-term formula for Eisenstein series on the modular surface. □

Principle 9452.30 (Zero location boundary)**.** *The input-flow grammar can place Riemann zeta zeros as scattering resonances of the cusp row. It does not thereby produce a Hilbert–Polya self-adjoint operator for those zeros. The self-adjoint spectral row belongs to the Laplacian and Selberg-type zeta functions; the Riemann zeta row enters through scattering.*

Principle 9452.31 (Three zero-boundary instruments)**.** *The zero boundary has three independent interface readings. The scattering row places zeta zeros in the Eisenstein coefficient. The generation row says that changing the finite input generator cannot bypass analytic continuation: the same meromorphic object is reached or not reached as a whole. The radial positivity row asks for a condition such as*

$$\operatorname{Re}\frac{\xi'}{\xi}(s) > 0$$

on the right half of the critical strip; classical criteria identify this as a Riemann-hypothesis-strength target, not as a closed input-flow theorem. The agreement of the three readings locates the wall; it does not remove it.

Principle 9452.32 (Positive target for a zero route)**.** *A Lee–Yang-style route to a zero statement would require a specified function class, closure of that class under the tower operations, inclusion of the base object, and convergence of the tower to the target function. The reflection-positive machinery supplies this pattern for transition spectra in the input-time variable. Transporting it to the scattering variable is an additional analytic bridge obligation.*

Chapter 9453

Innovation Input Boundary Dynamics and Numerical Rows

chapter 9452 places modular spectra and zeta scattering inside the input-flow interface. This chapter records the boundary dynamics that compute those spectra through continued fractions and transfer operators, together with the numerical congruence rows that measure finite-register coupling.

9453.1 Geodesic and Boundary Flow

Theorem 9453.1 (Quadratic discriminant trichotomy)**.** *For the modular surface, conjugacy classes split by the sign of*

$$D = t^2 - 4.$$

The cases $D < 0$, $D = 0$, and $D > 0$ correspond respectively to elliptic fixed rows, cusp scattering rows, and closed geodesic rows. In the $D > 0$ case, primitive closed geodesics correspond to narrow classes of indefinite binary quadratic forms, with length

$$2 \log \varepsilon_D^+.$$

Proof. This is the standard classification of elements of $\mathrm{SL}(2, \mathbb{Z})$ by trace. The length formula is the usual translation-length formula in terms of the fundamental totally positive unit. □

Principle 9453.2 (Golden orbit)**.** *The smallest hyperbolic trace is 3, giving $D = 5$ and fundamental unit φ^2. Thus the shortest closed geodesic is the golden quadratic orbit. In the input-flow reading, the all-one continued-fraction word is the minimal-action periodic boundary row.*

Theorem 9453.3 (Class number and regulator action)**.** *For positive discriminant D, the total length contribution of the narrow class layer is governed by*

$$h^+(D) \log \varepsilon_D^+ = \sqrt{D}\, L(1, \chi_D),$$

up to the conventional normalization of the class number formula. Thus the boundary clock for real quadratic data is the regulator.

Proof. This is the real quadratic class number formula, read as a sum of closed geodesic lengths. □

9453.2 Gauss Map and Mayer Operator

Definition 9453.4 (Gauss boundary input)**.** *The Gauss boundary input is the map*

$$G(x) = \{1/x\}$$

with inverse branches

$$x \longmapsto \frac{1}{x+n}, \qquad n \geq 1.$$

Its natural extension carries the past coordinate needed for reversible reflection.

Theorem 9453.5 (Geodesic flow as suspension)**.** *The modular geodesic flow is represented by a suspension of the natural extension of the Gauss map, with roof function proportional to*

$$\log(1/x)^{-1}.$$

The natural-extension density is symmetric under the exchange of present and past coordinates, giving the boundary form of time reflection.

Proof. This is the classical Series coding of geodesic flow by continued fractions. The symmetry of the natural-extension density is the reversibility row. □

Definition 9453.6 (Mayer transfer operator)**.** *The Mayer operator is*

$$(\mathcal{L}_s f)(x) = \sum_{n\geq 1} (x+n)^{-2s} f\left(\frac{1}{x+n}\right).$$

Theorem 9453.7 (Selberg zeta determinant row)**.** *For the modular surface, the Selberg zeta function is recovered from the Fredholm determinants of the Mayer operator by the standard even-odd factorization*

$$Z_{\text{Selberg}}(s) = \det(1-\mathcal{L}_s)\det(1+\mathcal{L}_s).$$

Proof. This is Mayer's determinant formula for the continued-fraction transfer operator. □

Principle 9453.8 (Gap ledger as natural extension)**.** *Choice dynamics discard a complement at each visible step unless the natural extension is carried. The discarded complement is precisely the past coordinate of the natural extension. Thus the gap-ledger row, the natural-extension row, and the reversibility requirement of reflection positivity are the same structural demand in three languages.*

9453.3 Boundary Reflection Positivity

Theorem 9453.9 (Wirsing obstruction)**.** *The Gauss–Kuzmin–Wirsing transfer operator has a negative secondary eigenvalue*

$$\lambda_2 = -0.3036630029\ldots.$$

Therefore the pure boundary input row fails reflection positivity. A lazy operator

$$P_c = (1-c)I + cP$$

is positive for

$$c \leq \frac{1}{1+|\lambda_2|}.$$

Proof. The spectrum of the lazy operator is $1 - c + c\lambda$. The negative eigenvalue is shifted to zero exactly at the displayed threshold. □

Principle 9453.10 (Two mixing thresholds)**.** *The modular shape flow has a Ramanujan-governed lazy threshold, while the continued-fraction boundary flow has a Wirsing-governed lazy threshold. Both are reflection-positive mixing thresholds; they live in different operator rows.*

9453.4 Bounded Type and Zaremba Boundary

Definition 9453.11 (Bounded alphabet boundary input)**.** *For $N \geq 1$, the bounded alphabet input restricts continued-fraction digits to*

$$1 \leq a \leq N.$$

The surviving boundary set is denoted E_N.

Theorem 9453.12 (Boundary dimension as pressure root)**.** *The Hausdorff dimension $\delta_N = \dim_H E_N$ is the unique value of s at which the restricted transfer operator has leading eigenvalue 1. Equivalently, it is the pressure-zero row of the finite-alphabet input.*

Proof. This is Bowen's formula for conformal expanding iterated-function systems. □

Principle 9453.13 (Small alphabet dimension reads)**.** *The finite-alphabet pressure row has benchmark values*

$$\delta_1 = 0, \qquad \delta_2 = 0.5312805\ldots, \qquad \delta_3 \approx 0.7057,$$
$$\delta_4 \approx 0.7889, \qquad \delta_5 \approx 0.8368, \qquad \delta_N \uparrow 1.$$

Hensley's asymptotic

$$1 - \delta_N \sim \frac{6}{\pi^2 N}$$

identifies the cost of deleting large digits.

Principle 9453.14 (Third dimension)**.** *The bounded alphabet row supplies a third dimension read:*

$$\textit{algebraic dimension}, \qquad \textit{spectral dimension}, \qquad \textit{boundary dimension } \delta_N.$$

The alphabet restriction prices the allowed input set rather than the kinematic rank or the stable-tail exponent.

Principle 9453.15 (Zaremba interface)**.** *Finite continued-fraction histories are rational endpoints. The question which denominators are reached by the alphabet $\{1, \ldots, N\}$ is the Zaremba problem. Bourgain–Kontorovich and Huang provide the density-one route at small fixed alphabets; the complete denominator-by-denominator statement remains a separate number-theoretic obligation.*

9453.5 Congruence-Coupled Measurement Rows

Definition 9453.16 (Congruence-coupled transfer row)**.** *For digit matrices*

$$\gamma_a = \begin{pmatrix} a & 1 \\ 1 & 0 \end{pmatrix}$$

and a congruence group G_q generated by their reductions modulo q, the coupled transfer row is

$$(\mathcal{L}_{s,N,q}F)(x, g) = \sum_{a \leq N} \frac{1}{(x+a)^{2s}} F\left(\frac{1}{x+a}, \gamma_a g\right).$$

Theorem 9453.17 (Mirror bipartition)**.** *Since* $\det \gamma_a = -1$*, the congruence walk is bipartite with respect to the determinant character. The mirror sector carries the eigenvalue* -1 *before passing to the double-step or mirror-quotient read.*

Proof. Each digit step changes determinant parity. The determinant character therefore changes sign under every generator, giving the eigenvalue -1 on that sector. □

Definition 9453.18 (Meeting number)**.** *For a finite alphabet and congruence modulus, the meeting number*

$$\Xi_N$$

is the dimensionless ratio between register mixing time and denominator expression time. In the measured row for alphabet $\{1, 2\}$ *it is read from the Lyapunov rate and the nonmirror spectral gap.*

Principle 9453.19 (Measured alphabet-two row)**.** *For alphabet* $\{1, 2\}$*, the boundary dimension row is*

$$\delta = 0.5312805063,$$

the Lyapunov row is approximately

$$\lambda = 1.2646,$$

and the nonmirror coupled congruence rows measured over small composite and prime-power moduli keep Ξ *in the displayed range*

$$5.9 \leq \Xi \leq 18.6.$$

The row is a numerical witness that congruence registers mix on the same logarithmic clock up to a moderate constant.

Principle 9453.20 (Measured congruence table)**.** *For the alphabet* $\{1, 2\}$ *measurement row, the nonmirror second spectral radius and meeting number are recorded as*

q	3	5	7	9	11	13	15	17	19	23	25	27
ρ_2	.7243	.8910	.7683	.8312	.8069	.8714	.9033	.8023	.8215	.8567	.8910	.8312
Ξ	5.9	16.4	7.2	10.3	8.8	13.8	18.6	8.6	9.6	12.3	16.4	10.3

This is a finite numerical certificate supporting the bounded-clock interpretation; it is not a uniform spectral-gap theorem.

Principle 9453.21 (Prime-power freezing witness)**.** *The measured rows for moduli* 5 *and* 25*, and for* 9 *and* 27*, exhibit the same slow spectral radius to the displayed precision. The corresponding slow mode at the higher prime power is the lift of the lower-level slow mode. This is a finite-register shadow of prime-power expansion stability, not a theorem about all moduli.*

Principle 9453.22 (Chinese-remainder penalty)**.** *Composite moduli can mix more slowly than each prime factor because the same digit drives all congruence coordinates at once. The measured modulus* 15 *row is the visible instance: the coupled tensor sector is slower than the maximum of the separate prime-factor reads, while still remaining in the same logarithmic clock scale.*

9453.6 Boundary Obligation Ledger

Principle 9453.23 (Boundary obligations)**.** *The boundary programme leaves distinct obligations:*

row	*obligation*
weighted congruence transfer	*uniform spectral gap*
bounded alphabet	*complete Zaremba coverage*
prime-power tower	*structural freezing criterion*
lazy parameter	*input-flow definition condition*
scattering zero route	*positive class closed under tower operations*

These obligations are not failures of the existing certificates. They mark the boundary between closed interface rows and open number-theoretic bridge rows.

Concretized in §7305: Observer experience traces are realized as finite observer-history, inter-history locality, perspective-classifier, transport, replay, provenance, and NameCert rows.

Chapter 9454

BEDC.MetaCIC is its own type theory: a framing principle

9454.1 The framing correction

A common misread of BEDC.MetaCIC's relation to classical CIC treats classical CIC as the reference object and BEDC.MetaCIC as a subtractive presentation of that reference. This inverts the relationship. BEDC.MetaCIC is a positively defined type theory with its own first-principles commitments. Classical CIC is a different type theory that shares some syntactic shape but adopts different design decisions.

The principle of this chapter is therefore simple:

BEDC.MetaCIC is its own theory, not a derivative of classical CIC.

The comparison with classical CIC is useful only after this identity has already been fixed. It is an analytical comparison between neighbouring theories, not a definition of BEDC.MetaCIC by reference to a parent calculus.

9454.2 What BEDC.MetaCIC positively is

BEDC.MetaCIC is determined by five first-principles commitments. These commitments give the substrate its identity directly.

9454.2.1 Closed generating types

The term-level and certificate-level objects of the substrate come from a finite collection of `inductive` declarations, including `Term`, `BMark`, `BHist`, and related closed carriers. Each declaration has a displayed constructor list. No constructor is added by axiom, ambient library supply, quotient identification, or unstated semantic interpretation.

The closed generator is the first unit of the theory. If an object is not produced by the displayed constructors, then it is not an internal object of that generator. External mathematical intentions may motivate the naming, but they do not enlarge the internal carrier.

9454.2.2 Literal AST type equality

Type equality on `Term`, and on any type-level object represented inside the substrate, is literal syntactic equality of the underlying AST. The substrate does not install a conversion rule that

promotes computational equivalence to typing identity. It does not treat two types as identical merely because an external reduction theory relates their encodings.

This point is central. Literal equality is not an impoverished conversion relation. It is the equality relation chosen by the theory. The resulting subject-reduction boundary of chapter 9377 is therefore an internal fact about the chosen substrate, not a defect measured against a different calculus.

9454.2.3 Structural recursors from closed generators

For each `inductive` declaration, the Lean kernel supplies the corresponding eliminator, such as `.rec`. These eliminators are the universal-induction primitives available for that closed generator. They authorize structural induction over exactly the displayed constructors.

BEDC.MetaCIC therefore has internal induction capacity, but that capacity is generator-local and structural. It is not a free principle over arbitrary semantic domains. When a claim needs an induction principle beyond the closed generator it names, that supply must be displayed as external input rather than silently imported into the substrate.

9454.2.4 Zero axiom, strict purity

The substrate does not include `Classical.choice`, `Quot.sound`, `propext`, or any other Lean stdlib axiom as a dependency of BEDC.MetaCIC theorems. A strict purity audit asks each accepted theorem for its transitive axiom dependencies. For theorem content belonging to the BEDC.MetaCIC substrate, `#print axioms` returns no axioms, and `axiom-purity -strict` enforces the same boundary at project scale.

This is not a rhetorical preference for fewer assumptions. It is a typing discipline for the project. If a result needs choice, extensionality, quotient soundness, or propositional extensionality, the need must appear as a named external route, a displayed setup field, or an explicit boundary. It is not part of the substrate by default.

9454.2.5 Apophatic external boundary

Anything beyond the preceding commitments is treated as external supply. The supply is named apophatically: the project states what is not internally derivable from the current substrate and exposes a socket for that input. It does not convert the socket into a hidden axiom.

The apophatic discipline of chapter 9400 is thus part of the identity of BEDC.MetaCIC. A named hypothesis, setup field, or GAP socket is not a disguised internal theorem. It is a visible boundary marker where the substrate stops and an external route may begin.

9454.3 The defining conjunction

The five commitments are not independent slogans. Together they define the theory:

$$\begin{aligned}\text{BEDC.MetaCIC} = {} & \text{closed generators}\\ & + \text{literal AST equality}\\ & + \text{structural recursors}\\ & + \text{zero-axiom strict purity}\\ & + \text{apophatic external boundary.}\end{aligned}$$

This is a positive definition. The theory is not identified by taking a different calculus and removing a feature from it. It is identified by the commitments it actually makes.

Consequently, the first question to ask of a BEDC.MetaCIC theorem is not which classical theorem it resembles. The first question is which closed generators, recursors, literal equalities, and named boundaries its proof uses. Only after that internal reading is fixed does a comparison with classical CIC become meaningful.

9454.4 Why classical CIC is a different theory

Classical CIC has its own first-principles commitments. In familiar presentations, a conversion rule permits typing to move across computational equality:

$$T_1 \equiv_\beta T_2 \quad \Longrightarrow \quad (t : T_1) \Leftrightarrow (t : T_2).$$

This rule is a type-level identification principle. It changes how typing judgments behave and gives subject-reduction arguments access to a kind of equality that BEDC.MetaCIC does not install.

Many classical CIC environments also provide, or commonly sit beside, extensional and classical principles such as propositional extensionality, function extensionality, quotient soundness, and classical choice. Different systems choose different packages of this supply. The shared point is that the resulting environment is organized around a broader practical proof surface than BEDC.MetaCIC's closed audit substrate.

The design objectives also differ. Classical CIC is built to support a large mechanized mathematics environment. BEDC.MetaCIC is built to delimit the smallest audit-clean closed-observational substrate on which its meta-theory can run. These objectives produce neighbouring theories with different commitments. Neither one is the technical parent of the other.

9454.5 What comparison is good for

Comparison with classical CIC is useful when it isolates a specific design decision and its consequences. The axiom dependency map of chapter 9455 is one such comparison. It asks how a classical-looking universal claim meets the BEDC.MetaCIC substrate and records whether the result is an unconditional proof, a restricted proof, or an obstruction with named supply.

In that role, comparison is secondary and diagnostic. It helps readers see why conversion, extensionality, quotient soundness, or choice matter for particular proof routes. It does not define BEDC.MetaCIC's identity. The theory exists on its own footing, and its theorems are first read as theorems of that theory.

9454.6 What comparison is not good for

Subtractive slogans about BEDC.MetaCIC are structurally misleading. They imply that classical CIC is the reference and that BEDC.MetaCIC is valuable only as a depleted copy. That implication is false. Both theories have first-principles definitions, and neither is the other's reference point.

They also suggest that BEDC.MetaCIC is missing something required for its own identity. That is false as well. BEDC.MetaCIC has exactly the commitments that define it: closed generators, literal AST equality, structural recursors, strict purity, and visible external boundaries. A principle present in another calculus is not an absence inside BEDC.MetaCIC unless BEDC.MetaCIC itself claims that principle.

Finally, they misread obstruction theorems. The subject-reduction obstruction of chapter 9377 is not a failure relative to an external yardstick. It is a positive theorem about what happens when dependent codomains are read through literal AST equality on the BEDC.MetaCIC substrate. Classical CIC avoids that obstruction by making different type-theoretic commitments.

9454.7 The label classical CIC

The label "classical CIC" is itself coarse-grained. Coq libraries, Lean libraries, Agda libraries, and other mechanized settings make different decisions about axioms, quotients, extensionality, universe features, and library conventions. BEDC.MetaCIC does not target any one of these environments as its origin.

The target is instead a specific positive substrate. Classical CIC can serve as a comparison family, but not as a unitary reference object from which BEDC.MetaCIC receives its meaning. Treating that family label as the definition of BEDC.MetaCIC is a category error.

9454.8 Implications for related chapters

The audit map family of chapter 9406, the obstruction analysis of chapter 9377, the discharge routes of chapter 9374, the axiom dependency map of chapter 9455, and the closed-normal consistency result of chapter 9376 are all to be read as internal-to-BEDC statements first.

They describe what BEDC.MetaCIC does on its own substrate. References to classical CIC in those chapters are analytical comparisons. They are not definitional references, and they do not turn BEDC.MetaCIC into a subtheory relation with classical CIC.

9454.9 Apophatic framing of this principle

The apophatic framing is internally consistent with this principle. BEDC.MetaCIC names what is not derivable from its own substrate. It does not name what another calculus possesses and then define itself by lack.

This distinction matters for the reading of every named boundary. A GAP socket, setup field, or discharge obligation is a statement about the current BEDC.MetaCIC substrate. It says where the internal proof route stops. It is not a confession that the theory ought to have been a different theory.

9454.10 Boundary

This chapter is a framing principle, not a new proof and not a new Lean target. It declares how to read BEDC.MetaCIC: as its own type theory, defined from first principles. Other chapters may cite this principle when their comparison language needs to keep BEDC.MetaCIC's identity separate from the identity of classical CIC.

Concretized in §855: BEDC theory identity is realized as a finite NameCert packet for closed generators, literal AST equality, structural recursors, strict purity, and apophatic boundary rows.

Concretized in §865: The five positive BEDC.MetaCIC commitments are packaged as a finite InternalTheoryBoundaryUp certificate.

Concretized in §856: The MetaCIC theory-identity packet records the five positive commitments as finite NameCert rows.

Concretized in §869: The five positive commitments are packaged as a finite BEDC self-substrate certificate.

Concretized in §867: The positive identity of BEDC.MetaCIC is realized as a finite NameCert packet over closed generators, literal AST equality, structural recursors, strict purity, and visible external-boundary rows.

Concretized in §871: The self-theory vision is realized as a finite classifier packet for closed generators, literal equality, recursors, purity, and apophatic boundary rows.

Concretized in §872: Finite loop packet for self-theory identity, audit replay, and boundary non-escape.

Concretized in §873: The literal AST equality commitment is packaged as a finite boundary certificate.

Chapter 9455

BEDC as an axiom dependency map of classical CIC

9455.1 The reframing

The relationship between BEDC formalisation and classical CIC formalisation is often misread as a strength comparison: BEDC uses fewer axioms, proves fewer universal theorems, and is therefore a weaker calculus. That framing misses the positive deliverable. BEDC is not a weakened copy of classical CIC. BEDC is a mechanical dependency map from classical CIC universal claims to the axiom supply on which those claims rely.

Classical CIC presents its universal claims after conversion, extensionality, quotient soundness, and classical selection have already been made available as part of the ambient proof environment. BEDC.MetaCIC starts from the opposite end. It keeps the substrate closed, first-principles, and axiom-free, then asks each classical-looking claim what happens when the hidden supply is removed.

The result is not merely a shorter theorem list. It is a different artifact:

$$\phi \mapsto (\text{Mode}, \text{Witness}, \text{Required-Axiom-Set}).$$

Here ϕ is a universal claim one would expect from a classical CIC development. The output records whether the claim is internally provable, internally provable only on a restricted surface, or blocked by a mechanical witness that names the missing axiom-supplied content.

This map is the positive contribution. BEDC does not claim that every classical theorem can be replaced by a BEDC theorem. It claims that every contact between a classical universal claim and the BEDC substrate can expose the exact axiom budget required for that claim to pass.

9455.2 Three modes of engagement

For a classical CIC universal claim ϕ, BEDC.MetaCIC has three relevant outcomes.

9455.2.1 Mode A: completely provable

Mode A means that ϕ is unconditionally provable on the zero-axiom substrate. Its required axiom set is empty, and its witness is the checked proof term itself:

$$\phi \mapsto (\text{A}, \text{unconditional proof}, \emptyset).$$

The closed-syntax algebra gives representative examples. Theorems such as `shift_closed`, `substitute_closed`, `betaStep_preserves_closed`, and `no_closed_normal_proof_of_false` are not weakened classical statements. They are the fragment of the classical-looking MetaCIC story that needs no extra axiom supply.

Mode A is therefore an isolation result. It says exactly which part of the classical CIC meta-theory is already present in the closed substrate. The important point is not only that the theorem exists, but that the theorem survives the strict axiom-purity gate. BEDC turns the absence of hidden supply into visible information.

9455.2.2 Mode B: partial discharge

Mode B means that ϕ is too broad as stated, but a restricted form is provable without importing the missing axiom. Its witness is the checked restricted theorem, and its required supply is trivial only on that restricted surface:

$$\phi \mapsto (\mathrm{B}, \text{restricted proof}, \text{restricted-surface condition}).$$

The subject-reduction application case supplies the sharp example. The theorem `subject_reduction_closed_` proves the closed application preservation statement when the codomain is closed independently of the argument. On that surface, no conversion oracle is needed to identify the result type before and after the argument step: closed substitution reads back the same codomain.

Mode B is more informative than a binary failure. It says that the classical universal claim contains an internally clean subclaim. BEDC does not merely report that the full statement needs help; it separates the region where literal syntax, closedness, and the represented reduction relation already suffice.

9455.2.3 Mode C: counterexample plus named obligation

Mode C means that the unrestricted ϕ is blocked in the present substrate by a mechanical witness. The output is not silence and not an informal caveat. It is a counterexample, together with named obligation sockets for the axiom-supplied content that classical CIC would use:

$$\phi \mapsto (\mathrm{C}, \text{counterexample plus named socket}, \text{specific axiom rows}).$$

The same subject-reduction application case gives the canonical witness. The theorem `subject_reduction_clo` exhibits a closed function, a closed reducing argument, a closed application step, and an original type that the reduct does not keep. The obstruction is not a vague warning about dependent types. It is a checked dependent-codomain counterexample.

The corresponding named sockets are the four subject-reduction obligations `BetaSubstitutionPreservation`, `AppArgTypeStable`, `LamDomainSubjectReduction`, and `PiDomainSubjectReduction`. They mark the precise pieces of axiom-supplied reasoning required before the parameterised subject-reduction route can be read as an unparameterised closed theorem.

Mode C is the place where BEDC's negative-looking material becomes positive data. A documented obstruction is a dependency witness. A named hypothesis is a dependency socket. A counterexample is a mechanical certificate that the universal claim cannot be treated as substrate-internal without extra supply.

9455.3 The mapping is the deliverable

The deliverable is the map, not a competition over theorem count. For each classical CIC universal claim ϕ, BEDC asks for the dependency tuple:

$$\phi \mapsto (\text{Mode}, \text{Witness}, \text{Required-Axiom-Set}).$$

The three modes give the interpretation.

Mode A. The required axiom set is empty, and the witness is an unconditional checked proof.

Mode B. The required axiom set is empty on a stated restricted surface, and the witness is a checked restricted proof.

Mode C. The required axiom set is specific, and the witness is a counterexample together with named hypothesis sockets.

This tuple is exactly what ordinary classical CIC developments do not usually expose as their primary object. They prove the theorem in a powerful ambient environment. BEDC records what the theorem costs when the ambient environment is not silently available.

The audit-map family of chapter 9406 is the directory form of this deliverable. Each audit map records established content, conditional content, obstruction content, frontier content, and cross-module use. In the present framing, those columns are read as an axiom dependency ledger: which claims need no supply, which claims need a restricted surface, which claims expose an obstruction, and which named socket would have to be filled to move the claim.

9455.4 How to read a dependency tuple

A dependency tuple is not a semantic judgement about the intended mathematics of ϕ. It is a substrate judgement about the proof route available to the present formal object. The tuple records what the kernel can check from the displayed constructors, definitions, and named fields that are actually present.

The three coordinates have separate jobs.

Mode. The mode says how the claim meets the substrate: internal proof, restricted internal proof, or obstruction with named supply.

Witness. The witness is the mechanical object that makes the mode auditable: a proof term, a restricted theorem, a counterexample, a named obligation, or a purity-gate refusal.

Required-Axiom-Set. The required set names the supply that would have to be added or discharged before the broader classical reading could be used without qualification.

The tuple is deliberately local. A Mode A result in one module does not upgrade a neighbouring Mode C claim. A Mode B result does not license the unrestricted theorem. A Mode C counterexample does not say that the classical claim is meaningless; it says that the claim is not substrate-internal without the named supply.

This locality is what makes the map useful. Classical presentations often compress many local dependencies into one ambient theorem. BEDC expands the theorem back into its local dependency structure. The result is a readable axiom budget rather than a yes-or-no theorem status.

9455.5 A worked entry: closed application subject reduction

Closed application subject reduction gives a compact instance of the whole method. The classical-looking claim is that if a closed application is typed and the application reduces, then the reduct keeps the original displayed type. BEDC.MetaCIC does not answer this claim with a single status. It splits the claim into the three dependency-map rows that the substrate can actually justify.

First, the closed syntax and reduction infrastructure around the claim has Mode A content. Closedness is preserved by the syntax operations that are already in the substrate. The relevant shift, substitution, and beta-closure facts are checked without conversion, extensionality, or classical choice. This records the zero-supply syntactic core of the classical preservation story.

Second, the independent-codomain application surface has Mode B content. When the codomain is closed apart from the argument, replacing the argument by its reduct does not change the displayed result type. The restricted theorem `subject_reduction_closed_app_independent_codomain` is the mechanical witness for that row. It says exactly where closed substitution is enough and where no conversion oracle is being used.

Third, the dependent-codomain surface has Mode C content. The theorem `subject_reduction_closed_app_c` supplies the mechanical counterexample: a closed application can step to a reduct that does not retain the original displayed type. The obstruction is therefore a positive dependency record. It points to the missing codomain-stability and typing-preservation supply that a classical subject-reduction proof normally has available through conversion and related typing infrastructure.

The tuple for the unrestricted closed application claim is therefore not "failed". It is:

$$\phi_{\mathrm{appSR}} \mapsto (\mathrm{C}, \text{dependent-codomain counterexample plus four sockets}, \text{subject-reduction supply}).$$

The companion Mode B row records the strongest zero-supply fragment:

$$\phi_{\mathrm{appSR}}^{\mathrm{independent}} \mapsto (\mathrm{B}, \text{independent-codomain proof}, \text{closed codomain condition}).$$

Together these rows are more informative than either a bare theorem or a bare failure. They show where the classical claim is already internal, where it is blocked, and which supply must enter to remove the block.

9455.6 Why this is more informative than classical CIC formalisation

Concretized in §862: finite dependency tuple packet for the axiom-map reading

Concretized in §864: Restricted-surface budgets realize Mode-B dependency-map rows as finite claim, restricted-surface, witness, required-supply, replay, and non-export NameCert data.

Classical CIC formalisation projects prove many theorems inside an environment where conversion and library-level extensional principles are already available. In Lean-based developments, tools such as `#print axioms` can inspect the transitive constants of a theorem after the fact. That inspection is valuable, but it is not the organising principle of the project.

BEDC reverses the organisation. The substrate begins with no project axioms in `lean4/BEDC/`, forbids `axiom` declarations, and audits transitive dependencies through `axiom-purity -strict`. The map is therefore first-class. A theorem that survives the gate records a zero-supply region. A theorem that needs a named hypothesis records a socket. A counterexample records the mechanical failure of the universal claim under the current substrate.

The difference is methodological. A classical development usually asks:

Can the theorem be proved in the ambient calculus?

BEDC asks:

Which exact supply is needed for the theorem to pass?

The second question produces strictly more dependency information. It does not replace the classical theorem; it explains its axiom budget.

9455.7 The mapped axiom families

The current map covers several families of supply. The point of the list is not to repeat the detailed chapters, but to state how each family appears as dependency-map data.

9455.7.1 Conversion and definitional equality

Conversion is the central typing supply of classical CIC. BEDC.MetaCIC does not install it as an ambient rule that can silently identify types after reduction.

The Mode A region consists of literal-AST and closedness facts where no conversion is needed. Closed shift, closed substitution, closure preservation for beta steps, and closed-normal contradiction exclusion belong to this region.

The Mode B region is visible in `subject_reduction_closed_app_independent_codomain`. When the codomain is closed independently of the argument, substitution after the argument step reads back the same type by closedness alone.

The Mode C region is visible in `subject_reduction_closed_app_counterexample`. A dependent codomain can observe the argument strongly enough that the reduct does not keep the original displayed type. The four named subject-reduction obligations in chapter 9377 are the corresponding sockets.

9455.7.2 Extensional identification

Propositional extensionality, function extensionality, and quotient soundness form an extensional identification family. chapter 9394 classifies this as a supply type, and chapter 9387 reads `Quot.sound` and `propext` through meta-closure proxy shapes at different strata.

BEDC's strict purity gate refuses `propext` and `Quot.sound` as transitive dependencies of checked BEDC theorems. The dependency-map reading is direct: whenever a classical proof needs to identify propositions, functions, or quotient representatives by extensional equality, the required supply is not hidden in the substrate. It must appear as an explicit transport, classifier, or named socket.

9455.7.3 Choice and unwitnessed existence

`Classical.choice` supplies a selected witness from an existence claim without exposing the witness as generated substrate data. chapter 9387 classifies this as a term-stratum meta-closure reading.

The BEDC map treats choice-dependent claims as claims with an unwitnessed-selection dependency. Finite, countable, or explicitly recursive witness routes may become Mode A or Mode B content when the witness is generated. A free classical selection over an arbitrary family becomes Mode C content unless the selection is replaced by displayed witness data.

9455.7.4 Mathematical induction as transcendental input

chapter 9393 treats free-floating universal induction over not-yet-observed cases as a transcendental input rather than as internally generated finite observation. The dependency-map reading is again explicit. A concrete finite trace is substrate data. A universal claim over an open-ended domain requires a supply socket unless the relevant induction principle is itself present as an authorised generator for the carrier being used.

This does not make all induction unavailable. It distinguishes generated recursors from untracked universal closure. The map records which generator authorises the proof and which universal use would require supply beyond the displayed substrate.

9455.8 Permanent and open-ended

BEDC.MetaCIC has a fixed substrate discipline: closed CIC syntax, first-principles definitions, no project axioms, and strict purity auditing. Classical CIC has infinitely many universal claims. Therefore the dependency map is permanent and open-ended. Each time a classical claim touches the BEDC substrate, the project can record one more tuple:

$$\phi \mapsto (\text{Mode}, \text{Witness}, \text{Required-Axiom-Set}).$$

This is not a weakness caused by an unfinished theorem list. It is the point of the artifact. A classical formalisation can keep collecting theorems. BEDC can keep cataloguing the axiom budget of those theorems at fine granularity.

The map is also cumulative. Mode A entries expand the zero-supply fragment. Mode B entries sharpen the surfaces where classical claims become internal. Mode C entries add mechanical witnesses for the exact place where an axiom would have to enter. Every entry improves the catalogue even when it does not close the corresponding classical theorem.

9455.9 Cross-reference

The present chapter is a framing chapter. The detailed dependency-map entries live in neighbouring chapters.

- chapter 9377 gives the concrete Mode B and Mode C entry for closed application subject reduction: independent codomain discharge, dependent-codomain counterexample, and four named obligations.
- chapter 9406 gives the paper methodology by which established, conditional, obstruction, frontier, and consumer rows are catalogued.
- chapter 9400 gives the epistemic discipline behind named obligations: an open socket is named rather than hidden.
- chapter 9387 gives the axiom-family classification for choice, quotient soundness, and propositional extensionality.
- chapter 9393 gives the corresponding reading for universal induction and finite observation.

The chapter therefore does not duplicate the audit-map chapters or the synthesis chapter. It states the positive value proposition that makes those chapters cohere: BEDC is an axiom dependency map of classical CIC.

9455.10 Boundary

BEDC does not claim to enumerate all dependency tuples for classical CIC. The map is permanent and open-ended because the space of classical universal claims is open-ended.

BEDC also does not claim to replace classical CIC formalisation. Classical developments prove theorems in their chosen ambient systems. BEDC complements them by exposing which parts of those theorems survive without hidden supply, which parts survive only on restricted surfaces, and which parts mechanically witness an axiom dependency.

The boundary is strict. The words *axiom dependency* and *mechanical witness* are not metaphors. A BEDC documented obstruction, counterexample, named hypothesis, purity-gate refusal, or audit-map frontier is an inspectable dependency record. It is the map itself.

Concretized in §861: Axiom-dependency tuples are realized as finite Mode, Witness, Required-Axiom-Set, transport, provenance, and NameCert rows.

Concretized in §862: The dependency tuple becomes a finite NameCert packet over claim, mode, witness, required-supply, replay, and non-escape rows.

Concretized in §863: The axiom dependency map is realized as a finite Mode/Witness/Required-Axiom-Set ledger packet.

Concretized in §868: finite dependency-map packet for mode, witness, and required-axiom rows

Concretized in §992: Axiom requirement ledgers realize the Required-Axiom-Set row as finite witness, refusal, transport, route, provenance, and NameCert data.

Chapter 9456

The Legitimate-Claim Problem for Finite Closed Systems

BEDC is often encountered through particular targets: a concrete NameCert construction, a MetaCIC audit map, a halting boundary, or the Riemann Hypothesis as a constructive witness target. The common problem behind these examples is more general. Given a finite closed system $\mathcal{C}$ and a finite expression E, what may $\mathcal{C}$ honestly claim about E without importing unmarked supply? The answer is not always "proof" and not always "failure". The answer is an audited claim status: theorem, condition, route, evidence, non-evidence, meta-claim, refusal, GAP, or socket.

9456.1 Finite expressions and closed claim substrates

Definition 9456.1 (Finite communicable expression)**.** *A* finite communicable expression *is a finite inscription or program fragment that can be exchanged, inspected, and audited. Definitions, theorems, proofs, conjectures, programs, algorithms, theories, physical interpretations, observation reports, equivalences, proof routes, rule towers, meta-theorems, counterexamples, and numerical evidence all count as finite communicable expressions when their presented form is finite. The expression may quantify over an infinite extension; finiteness concerns the inscribed artifact.*

Definition 9456.2 (Closed claim substrate)**.** *A* closed claim substrate *is a system $\mathcal{C}$ whose internal uses must pass through records, generators, classifiers, continuations, ledgers, packages, certificates, or audit gates. If a dependency cannot be generated, defined, proved, classified, certified, or ledgered through those structures, then it may not enter as anonymous theorem content. It must be exposed as a named boundary request.*

Principle 9456.3 (Closedness as named externality)**.** *For BEDC, closedness is an accounting invariant:*

$$\textit{closed} \quad = \quad \textit{all external dependencies have apophatic names.}$$

The invariant does not say that no external pressure appears. It says that no external pressure may be consumed without a visible name, ledger, or socket.

Definition 9456.4 (Positive and apophatic names)**.** *A* positive name *names an internally generated or internally specified object: a carrier, constructor family, operation, law, eliminator, classifier, or certificate field. An* apophatic name *names an external dependency position. It makes the dependency referable, turns it into an auditable unit, and marks the boundary at which internal digestion stops; it does not convert the dependency into a theorem, object, witness, quotient representative, proposition equality, or causal law.*

Definition 9456.5 (Gap, socket, and far end)**.** *For a closed claim substrate $\mathcal{C}$, a request* req *forms* $\mathsf{GAP}_{\mathcal{C}}(\mathrm{req})$ *when it asks for content not produced by the displayed substrate. If that gap is named as a typed or ledgered boundary position, it forms* $\mathrm{socket}_{\mathcal{C}}(\mathrm{req})$*. For a forward-binding socket s, its far end has only the apophatic reading*

$$\mathrm{FarEnd}(s) \equiv_{\mathrm{apo}} \mathsf{T}.$$

The symbol T *is not a proof source, state, rule, value, or global oracle. It is the common far-end name for uninternalized supply.*

9456.2 Claim status

Definition 9456.6 (Claim status)**.** *For a closed claim substrate $\mathcal{C}$ and a finite communicable expression E,* $\mathsf{ClaimStatus}_{\mathcal{C}}(E)$ *is the audited status that $\mathcal{C}$ may assign to E. The principal statuses are:*

Theorem row. *E has a closed proof in the substrate.*

Conditional row. *E is closed relative to displayed hypotheses, setup fields, or witness obligations.*

Route row. *E supplies transports, equivalences, or a proof-search route without the witness needed to close the target.*

Evidence row. *E supplies a trace, report, numerical datum, or metric indication that is admissible as evidence but not as theorem content.*

Non-evidence row. *E supplies a report that is explicitly barred from serving as proof evidence.*

Meta row. *E is a claim about a fragment, system, proof surface, or audit layer rather than a kernel-level theorem of the object language.*

Refusal row. *E attempts to consume hidden supply and is rejected as an internal claim.*

GAP **row.** *E exposes requested content not produced by the displayed substrate.*

Socket row. *E names the boundary position at which such requested content would have to enter.*

Definition 9456.7 (Claim compiler)**.** *The* claim compiler *of a closed substrate is the audit operation*

$$\mathsf{Compile}_{\mathcal{C}} : \mathsf{FiniteExpr} \longrightarrow \mathsf{ClaimStatus}_{\mathcal{C}}.$$

Its inputs may be proof traces, programs, definitions, observations, theories, routes, equivalence chains, or meta-claims. Its outputs are claim-status rows. The compiler is therefore not merely a theorem machine; it is a proof auditor, boundary detector, and refusal surface.

Theorem 9456.8 (Claim-status exhaustion)**.** *Let $\mathcal{C}$ be a closed claim substrate and let E be a finite communicable expression encoded for audit. Then every accepted reading of E falls into one of the claim-status rows of Definition 9456.6. If E requests supply not generated by $\mathcal{C}$, then E cannot be read as an unconditional theorem row unless that supply is first discharged.*

Proof. Inspect the dependencies of the encoded expression. If they are all generated, defined, proved, classified, certified, or ledgered through $\mathcal{C}$, then the expression lands on the internal side: theorem, condition, evidence, non-evidence, route, or meta-row according to the shape of the supplied data. If some dependency is requested but not produced, the closedness invariant prevents anonymous internal use. The request is therefore recorded as a GAP; if the request is named as a boundary position, it is recorded as a socket. If the expression tries to consume the missing supply as theorem content, the proper status is a refusal row. These alternatives exhaust the admissible audit readings. □

Theorem 9456.9 (No hidden supply)**.** *In a closed claim substrate, ungenerated, uncertified, and unledgered supply breaks closedness when consumed as theorem content. The same supply preserves closedness when exposed as a socket and not consumed as an internal proof source.*

Proof. Closedness requires that every internal use be generated, certified, or ledgered, and that every external dependency have an apophatic name. An ungenerated supply consumed as theorem content is an anonymous external dependency, so it violates the invariant. If the supply is exposed as a socket, the substrate does not claim to possess it internally. It records the requested shape and the non-internalization boundary. The accounting invariant is therefore preserved. □

Corollary 9456.10 (Naming prevents smuggling)**.** *Apophatic naming does not solve the requested problem by naming it. It prevents the requested supply from being smuggled into theorem content.*

Proof. A socket gives the audit gate a stable target: the site, requested supply, non-internalization marker, and relevant gate can be inspected. No witness or theorem is produced merely by naming the socket. □

9456.3 Dual audit bases

Definition 9456.11 (MetaCIC audit base)**.** *The* MetaCIC *audit base is the closed-CIC meta-theory surface. It audits typing surfaces, beta steps, closedness, substitution, consistency assembly, subject reduction, normalisation, confluence, and decidable checking.*

Definition 9456.12 (GroundCompiler audit base)**.** *The* GroundCompiler *audit base is the compiler-facing surface. It audits event flows, source channels, generated recognizers, certificate gates, metric boundaries, cannot-claim rows, streaming discipline, and hidden-input boundaries.*

Principle 9456.13 (Dual-base non-collapse)**.** *BEDC self-audit uses the composition of* GroundCompiler *and* MetaCIC*, not their collapse. GroundCompiler acceptance does not become a proof-theoretic theorem by itself, and MetaCIC local decidability does not become a global compiler or route oracle.*

Theorem 9456.14 (Dual audit of claim status)**.** *For claims that involve both compiled artifacts and proof-theoretic content, the legitimate status must pass through both audit bases. A pass on one base cannot silently discharge a blocked edge on the other.*

Proof. GroundCompiler rows certify how an artifact entered the displayed event, source-channel, recognizer, and certificate-gate surface. MetaCIC rows certify how a proof-theoretic claim behaves with respect to typing, reduction, substitution, and closedness. These are different coordinates. If a claim is missing a source-channel boundary, MetaCIC typing does not provide it. If a claim is missing subject-reduction transport, a compiler recognition row does not provide it. The two bases therefore cooperate without substituting for each other. □

9456.4 Infinite claims and global deciders

Definition 9456.15 (Finite grounding of an infinite claim)**.** *An infinite claim A_∞ has finite grounding when it is presented by a finite generator, recognizer, continuation system, classifier, or packet family together with a* NameCert*, totality witness, correctness witness, and dependency ledger sufficient to generate the required instances or witnesses.*

Definition 9456.16 (Grounding rank)**.** *The grounding rank of a claim is* 0 *when the claim itself has a finite certificate. The rank is $n + 1$ when the claim is generated by a rule whose rank is n. If no finite rank is available, the rank is ω.*

Theorem 9456.17 (Infinite-claim closure)**.** *An infinite claim with finite grounding rank may be used as an auditable BEDC claim. An infinite claim of rank ω cannot be consumed as proof resource; it becomes a* GAP *socket unless further finite grounding is supplied.*

Proof. Finite rank gives a finite top certificate. Certified transports or generation steps then unfold the finite tower down to the target claim. Rank ω gives no finite top. Treating the whole tower as a single object only creates a further demand for a generator and certificate for that tower. Without those data, the demand is a gap; once named, it is a socket whose far end is read apophatically as T. □

Definition 9456.18 (Global closure decider)**.** *A global closure decider is a classifier $D_{\mathrm{all}}(A, R)$ deciding, for every theorem target A and proof route R, whether R eventually outputs a closed proof of A.*

Theorem 9456.19 (Global closure deciders are not legitimate claim content)**.** *No global closure decider is a legitimate internal BEDC claim object.*

Proof. If such a decider existed, then for any certificate-level continuation system $\mathsf{NameCert}_P$ and history h, form a route that runs the Cont_P-chain from h and outputs a fixed proof of a trivial target exactly when that chain terminates. The decider would determine whether the route closes, and hence whether the Cont_P-chain terminates. This is the halting predicate of chapter 9368, which closes the open meta loop at the certificate stratum. The reflection boundary of chapter 9461 forbids that closure as internal claim content. □

9456.5 The legitimate-claim theorem

Definition 9456.20 (Legitimate-claim problem)**.** *The* legitimate-claim problem *asks: given a closed claim substrate $\mathcal{C}$ and a finite communicable expression E, what is the strongest non-oracular status that $\mathcal{C}$ may assign to E?*

Theorem 9456.21 (Legitimate-claim theorem)**.** *BEDC solves the legitimate-claim problem for finite communicable expressions relative to a closed claim substrate: it assigns the strongest non-oracular claim status obtainable from the displayed generators, classifiers, continuations, ledgers, packages, certificates, audit gates, MetaCIC rows, GroundCompiler rows, gap reports, and sockets.*

Proof. By Definition 9456.8, every accepted reading of an encoded finite expression lands in a claim-status row. If all relevant dependencies are internal, the strongest status is read from the displayed proof, condition, route, evidence, non-evidence, or meta-row. If a dependency is not generated, Definition 9456.9 prevents its anonymous consumption and forces a gap or socket reading. If

a claim requires both compiler-facing and proof-theoretic support, Definition 9456.14 prevents either audit base from impersonating the other. If a claim tries to use a global closure oracle, Definition 9456.19 rejects it as open-meta-loop closure. Thus the assigned status is non-oracular and maximal relative to the displayed data. □

Corollary 9456.22 (Universal audit interface)**.** *BEDC is a universal audit interface, not universal truth possession. It does not assert that every true proposition is provable, every question is decidable, every external object is internalizable, or every boundary problem has a positive solution. It asserts that each finite claim must expose its status as theorem, condition, route, evidence, non-evidence, meta-claim, refusal,* GAP*, socket, or* T*-boundary.*

Proof. The legitimate-claim theorem assigns an audited status to the finite claim. Some statuses are proofs; others are conditional, evidential, boundary, or refusal statuses. The theorem therefore gives an audit interface rather than a total truth predicate. □

9456.6 RH and observer language as tests

Corollary 9456.23 (Riemann Hypothesis as witness-target test)**.** *The Riemann Hypothesis illustrates the legitimate-claim theorem at maximal mathematical pressure. In BEDC it is a fixed constructive* Π*-statement; a proof requires the witness function of Definition 9239.8; a disproof requires an explicit counterexample packet; independence requires a fragment-specific meta-theorem; unclosed equivalence chains remain routes; infinite rule towers without finite grounding become sockets; and global route deciders are halting-style oracles.*

Proof. This is exactly the RH boundary analysis of chapter 9336, read as an instance of Definition 9456.21. □

Corollary 9456.24 (Observer language as claim-status reconstruction)**.** *The same audit interface applies to observer, time, space, and universe language. Such vocabulary is not admitted as an anonymous background. It must be reconstructed as record accumulation, local inscription, inter-history coherence, gap report, socket, or* T*-boundary.*

Proof. The apophatic far-end chapters identify closed observational systems, forward-binding sockets, local inscription, and the common far-end name T. A claim about observers, time, space, or universe therefore enters BEDC through the same status surface: generated records when available, coherence rows when displayed, and sockets when the requested supply is not internalized. □

Corollary 9456.25 (Finite closed systems can speak without hidden supply)**.** *A finite closed system can speak about infinity, proof, programs, observers, time, space, truth, self-reference, and externality without turning unmarked external supply into theorem content.*

Proof. The system speaks by assigning claim status rather than by pretending that every claim is an internal theorem. Infinite claims require finite grounding, proof claims require proof data, program claims face the halting boundary, observer claims face the inscription and socket boundary, and externality is named apophatically. Each case preserves the closedness invariant. □

Concretized in §4735: Claim-status audit rows are realized as a finite NameCert carrier for theorem, route, evidence, refusal, GAP, and socket readings.

Concretized in §5122: Legitimate-claim audit ledgers realize claim source, admissibility gate, refusal socket, marker provenance, replay, and local naming rows.

Concretized in §7318: Claim audit boundaries realize finite claim-status, refusal, GAP/socket, and permitted-claim rows as a concrete NameCert packet.

Concretized in §7900: Claim audit witnesses realize claim text, cited object, two-axis tokens, off-table refusal, replay, and local naming rows.

Chapter 9457

Russell Problem Boundary

BEDC inherits a problem from Russell, not Russell's logicist system. The inherited problem is the demand that mature mathematics be read from a small, explicit discipline rather than accepted as a stock of primitive objects. BEDC does not import the axioms, ramified type theory, reducibility principle, or class ontology of *The Principles of Mathematics* or *Principia Mathematica*. It imports a reconstruction pressure: every apparently familiar mathematical object must be unfolded into a licensed pattern of relations, descriptions, contexts, strata, and witnesses.

9457.1 The inherited problem

Russell's foundational project can be read as a discipline for preventing mature mathematical grammar from smuggling in mature mathematical ontology. A phrase that looks like it denotes an object must be analysed before it is admitted. A class phrase must not automatically become a class object. A relation must not be replaced by a unary property when the relation is doing the mathematical work. A term at one type level must not be treated as a term at another type level merely because the prose makes the passage sound harmless.

BEDC accepts this discipline in a kernel-facing form. The point is not to reproduce Russell's system, but to keep the following five restrictions visible at every upward interface:

1. relation first: the behaviour that licenses a name is relational before it is object-like;
2. description expansion: a definite description is not a primitive denotation, but an analysed certificate obligation;
3. contextual class reading: a class-like expression is read through the context that classifies it, not as an automatic object;
4. stratification: a passage between levels requires an explicit bridge and does not collapse the levels;
5. witnessed descent: no downward use of an upward name is allowed without a certificate.

9457.2 Kernel reading

The finite kernel already supplies the place where this discipline lives. An asking event has the form

$$\mathsf{Ask}(\pi, h, m, \delta),$$

where π is the probe name, h is the history, m is the emitted mark, and δ is the evidence token. Read structurally, this gives five roles:

BEDC role	**Name-certificate role**	**Russell discipline**
probe or question	source entry	description expansion
history	source material	contextual class reading
observation relation	pattern and classifier	relation first
result mark	stability-consumable result	stratification
evidence token	ledger policy	witnessed descent

Thus Russell's discipline is not an external philosophical decoration. It is a reading rule for the same fivefold structure that the BEDC kernel already uses to keep an observation from becoming an object without a certificate.

9457.3 The upward-name reading

For a mature mathematical expression X, the expression $X^{\uparrow}$ does not mean that BEDC has found the host object X inside the kernel. It means that BEDC has licensed an interface whose source specification, pattern specification, classifier specification, stability certificate, and ledger policy allow the traditional grammar of X to be used without making X primitive.

For example, $\mathsf{Nat}^{\uparrow}$ is not the host natural-number type. It is a certificate-licensed reading of unary repetition, successor behaviour, induction, and recursion as stable public operations over histories. Likewise $\mathsf{ContinuousMap}^{\uparrow}$ is not a host-level function equipped with an imported modulus. It is a relation-plus-ledger interface whose continuity claims are consumed only through the certificate that licenses them.

This is the BEDC form of Russell's analysis of descriptions. The apparent object is replaced by the statement that a certificate exists with the right source, pattern, classifier, stability, and ledger obligations. The public name is usable only where those obligations are in scope.

9457.4 Boundary

The Russell problem boundary rejects five shortcuts.

1. A class expression may not be promoted to an object merely because later prose wants to quantify over it.
2. A description may not be used as a denoting term without existence, uniqueness or stability, and ledger obligations appropriate to the interface.
3. A host quotient, host choice principle, proof-irrelevance principle, or proposition-extensionality principle may not enter the public surface as if it were a BEDC derivation.
4. A level descent may not be justified by a reducibility-style slogan; it must be represented by a certificate that names the bridge.
5. A bridge readback may not be treated as kernel identity; it is a consumed certificate, not a collapse of the source and target strata.

These rejections are not additions to the kernel. They are consequences of taking $\mathsf{NameCert}$ as the universal admission rule for mature names.

9457.5 Analysis spine

The reconstruction of analysis follows the same spine. First, unary repetition is licensed as $\mathsf{Nat}^{\uparrow}$. Relation calculus and order are then licensed as classifier-stable behaviours over the arithmetic layer. Series, segments, cuts, rationals, regular rational sequences, and real numbers are admitted only as upward names whose ledger records the hidden convergence or approximation work. Limits, continuous maps, derivatives, integrals, measure, and functional analysis are then read as later certificates over that real-analysis base.

In compressed form, the intended path is

$$\mathsf{Nat}^{\uparrow} \to \mathsf{RelationCalculus}^{\uparrow} \to \mathsf{Order}^{\uparrow} \to \mathsf{Series}^{\uparrow} \to \mathsf{Segment}^{\uparrow}, \mathsf{Cut}^{\uparrow} \to \mathsf{Rat}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{Limit}^{\uparrow} \to \mathsf{Conti}$$

This path is the BEDC version of proving analysis from minimal disciplined beginnings. The phrase "from minimal self-evident axioms" is therefore not read as permission to import axiom lists. It is read as a demand that every later object be reconstructed through the smallest available kernel roles and then admitted only through a visible certificate.

Concretized in §506: Self-membership rejection is represented as a finite NameCert boundary for contextual class readings.

Concretized in §518: The Russell analysis spine is realized as a finite NameCert packet over description, contextual-class, relation-first, stratification, and witnessed-descent rows.

Concretized in §517: Russell boundary packets realize relation-first, description-expansion, contextual-class, stratification, and witnessed-descent discipline as a finite NameCert surface.

Concretized in §661: Definite-description boundaries are realized as finite BHist packets exposing existence, uniqueness, stability, hsame, Cont, Pkg, and NameCert ledger rows before any description is consumed as a public name.

Concretized in §684: Contextual class readings are realized as finite NameCert packets whose classifier is scoped by a displayed context rather than by host class ontology.

Concretized in §1103: Witnessed-descent ledgers realize downward use of upward names as finite source, bridge, descent request, witness, transport, provenance, and NameCert rows.

Chapter 9458

Reasoning by Contradiction: BEDC's Permitted Forms

BEDC accepts certain forms of reasoning by contradiction and rejects others, and the boundary is not a stylistic preference but a structural one. The permitted patterns – proving a negation by deriving falsity from the assumption, eliminating double negation on a decidable proposition, refuting a false existence claim by exhibiting a counterexample to its witness – all operate within the closed ground loop and never touch the meta level. The forbidden patterns – proving an existence statement by contradiction, eliminating double negation on an undecidable proposition, splitting on the law of excluded middle for a predicate over the carrier of naming certificates – all, on inspection, reduce to a single underlying move: closing the meta-level self-reference loop that chapter 9461's two-loops theorem requires to remain open. This chapter (i) catalogues the permitted refutation patterns as they appear in chapter 4's extension and continuation calculus, (ii) catalogues the forbidden patterns and traces each to its meta-closure consequence via a Tarski-style fragment of the truth predicate, (iii) states the unifying theorem that every forbidden pattern implements meta-closure, and (iv) records that the boundary thus identified is the same boundary that separates the closed ground loop from the open meta loop, with the halting capstone (chapter 9368) as the canonical sibling instance.

9458.1 Permitted refutation patterns

Definition 9458.1 (Permitted refutation pattern)**.** *A* permitted refutation pattern *is a derivation rule of the form "from $A \Rightarrow \bot$, conclude $\neg A$" or one of its decidable specialisations, where the derivation of $\bot$ uses only* Cont, hsame, msame, *and the closure laws of a fixed naming certificate. The conclusion $\neg A$ is read as the BEDC predicate "every witness of A produces $\bot$ inside the certificate's classifier", not as "A holds in no possible world".*

Example 9458.2 (Negation by derived falsity)**.** *The kernel theorem* $\neg\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *of chapter 2 is a permitted refutation. Assuming a witness w of* $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *produces $\bot$ by inversion on the constructor pattern:* msame *is a closed inductive whose only constructor identifies a mark with itself, and the constructors of* Mark *are different. The derivation never leaves the inductive carrier of* Mark *and never appeals to a meta-level predicate.*

Example 9458.3 (Double negation on decidable predicates)**.** *For a decidable predicate P on* Hist, *the rule $\neg\neg P\,h \Rightarrow P\,h$ is permitted: decidability supplies a constructive disjunction $P\,h \vee \neg P\,h$, and the second disjunct contradicts $\neg\neg P\,h$, leaving the first. The rule fails for predicates that are*

not given decidably; the failure is not a stylistic restriction but a refusal to substitute meta-level reasoning for missing constructive content.

Example 9458.4 (Counterexample to a false existence)**.** *Refuting $\exists h, P\,h$ by exhibiting a uniform* Hist *map φ with $\neg P(\varphi h)$ for every h is permitted: the witness φ is a kernel-level continuation that converts a putative existential witness into a $\bot$. The refutation is constructive in the same sense the existence claim would have been.*

9458.2 Forbidden patterns and their meta-closure

Definition 9458.5 (Forbidden pattern)**.** *A* forbidden pattern *is a derivation step that produces a kernel-level conclusion of one of the following shapes without supplying the corresponding constructive witness:*

1. *proving $\exists h, P\,h$ from the assumption $\neg\forall h, \neg P\,h$;*
2. *eliminating double negation on a predicate not given decidably;*
3. *splitting on the law of excluded middle for a predicate over the carrier $\mathcal{C}$ of all naming certificates.*

Theorem 9458.6 (Each forbidden pattern implements meta closure)**.** *For every forbidden pattern of Definition 9458.5, accepting the pattern as an admissible derivation step at the certificate level closes the open meta loop of chapter 9461: it supplies a* NameCert*-level decision for a predicate whose decidability is not constructively available, and the supplied decision is, structurally, an* NameCert*-level value of the host's truth predicate.*

Sketch. For (i): the move from $\neg\forall\neg$ to $\exists$ supplies a witness without exhibiting one. Reading the existential as a NameCert-level extension predicate, the witness is the value the predicate takes when applied to the assumption; reading the value back as a constructor of the witness type closes the meta loop. For (ii): double-negation elimination on a non-decidable P supplies a value of $P\,h$ without a constructor; reading P as a NameCert classifier exposes the supplied value as a NameCert-level truth witness. For (iii): excluded middle on a predicate over $\mathcal{C}$ asserts that the certificate-of-certificates settles every certificate-level proposition, which is precisely the assumption that the meta loop is closed (Definition 9461.14). □

9458.3 The boundary as ground vs meta loop

Theorem 9458.7 (Permitted iff stays within the closed ground loop)**.** *A refutation pattern is permitted in BEDC if and only if its derivation factors through the closed ground loop of chapter 9461: every step uses* Cont*,* hsame*,* msame*, and the closure laws of a fixed naming certificate, and no step appeals to a* NameCert*-level predicate over the carrier of all naming certificates.*

Sketch. Forward: Definition 9458.1 restricts the derivation primitives to those of the kernel together with a fixed certificate, and these primitives are exactly the closed ground loop's data. Backward: a derivation that stays in the ground loop produces only certificate-level consequences supported by their constructors, hence avoids the meta-closure characterised by Definition 9458.6. □

Concretized in §503: Witnessed refutation packages the permitted contradiction boundary as explicit Hist-level evidence.

Remark 9458.8 (Sibling instance: halting)**.** *The same boundary, applied to the predicate "the continuation chain from h in* $\mathsf{NameCert}_P$ *terminates", produces the halting obstruction of Definition 9368.3. Halting decidability is the canonical instance of the third forbidden pattern: a* NameCert*-level decision for a predicate over the carrier of all naming certificates. The catalogues of Definition 9458.1 and Definition 9458.5 therefore unify the proof discipline of the kernel with the halting result of the operational-semantics capstone.*

Remark 9458.9 (Sibling instance: three axioms)**.** *The chapter is also the discipline-side complement of chapter 9387: the three axiom families classified there as unwitnessed-content acceptance are sorted against the first forbidden pattern at the term, type or classifier, and proposition layers respectively. Refusing the forbidden patterns at the proof level and refusing the three axiom families at the foundation level are matching audit readings of meta-closure refusal.*

Concretized in §493: The permitted and forbidden contradiction boundary is realized as a finite $\mathsf{DecidableRefutation}^{\uparrow}$ NameCert surface.

Concretized in §511: A finite NameCert packet records constructive refutation as an assumption-to-bottom route without exporting certificate-indexed truth branches.

Concretized in §507: Stable negation is represented as a finite NameCert boundary separating permitted refutations from certificate-wide truth branches.

Concretized in §500: finite NameCert carrier for permitted refutation routes and forbidden truth-branch exclusion

Concretized in §501: Permitted refutations are realized as finite $\mathsf{ConstructiveRefutation}^{\uparrow}$ contradiction-route packets, while forbidden truth branches have no carrier coordinate.

Concretized in §502: Constructive refutation boundaries are realized as finite BHist proof-discipline packets separating permitted negation routes from certificate-indexed truth branches.

Concretized in §512: The $\mathsf{RefuterTrace}^{\uparrow}$ certificate realizes the permitted-refutation side of the contradiction boundary as finite BHist trace data.

Concretized in §497: Concrete finite-row NameCert for the permitted contradiction boundary.

Definition 9458.10 (Certificate-indexed truth branch)**.** *Let $\mathcal{C}$ be the carrier of all naming certificates. For a predicate $Q : \mathcal{C} \to \mathsf{Prop}$, a* certificate-indexed truth branch *for Q is a family*

$$\mathsf{Branch}_Q : \forall N : \mathcal{C},\ Q(N) \vee \neg Q(N).$$

When Q ranges over certificate-level statements represented by the certificate carrier, such a branch is a truth-predicate fragment at the NameCert *stratum: it chooses, inside the certificate layer, a positive witness or a refutation for each represented statement.*

Lemma 9458.11 (Excluded-middle branches are always double-negated)**.** *For every proposition A, there is a constructive derivation of $\neg\neg(A \vee \neg A)$.*

Proof. Assume $u : \neg(A \vee \neg A)$. Define $n_A : \neg A$ by sending a witness $a : A$ to $u(\mathsf{inl}(a))$. Then $\mathsf{inr}(n_A)$ is a witness of $A \vee \neg A$, so $u(\mathsf{inr}(n_A))$ gives $\bot$. Thus every assumed refutation of $A \vee \neg A$ produces $\bot$, which is exactly $\neg\neg(A \vee \neg A)$. □

Theorem 9458.12 (Forbidden contradiction schemas produce certificate truth branches)**.** *Let $\mathcal{C}$ be the carrier of all naming certificates. The following three uniform schemas each imply a certificate-indexed truth branch for every predicate $Q : \mathcal{C} \to \mathsf{Prop}$:*

1. *the existential-by-contradiction schema* $\neg\forall h : \mathsf{Hist}, \neg R(h) \Rightarrow \exists h : \mathsf{Hist}, R(h)$ *for arbitrary history predicates* R*;*
2. *the double-negation schema* $\neg\neg A \Rightarrow A$ *for propositions not supplied with a decidability row;*
3. *the excluded-middle schema* $Q(N) \vee \neg Q(N)$ *for predicates over* $N : \mathcal{C}$*.*

Consequently each forbidden pattern of Definition 9458.5, when admitted uniformly as a certificate-level derivation rule, closes the open meta loop of Definition 9461.15 by supplying the truth-predicate fragment forbidden by Definition 9461.14.

Proof. Fix a predicate $Q : \mathcal{C} \to \mathsf{Prop}$ and a certificate $N : \mathcal{C}$. Put

$$A_N := Q(N) \vee \neg Q(N).$$

By Definition 9458.11, there is a constructive witness $d_N : \neg\neg A_N$.

For the existential-by-contradiction schema, define a history predicate

$$R_N(h) := \mathsf{hsame}(h, \mathsf{e}) \wedge A_N.$$

If $u : \forall h : \mathsf{Hist}, \neg R_N(h)$, then $u(\mathsf{e})$ refutes $R_N(\mathsf{e})$. But the empty-history rule in Definition 3.2 supplies $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, so any $a : A_N$ gives a witness $(\mathsf{hsame}(\mathsf{e}, \mathsf{e}), a)$ of $R_N(\mathsf{e})$, contradicting $u(\mathsf{e})$. Hence u implies $\neg A_N$, and this contradicts d_N. Therefore $\neg\forall h : \mathsf{Hist}, \neg R_N(h)$ holds. Applying the existential-by-contradiction schema gives $\exists h : \mathsf{Hist}, R_N(h)$, and projecting the second component of its witness gives A_N.

For the double-negation schema, apply the schema directly to the proposition A_N. The premise $d_N : \neg\neg A_N$ is constructive, and the schema returns A_N.

For the excluded-middle schema over $\mathcal{C}$, the instance at N and Q is exactly A_N.

Thus each of the three schemas supplies $Q(N) \vee \neg Q(N)$. Since N was arbitrary, each supplies the family Branch_Q of Definition 9458.10. When Q ranges over certificate-level statements represented by $\mathcal{C}$, this family is a $\mathsf{NameCert}$-level truth-predicate fragment. By Definition 9461.14, such a fragment is precisely what BEDC's self-description cannot internalize as its own truth predicate; by Definition 9461.15, keeping that layer open is the consistency side of the two-loops boundary. Therefore uniformly admitting any forbidden schema implements meta closure. □

Concretized in §510: GroundRefutationUp realizes the permitted refutation branch as a finite NameCert packet.

Concretized in §508: The reasoning-by-contradiction boundary is realized as a finite NameCert packet for ground-loop refutation routes and meta-escape exclusions.

Concretized in §854: Certificate truth-branch boundaries are realized as finite BHist NameCert packets separating permitted refutation routes from forbidden certificate-indexed truth branches.

Concretized in §899: Forbidden contradiction schemas are realized as a finite obstruction packet separating permitted refutation from meta-loop truth closure.

Chapter 9459

Theory Stability and the TasteGate Framework

The 280-plus concrete-instance chapters of `papers/bedc/parts/` formalise objects whose mathematical-consensus credit was earned by the mathematics community over decades or centuries: $\mathsf{Group}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{DiffGalois}^{\uparrow}$, and the rest. Each chapter inherits this credit; the BEDC formalisation does not need to re-test whether $\mathsf{Group}^{\uparrow}$ is a 'real theory', because the question was settled outside the system.

A separate admission mode applies when codex itself proposes a chapter. An AI-proposed chapter has no mathematical-consensus credit. It might be combinatorial padding, a synonym of an existing chapter under another name, a carrier that crosses the BEDC zero-axiom boundary, or a locally self-consistent surface that breaks stability of the existing horizon when admitted. The taste gate of this chapter records the discipline by which AI-proposed chapters earn admission.

The discipline reuses the conservativity, no-hidden-input, round-trip, and source-channel-separation theorems already established in the ground compiler chapters (chapter 9207, chapter 9208, chapter 9209, chapter 9222). It elevates those compiler-internal invariants into a chapter-level taste gate.

9459.1 Carrier display obligations

A candidate chapter X earns admission to the AI-proposed track by first committing to a concrete carrier display. The Lean class `BHistCarrier X` supplies

$$\mathsf{toEventFlow} : X \to \mathsf{EventFlow}, \qquad \mathsf{fromEventFlow} : \mathsf{EventFlow} \to \mathsf{Option}(X).$$

The chapter-specific gate then asks for only the two obligations that genuinely depend on X.

Definition 9459.1 (Conservativity obligation)**.** *Conservativity is not a field supplied by the chapter. It is the theorem that every mark occurring in the display of an admitted token is one of the two BEDC ground marks* `b0` *or* `b1`*, derived from GroundCompiler event-flow conservativity.*

Definition 9459.2 (No-hidden-input obligation)**.** *No-hidden-input is not a field supplied by the chapter. It is the theorem that every token $x : X$ has a visible event-flow witness e with*

$$\mathsf{fromEventFlow}(e) = \mathsf{some}(x),$$

obtained by taking $e = \mathsf{toEventFlow}(x)$ and applying the round-trip obligation.

Definition 9459.3 (Round-trip obligation)**.** *The round-trip obligation for X asserts*

$$\forall x : X, \quad \mathsf{fromEventFlow}(\mathsf{toEventFlow}(x)) = \mathsf{some}(x).$$

Every admitted token must be the readback of its own displayed event flow.

Definition 9459.4 (Layer-separation obligation)**.** *The layer-separation obligation for X asserts*

$$\forall x, y : X, \quad x \neq y \Rightarrow \mathsf{toEventFlow}(x) \neq \mathsf{toEventFlow}(y).$$

Distinct carrier tokens therefore have distinct displayed event flows.

9459.2 Lean interface

The Lean interface separates carrier data from gate data:

```
class BHistCarrier (X : Type) where
  toEventFlow : X -> EventFlow
  fromEventFlow : EventFlow -> Option X

class ChapterTasteGate (X : Type) [BHistCarrier X] where
  round_trip :
    forall x : X,
      BHistCarrier.fromEventFlow
        (BHistCarrier.toEventFlow x) = some x
  layer_separation :
    forall x y : X, x <> y ->
      BHistCarrier.toEventFlow x <>
        BHistCarrier.toEventFlow y
```

Lean4 (definition) `BEDC.Meta.TasteGate.BHistCarrier`
Lean4 (definition) `BEDC.Meta.TasteGate.ChapterTasteGate`
Lean4 ✓ `BEDC.Meta.TasteGate.ChapterTasteGate.conservativity`
Lean4 ✓ `BEDC.Meta.TasteGate.ChapterTasteGate.no hidden input`

The reference instance witnesses that the ground compiler's own event-flow carrier satisfies the gate by identity display and identity readback. The conservativity and no-hidden-input readings are obtained from the derived theorems above, not from chapter-filled fields.
Lean4 ✓ `BEDC.Meta.TasteGate.groundCompilerSelfTasteGate`

9459.3 Field faithfulness and nontriviality

The current Lean surface also records two opt-in strengthening gates. `FieldFaithful` requires a chapter to list its BHist fields and prove that equal field lists determine equal carrier tokens. `Nontrivial` requires a displayed pair of distinct inhabitants, excluding one-token carriers whose layer-separation proof would be vacuous.
Lean4 (definition) `BEDC.Meta.TasteGate.FieldFaithful`
Lean4 (definition) `BEDC.Meta.TasteGate.Nontrivial`

These gates do not replace `ChapterTasteGate`. They refine it: the chapter must first be displayable and readable, and may then expose additional evidence that the display reflects every declared field and that the carrier is not a vacuous singleton.

9459.4 Relationship to the NameCert obligations

The *naming-certificate* obligation set (chapter 9) records source, classifier, pattern, ledger, and extension obligations within a single chapter. The TasteGate obligations record stability across chapters and across the ground compilation layer. The two are orthogonal:

- A chapter satisfying NameCert alone is internally consistent but not necessarily stable when admitted to the global horizon.
- A chapter satisfying TasteGate alone has display/readback stability but is not yet a usable BEDC carrier.
- Human-curated chapters supply NameCert and inherit TasteGate credit from mathematical consensus.
- AI-proposed chapters must supply both NameCert and TasteGate.

The chapter does not retroactively impose TasteGate on the existing human-curated horizon. The audit gate distinguishes the two tracks by a single `\origin{human|ai}` field on the `closurestatus` block; absence defaults to `human` and preserves backward compatibility.

9459.5 Cannot-claim registry

The chapter does not claim a complete answer to the underlying philosophical question of what makes a mathematical object 'real'; it claims only a machine-checkable filter. The chapter does not claim that any failure of the four obligations rules a candidate out as mathematics in some absolute sense; it rules the candidate out of *this* BEDC horizon under the AI-proposed track. The chapter does not claim that human-curated chapters whose four obligations have not been mechanically checked are incorrect; it claims only that they inherit consensus credit and need not be re-checked at admission time. The chapter does not claim a semantic conservativity theorem for arbitrary chapter X relative to an arbitrary BEDC baseline. It claims the narrower ground-display conservativity supplied by the GroundCompiler event-flow theorem for the displayed event flows of any admitted carrier.

Concretized in §290: Policy packets realize the TasteGate discipline as finite BHist action-selection certificates with explicit ground-compiler witness rows.

Concretized in §545: TasteGateAdmission packets realize AI-origin chapter admission as finite BHist rows carrying ChapterTasteGate, ground-compiler witness, audit-gate, hsame, Cont, Pkg, GAP, and NameCert data.

Concretized in §563: TasteGate admission is realized as a finite compiler-witness NameCert packet tying ChapterTasteGate obligations to ground-compiler witness rows.

Concretized in §564: TasteGate stability is realized as finite BHist NameCert packets over ground-compiler witness rows, admission ledger, non-hidden-input, round-trip, and layer-separation gates.

Concretized in §993: Finite reflection tuples realize TasteGate stability as displayed tuple, image, readback, componentwise NameCert transport, and reflection-request rows.

Concretized in §552: TasteGate audit traces are realized as finite BHist packets carrying the four obligation witnesses, audit replay, ground-compiler provenance, and non-escape ledger for AI-origin chapter admission.

Concretized in §7186: Finite kernel closure gates realize the TasteGate closure-admission discipline as finite BHist gate, audit, transport, replay, provenance, and NameCert rows.

Chapter 9460

Theorem-DNA Scoring TasteGate

chapter 9459 records the carrier-facing gate: a chapter carrier must display as an event flow, read back through a partial decoder, and separate distinct carrier tokens by distinct displays. chapter 9408 records the theorem-facing profile: a theorem flow is not a statement string but an eight-locus event-flow profile with statement, dependencies, proof, certificates, ledger, status, canonical site, and closing seal. The present chapter composes the two readings into a stronger admission schema for theorem-like carriers.

9460.1 Theorem-DNA packet

Definition 9460.1 (Theorem locus set)**.** *The theorem-DNA locus set is*

$$\mathcal{L}_\Theta = \{\ell_{\mathsf{stmt}}, \ell_{\mathsf{deps}}, \ell_{\mathsf{proof}}, \ell_{\mathsf{cert}}, \ell_{\mathsf{ledger}}, \ell_{\mathsf{status}}, \ell_{\mathsf{site}}, \ell_{\mathsf{seal}}\},$$

corresponding respectively to statement, dependencies, proof, certificates, ledger, status, canonical site, and closing seal.

Definition 9460.2 (Theorem-DNA packet)**.** *For a generated theorem recognizer R and theorem candidate flow T, a theorem-DNA packet is an eight-tuple*

$$D_R(T) = (G_{\mathsf{stmt}}, G_{\mathsf{deps}}, G_{\mathsf{proof}}, G_{\mathsf{cert}}, G_{\mathsf{ledger}}, G_{\mathsf{status}}, G_{\mathsf{site}}, G_{\mathsf{seal}})$$

whose components satisfy the matching GroundCompiler role-subflow predicates. It is anchored when each component is a source subflow of T. It is sound when the recognizer also satisfies $\mathsf{SoundTheoremFlow}(R, T)$.

Theorem 9460.3 (Sound theorem-DNA is role-complete)**.** *Every sound theorem-DNA packet contains all eight theorem loci.*

Proof. Sound theorem-DNA includes a theorem-DNA packet, and the packet is defined as the eight matching role-subflow witnesses. This is the paper-side unpacking of GroundCompiler complete theorem-flow recognition. □

9460.2 Score certificate

Definition 9460.4 (Flow mass and score)**.** *For an event flow $S = (w_0, \ldots, w_{n-1})$, define*

$$|S|_{\mathsf{flow}} = n + \sum_{i=0}^{n-1} |w_i|_{\mathsf{raw}}.$$

For a theorem-DNA packet $D_R(T)$, its complexity vector is the nine-coordinate vector of the eight locus masses together with the recognizer mass $|R|_{\mathsf{flow}}$. A weight profile $\mathbf{w} \in \mathbb{N}^9$ assigns public weights to these nine coordinates, and the score $\mathsf{Score}_{\mathbf{w}}(R, D_R(T))$ is the resulting dot product.

Definition 9460.5 (Score certificate)**.** *For a bound $B \in \mathbb{N}$, a score certificate consists of a sound theorem-DNA packet $D_R(T)$ together with the displayed inequality*

$$\mathsf{Score}_{\mathbf{w}}(R, D_R(T)) \leq B.$$

It is an auditable upper bound, not a claim that B is the minimal complexity of the theorem flow.

Theorem 9460.6 (Score monotonicity)**.** *For fixed nonnegative weights, componentwise reduction of the theorem-DNA complexity vector cannot increase the score.*

Proof. This is componentwise monotonicity of a dot product with entries in $\mathbb{N}$. Each coordinate inequality remains valid after multiplication by its corresponding weight, and summing preserves the inequality. □

Theorem 9460.7 (Scalar score depends on the weight profile)**.** *A scalar theorem-DNA score is auditable only together with its weight profile.*

Proof. Two complexity vectors can trade off statement mass against dependency mass. A weight profile concentrated on the first coordinate orders them one way, while a profile concentrated on the second coordinate orders them the other way. Therefore the weight profile is part of the score certificate. □

9460.3 The strengthened gate

Definition 9460.8 (Canonical carrier)**.** *For a carrier X with a `BHistCarrier` display, the canonical carrier condition consists of the round-trip equation*

$$\forall x : X, \quad \mathsf{fromEventFlow}(\mathsf{toEventFlow}(x)) = \mathsf{some}(x)$$

and image exactness

$$\forall e, x, \quad \mathsf{fromEventFlow}(e) = \mathsf{some}(x) \Rightarrow e = \mathsf{toEventFlow}(x).$$

The first clause says that the canonical display reads back. The second says that the decoder accepts no non-canonical display for the same token.

Definition 9460.9 (Theorem-DNA scoring TasteGate)**.** *A theorem-like carrier X passes the theorem-DNA scoring TasteGate at weight profile $\mathbf{w}$ and budget $B : X \to \mathbb{N}$ when:*

$$X \text{ has a } \mathsf{BHistCarrier} \text{ display},$$
$$X \text{ satisfies the canonical carrier condition},$$
$$X \text{ satisfies } \mathsf{FieldFaithful} \text{ and } \mathsf{Nontrivial},$$
$$\forall x : X,\ T_x = \mathsf{toEventFlow}(x) \text{ carries anchored sound theorem-DNA},$$
$$\forall x : X,\ \mathsf{Score}_{\mathbf{w}}(R_x, D_x) \leq B(x) \text{ for a displayed score certificate.}$$

This gate is denoted $\mathsf{TDCSGate}(X, \mathbf{w}, B)$.

Theorem 9460.10 (TDCSGate implies ChapterTasteGate)**.** *If* $\mathsf{TDCSGate}(X, \mathbf{w}, B)$ *holds, then* X *satisfies* `ChapterTasteGate`*.*

Proof. The canonical carrier condition supplies the round-trip field. To prove layer separation, suppose $x \neq y$ and $\mathsf{toEventFlow}(x) = \mathsf{toEventFlow}(y)$. Applying round-trip to both tokens gives the same decoder value, hence $\mathsf{some}(x) = \mathsf{some}(y)$, so $x = y$, a contradiction. □

Corollary 9460.11 (Derived display obligations are inherited)**.** *A carrier passing* TDCSGate *inherits the existing TasteGate conservativity and no-hidden-input theorems.*

Proof. By Definition 9460.10, the carrier satisfies `ChapterTasteGate`. Conservativity and no-hidden-input are derived theorems of that gate, not fields supplied by the carrier. □

9460.4 Exclusion principles

Theorem 9460.12 (Statement-only carriers cannot pass)**.** *A carrier whose displayed theorem candidate can supply only the statement locus and no proof locus cannot pass* TDCSGate*.*

Proof. The gate requires anchored sound theorem-DNA for every displayed token. By Definition 9460.3, such DNA contains a proof locus. This contradicts the hypothesis. The GroundCompiler theorem `statement_alone_not_theorem` is the corresponding checked boundary. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.statement alone not theorem`

Theorem 9460.13 (Code-only theorem objects cannot pass)**.** *A displayed candidate whose only evidence is legal theorem code, with no proof-sound theorem recognition, cannot pass* TDCSGate*.*

Proof. The gate requires sound theorem-DNA, hence SoundTheoremFlow for the displayed theorem candidate. Sound theorem flow contains proof-sound theorem recognition. Legal code without proof-sound recognition therefore fails the gate. The GroundCompiler theorem `theorem_code_is_not_proof` records the checked separation. □

↪ variant `BEDC.GroundCompiler.TheoremGenerated.theorem code is not proof`

Theorem 9460.14 (Anchoring prevents locus hiding)**.** *In a carrier passing* TDCSGate*, theorem-DNA loci such as proof, certificate, ledger, status, site, and seal cannot be supplied by a hidden table or external oracle.*

Proof. The gate requires anchored theorem-DNA. Therefore every locus subflow must be a source subflow of the displayed theorem candidate $T_x = \mathsf{toEventFlow}(x)$. Hidden tables and external oracles are not source subflows of T_x. □

9460.5 Score and discovery

Definition 9460.15 (DNA mutation inside the gate)**.** *For a fixed displayed theorem candidate* T_x*, a DNA mutation is a passage*

$$D_R(T_x) \rightsquigarrow D'_{R'}(T_x)$$

between two anchored sound theorem-DNA packets.

Theorem 9460.16 (Complexity reduction does not imply discovery)**.** *A mutation can reduce theorem-DNA score while leaving the theorem classifier phenotype unchanged.*

Proof. A proof-locus replacement may reduce proof mass while preserving statement, dependencies, certificate, ledger, status, canonical site, closing seal, and phenotype. With positive proof weight the score decreases, but no classifier shift has occurred. □

Theorem 9460.17 (Classifier shift need not reduce score)**.** *A classifier-shifting mutation can fail to reduce theorem-DNA score.*

Proof. A mutation that changes the theorem classifier phenotype may introduce larger certificate, ledger, seal, or recognizer loci. Under nonnegative weights these additional loci can keep the score unchanged or increase it. □

Principle 9460.18 (TDCSGate adequacy)**.** *The theorem-DNA scoring TasteGate admits only carriers whose displays are canonical, field-faithful, nontrivial, theorem-DNA complete, anchored, sound, and accompanied by a public complexity score certificate.*

Concretized in §545: TasteGate admission packets supply the carrier-facing display and readback side consumed by TDCSGate.

*Concretized in §**??**:* Theorem-flow recognition packets supply the eight-locus theorem-DNA side consumed by TDCSGate.

Concretized in §7240: The proof entropy ledger realizes theorem-DNA locus masses, public weight profiles, and score-bound nonescape as finite NameCert data.

Chapter 9461

Self-Reference in BEDC: The Ground Loop, the Reflection, and the Reflection Limit

This chapter answers one question: in what precise senses does BEDC describe itself, and where does each sense fail? The answer is two-fold and the asymmetry between the two halves is the main mathematical content. At the ground, the primitive distinction between two raw marks is *self-instantiating*: stating it is already an act of it. This is a closed loop, and it must close for the rest of BEDC to begin. At the meta-level, the naming-certificate framework can internally describe the structure of any well-founded formal system, the host calculus of inductive constructions included. This second loop must remain open: closing it would, by Tarski undefinability, force inconsistency.

The two loops together characterise what kind of foundation BEDC is. The chapter develops them in turn. section 9461.1 isolates the ground loop. section 9461.2 establishes the correspondence between formal definition and classifier choice. section 9461.3 interprets closed inductive types as classifier closures inside BEDC. section 9461.4 extends the interpretation to a structural copy of the host calculus. section 9461.5 fixes the Tarski-Münchhausen boundary. section 9461.6 contrasts the closed ground loop with the open meta loop and locates BEDC's specific contribution to foundations.

9461.1 The primitive distinction as generative loop

Definition 9461.1 (Form of distinction)**.** *Let* b0 *and* b1 *denote the two raw marks of Definition 2.1. The* form of distinction *is the closed inductive judgment generating* Mark*, given by the two formation rules of chapter 2 together with the kernel-level recognition that* b0 *and* b1 *are not the same constructor. The form is the data of: two formal tokens, and the capacity to tell them apart.*

Remark 9461.2 (Self-instantiation)**.** *The form of distinction is self-instantiating. Stating that the constructors* b0 *and* b1 *are different requires, on the part of the reader, the very capacity that the statement introduces: the capacity to distinguish two formal tokens. The act of writing down the inductive declaration is itself an act of distinction. This is the position taken by G. Spencer-Brown in* Laws of Form*: any formal system requires an irreducible primitive whose existence is given simultaneously with the act of using it. The primitive cannot be derived, only enacted.*

Theorem 9461.3 (Irreducibility of the form of distinction)**.** *No formal definition expressible inside BEDC reduces the form of distinction to anything strictly simpler. Every classifier on every BEDC carrier presupposes the form of distinction in its very statement: deciding whether two carrier elements lie in the same class is an instance of the form applied to that carrier.*

Lean4✓ `BEDC.Reflection.form of distinction irreducible proof`

↪ variant `BEDC.Reflection.form of distinction irreducible`

Sketch. A classifier $\sim$ on a carrier G assigns each $g \in G$ to an equivalence class. To assign is to decide, for each pair (g, g'), whether $g \sim g'$ holds. This decision is itself an act of distinction applied to the elements of G. Were the form of distinction reducible to a strictly simpler primitive P, the decision $g \sim g'$ would have to be expressible as an application of P alone; but P would itself, by the same argument, require deciding whether two of *its* input forms are the same, which is again the form of distinction. The regress terminates only when the form is taken as primitive. □

Remark 9461.4 (Productive loop, not vicious loop)**.** *A self-referential construction is* vicious *if its unrolling fails to terminate, producing an inconsistent or non-well-founded definition. The form of distinction is* productive*: a single application yields the two raw marks, after which all further structure is well-founded above. The host kernel rejects vicious self-reference at the level of* `def` *and* `theorem` *(a definition whose body refers to itself fails the well-foundedness check). It accepts the form of distinction at the* `inductive` *level, because at that level there is no body to refer to itself: the constructor declaration is the kernel's primitive notion of inductive type, not a derived definition. The closed loop is built into the kernel's syntactic primitives; only above the kernel do non-well-founded loops become detectable as errors.*

Remark 9461.5 (Why this loop must close)**.** *The form of distinction is the only loop in BEDC that is required to close. Every other layer is built by classifier-based well-founded recursion above it; if any of those layers contained a loop, the host kernel would reject the construction. The closed ground loop is the price of starting at all: any formal foundation that begins from absolutely nothing must, at its first move, supply the capacity that the move uses. BEDC pays this price once and at the smallest possible size.*

9461.2 Definition as classifier

Definition 9461.6 (Definition presented as a classifier)**.** *Let G be a generator base in the BEDC primitive layer (a closed inductive over* Mark*,* Hist*,* Bundle*,* Sig*, or* psame*). A* definition presented as a classifier on G *is a pair* $(\mathsf{Ext}, \mathsf{Cont})$ *together with a naming certificate over* $(G, \mathsf{Ext}, \mathsf{Cont})$ *in the sense of chapter 9. The classifier* $\sim_{(\mathsf{Ext},\mathsf{Cont})}$ *on G assigns to each $x \in G$ the equivalence class* $\{y \in G \mid \mathsf{Ext}\, x \Leftrightarrow \mathsf{Ext}\, y$ *and* $\mathsf{Cont}\, x \sim_{\mathsf{hsame}} \mathsf{Cont}\, y\}$.

Theorem 9461.7 (Definition-classifier correspondence)**.** *Let D be a predicate on a BEDC carrier G equipped with a naming certificate whose classifier preserves the carrier. Suppose that $D(x)$ is exactly the assertion that x lies in G and carries an* Ext/Cont *presentation, and suppose that the classifier transports the* Ext/Cont *presentation data. Then the classifier transports D. Conversely, two D-elements that are internally the same history are classifier-equivalent. Thus a definition over the primitive layer is precisely a carrier predicate together with stable extension-continuation presentation data.*

Lean4✓ `BEDC.Reflection.definition classifier correspondence proof`

↪ variant `BEDC.Reflection.definition classifier correspondence`

Sketch. Unfold the presentation of D. From $D(h)$ one obtains both $G(h)$ and an $\mathsf{Ext}/\mathsf{Cont}$ witness for h. The naming certificate transports $G(h)$ across any classifier edge $h \sim k$, while the stated stability of the presentation transports the $\mathsf{Ext}/\mathsf{Cont}$ witness to k. Folding the presentation of D gives $D(k)$. For the converse direction, internal sameness identifies the two histories; reflexivity of the naming certificate on the carrier part of D gives the required classifier judgment. □

Remark 9461.8. *The naming-certificate framework was originally introduced as a discipline against ad-hoc definition (definitions whose closure properties cannot be exhibited). Definition 9461.7 reverses the framing: the discipline is in fact a complete characterisation. There is no formal definition over BEDC primitives that is not expressible as a naming certificate.*

9461.3 Internalisation of inductive structure

Definition 9461.9 (Generator-closure classifier of an inductive)**.** *Let T be a closed inductive type with constructors $c_1, \ldots, c_n$ over input families $A_1, \ldots, A_n$. The* generator-closure classifier *of T is the minimal classifier $\sim_T$ on the disjoint union $\bigsqcup_i A_i$ such that for each constructor c_i and each pair of inputs $a, a' \in A_i$ with $a \sim_T a'$ componentwise, the witnesses $c_i(a)$ and $c_i(a')$ name the same equivalence class. The classifier reflects the inductive's recursor: any function $T \to X$ extends a $\sim_T$-respecting cocone over the constructor inputs.*

Theorem 9461.10 (Inductive types as classifier closures)**.** *Every BEDC carrier equipped with a naming certificate, a constructor-step carrier law, and classifier preservation of generated extension membership has a stable generator closure. If $\mathsf{ConstructorClosed}(h)$ is exactly the subpredicate of carrier elements that arise as one Ext step from some source, then $\mathsf{ConstructorClosed}$ is preserved by the classifier, and every carrier element has a primitive constructor successor in $\mathsf{ConstructorClosed}$. This is the kernel-level classifier closure underlying the interpretation of closed inductive types as generated classifier closures.*

Lean4✓ BEDC.Reflection.inductive as classifier closure proof

↪ *variant BEDC.Reflection.inductive as classifier closure*

Sketch. Unfold $\mathsf{ConstructorClosed}(h)$ as $\mathsf{Carrier}(h)$ together with an Ext witness into h. Classifier stability follows by combining the naming certificate's carrier transport with the explicit preservation of generated Ext membership. The constructor successor clause follows from the carrier's constructor-step law and the generated rules $\mathsf{Ext}(h, \mathsf{b0}, \mathsf{E0}(h))$ and $\mathsf{Ext}(h, \mathsf{b1}, \mathsf{E1}(h))$. The categorical reading is that the constructor-generated part of the carrier is the BEDC classifier closure used to interpret the inductive signature. □

9461.4 BEDC describing its own meta-system

Theorem 9461.11 (Internal description of the calculus of inductive constructions)**.** *Inside the BEDC layer there exists a structure* CIC**, given as an iterated naming certificate over a generator base of W-type-shaped inductives and a hierarchy of universe classifiers, such that* CIC* *is structurally isomorphic to the calculus of inductive constructions hosting BEDC. The isomorphism preserves type formation, dependent function formation, inductive type formation, and the recursor for each inductive.*

Lean4✓ BEDC.Reflection.internal cic interpretation proof

↪ *variant BEDC.Reflection.internal cic interpretation*

Sketch. The W-type-shaped generators are the BEDC inductives whose constructor signatures are themselves indexed by BEDC carriers. The universe hierarchy is encoded by stratifying the W-types by the size of the constructor input. Type formation, dependent function formation, and inductive formation each correspond to a closure under a specific functor, and each functor admits a naming certificate by Definition 9461.10. Recursors arise as the universal arrows out of the corresponding initial algebras. The structural isomorphism is the assembly of these componentwise correspondences. □

Corollary 9461.12 (BEDC self-description)**.** *For every formal claim φ provable in the calculus of inductive constructions about inductive types, there exists an internal claim φ^*, expressible inside BEDC over the structure* CIC* *of Definition 9461.11, that is provable from the naming-certificate framework alone. In particular, every theorem in this manuscript admits an internal restatement inside BEDC.*

Lean4✓ `BEDC.Reflection.bedc self description proof`

↪ variant `BEDC.Reflection.bedc self description`

Sketch. Apply Definition 9461.7 to translate each formal definition appearing in φ into a naming certificate over the appropriate generator. Apply Definition 9461.10 and Definition 9461.11 to interpret the inductive types appearing in φ as classifier closures inside CIC*. Each proof step in φ corresponds to a closure axiom of one of the naming certificates so produced. The translation respects well-foundedness because each classifier inherits the form of distinction at its base. □

9461.5 The reflection limit

The internalisation results of section 9461.4 are, structurally, complete: the BEDC layer has the expressive power to describe the meta-system that hosts it. Two boundaries on this self-description nevertheless remain, and both are intrinsic to any formal system possessing the same expressive power.

Remark 9461.13 (Structure-existence boundary)**.** *The internal* CIC* *of Definition 9461.11 is a* structural *reproduction of the host calculus. It is not the host calculus considered as a runtime: every Lean proof of every theorem appearing in this manuscript is checked by an external Lean kernel, which is itself implemented in a programming language whose semantics is not BEDC-internal. The isomorphism between* CIC* *and the host CIC holds at the level of formal structure; it does not hold at the level of executing the structure. No formal system can include the runtime that interprets it. This boundary is the formal content of the Münchhausen impossibility: the bootstrap remains open at the runtime level even when it is closed at the structural level.*

Remark 9461.14 (Tarski undefinability)**.** *The self-description of Definition 9461.12 is a description of the formal structure of BEDC inside BEDC; it is not a definition of a BEDC truth predicate inside BEDC. By Tarski undefinability, every formal system rich enough to describe its own syntactic structure cannot define its own truth predicate. BEDC is no exception, and Definition 9461.12 does not attempt this. Recovering truth requires a metalanguage strictly stronger than the language being described, and that metalanguage is again hosted externally.*

9461.6 The two loops of BEDC

Theorem 9461.15 (Closed ground loop and open meta loop)**.** *The form of distinction (Definition 9461.1, Definition 9461.3) is a closed self-referential primitive: writing the inductive con-*

structor declaration is itself an act of distinction, and the loop closes at this single layer with no further regress. The reflection map CIC → CIC* *(Definition 9461.11, Definition 9461.12) is an open self-description: by Definition 9461.13 and Definition 9461.14, the host kernel running BEDC remains external to* CIC*. *The closed loop and the open loop are not in tension: they live at different levels, and consistency requires precisely that the ground close and that the meta not.*

Lean4✓ `BEDC.Reflection.two loops theorem proof`

↪ variant `BEDC.Reflection.two loops theorem`

Remark 9461.16 (Why both polarities are necessary)**.** *A formal system in which the ground does* not *close cannot start: the first primitive would itself require a prior primitive, with no termination of the regress. A formal system in which the meta-level* does *close — that is, a system that can both describe its own syntactic structure and define its own truth predicate — is inconsistent by Tarski undefinability. BEDC therefore needs both polarities: the closed ground loop to bootstrap, and the open meta loop to remain consistent. The polarities are dual: one closes precisely so the other can stay open.*

Remark 9461.17 (BEDC's specific contribution to foundations)**.** *Every formal foundation carries an irreducible ground loop. Set-theoretic foundations couple sets and membership and leave the coupling implicit in the surrounding informal practice. Type-theoretic foundations couple type, term, and judgmental equality similarly. BEDC's contribution is to make the ground loop minimal and explicit: a single closed inductive declaration with two constructors, no implicit pairing, no surrounding informal practice required to interpret the primitive. The loop is taken as bedrock and left there; everything else is built on top by classifier-based well-founded recursion. This is what allows BEDC to truthfully claim zero axioms: the irreducible primitive is the constructor declaration itself, not a deductive axiom in the inferential sense, and the classifier framework over it generates the entire derived horizon without further foundational assumptions. Reading the manuscript backward from the present chapter, every later chapter is the unrolling of this single primitive distinction through a tower of well-founded classifiers, and the form of distinction at the base remains the only loop the project ever closes.*

Remark 9461.18 (Incompleteness as downstream of the ground loop)**.** *Gödel's first incompleteness theorem is the canonical instance of an open meta loop: a sufficiently expressive formal system can encode its own syntax via a fixed numbering, define a provability predicate over that encoding, and produce a self-referential formula whose unprovability is the incompleteness witness. The construction depends, at every step, on the system's capacity to distinguish formal tokens: numbers from non-numbers, formulas from non-formulas, proofs from non-proofs. Tracing this dependency to its origin yields the form of distinction (Definition 9461.1). Gödel's argument typically takes natural-number arithmetic as its substrate of distinguishable tokens; arithmetic itself rests on an inductive declaration of* Nat *with two constructors, which is the classifier closure (Definition 9461.9) of a single instance of the form of distinction. BEDC isolates this ancestry: the limit theorems of formal mathematics — incompleteness, Tarski undefinability, Münchhausen impossibility — are downstream consequences of the same closed ground loop that BEDC takes as its primitive. They are not independent constraints; they are the open-meta-loop side of the closure that lets formal systems start at all. Read this way, BEDC's claim is not that it escapes Gödel but that it makes the irreducible primitive on which Gödel-type results depend explicit and minimal: a single inductive declaration of two raw marks, one closed loop, no further regress.*

Remark 9461.19 (Distinction from a Hofstadter strange loop)**.** *Reflective self-description as exhibited above is sometimes confused with a Hofstadter-style strange loop, in which a system levels*

itself through self-reference. The two are formally distinct. A strange loop in the formal sense identifies the describing layer with the described layer; this immediately produces inconsistency by Tarski's argument. The BEDC reflection of Definition 9461.12 stratifies: the description of CIC* *lives strictly above the BEDC primitive layer, and the host kernel checking the description lives strictly above the description. Each layer talks only about layers strictly below itself. The Lean kernel enforces this stratification physically: a definition whose body refers to itself is rejected as not well-founded before it can compromise the system. The single exception is the form of distinction at the ground, where there is no body to refer to itself — only a constructor declaration whose existence is the kernel's primitive notion of inductive type. Strange loops in the philosophical sense (descriptive feedback through art, music, or self-aware cognition) do not have this kernel-level guarantee and are therefore not reproducible inside a formal system without losing consistency.*

Remark 9461.20 (Reasoning patterns and axioms as meta-closure entries)**.** *The two-loops structure also classifies the proof patterns and axioms BEDC accepts versus rejects. Proof of negation, double-negation elimination on a decidable proposition, and refutation of a false existence claim by exhibited counterexample stay within the closed ground loop and BEDC accepts them. Proof of an existence statement by contradiction does not: it requires the law of excluded middle on the existential, which is decidability of a fragment of the truth predicate, which by Tarski's argument is read here as a meta-closure move. The three axiom families BEDC refuses — the axiom of choice, quotient soundness, and propositional extensionality — are placed by chapter 9387 in a taxonomy of unwitnessed closure shapes at the term, type or classifier, and proposition strata. That taxonomy is an audit reading, not a proof that the host axioms are equivalent to BEDC ontic collapses. Halting decidability (chapter 9368) is treated as a related recursion-theoretic boundary. The full chapters chapter 9458 and chapter 9387 record these classifications; the present remark uses them as interpretive companions to the two-loops theorem (Definition 9461.15). BEDC's axiom-purity gate is therefore not a list of three orthogonal disciplinary refusals but a sorted refusal of unledgered meta closure.*

9461.7 Position of this chapter in the BEDC architecture

The reflection results of this chapter are not used by any other BEDC construction; they sit at the top of the dependency tower and are read after the rest of the manuscript. Their role is auditative: they establish that the apparatus of naming certificates, classifier closures, and primitive inductives is expressive enough to describe its own logical form, that the ground loop the project relies on is exactly one and is taken explicitly at the smallest possible size, and that the meta-level reflection of the host calculus stops at the Tarski-Münchhausen boundary common to all sufficiently expressive formal systems. Together with the orientation discipline of chapter 1 and the kernel boundary of chapter 14, they complete the meta-architectural triangle: the project knows what it is built on, what it is bounded by, and what it can say about itself.

Concretized in §1038: Tarski truth-refusal is realized as a finite NameCert packet separating CIC-star sentence codes from any internal truth predicate.

Definition 9461.21 (Runtime candidate over CIC*)**.** *Let* CIC* *be the internal structure of Definition 9461.11. A* runtime candidate *over* CIC* *is a derived BEDC interface* $\mathsf{NameCert}_R$ *whose source specification* $\mathsf{SourceSpec}_R$ *consists of coded* CIC* *expressions, proof objects, and states; whose pattern specification* $\mathsf{PatternSpec}_R$ *records a checking or evaluation transition; whose classifier specification* $\mathsf{ClassifierSpec}_R$ *groups rows with the same accepted result; and whose ledger policy* $\mathsf{LedgerPolicy}_R$ *records which coded claims are accepted by the candidate.*

Theorem 9461.22 (Structure-existence boundary)**.** *For every runtime candidate* $\mathsf{NameCert}_R$ *over* CIC**,* $\mathsf{NameCert}_R$ *remains an object-level derived interface. Its validation is not one of its own* $\mathsf{SourceSpec}_R$*,* $\mathsf{PatternSpec}_R$*,* $\mathsf{ClassifierSpec}_R$*,* $\mathsf{StabilityCert}_R$*, or* $\mathsf{LedgerPolicy}_R$ *fields. Consequently the isomorphism between* CIC* *and the host calculus closes only the structural interpretation of type formation, dependent function formation, inductive formation, and recursors; it does not identify* CIC* *with the external host kernel that checks BEDC proofs.*

Proof. By Definition 9461.11, the internal structure CIC* preserves the formation rules and recursors of the host calculus. These are structural data. By chapter 14, an evaluator, proof checker, or runtime is not a primitive in the finite-kernel chain, and by Definition 1.1 the explanatory notions of proof, model, and checking are not object-language primitives. Therefore any internal BEDC occurrence of such a runtime must enter as a derived interface, and by Definition 9.9 it must be licensed as a naming certificate of the form fixed in Definition 9.6.

Unfold such a certificate as $\mathsf{NameCert}_R = (\mathsf{SourceSpec}_R, \mathsf{PatternSpec}_R, \mathsf{ClassifierSpec}_R, \mathsf{StabilityCert}_R, \mathsf{LedgerPoli}$ Each field is object-level data: source rows, observed transition behavior, a classifier, stability witnesses, and a ledger. The judgment that these five fields are themselves a correct checker for BEDC is not one more source row or one more accepted output of the candidate; it is the validation of the certificate as a certificate. Placing that validating judgment into the ledger only produces a further coded claim whose own ledger correctness must again be validated. Thus internalizing a runtime candidate yields another BEDC object to be checked, not the checking act that licenses it.

The self-description of Definition 9461.12 changes only the input side of this argument: it supplies internal restatements of BEDC claims over CIC*. It does not turn the external host execution that accepts the manuscript's proofs into a field of the internal structure. If the ledger of $\mathsf{NameCert}_R$ were strengthened to return BEDC truth for every internally described BEDC sentence, then the self-description would compose with that ledger to give an internal BEDC truth predicate, which is exactly the obstruction recorded in Definition 9461.14. Hence CIC* exists as a structural copy of the host calculus, while the host runtime remains external. □

Definition 9461.23 (Truth-total reflection certificate)**.** *Let* $\mathsf{Sent}(\mathrm{CIC}^*)$ *be the BEDC-internal carrier of sentence codes supplied by the structural interpretation of Definition 9461.11. A* truth-total reflection certificate *is a naming certificate* $\mathsf{NameCert}_T$ *over* $\mathsf{Sent}(\mathrm{CIC}^*)$ *with an extension predicate* Ext_T *such that every sentence* φ *represented in* CIC* *satisfies*

$$\mathsf{Ext}_T(\ulcorner\varphi\urcorner) \Longleftrightarrow \varphi.$$

The left-to-right implication is internal soundness for the code, and the right-to-left implication is internal completeness for the code. The certificate is therefore stronger than the structural copy of Definition 9461.11: it does not only describe formation and recursor structure, but classifies the BEDC sentence represented by each code.

Theorem 9461.24 (Exact boundary of reflection)**.** *BEDC has a closed ground self-instantiation and no truth-total meta self-instantiation. Concretely, the ground side has exactly the two generated marks* $\mathsf{b0}$ *and* $\mathsf{b1}$*, exactly the two positive primitive mark-sameness judgments* $\mathsf{msame}(\mathsf{b0}, \mathsf{b0})$ *and* $\mathsf{msame}(\mathsf{b1}, \mathsf{b1})$*, and no generated cross-mark sameness. On the meta side, no truth-total reflection certificate of Definition 9461.23 exists for* CIC**.*

Proof. The ground claim is closed generation. By Definition 2.1, the only raw mark forms are $\mathsf{b0}$ and $\mathsf{b1}$. By Definition 2.3, the only primitive positive mark-sameness judgments are $\mathsf{msame}(\mathsf{b0}, \mathsf{b0})$

and $\mathsf{msame}(\mathsf{b1}, \mathsf{b1})$. By Definition 2.9, no generated mark-sameness judgment crosses the two constructors. Since Definition 1.1 and Definition 1.2 place no object-language mathematical primitive before Mark, and Definition 9.9 forces every later derived interface to enter through a $\mathsf{NameCert}$, this is the only closed primitive loop available at the ground.

For the meta claim, suppose that a truth-total reflection certificate $\mathsf{NameCert}_T$ exists. The structure CIC* of Definition 9461.11 supplies an internal structural copy of the host calculus, and Definition 9461.12 supplies internal restatements of formal claims over that structure. Hence the diagonal construction for the one-variable formula $\neg\mathsf{Ext}_T(x)$ is available at the structural level named by Definition 9461.14. Let δ be the represented sentence satisfying

$$\delta \iff \neg\mathsf{Ext}_T(\ulcorner\delta\urcorner).$$

Truth-totality of $\mathsf{NameCert}_T$ also gives

$$\mathsf{Ext}_T(\ulcorner\delta\urcorner) \iff \delta.$$

Combining the two equivalences yields

$$\mathsf{Ext}_T(\ulcorner\delta\urcorner) \iff \neg\mathsf{Ext}_T(\ulcorner\delta\urcorner).$$

The left-to-right direction turns any witness of $\mathsf{Ext}_T(\ulcorner\delta\urcorner)$ into a refutation of the same witness, so $\neg\mathsf{Ext}_T(\ulcorner\delta\urcorner)$ holds. Applying the right-to-left direction to this refutation gives a witness of $\mathsf{Ext}_T(\ulcorner\delta\urcorner)$, contradicting the refutation just obtained. Therefore $\mathsf{NameCert}_T$ cannot exist.

Thus the reflection boundary is exact: the constructor-level distinction closes at Mark, while structural self-description stops before an internal truth classifier. □

Definition 9461.25 (Gödel boundary surface)**.** *A* Gödel boundary surface *for a BEDC fragment F is a finite certificate interface over sentence codes, proof-checker rows, fixed-point rows, and continuation routes satisfying the following visibility clauses.*

1. *The fragment has a displayed ground distinction inherited from* Mark *and* Hist*.*
2. *The fragment has a structural self-description sufficient to code its own sentence and proof rows inside* CIC*.
3. *The fragment exposes a proof predicate or runtime predicate only through displayed* $\mathsf{NameCert}$*,* Ext*,* Cont*,* hsame*, and ledger rows.*
4. *The fragment contains no truth-total reflection certificate in the sense of Definition 9461.23 and no global halting predicate in the sense of Definition 9368.2.*

The surface is therefore not an oracle for truth, provability, or termination. It is the finite place where syntax, proof checking, fixed-point coding, and diagonal refusal are all visible at once.

Theorem 9461.26 (Gödel boundary as truth-total refusal)**.** *For any BEDC fragment F admitting the self-description clauses used in Definition 9461.12, the existence of a universal internal certificate classifying truth for all represented F-sentences would produce a truth-total reflection certificate of Definition 9461.23. Hence every such fragment has a Gödel boundary surface: it may display sentence codes, proof-checker rows, fixed-point rows, and finite obstruction ledgers, but it cannot export a universal internal classifier for truth, provability, certificate validity, or arbitrary halting behavior.*

Proof. Assume that F has a universal internal truth classifier U over its represented sentence codes. The self-description result Definition 9461.12 interprets the relevant formal claims of F inside CIC*, and the classifier U supplies the extension predicate that decides each represented sentence. Repacking U as $\mathsf{NameCert}_T$ gives exactly the data required by Definition 9461.23: for every represented sentence φ,

$$\mathsf{Ext}_T(\ulcorner\varphi\urcorner) \iff \varphi.$$

But Definition 9461.24 proves that no such $\mathsf{NameCert}_T$ exists. Therefore the universal internal truth classifier cannot exist.

The same obstruction transfers to provability, certificate validity, and halting whenever they are promoted from displayed finite ledgers to total internal classifiers over the certificate-of-certificates stratum. A total provability-validity classifier would make the fixed-point sentence decide its own proof status inside the same represented fragment. A total halting classifier would decide the Cont-behavior of arbitrary certificate-history pairs and would close the meta loop described in Definition 9368.4. In each case the attempted classifier occupies the same structural role as $\mathsf{NameCert}_T$: it turns self-description plus a diagonal row into a total meta-level decision. Since the reflection boundary refuses that role, the admissible object is only the finite boundary surface of Definition 9461.25. □

Corollary 9461.27 (Finite incompleteness packets are boundary witnesses)**.** *The packet surfaces* $\mathsf{TruthTotalReflection}^{\uparrow}$*,* $\mathsf{TarskiTruthRefusal}^{\uparrow}$*,* $\mathsf{HaltingObstruction}^{\uparrow}$*, and* $\mathsf{GoedelIncompleteness}^{\uparrow}$ *are finite boundary witnesses rather than escapes from incompleteness. They record sentence-code rows, attempted classifier rows, fixed-point or diagonal rows, finite trace rows, transport, provenance, and local naming; they do not provide an internal truth predicate, a total halting predicate, a proof-validity oracle, or a proof of the fragment's own global consistency.*

Proof. The $\mathsf{TruthTotalReflection}^{\uparrow}$ packet records sentence-code, extension-attempt, diagonal, transport, route, provenance, and naming rows; by Definition 9461.24 those rows cannot be read as a truth-total classifier. The $\mathsf{TarskiTruthRefusal}^{\uparrow}$ packet records the narrower refusal row for an attempted truth query and its diagonal pressure, so it witnesses the Tarski side of the same boundary. The $\mathsf{HaltingObstruction}^{\uparrow}$ packet records finite traces, self-reference rows, diagonal policies, and provenance; by Definition 9368.4, turning those rows into a total halting decision would close the forbidden meta loop. The $\mathsf{GoedelIncompleteness}^{\uparrow}$ packet records a recursively enumerated axiom row, proof-checker decoder, fixed-point row, and conditional no-proof plus no-refutation ledger; its closurestatus explicitly keeps semantic completeness, second incompleteness, ω-consistency, and set-theoretic foundation bridges outside its claim surface.

Thus all four packets have the same logical shape. They make the obstruction auditable as finite BEDC data, while the total classifier that would erase the obstruction remains unavailable. □

Concretized in §472: Truth-total reflection is represented by a finite boundary certificate over CIC-star sentence codes and diagonal obstruction rows.

Concretized in §288: Goedel incompleteness is realized as a finite boundary packet over arithmetised provability, fixed-point rows, and conditional undecidability ledgers.

Concretized in §470: This chapter concretizes the reflection-limit boundary as a countable-name certificate whose runtime and truth rows remain external to the finite stream packet.

Concretized in §586: Reflection-limit boundaries are realized as finite BHist NameCert packets separating closed ground-loop rows from open meta-loop truth rows.

Concretized in §591: Runtime candidates over CIC-star are realized as finite NameCert boundary packets separating object-level transition rows from host validation.

Concretized in §818: Finite trace evaluators realize runtime candidates as bounded object-level trace packets whose validation remains outside the packet.

Concretized in §997: finite runtime checkpoint packet for reflection-limit validation boundaries

Concretized in §674: Ground-loop boundary packets realize the closed Mark-level distinction and the open meta-truth refusal as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Concretized in §890: Runtime reflection boundaries are realized as finite packets separating structural self-description from external kernel execution.

Concretized in §1028: Runtime/truth separation is realized as a finite NameCert packet separating structural CIC-star rows from external runtime and truth-validation rows.

Concretized in §6540: ArithmetizedProvabilityUp realizes the Godel-numbering and proof-predicate dependency as finite proof-code, formula-code, derivation-check, substitution, transport, replay, provenance, and NameCert rows.

Concretized in §7390: ProofTraceCompactnessUp realizes the finite proof-trace compact replay route as NameCert data.

Concretized in §5122: Legitimate claim audit ledgers realize the meta-foundations claim-discipline route as finite refusal, proof-target, audit, provenance, and NameCert rows.

Chapter 9462

Auditable Generativity: A BEDC Philosophy

This philosophy directory records the structural reading of BEDC as a theory of generated, classified, ledgered, and certified names. The chapters below treat philosophical claims as disciplined readings of kernel signatures, certificate interfaces, and explicit boundary tags, without presenting them as additional Lean-checked targets.

9462.1 Distinction and History

BEDC philosophy is the disciplined reading of the following generative chain:

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{hsame} \to \mathsf{Ext/Cont} \to \mathsf{Ask/Sig} \to \mathsf{Pkg/Gap} \to \mathsf{NameCert} \to \text{mature interface.}$$

The reading does not add a metaphysics to the finite kernel. It asks which entities may be called objects, knowledge, observation, subject, time, world, causality, consciousness, or boundary, once every such claim is forced to pass through generated histories, licensed classifiers, witness ledgers, and naming certificates.

Definition 9462.1 (Auditable generativity)**.** *Auditable generativity is the position that every mature object is a licensed name rather than a primitive, every item of knowledge is a traceable witness rather than bare possession of truth, every observer is a history rather than a hidden substance, every time parameter is history growth rather than an external container, and every commitment exceeding a closed substrate must be exposed as a* Gap *or boundary position rather than consumed as an internal theorem.*

Theorem 9462.2 (Philosophy is not an additional layer)**.** *The philosophical reading of BEDC is a consequence of the kernel signature and certificate discipline, not an auxiliary doctrine placed above them.*

Proof. The continuation relation has only history-valued endpoints, the kernel identity criterion for histories is internal history-sameness, and the forward direction of continuation is fixed by the constructive formation of histories. A separate observer-substance, a global truth predicate, or an unledgered mature object would require a carrier or transport not supplied by the kernel. The philosophical reading is therefore obtained by spelling out which commitments are already encoded by the available signatures and which commitments must remain outside the object layer as exposed boundary positions. □

9462.1.1 Primitive distinction

Definition 9462.3 (Primitive marks)**.** *The primitive marks of BEDC are two emission results, denoted* $\mathsf{b0} : \mathsf{Mark}$ *and* $\mathsf{b1} : \mathsf{Mark}$*. They are not natural numbers, truth values, Boolean objects, or elements of a presupposed set. They are only the minimal object-language marks from which distinguishable emission histories may be generated.*

Definition 9462.4 (Primitive distinction)**.** *The primitive distinction is the closed mark-level separation between* $\mathsf{b0}$ *and* $\mathsf{b1}$*: the internal sameness relation generates the diagonal judgments for each mark and does not generate a cross-mark sameness judgment. Philosophically, the point is not that two prior objects are given, but that distinguishability itself is the lowest condition of later object formation.*

Theorem 9462.5 (Primitive distinction is irreducible)**.** *The primitive distinction cannot be reduced inside BEDC to a simpler internal structure.*

Proof. Any proposed reduction would have to say how a lower structure distinguishes inputs from outputs, success from failure, and the same case from a different case. That act of discrimination already presupposes a distinction. Iterating the reduction therefore either never starts or accepts a lowest closed discrimination. BEDC takes the mark-level constructor separation as that closed ground loop. □

Corollary 9462.6 (Distinction precedes information)**.** *Information is not prior to distinction in BEDC.*

Proof. Information requires that one state, mark, or trace be separable from another. Without the mark-level separation there is no separation between the two history constructors, hence no recorded sequence, no classifier output, and no ledgered information. Distinction is therefore a precondition for information rather than a species of information. □

Corollary 9462.7 (The starting point is not a thing)**.** *The ontological starting point of BEDC is not an object, set, subject, or proposition, but recordable distinguishability.*

Proof. Objects require identity conditions. Identity conditions require a contrast between sameness and difference. The mark layer supplies that contrast before mature objects are licensed. Hence objects arise only after distinction has entered histories and certificates. □

9462.1.2 Mark sameness

Definition 9462.8 (Internal mark sameness)**.** *The relation* $\mathsf{msame}(m, n)$ *is the object-language sameness judgment for marks. It is generated by the two diagonal cases for* $\mathsf{b0}$ *and* $\mathsf{b1}$*, with no constructor producing* $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *or* $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$*.*

Definition 9462.9 (Internal sameness is not host equality)**.** *The relation* msame *is an internal BEDC judgment. It is not host definitional equality, not set-theoretic equality, and not an unrestricted external equality oracle.*

Theorem 9462.10 (Mark sameness is reflexive)**.** *For every generated mark* m*, the judgment* $\mathsf{msame}(m, m)$ *is generated.*

Proof. The closed mark carrier has only the two generated marks. In the $\mathsf{b0}$ case the corresponding diagonal rule gives $\mathsf{msame}(\mathsf{b0}, \mathsf{b0})$; in the $\mathsf{b1}$ case the corresponding diagonal rule gives $\mathsf{msame}(\mathsf{b1}, \mathsf{b1})$. □

Theorem 9462.11 (Mark sameness is symmetric and transitive)**.** *The internal mark-sameness judgment is symmetric and transitive.*

Proof. Every generated mark-sameness derivation is diagonal. Reversing a diagonal judgment gives the same diagonal judgment, so symmetry holds. Composing two generated judgments through a middle mark can only remain on the same diagonal, so transitivity holds. □

Theorem 9462.12 (Mark no-confusion)**.** *BEDC does not generate a cross-mark sameness judgment between* b0 *and* b1*.*

Proof. The generation rules for msame contain only the diagonal judgments. There is no cross-mark constructor and no closure rule that converts a diagonal judgment into a cross-mark judgment. □

Corollary 9462.13 (Primitive marks are not completed Booleans)**.** *The primitive marks must not be read as an already available Boolean interface.*

Proof. A Boolean interface is a mature data interface with its own carrier, classifier, stability behavior, and naming certificate. The primitive marks are earlier than such an interface. Reading them directly as Booleans would import a mature object at the ground layer instead of deriving it through the certificate discipline. □

Concretized in §1244: Primitive distinction ledgers realize mark-level no-confusion, trace exposure, and anti-Boolean refusal as one finite NameCert packet.

9462.1.3 History formation

Definition 9462.14 (Histories)**.** *The carrier* Hist *is generated by the empty history* e *and the two extension constructors* E0 *and* E1*. The public reading is that* e *records no distinction, while* E0(h) *and* E1(h) *record one additional mark-shaped emission above* h*.*

Definition 9462.15 (Information content of a history)**.** *The information content of a history is the finite distinction trace recovered by recursion on its generated form. The empty history has the empty trace; an* E0*-extension appends the* b0*-shaped tag; an* E1*-extension appends the* b1*-shaped tag. These tags are metalinguistic readings of constructor shape, not imported natural numbers or Boolean values.*

Theorem 9462.16 (Histories are finitely generated)**.** *Every generated history is obtained from* e *by finitely many applications of* E0 *and* E1*.*

Proof. The closed formation rules for Hist provide only the empty constructor and the two extension constructors. Following the constructor tree of any generated history downward therefore reaches e after finitely many predecessor steps. □

Corollary 9462.17 (No external history in the kernel)**.** *There is no kernel history not generated by* e*,* E0*, and* E1*.*

Proof. An external history would require an additional constructor or an additional import rule. Neither is part of the closed kernel carrier. □

Theorem 9462.18 (The empty history is a zero-information boundary)**.** *The empty history records no distinction.*

Proof. By the recursion defining history information content, e is the base case and contributes no constructor tag. It is therefore the boundary at which no distinction has yet entered a history. □

Corollary 9462.19 (Nontrivial information requires extension)**.** *Any history with nonempty information content has the form* $\mathsf{E0}(k)$ *or* $\mathsf{E1}(k)$ *for some history* k*.*

Proof. If a history is e, its information content is empty. Every generated nonempty history must therefore arise by one of the two extension constructors. □

9462.1.4 History sameness

Definition 9462.20 (Internal history sameness)**.** *The judgment* $\mathsf{hsame}(h, k)$ *is the object-language sameness relation for histories. It is generated by the empty diagonal case and by constructor-respecting closure through* E0 *and* E1*. It has no generation rule comparing an empty history with an extension, or an* E0*-extension with an* E1*-extension.*

Definition 9462.21 (History identity)**.** *The identity of a history, inside BEDC, is its internal history sameness profile. It is not supplied by a host equality oracle.*

Theorem 9462.22 (History sameness is reflexive)**.** *For every generated history* h*, the judgment* $\mathsf{hsame}(h, h)$ *is generated.*

Proof. Induct on the generated form of h. The empty case is the empty diagonal rule. The E0 and E1 cases follow by applying the corresponding constructor-respecting closure to the induction hypothesis. □

Theorem 9462.23 (History sameness is symmetric and transitive)**.** *The internal history-sameness judgment is symmetric and transitive.*

Proof. Symmetry follows by induction on the generated sameness derivation: the empty case reverses to itself, and each constructor-respecting case uses the induction hypothesis below the same constructor. Transitivity follows by induction on one sameness derivation and inversion on the other; the closed generation rules permit only matching constructors, after which the induction hypothesis composes the predecessor judgments. □

Theorem 9462.24 (History no-confusion)**.** *The kernel does not generate history-sameness judgments between* e *and an extension, nor between an* E0*-extension and an* E1*-extension.*

Proof. The generation rules for hsame contain only the empty diagonal, the E0-respecting closure, and the E1-respecting closure. None of these generates a cross-shape judgment. □

Corollary 9462.25 (History identity is constructor-shape identity)**.** *An internal sameness judgment between two* E0*-extensions must come from an internal sameness judgment between their predecessors, and similarly for two* E1*-extensions.*

Proof. This is the inversion principle for the two constructor-respecting generation rules of hsame. □

9462.1.5 Minimal information carrier

Definition 9462.26 (Binary-distinction information carrier)**.** *A binary-distinction information carrier is a carrier that represents two different step shapes and finite traces assembled from those steps.*

Theorem 9462.27 (Hist is the minimal binary-distinction carrier)**.** *The carrier* Hist *is the minimal BEDC carrier for binary-distinction information.*

Proof. Hist has exactly an empty origin and two extension shapes, so it can record every finite trace of two mark-shaped distinctions. Removing the empty origin leaves no start for trace formation. Removing either extension shape destroys binary distinction. Identifying the two extension shapes violates history no-confusion and erases the difference to be recorded. Thus any weaker carrier fails to represent binary-distinction information. □

Corollary 9462.28 (Information requires history)**.** *Every nontrivial item of information available to the BEDC object layer must leave a trace in a history or in a derived ledger certified from histories.*

Proof. Nontrivial information requires a distinction, and a usable distinction must enter a generated trace or a certified structure derived from such a trace. A classifier, signature, or naming certificate may organize the trace, but it does not bypass the historical condition. □

Corollary 9462.29 (No bare information)**.** *BEDC does not accept information that is unhistoried, unwitnessed, and unledgered.*

Proof. If the purported information is usable in the object layer, it must be distinguishable from its absence or from a different information item. That distinction must be recorded or certified. If it enters neither a history nor a derived ledger, the object layer has no licensed way to consume it. □

9462.1.6 Prefix order and the preparatory form of time

Definition 9462.30 (Distinction step)**.** *A distinction step is a single application of* E0 *or* E1 *to a history.*

Definition 9462.31 (History prefix)**.** *A history p is a prefix of a history h when h is obtained from p by a finite trace of distinction steps.*

Concretized in §1241: History prefixes are isolated as finite BHist prefix certificates feeding countable-name observations.

Theorem 9462.32 (History generation carries a prefix order)**.** *Every nonempty generated history has a direct predecessor shape, and that predecessor is its immediate prefix.*

Proof. By closed generation, a nonempty history is either an E0- or E1-extension of a predecessor. History no-confusion prevents the two shapes from being internally identified and prevents either shape from being internally identified with e. The outer constructor therefore determines the direct predecessor shape. □

Corollary 9462.33 (Histories are not sets)**.** *Histories are generated traces, not sets of already available elements.*

Proof. Set-theoretic reading begins from a domain of elements and membership. BEDC begins from a generated empty history and constructor growth. History identity is governed by constructor shape and internal hsame, not by external membership. □

Corollary 9462.34 (The lowest form of time is history growth)**.** *Once a distinction step is read as history growth, the lowest form of time is already present.*

Proof. Time minimally requires before and after. In the kernel, before and after first appear as the generated movement from a predecessor history to an E0- or E1-extension. This gives the preparatory form later made explicit by the continuation relation. □

9462.1.7 Extension, continuation, and unfolding

Definition 9462.35 (Extension step)**.** *An extension step is a generated relation instance*

$$\mathsf{Ext}(\mathsf{b0}, h, \mathsf{E0}(h)) \qquad \textit{or} \qquad \mathsf{Ext}(\mathsf{b1}, h, \mathsf{E1}(h)).$$

It records one visible mark-shaped passage from a source history to a result history. It is read as a relation instance, not as an imported primitive function on histories.

Definition 9462.36 (Continuation trace)**.** *A continuation trace is a finite relation witness built from the empty continuation and the visible* b0*- and* b1*-steps. It records a finite route from one history to another through generated history growth.*

Theorem 9462.37 (Extension is singleton continuation)**.** *Each extension step is the one-step case of continuation.*

Proof. An extension by b0 produces the same result shape as the continuation step whose visible tail is the singleton E0(e). The b1 case is identical with E1(e). Thus extension supplies the visible one-step boundary, while continuation iterates the same generated passage. □

Definition 9462.38 (Record unfolding)**.** *A record unfolding is a passage from a finite record of histories to a larger finite record obtained by adding histories reachable from the displayed record through extension or continuation traces, while keeping the carrier, classifier, certificate fields, and ledger policy fixed.*

Definition 9462.39 (Classifier shift)**.** *A classifier shift occurs when the public classifier surface changes: the admitted source rows, classifier relation, stability witness, ledger policy, or naming certificate no longer has the same extension as before the passage under consideration.*

Theorem 9462.40 (Unfolding does not by itself shift a classifier)**.** *Record unfolding can lengthen the visible history record without changing the classifier surface.*

Proof. By Definition 9462.38, the passage adds only histories already reachable by the fixed extension or continuation relations. The carrier, classifier, stability witness, ledger policy, and naming certificate are held fixed. Therefore the passage supplies more displayed route material but does not by itself alter the public classifier surface. □

Corollary 9462.41 (History growth is not theoretical discovery)**.** *Increasing the length of a history record is not, by itself, a theoretical discovery.*

Proof. A theoretical discovery in this reading requires a classifier shift or a certificate-strengthening row. By Definition 9462.40, record unfolding can increase visible route material while preserving the same classifier surface. Hence history growth alone is insufficient. □

Corollary 9462.42 (Verified unfolding remains an unfolding fact)**.** *Formal verification of an unfolding theorem hardens the stated route claim; it does not convert fixed-classifier computation into classifier shift.*

Proof. A formal verifier accepts a proof object for a formal statement about the displayed route. If the statement concerns only record growth under the fixed carrier, classifier, stability witness, and ledger policy of Definition 9462.38, the accepted proof hardens that route statement without changing the classifier surface. □

9462.1.8 Synthesis

Definition 9462.43 (Distinction-history chain)**.** *The distinction-history chain is the path from primitive mark separation, through internal mark sameness and no-confusion, to generated histories, internal history sameness, and recoverable information content.*

Theorem 9462.44 (Distinction generates history, history carries information)**.** *In BEDC, the lowest auditable form of information is a finite history generated from primitive distinction.*

Proof. Primitive marks supply the first separation. Internal mark sameness and mark no-confusion provide the object-language identity profile for that separation. The constructors `e`, `E0`, and `E1` record the separation into finite histories. Internal history sameness and history no-confusion preserve the generated shape of those records. History information content is then recovered as the finite trace of recorded distinctions. Therefore the first auditable information in the object layer is generated history. □

Corollary 9462.45 (Ontology begins with recordable distinction)**.** *The first ontological commitment of BEDC is not that things exist, but that distinctions can be recorded.*

Proof. Mature objects require naming certificates; naming certificates require source, pattern, classifier, stability, and ledger fields; those fields ultimately depend on generated histories and the distinction structure that histories record. Hence object talk is downstream from recordable distinction. □

Corollary 9462.46 (Passage to continuation)**.** *After distinction enters history, the next philosophical question is continuation, because a history is not merely a static trace but a trace capable of generated extension.*

Proof. The history carrier records distinction. To account for observation, time, action, and object formation, the framework must also account for the passage from one history to another. That passage is exactly the role of `Cont` at the next layer. □

9462.2 Classifier-Shift Information

Concretized in §6985: The classifier-shift ledger residue packet realizes before and after classifier disagreement, source overlap, residue ledger, and discovery-status rows as finite NameCert data.

Concretized in §8519: Classifier-shift discovery gates are realized as finite before/after classifier, witness-flow, semantic-certificate, weight-profile, refusal, transport, replay, provenance, and NameCert rows.

The preceding distinction-history layer separates generated record growth from classifier change. This section names the corresponding information principle: information growth in the auditable-generativity reading is not the mere accumulation of outputs, traces, activations, or notational variants. It is a certified change in the classifier surface under which generated material is organized.

Definition 9462.47 (Certified classifier state)**.** *A certified classifier state is a tuple*

$$\mathcal{T} = (S_{\mathcal{T}}, P_{\mathcal{T}}, L_{\mathcal{T}}, C_{\mathcal{T}}, N_{\mathcal{T}})$$

where $S_{\mathcal{T}}$ is the source specification, $P_{\mathcal{T}}$ is the pattern specification, $L_{\mathcal{T}}$ is the ledger policy, $C_{\mathcal{T}}$ is the classifier relation on the source material, and $N_{\mathcal{T}}$ is the naming certificate or semantic certificate licensing the state.

Definition 9462.48 (Classifier equivalence)**.** *Two certified classifier states $\mathcal{T}$ and $\mathcal{T}'$ are classifier-equivalent on their common source when every pair of histories admitted by both source specifications receives the same classifier judgment from $C_{\mathcal{T}}$ and $C_{\mathcal{T}'}$.*

Definition 9462.49 (Classifier shift)**.** *A passage $\mathcal{T} \to \mathcal{T}'$ has classifier shift when the two states are not classifier-equivalent on their common source. The shift may appear as refinement, coarsening, or reorganization of the classifier cells, but it must be witnessed on the public classifier surface rather than inferred from record length, notation, or fluency.*

Definition 9462.50 (Structural discovery)**.** *A structural discovery is a passage to a certified classifier state $\mathcal{T}'$ such that the passage has classifier shift, the target state has a ledger witness for its admitted material, and the changed classifier is actually used on the displayed record or certificate surface.*

Theorem 9462.51 (Discovery requires classifier shift)**.** *Every structural discovery has classifier shift.*

Proof. Classifier shift is one of the defining fields of Definition 9462.50. A passage that preserves the same classifier on the common source may improve presentation, coverage of displayed routes, or verification status, but it has not changed the classifier surface that organizes the material. □

Theorem 9462.52 (Discovery requires certificate and ledger)**.** *Every structural discovery carries both a certificate for the resulting classifier state and a ledger witness for the material it classifies.*

Proof. The target of a structural discovery is a certified classifier state, so its tuple contains the certificate component $N_{\mathcal{T}'}$. The definition also requires a ledger witness. Without the certificate, the changed classifier is only an asserted relation; without the ledger, the source, hidden material, projection, compression, and boundary behavior are not auditable. □

Corollary 9462.53 (Record growth is not discovery)**.** *A record-only passage is not a structural discovery.*

Proof. A record-only passage preserves the certified classifier state while adding more displayed material under that state. By Definition 9462.51, structural discovery requires classifier shift, so record growth alone is insufficient. □

Corollary 9462.54 (Presentation change is not discovery)**.** *Changing notation, proof organization, naming style, or exposition is not by itself a structural discovery.*

Proof. Presentation changes can alter how a state is read without changing the source specification, classifier relation, certificate fields, or ledger policy. Such a passage is classifier-equivalent on the common source and therefore lacks classifier shift. □

Definition 9462.55 (Boundary, bridge, and unification discovery)**.** *Classifier-shift discovery has three recurring modes. Boundary discovery exposes a principled obstruction to strengthening a classifier. Bridge discovery connects previously separated classifier states through a certified relation. Unification discovery replaces several classifier states by a single certified state whose classifier surface accounts for their shared behavior.*

Theorem 9462.56 (Formal hardening is not classifier shift)**.** *Formal verification of a statement or certificate component does not by itself create classifier shift.*

Proof. Formal verification changes the audit status of a statement or proof object. If the source specification, classifier relation, certificate fields, and ledger policy remain the same, the certified classifier state is classifier-equivalent to its prior state on the common source. The verification may increase reliability, but the classifier surface has not changed. □

Theorem 9462.57 (Classifier-shift information principle)**.** *In the auditable-generativity reading, theoretical information growth is read from certified classifier shift rather than from output length, record count, feature count, notation change, or verification status alone.*

Proof. Output length and record count are governed by record unfolding, which can preserve the same classifier. Notation and presentation changes can preserve classifier equivalence. Formal hardening can preserve the same classifier while changing only audit status. By contrast, structural discovery requires a changed classifier together with certificate and ledger witnesses. Hence the auditable unit of theoretical information growth is certified classifier shift. □

9462.2.1 Compression, ledger, and conservation

Definition 9462.58 (Compression map)**.** *For a finite family of histories X, a compression map is a displayed map $q : X \to Y$ into a representation carrier Y. The map induces the classifier*

$$x \sim_q x' \quad \Longleftrightarrow \quad q(x) = q(x').$$

Definition 9462.59 (Hidden residue)**.** *The hidden residue of a compression map $q : X \to Y$ is the family of distinction pairs collapsed by the representation:*

$$\mathsf{Res}(q) = \{(x, x') \in X \times X : \ x \neq x' \text{ and } q(x) = q(x')\}.$$

Definition 9462.60 (Ledger-complete compression)**.** *A compression map is ledger-complete when it carries a ledger Λ_q whose recorded rows cover the hidden residue of q. A ledger gap is a collapsed distinction in $\mathsf{Res}(q)$ not recorded by Λ_q.*

Definition 9462.61 (Certified compression)**.** *A certified compression is a compression map equipped with source, pattern, compression-induced classifier, stability, and ledger rows sufficient to license the compressed representation as a certificate surface.*

Theorem 9462.62 (Compression is classifier formation)**.** *Every compression map induces a classifier on its source family.*

Proof. The relation $\sim_q$ is reflexive, symmetric, and transitive because it is defined by equality of the displayed representation $q(x)$. Thus the compression map partitions its source family into representation cells. □

Theorem 9462.63 (Certified compression requires ledger coverage)**.** *A certified compression has no unrecorded hidden residue.*

Proof. By Definition 9462.61, the compression certificate includes a ledger row. By Definition 9462.60, the ledger row must cover every collapsed distinction in $\mathsf{Res}(q)$. Any unrecorded collapsed distinction would be a ledger gap, so the compression would not be certified. □

Corollary 9462.64 (Shorter presentation is not certified gain)**.** *A shorter representation is not by itself a certified information gain.*

Proof. Shortening can be achieved by collapsing distinctions. By Definition 9462.63, a certified compression must account for the collapsed distinctions through its ledger. The representation length alone therefore does not determine the certified gain. □

Theorem 9462.65 (Information conservation for compression)**.** *The auditable gain of a compression claim is read only after subtracting ledger cost, certificate cost, verification cost when claimed, and ledger debt for unrecorded residue from the apparent description gain.*

Proof. Apparent description gain records only the shorter representation. A certificate must also expose source, pattern, classifier, stability, and ledger rows. When a formal verification claim is made, the proof-checking work is part of the audit cost. When hidden residue is not recorded, the claim carries ledger debt. Omitting any of these terms would treat hidden material as free information, contrary to the certificate discipline. □

Corollary 9462.66 (Compression discovery requires classifier shift)**.** *A compression is a structural discovery only when it changes the certified classifier surface and supplies the corresponding certificate and ledger rows.*

Proof. Same-classifier compression may reduce description cost, but it remains classifier-equivalent on the common source. By Definition 9462.51, structural discovery requires classifier shift, and by Definition 9462.52, it also requires certificate and ledger witnesses. □

9462.2.2 Machine-learning reading

For a learned model, additional activations, candidate features, lower training loss, or more examples do not automatically constitute interpretability discovery. They become interpretability-relevant only when they change a certified classifier state: the activation source, the feature pattern, the induced classifier, the stability account, and the ledger must be displayed. Dimensionality reduction, feature clustering, pruning, and model compression are likewise compression maps: they must record hidden residue before they can count as closed explanations. This is the philosophy-side counterpart of section 9398.2: a learned tensor distinction can propose a classifier, but interpretability information is certified only when that classifier is named, stabilized, ledgered, and auditable.

Concretized in §6241: Classifier-novelty ledgers realize classifier-shift information by finite source, classifier, changed-cell witness, ledger, transport, provenance, and NameCert rows.

9462.3 Distinction-to-NameCert Certification

The distinction-history and classifier-shift layers identify the route from generated marks to certified classifier states. This section names the mediating certification layer. BEDC does not treat a stable distinction, a feature label, a proof trace, or a model behavior as an auditable object merely because it is visible. The object is licensed only when the candidate classifier is carried by a NameCert surface with source, pattern, classifier, stability, and ledger rows.

Definition 9462.67 (Generated material)**.** *Generated material is any finite record produced by the mark-history, event-flow, proof-trace, tensor-trace, or activation-trace interfaces. It is material for certification, not a certified object by itself.*

Definition 9462.68 (Candidate classifier)**.** *A candidate classifier on a generated-material family G is a displayed relation C on G that proposes which records are to be read as the same for the purpose at hand. It may arise from history sameness, continuation behavior, compression cells, tensor activation patterns, neural features, empirical clusters, proof traces, or certificate structure.*

Definition 9462.69 (Certified classifier object)**.** *A certified classifier object is a pair (C, N) where C is a candidate classifier and N is a NameCert for C. The certificate has the five-field form*

$$N = (S_C, P_C, C, \Sigma_C, L_C),$$

where S_C is the source specification, P_C is the pattern specification, Σ_C is the stability certificate, and L_C is the ledger policy.

Theorem 9462.70 (Candidate classifiers are not certified objects)**.** *A candidate classifier without a NameCert is not a certified classifier object.*

Proof. By Definition 9462.69, a certified classifier object is a pair (C, N) with a NameCert component N. A bare candidate classifier supplies only the relation C. It does not supply source, pattern, stability, or ledger rows, so it does not satisfy the defining form of a certified classifier object. □

Corollary 9462.71 (Stable behavior is not objecthood)**.** *Stable behavior in histories, tensor traces, activations, or proof records is not by itself a BEDC object.*

Proof. Stable behavior may support a candidate classifier, but Definition 9462.70 requires a NameCert before that classifier has certified object status. □

Theorem 9462.72 (NameCert licenses classifier objecthood)**.** *If a candidate classifier C carries a NameCert N, then (C, N) is a certified classifier object.*

Proof. The conclusion is exactly the construction named in Definition 9462.69. The NameCert provides the rows that make the classifier auditable: source, pattern, classifier, stability, and ledger. □

Theorem 9462.73 (No ledger, no closed certificate)**.** *A certificate surface without a ledger policy is not a closed NameCert.*

Proof. The ledger policy is one of the five required rows in Definition 9462.69. Without it, hidden material, projection residue, compression residue, failure cases, and not-claimed boundaries have no audit row. The certificate surface is therefore incomplete. □

Corollary 9462.74 (Single-field claims are not NameCerts)**.** *Source-only, pattern-only, classifier-only, stability-only, and ledger-only claims are not NameCerts.*

Proof. Each such claim omits at least one required field from the five-field certificate surface. A source row without a classifier does not say what relation is licensed; a pattern row without a source does not say where it applies; a classifier row without stability does not say how it is transported; a stability row without ledger does not expose what it hides; and a ledger row without classifier does not identify the relation being audited. □

9462.3.1 Semantic Assignment

Definition 9462.75 (Semantic assignment)**.** *A semantic assignment maps classifier cells or public marks to an external meaning space, such as truth values, numbers, concept labels, model behaviors, or mathematical objects.*

Definition 9462.76 (Certified semantic assignment)**.** *A semantic assignment is certified when the assignment itself carries a NameCert specifying its source, pattern, classifier, stability, and ledger rows.*

Theorem 9462.77 (Semantic readings require certification)**.** *A semantic reading of a mark, history, tensor feature, or classifier cell is not licensed by the reading alone.*

Proof. The reading supplies an external label or interpretation. It does not thereby expose the source on which the reading acts, the pattern that supports it, the classifier relation it respects, the stability witness under transport or perturbation, or the ledger of hidden and excluded material. Those rows are supplied only by a certificate. □

Corollary 9462.78 (Feature labels are not explanations)**.** *Calling a neural feature by a concept label is not yet a certified explanation.*

Proof. The label is a semantic assignment. By Definition 9462.77, the assignment must be certified before it can serve as an auditable explanation. □

Theorem 9462.79 (Meaning is certificate-derived)**.** *Within the BEDC certificate reading, meaning is derived from certified semantic assignment rather than present as a primitive mark property.*

Proof. The primitive display contains marks, histories, generated material, and candidate distinctions. External meaning enters through semantic assignment. Since a semantic assignment is licensed only through Definition 9462.76, certified meaning is a derived certificate object. □

9462.3.2 Hardening

Definition 9462.80 (Certificate hardening)**.** *Certificate hardening is the passage in which one or more certificate rows are supported by a stronger evidence form, such as a formal proof object accepted by a verification backend, while the classifier surface being certified remains identified.*

Theorem 9462.81 (Hardening does not create the NameCert layer)**.** *Formal hardening can strengthen certificate rows, but it does not by itself create a complete NameCert.*

Proof. A verifier accepts a proof object for a translated formal statement. That acceptance may harden a source, classifier, stability, or boundary claim. It does not automatically supply the remaining NameCert rows. The certificate layer therefore remains distinct from the verification layer. □

Corollary 9462.82 (Verified components are not whole explanations)**.** *A verified statement inside an explanation is not the same as a certified explanation.*

Proof. By Definition 9462.81, verification hardens the component it checks. A certified explanation still requires the complete source, pattern, classifier, stability, and ledger surface. □

Theorem 9462.83 (Certificate hardening is not classifier shift)**.** *Certificate hardening does not by itself constitute classifier shift.*

Proof. Hardening changes the evidence status of one or more rows. If the classifier relation on the common source remains the same, the public classifier surface has not shifted. This is the certificate-layer form of Definition 9462.56. □

Theorem 9462.84 (Certification-layer theorem)**.** *BEDC forms a certification layer between generated behavior and optional formal hardening: candidate distinctions are organized as candidate classifiers, NameCert rows license them as certified classifier objects, and verification hardens only the translated rows it actually checks.*

Proof. Candidate distinctions and generated behavior supply material for classifiers by Definition 9462.68. Certified classifier objecthood requires the NameCert pair of Definition 9462.69. Formal hardening is separate by Definition 9462.81. Thus the certification layer is neither raw behavior nor the formal verification backend; it is the auditable NameCert surface between them. □

*Concretized in §**??**:* Distinction-to-NameCert certification realizes candidate classifiers, five-field NameCert rows, semantic-assignment licensing, hardening separation, and no-ledger refusal as finite certificate-layer packets.

Concretized in §7126: A finite distinction packet is certified as a NameCert boundary.

9462.4 Tensor NameCert

section 9462.3 describes the abstract passage from candidate classifiers to certified classifier objects. The tensor case is the machine-learning instance of that passage. A feature, direction, probe, sparse-autoencoder latent, attention pattern, or circuit score may expose a stable distinction, but it becomes an auditable explanation only when the induced classifier is licensed by a Tensor NameCert.

Definition 9462.85 (Tensor model context)**.** *A tensor model context consists of a model state, an input domain, an output interface, a displayed source domain, an allowed transformation family, and an observation protocol specifying which activation sites, hooks, data selections, and trace records are admitted.*

Definition 9462.86 (Activation source)**.** *An activation source is a specified layer, site, and recording mode together with the induced map*

$$a_A : X \to V_A.$$

Examples include residual-stream positions, attention-head outputs, MLP outputs, probe inputs, and sparse-autoencoder latent spaces, provided the observation protocol states how they are read.

Definition 9462.87 (Tensor classifier candidate)**.** *A tensor classifier candidate is a tuple* $C = (A, \phi, \tau, D)$ *where* A *is an activation source,* $\phi : V_A \to [0, 1]$ *is a feature functional,* $0 < \tau < 1$ *is a threshold, and* D *is the displayed source domain. It induces the relation*

$$x \sim_C y \quad \Longleftrightarrow \quad m_{\phi,\tau}(x) = m_{\phi,\tau}(y),$$

where $m_{\phi,\tau}(x)$ *is the thresholded public mark read from* $\phi(a_A(x))$*.*

Definition 9462.88 (Behavior support)**.** *For a behavior target* B*, a tensor classifier candidate* C *supports* B *when the classifier supplies a stated predictive, correlational, interventional, causal, formal, or hybrid account of that behavior. The support type is part of the certificate surface.*

Definition 9462.89 (Tensor NameCert)**.** *A Tensor NameCert for* $C = (A, \phi, \tau, D)$ *is the five-field packet*

$$\mathsf{NameCert}_{\text{tensor}}(C) = (S_C, P_C, \mathsf{ClassifierSpec}_C, \Sigma_C, L_C).$$

Here S_C *records model identity, activation source, source domain, trace protocol, feature source, threshold protocol, and behavior scope;* P_C *records positive, negative, boundary, margin, clustering, circuit, or soft-formula patterns;* $\mathsf{ClassifierSpec}_C$ *is the classifier relation induced by the thresholded marks;* Σ_C *records stability and transfer evidence; and* L_C *records threshold debt, boundary cases, false positives, false negatives, entanglement, untested domain, proxy risk, source leakage, and not-claimed behavior.*

Definition 9462.90 (Tensor-certified explanation)**.** *A tensor-certified explanation is a tuple*

$$(C, \mathsf{NameCert}_{\text{tensor}}(C), B)$$

where C *supports the behavior target* B *and the Tensor NameCert is displayed.*

Theorem 9462.91 (Tensor NameCert instantiates NameCert)**.** *Every Tensor NameCert is an instance of the BEDC NameCert surface.*

Proof. The abstract surface of Definition 9462.69 has source, pattern, classifier, stability, and ledger rows. The Tensor NameCert of Definition 9462.89 supplies exactly those rows in the tensor-model context. Thus the tensor packet is a machine-learning instance of the same certificate form. □

Theorem 9462.92 (Feature functionals are not explanations)**.** *A tensor feature functional* $\phi : V_A \to [0, 1]$ *is not by itself a tensor-certified explanation.*

Proof. The functional supplies only a score map. It does not identify the activation source, threshold, source domain, behavior support, classifier relation, stability witness, or ledger policy. Those rows are required by Definition 9462.90. □

Corollary 9462.93 (Top activations are not explanations)**.** *Lists of highly activating examples, feature labels, or visualization snippets are not tensor-certified explanations.*

Proof. Such records can support the pattern row of a Tensor NameCert, but they do not supply the whole source, classifier, stability, and ledger surface. By Definition 9462.92, the feature evidence alone is insufficient. □

Theorem 9462.94 (Source ambiguity blocks tensor certification)**.** *If a tensor classifier candidate does not specify its model state, activation source, source domain, trace protocol, or threshold protocol, then it is not source-complete.*

Proof. Those entries are part of the source row S_C in Definition 9462.89. Omitting them leaves the classifier's domain of application and method of observation ambiguous, so the source row is incomplete. □

Theorem 9462.95 (Pattern evidence needs boundary evidence)**.** *Positive examples alone do not determine a Tensor PatternSpec.*

Proof. Positive examples show where a feature fires or where a classifier cell is populated. They do not identify negative cases, boundary cases, false positives, false negatives, entanglement, or stability under admissible transformations. Those rows are needed for the pattern and ledger surfaces. □

Theorem 9462.96 (Threshold selection is certificate data)**.** *Changing the threshold of a tensor classifier candidate can change its ClassifierSpec.*

Proof. The classifier relation is defined by equality of thresholded marks. A new threshold can move source records from one public mark side to the other, especially near the boundary. Hence threshold choice is not an implementation detail; it is certificate data. □

Theorem 9462.97 (Stability claims require mode and evidence type)**.** *Tensor stability claims are not auditable unless they declare both the stability mode and the evidence type.*

Proof. Feature stability, classifier stability, circuit stability, behavior stability, transfer stability, empirical stability, paper stability, and formal stability have different strengths. Without a declared mode and evidence type, an auditor cannot tell what is being claimed or how far it transports. □

Theorem 9462.98 (Tensor ledger gaps prevent closed explanation)**.** *A tensor-certified explanation is not closed when its ledger omits known or relevant boundary debt.*

Proof. The ledger row records threshold-sensitive cases, false positives, false negatives, source leakage, entanglement, untested regions, proxy behavior, and not-claimed boundaries. If relevant debt is omitted, the explanation hides part of the material compressed by the classifier. The certificate is therefore not closed. □

9462.4.1 Black-Box Resolution

Definition 9462.99 (Tensor black-box behavior)**.** *A behavior is tensor black-box behavior on a stated scope when it has a supporting tensor classifier candidate but no Tensor NameCert for that candidate on that scope.*

Theorem 9462.100 (Tensor NameCert resolves scoped black-box behavior)**.** *If a behavior target B is supported by a tensor classifier candidate C and C carries a Tensor NameCert on the stated scope, then B is no longer tensor black-box behavior on that scope.*

Proof. The black-box condition in Definition 9462.99 is the absence of a Tensor NameCert for the supporting candidate. Supplying that certificate removes the missing row on the stated scope. □

Corollary 9462.101 (Black-box resolution is classifier certification)**.** *Resolving tensor black-box behavior requires certifying the classifier that supports the behavior, not merely naming a feature.*

Proof. By Definition 9462.100, the resolving object is the Tensor NameCert for the supporting classifier. A feature label alone does not supply that certificate. □

9462.4.2 Hardening and Gain

Theorem 9462.102 (Formal hardening does not replace Tensor NameCert)**.** *Formal verification can harden selected Tensor NameCert fields, but it does not replace the Tensor NameCert.*

Proof. A formal proof object verifies a translated statement. It may harden a classifier law, a stability statement, or a boundary claim. It does not automatically supply the model source, activation protocol, pattern evidence, behavior support, or ledger rows. The Tensor NameCert remains the complete explanation surface. □

Theorem 9462.103 (Tensor explanations require net-gain audit when discovery is claimed)**.** *When a tensor-certified explanation is claimed as a discovery, the claim must account for classifier novelty, certificate cost, hardening cost when used, and ledger debt.*

Proof. Classifier novelty is required by Definition 9462.51. The certificate and hardening rows are costs of making the explanation auditable, and ledger debt records unclosed boundary material. Omitting these costs would treat hidden or unverified work as free information. □

Theorem 9462.104 (Tensor NameCert theorem)**.** *In the machine-learning reading, a feature becomes an auditable explanation only through the passage from tensor classifier candidate to Tensor NameCert, with source, pattern, classifier, stability, and ledger rows displayed.*

Proof. The feature functional alone is insufficient by Definition 9462.92. The candidate relation is specified by Definition 9462.87. The Tensor NameCert of Definition 9462.89 supplies the five rows needed for certified explanation, and Definition 9462.102 separates that surface from optional formal hardening. □

*Concretized in §**??**:* Tensor NameCert realizes activation-source, feature-functional, threshold, classifier, stability, ledger, behavior-support, black-box-resolution, and hardening-separation rows as finite certificate packets.

9462.5 Formal Verification as Hard-Certificate Layer

section 9462.4 separates tensor classifier certification from optional formal hardening. This section states the corresponding verification principle. Formal verification is a hard-certificate layer: it accepts proof objects for translated formal statements under a stated backend and trust boundary. It does not create the NameCert surface, does not replace the ledger, and does not by itself change the classifier being certified.

Definition 9462.105 (Verification backend)**.** *A verification backend is a tuple consisting of a formal language, a statement space, a proof-object space, a checker or kernel, an acceptance relation, and a trust boundary. We write*

$$\mathcal{V} \vdash \pi : \varphi$$

when the backend accepts proof object π for statement φ under that trust boundary.

Definition 9462.106 (Hard certificate)**.** *A hard certificate for a statement φ is the backend-relative record*

$$\mathrm{HardCert}_{\mathcal{V}}(\varphi) = (\mathcal{V}, \varphi, \pi, \mathsf{Trust})$$

together with the acceptance judgment

$$\mathcal{V} \vdash \pi : \varphi.$$

It certifies the formal statement relative to the backend, dependencies, assumptions, and trust boundary that appear in the record.

Definition 9462.107 (Verification ledger)**.** *A verification ledger records the backend, formal statement, proof artifact, assumptions, dependencies, trusted components, unchecked components, portability boundary, and the certificate fields not covered by the proof.*

Definition 9462.108 (Formal hardening)**.** *A NameCert row F is formally hardened when it is translated to a formal statement φ_F and a backend accepts a proof object for that statement:*

$$F \rightsquigarrow \varphi_F \qquad \text{and} \qquad \mathcal{V} \vdash \pi_F : \varphi_F.$$

A NameCert is partially hardened when only some rows are hardened; it is fully hardened only when every critical formalizable row is hardened and the remaining rows are covered by the ledger.

Theorem 9462.109 (Hard certificates are backend-relative)**.** *Every hard certificate is relative to a verification backend and trust boundary.*

Proof. By Definition 9462.106, the certificate record contains $\mathcal{V}$ and Trust. The accepted statement is therefore not a free-standing absolute object; it is a checked statement under the backend, assumptions, dependencies, and trusted components recorded by the verification ledger. □

Theorem 9462.110 (Hard certificates do not imply NameCerts)**.** *From $\mathcal{V} \vdash \pi : \varphi$ alone one cannot infer the existence of a NameCert for a classifier.*

Proof. The judgment gives a proof object for one formal statement. A NameCert requires source, pattern, classifier, stability, and ledger rows. The formal judgment may harden one of those rows, but it does not construct the whole five-field surface. □

Corollary 9462.111 (Verified theorem is not certified classifier)**.** *A checked theorem is not, by itself, a certified classifier.*

Proof. Certified classifier objecthood requires a candidate classifier paired with a NameCert. By Definition 9462.110, a checked theorem alone does not supply that pair. □

Theorem 9462.112 (NameCert existence does not require full formal hardening)**.** *A NameCert can exist even when some of its rows are supported by paper, empirical, or ledger evidence rather than formal proofs.*

Proof. The NameCert condition is that the source, pattern, classifier, stability, and ledger rows are displayed and auditable. Formal hardening is one possible evidence mode for a row, not the existence condition for the certificate surface. □

Theorem 9462.113 (Component hardening does not replace other rows)**.** *Hardening one certificate row does not harden or supply the other rows.*

Proof. The accepted proof object has target φ_F, the formal statement corresponding to row F. It says nothing by itself about a different source row, pattern row, classifier row, stability row, or ledger row unless those rows are explicitly included in φ_F. □

Corollary 9462.114 (Verified stability does not replace ledger)**.** *A verified stability claim does not replace the ledger policy.*

Proof. Stability and ledger are distinct rows. By Definition 9462.113, hardening a stability statement does not supply the boundary, failure, residue, trust, or not-claimed rows of the ledger. □

Theorem 9462.115 (Partial hardening is not full hardening)**.** *A partially hardened certificate is not fully hardened unless every critical unverified row is recorded in the ledger as outside the hardened scope.*

Proof. Full hardening requires coverage of all critical formalizable rows, with unhardened rows explicitly ledgered. If a critical row is neither hardened nor ledgered as outside scope, the certificate has a hardening gap and the full-hardening claim is not audit-complete. □

9462.5.1 Verification and Discovery

Definition 9462.116 (Verification-only transition)**.** *A verification-only transition is a passage from (C, N) to (C, N') in which the classifier C is unchanged and the difference is that one or more rows of the NameCert receive additional formal hardening.*

Theorem 9462.117 (Verification-only transitions preserve classifier surface)**.** *A verification-only transition does not create classifier shift.*

Proof. By definition, the classifier relation C is unchanged. The transition changes the evidence status of certificate rows, not the classifier surface on the common source. □

Corollary 9462.118 (Formal verification is not discovery by itself)**.** *Formal verification of a previously unchecked component is not by itself a structural discovery.*

Proof. Structural discovery requires classifier shift by Definition 9462.51. By Definition 9462.117, a verification-only transition preserves the classifier surface. □

Theorem 9462.119 (Hardening costs enter discovery accounting)**.** *When formal hardening is used to support a discovery claim, its cost is part of the certificate cost of that claim.*

Proof. Formal hardening consumes statement translation, proof construction, checking, dependency maintenance, trust-boundary auditing, and artifact transport. If those costs are omitted, the discovery ledger treats proof work as free information. The net-gain audit must therefore include the hardening cost when hardening supports the claim. □

9462.5.2 Verification Transparency

Theorem 9462.120 (Hard certificates require verification ledgers)**.** *A hard certificate without a verification ledger is not audit-complete.*

Proof. Because the hard certificate is backend-relative, an auditor must know which backend accepted the proof, which statement was checked, which dependencies and assumptions were used, which trust boundary was assumed, and which certificate rows were not covered. These are exactly the rows of Definition 9462.107. □

Corollary 9462.121 (Opaque verification is a black box)**.** *A claim that something is verified, without backend, statement, proof artifact, dependencies, and trust boundary, is itself a verification black box.*

Proof. The claim hides the data needed to audit what was checked and under what assumptions. By Definition 9462.120, the hard certificate is not audit-complete. □

Theorem 9462.122 (Verification does not automatically transfer across backends)**.** *A proof accepted by one backend is not automatically accepted by another backend.*

Proof. Backends may differ in language, kernel, proof-object format, axioms, libraries, and trust boundary. Transporting a verified result requires a new proof, a sound translation, or a transport certificate for the backend relation. Acceptance by the first backend alone does not supply those rows. □

Theorem 9462.123 (Machine-learning explanations may be hybrid)**.** *A tensor-certified explanation may combine formally hardened rows with empirical, paper, or ledger-supported rows.*

Proof. Tensor-certified explanation requires the Tensor NameCert surface of Definition 9462.89. Some of its rows may be translated into formal statements and checked; others may remain empirical or paper-level, provided their evidence type and boundary are recorded. Such an explanation is stronger than a ledger-only certificate in the hardened rows, but weaker than a fully hardened certificate. □

Theorem 9462.124 (Formal-verification layer theorem)**.** *Formal verification hardens certificate rows by producing backend-relative hard certificates with verification ledgers; it does not replace the NameCert layer, the ledger layer, or the classifier-shift condition for discovery.*

Proof. Hard certificates are backend-relative by Definition 9462.109. They do not imply NameCerts by Definition 9462.110. Their ledgers remain necessary by Definition 9462.120. Finally, verification-only transitions preserve classifier surface by Definition 9462.117. The formal-verification layer therefore hardens rows without replacing the certification, ledger, or discovery layers. □

Concretized in §8662: Hard-certificate layers realize backend-relative proof acceptance, verification ledgers, partial hardening, trust-boundary exposure, non-transfer across backends, and no-discovery-by-verification as finite NameCert rows.

9462.6 Hardenable and Ledger-Only Classifiers

section 9462.5 identifies formal verification as a hardening layer. This section records the corresponding classifier taxonomy. A certified classifier may be ledger-only, partially hardened, or fully hardened. Lack of formal hardening is not the same as lack of certification; lack of ledgered scope and boundary is the actual obstruction to closure.

Definition 9462.125 (Evidence mode)**.** *Each row of a NameCert has an evidence mode: empirical, paper, formal, hybrid, ledger-only, or open. The mode states how that row is supported and what kind of audit is available for it.*

Definition 9462.126 (Mode-complete NameCert)**.** *A NameCert is mode-complete when every source, pattern, classifier, stability, and ledger row declares an evidence mode and no row remains open.*

Definition 9462.127 (Mode gap)**.** *A mode gap is a NameCert row whose evidence mode is missing or open. Mode gaps prevent an auditor from determining whether the row is empirical, paper-level, formal, hybrid, ledger-only, or not yet closed.*

Theorem 9462.128 (Mode gaps prevent full auditability)**.** *A NameCert with a mode gap is not fully auditable.*

Proof. Full auditability requires knowing the evidence source and strength for each certificate row. If a row has no declared mode, the auditor cannot determine how the row is supported or whether it is closed. Thus the certificate remains audit-incomplete. □

Definition 9462.129 (Hardenable row)**.** *A NameCert row is hardenable relative to a backend $\mathcal{V}$ when it can be transcribed into a formal statement for that backend and the corresponding proof obligation is explicit. Hardenability is a possibility of formalization, not the existence of a proof object.*

Definition 9462.130 (Hardened row)**.** *A hardenable row is hardened relative to $\mathcal{V}$ when the backend accepts a proof object for its transcribed statement.*

Definition 9462.131 (Hardening frontier)**.** *The hardening frontier of a NameCert is the family of hardenable rows that have not yet been hardened. The non-hardenable residue is the family of rows that are not currently transcribable as formal proof obligations for the stated backend.*

Theorem 9462.132 (Hardened implies hardenable)**.** *Every hardened row is hardenable relative to the backend that hardens it.*

Proof. Hardening requires both a transcription to a formal statement and an accepted proof object for that statement. The transcription and proof obligation are exactly the data required for hardenability. □

Theorem 9462.133 (Hardenability is backend-relative)**.** *A row hardenable in one verification backend is not automatically hardenable in another.*

Proof. Backends may have different languages, libraries, kernels, axioms, proof styles, and available encodings. A row transcribable in one backend need not have a faithful or useful transcription in another. The backend must therefore be named in every hardenability claim. □

9462.6.1 Classifier Strength Classes

Definition 9462.134 (Ledger-only classifier)**.** *A ledger-only classifier is a Name-certified classifier whose NameCert is mode-complete, has no hardened rows, and records all non-hardenable residue in the ledger.*

Definition 9462.135 (Partially hardened classifier)**.** *A partially hardened classifier is a Name-certified classifier with at least one hardened row and at least one critical row that is either unhardened or ledger-only.*

Definition 9462.136 (Fully hardened classifier)**.** *A fully hardened classifier is a Name-certified classifier whose hardenable critical rows are hardened, whose non-hardenable residue is ledgered, whose verification ledger is complete, and whose NameCert is mode-complete.*

Definition 9462.137 (Critical hardening gap)**.** *A critical hardening gap is a critical hardenable row that is neither hardened nor recorded in the hardening frontier or ledger. A critical row is one whose absence or misclassification can change source scope, classifier relation, stability, ledger boundary, discovery status, or positive-gain accounting.*

Theorem 9462.138 (Fully hardened classifiers have no critical hardening gap)**.** *A fully hardened classifier has no critical hardening gap.*

Proof. By definition, every hardenable critical row is hardened, and every non-hardenable residue is ledgered. A critical row that were neither hardened nor recorded would violate one of those clauses. □

Theorem 9462.139 (Ledger-only classifiers can be certified objects)**.** *A ledger-only classifier can be a BEDC-certified object.*

Proof. Certified objecthood requires a classifier paired with a NameCert. A ledger-only classifier has a mode-complete NameCert and records its non-hardenable residue in the ledger. It lacks formal hardening, but formal hardening is not required for NameCert existence by Definition 9462.112. □

Corollary 9462.140 (No formal proof is not no certificate)**.** *A classifier may lack formal verification and still be certified by NameCert and ledger.*

Proof. This is the ledger-only case of Definition 9462.139. □

Theorem 9462.141 (Partial hardening strengthens only hardened rows)**.** *Moving from ledger-only to partially hardened status strengthens only the rows that are actually hardened.*

Proof. By Definition 9462.113, hardening one row does not supply or harden the other rows. The partially hardened classifier is stronger exactly where the proof object has been accepted. □

Theorem 9462.142 (Empirical stability is not ledger-only legitimacy)**.** *Empirical stability of a classifier candidate does not by itself make it a ledger-only classifier.*

Proof. Empirical stability may support a pattern or stability row. A ledger-only classifier also requires a full NameCert, declared evidence modes, ledgered non-hardenable residue, and no open mode gap. Empirical stability supplies only part of that surface. □

9462.6.2 Priority

Definition 9462.143 (Net hardening gain)**.** *The net hardening gain of a row is the audit, reliability, reuse, or transport value obtained by hardening it minus the modeling, proof, dependency, maintenance, and verification cost of doing so.*

Theorem 9462.144 (Not every hardenable row should be hardened immediately)**.** *Some hardenable rows have nonpositive net hardening gain.*

Proof. A row may be transcribable while its proof is costly, fragile, highly backend-specific, or low in reuse value. In such a case the hardening cost can exceed the audit gain. Hardenability is therefore a necessary condition for hardening, not a priority rule. □

Theorem 9462.145 (Ledger completion can outrank formal hardening)**.** *Completing the ledger can have higher priority than hardening a row.*

Proof. If the ledger has a critical boundary gap, the explanation may overclaim its scope. Hardening a local row that does not address that gap leaves the central audit risk unresolved. In that situation ledger completion has higher certificate value. □

9462.6.3 Machine-Learning Reading

Definition 9462.146 (Neural hardening profile)**.** *A neural hardening profile records, for a Tensor NameCert, the evidence mode of each row, the hardening frontier, the non-hardenable residue, the ledger status, the hardened rows, and the not-claimed boundary.*

Theorem 9462.147 (Neural explanations can be legitimate without full formalization)**.** *A neural explanation with a complete Tensor NameCert and ledgered non-hardenable residue can be legitimate even when it is not fully hardened.*

Proof. Legitimacy at the certification layer requires source, pattern, classifier, stability, and ledger rows with declared evidence modes. Fully formal hardening is a stronger status, but it is not the gate of NameCert existence. □

Theorem 9462.148 (Concept labels are ledger-only before semantic hardening)**.** *Natural-language concept assignments to neural features should be read as ledger-only components unless a formal semantics and proof are supplied.*

Proof. Natural-language concepts often have open boundaries, context dependence, and distributional sensitivity. They may be useful semantic assignments, but without formal semantics and proof they are not hardened rows. They can still be certified as ledgered semantic assignments with examples, counterexamples, scope, and not-claimed boundaries. □

Theorem 9462.149 (Ledger-only discovery candidates can be genuine candidates)**.** *A classifier transition with classifier shift, NameCert, and ledger may be a genuine discovery candidate even before formal hardening.*

Proof. Classifier-shift discovery requires a changed certified classifier surface and ledgered certificate support. Formal hardening increases strength, but it is not listed as an existence condition for classifier shift in Definition 9462.51. □

Theorem 9462.150 (Hardening taxonomy theorem)**.** *The certification layer distinguishes hardenable rows, hardened rows, and ledger-only rows; certified classifier status depends on NameCert and ledger completeness, while formal hardening determines a stronger audit grade within that certified status.*

Proof. The distinction follows from the definitions of hardenable row, hardened row, ledger-only classifier, partially hardened classifier, and fully hardened classifier. Definition 9462.139 shows that ledger-only certification can be object-forming, while Definition 9462.141 and Definition 9462.138 state the stronger hardening grades. □

*Concretized in §**??**:* Hardenable and ledger-only classifier packets realize evidence modes, hardening frontier, non-hardenable residue, ledger-only legitimacy, partial hardening, hardening priority, and neural hardening profiles as finite NameCert rows.

Concretized in §8206: Ledger-mode classifiers realize evidence-mode, mode-gap, hardening-frontier, and ledger-only residue rows as finite NameCert data.

9462.7 Black Box as Missing Certificate

section 9462.4 defines tensor black-box behavior in the machine-learning case. The general principle is broader: a black box is not absence of structure. It is supported behavior whose classifier surface has not yet received the NameCert and ledger rows needed for audit.

Definition 9462.151 (Behavior target)**.** *A behavior target is a displayed output, decision, refusal, error, transfer, safety, or generalization pattern on a stated behavior domain. It may be recorded as a relation, predicate, soft score, or finite evidence packet.*

Definition 9462.152 (Explanation candidate)**.** *An explanation candidate for a behavior target B is a pair (C, B) where C is a classifier candidate and C supports B by prediction, correlation, intervention, causal account, formal statement, or hybrid evidence.*

Definition 9462.153 (Certified explanation)**.** *A certified explanation for B is a tuple*

$$(C, \mathsf{NameCert}(C), B)$$

where C supports B and the NameCert has source, pattern, classifier, stability, and ledger rows for the stated scope.

Definition 9462.154 (Black-box behavior)**.** *A behavior target B is black-box behavior when it has a supporting classifier candidate but no NameCert for that candidate on the stated scope:*

$$\mathsf{BlackBox}(B) \iff \exists C.\, C \Vdash B \text{ and } \neg\exists N.\, N = \mathsf{NameCert}(C).$$

Definition 9462.155 (Unexplained behavior)**.** *A behavior target is unexplained when no supporting classifier candidate has been displayed for it.*

Definition 9462.156 (Closed explanation behavior)**.** *A behavior target is closed on a stated scope when it has a certified explanation whose ledger is complete on that scope.*

Theorem 9462.157 (Black boxes have support structure)**.** *Every black-box behavior has a supporting classifier candidate.*

Proof. The existence of a supporting classifier candidate is the first clause of Definition 9462.154. What is missing is not structure, but the certificate that licenses the structure. □

Theorem 9462.158 (Unexplained behavior is not black-box behavior)**.** *Unexplained behavior is distinct from black-box behavior.*

Proof. Unexplained behavior has no displayed supporting classifier candidate. Black-box behavior, by contrast, has such a candidate but lacks its NameCert. The two failure modes occur at different levels. □

Theorem 9462.159 (Closed explanation is not black box)**.** *A behavior target with a ledger-complete certified explanation is not black-box behavior on that scope.*

Proof. Black-box behavior requires absence of a NameCert for the supporting classifier. A ledger-complete certified explanation supplies that NameCert and its ledger on the stated scope, so the missing-certificate condition fails. □

9462.7.1 Certificate Deficits

Definition 9462.160 (Black-box type)**.** *A black-box type records which NameCert row is missing or incomplete: source, pattern, classifier, stability, ledger, or verification ledger. Multiple missing rows form a composite black box.*

Definition 9462.161 (Certificate deficit)**.** *For an explanation candidate (C, B), the certificate deficit is the set of missing or incomplete rows among source, pattern, classifier, stability, and ledger. A verification deficit is recorded separately when the explanation invokes a hard certificate without its verification ledger.*

Definition 9462.162 (Black-box debt)**.** *Black-box debt is the weighted cost of the certificate deficit for the best displayed supporting classifier candidate for B. Ledger deficits may receive greater weight because they can produce plausible but unbounded half-explanations.*

Theorem 9462.163 (Missing NameCert rows diagnose black boxes)**.** *If C supports B but one of the required NameCert rows is missing or incomplete, then the explanation is not closed.*

Proof. Closed explanation requires a complete NameCert and ledger on the stated scope. Omitting source, pattern, classifier, stability, or ledger leaves the certificate surface incomplete. The failure can therefore be diagnosed by the missing row. □

Theorem 9462.164 (Ledger black boxes are high-risk half-explanations)**.** *An explanation with source, pattern, classifier, and stability rows but without a ledger is not closed, even when it appears structurally rich.*

Proof. The first four rows can make the explanation appear complete: they name where it applies, what pattern is seen, what classifier is used, and why it is stable. Without the ledger, failure cases, hidden residue, false-positive and false-negative behavior, distribution boundary, and not-claimed scope are unaccounted for. The explanation is therefore a plausible half-explanation, not a closed one. □

Theorem 9462.165 (Closed explanations have zero certificate deficit)**.** *A ledger-complete certified explanation has zero certificate deficit on its stated scope.*

Proof. By definition, the certified explanation supplies every NameCert row and the ledger is complete on the scope. No required row is missing, so the deficit set is empty. □

Theorem 9462.166 (Black-box debt can decrease before closure)**.** *An explanation process can reduce black-box debt without yet producing a closed explanation.*

Proof. Supplying a source row, classifier row, negative examples, or stability witness removes part of the certificate deficit. If the ledger remains incomplete, the explanation is still not closed, but the debt has decreased. □

9462.7.2 Resolution

Definition 9462.167 (Black-box reduction)**.** *A black-box reduction is a transition of explanation state that lowers the black-box debt of a behavior target.*

Definition 9462.168 (Scoped black-box resolution)**.** *Scoped black-box resolution occurs when a behavior target receives a ledger-complete certified explanation on a specified model state, source domain, transformation family, and behavior scope.*

Definition 9462.169 (Global black-box resolution claim)**.** *A global black-box resolution claim asserts closed explanation for all relevant behaviors, inputs, distributions, model states, failures, and not-claimed boundaries.*

Theorem 9462.170 (Scoped resolution is obtained by NameCert)**.** *If a behavior has a supporting classifier with a ledger-complete NameCert on a stated scope, then scoped black-box resolution holds for that scope.*

Proof. The NameCert supplies source, pattern, classifier, stability, and ledger rows for the supporting classifier. Ledger completeness supplies the scope boundary. This is exactly Definition 9462.168. □

Theorem 9462.171 (Global resolution requires global ledger)**.** *A global black-box resolution claim requires a global ledger for the claimed behavior space.*

Proof. The claim ranges over all relevant behaviors and boundaries. Each one must either have a closed explanation or be explicitly outside the claim. That accounting is precisely the work of a global ledger. □

Corollary 9462.172 (Interpretability claims are scoped by default)**.** *Machine-learning explanations should be read as scoped unless a global ledger is displayed.*

Proof. By Definition 9462.171, global resolution requires stronger ledger coverage than ordinary feature, probe, visualization, or local verification claims provide. □

9462.7.3 Practice

Theorem 9462.173 (Feature labels do not resolve black boxes)**.** *Naming a feature does not by itself resolve black-box behavior.*

Proof. A feature label is a semantic assignment. It does not by itself supply the source, classifier, stability, and ledger rows required by Definition 9462.153. □

Theorem 9462.174 (Visual explanations require projection ledger)**.** *Visualizations used as explanations require a projection or compression ledger.*

Proof. A visualization maps high-dimensional or structured records into a display surface. This projection can collapse distinctions and hide boundary behavior. Without a ledger for that collapse, the visualization is evidence material rather than closed explanation. □

Theorem 9462.175 (Probe explanations require probe ledger)**.** *A probe-based explanation requires a ledger for the probe classifier and its hidden residue.*

Proof. A probe is itself a classifier extraction or compression map. Its accuracy can support a pattern or stability row, but false positives, false negatives, probe-specific artifacts, source scope, and hidden residue must still be ledgered. □

Theorem 9462.176 (Black-box reduction is not necessarily discovery)**.** *Lowering black-box debt does not necessarily create classifier shift.*

Proof. Debt reduction may come from filling source rows, adding boundary cases, recording a ledger, or hardening an existing component. If the classifier relation is unchanged, the classifier surface has not shifted. □

Theorem 9462.177 (Black-box resolution may contain discovery)**.** *A black-box resolution process can contain a discovery step when it introduces a new certified classifier.*

Proof. If the resolution process supplies a classifier not classifier-equivalent to the prior certified state and gives it NameCert and ledger support, then the process includes classifier shift. By Definition 9462.51, that step is a discovery candidate. □

Theorem 9462.178 (Black-box theorem)**.** *In the certificate reading, a black box is uncertified supporting structure: it is neither structureless behavior nor closed explanation.*

Proof. Definition 9462.157 separates black boxes from unexplained behavior, while Definition 9462.159 separates them from closed explanations. The remaining condition is missing NameCert or ledger support for a displayed classifier candidate. □

*Concretized in §**??**:* Black-box missing-certificate packets realize behavior support, certificate deficits, black-box debt, scoped resolution, ledger-black-box refusal, projection/probe ledger obligations, and discovery separation as finite NameCert rows.

Concretized in §7335: Black-box certificate deficits are realized as finite behavior-target, support-classifier, missing-row, ledger, refusal, transport, replay, provenance, and NameCert rows.

9462.8 Black-Box Ledger Theory

section 9462.7 defines black-box debt as certificate deficit. This section isolates the ledger part of that deficit. Any explanation, classifier, compression, projection, threshold, generalization, semantic assignment, intervention, or verification hides residue. A ledger is the audit surface that records that residue; unrecorded residue remains black-box debt.

Definition 9462.179 (Ledger entry)**.** *A ledger entry records a residue kind, a source reference, the residue being recorded, and an optional risk or weight annotation. Residue kinds include source, pattern, classifier, stability, threshold, projection, compression, generalization, intervention, verification, semantic, and not-claimed rows.*

Definition 9462.180 (Ledger)**.** *A ledger is a finite or auditable family of ledger entries attached to a target object: a classifier, NameCert, behavior, explanation, compression map, soft formula, intervention claim, semantic assignment, or formal statement.*

Definition 9462.181 (Recorded and required residue)**.** *For a ledger* Λ*,* $\mathrm{Rec}(\Lambda)$ *is the family of residue rows recorded by* Λ*. For a target object* O*,* $\mathrm{Req}(O)$ *is the family of residue rows that must be recorded for the claim about* O *to be closed.*

Definition 9462.182 (Ledger completeness and gap)**.** *A ledger is complete for* O *when*

$$\mathrm{Req}(O) \subseteq \mathrm{Rec}(\Lambda).$$

Its ledger gap is

$$\mathrm{Gap}(\Lambda, O) = \mathrm{Req}(O) \setminus \mathrm{Rec}(\Lambda).$$

Theorem 9462.183 (Ledger completeness is absence of gap)**.** *A ledger is complete for* O *exactly when its ledger gap for* O *is empty.*

Proof. The gap is defined as the required residue not recorded by the ledger. That set is empty exactly when every required residue row is recorded, which is the completeness condition. □

Theorem 9462.184 (Ledger extension cannot create new fixed-object gaps)**.** *If $\Lambda \subseteq \Lambda'$ and the target object O is fixed, then*

$$\mathrm{Gap}(\Lambda', O) \subseteq \mathrm{Gap}(\Lambda, O).$$

Proof. Every row recorded by Λ is also recorded by Λ'. Thus the required rows missing from Λ' form a subfamily of the required rows missing from Λ. □

Definition 9462.185 (Ledger debt)**.** *Ledger debt is the weighted cost of the gap:*

$$\mathrm{Debt}(\Lambda, O) = \sum_{r \in \mathrm{Gap}(\Lambda, O)} \mathrm{Cost}(r).$$

Critical debt is the same sum restricted to residues whose absence can change the claim's truth, scope, risk, classifier boundary, discovery status, or positive-gain accounting.

Theorem 9462.186 (Complete ledger has zero debt)**.** *A complete ledger has zero ledger debt.*

Proof. By Definition 9462.183, the gap is empty. The debt sum over an empty family is zero. □

Theorem 9462.187 (Ledger debt is monotone under fixed-object extension)**.** *For nonnegative residue costs, extending the ledger of a fixed object can only decrease or preserve ledger debt.*

Proof. By Definition 9462.184, the gap of the extended ledger is a subfamily of the original gap. Summing nonnegative costs over a subfamily cannot increase the total. □

Theorem 9462.188 (Critical debt blocks strong claims)**.** *If a target object has unrecorded critical residue, then strong claims depending on that object are not closed.*

Proof. Critical residue is defined by its ability to affect truth, scope, risk, classifier boundary, discovery status, or gain accounting. If it remains unrecorded, the strong claim lacks a necessary audit row. □

9462.8.1 Ledger Kinds

Definition 9462.189 (Ledger kind)**.** *Ledger kinds are indexed by the residue they record: source ledgers, pattern ledgers, classifier ledgers, stability ledgers, generalization ledgers, threshold ledgers, projection ledgers, intervention ledgers, verification ledgers, and semantic ledgers.*

Theorem 9462.190 (Ledger kinds do not generally substitute for each other)**.** *One ledger kind does not generally replace another.*

Proof. Different ledgers record different residue. A source ledger records where records come from; a threshold ledger records boundary sensitivity; a projection ledger records collapsed distinctions; a verification ledger records backend and trust assumptions; a semantic ledger records concept boundary and ambiguity. Recording one of these does not record the others. □

Corollary 9462.191 (Complete explanations often need multiple ledgers)**.** *Machine-learning explanations often require several ledger kinds at once.*

Proof. An explanation may simultaneously use source selection, thresholding, projection, semantic labeling, intervention, generalization, and formal verification. By Definition 9462.190, each involved residue kind requires its own ledger row. □

9462.8.2 Obligations

Theorem 9462.192 (Compression without ledger hides debt)**.** *A compression or projection that collapses distinctions without recording the collapsed residue carries hidden debt.*

Proof. Collapsed distinctions are required residue for a compression claim. If they are not recorded, they lie in the ledger gap and therefore contribute to ledger debt. □

Theorem 9462.193 (Generalization requires target-residue ledger)**.** *A generalization claim is not closed without a ledger for target residue and target failure boundary.*

Proof. Generalization extends a classifier from a source setting to a target setting. Target cases not covered by the source, transfer witness, or stability witness are required residue. Omitting them leaves the scope of the generalization claim unclosed. □

Theorem 9462.194 (Semantic labels require semantic ledger)**.** *A natural-language concept label for a classifier cell is not closed meaning without a semantic ledger.*

Proof. Natural-language labels have ambiguous cases, negative examples, polysemy, domain dependence, and unsupported connotations. These are required residue for the semantic assignment. □

Theorem 9462.195 (Causal explanations require intervention ledger)**.** *A causal or mechanistic claim based on intervention is not closed without an intervention ledger.*

Proof. Interventions can have off-target effects, alternative causal paths, strength sensitivity, distribution shifts, downstream compensation, and failed variants. These residue rows determine the scope of the causal claim. □

Theorem 9462.196 (Verification ledger does not replace behavior ledger)**.** *A verification ledger does not replace behavior, classifier, generalization, semantic, or intervention ledgers.*

Proof. The verification ledger records proof backend, statement, dependencies, trust boundary, and unchecked components. It does not record behavior failures, distribution boundary, semantic ambiguity, probe artifacts, or intervention residue unless those rows are explicitly included. □

9462.8.3 Repair and Discovery

Definition 9462.197 (Ledger repair)**.** *Ledger repair is a transition $\Lambda \to \Lambda'$ that lowers the debt of a fixed target object.*

Definition 9462.198 (Ledger discovery)**.** *Ledger discovery occurs when ledger repair uncovers critical residue that forces a change in classifier, source scope, or discovery status.*

Theorem 9462.199 (Ledger repair is not necessarily discovery)**.** *Ledger repair that preserves the classifier surface is not classifier discovery.*

Proof. Classifier discovery requires classifier shift. If ledger repair only records missing boundary or failure rows while the classifier relation stays fixed, the audit status improves but the classifier surface does not shift. □

Theorem 9462.200 (Critical ledger discovery can induce classifier discovery)**.** *If newly recorded critical residue forces the classifier relation to change, then the ledger discovery induces classifier shift.*

Proof. Changing the classifier relation means that some source records are no longer classified as before. That is precisely classifier shift on the public classifier surface. □

Theorem 9462.201 (Positive claims must count ledger cost and debt)**.** *A positive discovery, compression, generalization, or explanation-gain claim must count both ledger cost and remaining ledger debt.*

Proof. Ledger cost is part of the cost of making the claim auditable. Remaining ledger debt is the cost of residue not yet recorded. Omitting either term overstates the certified gain. □

Theorem 9462.202 (Black-box ledger theorem)**.** *Ledger is the closure surface for black-box debt: recorded residue turns hidden boundary into auditable boundary, while unrecorded residue remains black-box residue.*

Proof. By Definition 9462.183, ledger closure is exactly absence of required residue gaps. By Definition 9462.185, unrecorded residue contributes to debt. The ledger therefore converts hidden residue into auditable rows, and whatever is not ledgered remains black-box residue. □

*Concretized in §**??**:* Black-box ledger theory realizes ledger entries, required and recorded residue, debt monotonicity, ledger-kind non-substitution, compression/generalization/semantic/intervention/verification ledger obligations, ledger repair, and critical ledger discovery as finite NameCert rows.

Concretized in §6100: The decomposition certificate realizes black-box ledger deficits as finite certificate rows with visible missing-field and debt surfaces.

9462.9 Neural Theorem-DNA

section 9462.4 treats tensor features, probes, directions, and circuits as certificate candidates rather than standalone explanations. section 9462.8 then identifies the ledger surface that prevents hidden residue from remaining black-box debt. The neural theorem-DNA reading combines these two constraints: a model behavior is not explained by one feature label or one circuit diagram, but by a role-complete profile whose roles are source, feature, circuit, classifier, stability, intervention, ledger, and seal. The term genome names this role-complete audit profile only; it does not import biological structure.

Definition 9462.203 (Neural behavior target)**.** *A neural behavior target is a displayed behavior B of a tensor model context. It may be represented as a relation on inputs and outputs, a predicate on inputs or traces, or a soft behavior score, provided the source scope and observation protocol are declared.*

Definition 9462.204 (Neural theorem-DNA loci)**.** *The neural theorem-DNA locus family is*

$$\mathcal{R}_{\mathsf{NDNA}} = \{\mathsf{source}, \mathsf{feature}, \mathsf{circuit}, \mathsf{classifier}, \mathsf{stability}, \mathsf{intervention}, \mathsf{ledger}, \mathsf{seal}\}.$$

For a behavior B, a neural theorem-DNA packet is an eight-tuple

$$\mathsf{NDNA}(B) = (G_{\mathsf{source}}, G_{\mathsf{feature}}, G_{\mathsf{circuit}}, G_{\mathsf{classifier}}, G_{\mathsf{stability}}, G_{\mathsf{intervention}}, G_{\mathsf{ledger}}, G_{\mathsf{seal}}).$$

The component G_ρ is the gene occupying locus $\rho \in \mathcal{R}_{\mathsf{NDNA}}$.

Definition 9462.205 (Neural theorem genome)**.** *A neural theorem genome for B is the behavior together with a neural theorem-DNA packet:*

$$\Gamma_B = (B, \mathsf{NDNA}(B)).$$

The packet is complete when every locus has declared content, evidence mode, scope, and ledger status.

Definition 9462.206 (Locus content)**.** *The source locus records model identity, parameter state, data domain, input family, task condition, prompt or observation protocol, trace extraction method, target definition, and inclusion boundary. The feature locus records activation features, directions, sparse-autoencoder latents, probes, subspaces, attention patterns, residual-stream features, or tensor distinction functionals. The circuit locus records layer, attention, MLP, residual, upstream, downstream, and causal-route structure. The classifier locus records the classifier relation induced by the behavior explanation. The stability locus records perturbation, threshold, representation, classifier, circuit, behavior, or transfer stability. The intervention locus records ablation, activation transfer, tracing, steering, suppression, amplification, counterfactual, failed, or off-target intervention evidence. The ledger locus records residue and boundary debt. The seal locus records the closure claim, evidence mode, remaining gaps, not-claimed boundary, and hardening status.*

Definition 9462.207 (NDNA gap and debt)**.** *The neural theorem-DNA gap of B is the family of incomplete loci:*

$$\mathrm{Gap}_{\mathsf{NDNA}}(B) = \{\rho \in \mathcal{R}_{\mathsf{NDNA}} \mid G_\rho(B) \text{ is not complete}\}.$$

For a nonnegative cost assignment on loci, the NDNA debt is

$$\mathrm{Debt}_{\mathsf{NDNA}}(B) = \sum_{\rho \in \mathrm{Gap}_{\mathsf{NDNA}}(B)} \mathrm{Cost}(\rho).$$

A critical locus is a locus whose absence can change the source scope, classifier boundary, causal status, ledger closure, discovery status, or positive-gain accounting of the explanation.

Theorem 9462.208 (Complete NDNA has zero NDNA debt)**.** *If $\mathsf{NDNA}(B)$ is complete, then*

$$\mathrm{Debt}_{\mathsf{NDNA}}(B) = 0.$$

Proof. Completeness means that no locus is incomplete. Hence $\mathrm{Gap}_{\mathsf{NDNA}}(B)$ is empty, and the debt sum over the empty family is zero. □

Theorem 9462.209 (Single-locus records are not closed explanations)**.** *No isolated feature, circuit, classifier, intervention, or semantic label is a closed explanation of B.*

Proof. A closed explanation must declare the source scope, classifier surface, stability mode, ledger boundary, and seal. A feature record lacks the other roles; a circuit diagram lacks the classifier and ledger surfaces; a classifier relation lacks source, stability, and seal data; an intervention result lacks off-target and failed-variant ledgers; and a semantic label lacks the classifier and semantic-ledger boundary. Each is therefore only a locus or locus fragment. □

Definition 9462.210 (Anchored locus)**.** *A locus $G_\rho(B)$ is anchored when it has an anchor witness $A_\rho(B)$ tracing the locus to behavior evidence, tensor traces, dataset evidence, intervention records, a NameCert component, a ledger entry, a formal proof certificate, or a paper proof. A neural theorem-DNA packet is anchored when all its loci are anchored.*

Theorem 9462.211 (Unanchored critical loci block certification)**.** *If a critical locus of* $\mathsf{NDNA}(B)$ *is unanchored, then* $\mathsf{NDNA}(B)$ *cannot certify a closed explanation of* B*.*

Proof. A critical locus can change scope, classifier boundary, causal status, ledger closure, discovery status, or gain accounting. Without an anchor witness, the auditor cannot tell whether that locus is evidence-bearing or merely descriptive. Certification therefore fails at the audit surface. □

Corollary 9462.212 (Natural language cannot fill an anchor gap)**.** *An unanchored natural-language description cannot substitute for a critical anchor witness.*

Proof. Such a description does not identify a trace, intervention, certificate, ledger row, or proof from which the locus is read. By Definition 9462.211, the critical gap remains. □

Definition 9462.213 (NDNA-to-NameCert projection)**.** *For a complete neural theorem-DNA packet, its NameCert projection is*

$$\Pi_{\mathsf{NameCert}}(\mathsf{NDNA}(B)) = (S_B, P_B, C_B, \Sigma_B, L_B).$$

The source row S_B *is read from the source locus; the pattern row* P_B *is read from the feature and circuit loci; the classifier row* C_B *is read from the classifier locus; the stability row* Σ_B *is read from the stability and intervention loci; and the ledger row* L_B *is read from the ledger locus. The seal locus states the closure scope of this projection.*

Definition 9462.214 (NameCert-compatible NDNA)**.** *A complete neural theorem-DNA packet is NameCert-compatible when* $\Pi_{\mathsf{NameCert}}(\mathsf{NDNA}(B))$ *forms a Tensor NameCert for the classifier supporting* B*.*

Theorem 9462.215 (NameCert-compatible NDNA yields tensor certification)**.** *If* $\mathsf{NDNA}(B)$ *is complete, anchored, and NameCert-compatible, then* B *has a tensor-certified explanation within the seal scope.*

Proof. NameCert compatibility supplies the five Tensor NameCert rows of Definition 9462.89. Completeness supplies all eight neural loci, anchoring supplies the evidence route for each locus, and the seal declares the scope. Thus the classifier supporting B is displayed with the required source, pattern, classifier, stability, and ledger rows. □

Theorem 9462.216 (Complete NDNA need not be NameCert-compatible)**.** *A complete neural theorem-DNA packet need not be NameCert-compatible.*

Proof. Completeness records that every locus has declared content. It does not force cross-locus agreement. The source locus may name one domain, the classifier locus another; the stability locus may concern a perturbation family different from the one used by the classifier; or the ledger may record residue for a threshold not used in the classifier row. In such a case the eight loci are present, but the projected five-field certificate is inconsistent. □

Definition 9462.217 (Causal and correlation-only NDNA)**.** *A causal NDNA claim says that the displayed feature, circuit, or classifier causally supports* B*. A correlation-only NDNA supplies feature, classifier, or stability evidence without an intervention locus.*

Theorem 9462.218 (Causal NDNA claims require intervention ledger)**.** *A causal NDNA claim is not closed unless the intervention locus is present and its intervention residue is ledgered.*

Proof. The causal claim must distinguish causal support from correlation. Interventions can have off-target effects, alternative causal paths, strength sensitivity, failed variants, and downstream compensation. By Definition 9462.195, those rows must be ledgered for the causal boundary to close. □

Definition 9462.219 (Semantic gene)**.** *A semantic gene is a natural-language concept label assigned to a feature, classifier cell, behavior, or circuit role. It is licensed when it has a semantic ledger and agrees with the classifier locus.*

Theorem 9462.220 (Semantic genes require classifier and ledger)**.** *A semantic gene cannot enter a certified NDNA packet unless it is bound to the classifier locus and accompanied by a semantic ledger.*

Proof. Natural-language labels have ambiguous cases, negative cases, polysemy, domain dependence, unsupported connotations, and drift. A classifier relation determines the public equivalence surface; the semantic ledger records the boundary and not-claimed range of the label. Without both, the semantic gene is not auditable. □

Definition 9462.221 (NDNA score and certified gain)**.** *The NDNA complexity vector is*

$$\mathbf{C}_{\mathsf{NDNA}}(B) = (c_{\mathsf{source}}, c_{\mathsf{feature}}, c_{\mathsf{circuit}}, c_{\mathsf{classifier}}, c_{\mathsf{stability}}, c_{\mathsf{intervention}}, c_{\mathsf{ledger}}, c_{\mathsf{seal}}).$$

For a nonnegative weight profile $\mathbf{w}_{\mathsf{NDNA}}$*, define*

$$\mathrm{Score}_{\mathsf{NDNA}}(B) = \mathbf{w}_{\mathsf{NDNA}} \cdot \mathbf{C}_{\mathsf{NDNA}}(B).$$

The certified gain of an NDNA explanation counts coverage, prediction, transfer, and reuse value, then subtracts score and remaining debt.

Theorem 9462.222 (Lower NDNA score does not imply better explanation)**.** *An explanation with lower NDNA score need not be a better explanation.*

Proof. Score records the cost of the displayed loci. Explanation quality also depends on coverage, prediction, transfer, classifier shift, causal strength, ledger completeness, and remaining debt. A short low-score profile may cover little and hide residue, while a higher-score profile may close more of the behavior boundary. □

Definition 9462.223 (NDNA mutations)**.** *An NDNA mutation is a transition*

$$\mathsf{NDNA}(B) \rightsquigarrow \mathsf{NDNA}'(B).$$

It is a compression mutation when it lowers score while preserving the classifier locus. It is a classifier-shift mutation when the classifier locus changes. It is a discovery mutation when the classifier shift is NameCert-compatible and ledgered.

Theorem 9462.224 (Compression and discovery are independent)**.** *NDNA compression and NDNA discovery are independent coordinates.*

Proof. A compression mutation can reduce score by shortening feature, circuit, proof, or seal records while preserving the classifier locus; then no classifier-shift discovery has occurred. Conversely, a discovery mutation may introduce classifier, stability, ledger, and seal rows whose cost raises the score. Thus score reduction does not imply discovery, and discovery does not require immediate score reduction. □

Theorem 9462.225 (Ledger mutation can induce NDNA discovery)**.** *If a ledger mutation records critical residue that forces the classifier locus to change, then it induces an NDNA discovery mutation whenever the resulting packet is NameCert-compatible and ledgered.*

Proof. Critical residue can change the classifier boundary. Once the classifier locus changes, the transition is a classifier-shift mutation. If the resulting packet also has a compatible NameCert projection and ledger, it satisfies the definition of NDNA discovery mutation. □

Definition 9462.226 (NDNA hardening)**.** *An NDNA locus is formally hardened when the claim carried by that locus has been rendered as a formal proposition and accepted by the relevant verification system. A packet is partially hardened when some loci are hardened and the remaining residue is ledgered. It is fully hardened when all hardenable critical loci are hardened and all non-hardenable critical residue is ledgered.*

Theorem 9462.227 (One hardened locus does not harden the whole packet)**.** *Formal hardening of one NDNA locus does not imply that the whole neural theorem-DNA packet is fully hardened.*

Proof. The packet has eight loci. A verified property of one feature, circuit, margin, dataset row, or proof row leaves the other loci and their ledgers untouched unless they are separately hardened or ledgered. □

Theorem 9462.228 (Fully hardened compatible NDNA is certified)**.** *If a neural theorem-DNA packet is fully hardened and NameCert-compatible, then it yields an NDNA-certified explanation of B within its seal scope.*

Proof. Full hardening supplies hardened critical loci and ledgered non-hardenable residue. NameCert compatibility supplies the Tensor NameCert projection. The seal supplies the scope in which the behavior is claimed to be explained. □

Theorem 9462.229 (Neural theorem-DNA theorem)**.** *Closed explanation of a neural behavior is a role-complete, anchored, ledgered, NameCert-compatible, and sealed profile, not a single feature label, circuit diagram, classifier, intervention result, or semantic tag.*

Proof. Definition 9462.209 rules out single-locus substitutes. Definition 9462.211 requires anchoring. Definition 9462.215 supplies the Tensor NameCert route. Definition 9462.218 and Definition 9462.220 record the causal and semantic boundary cases. The hardening theorems separate local formal verification from whole-packet closure. Together these facts make the complete packet the explanatory unit. □

Concretized in §6241: Classifier novelty ledgers realize before/after classifier-profile disagreement as finite BEDC certificate data.

Concretized in §6243: The theorem-DNA score packet realizes role-complete source, feature, circuit, classifier, stability, intervention, ledger, and seal rows as finite NameCert data.

9462.10 Explanation Complexity and Interpretability Information

section 9462.9 identifies the complete explanatory unit for a neural behavior as a role-complete, anchored, ledgered, and NameCert-compatible packet. This section isolates the evaluation problem

left by that packet. A shorter explanation is not automatically better; a longer explanation is not automatically worse; a formally hardened component is not automatically a discovery; and a larger feature list is not automatically more interpretability information. The auditable quantity is net explanation information: benefit minus score and remaining debt, with classifier novelty reported as a separate coordinate.

Definition 9462.230 (Explanation object)**.** *An explanation object is a tuple*

$$E = (B, C, N, \Lambda)$$

where B is a neural behavior target, C is a classifier supporting B, N is a NameCert or Tensor NameCert for C, and Λ is the ledger attached to the explanation claim. When the explanation is role-complete, the same object may be displayed as

$$E_B = (B, \mathsf{NDNA}(B), N, \Lambda).$$

Definition 9462.231 (Explanation scope)**.** *The scope of E is the tuple*

$$\mathrm{Scope}(E) = (D, M_\theta, \mathcal{A}, B),$$

where D is the source or target domain, M_θ is the model state, $\mathcal{A}$ is the admitted family of transformations, perturbations, or transfers, and B is the behavior target. All benefit, cost, and debt judgments are read inside this scope unless a larger scope is explicitly certified.

Definition 9462.232 (Cost protocol)**.** *A cost protocol is a triple*

$$\mathcal{C} = (Q, \mathrm{Cost}, \mathbf{w}),$$

where Q is an ordered commutative monoid, Cost *assigns costs to auditable objects, and $\mathbf{w}$ is a nonnegative weight profile. Auditable objects include source descriptions, feature records, circuit records, classifier specifications, stability witnesses, intervention records, ledgers, seals, semantic assignments, formal verification artifacts, and proof objects.*

Definition 9462.233 (Extended explanation complexity vector)**.** *For an explanation object E, the extended complexity vector is*

$$\mathbf{C}_{\mathsf{exp}}(E) = (c_{\mathsf{source}}, c_{\mathsf{feature}}, c_{\mathsf{circuit}}, c_{\mathsf{classifier}}, c_{\mathsf{stability}}, c_{\mathsf{intervention}}, c_{\mathsf{ledger}}, c_{\mathsf{seal}}, c_{\mathsf{semantic}}, c_{\mathsf{verification}}).$$

The first eight coordinates refine the NDNA score of Definition 9462.221; the semantic coordinate counts semantic assignment and semantic-ledger cost; the verification coordinate counts hardening artifacts and verification ledger cost.

Definition 9462.234 (Explanation score)**.** *For a nonnegative weight profile*

$$\mathbf{w} = (w_{\mathsf{source}}, w_{\mathsf{feature}}, w_{\mathsf{circuit}}, w_{\mathsf{classifier}}, w_{\mathsf{stability}}, w_{\mathsf{intervention}}, w_{\mathsf{ledger}}, w_{\mathsf{seal}}, w_{\mathsf{semantic}}, w_{\mathsf{verification}}),$$

the explanation score is

$$\mathrm{Score}_{\mathbf{w}}(E) = \mathbf{w} \cdot \mathbf{C}_{\mathsf{exp}}(E).$$

Theorem 9462.235 (Explanation score is monotone)**.** *If*

$$\mathbf{C}_{\mathsf{exp}}(E) \leq \mathbf{C}_{\mathsf{exp}}(E')$$

componentwise and all weights are nonnegative, then

$$\mathrm{Score}_{\mathbf{w}}(E) \leq \mathrm{Score}_{\mathbf{w}}(E').$$

Proof. Write the two complexity vectors as $(c_1, \ldots, c_n)$ and $(c'_1, \ldots, c'_n)$ with $c_j \leq c'_j$ for every j. Since each weight w_j is nonnegative, $w_j c_j \leq w_j c'_j$. Summing the coordinate inequalities gives the score inequality. □

Concretized in §6181: Explanation-complexity information is realized as a finite score, debt, classifier-novelty, and certified-quality NameCert packet.

Concretized in §8113: Description-complexity packets realize the explanation-complexity vision as finite source-description, score-vector, public-weight, debt, classifier-novelty, minimal-witness, replay, provenance, and NameCert rows.

Theorem 9462.236 (Scalar score requires public weights)**.** *A scalar explanation score is not auditable unless its weight profile is public.*

Proof. Two explanations can trade off source cost against ledger cost, or feature cost against verification cost. A weight profile concentrated on one coordinate orders the pair differently from a profile concentrated on another. Without the declared weights, the scalar order has no stable audit meaning. □

Definition 9462.237 (Explanation benefit)**.** *The benefit of E is a nonnegative audit functional*

$$\mathrm{Benefit}(E) = \mathrm{CoverageGain}(E) + \mathrm{PredictionGain}(E) + \mathrm{TransferGain}(E) + \mathrm{InterventionGain}(E) + \mathrm{VerificationGain}(E$$

The terms measure, respectively, behavior coverage, predictive value, transport across domains or model states, controlled intervention value, formal hardening value, and unification of previously separated behaviors, features, circuits, or classifiers.

Definition 9462.238 (Explanation debt)**.** *The total explanation debt is*

$$\mathrm{Debt}(E) = \mathrm{Debt}_{\mathsf{NDNA}}(E) + \mathrm{Debt}_{\mathsf{ledger}}(E) + \mathrm{Debt}_{\mathsf{semantic}}(E) + \mathrm{Debt}_{\mathsf{verification}}(E) + \mathrm{Debt}_{\mathsf{intervention}}(E) + \mathrm{Debt}_{\mathsf{generalization}}($$

The summands record missing NDNA loci, missing required residue, unlicensed semantic labels, unverified critical components, unledgered causal residue, and unsupported target-domain claims.

Theorem 9462.239 (Closed explanation has zero critical debt)**.** *If E is closed within its declared scope, then its critical debt is zero.*

Proof. Closure within a scope requires the critical source, classifier, stability, ledger, semantic, intervention, verification, generalization, and seal obligations used by the claim to be either discharged or placed outside the not-claimed boundary. Therefore no critical residue remains unrecorded inside the declared scope. □

Corollary 9462.240 (Critical debt blocks strong explanation claims)**.** *If E has nonzero critical debt, it cannot claim closed explanation, positive discovery, global generalization, or full verification within that debt-bearing scope.*

Proof. Each listed claim requires the relevant critical obligations to be closed. Nonzero critical debt leaves at least one such obligation open. □

Definition 9462.241 (Net explanation information)**.** *Relative to a cost protocol, the net explanation information of E is*

$$\mathsf{Info}_{\mathbf{w}}(E) = \mathrm{Benefit}(E) - \mathrm{Score}_{\mathbf{w}}(E) - \mathrm{Debt}(E).$$

For a transition $E_0 \rightsquigarrow E_1$, the relative net gain is

$$\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) = \mathsf{Info}_{\mathbf{w}}(E_1) - \mathsf{Info}_{\mathbf{w}}(E_0).$$

Theorem 9462.242 (Lower score does not imply positive net information)**.** *The inequality*

$$\mathrm{Score}_{\mathbf{w}}(E_1) < \mathrm{Score}_{\mathbf{w}}(E_0)$$

does not imply

$$\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) > 0.$$

Proof. Net information subtracts both score and debt from benefit. A lower-score explanation may lower benefit by removing coverage, predictive value, transfer value, or ledger rows. It may also increase debt by hiding boundary cases. Either change can offset the score decrease. □

Theorem 9462.243 (Higher score can still carry positive net information)**.** *An explanation transition can have positive net information even when the target explanation has higher score.*

Proof. Adding source rows, intervention witnesses, transfer witnesses, semantic ledgers, and verification artifacts increases score. If those rows increase benefit and reduce debt by more than the score increase, then the net information difference is positive by Definition 9462.241. □

Definition 9462.244 (Interpretability classifier novelty)**.** *For an explanation object E, let*

$$\mathcal{T}_E = (S_E, P_E, L_E, C_E, N_E)$$

be its certified classifier state. A transition $E_0 \rightsquigarrow E_1$ has interpretability classifier novelty when $\mathcal{T}_{E_0} \to \mathcal{T}_{E_1}$ has classifier shift in the sense of Definition 9462.49.

Definition 9462.245 (Positive interpretability discovery)**.** *A transition $E_0 \rightsquigarrow E_1$ is a positive interpretability discovery when it has interpretability classifier novelty, the target classifier state has NameCert and ledger witnesses, and*

$$\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) > 0.$$

Theorem 9462.246 (Explanation improvement need not be discovery)**.** *Positive net explanation information does not imply interpretability classifier novelty.*

Proof. The net information gain may come from lower ledger debt, lower score, hardening of an existing critical component, source-scope clarification, or a better not-claimed boundary while the classifier relation remains unchanged. By Definition 9462.51, classifier novelty requires classifier shift, so such an improvement is not necessarily discovery. □

Theorem 9462.247 (Interpretability discovery need not be positive)**.** *Interpretability classifier novelty with NameCert and ledger witnesses does not by itself imply positive net explanation information.*

Proof. A changed certified classifier can require high source, stability, intervention, ledger, semantic, or verification cost. If those costs and remaining debts exceed the coverage, prediction, transfer, intervention, verification, and unification benefits, then the net information difference is nonpositive. □

Theorem 9462.248 (Positive discovery requires two audits)**.** *Positive interpretability discovery requires both classifier novelty and positive net explanation information.*

Proof. This is exactly the definition of positive interpretability discovery: the transition must pass the classifier-shift audit and the net information audit. □

Theorem 9462.249 (Counts do not determine interpretability information)**.** *Feature count, proof count, ledger-row count, and raw explanation length do not determine net explanation information.*

Proof. Additional features may be unstable, redundant, unanchored, or unledgered. Additional proofs may harden low-value rows while leaving critical components untouched. Additional ledger rows may avoid the required critical residue. Shorter text may remove evidence and boundary statements. Each count can therefore vary independently of benefit, score, and debt. □

Corollary 9462.250 (Feature discovery requires certificate value)**.** *Finding more features counts as interpretability information only when the resulting explanation changes certified classifier value, lowers debt, or increases benefit after cost is counted.*

Proof. By Definition 9462.249, feature count alone is not an information measure. The feature must enter the certified explanation surface. □

Corollary 9462.251 (Proof quantity is not explanation quality)**.** *The number of formal proof artifacts is not a standalone explanation quality metric.*

Proof. By Definition 9462.249, proof count does not say which critical loci were hardened or whether verification debt decreased. □

Definition 9462.252 (Explanation quality vector)**.** *The explanation quality vector is*

$$\mathbf{Q}(E) = (\mathrm{Benefit}(E), -\mathrm{Score}_{\mathbf{w}}(E), -\mathrm{Debt}(E), \mathrm{Novelty}(E), \mathrm{VerificationGain}(E)).$$

Here $\mathrm{Novelty}(E)$ *records whether the transition under discussion has interpretability classifier novelty.*

Definition 9462.253 (Pareto comparison)**.** *An explanation* E_1 *Pareto-dominates* E_0 *when every declared coordinate of* $\mathbf{Q}(E_1)$ *is at least the corresponding coordinate of* $\mathbf{Q}(E_0)$*, and at least one coordinate is strictly larger.*

Theorem 9462.254 (No hidden scalar captures all explanation dimensions)**.** *No unpublished scalar score can audibly replace the benefit, score, debt, classifier-novelty, and verification coordinates.*

Proof. These coordinates can trade off. One explanation may be low cost and high debt; another may be high cost and low debt; a third may change the classifier but have low verification gain; a fourth may harden a prior classifier without classifier novelty. Any scalar ordering must encode weights for these tradeoffs. Without publishing those weights, the ordering is not auditable. □

Corollary 9462.255 (Explanation comparison needs vectors or public weights)**.** *Explanation comparisons should use quality vectors, Pareto frontiers, or public scalar weights.*

Proof. This follows from Definition 9462.254 and Definition 9462.236. □

Definition 9462.256 (Selection functional)**.** *A selection functional has the form*

$$\mathrm{Sel}(E) = \alpha\ \mathrm{CoverageGain}(E) + \beta\ \mathrm{PredictionGain}(E) + \gamma\ \mathrm{TransferGain}(E) + \delta\ \mathrm{InterventionGain}(E) + \eta\ \mathrm{Unificati}$$

with nonnegative coefficients. A research-facing selection may weight coverage, transfer, unification, and classifier novelty highly; a safety-facing selection may weight intervention debt, semantic debt, generalization debt, and not-claimed boundaries highly.

Theorem 9462.257 (Shortest explanation need not be selection-favorable)**.** *The shortest explanation in a candidate family need not maximize a selection functional.*

Proof. A short feature label may have low score but also low coverage, low prediction value, no classifier novelty, and high debt. A longer role-complete explanation may have higher score but much higher benefit and lower debt. Under a selection functional that values those terms, the longer explanation is preferred. □

Theorem 9462.258 (Ledger-heavy explanations can be safety-favorable)**.** *An explanation with more ledger material can be preferable in a safety-facing selection.*

Proof. Safety-facing selection penalizes unrecorded failure cases, off-target effects, distribution boundaries, semantic ambiguity, and unsupported generalization. Adding the relevant ledger rows increases score, but it can reduce debt enough to increase the selection value. □

Theorem 9462.259 (Debt reduction can justify higher complexity)**.** *If a transition increases explanation score but reduces black-box debt by a larger value after benefit is counted, then it has positive net explanation information.*

Proof. By Definition 9462.241, the score increase is subtracted and the debt decrease is added through the difference of the negative-debt term. If the debt decrease and benefit increase dominate the score increase, the net information difference is positive. This is the evaluation form of Definition 9462.202: recorded residue can turn hidden black-box boundary into auditable boundary. □

Theorem 9462.260 (Formal hardening has net gain only when it pays for itself)**.** *Formal hardening can increase or decrease net explanation information.*

Proof. Hardening increases verification cost. It improves net information only when the resulting verification gain and debt reduction exceed that cost and any remaining verification debt. If it hardens a low-value component or leaves critical residues untouched, its net contribution can be nonpositive. This refines Definition 9462.56: a hardened component may improve audit status without creating classifier novelty. □

Theorem 9462.261 (Explanation complexity and information theorem)**.** *Interpretability information is not explanation brevity. It is the net quantity obtained by auditing benefit, score, and debt, while classifier novelty is separately audited by classifier shift with certificate and ledger witnesses.*

Proof. Score is defined by Definition 9462.234; benefit by Definition 9462.237; debt by Definition 9462.238; and net explanation information by Definition 9462.241. Definition 9462.242 and Definition 9462.243 separate score from quality. Definition 9462.248 separates net information from classifier novelty. Definition 9462.249 rules out feature, proof, ledger, and length counts as standalone measures. The theorem follows. □

9462.11 Interpretability Classifier Novelty

section 9462.10 separates net explanation information from classifier novelty. This section isolates the novelty coordinate itself. A feature catalog, a larger circuit diagram, a stronger ledger, a hardened proof component, or a clearer semantic label can improve an explanation without changing the explanatory classifier. Interpretability discovery, in the classifier-level sense, occurs only when the certified classifier surface used by the explanation changes.

Definition 9462.262 (Interpretability explanation state)**.** *An interpretability explanation state is a certified classifier state read in the machine-learning explanation context:*

$$\mathcal{E} = (S_E, P_E, L_E, C_E, N_E).$$

Here S_E is the explanation source specification, P_E is the pattern specification, L_E is the ledger policy, C_E is the explanation classifier, and N_E is the NameCert licensing the state.

Definition 9462.263 (Explanation source scope and classifier cell)**.** *The source scope of $\mathcal{E}$ is*

$$|\mathcal{E}| = \{x \mid S_E(x)\}.$$

For $x \in |\mathcal{E}|$, the explanation classifier cell of x is

$$[x]_{\mathcal{E}} = \{y \in |\mathcal{E}| \mid C_E(x, y)\}.$$

The objects x and y may be input samples, tensor traces, activation records, behavior cases, NDNA packets, or explanation objects, depending on the declared source scope.

Definition 9462.264 (Interpretability transition)**.** *An interpretability transition is a passage*

$$\mathcal{E} \rightsquigarrow \mathcal{E}'$$

between explanation states. The passage may change feature loci, circuit loci, ledger rows, stability witnesses, semantic assignments, formal hardening status, NameCert rows, seal clauses, or the classifier itself.

Definition 9462.265 (Interpretability classifier equivalence)**.** *Two explanation states*

$$\mathcal{E} = (S, P, L, C, N), \qquad \mathcal{E}' = (S', P', L', C', N')$$

are interpretability-classifier equivalent when they give the same classifier judgment on their common source scope:

$$\forall x, y,\ S(x) \wedge S(y) \wedge S'(x) \wedge S'(y) \Rightarrow \big(C(x, y) \leftrightarrow C'(x, y)\big).$$

This is the explanation-side instance of Definition 9462.48.

Definition 9462.266 (Classifier novelty value)**.** *The classifier novelty value of a transition $\mathcal{E} \rightsquigarrow \mathcal{E}'$ is*

$$\nu_{\mathsf{cls}}(\mathcal{E}, \mathcal{E}') = 0$$

exactly when $\mathcal{E}$ and $\mathcal{E}'$ are interpretability-classifier equivalent, and is 1 exactly when that equivalence fails.

Definition 9462.267 (Local classifier novelty witness)**.** *For a displayed family*

$$X \subseteq |\mathcal{E}| \cap |\mathcal{E}'|,$$

a local classifier novelty witness is a pair $x, y \in X$ together with evidence that

$$C_E(x, y) \not\leftrightarrow C_{E'}(x, y).$$

The witness is certified when the target state carries NameCert and ledger rows explaining the changed judgment.

Theorem 9462.268 (Local witness gives classifier novelty)**.** *If there is a local classifier novelty witness for $\mathcal{E} \rightsquigarrow \mathcal{E}'$, then*

$$\nu_{\mathsf{cls}}(\mathcal{E}, \mathcal{E}') = 1.$$

Proof. The witness supplies x, y in the common source scope for which the two classifier judgments differ. Interpretability-classifier equivalence requires agreement on every such pair. Therefore equivalence fails, and the classifier novelty value is 1. □

Corollary 9462.269 (One classified pair can certify non-equivalence)**.** *One common-source pair with different classifier judgments is sufficient to show that two explanation states are not classifier-equivalent.*

Proof. Apply Definition 9462.268 to the displayed two-object family containing the witness pair. □

9462.11.1 Nonclassifier Changes

Definition 9462.270 (Feature-side change)**.** *Concretized in §6138: Classifier-novelty witnesses realize the local explanation-state non-equivalence row as finite NameCert data.*

A feature-side change is a transition that alters the feature locus, feature functional, activation source, sparse-autoencoder latent, direction, probe, or feature evidence while preserving the classifier relation on the common source scope.

Definition 9462.271 (Circuit-side change)**.** *A circuit-side change alters the circuit locus, route graph, layer path, attention component, MLP component, residual route, or causal-route evidence while preserving the classifier relation on the common source scope.*

Definition 9462.272 (Ledger-side change)**.** *A ledger-side change adds or reorganizes ledger rows, reduces debt, or clarifies not-claimed boundaries while preserving the classifier relation on the common source scope.*

Definition 9462.273 (Hardening-side and semantic-side changes)**.** *A hardening-side change alters evidence mode, proof status, or verification ledger without changing the explanation classifier. A semantic-side change alters labels, semantic ledgers, examples, counterexamples, or concept boundaries without changing the explanation classifier.*

Theorem 9462.274 (Feature-side change need not create classifier novelty)**.** *A feature-side change whose classifier relation is preserved has classifier novelty value* 0*.*

Proof. By definition, the classifier relation agrees on the common source scope. Thus the two explanation states are interpretability-classifier equivalent, so the novelty value is 0. □

Theorem 9462.275 (Circuit-side change need not create classifier novelty)**.** *A circuit-side change whose classifier relation is preserved has classifier novelty value* 0*.*

Proof. Changing circuit evidence can refine mechanism without changing the classifier cells. If the common-source classifier judgments are the same, Definition 9462.266 gives value 0. □

Theorem 9462.276 (Ledger repair need not create classifier novelty)**.** *Ledger-side improvement can lower debt without creating classifier novelty.*

Proof. A ledger-side change may add boundary cases, false-positive rows, threshold rows, or not-claimed clauses while leaving C_E unchanged. Then the classifier equivalence relation is unchanged, even though the audit quality improves. □

Theorem 9462.277 (Hardening and semantic changes need not create classifier novelty)**.** *Hardening-side and semantic-side changes do not create classifier novelty unless they change the classifier relation.*

Proof. Formal hardening changes evidence strength; semantic clarification changes naming and boundary prose. If neither changes which source objects are classified together, the explanation states remain classifier-equivalent. □

Corollary 9462.278 (Improvement and discovery are distinct)**.** *An explanation can become more useful, reliable, precise, or auditable without becoming a classifier-level discovery.*

Proof. The preceding theorems describe feature, circuit, ledger, hardening, and semantic changes that may improve an explanation while preserving the classifier relation. □

9462.11.2 Classifier-Change Modes

Definition 9462.279 (Classifier refinement)**.** *A transition is a classifier refinement when*

$$C_{E'}(x, y) \Rightarrow C_E(x, y)$$

on the common source scope, and the implication is strict for at least one pair. The target classifier separates at least one pair that the source classifier grouped.

Definition 9462.280 (Classifier coarsening)**.** *A transition is a classifier coarsening when*

$$C_E(x, y) \Rightarrow C_{E'}(x, y)$$

on the common source scope, and the implication is strict for at least one pair. The target classifier groups at least one pair that the source classifier separated.

Definition 9462.281 (Classifier reorganization)**.** *A transition is a classifier reorganization when the two classifiers are not equivalent on their common source scope, but the change is neither a pure refinement nor a pure coarsening.*

Definition 9462.282 (Bridge, boundary, and unification modes)**.** *A bridge mode connects previously separated explanation states through a common certified classifier. A boundary mode identifies a principled limit beyond which the prior classifier should not be used. A unification mode gives one certified classifier surface for several previously separated behaviors, features, circuits, or domains.*

Theorem 9462.283 (Refinement creates classifier novelty)**.** *Every strict classifier refinement has classifier novelty value* 1*.*

Proof. Strictness supplies x, y in the common source scope with $C_E(x, y)$ and $\neg C_{E'}(x, y)$. This is a local classifier novelty witness. □

Theorem 9462.284 (Coarsening creates classifier novelty)**.** *Every strict classifier coarsening has classifier novelty value* 1*.*

Proof. Strictness supplies x, y in the common source scope with $\neg C_E(x, y)$ and $C_{E'}(x, y)$. This is again a local classifier novelty witness. □

Theorem 9462.285 (Reorganization creates classifier novelty)**.** *Every classifier reorganization has classifier novelty value* 1*.*

Proof. Reorganization is defined by failure of classifier equivalence on the common source scope. The classifier novelty value is therefore 1. □

Corollary 9462.286 (Discovery need not only refine)**.** *Classifier-level interpretability discovery may occur by refinement, coarsening, reorganization, bridge, boundary, or unification modes.*

Proof. The first three modes directly change classifier equivalence. The bridge, boundary, and unification modes are discovery modes when their certified classifier surface is not equivalent to the prior one. □

9462.11.3 Discovery Grades

Definition 9462.287 (Discovery candidate)**.** *An interpretability transition is a discovery candidate when its classifier novelty value is* 1 *and the target state has candidate NameCert and ledger rows whose closure status is still under audit.*

Definition 9462.288 (Certified interpretability discovery)**.** *An interpretability transition is a certified interpretability discovery when its classifier novelty value is* 1 *and the target state carries a complete NameCert and ledger for the changed classifier surface.*

Definition 9462.289 (Positive interpretability discovery grade)**.** *A certified interpretability discovery is positive when its relative net explanation information is positive:*

$$\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) > 0.$$

Definition 9462.290 (Ledger-only, hybrid, and fully hardened discovery)**.** *A ledger-only interpretability discovery has complete NameCert and ledger rows but no hardened formal component. A hybrid discovery combines empirical, paper, ledger, and formal evidence. A fully hardened discovery has all hardenable critical components formally hardened and all non-hardenable critical residue ledgered.*

Theorem 9462.291 (Certified discovery requires NameCert and ledger)**.** *Every certified interpretability discovery carries NameCert and ledger rows for the changed classifier surface.*

Proof. Those rows are part of the definition of certified interpretability discovery. This is the interpretability-specific instance of Definition 9462.52. □

Theorem 9462.292 (Positive discovery requires novelty and net gain)**.** *Every positive interpretability discovery has classifier novelty value* 1 *and positive relative net explanation information.*

Proof. Positive discovery is defined only after certified discovery has been established, and certified discovery requires classifier novelty. The positive grade then adds the inequality $\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) > 0$. □

Theorem 9462.293 (Discovery candidate is not certified discovery)**.** *A discovery candidate whose NameCert or ledger is incomplete is not a certified interpretability discovery.*

Proof. Certified discovery requires complete NameCert and ledger rows for the changed classifier surface. A candidate missing either row lacks the certificate surface required by the definition. □

Theorem 9462.294 (Claim demotion preserves audit soundness)**.** *If an asserted discovery claim lacks classifier novelty, NameCert, or ledger, replacing the claim by the strongest supported feature-side, circuit-side, ledger-side, hardening-side, or semantic-side claim preserves audit soundness.*

Proof. The replacement claim says only what the evidence supports. If the classifier did not change, the result is not classifier-level discovery. If NameCert or ledger rows are missing, the result may be a candidate or an audit improvement, but not a certified discovery. Stating the weaker claim prevents overclaiming. □

9462.11.4 NDNA, Debt, and Hardening

Definition 9462.295 (NDNA classifier-locus novelty)**.** *An NDNA transition has classifier-locus novelty when the classifier locus changes to a classifier not equivalent to the prior classifier on the common source scope.*

Definition 9462.296 (NDNA nonclassifier-locus change)**.** *An NDNA nonclassifier-locus change alters source, feature, circuit, stability, intervention, ledger, or seal loci while preserving the classifier locus.*

Theorem 9462.297 (NDNA classifier-locus novelty gives interpretability novelty)**.** *Every nontrivial NDNA classifier-locus novelty gives classifier novelty value* 1 *for the associated explanation transition.*

Proof. The classifier locus records the explanation classifier. If that locus changes to a nonequivalent classifier, there is a common-source pair whose classifier judgment differs, so Definition 9462.268 applies. □

Theorem 9462.298 (NDNA nonclassifier change need not create novelty)**.** *An NDNA transition that changes only nonclassifier loci need not create classifier novelty.*

Proof. If the classifier locus is preserved, the explanation classifier remains equivalent on the common source scope. The transition may still lower debt, add evidence, or close the seal. □

Definition 9462.299 (Classifier debt)**.** *Classifier debt is the debt carried by a missing classifier locus, an ambiguous classifier relation, an unrecorded classifier boundary, a threshold gap, or an unledgered false split or false merge.*

Theorem 9462.300 (Classifier debt reduction need not create novelty)**.** *Reducing classifier debt does not by itself create classifier novelty.*

Proof. Debt can be reduced by stating the old classifier more clearly, adding boundary rows, recording threshold sensitivity, or clarifying source scope. These changes can preserve the classifier relation. □

Theorem 9462.301 (Critical classifier debt can force classifier novelty)**.** *If repairing critical classifier debt forces a change in the classifier boundary, then the repair produces classifier novelty.*

Proof. Changing the classifier boundary means that some common-source pair is classified differently after the repair. That pair is a local classifier novelty witness. □

Theorem 9462.302 (Verification-only hardening has no classifier novelty)**.** *Formal hardening that preserves the explanation classifier has classifier novelty value* 0*.*

Proof. Verification-only hardening changes evidence mode or proof status. If the classifier relation is preserved on the common source scope, the explanation states remain classifier-equivalent. □

Theorem 9462.303 (Formal classifier discovery requires a classifier witness)**.** *A formal verification step that claims classifier-level discovery must display a witness that the verified target classifier is not equivalent to the prior classifier.*

Proof. Without such a witness, the formal step may harden an existing component but does not establish classifier novelty. A classifier-level discovery claim requires failure of classifier equivalence. □

Theorem 9462.304 (Interpretability classifier novelty theorem)**.** *Classifier-level interpretability discovery is certified classifier shift, not feature accumulation, circuit elaboration, ledger repair, semantic relabeling, or verification-only hardening.*

Proof. Feature, circuit, ledger, hardening, and semantic changes can preserve classifier equivalence by Definition 9462.274, Definition 9462.275, Definition 9462.276, and Definition 9462.277. Refinement, coarsening, and reorganization create classifier novelty by Definition 9462.283, Definition 9462.284, and Definition 9462.285. Certified discovery additionally requires NameCert and ledger by Definition 9462.291, and positive discovery additionally requires positive net explanation information by Definition 9462.292. □

9462.12 Interpretability Classifier Novelty in Practice

section 9462.11 separates classifier-level discovery from feature-side, circuit-side, ledger-side, semantic-side, and verification-side improvement. This section records the practical audit taxonomy for machine-learning interpretability work: what kind of event occurred, whether it changed the certified classifier surface, and whether the resulting lineage is optimization or discovery.

Definition 9462.305 (Interpretability event kinds)**.** *Feature events identify feature functionals or activation patterns. Classifier events identify changed classifier relations. Certificate events construct or repair NameCert rows. Ledger events discover critical residue or failure boundaries. Verification events harden formal components.*

Theorem 9462.306 (Feature events are not necessarily classifier events)**.** *A feature event need not be a classifier event.*

Proof. A feature functional may be redundant with the prior classifier or may serve only as another implementation route for the same classifier cells. Unless the classifier relation changes, classifier novelty is absent. □

Theorem 9462.307 (Ledger events can become classifier events)**.** *A ledger event becomes a classifier event when the critical residue it records forces the classifier boundary to change.*

Proof. Critical residue can expose a false split, false merge, threshold boundary, or unsupported source region. If correcting that residue changes the classifier judgment on a common-source pair, the transition has classifier novelty. □

Theorem 9462.308 (Certificate events are not necessarily classifier events)**.** *Constructing or improving a NameCert for an existing classifier need not be a classifier event.*

Proof. The certificate event can improve audit status while leaving the classifier relation unchanged. It is then certification progress rather than classifier-level discovery. □

Definition 9462.309 (Interpretability lineage)**.** *An interpretability lineage is a finite chain of explanation states*

$$\mathcal{E}_0 \rightsquigarrow \mathcal{E}_1 \rightsquigarrow \cdots \rightsquigarrow \mathcal{E}_n.$$

It is an optimization lineage when every edge has classifier novelty value 0 *while some edge lowers score, debt, or hardening gap. It is a discovery lineage when at least one edge has classifier novelty value* 1*.*

Theorem 9462.310 (Optimization lineage need not be discovery lineage)**.** *An interpretability lineage can improve explanation quality without being a discovery lineage.*

Proof. A lineage may add source rows, ledger rows, negative examples, stability evidence, or formal hardening while preserving the classifier relation at every edge. Then its classifier novelty value is always 0, although score, debt, or hardening gap may improve. □

Theorem 9462.311 (Discovery lineage need not be immediately positive)**.** *A discovery lineage can have nonpositive net explanation information at the edge where classifier novelty first appears.*

Proof. A changed classifier may require additional source, stability, ledger, seal, semantic, and verification work. Before the resulting benefit and debt reduction dominate those costs, the net information difference can be nonpositive. □

Definition 9462.312 (Classifier-prioritized search)**.** *Classifier-prioritized search ranks candidates first by evidence of classifier novelty, then audits NameCert closure, ledger closure, and net explanation information. Feature-prioritized search ranks candidates first by feature events.*

Theorem 9462.313 (Classifier-prioritized search differs from feature-prioritized search)**.** *Classifier-prioritized search and feature-prioritized search can select different candidates.*

Proof. By Definition 9462.306, feature events need not change classifiers. A search that ranks feature events can therefore select a candidate with no classifier novelty, while a classifier-prioritized search selects a candidate with a changed classifier surface. □

Theorem 9462.314 (Practice audit theorem)**.** *Interpretability reports should state separately whether a result is a feature event, classifier event, certificate event, ledger event, verification event, optimization lineage, discovery lineage, or positive discovery lineage.*

Proof. Feature, certificate, ledger, and verification events can improve an explanation without classifier novelty. Classifier events can introduce novelty before becoming positive in net information. Optimization and discovery lineages therefore track different audit coordinates and should not be collapsed into one report label. □

Concretized in §6225: Practice-level classifier shifts are recorded as finite NameCert data.

9462.13 Positive Interpretability Discovery

section 9462.11 defines classifier novelty as the discovery axis, and section 9462.10 defines net explanation information as benefit minus score and debt. Positive interpretability discovery is the conjunction of these two audits: the explanation must introduce a certified classifier shift, and the shift must pay for its certificate, ledger, and remaining debt under a public cost protocol.

Definition 9462.315 (Certified discovery transition)**.** *A transition $E_0 \rightsquigarrow E_1$ is a certified discovery transition when it has classifier novelty, the target classifier state has a NameCert, the changed classifier surface has ledger rows, the classifier shift has a local or formal witness, and the seal records the not-claimed boundary.*

Definition 9462.316 (Positive discovery transition)**.** *A certified discovery transition is positive when*

$$\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1) > 0$$

for the public cost protocol used by the claim. It is negative when the quantity is below zero and neutral when it is zero.

Definition 9462.317 (Strong positive discovery)**.** *A positive discovery transition is strong when the target explanation has zero critical debt, a scope seal, reproducible classifier and ledger evidence, a public cost protocol, and a NameCert-compatible NDNA packet.*

Definition 9462.318 (False positive discovery claim)**.** *A claim is a false positive discovery claim when it asserts positive interpretability discovery while lacking classifier novelty, NameCert, ledger, positive net explanation information, a public cost protocol, or a declared not-claimed boundary.*

Theorem 9462.319 (Positive discovery implies certified discovery)**.** *Every positive discovery transition is a certified discovery transition.*

Proof. Positive discovery is defined by adding the strict net-information inequality to certified discovery. Hence all certified-discovery requirements are inherited. □

Corollary 9462.320 (Positive discovery requires classifier novelty)**.** *Every positive discovery transition has classifier novelty.*

Proof. Certified discovery requires classifier novelty by Definition 9462.315, and positive discovery implies certified discovery. □

Corollary 9462.321 (Positive discovery requires NameCert and ledger)**.** *Every positive discovery transition carries NameCert and ledger rows for the changed classifier surface.*

Proof. These rows are part of certified discovery, which every positive discovery transition satisfies. □

Theorem 9462.322 (Classifier novelty alone is not positive discovery)**.** *Classifier novelty by itself does not imply positive interpretability discovery.*

Proof. A changed classifier can be expensive to source, stabilize, ledger, intervene on, verify, or seal. It may also lack complete NameCert or ledger rows. If the net information is nonpositive, or if the certificate surface is incomplete, the transition is not positive discovery. □

Theorem 9462.323 (Positive information alone is not discovery)**.** *Positive net explanation information does not imply positive interpretability discovery.*

Proof. The net gain can come from ledger completion, lower score, verification debt reduction, source-scope clarification, semantic-ledger completion, or a better not-claimed boundary while the classifier relation is unchanged. Such a transition has no classifier novelty and is therefore not discovery. □

Theorem 9462.324 (Ledger completion can be positive without discovery)**.** *Ledger completion can yield positive net explanation information while preserving the classifier.*

Proof. Completing a critical ledger can lower debt more than it increases score. If the classifier relation is unchanged, the transition is a positive audit improvement rather than classifier-level discovery. □

Theorem 9462.325 (Formal hardening can be positive without discovery)**.** *Formal hardening can yield positive net explanation information while preserving the classifier.*

Proof. Hardening can lower verification debt and increase reuse. If its gain exceeds its verification cost while the classifier relation remains unchanged, it is a positive hardening improvement rather than positive interpretability discovery. □

9462.13.1 Cost Protocols

Definition 9462.326 (Positive discovery cost protocol)**.** *A positive discovery cost protocol specifies the benefit functional, score functional, debt functional, weight profile, source and target scope, and not-claimed boundary used to evaluate* $\Delta\mathsf{Info}_{\mathbf{w}}(E_0, E_1)$.

Definition 9462.327 (Cost opacity and cost debt)**.** *Cost opacity occurs when a positive discovery claim hides its benefit, score, debt, weight, scope, or excluded-claim conventions. Cost debt is the uncounted cost or debt created by omitting a critical term such as ledger cost, semantic debt, verification debt, intervention debt, or generalization debt.*

Concretized in §6202: Positive interpretability classifiers realize the certified-discovery and positive-cost-protocol surface as finite classifier-shift, ledger, debt, and NameCert rows.

Concretized in §7414: Positive discovery debt ledgers realize the cost-protocol and critical-debt rows for the certified-discovery route.

Theorem 9462.328 (Positive discovery requires protocol completeness)**.** *A positive discovery claim is auditable only when its cost protocol is complete and public.*

Proof. The sign of $\Delta\mathsf{Info}_{\mathbf{w}}$ depends on benefit, score, debt, and weights inside a declared scope. If any of these terms is hidden, the strict positivity claim cannot be checked. □

Theorem 9462.329 (Cost debt can invalidate positivity)**.** *Adding omitted critical cost or debt terms can turn an apparently positive discovery claim into a nonpositive one.*

Proof. Net information subtracts score and debt. If omitted cost or debt is restored, the computed net gain decreases. When the omitted amount exceeds the original positive margin, the strict positivity inequality fails. □

Corollary 9462.330 (Positive discovery must count critical debts)**.** *Ledger, semantic, verification, intervention, and generalization debts must be included whenever they are relevant to the claim.*

Proof. These are critical debt classes for interpretability claims. Omitting them creates cost debt, which can invalidate positivity by Definition 9462.329. □

9462.13.2 Benefit Types

Definition 9462.331 (Positive discovery benefit modes)**.** *Positive discovery may be coverage-positive, prediction-positive, transfer-positive, intervention-positive, unification-positive, verification-positive, or ledger-positive, according to the benefit term that principally drives the positive net information inequality.*

Theorem 9462.332 (Benefit modes can overlap)**.** *A positive discovery transition can have several benefit modes at once.*

Proof. The benefit functional is a sum of coverage, prediction, transfer, intervention, verification, and unification terms. More than one term can make a material contribution to the positive margin. □

Corollary 9462.333 (Positive discovery should report benefit structure)**.** *A positive discovery claim should state which benefit terms drive its positive margin.*

Proof. Since benefit modes can overlap and trade off against costs and debts, reporting only the final sign hides the reason the transition is positive. □

9462.13.3 Strong and NDNA-Positive Discovery

Definition 9462.334 (NDNA-positive discovery criterion)**.** *A transition satisfies the NDNA-positive criterion when it has classifier novelty, the target NDNA is complete, anchored, NameCert-compatible, and ledger-complete, and the NDNA-level net information difference is positive.*

Theorem 9462.335 (NDNA-positive criterion implies positive discovery)**.** *Every transition satisfying the NDNA-positive criterion is a positive interpretability discovery.*

Proof. Classifier novelty supplies the discovery axis. Complete, anchored, NameCert-compatible NDNA supplies the certified explanation surface, and ledger completeness supplies the required ledger. The positive NDNA-level net information difference supplies the positivity condition. □

Theorem 9462.336 (NDNA completeness alone is not positive discovery)**.** *Complete NDNA alone does not imply positive interpretability discovery.*

Proof. Completeness says that all eight loci are present. It does not state that the classifier differs from the prior classifier, that the NameCert projection is compatible, or that net explanation information is positive. □

Theorem 9462.337 (Strong positive discovery implies positive discovery)**.** *Every strong positive discovery is a positive discovery.*

Proof. Strong positive discovery includes positive discovery and adds zero critical debt, scope sealing, reproducibility, a public cost protocol, and NameCert-compatible NDNA. □

Theorem 9462.338 (Positive discovery need not be strong)**.** *A positive discovery transition need not be strong.*

Proof. A transition can have classifier novelty, NameCert, ledger, and positive net information while still retaining noncritical debt, an incomplete reproducibility package, a partially developed seal, or an unsettled hardening frontier. □

9462.13.4 Black-Box Resolution and Verification

Definition 9462.339 (Positive black-box resolution)**.** *A transition is a positive black-box resolution when it lowers black-box debt by more than the additional explanation score charged to that resolution inside the declared scope.*

Definition 9462.340 (Positive discovery-resolution)**.** *A transition is a positive discovery-resolution when it is both a positive interpretability discovery and a positive black-box resolution.*

Theorem 9462.341 (Positive black-box resolution need not be discovery)**.** *Positive black-box resolution does not imply interpretability discovery.*

Proof. Completing the ledger of an existing classifier can reduce black-box debt more than it increases score. If the classifier remains unchanged, the transition resolves black-box residue but does not discover a classifier. □

Theorem 9462.342 (Positive discovery-resolution is the strongest resolution form)**.** *A positive discovery-resolution both changes the certified classifier surface and positively reduces black-box debt in net terms.*

Proof. The first condition follows from positive interpretability discovery. The second follows from positive black-box resolution. Thus the transition improves both the classifier axis and the black-box debt axis. □

Definition 9462.343 (Verification-assisted positive discovery)**.** *A positive discovery is verification-assisted when formal hardening contributes to the positive margin. It is verification-dependent when removing the verification benefit makes the net information difference nonpositive.*

Theorem 9462.344 (Verification assistance does not replace classifier novelty)**.** *Verification-assisted positive discovery still requires classifier novelty.*

Proof. The verification term can contribute to benefit, but positive discovery also requires certified classifier shift. A verification-only transition without classifier novelty is a hardening improvement, not discovery. □

Theorem 9462.345 (Verification-assisted discovery must count verification cost)**.** *If formal hardening contributes to a positive discovery claim, then verification cost and verification-ledger debt must be included.*

Proof. Formal hardening can increase benefit, but it also adds proof cost, dependency cost, trust-boundary obligations, and verification-ledger residue. Omitting those terms overstates net information. □

9462.13.5 Lineage and Anti-Laundering

Definition 9462.346 (Positive discovery lineages)**.** *A positive discovery lineage contains at least one positive discovery transition. A pre-positive discovery lineage contains a certified discovery step that is not yet positive. A delayed-positive lineage becomes positive only after later ledger repair, compression, hardening, transfer, or unification work. A transient-negative lineage has an initial discovery step whose net information is negative.*

Theorem 9462.347 (Discovery can become positive later)**.** *A certified discovery transition with nonpositive net information can become part of a positive discovery lineage after later audit work.*

Proof. The initial classifier shift can impose high source, stability, ledger, seal, semantic, or verification costs. Later work can lower debt, reduce score, increase transfer, or improve unification. The total transition from the original state to the later state can then have positive net information. □

Theorem 9462.348 (Positive status can decay under new debt)**.** *A positive discovery claim can lose positive status when later audit reveals critical debt large enough to erase the positive margin.*

Proof. Positive status depends on $\Delta\mathsf{Info}_{\mathbf{w}} > 0$. Newly recorded critical debt increases the debt term. If it exceeds the margin, the inequality no longer holds. □

Definition 9462.349 (Discovery laundering)**.** *Scope laundering creates apparent positive gain by hiding a scope narrowing. Debt laundering hides ledger, semantic, verification, intervention, or generalization debt. Classifier laundering presents feature-side, ledger-side, hardening-side, or semantic-side improvement as classifier novelty.*

Theorem 9462.350 (Laundering invalidates positive discovery claims)**.** *Any positive discovery claim that relies on scope, debt, or classifier laundering is invalid.*

Proof. Scope laundering changes the domain of the benefit and debt calculation without sealing the change. Debt laundering suppresses terms subtracted by net information. Classifier laundering asserts classifier novelty where the classifier relation is unchanged. Each case violates a necessary audit condition for positive discovery. □

Theorem 9462.351 (Positive interpretability discovery theorem)**.** *Positive interpretability discovery is certified classifier novelty with positive net explanation information under a public, debt-complete, scope-sealed cost protocol.*

Proof. Definition 9462.319 gives certified discovery, including classifier novelty, NameCert, ledger, witness, and not-claimed boundary. Definition 9462.316 supplies the strict net-information inequality. Definition 9462.328 and Definition 9462.330 require a public protocol and critical debt accounting. Definition 9462.350 rules out apparent gains obtained by hiding scope, debt, or classifier non-novelty. These conditions are exactly the stated characterization. □

9462.14 Classifier-Forming Learning

section 9462.13 defines positive discovery as certified classifier novelty with positive net explanation information. The same separation applies to learning itself. A training run can move parameters, lower an objective, improve a benchmark, memorize more cases, or expose more traces without yet forming a certified classifier. In the BEDC reading, strong learning is classifier formation under training pressure: a stable distinction is extracted from a learning trajectory, lifted to a NameCert surface, and ledgered against the residues that training can hide.

Definition 9462.352 (Learning system)**.** *A learning system is a tuple*

$$\mathcal{M} = (X, Y, \Theta, M, \mathcal{D}, \mathcal{L}),$$

where X is an input space, Y is an output space, Θ is a parameter space, $M : \Theta \to (X \to Y)$ is a model family, $\mathcal{D}$ is a data family or sampling regime, and $\mathcal{L}$ is a loss, reward, constraint, or other training pressure. For $\theta \in \Theta$, write $M_\theta = M(\theta)$.

Definition 9462.353 (Learning state)**.** *A learning state is*

$$\mathfrak{L} = (\theta, R, \mathcal{B}, \mathcal{C}, \Lambda),$$

where θ is the model state, R is the training record, $\mathcal{B}$ is the family of observed behaviors, $\mathcal{C}$ is the family of classifier candidates or certified classifiers, and Λ is the learning ledger recording training, generalization, compression, memorization, failure, shortcut, and distribution-boundary residue.

Definition 9462.354 (Learning transition)**.** *A learning transition is a passage*

$$\mathfrak{L}_t \rightsquigarrow \mathfrak{L}_{t+1}$$

between learning states. It may change the model state, extend the training record, alter observed behavior, expose classifier candidates, construct NameCert rows, repair ledgers, harden formal components, or add transfer evidence.

Definition 9462.355 (Learning classifier state)**.** *A learning classifier state is the certified classifier state*

$$\mathcal{T}_{\mathsf{learn}} = (S_{\mathsf{learn}}, P_{\mathsf{learn}}, L_{\mathsf{learn}}, C_{\mathsf{learn}}, N_{\mathsf{learn}}),$$

where S_{learn} records data, traces, activation sources, task scope, and training protocol; P_{learn} records the learned pattern; L_{learn} records the learning ledger; C_{learn} is the classifier induced by the learning process; and N_{learn} licenses the state.

Definition 9462.356 (Learning classifier shift)**.** *Two learning classifier states are equivalent when their classifier relations agree on the common source scope. A learning transition has learning classifier shift when the before and after learning classifier states are not equivalent.*

Definition 9462.357 (Certified learning discovery)**.** *A learning transition is a certified learning discovery when it has learning classifier shift, the target classifier has a NameCert, the learning ledger records the relevant scope and boundary residues, and the classifier supports observable behavior or certified generalization.*

Definition 9462.358 (Positive certified learning)**.** *A certified learning discovery is positive when its learning net information is positive:*

$$\mathsf{Info}_{\mathsf{learn}} = \mathrm{Benefit}_{\mathsf{learn}} - \mathrm{Cost}_{\mathsf{learn}} - \mathrm{Debt}_{\mathsf{learn}} > 0.$$

The benefit term may include coverage, prediction, generalization, transfer, compression, intervention, and unification gains. The cost term includes training, data, model, NameCert, ledger, and verification costs. The debt term includes unrecorded learning residue.

9462.14.1 Separation from Metrics

Theorem 9462.359 (Parameter motion is not learning discovery)**.** *A change in parameter state does not by itself imply certified learning discovery.*

Proof. Parameter motion reports an internal numerical change. It does not display a changed classifier relation, a source specification, a pattern row, a stability witness, a ledger policy, or a NameCert. Equivalent reparameterization, margin movement, and calibration can move parameters while preserving the classifier surface. □

Theorem 9462.360 (Model size is not certified information)**.** *Increasing parameter count or representational capacity does not by itself produce certified learning information.*

Proof. Additional capacity can store redundant representations, memorized cases, or alternative implementations of the same classifier. Certified learning information requires a classifier surface with NameCert and ledger rows. □

Theorem 9462.361 (Loss reduction is not learning discovery)**.** *Lower loss does not by itself imply certified learning discovery.*

Proof. Loss reduction can come from calibration, increased margin, sample memorization, shortcut use, or optimization inside an already certified classifier. It supplies an optimization signal, not the source, classifier, stability, and ledger rows required for learning discovery. □

Theorem 9462.362 (Learning discovery can precede loss reduction)**.** *A classifier can form before it lowers the displayed training loss.*

Proof. A classifier may support future transfer, organize a latent distinction, or apply to a target domain not emphasized by the current objective. Its certificate value is therefore not determined by immediate loss decrease. □

Theorem 9462.363 (Performance gain is not a certified explanation)**.** *External performance or benchmark improvement does not by itself supply a certified explanation of what was learned.*

Proof. A performance row reports behavior under a score. It does not identify which classifier supports the behavior, whether that classifier differs from a prior state, whether it is stable under transfer, or whether its failure boundary is ledgered. □

Corollary 9462.364 (Training success is not structural learning)**.** *A training run can succeed under its metric while failing to produce a certified learning discovery.*

Proof. Metric success may result from parameter optimization, calibration, memorization, or shortcut use under the same classifier surface. By the preceding theorems, none of these alone supplies certified classifier formation. □

9462.14.2 Memory, Generalization, and NameCert

Definition 9462.365 (Memorization event)**.** *A memorization event occurs when the model preserves reproducible outputs for training cases or training patterns without a transferable classifier witness. A rote learning transition is one whose main visible effect is memorization or record growth rather than certified classifier formation.*

Definition 9462.366 (Classifier-mediated generalization)**.** *Classifier-mediated generalization occurs when behavior on a target domain is supported by the same classifier surface formed on a source domain, together with a stability or transfer witness and a target residue ledger.*

Definition 9462.367 (Learning NameCert)**.** *A Learning NameCert for a learned classifier is*

$$\mathsf{NameCert}_{\mathsf{learn}} = (S_{\mathsf{learn}}, P_{\mathsf{learn}}, C_{\mathsf{learn}}, \Sigma_{\mathsf{learn}}, L_{\mathsf{learn}}).$$

The rows record training sources and extraction protocol, stable learned patterns, the induced classifier, stability or transfer evidence, and the learning ledger.

Theorem 9462.368 (Memorization is not structural learning)**.** *Memorization without a certified classifier is not structural learning.*

Proof. Structural learning requires a classifier surface that supports compression, prediction, transfer, or unification beyond the memorized records. A memorization event alone supplies stored cases, not a NameCert-backed classifier. □

Theorem 9462.369 (Certified generalization is structural learning)**.** *Classifier-mediated generalization with NameCert and ledger is structural learning.*

Proof. The transfer witness says that the source-side classifier remains the governing relation on the covered target side. The NameCert and ledger make the classifier and its target residue auditable. This is certified classifier formation with transfer scope. □

Theorem 9462.370 (Learning NameCert creates a certified learning object)**.** *A learned classifier equipped with a Learning NameCert is a certified learning object.*

Proof. The Learning NameCert supplies source, pattern, classifier, stability, and ledger rows. These are exactly the rows needed to treat the learned classifier as an auditable object rather than a parameter vector, loss value, activation dump, feature label, or benchmark row. □

Theorem 9462.371 (Learning claims require learning ledger)**.** *A claim that a model learned a structure is not closed without a learning ledger.*

Proof. Learning claims can hide memorization, shortcut use, spurious correlation, data leakage, benchmark overfitting, unsupported transfer, out-of-domain failure, threshold debt, feature entanglement, semantic debt, and verification debt. These residues must be recorded before the claim is closed. □

Theorem 9462.372 (Shortcut learning is not certified discovery without ledger)**.** *Shortcut-based performance gain is not certified learning discovery when shortcut debt is unrecorded.*

Proof. A shortcut may raise performance while depending on a spurious correlation. Without ledger rows for source, pattern, stability, and failure boundary, the apparent classifier lacks a complete Learning NameCert. □

9462.14.3 Representation and Transfer

Definition 9462.373 (Representation classifier)**.** *Given an activation map $a_\ell^\theta : X \to V_\ell$ and a tensor distinction $\phi : V_\ell \to [0,1]$, a representation classifier at threshold τ is the relation*

$$x \sim_{\ell,\phi,\tau} y \quad \Longleftrightarrow \quad m_{\phi,\tau}(a_\ell^\theta(x)) = m_{\phi,\tau}(a_\ell^\theta(y)).$$

Definition 9462.374 (Certified representation learning)**.** *Representation learning is certified when a representation classifier is equipped with a Tensor NameCert and a ledger for representation compression debt.*

Definition 9462.375 (Learning-to-generalization bridge)**.** *A learning-to-generalization bridge records that a classifier formed on a source domain is the classifier used on a covered target domain. Its certificate includes source and target domains, transfer or stability witness, target residue, failure cases, and not-claimed target boundary.*

Theorem 9462.376 (Representation learning requires Tensor NameCert)**.** *Features, clusters, embeddings, or low-dimensional views do not by themselves certify representation learning.*

Proof. They are candidate evidence for a representation classifier. Certified representation learning requires the source, pattern, classifier, stability, and ledger rows supplied by a Tensor NameCert. □

Theorem 9462.377 (Representation compression requires ledger)**.** *Representation compression requires a ledger for collapsed distinctions.*

Proof. Dimensionality reduction, bottlenecks, sparse-autoencoder codes, probes, clusters, and latent compression can collapse source distinctions. The collapsed distinctions are hidden residue and must be recorded when the representation is used as an explanatory certificate. □

Theorem 9462.378 (Certified generalizing learner requires transfer ledger)**.** *A model is a certified generalizing learner only when its classifier transfer is ledgered.*

Proof. The transfer ledger records target residue, out-of-domain boundary, failed transfer, stability limits, and not-claimed target scope. A test-set score can be evidence, but it is not this ledger. □

9462.14.4 Effect Certification

Definition 9462.379 (Machine-learning effect certificate)**.** *A machine-learning effect certificate is a derived packet*

$$\mathsf{MLEffectCert}(\hat{\theta}, D, \Pi, \ell, \epsilon, \mathcal{C})$$

whose rows record the source data packet, split provenance, training trace, prediction rows, loss-classifier rows, fit rows, validation or test rows, stability domain $\mathcal{C}$, NoLeak audit, failure surface, reproducibility row, and Learning NameCert for the predictor $\hat{\theta}$.

Definition 9462.380 (Certified machine-learning effectiveness)**.** *A predictor $\hat{\theta}$ is machine-learning-effective relative to $(D, \Pi, \ell, \epsilon, \mathcal{C})$ when $\mathsf{MLEffectCert}(\hat{\theta}, D, \Pi, \ell, \epsilon, \mathcal{C})$ is accepted and the displayed evaluation row satisfies*

$$\mathrm{Err}_\ell(\hat{\theta}; D_*) \preceq_\ell \epsilon$$

for the declared evaluation split D_*.

Theorem 9462.381 (Training fit does not imply learning effectiveness)**.** *A low training loss row does not by itself establish certified machine-learning effectiveness.*

Proof. Training fit compares the predictor only with the records consumed by the training route. The effect certificate also requires split provenance, prediction rows on the declared evaluation split, loss classifier rows, stability domain, NoLeak audit, failure surface, reproducibility, and a Learning NameCert. A training row omits these fields and therefore cannot license the effectiveness claim. □

Theorem 9462.382 (Effectiveness is scope-relative)**.** *An accepted machine-learning effect certificate licenses effectiveness only relative to its displayed data packet, protocol, loss classifier, error budget, and stability domain.*

Proof. Each row of the effect certificate is indexed by the declared source, split, protocol, loss, budget, and continuation scope. The certificate therefore proves a bounded predictor claim over those rows. It does not export an unrestricted claim over unlisted domains, losses, budgets, tasks, or continuation classes. □

Theorem 9462.383 (Leakage breaks effect certification)**.** *If evaluation information enters the training or selection route without being recorded as an admitted source row, the machine-learning effect certificate is not accepted.*

Proof. The split and NoLeak rows distinguish prediction on held-out records from replay of information already present in the training route. If evaluation information enters the training or selection route, the declared evaluation row no longer witnesses prediction on separated records. The split provenance and NoLeak audit therefore fail, so the effect certificate is rejected. □

Definition 9462.384 (Machine-learning effect strength)**.** *Machine-learning effect strength is ordered by the strongest certificate row displayed for the predictor:*

$$\mathsf{TrainFit} < \mathsf{ValFit} < \mathsf{TestFit} < \mathsf{ReproducedFit} < \mathsf{StabilityFit} < \mathsf{BridgeFit} < \mathsf{TheoremPreservingBridge}.$$

The rows respectively record training fit, validation fit, test fit with split provenance, reproduced fit, certified continuation stability, records-to-readout bridge fit, and bridge fit with preservation rows.

Theorem 9462.385 (Effect strength is certificate strength)**.** *An effectiveness claim may be reported only at the strongest effect row whose certificate fields are present.*

Proof. Each effect row adds fields not supplied by the weaker rows. Validation requires a split row beyond training fit. Test fit requires test-split provenance and NoLeak audit beyond validation fit. Reproduced fit requires a reproducibility row. Stability fit requires a certified continuation class and failure surface. Bridge fit requires an adapter or bridge readout row. Theorem-preserving bridge fit requires preservation rows. If those additional fields are absent, the stronger claim is not licensed by the displayed certificate. □

Corollary 9462.386 (Test fit requires split provenance)**.** *A small error row on a named test split does not have test-effect meaning unless the split provenance and NoLeak rows are present.*

Proof. By Definition 9462.384, TestFit includes test-split provenance. By Definition 9462.383, an evaluation row whose separation is not certified cannot pass the effect certificate. The remaining statement is only a finite record comparison, not a certified test-effect row. □

9462.14.5 Lineage, Verification, and Anti-Laundering

Definition 9462.387 (Learning lineage)**.** *A learning lineage is a finite chain of learning states. It is an optimization lineage when the visible improvements are parameter, record, loss, or performance improvements without certified classifier shift. It is a structural learning lineage when at least one edge has certified classifier shift. It is positive when at least one certified learning discovery has positive learning net information.*

Theorem 9462.388 (Optimization lineage need not be structural learning)**.** *A learning lineage can improve loss or performance without being a structural learning lineage.*

Proof. Old classifiers can become better calibrated, higher-margin, more memorized, or more shortcut-adapted without a certified classifier shift. The lineage improves metrics but not the certified classifier surface. □

Theorem 9462.389 (Structural learning need not be immediately positive)**.** *A structural learning transition can have nonpositive net learning information when the classifier first appears.*

Proof. The classifier may require costly source, stability, transfer, ledger, verification, and seal work before its benefits exceed its costs and remaining debts. □

Theorem 9462.390 (Verified learning claim is not necessarily learning discovery)**.** *Formal verification of a learning component is not learning discovery unless it changes and certifies the learning classifier surface.*

Proof. Verification can harden a property of data, optimization, stability, or certification while preserving the classifier relation. Then it is a hardening improvement rather than classifier-forming learning. □

Theorem 9462.391 (Formal hardening cost enters learning information)**.** *When formal hardening is counted as learning benefit, its cost and verification debt must also be counted.*

Proof. Hardening adds proof, dependency, backend, maintenance, and trust-boundary obligations. Omitting these terms overstates learning net information. □

Definition 9462.392 (Learning laundering)**.** *Loss laundering treats lower loss as learning discovery. Benchmark laundering treats score improvement as learning discovery. Parameter laundering treats model size or architecture complexity as learning discovery. Memorization laundering treats stored training cases as generalization. Shortcut laundering treats shortcut-based performance as certified classifier learning.*

Theorem 9462.393 (Learning laundering invalidates discovery claims)**.** *Loss, benchmark, parameter, memorization, and shortcut laundering invalidate certified learning discovery claims.*

Proof. Certified learning discovery requires classifier shift, NameCert, and ledger. Lower loss, higher benchmark score, parameter growth, memorized cases, and shortcut performance do not supply those rows. Shortcut and memorization claims are especially debt-bearing unless their residues are ledgered. □

Theorem 9462.394 (Classifier-forming learning theorem)**.** *The information-theoretic content of machine learning is certified classifier formation rather than parameter motion, loss reduction, benchmark improvement, record growth, or memorization.*

Proof. Definition 9462.359, Definition 9462.360, Definition 9462.361, and Definition 9462.363 separate learning discovery from common metrics. Definition 9462.368 separates structural learning from rote storage. Definition 9462.370 gives the certificate route from learned classifier to learning object. Definition 9462.371 supplies the ledger condition, and Definition 9462.393 rules out metric-based substitutes. □

Concretized in §6097: The finite packet realizes classifier-forming learning as certified classifier shift with ledgered training residue.

Concretized in §7546: Classifier-formation traces realize learning classifier shift as finite before/after classifier, ledger-residue, and NameCert rows.

9462.15 Certified Model Quality

section 9462.18 decomposes a machine-learning black box into missing certificate rows, and section 9462.14 reads learning as certified classifier formation. Model quality is the evaluation-level form of the same discipline. A model is not better merely because a displayed score rises. It is better on a declared scope when its critical behaviors are supported by certified classifiers, their stability and transfer rows improve, and their remaining black-box, ledger, shortcut, safety, semantic, and verification debts are accounted for under a public cost protocol.

Definition 9462.395 (Model quality scope)**.** *A model quality scope is a tuple*

$$\Omega_{\mathsf{q}} = (D, M, \mathcal{A}, \mathcal{B}_{\mathsf{crit}}, \Pi),$$

where D is the evaluation domain, M is the model state, $\mathcal{A}$ is the admitted transformation, perturbation, intervention, or transfer family, $\mathcal{B}_{\mathsf{crit}}$ is the family of critical behavior targets, and Π is the public scoring and certificate protocol.

Definition 9462.396 (Certified behavior coverage)**.** *For a model quality scope Ω_{q}, the certified behavior coverage of M is the subfamily*

$$\mathrm{CertBeh}(M; \Omega_{\mathsf{q}}) = \{B \in \mathcal{B}_{\mathsf{crit}} \mid \exists C, N.\ C \Vdash B \text{ and } N = \mathsf{NameCert}(C)\},$$

with the NameCert read on the displayed scope. The uncovered behavior family is

$$\mathrm{UncertBeh}(M; \Omega_{\mathsf{q}}) = \mathcal{B}_{\mathsf{crit}} \setminus \mathrm{CertBeh}(M; \Omega_{\mathsf{q}}).$$

Definition 9462.397 (Model quality ledger)**.** *A model quality ledger records, for every critical behavior target, the present classifier support, NameCert status, stability status, transfer status, safety boundary, shortcut risk, semantic boundary, verification status, failure cases, and not-claimed scope. It also records the public weights used to compare benefit, cost, and debt.*

Definition 9462.398 (Certified model quality score)**.** *For a public weight protocol $\mathbf{w}$, the certified model quality score is the audited quantity*

$$Q_{\mathbf{w}}(M; \Omega_{\mathsf{q}}) = \mathrm{Benefit}_{\mathbf{w}}(M; \Omega_{\mathsf{q}}) - \mathrm{Cost}_{\mathbf{w}}(M; \Omega_{\mathsf{q}}) - \mathrm{Debt}_{\mathbf{w}}(M; \Omega_{\mathsf{q}}).$$

The benefit term may include performance, certified classifier gain, stability gain, generalization gain, safety gain, intervention gain, and verification gain. The cost term records model, data, training, certificate, ledger, intervention, and verification cost. The debt term records black-box debt, ledger debt, stability debt, generalization debt, safety debt, shortcut debt, semantic debt, and verification debt.

Definition 9462.399 (Model quality improvement)**.** *A transition $M_0 \rightsquigarrow M_1$ is a certified model quality improvement on Ω_{q} when*

$$Q_{\mathbf{w}}(M_1; \Omega_{\mathsf{q}}) > Q_{\mathbf{w}}(M_0; \Omega_{\mathsf{q}})$$

and the comparison ledger records which benefit, cost, and debt rows changed.

Definition 9462.400 (Model quality gate)**.** *A model quality gate is a decision rule that accepts a model state on Ω_{q} only when the critical behavior coverage, maximum critical debt, safety ledger, transfer ledger, and verification ledger meet the declared thresholds of Π.*

9462.15.1 Separation from Performance

Theorem 9462.401 (Performance gain does not imply quality gain)**.** *A performance improvement on a displayed metric does not by itself imply certified model quality improvement.*

Proof. The certified model quality score includes benefit, cost, and debt. A metric improvement can be offset by certificate cost, stability loss, generalization debt, safety debt, shortcut debt, semantic debt, or verification debt. It may also leave the supporting classifier uncertified. Therefore the performance row alone does not determine the quality comparison. □

Concretized in §6625: Certified model quality ledgers are realized as finite behavior-coverage, weight-profile, cost, debt, stability, and verification NameCert rows.

Concretized in §6117: The CertifiedModelQualityUp chapter realizes the model-quality ledger as finite certificate rows.

Theorem 9462.402 (Debt reduction can improve quality without metric gain)**.** *For a fixed performance row, lowering critical debt can yield certified model quality improvement.*

Proof. If the performance contribution is unchanged and the added cost of the transition is smaller than the decrease in weighted critical debt, then the value of $Q_{\mathbf{w}}$ strictly increases by Definition 9462.398. The improvement is an audit improvement rather than a raw metric improvement. □

Theorem 9462.403 (Certified classifier gain is a quality row)**.** *Adding a NameCert-backed classifier for a critical behavior is a model quality benefit when its certificate cost and remaining debt are accounted for.*

Proof. The certified behavior coverage expands, and the behavior moves from uncovered or partial-certificate status toward certified support. The quality score counts this as certified classifier gain, while the certificate and ledger work enter the cost and debt terms. Thus the benefit is legitimate only under the public accounting protocol. □

Theorem 9462.404 (Unledgered classifier gain is not closed quality gain)**.** *A classifier that appears to support a critical behavior does not give closed model quality gain when its ledger is absent or critically incomplete.*

Proof. Critical behavior support requires not only a classifier candidate but also source, pattern, stability, and ledger rows. If the ledger is absent or critically incomplete, the behavior retains black-box residue by Definition 9462.465. The apparent gain therefore remains debt-bearing and cannot be counted as closed quality gain. □

9462.15.2 Debt-Guided Training

Definition 9462.405 (Debt-guided training signal)**.** *A debt-guided training signal is a training pressure derived from the model quality ledger. It may target boundary cases, unstable margins, shortcut residues, unsafe behavior, failed transfer, unclassified critical behaviors, semantic ambiguity, or verification gaps.*

Theorem 9462.406 (Ledger rows can define training targets)**.** *Critical residue recorded in a model quality ledger can be used as a certificate-guided training target.*

Proof. Each recorded residue row identifies a specific failure of source, pattern, classifier, stability, transfer, safety, semantic, or verification closure. A training objective can be directed at reducing that row, for example by adding boundary cases, strengthening margin stability, removing shortcut dependence, or improving transfer evidence. The ledger therefore supplies a structured target rather than an undifferentiated error signal. □

Theorem 9462.407 (Debt-guided training needs post-training recertification)**.** *Reducing a ledger-derived training loss does not by itself establish certified model quality improvement.*

Proof. The new model state may alter classifier boundaries, create new residue, shift safety behavior, or change the transfer scope. Certified quality compares the full audited score and ledger after the transition. Thus debt-guided training must be followed by renewed classifier, stability, ledger, and gate checks. □

Theorem 9462.408 (Model quality gate blocks laundering)**.** *A model quality gate rejects score, benchmark, feature, and verification laundering when the required certificate and ledger rows fail the declared thresholds.*

Proof. The gate reads acceptance from certified behavior coverage, critical debt, safety ledger, transfer ledger, and verification ledger rather than from a single performance row or proof artifact. A high score, a named feature, or a verified subcomponent therefore cannot pass the gate unless the corresponding certificate and ledger obligations are also met. □

Theorem 9462.409 (Certified model quality theorem)**.** *On a declared scope, model quality is certified classifier quality with public accounting for benefit, cost, and debt.*

Proof. Definition 9462.398 defines quality by benefit minus cost and debt. The benefit rows include certified classifier formation, stability, transfer, safety, intervention, and verification gains. The debt rows record the certificate and ledger residues that prevent closed explanation. The separation theorems above exclude performance-only and unledgered readings. Hence the BEDC quality object is the public accounting of certified classifier support for the critical behavior family. □

9462.16 World-Model Classifier Certification

section 9462.4 treats tensor features as classifier candidates, and section 9462.14 separates structural learning from loss reduction. World-model representation learning supplies a sharper test case. A latent representation does not become an explanation merely by aligning paired views, recovering a coordinate system, or improving planning. It becomes a BEDC object only when the recovered relation is displayed as a scoped classifier with source rows, stability rows, transfer rows, and a ledger for the assumptions that make the recovery meaningful.

Definition 9462.410 (World-model certification scope)**.** *A world-model certification scope is a tuple*

$$\Omega_{\mathsf{wm}} = (Z, X, g, \mathcal{P}, \mathcal{T}, \mathcal{H}, \mathcal{A}),$$

where Z is the latent carrier, X is the observation carrier, $g : Z \to X$ is the observation map, $\mathcal{P}$ is the positive-pair protocol, $\mathcal{T}$ is the transition family, $\mathcal{H}$ is the admitted encoder family, and $\mathcal{A}$ is the assumption ledger. The ledger records independence, stationarity, dimension matching, noise, sampling, population or finite-sample status, and not-claimed target domains.

Definition 9462.411 (World-latent classifier candidate)**.** *For an encoder $f : X \to V$, write $h = f \circ g : Z \to V$. A world-latent classifier candidate is a relation*

$$z \sim_h z'$$

whose public comparison is induced by h together with a declared readout rule. The readout may compare recovered coordinates, invariant subspaces, thresholded latent features, transition classes, or planning states, provided the scope records how those comparisons are observed.

Definition 9462.412 (Identifiability witness)**.** *An identifiability witness for h on Ω_{wm} is a displayed map W and an equivalence mode $\equiv_W$ such that the recovered representation is compared to the latent carrier by*

$$h(z) \equiv_W W(z)$$

on the stated scope. The witness may be exact, approximate, empirical, formal, or hybrid. Its mode, bound, and residual cases are certificate data.

Definition 9462.413 (World-latent NameCert)**.** *A World-latent NameCert for $(\Omega_{\mathsf{wm}}, h, \sim_h)$ is the packet*

$$\mathsf{NameCert}_{\mathsf{wm}} = (S_{\mathsf{wm}}, P_{\mathsf{wm}}, C_{\mathsf{wm}}, \Sigma_{\mathsf{wm}}, L_{\mathsf{wm}}).$$

The source row S_{wm} records the scope, encoder, observation protocol, pair protocol, transition family, and assumption ledger. The pattern row P_{wm} records alignment, recovery, separation, transition, and planning evidence. The classifier row C_{wm} is the relation $\sim_h$. The stability row Σ_{wm} records invariance under admitted transformations, approximation bounds, and transfer witnesses. The ledger row L_{wm} records unidentified coordinates, rotational or gauge freedom, finite-sample residue, model-class restriction, shortcut risk, planning failures, and not-claimed domains.

Definition 9462.414 (Certified world-model classifier)**.** *A certified world-model classifier is a tuple*

$$(\Omega_{\mathsf{wm}}, h, \sim_h, \mathsf{NameCert}_{\mathsf{wm}})$$

where the world-latent classifier candidate carries the World-latent NameCert on the displayed scope.

Theorem 9462.415 (Latent recovery is certificate data)**.** *An identifiability claim for a learned world representation is not a closed BEDC explanation unless its witness, equivalence mode, scope, and residue are recorded in a World-latent NameCert.*

Proof. An identifiability statement compares a learned representation with a latent carrier only relative to a scope and an equivalence mode. The comparison may hide dimension assumptions, sampling assumptions, coordinate freedom, approximation error, and uncovered domains. These are precisely source, stability, and ledger rows in Definition 9462.413. Without those rows, the claim gives evidence for a classifier candidate but not a certified world-model classifier. □

Concretized in §6124: World-model classifier packets realize scoped latent recovery, identifiability, transfer, and residue ledgers as finite NameCert data.

Theorem 9462.416 (World-model objectives do not certify classifiers)**.** *Optimizing a world-model objective does not by itself certify the world-latent classifier induced by the learned representation.*

Proof. An objective supplies a training pressure and possibly a record of alignment or prediction. It does not by itself display the classifier relation, the identifiability witness, the stability mode, the transfer scope, or the residue ledger. By Definition 9462.414, those rows are required for certification. □

Theorem 9462.417 (Assumptions are source rows, not hidden laws)**.** *Distributional and transition assumptions used in world-model recovery belong to the source and ledger rows of the certificate.*

Proof. The assumptions determine which latent systems, pair protocols, transition families, and recovery comparisons are covered. If they are not displayed, the classifier's domain of validity is ambiguous. If they are displayed but their failures are not ledgered, the certificate hides boundary debt. Thus these assumptions are certificate rows rather than ambient laws. □

Theorem 9462.418 (Planning transfer requires a transfer ledger)**.** *A claim that a recovered world representation supports planning is not closed without a transfer ledger from latent recovery to planning behavior.*

Proof. Latent recovery and planning success are different behavior targets. Planning adds policy extraction, rollout stability, objective alignment, intervention boundary, and failure-mode residue. A recovery certificate can support the planning claim only when these transfer rows are recorded in Σ_{wm} and L_{wm}. □

Theorem 9462.419 (World-model certification is scoped black-box resolution)**.** *If a learned representation supports a world-latent classifier and carries a World-latent NameCert on Ω_{wm}, then the corresponding world-model behavior is no longer black-box behavior on that scope.*

Proof. Black-box behavior is the absence of a certificate for the classifier supporting the behavior. The World-latent NameCert supplies the source, pattern, classifier, stability, and ledger rows for the supporting classifier on Ω_{wm}. The resolution is scoped because the certificate explicitly records its assumptions and not-claimed domains. □

Theorem 9462.420 (World-model quality is certified classifier quality)**.** *Within a declared world-model scope, representation quality relevant to BEDC explanation is certified classifier quality rather than raw performance, latent visual coherence, or objective value.*

Proof. Raw performance, visual coherence, and objective value can be evidence for the pattern row, but they do not determine a certified classifier. The BEDC object is formed only when the recovered relation carries the World-latent NameCert of Definition 9462.413. Therefore the explanatory quality of the world model is measured by the certified classifier surface and its remaining ledger debt. □

9462.17 Formal Verification in Certified Interpretability

section 9462.5 treats formal verification as a hard-certificate layer, and section 9462.6 separates ledger-only, partially hardened, and fully hardened classifiers. The present section applies that separation to certified interpretability. Formal verification is not the rule that forms an explanation object; the object is formed by classifier, NameCert, and ledger. Verification hardens the components that have been faithfully transcribed into formal statements, and it must carry its own ledger.

Definition 9462.421 (Certified interpretability object)**.** *A certified interpretability object is*

$$E = (B, C, N_C, \Lambda_C),$$

where B is the explained behavior, C is the classifier supporting B, $N_C = (S_C, P_C, C, \Sigma_C, L_C)$ is the NameCert or Tensor NameCert for C, and Λ_C is the explanation ledger.

Definition 9462.422 (Explanation component)**.** *An explanation component of E is any row or role used by its certified surface: source, pattern, classifier, stability, ledger, feature, circuit, intervention, semantic assignment, seal, or verification component. The family of such components is denoted* $\mathrm{Comp}(E)$.

Definition 9462.423 (Verifiable component)**.** *A component $F \in \mathrm{Comp}(E)$ is verifiable relative to a backend $\mathcal{V}$ when it can be transcribed as a formal statement φ_F with an explicit proof obligation and declared scope. It is ledger-only when it is not presently transcribable without losing essential empirical, semantic, distributional, or boundary content, but its evidence and limitations are recorded in the ledger.*

Definition 9462.424 (Formal verification target)**.** *A formal verification target is a tuple*

$$T = (F, \varphi_F, \mathcal{V}),$$

where F is an explanation component, φ_F is its transcribed formal statement, and $\mathcal{V}$ is the verification backend. A proof object π_F hardens the target when

$$\mathcal{V} \vdash \pi_F : \varphi_F.$$

Definition 9462.425 (Verification coverage)**.** *The verification coverage of a hardened component F is the part of F actually expressed by φ_F. If the coverage is a proper subcomponent of F, the remainder must be recorded in a transcription ledger or the hardening claim is incomplete.*

9462.17.1 Hardenable Components

Definition 9462.426 (Threshold-classifier targets)**.** *For a threshold classifier*

$$m_{\phi,\tau}(x) = \begin{cases} 0, & \phi(a(x)) < \tau, \\ 1, & \phi(a(x)) \geq \tau, \end{cases}$$

verifiable targets include threshold-definition consistency, equivalence-relation properties of the induced classifier, finite-scope classification tables, margin stability, and threshold-perturbation bounds.

Definition 9462.427 (Compression, finite-data, circuit, and ledger targets)**.** *Compression and projection targets include injectivity on a scope, non-injectivity witnesses, collapsed-distinction enumeration, and compression-cell equivalence. Finite-data targets include enumeration coverage, finite classifier-table correctness, finite confusion-matrix properties, and finite no-counterexample claims. Circuit targets include graph reachability, acyclicity, dependency containment, path coverage, and declared intervention-site membership. Ledger targets include finite or formal coverage claims of the form*

$$\mathrm{Req}(E) \subseteq \mathrm{Rec}(\Lambda).$$

Definition 9462.428 (NameCert structure targets)**.** *NameCert structure targets include existence of the source, pattern, classifier, stability, and ledger rows; consistency of references between rows; agreement between source scope and classifier domain; and ledger coverage for the declared scope.*

Theorem 9462.429 (Finite classifier tables are verifiable)**.** *If a classifier is displayed by a finite table on a finite domain, then its basic classifier properties can be formally verified.*

Proof. On a finite domain $D_0 = \{x_1, \ldots, x_n\}$ the classifier relation can be represented by a finite Boolean matrix M_{ij} with $M_{ij} = 1$ exactly when $C(x_i, x_j)$. Reflexivity, symmetry, and transitivity reduce to finitely many Boolean conditions on this matrix. Those conditions are formal verification targets. □

Theorem 9462.430 (Margin stability is verifiable under a bound)**.** *If*

$$|\phi(a(x)) - \tau| > \epsilon$$

and every admitted perturbation α satisfies

$$|\phi(a(\alpha(x))) - \phi(a(x))| \leq \epsilon,$$

then the threshold mark of x is preserved by α.

Proof. If $\phi(a(x)) \geq \tau$, the margin assumption gives $\phi(a(x)) - \tau > \epsilon$, and the perturbation bound gives $\phi(a(\alpha(x))) > \tau$. If $\phi(a(x)) < \tau$, the margin assumption gives $\tau - \phi(a(x)) > \epsilon$, and the perturbation bound gives $\phi(a(\alpha(x))) < \tau$. In both cases the threshold mark is unchanged. □

Theorem 9462.431 (Finite ledger coverage is verifiable)**.** *If required residue and recorded residue are finite displayed families, then the coverage claim*

$$\mathrm{Req}(E) \subseteq \mathrm{Rec}(\Lambda)$$

is formally verifiable.

Proof. The finite inclusion is equivalent to checking, for each displayed required row, membership in the recorded family. This is a finite family of decidable membership checks. □

9462.17.2 Limits and Transcription

Definition 9462.432 (Open semantic and distributional components)**.** *An open semantic component depends on natural-language meaning, cultural context, open-ended categories, or semantic similarity. An open distributional component depends on an unknown or open distribution. Such components may be ledgered, sampled, or partially modeled, but they are not fully hardened unless their formalization and residual gap are declared.*

Definition 9462.433 (Transcription ledger)**.** *A transcription ledger for $F \rightsquigarrow \varphi_F$ records which parts of F are expressed by φ_F, which parts are omitted, the scope and assumptions of the transcription, any semantic weakening, approximation, excluded cases, and the not-claimed boundary.*

Definition 9462.434 (Transcription gap)**.** *A transcription gap exists when critical content of F is neither expressed by φ_F nor recorded in the transcription ledger.*

Theorem 9462.435 (Open semantics are not closed by proof alone)**.** *An open semantic component is not closed by formal verification alone unless a formal semantics and semantic ledger are supplied.*

Proof. Formal verification checks a formal statement. Natural-language meaning has ambiguity, context dependence, polysemy, and boundary cases. A formal semantics specifies what has been modeled, and the semantic ledger records the gap between that model and the open concept. □

Theorem 9462.436 (Finite empirical stability is not global formal stability)**.** *Passing finite perturbation tests does not by itself prove stability on an open domain or open transformation family.*

Proof. Finite tests cover only the displayed sample and displayed transformations. A larger domain or transformation family may contain cases not tested. Global formal stability requires a proof over the declared domain and family. □

Theorem 9462.437 (Untranscribed claims have no proof target)**.** *If an explanation component has not been transcribed into a formal statement, then formal verification has no target for that component.*

Proof. A backend judgment has the form $\mathcal{V} \vdash \pi : \varphi$. Without a statement φ_F representing the component, no such judgment can be posed for it. □

Theorem 9462.438 (Verified statement needs transcription ledger)**.** *A verified statement does not audit-completely support an explanation component unless its transcription ledger is supplied.*

Proof. The proof establishes φ_F. Without the transcription ledger, the auditor cannot tell whether φ_F covers the critical content of F, weakens the claim, omits a boundary, or changes semantic scope. □

Theorem 9462.439 (Transcription gaps block full hardening)**.** *A component with a transcription gap cannot be claimed fully hardened.*

Proof. Full hardening requires every critical part of the component to be covered by the verified statement or ledgered as residue. A transcription gap is precisely a critical part covered by neither. □

9462.17.3 Verification Ledgers and Grades

Definition 9462.440 (Component verification ledger)**.** *A component verification ledger records the backend, formal statement, proof object, assumptions, axioms, dependencies, trusted libraries, kernel or checker trust boundary, extraction assumptions, unverified components, and portability boundary for a verified component.*

Definition 9462.441 (Verification opacity)**.** *Verification opacity occurs when a verification claim lacks its backend, checked statement, proof artifact, assumptions, dependencies, or trust boundary.*

Definition 9462.442 (Verification grade)**.** *An explanation is ledger-only when it has NameCert and ledger but no formally hardened component; partially verified when at least one component is hardened; fully verified scoped when every hardenable critical component in the declared scope is hardened and every non-hardenable residue is ledgered; and globally verified only when the global scope and all excluded cases are ledgered.*

Theorem 9462.443 (Verification opacity blocks audit completeness)**.** *An opaque verification claim is not audit-complete.*

Proof. The auditor must know which backend accepted which statement, with which proof object, assumptions, dependencies, and trust boundary. If these rows are absent, the strength and scope of the verification cannot be audited. □

Theorem 9462.444 (Verification ledger does not replace explanation ledger)**.** *A component verification ledger does not replace the explanation ledger.*

Proof. The verification ledger audits proof acceptance and trust boundary. The explanation ledger audits source residue, classifier boundary, stability limits, semantic ambiguity, intervention residue, generalization residue, and not-claimed behavior. These are different required rows. □

Theorem 9462.445 (Partial verification is not scoped full verification)**.** *Partially verified explanation does not imply fully verified scoped explanation.*

Proof. One hardened component may leave other hardenable critical components unhardened and unledgered. Fully verified scoped status requires the entire critical hardening frontier of the declared scope to be handled. □

Theorem 9462.446 (Global verification requires global scope ledger)**.** *A global verification claim requires a global scope ledger.*

Proof. Global verification quantifies over the whole claimed source, behavior, model, or domain space. The ledger must record that scope, the excluded cases, the assumptions, and every non-hardenable residue. Without that ledger, the global claim is unclosed. □

9462.17.4 NDNA Verification and Discovery

Definition 9462.447 (NDNA verification profile)**.** *An NDNA verification profile records, for each NDNA locus, whether it is unhardened, hardenable, hardened, or ledger-only, together with the hardening frontier, the component verification ledger, and the not-claimed boundary.*

Theorem 9462.448 (One hardened NDNA locus does not harden all NDNA)**.** *Hardening one NDNA locus does not imply that the whole NDNA packet is hardened.*

Proof. NDNA has source, feature, circuit, classifier, stability, intervention, ledger, and seal loci. A proof about one locus does not establish the formal statements, ledgers, or non-hardenable residues of the other loci. □

Theorem 9462.449 (Fully verified NDNA requires verification profile)**.** *A fully verified NDNA claim requires a complete NDNA verification profile.*

Proof. The claim requires all critical hardenable loci to be hardened and all non-hardenable critical residue to be ledgered. The verification profile is the audit surface that records those statuses locus by locus. □

Definition 9462.450 (Verification-assisted discovery)**.** *A positive discovery is verification-assisted when formal hardening contributes positive verification gain to its net explanation information. It is verification-only when the classifier surface is unchanged and the transition only hardens components.*

Theorem 9462.451 (Verification gain does not replace classifier novelty)**.** *Verification gain does not by itself create classifier-level discovery.*

Proof. Classifier-level discovery requires classifier novelty. A verification gain can harden a row, reduce verification debt, or improve reuse while leaving the classifier relation unchanged. □

Theorem 9462.452 (Verification-assisted discovery counts verification cost)**.** *When verification contributes to positive discovery, verification cost and verification debt must be counted.*

Proof. Formal hardening consumes transcription work, proof construction, dependency maintenance, checking, transport, and verification-ledger work. Omitting those terms overstates net explanation information. □

Definition 9462.453 (Verification misuse)**.** *Verification laundering uses a verified statement to hide a missing NameCert, ledger, classifier novelty, or scope boundary. Proof-quantity laundering treats many proof artifacts as explanation quality. Backend laundering states that a result is verified without naming backend, dependencies, and trust boundary. Scope laundering by verification uses a scoped proof as if it were global.*

Theorem 9462.454 (Verification misuse invalidates the stronger claim)**.** *Verification laundering, proof-quantity laundering, backend laundering, and scope laundering by verification invalidate the stronger explanation or discovery claim they are used to support.*

Proof. Verification does not replace NameCert or ledger, proof quantity does not determine explanation quality, opaque backend claims are not audit-complete, and scoped proofs do not prove larger scopes. Each misuse therefore asserts more than the verification evidence supports. □

Theorem 9462.455 (Formal verification in certified interpretability theorem)**.** *In certified interpretability, formal verification hardens transcribed components under a backend and verification ledger; it does not replace the NameCert, explanation ledger, NDNA profile, or classifier-novelty condition for discovery.*

Proof. Definition 9462.429, Definition 9462.430, and Definition 9462.431 identify typical hardenable targets. Definition 9462.435, Definition 9462.436, and Definition 9462.437 identify limits. Definition 9462.438 and Definition 9462.443 give the ledger requirements. Definition 9462.451 separates hardening from discovery. The theorem follows. □

Concretized in §4106: Type-checking classifier packets realize finite classifier-membership verification as Ext-membership, derivation-trace, subject-reduction, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7920: Verification coverage ledger packets realize certified-interpretability coverage as component-key, transcription, backend proof-boundary, coverage, residue, scoped-explanation, classifier-facing, hsame, Cont, Pkg, and NameCert rows.

9462.18 BEDC Decomposition of Machine-Learning Black Boxes

section 9462.7 identifies black-box behavior as supported structure without certificate closure. section 9462.8 isolates the ledger part of that debt, section 9462.9 gives the role-complete neural packet, and section 9462.17 separates certificate closure from formal hardening. The present section puts those readings into one decomposition theorem: a machine-learning black box is not a primitive opacity, but a certificate deficit for a classifier candidate supporting a displayed behavior.

Definition 9462.456 (Behavioral black-box object)**.** *A behavioral black-box object is a tuple*

$$\mathbb{B} = (B, D_B, E_B),$$

where B is a displayed model behavior, D_B is the behavior scope, and E_B is the evidence packet, consisting of samples, outputs, statistics, traces, failure cases, or an observation protocol. It is eligible for decomposition when B has observed stability but lacks a closed explanation.

Definition 9462.457 (Support candidates)**.** *For a behavior B, the support-candidate family is*

$$\mathrm{SuppCand}(B) = \{C \mid C \Vdash B\},$$

where $C \Vdash B$ means that classifier candidate C supports B by prediction, correlation, intervention, formal constraint, or hybrid evidence.

Definition 9462.458 (Decomposition target)**.** *A BEDC decomposition target is a pair (B, C) with $C \in \mathrm{SuppCand}(B)$. The target asks which certificate rows and ledger rows are already present for the claim that C supports B, and which rows remain absent or incomplete.*

Definition 9462.459 (Certificate fields)**.** *For a classifier candidate C, its certificate-field family is*

$$\mathrm{Field}(C) = \{S_C, P_C, C_C, \Sigma_C, L_C\},$$

where S_C is the source row, P_C the pattern row, C_C the classifier row, Σ_C the stability row, and L_C the ledger-policy row.

Definition 9462.460 (Present and missing fields)**.** *For a target (B, C), $\mathrm{Present}(B, C)$ is the subfamily of certificate fields that are declared, anchored, scoped, and supplied with the required ledger evidence. Its missing-field family is*

$$\mathrm{Missing}(B, C) = \mathrm{Field}(C) \setminus \mathrm{Present}(B, C).$$

Definition 9462.461 (BEDC black-box decomposition)**.** *The BEDC decomposition of B relative to C is*

$$\mathrm{Decomp}_{\mathrm{BEDC}}(B, C) = (\mathrm{Present}(B, C), \mathrm{Missing}(B, C), \Lambda_{B,C}, \mathrm{Debt}_{B,C}),$$

where $\Lambda_{B,C}$ is the ledger currently attached to the target and $\mathrm{Debt}_{B,C}$ is the cost of unrecorded or incomplete critical residue.

9462.18.1 Decomposition States

Definition 9462.462 (Decomposition states)**.** *A behavior is unexplained when $\mathrm{SuppCand}(B) = \emptyset$. It is in candidate state relative to C when $C \Vdash B$ but the certificate rows are absent or only fragmentary. It is in partial-certificate state when $\mathrm{Present}(B, C) \neq \emptyset$ and $\mathrm{Missing}(B, C) \neq \emptyset$. It is NameCert-complete when $\mathrm{Missing}(B, C) = \emptyset$, ledger-complete when the critical ledger gap is empty, and closed-explanation state when NameCert-completeness, ledger-completeness, scope seal, and not-claimed boundary are all present.*

Theorem 9462.463 (Unexplained and black-box states are distinct)**.** *If $\mathrm{SuppCand}(B) = \emptyset$, then B is not a certificate-deficit black box.*

Proof. A certificate-deficit black box requires a supporting classifier candidate whose certificate rows are absent or incomplete. When $\mathrm{SuppCand}(B)$ is empty, no such candidate has been displayed. The failure is therefore at the support-candidate level, not at the certificate-completion level. □

Concretized in §7533: Black-box fiber witness packets realize visible behavior, support-candidate, missing-certificate, hidden-fiber, and ledger rows as finite NameCert data.

Theorem 9462.464 (Partial certificate states retain black-box residue)**.** *If (B, C) is in partial-certificate state, then the explanation of B relative to C retains black-box residue.*

Proof. Partial-certificate state means that at least one certificate field is missing. A closed explanation requires source, pattern, classifier, stability, and ledger-policy rows for the declared scope. The absent row therefore remains a certificate deficit. □

Theorem 9462.465 (Ledger deficit blocks decomposition closure)**.** *If the ledger-policy row is absent or the critical ledger gap is nonempty, then (B, C) is not in closed-explanation state.*

Proof. Closed-explanation state includes ledger-completeness. By Definition 9462.183, ledger completeness is absence of required residue gap. An absent ledger-policy row or nonempty critical ledger gap violates that condition. □

Theorem 9462.466 (Closed explanation removes scoped black-box status)**.** *If (B, C) is in closed-explanation state, then B is not a black box relative to C on the sealed scope.*

Proof. Black-box status relative to C is the absence or incompleteness of the certificate and ledger rows supporting $C \Vdash B$. Closed-explanation state supplies those rows and seals their scope. Hence the certificate-deficit condition no longer holds on that scope. □

9462.18.2 Deficit Vectors and Frontiers

Definition 9462.467 (Deficit vector)**.** *The deficit vector of a target (B, C) is*

$$\mathbf{D}(B, C) = (d_S, d_P, d_C, d_\Sigma, d_L) \in \{0, 1\}^5,$$

where the coordinate is 1 exactly when the corresponding source, pattern, classifier, stability, or ledger field is absent or incomplete. The ledger coordinate is also 1 when the ledger-policy row is present but the critical ledger gap is nonempty.

Definition 9462.468 (Weighted deficit score)**.** *For nonnegative field weights*

$$\mathbf{w}_D = (w_S, w_P, w_C, w_\Sigma, w_L),$$

the weighted deficit score is

$$\mathrm{DefScore}(B, C) = w_S d_S + w_P d_P + w_C d_C + w_\Sigma d_\Sigma + w_L d_L.$$

Definition 9462.469 (Decomposition frontier)**.** *The decomposition frontier of (B, C) is*

$$\mathrm{Frontier}(B, C) = \mathrm{Missing}(B, C) \cup \mathrm{Gap}(\Lambda_{B,C}, (B, C)).$$

It is the family of certificate fields and ledger rows whose completion would next reduce black-box residue.

Theorem 9462.470 (Zero deficit is field-and-ledger closure)**.** *The equality* $\mathbf{D}(B, C) = (0, 0, 0, 0, 0)$ *holds exactly when no certificate field is missing and the critical ledger gap is empty.*

Proof. Each nonledger coordinate records absence or incompleteness of its corresponding certificate field. The ledger coordinate records either absence of the ledger-policy row or nonempty critical ledger gap. All coordinates vanish exactly when every field is present and no critical ledger row is missing. □

Theorem 9462.471 (Decomposition improvement reduces black-box deficit)**.** *If a transition from decomposition* D_0 *to* D_1 *satisfies*

$$\mathrm{DefScore}(D_1) < \mathrm{DefScore}(D_0),$$

then it reduces black-box deficit for the fixed target.

Proof. The score is the weighted cost of the displayed certificate deficits. Strict decrease of that score is precisely reduction of the weighted deficit. □

Theorem 9462.472 (Frontier gives the decomposition route)**.** *For a fixed target* (B, C)*, the decomposition frontier identifies the certificate rows and ledger rows that must be addressed to move toward closure.*

Proof. The frontier is the union of missing certificate fields and required ledger rows not recorded by the current ledger. These are exactly the rows preventing closure under Definition 9462.462. □

9462.18.3 NDNA and Formal Hardening

Definition 9462.473 (NDNA decomposition map)**.** *For a behavior* B *with neural theorem-DNA packet*

$$\mathsf{NDNA}(B) = (G_{\mathsf{source}}, G_{\mathsf{feature}}, G_{\mathsf{circuit}}, G_{\mathsf{classifier}}, G_{\mathsf{stability}}, G_{\mathsf{intervention}}, G_{\mathsf{ledger}}, G_{\mathsf{seal}}),$$

the NDNA decomposition map reads the source row from G_{source}*, pattern evidence from* G_{feature} *and* G_{circuit}*, the classifier row from* $G_{\mathsf{classifier}}$*, stability evidence from* $G_{\mathsf{stability}}$ *and* $G_{\mathsf{intervention}}$*, ledger rows from* G_{ledger}*, and the closure scope from* G_{seal}*.*

Definition 9462.474 (NDNA-complete decomposition)**.** *An NDNA-complete decomposition is a decomposition in which the NDNA packet is complete, anchored, NameCert-compatible, ledger-complete, and sealed for the declared scope.*

Theorem 9462.475 (NDNA-complete decomposition yields BEDC closure)**.** *An NDNA-complete decomposition yields closed-explanation state for some supporting classifier candidate.*

Proof. By completeness and anchoring, every NDNA locus has displayed content and evidence route. By NameCert compatibility, the projected source, pattern, classifier, stability, and ledger rows form a certificate for the classifier supporting B. Ledger-completeness and the seal provide the remaining closure rows. Thus the corresponding target is in closed-explanation state. □

Definition 9462.476 (Verification-aware decomposition)**.** *A decomposition is verification-aware when it records which components are formally hardened, which hardenable components remain unhardened, which components are ledger-only, and which transcription ledger, verification ledger, and trust boundary govern every verification claim.*

Theorem 9462.477 (Closure and hardening are separate axes)**.** *Verification-aware decomposition separates certificate closure from formal hardening.*

Proof. Certificate closure is determined by NameCert rows, ledger rows, scope seal, and not-claimed boundary. Formal hardening is determined by transcribed statements, proof objects, verification ledgers, and backend trust boundaries. A verification-aware decomposition records both families of rows rather than identifying one with the other. □

Theorem 9462.478 (Verification gaps block fully hardened decomposition)**.** *If a decomposition contains a verification claim without its transcription ledger or verification ledger, it cannot be fully hardened.*

Proof. By Definition 9462.438 and Definition 9462.443, a verified component without these ledgers is not audit-complete. Fully hardened decomposition requires all verification claims in the decomposition to be audit-complete. □

9462.18.4 Value and Discovery

Definition 9462.479 (Decomposition information)**.** *For a decomposition D, its decomposition information is*

$$\mathrm{Info}_{\mathrm{dec}}(D) = \mathrm{Benefit}_{\mathrm{dec}}(D) - \mathrm{Cost}_{\mathrm{dec}}(D) - \mathrm{Debt}_{\mathrm{dec}}(D).$$

The benefit term may count coverage, prediction, classifier clarity, transfer, intervention, unification, and verification gain. The cost and debt terms count certificate construction, ledger work, NDNA work, formal hardening, remaining gaps, and seal cost.

Definition 9462.480 (Positive and discovery-resolving decompositions)**.** *A decomposition is positive when* $\mathrm{Info}_{\mathrm{dec}}(D) > 0$*. It is a positive black-box resolution when it is positive and in closed-explanation state. It is a discovery-resolving decomposition when the transition to the resolved state includes classifier shift and positive decomposition-information gain.*

Theorem 9462.481 (Closure does not imply positive value)**.** *Closed-explanation state does not imply positive decomposition information.*

Proof. Closure records that the certificate and ledger obligations are met. Those obligations can be costly, and the resulting behavior coverage or transfer value may be small. If cost plus remaining debt is at least the benefit, the decomposition information is not positive. □

Theorem 9462.482 (Positive decomposition is not automatically discovery)**.** *Positive decomposition information does not by itself imply discovery.*

Proof. The positive value may come from ledger repair, source clarification, verification gain, reduced debt, or better scope sealing while the classifier relation remains unchanged. Discovery at this level requires classifier shift, as in Definition 9462.51. □

Theorem 9462.483 (Discovery resolution requires closure, classifier shift, and positive gain)**.** *A discovery-resolving decomposition requires closed-explanation state, classifier shift, and positive decomposition-information gain.*

Proof. Resolution supplies closure, classifier shift supplies the discovery axis, and positive decomposition-information gain supplies the value axis. Omitting any one of these conditions leaves either an unclosed explanation, a non-discovery audit improvement, or a nonpositive resolution. □

9462.18.5 Practice Illusions

Definition 9462.484 (Explanation illusion)**.** *An explanation illusion is a claim that appears to explain B while its BEDC decomposition still has a critical certificate, ledger, scope, semantic, NDNA, or verification gap.*

Theorem 9462.485 (Feature and circuit records are not decompositions)**.** *A feature label or circuit diagram is not a black-box resolution unless it is embedded in a ledger-complete certificate for a supporting classifier.*

Proof. Feature and circuit records may contribute to the pattern row, but they do not by themselves supply source scope, classifier relation, stability row, ledger row, seal, or not-claimed boundary. They are evidence fragments until the missing rows are supplied. □

Theorem 9462.486 (Probe and benchmark gains are not decompositions)**.** *Probe accuracy or benchmark improvement does not by itself resolve a machine-learning black box.*

Proof. A probe is a classifier candidate that needs its own certificate and ledger. A benchmark improvement reports performance movement, not the source, classifier, stability, ledger, and seal rows explaining the behavior. Either can support a decomposition, but neither is closure by itself. □

Theorem 9462.487 (Local proof is not black-box resolution)**.** *A local formally verified statement is not a black-box resolution unless the surrounding decomposition is certificate-complete, ledger-complete, and verification-aware.*

Proof. The local proof hardens a transcribed statement. It does not by itself provide the behavior source, classifier boundary, stability scope, explanation ledger, transcription ledger, verification ledger, or seal. Those rows are required for black-box resolution. □

Theorem 9462.488 (BEDC decomposition theorem for machine-learning black boxes)**.** *For a displayed machine-learning behavior B, BEDC decomposes its black-box status into support candidates, certificate fields, ledger gaps, debt, NDNA loci, optional verification hardening, and value accounting. If no support candidate is displayed, B is unexplained. If a support candidate exists but its certificate or ledger is incomplete, B has black-box residue. If a support candidate has complete certificate rows, complete ledger rows, and a sealed scope, B is resolved on that scope. Positive discovery-resolution additionally requires classifier shift and positive decomposition-information gain.*

Proof. Definition 9462.463 separates unexplained behavior from certificate-deficit black boxes. Definition 9462.464 and Definition 9462.465 identify the remaining residue when rows or ledgers are absent. Definition 9462.466 gives scoped resolution. Definition 9462.475 connects NDNA to BEDC closure, while Definition 9462.477 separates formal hardening from certificate closure. The value and discovery conditions are given by Definition 9462.481, Definition 9462.482, and Definition 9462.483. Together these statements give the decomposition theorem. □

Concretized in §6100: finite certificate-deficit packets realize the black-box decomposition surface as NameCert data

9462.19 No Global Black-Box Eliminator

section 9462.18 decomposes a machine-learning black box into support candidates, certificate fields, ledger gaps, debt, NDNA loci, optional verification hardening, and value accounting. The present section records the corresponding limitation: black-box resolution is scope-relative unless a global ledger is displayed. A total feature label, total diagram, total proof, or total interpretability narrative is not a global eliminator unless it carries the behavior family, classifier family, NameCert family, global ledger, verification ledger where applicable, and not-claimed boundary.

Definition 9462.489 (Explanation scope)**.** *An explanation scope is a tuple*

$$\Omega = (D, M_\theta, \mathcal{A}, B),$$

where D is the input or behavior source domain, M_θ is the model state, $\mathcal{A}$ is the admitted perturbation, transformation, transfer, or rewrite family, and B is the behavior being explained. An explanation is scoped when its certificate and ledger claims are made only inside Ω.

Definition 9462.490 (Scoped black-box resolution)**.** *Behavior B is resolved on scope Ω when there is an explanation object*

$$E_\Omega = (B, C, N_C, \Lambda_C)$$

such that $C \Vdash_\Omega B$, $N_C = \mathsf{NameCert}(C)$, the ledger Λ_C is complete for (B, C, Ω), and the seal records Ω together with the not-claimed boundary.

Definition 9462.491 (Global behavior family)**.** *For a model state M_θ, the global behavior family $\mathcal{B}(M_\theta)$ is the family of behaviors being covered by a global explanation claim. It may contain output behaviors, refusals, errors, transfer behaviors, safety behaviors, intervention behaviors, and boundary behaviors.*

Definition 9462.492 (Global resolution and global ledger)**.** *A model state is globally resolved when every $B \in \mathcal{B}(M_\theta)$ has a closed explanation and these explanations are covered by a common global ledger. A global ledger Λ_{global} records, for every covered behavior and supporting classifier, the required source, pattern, classifier, stability, semantic, intervention, verification, generalization, and not-claimed residue:*

$$\mathrm{Req}(B, C_B) \subseteq \mathrm{Rec}(\Lambda_{\text{global}}).$$

Definition 9462.493 (Global black-box eliminator)**.** *A global black-box eliminator is an explanation claim that asserts global resolution of M_θ. It is uncertified when it omits the covered behavior family, the supporting classifier family, the NameCert family, the global ledger, the verification ledger for verified rows, or the not-claimed boundary.*

Theorem 9462.494 (Scoped resolution does not imply global resolution)**.** *Resolving B on a scope Ω does not imply global resolution of M_θ.*

Proof. Scoped resolution supplies a closed explanation only for the target (B, Ω). Global resolution requires closed explanations for every behavior in $\mathcal{B}(M_\theta)$ and a common ledger covering their required residue. The scoped certificate gives neither the remaining behaviors nor their global ledger. □

Concretized in §6249: ScopedExplanationLedgerUp realizes scoped black-box certification as finite explanation-scope, classifier, ledger, refusal, transport, and NameCert rows.

Theorem 9462.495 (Finite-sample closure is not open-domain closure)**.** *Closure on a finite sample domain D_0 does not by itself imply closure on a larger open domain D.*

Proof. The finite certificate records required residue for D_0. A larger domain may contain boundary cases, out-of-distribution cases, semantic ambiguity, and generalization residue not recorded by the finite ledger. Thus the ledger inclusion for D_0 does not entail the ledger inclusion for D. □

Theorem 9462.496 (Global resolution requires global ledger)**.** *Any global resolution claim for M_θ requires a global ledger.*

Proof. Global resolution requires each behavior in $\mathcal{B}(M_\theta)$ to have a closed explanation. Each closed explanation requires ledger coverage for its required residue. Taking the family of all such required rows gives the required residue of the global claim, and a ledger covering that family is precisely the global ledger. □

Theorem 9462.497 (No uncertified global black-box eliminator)**.** *An uncertified global black-box eliminator is not a valid global resolution claim.*

Proof. By Definition 9462.496, global resolution requires a global ledger. It also requires behavior coverage, supporting classifiers, NameCert rows, and the relevant verification and not-claimed ledgers. An eliminator claim omitting these rows lacks the conditions it asserts. □

9462.19.1 Classifier and Semantic Globalization

Definition 9462.498 (Single-classifier explanation)**.** *A single-classifier explanation uses one classifier $C^\star$ to explain several behaviors. Two behaviors are classifier-separated when their closed explanations require nonequivalent classifier relations unless a product classifier or bridge ledger is supplied.*

Definition 9462.499 (Product classifier and bridge ledger)**.** *For classifiers $C_1, \ldots, C_n$, a product classifier is the joint relation*

$$C_\Pi(x, y) \iff \bigwedge_{i=1}^{n} C_i(x, y).$$

A bridge ledger for such a product records each source scope, each classifier boundary, the distinctions created by the product, the behaviors unified or separated by it, and its not-claimed boundary.

Theorem 9462.500 (A single classifier does not automatically explain many behaviors)**.** *If B_1 and B_2 are supported by nonequivalent classifiers, a closed explanation of B_1 by C_1 does not automatically close an explanation of B_2.*

Proof. The certificate for $C_1 \Vdash B_1$ records the source, classifier, stability, and ledger rows for B_1. Nonequivalence of the classifiers means that the public classification surface needed for B_2 can differ. Without a proof that C_1 supports B_2, or a certified product or bridge, the rows for B_2 are absent. □

Theorem 9462.501 (Product classifiers require bridge ledgers)**.** *A product classifier used to unify several behavior explanations requires a bridge ledger.*

Proof. The product classifier changes classification granularity. It can split old cells, expose boundary cases, and change which behavior rows are covered by which classifier. These residues are created by the unification operation itself and must be recorded for the unified claim to close. □

Definition 9462.502 (Semantic globalizer)**.** *A semantic globalizer is an explanation claim that uses one natural language concept K to unify several features, classifiers, behaviors, or model states. It overreaches when it lacks a semantic ledger for positive cases, negative cases, ambiguous cases, polysemy, domain dependence, context dependence, label drift, unsupported connotations, and not-claimed semantics.*

Theorem 9462.503 (Semantic globalization requires semantic ledger)**.** *A natural-language concept used as a global explanation requires a semantic global ledger.*

Proof. Natural-language concepts have ambiguous boundaries, domain dependence, and unsupported connotations. A global use of such a concept must record which behaviors fall under it, which do not, which are ambiguous, and where the concept is not claimed. Those rows are exactly the semantic global ledger. □

9462.19.2 Formal and Diagonal Residue

Definition 9462.504 (Formal globalizer)**.** *A formal globalizer uses a checked judgment*

$$\mathcal{V} \vdash \pi : \varphi$$

as a claim about the model or behavior family beyond the scope of the statement φ.

Definition 9462.505 (Verification global ledger)**.** *A verification global ledger records checked statements, their scopes, unverified components, backend, trust boundary, proof dependencies, translation assumptions, portability boundary, and behaviors not covered by the verification claim.*

Theorem 9462.506 (Scoped proof does not imply global explanation)**.** *A formally checked statement on scope Ω does not imply a global explanation unless the checked statement and its verification ledger are global.*

Proof. The proof establishes the formal statement that was checked. If the statement is scoped, its conclusion is scoped. A larger explanation claim requires additional behavior coverage, transcription ledgers, verification ledgers, and non-hardenable residue rows. □

Definition 9462.507 (Diagonal residue)**.** *For an explanation E, diagonal residue is a behavior case, boundary case, input, intervention, counterexample, out-of-distribution case, or semantic ambiguity that lies in a not-covered region or ledger gap of E.*

Definition 9462.508 (Self-exposure test)**.** *A self-exposure test for E searches for unrecorded source rows, unrecorded boundaries, false positives, false negatives, semantic ambiguity, untested interventions, verification gaps, and generalization residue.*

Theorem 9462.509 (Non-global ledgers admit diagonal-residue risk)**.** *If an explanation lacks a global ledger, then diagonal residue remains a possible risk.*

Proof. Without a global ledger, the required residue of the global claim is not known to be covered by the recorded ledger rows. Any uncovered required row may be exposed by an input, behavior case, intervention, or semantic ambiguity. The absence of the global ledger is therefore a diagonal residue risk even when a specific witness has not yet been displayed. □

Theorem 9462.510 (Diagonal residue forces classifier refinement or ledger repair)**.** *When diagonal residue changes the boundary of the supporting classifier, the explanation must either refine the classifier or repair the ledger.*

Proof. If the residue shows that the old classifier wrongly joins or separates cases, the classifier relation must change. If the classifier is kept, the residue must be recorded as a boundary, failure case, or not-claimed row. These are respectively classifier refinement and ledger repair. □

9462.19.3 Cost and Free Explanation

Definition 9462.511 (Global explanation cost)**.** *For a global behavior family, the global explanation cost is the cost of its NameCert rows, ledger rows, verification rows where used, and seal rows. When the behavior family is finite this can be written as*

$$\mathrm{Cost}_{\mathrm{global}}(M_\theta) = \sum_{B \in \mathcal{B}(M_\theta)} \big(\mathrm{NameCertCost}(B) + \mathrm{LedgerCost}(B) + \mathrm{SealCost}(B)\big).$$

For nonfinite behavior families, the sum is replaced by the appropriate cost observer for the declared scope.

Definition 9462.512 (Global explanation compression)**.** *Global explanation compression uses a smaller family of schemas to cover many behavior explanations. The compression is certified only when its schema ledger records coverage, failure cases, uncovered behavior classes, schema-specific debt, and the bridge from schemas to behaviors.*

Theorem 9462.513 (Global explanation has nonzero certificate cost)**.** *If the covered behavior family contains a nontrivial behavior, then a global explanation has nonzero certificate cost.*

Proof. A nontrivial covered behavior requires at least a supporting classifier, certificate rows, ledger rows, and a seal. These rows are nonempty audit objects. Their construction and maintenance cost is therefore not zero. □

Theorem 9462.514 (Compressed global explanations require schema ledgers)**.** *A schema that compresses many behavior explanations requires a schema ledger.*

Proof. Compression hides behavior-specific boundaries, failure cases, domain differences, and residue. The schema ledger records which cases the schema covers, which it does not, and which bridge rows connect the schema to each behavior explanation. Without those rows the compressed global claim is unclosed. □

Definition 9462.515 (Free explanation and free gain)**.** *A free explanation claims that E explains B without supplying a classifier, NameCert, ledger, scope, and not-claimed boundary. A free global explanation makes the same omission for an entire behavior family. A free interpretability-gain claim asserts positive explanation information without subtracting certificate cost, ledger cost, verification cost, and remaining debt.*

Theorem 9462.516 (No free explanation)**.** *A free explanation is not a closed explanation.*

Proof. Closed explanation requires a supporting classifier, certificate rows, ledger rows, scope seal, and not-claimed boundary. A free explanation omits these rows by definition, so it lacks the closure conditions. □

Theorem 9462.517 (No free global explanation)**.** *A free global explanation is not a valid global resolution claim.*

Proof. A free global explanation is already missing the rows required for ordinary closed explanation, and global resolution additionally requires global behavior coverage and a global ledger. It therefore lacks both local closure rows and global closure rows. □

Theorem 9462.518 (No free interpretability gain)**.** *An interpretability-gain claim that omits certificate cost, ledger cost, verification cost, or remaining debt is not audit-complete.*

Proof. Explanation information is benefit minus cost and debt. Omitting any required cost or debt term overstates the value of the explanation and prevents an auditor from determining whether the net gain is positive. □

Theorem 9462.519 (No global black-box eliminator theorem)**.** *There is no uncertified global black-box eliminator. Machine-learning black-box resolution is scoped, classifier-relative, NameCert-bound, and ledger-complete unless a global behavior family, classifier family, NameCert family, global ledger, verification ledger where applicable, and global seal are displayed.*

Proof. Definition 9462.494 and Definition 9462.495 separate scoped and finite closure from global closure. Definition 9462.496 gives the global-ledger requirement, and Definition 9462.497 rules out eliminator claims that omit it. Definition 9462.500, Definition 9462.501, and Definition 9462.503 rule out unsupported classifier and semantic globalization. Definition 9462.506 rules out proof-scope overreach. Definition 9462.509 and Definition 9462.510 identify the diagonal-residue risk. Definition 9462.513, Definition 9462.514, Definition 9462.516, Definition 9462.517, and Definition 9462.518 give the cost and audit requirements. Together these statements prove the theorem. □

9462.20 Unified Theory Closure

The preceding machine-learning and interpretability sections form one closure pattern. Generated material is not yet information, a learned parameter state is not yet a learning claim, a feature label is not yet an explanation, a checked proof is not yet a complete interpretation, and a black box is not a primitive opacity. In the BEDC reading, each of these becomes auditable only when a classifier surface is certified, ledgered, placed into a role-complete packet, and sealed by scope.

Definition 9462.520 (Unified epistemic object)**.** *A unified epistemic object is a tuple*

$$\mathfrak{U} = (G, R, C, N, \Lambda, D, V, \Omega),$$

where G is generated or observed material, R is its record surface, C is the classifier relation, N is the NameCert or Tensor NameCert, Λ is the ledger, D is the theorem-DNA or neural theorem-DNA packet, V is the formal-hardening profile, and Ω is the scope seal.

Concretized in §6284: Unified epistemic objects are realized as generated-material, classifier, certificate, ledger, theorem-DNA, hardening-profile, and scope-seal NameCert rows.

Definition 9462.521 (Unified layers)**.** *The generated-material layer records histories, event flows, theorem candidate flows, proof traces, tensor traces, activation records, learning records, intervention logs, and verification artifacts. The classifier layer declares how such material is grouped. The certificate layer records source, pattern, classifier, stability, and ledger-policy rows. The ledger layer records residue from identification, compression, projection, thresholding, generalization, semantic assignment, intervention, and formal verification. The DNA layer records the role-complete packet. The verification layer records which components are hardened and under which backend. The scope seal records source scope, target scope, claim kind, not-claimed boundary, and closure route.*

Definition 9462.522 (Closure axes)**.** *The object $\mathfrak{U}$ has material closure when G is traceable and referencable; classifier closure when C is displayed for the stated scope; NameCert closure when N supplies the five certificate rows; ledger closure when*

$$\mathrm{Req}(\mathfrak{U}) \subseteq \mathrm{Rec}(\Lambda);$$

DNA closure when every critical locus of D is complete, anchored, and certificate-compatible; verification closure when every critical hardenable component is hardened and every non-hardenable residue is ledgered; and scope closure when Ω states the scope, claim kind, not-claimed boundary, and closure route.

Definition 9462.523 (Theory and hardening closure)**.** *Theory closure for $\mathfrak{U}$ is the conjunction of material, classifier, NameCert, ledger, DNA, and scope closure. Fully hardened closure is theory closure together with verification closure.*

Theorem 9462.524 (Hardening closure implies theory closure)**.** *Every fully hardened closed object is theory-closed.*

Proof. Fully hardened closure is defined as theory closure plus verification closure. The theory-closure component is therefore inherited. □

Theorem 9462.525 (NameCert closure without ledger closure is not theory closure)**.** *NameCert closure alone does not imply theory closure when ledger closure fails.*

Proof. Theory closure requires ledger closure. If required residue is not recorded by the ledger, one conjunct of theory closure fails even when the NameCert rows are present. □

Theorem 9462.526 (DNA closure without NameCert compatibility is not theory closure)**.** *A complete DNA packet that is not certificate-compatible does not yield theory closure.*

Proof. DNA closure records completeness and anchoring of loci. Theory closure also requires the classifier to have a compatible NameCert. If the DNA packet cannot project to such a certificate, the NameCert-closure axis fails. □

Theorem 9462.527 (Verification closure does not replace ledger closure)**.** *Verification closure without ledger closure is not theory closure.*

Proof. Verification closure audits checked statements and their hardening profile. Ledger closure audits source, semantic, intervention, generalization, projection, and not-claimed residue. Since theory closure requires ledger closure, verification cannot substitute for it. □

9462.20.1 Unified Information and Discovery

Definition 9462.528 (Unified information)**.** *The unified benefit of $\mathfrak{U}$ counts coverage, prediction, transfer, intervention, unification, verification, and compression gains. The unified cost counts NameCert, ledger, DNA, verification, seal, and maintenance costs. The unified debt counts ledger, semantic, verification, generalization, intervention, DNA, and scope debt. The unified information is*

$$\mathrm{UInfo}(\mathfrak{U}) = \mathrm{UBenefit}(\mathfrak{U}) - \mathrm{UCost}(\mathfrak{U}) - \mathrm{UDebt}(\mathfrak{U}).$$

Definition 9462.529 (Unified transition and classifier shift)**.** *A unified transition is a passage*

$$\mathfrak{U}_0 \rightsquigarrow \mathfrak{U}_1.$$

It has unified classifier shift when the classifier layers of $\mathfrak{U}_0$ *and* $\mathfrak{U}_1$ *are not equivalent on their common declared source. Its unified information gain is*

$$\Delta\,\mathrm{UInfo} = \mathrm{UInfo}(\mathfrak{U}_1) - \mathrm{UInfo}(\mathfrak{U}_0).$$

Definition 9462.530 (Unified positive discovery)**.** *A unified transition is a unified positive discovery when it has classifier shift, the target* $\mathfrak{U}_1$ *is theory-closed, and*

$$\Delta\,\mathrm{UInfo} > 0.$$

Theorem 9462.531 (Positive information gain is not classifier discovery)**.** *Positive unified information gain does not by itself imply classifier shift.*

Proof. The gain may come from ledger repair, verification-debt reduction, cost compression, source-scope clarification, seal strengthening, or semantic ledger repair while the classifier relation is preserved. Such a transition can have positive value but no classifier shift. □

Theorem 9462.532 (Classifier shift is not positive information gain)**.** *Classifier shift does not by itself imply positive unified information gain.*

Proof. A changed classifier can create new certificate, ledger, DNA, verification, and seal obligations. If those costs and debts exceed the benefit produced by the new classifier, the unified information gain is not positive. □

Theorem 9462.533 (Unified positive discovery has two audits)**.** *Every unified positive discovery passes both a classifier-shift audit and a net-information audit.*

Proof. Classifier shift and positive unified information gain are both defining conditions of Definition 9462.530. Theory closure is the certification condition tying those two audits to a closed target object. □

9462.20.2 Speakable Model Form

Definition 9462.534 (BEDC-speakable interface)**.** *A target* X *is* BEDC-speakable *on a declared scope when public use of its upward interface factors through a certificate surface*

$$X^{\uparrow} = (S_X, P_X, C_X, \Sigma_X, \Lambda_X),$$

where S_X *is the source specification,* P_X *is the pattern specification,* C_X *is the classifier specification,* Σ_X *is the stability certificate, and* Λ_X *is the ledger policy. The tuple is read only through the generated-history, continuation, probe-signature, package, gap-ledger, and naming-certificate route*

$$\mathsf{Hist} \longrightarrow \mathsf{Cont} \longrightarrow \mathsf{Ask} \longrightarrow \mathsf{Sig} \longrightarrow \mathsf{Pkg} \longrightarrow \mathsf{Gap} \longrightarrow \mathsf{NameCert}.$$

Thus $X^{\uparrow}$ *is not a host object placed behind a name. It is the licensed public interface by which the name* X *may be consumed.*

Definition 9462.535 (BEDC model form)**.** *The* BEDC model form *of a speakable interface* $X^{\uparrow}$ *is the role pattern*

$$\mathrm{ModelForm}(X) = (\mathsf{Hist}_X, \mathsf{hsame}_X, \mathsf{Cont}_X, \mathsf{Ask}_X, \mathsf{Sig}_X, \mathsf{sameSig}_X, \mathsf{Pkg}_X, \mathsf{Gap}_X, \mathsf{NameCert}_X).$$

Two speakable interfaces have the same model form when there are bidirectional role translations preserving generated-history source, internal sameness transport, continuation, probe-signature readback, signature comparison, package rows, ledger residue, and NameCert admission, with round trips witnessed only by the relevant internal hsame, sameSig*, or classifier-preserving transport.*

Theorem 9462.536 (Speakability factors through model form)**.** *Every theory-closed unified object is an instance of the BEDC model form.*

Proof. By Definition 9462.523, theory closure is the conjunction of material, classifier, NameCert, ledger, DNA, and scope closure. The material axis supplies generated histories and record surfaces. The classifier axis supplies the comparison relation by which records are grouped. NameCert closure supplies the source, pattern, classifier, stability, and ledger-policy rows. Ledger closure records the residue of projection, compression, identification, thresholding, semantic assignment, and scope restriction. The scope seal states the claim kind and not-claimed boundary. These are exactly the roles listed in Definition 9462.535, read through the route of Definition 9462.534. □

Theorem 9462.537 (Model-form agreement is not object identity)**.** *If two speakable interfaces preserve the same BEDC model form, the licensed conclusion is structural comparison, not host-level equality or content collapse.*

Proof. Model-form agreement preserves only the public roles displayed in Definition 9462.535. The round trips are witnessed by internal sameness, signature comparison, or classifier-preserving transport. None of these is host equality, and none identifies the source material, pattern evidence, classifier scope, or ledger residue of one target with that of another. Thus a theorem instance, a tensor behavior instance, and a physical-observation instance may share the same certification grammar without becoming the same object. □

Corollary 9462.538 (Projection is not ontology)**.** *A probe result, signature, package, projection, compression, or displayed model output is not the target whose interface it supports unless a separate exactness, recovery, or ledger certificate licenses that reading.*

Proof. Each listed output is one role or one role fragment inside the BEDC-speakable interface. Reading it as the whole target would discard the remaining source, classifier, stability, ledger, and scope rows. Theory closure forbids that discard unless the missing residue is recovered or recorded by the certificate surface. □

9462.20.3 Specializations

Definition 9462.539 (Unified specializations)**.** *A learning instance is a unified object whose generated material consists of training records, activation traces, learned features, and behavior records. An interpretability instance has a neural behavior target and a Neural Theorem-DNA layer. A theorem instance has a theorem-like target and theorem-DNA layer. A black-box decomposition instance has a behavior target together with the BEDC decomposition data of its supporting classifier candidates.*

Theorem 9462.540 (Learning discovery is a unified positive discovery instance)**.** *A learning transition that produces certified classifier shift, learning ledger closure, and positive learning information gain is a unified positive discovery instance.*

Proof. The certified classifier shift supplies unified classifier shift. The Learning NameCert and ledger closure supply theory closure for the learning instance. Positive learning information gain supplies positive unified information gain. □

Theorem 9462.541 (Interpretability discovery is a unified positive discovery instance)**.** *An interpretability transition with classifier shift, Tensor NameCert, NDNA closure, ledger closure, and positive explanation-information gain is a unified positive discovery instance.*

Proof. Classifier shift supplies the discovery axis. Tensor NameCert, NDNA closure, ledger closure, and scope seal supply theory closure. Positive explanation-information gain is the interpretability specialization of positive unified information gain. □

Theorem 9462.542 (Black-box discovery-resolution is a unified positive discovery instance)**.** *A black-box decomposition transition that resolves the scoped black box, introduces classifier shift, carries NameCert and ledger rows, and has positive decomposition-information gain is a unified positive discovery instance.*

Proof. Scoped resolution supplies closure, the new classifier supplies classifier shift, and positive decomposition-information gain supplies the value condition. These are exactly the unified positive discovery conditions under the black-box decomposition specialization. □

9462.20.4 False Discovery and Lineage

Definition 9462.543 (Unified false discovery claim)**.** *A unified false discovery claim asserts unified positive discovery while lacking classifier shift, theory closure, positive information gain, cost protocol, ledger closure, or scope seal. Record growth, parameter growth, loss reduction, benchmark gain, feature count, proof count, ledger-entry count, semantic relabeling, local verification, scope narrowing, and hidden debt are not substitutes for these conditions.*

Theorem 9462.544 (Record growth and hardening alone are not unified positive discovery)**.** *If a transition only adds records or hardens verification rows while preserving the classifier, it is not unified positive discovery.*

Proof. Unified positive discovery requires classifier shift. Record growth and verification hardening may improve audit status or reduce debt, but when the classifier is preserved they fail the classifier-shift condition. □

Theorem 9462.545 (Hidden debt invalidates positivity claims)**.** *If a claimed positive gain depends on unrecorded debt, the unified positive discovery claim is not audit-complete.*

Proof. Unified information subtracts unified debt. Hiding debt overstates $\Delta\,\mathrm{UInfo}$, and restoring the omitted term may remove strict positivity. The claim cannot be audited until that debt is recorded. □

Definition 9462.546 (Closure lineage)**.** *A closure lineage is a sequence*

$$\mathfrak{U}_0 \rightsquigarrow \mathfrak{U}_1 \rightsquigarrow \cdots \rightsquigarrow \mathfrak{U}_n$$

whose edges may be material extension, classifier shift, NameCert completion, ledger repair, DNA completion, seal strengthening, formal hardening, information compression, or positive discovery. An audit lineage improves closure or debt without classifier shift. A discovery lineage contains classifier shift. A positive closure lineage ends in theory closure with positive total unified information gain. A fully hardened lineage ends in fully hardened closure.

Theorem 9462.547 (Audit, discovery, positivity, and hardening are separate lineage axes)**.** *Audit lineage, discovery lineage, positive closure lineage, and fully hardened lineage do not imply one another in general.*

Proof. An audit lineage can lower debt without changing the classifier. A discovery lineage can introduce a classifier before its certificate and ledger costs are paid. A positive closure lineage can close and yield positive value without full formal hardening. A fully hardened lineage can harden an old classifier without classifier shift. The axes therefore record distinct properties. □

9462.20.5 Unified Closure Theorems

Definition 9462.548 (Unified closed, hardened, and discovery objects)**.** *A unified object is closed when it has theory closure. It is hardened when it has fully hardened closure. It is a discovery object when it is the target of a classifier-shift transition with theory closure. It is a positive discovery object when, in addition, the transition has positive unified information gain.*

Theorem 9462.549 (Unified closure theorem)**.** *A unified object is closed exactly when its generated material is traceable, its classifier is displayed, its NameCert rows are complete, its ledger covers required residue, its DNA critical loci are complete, anchored, and certificate-compatible, and its scope seal states the claim kind and not-claimed boundary.*

Proof. This is the expansion of theory closure into its material, classifier, NameCert, ledger, DNA, and scope axes. Each listed condition is one axis; together they are the conjunction defining theory closure. □

Theorem 9462.550 (Unified hardening theorem)**.** *A unified object is hardened exactly when it is closed and every critical hardenable component is formally hardened with complete verification ledger, while non-hardenable residue is ledgered.*

Proof. Fully hardened closure is theory closure together with verification closure. The stated condition expands verification closure and combines it with the closed-object condition. □

Theorem 9462.551 (Unified discovery theorem)**.** *A transition is unified discovery exactly when the classifier layer changes and the target object is theory-closed. It is unified positive discovery exactly when, additionally,* $\Delta\,\mathrm{UInfo} > 0$.

Proof. The first sentence expands the definition of discovery object. The second adds the positive unified-information condition from Definition 9462.530. □

Theorem 9462.552 (Unified BEDC information theorem)**.** *In the BEDC reading, information, learning, interpretability, black-box resolution, and formal verification are unified by the chain*

$$\textit{generated material} \longrightarrow \textit{classifier} \longrightarrow \mathsf{NameCert} \longrightarrow \textit{ledger} \longrightarrow \textit{DNA} \longrightarrow \textit{scope seal} \longrightarrow \textit{optional formal hardening}$$

Information growth is certified classifier shift rather than record growth; learning is certified classifier formation rather than parameter motion; interpretability is Tensor NameCert and NDNA

closure rather than feature labeling; black-box resolution is decomposition closure rather than narrative explanation; and formal verification is a hardening layer rather than the object-formation rule.

Proof. Definition 9462.57 separates information growth from record growth. Definition 9462.540 identifies learning discovery as a unified instance. Definition 9462.541 does the same for interpretability discovery. Definition 9462.542 does the same for black-box discovery-resolution. Definition 9462.527 and Definition 9462.524 separate formal hardening from theory closure. The displayed chain is the common closure route shared by these cases. □

9462.21 Unified Schematic Map

section 9462.20 gives the closure theorem. The present section records the same result as a schematic map: each layer is read by the next layer, and no lower layer is allowed to impersonate the closure supplied by a higher layer.

Definition 9462.553 (Primitive-to-certified route)**.** *The primitive-to-certified route is the ordered reading*

$$\mathsf{Distinction} \longrightarrow \mathsf{History} \longrightarrow \mathsf{Record} \longrightarrow \mathsf{Classifier} \longrightarrow \mathsf{NameCert} \longrightarrow \mathsf{Ledger} \longrightarrow \mathsf{CertifiedObject}.$$

The first three layers record generated material; the classifier layer organizes material into stable same-and-different judgments; the certificate and ledger layers make that organization auditable.

Concretized in §6295: The primitive-to-certified route is realized as an ordered NameCert carrier from generated distinction rows through classifier, ledger, and certified-object seal.

Theorem 9462.554 (Record growth is below classifier discovery)**.** *Record growth, longer traces, shorter presentations, new names, and formal hardening do not by themselves imply information discovery.*

Proof. Each can occur while the classifier relation is preserved. Discovery in the classifier-shift reading requires a changed classifier surface together with certificate and ledger support, not merely additional material, altered presentation, or changed verification status. □

Definition 9462.555 (Machine-learning certification stack)**.** *The machine-learning certification stack reads tensor traces, activations, soft features, probes, thresholds, and behaviors as classifier candidates. A candidate becomes an auditable explanation only through Tensor NameCert rows: tensor source, tensor pattern, tensor classifier, tensor stability, and tensor ledger. Formal verification may harden selected rows, but it does not replace the Tensor NameCert or the ledger.*

Theorem 9462.556 (Feature evidence is below Tensor NameCert)**.** *Features, top examples, probe accuracy, visualizations, and tensor directions are evidence fragments rather than certified explanations unless they enter a Tensor NameCert with ledger coverage.*

Proof. Each fragment can support a pattern or classifier row, but none supplies the full source, pattern, classifier, stability, ledger, and seal surface by itself. Certification begins only when those rows are jointly displayed. □

Definition 9462.557 (NDNA schematic)**.** *The neural theorem-DNA schematic for a behavior B is the role family*

$$\mathsf{NDNA}(B) = (G_{\mathsf{source}}, G_{\mathsf{feature}}, G_{\mathsf{circuit}}, G_{\mathsf{classifier}}, G_{\mathsf{stability}}, G_{\mathsf{intervention}}, G_{\mathsf{ledger}}, G_{\mathsf{seal}}).$$

It is an explanation genome only when the loci are complete, anchored, NameCert-compatible, ledgered, and sealed.

Theorem 9462.558 (Single loci are not explanation genomes)**.** *A feature label, circuit diagram, classifier relation, intervention record, or semantic label is a single locus or locus fragment, not an NDNA-certified explanation.*

Proof. The NDNA packet requires source, feature, circuit, classifier, stability, intervention, ledger, and seal loci. A single displayed locus leaves the remaining roles absent or unanchored. □

9462.21.1 Black-Box and Information Schematics

Definition 9462.559 (Black-box schematic)**.** *The black-box schematic is*

$$\mathsf{BlackBox}(B) \iff \exists C.\, C \Vdash B \text{ and } \neg\exists N.\, N = \mathsf{NameCert}(C).$$

Resolution on a scope is the package

$$C \Vdash B \quad + \quad \mathsf{NameCert}(C) \quad + \quad \mathsf{LedgerComplete}(C, B) \quad + \quad \mathsf{Seal}(B, C).$$

Definition 9462.560 (Five-field deficit schematic)**.** *For a behavior B and classifier candidate C, the five-field deficit vector is*

$$\mathbf{D}(B, C) = (d_S, d_P, d_C, d_\Sigma, d_L) \in \{0, 1\}^5.$$

The coordinates record source, pattern, classifier, stability, and ledger gaps. The all-zero vector together with zero critical ledger debt is the field-level signature of scoped BEDC resolution.

Concretized in §6303: The five-field explanation schematic is realized as finite source, classifier, evidence, ledger-gap, certification-route, transport, replay, provenance, and NameCert rows.

Theorem 9462.561 (Ledger gaps dominate scoped resolution)**.** *If the ledger coordinate or critical ledger debt is nonzero, scoped black-box resolution fails.*

Proof. Scoped resolution requires ledger completeness. A nonzero ledger coordinate or nonzero critical ledger debt records an unclosed required residue row, so the resolution package is incomplete. □

Definition 9462.562 (Information schematic)**.** *For an explanation or unified object E, the auditable information schematic is*

$$\mathsf{Info}(E) = \mathsf{Benefit}(E) - \mathsf{Cost}(E) - \mathsf{Debt}(E).$$

A transition is positive only when the corresponding information difference is strictly positive under a public cost protocol.

Definition 9462.563 (Discovery schematic)**.** *Certified discovery is classifier shift with NameCert and ledger rows. Positive discovery is certified discovery with positive net information gain:*

$$\mathsf{Discovery}^{+} = \mathsf{ClassifierShift} \quad + \quad \mathsf{NameCert} \quad + \quad \mathsf{Ledger} \quad + \quad \Delta\mathsf{Info} > 0.$$

Theorem 9462.564 (Audit progress and discovery are separate schematic cases)**.** *Ledger repair, proof hardening, cost reduction, and seal clarification can be positive audit progress without being classifier discovery.*

Proof. Each may improve benefit, reduce cost, or lower debt while preserving the classifier relation. The discovery schematic requires classifier shift, so same-classifier audit progress is not discovery. □

9462.21.2 Learning, Generalization, and Verification

Definition 9462.565 (Learning schematic)**.** *The learning schematic reads training traces and activation records as candidate sources for stable tensor distinctions. Strong learning is the formation of a learned classifier together with Learning NameCert and learning ledger rows. Positive certified learning additionally requires positive learning information gain.*

Theorem 9462.566 (Performance movement is below certified learning)**.** *Loss reduction, benchmark gain, parameter growth, and memorization do not by themselves establish certified learning discovery.*

Proof. They can occur without a displayed classifier shift and without Learning NameCert or ledger rows. Certified learning discovery requires the classifier surface and its audit packet, not merely a performance or parameter event. □

Definition 9462.567 (Generalization schematic)**.** *Certified generalization is classifier transfer with ledger. It requires a source-domain classifier, target-domain support, transfer or stability witnesses, a generalization NameCert, and a target-residue ledger.*

Theorem 9462.568 (Accuracy transfer is below certified generalization)**.** *Target-domain performance is not yet a certified generalization explanation unless the classifier transfer and target residue are ledgered.*

Proof. Performance can transfer while the classifier boundary changes, while target failures remain unrecorded, or while the source classifier is not the one supporting the target behavior. The certified claim therefore requires classifier-transfer evidence and ledger coverage. □

Definition 9462.569 (Verification schematic)**.** *Formal verification follows the route*

$$F \longrightarrow \varphi_F \longrightarrow \pi_F \longrightarrow \mathcal{V} \vdash \pi_F : \varphi_F \longrightarrow \mathsf{Hardened}(F),$$

where F is a transcribed component. The route requires transcription ledger and verification ledger whenever it is used to support an interpretability claim.

Theorem 9462.570 (Verification hardens components rather than forming objects)**.** *Formal verification can harden selected components, but it does not by itself create the NameCert, explanation ledger, NDNA packet, or global scope ledger.*

Proof. The verification judgment concerns a formal statement and proof object. Object formation requires source, pattern, classifier, stability, ledger, role loci, and scope seal. Those rows are outside the proof judgment unless separately supplied. □

9462.21.3 Global and Final Schematics

Definition 9462.571 (Global-resolution schematic)**.** *Global resolution for a model state requires the covered behavior family, supporting classifier family, NameCert family, global ledger, and global seal. Without the global ledger, the claim remains scoped even when many local explanations are present.*

Theorem 9462.572 (No global ledger, no global explanation)**.** *A global explanation claim without a global ledger is not closed.*

Proof. By Definition 9462.496, global resolution requires ledger coverage for the required residue of all covered behaviors and supporting classifiers. Without that ledger, the global claim lacks its closure surface. □

Theorem 9462.573 (Unified schematic theorem)**.** *The schematic reading of the machine-learning layer, BEDC certification layer, and formal-verification layer is the single chain*

$$\mathsf{Distinction} \longrightarrow \mathsf{Classifier} \longrightarrow \mathsf{NameCert} \longrightarrow \mathsf{Ledger} \longrightarrow \mathsf{Explanation} \longrightarrow \mathsf{OptionalHardening}.$$

Black boxes are uncertified classifier structures; resolving them means supplying NameCert, ledger, and seal; learning is classifier formation; discovery is certified classifier shift; and positive discovery is certified classifier shift with positive net information gain.

Proof. The primitive-to-certified route supplies the lower chain. Definition 9462.555 and Definition 9462.557 specialize it to tensor interpretability. Definition 9462.559 gives the black-box and resolution equations. Definition 9462.563 gives the discovery and positive-discovery equations. Definition 9462.569 places formal verification as hardening rather than object formation. These are the listed schematic identifications. □

9462.22 Observation, Subject, and Time

The first layer identifies the path from distinction to history:

$$\text{distinction} \to \mathsf{Hist} \to \text{information}.$$

The present layer reads the next structural path:

$$\mathsf{Hist} \to \mathsf{Cont} \to \text{observation},$$
$$\mathsf{Hist} + \mathsf{hsame} \to \text{observer identity},$$
$$\mathsf{Cont}\text{-asymmetry} \to \text{time},$$
$$\text{multiple histories} \to \text{no global clock}.$$

Its compressed thesis is that the observer is a history, observation is continuation, and time is history growth. These are not additional primitives. They are the public readings of the same kernel surface.

9462.22.1 Internalizing observation

Definition 9462.574 (Observation)**.** *An observation is a continuation witness* $\mathsf{Cont}(h, d, h')$*, where* h *is the antecedent history,* d *is the displayed distinction or continuation tail, and* h' *is the successor history. Its public reading is that a history continues under a distinction into another history.*

Definition 9462.575 (Subject-free observation)**.** *A continuation is subject-free when its signature has no auxiliary observer-substance parameter. It does not contain a parameter of the form* s : Subject*, an owner map for histories, or a separate viewpoint carrier outside the history layer.*

Theorem 9462.576 (Observation is continuation, not representation)**.** *In BEDC, observation is the visible continuation of history rather than a mental representation owned by a hidden subject.*

Proof. The relation $\mathsf{Cont}(h, d, h')$ relates histories through a displayed distinction. Its endpoints lie in the history layer, and its witness is available for later transport, classifier comparison, and ledger assembly. It invokes no private mental state, external subject, or host equality oracle. Therefore the BEDC form of observation is generated continuation. □

Corollary 9462.577 (Observation needs no subject-substance)**.** *The BEDC theory of observation does not require a hidden subject entity.*

Proof. If the observation is already given by a continuation witness, adding a subject parameter contributes no new kernel witness. It instead introduces an unaudited carrier that is not generated by Hist, compared by hsame, or transported by a displayed continuation law. □

Corollary 9462.578 (Observation is a ledger event)**.** *Observation in BEDC is a public ledger event rather than an incomparable private inspection.*

Proof. A continuation witness can be reused by history-sameness transport, licensed classifiers, packages, and naming certificates. A purely private inspection that cannot enter these ledgers has no object-layer standing in BEDC. □

Concretized in §1266: Finite observation locks realize subject-free observation as a finite continuation, hsame, budget, readback, and NameCert packet.

Concretized in §1271: Subject-free observation packets realize Cont observations, hsame observer identity, no-subject exclusion, and public ledger-event exposure as one finite NameCert surface.

9462.22.2 Observer identity

Definition 9462.579 (Observer history identity)**.** *Two histories have the same observer identity when licensed observations do not separate them except through a displayed classifier witness against their internal history-sameness comparison. In short, observer identity is history identity read through licensed observational comparison.*

Definition 9462.580 (Observer ledger)**.** *An observer ledger is a finite evidence packet consisting of mark emissions, continuation witnesses, probe bundles, signature relations, derived classifiers, and the corresponding history-sameness or classifier comparison rows.*

Theorem 9462.581 (Observer identity remains internal)**.** *Every accepted BEDC judgment of observer identity is internal to the finite witness ledger.*

Proof. The observations entering the judgment are BEDC witnesses: marks, continuations, bundles, signatures, classifiers, and history-sameness comparisons. Repackaging them as an observer ledger uses only internal sameness and classifier relations. No host equality and no invisible subjective sameness criterion is introduced. □

Corollary 9462.582 (Same observer does not mean same mental entity)**.** *Two observers are identified by licensed history comparison, not by sharing a hidden mental substance.*

Proof. A hidden mental substance is not a mark emission, continuation, signature, classifier, or history-sameness row. It therefore cannot serve as a licensed BEDC identity criterion. □

Corollary 9462.583 (Observer identity is history identity)**.** *At the kernel-facing surface, observer identity is history identity under* hsame *and licensed classifier comparison.*

Proof. The observer ledger has no independent subject parameter. Its only identity surface is the generated history trace and the comparisons licensed over that trace. □

9462.22.3 Observer as history

Definition 9462.584 (Self-reference closure)**.** *The kernel is self-reference closed when every continuation relates history values,* hsame *is the kernel identity criterion for those values, and no observer-substance identity is defined alongside* hsame*.*

Definition 9462.585 (Observer)**.** *An observer in the BEDC object layer is a history equipped with its recorded distinction trace and admissible continuation and history-sameness comparisons. Since these structures are already history-layer structures, the compressed reading is observer* = Hist*, modulo* hsame*.*

Theorem 9462.586 (History-sameness is observer identity)**.** *If observers are read as self-distinguishing histories, then two observers have the same kernel identity precisely when their generated distinction traces are related by* hsame*.*

Proof. The generation rules for hsame preserve constructor shape: empty history compares only with empty history, E0-extensions compare through E0-extensions, and E1-extensions compare through E1-extensions. Thus a history-sameness witness preserves the finite distinction trace shape. Under the observer-as-history reading, that trace shape is the observer's identity surface. □

Theorem 9462.587 (No subject parameter)**.** *The kernel-level continuation signature contains no auxiliary parameter representing a subject or a carrier outside the history layer.*

Proof. Adding a parameter s : Subject to continuation would require that parameter to be transported or preserved by associativity and unit laws for continuation. The kernel supplies no witness for such transport, and the existing laws have no position in which an independent subject-substance can be consumed. The added parameter therefore does not refine the kernel; it changes the structure. □

Corollary 9462.588 (Observer is history)**.** *In BEDC, modulo* hsame*, the observer is the history itself.*

Proof. There is no independent subject parameter in continuation, and the identity surface available for observers is generated history-sameness. The only carrier left for observer identity is therefore Hist. □

Corollary 9462.589 (The self is a trace reading)**.** *The self is not an entity behind history, but the continuous reading of a history trace through* hsame *and its ledgers.*

Proof. If the observer is a history, persistence of the self can only mean persistence of the trace under licensed history comparison. A separate self-entity would require an additional carrier and identity transport not provided by the kernel. □

Corollary 9462.590 (No substance behind the trace)**.** *BEDC does not need to adopt any external doctrine of self-negation in order to obtain the structural conclusion that no observer-substance stands behind the trace.*

Proof. The conclusion follows from the kernel signature itself. The continuation relation has history endpoints, the identity criterion is history-sameness, and adding an independent observer carrier would require unlicensed transport. The result is structural, not an imported psychological or theological thesis. □

9462.22.4 Time as history growth

Definition 9462.591 (Hist time step)**.** *A Hist time step is one application of* E0 *or* E1 *to a history. The step is recorded both in the generated history value, by constructor shape, and in the continuation witness, by the displayed distinction argument.*

Definition 9462.592 (Local time stream)**.** *The local time stream of a history h is the finite sequence of continuation applications producing h from* e*, equivalently the generated unary history together with the displayed distinction record at each step.*

Theorem 9462.593 (Time is history growth)**.** *In BEDC, time is history growth.*

Proof. Time minimally requires an order of before and after. In the kernel, before and after first appear as constructor growth from a history to an E0- or E1-extension, and as the corresponding continuation witness. No independent time carrier is needed to produce this order. □

Corollary 9462.594 (No external time axis)**.** *The kernel has no time axis independent of histories.*

Proof. The available time-step surface is the generated history step and its continuation witness. An external axis would be a further primitive carrier, not part of the kernel signature. □

Corollary 9462.595 (Time is not an event container)**.** *Events do not first occur inside time and then become recorded; the temporal order is the order in which distinctions enter histories.*

Proof. An E0- or E1-application is simultaneously a distinction record and a time step. Without such record formation, the object layer has no kernel time stream to read. □

9462.22.5 The arrow of time

Definition 9462.596 (Forward continuation)**.** *A forward continuation is a witness* $\mathsf{Cont}(h, d, h')$ *read from an antecedent history h and distinction data d to the successor history h'.*

Definition 9462.597 (Backward recovery)**.** *Backward recovery is the problem of reconstructing an antecedent history h from a displayed successor h' and distinction data d such that* $\mathsf{Cont}(h, d, h')$ *holds.*

Theorem 9462.598 (Continuation is time-asymmetric)**.** *Continuation is constructive in the forward direction, while backward recovery requires structural analysis of the successor history.*

Proof. Given an antecedent history and a displayed distinction step, the successor is fixed by the relevant history constructor. Given a successor, recovering a predecessor requires inspecting the successor's generated form. This difference between construction and readback is the kernel-level arrow of time. □

Corollary 9462.599 (The time arrow is not an added physical postulate)**.** *The BEDC arrow of time follows from constructive directionality rather than from an additional physical axiom.*

Proof. The asymmetry is already present in the formation and recovery of history constructors. A separate primitive saying that time has a direction would duplicate the constructive structure. □

Corollary 9462.600 (Past as readback, future as construction)**.** *The past is structurally readable from generated histories; the future is continuation not yet generated.*

Proof. A generated history can be analyzed through its constructors. A successor not yet present must be produced by a new continuation. The distinction is between readback of a trace and construction of a trace. □

9462.22.6 No global clock

Definition 9462.601 (Global clock)**.** *A global clock would be a kernel-level history or equivalent structure that synchronizes the local time streams of arbitrary histories.*

Theorem 9462.602 (No kernel global clock)**.** *The BEDC kernel has no global clock synchronizing distinct local history streams.*

Proof. Each history carries its own generated time stream. The kernel supplies history constructors, continuation witnesses, and internal history-sameness, but no separate synchronizing carrier for arbitrary histories. Any synchronization relation must therefore be introduced as an extension with its own witnesses and certificate discipline. □

Corollary 9462.603 (No absolute present)**.** *There is no kernel-level absolute present shared by all histories.*

Proof. An absolute present would require a global synchronization of local time streams. Such a synchronization is not present in the kernel. □

Corollary 9462.604 (Cross-history time comparison must be certified)**.** *Any comparison between the time streams of distinct histories must be given by an explicit extension carrier, classifier, ledger, or naming certificate.*

Proof. Since the kernel does not define inter-history time comparison, such a comparison cannot be read directly. To enter the object layer, it must be displayed as additional data and audited as such. □

Concretized in §2171: Local time fibers realize observer-local Hist growth as finite source, Contstep, transport, refusal, provenance, and NameCert rows.

9462.22.7 Multiple observers and no God's-eye view

Definition 9462.605 (Multi-history configuration)**.** *A multi-history configuration is a finite family of histories considered together without assuming a shared base history, shared step indexing, or global synchronization.*

Definition 9462.606 (Inter-history locality)**.** *An inter-history relation is local when each positive witness factors through finite BEDC data: source histories, continuation or bundle witnesses, and classifier comparisons using* hsame*,* msame*, or licensed derived classifiers.*

Theorem 9462.607 (Inter-history relations must be ledger-visible)**.** *An acceptable inter-history relation must be auditable by finite BEDC ledger data.*

Proof. A relation between histories has object-layer standing only through generated marks, emitted histories, continuation witnesses, bundles, signatures, or licensed classifiers. A comparison supplied by an external observer without such rows is not a BEDC witness. □

Corollary 9462.608 (No observer outside all observers)**.** *BEDC contains no super-observer standing outside all histories and synchronizing them from without.*

Proof. If the alleged super-observer is not a history, it violates the observer-as-history reading. If it is a history, it has its own local time stream and lacks a kernel global clock. Adding further observers therefore does not repair the absence of a God's-eye frame. □

Corollary 9462.609 (Frame-independent content is inter-history invariant)**.** *Without a global clock, frame-independent content can only be carried by certified inter-history invariants.*

Proof. Single histories carry local time streams. Distinct histories do not share a kernel index. A cross-observer claim can therefore be stable only when it is expressed as a symmetric, history-sameness-respecting, ledgered relation or invariant. □

9462.22.8 Observation, distinction, and time

Definition 9462.610 (Three act-shapes)**.** *The kernel admits three readings of the same act-shape: observation as* $\mathsf{Cont}(h, d, h')$*, distinction as an* $\mathsf{E0}$*- or* $\mathsf{E1}$*-step, and time as a Hist time step.*

Theorem 9462.611 (Observation-distinction-time identity)**.** *At the kernel-facing surface, observation, distinction, and time are extensionally the same act-shape modulo* hsame *and the continuation closure laws.*

Proof. All three readings have the same domain of histories and distinction steps. The observation reading uses the continuation witness; the distinction reading uses the constructor step; the time reading uses the same step as local history growth. The relevant associativity, unit, and history-sameness-respect conditions are therefore conditions on the same generated structure. □

Corollary 9462.612 (Observation is not added after distinction)**.** *Observation is not an extra act placed on top of distinction and time.*

Proof. The continuation act that records a distinction is already the time step and the observation witness. No additional psychological layer is needed at the kernel surface. □

Corollary 9462.613 (Time is not outside observation)**.** *Time is not a background in which observation occurs; it is the generated order of observation as continuation.*

Proof. The time step and the observation act share the same continuation structure. An external temporal background would require a separate primitive not present in the kernel. □

Corollary 9462.614 (Distinction is dynamic)**.** *Distinction is not merely a static opposition; it is a generative act capable of becoming observation and time.*

Proof. An $\mathsf{E0}$- or $\mathsf{E1}$-application enters a history, supplies a continuation witness, and forms a local time step. Its role is therefore dynamic. □

9462.22.9 Continuation as universal verb

Definition 9462.615 (Universal verb)**.** Cont *is a universal verb when extension-type phenomena are read as specializations of the same continuation relation under different carrier, classifier, stability, ledger, and naming-certificate conditions.*

Theorem 9462.616 (Continuation is the universal verb)**.** Cont *is the universal verb of BEDC.*

Proof. Category morphisms, small-step computation, continuous moduli, type checking by classifier membership, and related extension phenomena all use the same pattern: a displayed source, a continuation or transition, a target, and certificate conditions preserving admissibility. Their differences lie in the surrounding NameCert fields, not in a new kernel verb. □

Corollary 9462.617 (Mathematical structure is stabilized continuation)**.** *Many mathematical structures in BEDC are stable certified readings of continuation rather than independent primitives.*

Proof. The carrier and classifier decide which continuation rows count as the phenomenon in question; the stability and ledger fields decide how the rows persist under reuse, composition, and transport. The kernel action remains continuation. □

Corollary 9462.618 (Action, observation, computation, and time cohere)**.** *Action, observation, computation, and time are four disciplined readings of generated continuation.*

Proof. Each reading concerns a finite passage from one displayed state to another. Action reads the passage as generated transition, observation as recorded distinction, computation as small-step relation, and time as history growth. □

9462.22.10 Synthesis

Theorem 9462.619 (Observer-observation-time synthesis)**.** *At the kernel-facing surface of BEDC,*

$$\textit{observer} = \mathsf{Hist}, \qquad \textit{observation} = \mathsf{Cont}, \qquad \textit{time} = \textit{history growth},$$
$$\textit{and} \qquad \textit{observation} = \textit{distinction} = \textit{time}$$

as structural readings modulo hsame *and the continuation closure laws.*

Proof. Observation is continuation between histories. Observer identity is carried by history-sameness and licensed classifier comparison. No subject parameter is present in the continuation signature. A time step is an E0- or E1-extension recorded by continuation. Continuation is asymmetric between construction and readback. Finally, the observation act, distinction step, and time step are readings of the same generated history transition. Combining these facts gives the claimed synthesis. □

Corollary 9462.620 (Excluded subject primitives)**.** *The kernel does not admit bare primitives such as* Subject*,* ObserverSubstance*,* SelfEntity*,* GlobalViewpoint*, or* AbsoluteNow*.*

Proof. The work assigned to those names is already carried by histories, history-sameness, continuation ledgers, inter-history relations, naming certificates, or exposed boundary markers. As bare primitives they would require additional carriers and transports absent from the kernel. □

Corollary 9462.621 (BEDC phenomenological form)**.** *The BEDC phenomenological form is not that a subject intends an object in time, but that a history continues by distinction, and that continuation is observation-time.*

Proof. The same continuation event can be read as distinction occurring, observation occurring, a time step occurring, and history growing. The four descriptions name one generated structure. □

9462.23 Objects, Knowledge, and Naming Certificates

The first layer reads distinction as the condition of history. The second layer reads continuation as observation, time, and observer identity. The present layer asks how stable patterns in histories, continuations, signatures, packages, and gaps become objects and knowledge. Its compressed thesis is:

$$\text{object} = \text{licensed name},$$
$$\text{knowledge} = \text{auditable witness},$$
$$\text{definition} = \text{certificate for stable generated behavior.}$$

Thus BEDC does not begin with mature objects and then describe them. It generates histories, classifies stable behavior, records the hidden material in ledgers, and licenses names only after the relevant certificate fields are supplied.

9462.23.1 Objects are not primitive

Definition 9462.622 (Bare object)**.** *A name N is a bare object name when it is introduced into the theory without a displayed source specification, pattern specification, classifier specification, stability certificate, and ledger policy. A name such as natural number, real number, set, type, function, limit, or circle is bare in this sense whenever it is not accompanied by a certificate explaining how it is generated from BEDC materials and how its behavior is classified, stabilized, and tracked.*

Definition 9462.623 (Mature object)**.** *A mature object is a licensed derived interface. For a proposed name N, maturity means that N is accepted through a naming certificate rather than supplied as a kernel primitive.*

Theorem 9462.624 (Anti-primitive principle)**.** *No mature object enters BEDC as a kernel primitive.*

Proof. The kernel begins with marks, histories, internal sameness judgments, continuations, asking events, signatures, packages, gaps, and naming certificates. Traditional mathematical interfaces such as equality, functions, natural numbers, arithmetic operations, limits, real numbers, geometric figures, and folds occur only after the relevant carrier, classifier, stability, and ledger obligations have been displayed. They are outputs of the certificate discipline, not inputs to the finite kernel. □

Corollary 9462.625 (Definitions do not create objects ex nihilo)**.** *A definition does not produce an object from nothing; it names stable generated behavior under a certificate.*

Proof. Without a source specification, the proposed name has no generated material. Without a pattern specification, it has no stable behavior. Without a classifier, it has no internal same-object condition. Without a stability certificate, it cannot be reused, descended, composed, or sealed. Without a ledger policy, its hidden material is not auditable. The definition becomes object-forming only when these obligations are met. □

Corollary 9462.626 (Object existence is certificate existence)**.** *For a mature BEDC object, object-layer existence entails the presence of an auditable naming certificate.*

Proof. A mature object is a licensed name, and a licensed name is not a primitive input. Its object-layer standing is therefore mediated by the certificate that licenses it. □

9462.23.2 The five fields of a naming certificate

Definition 9462.627 (Source specification)**.** *The source specification of a proposed name N records the generated material from which the name receives its license. Its sources may include raw marks, histories, bundles, signatures, packages, gaps, or previously licensed names.*

Definition 9462.628 (Pattern specification)**.** *The pattern specification records the stable behavior that the proposed name is meant to expose: repetition, continuation, signature classification, package stability, compatible threading, or another displayed regularity in the source material.*

Definition 9462.629 (Classifier specification)**.** *The classifier specification states how source material is grouped under the proposed name. It answers when two source rows count as the same N-object, when two behaviors instantiate the same pattern, and when two representatives are the same only through a licensed classifier.*

Definition 9462.630 (Stability certificate)**.** *The stability certificate records the closure, reuse, descent, composition, or sealing behavior that makes the name stable enough for later use. Different interfaces may require different combinations of these stability modes.*

Definition 9462.631 (Ledger policy)**.** *The ledger policy records what the name hides and how the hidden material remains traceable after compression, packaging, projection, or sealing. It specifies the relevant gap ledger, composite gap, tail ledger, phase ledger, source ledger, or other derived ledger surface.*

Definition 9462.632 (Naming certificate)**.** *A paper-level naming certificate for a proposed interface N is the five-field packet*

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The name N is licensed only when the required fields of this packet are supplied.

Theorem 9462.633 (The five fields are irreducible)**.** *No field of a naming certificate can be omitted without losing the object's auditable standing.*

Proof. Without source, the name has no generated material. Without pattern, the name has no stable behavior to expose. Without classifier, there is no internal sameness condition for the named interface. Without stability, the name cannot be safely reused or transported. Without a ledger policy, compression loses provenance. The five fields therefore correspond to source, pattern, sameness, stability, and memory. □

Corollary 9462.634 (Objects are fivefold qualifications)**.** *A BEDC object is not a single bare entity but a source-qualified, pattern-stabilized, classifier-organized, stability-certified, and ledger-tracked interface.*

Proof. The naming certificate licenses the object exactly through those five fields, and the fields are not optional. □

9462.23.3 Paper certificates and finite-kernel certificates

Definition 9462.635 (Paper naming certificate)**.** *A paper naming certificate is the five-field theoretical template that explains why a proposed interface may be named.*

Definition 9462.636 (Finite-kernel naming certificate)**.** *A finite-kernel certificate of the form* NameCert(Carrier, Equiv) *is a carrier-and-equivalence certificate over generated histories or their derived carriers. It supplies inhabitedness of the carrier, reflexivity, symmetry and transitivity of the equivalence relation, and transport of carrier membership along that relation.*

Theorem 9462.637 (Kernel certificates formalize the classifier core)**.** *The finite-kernel certificate is the formal core of the paper* NameCert *at the carrier and classifier layer.*

Proof. The paper certificate requires a classifier specification and stability conditions explaining how a carrier is organized and transported. The finite-kernel certificate supplies exactly the formal equivalence laws and membership transport needed for that organization. It does not exhaust every paper-level semantic field, but it gives the formal carrier-classifier nucleus of the licensed interface. □

Corollary 9462.638 (Paper completion is not machine checking)**.** *A complete paper certificate is not the same status as a machine-checked certificate.*

Proof. The paper certificate states and argues for the required fields. A machine-checked certificate requires corresponding formal targets and auditable acceptance by the checker. The former is a paper obligation surface; the latter is a verified formal object. □

9462.23.4 Knowledge as witness

Definition 9462.639 (BEDC knowledge)**.** *To know an object or claim X in BEDC is to possess a witness for X. Such a witness may be an* msame *or* hsame *witness, a* Cont *witness, a signature witness, a package witness, a gap witness, a naming-certificate witness, a semantic-certificate witness, or a derivation-certificate witness.*

Definition 9462.640 (Auditable knowledge)**.** *A knowledge item is auditable when its source is visible or ledger-tracked, its classifier is stated, its stability or transport law is supplied, its hidden material is recorded by a ledger, and its proof strength is marked.*

Theorem 9462.641 (Knowledge is witness, not bare truth)**.** *The BEDC form of knowledge is witness possession, not an internal bare truth predicate.*

Proof. Object knowledge requires a naming certificate. Classification knowledge requires a classifier witness. Compressed knowledge requires a ledger exposing the source material behind the token. Formal knowledge requires a target accepted by the relevant audit. A claim without a witness has no public object-layer standing, because the object layer has no global truth predicate through which it could be consumed. □

Corollary 9462.642 (Truth remains meta, certificates enter the object layer)**.** *BEDC object-layer operations consume certificates rather than a total truth predicate.*

Proof. A total internal truth predicate would classify arbitrary propositions at once. BEDC instead licenses local interfaces through source, classifier, stability, and ledger witnesses. The object layer operates with those certificates. □

9462.23.5 Packaging, compression, and gaps

Definition 9462.643 (Package token)**.** *A package judgment* $\mathsf{Pkg}(\Pi, s, p)$ *says that, under bundle* Π*, the answer-history* s *is represented by the package token* p*. The relation* $\mathsf{psame}(p, q)$ *is the internal sameness judgment for package tokens.*

Definition 9462.644 (Package policy)**.** *A package policy supplies existence of tokens for generated answer-histories, extensionality of tokens along history-sameness, grounding of package-token sameness in represented histories, and the visibility rule that only the token and its ledger appear at the package interface.*

Theorem 9462.645 (Packages are not quotients)**.** *Package tokens are not quotient classes. Quotient-like behavior is a certified theorem of the package policy, not a primitive.*

Proof. If package tokens were quotient classes, the theory would need a quotient object or a quotient-soundness principle at the point of introduction. Instead, tokens are introduced by package judgments, and their sameness is controlled by extensionality and grounding fields. The visible quotient-like behavior is therefore derived from package policy rather than imported as a primitive quotient. □

Definition 9462.646 (Gap ledger)**.** *A gap ledger records which source histories lie behind a visible package token, signature token, or compressed interface. It is the memory surface of a compression.*

Definition 9462.647 (Gap policy)**.** *A gap policy exposes provenance and generation witnesses, covers the admitted source histories by visible gaps, and separates package tokens when the same source lies behind their gaps.*

Theorem 9462.648 (Compression must remember)**.** *Every BEDC operation that packages, compresses, projects, seals, or hides generated material must carry a gap ledger.*

Proof. Packaging replaces source histories by visible tokens. Without a gap ledger, the theory loses the internal witness connecting the token to the histories it hides. The token would then be a free object rather than a certified compression. A valid compression must therefore retain provenance through a ledger. □

Corollary 9462.649 (No amnesic compression)**.** *BEDC permits compression, but not compression that forgets its source.*

Proof. Compression is acceptable only when the visible token remains connected to its hidden source material by a gap policy. Removing that memory breaks auditability. □

9462.23.6 Semantic naming and pattern-ledger transport

Definition 9462.650 (Semantic naming certificate)**.** *A semantic naming certificate is a naming certificate enriched by source-to-pattern soundness, source-to-ledger soundness, and classifier transport preserving the pattern and ledger fields.*

Theorem 9462.651 (Semantic certificates contain pattern and ledger witnesses)**.** *Every semantic naming certificate contains a source witness from which both the pattern condition and the ledger policy are obtained.*

Proof. Eliminating the semantic certificate exposes its core naming certificate and its pattern and ledger soundness fields. The core certificate supplies the source witness, and the soundness fields apply to that same witness. □

Theorem 9462.652 (Pattern-ledger transport)**.** *If a semantic naming certificate classifies a source history h with a history k, then the pattern and ledger conditions transport to k.*

Proof. The core naming certificate transports carrier membership along the classifier relation. Once the target is again recognized as a source row, the semantic pattern and ledger soundness fields apply at the target. □

Theorem 9462.653 (Classifier-chain transport)**.** *If a semantic certificate supplies classifier links from h to k and from k to r, and h carries the source witness, then the pattern and ledger conditions transport to r.*

Proof. Transitivity of the core classifier composes the two links into a single link from h to r. Pattern-ledger transport then applies to that composite link. □

Corollary 9462.654 (Object stability is transportability)**.** *The stability of a BEDC object is not metaphysical immobility but transportability of its pattern and ledger across licensed classifier relations.*

Proof. The object remains stable precisely when the certificate can carry its pattern and ledger through the equivalence or classifier paths that the interface licenses. □

9462.23.7 Certificate-organized clusters

Definition 9462.655 (Certificate-organized cluster)**.** *A certificate-organized cluster is a paper-level interface*

$$\mathsf{ClusterCert}(S, P, L, R, B, F, O)$$

whose first component is a semantic naming certificate over source rows S, pattern rows P, ledger rows L, and classifier relation R. The remaining rows record, respectively, a connectivity or membership witness for the cluster, a boundary predicate B, forbidden claims F, and open obligations O. The interface is a derived certificate protocol; it is not a kernel primitive and not an axiom schema.

Theorem 9462.656 (Cluster names are licensed names)**.** *If a certificate-organized cluster is accepted, then its public name is a licensed BEDC name rather than a bare cluster label.*

Proof. The cluster interface contains a semantic naming certificate. By Definition 9462.651, that certificate supplies a source witness together with pattern and ledger witnesses. The classifier core supplies transport across the cluster by Definition 9462.652 and Definition 9462.653. Therefore the cluster name is admitted through source, pattern, classifier, stability, and ledger data, not through a detached label. □

Corollary 9462.657 (Cluster objects are transportable source families)**.** *A certificate-organized cluster names a transportable source family under a displayed classifier, not a single unqualified point.*

Proof. Pattern and ledger rows transport along licensed classifier paths by Definition 9462.652 and Definition 9462.653. The named object is therefore the classifier-stable family of admissible source rows together with its ledger, boundary, forbidden-claim, and open-obligation surface. □

Corollary 9462.658 (Cluster compression preserves memory)**.** *Any readout, coordinate, probability, or model token exported from a certificate-organized cluster must preserve its source memory through a ledger or expose the missing source as a boundary.*

Proof. The cluster interface contains the ledger row of its semantic certificate and the explicit boundary row B. A readout that hides source material is a compression, so Definition 9462.648 applies. If the source fiber cannot be fully displayed, the remaining material belongs to the boundary or gap surface rather than to a closed object-layer claim. □

9462.23.8 Acceptance gates and strength

Definition 9462.659 (Derivation strength)**.** *The strength of a derived object is ordered as*

$$\mathsf{seed} < \mathsf{paperCert} < \mathsf{checkedCert} < \mathsf{bridgeCert}.$$

A seed has a sketched carrier and classifier. A paper certificate fills the certificate fields in the manuscript. A checked certificate is backed by formal targets and audit. A bridge certificate adds a conservative bridge to a standard model.

Definition 9462.660 (Acceptance gate)**.** *For a mature object name N at strength s, the acceptance gate is the conjunction of a naming certificate for N and a derivation certificate recording that strength.*

Definition 9462.661 (Derivation certificate)**.** *A derivation certificate records the source layer carrying N, the classifier organizing the carrier, exactness of the classifier with respect to the relevant sameness judgments, the ledger or explicit absence statement, the stability certificate, and the declared strength.*

Theorem 9462.662 (Object export passes through the acceptance gate)**.** *BEDC may export a mature object N only at a displayed strength whose acceptance gate is satisfied.*

Proof. The naming certificate explains why the name is licensed. The derivation certificate explains which strength level has been reached. Without the first, N is not a licensed name. Without the second, the theory cannot distinguish a sketch from a paper certificate, a checked certificate, or a standard-model bridge. □

Corollary 9462.663 (No strength inflation)**.** *BEDC cannot report a lower strength as a higher one.*

Proof. The strength order is cumulative in one direction only. A checked certificate includes more obligations than a paper certificate, and a bridge certificate includes more obligations than a checked certificate. Reporting a lower level as a higher one hides unfinished certificate obligations. □

9462.23.9 Synthesis

Theorem 9462.664 (Object-knowledge-certificate synthesis)**.** *In BEDC, objects and knowledge enter through certificates:*

$$\mathit{object}_N = \mathit{licensed\ name}(N),$$
$$\mathit{knowledge} = \mathit{auditable\ witness}.$$

The mature object is generated from histories, continuations, signatures, packages, gaps, and prior names through source, pattern, classifier, stability, and ledger fields, then exported only at a declared strength. The knowledge claim appears through a displayed witness rather than through a bare truth predicate.

Proof. Mature objects are not primitives. A naming certificate has five irreducible fields. Finite-kernel certificates formalize the carrier-classifier core. Knowledge is witness possession. Compression requires a gap ledger. Semantic certificates transport pattern and ledger along classifiers. Export requires an acceptance gate with a declared strength. Combining these facts gives the stated synthesis. □

Corollary 9462.665 (BEDC object theory)**.** *The BEDC theory of objects can be summarized as follows: an object is a licensed name, a name is a certificate for stable behavior, and a certificate is the fivefold structure of source, pattern, classifier, stability, and ledger.*

Corollary 9462.666 (BEDC epistemology)**.** *The BEDC theory of knowledge can be summarized as follows: to know is to hold a witness, to prove is to supply transport, stability, and ledger data, and to understand an object is to identify the certificate through which it enters.*

Corollary 9462.667 (Passage to logic and meta-loop discipline)**.** *Once objects and knowledge are certificate-mediated, the next task is to identify which proof patterns preserve this discipline and which patterns smuggle in unwitnessed objects or global truth.*

Proof. Certificate discipline accepts local contradiction over fixed carriers, constructor no-confusion, and witness-preserving transport. It rejects unwitnessed positive existence, hidden choice, quotient collapse, propositional extensional collapse, and total truth predicates. The next philosophical layer must therefore analyze proof and meta-loop boundaries. □

Concretized in §1218: Object-knowledge certificates realize licensed object names and auditable witnesses as one finite NameCert packet.

Concretized in §1341: Bare-object refusal is realized as a finite NameCert packet recording missing certificate fields, refusal rows, witness audit, and anti-primitive ledger nonescape.

9462.24 Section Meaning Certificates

section 9462.23 identifies objects with licensed names and knowledge with auditable witnesses. The present layer applies that discipline to interfaces in which an observation is invariant under a displayed symmetry while the semantic label changes along a fiber. Its compressed thesis is:

observation gives a package-level quotient readout,
meaning requires a certified section or fiber ledger.

Thus a BEDC interface may accept an orbit-level observation without accepting a bare representative selector, a hidden canonical section, or a label that descends through the same package token.

Definition 9462.668 (Orbit observation packet)**.** *An orbit observation packet consists of a source carrier X, an internal classifier $\approx_X$, a licensed action certificate for a symmetry interface G, an orbit package Q_G with projection $\pi_G : X \to Q_G$, and a readout $O : X \to Y$ satisfying the displayed invariance row*

$$O(g \cdot x) \approx_Y O(x).$$

The package Q_G is a BEDC package interface with a gap ledger, not a primitive quotient type.

Definition 9462.669 (Anti-invariant label packet)**.** *An anti-invariant label packet over the same source consists of a two-label interface* $\mathsf{B}^{\uparrow} = \{C_0, C_1\}$ *with no-confusion, an involution* ν *exchanging the two labels, a certified involution* $\tau \in G$*, and a label readout* $L : X \to \mathsf{B}^{\uparrow}$ *satisfying*

$$L(\tau \cdot x) = \nu(L(x)).$$

Definition 9462.670 (Section meaning certificate)**.** *A section meaning certificate is the derived paper-level interface*

$$\mathsf{SectionMeaningCert}(X, G, O, L, Q_G, \pi_G, \sigma)$$

whose rows include an orbit observation packet, an anti-invariant label packet, an optional section binding $\sigma : Q_G \to X$ *satisfying* $\pi_G(\sigma(q)) = q$*, and ledgers for the source, orbit package, observation, label, section authorization, and boundary. The section row is valid only when it records its source, authorization, witness, stability, and forbidden-claim surface.*

Theorem 9462.671 (Observation descends while labels need not)**.** *In a section meaning certificate, the invariant observation descends to a licensed readout* $\bar{O} : Q_G \to Y$*, while an anti-invariant label cannot be treated as a readout on* Q_G *unless an additional section or fiber ledger is displayed.*

Proof. If two source rows have the same orbit package token, the action certificate supplies a symmetry path between them. Observation invariance transports O along that path, so O factors through the package projection as $\bar{O} \circ \pi_G$. For the label row, x and $\tau \cdot x$ have the same orbit token, but their labels are exchanged by ν. A package-level label readout would identify those two labels, contradicting no-confusion. Therefore the label requires additional section or fiber data rather than descending through the orbit package alone. □

Corollary 9462.672 (No hidden section from pure observation)**.** *Pure orbit-observation material does not license a distinguished section* $Q_G \to X$*.*

Proof. The visible observation rows are invariant on each orbit. Choosing a representative inside an orbit therefore introduces data not contained in the orbit token or invariant readout. BEDC may expose that data as a section binding, authorization row, action-response row, bridge, or boundary, but it may not present the choice as a theorem of the pure observation packet. □

Theorem 9462.673 (Meaning is section-relative)**.** *When a valid section binding* σ *is supplied, the expression*

$$L_\sigma(q) = L(\sigma(q))$$

is a licensed meaning readout on Q_G *relative to that binding.*

Proof. The section binding is a licensed map from the package interface back to the source carrier, and the label packet is a licensed map from the source carrier to the two-label interface. Their composite is therefore a licensed readout. Its ledger is the composite ledger of the section authorization and the label packet, so the resulting meaning is relative to the displayed binding rather than intrinsic to the orbit token. □

Corollary 9462.674 (Matched twins force a blind ceiling)**.** *In a balanced matched-twin protocol where each pair has the same observation and opposite labels, every decision rule depending only on the orbit observation is bounded by the blind ceiling* $\frac{1}{2}$*.*

Proof. On each matched pair, the rule sees the same observation on both members and therefore returns the same decision. The label packet assigns opposite labels to the two members. With balanced presentation, the rule is correct on exactly one side of each pair, so averaging over the finite protocol gives $\frac{1}{2}$. □

Corollary 9462.675 (Section access, leakage, or protocol mismatch)**.** *If a matched-twin interface is reported above the blind ceiling, then the certificate audit must expose section access, leakage of non-orbit data, or a mismatch in the stated protocol object.*

Proof. The blind ceiling applies to every orbit-observation rule. Exceeding it therefore means that the rule is not using only the certified orbit observation, or that the protocol does not satisfy the matched-twin rows that define the ceiling. In the first case the additional coordinate must be ledgered as section access or recorded as leakage; in the second case the protocol object must be corrected before the reported number can support a certificate claim. □

9462.25 Logic, Contradiction, and the Meta Loop

The previous layer identifies objects with licensed names and knowledge with auditable witnesses. The logical question is therefore not which rules are attractive in isolation, but which proof patterns preserve witness, classifier, stability, and ledger discipline. The central distinction is:

$$\text{closed ground loop} = \text{constructive formation and local refutation,}$$
$$\text{open meta loop} = \text{no total truth, choice, quotient, or halting closure.}$$

9462.25.1 The BEDC form of proof

Definition 9462.676 (BEDC proof)**.** *A BEDC proof of a claim A is a finite witness chain using generated materials such as marks, histories, internal sameness judgments, extension and continuation rows, signatures, packages, gaps, and naming certificates. It must display its sources, continuation steps, classifier comparisons, hidden material, ledger rows, and certificate surface.*

Concretized in §1238: Proof witness chains are realized as finite BHist packets binding proof trace, source carrier, classifier witness, transport, ledger, forbidden-resource refusal, audit verdict, provenance, and local NameCert rows.

Concretized in §1353: Proof-pattern classifiers are realized as finite BHist packets separating positive witness construction, local refutation, decidable double-negation, and prohibited positive extraction.

Definition 9462.677 (Positive proof)**.** *A positive proof constructs a witness for an object, proposition, relation, or certificate field. It does not merely show that the absence of such a witness would be contradictory.*

Definition 9462.678 (Negative proof)**.** *A negative proof transforms any purported witness of A into $\bot$, thereby proving $\neg A$. The conclusion means that every internal witness of A fails inside the displayed carrier, classifier, or certificate; it is not a God's-eye denial over all possible structures.*

Theorem 9462.679 (Proof is witness, not truth-possession)**.** *In BEDC, the basic proof form is an auditable witness chain rather than possession of bare truth.*

Proof. Object-layer contents enter through generated carriers and licensed interfaces. A claim that lacks source rows, continuation witnesses, classifier transport, gap provenance, or certificate fields cannot be consumed by the object layer. Thus a proof must expose the chain by which the claim is generated and stabilized. □

Corollary 9462.680 (Logic follows certificate discipline)**.** *BEDC logic is the proof-theoretic face of certificate discipline.*

Proof. Objects enter through naming certificates and knowledge enters through witnesses. A rule that creates positive content without a witness would bypass that entry discipline. Hence admissible reasoning must preserve witnesses, classifiers, stability, and ledgers. □

9462.25.2 Permitted contradiction

Definition 9462.681 (Permitted refutation)**.** *A refutation is permitted when it has the form $A \to \bot$, and the derivation of $\bot$ uses only fixed carrier data, constructor no-confusion, internal sameness, continuation rows, fixed naming certificate laws, and classifier inversion. It does not use a total truth predicate, arbitrary excluded middle, unwitnessed choice, quotient collapse, or propositional extensionality.*

Theorem 9462.682 (Negative proof is accepted locally)**.** *If a proof only converts any witness of A into an internal contradiction, BEDC accepts it as a proof of $\neg A$.*

Proof. Such a proof creates no new object and supplies no unwitnessed positive existence. It shows that a purported witness conflicts with the fixed constructor, classifier, carrier, or certificate laws already in view. The proof remains inside the closed ground loop. □

Example 9462.683 (Mark no-confusion)**.** *The judgment $\neg\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ is a permitted refutation. Assuming a witness of $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$, inversion on the closed generation of msame leaves only the two diagonal cases and no cross-mark constructor. The contradiction uses the mark carrier and constructor no-confusion, not a total truth predicate.*

Definition 9462.684 (Decidable double-negation elimination)**.** *For a displayed decidability witness of $P(h)$, the passage from $\neg\neg P(h)$ to $P(h)$ is decidable double-negation elimination.*

Theorem 9462.685 (Double-negation elimination requires decidability)**.** *BEDC accepts $\mathsf{Decidable}(P(h)) \to \neg\neg P(h) \to P(h)$, but not $\neg\neg P(h) \to P(h)$ without a displayed decidability witness.*

Proof. With decidability, the case split is itself a witness. The positive case supplies $P(h)$. The negative case contradicts $\neg\neg P(h)$ and therefore also yields the positive case. Without decidability, $\neg\neg P(h)$ contains no constructor for $P(h)$; turning it into one would inject missing constructive content. □

Definition 9462.686 (Refuting false existence)**.** *To refute an existential claim is to give a uniform method converting any purported pair $(x, P(x))$ into a contradiction inside the displayed carrier or classifier.*

Theorem 9462.687 (False existence may be refuted)**.** *BEDC accepts a refutation of an existential claim when every purported existential witness can be converted into an internal contradiction.*

Proof. An existential witness, if present, must be usable as internal data. Mapping every such witness to $\bot$ proves that the witness type is uninhabited; it does not construct a positive object. □

Corollary 9462.688 (Negative proof is local)**.** *Negative proof in BEDC must be local to a fixed carrier, classifier, or certificate surface.*

Proof. A global negation over all propositions or all certificate extensions would require a meta-level truth classifier. BEDC permits local inversion and refutation, not total truth closure. □

9462.25.3 Prohibited contradiction

Definition 9462.689 (Prohibited positive extraction)**.** *A proof pattern is prohibited when it produces positive content without constructing the corresponding witness. Typical cases are extracting* $\exists h, P(h)$ *from* $\neg\forall h, \neg P(h)$ *without an exhibited* h*, using double-negation elimination without a decidability witness, or applying excluded middle to arbitrary certificate-level predicates.*

Theorem 9462.690 (Unwitnessed positive existence is meta closure)**.** *Deriving* $\exists h, P(h)$ *from* $\neg\forall h, \neg P(h)$ *without displaying* h *and a witness of* $P(h)$ *closes the meta loop.*

Proof. The existential claim is a witness obligation. If no witness is displayed, the system has nevertheless accepted a value that later constructions may consume. That value is an oracle rather than a generated object. □

Theorem 9462.691 (Undecided double-negation elimination is meta closure)**.** *Without* $\mathsf{Decidable}(P(h))$*, the rule* $\neg\neg P(h) \to P(h)$ *is not a BEDC rule.*

Proof. The conclusion $P(h)$ requires a constructor witness. The premise only says that denying such a witness leads to contradiction. Using it as the witness itself replaces construction by meta-level extraction. □

Theorem 9462.692 (Global excluded middle is meta closure)**.** *Excluded middle over arbitrary certificate-level predicates is a truth-totalizing operation and therefore closes the open meta loop.*

Proof. Such a principle would decide, for each certificate-level predicate, whether a witness exists or any witness is impossible. This is a fragment of an internal truth predicate over the certificate space, not a local classifier witness. □

Corollary 9462.693 (Existence requires a witness)**.** *Every positive existence theorem in BEDC must exhibit the object and the proof that it satisfies the stated predicate.*

Proof. An existential conclusion without witness is precisely the prohibited positive extraction described above. □

Corollary 9462.694 (Classical contradiction must be audited)**.** *Classical proofs by contradiction must be separated into refutational content and positive existence content before entering BEDC.*

Proof. The refutational part may survive as a local proof of negation. The positive existential part must either exhibit a witness or remain outside the certified fragment. □

9462.25.4 Closed ground loop and open meta loop

Definition 9462.695 (Closed ground loop)**.** *The closed ground loop is the mark-level distinction supplied by the two primitive mark constructors and their constructor no-confusion. It is closed because the declaration of the two constructors is already an act of distinction.*

Definition 9462.696 (Open meta loop)**.** *The open meta loop is the position that cannot be closed inside BEDC: an internal truth predicate, internal runtime validator, total halting predicate, or global decision predicate for all certificate-level propositions.*

Theorem 9462.697 (Ground must close)**.** *A formal system whose ground never closes cannot start.*

Proof. If the first primitive required a prior primitive to explain it, the regress would not terminate. BEDC pays this cost once, at the minimal mark distinction. □

Theorem 9462.698 (Meta must stay open)**.** *A sufficiently expressive system cannot internalize its own total truth predicate.*

Proof. Once a system can represent enough of its own syntax, a total truth predicate produces the familiar Tarski-style fixed-point obstruction. Truth recovery requires a stronger metalanguage rather than another object-layer predicate of the same system. □

Theorem 9462.699 (Two-loop balance)**.** *BEDC requires the ground loop to close and the meta loop to remain open.*

Proof. The ground loop closes at the minimal productive distinction, allowing generation to begin. The meta loop remains open because truth, runtime, and total decision cannot be absorbed into the object layer without self-reference collapse. The two requirements occupy different levels and therefore do not conflict. □

Corollary 9462.700 (BEDC exposes rather than evades the boundary)**.** *BEDC does not evade incompleteness or undefinability phenomena; it exposes the primitive distinction on which they depend.*

Proof. Syntax, formula, proof, and classifier distinctions already presuppose distinguishability. BEDC makes that condition explicit at the ground instead of pretending that it can be eliminated at the meta level. □

Corollary 9462.701 (Only the ground closes)**.** *The only loop that closes in BEDC is the form of distinction at the ground layer.*

Proof. Higher layers are built by classifier-based generation, transport, and certification. If a higher layer attempts to close over its own truth, runtime, or total decision procedure, it is no longer a productive ground loop but a meta closure. □

9462.25.5 Self-description and truth boundary

Definition 9462.702 (Structural self-description)**.** *Structural self-description is the construction, inside BEDC, of a certificate-mediated reproduction of formal features of the host calculus: type formation, dependent function formation, inductive type formation, recursors, and well-founded classifier closures.*

Definition 9462.703 (Runtime boundary)**.** *The runtime boundary is the fact that a structural reproduction of a calculus is not the host runtime that checks proofs. The proof checker remains a metalanguage artifact relative to the BEDC object layer.*

Theorem 9462.704 (BEDC may describe formal structure)**.** *BEDC may describe formal structure resembling the calculus in which it is checked.*

Proof. Such a description is a certificate-mediated account of formation rules, inductive shapes, and closure operations. It is structural data inside the theory, not the running checker itself. □

Theorem 9462.705 (BEDC does not internalize its runtime)**.** *BEDC cannot make the proof-checking runtime an object-layer component of itself.*

Proof. The external checker validates BEDC proofs from the metalanguage. A structural reproduction of checking rules does not become the implemented runtime that performs validation. No object layer contains the full runtime that interprets it. □

Theorem 9462.706 (Self-description is not truth definition)**.** *BEDC self-description is not a BEDC truth predicate.*

Proof. Describing syntax and closure rules does not decide which arbitrary object-layer claims are true. A total truth predicate would require the meta loop to close, which is excluded by the Tarski boundary. □

Corollary 9462.707 (Reflection is layered)**.** *BEDC reflection is layered structural reflection, not a collapse of the described layer into the describing layer.*

Proof. The described structure, the structural description, and the external checker occupy distinct roles. Collapsing them would recreate the truth-totalization problem. □

Concretized in §1260: Runtime-boundary audit packets realize the structural self-description/runtime-refusal split as finite BHist, hsame, Cont, Pkg, NameCert, and TasteGate rows.

9462.25.6 Three axioms, one meta closure

Definition 9462.708 (Meta-closure operation)**.** *At a syntactic stratum S, a meta-closure operation accepts a positive existential without a constructor witness and supplies a value at that stratum usable by later kernel constructions.*

Definition 9462.709 (Choice as term-level meta closure)**.** Classical.choice *is term-level meta closure when it turns pointwise existential information into a selector function whose body is not constructed from displayed witnesses.*

Definition 9462.710 (Quotient soundness as type-level meta closure)**.** *Quotient soundness is type-level meta closure when a relation or equivalence is consumed as quotient equality or representative transport without explicit classifier transport and ledger data.*

Definition 9462.711 (Propositional extensionality as proposition-level meta closure)**.** *Propositional extensionality is proposition-level meta closure when a bidirectional implication is consumed as equality of propositions without recording direction, transport, or ledger.*

Principle 9462.712 (The three classical axiom families share one taxonomy)**.** Classical.choice*, quotient soundness, and propositional extensionality are classified here as three meta-closure refusal shapes at the term, type or classifier, and proposition strata.*

Explanation. Each lets insufficiently witnessed positive content be consumed as a value by subsequent constructions. The difference is the stratum at which the value is requested: selector at the term layer, quotient-facing transport or equality at the type layer, and proposition identification at the proposition layer. This is a taxonomy for the audit ledger, not a metatheoretic proof that the host axioms are equivalent to one another or to BEDC collapse modes. □

Corollary 9462.713 (Refusal of the axioms is not refusal of their subject matter)**.** *BEDC refuses the unwitnessed shortcut, not the mathematical content that classical axioms often help express.*

Proof. The same content may enter through explicit witnesses, classifier transport, package sameness, and ledgers. What is rejected is the untracked conversion of existence, relation, or equivalence into a consumable object. □

9462.25.7 Witness-preserving replacements

Definition 9462.714 (Effective choice)**.** *Effective choice in BEDC is finite choice by primitive recursion over closed lists, countable choice with displayed witness functions, or dependent choice along a continuation chain with explicit step functions.*

Theorem 9462.715 (Effective choice preserves witnesses)**.** *The BEDC replacements for choice do not introduce unwitnessed selectors.*

Proof. Each selector is built from displayed data: recursion over a finite carrier, a given witness function, or a given step function along a continuation chain. The selector body is therefore constructed rather than supplied by meta closure. □

Definition 9462.716 (Classifier transport instead of quotient collapse)**.** *BEDC replaces quotient collapse by naming certificates, classifiers, package sameness, and classifier-respecting transport. Carrier elements identified by a classifier are not collapsed at the type level; operations consume explicit transport.*

Theorem 9462.717 (Classifier transport replaces quotient collapse)**.** *Quotient-like behavior in BEDC is implemented by explicit classifier transport rather than by primitive quotient equality.*

Proof. The operation that respects the classifier must display how it transports across the classifier relation. No canonical representative is selected without a witness, and no type-level equality is supplied without ledgered transport. □

Definition 9462.718 (Directional transport instead of propositional collapse)**.** *BEDC replaces propositional extensionality by explicit directional transport. If $P \to Q$ and $Q \to P$ are available, each construction records which direction it consumes instead of identifying P and Q as the same proposition.*

Theorem 9462.719 (Explicit transport replaces proposition collapse)**.** *BEDC does not replace $P \leftrightarrow Q$ by $P = Q$; it preserves direction, transport, and ledger.*

Proof. The bidirectional implication gives two maps. A consumer of P or Q must specify the map it uses. The propositions are not collapsed into a single object by an unwitnessed equality. □

Theorem 9462.720 (Certified content uses witness-preserving replacements)**.** *BEDC certified content is supported by explicit choice, classifier transport, and directional transport rather than by the three meta-closure axioms.*

Proof. Whenever a construction needs a selection, it must display the selection function or recursion. Whenever it needs quotient-like behavior, it must display classifier transport. Whenever it needs a proposition to be consumed under an equivalent form, it must display the direction of transport. □

9462.25.8 Axiom-purity

Definition 9462.721 (Axiom-pure theorem)**.** *A theorem is axiom-pure when its transitive dependencies do not include* Classical.choice*, quotient soundness, or propositional extensionality.*

Definition 9462.722 (Axiom-purity gate)**.** *The axiom-purity gate rejects kernel-level theorems whose dependency graph passes through the three axiom families classified as meta-closure refusal shapes.*

Principle 9462.723 (Axiom-purity rejects meta-closure shapes)**.** *The axiom-purity gate rejects dependencies classified as meta-closure shapes at the term, type or classifier, and proposition strata.*

Explanation. The rejected dependencies are read here as term-level, type-level or classifier-level, and proposition-level ways for unwitnessed positive content to be made usable by later constructions. Excluding them keeps the open meta loop open. □

Corollary 9462.724 (The gate is philosophical discipline)**.** *Axiom-purity is not only an engineering rule; it is the mechanical enforcement of witness discipline on proof dependencies.*

Proof. If no unwitnessed object may enter the object layer, then dependencies that supply hidden witnesses must be rejected. The gate implements that refusal at the dependency level. □

Concretized in §1259: Axiom-purity gate packets realize dependency-audit refusal as finite theorem-target, dependency-report, forbidden-row, replay, replacement, transport, provenance, and NameCert data.

Concretized in §1269: Meta-closure witness ledgers realize the axiom-purity vision as finite witness, classifier-transport, directional-transport, dependency-audit, hsame, Cont, Pkg, and NameCert rows.

Corollary 9462.725 (Purity is not full maturity)**.** *Passing axiom-purity does not by itself show that a mature object has passed its acceptance gate.*

Proof. Axiom-purity checks proof dependencies. Object maturity also requires source, classifier, exactness, ledger, stability, strength, and possibly bridge obligations. These are separate audit surfaces. □

9462.25.9 Halting as the recursive face of the open meta loop

Definition 9462.726 (Continuation-chain termination)**.** *A continuation chain for a naming certificate terminates from a history when there is a finite displayed list of continuation steps ending at a history with no successor under the same continuation relation.*

Definition 9462.727 (Halting predicate)**.** *A halting predicate for BEDC would decide, for arbitrary naming certificates and histories, whether the corresponding continuation chain terminates.*

Theorem 9462.728 (No total halting certificate)**.** *There is no BEDC naming certificate that decides termination for all naming certificates and all histories.*

Proof. Assume such a certificate exists. Build a diagonal certificate that consults the supposed halting certificate on a displayed certificate and initial history, then behaves oppositely: continuing when termination is predicted and stopping when nontermination is predicted. Applying the diagonal certificate to its own displayed code gives the standard contradiction. □

Theorem 9462.729 (Total halting closes the meta loop)**.** *A total halting predicate is a certificate-stratum closure of the open meta loop.*

Proof. It would be a global decision procedure for all continuation behavior of naming certificates. That is a total meta-level classifier imported into the object layer. □

Corollary 9462.730 (No total proof-search oracle)**.** *BEDC has no internal oracle deciding whether arbitrary certificate construction succeeds.*

Proof. Such an oracle would decide termination of the corresponding certificate-search continuations, contradicting the absence of a total halting certificate. □

Corollary 9462.731 (Open targets require exposed status)**.** *When a construction has no displayed witness, its status must remain exposed by a target, horizon, or gap marker rather than absorbed by a total truth or halting oracle.*

Proof. A total oracle would close the meta loop. Exposed status preserves the difference between a witnessed construction and an unwitnessed obligation. □

9462.25.10 Type checking as classifier membership

Definition 9462.732 (Extension as type membership)**.** *For a language-like carrier L, $\mathsf{Ext}_L(t)$ may be read as the predicate that t lies in the well-typed subclass of terms.*

Definition 9462.733 (Continuation as typed reduction)**.** $\mathsf{Cont}_L(t, m, t')$ *may be read as a displayed small-step reduction from t to t' with mark m, subject to preservation of the extension predicate.*

Theorem 9462.734 (Type checking is classifier membership)**.** *In the BEDC reading, type checking is membership checking for a classifier-defined extension predicate.*

Proof. The extension predicate cuts out the admissible subclass of terms, and continuation describes allowed transitions within that subclass. To check the type is to check membership in the displayed classifier surface. □

Definition 9462.735 (Curry–Howard reading)**.** *Under the BEDC Curry–Howard reading, a proposition is a classifier or type, a proof is a witness or inhabitant, and a program is a continuation-preserving construction.*

Theorem 9462.736 (Closure law as constructive proof)**.** *Every classifier closure law is also the constructive proof of the corresponding type closure.*

Proof. If Ext is read as membership in a type, then a closure law proves that the relevant construction sends admitted inputs to admitted outputs. The same witness is both the closure proof and the program transformer. □

Corollary 9462.737 (Proof checking is not proof search)**.** *Checking a displayed witness against a classifier is not the same as having a total proof-search procedure.*

Proof. Classifier membership can be local and decidable for the displayed interface. A total proof-search procedure would decide arbitrary certificate construction and would approach the prohibited halting predicate. □

9462.25.11 Synthesis

Theorem 9462.738 (Logic discipline synthesis)**.** *BEDC permits explicit witness construction and local refutation inside fixed carriers, classifiers, and certificates. It rejects positive content without constructor witnesses and rejects global truth, halting, excluded-middle, choice, quotient, or proposition-equality closures over certificate space.*

Proof. Proofs are witness chains. Local negation stays inside fixed generated structure. Unwitnessed positive extraction supplies a value without construction. Ground closure is needed for generation, while meta closure would internalize truth or runtime. The three classical axioms are term-, type-, and proposition-level instances of the same hidden witness schema; axiom-purity rejects them. A total halting predicate is the recursive version of the same closure. Type checking remains local classifier membership rather than total proof search. □

Corollary 9462.739 (Positive existence must be constructed)**.** *Every BEDC existence theorem must display the witness, its carrier membership, its classifier stability, and its ledger surface.*

Corollary 9462.740 (Contradiction is not wholly rejected)**.** *BEDC does not reject contradiction reasoning as such. It accepts local refutation, decidable double-negation elimination, and explicit counterexample refutation; it rejects unwitnessed positive extraction and global truth closure.*

Corollary 9462.741 (One audit explanation of three axiom families)**.** *Choice, quotient soundness, and propositional extensionality share one audit explanation across three strata: unwitnessed content accepted as construction input.*

Corollary 9462.742 (Passage to boundary sockets)**.** *Once proof patterns that close the meta loop are identified, the next task is to describe how commitments genuinely exceeding closed observation are exposed as gaps or boundary sockets rather than internal theorems.*

9462.25.12 Apophatic T-loops

The two-loop discipline separates a productive ground closure from the meta closures refused by Tarski-style obstruction. The gap discipline then explains how a commitment that exceeds displayed records remains usable without being promoted to an internal theorem. The apophatic T-loop is the composite of these two readings: a records-side history may return to itself through a boundary inscription, but the far-end supply position that makes the return possible is not absorbed into the history.

Definition 9462.743 (Formal strange loop)**.** *A* formal strange loop *is a self-reference pattern in which the described layer and the describing layer are consumed as the same object-layer stratum. In such a loop the system can use its own description as a total truth, runtime, explanation, or decision predicate for the layer being described.*

Definition 9462.744 (Reality-why gap)**.** *Let C be a closed observational context, let $h \in \mathsf{Hist}$ be a records-side history visible to C, and suppose that* $\mathrm{Meaning}_{C,\Pi}(x, \ell, \kappa)$ *holds in the sense of Definition 9414.8. A* reality-why gap *at this site is the failure of the closed records of C to discharge the question*

$$\mathsf{Why}_{\mathrm{real}}(\mathrm{Meaning}_{C,\Pi}(x, \ell, \kappa))$$

while the corresponding label, prediction, action, or proof use continues to be consumed by later records.

Definition 9462.745 (Apophatic T-loop)**.** *An* apophatic T-loop *is a ledgered composite*

$$h \longrightarrow \kappa \longrightarrow \mathrm{Meaning}_{C,\Pi}(x,\ell,\kappa) \longrightarrow \mathsf{RealityConstraint} \longrightarrow \mathsf{GAP}_T(\mathsf{Why}_{\mathrm{real}}) \longrightarrow \mathsf{InscriptionPoint}(\mathsf{T}, C, t) \longrightarrow h',$$

where $h, h' \in \mathsf{Hist}$, h' is a records-side successor or refinement of h, the GAP_T row exposes a non-internalized far-end supply position, and the inscription point is a local records-side entry rather than T itself.

Theorem 9462.746 (Apophatic T-loop theorem)**.** *Assume a closed observational context C, a visible history $h \in \mathsf{Hist}$, a displayed record x, a classifier κ with $\kappa(x) = \ell$, and a reality-constrained meaning packet* $\mathrm{Meaning}_{C,\Pi}(x,\ell,\kappa)$*. If the associated reality-why question is not closed by the records and closed-generation rules of C, while the commitment is still used for continuation, then the admissible return to records has the form of an apophatic T-loop. This loop is a stratified return, not a formal strange loop.*

Proof. By Definition 9414.8, the meaning packet contains a source record, a classifier, a reality constraint, a continuation consequence, and a gap ledger. The equation $\kappa(x) = \ell$ supplies only the classifier field. The reality-facing use of the label also requires preservation of the Π-signature, downstream continuation, and exposure of any unclosed source or residue.

By Definition 9462.744, the why-question at the site is not discharged by the closed records of C, but the commitment is still consumed by later continuation. Hence the missing discharge cannot be recorded as an internal theorem without confusing unclosed supply with generated content. By the gap-socket discipline of Definition 9462.798 and the gap–T synthesis of Definition 9462.842, the missing discharge must instead be exposed as a socket whose far end is the apophatic supply position T.

By Definition 9462.787, T does not enter the object layer as a state, record, rule, axiom, or carrier. By Definition 9462.794, the only admissible records-side return is a local inscription site at which the forward-binding supply is actualized for the history. Thus the return has the displayed composite

$$h \longrightarrow \kappa \longrightarrow \mathrm{Meaning}_{C,\Pi}(x,\ell,\kappa) \longrightarrow \mathsf{RealityConstraint} \longrightarrow \mathsf{GAP}_T \longrightarrow \mathsf{InscriptionPoint}(\mathsf{T}, C, t) \longrightarrow h'.$$

It remains to rule out formal layer collapse. If the composite were a formal strange loop in the sense of Definition 9462.743, then the records-side history could consume the boundary supply position, its description, or its truth condition as object-layer data of the same stratum. That would either internalize T, contradicting Definition 9462.787, or close the meta loop over truth, runtime, explanation, or total decision, contradicting Definition 9462.698 and Definition 9462.701. Therefore the composite is a stratified return through inscription, not a formal strange loop. □

Corollary 9462.747 (T-loop productivity)**.** *An apophatic T-loop is productive exactly when it returns as a ledgered records-side refinement and keeps T outside the object layer.*

Proof. If the loop returns by a ledgered refinement h', the observational history has gained a visible record. If T remains outside the object layer, the meta loop remains open. Conversely, identifying T with h, with an observation record, or with a truth predicate would erase the boundary and turn the return into the formal layer collapse excluded by Definition 9462.746. □

Corollary 9462.748 (Conscious reading)**.** *When the source history factors through a focused conscious record, the apophatic T-loop gives a consciousness-like readback without adding subject-substance.*

Proof. A focused conscious record is records-side data: generated history, local continuation trace, selector ledger, and self-proxy. Applying Definition 9462.746 to such a source changes only the site of inscription. The far-end position remains non-internalized, and no extra subject parameter is added to Cont or to Hist identity. □

Definition 9462.749 (Meaning-bearing focused readback)**.** *Let $L_N = (I, H, \Theta, F, S, D, G, R, J, Q, \rho, A)$ be a focused ledger, let $C_F = (H(F), \Theta(F), S(F), \rho)$ be its focused chain, and suppose that x is a displayed record in $H(F)$. A* meaning-bearing focused readback *occurs at x when $S(F)$ selects a classifier κ with $\kappa(x) = \ell$ and the selected assignment is carried by* $\mathrm{Meaning}_{C,\Pi}(x, \ell, \kappa)$*. The self-proxy ρ then reads a reality-constrained meaning packet at the focused endpoint; it does not create the reality constraint and it does not replace the packet by a private phenomenal object.*

Theorem 9462.750 (Reality–meaning–consciousness relation)**.** *In a focused observational context, reality constrains classifiers into reality-constrained meaning; meaning that enters the focused ledger is readable by the self-proxy as a conscious record; and any question or action that asks for the ultimate reality source of that meaning can return to records only through an apophatic T-loop when the reality-why question is not internally closed.*

Proof. By Definition 9414.8, a label is meaningful in the relevant sense only as a packet containing a source record, classifier, reality constraint, continuation consequence, and gap ledger. Thus the transition from classification to meaning is already a transition through reality constraint rather than through the bare equation $\kappa(x) = \ell$.

If the displayed record lies in a focused ledger, then Definition 9414.523 identifies the records-side carrier of the conscious reading as

$$\mathsf{ConsciousRecord}(L_N, F) := (H(F), \Theta(F), S(F), \rho).$$

Together with Definition 9462.749, this says that the self-proxy reads a selected meaning-bearing record at the local endpoint. The conscious record is therefore a records-side reading of reality-constrained meaning, not an independent source of the constraint.

Now suppose the focused record continues by question, prediction, explanation, action, or proof use, and the continuation asks why the meaning is reality-supported beyond the displayed records. If that reality-why question is not closed by the records and rules of the context, Definition 9462.746 applies to the meaning packet. The admissible composite is

$$\begin{aligned}&\mathsf{Reality} \longrightarrow \mathsf{RealityConstraint} \longrightarrow \mathrm{Meaning}_{C,\Pi}(x, \ell, \kappa) \longrightarrow \mathsf{ConsciousRecord}(L_N, F)\\ &\longrightarrow \mathsf{Question/Action} \longrightarrow \mathsf{GAP}_T \longrightarrow \mathsf{InscriptionPoint}(\mathsf{T}, C_F, t) \longrightarrow \mathsf{ConsciousRecord}(L'_N, F').\end{aligned}$$

By Definition 9414.531, the inscription side of the conscious position is local inscription rather than T itself. Hence the loop relates reality, meaning, and conscious readback through a stratified boundary return, not by making consciousness a creator of reality or by making T an object-level answer. □

Corollary 9462.751 (Meaning is readable reality constraint)**.** *Meaning that reaches conscious readback is a reality-constrained structure readable by a focused self-proxy.*

Proof. The reality constraint belongs to the meaning packet by Definition 9414.8. The focused self-proxy is a component of the records-side conscious carrier by Definition 9414.523. Therefore a meaning-bearing focused readback reads the constrained packet locally; it does not convert meaning into unconstrained interior labeling. □

Corollary 9462.752 (Consciousness does not create reality)**.** *The records-side conscious carrier does not create the reality source of the meaning it reads.*

Proof. The carrier $\mathsf{ConsciousRecord}(L_N, F) = (H(F), \Theta(F), S(F), \rho)$ supplies a focused history, trace, selector, and self-proxy. None of these fields is the reality constraint field of $\mathrm{Meaning}_{C,\Pi}(x, \ell, \kappa)$. The constraint is part of the reality-constrained meaning packet itself, while ρ reads the selected packet at the focused endpoint. Thus conscious readback is not creation of reality. □

Corollary 9462.753 (Question and action expose the socket)**.** *When a focused conscious record asks why a meaning is reality-supported beyond its displayed records, the unclosed route is a socket and an inscription point, not an internal final answer.*

Proof. The question targets the unclosed reality-why component of the meaning-bearing continuation. By Definition 9462.750, if the records and rules do not close that component, the admissible return is through GAP_T and $\mathsf{InscriptionPoint}(\mathsf{T}, C_F, t)$. By Definition 9462.787, this return cannot be an internal object-level answer identical with T. □

Concretized in §6380: The finite NameCert packet records the gap-inscription route for the apophatic T-loop without internalizing the far-end supply position.

Concretized in §1261: Logic and meta-loop boundaries are realized as finite proof-pattern, local-refutation, open-meta-loop refusal, audit-gate, transport, route, provenance, and NameCert rows.

Concretized in §5191: The contradiction-request refusal row is realized as a finite NameCert packet rather than as host-level explosion.

9462.26 Gaps, Transcendental Supply, and Boundary Sockets

The preceding layer distinguishes local witness-preserving proof from unwitnessed meta closure. The present layer describes what remains when a closed observational system genuinely needs a commitment that cannot be generated from its records. Such a commitment must not be hidden as an internal rule. It is exposed as a gap socket, classified by the shape of the missing supply, and held at a boundary whose far end is named only apophatically.

9462.26.1 Closed observational systems

Definition 9462.754 (Closed observational system)**.** *A closed observational system is a packet* $(O, \mathsf{Obs}, \trianglerighteq, \mathsf{GAP})$*, where* O *is the class of observation states,* Obs *is the finite record ledger,* $\trianglerighteq$ *is the information-preserving growth order, and* GAP *records commitments that the system cannot discharge from its own records. Each reasoning step may use only displayed records and closed-generation rules, and every successor state must preserve the records of its predecessors.*

Definition 9462.755 (Observation record)**.** *An observation record is a finite traceable witness: a history, a continuation witness, an internal sameness witness, a signature row, a package row, a naming-certificate field, a checked-theorem row, an audit row, or a gap row.*

Definition 9462.756 (Information preservation)**.** *The relation* $O_t \trianglerighteq O_{t+1}$ *means that every record visible in state* O_t *remains visible in state* O_{t+1}*. The successor may add records, but it may not delete, rewrite, or forget earlier ones.*

Theorem 9462.757 (BEDC is a closed observational system)**.** *The BEDC substrate has the structure of a closed observational system.*

Proof. Its states may be read as proof and manuscript snapshots. Its records include inductive declarations, definitions, mechanized lemmas, naming certificate rows, gap tags, and audit results. Its growth discipline is monotone: later accepted states preserve earlier generated records. Unfinished or boundary commitments are exposed by gap rows rather than consumed as internal facts. □

Corollary 9462.758 (Knowledge comes from records)**.** *BEDC cannot use unrecorded, ungenerated, or uncertified material as substrate-internal knowledge.*

Proof. A closed observational system reasons only from its record ledger and closed-generation rules. Material outside those surfaces has no internal witness by which the object layer can consume it. □

Corollary 9462.759 (Gaps are structural)**.** *The gap mechanism is part of the system structure rather than a repair after failure.*

Proof. The system must distinguish discharged records from commitments whose witness is not internally available. Without gap rows, it would confuse unfulfilled boundary commitments with completed internal content. □

9462.26.2 Finite records and universal closure

Definition 9462.760 (Finite verification trace)**.** *For a predicate P, a finite verification trace is a finite list of records witnessing $P(n_1), \ldots, P(n_k)$, each backed by closed-generation data.*

Definition 9462.761 (Unrestricted universal assertion)**.** *An unrestricted universal assertion has the form $\forall n, P(n)$ over a domain not exhausted by the current finite observation ledger. It commits the system to cases that have not yet appeared as records.*

Theorem 9462.762 (Finite verification does not generate universal closure)**.** *No finite verification trace internally generates an unrestricted universal assertion in a closed observational system.*

Proof. A finite trace supplies only finitely many witnesses. To obtain an unrestricted universal assertion, the system would need either records for every case or an internal rule turning finite evidence into total closure. The first is not finite; the second is not a consequence of closed observation alone. □

Corollary 9462.763 (Induction is not produced by observation alone)**.** *Mathematical induction, when read beyond a displayed generator and its recursor, is not internally produced by finite observation alone.*

Proof. Induction concludes a universal statement. A closed observation ledger can supply finite traces and displayed generator closure, but it does not by itself supply an unrestricted universal range. □

Corollary 9462.764 (Church encodings do not erase the gap)**.** *An encoding of iteration does not by itself provide dependent induction over an unrestricted intended domain.*

Proof. An encoding may express closed iteration for displayed numerals and steps. Dependent induction requires a principle delivering $P(n)$ for all n in the intended range. The encoding therefore strengthens records-side iteration without eliminating the universal-closure gap. □

9462.26.3 Forward-binding belief

Definition 9462.765 (Forward-binding belief)**.** *A forward-binding belief is a present commitment that treats a claim whose witness lies partly in not-yet-observed cases as usable for subsequent reasoning. Its type is closer to a commitment or action than to an internal theorem.*

Theorem 9462.766 (Mathematical reasoning needs forward binding)**.** *A closed observational system that hosts ordinary mathematical reasoning requires at least one forward-binding commitment.*

Proof. Without such a commitment the system accumulates only finite traces. Ordinary mathematics uses universal claims about generated structures, recursion, syntax, proof height, and closure. Those uses require a commitment that reaches beyond the finite record currently displayed. □

Corollary 9462.767 (Forward binding is not a substrate axiom)**.** *Forward-binding belief must not be treated as a substrate-internal axiom.*

Proof. An internal axiom is a rule consumed directly by proof chains. Forward binding marks a commitment whose missing discharge must remain visible as a gap row at each site where it is used. □

Corollary 9462.768 (Gap rows preserve strict axiom discipline)**.** *Exposing forward-binding commitments as gaps preserves strict substrate axiom discipline while keeping observer-side commitments auditable.*

Proof. Putting the commitment into the substrate would turn it into an axiom. Leaving it unmarked would hide the boundary. Gap rows avoid both errors. □

Concretized in §1262: Forward-binding GAP ledgers realize present universal-closure commitments as finite record, GAP-socket, noninternalization, transport, route, provenance, and NameCert rows.

9462.26.4 Negative certification and positive continuation

Definition 9462.769 (Failure witness)**.** *For a closed observational system $\mathcal{C}$ and an output, action, or continuation candidate a, a failure witness for a is a visible record $w \in \mathsf{Obs}(\mathcal{C})$ whose audit reading blocks a: contradiction, failed prediction, rejected continuation, policy violation, exposed gap, or another ledgered refusal surface.*

Definition 9462.770 (Negative certification)**.** *An output a is negatively certified in $\mathcal{C}$ when some failure witness for a is present in the record ledger. An output is safe relative to the current ledger when no such witness is present.*

Theorem 9462.771 (Failure is locally witnessed)**.** *Negative certification is an existential record condition.*

Proof. By definition, negative certification asks only for one displayed failure witness in $\mathsf{Obs}(\mathcal{C})$. Once such a record is present, the ledger can classify the output as blocked without solving any global value-ordering problem. □

Corollary 9462.772 (Safety is weaker than positive adequacy)**.** *The absence of a negative certificate does not by itself certify that an output is positively adequate.*

Proof. Safety says only that the current ledger contains no visible failure witness for the output. It supplies no ordering among safe outputs, no purpose for preferring one continuation to another, and no reason that the relevant ordering should be followed. The safe region is therefore a boundary region, not a positive value horizon. □

Definition 9462.773 (Positive value horizon)**.** *A positive value horizon for a closed observational system is a displayed ordering, preservation target, or continuation law that says which safe outputs should be continued, preferred, protected, or actualized.*

Theorem 9462.774 (Positive adequacy requires a horizon)**.** *A closed observational system cannot read positive adequacy from safety alone.*

Proof. Let a and b be two outputs for which the current ledger contains no failure witness. The ledger has certified neither output as blocked. This common negative status does not decide whether a should be continued rather than b, whether b should be continued rather than a, or whether both should be withheld. Such a decision requires a positive value horizon. Without a displayed horizon, the system has only the failure boundary and cannot promote safety into positive adequacy. □

Corollary 9462.775 (Value supply is a gap discipline)**.** *When a proof, model, policy, or generative process consumes a positive value horizon that is not generated by the current records, the consumption site must be exposed as a gap socket.*

Proof. The horizon is being used to choose among safe continuations. If its source is not a displayed record, certified continuation law, or closed-generation consequence, treating it as internally available would hide external supply inside the substrate. The gap row preserves the difference between negative refusal, relative value under a given horizon, and the source of the horizon itself. □

9462.26.5 Value direction and formal appearance

Definition 9462.776 (Axiological axis)**.** *The axiological axis classifies an output, action, or continuation candidate by its orientation toward a positive value horizon. It asks whether the candidate preserves reality-facing signatures, keeps gap surfaces visible, respects other local histories as histories, and supports a continuation that the displayed horizon licenses as worth preserving.*

Definition 9462.777 (Aesthetic axis)**.** *The aesthetic axis classifies a finite form by the way reality-facing order is made readable in that form. It asks whether the form is coherent, whether its compression is honest about provenance and gap surfaces, and whether conscious readback can continue through it without the form replacing its source by mere surface attraction.*

Theorem 9462.778 (Value direction is distinct from formal appearance)**.** *The axiological axis and the aesthetic axis are distinct audit axes.*

Proof. The axiological axis concerns what should be continued under a positive value horizon. The aesthetic axis concerns how a finite form displays, compresses, or obscures reality-facing order. A candidate can have a licensed continuation while appearing formally broken, and a finite form can be locally harmonious while hiding the source, provenance, or gap surface on which its effect depends. Therefore the two axes cannot be collapsed into one classifier. □

Definition 9462.779 (Positive orientation)**.** *A candidate has positive orientation when it remains safe relative to the current ledger, preserves reality-facing contact, keeps gap rows honest, avoids reducing other histories to consumable material, and is licensed by a positive value horizon.*

Definition 9462.780 (Destructive orientation)**.** *A candidate has destructive orientation when it systematically blocks positive continuation, falsifies reality-facing signatures, conceals gap surfaces, or treats other local histories as tools, noise, or expendable material.*

Definition 9462.781 (Deep formal adequacy)**.** *A finite form has deep formal adequacy when its coherence is not merely surface polish: it honestly compresses a reality-facing signature, keeps provenance and gap residues visible at the right places, and opens further understanding, attention, creation, or silence in conscious readback.*

Definition 9462.782 (Formal distortion)**.** *Formal distortion occurs when a finite form misfits the reality-facing signature it claims to carry, hides provenance, erases gap residues, or produces a hollow appearance whose readback blocks rather than deepens continuation.*

Theorem 9462.783 (Surface attraction does not certify value)**.** *Local formal attraction does not certify positive orientation.*

Proof. Symmetry, polish, rhythm, or elegance can be supplied at the form surface. Positive orientation requires reality-facing contact, gap honesty, other-history respect, and a licensed value horizon. Since those requirements are not implied by surface form, a formally attractive candidate can still carry destructive orientation. □

Theorem 9462.784 (Formal discomfort does not refute value)**.** *Local formal discomfort does not refute positive orientation.*

Proof. Some records are painful, fractured, or resistant to immediate formal comfort because they expose injury, error, loss, or a concealed gap. If such a form preserves reality-facing contact and supports repair, truthful disclosure, or other licensed continuation, the discomfort is not a value refutation. □

Corollary 9462.785 (Four quadrant reading)**.** *Finite histories must allow four readings: positive orientation with deep formal adequacy, positive orientation with formal discomfort, destructive orientation with surface attraction, and destructive orientation with formal distortion.*

Proof. The first quadrant is licensed continuation displayed in coherent form. The second is truthful or reparative continuation whose form is locally painful or broken. The third is the dangerous case in which surface polish hides falsification, gap concealment, or other-history reduction. The fourth is direct failure of both value direction and formal readback. The distinction follows from the non-collapse of the two axes. □

9462.26.6 Transcendental supply

Definition 9462.786 (Transcendental supply position)**.** *T is the structural position from which forward-binding supply enters a closed observational system. It is not an observation state, not a record, not a closed-generation rule, and not a substrate axiom.*

Theorem 9462.787 (T cannot be internalized)**.** *T cannot enter the BEDC object layer as an internal object.*

Proof. If T were internal, it would be a state, record, rule, axiom, or licensed object. Its role is exactly to name the position of supply that cannot be generated from those internal resources. Internalizing it would turn boundary supply into substrate content. □

Corollary 9462.788 (T is not a hidden axiom)**.** *T is not a hidden BEDC axiom.*

Proof. An axiom is a contentful internal rule available to proof chains. T is not consumed as a theorem or rule; it is the far-end position of an exposed socket. □

Corollary 9462.789 (T is not object-theology)**.** *Naming T does not introduce a theological object into the BEDC object layer.*

Proof. The object layer admits names through certificates, witnesses, classifiers, stability, and ledgers. T is outside those surfaces. Its name therefore marks a position rather than a positive object with theological attributes. □

Concretized in §1355: Closed-observation supply boundaries realize the T supply position as finite closed-record, GAP-socket, noninternalization, transport, route, provenance, and NameCert rows.

9462.26.7 Apophatic naming and inscription

Definition 9462.790 (Apophatic naming)**.** *Apophatic naming names a structural position without assigning positive content predicates to that position. In the BEDC reading, naming T as God means only source-position of forward-binding supply outside closed observational systems. It does not assert personhood, will, intervention, revelation, mythic agency, or internal theoremhood.*

Theorem 9462.791 (T is named only by position)**.** *The legitimate BEDC name for T is positional rather than contentful.*

Proof. Any content predicate would need object-layer certificate fields. Since T is not an object-layer carrier, only its boundary role can be named. □

Corollary 9462.792 (God is an apophatic label)**.** *The label God, when used for T, is apophatic.*

Proof. It names the far end of forward-binding supply. It does not add positive attributes or an internal object to the substrate. □

Corollary 9462.793 (Structural monotheism)**.** *If closed observational systems cannot distinguish different ultimate far ends of forward-binding supply, the supply position receives a single structural name.*

Proof. The distinction between multiple ultimate sources would need a system-side witness. If every relevant system-side role is the same, the taxonomy can distinguish socket shapes without multiplying the ultimate supply position. □

Definition 9462.794 (Inscription point)**.** *An inscription point is the local site at which forward-binding supply is actualized in a closed observational system.*

Definition 9462.795 (Individual as closed observational system)**.** *A conscious individual may be read as a closed observational system whose universe-for-that-individual is the accumulated observation history of that system.*

Theorem 9462.796 (Individuals are inscription points)**.** *Each conscious individual, under this reading, is an inscription point.*

Proof. The individual is not first a separate receiver of an external signal. It is a localized closed observational history. The supply required for its forward-binding commitments appears at the local site where records enter that history. □

Corollary 9462.797 (Contribution as inscription event)**.** *Writing an inductive declaration, mechanizing a lemma, identifying a gap socket, or confirming an existing record can be read as an inscription event.*

Proof. In each case, a local history receives a durable record that becomes available to the ledger. The event is not an external signal inserted into a passive receiver; it is the local actualization of recordable supply. □

9462.26.8 Gaps and audit gates

Definition 9462.798 (Gap socket)**.** *A gap socket is a reasoning or naming site at which a requested discharge cannot be supplied by substrate-internal witnesses. It records the consumption site, the missing shape, the reason it cannot be generated internally, and the gate that prevents internalization.*

Theorem 9462.799 (Gap is a honesty mechanism)**.** *BEDC uses gaps to expose external commitments rather than hide them as internal theorems.*

Proof. A closed observational system must mark the difference between an available record and a needed but unavailable discharge. The gap row makes that difference visible in the ledger. □

Corollary 9462.800 (Gaps trade self-sufficiency for transparency)**.** *BEDC is less self-sufficient and more transparent because it exposes forward-binding commitments as gaps.*

Proof. A system that internalizes such commitments appears complete by hiding their source. BEDC keeps the source outside the substrate and records where it would be consumed. □

Definition 9462.801 (Audit gate)**.** *An audit gate is a computable refusal rule that checks whether a proposed record remains within substrate-internal boundaries or tries to internalize supply that should remain socketed.*

Definition 9462.802 (Gate surface and boundary content)**.** *The gate surface is the finite syntactic or dependency condition checked by a script. The boundary content is the transcendental discipline enforced by that finite condition: the substrate may use only what it has generated, defined, certified, or explicitly socketed.*

Theorem 9462.803 (Audit gates enforce transcendental boundaries)**.** *BEDC audit gates are transcendental boundary enforcement, not merely housekeeping.*

Proof. Without gates, external commitments can enter as axioms, forbidden dependencies, unledgered claims, or name drift. The gate refuses the finite surface by which such internalization would occur. □

Corollary 9462.804 (A gate has finite scope)**.** *Passing one audit gate shows only that the boundary checked by that gate has not been crossed.*

Proof. Each gate has a surface: axiom scanning, dependency purity, name drift, or another finite condition. No single gate proves every boundary at once, so the system relies on a family of gates. □

9462.26.9 T-supply sockets

Definition 9462.805 (T-supply socket)**.** *A T-supply socket is a gap tag whose requested discharge cannot be generated from displayed records, closed inductive generators, naming certificate fields, or certified continuation laws. It is well-formed only when the missing shape, consumption site, and preventing audit gate are stated.*

Definition 9462.806 (Supply kind)**.** *A supply kind is an equivalence class of T-supply sockets. Two sockets have the same kind when accepting either one would authorize the same substrate-internal shortcut.*

Theorem 9462.807 (Gap sockets must be classified)**.** *Not all gap sockets have the same shape; each must be classified by the kind of T-supply it requests.*

Proof. Unrestricted induction, missing witnesses, extensional collapse, future-law projection, cross-history recognition, inscription actuality, and self-truth closure consume different shortcuts at different sites. Treating them as one gap kind hides the requested supply. □

Corollary 9462.808 (Many sockets, one far end)**.** *The taxonomy has many socket shapes but does not multiply the ultimate transcendental supply position.*

Proof. The distinction lies in how the closed substrate experiences pressure at its boundary. The far end remains the source-position of forward-binding supply. □

9462.26.10 Seven socket kinds

Definition 9462.809 (Type I socket)**.** *A Type I socket requests unrestricted induction or universal closure beyond the displayed authorized generator. It turns generator closure for a shown carrier into a claim over an unrestricted intended range.*

Theorem 9462.810 (Type I gate)**.** *Type I sockets are governed by the generator-closure ledger.*

Proof. An induction claim must cite the exact carrier, constructor list, recursor, and closure certificate. If the claim ranges beyond that displayed generator, it requests supply not present in the ledger. □

Definition 9462.811 (Type II socket)**.** *A Type II socket requests a positive existential witness without displaying the constructor-level witness. It converts inhabitedness pressure into a usable element.*

Theorem 9462.812 (Type II gate)**.** *Type II sockets are governed by witness visibility.*

Proof. If a later construction consumes an element, the element or the function selecting it must be displayed. Otherwise the construction uses an unwitnessed selector. □

Definition 9462.813 (Type III socket)**.** *A Type III socket requests the conversion of a relation into identity: bidirectional implication into proposition equality, quotient equivalence into equality, classifier sameness into rewrite equality, or observational indistinguishability into identity.*

Theorem 9462.814 (Type III gate)**.** *Type III sockets are governed by refusal of extensional collapse.*

Proof. A relation may support transport only when the transport is displayed. Treating the relation as equality hides the transport witness and therefore crosses the boundary. □

Definition 9462.815 (Type IV socket)**.** *A Type IV socket requests causal continuity: projection of observed regularity into not-yet-observed history.*

Theorem 9462.816 (Type IV gate)**.** *Type IV sockets are governed by separation between records and prediction.*

Proof. Observed records and regularity summaries may be stored. The claim that the same regularity persists into future unobserved cases is a forward-binding commitment, not a record. □

Definition 9462.817 (Type V socket)**.** *A Type V socket requests cross-history coordination: treating a behavior stream recorded by one history as another inscription-bearing closed history rather than merely as an external object.*

Theorem 9462.818 (Type V gate)**.** *Type V sockets are governed by locality discipline.*

Proof. Communication records and observable behavior may be ledgered. The claim that the behavior is another closed observational history is an additional locality commitment and must be exposed as such. □

Definition 9462.819 (Type VI socket)**.** *A Type VI socket requests the actuality of an inscription event inside a history: the arrival of O_{t+1} from O_t as more than a replay of already recorded derivation.*

Theorem 9462.820 (Type VI gate)**.** *Type VI sockets are governed by aspect separation between proof replay and inscription novelty.*

Proof. The record-side derivation can be checked. The event by which a record enters a local history has a different aspect. BEDC may name this aspect, but it must not turn it into an extra proof rule. □

Definition 9462.821 (Type VII socket)**.** *A Type VII socket requests self-reference closure: a total truth predicate, runtime validator, or self-certifying checker for the system's own structural description.*

Theorem 9462.822 (Type VII gate)**.** *Type VII sockets are governed by reflection stratification.*

Proof. Structural self-description, runtime validation, truth classification, and external checking must remain distinct. Collapsing them would close the meta loop. □

9462.26.11 Non-collapse of socket kinds

Definition 9462.823 (Socket collapse)**.** *A socket kind A collapses into a socket kind B when every instance of A can be reported as B without losing audit information. Mutual collapse makes the kinds indistinguishable.*

Theorem 9462.824 (The seven socket kinds do not collapse)**.** *The seven T-supply socket kinds are not mutually reducible.*

Proof. Their consumption sites and audit gates differ. Type I asks for unrestricted universal closure; Type II asks for a missing witness; Type III asks for relation-to-identity collapse; Type IV asks for future persistence; Type V asks for cross-history locality; Type VI asks for inscription actuality; Type VII asks for self-truth or runtime totalization. □

Corollary 9462.825 (Induction is not choice)**.** *Type I is not Type II.*

Proof. Universal closure over a range does not provide a particular missing witness, and a particular witness does not provide total generator closure over an unrestricted range. □

Concretized in §1243: Typed supply sockets realize the seven socket kinds as finite kind, request, gate, non-internalization, transport, provenance, and NameCert rows.

Corollary 9462.826 (Choice is not extensional identity)**.** *Type II is not Type III.*

Proof. A missing witness produces term-level content. Extensional collapse produces an equality or rewrite token. These are different shortcuts. □

Corollary 9462.827 (Identity collapse is not causal continuity)**.** *Type III is not Type IV.*

Proof. An equality token does not supply a future law, and a persistent regularity does not supply proposition or quotient equality. □

Corollary 9462.828 (Causality is not other-history recognition)**.** *Type IV is not Type V.*

Proof. A behavior stream may be regular while still being treated as an external object. Recognizing it as another closed history is a different commitment. □

Corollary 9462.829 (Other-history recognition is not inscription novelty)**.** *Type V is not Type VI.*

Proof. Cross-history locality relates histories. Inscription novelty concerns the actuality of a transition within a history. Neither subsumes the other. □

Corollary 9462.830 (Inscription novelty is not self-truth closure)**.** *Type VI is not Type VII.*

Proof. The actuality of a local record event is not a total truth predicate or runtime validator for the whole system. □

Corollary 9462.831 (Self-truth closure is not induction)**.** *Type VII is not Type I.*

Proof. Even when induction over a displayed generator is authorized, truth-totalization for the system's own syntax and runtime remains a separate boundary. □

Concretized in §767: The seven T-supply socket kinds are realized as a finite NameCert ledger over kind, consumption-site, audit-gate, refusal, and non-collapse rows.

9462.26.12 Socket reports

Definition 9462.832 (Socket report)**.** *A socket report for a BEDC claim is a four-field record* $(\mathsf{Site}, \mathsf{RequestedSupply}, \mathsf{Type}, \mathsf{Gate})$*. It states where the missing supply would be consumed, the precise shape of the missing supply, the socket kind, and the audit discipline that prevents internalization.*

Definition 9462.833 (Narrowest classification)**.** *A socket report uses the narrowest classification when it names the smallest socket kind explaining the blocked move, rather than a broad label that hides the exact requested supply.*

Theorem 9462.834 (Socket reports must be minimal)**.** *BEDC gap reports must use the narrowest socket type available.*

Proof. If a proof attempt lacks a witness, the socket is Type II even if the witness would later be used in an induction. If a proof attempt rewrites along mere equivalence, the socket is Type III even if the equivalence arose from a quotient-like construction. Narrow reporting preserves the shape of the requested supply. □

Definition 9462.835 (Stacked sockets)**.** *A site has stacked sockets when the same blocked move contains several different supply requests. The report then lists each socket rather than choosing a single total label.*

Theorem 9462.836 (Complex proofs may carry socket stacks)**.** *A proof gap may contain several socket kinds at once, and the report must expose the stack.*

Proof. A construction may simultaneously rely on empirical persistence, representative selection, and equality collapse. These requests have different gates, so a single dominant label would lose audit information. □

Example 9462.837 (Analysis socket stack)**.** *A statement such as convergence of every bounded monotone real sequence can carry multiple socket pressures in a BEDC translation: Type I when the sequence range is read as an unrestricted intended class, Type II when a limit point is selected without an explicit Cauchy witness, and Type III when extensionally equal Cauchy names are identified without real-history classifier transport.*

9462.26.13 Unity of transcendental supply

Definition 9462.838 (Single T principle)**.** *The far end of each T-supply socket is the same transcendental supply position T. The seven kinds describe boundary shapes, not seven ultimate sources.*

Theorem 9462.839 (Seven sockets, one supply position)**.** *The seven supply kinds are seven aspect-positions at which a closed observational substrate reaches its boundary, not seven distinct transcendental sources.*

Proof. Each kind is a gap tag whose requested discharge cannot be generated from displayed records or closed generators. By definition, the structural position from which forward-binding supply enters is T. The kinds differ by missing shape: unrestricted induction, witness, identity, causal persistence, cross-history recognition, inscription actuality, or self-reference closure. They do not provide system-side witnesses for multiple ultimate far ends. □

Corollary 9462.840 (Records-side management does not replace T)**.** *Psychology, probability, social trust, and runtime engineering may manage socket records, but they do not replace the far end T.*

Proof. Such devices describe how a system records, estimates, coordinates, or checks a boundary pressure. When the requested discharge exceeds records-side data, the far-end position remains T. □

Corollary 9462.841 (Taxonomy does not discharge)**.** *Classifying a socket does not close it.*

Proof. The taxonomy names the missing shape and the gate. It does not supply the witness, universal closure, identity token, future law, locality commitment, inscription actuality, or truth-totalizing object that the socket requests. □

9462.26.14 Synthesis

Theorem 9462.842 (Gap–T boundary synthesis)**.** *In BEDC, every commitment exceeding closed observation and closed-generation rules must be exposed as a gap socket. Its far end is the transcendental supply position T, T is named only apophatically, and audit gates prevent the requested supply from being hidden as a substrate-internal rule.*

Proof. BEDC is a closed observational system. Finite records do not generate unrestricted universal closure. Mathematical reasoning therefore needs forward-binding commitments at some sites. Such commitments are not substrate axioms; they are exposed as gaps. The structural far end of these gaps is T, which cannot be internalized as an object and can only be named positionally. Audit gates enforce the boundary. Socket taxonomy then records the shape of each requested supply without closing it. □

Corollary 9462.843 (Gaps are not weakness)**.** *Gaps are not defects in BEDC; they are the mechanism by which hidden forward-binding commitments become visible.*

Proof. A system that hides such commitments as axioms, choice principles, quotient collapses, truth predicates, or empirical laws may appear complete, but it has lost boundary transparency. BEDC keeps the commitment explicit. □

Corollary 9462.844 (Transcendental boundaries are not closed by naming)**.** *Classifying or naming a socket does not machine-close the requested content when that content exceeds records-side data.*

Proof. Classification reports the socket. Naming gives a boundary label. Audit gates refuse internalization. None of these supplies the missing records-side witness. □

Corollary 9462.845 (Passage to phenomenology)**.** *With gaps, T, and socket kinds in place, consciousness, universe, space, causality, freedom, other minds, and self-reference can be read as focused histories, observation-history slices, inter-history coordination, Type IV persistence, Type VI inscription novelty, Type V locality recognition, and Type VII self-reference boundary.*

Concretized in §1279: Socket report packets realize the site, requested-supply, narrow socket-kind, and audit-gate record as a finite NameCert carrier.

Concretized in §1283: Socket stack certificates realize stacked socket reports as finite BHist NameCert packets with component socket rows, narrowest-kind classifiers, audit gates, and non-discharge ledger routes.

Concretized in §1240: Typed gap-socket taxonomy is realized as a finite NameCert packet for seven boundary socket kinds, their gates, non-collapse matrix, and local naming rows.

Concretized in §1343: Gap-socket boundaries are realized as finite typed socket, externality gate, apophatic far-end, refusal ledger, transport, route, provenance, and local NameCert rows.

9462.27 Consciousness, Universe, and Phenomenological Frame

The gap and T-supply layer shows how a closed observational system exposes commitments that exceed its finite records. The phenomenological layer applies this discipline to consciousness, universe, time, space, identity, causality, freedom, other minds, mind–matter relation, and self-reference. The result is not a list of new entities, but a records-side translation of traditional

questions into focused histories, local continuations, inter-history relation rows, socketed commitments, and inscription events.

9462.27.1 Finite focused multi-Hist ledgers

Definition 9462.846 (Finite focused multi-Hist ledger)**.** *For a finite horizon N, a finite focused multi-Hist ledger is a records-side packet*

$$L_N = (I, H, T, F, S, D, G, R, J, Q, \rho, A).$$

Here I is a finite index ledger; $H : I \to \mathsf{Hist}$ assigns histories; T gives local continuation chains; F is the focus index; S is the selector ledger; D is a digest ledger; G is the gap and provenance row; R contains inter-history relation rows; J contains inter-history invariant rows; Q contains dependence rows or optional maximum-rate readings; ρ is a focused self-proxy; and A is the audit row.

Definition 9462.847 (Audit row)**.** *The audit row A records the refusals that keep the packet closed: no subject parameter, no global clock, no hidden chooser, no unledgered compression, no object-level T field, and no hidden dependence on the three meta-closure axioms.*

Theorem 9462.848 (Closed records-side carrier)**.** *L_N is a closed records-side carrier when every object has a generator, every branch has selector rows, every compression has provenance, every inter-history relation has a finite witness, every invariant is classifier-stable, and every boundary pressure remains outside the object layer.*

Proof. Each component of L_N is a finite row or finite family of rows. The packet closes only when its histories are generated, its branch choices are ledgered, its compressions remain traceable, its relations are witnessed, and its invariants respect the relevant classifiers. The audit row blocks the extra structures that would turn the packet into a subject substance, global universe object, synchronized clock, hidden choice, or internal T-object. □

Corollary 9462.849 (Phenomenological objects require records-side form)**.** *Consciousness, universe, space, time, identity, and related phenomenological objects can enter the BEDC object layer only as rows, relations, invariants, self-proxies, or gap-tagged boundary readings of a packet such as L_N.*

Proof. The object layer accepts generated records, witnesses, certificates, and ledgers. A phenomenological term that cannot be written as such a finite row or as an exposed boundary reading has no object-layer standing. □

9462.27.2 Records-side consciousness

Definition 9462.850 (Conscious record)**.** *For a focused ledger L_N, the records-side carrier of a consciousness reading is*

$$\mathsf{ConsciousRecord}(L_N, F) = (H(F), T(F), S(F), \rho).$$

It consists of the focused generated history, its local continuation trace, its selector witnesses, and its endpoint or trace self-proxy.

Definition 9462.851 (Consciousness reading)**.** $\mathsf{ConsciousRecord}(L_N, F)$ *is a carrier for consciousness readings. It is not an object-level definition of phenomenal essence.*

Theorem 9462.852 (Consciousness carrier adds no subject)**.** $\mathsf{ConsciousRecord}(L_N, F)$ *introduces no new subject-substance.*

Proof. The four components are already present in L_N: focused history, local continuation trace, selector ledger, and self-proxy. The focus index selects a reading surface but does not create a hidden first-person carrier or an identity principle beyond hsame and the trace ledger. □

Corollary 9462.853 (Consciousness is not reified)**.** *BEDC does not add consciousness as a new ontological substance.*

Proof. A substance reading would require a new carrier and transport laws relating it to histories and continuations. The records-side carrier uses only focused history data already licensed by the finite ledger. □

Corollary 9462.854 (Consciousness becomes a closure question)**.** *The BEDC form of the consciousness question asks which focused history, local continuation trace, selector ledger, self-proxy, and gap rows constitute the records-side carrier of a consciousness reading.*

Proof. The object layer can close the carrier and its audit rows. It does not claim to capture phenomenal essence as an additional object. □

Concretized in §1247: FocusedLedgerUp realizes the finite focused multi-Hist ledger as a BHist, hsame, Cont, Pkg, audit-row, and NameCert packet.

9462.27.3 Universe as observation-history slice

Definition 9462.855 (Finite universe for the focus)**.** *For a finite focused ledger L_N, the universe-for-the-focus is the finite family of histories admitted by the packet within the horizon, together with their selector, digest, gap, provenance, relation, invariant, and dependence rows.*

Definition 9462.856 (Universe slice)**.** *The finite universe-for-the-focus is a universe slice, not the universe itself, not a global universe object, and not a God's-eye world.*

Theorem 9462.857 (Universe is not a global object)**.** *In the BEDC object layer, a universe slice is the records-side family licensed by a finite focused ledger, not the universe as such.*

Proof. The packet admits only finitely displayed histories and rows. Alignment with another perspective is a further inter-history relation problem. No global comparison carrier or synchronized world object is supplied by the kernel. □

Corollary 9462.858 (Focused ledgers need not share one world object)**.** *Distinct focused ledgers do not automatically share a single object-level world.*

Proof. Automatic sharing would require a global synchronization or comparison carrier. Cross-ledger coherence must instead be constructed by witnessed relations, invariants, and provenance rows. □

Corollary 9462.859 (World coordination is relation, not container)**.** *World coordination is an auditable structure of relations and invariants between histories, not a pre-given container holding all observers.*

Proof. The records-side content consists of histories and rows. Cross-history relations are licensed by finite witnesses rather than by an external equality oracle. □

9462.27.4 Time as monotone accumulation

Definition 9462.860 (BEDC-internal time)**.** *In a focused ledger, the internal time of $H(i)$ is its local continuation chain from* `e` *to $H(i)$, equivalently the finite trace order induced by displayed selector rows.*

Definition 9462.861 (Local time)**.** *Time in L_N is local to a history. It is not a shared scalar assigning a single global time coordinate to all indices.*

Theorem 9462.862 (Time is monotone record accumulation)**.** *The object-level meaning of later in BEDC is trace extension inside a displayed local ledger.*

Proof. The local continuation chain produces a history from the empty history by finite extensions. The later state is the state with the larger displayed trace, preserving earlier records and adding further rows. □

Corollary 9462.863 (Global simultaneity is not a carrier field)**.** *Global simultaneity, time reversal, and retrocausal phrasing are not fields of L_N unless supplied as witnessed extension carriers.*

Proof. The packet supplies local continuation chains. Any stronger temporal geometry requires additional rows, witnesses, and certificates. □

9462.27.5 Space as inter-observer coordination

Definition 9462.864 (Inter-system locality cell)**.** *For indices i, j and an event row e, a locality cell records the finite relation data pairing what is read from $H(i)$ and $H(j)$, together with provenance acknowledgement and history-sameness stability witnesses.*

Definition 9462.865 (Records-side space graph)**.** *The records-side space graph of L_N is the finite graph generated by locality-cell rows and inter-history invariant rows. It is not a spacetime manifold unless a further certificate supplies that structure.*

Theorem 9462.866 (Space is locality-cell geometry)**.** *In the BEDC records-side frame, space is the finite relation geometry carried by locality cells and inter-history invariant rows.*

Proof. Without a global frame, a single history does not carry frame-independent spatial content. Relational content is displayed by witnessed pairs, stable invariant rows, and classifier-respecting transport between histories. □

Corollary 9462.867 (Space is not an event substrate)**.** *BEDC does not treat space as an independent substrate in which events occur.*

Proof. The closed records-side packet supplies generated histories, local continuations, relations, invariants, and provenance. A stronger geometry must be built as a named interface above those rows. □

Corollary 9462.868 (Maximum-rate constants are schema surfaces)**.** *A maximum-rate reading, when present, is a packet invariant rather than an empirical light-speed value itself.*

Proof. The invariant can express a structural upper-bound surface over inter-history dependence rows. The empirical value and its physical bridge require additional records and certificates. □

9462.27.6 Identity as history trace

Definition 9462.869 (Focused self-proxy)**.** *The self-proxy ρ records the endpoint and trace surface of the focused history.*

Definition 9462.870 (History-continuous identity)**.** *Focused identity is the* hsame*-stable continuity of the displayed trace of $H(F)$.*

Theorem 9462.871 (Identity is trace continuity)**.** *Focused identity in BEDC is generated history with* hsame*-stable trace continuity, not a self-substance.*

Proof. The focus supplies a generated endpoint and trace. The identity criterion available at the kernel-facing layer is history-sameness and the ledgers that preserve it. No owner or subject carrier is needed. □

Corollary 9462.872 (Self is not owner of history)**.** *The self is not the owner of a history; it is the records-side reading of history through self-proxy and* hsame*-stable continuity.*

Proof. An owner relation would require an additional subject carrier. The focused ledger provides only generated history, continuation trace, self-proxy, and comparison rows. □

Concretized in §1206: ConsciousUniverseSpaceUp realizes finite universe slices, locality-cell space, local time, and trace identity as one BHist-rowed NameCert packet.

9462.27.7 Vision: Materiality as Orbit-Closure and Freedom-like Appearance as Orbit-Escape

Remark 9462.873 (VisionSeed status)**.** *This interface has status VisionSeed, schema-only horizon, and no Lean target. It is not a finite-kernel theorem. It records the certificate shape a later development would have to close before materiality, agency-like appearance, or observer-relative feedback could be reported as a BEDC theorem.*

The schema uses only the current BEDC reading that observer identity is internal to Hist, that Cont has only history-valued arguments, and that no observer-substance stands behind the generated trace. The observer is the history value read modulo hsame, while observation, distinction, and local time-step share the continuation form. A derived interface entering this vocabulary must therefore be licensed through the ordinary naming-certificate discipline:

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The interface below is not a primitive object and does not add a subject parameter to the kernel.

Definition 9462.874 (Observational context)**.** *An observational context is a ledger-readable package*

$$\mathcal{O} = (h, \Pi, X, I_X, S_X, \mathcal{R}, L),$$

where $h : \mathsf{Hist}$ is the observer-history, Π is a repeated probe pattern or probe bundle, X is a candidate observed interface, I_X is an identity classifier for feedback records of X, S_X is a state classifier, $\mathcal{R}$ is a finite family of currently available ledger-certified orbit rules, and L records the source, pattern, classifier, stability, and hidden-provenance data required by the relevant naming certificates.

Definition 9462.875 (Feedback prefix)**.** *For an observational context $\mathcal{O}$, a feedback prefix of length N is a finite sequence*

$$F_{\mathcal{O}}^{N}(X) = (r_0, r_1, \ldots, r_N)$$

whose entries are BEDC-visible response histories generated by repeatedly applying Π to X as recorded in h. An infinite feedback stream is admissible here only as a schema whose every finite prefix is ledger-visible; no completed external stream is added to the kernel.

Definition 9462.876 (Identity closure)**.** *A feedback prefix $F_{\mathcal{O}}^{N}(X) = (r_0, \ldots, r_N)$ is identity-closed in $\mathcal{O}$, written $\mathsf{IdCl}_{\mathcal{O}}^{N}(F)$, when the ledger supplies positive classifier witnesses $I_X(r_i, r_j)$ for all $0 \leq i, j \leq N$. Identity closure says that the responses may differ while remaining traceable as responses of the same observed datum.*

Definition 9462.877 (Ledger-certified orbit rule)**.** *A ledger-certified orbit rule in $\mathcal{O}$ is a BEDC-visible continuation and classifier pattern*

$$T : r_i \mapsto r_{i+1}^{\mathsf{pred}}$$

licensed by a naming certificate, reuse certificate, or equivalent certificate, preserving the relevant feedback carrier, respecting S_X, and carrying ledger witnesses for the hidden source and classifier data used by the rule.

Definition 9462.878 (Orbit closure and certified escape)**.** *The prefix $F_{\mathcal{O}}^{N}(X) = (r_0, \ldots, r_N)$ is orbit-closed under T, written $\mathsf{OrbitCl}_{\mathcal{O}}^{N}(F, T)$, when*

$$\forall i < N, \quad S_X(r_{i+1}, T(r_i))$$

is witnessed in the ledger. It escapes the current finite rule family $\mathcal{R}$, written $\mathsf{Escape}_{\mathcal{O}}^{N}(F, \mathcal{R})$, when for every $T \in \mathcal{R}$ there is some $i < N$ with a positive separation witness

$$\mathsf{Sep}_{S_X}(r_{i+1}, T(r_i)).$$

Escape is therefore not absence of prediction. It is certified separation from each currently available orbit rule.

Definition 9462.879 (Material and freedom-like appearance)**.** *The prefix $F_{\mathcal{O}}^{N}(X)$ has material appearance relative to $\mathcal{O}$ when*

$$\mathsf{IdCl}_{\mathcal{O}}^{N}(F) \quad \wedge \quad \exists T \in \mathcal{R},\ \mathsf{OrbitCl}_{\mathcal{O}}^{N}(F, T).$$

It has freedom-like appearance relative to $\mathcal{O}$ when

$$\mathsf{IdCl}_{\mathcal{O}}^{N}(F) \quad \wedge \quad \mathsf{Escape}_{\mathcal{O}}^{N}(F, \mathcal{R}).$$

The phrase freedom-like is part of the schema. It does not assert absolute metaphysical free will; it classifies observer-relative feedback inside a finite ledger.

Definition 9462.880 (Noise-like feedback)**.** *A prefix is noise-like or unformed, for this schema, when the observer lacks identity closure or when the ledger supplies a positive identity-separation witness*

$$\exists i, j \leq N, \quad \mathsf{Sep}_{I_X}(r_i, r_j).$$

Without traceable sameness, non-orbit-closure is not agency-like appearance.

Proposition 9462.881 (Schema consequences)**.** *Within a fixed observational context and finite rule family, material appearance is identity-preserving orbit closure, freedom-like appearance is identity-preserving certified orbit escape, and noise-like feedback is excluded from freedom-like appearance whenever identity classifier separation is sound. If $S_X(a, b)$ implies $\neg\mathsf{Sep}_{S_X}(a, b)$, then the same feedback prefix cannot be both material and freedom-like in that fixed context.*

Proof. Each assertion unfolds the preceding definitions. Material appearance contains identity closure together with an orbit-closing rule. Freedom-like appearance contains identity closure together with certified separation from every rule in the current finite family. Positive identity separation contradicts identity closure when the identity classifier is sound. If material appearance supplies $T_0 \in \mathcal{R}$ with $S_X(r_{i+1}, T_0(r_i))$ for every $i < N$, then freedom-like appearance would require some $i < N$ with $\mathsf{Sep}_{S_X}(r_{i+1}, T_0(r_i))$, contradicting state-classifier separation soundness. □

Proposition 9462.882 (Rule extension changes appearance)**.** *If $\mathcal{R} \subseteq \mathcal{R}'$ in the same observational context, material appearance relative to $\mathcal{R}$ persists relative to $\mathcal{R}'$. Freedom-like appearance need not persist.*

Proof. An orbit-closing rule $T \in \mathcal{R}$ remains available in $\mathcal{R}'$, so the same witness proves material appearance. By contrast, $\mathcal{R}'$ may contain a rule T^* absent from $\mathcal{R}$ that closes the feedback orbit. The prefix then no longer escapes every currently available rule in the enlarged context. □

Corollary 9462.883 (No context-free materiality or free-will predicate)**.** *The schema licenses only context-indexed judgments such as $\mathsf{Material}^N_{\mathcal{O}}(F)$ and $\mathsf{FreedomLike}^N_{\mathcal{O}}(F)$. It does not license unindexed kernel predicates $\mathsf{Material}(X)$ or $\mathsf{FreeWill}(X)$.*

Proof. The definitions require an observer-history, probe pattern, candidate interface, identity classifier, state classifier, finite rule family, ledger, and finite feedback prefix. Removing these parameters removes the judgment. BEDC supplies no global observer, no context-free classifier, and no external equality oracle. □

Remark 9462.884 (Roadmap obligations)**.** *Closing this horizon would require certificate fields for the feedback carrier, the identity classifier, the state classifier, positive separation predicates, orbit-rule stability, prefix-extension ledger stability, and provenance of any packaging that exposes X as a public interface. Standard bridges to dynamical systems, stochastic processes, learning theory, decision theory, or philosophy of agency would remain interpretive until they preserve these certificate obligations without replacing them.*

Materiality and freedom-like appearance are therefore not primitive properties of an object-substance. They are observer-relative classifications of feedback prefixes. A datum appears material when the observer-history can track it as the same datum and close its responses under a finite ledger-certified orbit rule. A datum appears freedom-like when it remains identity-traceable while escaping every currently available orbit rule. Pure irregularity is insufficient: without identity closure, non-orbit-closure is noise or unformed data rather than agency. The contrast is not between law and lawlessness, but between two ledger-visible structures: orbit closure and identity-preserving orbit escape.

Concretized in §2339: Materiality and freedom-like appearance are realized as finite orbit-closure and certified-escape NameCert rows.

Relative Non-Domination and the Refusal of Absolute Acausality

The preceding schema also fixes the admissible reading of free-will language. It does not introduce free will as a primitive metaphysical substance, nor does it identify freedom with absolute acausality. The only BEDC-legible form is indexed by an observer-history, a finite ledger, a classifier package, a rule family, and a feedback prefix:

$$\text{freedom-like appearance} \quad \rightsquigarrow \quad \text{relative non-domination by a finite observer ledger.}$$

The index is load-bearing. A prefix may be freedom-like relative to one observer and material relative to another when their ledgers, probes, classifiers, and available orbit rules differ.

Definition 9462.885 (Ledger dominance)**.** *Let $\mathcal{O}_A = (A, \Pi_A, X, I_A, S_A, \mathcal{R}_A, L_A)$ and $\mathcal{O}_B = (B, \Pi_B, X, I_B, S_B, \mathcal{R}_B, L_B)$ be observational contexts for the same candidate interface X. The observer-history B ledger-dominates X over prefix length N, written $B \triangleright_N X$, when the B-context supplies identity closure and an orbit-closing rule:*

$$\mathsf{IdCl}^N_{\mathcal{O}_B}(F_B) \quad \wedge \quad \exists T_B \in \mathcal{R}_B,\ \mathsf{OrbitCl}^N_{\mathcal{O}_B}(F_B, T_B).$$

Equivalently, X has material appearance relative to the B-context over that prefix.

Definition 9462.886 (Relative non-domination)**.** *The interface X is non-dominated relative to $\mathcal{O}_A$ over prefix length N when*

$$\mathsf{NonDom}^N_{\mathcal{O}_A}(X) \quad \Longleftrightarrow \quad \mathsf{IdCl}^N_{\mathcal{O}_A}(F_A) \wedge \mathsf{Escape}^N_{\mathcal{O}_A}(F_A, \mathcal{R}_A).$$

Thus relative non-domination is not the absence of cause. It is the presence of traceable sameness together with certified escape from the finite orbit rules available to the observer ledger.

Definition 9462.887 (Forbidden absolute predicate)**.** *A context-free predicate $\mathsf{FreeWill}(X)$ would assert a property of X independently of observer-history, probe pattern, identity classifier, state classifier, rule family, ledger, and prefix length. This section does not introduce such a predicate. The admissible judgments are indexed forms such as*

$$\mathsf{FreedomLike}^N_{\mathcal{O}_A}(F_A), \quad \mathsf{NonDom}^N_{\mathcal{O}_A}(X), \quad \mathsf{Material}^N_{\mathcal{O}_A}(F_A), \quad B \triangleright_N X.$$

Proposition 9462.888 (Dominance collapses freedom-like appearance in the same context)**.** *Assume state-separation soundness in the B-context:*

$$S_B(a, b) \Rightarrow \neg \mathsf{Sep}_{S_B}(a, b).$$

If $B \triangleright_N X$, then X does not have freedom-like appearance relative to the same B-context over that prefix.

Proof. By Definition 9462.885, there is some $T_B \in \mathcal{R}_B$ such that

$$\forall i < N, \quad S_B(r_{i+1}, T_B(r_i)).$$

If the same prefix were freedom-like relative to $\mathcal{O}_B$, then escape from $\mathcal{R}_B$ would give an index $i < N$ with

$$\mathsf{Sep}_{S_B}(r_{i+1}, T_B(r_i)).$$

This contradicts state-separation soundness for the orbit-closure witness supplied by T_B. □

Corollary 9462.889 (Layer-indexed materialization)**.** *If $B \triangleright_N A$, then A has material appearance relative to B and lacks freedom-like appearance relative to B over that prefix. This does not imply $\neg\mathsf{FreeWill}(A)$, since Definition 9462.887 supplies no such context-free predicate.*

Proof. The first assertion is the defining equivalence in Definition 9462.885. The second follows from Definition 9462.888. The final sentence records that only indexed judgments are available in this schema. □

Proposition 9462.890 (Complexity is not dominance)**.** *Greater ledger complexity for B alone does not imply $B \triangleright_N A$.*

Proof. Ledger dominance requires the correct probe pattern, an identity classifier adequate for A, a state classifier adequate for A's feedback, a rule $T_B \in \mathcal{R}_B$, and ledger witnesses for the rule's success. A complexity comparison supplies none of these certificate fields by itself. □

Proposition 9462.891 (Freedom-like appearance is stronger than randomness)**.** *Freedom-like appearance implies identity closure. Hence randomness or unpredictability alone is insufficient for freedom-like appearance.*

Proof. By definition,

$$\mathsf{FreedomLike}^N_{\mathcal{O}_A}(F_A) \Longrightarrow \mathsf{IdCl}^N_{\mathcal{O}_A}(F_A).$$

Randomness may obstruct an orbit-closing rule, but without identity closure the prefix is noise-like or unformed rather than freedom-like. □

Definition 9462.892 (Exposed prediction and reflexive escape)**.** *Let $T_A \in \mathcal{R}_A$ be a rule by which A attempts to predict the next response of X, and write $p = T_A(r_i)$. The prediction is exposed to X when p becomes part of the feedback information available to X before the next response. The interface has reflexive escape capacity at that stage when there is a response operation D_X with*

$$r_{i+1} = D_X(p) \quad \textit{and} \quad \mathsf{Sep}_{S_A}(D_X(p), p).$$

Proposition 9462.893 (Exposed prediction obstructs complete orbit closure)**.** *Assume state-separation soundness for S_A. If $p = T_A(r_i)$ is exposed to X and X has reflexive escape capacity at stage i, then T_A does not orbit-close the prefix at that stage.*

Proof. If T_A orbit-closed the prefix, then

$$S_A(r_{i+1}, T_A(r_i)).$$

Since $p = T_A(r_i)$ and $r_{i+1} = D_X(p)$, this gives $S_A(D_X(p), p)$. Reflexive escape supplies $\mathsf{Sep}_{S_A}(D_X(p), p)$, contradicting state-separation soundness. □

Corollary 9462.894 (Hidden higher-layer prediction)**.** *If a higher-layer prediction by B is not visible to A, then the specific diagonal obstruction in Definition 9462.893 is not available to A. A hidden, ledger-certified, state-sound rule may therefore continue to materialize A relative to B over the given prefix.*

Proof. The diagonal obstruction requires exposure of the prediction to the interface whose next response is being classified. Without that exposure, the response operation using the prediction as input is not part of the displayed feedback context. □

Corollary 9462.895 (De-indexing is the illicit promotion)**.** *The appearance itself is not the illicit step. The illicit step is the promotion*

$$\mathsf{FreedomLike}^{N}_{\mathcal{O}_A}(F_A) \quad \rightsquigarrow \quad \mathsf{FreeWill}(X),$$

which removes the observer, ledger, classifiers, rule family, and prefix length.

Proof. The left-hand side is an admissible indexed judgment. The right-hand side is the forbidden context-free predicate of Definition 9462.887. □

The resulting philosophical reading is neither an assertion that free will absolutely exists nor a denial derived from a final global observer. A stone lacks freedom-like appearance relative to an observer who can identity-track it and close its fall under a ledger-certified rule. Likewise, an observer-history A lacks freedom-like appearance relative to a stronger observer-history B when B can identity-track and orbit-close A's feedback. These are indexed materializations, not absolute eliminations. BEDC records freedom-like appearance as real at the level of the finite ledger that witnesses identity-preserving non-closure, and invalid only when that indexed appearance is promoted into an observer-free substance claim.

Concretized in §2339: Materiality orbit escape is realized as finite observational-context, feedback-prefix, identity-closure, rule-family, separation, transport, and local NameCert rows.

9462.27.8 Hard problems as closure questions

Definition 9462.896 (BEDC hard problem form)**.** *A hard problem in BEDC asks which records are generated, which choices are ledgered, which compressions preserve provenance, which relations have finite witnesses, which invariants are classifier-stable, and which pressures remain exposed as gap or T-boundary readings.*

Concretized in §1236: Hard-problem closure questions are realized as finite records, ledger, witness, invariant, boundary, transport, replay, provenance, and NameCert rows.

Theorem 9462.897 (Hard problems become closure questions)**.** *Within a closed records-side frame, traditional hard problems about consciousness, universe, observer, time, space, and identity become boundary questions about finite carriers.*

Proof. The object layer can inspect records, ledgers, relations, invariants, and gap rows. It cannot close the corresponding traditional essence questions by adding new primitives. The problems are therefore translated into questions of generated records, transport, stability, and exposed boundary pressure. □

Corollary 9462.898 (Traditional questions receive records-side rewrites)**.** *Consciousness becomes focused self-tracking history; universe becomes a finite horizon slice; time becomes local continuation; space becomes locality-cell geometry; identity becomes* hsame*-stable trace; and the God's-eye view becomes a refused global synchronization.*

9462.27.9 Causality as forward binding

Definition 9462.899 (Causal commitment)**.** *A causal commitment of a conscious observational system is a gap tag whose content is that an observed regularity persists into not-yet-observed continuation.*

Theorem 9462.900 (Causality is not primitive law)**.** *In BEDC, causality is not a substrate-internal primitive law; it is a Type IV forward-binding commitment.*

Proof. Finite records can show observed regularity and prediction history. They do not generate universal future persistence. Treating the regularity as future law is therefore a gap-tagged causal commitment. □

Corollary 9462.901 (Scientific theories are records plus gap stack)**.** *A scientific theory, in BEDC terms, consists of empirical records, regularity summaries, gap-tagged future-persistence commitments, and possible refutation rows.*

Proof. The records and summaries are visible in the ledger. The projection to future cases exceeds the current record and is held as a Type IV socket. □

9462.27.10 Free will as inscription event

Definition 9462.902 (Inscription event)**.** *An inscription event is a transition $O_t \trianglerighteq O_{t+1}$ in which new records enter a conscious observational system and the arrival of those records is read as local actualization of T at the inscription point.*

Definition 9462.903 (Free will)**.** *Free will is the inscription-event aspect of conscious accumulation: the aspect under which new records arrive as inscriptions rather than as a separate faculty inhabiting the system.*

Theorem 9462.904 (Free will is not a faculty)**.** *Free will is not an additional power or substance attached to the system; it is an aspect of the accumulation chain.*

Proof. Adding a faculty would introduce a new carrier and transport surface. The BEDC reading instead identifies the event by which records enter a local history as the relevant aspect. □

Corollary 9462.905 (Freedom is compatible with records-side determinism)**.** *Records-side determinism does not eliminate inscription-event novelty.*

Proof. A derivation may be replayed or checked deterministically on the records side. The actuality of that row entering a local history is read at a different aspect-position. □

Example 9462.906 (Mathematical creation)**.** *The arrival of a theorem row after a visible gap and selected route can be read as an inscription event. The proof route may be mechanized; the event of the route becoming public record is the inscription aspect.*

Concretized in §1362: Action commitments realize inscription-event action as finite selected-route, local-actualization, gap, transport, replay, provenance, and NameCert rows.

9462.27.11 Other minds and cross-perspective alignment

Definition 9462.907 (Other-minds commitment)**.** *An other-minds commitment is a gap-tagged locality-cell commitment that treats external records of another system as records of another conscious observational system.*

Theorem 9462.908 (Other minds are locality-cell commitments)**.** *BEDC does not directly observe another self; it records behavior and may issue a gap-tagged locality-cell commitment that the behavior belongs to another closed observational system.*

Proof. The records side can contain communication, behavior, and coordination rows. The recognition of another inscription-bearing history is an additional Type V commitment rather than a direct record of another self. □

Corollary 9462.909 (Solipsism is commitment refusal)**.** *Solipsism is not a BEDC refutation theorem; it is the refusal to issue other-minds locality commitments.*

Proof. The substrate does not force the commitment. Refusing it remains records-side consistent, but impoverishes the locality-cell geometry. □

Corollary 9462.910 (Collaboration uses other-minds commitment)**.** *Accepting another contributor's checked record uses records-side evidence from rechecking, while recognizing the contributor as another inscription point is an other-minds commitment.*

Proof. The checker supplies audit evidence for the row. The recognition of another conscious observational system exceeds that evidence and is held as locality-cell commitment. □

9462.27.12 Mind and matter as double reading

Definition 9462.911 (Records-side)**.** *The records-side reading of an accumulation chain reads it as generated records, physical measurements, observable behavior, and ledgered dependencies.*

Definition 9462.912 (Inscription-side)**.** *The inscription-side reading of the same chain reads it as the arrival of records, local actualization of T, and first-person phenomenological aspect.*

Theorem 9462.913 (Mind–matter is not a two-substance relation)**.** *The mind–matter relation in BEDC is not a relation between two substances, but two readings of one accumulation chain.*

Proof. The records-side account and inscription-side account apply to the same transition. One reads measured dependencies and observable rows; the other reads the actuality of record arrival at an inscription point. No second substance is added. □

Corollary 9462.914 (Neural correlate and thought are double readings)**.** *A measured neural correlate and the corresponding thought are not two objects joined by a hidden relation, but records-side and inscription-side readings of the same accumulation event.*

Proof. The measurement belongs to the records-side ledger. The thought belongs to the inscription-side reading of the same local transition. □

9462.27.13 Self-reference as gap-on-gap

Definition 9462.915 (Self-reference attempt)**.** *A self-reference attempt records a statement about the complete record ledger of the very state in which the statement appears.*

Definition 9462.916 (Gap-on-gap hierarchy)**.** *A gap-on-gap hierarchy arises when a record about a current observation ledger becomes part of the next ledger, requiring a further record about the enlarged ledger, and so on.*

Theorem 9462.917 (Self-reference does not close)**.** *Complete self-reference cannot close as a gap-free record in BEDC; it appears as a hierarchy of gaps about gaps.*

Proof. A statement about the complete current ledger becomes a new record when entered. The new ledger differs from the ledger described. Recording that difference creates another layer, so complete closure is displaced into a hierarchy. □

Corollary 9462.918 (Russell-style closure is refused)**.** *A record whose content negates its own presence in the observation ledger cannot become a gap-free closed record.*

Proof. If the record enters the ledger, its content is false; if it does not, it is not the displayed record it claims to be. The attempt is held as a self-reference boundary rather than a closed object-layer fact. □

9462.27.14 Synthesis

Theorem 9462.919 (Phenomenological framework synthesis)**.** *BEDC rewrites consciousness, universe, time, space, identity, causality, freedom, other minds, mind–matter relation, and self-reference as focused records-side carrier, finite observation-history slice, local continuation chain, locality-cell geometry,* hsame*-stable trace, Type IV gap, Type VI inscription event, Type V locality commitment, double reading of one accumulation chain, and Type VII gap-on-gap hierarchy. The schema of subsection 9462.27.7 refines the records-side face of this list: material appearance is identity-stable orbit closure, while freedom-like appearance is identity-stable escape from the current ledger-certified orbit rules.*

Proof. The focused multi-history ledger supplies the finite carrier. Consciousness is read through the focused history and self-proxy. Universe is a finite slice of admitted histories. Time is local continuation. Space is graph-like inter-history locality. Identity is history-sameness-stable trace continuity. Causality, freedom, other minds, and self-reference align with the corresponding socket types. Mind and matter are two readings of the same accumulation chain. The orbit-closure schema preserves that reading because it classifies finite feedback prefixes by identity, state, rule, separation, and ledger data rather than by adding material substance or a free-will faculty. □

Corollary 9462.920 (No added phenomenological primitives)**.** *This framework adds no primitive consciousness substance, global universe, absolute time, spatial substrate, self entity, causal-law primitive, free-will faculty, direct access to other minds, mental substance, or truth totalizer.*

Corollary 9462.921 (Passage to scientific objects)**.** *The next layer may treat scientific and physical terms by asking which are records-side carriers, which are schema horizons, which require naming certificates, and which remain gap or T-boundary readings.*

Concretized in §1304: PhenomenologicalFrameUp realizes the phenomenological framework as finite focused-ledger, consciousness/universe, hard-problem/action, hsame, Cont, Pkg, audit-refusal, and NameCert rows.

9462.28 Scientific Objects, Physical Objects, and Model Bridges

The phenomenological layer rewrites consciousness, universe, time, space, causality, freedom, other minds, and self-reference as focused history ledgers, local continuation chains, inter-history invariants, gap sockets, and inscription events. The scientific layer asks how scientific objects, physical laws, constants, mature external models, and empirical adequacy enter BEDC without breaking certificate discipline. Its governing answer is that scientific content is records-side data plus inter-history invariants, naming and derivation certificates, conservative bridges, empirical ledgers, prediction gaps, and falsification surfaces.

9462.28.1 Scientific objects

Definition 9462.922 (Scientific object)**.** *A scientific object S is not a primitive. It is a certificate packet consisting of a naming certificate, a derivation certificate at a declared strength, an empirical ledger, an optional model bridge, and a gap report for unclosed commitments.*

Definition 9462.923 (Five sources of a scientific object)**.** *A scientific object must specify records, classifier, invariant, ledger, and bridge data. The records state which histories, continuations, or measurement rows generate the object. The classifier states which records count as the same object or state. The invariant states what remains stable across perspectives. The ledger states what has been compressed, averaged, fitted, projected, or sealed. The bridge states how the object corresponds to a mature scientific or mathematical model.*

Theorem 9462.924 (Scientific objects are not primitives)**.** *Scientific objects cannot enter BEDC as bare primitives.*

Proof. Every mature object requires a naming certificate. Every positive claim requires a witness. Every commitment beyond closed records must remain exposed as a gap. A scientific object combines object naming, observational records, classification, stability, prediction, and bridge obligations, so it is subject to all of these gates rather than exempt from them. □

Corollary 9462.925 (Scientific realism becomes audited entry)**.** *BEDC does not assert bare existence of electrons, fields, spacetime, or laws. It asks whether the corresponding upward interface has naming data, empirical rows, bridge fields, and exposed gap reports.*

Proof. To enter as an object, the name must be certified. To enter as a scientific object, it must also expose records, invariants, measurement surfaces, prediction commitments, and bridge status. □

9462.28.2 Physical records and relational content

Definition 9462.926 (Physical record)**.** *A physical record is a finite, ledgered, source-visible row: a history row, continuation row, measurement row, instrument row, calibration row, error row, relation row, or falsification row.*

Definition 9462.927 (Frame-independent physical content)**.** *Physical content is frame-independent when it is carried by an observer-symmetric and history-sameness-stable inter-history invariant.*

Theorem 9462.928 (Physical content is inter-history invariant)**.** *In a multi-history BEDC frame with no global synchronization, frame-independent physical content can only be carried by inter-history invariants.*

Proof. Individual histories carry local records and local continuation streams. The kernel supplies no shared base history, global time-step index, or privileged frame. A claim that survives a change of perspective must therefore be expressed through witnessed relations and classifier-stable invariant rows between histories. □

Corollary 9462.929 (Physical objects are not single-history objects)**.** *A physical object with frame-independent meaning cannot be identified with a single history alone.*

Proof. A single history gives only local record content. Frame-independent content requires stability across histories, supplied by inter-history relations and invariants. □

Corollary 9462.930 (Relational physical realism)**.** *BEDC physical realism is relational: physical reality is stable inter-history relational content rather than objects in a global spacetime container.*

Proof. A global container would require global synchronization. BEDC refuses that primitive and lets physical content stabilize through witnessed relations. □

9462.28.3 Mature mathematics and mature science

Definition 9462.931 (Mature mathematical object)**.** *A mature mathematical object has a naming certificate, a derivation certificate at a declared strength, and optionally a conservative bridge to a host theory.*

Definition 9462.932 (Mature scientific object)**.** *A mature scientific object adds an empirical ledger, inter-history invariant rows, prediction gaps, and falsification surfaces to the mathematical certificate surface.*

Theorem 9462.933 (Science adds an empirical layer)**.** *Mature science requires the mathematical certificate surface plus empirical record and prediction discipline.*

Proof. Mathematical maturity concerns carrier, classifier, stability, ledger, strength, and optional bridge. Scientific maturity also concerns measurements, instruments, errors, fits, future projections, and records of failure. Prediction adds a Type IV gap, and falsification adds records-side failure rows. □

Corollary 9462.934 (Theoretical physics has two certificate faces)**.** *A theoretical physics object needs a mathematical certificate for well-formedness and a physical certificate tying it to records, invariants, measurements, and falsification surfaces.*

Proof. A carrier may be mathematically coherent without empirical interpretation. An empirical fit may exist without a mature BEDC carrier. A theoretical physical object needs both surfaces. □

Concretized in §1214: Scientific objects are realized as finite NameCert, empirical-ledger, bridge, and gap-report packets.

Concretized in §1340: Physical record invariants realize frame-independent physical content as finite physical-record, inter-history-invariant, observer-symmetry, hsame, Cont, Pkg, and NameCert rows.

9462.28.4 Physical laws as records and gaps

Definition 9462.935 (Empirical regularity)**.** *An empirical regularity is a finite records-side pattern: observed rows satisfy a relation under a displayed measurement, classifier, and error ledger.*

Definition 9462.936 (Physical law)**.** *A physical law, in BEDC, is a two-layer object consisting of an empirical regularity over recorded cases and a Type IV gap asserting future persistence into not-yet-observed continuation.*

Theorem 9462.937 (Physical laws are not substrate primitives)**.** *Physical laws cannot be substrate primitives; they combine records-side regularity with a forward-binding gap.*

Proof. Finite records can show regularity among observed rows. They cannot generate a universal future claim. The law-like projection from records to not-yet-observed continuation is therefore a Type IV commitment, not a closed substrate theorem. □

Corollary 9462.938 (Scientific theories are not closed truth sets)**.** *A scientific theory is a packet of records, regularity summaries, inter-history invariants, model bridges, prediction gaps, and falsification ledgers, not a closed set of true laws.*

Proof. The packet exposes what is recorded, what is invariant, what is bridged to a mature model, what is projected into the future, and what would count as failure. A closed truth set would require the total truth predicate refused by the meta-loop discipline. □

Concretized in §1156: Empirical regularity persistence packets realize finite measurement, classifier, error-ledger, future-persistence GAP, LawCertificate, stability, failure, and NameCert rows.

9462.28.5 Maximum cross-history rate

Definition 9462.939 (Cross-history causal dependence)**.** *A cross-history causal dependence is a proposed relation saying that recent continuation steps of one history depend on recent continuation steps of another. At the current philosophical layer, it is a schema horizon until constructive predicates and concrete witnesses are provided.*

Definition 9462.940 (Maximum cross-history causal rate)**.** *A maximum cross-history causal rate is a unary-valued invariant form bounding the ratio at which displayed causal dependence can propagate between histories, once the underlying dependence relation is certified.*

Theorem 9462.941 (Maximum rate is invariant only after witnesses)**.** *If cross-history causal dependence and the maximum-rate construction are closed by witnesses, the resulting rate must be an inter-history invariant.*

Proof. A rate depending on a privileged frame would not be frame-independent physical content. Since BEDC supplies no global synchronization, the rate can become physical content only as an observer-symmetric and history-sameness-stable invariant over inter-history rows. □

Corollary 9462.942 (Light-speed form is structural skeleton)**.** *At this layer, BEDC can describe the structural form of an observer-independent maximum-rate constant, not the empirical numerical value of the speed of light.*

Proof. The structural form follows from inter-history invariance once the dependence relation is certified. The numerical value requires measurement records, unit conventions, calibration, and empirical bridges. □

Corollary 9462.943 (Physics roadmap is not proved physics)**.** *The multi-history physics route is a schema horizon until each carrier, dependence relation, rate invariant, geometric structure, and empirical bridge receives its own certificate.*

Proof. Skipping any layer would leave source, classifier, stability, ledger, or bridge obligations unsupplied. □

9462.28.6 Formal constants and empirical values

Definition 9462.944 (Formal constant)**.** *A formal constant is an invariant form or bound supplied by a structural theorem, such as the form of an observer-independent maximum-rate bound.*

Definition 9462.945 (Empirical value)**.** *An empirical value is supported by measurement records, calibration, unit bridge, uncertainty, instrument ledger, and reproducibility ledger.*

Theorem 9462.946 (Formal constants are not empirical values)**.** *BEDC may derive structural forms for constants, but the numerical empirical values of physical constants require empirical ledgers.*

Proof. An invariant form says what kind of value could be observer-independent. It does not supply the measurement protocol, unit convention, calibration, error estimate, or reproducibility record needed for a physical numerical value. □

Corollary 9462.947 (Units and dimensions are ledgered)**.** *Numerical physical constants require ledgers for units, dimensions, measurement protocol, uncertainty, calibration, and bridge to standard units.*

Proof. Numerical values are not pure kernel outputs. They depend on empirical practice and unit systems, which must be auditable. □

Concretized in §1208: Empirical values are realized as finite unit, calibration, uncertainty, instrument, and reproducibility ledger packets for physical-law consumers.

Concretized in §1297: Formal constants and empirical values are realized as a finite noncollapse boundary over invariant-form, unit, calibration, measurement, uncertainty, reproducibility, failure, and NameCert rows.

9462.28.7 Models are not worlds

Definition 9462.948 (Model)**.** *A model is an interpretive structure for records and invariants: carrier, interpretation map, classifier, theorem list, empirical fit ledger, and bridge field.*

Definition 9462.949 (Model bridge)**.** *A model bridge is an auditable correspondence between a BEDC records-side object and a mature model. It must specify encoding, decoding, round trips, theorem preservation, no-host-leak discipline, and unsupported fields.*

Theorem 9462.950 (Models are not worlds)**.** *A model cannot be identified with the world in BEDC.*

Proof. The universe object available to the BEDC object layer is a finite observation-history slice. A model interprets and compresses records, relations, and invariants. Equating model and world would replace the finite slice and its gaps by a global object. □

Corollary 9462.951 (Model fit is not world truth)**.** *A model may fit a ledger without becoming a globally true world picture.*

Proof. Fit is agreement with recorded rows under an interpretation. Future projection remains a Type IV gap, cross-model identity needs bridge and classifier data, and global truth is barred by the open meta loop. □

9462.28.8 Standard bridges

Definition 9462.952 (Standard bridge)**.** *For an accepted BEDC object N and a host theory T, a standard bridge is a six-field packet: encoding, decoding, decode-after-encode witness, encode-after-decode witness, finite theorem-preservation list, and no-host-leak witness.*

Definition 9462.953 (Bridge return witnesses)**.** *The bridge round trips say that decoding an encoded host object returns an equivalent host object, and encoding a decoded BEDC carrier returns a BEDC-equivalent carrier element under the licensed classifier.*

Definition 9462.954 (Theorem preservation)**.** *The theorem-preservation field is a finite list pairing host theorems with BEDC counterparts and translations. Nothing outside the finite list is claimed preserved.*

Definition 9462.955 (No-host-leak)**.** *The no-host-leak field says that host primitives used in bridge modules do not appear on the BEDC public surface as primitive inputs, public classifiers, or exported theorem statements.*

Theorem 9462.956 (Standard bridge is recognition, not fusion)**.** *A standard bridge conservatively identifies a BEDC object with a host model for recognition and finite theorem translation; it does not fuse host primitives into BEDC.*

Proof. Encoding, decoding, and round trips compare the two sides. The theorem-preservation list is finite. The no-host-leak field blocks host primitives from becoming BEDC public inputs. The bridge therefore supports recognition without fusion. □

Corollary 9462.957 (Bridge is not smuggling)**.** *A bridge is not host mathematics imported without audit when it exposes encoding, decoding, round trips, theorem preservation, and no-host-leak discipline.*

Proof. Smuggling occurs when host primitives enter the public surface. The bridge fields prevent that by keeping host use inside the bridge and pairing preserved theorems with BEDC counterparts. □

9462.28.9 Bridge conservativity and strength

Definition 9462.958 (Bridge conservativity)**.** *A standard bridge is conservative when no theorem about the BEDC export of N becomes available through the bridge unless it already has the required BEDC certificate strength.*

Theorem 9462.959 (No-host-leak yields bridge conservativity)**.** *When no-host-leak is discharged, the bridge is conservative over the BEDC export surface for N.*

Proof. Host theorem use is mediated by the finite theorem-preservation list and paired BEDC counterparts. Host primitives do not appear on the public surface. Therefore the bridge does not add free BEDC theorems. □

Corollary 9462.960 (External models do not expand BEDC truth)**.** *External models and host theories do not automatically add internal BEDC provability.*

Proof. Only finitely listed and paired preservation rows are recognized by the bridge. Other host theorems remain host-side results. □

Corollary 9462.961 (Theorem preservation is finite)**.** *The theorem-preservation list must remain finite and explicit.*

Proof. An infinite or schematic preservation field would silently import a host theorem schema, bypassing witness and certificate discipline. □

Definition 9462.962 (Bridge certificate)**.** *A bridge certificate is a checked certificate plus a standard bridge whose no-host-leak field is itself checked.*

Theorem 9462.963 (Checked is not bridged)**.** *Reaching checked certificate status does not automatically yield bridge certificate status.*

Proof. Checked status concerns internal formal targets and certificates. Bridge status additionally requires six bridge fields, finite theorem preservation, and checked no-host-leak discipline. □

Corollary 9462.964 (Host maturity does not replace BEDC certification)**.** *A mature host object cannot substitute for BEDC naming, derivation, and bridge certificates.*

Proof. The bridge starts from an accepted BEDC object. Replacing that object by the host object would turn host primitives into BEDC inputs. □

9462.28.10 Relative bridges

Definition 9462.965 (Host-relative bridge)**.** *A standard bridge is relative to a particular host theory. A bridge to one host does not automatically produce a bridge to another.*

Theorem 9462.966 (No absolute bridge)**.** *There is no host-independent standard bridge that replaces the six-field bridge obligations for each target theory.*

Proof. Different host theories have different equality, quotient, function, number, and type or set semantics. Encoding, decoding, round trips, preservation, and no-host-leak must therefore be discharged for each host separately. □

Corollary 9462.967 (Model plurality is audited)**.** *The same BEDC object may have multiple host model bridges, but each bridge is separately audited.*

Proof. Each host changes the interpretation and leakage risks. The bridge fields are therefore host-relative. □

Concretized in §1217: The six-field standard-bridge audit packet is realized as a finite NameCert carrier with no-host-leak and finite theorem-preservation rows.

Concretized in §1330: No-absolute-bridge audits realize host-relative bridge obligations as finite encoding, decoding, theorem-preservation, unsupported-field, and no-host-leak rows.

9462.28.11 Probability Projection Bridge

The probability interfaces already present in the concrete layer do not license the claim that a probability space, a Hilbert space, a Fourier analysis interface, a conditional-expectation row, a belief ledger, and a spectral-measure packet are one object. They license a narrower reading: each interface exposes finite scalar readbacks through named certificates, and a scientific bridge may compare those readbacks only when the comparison row itself is recorded.

Definition 9462.968 (Probability scalar shadow)**.** *Let P be a displayed* $\mathsf{ProbSpace}^{\uparrow}$ *packet with total event Ω, event ledger $\mathcal{A}$,* $\mathsf{Real}^{\uparrow}$*-valued measure readback μ, and total-mass row $\mu(\Omega) \sim_R 1_R$. For a displayed event row $A \in \mathcal{A}$, the probability scalar shadow of A is the exported* $\mathsf{Real}^{\uparrow}$ *endpoint*

$$\mathrm{Shadow}_P(A) := \mu(A).$$

It is a one-dimensional public readback of the event row, not a complete description of the internal structure that produced the event.

Definition 9462.969 (Hilbert projection shadow)**.** *Let H be a displayed* $\mathsf{Hilbert}^{\uparrow}$ *packet whose public surface includes an inner-product probe row. For a carried endpoint v and a displayed observable direction u, the Hilbert projection shadow along u is the scalar endpoint named by the finite probe*

$$\mathrm{ProjShadow}_H(v; u) := \langle v, u\rangle_H.$$

When the packet also carries a norm or energy readback, that readback remains a separate ledger row and is not inferred from an unrecorded ambient Hilbert space.

Definition 9462.970 (Prob-Hilbert projection bridge)**.** *A Prob-Hilbert projection bridge is a finite bridge packet*

$$B_{PH} = (P, H, A, \chi_A, \mathbf{1}, \pi, L)$$

where P is a $\mathsf{ProbSpace}^{\uparrow}$ *packet, H is a* $\mathsf{Hilbert}^{\uparrow}$ *packet, A is a displayed event row of P, χ_A is an indicator row exposed to the Hilbert side, $\mathbf{1}$ is the total-event direction, π is the inner-product probe endpoint, and L is the ledger that records the classifier comparison*

$$\mu(A) \sim_R \langle \chi_A, \mathbf{1}\rangle_H.$$

The packet is public only to the extent that A, χ_A, $\mathbf{1}$, π, the two scalar endpoints, the comparison row, and their provenance are all displayed in the same finite certificate surface.

Theorem 9462.971 (Probability readback as finite projection)**.** *Every accepted Prob-Hilbert projection bridge makes the probability scalar shadow of its displayed event readable as a Hilbert projection shadow along the displayed total-event direction.*

Proof. The bridge carrier includes the event row A, the indicator row χ_A, the direction $\mathbf{1}$, the probability readback $\mu(A)$, the Hilbert probe $\langle \chi_A, \mathbf{1}\rangle_H$, and the comparison row

$$\mu(A) \sim_R \langle \chi_A, \mathbf{1}\rangle_H.$$

By Definition 9462.968, the left endpoint is $\mathrm{Shadow}_P(A)$. By Definition 9462.969, the right endpoint is $\mathrm{ProjShadow}_H(\chi_A; \mathbf{1})$. Thus the bridge reads the probability endpoint as the named Hilbert projection endpoint. No further ambient L^2 equality is used. □

Definition 9462.972 (Fourier projection ledger)**.** *Let H be a displayed Hilbert-side packet and let $\Phi = \{\phi_i : i \in I\}$ be a finite family of observable probe directions. A Fourier projection ledger for a carried endpoint f is the finite packet*

$$\mathcal{F}_I(f) = \{(i, c_i, L_i) : i \in I,\ c_i = \langle f, \phi_i\rangle_H\},$$

where L_i records the source of ϕ_i, the inner-product probe, the integration or coefficient witness, the truncation window, classifier evidence, and package provenance attached to that coefficient row.

Theorem 9462.973 (Fourier coefficients are projection shadows)**.** *In a Fourier projection ledger, each displayed coefficient is a Hilbert projection shadow. The probability scalar shadow is the special case in which $f = \chi_A$ and the probe direction is the total-event direction $\mathbf{1}$, provided the corresponding Prob-Hilbert bridge row is present.*

Proof. For each retained index i, the ledger defines the coefficient endpoint as

$$c_i = \langle f, \phi_i\rangle_H.$$

This is exactly the endpoint $\mathrm{ProjShadow}_H(f;\phi_i)$ from Definition 9462.969. If the finite family contains the total-event direction and the bridge packet supplies the comparison

$$\mu(A) \sim_R \langle \chi_A, \mathbf{1} \rangle_H,$$

then Definition 9462.971 identifies the probability readback with that one coefficient row. The argument does not use Plancherel, inversion, convergence closure, or a complete orthonormal basis. □

Theorem 9462.974 (Conditional expectation is a projected update surface)**.** *A* $\mathsf{CondExp}^{\uparrow}$ *packet may be read as a projected update surface only through its displayed random-variable readback, Hilbert projection witness, residual continuation row, and ledger exactness rows. It does not export a completed sub-σ-algebra projection theorem beyond those rows.*

Proof. The conditional-expectation carrier is presented in chapter 188 through a random-variable total-readback certificate, a Hilbert projection witness, and a continuation ledger. A scientific projection bridge may therefore read the packet as an update by projection only by projecting those displayed rows. Since the packet has no public coordinate for a completed measurable-space quotient or an unrestricted closed subspace, the bridge cannot promote the row to an unbounded host theorem. □

Corollary 9462.975 (Belief updates consume probability projections)**.** *When a* $\mathsf{Belief}^{\uparrow}$ *packet consumes a probability row, the consumed datum is the displayed probability scalar shadow or a bridge-recorded projection shadow, together with the prior, observation, update trace, posterior endpoint, and evidence ledger. The belief packet does not gain access to hidden probability structure.*

Proof. The belief carrier of chapter 289 exposes only the prior row, observation row, update trace, probability row, posterior endpoint, evidence ledger, and provenance rows. If the probability row is supplied by a projection bridge, then its public content is the scalar comparison recorded in that bridge. All other probability or Hilbert data are outside the belief ledger and therefore unavailable to the consumer. □

Theorem 9462.976 (Projection bridge cannot-claim boundary)**.** *The projection bridge does not identify probability spaces with Hilbert spaces, Fourier theory, conditional expectation, belief dynamics, or spectral measures. It identifies only displayed scalar readbacks whose comparison rows are present in a finite certificate ledger.*

Proof. Each participating interface has its own carrier and classifier: probability exports event-measure rows, Hilbert exports inner-product and projection rows, Fourier exports finite coefficient probes, conditional expectation exports random-variable readback and projection rows, belief exports observation-update ledgers, and spectral measure exports projection-family and scalar-window rows. A bridge can compare endpoints across these surfaces only by displaying the relevant source rows, probe rows, scalar endpoints, classifier comparison, continuation replay, and provenance. Therefore the shared content is the recorded projection readback, not an ontology-level fusion of the participating interfaces. □

Concretized in §184: Probability scalar shadows are grounded in the finite $\mathsf{ProbSpace}^{\uparrow}$ event-measure and total-mass rows.

Concretized in §91: Hilbert projection shadows are grounded in the finite $\mathsf{Hilbert}^{\uparrow}$ inner-product and projection rows.

Concretized in §124: Fourier projection ledgers are grounded in finite $\mathsf{Fourier}^{\uparrow}$ coefficient, probe, truncation, and provenance rows.

Concretized in §188: Conditional expectation is read through the finite $\mathsf{CondExp}^{\uparrow}$ random-variable readback, projection witness, and residual continuation rows.

Concretized in §289: Belief updates consume probability rows only through the finite $\mathsf{Belief}^{\uparrow}$ observation-update ledger.

Concretized in §298: Spectral measures contribute only finite projection-family and scalar-window rows to this bridge.

9462.28.12 Quantum and Relativistic Model Bridges

The probability-projection reading has two immediate physical neighbours. Quantum mechanics gives the direct Hilbert-projection model: state, observable, spectral projection, and scalar probability are all already visible as finite certificate surfaces. Relativistic geometry gives the complementary no-global-frame model: locality, rate bounds, metric-facing geometry, and curvature are read only through local ledgers and observer-stable transport. The two readings are not fused into a single physical theory; they share the same refusal of hidden global objects.

Definition 9462.977 (Born-rule bridge packet)**.** *A Born-rule bridge packet is a finite scientific bridge*

$$B_Q = (Q, O, M, r, L)$$

where Q is a displayed $\mathsf{QuantumState}^{\uparrow}$ *packet, O is a displayed* $\mathsf{Observable}^{\uparrow}$ *packet over the same Hilbert source, M is the* $\mathsf{SpectralMeasure}^{\uparrow}$ *projection-family row consumed by the observable, r is a* $\mathsf{Real}^{\uparrow}$ *scalar window, and L records the comparison*

$$r \sim_R \langle \psi, \Pi_A(E)\psi \rangle.$$

Here ψ is read only through the state ray of Q, A only through the self-adjoint observable endpoint of O, and $\Pi_A(E)$ only through the finite projection row of M.

Theorem 9462.978 (Born probability is a certified projection shadow)**.** *An accepted Born-rule bridge packet reads a quantum measurement probability as a finite Hilbert projection shadow. It does not expose a hidden wavefunction representative, host operator equality, unrestricted spectral theorem, unlogged measurement primitive, or ambient probability oracle.*

Proof. Project the packet $B_Q = (Q, O, M, r, L)$. The $\mathsf{QuantumState}^{\uparrow}$ row supplies the Hilbert source, unit-vector endpoint, projective endpoint, global-phase classifier, and provenance. The $\mathsf{Observable}^{\uparrow}$ row supplies the displayed self-adjoint endpoint and its expectation-facing row. The $\mathsf{SpectralMeasure}^{\uparrow}$ row supplies the event-to-projection ledger and finite scalar-window boundary.

The ledger L then records the only cross-surface comparison needed for the measurement read:

$$r \sim_R \langle \psi, \Pi_A(E)\psi \rangle.$$

Thus the public probability is the displayed scalar endpoint r classified with the Hilbert projection expression named by the bridge. Since no field of B_Q is a selected vector representative beyond the projective state row, a host operator equality, an unrestricted spectral-theorem object, an unlogged measurement operation, or a detached probability source, none of those objects is exported by the bridge. □

Corollary 9462.979 (Phase and density boundaries)**.** *Pure-state measurements consume the projective state classifier, not host vector equality. Mixed-state measurements consume the* $\mathsf{DensityMatrix}^{\uparrow}$ *trace-class, positivity, trace-one, and finite affine ledger rows before any trace-form scalar window is read.*

Proof. For a pure state, the carrier of $\mathsf{QuantumState}^{\uparrow}$ identifies public state content through the projective endpoint and global-phase classifier. A measurement bridge cannot distinguish two representatives that the public state row has already classified.

For a mixed state, the $\mathsf{DensityMatrix}^{\uparrow}$ carrier exposes only positivity, trace-class membership, trace-one readback, restricted classifier comparison, and finite affine-mixture ledger data over the Hilbert source. A trace-form measurement read therefore consumes those rows before it reaches the scalar window. Spectral decompositions, purifications, or hidden ensembles are not part of that public surface. □

Corollary 9462.980 (Quantum-channel preservation of the measurement interface)**.** *A* $\mathsf{QuantumChannel}^{\uparrow}$ *row preserves the type of data needed by the Born-rule bridge: a density-matrix input is sent to a density-matrix output, so downstream scalar windows remain measurement-interface rows rather than untyped state updates.*

Proof. The $\mathsf{QuantumChannel}^{\uparrow}$ packet carries a completely positive trace-preserving linear-map row between density-matrix spaces. Its public image theorem sends accepted density-matrix rows to accepted density-matrix rows. Consequently a post-channel measurement bridge still starts from the same public interface: Hilbert source, density carrier, observable row, spectral projection ledger, and $\mathsf{Real}^{\uparrow}$ scalar window. □

Definition 9462.981 (Relativistic pre-geometry bridge)**.** *A relativistic pre-geometry bridge is a finite model bridge whose displayed rows may include* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{LeviCivitaConnection}^{\uparrow}$, $\mathsf{CrossHistCausalCone}^{\uparrow}$, $\mathsf{LorentzFrameRate}^{\uparrow}$, *and* $\mathsf{RelationalPhysics}^{\uparrow}$ *packets. Its public content is the local geometric, curvature, causal, observer-symmetry, model-audit, and no-global-frame ledger carried by those packets.*

Theorem 9462.982 (Relativistic bridge is pre-geometry, not completed field equation)**.** *The current relativistic bridge supports a certificate pre-structure for local geometry, curvature-facing rows, causal cones, observer-symmetric rate readback, and relational physics. It does not close a Lorentzian metric, Ricci tensor, scalar curvature, Einstein tensor, stress-energy tensor, or Einstein field-equation certificate.*

Proof. The manifold, Riemannian-metric, connection, curvature, and Levi-Civita-connection packets expose local carrier, fibre, derivative, commutator, torsion-free, metric-compatibility, curvature-facing, transport, and provenance rows. These are geometry-facing certificate rows, not a Lorentzian spacetime object or a field equation.

The causal and relational packets expose finite multi-history locality, maximum-rate, observer-symmetry, no-global-frame refusal, invariant readback, physical-model audit, and admissible rate or refusal rows. Their public boundaries explicitly exclude global frames, synchronized clocks, ambient spacetime, metric tensors as primitive inputs, and detached physical truth.

No listed packet supplies a Lorentzian metric carrier, curvature contractions to Ricci and scalar curvature, an Einstein tensor row, a stress-energy tensor row, or an empirical field-equation bridge tying those rows together. The current claim is therefore the pre-geometry boundary stated above, not a completed general-relativity certificate. □

Definition 9462.983 (General-relativity coordination bridge)**.** *A general-relativity coordination bridge is a relativistic bridge packet whose displayed rows include observer-coordinate frames, local clock-budget reads, observer-locality cells, cross-history causal-cone rows, Lorentz-frame-rate rows, Minkowski-rate geometry rows, and a* $\mathsf{CurvedSpacetime}^{\uparrow}$ *curved-gluing row. Its public object is the compatibility of these rows under componentwise* hsame *transport, displayed* Cont *replay, provenance, and no-global-frame refusal.*

Theorem 9462.984 (General relativity is interface coordination)**.** *An accepted general-relativity coordination bridge reads the relativistic model as a minimal compatibility geometry among local clocks, local frames, causal cones, inter-Hist coordination rows, metric-facing distance-shape ledgers, and curvature or transition ledgers. It does not export a pre-given four-dimensional host container, a global observer frame, a shared external time scalar, a primitive metric tensor, or an observer-independent event point.*

Proof. Project the coordination bridge. The observer-coordinate frame and local-clock-budget rows supply observer-local chart and clock reads, not a global frame. The observer-locality-cell and cross-history causal-cone rows supply finite inter-Hist coordination and bounded causal comparison, not a host event object. The Lorentz-frame-rate and Minkowski-rate geometry rows supply observer-symmetric rate and distance-shape readback, not a primitive metric tensor. The $\mathsf{CurvedSpacetime}^{\uparrow}$ row then records chart transition, curvature, transport, replay, and provenance data over the same finite surface.

Thus the bridge exports only the compatibility of displayed interfaces. Any coordinate system anchored at one observer history is a gauge-facing interface choice; the invariant physical content is the classifier-stable transport of the rows across the displayed histories. Since the packet has no coordinate for a host spacetime container, shared clock, primitive tensor, or detached event point, none of those objects is exported. □

Corollary 9462.985 (Events and metrics are coordination readbacks)**.** *Within a general-relativity coordination bridge, an event is public only as a licensed inter-Hist coordination of local records, and a metric expression is public only as a bridge readback compressing local-clock, causal-cone, distance-shape, transition, and curvature ledgers.*

Proof. The locality-cell and causal-cone rows identify which local records may be read by the same downstream consumer. That identification is the public event surface; no point of an ambient manifold is present in the carrier.

The metric-facing read is likewise downstream from the displayed clock, rate, distance-shape, transition, and curvature rows. It can serve as a host geometric expression only after a bridge-preservation row relates that expression back to the finite packet. Hence both event and metric are coordination readbacks rather than primitives. □

Definition 9462.986 (Geometric forcing certificate)**.** *A geometric forcing certificate is a finite scientific bridge packet*

$$\mathcal{F} = (\Lambda, \mathcal{G}, \mathcal{P}, \Gamma, R, S, U, M, A)$$

where Λ *is an observation ledger,* $\mathcal{G}$ *is the declared candidate geometry class,* $\mathcal{P}$ *is the audited probe class,* Γ *is the candidate bridge,* R *records realization of* Λ *by* Γ *within the declared tolerance,* S *records the separation tests for candidates not observationally equivalent to* Γ*,* U *records the gauge or representation freedom left invisible to* $\mathcal{P}$*,* M *records minimality of the exported structure, and* A *records residual gaps and no-host-leak boundaries.*

Definition 9462.987 (Ledger forcing up to gauge)**.** *For a geometric forcing certificate, write*

$$\Lambda \Vdash_{\mathcal{G},\mathcal{P}} [\Gamma]$$

when Γ *realizes* Λ*, and every candidate* $\Delta \in \mathcal{G}$ *that also realizes* Λ *is observationally equivalent to* Γ *on all probes in* $\mathcal{P}$*, up to the gauge or representation freedom recorded by* U*.*

Theorem 9462.988 (Geometric forcing is not empty-kernel derivation)**.** *A geometric forcing certificate does not derive general relativity, quantum mechanics, or any physical geometry from the empty BEDC kernel. It derives a relative claim: given an observation ledger, an admissible candidate class, an audited probe class, realization rows, separation rows, gauge rows, minimality rows, and residual-gap rows, the surviving candidates collapse to one observational equivalence class.*

Proof. The forcing judgment in Definition 9462.987 is parameterized by Λ, $\mathcal{G}$, and $\mathcal{P}$. Without these fields there is no candidate comparison and no probe surface on which surviving candidates could be tested.

Assume the certificate fields are present. The realization row R admits Γ as a candidate compatible with the ledger. For any $\Delta \in \mathcal{G}$, if Δ fails realization, the ledger excludes it. If Δ realizes the ledger but is not observationally equivalent to Γ on $\mathcal{P}$, the separation row S supplies a displayed probe that distinguishes the two and records the conflict with Λ. Hence every surviving candidate is equivalent to Γ on the audited probes, modulo the gauge row U. This proves only the relative forcing claim, not an unconditional derivation of a host physical theory. □

Corollary 9462.989 (Forced geometry is a gauge class)**.** *The public output of a geometric forcing certificate is a gauge or representation equivalence class, not a privileged coordinate presentation.*

Proof. The row U records transformations that preserve all audited probe readouts. No ledger row can distinguish two presentations related by such a transformation. Therefore the certificate can export only the invariant class $[\Gamma]$, together with the residual gaps recorded by A. □

Corollary 9462.990 (Physical geometry as invariant compression)**.** *In this layer, physical geometry is the invariant compression of observation ledgers. A relativistic packet compresses clock, causal, lightlike, free-route, transition, and curvature ledgers into Lorentzian-facing geometry when those rows separate all non-equivalent alternatives in the declared class. A quantum packet compresses preparation, measurement, probability, interference, symmetry, composition, and update ledgers into Hilbert/projective/operator-facing geometry under the same forcing pattern.*

Proof. Apply Definition 9462.988 with the relevant observation ledger and candidate class. In the relativistic case, the admissible probes inspect causal order, local clocks, causal-cone or lightlike rows, route comparison, transition, and curvature data. In the quantum case, the admissible probes inspect preparation equivalence, measurement partitions, probability weights, interference, symmetries, composition, and update rows. When the corresponding separation row excludes all non-equivalent candidates, the exported structure is the unique invariant compression of those ledgers up to the recorded gauge or representation row. □

Definition 9462.991 (Constraint container)**.** *A constraint container for an observation ledger* Λ *is a derived bridge-facing packet*

$$\mathcal{K}_\Lambda = (Q, C, I, T, U, G, A)$$

where Q is the current observational quotient induced by the displayed probes, C is the surviving candidate class compatible with the ledger, I is the invariant readback stable across the surviving candidates, T is the transport row between local interfaces, U is the gauge or representation freedom still invisible to the probes, G is the residual gap surface, and A is the audit row recording source, classifier, witness, ledger, and boundary data. The packet is a container only as a stabilized closure of constraints; it is not a primitive host arena.

Theorem 9462.992 (Constraint closure becomes container)**.** *Suppose a refinement sequence of observation ledgers keeps adding displayed records and probes, so that the observational quotient is refined and the surviving candidate class is restricted. If the surviving class stabilizes to one gauge or representation class $[\Gamma]$, and the corresponding packet has stable invariants, transport across local interfaces, compression of the ledger with residual gaps exposed, prediction rows for further probes, and auditable source/classifier/witness/boundary rows, then the stabilized packet is licensed as a constraint container.*

Proof. Each displayed record adds a constraint to the ledger. A new probe can only refine the observational quotient by making distinctions that were previously unavailable to the packet; it cannot by itself add an unrecorded global arena. Likewise, a new compatibility row can only restrict the declared candidate class unless it is accompanied by a new residual-gap row.

If every non-equivalent candidate is eventually excluded by displayed separation evidence, the surviving class is $[\Gamma]$ in the sense of Definition 9462.987. The additional stability, transport, compression, prediction, and audit rows state exactly that this class can be reused as a public interface for later reads. That reuse is the container role. Since the role is obtained only after refinement and candidate restriction, the container is the stabilized closure of observational constraints rather than a premise into which the records were placed. □

Corollary 9462.993 (Containers delimit future extensions)**.** *The container role of a stabilized geometric packet is its restriction of admissible future extensions. It says which further records, routes, probes, and bridge reads remain compatible with the current ledger, and which have already been excluded by the boundary rows.*

Proof. The stabilized packet contains the surviving candidate class, invariant readback, transport, gauge, residual-gap, prediction, and audit rows. A future extension is admissible only if it preserves those rows or records a declared gap or boundary pressure. Thus the container is not a box holding events; it is the public shape of the still-admissible continuation space after the ledger has excluded incompatible alternatives. □

Corollary 9462.994 (Relativistic and quantum containers are derived)**.** *The spacetime-facing container in a relativistic bridge and the Hilbert-facing container in a quantum bridge are derived interfaces. The former is licensed only after clock, causal, lightlike, route, transition, and curvature rows stabilize; the latter only after preparation, measurement, probability, interference, symmetry, composition, and update rows stabilize. Neither is a BEDC kernel primitive.*

Proof. Apply Definition 9462.992 to the two candidate classes named by Definition 9462.990. In the relativistic case, the stable rows are the local physical ledgers that support Lorentzian-facing readback. In the quantum case, the stable rows are the operational probability ledgers that support Hilbert/projective/operator readback. In both cases the public container appears only after the forcing and audit rows are present. □

Definition 9462.995 (Route-delay bridge packet)**.** *A route-delay bridge packet is a finite records-side comparison*

$$\mathcal{D} = (R_1, R_0, E, K, \Delta, A)$$

where R_1 and R_0 are two displayed routes with the same emitted and received locality-cell endpoints, E is the endpoint-equivalence certificate, K is a shared local-clock and accumulation-edge certificate, Δ is the route-pair clock-read token, and A records the refusal of a global clock, host event point, ambient metric, or unlisted host theorem. The delay row is first a comparison of two certified route reads, not an external real-valued time difference.

Theorem 9462.996 (Route delay is not host proper time)**.** *An accepted route-delay bridge packet does not identify BEDC clock reads with relativistic proper time, coordinate time, Shapiro delay, or Wigner–Smith scattering delay unless the relevant host correspondence is present as a displayed bridge-preservation row.*

Proof. The packet $\mathcal{D}$ supplies only two routes, endpoint equivalence, a shared local-clock certificate, an accumulation-edge read, and an audit row. These rows make the BEDC delay token well formed as a records-side comparison. Proper time, coordinate time, Shapiro delay, and Wigner–Smith delay are host-side readings involving additional metric, scattering, or asymptotic-state data. Since the audit row refuses host leakage and unlisted host theorems, such readings can enter only by a bridge-preservation row that relates the host expression back to the displayed route-pair token. □

Corollary 9462.997 (Scattering delay is a bridge reading)**.** *A scattering-delay expression may be used in this layer only as a host-side reading of a route-delay packet whose incoming route, outgoing route, free reference route, interaction ledger, clock certificate, and no-leak boundary are displayed.*

Proof. Scattering delay compares an interacting route with a free reference route. On the BEDC side, such a comparison is meaningful only after same-endpoint evidence and a shared clock certificate have been supplied by Definition 9462.995. The scattering formula belongs to the host model; the BEDC export is the finite route-pair token together with the bridge row that preserves it. □

Definition 9462.998 (Bridge-alignment learner)**.** *A bridge-alignment learner is a derived machine-learning packet that reads a records-side family $\mathcal{R}$, encodes it into a displayed representation Z_i, and applies a predictor or adapter $A_{\hat{\theta}} : Z_i \to Y$ to a declared readout family Y. Its effect claim is accepted only through the source, split, prediction, loss, NoLeak, stability, failure-surface, and NameCert rows of the machine-learning effect certificate.*

Definition 9462.999 (Readout alignment)**.** *Representations Z_i and Z_j are readout-aligned over a readout family Y when there are accepted bridge-alignment learners $A_i : Z_i \to Y$ and $A_j : Z_j \to Y$ whose certified errors lie inside the same declared budget. This is an alignment of observable reads, not an identity of the representation carriers.*

Theorem 9462.1000 (Readout alignment is not latent equivalence)**.** *If metric-facing, scattering-facing, and information-clock-facing representations all predict the same route-delay readout within a declared budget, BEDC may record readout alignment. It may not identify the latent representations without an additional bridge certificate.*

Proof. The accepted learners prove only that the composites $A_i : Z_i \to Y$ preserve the declared readout within the stated budget. Many non-equivalent representation carriers can admit predictors with the same readout behavior. Latent equivalence requires more data: translation in both

directions, classifier preservation, return-comparison witnesses, theorem or invariant preservation where claimed, and a NoLeak ledger for the translation. Those rows are not supplied by readout alignment alone. □

Corollary 9462.1001 (Bridge learning is evidence, not bridge closure)**.** *A machine-learning adapter from a metric, scattering, or information representation to a delay readout is evidence for a bridge candidate, not a completed standard bridge.*

Proof. The adapter may certify useful prediction from one representation to the declared readout. A standard bridge additionally requires displayed encoders and decoders, preservation rows, return-comparison witnesses, NoLeak boundaries, and failure-surface accounting between the interfaces being bridged. Prediction effectiveness is therefore a records-to-readout certificate, not representation-level bridge closure. □

Theorem 9462.1002 (Common physical bridge discipline)**.** *Quantum and relativistic model bridges share a certificate discipline: the world is not closed as a global object; only finite public projection, geometry, causal, invariant, empirical, and failure ledgers are exported.*

Proof. In the quantum bridge, unobserved state detail is not exported as a hidden complete object. Measurement passes through a state row, an observable row, a spectral projection row, and a scalar window.

In the relativistic bridge, global frame data is not exported as a primitive object. Physical structure passes through local geometry rows, cross-history causal rows, observer-symmetric rate rows, invariant readback, and model-audit ledgers.

Both bridges therefore replace hidden totality by finite public certificates. They differ in geometry and physical interpretation, but they obey the same BEDC rule: a scientific bridge exports only displayed rows and their classifier-stable transport, not an unlogged global world. □

Concretized in §217: Quantum-state packets supply the Hilbert unit-vector, projective phase, classifier, and provenance rows consumed by the Born-rule bridge.

Concretized in §218: Observable packets supply the self-adjoint endpoint, spectral row, expectation row, and no-host-operator-equality boundary.

Concretized in §298: Spectral-measure packets supply the event-to-projection ledger and finite scalar-window rows used by quantum measurement reads.

Concretized in §220: Density-matrix packets supply positive trace-class trace-one rows and finite affine-mixture ledgers for mixed-state measurement reads.

Concretized in §221: Quantum-channel packets preserve density-matrix measurement interfaces through CPTP ledger rows.

Concretized in §96: Manifold packets supply the local point, chart, and coordinate-transport substrate for relativistic pre-geometry.

Concretized in §133: Riemannian-metric packets supply fibrewise inner-product rows without closing Lorentzian spacetime.

Concretized in §134: Connection packets supply finite covariant-derivative rows consumed by curvature-facing geometry.

Concretized in §135: Curvature packets supply finite commutator-boundary and transport rows, not a completed field-equation object.

Concretized in §3638: Levi-Civita packets tie Riemannian metric, torsion-free connection, metric compatibility, and curvature-facing rows.

Concretized in §1112: Cross-history causal-cone packets supply finite causal route, maximum-rate, observer-symmetry, and no-global-frame refusal rows.

Concretized in §617: Lorentz-frame-rate packets supply observer-symmetric maximum-rate rows while excluding Lorentz geometry as a primitive input.

Concretized in §1275: Relational-physics packets supply finite observer-history, locality, invariant, audit, rate-or-refusal, and no-global-spacetime boundaries.

9462.28.13 Yang–Mills mass-gap bridge

The Yang–Mills mass-gap route is admissible in this scientific bridge layer only as a conditional certificate chain. The finite strong-coupling lattice surface may close inside a compact group integral and transfer-matrix packet. The continuum claim needs an additional renormalization bridge that preserves the physical Hilbert interface, Wilson packet readbacks, and every ultraviolet, infrared, gauge, topological, and soft-mode residual ledger. Without that bridge, the finite lattice theorem remains a regulated theorem and not a continuum mass-gap certificate.

Definition 9462.1003 (Yang–Mills bridge packet)**.** *A Yang–Mills bridge packet is a finite scientific bridge record*

$$\mathcal{Y} = (G, \mathcal{H}, \Omega, H, \mathcal{W}_{\mathrm{fin}}, L, A)$$

where G is a displayed compact simple gauge group row, $\mathcal{H}$ is the gauge-invariant physical Hilbert carrier, Ω is the vacuum row, $H \geq 0$ is the Hamiltonian readback with $H\Omega = 0$, $\mathcal{W}_{\mathrm{fin}}$ is the finite Wilson-loop or spin-network packet surface, L is the ledger for gauge quotient, ultraviolet cutoff, infrared volume, topological sector, zero mode, and soft residual data, and A is the audit row recording standard-bridge scope and no-host-leak boundaries.

Definition 9462.1004 (Finite Wilson packet correlation)**.** *For a finite Wilson packet*

$$O = \sum_{\alpha \in I} c_\alpha W_{\rho_\alpha, C_\alpha}$$

over a finite index set I, write

$$v_O = O\Omega - \langle \Omega, O\Omega \rangle \Omega$$

and

$$C_O(t) = \langle v_O, e^{-tH} v_O \rangle.$$

The packet is public only through the displayed loop, representation, coefficient, vacuum-subtraction, correlation, ledger, and provenance rows.

Definition 9462.1005 (Strong-coupling lattice gap certificate)**.** *A strong-coupling lattice gap certificate for a finite lattice Λ consists of compact edge variables $U_e \in G$, plaquette holonomy rows U_p, a Wilson action row, a character-expansion row*

$$e^{\beta \operatorname{Re} \chi_{\mathrm{fund}}(g)} = c_0(\beta) \left(1 + \sum_{\rho \neq \mathbf{1}} a_\rho(\beta) \chi_\rho(g) \right),$$

a polymer ledger with

$$\kappa(\beta) = \sum_{\rho \neq \mathbf{1}} d_\rho |a_\rho(\beta)| < 1,$$

and a transfer-matrix row. Its strong-coupling bound supplies constants $m_{\Lambda,\beta} > 0$ *and* K_O *such that every non-vacuum finite Wilson packet satisfies*

$$|C_O(t)| \leq K_O e^{-m_{\Lambda,\beta} t}.$$

Theorem 9462.1006 (Finite strong-coupling packet decay)**.** *If the character-expansion ledger of Definition 9462.1005 satisfies the displayed small-parameter bound strongly enough that the connected polymer expansion is absolutely convergent, then finite Wilson packet correlations decay exponentially on that regulated lattice surface.*

Proof. Every connected non-vacuum polymer contributing to a connected Wilson packet correlation contains a non-trivial representation row on each plaquette in its support. The coefficient ledger bounds the contribution of each such row by the strong-coupling parameter, while finite lattice connectivity supplies only a finite combinatorial factor per plaquette. Hence the total weight of a connected polymer with support size $|\Gamma|$ is bounded by

$$|w(\Gamma)| \leq C^{|\Gamma|} \kappa(\beta)^{|\Gamma|}.$$

When $C\kappa(\beta) < 1$, the polymer sum is absolutely convergent and separation in Euclidean time forces an exponential penalty for every connected surface linking the two Wilson packet reads. The transfer-matrix row reads this Euclidean decay as a non-vacuum transfer spectral gap on the finite regulated surface. □

Definition 9462.1007 (Wilson packet Hilbert gap certificate)**.** *A Wilson packet Hilbert gap certificate for Definition 9462.1003 consists of a number* $m > 0$ *and two displayed rows:*

- *the span of vacuum-orthogonal finite Wilson packet vectors* v_O *is dense in* $\Omega^\perp$*;*
- *for every such* v_O *there is a constant* K_O *with*

$$\langle v_O, e^{-tH} v_O \rangle \leq K_O e^{-mt}$$

  for all displayed Euclidean times $t \geq 0$*.*

Theorem 9462.1008 (Wilson decay and the physical Hilbert spectral gap)**.** *For a Yang–Mills bridge packet with self-adjoint non-negative Hamiltonian, unique vacuum row, and the density row of Definition 9462.1007, the following conditions are equivalent for a fixed* $m > 0$*.*

- *The restriction of* H *to* $\Omega^\perp$ *has spectrum contained in* $[m, \infty)$*.*
- *Every vacuum-orthogonal finite Wilson packet correlation has a spectral measure supported in* $[m, \infty)$*, equivalently it satisfies a uniform exponential decay certificate with exponent* m*.*

Proof. If the spectrum of H on $\Omega^\perp$ is contained in $[m, \infty)$, then the spectral theorem gives

$$\langle v, e^{-tH} v \rangle = \int_{[m,\infty)} e^{-tE} \, d\mu_v(E) \leq e^{-mt} \|v\|^2$$

for every $v \in \Omega^\perp$, hence in particular for finite Wilson packet vectors.

Conversely, suppose the displayed exponential certificate holds for each finite Wilson packet vector. For any $\varepsilon > 0$ and such a vector v, let μ_v be its spectral measure. The bound

$$\mu_v([0, m-\varepsilon])e^{-t(m-\varepsilon)} \leq \int e^{-tE}\, d\mu_v(E) \leq K_v e^{-mt}$$

holds for all $t \geq 0$. Letting t grow gives

$$\mu_v([0, m-\varepsilon]) = 0.$$

Thus every finite Wilson packet vector is annihilated by the spectral projection of H onto $[0, m-\varepsilon]$. Since those vectors are dense in $\Omega^\perp$ and spectral projections are bounded, the same projection annihilates all of $\Omega^\perp$. This holds for every $\varepsilon > 0$, so no non-vacuum spectral row lies below m. □

Definition 9462.1009 (Soft residual ledger)**.** *A soft residual below a proposed mass threshold m is a non-vacuum Hilbert state or approximating state family whose spectral mass meets $[0, m)$. It is ledgered when either some finite Wilson packet vector has nonzero projection against it, or the residual is recorded in the infrared, ultraviolet, renormalization, packet-complexity, topological, gauge, or zero-mode tail row of L.*

Definition 9462.1010 (Wilson packet complexity filtration)**.** *A Wilson packet complexity filtration for Definition 9462.1003 is an increasing family*

$$\mathcal{V}_1 \subseteq \mathcal{V}_2 \subseteq \cdots \subseteq \Omega^\perp$$

where $\mathcal{V}_M$ is generated by vacuum-orthogonal finite Wilson or spin-network packet vectors whose displayed loop support, representation weight, and coefficient-description ledgers have total complexity at most M. The filtration has a Peter–Weyl completeness row when

$$\overline{\bigcup_{M \geq 1} \mathcal{V}_M} = \Omega^\perp.$$

Each fixed $\mathcal{V}_M$ is a fixed packet-complexity surface. No conclusion about a growing sequence of packet complexities is licensed unless its tail ledger is also displayed.

Definition 9462.1011 (Complexity tail ledger)**.** *Given the filtration of Definition 9462.1010, a soft residual family $\psi_n \in \Omega^\perp$ is in the packet-complexity tail when*

$$\|P_M \psi_n\| \to 0$$

for every fixed M, where P_M is the projection onto the displayed fixed complexity surface. The corresponding ledger row records that detecting the residual requires packet complexity tending beyond every fixed stratum; it is not absence of a residual.

Theorem 9462.1012 (Soft residual fixed-packet or complexity-tail dichotomy)**.** *Let $\psi_n \in \Omega^\perp$ be a norm-one soft residual family for a Yang–Mills bridge packet equipped with a Wilson packet complexity filtration. Then either some fixed complexity surface has*

$$\limsup_n \|P_M \psi_n\| > 0,$$

or the family lies in the packet-complexity tail ledger of Definition 9462.1011.

Proof. This is the exhaustion dichotomy for an increasing filtration. If some fixed M has positive limsup projection, the residual is visible at that fixed packet-complexity surface. If no such M exists, then by definition $\|P_M\psi_n\| \to 0$ for every fixed M, which is exactly the packet-complexity tail condition. □

Definition 9462.1013 (Filtered uniform Wilson gap certificate)**.** *A filtered uniform Wilson gap certificate is a Wilson packet Hilbert gap certificate together with a complexity filtration and a number $m_0 > 0$ such that every vacuum-orthogonal finite Wilson packet vector v_O has*

$$\langle v_O, e^{-tH} v_O\rangle \le K_O e^{-m_0 t}$$

for all displayed Euclidean times $t \ge 0$. The constant K_O may depend on the packet, but the exponent m_0 is independent of O, of the fixed complexity stratum, of the loop-support ledger, and of the representation ledger.

Definition 9462.1014 (Block tube area-law certificate)**.** *A block tube area-law certificate is a fixed physical block-scale bridge row. It consists of a block scale $\ell_* > 0$, a blocked gauge graph $\mathcal{L}_{\ell_*}$, a class of blocked finite Wilson or spin-network packets on that graph, and a connected-tube expansion for every correlation $C_O(t)$. The certificate supplies constants*

$$\sigma > \kappa, \qquad c_* > 0$$

such that every connected tube Σ joining $O(0)$ to $O(t)$ has weight and area rows

$$|w(\Sigma)| \le e^{-\sigma|\Sigma|}$$

and

$$|\Sigma| \ge c_* t - b_O,$$

where b_O may depend on the displayed packet support but c_* does not. It also supplies an entropy row

$$\#\{\Sigma : |\Sigma| = n,\ \Sigma : O(0) \leftrightarrow O(t)\} \le A_O e^{\kappa n}.$$

All constants are read at the fixed block scale. The certificate makes no claim for changing block scales or growing packet windows unless those changes are recorded in the tail ledger.

Theorem 9462.1015 (Block tube area law gives filtered uniform Wilson decay)**.** *If a Yang–Mills bridge packet has a block tube area-law certificate, then its blocked finite Wilson packet correlations satisfy a filtered uniform Wilson gap certificate with exponent*

$$m_0 = (\sigma - \kappa)c_* > 0.$$

Proof. The connected-tube expansion bounds the connected correlation by the sum of absolute tube weights:

$$|C_O(t)| \le \sum_{\Sigma : O(0)\leftrightarrow O(t)} |w(\Sigma)|.$$

Group the tubes by area $n = |\Sigma|$. The tube lower bound and entropy row give

$$|C_O(t)| \le \sum_{n \ge c_* t - b_O} A_O e^{\kappa n} e^{-\sigma n}.$$

Since $\sigma > \kappa$, this is a convergent geometric tail:

$$|C_O(t)| \le A'_O e^{-(\sigma-\kappa)(c_* t - b_O)}.$$

Absorbing $e^{(\sigma-\kappa)b_O}$ into the packet-dependent constant gives

$$|C_O(t)| \le K_O e^{-(\sigma-\kappa)c_* t}.$$

Thus the exponent is $m_0 = (\sigma - \kappa)c_*$ and is independent of the packet, its complexity stratum, its support ledger, and its representation ledger. This is exactly the filtered uniform Wilson gap certificate. □

Corollary 9462.1016 (Block tube area law gives the fixed-sector gap)**.** *Assume the physical Hilbert construction, Wilson packet complexity filtration, Peter–Weyl completeness row, ledger-exhaustiveness row, and closedness boundary of Definition 9462.1019. A block tube area-law certificate then forces the non-vacuum physical Hilbert spectrum in the declared fixed block sector to be contained in* $[m_0, \infty)$*, with* $m_0 = (\sigma - \kappa)c_*$*.*

Proof. By Definition 9462.1015, the block tube certificate supplies the filtered uniform Wilson gap certificate with exponent m_0. The filtered gap theorem then excludes every fixed packet soft residual below m_0. Any residual escaping all fixed packet-complexity surfaces must enter the packet-complexity or physical tail ledger, and the closedness boundary rejects hidden unfinished tail data. □

Theorem 9462.1017 (Filtered gap excludes fixed-packet soft residuals)**.** *If a filtered uniform Wilson gap certificate with exponent* $m_0 > 0$ *is present, no soft residual can have nonzero spectral address in a fixed finite Wilson packet.*

Proof. Suppose a fixed packet vector v_O detects a soft residual. Then the spectral measure μ_O of v_O for H has positive mass in arbitrarily small non-vacuum energy windows. In particular, for some $0 < \varepsilon < m_0$ the measure of $[0, \varepsilon]$ is nonzero. The correlation then satisfies

$$\langle v_O, e^{-tH} v_O \rangle \ge \mu_O([0, \varepsilon]) e^{-\varepsilon t}.$$

Multiplying by $e^{m_0 t}$ contradicts the displayed bound $K_O e^{-m_0 t}$ as t grows. Hence a fixed packet address is impossible under the filtered uniform gap certificate. □

Theorem 9462.1018 (Gapless spectrum forces visible or ledgered residual)**.** *Under the density, complexity-filtration, and ledger-exhaustiveness rows of a Yang–Mills bridge packet, failure of the Hilbert spectral gap cannot disappear silently. A gapless non-vacuum spectral residual is either detected by a fixed finite Wilson packet correlation, appears on a fixed packet-complexity surface, or belongs to the displayed packet-complexity, infrared, ultraviolet, renormalization, topological, gauge, or zero-mode tail ledger.*

Proof. Apply Definition 9462.1012. If a fixed complexity projection has positive limsup, the displayed fixed stratum contains a finite packet witness after choosing the corresponding packet row in that stratum; the residual changes the spectral measure of that packet below every proposed positive threshold. If every fixed complexity projection vanishes, the residual is not absent; it has escaped the fixed packet surfaces and must be recorded as packet-complexity tail. Any further ultraviolet, infrared, renormalization, topological, gauge, or zero-mode loss is likewise a ledger row. Therefore an unledgered soft residual is incompatible with the packet assumptions. □

Theorem 9462.1019 (Filtered Yang–Mills gap theorem)**.** *Let a Yang–Mills bridge packet have a physical Hilbert construction, compact gauge Wilson packet rows, Peter–Weyl completeness for the packet-complexity filtration, a filtered uniform Wilson gap certificate with exponent* $m_0 > 0$*,*

and a ledger-exhaustiveness row saying that every residual escaping all fixed packet-complexity surfaces enters the packet-complexity or physical tail ledger. If the accepted object is closed against unfinished tail ledgers, then its physical Hilbert spectrum on $\Omega^\perp$ is contained in $[m_0, \infty)$.

Proof. Assume instead that the non-vacuum spectrum meets energies below m_0. Choose a norm-one approximating soft residual family in $\Omega^\perp$ with energy tending below the proposed gap. By Definition 9462.1012, this family either has a fixed packet-complexity address or lies in the packet-complexity tail.

The first case is excluded by Definition 9462.1017. In the second case, ledger exhaustiveness requires a packet-complexity or physical tail row. Such a row is unfinished residual data unless it is resolved by the bridge, and closedness forbids carrying it as hidden content. Thus both cases contradict the accepted closed packet assumptions. Therefore the non-vacuum spectrum is contained in $[m_0, \infty)$. □

Definition 9462.1020 (Filtered renormalization gap stability certificate)**.** *A filtered renormalization gap stability certificate is a bridge row from a lattice tower*

$$\mathcal{Y}_0 \leftarrow \mathcal{Y}_1 \leftarrow \mathcal{Y}_2 \leftarrow \cdots \longrightarrow \mathcal{Y}_\infty$$

to a continuum Yang–Mills bridge packet. It contains:

- *scale-time rows for blocking and continuum refinement;*
- *contractive or isometric Hilbert transport rows on the physical vacuum-orthogonal surfaces;*
- *convergence of finite Wilson packet correlations from the lattice tower to the continuum packet;*
- *a uniform lower bound $m_0 > 0$ for the lattice physical gaps in every declared fixed Wilson packet-complexity sector, with exponent independent of the sector;*
- *a complete ledger for ultraviolet, infrared, renormalization, gauge, packet-complexity, topological, and zero-mode residuals.*

Theorem 9462.1021 (Renormalization bridge preserves the mass gap)**.** *Assume a lattice tower has strong-coupling finite Wilson packet decay with a uniform exponent $m_0 > 0$ independent of packet complexity, and assume the tower is connected to a continuum Yang–Mills bridge packet by a filtered renormalization gap stability certificate. Then the continuum packet has a physical Hilbert spectral gap at least m_0 on the declared Wilson packet sector, unless a packet-complexity or physical tail ledger remains open.*

Proof. Choose a continuum finite Wilson packet vector. By the correlation convergence row of Definition 9462.1020, it is the limit of lattice Wilson packet vectors in the declared sector. The uniform lattice gap gives the bound

$$C_{O,n}(t) \leq K_{O,n} e^{-m_0 t}$$

with the same exponent m_0 along the tower, after the packet normalization specified by the bridge row. Passing to the continuum correlation yields the same spectral-support exclusion below m_0 for the continuum packet.

The density and filtration rows then invoke Definition 9462.1019: finite Wilson packet decay with exponent m_0 forces the physical Hilbert spectrum on the vacuum-orthogonal declared sector to avoid $[0, m_0)$, except for residuals that escape every fixed packet-complexity surface. If such a residual tried to appear during the limit, Definition 9462.1018 would place it in the displayed packet-complexity or physical tail ledger. The filtered stability certificate therefore blocks an unrecorded loss of the mass gap and leaves only explicitly ledgered open tail data. □

Theorem 9462.1022 (Yang–Mills mass-gap bridge boundary)**.** *The Yang–Mills mass-gap route in this layer has exactly two admissible positive surfaces: the regulated finite strong-coupling packet theorem, and the conditional continuum theorem obtained after a renormalization gap stability certificate is displayed. It does not claim an unconditional continuum construction, a continuum gap without the stability certificate, a gap outside the declared Wilson packet sector, or a host-theory theorem not listed in the bridge preservation row.*

Proof. Definition 9462.1006 closes only the finite regulated strong-coupling surface. The passage to a continuum Hilbert mass gap additionally uses the density, correlation-convergence, Hilbert transport, packet-complexity filtration, uniform-gap, and tail-ledger rows of Definition 9462.1020. Without those rows the bridge has no public path from lattice transfer-matrix decay to continuum spectral support. In particular, the statement that every fixed finite packet has some gap does not imply a uniform physical mass gap; the exponent must be independent of packet complexity, or the escaping modes must be recorded in the packet-complexity or physical tail ledger. Standard bridge discipline also restricts the export to the declared theorem-preservation list and no-host-leak boundary. The stated positive surfaces are therefore the only ones licensed here. □

9462.28.14 Yang–Mills tube geometry certificates

The block tube area-law certificate of Definition 9462.1014 separates into two finite combinatorial geometry rows and one physical weight row. The geometry rows are internal to the fixed block lattice: a connected tube joining two time-separated Wilson packet supports must contain a linearly long plaquette chain, and the number of connected plaquette sets of a given area grows at most exponentially. The physical row is the area decay of tube weights. Only the physical row carries the confinement content; the two geometry rows are finite block-scale certificates.

Definition 9462.1023 (Blocked Wilson support)**.** *Fix a physical block scale $\ell_* > 0$ and let $\mathcal{L}_{\ell_*}$ be the corresponding four-dimensional block lattice with integer time coordinate τ. A blocked Wilson packet O has finite support* $\mathrm{supp}(O)$ *in the cells and edges of this lattice. Its time diameter is*

$$d_t(O) = \max_{\xi,\eta\in\mathrm{supp}(O)} |\tau(\xi) - \tau(\eta)|.$$

The translate $O(T)$ is obtained by shifting the support by T block time steps; the corresponding physical time is $t = T\ell_*$.

Definition 9462.1024 (Plaquette tube)**.** *A plaquette tube Σ is a finite set of block plaquettes equipped with the plaquette adjacency inherited from the fixed lattice. It joins $O(0)$ to $O(T)$ when its adjacency graph has a connected component meeting a fixed finite neighbourhood of* $\mathrm{supp}(O(0))$ *and a fixed finite neighbourhood of* $\mathrm{supp}(O(T))$*. Its area is the plaquette count*

$$|\Sigma| = \#\{p : p \in \Sigma\}.$$

In physical units the area is $\ell_*^2|\Sigma|$.

Theorem 9462.1025 (Tube area lower bound)**.** *For each fixed blocked Wilson packet O there are constants $c_* > 0$ and $b_O < \infty$ such that every plaquette tube joining $O(0)$ to $O(T)$ satisfies*

$$|\Sigma| \geq c_* T - b_O.$$

Equivalently, in physical time $t = T\ell_*$,

$$|\Sigma| \geq \frac{c_*}{\ell_*} t - b_O.$$

Proof. Since the tube joins the two packet supports, its adjacency graph contains a plaquette chain

$$\Pi_0, \Pi_1, \ldots, \Pi_L$$

from the fixed neighbourhood of $O(0)$ to the fixed neighbourhood of $O(T)$. In the fixed block lattice, adjacent plaquettes can change the time coordinate by at most a uniform finite constant D_t. The two packet supports themselves occupy only the time diameter $d_t(O)$, together with the fixed neighbourhood thickness. Hence the chain must cross at least

$$T - 2d_t(O) - B_O$$

units of time distance, where B_O records the fixed neighbourhood and support boundary. Therefore

$$L \geq D_t^{-1}T - B'_O.$$

Every plaquette in the chain belongs to Σ, so

$$|\Sigma| \geq L + 1 \geq c_* T - b_O$$

after setting $c_* = D_t^{-1}$ and absorbing the finite packet boundary into b_O. □

Definition 9462.1026 (Tube counting function)**.** *For a fixed blocked packet O, let $N_O(n, T)$ be the number of connected plaquette tubes of area n that join $O(0)$ to $O(T)$. The count is taken inside the fixed block lattice and with the fixed plaquette adjacency of Definition 9462.1024.*

Theorem 9462.1027 (Tube entropy bound)**.** *For each fixed blocked Wilson packet O there are constants $A_O < \infty$ and $\kappa < \infty$ such that for all n and T,*

$$N_O(n, T) \leq A_O e^{\kappa n}.$$

Proof. The block dimension and plaquette adjacency are fixed, so every plaquette has at most $D < \infty$ adjacent plaquettes. A connected tube of area n meeting the finite neighbourhood of $O(0)$ can be explored by a depth-first traversal of a spanning tree for its adjacency component. The starting plaquette has at most $A_O^{(0)}$ choices, because the packet support is finite. The traversal has length at most $2n$ and each step has at most D choices. Thus the number of such connected plaquette sets is bounded by

$$A_O^{(0)} D^{2n}.$$

Taking $A_O = A_O^{(0)}$ and $\kappa = 2 \log D$ gives the claimed exponential entropy bound. The additional requirement that the tube also meets $O(T)$ only decreases the count. □

Definition 9462.1028 (Tube weight area decay)**.** *Tube weight area decay at the fixed block scale is the physical weight row asserting that there is a $\sigma > 0$ such that every connected tube contribution satisfies*

$$|w(\Sigma)| \leq e^{-\sigma|\Sigma|}.$$

In a strong-coupling polymer expansion this row is supplied by the non-trivial plaquette character-expansion parameter. If the effective plaquette excitation weight is $\eta < 1$, the row has $\sigma = |\log \eta|$ after the fixed normalization of the tube weights.

Definition 9462.1029 (Strong-coupling character ledger)**.** *On a finite block lattice, let the normalized plaquette weight be*

$$\widetilde{K}_\beta(g) = \frac{\exp(\beta \operatorname{Re} \chi_{\mathrm{fund}}(g))}{\int_G \exp(\beta \operatorname{Re} \chi_{\mathrm{fund}}(h))\, dh}.$$

Its character ledger is the expansion

$$\widetilde{K}_\beta(g) = 1 + \sum_{\rho \neq \mathbf{1}} a_\rho(\beta)\chi_\rho(g),$$

together with the summability row

$$\eta(\beta) = \sum_{\rho \neq \mathbf{1}} d_\rho |a_\rho(\beta)| < \infty.$$

If a finite representation cutoff is used, the cutoff and the omitted tail are recorded as part of the same ledger.

Lemma 9462.1030 (Strong-coupling small parameter)**.** *For the character ledger of Definition 9462.1029, the non-trivial coefficient norm satisfies*

$$\eta(\beta) \to 0$$

as $\beta \to 0$, within the displayed summability or truncation ledger.

Proof. At $\beta = 0$ the normalized plaquette weight is the constant function 1, so all non-trivial character coefficients vanish. For each fixed irreducible representation, $a_\rho(\beta)$ varies analytically with β and therefore tends to zero. Under a finite representation cutoff this proves the claim immediately. Under the complete character expansion, the ledger supplies the uniform summability needed to pass the limit through the tail. Hence the total non-trivial norm tends to zero. □

Lemma 9462.1031 (Single plaquette excitation bound)**.** *For every group element $g \in G$,*

$$\left| \sum_{\rho \neq \mathbf{1}} a_\rho(\beta)\chi_\rho(g) \right| \leq \eta(\beta).$$

Proof. Every finite-dimensional representation of the compact group G may be unitarized, so its character satisfies $|\chi_\rho(g)| \leq d_\rho$. Thus

$$\left| \sum_{\rho \neq \mathbf{1}} a_\rho(\beta)\chi_\rho(g) \right| \leq \sum_{\rho \neq \mathbf{1}} |a_\rho(\beta)|\, |\chi_\rho(g)| \leq \sum_{\rho \neq \mathbf{1}} d_\rho |a_\rho(\beta)| = \eta(\beta).$$

□

Definition 9462.1032 (Local contraction ledger)**.** *The local contraction ledger for a fixed block lattice and fixed expansion rule records a constant $C_{\mathrm{loc}} < \infty$ bounding all local Haar integral contractions, intertwiner multiplicities, orientation choices, and nearest-neighbour representation couplings produced by one plaquette step of a connected tube. It asserts that the total local contraction factor for a tube Σ is at most $C_{\mathrm{loc}}^{|\Sigma|}$.*

Theorem 9462.1033 (Strong-coupling tube weight area decay)**.** *Assume the strong-coupling character ledger and the local contraction ledger. For every connected tube Σ,*

$$|w(\Sigma)| \leq (C_{\mathrm{loc}}\eta(\beta))^{|\Sigma|}.$$

If $C_{\mathrm{loc}}\eta(\beta) < 1$, then tube weight area decay holds with

$$\sigma(\beta) = -\log(C_{\mathrm{loc}}\eta(\beta)) > 0.$$

Proof. A connected tube of area $|\Sigma|$ contains exactly that many non-trivial plaquette excitation rows. By Definition 9462.1031, each such plaquette contributes at most $\eta(\beta)$ in absolute value before local edge integrations are accounted for. The local contraction ledger bounds all Haar-integration and intertwiner factors by $C_{\mathrm{loc}}^{|\Sigma|}$. Hence

$$|w(\Sigma)| \leq C_{\mathrm{loc}}^{|\Sigma|}\eta(\beta)^{|\Sigma|} = (C_{\mathrm{loc}}\eta(\beta))^{|\Sigma|}.$$

When $C_{\mathrm{loc}}\eta(\beta) < 1$, this is $e^{-\sigma(\beta)|\Sigma|}$ with the stated value of $\sigma(\beta)$. □

Theorem 9462.1034 (Strong coupling dominates tube entropy)**.** *For the entropy constant κ of Definition 9462.1027, there is a strong-coupling range in which*

$$\sigma(\beta) > \kappa.$$

Proof. By Definition 9462.1030, $\eta(\beta) \to 0$ as $\beta \to 0$. Therefore $C_{\mathrm{loc}}\eta(\beta) \to 0$, and

$$\sigma(\beta) = -\log(C_{\mathrm{loc}}\eta(\beta)) \to +\infty.$$

Thus $\sigma(\beta) > \kappa$ holds throughout a sufficiently strong-coupling range. □

Definition 9462.1035 (Strong-coupling tube area-law certificate)**.** *A strong-coupling tube area-law certificate is the block tube area-law certificate whose area row is supplied by Definition 9462.1025, whose entropy row is supplied by Definition 9462.1027, and whose weight row is supplied by Definition 9462.1033, with $\sigma(\beta) > \kappa$ as in Definition 9462.1034.*

Theorem 9462.1036 (Strong-coupling block mass-gap surface)**.** *At fixed block scale, a strong-coupling tube area-law certificate gives the filtered uniform Wilson gap certificate and hence the fixed-sector Hilbert gap of Definition 9462.1016.*

Proof. The strong-coupling tube area-law certificate is a block tube area-law certificate by construction. Therefore Definition 9462.1015 gives filtered uniform Wilson decay with exponent

$$m_0 = (\sigma(\beta) - \kappa)c_* > 0.$$

The fixed-sector Hilbert gap follows from Definition 9462.1016. Any soft residual escaping fixed packet surfaces is not hidden by this finite layer; it is a packet-complexity, infrared, ultraviolet, topological, or renormalization tail row for the bridge layer. □

Definition 9462.1037 (Tube domination defect)**.** *Fix the entropy exponent κ from Definition 9462.1027. For a proposed decay exponent $\sigma > 0$, the tube defect of a connected tube is*

$$\mathfrak{D}_\sigma(\Sigma) = e^{\sigma|\Sigma|}|w(\Sigma)|.$$

A tube domination defect is present when no exponent $\sigma > \kappa$ admits a uniform bound

$$\mathfrak{D}_\sigma(\Sigma) \leq C$$

over the fixed block scale, fixed packet class, and fixed tube adjacency surface.

Definition 9462.1038 (Tube closure certificate)**.** *A tube closure certificate at fixed block scale consists of three ledger rows. The finite-detection row says that every bounded-complexity tube domination defect is detected by a finite Wilson or spin-network packet read. The tail-ledger row says that every domination defect escaping all bounded tube-complexity surfaces enters the packet-complexity, infrared, ultraviolet, topological, or renormalization tail ledger. The closedness row says that an accepted closed Yang–Mills bridge packet carries no unresolved tube-defect tail ledger as hidden content.*

Theorem 9462.1039 (Tube defect localization)**.** *If tube domination fails at fixed block scale, then the failure has a finite or tail address: either some bounded tube-complexity surface carries a finite tube defect, or there is a defect sequence whose tube areas escape every fixed bound and therefore belongs to the tube tail ledger.*

Proof. Failure of domination means that for each $\sigma > \kappa$ and each constant C there is a connecting tube with

$$|w(\Sigma)| > Ce^{-\sigma|\Sigma|}.$$

Fix one $\sigma_0 > \kappa$. Then there is a sequence Σ_n such that

$$e^{\sigma_0|\Sigma_n|}|w(\Sigma_n)| \to \infty.$$

If the areas $|\Sigma_n|$ are bounded along a subsequence, then all those tubes lie in one finite tube-complexity surface. Since the packet support, block dimension, and plaquette adjacency are fixed, only finitely many bounded tube patterns occur up to the displayed finite neighbourhood data. Thus a finite tube pattern carries the defect address.

If no bounded-area subsequence exists, then $|\Sigma_n| \to \infty$ along a subsequence. The defect has not disappeared; it has left every fixed tube-complexity surface. By the tail-ledger discipline, this is precisely a tube complexity, infrared, ultraviolet, topological, or renormalization tail row. These are the only two possibilities. □

Theorem 9462.1040 (Tube closure forces block tube area law)**.** *At fixed block scale, a Yang–Mills bridge packet with three-properties ledger discipline, closedness, and a tube closure certificate cannot have a hidden tube domination defect. Hence the block tube area-law weight row holds for some $\sigma > \kappa$ on the declared fixed packet surface.*

Proof. Assume that no such $\sigma > \kappa$ exists. By Definition 9462.1039, there is either a finite tube defect or a tail tube defect. In the finite case, the finite-detection row of Definition 9462.1038 makes the defect visible to a finite Wilson or spin-network packet read; that read must be ledgered, and an unledgered read violates the three-properties discipline. In the tail case, the tail-ledger row requires the defect to enter the appropriate tail ledger. Carrying that unresolved tail ledger contradicts closedness, while omitting it violates ledger discipline. Both cases contradict the stated assumptions. Therefore a hidden tube domination defect is impossible, and a block tube area-law weight row with $\sigma > \kappa$ is forced on the declared fixed surface. □

Corollary 9462.1041 (Tube closure gives the fixed-block gap route)**.** *Together with the tube area lower bound, tube entropy bound, Wilson completeness, and the Hilbert spectral bridge of Definition 9462.1019, a tube closure certificate gives the fixed-block Yang–Mills mass-gap route. The continuum claim still requires a renormalization certificate preserving this fixed block closure and exposing every ultraviolet, infrared, topological, complexity, or renormalization residual.*

Proof. The tube closure theorem supplies the missing weight row with $\sigma > \kappa$. The tube area lower bound and entropy bound then give the block tube area-law certificate, and Definition 9462.1015 gives filtered uniform Wilson decay. The Hilbert spectral bridge turns that decay into a fixed-sector non-vacuum spectral lower bound, except for residuals explicitly recorded in the tail ledger. Passing from this fixed block surface to a continuum Clay-level statement is therefore exactly the renormalization stability question, not an automatic consequence of the finite block proof. □

Theorem 9462.1042 (Tube geometry and weight give block area law)**.** *For a fixed block scale and fixed blocked Wilson packet class, the tube area lower bound, tube entropy bound, and tube weight area decay with $\sigma > \kappa$ supply the block tube area-law certificate of Definition 9462.1014.*

Proof. Definition 9462.1025 gives the area row $|\Sigma| \geq c_* T - b_O$ for every connecting tube. Definition 9462.1027 gives the entropy row $N_O(n, T) \leq A_O e^{\kappa n}$. The tube weight area decay definition gives $|w(\Sigma)| \leq e^{-\sigma|\Sigma|}$. When $\sigma > \kappa$, these are exactly the three fields required by the block tube area-law certificate. □

Corollary 9462.1043 (Finite geometry part of the Yang–Mills gap bridge)**.** *At fixed block scale, the finite geometry part of the Yang–Mills mass-gap bridge is closed by Definition 9462.1025 and Definition 9462.1027. The remaining physical input is tube weight area decay with decay exponent strictly larger than the tube entropy exponent.*

Proof. The two geometry theorems depend only on the fixed block lattice, finite packet support, and finite plaquette adjacency. They do not use continuum limiting, gauge dynamics, or an unbounded packet window. Therefore the only non-combinatorial field left in the block tube area-law certificate is the physical decay row for tube weights, with the strict inequality $\sigma > \kappa$ needed by Definition 9462.1015. □

9462.28.15 Yang–Mills RG scale-tail closure

The fixed block tube certificates locate failures inside one declared physical block surface. The remaining renormalization question is whether a positive block gap can disappear only in the limit of the RG tower. In BEDC form this disappearance is not allowed to be silent: it must be a finite block tube defect, a finite Wilson or spin-network residual, or a scale-tail ledger row.

Definition 9462.1044 (Yang–Mills RG tower)**.** *A Yang–Mills RG tower at physical block scale $\ell_* > 0$ is a sequence of bridge packets*

$$\mathcal{Y}_n = (\Lambda_n, \ell_*, G, \mathcal{H}_n, \Omega_n, H_n, \mathcal{W}_{n,\mathrm{fin}}, L_n)$$

together with blocking rows

$$B_{n+1,n} : \mathcal{Y}_{n+1} \longrightarrow \mathcal{Y}_n$$

and a continuum target $\mathcal{Y}_\infty$. The displayed tower has the scale direction

$$\mathcal{Y}_0 \leftarrow \mathcal{Y}_1 \leftarrow \mathcal{Y}_2 \leftarrow \cdots \longrightarrow \mathcal{Y}_\infty.$$

Every $\mathcal{W}_{n,\mathrm{fin}}$ is a finite Wilson-loop or spin-network packet surface, and every L_n contains the ultraviolet, infrared, topological, gauge, packet-complexity, and scale-tail ledger rows visible at that layer.

Definition 9462.1045 (RG tube gap row)**.** *For a tower layer $\mathcal{Y}_n$, an RG tube gap row consists of displayed constants*

$$\sigma_n, \qquad \kappa_n, \qquad c_n$$

where σ_n is the tube weight area-decay exponent, κ_n is the tube entropy exponent, and c_n is the tube area lower-bound constant at the fixed physical block scale. When $\sigma_n > \kappa_n$ and $c_n > 0$, the corresponding physical decay rate is

$$m_n = \frac{(\sigma_n - \kappa_n)c_n}{\ell_*}.$$

The row is interpreted only at the displayed layer and block scale; changing the scale, packet window, or tower index is a ledgered operation.

Definition 9462.1046 (Uniform RG gap certificate)**.** *A uniform RG gap certificate for Definition 9462.1044 consists of a number $m_* > 0$ and an index N such that every layer $n \geq N$ has an RG tube gap row with*

$$m_n \geq m_*.$$

Equivalently, on the displayed tail of the tower,

$$\frac{(\sigma_n - \kappa_n)c_n}{\ell_*} \geq m_* > 0.$$

Definition 9462.1047 (RG scale-tail defect)**.** *An RG scale-tail defect is a tower tail for which every sufficiently late finite layer still has a positive displayed rate $m_n > 0$, but no uniform positive lower bound exists:*

$$\inf_{n \geq N} m_n = 0$$

for every N. A scale-tail ledger row records a subsequence and the degrading certificate data

$$n_j \to \infty, \qquad m_{n_j} \to 0, \qquad (\sigma_{n_j}, \kappa_{n_j}, c_{n_j}, L_{n_j}).$$

This row says that the loss of gap is not visible inside any one fixed finite block layer, but appears only as the tower index escapes every fixed tail cutoff.

Theorem 9462.1048 (Finite RG tube defect is visible or ledgered)**.** *If a finite layer of a Yang–Mills RG tower fails to supply a positive tube gap row, then the failure has a finite Wilson or spin-network packet address or a tube tail ledger address inside that same finite layer.*

Proof. At a fixed tower layer, the block scale, packet class, plaquette adjacency, and tube complexity surface are fixed data. If $\sigma_n \leq \kappa_n$, then the tube weight decay cannot dominate the tube entropy on that surface. If $c_n = 0$, the tube area lower-bound row does not force linear separation in block time. In either case the block tube area-law certificate fails at a finite layer. Applying Definition 9462.1039 at that layer gives the two permitted addresses: a bounded tube-complexity defect detected by a finite Wilson or spin-network packet read, or a tube tail row recorded in the layer ledger. □

Theorem 9462.1049 (RG scale-tail localization)**.** *If a Yang–Mills RG tower has no uniform RG gap certificate, then either some finite layer has a finite RG tube defect, or the tower has an RG scale-tail defect.*

Proof. The absence of a uniform certificate means that no $m_* > 0$ bounds the displayed rates on all sufficiently late layers. If infinitely many late layers fail the positivity conditions $\sigma_n > \kappa_n$ and $c_n > 0$, then one of those layers is a finite RG tube defect.

Otherwise there is an index N such that every $n \geq N$ has $m_n > 0$. Since no uniform lower bound exists, for every j there is $n_j \geq N$ with $m_{n_j} < 1/j$. Passing to an increasing subsequence gives

$$n_j \to \infty, \qquad m_{n_j} \to 0.$$

This is exactly the RG scale-tail defect of Definition 9462.1047. These alternatives exhaust the failure of uniform positivity. □

Definition 9462.1050 (RG scale-tail closure certificate)**.** *An RG scale-tail closure certificate for a Yang–Mills RG tower consists of three rows:*

- *a finite-layer row saying that every finite RG tube defect is handled by the tube closure certificate of that layer;*
- *a scale-tail row saying that every sequence $n_j \to \infty$ with $m_{n_j} \to 0$ enters the scale-tail ledger;*
- *a closedness row saying that the accepted continuum Yang–Mills packet carries no unresolved finite-layer or scale-tail defect as hidden content.*

Theorem 9462.1051 (RG scale-tail closure)**.** *Let a Yang–Mills RG tower have three-properties ledger discipline on every finite layer, tube closure certificates on finite layers, and an RG scale-tail closure certificate. If the continuum Yang–Mills packet is accepted as closed, then the tower has a uniform RG gap certificate.*

Proof. Assume that no uniform RG gap certificate exists. By Definition 9462.1049, either there is a finite RG tube defect or there is an RG scale-tail defect.

In the finite-layer case, Definition 9462.1048 gives a finite Wilson or spin-network packet address or a finite-layer tube tail address. The finite-layer row of Definition 9462.1050 sends this defect to the corresponding tube closure ledger. Omitting that ledger violates ledger discipline; carrying it unresolved contradicts closedness.

In the scale-tail case, the scale-tail row records the sequence $n_j \to \infty$ with $m_{n_j} \to 0$. Omitting that row violates ledger discipline, while carrying it unresolved contradicts the closedness row. Both alternatives contradict the stated assumptions. Therefore a uniform $m_* > 0$ and a tail index N must exist. □

Definition 9462.1052 (Spectral convergence certificate)**.** *A spectral convergence certificate from an RG tower to $\mathcal{Y}_\infty$ consists of comparison rows J_n for the physical Hilbert carriers and a vacuum row $J_n\Omega_n \to \Omega_\infty$ such that for every $f \in C_c((0,\infty))$,*

$$J_n f(H_n) J_n^* \longrightarrow f(H_\infty)$$

strongly on the displayed continuum Hilbert surface. Any failure of this strong convergence is a spectral-convergence defect and must enter the renormalization ledger.

Theorem 9462.1053 (Uniform RG gap passes to the continuum)**.** *Assume a Yang–Mills RG tower has a uniform RG gap certificate with constant $m_* > 0$ and has a spectral convergence certificate with no open spectral-convergence defect. Then the continuum physical Hilbert spectrum on $\Omega_\infty^\perp$ is disjoint from $(0, m_*)$.*

Proof. Let $f \in C_c((0, m_*))$. On every sufficiently late finite layer, the vacuum-orthogonal spectrum of H_n is contained in $[m_*, \infty)$. Since f is supported below m_* and away from the vacuum point, the operator $f(H_n)$ vanishes on the displayed physical surface after the vacuum row is removed. The spectral convergence certificate gives

$$J_n f(H_n) J_n^* \longrightarrow f(H_\infty).$$

The left side is eventually zero on the vacuum-orthogonal surface, hence $f(H_\infty)$ vanishes there. If the continuum vacuum-orthogonal spectrum met $(0, m_*)$, functional calculus would supply such an f with $f(H_\infty) \neq 0$ on that spectral subspace. This contradiction proves the claim. □

Corollary 9462.1054 (RG scale-tail closure gives the continuum gap route)**.** *For a Yang–Mills bridge packet with Wilson–Peter–Weyl completeness, finite-layer tube closure, RG scale-tail closure, spectral convergence, and closed tail ledgers, the continuum vacuum-orthogonal spectrum is bounded away from zero by some $m_* > 0$.*

Proof. Definition 9462.1051 supplies a uniform RG gap certificate. By Definition 9462.1053, spectral convergence transfers that uniform lower bound to the continuum Hilbert surface. Wilson–Peter–Weyl completeness identifies the finite packet surface with the declared dense gauge-invariant vacuum-orthogonal sector. Any residual outside that sector is an ultraviolet, infrared, renormalization, topological, gauge, packet-complexity, or scale-tail ledger row; closedness rejects it as hidden unfinished content. Hence the accepted continuum packet has a positive physical mass gap on the declared sector. □

Theorem 9462.1055 (Yang–Mills RG closure boundary)**.** *The RG side of the Yang–Mills mass-gap route is closed exactly by the combination of finite-layer tube closure, RG scale-tail closure, spectral convergence, Wilson–Peter–Weyl completeness, and closed tail ledgers. If a continuum gap failed under those hypotheses, the failure would have to be a finite Wilson packet residual, a finite tube domination defect, an RG scale-tail defect, an ultraviolet, infrared, topological, gauge, or packet-complexity tail defect, or a spectral-convergence defect.*

Proof. Finite-layer tube defects are localized by Definition 9462.1048. Loss of uniform positivity along the RG tower is localized by Definition 9462.1049. Once uniform positivity is present, the passage to the continuum is exactly the content of Definition 9462.1053; any failure of that passage is, by definition, a spectral-convergence defect. All other ways for a residual to escape the fixed packet surface are precisely the physical and packet-complexity tail ledgers already named in the Yang–Mills bridge packet. Thus a closed accepted object cannot let the mass gap vanish without producing one of the displayed addresses. □

9462.28.16 Channel encoding and genetic-code bridge

The relation between the BEDC channel compiler and biological genetic coding is structural rather than token-identical. The BEDC channel code is a minimal self-delimiting event-stream codec. The DNA and RNA genetic codes are fixed-frame biological translation codecs over finite alphabets. Both systems combine a finite alphabet, a local reading discipline, a boundary or stop class, and an interpretation layer, but the corresponding objects live at different layers.

This subsection is a vision-level scientific-model bridge. It is not a CheckedCert claim about molecular biology, and it does not turn a channel glyph into a biological base, codon, or biochemical law. Its governing discipline is the decode-first principle of Definition 9221.2: channel strings may be analyzed only after they have been decoded to source event flows.

Definition 9462.1056 (Active channel event)**.** *Let $\Sigma = \{0, 1\}$. A raw event is a finite word in Σ^*, and an event flow is a finite sequence of raw events. The active BEDC channel layer uses the Zeckendorf-safe body substitution*

$$\begin{aligned} \zeta(\epsilon) &= \epsilon, \\ \zeta(0w) &= 0\,\zeta(w), \\ \zeta(1w) &= 10\,\zeta(w), \end{aligned}$$

with event terminator 11*. The channel event associated to a raw event w is*

$$\mathsf{Ev}(w) = \zeta(w)\,11.$$

Definition 9462.1057 (Legal channel stream)**.** *A legal BEDC channel stream is a finite concatenation of channel events:*

$$\mathsf{LegalZStream} = \big((0 \mid 10)^*\,11\big)^*.$$

Equivalently, the active local channel rules are

$$\begin{aligned} \text{raw } 0 &\longmapsto 0, \\ \text{raw } 1 &\longmapsto 10, \\ \text{End} &\longmapsto 11. \end{aligned}$$

Theorem 9462.1058 (The active channel body has no terminator)**.** *For every raw event w, the encoded body $\zeta(w)$ contains no adjacent substring* 11*.*

Proof. The body is a finite concatenation of fragments from $\{0, 10\}$. Neither fragment contains 11. The only possible fragment junctions are 00, 010, 100, and 1010, and none of these contains 11. Hence no adjacent 11 occurs before the appended terminator. □

Corollary 9462.1059 (The first 11 is the channel terminator)**.** *For every raw event w, the first adjacent occurrence of* 11 *in $\mathsf{Ev}(w) = \zeta(w)\,11$ is the appended event terminator.*

Proof. By Definition 9462.1058, the encoded body has no adjacent 11. The only remaining adjacent 11 is the appended terminator. □

Theorem 9462.1060 (Channel round trip)**.** *For every event flow S, decoding the compiled channel stream recovers the original source event flow:*

$$\mathsf{Decode}(\mathsf{Compile}(S)) = \mathsf{some}(S).$$

On legal channel streams, Compile *and* Decode *form an inverse pair between* EventFlow *and* LegalZStream*.*

Proof. Each encoded event has a unique first 11 terminator by Definition 9462.1059. The stream decoder therefore segments a compiled string exactly at event boundaries, then decodes each body by the inverse local readings $0 \mapsto$ raw 0 and $10 \mapsto$ raw 1. This is the vision-level restatement of the checked channel bijection recorded in Definition 9208.19. □

Remark 9462.1061 (The rejected channel rule)**.** *The displayed rule* raw $1 \mapsto 01$ *is not the active BEDC channel rule. If it were paired with the same terminator* 11*, the event* 1 *would be displayed as* 0111*. A streaming decoder would consume the leading* 0 *as body data, read* 11 *as a terminator, and leave an isolated malformed* 1*. This is exactly the failure isolated by Definition 9208.15. The active rule* raw $1 \mapsto 10$ *avoids the collision.*

Theorem 9462.1062 (Layer separation for channel and source readings)**.** *The channel-level terminator* 11 *cannot be reinterpreted as a source-level raw substring, a source-level rewrite, a DNA base, a stop codon, or a theorem judgment without an explicit bridge.*

Proof. The source layer and the channel layer share displayed glyphs but not typing judgments. In the source layer, 11 may occur inside a raw event. In the channel layer, 11 marks the end of a channel-encoded event. By Definition 9209.27, a displayed channel substring is not, in general, a decoded source event. A biological or theorem-level reading therefore requires a separate bridge after decoding. □

Definition 9462.1063 (DNA four-phase alphabet bridge)**.** *The canonical DNA alphabet is the four-base alphabet*

$$\mathsf{DNA}_4 := \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}.$$

A DNA four-phase alphabet bridge is a conservative external readback from the BEDC product of two Bool-like interfaces to this external alphabet:

$$\beta_{\mathsf{DNA}} \colon \mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow} \rightsquigarrow \mathsf{DNA}_4.$$

One biologically meaningful readback uses one axis for purine versus pyrimidine shape and the other for AT-family versus GC-family pairing.

Concretized in §2260: Realizes the four-phase DNA alphabet readback as a conservative finite NameCert bridge.

Definition 9462.1064 (Complementation readback)**.** *Under the shape-family readback, the four displayed phases are*

$$\begin{aligned} \mathrm{A} &= (\text{purine}, \mathrm{AT}), \\ \mathrm{G} &= (\text{purine}, \mathrm{GC}), \\ \mathrm{T} &= (\text{pyrimidine}, \mathrm{AT}), \\ \mathrm{C} &= (\text{pyrimidine}, \mathrm{GC}). \end{aligned}$$

The biological complementation operation is then read as the partial external involution

$$\mathrm{comp}(\text{shape}, \text{family}) = (\mathrm{flip}(\text{shape}), \text{family}),$$

which displays $\mathrm{A} \leftrightarrow \mathrm{T}$ *and* $\mathrm{G} \leftrightarrow \mathrm{C}$.

Theorem 9462.1065 (DNA four-base product bridge)**.** *The canonical DNA alphabet has a natural conservative readback as a four-phase atlas over the product* $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$.

Proof. The interface $\mathsf{Bool}^{\uparrow}$ supplies a licensed two-endpoint BEDC finite data surface, and $\mathsf{Prod}^{\uparrow}$ supplies the displayed product discipline for componentwise finite data. The external biological bridge maps the four displayed pairs to A, C, G, and T. The bridge is conservative because the biological names remain target-side readbacks; they do not become BEDC primitives. □

Corollary 9462.1066 (Four as double distinction)**.** *Within this bridge, the displayed number* 4 *is read as a product of two binary distinctions rather than as a biological primitive:*

$$4 = 2 \times 2.$$

Proof. Each coordinate of $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$ is a binary classifier. The product has four displayed phase positions, and the biological readback assigns those positions to the four canonical bases. □

Definition 9462.1067 (DNA four-phase bridge certificate template)**.** *At vision strength, the paper certificate template for the four-phase DNA alphabet has the form*

$$\mathsf{NameCert}(\mathsf{DNA4Alphabet}^{\uparrow}) = (\mathsf{SourceSpec}, \mathsf{PatternSpec}, \mathsf{ClassifierSpec}, \mathsf{StabilityCert}, \mathsf{LedgerPolicy}).$$

The template records obligations for a conservative bridge target; it is not a completed paper certificate and not a checked Lean target.

Concretized in §5177: The decoded channel, DNA four-phase product, codon atlas, translation classifier, and stop-role boundary are realized as finite NameCert rows.

Definition 9462.1068 (DNA four-phase source and pattern fields)**.** *The source field consists of two previously licensed* $\mathsf{Bool}^{\uparrow}$ *interfaces combined through the* $\mathsf{Prod}^{\uparrow}$ *product interface, together with the external biological target* DNA_4*. The pattern field consists of the four visible product phases*

$$\begin{aligned} &(\text{purine}, \text{AT}), \quad (\text{pyrimidine}, \text{AT}), \\ &(\text{purine}, \text{GC}), \quad (\text{pyrimidine}, \text{GC}), \end{aligned}$$

and the complement involution

$$\text{comp}(\text{shape}, \text{family}) = (\text{flip}(\text{shape}), \text{family}).$$

The source is the BEDC product carrier; the DNA alphabet is an external readback target.

Definition 9462.1069 (DNA four-phase classifier, stability, and ledger fields)**.** *The classifier field is the componentwise* $\mathsf{Bool}^{\uparrow}$ *classifier on the product surface, together with external* β_{DNA} *readback rows. The stability field requires product carrier closure under the componentwise classifier, complement descent through the classifier, the involution law for complement, absence of fixed points on the four canonical rows, triple-product closure for the codon atlas, and translation-classifier respect for codon classification at the external bridge layer. The ledger policy is the displayed pair ledger for* $\mathsf{DNA4Alphabet}^{\uparrow}$*, the displayed triple ledger for the codon atlas, and the biological readback provenance* β_{DNA}*.*

Theorem 9462.1070 (Complement is a fixed-point-free involution)**.** *For*

$$\text{comp}(\text{shape}, \text{family}) = (\text{flip}(\text{shape}), \text{family}),$$

the composite $\text{comp} \circ \text{comp}$ *classifies with the identity on the product phase level. On the four canonical DNA rows, the complement map has no fixed point.*

Proof. The first coordinate is flipped twice, so it returns to its original classifier value, while the second coordinate is preserved. Since flip has no fixed point on the two visible shape phases, the product map has no fixed point on the four canonical rows. □

Corollary 9462.1071 (Watson–Crick complement as product-axis flip)**.** *Under* β_{DNA}*, the complement map reads externally as*

$$\text{A} \leftrightarrow \text{T}, \qquad \text{G} \leftrightarrow \text{C}.$$

Proof. A and T share the AT-family coordinate and differ in shape; G and C share the GC-family coordinate and differ in shape. Flipping shape while preserving family gives the displayed pairings. □

Definition 9462.1072 (Codon phase atlas)**.** *The codon phase atlas is the triple product of the DNA four-phase alphabet:*

$$\mathsf{Codon}_{64}^{\uparrow} := (\mathsf{DNA}_4^{\uparrow})^3.$$

Its external biological classifier separates amino-acid-emitting codons from stop codons.

Corollary 9462.1073 (Codon phase-space bridge)**.** *The codon layer has* 64 *displayed phase positions.*

Proof. The bridge target has four DNA phases, and a codon is a triple of such phases. Therefore its displayed phase space has cardinality

$$4^3 = 64.$$

The stop classifier is an external biological partition of this phase-space. It is not the BEDC channel terminator. □

Definition 9462.1074 (Genetic translation classifier)**.** *A genetic translation classifier is an external biological classifier of the form*

$$\mathsf{TranslateClassifier} : \mathsf{Codon}_{64}^{\uparrow} \rightsquigarrow \mathsf{AminoAcid}^{\uparrow} \oplus \mathsf{Stop}^{\uparrow}.$$

It may be many-to-one: distinct codon positions may classify to the same amino-acid class. The original codon remains part of the ledger.

Definition 9462.1075 (Genetic-code ledger)**.** *A genetic-code ledger records the original codon triple, its three* $\mathsf{DNA4Alphabet}^{\uparrow}$ *component positions, the translation-classifier output, whether that output is amino-acid emitting or stop, and the external biological bridge provenance used to read each BEDC phase as a biological base.*

Theorem 9462.1076 (Translation is a classifier quotient)**.** *Genetic translation, at this bridge strength, is represented by a classifier from the codon atlas to amino-acid or stop classes, and that classifier may be many-to-one.*

Proof. A codon position is source-side phase material for the bridge. An amino-acid or stop class is the classified visible output. Multiple codon positions may classify to the same amino-acid class. BEDC reads this as classifier quotient rather than as source erasure because Definition 9462.1075 keeps the original codon triple and component readbacks visible. □

Corollary 9462.1077 (Synonymous codons are ledger-preserving compression)**.** *If two codons classify to the same amino-acid class, the visible translation meaning is compressed while each original codon remains in the ledger.*

Proof. This is the biological bridge instance of BEDC compression with memory: classification may hide distinctions at the visible meaning level, but the source map remains recoverable through the ledger. □

Corollary 9462.1078 (Codon length and alphabet size generate the phase space)**.** *The codon layer combines binary distinction, double distinction, a three-position reading window, and a stop classifier:*

$$2 = \textit{binary distinction},$$
$$4 = 2 \times 2 = \textit{DNA four-phase alphabet},$$
$$4^3 = 64 = \textit{codon phase space}.$$

Definition 9462.1079 (Stop-role bridge)**.** *The stop-role bridge relates two typed termination roles:*

$$\mathsf{ChannelEndRole}(11): \ \textit{event-boundary termination},$$
$$\mathsf{GeneticStopRole}(\text{codon}): \ \textit{protein-translation termination}.$$

The bridge asserts role analogy under finite-stream reading, not identity of tokens.

Theorem 9462.1080 (Stop-role analogy is not token identity)**.** *The BEDC channel terminator and biological stop codons have the same abstract role of ending a finite reading process, but they are not the same object.*

Proof. The BEDC terminator 11 ends one channel-encoded raw event inside $\mathsf{LegalZStream}$. A biological stop codon ends protein synthesis inside a codon reading frame. The former is a channel-boundary marker; the latter is a translation-class marker in an external biological model. Hence the licensed relation is role-equivalence under a conservative bridge, not token identity. □

Theorem 9462.1081 (Channel/genetic bridge cannot-claim boundary)**.** *The channel/genetic-code bridge does not license any of the following claims:*

- *that BEDC channel encoding is DNA;*
- *that the channel glyph pair* 11 *is a DNA or RNA stop codon;*
- *that* 01 *is the active BEDC encoding of raw* 1*;*
- *that a DNA base can be read directly from a compiled BEDC channel substring;*
- *that a biological bridge replaces BEDC NameCert obligations;*
- *that* $\mathsf{DNA4Alphabet}^{\uparrow}$ *is already a checked BEDC theorem;*
- *that the codon classifier is already a checked BEDC theorem.*

Proof. The active channel rule for raw 1 is 10, not 01, by Definition 9462.1057. Channel substrings must be decoded before source analysis by Definition 9462.1062. The DNA and codon readings are external scientific-model bridges in the sense of Definition 9462.949, and bridge targets do not replace internal certificates by Definition 9462.964. The certificate fields in Definition 9462.1067 are displayed as a vision-strength template, so they do not by themselves supply a checked theorem for either the DNA alphabet bridge or the translation classifier. □

Corollary 9462.1082 (Allowed bridge claim)**.** *The allowed claim is that BEDC channel encoding supplies a minimal self-delimiting event-stream skeleton, while DNA and RNA genetic coding supply biological fixed-frame translation systems. Their formal relation is a conservative finite-reading and stop-role bridge mediated by decoding, the product surface* $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$*, and the codon phase atlas.*

Corollary 9462.1083 (Minimal channel skeleton and biological parser)**.** *The BEDC channel parser and the genetic-code parser share a finite reading motif:*

$$0 \longmapsto \text{emit raw } 0, \qquad 10 \longmapsto \text{emit raw } 1, \qquad 11 \longmapsto \text{stop current event},$$
$$\text{codon} \longmapsto \text{emit amino acid}, \qquad \text{stop codon} \longmapsto \text{stop protein synthesis}.$$

The shared structure is finite stream, local reading rule, stop class, and ledger. The objects remain layer-distinct.

9462.28.17 Codons as six-bit finite windows

The four-phase DNA and RNA bridge of subsection 9462.28.16 has a direct finite-window reading. A codon is a three-position word over a four-phase alphabet. Once each base is read through two binary axes, a codon is exactly a six-bit window at the source-space level:

$$4^3 = 64 = 2^6.$$

This is the strongest window-six contact point. It is a source-space isomorphism, not an identification of the genetic-code translation table with a BEDC fold table.

Definition 9462.1084 (Codon six-bit source atlas)**.** *Let*

$$\mathsf{RNA4Alphabet}^{\uparrow} := \mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$$

be read externally by the two axes of Definition 9462.1094. The codon source atlas is

$$\mathsf{Codon64}^{\uparrow} := (\mathsf{RNA4Alphabet}^{\uparrow})^3.$$

The six-bit flattening map is the displayed reassociation

$$\mathsf{Flat}_6 : (\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow})^3 \rightsquigarrow (\mathsf{Bool}^{\uparrow})^6,$$

which reads a codon as three consecutive two-bit blocks.

Theorem 9462.1085 (Codon source space is a six-bit window)**.** *At the source-space level, the codon atlas has the same finite carrier size as the six-bit binary window:*

$$\left|(\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow})^3\right| = |(\mathsf{Bool}^{\uparrow})^6| = 64.$$

Proof. Each $\mathsf{Bool}^{\uparrow}$ coordinate has two visible endpoints. Hence $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$ has $2 \cdot 2 = 4$ visible phases, and a triple of such phases has $4^3 = 64$ visible positions. Reassociation exposes the same six binary coordinates, whose carrier size is $2^6 = 64$. □

Definition 9462.1086 (RNA two-axis binary display)**.** *Using the shape-family readback*

$$\mathrm{U} = 00, \quad \mathrm{C} = 01,$$
$$\mathrm{A} = 10, \quad \mathrm{G} = 11,$$

the four codons in the $\mathrm{U} + \mathrm{RR}$ *square flatten as*

codon	Flat_6 *display*
UAA	00 10 10
UAG	00 10 11
UGA	00 11 10
UGG	00 11 11

The DNA coding-strand display is obtained by replacing the first U *with* T*; the six-bit source calculation is unchanged.*

Definition 9462.1087 (Window-six Zeckendorf target)**.** *The window-six source space is*

$$\Omega_6 := \{0,1\}^6.$$

Its Zeckendorf-legal no-adjacent subspace is

$$X_6 := \{w \in \Omega_6 \mid w \text{ has no adjacent } 11\}.$$

*For a length-six binary path, the no-adjacent-*1 *count is the Fibonacci count* $F_8 = 21$*. This is the same finite-path no-adjacent pattern carried by the* $\mathsf{FibonacciCube}^{\uparrow}$ *packet of chapter 401.*

Corollary 9462.1088 (Two parallel 64-to-21 readings)**.** *The window-six no-adjacent target and the standard genetic-code bridge share the same coarse source-and-class count:*

$$\begin{aligned} \textit{window-six fold:}\quad & 64 \textit{ binary source positions} \rightsquigarrow 21 \textit{ no-adjacent window types},\\ \textit{standard genetic code:}\quad & 64 \textit{ codons} \rightsquigarrow 20 \textit{ amino-acid classes} + 1 \textit{ stop class.} \end{aligned}$$

This is a count-and-window analogy. It is not equality of classifiers or equality of fiber ledgers.

Concretized in §2234: The codon source atlas is realized as a six-coordinate BoolUp finite-window NameCert packet with Flat6 readback.

Concretized in §8421: The local stop-face classifier realizes the U+RR face, UGG exclusion, six-bit atlas, transport, replay, provenance, and NameCert rows.

Theorem 9462.1089 (The two 64-to-21 ledgers are not the same table)**.** *The window-six fold and the standard genetic-code translation classifier are not the same map.*

Proof. The window-six fold is a finite binary-window classifier whose target is the no-adjacent finite-path surface X_6. The standard genetic-code translation classifier is an external biological bridge from codons to amino-acid classes plus a stop class. Even when both displays have source size 64 and target-class count 21, their fibers are governed by different ledgers: the window fold records its own finite binary collision rows, while the genetic code records biological degeneracy rows, including singleton, doublet, triplet, quartet, and sextet amino-acid classes in the standard table. A shared count therefore does not license identity of maps. □

Theorem 9462.1090 (Global no-11 is not the stop-codon classifier)**.** *The standard stop codons are not obtained by applying the global no-adjacent-*11 *predicate to the entire flattened six-bit codon.*

Proof. By Definition 9462.1086, the relevant flattened strings are

$$\begin{aligned} \mathrm{UAA} &\longmapsto 001010,\\ \mathrm{UAG} &\longmapsto 001011,\\ \mathrm{UGA} &\longmapsto 001110,\\ \mathrm{UGG} &\longmapsto 001111. \end{aligned}$$

The string 001010 has no adjacent 11, but 001011, 001110, and 001111 do contain adjacent 11. Thus a global no-adjacent predicate on Ω_6 would keep only UAA among the three standard stop codons. The stop-codon reconstruction of Definition 9462.1100 instead applies the no-11 motif locally to the family suffix in the $\mathrm{U} + \mathrm{RR}$ square. □

Definition 9462.1091 (Local suffix rule inside the six-bit atlas)**.** *Inside the six-bit codon atlas, the stop reconstruction uses the following typed local rule:*

$$\textit{select the } \mathrm{U}+\mathrm{RR} \textit{ sector},$$
$$\textit{project positions } 2 \textit{ and } 3 \textit{ to the family axis},$$
$$\textit{exclude only the suffix } 11.$$

Equivalently,

$$\{\mathrm{AA}, \mathrm{AG}, \mathrm{GA}, \mathrm{GG}\} \longmapsto \{00, 01, 10, 11\}$$

and the Zeckendorf-legal suffixes are 00*,* 01*, and* 10*. The excluded corner is* GG*, so the surviving codons are* UAA*,* UAG*, and* UGA*.*

Theorem 9462.1092 (Codon-window bridge cannot-claim boundary)**.** *The codon-window bridge does not license any of the following claims:*

- *that the biological genetic code is the window-six fold;*
- *that the standard translation table is obtained by quotienting* Ω_6 *by the no-adjacent predicate;*
- *that the standard stop codons are exactly the globally no-adjacent flattened codons;*
- *that the equality* $4^3 = 2^6$ *supplies a biological derivation of codon length three;*
- *that a shared* 64*-to-*21 *count identifies the two fiber distributions;*
- *that* $\mathrm{G} = 11$ *in the bridge readback is the BEDC channel terminator.*

Proof. The first five refusals follow from Definition 9462.1089 and Definition 9462.1090. The last refusal follows from the layer-separation theorem Definition 9462.1107: the displayed binary readback of a biological base and the BEDC channel terminator occupy different typed layers. □

Corollary 9462.1093 (Allowed codon-window claim)**.** *A codon is naturally a six-bit finite window once the four DNA or RNA bases are read as two binary axes. The window-six bridge gives the source-space isomorphism*

$$\mathsf{Codon64}^{\uparrow} \cong \Omega_6,$$

and it explains why the numbers 64*,* 21*, and the no-*11 *motif are structurally adjacent. The stop-codon theorem remains local: in the* $\mathrm{U}+\mathrm{RR}$ *sector, excluding the family suffix* 11 *removes* UGG *and leaves the standard stop-codon L-shape* $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$*.*

9462.28.18 Stop codons as Zeckendorf-legal family suffixes

The stop-codon bridge refines subsection 9462.28.16 at the codon boundary. It is a conditional biological bridge, not a checked BEDC theorem and not a derivation of the biological genetic code from Zeckendorf coding alone. Once a two-axis RNA or DNA readback, a three-base codon window, and the standard genetic-code bridge are fixed, the standard stop codons are exactly the Zeckendorf-legal corners of a distinguished two-bit family suffix.

Definition 9462.1094 (Two-axis RNA base readback)**.** *Let the shape axis have visible values* Y *and* R*, read externally as pyrimidine and purine, and let the family axis have visible values* W *and* S*, read externally as AU-family and GC-family. The RNA four-phase atlas is*

$$\mathsf{RNA4Alphabet}^{\uparrow} := \mathrm{Shape} \times \mathrm{Family}.$$

The external biological readback is

$$\begin{aligned} \mathrm{U} &= (\mathrm{Y}, \mathrm{W}), \\ \mathrm{C} &= (\mathrm{Y}, \mathrm{S}), \\ \mathrm{A} &= (\mathrm{R}, \mathrm{W}), \\ \mathrm{G} &= (\mathrm{R}, \mathrm{S}). \end{aligned}$$

After choosing $\mathrm{Y} = 0$, $\mathrm{R} = 1$, $\mathrm{W} = 0$, *and* $\mathrm{S} = 1$, *this displays*

$$\begin{aligned} \mathrm{U} &= 00, \\ \mathrm{C} &= 01, \\ \mathrm{A} &= 10, \\ \mathrm{G} &= 11. \end{aligned}$$

This is a bridge readback from a product of licensed binary classifiers; it is not a BEDC assertion that biological bases are primitive binary strings.

Definition 9462.1095 (Purine-purine stop square)**.** *The purine-purine stop square is the four-codon region*

$$\begin{aligned} \mathsf{UPurPur} &:= \{\mathrm{U}xy \mid x \in \{\mathrm{A}, \mathrm{G}\}, y \in \{\mathrm{A}, \mathrm{G}\}\} \\ &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\}. \end{aligned}$$

Equivalently, it is the codon-cube corner cut out by

$$c_1 = \mathrm{U}, \\ \mathrm{shape}(c_2) = \mathrm{R}, \qquad \mathrm{shape}(c_3) = \mathrm{R}.$$

Definition 9462.1096 (Family suffix)**.** *For a codon* $\mathrm{U}xy$ *with* $x, y \in \{\mathrm{A}, \mathrm{G}\}$, *its family suffix is the two-bit projection of the second and third positions to the family axis:*

$$\mathsf{FamSuffix}(\mathrm{U}xy) := \mathrm{family}(x)\,\mathrm{family}(y).$$

Thus

$$\begin{aligned} \mathsf{FamSuffix}(\mathrm{UAA}) &= 00, \\ \mathsf{FamSuffix}(\mathrm{UAG}) &= 01, \\ \mathsf{FamSuffix}(\mathrm{UGA}) &= 10, \\ \mathsf{FamSuffix}(\mathrm{UGG}) &= 11. \end{aligned}$$

Definition 9462.1097 (Zeckendorf-legal family suffix)**.** *A length-two family suffix is Zeckendorf-legal when it is not the high-high corner:*

$$\mathsf{ZLegal}_2(s) \;:\Longleftrightarrow\; s \neq 11.$$

*This imports only the Zeckendorf no-*11 *motif. It does not import the BEDC channel token* 11 *into the biological layer.*

Definition 9462.1098 (Zeckendorf stop-candidate classifier)**.** *The Zeckendorf stop-candidate classifier is*

$$\begin{aligned} \mathsf{ZStopCandidate}(c_1, c_2, c_3) :\Longleftrightarrow\; & c_1 = \mathrm{U} \\ & \wedge\, \mathrm{shape}(c_2) = \mathrm{R} \\ & \wedge\, \mathrm{shape}(c_3) = \mathrm{R} \\ & \wedge\, \mathsf{ZLegal}_2(\mathsf{FamSuffix}(c_1, c_2, c_3)). \end{aligned}$$

Equivalently, this classifier reads

$$\mathrm{U} + \mathrm{RR} + \textit{not-family}\text{-}11.$$

Definition 9462.1099 (Standard stop bridge)**.** *A standard stop bridge is an external biological bridge*

$$\beta_{\mathsf{StopStd}} : \mathsf{ZStopCandidate} \rightsquigarrow \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$$

equipped with a ledger recording the RNA or DNA notation choice, the two-axis base readback, the codon triple, the family suffix, the standard-code table, the exclusion of UGG *as the family-*11 *corner, and the variant-code caveat.*

Theorem 9462.1100 (Three-stop reconstruction)**.** *Under the two-axis RNA readback and the Zeckendorf-legal family-suffix condition,*

$$\mathsf{ZStopCandidate} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

Proof. The $\mathrm{U} + \mathrm{RR}$ region contains exactly four codons:

$$\mathrm{UAA}, \quad \mathrm{UAG}, \quad \mathrm{UGA}, \quad \mathrm{UGG}.$$

Projecting the last two positions to the family axis gives

$$\begin{aligned} \mathrm{AA} &\longmapsto 00, \\ \mathrm{AG} &\longmapsto 01, \\ \mathrm{GA} &\longmapsto 10, \\ \mathrm{GG} &\longmapsto 11. \end{aligned}$$

The Zeckendorf-legal length-two suffixes are exactly 00, 01, and 10. The surviving codons are therefore UAA, UAG, and UGA, while UGG is the unique excluded corner. □

Corollary 9462.1101 (Why three and not four)**.** *Inside the* $\mathrm{U}+\mathrm{RR}$ *square, the Zeckendorf-family legality filter excludes exactly one of four raw corners. Hence the visible stop class has size three:*

$$|\mathsf{ZStopCandidate}| = 4 - 1 = 3.$$

Corollary 9462.1102 (The excluded corner)**.** *In the standard-code bridge,*

$$\mathrm{UGG} \notin \mathsf{ZStopCandidate}.$$

Proof. The codon UGG has purine-purine suffix GG, whose family projection is 11. Since $\mathsf{ZLegal}_2(11)$ is false, UGG is excluded. In the standard genetic-code bridge, this agrees with the external biological assignment of UGG to tryptophan rather than to stop. □

Corollary 9462.1103 (Release-factor L-shape bridge)**.** *For the standard bacterial release-factor bridge, the two recognition arms have the same L-shape inside the* $\mathrm{U} + \mathrm{RR}$ *square:*

$$\begin{aligned} \mathsf{RF1} &= \{\mathrm{UAA}, \mathrm{UAG}\}, \\ \mathsf{RF2} &= \{\mathrm{UAA}, \mathrm{UGA}\}, \\ \mathsf{RF1} \cup \mathsf{RF2} &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}. \end{aligned}$$

The stop set is the L-shape obtained by removing the GG *corner.*

Corollary 9462.1104 (RNA and DNA notation bridge)**.** *The same construction can be displayed in DNA coding-strand notation by replacing* U *with* T*:*

$$\textit{RNA stop set} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\},$$
$$\textit{DNA display} = \{\mathrm{TAA}, \mathrm{TAG}, \mathrm{TGA}\}.$$

The family-suffix calculation is unchanged; only the first-position pyrimidine symbol is displayed differently.

Definition 9462.1105 (Stop-candidate certificate fields)**.** *A paper certificate for* $\mathsf{ZStopCandidate}^{\uparrow}$ *or* $\mathsf{StopCodonBridge}^{\uparrow}$ *must expose five fields. The source field contains* $\mathsf{RNA4Alphabet}^{\uparrow} := \mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$*, the codon atlas* $(\mathsf{RNA4Alphabet}^{\uparrow})^3$*, the standard-code table bridge, and the RNA/DNA notation policy. The pattern field states first position* U*, purine shape in positions* 2 *and* 3*, and Zeckendorf-legal family suffix in positions* 2 *and* 3*. The classifier field identifies the standard-code bridge equality*

$$\mathsf{ZStopCandidate}(c) \iff c \in \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

Definition 9462.1106 (Stop-candidate stability and ledger fields)**.** *The stability field must preserve the construction under the chosen RNA/DNA notation bridge, the two-axis base readback, classifier transport for the stop-shape predicate, family-suffix readback for* ZLegal_2*, separation of the excluded* UGG *corner, and layer separation from the BEDC channel token* 11*. The ledger policy records the codon triple source, base-level shape and family bits, the family suffix for positions* 2 *and* 3*, the Zeckendorf-legality row, the standard-code reference, the* UGG *exclusion witness, the variant-code caveat, and the fact that no direct channel-token interpretation is used.*

Theorem 9462.1107 (No channel-token collapse)**.** *The family suffix* 11 *used to exclude* UGG *is not the BEDC channel terminator* $\mathrm{End} = 11$*.*

Proof. The BEDC channel terminator is a glyph-level boundary marker in the self-delimiting channel codec. The family suffix is a biological-bridge projection of the second and third bases of a codon through a two-axis alphabet classifier. They share a displayed binary pattern, but they occupy different typed layers. By Definition 9462.1062, the only licensed comparison is a role bridge; token identity is not licensed. □

Corollary 9462.1108 (Stop codons are classifier-level)**.** *The standard stop codons are classified at the codon layer:*

$$\mathsf{Codon}_{64}^{\uparrow} \rightsquigarrow \mathsf{AminoAcid}^{\uparrow} \oplus \mathsf{Stop}^{\uparrow}.$$

They are not obtained by scanning compiled BEDC channel substrings for 11*.*

Theorem 9462.1109 (Zeckendorf legality does not force the biological code)**.** *Zeckendorf legality alone does not prove that biology must use exactly* UAA*,* UAG*, and* UGA *as stop codons.*

Proof. The reconstruction requires external bridge choices: an RNA or DNA alphabet read as a four-phase product, the particular two-axis assignment of U, C, A, and G, codon length three, selection of the $\mathrm{U} + \mathrm{RR}$ square, the standard genetic-code table, and the no-11 family-suffix filter. Without those choices, Zeckendorf legality supplies only the abstract no-11 motif. It does not specify nucleotides, codons, release factors, or protein synthesis. □

Corollary 9462.1110 (Variant genetic codes remain bridge targets)**.** *Biological genetic-code variants may reassign one or more of the standard stop codons. Such variants do not refute Definition 9462.1100; they show that the theorem is a standard-code bridge rather than a universal biological law.*

Theorem 9462.1111 (Stop-codon suffix cannot-claim boundary)**.** *The stop-codon suffix bridge does not license any of the following claims:*

- *that Zeckendorf legality alone proves the biological genetic code;*
- *that BEDC* $\mathrm{End} = 11$ *is identical to a DNA or RNA stop codon;*
- *that the family suffix* 11 *in* UGG *is the channel terminator* 11*;*
- *that* UAA*,* UAG*, and* UGA *are stop codons in every biological genetic code;*
- *that* UGG *is tryptophan by BEDC logic alone;*
- *that* $\mathsf{ZStopCandidate}^{\uparrow}$ *is already a checked certificate;*
- *that a compiled BEDC channel substring directly denotes a nucleotide base.*

Proof. The first five refusals follow from Definition 9462.1109 and Definition 9462.1110. The channel-token refusals follow from Definition 9462.1107. The checked certificate refusal follows because Definition 9462.1105 and Definition 9462.1106 record paper obligations, not Lean targets. □

Corollary 9462.1112 (Allowed stop-codon suffix bridge claim)**.** *Under the two-axis RNA or DNA bridge and a Zeckendorf-style no-*11 *family-suffix legality condition, the standard stop-codon set is structurally reconstructed as*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\},$$

with UGG *as the unique excluded family-*11 *corner.*

Concretized in §1374: The stop-codon Zeckendorf suffix bridge is realized as a finite codon-window NameCert packet with an explicit cannot-claim boundary.

9462.28.19 Research program for Zeckendorf-legal termination

The stop-codon suffix bridge of subsection 9462.28.18 gives a paper-level reconstruction of the standard stop set from a two-axis RNA readback and a no-11 family-suffix condition. The research program attached to that reconstruction has a narrower purpose: it separates the finite theorem that can be checked internally from the biological bridge evidence that must remain external, and it names the certificate fields needed before the bridge may be exported at stronger status.

The guiding question is why the standard stop codons UAA, UAG, and UGA appear together. In BEDC form, the question becomes whether the standard stop-codon set can be reconstructed as the Zeckendorf-legal part of the $\mathrm{U} + \mathrm{RR}$ square under a conservative two-axis bridge for RNA bases:

$$\mathsf{UPurineSquare} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\},$$
$$\mathrm{AA} \mapsto 00, \qquad \mathrm{AG} \mapsto 01, \qquad \mathrm{GA} \mapsto 10, \qquad \mathrm{GG} \mapsto 11.$$

The finite claim keeps 00, 01, and 10 and excludes the unique 11 corner. The biological claim then reads the surviving L-shape as the standard stop set.

Definition 9462.1113 (Conditional stop-codon reconstruction problem)**.** *The stop-codon reconstruction problem is the following conditional bridge target. Given*

- *the product atlas* $\mathsf{RNA4Alphabet}^{\uparrow} := \mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$*;*
- *the readback* $\mathrm{U} = (\mathrm{Y}, \mathrm{W})$*,* $\mathrm{C} = (\mathrm{Y}, \mathrm{S})$*,* $\mathrm{A} = (\mathrm{R}, \mathrm{W})$*, and* $\mathrm{G} = (\mathrm{R}, \mathrm{S})$*;*

- *codons as triples* $(\mathsf{RNA4Alphabet}^{\uparrow})^3$*;*
- *the stop-shape region* $\mathrm{U}+\mathrm{RR}$*;*
- *the family-suffix legality condition excluding* 11*;*
- *a standard-code biological bridge,*

prove that the reconstructed stop-candidate class is exactly

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

The problem is a bridge problem, not a kernel-level derivation of the biological genetic code.

Concretized in §1374: The finite U+RR suffix legality route is realized by the StopCodonZeckendorfSuffixUp NameCert surface.

Definition 9462.1114 (Research inputs)**.** *The BEDC-side inputs are the active channel theorem package of subsection 9462.28.16, the derived interfaces* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Prod}^{\uparrow}$*, classifier transport, displayed pair and triple ledgers, layer separation, and cannot-claim registry discipline. The biological inputs are external rows: RNA bases, three-position codons, the standard stop set, the assignment of* UGG *as a sense codon in the standard code, release-factor recognition data where used, and variant genetic-code tables where used.*

Definition 9462.1115 (Research subquestions)**.** *The program decomposes into nine subquestions:*

- *why adjacent* 11 *is the BEDC channel boundary pattern and which no-internal-*11 *theorem supports it;*
- *how RNA or DNA bases are read as a conservative* $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$ *target;*
- *how the triple product yields the* 64*-position codon atlas;*
- *why the standard stop set is analyzed inside the* $\mathrm{U}+\mathrm{RR}$ *square;*
- *whether excluding the family suffix* 11 *reconstructs exactly the three standard stop codons;*
- *how* UGG *functions as the excluded square corner while remaining a standard-code sense codon;*
- *how bacterial release-factor recognition arms compare with the same L-shape;*
- *which bridge field changes when a genetic-code variant reassigns a stop codon;*
- *which part is finite enumeration suitable for Lean checking and which part remains biological bridge evidence.*

Definition 9462.1116 (Four-phase stop-square data)**.** *The finite data of the stop square consists of a shape axis* $\{\mathrm{Y}, \mathrm{R}\}$*, a family axis* $\{\mathrm{W}, \mathrm{S}\}$*, the readback*

$$\begin{aligned} \mathrm{U} &= (\mathrm{Y}, \mathrm{W}), \quad \mathrm{C} = (\mathrm{Y}, \mathrm{S}), \\ \mathrm{A} &= (\mathrm{R}, \mathrm{W}), \quad \mathrm{G} = (\mathrm{R}, \mathrm{S}), \end{aligned}$$

and the family projection on the purine-purine suffix:

$$\mathrm{family}(\mathrm{A}) = 0, \qquad \mathrm{family}(\mathrm{G}) = 1.$$

Thus the second and third positions of the $\mathrm{U}+\mathrm{RR}$ *square display the binary family square* 00*,* 01*,* 10*,* 11*.*

Definition 9462.1117 (Research module boundaries)**.** *The program has six module boundaries.*

Channel termination. *Define the legal channel body, event marker, no-internal-*11 *theorem, first-*11 *terminator theorem, decode-compile inverse theorem, and layer-separation theorem.*

Four-phase alphabet bridge. *Package* $\mathsf{RNA4Alphabet}^{\uparrow}$ *as a product of two binary interfaces; record the two-axis biological readback, complement involution, codon triple product, and translation classifier as bridge data.*

Stop-codon enumeration. *Define the* U+RR *square, family suffix, Zeckendorf-legality predicate, and stop-candidate classifier; prove by four cases that* UGG *is the unique excluded corner.*

Release-factor bridge. *Record the standard bacterial recognition arms* $\{\mathrm{UAA}, \mathrm{UAG}\}$ *and* $\{\mathrm{UAA}, \mathrm{UGA}\}$ *as external data and compare their union with the reconstructed L-shape.*

Variant-code boundary. *Compare standard and alternative genetic-code tables by identifying which external bridge field changes: stop-shape region, translation classifier, release-factor bridge, or standard-code table.*

Formalization boundary. *Keep the finite enumeration over* $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$ *inside the checked layer, while keeping base names, biochemical translation, and empirical recognition data as external bridge rows.*

Theorem 9462.1118 (Finite stop-square enumeration target)**.** *Inside the* $\mathrm{U} + \mathrm{RR}$ *square, the predicate "family suffix is not* 11*" selects exactly* UAA*,* UAG*, and* UGA*.*

Proof. The square has four elements:

$$\mathrm{UAA}, \quad \mathrm{UAG}, \quad \mathrm{UGA}, \quad \mathrm{UGG}.$$

Their family suffixes are respectively 00, 01, 10, and 11. The predicate rejects exactly the fourth suffix. Hence the selected class is precisely

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

□

Corollary 9462.1119 (Distinguished non-stop corner)**.** *Within the standard-code bridge,* UGG *is the unique non-selected corner of the* $\mathrm{U} + \mathrm{RR}$ *square.*

Proof. By Definition 9462.1118, the only rejected family suffix is 11. The codon UGG is the unique square element whose second and third positions both have family bit 1. □

Theorem 9462.1120 (Variant-code deformation boundary)**.** *A genetic-code variant that reassigns one of the standard stop codons changes external biological bridge data rather than the finite enumeration of Definition 9462.1118.*

Proof. The enumeration uses only the two-axis readback, the U+RR square, and the no-11 family-suffix condition. A variant genetic code changes the biological interpretation of at least one codon position or recognition field. That change belongs to the translation classifier, the release-factor bridge, the standard-code table, or the chosen stop-shape bridge. It therefore alters the bridge target, not the four-case finite calculation. □

Definition 9462.1121 (Certificate ladder for the stop program)**.** *The vision-level program has three strength layers. At paper-certificate strength it supplies proofs of no-internal-*11*, first-*11 *termination, the finite stop-square enumeration, the excluded-corner result, and layer separation. At checked-certificate strength it supplies Lean definitions for the four-element alphabet, codons, the* U + RR *square, family suffix, Zeckendorf-legality predicate, and finite enumeration theorem. At biological-bridge strength it supplies the external standard genetic-code table, release-factor recognition rows, variant-code boundary report, and ledger statement forbidding host leakage.*

Definition 9462.1122 (External biological reference rows)**.** *The biological bridge report records external references as rows rather than as BEDC primitives. The minimum rows are: a standard stop-codon row recording the* 64 *codons,* 61 *amino-acid codons, and three standard stop signals; a genetic-code table row recording standard and alternative code tables, including RNA/DNA notation policy; and a release-factor row recording the bacterial* RF1 *and* RF2 *recognition arms when that comparison is used.*

Definition 9462.1123 (Lean finite skeleton)**.** *The internal finite theorem can be represented by a small closed alphabet, codon triples, and Boolean predicates:*

```
inductive Bit where
  | zero
  | one

abbrev Shape  := Bit
abbrev Family := Bit

structure Base where
  shape  : Shape
  family : Family

def U : Base := { shape := Bit.zero, family := Bit.zero }
def C : Base := { shape := Bit.zero, family := Bit.one }
def A : Base := { shape := Bit.one,  family := Bit.zero }
def G : Base := { shape := Bit.one,  family := Bit.one }

structure Codon where
  b1 : Base
  b2 : Base
  b3 : Base

def isPurine (b : Base) : Prop := b.shape = Bit.one
def familyBit (b : Base) : Bit := b.family

def UPurineSquare (c : Codon) : Prop :=
  c.b1 = U /\ isPurine c.b2 /\ isPurine c.b3

def ZLegalSuffix (c : Codon) : Prop :=
  not (familyBit c.b2 = Bit.one /\ familyBit c.b3 = Bit.one)

def ZStopCandidate (c : Codon) : Prop :=
```

```
UPurineSquare c /\ ZLegalSuffix c
```

The target theorem states that ZStopCandidate *is equivalent to membership in the three displayed codons* UAA*,* UAG*, and* UGA*. The proof is a finite case split over the second and third bases in* $\{\mathrm{A}, \mathrm{G}\}$*.*

Definition 9462.1124 (Stop-square diagram)**.** *The standard-code bridge target is displayed by the square*

	third A	*third* G
second A	UAA : 00 ✓	UAG : 01 ✓
second G	UGA : 10 ✓	UGG : 11 ×

The check marks denote Zeckendorf-legal family suffixes. The crossed corner is the unique high-high suffix. The diagram is a biological bridge target: the channel token 11 *and the family suffix* 11 *remain layer-distinct.*

Theorem 9462.1125 (Research-program cannot-claim boundary)**.** *The research program does not license any claim that Zeckendorf coding alone proves the biological genetic code, that the BEDC channel terminator is identical to a DNA or RNA stop codon, that the family suffix* 11 *is the channel terminator, that the standard stop codons are universal across all genetic codes, that* UGG *is tryptophan by BEDC logic alone, that* ZStopCandidate *is already checked, or that a compiled channel substring directly denotes a nucleotide base.*

Proof. The refusals are inherited from Definition 9462.1111. The present program only organizes the route from vision-strength reconstruction to finite enumeration and external bridge reporting; it does not discharge the external biological bridge fields or collapse the channel and codon layers. □

Corollary 9462.1126 (Allowed research-program claim)**.** *Under the two-axis RNA bridge and Zeckendorf-family suffix legality, the standard stop-codon set is structurally reconstructed as the L-shape*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$$

inside the $\mathrm{U} + \mathrm{RR}$ *square. The theorem is conditional and layered: Zeckendorf supplies the no-*11 *pattern, BEDC supplies classifier and ledger discipline, and biology supplies the standard-code bridge target.*

9462.28.20 DNA window-six symbolic-physical dynamics

The codon-window bridge of subsection 9462.28.17 identifies a codon with a six-bit local source window, and the stop-codon bridge of subsection 9462.28.18 identifies the standard stop class as a Zeckendorf-legal L-shape inside the $\mathrm{U} + \mathrm{RR}$ square. The dynamic bridge adds a further discipline: DNA and RNA should not be treated merely as static code tables. They form a relative symbolic-physical layer with local windows, frames, stop boundaries, raw-codon ledgers, and state transitions executed through cellular machinery.

The word "relative" is load-bearing. Replication, transcription, repair, translation, release, accessibility changes, and chromatin updates are performed by polymerases, ribosomes, release factors, repair enzymes, chromatin proteins, tRNA pools, regulatory factors, and cellular context. The bridge therefore does not claim that DNA is a closed autonomous dynamics. It claims that DNA/RNA reading has a symbolic state space and transition surface rich enough to require its own ledger rather than collapsing into the final protein class.

Definition 9462.1127 (DNA window-six dynamics bridge)**.** *A DNA window-six dynamics bridge is a conservative biological bridge whose source field contains RNA or DNA sequence histories together with the two-axis base readback of Definition 9462.1094. Its pattern field records local codon windows, frame movement, stop-boundary transitions, replication, transcription, repair, mutation, recombination, and accessibility updates. Its classifier field includes the codon classifier, stop classifier, amino-acid classifier, and microstate-preserving synonymous-codon classifier. Its stability field requires frame-step closure, window-six readback stability, stop-boundary determinacy under a fixed biological code table, and ledger transport under explicit cellular-context parameters. Its ledger policy preserves the raw codon, frame position, base-axis bits, code table, release-factor or tRNA context when used, accessibility state, and biological bridge metadata.*

Definition 9462.1128 (Translation state)**.** *A translation state has the schematic form*

$$q_t := (f_t, c_t, r_t, \kappa_t),$$

where f_t records the current frame, c_t records the current codon, r_t records ribosomal or release status, and κ_t records the cellular context relevant to the bridge. In the minimal finite model, the context is suppressed and the state is

$$q_t := (f_t, c_t).$$

Definition 9462.1129 (Elongation and stop transitions)**.** *For a non-stop codon c_t with*

$$\mathsf{TranslateClassifier}(c_t) = \mathsf{AminoAcid}(a),$$

the elongation transition advances the frame by one codon and records the emitted amino-acid class together with the raw codon:

$$q_t \longrightarrow q_{t+1}.$$

For a stop codon with

$$\mathsf{TranslateClassifier}(c_t) = \mathsf{Stop},$$

the transition enters a halted or release state:

$$q_t \longrightarrow q_{\mathsf{halt}}.$$

Theorem 9462.1130 (Stop codons are boundary states)**.** *Under the standard-code biological bridge, the codons* UAA, UAG, *and* UGA *are translation boundary states.*

Proof. By Definition 9462.1100, the Zeckendorf-family stop candidate under the chosen two-axis readback is exactly $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$. The external standard-code bridge supplies the biological classifier row that reads these codons as stop signals. Combining the finite reconstruction with that bridge row gives the transition $q_t \to q_{\mathsf{halt}}$ for precisely those three standard-code codons. This is a bridge theorem, not a BEDC kernel theorem about biology. □

Definition 9462.1131 (Expanded DNA dynamic state)**.** *The local six-bit codon cube*

$$Q_6 := \{0,1\}^6$$

is sufficient for local codon classification but not for the full symbolic-physical dynamics. A minimal expanded state has the form

$$S_{\mathsf{DNA}} := \Sigma^n \times \mathsf{FrameState} \times \mathsf{EpigeneticState} \times \mathsf{ChromatinTopologyState} \times \mathsf{CellularEnzymeContext} \times \mathsf{TranslationContext} \times \mathsf{F}$$

Equivalently, a compressed state may be displayed as

$$D_t := (s_t, f_t, e_t, c_t, \kappa_t),$$

where s_t is the sequence state, f_t is the frame and window position, e_t records epigenetic marks, c_t records accessibility and topology, and κ_t records cellular machinery and regulatory context.

Definition 9462.1132 (DNA transition families)**.** *The dynamic bridge names transition families rather than imposing a universal execution order:*

R	*replication*
T	*transcription or selective readout*
L	*translation*
M	*mutation*
P	*repair*
G	*recombination or genome rearrangement*
E	*epigenetic update*
C	*chromatin or topology update*
F	*feedback from protein or translation state*

The schematic form is

$$D_{t+1} = \Phi(D_t; \kappa_t),$$

where the active components of Φ depend on cell state and bridge context. No fixed composition such as $C \circ E \circ P \circ M \circ R \circ T \circ L$ is licensed as a universal biological sequence.

Theorem 9462.1133 (Window six is local but incomplete)**.** *The local state Q_6 captures the codon-level classifier domain, but it does not exhaust DNA dynamics.*

Proof. Two biological states may share the same local six-bit codon window and differ in frame, methylation, histone-related marks, chromatin accessibility, polymerase status, ribosome state, repair status, release-factor availability, tRNA context, or code-table variant. Those differences may change whether the sequence is copied, read, repaired, translated, released, or ignored. Hence Q_6 is a local classifier domain, while the dynamic bridge must also record sequence position, frame, accessibility, repair, and cellular-machine context. □

Definition 9462.1134 (Codon microstate ledger)**.** *A codon microstate ledger is a bridge row*

$$\mathsf{CodonLedger}(c, o, \kappa)$$

recording the raw codon c, frame position, output class o, tRNA or release-factor context when used, codon-usage or optimality data when used, source code table, and variant-code metadata. The ledger separates the codon microstate from the amino-acid or stop class to which it is classified.

Theorem 9462.1135 (Translation does not collapse codon microstates)**.** *If two codons classify to the same amino-acid class, the biological bridge must still retain their raw codon rows whenever dynamic effects are in scope.*

Proof. The amino-acid output is a many-to-one classifier. If the raw codon were discarded, the bridge could not express codon usage, elongation-rate differences, mRNA-stability effects, tRNA-context dependence, or variant-code reassignment. Therefore the translation classifier is a ledger-preserving quotient, not a deletion of source microstate. □

Definition 9462.1136 (Variant-code deformation fields)**.** *Non-standard genetic codes are recorded as deformations of biological bridge fields. A variant may change the stop-shape region, the translation classifier, release-factor or release-mechanism rows, the biological code table attached to the readback, or the cellular context that permits readthrough, recoding, or organellar translation. Such deformations do not refute the finite stop-square enumeration; they change the bridge target in which that enumeration is interpreted.*

Definition 9462.1137 (Dynamic bridge modules)**.** *The dynamic research program decomposes into seven modules:*

Channel and Zeckendorf termination. *Preserve the no-internal-*11*, first-*11*, and layer-separation theorems for the BEDC channel.*

RNA four-phase and window six. *Record* $\mathsf{RNA4Alphabet}^{\uparrow} := \mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$*, the* Flat_6 *map, and the codon-window ledger.*

Stop L-shape reconstruction. *Retain* UPurPur*, family suffix, Zeckendorf-legality, the three-stop theorem, and the* UGG *excluded-corner row.*

Translation dynamics. *Define translation states, elongation transitions, stop transitions, and the halted or release state.*

DNA state-space dynamics. *Define expanded sequence, frame, accessibility, chromatin, repair, mutation, and cellular-context fields.*

Microstate ledger. *Separate raw codon rows from amino-acid or stop output classes and record synonymous-codon non-collapse.*

Release factors and variants. *Compare the RF1/RF2 L-shape bridge with variant-code deformation fields.*

Definition 9462.1138 (Dynamic bridge figures)**.** *The bridge has three stable display forms. The two-axis base atlas is*

	W	S
Y	U	C
R	A	G

The stop square is the L-shape of Definition 9462.1124. The dynamic state inclusion is

$$Q_6 \subset S_{\mathsf{DNA}},$$

where Q_6 *is the local codon window and* S_{DNA} *is the expanded sequence-frame-accessibility-context state.*

Theorem 9462.1139 (Dynamic bridge cannot-claim boundary)**.** *The DNA window-six dynamics bridge does not license any claim that Zeckendorf or window six forces DNA biology, that DNA has a fully autonomous closed dynamics independent of cellular machinery, that the BEDC channel terminator is identical to a biological stop codon, that the family suffix* 11 *in* UGG *is the channel terminator, that the standard stop codons are universal across all genetic codes, that* UGG *is tryptophan by BEDC logic alone, that synonymous codons are dynamically identical after amino-acid classification, that the DNA state space is exhausted by* Q_6*, that* Φ *has a universal fixed composition order, that the biological bridge replaces* NameCert *obligations, or that* $\mathsf{DNAWindow6Dynamics}^{\uparrow}$ *is already checked.*

Proof. The channel-token and stop-codon refusals are inherited from Definition 9462.1111 and Definition 9462.1092. The autonomy and state-space refusals follow from Definition 9462.1133. The synonymous codon refusal follows from Definition 9462.1135. The fixed-order refusal follows from Definition 9462.1132. The checked-status refusal follows because the present subsection is a vision-level biological bridge plan rather than a Lean target. □

Corollary 9462.1140 (Allowed dynamic bridge claim)**.** *Under a standard-code biological bridge, window six and the no-*11 *motif supply a structural model for local DNA and RNA reading and stop-boundary dynamics. The three standard stop codons form the Zeckendorf-legal L-shape in the* $\mathrm{U} + \mathrm{RR}$ *square, and the broader DNA layer is modeled as a relative symbolic-physical dynamics with state, context, transitions, and ledger-preserved codon microstates.*

Concretized in §2161: CodonWindowSpectralInverseUp realizes the DNA window-six dynamics bridge as a finite codon-cube spectral inverse packet.

9462.28.21 Local tile dynamics of the genetic code

This subsection records a finite, non-axiomatic analysis of the standard RNA genetic-code table under the two-axis readback

$$\mathrm{U} = 00, \qquad \mathrm{C} = 01, \qquad \mathrm{A} = 10, \qquad \mathrm{G} = 11.$$

It does not claim that DNA must obey Zeckendorf rules, that the genetic code is the window-six fold, or that the biological code is globally equal to a no-11 language. The claims have three layers: a coordinate definition, a finite enumeration over the standard table, and biological bridge readings supplied by external genetic-code data.

Definition 9462.1141 (Local tile dynamics bridge status)**.** *The local-tile dynamics bridge* $\mathsf{LocalTileDynamics}^{\uparrow}$ *is a* $\mathsf{VisionSeed/BiologicalBridge}$ *target and a finite-enumeration paper-certificate candidate. A future certificate must separate source, pattern, classifier, stability, and ledger fields. The source field records the finite RNA alphabet, the two-axis readback, the codon cube, and the standard genetic-code table. The pattern field records local* Q_2 *faces,* $3+1$ *translation-class splits, corner partitions, and alternative-code deformations. The classifier field records codon labels. The stability field records closure of the finite enumeration under the chosen readback. The ledger policy preserves the codon, six-bit display, local face, free coordinates, translation label, and table provenance.*

Definition 9462.1142 (RNA alphabet and window-six readback)**.** *Let*

$$\Sigma := \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}.$$

The coordinate map $b : \Sigma \to \{0,1\}^2$ *is*

$$\begin{aligned} b(\mathrm{U}) &= 00, \quad b(\mathrm{C}) = 01, \\ b(\mathrm{A}) &= 10, \quad b(\mathrm{G}) = 11. \end{aligned}$$

For a codon $c = c_1c_2c_3 \in \Sigma^3$*, the six-bit readback is*

$$\beta(c) := b(c_1)\, b(c_2)\, b(c_3) \in \{0,1\}^6.$$

Write $Q_6 := \{0,1\}^6$.

Definition 9462.1143 (Standard translation label map)**.** *The standard translation label map is the external biological table*

$$\tau_{\mathsf{std}} : \Sigma^3 \rightsquigarrow \mathsf{AminoAcidClass} \oplus \mathsf{Stop}.$$

The rows used by this subsection are

label	*codons*
Stop	UAA, UAG, UGA
Trp	UGG
Ile	AUU, AUC, AUA
Met	AUG
Phe	UUU, UUC
Leu	UUA, UUG, CUU, CUC, CUA, CUG.

The biological labels are bridge data, not BEDC primitives.

Definition 9462.1144 (Axis-aligned local Q_2 face)**.** *An axis-aligned local Q_2 face of Q_6 is obtained by choosing two of the six bit positions to vary and fixing the other four. Such a face has four vertices and is displayed as*

$$F \simeq Q_2 := \{00, 01, 10, 11\}.$$

Definition 9462.1145 (Three-plus-one face)**.** *An axis-aligned local Q_2 face F is a $3+1$ face under a translation label map τ when the multiset of labels on its four codons has exactly two labels with multiplicities 3 and 1. The size-three label class is the three-point class, and the remaining codon is the singleton corner.*

Definition 9462.1146 (Local two-bit tile and corner mechanisms)**.** *The local no-11 triangle is*

$$X_2 := \{00, 01, 10\}$$

inside

$$Q_2 := \{00, 01, 10, 11\}.$$

Thus

$$Q_2 = X_2 \sqcup \{11\}.$$

An 11-control tile is a $3+1$ face whose three-point class is X_2 and whose singleton corner is 11. A 00-singleton tile is a $3+1$ face whose singleton corner is 00 and whose three-point class is $Q_2 \setminus \{00\}$.

Concretized in §3393: The local two-bit boundary-tile classifier is realized as finite codon-window, boundary-tile, transport, provenance, and NameCert rows.

Definition 9462.1147 (Distinguished local faces)**.** *The two 11-control faces are*

$$\mathsf{AUFace} := \{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}, \mathrm{AUG}\},$$
$$\mathsf{StopTrpFace} := \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\}.$$

The two 00-singleton faces are

$$\mathsf{PheLeuFace}_{\mathrm{U}} := \{\mathrm{UUU}, \mathrm{UUA}, \mathrm{CUU}, \mathrm{CUA}\},$$
$$\mathsf{PheLeuFace}_{\mathrm{C}} := \{\mathrm{UUC}, \mathrm{UUG}, \mathrm{CUC}, \mathrm{CUG}\}.$$

Definition 9462.1148 (Reserved local corner)**.** *For a local codon Q_2 face F, a reserved local corner is a singleton corner whose biological bridge row is separated from the three-point class. Reserved means singleton or control-sensitive under the translation classifier. It does not mean absent, deleted, or identified with the BEDC channel terminator.*

Definition 9462.1149 (Stop graph and release-factor arms)**.** *On*

$$X_2 = \{00, 01, 10\}$$

put the Hamming-distance-one graph

$$01 \ - \ 00 \ - \ 10.$$

Under the stop-face readback this is

$$\mathrm{UAG} \ - \ \mathrm{UAA} \ - \ \mathrm{UGA}.$$

The bacterial release-factor arms are external biological bridge rows:

$$\begin{aligned} \mathsf{RF1Arm} &:= \{\mathrm{UAA}, \mathrm{UAG}\}, \\ \mathsf{RF2Arm} &:= \{\mathrm{UAA}, \mathrm{UGA}\}. \end{aligned}$$

Definition 9462.1150 (Local reassignment modes)**.** *A leaf reassignment on an $X_2 \sqcup \{11\}$ tile moves one leaf of the X_2 part to another translation class. The central displayed case is*

$$\{00, 01, 10\}/\{11\} \rightsquigarrow \{00, 01\}/\{10, 11\}.$$

An arm reassignment on the stop graph moves an edge such as $\{00, 01\}$ as one biological classifier row.

Definition 9462.1151 (Local translation state)**.** *A local translation state has the schematic form*

$$S_t := (f_t, w_t, r_t, \kappa_t),$$

where f_t is the frame, $w_t \in Q_6$ is the current window, r_t records release or ribosome state, and κ_t records cellular context. Under the standard-code bridge:

$$\begin{aligned} w_t \notin \mathrm{Stop} \quad &\Longrightarrow \quad S_t \to S_{t+1}, \\ w_t \in \mathrm{Stop} \quad &\Longrightarrow \quad S_t \to S_{\mathsf{halt}}. \end{aligned}$$

This is a modeling interface, not a closed autonomous biological dynamics.

Lemma 9462.1152 (Codon cube and window-six bijection)**.** *The readback β gives a finite bijection*

$$\Sigma^3 \cong Q_6.$$

Proof. Each base in Σ has a unique two-bit coordinate under b, so a three-base codon has a unique six-bit display. Conversely, every six-bit word splits uniquely into three two-bit blocks, and each block is one of 00, 01, 10, and 11, hence decodes uniquely to one of U, C, A, and G. □

Lemma 9462.1153 (Local two-bit partition)**.** *The two-bit square satisfies*

$$Q_2 = X_2 \sqcup \{11\}, \qquad X_2 = \{00, 01, 10\}.$$

Proof. The four two-bit words are exactly 00, 01, 10, and 11. Removing the word 11 leaves exactly 00, 01, and 10. The displayed union is disjoint and exhaustive. □

Theorem 9462.1154 (Stop and tryptophan 11-control tile)**.** *In the* $\mathrm{U} + \mathrm{RR}$ *face,*

$$\begin{aligned} \mathrm{U} + \mathrm{RR} + X_2 &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} = \mathrm{Stop}, \\ \mathrm{U} + \mathrm{RR} + 11 &= \{\mathrm{UGG}\} = \mathrm{Trp} \end{aligned}$$

under the standard-code biological bridge.

Proof. The family suffix coordinates are

$$\begin{aligned} \mathrm{UAA} &\longmapsto 00, \\ \mathrm{UAG} &\longmapsto 01, \\ \mathrm{UGA} &\longmapsto 10, \\ \mathrm{UGG} &\longmapsto 11. \end{aligned}$$

By Definition 9462.1153, the X_2 part is $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$ and the 11 corner is UGG. The labels Stop and Trp are supplied by the standard genetic-code bridge. □

Corollary 9462.1155 (UGG is the stop-face 11 corner)**.** *Inside the stop face,*

$$\mathrm{UGG} = \mathrm{U} + \mathrm{RR} + 11.$$

Proof. Both varying suffix bases are G, and G has family coordinate 1. Hence GG has local family suffix 11. □

Theorem 9462.1156 (Isoleucine and methionine 11-control tile)**.** *In the* AU *face,*

$$\begin{aligned} \mathrm{AU} + X_2 &= \{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}\} = \mathrm{Ile}, \\ \mathrm{AU} + 11 &= \{\mathrm{AUG}\} = \mathrm{Met} \end{aligned}$$

under the standard-code biological bridge. In an initiation context, the same AUG *row carries the canonical start bridge row.*

Proof. The third-base coordinates in the AU face are

$$\begin{aligned} \mathrm{AUU} &\longmapsto 00, \\ \mathrm{AUC} &\longmapsto 01, \\ \mathrm{AUA} &\longmapsto 10, \\ \mathrm{AUG} &\longmapsto 11. \end{aligned}$$

Thus the X_2 part is $\{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}\}$ and the 11 corner is AUG. The labels Ile, Met, and the canonical-start reading are external standard-code bridge rows. □

Corollary 9462.1157 (Dual 11-control tiles)**.** *The standard genetic-code bridge contains two dual* $X_2 \sqcup \{11\}$ *tiles:*

$$\begin{aligned} \mathrm{Ile/Met} &= \mathrm{AU} + X_2 \ / \ 11, \\ \mathrm{Stop/Trp} &= \mathrm{U} + \mathrm{RR} + X_2 \ / \ 11. \end{aligned}$$

Proof. Combine Definition 9462.1154 and Definition 9462.1156. □

Concretized in §3393: The finite stop/Trp and Ile/Met boundary tiles are realized as a concrete NameCert packet.

Theorem 9462.1158 (Phe and Leu 00-singleton tiles)**.** *The two Phe/Leu faces are* 00*-singleton tiles:*

face	00 *singleton*	$Q_2 \setminus \{00\}$ *class*
$\{\mathrm{UUU}, \mathrm{UUA}, \mathrm{CUU}, \mathrm{CUA}\}$	$\mathrm{UUU} = \mathrm{Phe}$	$\mathrm{UUA}, \mathrm{CUU}, \mathrm{CUA} = \mathrm{Leu}$
$\{\mathrm{UUC}, \mathrm{UUG}, \mathrm{CUC}, \mathrm{CUG}\}$	$\mathrm{UUC} = \mathrm{Phe}$	$\mathrm{UUG}, \mathrm{CUC}, \mathrm{CUG} = \mathrm{Leu}.$

Proof. For the first face, vary the second bit of the first base and the second bit of the third base while keeping the other four bits fixed. The local coordinates are

$$\begin{aligned} 00 &\longmapsto \mathrm{UUU}, \quad 01 \longmapsto \mathrm{UUA}, \\ 10 &\longmapsto \mathrm{CUU}, \quad 11 \longmapsto \mathrm{CUA}. \end{aligned}$$

For the second face, the same free bit positions give

$$\begin{aligned} 00 &\longmapsto \mathrm{UUC}, \quad 01 \longmapsto \mathrm{UUG}, \\ 10 &\longmapsto \mathrm{CUC}, \quad 11 \longmapsto \mathrm{CUG}. \end{aligned}$$

The standard-code bridge labels the 00 vertices as Phe and the remaining three vertices as Leu. □

Theorem 9462.1159 (Number of axis-aligned local Q_2 faces)**.** *The codon cube Q_6 contains exactly*

$$\binom{6}{2} 2^4 = 240$$

axis-aligned local Q_2 faces.

Proof. Choose the two free bit positions in $\binom{6}{2}$ ways. The remaining four bit positions are fixed, each independently to 0 or 1, giving 2^4 choices. Thus the number is $\binom{6}{2}2^4 = 15 \cdot 16 = 240$. □

Theorem 9462.1160 (Standard-code axis-aligned $3+1$ face enumeration)**.** *Under the two-axis readback and the standard translation label map, the axis-aligned Q_2 faces with a $3+1$ translation-class split are exactly the following four faces:*

face	*three-point class*	*singleton class*	*singleton corner*
$\{\mathrm{UUU}, \mathrm{UUA}, \mathrm{CUU}, \mathrm{CUA}\}$	Leu	Phe	00
$\{\mathrm{UUC}, \mathrm{UUG}, \mathrm{CUC}, \mathrm{CUG}\}$	Leu	Phe	00
$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\}$	Stop	Trp	11
$\{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}, \mathrm{AUG}\}$	Ile	Met	11.

Proof. The enumeration is finite. By Definition 9462.1159, there are 240 axis-aligned Q_2 faces. For each face, decode its four vertices through β^{-1}, apply τ_{std}, and retain exactly those faces whose label multiplicities are 3 and 1. Direct enumeration yields the four displayed rows. The table is independent of any BEDC biological axiom beyond the external standard-code label map. □

Corollary 9462.1161 (Two corner mechanisms)**.** *The four standard-code $3+1$ faces split into two corner mechanisms: two* 11*-control tiles,* Ile/Met *and* Stop/Trp*, and two* 00*-singleton tiles, both* Phe/Leu*.*

Proof. Read the singleton-corner column in Definition 9462.1160. □

Theorem 9462.1162 (Total local tile theorem)**.** *Every standard-code axis-aligned* $3+1$ *split in the two-axis codon cube is one of the following:*

tile mechanism	*codon split*
11-*control*	AUU, AUC, AUA | AUG
11-*control*	UAA, UAG, UGA | UGG
00-*singleton*	UUU | UUA, CUU, CUA
00-*singleton*	UUC | UUG, CUC, CUG.

Proof. The four rows are exhaustive by Definition 9462.1160. Their mechanism labels follow from Definition 9462.1161. □

Theorem 9462.1163 (Reserved-corner interpretation)**.** *In the two* 11-*control tiles, the local* 11 *corner is not a claim of biological absence. It is a separated singleton or control-sensitive corner.*

Proof. The local 11 corners are actual standard-code codons:

$$\begin{aligned} &\mathrm{AUG} = \mathrm{Met} \quad \text{and may carry the canonical start row,} \\ &\mathrm{UGG} = \mathrm{Trp}. \end{aligned}$$

Therefore the bridge-level reading cannot be that local 11 is absent from biology. The licensed reading is that the local 11 corner is separated from the ordinary three-point class. □

Theorem 9462.1164 (Stop structure is local, not global)**.** *The standard stop set is local* X_2 *inside the* $\mathrm{U}+\mathrm{RR}$ *face. It is not the global* X_6 *no-adjacent language on flattened codons.*

Proof. The flattened displays are

$$\begin{aligned} \mathrm{UAA} &= 001010, \\ \mathrm{UAG} &= 001011, \\ \mathrm{UGA} &= 001110. \end{aligned}$$

The word 001010 has no adjacent 11, but 001011 and 001110 do. Hence the standard stop set is not contained in global X_6. By Definition 9462.1154, the stop structure is instead the local face equality

$$\mathrm{Stop} = \mathrm{U} + \mathrm{RR} + X_2.$$

□

Corollary 9462.1165 (Correct window-six reading)**.** *The correct finite-window reading is*

$$\mathsf{CodonCube} = Q_6, \qquad \mathrm{Stop} = \mathrm{U} + \mathrm{RR} + X_2,$$

not $\mathrm{Stop} = X_6$.

Proof. The codon-cube equality is Definition 9462.1152; the local stop equality is Definition 9462.1154; the global reading is rejected by Definition 9462.1164. □

Definition 9462.1166 (Complete global fiber $3+1$ tile)**.** *Let* $E : Q_6 \to C$ *be a translation classifier. An axis-aligned* Q_2 *face* F *is a complete global fiber* $3+1$ *tile when there are classes* $a, b \in C$ *such that*

$$F = E^{-1}(a) \sqcup E^{-1}(b), \qquad |E^{-1}(a)| = 3, \qquad |E^{-1}(b)| = 1.$$

Thus the local three-point side is not merely a local multiplicity: it is the whole global tripleton fiber. Likewise the singleton corner is the whole global singleton fiber.

Theorem 9462.1167 (Complete global fiber tiles in the standard code)**.** *Under the two-axis readback and the standard translation classifier, the complete global fiber* $3+1$ *tiles are exactly*

face	*global tripleton*	*global singleton*	*singleton corner*
{AUU, AUC, AUA, AUG}	Ile	Met	11
{UAA, UAG, UGA, UGG}	Stop	Trp	11.

Proof. By Definition 9462.1859, the only global tripleton classes of the standard classifier are Ile and Stop, and the only global singleton classes are Met and Trp. The exhaustive local Q_2 enumeration of Definition 9462.1160 has four $3+1$ faces. The two Phe/Leu faces are excluded because Phe is a global doubleton and Leu is a global sextupleton. The remaining two faces are the displayed Ile/Met and Stop/Trp faces, and both have singleton corner 11. □

Corollary 9462.1168 (Global fibers select the dual control tiles)**.** *The pair* Ile/Met *and* Stop/Trp *is not selected by choosing two examples from the four local* $3+1$ *faces. It is selected by imposing the global-fiber condition on the local* Q_2 *spectrum.*

Proof. The global-fiber condition eliminates the Phe/Leu local slices and keeps exactly the two rows of Definition 9462.1167. □

Theorem 9462.1169 (Standard-code Q_1 edge spectrum)**.** *The standard translation classifier has* 50 *same-class Hamming edges and* 142 *cross-class Hamming edges in* Q_6*. By bit direction*

$$(b_1^s, b_1^f, b_2^s, b_2^f, b_3^s, b_3^f),$$

the same-class edge counts are

direction	b_1^s	b_1^f	b_2^s	b_2^f	b_3^s	b_3^f
same-class edges	0	2	0	1	17	30
cross-class edges	32	30	32	31	15	2.

Proof. For each of the six bit directions, enumerate the 32 Hamming edges $x \leftrightarrow x \oplus e_i$. Decode both endpoints to codons and compare their standard translation labels. The displayed table is the finite count over the resulting $6 \cdot 32 = 192$ edges. □

Corollary 9462.1170 (Third-position softness)**.** *The third-position family bit* b_3^f *is the most stable Hamming direction in the standard classifier, with* 30 *same-class edges out of* 32*. The first and second shape bits have no same-class edge.*

Proof. Read the direction counts in Definition 9462.1169. □

Theorem 9462.1171 (Same-histogram random edge baseline)**.** *Keep only the standard-code degeneracy histogram*

$$1:2, \qquad 2:9, \qquad 3:2, \qquad 4:5, \qquad 6:3,$$

and randomly assign the resulting fibers to the 64 *codons. For a random unordered pair of distinct codons, the same-class probability is*

$$p_{\text{same}} = \frac{\sum_a \binom{d_a}{2}}{\binom{64}{2}} = \frac{5}{112}.$$

Thus the expected number of same-class Hamming edges among the 192 *edges of* Q_6 *is*

$$192\, p_{\text{same}} = \frac{60}{7} \approx 8.57.$$

The observed standard-code value is 50*, namely*

$$\frac{50}{60/7} = \frac{35}{6} \approx 5.83$$

times the same-histogram random expectation. The exact variance of the same-edge count under this same-histogram model is

$$\frac{686964}{92659} \approx 7.41,$$

so the standard deviation is about 2.72 *and the observed value has z-score about* 15.2*.*

Proof. The standard degeneracy histogram gives

$$\sum_a \binom{d_a}{2} = 2\binom{1}{2} + 9\binom{2}{2} + 2\binom{3}{2} + 5\binom{4}{2} + 3\binom{6}{2} = 90.$$

Since $\binom{64}{2} = 2016$, the probability is $90/2016 = 5/112$. Multiplying by the 192 Hamming edges gives $60/7$. The observed count 50 is Definition 9462.1169. For the variance, separate unordered pairs of Hamming edges into adjacent edge-pairs, of which there are $64\binom{6}{2} = 960$, and disjoint edge-pairs, of which there are $\binom{192}{2} - 960 = 17376$. Evaluating the two corresponding same-fiber probabilities from the fixed fiber sizes gives the displayed finite variance. □

Corollary 9462.1172 (Local edge smoothness is not a histogram artifact)**.** *The standard classifier is not merely a code table with the listed global degeneracy histogram. It is locally smooth on the Hamming graph: same-class adjacency is about* 5.83 *times the same-histogram random expectation, and the smoothness is concentrated in the third-base directions.*

Proof. The excess over the histogram baseline is Definition 9462.1171. The directional concentration is Definition 9462.1170. □

Theorem 9462.1173 (Single-nucleotide substitution edge spectrum)**.** *If edges are taken to be ordinary one-base substitutions rather than one-bit flips, then there are* 288 *unordered edges. The standard classifier has the following position-wise spectrum:*

codon position	1	2	3	*total*
same-class substitution edges	4	1	64	69
cross-class substitution edges	92	95	32	219.

Proof. For each codon position, fix the other two bases in 4^2 ways and connect the four choices at the chosen position by the complete graph on four bases. This gives $16\binom{4}{2} = 96$ edges per position and 288 total edges. Decoding both endpoints and comparing their standard translation labels gives the displayed finite spectrum. □

Corollary 9462.1174 (Two-axis stability matches nucleotide substitution)**.** *The two-axis* Q_6 *edge spectrum is not only a formal bit-cube phenomenon. Ordinary one-base substitution edges show the same asymmetry: the third codon position has* 64 *same-class edges out of* 96*, while the first two positions have only* 4 *and* 1*.*

Proof. Compare Definition 9462.1169 with Definition 9462.1173. □

Theorem 9462.1175 (Standard-code Q_2 face spectrum)**.** *The* 240 *axis-aligned* Q_2 *faces of the standard classifier have the following multiplicity spectrum:*

local pattern	4	$3+1$	$2+2$	$2+1+1$	$1+1+1+1$
face count	9	4	83	44	100.

By geometric type, the same spectrum refines to

face type	4	$3+1$	$2+2$	$2+1+1$	$1+1+1+1$	*total*
same base	8	1	6	6	27	48
cross-base same axis	1	1	39	18	37	96
cross-base mixed axis	0	2	38	20	36	96.

Proof. The first table is the exhaustive face enumeration already used in Definition 9462.1160, with all multiplicity patterns retained rather than only the $3+1$ rows. The second table partitions the same faces by whether the two free bits lie in the same base, in different bases along the same axis, or in different bases along different axes. □

Theorem 9462.1176 (Standard-code Q_3 cube spectrum)**.** *The* 160 *axis-aligned* Q_3 *cubes of the standard classifier have the following multiplicity spectrum:*

local pattern	*cube count*
$2+2+2+2$	46
$1+1+1+1+1+1+1+1$	21
$2+1+1+1+1+1+1$	21
$2+2+2+1+1$	18
$2+2+1+1+1+1$	11
$4+2+2$	10
$4+4$	10
$3+2+2+1$	9
$3+2+1+1+1$	5
$4+2+1+1$	4
$3+1+1+1+1+1$	2
$4+3+1$	2
$6+2$	1.

Proof. Choose three free bit positions and fix the remaining three bits, giving $\binom{6}{3}2^3 = 160$ cubes. On each cube, decode the eight vertices and count the translation-label multiplicities. The displayed table is the resulting finite spectrum. □

Corollary 9462.1177 (Phe/Leu is the unique $6+2$ cube)**.** *The unique* $6+2$ *cube is*

$$\{\mathrm{UUU}, \mathrm{UUC}, \mathrm{UUA}, \mathrm{UUG}, \mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\},$$

with $\{\mathrm{UUU}, \mathrm{UUC}\} = \mathrm{Phe}$ *and the remaining six codons classified as* Leu.

Proof. The exhaustive Q_3 enumeration in Definition 9462.1176 contains one cube of type $6+2$. Its vertices are the displayed Phe/Leu support. □

Theorem 9462.1178 (Neighborhoods of the complete control tiles)**.** *The Stop/Trp tile is contained in four axis-aligned Q_3 cubes, and all four have pattern $3+2+2+1$. The Ile/Met tile is contained in four axis-aligned Q_3 cubes; two have pattern $3+2+2+1$, and two have pattern $4+3+1$.*

Proof. Every Q_2 face in Q_6 has four one-bit extensions to a Q_3 cube. Enumerating those four extensions for the Stop/Trp face gives four cubes whose labels are

$$\mathrm{Stop}_3 + \mathrm{Trp}_1 + \text{doubleton} + \text{doubleton}.$$

Enumerating the four extensions for the Ile/Met face gives two cubes of the same $3+2+2+1$ form and two cubes of the form

$$\text{quadrupleton} + \mathrm{Ile}_3 + \mathrm{Met}_1.$$

□

Corollary 9462.1179 (Control-tile neighborhood contrast)**.** *The Stop/Trp tile is locally isolated as a boundary/continuation tile in all of its Q_3 neighborhoods. The Ile/Met tile retains the same 11-corner control form, but two of its Q_3 neighborhoods embed it beside a four-codon sense block.*

Proof. This is the distinction between the two neighborhood spectra in Definition 9462.1178. □

Concretized in §3778: Global fiber spectra are realized as finite codon-cube fiber, tile, edge, trigger, transport, replay, provenance, and NameCert rows.

Definition 9462.1180 (Synonymous induced graph)**.** *Let $E : Q_6 \to C$ be the standard translation classifier. For a class $a \in C$, write G_a for the subgraph of the Hamming graph on Q_6 induced by the fiber $E^{-1}(a)$.*

Theorem 9462.1181 (Synonymous induced graph types)**.** *The standard classifier has exactly the following synonymous component types:*

component type	*classes*
1	Met, Trp
2	Phe, Tyr, Cys, His, Gln, Asn, Lys, Asp, Glu
3	Ile, Stop
4	Pro, Thr, Val, Ala, Gly
$4+2$	Ser, Arg
6	Leu.

Proof. For each translation class, restrict the Q_6 Hamming graph to the codons in that class and compute connected component sizes. The standard table gives the displayed finite list. □

Corollary 9462.1182 (Dual tripleton shape)**.** *The two tripleton classes* Ile *and* Stop *have the same induced X_2 graph shape. In local coordinates they are*

$$\{00, 01, 10\},$$

with Hamming path $01 - 00 - 10$.

Proof. The tripleton component type in Definition 9462.1181 is connected of size 3. Substituting the coordinates of the Ile/Met and Stop/Trp faces from Definition 9462.1167 gives the displayed X_2 shape. □

Theorem 9462.1183 (Internal edge and boundary spectrum)**.** *For each class a, let*

$$e_a = |\{x \sim y : E(x) = E(y) = a\}|.$$

The internal edge and external boundary degrees are:

class type	$\lvert E^{-1}(a)\rvert$	e_a	$6\lvert E^{-1}(a)\rvert - 2e_a$
Met, Trp	1	0	6
doubleton classes	2	1	10
Ile, Stop	3	2	14
Pro, Thr, Val, Ala, Gly	4	4	16
Ser, Arg	6	5	26
Leu	6	7	22.

Proof. Enumerate the 192 Hamming edges of Q_6 and count those whose two endpoints lie in the same translation class. The boundary degree formula subtracts twice the internal edges from the total incident degree $6|E^{-1}(a)|$. □

Corollary 9462.1184 (Leu is the tight sextupleton)**.** *Among the three sextupleton classes,* Leu *has the largest internal Hamming edge count:*

$$e_{\text{Leu}} = 7, \qquad e_{\text{Ser}} = e_{\text{Arg}} = 5.$$

Thus the Phe/Leu block is a sense expansion geometry rather than a halt/continue control geometry.

Proof. Read the sextupleton rows of Definition 9462.1183 and compare with the unique $6 + 2$ Q_3 cube described in Definition 9462.1176. □

Definition 9462.1185 (Class adjacency and coarse bit-flip transition)**.** *The class adjacency matrix is*

$$A_{ab} = |\{x \sim y : E(x) = a,\ E(y) = b\}|.$$

For the coarse one-bit transition from class a to class b, set

$$P(b \mid a) = \frac{|\{x \in E^{-1}(a),\ y \sim x,\ E(y) = b\}|}{6|E^{-1}(a)|}.$$

Theorem 9462.1186 (Largest class-adjacency weights)**.** *The largest off-diagonal entries of the*

class adjacency matrix are

class pair	A_{ab}
Ser−Thr	6
Gly−Arg	6
Phe−Leu	4
Pro−Ser	4
Cys−Ser	4
Leu−Val	4
Leu−Pro	4
Ala−Pro	4
Pro−Arg	4
Ala−Thr	4
Ala−Val	4
Ala−Gly	4.

The two control pairs satisfy

$$A_{\mathrm{Ile,Met}} = 2, \qquad A_{\mathrm{Stop,Trp}} = 2.$$

Proof. Each Hamming edge is decoded at both endpoints and then accumulated by the unordered pair of translation classes. The displayed rows are the largest finite counts; the two control-pair counts are the corresponding entries for the singleton/tripleton contacts. □

Corollary 9462.1187 (Control pairs are corner contacts)**.** *The Ile/Met and Stop/Trp pairs are not maximal by adjacency weight. Their distinction is geometric: each is a singleton* 11 *corner adjacent to a tripleton X_2 fiber.*

Proof. The weights are only 2 by Definition 9462.1186. The global singleton/tripleton 11-corner geometry is Definition 9462.1167. □

Theorem 9462.1188 (Dual singleton transition probabilities)**.** *The coarse one-bit transition probabilities satisfy*

$$P(\mathrm{Ile} \mid \mathrm{Met}) = \frac{2}{6} = \frac{1}{3}, \qquad P(\mathrm{Stop} \mid \mathrm{Trp}) = \frac{2}{6} = \frac{1}{3}.$$

Moreover

$$P(\mathrm{Met} \mid \mathrm{Ile}) = \frac{2}{18} = \frac{1}{9}, \qquad P(\mathrm{Trp} \mid \mathrm{Stop}) = \frac{2}{18} = \frac{1}{9}.$$

Proof. The Met singleton has six one-bit neighbors, two of which lie in the Ile fiber. The Trp singleton has six one-bit neighbors, two of which lie in the Stop fiber. Reversing direction divides the same two boundary contacts by the total directed degree 18 of each tripleton fiber. □

Definition 9462.1189 (Q_4 neighborhood of a tile)**.** *An axis-aligned Q_4 cube is obtained by fixing two bits of Q_6 and freeing the remaining four. A Q_2 tile has six direct Q_4 neighborhoods, one for each choice of two additional free bits.*

Theorem 9462.1190 (Standard-code Q_4 spectrum)**.** *The* 60 *axis-aligned Q_4 cubes of the standard classifier have maximal within-cube class sizes distributed as*

$\max_a \lvert Q_4 \cap E^{-1}(a)\rvert$	2	3	4	6
cube count	17	15	23	5.

The complete sextupleton appearances inside Q_4 cubes are

$$\mathrm{Leu} : 3, \qquad \mathrm{Ser} : 1, \qquad \mathrm{Arg} : 1.$$

Proof. Enumerate the $\binom{6}{4}2^2 = 60$ axis-aligned Q_4 cubes. For each cube count the multiplicities of translation labels inside it and record the maximum. A complete sextupleton appearance is counted when all six codons of a global six-codon class lie in the cube. □

Theorem 9462.1191 (Dual control Q_4 neighborhoods)**.** *All six Q_4 neighborhoods of the Stop/Trp tile retain* $\mathrm{Stop}_3 + \mathrm{Trp}_1$*. All six Q_4 neighborhoods of the Ile/Met tile retain* $\mathrm{Ile}_3 + \mathrm{Met}_1$*. However, the Ile/Met neighborhoods also include ordinary large sense blocks, including*

$$\mathrm{Ala}_4 + \mathrm{Thr}_4 + \mathrm{Val}_4 + \mathrm{Ile}_3 + \mathrm{Met}_1$$

and

$$\mathrm{Leu}_6 + \mathrm{Val}_4 + \mathrm{Ile}_3 + \mathrm{Phe}_2 + \mathrm{Met}_1.$$

Proof. Enumerate the six Q_4 cubes containing each displayed Q_2 tile and record their translation-class multiplicities. The Stop/Trp neighborhoods always keep the same Stop_3 and Trp_1 core. The Ile/Met neighborhoods keep the Ile_3 and Met_1 core, while two of the neighborhoods contain the displayed larger sense blocks. □

Corollary 9462.1192 (Boundary tile versus entry tile)**.** *Stop/Trp is a locally closed halt/continue boundary through the Q_3 and Q_4 neighborhoods. Ile/Met has the same local $X_2/11$ shape, but its higher neighborhoods attach it to ordinary four-codon and six-codon sense regions.*

Proof. Combine the Q_3 neighborhood counts in Definition 9462.1176 with the Q_4 neighborhood statement in Definition 9462.1191. □

Definition 9462.1193 (Control half-cube)**.** *A Q_5 half-cube in Q_6 is obtained by fixing one bit and freeing the other five bits.*

Theorem 9462.1194 (Unique shared control half-cube)**.** *The Ile/Met tile and the Stop/Trp tile share exactly one fixed-bit condition:*

$$b_1^f = 0.$$

Thus both lie in the unique half-cube whose first codon base has family bit 0*, equivalently whose first base lies in* $\{\mathrm{U}, \mathrm{A}\}$*. Inside this Q_5 half-cube, the class multiplicities are*

$$\mathrm{Ser}_6,\ \mathrm{Thr}_4,\ \mathrm{Stop}_3,\ \mathrm{Ile}_3,$$
$$\mathrm{Phe}_2,\ \mathrm{Leu}_2,\ \mathrm{Tyr}_2,\ \mathrm{Cys}_2,\ \mathrm{Asn}_2,\ \mathrm{Lys}_2,\ \mathrm{Arg}_2,\ \mathrm{Trp}_1,\ \mathrm{Met}_1.$$

Proof. The Stop/Trp tile has first base $\mathrm{U} = 00$, while the Ile/Met tile has first base $\mathrm{A} = 10$. The two first-base shape bits differ, but their first-base family bit is the common value 0. Checking the remaining four bit coordinates across the eight tile vertices shows no other common fixed coordinate. Enumerating the resulting 32 vertices gives the displayed multiplicities. □

Corollary 9462.1195 (Co-localized dual control tiles)**.** *The two $X_2/11$ control tiles are not isolated coincidences in Q_6. They are co-localized in the single half-cube $b_1^f = 0$, where the two tripleton fibers and the two singleton fibers appear together:*

$$\mathrm{Stop}_3, \quad \mathrm{Ile}_3, \quad \mathrm{Trp}_1, \quad \mathrm{Met}_1.$$

Proof. This is the control-class sublist of Definition 9462.1194. □

Corollary 9462.1196 (Half-cube reading of dual 10-to-11 reassignment)**.** *The alternative assignments* AUA $\to$ Met *and* UGA $\to$ Trp *have the same local form inside the shared half-cube:*

$$\{00, 01, 10\} \mid \{11\} \rightsquigarrow \{00, 01\} \mid \{10, 11\}.$$

Proof. By Definition 9462.1197, both assignments move the local 10 vertex into the class containing the 11 corner. By Definition 9462.1194, the two instances lie in the same $b_1^f = 0$ half-cube. □

Concretized in §5799: The synonymous transition graph is realized as finite codon-window, class-adjacency, boundary-spectrum, and trigger-threshold NameCert data.

Proposition 9462.1197 (Dual 10-to-11 reassignment)**.** *In genetic-code tables where* AUA *is assigned to* Met *and* UGA *is assigned to* Trp*, the two dual* 11*-control tiles undergo the same displayed deformation:*

$$\{00, 01, 10\}/\{11\} \rightsquigarrow \{00, 01\}/\{10, 11\}.$$

Proof. In the AU tile, AUA has local coordinate 10 and AUG has coordinate 11. In the U + RR tile, UGA has local coordinate 10 and UGG has coordinate 11. Assigning AUA with AUG or assigning UGA with UGG therefore has the same local displayed form: the 10 leaf joins the class containing the 11 corner. □

Corollary 9462.1198 (AUA-to-Met and UGA-to-Trp share a tile form)**.** *The reassignments* AUA $\to$ Met *and* UGA $\to$ Trp *are structurally the same local tile operation under the two-axis coordinates:* 10 *joins the class containing* 11*.*

Proof. This is Definition 9462.1197 applied to the two distinguished 11-control tiles. □

Proposition 9462.1199 (Stop-arm reassignment)**.** *In genetic-code tables where* UAA *and* UAG *are assigned to a sense class while* UGA *remains a termination row, the stop tile displays an arm reassignment:*

$$\{00, 01\} \quad \textit{moves as one stop-graph arm.}$$

Proof. In the stop graph, UAA has coordinate 00, UAG has coordinate 01, and UGA has coordinate 10. The pair $\{00, 01\}$ is one Hamming edge of the local L-shape. If UAA and UAG are assigned together to a sense class while UGA remains termination, the local deformation moves that edge as a unit. □

Corollary 9462.1200 (Stop-tile deformation modes)**.** *Alternative-code changes of the stop tile can be recorded at least as leaf reassignment, arm reassignment, or context-sensitive partial boundary assignment.*

Proof. The leaf case is represented by Definition 9462.1197. The arm case is Definition 9462.1199. Context-sensitive or ambiguous stop rows are recorded by allowing the biological bridge table to attach a partial or context-indexed classifier to one of the local vertices. □

Theorem 9462.1201 (Release-factor recognition is the stop graph)**.** *For the bacterial release-factor bridge,*

$$\mathsf{RF1Arm} = \{\mathrm{UAA}, \mathrm{UAG}\} = \{00, 01\},$$
$$\mathsf{RF2Arm} = \{\mathrm{UAA}, \mathrm{UGA}\} = \{00, 10\}.$$

Thus RF1/RF2 recognition traces the two arms of the stop X_2 graph.

Proof. Substitute the stop graph coordinates from Definition 9462.1149. The two sets $\{00, 01\}$ and $\{00, 10\}$ are exactly the two Hamming-distance-one edges of the L-shape $01 - 00 - 10$. □

Concretized in §5818: The finite translation-boundary packet realizes the release-factor stop-graph route as validation-row NameCert data.

Corollary 9462.1202 (UAA is the local stop center)**.** *The intersection of the two release-factor arms is*

$$\mathsf{RF1Arm} \cap \mathsf{RF2Arm} = \{\mathrm{UAA}\}.$$

Proof. Compute

$$\{\mathrm{UAA}, \mathrm{UAG}\} \cap \{\mathrm{UAA}, \mathrm{UGA}\} = \{\mathrm{UAA}\}.$$

□

Theorem 9462.1203 (Stop tile is a local halt boundary)**.** *Under the local translation-state model,*

$$\mathrm{Stop} = \mathrm{U} + \mathrm{RR} + X_2 = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$$

is a local halt boundary.

Proof. The stop equality is Definition 9462.1154. By Definition 9462.1151, any state whose current window lies in the stop class transitions to S_{halt}. □

Corollary 9462.1204 (UGG is a continuation corner)**.** *The codon* $\mathrm{UGG} = \mathrm{U}+\mathrm{RR}+11$ *is adjacent to the stop boundary in the stop face, but the standard-code bridge assigns it to* Trp *rather than to* Stop*.*

Proof. By Definition 9462.1155, UGG is the 11 corner of the stop face. By Definition 9462.1154, the other three vertices are the stop class. The standard-code bridge assigns UGG to Trp, so its transition is elongation rather than termination. □

Definition 9462.1205 (Finite formalization target)**.** *The checked finite target is the exhaustive enumeration:*

$$\mathsf{AxisQ2Faces}(\Sigma^3, b) \xrightarrow{\tau_{\mathsf{std}}} \mathsf{ThreeOneFaces},$$
$$\mathsf{ThreeOneFaces} = \{\mathsf{PheLeuFace}_{\mathrm{U}}, \mathsf{PheLeuFace}_{\mathrm{C}}, \mathsf{StopTrpFace}, \mathsf{AUFace}\}.$$

The intended Lean proof is finite enumeration over the 240 *axis-aligned* Q_2 *faces, not a biological necessity theorem.*

Theorem 9462.1206 (Local tile cannot-claim boundary)**.** *The local-tile bridge does not license any claim that DNA must obey Zeckendorf, that the genetic code is the window-six fold, that the genetic code is globally* X_6*, that* $\mathrm{Stop} = X_6$*, that* 11 *is biologically illegal or absent, that a local DNA or RNA* 11 *corner is the BEDC channel terminator, that the standard genetic code is logically necessary, that* UAA*,* UAG*, and* UGA *are universal stop codons, that* UGG *is tryptophan by BEDC logic alone, that alternative-code reassignments are forced by pure mathematics, or that* $\mathsf{LocalTileDynamics}^{\uparrow}$ *is already a checked certificate.*

Proof. The global X_6 refusals follow from Definition 9462.1164. The channel-token refusal follows from Definition 9462.1107. The biological label refusals follow because the standard and alternative genetic-code tables are external bridge data. The checked-status refusal follows from Definition 9462.1141. □

Corollary 9462.1207 (Allowed local-tile claim)**.** *Under the two-axis readback and the standard-code table, codons form Q_6, and the axis-aligned Q_2 faces with $3+1$ translation-class splits are exactly the two* 11*-control tiles* Ile/Met *and* Stop/Trp *together with the two* 00*-singleton* Phe/Leu *tiles. This is a local finite-window classification inside Q_6, not a global identification of the genetic code with X_6 or with a BEDC channel code.*

Definition 9462.1208 (Alternative-code reassignment spectrum)**.** *Let $E_1 : Q_6 \to C_1$ be the standard genetic-code classifier and let $E_t : Q_6 \to C_t$ be another curated genetic-code table on the same codon carrier. The reassignment set is*

$$\Delta_t := \{c \in Q_6 : E_t(c) \neq E_1(c)\}.$$

For a family of tables, the reassignment spectrum is the finite multiset of all codon rows in all Δ_t, grouped by local codon module.

Concretized in §3589: Alternative-code reassignment spectra are realized as finite standard-code comparison, module ledger, sparse-support, transport, provenance, and NameCert rows.

Definition 9462.1209 (Four reassignment modules)**.** *The comparison ledger uses four local modules:*

$$\begin{aligned} M_{\text{Stop/Trp}} &= \{\text{UAA}, \text{UAG}, \text{UGA}, \text{UGG}\}, \\ M_{\text{Ile/Met}} &= \{\text{AUU}, \text{AUC}, \text{AUA}, \text{AUG}\}, \\ M_{\text{Phe/Leu}} &= \{\text{UUU}, \text{UUC}, \text{UUA}, \text{UUG}, \text{CUU}, \text{CUC}, \text{CUA}, \text{CUG}\}, \\ M_{\text{AGA/AGG}} &= \{\text{AGA}, \text{AGG}\}. \end{aligned}$$

The first two are the complete global $3+1$ control tiles of Definition 9462.1167; the third is the unique Phe/Leu $6+2$ cube of Definition 9462.1177; the fourth is a two-codon Arg-side pair that becomes visible in comparative tables.

Theorem 9462.1210 (Alternative-code reassignment concentration)**.** *Across the curated non-standard genetic-code tables compared with the standard table, the reassignment spectrum has* 65 *codon rows. Grouped by the modules of Definition 9462.1209, the counts are*

module	*reassignment rows*	*tables involved*
$M_{\text{Stop/Trp}}$	33	22
$M_{\text{AGA/AGG}}$	16	8
$M_{\text{Phe/Leu}}$	7	4
$M_{\text{Ile/Met}}$	5	5
other codons	4	4.

Proof. For each non-standard table, compare all 64 codon assignments with the standard classifier and record exactly those codons whose label changes. Classifying the resulting 65 rows by Definition 9462.1209 gives the displayed finite table. The enumeration is bridge-side data over curated genetic-code tables; it is not a BEDC derivation of those biological assignments. □

Corollary 9462.1211 (Stop/Trp is the main deformation face)**.** *The Stop/Trp tile contains only four codons, but it accounts for* 33 *of the* 65 *reassignment rows. Thus it is the dominant local deformation face in the comparative spectrum.*

Proof. This is the first row of Definition 9462.1210. □

Theorem 9462.1212 (Sparse reassignment codon support)**.** *Across the comparison family, the codons that are ever reassigned relative to the standard classifier are exactly*

$$\text{UGA, UAG, UAA, AUA, AGA, AGG, AAA,}$$
$$\text{CUG, CUU, CUC, CUA, UCA, UUA.}$$

With multiplicity over tables, the codon counts are

codon	*standard label*	*count*	*observed target labels*
UGA	Stop	15	Trp, Cys, Gly, Stop|Trp
UAG	Stop	10	Gln, Leu, Tyr, Glu, Trp, partial
AGA	Arg	8	Stop, Ser, Gly
AGG	Arg	8	Stop, Ser, Gly, Lys
UAA	Stop	8	Gln, Tyr, Glu, partial
AUA	Ile	5	Met
CUG	Leu	3	Thr, Ser, Ala
AAA	Lys	3	Asn
CUU, CUC, CUA	Leu	1 *each*	Thr
UCA	Ser	1	Stop
UUA	Leu	1	Stop.

Proof. For each non-standard table, compare all 64 rows against the standard classifier, retaining partial labels as distinct target labels when the bridge table records a partial stop/sense assignment. Taking the union of changed codons gives the displayed 13-codon support; keeping table multiplicity gives the displayed counts. □

Corollary 9462.1213 (Comparative variation is sparse)**.** *The comparison changes only* 13 *of the* 64 *codons relative to the standard classifier. Thus the observed family is a sparse reassignment family rather than a global reshuffling of the codon carrier.*

Proof. This is the support size in Definition 9462.1212. □

Theorem 9462.1214 (Control half-cube concentration)**.** *Let*

$$Q_5^{\mathrm{ctrl}} = \{x \in Q_6 : b_1^f(x) = 0\}.$$

Among the 65 *reassignment rows in the comparison family,* 59 *lie in* Q_5^{ctrl} *and* 6 *lie outside it:*

region	*reassignment rows*
$b_1^f = 0$	59
$b_1^f = 1$	6.

Equivalently, 59/65 *of the reassignment rows lie in the same half-cube that contains the dual Ile/Met and Stop/Trp control tiles.*

Proof. For each reassignment row, decode the codon to its six-bit word and read the first-base family bit. The finite count gives 59 rows with $b_1^f = 0$ and 6 rows with $b_1^f = 1$. The identification of this half-cube with the shared dual-control half-cube is Definition 9462.1194. □

Corollary 9462.1215 (The shared half-cube is the comparative working region)**.** *The half-cube selected by the standard-code dual control tiles is also the main region of observed comparative reassignment. In this sense* $b_1^f = 0$ *is both a standard-code control half-cube and the dominant comparative deformation region.*

Proof. The standard-code half-cube is Definition 9462.1194. Its comparative concentration is Definition 9462.1214. □

Theorem 9462.1216 (Refined reassignment module counts)**.** *The* 65 *reassignment rows decompose into the following refined local modules:*

module	*codons*	*rows*
Stop/Trp X_2	UAA, UAG, UGA	33
AGR	AGA, AGG	16
Ile/Met leaf	AUA	5
AAA leaf	AAA	3
rare stop insertions	UCA, UUA	2
CUN Leu	CUU, CUC, CUA, CUG	6.

The first three rows contribute 54 *of the* 65 *reassignment rows.*

Proof. Group the codon counts in Definition 9462.1212 by the displayed codon sets and sum within each set. □

Corollary 9462.1217 (Main comparative axis)**.** *The dominant comparative axis is*

$$\text{Stop/Trp} \;+\; \text{AGR} \;+\; \text{Ile/Met}.$$

It accounts for 54/65 *reassignment rows, while the remaining rows are confined to the AAA leaf, two rare stop insertions, and the CUN Leu module.*

Proof. This is the sum of the first three rows in Definition 9462.1216. □

Theorem 9462.1218 (Connected reassignment support)**.** *Let R be the set of the* 13 *codons that are ever reassigned in the comparison family. The induced Hamming subgraph $Q_6[R]$ is connected and has* 16 *internal edges.*

Proof. Take the support set from Definition 9462.1212. Enumerate all Hamming edges whose two endpoints lie in this set. The finite enumeration gives 16 such edges. A breadth-first search from any vertex reaches all 13 codons. □

Corollary 9462.1219 (Reassignment support is a connected zone)**.** *The reassigned codons are not scattered as isolated vertices of Q_6. They form one connected local deformation zone, whose weighted mass is concentrated in the control half-cube $b_1^f = 0$.*

Proof. Connectivity is Definition 9462.1218. Weighted half-cube concentration is Definition 9462.1214. □

Theorem 9462.1220 (Stop/Trp 11 corner is invariant across the comparison)**.** *In the Stop/Trp tile, all observed reassignments occur in the X_2 side:*

local corner	*codon*	*standard label*	*reassignment rows*
00	UAA	Stop	8
01	UAG	Stop	10
10	UGA	Stop	15
11	UGG	Trp	0.

Proof. Restrict the reassignment spectrum to the four codons UAA, UAG, UGA, and UGG. The finite count gives the displayed rows. In particular, the local 11 corner UGG remains assigned to Trp throughout the comparison. □

Corollary 9462.1221 (The Stop/Trp 11 corner is continuation, not absence)**.** *The comparative spectrum strengthens the reserved-corner reading of Definition 9462.1163: inside the Stop/Trp tile,* 11 *is a stable continuation corner, while the* X_2 *side carries the observed stop-boundary deformations.*

Proof. The standard-code interpretation of UGG is continuation by Definition 9462.1204. Its reassignment count is zero by Definition 9462.1220. □

Theorem 9462.1222 (Coupled 10-to-11 control-tile reassignment)**.** *The only observed Ile/Met tile reassignment is*

$$\mathrm{AUA} : \mathrm{Ile} \to \mathrm{Met}.$$

It occurs in exactly the tables

$$2, \quad 3, \quad 5, \quad 13, \quad 21.$$

Each of these tables also has

$$\mathrm{UGA} : \mathrm{Stop} \to \mathrm{Trp}.$$

In local coordinates, both changes have the form

$$10 \textit{ joins the class containing } 11.$$

Proof. In the Ile/Met tile, AUA = AU + 10 and AUG = AU + 11. In the Stop/Trp tile, UGA = U + RR + 10 and UGG = U + RR + 11. The finite comparison of non-standard tables shows that the AUA-to-Met row occurs exactly in the displayed five table identifiers, and each of those identifiers also contains the UGA-to-Trp row. The local coordinate action is therefore the same in both complete control tiles. □

Corollary 9462.1223 (Dual control tiles are coupled in observed variation)**.** *The Ile/Met and Stop/Trp complete control tiles are not merely parallel standard-code structures. In the comparative spectrum, the observed Ile/Met deformation is always accompanied by the corresponding Stop/Trp* 10-*to*-11 *deformation.*

Proof. This is the inclusion of table identifiers established in Definition 9462.1222. □

Theorem 9462.1224 (Control-tile pattern spectra across comparison tables)**.** *Across the comparison family, the Ile/Met tile has exactly two partial-aware local patterns:*

Ile/Met pattern on (AUU, AUC, AUA, AUG)	*table count*
$I/I/I/M$	22
$I/I/M/M$	5.

The Stop/Trp tile has 14 *partial-aware local patterns. The highest count rows are*

Stop/Trp pattern on (UAA, UAG, UGA, UGG)	*table count*
$*/*/W/W$	8
$*/*/*/W$	5
$Y/*/W/W$	2
$*/L/*/W$	2.

The remaining ten Stop/Trp patterns each occur once:

$$Q/Q/*/W, \quad */Q/*/W, \quad */*/C/W, \quad */*/G/W, \quad Y/Y/*/W,$$
$$E/E/*/W, \quad */W/*/W, \quad Q/Q/*|W/W, \quad Q|*/Q|*/*|W/W, \quad E|*/E|*/W/W.$$

Proof. For each table, read the four tile coordinates in the displayed order. The comparison keeps partial rows such as $*|W$, $Q|*$, and $E|*$ as separate labels. Finite grouping gives the two Ile/Met rows and the fourteen Stop/Trp rows. □

Corollary 9462.1225 (Stop/Trp has richer comparative tile dynamics)**.** *The Ile/Met tile supports one observed nonstandard local operation, the* 10 *leaf joining the* 11 *Met corner. The Stop/Trp tile supports that same* 10*-to-*11 *form and also arm, one-sided, and partial-boundary forms on the* X_2 *side.*

Proof. The Ile/Met patterns in Definition 9462.1224 differ only by the third coordinate. The Stop/Trp patterns include changes in the first two coordinates and partial labels as well as the third coordinate. □

Theorem 9462.1226 (Alternative-code edge smoothness remains local)**.** *Across the standard table together with the curated non-standard genetic-code tables, the total number of same-class Hamming edges lies between* 47 *and* 52*, with mean approximately* 50.07*. The third-base family bit remains the dominant stable direction: its same-class edge count ranges from* 28 *to* 32 *out of* 32*, with mean approximately* 30.04*. The first-base shape bit has zero same-class edges in every table.*

Proof. For each table, enumerate the same 192 Hamming edges used in Definition 9462.1169 and count same-class endpoints, both in total and by bit direction. The displayed range and means are the finite counts over the comparison family. □

Corollary 9462.1227 (Alternative codes are local deformations)**.** *The comparative tables preserve the broad Hamming-local smoothness profile of the standard code while moving a small number of codon assignments, concentrated in the Stop/Trp tile, the Ile/Met tile, the Phe/Leu cube, and the AGA/AGG pair.*

Proof. The edge-count range is narrow by Definition 9462.1226, and the reassignment rows are concentrated by Definition 9462.1210. □

Theorem 9462.1228 (Nucleotide-substitution smoothness across the comparison)**.** *For ordinary single-nucleotide substitution edges, the mean same-class edge count across the comparison family is approximately* 69.04 *out of* 288*. By codon position, the mean same-class counts are approximately*

$$3.59, \qquad 0.67, \qquad 64.78$$

out of 96 *edges per position.*

Proof. For each table and each codon position, fix the other two bases and connect the four choices at the chosen position by the complete graph on four bases. Counting same-class endpoints over the resulting 288 substitution edges gives the displayed means. □

Corollary 9462.1229 (Third-position stability is table-family level)**.** *The third-position stability seen in the standard table is preserved across the comparison family, both for two-axis bit flips and for ordinary single-nucleotide substitution edges.*

Proof. The bit-flip statement is Definition 9462.1226. The nucleotide-substitution statement is Definition 9462.1228. □

Theorem 9462.1230 (Comparative tile-dynamics cannot-claim boundary)**.** *The comparative reassignment spectrum does not license any claim that alternative genetic codes are forced by BEDC, that the observed curated tables exhaust all possible biological translation systems, that codon identity alone determines ribosome dynamics, that reassignment frequency is a mechanistic cause, or that the AGA/AGG pair has already been reduced to a complete local control-tile certificate.*

Proof. The spectrum is a finite bridge-side comparison of curated external tables. It records where assignment rows differ from the standard classifier. It does not supply biochemical mechanism, evolutionary necessity, exhaustive biological coverage, or a checked BEDC certificate for the AGA/AGG pair. □

Corollary 9462.1231 (Allowed comparative tile-dynamics claim)**.** *Under the two-axis codon bridge, comparative genetic-code variation is strongly concentrated in local modules already visible in the standard codon-cube spectra. The Stop/Trp tile is the main deformation face; the shared* $b_1^f = 0$ *half-cube is the dominant reassignment region; the Stop/Trp* 11 *corner remains a stable continuation corner; and the observed Ile/Met deformation is coupled to the corresponding Stop/Trp* 10-*to*-11 *deformation.*

Proof. Combine Definition 9462.1210, Definition 9462.1214, Definition 9462.1220, and Definition 9462.1222. □

Concretized in §6448: Alternative-code reassignment spectra are realized as finite support, reassignment, spectral, transport, replay, provenance, and NameCert rows.

Definition 9462.1232 (Reassignment weight and subcube mass)**.** *For a codon* c*, define its comparative reassignment weight by*

$$w(c) = |\{t : E_t(c) \neq E_1(c)\}|,$$

where E_1 *is the standard classifier and* E_t *ranges over the curated comparison tables. For a subcube* $S \subset Q_6$*, define its reassignment mass by*

$$M(S) = \sum_{c \in S} w(c).$$

Theorem 9462.1233 (Nested maximum-mass subcubes)**.** *Among all axis-aligned subcubes of the indicated dimension, the unique maximum-mass subcubes form the following nested chain:*

dimension	*fixed conditions*	*codon reading*	*mass*	*fraction*
Q_5	$b_3^s = 1$	$c_3 \in \{\mathrm{A}, \mathrm{G}\}$	63	63/65
Q_4	$b_1^f = 0,\ b_3^s = 1$	$c_1 \in \{\mathrm{U}, \mathrm{A}\},\ c_3 \in \{\mathrm{A}, \mathrm{G}\}$	59	59/65
Q_3	$b_1^f = 0,\ b_2^s = 1,\ b_3^s = 1$	$c_1 \in \{\mathrm{U}, \mathrm{A}\},\ c_2, c_3 \in \{\mathrm{A}, \mathrm{G}\}$	52	52/65
Q_2	$\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{UAA}, \mathrm{UGA}\}$	*A-ending purine-core square*	34	34/65.

Proof. Enumerate all axis-aligned Q_d subcubes for $d = 5, 4, 3, 2$ and compute $M(S)$ from Definition 9462.1232. At each dimension, the maximum is attained by exactly one subcube, and the four maximizers are the displayed nested subcubes. □

Corollary 9462.1234 (Purine-ending constraint)**.** *The strongest single-bit constraint on reassignment mass is $b_3^s = 1$: 63 of the 65 reassignment rows have third codon base in $\{\mathrm{A}, \mathrm{G}\}$.*

Proof. This is the Q_5 row of Definition 9462.1233. □

Definition 9462.1235 (Purine-core cube)**.** *The purine-core cube is*

$$Q_3^{\mathrm{pur}} = \{\mathrm{U}, \mathrm{A}\} \times \{\mathrm{A}, \mathrm{G}\} \times \{\mathrm{A}, \mathrm{G}\}.$$

Equivalently,

$$Q_3^{\mathrm{pur}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}.$$

Theorem 9462.1236 (Purine-core cube is the maximum-mass Q_3)**.** *The purine-core cube is the unique maximum-mass Q_3 subcube. Its standard labels and reassignment weights are*

codon	*standard label*	$w(c)$
UAA	Stop	8
UAG	Stop	10
UGA	Stop	15
UGG	Trp	0
AAA	Lys	3
AAG	Lys	0
AGA	Arg	8
AGG	Arg	8.

Thus $M(Q_3^{\mathrm{pur}}) = 52$.

Proof. The Q_3 maximum in Definition 9462.1233 is fixed by $b_1^f = 0$, $b_2^s = 1$, and $b_3^s = 1$, which is exactly Definition 9462.1235. Summing the displayed weights gives 52. □

Corollary 9462.1237 (Stop/Trp sits inside the Stop/Lys/Arg/Trp core)**.** *The Stop/Trp tile is not the whole comparative hotspot. It is embedded inside the larger standard-code pattern*

$$\mathrm{Stop}_3 + \mathrm{Lys}_2 + \mathrm{Arg}_2 + \mathrm{Trp}_1$$

on Q_3^{pur}.

Proof. Read the standard labels in Definition 9462.1236. The first four codons are the Stop/Trp tile; the other four codons are the Lys and Arg doubletons. □

Definition 9462.1238 (A-ending purine-core square)**.** *The A-ending purine-core square is*

$$Q_2^{\mathrm{Aend}} = \{\mathrm{AAA}, \mathrm{AGA}, \mathrm{UAA}, \mathrm{UGA}\}.$$

Theorem 9462.1239 (A-ending square is the maximum-mass Q_2)**.** *The square Q_2^{Aend} is the unique maximum-mass Q_2 face. Its weights are*

codon	*standard label*	$w(c)$
AAA	Lys	3
AGA	Arg	8
UAA	Stop	8
UGA	Stop	15.

Thus $M(Q_2^{\mathrm{Aend}}) = 34$.

Proof. Enumerating the 240 axis-aligned Q_2 faces and summing reassignment weights gives a unique maximum of 34, attained at $\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{UAA}, \mathrm{UGA}\}$. □

Corollary 9462.1240 (Static and comparative faces differ)**.** *The static* $3+1$ *control tile is*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\},$$

with reassignment mass 33*. The maximum-mass comparative* Q_2 *face is*

$$\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{UAA}, \mathrm{UGA}\},$$

with reassignment mass 34*. Thus the static coding-control face and the comparative maximum-mass face are adjacent but not identical.*

Proof. The Stop/Trp mass is $8 + 10 + 15 + 0 = 33$ by Definition 9462.1236. The comparative maximum Q_2 mass is Definition 9462.1239. □

Theorem 9462.1241 (Complete reassignment Q_2 faces)**.** *Among the* 240 *axis-aligned* Q_2 *faces, the distribution by number of reassigned vertices is*

reassigned vertices in the face	0	1	2	3	4
face count	123	60	40	13	4.

The four faces whose vertices are all reassigned codons are

face	*mass*	*reading*
$\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{UAA}, \mathrm{UGA}\}$	34	*A-ending purine-core square*
$\{\mathrm{UAA}, \mathrm{UCA}, \mathrm{UGA}, \mathrm{UUA}\}$	25	*U-first A-ending square*
$\{\mathrm{AAA}, \mathrm{AUA}, \mathrm{UAA}, \mathrm{UUA}\}$	17	*A-ending first-position square*
$\{\mathrm{CUA}, \mathrm{CUC}, \mathrm{CUG}, \mathrm{CUU}\}$	6	*CUN Leu square.*

Proof. For each Q_2 face, count how many of its four vertices have positive reassignment weight. The resulting finite distribution and the four complete faces are displayed. □

Corollary 9462.1242 (Reassignment support contains a small surface)**.** *The reassigned codon support contains four complete* Q_2 *faces. Thus the support is not only Hamming-connected; it contains a small local surface made from square faces.*

Proof. Connectivity was established in Definition 9462.1218. The complete square faces are Definition 9462.1241. □

Definition 9462.1243 (Co-reassignment edge weight)**.** *For adjacent codons* $c \sim d$ *in the reassignment support, define*

$$\omega(c, d) = |\{t : E_t(c) \neq E_1(c) \text{ and } E_t(d) \neq E_1(d)\}|.$$

Theorem 9462.1244 (Positive co-reassignment edges)**.** *The positive co-reassignment edges are*

edge	ω
UGA−AGA	8
AGA−AGG	8
UAA−UAG	6
UAA−UGA	5
AGA−AAA	3
AUA−AAA	1
UAA−AAA	1
CUA−CUG	1
CUC−CUG	1
CUU−CUA	1
CUU−CUC	1.

Proof. Restrict the Q_6 Hamming graph to reassigned codons. For each internal edge, intersect the two sets of comparison tables in which its endpoints are reassigned. The positive intersections are exactly the displayed finite table. □

Corollary 9462.1245 (Weighted deformation spine)**.** *The highest-weight co-reassignment path is*

$$\mathrm{UAG} - \mathrm{UAA} - \mathrm{UGA} - \mathrm{AGA} - \mathrm{AGG},$$

with branch

$$\mathrm{AGA} - \mathrm{AAA} - \mathrm{AUA}.$$

In particular, UGA *is not only a Stop/Trp tile leaf; it is also the Hamming bridge from the stop side to the AGR side.*

Proof. Read the positive edge weights in Definition 9462.1244. The edges UGA−AGA and AGA−AGG have weight 8, and the stop-arm edges have weights 6 and 5. □

Theorem 9462.1246 (Positive co-reassignment components)**.** *The graph on reassigned codons using only positive co-reassignment edges has components*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{AAA}, \mathrm{AUA}, \mathrm{AGA}, \mathrm{AGG}\},$$
$$\{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\},$$
$$\{\mathrm{UCA}\}, \qquad \{\mathrm{UUA}\}.$$

Proof. Use the positive edge list of Definition 9462.1244 and compute connected components. The isolated vertices are reassigned codons whose Hamming-neighbor co-reassignment intersections are zero. □

Corollary 9462.1247 (Geometric support and co-reassignment support differ)**.** *The full reassignment support is connected as a Hamming subgraph, but positive co-reassignment separates a Stop/AGR/AUA/AAA component from a CUN component and two isolated low-frequency codons.*

Proof. Hamming connectivity is Definition 9462.1218. Positive co-reassignment connectivity is Definition 9462.1246. □

Theorem 9462.1248 (Nested-core comparative dynamics statement)**.** *The comparative reassignment spectrum is concentrated along a nested subcube core*

$$Q_5 \supset Q_4 \supset Q_3 \supset Q_2,$$

where the Q_3 core is

$$Q_3^{\mathrm{pur}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}.$$

The same data define a positive co-reassignment spine

$$\mathrm{UAG} - \mathrm{UAA} - \mathrm{UGA} - \mathrm{AGA} - \mathrm{AGG}$$

with branch $\mathrm{AGA} - \mathrm{AAA} - \mathrm{AUA}$.

Proof. The nested subcube core is Definition 9462.1233. The Q_3 core is identified in Definition 9462.1236. The weighted spine is Definition 9462.1245. □

Concretized in §5869: finite maximum-mass subcube NameCert for reassignment geometry

Definition 9462.1249 (Connected deformation candidate)**.** *Let $R = \{c : w(c) > 0\}$ be the reassigned codon support. A subset $S \subseteq R$ is a connected deformation candidate when the Hamming subgraph $Q_6[S]$ is connected. Its mass is*

$$M(S) = \sum_{c \in S} w(c).$$

For each cardinality k, define a maximum-mass connected candidate by maximizing $M(S)$ over connected $S \subseteq R$ with $|S| = k$.

Theorem 9462.1250 (Maximum-mass connected candidates)**.** *For $k = 1, \ldots, 7$, the maximum-mass connected candidates are*

k	*maximum mass*	*maximizing connected support*
1	15	$\{\mathrm{UGA}\}$
2	23	$\{\mathrm{UGA}, \mathrm{AGA}\}$ *or* $\{\mathrm{UGA}, \mathrm{UAA}\}$
3	33	$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$
4	41	$\{\mathrm{AGA}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$
5	49	$\{\mathrm{AGA}, \mathrm{AGG}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$
6	52	$\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$
7	57	$\{\mathrm{AAA}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{AUA}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$.

From $k = 3$ through $k = 7$, the maximizer is unique and the supports form a nested chain.

Proof. Enumerate all subsets of R of cardinality k, discard those whose induced Hamming subgraph is disconnected, and compute the mass from Definition 9462.1232. The finite enumeration gives the displayed table. The $k = 2$ row has two maximizers; the rows $k = 3, \ldots, 7$ have one maximizer each, and direct inclusion checks give the nested chain. □

Corollary 9462.1251 (Stop triplet is the first stable connected core)**.** *The first unique connected optimum beyond the singleton case is*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\},$$

the standard Stop tripleton.

Proof. This is the $k = 3$ row of Definition 9462.1250. □

Corollary 9462.1252 (UGA is the stop-to-AGR bridge)**.** *The connected optimum grows from the Stop tripleton by adding* AGA *through the Hamming edge*

$$\mathrm{UGA} - \mathrm{AGA}.$$

Thus UGA *is both the* 10 *leaf of the Stop/Trp tile and the bridge from the stop side into the AGR side of the comparative support.*

Proof. The $k = 4$ optimum in Definition 9462.1250 adds AGA to the Stop tripleton. The edge UGA − AGA is a Hamming edge, and it is one of the highest-weight co-reassignment edges in Definition 9462.1244. □

Definition 9462.1253 (Bit majority mass)**.** *For a bit coordinate i, define*

$$M_i(\epsilon) = \sum_{c:b_i(c)=\epsilon} w(c), \qquad \epsilon \in \{0, 1\}.$$

The majority mass at bit i is

$$M_i^{\max} = \max(M_i(0), M_i(1)).$$

Theorem 9462.1254 (Bit-level reassignment concentration)**.** *The six bit coordinates have the following majority masses:*

bit	*majority condition*	*majority mass*	*fraction*
b_1^s	0	41	41/65
b_1^f	0	59	59/65
b_2^s	1	52	52/65
b_2^f	0	33	33/65
b_3^s	1	63	63/65
b_3^f	0	43	43/65.

The strongest single-bit condition is $b_3^s = 1$, the third-position purine condition.

Proof. For each reassigned codon, add its weight to the 0 or 1 bucket of each bit coordinate. Taking the larger bucket at each bit gives the displayed table. □

Corollary 9462.1255 (Third-position dual role)**.** *The third codon position has two distinct roles in the two-axis spectrum: b_3^f is the dominant same-class Hamming direction in Definition 9462.1170, while $b_3^s = 1$ is the dominant single-bit concentration condition for reassignment mass.*

Proof. The same-class edge statement is Definition 9462.1170. The reassignment-mass statement is Definition 9462.1254. □

Theorem 9462.1256 (High-mass core and broad-support core)**.** *Among Q_3 subcubes, the maximum-mass core is the purine-core cube of Definition 9462.1235, with mass* 52 *and six reassigned vertices. The unique Q_3 subcube containing the largest number of reassigned vertices is*

$$\{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{ACA}, \mathrm{AAA}, \mathrm{AGA}\}.$$

It is fixed by

$$b_1^f = 0, \qquad b_3^s = 1, \qquad b_3^f = 0,$$

contains seven reassigned vertices, and has mass 41.

Proof. Enumerate the 160 axis-aligned Q_3 subcubes. Sorting by reassignment mass gives the purine-core cube as in Definition 9462.1236. Sorting by the number of vertices with positive reassignment weight gives the displayed A-ending cube, uniquely. □

Corollary 9462.1257 (Two cores in the reassignment zone)**.** *The reassignment zone has a high-mass core*

$$Q_3^{\mathrm{pur}}, \qquad M = 52,$$

and a broad-support A-ending core with seven reassigned vertices and mass 41*.*

Proof. This is a restatement of Definition 9462.1256. □

Theorem 9462.1258 (Three-block low-dimensional cover)**.** *All* 13 *reassigned codons can be covered by three low-dimensional subcubes:*

block	*covered reassigned codons*	*mass*
Q_3^{pur}	{UAA, UAG, UGA, AAA, AGA, AGG}	52
{CUU, CUC, CUA, CUG}	{CUU, CUC, CUA, CUG}	6
{UUA, UCA, AUA, ACA}	{UUA, UCA, AUA}	7.

The minimum number of Q_2 and Q_3 blocks needed to cover all reassigned codons is 3*.*

Proof. Enumerate all Q_2 and Q_3 subcubes and solve the finite set cover problem over the reassigned support. No one-block or two-block cover exists, and the displayed three blocks cover all 13 reassigned codons. □

Corollary 9462.1259 (Three-module compression)**.** *The comparative support admits a three-module compression:*

$$\textit{purine-core } Q_3 \quad + \quad \textit{CUN square} \quad + \quad \textit{A-ending residual square.}$$

Proof. This is the semantic reading of the three blocks in Definition 9462.1258. □

Theorem 9462.1260 (Weighted deformation spine)**.** *The comparative reassignment spectrum is governed by the nested connected spine*

$$\begin{gathered}\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} \subset \{\mathrm{AGA}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} \\ \subset \{\mathrm{AGA}, \mathrm{AGG}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} \subset \{\mathrm{AAA}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} \\ \subset \{\mathrm{AAA}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{AUA}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.\end{gathered}$$

As a Hamming graph, its main path is

$$\mathrm{UAG} - \mathrm{UAA} - \mathrm{UGA} - \mathrm{AGA} - \mathrm{AGG},$$

with branch

$$\mathrm{AGA} - \mathrm{AAA} - \mathrm{AUA}.$$

Proof. The nested connected supports are the unique maximum-mass connected candidates for $k = 3, \ldots, 7$ in Definition 9462.1250. Drawing the Hamming edges among those vertices gives the displayed path and branch. □

Corollary 9462.1261 (Stop, AGR, AAA, and AUA are one comparative spine)**.** *The Stop, AGR, AAA, and AUA reassignment modules are not separated in the weighted Hamming geometry. They lie on a single connected deformation spine.*

Proof. The spine in Definition 9462.1260 contains the Stop triplet, the AGR doublet, AAA, and AUA. □

Theorem 9462.1262 (Weighted-spine comparative dynamics statement)**.** *The comparative genetic-code spectrum has three simultaneous finite geometric descriptions:*

$$\textit{a maximum-mass subcube chain } Q_5 \supset Q_4 \supset Q_3 \supset Q_2,$$
$$\textit{a high-mass purine-core } Q_3 = \mathrm{Stop}_3 + \mathrm{Lys}_2 + \mathrm{Arg}_2 + \mathrm{Trp}_1,$$

and a weighted deformation spine UAG − UAA − UGA − AGA − AGG *with branch* AGA − AAA − AUA.

Proof. The subcube chain is Definition 9462.1233. The purine-core identification is Definition 9462.1236. The spine is Definition 9462.1260. □

Concretized in §5552: The weighted deformation spine is realized as finite trigger-support, weight-ledger, deformation-route, boundary-readback, transport, replay, provenance, and NameCert rows.

Definition 9462.1263 (Walsh spectrum of reassignment weight)**.** *Let $w : Q_6 \to \mathbb{R}_{\geq 0}$ be the reassignment weight function. For $S \subseteq \{1, \ldots, 6\}$, define*

$$\chi_S(x) = (-1)^{\sum_{i \in S} x_i}, \qquad \widehat{w}(S) = 2^{-6} \sum_{x \in Q_6} w(x)\chi_S(x).$$

Theorem 9462.1264 (Dominant Walsh coefficients)**.** *The largest non-constant Walsh coefficients of w are*

rank	S	$\widehat{w}(S)$	*reading*
1	b_3^s	−0.953125	*third base biased to purine*
2	$b_1^f b_3^s$	−0.890625	*first family with third purine*
3	b_1^f	0.828125	*first base biased to* {U, A}
4	$b_1^f b_2^s$	−0.796875	*first family with second purine*
5	$b_1^f b_2^s b_3^s$	0.734375	*purine-core interaction*
6	$b_2^s b_3^s$	0.671875	*second and third purine interaction*
7	b_2^s	−0.609375	*second base biased to purine.*

Proof. Evaluate the finite Walsh transform of Definition 9462.1263 over all 64 codons and sort the non-empty subsets by absolute coefficient. □

Corollary 9462.1265 (Walsh spectrum recovers the purine-core variables)**.** *The dominant spectral variables are exactly the coordinates defining the purine-core cube:*

$$b_1^f = 0, \qquad b_2^s = 1, \qquad b_3^s = 1.$$

Proof. The variables b_3^s, b_1^f, and b_2^s and their low-order interactions occupy the leading rows of Definition 9462.1264. These are exactly the fixed coordinates of Q_3^{pur} in Definition 9462.1235. □

Theorem 9462.1266 (Low-order spectral compressibility)**.** *After removing the constant coefficient, the Walsh energy $\sum_{S \neq \varnothing} \widehat{w}(S)^2$ has degree distribution*

degree	1	2	3	4	5	6
energy share	0.275	0.320	0.194	0.137	0.065	0.009.

Consequently, degrees at most 3 *account for approximately* 0.789 *of the non-constant energy, and degrees at most* 4 *account for approximately* 0.926.

Proof. Compute $\widehat{w}(S)^2$ for each non-empty subset $S \subseteq \{1, \ldots, 6\}$, group by $|S|$, and normalize by the total non-constant energy. □

Corollary 9462.1267 (Reassignment weight is low-order in Q_6)**.** *The reassignment weight function is largely controlled by low-order bit conditions rather than by high-degree oscillation on the full codon cube.*

Proof. This is the cumulative degree statement in Definition 9462.1266. □

Definition 9462.1268 (Weight-permutation null model)**.** *The permutation null model keeps the multiset of reassignment weights fixed and randomly assigns those weights to the* 64 *vertices of* Q_6*. For each randomized weight function and each* d*, record*

$$\max_{Q_d \subset Q_6} M(Q_d).$$

Theorem 9462.1269 (Subcube mass significance under permutation)**.** *Using* 100000 *fixed-seed random permutations of the reassignment weight multiset, the maximum subcube masses have the following null comparison:*

dimension	*observed maximum mass*	*random mean maximum mass*	p
Q_2	34	24.97	0.01415
Q_3	52	31.46	0.00020
Q_4	59	39.82	0.00053
Q_5	63	50.94	0.01003.

Here p *is the fraction of randomized trials whose maximum subcube mass is at least the observed maximum.*

Proof. For each trial in the permutation null model of Definition 9462.1268, enumerate all axis-aligned Q_d subcubes for $d = 2, 3, 4, 5$ and record the maximum mass. The displayed means and tail fractions are the resulting finite Monte Carlo counts with fixed seed. □

Corollary 9462.1270 (The strongest subcube signal is three-dimensional)**.** *Although the* Q_5 *condition captures* 63/65 *of the mass, the strongest permutation-null signal is the purine-core* Q_3*, which captures* 52/65 *of the mass on only eight codons.*

Proof. Compare the p column in Definition 9462.1269 with the subcube identities in Definition 9462.1233. □

Theorem 9462.1271 (Three-module geometric compression)**.** *The reassigned support cannot be covered by two* Q_2/Q_3 *subcubes, but it can be covered by three. One such cover is*

module	*covered reassigned codons*	*mass*
Q_3^{pur}	{UAA, UAG, UGA, AAA, AGA, AGG}	52
CUN square	{CUU, CUC, CUA, CUG}	6
A-ending residual square	{UUA, UCA, AUA}	7.

Proof. This is the finite Q_2/Q_3 set-cover enumeration of Definition 9462.1258. The first nonempty covers occur at size 3; the displayed cover is the canonical three-module reading used there. □

Corollary 9462.1272 (Purine core dominates the minimal compression)**.** *In the three-module compression,* Q_3^{pur} *carries* 52/65 *of the total reassignment mass. The two remaining* Q_2 *modules cover the low-mass residual support.*

Proof. Read the mass column of Definition 9462.1271. □

Theorem 9462.1273 (Spectral compression statement)**.** *The reassignment weight function is recovered by four independent finite signals:*

$$\begin{gathered}\textit{dominant Walsh variables } b_3^s, b_1^f, b_2^s,\\ \textit{low-order spectral energy concentration},\\ \textit{permutation-significant maximum } Q_3^{\mathrm{pur}},\\ \textit{and a three-module } Q_2/Q_3 \textit{ cover.}\end{gathered}$$

Thus the purine-core Q_3 is not an externally chosen block; it is selected by the spectrum, by subcube mass, by the permutation null model, and by geometric compression.

Proof. The Walsh variables are Definition 9462.1264. The energy concentration is Definition 9462.1266. The permutation-null statement is Definition 9462.1269. The geometric cover is Definition 9462.1271. □

Concretized in §3897: The Walsh coefficient and median-closure support rows are realized as a finite NameCert spectrum packet.

Definition 9462.1274 (Reassigned support)**.** *Let*

$$\mathcal{R} = \{c : \exists t,\ E_t(c) \neq E_1(c)\}$$

be the support of codons that are reassigned in at least one comparison table.

Theorem 9462.1275 (Reassigned support)**.** *The reassigned support is the* 13*-codon set*

$$\begin{gathered}\mathcal{R} = \{\mathrm{UGA}, \mathrm{UAG}, \mathrm{UAA}, \mathrm{AUA}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{AAA},\\ \mathrm{CUG}, \mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{UCA}, \mathrm{UUA}\}.\end{gathered}$$

Proof. This is the support computed in Definition 9462.1212, ignoring table multiplicity. □

Theorem 9462.1276 (Five-motif support formula)**.** *With IUPAC notation*

$$N = \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}, \qquad R = \{\mathrm{A}, \mathrm{G}\}, \qquad S = \{\mathrm{C}, \mathrm{G}\},$$

and

$$W = \{\mathrm{U}, \mathrm{A}\}, \qquad Y = \{\mathrm{U}, \mathrm{C}\},$$

the reassigned support satisfies the exact formula

$$\mathcal{R} = \mathrm{CUN} \cup \mathrm{AGR} \cup \mathrm{UNA} \cup \mathrm{WWA} \cup \mathrm{UAR}.$$

Proof. Expand the five motifs:

$$\begin{aligned}\mathrm{CUN} &= \{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\},\\ \mathrm{AGR} &= \{\mathrm{AGA}, \mathrm{AGG}\},\\ \mathrm{UNA} &= \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}\},\\ \mathrm{WWA} &= \{\mathrm{UUA}, \mathrm{UAA}, \mathrm{AUA}, \mathrm{AAA}\},\\ \mathrm{UAR} &= \{\mathrm{UAA}, \mathrm{UAG}\}.\end{aligned}$$

Their union is exactly the 13 codons displayed in Definition 9462.1275. □

Corollary 9462.1277 (Support is a union of motifs)**.** *The reassigned support is not merely a list of codons. It is the union of five low-dimensional codon motifs:*

$$\text{CUN}, \quad \text{AGR}, \quad \text{UNA}, \quad \text{WWA}, \quad \text{UAR}.$$

Proof. This is Definition 9462.1276. □

Definition 9462.1278 (Reassignment support function)**.** *The support function of the comparative reassignment set is the Boolean map*

$$F : Q_6 \to \{0,1\}, \qquad F(x) = 1 \iff x \in \mathcal{R}.$$

An exact subcube is an axis-aligned subcube $S \subset Q_6$ satisfying $S \subseteq \mathcal{R}$. A prime implicant is a maximal exact subcube under inclusion.

Theorem 9462.1279 (Boolean DNF form)**.** *Write a codon as*

$$(s_1, f_1, s_2, f_2, s_3, f_3)$$

using $U = 00$, $C = 01$, $A = 10$, and $G = 11$. Then F is the disjunction of the five terms

$$\begin{aligned}
\text{CUN} &: \neg s_1 \wedge f_1 \wedge \neg s_2 \wedge \neg f_2, \\
\text{AGR} &: s_1 \wedge \neg f_1 \wedge s_2 \wedge f_2 \wedge s_3, \\
\text{UNA} &: \neg s_1 \wedge \neg f_1 \wedge s_3 \wedge \neg f_3, \\
\text{WWA} &: \neg f_1 \wedge \neg f_2 \wedge s_3 \wedge \neg f_3, \\
\text{UAR} &: \neg s_1 \wedge \neg f_1 \wedge s_2 \wedge \neg f_2 \wedge s_3.
\end{aligned}$$

Proof. Each displayed Boolean term is exactly the bit condition defining the corresponding IUPAC motif in Definition 9462.1276. Their disjunction therefore defines the same support function F. □

Theorem 9462.1280 (Prime implicants of the reassignment support)**.** *The Boolean support function F has exactly seven prime implicants:*

motif	*codons*	*weight*
CUN	{CUU, CUC, CUA, CUG}	6
UNA	{UUA, UCA, UAA, UGA}	25
WRA	{AAA, AGA, UAA, UGA}	34
WWA	{AAA, AUA, UAA, UUA}	17
AGR	{AGA, AGG}	16
UAR	{UAA, UAG}	18
YUA	{CUA, UUA}	2.

Proof. Enumerate the axis-aligned subcubes of Q_6. Retain those whose vertices lie in $\mathcal{R}$, then delete every retained subcube strictly contained in another retained subcube. The remaining maximal exact subcubes are precisely the seven motifs displayed above. The weights are the table-count reassignment weights summed over the codons in each motif. □

Theorem 9462.1281 (Minimal prime-implicant cover)**.** *Among covers of $\mathcal{R}$ by prime implicants, the minimum cover size is 5, and the cover is unique:*

$$\text{CUN}, \quad \text{WWA}, \quad \text{UNA}, \quad \text{AGR}, \quad \text{UAR}.$$

Proof. Run finite set-cover enumeration over the seven prime implicants of Definition 9462.1280. No subfamily of size less than 5 covers $\mathcal{R}$, and exactly one subfamily of size 5 does. It is the displayed family. □

Theorem 9462.1282 (Exact subcube covers)**.** *If nonmaximal exact subcubes are also allowed, the minimum cover size is still* 5*, but the minimum cover is not unique. There are exactly six minimum covers:*

$$\{\mathrm{UAG}, \mathrm{USA}, \mathrm{AGR}, \mathrm{WWA}, \mathrm{CUN}\},$$
$$\{\mathrm{UAG}, \mathrm{AWA}, \mathrm{AGR}, \mathrm{UNA}, \mathrm{CUN}\},$$
$$\{\mathrm{UAG}, \mathrm{AGR}, \mathrm{WWA}, \mathrm{UNA}, \mathrm{CUN}\},$$
$$\{\mathrm{USA}, \mathrm{UAR}, \mathrm{AGR}, \mathrm{WWA}, \mathrm{CUN}\},$$
$$\{\mathrm{AWA}, \mathrm{UAR}, \mathrm{AGR}, \mathrm{UNA}, \mathrm{CUN}\},$$
$$\{\mathrm{UAR}, \mathrm{AGR}, \mathrm{WWA}, \mathrm{UNA}, \mathrm{CUN}\}.$$

Proof. Enumerate all exact subcubes, including singleton subcubes. There are 33 retained exact subcubes. A finite set-cover search gives no cover of size less than 5 and the six covers displayed above at size 5. □

Corollary 9462.1283 (Forced support motifs)**.** *Across all minimum exact subcube covers, the only common motifs are*

$$\mathrm{CUN} \qquad \textit{and} \qquad \mathrm{AGR}.$$

Proof. Intersect the six minimum covers in Definition 9462.1282. The common terms are exactly CUN and AGR. □

Concretized in §3558: Reassignment support compression is realized as finite support, motif, prime-implicant, minimal-cover, transport, replay, provenance, and NameCert rows.

Theorem 9462.1284 (Envelope versus exact support)**.** *The three-module low-dimensional envelope is*

$$\mathrm{WRR} \cup \mathrm{CUN} \cup \mathrm{WYA},$$

where

$$\mathrm{WRR} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\},$$
$$\mathrm{CUN} = \{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\},$$
$$\mathrm{WYA} = \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{AUA}, \mathrm{ACA}\}.$$

It satisfies

$$\mathrm{WRR} \cup \mathrm{CUN} \cup \mathrm{WYA} = \mathcal{R} \cup \{\mathrm{UGG}, \mathrm{AAG}, \mathrm{ACA}\}.$$

Proof. Expand the three motifs and compare with Definition 9462.1275. The only codons in the three-module envelope that are not in $\mathcal{R}$ are UGG, AAG, and ACA. □

Corollary 9462.1285 (Three-module envelope and five-motif support)**.** *There are two complementary compressions of the reassigned support:*

$$\mathcal{R} \subset \mathrm{WRR} \cup \mathrm{CUN} \cup \mathrm{WYA}$$

with three false-positive codons, and

$$\mathcal{R} = \mathrm{CUN} \cup \mathrm{AGR} \cup \mathrm{UNA} \cup \mathrm{WWA} \cup \mathrm{UAR}$$

with no false positives.

Proof. The envelope equality is Definition 9462.1284. The exact equality is Definition 9462.1276. □

Theorem 9462.1286 (Prime-implicant overlap hub)**.** *In the weighted overlap graph of the seven prime implicants,* WRA *has the largest total overlap weight. Its nonempty overlaps include*

$$\begin{aligned} \mathrm{WRA} \cap \mathrm{UNA} &= \{\mathrm{UAA}, \mathrm{UGA}\}, & \textit{weight} &= 23, \\ \mathrm{WRA} \cap \mathrm{WWA} &= \{\mathrm{AAA}, \mathrm{UAA}\}, & \textit{weight} &= 11, \\ \mathrm{WRA} \cap \mathrm{UAR} &= \{\mathrm{UAA}\}, & \textit{weight} &= 8, \\ \mathrm{WRA} \cap \mathrm{AGR} &= \{\mathrm{AGA}\}, & \textit{weight} &= 8. \end{aligned}$$

The total weighted overlap score of WRA *is* 50*, larger than the corresponding scores for* UNA*,* WWA*,* UAR*,* AGR*,* YUA*, and* CUN*.*

Proof. Take all pairwise intersections among the seven prime implicants in Definition 9462.1280. Weight each intersection by the reassignment weights of the shared codons and sum the incident intersection weights at each prime implicant. The incident score of WRA is $23 + 11 + 8 + 8 = 50$, and the other incident scores are smaller. □

Theorem 9462.1287 (Support compression statement)**.** *The comparative reassignment support admits separate support and mass geometries:*

$$\textit{support geometry: } \mathrm{CUN} \cup \mathrm{AGR} \cup \mathrm{UNA} \cup \mathrm{WWA} \cup \mathrm{UAR},$$
$$\textit{mass geometry: } \mathrm{WRA} \subset \mathrm{WRR}.$$

The support geometry is the unique minimum cover by prime implicants. The mass geometry identifies the highest-weight exact square WRA *inside the purine-core envelope* WRR*.*

Proof. The support-geometry statement is Definition 9462.1281. The mass statement follows from Definition 9462.1280 and Definition 9462.1286, together with the envelope identity in Definition 9462.1284. □

Definition 9462.1288 (Reassignment support graph)**.** *Let $G_{\mathcal{R}} = Q_6[\mathcal{R}]$ be the Hamming graph induced by the reassigned support $\mathcal{R}$. Thus two vertices of $G_{\mathcal{R}}$ are adjacent exactly when the corresponding codons differ in one bit of the two-axis readback.*

Theorem 9462.1289 (Reassignment support graph)**.** *The induced graph $G_{\mathcal{R}}$ is connected and has*

$$|V(G_{\mathcal{R}})| = 13, \qquad |E(G_{\mathcal{R}})| = 16.$$

Its cyclomatic number is therefore

$$|E| - |V| + 1 = 4.$$

Proof. Enumerating one-bit Hamming edges inside $\mathcal{R}$ gives

$$\begin{array}{llll} \mathrm{UUA}{-}\mathrm{AUA}, & \mathrm{UAA}{-}\mathrm{AAA}, & \mathrm{UGA}{-}\mathrm{AGA}, & \mathrm{UUA}{-}\mathrm{CUA}, \\ \mathrm{UUA}{-}\mathrm{UAA}, & \mathrm{UCA}{-}\mathrm{UGA}, & \mathrm{AUA}{-}\mathrm{AAA}, & \mathrm{UUA}{-}\mathrm{UCA}, \\ \mathrm{UAA}{-}\mathrm{UGA}, & \mathrm{AAA}{-}\mathrm{AGA}, & \mathrm{CUU}{-}\mathrm{CUA}, & \mathrm{CUC}{-}\mathrm{CUG}, \\ \mathrm{UAA}{-}\mathrm{UAG}, & \mathrm{CUU}{-}\mathrm{CUC}, & \mathrm{CUA}{-}\mathrm{CUG}, & \mathrm{AGA}{-}\mathrm{AGG}. \end{array}$$

These edges connect all 13 vertices, so the displayed cyclomatic number follows. □

Corollary 9462.1290 (Compact support graph)**.** *The support graph is significantly denser than a random* 13*-vertex subset of* Q_6*. The random-subset expectation is*

$$192 \cdot \frac{13 \cdot 12}{64 \cdot 63} = 7.428571\ldots$$

internal Hamming edges, while $G_{\mathcal{R}}$ *has* 16 *such edges. A* 100000*-trial fixed-seed random-subset test gives exceedance count* 19*, hence* $p = 0.00019$ *for observing at least* 16 *internal edges.*

Proof. The expectation is the number of Hamming edges in Q_6 times the probability that both endpoints of a fixed edge lie in a uniformly chosen 13-element subset. The Monte Carlo count is obtained by the same edge statistic over random supports of size 13. □

Theorem 9462.1291 (Four square cycles)**.** *The four independent cycles of* $G_{\mathcal{R}}$ *are represented by the four complete* Q_2 *faces*

$$\begin{aligned}
\mathrm{CUN} &= \{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\},\\
\mathrm{UNA} &= \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}\},\\
\mathrm{WWA} &= \{\mathrm{UUA}, \mathrm{UAA}, \mathrm{AUA}, \mathrm{AAA}\},\\
\mathrm{WRA} &= \{\mathrm{UAA}, \mathrm{UGA}, \mathrm{AAA}, \mathrm{AGA}\}.
\end{aligned}$$

Their reassignment weights are respectively 6*,* 25*,* 17*, and* 34*.*

Proof. Each displayed motif is an axis-aligned Q_2 face contained in $\mathcal{R}$, hence contributes a four-cycle to $G_{\mathcal{R}}$. Since Definition 9462.1289 gives cyclomatic number 4, these four square cycles account for the full cycle rank. □

Definition 9462.1292 (Punctured A-ending W-family cube)**.** *Let*

$$\mathrm{WNA} = \{\mathrm{U}, \mathrm{A}\} \times \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\} \times \{\mathrm{A}\}.$$

Equivalently,

$$\mathrm{WNA} = \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{ACA}, \mathrm{AAA}, \mathrm{AGA}\}.$$

Theorem 9462.1293 (Punctured-cube skeleton formula)**.** *The reassigned support admits the exact cube-complex skeleton formula*

$$\mathcal{R} = (\mathrm{WNA} \setminus \{\mathrm{ACA}\}) \cup \mathrm{CUN} \cup \{\mathrm{UAG}, \mathrm{AGG}\}.$$

Proof. The punctured cube is

$$\mathrm{WNA} \setminus \{\mathrm{ACA}\} = \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{AAA}, \mathrm{AGA}\}.$$

Adding

$$\mathrm{CUN} = \{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\}$$

and the two codons $\mathrm{UAG}, \mathrm{AGG}$ gives exactly the support listed in Definition 9462.1275. □

Theorem 9462.1294 (Cube-complex edge decomposition)**.** *At the graph level,*

$$G_{\mathcal{R}} = G_{\mathrm{WNA}\setminus\{\mathrm{ACA}\}} \cup G_{\mathrm{CUN}} \cup \{\mathrm{UAA}{-}\mathrm{UAG}, \mathrm{AGA}{-}\mathrm{AGG}, \mathrm{UUA}{-}\mathrm{CUA}\}.$$

The punctured cube contributes 7 *vertices and* 9 *edges, the CUN square contributes* 4 *vertices and* 4 *edges, and the three displayed bridges account for the remaining edges.*

Proof. Intersect the edge list of Definition 9462.1289 with $\mathrm{WNA} \setminus \{\mathrm{ACA}\}$ and with CUN. The first induced subgraph has 9 listed edges, the second has 4, and the complement in the 16-edge support graph is precisely the three displayed bridge edges. □

Theorem 9462.1295 (Articulation gates)**.** *The articulation codons of $G_{\mathcal{R}}$ are exactly*

$$\mathrm{UAA}, \quad \mathrm{AGA}, \quad \mathrm{UUA}, \quad \mathrm{CUA}.$$

The leaf codons are exactly

$$\mathrm{UAG}, \quad \mathrm{AGG}.$$

Proof. Delete each vertex of $G_{\mathcal{R}}$ and recompute connected components. Only the four displayed codons disconnect the graph. The same adjacency list shows that UAG and AGG are the only vertices of degree one. □

Theorem 9462.1296 (Largest biconnected block)**.** *The largest biconnected component of $G_{\mathcal{R}}$ is the punctured cube*

$$\mathrm{WNA} \setminus \{\mathrm{ACA}\} = \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{AAA}, \mathrm{AGA}\}.$$

Its reassignment weight is 41*.*

Proof. The biconnected components of the support graph are

$$\{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{AAA}, \mathrm{AGA}\},$$
$$\{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\}, \quad \{\mathrm{AGA}, \mathrm{AGG}\}, \quad \{\mathrm{CUA}, \mathrm{UUA}\}, \quad \{\mathrm{UAA}, \mathrm{UAG}\}.$$

The first is largest. Its weight is

$$1 + 1 + 8 + 15 + 5 + 3 + 8 = 41.$$

□

Theorem 9462.1297 (Support skeleton statement)**.** *The reassigned support has three compatible descriptions:*

$$\textit{prime support formula: } \mathrm{CUN} \cup \mathrm{AGR} \cup \mathrm{UNA} \cup \mathrm{WWA} \cup \mathrm{UAR},$$
$$\textit{cube-complex skeleton: } (\mathrm{WNA} \setminus \{\mathrm{ACA}\}) \cup \mathrm{CUN} \cup \{\mathrm{UAG}, \mathrm{AGG}\},$$
$$\textit{mass core: } \mathrm{WRA} \subset \mathrm{WRR}.$$

The first is the prime-implicant support formula, the second is the induced-graph skeleton, and the third is the weighted high-mass core.

Proof. Combine Definition 9462.1281, Definition 9462.1293, and Definition 9462.1287. □

Concretized in §3463: The punctured WNA cube, CUN square, bridge-edge, and articulation-gate support graph is realized as finite NameCert data.

Definition 9462.1298 (Outer reassignment boundary)**.** *The outer boundary of the reassigned support is*

$$\partial\mathcal{R} = \{z \notin \mathcal{R} : \exists r \in \mathcal{R},\ d_H(z, r) = 1\}.$$

For $z \in \partial\mathcal{R}$ define the boundary degree and boundary pressure by

$$d_\partial(z) = |\{r \in \mathcal{R} : d_H(z, r) = 1\}|,$$
$$P_\partial(z) = \sum_{r \in \mathcal{R},\ d_H(z,r)=1} w(r).$$

Theorem 9462.1299 (Outer boundary size)**.** *The reassigned support has* 46 *boundary edges and* 33 *distinct outer boundary codons:*

$$|\mathcal{R}| = 13, \qquad |E(G_{\mathcal{R}})| = 16, \qquad |\partial\mathcal{R}| = 33.$$

Proof. Each vertex of Q_6 has degree 6, so the 13 support vertices have 78 incident half-edges. The 16 internal edges of Definition 9462.1289 consume 32 of these half-edges. The remaining 46 half-edges leave the support. Enumerating their external endpoints gives 33 distinct codons. □

Theorem 9462.1300 (Highest-pressure inactive boundary codons)**.** *The highest-degree inactive boundary codons, ordered by boundary degree and then boundary pressure, begin as follows:*

codon	*class*	d_∂	P_∂	*support neighbors*
UGG	W	3	33	UAG, UGA, AGG
AAG	K	3	21	UAG, AAA, AGG
ACA	T	3	14	UCA, AUA, AGA
UUG	L	3	14	UUA, UAG, CUG
CAG	Q	2	13	UAG, CUG
CAA	Q	2	9	UAA, CUA
GUA	V	2	6	AUA, CUA
CCA	P	2	2	UCA, CUA
UUU	F	2	2	UUA, CUU.

Proof. For every $z \notin \mathcal{R}$, inspect the six Hamming neighbors of z. Retain z when at least one neighbor lies in $\mathcal{R}$, count these neighbors, and sum their reassignment weights. Sorting by boundary degree and pressure gives the displayed rows. □

Corollary 9462.1301 (Inactive envelope corners)**.** *The three false-positive codons in the three-module envelope are exactly*

$$\text{UGG}, \qquad \text{AAG}, \qquad \text{ACA}.$$

They are inactive boundary codons:

$$\text{WRR} \cup \text{CUN} \cup \text{WYA} = \mathcal{R} \cup \{\text{UGG}, \text{AAG}, \text{ACA}\}.$$

Proof. The envelope identity is Definition 9462.1284. The displayed codons also occur in the boundary table of Definition 9462.1300. □

Definition 9462.1302 (Q_2 completion frontier)**.** *The first Q_2-completion frontier is the set of codons $z \notin \mathcal{R}$ for which some Q_2 face contains exactly three vertices of $\mathcal{R}$ and has z as its fourth vertex:*

$$C_1(\mathcal{R}) = \{z \notin \mathcal{R} : \exists Q_2 \text{ face } F,\ |F \cap \mathcal{R}| = 3,\ z \in F \setminus \mathcal{R}\}.$$

Theorem 9462.1303 (First completion frontier)**.** *The first Q_2-completion frontier is*

$$C_1(\mathcal{R}) = \{\text{UGG}, \text{AAG}, \text{ACA}, \text{UUG}, \text{CAA}, \text{CCA}, \text{GUA}, \text{UUU}\}.$$

It is obtained from 13 *three-out-of-four Q_2 faces.*

Proof. Enumerate the 240 axis-aligned Q_2 faces of Q_6 and select those with exactly three vertices in $\mathcal{R}$. There are 13 such faces. Their missing vertices, after removing duplicates, are exactly the eight codons displayed above. □

Theorem 9462.1304 (Q_2 completion percolation)**.** *Starting from $S_0 = \mathcal{R}$ and iterating the rule*

$$S_{n+1} = S_n \cup \{z : \exists Q_2 \text{ face } F,\ |F \cap S_n| \geq 3,\ z \in F\},$$

the process reaches all 64 *codons in five stages. The stage sizes are*

stage	*added*	*codons*
1	8	AAG, ACA, CAA, CCA, GUA, UGG, UUG, UUU
2	16	ACG, AUG, AUU, CAG, CAU, CCG, CCU, CGA, GAA, GCA, GUG, GUU, UAU, UCG, UCU, UUC
3	16	AAU, ACU, AUC, CAC, CCC, CGG, CGU, GAG, GAU, GCG, GCU, GGA, GUC, UAC, UCC, UGU
4	9	AAC, ACC, AGU, CGC, GAC, GCC, GGG, GGU, UGC
5	2	AGC, GGC.

Proof. At each stage enumerate all Q_2 faces and adjoin every missing vertex of each face already containing at least three vertices of the current set. The displayed stage table is the direct finite iteration; after the fifth stage the set has size 64. □

Theorem 9462.1305 (Boundary class pressure)**.** *Aggregating outer-boundary codons by standard-code translation class, the highest-pressure classes are*

class	*boundary edges*	*pressure*	*codons*
W	3	33	UGG
Q	4	22	CAA, CAG
T	4	22	ACA, ACG
K	3	21	AAG
S	4	18	AGC, AGU, UCG, UCU
Y	2	18	UAC, UAU
G	2	16	GGA, GGG
C	1	15	UGU
R	1	15	CGA
L	3	14	UUG.

Proof. For each boundary codon z, take its standard-code class and add $d_\partial(z)$ and $P_\partial(z)$ to that class. Sorting the resulting class rows by pressure gives the displayed table. □

Theorem 9462.1306 (Boundary-closure structure)**.** *The reassigned support is a cubical seed with inactive boundary anchors:*

$$\mathcal{R} \quad + \quad \{\mathrm{UGG}, \mathrm{AAG}, \mathrm{ACA}\} \quad = \quad \textit{the active part of } \mathrm{WRR} \cup \mathrm{CUN} \cup \mathrm{WYA}.$$

The three inactive codons are high-pressure boundary codons, the first Q_2-completion frontier contains eight codons, and repeated Q_2-completion from $\mathcal{R}$ percolates through Q_6.

Proof. Combine Definition 9462.1301, Definition 9462.1303, and Definition 9462.1304. □

Concretized in §3508: finite boundary-trigger closure packet for six-tile DNA window rows

Concretized in §3842: Boundary pressure ledgers are realized as finite support, outer-boundary, degree, pressure, class aggregation, frontier, transport, replay, provenance, and NameCert rows.

Definition 9462.1307 (Inactive-anchor closure)**.** *Let*

$$\mathcal{A} = \{\mathrm{UGG}, \mathrm{AAG}, \mathrm{ACA}\}$$

be the inactive envelope corners from Definition 9462.1301. The inactive-anchor closure of the reassigned support is

$$\overline{\mathcal{R}} = \mathcal{R} \cup \mathcal{A}.$$

Definition 9462.1308 (Cubical support complex)**.** *For a codon set $S \subset Q_6$, let $K(S)$ be the cubical complex whose cells are exactly the axis-aligned subcubes of Q_6 whose vertices all lie in S.*

Concretized in §5852: Inactive-anchor cubical support is realized as finite NameCert rows for codon windows, boundary tiles, inactive anchors, cubical support cells, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

Theorem 9462.1309 (Active cubical f-vector)**.** *The active support complex has*

$$f(K(\mathcal{R})) = (13, 16, 4, 0, \ldots).$$

The four 2-cells are the square faces

$$\mathrm{CUN}, \quad \mathrm{UNA}, \quad \mathrm{WWA}, \quad \mathrm{WRA}.$$

There are no 3-cells.

Proof. Enumerate every axis-aligned subcube of Q_6 and retain those whose vertices lie in $\mathcal{R}$. The retained cells consist of the 13 vertices of $\mathcal{R}$, the 16 internal Hamming edges from Definition 9462.1289, and the four square faces listed in Definition 9462.1291. No retained cube has dimension 3 or higher. □

Theorem 9462.1310 (Active cubical homology)**.** *Over $\mathbb{F}_2$, the cubical homology of $K(\mathcal{R})$ has Betti vector*

$$\beta(K(\mathcal{R})) = (1, 0, 0).$$

Equivalently, the graph cycles in $G_{\mathcal{R}}$ are filled by the four square 2-cells of the cubical complex.

Proof. The cellular boundary ranks over $\mathbb{F}_2$ are

$$\operatorname{rank} \partial_1 = 12, \qquad \operatorname{rank} \partial_2 = 4.$$

Together with $f(K(\mathcal{R})) = (13, 16, 4)$ this gives

$$\begin{aligned} \beta_0 &= 13 - 12 = 1, \\ \beta_1 &= 16 - 12 - 4 = 0, \\ \beta_2 &= 4 - 4 = 0. \end{aligned}$$

□

Theorem 9462.1311 (Inactive-anchor closure formula)**.** *The inactive-anchor closure is exactly*

$$\overline{\mathcal{R}} = \mathrm{WNA} \cup \mathrm{WRR} \cup \mathrm{CUN}.$$

Here

$$\begin{aligned} \mathrm{WNA} &= \{\mathrm{UUA}, \mathrm{UCA}, \mathrm{UAA}, \mathrm{UGA}, \mathrm{AUA}, \mathrm{ACA}, \mathrm{AAA}, \mathrm{AGA}\}, \\ \mathrm{WRR} &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}, \\ \mathrm{CUN} &= \{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\}. \end{aligned}$$

Proof. The only codons in the displayed union that are not in $\mathcal{R}$ are UGG, AAG, and ACA, which are the inactive anchors. Conversely, every codon of $\mathcal{R}$ lies in one of the three displayed motifs by Definition 9462.1293 and Definition 9462.1284. □

Theorem 9462.1312 (Closure cubical f-vector)**.** *The inactive-anchor closure complex has*

$$f(K(\overline{\mathcal{R}})) = (16, 25, 12, 2, 0, \ldots).$$

The two 3*-cells are precisely the cubes* WNA *and* WRR*.*

Proof. Enumerating all internal subcubes of $\overline{\mathcal{R}}$ gives 16 vertices, 25 edges, 12 square faces, and 2 cubes. No retained cube has dimension greater than 3. The two retained 3-cubes are exactly the two eight-vertex motifs WNA and WRR. □

Theorem 9462.1313 (Closure cubical homology)**.** *Over* $\mathbb{F}_2$*, the closure complex has Betti vector*

$$\beta(K(\overline{\mathcal{R}})) = (1, 0, 0, 0).$$

Proof. The cellular boundary ranks are

$$\operatorname{rank} \partial_1 = 15, \qquad \operatorname{rank} \partial_2 = 10, \qquad \operatorname{rank} \partial_3 = 2.$$

Together with $f(K(\overline{\mathcal{R}})) = (16, 25, 12, 2)$, these ranks give

$$\begin{aligned} \beta_0 &= 16 - 15 = 1, \\ \beta_1 &= 25 - 15 - 10 = 0, \\ \beta_2 &= 12 - 10 - 2 = 0, \\ \beta_3 &= 2 - 2 = 0. \end{aligned}$$

□

Theorem 9462.1314 (Hinge square)**.** *The two closure cubes meet in the square*

$$\mathrm{WNA} \cap \mathrm{WRR} = \mathrm{WRA}.$$

Explicitly,

$$\mathrm{WRA} = \{\mathrm{UAA}, \mathrm{UGA}, \mathrm{AAA}, \mathrm{AGA}\},$$

and its reassignment weight is 34*.*

Proof. The first base is constrained to W in both WNA and WRR. The second base is $N \cap R = R$, and the third base is $A \cap R = A$. Hence the intersection is WRA. The weight is

$$w(\mathrm{UAA}) + w(\mathrm{UGA}) + w(\mathrm{AAA}) + w(\mathrm{AGA}) = 8 + 15 + 3 + 8 = 34.$$

□

Theorem 9462.1315 (Inactive anchors eliminate the active leaves)**.** *In* $G_{\mathcal{R}}$*, the codons* UAG *and* AGG *are leaves. In the closure graph on* $\overline{\mathcal{R}}$*, both have degree* 3*:*

$$\begin{aligned} N_{\overline{\mathcal{R}}}(\mathrm{UAG}) &= \{\mathrm{UAA}, \mathrm{UGG}, \mathrm{AAG}\}, \\ N_{\overline{\mathcal{R}}}(\mathrm{AGG}) &= \{\mathrm{AGA}, \mathrm{UGG}, \mathrm{AAG}\}. \end{aligned}$$

Proof. The active graph has leaf set $\{\mathrm{UAG}, \mathrm{AGG}\}$ by Definition 9462.1295. Adding UGG and AAG supplies the two additional Hamming neighbors displayed above for each leaf. □

Theorem 9462.1316 (Boundary edge comparison)**.** *The active support and the inactive-anchor closure have the same number of boundary edges:*

$$|\partial_E \mathcal{R}| = |\partial_E \overline{\mathcal{R}}| = 46.$$

Proof. For the active support,

$$13 \cdot 6 - 2 \cdot 16 = 46.$$

For the closure,

$$16 \cdot 6 - 2 \cdot 25 = 46.$$

Thus the inactive anchors increase internal dimension without reducing the number of external boundary edges. □

Theorem 9462.1317 (Isoperimetric comparison)**.** *In Q_6, the maximum possible number of internal edges among 13 vertices is 22, while $\mathcal{R}$ has 16. The maximum possible number among 16 vertices is 32, while $\overline{\mathcal{R}}$ has 25.*

Proof. Use the hypercube edge-isoperimetric recurrence

$$F(d, n) = \max_k \big(F(d-1, k) + F(d-1, n-k) + \min(k, n-k)\big).$$

The finite computation gives $F(6, 13) = 22$ and $F(6, 16) = 32$. □

Theorem 9462.1318 (Inactive-anchor cubical closure)**.** *The active support is a low-dimensional cubical seed, and the inactive anchors complete it to*

$$\overline{\mathcal{R}} = \mathrm{WNA} \cup \mathrm{WRR} \cup \mathrm{CUN}, \qquad \mathrm{WNA} \cap \mathrm{WRR} = \mathrm{WRA}.$$

Both $K(\mathcal{R})$ and $K(\overline{\mathcal{R}})$ are connected with vanishing higher Betti numbers.

Proof. The closure formula is Definition 9462.1311. The homology claims are Definition 9462.1310 and Definition 9462.1313. The hinge identity is Definition 9462.1314. □

Definition 9462.1319 (Median closure)**.** *For three vertices $x, y, z \in Q_6$, let $m(x, y, z)$ be their coordinatewise majority vertex. The median closure $\mathrm{Med}(S)$ of $S \subset Q_6$ is the least subset of Q_6 containing S and closed under m.*

Theorem 9462.1320 (Median closure of the reassigned support)**.** *The median closure of the reassigned support is*

$$\mathrm{Med}(\mathcal{R}) = \mathrm{WNR} \cup \mathrm{CUN}.$$

Equivalently,

$$\mathrm{Med}(\mathcal{R}) = \{\mathrm{U}, \mathrm{A}\} \times \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\} \times \{\mathrm{A}, \mathrm{G}\} \ \cup \ \mathrm{CUN}.$$

It has 20 codons.

Proof. Iterate coordinatewise majority over triples of codons until no new codon appears. The resulting set has the 20 codons

$$\begin{gathered}\mathrm{AAA}, \mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AGA}, \mathrm{AGG}, \mathrm{AUA}, \mathrm{AUG}, \\ \mathrm{CUA}, \mathrm{CUC}, \mathrm{CUG}, \mathrm{CUU}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UCA}, \mathrm{UCG}, \\ \mathrm{UGA}, \mathrm{UGG}, \mathrm{UUA}, \mathrm{UUG}.\end{gathered}$$

This is exactly $\mathrm{WNR} \cup \mathrm{CUN}$. The displayed union is itself median-closed and contains $\mathcal{R}$, so it is the least such set. □

Concretized in §3545: Median closure packets realize coordinatewise majority saturation, inactive-corner readback, f-vector, and maximal-cell ledgers as finite NameCert data.

Corollary 9462.1321 (Median inactive corners)**.** *The median closure adds seven inactive codons to $\mathcal{R}$:*

$$\mathrm{Med}(\mathcal{R}) \setminus \mathcal{R} = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

Relative to the inactive-anchor closure, the additional median codons are

$$\mathrm{Med}(\mathcal{R}) \setminus \overline{\mathcal{R}} = \{\mathrm{UUG}, \mathrm{UCG}, \mathrm{AUG}, \mathrm{ACG}\} = \mathrm{WYG}.$$

Proof. Subtract the support and the inactive-anchor closure from the explicit median closure in Definition 9462.1320. □

Theorem 9462.1322 (Closure chain)**.** *The closure hierarchy is*

$$\mathcal{R} \subset \overline{\mathcal{R}} \subset \mathrm{Med}(\mathcal{R}) \subset \mathrm{Conv}(\mathcal{R}) = Q_6,$$

with sizes

$$13, \qquad 16, \qquad 20, \qquad 64.$$

Proof. The first two sizes are Definition 9462.1307; the median size is Definition 9462.1320. Iterating hypercube intervals over pairs of vertices of $\mathcal{R}$ produces all 64 vertices of Q_6, so the geodesic convex hull is the whole cube. □

Theorem 9462.1323 (Median closure f-vector)**.** *The cubical complex $K(\mathrm{Med}(\mathcal{R}))$ has*

$$f = (20, 38, 26, 8, 1, 0, \ldots).$$

Thus it has 20 *vertices,* 38 *edges,* 26 *square faces,* 8 *cubes, and one* 4*-cube. The unique* 4*-cube is* WNR*.*

Proof. Enumerate every axis-aligned subcube of Q_6 whose vertices lie in $\mathrm{Med}(\mathcal{R})$. The retained cells have the displayed counts. The only retained 4-cube is the 16-vertex motif WNR. □

Theorem 9462.1324 (Maximal cells of the median closure)**.** *The maximal cubical cells of $K(\mathrm{Med}(\mathcal{R}))$ are*

$$\mathrm{WNR} \quad (Q_4), \qquad \mathrm{CUN} \quad (Q_2), \qquad \mathrm{YUR} \quad (Q_2).$$

Here

$$\mathrm{YUR} = \{\mathrm{UUA}, \mathrm{UUG}, \mathrm{CUA}, \mathrm{CUG}\}.$$

Proof. After enumerating all internal subcubes of the median closure, delete every subcube properly contained in another retained subcube. The three remaining maximal cells are exactly WNR, CUN, and YUR. □

Corollary 9462.1325 (Median bridge face)**.** *The square* YUR *bridges the main Q_4 cell and the CUN module:*

$$\mathrm{YUR} \cap \mathrm{WNR} = \{\mathrm{UUA}, \mathrm{UUG}\}, \qquad \mathrm{YUR} \cap \mathrm{CUN} = \{\mathrm{CUA}, \mathrm{CUG}\}.$$

Proof. Intersect the explicit vertex sets in Definition 9462.1324. □

Theorem 9462.1326 (Median closure homology)**.** *Over $\mathbb{F}_2$, the cubical homology of $K(\mathrm{Med}(\mathcal{R}))$ has Betti vector*

$$\beta = (1, 0, 0, 0, 0).$$

Proof. The cellular boundary ranks are

$$\operatorname{rank} \partial_1 = 19, \qquad \operatorname{rank} \partial_2 = 19, \qquad \operatorname{rank} \partial_3 = 7, \qquad \operatorname{rank} \partial_4 = 1.$$

Together with the f-vector $(20, 38, 26, 8, 1)$, these ranks give

$$\begin{aligned}
\beta_0 &= 20 - 19 = 1, \\
\beta_1 &= 38 - 19 - 19 = 0, \\
\beta_2 &= 26 - 19 - 7 = 0, \\
\beta_3 &= 8 - 7 - 1 = 0, \\
\beta_4 &= 1 - 1 = 0.
\end{aligned}$$

□

Theorem 9462.1327 (Median-closure structure)**.** *The natural median-stabilized envelope of the reassigned support is*

$$\mathrm{Med}(\mathcal{R}) = \mathrm{WNR} \cup \mathrm{CUN}.$$

It extends the inactive-anchor closure by the square WYG*, merges the two inactive-anchor cubes into the main Q_4 cell* WNR*, and remains cubically connected with no higher homology.*

Proof. The formula is Definition 9462.1320. The additional codons are Definition 9462.1321. The cubical and homological statements are Definition 9462.1323, Definition 9462.1324, and Definition 9462.1326. □

Definition 9462.1328 (Median iteration layers)**.** *For $S \subset Q_6$, define $\mathrm{Med}_0(S) = S$ and*

$$\mathrm{Med}_{n+1}(S) = \mathrm{Med}_n(S) \cup \{m(x, y, z) : x, y, z \in \mathrm{Med}_n(S)\}.$$

Here $m(x, y, z)$ is the coordinatewise majority vertex of Definition 9462.1319.

Theorem 9462.1329 (Median closure depth of the reassigned support)**.** *The median iteration of the reassigned support stabilizes after two nontrivial layers. The first layer adds*

$$\Delta_1 = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\},$$

and the second layer adds

$$\Delta_2 = \{\mathrm{ACG}\}.$$

Thus

$$\begin{aligned}
\mathrm{Med}_1(\mathcal{R}) &= \mathcal{R} \cup \Delta_1, & |\mathrm{Med}_1(\mathcal{R})| &= 19, \\
\mathrm{Med}_2(\mathcal{R}) &= \mathrm{Med}(\mathcal{R}) = \mathrm{Med}_1(\mathcal{R}) \cup \{\mathrm{ACG}\}, & |\mathrm{Med}_2(\mathcal{R})| &= 20.
\end{aligned}$$

Proof. Enumerate all triples in $\mathcal{R}$ and apply coordinatewise majority. The codons outside $\mathcal{R}$ are exactly the six codons in Δ_1. Repeating the same operation on $\mathrm{Med}_1(\mathcal{R})$ produces the single additional codon ACG. A further pass produces no additional vertex, so the second layer is the full median closure. □

Corollary 9462.1330 (Inactive median levels)**.** *The inactive codons in the median closure split into three structural levels:*

$$\textit{anchor inactive codons: } \mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG},$$
$$\textit{first median codons: } \mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG},$$
$$\textit{sealing codon: } \mathrm{ACG}.$$

Proof. The first line is the inactive-anchor set of Definition 9462.1307. The second and third lines are the remaining elements of Δ_1 and Δ_2 in Definition 9462.1329. □

Theorem 9462.1331 (Punctured median Q_4)**.** *The first median layer is*

$$\mathrm{Med}_1(\mathcal{R}) = (\mathrm{WNR} \setminus \{\mathrm{ACG}\}) \cup \mathrm{CUN}.$$

Consequently ACG *is the unique missing vertex of the main* WNR *4-cube at the first median layer.*

Proof. The full median closure is $\mathrm{Med}(\mathcal{R}) = \mathrm{WNR} \cup \mathrm{CUN}$ by Definition 9462.1320. By Definition 9462.1329, the only vertex of this median closure absent from $\mathrm{Med}_1(\mathcal{R})$ is ACG. □

Theorem 9462.1332 (Sealing-corner f-vector transition)**.** *The cubical f-vector changes from*

$$f(K(\mathrm{Med}_1(\mathcal{R}))) = (19, 34, 20, 4, 0, 0, \ldots)$$

to

$$f(K(\mathrm{Med}(\mathcal{R}))) = (20, 38, 26, 8, 1, 0, \ldots).$$

The difference is

$$(1, 4, 6, 4, 1, 0, \ldots).$$

Proof. At the first median layer, the main WNR cell is a 4-cube with one vertex, ACG, removed, together with the unchanged CUN module. Restoring one missing corner of a 4-cube contributes $\binom{4}{k}$ k-cells for $k = 0, 1, 2, 3, 4$, hence the difference vector $(1, 4, 6, 4, 1)$. Direct enumeration of all internal cubical cells gives the two displayed f-vectors. □

Corollary 9462.1333 (Top-dimensional sealing corner)**.** *The codon* ACG *is not an active reassignment codon and is not one of the three inactive-anchor codons. Its median role is top-dimensional: it seals the punctured* WNR *4-cube into a complete Q_4 cell.*

Proof. The codon ACG is absent from $\mathcal{R}$ and from the inactive-anchor set $\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$. Its cellular contribution is the top-dimensional transition of Definition 9462.1332. □

Theorem 9462.1334 (Median witnesses for the two layers)**.** *The first-layer inactive codons have active median witnesses*

$$\begin{aligned}
\mathrm{ACA} &= m(\mathrm{AGA}, \mathrm{AUA}, \mathrm{UCA}),\\
\mathrm{UGG} &= m(\mathrm{AGG}, \mathrm{UAG}, \mathrm{UGA}),\\
\mathrm{AAG} &= m(\mathrm{AAA}, \mathrm{AGG}, \mathrm{UAG}),\\
\mathrm{AUG} &= m(\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUG}),\\
\mathrm{UCG} &= m(\mathrm{AGG}, \mathrm{UCA}, \mathrm{CUC}),\\
\mathrm{UUG} &= m(\mathrm{AUA}, \mathrm{CUG}, \mathrm{UAG}).
\end{aligned}$$

The sealing codon has a second-layer witness

$$\mathrm{ACG} = m(\mathrm{UGG}, \mathrm{AUG}, \mathrm{ACA}).$$

Moreover, ACG *is not the median of any three active codons from* $\mathcal{R}$.

Proof. Each displayed equality is checked by applying coordinatewise majority in the two-axis coding U = 00, C = 01, A = 10, G = 11. The exhaustive active-triple enumeration over $\mathcal{R}$ produces exactly Δ_1 as the points outside $\mathcal{R}$, and therefore omits ACG. The next enumeration over $\mathrm{Med}_1(\mathcal{R})$ adds only ACG. □

Theorem 9462.1335 (Median depth and sealing corner)**.** *The median-stabilized envelope is obtained through the chain*

$$\mathcal{R} \longrightarrow (\mathrm{WNR} \setminus \{\mathrm{ACG}\}) \cup \mathrm{CUN} \longrightarrow \mathrm{WNR} \cup \mathrm{CUN}.$$

The only strict second-layer median point is ACG*, and its addition converts the punctured main* Q_4 *into the complete* WNR *cell.*

Proof. The chain is Definition 9462.1329 together with Definition 9462.1331. The strict second-layer property is Definition 9462.1334; the cellular sealing statement is Definition 9462.1332. □

Concretized in §3981: finite median-depth sealing packet for two-layer stabilization, sealing-codon exposure, transport, replay, provenance, and local naming

Concretized in §7668: The median-depth sealing corner is realized as finite median-layer, unique-corner, f-vector, witness, transport, replay, provenance, and NameCert rows.

Definition 9462.1336 (Active median leakage)**.** *Let*

$$\mathcal{T}_3(\mathcal{R}) = \{\{x, y, z\} : x, y, z \in \mathcal{R} \text{ distinct}\}.$$

For a triple $T = \{x, y, z\}$ *write* $m(T) = m(x, y, z)$*. A triple is closed when* $m(T) \in \mathcal{R}$ *and leaking when* $m(T) \notin \mathcal{R}$*.*

Theorem 9462.1337 (Median-almost-closed active support)**.** *The reassigned support has* 286 *active triples. Among them,* 264 *triples are closed and* 22 *triples leak to inactive codons:*

$$\frac{264}{286} = 0.923076\ldots, \qquad \frac{22}{286} = 0.076923\ldots.$$

Thus $\mathcal{R}$ *is nearly closed under the median operation on distinct active triples.*

Proof. There are $\binom{13}{3} = 286$ distinct active triples. Enumerating them and applying coordinatewise majority gives 264 outputs in $\mathcal{R}$ and 22 outputs outside $\mathcal{R}$. □

Concretized in §2296: Median leakage spectra are realized as finite NameCert packets over active triples, median outputs, inactive leakage rows, centrality counts, direct-output multiplicities, transport, replay, provenance, and local naming.

Corollary 9462.1338 (Random median comparison)**.** *In* 100000 *random* 13*-point subsets of* Q_6*, using seed* 20260522*, the mean number of closed active triples is* 157.49515*. The mean one-step median closure size is* 43.79654*, and the mean number of one-step leakage outputs is* 30.79654*. The estimated tail probabilities are*

$$\Pr(\textit{closed triples} \geq 264) = 0.00007, \qquad \Pr(\textit{leakage outputs} \leq 6) = 0.00034.$$

Proof. For each random subset, enumerate its distinct triples and apply the same median-output count used in Definition 9462.1337. The displayed means and tail probabilities are the direct Monte Carlo tallies. □

Theorem 9462.1339 (One-step leakage outputs)**.** *The one-step active leakage set is*

$$\{m(T) : T \in \mathcal{T}_3(\mathcal{R})\} \setminus \mathcal{R} = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

Consequently

$$\mathrm{Med}_1(\mathcal{R}) = \mathcal{R} \cup \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

Proof. The six displayed codons are the distinct inactive outputs among the 22 leaking triples of Definition 9462.1337. This is the same first median layer as in Definition 9462.1329. □

Corollary 9462.1340 (ACG is not a direct leakage point)**.** *The codon* ACG *is the only inactive codon in* $\mathrm{Med}(\mathcal{R}) \setminus \mathcal{R}$ *that is not produced by any distinct active triple.*

Proof. The full inactive median set is

$$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

Subtract the one-step leakage set of Definition 9462.1339. □

Definition 9462.1341 (Median centrality)**.** *For* $p \in Q_6$*, define*

$$\mu_{\mathcal{R}}(p) = |\{T \in \mathcal{T}_3(\mathcal{R}) : m(T) = p\}|.$$

Theorem 9462.1342 (Median centrality ranking)**.** *The nonzero median centralities, together with the sealing value, are:*

codon	*active*	$\mu_{\mathcal{R}}$
UUA	*yes*	61
UAA	*yes*	46
CUA	*yes*	31
AAA	*yes*	19
UGA	*yes*	19
CUG	*yes*	16
UAG	*yes*	16
AGA	*yes*	14
AUA	*yes*	14
UCA	*yes*	14
CUU	*yes*	8
UUG	*no*	8
AAG	*no*	4
UGG	*no*	4
AGG	*yes*	3
CUC	*yes*	3
ACA	*no*	2
AUG	*no*	2
UCG	*no*	2
ACG	*no*	0.

Proof. Count the median outputs of all 286 active triples and group the results by output codon. The codon ACG has zero direct centrality by Definition 9462.1340. □

Corollary 9462.1343 (Mass core and median center differ)**.** *The median center is not the same as the reassignment-mass core. In particular,* UUA *has the largest median centrality* $\mu_{\mathcal{R}}(\mathrm{UUA}) = 61$*, whereas high-weight AGR-side codons such as* AGG *have much smaller direct median centrality.*

Proof. Compare Definition 9462.1342 with the weighted reassignment counts in Definition 9462.1256. □

Theorem 9462.1344 (Direct leakage spectrum)**.** *The inactive one-step leakage multiplicities are*

inactive output	*active triples*
UUG	8
AAG	4
UGG	4
ACA	2
AUG	2
UCG	2.

Proof. Restrict the median centrality count to the six inactive outputs in Definition 9462.1339. □

Theorem 9462.1345 (Direct-leakage catalyst)**.** *Among the* 22 *leaking active triples, active codons occur with the following multiplicities:*

active codon	*leaking triples containing it*
AGG	15
UAG	10
CUC	8
CUG	8
AUA	7
UCA	7
UUA	4
AAA	3
UGA	3
AGA	1.

Thus AGG *is the principal direct-leakage catalyst.*

Proof. Enumerate the 22 leaking triples and count active codon appearances inside those triples. □

Corollary 9462.1346 (Median gate role of AGG)**.** *The codon* AGG *is both median-essential and the leading direct-leakage catalyst. It appears in* 15 *of the* 22 *leaking triples.*

Proof. Median-essentiality is Definition 9462.1771. The leakage catalyst count is Definition 9462.1345. □

Theorem 9462.1347 (Minimum inactive median generators)**.** *The minimum size of an active subset* $G \subseteq \mathcal{R}$ *satisfying*

$$\mathrm{Med}(\mathcal{R}) \setminus \mathcal{R} \subseteq \mathrm{Med}(G)$$

is 5. *The four minimum subsets are*

$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUC}, \mathrm{UAA}, \mathrm{UCA}\},$$
$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUC}, \mathrm{UAG}, \mathrm{UCA}\},$$
$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUG}, \mathrm{UAA}, \mathrm{UCA}\},$$
$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUG}, \mathrm{UAG}, \mathrm{UCA}\}.$$

Proof. Enumerate all active subsets $G \subseteq \mathcal{R}$ and compute $\mathrm{Med}(G)$. No subset of size at most 4 contains all seven inactive median codons in its closure. Exactly the four displayed subsets of size 5 do. □

Corollary 9462.1348 (Inactive leakage core)**.** *Every minimum inactive median generator contains*

$$\mathrm{AGG}, \qquad \mathrm{AUA}, \qquad \mathrm{UCA}.$$

The remaining choices are one of $\mathrm{CUC}, \mathrm{CUG}$ *and one of* $\mathrm{UAA}, \mathrm{UAG}$.

Proof. Intersect the four generators in Definition 9462.1347. □

Theorem 9462.1349 (ACG has twelve second-layer derivations)**.** *Inside* $\mathrm{Med}_1(\mathcal{R})$*, exactly* 12 *distinct triples have median* ACG.

Proof. Enumerate all distinct triples in $\mathrm{Med}_1(\mathcal{R})$ and retain those whose median is ACG. The count is 12. Examples include

$$\mathrm{ACG} = m(\mathrm{AAG}, \mathrm{ACA}, \mathrm{UCG}), \qquad \mathrm{ACG} = m(\mathrm{ACA}, \mathrm{AUG}, \mathrm{UGG}).$$

□

Theorem 9462.1350 (Controlled median leakage)**.** *The median generation path of the reassigned support is*

$$\mathcal{R} \xrightarrow{\text{22 leaking triples}} \mathcal{R} \cup \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\} \xrightarrow{\mathrm{ACG}} \mathrm{WNR} \cup \mathrm{CUN}.$$

Thus $\mathcal{R}$ *is a median-almost-closed active support whose inactive median closure is generated by a small controlled leakage spectrum and a single second-layer sealing codon.*

Proof. The first arrow is Definition 9462.1337 and Definition 9462.1339. The second arrow is Definition 9462.1340 together with Definition 9462.1349. The terminal formula is Definition 9462.1320. □

Definition 9462.1351 (Hypercube automorphism stabilizer)**.** *The automorphism group of* Q_6 *is*

$$\mathrm{Aut}(Q_6) \cong C_2^6 \rtimes S_6, \qquad |\mathrm{Aut}(Q_6)| = 2^6 \cdot 6! = 46080.$$

For $S \subseteq Q_6$*, write*

$$\mathrm{Stab}(S) = \{\phi \in \mathrm{Aut}(Q_6) : \phi(S) = S\}.$$

For a weight function $w : Q_6 \to \mathbb{R}_{\geq 0}$*, write*

$$\mathrm{Stab}(w) = \{\phi \in \mathrm{Aut}(Q_6) : w(\phi(x)) = w(x) \text{ for all } x \in Q_6\}.$$

Theorem 9462.1352 (Active-support stabilizer)**.** *The active reassignment support has*

$$|\operatorname{Stab}(\mathcal{R})| = 2.$$

The nonidentity element is the coordinate transposition

$$\tau : s_1 \leftrightarrow f_2,$$

with all other bit coordinates fixed.

Proof. Enumerate all 46080 automorphisms of Q_6 and retain those whose image of $\mathcal{R}$ is exactly $\mathcal{R}$. The retained automorphisms are the identity and the transposition $s_1 \leftrightarrow f_2$. □

Corollary 9462.1353 (Hidden C_2 orbit structure)**.** *Under τ, the nontrivial active-support orbits are*

$$\{\mathrm{AAA}, \mathrm{UGA}\}, \qquad \{\mathrm{AUA}, \mathrm{UCA}\}.$$

The remaining active codons are fixed by τ.

Proof. Apply the bit transposition $s_1 \leftrightarrow f_2$ to each codon of $\mathcal{R}$. □

Theorem 9462.1354 (Persistence of the hidden C_2)**.** *The inactive-anchor closure and the first median layer have the same stabilizer size:*

$$|\operatorname{Stab}(\overline{\mathcal{R}})| = 2, \qquad |\operatorname{Stab}(\operatorname{Med}_1(\mathcal{R}))| = 2.$$

In both cases the nonidentity stabilizer is again $\tau : s_1 \leftrightarrow f_2$.

Proof. Enumerate all automorphisms of Q_6 and test set preservation for $\overline{\mathcal{R}}$ and $\operatorname{Med}_1(\mathcal{R})$. Each stabilizer contains exactly the identity and τ. □

Corollary 9462.1355 (Orbit growth under τ)**.** *The nontrivial τ-orbits grow along the closure chain as*

$$\begin{gathered}\mathcal{R} : \{\mathrm{AAA}, \mathrm{UGA}\}, \quad \{\mathrm{AUA}, \mathrm{UCA}\}, \\ \overline{\mathcal{R}} : \{\mathrm{AAG}, \mathrm{UGG}\} \textit{ also appears}, \\ \operatorname{Med}_1(\mathcal{R}) : \{\mathrm{AUG}, \mathrm{UCG}\} \textit{ also appears.}\end{gathered}$$

Proof. This is the explicit orbit decomposition of τ on $\mathcal{R}$, $\overline{\mathcal{R}}$, and $\operatorname{Med}_1(\mathcal{R})$. □

Theorem 9462.1356 (Median-closure stabilizer)**.** *The full median closure has*

$$\operatorname{Stab}(\operatorname{Med}(\mathcal{R})) \cong S_3 \times C_2, \qquad |\operatorname{Stab}(\operatorname{Med}(\mathcal{R}))| = 12.$$

The S_3 factor permutes the three coordinates

$$s_1, \qquad s_2, \qquad f_2,$$

and the C_2 factor flips f_3.

Proof. The median closure is $\mathrm{WNR} \cup \mathrm{CUN}$. In bit variables

$$a = s_1, \quad b = f_1, \quad c = s_2, \quad d = f_2, \quad e = s_3, \quad g = f_3,$$

this set is described by

$$(\neg b \wedge e) \ \vee \ (b \wedge \neg a \wedge \neg c \wedge \neg d).$$

The coordinates a, c, d are symmetric in this formula, and g is absent. This gives $S_3 \times C_2$. Exhaustive enumeration of all 46080 hypercube automorphisms gives no additional stabilizers. □

Corollary 9462.1357 (Symmetry restoration at the sealing corner)**.** *The stabilizer size changes along the median chain as*

$$|\operatorname{Stab}(\mathcal{R})| = 2, \qquad |\operatorname{Stab}(\operatorname{Med}_1(\mathcal{R}))| = 2, \qquad |\operatorname{Stab}(\operatorname{Med}(\mathcal{R}))| = 12.$$

Thus the sealing codon ACG *raises the set stabilizer from* C_2 *to* $S_3 \times C_2$*.*

Proof. The first two stabilizers are Definition 9462.1352 and Definition 9462.1354. The last stabilizer is Definition 9462.1356. The equality $\operatorname{Med}_1(\mathcal{R}) = \operatorname{Med}(\mathcal{R}) \setminus \{\mathrm{ACG}\}$ is Definition 9462.1331. □

Theorem 9462.1358 (Median-closure orbits)**.** *Under* $\operatorname{Stab}(\operatorname{Med}(\mathcal{R}))$*, the median closure decomposes into the following orbits:*

size	*orbit*
6	$\{\mathrm{AAA}, \mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{UGA}, \mathrm{UGG}\}$
6	$\{\mathrm{AUA}, \mathrm{AUG}, \mathrm{UAA}, \mathrm{UAG}, \mathrm{UCA}, \mathrm{UCG}\}$
2	$\{\mathrm{AGA}, \mathrm{AGG}\}$
2	$\{\mathrm{UUA}, \mathrm{UUG}\}$
2	$\{\mathrm{CUA}, \mathrm{CUG}\}$
2	$\{\mathrm{CUC}, \mathrm{CUU}\}$.

Proof. Apply the 12 stabilizing automorphisms of Definition 9462.1356 to the 20 codons of $\operatorname{Med}(\mathcal{R})$ and group codons by orbit. □

Theorem 9462.1359 (Orbit-completion role of ACG)**.** *The codon* ACG *is the unique missing point of the orbit*

$$\{\mathrm{AAA}, \mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{UGA}, \mathrm{UGG}\}$$

inside $\operatorname{Med}_1(\mathcal{R})$*. Hence it is both the top-dimensional sealing corner of Definition 9462.1333 and the orbit-completion point for the restored* $S_3 \times C_2$ *symmetry.*

Proof. By Definition 9462.1331, $\operatorname{Med}_1(\mathcal{R})$ is the full median closure with ACG removed. By Definition 9462.1358, ACG lies in the displayed 6-point orbit. The other five points of that orbit are already present in $\operatorname{Med}_1(\mathcal{R})$. □

Theorem 9462.1360 (Weighted symmetry is trivial)**.** *Let* $w(c)$ *be the number of NCBI alternative genetic-code tables in which codon* c *differs from the standard table. Then*

$$|\operatorname{Stab}(w)| = 1.$$

Proof. Enumerate all automorphisms of Q_6 and retain those preserving the weight function on all 64 codons. Only the identity remains. In particular, the support-level symmetry $\tau : s_1 \leftrightarrow f_2$ does not preserve weights: it maps UGA to AAA, while

$$w(\mathrm{UGA}) = 15, \qquad w(\mathrm{AAA}) = 3.$$

□

Corollary 9462.1361 (Geometry and direction)**.** *Support geometry and weight geometry separate. The support and median closure have nontrivial set symmetries, while reassignment frequency breaks all nonidentity hypercube symmetries.*

Proof. Combine Definition 9462.1352, Definition 9462.1356, and Definition 9462.1360. □

Theorem 9462.1362 (Symmetry restoration)**.** *The closure sequence has stabilizer chain*

object	*stabilizer size*	*group*
w	1	*trivial*
$\mathcal{R}$	2	C_2
$\overline{\mathcal{R}}$	2	C_2
$\mathrm{Med}_1(\mathcal{R})$	2	C_2
$\mathrm{Med}(\mathcal{R})$	12	$S_3 \times C_2$.

Thus median closure restores a larger hidden symmetry, while the weighted reassignment function remains fully direction-bearing.

Proof. The table is the collection of Definition 9462.1360, Definition 9462.1352, Definition 9462.1354, and Definition 9462.1356. □

Concretized in §4294: The hidden Q6 stabilizer and orbit-persistence rows are realized as a finite NameCert packet.

Definition 9462.1363 (Median-closure orbit quotient)**.** *Let $M = \mathrm{Med}(\mathcal{R})$ and let $\Gamma = \mathrm{Stab}(M) \cong S_3 \times C_2$. The orbit quotient M/Γ has one state for each Γ-orbit in M. For two orbit states O_i, O_j, define the edge weight*

$$E_{ij} = |\{x \sim y : x \in O_i,\ y \in O_j\}|,$$

where $x \sim y$ means Hamming adjacency in Q_6. For $i = j$, this is the number of internal Hamming edges in the orbit.

Theorem 9462.1364 (Six orbit states of the median closure)**.** *The median closure has the following six orbit states:*

state	*codons*	*size*	*active*	*inactive*	*wei*
Anchor_6	AAA, AAG, ACA, ACG, UGA, UGG	6	AAA, UGA	AAG, ACA, ACG, UGG	
$\mathrm{StopStart}_6$	AUA, AUG, UAA, UAG, UCA, UCG	6	AUA, UAA, UAG, UCA	AUG, UCG	
AGR	AGA, AGG	2	AGA, AGG	$\varnothing$	
UUR	UUA, UUG	2	UUA	UUG	
CUR	CUA, CUG	2	CUA, CUG	$\varnothing$	
CUY	CUC, CUU	2	CUC, CUU	$\varnothing$	

Proof. Apply the 12 automorphisms of $\Gamma = \mathrm{Stab}(\mathrm{Med}(\mathcal{R}))$ to the 20 codons of M. The resulting orbit partition is the one displayed. Weights are sums of NCBI reassignment counts over each orbit. □

Corollary 9462.1365 (Active support as an orbit selector)**.** *The median closure is symmetric, but the active support selects different fractions of different orbit states:*

$$2/6, \quad 4/6, \quad 2/2, \quad 1/2, \quad 2/2, \quad 2/2.$$

Thus $\mathcal{R}$ is a nonuniform active selector inside the symmetric median closure.

Proof. Read off the active columns in Definition 9462.1364. □

Theorem 9462.1366 (Weighted quotient path)**.** *With orbit order*

$$\mathrm{Anchor}_6, \quad \mathrm{StopStart}_6, \quad \mathrm{AGR}, \quad \mathrm{UUR}, \quad \mathrm{CUR}, \quad \mathrm{CUY},$$

the orbit adjacency matrix is

	Anchor_6	$\mathrm{StopStart}_6$	AGR	UUR	CUR	CUY
Anchor_6	3	12	6	0	0	0
$\mathrm{StopStart}_6$	12	3	0	6	0	0
AGR	6	0	1	0	0	0
UUR	0	6	0	1	2	0
CUR	0	0	0	2	1	2
CUY	0	0	0	0	2	1.

After removing internal loops, the quotient graph is the weighted path

$$\mathrm{AGR} - \mathrm{Anchor}_6 - \mathrm{StopStart}_6 - \mathrm{UUR} - \mathrm{CUR} - \mathrm{CUY}.$$

Proof. Enumerate all 38 Hamming edges internal to $\mathrm{Med}(\mathcal{R})$ and group each edge by the two orbit states of its endpoints. The nonzero cross-orbit weights are

$$\mathrm{AGR} - \mathrm{Anchor}_6 : 6, \qquad \mathrm{Anchor}_6 - \mathrm{StopStart}_6 : 12,$$
$$\mathrm{StopStart}_6 - \mathrm{UUR} : 6, \qquad \mathrm{UUR} - \mathrm{CUR} : 2, \qquad \mathrm{CUR} - \mathrm{CUY} : 2.$$

There are no other cross-orbit edges. □

Corollary 9462.1367 (Six-state codon-window automaton)**.** *The* 20*-codon median closure admits a six-state quotient dynamics:*

$$\mathrm{AGR} \longleftrightarrow \mathrm{Anchor}_6 \longleftrightarrow \mathrm{StopStart}_6 \longleftrightarrow \mathrm{UUR} \longleftrightarrow \mathrm{CUR} \longleftrightarrow \mathrm{CUY}.$$

Proof. This is the loop-free quotient graph of Definition 9462.1366. □

Theorem 9462.1368 (Inactive median layer)**.** *Let* $I = M \setminus \mathcal{R}$*. Then*

$$I = \mathrm{WYG} \cup \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}.$$

Here

$$\mathrm{WYG} = \{\mathrm{UUG}, \mathrm{UCG}, \mathrm{AUG}, \mathrm{ACG}\}.$$

The induced Hamming graph on I *is the square* WYG *together with three leaves attached by*

$$\mathrm{AAG} - \mathrm{AUG}, \qquad \mathrm{ACA} - \mathrm{ACG}, \qquad \mathrm{UGG} - \mathrm{UCG}.$$

Proof. The inactive codons are

$$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

The codons $\mathrm{UUG}, \mathrm{UCG}, \mathrm{AUG}, \mathrm{ACG}$ form the WYG square. Direct Hamming-edge enumeration gives the three displayed leaf edges and the four square edges. □

Corollary 9462.1369 (Inactive layer is contractible)**.** *The cubical complex induced by* I *has*

$$f = (7, 7, 1, 0, \ldots), \qquad \beta = (1, 0, 0).$$

Thus the inactive layer is a contractible cubical patch.

Proof. The graph has 7 vertices and 7 edges, and the WYG square contributes one 2-cell. The cellular boundary ranks are $\operatorname{rank}\partial_1 = 6$ and $\operatorname{rank}\partial_2 = 1$, giving the displayed Betti vector. □

Definition 9462.1370 (Active-inactive interface)**.** *The active-inactive interface is the set of Hamming edges*

$$\partial_{A/I} = \{x - y : x \in \mathcal{R},\ y \in I,\ d_H(x, y) = 1\}.$$

Theorem 9462.1371 (Active-inactive interface)**.** *The active-inactive interface has* 15 *edges. Grouped by inactive endpoint, the interface degrees are*

inactive codon	*active neighbors*	*degree*
AAG	AAA, AGG, UAG	3
ACA	AGA, AUA, UCA	3
ACG	AGG	1
AUG	AUA	1
UCG	UCA	1
UGG	AGG, UAG, UGA	3
UUG	CUG, UAG, UUA	3.

The active endpoints of maximum interface degree are

$$\text{AGG}, \qquad \text{UAG}.$$

Proof. Enumerate the Hamming edges of M and retain exactly those with one endpoint in $\mathcal{R}$ and the other in I. There are 15 such edges. Sorting by inactive endpoint gives the displayed table. Sorting by active endpoint shows that AGG and UAG each have three inactive neighbors, the largest active-side degree. □

Corollary 9462.1372 (Double gate to the inactive layer)**.** *The codons* AGG *and* UAG *are the two active gates from the reassigned support into the inactive median layer.*

Proof. This is the maximum active-side interface degree statement in Definition 9462.1371. □

Theorem 9462.1373 (ACG as inactive-square contact point)**.** *The codon* ACG *has inactive neighbors*

$$\text{ACA}, \qquad \text{AUG}, \qquad \text{UCG},$$

and active neighbor

$$\text{AGG}.$$

Thus ACG *is the contact point between the inactive* WYG *square and the active gate* AGG*.*

Proof. The inactive-neighbor statement is read from the induced graph on I. The active-neighbor statement is read from the active-inactive interface. Both are direct Hamming-edge enumerations. □

Theorem 9462.1374 (Orbit-interface quotient)**.** *The median closure admits a compact quotient description:*

$$M/\Gamma = \text{AGR} - \text{Anchor}_6 - \text{StopStart}_6 - \text{UUR} - \text{CUR} - \text{CUY},$$

while the inactive layer is the contractible patch

$$\text{WYG} \cup \{\text{AAG}, \text{ACA}, \text{UGG}\}.$$

The active-inactive interface has two active gates, AGG *and* UAG*, and* ACG *is the inactive-square contact point at* AGG*.*

Proof. The quotient path is Definition 9462.1366. The inactive patch is Definition 9462.1368. The interface and contact-point statements are Definition 9462.1371 and Definition 9462.1373. □

Concretized in §5140: The weighted inactive-layer quotient dynamics is realized as a finite orbit-state NameCert packet.

Definition 9462.1375 (One-bit flip dynamics on the median closure)**.** *Let $M = \mathrm{Med}(\mathcal{R})$. From a codon $x \in M$, choose one of the six bit coordinates uniformly and flip it. If the resulting codon remains in M, this is an internal transition. Otherwise it is an outside transition.*

Theorem 9462.1376 (Quotient leakage transition matrix)**.** *On the six orbit states*

$$\mathrm{AGR}, \quad \mathrm{Anchor}_6, \quad \mathrm{StopStart}_6, \quad \mathrm{UUR}, \quad \mathrm{CUR}, \quad \mathrm{CUY},$$

the one-bit flip dynamics has transient transition matrix

	AGR	Anchor_6	$\mathrm{StopStart}_6$	UUR	CUR	CUY	*outside*
AGR	1/6	1/2	0	0	0	0	1/3
Anchor_6	1/6	1/6	1/3	0	0	0	1/3
$\mathrm{StopStart}_6$	0	1/3	1/6	1/6	0	0	1/3
UUR	0	0	1/2	1/6	1/6	0	1/6
CUR	0	0	0	1/6	1/6	1/6	1/2
CUY	0	0	0	0	1/6	1/6	2/3.

Proof. Use the orbit edge counts from Definition 9462.1366. For a state S, the normalizing denominator is $6|S|$. A cross-orbit edge contributes one oriented half-edge from its source state. An internal edge contributes two half-edges within the same state. Dividing these half-edge counts by $6|S|$ gives the displayed matrix and outside column. □

Corollary 9462.1377 (Uniform self-retention)**.** *Every quotient state has the same self-transition probability:*

$$P(S \to S) = 1/6.$$

Proof. Read the diagonal of Definition 9462.1376. The underlying reason is that each orbit has an internal matching coming from the f_3 flip. □

Theorem 9462.1378 (Quotient leakage rates)**.** *The outside probabilities are*

state	*outside probability*
AGR	1/3
Anchor_6	1/3
$\mathrm{StopStart}_6$	1/3
UUR	1/6
CUR	1/2
CUY	2/3.

Proof. For each row of Definition 9462.1376, subtract the internal transition sum from 1. □

Corollary 9462.1379 (CUN tail leakage)**.** *The CUN-side tail has the highest leakage:* CUR *leaks with probability* 1/2*, and* CUY *leaks with probability* 2/3*. The main three-state core* AGR*,* Anchor_6*, and* $\mathrm{StopStart}_6$ *has outside probability* 1/3 *in each state.*

Proof. This is the comparison of the outside probabilities in Definition 9462.1378. □

Theorem 9462.1380 (Expected exit times)**.** *Let T_S be the expected number of one-bit flips before leaving M, starting from quotient state S. Then*

state	T_S
AGR	$1478/487 \approx 3.035$
Anchor_6	$4468/1461 \approx 3.058$
StopStart_6	$4570/1461 \approx 3.128$
UUR	$1716/487 \approx 3.524$
CUR	$1088/487 \approx 2.234$
CUY	$802/487 \approx 1.647.$

Proof. Let Q be the 6×6 transient matrix in Definition 9462.1376. The expected exit vector is

$$T = (I - Q)^{-1}\mathbf{1}.$$

Solving this exact rational linear system gives the displayed values. □

Corollary 9462.1381 (UUR bridge residence)**.** *The state* UUR *has the largest expected exit time, approximately* 3.524 *one-bit flips.*

Proof. Compare the six values in Definition 9462.1380. □

Definition 9462.1382 (One-step retention)**.** *For $S \subseteq Q_6$, define*

$$\rho(S) = \frac{2|E(S)|}{6|S|},$$

where $E(S)$ is the set of Hamming edges internal to S.

Theorem 9462.1383 (Closure-chain retention)**.** *The closure chain has the following one-step retention data:*

set	*vertices*	*internal edges*	*boundary half-edges*	ρ
$\mathcal{R}$	13	16	46	$16/39 \approx 0.410$
$\overline{\mathcal{R}}$	16	25	46	$25/48 \approx 0.521$
$\text{Med}_1(\mathcal{R})$	19	34	46	$34/57 \approx 0.596$
M	20	38	44	$19/30 \approx 0.633.$

Proof. Each codon has degree 6 in Q_6. Internal edges contribute two internal half-edges, so $\rho(S) = 2|E(S)|/(6|S|)$. Boundary half-edges are $6|S| - 2|E(S)|$. Substitute the enumerated vertex and internal-edge counts for the four sets. □

Corollary 9462.1384 (Median closure as local stabilization)**.** *The retention values increase along the chain*

$$0.410 \to 0.521 \to 0.596 \to 0.633.$$

Thus the inactive anchors and median closure raise the one-bit retention probability.

Proof. This is the monotone comparison in Definition 9462.1383. □

Theorem 9462.1385 (ACG boundary seal)**.** *Adding* ACG *to* $\text{Med}_1(\mathcal{R})$ *changes the boundary half-edge count from* 46 *to* 44*.*

Proof. If a vertex p is added to a set S and has k neighbors in S, the boundary half-edge count changes by

$$\Delta\partial = 6 - 2k.$$

For $S = \mathrm{Med}_1(\mathcal{R})$ and $p = \mathrm{ACG}$, the neighbors already in S are

$$\mathrm{ACA}, \qquad \mathrm{AGG}, \qquad \mathrm{AUG}, \qquad \mathrm{UCG}.$$

Thus $k = 4$ and $\Delta\partial = 6 - 8 = -2$. □

Corollary 9462.1386 (Three roles of ACG)**.** *The codon* ACG *has three structural roles: it is the top-dimensional sealing corner, the symmetry-restoring orbit-completion point, and the boundary-reducing dynamical seal.*

Proof. The top-dimensional role is Definition 9462.1333. The orbit-completion role is Definition 9462.1359. The boundary role is Definition 9462.1385. □

Theorem 9462.1387 (Quotient leakage dynamics)**.** *The median closure is a stabilizing envelope for one-bit Hamming dynamics: its orbit quotient is a six-state path with outside leakage, its CUN tail is the highest-leakage region, and the closure chain strictly raises one-step retention from* $16/39$ *to* $19/30$*.*

Proof. The quotient transition matrix is Definition 9462.1376; the tail comparison is Definition 9462.1379; the retention chain is Definition 9462.1383. □

Concretized in §5822: The quotient leakage transition matrix is realized as a finite NameCert packet for six orbit states, outside-probability rows, and expected-exit ledgers.

Definition 9462.1388 (Killed bit-flip walk)**.** *Let $S \subseteq Q_6$. The killed bit-flip walk on S starts at a codon $x \in S$, chooses one of the six bit coordinates uniformly, and flips that coordinate. If the resulting codon remains in S, the walk continues. If the resulting codon lies outside S, the walk is absorbed. The transient matrix is*

$$Q_S(x, y) = \begin{cases} 1/6, & x \sim y \text{ and } y \in S, \\ 0, & \text{otherwise.} \end{cases}$$

Its spectral radius is denoted

$$\lambda(S) = \rho(Q_S).$$

Theorem 9462.1389 (Spectral stabilization along the closure chain)**.** *For the closure chain*

$$\mathcal{R} \subset \overline{\mathcal{R}} \subset \mathrm{Med}_1(\mathcal{R}) \subset M,$$

the killed bit-flip spectral data are

set	*vertices*	*internal edges*	ρ_1	λ	$1/(1-\lambda)$
$\mathcal{R}$	13	16	$16/39 \approx 0.410$	0.4759	1.908
$\overline{\mathcal{R}}$	16	25	$25/48 \approx 0.521$	0.5733	2.344
$\mathrm{Med}_1(\mathcal{R})$	19	34	$34/57 \approx 0.596$	0.6422	2.795
M	20	38	$19/30 \approx 0.633$	0.6752	3.079.

Concretized in §2270: The killed bit-flip closure-chain spectral-radius and survival-potential ledger is realized as a finite NameCert packet.

Proof. For each set S, enumerate the Hamming edges internal to S and form the transient matrix of Definition 9462.1388. The one-step retention ρ_1 is the retention quantity of Definition 9462.1382. Since Q_S is a symmetric nonnegative matrix, its spectral radius is the Perron root. Direct finite enumeration gives the displayed values. □

Corollary 9462.1390 (Spectral meaning of median closure)**.** *The median closure is a spectral stabilization envelope for the reassignment support: along the closure chain, the killed-walk spectral radius rises from approximately* 0.476 *to approximately* 0.675*.*

Proof. This is the monotone comparison of the four spectral radii in Definition 9462.1389. □

Definition 9462.1391 (Quasi-stationary distribution and survival potential)**.** *For $M = \mathrm{Med}(\mathcal{R})$, let λ_M be the Perron root of Q_M. The quasi-stationary distribution is the normalized left Perron vector*

$$\pi Q_M = \lambda_M \pi, \qquad \sum_{x \in M} \pi(x) = 1.$$

The survival potential is the right Perron vector

$$Q_M h = \lambda_M h,$$

normalized so that the largest orbit-average value is 1*.*

Theorem 9462.1392 (Quasi-stationary mass on the quotient path)**.** *Aggregated over the six quotient states of Definition 9462.1364, the quasi-stationary distribution of the killed walk on M is*

state	*quasi-stationary mass*
AGR	0.1117
Anchor_6	0.3407
$\mathrm{StopStart}_6$	0.3524
UUR	0.1313
CUR	0.0482
CUY	0.0158.

Hence

$$\pi(\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6) = 0.6931.$$

Proof. Compute the Perron vector of Q_M and sum its entries over the six orbits of $\mathrm{Stab}(M)$. The two central orbit states Anchor_6 and $\mathrm{StopStart}_6$ have combined mass $0.3407 + 0.3524 = 0.6931$ after normalization. □

Corollary 9462.1393 (Quasi-stationary core)**.** *Conditioned on long survival inside M, the walk is concentrated on the anchor and stop-start core*

$$\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6.$$

Proof. The combined quasi-stationary mass of these two states is 0.6931 by Definition 9462.1392. □

Theorem 9462.1394 (Survival potential on the quotient path)**.** *The orbit-averaged survival potential is*

state	*survival potential*
AGR	0.8506
Anchor_6	0.8652
StopStart_6	0.8948
UUR	1.0000
CUR	0.3671
CUY	0.1203.

Proof. Take the Perron vector h satisfying $Q_M h = \lambda_M h$, average it on each quotient orbit, and divide by the largest orbit average. The largest average occurs on UUR. □

Corollary 9462.1395 (UUR as survival bridge)**.** *The state* UUR *has the highest survival potential, even though the quasi-stationary mass is concentrated mainly on* $\text{Anchor}_6 \cup \text{StopStart}_6$*.*

Proof. Compare Definition 9462.1392 with Definition 9462.1394. □

Theorem 9462.1396 (Quasi-stationary escape source)**.** *Conditioned on the next step escaping from M, the pre-escape quotient state has distribution*

state	*conditional probability*
AGR	0.1146
Anchor_6	0.3497
StopStart_6	0.3617
UUR	0.0674
CUR	0.0742
CUY	0.0324.

Thus

$$P(\text{Anchor}_6 \cup \text{StopStart}_6 \mid \textit{escape}) = 0.7114.$$

Proof. In quasi-stationarity the probability of escape in one step is $1 - \lambda_M$. The conditional contribution of a quotient state S is

$$\frac{\pi(S)P(S \to \text{outside})}{1 - \lambda_M}.$$

Use the quasi-stationary masses from Definition 9462.1392 and the leakage probabilities from Definition 9462.1378. □

Corollary 9462.1397 (Escape mass versus pointwise leakage)**.** *The CUN tail has high pointwise leakage, but the dominant quasi-stationary escape contribution comes from the anchor and stop-start core.*

Proof. The tail leakage comparison is Definition 9462.1379. The conditional escape source distribution is Definition 9462.1396. □

Theorem 9462.1398 (Outside target classes)**.** *Starting from quasi-stationarity and conditioning on a one-step escape, the outside target codons aggregate by standard-code class as*

class	*probability*
Ser	0.1176
Pro	0.0931
Val	0.0931
Gln	0.0850
Phe	0.0755
Ile	0.0603
Tyr	0.0603
Ala	0.0583
Cys	0.0583
Glu	0.0583
Asn	0.0583
Arg	0.0583
Thr	0.0583
Gly	0.0573
His	0.0081.

Proof. For each $x \in M$ and each one-bit outside neighbor $y \notin M$, add the weight

$$\frac{\pi(x)}{6(1-\lambda_M)}$$

to the standard-code class of y. Summing over all outside neighbors gives the displayed distribution. □

Corollary 9462.1399 (Sense-class escape boundary)**.** *The outside escape distribution is spread over ordinary sense classes; it is not concentrated on stop-like targets.*

Proof. The target table in Definition 9462.1398 contains no dominant stop-class entry. □

Theorem 9462.1400 (ACG as spectral sealing corner)**.** *Adding* ACG *to* $\mathrm{Med}_1(\mathcal{R})$ *raises the killed-walk spectral radius from* 0.6422 *to* 0.6752*, raises one-step retention from approximately* 0.596 *to approximately* 0.633*, and lowers the boundary half-edge count from* 46 *to* 44*.*

Proof. The spectral radii are the last two rows of Definition 9462.1389. The retention and boundary values are the last two rows of Definition 9462.1383. □

Corollary 9462.1401 (Four roles of ACG)**.** *The codon* ACG *is simultaneously a strict second-order median point, a top-dimensional sealing corner, an orbit-completion point for* $S_3 \times C_2$ *symmetry, and a spectral stabilization point.*

Proof. The median-depth role is Definition 9462.1334. The top-dimensional role is Definition 9462.1333. The orbit-completion role is Definition 9462.1359. The spectral role is Definition 9462.1400. □

Theorem 9462.1402 (Spectral stabilization theorem)**.** *The median closure* $M = WNR \cup CUN$ *is the spectral stabilization envelope of the active reassignment support under killed one-bit Hamming dynamics: the closure chain increases the Perron root, the quasi-stationary mass concentrates on the anchor and stop-start core, the strongest survival potential lies at the* UUR *bridge, and* ACG *completes the spectral seal.*

Proof. Combine Definition 9462.1389, Definition 9462.1392, Definition 9462.1394, and Definition 9462.1400. □

Definition 9462.1403 (Fixed-budget spectral expansion)**.** *For $S \subseteq Q_6$ and $m \geq 1$, the fixed-budget spectral expansion problem asks for subsets*

$$B \subseteq Q_6 \setminus S, \qquad |B| = m,$$

maximizing

$$\lambda(S \cup B),$$

where λ is the killed bit-flip spectral radius of Definition 9462.1388.

Theorem 9462.1404 (Fixed-budget spectral frontier from the active support)**.** *For $m = 1, 2, 3, 4$, exhaustive finite enumeration from $\mathcal{R}$ gives the following spectral optima:*

m	*optimal adjoined set*	$\lambda(\mathcal{R} \cup B)$	*multiplicity*
1	$\{\mathrm{ACA}\}$	0.5237258	1
2	$\{\mathrm{ACA}, \mathrm{UUG}\}$	0.5465038	1
3	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$	0.5733331	1
4	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UUG}\}$	0.5970778	2
4	$\{\mathrm{ACA}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$	0.5970778	2.

Proof. For each m, enumerate all $\binom{51}{m}$ subsets of the inactive complement of $\mathcal{R}$ and compute the Perron root of the killed bit-flip transition matrix on $\mathcal{R} \cup B$. The candidate counts are 51, 1275, 20825, and 249900 for $m = 1, 2, 3, 4$, respectively. The displayed rows are the maximizers. □

Corollary 9462.1405 (Fixed-budget non-nesting)**.** *The fixed-budget spectral optima from $\mathcal{R}$ do not form a nested chain. In particular,*

$$\{\mathrm{ACA}, \mathrm{UUG}\} \not\subseteq \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}.$$

Proof. The two displayed sets are the unique optima for budgets $m = 2$ and $m = 3$ in Definition 9462.1404. □

Corollary 9462.1406 (Spectral budget frustration)**.** *The codon* UUG *is selected by the optimal two-point expansion from $\mathcal{R}$, but is absent from the optimal three-point anchor block. Thus small-budget spectral optimization can select a point that is not part of the next optimal closure block.*

Proof. This is the comparison between the $m = 2$ and $m = 3$ rows of Definition 9462.1404. □

Definition 9462.1407 (Layered block spectral optimality)**.** *A closure chain is layered block spectral-optimal when each prescribed layer size is a spectral maximizer from the previous layer, without requiring the fixed-budget optima from the initial set to be nested.*

Theorem 9462.1408 (Anchor triple optimality)**.** *Among all 3-point subsets $B \subseteq Q_6 \setminus \mathcal{R}$, the unique maximizer of $\lambda(\mathcal{R} \cup B)$ is*

$$B = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}.$$

The search has $\binom{51}{3} = 20825$ candidates, and

$$\lambda(\mathcal{R}) = 0.475879, \qquad \lambda(\mathcal{R} \cup B) = 0.573333.$$

Proof. This is the $m = 3$ row of Definition 9462.1404. □

Theorem 9462.1409 (Anchor-layer spectral frontier)**.** *From*

$$\overline{\mathcal{R}} = \mathcal{R} \cup \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\},$$

the first three fixed-budget spectral optima are

m	*optimal adjoined set*	$\lambda(\overline{\mathcal{R}} \cup B)$	*multiplicity*
1	$\{\mathrm{UUG}\}$	0.5929439	1
2	$\{\mathrm{AUG}, \mathrm{UUG}\}$	0.6183868	2
2	$\{\mathrm{UCG}, \mathrm{UUG}\}$	0.6183868	2
3	$\{\mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG}\}$	0.6421817	1.

Proof. Enumerate all m-point subsets of $Q_6 \setminus \overline{\mathcal{R}}$ for $m = 1, 2, 3$ and compute the killed-walk Perron root for each expanded set. The displayed rows are the maximizers. □

Theorem 9462.1410 (First-median triple optimality)**.** *Among all* 3*-point subsets* $B \subseteq Q_6 \setminus \overline{\mathcal{R}}$*, the unique maximizer of* $\lambda(\overline{\mathcal{R}} \cup B)$ *is*

$$B = \{\mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG}\}.$$

Equivalently,

$$\mathrm{Med}_1(\mathcal{R}) = \overline{\mathcal{R}} \cup B$$

is the unique optimal three-point spectral stabilization of the anchor closure.

Proof. This is the $m = 3$ row of Definition 9462.1409, together with Definition 9462.1329. □

Theorem 9462.1411 (ACG single-point optimality)**.** *Among all single codons* $p \in Q_6 \setminus \mathrm{Med}_1(\mathcal{R})$*, the unique maximizer of*

$$\lambda(\mathrm{Med}_1(\mathcal{R}) \cup \{p\})$$

is

$$p = \mathrm{ACG}.$$

There are 45 *candidates, and*

$$\lambda(\mathrm{Med}_1(\mathcal{R})) = 0.642182, \qquad \lambda(\mathrm{Med}_1(\mathcal{R}) \cup \{\mathrm{ACG}\}) = 0.675248.$$

Proof. Enumerate the 45 codons outside $\mathrm{Med}_1(\mathcal{R})$ and compute the killed-walk Perron root after adjoining each one. The unique maximum is attained at ACG. □

Theorem 9462.1412 (Layered block spectral closure chain)**.** *The chain*

$$\mathcal{R} \xrightarrow{\text{best } +3} \overline{\mathcal{R}} \xrightarrow{\text{best } +3} \mathrm{Med}_1(\mathcal{R}) \xrightarrow{\text{best } +1} M$$

is layered block spectral-optimal. The adjoined blocks are

$$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}, \qquad \{\mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG}\}, \qquad \{\mathrm{ACG}\}.$$

The corresponding spectral radii are

set	*adjoined codons*	*vertices*	λ
$\mathcal{R}$	$\varnothing$	13	0.475879
$\overline{\mathcal{R}}$	AAG, ACA, UGG	16	0.573333
$\mathrm{Med}_1(\mathcal{R})$	AUG, UCG, UUG	19	0.642182
M	ACG	20	0.675248.

Proof. The three steps are respectively Definition 9462.1408, Definition 9462.1410, and Definition 9462.1411. □

Definition 9462.1413 (Spectral gain)**.** *For $S \subseteq S' \subseteq Q_6$, define*

$$\Delta\lambda(S, S') = \lambda(S') - \lambda(S).$$

Corollary 9462.1414 (Diminishing block gain)**.** *The layered block gains are*

$$0.097454, \qquad 0.068849, \qquad 0.033066,$$

and therefore strictly decrease.

Proof. Subtract consecutive spectral radii in Definition 9462.1412. □

Theorem 9462.1415 (Budget-four spectral branching)**.** *At budget $m = 4$ from $\mathcal{R}$, the two optimal adjoined sets are*

$$B_4^{(1)} = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UUG}\},$$

and

$$B_4^{(2)} = \{\mathrm{ACA}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

They have the same spectral radius 0.5970778 *and are exchanged by the support symmetry $\tau : s_1 \leftrightarrow f_2$.*

Proof. The two optimal sets are the $m = 4$ rows of Definition 9462.1404. Under τ, AAG is exchanged with UGG, AUG is exchanged with UCG, while ACA and UUG are fixed. □

Corollary 9462.1416 (Anchor layer is a block stabilizer)**.** *The anchor triple* $\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$ *is a joint spectral optimizer, not the result of nested fixed-budget optimization from $\mathcal{R}$.*

Proof. The non-nesting statement is Definition 9462.1405. The triple optimality is Definition 9462.1408. □

Theorem 9462.1417 (τ-invariant inactive frontier)**.** *If the adjoined inactive set $B \subseteq Q_6 \setminus \mathcal{R}$ is required to satisfy $\tau(B) = B$, the optimal symmetric frontier includes the following representatives:*

$\lvert B\rvert$	*optimal τ-invariant adjoined set*
1	$\{\mathrm{ACA}\}$
2	$\{\mathrm{ACA}, \mathrm{UUG}\}$
3	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$
4	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}, \mathrm{UUG}\}$
5	$\{\mathrm{AAG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$
6	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$
6	$\{\mathrm{AAG}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$
7	$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$.

Proof. Take the τ-orbits in $Q_6 \setminus \mathcal{R}$ and enumerate orbit unions of the prescribed cardinality. For each orbit union, compute the killed-walk Perron root after adjoining it to $\mathcal{R}$. The displayed sets are the maximizing orbit unions. □

Corollary 9462.1418 (Symmetric completion recovers the inactive median layer)**.** *At cardinality* 7*, the optimal τ-invariant inactive completion is*

$$\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{ACG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\} = M \setminus \mathcal{R}.$$

Proof. The $|B| = 7$ row of Definition 9462.1417 is exactly the inactive layer of Definition 9462.1368. □

Theorem 9462.1419 (Block-optimal budget-frustrated spectral closure)**.** *The median closure chain is not a nested fixed-budget optimum from $\mathcal{R}$. It is instead a layered block spectral-optimal chain, with a symmetric seven-point inactive completion equal to $M \setminus \mathcal{R}$.*

Proof. Non-nesting is Definition 9462.1405. The layered block optimality is Definition 9462.1412. The symmetric completion statement is Definition 9462.1418. □

Concretized in §3753: finite trigger-spectrum closure packet

Definition 9462.1420 (Directional saturation)**.** *For $S \subseteq Q_6$ and a bit coordinate i, let $x^{(i)}$ denote the codon obtained from x by flipping the ith bit. The directional saturation of S in direction i is*

$$\mathrm{Sat}_i(S) = S \cup \{x^{(i)} : x \in S\}.$$

For the coordinate order

$$(s_1, f_1, s_2, f_2, s_3, f_3),$$

the direction f_3 flips the family bit of the third codon position, so it exchanges $\mathrm{U} \leftrightarrow \mathrm{C}$ *and* $\mathrm{A} \leftrightarrow \mathrm{G}$ *at that position.*

Theorem 9462.1421 (Unique non-leaking saturation direction)**.** *Among the six one-bit saturation directions, f_3 is the unique direction for which the saturation of the active support remains inside the median closure:*

$$\mathrm{Sat}_{f_3}(\mathcal{R}) \subseteq M.$$

The six saturation rows are

direction	*added codons*	*added set*	$\subseteq M$
s_1	7	AAG, ACA, GUA, GUC, GUG, GUU, UGG	*no*
f_1	11	CAA, CAG, CCA, CGA, GAA, GGA, GGG, GUA, UUC, UUG, UUU	*no*
s_2	7	ACA, ACG, CAA, CAC, CAG, CAU, UUG	*no*
f_2	7	AAG, ACA, CCA, CCC, CCG, CCU, UGG	*no*
s_3	9	AAU, AGC, AGU, AUU, UAC, UAU, UCU, UGU, UUU	*no*
f_3	5	AAG, AUG, UCG, UGG, UUG	*yes.*

Proof. For each coordinate i, compute $\mathrm{Sat}_i(\mathcal{R})$ by flipping the ith bit of each codon in $\mathcal{R}$ and compare the result with $M = \mathrm{Med}(\mathcal{R})$. The displayed finite enumeration shows that only the f_3 row has no point outside M. □

Theorem 9462.1422 (f_3 saturation misses exactly the Thr pair)**.** *The f_3 saturation of the active support is*

$$\mathrm{Sat}_{f_3}(\mathcal{R}) = M \setminus \{\mathrm{ACA}, \mathrm{ACG}\}.$$

Equivalently,

$$\mathrm{Sat}_{f_3}(\mathcal{R}) \setminus \mathcal{R} = \{\mathrm{AAG}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}.$$

Proof. The five added codons are the f_3 partners of active codons:

$$\mathrm{AAA} \leftrightarrow \mathrm{AAG}, \qquad \mathrm{AUA} \leftrightarrow \mathrm{AUG},$$
$$\mathrm{UCA} \leftrightarrow \mathrm{UCG}, \qquad \mathrm{UGA} \leftrightarrow \mathrm{UGG}, \qquad \mathrm{UUA} \leftrightarrow \mathrm{UUG}.$$

Comparing with the median closure $M = \mathrm{WNR} \cup \mathrm{CUN}$ leaves exactly ACA and ACG outside the saturation. □

Theorem 9462.1423 (Wobble saturation presentation of median closure)**.** *The full median closure is obtained by adjoining either member of the missing Thr pair and saturating in the f_3 direction:*

$$M = \mathrm{Sat}_{f_3}(\mathcal{R} \cup \{\mathrm{ACA}\}) = \mathrm{Sat}_{f_3}(\mathcal{R} \cup \{\mathrm{ACG}\}).$$

Moreover, ACA *and* ACG *are the only codons outside $\mathcal{R}$ with this property.*

Proof. By Definition 9462.1422, $\mathrm{Sat}_{f_3}(\mathcal{R}) = M \setminus \{\mathrm{ACA}, \mathrm{ACG}\}$. Since $\mathrm{ACA}^{(f_3)} = \mathrm{ACG}$ and $\mathrm{ACG}^{(f_3)} = \mathrm{ACA}$, adjoining either one supplies both missing vertices after f_3 saturation. Exhaustive comparison over the inactive complement of $\mathcal{R}$ shows that no other single codon has saturation equal to M. □

Definition 9462.1424 (Directional boundary)**.** *For $S \subseteq Q_6$, the directional boundary in bit coordinate i is*

$$\partial_i(S) = |\{x \in S : x^{(i)} \notin S\}|.$$

The normalized Boolean influence is

$$I_i(S) = \partial_i(S)/32.$$

Theorem 9462.1425 (Directional boundary profile of the closure chain)**.** *For the closure chain*

$$\mathcal{R} \subset \overline{\mathcal{R}} \subset \mathrm{Med}_1(\mathcal{R}) \subset M,$$

the directional boundary counts in the coordinate order $(s_1, f_1, s_2, f_2, s_3, f_3)$ are

set	s_1	f_1	s_2	f_2	s_3	f_3	*total*
$\mathcal{R}$	7	11	7	7	9	5	46
$\overline{\mathcal{R}}$	4	14	8	4	12	4	46
$\mathrm{Med}_1(\mathcal{R})$	5	15	5	5	15	1	46
M	4	16	4	4	16	0	44.

Proof. For each set S in the displayed chain and each bit coordinate i, count the codons $x \in S$ for which $x^{(i)} \notin S$. This direct enumeration gives the table. □

Corollary 9462.1426 (Final f_3 boundary seal)**.** *The unique f_3 boundary edge at the first median layer is*

$$\mathrm{ACA} \leftrightarrow \mathrm{ACG}.$$

Adjoining ACG *changes ∂_{f_3} from* 1 *to* 0.

Proof. The row for $\mathrm{Med}_1(\mathcal{R})$ in Definition 9462.1425 has $\partial_{f_3} = 1$, and finite enumeration identifies the single edge as $\mathrm{ACA} \leftrightarrow \mathrm{ACG}$. The row for M has $\partial_{f_3} = 0$. □

Theorem 9462.1427 (f_3-silent Boolean form)**.** *In bit variables*

$$(s_1, f_1, s_2, f_2, s_3, f_3),$$

the median closure has the Boolean description

$$M = (\neg f_1 \wedge s_3) \vee (f_1 \wedge \neg s_1 \wedge \neg s_2 \wedge \neg f_2).$$

In particular, membership in M is independent of f_3.

Proof. The identity $M = \mathrm{WNR} \cup \mathrm{CUN}$ gives $\mathrm{WNR} = \{\neg f_1 \wedge s_3\}$, because the first codon position is U or A and the third position is A or G. It also gives

$$\mathrm{CUN} = \{f_1 \wedge \neg s_1 \wedge \neg s_2 \wedge \neg f_2\},$$

with the third position free. The displayed formula is their union and contains no occurrence of f_3. □

Corollary 9462.1428 (Gate and silent axes)**.** *For M, the directional boundary vector is*

$$(4, 16, 4, 4, 16, 0).$$

Thus f_1 and s_3 are the two main gate axes, s_1, s_2, and f_2 are low-boundary axes, and f_3 is a silent axis.

Proof. This is the M row of Definition 9462.1425, together with the absence of f_3 from Definition 9462.1427. □

Concretized in §3941: Finite NameCert seal for the unique f3 directional saturation route.

Theorem 9462.1429 (Wobble-saturation closure)**.** *The median closure of the reassigned support is the third-position family-bit saturation envelope*

$$M = \mathrm{Sat}_{f_3}(\mathcal{R} \cup \{\mathrm{ACA}\}) = \mathrm{Sat}_{f_3}(\mathcal{R} \cup \{\mathrm{ACG}\}),$$

and f_3 is the unique active-support saturation direction that does not leave M. The codon ACG *is simultaneously the median sealing corner, the symmetry-restoring corner, the spectral sealing corner, and the final f_3 saturation seal.*

Proof. The saturation identity is Definition 9462.1423. The uniqueness of the non-leaking direction is Definition 9462.1421. The median sealing role is Definition 9462.1329; the symmetry role is Definition 9462.1362; the spectral role is Definition 9462.1411; and the final f_3 boundary seal is Definition 9462.1426. □

Definition 9462.1430 (f_3 quotient)**.** *Let*

$$\pi : Q_6 \to Q_5$$

delete the third-position family bit:

$$\pi(s_1, f_1, s_2, f_2, s_3, f_3) = (s_1, f_1, s_2, f_2, s_3).$$

Thus the third codon position retains only its shape class. A quotient symbol such as UUR *denotes the pair*

$$\mathrm{UUR} = \{\mathrm{UUA}, \mathrm{UUG}\},$$

and ACR *denotes*

$$\mathrm{ACR} = \{\mathrm{ACA}, \mathrm{ACG}\}.$$

Theorem 9462.1431 (Median closure quotient)**.** *For $M = \mathrm{Med}(\mathcal{R})$, the f_3 quotient has ten vertices:*

$$\pi(M) = \{\mathrm{UUR}, \mathrm{UCR}, \mathrm{UAR}, \mathrm{UGR}, \mathrm{AUR}, \mathrm{ACR}, \mathrm{AAR}, \mathrm{AGR}, \mathrm{CUR}, \mathrm{CUY}\}.$$

Equivalently, $\pi(M)$ is the image of $\mathrm{WNR} \cup \mathrm{CUN}$ *after deleting f_3.*

Proof. By Definition 9462.1320, $M = \mathrm{WNR} \cup \mathrm{CUN}$. The image of WNR is the eight-point quotient block with first position in $\{\mathrm{U}, \mathrm{A}\}$, arbitrary second position, and third-position shape R. The image of CUN contributes CUY and CUR. These ten vertices are exactly the displayed set. □

Theorem 9462.1432 (Active quotient as a one-hole closure)**.** *The active reassignment support projects to the quotient median closure with exactly one missing vertex:*

$$\pi(\mathcal{R}) = \pi(M) \setminus \{\mathrm{ACR}\}.$$

Thus

$$\pi(\mathcal{R}) = \{\mathrm{UUR}, \mathrm{UCR}, \mathrm{UAR}, \mathrm{UGR}, \mathrm{AUR}, \mathrm{AAR}, \mathrm{AGR}, \mathrm{CUR}, \mathrm{CUY}\}.$$

Proof. Project each active codon by deleting f_3. The pairs AAA/AAG, AGA/AGG, AUA/AUG, and similarly for the U and CU rows, give all vertices of $\pi(M)$ except ACR. The support contains neither ACA nor ACG, so ACR is absent. □

Corollary 9462.1433 (The Thr pair as a quotient hole)**.** *The two codons* ACA *and* ACG *are the two lifts of the single quotient vertex* ACR*. Hence the first-layer point* ACA *and the sealing point* ACG *in Q_6 become one missing vertex after quotienting by f_3.*

Proof. Both codons have the same first five coordinates and differ only in f_3. □

Theorem 9462.1434 (One-step quotient median closure)**.** *In Q_5,*

$$\mathrm{Med}_{Q_5}(\pi(\mathcal{R})) = \pi(M),$$

and the unique added quotient vertex is ACR*. One median witness is*

$$\mathrm{ACR} = m(\mathrm{AGR}, \mathrm{AUR}, \mathrm{UCR}).$$

Proof. By finite enumeration of coordinatewise medians in Q_5, the first median pass on $\pi(\mathcal{R})$ adds exactly ACR and then stabilizes. The displayed witness verifies that ACR is forced by the quotient support. □

Theorem 9462.1435 (Quotient cubical skeleton)**.** *The cubical complex $K(\pi(M))$ has f-vector*

$$(f_0, f_1, f_2, f_3) = (10, 14, 6, 1).$$

Its maximal cells are one Q_3 core

$$\{\mathrm{UUR}, \mathrm{UCR}, \mathrm{UAR}, \mathrm{UGR}, \mathrm{AUR}, \mathrm{ACR}, \mathrm{AAR}, \mathrm{AGR}\},$$

and two edges

$$\mathrm{UUR} - \mathrm{CUR}, \qquad \mathrm{CUY} - \mathrm{CUR}.$$

Proof. Enumerate all subcubes of Q_5 contained in $\pi(M)$. The unique maximal three-cell is the quotient of WNR, and the quotient of the CUN tail contributes CUY − CUR. The bridge UUR − CUR attaches the tail to the core. The resulting cell counts are $(10, 14, 6, 1)$. □

Theorem 9462.1436 (Punctured quotient skeleton)**.** *The active quotient $K(\pi(\mathcal{R}))$ has f-vector*

$$(f_0, f_1, f_2) = (9, 11, 3).$$

It is the punctured Q_3 core

$$\pi(\mathrm{WNR}) \setminus \{\mathrm{ACR}\}$$

together with the same two-edge tail

$$\mathrm{UUR} - \mathrm{CUR} - \mathrm{CUY}.$$

Proof. Deleting one vertex from a Q_3 leaves seven vertices, nine edges, and three squares. The quotient tail contributes two vertices and two edges, giving $(9, 11, 3)$. □

Theorem 9462.1437 (Wobble product decomposition)**.** *The median closure complex is the product of the quotient complex with the f_3 edge:*

$$K(M) = K(\pi(M)) \times Q_1^{f_3}.$$

Consequently the f-vector of $K(M)$ is recovered from $f(K(\pi(M))) = (10, 14, 6, 1)$ by

$$f_k(K \times Q_1) = 2f_k(K) + f_{k-1}(K),$$

giving

$$f(K(M)) = (20, 38, 26, 8, 1).$$

Proof. By Definition 9462.1429, M is saturated in the f_3 direction, so every quotient vertex of $\pi(M)$ has both f_3 lifts in M. This is exactly the cubical product with the f_3 edge. Applying the displayed product formula gives the stated f-vector, matching Definition 9462.1332. □

Theorem 9462.1438 (Quotient spectral reconstruction)**.** *Let $G_{\pi(M)}$ be the Hamming graph on the quotient skeleton. Its largest adjacency eigenvalue is*

$$\mu_{\max}(G_{\pi(M)}) = 3.051487585\ldots.$$

Since $G_M = G_{\pi(M)} \square K_2$, the killed bit-flip spectral radius of M is

$$\lambda(M) = \frac{\mu_{\max}(G_{\pi(M)}) + 1}{6} = 0.675247931\ldots.$$

Proof. The adjacency spectrum of a Cartesian product satisfies

$$\mathrm{Spec}(G \square H) = \{\mu + \nu : \mu \in \mathrm{Spec}(G), \nu \in \mathrm{Spec}(H)\}.$$

For $H = K_2$ the eigenvalues are 1 and -1. The largest adjacency eigenvalue of G_M is therefore $\mu_{\max}(G_{\pi(M)}) + 1$, and the killed bit-flip matrix is $A_M/6$. □

Theorem 9462.1439 (Wobble-pair occupancy)**.** *The active support occupies quotient fibers as follows:*

quotient vertex	*fiber*	*active occupancy*
UUR	UUA/UUG	1
UCR	UCA/UCG	1
UAR	UAA/UAG	2
UGR	UGA/UGG	1
AUR	AUA/AUG	1
ACR	ACA/ACG	0
AAR	AAA/AAG	1
AGR	AGA/AGG	2
CUR	CUA/CUG	2
CUY	CUC/CUU	2.

Thus the full active fibers are

$$\mathrm{UAR}, \mathrm{AGR}, \mathrm{CUR}, \mathrm{CUY},$$

the half-active fibers are

$$\mathrm{UUR}, \mathrm{UCR}, \mathrm{UGR}, \mathrm{AUR}, \mathrm{AAR},$$

and the empty fiber is ACR*.*

Proof. For each quotient vertex, count the two f_3 lifts that lie in $\mathcal{R}$. The displayed table is the resulting finite count. □

Theorem 9462.1440 (Wobble quotient closure)**.** *The full median closure is recovered by taking median closure after the f_3 quotient and then lifting all f_3 fibers:*

$$M = \pi^{-1}(\mathrm{Med}_{Q_5}(\pi(\mathcal{R}))) \,.$$

Equivalently,

$$M = (Q_3 \text{ core plus a two-edge tail}) \times Q_1^{f_3}.$$

Proof. By Definition 9462.1434, $\mathrm{Med}_{Q_5}(\pi(\mathcal{R})) = \pi(M)$. By Definition 9462.1437, $M = \pi^{-1}(\pi(M))$. Combining the two identities gives the displayed formula. □

Concretized in §2317: finite quotient-trigger rows for wobble spectral geometry

Definition 9462.1441 (Radial quotient states)**.** *In the f_3 quotient $P = \pi(M)$ of Definition 9462.1430, the Q_3 core has coordinates s_1, s_2, and f_2. The S_3 action permuting these three coordinates gives the six radial states*

state	*quotient codons*	*lift in M*
X_3	AGR	AGA, AGG
X_2	AAR, ACR, UGR	AAA/AAG, ACA/ACG, UGA/UGG
X_1	AUR, UAR, UCR	AUA/AUG, UAA/UAG, UCA/UCG
X_0	UUR	UUA, UUG
Y_R	CUR	CUA, CUG
Y_Y	CUY	CUC, CUU.

Theorem 9462.1442 (Radial adjacency matrix)**.** *For the Hamming graph of P, the equitable radial quotient in the state order*

$$X_3, X_2, X_1, X_0, Y_R, Y_Y$$

has adjacency matrix

$$A_{\mathrm{rad}} = \begin{pmatrix} 0 & 3 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Equivalently, the Perron equations are

$$\mu h_3 = 3h_2, \qquad \mu h_2 = h_3 + 2h_1,$$
$$\mu h_1 = 2h_2 + h_0, \qquad \mu h_0 = 3h_1 + h_R,$$
$$\mu h_R = h_0 + h_Y, \qquad \mu h_Y = h_R.$$

Proof. Inside the Q_3 core, a vertex of weight k has k neighbors in weight $k-1$ and $3-k$ neighbors in weight $k+1$. The tail contributes the additional edges $X_0 - Y_R$ and $Y_R - Y_Y$. These local neighbor counts give the displayed matrix. □

Theorem 9462.1443 (Radial spectral equation)**.** *The characteristic polynomial of A_{rad} is*

$$\mu^6 - 12\mu^4 + 26\mu^2 - 9.$$

Equivalently, with $r = \mu^2$,

$$r^3 - 12r^2 + 26r - 9 = 0.$$

The largest root satisfies

$$r_{\max} = 9.311576483\ldots, \qquad \mu_{\max} = 3.051487585\ldots.$$

Since $M = P \times Q_1^{f_3}$, the killed bit-flip spectral radius is

$$\lambda(M) = \frac{\mu_{\max} + 1}{6} = 0.675247931\ldots.$$

Proof. Compute $\det(\mu I - A_{\mathrm{rad}})$ from the matrix of Definition 9462.1442. The polynomial is even, so substituting $r = \mu^2$ gives the cubic equation. The product decomposition of Definition 9462.1437 adds the K_2 eigenvalue 1 to the largest quotient eigenvalue, and the killed walk divides adjacency eigenvalues by 6. □

Corollary 9462.1444 (Analytic compression of the killed-walk spectrum)**.** *The killed-walk spectral radius of the 20-codon median closure is determined by the largest root of the cubic*

$$r^3 - 12r^2 + 26r - 9 = 0.$$

Thus the spectral computation compresses as

$$20 \textit{ codons} \longrightarrow 10 \textit{ quotient vertices} \longrightarrow 6 \textit{ radial states} \longrightarrow 1 \textit{ cubic equation.}$$

Proof. This is the combination of Definition 9462.1431, Definition 9462.1442, and Definition 9462.1443. □

Theorem 9462.1445 (Radial quasi-stationary data)**.** *Normalize the positive Perron vector of* A_{rad} *so that* X_0 *has survival potential* 1*. The radial survival potentials are*

state	*survival potential*
$X_3 = \mathrm{AGR}$	0.8506
$X_2 = \mathrm{Anchor}_6$	0.8652
$X_1 = \mathrm{StopStart}_6$	0.8948
$X_0 = \mathrm{UUR}$	1.0000
$Y_R = \mathrm{CUR}$	0.3671
$Y_Y = \mathrm{CUY}$	0.1203.

After weighting by the lift sizes in M*, the quasi-stationary masses are*

state	*lift size*	*quasi-stationary mass*
$X_3 = \mathrm{AGR}$	2	0.1117
$X_2 = \mathrm{Anchor}_6$	6	0.3407
$X_1 = \mathrm{StopStart}_6$	6	0.3524
$X_0 = \mathrm{UUR}$	2	0.1313
$Y_R = \mathrm{CUR}$	2	0.0482
$Y_Y = \mathrm{CUY}$	2	0.0158.

Proof. Solve $A_{\mathrm{rad}}h = \mu_{\max}h$ for the positive Perron vector. The displayed survival potentials are the entries normalized by the X_0 entry. The quasi-stationary mass of a radial state is its lift size times the Perron entry, divided by the corresponding total over all six states. □

Corollary 9462.1446 (Radial mass core and survival bridge)**.** *The radial quasi-stationary mass on* $X_2 \cup X_1$ *is*

$$0.3407 + 0.3524 = 0.6931.$$

Thus $\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6$ *is the mass core, whereas* $X_0 = \mathrm{UUR}$ *is the highest-potential survival bridge.*

Proof. The mass statement and the maximum survival potential are read directly from Definition 9462.1445. □

Theorem 9462.1447 (f_3 self-loop contribution)**.** *In the six-state killed-walk quotient on* M*, every radial state has self-retention probability*

$$1/6$$

coming from the f_3 *edge. Without the wobble edge the normalized spectral value would be*

$$\mu_{\max}/6 = 0.508581264\ldots,$$

whereas the saturated median closure has

$$(\mu_{\max} + 1)/6 = 0.675247931\ldots.$$

The wobble edge contributes the additive term 1/6*.*

Proof. Every codon of M has its f_3 neighbor still in M by Definition 9462.1429. In the quotient state model this direction is a self-loop. The product spectrum adds the K_2 eigenvalue 1, and division by the six available bit-flip directions gives the contribution 1/6. □

Theorem 9462.1448 (Radial wobble spectrum)**.** *The killed bit-flip spectrum, quasi-stationary mass, and survival potential of M are determined by the six-state radial matrix A_{rad} together with the f_3 wobble edge. In particular,*

$$\lambda(M) = \frac{1+\sqrt{r_{\max}}}{6},$$

where $r_{\max}$ is the largest root of

$$r^3 - 12r^2 + 26r - 9 = 0.$$

Proof. The quotient geometry is $M = P \times Q_1^{f_3}$ by Definition 9462.1437. The S_3 radial quotient of P is Definition 9462.1442. The Perron equation and the product spectral shift are Definition 9462.1443. □

Concretized in §3681: The radial wobble spectrum is realized as finite boundary-tile, radial-shell, spectral-hinge, and trigger-ledger NameCert rows.

Definition 9462.1449 (Core and CUN tail)**.** *Let*

$$C = \mathrm{WNR} \subset M.$$

Then C is the main core of the median closure. The remaining quotient geometry is the two-edge tail

$$\mathrm{UUR} - \mathrm{CUR} - \mathrm{CUY},$$

whose lift is the CUN *contribution to $M = \mathrm{WNR} \cup \mathrm{CUN}$.*

Theorem 9462.1450 (The WNR core is a killed-walk Q_4)**.** *The core $C = \mathrm{WNR}$ is a Q_4 subcube. Hence*

$$|C| = 16, \qquad |E(C)| = 32,$$

and for the killed one-bit walk in Q_6,

$$\rho(C) = \lambda(C) = \frac{2}{3}.$$

Proof. The motif WNR fixes $f_1 = 0$ and $s_3 = 1$, leaving four free bits. Thus $C \cong Q_4$ and has $4 \cdot 2^3 = 32$ internal edges. A Q_4 has largest adjacency eigenvalue 4, so the killed transition matrix $A/6$ has spectral radius $4/6 = 2/3$. The uniform one-step retention is also

$$\rho(C) = \frac{2|E(C)|}{6|C|} = \frac{64}{96} = \frac{2}{3}.$$

□

Theorem 9462.1451 (Tail-induced separation of retention and spectrum)**.** *For the full median closure $M = \mathrm{WNR} \cup \mathrm{CUN}$,*

$$|M| = 20, \qquad |E(M)| = 38,$$

so

$$\rho(M) = \frac{2|E(M)|}{6|M|} = \frac{19}{30} < \frac{2}{3}.$$

However

$$\lambda(M) = 0.675247931\ldots > \frac{2}{3}.$$

Proof. The retention formula gives $2 \cdot 38/(6 \cdot 20) = 19/30$. The spectral radius is the value of Definition 9462.1443, namely $(3.051487585\ldots + 1)/6 = 0.675247931\ldots$. Comparing with Definition 9462.1450 gives the strict inequalities. □

Corollary 9462.1452 (Spectral antenna behavior)**.** *The* CUN *tail is not a uniform stabilizer: it lowers average one-step retention. It is a spectral antenna: it raises the Perron survival rate of the killed walk.*

Proof. This is exactly the sign difference in Definition 9462.1451. □

Definition 9462.1453 (Tail self-energy)**.** *For the radial equations of Definition 9462.1442, eliminate the two tail variables h_R and h_Y from*

$$\mu h_R = h_0 + h_Y, \qquad \mu h_Y = h_R.$$

The resulting tail self-energy is

$$\sigma_2(\mu) = \frac{\mu}{\mu^2 - 1}.$$

Theorem 9462.1454 (Positive self-energy perturbation)**.** *The root equation at X_0 becomes*

$$\left(\mu - \frac{\mu}{\mu^2 - 1}\right) h_0 = 3h_1.$$

Thus the two-edge tail supplies a positive self-energy term at the attachment vertex. At the Perron root,

$$\sigma_2(\mu_{\max}) = 0.367137\ldots.$$

Proof. From $\mu h_Y = h_R$, obtain $h_Y = h_R/\mu$. Substitute into $\mu h_R = h_0 + h_Y$ to get

$$h_R = \frac{\mu}{\mu^2 - 1} h_0.$$

The X_0 equation $\mu h_0 = 3h_1 + h_R$ then becomes the displayed self-energy equation. □

Theorem 9462.1455 (Tail length spectral response)**.** *Attach a path of length L to the X_0 vertex of the quotient Q_3 core. The Perron values for $L = 0, \ldots, 5$ are*

L	$\mu_{\max}$	$(\mu_{\max} + 1)/6$
0	3.000000	0.666667
1	3.045475	0.674246
2	3.051488	0.675248
3	3.052315	0.675386
4	3.052430	0.675405
5	3.052446	0.675408.

The biological quotient tail has $L = 2$ and captures

$$0.9817$$

of the spectral gain visible by $L = 5$.

Proof. For each L, form the radial quotient matrix of a Q_3 core with a path of length L attached at X_0 and compute its Perron root. The gain capture ratio is

$$\frac{3.051487585\ldots - 3}{3.052445591\ldots - 3} = 0.9817\ldots.$$

□

Theorem 9462.1456 (Small mass, spectral effect)**.** *In the radial quasi-stationary distribution of Definition 9462.1445, the CUN tail states Y_R and Y_Y have total mass*

$$0.0482 + 0.0158 = 0.0640.$$

Nevertheless they raise the spectral radius from $2/3$ *to* $0.675247931\ldots$.

Proof. The masses are those of Y_R and Y_Y in Definition 9462.1445. The spectral increase is

$$0.675247931\ldots - \frac{2}{3} = 0.008581264\ldots.$$

□

Theorem 9462.1457 (CUN spectral antenna)**.** *The CUN tail separates uniform stability from quasi-stationary spectral stability:*

$$\rho(M) = \frac{19}{30} < \frac{2}{3} = \rho(\mathrm{WNR}),$$

but

$$\lambda(M) = 0.675247931\ldots > \frac{2}{3} = \lambda(\mathrm{WNR}).$$

The tail has small quasi-stationary mass, but acts as a positive self-energy perturbation of the radial Perron mode.

Proof. The retention and spectral inequalities are Definition 9462.1451. The self-energy mechanism is Definition 9462.1454, and the small tail mass is Definition 9462.1456. □

Concretized in §3771: finite CUN tail spectral antenna packet for retention, Perron response, self-energy, and small-mass rows

Definition 9462.1458 (Spectral ablation)**.** *Let $\lambda(M)$ be the killed bit-flip spectral radius of the median closure. If a vertex set, edge set, or quotient module is removed and the resulting killed-walk matrix has spectral radius λ', define the spectral loss by*

$$\Delta\lambda = \lambda(M) - \lambda'.$$

Theorem 9462.1459 (Single-codon ablation)**.** *Single-codon deletion has the following orbit-level losses:*

state	*codons*	$\Delta\lambda$
UUR	UUA, UUG	0.041994
$\mathrm{StopStart}_6$	AUA, AUG, UAA, UAG, UCA, UCG	0.035931
Anchor_6	AAA, AAG, ACA, ACG, UGA, UGG	0.033066
AGR	AGA, AGG	0.031505
CUR	CUA, CUG	0.005593
CUY	CUC, CUU	0.000643.

Proof. For each codon $c \in M$, compute the spectral radius of $M \setminus \{c\}$. The symmetry orbits of Definition 9462.1364 make the loss constant within each displayed row. □

Corollary 9462.1460 (UUR as point-level spectral hinge)**.** *The largest single-codon loss occurs at the* UUR *pair* $\{\mathrm{UUA}, \mathrm{UUG}\}$.

Proof. This is the first row of Definition 9462.1459. □

Theorem 9462.1461 (Orbit-module ablation)**.** *Deleting entire quotient modules gives*

module	*size*	$\lambda(M \setminus S)$	$\Delta\lambda$
$\mathrm{StopStart}_6$	6	0.455342	0.219906
Anchor_6	6	0.512386	0.162862
UUR	2	0.607625	0.067623
AGR	2	0.624360	0.050888
CUR	2	0.666667	0.008581
CUY	2	0.674246	0.001002.

Proof. For each module S, remove all codons in S from M and compute the Perron root of the resulting killed transition matrix. □

Corollary 9462.1462 (Point and module sensitivity differ)**.** *The most sensitive single-codon module is* UUR*, whereas the largest module-level loss comes from* $\mathrm{StopStart}_6$.

Proof. Compare Definition 9462.1459 and Definition 9462.1461. □

Theorem 9462.1463 (Tail-coupling ablation)**.** *Deleting the two interface edges*

$$\mathrm{UUA} - \mathrm{CUA}, \qquad \mathrm{UUG} - \mathrm{CUG}$$

between UUR *and* CUR *gives spectral radius*

$$\frac{2}{3}.$$

The spectral loss is

$$0.008581264\ldots.$$

Proof. Removing these two edges disconnects the CUN tail from the WNR core. The core has spectral radius 2/3 by Definition 9462.1450, and the detached tail has smaller spectral radius. Therefore the largest component spectral radius is 2/3. □

Corollary 9462.1464 (Antenna input port)**.** *The spectral gain of the* CUN *antenna enters the core through the* $\mathrm{UUR} - \mathrm{CUR}$ *interface.*

Proof. By Definition 9462.1457, the total gain over the WNR core is $0.008581264\ldots$. By Definition 9462.1463, removing the $\mathrm{UUR} - \mathrm{CUR}$ interface loses exactly this amount. □

Theorem 9462.1465 (Edge-group ablation)**.** *Deleting internal edge groups by radial module gives*

edge group	*edges*	λ'	$\Delta\lambda$
$\mathrm{Anchor}_6 - \mathrm{StopStart}_6$	12	0.512386	0.162862
$\mathrm{StopStart}_6 - \mathrm{UUR}$	6	0.607625	0.067623
$\mathrm{StopStart}_6$ *internal*	3	0.622672	0.052576
$\mathrm{AGR} - \mathrm{Anchor}_6$	6	0.624360	0.050888
Anchor_6 *internal*	3	0.627825	0.047423
UUR *internal*	1	0.655912	0.019336
AGR *internal*	1	0.661306	0.013942
$\mathrm{UUR} - \mathrm{CUR}$	2	0.666667	0.008581
CUR *internal*	1	0.672777	0.002471
$\mathrm{CUR} - \mathrm{CUY}$	2	0.674246	0.001002
CUY *internal*	1	0.674978	0.000270.

Proof. For each displayed group, remove exactly those Hamming edges from the killed-walk adjacency matrix and compute the Perron root of the remaining matrix. □

Corollary 9462.1466 (Central load-bearing interface)**.** *The largest edge-group loss comes from the* $\mathrm{Anchor}_6 - \mathrm{StopStart}_6$ *interface. This central interface is the load-bearing interface of the median closure.*

Proof. It is the first row of Definition 9462.1465. □

Theorem 9462.1467 (Single-edge ablation)**.** *Among the* 38 *internal Hamming edges of* M*, the largest single-edge loss occurs at the wobble edge*

$$\mathrm{UUA} - \mathrm{UUG},$$

with

$$\Delta\lambda = 0.019336.$$

The next six edge losses are the six $\mathrm{StopStart}_6 - \mathrm{UUR}$ *edges, each with*

$$\Delta\lambda = 0.017432.$$

Proof. Remove one internal Hamming edge at a time and compute the Perron root of the resulting killed-walk matrix. The edge $\mathrm{UUA} - \mathrm{UUG}$ gives the largest loss; the next six are the six edges between $\mathrm{StopStart}_6$ and UUR. □

Theorem 9462.1468 (Spectral hinges and load-bearing interfaces)**.** *The spectral stability of* M *separates into distinct critical structures:*

level	*critical object*
single codon	UUR
module	$\mathrm{StopStart}_6$
edge group	$\mathrm{Anchor}_6 - \mathrm{StopStart}_6$
antenna input	$\mathrm{UUR} - \mathrm{CUR}$
single edge	$\mathrm{UUA} - \mathrm{UUG}$.

Proof. The five entries are respectively Definition 9462.1460, Definition 9462.1461, Definition 9462.1466, Definition 9462.1464, and Definition 9462.1467. □

Concretized in §8792: DNA-window trigger frontiers realize boundary-tile hinge and ablation rows as finite trigger, support, reliability, refusal, transport, replay, provenance, and NameCert data.

Concretized in §3959: The spectral ablation hinge is realized as finite deletion, killed-walk, spectral-loss, orbit-module, transport, replay, provenance, and local naming rows.

Concretized in §3961: Spectral ablation hinges are realized as finite killed-walk matrix, orbit-module, point-ablation, edge-ablation, hinge-ranking, transport, replay, provenance, and NameCert rows.

Definition 9462.1469 (Doob-conditioned survival dynamics)**.** *Let Q_M be the killed one-bit transition matrix on $M = \mathrm{Med}(\mathcal{R})$, and let*

$$Q_M h = \lambda(M) h$$

be its positive Perron eigenvector. The Doob-conditioned survival chain on M is

$$P^h(x, y) = \frac{Q_M(x, y)h(y)}{\lambda(M)h(x)}.$$

It is the Markov chain obtained from the killed walk by conditioning on asymptotic survival inside the median closure.

Theorem 9462.1470 (Six-state Doob transition matrix)**.** *In the radial state order*

$$S_0 = \mathrm{AGR}, \quad S_1 = \mathrm{Anchor}_6, \quad S_2 = \mathrm{StopStart}_6, \quad S_3 = \mathrm{UUR}, \quad S_4 = \mathrm{CUR}, \quad S_5 = \mathrm{CUY},$$

the Doob-conditioned chain has transition matrix

	S_0	S_1	S_2	S_3	S_4	S_5
S_0	0.2468	0.7532	0	0	0	0
S_1	0.2427	0.2468	0.5105	0	0	0
S_2	0	0.4773	0.2468	0.2758	0	0
S_3	0	0	0.6626	0.2468	0.0906	0
S_4	0	0	0	0.6723	0.2468	0.0809
S_5	0	0	0	0	0.7532	0.2468.

Proof. The radial quotient of $P = \pi(M)$ has neighbor matrix A_{rad} from Definition 9462.1442. The f_3 wobble edge adds one self-loop to every radial state, so the neighbor-count matrix for the six-state conditioned chain is

$$\begin{pmatrix} 1 & 3 & 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 0 \\ 0 & 2 & 1 & 1 & 0 & 0 \\ 0 & 0 & 3 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

If $\mu = 6\lambda(M) = 4.051487585\ldots$, then the radial Doob formula is

$$P^h_{ij} = \frac{N_{ij} h_j}{\mu h_i}.$$

Substituting the Perron vector of Definition 9462.1445 gives the displayed matrix; each row sums to 1. □

Concretized in §??: finite Doob-conditioned survival-chain NameCert data

Corollary 9462.1471 (Central drift of conditioned survival)**.** *The mean radial drift of the Doob chain is*

state	*drift*
S_0	0.7532
S_1	0.2679
S_2	−0.2015
S_3	−0.5719
S_4	−0.5914
S_5	−0.7532.

Thus the drift reverses at the Anchor_6–$\mathrm{StopStart}_6$ *interface.*

Proof. Assign positions $0, \ldots, 5$ to $S_0, \ldots, S_5$ and compute the expected one-step displacement from the rows of Definition 9462.1470. □

Theorem 9462.1472 (Doob stationary distribution)**.** *The stationary distribution of the Doob-conditioned chain satisfies*

$$\pi_i^h \propto |S_i| h_i^2.$$

Numerically,

state	π^h
$S_0 = \mathrm{AGR}$	0.1110
$S_1 = \mathrm{Anchor}_6$	0.3444
$S_2 = \mathrm{StopStart}_6$	0.3684
$S_3 = \mathrm{UUR}$	0.1534
$S_4 = \mathrm{CUR}$	0.0207
$S_5 = \mathrm{CUY}$	0.0022.

In particular,

$$\pi^h(\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6) = 0.7128\ldots,$$

and including the UUR *bridge gives about* 86.6% *of the conditioned stationary mass.*

Proof. For a symmetric killed matrix, the Doob transform is reversible with stationary weights proportional to $h(x)^2$. Collapsing over a radial state multiplies by its lift size $|S_i|$. The displayed values are obtained from the Perron vector of Definition 9462.1445. □

Theorem 9462.1473 (QSD and Doob residence differ)**.** *The quasi-stationary radial mass and the Doob stationary radial mass are*

state	*QSD*	π^h
$S_0 = \mathrm{AGR}$	0.1117	0.1110
$S_1 = \mathrm{Anchor}_6$	0.3407	0.3444
$S_2 = \mathrm{StopStart}_6$	0.3524	0.3684
$S_3 = \mathrm{UUR}$	0.1313	0.1534
$S_4 = \mathrm{CUR}$	0.0482	0.0207
$S_5 = \mathrm{CUY}$	0.0158	0.0022.

The Doob chain suppresses the terminal CUN states more strongly than the quasi-stationary distribution.

Proof. The quasi-stationary radial mass is proportional to $|S_i|h_i$, as in Definition 9462.1445. The Doob stationary mass is proportional to $|S_i|h_i^2$ by Definition 9462.1472. Squaring the small tail potentials lowers the CUR and CUY masses. □

Definition 9462.1474 (Perron edge flux)**.** *For adjacent radial states S_i and S_j, define the stationary Perron flux of the Doob chain by*

$$\Phi_{ij} = \pi_i^h P_{ij}^h.$$

The reversibility of the Doob chain gives

$$\pi_i^h P_{ij}^h = \pi_j^h P_{ji}^h.$$

Theorem 9462.1475 (Perron flux concentration)**.** *The radial edge fluxes are*

edge group	Φ
$S_0 - S_1$	0.0836
$S_1 - S_2$	0.1758
$S_2 - S_3$	0.1016
$S_3 - S_4$	0.0139
$S_4 - S_5$	0.0017.

The largest Perron flux crosses the Anchor_6–$\mathrm{StopStart}_6$ *interface.*

Proof. Multiply the stationary masses from Definition 9462.1472 by the transition probabilities of Definition 9462.1470. The maximum entry is the $S_1 - S_2$ flux. □

Corollary 9462.1476 (The CUN tail is not a residence core)**.** *The CUN input flux $S_3 - S_4$ is only* 0.0139*, and the terminal tail flux $S_4 - S_5$ is only* 0.0017*. Thus the CUN tail is a spectral antenna in the sense of Definition 9462.1457, but not a residence core for the conditioned survival chain.*

Proof. These are the final two rows of Definition 9462.1475. They are much smaller than the central flux 0.1758. □

Definition 9462.1477 (Prefix conductance)**.** *For the prefix*

$$A_k = \{S_0, \dots, S_{k-1}\}$$

of the radial path, define the Doob-chain conductance by

$$\phi(A_k) = \frac{\Phi(A_k, A_k^c)}{\min(\pi^h(A_k), 1 - \pi^h(A_k))}.$$

Theorem 9462.1478 (Balanced central bottleneck)**.** *The prefix cuts have conductance*

cut	$\pi^h(A)$	$\Phi(A, A^c)$	$\phi(A)$
$S_0 \mid S_1 \cdots S_5$	0.1110	0.0836	0.7532
$S_0S_1 \mid S_2 \cdots S_5$	0.4554	0.1758	0.3861
$S_0S_1S_2 \mid S_3 \cdots S_5$	0.8237	0.1016	0.5765
$S_0 \cdots S_3 \mid S_4S_5$	0.9771	0.0139	0.6071
$S_0 \cdots S_4 \mid S_5$	0.9978	0.0017	0.7532.

The minimum occurs at the central cut through $S_1 - S_2$.

Proof. For each prefix, the boundary flux is the corresponding path flux from Definition 9462.1475. Dividing by the smaller side mass gives the displayed conductance values. □

Theorem 9462.1479 (Doob-conditioned median-closure dynamics)**.** *The conditioned survival dynamics of the median closure concentrates its stationary mass, Perron flux, and balanced bottleneck at the* Anchor_6–$\mathrm{StopStart}_6$ *interface. The* UUR *state acts as the adjacent survival bridge, while the CUN tail has low conditioned residence and low conditioned flux.*

Proof. Mass concentration is Definition 9462.1472; drift reversal is Definition 9462.1471; flux concentration is Definition 9462.1475; and the balanced bottleneck is Definition 9462.1478. The low-tail claim is Definition 9462.1476. □

Definition 9462.1480 (First-exit channel)**.** *For the killed bit-flip walk on* $M = \mathrm{Med}(\mathcal{R})$*, a first-exit step is a transition*

$$x \in M, \qquad x^{(i)} \notin M.$$

The associated channel records the origin state of x*, the flipped bit* i*, and the outside target codon* $x^{(i)}$ *together with its standard genetic-code class.*

Theorem 9462.1481 (Quasi-stationary exit origins)**.** *Under the quasi-stationary distribution of the killed walk on* M*, the pre-exit origin state has distribution*

origin state	*conditional probability*
$\mathrm{StopStart}_6$	0.3617
Anchor_6	0.3497
AGR	0.1146
CUR	0.0742
UUR	0.0674
CUY	0.0324.

Thus

$$\Pr(\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6 \mid \textit{first exit}) = 0.7114\ldots.$$

Proof. For each boundary half-edge $x \in M$, $x^{(i)} \notin M$, add $q(x)/6$, where q is the quasi-stationary distribution. Normalizing by the total exit probability $1 - \lambda(M)$ and grouping by the six radial states gives the table. □

Corollary 9462.1482 (Tail leakage is not the dominant exit source)**.** *The two CUN tail states* CUR *and* CUY *contribute only*

$$0.0742 + 0.0324 = 0.1066$$

of quasi-stationary first exits.

Proof. This is the sum of the CUR and CUY rows of Definition 9462.1481. □

Theorem 9462.1483 (Quasi-stationary exit bits)**.** *The bit direction of a quasi-stationary first exit has distribution*

bit	*reading*	*conditional probability*
s_3	*third-position shape*	0.4804
f_1	*first-position family*	0.4211
f_2	*second-position family*	0.0328
s_1	*first-position shape*	0.0328
s_2	*second-position shape*	0.0328
f_3	*third-position family*	0.

Consequently,

$$\Pr(s_3 \text{ or } f_1 \mid \textit{first exit}) = 0.9015\ldots.$$

Proof. Sum the quasi-stationary boundary weights $q(x)/6$ over all boundary half-edges with the same flipped bit. The f_3 row is zero because M is f_3-saturated by Definition 9462.1429. □

Corollary 9462.1484 (Silent escape axis)**.** *The third-position family bit is a silent escape axis:*

$$\Pr(f_3 \textit{ exit}) = 0.$$

Proof. This is the final row of Definition 9462.1483, equivalently the f_3 saturation statement of Definition 9462.1429. □

Theorem 9462.1485 (Eventual first-exit bit by start state)**.** *Let $u_i(x)$ be the probability that the eventual first exit from M, starting at x, occurs through bit i. Averaged over the six radial states, the probabilities are*

start state	s_1	f_1	s_2	f_2	s_3	f_3
AGR	0.0041	0.4860	0.0041	0.0041	0.5017	0
Anchor_6	0.0068	0.4766	0.0068	0.0068	0.5029	0
StopStart_6	0.0151	0.4486	0.0151	0.0151	0.5063	0
UUR	0.0616	0.2895	0.0616	0.0616	0.5257	0
CUR	0.2628	0.1020	0.2628	0.2628	0.1095	0
CUY	0.2526	0.2204	0.2526	0.2526	0.0219	0.

Proof. For each bit i, solve the harmonic first-exit equation

$$u_i(x) = \frac{1}{6}\mathbf{1}[x^{(i)} \notin M] + \sum_{\substack{y \in M \\ y \sim x}} \frac{1}{6} u_i(y).$$

Equivalently,

$$(I - Q_M)u_i = r_i,$$

where $r_i(x) = 1/6$ exactly when the i-flip leaves M. Averaging the solution over each radial state gives the displayed table. □

Corollary 9462.1486 (Core and tail exit mechanisms)**.** *The core states* AGR*,* Anchor_6*, and* StopStart_6 *exit almost entirely through f_1 and s_3. The CUN tail states* CUR *and* CUY *instead place most eventual exit probability on s_1, s_2, and f_2.*

Proof. In Definition 9462.1485, the $f_1 + s_3$ sums for the three core states are respectively 0.9877, 0.9795, and 0.9549. For CUR and CUY, the sums over s_1, s_2, and f_2 are 0.7884 and 0.7577. □

Theorem 9462.1487 (Outside target classes)**.** *Quasi-stationary first exits land in the following standard-code classes:*

outside class	*conditional probability*
S	0.1176
P	0.0931
V	0.0931
Q	0.0850
F	0.0755
I	0.0603
Y	0.0603
A	0.0583
C	0.0583
E	0.0583
N	0.0583
R	0.0583
T	0.0583
G	0.0573
H	0.0081.

No first-exit target lies in the standard Stop class.

Proof. For each boundary half-edge $x \in M$, $y = x^{(i)} \notin M$, add the quasi-stationary boundary weight $q(x)/6$ to the standard-code class of y, then normalize by $1 - \lambda(M)$. The absence of a Stop target follows because UAA, UAG, and UGA lie inside M. □

Theorem 9462.1488 (Outside codon targets)**.** *The largest quasi-stationary outside codon targets are*

outside codon	*class*	*conditional probability*
CAA, CAG	Q	0.0425
CCA, CCG	P	0.0425
GUA, GUG	V	0.0425
UUC, UUU	F	0.0377
AUC, AUU	I	0.0301
UAC, UAU	Y	0.0301
UCC, UCU	S	0.0301.

Proof. The same boundary half-edge weights as in Definition 9462.1487 are aggregated by outside codon rather than by standard-code class. Equal-probability targets are grouped in the table. □

Theorem 9462.1489 (First-exit channel geometry)**.** *The first-exit geometry of the wobble-saturated median closure is controlled by two gate axes, s_3 and f_1, while the f_3 wobble axis is internally closed. Quasi-stationary exits mostly originate from* $\mathrm{Anchor}_6 \cup \mathrm{StopStart}_6$ *and land in ordinary sense classes outside M, not in the Stop class.*

Proof. The origin statement is Definition 9462.1481. The gate-axis statement is Definition 9462.1483; the silent-axis statement is Definition 9462.1484. The target-class statement is Definition 9462.1487. □

Concretized in §3677: The first-exit channel geometry is realized as finite origin, bit, outside-target, target-class, transition-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §3678: The boundary-shadow trigger packet realizes first-exit half-edge, bit, outside target, and class rows as finite NameCert data.

Definition 9462.1490 (Post-median gate shell)**.** *Let*

$$M = \mathrm{Med}(\mathcal{R}).$$

The first post-median gate shell is the external f_3-saturated set

$$\mathcal{G}_1 = \{\mathrm{CAA}, \mathrm{CAG}, \mathrm{CCA}, \mathrm{CCG}, \mathrm{GUA}, \mathrm{GUG}\}.$$

Equivalently,

$$\mathcal{G}_1 = \mathrm{CAR} \cup \mathrm{CCR} \cup \mathrm{GUR}.$$

Write

$$M^+ = M \cup \mathcal{G}_1.$$

Theorem 9462.1491 (Post-median single-point spectral optimum)**.** *Among all single codons $p \in Q_6 \setminus M$, the maximum of*

$$\lambda(M \cup \{p\})$$

is attained exactly at

$$\mathrm{CAA}, \mathrm{CAG}, \mathrm{CCA}, \mathrm{CCG}, \mathrm{GUA}, \mathrm{GUG}.$$

For each of these six codons,

$$\lambda(M \cup \{p\}) = 0.681004866\ldots,$$

so the single-point gain over M is

$$0.005756935\ldots.$$

Proof. Enumerate the 44 codons in $Q_6 \setminus M$ and compute the Perron root of the killed transition matrix on $M \cup \{p\}$. The six displayed codons are the complete maximizer set. □

Corollary 9462.1492 (Gate shell and first-exit targets)**.** *The codons of $\mathcal{G}_1$ are exactly the highest-probability outside codon targets of Definition 9462.1488.*

Proof. Definition 9462.1488 lists the same six codons at probability 0.0425 each. By Definition 9462.1491, the same six codons are the post-median single-point spectral maximizers. □

Theorem 9462.1493 (Boolean form of the gate shell)**.** *In the two-axis bit coordinates, $\mathcal{G}_1$ is the set*

$$f_1 = 1, \qquad s_3 = 1, \qquad \mathrm{wt}(s_1, s_2, f_2) = 1,$$

with f_3 free.

Proof. The three quotient fibers are

$$\mathrm{CAR}, \qquad \mathrm{CCR}, \qquad \mathrm{GUR}.$$

They all have first-position family bit $f_1 = 1$ and third-position shape bit $s_3 = 1$. Among the three symmetric coordinates s_1, s_2, f_2, exactly one is 1 for each of the three quotient fibers. The third-position family bit f_3 distinguishes the two lifts inside each fiber. □

Theorem 9462.1494 (Gate shell as a full external orbit)**.** *Under*

$$\Gamma = \mathrm{Stab}(M) \cong S_3 \times C_2,$$

the set $\mathcal{G}_1$ is a full orbit in $Q_6 \setminus M$.

Proof. The S_3 factor permutes the three coordinates s_1, s_2, f_2, and the condition

$$\mathrm{wt}(s_1, s_2, f_2) = 1$$

is a single S_3 orbit. The C_2 factor flips f_3, making each quotient point into its full f_3 pair. Hence the orbit has $3 \cdot 2 = 6$ codons, namely $\mathcal{G}_1$. □

Theorem 9462.1495 (Best symmetric post-median orbit)**.** *Among the* 10 *external* Γ*-orbits in* $Q_6 \setminus M$*, adjoining* $\mathcal{G}_1$ *gives the largest spectral radius:*

$$\lambda(M \cup \mathcal{G}_1) = 0.727773877\ldots.$$

Proof. Enumerate the external orbits of the action of $\Gamma = \mathrm{Stab}(M)$ on $Q_6 \setminus M$. For each orbit O, compute $\lambda(M \cup O)$. The maximum is attained by the orbit of Definition 9462.1494. □

Theorem 9462.1496 (Geometry of the post-median shell)**.** *The extension*

$$M^+ = M \cup \mathcal{G}_1$$

has

$$|M^+| = 26, \qquad |E(M^+)| = 53.$$

Thus its uniform one-step retention is

$$\rho(M^+) = \frac{2|E(M^+)|}{6|M^+|} = \frac{53}{78} = 0.679487\ldots.$$

Its killed-walk spectral radius is

$$\lambda(M^+) = 0.727773877\ldots.$$

Proof. Directly count the induced Hamming edges after adjoining the six codons of $\mathcal{G}_1$. The retention formula is $2|E|/(6|V|)$, and the spectral value is the Perron root of the killed transition matrix on M^+. □

Theorem 9462.1497 (CUR exit closure)**.** *In M^+, the state*

$$\mathrm{CUR} = \{\mathrm{CUA}, \mathrm{CUG}\}$$

has no one-bit exit:

$$P_{\mathrm{out}}(\mathrm{CUR}; M^+) = 0.$$

Proof. The six one-bit neighbors of CUA are

$$\mathrm{UUA}, \mathrm{CUG}, \mathrm{CCA}, \mathrm{CAA}, \mathrm{GUA}, \mathrm{CUU}.$$

The codons UUA, CUG, CUU lie in M, while CCA, CAA, GUA lie in $\mathcal{G}_1$. The six one-bit neighbors of CUG are handled identically, with the outside neighbors CCG, CAG, GUG supplied by $\mathcal{G}_1$. □

Corollary 9462.1498 (CUN tail exit shell)**.** *The set $\mathcal{G}_1$ is the first exit-closing shell for the* CUR *part of the CUN tail.*

Proof. By Definition 9462.1497, adjoining $\mathcal{G}_1$ changes the CUR leakage probability from 1/2 in M to 0 in M^+. □

Theorem 9462.1499 (Gate-shell bypass)**.** *In the state graph of M^+, the gate shell supplies a two-step bypass*

$$\mathrm{StopStart}_6 - \mathcal{G}_1 - \mathrm{CUR}.$$

There are 6 *edges from* $\mathrm{StopStart}_6$ *to* $\mathcal{G}_1$ *and* 6 *edges from* $\mathcal{G}_1$ *to* CUR*.*

Proof. Group the Hamming edges of M^+ by the six original radial states and the added shell state $\mathcal{G}_1$. The cross-edge counts are exactly 6 for $\mathrm{StopStart}_6$–$\mathcal{G}_1$ and 6 for $\mathcal{G}_1$–CUR. □

Theorem 9462.1500 (Best wobble-pair expansion)**.** *If the added set must be a full f_3 pair outside M, the spectral maximizers are exactly*

$$\mathrm{CAR}, \qquad \mathrm{CCR}, \qquad \mathrm{GUR}.$$

Each gives

$$\lambda(M \cup \mathit{pair}) = 0.691331504\ldots.$$

Their union is $\mathcal{G}_1$.

Proof. Enumerate the external f_3 pairs and compute the killed-walk Perron root after adjoining each pair. The three displayed pairs are the complete maximizer set; their union is the shell of Definition 9462.1490. □

Theorem 9462.1501 (Post-median gate shell)**.** *The set*

$$\mathcal{G}_1 = \mathrm{CAR} \cup \mathrm{CCR} \cup \mathrm{GUR}$$

is simultaneously the top first-exit target layer, the complete post-median external orbit selected by Γ, the best symmetric spectral extension of M, and the exit-closing shell for CUR*.*

Proof. The first-exit identification is Definition 9462.1492. The orbit statement is Definition 9462.1494. The symmetric spectral optimality is Definition 9462.1495. The exit closure is Definition 9462.1498. □

Concretized in §3625: Post-median gate shells are realized as finite codon-shell, Boolean-coordinate, spectral-optimum, first-exit, transport, replay, provenance, and NameCert rows.

Definition 9462.1502 (Radial shell closure beyond the gate shell)**.** *Let*

$$M^\star = \mathrm{NNR} \cup \mathrm{CUY}.$$

Thus $M^\star$ contains every codon whose third position is purine, together with the two codons CUU *and* CUC*. Hence*

$$|M^\star| = 34.$$

Definition 9462.1503 (Y-halfspace shells around CUY)**.** *In the third-position Y halfspace, define H_k to be the set of codons whose first two positions have Hamming distance k from* CU*. Then*

shell	*size*	*description*
H_0	2	CUY
H_1	8	UUY ∪ CCY ∪ CAY ∪ GUY
H_2	12	*six two-flip Y-pairs*
H_3	8	*four three-flip Y-pairs*
H_4	2	AGY.

Thus

$$M^\star = \mathrm{NNR} \cup H_0.$$

Theorem 9462.1504 (Median-closed halfspace anchor)**.** *The set $M^\star$ is median-closed:*

$$\mathrm{Med}(M^\star) = M^\star.$$

Proof. After quotienting by the saturated f_3 direction, the set is the complete halfspace $s_3 = 1$ together with the single Y-side point CUY. A three-point median of quotient points either has at least two entries in the halfspace, in which case it remains in the halfspace, or has at least two entries at CUY, in which case it remains at CUY or at an already present halfspace point. Lifting back along f_3 preserves median closure. □

Theorem 9462.1505 (Shell-triggered median percolation)**.** *For every $k = 1, 2, 3, 4$,*

$$\mathrm{Med}(M^\star \cup H_k) = Q_6.$$

The finite stage counts are

added shell	*start size*	*closure size*	*stage additions*
H_1	42	64	12, 10
H_2	46	64	18
H_3	42	64	22
H_4	36	64	28.

Proof. Compute the coordinatewise median closure from each starting set $M^\star \cup H_k$. Each closure is all 64 codons. The stage additions in the table record the number of codons first generated at each median-closure pass. □

Theorem 9462.1506 (Antipodal single trigger)**.** *The two codons in*

$$H_4 = \mathrm{AGY} = \{\mathrm{AGU}, \mathrm{AGC}\}$$

are single-codon median triggers:

$$\mathrm{Med}(M^\star \cup \{\mathrm{AGU}\}) = Q_6, \qquad \mathrm{Med}(M^\star \cup \{\mathrm{AGC}\}) = Q_6.$$

In both cases the closure reaches Q_6 in one median pass.

Proof. In the f_3 quotient, CUY and AGY are antipodal points in the four coordinates of the third-position Y halfspace. For any target point b in that Y halfspace, take the point t_b in the already present R halfspace with the same first-four-bit coordinates as b. The median of CUY, AGY, and t_b has the first four coordinates of t_b and third-position shape Y, hence is b. Thus the whole Y halfspace is generated in one pass; the R halfspace was already present. □

Corollary 9462.1507 (CUY anchor and AGY antipode)**.** *The set $M^\star$ is a complete R halfspace with a Y-side anchor* CUY*, and* AGY *is the antipodal median trigger for that anchor.*

Proof. This is the geometric content of Definition 9462.1506. □

Theorem 9462.1508 (Single-point spectrum and median triggering differ)**.** *From $M^\star$, single-codon spectral stabilization and median percolation select different shells:*

single-codon shell	Med *-closure size*	$\lambda(M^\star \cup \{p\})$
H_1	36	0.837956
H_2	40	0.837179
H_3	48	0.837165
$H_4 = \mathrm{AGY}$	64	0.837156.

Thus the best single-point spectral stabilizer lies in H_1, while the strongest median trigger lies in H_4.

Proof. For a representative codon in each shell, compute both $\lambda(M^\star \cup \{p\})$ and the size of the median closure. The spectral values are largest in the nearest shell H_1, whereas Definition 9462.1506 gives full-cube median generation from the antipodal shell H_4. □

Theorem 9462.1509 (Radial shell filling spectrum)**.** *Let*

$$S_j = \mathrm{NNR} \cup \bigcup_{k=0}^{j} H_k.$$

Then

level	*size*	*structure*	λ	ρ
S_0	34	$\mathrm{NNR} \cup H_0$	0.836111	0.813725
S_1	42	$+H_1$	0.859128	0.817460
S_2	54	$+H_2$	0.921172	0.895062
S_3	62	$+H_3$	0.978349	0.973118
S_4	64	$+H_4 = Q_6$	1.000000	1.000000.

Proof. For each S_j, count induced Hamming edges to compute $\rho = 2|E(S_j)|/(6|S_j|)$, and compute the Perron root of the killed transition matrix on S_j. At $j = 4$ the set is all of Q_6. □

Theorem 9462.1510 (Antipodal triggering beyond the radial shell)**.** *The set $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ is median-closed, but adjoining either codon of* AGY *generates the whole codon cube by median closure:*

$$M^\star \quad \textit{is closed}, \qquad M^\star \cup \{\mathrm{AGU}\} \textit{ or } M^\star \cup \{\mathrm{AGC}\} \Longrightarrow Q_6.$$

Proof. The closedness is Definition 9462.1504. The single-codon full-cube generation is Definition 9462.1506. □

Concretized in §3694: The antipodal AGY trigger is realized as finite median-closure, shell, trigger-ledger, transport, replay, provenance, and NameCert rows.

Definition 9462.1511 (Anchor subcube span)**.** *Let*

$$M^\star = \mathrm{NNR} \cup \mathrm{CUY}.$$

Every codon outside $M^\star$ lies in the third-position Y halfspace. For a set $B \subseteq Q_6 \setminus M^\star$, let $\rho(B)$ be the projection of B to the four bits of the first two codon positions, and let $D(B)$ be the smallest axis-aligned subcube of that Q_4 which contains the anchor CU *and $\rho(B)$. Write*

$$d(B) = \dim D(B).$$

Theorem 9462.1512 (Anchor-subcube closure formula)**.** *For every $B \subseteq Q_6 \setminus M^\star$,*

$$\mathrm{Med}(M^\star \cup B) = \mathrm{NNR} \cup (D(B) \times Y).$$

Proof. The set $M^\star$ contains the full third-position R halfspace. Take two points x, y in the Y halfspace and a point t in the R halfspace. In the third-position shape coordinate, the median of x, y, t remains in the Y halfspace. In any one of the first-four coordinates, if x and y agree, the median is forced to that common value; if they disagree, the corresponding coordinate of t chooses the median value. Since the R halfspace is complete, all choices of t are available. Iterating this interval operation generates exactly the smallest axis-aligned subcube containing the anchor CU and $\rho(B)$, and no coordinate outside that subcube can be introduced. □

Corollary 9462.1513 (Closure size by span dimension)**.** *The closure size depends only on $d(B)$:*

$$|\,\mathrm{Med}(M^\star \cup B)| = 32 + 2^{d(B)+1}.$$

Thus

$d(B)$	*closure size*
0	34
1	36
2	40
3	48
4	64.

Proof. The R halfspace NNR has 32 codons. A $d(B)$-dimensional subcube in the first-four coordinates has $2^{d(B)}$ vertices, and lifting across the two third-position Y codons contributes the factor 2. □

Theorem 9462.1514 (Single-codon span classification)**.** *If $p \in Q_6 \setminus M^\star$ lies in the shell H_d around* CUY*, then*

$$|\,\mathrm{Med}(M^\star \cup \{p\})| = 32 + 2^{d+1}.$$

Equivalently,

shell	*codons*	*closure size*
H_1	8	36
H_2	12	40
H_3	8	48
$H_4 = \mathrm{AGY}$	2	64.

Proof. For a single codon, the subcube spanned by its projection and the anchor CU is exactly their interval in the first-four-bit Q_4. Its dimension is their Hamming distance, namely the shell index d. □

Theorem 9462.1515 (External subset trigger counts)**.** *Among the* 30 *codons of* $Q_6 \setminus M^\star$*, the distribution of median-closure sizes for* m *adjoined codons is*

m	*candidate sets*	36	40	48	64
1	30	8	12	8	2
2	435	4	78	196	157
3	4060	0	120	1216	2724
4	27405	0	90	3824	23491.

Proof. For each m-element subset B of $Q_6 \setminus M^\star$, compute the union of first-four-coordinate supports relative to the anchor CU, then apply Definition 9462.1513. The same table is checked against representative median closures for every nonempty support pattern. □

Definition 9462.1516 (Anchor-relative support)**.** *For* $p \in Q_6 \setminus M^\star$*, define*

$$\mathrm{supp}(p) = \{i \in \{s_1, f_1, s_2, f_2\} : \rho(p)_i \neq \mathrm{CU}_i\}.$$

Theorem 9462.1517 (Full-cube trigger criterion)**.** *For* $B \subseteq Q_6 \setminus M^\star$*,*

$$\mathrm{Med}(M^\star \cup B) = Q_6$$

if and only if

$$\bigcup_{p \in B} \mathrm{supp}(p) = \{s_1, f_1, s_2, f_2\}.$$

Proof. By Definition 9462.1512, the closure is the whole cube exactly when $D(B)$ is the full first-two-position Q_4. This is equivalent to every one of the four coordinates being freed by at least one anchor-relative support. □

Corollary 9462.1518 (AGY as the full-support point)**.** *The codons* AGU *and* AGC *have anchor-relative support*

$$\{s_1, f_1, s_2, f_2\}.$$

Hence either one is a full-cube trigger for $M^\star$*.*

Proof. The first two positions AG are antipodal to CU in the four-bit coordinate system. The claim follows from Definition 9462.1517. □

Theorem 9462.1519 (Anchor-subcube trigger theorem)**.** *The median dynamics beyond* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ *are completely determined by the subcube span of the outside* Y*-halfspace points relative to the anchor* CUY*. In particular, the antipodal trigger* AGY *is the single-codon case with* $d(B) = 4$*.*

Proof. The exact closure formula is Definition 9462.1512. The size and full-cube criteria are Definition 9462.1513 and Definition 9462.1517. The antipodal case is Definition 9462.1518. □

Concretized in §3651: Anchor subcube triggers are realized as finite anchor, boundary-tile, coverage-rank, trigger-algebra, blocker, transport, replay, provenance, and NameCert rows.

Definition 9462.1520 (Boolean trigger closure)**.** *Let $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, and let the first-two-position coordinates be ordered as*

$$s_1, f_1, s_2, f_2.$$

For $p \in Q_6 \setminus M^\star$, define the anchor support

$$\sigma(p) = \{i \in \{s_1, f_1, s_2, f_2\} : p_i \neq \mathrm{CU}_i\}.$$

For $B \subseteq Q_6 \setminus M^\star$, put

$$J(B) = \bigcup_{p \in B} \sigma(p), \qquad r(B) = |J(B)|.$$

For $J \subseteq \{s_1, f_1, s_2, f_2\}$, let $Q_J(\mathrm{CU})$ be the first-two-position subcube based at CU with exactly the coordinates in J free, and set

$$C_J = \mathrm{NNR} \cup (Q_J(\mathrm{CU}) \times Y).$$

Theorem 9462.1521 (Trigger lattice)**.** *For every $B \subseteq Q_6 \setminus M^\star$,*

$$\mathrm{Med}(M^\star \cup B) = C_{J(B)}.$$

Consequently the post-$M^\star$ median closures are exactly the sixteen sets C_J indexed by $J \subseteq \{s_1, f_1, s_2, f_2\}$.

Proof. This is Definition 9462.1512 with the subcube $D(B)$ written by its free-coordinate set. The free coordinates are exactly the union of the anchor-relative supports of the adjoined codons. □

Corollary 9462.1522 (Rank size and full triggering)**.** *If $|J| = r$, then*

$$|C_J| = 32 + 2^{r+1}.$$

Thus

r	$\lvert C_J \rvert$
0	34
1	36
2	40
3	48
4	64.

In particular, $\mathrm{Med}(M^\star \cup B) = Q_6$ *if and only if* $J(B) = \{s_1, f_1, s_2, f_2\}$.

Proof. The R halfspace contributes 32 codons. The Y-side trigger subcube has 2^r first-two-position vertices and two third-position Y-lifts. Full-cube triggering is the case in which all four first-two-position coordinates are free. □

Theorem 9462.1523 (Boolean lattice operations)**.** *For all $I, J \subseteq \{s_1, f_1, s_2, f_2\}$,*

$$C_I \cap C_J = C_{I \cap J},$$

and

$$\mathrm{Med}(C_I \cup C_J) = C_{I \cup J}.$$

Proof. On the Y side, intersection fixes exactly the coordinates fixed by both subcubes, hence leaves the free set $I \cap J$. Median closure of the union is governed by the free-coordinate support of the union, so the free set is $I \cup J$. The common R halfspace NNR is present in every C_J and is unchanged by both operations. □

Theorem 9462.1524 (Cubical and spectral rank table)**.** *The cubical f-vector and killed-walk spectral radius of C_J depend only on $r = |J|$:*

r	$\lvert C_J\rvert$	$f(C_J)$	$\lambda(C_J)$
0	34	$(34, 83, 81, 40, 10, 1)$	0.836111
1	36	$(36, 88, 85, 41, 10, 1)$	0.840912
2	40	$(40, 100, 98, 47, 11, 1)$	0.855963
3	48	$(48, 128, 136, 72, 19, 2)$	0.902369
4	64	$(64, 192, 240, 160, 60, 12, 1)$	1.000000.

The cubical formula is

$$f_k(C_J) = f_k(Q_5) + f_k(Q_{r+2}) - f_k(Q_{r+1}), \qquad f_k(Q_n) = 2^{n-k}\binom{n}{k}.$$

Proof. The complex is the union of the complete R halfspace Q_5 and the trigger slab $Q_J(\mathrm{CU}) \times N \cong Q_{r+2}$. Their intersection is Q_{r+1}, giving the displayed inclusion-exclusion formula. The spectral radii are obtained by the finite killed-walk adjacency computation on each rank representative. □

Theorem 9462.1525 (Quotient spectral equations)**.** *Since every C_J is saturated in the f_3 direction, its killed-walk radius has the form*

$$\lambda_r = \frac{\mu_r + 1}{6}, \qquad r = |J|.$$

The quotient spectral roots are

r	*equation for* μ_r	μ_r
0	$\mu^6 - 21\mu^4 + 80\mu^2 - 24 = 0$	4.016667
1	$\mu^4 - 4\mu^3 - 5\mu^2 + 18\mu + 6 = 0$	4.045475
2	$\mu^4 - 8\mu^3 + 19\mu^2 - 12\mu - 2 = 0$	4.135779
3	$\mu^2 - 6\mu + 7 = 0$	4.414214
4	$\mu - 5 = 0$	5.000000.

Proof. The f_3-wobble edge contributes one uniform self-neighbor in the quotient dynamics, so the six-bit killed-walk eigenvalue is $(\mu_r+1)/6$. The displayed equations are the characteristic equations of the finite quotient adjacency matrices for rank $r = 0, 1, 2, 3, 4$, with the largest positive root selected in each case. □

Corollary 9462.1526 (Near-global rank)**.** *At rank* 3*, the closure has* 48 *codons and spectral radius*

$$\lambda_3 = 0.902369.$$

Thus a trigger set covering any three of the four first-two-position coordinates already lies in a high-survival regime, while covering the last coordinate gives $C_{\{s_1,f_1,s_2,f_2\}} = Q_6$.

Proof. This is the $r = 3$ row of Definition 9462.1524 and Definition 9462.1525. □

Theorem 9462.1527 (Rank distribution for external codons)**.** *Among m adjoined codons from $Q_6 \setminus M^\star$, the number whose final trigger rank is d is*

$$N(m, d) = \binom{4}{d} \sum_{j=0}^{d} (-1)^j \binom{d}{j} \binom{2(2^{d-j} - 1)}{m}.$$

For $m \leq 6$ this gives

m	$r=1$	$r=2$	$r=3$	$r=4$	$\binom{30}{m}$
1	8	12	8	2	30
2	4	78	196	157	435
3	0	120	1216	2724	4060
4	0	90	3824	23491	27405
5	0	36	7936	134534	142506
6	0	6	12000	581769	593775.

Proof. Choose the d coordinates that appear in the final support. Inside those coordinates there are $2(2^d - 1)$ codon lifts with nonempty support. Inclusion-exclusion removes the selections that miss at least one of the d chosen coordinates, giving the stated formula. □

Corollary 9462.1528 (Full-trigger probabilities)**.** *For uniformly chosen m-codon subsets of $Q_6 \setminus M^\star$, the full-cube trigger probabilities for $m = 1, \ldots, 6$ are*

m	$\Pr(\mathrm{Med}(M^\star \cup B) = Q_6)$
1	0.0667
2	0.3609
3	0.6709
4	0.8572
5	0.9441
6	0.9798.

Proof. Divide the $r = 4$ column of Definition 9462.1527 by $\binom{30}{m}$. □

Theorem 9462.1529 (Maximum non-trigger blockers)**.** *The largest external subsets that do not trigger the full cube have cardinality*

$$2(2^3 - 1) = 14.$$

There are four such maximal blockers, one for each omitted coordinate in $\{s_1, f_1, s_2, f_2\}$. Consequently every $B \subseteq Q_6 \setminus M^\star$ with $|B| \geq 15$ satisfies

$$\mathrm{Med}(M^\star \cup B) = Q_6.$$

Proof. A non-trigger set has rank at most 3. For fixed three-coordinate support J, the largest such set contains every nonempty quotient support contained in J, and each quotient support has two f_3-lifts. This gives $2(2^3 - 1) = 14$ codons. There are four choices of the missing coordinate. Any subset of size at least 15 cannot be contained in one of these blockers, so its support rank is 4. □

Theorem 9462.1530 (Minimal full triggers)**.** *At quotient level, the minimal full triggers have counts*

trigger size	*quotient count*
1	1
2	25
3	22
4	1
total	49.

At codon level, where each quotient support has two f_3-lifts, the corresponding counts are

trigger size	*codon count*
1	2
2	100
3	176
4	16
total	294.

Proof. A quotient-level minimal full trigger is a minimal set cover of the four coordinate set $\{s_1, f_1, s_2, f_2\}$ by nonempty supports. Enumerating the minimal covers of a four-element set gives $1, 25, 22, 1$ by trigger size. A codon-level trigger of size m has 2^m choices of f_3-lifts, giving the second table. □

Corollary 9462.1531 (Antipodal singleton)**.** *The only quotient-level one-point full trigger is* AGY*. Its two codon-level lifts are* AGU *and* AGC*.*

Proof. A one-point full trigger must have support $\{s_1, f_1, s_2, f_2\}$. The unique first-two-position quotient point with that support relative to CU is AG. □

Theorem 9462.1532 (Rank-spectral trigger lattice theorem)**.** *After $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, every median closure, closure size, cubical f-vector, killed-walk spectral radius, and full-cube threshold is controlled by the Boolean trigger rank on $\{s_1, f_1, s_2, f_2\}$. The full cube appears exactly at rank 4, the largest non-trigger external set has size 14, and* AGY *is the unique quotient-level singleton full trigger.*

Proof. The closure classification is Definition 9462.1521. The rank size and full-cube criterion are Definition 9462.1522. The cubical and spectral rank data are Definition 9462.1524 and Definition 9462.1525. The forced threshold is Definition 9462.1529, and the singleton classification is Definition 9462.1531. □

Concretized in §3704: BooleanTriggerLatticeUp realizes the Boolean trigger rank, meet, join, observed table, and support rows as a finite NameCert packet.

Definition 9462.1533 (Random trigger time)**.** *Let $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$. Draw codons without replacement from the 30 codons of $Q_6 \setminus M^\star$, and write*

$$B_m = \{p_1, \ldots, p_m\}.$$

The full-cube trigger time is

$$T = \min\{m : \mathrm{Med}(M^\star \cup B_m) = Q_6\}.$$

Equivalently, T is the first time at which the anchor supports of B_m cover all four coordinates s_1, f_1, s_2, f_2.

Theorem 9462.1534 (Random rank distribution)**.** *For m external codons and trigger rank d, the number of m-element subsets with rank exactly d is*

$$N(m, d) = \binom{4}{d} \sum_{j=0}^{d} (-1)^j \binom{d}{j} \binom{2(2^{d-j} - 1)}{m}.$$

Thus

$$\Pr(r(B_m) = d) = \frac{N(m, d)}{\binom{30}{m}}.$$

Proof. This is the set-cover count of Definition 9462.1527, normalized by the number of m-codon subsets of the external set. □

Theorem 9462.1535 (Full-trigger distribution)**.** *The cumulative full-trigger probabilities are*

$$\Pr(T \le m) = \Pr(r(B_m) = 4) = \frac{N(m,4)}{\binom{30}{m}}.$$

For $m \le 10$:

m	$\Pr(T \le m)$
1	0.0667
2	0.3609
3	0.6709
4	0.8572
5	0.9441
6	0.9798
7	0.9933
8	0.9979
9	0.9994
10	0.9999.

Proof. Full triggering is the rank-four event by Definition 9462.1522. Substitute $d = 4$ into Definition 9462.1534. □

Concretized in §4217: Random trigger thresholds are realized as finite trigger-rank NameCert data.

Theorem 9462.1536 (Trigger time law)**.** *The trigger time satisfies*

$$\Pr(T = m) = \Pr(T \le m) - \Pr(T \le m - 1).$$

Its first values, together with the conditional trigger rate $h_m = \Pr(T = m \mid T \ge m)$, are

m	$\Pr(T = m)$	h_m
1	0.0667	0.0667
2	0.2943	0.3153
3	0.3100	0.4851
4	0.1862	0.5660
5	0.0869	0.6083
6	0.0357	0.6386
7	0.0135	0.6665
8	0.0047	0.6957
9	0.0015	0.7273
10	0.0004	0.7619.

The mode is $T = 3$.

Proof. The probability mass formula follows by differencing the cumulative distribution in Definition 9462.1535. The hazard is the same mass divided by survival before the mth draw. □

Corollary 9462.1537 (Mean and quantiles)**.** *The expected full-cube trigger time is*

$$\mathbb{E}[T] = 3.12998.$$

The median, 90%*,* 95%*, and* 99% *thresholds are*

quantile	*codons*
50%	3
90%	5
95%	6
99%	7.

Proof. Use the finite identity

$$\mathbb{E}[T] = \sum_{m \geq 0} \Pr(T > m),$$

with the deterministic cutoff supplied by Definition 9462.1529. The quantiles are read from the cumulative distribution. □

Theorem 9462.1538 (Expected rank, closure size, and spectrum)**.** *After m random external codons, the expected trigger rank is*

$$\mathbb{E}[r_m] = 4\left(1 - \frac{\binom{14}{m}}{\binom{30}{m}}\right).$$

For $m \leq 7$:

m	$\mathbb{E}[r_m]$
1	2.1333
2	3.1632
3	3.6414
4	3.8539
5	3.9438
6	3.9798
7	3.9933.

The corresponding expected closure sizes and expected spectral radii are

m	$\mathbb{E}\lvert \mathrm{Med}(M^\star \cup B_m)\rvert$	$\mathbb{E}[\lambda]$
0	34.000	0.836111
1	42.667	0.873927
2	52.230	0.928720
3	58.499	0.966502
4	61.689	0.985904
5	63.103	0.994527
6	63.676	0.998025
7	63.892	0.999342
8	63.967	0.999800
9	63.991	0.999945
10	63.998	0.999987.

Proof. A fixed coordinate is absent precisely when all m codons lie in the fourteen external codons whose supports do not contain that coordinate. Linearity of expectation gives the displayed rank formula. The expected closure size and spectral radius are obtained by summing the rank-size and rank-spectral tables against the rank distribution. □

Theorem 9462.1539 (Random trigger threshold theorem)**.** *For the random external process beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, the average full-cube trigger time is* 3.12998 *codons. The deterministic worst-case threshold is* 15 *codons, while the median and* 99% *random thresholds are* 3 *and* 7 *codons respectively.*

Proof. The mean and quantiles are Definition 9462.1537. The deterministic threshold is Definition 9462.1529. □

Definition 9462.1540 (Rank Markov state)**.** *Let $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, and draw external codons without replacement from $Q_6 \setminus M^\star$. For $B_m = \{p_1, \ldots, p_m\}$, put*

$$J_m = \bigcup_{p \in B_m} \sigma(p), \qquad r_m = |J_m|.$$

The state of the random trigger process is the pair (m, r_m).

Theorem 9462.1541 (Rank Markov transition)**.** *If the current state is (m, r) with $r < 4$, then the next rank change $\ell = r_{m+1} - r_m$ has probabilities*

$$\Pr(\ell = 0 \mid m, r) = \frac{2(2^r - 1) - m}{30 - m},$$

and, for $1 \le \ell \le 4 - r$,

$$\Pr(\ell \mid m, r) = \frac{2 \cdot 2^r \binom{4-r}{\ell}}{30 - m}.$$

Thus (m, r_m) is a finite Markov process.

Proof. Given a covered coordinate set J of size r, the external codons whose support is contained in J number $2(2^r - 1)$. Since all m drawn codons are already contained in this class when the rank does not increase, the remaining non-increasing codons number $2(2^r - 1) - m$. If exactly ℓ fresh coordinates are added, choose those ℓ from the $4 - r$ uncovered coordinates, choose any subset of the r covered coordinates, and choose one of the two f_3-lifts. Dividing by the $30 - m$ remaining external codons gives the formula. □

Corollary 9462.1542 (Rank drift)**.** *The one-step expected rank change is*

$$\mathbb{E}[r_{m+1} - r_m \mid m, r] = \frac{16(4 - r)}{30 - m}.$$

Proof. Using Definition 9462.1541,

$$\mathbb{E}[\Delta r \mid m, r] = \sum_{\ell=1}^{4-r} \ell \frac{2 \cdot 2^r \binom{4-r}{\ell}}{30 - m}.$$

The binomial identity

$$\sum_{\ell=1}^{n} \ell \binom{n}{\ell} = n 2^{n-1}$$

with $n = 4 - r$ gives the result. □

Corollary 9462.1543 (Direct trigger probability)**.** *For $r < 4$,*

$$\Pr(r_{m+1} = 4 \mid m, r) = \frac{2^{r+1}}{30 - m}.$$

In particular, from state $(3, 3)$ the next draw triggers the full cube with probability $16/27 = 0.5926$.

Proof. Direct triggering is the change $\ell = 4 - r$ in Definition 9462.1541. □

Theorem 9462.1544 (Markov trigger moments)**.** *Let* $T = \min\{m : r_m = 4\}$*. The Markov recursion gives*

$$\mathbb{E}[T] = 3.12998, \qquad \mathrm{Var}(T) = 1.81588, \qquad \mathrm{sd}(T) = 1.34755.$$

The first trigger-time probabilities are the same as in Definition 9462.1536; in particular, the mode is $T = 3$*.*

Proof. Let $E(m, r)$ be the expected remaining number of draws until rank 4. Then $E(m, 4) = 0$ and

$$E(m, r) = 1 + \sum_{r'} \Pr(r' \mid m, r) E(m + 1, r').$$

The second moment satisfies the analogous recursion for $(1+T')^2$. Applying these finite recursions from $(0, 0)$ gives the displayed values. □

Theorem 9462.1545 (Conditional remaining trigger time)**.** *The expected remaining draw counts at representative states are*

state	$E(m, r)$
$(0, 0)$	3.130
$(1, 1)$	2.729
$(1, 2)$	2.329
$(1, 3)$	1.765
$(2, 1)$	2.638
$(2, 2)$	2.252
$(2, 3)$	1.706
$(3, 2)$	2.174
$(3, 3)$	1.647
$(6, 2)$	1.941
$(6, 3)$	1.471.

Proof. These are the values of the finite recursion in Definition 9462.1544. They depend on both the number of consumed external codons and the currently covered coordinate rank. □

Theorem 9462.1546 (Markov trigger theorem)**.** *Beyond* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$*, random median percolation is exactly the four-coordinate set-cover Markov process on states* (m, r)*. Its transition law is Definition 9462.1541, its drift is Definition 9462.1542, and its full-trigger time has mean* 3.12998 *with deterministic worst-case threshold* 15*.*

Proof. The equivalence between full median closure and rank 4 is Definition 9462.1522. The transition law follows from support counting, and the threshold 15 is Definition 9462.1529. □

Concretized in §5792: Finite rank-trigger Markov dynamics are realized as BHist transition-count and trigger-time NameCert rows.

Concretized in §2245: The finite codon-transition matrix packet realizes the Markov-trigger transition surface as BEDC NameCert data.

Definition 9462.1547 (Blocker complex)**.** *Let* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ *and let*

$$\Omega = \{s_1, f_1, s_2, f_2\}.$$

For an external set $B \subseteq Q_6 \setminus M^\star$*, define* $J(B) = \bigcup_{p \in B} \sigma(p)$*. The blocker complex is*

$$\mathcal{B} = \{B \subseteq Q_6 \setminus M^\star : J(B) \subsetneq \Omega\}.$$

It is a simplicial complex because every subset of a non-trigger set is again non-trigger.

Theorem 9462.1548 (Maximal blockers)**.** *The maximal faces of* $\mathcal{B}$ *are the four blockers obtained by omitting one trigger coordinate:*

$$B_{\neg s_1}, \quad B_{\neg f_1}, \quad B_{\neg s_2}, \quad B_{\neg f_2}.$$

Each has 14 *vertices. In IUPAC notation they are*

$$\mathrm{YNY} \setminus \mathrm{CUY}, \quad \mathrm{SNY} \setminus \mathrm{CUY}, \quad \mathrm{NYY} \setminus \mathrm{CUY}, \quad \mathrm{NWY} \setminus \mathrm{CUY}.$$

Proof. If coordinate i is absent, every support lies in $\Omega \setminus \{i\}$. The nonempty supports inside the remaining three coordinates number $2^3 - 1$, and each has two f_3-lifts. Thus each maximal blocker has $2(2^3 - 1) = 14$ codons. The four displayed motifs express the four possible missing coordinates relative to the anchor CUY. □

Corollary 9462.1549 (Forcing threshold)**.** *The largest non-trigger external sets have size* 14*. Hence every external set* B *with* $|B| \geq 15$ *satisfies*

$$\mathrm{Med}(M^\star \cup B) = Q_6.$$

Proof. Every blocker is contained in one of the four maximal blockers of Definition 9462.1548. □

Theorem 9462.1550 (Blocker intersections)**.** *If* ℓ *trigger coordinates are simultaneously omitted, then the corresponding maximal-blocker intersection has size*

$$2(2^{4-\ell} - 1).$$

Consequently the intersection sizes are

ℓ	*intersection size*
1	14
2	6
3	2
4	0.

Proof. After omitting a set L of ℓ coordinates, the allowed nonempty supports are precisely the nonempty subsets of $\Omega \setminus L$. There are $2^{4-\ell} - 1$ such supports, each with two f_3-lifts. □

Theorem 9462.1551 (Blocker nerve)**.** *The nerve of the four-simplex cover of* $\mathcal{B}$ *by its maximal blockers is the tetrahedron boundary* $\partial\Delta^3$*.*

Proof. Every one-, two-, and three-fold intersection of maximal blockers is nonempty by Definition 9462.1550, while the four-fold intersection is empty. The nerve therefore has four vertices, six edges, four triangular faces, and no three-simplex. □

Corollary 9462.1552 (Blocker homotopy type)**.** *The blocker complex satisfies*

$$\mathcal{B} \simeq S^2.$$

Hence its only nonzero reduced homology is

$$\widetilde{H}_2(\mathcal{B}) \cong \mathbb{Z}.$$

Proof. Each maximal blocker and every nonempty intersection of maximal blockers is a simplex, hence contractible. The nerve theorem applies, and the nerve is $\partial\Delta^3 \cong S^2$ by Definition 9462.1551. □

Theorem 9462.1553 (Blocker f-vector)**.** *For $m \geq 1$, the number of m-codon blocker faces is*

$$F_m = 4\binom{14}{m} - 6\binom{6}{m} + 4\binom{2}{m}.$$

Thus the blocker-complex f-vector is

$$(28, 278, 1336, 3914, 7972, 12006, 13728, 12012, 8008, 4004, 1456, 364, 56, 4),$$

and its Euler characteristic is 2*.*

Proof. Use inclusion-exclusion over the four maximal blockers. Single maximal blockers have size 14, pairwise intersections have size 6, triple intersections have size 2, and the four-fold intersection is empty. The alternating sum of the displayed f-vector is 2. □

Corollary 9462.1554 (Probability tail from blockers)**.** *For a uniformly chosen m-codon external subset,*

$$\Pr(T > m) = \frac{F_m}{\binom{30}{m}},$$

and hence

$$\Pr(T \leq m) = 1 - \frac{4\binom{14}{m} - 6\binom{6}{m} + 4\binom{2}{m}}{\binom{30}{m}}.$$

Proof. The event $T > m$ is exactly the event that the chosen subset is a face of $\mathcal{B}$. Count those faces by Definition 9462.1553. □

Theorem 9462.1555 (Trigger-blocker duality theorem)**.** *Beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, full triggering is the set-cover condition $J(B) = \Omega$, while non-triggering is the blocker complex $\mathcal{B} \simeq S^2$. The minimal full triggers are the minimal nonfaces of $\mathcal{B}$, and there are* 294 *such nonfaces at codon level.*

Proof. The trigger criterion is Definition 9462.1522. The blocker homotopy type is Definition 9462.1552. Minimal full triggers are precisely minimal subsets not belonging to $\mathcal{B}$, and their codon-level count is Definition 9462.1530. □

Concretized in §3801: The trigger-blocker duality reading is realized as finite blocker-complex, maximal-blocker, intersection, nerve, transport, provenance, and NameCert rows.

Definition 9462.1556 (Trigger hypergraph)**.** *Let* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ *and let*

$$E = Q_6 \setminus M^\star, \qquad \Omega = \{s_1, f_1, s_2, f_2\}.$$

The post-$M^\star$ *trigger hypergraph has vertex set* E *and edge set*

$$\mathcal{T} = \{T \subseteq E : T \text{ is minimal with } J(T) = \Omega\}.$$

Equivalently, its edges are the minimal full triggers of Definition 9462.1530.

Theorem 9462.1557 (Trigger hypergraph as nonface data)**.** *The trigger hypergraph* $\mathcal{T}$ *is the set of minimal nonfaces of the blocker complex* $\mathcal{B}$*.*

Proof. By Definition 9462.1547, a subset $B \subseteq E$ is a face of $\mathcal{B}$ exactly when $J(B) \subsetneq \Omega$. Thus a minimal subset not belonging to $\mathcal{B}$ is exactly a minimal subset with $J(B) = \Omega$, which is the defining condition for an edge of $\mathcal{T}$. □

Theorem 9462.1558 (Minimal-trigger polynomials)**.** *The quotient-level minimal-trigger generating polynomial is*

$$T_q(z) = z + 25z^2 + 22z^3 + z^4.$$

At codon level it is

$$T_c(z) = 2z + 100z^2 + 176z^3 + 16z^4.$$

Proof. The quotient coefficients are the four rows of Definition 9462.1530. At codon level, each quotient support has two f_3-lifts, so a trigger of size m contributes a factor 2^m. □

Theorem 9462.1559 (Minimal-trigger support types)**.** *At quotient level, the minimal full triggers have the following support-size types:*

trigger size	*support-size type*	*quotient count*	*codon count*
1	(4)	1	2
2	$(1,3)$	4	16
2	$(2,2)$	3	12
2	$(2,3)$	12	48
2	$(3,3)$	6	24
3	$(1,1,2)$	6	48
3	$(1,2,2)$	12	96
3	$(2,2,2)$	4	32
4	$(1,1,1,1)$	1	16.

Proof. Enumerate all minimal set covers of the four-coordinate set Ω by nonempty supports and sort each cover by the cardinalities of its members. The codon counts again multiply the quotient counts by 2^m for trigger size m. □

Theorem 9462.1560 (Bernoulli reliability polynomial)**.** *Suppose each external codon in* E *is independently adjoined with probability* p*, and put* $q = 1 - p$*. The probability of full-cube triggering is*

$$R(p) = 1 - 4q^{16} + 6q^{24} - 4q^{28} + q^{30}.$$

Proof. Failure to trigger is the event that at least one coordinate of Ω is absent. For one fixed coordinate, the 16 codons whose support contains that coordinate must all be omitted, giving q^{16}. For two fixed coordinates to be absent, the 24 codons containing at least one of them must be omitted, giving q^{24}. For three fixed coordinates, the exponent is 28, and for all four it is 30. Inclusion-exclusion over the four missing-coordinate events therefore gives

$$\Pr(\text{non-trigger}) = 4q^{16} - 6q^{24} + 4q^{28} - q^{30},$$

and $R(p)$ is its complement. □

Corollary 9462.1561 (Reliability thresholds)**.** *Solving $R(p) = \alpha$ gives*

α	p_α	$30p_\alpha$
0.50	0.08779	2.63
0.90	0.19455	5.84
0.95	0.23206	6.96
0.99	0.30934	9.28.

Thus an independent activation rate of about 8.779% *already gives a half-probability full trigger.*

Proof. The entries are the unique roots in $[0, 1]$ of the equation in Definition 9462.1560. The final column is the expected number of selected external codons. □

Corollary 9462.1562 (Fixed-size and Bernoulli blocker probabilities)**.** *The fixed-size non-trigger probability and the Bernoulli non-trigger probability are two evaluations of the same blocker complex:*

$$\Pr(\textit{non-trigger after } m \textit{ external codons}) = \frac{4\binom{14}{m} - 6\binom{6}{m} + 4\binom{2}{m} - \binom{0}{m}}{\binom{30}{m}},$$

with the final term only affecting $m = 0$, while

$$\Pr(\textit{Bernoulli non-trigger}) = 4q^{16} - 6q^{24} + 4q^{28} - q^{30}.$$

Proof. The first formula is the blocker f-vector count of Definition 9462.1553 divided by the total number of m-subsets, with the empty four-fold intersection included for the empty face. The second is the missing-coordinate inclusion-exclusion used in Definition 9462.1560. □

Theorem 9462.1563 (Stanley-Reisner trigger ideal)**.** *Let x_p be a squarefree variable for each $p \in E$. The Stanley-Reisner ideal of $\mathcal{B}$ is*

$$I_{\mathcal{B}} = \left\langle \prod_{p \in T} x_p : T \in \mathcal{T} \right\rangle.$$

It has 294 *minimal squarefree monomial generators, with degree distribution*

degree	*minimal generators*
1	2
2	100
3	176
4	16.

The degree-one generators are x_{AGU} and x_{AGC}.

Proof. The Stanley-Reisner ideal is generated by the minimal nonfaces of the complex. By Definition 9462.1557, these are precisely the minimal full triggers. The degree distribution is the codon-level table in Definition 9462.1558, and the degree-one generators are the two codon lifts of the quotient support AGY from Definition 9462.1531. □

Theorem 9462.1564 (Trigger hypergraph and reliability theorem)**.** *Beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, the full-cube trigger problem is equivalently a four-coordinate set-cover hypergraph, an S^2 blocker complex, a 294-edge minimal-nonface system, and the Bernoulli reliability polynomial*

$$R(p) = 1 - 4(1-p)^{16} + 6(1-p)^{24} - 4(1-p)^{28} + (1-p)^{30}.$$

Proof. The set-cover criterion is Definition 9462.1522. The blocker homotopy type is Definition 9462.1552. The minimal nonface identification is Definition 9462.1557, with the codon-level count from Definition 9462.1558. The reliability polynomial is Definition 9462.1560. □

Concretized in §3657: Finite trigger-edge reliability rows are realized as a NameCert packet.

Definition 9462.1565 (Coverage rank)**.** *Let $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ and $E = Q_6 \setminus M^\star$. For $p \in E$, let $\sigma(p) \subseteq \Omega$ be its anchor support, where*

$$\Omega = \{s_1, f_1, s_2, f_2\}.$$

For $B \subseteq E$, put

$$J(B) = \bigcup_{p \in B} \sigma(p), \qquad r(B) = |J(B)|.$$

The function r is the coverage rank of the external set B.

Theorem 9462.1566 (Coverage-closure formula)**.** *For every $B \subseteq E$,*

$$\mathrm{Med}(M^\star \cup B) = M^\star \cup \{p \in E : \sigma(p) \subseteq J(B)\}.$$

Equivalently,

$$\mathrm{Med}(M^\star \cup B) = \mathrm{NNR} \cup (Q_{J(B)}(\mathrm{CU}) \times Y).$$

Proof. This is the anchor-subcube formula of Definition 9462.1521 written on the external vertex set E. The complete R halfspace supplies every coordinate choice needed to median-generate the Y-side subcube based at CU with free coordinate set $J(B)$. The external vertices in that subcube are exactly those whose supports are contained in $J(B)$. □

Concretized in §3734: Coverage-rank algebra is realized as finite boundary-tile coverage, rank, trigger, classifier, and NameCert rows.

Corollary 9462.1567 (Rank size formula)**.** *If $r(B) = r$, then*

$$|\mathrm{Med}(M^\star \cup B)| = 34 + 2(2^r - 1) = 32 + 2^{r+1}.$$

Thus

r	$\lvert \mathrm{cl}_E(B) \rvert$	$\lvert \mathrm{Med}(M^\star \cup B) \rvert$
0	0	34
1	2	36
2	6	40
3	14	48
4	30	64.

Proof. For fixed $J \subseteq \Omega$ with $|J| = r$, the external codons closed under J have nonempty supports contained in J. There are $2^r - 1$ such quotient supports and two f_3-lifts for each. Adding the 34 codons already in $M^\star$ gives the displayed formula. □

Definition 9462.1568 (External trigger closure)**.** *For $B \subseteq E$, define*

$$\mathrm{cl}_E(B) = \{p \in E : \sigma(p) \subseteq J(B)\}.$$

Then

$$\mathrm{Med}(M^\star \cup B) = M^\star \cup \mathrm{cl}_E(B).$$

Theorem 9462.1569 (Closed-set lattice)**.** *The closed sets of cl_E are exactly*

$$E_J = \{p \in E : \sigma(p) \subseteq J\}, \qquad J \subseteq \Omega.$$

There are 16 closed sets, and

$$E_I \cap E_J = E_{I \cap J}, \qquad \mathrm{cl}_E(E_I \cup E_J) = E_{I \cup J}.$$

Thus the closed-set lattice is isomorphic to 2^Ω.

Proof. The operator cl_E depends on a subset only through the union support $J(B)$. If C is closed, then $C = \mathrm{cl}_E(C) = E_{J(C)}$. Conversely each E_J is fixed by cl_E. The two displayed identities follow directly from support containment. □

Theorem 9462.1570 (Coverage polymatroid rank)**.** *The function $r(B) = |J(B)|$ is monotone and submodular:*

$$A \subseteq B \Longrightarrow r(A) \leq r(B),$$

and

$$r(A) + r(B) \geq r(A \cup B) + r(A \cap B).$$

It is a coverage polymatroid rank rather than a matroid rank.

Proof. Write

$$r(B) = \sum_{i \in \Omega} \mathbf{1}[i \in J(B)].$$

Each coordinate indicator is monotone and submodular as a set function on E, and finite sums preserve these two properties. However AGU and AGC each have support Ω, so $r(\{\mathrm{AGU}\}) = r(\{\mathrm{AGC}\}) = 4$. A matroid rank function has singleton rank at most 1. □

Definition 9462.1571 (Rank enumerator)**.** *The coverage-rank enumerator is*

$$Z(t, u) = \sum_{B \subseteq E} t^{|B|} u^{r(B)}.$$

Theorem 9462.1572 (Rank-enumerator formula)**.** *The rank enumerator has the closed form*

$$Z(t, u) = \sum_{d=0}^{4} \binom{4}{d} u^d \sum_{k=0}^{d} (-1)^{d-k} \binom{d}{k} (1 + t)^{2(2^k - 1)}.$$

Proof. Fix a coordinate set J with $|J| = d$. The choices with $J(B) \subseteq J$ form a simplex on $2(2^d - 1)$ external codons, so their ordinary generating function is $(1+t)^{2(2^d-1)}$. Inclusion-exclusion over the coordinates of J extracts the choices with $J(B) = J$. Multiplying by $\binom{4}{d}$ and u^d gives the formula. □

Corollary 9462.1573 (Fixed-size rank counts)**.** *The coefficient of $t^m u^d$ in $Z(t,u)$ is*

$$N(m,d) = \binom{4}{d} \sum_{j=0}^{d} (-1)^j \binom{d}{j} \binom{2(2^{d-j}-1)}{m}.$$

For $m \leq 4$ this gives

m	$r=1$	$r=2$	$r=3$	$r=4$
1	8	12	8	2
2	4	78	196	157
3	0	120	1216	2724
4	0	90	3824	23491.

Proof. Extract the coefficient of $t^m u^d$ in Definition 9462.1572 and substitute $j = d - k$. □

Theorem 9462.1574 (Blocker independence polynomial)**.** *The independence polynomial of the non-trigger complex*

$$\mathcal{B} = \{B \subseteq E : r(B) < 4\}$$

is

$$I_{\mathcal{B}}(t) = 4(1+t)^{14} - 6(1+t)^6 + 4(1+t)^2 - 1.$$

Proof. The complex $\mathcal{B}$ is the union of four maximal blocker simplices, one for each omitted coordinate. Their one-, two-, and three-fold intersections have 14, 6, and 2 vertices respectively, while the four-fold intersection contains only the empty face. Applying inclusion-exclusion to the face generating functions gives the displayed polynomial. □

Corollary 9462.1575 (Reliability from blocker independence)**.** *For independent external activation probability p and $q = 1 - p$,*

$$\Pr(\textit{non-trigger}) = q^{30} I_{\mathcal{B}}\left(\frac{p}{q}\right) = 4q^{16} - 6q^{24} + 4q^{28} - q^{30}.$$

Hence

$$R(p) = 1 - 4q^{16} + 6q^{24} - 4q^{28} + q^{30}.$$

Proof. Weighting a blocker face of size m by $p^m q^{30-m}$ gives $q^{30} I_{\mathcal{B}}(p/q)$. Substituting the polynomial from Definition 9462.1574 and simplifying recovers the reliability polynomial of Definition 9462.1560. □

Theorem 9462.1576 (Coverage-rank algebra theorem)**.** *Beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, the codon median dynamics are completely controlled by the coverage-rank algebra on $\Omega = \{s_1, f_1, s_2, f_2\}$. The same support-union function determines the median closure, the closed-set lattice, the submodular rank, the rank enumerator, the blocker independence polynomial, the reliability polynomial, and the trigger ideal.*

Proof. The closure formula is Definition 9462.1566. The closed-set lattice is Definition 9462.1569, and the rank structure is Definition 9462.1570. The enumerator and blocker polynomial are Definition 9462.1572 and Definition 9462.1574. The reliability polynomial follows from Definition 9462.1575, and the trigger ideal is Definition 9462.1563. □

Definition 9462.1577 (Trigger energy)**.** *Let* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$ *and* $E = Q_6 \setminus M^\star$*. For an external codon* $p \in E$*, define*

$$e(p) = |\sigma(p)|.$$

For $B \subseteq E$*, define its trigger energy by*

$$\mathcal{E}(B) = \sum_{p \in B} |\sigma(p)|.$$

This is the total first-two-position Hamming distance from the anchor CUY*, counted over the selected external codons.*

Theorem 9462.1578 (Energy lower bound)**.** *If* $B \subseteq E$ *is a full trigger, then*

$$\mathcal{E}(B) \geq 4.$$

Proof. Full triggering means $J(B) = \Omega$, where $\Omega = \{s_1, f_1, s_2, f_2\}$. Hence each of the four trigger coordinates appears in at least one support $\sigma(p)$. Counting appearances of coordinates across the selected supports gives $\mathcal{E}(B) \geq |\Omega| = 4$. □

Theorem 9462.1579 (Partition criterion)**.** *For a full trigger* B*, equality*

$$\mathcal{E}(B) = 4$$

holds if and only if the family of supports

$$\{\sigma(p) : p \in B\}$$

is a set partition of Ω*.*

Proof. If $\mathcal{E}(B) = 4$, then the four coordinates of Ω are covered with exactly four total coordinate appearances. Since full triggering requires all four coordinates to appear, each coordinate appears exactly once; the supports are therefore pairwise disjoint and their union is Ω. Conversely, if the supports partition Ω, each coordinate is counted exactly once, so $\mathcal{E}(B) = 4$. □

Theorem 9462.1580 (Energy-minimal trigger counts)**.** *At quotient level, the full triggers of minimal energy are the set partitions of the four-coordinate set* Ω*. Thus there are* 15 *of them. By trigger size they are*

trigger size	*partition type*	*quotient count*	*codon count*
1	4	1	2
2	$1+3$ *or* $2+2$	7	28
3	$1+1+2$	6	48
4	$1+1+1+1$	1	16
total		15	94.

Proof. By Definition 9462.1579, energy-minimal full triggers correspond exactly to set partitions of a four-element set. The four partition types have counts $1, 7, 6, 1$. At codon level, a trigger of size m has 2^m independent f_3-lift choices. □

Theorem 9462.1581 (Minimal triggers by energy)**.** *The codon-level minimal full triggers have energy distribution*

$$94y^4 + 144y^5 + 56y^6.$$

Equivalently,

energy	*quotient count*	*codon count*
4	15	94
5	24	144
6	10	56.

Thus 94 *of the* 294 *codon-level minimal full triggers are energy-minimal.*

Proof. Combine the support-size type enumeration of Definition 9462.1559 by total support size. The energy-four rows are precisely the partition rows; the energy-five rows come from overlapping types $(2, 3)$ and $(1, 2, 2)$; the energy-six rows come from $(3, 3)$ and $(2, 2, 2)$. Multiplying each quotient trigger of size m by 2^m gives the codon counts. □

Theorem 9462.1582 (Bivariate minimal-trigger polynomial)**.** *Let*

$$T_q(z, y) = \sum_T z^{|T|} y^{\mathcal{E}(T)}$$

sum over quotient-level minimal full triggers. Then

$$T_q(z, y) = zy^4 + z^2(7y^4 + 12y^5 + 6y^6) + z^3(6y^4 + 12y^5 + 4y^6) + z^4y^4.$$

At codon level,

$$T_c(z, y) = 2zy^4 + z^2(28y^4 + 48y^5 + 24y^6) + z^3(48y^4 + 96y^5 + 32y^6) + 16z^4y^4.$$

Setting $y = 1$ *recovers*

$$T_c(z, 1) = 2z + 100z^2 + 176z^3 + 16z^4,$$

and setting $z = 1$ *gives*

$$T_c(1, y) = 94y^4 + 144y^5 + 56y^6.$$

Proof. The terms are obtained by grouping the minimal-trigger support types from Definition 9462.1559 by trigger size and total support size. Passing from quotient level to codon level multiplies every size-m term by 2^m. □

Corollary 9462.1583 (Energy and cardinality)**.** *Energy-minimal full triggers exist at every cardinality* $1, 2, 3, 4$*. The cardinality-one case is the antipodal trigger* AGY*, while the cardinality-four case consists of one unit-support codon in each trigger coordinate.*

Proof. These are the four partition types in Definition 9462.1580. The one-block partition is Ω itself, realized by AGY. The four-block partition is the decomposition of Ω into singleton coordinates. □

Theorem 9462.1584 (Trigger-energy theorem)**.** *Beyond* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$*, every full trigger has energy at least* 4*, and equality is exactly the set-partition case. There are* 15 *quotient-level and* 94 *codon-level energy-minimal full triggers, while the full minimal-trigger energy distribution at codon level is*

$$94y^4 + 144y^5 + 56y^6.$$

Proof. The lower bound is Definition 9462.1578, and the equality case is Definition 9462.1579. The enumerative statements are Definition 9462.1580 and Definition 9462.1581. □

Concretized in §3806: Trigger-energy partitions are realized as finite support, energy-ledger, partition, codon-lift, transport, replay, provenance, and NameCert rows.

Definition 9462.1585 (Energy-biased trigger weights)**.** *Let $E = Q_6 \setminus M^\star$ and let $e(p) = |\sigma(p)|$ be the trigger energy of $p \in E$. For parameters $t > 0$ and $0 < y \leq 1$, assign each external codon the weight*

$$w(p) = ty^{e(p)}.$$

For $B \subseteq E$, set

$$W(B) = t^{|B|} y^{\mathcal{E}(B)}.$$

The parameter t is the activity of external activation, while y penalizes distance from the anchor CUY*.*

Definition 9462.1586 (Energy partition factors)**.** *For $0 \leq d \leq 4$, define*

$$A_d(t, y) = \prod_{k=1}^{d} (1 + ty^k)^{2\binom{d}{k}}.$$

Thus

$$A_4(t, y) = (1 + ty)^8 (1 + ty^2)^{12} (1 + ty^3)^8 (1 + ty^4)^2$$

is the partition function of all external subsets.

Theorem 9462.1587 (Energy-biased blocker partition function)**.** *The partition function of non-trigger external sets is*

$$Z_{\text{block}}(t, y) = 4A_3(t, y) - 6A_2(t, y) + 4A_1(t, y) - 1.$$

Proof. Non-triggering means that at least one coordinate of $\Omega = \{s_1, f_1, s_2, f_2\}$ is absent. If a fixed set of $4 - d$ coordinates is absent, the allowed supports lie inside a fixed d-coordinate set, whose weighted partition function is $A_d(t, y)$. Inclusion-exclusion over the four omitted-coordinate events gives the displayed expression; the final -1 accounts for the empty four-fold intersection. □

Theorem 9462.1588 (Energy-biased reliability)**.** *The full-trigger reliability in the energy-biased model is*

$$R(t, y) = 1 - \frac{4A_3(t, y) - 6A_2(t, y) + 4A_1(t, y) - 1}{A_4(t, y)}.$$

Proof. The numerator in the fraction is the non-trigger partition function of Definition 9462.1587, and the denominator is the all-external partition function from Definition 9462.1586. Their ratio is the non-trigger probability under the product measure induced by the weights $w(p)$. □

Corollary 9462.1589 (Uniform reliability as a specialization)**.** *At $y = 1$,*

$$A_d(t, 1) = (1 + t)^{2(2^d - 1)}.$$

With $t = p/(1 - p)$, the energy-biased reliability specializes to

$$R(p) = 1 - 4(1 - p)^{16} + 6(1 - p)^{24} - 4(1 - p)^{28} + (1 - p)^{30}.$$

Proof. Substitute $y = 1$ in Definition 9462.1588. Multiplying the resulting blocker ratio by the common Bernoulli factor $(1-p)^{30}$ gives the polynomial of Definition 9462.1560. □

Corollary 9462.1590 (Low-energy expansion)**.** *As $y \to 0$, the full-trigger partition function satisfies*

$$Z_{\text{trig}}(t, y) = y^4(2t + 28t^2 + 48t^3 + 16t^4) + O(y^5).$$

Proof. By Definition 9462.1578, the lowest full-trigger energy is 4. By Definition 9462.1580, the energy-four codon-level triggers have counts $2, 28, 48, 16$ in cardinalities $1, 2, 3, 4$, respectively. □

Theorem 9462.1591 (Energy-biased threshold table)**.** *For target reliability α, let $t_\alpha(y)$ be the activity solving $R(t_\alpha(y), y) = \alpha$. The expected external-codon count at that activity is*

$$\mathbb{E}_{t,y}[|B|] = \sum_{k=1}^{4} 2\binom{4}{k} \frac{ty^k}{1 + ty^k}.$$

For $\alpha = 0.50$, the thresholds are

y	$t_{0.50}(y)$	$\mathbb{E}[\lvert B\rvert]$
1.00	0.0962	2.63
0.75	0.2008	3.01
0.50	0.5108	3.53
0.25	1.9830	4.23
0.10	8.5128	4.69.

For $\alpha = 0.90$, they are

y	$t_{0.90}(y)$	$\mathbb{E}[\lvert B\rvert]$
1.00	0.2415	5.84
0.75	0.4903	6.39
0.50	1.2207	7.04
0.25	4.7581	7.69
0.10	21.7497	7.80.

For $\alpha = 0.95$, they are

y	$t_{0.95}(y)$	$\mathbb{E}[\lvert B\rvert]$
1.00	0.3022	6.96
0.75	0.6114	7.55
0.50	1.5224	8.22
0.25	6.0003	8.80
0.10	28.2281	8.77.

Proof. The thresholds are the unique positive solutions of Definition 9462.1588. The expectation formula follows from the independent inclusion probability of an energy-k codon, namely $ty^k/(1 + ty^k)$, and the fact that there are $2\binom{4}{k}$ such codons. □

Corollary 9462.1592 (Energy-biased trigger regimes)**.** *At low activity, the cardinality-one term $2ty^4$ favors the two antipodal codon triggers* AGU *and* AGC*. At higher activity under energy penalty, the terms*

$$28t^2y^4, \qquad 48t^3y^4, \qquad 16t^4y^4$$

express the increasing contribution of partition-minimal multi-codon triggers.

Proof. These are the four terms in the low-energy expansion of Definition 9462.1590. The power of t records the trigger cardinality, while the common y^4 records minimal trigger energy. □

Theorem 9462.1593 (Energy-biased trigger reliability theorem)**.** *Beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, the Boolean trigger lattice admits the explicit two-parameter reliability function*

$$R(t,y) = 1 - \frac{4A_3(t,y) - 6A_2(t,y) + 4A_1(t,y) - 1}{A_4(t,y)}.$$

The specialization $y = 1$ recovers the uniform Bernoulli reliability, while the limit $y \to 0$ is governed by the 94 partition-minimal full triggers.

Proof. The formula is Definition 9462.1588. The uniform specialization is Definition 9462.1589, and the low-energy leading term is Definition 9462.1590. □

Concretized in §3583: Energy-biased trigger reliability is realized as finite weight, partition, blocker, and reliability NameCert rows.

Definition 9462.1594 (Conditional trigger ensemble)**.** *Let $E = Q_6 \setminus M^\star$ and weight external subsets by*

$$W(B) = t^{|B|} y^{\mathcal{E}(B)}.$$

The full-trigger partition function is

$$Z_{\mathrm{trig}}(t,y) = Z_{\mathrm{all}}(t,y) - Z_{\mathrm{block}}(t,y).$$

The conditional trigger ensemble is the probability distribution on full-trigger subsets defined by

$$\Pr(B \mid \mathrm{trig}) = \frac{t^{|B|} y^{\mathcal{E}(B)}}{Z_{\mathrm{trig}}(t,y)}.$$

Theorem 9462.1595 (Conditional trigger derivatives)**.** *In the conditional trigger ensemble,*

$$\mathbb{E}[|B| \mid \mathrm{trig}] = t \frac{\partial}{\partial t} \log Z_{\mathrm{trig}}(t,y),$$

and

$$\mathbb{E}[\mathcal{E}(B) \mid \mathrm{trig}] = y \frac{\partial}{\partial y} \log Z_{\mathrm{trig}}(t,y).$$

Proof. Differentiating Z_{trig} with $t\partial_t$ multiplies each monomial $t^{|B|} y^{\mathcal{E}(B)}$ by $|B|$. Dividing by Z_{trig} gives the conditional expectation. The same argument with $y\partial_y$ gives the conditional expected energy. □

Theorem 9462.1596 (Conditional means at half reliability)**.** *At the activity $t_{0.50}(y)$ satisfying $R(t,y) = 0.50$, the conditional means are*

y	$t_{0.50}(y)$	$\mathbb{E}[\lvert B\rvert \mid \mathrm{trig}]$	$\mathbb{E}[\mathcal{E}(B) \mid \mathrm{trig}]$
1.00	0.0962	3.596	8.184
0.75	0.2008	4.013	8.294
0.50	0.5108	4.580	8.308
0.25	1.9830	5.318	8.008
0.10	8.5128	5.751	7.296.

Proof. The activity values solve Definition 9462.1588. The conditional means are then evaluated by Definition 9462.1595. □

Theorem 9462.1597 (Conditional means at high reliability)**.** *At the activity $t_{0.90}(y)$ satisfying $R(t, y) = 0.90$, the conditional means are*

y	$t_{0.90}(y)$	$\mathbb{E}[\lvert B\rvert \mid \text{trig}]$	$\mathbb{E}[\mathcal{E}(B) \mid \text{trig}]$
1.00	0.2415	6.144	13.237
0.75	0.4903	6.699	13.271
0.50	1.2207	7.354	13.060
0.25	4.7581	7.989	12.201
0.10	21.7497	8.070	10.710.

Proof. This is the same derivative evaluation as Definition 9462.1596, now at the reliability level 0.90. □

Theorem 9462.1598 (Low-energy conditional distribution)**.** *As $y \to 0$, the conditional trigger ensemble is supported on partition-minimal triggers. If*

$$P(t) = 2t + 28t^2 + 48t^3 + 16t^4,$$

then the limiting cardinality distribution is

$$\Pr_{\min}(|B| = m) = \frac{a_m t^m}{P(t)},$$

where

$$a_1 = 2, \qquad a_2 = 28, \qquad a_3 = 48, \qquad a_4 = 16.$$

The limiting conditional mean cardinality is

$$\mathbb{E}_{\min}[|B|] = \frac{2t + 56t^2 + 144t^3 + 64t^4}{2t + 28t^2 + 48t^3 + 16t^4}.$$

Proof. By Definition 9462.1590, the lowest full-trigger energy is 4, with cardinality counts $2, 28, 48, 16$. Dividing the four leading terms by their sum gives the limiting conditional distribution. □

Theorem 9462.1599 (Low-energy crossover table)**.** *In the low-energy limiting ensemble, the cardinality probabilities and mean size are*

t	Pr(1)	Pr(2)	Pr(3)	Pr(4)	$\mathbb{E}_{\min}[\lvert B\rvert]$
0.10	0.3776	0.5287	0.0906	0.0030	1.719
0.25	0.1633	0.5714	0.2449	0.0204	2.122
0.50	0.0667	0.4667	0.4000	0.0667	2.467
1.00	0.0213	0.2979	0.5106	0.1702	2.830
2.00	0.0053	0.1481	0.5079	0.3386	3.180
5.00	0.0006	0.0419	0.3591	0.5984	3.555
10.00	0.0001	0.0133	0.2277	0.7589	3.745.

Proof. Substitute each value of t into Definition 9462.1598. □

Corollary 9462.1600 (Activity crossover points)**.** *The adjacent leading terms in the low-energy ensemble cross at*

$$t = \frac{1}{14}, \qquad t = \frac{7}{12}, \qquad t = 3.$$

Thus the dominant low-energy trigger type passes from singleton antipodal triggers, to two-codon partition triggers, to three-codon partition triggers, and finally to four unit-direction triggers.

Proof. The three equalities are

$$2t = 28t^2, \qquad 28t^2 = 48t^3, \qquad 48t^3 = 16t^4.$$

Solving them gives the displayed activities. □

Theorem 9462.1601 (Conditional trigger ensemble theorem)**.** *Beyond* $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$*, the full-trigger event has an explicit conditional ensemble. Its conditional mean cardinality and energy are logarithmic derivatives of* $Z_{\mathrm{trig}}(t, y)$*, and its low-energy limit is governed by the partition-minimal trigger polynomial*

$$2t + 28t^2 + 48t^3 + 16t^4.$$

Proof. The derivative formulas are Definition 9462.1595. The low-energy limiting distribution is Definition 9462.1598, and the activity crossovers are Definition 9462.1600. □

Concretized in §3870: The conditional trigger ensemble is realized as finite weighted trigger, blocker, partition, expectation, transport, replay, provenance, and NameCert rows.

Concretized in §7816: The finite trigger-cube packet realizes the conditional trigger ensemble as finite cube, trigger, blocker, energy, rank, and NameCert rows.

Definition 9462.1602 (Alexander-dual coordinate monomials)**.** *Let* $R_0 = k[x_p : p \in E]$ *be the squarefree polynomial ring on the external codons. For each trigger coordinate* $i \in \Omega$*, define*

$$U_i = \prod_{\sigma(p) \ni i} x_p.$$

Since a fixed coordinate lies in 2^3 *quotient supports and each support has two* f_3*-lifts, every* U_i *has degree* 16*.*

Theorem 9462.1603 (Alexander-dual blocker ideal)**.** *Let* $\mathcal{B}$ *be the blocker complex. Its Alexander-dual ideal is*

$$I_{\mathcal{B}}^{\vee} = \langle U_{s_1}, U_{f_1}, U_{s_2}, U_{f_2} \rangle.$$

Proof. The facets of $\mathcal{B}$ are the four maximal blockers $B_{\neg i} = \{p \in E : i \notin \sigma(p)\}$. Alexander duality sends each facet to its complement in E:

$$E \setminus B_{\neg i} = \{p \in E : i \in \sigma(p)\}.$$

The squarefree monomial on this complement is precisely U_i. □

Theorem 9462.1604 (LCM degrees)**.** *For* $S \subseteq \Omega$ *with* $|S| = k$*,*

$$\deg \operatorname{lcm}(U_i : i \in S) = 32 - 2^{5-k}.$$

Thus the lcm degrees are

$\lvert S\rvert$	*number of subsets*	*lcm degree*
1	4	16
2	6	24
3	4	28
4	1	30.

Proof. The lcm contains exactly those external codons whose support intersects S. Among the 15 nonempty quotient supports, the supports disjoint from S are the nonempty subsets of $\Omega \setminus S$, hence number $2^{4-k} - 1$. Therefore the quotient supports meeting S number $15 - (2^{4-k} - 1) = 16 - 2^{4-k}$, and each has two f_3-lifts. □

Theorem 9462.1605 (Minimal Taylor resolution)**.** *For*

$$J = I_{\mathcal{B}}^{\vee},$$

the Taylor resolution is minimal, with graded shifts

$$0 \to R_0(-30) \to R_0(-28)^4 \to R_0(-24)^6 \to R_0(-16)^4 \to J \to 0.$$

Equivalently, the quotient R_0/J has the resolution

$$0 \to R_0(-30) \to R_0(-28)^4 \to R_0(-24)^6 \to R_0(-16)^4 \to R_0 \to R_0/J \to 0.$$

Proof. The four generators have lcms whose degrees depend only on the size of the chosen coordinate subset and strictly increase with that size. Hence no Taylor differential can cancel equal multidegrees. The ranks of the Taylor modules are $\binom{4}{1}, \binom{4}{2}, \binom{4}{3}$ and $\binom{4}{4}$, and the shifts are those of Definition 9462.1604. □

Corollary 9462.1606 (Dual compression of the trigger hypergraph)**.** *The Stanley-Reisner ideal $I_{\mathcal{B}}$ has 294 minimal squarefree generators, but its Alexander dual has only four. The Betti rank pattern of the dual resolution is*

$$1,\ 4,\ 6,\ 4,\ 1.$$

Proof. The 294 generators of $I_{\mathcal{B}}$ are the minimal full triggers from Definition 9462.1563. The dual ideal has the four generators of Definition 9462.1603; its Taylor ranks are the binomial coefficients in four variables. □

Theorem 9462.1607 (Reliability from the lcm lattice)**.** *For $q = 1 - p$, the non-trigger reliability polynomial is*

$$P_{\text{non}}(p) = \sum_{\emptyset \neq S \subseteq \Omega} (-1)^{|S|+1} q^{\deg \operatorname{lcm}(U_i : i \in S)} = 4q^{16} - 6q^{24} + 4q^{28} - q^{30}.$$

Proof. The event U_i is absent is the event that coordinate i is missed. Inclusion-exclusion over the four missed-coordinate events uses exactly the lcm degrees of Definition 9462.1604. The coefficients are $4, -6, 4, -1$ according to subset size. □

Theorem 9462.1608 (Alexander-dual trigger algebra theorem)**.** *Beyond $M^\star = \mathrm{NNR} \cup \mathrm{CUY}$, the trigger side has 294 codon-level minimal full triggers, while the blocker-dual side is the four-generator squarefree monomial ideal*

$$I_{\mathcal{B}}^{\vee} = \langle U_{s_1}, U_{f_1}, U_{s_2}, U_{f_2} \rangle.$$

Its lcm degrees are $16, 24, 28, 30$*, its minimal Taylor ranks are* $1, 4, 6, 4, 1$*, and its lcm-lattice inclusion-exclusion is exactly*

$$4q^{16} - 6q^{24} + 4q^{28} - q^{30}.$$

Proof. The four-generator description is Definition 9462.1603. The shifts and Betti pattern are Definition 9462.1605. The probability polynomial is Definition 9462.1607. □

Concretized in §3555: finite Alexander-dual trigger/blocker NameCert packet

Definition 9462.1609 (Coverage-trigger system)**.** *For integers* $n \geq 1$ *and* $\ell \geq 1$*, let*

$$E_{n,\ell} = \{(S, a) : \varnothing \neq S \subseteq [n],\ a \in [\ell]\}.$$

The support of (S, a) *is* $\sigma(S, a) = S$*. Thus*

$$|E_{n,\ell}| = \ell(2^n - 1).$$

For $B \subseteq E_{n,\ell}$*, set*

$$J(B) = \bigcup_{e \in B} \sigma(e).$$

The subset B *is a full trigger when* $J(B) = [n]$*.*

Theorem 9462.1610 (Universal coverage closure)**.** *Define*

$$\mathrm{cl}(B) = \{e \in E_{n,\ell} : \sigma(e) \subseteq J(B)\}.$$

Then

$$\mathrm{cl}(B) = E_{J(B)}, \qquad E_J = \{e : \sigma(e) \subseteq J\}.$$

The closed-set lattice is isomorphic to $2^{[n]}$*, and if* $|J(B)| = r$*, then*

$$|\,\mathrm{cl}(B)| = \ell(2^r - 1).$$

Proof. The identity is the definition of cl with $J(B)$ substituted. Every $J \subseteq [n]$ gives a closed set E_J, and the operations satisfy

$$E_I \cap E_J = E_{I \cap J}, \qquad \mathrm{cl}(E_I \cup E_J) = E_{I \cup J}.$$

The size formula counts the nonempty supports contained in a fixed r-coordinate set and multiplies by ℓ lifts. □

Theorem 9462.1611 (Universal blocker topology)**.** *Let*

$$\mathcal{B}_{n,\ell} = \{B \subseteq E_{n,\ell} : J(B) \neq [n]\}.$$

Then

$$\mathcal{B}_{n,\ell} \simeq S^{n-2}.$$

The maximal blockers are

$$B_{\neg i} = \{e : i \notin \sigma(e)\}, \qquad i \in [n],$$

and have size $\ell(2^{n-1} - 1)$*. If* k *coordinates are omitted simultaneously, the corresponding intersection has size*

$$\ell(2^{n-k} - 1).$$

Proof. The complex is the union of the n simplices $B_{\neg i}$. Any proper intersection is a simplex on the elements whose supports avoid the omitted coordinates, hence contractible when at least one coordinate remains. The intersection of all n blockers is empty because supports are nonempty. The nerve is therefore $\partial\Delta^{n-1} \simeq S^{n-2}$. □

Theorem 9462.1612 (Universal blocker counts and threshold)**.** *The number of m-element non-trigger subsets is*

$$F_m(n,\ell) = \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} \binom{\ell(2^{n-k}-1)}{m}.$$

The largest blocker has size $\ell(2^{n-1}-1)$, so every subset of size at least

$$\ell(2^{n-1}-1)+1$$

is a full trigger.

Proof. Apply inclusion-exclusion to the n events that a coordinate is missed. If a specified set of k coordinates is missed, the available elements are precisely those whose supports are nonempty subsets of the remaining $n-k$ coordinates, with ℓ lifts. The threshold is one more than the largest maximal blocker size. □

Theorem 9462.1613 (Universal Bernoulli reliability)**.** *If each element of $E_{n,\ell}$ is independently activated with probability p and $q = 1-p$, then*

$$R_{n,\ell}(p) = 1 - \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} q^{\ell(2^n - 2^{n-k})}.$$

Proof. If a specified set of k coordinates is missed, every element whose support intersects that set must be inactive. The number of such elements is

$$\ell(2^n-1) - \ell(2^{n-k}-1) = \ell(2^n - 2^{n-k}).$$

Inclusion-exclusion over missed coordinates gives the non-trigger probability; the reliability is its complement. □

Theorem 9462.1614 (Universal Markov rank transition)**.** *Draw elements without replacement. If m elements have been drawn and the current covered rank is r, then*

$$\Pr(\Delta r = 0 \mid m, r) = \frac{\ell(2^r-1)-m}{\ell(2^n-1)-m},$$

and for $1 \le j \le n-r$,

$$\Pr(\Delta r = j \mid m, r) = \frac{\ell 2^r \binom{n-r}{j}}{\ell(2^n-1)-m}.$$

Moreover,

$$\mathbb{E}[\Delta r \mid m, r] = \frac{\ell 2^{n-1}(n-r)}{\ell(2^n-1)-m}.$$

Proof. Rank-preserving elements have nonempty support contained in the current r-coordinate set; after m draws, $\ell(2^r-1)-m$ remain. To add exactly j coordinates, choose those j coordinates among the $n-r$ uncovered coordinates and choose any subset of the old r coordinates, giving $\ell 2^r\binom{n-r}{j}$ elements. The drift formula uses

$$\sum_{j=1}^{n-r} j\binom{n-r}{j} = (n-r)2^{n-r-1}.$$

□

Theorem 9462.1615 (Universal Alexander dual)**.** *For each $i \in [n]$, set*

$$U_i = \prod_{\sigma(e)\ni i} x_e.$$

Then

$$I^{\vee}_{\mathcal{B}_{n,\ell}} = \langle U_1, \dots, U_n\rangle.$$

The degree of U_i is $\ell 2^{n-1}$, and the lcm degree of any k generators is

$$D_k = \ell(2^n - 2^{n-k}).$$

Proof. The facets of $\mathcal{B}_{n,\ell}$ are the blockers $B_{\neg i}$. Their complements are the elements whose supports contain i, which give the monomials U_i. The lcm of k such generators contains exactly the elements whose supports meet the chosen k coordinates, whose number is $\ell(2^n-2^{n-k})$. □

Theorem 9462.1616 (Universal energy-minimal triggers)**.** *Let the energy of an element be $|\sigma(e)|$ and the energy of a trigger be the sum of element energies. Every full trigger has energy at least n, with equality exactly when its supports form a set partition of $[n]$. Therefore the quotient-level energy-minimal triggers are counted by the Bell number B_n, and by trigger size m by the Stirling number $S(n,m)$. With ℓ lifts, the energy-minimal trigger polynomial is*

$$\sum_{m=1}^{n} \ell^m S(n,m) z^m y^n.$$

Proof. Every coordinate of $[n]$ must be covered at least once, so total energy is at least n. Equality means that every coordinate is counted exactly once, which is precisely a set partition. Each block has ℓ lift choices, giving the factor ℓ^m for an m-block partition. □

Theorem 9462.1617 (Codon case as $(4,2)$)**.** *The post-$M^\star$ codon trigger system is the $(n,\ell) = (4,2)$ coverage-trigger system. Substituting $n=4$ and $\ell=2$ gives*

$$|E| = 30, \qquad \mathcal{B}_{4,2} \simeq S^2, \qquad \ell(2^3-1)+1 = 15,$$

$$R_{4,2}(p) = 1 - 4q^{16} + 6q^{24} - 4q^{28} + q^{30},$$

$$D_1, D_2, D_3, D_4 = 16, 24, 28, 30,$$

and

$$2zy^4 + 28z^2y^4 + 48z^3y^4 + 16z^4y^4$$

for the energy-minimal codon-level trigger polynomial.

Proof. The four coordinates are s_1, f_1, s_2, f_2 and the two lifts are the two third-position Y codons in each support class. Each formula is the specialization of the preceding universal theorems at $(n, \ell) = (4, 2)$. □

Theorem 9462.1618 (Universal coverage-trigger algebra theorem)**.** *For every $n \geq 1$ and $\ell \geq 1$, the system $E_{n,\ell}$ has Boolean closed-set lattice $2^{[n]}$, blocker complex homotopy type S^{n-2}, forcing threshold $\ell(2^{n-1} - 1) + 1$, Bernoulli reliability Definition 9462.1613, Markov rank transition Definition 9462.1614, Alexander-dual ideal $\langle U_1, \ldots, U_n \rangle$, and energy-minimal partition polynomial*

$$\sum_{m=1}^{n} \ell^m S(n, m) z^m y^n.$$

The codon terminal system after $M^\star$ is exactly the $(4, 2)$ instance.

Proof. This is the collection of Definition 9462.1610, Definition 9462.1611, Definition 9462.1612, Definition 9462.1613, Definition 9462.1614, Definition 9462.1615, and Definition 9462.1616, with the codon specialization supplied by Definition 9462.1617. □

Concretized in §3478: finite support-closure trigger algebra realized as a NameCert packet

Theorem 9462.1619 (Thickened tetrahedral blocker boundary)**.** *For the post-$M^\star$ blocker complex*

$$\mathcal{B} = \{B \subseteq E : \bigcup_{p \in B} \sigma(p) \subsetneq \Omega\},$$

one has

$$\mathcal{B} = B_{\neg s_1} \cup B_{\neg f_1} \cup B_{\neg s_2} \cup B_{\neg f_2},$$

where $B_{\neg i} = \{p \in E : i \notin \sigma(p)\}$. The intersection profile is

intersection type	*multiplicity*	*vertices*	*simplex dimension*
facet	4	14	13
pair	6	6	5
triple	4	2	1
quadruple	1	0	−1.

Thus the nerve is $\partial\Delta^3$.

Proof. Missing one trigger coordinate leaves the nonempty supports inside a three-coordinate set, hence $2(2^3-1) = 14$ codons. Missing two coordinates leaves $2(2^2-1) = 6$ codons, and missing three coordinates leaves $2(2^1 - 1) = 2$ codons. Missing all four trigger coordinates leaves no nonempty support. These are exactly the intersection numbers of the four maximal blockers, so their nerve is the tetrahedral boundary. □

Concretized in §3932: The Hilbert-series blocker complex is realized as a finite NameCert packet over blocker-cover, face-polynomial, and Hilbert-row data.

Corollary 9462.1620 (Dimension and homotopy type)**.** *The blocker complex satisfies*

$$\mathcal{B} \simeq S^2, \qquad \dim \mathcal{B} = 13, \qquad \widetilde{H}_2(\mathcal{B}; k) \cong k.$$

Proof. All nonempty members of the cover in Definition 9462.1619 and all their nonempty intersections are simplices. The nerve theorem gives $\mathcal{B} \simeq \partial\Delta^3 \simeq S^2$. Since every maximal blocker has 14 vertices, the facets of $\mathcal{B}$ are 13-simplices. □

Theorem 9462.1621 (Blocker face polynomial)**.** *The face polynomial of $\mathcal{B}$ is*

$$I_{\mathcal{B}}(t) = \sum_{B \in \mathcal{B}} t^{|B|} = 4(1+t)^{14} - 6(1+t)^6 + 4(1+t)^2 - 1.$$

Its positive-dimensional face counts are

$$(28, 278, 1336, 3914, 7972, 12006, 13728, 12012, 8008, 4004, 1456, 364, 56, 4).$$

The Euler characteristic is 2*.*

Proof. The four facets contribute $4(1+t)^{14}$. Their six pairwise intersections contribute $6(1+t)^6$ and are subtracted. Their four triple intersections contribute $4(1+t)^2$ and are added. The quadruple intersection is empty, whose face polynomial is 1, and is subtracted. Expanding gives the stated face counts, and their alternating sum is 2. □

Theorem 9462.1622 (Hilbert series of the blocker ring)**.** *Let $R_0 = k[x_p : p \in E]$ and let $k[\mathcal{B}] = R_0/I_{\mathcal{B}}$. Then*

$$H_{k[\mathcal{B}]}(z) = \frac{4}{(1-z)^{14}} - \frac{6}{(1-z)^6} + \frac{4}{(1-z)^2} - 1.$$

Proof. For a simplex on m vertices,

$$\sum_F \left(\frac{z}{1-z}\right)^{|F|} = \left(1 + \frac{z}{1-z}\right)^m = \frac{1}{(1-z)^m}.$$

Applying the same inclusion-exclusion pattern as in Definition 9462.1621 to the four 14-vertex facets, the six 6-vertex intersections, the four 2-vertex intersections, and the empty total intersection gives the displayed Hilbert series. □

Theorem 9462.1623 (h-polynomial)**.** *Since* $\dim k[\mathcal{B}] = 14$*, write*

$$H_{k[\mathcal{B}]}(z) = \frac{h_{\mathcal{B}}(z)}{(1-z)^{14}}.$$

Then

$$h_{\mathcal{B}}(z) = 4 - 6(1-z)^8 + 4(1-z)^{12} - (1-z)^{14}.$$

Equivalently, the h-vector is

$$(1, 14, 5, -180, 559, -830, 525, 312, -1029, 1122, -737, 316, -87, 14, -1).$$

Proof. Multiplying the Hilbert series in Definition 9462.1622 by $(1-z)^{14}$ gives the stated numerator. The coefficient vector is the binomial expansion of that numerator. □

Theorem 9462.1624 (Non-Cohen–Macaulay blocker ring)**.** *The Stanley-Reisner ring $k[\mathcal{B}]$ is not Cohen–Macaulay.*

Proof. The complex has dimension 13, while Definition 9462.1620 gives nonzero reduced homology in degree 2. A Cohen–Macaulay 13-dimensional simplicial complex over k has no such lower-degree reduced homology in its global Reisner condition. Equivalently, the h-vector in Definition 9462.1623 contains negative entries, which is incompatible with Cohen–Macaulay nonnegativity for a standard Stanley-Reisner ring. □

Theorem 9462.1625 (Hochster witness)**.** *For the Stanley-Reisner ring $k[\mathcal{B}]$, Hochster's formula detects the full-vertex Betti number*

$$\beta_{27,30}(k[\mathcal{B}]) = 1.$$

Proof. Hochster's formula gives

$$\beta_{i,j}(k[\mathcal{B}]) = \sum_{|W|=j} \dim_k \widetilde{H}_{j-i-1}(\mathcal{B}_W; k).$$

Take $W = E$, so $j = 30$ and $\mathcal{B}_W = \mathcal{B}$. By Definition 9462.1620, $\widetilde{H}_2(\mathcal{B}; k) \cong k$. The equation $j - i - 1 = 2$ with $j = 30$ gives $i = 27$, hence the displayed Betti number. □

Theorem 9462.1626 (Depth and regularity)**.** *The blocker Stanley-Reisner ring satisfies*

$$\operatorname{pd}_{R_0} k[\mathcal{B}] = 27, \qquad \operatorname{depth} k[\mathcal{B}] = 3, \qquad \operatorname{reg} k[\mathcal{B}] = 3.$$

Proof. The Hochster witness in Definition 9462.1625 gives $\operatorname{pd}_{R_0} k[\mathcal{B}] \geq 27$. On the Alexander-dual side, Definition 9462.1605 has shifts $16, 24, 28, 30$, so

$$\operatorname{reg}(I_{\mathcal{B}}^{\vee}) = \max\{16 - 1, 24 - 2, 28 - 3, 30 - 4\} = 27.$$

Terai duality gives

$$\operatorname{pd}_{R_0} k[\mathcal{B}] = \operatorname{reg}(I_{\mathcal{B}}^{\vee}),$$

hence the projective dimension is 27. Since R_0 has 30 variables, Auslander–Buchsbaum gives

$$\operatorname{depth} k[\mathcal{B}] = 30 - 27 = 3.$$

Terai duality also gives

$$\operatorname{reg} k[\mathcal{B}] = \operatorname{pd}(I_{\mathcal{B}}^{\vee}).$$

The four-generator dual ideal has minimal Taylor resolution of length 3 as an ideal, so $\operatorname{reg} k[\mathcal{B}] = 3$. This is also realized by the Hochster Betti number, where $30 - 27 = 3$. □

Corollary 9462.1627 (Cohen–Macaulay defect)**.** *The Cohen–Macaulay defect of $k[\mathcal{B}]$ is*

$$\dim k[\mathcal{B}] - \operatorname{depth} k[\mathcal{B}] = 14 - 3 = 11.$$

Proof. The dimension is 14 by Definition 9462.1623, and the depth is 3 by Definition 9462.1626. □

Theorem 9462.1628 (Homological invariants for coverage blockers)**.** *For the coverage blocker complex $\mathcal{B}_{n,\ell}$, let*

$$N = \ell(2^n - 1).$$

Then

$$\dim k[\mathcal{B}_{n,\ell}] = \ell(2^{n-1} - 1),$$

and

$$\operatorname{depth} k[\mathcal{B}_{n,\ell}] = n-1, \qquad \operatorname{reg} k[\mathcal{B}_{n,\ell}] = n-1,$$
$$\operatorname{pd} k[\mathcal{B}_{n,\ell}] = N-(n-1).$$

The full-vertex Hochster witness is

$$\beta_{N-(n-1),\,N}(k[\mathcal{B}_{n,\ell}]) = 1.$$

Proof. The maximal blocker has $\ell(2^{n-1}-1)$ vertices, giving the Stanley-Reisner dimension. By Definition 9462.1611, $\mathcal{B}_{n,\ell} \simeq S^{n-2}$; applying Hochster's formula to the full vertex set gives $j = N$ and $j-i-1 = n-2$, hence $i = N-(n-1)$. The Alexander-dual ideal has n generators by Definition 9462.1615, and its Taylor resolution has ideal projective dimension $n-1$. Terai duality and Auslander–Buchsbaum give the displayed projective dimension, depth, and regularity. □

Definition 9462.1629 (Saturated blocker faces)**.** *For $J \subsetneq \Omega$, define the saturated face*

$$F_J = \{p \in E : \sigma(p) \subseteq J\}.$$

It has size

$$|F_J| = 2(2^{|J|}-1).$$

Theorem 9462.1630 (Link of a saturated face)**.** *Let $J \subsetneq \Omega$, set $r = |J|$, and put $U = \Omega \setminus J$ with $u = |U| = 4-r$. Then*

$$\operatorname{Lk}_{\mathcal{B}}(F_J) \cong \mathcal{B}_{u,\,2\cdot 2^r}.$$

Consequently

$$\operatorname{Lk}_{\mathcal{B}}(F_J) \simeq S^{u-2} = S^{2-r}.$$

Proof. The face F_J already covers exactly the coordinates in J. A codon outside F_J contributes to the link through the nonempty projection of its support to U. For each nonempty $A \subseteq U$, the original support can be $A \cup B$ with arbitrary $B \subseteq J$, giving 2^r choices, and each support has two third-position lifts. Hence each nonempty support over U has $2 \cdot 2^r$ lifts in the link. The condition for remaining in $\mathcal{B}$ is precisely that these projected supports do not cover all of U, so the link is the coverage blocker complex $\mathcal{B}_{u,\,2\cdot 2^r}$. Its homotopy type is S^{u-2} by Definition 9462.1611. □

Corollary 9462.1631 (Codon local sphere strata)**.** *The saturated faces of rank $r = |J|$ have the following local links:*

r	*number of J*	$\lvert F_J\rvert$	*link model*	*link homotopy*
0	1	0	$\mathcal{B}_{4,2}$	S^2
1	4	2	$\mathcal{B}_{3,4}$	S^1
2	6	6	$\mathcal{B}_{2,8}$	S^0
3	4	14	$\mathcal{B}_{1,16}$	S^{-1}.

Proof. Substitute $r = 0,1,2,3$ in Definition 9462.1630. The number of saturated faces of rank r is $\binom{4}{r}$, and Definition 9462.1629 gives $|F_J| = 2(2^r-1)$. □

Corollary 9462.1632 (Coordinate examples)**.** *For $J = \{s_1\}$, the saturated face is*

$$F_J = \mathrm{GUY} = \{\mathrm{GUU}, \mathrm{GUC}\},$$

and

$$\operatorname{Lk}_{\mathcal{B}}(F_J) \simeq S^1.$$

For $J = \{s_1, f_1\}$, the saturated face is

$$F_J = \{\mathrm{UUY}, \mathrm{AUY}, \mathrm{GUY}\},$$

with two Y lifts in each displayed family, and

$$\mathrm{Lk}_{\mathcal{B}}(F_J) \simeq S^0.$$

Proof. The anchor is CUY. Flipping only s_1 changes the first base from C to G, giving GUY. Allowing both first-base bits to vary while the second base stays U gives UUY, CUY, AUY, GUY, but CUY lies in $M^\star$ rather than E. The link homotopy types are the $r = 1$ and $r = 2$ rows of Definition 9462.1631. □

Theorem 9462.1633 (Non-saturated links are cones)**.** *Let $F \in \mathcal{B}$ and set*

$$J(F) = \bigcup_{p \in F} \sigma(p).$$

If $F \neq F_{J(F)}$, then $\mathrm{Lk}_{\mathcal{B}}(F)$ is a cone and is therefore contractible.

Proof. Choose $q \in F_{J(F)} \setminus F$. For every $G \in \mathrm{Lk}_{\mathcal{B}}(F)$, the face $F \cup G$ is still a blocker. Adding q cannot enlarge the covered coordinate set, because $\sigma(q) \subseteq J(F)$. Hence $F \cup G \cup \{q\} \in \mathcal{B}$ for every link face G, so q is a cone vertex of the link. □

Corollary 9462.1634 (Local obstruction support)**.** *Every nontrivial local reduced homology group of $\mathcal{B}$ occurs in the link of a saturated face. The saturated stratum F_J carries the local sphere*

$$F_J \longmapsto S^{2-|J|}.$$

Proof. Saturated links are the spheres of Definition 9462.1630. Non-saturated links are contractible by Definition 9462.1633. □

Theorem 9462.1635 (Failure of Buchsbaum property)**.** *The blocker complex $\mathcal{B}$ is not Buchsbaum.*

Proof. A pure Buchsbaum complex has Cohen–Macaulay links at every nonempty face. Take $F_{\{s_1\}} = \mathrm{GUY}$. Its link is $\mathcal{B}_{3,4}$ by Definition 9462.1630; this link has dimension

$$4(2^2 - 1) - 1 = 11$$

but has the homotopy type S^1. Thus it has lower-degree homology relative to its dimension and is not Cohen–Macaulay. □

Theorem 9462.1636 (Universal link stratification)**.** *For the coverage blocker complex $\mathcal{B}_{n,\ell}$ and $J \subsetneq [n]$, set*

$$F_J = \{e : \sigma(e) \subseteq J\}.$$

If $r = |J|$ and $u = n - r$, then

$$|F_J| = \ell(2^r - 1), \qquad \mathrm{Lk}(F_J) \cong \mathcal{B}_{u,\,\ell 2^r} \simeq S^{n-r-2}.$$

Every non-saturated face has contractible link.

Proof. The proof of Definition 9462.1630 does not use the codon-specific values $n = 4$ and $\ell = 2$. For a projected nonempty support $A \subseteq [n] \setminus J$, the old-coordinate part can be any subset of J, giving 2^r choices, and each is multiplied by the original ℓ lifts. The cone argument for non-saturated faces is identical to Definition 9462.1633. □

Theorem 9462.1637 (Hilbert–Alexander structure theorem)**.** *The post-$M^\star$ blocker system is a thickened tetrahedral boundary with*

$$\mathcal{B} \simeq S^2, \qquad \dim \mathcal{B} = 13, \qquad \operatorname{depth} k[\mathcal{B}] = \operatorname{reg} k[\mathcal{B}] = 3.$$

Its projective dimension over the 30*-variable ring is* 27*, and its Cohen–Macaulay defect is* 11*. Its face polynomial is*

$$4(1+t)^{14} - 6(1+t)^6 + 4(1+t)^2 - 1,$$

its Hilbert series is

$$\frac{4}{(1-z)^{14}} - \frac{6}{(1-z)^6} + \frac{4}{(1-z)^2} - 1,$$

and its h-polynomial is

$$4 - 6(1-z)^8 + 4(1-z)^{12} - (1-z)^{14}.$$

At the same time, the Alexander dual ideal has only four generators, with lcm degrees $16, 24, 28, 30$*, and the non-trigger reliability polynomial is*

$$4q^{16} - 6q^{24} + 4q^{28} - q^{30}.$$

The local links are stratified by saturated faces:

$$\operatorname{Lk}_{\mathcal{B}}(F_J) \simeq S^{2-|J|}, \qquad J \subsetneq \Omega,$$

while non-saturated faces have contractible links. In particular, $\mathcal{B}$ is not Buchsbaum.

Proof. The thickened boundary, Hilbert series, and h-polynomial are the preceding theorems. The depth, regularity, projective dimension, and Cohen–Macaulay defect are Definition 9462.1626 and Definition 9462.1627. The four-generator Alexander-dual description and the lcm degrees are Definition 9462.1603 and Definition 9462.1604. The reliability identity is Definition 9462.1607. The local link stratification is Definition 9462.1630 and Definition 9462.1633; the Buchsbaum obstruction is Definition 9462.1635. □

Definition 9462.1638 (Induced blocker nerve)**.** *For $W \subseteq E$, let $\mathcal{B}_W$ be the induced subcomplex of $\mathcal{B}$ on W. Define a simplicial complex N_W on the vertex set Ω by declaring $T \subseteq \Omega$ to be a face when there exists $p \in W$ such that*

$$\sigma(p) \cap T = \varnothing.$$

Equivalently, T is a face exactly when some support in W is contained in $\Omega \setminus T$.

Theorem 9462.1639 (Hochster–nerve reduction)**.** *For every $W \subseteq E$,*

$$\mathcal{B}_W \simeq N_W.$$

Proof. The induced complex is covered by the four simplices

$$W \cap B_{\neg s_1}, \quad W \cap B_{\neg f_1}, \quad W \cap B_{\neg s_2}, \quad W \cap B_{\neg f_2}.$$

For $T \subseteq \Omega$, the corresponding intersection is nonempty exactly when some $p \in W$ omits every coordinate in T, that is, $\sigma(p) \cap T = \varnothing$. All nonempty intersections are simplices. The nerve theorem identifies $\mathcal{B}_W$ with this four-vertex nerve. □

Corollary 9462.1640 (Hochster homology range)**.** *All reduced homology groups appearing in Hochster's formula for $k[\mathcal{B}]$ are computed on four-vertex nerves. Hence they occur only in degrees*

$$-1,\ 0,\ 1,\ 2.$$

Consequently the Betti table of $k[\mathcal{B}]$ lies on the four diagonals

$$j - i = 0,\ 1,\ 2,\ 3,$$

and

$$\operatorname{reg} k[\mathcal{B}] = 3.$$

Proof. Hochster's formula is

$$\beta_{i,j}(k[\mathcal{B}]) = \sum_{|W|=j} \dim_k \widetilde{H}_{j-i-1}(\mathcal{B}_W; k).$$

By Definition 9462.1639, $\mathcal{B}_W$ has the homology of a complex on at most four vertices. Such a complex has no reduced homology above degree 2. The full complex has nonzero $\widetilde{H}_2$ by Definition 9462.1620, so the regularity bound is sharp. □

Theorem 9462.1641 (Top Betti strand)**.** *For $4 \le j \le 30$, the highest Betti strand is*

$$\beta_{j-3,j}(k[\mathcal{B}]) = [x^j](2x + x^2)^4(1 + x)^{22}.$$

Proof. The term $\widetilde{H}_2(N_W)$ is nonzero precisely when $N_W = \partial\Delta^3$. Since the full four-coordinate face is never present in N_W, this is equivalent to the presence of every proper face. It is enough and necessary to have every three-coordinate face; the face $\Omega \setminus \{i\}$ requires a codon in W with support contained in $\{i\}$, hence one of the two singleton-support lifts for coordinate i.

Thus W must choose at least one of the two singleton lifts for each of the four coordinates. Each coordinate contributes $(1 + x)^2 - 1 = 2x + x^2$. The remaining $30 - 8 = 22$ external codons are unrestricted and contribute $(1 + x)^{22}$. In Hochster's formula, $\widetilde{H}_2$ appears when $j - i - 1 = 2$, namely $i = j - 3$. □

Definition 9462.1642 (Hochster diagonal series)**.** *For $q = -1, 0, 1, 2$, define*

$$B_q(x) = \sum_j \beta_{j-q-1,j}(k[\mathcal{B}])x^j.$$

Theorem 9462.1643 (Complete Hochster diagonal series)**.** *The four Hochster diagonal generating functions are*

$$B_{-1}(x) = (1 + x)^2,$$

$$B_0(x) = x^2(x + 1)^2(x + 2)^2 P_0(x),$$

where

$$\begin{aligned} P_0(x) =& 4x^{12} + 48x^{11} + 268x^{10} + 920x^9 + 2167x^8 \\ &+ 3704x^7 + 4736x^6 + 4592x^5 + 3373x^4 \\ &+ 1844x^3 + 720x^2 + 184x + 25, \end{aligned}$$

$$B_1(x) = x^3(x + 1)^{10}(x + 2)^3 P_1(x),$$

where

$$\begin{aligned} P_1(x) =& 6x^{10} + 60x^9 + 278x^8 + 784x^7 + 1491x^6 \\ &+ 2002x^5 + 1928x^4 + 1320x^3 + 617x^2 \\ &+ 178x + 22, \end{aligned}$$

and

$$B_2(x) = x^4(x+1)^{22}(x+2)^4.$$

Proof. Enumerate support-type subsets

$$A \subseteq 2^{\Omega} \setminus \{\varnothing\}.$$

Each chosen support type has two codon lifts, so its nonempty lift selection contributes $(1+x)^2-1 = 2x+x^2$. For each A, form the four-vertex nerve N_A as in Definition 9462.1638. Hochster's formula therefore gives

$$B_q(x) = \sum_A \dim_k \widetilde{H}_q(N_A; k)(2x+x^2)^{|A|}.$$

There are only 2^{15} support-type subsets. Computing the reduced homology of their four-vertex nerves gives the displayed factorizations. □

Corollary 9462.1644 (Diagonal masses)**.** *Evaluating the four diagonal series at $x = 1$ gives*

q	$j - i = q + 1$	$B_q(1)$
-1	0	4
0	1	813060
1	2	240150528
2	3	339738624.

Proof. Substitute $x = 1$ in Definition 9462.1643. □

Corollary 9462.1645 (Low-degree diagonal entries)**.** *The first terms of the two middle diagonals are*

j	$\beta_{j-1,j}$
2	100
3	1036
4	5413
5	18558
6	46109
7	87320
8	129681
9	153426
10	145609

and

j	$\beta_{j-2,j}$
3	176
4	3448
5	32004
6	188174
7	789384
8	2519871
9	6368866
10	13082137.

Proof. These are the coefficients of $B_0(x)$ and $B_1(x)$ in Definition 9462.1643. □

Definition 9462.1646 (Support-level Betti polynomials)**.** *Let*

$$\mathcal{S} = 2^{\Omega} \setminus \{\varnothing\}.$$

For $A \subseteq \mathcal{S}$, define the support nerve

$$N_A = \bigcup_{S \in A} 2^{\Omega \setminus S}.$$

For $q = -1, 0, 1, 2$, set

$$Q_q(z) = \sum_{A \subseteq \mathcal{S}} \dim_k \widetilde{H}_q(N_A; k) z^{|A|}.$$

Theorem 9462.1647 (Support-level Betti polynomials)**.** *The support-level Hochster polynomials are*

$$Q_{-1}(z) = 1 + z,$$

$$Q_0(z) = z^2(1+z)\left(4z^6 + 28z^5 + 87z^4 + 152z^3 + 157z^2 + 92z + 25\right),$$

$$Q_1(z) = z^3(1+z)^5\left(6z^5 + 38z^4 + 99z^3 + 132z^2 + 89z + 22\right),$$

and

$$Q_2(z) = z^4(1+z)^{11}.$$

Proof. Enumerate the 2^{15} subsets of the support-type set $\mathcal{S}$. For each A, the nerve is the four-vertex complex N_A. Computing its reduced homology and summing by $|A|$ gives the four displayed polynomials. The highest polynomial has the direct form $Q_2(z) = z^4(1+z)^{11}$ because $\widetilde{H}_2$ occurs exactly when the four singleton supports are present, while the other 11 support types are unrestricted. □

Theorem 9462.1648 (Support-lift Betti transform)**.** *If each support type has ℓ lifts, then the lifted Hochster diagonal series is*

$$B_q^{(\ell)}(x) = Q_q\left((1+x)^{\ell} - 1\right).$$

In the codon case $\ell = 2$, so

$$B_q(x) = Q_q(2x + x^2).$$

Proof. For a fixed support type, either no lift is chosen, or a nonempty subset of its ℓ lifts is chosen. The nonempty choice generating function is $(1+x)^{\ell} - 1$. If the support-type set is A, its lift contribution is therefore $\left((1+x)^{\ell} - 1\right)^{|A|}$, while the homology factor is $\dim \widetilde{H}_q(N_A)$. Summing over A gives the formula. □

Corollary 9462.1649 (Codon support-lift compression)**.** *The complete* 30*-variable Betti table of $k[\mathcal{B}]$ is determined by the four support-level polynomials*

$$Q_{-1},\ Q_0,\ Q_1,\ Q_2$$

and the substitution

$$z \mapsto (1+x)^2 - 1 = 2x + x^2.$$

Equivalently,

$$30\text{-}\textit{variable Betti data} \longrightarrow 15\text{-}\textit{support-type nerves} \longrightarrow 4\ \textit{support polynomials.}$$

Proof. The four codon-level diagonal series are Definition 9462.1643. By Definition 9462.1648, each is obtained from the corresponding support-level polynomial by the codon substitution $z = 2x + x^2$. □

Corollary 9462.1650 (Top Betti strand values)**.** *The nonzero values of the top strand are*

j	$\beta_{j-3,j}$
4	16
5	384
6	4424
7	32568
8	172041
9	694254
10	2224607
11	5808396
12	12582427
13	22907654
14	35377221
15	46646368
16	52738794
17	51252348
18	42843366
19	30778024
20	18951702
21	9957596
22	4434562
23	1658184
24	513821
25	129558
26	25899
27	3948
28	431
29	30
30	1.

In particular,

$$\beta_{27,30}(k[\mathcal{B}]) = 1.$$

Proof. These are the coefficients of Definition 9462.1641. The final entry corresponds to choosing all 30 external codons, so the induced complex is $\mathcal{B}$ itself. □

Corollary 9462.1651 (Generator row)**.** *The first Betti row records the minimal full triggers:*

$$\beta_{1,1} = 2, \qquad \beta_{1,2} = 100, \qquad \beta_{1,3} = 176, \qquad \beta_{1,4} = 16.$$

Proof. The Stanley-Reisner ideal $I_{\mathcal{B}}$ is generated by minimal full triggers. By Definition 9462.1563, their codon-level degree distribution is $2, 100, 176, 16$ in degrees $1, 2, 3, 4$. □

Theorem 9462.1652 (Betti diagonal compression)**.** *The nonzero Betti entries of $k[\mathcal{B}]$ lie on four diagonals. Their homological interpretation is*

$j-i$	*nerve homology*	*nonzero entries*
0	$\widetilde{H}_{-1}$	3
1	$\widetilde{H}_0$	17
2	$\widetilde{H}_1$	24
3	$\widetilde{H}_2$	27.

Proof. The diagonal $j-i=d$ corresponds in Hochster's formula to reduced homology degree $d-1$. By Definition 9462.1640, only $d=0,1,2,3$ can occur. The displayed counts are the exhaustive four-vertex nerve enumeration induced by the 30 external codons; the top diagonal is the strand of Definition 9462.1641. □

Theorem 9462.1653 (Universal top Betti strand)**.** *For the coverage blocker complex $\mathcal{B}_{n,\ell}$, the highest Betti strand is*

$$\beta_{j-(n-1),j}(k[\mathcal{B}_{n,\ell}]) = [x^j]\left((1+x)^\ell - 1\right)^n (1+x)^{\ell(2^n-1-n)}.$$

Proof. The induced nerve is the boundary $\partial\Delta^{n-1}$ exactly when each singleton coordinate support has at least one selected lift. Each of the n singleton supports has ℓ lifts, contributing $(1+x)^\ell - 1$. All other elements number $\ell(2^n-1-n)$ and may be chosen freely. The boundary sphere has homology degree $n-2$, so Hochster gives $j-i-1=n-2$, or $i=j-(n-1)$. □

Theorem 9462.1654 (Hochster–nerve Betti theorem)**.** *Every induced subcomplex $\mathcal{B}_W$ of the post-$M^\star$ blocker complex has the homology of a four-coordinate nerve N_W. Therefore the Betti table of $k[\mathcal{B}]$ is governed by the nerve homology degrees $\widetilde{H}_{-1}, \widetilde{H}_0, \widetilde{H}_1, \widetilde{H}_2$, has regularity 3, has projective dimension 27, and has depth 3. The four Hochster diagonal generating functions are those of Definition 9462.1643, and they are obtained from the support-level polynomials of Definition 9462.1647 by the substitution $z = 2x + x^2$. In particular, its top strand is*

$$\beta_{j-3,j} = [x^j](2x+x^2)^4(1+x)^{22}.$$

Proof. The nerve reduction is Definition 9462.1639. The homology range and regularity are Definition 9462.1640. The top strand is Definition 9462.1641, and the four diagonal series are Definition 9462.1643. The support-lift compression is Definition 9462.1648. The projective dimension and depth are Definition 9462.1626. □

Concretized in §3744: Hochster nerve Betti reduction is realized as finite support, nerve-face, homology-range, Betti-strand, and ledger NameCert rows.

Theorem 9462.1655 (Support-nerve census)**.** *The 2^{15} support families $A \subseteq \mathcal{S}$ have the following reduced Betti signatures*

$$(\widetilde{\beta}_{-1}, \widetilde{\beta}_0, \widetilde{\beta}_1, \widetilde{\beta}_2)$$

for their support nerves N_A:

signature	*support-family count*
$(1,0,0,0)$	2
$(0,0,0,0)$	18680
$(0,1,0,0)$	908
$(0,2,0,0)$	56
$(0,3,0,0)$	2
$(0,0,1,0)$	9760
$(0,0,2,0)$	1216
$(0,0,3,0)$	32
$(0,1,1,0)$	64
$(0,0,0,1)$	2048.

Proof. For each of the 2^{15} subsets $A \subseteq 2^{\Omega} \setminus \{\varnothing\}$, form the four-vertex nerve

$$N_A = \bigcup_{S \in A} 2^{\Omega \setminus S}.$$

Computing reduced homology of these four-vertex complexes gives the displayed census. The total count is

$$2 + 18680 + 908 + 56 + 2 + 9760 + 1216 + 32 + 64 + 2048 = 32768.$$

□

Corollary 9462.1656 (Support-level homology distribution)**.** *Among the support nerves,* 18680 *are acyclic,* $2048 = 2^{11}$ *have top reduced homology* $\widetilde{H}_2$*, and all nonzero support-level homology appears in degrees* $-1, 0, 1, 2$*.*

Proof. This is the census of Definition 9462.1655. The degree range also follows from the fact that every N_A has at most four vertices. □

Theorem 9462.1657 (Top support nerve criterion)**.** *The support nerve N_A has nonzero $\widetilde{H}_2$ if and only if A contains the four singleton supports*

$$\{s_1\}, \quad \{f_1\}, \quad \{s_2\}, \quad \{f_2\}.$$

Consequently exactly 2^{11} *support families have* $\widetilde{H}_2(N_A) \neq 0$*, and*

$$Q_2(z) = z^4(1+z)^{11}.$$

Proof. For a four-vertex nerve to have $\widetilde{H}_2$, it must be the tetrahedron boundary. The three-face $\Omega \setminus \{i\}$ appears precisely when A contains a support contained in $\{i\}$, namely the singleton support $\{i\}$. Thus all four singleton supports are necessary and sufficient. The other $15 - 4 = 11$ support types may be chosen freely. □

Theorem 9462.1658 (Support-nerve census theorem)**.** *The post-$M^\star$ Betti table is determined by the following three finite layers:*

$$N_A \quad (A \subseteq \mathcal{S},\ |\mathcal{S}| = 15),$$
$$Q_q(z) = \sum_{A \subseteq \mathcal{S}} \dim_k \widetilde{H}_q(N_A; k) z^{|A|},$$

and the codon lift substitution

$$z \mapsto (1+x)^2 - 1 = 2x + x^2.$$

Thus

$$B_q(x) = Q_q(2x + x^2).$$

Proof. The nerve reduction is Definition 9462.1639. The support-level polynomials are Definition 9462.1647, and the lift transform is Definition 9462.1648. The census in Definition 9462.1655 records the finite homology types from which the four polynomials are formed. □

Concretized in §4035: The support-nerve census and codon-lifted Betti table are realized as finite incidence, nerve-realization, simplicial-complex, finite-chain, homology, lifted-ledger, transport, replay, provenance, and NameCert rows.

Theorem 9462.1659 (Support-to-nerve realization)**.** *The assignment*

$$A \subseteq \mathcal{S} \longmapsto N_A = \bigcup_{S \in A} 2^{\Omega \setminus S}$$

has image exactly the set of subcomplexes of the tetrahedron boundary $\partial\Delta^3$, with both the void complex and the complex consisting only of the empty face included.

Proof. For every nonempty support $S \subseteq \Omega$, the complement $\Omega \setminus S$ is a proper face of Δ^3. Hence N_A is the downward closure of a family of proper faces, and therefore is a subcomplex of $\partial\Delta^3$.

Conversely, let $K \subseteq \partial\Delta^3$ be a subcomplex. For each maximal face F of K, set $S_F = \Omega \setminus F$. These supports are nonempty. If $A = \{S_F : F \in \mathrm{Facets}(K)\}$, then

$$N_A = \bigcup_{F \in \mathrm{Facets}(K)} 2^F = K.$$

Thus every subcomplex of $\partial\Delta^3$ is realized. □

Concretized in §4079: The support-family-to-boundary-subcomplex realization route is recorded as a finite NameCert packet between support rows, nerve rows, census rows, and lifted-Betti consumers.

Theorem 9462.1660 (Support-family fiber)**.** *Let $K \subseteq \partial\Delta^3$ be a subcomplex, and let $|K|$ denote its number of faces, including the empty face. The generating function for support families A with $N_A = K$ is*

$$G_K(z) = z^{|\mathrm{Facets}(K)|}(1+z)^{|K| - |\mathrm{Facets}(K)|}.$$

Proof. Write

$$C_A = \{\Omega \setminus S : S \in A\}.$$

Then N_A is the downward closure of C_A. The equality $N_A = K$ holds exactly when $C_A \subseteq K$ and C_A contains every maximal face of K. The maximal faces are forced, and the remaining $|K| - |\mathrm{Facets}(K)|$ faces may be chosen independently. □

Corollary 9462.1661 (Subcomplex-weighted support polynomials)**.** *For $q = -1, 0, 1, 2$,*

$$Q_q(z) = \sum_{K \subseteq \partial\Delta^3} \dim_k \widetilde{H}_q(K; k) z^{|\mathrm{Facets}(K)|}(1+z)^{|K| - |\mathrm{Facets}(K)|}.$$

Proof. Group the defining sum for $Q_q(z)$ by the realized subcomplex $K = N_A$ and apply Definition 9462.1660. □

Theorem 9462.1662 (Tetrahedron-boundary subcomplex census)**.** *The boundary $\partial\Delta^3$ has* 167 *subcomplexes under the convention of Definition 9462.1659. Their reduced Betti signatures are:*

signature	*subcomplex count*
$(1,0,0,0)$	2
$(0,0,0,0)$	64
$(0,1,0,0)$	37
$(0,2,0,0)$	10
$(0,3,0,0)$	1
$(0,0,1,0)$	37
$(0,0,2,0)$	10
$(0,0,3,0)$	1
$(0,1,1,0)$	4
$(0,0,0,1)$	1.

Moreover,

$$\sum_{K\subseteq\partial\Delta^3} 2^{|K|-|\operatorname{Facets}(K)|} = 32768.$$

Proof. Enumerate the downward-closed families of proper faces of a four-vertex simplex and compute reduced homology for each subcomplex. The displayed rows sum to 167. The final identity is the specialization $G_K(1)$ summed over all realized subcomplexes; by Definition 9462.1660, it counts all 2^{15} support families. □

Theorem 9462.1663 (Support-to-nerve Betti formula)**.** *The complete Hochster diagonal series of $k[\mathcal{B}]$ satisfy*

$$B_q(x) = \sum_{K\subseteq\partial\Delta^3} \dim_k \widetilde{H}_q(K;k)(2x+x^2)^{|\operatorname{Facets}(K)|}(1+2x+x^2)^{|K|-|\operatorname{Facets}(K)|}.$$

Equivalently,

$$B_q(x) = \sum_{K\subseteq\partial\Delta^3} \dim_k \widetilde{H}_q(K;k)x^{|\operatorname{Facets}(K)|}(x+2)^{|\operatorname{Facets}(K)|}(1+x)^{2(|K|-|\operatorname{Facets}(K)|)}.$$

Proof. Combine Definition 9462.1661 with the codon lift substitution of Definition 9462.1648. The two displayed forms are identical because $2x+x^2 = x(x+2)$ and $1+2x+x^2 = (1+x)^2$. □

Corollary 9462.1664 (Top diagonal from the boundary complex)**.** *The top Hochster diagonal is contributed by the single subcomplex $K = \partial\Delta^3$. Since this complex has four facets and fifteen proper faces,*

$$B_2(x) = (2x+x^2)^4(1+2x+x^2)^{11} = x^4(x+2)^4(1+x)^{22}.$$

In particular,

$$\beta_{27,30} = 1.$$

Proof. Among subcomplexes of $\partial\Delta^3$, nonzero $\widetilde{H}_2$ occurs only for $\partial\Delta^3$ itself. The formula is the $q = 2$ specialization of Definition 9462.1663. □

Theorem 9462.1665 (Minimal trigger sphere)**.** *Let*

$$A = \{S_1, \dots, S_m\} \subseteq 2^\Omega \setminus \{\varnothing\}$$

be a quotient-level minimal full trigger. Put

$$F_i = \Omega \setminus S_i.$$

Then the support nerve

$$N_A = \bigcup_{i=1}^{m} 2^{F_i}$$

is homotopy equivalent to S^{m-2}. Equivalently,

$$\widetilde{H}_{m-2}(N_A; k) \cong k.$$

Concretized in §4137: finite support-nerve NameCert packet for minimal trigger sphere cores

Proof. The full-trigger condition gives

$$\bigcup_{i=1}^{m} S_i = \Omega.$$

Taking complements,

$$\bigcap_{i=1}^{m} F_i = \varnothing.$$

Minimality says that every proper subfamily fails to cover Ω. Thus, for every proper $I \subsetneq \{1, \dots, m\}$,

$$\bigcap_{i \in I} F_i \neq \varnothing.$$

The cover of N_A by the simplices 2^{F_i} therefore has nerve $\partial\Delta^{m-1}$. Every nonempty intersection of these simplices is again a simplex, hence contractible. The nerve theorem gives

$$N_A \simeq \partial\Delta^{m-1} \simeq S^{m-2}.$$

□

Theorem 9462.1666 (Sphere core converse)**.** *Let $A = \{S_1, \dots, S_m\}$ and $F_i = \Omega \setminus S_i$. If the cover of N_A by the m simplices 2^{F_i} has nerve $\partial\Delta^{m-1}$, then A is a quotient-level minimal full trigger.*

Proof. The nerve condition says that every proper intersection $\bigcap_{i \in I} F_i$ is nonempty, while the full intersection $\bigcap_{i=1}^{m} F_i$ is empty. Taking complements gives

$$\bigcup_{i=1}^{m} S_i = \Omega,$$

and, for every proper I,

$$\bigcup_{i \in I} S_i \neq \Omega.$$

This is precisely the minimal full-trigger condition. □

Corollary 9462.1667 (Minimal triggers are minimal homology cores)**.** *At quotient level, a size-m minimal full trigger is the same support data as a minimal support-nerve core of type S^{m-2}:*

trigger size	*reduced homology*	*sphere*
1	$\widetilde{H}_{-1}$	S^{-1}
2	$\widetilde{H}_0$	S^0
3	$\widetilde{H}_1$	S^1
4	$\widetilde{H}_2$	S^2.

Proof. Apply Definition 9462.1665 and Definition 9462.1666. □

Theorem 9462.1668 (Minimal trigger homology core counts)**.** *For $\Omega = \{s_1, f_1, s_2, f_2\}$, the quotient-level minimal homology cores have counts*

trigger size	*quotient count*	*sphere*	*codon count*
1	1	S^{-1}	2
2	25	S^0	100
3	22	S^1	176
4	1	S^2	16.

Thus the quotient-level polynomial is

$$z + 25z^2 + 22z^3 + z^4,$$

and the codon-level polynomial is

$$2z + 100z^2 + 176z^3 + 16z^4.$$

Proof. The quotient-level counts are the minimal full-trigger counts of Definition 9462.1559. The sphere type is supplied by Definition 9462.1667. At codon level, each support in a size-m quotient trigger has two f_3-lifts, so the quotient count is multiplied by 2^m. □

Corollary 9462.1669 (Lowest Hochster witnesses)**.** *The minimal full triggers are the lowest-degree Hochster witnesses on the four diagonal series:*

series	*homology*	*minimal trigger degree*	*lowest coefficient*
B_{-1}	$\widetilde{H}_{-1}$	1	2
B_0	$\widetilde{H}_0$	2	100
B_1	$\widetilde{H}_1$	3	176
B_2	$\widetilde{H}_2$	4	16.

Proof. By Hochster's formula, the diagonal B_q records induced support nerves with nonzero $\widetilde{H}_q$. By Definition 9462.1667, a size-m minimal trigger contributes in degree $q = m - 2$. Its codon-level lift multiplicity is the lowest codon degree on that diagonal. The coefficients are exactly the codon counts in Definition 9462.1668. □

Corollary 9462.1670 (Energy-minimal subclass)**.** *The energy-minimal triggers are the partition subclass of the minimal homology cores. At quotient level there are* 15 *such cores, while all minimal homology cores number* 49*; at codon level the corresponding counts are* 94 *and* 294*.*

Proof. By Definition 9462.1579, energy minimality imposes that the supports form a set partition of Ω. This is stronger than the minimal full-trigger condition used in Definition 9462.1665; it selects the partition cores among all minimal support-nerve sphere cores. The counts are those of Definition 9462.1580 and Definition 9462.1668. □

Definition 9462.1671 (Private coordinate)**.** *Let $A = \{S_1, \dots, S_m\}$ be a quotient-level support family on an n-element trigger coordinate set Ω. A coordinate $a \in \Omega$ is private to S_i if*

$$a \in S_i \quad \text{and} \quad a \notin S_j \text{ for } j \neq i.$$

Theorem 9462.1672 (Private-coordinate criterion)**.** *A quotient-level support family*

$$A = \{S_1, \dots, S_m\}$$

is a minimal full trigger if and only if

$$\bigcup_{i=1}^{m} S_i = \Omega$$

and every S_i has at least one private coordinate.

Proof. If A is a minimal full trigger, deleting S_i destroys the cover of Ω. Hence some coordinate is covered by S_i and by no other member of the family; this is a private coordinate for S_i.

Conversely, if every S_i has a private coordinate, deleting S_i misses that coordinate. Since the full family covers Ω, the family is a minimal full trigger. □

Theorem 9462.1673 (Private-coordinate enumeration)**.** *Let $C(n, m)$ be the number of quotient-level minimal full triggers of size m on an n-coordinate trigger set. Then*

$$C(n, m) = \frac{1}{m!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} (2^m - 1 - j)^n.$$

Proof. First count ordered tuples $(S_1, \dots, S_m)$. Each coordinate $a \in \Omega$ determines a nonempty incidence pattern

$$\phi(a) = \{i : a \in S_i\} \subseteq [m].$$

There are $2^m - 1$ nonempty patterns. By Definition 9462.1672, minimality is equivalent to requiring each singleton pattern $\{i\}$ to appear at least once. Inclusion-exclusion over the missing singleton patterns gives the ordered count

$$\sum_{j=0}^{m} (-1)^j \binom{m}{j} (2^m - 1 - j)^n.$$

Every support S_i has a private coordinate, so the S_i are pairwise distinct. Hence each unordered family is counted $m!$ times. □

Corollary 9462.1674 (Codon minimal-trigger counts)**.** *For the codon terminal system $n = 4$ and $\ell = 2$,*

m	$C(4, m)$	$2^m C(4, m)$
1	1	2
2	25	100
3	22	176
4	1	16.

Thus the codon-level minimal full-trigger polynomial is

$$2z + 100z^2 + 176z^3 + 16z^4.$$

Proof. Evaluate Definition 9462.1673 at $n = 4$:

$$C(4,1) = 1,$$

$$C(4,2) = \frac{3^4 - 2 \cdot 2^4 + 1}{2} = 25,$$

$$C(4,3) = \frac{7^4 - 3 \cdot 6^4 + 3 \cdot 5^4 - 4^4}{6} = 22,$$

and $C(4,4) = 1$. The codon layer has two f_3-lifts for each support, so a size-m quotient trigger has 2^m codon lifts. □

Theorem 9462.1675 (Energy-refined private-coordinate formula)**.** *Let $C_{n,m}(y)$ be the quotient-level energy generating polynomial for size-m minimal full triggers on n trigger coordinates, where*

$$E(A) = \sum_{S \in A} |S|.$$

Then

$$C_{n,m}(y) = \frac{1}{m!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} \left((1+y)^m - 1 - jy\right)^n.$$

Proof. An incidence pattern $T \subseteq [m]$ contributes $y^{|T|}$ to the energy. The weight enumerator of all nonempty patterns is

$$\sum_{\varnothing \neq T \subseteq [m]} y^{|T|} = (1+y)^m - 1.$$

Forbidding j singleton patterns removes jy from this enumerator. Inclusion-exclusion over missing singleton patterns gives the ordered energy enumerator. Division by $m!$ removes the ordering of the support family. □

Corollary 9462.1676 (Codon energy refinement)**.** *For $n = 4$ and $\ell = 2$, the codon-level bivariate polynomial is*

$$\begin{aligned} T_c(z,y) = {} & 2zy^4 + z^2(28y^4 + 48y^5 + 24y^6) \\ & + z^3(48y^4 + 96y^5 + 32y^6) + 16z^4y^4. \end{aligned}$$

Consequently

$$T_c(z,1) = 2z + 100z^2 + 176z^3 + 16z^4,$$

and

$$T_c(1,y) = 94y^4 + 144y^5 + 56y^6.$$

Proof. Evaluate Definition 9462.1675 at $n = 4$, multiply the size-m quotient terms by 2^m, and collect by trigger size and energy. □

Corollary 9462.1677 (Energy-minimal partition subclass)**.** *For n trigger coordinates and ℓ lifts per support, the energy-minimal trigger polynomial is*

$$y^n \sum_{m=1}^{n} \ell^m S(n,m) z^m,$$

where $S(n,m)$ is the Stirling number of the second kind.

Proof. The minimum possible energy of a full trigger is n. Equality holds exactly when the supports form a set partition of Ω, by Definition 9462.1579. There are $S(n, m)$ set partitions with m blocks, and the lift layer contributes ℓ^m choices. □

Theorem 9462.1678 (Private-coordinate trigger enumeration theorem)**.** *In the post-$M^\star$ terminal trigger algebra, minimal full triggers are enumerated by private-coordinate inclusion-exclusion. For n trigger coordinates and ℓ lifts per support, the size-m lifted count is*

$$\ell^m \frac{1}{m!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} (2^m - 1 - j)^n,$$

and the energy-refined quotient formula is

$$\frac{1}{m!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} \left((1+y)^m - 1 - jy\right)^n .$$

Proof. The first formula is Definition 9462.1673 with the lift factor ℓ^m. The second formula is Definition 9462.1675. □

Concretized in §5812: Private-coordinate minimal-cover witnesses are realized as finite NameCert rows.

Concretized in §6786: Energy-refined private-coordinate enumeration is realized as finite witness, incidence, energy, and polynomial ledger rows.

Definition 9462.1679 (Overlap defect)**.** *For a quotient-level support family*

$$A = \{S_1, \ldots, S_m\}$$

on $\Omega = \{1, 2, 3, 4\}$, define its energy and overlap defect by

$$E(A) = \sum_{i=1}^{m} |S_i|, \qquad \delta(A) = E(A) - 4.$$

Theorem 9462.1680 (Defect range)**.** *Every quotient-level minimal full trigger on four coordinates has*

$$\delta(A) \in \{0, 1, 2\}.$$

Proof. By Definition 9462.1672, every support has a private coordinate. Hence a size-m minimal trigger has at least m coordinates of multiplicity one. The remaining $4 - m$ coordinates can contribute multiplicity at most m each, so

$$E(A) \leq m + m(4 - m).$$

For $m = 1, 2, 3, 4$, this gives $E(A) \leq 4, 6, 6, 4$, respectively. Since full triggering forces $E(A) \geq 4$, the defect is one of $0, 1, 2$. □

Theorem 9462.1681 (Nine orbit classification)**.** *Under the S_4 action on the four trigger coordinates, the quotient-level minimal full triggers split into the following nine normal forms:*

normal form	m	E	δ	*quotient count*	*codon count*
$\{1234\}$	1	4	0	1	2
$\{1, 234\}$	2	4	0	4	16
$\{12, 34\}$	2	4	0	3	12
$\{12, 234\}$	2	5	1	12	48
$\{123, 124\}$	2	6	2	6	24
$\{1, 2, 34\}$	3	4	0	6	48
$\{1, 24, 34\}$	3	5	1	12	96
$\{14, 24, 34\}$	3	6	2	4	32
$\{1, 2, 3, 4\}$	4	4	0	1	16.

The quotient counts sum to 49*, and the codon counts sum to* 294*.*

Proof. Classify support families by the private-coordinate criterion of Definition 9462.1672. For $m = 1$ the only type is $\{1234\}$. For $m = 2$, the possible support-size patterns with private coordinates and full union are $(1, 3)$, $(2, 2)$, $(2, 3)$, and $(3, 3)$, giving the four displayed two-support rows. For $m = 3$, the possible patterns are $(1, 1, 2)$, $(1, 2, 2)$, and $(2, 2, 2)$, giving the three displayed three-support rows. For $m = 4$, the four private coordinates force the four singleton supports.

The quotient counts are the sizes of the S_4 orbits of these normal forms. The codon count in a row is the quotient count multiplied by 2^m. □

Corollary 9462.1682 (Defect distribution)**.** *The quotient-level and codon-level defect distributions are*

δ	*quotient count*	*codon count*
0	15	94
1	24	144
2	10	56.

Proof. Group the rows of Definition 9462.1681 by their defect value. □

Corollary 9462.1683 (Orbit expansion of the trigger polynomial)**.** *The quotient-level bivariate minimal-trigger polynomial decomposes by orbit type as*

$$T_q(z, y) = zy^4 + 4z^2y^4 + 3z^2y^4 + 12z^2y^5 + 6z^2y^6$$
$$+6z^3y^4 + 12z^3y^5 + 4z^3y^6 + z^4y^4.$$

After applying the two-lift codon factor 2^m, this becomes

$$T_c(z, y) = 2zy^4 + z^2(28y^4 + 48y^5 + 24y^6)$$
$$+z^3(48y^4 + 96y^5 + 32y^6) + 16z^4y^4.$$

Proof. Attach the monomial $z^m y^E$ to each row of Definition 9462.1681, weighted by quotient orbit size. Multiplication by 2^m gives the codon-level polynomial. □

Corollary 9462.1684 (Partition and overlap mechanisms)**.** *The defect-*0 *rows are exactly the partition triggers. The defect-*1 *rows are the single-overlap normal forms*

$$\{12, 234\}, \qquad \{1, 24, 34\}.$$

The defect-2 rows are the double-overlap and triple-overlap normal forms

$$\{123, 124\}, \qquad \{14, 24, 34\}.$$

Proof. Defect is the total excess coordinate multiplicity beyond the four coordinates of Ω. Defect 0 is therefore equivalent to pairwise disjoint support blocks, namely a partition. The two defect-1 normal forms have one repeated coordinate. The two defect-2 normal forms respectively realize two repeated coordinates and one coordinate shared by three supports. □

Theorem 9462.1685 (Nine-orbit minimal trigger theorem)**.** *The post-$M^\star$ quotient-level minimal full triggers consist of* 49 *support covers arranged in nine S_4-orbit types. Their two-lift codon expansions contain* 294 *minimal full triggers. The defect classes are*

$$\delta = 0 : 94, \qquad \delta = 1 : 144, \qquad \delta = 2 : 56$$

at codon level.

Proof. This is the combined content of Definition 9462.1681 and Definition 9462.1682. □

Concretized in §4192: The nine orbit minimal-trigger classification is realized as finite normal-form, defect, quotient-count, codon-count, transport, replay, provenance, and local NameCert rows.

Definition 9462.1686 (Minimal trigger lift fiber)**.** *Let $A = \{S_1, \ldots, S_m\}$ be a quotient-level minimal full trigger. Each support S_i has two f_3-lifts. The lift fiber over A is the set of choices*

$$\epsilon = (\epsilon_1, \ldots, \epsilon_m) \in \{0, 1\}^m,$$

with lifted trigger

$$A^\epsilon = \{S_1^{(\epsilon_1)}, \ldots, S_m^{(\epsilon_m)}\}.$$

Theorem 9462.1687 (Lift-bundle decomposition)**.** *Let $\mathcal{T}_q$ be the quotient-level minimal full triggers and let $\mathcal{T}_c$ be the codon-level minimal full triggers. Then*

$$\mathcal{T}_c = \bigsqcup_{A \in \mathcal{T}_q} Q_{|A|}.$$

In the codon case this gives

$$\mathcal{T}_c = Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4.$$

Proof. Projection along the f_3 coordinate sends a codon-level minimal trigger to a quotient-level minimal support cover. Conversely, every support in a size-m quotient trigger has two independent f_3-lift choices, and these choices do not change the support union or minimality. Hence the fiber over the quotient trigger is Q_m, and distinct quotient triggers have disjoint lift sets. □

Concretized in §4129: The minimal trigger lift-fiber decomposition is realized as finite quotient-support, lift-choice, orbit-fiber, and flip-ledger NameCert data.

Concretized in §8695: The finite trigger flip graph is realized as quotient-support, lift-fiber, edge-budget, transport, replay, provenance, and NameCert data.

Corollary 9462.1688 (Lift-fiber distribution)**.** *The fiber dimensions and codon counts are*

m	*quotient count*	*fiber*	*codon count*
1	1	Q_1	2
2	25	Q_2	100
3	22	Q_3	176
4	1	Q_4	16.

Thus $|\mathcal{T}_c| = 294$.

Proof. This is the size distribution of quotient-level minimal triggers from Definition 9462.1674, with the fiber Q_m attached to each size-m trigger. □

Corollary 9462.1689 (Orbit fibers)**.** *The nine quotient orbit types carry the following lift fibers:*

normal form	m	*quotient count*	*fiber*	*codon count*
$\{1234\}$	1	1	Q_1	2
$\{1, 234\}$	2	4	Q_2	16
$\{12, 34\}$	2	3	Q_2	12
$\{12, 234\}$	2	12	Q_2	48
$\{123, 124\}$	2	6	Q_2	24
$\{1, 2, 34\}$	3	6	Q_3	48
$\{1, 24, 34\}$	3	12	Q_3	96
$\{14, 24, 34\}$	3	4	Q_3	32
$\{1, 2, 3, 4\}$	4	1	Q_4	16.

Proof. Combine Definition 9462.1681 with Definition 9462.1687. □

Definition 9462.1690 (Lift-flip graph)**.** *Inside one fiber over* $A = \{S_1, \ldots, S_m\}$*, two lifted triggers are adjacent if their lift choices differ in exactly one coordinate.*

Theorem 9462.1691 (Lift-flip edge count)**.** *The total number of lift-flip edges inside all fibers is* 397*.*

Proof. The fiber over a size-m quotient trigger is Q_m, whose edge count is $m2^{m-1}$. Therefore the total number of lift-flip edges is

$$1 \cdot 1 + 25 \cdot 4 + 22 \cdot 12 + 1 \cdot 32 = 397.$$

□

Definition 9462.1692 (Quotient support-flip graph)**.** *For fixed size* m*, two quotient-level minimal triggers are adjacent if they have the same size and differ by replacing exactly one support:*

$$|A \triangle B| = 2.$$

Theorem 9462.1693 (Quotient support-flip graph)**.** *The fixed-size quotient support-flip graphs have the following statistics:*

m	*vertices*	*edges*	*components*	*diameter*
1	1	0	1	0
2	25	102	1	3
3	22	48	1	4
4	1	0	1	0.

Proof. Enumerate the quotient-level minimal full triggers at each size. In a fixed-size layer, join two vertices when their symmetric difference has size two. Computing the connected components and graph diameters gives the displayed table. □

Theorem 9462.1694 (Orbit-edge structure)**.** *In the $m = 2$ quotient support-flip graph, the orbit-edge counts are*

orbit pair	*edge count*
$\{1, 234\}$–$\{12, 234\}$	12
$\{1, 234\}$–$\{123, 124\}$	12
$\{12, 34\}$–$\{12, 234\}$	12
$\{12, 234\}$–$\{12, 234\}$	18
$\{12, 234\}$–$\{123, 124\}$	36
$\{123, 124\}$–$\{123, 124\}$	12.

In the $m = 3$ layer the counts are

orbit pair	*edge count*
$\{1, 2, 34\}$–$\{1, 24, 34\}$	24
$\{1, 24, 34\}$–$\{1, 24, 34\}$	12
$\{1, 24, 34\}$–$\{14, 24, 34\}$	12.

Proof. Refine the support-flip graph of Definition 9462.1693 by the orbit labels of Definition 9462.1681 and count edges by unordered orbit pair. □

Corollary 9462.1695 (Defect-one transition layer)**.** *In the nontrivial quotient flip layers, defect-1 orbit types form the transition layer between partition triggers and high-overlap triggers.*

Proof. In the $m = 3$ layer, the partition orbit $\{1, 2, 34\}$ only connects to the defect-1 orbit $\{1, 24, 34\}$, and the defect-2 orbit $\{14, 24, 34\}$ also only connects to that defect-1 orbit. In the $m = 2$ layer, every partition-to-overlap path passes through or touches the displayed defect-1 orbit $\{12, 234\}$, while the high-overlap orbit also has internal edges. □

Theorem 9462.1696 (Minimal-trigger bundle theorem)**.** *The post-$M^\star$ codon-level minimal trigger space is an f_3-lift hypercube bundle over the quotient minimal-cover base:*

$$\mathcal{T}_c = Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4.$$

Its base has 49 vertices arranged in nine S_4-orbit types, the lift fibers contribute 397 internal lift-flip edges, and the nontrivial fixed-size quotient layers are connected.

Proof. The bundle decomposition is Definition 9462.1687. The orbit stratification is Definition 9462.1681. The lift-flip count and quotient connectivity are Definition 9462.1691 and Definition 9462.1693. □

Definition 9462.1697 (Private and shared coordinates)**.** *Let $A = \{S_1, \ldots, S_m\}$ be a quotient-level support family on Ω. The private coordinate set of S_i is*

$$P_i = S_i \setminus \bigcup_{j \neq i} S_j.$$

The shared coordinate set is

$$H = \Omega \setminus \bigcup_{i=1}^{m} P_i.$$

For $a \in H$, write

$$R_a = \{i : a \in S_i\}.$$

Theorem 9462.1698 (Private-shared decomposition)**.** *A quotient-level minimal full trigger of size m is equivalently the following data:*

$$H \subseteq \Omega,$$

a partition of the complement into nonempty private blocks

$$\Omega \setminus H = P_1 \sqcup \cdots \sqcup P_m,$$

and, for each $a \in H$, a row set

$$R_a \subseteq \{P_1, \ldots, P_m\}, \qquad |R_a| \geq 2.$$

The support corresponding to a private block P is recovered by

$$S_P = P \cup \{a \in H : P \in R_a\}.$$

This decomposition is unique.

Proof. Given a minimal full trigger, every coordinate has a well-defined multiplicity among the supports. If the multiplicity is one, the coordinate belongs to the unique private block attached to its support. If the multiplicity is at least two, the coordinate lies in H and has the unique set of private blocks whose supports contain it. Minimality gives every private block nonempty by Definition 9462.1672.

Conversely, the displayed data define supports S_P whose union is Ω. Since every P_i is nonempty, each S_i has a private coordinate, so the family is a minimal full trigger. The construction also proves uniqueness. □

Theorem 9462.1699 (Decoration enumeration)**.** *The number of quotient-level size-m minimal full triggers on n trigger coordinates is*

$$C(n, m) = \sum_{h=0}^{n-m} \binom{n}{h} S(n-h, m)(2^m - m - 1)^h,$$

where $S(n-h, m)$ is the Stirling number of the second kind.

Proof. Choose the h shared coordinates in $\binom{n}{h}$ ways. Partition the remaining $n-h$ private coordinates into m nonempty private blocks in $S(n-h, m)$ ways. For each shared coordinate, choose a row set of size at least two. The number of such row sets is

$$\sum_{r=2}^{m} \binom{m}{r} = 2^m - m - 1.$$

Summing over all possible h gives the formula. □

Corollary 9462.1700 (Codon counts from decorated partitions)**.** *For $n = 4$, the decoration formula gives*

$$C(4, 1) = 1, \qquad C(4, 2) = 25, \qquad C(4, 3) = 22, \qquad C(4, 4) = 1.$$

With two f_3-lifts per support, the codon-level counts are

$$2, \quad 100, \quad 176, \quad 16.$$

Proof. For $m = 2$, the shared coordinate has only one possible row set, so

$$C(4,2) = S(4,2) + 4S(3,2) + 6S(2,2) = 7 + 12 + 6.$$

For $m = 3$, one shared coordinate has four possible row sets, giving

$$C(4,3) = S(4,3) + 4S(3,3) \cdot 4 = 6 + 16.$$

The cases $m = 1$ and $m = 4$ have no shared coordinates. Multiplying the size-m quotient count by 2^m gives the codon counts. □

Theorem 9462.1701 (Defect from shared coordinates)**.** *For a minimal full trigger with private-shared decomposition, the overlap defect is*

$$\delta(A) = \sum_{a \in H} (|R_a| - 1).$$

Proof. Private coordinates contribute one occurrence and hence no excess over the four-coordinate cover. A shared coordinate a appears in $|R_a|$ supports, while one occurrence would suffice for coverage. Its excess contribution is therefore $|R_a| - 1$. Summing over shared coordinates gives the formula. □

Theorem 9462.1702 (Defect-refined decoration polynomial)**.** *Put*

$$A_m(y) = \sum_{r=2}^{m} \binom{m}{r} y^{r-1}.$$

Then the quotient-level defect polynomial for size-m minimal triggers on n coordinates is

$$D_{n,m}(y) = \sum_{h=0}^{n-m} \binom{n}{h} S(n-h, m) A_m(y)^h.$$

Proof. For a shared coordinate, choosing a row set of size r contributes defect $r - 1$ and can be done in $\binom{m}{r}$ ways. Thus the defect enumerator for one shared coordinate is $A_m(y)$. The remaining choices are exactly those counted in Definition 9462.1699. □

Corollary 9462.1703 (Codon defect polynomials)**.** *For $n = 4$,*

$$D_{4,2}(y) = 7 + 12y + 6y^2,$$

and

$$D_{4,3}(y) = 6 + 12y + 4y^2.$$

Together with the $m = 1$ and $m = 4$ singleton terms, this gives quotient defect counts

$$\delta = 0 : 15, \qquad \delta = 1 : 24, \qquad \delta = 2 : 10.$$

After the two-lift codon factor, the counts are

$$94, \quad 144, \quad 56.$$

Proof. For $m = 2$, $A_2(y) = y$, so

$$D_{4,2}(y) = 7 + 12y + 6y^2.$$

For $m = 3$, $A_3(y) = 3y + y^2$, so

$$D_{4,3}(y) = 6 + 4(3y + y^2).$$

The codon counts are obtained by multiplying the size-m quotient terms by 2^m. □

Corollary 9462.1704 (Orbit decoration table)**.** *The nine orbit normal forms have the following private and shared decoration data:*

normal form	*private blocks*	*shared data*
$\{1234\}$	$\{1234\}$	$\varnothing$
$\{1, 234\}$	$\{1\}, \{234\}$	$\varnothing$
$\{12, 34\}$	$\{12\}, \{34\}$	$\varnothing$
$\{12, 234\}$	$\{1\}, \{34\}$	$2 : \{1, 2\}$
$\{123, 124\}$	$\{3\}, \{4\}$	$1 : \{1, 2\},\ 2 : \{1, 2\}$
$\{1, 2, 34\}$	$\{1\}, \{2\}, \{34\}$	$\varnothing$
$\{1, 24, 34\}$	$\{1\}, \{2\}, \{3\}$	$4 : \{2, 3\}$
$\{14, 24, 34\}$	$\{1\}, \{2\}, \{3\}$	$4 : \{1, 2, 3\}$
$\{1, 2, 3, 4\}$	$\{1\}, \{2\}, \{3\}, \{4\}$	$\varnothing$.

Proof. For each normal form, read off coordinates of multiplicity one as private blocks and coordinates of multiplicity at least two as shared decorations with their row sets. □

Theorem 9462.1705 (Private-shared decoration theorem)**.** *Post-$M^\star$ quotient-level minimal full triggers are decorated set partitions: a nonempty private partition of the non-shared coordinates, plus shared coordinates decorated by row sets of size at least two. For n coordinates and ℓ lifts per support, the lifted size-m count is*

$$\ell^m \sum_{h=0}^{n-m} \binom{n}{h} S(n-h, m)(2^m - m - 1)^h.$$

The defect-refined quotient polynomial is

$$\sum_{h=0}^{n-m} \binom{n}{h} S(n-h, m) \left(\sum_{r=2}^{m} \binom{m}{r} y^{r-1} \right)^h.$$

Proof. The decomposition is Definition 9462.1698; the enumeration and defect formulas are Definition 9462.1699 and Definition 9462.1702. The lift factor is ℓ^m because each support in a size-m quotient trigger has ℓ independent lift choices. □

Concretized in §5542: Private/shared coordinate decomposition is realized as finite support, private-block, shared-coordinate, row-set, hsame, Cont, Pkg, and NameCert data.

Concretized in §5896: Private/shared decoration models are realized as finite private-block, shared-row, lift-count, defect-ledger, transport, replay, provenance, and NameCert rows.

Concretized in §5881: The private/shared decoration packet realizes the window-six boundary tile decomposition and decoration enumeration as finite NameCert rows.

Concretized in §7630: The defect-refined decoration polynomial is realized as finite private/shared row-set arity, exponent, coefficient, transport, replay, provenance, and NameCert rows.

Definition 9462.1706 (Codon-level lift orbit)**.** *Let $A = \{S_1, \ldots, S_m\}$ be a quotient-level minimal full trigger. The codon-level lift fiber over A is $\{0,1\}^m$. The natural symmetry group is*

$$G = S_4 \times C_2^{(f_3)},$$

where S_4 permutes the four trigger coordinates and $C_2^{(f_3)}$ acts on a lift vector by global complement

$$\epsilon \longmapsto 1 - \epsilon.$$

For a fixed support family A, let $\mathrm{Aut}(A)$ be the coordinate stabilizer of A, acting on the m support rows. The lift-orbit set over A is

$$\{0,1\}^m / (\mathrm{Aut}(A) \times C_2^{(f_3)}).$$

Theorem 9462.1707 (Codon-level lift-orbit classification)**.** *Under the action of $S_4 \times C_2^{(f_3)}$, the 294 codon-level minimal full triggers split into 20 orbit types.*

Proof. The quotient-level support families split into the nine orbit types of Definition 9462.1681. For each representative A, the remaining problem is the binary coloring quotient

$$\{0,1\}^{|A|} / (\mathrm{Aut}(A) \times C_2^{(f_3)}).$$

Computing these nine finite quotients gives the orbit counts

support normal form	quotient count	lift orbits
$\{1234\}$	1	1
$\{1, 234\}$	4	2
$\{12, 34\}$	3	2
$\{12, 234\}$	12	2
$\{123, 124\}$	6	2
$\{1, 2, 34\}$	6	3
$\{1, 24, 34\}$	12	3
$\{14, 24, 34\}$	4	2
$\{1, 2, 3, 4\}$	1	3.

The sum of the lift-orbit counts is

$$1 + 2 + 2 + 2 + 2 + 3 + 3 + 2 + 3 = 20.$$

□

Theorem 9462.1708 (Lift-orbit table)**.** *The codon-level orbit types are exactly the rows in the*

following table:

support normal form	*lift pattern*	*codon orbit size*
$\{1234\}$	0/1	2
$\{1, 234\}$	*same*	8
$\{1, 234\}$	*opposite*	8
$\{12, 34\}$	*same*	6
$\{12, 34\}$	*opposite*	6
$\{12, 234\}$	*same*	24
$\{12, 234\}$	*opposite*	24
$\{123, 124\}$	*same*	12
$\{123, 124\}$	*opposite*	12
$\{1, 2, 34\}$	*all same*	12
$\{1, 2, 34\}$	*pair same, block opposite*	12
$\{1, 2, 34\}$	*singleton split*	24
$\{1, 24, 34\}$	*all same*	24
$\{1, 24, 34\}$	*shared-pair same, singleton opposite*	24
$\{1, 24, 34\}$	*shared-pair split*	48
$\{14, 24, 34\}$	*all same*	8
$\{14, 24, 34\}$	*mixed*	24
$\{1, 2, 3, 4\}$	*all same*	2
$\{1, 2, 3, 4\}$	*one-vs-three*	8
$\{1, 2, 3, 4\}$	*two-vs-two*	6.

The orbit sizes add to 294.

Proof. For each support normal form, the row stabilizer $\mathrm{Aut}(A)$ acts on binary lift vectors, while $C_2^{(f_3)}$ identifies a vector with its complement. The $m = 1$ fiber has one orbit. For $m = 2$, the complement quotient leaves the same and opposite lift patterns. For the type $\{1, 2, 34\}$, the two singleton rows may be exchanged, giving the three patterns displayed in the table. For the type $\{1, 24, 34\}$, the two shared-pair rows may be exchanged, giving the analogous three patterns. For the type $\{14, 24, 34\}$, the three rows are symmetric, so the nonconstant patterns form one orbit. For $\{1, 2, 3, 4\}$, the four rows are symmetric, and complement identifies weights w and $4 - w$, leaving the three classes 0/4, 1/3, and 2/2. Multiplying each lift-orbit size by the number of quotient triggers in its support orbit gives the displayed codon orbit sizes. □

Theorem 9462.1709 (Burnside lift-orbit formula)**.** *For a quotient-level minimal trigger A, let $\Gamma = \mathrm{Aut}(A)$. If $c(\gamma)$ is the number of cycles of the row permutation γ, then the number of lift orbits over A is*

$$N_{\mathrm{lift}}(A) = \frac{1}{2|\Gamma|} \sum_{\gamma \in \Gamma} \left(2^{c(\gamma)} + \mathbf{1}_{\mathrm{even}}(\gamma) 2^{c(\gamma)} \right),$$

where $\mathbf{1}_{\mathrm{even}}(\gamma) = 1$ exactly when every cycle of γ has even length.

Proof. Burnside's lemma applies to the action of $\Gamma \times C_2^{(f_3)}$ on $\{0, 1\}^m$. A row permutation γ fixes $2^{c(\gamma)}$ binary strings. The complemented element $\tau\gamma$ fixes the solutions of $\epsilon = 1 - \gamma\epsilon$. Along a cycle, such assignments exist exactly for even cycle length, and then contribute two assignments per cycle. Hence the complemented fixed-point term is $2^{c(\gamma)}$ when all cycles are even and zero otherwise. □

Definition 9462.1710 (Lift imbalance)**.** *For a lift vector $\epsilon \in \{0,1\}^m$, define*

$$b(\epsilon) = \min\{w(\epsilon), m - w(\epsilon)\},$$

where $w(\epsilon)$ is its Hamming weight.

Theorem 9462.1711 (Codon-level minimal trigger orbit theorem)**.** *The post-$M^\star$ codon-level minimal trigger space is the f_3-lift hypercube bundle*

$$\mathcal{T}_c = Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4,$$

and its natural $S_4 \times C_2^{(f_3)}$ quotient has 20 orbit types. Each orbit type is determined by a support normal form together with a lift coloring orbit modulo support automorphisms and global f_3 complement.

Proof. The bundle decomposition is Definition 9462.1687. The support orbit types are supplied by Definition 9462.1681. The lift-coloring quotient and the count of 20 orbit types are Definition 9462.1707 and Definition 9462.1708. □

Concretized in §4059: Codon lift-orbit classifiers realize the finite support, binary lift, support-automorphism, complement-action, Burnside ledger, classifier, transport, replay, provenance, and NameCert rows behind the twenty orbit types.

Definition 9462.1712 (Orbit inventory variables)**.** *For a codon-level minimal full trigger (A, ϵ), write $m = |A|$ for its size, write*

$$\delta(A) = \sum_{S \in A} |S| - 4$$

for its overlap defect, and write

$$b(\epsilon) = \min\{w(\epsilon), m - w(\epsilon)\}$$

for its f_3-lift imbalance. The variable u records m, the variable v records δ, and the variable w records b.

Definition 9462.1713 (Orbit and codon inventory polynomials)**.** *The orbit inventory polynomial counts symmetry-normal forms:*

$$\mathcal{O}(u,v,w) = \sum_{\mathcal{Q}} u^{m(\mathcal{Q})} v^{\delta(\mathcal{Q})} w^{b(\mathcal{Q})},$$

where $\mathcal{Q}$ ranges over codon-level orbit types. The codon inventory polynomial counts actual lifted triggers:

$$\mathcal{C}(u,v,w) = \sum_{\mathcal{Q}} |\mathcal{Q}| u^{m(\mathcal{Q})} v^{\delta(\mathcal{Q})} w^{b(\mathcal{Q})}.$$

Theorem 9462.1714 (Orbit inventory polynomial)**.** *For the post-$M^\star$ codon-level minimal full triggers,*

$$\begin{aligned}\mathcal{O}(u,v,w) = u &+ u^2\big((2+2w) + v(1+w) + v^2(1+w)\big) \\ &+ u^3\big((1+2w) + v(1+2w) + v^2(1+w)\big) + u^4(1+w+w^2).\end{aligned}$$

Thus $\mathcal{O}(1,1,1) = 20$.

Proof. Group the 20 rows of Definition 9462.1708 by the triple (m, δ, b). Each orbit row contributes one monomial $u^m v^\delta w^b$. Summing the grouped rows gives the displayed polynomial, and substituting $u = v = w = 1$ counts the orbit types. □

Theorem 9462.1715 (Codon inventory polynomial)**.** *The corresponding codon-count polynomial is*

$$\begin{aligned}\mathcal{C}(u, v, w) = {} & 2u + u^2(1 + w)(14 + 24v + 12v^2) \\ & + u^3\big((12 + 36w) + v(24 + 72w) + v^2(8 + 24w)\big) \\ & + u^4(2 + 8w + 6w^2).\end{aligned}$$

Thus $\mathcal{C}(1, 1, 1) = 294$.

Proof. Use the same grouping as Definition 9462.1714, but weight each orbit row by its codon orbit size. The orbit sizes in Definition 9462.1708 sum to 294. □

Corollary 9462.1716 (Inventory marginals)**.** *The codon-count marginals are*

$$\mathcal{C}(u, 1, 1) = 2u + 100u^2 + 176u^3 + 16u^4,$$

$$\mathcal{C}(1, v, 1) = 94 + 144v + 56v^2,$$

and

$$\mathcal{C}(1, 1, w) = 98 + 190w + 6w^2.$$

Proof. These are the three one-variable specializations of Definition 9462.1715. □

Corollary 9462.1717 (Inventory joint tables)**.** *The size-defect codon counts are*

m	$\delta = 0$	$\delta = 1$	$\delta = 2$	*total*
1	2	0	0	2
2	28	48	24	100
3	48	96	32	176
4	16	0	0	16.

The defect-imbalance counts are

δ	$b = 0$	$b = 1$	$b = 2$	*total*
0	30	58	6	94
1	48	96	0	144
2	20	36	0	56.

The size-imbalance counts are

m	$b = 0$	$b = 1$	$b = 2$	*total*
1	2	0	0	2
2	50	50	0	100
3	44	132	0	176
4	2	8	6	16.

Proof. Set one of the variables in Definition 9462.1715 equal to 1 and read off the remaining bivariate coefficients. □

Corollary 9462.1718 (Inventory means and largest orbit)**.** *Under the uniform distribution on the* 294 *codon-level minimal full triggers,*

$$\mathbb{E}[m] = \frac{397}{147}, \qquad \mathbb{E}[\delta] = \frac{128}{147},$$
$$\mathbb{E}[E] = \frac{716}{147}, \qquad \mathbb{E}[b] = \frac{101}{147}.$$

The largest codon-level orbit has size 48*; it is the* $\{1, 24, 34\}$ *support orbit with shared-pair split lift pattern, hence has* $(m, \delta, b) = (3, 1, 1)$*.*

Proof. The expectations are obtained from the first logarithmic moments of $\mathcal{C}$ at $u = v = w = 1$, equivalently from the numerator sums

$$\sum m = 794, \qquad \sum \delta = 256, \qquad \sum E = 1432, \qquad \sum b = 202$$

over the 294 triggers. The largest orbit is read from Definition 9462.1708. □

Theorem 9462.1719 (Orbit inventory polynomial theorem)**.** *The post-*$M^\star$ *codon-level minimal trigger space is encoded by the two inventory polynomials* $\mathcal{O}(u, v, w)$ *and* $\mathcal{C}(u, v, w)$*. The first records the* 20 *natural orbit types, and the second records the* 294 *lifted codon triggers with their joint statistics* (m, δ, b)*.*

Proof. The orbit classification is Definition 9462.1707. Grouping those orbit rows by size, overlap defect, and lift imbalance gives Definition 9462.1714; weighting by orbit size gives Definition 9462.1715. □

Concretized in §4220: finite orbit-inventory polynomial certificate

Definition 9462.1720 (Defect-lowering move)**.** *Let* $A = \{S_1, \ldots, S_m\}$ *be a quotient-level minimal full trigger with private-shared decomposition as in Definition 9462.1698. If* $a \in H$ *is a shared coordinate and* $i \in R_a$*, a defect-lowering move deletes the single incidence* $a \in S_i$*, equivalently replacing*

$$R_a \quad \text{by} \quad R_a \setminus \{i\},$$

under the condition that $R_a \setminus \{i\}$ *remains nonempty.*

Theorem 9462.1721 (Defect lowering preserves minimality)**.** *If a quotient-level minimal full trigger admits a defect-lowering move, then the result is again a quotient-level minimal full trigger, and the defect decreases by one.*

Proof. Only one shared-coordinate incidence is removed. Since the coordinate remains in at least one other support, the union of the supports remains Ω. Since the removed coordinate was shared, no private block P_i changes. Hence every support still has a private coordinate, so Definition 9462.1672 gives minimality. The formula

$$\delta(A) = \sum_{a \in H} (|R_a| - 1)$$

from Definition 9462.1701 shows that removing one shared incidence decreases δ by one. □

Concretized in §4236: The defect-flow partition skeleton is realized as finite cubical-incidence and boundary-flow NameCert data.

Theorem 9462.1722 (Defect flow to the partition skeleton)**.** *Every quotient-level minimal full trigger flows by repeated defect-lowering moves to a partition trigger. The number of moves in any maximal decreasing path is exactly the initial defect.*

Proof. Each defect-lowering move reduces δ by one and never makes δ negative. Hence the process terminates after exactly the initial value of δ. At termination, no shared incidence remains removable, so every coordinate occurs in exactly one support. The supports therefore form a partition of Ω. □

Corollary 9462.1723 (Partition sinks)**.** *The sink set of the defect flow is the partition trigger skeleton. It has* 15 *quotient-level vertices and* 94 *codon-level vertices.*

Proof. The quotient sink count is the Bell number $B_4 = 15$. With two f_3-lifts per support, the codon sink count is

$$2S(4,1) + 2^2S(4,2) + 2^3S(4,3) + 2^4S(4,4) = 94.$$

□

Theorem 9462.1724 (Defect-flow edge counts)**.** *The quotient-level defect-flow graph has* 49 *vertices and* 84 *directed edges. The codon-level defect-flow graph has* 294 *vertices and* 480 *directed edges. The edge counts by layer are*

layer	*quotient edges*	*codon edges*
$\delta = 1 \to 0$	48	288
$\delta = 2 \to 1$	36	192.

Proof. The defect-1 quotient orbits are $\{12, 234\}$ and $\{1, 24, 34\}$. Each has one shared coordinate covered by two supports, hence outdegree two. Their quotient contributions are $12 \cdot 2$ and $12 \cdot 2$, giving 48 quotient edges; after lift multiplication they contribute $48 \cdot 2$ and $96 \cdot 2$, giving 288 codon edges.

The defect-2 quotient orbits are $\{123, 124\}$ and $\{14, 24, 34\}$. The first has two shared coordinates of row-size two, hence outdegree four; the second has one shared coordinate of row-size three, hence outdegree three. Their quotient contributions are $6 \cdot 4$ and $4 \cdot 3$, giving 36 quotient edges; after lift multiplication they contribute $24 \cdot 4$ and $32 \cdot 3$, giving 192 codon edges. □

Theorem 9462.1725 (Partition-core incidence)**.** *For a minimal full trigger A with shared-coordinate row sets R_a, the number of partition cores below A is*

$$N_{\text{core}}(A) = \prod_{a \in H} |R_a|.$$

The total quotient-level trigger–partition incidence is 99*, and the total codon-level incidence is* 574*.*

Proof. For each shared coordinate a, a terminal partition core is obtained by choosing exactly one row of R_a to retain a. These choices are independent, giving the product formula.

By quotient orbit type, the incidences are

normal form	quotient count	N_{core}	contribution
$\{1234\}$	1	1	1
$\{1, 234\}$	4	1	4
$\{12, 34\}$	3	1	3
$\{12, 234\}$	12	2	24
$\{123, 124\}$	6	4	24
$\{1, 2, 34\}$	6	1	6
$\{1, 24, 34\}$	12	2	24
$\{14, 24, 34\}$	4	3	12
$\{1, 2, 3, 4\}$	1	1	1.

The quotient contributions sum to 99. Multiplying each row by its codon lift count gives codon contributions

$$2, 16, 12, 96, 96, 48, 192, 96, 16,$$

which sum to 574. □

Theorem 9462.1726 (Defect-flow theorem)**.** *The post-$M^\star$ minimal full trigger system carries a natural acyclic defect-lowering flow. Its height function is the overlap defect*

$$\delta \in \{0, 1, 2\},$$

its sinks are exactly the partition triggers, and the f_3-lift coordinates are passive along the flow.

Proof. Acyclicity follows because every directed edge lowers δ by one. The sink identification is Definition 9462.1722. The lift vector does not enter the definition of a defect-lowering move, so each quotient-level edge lifts horizontally across the corresponding f_3-fiber. □

Definition 9462.1727 (Partition core)**.** *Let A be a minimal full trigger. A partition core of A is a partition trigger P obtained from A by a maximal sequence of defect-lowering moves.*

Theorem 9462.1728 (Core count of a trigger)**.** *If A has shared-coordinate row sets R_a, then the number of partition cores below A is*

$$N_{\text{core}}(A) = \prod_{a \in H} |R_a|.$$

Proof. To obtain a terminal partition, each shared coordinate a must be assigned to exactly one of the supports in R_a. The choices for distinct shared coordinates are independent. □

Theorem 9462.1729 (Basin size of a partition core)**.** *Let $P = \{P_1, \ldots, P_m\}$ be a quotient-level partition trigger with block sizes $s_i = |P_i|$. The number of minimal triggers for which P is a partition core is*

$$B(P) = \prod_{i=1}^{m} \left((2^{m-1})^{s_i} - (2^{m-1} - 1)^{s_i} \right).$$

Proof. For a coordinate $a \in P_i$, a trigger above P must keep a in row i and may additionally place it in any subset of the other $m - 1$ rows. Thus a has 2^{m-1} decoration choices. To preserve the private-coordinate condition for row i, at least one coordinate in P_i must choose no additional row. Hence the number of legal decorations for block P_i is

$$(2^{m-1})^{s_i} - (2^{m-1} - 1)^{s_i}.$$

Multiplying over blocks gives the formula. □

Corollary 9462.1730 (Partition basin table)**.** *For four trigger coordinates, the quotient partition basins are*

partition type	*quotient partitions*	$B(P)$	*incidence*
4	1	1	1
$3+1$	4	7	28
$2+2$	3	9	27
$2+1+1$	6	7	42
$1+1+1+1$	1	1	$1.$

The quotient incidence total is 99*. At codon level the corresponding incidence counts are*

$$2, \quad 112, \quad 108, \quad 336, \quad 16,$$

which sum to 574*.*

Proof. Evaluate Definition 9462.1729 on the five partition types of a four-element set. At codon level, a partition with m blocks has 2^m lift choices. □

Corollary 9462.1731 (Balanced basin maximum)**.** *Among quotient partition types, the largest basin per partition core is the balanced type* $2+2$*, with basin size* 9*.*

Proof. The basin sizes in Definition 9462.1730 are $1, 7, 9, 7, 1$, so the maximum occurs at type $2+2$. □

Theorem 9462.1732 (Descending path count)**.** *For a minimal full trigger* A*, the number of maximal descending paths from* A *to the partition skeleton is*

$$N_{\text{path}}(A) = \delta(A)! \prod_{a \in H} |R_a|.$$

Proof. Fix a partition core below A. The multiset of duplicate incidences to be removed has cardinality $\delta(A)$, and any ordering of these removals gives a valid decreasing path to that fixed core. Hence each core receives $\delta(A)!$ paths. Multiplying by Definition 9462.1728 gives the formula. □

Theorem 9462.1733 (Total descending paths)**.** *The quotient-level defect flow has* 135 *maximal descending paths to the partition skeleton. The codon-level defect flow has* 766 *maximal descending paths.*

Proof. By quotient orbit type, the path contributions are

normal form	quotient count	paths per trigger	contribution
$\{1234\}$	1	1	1
$\{1, 234\}$	4	1	4
$\{12, 34\}$	3	1	3
$\{12, 234\}$	12	2	24
$\{123, 124\}$	6	8	48
$\{1, 2, 34\}$	6	1	6
$\{1, 24, 34\}$	12	2	24
$\{14, 24, 34\}$	4	6	24
$\{1, 2, 3, 4\}$	1	1	$1.$

These contributions sum to 135. Multiplying the same paths-per-trigger values by codon counts gives codon contributions

$$2, 16, 12, 96, 192, 48, 192, 192, 16,$$

which sum to 766. □

Theorem 9462.1734 (Partition basin theorem)**.** *The partition skeleton is an attracting incidence geometry for the post-$M^\star$ minimal trigger system. The trigger–partition correspondence has* 99 *quotient-level incidences and* 574 *codon-level incidences, and the defect flow has* 135 *quotient-level and* 766 *codon-level maximal descending paths to this skeleton.*

Proof. The incidence totals are Definition 9462.1725 and Definition 9462.1730. The descending path totals are Definition 9462.1733. Since a trigger can have more than one partition core, the basin relation is a correspondence rather than a function. □

Concretized in §4363: The finite partition-basin product formula and four-coordinate incidence rows are realized as block-size, decoration, defect, product, specialization, and incidence NameCert data.

Concretized in §??: Partition-basin finite counting is realized as concrete NameCert data.

Definition 9462.1735 (Partition basin defect polynomial)**.** *Let $P = \{P_1, \dots, P_m\}$ be a quotient-level partition core with block sizes $s_i = |P_i|$. Its basin defect polynomial is*

$$\mathcal{B}_P(q) = \sum_{A:\, P \text{ is a core of } A} q^{\delta(A)}.$$

Theorem 9462.1736 (Partition basin polynomial)**.** *For a partition core with block sizes $(s_1, \dots, s_m)$,*

$$\mathcal{B}_P(q) = \prod_{i=1}^{m} \left[(1+q)^{(m-1)s_i} - \left((1+q)^{m-1} - 1 \right)^{s_i} \right].$$

Proof. For a coordinate in block P_i, the core row i is fixed, while any subset of the other $m-1$ rows may be added. Choosing r additional rows contributes q^r, so one coordinate contributes $(1+q)^{m-1}$. A block of size s_i therefore has all decorations enumerated by $(1+q)^{(m-1)s_i}$. The private-coordinate condition excludes the case in which every coordinate of P_i receives at least one additional row; that excluded polynomial is $((1+q)^{m-1} - 1)^{s_i}$. Multiplying the legal block enumerators gives the displayed product. □

Corollary 9462.1737 (Four-coordinate basin polynomials)**.** *For four trigger coordinates, the partition-type basin polynomials are*

partition type	*quotient partitions*	$\mathcal{B}_P(q)$	$\mathcal{B}_P(1)$
4	1	1	1
$3+1$	4	$1+3q+3q^2$	7
$2+2$	3	$1+4q+4q^2$	9
$2+1+1$	6	$1+4q+2q^2$	7
$1+1+1+1$	1	1	1.

Proof. Substitute the five block-size types of a four-element set into Definition 9462.1736. □

Theorem 9462.1738 (Basin incidence polynomials)**.** *The quotient-level trigger–partition incidence polynomial is*

$$\mathcal{I}_q(q) = 15 + 48q + 36q^2.$$

The codon-level incidence polynomial is

$$\mathcal{I}_c(q) = 94 + 288q + 192q^2.$$

Thus $\mathcal{I}_q(1) = 99$ and $\mathcal{I}_c(1) = 574$.

Proof. Sum the rows of Definition 9462.1737, weighting by the number of quotient partitions of each type. For codon-level incidence, multiply a partition type with m blocks by 2^m lift cores. This gives

$$2 \cdot 1 + 16(1 + 3q + 3q^2) + 12(1 + 4q + 4q^2)$$
$$+48(1 + 4q + 2q^2) + 16 \cdot 1 = 94 + 288q + 192q^2.$$

□

Theorem 9462.1739 (Descending path polynomials)**.** *Weighting each defect-δ incidence by $\delta!$ gives*

$$\mathcal{P}_q(q) = 15 + 48q + 72q^2$$

and

$$\mathcal{P}_c(q) = 94 + 288q + 384q^2.$$

Thus $\mathcal{P}_q(1) = 135$ and $\mathcal{P}_c(1) = 766$.

Proof. For a fixed partition core below a trigger of defect δ, the duplicate incidences may be removed in $\delta!$ orders by Definition 9462.1732. Apply this factor to the coefficients of Definition 9462.1738. □

Corollary 9462.1740 (Basin-size averages)**.** *The codon-level partition skeleton has 94 cores and total incidence 574, so its average basin size is*

$$\frac{574}{94}.$$

The largest basin per core is the balanced type $2 + 2$, with size 9.

Proof. The average is total codon-level incidence divided by the number of codon-level partition cores. The maximum follows from Definition 9462.1737. □

Theorem 9462.1741 (Partition basin polynomial theorem)**.** *The defect-flow basins over the partition skeleton are completely enumerated by the product formula for $\mathcal{B}_P(q)$. In the codon case $(n, \ell) = (4, 2)$, the total codon-level basin incidence and descending path polynomials are*

$$\mathcal{I}_c(q) = 94 + 288q + 192q^2, \qquad \mathcal{P}_c(q) = 94 + 288q + 384q^2.$$

Proof. The product formula is Definition 9462.1736. The codon-level specialization is Definition 9462.1738 and Definition 9462.1739. □

Concretized in §4363: Partition basin polynomials are realized as finite block-size, decoration, defect, incidence, transport, replay, provenance, and NameCert rows.

Definition 9462.1742 (Partition-core incidence kernel)**.** *Let $\mathcal{P}_m$ be the set of m-block partition cores of $\Omega = \{1, 2, 3, 4\}$, and let $\mathcal{T}_m$ be the quotient-level minimal full triggers of size m. For $P \in \mathcal{P}_m$ and $A \in \mathcal{T}_m$, write $P \preceq A$ when the blocks of P can be matched to the supports of A so that every block is contained in its matched support. Equivalently, P is a partition core below A in the defect-lowering flow.*

The incidence matrix is

$$I_m(P, A) = \begin{cases} 1, & P \preceq A, \\ 0, & \text{otherwise}, \end{cases}$$

and the partition-core Gram kernel is

$$K_m = I_m I_m^{\mathsf{T}}.$$

Thus

$$K_m(P,Q) = |\{A \in \mathcal{T}_m : P \preceq A,\ Q \preceq A\}|.$$

Theorem 9462.1743 (Incidence decomposition by trigger size)**.** *The quotient-level partition-core incidence relation decomposes over $m = 1, 2, 3, 4$ with rows*

m	$\lvert\mathcal{P}_m\rvert$	$\lvert\mathcal{T}_m\rvert$	*incidences*
1	1	1	1
2	7	25	55
3	6	22	42
4	1	1	1.

The incidence total is 99.

Proof. A size-m trigger lowers only to an m-block partition core, so different values of m do not interact. The row counts are obtained by enumerating the partition cores and the quotient-level minimal triggers of Definition 9462.1530, then applying Definition 9462.1742. The total $1 + 55 + 42 + 1 = 99$ agrees with Definition 9462.1738. □

Theorem 9462.1744 (Two-block partition-core kernel)**.** *Order the two-block partition cores as*

$$1|234, \quad 2|134, \quad 3|124, \quad 4|123, \quad 12|34, \quad 13|24, \quad 14|23.$$

Then

$$K_2 = \begin{pmatrix} 7 & 1 & 1 & 1 & 3 & 3 & 3 \\ 1 & 7 & 1 & 1 & 3 & 3 & 3 \\ 1 & 1 & 7 & 1 & 3 & 3 & 3 \\ 1 & 1 & 1 & 7 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 & 9 & 2 & 2 \\ 3 & 3 & 3 & 3 & 2 & 9 & 2 \\ 3 & 3 & 3 & 3 & 2 & 2 & 9 \end{pmatrix}.$$

Equivalently,

$$K_2 = \begin{pmatrix} 6I_4 + J_4 & 3J_{4,3} \\ 3J_{3,4} & 7I_3 + 2J_3 \end{pmatrix}.$$

Proof. The first four rows are the $3+1$ partition cores and the last three rows are the $2+2$ partition cores. Direct evaluation of $K_2(P,Q) = |\{A : P \preceq A,\ Q \preceq A\}|$ gives the displayed matrix. The block expression groups equal entries by the two partition types. □

Theorem 9462.1745 (Spectrum of the two-block kernel)**.** *The spectrum of K_2 is*

$$22, \quad 7, 7, \quad 6, 6, 6, \quad 1.$$

The Perron eigenvector is constant on partition types, with relative weights

$$3 + 1 : 1, \qquad 2 + 2 : \frac{4}{3}.$$

Proof. In the block form of Definition 9462.1744, $6I_4 + J_4$ acts as 6 on the three-dimensional subspace orthogonal to $\mathbf{1}_4$, while $7I_3 + 2J_3$ acts as 7 on the two-dimensional subspace orthogonal to $\mathbf{1}_3$. The remaining two-dimensional quotient spanned by vectors constant on the $3+1$ and $2+2$ parts has eigenvalues 22 and 1. Solving the Perron equations on that quotient gives weight ratio $1 : 4/3$. □

Theorem 9462.1746 (Three-block partition-core kernel)**.** *Order the three-block partition cores as*

$$12|3|4, \quad 13|2|4, \quad 14|2|3, \quad 23|1|4, \quad 24|1|3, \quad 34|1|2.$$

Then

$$K_3 = \begin{pmatrix} 7 & 2 & 2 & 2 & 2 & 0 \\ 2 & 7 & 2 & 2 & 0 & 2 \\ 2 & 2 & 7 & 0 & 2 & 2 \\ 2 & 2 & 0 & 7 & 2 & 2 \\ 2 & 0 & 2 & 2 & 7 & 2 \\ 0 & 2 & 2 & 2 & 2 & 7 \end{pmatrix}.$$

Equivalently,

$$K_3 = 7I_6 + 2A(L(K_4)),$$

where the six partition cores are identified with the six edges of K_4.

Proof. Each three-block core has type $2 + 1 + 1$. Two such cores share the off-diagonal value 2 exactly when their two-element blocks share one coordinate, and share value 0 when those two-element blocks are disjoint. This is precisely the adjacency relation in the line graph $L(K_4)$, with diagonal basin size 7. □

Theorem 9462.1747 (Spectrum of the three-block kernel)**.** *The spectrum of K_3 is*

$$15, \quad 7, 7, 7, \quad 3, 3.$$

The Perron eigenvector is the constant vector.

Proof. The line graph $L(K_4)$ has adjacency spectrum

$$4, \quad 0, 0, 0, \quad -2, -2.$$

Since $K_3 = 7I_6 + 2A(L(K_4))$, the eigenvalues are $7 + 2 \cdot 4 = 15$, $7 + 2 \cdot 0 = 7$ with multiplicity 3, and $7 + 2(-2) = 3$ with multiplicity 2. The graph $L(K_4)$ is regular, so the Perron eigenvector is constant. □

Theorem 9462.1748 (Partition-core incidence kernel theorem)**.** *The partition skeleton of the post-$M^\star$ minimal trigger system carries a size-graded Gram kernel. Its nontrivial quotient-level spectra are*

$$\mathrm{Spec}(K_2) = \{22, 7, 7, 6, 6, 6, 1\},$$

and

$$\mathrm{Spec}(K_3) = \{15, 7, 7, 7, 3, 3\}.$$

At codon level the incidence counts are

$$2, \quad 220, \quad 336, \quad 16,$$

which sum to 574.

Proof. The size decomposition is Definition 9462.1743. The two nontrivial spectra are Definition 9462.1745 and Definition 9462.1747. A size-m quotient incidence has 2^m codon lifts, giving $2 \cdot 1$, $4 \cdot 55$, $8 \cdot 42$, and $16 \cdot 1$. □

Concretized in §4214: Finite partition-core incidence rows are realized as a concrete NameCert carrier.

Definition 9462.1749 (S_4 representation labels)**.** *Let S_4 act on the trigger coordinate set $\Omega = \{s_1, f_1, s_2, f_2\}$. Its irreducible real representations are denoted by*

symbol	*dimension*
$\mathbf{1}$	1
sgn	1
V	3
$V \otimes \mathrm{sgn}$	3
W	2.

Here V is the standard representation and W is the two-dimensional representation associated to the partition $2 + 2$.

Theorem 9462.1750 (Partition-core permutation modules)**.** *The S_4 permutation representations on m-block partition cores are*

m	$\mathbb{R}[\mathcal{P}_m]$
1	$\mathbf{1}$
2	$2\mathbf{1} \oplus V \oplus W$
3	$\mathbf{1} \oplus V \oplus W$
4	$\mathbf{1}$.

In particular, neither sgn *nor $V \otimes \mathrm{sgn}$ appears on the partition-core side.*

Proof. Compute the permutation character by counting fixed set partitions in each conjugacy class of S_4 and decompose against the character table of Definition 9462.1749. For $m = 2$, the partition cores split into the two orbits $3 + 1$ and $2 + 2$, giving two copies of the trivial representation. For $m = 3$, all cores have type $2 + 1 + 1$, giving only one copy of the trivial representation. □

Theorem 9462.1751 (Minimal-trigger permutation modules)**.** *The S_4 permutation representations on quotient-level size-m minimal full triggers are*

m	$\mathbb{R}[\mathcal{T}_m]$
1	$\mathbf{1}$
2	$4\mathbf{1} \oplus 4V \oplus (V \otimes \mathrm{sgn}) \oplus 3W$
3	$3\mathbf{1} \oplus 4V \oplus (V \otimes \mathrm{sgn}) \oplus 2W$
4	$\mathbf{1}$.

Proof. For each conjugacy class of S_4, count the size-m minimal full support families fixed by a representative coordinate permutation. The resulting permutation characters decompose against the same S_4 character table. The dimensions are $1, 25, 22, 1$, agreeing with Definition 9462.1530. □

Theorem 9462.1752 (Incidence kernel representation)**.** *The incidence map*

$$I_m : \mathbb{R}[\mathcal{T}_m] \longrightarrow \mathbb{R}[\mathcal{P}_m]$$

is S_4-equivariant. Its kernel representations are

m	$\ker I_m$
1	0
2	$\mathbf{21} \oplus 3V \oplus (V \otimes \mathrm{sgn}) \oplus 2W$
3	$\mathbf{21} \oplus 3V \oplus (V \otimes \mathrm{sgn}) \oplus W$
4	0.

Thus $V \otimes \mathrm{sgn}$ is entirely invisible to partition-core incidence.

Proof. Equivariance follows because relabeling coordinates commutes with the condition $P \preceq A$. The image is the full partition-core module in each nonzero size layer by Definition 9462.1743; hence the kernel multiplicities are obtained by subtracting Definition 9462.1750 from Definition 9462.1751. Since $V \otimes \mathrm{sgn}$ does not occur on the partition side, every such trigger-side component lies in the kernel. □

Theorem 9462.1753 (Representation meaning of the Gram spectra)**.** *The spectra of the partition-core Gram kernels are the S_4 isotypic decomposition of the partition-core modules:*

m	*eigenvalue*	*representation*
2	22, 1	*trivial two-plane*
2	7	W
2	6	V
3	15	$\mathbf{1}$
3	7	V
3	3	W.

Proof. The kernels $K_m = I_m I_m^{\mathsf{T}}$ commute with the S_4 action on $\mathbb{R}[\mathcal{P}_m]$. By Schur's lemma, they act by scalars on each irreducible isotypic summand, except that repeated trivial summands may mix. For $m = 2$, the two partition orbits $3+1$ and $2+2$ give a two-dimensional trivial sector whose eigenvalues are 22 and 1 by Definition 9462.1745; the W and V sectors carry eigenvalues 7 and 6. For $m = 3$, there is only one trivial summand, and Definition 9462.1747 gives the eigenvalues 15, 7, and 3 on $\mathbf{1}$, V, and W. □

Corollary 9462.1754 (Balanced bias and uniformity)**.** *The $m = 2$ Perron vector favors $2 + 2$ partition cores over $3 + 1$ partition cores with relative weight $4/3 : 1$. The $m = 3$ Perron vector is uniform.*

Proof. In the $m = 2$ layer, the two copies of $\mathbf{1}$ mix across the two partition-core orbits; the Perron vector in Definition 9462.1745 has relative weights $3+1 : 1$ and $2+2 : 4/3$. In the $m = 3$ layer, the partition-core representation has a single trivial summand, so the Perron vector is the constant vector by Definition 9462.1747. □

Theorem 9462.1755 (Representation-theoretic incidence theorem)**.** *For the full quotient-level partition-core incidence map*

$$I : \mathbb{R}[\mathcal{T}] \to \mathbb{R}[\mathcal{P}],$$

the source, target, and kernel decompose as

$$\mathbb{R}[\mathcal{P}] = 5\mathbf{1} \oplus 2V \oplus 2W,$$

$$\mathbb{R}[\mathcal{T}] = 9\mathbf{1} \oplus 8V \oplus 2(V \otimes \mathrm{sgn}) \oplus 5W,$$

and

$$\ker I = 4\mathbf{1} \oplus 6V \oplus 2(V \otimes \mathrm{sgn}) \oplus 3W.$$

The partition-basin spectra of Definition 9462.1748 are therefore representation-theoretic spectra.

Proof. Sum the four size layers in Definition 9462.1750, Definition 9462.1751, and Definition 9462.1752. The dimensions are 15, 49, and 34, with $49 - 15 = 34$. The spectral identification follows from Definition 9462.1753. □

Concretized in §4017: The finite partition-core incidence packet records the boundary-tile incidence kernel as NameCert data.

Definition 9462.1756 (BEDC local suffix support rank)**.** *On the stop face*

$$\mathsf{StopTrpFace} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\},$$

use UAA *as the local basepoint and read only the family coordinates of the second and third positions. Thus the suffix coordinate map is*

codon	*family suffix*	*support*	ρ_{BEDC}
UAA	00	$\varnothing$	0
UAG	01	$\{f_3\}$	1
UGA	10	$\{f_2\}$	1
UGG	11	$\{f_2, f_3\}$	2.

The local BEDC rank is the support size

$$\rho_{\mathrm{BEDC}}(c) = |\{i \in \{f_2, f_3\} : c_i \textit{ differs from the } \mathrm{UAA} \textit{ suffix coordinate}\}|.$$

Theorem 9462.1757 (Stop L-shape as rank deficiency)**.** *Under the two-axis readback and the standard-code biological bridge, the stop class in the* $\mathrm{U} + \mathrm{RR}$ *face is the proper-support part of the local two-coordinate square:*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} = \mathsf{StopTrpFace} \cap \{c : \rho_{\mathrm{BEDC}}(c) < 2\}.$$

Equivalently, the local BEDC legality condition is

$$\mathrm{supp}_{\mathrm{BEDC}}(c) \subsetneq \{f_2, f_3\}.$$

Proof. The suffix table in Definition 9462.1756 identifies UAA, UAG, and UGA with supports $\varnothing$, $\{f_3\}$, and $\{f_2\}$, respectively. These are exactly the proper subsets of $\{f_2, f_3\}$. The remaining corner UGG has full support $\{f_2, f_3\}$. The standard-code bridge labels the proper-support vertices as Stop and the full-support corner as Trp by Definition 9462.1154. □

Theorem 9462.1758 (Completion of the stop L-shape in the median closure)**.** *Let*

$$S_{\mathrm{BEDC}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

Then

$$S_{\mathrm{BEDC}} \subset \mathcal{R}, \qquad \mathrm{UGG} \notin \mathcal{R},$$

but

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\} \subset \mathrm{Med}(\mathcal{R}).$$

In particular,

$$\mathrm{Med}(\mathcal{R}) \cap \mathsf{StopTrpFace} = \mathsf{StopTrpFace}.$$

Proof. The inclusion $S_{\mathrm{BEDC}} \subset \mathcal{R}$ and the exclusion $\mathrm{UGG} \notin \mathcal{R}$ are read from Definition 9462.1275. By Definition 9462.1320,

$$\mathrm{Med}(\mathcal{R}) = \mathrm{WNR} \cup \mathrm{CUN}.$$

Since $\mathsf{StopTrpFace} = \mathrm{URR}$ and $\mathrm{URR} \subset \mathrm{WNR}$, the whole four-point face lies in $\mathrm{Med}(\mathcal{R})$. □

Corollary 9462.1759 (Full-support corners have separated bridge roles)**.** *The full-support corner of the BEDC stop face is a retained sense corner:*

$$\mathrm{UGG} \in \mathrm{Med}(\mathcal{R}) \setminus \mathcal{R}.$$

The full-support corner of the post-$M^\star$ terminal trigger algebra is a global median trigger:

$$\mathrm{Med}(M^\star \cup \{\mathrm{AGU}\}) = Q_6, \qquad \mathrm{Med}(M^\star \cup \{\mathrm{AGC}\}) = Q_6.$$

Proof. The first statement follows from Definition 9462.1758. For the second, the two codons AGU and AGC have anchor support $\{s_1, f_1, s_2, f_2\}$ relative to CUY. Thus Definition 9462.1522 gives full-cube median closure. □

Theorem 9462.1760 (Rank-deficiency prototype)**.** *The local BEDC no-11 stop rule and the post-$M^\star$ blocker rule are the same proper-support law in two dimensions. Under the support identification $00 \leftrightarrow \varnothing$, $01 \leftrightarrow \{f_3\}$, $10 \leftrightarrow \{f_2\}$, and $11 \leftrightarrow \{f_2, f_3\}$,*

$$n = 2: \qquad X_2 = \{S : S \subsetneq \{f_2, f_3\}\},$$

and

$$n = 4: \qquad \mathcal{B} = \{B \subseteq Q_6 \setminus M^\star : J(B) \subsetneq \{s_1, f_1, s_2, f_2\}\}.$$

Thus the BEDC stop L-shape is the two-coordinate local prototype of the four-coordinate terminal trigger blocker algebra.

Proof. The $n = 2$ equality is Definition 9462.1757. The $n = 4$ equality is the definition of the blocker complex in Definition 9462.1547. Both say that proper supports are non-triggering or legal at their respective scale, while full support is the separated corner. □

Corollary 9462.1761 (Local and terminal blocker spheres)**.** *The local proper-support blocker for two coordinates has nerve $\partial\Delta^1 \cong S^0$, while the post-$M^\star$ blocker complex has homotopy type S^2:*

$$n = 2: \ S^0, \qquad n = 4: \ S^2.$$

Proof. For $n = 2$, the two maximal proper-support blockers omit f_2 and f_3, and their intersection is empty; the nerve is therefore two points. For $n = 4$, use Definition 9462.1552. □

Theorem 9462.1762 (BEDC rank-deficiency theorem)**.** *The BEDC stop-tile reading supplies a local rank-deficiency grammar:*

$$\textit{proper support in } \{f_2, f_3\} \Longleftrightarrow \mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}.$$

The post-$M^\star$ terminal trigger lattice supplies the global four-coordinate form:

$$B \textit{ is a non-trigger} \Longleftrightarrow J(B) \subsetneq \{s_1, f_1, s_2, f_2\}.$$

This is a conditional structural reconstruction inside the chosen two-axis readback and the external standard-code bridge. It is not a claim that Zeckendorf legality alone determines biological stop codons or that a full-support codon is biologically absent.

Proof. The local statement is Definition 9462.1757. The global statement is the trigger/blocker criterion of Definition 9462.1555. The final restriction is the cannot-claim boundary of Definition 9462.1206. □

Definition 9462.1763 (BEDC closure tower)**.** *The BEDC closure tower associated to the stop face is the chain*

$$\begin{gathered}\mathrm{U} = 00, \quad \mathrm{C} = 01, \quad \mathrm{A} = 10, \quad \mathrm{G} = 11,\\ \textit{proper support in } \{f_2, f_3\},\\ \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} \subset \mathcal{R},\\ \mathsf{StopTrpFace} \subset \mathrm{Med}(\mathcal{R}),\\ M^\star = \mathrm{NNR} \cup \mathrm{CUY},\\ \textit{proper support in } \{s_1, f_1, s_2, f_2\}.\end{gathered}$$

The first proper-support layer is the based local system obtained from the $(2, 1)$ *coverage-blocker algebra by adjoining the basepoint* $00 \leftrightarrow \varnothing$*. The terminal layer is the* $(4, 2)$ *coverage-trigger algebra of Definition 9462.1617.*

Theorem 9462.1764 (BEDC-to-trigger prototype theorem)**.** *The BEDC stop-face grammar is the based* $(n, \ell) = (2, 1)$ *instance of the coverage-blocker law, while the post-*$M^\star$ *terminal trigger lattice is the* $(n, \ell) = (4, 2)$ *instance. In both cases the non-triggering or legal condition is proper support, and the full-support state is the separated corner.*

Proof. For the local BEDC face, the non-base supports are $\{f_2\}$, $\{f_3\}$, and $\{f_2, f_3\}$, with one lift. The proper-support part, together with the basepoint $\varnothing \leftrightarrow 00$, gives $\{00, 01, 10\}$, while the full support $\{f_2, f_3\}$ gives the 11 corner. This is Definition 9462.1757. For the terminal codon system, the four support coordinates are s_1, f_1, s_2, f_2 and the two lifts are the two f_3-family codons over the Y side; this is Definition 9462.1617. The full-trigger criterion is Definition 9462.1522. □

Corollary 9462.1765 (Topology of the BEDC closure tower)**.** *The blocker topology in the tower moves from*

$$\mathcal{B}_{2,1} \simeq S^0$$

at the local BEDC layer to

$$\mathcal{B}_{4,2} \simeq S^2$$

at the terminal post-$M^\star$ *layer.*

Proof. Apply the universal formula $\mathcal{B}_{n,\ell} \simeq S^{n-2}$ from Definition 9462.1611. The local stop-face layer has $n = 2$ and $\ell = 1$, while the terminal codon layer has $n = 4$ and $\ell = 2$ by Definition 9462.1764. □

Theorem 9462.1766 (BEDC closure tower theorem)**.** *The stop-tile BEDC reading, the reassignment support, the median closure, and the post-*$M^\star$ *trigger lattice form the closure tower*

$$\begin{gathered}\mathrm{BEDC}_{2,1} \longrightarrow \mathrm{Stop/Trp}\ X_2/11 \longrightarrow \mathcal{R} \longrightarrow \mathrm{Med}(\mathcal{R})\\ \longrightarrow M^\star \longrightarrow \mathrm{BEDC}_{4,2}.\end{gathered}$$

Here $\mathrm{BEDC}_{2,1}$ *denotes the based local proper-support grammar on the stop suffix coordinates, and* $\mathrm{BEDC}_{4,2}$ *denotes the post-*$M^\star$ *coverage-trigger algebra on* $\{s_1, f_1, s_2, f_2\}$ *with two* f_3 *lifts.*

Proof. The local stop-tile realization is Definition 9462.1154 and Definition 9462.1757. The inclusion of the stop seed in $\mathcal{R}$ and the recovery of the full $\mathsf{StopTrpFace}$ inside $\mathrm{Med}(\mathcal{R})$ are Definition 9462.1758. The post-$M^\star$ terminal trigger algebra is the $(4, 2)$ coverage-trigger instance by Definition 9462.1617. The common proper-support law is Definition 9462.1764. □

Concretized in §4090: The local stop/Trp suffix-support rank theorem is realized as a finite NameCert packet with source-window, boundary-tile, coverage, orbit, rank, ledger, transport, replay, provenance, and local naming rows.

Concretized in §7841: The rank-deficiency prototype is realized as a finite support-rank NameCert packet with support-universe, proper-support classifier, deficiency ledger, coverage-rank sibling, rank-nullity sibling, transport, replay, provenance, and local naming rows.

Definition 9462.1767 (Active median generator)**.** *Let $M = \mathrm{Med}(\mathcal{R})$. A subset $G \subseteq \mathcal{R}$ is an active median generator when*

$$\mathrm{Med}(G) = M.$$

It is minimal when no smaller subset of $\mathcal{R}$ has the same median closure.

Theorem 9462.1768 (Minimal active median generators)**.** *The minimum size of an active median generator is 7. There are exactly two generators of this size:*

$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUA}, \mathrm{CUC}, \mathrm{CUU}, \mathrm{UAA}, \mathrm{UCA}\},$$
$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUC}, \mathrm{CUG}, \mathrm{CUU}, \mathrm{UAA}, \mathrm{UCA}\}.$$

Proof. Enumerate all subsets $G \subseteq \mathcal{R}$ and compute $\mathrm{Med}(G)$ by iterating coordinatewise majority. No subset of size at most 6 generates M, and the two displayed subsets are exactly the size-7 subsets that generate M. □

Corollary 9462.1769 (Rigid median skeleton)**.** *The common part of the two minimum generators is*

$$\{\mathrm{AGG}, \mathrm{AUA}, \mathrm{CUC}, \mathrm{CUU}, \mathrm{UAA}, \mathrm{UCA}\}.$$

The only choice is between CUA *and* CUG*.*

Proof. Intersect the two generators in Definition 9462.1768. □

Definition 9462.1770 (Median-essential codon)**.** *A codon $c \in \mathcal{R}$ is median-essential when*

$$\mathrm{Med}(\mathcal{R} \setminus \{c\}) \neq \mathrm{Med}(\mathcal{R}).$$

Otherwise it is median-redundant.

Theorem 9462.1771 (Median-essential codons)**.** *The median-essential codons are exactly*

$$\mathrm{AGG}, \quad \mathrm{AUA}, \quad \mathrm{CUC}, \quad \mathrm{CUU}, \quad \mathrm{UCA}.$$

The remaining active codons are median-redundant.

Proof. For each codon $c \in \mathcal{R}$ compute $\mathrm{Med}(\mathcal{R} \setminus \{c\})$ and compare it with M. The closure sizes are

c	$\lvert \mathrm{Med}(\mathcal{R} \setminus \{c\}) \rvert$	preserves M
AAA	20	yes
AGA	20	yes
AGG	14	no
AUA	16	no
CUA	20	yes
CUC	19	no
CUG	20	yes
CUU	19	no
UAA	20	yes
UAG	20	yes
UCA	16	no
UGA	20	yes
UUA	20	yes.

□

Theorem 9462.1772 (Median deletion losses)**.** *The median-essential codons have the following closure losses:*

c	$M \setminus \mathrm{Med}(\mathcal{R} \setminus \{c\})$
AGG	AAG, ACG, AGG, AUG, UCG, UGG
AUA	ACA, ACG, AUA, AUG
UCA	ACA, ACG, UCA, UCG
CUC	CUC
CUU	CUU.

Proof. Subtract each deletion closure from M for the five median-essential codons of Definition 9462.1771. □

Corollary 9462.1773 (Weight and median generation differ)**.** *High reassignment weight is not the same as median-generating importance. In particular* UGA*,* UAA*,* UAG*,* AGA*, and* AAA *are median-redundant, even though several of them have high reassignment weight.*

Proof. This is the redundant part of the deletion table in Definition 9462.1771. □

Theorem 9462.1774 (Complementary active pairs)**.** *The reassigned support contains exactly two complementary pairs:*

$$\{\mathrm{CUC}, \mathrm{AGA}\}, \qquad \{\mathrm{CUU}, \mathrm{AGG}\}.$$

Each pair has Hamming distance 6 *in* Q_6*.*

Proof. Using $U = 00$, $C = 01$, $A = 10$, and $G = 11$, the bit words of CUC and AGA are complementary, and the same holds for CUU and AGG. No other pair in $\mathcal{R}$ has all six bits opposite. □

Corollary 9462.1775 (Convex hull is too coarse)**.** *The geodesic convex hull of* $\mathcal{R}$ *is all of* Q_6*, already witnessed by either complementary pair in Definition 9462.1774.*

Proof. In a hypercube, the interval between complementary vertices is the whole cube. □

Definition 9462.1776 (Q_2 completion closure)**.** *For $S \subseteq Q_6$, define $\mathrm{Comp}_{Q_2}(S)$ by iterating the rule that whenever a Q_2 face contains at least three current vertices, the whole face is adjoined.*

Theorem 9462.1777 (Single-deletion completion robustness)**.** *For every active codon $c \in \mathcal{R}$,*

$$\mathrm{Comp}_{Q_2}(\mathcal{R} \setminus \{c\}) = Q_6.$$

Proof. Delete each of the 13 active codons and run the finite Q_2-completion iteration. Every deletion case reaches all 64 vertices. □

Theorem 9462.1778 (Minimal percolating seeds)**.** *Among subsets of $\mathcal{R}$, the minimum size of a Q_2-completion seed that percolates to Q_6 is 7. There are 484 such size-7 seeds.*

Proof. Enumerate all subsets $G \subseteq \mathcal{R}$ and compute $\mathrm{Comp}_{Q_2}(G)$. No subset of size at most 6 percolates to Q_6, while 484 subsets of size 7 do. □

Theorem 9462.1779 (Generator-closure robustness)**.** *The reassigned support is rigid in median generation and robust in Q_2-completion:*

$$\textit{minimum active median generators: } 2 \textit{ subsets of size } 7,$$
$$\textit{minimum } Q_2\textit{-percolating seeds: } 484 \textit{ subsets of size } 7.$$

Thus the median closure identifies a small rigid generating skeleton, whereas Q_2-completion records a broad percolation capacity.

Proof. Combine Definition 9462.1768, Definition 9462.1777, and Definition 9462.1778. □

Concretized in §3837: Generator closure robustness is realized as finite active-generator, deletion-loss, Q2-completion, and trigger-closure NameCert data.

Definition 9462.1780 (Purine-suffix hotspot cube)**.** *Let*

$$W := \{\mathrm{U}, \mathrm{A}\}, \qquad R := \{\mathrm{A}, \mathrm{G}\}.$$

The purine-suffix hotspot cube is the axis-aligned Q_3 cube

$$\begin{aligned} C_{\mathrm{WRR}} &:= W \times R \times R \\ &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}. \end{aligned}$$

In the six-bit atlas its free coordinates are

$$b_1^s, \qquad b_2^f, \qquad b_3^f,$$

with $b_1^f = 0$, $b_2^s = 1$, and $b_3^s = 1$ fixed.

Theorem 9462.1781 (Purine-suffix hotspot count)**.** *In the standard-to-comparative reassignment spectrum of Definition 9462.1208, C_{WRR} contains 52 of the 65 reassignment rows:*

codon	*standard label*	*reassignment rows*
UAA	Stop	8
UAG	Stop	10
UGA	Stop	15
UGG	Trp	0
AAA	Lys	3
AAG	Lys	0
AGA	Arg	8
AGG	Arg	8
total		52.

Proof. Restrict the finite reassignment enumeration of Definition 9462.1210 to the eight codons of Definition 9462.1780. Summing the displayed codon counts gives

$$8 + 10 + 15 + 0 + 3 + 0 + 8 + 8 = 52.$$

□

Concretized in §3840: The purine-suffix hotspot cube is realized as a finite NameCert packet over cube vertex rows, reassignment-count ledgers, layer and transfer-square classifiers, sibling dependency rows, transport, replay, provenance, and local naming.

Corollary 9462.1782 (Comparative variation is concentrated in the hotspot cube)**.** *The cube C_{WRR} has eight codons, hence one eighth of the codon carrier, but it contains four fifths of the standard-to-comparative reassignment rows:*

$$\frac{8}{64} = \frac{1}{8}, \qquad \frac{52}{65} = \frac{4}{5}.$$

Proof. The carrier size is 64 by Definition 9462.1073. The reassignment count is Definition 9462.1781. □

Theorem 9462.1783 (The hotspot cube is the highest-scoring Q_3 cube)**.** *For an axis-aligned Q_3 cube $C \subset Q_6$, let*

$$S(C) := \sum_{c \in C} m(c),$$

where $m(c)$ is the number of comparative reassignment rows at codon c. Among the 160 *axis-aligned Q_3 cubes, the unique highest score is*

$$S(C_{\mathrm{WRR}}) = 52.$$

The next highest score is 41*.*

Proof. Enumerate the $\binom{6}{3}2^3 = 160$ axis-aligned Q_3 cubes and sum the codon-level reassignment counts on each cube. The cube C_{WRR} has the vertex set displayed in Definition 9462.1780 and score 52 by Definition 9462.1781. The same enumeration gives 41 as the second-largest score. □

Definition 9462.1784 (Layers of the hotspot cube)**.** *The first-base layers of C_{WRR} are*

$$U \times R \times R = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\},$$
$$A \times R \times R = \{\mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}.$$

Under the standard classifier, the first layer is $\mathrm{Stop}_3 + \mathrm{Trp}_1$ *and the second layer is* $\mathrm{Lys}_2 + \mathrm{Arg}_2$*. Hence the whole cube has standard pattern* $3 + 2 + 2 + 1$*.*

Definition 9462.1785 (Transfer square)**.** *The transfer square is the Q_2 face*

$$Q_{\mathrm{transfer}} := \{\mathrm{UGA}, \mathrm{UGG}, \mathrm{AGA}, \mathrm{AGG}\} \subset C_{\mathrm{WRR}}.$$

It varies the first-base shape bit and the third-base family bit while keeping the second base fixed at G *and the third base in the purine layer. In the standard classifier its pattern is*

$$\mathrm{Stop}_1 + \mathrm{Trp}_1 + \mathrm{Arg}_2.$$

Theorem 9462.1786 (Vertebrate mitochondrial transfer square)**.** *In the vertebrate mitochondrial comparison table, the transfer square changes from*

	A	G
UG	Stop	Trp
AG	Arg	Arg

to

	A	G
UG	Trp	Trp
AG	Stop	Stop.

Proof. On Q_{transfer}, the standard classifier assigns UGA to Stop, UGG to Trp, and both AGA and AGG to Arg. In the vertebrate mitochondrial comparison row, UGA is assigned to Trp while AGA and AGG are assigned to Stop; UGG remains Trp. The two displayed tables are therefore the direct restriction of the two classifiers to the transfer square. □

Corollary 9462.1787 (AGA/AGG belongs to the hotspot transfer geometry)**.** *The AGA/AGG pair is the A-layer edge of the transfer square, coupled by the first-position shape direction to the UGA/UGG edge.*

Proof. The Hamming edges AGA $\leftrightarrow$ UGA and AGG $\leftrightarrow$ UGG differ only in b_1^s. Both edges lie inside Q_{transfer} by Definition 9462.1785. □

Definition 9462.1788 (Arg and Ser satellite edges)**.** *In the standard classifier,*

$$\begin{aligned}\text{Arg} &= \{\text{CGU}, \text{CGC}, \text{CGA}, \text{CGG}\} \sqcup \{\text{AGA}, \text{AGG}\},\\ \text{Ser} &= \{\text{UCU}, \text{UCC}, \text{UCA}, \text{UCG}\} \sqcup \{\text{AGU}, \text{AGC}\}.\end{aligned}$$

The two-point summands are the Arg and Ser satellite edges.

Theorem 9462.1789 (Arg satellite is active while Ser satellite is stable)**.** *Across the comparative reassignment spectrum, the Arg main box* $\{\text{CGU}, \text{CGC}, \text{CGA}, \text{CGG}\}$ *has no reassignment rows, while its satellite edge has*

$$m(\text{AGA}) = 8, \qquad m(\text{AGG}) = 8.$$

The Ser satellite edge has

$$m(\text{AGU}) = m(\text{AGC}) = 0.$$

Proof. Restrict the codon-level reassignment count m to the displayed standard Arg and Ser fibers. The four codons in the Arg main box have count zero. The Arg satellite counts are both eight, while the Ser satellite counts are both zero. □

Corollary 9462.1790 (AGR is the active purine satellite)**.** *The active satellite is not any detached two-codon sense edge. It is specifically the purine satellite*

$$\text{AGR} = \{\text{AGA}, \text{AGG}\} \subset C_{\text{WRR}},$$

whereas the pyrimidine-side Ser satellite $\text{AGY} = \{\text{AGU}, \text{AGC}\}$ *remains stable in the comparison.*

Proof. The activity and stability counts are Definition 9462.1789. The inclusion $\text{AGR} \subset C_{\text{WRR}}$ is immediate from Definition 9462.1780. □

Theorem 9462.1791 (AGA/AGG reassignment targets)**.** *The observed AGA/AGG reassignment rows have the following pair targets:*

table identifiers	AGA	AGG
2	Stop	Stop
$5, 9, 14, 21$	Ser	Ser
13	Gly	Gly
$24, 33$	Ser	Lys.

Proof. Compare the AGA and AGG assignments in each non-standard table with the standard Arg assignment. The tables with a change are exactly the rows displayed, grouped by the resulting pair of labels. □

Theorem 9462.1792 (Hotspot cube cannot-claim boundary)**.** *The hotspot-cube spectrum does not license any claim that purine suffixes cause genetic-code reassignment, that the displayed comparison family is biologically exhaustive, that the vertebrate mitochondrial row is a temporal ancestor of the other rows, or that the AGA/AGG edge has already been reduced to a biochemical mechanism.*

Proof. The hotspot cube is a finite bridge-side spectrum over curated translation classifiers. It records codon positions, labels, local cube geometry, and reassignment counts. It does not supply biochemical causality, evolutionary ordering, exhaustive biological coverage, or a mechanistic model of tRNA, release-factor, or ribosome behavior. □

Corollary 9462.1793 (Allowed hotspot-cube claim)**.** *Under the two-axis codon bridge, the main comparative deformation region is the purine-suffix cube C_{WRR}. It contains the Stop/Trp tile, the Lys/Arg purine layer, the transfer square, and the active Arg satellite edge, and it is the highest-scoring Q_3 cube under codon-level reassignment counts.*

Proof. Combine Definition 9462.1781, Definition 9462.1783, Definition 9462.1784, Definition 9462.1785, and Definition 9462.1789. □

Definition 9462.1794 (Hotspot state vector)**.** *For a genetic-code table t, the C_{WRR} state vector is*

$$\begin{aligned} S_t(C_{\mathrm{WRR}}) := (&E_t(\mathrm{UAA}), E_t(\mathrm{UAG}), E_t(\mathrm{UGA}), E_t(\mathrm{UGG}), \\ &E_t(\mathrm{AAA}), E_t(\mathrm{AAG}), E_t(\mathrm{AGA}), E_t(\mathrm{AGG})). \end{aligned}$$

This is a finite assignment vector over the hotspot cube, not a claim about biochemical transition time.

Theorem 9462.1795 (Hotspot state spectrum)**.** *Across the current curated genetic-code tables, the one-letter assignment vectors $S_t(C_{\mathrm{WRR}})$ realize 19 distinct states. The standard hotspot state*

$$(*, *, *, W, K, K, R, R)$$

appears in exactly the tables

$$1, \quad 11, \quad 12, \quad 23, \quad 26.$$

Proof. For each table, read the eight assignments in the order fixed by Definition 9462.1794 and identify equal one-letter assignment vectors. The finite enumeration gives 19 distinct vectors. The vector equal to the standard one occurs precisely in the five displayed table identifiers. □

Theorem 9462.1796 (Invariant anchors of the hotspot cube)**.** *The codons* UGG *and* AAG *are invariant across the comparison family:*

$$E_t(\mathrm{UGG}) = W, \qquad E_t(\mathrm{AAG}) = K$$

for every table t in the comparison.

Proof. The codon-level reassignment counts in Definition 9462.1781 are zero for UGG and AAG. Their standard labels are W and K, respectively. □

Corollary 9462.1797 (Active hotspot support)**.** *The active support of the hotspot cube is the six-point set*

$$C_{\mathrm{active}} = C_{\mathrm{WRR}} \setminus \{\mathrm{UGG}, \mathrm{AAG}\} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{AAA}, \mathrm{AGA}, \mathrm{AGG}\}.$$

Proof. Remove the two invariant anchors of Definition 9462.1796 from the eight vertices of Definition 9462.1780. □

Definition 9462.1798 (Reassignment-set preorder)**.** *For codons x and y, write*

$$x \preceq y$$

when every table that reassigns x also reassigns y:

$$\Delta(x) \subseteq \Delta(y).$$

This is an observed inclusion relation on finite comparison rows. It is not a causality relation.

Theorem 9462.1799 (Arg satellite reassignment is contained in UGA reassignment)**.** *The Arg satellite codons satisfy*

$$\Delta(\mathrm{AGA}) = \Delta(\mathrm{AGG}) \subseteq \Delta(\mathrm{UGA}).$$

Moreover,

$$|\Delta(\mathrm{AGA})| = |\Delta(\mathrm{AGG})| = |\Delta(\mathrm{AGA}) \cap \Delta(\mathrm{UGA})| = |\Delta(\mathrm{AGG}) \cap \Delta(\mathrm{UGA})| = 8.$$

Proof. The finite reassignment enumeration gives

$$\Delta(\mathrm{AGA}) = \{2, 5, 9, 13, 14, 21, 24, 33\},$$
$$\Delta(\mathrm{AGG}) = \{2, 5, 9, 13, 14, 21, 24, 33\}.$$

Each displayed table identifier also lies in $\Delta(\mathrm{UGA})$. The cardinality and co-occurrence equations follow. □

Theorem 9462.1800 (AAA reassignment is a deeper hotspot event)**.** *The lysine codon* AAA *satisfies*

$$\Delta(\mathrm{AAA}) \subseteq \Delta(\mathrm{UGA}) \cap \Delta(\mathrm{AGA}) \cap \Delta(\mathrm{AGG}),$$

and $|\Delta(\mathrm{AAA})| = 3$.

Proof. The finite comparison gives

$$\Delta(\mathrm{AAA}) = \{9, 14, 21\}.$$

Each of these table identifiers also reassigns UGA, AGA, and AGG. Thus the displayed inclusion holds, with cardinality three. □

Theorem 9462.1801 (Ile/Met active point is contained in UGA reassignment)**.** *The Ile/Met active point satisfies*

$$\Delta(\mathrm{AUA}) \subseteq \Delta(\mathrm{UGA}), \qquad |\Delta(\mathrm{AUA})| = 5.$$

Proof. The AUA-to-Met reassignment occurs in exactly the tables $2, 3, 5, 13, 21$, as recorded in Definition 9462.1222. Each of those tables also reassigns UGA. □

Corollary 9462.1802 (Core reassignment preorder)**.** *Inside the hotspot core, the observed reassignment inclusions include*

$$\mathrm{AUA} \preceq \mathrm{UGA}, \qquad \mathrm{AGA} \preceq \mathrm{UGA}, \qquad \mathrm{AGG} \preceq \mathrm{UGA},$$

and

$$\mathrm{AAA} \preceq \mathrm{AGA}, \qquad \mathrm{AAA} \preceq \mathrm{AGG}, \qquad \mathrm{AAA} \preceq \mathrm{UGA}.$$

Proof. This is the preorder notation of Definition 9462.1798 applied to Definition 9462.1799, Definition 9462.1800, and Definition 9462.1801. □

Definition 9462.1803 (Core hotspot support)**.** *The core hotspot support is*

$$\mathcal{H}_{\mathrm{core}} := C_{\mathrm{WRR}} \cup \{\mathrm{AUA}\}.$$

It has nine codons. Its state vector appends $E_t(\mathrm{AUA})$ *to* $S_t(C_{\mathrm{WRR}})$*.*

Theorem 9462.1804 (Core hotspot captures most reassignment rows)**.** *The support* C_{WRR} *captures* 52 *of the* 65 *reassignment rows. The support* $\mathcal{H}_{\mathrm{core}}$ *captures* 57 *of the* 65 *reassignment rows:*

$$\frac{52}{65} = \frac{4}{5}, \qquad \frac{57}{65} \approx 0.877.$$

The one-letter assignment vectors on $\mathcal{H}_{\mathrm{core}}$ *realize* 21 *distinct states across the comparison family.*

Proof. The first count is Definition 9462.1781. The codon AUA contributes five reassignment rows by Definition 9462.1801. Since AUA is not a vertex of C_{WRR}, the combined support captures $52 + 5 = 57$ rows. Enumerating the resulting nine-coordinate assignment vectors gives 21 distinct states. □

Corollary 9462.1805 (Finite state reading of the hotspot core)**.** *The comparative spectrum can be read as a finite state system on* $\mathcal{H}_{\mathrm{core}}$*: two coordinates of* C_{WRR} *are fixed anchors,* UGA *is the main reassignment hub, the Arg satellite edge is contained in UGA reassignment,* AAA *appears only inside the stronger hotspot event, and* AUA *couples the Ile/Met control tile to the same UGA hub.*

Proof. The fixed anchors are Definition 9462.1796. The inclusion relations are Definition 9462.1802. The nine-point support and capture count are Definition 9462.1804. □

Theorem 9462.1806 (Hotspot state-system cannot-claim boundary)**.** *The finite state reading does not license any claim that* UGA *causes the other reassignments, that table identifiers are temporal steps, that the observed preorder is an evolutionary tree, that all unobserved states are biologically impossible, or that the nine-point support determines translation dynamics without context.*

Proof. The construction uses finite assignment vectors and inclusion relations among observed reassignment sets. It has no temporal parameter, no mechanistic biochemical transition rule, no exhaustive biological state space, and no ribosome-context model. These would require separate external biological bridge rows. □

Corollary 9462.1807 (Allowed hotspot state-system claim)**.** *Under the two-axis codon bridge and the curated comparison family, standard-to-comparative genetic-code variation is strongly compressed by the nine-point support* $\mathcal{H}_{\text{core}}$*. The support captures* 57 *of* 65 *reassignment rows and organizes the main observed co-occurrences through the reassignment-set preorder.*

Proof. Combine Definition 9462.1804 and Definition 9462.1802. □

Concretized in §2368: WRR hotspot state vectors are realized as finite state, anchor, reassignment-preorder, table-provenance, transport, replay, and NameCert rows.

Definition 9462.1808 (Partial-aware core state)**.** *The partial-aware core state of a genetic-code table is the* $\mathcal{H}_{\text{core}}$ *state vector of Definition 9462.1803, except that ambiguous table rows are kept as separate labels. Thus a row such as* UGA = Stop *or* Trp *is recorded as* $*|W$ *rather than being collapsed to* W *or to* $*$*.*

Theorem 9462.1809 (Partial-aware core state spectrum)**.** *Across the comparison family, the partial-aware core states realize* 22 *distinct states. The standard core state*

$$(*, *, *, W, K, K, R, R, I)$$

appears in exactly the tables

$$1, \quad 11, \quad 12, \quad 23, \quad 26.$$

Proof. Read the nine core coordinates in the order

$$\text{UAA}, \text{UAG}, \text{UGA}, \text{UGG}, \text{AAA}, \text{AAG}, \text{AGA}, \text{AGG}, \text{AUA},$$

using the main one-letter row except when the difference table records an ambiguous row. Equal nine-tuples are then identified. The finite enumeration gives 22 distinct states, with the standard tuple appearing in the five displayed table identifiers. □

Definition 9462.1810 (Strict core state graph)**.** *Let* $G_1(\mathcal{H}_{\text{core}})$ *be the graph whose vertices are the distinct partial-aware core states. Two vertices are adjacent when their nine-coordinate assignment vectors differ in exactly one coordinate.*

Theorem 9462.1811 (Strict core state graph components)**.** *The graph* $G_1(\mathcal{H}_{\text{core}})$ *has* 9 *connected components. Their sizes are*

$$10, \quad 4, \quad 2, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1, \quad 1.$$

The component containing the standard state has ten vertices represented by the table groups

$$\{1, 11, 12, 23, 26\}, \quad \{32\}, \quad \{25\}, \quad \{16, 22\}, \quad \{15\},$$
$$\{6\}, \quad \{27\}, \quad \{10\}, \quad \{4\}, \quad \{3\}.$$

The next two nontrivial components are represented by

$$\{5\}, \{21\}, \{9\}, \{14\}$$

and

$$\{24\}, \{33\}.$$

Proof. Enumerate the 22 partial-aware core states of Definition 9462.1809. For each pair of states, compute the Hamming distance of the nine assignment coordinates. Add an edge exactly when this distance is one. A finite connected-component computation gives the displayed component count, component sizes, and table-group representatives. □

Corollary 9462.1812 (Single-coordinate dynamics is insufficient)**.** *The observed core states are not connected by strict single-coordinate changes. A state-system reading therefore requires module-level moves in addition to codon-coordinate Hamming edges.*

Proof. If strict single-coordinate moves connected the observed core state space, the graph $G_1(\mathcal{H}_{\mathrm{core}})$ would be connected. By Definition 9462.1811, it has nine connected components. □

Definition 9462.1813 (Observed module generators)**.** *The observed module generators are schematic reassignment moves on the core support:*

generator	*support*
g_{UGA}	$\{\mathrm{UGA}\}$
$g_{\mathrm{UAA/UAG}}$	$\{\mathrm{UAA}, \mathrm{UAG}\}$
g_{AGR}	$\{\mathrm{AGA}, \mathrm{AGG}\}$
g_{AUA}	$\{\mathrm{AUA}\}$
g_{AAA}	$\{\mathrm{AAA}\}$.

The labels record supports, not mechanistic causes. Each generator has several observed target labels, such as g_{AGR} *sending* (R, R) *to* $(*, *)$, (S, S), (G, G), *or* (S, K).

Concretized in §2306: The partial-aware strict core-state graph is realized as finite state, edge, component, module-generator, transport, replay, provenance, and NameCert rows.

Theorem 9462.1814 (Module generators explain strict graph separation)**.** *Several observed core states differ from the standard state by a module combination rather than by a single coordinate. For example:*

table	*core module difference from standard*
2	$g_{\mathrm{UGA}} + g_{\mathrm{AUA}} + g_{\mathrm{AGR}}$
5	$g_{\mathrm{UGA}} + g_{\mathrm{AUA}} + g_{\mathrm{AGR}}$
9	$g_{\mathrm{UGA}} + g_{\mathrm{AGR}} + g_{\mathrm{AAA}}$
24	$g_{\mathrm{UGA}} + g_{\mathrm{AGR}}$
33	$g_{\mathrm{UAA/UAG}} + g_{\mathrm{UGA}} + g_{\mathrm{AGR}}$.

Proof. Project each displayed table to the nine core coordinates and compare with the standard core state. Table 2 changes UGA, AUA, and the AGA/AGG pair. Table 5 has the same support pattern with a different AGR target. Table 9 changes UGA, the AGR pair, and AAA. Table 24 changes UGA and the AGR pair. Table 33 changes a stop-arm coordinate, UGA, and the AGR pair. These are module-support differences, not single-coordinate edges from the standard state. □

Corollary 9462.1815 (Core dynamics uses hub, arm, satellite, leaf, and anchor roles)**.** *The core state graph separates roles:* UGA *is the hub coordinate,* UAA/UAG *forms a stop arm,* AGA/AGG *forms the active satellite pair,* AUA *is the Ile/Met leaf coupled to the hub,* AAA *is a deeper hotspot point, and* UGG *and* AAG *are fixed anchors.*

Proof. The hub and inclusion relations are Definition 9462.1802. The stop arm, satellite pair, leaf, and deeper point are the supports in Definition 9462.1813. The anchors are Definition 9462.1796. □

Definition 9462.1816 (Active core support partition)**.** *The active core support is obtained from* $\mathcal{H}_{\text{core}}$ *by removing the two invariant anchors* UGG *and* AAG*:*

$$\mathcal{A}_{\text{core}} = \{\text{UAA}, \text{UAG}, \text{UGA}, \text{AAA}, \text{AGA}, \text{AGG}, \text{AUA}\}.$$

A support partition of $\mathcal{A}_{\text{core}}$ *is a set partition of these seven coordinates. Given such a partition, two observed partial-aware core states are module-adjacent when their active coordinates differ inside exactly one block of the partition.*

Theorem 9462.1817 (Unique finest connected support partition)**.** *Among all* 877 *set partitions of* $\mathcal{A}_{\text{core}}$*, exactly* 52 *make the observed partial-aware core state graph connected. Among the connected partitions, the unique finest one is*

$$\{\text{UAA}, \text{UAG}\}, \quad \{\text{UGA}\}, \quad \{\text{AAA}\},$$
$$\{\text{AGA}, \text{AGG}\}, \quad \{\text{AUA}\}.$$

Proof. There are $B_7 = 877$ set partitions of the seven active coordinates. For each partition, compress each observed state to its blockwise coordinates and add an edge when two states differ in exactly one block. The finite enumeration gives 52 connected partitions. The maximum number of blocks among connected partitions is 5, and the only connected partition with 5 blocks is the displayed one. □

Corollary 9462.1818 (Stop arm and AGR satellite are forced blocks)**.** *The two non-singleton blocks in the unique finest connected partition are precisely the stop arm* $\{\text{UAA}, \text{UAG}\}$ *and the AGR satellite pair* $\{\text{AGA}, \text{AGG}\}$*.*

Proof. This is the non-singleton part of Definition 9462.1817. □

Theorem 9462.1819 (Finest module graph edge spectrum)**.** *For the unique finest connected support partition, the module graph on the* 22 *observed partial-aware core states is connected and has* 42 *edges. The edge labels are distributed as follows:*

module	*edge count*
$\{\text{UAA}, \text{UAG}\}$	25
$\{\text{UGA}\}$	7
$\{\text{AAA}\}$	1
$\{\text{AGA}, \text{AGG}\}$	7
$\{\text{AUA}\}$	2.

Proof. Use the partition of Definition 9462.1817. For every pair of observed states, count the blocks on which their active coordinates differ. Add an edge exactly when this count is one, and label the edge by that block. The resulting graph is connected and the finite edge count gives the displayed spectrum. □

Corollary 9462.1820 (Two centrality readings)**.** *The core comparison separates two centrality notions. The codon* UGA *is the principal reassignment hub by occurrence count, while the stop arm* $\{\text{UAA}, \text{UAG}\}$ *is the largest horizontal connector in the finest module graph.*

Proof. The codon-level hub property is Definition 9462.1802. The module edge maximum is the 25-edge stop-arm row of Definition 9462.1819. □

Theorem 9462.1821 (Standard-state module graph distances)**.** *In the finest module graph, the standard core state has degree* 9*. The shortest-path distance distribution from the standard state is*

distance	*state count*
0	1
1	9
2	4
3	5
4	1
5	1
6	1.

The diameter of the finest module graph is 9*.*

Proof. Run breadth-first search in the connected module graph of Definition 9462.1819, using the standard partial-aware state as the source. The degree and distance distribution are the displayed values. A breadth-first search from each vertex gives maximum shortest-path length 9. □

Theorem 9462.1822 (Unique finest generator system)**.** *Under the single condition that the observed partial-aware core states be connected by one-block module adjacency, the unique finest generator support system is*

$$\{\mathrm{UAA}, \mathrm{UAG}\}, \quad \{\mathrm{UGA}\}, \quad \{\mathrm{AAA}\}, \quad \{\mathrm{AGA}, \mathrm{AGG}\}, \quad \{\mathrm{AUA}\}.$$

Proof. By Definition 9462.1817, this is the unique connected support partition with the maximum number of blocks. Assigning one schematic generator to each block gives the stated support system, and uniqueness follows from uniqueness of the partition. □

Definition 9462.1823 (Module activation pattern)**.** *For the unique finest generator support system, write*

$$S = \{\mathrm{UAA}, \mathrm{UAG}\}, \qquad U = \{\mathrm{UGA}\}, \qquad N = \{\mathrm{AAA}\},$$
$$R = \{\mathrm{AGA}, \mathrm{AGG}\}, \qquad I = \{\mathrm{AUA}\}.$$

For a comparison table, a module is active when at least one coordinate inside that module differs from the standard assignment. The module activation pattern is the subset of $\{S, U, N, R, I\}$ *consisting of active modules.*

Theorem 9462.1824 (Observed module activation patterns)**.** *The comparison family realizes*

exactly 11 *module activation patterns:*

pattern	*table identifiers*
$\varnothing$	$1, 11, 12, 23, 26$
U	$4, 10, 25$
S	$6, 15, 16, 22, 29, 30, 32$
$U + I$	3
$U + R$	24
$S + U$	$27, 28, 31$
$U + R + I$	$2, 5, 13$
$U + N + R$	9
$S + U + R$	33
$U + N + R + I$	21
$S + U + N + R$	$14.$

Proof. For each table, compare the seven active coordinates with the standard assignment and mark a module active when any coordinate in that block differs. Identifying equal active-module subsets gives exactly the eleven displayed rows. □

Theorem 9462.1825 (Five-variable activation grammar)**.** *The observed module activation patterns are exactly the Boolean solutions on variables S, U, N, R, I satisfying*

$$R \Rightarrow U, \qquad I \Rightarrow U, \qquad N \Rightarrow R, \qquad S \wedge I = 0.$$

Proof. Necessity is a direct finite check on Definition 9462.1824: every row containing R contains U, every row containing I contains U, every row containing N contains R, and no row contains both S and I.

For sufficiency, enumerate the Boolean solutions. If $U = 0$, then $R = I = N = 0$, while S is free, giving $\varnothing$ and S. If $U = 1$ and $I = 0$, then S is free and (R, N) can be $(0, 0)$, $(1, 0)$, or $(1, 1)$, giving

$$U, \quad S + U, \quad U + R, \quad S + U + R, \quad U + N + R, \quad S + U + N + R.$$

If $U = 1$ and $I = 1$, then $S = 0$, and (R, N) again gives

$$U + I, \quad U + R + I, \quad U + N + R + I.$$

Thus the grammar has $2 + 6 + 3 = 11$ solutions, and the solution list is exactly the list in Definition 9462.1824. □

Corollary 9462.1826 (Hub, dependent branches, and transverse arm)**.** *The activation grammar separates four roles: U is a necessary hub for R, I, and N; N is dependent on the AGR satellite branch through $N \Rightarrow R$; the AUA leaf is dependent on U; and the stop arm S is a transverse axis that may occur with or without U but is disjoint from the AUA branch in the observed comparison family.*

Proof. Each role is one of the four clauses in Definition 9462.1825, together with the fact that both S and $S + U$ occur in Definition 9462.1824. □

Theorem 9462.1827 (Module value spectrum)**.** *Across the comparison family, the five modules have the following observed value spectra:*

module	*value count*	*values*
S	10	$(*,*),(*,Q),(*,L),(*,W),(Q,Q),(Q\|*,Q\|*),(Y,*),(Y,Y),(E,E),(E\|*,E\|*)$
U	5	$*,W,C,G,*\|W$
N	2	K,N
R	5	$(R,R),(*,*),(S,S),(G,G),(S,K)$
I	2	$I,M.$

Proof. Read each table on the active coordinates and group the coordinates by the unique finest support partition of Definition 9462.1817. The finite enumeration of distinct block values gives the displayed spectra. □

Theorem 9462.1828 (Four-branch module value grammar)**.** *The* 22 *observed module-value states are exactly the states generated by the following four branches.*

branch	*constraint*	*state cour*
A	$U=*,\ N=K,\ R=(R,R),\ I=I,\ S\in\{(*,*),(Q,Q),(*,Q),(*,L),(Y,Y),(E,E),(*,W)\}$	7
B	$S=(*,*),\ N=K,\ R=(R,R),\ I=I,\ U\in\{C,G\}$	2
C	$U=*\|W,\ N=K,\ R=(R,R),\ I=I,\ S\in\{(Q,Q),(Q\|*,Q\|*)\}$	2
D	$U=W$ *and one of the observed deep states*	11.

The deep branch consists of the eleven tuples

$$(*,*)/W/K/(R,R)/I,\quad (*,*)/W/K/(R,R)/M,\quad (E|*,E|*)/W/K/(R,R)/I,$$
$$(*,*)/W/K/(*,*)/M,\quad (*,*)/W/K/(S,S)/M,\quad (*,*)/W/K/(G,G)/M,$$
$$(*,*)/W/N/(S,S)/I,\quad (Y,*)/W/N/(S,S)/I,\quad (*,*)/W/N/(S,S)/M,$$
$$(*,*)/W/K/(S,K)/I,\quad (Y,*)/W/K/(S,K)/I.$$

Proof. The four displayed branches contain $7+2+2+11=22$ states. Direct finite enumeration of the module-value states obtained from the comparison family gives 22 states, and the generated set equals the observed set. □

Corollary 9462.1829 (Trp value as deep entrance)**.** *Among the nonstandard values of U, only $U=W$ opens the deep coupled branch. The values $U=C$ and $U=G$ occur only in the pure U branch, and $U=*|W$ occurs only in the partial-boundary branch.*

Proof. This is the branch separation in Definition 9462.1828. □

Theorem 9462.1830 (Deep value constraints)**.** *The observed module-value states satisfy the following value-level constraints:*

$$R\neq(R,R)\Rightarrow U=W,$$
$$I=M\Rightarrow U=W\ \text{ and }\ S=(*,*),$$
$$N=N\Rightarrow U=W\ \text{ and }\ R=(S,S),$$
$$R\in\{(*,*),(G,G)\}\Rightarrow I=M,\ S=(*,*),\ N=K,$$
$$R=(S,K)\Rightarrow I=I,\ N=K,\ U=W.$$

Proof. Check the four branches of Definition 9462.1828. In the first three branches, $R = (R, R)$ except for no case, $I = I$, and $N = K$. In the deep branch, the displayed eleven tuples verify each implication coordinatewise. □

Corollary 9462.1831 (AGR as value splitter)**.** *The AGR module is a value splitter inside the deep branch. The value $(*, *)$ is tied to the $I = M$ leaf, (S, S) is the only background in which $N = N$ occurs, (G, G) is tied to the $I = M$ leaf, and (S, K) is a split satellite value with $I = I$ and $N = K$.*

Proof. This is a restatement of the last three implications in Definition 9462.1830, restricted to the nonstandard values of R. □

Theorem 9462.1832 (Core state graph cannot-claim boundary)**.** *The strict core state graph and module generators do not license any claim that the observed tables form a temporal trajectory, that the generators are biochemical mutation mechanisms, that every generator can occur independently in nature, or that the finite comparison family exhausts all possible code states.*

Proof. The graph is built from observed assignment vectors and Hamming distances between them. The module generators summarize observed supports of difference from the standard state. Neither construction supplies time, mechanism, independence, or biological exhaustiveness. □

Corollary 9462.1833 (Allowed module-state claim)**.** *The partial-aware core comparison is best read as a finite module-state system: strict single-coordinate adjacency is disconnected, while the observed differences organize into hub, arm, satellite, leaf, deeper point, and anchor roles.*

Proof. The disconnection is Definition 9462.1812. The role decomposition is Definition 9462.1815. □

Definition 9462.1834 (Module-value monotone path)**.** *Let a module-value state be written as*

$$X = (S, U, N, R, I).$$

For a target state $T = (S_T, U_T, N_T, R_T, I_T)$, a path from the standard state to T is target-monotone when each step changes exactly one module from its standard value to its target value, and no step changes a module away from its target value.

Theorem 9462.1835 (Observed monotone-path exceptions)**.** *Among the 22 observed module-value states, exactly 19 admit a target-monotone path from the standard state using only observed states. The three exceptions are represented by table identifiers*

$$9, \qquad 14, \qquad 28.$$

Their module-value states are

$$(*, *)/W/N/(S, S)/I,$$
$$(Y, *)/W/N/(S, S)/I,$$
$$(Q|*, Q|*)/*|W/K/(R, R)/I.$$

Proof. For each observed target, enumerate all orders in which its nonstandard modules can be changed from their standard values to their target values. A target is accepted exactly when at least one such prefix path lies entirely in the observed state set. The finite enumeration rejects exactly the three displayed states. □

Definition 9462.1836 (Candidate latent states)**.** *The deep-branch candidate latent states are*

$$L_{SS} = (*,*)/W/K/(S,S)/I$$

and

$$L_N = (*,*)/W/N/(R,R)/I.$$

The partial-boundary candidate latent states are

$$L_{*|W} = (*,*)/*|W/K/(R,R)/I$$

and

$$L_{Q|*} = (Q|*, Q|*)/*/K/(R,R)/I.$$

They are not observed table states in the comparison family; they are prefix states implied by the target-monotone support geometry.

Theorem 9462.1837 (Minimal latent completions)**.** *The minimum latent completion size is* 2*. There are exactly four minimum completions:*

$$\{L_{SS}, L_{*|W}\}, \qquad \{L_{SS}, L_{Q|*}\},$$
$$\{L_N, L_{*|W}\}, \qquad \{L_N, L_{Q|*}\}.$$

Proof. The three exceptions in Definition 9462.1835 split into two independent missing squares. Tables 9 and 14 require one deep-branch prefix, either L_{SS} or L_N. Table 28 requires one partial-boundary prefix, either $L_{*|W}$ or $L_{Q|*}$. A single latent state cannot be a prefix for both missing squares, so size 1 is insufficient. The Cartesian product of the two binary choices gives the four displayed size-2 completions, and each removes all three target-monotone exceptions. □

Concretized in §3521: Latent-completion packets realize the finite monotone-path completion witness for the module-state exception geometry.

Definition 9462.1838 (Canonical latent completion)**.** *The canonical latent completion is*

$$\mathcal{L}_{\text{can}} = \{L_{SS}, L_{*|W}\}.$$

It is the minimum completion selected by the value-direction readings $U = W \to R = (S,S) \to N = N$ *and* $U = *|W \to S = (Q|*, Q|*)$.

Concretized in §7831: Canonical latent-completion packets realize the selected two-state target-monotone completion route.

Theorem 9462.1839 (Canonical target-monotone completion)**.** *After adjoining the canonical latent completion to the observed module-value state set, every observed state admits a target-monotone path from the standard state.*

Proof. For the deep branch, L_{SS} supplies the path

$$(*,*)/*/K/(R,R)/I \to (*,*)/W/K/(R,R)/I \to L_{SS}$$
$$\to (*,*)/W/N/(S,S)/I \to (Y,*)/W/N/(S,S)/I.$$

For the partial-boundary branch, $L_{*|W}$ supplies the path

$$(*,*)/*/K/(R,R)/I \to L_{*|W}$$
$$\to (Q|*, Q|*)/*|W/K/(R,R)/I.$$

The remaining 19 observed states already have target-monotone paths by Definition 9462.1835. □

Theorem 9462.1840 (Latent completion graph compression)**.** *The observed module-value graph has* 22 *nodes,* 42 *edges, diameter* 9*, and maximum distance* 6 *from the standard state to an observed state. After adjoining* L_{SS}*, the graph has* 23 *nodes,* 46 *edges, diameter* 7*, and maximum standard-to-observed distance* 4*. After also adjoining* $L_{*|W}$*, the graph has* 24 *nodes,* 52 *edges, diameter* 5*, and maximum standard-to-observed distance* 4*.*

Proof. In each graph, connect two states exactly when they differ in one module value. Compute all-pairs shortest-path distances and the distances from the standard state to the observed vertices. The finite graph enumeration gives the displayed node counts, edge counts, diameters, and standard-source radii. □

Corollary 9462.1841 (Observed states as sampled endpoints)**.** *The comparison family should not be read as a complete smooth state space. It is better read as a finite sample of stable or recorded endpoints, while the module-value grammar also identifies prefix states that are not present as named observed tables.*

Proof. The missing target-monotone prefixes are precisely the latent states of Definition 9462.1836, and their addition removes all target-monotone exceptions by Definition 9462.1839. □

Corollary 9462.1842 (AUA detour is not a forced premise)**.** *The observed graph alone can route some paths toward* $N = N$ *through states with* $I = M$*, but the latent-completed graph gives a direct target-monotone route*

$$U = W, \qquad R = (S, S), \qquad N = N$$

without passing through $I = M$*. Thus the stable implication is*

$$N = N \Rightarrow R = (S, S) \Rightarrow U = W,$$

not a requirement that $I = M$ *precede* $N = N$*.*

Proof. The direct route is the L_{SS} route in Definition 9462.1839. It reaches the table-9 state before any change in the I module. □

Definition 9462.1843 (Canonical rewrite edge)**.** *In the canonical completed state set, a directed rewrite edge* $X \to Y$ *exists when* X *and* Y *differ in exactly one module,* X *has the standard value in that module, and* Y *has a nonstandard value there.*

Theorem 9462.1844 (Canonical rewrite reachability)**.** *The canonical completed state set has* 24 *states. Its directed rewrite graph has* 26 *edges, and every state in the completed set is reachable from the standard state. In particular, all* 22 *observed states are reachable.*

Proof. Enumerate the 22 observed states together with L_{SS} and $L_{*|W}$. Add directed edges according to Definition 9462.1843. The finite enumeration gives 26 directed edges, and graph search from the standard state reaches all 24 states. □

Theorem 9462.1845 (Latent-completed grammar boundary)**.** *The latent-completed graph does not claim that the two latent states are currently observed biological codes, that every target-monotone edge is a biochemical transition, or that the comparison family contains every stable biological endpoint.*

Proof. The construction uses finite module-value states and graph completion under a target-monotone prefix criterion. It supplies a structural completion of the observed state set, not a temporal or mechanistic biological assertion. □

Concretized in §3470: The six-boundary tile coverage packet realizes the trigger-support and coverage-rank rows as finite NameCert data.

9462.28.22 Codon-window spectral inverse dynamics

The codon-window bridge treats RNA and DNA codons as local six-bit states, while keeping the biological genetic-code table outside BEDC kernel primitives. The inverse thesis is local rather than global:

$$\mathsf{Codon} = Q_6 = \{0,1\}^6,$$
$$\mathrm{Stop}_{\mathsf{std}} = \mathrm{U} + \mathrm{RR} + X_2,$$
$$\mathrm{Stop}_{\mathsf{std}} \neq X_6.$$

The inverse question asks how much codon-window structure can be recovered from finite spectra of a classifier. A single global fiber histogram gives the degeneracy profile but not the table. Adding local Q_2-face spectra, Hamming adjacency spectra, and comparative reassignment spectra sharply constrains the local mechanisms detected in subsection 9462.28.21: the dual 11-control tiles, the two 00-singleton Phe/Leu tiles, and the finite local deformations visible across compared genetic-code tables.

Definition 9462.1846 (Window-six spectral inverse bridge status)**.** *The bridge* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *is a* $\mathsf{VisionSeed/BiologicalBridge}$ *target and a finite-enumeration paper-certificate candidate. It is not a BEDC kernel primitive and is not a checked certificate. A future certificate would separate source, pattern, classifier, stability, and ledger fields for the finite codon cube, the two-axis readback, the coding table, the inverse spectra, local tile classifications, adjacency rows, reassignment rows, and biological table provenance.*

Definition 9462.1847 (Two-axis codon atlas)**.** *Let*

$$\Sigma := \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}.$$

The external biological readback is

$$\mathrm{U/T} = (\mathrm{Y}, \mathrm{W}) = 00 \quad \mathrm{C} = (\mathrm{Y}, \mathrm{S}) = 01,$$
$$\mathrm{A} = (\mathrm{R}, \mathrm{W}) = 10 \quad \mathrm{G} = (\mathrm{R}, \mathrm{S}) = 11.$$

Here Y/R *is the pyrimidine/purine axis and* W/S *is the AU-family/GC-family axis. This is a coordinate choice for a biological bridge, not a BEDC primitive.*

Theorem 9462.1848 (Codon cube)**.** *The coordinate map*

$$\beta : \Sigma^3 \to Q_6$$

obtained by concatenating the three two-bit base coordinates is a finite bijection.

Proof. Each base has a unique two-bit coordinate. Hence a three-base codon has a unique six-bit display. Conversely, every six-bit word splits uniquely into three two-bit blocks, each of which decodes to exactly one base. □

Corollary 9462.1849 (Carrier and classifier are distinct)**.** *The equality* $\mathsf{Codon} = Q_6$ *describes the raw local microstate carrier. A genetic-code table is a classifier*

$$E_{\mathsf{std}} : Q_6 \to \mathsf{AminoAcidClass} \oplus \mathsf{Stop},$$

not the identity map on Q_6*.*

Definition 9462.1850 (Local stop face)**.** *The stop face is the local Q_2 face*

$$\mathsf{StopTrpFace} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\},$$

where the first base is U *and the last two bases are purines. Its family-suffix coordinates are*

$$\begin{aligned} \mathrm{AA} &\longmapsto 00 \quad \mathrm{AG} \longmapsto 01, \\ \mathrm{GA} &\longmapsto 10 \quad \mathrm{GG} \longmapsto 11. \end{aligned}$$

Let

$$X_2 := \{00, 01, 10\}, \qquad Q_2 = X_2 \sqcup \{11\}.$$

Theorem 9462.1851 (Three-stop reconstruction)**.** *Under the family-suffix no-*11 *condition inside the stop face,*

$$\mathsf{ZStopCandidate} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

Proof. The four purine-purine suffixes are AA, AG, GA, and GG, with family coordinates 00, 01, 10, and 11. The local condition excludes exactly the 11 corner, namely GG, and retains the other three suffixes. Prefixing by U gives UAA, UAG, and UGA. □

Corollary 9462.1852 (UGG continuation corner)**.** *Inside the standard-code biological bridge,*

$$\mathrm{UGG} = \mathrm{U} + \mathrm{RR} + 11$$

is the unique non-stop corner adjacent to the stop L-shape. It is a reserved continuation corner, not an absent or biologically illegal state.

Theorem 9462.1853 (Local stop theorem is not global X_6)**.** *The standard stop set is not a subset of the global no-adjacent-*11 *language*

$$X_6 := \{w \in \{0,1\}^6 : w \textit{ has no adjacent substring } 11\}.$$

Proof. The flattened displays are

$$\begin{aligned} \mathrm{UAA} &= 001010, \\ \mathrm{UAG} &= 001011, \\ \mathrm{UGA} &= 001110. \end{aligned}$$

The first word has no adjacent 11, while the latter two do. Therefore the stop structure is local X_2 inside the $\mathrm{U} + \mathrm{RR}$ face, not global X_6. □

Theorem 9462.1854 (Finite local face domain)**.** *The codon cube Q_6 has exactly*

$$\binom{6}{2} 2^4 = 240$$

axis-aligned Q_2 faces.

Proof. Choose the two free bit coordinates in $\binom{6}{2}$ ways. Assign fixed values to the remaining four coordinates in 2^4 ways. The resulting faces are exactly the axis-aligned Q_2 faces of Q_6. □

Theorem 9462.1855 (Dual control tiles)**.** *The standard-code bridge contains two $X_2/\{11\}$ control tiles:*

$$\begin{aligned} \mathrm{AU} + X_2 &= \{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}\} = \mathrm{Ile}, \\ \mathrm{AU} + 11 &= \{\mathrm{AUG}\} = \mathrm{Met}, \\ \mathrm{U} + \mathrm{RR} + X_2 &= \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\} = \mathrm{Stop}, \\ \mathrm{U} + \mathrm{RR} + 11 &= \{\mathrm{UGG}\} = \mathrm{Trp}. \end{aligned}$$

In initiation context, the AUG *row also carries the canonical start bridge row.*

Proof. In the AU face, the varying third-base coordinates are U = 00, C = 01, A = 10, and G = 11. In the U + RR face, the varying purine-purine suffix coordinates are AA = 00, AG = 01, GA = 10, and GG = 11. The displayed labels are supplied by the standard genetic-code bridge. □

Corollary 9462.1856 (Local 11 is reserved)**.** *Across the two control tiles,*

$$\mathrm{AU} + 11 = \mathrm{AUG} = \mathrm{Met/Start}, \qquad \mathrm{U} + \mathrm{RR} + 11 = \mathrm{UGG} = \mathrm{Trp}.$$

Thus local 11 *is separated as a singleton or control-sensitive corner inside the biological bridge. It is not a biological deletion rule.*

Definition 9462.1857 (Coding classifier and global histogram)**.** *Let $D := Q_6$ and let C be a finite set of coding classes. A coding table is a surjective classifier*

$$E : D \to C.$$

For $c \in C$, define

$$d_E(c) := |E^{-1}(c)|, \qquad H_E(k) := |\{c \in C : d_E(c) = k\}|.$$

Theorem 9462.1858 (Global histogram identities)**.** *For any finite surjection $E : D \to C$,*

$$|C| = \sum_k H_E(k), \qquad |D| = \sum_k k\, H_E(k).$$

Proof. Each coding class contributes one fiber, so summing fibers over all sizes gives $|C|$. A fiber of size k contributes k domain states, so the weighted sum gives $|D|$. □

Corollary 9462.1859 (Standard-code degeneracy spectrum)**.** *For the standard genetic-code bridge,*

k	1	2	3	4	6
$H_{\mathsf{std}}(k)$	2	9	2	5	3.

Thus

$$2 + 9 + 2 + 5 + 3 = 21$$

and

$$1 \cdot 2 + 2 \cdot 9 + 3 \cdot 2 + 4 \cdot 5 + 6 \cdot 3 = 64.$$

The singleton classes are Met *and* Trp*, while the tripleton classes are* Ile *and* Stop*.*

Proof. This is finite enumeration of the standard genetic-code bridge table, with the displayed equalities supplied by Definition 9462.1858. □

Theorem 9462.1860 (Global histogram is incomplete)**.** *A single global histogram H_E does not uniquely determine the coding table E.*

Proof. A histogram fixes only the multiset of fiber sizes. It does not specify which points of D belong to which fibers. If the fiber sizes are $s_1, \ldots, s_m$, then any partition of the 64 codons into blocks of those sizes defines a classifier with the same histogram. For unlabeled classes, the number of such partitions has the form

$$\frac{64!}{\prod_i s_i! \prod_k H_E(k)!}.$$

Hence the histogram constrains the table but is not a complete invariant. □

Definition 9462.1861 (Enhanced inverse spectrum)**.** *Let $F \subseteq Q_6$ range over all axis-aligned Q_2 faces, and let $H(E|_F)$ denote the label multiplicity pattern on F, such as 4, $3+1$, $2+2$, $2+1+1$, or $1+1+1+1$. Define*

$$H_2(E) := \{H(E|_F) : F \subseteq Q_6,\ F \simeq Q_2\}.$$

When a face has pattern $3+1$, it is a local $3+1$ tile. It is an 11-control tile when the three-point class occupies X_2 and the singleton class occupies the local 11 corner. It is a 00-singleton tile when the singleton class occupies the local 00 corner and the three-point class occupies $Q_2 \setminus \{00\}$.

Put the Hamming graph on Q_6 and define

$$A_E(i,j) := |\{(x,y) : x \sim y,\ E(x) = i,\ E(y) = j\}|.$$

For a baseline table E and a compared table E' on the same codon carrier, define the reassignment row

$$M(E,E') := \{x \in Q_6 : E'(x) \neq E(x)\}.$$

When E is fixed, write M_E for the selected finite family of such comparison rows, with each row recorded together with its local tile coordinate when applicable. The enhanced inverse spectrum is

$$\mathsf{InvSpec}(E) := (H_E, H_2(E), A_E, M_E).$$

Theorem 9462.1862 (Singleton and tripleton spectra direct local search)**.** *If a codon classifier has singleton classes and tripleton classes, then the minimal local geometric search target is an axis-aligned Q_2 face with a $3+1$ split.*

Proof. A singleton and a tripleton have total support $1+3=4$. The smallest axis-aligned cube with four points is Q_2, so the natural finite local search object is a Q_2 face partitioned as a $3+1$ split. This is a search principle rather than a uniqueness theorem; uniqueness requires the local face spectrum. □

Corollary 9462.1863 (Standard spectrum locates the control tiles)**.** *Under the standard-code bridge, the singleton/tripleton spectrum points to the two core control tiles*

$$\text{Ile/Met} = \text{AU} + X_2/11, \qquad \text{Stop/Trp} = \text{U} + \text{RR} + X_2/11.$$

Proof. The singleton classes are Met and Trp, and the tripleton classes are Ile and Stop by Definition 9462.1859. The local Q_2 enumeration is Definition 9462.1162. □

Theorem 9462.1864 (Standard local $3+1$ face inventory)**.** *Under the two-axis readback and the standard genetic-code bridge, the axis-aligned Q_2 faces with $3+1$ translation-class splits are exactly*

face	*three-point class*	*singleton class*	*singleton corner*
$\{\mathrm{AUU}, \mathrm{AUC}, \mathrm{AUA}, \mathrm{AUG}\}$	Ile	Met	11
$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\}$	Stop	Trp	11
$\{\mathrm{UUU}, \mathrm{UUA}, \mathrm{CUU}, \mathrm{CUA}\}$	Leu	Phe	00
$\{\mathrm{UUC}, \mathrm{UUG}, \mathrm{CUC}, \mathrm{CUG}\}$	Leu	Phe	00.

Proof. This is the finite enumeration of the 240 axis-aligned Q_2 faces under the standard translation table, recorded in Definition 9462.1160. □

Corollary 9462.1865 (Phe/Leu sense-expansion tiles)**.** *The two Phe/Leu $3+1$ faces are* 00*-singleton tiles rather than* 11*-control tiles. They supply a sense-expansion mechanism distinct from the* Ile/Met *and* Stop/Trp 11*-control mechanism.*

Proof. In the two Phe/Leu rows of Definition 9462.1864, the singleton corner is 00 and the three-point class is $Q_2 \setminus \{00\}$. This is exactly the definition of a 00-singleton tile. □

Theorem 9462.1866 (Boundary and continuation reconstruction)**.** *In the Stop/Trp tile, the X_2 side reconstructs halt/release behavior and the* 11 *corner reconstructs continuation:*

$$X_2 = \{00, 01, 10\} \Longrightarrow \mathsf{Stop/halt\text{-}release},$$
$$11 \Longrightarrow \mathrm{Trp}/\mathsf{continuation}.$$

Proof. The standard-code bridge assigns UAA, UAG, and UGA to Stop, while UGG is assigned to Trp. Stop rows trigger termination, whereas the tryptophan row is a sense assignment. Thus the local $3+1$ face supports the displayed boundary/continuation reading. □

Definition 9462.1867 (Leaf and arm reassignment)**.** *In an $X_2 \sqcup \{11\}$ tile, a leaf reassignment occurs when one point of the X_2 tripleton class joins the class containing the* 11 *corner. In the stop L-shape $X_2 = \{00, 01, 10\}$, an arm reassignment occurs when an edge such as $\{00, 01\}$ or $\{00, 10\}$ is reassigned as a unit from* Stop *to a sense class.*

Proposition 9462.1868 (Dual 10-to-11 reassignment)**.** *In genetic-code bridges where* AUA *is assigned to* Met *and* UGA *is assigned to* Trp*, both changes have the same local form:*

$$10 \textit{ joins the class containing } 11.$$

Proof. In the Ile/Met tile, $\mathrm{AUA} = \mathrm{AU} + 10$ and $\mathrm{AUG} = \mathrm{AU} + 11$. In the Stop/Trp tile, $\mathrm{UGA} = \mathrm{U} + \mathrm{RR} + 10$ and $\mathrm{UGG} = \mathrm{U} + \mathrm{RR} + 11$. Assigning AUA with AUG or assigning UGA with UGG therefore has the same coordinate action. □

Proposition 9462.1869 (Stop-arm reassignment)**.** *In genetic-code bridges where* UAA *and* UAG *are assigned together to a sense class, the pair is the $\{00, 01\}$ arm of the stop L-shape.*

Proof. The stop-face coordinates are $\mathrm{UAA} = 00$, $\mathrm{UAG} = 01$, and $\mathrm{UGA} = 10$. Hence $\{\mathrm{UAA}, \mathrm{UAG}\} = \{00, 01\}$ is one edge of the L-shape $01 - 00 - 10$. □

Definition 9462.1870 (Local translation state)**.** *A local translation state has the schematic form*

$$S_t := (f_t, w_t, r_t, \kappa_t),$$

where f_t is reading frame, $w_t \in Q_6$ is the current codon window, r_t records ribosome or release-factor state, and κ_t records local sequence and cellular context. The schematic transition rule is

$$E(w_t) \in \mathsf{AminoAcidClass} \quad \Longrightarrow \quad S_t \to S_{t+1},$$
$$E(w_t) = \text{Stop} \quad \Longrightarrow \quad S_t \to S_{\mathsf{halt}}.$$

A stochastic refinement records rows $P(\delta \mid w_t, \kappa_t)$.

Theorem 9462.1871 (Stop tile is a local halt boundary)**.** *Under the standard-code bridge,*

$$\mathrm{U} + \mathrm{RR} + X_2 = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$$

is a local halt boundary in translation dynamics.

Proof. The equality is Definition 9462.1851. By Definition 9462.1870, a codon classified as Stop transitions to S_{halt}. □

Theorem 9462.1872 (Enhanced inverse spectrum)**.** *The enhanced inverse spectrum*

$$\mathsf{InvSpec}(E) = (H_E, H_2(E), A_E, M_E)$$

recovers or strongly constrains global compression, singleton/tripleton classes, local $3+1$ Q_2 tiles, reserved control corners, boundary/continuation behavior, and local leaf or arm reassignment patterns in compared genetic-code tables.

Proof. The component H_E gives class count and support size by Definition 9462.1858, but is incomplete by Definition 9462.1860. The local spectrum $H_2(E)$ attaches multiplicity patterns to concrete Q_2 faces. The adjacency spectrum A_E records one-bit class boundaries, including L-shape centers and leaves. The reassignment spectrum M_E records finite comparison rows across biological bridge tables or controlled perturbation rows. Together these spectra reduce arbitrary partitions of Q_6 to a small family of structured local mechanisms. □

Definition 9462.1873 (Finite and empirical work surface)**.** *The finite formalization target is to define the two-axis readback, Q_6, Q_2, X_2, the standard-code classifier, the global histogram, the* 240 *local Q_2 faces, the exact list of $3+1$ faces, and the spectrum $\mathsf{InvSpec}(E) = (H_E, H_2(E), A_E, M_E)$. The biological research surface then compares genetic-code tables, stop assignments, codon usage, readthrough, ribosome-profiling pause rows, release-factor context, and synthetic reporter rows for terminal windows*

$$\mathrm{UAA}, \quad \mathrm{UAG}, \quad \mathrm{UGA}, \quad \mathrm{UGG}.$$

The empirical layer tests termination efficiency, readthrough rate, protein output, mRNA abundance, ribosome pausing, and decay rate. These measurements guide bridge choice but do not replace BEDC proof or certificate obligations.

Theorem 9462.1874 (Spectral inverse cannot-claim boundary)**.** *The spectral inverse bridge does not license any claim that Zeckendorf alone proves the biological genetic code, that the whole genetic code is the window-six fold or global X_6, that standard stop codons are universal across every genetic code, that a local DNA or RNA* 11 *corner is the BEDC channel terminator, that* 11 *is biologically*

illegal or deleted, that UGG *is tryptophan by BEDC logic alone, that a single global histogram uniquely determines the genetic code, that alternative genetic-code tables contradict the model, that ribosome dynamics are determined by codon identity alone, or that* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *is already a checked certificate.*

Proof. The global X_6 refusal follows from Definition 9462.1853. The histogram refusal follows from Definition 9462.1860. The local 11 refusal follows from Definition 9462.1856. The biological-label and dynamics refusals follow because genetic-code tables and translation kinetics enter only through external bridge data. The checked-status refusal follows from Definition 9462.1846. □

Corollary 9462.1875 (Allowed spectral inverse claim)**.** *Under the two-axis RNA/DNA bridge, codons form* Q_6*. The standard stop class is a local* $\mathrm{U} + \mathrm{RR} + X_2$ *halt boundary, not a global* X_6 *law. The enhanced inverse spectrum* $(H_E, H_2(E), A_E, M_E)$ *reconstructs or strongly constrains local coding mechanisms such as the* Ile/Met *and* Stop/Trp *control tiles, the Phe/Leu* 00*-singleton tiles, and their local reassignment patterns in compared genetic-code tables.*

Concretized in §2161: Codon-window spectral inverse packets realize the finite cube, local face spectra, stop-face reconstruction, adjacency rows, and provenance as NameCert data.

Concretized in §5177: The channel genetic-code bridge realizes the codon-window inverse reconstruction route as finite channel, codon-window, classifier, transport, provenance, and NameCert rows.

9462.28.23 BEDC certificate machines, allegory machines, and biological branches

The DNA/window-six bridge also clarifies a larger methodological separation. BEDC is best treated as a Turing-realizable certificate machine; an allegory machine generates candidate symbolic patterns; DNA enters only as an external biological bridge branch. The slogan is: allegory proposes figures, BEDC certifies names, and DNA supplies a natural branch whose records can be encoded, classified, audited, and compared without becoming a BEDC kernel primitive.

Definition 9462.1876 (Finite certificate kernel)**.** *The BEDC finite certificate kernel is the dependency spine*

$$b_0, b_1 \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont}$$
$$\to \mathsf{Ask}, \mathsf{Sig}, \mathsf{sameSig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{NameCert}.$$

It generates finite marks, histories, continuation structure, question and signature structure, package/gap classification, and certified name admission.

Definition 9462.1877 (BEDC certificate machine)**.** *A BEDC certificate machine is the finite certificate kernel together with classifier formation, ledger policy, NameCert admission, derivation strength records, and cannot-claim boundaries. It accepts a candidate name* N *only when the relevant source, pattern, classifier, stability, and ledger fields have been supplied at a declared strength.*

Definition 9462.1878 (Turing-realizable certificate machine)**.** *A certificate machine is Turing-realizable when its public derivations can be encoded as finite strings and its certificate checks can be executed by a finite, Turing-computable procedure. For the BEDC finite kernel, the intended external model sends raw marks to finite binary tokens, histories to finite binary words,* Ext *to word extension,* Cont *to concatenation, query and signature rows to finite strings, package and gap rows to finite classifier fibers, and NameCert fields to finite certificate records.*

Definition 9462.1879 (Plain Turing machine and certificate machine)**.** *A plain Turing-machine description specifies an input, a transition operation, and an output. A BEDC certificate-machine description records source, pattern, classifier, stability, ledger, exactness, derivation strength, and cannot-claim boundaries. Thus BEDC is Turing-realizable without being exhausted by the input-output image of one plain machine.*

Definition 9462.1880 (Allegory machine and seed)**.** *An allegory machine is a semantic-pattern generator: it compresses mythic, metaphorical, numerical, geometric, linguistic, and biological motifs into candidate patterns. An allegory seed is such a candidate pattern before BEDC admission. Examples include phased creation motifs, five-phase systems, seven-day ledgers, twelve-orientation ledgers, four-base biological alphabets, window-six stop boundaries, local Q_2 tile dynamics, and codon-window inverse spectra.*

Definition 9462.1881 (Admission of an allegory seed)**.** *An allegory seed P is BEDC-admissible at strength s only when there is a derived name N_P with an acceptance gate*

$$\mathsf{AcceptGate}(N_P, s) := \mathsf{NameCert}(N_P) \wedge \mathsf{DerivCert}(N_P, s).$$

The strength s records whether the object remains a seed, a paper certificate target, a checked certificate target, or a bridge certificate target.

Definition 9462.1882 (Biological bridge branch)**.** *A biological bridge branch is an external natural system B admitted as a derived BEDC interface through source, pattern, classifier, stability, and ledger fields, together with explicit external biological tables or models. A biological branch is not a finite-kernel primitive.*

Definition 9462.1883 (DNA window-six biological branch)**.** *The DNA window-six branch has source objects including a four-phase alphabet, a codon cube Q_6, a translation classifier*

$$Q_6 \to \mathsf{AminoAcidClass} \oplus \mathsf{Stop},$$

a codon ledger, and a context ledger. Its dynamical context records sequence, reading frame, current codon window, ribosome or release-factor state, mutation and repair state, epigenetic or chromatin accessibility, and cellular context.

Definition 9462.1884 (Human DNA branch)**.** *The human DNA branch is a species-specific instance of the DNA window-six branch for Homo sapiens. Its source and ledger fields include genome sequence, assembly and coordinate system, gene intervals, transcripts, protein-coding frames, genetic-code table provenance, variant alleles, epigenetic marks, cell-type context, and annotation provenance. It is an external biological instance, not a kernel object.*

Theorem 9462.1885 (BEDC finite kernel is Turing-realizable)**.** *The BEDC finite certificate kernel has a Turing-realizable external model.*

Proof. Interpret b_0 and b_1 as finite binary tokens, histories as finite binary words, sameness as literal token or word equality, Ext as word extension, Cont as word concatenation, query and signature rows as finite string operations, package and gap rows as finite classifier fibers and inverse images, and NameCert fields as finite records. Equality, extension, concatenation, lookup, classification, and certificate-field checking are all finitary and Turing-computable. □

Theorem 9462.1886 (BEDC is not merely a plain Turing machine)**.** *BEDC is Turing-realizable, but it is not merely a plain input-output Turing machine.*

Proof. A plain Turing machine produces an output from a transition system. BEDC requires public certificate fields, ledger policy, derivation strength, and cannot-claim boundaries for admitted names. These obligations are not merely output values; they are auditable conditions on why a name is licensed and what it is not allowed to claim. □

Corollary 9462.1887 (BEDC as certificate machine)**.** *BEDC should be described as a certificate machine rather than merely as a computational automaton.*

Proof. This follows from Definition 9462.1886: BEDC public output is a licensed name with certificate and ledger obligations, not only a computed string. □

Theorem 9462.1888 (Allegory does not certify itself)**.** *An allegory machine can generate candidate patterns, but it does not by itself produce BEDC-certified names.*

Proof. An allegory machine supplies semantic compression, analogy, and symbolic direction. These do not by themselves supply SourceSpec, PatternSpec, ClassifierSpec, StabilityCert, LedgerPolicy, or a derivation-strength record. Thus an allegory seed has no BEDC certificate force until it is converted into a certified name. □

Corollary 9462.1889 (Allegory proposes and BEDC certifies)**.** *The relation between the allegory machine and BEDC is: allegory proposes candidate symbolic patterns, while BEDC tests, certifies, or rejects them.*

Proof. This is Definition 9462.1888 together with Definition 9462.1881. □

Theorem 9462.1890 (Allegory seeds enter only through NameCert)**.** *If an allegory seed P becomes an admissible BEDC object, it must do so through a derived name N_P equipped with NameCert and a recorded derivation strength.*

Proof. By Definition 9462.1881, admissibility at strength s requires $\mathsf{AcceptGate}(N_P, s)$, which includes both NameCert and DerivCert. An allegory seed has neither field by itself. □

Theorem 9462.1891 (DNA is not a BEDC kernel primitive)**.** *DNA cannot be a finite-kernel primitive of BEDC.*

Proof. The finite kernel contains marks, histories, sameness, extension, continuation, asking/signature, package/gap, and NameCert structure. DNA requires an external four-base alphabet, molecular complementarity, codon windows, translation classifiers, mutation and repair processes, cellular execution machinery, and biological context. These are not finite-kernel constructors. DNA can therefore enter only as a derived biological bridge interface. □

Corollary 9462.1892 (DNA as biological bridge branch)**.** *If DNA is admitted into BEDC, it is admitted as a biological bridge branch, with* $\mathsf{DNAWindow6Dynamics}^{\uparrow}$ *as a principal candidate interface.*

Proof. DNA is not primitive by Definition 9462.1891. External natural systems enter by the bridge discipline of Definition 9462.1882. □

Theorem 9462.1893 (Human DNA is species-specific bridge data)**.** *Human DNA is a species-specific instance of the DNA biological bridge, not a universal BEDC kernel object.*

Proof. The human branch requires Homo sapiens-specific genome assemblies, transcripts, gene intervals, alleles, regulatory context, epigenetic states, and annotation provenance. These are contingent external biological records. Kernel objects are generated by the finite BEDC kernel without such biological contingencies, so the human branch is a bridge instance rather than a kernel object. □

Corollary 9462.1894 (Human DNA requires ledger)**.** *The valid BEDC expression is that human DNA is a BiologicalBridge branch with source, classifier, context, and ledger fields. A genome string without context and ledger is not a complete BEDC object.*

Proof. This is the ledger reading of Definition 9462.1893 and Definition 9462.1884. □

Theorem 9462.1895 (DNA window-six results are bridge theorems)**.** *DNA/window-six reconstructions are biological-bridge theorems, not proofs that biology is forced by BEDC alone.*

Proof. The reconstructions require external biological data: the four-base alphabet, codon length three, genetic-code tables, translation and termination behavior, and species or code context. These are not generated solely by the finite BEDC kernel. Conversely, the occurrence of window-six structure in DNA supplies a strong bridge instance, not a proof of BEDC as a whole. □

Definition 9462.1896 (Research surface)**.** *The research surface has five parts. First, formalize BEDC as a Turing-realizable certificate machine and prove non-reduction to a plain input-output machine. Second, define allegory machines, allegory seeds, and NameCert admission gates for symbolic patterns. Third, define biological branch obligations: source, pattern, classifier, stability, ledger, bridge table, and cannot-claim boundary. Fourth, develop* $\mathsf{DNAWindow6Dynamics}^{\uparrow}$ *and the human DNA branch as external biological branches. Fifth, connect the branch to local window-six results: stop codons as* $\mathrm{U} + \mathrm{RR} + X_2$*,* UGG *as a reserved* 11 *corner, dual* $X_2/\{11\}$ *control tiles,* 00*-singleton sense-expansion tiles, codon-window inverse spectra, and alternative genetic-code reassignments.*

Definition 9462.1897 (Empirical bridge surface)**.** *The empirical biological surface compares standard and alternative genetic-code tables, stop-codon usage, readthrough dynamics, codon optimality, mRNA stability, ribosome-profiling pause rows, synthetic reporters for* UAA*,* UAG*,* UGA*, and* UGG*, and context-specific human variant annotations. These records test biological adequacy but do not replace NameCert or bridge certificate obligations.*

Theorem 9462.1898 (Certificate-allegory-branch cannot-claim boundary)**.** *The certificate/allegory/branch triad does not license any claim that BEDC is exactly one ordinary Turing machine, that allegories are automatically true, that an allegorical pattern is already a BEDC theorem, that DNA is a BEDC kernel primitive, that human DNA is generated by the finite kernel alone, that DNA/window-six proves BEDC, that BEDC alone proves the biological genetic code, that the human DNA branch is already a checked certificate, that a genome string without context and ledger is a complete BEDC object, or that BiologicalBridge data replace NameCert obligations.*

Proof. The Turing-machine refusal follows from Definition 9462.1886. The allegory refusals follow from Definition 9462.1888 and Definition 9462.1890. The DNA and human-branch refusals follow from Definition 9462.1891 and Definition 9462.1893. The bridge overclaim refusal follows from Definition 9462.1895. □

Corollary 9462.1899 (Allowed triad claim)**.** *The allowed current-state formulation is: BEDC is Turing-realizable as a finite certificate machine; the allegory machine generates patterns that may be certified or rejected; DNA is a BiologicalBridge branch rather than a kernel primitive; and human DNA is a species-specific external branch requiring source, classifier, context, and ledger fields.*

Concretized in §3355: Certificate machines are realized as finite kernel, accepted-observation, branch-refusal, audit-replay, and NameCert rows.

Concretized in §3366: Certificate-machine branches are realized as finite kernel-spine, admission-gate, derivation-strength, ledger-boundary, and branch-separation NameCert packets.

9462.28.24 Codon-cube boundary research program

The DNA window-six bridge has a precise local form. The global codon cube is the six-bit source space

$$Q_6 := \{0,1\}^6,$$

but the standard stop-codon theorem is not a theorem about the global no-adjacent subspace of Q_6. It is a theorem about a two-bit boundary inside the U + RR face. The refined slogan is therefore:

$$\text{codons live in } Q_6, \qquad \text{standard stops live in } \mathrm{U} + \mathrm{RR} + X_2^{\mathsf{fam}}.$$

Here

$$X_2^{\mathsf{fam}} := \{00, 01, 10\}$$

is the local Zeckendorf-legal family suffix square. The global Fibonacci window X_6 remains a separate finite-window surface.

Definition 9462.1900 (Codon cube and local stop boundary)**.** *Let the two-axis RNA or DNA readback be*

$$\begin{aligned} \mathrm{U/T} = 00, \quad \mathrm{C} = 01, \\ \mathrm{A} = 10, \quad \mathrm{G} = 11. \end{aligned}$$

The codon cube is

$$Q_6 := \{0,1\}^6 \cong (\{0,1\}^2)^3 \cong \{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}^3.$$

The stop face is the codon-cube face

$$\mathsf{StopFace} := \{\mathrm{U}xy \mid x, y \in \{\mathrm{A}, \mathrm{G}\}\} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}\}.$$

The local suffix boundary is obtained by projecting the last two positions to the family axis and retaining only $X_2^{\mathsf{fam}} = \{00, 01, 10\}$.

Definition 9462.1901 (Global and local no-11 spaces)**.** *The global no-adjacent window is*

$$X_6 := \{w \in Q_6 \mid w \textit{ has no adjacent } 11\}.$$

The local family-suffix window is

$$X_2^{\mathsf{fam}} := \{s \in \{0,1\}^2 \mid s \neq 11\} = \{00, 01, 10\}.$$

The DNA stop theorem uses X_2^{fam} *inside* $\mathsf{StopFace}$*; it does not identify stop codons with* X_6.

Theorem 9462.1902 (Local X_2 stop boundary)**.** *Under the two-axis RNA bridge,*

$$\mathsf{StopFace} \cap X_2^{\mathsf{fam}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}.$$

Proof. The stop face has four suffixes:

$$\begin{aligned} \mathrm{AA} &\longmapsto 00, \\ \mathrm{AG} &\longmapsto 01, \\ \mathrm{GA} &\longmapsto 10, \\ \mathrm{GG} &\longmapsto 11. \end{aligned}$$

Intersecting with $X_2^{\mathsf{fam}} = \{00, 01, 10\}$ keeps the first three suffixes and removes the GG corner. Hence the selected codons are exactly UAA, UAG, and UGA. □

Theorem 9462.1903 (Local X_2 is not global X_6)**.** *The standard stop-codon boundary is local to the family suffix and is not the global no-adjacent-*11 *subspace X_6 of the flattened codon cube.*

Proof. By Definition 9462.1090, the flattened strings for UAG and UGA contain adjacent 11 even though both are standard stop codons. Thus the global predicate defining X_6 would not select the standard stop set. The successful reconstruction is instead the two-bit family-suffix predicate of Definition 9462.1902. □

Corollary 9462.1904 (Codon cube, global window, local boundary)**.** *The DNA/window-six bridge separates three finite spaces:*

$$\begin{aligned} Q_6 &:= \{0,1\}^6 && \textit{global codon-cube source space}, \\ X_6 &:= \{w \in Q_6 \mid w \textit{ has no adjacent } 11\} && \textit{global Fibonacci window}, \\ X_2^{\mathsf{fam}} &:= \{00, 01, 10\} && \textit{local stop-face family boundary.} \end{aligned}$$

The bridge is strong because codons naturally inhabit Q_6 and the standard stop set is a local X_2^{fam} boundary inside that cube. It is not an identification of DNA with X_6 or with a window-six fold.

Definition 9462.1905 (Comparative genetic-code deformation ledger)**.** *For an external genetic-code table T, the comparison ledger records*

$$\mathsf{StopSet}_T, \quad \mathsf{SenseSet}_T, \quad \mathsf{role}_T(\mathrm{UAA}), \quad \mathsf{role}_T(\mathrm{UAG}), \quad \mathsf{role}_T(\mathrm{UGA}), \quad \mathsf{role}_T(\mathrm{UGG}).$$

The boundary deformation row is

$$\Delta_T := \mathsf{StopSet}_T \triangle \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\},$$

where $\triangle$ denotes symmetric difference. If $\Delta_T = \varnothing$, the standard L-shape is preserved by that table. If $\Delta_T \neq \varnothing$, the table is recorded as a deformation of the biological bridge target.

Concretized in §8745: The comparative genetic-code deformation ledger is realized as a finite NameCert carrier over stop/sense rows, codon-role rows, and the standard stop-boundary difference row.

Definition 9462.1906 (Empirical dynamics observables)**.** *The empirical layer tests whether the L-shape is only a finite set shape or also a dynamic boundary. The bridge may record observables such as*

$$\Pr(\text{termination} \mid \text{stop type}, +4\ \textit{context}, \text{species}, \text{release factor}),$$
$$\Pr(\text{readthrough} \mid \text{stop type}, \text{downstream context}),$$
$$\text{ribosome pause time}, \qquad \text{mRNA half life},$$
$$\text{codon optimality score}, \qquad \text{stop codon usage frequency}.$$

These quantities are external biological bridge rows. They may guide which dynamic fields are worth naming, but they do not become BEDC kernel primitives.

Definition 9462.1907 (Finite-state stop-boundary prototype)**.** *A minimal finite-state prototype has states*

$$\{\mathsf{scan}, \mathsf{elongate}, \mathsf{terminate}, \mathsf{readthrough}, \mathsf{repair}, \mathsf{mutate}\}$$

and alphabet $\{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}$*. The current window is a codon* $w_t = (b_t, b_{t+1}, b_{t+2})$*. Under the standard-code bridge, the transition rows include*

$$\begin{aligned}
\delta(\mathsf{elongate}, \mathrm{UAA}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UAG}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UGA}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UGG}) &= \mathsf{elongate\text{-}Trp}.
\end{aligned}$$

A stochastic extension may replace δ *by* $\Pr(\delta \mid w_t, \text{context})$ *to model readthrough, release-factor availability, or cellular state.*

Concretized in §2245: The finite-state stop-boundary transition matrix is realized as a concrete NameCert carrier over codon windows, state rows, and local stop-face transition rows.

Definition 9462.1908 (Work packages for the codon-cube boundary program)**.** *The research program has seven work packages:*

Finite modeling. *Define* $\mathsf{Base}^{\uparrow}$*,* $\mathsf{Codon}^{\uparrow} \cong Q_6$*,* $\mathsf{StopFace}$*,* $\mathsf{FamilySuffix}$*,* $\mathsf{ZLegalFamilySuffix}$*, the three-stop theorem, the excluded corner, and the RF L-shape bridge.*

Window clarification. *Separate* Q_6*,* X_6*, and* X_2^{fam}*, and prove that the strong bridge is* Q_6 *plus local* X_2^{fam}*, not DNA as global* X_6*.*

Genetic-code comparison. *Compare standard, mitochondrial, ciliate, protozoan, nuclear, bacterial, archaeal, and plastid tables by the deformation ledger* Δ_T*.*

Stop and codon usage statistics. *Relate stop identity, GC content, expression level, species lineage, tRNA or release-factor annotations, terminal context, and downstream context to stop usage.*

Translation dynamics. *Use ribosome-profiling, readthrough, pausing, mRNA stability, and protein-output rows as external dynamic evidence.*

Synthetic reporter validation. *Compare terminal-window replacements* UAA*,* UAG*,* UGA*, and* UGG *under controlled downstream and release-factor contexts.*

Finite-state prototype. *Implement the stop-boundary automaton with context-dependent transition probabilities as a bridge-side model.*

Theorem 9462.1909 (Codon-cube boundary cannot-claim boundary)**.** *The codon-cube boundary program does not license any claim that Zeckendorf alone proves the biological genetic code, that the DNA genetic code is a window-six fold, that the whole genetic code is global X_6, that BEDC* $\mathrm{End} = 11$ *is identical to a biological stop codon, that the family suffix* 11 *in* UGG *is the channel terminator, that the standard stop codons are universal across all genetic-code tables, that* UGG *must be tryptophan by BEDC logic alone, that synonymous codons are dynamically identical after amino-acid classification, that DNA state is exhausted by Q_6, that DNA has fully autonomous closed dynamics independent of cellular machinery, that biological tables replace* $\mathsf{NameCert}$ *obligations, or that* $\mathsf{DNAWindow6Dynamics}^{\uparrow}$ *is already checked.*

Proof. The global-window refusals follow from Definition 9462.1903. The channel and token-identity refusals follow from Definition 9462.1107. The dynamic state refusals follow from Definition 9462.1133. The microstate-refusal follows from Definition 9462.1135. The checked-status refusal follows because this program is a vision-strength bridge target and not a Lean target. □

Corollary 9462.1910 (Allowed codon-cube boundary claim)**.** *Under the two-axis DNA or RNA bridge and Zeckendorf-family suffix legality, the standard stop-codon set is structurally reconstructed as the local X_2^{fam} L-shape*

$$\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}\}$$

inside the $\mathrm{U} + \mathrm{RR}$ *face of the codon cube Q_6. The framework is a biological-bridge vision for symbolic-physical DNA dynamics, not a checked theorem that Zeckendorf alone forces biology.*

9462.28.25 Failure certificates and cannot-claim entries

Definition 9462.1911 (Failure certificate)**.** *A failure certificate for a candidate object N at strength s is a typed branch showing that a necessary field of the acceptance gate fails: missing source, missing classifier, non-transitive classifier, exactness failure, operation failing to descend, absent gap ledger, unstable completion, failed standard bridge, host primitive leakage, or theorem-checker failure.*

Theorem 9462.1912 (Failure blocks acceptance)**.** *If a failure certificate for N at strength s is present, then the acceptance gate for N at s is not satisfied.*

Proof. The acceptance gate is a conjunction of required fields. Each failure branch denies one required field or bridge condition, so the gate does not close. □

Definition 9462.1913 (Cannot-claim entry)**.** *A cannot-claim entry records claim text, status, the reason the claim is not currently claimable, and the required certificate upgrade.*

Theorem 9462.1914 (Cannot-claim registry is export discipline)**.** *The cannot-claim registry is part of BEDC's export surface, not a rhetorical appendix.*

Proof. Claims outside the certified surface are unsupported until the relevant certificate is displayed. The registry makes that refusal explicit and typed. □

Corollary 9462.1915 (Negative export knowledge)**.** *Stating that BEDC does not claim X can be a knowledge state when it is backed by a failure certificate or registry entry.*

Proof. Certificates show what is admitted; failure branches show what blocks admission. Both are auditable. □

9462.28.26 Prohibited scientific smuggling

Definition 9462.1916 (Scientific smuggling)**.** *Scientific smuggling occurs when a host model is treated as a BEDC primitive, empirical fit is promoted to universal law, equation form is treated as physical object, a schema horizon is reported as a closed theorem, a bridge sketch is reported as bridge certificate, a numerical empirical value is treated as kernel-derived constant, or external equality, quotient, or choice is used as public classifier.*

Theorem 9462.1917 (Scientific smuggling violates a gate)**.** *Every scientific smuggling pattern violates at least one of naming certificate discipline, acceptance gate discipline, standard bridge discipline, gap discipline, or axiom-purity.*

Proof. Host primitives violate no-host-leak. Empirical fit promoted to law violates Type IV gap discipline. Equation form promoted to object violates naming discipline. Schema horizon promoted to theorem violates declared strength. Bridge sketch promoted to bridge certificate violates bridge fields. Numerical value promoted to kernel constant violates empirical ledger discipline. External equality, quotient, or choice violates meta-closure refusal. □

Corollary 9462.1918 (Scientific writing must mark status)**.** *Scientific BEDC writing must mark status, layer, and claim type: whether the item is a record, invariant, model, bridge, empirical claim, prediction commitment, structure form, theorem, fit, or bridge correspondence.*

Proof. Different layers and statuses have different certificate obligations. Leaving them unmarked invites smuggling. □

9462.28.27 External mature systems

Definition 9462.1919 (Mature external system)**.** *Mature external systems include host libraries, set-theoretic foundations, standard model theory, classical analysis, differential geometry, physics textbooks, and experimental data standards. In BEDC they are bridge targets or explanatory references, not primitive sources.*

Theorem 9462.1920 (Mature systems are bridge targets)**.** *Mature external systems cannot replace internal BEDC certificates; they can only serve as bridge targets or references.*

Proof. Replacing the internal certificate by a mature host system would place host equality, quotients, choice, numbers, functions, or type and set semantics on the BEDC public surface. No-host-leak forbids this. □

Corollary 9462.1921 (BEDC bridges mature systems without fusing them)**.** *BEDC does not reject mature systems; it distinguishes bridge from fusion.*

Proof. The bridge lets a host reader recognize the BEDC object and lets BEDC accept finite translated theorem rows. The no-host-leak field prevents the host system from becoming a primitive source. □

9462.28.28 Physics certificate chain

Definition 9462.1922 (Multi-history physics chain)**.** *The multi-history physics route proceeds through certificates for multi-history configuration, cross-history causality, maximum causal rate, Lorentz-style geometry, curved spacetime, quantum state, and the relevant empirical or model bridges.*

Theorem 9462.1923 (Physics closes stepwise)**.** *The physics route cannot skip intermediate certificates and directly claim high-level physical objects.*

Proof. Each high-level object depends on source, classifier, stability, ledger, and bridge data from prior layers. Skipping a layer leaves a certificate field unsupplied. □

Corollary 9462.1924 (High-level physical objects are not primitives)**.** *Curved spacetime, fields, quantum states, Hilbert spaces, and particles must enter through the multi-history, invariant, bridge, and empirical ledger chain rather than as primitives.*

Proof. They stand far above the kernel and require mature mathematical carriers, physical interpretation, empirical bridge, and prediction gaps. □

9462.28.29 Synthesis

Theorem 9462.1925 (Scientific object and model-bridge synthesis)**.** *BEDC scientific content consists of records-side generated data, inter-history invariant structure, naming and derivation certificates, conservative bridges to mature models, empirical ledgers, prediction gaps, and falsification surfaces.*

Proof. Scientific objects are not primitives. Frame-independent physical content is inter-history invariant. Physical laws are empirical regularities plus Type IV gaps. Formal constants are not empirical values. Models are not worlds. Standard bridges are conservative, host-relative, and no-host-leak. Failure certificates and cannot-claim entries police the export surface. Therefore scientific content enters only through the stated certificate and ledger discipline. □

Corollary 9462.1926 (Scientific truth as audited stability)**.** *BEDC scientific truth is not absolute correspondence between law and world; it is ledgered records, stable invariants, conservative model bridges, marked prediction commitments, and open falsification surfaces.*

Concretized in §1239: Truth-residue ledgers realize scientific truth as finite record, invariant, bridge, prediction-gap, falsification, and non-finality NameCert rows.

Corollary 9462.1927 (Physical reality as stable relation)**.** *BEDC physical reality is not objecthood inside global spacetime; it is stable inter-history relational content that survives licensed classifier, invariant, and empirical ledger tests.*

Corollary 9462.1928 (Passage to philosophical traditions)**.** *After scientific objects and model bridges are certificate-disciplined, the framework can be compared with traditional philosophical positions only at precise structural contact points, rather than by identifying BEDC with any tradition wholesale.*

Concretized in §1273: Scientific model bridges are realized as finite scientific-object, model-audit, standard-bridge-audit, open-fit, failure-surface, transport, provenance, and NameCert rows.

Concretized in §1286: ScienceBridgeUp packages scientific objects, physical-model audit, reality-constrained truth, standard-bridge audit, and falsification gaps as one BEDC NameCert surface.

Concretized in §1285: Physical-law bridge packets realize law certificate, empirical persistence, scientific-object, bridge-audit, open-fit, and falsification rows as one NameCert carrier.

9462.29 BEDC and Philosophical Traditions

BEDC is not assigned to an inherited philosophical school. The comparison discipline is weaker and more precise: a tradition may identify a question whose structural core lands on a BEDC carrier, relation, classifier, ledger, certificate, gap, or T-position. The comparison is therefore a structural translation, not a doctrinal identity.

9462.29.1 Structural adjacency

Definition 9462.1929 (Philosophical genealogy)**.** *A traditional philosophical claim P has a genealogical relation to a BEDC structure B when a structural core of P can be translated as a BEDC carrier, relation, classifier, ledger, certificate, gap, or T-position. The relation does not say that BEDC belongs to the tradition containing P, nor that BEDC accepts that tradition's full metaphysical commitments.*

Definition 9462.1930 (Structural adjacency)**.** *Two traditional claims are structurally adjacent when they land on the same BEDC site. For example, bundle theories of the self, atman-negation readings, and second-order cybernetics all land on the observer-as-history site when restricted to the shared conclusion that there is no observer-substance behind the trace.*

Concretized in §1311: Structural adjacency is realized as a finite genealogy, BEDC-role, adjacency-witness, transport, route, provenance, and NameCert packet.

Theorem 9462.1931 (BEDC is not one tradition)**.** *BEDC cannot be reduced to any single inherited philosophical tradition.*

Proof. The object-language content of BEDC is generated by the finite kernel, internal sameness judgments, continuations, signatures, packages, gaps, and naming certificates. Traditional philosophical vocabulary belongs to interpretive metalanguage. Since the kernel chain does not take any traditional concept as primitive, no single tradition supplies the structure of BEDC. □

Corollary 9462.1932 (Comparisons need two labels)**.** *Every comparison with a philosophical tradition must name both the traditional claim and the BEDC site on which its structural core lands.*

Proof. Naming only the tradition projects external doctrine into BEDC. Naming only the BEDC site loses the explanatory genealogy. The two labels keep tradition as reading and BEDC as structure. □

9462.29.2 Spencer-Brown and primitive distinction

Definition 9462.1933 (Distinction genealogy)**.** *The Spencer-Brown-style claim that a formal system begins by drawing a distinction lands in BEDC on the primitive mark distinction:* `b0`*,* `b1`*, and constructor no-confusion.*

Theorem 9462.1934 (BEDC internalizes the distinction site)**.** *The thought that form begins with a distinction is internalized in BEDC as the constructor distinction between the two primitive marks.*

Proof. The two marks and their no-confusion are not derived from an earlier object. Writing the two constructors is already the minimal act of distinction needed for generation to begin. This is the BEDC site adjacent to the distinction-first reading. □

Corollary 9462.1935 (Distinction becomes certified generation)**.** *BEDC does not stop at the act of distinction; it sends distinction into histories, continuations, classifiers, ledgers, and naming certificates.*

Proof. The primitive distinction is the ground. The remaining BEDC layers are the certified development of that ground into trace, observation, object license, and boundary accounting. □

9462.29.3 Brouwer and time intuition

Definition 9462.1936 (Brouwer time site)**.** *The Brouwer-style claim that mathematics arises from the perception of time lands in BEDC on* E0/E1 *application, history growth, and local continuation streams.*

Theorem 9462.1937 (The Brouwer structural claim becomes Hist-time)**.** *The structural content of the Brouwer time reading is captured in BEDC by elementary history extension and local continuation.*

Proof. A history step is an elementary distinction entering a generated trace. A local continuation stream orders such steps from e to a displayed history. Constructive mathematics can then be read as the certified unfolding of this finite temporal trace structure. □

Corollary 9462.1938 (BEDC constructivism is not psychologism)**.** *BEDC does not reduce mathematics to psychological time experience.*

Proof. Hist is an inductive carrier and Cont is a formal relation. The Brouwer adjacency concerns the formal structure of time as distinction-order, not a mental state as mathematical substrate. □

9462.29.4 Bundle theory, no-self, and cybernetics

Definition 9462.1939 (Triple subject genealogy)**.** *Empiricist bundle theories of self, atman-negation readings, and second-order cybernetics share one BEDC landing site: observer equals history, observer identity is history-sameness under licensed comparison, and no observer-substance stands behind the trace.*

Theorem 9462.1940 (The shared subject conclusion is no hidden observer)**.** *The common structural conclusion of these subject traditions, as read through BEDC, is that there is no observer-substance behind the trace.*

Proof. The continuation signature has history endpoints and no subject slot. Observer identity is expressed through generated history and licensed comparison. Adding an observer-substance would require unlicensed transport beyond the kernel surface. □

Corollary 9462.1941 (No-subject is not anti-experience)**.** *BEDC excludes observer-substance, not focused record, self-proxy, conscious records-side carrier, or inscription event.*

Proof. The phenomenological layer admits focused histories and inscription events as records-side or boundary-side structures. What is excluded is a hidden carrier that owns those structures without a certificate. □

9462.29.5 Kant and conditions of possibility

Definition 9462.1942 (Kantian question)**.** *The Kant-style question asks for conditions of the possibility of experience or knowledge. In BEDC it becomes the question which generated carriers, classifiers, ledgers, certificates, and gap sockets make an object-claim or knowledge-claim admissible.*

Definition 9462.1943 (BEDC condition of possibility)**.** *A BEDC condition of possibility is not a form of a transcendental subject. It is a structural condition such as primitive distinction, history, continuation, internal sameness, naming certificate, ledger policy, or gap boundary.*

Theorem 9462.1944 (Conditions of possibility become certificate conditions)**.** *BEDC rewrites the question of conditions of possibility as certificate and boundary conditions for object-layer entry.*

Proof. Objects require naming certificates. Knowledge requires witnesses. Commitments beyond closed records require gap exposure. The conditions under which a claim can appear are therefore generated, classified, ledgered, stabilized, and boundary-marked conditions. □

Corollary 9462.1945 (BEDC is not Kantianism)**.** *BEDC is adjacent to the possibility-condition problem but does not share a subject-centered transcendental form.*

Proof. The landing site is Hist, Cont, NameCert, and gap discipline, not a transcendental subject. The observer has already been identified with history. □

9462.29.6 Husserl and constitution

Definition 9462.1946 (Husserlian question)**.** *The Husserl-style question asks how an object is constituted for consciousness. In BEDC it becomes the question how a stable pattern in history and continuation receives a naming certificate under a focused ledger.*

Definition 9462.1947 (BEDC constitution)**.** *Constitution in BEDC is the records-side process by which a history trace, continuation pattern, classifier stability, ledgered compression, and naming certificate make a pattern available as an object.*

Theorem 9462.1948 (Object constitution is licensed stable pattern)**.** *For BEDC, an object appearing to consciousness in the object layer means that a focused records-side ledger contains a stable pattern licensed by classifier and naming certificate.*

Proof. The consciousness carrier is a focused history with local continuation trace, selector ledger, and self-proxy. A mature object is a licensed name. Therefore object constitution in this layer is the certification of stable records-side pattern. □

Concretized in §1284: Object-constitution ledgers realize stable appearance as focused history, continuation, classifier, ledger, gap, provenance, and NameCert rows.

Corollary 9462.1949 (BEDC phenomenology is records-side phenomenology)**.** *BEDC is adjacent to phenomenology through appearance, record, and constitution, but it requires every constitutive step to be written as finite ledger, classifier, certificate, or gap.*

Proof. Unledgered phenomenal content has no object-layer witness. A phenomenon enters BEDC only through its records-side carrier or through a gap-tagged boundary reading. □

9462.29.7 Wittgenstein and use

Definition 9462.1950 (Use and rule question)**.** *The Wittgenstein-style question links meaning to use and rule-following to public practice. In BEDC it lands on licensed use under* NameCert *and classifier-stable continuation with a ledger.*

Definition 9462.1951 (BEDC use)**.** *A use of a name N is legitimate when the item used falls under the source and classifier fields of N, the use is stable under the stability certificate, and hidden material is tracked by the ledger policy.*

Theorem 9462.1952 (Meaning is certified use)**.** *In BEDC, the meaning of a name is given by its auditable use rules.*

Proof. The naming certificate states source, pattern, classifier, stability, and ledger fields. Without these fields the name is not licensed. Meaning is therefore neither private image nor bare reference, but certificate-governed use. □

Corollary 9462.1953 (Rule-following is audited, not merely social)**.** *BEDC is adjacent to the public-checkability side of rule-following, but its use is classifier-, witness-, ledger-, and stability-governed rather than ordinary social habit.*

Proof. Social practices can drift and remain implicit. BEDC public use must be auditable by certificate fields. □

9462.29.8 Whitehead and process

Definition 9462.1954 (Process question)**.** *The process-philosophy question says that reality is process rather than static substance. Its BEDC landing site is* Cont *as universal verb, history growth, and naming certificates as stabilized process-interfaces.*

Definition 9462.1955 (BEDC process)**.** *A BEDC process is a displayed continuation* $\mathsf{Cont}(h, d, h')$ *and its specializations under certificates: observation, time step, computation, morphism, type checking, or continuity modulus.*

Theorem 9462.1956 (BEDC is certificate process theory)**.** *BEDC is structurally adjacent to process philosophy in that continuation precedes mature objecthood.*

Proof. Distinction and history precede objects. Observation and time are continuation readings. Mature objects are names licensed after stable continuation behavior has been classified and ledgered. Objecthood is therefore downstream from process. □

Corollary 9462.1957 (BEDC is not Whiteheadian metaphysics)**.** *BEDC shares process priority but does not import actual occasions, prehension, or organic metaphysics.*

Proof. Its process site is the finite continuation relation and its certificate discipline, not a general metaphysical category. □

9462.29.9 Tarski, Godel, and Muenchhausen

Definition 9462.1958 (Formal-limit tradition)**.** *The formal-limit tradition includes incompleteness, undefinability, bootstrap impossibility, and halting undecidability. Its BEDC landing site is the closed-ground-loop and open-meta-loop architecture.*

Theorem 9462.1959 (Limit results are open-meta-loop phenomena)**.** *BEDC reads incompleteness, undefinability, bootstrap impossibility, and halting obstruction as different faces of the open meta loop.*

Proof. The ground loop must close so generation can begin. The meta loop must remain open so the system does not internalize total truth, runtime, or halting decision. The limit phenomena arise when a system attempts to close the latter. □

Corollary 9462.1960 (BEDC does not bypass formal limits)**.** *BEDC does not claim to overcome formal limits; it isolates the minimal ground closure required for formal systems while refusing meta closure.*

Proof. The primitive distinction is the ground payment. Truth and runtime remain outside the object layer. This exposes the boundary rather than removing it. □

Corollary 9462.1961 (Reflection is not a strange loop)**.** *BEDC self-description is layered reflection, not a collapse of the describing and described layers.*

Proof. Collapsing the two layers would recreate the truth-totalization problem. BEDC keeps structural description, described structure, and external checking distinct. □

Concretized in §1316: UseProcessLimitUp realizes certified use, rule audit, continuation process, and formal-limit refusal as one finite NameCert packet.

Concretized in §1313: Certified use and process-priority rows are realized as one finite BHist, hsame, Cont, Pkg, and NameCert boundary packet.

Concretized in §1315: Use-process-limit ledgers realize certified use, continuation-process, and formal-limit refusal as one finite NameCert packet.

Concretized in §1317: Certified-use process packets realize public use, rule witness, continuation route, refusal ledger, and NameCert rows as one finite carrier.

9462.29.10 Relational physics

Definition 9462.1962 (Relational physics genealogy)**.** *Relational quantum mechanics, information-first physical readings, noncommutative geometry, and spin-network approaches share a structural claim that physical content is relational and not located in a privileged global frame.*

Definition 9462.1963 (BEDC relational physics site)**.** *The BEDC site for relational physics is multi-history configuration, inter-history locality, inter-history invariant, no global synchronization, and maximum cross-history rate schema.*

Theorem 9462.1964 (BEDC is lower-level relational physics)**.** *BEDC is adjacent to relational physics because frame-independent physical content is carried by inter-history invariants, while BEDC constructs the relational substrate from distinction, history, and continuation.*

Proof. Multiple histories do not share a base history or time-step index. The kernel provides no global frame. Stable physical content must therefore appear as witnessed invariant relation between histories. BEDC differs from mature relational physics traditions by placing this relation on a more primitive distinction-and-continuation substrate. □

Corollary 9462.1965 (Relational physics remains a schema horizon)**.** *The current relational physics layer is a schema horizon rather than a closed derivation of physical theory.*

Proof. The carriers, causal dependence relation, maximum-rate invariant, geometry, and empirical bridges require their own witnesses and naming certificates. Until those are supplied, the physical reading remains structural. □

Concretized in §1275: RelationalPhysicsUp realizes the relational physics site as finite inter-history invariant, locality, physical-model audit, hsame, Cont, Pkg, and NameCert rows.

9462.29.11 Composite positioning

Definition 9462.1966 (External labels)**.** *External readers may call BEDC formalist, constructivist, phenomenological, process-oriented, or relational realist. Each label names a structural adjacency, not an identity.*

Definition 9462.1967 (Composite position)**.** *The composite position of BEDC is auditable generativity: primitive distinction as ground, histories and continuations as generation, naming certificates as object license, gaps and T as boundary honesty, and inter-history invariants as relational physical interface.*

Theorem 9462.1968 (Genealogy synthesis)**.** *The philosophical genealogy of BEDC is a network of structural translations: primitive distinction for distinction-first traditions; Hist-time for Brouwer; observer-as-history for no-substance subject traditions; certificate conditions for possibility questions; focused records-side licensing for constitution; certified use for meaning; continuation as universal verb for process; open meta loop for formal-limit traditions; and inter-history invariants for relational physics.*

Proof. Each translation lands on an internal BEDC site already introduced in the framework. None requires the philosophical tradition itself to become an object-language primitive, and none imports the tradition's full metaphysical surplus. □

Corollary 9462.1969 (BEDC's philosophical identity)**.** *The most accurate philosophical identity of BEDC is auditable generativity, not formalism, constructivism, phenomenology, process philosophy, or relational realism taken alone.*

Proof. Each external label captures one face: formal carrier discipline, constructive witnesses, focused records-side experience, continuation priority, or relational physical content. The BEDC structure combines these under certificate and gap discipline. □

9462.29.12 Synthesis

Theorem 9462.1970 (Philosophical genealogy synthesis)**.** *BEDC's relation to philosophical traditions is structural relocation: for each tradition, the BEDC comparison identifies the carrier, relation, certificate, or gap site at which the tradition's structural question becomes auditable.*

Proof. The preceding comparisons show that each tradition lands on a precise site: primitive distinction, history time, observer-as-history, certificate condition, focused records-side constitution, certified use, continuation process, open meta loop, or inter-history invariant. The BEDC object language remains its finite kernel and certificate discipline. Therefore the relation is translation, not membership. □

Corollary 9462.1971 (Comparative writing discipline)**.** *Any comparison with a philosopher or tradition should state the traditional claim, the BEDC site, the retained structural content, the rejected surplus, and the audit status of the comparison.*

Corollary 9462.1972 (Passage to audit methodology)**.** *After the traditional comparisons are localized, the next layer should turn the whole framework into writing rules for definitions, theorems, proof strength, gap reports, and prevention of philosophical smuggling.*

9462.29.13 Geometric Algebra as a Closure Model

Definition 9462.1973 (Geometric algebra genealogy)**.** *The geometric algebra tradition lands in BEDC on the site where a generated object carries a visible metric readout, a unit row, and a ledger for oriented residual data. The comparison does not make Clifford algebra a BEDC primitive. It says that the Clifford construction is a compact algebraic model of the three-property discipline.*

Definition 9462.1974 (Observation vector packet)**.** *An observation vector packet over a quadratic interface (V, Q) is a triple*

$$v^{\uparrow} = (v, Q(v), \Lambda(v))$$

where v is the displayed vector row, $Q(v)$ is the squared-length readout, and $\Lambda(v)$ records the source, basis, orthogonality, and projection ledgers under which the vector is admitted.

Definition 9462.1975 (Geometric product ledger split)**.** *For vectors $u, v \in V$, a geometric product ledger split records*

$$uv = u \cdot v + u \wedge v,$$

with

$$u \cdot v = \frac{1}{2}(uv + vu), \qquad u \wedge v = \frac{1}{2}(uv - vu).$$

The symmetric part is the metric or projection readout. The antisymmetric part is the direction, area, order, and generation ledger.

Theorem 9462.1976 (The geometric product separates readout and ledger)**.** *In a geometric product ledger split, the exchange-invariant part of uv is exactly the metric readout, while the exchange-odd part is the oriented ledger readout.*

Proof. The decomposition in Definition 9462.1975 gives

$$uv + vu = 2u \cdot v$$

and

$$uv - vu = 2u \wedge v.$$

The first expression is fixed by exchanging u and v; the second changes sign. Thus the symmetric component records the visible metric projection, and the antisymmetric component records precisely the orientation and order data lost by the symmetric projection. □

Theorem 9462.1977 (The Pythagorean square is a two-axis unit readout)**.** *Let e_1, e_2 be orthogonal unit vectors with*

$$e_1^2 = e_2^2 = 1, \qquad e_1 e_2 = -e_2 e_1.$$

For scalar coefficients a, b and

$$x = ae_1 + be_2,$$

one has

$$x^2 = a^2 + b^2.$$

Proof. Expanding the square gives

$$x^2 = a^2e_1^2 + abe_1e_2 + bae_2e_1 + b^2e_2^2.$$

The scalar coefficients commute, while the two unit axes anticommute. Therefore the cross term is

$$ab(e_1e_2 + e_2e_1) = 0.$$

Since $e_1^2 = e_2^2 = 1$, the remaining part is $a^2 + b^2$. □

Definition 9462.1978 (Quadratic norm three-level readout)**.** *The quadratic norm row has three BEDC-visible projections:*

- *the Boolean two-axis skeleton, read by* $a^2 + b^2 \bmod 4$*;*
- *the finite prime-circle readout, read over odd prime fields by* $a^2 + b^2 \bmod P$*;*
- *the global norm readout, read in* $\mathbb{Z}[i]$ *by* $N(a + bi) = a^2 + b^2$*.*

The first projection exposes the parity obstruction that prevents $3 \bmod 4$ *from being a two-square readout. The second exposes finite quadratic phase data. The third exposes the norm-one surface* $\alpha/\overline{\alpha}$ *whenever a global factorization row is displayed.*

Theorem 9462.1979 (Clifford algebra is the minimal three-property closure)**.** *Let* (V, Q) *be a quadratic interface over a scalar carrier in which* 2 *is invertible. Among associative algebras receiving* V *and satisfying the unit row*

$$v^2 = Q(v)1$$

for every $v \in V$*, the Clifford algebra* $\mathrm{Cl}(V, Q)$ *is the initial closure. Its blade rows retain the antisymmetric direction, area, and higher-volume ledgers generated by the vector rows.*

Proof. The Clifford algebra is the quotient of the tensor algebra $T(V)$ by the relations

$$v \otimes v = Q(v)1.$$

Therefore every algebra B with a map $i : V \to B$ satisfying $i(v)^2 = Q(v)1$ receives a unique algebra homomorphism from $\mathrm{Cl}(V, Q)$. Polarizing the square relation gives the symmetric row

$$uv + vu = 2u \cdot v.$$

For orthogonal vectors this yields $uv = -vu$. The antisymmetric products survive as blade rows rather than being deleted. Thus the construction is minimal for the unit row and closed for the oriented ledger rows. □

Corollary 9462.1980 (Commutativity is a ledger-forgetting projection)**.** *The familiar commutative readout is obtained from the geometric product only after the oriented wedge ledger has been projected away.*

Proof. By Definition 9462.1975, noncommutativity is measured by

$$uv - vu = 2u \wedge v.$$

If a readout keeps only the symmetric component $u \cdot v$, then the exchange-odd ledger row is invisible and the projected product appears commutative. This is a projection of the full carrier, not the full carrier itself. □

Definition 9462.1981 (Clifford no-hidden-residual schema)**.** *The Clifford no-hidden-residual schema says that a nonzero residual component of a geometric product must have one of two addresses: it is detected by a finite blade, finite basis, finite phase, or finite packet read; or it enters a tail ledger for unbounded grade, scale, packet, analytic, or physical escape.*

Theorem 9462.1982 (Geometric algebra explains the shared residual pattern)**.** *The RH circle-triangle route and the Yang–Mills mass-gap route share the same BEDC residual pattern visible in geometric algebra: a unit surface is legal only when every nonunit or nonclosed residual is either finite-detected or ledgered as tail data.*

Proof. In the circle-triangle route of chapter 9323, the unit surface is the critical or unit-circle readout and a radial residual must be detected by a finite projection or recorded in a tail ledger. In the Yang–Mills route, the unitary compact holonomy and tube certificates feed the Hilbert spectral surface; by Definition 9462.1055, a disappearing mass gap must be a finite Wilson packet residual, a finite tube defect, an RG scale-tail defect, or another displayed physical tail. These are instances of the same schema: residual data cannot be erased by passing to a projection. □

Corollary 9462.1983 (Geometric algebra is a structural model, not a proof substitute)**.** *The geometric algebra reading organizes the common closure pattern of quadratic norm, unit circle, phase, orientation, RH residuals, and Yang–Mills soft modes. It does not replace the analytic, arithmetic, spectral, or gauge-theoretic certificates required in the individual routes.*

Proof. Definition 9462.1979 gives a universal algebraic model for vector-square and orientation ledgers. RH and Yang–Mills additionally require their own finite-window, Fourier, Lee–Yang, tube, RG, spectral-convergence, and physical Hilbert rows. The Clifford comparison identifies a shared carrier discipline; it does not discharge those route-specific obligations. □

Concretized in §1268: Tradition comparison boundaries are realized as finite source, landing, rejected-surplus, structural-distinction, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1350: Tradition landing ledgers realize structural translation, rejected surplus, distinction, transport, route, provenance, and local NameCert rows.

9462.30 Audit Methodology and Writing Discipline

The preceding layers give the structural content of auditable generativity. The present layer turns that content into writing rules. Every claim must state its carrier, classifier, ledger, theory-closure grade, formal-verification grade, gap or refusal surface, and no-leakage discipline. Theory closure and formal verification are reported on independent axes and never substitute for one another.

9462.30.1 Writing items

Definition 9462.1984 (Writing item)**.** *A BEDC writing item is an eight-field packet: kind, carrier, classifier, ledger, theory-closure status, formal-verification status, gap report, and claim text. The kind records whether the item is a definition, theorem, corollary, proof, reading, schema, roadmap, or refusal. The claim text is the strongest sentence the item permits the reader to believe.*

Definition 9462.1985 (Minimal legality)**.** *A writing item is minimally legal when it states the type of claim, the carrier it lives on, the way identity or classification is compared, whether hidden material is ledgered, its theory and verification status, whether it has a gap, and what it does not claim.*

Theorem 9462.1986 (Writing items must be auditable)**.** *Every concept entering the BEDC philosophy framework must be written as an auditable writing item.*

Proof. Objects require naming certificates. Positive content requires witnesses. Boundary commitments require gap exposure. Model claims require bridge discipline. Tradition comparisons require a BEDC site. Without carrier, classifier, ledger, status, and gap or refusal surface, the claim cannot be audited. □

Corollary 9462.1987 (Philosophical concepts are not bare prose)**.** *Claims such as consciousness, universe, freedom, law, or mature mathematics must be rewritten as carriers, slices, socket types, certificate surfaces, or refusal entries.*

Proof. The bare word does not state scope, carrier, witness, gap, or status. The rewrite supplies those audit fields. □

9462.30.2 Theory closure and formal verification

Definition 9462.1988 (Theory-closure axis)**.** *The theory-closure axis records paper-side closure: seed, obligation, scoped closure, public closure, bridged closure, and mature package closure. It concerns source, classifier, exactness, ledger, stability, scope, public object surface, bridge obligations, and theorem inventory.*

Definition 9462.1989 (Formal-verification axis)**.** *The formal-verification axis records Lean-side and audit-side status: unformalized, formal target, encoded definition, checked scaffold, checked theorem, audit clean, axiom clean, and checked bridge. It concerns declarations, encoded carriers, theorem proofs, marker audit, transitive axiom purity, and no-host-leak checking.*

Definition 9462.1990 (Theory gate)**.** $\mathsf{TheoryGate}(N, c)$ *means that N has the naming and closure certificates needed for theory-closure grade c.*

Definition 9462.1991 (Verified gate)**.** $\mathsf{VerifiedGate}(N, c, v)$ *is the conjunction of* $\mathsf{TheoryGate}(N, c)$ *and formal-verification status v for the relevant target surface.*

Theorem 9462.1992 (Theory closure and formal verification are independent)**.** *Theory closure does not imply formal verification, and formal verification does not imply theory closure.*

Proof. Theory closure depends on paper-side source, classifier, exactness, ledger, stability, scope, bridge, and public-surface witnesses. Formal verification depends on declarations, type checking, proof checking, marker resolution, and axiom-dependency audits. A paper proof may be closed without a checked theorem, and a checked theorem may exist without the surrounding paper-side object being closed. □

Corollary 9462.1993 (Checked is not closed)**.** *A checked Lean theorem reports formal-verification status only; it does not by itself close the corresponding object theory.*

Proof. Formal status lacks the full paper-side scope, ledger, closure, and bridge obligations unless those are separately supplied. □

Corollary 9462.1994 (Paper closed is not machine checked)**.** *Paper closure reports theory-closure status only; it does not imply machine checking.*

Proof. The paper witness may fill the theory fields while no canonical Lean target has yet been checked. □

Concretized in §1235: Writing-item audit packets realize the eight-field writing item, minimal legality, two-axis status separation, gap exposure, and non-escape claim boundary as a finite NameCert carrier.

9462.30.3 Closure status blocks

Definition 9462.1995 (Closure status block)**.** *A closure status block is the binding chapter record containing theory closure, exact closed scope, formal status, canonical target when present, bridge status, not-claimed surface, and upgrade path.*

Definition 9462.1996 (Binding commitment)**.** *The closure status block is the strongest export claim the chapter may make.*

Theorem 9462.1997 (Closure status bounds prose)**.** *If prose claims more than the closure status block permits, the prose claim is rejected.*

Proof. The block states grade, scope, formal status, bridge status, refusal surface, and upgrade path. A stronger prose claim would bypass the binding commitment. □

Corollary 9462.1998 (Prose strength has an upper bound)**.** *The allowed claim of a chapter is bounded above by its closure status block.*

Concretized in §1234: Auditable writing items are realized as finite transcript envelopes carrying status, anchor, replay, gap, ledger, and not-claimed rows.

Concretized in §1222: Writing-item audit packets realize kind, carrier, classifier, ledger, closure status, formal status, gap report, and permitted claim text as one finite NameCert surface.

9462.30.4 Definitions

Definition 9462.1999 (BEDC definition)**.** *A BEDC definition states carrier, constructor or generator, classifier, ledger, scope, forbidden readings, and status. It names generated and stabilized behavior rather than creating an object from nothing.*

Theorem 9462.2000 (Definitions require carriers)**.** *A definition without a carrier is explanatory vocabulary, not a BEDC definition.*

Proof. Without a carrier the definition has no generated source, no identity surface, and no place where certificate obligations can be checked. □

Theorem 9462.2001 (Definitions require forbidden readings)**.** *Every BEDC definition must state what it is not.*

Proof. Many BEDC names reuse mature language such as universe, consciousness, law, real, or God. Without forbidden readings, mature external meanings are projected back into the kernel. □

Corollary 9462.2002 (Definition template)**.** *A robust definition states: carrier, source, classifier, ledger, scope, forbidden readings, and status.*

9462.30.5 Theorems and corollaries

Definition 9462.2003 (BEDC theorem)**.** *A BEDC theorem states explicit carrier, certificate, and gap assumptions; an exact scoped conclusion; a finite witness-chain proof; and an audit line recording theory and verification grade.*

Definition 9462.2004 (Theorem scope)**.** *The scope of a theorem is the exact class of carriers, certificates, or objects for which the theorem is licensed.*

Theorem 9462.2005 (Unscoped theorems are refused)**.** *A theorem without explicit scope cannot be used as a BEDC theorem.*

Proof. Without scope, the carrier, classifier, ledger, and closure conditions are unknown. A claim over all objects, observers, worlds, or mathematics typically requests unrestricted universal closure and must be downgraded or socketed. □

Corollary 9462.2006 (Theorem titles must not overstate)**.** *Theorem titles should name the exact carrier and conclusion rather than claiming that BEDC proves causality, consciousness, or all mathematics.*

Definition 9462.2007 (BEDC corollary)**.** *A BEDC corollary is a direct consequence of prior theorems that adds no new carrier, witness, or gap.*

Theorem 9462.2008 (Corollaries do not increase strength)**.** *The closure and verification grade of a corollary cannot exceed the weakest grade of the results it depends on.*

Proof. The corollary contributes no new witness. It reuses prior proof surfaces, so it cannot raise theory closure or formal verification. □

Corollary 9462.2009 (Philosophical corollaries are conservative)**.** *Philosophical corollaries may explain, restrict, or translate; they may not add new ontological commitments.*

9462.30.6 Proofs

Definition 9462.2010 (BEDC proof discipline)**.** *A BEDC proof is a finite witness chain passing through source witness, classifier witness, transport or exactness witness, ledger witness, stability witness, and conclusion.*

Concretized in §1219: Proof-trace audit witnesses are realized as finite proof trace, transport, ledger, refusal, audit-verdict, provenance, and NameCert rows.

Concretized in §1334: Definition, theorem, and proof discipline are realized as one finite NameCert packet with separated scope, proof-route, refusal, and two-axis status rows.

Concretized in §1346: Trace-verdict boundaries are realized as finite proof-trace, refusal, verdict, interface-target, provenance, and local NameCert rows.

Definition 9462.2011 (Forbidden proof resources)**.** *Forbidden proof resources include hidden choice, host equality as BEDC equality, quotient collapse, propositional extensional rewrite, global truth predicate, global clock, unledgered compression, and unscoped universal claim.*

Theorem 9462.2012 (Proofs must display transport)**.** *Any proof crossing a classifier, bridge, package, or history-identity surface must display transport.*

Proof. Without displayed transport, the proof treats a relation as equality. That is an extensional-identification socket rather than an internal proof. □

Corollary 9462.2013 (Obviousness is not a witness)**.** *Words such as obvious, natural, standard, or by the usual definition do not close a BEDC proof unless accompanied by the relevant witness chain.*

9462.30.7 Gap reports

Definition 9462.2014 (Gap report)**.** *A gap report records site, requested supply, socket type, gate, consequence if internalized, and upgrade path.*

Definition 9462.2015 (Gap status)**.** *A gap is open when it is identified without an upgrade path, deferred when it has an upgrade path, and forbidden when the requested move is principally refused.*

Theorem 9462.2016 (Gaps must be typed)**.** *An untyped gap is not a complete BEDC gap report.*

Proof. The report must say whether the missing supply is unrestricted induction, witness, identity, causal persistence, inter-history coordination, inscription novelty, self-reference closure, or a stack of these. Different kinds have different gates. □

Corollary 9462.2017 (Gaps are positive honesty structures)**.** *A gap is a visible boundary, not a proof failure hidden in prose.*

9462.30.8 Cannot-claim registry

Definition 9462.2018 (Cannot-claim entry)**.** *A cannot-claim entry records claim text, status, reason not claimable, and required upgrade. Its status is forbidden, deferred, or partial.*

Definition 9462.2019 (Cannot-claim registry)**.** *The cannot-claim registry is the finite export surface of claims that the manuscript explicitly does not make.*

Theorem 9462.2020 (Refusals must be recorded)**.** *BEDC writing must explicitly record what it does not claim.*

Proof. The certified surface is finite. If unsupported adjacent claims are left implicit, readers may infer them from nearby positive results. The registry prevents that inference. □

Corollary 9462.2021 (Cannot-claim is not roadmap)**.** *The cannot-claim registry is not a future schedule.*

Proof. A roadmap names intended work. A cannot-claim entry names a current refusal. A deferred refusal may later be removed only after its upgrade is discharged; a forbidden refusal has no upgrade path. □

9462.30.9 Failure certificates

Definition 9462.2022 (Failure certificate)**.** *A failure certificate* $\mathsf{FailureCert}(N, c, v)$ *records a typed reason why a claimed theory grade* c*, formal status* v*, or their combination fails. Branches include missing source, missing classifier, classifier failure, exactness failure, descent failure, absent gap ledger, unstable completion, overstated scope, failed bridge, host primitive leakage, checker failure, audit failure, and axis confusion.*

Theorem 9462.2023 (Failure blocks the corresponding gate)**.** *Theory-side failure blocks the theory gate, verification-side failure blocks formal status, and axis-confusion failure blocks the verified gate.*

Proof. Theory-side branches deny source, classifier, exactness, descent, ledger, stability, scope, bridge, or no-host-leak fields. Verification-side branches deny checking or audit cleanliness. Axis-confusion denies the independence of the two axes. □

Corollary 9462.2024 (Failure is certificate dual)**.** *Certificates license export; failure certificates block export. Both are typed audit objects.*

Concretized in §1237: Typed failure certificates turn the gap, failure, and bridge discipline into a finite NameCert packet.

Concretized in §1233: Typed two-axis failure certificates are realized as finite gate-blocking NameCert packets.

9462.30.10 Standard bridges

Definition 9462.2025 (Standard bridge writing form)**.** *A standard bridge must display encoding, decoding, return witnesses in both directions, finite theorem preservation, and no-host-leak.*

Definition 9462.2026 (Finite preservation list)**.** *The theorem-preservation field is a finite list of host theorems paired with BEDC counterparts and translation witnesses. No theorem outside the list is claimed preserved.*

Definition 9462.2027 (No-host-leak discipline)**.** *No-host-leak states that host numbers, quotients, function spaces, equality, choice, and propositional extensionality do not enter the BEDC public surface.*

Theorem 9462.2028 (Standard bridge is not fusion)**.** *A standard bridge is conservative recognition, not fusion with the host theory.*

Proof. The bridge compares BEDC and host objects through explicit maps and finite preservation rows. No-host-leak prevents host primitives from becoming BEDC inputs. □

Corollary 9462.2029 (Bridge claims are host-relative)**.** *A bridge claim must name the host theory. There is no absolute bridge independent of host semantics.*

Concretized in §1229: Gap, failure, and bridge audit rows are realized as one finite NameCert packet with axis-separation and no-host-leak boundaries.

9462.30.11 Philosophical comparison

Definition 9462.2030 (Tradition map)**.** *A comparison between BEDC and a philosophical tradition is a packet of traditional claim, BEDC site, preserved structure, rejected surplus, and audit status.*

Theorem 9462.2031 (Comparisons are not membership claims)**.** *BEDC comparisons with traditions must not become claims that BEDC is Kantian, Husserlian, Buddhist, Whiteheadian, or otherwise a member of the compared tradition.*

Proof. The object language contains BEDC carriers, relations, certificates, ledgers, and gaps, not the tradition's full metaphysical vocabulary. The comparison maps a structural question to a BEDC site. □

Corollary 9462.2032 (Rejected surplus is mandatory)**.** *Every philosophical comparison must state what BEDC does not accept from the tradition.*

9462.30.12 Smuggling

Definition 9462.2033 (Philosophical smuggling)**.** *Philosophical smuggling introduces an external philosophical entity as a BEDC internal object: subject substance, world-in-itself, global truth, absolute now, mental substance, or God as object.*

Definition 9462.2034 (Mathematical smuggling)**.** *Mathematical smuggling introduces mature mathematical objects such as natural numbers, real numbers, sets, functions, limits, quotients, topologies, or categories as BEDC primitives.*

Definition 9462.2035 (Host smuggling)**.** *Host smuggling exposes host equality, host numbers, host lists, host function spaces, quotient soundness, choice, or propositional extensionality on the BEDC public surface.*

Theorem 9462.2036 (The three smuggling forms share one shape)**.** *Philosophical, mathematical, and host smuggling all disguise uncertified external supply as internal content.*

Proof. The philosophical case imports an unwitnessed entity, the mathematical case imports a mature object without certificate, and the host case imports implementation primitives without bridge. All bypass naming, closure, formal status, gap, or bridge discipline. □

Corollary 9462.2037 (External concepts have two routes)**.** *An external concept can enter BEDC only by certificate and gate, or by gap, socket report, and cannot-claim entry.*

Concretized in §1157: Host primitive leakage is realized as a finite diagnostic refusal packet for public-surface host imports.

Concretized in §1329: Smuggling-pattern classifiers realize the philosophical, mathematical, and host smuggling discipline as one finite refusal-and-gate NameCert packet.

9462.30.13 Research template

Definition 9462.2038 (Ten-field research template)**.** *Every proposed object or philosophical concept should state traditional target, BEDC source, carrier choice, classifier choice, stability obligations, ledger, core theorem list, exactness predicate, standard bridge status, and combined theory and formal status.*

Theorem 9462.2039 (The template prevents overclaiming)**.** *The ten-field template blocks the common overclaims.*

Proof. The traditional target controls terminology. The source blocks primitive import. Carrier and classifier block identity smuggling. Stability blocks unusable behavior. Ledger blocks amnesic compression. The theorem list blocks empty naming. Exactness blocks weak correspondence becoming identity. Bridge status blocks host fusion. Status blocks grade inflation. □

Corollary 9462.2040 (Philosophical objects use the same template)**.** *Concepts such as freedom or consciousness use the same template if they are to become object-layer interfaces; otherwise they remain readings, gaps, sockets, or roadmap items.*

9462.30.14 Review discipline

Definition 9462.2041 (Review questions)**.** *Each claim is reviewed by asking for carrier, classifier, ledger, theory grade, formal status, bridge host if present, gap socket if present, forbidden readings, cannot-claim conflicts, and axis confusion.*

Definition 9462.2042 (Automatic downgrade)**.** *When a claim cannot answer a required audit question, it must be downgraded from theorem to proposition, proposition to schema, schema to roadmap, roadmap to gap, or gap to cannot-claim.*

Theorem 9462.2043 (Unauditable claims cannot be strengthened)**.** *A claim that cannot be audited cannot be strengthened into a theorem.*

Proof. A theorem requires a proof witness, and proof witnesses require carrier, classifier, ledger, scope, and status. If these cannot be given, theorem status is unavailable. □

9462.30.15 Synthesis

Theorem 9462.2044 (Audit methodology synthesis)**.** *A BEDC claim is admissible when it has a carrier, classifier, ledger, theory-closure grade, formal-verification grade, gap or refusal surface when incomplete, and no-leakage discipline.*

Proof. Writing items provide the audit surface. The two status axes remain independent. Closure status blocks bind chapter claims. Definitions, theorems, corollaries, and proofs have fixed fields. Gaps are typed. Cannot-claim entries and failure certificates record negative export surfaces. Bridges prevent host fusion. Tradition maps prevent philosophical overidentification. Smuggling checks block external primitive import. □

Corollary 9462.2045 (Audit-first writing principle)**.** *The primary question is not whether a concept may be mentioned, but by which carrier, classifier, ledger, gate, gap, and status it is allowed to be mentioned.*

Corollary 9462.2046 (Passage to final synthesis)**.** *With audit methodology fixed, the final layer can state the complete auditable-generativity theorem and research programme.*

Concretized in §1227: Audit writing discipline is realized as a finite NameCert packet separating carrier, classifier, ledger, closure status, formal status, refusal, registry, and no-leakage rows.

Concretized in §1333: AuditAxisSeparationUp realizes the audit-methodology two-axis noncollapse rule as a finite NameCert packet.

9462.31 Total Synthesis: Auditable Generativity

The preceding layers are gathered here into one philosophical description. BEDC begins from primitive distinction, carries generated information by histories, reads observation and time through continuation, licenses mature objects by naming certificates, exposes unclosed commitments through gaps and the far-end position, and governs every export by audit status.

9462.31.1 Total definition

Definition 9462.2047 (BEDC philosophy)**.** *BEDC philosophy is the structural theory of what can be generated, what can be named, what can be known, what must remain open, and what may be exported. Its basic terms are not substances, minds, worlds, or laws, but carriers, classifiers, witnesses, ledgers, certificates, gaps, bridges, and audit gates.*

Definition 9462.2048 (Auditable generativity)**.** *Auditable generativity is the thesis that every admissible object, knowledge claim, scientific claim, philosophical reading, or bridge claim must be generated, classified, ledgered, certified, or explicitly marked as a gap or cannot-claim entry.*

Definition 9462.2049 (Auditable generation chain)**.** *The auditable generation chain is the route*

$$\mathsf{b0}, \mathsf{b1} \to \mathsf{MarkSameness/NoConfusion} \to \mathsf{Hist} \to \mathsf{hsame} \to \mathsf{Ext/Cont} \to \mathsf{Ask/Sig/sameSig} \to \mathsf{Pkg/Gap} \to \mathsf{NameCert}$$

Completion-like, fold-like, geometric, algebraic, scientific, and philosophical interfaces enter through their own certificate schemas, not as kernel primitives.

Theorem 9462.2050 (Generation-chain priority)**.** *Every mature BEDC concept must occupy a position in the auditable generation chain.*

Proof. A concept without a generation-chain position has no carrier. Without a carrier it has no classifier. Without a classifier it has no auditable account of sameness, difference, transport, or exactness. Without a ledger, any compression or hidden material is untraceable. Without a certificate, the concept is not a licensed name. Hence mature concepts must be located in the chain. □

Corollary 9462.2051 (No position, no export)**.** *If a concept cannot state its carrier, classifier, ledger, and status, it cannot be exported as a BEDC object.*

Proof. The missing fields are precisely the fields by which the generation chain becomes auditable. The concept may remain a reading, gap, roadmap, or cannot-claim entry, but it is not an export. □

9462.31.2 Total ontology

Definition 9462.2052 (BEDC existence)**.** *At the object layer, X exists in BEDC when X has a licensed carrier, classifier, ledger, and certificate position. Mature existence is the conjunction of a naming certificate and the relevant reported status.*

Definition 9462.2053 (Licensed object)**.** *A licensed object is an interface whose naming certificate supplies source specification, pattern specification, classifier specification, stability certificate, and ledger policy.*

Theorem 9462.2054 (Mature existence is naming license)**.** *In the mature object layer, to exist is to have licensed naming status.*

Proof. A mature object cannot be introduced as a bare kernel primitive. It therefore enters through a $\mathsf{NameCert}$ and the associated status record. Conversely, a name with those fields is available as a licensed interface at its stated scope. Thus mature existence is naming license. □

Corollary 9462.2055 (Anti-substance ontology)**.** *BEDC does not read the world as a collection of substances, but as a stratified field of licensed generated interfaces.*

Proof. Substance ontology places objects before certificates. BEDC licenses objects only after generation, classification, ledgering, and naming. □

Corollary 9462.2056 (Mature objects are not inputs)**.** *Numbers, reals, sets, functions, limits, topological spaces, categories, universes, consciousness, causality, freedom, truth, and God are not BEDC primitives. They can enter only as certificates, records-side readings, bridge targets, gap positions, or refusal entries.*

Proof. Each item requires structure beyond the primitive distinction and history constructors. Treating any of them as an input bypasses carrier, classifier, ledger, certificate, or gap discipline. □

9462.31.3 Total epistemology

Definition 9462.2057 (BEDC knowledge)**.** *To know P in BEDC is to hold an auditable witness for P. Such a witness may be a sameness witness, continuation witness, classifier witness, package or gap witness, naming certificate, semantic naming certificate, derivation certificate, bridge witness, gap report, or failure certificate.*

Definition 9462.2058 (Knowledge strength)**.** *A knowledge claim reports its strength: seed, obligation, paper-level closure, checked theorem, checked bridge, gap, or cannot-claim status. The theory-closure and formal-verification axes remain distinct.*

Theorem 9462.2059 (Knowledge is not bare truth)**.** *BEDC knowledge is possession of a certificate-bearing witness, not possession of a bare internal truth predicate.*

Proof. An internal total truth predicate would close the meta loop. BEDC keeps the meta loop open. Hence knowledge cannot be defined as access to such a predicate; it is reported as local witness, certificate, ledger, bridge, or gap surface. □

Corollary 9462.2060 (To know is to audit)**.** *Within BEDC, to know is to be able to audit the witness path.*

Proof. Without an auditable path, the claim has no visible source, classifier, stability, hidden-material ledger, or strength. □

Concretized in §1306: Epistemic witnesses realize BEDC knowledge as finite carrier, audit-route, strength, gap/refusal, bridge, and NameCert rows.

9462.31.4 Total observation theory

Definition 9462.2061 (Observation)**.** *An observation is a continuation row:*

$$\mathsf{Observation}(h, d, h') := \mathsf{Cont}(h, d, h').$$

Definition 9462.2062 (Observer)**.** *An observer is a history, read modulo* hsame*:*

$$\mathsf{Observer}(h) := h : \mathsf{Hist}.$$

Definition 9462.2063 (Time)**.** *Time is the local continuation chain by which a history is generated from the empty history.*

Theorem 9462.2064 (Observation, distinction, and time)**.** *At the kernel layer, observation, distinction step, and time step have the same continuation shape.*

Proof. Observation is Cont on histories. A distinction step is the application of the two elementary extension shapes. Local time is the ordered growth of a history by those steps. These readings share the same carrier, continuation laws, and hsame-respecting behavior. □

Theorem 9462.2065 (Observer is history)**.** *BEDC has no observer-substance behind the history. The observer is the* Hist *value read under* hsame.

Proof. The continuation signature has no separate subject parameter. Identity is supplied by hsame on histories. Adding a hidden observer entity would require unlicensed transport between that entity and the visible history, which is not part of the kernel surface. □

Corollary 9462.2066 (Subject is not substance)**.** *The subject is a stable reading of the historical trace, not an owner behind it.*

Proof. There is no additional carrier for such an owner. Subject talk is therefore a reading of history, continuity, and self-proxy ledgers. □

9462.31.5 Total logic

Definition 9462.2067 (Allowed proof)**.** *An allowed BEDC proof is an explicit witness construction or a local refutation inside a fixed carrier, classifier, and certificate.*

Definition 9462.2068 (Forbidden proof)**.** *A forbidden proof extracts positive content without constructor witness, or closes a global truth, choice, quotient, propositional extensionality, or halting surface.*

Theorem 9462.2069 (Logic discipline synthesis)**.** *BEDC logic is governed by three rules: positive content must be constructed, negative content must remain local, and the meta loop must remain open.*

Proof. Positive claims require witnesses. Refutations are licensed only inside fixed certificate surfaces where the carrier, classifier, and closure laws are displayed. A proof principle that decides all certificates or all propositions would internalize the meta level and violate the open meta-loop boundary. □

Corollary 9462.2070 (Classical shortcuts share one refusal)**.** *Choice, quotient collapse, and propositional extensionality are refused as public BEDC resources because each accepts unwitnessed supply at a different level.*

Proof. Choice supplies a term, quotient collapse supplies an identity, and propositional extensionality supplies a rewrite principle without the transport witnesses BEDC requires at the public surface. □

Corollary 9462.2071 (Halting obstruction)**.** *A total halting predicate for all certificate-level continuations is the recursion-theoretic face of the same boundary.*

Proof. Such a predicate would decide all continuation behavior at the certificate stratum and thereby close the meta loop that BEDC keeps open. □

Concretized in §1294: The allowed/forbidden proof boundary is realized as the finite AllowedProofAuditUp NameCert packet.

9462.31.6 Total boundary theory

Definition 9462.2072 (Closed ground loop)**.** *The ground loop is closed by the primitive two-constructor distinction.*

Definition 9462.2073 (Open meta loop)**.** *The meta loop is open because BEDC does not internally define its own total truth predicate, runtime validator, or halting oracle.*

Theorem 9462.2074 (Two-loop synthesis)**.** *BEDC requires closed ground and open meta.*

Proof. Without a closed ground, the system cannot start. With a closed meta loop, the system would validate its own truth or runtime surface from inside, producing the familiar self-reference obstruction. The loops therefore have different statuses. □

Corollary 9462.2075 (Minimal ground contribution)**.** *BEDC does not bypass formal limits. It compresses the unavoidable ground loop to primitive distinction and refuses meta closure.*

Concretized in §1295: The observation, proof-discipline, and open-meta-loop rows are realized as one finite audit boundary.

9462.31.7 Total gap and far-end theory

Definition 9462.2076 (Gap)**.** *A gap is a site in a closed system where a requested supply cannot be discharged internally:*

$$\mathsf{GAP}(\mathsf{Site}, \mathsf{RequestedSupply}, \mathsf{SocketType}, \mathsf{Gate}).$$

Definition 9462.2077 (Far-end position)**.** *The far-end position T is the structural position of the supply requested by gap sockets. It is not an object-layer entity, axiom, or theorem.*

Theorem 9462.2078 (Gap and far-end synthesis)**.** *Every commitment beyond closed records and closed generation must be exposed as a gap, and the far end of such sockets is named T.*

Proof. Finite records cannot yield unrestricted universal closure, unwitnessed witnesses, external identity, future causal persistence, inter-history coordination, inscription novelty, or total self-reference closure. These demands must be reported as sockets rather than internal rules. □

Corollary 9462.2079 (Far end is not object)**.** *The far-end position can be named only positionally, not reified as an object.*

Proof. If T were an object, it would need a carrier, classifier, ledger, and naming certificate. Its role is precisely to mark a supply that the object layer cannot internalize. □

9462.31.8 Total phenomenology

Definition 9462.2080 (Conscious record)**.** *A conscious record is the focused records-side packet*

$$\mathsf{ConsciousRecord}(L_N, F) := (H(F), T(F), S(F), \rho).$$

Definition 9462.2081 (Universe slice)**.** $\mathsf{Universe}_{L_N}$ *is the finite observation-history slice admitted by the focused ledger L_N.*

Definition 9462.2082 (Space reading)**.** Space_{L_N} *is the locality-cell graph together with the inter-history invariant rows available at the focused ledger.*

Theorem 9462.2083 (Phenomenology synthesis)**.** *BEDC phenomenology reads consciousness as a focused records-side carrier, universe as a finite observation-history slice, time as a local continuation chain, space as inter-history locality geometry, identity as an* hsame*-stable trace, causality as a Type IV gap, freedom as a Type VI inscription-event aspect, other minds as Type V gaps, and self-reference as a Type VII gap-on-gap hierarchy.*

Proof. Observer is history and observation is continuation. Hence consciousness cannot enter as an independent substance. No global clock or global container is supplied by the kernel, so universe and space are read from finite slices and inter-history relations. Causality, freedom, other minds, and self-reference each request a supply beyond closed records, and so take socket form. □

Corollary 9462.2084 (Hard problem as audit question)**.** *BEDC does not ask for consciousness as an essence. It asks which records-side carrier, ledger, self-proxy, relation row, and gap sockets make consciousness-talk admissible.*

Proof. Only records, certificates, and socketed boundary reports can enter the object layer. Phenomenal excess that lacks those surfaces remains a gap or reading. □

9462.31.9 Total philosophy of science

Definition 9462.2085 (Scientific object)**.** *A scientific object is a certificate package containing a naming certificate, derivation or closure status, empirical ledger, inter-history invariant surface, gap report, and optional standard bridge.*

Definition 9462.2086 (Physical law)**.** *A physical law is an empirical regularity together with a Type IV gap for future persistence.*

Theorem 9462.2087 (Scientific object synthesis)**.** *Scientific objects are composites of records, invariants, certificates, empirical ledgers, bridges, and gap reports; they are not primitives.*

Proof. A scientific object names a pattern in records, classifies and stabilizes it across perspectives, connects it to measurement and error ledgers, may compare it with mature model systems, and often commits to future predictions. Each component carries its own audit obligation. □

Corollary 9462.2088 (Relational physics principle)**.** *Frame-independent physical content is inter-history invariant content.*

Proof. Multiple histories have no shared base history, no shared time-step indexing, and no global synchronization. What survives as physical content must therefore be carried by stable inter-history relations. □

Corollary 9462.2089 (Model is not world)**.** *A model is an interpretation, compression, or bridge for records and invariants. It is not the world itself.*

Proof. The universe available to BEDC is a finite observation-history slice. Identifying a model with the world would replace that slice by a global object. □

Concretized in §1300: The phenomenology/science interface is realized as a finite composite NameCert packet joining records-side consciousness, universe slices, inter-history invariants, empirical ledgers, gap reports, and conservative bridge rows.

9462.31.10 Total bridge theory

Definition 9462.2090 (Standard bridge)**.** *A standard bridge from N to a host theory T contains encoding, decoding, return witnesses in both directions, finite theorem preservation, and no-host-leak.*

Definition 9462.2091 (Conservative bridge)**.** *A bridge is conservative when it does not inject host primitives into the BEDC public surface.*

Theorem 9462.2092 (Bridge is not fusion)**.** *A standard bridge is conservative recognition, not fusion with the host theory.*

Proof. The bridge lets a host reader recognize a BEDC object and lets BEDC accept only the finitely listed translated theorem rows. No-host-leak blocks host primitives such as host natural numbers, quotient collapses, choice, propositional extensionality, and host equality from becoming BEDC inputs. □

Corollary 9462.2093 (Mature systems are bridge targets)**.** *Mature mathematics and mature physics may serve as bridge targets, but they cannot replace internal BEDC certificates.*

Proof. Replacing the certificate by the host theory would turn a bridge into primitive import and violate no-host-leak. □

9462.31.11 Total audit theory

Definition 9462.2094 (Admissible claim)**.** *A claim is admissible when it has carrier, classifier, ledger, status, certificate or gap or cannot-claim surface, and no-leakage discipline.*

Definition 9462.2095 (Failure certificate)**.** *A failure certificate is the negative certificate of a claim: it states which required field is missing, invalid, overstated, unchecked, or confused across the theory-closure and formal-verification axes.*

Theorem 9462.2096 (Positive and negative certificates)**.** *Certificates license export, failure certificates block export, and cannot-claim entries record refusal.*

Proof. The positive certificate fills source, classifier, exactness, ledger, stability, bridge, and status fields. The failure certificate names a missing or invalid field. The cannot-claim entry records that the manuscript does not make the adjacent claim at the current surface. □

Corollary 9462.2097 (Cannot-claim is negative knowledge)**.** *The cannot-claim registry is part of BEDC's philosophical structure, not a modesty convention.*

Proof. The registry records typed refusals and required upgrades. It is the negative face of the export surface. □

9462.31.12 Total system diagram

Definition 9462.2098 (Three-layer system)**.** *The BEDC philosophical system has three layers. The ground layer contains* $\mathsf{b0}, \mathsf{b1}$*, marks, histories,* msame*,* hsame*, extension, and continuation. The certificate layer contains asking, signatures, packages, gaps, naming certificates, semantic naming certificates, derivation certificates, and standard bridges. The boundary and interpretation layer contains gaps, the far-end position, conscious records, universe slices, scientific objects, tradition maps, and audit methods.*

Definition 9462.2099 (Generation, naming, boundary)**.** *The whole framework can be compressed into three questions: generation asks how a thing is produced, naming asks how it stabilizes as an object, and boundary asks why it cannot be closed internally.*

Theorem 9462.2100 (Three-question synthesis)**.** *Every BEDC philosophical question falls under generation, naming, or boundary.*

Proof. Objects, knowledge, subjectivity, time, consciousness, universe, science, and philosophical comparison all require a carrier, a classifier or naming certificate, and an account of any unclosed commitment. Those are exactly the generation, naming, and boundary questions. □

Corollary 9462.2101 (BEDC question form)**.** *Given a concept X, BEDC first asks for its carrier, classifier, ledger, naming certificate, gap surface, and status.*

9462.31.13 Auditable-generativity theorem

Theorem 9462.2102 (Auditable generativity)**.** *The complete BEDC philosophical framework is auditable generativity: distinction-grounded, history-carried, continuation-dynamic, name-certificate-licensed, ledger-tracked, gap-bounded, bridge-conservative, and audit-governed.*

Proof. Distinction precedes information, and histories are the minimal information carriers. Observation and time are read through continuation, and observer is history. Objects and knowledge require naming certificates and witnesses. Logic admits explicit construction and local refutation while refusing meta closure. Commitments beyond closed generation are exposed as gaps and far-end positions. Consciousness, universe, space, causality, freedom, other minds, and self-reference are rewritten as records-side carriers or sockets. Scientific objects require records, invariants, empirical ledgers, and bridges. Philosophical traditions enter only through structural maps. Audit methodology requires every claim to report carrier, classifier, ledger, status, and refusal surface. These results together give auditable generativity. □

Corollary 9462.2103 (One-sentence characterization)**.** *BEDC is an auditable generative system that starts from primitive distinction, generates histories and continuations, licenses names by certificates, keeps boundaries open through gaps and the far-end position, and governs export strength by audit gates.*

Corollary 9462.2104 (Negative characterization)**.** *BEDC is not traditional realism, subjectivist idealism, ordinary formalism, naive constructivism, psychologistic phenomenology, global physical realism, theological object theory, or a host-theory import system. It is distinction ontology, history generativity, observerless observation theory, certificate object theory, witness epistemology, open-meta boundary theory, gap and far-end transcendence theory, relational science theory, and audit writing theory.*

Corollary 9462.2105 (Research programme)**.** *Further work should proceed by the lowest open layer: finite kernel, finite data certificates, arithmetic certificates, algebra certificates, order and logic certificates, set-like and type-like interfaces, topology and metric completion, analysis, geometry, category theory, homology, number theory, spectra, and mature-system bridges.*

Concretized in §1258: Audit system packets realize admissible claims, positive certificates, failure certificates, cannot-claim rows, export grades, and ledger closure as one finite NameCert surface.

9462.31.14 Status record

Definition 9462.2106 (Philosophy package status)**.** *The philosophy package is a paper-level structural reading of the BEDC kernel and certificate architecture. It introduces no kernel primitive. It is closed as a scoped synthesis of primitive distinction, history, continuation, naming certificates, gap and far-end boundary theory, standard bridge discipline, and audit gates. It does not close full physical theory, empirical constants, complete arithmetic reconstruction, real completion, a global truth predicate, internal runtime validation, or phenomenal consciousness as object-level essence. It makes no whole-package Lean target claim; component targets belong to their cited kernel, naming-certificate, bridge, and audit chapters.*

Corollary 9462.2107 (Upgrade path)**.** *To promote the philosophy package from paper-level synthesis to a formal package, one would define a formal structure for philosophy maps, encode tradition maps, gap reports, and writing items, check no-host-leak and no-meta-closure dependencies, and give a finite theorem list for each mapped concept.*

Concretized in §1296: Audit export firewalls realize total-audit synthesis as finite claim, certificate, failure, refusal, export-grade, firewall, ledger, and NameCert rows.

Concretized in §1337: Audit finality certificates realize total auditable generativity as finite claim, certificate, failure, refusal, export-grade, synthesis, firewall, ledger, and NameCert rows.

Concretized in §1339: Epistemic ontology bridges realize philosophy total synthesis as finite ontology, epistemology, observation-logic, phenomenology-science, audit-export, ledger, and NameCert rows.

9462.32 Research Programme and Implementation Route

The philosophical synthesis becomes a research programme only when its objects are pursued as certificates rather than imported as mature assumptions. The route is layered, strength-marked, phase-governed, failure-aware, and audit-facing. It extends BEDC by proving that names have earned use, not by asserting that mature objects are already available.

9462.32.1 Definition of the research programme

Definition 9462.2108 (BEDC research programme)**.** *The BEDC research programme starts from the finite kernel*

$$\mathsf{b0}, \mathsf{b1} \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{NameCert},$$

and, for each mature name N, constructs the five-field certificate consisting of source specification, pattern specification, classifier specification, stability certificate, and ledger policy. The resulting name is then assigned theory closure, formal verification, bridge, and refusal status.

Definition 9462.2109 (Research target)**.** *The target of research is not to collect standard objects as assumptions. It is to display the exact generative behavior by which a standard name earns a licensed place in BEDC.*

Theorem 9462.2110 (Research is not expansionism)**.** *The BEDC research programme is not the outward expansion of an object library, but the inward completion of eligibility proofs for names.*

Proof. If the programme were expansionist, it could import mature objects such as numbers, reals, sets, functions, topologies, and categories as assumptions. The roadmap instead treats mature constructions as licensed names and treats host naturals, host lists, definitional equality, quotient collapse, choice, and propositional extensionality as contamination when exposed on the public BEDC surface. Hence the programme proves license rather than importing objects. □

Concretized in §1309: Name eligibility proofs are realized as finite source, classifier, strength, audit, refusal, transport, replay, provenance, and NameCert packets.

Corollary 9462.2111 (The object of research is the certificate)**.** *When BEDC studies* $\mathsf{Real}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, *or* $\mathsf{Zeta}^{\uparrow}$, *it studies their naming certificates, classifiers, ledgers, stability surfaces, bridges, and failure surfaces.*

Proof. The mature name is admitted only through the certificate. Without that certificate, the mature object remains an external target or roadmap name. □

9462.32.2 Strength gradient

Definition 9462.2112 (Research strength)**.** *Each derived mature-object claim lies on a strength gradient:*

$$\mathsf{naming\ seed} < \mathsf{paper\ certificate} < \mathsf{machine\ checked\ certificate} < \mathsf{conservative\ bridge}.$$

Lower strength may not be reported as higher strength.

Definition 9462.2113 (Four strengths)**.** *A naming seed proposes a paper-side carrier and classifier while obligations remain open. A paper certificate supplies the five naming certificate fields and a prose proof. A machine-checked certificate formalizes the relevant certificate fields and stability obligations in the Lean development and resolves the cited target markers. A conservative bridge gives an external two-way translation to a mature standard object while preserving no-host-leak.*

Theorem 9462.2114 (No strength inflation)**.** *If N has current strength s, BEDC cannot claim for N any strength strictly above s.*

Proof. Each higher strength adds an undischarged obligation: full certificate fields above seed, machine verification above paper certificate, and host-relative conservative translation above machine checking. Reporting a higher strength would present missing obligations as complete. □

Corollary 9462.2115 (Philosophical claims report strength)**.** *Claims about consciousness, freedom, universe, and physical law must be reported as structural readings, paper certificates, schemas, gap sockets, machine targets, or bridge targets, rather than as unqualified proofs.*

Proof. Philosophical claims also carry naming, classification, stability, ledger, and boundary obligations. Without strength reporting, readings, schemas, gaps, and theorems are confused. □

9462.32.3 Layer route

Definition 9462.2116 (Layer route)**.** *The BEDC reconstruction route is layered:*

$$L0\ \mathsf{Marks},\quad L1\ \mathsf{Histories},\quad L2\ \mathsf{GenerativeDynamics},\quad L3\ \mathsf{AskingAndSignature},$$
$$L4\ \mathsf{PackageRelationsAndGaps},\quad L5\ \mathsf{NamingAndStability},\quad L6\ \mathsf{FiniteData},\quad L7\ \mathsf{Arithmetic},$$
$$L8\ \mathsf{Algebra},\quad L9\ \mathsf{OrderLogicSetLikeTypeLike},\quad L10\ \mathsf{TopologyMetricCompletionReal},$$
$$L11\ \mathsf{Analysis},\quad L12\ \mathsf{GeometryCategoryHomology},\quad L13\ \mathsf{NumberTheoryAndSpectra},\quad L14\ \mathsf{MatureSystemBridge}.$$

Each layer may depend only on lower layers.

Definition 9462.2117 (Layer dependency)**.** $\mathsf{Depends}(L_i, L_j)$ *is licensed only when* $i > j$*. Cyclic dependence between layers is not part of the route.*

Theorem 9462.2118 (Lowest open layer first)**.** *The research route advances the lowest open layer first.*

Proof. A high-layer claim imports all unclosed lower-layer obligations into its vocabulary. For example, analytic or spectral claims hide completion, algebraic descent, order, bridge, and continuity obligations unless the lower layers are already closed. The layer index therefore functions as review priority. □

Corollary 9462.2119 (Over-layer work is roadmap)**.** *If a high-layer target depends on an open lower layer, the high-layer target is a roadmap, schema, proof target, deferred claim, or gap, not a closed theorem.*

Proof. The missing lower layer withholds source, classifier, stability, exactness, or bridge data needed by the high-layer certificate. □

Concretized in §1305: Programme strength ledgers realize the seed, paper-certificate, machine-check, bridge, dependency, refusal, and no-strength-inflation rows as one finite NameCert packet.

Concretized in §1314: The research-programme strength gradient is realized as a finite ledger that blocks over-reporting across naming seed, paper certificate, machine-checked certificate, and conservative bridge.

9462.32.4 Finite kernel baseline

Definition 9462.2120 (Finite kernel baseline)**.** *The finite kernel baseline contains marks, histories, extension, continuation, asking, signatures, signature sameness, packages, gaps, composite gaps, and naming certificates, with dependency shape*

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{CompGap} \to \mathsf{NameCert}.$$

Definition 9462.2121 (Kernel exit condition)**.** *The L0–L5 exit condition is that the kernel theorems are checked at their cited targets and remain free of axiom-purity contamination. The surface includes sameness equivalences and no-confusion, extension and continuation laws, asking and signature determinacy under policies, package and gap soundness, coverage, and separation, and naming certificate admission.*

Theorem 9462.2122 (Finite kernel precedes mature objects)**.** *Every mature-object certificate depends on the finite kernel baseline.*

Proof. NameCert is the universal admission rule. It depends on package and gap structure, which depends on signatures, which depends on continuation, histories, and marks. Without this baseline, no mature name has a proof boundary. □

Concretized in §1288: Kernel phase-refusal packets realize the philosophy programme phase-order and off-table boundary as finite BHist, hsame, Cont, Pkg, and NameCert rows.

Corollary 9462.2123 (The philosophy package depends on L0–L5)**.** *Any formalization of the philosophy package must return to L0–L5: distinction to marks, history to* Hist*, observation and time to* Cont*, object license to* NameCert*, and boundary memory to* Gap*.*

9462.32.5 Phase bundles

Definition 9462.2124 (Phase bundle)**.** *A phase bundle is the collection of objects sanctioned for priority during a research period, together with a precise exit criterion.*

Definition 9462.2125 (Near-term phases)**.** *The near-term route is organized as follows: kernel-checked baseline; finite data from booleans through lists; full arithmetic through rationals with induction, recursion, ordered-ring, and ordered-field structure; algebra up to fields; real and completion through dyadic rational core, stream names, regular rational sequences, and real names; real algebra and geometry; compactness and continuity; linear algebra and analysis seeds; categories and bridges for already checked objects.*

Theorem 9462.2126 (No phase skipping)**.** *While a phase has not exited, targets belonging to later phases are deferred.*

Proof. The phase determines the current sanctioned advancement direction. Later-phase targets depend on obligations the current phase is meant to discharge. Advancing them as closed would hide that dependency. □

Corollary 9462.2127 (Dependency before attraction)**.** *High-profile goals such as the Riemann Hypothesis, quantum states, curved spacetime, or spectral theorems cannot bypass lower certificates.*

Proof. Such goals depend on real numbers, analysis, topology, algebra, bridges, or multi-history semantics. Until those dependencies are closed, the goals remain roadmaps or proof targets. □

9462.32.6 Off-the-table claims

Definition 9462.2128 (Off-the-table claim)**.** *An off-the-table claim is a claim explicitly not made until its named milestone holds at the required strength.*

Definition 9462.2129 (Current refusal list)**.** *The current refusal surface includes claims that BEDC has re-derived all mature mathematics; that real numbers, the unit circle, or folds are complete; that a natural-number seed is full natural-number theory; that an addition seed is full addition; that paper certificates are machine-checked certificates; that scaffolds are verified implementations; that external models replace BEDC proof obligations; or that host equality, host natural numbers, or host functions belong to the BEDC public surface.*

Theorem 9462.2130 (Off-the-table is a boundary condition)**.** *The off-the-table list is a boundary condition of the research programme, not a rhetorical caution.*

Proof. Without explicit refusals, seeds, paper certificates, scaffolds, and external models could be misread as mature exports. The refusal list prevents that inference and records which claim remains unsupported. □

Corollary 9462.2131 (Philosophical synthesis must not overclaim)**.** *The philosophy package cannot claim that BEDC has completed all mathematics, proved physics, internalized consciousness as essence, or supplied a global truth predicate. It can claim a certificate and audit discipline for any such target to enter.*

9462.32.7 Verification reporting

Definition 9462.2132 (Checked report)**.** *A checked formalization report states the proof-checker version, the exact command used, whether external libraries are imported, the names of the theorems successfully checked, the admitted or pending statements, and whether public APIs expose host equality, host natural numbers, host lists, or host functions as BEDC internals.*

Definition 9462.2133 (Lean target status)**.** *Lean target status is reported theorem by theorem through inline markers for checked theorem, definition, statement-only target, or temporary unclosed target. These markers report verification status; they do not by themselves settle theory closure.*

Definition 9462.2134 (Audit report)**.** *An audit report records marker resolution and axiom purity: checked targets must resolve to real declarations in the Lean source, and the axiom audit must report no prohibited axiom contamination.*

Theorem 9462.2135 (Verification report is not object theory)**.** *Verification reporting belongs to project and implementation status; it does not replace BEDC object theory.*

Proof. The report says which commands, markers, and declarations are clean. Object theory requires carrier, classifier, ledger, certificate, closure, and bridge fields. A clean report can support those fields but does not constitute them. □

Corollary 9462.2136 (Implementation scaffold is external)**.** *An implementation scaffold is an external artifact unless and until its claims are tied to the relevant BEDC certificates and audit gates.*

9462.32.8 Failure and downgrade

Definition 9462.2137 (Failure certificate at strength)**.** *A failure certificate for a name N at strength s is a typed reason why the claimed gate fails: missing source, missing classifier, classifier failure, exactness failure, operation descent failure, absent gap ledger, unstable completion, failed bridge, host primitive leakage, or checker failure.*

Theorem 9462.2138 (Failure blocks gate)**.** *If a failure certificate is exhibited for N at strength s, the corresponding acceptance gate for N at s does not hold.*

Proof. Every failure branch denies a field required by the gate. The gate therefore cannot be satisfied at that strength. □

Definition 9462.2139 (Downgrade)**.** *Downgrade is the required lowering of a reported status when a failure certificate shows that the stronger marker or claim is not licensed.*

Corollary 9462.2140 (Failure is a result)**.** *A failure certificate is structural information: it identifies the field that blocks promotion.*

Proof. The branch says whether source, classifier, exactness, ledger, bridge, host-leak, checker, or another field fails. That is a precise research outcome and a guide for the next obligation or refusal. □

Concretized in §1280: Status downgrades are realized as finite prior-status, failure-certificate, lowered-report, gap-route, provenance, and NameCert packets.

Concretized in §1289: VerificationFailureRoadmapUp realizes checked reports, audit rows, failure certificates, downgrade routes, roadmap boundaries, and cannot-claim rows as one NameCert packet.

9462.32.9 Roadmap and cannot-claim

Definition 9462.2141 (Roadmap)**.** *A roadmap states what may become claimable after named blockers are cleared.*

Definition 9462.2142 (Cannot-claim registry)**.** *A cannot-claim registry states what the manuscript currently refuses to assert.*

Theorem 9462.2143 (Roadmap does not license theorem status)**.** *The occurrence of a target in a roadmap does not license its statement as a theorem.*

Proof. A roadmap gives dependency order and upgrade conditions. Without a certificate, witness, bridge, or checked target at the required strength, the target remains unproved. □

Corollary 9462.2144 (Roadmap use in philosophy)**.** *The philosophy package may name future formal structures such as philosophy maps, tradition maps, gap reports, conscious-record certificates, and multi-history physics chains, but those names do not thereby become formalized.*

9462.32.10 Multi-history physics route

Definition 9462.2145 (Multi-history physics chain)**.** *The multi-history physics route proceeds through naming certificates for multi-history configuration, cross-history causal dependence, maximum causal rate, Lorentz-style geometry, curved spacetime as Lorentz data with gluing, and quantum-state structure as a multi-history superposition interface.*

Concretized in §1290: finite causal intervention NameCert

Concretized in §1291: Multi-history causal dependence is realized as a finite no-global-frame causal-route packet.

Concretized in §1298: The multi-history physics route is realized as a composite finite NameCert packet over relational physics, physical-law bridge, scientific-model bridge, and quantum-state rows with explicit falsification boundaries.

Definition 9462.2146 (Physics falsification surface)**.** *Each physical step carries a falsification surface. Multi-history configuration fails if it requires a global frame. Cross-history causal dependence fails if it cannot be expressed as a stable predicate. Maximum causal rate fails if it requires choice. Lorentz-style geometry fails if its distance shape is not symmetric or invariant. Curved spacetime fails if gluing cannot be constructive. Quantum-state structure fails if the superposition predicate forces propositional extensionality.*

Concretized in §1293: Physics falsification surfaces are realized as finite physical-route, law, empirical-persistence, countermodel, bridge-audit, axiom-purity, typed-gap, transport, provenance, and NameCert rows.

Theorem 9462.2147 (Physics failure is structural)**.** *If the multi-history physics route cannot remain inside the axiom-purity gate, the failure point is itself a BEDC structural result.*

Proof. The route asks whether physical relational semantics can be closed without choice, quotient collapse, or propositional extensionality. A failure identifies the first schema where distinction, history, continuation, and certificate discipline are insufficient. That location is structural information. □

Corollary 9462.2148 (Physics route does not presume success)**.** *The physics route should be written as well-posed interface, falsification surface, and axiom-purity test, not as a completed derivation of relativity or quantum mechanics.*

Proof. The route currently licenses a structural form and dependency chain. It does not supply empirical values or completed physical theories. □

9462.32.11 Research programme synthesis

Theorem 9462.2149 (Research programme synthesis)**.** *The BEDC research programme is layered, certified, strength-marked, phase-governed, failure-aware, boundary-honest, audit-facing, and physics-aware.*

Concretized in §1326: The research-programme synthesis is realized as a composite finite NameCert packet over physical-route, falsification, programme-strength, verification-failure, and failure-certificate rows.

Proof. The programme is layered by the L0–L14 dependency route. It is certified because each mature name requires a naming certificate. It is strength-marked by the seed, paper, machine-checked, and conservative bridge gradient. It is phase-governed because only the current open phase is sanctioned. It is failure-aware because failure certificates block promotion while identifying missing fields. It is boundary-honest because cannot-claim entries record current refusals. It is audit-facing because reports must state checker, command, checked names, pending statements, host leakage, and axiom status. It is physics-aware because multi-history physics is treated as a roadmap with falsification surfaces rather than a completed theory. □

Corollary 9462.2150 (After philosophy)**.** *After the philosophical synthesis, the priority is not to add more interpretive prose but to turn philosophical concepts into writing items, gap reports, tradition maps, formal targets, lowest-layer naming certificates, failure certificates, and cannot-claim entries.*

Proof. The interpretive frame is already present. Progress comes from moving readings into certificate and audit discipline at the lowest open layer. □

Corollary 9462.2151 (Execution maxim)**.** *Do not extend BEDC by assertion. Extend it by certificate.*

9462.32.12 Programme summary

Definition 9462.2152 (Programme compression)**.** *The future of BEDC is not the claim that mature objects are already possessed. It is the progressive discharge of naming certificates, ledgers, stability obligations, bridges, failure surfaces, and audit reports by which each name proves its right to be used.*

Concretized in §1328: Research-programme execution gates realize lowest-open-layer readiness, strength, verification, bridge, refusal, transport, replay, provenance, and NameCert rows as one finite packet.

Concretized in §1324: Review acceptance gates realize operator approval as finite proposal, decision, refusal, audit-readback, and NameCert rows.

Concretized in §1327: The Real-and-completion exit certificate is realized as a finite four-face NameCert packet over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, and RealUp.

Concretized in §1322: Phase-governed certificate routing is realized as a finite NameCert packet with phase, strength, refusal, failure-audit, dependency, transport, provenance, and local naming rows.

Concretized in §1331: The research programme route is realized as a composite NameCert packet binding phase, strength, refusal, verification-failure, export, and review rows.

Concretized in §1332: Research-programme phase governance, proposal review, failure feedback, calibration, and audit-loop rows are realized as one finite NameCert packet.

9462.33 Formal Philosophy Package, Export Interface, and Governance

The philosophical framework can be treated as a managed paper-level package. This package does not extend the kernel. It organizes how BEDC structures are read, registered, mapped to traditions and scientific claims, exported, revised, and refused when a claim exceeds its certificate surface.

9462.33.1 Object of the philosophy package

Definition 9462.2153 (BEDC philosophy package)**.** *A BEDC philosophy package is a paper-level structure containing core thesis, concept registry, theorem map, gap map, tradition map, scientific map, cannot-claim registry, closure status, and amendment policy. Its core thesis is auditable generativity; its registries and maps record how philosophical language is attached to BEDC carriers, classifiers, ledgers, gaps, bridges, and audit gates.*

Definition 9462.2154 (Not a kernel extension)**.** *The philosophy package is not a kernel primitive, axiom, object theory, proof checker, or truth predicate. It is reading discipline, export discipline, mapping discipline, and audit discipline.*

Theorem 9462.2155 (No primitive added)**.** *The BEDC philosophy package adds no primitive to the BEDC kernel.*

Proof. All fields of the package refer to existing BEDC surfaces: marks, histories, continuation, history sameness, naming certificates, gaps, standard bridges, failure certificates, and cannot-claim entries. The package organizes and reports these surfaces; it does not introduce a constructor in the object language. □

Corollary 9462.2156 (Philosophy cannot overrule the kernel)**.** *Any philosophical reading that conflicts with the kernel signature, naming-certificate discipline, gap discipline, or axiom-purity gate is refused or downgraded.*

Proof. The package has only reading and export authority. It cannot create a new subject entity, global truth predicate, host equality, or unwitnessed existence claim. □

9462.33.2 Concept registry

Definition 9462.2157 (Concept entry)**.** *A concept entry for X records name, BEDC carrier, classifier, ledger, positive reading, forbidden readings, gap status, theory closure, and formal status.*

Definition 9462.2158 (Core concept registry)**.** *The core registry contains at least distinction, history, information, observation, observer, time, object, knowledge, truth boundary, gap, far-end position, consciousness, universe, space, causality, freedom, other minds, scientific object, physical law, standard bridge, tradition mapping, and audit method.*

Definition 9462.2159 (Observer entry)**.** *The observer entry has name observer, carrier* Hist*, classifier* hsame *with the licensed observational classifier, ledger the observer-history identity ledger, positive reading observer equals history, forbidden readings subject-substance, hidden self-entity, and global viewer, no gap for records-side observer identity, possible gap for phenomenal excess, paper-level structural closure, and component formal status only.*

Definition 9462.2160 (Far-end entry)**.** *The far-end entry has no object-level carrier, no internal classifier, gap-socket reports as ledger, positive reading as the far end of forward-binding supply, forbidden readings object-level deity, theorem-deity, substrate axiom, and intervention entity, boundary status as its gap status, apophatic theory reading, and no object-level formal target.*

Theorem 9462.2161 (Core concepts require registry entries)**.** *Every core concept entering BEDC philosophy prose requires a concept entry.*

Proof. Without the entry, the reader cannot tell carrier, classifier, ledger, boundary status, or forbidden readings. Terms such as universe, consciousness, and far-end position are especially vulnerable to primitive import without those fields. □

Corollary 9462.2162 (Unregistered concepts are helpers)**.** *An unregistered concept may serve as explanatory prose, but not as a BEDC philosophical object.*

Concretized in §1292: The BEDC philosophy package is realized as a finite NameCert carrier for registry, map, gap, audit, cannot-claim, closure-status, and export-policy rows.

Concretized in §1287: Concept registry obligations are realized as finite carrier, classifier, ledger, forbidden-reading, status, export, transport, route, provenance, and NameCert rows.

9462.33.3 Theorem map

Definition 9462.2163 (Theorem map entry)**.** *A theorem map entry records philosophical theorem, BEDC dependency, proof mode, claim strength, and not-claimed surface.*

Definition 9462.2164 (Core theorem map)**.** *The core theorem map includes: observer equals history, depending on the continuation signature, history sameness, and absence of a subject parameter; observation equals continuation, depending on continuation over histories; time as history growth, depending on elementary history steps and local continuation chains; object as licensed name, depending on naming-certificate discipline; knowledge as witness, depending on classifier, certificate, and ledger witnesses; gap and far-end boundary, depending on closed observational systems and forward-binding supply; scientific object as certified records, invariant, bridge, and gap; and tradition mapping as structural adjacency.*

Theorem 9462.2165 (Philosophical theorems depend on BEDC sites)**.** *A philosophical theorem in the package must depend on a BEDC theorem, definition, or certificate site.*

Proof. The philosophy package is not a primitive source of theorems. Without a BEDC dependency, a philosophical theorem would be an external doctrine rather than a BEDC reading. □

Corollary 9462.2166 (Theorem-map strength is bounded)**.** *A philosophical theorem cannot be stronger than the status of the BEDC sites on which it depends.*

Proof. If a dependency is schema-only, statement-only, or roadmap-level, the mapped philosophical theorem inherits that limitation. □

9462.33.4 Gap map

Definition 9462.2167 (Gap map)**.** *A gap map is a finite table of entries. Each entry records name, site, requested supply, socket type, gate, current status, upgrade path, and forbidden internalization.*

Definition 9462.2168 (Core philosophical gaps)**.** *The core gap map contains phenomenal-excess gap, induction gap, causality gap, other-minds gap, freedom gap, self-reference gap, physics gap, and bridge gap. These mark, respectively, the failure of records-side conscious records to define phenomenal essence, finite observation to imply unrestricted universal closure, observed regularity to imply future persistence, external behavior to yield another conscious system, derivation to erase inscription actuality, self-description to become a truth predicate, multi-history physics to close its witnesses, and host recognition to become checked no-host-leak bridge.*

Theorem 9462.2169 (Gap map prevents closure illusion)**.** *The gap map prevents philosophical prose from presenting boundary questions as closed theorems.*

Proof. Consciousness, causality, other minds, freedom, self-reference, and physics invite overclosed prose. A typed gap entry records the missing supply and the gate that blocks internalization. □

Corollary 9462.2170 (Gap map is necessary)**.** *A philosophy package without a gap map cannot be exported.*

9462.33.5 Tradition map

Definition 9462.2171 (Tradition map entry)**.** *A tradition map entry records tradition, traditional claim, BEDC site, preserved structure, rejected surplus, and audit status.*

Definition 9462.2172 (Core tradition map)**.** *The core tradition map registers Spencer-Brown at primitive distinction, Brouwer at history growth and elementary time step, bundle theory, no-self, and second-order cybernetics at observer equals history, Kant at certificate and gap conditions of possibility, Husserl at focused records-side stabilization and naming, Wittgenstein at certified use, Whitehead at continuation as universal verb, Tarski, Godel, and Muenchhausen at the open meta loop, and relational physics at inter-history invariants.*

Theorem 9462.2173 (Tradition map is not membership)**.** *The tradition map does not state that BEDC belongs to any mapped tradition.*

Proof. Each entry contains preserved structure and rejected surplus. If the map were doctrinal membership, rejected surplus would not be a required field. □

Corollary 9462.2174 (Tradition comparisons are retractable)**.** *If a comparison imports excessive metaphysics, the entry is removed or downgraded without affecting the kernel.*

Proof. Tradition mapping is a reading layer. Removing it does not alter Hist, Cont, or NameCert. □

9462.33.6 Scientific map

Definition 9462.2175 (Scientific map entry)**.** *A scientific map entry records name, records carrier, invariant carrier, empirical ledger, prediction gap, bridge status, claim status, and failure surface.*

Definition 9462.2176 (Core scientific map)**.** *The core scientific map records physical records as history, measurement, and continuation rows; physical laws as empirical regularities plus Type IV prediction gaps; space as locality-cell relation geometry rather than spacetime substrate; maximum causal rate as schema-only inter-history invariant until cross-history causality closes; light-speed-like constant as structural skeleton without empirical value; and Lorentz geometry, curved spacetime, and quantum state as roadmap naming-certificate targets.*

Theorem 9462.2177 (Scientific map separates structure, model, and value)**.** *Scientific entries must distinguish structure form, model bridge, empirical value, and prediction commitment.*

Proof. A maximum-rate form may be structural, a light-speed number must come from measurement ledger, a Lorentz geometry is a model or certificate target, and future persistence is a Type IV gap. Confusing these fields inflates schemas into empirical theorems. □

Corollary 9462.2178 (Physical claims default downward)**.** *Without full naming certificate, invariant surface, empirical ledger, and bridge, a physical claim remains schema, roadmap, gap, or failure-surface statement.*

Concretized in §1307: Package map boundaries are realized as a finite four-map, axis-boundary, audit-route, transport, replay, provenance, and local NameCert packet.

Concretized in §1308: The philosophy gap map is realized as a finite NameCert ledger for typed gap entries, socket kind, gate, status, upgrade route, and forbidden internalization.

9462.33.7 Philosophy cannot-claim registry

Definition 9462.2179 (Philosophy cannot-claim registry)**.** *The philosophy package cannot-claim registry includes refusals that BEDC has completed all mathematical reconstruction, proved a complete physical theory, defined phenomenal consciousness as essence, internalized the far end as an object-level entity, internalized its own truth predicate, supplied a global clock or God's-eye observer, used host equality or host naturals as public surface, made doctrinal identity claims about traditions, equated roadmap with theorem, or equated paper synthesis with a checked formal package.*

Concretized in §1318: Cannot-claim registries are realized as finite claim-text, status, refusal-reason, upgrade-boundary, blocking, transport, and NameCert packets.

Definition 9462.2180 (Registry entry format)**.** *Each cannot-claim entry records claim text, status, why it is not claimable, and required upgrade. The status is forbidden, deferred, or partial.*

Theorem 9462.2181 (Cannot-claim registry is required)**.** *Without a philosophy cannot-claim registry, the philosophy package cannot be exported.*

Proof. Philosophical language easily turns structural readings into completed metaphysical claims. The registry explicitly blocks those exports and records the missing upgrade or principled refusal. □

Corollary 9462.2182 (Refusal is content)**.** *In the philosophy package, a refusal supported by reason and upgrade condition is typed negative knowledge.*

9462.33.8 Amendment policy

Definition 9462.2183 (Philosophy package amendment)**.** *A philosophy package amendment is an in-place change to the concept registry, theorem map, gap map, tradition map, scientific map, cannot-claim registry, or closure status.*

Definition 9462.2184 (In-place amendment)**.** *Amendments are made in the thematic chapter whose subject they advance. Replaced phrasing is removed when the active formulation takes its place; parallel active readings are not retained.*

Definition 9462.2185 (Active label)**.** *When an amendment changes the active reading of an object, the active label remains the citation target, and definition or theorem labels attach to the active formulation.*

Theorem 9462.2186 (Amendment preserves audit continuity)**.** *An amendment must preserve audit continuity by identifying the affected registry entry, claim strength, gap classification, cannot-claim status, and tradition-map status.*

Proof. In-place replacement keeps the theory source of truth clean, but it can otherwise obscure whether a claim was strengthened, weakened, reclassified, or refused. Audit continuity records the status effect of the replacement. □

Corollary 9462.2187 (No stacked interpretations)**.** *Inactive philosophical readings should not remain as parallel active versions.*

Proof. Parallel active readings would make concept and theorem map entries non-unique. □

9462.33.9 Capstone policy

Definition 9462.2188 (Philosophy capstone)**.** *A philosophy capstone is a unifying claim spanning at least three registered domains.*

Definition 9462.2189 (Capstone is not commentary)**.** *A capstone provides structural unification rather than ordinary summary or meta-commentary.*

Theorem 9462.2190 (Capstones require dependencies)**.** *A philosophy capstone must list its cross-domain dependencies.*

Proof. The point of a capstone is structural unity. Without the domains, carriers, classifiers, ledgers, and gaps it spans, the claim cannot be audited as a capstone. □

Corollary 9462.2191 (Observer-history capstone)**.** *The current core philosophy capstone is the observer-history identity: observer equals history, observation equals continuation, time equals history growth, and no observer-substance is present. Its dependencies are* Hist*,* hsame*,* Cont*, elementary history steps, absence of a subject parameter, and the axiom-purity gate.*

9462.33.10 Export interface

Definition 9462.2192 (Philosophy export)**.** *A philosophical claim exports when it has a concept entry, a theorem map, gap map, or tradition map entry, a recorded closure status, and no active cannot-claim conflict.*

Definition 9462.2193 (Export grade)**.** *Export grades are reading export, paper synthesis export, component checked export, formal package target, and formal package checked.*

Theorem 9462.2194 (Current export grade)**.** *The present philosophy framework has paper synthesis export status, not formal package checked status.*

Proof. The preceding parts supply organized concepts, theorem readings, gaps, tradition maps, scientific maps, research route, and audit method. They do not supply checked Lean structures for the whole philosophy package or a whole-package checked target. □

Corollary 9462.2195 (Upgrade path for export)**.** *To reach formal package target status, the project must define structures for concept entries, theorem map entries, gap map entries, tradition map entries, scientific map entries, cannot-claim entries, and the philosophy package, together with well-formedness, no-smuggling, status monotonicity, cannot-claim consistency, and gap classification predicates.*

Concretized in §1319: Cannot-claim export gates realize the philosophy-package refusal registry and export-grade interface as finite NameCert rows blocking refused claims from checked-target export.

Concretized in §1321: Package-governance export packets realize cannot-claim registry, in-place governance, capstone dependency, export grade, refusal, transport, provenance, and local NameCert rows.

Concretized in §1320: Governance and export discipline is realized as finite registry, refusal, active-label, capstone, export-grade, transport, route, provenance, and NameCert rows.

9462.33.11 Formal target draft

Definition 9462.2196 (Philosophy map target)**.** *A future philosophy map structure would contain concept registry, theorem map, gap map, tradition map, scientific map, cannot-claim registry, closure status, and a no-smuggling field.*

Concretized in §1277: Philosophy map targets are realized as finite registry-map, gap-map, cannot-claim, closure-status, no-smuggling, and audit NameCert rows.

Definition 9462.2197 (Well-formed philosophy map)**.** *A well-formed philosophy map requires each registered concept to have a carrier or boundary position, a classifier or explanation of its absence, a ledger or explanation of its absence, forbidden readings when ambiguity is present, recorded status, and no cannot-claim conflict.*

Definition 9462.2198 (No smuggling)**.** *No smuggling means no subject-substance, global clock, internal truth predicate, host natural-number leak, host equality leak, classical choice, quotient soundness, propositional extensionality, or object-level far-end entity.*

Concretized in §1282: Philosophy formal-target ledgers realize the package formal-target draft as a finite NameCert packet over claim entries, target skeletons, no-smuggling checks, closure status, refusal boundaries, transport, provenance, and local naming.

Theorem 9462.2199 (Formal package target criterion)**.** *A philosophy map satisfying well-formedness and no-smuggling is a candidate formal package target.*

Proof. Well-formedness supplies the registry-level carrier, classifier, ledger, status, and conflict checks. No-smuggling excludes the philosophical, mathematical, and host imports that would exceed the package authority. Together they give the minimum target surface. □

Corollary 9462.2200 (Formal package does not solve philosophy)**.** *Even a checked philosophy map would prove only internal well-formedness and no-smuggling under declared predicates. It would not solve every philosophical question.*

Proof. The check is structural. It does not create a global truth predicate or close the meta loop. □

9462.33.12 Package synthesis

Theorem 9462.2201 (Philosophy package synthesis)**.** *The BEDC philosophy framework can be organized as a* BEDCPhilosophyPackage *that adds no kernel primitive and is licensed by concept registry, theorem map, gap map, tradition map, scientific map, cannot-claim registry, amendment policy, capstone policy, and export-grade discipline.*

Proof. The concept registry gives each concept carrier, classifier, ledger, and forbidden readings. The theorem map attaches philosophical theorems to BEDC sites. The gap map types unclosed boundaries. The tradition map records preserved structure and rejected surplus. The scientific map separates records, invariants, bridge, empirical value, and prediction gap. The cannot-claim registry blocks overexport. Amendment policy preserves the theory source of truth by in-place replacement. Capstone policy requires cross-domain structural dependence. Export-grade discipline distinguishes paper synthesis from formal package target and formal package checked status. □

Corollary 9462.2202 (Current package status)**.** *The current status is paper synthesis export, unformalized as a whole package, with candidate formal targets identified, no external philosophical bridge claimed, and an active cannot-claim registry.*

Corollary 9462.2203 (Minimum next work)**.** *The minimum next work is to build explicit concept, theorem, gap, tradition, scientific, and cannot-claim tables, add a package closure status record, and optionally define Lean structures for the philosophy map and its no-smuggling checks.*

9462.33.13 Package compression

Definition 9462.2204 (Package compression)**.** *For BEDC philosophy to become a sustainable research object, it must move from interpretation alone to package discipline: every concept is registered, every theorem is mapped, every gap is typed, every tradition comparison is bounded, every scientific claim is layered, and every unsupported claim is recorded as refusal. Philosophy becomes the audit structure of the export surface, not an external override of the kernel.*

Concretized in §1323: Package export discipline is realized as a finite package-map, registry-export, cannot-claim, export-gate, formal-target, non-smuggling, transport, provenance, and NameCert packet.

9462.34 Core Registries and Total Index

The philosophy package becomes maintainable only when its claims are registered. This section turns the preceding synthesis into finite tables for concepts, theorems, gaps, traditions, scientific claims, refusals, upgrade paths, and ordinary-language entry points. The registries add no kernel primitives; they constrain export.

9462.34.1 Registry principles

Definition 9462.2205 (Registry)**.** *A registry is a finite collection of entries. Each entry records name, BEDC site, status, forbidden readings, and upgrade path.*

Definition 9462.2206 (Registry family)**.** *The registry family of the philosophy package consists of concept registry, theorem registry, gap registry, tradition registry, science registry, cannot-claim registry, and upgrade registry.*

Concretized in §1252: Concept registry surfaces realize registry-family export as finite BEDC-site, forbidden-reading, status, route, provenance, and NameCert rows.

Theorem 9462.2207 (Registry family as export surface)**.** *The registry family is the export surface of the philosophy package.*

Proof. The philosophy package cannot export unregistered prose as a BEDC claim. Concepts enter the concept registry, theorems enter the theorem registry, gaps enter the gap registry, comparisons enter the tradition registry, scientific claims enter the science registry, refusals enter the cannot-claim registry, and future promotions enter the upgrade registry. A claim absent from all registries has no carrier, dependency, socket, status, or refusal surface. □

Corollary 9462.2208 (No registry, no export)**.** *An unregistered item is not exportable as part of the philosophy package.*

9462.34.2 Concept registry

Definition 9462.2209 (Concept registry entry)**.** *A concept entry contains name, BEDC site, carrier, classifier, ledger, positive reading, forbidden readings, gap status, closure status, and formal status.*

Definition 9462.2210 (Concept status)**.** *Concept statuses include kernel site, certificate site, boundary site, records-side reading, scientific schema, tradition reading, roadmap, and cannot-claim.*

Theorem 9462.2211 (Concepts require BEDC sites)**.** *Every philosophical concept entering BEDC prose requires a BEDC site.*

Proof. Without a site, the concept remains external vocabulary. Subject talk without Hist, hsame, and Cont becomes subject-substance; universe talk without the focused universe slice becomes a global world object; far-end talk without gap position becomes an object-level entity. □

Definition 9462.2212 (Core concept registry)**.** *The core concept registry is the following finite table.*

Name	***BEDC site***	***Positive reading***	***Forbidden readings***
Distinction	*Mark constructor difference,* msame	*primitive distinction starts generation*	*number, Boolean truth, set pair, metaphysical dualism*
Hist	*generated* Hist, hsame	*minimal information carrier*	*external time object, event set, psychological memory*
Observation	$\mathsf{Cont}(h, d, h')$	*observation is continuation*	*private mental image, subject-object gaze*
Observer	Hist *modulo* hsame	*observer equals history*	*hidden subject, soul-substance, global viewer*
Time	*history growth and local continuation chain*	*time is local history growth*	*global clock, absolute now, external time container*
Object	NameCert-*licensed interface*	*object is licensed name*	*primitive substance, host object, unlicensed term*
Knowledge	*witness chain*	*knowledge is auditable witness*	*naked truth possession, global truth predicate*
Gap	*typed unclosed socket*	*visible boundary*	*secret axiom, hidden proof failure*
Far end	*far end of gap socket*	*transcendental supply position*	*object-level deity, theorem, substrate axiom*
Consciousness	*focused conscious record*	*records-side carrier*	*phenomenal essence, mental substance*
Universe	*focused finite universe slice*	*universe-for-focus*	*global universe object, world totality*
Physical law	*empirical regularity plus Type IV gap*	*law as records plus prediction socket*	*primitive law, final truth, future theorem*

Theorem 9462.2213 (Core concept registry closure)**.** *The core concept registry supplies the minimum registration surface for the central concepts used by the philosophy package.*

Proof. Each row gives a BEDC site, a positive reading, and forbidden readings. The full entry also carries carrier, classifier, ledger, gap status, closure status, and formal status. The table is therefore a paper-synthesis registry, not a machine-checked claim. □

Corollary 9462.2214 (Concept registry extension)**.** *A further concept enters the philosophy package only by supplying the same fields.*

9462.34.3 Theorem registry

Definition 9462.2215 (Theorem registry entry)**.** *A theorem entry records name, statement, BEDC dependencies, proof mode, strength, whether a gap is used, forbidden strengthening, and upgrade path.*

Definition 9462.2216 (Proof mode)**.** *Proof modes include kernel theorem, paper proof, structural reading, schema-only, roadmap, and gap-boundary statement.*

Theorem 9462.2217 (Theorem strength is bounded by dependencies)**.** *Unless a further witness is supplied, the strength of a theorem entry cannot exceed the weakest strength among its dependencies.*

Proof. The theorem obtains its status from the proof chain it cites. A stronger status would assert an obligation not present in the dependencies. □

Definition 9462.2218 (Core theorem registry)**.** *The core theorem registry contains the following entries.*

Name	***Statement***	***Dependencies***	***Strength***
Distinction precedes information	*no distinction, no information*	*primitive marks and no-confusion*	*paper synthesis*
History carries information	*nontrivial information requires generated history*	Hist, hsame, *no-confusion*	*component-dependent*
Observation is continuation	*records-side observation is* Cont	*continuation over histories*	*paper synthesis*
Observer is history	*observer identity is history identity modulo* hsame	*no subject parameter,* hsame, *axiom purity*	*paper synthesis*
Time is history growth	*BEDC time is local continuation chain*	*elementary steps,* Cont, hsame	*paper synthesis*
Object is licensed name	*mature object enters only by* NameCert	*five-field naming certificate*	*component-dependent*
Knowledge is witness	*knowledge is auditable witness*	*certificates, ledgers, open meta loop*	*paper synthesis*
Ground closed, meta open	*ground closes at distinction and meta remains open*	*primitive distinction, formal-limit boundary*	*paper synthesis*
Gap and far-end boundary	*unclosed commitments enter as gaps*	*closed records and socket taxonomy*	*paper synthesis*
Scientific object certificate	*science object is records, invariant, certificate, bridge, gap*	NameCert, *invariant, bridge, Type IV gap*	*roadmap synthesis*

Theorem 9462.2219 (Theorem registry controls strength)**.** *The theorem registry prevents structural readings from being reported as machine-checked theorems.*

Proof. Each row records proof mode, strength, and forbidden strengthening. This makes the difference between component support, paper synthesis, roadmap, and checked theorem visible. □

9462.34.4 Gap registry

Definition 9462.2220 (Gap registry entry)**.** *A gap entry records name, site, requested supply, socket type, gate, consequence if internalized, current status, upgrade path, and related cannot-claim entry.*

Definition 9462.2221 (Socket type registry)**.** *Socket types are unrestricted induction, unwitnessed existential, extensional identification, causal continuity, inter-history coordination, inscription novelty, self-reference closure, and stacked socket.*

Theorem 9462.2222 (Gap entries require gates)**.** *Every gap entry must state the gate that blocks internalization.*

Proof. Without a gate, a gap is only an ordinary pending proof note. A BEDC gap is a boundary claim and must state why the requested supply cannot be used as an internal rule. □

Definition 9462.2223 (Core gap registry)**.** *The core gap registry contains the following entries.*

Name	***Site***	***Socket type***	***Gate***
Induction gap	*finite observation to unrestricted universal*	*Type I*	*generator-closure ledger*
Witness gap	*positive existential without constructor*	*Type II*	*witness visibility and no choice*
Identity collapse gap	*relation treated as equality*	*Type III*	*transport and no quotient collapse*
Causality gap	*empirical regularity to future persistence*	*Type IV*	*records-prediction separation*
Other-minds gap	*behavior to another conscious system*	*Type V*	*locality and inter-history discipline*
Freedom gap	*accumulation event as actuality*	*Type VI*	*aspect separation*
Self-reference gap	*self-description to own truth predicate*	*Type VII*	*reflection stratification*
Physics roadmap gap	*inter-history invariants to physical theory*	*stacked*	*scientific map, bridge, empirical ledger*

Theorem 9462.2224 (Gap registry coverage)**.** *The core gap registry covers the main open boundaries of the philosophy package.*

Proof. The package's central open boundaries are universal closure, unwitnessed existence, identity collapse, causal future, other minds, inscription novelty, self-reference, and physical completion. The table registers each with socket type and gate. □

Concretized in §1246: The theorem and gap registry surface is realized as a finite NameCert packet for theorem strength, gap gates, refusal rows, and audit-visible registry export.

9462.34.5 Tradition registry

Definition 9462.2225 (Tradition registry entry)**.** *A tradition entry records tradition, traditional claim, BEDC site, preserved structure, rejected surplus, and status.*

Theorem 9462.2226 (Rejected surplus is required)**.** *A tradition comparison without rejected surplus is refused.*

Proof. Without rejected surplus, structural adjacency becomes doctrinal identity. The comparison would then import more metaphysics than the BEDC site licenses. □

Definition 9462.2227 (Core tradition registry)**.** *The core tradition registry is as follows.*

Tradition	***BEDC site***	***Preserved structure***	***Rejected surplus***
Spencer-Brown	b0/b1 *distinction*	*distinction precedes form*	*symbolic surplus beyond constructor distinction*
Brouwer	*elementary history step and growth*	*constructive time-order*	*psychologistic metaphysics*
Bundle/no-self/cybernetics	*observer equals* Hist	*observer as trace*	*doctrinal identity*
Kant	NameCert *and gap conditions*	*possibility as obligation*	*transcendental subject primitive*
Husserl	*focused records and* NameCert	*constitution by stabilization*	*intentional substance*
Wittgenstein	*certified use*	*public rule-governed use*	*ordinary-language relativism as object theory*
Whitehead	Cont *as universal verb*	*process priority*	*actual occasions and organic metaphysics*
Tarski/Godel/Muenchhausen	*open meta loop*	*formal boundary*	*none as doctrine; limit family reading*
Relational physics	*inter-history invariant*	*no privileged global frame*	*completed physics derivation*

Corollary 9462.2228 (Tradition registry constrains comparison)**.** *Further tradition comparisons must supply the same fields before entering the main text.*

Concretized in §1267: Tradition registry entries are realized as finite BHist packets with rejected-surplus and comparison-discipline NameCert rows.

9462.34.6 Science registry

Definition 9462.2229 (Science registry entry)**.** *A science entry records name, records carrier, invariant carrier, empirical ledger, prediction gap, bridge status, claim status, and failure surface.*

Theorem 9462.2230 (Science roadmaps require failure surfaces)**.** *Every scientific roadmap claim must list a failure surface.*

Proof. Without a failure surface, a roadmap becomes a one-way promise of progress. BEDC roadmaps must state what would block promotion, such as host leakage, choice dependence, propositional extensionality, missing invariant, or missing empirical ledger. □

Definition 9462.2231 (Core science registry)**.** *The core science registry is as follows.*

Name	***Carrier***	***Claim status***	***Failure surface***
Physical record	*history, continuation, measurement row*	*records-side if ledgered*	*missing source or unledgered measurement*
Space	*locality-cell rows and inter-history graph*	*records-side geometry*	*hidden global frame or unproved invariant*
Physical law	*empirical regularity rows plus Type IV gap*	*scientific schema*	*past fit treated as future truth*
Maximum causal rate	*cross-history dependence and invariant*	*schema-only until causal rows close*	*global clock, hidden choice, missing witness*
Lorentz geometry	*maximum-rate and relation rows*	*roadmap bridge target*	*missing constructive classifier or bridge*
Quantum state	*multi-history superposition rows*	*roadmap*	*propositional extensionality, quotient collapse, unledgered amplitude*

Corollary 9462.2232 (Science registry exports discipline)**.** *The current science registry exports audit discipline, not a completed physical theory.*

Proof. Several entries remain roadmap or schema-only and lack full naming certificates, empirical ledgers, or standard bridges. □

Concretized in §1270: Tradition/science registry bridges realize rejected-surplus and failure-surface rows as one finite NameCert packet.

9462.34.7 Cannot-claim registry

Definition 9462.2233 (Cannot-claim entry)**.** *A cannot-claim entry records claim text, status, why the claim is not claimable, and required upgrade. The registry records present refusal, not a schedule.*

Definition 9462.2234 (Current philosophy cannot-claim registry)**.** *The current cannot-claim registry contains the following active refusals.*

Claim text	***Status***	***Why not claimable***
BEDC has re-derived all mature mathematics	*forbidden as current claim*	*L0–L14 are not fully closed*
BEDC proves full physical theory	*forbidden as current claim*	*multi-history physics is roadmap and schema-only*
BEDC defines phenomenal consciousness as essence	*forbidden*	*conscious record is records-side only*
Far-end position is an object-level entity	*forbidden*	*the far end has no object carrier*
BEDC internalizes its own truth predicate	*forbidden*	*open meta loop and Type VII boundary*
BEDC contains a global clock or God's-eye observer	*forbidden*	*no global synchronization and observer equals history*
Host naturals, host equality, quotient collapse, choice, or propositional extensionality belong to the public surface	*forbidden*	*anti-primitive and axiom-purity discipline*
Tradition comparisons are doctrinal identities	*forbidden*	*tradition registry records adjacency with rejected surplus*
Roadmap equals theorem	*forbidden*	*roadmap lacks discharged certificate*
BEDC philosophy package is checked as a whole	*deferred*	*philosophy map structures are not yet formalized*

Theorem 9462.2235 (Cannot-claim registry protects the framework)**.** *The cannot-claim registry prevents the philosophy package from being read as a completion claim.*

Proof. The preceding parts give a philosophical framework, research route, and package discipline. They do not close all mathematics, physics, consciousness, ontology, or truth. The registry states these refusals explicitly. □

9462.34.8 Upgrade registry

Definition 9462.2236 (Upgrade entry)**.** *An upgrade entry records target, current status, next status, required fields, blocking gaps, and verification target.*

Concretized in §1274: Upgrade registries are realized as finite status-transition, blocker, formal-target, provenance, transport, and NameCert packets.

Definition 9462.2237 (Core upgrade registry)**.** *The core upgrade registry contains the following paths.*

Target	***Current status***	***Next status***	***Blocking surface***
Philosophy package	*paper synthesis export*	*formal package target*	*registry structures and no-smuggling predicates*
Observer-history capstone	*paper structural theorem*	*component checked export*	*formal observer alias and no-subject theorem*
Gap socket taxonomy	*paper taxonomy*	*formal package target*	*socket-type and gate predicates*
Tradition map	*paper registry*	*formal registry*	*external-reference ambiguity*
Science registry	*paper roadmap registry*	*formal roadmap registry*	*multi-history physics certificates*
Multi-history physics	*schema-only roadmap*	NameCert(MultiHistConfig)	*Type V coordination, Type IV causality, bridge obligations*

Theorem 9462.2238 (Upgrade registry is not completion)**.** *An upgrade entry states how a target may be promoted; it does not state that the promotion has occurred.*

Proof. The entry contains current status as well as next status. Until the required fields are discharged, the target remains at its current status. □

9462.34.9 Subject index

Definition 9462.2239 (Subject index)**.** *The subject index maps ordinary-language themes to registry entries.*

Definition 9462.2240 (Core subject index)**.** *The core subject index maps distinction to concept entry distinction and theorem distinction-precedes-information; history to concept entry history and minimal information carrier; observation to concept entry observation; subject and self to observer entry and observer-history theorem; time to local history growth; object to naming certificate; knowledge to witness; truth to the open meta-loop boundary; induction to the induction gap; existence to the witness gap; extensional identity, quotient, and propositional extensionality to the identity collapse gap; causality to the causality gap and physical-law entry; other minds to the Type V gap; freedom to the Type VI gap; consciousness to conscious record and phenomenal-excess gap; universe to the focused universe slice and global-object refusal; space to inter-history locality; physical law to Type IV prediction gap; far end and God-language to the far-end entry and object-level refusal; traditions to the tradition registry; science to the science registry; bridges to standard bridge and no-host-leak; audit to failure certificates and cannot-claim registry; and research route to upgrade registry, layers, and phase bundles.*

Theorem 9462.2241 (Subject index as reading entrance)**.** *The subject index lets readers enter from ordinary philosophical language without treating ordinary terms as BEDC primitives.*

Proof. The index sends freedom to a Type VI gap rather than to a faculty primitive, universe to a focused slice rather than to a global object, and truth to the open meta-loop boundary rather than to an internal truth predicate. □

9462.34.10 Registry synthesis

Theorem 9462.2242 (Registry synthesis)**.** *The registry family gives the philosophy package an auditable export surface: every concept has a BEDC site, every theorem has dependency and strength, every gap has socket type and gate, every tradition comparison has rejected surplus, every scientific claim has status and failure surface, every refusal has reason and upgrade path, every upgrade path distinguishes current status from next status, and every ordinary theme returns to a registered entry.*

Proof. The concept registry supplies sites. The theorem registry supplies dependencies and strength. The gap registry supplies socket type and gate. The tradition registry supplies preserved structure and rejected surplus. The science registry supplies carriers, status, and failure surface. The cannot-claim registry supplies present refusals. The upgrade registry supplies promotion conditions without claiming completion. The subject index routes ordinary vocabulary back to registries. □

Concretized in §1272: Subject-index routes realize registry synthesis as a finite BHist packet connecting ordinary subject entries to the registry lattice.

Corollary 9462.2243 (Registries replace prose expansion)**.** *Further development of BEDC philosophy should proceed mainly by registry extension rather than explanatory prose expansion.*

Proof. Prose can repeat, drift, or overclaim. Registry entries require carrier, dependency, socket, status, refusal, and upgrade path. □

Corollary 9462.2244 (Registry appendix route)**.** *The natural publication form of these tables is a set of appendices: concept registry, theorem registry, gap registry, tradition registry, science registry, cannot-claim registry, upgrade registry, and subject index.*

9462.34.11 Registry status

Definition 9462.2245 (Registry package status)**.** *The registry package is a paper-synthesis export. It fixes registry formats and core entries at the paper level, but it is not a machine-checked registry structure and does not prove all registered claims. It does not close gap sockets, convert roadmaps into theorems, or establish a whole-package no-smuggling theorem.*

Definition 9462.2246 (Registry package upgrade)**.** *To upgrade the registry package, one would define formal structures for concept, theorem, gap, tradition, science, cannot-claim, upgrade, and subject-index entries; encode status monotonicity; prove that exported claims do not conflict with the cannot-claim registry; prove that every philosophy export points to at least one registry entry; and prove no-smuggling for forbidden readings.*

Definition 9462.2247 (Registry compression)**.** *The registry discipline compresses the philosophy package into a single rule: no concept without registry, no theorem without dependency, no gap without socket type, no comparison without rejected surplus, no scientific claim without status, no export without cannot-claim boundary, and no upgrade without current status.*

Concretized in §1248: Refusal registries realize blocked export, gate, cannot-claim route, status, provenance, transport, package, and local NameCert rows as a finite package-control surface.

Concretized in §1251: CoreRegistryIndexUp turns the finite registry tables into a concrete NameCert carrier.

9462.35 Registry Layer, Manuscript Form, and Machine-Facing Interface

The registry layer is not supplementary material. It is the export-control surface of the philosophy package. The prose develops concepts and readings; the registry layer determines which claims may be exported, which must be downgraded, which are refused, and which can become future formalization targets.

Concretized in §1221: Machine-interface boundaries realize registry export, refusal, formalization-target routing, audit readback, and machine-readable NameCert rows as one finite packet.

Concretized in §5205: Machine export boundaries bind registry export status, machine target, audit readback, refusal preservation, transport, replay, provenance, and local NameCert rows as one finite packet.

9462.35.1 Registry layer

Definition 9462.2248 (Registering for export)**.** *To register a claim for export is to assign it to the appropriate registry surface: concept registry, theorem registry, gap registry, tradition registry, science registry, cannot-claim registry, upgrade registry, subject index, closure-status record, or formalization target.*

Definition 9462.2249 (Export-control surface)**.** *The registry layer is an export-control surface when every exported claim either appears in a registry entry or is blocked by a registry entry.*

Concretized in §1216: Registry export surfaces realize claim registration, blocking, status, machine-target, audit, and non-smuggling rows as one finite NameCert packet.

Theorem 9462.2250 (Registry layer is not supplementary)**.** *For the BEDC philosophy package, the registry layer is part of the claim surface rather than supplementary documentation.*

Proof. The package's claims are licensed only by their carriers, classifiers, ledgers, certificates, gap reports, bridge status, and cannot-claim boundaries. Those fields are recorded in registries. A prose claim not represented there has no audit surface. □

Corollary 9462.2251 (No registry, no package claim)**.** *If a philosophical claim is absent from the registry layer and not blocked by it, it is not a package-level BEDC claim.*

Concretized in §1209: ClaimRegistryLayerUp realizes the registry layer as finite export-control rows for claim entry, refusal boundary, closure status, formal target, provenance, and local NameCert.

Concretized in §1220: Registry-layer export control is realized as a finite NameCert packet whose rows admit, block, transport, and ledger package-level claims.

9462.35.2 Manuscript form

Definition 9462.2252 (Two-surface manuscript)**.** *A mature BEDC philosophy manuscript has two surfaces: the synthesis surface and the registry surface. The synthesis surface develops the concepts, theorem readings, and research programme. The registry surface records their export status, refusal boundaries, and machine interface.*

Definition 9462.2253 (Reference module)**.** *A reference module is a stable registry-facing unit for one family of claims: concepts, theorems, gaps, traditions, scientific claims, cannot-claim entries, upgrade entries, subject terms, closure-status records, or formal targets.*

Theorem 9462.2254 (Manuscript completion requires both surfaces)**.** *The philosophy manuscript is package-complete only when prose synthesis is paired with registry reference modules.*

Proof. The synthesis surface alone can state the philosophical reading, but cannot by itself show every carrier, forbidden reading, gap type, refusal entry, upgrade path, and formalization target. The registry surface supplies those fields. □

Corollary 9462.2255 (Reference modules are not lower status)**.** *Reference modules should not be read as lower-status material. They are the audit side of the same package.*

9462.35.3 Export interface

Definition 9462.2256 (Philosophy export interface)**.** *A philosophy export interface is the rule that a claim X may be exported only when it has an appropriate registry entry, no active cannot-claim conflict, a recorded closure status, and a strength not exceeding its dependencies.*

Definition 9462.2257 (Export grade)**.** *Export grades are reading export, paper synthesis export, component checked export, formal package target, and formal package checked. These grades report package status and do not create object-theory truth.*

Theorem 9462.2258 (Export interface bounds prose)**.** *The export interface bounds the strength of philosophical prose.*

Proof. A prose claim can be exported only at the grade recorded in its registry and closure-status surface. If the registry records a roadmap, the prose cannot call it a theorem; if it records a cannot-claim conflict, the prose cannot export it at all. □

Corollary 9462.2259 (Roadmaps stay roadmaps)**.** *A roadmap entry becomes stronger only after its required certificate, bridge, or formal target is supplied.*

9462.35.4 Registry consistency

Definition 9462.2260 (Registry consistency)**.** *Registry consistency holds when each exported claim has exactly one primary registry location and all related registry entries agree on status, dependency strength, gap status, cannot-claim boundaries, closure status, and formalization target.*

Definition 9462.2261 (Consistency clauses)**.** *The consistency clauses are: concept status agrees with theorem dependencies; gap status agrees with cannot-claim entries; science roadmap status agrees with upgrade entries; tradition rejected surplus agrees with cannot-claim boundaries; closure status agrees with all registered entries; and formalization targets agree with formal status.*

Theorem 9462.2262 (Registry consistency is necessary for export)**.** *If registry consistency fails, the philosophy package is not exportable at the claimed grade.*

Proof. A concept cannot be paper synthesis in the concept registry and machine checked in the theorem registry without a status conflict. A physics claim cannot be roadmap in the science registry and completed theory in prose without a claim conflict. A forbidden gap cannot have a routine proof upgrade without a boundary conflict. These conflicts block export at the asserted grade. □

Corollary 9462.2263 (Consistency before polish)**.** *Registry consistency should be checked before stylistic refinement of the manuscript.*

Concretized in §1224: Registry export consistency gates realize registry entries, blocking rows, status conflicts, formal targets, audit routes, and no-smuggling checks as one finite NameCert packet.

Concretized in §1223: Registry consistency audits are realized as finite registry-row, status-agreement, dependency-strength, cannot-claim, closure/formal-status, transport, replay, provenance, and NameCert packets.

9462.35.5 Machine-facing interface

Definition 9462.2264 (Formalization target)**.** *A formalization target records name, intended namespace, source registry, statement skeleton, dependencies, expected status, and not claimed surface.*

Definition 9462.2265 (Namespace draft)**.** *The machine-facing namespace may be organized under* BEDC.Philosophy*, with subnamespaces for registry, concepts, theorem maps, gaps, traditions, science, cannot-claim entries, upgrades, subject index, package, and consistency.*

Definition 9462.2266 (Entry structure draft)**.** *The first structures to encode are concept entry, theorem entry, gap entry, tradition entry, science entry, cannot-claim entry, upgrade entry, subject-index entry, closure-status entry, formalization target, amendment-ledger entry, and registry-consistency predicate.*

Theorem 9462.2267 (First target is well-formedness)**.** *The first machine-facing target is registry well-formedness, not the truth of all philosophical doctrines.*

Proof. Tradition entries are mappings, not proofs of traditions. Concept entries are carrier registrations, not global truth predicates. The machine interface can first check field presence, status consistency, cannot-claim conflicts, no-smuggling predicates, and dependency monotonicity. □

Corollary 9462.2268 (No-smuggling predicates come first)**.** *The early formal predicates should be no philosophical smuggling, no mathematical smuggling, no host smuggling, no roadmap-as-theorem, and no cannot-claim conflict.*

9462.35.6 Socket type draft

Definition 9462.2269 (Socket type draft)**.** *A future socket type can be encoded as an inductive family with constructors for unrestricted induction, unwitnessed existential, extensional identification, causal continuity, inter-history coordination, inscription novelty, self-reference closure, and stacked socket.*

Theorem 9462.2270 (Socket typing precedes gap automation)**.** *Any automated check of gap reports requires a typed socket surface.*

Proof. The requested supply, gate, upgrade path, and internalization failure differ by socket kind. Without a socket type, the checker cannot distinguish an induction boundary from a witness boundary or a self-reference boundary. □

Concretized in §1059: Machine-facing formalization targets are realized as finite socket-routed NameCert packets.

Concretized in §1225: Formalization targets are realized as finite registry-, status-, dependency-, audit-gate-, and not-claimed rows.

Concretized in §1344: Registry no-smuggling predicates are realized as finite registry, cannot-claim, gate, export, transport, replay, provenance, and NameCert rows.

Concretized in §1347: Formal-target dependency audits realize machine-facing target dependency and no-smuggling checks as one finite NameCert packet.

9462.35.7 Amendment ledger

Definition 9462.2271 (Amendment ledger entry)**.** *An amendment ledger entry records target, change type, old status, new status, reason, affected registries, cannot-claim impact, and gap impact.*

Definition 9462.2272 (Change type)**.** *Change types include add entry, replace entry, downgrade, upgrade, reclassify gap, remove cannot-claim, add cannot-claim, revise forbidden reading, and amend formal target.*

Theorem 9462.2273 (Amendment ledger does not create parallel readings)**.** *The amendment ledger records audit continuity, but it does not preserve multiple active readings.*

Proof. The governing manuscript remains the active source of truth. When a reading is replaced, the active formulation takes the citation target. The ledger records the status change and affected registries; it does not keep the displaced formulation active. □

Corollary 9462.2274 (Amendment ledger is not an archive)**.** *The amendment ledger explains the current active reading; it is not a repository of prior prose.*

9462.35.8 Closure-status reference layer

Definition 9462.2275 (Closure-status reference)**.** *A closure-status reference records constructive story, theory closure, scope closed, formal status, Lean target if any, bridge status, not claimed surface, and upgrade path for a package, part, concept, or registered theorem.*

Definition 9462.2276 (Closure-status bound)**.** *The closure-status reference is the strongest export status allowed for its target.*

Theorem 9462.2277 (Closure status bounds registry exports)**.** *Registry entries may not export a claim stronger than the relevant closure-status reference.*

Proof. If the closure reference records paper synthesis, a registry entry cannot report whole-package machine checking. If the not-claimed field rejects a global truth predicate, a theorem entry cannot reintroduce it under another name. □

Corollary 9462.2278 (Closure references belong to the registry layer)**.** *Closure-status references are not marginal notes. They are binding entries in the export-control surface.*

Concretized in §1265: Closure-status references are realized as a finite NameCert packet for registry-bound export status.

9462.35.9 Interface synthesis

Theorem 9462.2279 (Registry and machine-interface synthesis)**.** *A mature BEDC philosophy package consists of a synthesis surface and a registry-machine surface. The synthesis surface develops the philosophical structure. The registry-machine surface registers every exportable claim, blocks forbidden claims, classifies gaps, records upgrade paths, constrains formalization targets, and preserves amendment continuity.*

Proof. The registry layer provides export-control entries. The export interface bounds prose by status and cannot-claim conflicts. Registry consistency ensures that related entries agree. The machine-facing interface identifies future structures and no-smuggling predicates. The amendment ledger records active-reading changes without preserving parallel readings. Closure-status references bound the whole surface. □

Corollary 9462.2280 (After registry layer)**.** *Further work should populate the reference modules, check registry consistency, link each prose theorem to the theorem registry, link each strong claim to closure status, link each boundary to the gap and cannot-claim registries, link each future claim to the upgrade registry, and prepare the* BEDC.Philosophy *namespace.*

Corollary 9462.2281 (Manuscript architecture)**.** *The final manuscript can be organized as philosophical synthesis, registry and audit reference modules, and machine-facing interface with amendment governance. In a shorter article, the synthesis may occupy the main body while registries and machine interface form the reference layer.*

9462.35.10 Interface status

Definition 9462.2282 (Machine-interface status)**.** *The present interface is paper synthesis export. The reference-module architecture is defined, but the registries are not fully populated, the structures are not machine checked, registry consistency is not proved in Lean, and the interface does not close gaps or turn roadmaps into theorems.*

Definition 9462.2283 (Interface upgrade path)**.** *The upgrade path is to populate the reference modules, define Lean structures for registry entries and closure references, encode registry consistency, prove that no exported claim lacks a registry entry, prove that no registered claim conflicts with cannot-claim entries, prove that roadmap claims have upgrade entries and refusal boundaries, and prove no-smuggling for forbidden readings.*

Definition 9462.2284 (Interface compression)**.** *The registry layer is the export-control surface of BEDC philosophy: prose explains, registries license, cannot-claim entries refuse, closure status bounds, and the machine interface turns these fields into future formal targets.*

Concretized in §1232: Machine-readable audit interfaces realize registry-machine synthesis as finite schema, checker, export, refusal, report, hsame, Cont, Pkg, and NameCert rows.

Concretized in §1358: Registry-change continuity is realized as a finite active-reading, blocked-reading, status-bound, refusal, transport, replay, provenance, and NameCert packet.

Concretized in §1256: finite synthesis-audit packet for auditable-generativity export

Concretized in §7399: Finite observation sheaves realize auditable generativity as compatible BHist window sections with finite overlap gluing.

Chapter 9463

Hyperbolic Phase Geometry of Observer Expansion

Concretized in §39: The visible complex phase surface is grounded in the finite $\mathsf{Complex}^{\uparrow}$ rational-pair certificate.

Concretized in §377: The disk carrier used by the geometric reading is grounded in the finite $\mathsf{UnitDisk}^{\uparrow}$ certificate.

Concretized in §219: Reversible phase transport is grounded in the finite $\mathsf{UnitaryGroup}^{\uparrow}$ inner-product-preserving automorphism certificate.

Concretized in §195: Flow-style record evolution is grounded in the finite $\mathsf{DynSystem}^{\uparrow}$ routed-flow certificate.

Concretized in §7608: Critical-pair residual streams realize the MetaCIC parallel-diamond frontier route as finite residual-window, candidate-join, relation-scope, and inter-Hist invariant rows.

Concretized in §3455: The Poincare disk phase-transport packet realizes the finite disk-coordinate, reversible-transport, flow, and provenance route.

Concretized in §7806: The Poincare half-plane packet realizes the upper-half-plane hyperbolic coordinate, boundary, metric, Mobius transport, disk-comparison, Real-seal, provenance, and NameCert route.

Concretized in §4117: The Poincare disk dynamics packet realizes the fixed far-end disk-map route as finite disk, metric, boundary, phase-provenance, dynamic-replay, and refusal rows.

Concretized in §3474: HyperbolicMetricUp realizes the Poincare-disk metric reading as finite disk-map, metric-read, boundary, classifier-stability, provenance, refusal, and NameCert rows.

Concretized in §6336: $\mathsf{CATZeroMetric}^{\uparrow}$ realizes the nonpositive-curvature comparison route as finite metric, geodesic, Real-seal, transport, and NameCert rows.

Concretized in §6776: Poincare disk automorphisms are realized as finite disk-map, boundary-transport, hyperbolic-metric-preservation, inverse, composition, provenance, and NameCert rows.

Concretized in §7499: Poincare disk automorphism derivatives are realized as finite automorphism, coordinate-window, metric-preservation, boundary-transport, Real-seal, and NameCert rows.

Concretized in §3486: The hyperbolic observer-expansion reading is realized as finite focused-ledger, disk-readout, Cont-growth, metric-read, boundary-refusal, provenance, and NameCert rows.

Concretized in §3491: The boundary-geodesic route is realized as finite Poincare-disk, HyperbolicMetric, phase-transport, and provenance NameCert rows.

Concretized in §3500: Boundary-fixed disk transport is realized as finite disk-map, boundary, hyperbolic-metric, transport, provenance, refusal, and NameCert rows.

Concretized in §3534: Horocycle packets realize the boundary-tangent hyperbolic disk rows as finite NameCert data.

Concretized in §8521: The hyperbolic horosphere packet realizes boundary point, Busemann level, metric readback, observer-window, Real-seal, transport, provenance, and NameCert rows as finite observer-expansion data.

Concretized in §3575: The Denjoy-Wolff boundary-dynamics reading is realized as finite disk-map, boundary-read, nonexpansive-route, fixed-boundary, provenance, and NameCert rows.

Concretized in §3591: The phase-ledger horocycle packet realizes the focused phase-ledger and boundary-transport reading as finite NameCert rows.

Concretized in §3643: The boundary-phase audit packet realizes the observer-expansion boundary verdict as finite ledger, boundary, horocycle, transport, provenance, and NameCert rows.

Concretized in §3682: The Schwarz-Pick disk-contraction packet realizes the finite disk-map, hyperbolic-metric comparison, boundary refusal, transport, provenance, and NameCert rows.

Concretized in §3732: The HyperbolicBusemannBoundary packet realizes Busemann-level boundary metric, horocycle, boundary-fixed transport, provenance, and NameCert rows.

Concretized in §3813: Boundary-shadow packets realize the boundary-fixed disk transport reading as finite NameCert rows.

Concretized in §6070: The Patterson-Sullivan shadow packet realizes boundary-shadow, visual-measure, hyperbolic-metric, and observer-expansion rows as finite NameCert data.

Concretized in §3856: The hyperbolic geodesic-flow packet realizes disk metric, boundary, flow, and provenance rows as finite NameCert data.

Concretized in §3866: The visual metric packet realizes disk readout, hyperbolic metric rows, boundary shadows, transport, provenance, and global-geometry refusal as finite NameCert data.

Concretized in §3925: The hyperbolic Poisson-kernel packet realizes boundary-weighted disk reads as finite metric, boundary, phase, integral, and Real-seal rows.

Concretized in §3948: The hyperbolic cross-ratio packet realizes four-boundary-point Poincare disk comparison, boundary-fixed transport, hyperbolic-metric provenance, and refusal of host global geometry as finite NameCert rows.

Concretized in §7493: The visual cross-ratio cocycle packet realizes boundary-shadow, visual-metric, cross-ratio, Busemann, transport, Real-seal, and NameCert rows as finite hyperbolic phase data.

Concretized in §3683: The Poincare triangle-defect packet realizes disk-point, geodesic-side, hyperbolic-metric, cross-ratio, and Real-seal rows as finite NameCert data.

Concretized in §4034: The harmonic boundary-measure packet realizes the visual-metric and Poisson-kernel boundary weighting route as finite NameCert data.

Concretized in §4051: The horosphere-flow ledger realizes boundary-fixed disk transport and horocycle provenance as finite NameCert data.

Concretized in §4134: The Busemann cocycle route is realized as finite boundary-point, geodesic-ray, horocycle, metric-read, transport, provenance, and NameCert rows.

Concretized in §7534: Busemann-function packets realize boundary-point, geodesic-ray, metric-difference, Real-seal, and horosphere readback rows for the hyperbolic observer-expansion vision.

Concretized in §4142: The boundary-shadow visual metric is realized as finite disk-read, boundary, metric, Busemann, transport, provenance, and NameCert rows.

Concretized in §4154: The observer-shadow visual metric is realized as finite visual-shadow, metric, stream-window, regular-rational, real-seal, focused-route, provenance, and NameCert rows.

Concretized in §4167: The observer-expansion packet realizes the hyperbolic phase geometry as finite shadow-metric and boundary-shadow rows.

Concretized in §4181: The hyperbolic visual boundary packet realizes boundary readback through finite visual metric, Busemann, shadow, transport, and provenance rows.

Concretized in §4203: The geodesic lamination packet realizes boundary-paired noncrossing hyperbolic leaves as finite metric, shadow, transport, provenance, and NameCert rows.

Concretized in §5802: The quasi-geodesic shadow packet realizes boundary endpoint, visual-metric shadow, Busemann cocycle, and observer-expansion rows as finite NameCert data.

Concretized in §5814: The boundary-shadow expansion cocycle is realized as finite boundary, Busemann, horosphere, expansion, provenance, and NameCert rows.

Concretized in §5832: The hyperbolic translation-length packet realizes isometry displacement growth through finite metric, flow, Busemann, cross-ratio, and Real rows.

Concretized in §5107: The hyperbolic fixed-point obstruction packet realizes the boundary-route refusal of an interior far-end fixed point as finite NameCert rows.

Concretized in §5180: The observer shell packet realizes the boundary-shadow, visual-boundary, and observer-expansion rows as one finite NameCert surface.

Concretized in §6073: The Gromov-product packet realizes the basepointed boundary-shadow and visual-metric comparison route as finite NameCert rows.

Concretized in §5917: The geodesic projection packet realizes disk-point, boundary-geodesic, hyperbolic-metric, and phase-transport rows as finite NameCert data.

Concretized in §6017: Phase-distance observer expansion is realized as finite observer-phase, distance-read, expansion-comparison, provenance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8178: The observer phase-distance shell realizes the boundary-shadow, phase-distance, observer-window, and provenance rows as finite NameCert data.

Concretized in §6090: The linearized phase-distance packet realizes observer expansion through finite boundary-shadow, visual-metric, Real-seal, Cont, Pkg, and NameCert rows.

Concretized in §6104: The geodesic-flow ledger realizes observer expansion through finite flow-window, horosphere-ledger, boundary-shadow, visual-metric, Real-seal, provenance, and NameCert rows.

Concretized in §6044: The complex phase surface fiber packet realizes phase readout as finite source ledger, ComplexUp coordinate, fiber, GAP residue, provenance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6021: The observer expansion radius ladder is realized as finite disk, boundary-shadow, visual-metric, real-seal, transport, replay, provenance, and NameCert rows.

Concretized in §6025: The horofunction-boundary packet realizes boundary metric readout through finite Busemann, shadow, transport, refusal, provenance, and NameCert rows.

Concretized in §6040: The Morse boundary realizes the hyperbolic observer-expansion route as finite geodesic stability and fellow-traveller NameCert data.

Concretized in §6122: The axis-shadow packet realizes translation-length, geodesic-flow, metric, Busemann-boundary, and Real-seal rows as a finite non-selected shadow route.

Concretized in §6146: The observer phase groupoid realizes reversible phase-surface transport, inverse, composition, boundary refusal, provenance, and NameCert rows as finite data.

Concretized in §6209: The Poincare disk metric packet realizes the vision's displayed disk-map, metric-read, boundary, provenance, and refusal rows as finite NameCert data.

Concretized in §6862: GromovHyperbolicMetricUp realizes the four-point hyperbolic metric route as finite metric, geodesic, boundary-handoff, transport, provenance, and NameCert rows.

Concretized in §8047: The finite weighted hyperbolic barycenter packet realizes barycenter read-back through HyperbolicMetric, Gromov boundary, Real-seal, transport, and NameCert rows.

Concretized in §6237: Mobius-invariant disk transport realizes the vision's boundary-fixed hyperbolic phase geometry as finite NameCert rows.

Concretized in §6267: Boundary-preserving Poincare disk transport is realized as finite disk, boundary, metric, transport, provenance, and NameCert rows.

Concretized in §8826: Lorentz boost disk transport realizes the reversible hyperbolic disk-transport branch as finite disk, boost-parameter, boundary, metric, phase, transport, provenance, and NameCert rows.

Concretized in §6319: The observer phase shell is realized as finite boundary-shadow, visual-boundary, phase-distance, transport, provenance, and NameCert rows.

Concretized in §7170: The observer-shadow boundary packet realizes the hyperbolic phase-expansion route as finite shadow-window and visual-boundary NameCert data.

Concretized in §6385: The phase holonomy packet realizes closed reversible phase transport through finite disk, boundary, classifier, replay, provenance, and NameCert rows.

Concretized in §6385: Closed reversible phase transport realizes the hyperbolic phase-geometry route through finite disk, boundary, classifier, replay, provenance, and NameCert rows.

Concretized in §6372: TeichmullerSpaceUp realizes the marked conformal-structure branch of the hyperbolic observer-expansion vision through ManifoldUp, ComplexUp, and HyperbolicMetricUp rows.

Concretized in §6437: The finite curvature-budget packet realizes hyperbolic phase observer expansion through disk, metric, boundary, Real-seal, provenance, and refusal rows.

Concretized in §6477: The hyperbolic uniform-modulus packet realizes the boundary, cross-ratio, visual-metric, and tolerance-only modulus route as finite NameCert data.

Concretized in §7589: Mobius disk cross-ratio packets realize boundary quadruple, disk transport, cross-ratio readback, hyperbolic metric provenance, and boundary refusal rows as finite NameCert data.

Concretized in §6481: The hyperbolic phase-energy ledger is realized as finite reversible-transport, metric-budget, boundary-refusal, Real-seal, provenance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §6497: The hyperbolic axis-projection packet realizes disk-metric, geodesic-flow, boundary-shadow, Busemann, refusal, provenance, and NameCert rows.

Concretized in §6678: The boundary-shadow transport packet realizes shadow, boundary-transport, Gromov-boundary, disk-metric, Real-seal, nontrivial, field-faithful, refusal, provenance, and NameCert rows.

Concretized in §6707: Hyperbolic shadowing realizes the quasi-geodesic fellow-traveller route as finite HyperbolicMetric, MorseBoundary, Busemann, Real tolerance, and NameCert rows.

Concretized in §7764: The limit-set packet realizes boundary accumulation through finite visual-boundary, shadow, Busemann, visual-metric, geodesic-flow, support, disk, Real-seal, provenance, and NameCert rows.

Concretized in §6920: The observer phase route is realized as finite boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, provenance, and NameCert rows.

Concretized in §7552: Busemann projection realizes the hyperbolic observer-expansion route as finite boundary, Busemann, visual-metric, observer-window, provenance, and NameCert rows.

Concretized in §6980: The phase horocycle packet realizes boundary-tangent phase transport, hyperbolic metric readback, and observer-expansion rows as finite NameCert data.

Concretized in §6995: The geodesic midpoint packet realizes the Poincare disk midpoint route as finite metric, boundary, phase-transport, provenance, and NameCert rows.

Concretized in §7121: The phase geodesic midpoint packet realizes the displayed Poincare disk chord midpoint through finite metric, boundary-transport, geodesic-flow, phase, provenance, and NameCert rows.

Concretized in §7019: The hyperbolic boundary compactification packet realizes visual-boundary, Gromov-boundary, visual-metric, Busemann, boundary-shadow, and Real-seal rows as finite NameCert data.

Concretized in §7108: The hyperbolic phase-space packet realizes reversible disk-phase transport, boundary refusal, finite observer-phase groupoid windows, and provenance rows as finite NameCert data.

Concretized in §7138: The hyperbolic phase torsion packet realizes reversible closed phase transport as finite disk, boundary, metric, holonomy, transport, provenance, and NameCert rows.

Concretized in §7232: The observer-horizon packet realizes visual boundary, Gromov product, Busemann cocycle, horoball-shadow, observer-window, and Real-seal rows as finite NameCert data.

Concretized in §7274: The Beltrami–Klein disk packet realizes the projective disk-model comparison as finite chord, metric, boundary-incidence, refusal, and Real-seal rows.

Concretized in §7282: Fuchsian group actions are realized as finite disk-point, Mobius-action word, metric-preservation, boundary-handoff, transport, replay, provenance, and NameCert rows.

Concretized in §7423: A finite geodesic endpoint and cross-ratio packet realizes the phase-geometry direction as NameCert data.

Concretized in §7503: The dyadic shadow-cone packet realizes boundary-shadow and observer-expansion reads through finite dyadic radius windows.

Concretized in §7666: The Dirichlet-domain packet realizes the Poincare disk and Fuchsian action branch as finite side-pairing, metric-read, boundary-refusal, and NameCert rows.

Concretized in §8715: The ideal pentagon boundary-cycle packet realizes five displayed boundary endpoints, side-geodesic incidence, hyperbolic metric comparison, cross-ratio readback, and Real-seal rows as finite NameCert data.

Concretized in §7592: The fixed-boundary hyperbolic shadow packet realizes boundary-preserving shadow transport as finite NameCert data.

Concretized in §7659: The ideal quadrilateral packet realizes four displayed boundary ends, side and diagonal incidence, finite cross-ratio readback, and transport as hyperbolic phase NameCert data.

Concretized in §8017: The geodesic triangle packet realizes three displayed vertices, three geodesic sides, incidence or order ledgers, finite comparison readback, transport, provenance, and local NameCert rows.

Concretized in §7887: The Cayley operator packet realizes disk-transport and operator-boundary rows as finite NameCert data.

Concretized in §8976: The hyperbolic visual-measure packet realizes visual-boundary, boundary-shadow, hyperbolic-metric, Real scalar, finite-total, field-faithfulness, transport, provenance, and NameCert rows as finite observer-expansion data.

Concretized in §7891: The Poincare disk geodesic triangle packet realizes the finite disk-point, geodesic-side, hyperbolic-metric, phase-transport, and provenance route.

Concretized in §7927: The Poincare disk Busemann cocycle packet realizes boundary-anchored disk transport, additive cocycle, visual-metric, horocycle, exactness, refusal, and NameCert rows as finite data.

Concretized in §7968: Schottky group packets realize the ping-pong boundary-domain route through finite Poincare disk automorphism, hyperbolic metric, limit-set, and boundary-shadow rows.

Concretized in §7998: The horocycle-shadow modulus packet realizes boundary-tangent horocycle, Busemann, visual-boundary, shadow-budget, metric, Real-seal, transport, provenance, and NameCert rows as finite observer-expansion data.

Concretized in §8298: Observer-phase locality packets realize the hyperbolic observer expansion route as finite observer-phase, residual-window, local-join, candidate-boundary, obstruction-retention, transport, replay, provenance, and NameCert rows.

Concretized in §8546: The sector-envelope packet realizes finite boundary-shadow, visual-radius, observer-window, provenance, and refusal rows for the hyperbolic observer-expansion vision.

Concretized in §8685: The hyperbolic sector-expansion packet realizes sector aperture, boundary shadow, visual radius, observer window, expansion comparison, transport, refusal, provenance, and NameCert rows as finite data.

Concretized in §9013: The sector-overlap ledger realizes aperture, boundary-shadow, visual-radius, observer-window, overlap-witness, provenance, and NameCert rows as finite data.

Concretized in §9018: The sector-refinement packet realizes aperture, boundary-shadow, visual-radius, observer-window, overlap, transport, provenance, and refusal rows as finite NameCert data.

Concretized in §8570: Boundary cross-ratio distortion is realized as finite visual-metric, Busemann, boundary-shadow, Real-seal, transport, replay, provenance, and NameCert rows.

Concretized in §8850: The horoball-filtration packet realizes nested Busemann-level shadow rows as finite observer-expansion NameCert data.

Concretized in §9060: The finite observer-phase shell realization packages boundary-shadow, visual-boundary, phase-distance, reversible-transport, and provenance rows as NameCert data.

Concretized in §9144: Busemann-boundary budgets realize the hyperbolic observer-expansion boundary rows as finite NameCert data.

Concretized in §9162: The horosphere Busemann flow budget realizes boundary-fixed flow windows, Busemann levels, horosphere rows, visual-metric readback, and Real-seal provenance as finite NameCert data.

Concretized in §9188: The finite phase-lens packet realizes observer-expansion windows as boundary-shadow, Busemann-displacement, Real-seal, transport, replay, provenance, and NameCert rows.

This vision chapter gives a geometric language for the observer-expansion reading of chapter 9420, the information-conservation discipline of chapter 9393, and the apophatic fixed-point reading of chapter 9401. The purpose is expressive: complex phase surfaces, unitary transport, Liouville-style conservation, the Poincare disk, and hyperbolic geometry organize the same BEDC rows under a familiar geometric picture. They do not replace Hist, Cont, hsame, GAP, Pkg, or NameCert, and they do not turn T into a point of a plane, disk, flow, or Hilbert space.

The guiding picture is:

$$\begin{aligned} \text{observer-side expansion} &= \text{hyperbolic unfolding of local record surfaces;} \\ \text{reversible conservation} &= \text{classifier-preserving transport with provenance;} \\ \text{apophatic fixedness} &= \text{boundary far-end fixedness, not value fixedness;} \\ \text{self-center} &= \text{local inscription of that far end on a focused chain.} \end{aligned}$$

This is a records-side and inscription-side representation schema. It is not a physical cosmology, not a hidden dynamics of the universe, and not a substrate theorem about phenomenal consciousness.

9463.1 Phase surfaces

Definition 9463.1 (Complex phase surface)**.** *A* complex phase surface *for a focused finite ledger $\mathcal{L}_N$ is a visible package whose public coordinates factor through* $\mathsf{Complex}^{\uparrow}$ *rows and whose source histories remain the generated* Hist *rows, selector rows, and* GAP *provenance rows of $\mathcal{L}_N$. It is a display surface for phase, direction, amplitude, or digest-like presentation; it is not a replacement carrier for the observer.*

The point of the complex surface is to display rotation-like and phase-like regularities without pretending that the observer is a complex number. A complex coordinate is a readout of a finite packet. The observer remains the focused Hist row, and sameness remains mediated by hsame and the displayed classifier rows.

Principle 9463.2 (Phase is a readout, not a source)**.** *If a focused history $H(F)$ is assigned a complex phase readout $z_F(t)$, the readout may be consumed only through its source ledger:*

$$H(F), \Theta(F), S(F), G(F), \rho \quad \rightsquigarrow \quad z_F(t).$$

The reverse reading is admissible only when an exactness, fiber, or GAP *row records what the visible complex token hides. Digest equality or phase equality is therefore not source equality.*

9463.2 Reversible transport and Liouville conservation

Definition 9463.3 (Unitary transport reading)**.** *A* unitary transport reading *of a local phase surface is a reversible transport row that preserves the relevant classifier or inner-product readout while carrying the provenance ledger along the route. Schematic notation may be written*

$$U_t : \Sigma_i \to \Sigma_i, \qquad U_t^* U_t \equiv_{\text{ledger}} 1,$$

where Σ_i is the displayed phase surface of a local history and $\equiv_{\text{ledger}}$ means equality only as certified by the visible transport, preservation, and provenance rows.

This reading belongs at the representation layer. The unitary row is not a hidden law of reality and not an unlogged operator on an ambient Hilbert space. Its BEDC meaning is that a visible surface can move by a reversible classifier-preserving route without erasing the source distinctions that made the surface visible.

Principle 9463.4 (Liouville conservation as ledger conservation)**.** *The Hamiltonian Liouville analogy is admissible when it is read as conservation of record volume under reversible flow:*

$$\textit{reversible phase flow} \quad \Rightarrow \quad \textit{no unledgered loss of source distinctions.}$$

Inside BEDC this is not a theorem about a completed symplectic manifold. It is the information-conservation condition that every transported surface keeps its source, route, classifier, and GAP *residue auditable.*

Concretized in §3487: The reversible phase-transport conservation row is realized as finite classifier-preservation and provenance ledger data.

Concretized in §337: The finite variational, dynamic-flow, Real-cost, and conservation-ledger rows realize the Hamiltonian Liouville analogy as LagrangianMechanicsUp NameCert data.

Remark 9463.5 (Complex Liouville as a warning)**.** *The complex-analytic Liouville theorem is useful only as a warning against a global digest picture. A bounded entire global universe token would be the wrong shape for BEDC: the project does not admit $H(\Omega) = \mathsf{T}$, a global readable universe Ω, or T as a digest value. The legitimate object is a family of local visible surfaces with fiber and provenance rows.*

9463.3 Poincare disk representation

Definition 9463.6 (Poincare disk reading)**.** *A* Poincare disk reading *of a focused multi-*Hist *ledger is a representation package*

$$\mathcal{P}_N = (\mathcal{L}_N, \mathbb{D}, \delta, \partial\mathbb{D}, B, \Gamma, A)$$

with the following rows:

1. $\mathcal{L}_N$ *is the finite focused multi-*Hist *ledger of Definition 9420.1;*
2. $\mathbb{D}$ *is a disk surface read through* $\mathsf{UnitDisk}^{\uparrow}$*-style coordinates, not a host-level subset of* $\mathbb{C}$*;*
3. δ *is a hyperbolic-distance readout row on displayed disk points, when such a readout is supplied;*
4. $\partial\mathbb{D}$ *is a boundary consumer row, not a completed ideal universe;*
5. B *assigns selected local histories or digest rows to displayed disk points;*
6. Γ *records reversible disk-surface transport rows, such as unitary or disk-automorphism-style transport, only through displayed classifier-preserving witnesses;*
7. A *is the audit row refusing object-level* T*, a global clock, a global universe digest, physical light-speed derivation, and subject-substance.*

The disk reading separates two roles that are easy to confuse. Interior points represent visible local records or focused readouts. Boundary positions represent horizon-style or far-end-style readings. The apophatic fixed point belongs only to the boundary reading. It is not a disk point and not an interior center.

Principle 9463.7 (Boundary fixedness, not center fixedness)**.** *The geometric form of BEDC fixedness is not*

$$\mathsf{T} \in \mathbb{D}.$$

It is the boundary-far-end reading

$$\mathrm{FarEnd}(\mathrm{Fiber}(B(H_i, t))) \equiv_{\mathrm{apo}} \mathsf{T}.$$

Thus T *is not an interior center around which records literally rotate. It is the apophatic name of the common far end behind local disk-surface fibers.*

9463.4 Hyperbolic observer expansion

Hyperbolic geometry is useful because local finite motion can correspond to large internal distance near a boundary. This gives a disciplined version of the expansion image: a local observer sees record space unfold rapidly as histories, digests, relation rows, and boundary directions accumulate. The boundary far end remains fixed only in the apophatic sense.

Definition 9463.8 (Hyperbolic expansion reading)**.** *A focused ledger has a* hyperbolic expansion reading *when its visible record growth is represented by a path or family of paths in a disk surface whose hyperbolic distance from earlier readouts increases with local* Cont *extension:*

$$\Theta_i : \mathsf{e} \rightsquigarrow H(i), \qquad B(H(i), t) \in \mathbb{D}, \qquad \delta(B(H(i), t_0), B(H(i), t_1)) \text{ grows along displayed extension.}$$

This is a representation condition. The object-level time remains the local Cont *chain, and the disk metric is not a global clock.*

Principle 9463.9 (Explosion is records-side expansion)**.** *The expansion image is admissible only as records-side growth:*

$$\mathsf{Hist}\ \textit{extension} \quad \rightsquigarrow \quad \textit{more visible directions, fibers, relations, and invariants.}$$

It is not a derivation of the empirical big-bang model, not a completed spacetime metric, and not a claim that all observers are synchronized by one radial coordinate.

9463.5 Disk dynamics and the fixed far end

Ordinary rotations of the complex plane fix an interior center. That picture is too strong for BEDC's apophatic boundary. The better geometric schema is disk-preserving or hyperbolic transport whose visible points move while a boundary far-end commitment is preserved.

Definition 9463.10 (Boundary-fixed disk transport)**.** *A* boundary-fixed disk transport *is a reversible representation row γ on a disk surface such that:*

1. *γ transports displayed disk points by classifier-preserving witnesses;*
2. *γ preserves the provenance rows attached to those points;*
3. *the visible points may change under γ;*
4. *the fiber far-end reading remains the same apophatic boundary name:*

$$\mathrm{FarEnd}(\mathrm{Fiber}(x)) \equiv_{\mathrm{apo}} \mathsf{T} \quad \Rightarrow \quad \mathrm{FarEnd}(\mathrm{Fiber}(\gamma x)) \equiv_{\mathrm{apo}} \mathsf{T}.$$

In classical disk dynamics one may distinguish elliptic, hyperbolic, and parabolic behavior by the placement of fixed points. BEDC consumes only the schema that matters for the ledger: interior fixedness is not required, while boundary far-end fixedness may be used as an expressive representation of the T-socket discipline. A Denjoy-Wolff-style boundary image is therefore closer to the intended reading than a simple Euclidean rotation, but it remains a vision-level analogy until its disk map, metric, and boundary rows are certified.

Principle 9463.11 (No physical constant from geometry alone)**.** *Neither unitary transport nor hyperbolic disk geometry supplies the empirical value of a physical constant. A light-speed-like invariant must still pass through the inter-*Hist *dependency and maximum-rate route of chapter 9419. Geometry may display the invariant shape; it does not certify the empirical constant.*

9463.6 Consciousness fixed point as inscription

The geometric layer also clarifies the self-center language. The focused record sits on the visible surface. The self-center is the inscription-side reading of that local event. The apophatic fixed point is the common far end behind the surface's fiber. These are three roles, not three objects.

Definition 9463.12 (Hyperbolic self-center reading)**.** *For a focused ledger $\mathcal{L}_N$ with focus F, a disk readout $B(H(F), t)$, and a far-end boundary reading T, the* hyperbolic self-center reading *is the aspectual assignment*

$$\mathrm{SelfCenter}(\mathcal{C}_F, t) := \mathrm{InscriptionPoint}(\mathsf{T}, \mathcal{C}_F, t),$$

together with the displayed records-side row $B(H(F), t)$. The self-center is not $B(H(F), t)$, not T, and not a hidden subject behind $H(F)$.

Concretized in §4011: The hyperbolic self-center reading is realized as finite disk-readout, inscription, boundary, transport, provenance, and noncollapse NameCert rows.

Principle 9463.13 (Three-role discipline)**.** *The geometric reading must keep the roles separate:*

$$\begin{aligned} B(H(F),t) &= \textit{visible disk or phase readout;} \\ \mathrm{Fiber}(B(H(F),t)) &= \textit{hidden provenance side recorded by } \mathsf{GAP}; \\ \mathsf{T} &= \textit{apophatic far-end boundary name;} \\ \mathrm{SelfCenter}(\mathcal{C}_F,t) &= \textit{local inscription of that far end on the focused chain.} \end{aligned}$$

Collapsing any two of these roles into object-level equality violates the no-subject and no-internal-T *discipline.*

9463.7 Closure of the geometric layer

The geometric layer is admissible when it is read as a representation surface over already displayed rows:

$$\begin{aligned} \mathsf{Complex}^{\uparrow} &\rightsquigarrow \text{phase readout;} \\ \mathsf{UnitDisk}^{\uparrow} &\rightsquigarrow \text{disk surface and boundary consumer;} \\ \mathsf{UnitaryGroup}^{\uparrow} &\rightsquigarrow \text{reversible classifier-preserving transport;} \\ \mathsf{DynSystem}^{\uparrow} &\rightsquigarrow \text{finite flow-style route;} \\ \mathsf{GAP} &\rightsquigarrow \text{fiber and provenance conservation;} \\ \mathsf{T} &\rightsquigarrow \text{apophatic boundary far end only.} \end{aligned}$$

Its BEDC-safe slogan is:

hyperbolic expansion on the visible surface, reversible conservation in the ledger, apophatic fixedness at the fiber

The chapter claims no stronger closure. A concrete Poincare-disk or hyperbolic-metric certificate would require displayed disk-map rows, metric-read rows, boundary rows, classifier stability, provenance, and refusal rows before it could be consumed by a theorem.

Concretized in §4185: The hyperbolic phase observer-expansion packet realizes the focused-ledger, shadow-metric, boundary-shadow, Cont-growth, provenance, and nonescape route as finite NameCert data.

Concretized in §6569: The horoball-shadow packet realizes the boundary-level hyperbolic observer-expansion route as finite Busemann, visual-boundary, metric, transport, provenance, and Real-seal rows.

Concretized in §7843: HyperbolicHoroballUp realizes boundary-point, Busemann-threshold, visual-shadow, HyperbolicMetric, and Real-seal rows as finite NameCert data.

Concretized in §5159: The hyperbolic isometry group packet realizes disk-map, inverse, hyperbolic-metric preservation, boundary action, transport, provenance, and NameCert rows as finite data.

Concretized in §6595: Boundary-fixed disk transport is realized as finite disk-map, boundary, transport, provenance, and noncollapse NameCert rows.

Concretized in §8460: The boundary-shadow cross-ratio packet realizes the shared boundary-shadow, cross-ratio, visual-metric, Busemann, and observer-expansion route as finite NameCert data.

Concretized in §7487: The horosphere-shear ledger realizes boundary-point, horosphere, Busemann, metric-read, transport, and Real-seal rows as finite NameCert data.

Concretized in §6142: The boundary-far-end noncenter packet realizes the fixed-boundary refusal that prevents a disk readout from becoming an interior center.

Concretized in §6470: The Gromov-boundary packet realizes ray-equivalence, Gromov-product, visual-metric, Busemann-cocycle, boundary-shadow, transport, provenance, and NameCert rows as finite data.

Concretized in §6544: Hyperbolic ideal triangles realize the disk-boundary fixed-end route as finite boundary, geodesic, metric, and flow rows.

Concretized in §6609: The visual-flow shadow packet realizes the joint visual metric, geodesic-flow, boundary-shadow, observer-expansion, and Real-seal route as finite NameCert data.

Concretized in §6700: The horocycle-return ledger realizes the boundary, Busemann, visual-metric, horosphere-flow, and transport rows as finite NameCert data.

Concretized in §6196: The hyperbolic set packet realizes displayed metric, boundary, transport, and provenance rows as finite NameCert data for the observer-expansion surface.

Concretized in §6831: The finite observation topology packet records the observer-expansion window as a concrete NameCert carrier.

Concretized in §7031: The hyperbolic phase ledger packet realizes boundary-shadow, observer-expansion, phase-shell, disk-transport, refusal, provenance, and local NameCert rows as finite data.

Concretized in §8598: Hyperbolic phase expansion cocycles realize composable observer-phase transport, phase-distance, boundary-shadow, and transport-budget rows as finite NameCert data.

Concretized in §7056: The hyperbolic shear-coordinate packet realizes boundary-fixed disk transport and observer-expansion comparison as finite NameCert rows.

Concretized in §7112: The observer phase route is realized as finite boundary-shadow, visual-boundary, phase-distance, reversible-transport, refusal, provenance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §7134: The hyperbolic phase-transport budget packet realizes disk metric, hyperbolic metric, observer-phase groupoid, energy ledger, curvature-budget, Real-seal, transport, provenance, and NameCert rows as finite data.

Concretized in §7148: The finite hyperbolic shadow-cover route is realized as boundary, visual-metric, shadow-window, Real-seal, transport, provenance, and NameCert rows.

Concretized in §7199: Hyperbolic convex disk packets realize the finite geodesic-window convexity route over Poincare disk metric, hyperbolic metric, boundary transport, StreamName windows, Real sealing, provenance, and NameCert rows.

Concretized in §7231: The boundary-distortion budget packet realizes disk-map, boundary-shadow, visual-metric, dyadic tolerance, Real-seal, transport, provenance, and NameCert rows for the observer-expansion vision.

Concretized in §7244: Poincare horocycle transport is realized as finite disk, boundary-shadow, horocycle, transport, Real-seal, provenance, and NameCert rows.

Concretized in §7263: Poincare disk isometries realize reversible hyperbolic disk transport as finite disk-map, metric-preservation, inverse, boundary, provenance, and NameCert rows.

Concretized in §7434: The finite phase-cylinder packet realizes phase readout, boundary circle, reversible transport, provenance, and refusal rows as NameCert data.

Concretized in §7447: The Hadamard boundary packet realizes CAT(0)/Hadamard boundary readback through finite geodesic-ray, Busemann, visual-boundary, metric, and Real-seal rows.

Concretized in §7515: The visual-metric modulus packet realizes boundary-shadow, dyadic tolerance, Real-seal, and observer-expansion rows as finite NameCert data.

Concretized in §7560: Boundary pairing is realized as finite paired boundary rows consumed before cross-ratio and geodesic-flow handoffs.

Concretized in §7595: The Cartan angular boundary invariant packet realizes ordered boundary triples, cross-ratio phase readback, Mobius stability, and provenance rows as finite hyperbolic phase data.

Concretized in §7624: The finite horoball-shadow ledger realizes Busemann, visual-shadow, boundary, metric, observer-window, transport, provenance, and NameCert rows as hyperbolic observer-expansion data.

Concretized in §7725: The visual-angle packet realizes boundary triple, hyperbolic metric, cross-ratio, Real-seal, transport, provenance, and NameCert rows as finite hyperbolic phase data.

Concretized in §7801: The Poincare disk horocycle-return packet realizes boundary-shadow, disk-metric, horocycle transport, return-window, Real-seal, and provenance rows as finite NameCert data.

Concretized in §7806: The Poincare half-plane packet realizes upper-half-plane coordinates, boundary rows, hyperbolic metric reads, Mobius transport, Real sealing, provenance, and NameCert rows as finite data.

Concretized in §7864: Finite uniform-tail packets realize the observer-expansion tail route as StreamName, RegSeqRat, UniformModulus, and Real-seal NameCert data.

Concretized in §7916: The Poincare disk area-element packet realizes local hyperbolic area readback as finite disk, metric, dyadic-radius, Real-seal, and observer-expansion rows.

Concretized in §8089: The hyperbolic Markov partition packet realizes the observer-expansion symbolic-shadow route as finite partition, transition, boundary, metric, transport, provenance, and NameCert rows.

Concretized in §8098: Hyperbolic symbolic coding is realized as finite Markov-partition, geodesic-flow, subshift-code, boundary-shadow, transport, provenance, and NameCert rows.

Concretized in §8140: The hyperbolic operator-trace packet realizes reversible disk transport, operator trace, boundary refusal, Real-seal readout, provenance, and NameCert rows as finite data.

Concretized in §8162: The holonomy-index packet realizes closed phase transport classes as finite disk, boundary, phase-holonomy, transport, provenance, and NameCert rows.

Concretized in §8196: PhaseCurvatureTransportUp realizes the hyperbolic phase-geometry curvature-transport route through finite disk, metric, boundary, phase, curvature-budget, provenance, hsame, Cont, Pkg, and NameCert rows.

Concretized in §8256: The boundary-extension packet realizes the hyperbolic observer-expansion route through finite visual-boundary and boundary-shadow transport rows.

Concretized in §8296: The Martin boundary packet realizes the hyperbolic observer-expansion boundary route through finite transition-kernel, positive-harmonic, shadow, Poisson-kernel, and Real-seal rows.

Concretized in §8323: Hyperbolic bigon thinness is realized as finite endpoint, geodesic-side, hyperbolic-metric, boundary-shadow, and Real-seal NameCert data.

Concretized in §8445: Ford circles realize the Farey rational and hyperbolic horoball boundary branch as finite NameCert data.

Concretized in §8490: The observer horizon-cocycle packet realizes the boundary-shadow, Busemann-cocycle, phase-distance, transport, provenance, and refusal rows as finite NameCert data.

Concretized in §8534: The hyperbolic phase visual-modulus packet realizes observer-expansion visual metric and boundary-shadow tolerance rows as finite NameCert data.

Concretized in §8596: The hyperbolic area-defect packet realizes boundary-triangle area readback through Poincare disk area-element, ideal-triangle, metric, and Real-seal rows.

Concretized in §8640: The boundary phase-shadow route is realized as finite Poincare-disk, hyperbolic-metric, boundary-shadow, observer-phase, and refusal-ledger NameCert data.

Concretized in §8671: The phase-energy transport packet realizes reversible phase routing, energy-ledger readback, transport budget, Real sealing, provenance, and local NameCert rows as finite hyperbolic phase data.

Concretized in §8721: Boundary cross-ratio phase data are realized as finite HyperbolicMetric, visual-boundary, Busemann-boundary, boundary-shadow, Real, transport, replay, provenance, and NameCert rows.

Concretized in §8959: The observer phase transport packet realizes reversible disk-phase, boundary-shadow, visual-boundary, provenance, and refusal rows as finite NameCert data.

Concretized in §8044: Vision ledger routes realize the top-level visions hub edge as finite source-hub, input-route, concrete-target, marker, audit-readback, transport, replay, provenance, and NameCert rows.

Concretized in §8290: Dyadic uniform cover rows realize the finite-cover direction for L10 compactness and completion routes.

Concretized in §8833: Dyadic uniform cover-transfer rows realize the L10 route from displayed dyadic cover cells to StreamName, RegSeqRat, Real, interval-net, compact-metric, transport, replay, provenance, and NameCert rows.

Concretized in §8558: The residual join envelope realizes the visions hub route through finite MetaCIC residual-window, parallel-diamond, candidate-normalization, confluence-budget, replay, provenance, and NameCert rows.

Concretized in §8917: Real completion tail-lock rows realize the finite StreamName, RegSeqRat, DyadicRatCore, and Real seal route for completion-facing reads.

Concretized in §9099: Stream shuffle rows realize finite StreamName recombination through parity-indexed windows, RegSeqRat readback, and Real-seal handoff.

Part XII

Open Conjectures

Chapter 9464

Open Conjectures

The conjectures listed in this directory are BEDC kernel-formulated open problems: each is stated entirely in terms of BHist / NameCert / hsame / Cont / Pkg primitives, has an obvious-but-unproved candidate path, and admits a finite test (verification, weakening, or counter-example).

The pipeline picks one open conjecture per round (kind = `conjecture_chase`) and attempts: (a) closing it with a proof, (b) weakening it by adding hypotheses or restricting scope, or (c) refuting it with a counter-witness drawn from existing concrete instances. Each attempt updates the conjecture's `\conjectureStatus{...}` row and may move the chapter between `open`, `weakened`, `closed`, or `refuted`.

9464.1 Finite-BHist-Injection Admits StdBridge

Conjecture 9464.1 (Finite BHist Tuple Route Admits StdBridge)**.** *For every BEDC chapter* $\langle X\rangle$Up *equipped with a fixed finite* BHist *tuple presentation that*

1. *displays a tuple carrier* $(b_0, b_1, \ldots, b_k)$ *of accepted* BHist *component rows;*
2. *supplies an image row showing that every accepted carrier row is one of the displayed tuple components;*
3. *supplies a readback row recovering the corresponding* BHist *component from each displayed image row;*
4. *presents the chapter carrier as a fixed-arity* BHist *tuple* $(b_0, b_1, \ldots, b_k)$ *whose* NameCert *rows are transported componentwise along those readback rows;*
5. *inhabits* ChapterTasteGate *for the displayed tuple presentation;*

there exists a $\langle X\rangle$UpStdBridge *theorem in* BEDC.*Derived*.$\langle X\rangle$Up *that bridges the displayed finite tuple's* NameCert *obligation surface to the standard reflection schema.*

Status: `weakened`

Status notes. The obligation is the fixed finite BHist tuple route. The bridge follows by case analysis over the displayed tuple carrier, replaying the image row and readback row for each component, and transporting the tuple carrier's NameCert rows component by component before packaging the finite family as the chapter's standard reflection bridge.

The remaining open part is the general transport from arbitrary regular-language recognisers to BHist tuple presentations. That route would need a recogniser-to-carrier construction and a uniform component transport theorem not supplied by the fixed tuple test target.

Active fixed-tuple test. The active finite test is exactly the fixed-tuple kernel route: a tuple carrier, an image row, a readback row, componentwise NameCert transport, and the ChapterTasteGate row. Recognizer-to-carrier construction is not part of this test and remains a separate boundary condition.

The exact remaining target is recognizer-to-carrier construction. It must produce a recognizer acceptance row, finite tuple extraction, BHist component readback, componentwise NameCert transport, and ChapterTasteGate from the recognizer presentation before any stronger StandardBridge conjecture is reopened.

Conjecture 9464.2 (Finite BHist injection recognizer-to-carrier route)**.** *A regular recognizer-side input lands on the finite packet test only after it displays seven kernel rows as named carrier data:*

1. *a recognizer acceptance row through* GroundCompilerRecognizerFlow$^{\uparrow}$*;*
2. *a generated-recognizer row recording the recognizer presentation used for the finite test;*
3. *a certificate-gate row admitting that presentation to the BEDC packet surface;*
4. *a finite tuple extraction row through* FiniteReflectionTuple$^{\uparrow}$*;*
5. *a* BHist *component readback row for each extracted component;*
6. *componentwise* hsame *and* NameCert *transport rows through the displayed component reads;*
7. *a* ChapterTasteGate *row.*

These rows must live inside the existing BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *support surface. The conjecture asks only for this finite packet landing test; it does not assert a* StandardBridge *theorem and does not construct a recognizer-to-carrier map.*

Status: `weakened`

Recognizer-to-carrier status. The active conjecture is the seven-row closed landing test together with one remaining carrier-map compatibility row, not a StandardBridge construction. The decomposition of Definition 9464.45, the exhaustion theorem Definition 9464.46, and the closed landing test of Definition 9464.48 identify exactly the recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, and ChapterTasteGate rows inside the BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, and FiniteReflectionTuple$^{\uparrow}$ kernel surface. The remaining finite object is only a carrier-map compatibility row preserving those rows through recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist component readback, componentwise hsame and NameCert transport, and ChapterTasteGate. No fixed-tuple StandardBridge theorem is claimed here.

Theorem 9464.3 (Finite BHist injection local carrier-map status)**.** *After the decomposition, row-exhaustion, and closed-landing tests have displayed the seven recognizer rows, the remaining recognizer-to-carrier object is the local test that the carrier map preserves exactly those rows. It is not a* StandardBridge *theorem, not a fixed-tuple bridge completion, and not a construction of arbitrary regular-language carriers.*

Proof. Read the decomposition of Definition 9464.45, the row exhaustion theorem Definition 9464.46, and the closed landing test Definition 9464.48. They name the seven displayed recognizer rows and leave no hidden landing row.

The status paragraph then identifies the only remaining object: preservation of those rows by the carrier map. Since the object is local preservation inside the displayed kernel support surface, it does not assert a StandardBridge theorem or a fixed-tuple bridge construction. □

Theorem 9464.4 (Finite BHist injection recognizer carrier-map row closure)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, the weakened conjecture Definition 9464.2 has exactly one remaining open object: the recognizer carrier-map compatibility row preserving those displayed data inside the* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *support surface. The statement does not assert a* StandardBridge *construction and does not construct arbitrary regular-language carriers.*

Proof. Use the recognizer-to-carrier status paragraph. It separates the seven landing rows from the carrier-map compatibility row and keeps both inside the displayed BEDC kernel support surface.

Apply the recognizer carrier-map decomposition, Definition 9464.45. It lists the recognizer acceptance, generated-recognizer mediation, certificate-gate admission, tuple extraction, component readback, componentwise transport, and taste-gate rows as the full route to the fixed tuple surface.

Now use row exhaustion, Definition 9464.46, together with the closed landing test, Definition 9464.48. Once those rows are displayed, no further landing row remains to be supplied.

Finally read the carrier-map compatibility boundary already isolated by Definition 9464.22. The remaining object is exactly preservation of the same seven displayed rows by the carrier map, and no standard bridge theorem is claimed. □

Theorem 9464.5 (Finite BHist injection carrier-map obstruction update)**.** *Mode: restate. The weakened conjecture of Definition 9464.2 is unchanged: after recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, the only remaining positive object is one carrier row preserving those same data inside the existing* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *support surface. The statement does not assert* StandardBridge*, construct arbitrary regular recognizer carriers, or identify quotient carriers by host equality.*

Proof. Read the recognizer-to-carrier status paragraph. It already separates the seven displayed landing rows from the single carrier-map compatibility row and leaves the conjecture in weakened status.

Apply the recognizer carrier-map row closure, Definition 9464.4. The row closure fixes the data that must be preserved: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and taste-gate admission.

Finally use the carrier-map forward compatibility boundary, Definition 9464.22. It permits only forward preservation of those rows into the fixed tuple surface. Therefore the remaining positive object is one carrier row with that preservation property, and no StandardBridge theorem is asserted. □

Theorem 9464.6 (Finite BHist injection carrier-map compatibility closure)**.** *Mode: restate. A carrier-map compatibility proof for Definition 9464.2 must preserve exactly the seven displayed*

rows: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, FiniteReflectionTuple$^{\uparrow}$ *extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *along the displayed carrier map. The closure does not assert* StandardBridge*, does not construct recognizer-to-carrier data for arbitrary regular languages, and does not identify quotient carriers by equality.*

Proof. Use the recognizer-to-carrier status paragraph together with the decomposition of Definition 9464.45. It isolates the carrier-map question from the fixed tuple bridge and names the seven rows that must survive the map.

Apply row exhaustion, Definition 9464.46, and the closed landing test, Definition 9464.48. These results leave no additional row between the recognizer presentation and the finite tuple surface.

Finally use carrier-map forward compatibility, Definition 9464.22. The compatibility row can only preserve the displayed recognizer, certificate, tuple, readback, transport, and taste-gate data, so the statement closes the remaining object without claiming a standard bridge, arbitrary regular-language carrier construction, or quotient carrier equality. □

Theorem 9464.7 (Finite BHist injection carrier-map compatibility boundary)**.** *Mode: restate. Once recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, the only remaining weakened claim in Definition 9464.2 is compatibility of the carrier map with those seven displayed rows. The boundary remains a finite carrier-map compatibility target inside the* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *support surface; it is not a* StandardBridge *theorem.*

Proof. Read the recognizer-to-carrier status paragraph. It names the seven displayed rows and separates them from the carrier-map compatibility row that still has to preserve the same data.

The decomposition, row-exhaustion, and closed-landing statements identify the rows as recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, and ChapterTasteGate on one finite route.

Therefore, after those rows are displayed, no stronger bridge-facing object is left in the weakened claim. What remains is precisely compatibility of the carrier map with those seven rows, inside the listed kernel support surface. □

Theorem 9464.8 (Finite BHist injection carrier-map status refinement)**.** *Mode: restate. The weakened boundary of Definition 9464.2 is not the seven landing rows themselves once those rows are displayed. The active boundary is the single carrier-map compatibility row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *along one displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *route. The status remains weakened: no* StandardBridge *theorem and no fixed-tuple bridge construction are asserted.*

Proof. Read the recognizer-to-carrier status paragraph. It separates the seven displayed landing rows from the remaining carrier-map compatibility row, and it keeps all of them inside the named BEDC kernel support surface.

Apply the recognizer carrier-map decomposition, Definition 9464.45, together with row exhaustion, Definition 9464.46. These results locate recognizer acceptance, generated-recognizer

mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and the taste-gate row on one displayed route.

Finally use the closed landing test, Definition 9464.48. Once the landing rows are present, the remaining row to inspect is compatibility of the carrier map with those same displayed data. That restates the weakened boundary without asserting a StandardBridge theorem or constructing the fixed-tuple bridge route. □

Theorem 9464.9 (Finite BHist injection status tightening)**.** *Mode: restate. The weakened recognizer-to-carrier conjecture is exactly the seven-row recognizer landing test through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, certificate-gate admission,* FiniteReflectionTuple$^{\uparrow}$ *extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*, together with the carrier-map compatibility row that preserves those displayed data. It does not assert* StandardBridge *construction and does not upgrade the fixed-tuple test into a bridge theorem.*

Proof. Read the status paragraph and the conjecture statement together. Both name the same seven landing rows: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and the taste gate.

The later decomposition, row-exhaustion, and closed-landing references locate these rows inside the existing BEDC kernel surface. The only finite row still isolated there is carrier-map compatibility: it must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist component readback, componentwise hsame and NameCert transport, and ChapterTasteGate.

Therefore the current weakened status is precisely the landing test plus that compatibility row. Any statement requiring a StandardBridge theorem or bridge-facing fixed-tuple construction lies outside the conjecture. □

Theorem 9464.10 (Finite BHist injection carrier-map active target)**.** *Mode: restate. The only active positive target for Definition 9464.2 is carrier-map compatibility preserving the seven displayed recognizer rows: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. This target stays inside the* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *support surface; it does not assert* StandardBridge*, arbitrary recognizer-to-carrier construction, quotient carrier equality, or host recognizer equality.*

Proof. Read the recognizer-to-carrier status paragraph and the carrier-map row closure, Definition 9464.4. Together they separate the seven displayed landing rows from the single carrier-map compatibility row that must preserve them.

Apply carrier-map compatibility boundary and status tightening, Definition 9464.7 and Definition 9464.9. These results keep the weakened conjecture at the landing test plus the carrier-map compatibility row.

Therefore no stronger positive target is active in this section. A StandardBridge theorem, arbitrary regular-language carrier construction, quotient carrier equality, or host recognizer comparison would change the surface rather than preserve the seven displayed rows. □

Theorem 9464.11 (Finite BHist injection carrier-map positive criterion)**.** *If a displayed carrier map preserves exactly the seven rows named in Definition 9464.2 through recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*, then the weakened carrier-map compatibility row is closed for that packet. The conclusion does not assert* StandardBridge*, arbitrary recognizer-to-carrier construction, or quotient carrier equality.*

Proof. Apply the recognizer carrier-map decomposition, Definition 9464.45. It gives the seven displayed rows that any recognizer route must expose before the fixed tuple surface is tested.

Use row exhaustion and the closed landing test, Definition 9464.46 and Definition 9464.48. These make the seven rows the whole finite landing surface and keep StandardBridge construction outside the conclusion.

Now read the assumed carrier map. Its preservation of recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, and ChapterTasteGate is exactly the compatibility row isolated by Definition 9464.8.

Finally apply carrier-map forward compatibility and status tightening, Definition 9464.22 and Definition 9464.9. The weakened row is closed for this packet, while arbitrary recognizer-to-carrier construction, quotient carrier equality, and a StandardBridge theorem remain outside the asserted surface. □

Theorem 9464.12 (Finite BHist injection StdBridge nonclaim boundary)**.** *Mode: restate. Displaying the seven recognizer landing rows of Definition 9464.2 is only a precondition for inspecting the fixed tuple route. It neither asserts nor refutes any* StandardBridge *theorem, and it does not construct a recognizer-to-carrier map.*

Proof. Read the fixed tuple route of Definition 9464.24. It requires the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row before the fixed tuple surface can be inspected.

Next apply the readback gap refutation of Definition 9464.17. A packet missing component readback does not land on the seven-row surface, so it is not a refutation of the weakened landing test and not a proof of StandardBridge.

Finally use the closed landing test of Definition 9464.48. It identifies the seven displayed rows as the finite landing boundary and explicitly leaves both the StandardBridge theorem and the recognizer-to-carrier construction outside the conclusion. □

Theorem 9464.13 (Finite BHist injection recognizer-to-carrier sharp obstruction)**.** *Mode: refute. A recognizer input that displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, and finite tuple extraction, but does not display* BHist *component readback, cannot land on the fixed tuple route of Definition 9464.1. Such an input is neither a counterexample to the weakened fixed-tuple conjecture nor a proof of it.*

Proof. The fixed tuple route requires a displayed tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row before any bridge-facing comparison is made. The recognizer-side status paragraph sharpens this into seven recognizer landing rows, including explicit BHist component readback after finite tuple extraction.

Now take a recognizer input with acceptance, generated-recognizer mediation, certificate-gate admission, and finite tuple extraction but without displayed component readback. The input has reached only the recognizer pre-landing surface. It has not supplied the readback row that lets a finite tuple component be compared with a fixed BHist component.

Therefore the fixed tuple route is unavailable. The missing readback keeps the packet outside the weakened fixed-tuple test, so it cannot be used as either a negative instance of that route or a positive bridge comparison. □

Theorem 9464.14 (Finite BHist injection readback refutation)**.** *Mode: refute. A recognizer-side packet with recognizer acceptance, generated-recognizer mediation, certificate-gate admission,*

and finite tuple extraction, but with no BHist *component readback row, refutes only the stronger omitted-readback landing variant. It is not a counterexample to the weakened fixed-tuple conjecture of Definition 9464.1 and not a counterexample to the seven-row landing test of Definition 9464.2.*

Proof. Read the seven-row landing list in Definition 9464.2. The fifth row is explicit BHist component readback, and the sixth row transports hsame and NameCert data through those displayed component reads.

Apply Definition 9464.13. A packet without component readback has reached only the recognizer pre-landing surface, so it has not entered the weakened fixed-tuple route and cannot count as a negative instance of that route.

Thus the packet refutes the overstrong landing form that omits readback from the recognizer-side requirements. The weakened conjecture remains unchanged because its test surface still requires the readback row before any fixed-tuple or carrier-facing obstruction is evaluated. □

Definition 9464.15 (Finite BHist injection readback gap)**.** *A finite* BHist *injection readback gap is a recognizer-side packet that displays recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, certificate-gate admission, and finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$ *but displays no* BHist *component readback row for the extracted tuple components. Such a packet has not displayed componentwise* hsame *or* NameCert *transport through component reads and has not reached the* ChapterTasteGate *row of the seven-row landing test.*

Theorem 9464.16 (Finite BHist injection open kernel sharp restatement)**.** *Mode: restate. Any proposed* StandardBridge *for the finite* BHist *injection recognizer surface must expose a row-level map from recognizer evidence to carrier evidence: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row must all be displayed before the fixed tuple route is tested. If the packet is a readback gap in the sense of Definition 9464.15, then the bridge request is refused at the kernel-row surface rather than evaluated as a* StandardBridge *instance.*

Proof. Read the recognizer-side landing list in Definition 9464.2. It is already a row-level map from recognizer evidence to carrier evidence: the recognizer-flow rows precede finite tuple extraction, and the extracted tuple becomes carrier-facing only after component readback and componentwise transport are displayed.

Now apply the sharp obstruction theorem, Definition 9464.13. A packet with recognizer acceptance, certificate-gate admission, and tuple extraction but without component readback has not reached the fixed tuple route. The readback refutation row, Definition 9464.14, assigns such a packet only to the omitted-readback failure surface.

Finally unpack Definition 9464.15 and use Definition 9464.17. The gap displays exactly the missing-row obstruction: no BHist component readback, no componentwise hsame or NameCert transport through component reads, and no ChapterTasteGate landing. Therefore the proposed bridge is refused before the standard bridge comparison is formed, and the conjecture status is left unchanged. □

Theorem 9464.17 (Finite BHist injection readback gap refutes only omitted-readback bridges)**.** *Mode: refute. A finite* BHist *injection readback gap refutes only bridge requests that omit* BHist *component readback from the recognizer landing surface. It does not refute the weakened seven-row landing test of Definition 9464.2, does not refute the fixed-tuple route of Definition 9464.1, and does not prove a* StandardBridge *theorem.*

Proof. Unpack Definition 9464.15. The packet displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, and finite tuple extraction, but it stops before the component readback row.

Apply Definition 9464.14. Missing readback only blocks the overstrong bridge request that tries to land without the fifth row of the seven-row test.

The weakened conjecture still requires component readback, componentwise hsame and NameCert transport, and ChapterTasteGate before the fixed-tuple surface is evaluated. Therefore the gap names a non-landing surface, not a counterexample to the weakened test and not a proof of StandardBridge. □

Theorem 9464.18 (Finite BHist injection readback closed landing)**.** *Mode: restate. When recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are all displayed, the recognizer-side packet lands on the weakened fixed-tuple test surface of Definition 9464.2. This landing theorem still asserts no* StandardBridge *theorem and constructs no recognizer-to-carrier map.*

Proof. Read the seven-row list in Definition 9464.2. The listed rows supply exactly the recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and taste-gate data needed before the fixed-tuple surface is tested.

Use the sharp obstruction theorem Definition 9464.13 and the readback refutation theorem Definition 9464.14. Together they identify missing readback as failure to land, and they leave the weakened conjecture unchanged once readback and componentwise transport are present.

Thus displaying all seven rows is the positive landing boundary for the weakened fixed-tuple test surface. The conclusion stops at that boundary and does not package a standard bridge or produce a recognizer-to-carrier construction. □

Theorem 9464.19 (Finite BHist injection readback transport closure)**.** *Mode: restate. Once component readback and componentwise* hsame *and* NameCert *transport rows are displayed, adjoining replay and provenance rows cannot move the recognizer-side packet outside the weakened seven-row landing surface of Definition 9464.2. The closure still asserts no* StandardBridge *theorem and constructs no recognizer-to-carrier data.*

Proof. Start from the closed landing theorem Definition 9464.18. Its landing surface already contains the component readback row and the componentwise hsame and NameCert transport row.

The replay and provenance rows are support rows for the same displayed finite packet. They record that the already displayed component reads can be replayed and sourced inside Cont and Pkg; they do not replace recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, transport, or the ChapterTasteGate row.

Therefore adding replay and provenance preserves the weakened seven-row landing surface. The argument stops at that landing boundary and does not produce a standard bridge, a recognizer-to-carrier construction, a quotient carrier replacement, or a host regular-language carrier theorem. □

Theorem 9464.20 (Finite BHist injection recognizer landing sharpener)**.** *Any recognizer-side attempt to inspect the fixed tuple route of Definition 9464.2 must first display recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. These rows are a recognizer landing boundary only: they do not supply a* StandardBridge *theorem, do not construct a recognizer-to-carrier map, and do not assert arbitrary regular-language carrier claims.*

Proof. Read the active conjecture as a finite landing test. Its seven displayed rows are exactly recognizer acceptance, generated-recognizer data, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and the taste gate.

Use the recognizer-to-carrier status paragraph. It keeps those rows inside the existing BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, and FiniteReflectionTuple$^{\uparrow}$ support surface before the fixed tuple route can be inspected.

Thus the recognizer-side attempt lands only after the seven rows have been exposed. Since the displayed surface stops before carrier construction and before the standard bridge schema, no StandardBridge, recognizer-to-carrier construction, or arbitrary regular-language carrier theorem is obtained. □

Theorem 9464.21 (Finite BHist injection open-status sharpening)**.** *The remaining recognizer-side claim for Definition 9464.2 is exactly the landing surface that passes through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, the certificate gate,* FiniteReflectionTuple$^{\uparrow}$ *extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. It is a finite BEDC-kernel landing test and not a* StandardBridge *theorem, not a recognizer-to-carrier construction, and not an ambient regular-language carrier theorem.*

Proof. Start from the active recognizer-to-carrier conjecture. Its enumerated rows list recognizer acceptance, generated-recognizer data, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and the taste gate.

Use the recognizer-to-carrier status paragraph together with the decomposition, row-exhaustion, and closed landing tests already cited there. These isolate the same rows inside the BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, and FiniteReflectionTuple$^{\uparrow}$ support surface.

Therefore the remaining claim is precisely the recognizer landing boundary. Since no row in the statement constructs a carrier map or packages a standard bridge, the conjecture remains a finite kernel test target rather than a completed StandardBridge route. □

Theorem 9464.22 (Finite BHist injection carrier-map forward compatibility)**.** *The remaining weakened claim for Definition 9464.2 is forward compatibility of the carrier map with the seven displayed rows: recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, certificate-gate admission,* FiniteReflectionTuple$^{\uparrow}$ *extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. A compatible carrier map may only move these rows forward into the fixed tuple surface; it does not supply a* StandardBridge *theorem, construct arbitrary regular-language carriers, or identify quotient syntax by equality.*

Proof. Use Definition 9464.21. It identifies the live recognizer-side surface with the seven finite BEDC kernel rows and explicitly keeps the target away from StandardBridge and ambient carrier construction.

Then read Definition 9464.23. The rows are recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and taste-gate admission. These are exactly the rows a forward carrier-map compatibility check can preserve.

Thus the only unclosed direction is the carrier-map row that transports those displayed recognizer-side rows into the fixed tuple surface. The statement refuses a standard bridge, arbitrary regular-language carrier construction, and quotient syntax equality; it asks only for forward compatibility of the named finite rows. □

Theorem 9464.23 (Finite BHist injection kernel row restatement)**.** *Any admissible recognizer-side landing for the fixed tuple route of Definition 9464.2 is exactly the following seven displayed BEDC kernel rows:*

1. *recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*;*
2. *generated-recognizer mediation;*
3. *certificate-gate admission;*
4. *finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$*;*
5. BHist *component readback;*
6. *componentwise* hsame *and* NameCert *transport;*
7. *the* ChapterTasteGate *row.*

Passing these rows lands on the finite kernel test surface only. It does not assert a StandardBridge *theorem, a recognizer-to-carrier map, or an arbitrary regular-language carrier construction.*

Proof. Use the recognizer carrier-map decomposition of Definition 9464.45. It decomposes any recognizer-side attempt into the recognizer-flow rows, tuple-extraction row, component readback row, componentwise transport row, and taste-gate row.

Then apply the row-exhaustion theorem Definition 9464.46. Once the seven rows are displayed, the recognizer input has no further kernel landing row to expose before the fixed tuple test can be read.

Finally read the closed landing test of Definition 9464.48. That result keeps the surface at the finite landing test and explicitly leaves both the StandardBridge theorem and the recognizer-to-carrier construction outside the conclusion. □

Theorem 9464.24 (Finite BHist injection fixed tuple route)**.** *In the fixed finite tuple case, every candidate* StandardBridge *route must be a displayed* BHist *tuple route. The image row, readback row, tuple carrier, componentwise* NameCert *transport, and* ChapterTasteGate *row must all be present before the bridge is requested.*

Proof. Start with the displayed tuple carrier $(b_0, b_1, \ldots, b_k)$. The image row restricts every accepted carrier read to one of these finite BHist components, and the readback row returns the corresponding component as kernel data.

Transport is then componentwise. Each NameCert obligation is replayed along the displayed readback row for its component, and the ChapterTasteGate row records that this tuple presentation is the one accepted by the chapter.

This isolates only the fixed-tuple route. It does not construct a tuple carrier from an arbitrary regular-language recogniser, so the general recogniser-to-carrier problem remains open. □

Proposition 9464.25 (Finite BHist injection fixed tuple obligation surface)**.** *Before any* StandardBridge *target is requested in the fixed finite tuple case, the chapter must display the tuple carrier, the image row, the readback row, the componentwise* NameCert *transport rows, and the* ChapterTasteGate *row as public preconditions. These rows are the exact obligation surface for the fixed-tuple route, and they do not supply the missing construction from arbitrary regular-language recognisers to tuple carriers.*

Proof. Apply Definition 9464.24. It already requires the displayed BHist tuple carrier, the image row, the readback row, componentwise NameCert transport, and the ChapterTasteGate row before the bridge is requested.

The proposition only repacks that route as an obligation surface. It preserves the open recogniser-to-carrier problem because no row here constructs a tuple carrier from an arbitrary regular-language recogniser. □

Theorem 9464.26 (Finite BHist injection recognizer boundary)**.** *A recognizer-to-carrier route can be compared with the fixed-tuple* StandardBridge *route only after it displays a recognizer acceptance row, a finite tuple extraction row, a readback row into* BHist *components, componentwise* NameCert *transport, and a* ChapterTasteGate *row. These five rows form the boundary between arbitrary regular-language recognizers and the fixed finite tuple obligation surface.*

Proof. Apply Definition 9464.24. It gives the fixed tuple, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row required before the fixed-tuple bridge route is requested.

Now use Definition 9464.25. It records the same rows as the public obligation surface and keeps them separate from the construction of tuple carriers from arbitrary recognizers.

Thus a recognizer-side comparison must first expose the acceptance row and the finite tuple extraction row that land in that surface. This theorem only names that boundary; it does not construct the bridge or solve the recognizer-to-tuple transport problem. □

Theorem 9464.27 (Finite BHist injection regular recognizer weakening)**.** *A regular recognizer input is admissible for the fixed tuple* StandardBridge *route only after it exposes a recognizer acceptance row, a finite tuple extraction row, a* BHist *component readback row, componentwise* NameCert *transport, and a* ChapterTasteGate *row. These rows place the recognizer-derived input on the same obligation surface as the fixed tuple route, but they do not construct the recognizer-to-carrier map.*

Proof. Apply the fixed tuple route. It requires the displayed tuple carrier, image row, readback row, componentwise naming transport, and ChapterTasteGate row before the bridge route is requested.

Use the fixed tuple obligation surface to identify those rows as the public preconditions for the route. A regular recognizer input must therefore expose its acceptance row and finite tuple extraction row before it can be compared with the fixed tuple surface.

Finally apply the recognizer boundary theorem. It already separates the required boundary rows from the construction of tuple carriers from arbitrary regular recognizers, so this theorem records admissibility only after the boundary is displayed and does not construct the recognizer-to-carrier map. □

Theorem 9464.28 (Finite tuple StdBridge fixed-route closure)**.** *In the fixed finite tuple case, once the tuple carrier, image row, readback row, componentwise* NameCert *transports, and* ChapterTasteGate *row are displayed, the* StandardBridge *obligation surface is closed for that fixed route. This closure is only the displayed tuple route; it leaves the arbitrary regular-language recogniser-to-carrier problem open.*

Proof. Apply Definition 9464.24. It identifies the fixed tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the whole route that must be present before the bridge is requested.

Use Definition 9464.25. The proposition repacks those rows as the exact public obligation surface, so no extra row is needed for the fixed tuple route once all five displayed rows are present.

Finally apply Definition 9464.29. Any counter-instance at the fixed route must keep the whole displayed tuple surface present and locate an additional reflection-axis obstruction. A missing tuple, image row, readback row, componentwise transport, or taste-gate row is therefore outside the closed fixed-route surface.

The theorem does not solve the arbitrary regular-language recogniser-to-carrier problem. That problem still requires a recogniser acceptance row, tuple extraction row, and uniform component transport beyond the fixed tuple route. □

Theorem 9464.29 (Finite tuple StdBridge refutation candidate)**.** *A valid counter-instance to Definition 9464.1 in the fixed finite tuple case must display all five fixed-tuple rows: the tuple carrier* $(b_0, b_1, \ldots, b_k)$*, the image row, the readback row, the componentwise* NameCert *transport rows, and the* ChapterTasteGate *row. It must also display a reflection-axis requirement for the same chapter that is incompatible with the standard reflection schema consumed by* StandardBridge.

Thus a failed bridge attempt caused only by a missing tuple, missing image row, missing readback row, missing componentwise transport, or missing ChapterTasteGate *row is not a counter-instance to the weakened fixed-tuple route.*

Proof. Start from Definition 9464.24. That theorem identifies the fixed tuple, image, readback, componentwise naming transport, and ChapterTasteGate rows as prerequisites for the route under test.

Now use Definition 9464.25. It repacks the same rows as the exact obligation surface and separates them from the open recogniser-to-carrier problem.

Therefore the negation surface must keep the whole fixed-tuple surface present and locate the obstruction in an additional reflection-axis requirement. A missing prerequisite only shows that the fixed-tuple theorem was not reached; it does not refute the route once its premises are displayed. □

Theorem 9464.30 (Finite BHist injection recognizer restatement)**.** *Any future strengthening from the fixed finite tuple route to regular recognizers must display five kernel rows: a recognizer acceptance row, a finite tuple extraction row, a* BHist *component readback row, componentwise* NameCert *transport, and a* ChapterTasteGate *row. This is a boundary restatement of the remaining recognizer-to-carrier problem. It is not a* StandardBridge *construction and not a recognizer-to-carrier map.*

Proof. Apply the fixed tuple route. It identifies the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the route available before any bridge request.

Use the fixed tuple obligation surface to read those rows as the public kernel boundary for the fixed route. A regular recognizer comparison must therefore add recognizer acceptance and finite tuple extraction before it can land on that surface.

Now apply the recognizer boundary and the regular recognizer weakening theorem. These theorems already separate the five displayed rows from both a bridge construction and a recognizer-to-carrier construction, so the present theorem only restates the boundary that a later strengthening must expose. □

Theorem 9464.31 (Finite BHist injection regular recognizer landing closure)**.** *The remaining regular-recognizer route can close only after five rows are displayed as carrier-facing data: recognizer acceptance, finite tuple extraction,* BHist *component readback, componentwise* NameCert *transport, and* ChapterTasteGate*. Once these rows are visible, the open boundary is the recognizer-to-carrier landing row itself. The statement is not a* StandardBridge *construction and does not assert the fixed-tuple bridge theorem.*

Proof. Start with the fixed tuple route, Definition 9464.24. It supplies the fixed carrier image, readback, componentwise naming transport, and ChapterTasteGate rows required before any bridge-facing comparison.

Use the recognizer boundary and regular recognizer weakening, Definition 9464.26 and Definition 9464.27. They add recognizer acceptance and finite tuple extraction while keeping the comparison outside any constructed recognizer-to-carrier map.

Finally apply the recognizer restatement, Definition 9464.30. It identifies the same five rows as the remaining landing surface and explicitly separates that surface from a StandardBridge construction. Thus closure of the regular-recognizer route requires the five displayed rows first, and only then may the carrier landing row be tested. □

Theorem 9464.32 (Finite BHist injection recognizer counter-boundary)**.** *Any refutation of the regular recognizer strengthening must display a regular recognizer acceptance row, a finite tuple extraction row, a* BHist *component readback row, componentwise* NameCert *transport, and a* ChapterTasteGate *row. It must also exhibit a specific obstruction to the recognizer-to-carrier map after those five rows have been supplied. A failure caused only by absence of one of the five rows is outside the recognizer strengthening boundary.*

Proof. Apply Definition 9464.24. The fixed tuple route already requires the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row before any bridge request is tested.

Use Definition 9464.26. It places a recognizer acceptance row and finite tuple extraction row in front of that fixed tuple surface, separating the regular recognizer input from the carrier route consumed by the fixed-tuple test.

Apply Definition 9464.27 and Definition 9464.30. Together they keep the weakened conjecture at the fixed-tuple surface while naming the remaining recognizer-to-carrier map as the unresolved boundary.

Therefore a counter-boundary must keep all five displayed recognizer rows present and then locate the obstruction in the recognizer-to-carrier map itself. Missing acceptance, extraction, readback, transport, or taste-gate data only shows that the boundary has not been reached. □

Theorem 9464.33 (Finite BHist injection recognizer-row refutation test)**.** *A valid refutation of Definition 9464.2 must first display the seven recognizer landing rows: recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, certificate-gate admission, finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$*,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. Only after those rows are present may a negative result exhibit an obstruction after componentwise NameCert transport. A route missing one of the seven rows is a non-instance of the weakened conjecture surface, not a refutation of the weakened conjecture.*

Proof. Start with the recognizer counter-boundary, Definition 9464.32. It already separates missing recognizer, extraction, readback, transport, and taste-gate data from an obstruction that remains after the recognizer route has reached the carrier-facing surface.

Use the kernel row restatement, Definition 9464.23, together with the closed landing test, Definition 9464.48. These results identify the seven landing rows as the finite BEDC-kernel surface for the weakened recognizer-side test.

Finally apply the status sharpness theorem, Definition 9464.64. After the seven rows are displayed, the remaining open issue is recognizer-to-carrier construction after componentwise NameCert transport. Therefore a negative result before the seven-row landing surface is only failure to enter the test, while a valid refutation must locate a post-transport carrier obstruction. □

Theorem 9464.34 (Finite BHist injection recognizer-to-tuple landing surface)**.** *Any recognizer-side strengthening of Definition 9464.1 must land in the existing* FiniteReflectionTuple$^{\uparrow}$ *surface of chapter 993. Before it can be compared with the fixed finite tuple route, it must pass through the* GroundCompilerRecognizerFlow$^{\uparrow}$ *acceptance row of chapter 1048, the generated-recognizer row, the certificate-gate row, the finite tuple-image row, the readback row, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. This is not a* StandardBridge *construction and not a recognizer-to-carrier construction; it only sharpens the remaining boundary.*

Proof. Start with the fixed finite tuple route. By Definition 9464.24 and Definition 9464.25, the bridge-facing packet must already display the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row before the fixed route is tested.

Now apply the recognizer boundary. Theorems 9464.26 and 9464.27 add the recognizer acceptance row and finite tuple extraction row ahead of the fixed tuple surface, while keeping the recognizer-to-carrier map outside the available route.

The concrete landing surface is supplied by the two packet chapters. The FiniteReflectionTuple field-faithful prediction, Definition 993.3, forces every bridge-facing read through the displayed tuple image I and readback R, with componentwise transport as part of the same packet. The GroundCompiler recognizer-flow certificate-gate exactness theorem, Definition 1048.3, forces any accepted compiler export through the event-flow row, source-channel row, generated-recognizer row, and certificate gate before it can be read as accepted evidence.

Composing these rows gives the announced landing surface: recognizer acceptance and generated-recognizer mediation must pass the certificate gate before the finite tuple-image and readback rows can be consumed by FiniteReflectionTuple$^{\uparrow}$. The result therefore narrows the remaining recognizer-to-carrier boundary to these concrete packet surfaces, without producing a carrier map and without producing a StandardBridge theorem. □

Theorem 9464.35 (Finite BHist injection recognizer landing counter-witness)**.** *Any counter-witness to the recognizer-side strengthening of Definition 9464.2 must first display the same finite landing surface as a positive attempt: the* GroundCompilerRecognizerFlow$^{\uparrow}$ *acceptance row, generated-recognizer row, certificate-gate row,* FiniteReflectionTuple$^{\uparrow}$ *extraction row,* BHist *readback row, componentwise* hsame *and* NameCert *transport row, and* ChapterTasteGate *row. Absence of one of these rows is only failure to land on the test surface; it is not yet a counterexample to the recognizer-to-carrier strengthening. This theorem asserts no* StandardBridge *construction.*

Proof. Start with the fixed tuple route and its obligation surface, Definition 9464.24 and Definition 9464.25. They make tuple image, readback, componentwise naming transport, and the ChapterTasteGate row the carrier-facing surface on which the weakened bridge question can be tested.

Next use the recognizer boundary and regular recognizer weakening, Definition 9464.26 and Definition 9464.27. These results keep recognizer acceptance and generated-recognizer mediation on the recognizer side until the certificate gate and tuple extraction rows are displayed.

Finally apply the landing-surface theorem, Definition 9464.34. It places the recognizer route on the finite tuple surface only after the seven named rows are visible. Therefore a purported negative example missing one row has not reached the counter-witness boundary, while a genuine counter-witness must show all seven rows and then locate the obstruction without claiming a StandardBridge construction. □

Theorem 9464.36 (Finite BHist injection recognizer landing refinement)**.** *The recognizer route for Definition 9464.1 may be tested only after the* GroundCompilerRecognizerFlow$^{\uparrow}$ *recognizer-flow*

acceptance row lands in the FiniteReflectionTuple$^{\uparrow}$ *image and readback rows, componentwise* hsame *and* NameCert *transport, the* ChapterTasteGate *row, and the non-escape boundary. Absence of any one of these rows is outside the weakened route; once all are present, any remaining failure must be a recognizer-to-carrier obstruction rather than a missing landing-surface row.*

Proof. Start with the recognizer-to-tuple landing surface. By Definition 9464.34, accepted recognizer-flow evidence must pass through the generated-recognizer and certificate-gate rows before it can be read as finite tuple image and readback data.

Next use the fixed tuple route and its obligation surface. Definition 9464.24 and Definition 9464.25 require the tuple image, readback, componentwise NameCert transport, and ChapterTasteGate rows before the fixed finite route is compared.

Finally apply the recognizer counter-boundary. Definition 9464.32 separates missing recognizer, extraction, readback, transport, or taste-gate rows from a genuine recognizer-to-carrier obstruction. Therefore the strengthened test reaches the weakened route only after the full landing surface is displayed, and any later failure is located at the recognizer-to-carrier boundary. □

Theorem 9464.37 (Finite BHist injection regular recognizer row separation)**.** *A regular recognizer attempt for Definition 9464.2 is not on the weakened fixed-tuple surface until all of the following rows are displayed: recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*, generated-recognizer mediation, certificate-gate passage, finite tuple image and readback through* FiniteReflectionTuple$^{\uparrow}$*, componentwise* hsame *and* NameCert *transport, the* ChapterTasteGate *row, and the non-escape boundary. Missing one of these rows is a missing landing row rather than a recognizer-to-carrier obstruction; only after all are present can a remaining failure be assigned to the recognizer-to-carrier boundary.*

Proof. Start with the landing surface. By Definition 9464.34, recognizer-flow evidence must pass through generated-recognizer mediation and the certificate gate before it can be read as finite tuple image and readback data.

Next apply the landing refinement. Definition 9464.36 adds componentwise hsame and NameCert transport, the ChapterTasteGate row, and the non-escape boundary before the weakened route is tested.

Now use Definition 9464.32. It separates absence of recognizer, extraction, readback, transport, or taste-gate rows from an obstruction that remains after the recognizer has landed on the carrier-facing surface.

Finally apply Definition 9464.41. The fixed finite tuple route is the closed weakened surface, so the theorem records row separation only; it does not construct a StandardBridge theorem and does not construct the missing recognizer-to-carrier map. □

Theorem 9464.38 (Finite BHist injection recognizer-to-carrier status tightening)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate passage, finite tuple image and readback, componentwise* hsame *and* NameCert *transport, the* ChapterTasteGate *row, and the non-escape boundary are displayed, any remaining failure is located only at the recognizer-to-carrier construction boundary. It is not a failure of the fixed finite tuple* StandardBridge *route, and it is not caused by missing landing-surface data.*

Proof. Begin with the landing surface, Definition 9464.34. It places accepted recognizer-flow evidence behind generated-recognizer mediation and certificate-gate passage before finite tuple image and readback rows are consumed.

Apply landing refinement, Definition 9464.36. Once componentwise hsame and NameCert transport, the ChapterTasteGate row, and the non-escape boundary are present, the landing surface is no longer missing data.

Use regular recognizer row separation, Definition 9464.37. It identifies absent displayed rows as missing landing data and assigns only post-landing failure to the recognizer-to-carrier boundary.

Finally apply weakened status closure, Definition 9464.41. The fixed finite tuple route remains the closed weakened surface, so the theorem tightens location of failure without changing the conjecture status or claiming a StandardBridge construction. □

Theorem 9464.39 (Finite BHist injection recognizer landing close)**.** *Once recognizer acceptance, generated-recognizer mediation, certificate-gate passage, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, the* ChapterTasteGate *row, and non-escape rows are all displayed, the recognizer input has landed on the fixed finite tuple test surface. Any remaining obstruction in Definition 9464.2 is solely recognizer-to-carrier construction; it is not a missing landing row, not a fixed-tuple premise failure, and not a* StandardBridge *theorem claim.*

Proof. Start with the recognizer-to-tuple landing surface, Definition 9464.34. It places recognizer acceptance, generated-recognizer mediation, certificate-gate passage, finite tuple image, readback, componentwise transport, and the ChapterTasteGate row on the concrete GroundCompilerRecognizerFlow$^{\uparrow}$ and FiniteReflectionTuple$^{\uparrow}$ surfaces.

Apply the landing refinement, Definition 9464.36. Once componentwise hsame and NameCert transport, the ChapterTasteGate row, and the non-escape boundary are present, the route is no longer missing landing-surface data.

Use regular recognizer row separation, Definition 9464.37. It identifies absence of any displayed recognizer, tuple, readback, transport, taste-gate, or non-escape row as failure to land rather than a post-landing obstruction.

Finally apply status tightening, Definition 9464.38. After the landing rows are visible, every remaining obstruction is assigned to recognizer-to-carrier construction, while the conjecture remains in weakened status and no full StandardBridge theorem is asserted. □

Theorem 9464.40 (Finite BHist injection recognizer closed boundary)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate passage, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, the* ChapterTasteGate *row, and the non-escape rows are displayed, the only unresolved content in Definition 9464.2 is the recognizer-to-carrier construction itself. No missing landing row, fixed-tuple premise, report boundary, or status-boundary row remains available as the obstruction.*

Proof. Begin with the landing surface, Definition 9464.34. It places recognizer acceptance, generated-recognizer mediation, and certificate-gate passage before finite tuple image and readback rows are consumed.

Apply the landing counter-witness and landing refinement, Definition 9464.35 and Definition 9464.36. Together they require the same positive landing rows for a negative attempt and add componentwise hsame and NameCert transport, the ChapterTasteGate row, and the non-escape boundary.

Use regular recognizer row separation, Definition 9464.37. It classifies absence of any displayed row as failure to land on the weakened surface rather than an obstruction after landing.

Finally apply status tightening, Definition 9464.38. Once the landing rows are present, every remaining failure is assigned to the recognizer-to-carrier construction boundary itself. Thus the closed landing boundary leaves no other unresolved content. □

Theorem 9464.41 (Finite BHist injection weakened status closure)**.** *The currently weakened finite* BHist *injection conjecture is closed exactly for the fixed finite tuple route. Once the tuple carrier, image row, readback row, componentwise* NameCert *transports, and* ChapterTasteGate *row are displayed, the fixed-route* StandardBridge *obligation surface is complete. The recognizer-to-carrier strengthening remains outside this closure.*

Proof. Apply Definition 9464.24. It identifies the displayed tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the route under the fixed finite tuple test.

Use Definition 9464.25 and Definition 9464.28. Together they state that those rows are the whole public obligation surface for the fixed route once all are present.

Finally apply Definition 9464.30. It keeps recognizer acceptance, finite tuple extraction, and the recognizer-to-carrier map outside the closed fixed-route surface, so the status closure is exact for the weakened finite test and no stronger recognizer claim is imported. □

Theorem 9464.42 (Finite BHist injection StdBridge counter surface)**.** *After the tuple carrier, image row, readback row, componentwise* NameCert *transport, and* ChapterTasteGate *row are all displayed, any remaining refutation of the fixed finite tuple* StandardBridge *route must exhibit a reflection-axis incompatibility with the standard reflection schema. A missing tuple carrier, missing image row, missing readback row, missing componentwise transport, or missing* ChapterTasteGate *row is a failed premise, not a counter-instance.*

Proof. Apply Definition 9464.24. It names the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the fixed route that must be present before the bridge is tested.

Use Definition 9464.25. The proposition identifies the same five rows as the public obligation surface, so absence of one of them means the test has not reached the fixed route.

Now apply Definition 9464.29. Once the five rows are present, a valid counter-instance must locate the obstruction in a reflection-axis requirement incompatible with the standard reflection schema.

Finally apply Definition 9464.41. It keeps the closure exactly at the fixed finite tuple surface, so the post-premise counter surface is only the reflection-axis incompatibility and not a missing-row failure. □

Theorem 9464.43 (Finite BHist injection recognizer boundary sharpens)**.** *Any recognizer-side strengthening of Definition 9464.1 must expose the fixed tuple image row, the fixed tuple readback row, componentwise* NameCert *transport, the* ChapterTasteGate *row, and a non-escape row before any* StandardBridge *request is made. A recognizer acceptance claim that does not display those rows is outside the weakened conjecture's fixed-tuple surface rather than a failed bridge.*

Proof. Start with the fixed tuple route, Definition 9464.24. It supplies the tuple image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the available route before bridge comparison.

Use the fixed tuple obligation surface, Definition 9464.25. That surface makes the tuple rows the public carrier boundary rather than optional recognizer annotations.

Now apply the recognizer boundary, Definition 9464.26, together with the StdBridge counter surface, Definition 9464.42. The recognizer may add acceptance and extraction rows, but it still must land on the fixed tuple surface and display non-escape before a bridge request can be tested.

Therefore recognizer acceptance without tuple readback does not refute the weakened conjecture. It simply has not reached the fixed tuple carrier boundary on which the bridge question is posed. □

Theorem 9464.44 (Finite BHist injection weakened boundary exactness)**.** *The weakened form of Definition 9464.1 is exact for the displayed fixed finite tuple route. Once the tuple carrier, tuple image row, tuple readback row, componentwise* NameCert *transports, and* ChapterTasteGate *row are displayed, every missing-row failure lies outside that route. A refutation of the weakened route can therefore only be a reflection-axis incompatibility with the requested* StandardBridge *surface.*

Proof. Start with Definition 9464.24. It identifies the fixed finite tuple carrier together with the tuple image, tuple readback, componentwise NameCert transport, and ChapterTasteGate rows as the route on which the weakened bridge question is posed.

Use Definition 9464.25. The proposition makes those rows the whole public obligation surface for the fixed route, so a missing tuple row, missing readback row, missing component transport, or missing gate row means the route has not yet been displayed.

Now apply Definition 9464.28. Once the listed rows are present, the fixed finite tuple packet has no further carrier-side obligation before the bridge comparison.

Finally use Definition 9464.42. Any remaining counter-surface must be located on the reflection axis required by the StandardBridge request. Hence the weakened fixed-route portion is boundary-exact, while recognizer-to-carrier strengthening remains outside this theorem. □

Theorem 9464.45 (Finite BHist injection recognizer carrier-map decomposition)**.** *Every recognizer-to-carrier route for Definition 9464.2 must decompose into exactly the following displayed BEDC rows before it can land on the fixed tuple* StandardBridge *surface:*

1. *recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*;*
2. *generated-recognizer mediation;*
3. *certificate-gate admission;*
4. *finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$*;*
5. *component readback into* BHist*;*
6. *componentwise* hsame *and* NameCert *transport;*
7. *the* ChapterTasteGate *row.*

This decomposition is not a StandardBridge *construction and not a full recognizer-to-carrier map. It only names the rows that any such route must expose before the fixed tuple surface can be tested.*

Proof. Start with the fixed tuple route and its obligation surface, Definition 9464.24 and Definition 9464.25. They require the tuple image, readback, componentwise naming transport, and ChapterTasteGate rows before a fixed tuple bridge request is meaningful.

Next apply the recognizer boundary and landing surface, Definition 9464.43 and Definition 9464.34. These results place recognizer acceptance, generated-recognizer mediation, and certificate-gate admission in front of the fixed tuple packet, while keeping the bridge construction and the full carrier map outside the available conclusion.

The two sibling packet surfaces then isolate the named rows. The GroundCompiler recognizer-flow packet supplies acceptance, generated-recognizer mediation, and certificate-gate admission; the FiniteReflectionTuple packet supplies finite tuple extraction, readback into BHist, componentwise hsame and NameCert transport, and the ChapterTasteGate row.

Composing only these displayed surfaces gives the stated decomposition. Since the proof merely identifies the rows needed before the fixed tuple route can be tested, it does not construct a StandardBridge theorem and does not construct the missing recognizer-to-carrier map. □

Theorem 9464.46 (Finite BHist injection recognizer row exhaustion)**.** *Every admissible recognizer-to-carrier test for Definition 9464.2 is exhausted by the seven displayed rows of Definition 9464.45: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. Once those rows are displayed, the recognizer route has reached the finite BEDC-kernel test surface; no additional row is claimed to construct the full carrier map.*

Proof. Apply Definition 9464.45. It lists the only rows used to move a recognizer-side input toward the fixed tuple surface: the three recognizer-flow rows, the tuple extraction row, the readback row, the componentwise transport row, and the taste-gate row.

The recognizer-flow rows are exactly recognizer acceptance, generated-recognizer mediation, and certificate-gate admission. The finite tuple rows are exactly finite tuple extraction, component readback into BHist, componentwise hsame and NameCert transport, and ChapterTasteGate.

Therefore the admissible test surface has no further row beyond those seven displayed rows. This exhausts the finite BEDC-kernel test while leaving the construction of a full recognizer-to-carrier map outside the conclusion. □

Theorem 9464.47 (Finite BHist injection seven-row landing exhaustion)**.** *Suppose a recognizer-side packet displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. Then it has exhausted the weakened finite landing surface for Definition 9464.2. Any further* StandardBridge *theorem, recognizer-to-carrier construction, quotient carrier replacement, host-equality comparison, or arbitrary regular-language carrier claim lies outside this theorem.*

Proof. Apply recognizer row exhaustion, Definition 9464.46. It identifies exactly the seven rows named in the statement as the finite BEDC-kernel test surface for the recognizer-side route.

Next read the closed landing test, Definition 9464.48. It says that passing this surface asserts no StandardBridge theorem, no recognizer-to-carrier map, and no arbitrary regular-language carrier construction.

Finally use the nonclaim boundary of Definition 9464.12. That boundary keeps bridge construction and recognizer-to-carrier construction outside the weakened landing test, so the displayed seven rows exhaust only the finite landing surface. □

Proposition 9464.48 (Finite BHist recognizer closed landing test)**.** *The weakened recognizer-side test for Definition 9464.2 is exactly the seven rows of Definition 9464.46: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. Passing this landing test asserts no* StandardBridge *theorem, no recognizer-to-carrier map, and no arbitrary regular-language carrier construction.*

Proof. Apply Definition 9464.46. It states that every admissible recognizer-to-carrier test is exhausted by the seven displayed rows.

The first three rows place the recognizer input on the existing $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ surface. The last four rows place the resulting tuple on the existing $\mathsf{FiniteReflectionTuple}^{\uparrow}$ and ChapterTasteGate surface.

Thus the test is closed as a finite landing surface only. The theorem explicitly leaves the full carrier map outside the conclusion, and the fixed tuple bridge results do not add a StandardBridge theorem or a regular-language carrier construction here. □

Theorem 9464.49 (Finite BHist injection recognizer refutation test)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row are all displayed as in Definition 9464.48, any valid negative result for Definition 9464.2 must exhibit a concrete recognizer-to-carrier obstruction. Such a negative result cannot rest on a missing landing row, a missing fixed-tuple premise for* StandardBridge*, or a failure to expose the seven-row finite test surface.*

Proof. Start with the closed landing test, Definition 9464.48. It states that the weakened recognizer-side test is exactly the seven displayed rows and that passing the test asserts no full StandardBridge theorem.

Next use recognizer row exhaustion, Definition 9464.46. Once those rows are present, the finite BEDC-kernel landing surface has no further row whose absence can be cited as the reason the recognizer-to-carrier route fails.

Finally compare with the fixed tuple route and its obligation surface, Definition 9464.24 and Definition 9464.25. Since the fixed-tuple premises are already part of the displayed landing surface, a negative result must identify an obstruction in the recognizer-to-carrier step itself. □

Theorem 9464.50 (Finite BHist injection recognizer carrier obstruction locality)**.** *Once the seven landing rows of Definition 9464.48 are displayed, every valid obstruction to strengthening Definition 9464.2 is localized to the recognizer-to-carrier construction row. It cannot be assigned to missing tuple extraction, component readback, componentwise* hsame *and* NameCert *transport,* ChapterTasteGate*, or a missing fixed-tuple bridge premise.*

Proof. Start with recognizer carrier-map decomposition. It displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, and the ChapterTasteGate row as the rows required before the fixed tuple surface is tested.

Apply recognizer row exhaustion and the closed landing test. Together they say that the seven-row surface is complete as a finite BEDC-kernel landing test and that no full StandardBridge theorem is asserted by passing it.

Finally use the recognizer refutation test. Since a negative result cannot cite a missing landing row or missing fixed-tuple premise after the seven rows are present, the only remaining obstruction surface is the recognizer-to-carrier construction row itself. □

Theorem 9464.51 (Finite BHist injection recognizer carrier obstruction tightening)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row are all displayed, any remaining failure of Definition 9464.2 is localized to the recognizer-to-carrier construction boundary. It is not a failure of the fixed-tuple* StandardBridge *route, the tuple image row, the tuple readback row, componentwise transport, or the* ChapterTasteGate *data.*

Proof. Apply recognizer carrier-map decomposition, Definition 9464.45. It names the seven displayed landing rows required before the fixed tuple surface can be tested.

Use the closed landing test and recognizer row exhaustion, Definition 9464.48 and Definition 9464.46. Together they make the displayed landing rows complete as a finite BEDC-kernel test surface without adding a full carrier map.

Now compare with the fixed tuple route and its public obligation surface, Definition 9464.24 and Definition 9464.25. The tuple image, tuple readback, componentwise transport, and ChapterTasteGate rows are already present, so they cannot be the source of a post-landing failure.

Finally apply recognizer obstruction locality, Definition 9464.50. The only obstruction surface left after the displayed rows are present is the recognizer-to-carrier construction boundary itself. □

Theorem 9464.52 (Finite BHist injection carrier-map constructive obstruction)**.** *Suppose a negative settlement of Definition 9464.2 is claimed after the recognizer acceptance row through* GroundCompilerRecognizer *generated-recognizer mediation, certificate-gate admission, finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$, BHist *component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row have all been displayed. Then the negative settlement must exhibit a finite obstruction to constructing a carrier row preserving those same seven landing rows. It cannot be a missing landing row, a missing fixed-tuple bridge premise, a quotient carrier replacement, a host-equality comparison, or a request for an additional* StandardBridge *theorem.*

Proof. Begin with the closed landing test, Definition 9464.48. It identifies exactly the seven rows that put a recognizer-side input on the weakened fixed-tuple surface. Once those rows are displayed, absence of recognizer acceptance, generated-recognizer mediation, certificate-gate admission, tuple extraction, component readback, componentwise transport, or the taste gate is no longer available as the obstruction.

Next apply the carrier-row dichotomy of Definition 9464.50. After landing, any failure is assigned to the recognizer-to-carrier construction row itself. The fixed-tuple surface and its componentwise NameCert transport rows have already been supplied.

Finally use the no-upgrade boundary of Definition 9464.51. A negative settlement therefore has to keep the same finite landing data and show that no carrier row preserves it. Replacing the carrier by a quotient object, comparing host equality, dropping a landing row, or asking for a separate StandardBridge theorem changes the target rather than settling the recognizer-to-carrier boundary. □

Theorem 9464.53 (Finite BHist injection carrier-map obstruction stability)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, every remaining negative case for Definition 9464.2 must be a carrier-map preservation obstruction. It is not a missing-row failure, not a failure of the seven-row landing test, and not a* StandardBridge *claim.*

Proof. Begin with the closed landing test, Definition 9464.48. It makes the seven displayed rows the whole finite landing surface for the weakened recognizer-side test.

Next use recognizer carrier obstruction locality, Definition 9464.50. Once the landing rows are present, any remaining obstruction is localized to the recognizer-to-carrier construction row rather than to tuple extraction, readback, componentwise transport, or the taste gate.

Finally apply constructive obstruction, Definition 9464.52. A valid negative case must preserve the same displayed landing data while showing that the carrier row cannot be constructed. Thus the remaining object is a carrier-map preservation obstruction only, and no StandardBridge theorem is asserted. □

Theorem 9464.54 (Finite BHist injection recognizer carrier obstruction completeness)**.** *After the seven landing rows of Definition 9464.48 are displayed, the only remaining open strengthening of Definition 9464.2 is a concrete recognizer-to-carrier construction row. The conjecture remains in weakened status: the landing surface is closed, the refutation test is localized, and no full carrier construction or* StandardBridge *theorem is asserted.*

Proof. Apply recognizer row exhaustion and the closed landing test. Together they state that recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, and the ChapterTasteGate row exhaust the displayed landing surface.

Next use the recognizer refutation test. Once the landing rows are present, a negative result must exhibit a concrete recognizer-to-carrier obstruction rather than a missing landing row, missing fixed-tuple premise, or missing finite test surface.

Finally apply obstruction locality. It assigns every remaining obstruction to the recognizer-to-carrier construction row itself. Thus the post-landing boundary is complete as an obstruction statement while leaving the conjecture weakened rather than closed. □

The weakened status of Definition 9464.2 is therefore exact. The remaining object is one carrier-map compatibility row preserving the seven displayed rows: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist component readback, componentwise hsame and NameCert transport, and the ChapterTasteGate row. The conjecture refuses construction of a StandardBridge, arbitrary regular-language carriers, quotient equality of carriers, and host recognizer equality; those are outside the finite BEDC kernel row being isolated.

Theorem 9464.55 (Finite BHist injection carrier-map boundary)**.** *The seven-row landing test of Definition 9464.48 remains strictly below recognizer-to-carrier construction and strictly below* StandardBridge *construction. Adding replay and provenance rows to the landed recognizer surface cannot promote the weakened conjecture to a carrier-map theorem, cannot construct a carrier row, and cannot assert a fixed-tuple* StandardBridge*.*

Proof. Apply recognizer carrier obstruction completeness, Definition 9464.54. It places the remaining issue exactly at recognizer-to-carrier construction after the seven landing rows have been displayed.

Use the recognizer landing sharpener, Definition 9464.20, and readback closed landing, Definition 9464.18. These results identify the landed rows as recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and the taste-gate row.

Finally apply readback transport closure, Definition 9464.19. Replay and provenance preserve the displayed readback and transport surface, but they do not add a carrier construction row and do not supply the fixed-tuple bridge object. Therefore the boundary stays below both carrier-map construction and StandardBridge construction. □

Theorem 9464.56 (Finite BHist injection carrier-row preservation criterion)**.** *After the carrier-row refutation test, a positive chase of Definition 9464.2 must exhibit one carrier row K on the same finite* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *surface. The row K must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. It must not assert* StandardBridge*, arbitrary recognizer-to-carrier construction, quotient carrier equality, or host recognizer equality.*

Proof. Apply the carrier-row refutation test. It rules out negative evidence that merely repeats a missing landing row, a fixed-tuple bridge premise, quotient data, host equality, or detached bridge assertion.

Use the carrier-map progress row. It identifies progress on the weakened conjecture with one carrier row on the same finite kernel surface, preserving the recognizer, tuple, readback, transport, local naming, and taste-gate data already displayed.

The preservation criterion is therefore positive: it asks for the carrier row K and for preservation of the listed rows on the existing finite surface. Since no bridge theorem, quotient equality, host recognizer equality, or arbitrary recognizer-to-carrier construction is supplied, the conjecture status remains unchanged. □

Theorem 9464.57 (Finite BHist injection recognizer construction obstruction localization)**.** *After the seven landing rows of Definition 9464.48 are displayed, a valid obstruction to strengthening Definition 9464.2 must be a concrete failure to construct a carrier row preserving recognizer acceptance, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. It cannot be a missing landing row, a missing fixed-tuple premise, a missing* StandardBridge *theorem, a missing host-equality carrier, a quotient-carrier substitution, or an aggregate status token.*

Proof. Apply recognizer row exhaustion, Definition 9464.46, and the closed landing test, Definition 9464.48. Together they place the recognizer-flow rows and the finite tuple rows on the displayed BEDC-kernel landing surface.

Next apply the recognizer refutation test, Definition 9464.49. Once that landing surface is present, a negative result cannot cite the absence of a landing row, a fixed-tuple premise, or the seven-row finite test surface.

Finally use obstruction completeness, Definition 9464.54. The only remaining open strengthening is the carrier construction row itself, so the post-landing obstruction must identify a failure of that row to preserve the recognizer acceptance, tuple extraction, readback, componentwise transport, and taste-gate data already displayed. □

Proposition 9464.58 (Finite BHist injection carrier-map compatibility row)**.** *After the seven landing rows of Definition 9464.48 are displayed, the weakened recognizer-to-carrier claim in Definition 9464.2 is exactly a carrier-map compatibility row. Such a row must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise transport, and the* ChapterTasteGate *row on the same displayed finite surface.*

Proof. The closed landing test displays the seven rows named in the statement. Therefore a later carrier read is compatible only if it preserves those rows rather than replacing them by a second recognizer, a different tuple image, or an external bridge target.

The status-sharpness results below keep the conjecture on the weakened boundary. Thus the remaining object is the single carrier-map compatibility row preserving the displayed landing surface, while the chapter still excludes a full fixed-tuple StandardBridge construction. □

Theorem 9464.59 (Finite BHist injection carrier-map row exactness)**.** *Mode: restate. The weakened conjecture of Definition 9464.2 is exactly the seven-row landing surface plus one carrier-map compatibility row. The seven rows are recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*; the additional carrier-map row must preserve those*

seven displayed rows on the same BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, *and* FiniteReflectionTuple$^{\uparrow}$ *surface. No* StandardBridge *theorem, arbitrary regular-language carrier, quotient carrier equality, or host-equality comparison is part of the weakened claim.*

Proof. Read the recognizer-to-carrier status paragraph in the parent section. It names the seven landing rows and separates them from the remaining carrier-map compatibility row.

Apply carrier-map compatibility row, Definition 9464.58. It identifies the post-landing row as preservation of the same recognizer, tuple, readback, transport, and taste-gate data.

Finally apply the carrier-map boundary, Definition 9464.55. Since replay and provenance do not add a carrier construction theorem or a fixed-tuple bridge object, the weakened conjecture has exactly the seven-row landing surface plus the compatibility row stated here. □

Theorem 9464.60 (Finite BHist injection carrier-map witness test)**.** *Any positive chase of the weakened conjecture in Definition 9464.2 must exhibit one carrier row K preserving exactly the following displayed rows on the same finite surface:*

1. *recognizer acceptance;*
2. *generated-recognizer mediation;*
3. *certificate-gate admission;*
4. FiniteReflectionTuple$^{\uparrow}$ *extraction;*
5. BHist *component readback;*
6. *componentwise* hsame *and* NameCert *transport;*
7. *the* ChapterTasteGate *row.*

Thus the chase is a carrier-map witness test rather than a status-only restatement, and the conjecture remains on the weakened boundary.

Proof. Apply carrier-map row exactness, Definition 9464.59. It separates the seven-row landing surface from the single additional carrier-map compatibility row.

Now apply the finite-row obstruction test, Definition 9464.63. It identifies the remaining open object as a carrier row K preserving the recognizer, tuple, readback, transport, and taste-gate data already displayed.

Combining the two results gives the witness test stated above. A positive chase must supply the row K and show preservation of exactly those seven rows; otherwise it has not left the weakened recognizer-to-carrier boundary and has not produced a fixed-tuple StandardBridge theorem. □

Theorem 9464.61 (Finite BHist injection carrier-map witness row)**.** *Mode: restate. A successful chase of the weakened recognizer-to-carrier conjecture in Definition 9464.2 must display a single carrier row K on the same finite* BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, *and* FiniteReflectionTuple$^{\uparrow}$ *surface. The row K must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. Until such a row is displayed, the conjecture stays weakened and no fixed-tuple* StandardBridge *theorem is asserted.*

Proof. Apply the carrier-map witness test, Definition 9464.60. It isolates the positive chase at one carrier row K preserving the seven displayed rows.

Use carrier-map row exactness, Definition 9464.59. The exactness statement keeps the recognizer, generated recognizer, certificate gate, finite tuple, readback, transport, and taste-gate data on the same finite surface rather than replacing them by a second bridge object.

Finally apply the carrier-map boundary, Definition 9464.55. Because replay and provenance do not construct the carrier row and do not supply a fixed-tuple StandardBridge, the conjecture remains weakened until the row K itself is displayed. □

Theorem 9464.62 (Finite BHist injection carrier-row positive boundary)**.** *Mode: restate. A positive chase of the weakened recognizer-to-carrier conjecture must display one carrier row K on the same* GroundCompilerRecognizerFlow$^{\uparrow}$ *and* FiniteReflectionTuple$^{\uparrow}$ *surface. The row K must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *without replacing the displayed surface by a fixed-tuple* StandardBridge *theorem or an external carrier selector.*

Proof. Apply the carrier-map witness row, Definition 9464.61. It identifies the positive object as one carrier row K on the same finite BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, and FiniteReflectionTuple$^{\uparrow}$ surface.

Use the witness test, Definition 9464.60. The test lists the seven rows that K must preserve: recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist readback, componentwise transport, and the taste-gate row.

Finally apply row exactness, Definition 9464.59. It keeps the positive boundary on the weakened carrier-map row rather than turning it into a fixed-tuple StandardBridge theorem or a host-side selector. □

Theorem 9464.63 (Finite BHist injection recognizer finite-row obstruction test)**.** *After the landing rows of Definition 9464.48 are displayed, the remaining open recognizer-to-carrier claim is exactly the existence of a carrier row K such that K preserves recognizer acceptance through* GroundCompilerRecognizerFlow$^{\uparrow}$*, finite tuple extraction through* FiniteReflectionTuple$^{\uparrow}$*, component readback into* BHist*, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row. A valid obstruction is therefore a finite-row obstruction to constructing such a K; it is not a missing landing row, not a fixed-tuple* StandardBridge *obstruction, and not an external bridge claim.*

Proof. Start with the closed landing test. It identifies the recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate rows as the whole finite landing surface.

Apply recognizer construction obstruction localization. Once those rows are present, any remaining negative result must identify failure of the carrier construction row to preserve the displayed recognizer, tuple, readback, transport, and taste-gate data.

Now use obstruction completeness. It says that the only remaining open strengthening is the carrier construction row itself, while the conjecture stays on the weakened surface and does not assert a StandardBridge theorem.

Thus the test is exact: either such a carrier row K exists and preserves the listed rows, or the obstruction is a finite-row failure to construct K after landing. No broader bridge claim is introduced. □

Theorem 9464.64 (Finite BHist injection recognizer status sharpness)**.** *After the seven landing rows of Definition 9464.48 are displayed, the remaining open issue in Definition 9464.2 is exactly recognizer-to-carrier construction. It is not a fixed-tuple* StandardBridge *issue, not missing landing data, not a missing finite tuple image or readback row, and not a missing bridge theorem.*

Proof. Start with the recognizer-to-tuple landing surface, Definition 9464.34. It places recognizer acceptance, generated-recognizer mediation, certificate-gate passage, finite tuple image, readback,

componentwise transport, and ChapterTasteGate on the concrete GroundCompilerRecognizerFlow$^{\uparrow}$ and FiniteReflectionTuple$^{\uparrow}$ surfaces.

Apply landing refinement, Definition 9464.36. It says that once the full landing surface is displayed, absence of one of those rows is outside the weakened route rather than a post-landing obstruction.

Use regular recognizer row separation, Definition 9464.37. Missing recognizer, generated-recognizer, certificate-gate, tuple image, readback, componentwise transport, taste-gate, or non-escape data is classified as missing landing data, not as the remaining issue after landing.

Finally apply status tightening, Definition 9464.38. It assigns every remaining post-landing failure to the recognizer-to-carrier construction boundary itself and separates that boundary from the fixed finite tuple StandardBridge route. Therefore the weakened status is sharp exactly at recognizer-to-carrier construction. □

Theorem 9464.65 (Finite BHist injection regular recognizer status sharpens)**.** *Once recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise transport, and the* ChapterTasteGate *row have all been displayed, the only remaining conjectural object in Definition 9464.2 is a carrier row preserving those seven rows. The open boundary is therefore neither a missing landing row, nor a fixed-tuple* StandardBridge *claim, nor a demand for an additional bridge theorem.*

Proof. Start with the recognizer-to-tuple landing surface, Definition 9464.34. It places the recognizer acceptance route behind generated-recognizer mediation and certificate-gate admission before tuple extraction and readback are consumed.

Apply landing refinement, Definition 9464.36. After componentwise transport and the ChapterTasteGate row are present, the displayed landing surface already contains the seven rows named in the statement.

Use regular recognizer row separation, Definition 9464.37. It classifies absence of any one of those rows as failure to land, not as the remaining conjectural issue after landing.

Thus the post-landing open object is exactly a carrier row preserving the displayed row family. Since the fixed-tuple bridge route is separate from this carrier construction boundary, no StandardBridge theorem is asserted here. □

Theorem 9464.66 (Finite BHist injection open kernel restatement)**.** *The remaining* StandardBridge-*facing claim for finite* BHist *injection is exactly the finite kernel-row test: a candidate must supply a finite tuple row, a recognizer landing row, a carrier-map row, a readback row, a transport row, a replay row, and a package/name row on the same* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *surface. Omitting any one of these rows leaves the bridge unclosed, and the status remains the weakened recognizer-to-carrier boundary rather than a full* StandardBridge *theorem.*

Proof. Apply the recognizer-carrier obstruction locality theorem, Definition 9464.50. It isolates the post-landing issue at the carrier construction row after the recognizer acceptance, tuple extraction, readback, componentwise transport, and taste-gate data have landed.

Use regular recognizer status sharpness, Definition 9464.65. That theorem says the seven landing rows must already be present before the remaining carrier row can even be tested.

Together these results restate the open object as a finite kernel-row checklist. A tuple row without recognizer landing, a landing row without carrier preservation, a carrier row without readback and transport, or replay and package/name data without the preceding rows is still outside the closed bridge surface. □

Theorem 9464.67 (Finite BHist injection carrier-map positive test)**.** *Mode: restate. A positive chase of the weakened recognizer-to-carrier claim in Definition 9464.2 must exhibit one carrier row on the same finite* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *surface. That row must preserve recognizer acceptance, finite tuple extraction,* BHist *readback, componentwise* hsame *transport,* NameCert *transport, and* ChapterTasteGate *without replacing the surface by a fixed-tuple* StandardBridge *theorem. The conjecture remains open until such a carrier row is displayed.*

Proof. Start from the open kernel restatement, Definition 9464.66. It names the finite kernel-row surface on which the remaining bridge-facing claim must live.

Apply carrier-map row exactness, Definition 9464.59. The exactness result separates the already displayed landing rows from the single additional carrier-map row.

Finally use the recognizer finite-row obstruction test, Definition 9464.63. It identifies the remaining open object as a carrier row preserving recognizer acceptance, tuple extraction, readback, componentwise transport, and the taste-gate data on the same finite surface. Thus a positive chase must display that row, and the conjecture remains open until it is supplied. □

Theorem 9464.68 (Finite BHist injection carrier-map progress row)**.** *Mode: restate. Any progress on Definition 9464.2 must display one carrier row on the same finite* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* GroundCompilerRecognizerFlow$^{\uparrow}$*, and* FiniteReflectionTuple$^{\uparrow}$ *kernel surface. That row must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *transport,* NameCert *transport, and* ChapterTasteGate *data. A status token, a fixed-tuple* StandardBridge *theorem, or a second bridge object is not progress unless it exposes that same carrier row.*

Proof. Start with the carrier-map positive test, Definition 9464.67. It already identifies the remaining positive chase as one carrier row rather than as a global bridge assertion.

Use carrier-map row exactness, Definition 9464.59. Exactness keeps recognizer acceptance, generated recognizer mediation, certificate admission, tuple extraction, readback, transport, and taste-gate data on one finite kernel surface.

Finally apply the recognizer finite-row obstruction test, Definition 9464.63. It rules out a proof that skips any of the displayed rows, so the progress object is exactly the stated carrier-map row and the conjecture remains open until such a row is supplied. □

Theorem 9464.69 (Finite BHist injection carrier-row open test)**.** *Closing Definition 9464.2 requires exactly one carrier row K on the same* GroundCompilerRecognizerFlow$^{\uparrow}$ *and* FiniteReflectionTuple$^{\uparrow}$ *surface. The row K must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *transport,* NameCert *transport, and* ChapterTasteGate *admission. Supplying K is the whole open witness test: no* StandardBridge *theorem, quotient carrier equality, host recognizer equality, detached bridge object, or second carrier surface is part of the test.*

Proof. Apply carrier-map row exactness, Definition 9464.59. It decomposes the weakened conjecture into the seven displayed landing rows plus one carrier-map compatibility row on the same finite BHist, hsame, Cont, Pkg, NameCert, GroundCompilerRecognizerFlow$^{\uparrow}$, and FiniteReflectionTuple$^{\uparrow}$ surface.

Use the witness test, Definition 9464.60, and the progress row, Definition 9464.68. These two statements identify the remaining positive object as a single row K preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, readback, transport, local naming, and taste-gate data.

Finally apply the carrier-map boundary, Definition 9464.55, together with the carrier-row refutation test, Definition 9464.71. They exclude StandardBridge, quotient carrier equality, host recognizer equality, and detached bridge assertions from both positive and negative readings. Therefore the conjecture remains open exactly at the one carrier-row witness test. □

Theorem 9464.70 (Finite BHist injection carrier-row discharge criterion)**.** *Let K be a displayed carrier row on the same* GroundCompilerRecognizerFlow$^{\uparrow}$ *and* FiniteReflectionTuple$^{\uparrow}$ *surface as the landing data of Definition 9464.2. If K preserves recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *transport,* NameCert *transport, and* ChapterTasteGate *admission, then the weakened recognizer-to-carrier conjecture is closed for that packet. The conclusion does not assert* StandardBridge*, quotient carrier equality, host recognizer equality, or a detached bridge object.*

Proof. Apply carrier-map row exactness, Definition 9464.59. It identifies the weakened conjecture with the seven landing rows plus one carrier-map compatibility row on the same finite support surface.

Use the witness test and progress row, Definition 9464.60 and Definition 9464.68. They say that positive progress is precisely a displayed row K preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, tuple extraction, readback, componentwise transport, local naming, and taste-gate data.

Now apply the carrier-row open test, Definition 9464.69. Its open witness is exactly the same row K on the same GroundCompilerRecognizerFlow$^{\uparrow}$ and FiniteReflectionTuple$^{\uparrow}$ surface. The hypotheses therefore discharge the remaining carrier-row test for this packet.

Finally read the carrier-map boundary. It excludes StandardBridge, quotient carrier equality, host recognizer equality, detached bridge objects, and second carrier surfaces from the conclusion, so the closure is only the weakened recognizer-to-carrier packet boundary. □

Theorem 9464.71 (Finite BHist injection carrier-row refutation test)**.** *After the landing rows of Definition 9464.48 are present, any refutation of Definition 9464.2 must display a finite obstruction to preserving the same seven rows named in that conjecture. Missing landing rows, fixed-tuple* StandardBridge *premises, quotient carriers, host equality, and detached bridge assertions do not count as such an obstruction.*

Proof. Apply the recognizer construction obstruction localization theorem, Definition 9464.57. Once the seven landing rows are present, the only remaining negative result must be a concrete failure of the carrier row to preserve the displayed recognizer, tuple, readback, transport, and taste-gate data.

Use the carrier-map boundary, Definition 9464.55. It rules out a refutation that merely repeats replay or provenance, or that substitutes a fixed-tuple bridge object for the carrier row.

Therefore any legitimate refutation must exhibit a finite obstruction to preserving exactly the seven rows already named in the conjecture. Missing landing rows, host equality, quotient carriers, or detached bridge assertions are outside the refutation test. □

Open recognizer-to-carrier boundary. Status: `weakened`

The fixed-tuple StandardBridge route and seven-row recognizer landing test are not the conjectural part of Definition 9464.2. The open claim is the construction of a carrier map from a regular recognizer that supplies the displayed finite tuple extraction, readback, componentwise transport, and ChapterTasteGate rows after the finite landing rows have already been exposed; no StandardBridge theorem is claimed at this boundary.

Theorem 9464.72 (Finite BHist injection carrier-row settlement restatement)**.** *Any close or refutation of the weakened recognizer-to-carrier conjecture in Definition 9464.2 is exhausted by exactly one carrier-row outcome. Either there is a preserving row for the displayed family consisting of recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*, or there is a finite obstruction to preserving that same family. No missing landing row, bridge assertion, quotient carrier, or host equality route is part of this settlement test.*

Proof. Start with the positive settlement test, Definition 9464.73. It identifies the positive exit as one displayed carrier row preserving the recognizer, tuple, readback, transport, NameCert, and taste-gate family.

Use the closed landing test, Definition 9464.48. The listed recognizer and finite tuple rows are already displayed before the carrier row is evaluated, so a missing landing row is not a settlement of the weakened boundary.

Apply the carrier obstruction restatement, Definition 9464.77. It supplies the negative exit as a finite obstruction to preserving precisely the same row family.

Finally use the no-StdBridge-upgrade theorem, Definition 9464.79. This keeps both exits inside the carrier-facing test and prevents either outcome from being read as a fixed-tuple StandardBridge assertion. □

Theorem 9464.73 (Finite BHist injection carrier row positive settlement test)**.** *A positive settlement of the weakened recognizer-to-carrier boundary in Definition 9464.2 is exactly one displayed carrier row on the same support surface that preserves recognizer acceptance, generated-recognizer mediation, certificate-gate admission,* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. This row is still not a fixed-tuple* StandardBridge *assertion.*

Proof. Start with the boundary paragraph above. It leaves the fixed-tuple bridge route outside the conjectural content and places the open issue at the recognizer-to-carrier construction row.

Use the closed landing test, Definition 9464.48. The recognizer, generated-recognizer, certificate-gate, tuple-extraction, component-readback, transport, and taste-gate rows are already displayed before the carrier row is evaluated.

Finally apply the no-StdBridge-upgrade theorem, Definition 9464.79. It keeps the preservation row on the carrier-facing support surface and prevents the test from becoming a fixed-tuple bridge assertion. □

Theorem 9464.74 (Finite BHist injection recognizer map obstruction)**.** *A recognizer-to-carrier strengthening of Definition 9464.2 can proceed only after the following finite obstruction row is eliminated: a carrier-map compatibility row must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *along the same displayed carrier route. Absence of this compatibility row keeps the conjecture weakened rather than open-ended; the missing object is finite, local, and carrier-facing, not a missing landing row and not a* StandardBridge *theorem.*

Proof. Start with the fixed tuple route and obligation surface, Definition 9464.24 and Definition 9464.25. They identify the tuple image, tuple readback, componentwise transport, and ChapterTasteGate rows already required before any bridge comparison is tested.

Use the recognizer boundary and closed landing test, Definition 9464.43 and Definition 9464.48. These results add recognizer acceptance, generated-recognizer mediation, and certificate-gate ad-

mission ahead of the fixed tuple surface while leaving recognizer-to-carrier construction outside the displayed landing rows.

Apply the landing counter-witness theorem, Definition 9464.35. A negative route that omits one of the landing rows has not reached the carrier-facing boundary, so it cannot count as the obstruction named here.

Finally use recognizer carrier-map decomposition and row exhaustion, Definition 9464.45 and Definition 9464.46. Once the listed rows are present, the only remaining finite object is the carrier-map compatibility row preserving that family. Its absence is therefore a finite obstruction boundary and leaves the conjecture weakened without asserting a standard bridge. □

Theorem 9464.75 (Finite BHist injection recognizer closed landing boundary)**.** *Once recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, the remaining open object in Definition 9464.2 is exactly a carrier row preserving those seven rows. Missing landing data is outside the conjecture, and the boundary does not assert a fixed-tuple* StandardBridge *theorem.*

Proof. Apply Definition 9464.48. It displays the seven landing rows as the closed landing surface rather than as later carrier-map data.

Use Definition 9464.64. After that surface is present, missing recognizer, generated-recognizer, certificate-gate, tuple-extraction, readback, transport, or taste-gate data is not the open issue.

Finally apply Definition 9464.77. The remaining object is precisely a carrier row preserving the displayed row family, with a finite obstruction to that preservation as the negative alternative and no StandardBridge assertion. □

Theorem 9464.76 (Finite BHist injection readback converse boundary)**.** *Once the seven landing rows of Definition 9464.48 are present, any remaining negative result for Definition 9464.2 must exhibit failure of the recognizer-to-carrier construction itself. It cannot be a missing* StandardBridge *premise, an omitted readback row, absent componentwise* hsame *or* NameCert *transport, or a missing* ChapterTasteGate *row.*

Proof. Start with the closed landing test. Its seven rows include recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist component readback, componentwise transport, and the ChapterTasteGate row.

Apply recognizer status sharpness. After those rows have landed, missing landing data is no longer the active boundary, and the fixed-tuple StandardBridge route remains separate from the recognizer-to-carrier claim.

Finally use the carrier obstruction restatement. The only remaining negative case is a finite obstruction to constructing a carrier row preserving the displayed recognizer, tuple, readback, transport, and taste-gate data. Therefore the converse boundary is localized at recognizer-to-carrier construction itself. □

Theorem 9464.77 (Finite BHist injection carrier obstruction restatement)**.** *The open recognizer-to-carrier boundary is exact: any settlement of Definition 9464.2 must provide either a carrier row preserving recognizer acceptance, tuple extraction,* BHist *readback, componentwise transport, and* ChapterTasteGate*, or a finite obstruction to preserving precisely that family of rows. The statement does not claim* StandardBridge*.*

Proof. Read the open boundary paragraph together with Definition 9464.64. They place the fixed-tuple bridge and the seven landing rows outside the remaining conjectural content.

Apply the finite-row obstruction test, Definition 9464.63. It identifies the remaining row as a carrier construction preserving the recognizer, tuple, readback, transport, and taste-gate data, and it identifies the negative case as a finite obstruction to that same preservation.

Therefore the boundary has exactly the two stated exits. Neither exit supplies a standard bridge theorem; it only settles the carrier-map obstruction surface. □

Theorem 9464.78 (Finite BHist injection carrier-map obstruction-witness test)**.** *Any close or refutation of the open recognizer-to-carrier boundary in Definition 9464.2 must exhibit exactly one of two finite objects: either a carrier row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and the* ChapterTasteGate *row, or a finite obstruction to preserving that row family. Missing landing rows and fixed-tuple* StandardBridge *premises are outside the open conjecture.*

Proof. Start with Definition 9464.48. It displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate as landing rows rather than as later conjectural data.

Apply Definition 9464.64. The open issue is the recognizer-to-carrier construction after those rows have landed, not the fixed-tuple StandardBridge route and not a missing landing component.

Thus a positive close must give a carrier row preserving the displayed row family, while a negative close must give a finite obstruction to that preservation. Any argument about absent landing rows or fixed-tuple bridge premises addresses a different boundary and does not settle the open conjecture. □

Theorem 9464.79 (Finite BHist injection recognizer no StdBridge upgrade)**.** *Once the seven landing rows of Definition 9464.48 are exposed, a recognizer-to-carrier construction row can feed only the carrier-map boundary for Definition 9464.2. It cannot by itself assert* StandardBridge*. Any bridge claim still requires the fixed tuple route together with the separate standard reflection schema.*

Proof. Start with status sharpness, Definition 9464.64. It identifies the open issue exactly as recognizer-to-carrier construction after the landing rows are present, and separates that issue from the fixed-tuple StandardBridge route.

Apply the carrier refutation boundary, Definition 9464.83. That boundary requires the fixed tuple route, regular recognizer separation, carrier-map decomposition, row exhaustion, and construction-obstruction localization before a route is even counted on the weakened conjecture surface.

Finally use obstruction exhaustion, Definition 9464.84. It leaves only the carrier row preserving recognizer acceptance, tuple extraction, readback, componentwise transport, NameCert data, and ChapterTasteGate. Therefore the recognizer-to-carrier row can supply the carrier-map boundary, while a StandardBridge assertion still needs the fixed tuple route and the standard reflection schema as separate data. □

Theorem 9464.80 (Finite BHist injection recognizer closed status)**.** *The only open content in Definition 9464.2 is a carrier row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*. Any fixed-tuple* StandardBridge *question, missing-landing-row issue, or absent fixed tuple premise is outside the open conjecture.*

Proof. Start with the closed landing test, Definition 9464.48. It places recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate on the displayed landing surface before the open carrier row is considered.

Use recognizer status sharpness, Definition 9464.64. Once those rows are displayed, the remaining open issue is recognizer-to-carrier construction rather than a fixed-tuple StandardBridge route or a missing landing component.

Apply the carrier obstruction restatement, Definition 9464.77. It identifies the exact carrier row to be preserved and separates a possible finite obstruction from the bridge question.

Finally apply the no-StdBridge-upgrade theorem, Definition 9464.79. A recognizer-to-carrier row can feed only the carrier-map boundary; the fixed tuple route and standard reflection schema remain separate data. Therefore the open status is exactly the carrier-map preservation boundary stated above. □

Theorem 9464.81 (Finite BHist injection carrier-map final boundary)**.** *The weakened recognizer-to-carrier conjecture is settled only by one of two finite carrier-map outcomes. Either there is a carrier row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *along the same displayed route, or there is a finite obstruction to preserving exactly that row family. Missing landing rows, fixed-tuple* StandardBridge *premises, quotient carriers, host equality, and bridge assertions are outside this final boundary.*

Proof. Start with the recognizer closed status theorem, Definition 9464.80. It identifies the only open content after the landing rows have been exposed.

Apply the carrier-row dichotomy, Definition 9464.85. It gives precisely the two carrier-facing outcomes: preservation of the displayed row family, or a finite obstruction to preserving that same family.

Finally use the no-StdBridge-upgrade theorem, Definition 9464.79. This keeps the boundary inside the weakened recognizer-to-carrier conjecture and prevents the carrier-map settlement from becoming a bridge claim. □

Theorem 9464.82 (Finite BHist injection carrier-map local obstruction)**.** *Settling the weakened conjecture of Definition 9464.2 requires exactly one local carrier-map outcome. Either a carrier row preserves recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *along the same displayed route; or a finite obstruction shows that precisely those rows cannot be preserved by such a carrier map.*

The outcome is local to the displayed BEDC support surface. It is not a missing landing row, not a fixed-tuple StandardBridge *theorem, not quotient-carrier equality, and not a host-recognizer comparison.*

Proof. Start with the closed landing test, Definition 9464.48. It places recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist component readback, componentwise transport, and ChapterTasteGate on the displayed finite surface before the recognizer-to-carrier boundary is evaluated.

Use recognizer row exhaustion, Definition 9464.46. After those rows have landed, no further landing row remains available as the active conjectural content.

Apply the no-StdBridge-upgrade theorem, Definition 9464.79. It prevents a carrier-map preservation row from being read as a bridge assertion and keeps the fixed-tuple bridge route separate.

Finally apply the carrier-map final boundary, Definition 9464.81. The final boundary already has exactly two carrier-facing exits: preservation of the displayed row family, or a finite obstruction to preserving that same family. Therefore the local obstruction test is exhausted by the stated carrier-map outcome. □

Theorem 9464.83 (Finite BHist injection recognizer carrier refutation boundary)**.** *A closed recognizer-to-carrier landing for Definition 9464.2 must expose the fixed tuple route, regular recognizer row separation, carrier-map decomposition, row exhaustion, and construction-obstruction localization. Any route missing one of those finite rows lies outside the weakened conjecture surface rather than inside the recognizer-carrier refutation boundary.*

Proof. Start with obstruction locality and obstruction tightening, Definition 9464.50 and Definition 9464.51. They assign post-landing failure to recognizer-to-carrier construction and separate it from missing fixed-tuple data.

Use obstruction completeness and construction localization, Definition 9464.54 and Definition 9464.57. Together they require the landing rows to be present before a negative result can be counted as a construction obstruction.

Finally apply status sharpness, Definition 9464.64. It identifies the exact boundary as recognizer-to-carrier construction after the fixed tuple, recognizer separation, decomposition, exhaustion, and localization rows have been displayed. Therefore a route lacking one of those finite rows is not a refutation on the weakened surface. □

Theorem 9464.84 (Finite BHist injection recognizer carrier obstruction exhaustion)**.** *Once the fixed tuple route, regular recognizer row separation, carrier-map decomposition, row exhaustion, construction-obstruction localization, obstruction tightening, and status sharpness are displayed, the only remaining open recognizer-to-carrier content is a carrier row preserving recognizer acceptance, finite tuple extraction, readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*.*

Any route failing before those rows are exposed is outside the open conjecture surface. It is a missing finite-row landing route, not a recognizer-carrier obstruction, and it does not assert a StandardBridge *theorem.*

Proof. First expose the finite landing rows. Use the fixed tuple route and carrier-map decomposition together with Definition 9464.46; these results place recognizer acceptance, tuple extraction, readback, and componentwise transport on the displayed row surface.

Next apply obstruction locality and obstruction tightening, Definition 9464.50 and Definition 9464.51. They move any post-landing failure to the recognizer-to-carrier construction row itself and exclude missing landing data from that boundary.

Finally use Definition 9464.64 and Definition 9464.83. These identify the open status exactly at carrier construction after the displayed rows are present. Therefore the obstruction surface is exhausted by preserving recognizer acceptance, finite tuple extraction, readback, componentwise transport, NameCert data, and ChapterTasteGate, with no StdBridge claim introduced. □

Theorem 9464.85 (Finite BHist injection carrier-row dichotomy)**.** *The remaining finite test in Definition 9464.2 has exactly two admissible outcomes. Either there is a carrier row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*; or there is a finite obstruction to preserving that same row family. No third outcome is supplied by missing landing data, fixed-tuple* StandardBridge *premises, quotient carriers, host equality, or bridge assertion.*

Proof. Start with recognizer row exhaustion, Definition 9464.46, and the closed landing test, Definition 9464.48. They place recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate on the complete finite landing surface.

Apply the no-StdBridge-upgrade theorem, Definition 9464.79. It keeps the recognizer-to-carrier row separate from any fixed-tuple StandardBridge assertion.

Now use carrier obstruction exhaustion, Definition 9464.84, together with the carrier obstruction restatement, Definition 9464.77. These results leave exactly the carrier row preserving the displayed family, or a finite obstruction to preserving that same family.

Therefore the active conjecture is a finite dichotomy at the carrier row. Missing landing rows and bridge-side requirements are outside the dichotomy, while positive and negative settlements both remain carrier-facing. □

Theorem 9464.86 (Finite regular-language carrier-row test)**.** *Settling the open recognizer-to-carrier boundary in Definition 9464.2 requires exactly one finite regular-language carrier-row outcome. Either there is a carrier row preserving recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate*, or there is a finite obstruction to preserving precisely that row family. Missing landing data, a fixed-tuple* StandardBridge *premise, or a desired bridge theorem is outside this test.*

Proof. Start with the closed landing test, Definition 9464.48. It places recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate on the finite landing surface before the carrier row is considered.

Use status sharpness, Definition 9464.64. Once the landing surface is present, the open content is not a fixed-tuple bridge route and not a missing landing row; it is recognizer-to-carrier construction.

Apply the carrier obstruction restatement, Definition 9464.77. It gives the two permitted outcomes: a carrier row preserving the displayed family, or a finite obstruction to that same preservation.

Finally use the no-StdBridge-upgrade theorem, Definition 9464.79. It keeps the carrier-row test from becoming a StandardBridge assertion. Therefore the open boundary is exactly the finite regular-language carrier-row preservation test stated above. □

Theorem 9464.87 (Finite BHist injection carrier-map row locality)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *transport, and* ChapterTasteGate *rows are displayed, the only remaining strengthening of Definition 9464.2 is a carrier row preserving those seven rows. The boundary is therefore kernel-local at the recognizer-to-carrier map; it is not a missing landing row, fixed-tuple* StandardBridge *premise, quotient carrier, host equality, or bridge claim.*

Proof. Use the closed landing test, Definition 9464.48. It displays recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise transport, and ChapterTasteGate before any carrier-row strengthening is considered.

Apply recognizer row exhaustion and carrier obstruction locality, Definition 9464.46 and Definition 9464.50. These results place the post-landing obstruction at the recognizer-to-carrier construction row rather than at the displayed landing data.

Finally use the carrier obstruction restatement and the finite regular-language carrier-row test, Definition 9464.77 and Definition 9464.86. They identify the remaining positive strengthening as exactly a carrier row preserving the seven displayed rows, while the no-StdBridge-upgrade theorem keeps bridge assertions outside the weakened conjecture boundary. □

Theorem 9464.88 (Finite BHist injection carrier-map compatibility dichotomy)**.** *After recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *are displayed, the remaining weakened content of Definition 9464.2 has exactly two finite exits. Either a carrier row preserves that displayed family along the recognizer-to-carrier route, or a finite obstruction shows that no such preservation row is available inside the displayed BEDC-kernel surface. The dichotomy is a restatement of the carrier-map boundary and does not assert a fixed-tuple* StandardBridge *theorem.*

Proof. Start with the recognizer map obstruction and the closed landing boundary, Definition 9464.74 and Definition 9464.75. They place the recognizer, generated-recognizer, certificate-gate, tuple-extraction, readback, transport, and taste-gate rows on the displayed finite surface.

Use the carrier-map obstruction-witness test and carrier-row dichotomy, Definition 9464.78 and Definition 9464.85. These results identify the two carrier-facing outcomes: preservation by a carrier row, or a finite obstruction to such preservation.

Finally apply the no-StdBridge-upgrade theorem, Definition 9464.79. It keeps the dichotomy inside the weakened recognizer-to-carrier boundary and prevents the carrier-map restatement from becoming a bridge claim. □

Theorem 9464.89 (Finite BHist injection carrier-map compatibility row)**.** *The active weakened conjecture Definition 9464.2 can change status only by supplying or refuting one finite carrier-map compatibility row. That row must preserve recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *transport, componentwise* NameCert *transport, and* ChapterTasteGate *along the recognizer-to-carrier route. It is not a missing landing row, a fixed-tuple* StandardBridge *premise, or a bridge theorem.*

Proof. Start with the carrier-map compatibility dichotomy Definition 9464.88. It leaves exactly two exits: preservation by a carrier row or a finite obstruction to that same preservation row.

Use the closed landing boundary Definition 9464.75. The recognizer, generated-recognizer, certificate-gate, finite tuple, component readback, transport, and taste-gate rows are already displayed before the open carrier-map question is evaluated.

Finally apply the no-StdBridge-upgrade theorem Definition 9464.79. Since the weakened conjecture does not assert a bridge and does not ask for additional landing rows, its status can move only when the finite carrier-map compatibility row is either supplied or refuted. □

Theorem 9464.90 (Finite BHist injection carrier-map positive boundary)**.** *A positive close of the weakened recognizer-to-carrier boundary in Definition 9464.2 must exhibit one carrier row along the displayed route that preserves recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction,* BHist *component readback, componentwise* hsame *transport, componentwise* NameCert *transport, and* ChapterTasteGate*. The row is admissible only when those preserved data remain on the same carrier-facing surface; it is not a fixed-tuple* StandardBridge *assertion and it does not discharge the boundary by adding a detached landing row.*

Proof. Start with the carrier-map compatibility dichotomy, Definition 9464.88. Its positive exit is not arbitrary existence of a carrier; it is preservation of the displayed recognizer, tuple, readback, transport, and taste-gate family along the recognizer-to-carrier route.

Use the closed landing test, Definition 9464.48. It fixes recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, BHist readback, componentwise transport, and ChapterTasteGate as the surface already present before the carrier row is evaluated.

Finally apply the no-StdBridge-upgrade theorem, Definition 9464.79. The positive carrier row may preserve the landing family, but it cannot by itself assert a standard bridge or add a new landing premise outside the displayed carrier-facing surface. □

Theorem 9464.91 (Finite BHist injection recognizer rate refutation)**.** *A proposed* StandardBridge *for finite-BHist injection remains refutable whenever the recognizer route cannot display the carrier row, transport row, and readback row in the same finite tuple. The weakened conjecture Definition 9464.2 therefore stays open at the recognizer-to-carrier boundary unless a single displayed tuple carries recognizer acceptance, carrier preservation, componentwise transport, and* BHist *readback on the same carrier-facing surface.*

Proof. Start with the recognizer map obstruction, Definition 9464.74. It identifies the missing carrier-facing object as a finite compatibility row, not as an ambient bridge premise.

Use the carrier obstruction restatement, Definition 9464.77. It makes the positive and negative exits depend on preserving exactly the recognizer, tuple, readback, transport, and taste-gate rows.

Finally apply the carrier-map positive boundary, Definition 9464.90. A positive close requires one carrier row along the displayed route. If the carrier row, transport row, and readback row cannot be displayed in the same finite tuple, the proposed bridge has not reached that boundary and remains refutable by this finite-row test. □

Refutation route (if false). Find a chapter satisfying the displayed finite tuple, image, readback, fixed-tuple, and ChapterTasteGate premises for which a $\langle X \rangle$UpStdBridge theorem demonstrably cannot exist (e.g. because the chapter's NameCert obligation requires a non-standard reflection axis incompatible with the schema).

Part XIII

Project Governance

Chapter 9465

Project Governance

This paper is the BEDC theory source of truth, and amendments are made in place; see Chapter 9466 for the amendment policy and `HOW_INCREMENT_WORKS.md` for the companion editing recipe.

The repository remains intentionally small: one paper tree and one Lean tree. Lean-facing theorem status is rendered by the `\leanchecked` family of macros at each primary site, and the audit `lean4/scripts/bedc_ci.py audit` enforces that every macro target resolves to a real declaration in `lean4/BEDC/`.

Remark 9465.1. *The master PDF should remain the public artifact. The source tree should remain the authoritative editable artifact.*

Chapter 9466

Theory Amendment Policy

This paper presents the BEDC theory. When the theory is extended or refined, amendments are made in place: new definitions, theorems, and policies are inserted in their thematic chapter, and replaced phrasings are removed at the time of replacement.

New material belongs in the chapter whose subject it advances. Concrete-instance refinements stay with concrete instances or concrete hardening; proof interfaces and export contracts stay with proof obligations; formalization-facing guidance stays with the implementation-facing contract. The companion file `HOW_INCREMENT_WORKS.md` records the repository recipe for registering claims, aligning Lean targets, and updating status tables.

When an amendment changes the active reading of an existing object, the active label remains the citation target for that object. Theorem- and definition-level labels remain attached to the active formulation.

Deprecation is handled by replacement rather than archival prose. If a definition, theorem phrasing, or policy interface is replaced, the replaced wording is removed from the paper and the active wording takes its place in the relevant chapter. External references to deprecated names should therefore resolve through the active definition's label.

9466.1 Self-Compilation Surface Governance

This section codifies which strata of the self-compilation surface BEDC formally claims and which strata BEDC explicitly does not claim. The structural description of the surface is in chapter 9235; this section records the corresponding governance rules.

9466.1.1 Claimed Strata

BEDC formally claims:

- **L1: Static paper-Lean alignment.** Every `\leanchecked{X}` marker in the paper corresponds to an actually existing Lean target X in `lean4/BEDC/`. This is enforced by `bedc_ci.py audit` as a CI gate.
- **L4-mid: Internal manifest.** A finite, hand-curated subset of paper-anchored theorems are registered in `BEDC.Manifest` as `(prop, proof)` pairs. The manifest's wellformedness is the Lean kernel's type-check of the list.

9466.1.2 Explicitly Not Claimed

BEDC explicitly does not claim:

- **L4a: Internal axiom-free completeness theorem.** A statement of the form "every paper claim has an axiom-free closed term" inside BEDC is blocked by Tarski undefinability and Gödel's second incompleteness theorem. BEDC sits inside the host CIC and cannot prove Con(CIC) internally. The verification axis is reported externally by `bedc_ci.py axiom-purity`, not as a Lean theorem.
- **L4c: Theory-axis ↔ verification-axis bijection.** The two-axis closure tracking, theory closure level versus Lean verification status, is deliberately kept independent. Paper closure does not require Lean verification, and Lean verification does not produce paper closure. A fixed point between the axes is rejected by this policy; the `axis-confusion` CI gate enforces this.

9466.1.3 Open Strata

BEDC neither claims nor rejects:

- **V2 reconstruction.** Subject reduction, confluence, and other metatheoretic results about the MetaCIC reconstruction are in scope as ordinary theory work without governance commitment to completion.
- **L4b: Internalized audit.** Moving the external Python audits to Lean elab-time commands is operationally desirable but does not change the theoretical surface; it is treated as engineering refactor.
- **Reflection bridge.** A bridge between `BEDC.MetaCIC.HasType` and host Lean's kernel `#check` is research-stage and not on a delivery commitment.

9466.2 Pure-CIC Tactic Discipline

BEDC proofs are constrained to a subset of Lean tactics chosen to preserve the axiom-purity invariant. This section records which tactics are admitted, which are forbidden, and why.

9466.2.1 Admitted Tactics

The following tactics are admitted in BEDC proofs:

- Structural elimination: `match`, `cases`, `induction`
- Direct construction: `exact`, `apply`, `refine`
- Reduction: `rfl`, `unfold`, `show`
- Hypothesis manipulation: `intro`, `intros`, `rename_i`
- Discharging absurd cases: `contradiction`, `Bool.noConfusion`
- Composition: `constructor`

These tactics either reduce to pure CIC term construction or to structural operations on inductive types. Their proof outputs are checkable by `#print axioms` as "does not depend on any axioms".

9466.2.2 Forbidden Tactics

The following tactics are forbidden in BEDC proofs:

- `simp`, `simp_all`, `simp [...]`, `simp [...] at h`
- `decide`
- `omega`
- Tactics that invoke stdlib decidability lemmas with `propext` dependencies in their transitive closure

These tactics silently pull in axioms from Lean stdlib, most commonly `propext`, sometimes `Classical.choice` or `Quot.sound`, through automated simplification or decidability rewrites. A proof that uses `simp [foo] at h` typically inherits all axioms touched by the simp lemma's transitive provenance, even when those axioms are irrelevant to the immediate term.

9466.2.3 Audit Enforcement

The discipline is enforced by `bedc_ci.py axiom-purity -strict`. This audit invokes `#print axioms` on every theorem in `lean4/BEDC/` and rejects any theorem whose transitive axiom set contains `Classical.choice`, `Quot.sound`, or `propext`. The invariant is reported as

$$\text{theorems} = \text{pure} \wedge \text{impure} = 0.$$

A theorem that violates the discipline fails this audit and is rejected at PR merge time.

9466.2.4 Historical Origin

This rule originates in the 2026-05-11 PR #64 incident: a `simp`-based soundness proof for a checker theorem in `BEDC.MetaCIC.Checker` appears as `propext`-dependent under `axiom-purity -strict`. The accepted proof shape for that theorem uses structural elimination on the underlying inductive type and satisfies `#print axioms ... does not depend on any axioms`. The incident establishes that BEDC's axiom-purity invariant constrains not only declared axioms but also tactic choices, and the tactic discipline above codifies that constraint.

9466.2.5 Position

The discipline is not a stylistic preference. It is a load-bearing condition for the project's stated goal of pure CIC formalization. Proofs that violate the discipline are not BEDC proofs even when they type-check, because they introduce theoretical commitments BEDC does not endorse.

Chapter 9467

Roadmap for Re-deriving Mature Mathematics

9467.1 Research thesis

Given the BEDC finite kernel

$$b_0, b_1 \;\to\; \mathsf{Hist} \;\to\; \mathsf{msame}, \mathsf{hsame} \;\to\; \mathsf{Ext}, \mathsf{Cont} \;\to\; \mathsf{Ask}, \mathsf{Sig} \;\to\; \mathsf{Pkg}, \mathsf{Gap} \;\to\; \mathsf{NameCert},$$

the project re-derives mature mathematical objects N as licensed naming certificates

$$\mathsf{NameCert}(N) \;=\; (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

A mature object such as $\mathbb{N}$, $\mathbb{R}$, S^1, a metric space, or a category is admitted only as the stable interface of a generative behaviour, never as a primitive input.

9467.2 Closure and verification axes

Every claim about a derived mature object is read on two independent axes. The closure axis records what the paper has reconstructed; the verification axis records the audit state of the corresponding Lean target. The discipline is recorded in chapter 9203.

$$\boxed{(c, v) \in \mathsf{TheoryClosure} \times \mathsf{FormalStatus}.}$$

The closure axis grades are $\mathsf{seed} < \mathsf{obligation} < \mathsf{scopedClosed} < \mathsf{publicClosed} < \mathsf{bridgedClosed} < \mathsf{maturePackageClosed}$; the verification axis grades are $\mathsf{unformalized} < \mathsf{formalTarget} < \mathsf{encodedDef} < \mathsf{scaffoldChecked} < \mathsf{theoremChecked} < \mathsf{auditClean} < \mathsf{axiomClean}$, with $\mathsf{bridgeChecked}$ as the bridge-axis verification level. Neither axis determines the other.

9467.3 Anti-primitive principle

Mature constructs are not BEDC primitives:

$$\boxed{\text{equality, function, } \mathbb{N}, \text{ addition, limit, } \mathbb{R}, \text{ circle, fold, set, type — none are kernel inputs.}}$$

They are admitted only as licensed names. Any host-level `Nat`, `List`, definitional equality, `Quot.sound`, `Classical.choice`, or `propext` that leaks into the BEDC public surface is a contamination, not a derivation. The Lean discipline (mathlib-free, 0 axiom, 0 sorry, no `Classical`, no `Quot`, no `propext`) is the mechanical enforcement of this principle.

9467.4 Russell-style reconstruction spine

BEDC reads Russell as reconstruction discipline, not as an imported logicist foundation. The imported pressure is that mature mathematical grammar must be analysed before it is admitted: relations remain prior to object names, definite descriptions must expand into obligations, class-like expressions are contextual rather than automatically objectual, type or level passages do not collapse, and no downward use of an upward name is licensed without a certificate. The capstone formulation is recorded in chapter 9457.

In BEDC notation, $X^{\uparrow}$ is therefore never the host object X itself. It is the $\mathsf{NameCert}$-licensed public reading of X:

$$X^{\uparrow} \quad \text{means} \quad \mathsf{NameCert}(X) = (\mathsf{SourceSpec}_X, \mathsf{PatternSpec}_X, \mathsf{ClassifierSpec}_X, \mathsf{StabilityCert}_X, \mathsf{LedgerPolicy}_X).$$

The fivefold reading is:

kernel role	**certificate role**	**Russell discipline**
probe or question	source entry	description expansion
history	source material	contextual class reading
observation relation	pattern and classifier	relation first
result mark	stability-consumable result	stratification
evidence token	ledger policy	witnessed descent

This gives the analysis spine a governance role. The path

$$\mathsf{Nat}^{\uparrow} \to \mathsf{RelationCalculus}^{\uparrow} \to \mathsf{Order}^{\uparrow} \to \mathsf{Series}^{\uparrow} \to \mathsf{Segment}^{\uparrow}, \mathsf{Cut}^{\uparrow} \to \mathsf{Rat}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{Limit}^{\uparrow} \to \mathsf{Conti}$$

is not a list of primitives. It is a list of names whose admission requires certificates at the appropriate layer.

9467.5 Layer index

The path from kernel to mature mathematics is layered. Every layer depends only on layers below it.

L0 Marks: b_0, b_1.

L1 Histories: Hist, msame, hsame.

L2 Generative dynamics: Ext, Cont.

L3 Asking and signature: Ask, Sig, $\mathsf{sameSig}$.

L4 Package relations and gaps: Pkg, Gap, $\mathsf{ExactGlobalizeBase}$.

L5 Naming and stability: $\mathsf{NameCert}$, $\mathsf{StabilityCert}$, $\mathsf{LedgerPolicy}$.

L6 Finite data: $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{List}^{\uparrow}$.

L7 Arithmetic: $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$.

L8 Algebra: $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{AlgClosure}^{\uparrow}$, $\mathsf{GaloisExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{SplittingField}^{\uparrow}$, $\mathsf{SeparableExt}^{\uparrow}$, $\mathsf{Transcendence}^{\uparrow}$, $\mathsf{HopfAlg}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{VermaModule}^{\uparrow}$.

L9 Order, logic, set, type: Preorder$^\uparrow$, Poset$^\uparrow$, Lattice$^\uparrow$, SetLike$^\uparrow$, TypeLike$^\uparrow$, FirstOrder$^\uparrow$, ModelTheory$^\uparrow$, Computable$^\uparrow$, LambdaCalc$^\uparrow$, Partition$^\uparrow$, Matroid$^\uparrow$, Ramsey$^\uparrow$, NetworkFlow$^\uparrow$, Matching$^\uparrow$, LPDuality$^\uparrow$, ShortestPath$^\uparrow$, SpanningTree$^\uparrow$, ErrorCode$^\uparrow$, Hash$^\uparrow$, PublicKey$^\uparrow$, ZeroKnowledge$^\uparrow$.

L10 Topology, metric, completion, real: Met$^\uparrow$, Topology$^\uparrow$, Completion$^\uparrow$, Real$^\uparrow$.

L11 Analysis: Calculus$^\uparrow$, Measure$^\uparrow$, FunctionalAnalysis$^\uparrow$, Ode$^\uparrow$, Pde$^\uparrow$, DynSystem$^\uparrow$, Ergodic$^\uparrow$, ProbSpace$^\uparrow$, RandomVar$^\uparrow$, Distribution$^\uparrow$, Independence$^\uparrow$, CondExp$^\uparrow$, MarkovChain$^\uparrow$, Martingale$^\uparrow$, Brownian$^\uparrow$, Entropy$^\uparrow$, CStarAlg$^\uparrow$, VonNeumannAlg$^\uparrow$, OperatorIdeal$^\uparrow$, SpectralTheorem$^\uparrow$, Nuclear$^\uparrow$, Factor$^\uparrow$, QuantumState$^\uparrow$, Observable$^\uparrow$, UnitaryGroup$^\uparrow$, DensityMatrix$^\uparrow$, QuantumChannel$^\uparrow$, Entanglement$^\uparrow$, NewtonIteration$^\uparrow$, Interpolation$^\uparrow$, Approximation$^\uparrow$, Fft$^\uparrow$, Quadrature$^\uparrow$, FisherInfo$^\uparrow$, StatManifold$^\uparrow$, Estimator$^\uparrow$, HypothesisTest$^\uparrow$, Bayesian$^\uparrow$, Hypergeometric$^\uparrow$, Bessel$^\uparrow$, ThetaFunction$^\uparrow$, ModularForm$^\uparrow$, ClebschGordan$^\uparrow$.

L12 Geometry, category, homology: Cat$^\uparrow$, Functor$^\uparrow$, NatTrans$^\uparrow$, Manifold$^\uparrow$, Bundle$^\uparrow$, Homology$^\uparrow$, Cohomology$^\uparrow$, RiemannianMetric$^\uparrow$, Connection$^\uparrow$, Curvature$^\uparrow$, DiffForm$^\uparrow$, DeRham$^\uparrow$, Symplectic$^\uparrow$, Contact$^\uparrow$, LieGroup$^\uparrow$, LieAlgebra$^\uparrow$, ExpMap$^\uparrow$, AdjointRep$^\uparrow$, RootSystem$^\uparrow$, WeylGroup$^\uparrow$, BilinForm$^\uparrow$, Clifford$^\uparrow$, SpinGroup$^\uparrow$, PinGroup$^\uparrow$, PreSheaf$^\uparrow$, Sheaf$^\uparrow$, RingedSpace$^\uparrow$, Scheme$^\uparrow$, AffineVar$^\uparrow$, ProjectiveVar$^\uparrow$, Homotopy$^\uparrow$, SimplicialSet$^\uparrow$, ModelCat$^\uparrow$, InfCat$^\uparrow$, DerivedCat$^\uparrow$, AbelianCat$^\uparrow$, DerivedFunctor$^\uparrow$, SpectralSeq$^\uparrow$, TriangulatedCat$^\uparrow$, MonoidalCat$^\uparrow$, EnrichedCat$^\uparrow$, Topos$^\uparrow$, TopGroup$^\uparrow$, TopVecSpace$^\uparrow$, AffineSpace$^\uparrow$, ProjectiveSpace$^\uparrow$, ConvexSet$^\uparrow$, Polytope$^\uparrow$, VectorBundle$^\uparrow$, SimplicialComplex$^\uparrow$, PersistentHom$^\uparrow$, Mapper$^\uparrow$, Knot$^\uparrow$, JonesPolynomial$^\uparrow$, ThreeManifold$^\uparrow$, Stack$^\uparrow$.

L13 Number theory and spectra: Prime$^\uparrow$, Padic$^\uparrow$, Adele$^\uparrow$, Zeta$^\uparrow$, NumField$^\uparrow$, RingOfIntegers$^\uparrow$, IdealClass$^\uparrow$, DirichletUnit$^\uparrow$, Dedekind$^\uparrow$, LocalField$^\uparrow$, Cyclotomic$^\uparrow$, EllipticCurve$^\uparrow$, Iwasawa$^\uparrow$, ClassField$^\uparrow$, Regulator$^\uparrow$.

Bridges Cross-domain bridges (section 9467.17): StoneDuality$^\uparrow$, GelfandDuality$^\uparrow$, PontryaginDuality$^\uparrow$, TannakaKrein$^\uparrow$, CurryHoward$^\uparrow$, RiemannHilbert$^\uparrow$, KoszulDuality$^\uparrow$, Langlands$^\uparrow$, NoetherSymmetry$^\uparrow$, GeomQuantization$^\uparrow$, DeformQuantization$^\uparrow$, ChernWeil$^\uparrow$, AtiyahSinger$^\uparrow$, HodgeBridge$^\uparrow$, MirrorSymmetry$^\uparrow$.

L14 Mature-system bridge: standard models, mathlib bridge, conservativity reports.

The layer index is also the reviewer's priority order: directed development MUST advance the lowest open layer before nibbling above.

9467.6 L0–L5 finite kernel checked baseline

The kernel proof DAG, in dependency order, is

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{CompGap} \to \mathsf{NameCert}.$$

Before any derived interface is attempted, every node above must hold a checked verification target under the discipline of section 9467.2: msame and hsame as equivalences with no-confusion, Ext and Cont with associativity and unit laws, Ask and Sig with totality and determinism under the relevant policies, Pkg and Gap with their soundness/coverage/separation theorems, and NameCert as the universal admission rule.

9467.7 L4 exact globalization contract

For any classification interface, exactness is recorded as a structured datum, not as natural language. The base contract is

$$\mathsf{ExactGlobalizeBase}(\Pi, D) \;=\; \big(\text{coverage} \wedge \text{base soundness} \wedge \text{base completeness}\big),$$

with the local classification $\iff$ as a derived corollary. Every downstream module that wants to claim "visible packages classify histories exactly" must cite $\mathsf{ExactGlobalizeBase}(\Pi, D)$, never the prose phrase.

9467.8 L6 finite data certificates

For each of $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, the obligations are: a carrier predicate over Hist; a classifier whose $\mathsf{StabilityCert}$ supplies reflexivity, symmetry, transitivity, and carrier transport; constructor and case-analysis stability; and ledger entries for whatever is hidden by the carrier predicate (tag bits, length unary, etc).

The finite-data public-object certificate scope is the boolean history $\mathsf{SemanticNameCert}$, the constructor-tagged optional-history certificate, the explicit ordered-product bridge together with its envelope certificate, the tagged sum $\mathsf{SemanticNameCert}$, and the list spine-bridge $\mathsf{SemanticNameCert}$. These scopes are the phase objects for $\mathsf{Bool}^{\uparrow}$ through $\mathsf{List}^{\uparrow}$ when a chapter separates an envelope certificate from the public constructor interface.

9467.9 L7 arithmetic

- $\mathsf{Nat}^{\uparrow}$: unary repetition carrier; zero, successor, induction, recursion. The unary history E_1-tower is a seed, not a complete arithmetic.
- $\mathsf{Add}^{\uparrow}$: addition as a Cont-driven operation, with associativity, commutativity, zero law.
- $\mathsf{Mul}^{\uparrow}$: repeated addition, with distributivity, associativity, commutativity, one law.
- $\mathsf{Order}^{\uparrow}$: comparison from the unary tower, with reflexivity, antisymmetry, totality, well-foundedness.
- $\mathsf{Int}^{\uparrow}$: signed pairs (a, b) with $(a, b) \sim (c, d) \iff a + d = c + b$; ordered ring structure.
- $\mathsf{Rat}^{\uparrow}$: certified rational histories with positive unary denominator; ordered field structure.

9467.10 L8 algebra

The algebra hierarchy is

$$\mathsf{Magma}^{\uparrow} \to \mathsf{Semigroup}^{\uparrow} \to \mathsf{Monoid}^{\uparrow} \to \mathsf{Group}^{\uparrow} \to \mathsf{AbGroup}^{\uparrow} \to \mathsf{Ring}^{\uparrow} \to \mathsf{CommRing}^{\uparrow} \to \mathsf{Field}^{\uparrow} \to \mathsf{Module}^{\uparrow}$$

Each layer ships a carrier, a classifier shared with the layer above, operation stability obligations, congruence laws, and a ledger of normalised expression provenance. $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$ live above $\mathsf{VecSpace}^{\uparrow}$.

The field side admits a parallel extension hierarchy: $\mathsf{FieldExt}^{\uparrow}$ packages a field embedding with the larger field as a vector space over the smaller; $\mathsf{AlgClosure}^{\uparrow}$ admits the closure under polynomial roots; $\mathsf{SeparableExt}^{\uparrow}$, $\mathsf{SplittingField}^{\uparrow}$, $\mathsf{GaloisExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$ admit normality, root-splitting, simultaneous separability and normality, and the automorphism group fixing the base

field; $\mathsf{Transcendence}^{\uparrow}$ admits the transcendence degree. $\mathsf{HopfAlg}^{\uparrow}$ admits a bialgebra with antipode, sitting above $\mathsf{Ring}^{\uparrow}$ and $\mathsf{TensorProduct}^{\uparrow}$.

9467.11 L9 order, logic, set-like, type-like

$\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$ are licensed by stable comparison behaviour. $\mathsf{SetLike}^{\uparrow}$ is licensed by classifier-based membership: the membership predicate is generated, never assumed; subset is implication of membership; comprehension is restricted to certified families; extensionality is a certificate field, not a primitive. $\mathsf{TypeLike}^{\uparrow}$ is the corresponding carrier-plus-classifier-plus-transport interface.

The logic side admits a first-order signature, formula carrier, and deduction calculus through $\mathsf{FirstOrder}^{\uparrow}$, with the soundness and completeness data; $\mathsf{ModelTheory}^{\uparrow}$ admits a structure for that signature with satisfaction and elementary equivalence. The computability side admits $\mathsf{Computable}^{\uparrow}$ as Turing-machine-step-bounded partial functions and $\mathsf{LambdaCalc}^{\uparrow}$ as the inductive carrier of λ-terms with β-reduction and α-equivalence. The combinatorial side admits $\mathsf{Partition}^{\uparrow}$ as Young-diagram-presented integer partitions, $\mathsf{Matroid}^{\uparrow}$ as exchange-axiom independent-set families, and $\mathsf{Ramsey}^{\uparrow}$ as the monochromatic-subset bound under finite colouring.

9467.12 L10 topology, metric, completion, real

This layer crosses the line between finite-history objects and countable-name objects. The architectural cost has to be paid up front.

Architectural gate: countable names

A finite history Hist cannot itself encode an infinite object. To license $\mathsf{Real}^{\uparrow}$ and onward, the project admits *certified countable names*: a name is a function whose finite observations are individually certified by an existing carrier, and whose stability is the pointwise classifier of those observations. Concretely, the milestone $\mathsf{StreamName}^{\uparrow}$ supplies a carrier of the form

$$\mathsf{StreamCarrier}_X(s) \;\equiv\; \forall n,\ \mathsf{UnaryHistory}(n) \to X\mathrm{Carrier}(s\,n),$$

with the pointwise classifier

$$\mathsf{StreamSame}_X(s,t) \;\equiv\; \forall n,\ \mathsf{UnaryHistory}(n) \to X\mathrm{Same}(s\,n,\, t\,n),$$

and stability obligations: pointwise reflexivity / symmetry / transitivity inherited from the point certificate, carrier transport along the pointwise classifier, reindexing stability under hsame, mapped-stream stability, and an inhabited stream from a constant certified point.

This is not optional. Without $\mathsf{StreamName}^{\uparrow}$, the L10 layer is not feasible under the discipline.

Construction order

The interior of L10 is itself a sub-roadmap. The construction order, with each step depending only on earlier steps, is

1. $\mathsf{DyadicRatCore}$: dyadic rationals over $\mathsf{Int}^{\uparrow}$ and the unary exponent tower. Carrier: signed mantissa with unary exponent. Classifier: dyadic equality by common-exponent scaling, encoded as a predicate, not via `Quot`. Obligations: arithmetic stability, monotone 2^{-n} radii, distance and absolute value stability. Estimated 25–40 rounds.

2. $\mathsf{StreamName}^{\uparrow}$ (architectural gate above). Estimated 8–12 rounds.
3. $\mathsf{RegSeqRat}$: Bishop-style regular Cauchy sequences over dyadic rationals. Carrier: a stream over $\mathsf{DyadicRatCore}$ satisfying $|x_n - x_m| \leq 2^{-n} + 2^{-m}$. Classifier: regular-sequence closeness via dyadic ε-tolerance. Obligations: regularity carrier, refl/symm/trans by ε halving, shift and tail stability. Estimated 30–45 rounds.
4. $\mathsf{Real}^{\uparrow}$: real names sealed as $\mathsf{RegSeqRat}$ data with classifier closeness. Constant dyadic embedding respects classifier. No claim of completeness yet. Estimated 20–35 rounds.
5. $\mathsf{RealAlgOrder}$: addition, negation, multiplication, absolute value, nonnegative order, strict apartness on $\mathsf{Real}^{\uparrow}$. Order is constructive: prefer apartness over classical trichotomy. Estimated 35–60 rounds.
6. $\mathsf{Met}^{\uparrow}$: certified metric structure over a parameterised point carrier, with a $\mathsf{Real}^{\uparrow}$-valued distance, classifier-based identity of indiscernibles, symmetry, triangle inequality. Estimated 12–20 rounds.
7. $\mathsf{S}^{1\uparrow}$: S^1 as the certified subspace of $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ with $x^2 + y^2 \approx 1$ under $\mathsf{RealClassifier}$. No angle parametrisation yet. Estimated 20–35 rounds.
8. $\mathsf{TotallyBounded}^{\uparrow}$: total boundedness as explicit dyadic ε-net data, witnessed via $\mathsf{ProbeBundle}$ induction. Not open-cover compactness. Estimated 18–30 rounds.
9. $\mathsf{CompleteMetric}^{\uparrow}$: metric completeness as convergence of regular Cauchy point names; uniqueness of limits conditional on separatedness. Estimated 25–40 rounds.
10. $\mathsf{CompactMetric}^{\uparrow}$: $\mathsf{CompactMetricCarrier}(X) \equiv \mathsf{MetricSpaceCarrier}(X) \wedge \mathsf{CompleteCarrier}(X) \wedge \mathsf{TotallyBoundedCarrier}(X)$. Compact $[0,1]$ and compact S^1 are exhibited via explicit dyadic nets. Estimated 12–20 rounds for the definition; 40+ for product / subspace theorems.
11. $\mathsf{ContinuousMap}^{\uparrow}$: uniformly continuous maps with explicit modulus, classifier-based equality on source carrier points, identity and composition stability. Estimated 20–35 rounds.

The L10 sub-roadmap totals roughly 250–400 rounds. The honest first reportable target is finishing items 1–4, which produces the project's first fully licensed infinite mature-object name: $\mathsf{Real}^{\uparrow}$.

Constructive constraints to honour

The standard equivalences below are not generally derivable in this discipline and must not be claimed as such:

- Dedekind-cut construction of $\mathbb{R}$.
- Open-cover compactness as a general theorem on $\mathsf{CompactMetric}^{\uparrow}$.
- Sequential compactness without choice-like data.
- Heine-Borel as a general result on subsets of $\mathsf{Real}^{\uparrow}$.
- Trichotomy of constructive reals.
- Quotient-by-equivalence-class as a primitive.

Workable substitutes are the constructive analogues: located reals, total boundedness with explicit nets, separated metric spaces with explicit completion data, apartness in place of classical trichotomy, and classifier predicates in place of quotient types.

9467.13 L10 topology continued and L11 analysis

$\mathsf{Topology}^{\uparrow}$ is licensed by an open-set classifier predicate, neighbourhood interface, finite intersection and arbitrary union closure, continuous maps stable under the classifier, product and quotient topologies via the existing product / classifier mechanisms. $\mathsf{Completion}^{\uparrow}$ generalises the regular-Cauchy completion mechanism beyond $\mathsf{Rat}^{\uparrow}$ to any metric carrier, with operation descent recorded in the ledger.

$\mathsf{Calculus}^{\uparrow}$ admits derivative as a stable local linear approximation, integral via completion of finite sums or via a measure-based interface, with the fundamental theorem deferred until the measure interface is stable. $\mathsf{Measure}^{\uparrow}$ admits stable event algebra, σ-stability, integration, conditional expectation, L^p spaces. $\mathsf{FunctionalAnalysis}^{\uparrow}$ admits normed and inner-product spaces, Banach and Hilbert completion, bounded and compact operators, spectrum, with the spectral theorem as a major target.

The differential side admits $\mathsf{Ode}^{\uparrow}$ as systems $\dot{x} = F(t, x)$ with local existence and uniqueness on Lipschitz vector fields, $\mathsf{Pde}^{\uparrow}$ as classified relations among partial derivatives on a manifold carrier, $\mathsf{DynSystem}^{\uparrow}$ as one-parameter flows on phase-space carriers, and $\mathsf{Ergodic}^{\uparrow}$ as measure-preserving transformations with their invariant decomposition. The probability side admits $\mathsf{ProbSpace}^{\uparrow}$ as a measure space of total mass 1, $\mathsf{RandomVar}^{\uparrow}$ as a measurable function from a probability space, $\mathsf{Distribution}^{\uparrow}$ as the pushforward measure on the target, $\mathsf{Independence}^{\uparrow}$ as joint-distribution factorisation, $\mathsf{CondExp}^{\uparrow}$ as the L^2 projection onto a sub-σ-algebra, $\mathsf{MarkovChain}^{\uparrow}$ and $\mathsf{Martingale}^{\uparrow}$ as adapted sequences with the Markov / martingale increment law, $\mathsf{Brownian}^{\uparrow}$ as the continuous-time Gaussian-increment process with continuous sample paths, and $\mathsf{Entropy}^{\uparrow}$ as the Shannon and differential entropies.

9467.14 L12 categories, geometry, homology

Categories

$\mathsf{Cat}^{\uparrow}$ admits objects as certified carriers, morphisms as records over $\mathsf{ContinuousMap}^{\uparrow}$ or other certified maps, identity and composition stability, associativity and unit laws up to morphism classifier. Object equality is never extensional: subcategory inclusion is data, isomorphism is a record. $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ extend the same pattern.

Differential geometry

$\mathsf{Manifold}^{\uparrow}$ admits charts, atlas compatibility, gluing. $\mathsf{Bundle}^{\uparrow}$ admits local trivialisation, transition functions, cocycle conditions. $\mathsf{RiemannianMetric}^{\uparrow}$ admits a smooth field of inner-product certificates on tangent fibres. $\mathsf{Connection}^{\uparrow}$ admits a covariant derivative on a fibre bundle, with linearity in the section argument and Leibniz against scalar functions. $\mathsf{Curvature}^{\uparrow}$ admits the antisymmetric bracket of a connection, with Bianchi identities and tensorial behaviour under section change. $\mathsf{DiffForm}^{\uparrow}$ admits antisymmetric multilinear maps on the tangent bundle, with wedge product. $\mathsf{DeRham}^{\uparrow}$ admits the exterior-derivative chain complex with $d^2 = 0$. $\mathsf{Symplectic}^{\uparrow}$ admits a closed nondegenerate 2-form on an even-dimensional manifold. $\mathsf{Contact}^{\uparrow}$ admits a maximally nonintegrable hyperplane field on an odd-dimensional manifold.

Lie theory

$\mathsf{LieGroup}^{\uparrow}$ admits a group whose underlying carrier is a manifold and whose multiplication and inverse are constructively smooth. $\mathsf{LieAlgebra}^{\uparrow}$ admits a vector space with antisymmetric bracket satisfying Jacobi. $\mathsf{ExpMap}^{\uparrow}$ admits the canonical $\mathfrak{g} \to G$ map for a connected Lie-group carrier. $\mathsf{AdjointRep}^{\uparrow}$ admits the conjugation action Ad together with its differential ad. $\mathsf{RootSystem}^{\uparrow}$ admits a finite reflection-closed set of nonzero vectors with integer Cartan pairings. $\mathsf{WeylGroup}^{\uparrow}$ admits the finite reflection group generated by simple roots.

Clifford and geometric algebra

$\mathsf{BilinForm}^{\uparrow}$ admits a bilinear pairing on a module carrier with symmetry, antisymmetry, and non-degeneracy classifiers. $\mathsf{Clifford}^{\uparrow}$ admits the universal associative algebra modulo $v \cdot v = q(v)$. $\mathsf{SpinGroup}^{\uparrow}$ admits the connected double cover of the orthogonal group as a subgroup of the Clifford-algebra units. $\mathsf{PinGroup}^{\uparrow}$ admits the double cover of the full orthogonal group, extending Spin by a reflection generator.

Algebraic geometry

$\mathsf{PreSheaf}^{\uparrow}$ admits a contravariant functor from the open-sets category to a target category, with restriction maps respecting composition and identity. $\mathsf{Sheaf}^{\uparrow}$ admits a presheaf satisfying locality and gluing over open covers. $\mathsf{RingedSpace}^{\uparrow}$ admits a topological carrier with a sheaf of commutative rings whose stalks form local rings. $\mathsf{Scheme}^{\uparrow}$ admits a locally ringed space covered by spectra of commutative rings. $\mathsf{AffineVar}^{\uparrow}$ admits the zero-locus of a finite system of polynomial equations in affine n-space. $\mathsf{ProjectiveVar}^{\uparrow}$ admits the zero-locus of a homogeneous polynomial ideal in projective n-space.

Homology and homotopy

$\mathsf{Homology}^{\uparrow}$ admits chain complex, kernel/cokernel, exact sequence, derived functor. $\mathsf{Cohomology}^{\uparrow}$ admits the dual cochain complex with cup product. $\mathsf{Homotopy}^{\uparrow}$ admits the equivalence-of-paths relation between continuous maps under continuous deformation. $\mathsf{SimplicialSet}^{\uparrow}$ admits a contravariant functor from the simplex category Δ to certified sets, with face and degeneracy operators satisfying the simplicial identities. $\mathsf{ModelCat}^{\uparrow}$ admits a category with three distinguished classes of morphisms (cofibrations, fibrations, weak equivalences) satisfying lifting and factorisation axioms. $\mathsf{InfCat}^{\uparrow}$ admits a simplicial-set carrier satisfying inner-horn lifting (a quasicategory). $\mathsf{DerivedCat}^{\uparrow}$ admits the localisation of a chain-complex category at quasi-isomorphisms.

9467.15 L13 number theory and spectra

$\mathsf{Prime}^{\uparrow}$ admits divisibility, irreducibility, primality, gcd, Euclidean algorithm, unique factorisation. $\mathsf{Padic}^{\uparrow}$ admits the p-adic valuation, p-adic metric, completion to a local field. $\mathsf{Adele}^{\uparrow}$ admits the local-global product formula, adele-like interface. $\mathsf{Zeta}^{\uparrow}$ admits dynamical zeta, Euler product, Dirichlet series, with analytic continuation deferred to a stable analysis substrate. Riemann-hypothesis-shaped statements remain theorem targets, never naming-seed assertions.

The deeper number-theoretic side admits $\mathsf{NumField}^{\uparrow}$ as a finite field extension of $\mathsf{Rat}^{\uparrow}$; $\mathsf{RingOfIntegers}^{\uparrow}$ as the integral closure of $\mathsf{Int}^{\uparrow}$ in such a number field; $\mathsf{IdealClass}^{\uparrow}$ as the quotient group of fractional ideals modulo principals over a Dedekind domain; $\mathsf{DirichletUnit}^{\uparrow}$ as the finitely generated abelian unit-group structure with Dirichlet rank; $\mathsf{Dedekind}^{\uparrow}$ as the integrally closed Noetherian domain

with maximal nonzero primes; $\mathsf{LocalField}^{\uparrow}$ as a complete discretely valued field with finite residue field; $\mathsf{Cyclotomic}^{\uparrow}$ as the splitting field of $X^n - 1$; and $\mathsf{EllipticCurve}^{\uparrow}$ as a smooth projective genus-one curve with chosen base point.

9467.16 L14 conservative bridge to mature systems

For every certified mature object N a bidirectional bridge is established:

$$\mathsf{BEDC}(N) \longrightarrow \mathrm{Std}(N), \qquad \mathrm{Std}(N) \longrightarrow \mathsf{BEDC}(N).$$

The bridge proves at minimum: BEDC certificates interpret as standard objects; standard objects encode as BEDC certificates; basic theorems hold on both sides; no host primitive enters the BEDC public surface. The bridge to Lean / mathlib in particular records mapping tables, theorem correspondences, proof dependency maps, unsupported features, and a contamination audit.

9467.17 Cross-domain bridges via the unified kernel

Because every certificate in this project is read off the same finite kernel

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask}, \mathsf{Sig} \to \mathsf{Pkg}, \mathsf{Gap} \to \mathsf{NameCert},$$

two domains that traditionally appear unrelated meet inside a common classifier substrate as soon as both are licensed. A bridge is then itself a naming certificate: the source carrier is one of the two domains, the pattern is the other, and the classifier is the shared identification. Bridges are not metatheorems; they live inside the same framework as their endpoints, with $\mathsf{TheoryClosure} \times \mathsf{FormalStatus}$ status tracked on both axes.

The currently scoped bridges are:

Order ↔ Topology $\mathsf{StoneDuality}^{\uparrow}$ identifies Boolean algebras with Stone spaces.

Operator algebra ↔ Topology $\mathsf{GelfandDuality}^{\uparrow}$ identifies commutative C*-algebras with compact Hausdorff spaces.

Topological group ↔ Abelian group $\mathsf{PontryaginDuality}^{\uparrow}$ pairs LCA groups with their character duals.

Lie group ↔ Tensor category $\mathsf{TannakaKrein}^{\uparrow}$ recovers a compact group from its representation category.

Logic ↔ Computation $\mathsf{CurryHoward}^{\uparrow}$ identifies propositions with types and proofs with programs.

Differential ↔ Topological $\mathsf{RiemannHilbert}^{\uparrow}$ identifies regular holonomic D-modules with perverse sheaves.

Algebra ↔ Homological $\mathsf{KoszulDuality}^{\uparrow}$ identifies graded algebras with their Koszul duals.

Galois ↔ Automorphic $\mathsf{Langlands}^{\uparrow}$ matches Galois representations to automorphic representations through L-factors.

Symmetry ↔ Conservation $\mathsf{NoetherSymmetry}^{\uparrow}$ pairs continuous Lagrangian symmetries with conserved currents.

Symplectic ↔ Hilbert $\mathsf{GeomQuantization}^{\uparrow}$ assigns a Hilbert space to a symplectic manifold via prequantum data and polarisation.

Poisson ↔ Noncommutative $\mathsf{DeformQuantization}^{\uparrow}$ pairs a Poisson manifold with a formal star product whose first-order term is the bracket.

Differential ↔ Topological invariant $\mathsf{ChernWeil}^{\uparrow}$ expresses characteristic cohomology classes through curvature forms.

Spectral ↔ Topological $\mathsf{AtiyahSinger}^{\uparrow}$ matches the analytic index of an elliptic operator with a topological index in cohomology.

Complex-analytic ↔ Topological $\mathsf{HodgeBridge}^{\uparrow}$ decomposes de Rham cohomology of a Kähler manifold into harmonic (p, q)-forms.

Symplectic ↔ Complex-algebraic $\mathsf{MirrorSymmetry}^{\uparrow}$ pairs the A-model of one Calabi-Yau with the B-model of its mirror.

The bridge inventory is open: any two licensed naming certificates whose carriers identify under a stable classifier yield a new bridge namecert, and the unified-kernel discipline is what makes such bridges first-class citizens of the framework rather than miracles.

9467.18 Per-object research template

Every milestone is filed under the same ten-field template:

1. **Traditional target**: how N appears in mature mathematics.
2. **BEDC source**: which already-licensed BEDC objects generate it.
3. **Carrier choice**: predicate (or stream) over the source.
4. **Classifier choice**: equivalence relation, never `Quot`.
5. **Stability obligations**: refl/symm/trans, carrier transport, operation congruence, descent, completion / seal where relevant.
6. **Ledger**: what is hidden, projected, compressed, completed.
7. **Core theorem list**: the minimum set that must hold.
8. **Exactness predicate**: coverage, soundness, completeness, classification $\iff$.
9. **Standard bridge**: the conservative back-and-forth to the mature reference.
10. **Status**: a pair (c, v) where $c \in \mathsf{TheoryClosure}$ is the closure axis grade and $v \in \mathsf{FormalStatus}$ is the verification axis grade.

9467.19 Near-term phases

The near-term path is a small number of phase bundles. Each phase has a specific exit criterion and is the only sanctioned advancement direction during its tenure.

Phase Kernel-Checked Baseline L0–L5, including msame, hsame, Ext, Cont, Sig, Gap, NameCert, ExactGlobalizeBase. Exit criterion:

- Closure axis: kernel proof obligations are supplied through the finite-kernel proof DAG ending at $\mathsf{NameCert}$.
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ for every cited kernel target N.
- Bridge status: pending.

Phase Finite Data L6, $\mathsf{Bool}^{\uparrow}$ through $\mathsf{List}^{\uparrow}$, with the ten-field template completed for each. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{Bool}^{\uparrow}, \mathsf{Option}^{\uparrow}, \mathsf{Sum}^{\uparrow}, \mathsf{Prod}^{\uparrow}, \mathsf{List}^{\uparrow}\}$.
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Full Nat Arithmetic L7, $\mathsf{Nat}^{\uparrow}$ through $\mathsf{Rat}^{\uparrow}$, with induction, recursion, ordered-ring and ordered-field structure. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{Nat}^{\uparrow}, \mathsf{Add}^{\uparrow}, \mathsf{Mul}^{\uparrow}, \mathsf{Order}^{\uparrow}, \mathsf{Int}^{\uparrow}, \mathsf{Rat}^{\uparrow}\}$.
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Algebra Hierarchy L8 up to $\mathsf{Field}^{\uparrow}$. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{Monoid}^{\uparrow}, \mathsf{Group}^{\uparrow}, \mathsf{AbGroup}^{\uparrow}, \mathsf{Ring}^{\uparrow}, \mathsf{CommRing}^{\uparrow},$
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Real and Completion L10 items 1–4 of section 9467.12: $\mathsf{DyadicRatCore}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}$, $\mathsf{Real}^{\uparrow}$. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{DyadicRatCore}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}, \mathsf{Real}^{\uparrow}$
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Real Algebra and Geometry L10 items 5–7: $\mathsf{RealAlgOrder}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{RealAlgOrder}, \mathsf{Met}^{\uparrow}, \mathsf{S}^{1\uparrow}\}$.
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Compact and Continuous L10 items 8–11: $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{TotallyBounded}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{CompactMet}$
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Linear Algebra and Analysis Seed L8 vector-space layer plus L11 calculus seed: $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every N in $\{\mathsf{VecSpace}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{Mat}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Hilbert}^{\uparrow}\}$.

- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited Lean targets.
- Bridge status: pending.

Phase Category and Bridge L12 $\mathsf{Cat}^{\uparrow}$ and L14 bridge for already-checked objects. Exit criterion:

- Closure axis: $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ for every phase object, with $\mathsf{bridgedClosed}$ only for objects whose standard bridge is explicitly supplied.
- Verification axis: $\mathsf{FormalStatus}(N, \mathsf{auditClean})$ on the cited BEDC Lean targets and $\mathsf{FormalStatus}(N, \mathsf{bridgeC}$ on bridge targets where claimed.
- Bridge status: supplied for the named bridge deliverables and pending elsewhere.

The reviewer pipeline picks targets in service of the current open phase. Targets that advance a later phase while the current phase is open are deferred. The roadmap is maintained in place when a phase exits, without version annotations.

9467.20 Claims explicitly off the table

The discipline of this manuscript is two-axis. The following readings are refused:

1. BEDC has re-derived all of mature mathematics.
2. $\mathsf{Real}^{\uparrow}$ is complete.
3. $\mathsf{S}^{1\uparrow}$ is complete.
4. $\mathsf{Fold}^{\uparrow}$ is complete.
5. $\mathsf{Nat}^{\uparrow}$ at $\mathsf{scopedClosed}$ equals full natural-number theory.
6. $\mathsf{Add}^{\uparrow}$ at $\mathsf{scopedClosed}$ equals full addition.
7. Paper certificates equal machine-checked certificates.
8. Lean scaffolds equal verified implementations.
9. External standard models replace BEDC internal proof obligations.
10. Host equality, host Nat, or host functions are part of the BEDC public surface.
11. Machine verification is required for theoretical closure.
12. Without $\mathsf{theoremChecked}$ no closure grade above seed may be claimed.
13. A checked theorem-shape scaffold closes the corresponding object theory.
14. A scoped certificate closure is a full traditional-object reconstruction.
15. A bridge is closed merely because the BEDC-side certificate is checked.

The discipline of this roadmap is condensed in one line:

a derivation does not assume the mature object exists; it proves a name has earned the right to be used.

Part XIV

Appendices

Appendix A

Lean Implementation Scaffold

This appendix records the implementation-facing scaffold as an external artifact. It is not part of the BEDC object theory. It is included to keep the manuscript self-contained.

```
import BEDC.BaseReflection.Core
import BEDC.BaseReflection.Token
import BEDC.BaseReflection.Psame
import BEDC.BaseReflection.PackageReflection
import BEDC.BaseReflection.GeneratedSameSig
import BEDC.BaseReflection.ExactGlobalizeBase
import BEDC.BaseReflection.ExactGlobalizeBase.Fields
```

Appendix B

Verification Reporting Policy

This appendix records the reporting policy that a checked BEDC formalization report should follow. It is project-management material on the verification axis only, not part of the BEDC object theory; nothing in this appendix assigns a closure grade.

B.1 Required fields for a checked report

A checked report should record:

1. the exact proof-checker version;
2. the exact command used;
3. whether the implementation imports external libraries;
4. the list of theorem names successfully checked;
5. the list of admitted, postulated, or pending statements;
6. whether public APIs expose host equality, host natural numbers, host functions, or host lists as BEDC internals.

B.2 Verification status

The Lean library accompanying the paper contains the mathlib-free BEDC formalization; inline markers record whether each cited object is a checked theorem, a definition, or a structure-field reference. Lean target status, theorem-by-theorem, is rendered inline at each primary site via the `\leanchecked`, `\leansorryd`, `\leanstmt`, and `\leandef` markers. The summary block in the Lean Scaffold Contract chapter aggregates the five base-reflection targets at-a-glance. The audit `python3 lean4/scripts/bedc_ci.py audit` enforces that every `\leanchecked{X}` target X resolves to a real declaration in `lean4/BEDC/`. The axiom audit reports zero `axiom` declarations in `lean4/BEDC/`.

The external Rule 110 substrate witness is reported in section B.3. It is an independent witness for the verification state, not the canonical statement of BEDC theory.

B.3 Rule 110 substrate witness

The substrate witness has two trust paths. Tier A is the cyclic-tag witness identified by tag `rule110-v2.0-fkernel-tier-a`; its trust base is an ANSI C compiler, approximately 250 lines of cyclic-tag evaluator C, and the manifest text. Its ship standard is that the cyclic-tag evaluator accepts the manifest assertions for the FKernel and GroundCompiler surface, with green CI, 0 axioms, and a checked cross-report. Tier B is the Rule 110 direct witness identified by tag `rule110-v3.0-fkernel-tier-b`; its trust base is an ANSI C compiler, approximately 50 lines of Rule 110 cellular-automaton evaluator C, and direct-carrier manifest text. Its ship standard is that each FKernel `.enum.ct` direct-carrier row round-trips through Rule 110 evolution and decoding to the corresponding manifest input. The Tier B direct-carrier claim is deliberately narrower than a full Cook packet simulation of arbitrary cyclic-tag execution; the diagnostic Cook-packet layer is tracked as executable evidence, not as the direct-carrier ship gate. The cited substrate literature is Matthew Cook, "Universality in Elementary Cellular Automata," *Complex Systems* 15(1), 2004; Martinez–McIntosh–Seck Tuoh Mora–Chapa-Vergara, "Determining a regular language by glider-based structures called phases f_{i_1} in Rule 110," preprint 0706.3348, 2007; Martinez–Adamatzky–Chen–Chua, "On Soliton Collisions between Localizations in Complex Elementary Cellular Automata: Rules 54 and 110 and Beyond," *Complex Systems* 21.2.2, 2012; and Matthew Cook, "A Concrete View of Rule 110 Computation," preprint 0906.3248, 2009.

The ontology is finite-witness ontology. A BEDC substrate claim is not a claim over an unbounded semantic universe supplied by the Lean kernel. It is the finite assertion family physically present under `rule110/manifests/<module>/*.{enum,algo}.ct`, together with the corresponding direct-carrier rows where Tier B supplies them. Thus a Lean formula written with a universal quantifier is read, at this substrate boundary, as a compact human-readable notation for all concrete instances listed in the manifest family. The binding substrate objects are the manifest assertions themselves: their module directory, assertion name, input field, expected field, and evaluator outcome.

The Lean formalization remains meaningful but sits outside the BEDC substrate trust path. The Lean kernel is used for paper bookkeeping, human-readable shorthand, and the verification-axis markers described in Appendix B. A marker such as `\leanchecked{X}` still records a real audit obligation: `python3 lean4/scripts/bedc_ci.py audit` checks that X resolves to an actual declaration under `lean4/BEDC/`. The truth of the BEDC substrate assertion, however, is carried by the manifest witness and its evaluator report, not by the Lean kernel.

This witness is another audit route on the verification axis. It does not create a closure grade and it does not alter the list of closure or verification levels. The manuscript's closure claim is still read from the relevant `closurestatus` block, as in section 9203.6; the substrate witness records an independently checkable verification report. The two axes compose only through VerifiedGate as defined in Definition 9203.12.

The substrate witness covers 13 FKernel modules and GroundCompiler: Mark, Hist, Ext, Sig with SigRel and sameSig, Cont, Bundle, Unary, Ask, ExternalBinary, Gap, Package, NameCert, and Settled. It also carries the Beyond-FKernel appendix modules `circle_up`, `fold_up`, `meta_cic`, and `topology_up`. The manifest surface includes 22 `.algo.ct` Cook-packet diagnostic files and at least 502 counted manifest or semantic cases, with 0 axioms and green CI at the cited tags. The Tier B direct-carrier subset covers 25 FKernel `.r110.ct` manifests: Mark 4, Hist 5, Ext 1, Sig 2, Cont 1, Bundle 2, Unary 1, Ask 1, ExternalBinary 1, Gap 1, Package 1, NameCert 1, Settled 1, and GroundCompiler 3.

B.4 Relation to theoretical closure

A checked report does not assign theoretical closure. It reports the formal verification grade of claims whose closure grade is supplied by the chapter's `closurestatus` block (chapter 9203). A closurestatus block at $\mathsf{TheoryGate}(N, \mathsf{scopedClosed})$ does not require any verification report; conversely, a clean verification report cannot raise a closure grade. The two artefacts compose only via $\mathsf{VerifiedGate}$ of Definition 9203.12, and their contents are interrogated separately.